

ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ



Πτυχιακή Εργασία

Εφαρμογή webGIS για την υποστήριξη
και παροχή πληροφοριών στην αεροναυτιλία.

Γιαννιός Αντώνιος

Π2013153

Επιβλέπων: Στεφανιδάκης Μιχαήλ

20 Αυγούστου 2021

Επιβλέπων

Μιχαήλ Στεφανιδάκης, Αναπληρωτής Καθηγητής,
Ιόνιο Πανεπιστήμιο

Τριμελής Επιτροπή

Μιχαήλ Στεφανιδάκης, Αναπληρωτής Καθηγητής,
Ιόνιο Πανεπιστήμιο

Θεόδωρος Ανδρόνικος, Αναπληρωτής Καθηγητής,
Ιόνιο Πανεπιστήμιο

Κωνσταντίνος Οικονόμου, Καθηγητής,
Ιόνιο Πανεπιστήμιο

Στη σύζυγό μου Αλίκη και στα παιδιά μου, Μαρία Άννα και Ευαγγελία.

Περίληψη

Η πτυχιακή εργασία με θέμα «Εφαρμογή webGIS για την υποστήριξη και παροχή πληροφοριών στην αεροναυτιλία» είναι μια προσπάθεια υλοποίησης εφαρμογής η οποία έχει σκοπό να αποτελέσει συμβουλευτικό και βοηθητικό μέσο στους υπαλλήλους των Πύργων Ελέγχου που παρέχουν έργο Πληροφοριών Πτήσεων Αεροδρομίου (Π.Π.Α) - Aerodrome Information Service (A.F.I.S). Η εφαρμογή βοηθά τον χρήστη να οπτικοποιήσει την κυκλοφορία των αεροσκαφών ενός αεροδρομίου, πάνω σε θεματικό αεροναυτικό χάρτη.

Χρήστες της εφαρμογής είναι οι υπάλληλοι της Υπηρεσίας Πολιτικής Αεροπορίας (Υ.Π.Α) που υπηρετούν στους Πύργους Ελέγχου μη ελεγχόμενων αεροδρομίων που δεν διαθέτουν εικόνα από σύστημα radar. Στα μικρά και μη ελεγχόμενα αεροδρόμια, η έλλειψη εικόνας radar, αυξάνει την προσπάθεια που πρέπει να καταβάλει ο ελεγκτής στον Πύργο Ελέγχου να διαχειριστεί την κυκλοφορία των αεροσκαφών, ειδικά όταν αυτή είναι αυξημένη. Συνήθως ο ελεγκτής σημειώνει σε ένα πρόχειρο χαρτί την αναφορά του πιλότου ενός αεροσκάφους για την θέση στην οποία βρίσκεται το αεροσκάφος και την πορεία που πρόκειται να ακολουθήσει. Η τακτική αυτή είναι ανεπαρκής όταν τα αεροσκάφη είναι αρκετά σε αριθμό, αυξάνοντας το άγχος του ελεγκτή και την πιθανότητα να κάνει λάθος στη διαχείριση της κυκλοφορίας. Η εφαρμογή μας προσπαθεί να δώσει λύση στο πρόβλημα αυτό, με την οπτικοποίηση της κίνησης των αεροσκαφών που κινούνται από και προς ένα αεροδρόμιο, πάνω σε θεματικό αεροπορικό χάρτη. Εκτός από την οπτικοποίηση της κυκλοφορίας η εφαρμογή μας προσφέρει και άλλες χρήσιμες λειτουργίες όπως η εύρεση των μετεωρολογικών δελτίων METARs και των αεροναυτικών αγγελιών NOTAMs ενός αεροδρομίου, την εύρεση πληροφοριών ενός αεροδρομίου που είναι δημοσιευμένες στο εγχειρίδιο Aeronautical Information Publication (AIP), την απεικόνιση των αεροδιαδρόμων, των waypoints, των τερματικών περιοχών MTMAs κ.α, ενός εναέριου χώρου Flight Information Region (FIR) μιας χώρας, την διαχείριση σχεδίων πτήσης και εμφάνιση της

προεπισκόπησης τους πάνω στον χάρτη και προσομοίωση της πτήσης των αεροσκαφών και της κίνησής τους σύμφωνα με τα σχέδια πτήσης τους.

Η εφαρμογή ανήκει στην κατηγορία των web εφαρμογών και υλοποιείται με τη χρήση μιας βάσης δεδομένων postgresSQL η οποία έχει καταχωρημένα στοιχεία που αφορούν την αεροναυτιλία, π.χ αεροδρόμια, ραδιοβοηθήματα, αεροδιαδρόμους, waypoints,τερματικές περιοχές MTMAs, πληροφορίες αεροδρομίων κ.α. Επίσης η εφαρμογή μας χρησιμοποιεί έναν διακομιστή (Geoserver) ο οποίος δημιουργεί τα απαραίτητα layers για την απεικόνιση του θεματικού αεροπορικού χάρτη. Ο χάρτης μαζί με τις λειτουργίες του, υλοποιείται στο front end μέρος της εφαρμογής με τη χρήση της βιβλιοθήκης openlayers και της επέκτασής της ol-ext σε γλώσσα javascript. Για την διασύνδεση του front end μέρους της εφαρμογής με τη βάση δεδομένων και το web χρησιμοποιείται ένας διακομιστής node.js.

Η εφαρμογή μας χρησιμοποιεί το γεωγραφικό σύστημα αναφοράς συντεταγμένων WGS 1984, EPSG:4326 και οι πληροφορίες που χρησιμοποιεί η βάση δεδομένων αντλήθηκαν από το εγχειρίδιο AIP Greece October 2020 και AIP Italy November 2020. Η σχεδίαση των αεροδιαδρόμων των FIR Ελλάδας και Ιταλίας έγινε με μη αυτοματοποιημένο τρόπο αντιγράφοντας τους από IFR χάρτες που περιλαμβάνονται στα παραπάνω εγχειρίδια.

Abstract

This thesis titled “Application webGIS for Aerodrome Flight Information Service (A.F.I.S)” is an attempt to provide an advisory and auxiliary mean for the employees of Hellenic Civil Aviation Authority who work at the control towers of A.F.I.S aerodromes. This application helps users to visualize air traffic onto an aeronautical map.

Users of the application are considered the employees of Hellenic Civil Aviation Authority, who work at the control towers of uncontrolled aerodromes which do not have access to radar data. In uncontrolled aerodromes, lack of radar data increases the effort of air traffic controller to handle air traffic, especially when this traffic is increased. The air traffic controller usually records on paper the pilot’s report which refers to current aircraft’s position and the route that the pilot intends to follow. This method is insufficient, when aircraft are too many, resulting into elevated air traffic controller’s stress and thus increasing the possibility for a mistake in air traffic management process. Our application aims to provide a solution to this problem, by visualizing air traffic (both inbound and outbound traffic to/from an airport) onto an aeronautical map. Furthermore, the application provides some other useful features like retrieving METARs and NOTAMs of an aerodrome, retrieving information of an aerodrome which is published in Aeronautical Information Publication (AIP), visualizing of airways, waypoints, Terminal Manoeuvring Areas (TMAs) and Flight Information Region (FIR) of a country, handling and previewing of flight plans and emulated projection of those flight paths in accordance with their flight plans.

The application is a web based application and it is implemented by using a PostgreSQL database through which the aeronautical data are retrieved for example aerodromes, nav aids, airways, waypoints, terminal areas (TMAs), aerodromes’ information etc. The application uses a server (Geoserver) which constructs the necessary layers for visualizing the aeronautical map. In the front end part of the application the map with its features is

implemented by using openlayers library and its extension ol-ext in javascript language. A node.js server is used in order to connect the front end part of the application to the database and the web.

The application uses the geodetic coordinates reference system WGS 1984, EPSG:4326 and the data used by the database was retrieved by AIP Greece October 2020 and AIP Italy November 2020 publications. The construction of airways data has taken place in non automatic way, by copying them manually from IFR maps which are included in the aforementioned publications.

Πρόλογος και Ευχαριστίες

Ευχαριστώ πρώτα από όλα τους καθηγητές του τμήματος Πληροφορικής του Ιονίου Πανεπιστημίου για όλες τις γνώσεις που αποκόμισα, την οικογένειά μου για την υπομονή και τη στήριξη στην προσπάθειά μου, τους φίλους και συνεργάτες μου της Υπηρεσίας Πολιτικής Αεροπορίας του αερολιμένα Κέρκυρας, για τις πολύτιμες ιδέες και γνώμες πάνω στις λειτουργίες που υλοποιεί η εφαρμογή αυτή.

Κέρκυρα, Αύγουστος 2021

Αντώνιος Γιαννιός

Περιεχόμενα

A' Εισαγωγή	1
B' Αεροναυτιλία και Μετεωρολογία	3
B'.1 ΑΕΡΟΝΑΥΤΙΛΙΑ.....	3
B'.1.1 Γεωγραφικές Συντεταγμένες.....	3
B'.1.2 World Geodetic System (WGS).....	4
B'.1.3 Coordinated Universal Time (UTC)	5
B'.1.4 Ενδείκτες Τοποθεσίας.....	5
B'.1.5 Ραδιοβοηθήματα	6
B'.1.6 Εγχειρίδιο A.I.P	7
B'.1.7 Waypoints	7
B'.1.8 Αεροδιάδρομοι.....	8
B'.1.9 Βαρομετρική Πίεση και Ύψος Πτήσης	8
B'.1.10 Επίπεδο Πτήσης	9
B'.1.11 Flight Information Region (FIR)	10
B'.1.12 Περιοχές CTR, TMA, ATZ	11
B'.1.13 Αεροδρόμιο	12
B'.1.14 Radar	12
B'.1.15 Κωδικός Αριθμός Squawk.....	13
B'.1.16 Κανόνες Πτήσης	14
B'.1.17 Σχέδιο Πτήσης	14

B'.1.18 Aeronautical Fixed Telecommunication Network.....	15
B'.1.19 NOTAM.....	16
B'.2 ΜΕΤΕΩΡΟΛΟΓΙΑ	18
B'.2.1 METAR.....	18
Γ' Λογισμικό της Εφαρμογής	21
Γ'.1 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ POSTGRESQL	21
Γ'.1.1 Επέκταση PostGIS και Εργαλείο Διαχείρισης pgAdmin	21
Γ'.2 ΔΙΑΚΟΜΙΣΤΗΣ GEOSERVER.....	23
Γ'.2.1 Διακομιστές χαρτών	23
Γ'.2.2 WMS	23
Γ'.2.3 Γραφικό Περιβάλλον Διαχείρισης	24
Γ'.2.4 Τρόπος Λειτουργίας.....	25
Γ'.2.5 Style Layer Descriptor	25
Γ'.2.6 Δημιουργία Layer	25
Γ'.3 ΔΙΑΚΟΜΙΣΤΗΣ NODE.JS.....	27
Γ'.3.1 Γενικά.....	27
Γ'.3.2 Βιβλιοθήκες του Node.js.....	27
Γ'.4 FRONT END.....	28
Δ' Αρχιτεκτονική της Εφαρμογής	31
Δ'.1 ΕΦΑΡΜΟΓΕΣ G.I.S	31
Δ'.2 ΓΕΝΙΚΗ ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ.....	32
Δ'.3 BLOCK ΔΙΑΓΡΑΜΜΑ ΚΑΙ ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ ΔΕΔΟΜΕΝΩΝ	32
Δ'.4 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ POSTGRESQL	35
Δ'.4.1 Πίνακας Waypoints_Layer	35
Δ'.4.2 Πίνακας Nav aids_Layer	36
Δ'.4.3 Πίνακας Aerodromes_Layer	37
Δ'.4.4 Πίνακας Airways	37
Δ'.4.5 Πίνακας Airways_Layer	38
Δ'.4.6 Πίνακας VFR_Layer	39
Δ'.4.7 Πίνακες LGC_Layer, LGD_Layer και LGM_Layer.....	40

Δ'.4.8	Πίνακας MTMA_Layer.....	41
Δ'.4.9	Πίνακας CTR_Layer	41
Δ'.4.10	Πίνακας FPL	42
Δ'.4.11	Πίνακας FLIGHTS	45
Δ'.4.12	Πίνακας AIP	46
Δ'.5	ΔΙΑΚΟΜΙΣΤΗΣ GEOSERVER.....	47
Δ'.5.1	Layers	47
Δ'.5.2	Παραμετροποίηση S.L.D	47
Δ'.6	ΔΙΑΚΟΜΙΣΤΗΣ NODE.JS	52
Δ'.6.1	Δημιουργία του Διακομιστή	52
Δ'.6.2	Route Path /coordsToAerodrome	53
Δ'.6.3	Route Path /coordinates	54
Δ'.6.4	Route Path /info	55
Δ'.6.5	Route Path /aerodrome.....	56
Δ'.6.6	Route Path /airway	56
Δ'.6.7	Route Path /FPL	57
Δ'.6.8	Route Path /Flights.....	58
Δ'.6.9	Route Path /getRoute	59
Δ'.6.10	Route Path /Reset.....	59
Δ'.6.11	Route Path /EndOfFlight	60
Δ'.6.12	Route Path /DeleteGPXFile	60
Δ'.6.13	Route Path /GetMyFlightPlansContent.....	60
Δ'.6.14	Route Path /ReadFPL.....	61
Δ'.6.15	Route Path /WriteFPL.....	62
Δ'.6.16	Route Path /WriteGPXFile	62
Δ'.6.17	Route Path /metar.....	63
Δ'.6.18	Route Path /notam.....	64
Δ'.6.19	Route Path /STAR.....	67
Δ'.6.20	Route Path /SID	68
Δ'.6.21	Route Path /GetSIDSTARContent.....	68
Δ'.7	FRONT END	69
Δ'.7.1	Navigation Bar	69
Δ'.7.2	Βασικός Χάρτης.....	70

Δ'.7.3	Layers.....	73
Δ'.7.4	Κουμπιά	74
Δ'.7.4.1	ΚΟΥΜΠΙΑ V, I, T, C, L, D, M, R.....	75
Δ'.7.4.2	ΚΟΥΜΠΙΑ F, S	77
Δ'.7.4.3	ΚΟΥΜΠΙΑ r, +, Z, B	79
Δ'.7.4.4	ΚΟΥΜΠΙΑ H, A, (Full Screen)	83
Δ'.7.5	Υπόλοιπες Λειτουργίες	84
Δ'.7.6	Μηχανισμός Μετακίνησης Φορμών	85
Δ'.7.7	Φόρμα Επιλογής.....	86
Δ'.7.8	Φόρμα AIP Info	88
Δ'.7.9	Φόρμα METAR.....	89
Δ'.7.10	Φόρμα NOTAM.....	90
Δ'.7.11	Φόρμα IMAGE	91
Δ'.7.12	Εμφάνιση Ώρας UTC και Χρονολογίας UTC.....	92
Δ'.7.13	Φόρμα FPL.....	93
Δ'.7.13.1	ΕΛΕΓΧΟΣ ΤΙΜΩΝ.....	93
Δ'.7.13.2	ΛΕΙΤΟΥΡΓΙΑ ΠΡΟΕΠΙΣΚΟΠΗΣΗΣ	95
Δ'.7.13.3	ΥΠΟΛΟΙΠΕΣ ΛΕΙΤΟΥΡΓΙΕΣ	97
Δ'.7.13.4	ΚΟΥΜΠΙΑ CLEAR FPL ΚΑΙ SUBMIT FPL.....	97
Δ'.7.14	Φόρμα Load FPL	99
Δ'.7.15	Φόρμα Save FPL	100
Δ'.7.16	Φόρμα Electronic Strip Bay	101
Δ'.7.17	Λειτουργία Απεικόνισης Πτήσης.....	103
Δ'.7.17.1	ΔΗΜΙΟΥΡΓΙΑ GPXFILE.....	103
Δ'.7.17.2	ΔΗΜΙΟΥΡΓΙΑ LAYER ΑΕΡΟΣΚΑΦΟΥΣ	104
Δ'.7.17.3	ΔΗΜΙΟΥΡΓΙΑ LAYER ΔΙΑΔΡΟΜΗΣ.....	105
Δ'.7.17.4	ΔΗΜΙΟΥΡΓΙΑ FEATURES	105
Δ'.7.17.5	ΚΙΝΗΣΗ FEATURES.....	104
Δ'.8	ΕΓΚΑΤΑΣΤΑΣΗ ΛΟΓΙΣΜΙΚΟΥ	107
Δ'.8.1	Βάση Δεδομένων PostgreSQL	107
Δ'.8.2	Διακομιστής Geoserver	108
Δ'.8.3	Διακομιστής Node.js	109
Δ'.8.4	Front End.....	110
Δ'.9	ΔΕΔΟΜΕΝΑ	110

Περιεχόμενα	xi
Ε' Συμπεράσματα – Μελλοντικές Επεκτάσεις	111
Βιβλιογραφία - Δικτυογραφία	113
Συντμήσεις	117
Γλωσσάρι Ξενικών Όρων	119
Ευρετήριο	123

Κατάλογος Εικόνων

Γ'.1 Αρχική σελίδα του εργαλείου pgAdmin	22
Γ'.2 Αρχική σελίδα του γραφικού περιβάλλοντος διαχείρισης του Geoserver ..	24
Γ'.3 Καθορισμός του data source, το οποίο θα έχει το layer.....	26
Δ'.1 Μπλοκ Διάγραμμα	33
Δ'.2 Διάγραμμα Ροής Δεδομένων	34
Δ'.3 Πρότυπο έντυπο κατά ICAO ενός σχεδίου πτήσης	43
Δ'.4 Παραμετροποίηση ενός style με το Style Editor	48
Δ'.5 Πρώτος βασικός χάρτης.....	71
Δ'.6 Δεύτερος βασικός χάρτης	71
Δ'.7 Δημιουργία του πρώτου χάρτη στο maptiler	72
Δ'.8 Κουμπιά V, I, T, C, L, D, M, R 47	75
Δ'.9 MTMA, CTR, VFR Layers.....	76
Δ'.10 LGC, LGD, LGM, LGR Layers.....	77
Δ'.11 Φόρμα FPL	78
Δ'.12 Φόρμα Electronic Strip Bay.....	79
Δ'.13 Κουμπιά r και +.....	80
Δ'.14 Κουμπί Z	80
Δ'.15 Αρχική οθόνη της εφαρμογής webGIS.....	84
Δ'.16 Ενδείκτης κλίμακας του χάρτη	85
Δ'.17 Φόρμα επιλογής	87
Δ'.18 Φόρμα AIP Info	89

Δ'.19 Φόρμα METAR	90
Δ'.20 Φόρμα NOTAM.....	91
Δ'.21 Φόρμα IMAGE	92
Δ'.22 Έλεγχος τιμών του πεδίου της διαδρομής	95
Δ'.23 Προεπισκόπηση του σχεδίου πτήσης.....	96
Δ'.24 Φόρμα Load FPL	100
Δ'.25 Φόρμα Save FPL.....	101
Δ'.26 Απεικόνιση πτήσης στο χάρτη	107

Κεφάλαιο Α΄

Εισαγωγή

ΤΟ θέμα της παρούσας πτυχιακής είναι η υποστήριξη και παροχή πληροφοριών στην αεροναυτιλία μέσω της εφαρμογής webGIS. Η εφαρμογή υλοποιήθηκε από τον Νοέμβριο του 2020 έως τον Ιούνιο του 2021.

Η εφαρμογή υλοποιήθηκε στα πλαίσια της προσπάθειας κάλυψης της ανάγκης που υπάρχει στα μικρά αεροδρόμια της χώρας, τα οποία παρέχουν υπηρεσία Aerodrome Flight Information Service (A.F.I.S) και δεν διαθέτουν εικόνα της κυκλοφορίας των αεροσκαφών. Στα συγκεκριμένα αεροδρόμια, οι υπάλληλοι που εκτελούν υπηρεσία στον Πύργο Ελέγχου δεν έχουν εικόνα από κάποιο ραντάρ ώστε να γνωρίζουν την ακριβή θέση των αεροσκαφών που βρίσκονται στην περιοχή ευθύνης τους. Η αποτύπωση της θέσης των αεροσκαφών γίνεται μόνο με αναφορά από τους πιλότους μέσω ασυρμάτου. Όταν η κυκλοφορία αποτελείται από λίγα αεροσκάφη, αυτό δεν αποτελεί πρόβλημα καθώς ο ελεγκτής σημειώνει τη θέση των αεροσκαφών πάνω σε χαρτί. Όταν όμως ο αριθμός των αεροσκαφών είναι μεγάλος προκύπτει η ανάγκη αποτύπωσης της θέσης τους με τέτοιο τρόπο, ώστε να μην αποτελεί πρόβλημα για τον ελεγκτή εναέριας κυκλοφορίας. Η οπτικοποίηση μέσω της εφαρμογής webGIS, έρχεται να δώσει λύση έστω ως συμβουλευτικό μέσο.

Η εφαρμογή δεν αποτελεί σε καμία περίπτωση αντικατάσταση ενός ραντάρ, αλλά ένα συμβουλευτικό μέσο στο οποίο ο ελεγκτής θα μπορεί να βλέπει τις θέσεις των αεροσκαφών στην οθόνη αντί να τις σημειώνει στο χαρτί.

Επίσης η εφαρμογή διαθέτει και άλλες χρήσιμες λειτουργικότητες οι οποίες βοηθούν σημαντικά τον ελεγκτή στο έργο του, όπως η εύρεση του παρόντα καιρού αεροδρομίου (METAR), η εύρεση των ενεργών αεροναυτικών αγγελιών (NO.T.AMs) κ.α.

Η εργασία αυτή αποτέλεσε μια καλή αφορμή στο να εμβαθύνουμε στη μελέτη του τρόπου λειτουργίας αντίστοιχων εφαρμογών Συστημάτων Γεωγραφικών Πληροφοριών (Σ.Γ.Π.), Geographic Information Systems (G.I.S.) καθώς και στις αντίστοιχες βιβλιοθήκες λογισμικού με τις οποίες αυτές υλοποιούνται. Επίσης η εργασία αυτή μας επέτρεψε να ασχοληθούμε περισσότερο με τον τρόπο με τον οποίο μια βάση δεδομένων λειτουργεί σε συνεργασία με μια web εφαρμογή.

Η εφαρμογή webGIS ανήκει στην κατηγορία των web εφαρμογών και αναπτύχθηκε σε γλώσσα javascript. Αποτελείται από το front end κομμάτι στο οποίο χρησιμοποιείται η βιβλιοθήκη openlayers μαζί με την επέκτασή της, την βιβλιοθήκη ol-ext, για την δημιουργία των αεροναυτικών θεματικών χαρτών.

Η εφαρμογή αποτελείται επίσης και από το back end μέρος, στο οποίο χρησιμοποιείται η βάση δεδομένων postgresQL, στην οποία αποθηκεύονται όλα τα δεδομένα της εφαρμογής και με την οποία επικοινωνεί ο Geoserver για να πάρει τα δεδομένα για την κατασκευή των layers. Με την βάση δεδομένων επικοινωνεί το front end μέρος μέσω του node.js server.

Στο δεύτερο κεφάλαιο θα κάνουμε μια σύντομη εισαγωγή στην αεροναυτιλία και στη μετεωρολογία, όπου θα αναλύσουμε βασικές έννοιες που χρησιμοποιούνται στην εργασία αυτή.

Στο τρίτο κεφάλαιο θα παρουσιάσουμε το λογισμικό που χρησιμοποιήθηκε για την υλοποίηση της εφαρμογής.

Στο τέταρτο κεφάλαιο θα αναλύσουμε την αρχιτεκτονική της εφαρμογής, παρουσιάζοντας το μπλοκ διάγραμμα και το διάγραμμα ροής δεδομένων της εφαρμογής. Στη συνέχεια θα αναλύσουμε τα αντίστοιχα μέρη από τα οποία αποτελείται και θα περιγράψουμε όσο το δυνατόν χωρίς πολλές λεπτομέρειες, ποιες ενέργειες κάνει ο κώδικας σε κάθε μέρος. Ειδικότερα, θα δούμε από ποιους πίνακες αποτελείται η βάση δεδομένων και πώς έγινε η προετοιμασία των δεδομένων αεροναυτιλίας, θα παρουσιάσουμε τα layers που κατασκευάζει ο διακομιστής Geoserver καθώς και παραδείγματα από τα styles που χρησιμοποιούμε για να «ντύσουμε» τα συγκεκριμένα layers, θα αναλύσουμε τα route paths που χρησιμοποιούμε στον διακομιστή node.js και τα responses που λαμβάνει από αυτόν η εφαρμογή, θα παρουσιάσουμε τις φόρμες και τα κουμπιά του front end καθώς και τις λειτουργικότητες που επιτελούνται στην εφαρμογή. Τέλος θα αναφερθούμε στην διαδικασία που ακολουθήθηκε για την εγκατάσταση του λογισμικού της εφαρμογής.

Στο πέμπτο κεφάλαιο θα αναφερθούμε στα συμπεράσματα έπειτα από την υλοποίηση της εφαρμογής, στις δυσκολίες που συναντήσαμε καθώς και στις μελλοντικές επεκτάσεις που μπορούν να γίνουν σε αυτή.

Κεφάλαιο Β΄

Αεροναυτιλία και Μετεωρολογία

ΣΤΟ κεφάλαιο αυτό θα κάνουμε μια σύντομη εισαγωγή στην αεροναυτιλία και στη μετεωρολογία, ώστε να μπορέσει ο αναγνώστης να καταλάβει χρήσιμες και βασικές έννοιες σχετικές με το αεροπορικό έργο.

Β΄.1 Αεροναυτιλία

Κρίναμε σκόπιμο να αποφευχθούν περιττές αναφορές σε όρους που δεν είναι σχετικοί με τον σκοπό της εργασίας αυτής, καθώς υπάρχουν πολλοί περισσότεροι όροι και διαδικασίες, οι οποίες όμως δεν μας ενδιαφέρουν. Επίσης κάποιες διαδικασίες και όροι απλουστεύονται στην εξήγηση, παρότι είναι στην πραγματικότητα πιο σύνθετοι, χάριν συντομίας και ευκολίας στην κατανόησή τους.

Η αεροναυτιλία είναι η επιστήμη που περιλαμβάνει τις μεθόδους και διαδικασίες που καθορίζουν τον σχεδιασμό και έλεγχο της πορείας ενός αεροσκάφους από ένα μέρος σε ένα άλλο, την εύρεση της θέσης του, χωρίς να παραβιαστούν οι νόμοι που αφορούν την αεροπορία και χωρίς να τεθεί σε κίνδυνο η ασφάλεια των πτήσεων. Η εύρεση της θέσης ενός αεροσκάφους αναφέρεται στον ασφαλή προσδιορισμό του στίγματός του, δηλαδή του γεωγραφικού του μήκους και πλάτους.

Γενικότερα, μπορούμε να πούμε πως η αεροναυτιλία αναφέρεται στις γνώσεις και στις μεθόδους που πρέπει να ακολουθηθούν ώστε να έχουμε μια ασφαλή πτήση.

B'.1.1 Γεωγραφικές Συντεταγμένες

Η θέση ενός αεροσκάφους προσδιορίζεται από δύο συντεταγμένες, το γεωγραφικό μήκος (longitude) και γεωγραφικό πλάτος (latitude). Για να βρούμε αυτές τις συντεταγμένες προβάλλουμε τη θέση του αεροσκάφους πάνω στην επιφάνεια της γης.

Γεωγραφικό πλάτος (latitude) (ϕ) ενός σημείου που βρίσκεται στην επιφάνεια της γης είναι η γωνία που σχηματίζει η κατακόρυφος του τόπου με το επίπεδο του ισημερινού. Το γεωγραφικό πλάτος χαρακτηρίζεται Βόρειο B (North N) ή Νότιο N (South S) ανάλογα σε ποιο ημισφαίριο βρίσκεται το συγκεκριμένο σημείο. [1]

Γεωγραφικό μήκος (longitude) (λ) ενός σημείου στην επιφάνεια της γης είναι η στερεή γωνία που σχηματίζεται από το επίπεδο του μεσημβρινού που διέρχεται από το εν λόγω σημείο με το επίπεδο του πρώτου μεσημβρινού. Το γεωγραφικό μήκος χαρακτηρίζεται Ανατολικό A (East E) ή Δυτικό Δ (West W) ανάλογα σε ποιο ημισφαίριο βρίσκεται το σημείο. [1]

Και τα δύο, μετρούνται σε μοίρες και υποδιαιρέσεις αυτών, πρώτα και δεύτερα λεπτά από $0^\circ - 90^\circ$ B ή $0^\circ - 90^\circ$ N (αρχής γενομένης της μέτρησης από τον ισημερινό του οποίου το γεωγραφικό πλάτος είναι 0°) και από $0^\circ - 180^\circ$ A ή $0^\circ - 180^\circ$ Δ (αρχής γενομένης της μέτρησης από τον πρώτο μεσημβρινό με γεωγραφικό μήκος 0°). [1]

Η γη διαιρείται σε 360 μεσημβρινούς και 180 παραλλήλους. Δίνοντας ένα παράδειγμα, οι συντεταγμένες ενός σημείου πάνω στον χάρτη, είναι $N30^\circ15'08''$ $E025^\circ06'24''$. Δηλαδή το γεωγραφικό του πλάτος είναι 30 μοίρες, 15 λεπτά και 8 δευτερόλεπτα βόρεια του ισημερινού και το γεωγραφικό του μήκος είναι 25 μοίρες, 6 λεπτά και 24 δευτερόλεπτα ανατολικά του πρώτου μεσημβρινού.

Χρήσιμη παρατήρηση είναι ότι 1 λεπτό γεωγραφικού πλάτους αντιστοιχεί σε απόσταση 1852 μέτρων, δηλαδή 1 ναυτικού μιλίου (nm). Όμως αυτό δεν ισχύει για το 1 λεπτό γεωγραφικού μήκους, εξ αιτίας του ελλειψοειδούς σχήματος της Γης.

B'.1.2 World Geodetic System (WGS)

Στην Γεωδαισία, το ελλειψοειδές αναφοράς (reference ellipsoid) είναι μία μαθηματικώς οριζόμενη επιφάνεια που προσεγγίζει το γεωειδές (geoid), δηλαδή το αληθινό σχήμα της Γης.

Ένα χερσαίο σύστημα αναφοράς το οποίο καλείται γεωδαιτικό σύστημα ή γεωδαιτικό datum, μπορεί να είναι παγκόσμιο, δηλαδή να αντιπροσωπεύει όλη τη Γη ή να είναι τοπικό, δηλαδή να αντιπροσωπεύει ένα ελλειψοειδές ορισμένου μέρους της Γης. [2]

Για παράδειγμα, μπορούμε να αναφέρουμε το γεωδαιτικό datum OSGB36 το οποίο αντιπροσωπεύει την περιοχή της Μεγάλης Βρετανίας και το οποίο είναι τοπικό.

Ένα παράδειγμα παγκόσμιου γεωδαιτικού datum είναι το WGS (World Geodetic System). Το WGS αποτελεί το πρότυπο datum που χρησιμοποιείται στη Χαρτογραφία (cartography), στην Γεωδαισία (Geodesy) και στην δορυφορική πλοήγηση (satellite navigation) περιλαμβανομένου και του Global Positioning System (GPS). [3]

Η νεότερη έκδοση του WGS είναι το WGS 84, η οποία εκδόθηκε από τις Η.Π.Α το 1984 και αναθεωρήθηκε το 2014. [3]

Το EPSG (EPSG Geodetic Parameter Dataset) είναι ένα δημόσιο μητρώο από γεωδαιτικά datums, γεωδαιτικά συστήματα αναφοράς ή συστήματα αναφοράς συντεταγμένων, ελλειψοειδών, μετασχηματισμούς συντεταγμένων και σχετιζόμενες μονάδες μέτρησης. Το κάθε ένα, προσδιορίζεται με έναν κωδικό (1024-32767). [4]

Το WGS 84 καλείται επίσης και ως WGS 1984, EPSG:4326.

Άλλο γνωστό dataset είναι το EPSG 3857 ή αλλιώς Web Mercator projection. Χρησιμοποιείται από πολλά Web εργαλεία χαρτογράφησης όπως το Google Maps και το OpenStreetMap.[4]

Υπάρχουν ακόμη EPSG τα οποία είναι τοπικά και χρησιμοποιούνται στη χαρτογράφηση συγκεκριμένων περιοχών – χωρών. Για παράδειγμα το EPSG 31254 το οποίο χρησιμοποιείται για την χώρα της Αυστρίας. [5]

Στην αεροναυτιλία η ταχύτητα ενός αεροσκάφους μετριέται σε κόμβους (knots). Ένας κόμβος είναι ίσος με το κλάσμα μίλια ανά ώρα. Έτσι, 100 κόμβοι αντιστοιχούν σε 185 χιλιόμετρα ανά ώρα.

B΄.1.3 Coordinated Universal Time (UTC)

Επίσης για να αποφευχθεί ενδεχόμενη παρανόηση λόγω των διαφορετικών χρονικών ζωνών της Γης, όλοι οι χρόνοι που χρησιμοποιούνται από τους πιλότους, τους ελεγκτές εναέριας κυκλοφορίας και κάθε εμπλεκόμενο στον κλάδο της αεροναυτιλίας, αναφέρονται σε χρόνο Συντονισμένης Παγκόσμιας Ώρας ή αλλιώς UTC (Coordinated Universal Time). Η ώρα UTC βασίζεται στα ατομικά ρολόγια και συγκεκριμένα στον χρόνο TAI (International Atomic Time). Έχει αντικαταστήσει την Ώρα Γκρήνουιτς (Greenwich Mean Time, GMT), δηλαδή την ώρα που αντιστοιχεί στην χρονική ζώνη του 1ου μεσημβρινού). Η ώρα UTC καλείται επίσης ώρα Ζούλου

(Zulu time) και συχνά προστίθεται στο τέλος το γράμμα Z. Για παράδειγμα η ώρα 1020Z σημαίνει 10 και 20 ώρα UTC. [6]

B'1.4 Ενδείκτες Τοποθεσίας

Ο Διεθνής Οργανισμός Πολιτικής Αεροπορίας ICAO (International Civil Aviation Organization) είναι μέλος του Οργανισμού Ηνωμένων Εθνών και ιδρύθηκε το 1947, έπειτα από την Σύμβαση του Σικάγο το 1944. Ρυθμίζει το διεθνές αεροπορικό δίκαιο με κύριο σκοπό την ασφάλεια των πτήσεων. Ανάμεσα στις υπόλοιπες αρμοδιότητές του είναι: η προτυποποίηση των ταξιδιωτικών και αεροπορικών εγγράφων σε παγκόσμιο ενιαίο τύπο, η κωδικοποίηση, διανομή και διαχείριση των κωδικών αυτών, για τα αεροδρόμια, τα αεροσκάφη και τις αεροπορικές εταιρίες σε παγκόσμιο επίπεδο, η ρύθμιση θεμάτων όπως υποδομές, επίπεδο θορύβου αεροσκαφών κ.α [7]

Ο ICAO δημοσιεύει ανά τακτά χρονικά διαστήματα το ICAO Document 7910, το οποίο αποτελεί ένα ευρετήριο, μέσα στο οποίο γίνεται αντιστοίχιση κάθε αεροδρομίου που υπάρχει στον κόσμο, με έναν μοναδικό κωδικό, που ορίζεται ως κωδικός αεροδρομίου (airport code) ή αλλιώς ενδείκτης τοποθεσίας (location indicator).

Οι εμπλεκόμενοι στην αεροναυτιλία, δεν χρησιμοποιούν στις μεταξύ τους επικοινωνίες την κανονική ονομασία ενός αεροδρομίου προκειμένου να αναφερθούν σε αυτό, αλλά αναφέρουν το συγκεκριμένο αεροδρόμιο με τον ενδείκτη τοποθεσίας του. Με αυτόν τον τρόπο αποφεύγονται οι χρονοτριβές, οι παρανοήσεις και τα ενδεχόμενα λάθη.

Ο κάθε κωδικός αποτελείται από 4 ψηφία και προσδιορίζει είτε ένα αεροδρόμιο είτε ένα FIR. Κάθε ψηφίο ανάλογα με τη θέση του, αναφέρεται σε συγκεκριμένη γεωγραφική περιοχή.

Το πρώτο ψηφίο από αριστερά αντιστοιχεί στην περιοχή ICAO, η οποία είναι μια μεγάλη γεωγραφική περιοχή. Π.χ το γράμμα L αντιστοιχεί στην νότια Ευρώπη και το γράμμα E στην βόρεια Ευρώπη.

Το δεύτερο ψηφίο αντιστοιχεί στην χώρα που βρίσκεται το αεροδρόμιο. Π.χ το γράμμα G αντιστοιχεί στην Ελλάδα και το γράμμα H στην Ολλανδία.

Το τρίτο και το τέταρτο ψηφίο αντιστοιχούν στην πόλη που βρίσκεται το αεροδρόμιο. Π.χ τα γράμματα KR αντιστοιχούν στην Κέρκυρα και τα γράμματα AM στο Άμστερνταμ. Υπάρχουν εξαιρέσεις ως προς το τρίτο και τέταρτο ψηφίο όταν ο κωδικός αναφέρεται σε FIR. Π.χ τα γράμματα BB αντιστοιχούν στο FIR BRINDISI.

Σύμφωνα με τα παραπάνω, ο κωδικός LGKR αντιστοιχεί στο αεροδρόμιο της Κέρκυρας, ο κωδικός EHAM αντιστοιχεί στο αεροδρόμιο του Άμστερνταμ και ο κωδικός LIBB στο FIR BRINDISI.

Β΄.1.5 Ραδιοβοηθήματα

Η ασφαλής πλοήγηση των αεροσκαφών επιτυγχάνεται με την χρήση ραδιοβοηθημάτων. Μέσω αυτών ένα αεροσκάφος μπορεί να εντοπίσει το στίγμα του στον χάρτη. Συνήθως βρίσκονται εντός κάποιου αεροδρομίου αλλά μπορεί να βρίσκονται και εκτός αυτού, σε συγκεκριμένη τοποθεσία. Η αρχή λειτουργίας τους στηρίζεται στο ζεύγος πομπός - δέκτης. Ο πομπός βρίσκεται στο έδαφος και μέσω μιας κεραίας εκπέμπει το χαρακτηριστικό όνομα του ραδιοβοηθήματος, σε μορφή μορσικού κώδικα. Το χαρακτηριστικό όνομα αποτελείται συνήθως από τρία γράμματα και σημειώνεται πάνω στον θεματικό αεροπορικό χάρτη. Για παράδειγμα το ραδιοβοήθημα με χαρακτηριστικό όνομα KRK βρίσκεται στην περιοχή της Λευκίμμης, νότια του νησιού της Κέρκυρας. Η λήψη του σήματος αυτού, γίνεται από έναν δέκτη ο οποίος βρίσκεται στο αεροσκάφος. Ο δέκτης αυτός δίνει την πληροφορία στο σύστημα πλοήγησης του αεροσκάφους και αποτυπώνεται πάνω σε χάρτη που παρακολουθεί ο πιλότος.

Υπάρχουν τα παρακάτω είδη ραδιοβοηθημάτων:

- NDB (Non-Directional Beacon)
- VOR (Very high frequency Omni-directional Radio range)
- DME (Distance Measuring Equipment)

Το ραδιοβοήθημα NDB είναι ένας μη κατευθυντικός ραδιοφάρος που εκπέμπει προς όλες τις κατευθύνσεις. Το σήμα αυτό λαμβάνεται από το αεροσκάφος από μια συσκευή που ονομάζεται ADF ((Automatic Direction Finder), γνωστό ως ραδιοπυξίδα. Το ADF δείχνει με μια βελόνα την κατεύθυνση από την οποία λαμβάνεται το σήμα του ραδιοβοηθήματος. Η εμβέλεια του NDB είναι περίπου από 50 έως 100 ναυτικά μίλια. Το NDB από μόνο του δεν μπορεί να βοηθήσει ένα αεροσκάφος να προσδιορίσει το ακριβές στίγμα του. Απαιτείται η ταυτόχρονη χρήση και δεύτερου NDB με το οποίο μπορεί ο πιλότος να βρει το ακριβές στίγμα του αεροσκάφους, βρίσκοντας το σημείο τομής τους ή της χρήσης του ραδιοβοηθήματος DME.

Το ραδιοβοήθημα VOR είναι ένας πανκατευθυντικός ραδιοφάρος πολύ υψηλής συχνότητας, που βρίσκεται συνήθως εντός κάποιου αεροδρομίου. Αποτελείται από πολλές κατευθυντικές κεραίες οι οποίες εκπέμπουν με συγκεκριμένη γωνία ανά μοίρα η κάθε μία, ακτινωτά γύρω

από το VOR. Ο δέκτης του αεροσκάφους λαμβάνει το σήμα μιας κεραίας από το VOR και μπορεί έτσι να προσδιορίσει την γωνία που έχει σε σχέση με το ραδιοβοήθημα και τον μαγνητικό βορρά. Η γωνία αυτή αναφέρεται και ως radial. Το VOR έχει εμβέλεια από 100 έως 200 ναυτικά μίλια και χρησιμοποιείται συνήθως σε πτήσεις με όργανα (IFR). Υπάρχουν τριών ειδών VOR: VORTAC, VOR-DME και VOR.

Το DME είναι ένας τηλεμετρικός ραδιοφάρος, ο οποίος βοηθά το αεροσκάφος να υπολογίσει την απόσταση του, από το ραδιοβοήθημα. Ουσιαστικά πρόκειται για ένα μικρό radar. Συνήθως το DME βρίσκεται μαζί με κάποιο άλλο ραδιοβοήθημα, π.χ VOR.

B'1.6 Εγχειρίδιο A.I.P

Όλα τα ραδιοβοηθήματα που υπάρχουν στο FIR μιας χώρας είναι δημοσιευμένα στο εγχειρίδιο Aeronautical Information Publication (A.I.P). Το AIP εγχειρίδιο εκδίδεται από την Υπηρεσία Πολιτικής Αεροπορίας της χώρας και περιέχει αεροναυτικές πληροφορίες (κανονισμούς, διαδικασίες, αεροναυτικούς χάρτες κ.α) που είναι απαραίτητες για την αεροναυτιλία.[8]

B'1.7 Waypoints

Στους θεματικούς αεροπορικούς χάρτες, προκειμένου να βοηθηθούν οι πιλότοι στην ασφαλή πλοήγηση των αεροσκαφών, υπάρχουν και τα νοητά σημεία στην επιφάνεια της γης, τα οποία ονομάζονται διασταυρώσεις (intersections ή waypoints). Τα σημεία αυτά είναι σταθερά και οι συντεταγμένες τους είναι δημοσιευμένες στο εγχειρίδιο AIP κάθε χώρας.

Αυτά μπορεί να είναι waypoints που ανήκουν σε IFR θεματικούς χάρτες (χρησιμοποιούν για πτήση με όργανα) ή VFR θεματικούς χάρτες (χρησιμοποιούν για πτήση εξ όψεως). Τα IFR waypoints έχουν 5 γράμματα ενώ τα VFR μπορεί να έχουν περισσότερα. Π.χ το waypoint PITAS είναι σημείο που βρίσκεται βόρεια της Κέρκυρας και είναι σημείο εξόδου προς το FIR της Αλβανίας ενώ το waypoint PELEKAS είναι VFR waypoint και βρίσκεται δυτικά της Κέρκυρας.

Β΄.1.8 Αεροδιάδρομοι

Εκτός από τα waypoints, στους θεματικούς αεροπορικούς χάρτες έχουμε τους αεροδιαδρόμους (airways). Πρόκειται για νοητές ευθείες οι οποίες ενώνουν συνήθως δύο waypoints, ή κάποιο waypoint με ένα ραδιοβοήθημα ή δύο ραδιοβοήθημα. Τα αεροσκάφη είναι υποχρεωμένα να βρίσκονται πάντα πάνω στο ίχνος ενός αεροδιαδρόμου.

Έχουν χαρακτηριστικό όνομα που αποτελείται από ένα γράμμα και έναν αριθμό. Για παράδειγμα ο αεροδιάδρομος B1 ενώνει τα waypoints SOKRI και VELOP, βόρεια και ανατολικά του νησιού των Κυθήρων. Οι διάδρομοι συνήθως ανήκουν σε θεματικούς IFR χάρτες. Έχουν ένα ελάχιστο ύψος και ένα μέγιστο. Π.χ ο αεροδιάδρομος B1 έχει ελάχιστο ύψος 9000 πόδια. Επίσης κάποιοι αεροδιάδρομοι είναι διπλής κατεύθυνσης και κάποιοι μονής.

Β΄.1.9 Βαρομετρική Πίεση και Ύψος Πτήσης

Ατμοσφαιρική πίεση ή Βαρομετρική πίεση ονομάζεται η πίεση που ασκεί η ατμόσφαιρα με το βάρος της στην επιφάνεια της Γης. Στην επιφάνεια της Γης η ατμοσφαιρική πίεση ισούται κατά μέσον όρο με το βάρος στήλης ύδατος ύψους 11 μέτρων περίπου ή με το βάρος στήλης υδραργύρου ύψους 760 χιλιοστών. [9] Η ποσότητα των μορίων του αέρα, όσο ανεβαίνουμε στην ατμόσφαιρα λιγοστεύει και κατά συνέπεια η ατμοσφαιρική πίεση. Το όργανο του αεροσκάφους που μετρά την ατμοσφαιρική πίεση, την μετρά μέσω του αέρα που εισέρχεται σε ένα σταθερό σωλήνα-pitot ο οποίος βρίσκεται συνήθως στο ρύγχος του αεροσκάφους. [10]

Το αεροσκάφος υπολογίζει το υψόμετρο στο οποίο πετά, με τη χρήση ενός στατικού συστήματος pitot (pitot static system). [11] Το όργανο αυτό αντισταθμίζει την ατμοσφαιρική πίεση που υπάρχει στο επίπεδο της θαλάσσης πάνω από το μέρος από το οποίο πετά. Η ατμοσφαιρική πίεση στο επίπεδο της θαλάσσης καλείται QNH και μετριέται σε hPa ή mb. Η τιμή της QNH είναι απαραίτητη σε ένα αεροσκάφος για να γνωρίζει το πραγματικό του ύψος. Λανθασμένη τιμή, ενδεχομένως να οδηγήσει σε ατύχημα αφού ο πιλότος υποθέτει ότι πετά σε συγκεκριμένο υψόμετρο, με πιθανή πρόσκρουση σε σταθερό εμπόδιο, π.χ κορυφή ενός βουνού.

Η εισαγωγή της τιμής QNH - Regional or airfield pressure setting - γίνεται από τον πιλότο χειροκίνητα και λαμβάνεται από τους μετεωρολογικούς σταθμούς του εκάστοτε τόπου ή

αναμεταδίδεται στον πιλότο από τον ελεγκτή.

Απόλυτο ύψος MSL (Mean Sea Level) είναι η κατακόρυφη απόσταση του αεροσκάφους από την μέση στάθμη της θάλασσας.

Σχετικό ύψος AGL (Above Ground Level) είναι η κατακόρυφη απόσταση από το έδαφος.

B'1.10 Επίπεδο Πτήσης

Επίπεδο πτήσης FL (Flight Level) είναι η κατακόρυφη απόσταση από το έδαφος όταν ο πιλότος έχει θέσει ως QNH την τιμή 1013 hPa. Εκφράζεται σε εκατοντάδες πόδια. Για παράδειγμα όταν ένα αεροσκάφος πετά στο FL 100 σημαίνει ότι πετά στα 10000 πόδια MSL όταν η πίεση σε αυτό το μέρος είναι 1013 hPa. Καταλαβαίνουμε ότι η πραγματική πίεση QNH μπορεί να είναι διαφορετική αλλά αυτό δεν μας πειράζει, γιατί έτσι επιτυγχάνουμε μια κανονικοποίηση ως προς το ύψος που πετούν τα αεροσκάφη ανεξαρτήτως του μέρους που βρίσκονται και χωρίς να χρειάζονται να ξέρουν την πραγματική QNH.

Έτσι δυο αεροσκάφη που πετούν στην ίδια περιοχή, αν το ένα πετά σε FL210 και το άλλο σε FL200, έχουν κάθετο διαχωρισμό 1000 ποδών, αφού έχουν το ίδιο QNH 1013 hPa.

Στην πραγματικότητα, ο πιλότος θέτει ως QNH την τιμή 1013 hPa, αφού ξεπεράσει κάποιο συγκεκριμένο ύψος το οποίο ονομάζεται transition altitude. Κάτω από το transition altitude θα πρέπει να βάλει την τιμή QNH που έχει το μέρος αυτό ώστε να εξασφαλίσει ότι θα γνωρίζει το πραγματικό του υψόμετρο πτήσης. Πάνω από το transition altitude του είναι αδιάφορη η πραγματική QNH. Το transition altitude μπορεί να είναι διαφορετικό ανά περιοχή. Για παράδειγμα, στην Τερματική Περιοχή Κέρκυρας (Kerkyra TMA) είναι 5000 πόδια ενώ στην Στρατιωτική Τερματική Περιοχή Λάρισας (Larissa MTMA) είναι 11000 πόδια.

Ελάχιστο απόλυτο ύψος ασφάλειας MSA (Minimum Safe Altitude) είναι το ύψος που μπορεί να πετάξει ένα αεροσκάφος με ασφάλεια σε μια περιοχή και καθορίζεται από το ύψος των τεχνητών ή φυσικών εμποδίων που υπάρχουν σε αυτήν.

B'1.11 Flight Information Region (FIR)

Περιοχή Πληροφοριών Πτήσης Flight Information Region (FIR) είναι μια περιοχή εναέριου χώρου στην οποία παρέχεται υπηρεσία Πληροφοριών Πτήσεων (Flight Information Service).

Ο Διεθνής Οργανισμός Πολιτικής Αεροπορίας ICAO εξουσιοδοτεί ποια χώρα θα έχει τον επιχειρησιακό έλεγχο ενός εναέριου χώρου. [12]

Συνήθως οι μικρές χώρες έχουν ένα μόνο FIR, π.χ Ελλάδα (ATHINAI FIR) ενώ οι μεγαλύτερες χώρες μπορεί να έχουν περισσότερα από ένα FIR π.χ Ιταλία (ROME FIR, MILAN FIR, BRINDISI FIR). Το ATHINAI FIR εκτείνεται από το έδαφος μέχρι τα 24500 πόδια (ή FL245). Ο εναέριος χώρος από τα 24500 πόδια (ή FL245) μέχρι τα 46000 (ή FL460) καλείται και ως UIR (Upper Information Region). Το ATHINAI FIR χωρίζεται σε δύο κατηγορίες, ελεγχόμενου και μη ελεγχόμενου εναέριου χώρου. Στον ελεγχόμενο εναέριο χώρο (controlled airspace) παρέχεται η υπηρεσία Ελέγχου Εναέριας Κυκλοφορίας (EEK). Στον μη ελεγχόμενο εναέριο χώρο (uncontrolled airspace) παρέχεται η υπηρεσία Πληροφοριών Πτήσεων (FIS) με πληροφορίες καιρού, ασφάλεια των πτήσεων, κυκλοφορίας. Τα ελληνικά αεροδρόμια παρόμοια χωρίζονται σε δύο κατηγορίες, ελεγχόμενων και μη ελεγχόμενων αεροδρομίων. Στα ελεγχόμενα αεροδρόμια, τα οποία είναι συνήθως τα μεγάλα διεθνή αεροδρόμια, π.χ Αθηνών, Κέρκυρας, Θεσσαλονίκης παρέχεται η υπηρεσία Ελέγχου Εναέριας Κυκλοφορίας (EEK) ATC (Air Traffic Control) από ελεγκτές της ΥΠΑ (Υπηρεσίας Πολιτικής Αεροπορίας). Στα μη ελεγχόμενα αεροδρόμια, τα οποία είναι συνήθως τα μικρότερα αεροδρόμια, π.χ Πάρου, Νάξου, Κυθήρων, Καστελόριζου παρέχεται η υπηρεσία Πληροφοριών Πτήσεων Αεροδρομίου (ΠΠΑ) AFIS (Aerodrome Information Service). Στα αεροδρόμια AFIS, η υπηρεσία αυτή παρέχεται από Τηλεπικοινωνιακούς (AFISO) ή Ηλεκτρονικούς (ATSEP) υπαλλήλους της ΥΠΑ και αφορά πληροφορίες όπως κίνηση αεροσκαφών (traffic), πληροφορίες παρόντος καιρού (METAR), αναμετάδοση οδηγιών από μονάδες EEK, πληροφορίες ασφάλειας πτήσεων. Τα αεροδρόμια AFIS δεν διαθέτουν εικόνα από Radar περιοχής σε αντίθεση με τα ελεγχόμενα αεροδρόμια. [13][14]

Β΄.1.12 Περιοχές CTR, TMA, ATZ

Το FIR περιλαμβάνει τις παρακάτω περιοχές:

Control Zone or Controlled Traffic Region (CTR) Ζώνη Ελέγχου

Η CTR είναι ο ελεγχόμενος εναέριος χώρος γύρω από ένα αεροδρόμιο. Τα αεροπλάνα που βρίσκονται ή πετούν μέσα στην CTR, είναι υποχρεωμένα να ακολουθούν τις οδηγίες του ελεγκτή εναέριας κυκλοφορίας που είναι υπεύθυνος για την συγκεκριμένη CTR. Η CTR κάθε αεροδρομίου έχει συγκεκριμένο ύψος μέχρι το οποίο εκτείνεται και το οποίο αναφέρεται στο εγχειρίδιο AIP. Η οριζόντια εμβέλεια της CTR είναι 10 χλμ.[15][16]

Terminal Manoeuvring Area (TMA) or Military Terminal Manoeuvring Area (MTMA) Τερματική Περιοχή ή Στρατιωτική Τερματική Περιοχή. Η TMA είναι ο ελεγχόμενος εναέριος χώρος που περιλαμβάνει ένα ή περισσότερα μεγάλα αεροδρόμια και ελέγχεται συνήθως από το μεγαλύτερο εξ αυτών. Η TMA βρίσκεται πάνω από την CTR και φτάνει συνήθως μέχρι τα 24000 πόδια (ή FL240). Αν περιλαμβάνει στρατιωτικό αεροδρόμιο η TMA αναφέρεται ως MTMA.[15][17].

Όλα τα αεροσκάφη που εισέρχονται στην περιοχή αυτή είναι υποχρεωμένα να λάβουν οδηγίες για το πώς θα προσεγγίσουν το αεροδρόμιο που θέλουν να προσγειωθούν και ποια διαδικασία άφιξης ST.AR θα εκτελέσουν εφόσον πετούν με κανόνες IFR. Μπορεί όμως ένα αεροσκάφος απλώς να διέρχεται από την περιοχή αυτή, αν έχει προορισμό κάποιο άλλο αεροδρόμιο. Σε αυτή την περίπτωση πάλι είναι υποχρεωμένο να καλέσει τον ελεγκτή προσέγγισης, αν βρίσκεται κάτω από το ανώτερο ύψος της τερματικής περιοχής. Αν βρίσκεται ψηλότερα από αυτό επικοινωνεί μόνο με το ΚΕ.ΠΑ.Θ (Κέντρο Ελέγχου Περιοχής Αθηνών) αν πρόκειται για την Ελλάδα.

Aerodrome Traffic Zone (ATZ) Ζώνη Κυκλοφορίας Αεροδρομίου. Η ATZ είναι ο εναέριος χώρος γύρω από ένα αεροδρόμιο, συνήθως μη ελεγχόμενο. [18]

B'1.13 Αεροδρόμιο

Ένα αεροδρόμιο περιλαμβάνει τα παρακάτω στοιχεία:

- Έναν διαδρόμο (runway) προσγείωσης - απογείωσης (συνήθως ο διάδρομος αυτός αναφέρεται λανθασμένα ως αεροδιάδρομος). Το κάθε άκρο εκ των δύο, του διαδρόμου έχει επάνω του έναν διψήφιο αριθμό, με τον οποίο προσδιορίζεται από τον ελεγκτή και από τους πιλότους. Ο αριθμός αυτός αντιστοιχεί στις μοίρες που βλέπει ο πιλότος στο όργανο πλοήγησης όταν προσεγγίζει από το συγκεκριμένο άκρο για να προσγειωθεί, περικοπτόμενος κατά το τελευταίο ψηφίο. Για παράδειγμα, ο διάδρομος προσγειώσεων απογείωσης του αεροδρομίου Ιωάννης Καποδίστριας της Κέρκυρας έχει άκρα 16 και 34. Αυτό σημαίνει ότι ένα αεροσκάφος που έρχεται να προσγειωθεί από τον διάδρομο 34, ο πιλότος του θα βλέπει στο όργανο πλοήγησης τον αριθμό 340 και θα προσγειωθεί από την πλευρά που βρίσκεται το Ποντικονήσι. Διευκρινίζεται ότι ο αριθμός 360 αντιστοιχεί στο μαγνητικό βορρά και το 180 στο μαγνητικό νότο.
- Το apron που είναι ο χώρος στάθμευσης και ανεφοδιασμού των αεροσκαφών
- τους συνδετήριους διαδρόμους (taxiways) που συνδέουν τον διάδρομο προσγείωσης με το apron.

- τον Πύργο Ελέγχου Αεροδρομίου (Π.Ε.Α) – Control Tower. Στον Π.Ε.Α βρίσκονται οι ελεγκτές εναέριας κυκλοφορίας (Ε.Ε.Κ) οι οποίοι επικοινωνούν με τους πιλότους των αεροσκαφών που βρίσκονται στον τομέα ευθύνης του αεροδρομίου. Τα αεροσκάφη αυτά, είναι αεροσκάφη που βρίσκονται στο αργον, αεροσκάφη που πρόκειται να απογειωθούν από το αεροδρόμιο, αεροσκάφη που κατευθύνονται προς αυτό για να προσγειωθούν ή υπερπιτάμενα αεροσκάφη τα οποία διέρχονται τον τομέα ευθύνης του αεροδρομίου και
- έχουν άλλον προορισμό. Συνήθως το έργο αυτό κατανέμεται σε τρεις ή παραπάνω ελεγκτές αναλόγως του φόρτου κυκλοφορίας της περιοχής.

Κάθε αεροσκάφος που εισέρχεται στον τομέα ευθύνης του αεροδρομίου οφείλει να επικοινωνήσει με τον ελεγκτή εναέριας κυκλοφορίας, από τον οποίο θα πάρει οδηγίες. Οι οδηγίες αυτές είναι ουσιαστικά η άδεια που δίνει ο ελεγκτής στο αεροσκάφος να ακολουθήσει μια συγκεκριμένη πορεία σε συγκεκριμένο ύψος.

B΄.1.14 Radar

Ο Ε.Ε.Κ έχει ως ευθύνη να διαχωρίσει με ασφάλεια, τα αεροσκάφη από άλλα αεροσκάφη και από άλλα φυσικά εμπόδια, π.χ βουνά ή κτήρια. Ο διαχωρισμός αυτός είναι κάθετος και οριζόντιος, δηλαδή καθ' ύψος και κατά απόσταση.

Για να επιτελέσει ο Ε.Ε.Κ το έργο του, έχει ως βοήθεια την εικόνα που λαμβάνει από το ραντάρ που βρίσκεται κοντά στο αεροδρόμιο (συνήθως στην κορυφή κάποιου βουνού από την οποία σαρώνει την συγκεκριμένη περιοχή).

Η εικόνα αυτή περιλαμβάνει τον χάρτη της περιοχής του αεροδρομίου. Πάνω σε αυτή εμφανίζονται ως στόχοι τα αεροσκάφη που πετούν την συγκεκριμένη στιγμή. Οι στόχοι αυτοί εμφανίζουν επιπλέον πληροφορίες όπως το χαρακτηριστικό κλήσης του αεροσκάφους, το ύψος που πετά, την ταχύτητά του, τον μοναδικό τετρανήφιο κωδικό αριθμό (squawk) που έχει εκδοθεί για το συγκεκριμένο αεροσκάφος από τον κεντρικό τομέα Ελέγχου Εναέριας Κυκλοφορίας. Σε μικρά και μη ελεγχόμενα αεροδρόμια συνήθως ο Πύργος Ελέγχου δεν έχει εικόνα της κυκλοφορίας στην περιοχή του από κάποιο radar και η θέση των αεροσκαφών λαμβάνεται μόνο δια μέσου αναφοράς (report) από τους πιλότους ανά τακτά χρονικά διαστήματα μέσω του ασύρματου.

Το radar που δίνει εικόνα στον Πύργο Ελέγχου αποτελείται από το πρωτεύον και το δευτερεύον radar. Το πρωτεύον ραντάρ PSR (Primary Surveillance Radar) [19] χρησιμοποιείται για την

απεικόνιση στόχων. Κάθε ένας στόχος αντιστοιχεί σε ένα αεροσκάφος που πετά στην περιοχή. Με το PSR radar ο ελεγκτής εναέριας κυκλοφορίας δεν μπορεί να δει άλλες πληροφορίες που αφορούν το αεροσκάφος εκτός από το στόχο που απεικονίζει την θέση του στο χάρτη.

Με τη χρήση του δευτερεύοντος radar SSR (Secondary Surveillance Radar) [20] ο ελεγκτής μπορεί να δει επιπλέον πληροφορίες που αφορούν το αεροσκάφος όπως το ύψος που πετά και τον αριθμό squawk. Αυτό επιτυγχάνεται με τη χρήση μιας συσκευής που βρίσκεται στο αεροσκάφος και ονομάζεται transponder. Ο transponder [21] επικοινωνεί με το SSR radar και μεταδίδει τις συγκεκριμένες πληροφορίες.

B'1.15 Κωδικός Αριθμός Squawk

Ο κωδικός αριθμός squawk (transponder code) [21] εκδίδεται από τον κεντρικό τομέα Ελέγχου Εναέριας Κυκλοφορίας κάθε χώρας, για την Ελλάδα ο τομέας αυτός είναι το Κ.Ε.Π.ΑΘ (Κέντρο Ελέγχου Περιοχής Αθηνών). Είναι μοναδικός για κάθε αεροσκάφος και εκδίδεται λίγη ώρα πριν την ενεργοποίηση και την απογείωση της πτήσης. Υπάρχουν κάποιοι δεσμευμένοι κωδικοί οι οποίες αναφέρονται σε διάφορες καταστάσεις. Π.χ 7500 Κατάσταση Αεροπειρατείας Aircraft Hijacking, 7600 Κατάσταση Απώλειας Ραδιοεπικοινωνιών Radio Failure, 7700 Κατάσταση Κινδύνου Emergency.

Όλες οι πτήσεις VFR έχουν τον ίδιο αριθμό squawk 7000. Οι πτήσεις που αναχωρούν από το αεροδρόμιο της Αθήνας Ελευθέριος Βενιζέλος έχουν κωδικούς από 7200 έως 7227 και όλες οι άλλες πτήσεις εντός του ATHINAI FIR από 7230 έως 7277. [22]

B'1.16 Κανόνες Πτήσης

Ένα αεροσκάφος μπορεί να πετάξει με δύο τρόπους, με κανόνες πτήσης εξ όψεως V.F.R (Visual Flight Rules) ή με κανόνες πτήσης με όργανα I.F.R (Instrument Flight Rules).

Στην πτήση εξ όψεως VFR, ο πιλότος πετάει μόνο κατά την διάρκεια της ημέρας και πρέπει να έχει ολοκληρώσει την πτήση του πριν δύσει ο ήλιος. Πετάει εξ όψεως χωρίς δηλαδή την χρήση ραδιοβοηθημάτων, ακολουθώντας VFR διαδρομές οι οποίες είναι δημοσιευμένες στο εγχειρίδιο AIP και αφορούν συγκεκριμένες περιοχές της Ελλάδας. Π.χ ATHINAI TMA. Παρόλα αυτά σε συγκεκριμένες περιπτώσεις κάνει χρήση των ραδιοβοηθημάτων. Π.χ για να προσεγγίσει ένα

αεροδρόμιο.

Στην πτήση IFR ο πιλότος πετάει κάνοντας χρήση των ραδιοβοηθημάτων που βρίσκονται στο έδαφος σε συνδυασμό με τα όργανα του αεροσκάφους (Flight Management System FMS). [23] Ακολουθεί το νοητό ίχνος των IFR αεροδιαδρόμων που είναι δημοσιευμένοι στο εγχειρίδιο AIP. Επίσης για να προσγειωθεί σε ένα αεροδρόμιο θα πρέπει να ακολουθήσει μια ενόργανη διαδικασία προσγείωσης ST.AR (Standard Arrival). Παρόμοια κάθε αεροσκάφος που πρόκειται να απογειωθεί θα πρέπει να ακολουθήσει μια ενόργανη διαδικασία απογείωσης SID (Standard Instrument Departure). Το κάθε αεροδρόμιο έχει τις δικές του STAR και SID για κάθε άκρο του διαδρόμου προσγειώσεων απογειώσεων και οι οποίες είναι δημοσιευμένες στο εγχειρίδιο AIP. Και το STAR και το SID έχουν ένα χαρακτηριστικό όνομα που αποτελείται από δύο μέρη. Το 1^ο μέρος είναι ένα σημείο waypoint που βρίσκεται κοντά στο αεροδρόμιο και το 2^ο μέρος είναι ένας αριθμός μαζί με ένα γράμμα που δείχνει από ποιο άκρο θα εκτελεστεί. Π.χ για το αεροδρόμιο των Κυθήρων ένα SID είναι το SOKRI 1K. Ο ελεγκτής που βρίσκεται στον Πύργο Ελέγχου αναφέρει στον πιλότο του αεροσκάφους ποια SID ή STAR θα πρέπει να εκτελέσει κατά την απογείωση ή προσγείωση.

Β΄.1.17 Σχέδιο Πτήσης

Κάθε αεροσκάφος πριν απογειωθεί ή πριν εισέλθει στον ελληνικό εναέριο χώρο είναι υποχρεωμένο να καταθέσει σχέδιο πτήσης. Με την κατάθεση του σχεδίου πτήσης οι αρμόδιες ελληνικές αρχές (πολιτικές και στρατιωτικές) που είναι επιφορτισμένες για την ασφάλεια των πτήσεων και του εθνικού εναέριου χώρου μαθαίνουν τα στοιχεία του αεροσκάφους και όσες άλλες πληροφορίες χρειάζονται (διαδρομή, προορισμός, επιβάτες, καύσιμα κ.λ.π.).

Το σχέδιο πτήσης είναι μια φόρμα η οποία περιλαμβάνει τα παρακάτω πεδία:

- Χαρακτηριστικό κλήσης του αεροσκάφους στον ασύρματο (Call Sign)
- Κανόνες πτήσης (Visual Flight Rules, πτήση εξ όψεως χωρίς τη χρήση οργάνων ή Instrument Flight Rules, πτήση με τη χρήση οργάνων)
- Τύπος πτήσης (πχ πτήση Εμπορική ή Γενικής Αεροπορίας ή Στρατιωτική πτήση)
- Τύπος αεροσκάφους
- Κατηγορία αεροδινών (αναταράξεις λόγω του βάρους του)
- Συσκευές πλοήγησης και εξοπλισμός που διαθέτει το αεροσκάφος
- Αεροδρόμιο αναχώρησης
- Ώρα αναχώρησης (σε διεθνή ώρα Γκρήνουιτς Universal Time Clock U.T.C)
- Ταχύτητα Πλεύσης (σε κόμβους ή χιλιόμετρα)

- Διαδρομή (Καθορισμένα σημεία και αεροδιάδρομοι τα οποία βρίσκονται σε θεματικούς αεροπορικούς χάρτες και χρησιμοποιούνται από τα αεροσκάφη)
- Αεροδρόμιο προορισμού
- Εκτιμώμενη διάρκεια πτήσης (εκφρασμένη σε ώρα)
- Λοιπές πληροφορίες (όνομα πιλότου, ημερομηνία πτήσεις, επιπλέον πληροφορίες που χρειάζεται να ξέρουν οι Ελεγκτές Εναέριας Κυκλοφορίας για την συγκεκριμένη πτήση)
- Ποσότητα καυσίμων που έχει το αεροσκάφος για την πτήση αυτή (εκφρασμένη σε χρονική διάρκεια πτήσης)
- Συνολικά άτομα που επιβαίνουν στο αεροσκάφος (πλήρωμα και επιβάτες)
- Συσκευές επικοινωνίας σε κατάσταση έκτακτης ανάγκης (UHF, VHF, ELT)
- Σωσίβια Jackets
- Σωσίβιες λέμβοι και χωρητικότητα αυτών (σε περίπτωση προσθαλάσωσης)
- Χρώμα αεροσκάφους
- Παρατηρήσεις
- Ονοματεπώνυμο του Κυβερνήτη

Στην αεροπορική ορολογία όταν ένα σχέδιο πτήσης έχει ενεργοποιηθεί, δηλαδή το αεροσκάφος το οποίο πραγματοποιεί την πτήση έχει απογειωθεί, αναφέρεται ότι το σχέδιο πτήσης είναι «ανοικτό» ή «έχει ανοίξει». Αντίστοιχα όταν ένα σχέδιο πτήσης έχει απενεργοποιηθεί, δηλαδή το αεροσκάφος το οποίο πραγματοποίησε την πτήση έχει προσγειωθεί τότε αναφέρεται ότι το σχέδιο πτήσης είναι «κλειστό» ή «έχει κλείσει».

B'1.18 Aeronautical Fixed Telecommunication Network

Η κατάθεση του σχεδίου πτήσης γίνεται μέσω του τμήματος Τηλεπικοινωνιών της Υπηρεσίας Πολιτικής Αεροπορίας που βρίσκεται στο αεροδρόμιο αναχώρησης. Η κατάθεση γίνεται μέσω του διεθνούς δικτύου σταθερών αεροναυτικών επικοινωνιών (Aeronautical Fixed Telecommunication Network A.F.T.N), το οποίο συνδέει όλα τα αεροδρόμια του κόσμου μεταξύ τους. Το δίκτυο AFTN είναι ένα δίκτυο που στηρίζεται πάνω στο πρωτόκολλο TCP/IP αλλά δεν έχει σύνδεση με το Διαδίκτυο. [24]

Τα τελευταία χρόνια όμως δίνεται η δυνατότητα ένα σχέδιο να κατατεθεί επί πληρωμή και μέσω web εφαρμογών, που συνδέονται μέσω πιστοποιημένης σύνδεσης με το AFTN.

Τα σχέδια πτήσης αποστέλλονται μέσω μηνύματων του AFTN, αποτελούμενα από χαρακτήρες κώδικα ASCII με τυποποιημένο πρωτόκολλο

Παρακάτω βλέπουμε ένα τυπικό μήνυμα AFTN που στο σώμα κειμένου περιλαμβάνει ένα σχέδιο πτήσης αεροσκάφους από το αεροδρόμιο της Κέρκυρας προς το αεροδρόμιο των Ιωαννίνων.

ZCZC

DD LGGGCFMU

120900 LGKRZTZX

(FPL-SXAJT-VG

-01C172/L-S/C

-LGKR1400

-N0120VFR IGOU YNN-

LGIO0030

-DOF/200715)

NNNN

B΄.1.19 NOTAM

Αεροναυτική Αγγελία NOTAM (Notice To Airmen) είναι μια αναγγελία με την οποία λαμβάνουν γνώση οι πιλότοι αεροσκαφών και άλλοι εμπλεκόμενοι στον κλάδο της αεροναυτιλίας, για θέματα που αφορούν την ασφάλεια των πτήσεων. Εκδίδεται από την Αρχή Πολιτικής Αεροπορίας μιας χώρας και αφορά θέματα όπως αλλαγές στον αεροναυτικό εξοπλισμό, παροχή υπηρεσιών, διαδικασίες ή κινδύνους που έχουν δημιουργηθεί (εμπόδια, κτήρια, σμήνη πουλιών κ.α), διεξαγωγή στρατιωτικών ασκήσεων κ.α. Κάθε αγγελία μπορεί να αφορά διαδρομές ή τοποθεσίες (π.χ αεροδρόμια). Υπάρχουν παραλλαγές αγγελιών που έχουν διαφορετική ονομασία όταν αφορούν συγκεκριμένα θέματα, όπως π.χ συγκέντρωση χιονιού πάνω στον διάδρομο προσγείωσης απογείωσης καλείται SNOWTAM, συγκέντρωση σμήνους πουλιών κοντά στα άκρα ή στον διάδρομο καλείται BIRDTAM, συγκέντρωση ηφαιστειακής τέφρας σε συγκεκριμένα ύψη και περιοχές καλείται ASHTAM. [25]

Ένα NOTAM μπορεί να έχει μόνιμη χρονική ισχύ (Permanent) ή περιορισμένη.

Όλα τα NOTAM που εκδίδονται παγκοσμίως μεταδίδονται μέσω του δικτύου AFTN. Επίσης κάθε αεροδρόμιο, ή μονάδα σε όλο τον κόσμο μπορεί να αναζητήσει ένα συγκεκριμένο NOTAM, γνωρίζοντας τον αριθμό έκδοσης και τον χρόνο κατά τον οποίο εκδόθηκε, καθώς και το FIR της χώρας ή το αεροδρόμιο που αφορά .

Η αεροναυτική αγγελία NOTAM αποτελείται από 5 πεδία, που αρχίζουν με τα γράμματα Q,A,B,C,D,E. Στο πεδίο Q προσδιορίζεται το στοιχείο το οποίο αφορά το NOTAM κωδικοποιημένο κατά ICAO. Στο πεδίο A προσδιορίζεται το αεροδρόμιο ή το FIR στο οποίο

ανήκει το στοιχείο που αφορά το NOTAM. Στα πεδία B,C προσδιορίζονται αντίστοιχα η χρονική αρχή και το χρονικό τέλος ισχύος του NOTAM, με format κατά ICAO, δηλαδή Έτος-Μήνας-Ημέρα-Ωρα-Λεπτά (σε χρόνο UTC). Στο πεδίο E προσδιορίζεται το θέμα του NOTAM, με συγκεκριμένη φρασεολογία η οποία χρησιμοποιεί συντομογραφίες κατά ICAO.

Παρακάτω βλέπουμε ένα τυπικό NOTAM.

Q) LGGG/QMRXX/IV/NBO /A /000/999/3936N01954E

A) LGKR

B) 2108030927

C) 2111032359

E) BACKTRACK OPERATIONS, AFTER LANDING OR PRIOR TO TAKE-OFF,
EFFECTIVE ONLY AT THE EXTREMITIES OF RWY 16/34 FOR ALL ACFT TYPES,
EXCEPT FOR ACFT LESS THAN FIVE THOUSAND SEVEN HUNDRED KGR 5700 MTOW.

Αναλύοντας το NOTAM βλέπουμε ότι αναφέρεται στο αεροδρόμιο της Κέρκυρας (LGKR), έχει χρονική ισχύ από 3 Αυγούστου 2021, 9.27 π.μ UTC έως 3 Νοεμβρίου 2021, 11.59 μ.μ UTC και αναφέρεται στην διαδικασία BACKTRACK με την οποία ένα αεροσκάφος αλλάζει κατεύθυνση 180° και η οποία επιτρέπεται να γίνει μόνο στα δύο άκρα του διαδρόμου 16/34. Αφορά μόνο τα μεγαλύτερα αεροσκάφη. Επίσης βλέπουμε ότι χρησιμοποιούνται οι συντομογραφίες ACFT και RWY, οι οποίες αντιστοιχούν στις λέξεις aircraft και runway.

B'2 Μετεωρολογία

Θα κάνουμε μια εισαγωγή στη Μετεωρολογία για να εξηγήσουμε κάποια στοιχεία που ανήκουν σε αυτήν και τα οποία χρησιμοποιούνται στην εργασία μας. Για λόγους συντομίας κάποια στοιχεία παραλείπονται γιατί δεν ενδιαφέρουν τον αναγνώστη.

B'.2.1 METAR

Για την πραγματοποίηση μιας πτήσης, είναι απαραίτητο στοιχείο, να γνωρίζει ο πιλότος τις καιρικές συνθήκες που επικρατούν σε όλη την διαδρομή, από το αεροδρόμιο αναχώρησης έως και το αεροδρόμιο προορισμού. Είτε η πτήση είναι IFR είτε VFR, ο πιλότος πριν μπει καν στο αεροσκάφος θα ενημερωθεί για τις καιρικές συνθήκες. Οι καιρικές συνθήκες μπορούν να επηρεάσουν την διεξαγωγή μιας πτήσης, από την αλλαγή πορείας μέχρι και τη ματαίωσή της.

Η ενημέρωση των πιλότων για τις καιρικές συνθήκες που επικρατούν σε ένα αεροδρόμιο γίνεται με τη μετεωρολογική αναφορά METAR (Aviation Routine Weather Report). [26]

Εκδίδεται από τους μετεωρολόγους του αεροδρομίου και αποστέλλεται από την EMY μέσω του δικτύου AFTN. Μέσω του δικτύου αυτού, μπορεί να ληφθεί από οποιοδήποτε εμπλεκόμενο, είτε αυτός είναι πιλότος, είτε επιμελητεία μιας εταιρείας, ή άλλη μονάδα αεροναυτιλίας που βρίσκεται σε κάποιο άλλο μέρος ή αεροδρόμιο σε όλο τον κόσμο. Επίσης το METAR προωθείται και στο διαδίκτυο από ιστοσελίδες καιρού ή άλλες υπηρεσίες που έχουν αντικείμενο το αεροπορικό έργο.

Το METAR εκδίδεται δύο φορές ανά ώρα, δηλαδή και 20 πρώτα λεπτά της ώρας το πρώτο δελτίο, και 50 πρώτα λεπτά της ώρας, το δεύτερο. Η παρατήρηση των μετεωρολόγων γίνεται 10 λεπτά πριν την έκδοσή του METAR. Στην περίπτωση που οι μετεωρολογικές συνθήκες αλλάξουν σημαντικά τότε εκδίδεται νέο δελτίο πριν την επόμενη καθορισμένη ώρα. Αποτελείται από τα παρακάτω μέρη:

- Αεροδρόμιο το οποίο αφορά το συγκεκριμένο METAR
- Ώρα έκδοσης σε χρόνο UTC
- Διεύθυνση (στο έδαφος) και ταχύτητα του ανέμου σε κόμβους
- Ορατότητα
- Ορατότητα στον διάδρομο προσγείωσης από τα δύο άκρα του, αν αυτή είναι κάτω από τα 1500 μέτρα
- Καιρικές συνθήκες Εκφράζονται με αρτικόλεξα. (π.χ βροχή RA, καταιγίδα TS, ψιχάλα DZ). Χρησιμοποιούνται μαζί και τα σύμβολα + και – όταν έχουμε αντίστοιχα ισχυρά ή ασθενή φαινόμενα.
- Νέφη. Αναφέρονται στο ποσοστό κάλυψης του ουρανού με νέφη, αν χωρίσουμε τον ουρανό σε 8 μέρη. Έτσι για κάλυψη νεφών 0/8 : Καθαρός ουρανός Sky clear (SKC), 1/8-2/8: Λίγα Few (FEW), 3/8-4/8: Διασκορπισμένα Scattered (SCT), 5/8-6/8: Σπασμένα Broken (BKN), 7/8-8/8: Πλήρης Overcast (OVC). Έπειτα ακολουθεί τριψήφιος αριθμός ο οποίος αναφέρεται σε εκατοντάδες πόδια και δείχνει σε ποιο ύψος είναι η βάση των συγκεκριμένων νεφών.
- Θερμοκρασία περιβάλλοντος υπό σκιά
- Σημείο δρόσου (είναι η θερμοκρασία κατά την οποία η ατμόσφαιρα είναι 100% κορεσμένη με υγρασία)
- Βαρομετρική πίεση QNH (Q)
- Άλλες πληροφορίες

Συχνά στο METAR συναντούμε το αρτικόλεξο CAVOK που αντιστοιχεί στο ceiling and visibility OK. Αναφέρεται σε συγκεκριμένες καιρικές συνθήκες, δηλαδή ορατότητα περισσότερη από 10 χιλιόμετρα, δεν υπάρχουν νέφη κάτω από τα 5000 πόδια και δεν υπάρχει καιρικό φαινόμενο στην περιοχή.

Παρακάτω βλέπουμε ένα τυπικό δελτίο METAR.

LGIO 231950Z 25015G30KT 9999 DZ FEW002 SCT004 BKN008 30/21Q1018 NOSIG

Αναλύοντας το METAR στα επιμέρους στοιχεία βλέπουμε ότι: αναφέρεται στο αεροδρόμιο των Ιωαννίνων (LGIO), έχει εκδοθεί στις 23 του τρέχοντος μήνα (23), στις 19 και 50 ώρα UTC (1950Z), ο επικρατών άνεμος είναι από 250 μοίρες 15 κόμβους (25015), ριπαίος στους 30 κόμβους (G), ορατότητα περισσότερα από 10 χιλιόμετρα (9999), ψιχαλίζει (DZ), νέφη few στα 200 πόδια (FEW002), νέφη scattered στα 400 πόδια (SCT004), νέφη broken στα 800 πόδια (BKN008), θερμοκρασία 30 βαθμούς Κελσίου - σημείο δρόσου 21 βαθμούς Κελσίου (30/21), βαρομετρική πίεση QNH 1018 hPa (Q1018).

Κεφάλαιο Γ΄

Λογισμικό της Εφαρμογής

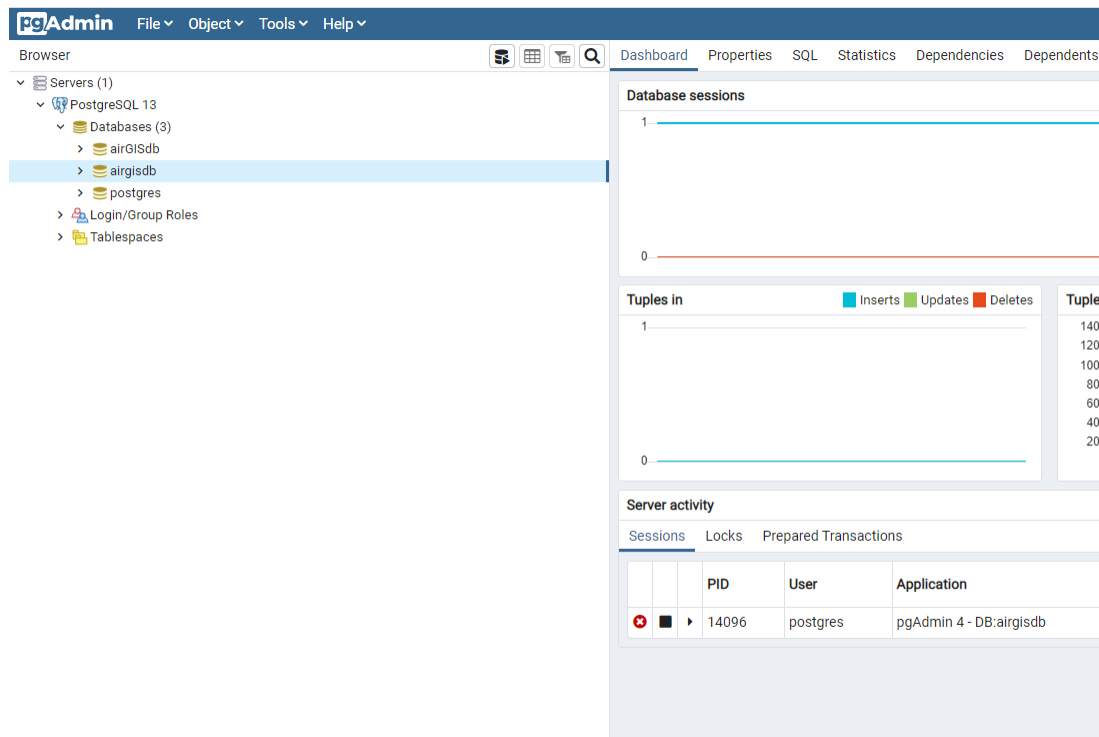
ΠΡΟΧΩΡΩΝΤΑΣ στη συνέχεια, θα περιγράψουμε το λογισμικό που χρησιμοποιήθηκε για την υλοποίηση της εφαρμογής webGIS και τις δυνατότητες που μας προσφέρει και για τις οποίες το επιλέξαμε.

Γ΄.1 Βάση Δεδομένων PostgreSQL

Η βάση δεδομένων που επιλέξαμε να χρησιμοποιήσουμε για την δημιουργία της εφαρμογής μας, είναι η PostgreSQL [27]. Είναι μια σχεσιακή βάση δεδομένων ανοικτού κώδικα (opensource) η οποία υποστηρίζει όλους τους τύπους δεδομένων (numeric, decimal, smallint, integer, bigint, real, double, char, varchar κ.α) που μας ενδιαφέρουν, είναι συμβατή με το σύνολο των ιδιοτήτων της ατομικότητας (atomicity), συνέπειας (consistency), απομόνωσης (isolation), μονιμότητας (durability) (ACID) και επιπλέον υποστηρίζει την επέκταση PostGIS, η οποία θα περιγραφεί στη συνέχεια. Η έκδοση της βάσης δεδομένων που χρησιμοποιούμε είναι η version 13. Ακόμη τρέχει σε όλα τα βασικά λειτουργικά συστήματα, υποστηρίζει πλήρως rollback, commit και διαθέτει γραφικό εργαλείο διαχείρισης, το pgAdmin.

Γ΄.1.1 Επέκταση PostGIS και Εργαλείο Διαχείρισης pgAdmin

Το γραφικό περιβάλλον διαχείρισης της βάσης δεδομένων ανοίγει στον φυλλομετρητή μας. Μέσα από αυτό μπορούμε να κάνουμε διάφορες ενέργειες όπως να δημιουργήσουμε μια βάση δεδομένων, να δημιουργήσουμε πίνακες, να δούμε τα περιεχόμενα της βάσης και των πινάκων της βάσης, να εκτελέσουμε κώδικα σε γλώσσα PL/pgSQL κ.α. Στο παρακάτω σχήμα βλέπουμε την αρχική σελίδα του εργαλείου pgAdmin.



Εικόνα Γ'1: Αρχική σελίδα του εργαλείου pgAdmin

Η επέκταση PostGIS [28] είναι λογισμικό ανοιχτού κώδικα το οποίο μας δίνει την δυνατότητα να δημιουργήσουμε γεωγραφικά αντικείμενα στην postgresQL βάση δεδομένων. Με αυτή μπορούμε να δημιουργήσουμε πίνακες οι οποίοι θα αναπαριστούν γεωμετρικά αντικείμενα όπως σημεία (Points), ευθείες (LineStrings), πολύγωνα (Polygons) κ.α. Ακόμη προσφέρει επιπλέον δυνατότητες όπως υπολογισμό απόστασης μεταξύ δύο γεωγραφικών σημείων, όπως επίσης υπολογισμό εμβαδού και υπολογισμό περιμέτρου μεταξύ περισσότερων σημείων. Επιπλέον προσφέρει τη δυνατότητα πράξεων όπως ένωσης και τομής μεταξύ γεωγραφικών περιοχών και την δυνατότητα δημιουργίας 3D διανυσματικών γεωδαιτικών σχημάτων.

Η εφαρμογή μας θα πρέπει να δημιουργεί πάνω σε έναν χάρτη, γεωμετρικά σχήματα, τα οποία αναπαριστούν στοιχεία όπως waypoints, αεροδιαδρόμους, CTRs, TMAs κ.α. από θεματικούς αεροπορικούς χάρτες. Τα στοιχεία αυτά είναι στην πραγματικότητα γεωμετρικά σχήματα τριών κατηγοριών: σημεία, ευθείες και πολύγωνα. Κάθε τέτοιο στοιχείο έχει κάποιες ιδιότητες, όπως είναι το όνομα, οι γεωγραφικές συντεταγμένες, το FIR της χώρας που ανήκει. Έτσι δημιουργούμε πίνακες των οποίων οι εγγραφές αποτελούν αυτά τα στοιχεία. Στην συνέχεια όπως θα δούμε η γραφική απεικόνισή τους πάνω στον χάρτη επιτυγχάνεται με την μεσολάβηση ενός διακομιστή ο οποίος θα λαμβάνει τα στοιχεία αυτά από την βάση δεδομένων και θα τα μετατρέπει σε γραφικά επίπεδα (layers).

Το εργαλείο pgAdmin διαχείρισης της postgresQL μας επιτρέπει εύκολα να δημιουργήσουμε τη βάση δεδομένων την οποία θα χρησιμοποιήσουμε στην εφαρμογή μας. Στην συνέχεια

δημιουργούμε τον πίνακα που θέλουμε κάθε φορά, χρησιμοποιώντας το εργαλείο Query Tool και τις αντίστοιχες εντολές σε γλώσσα SQL.

Γ΄.2 Διακομιστής Geoserver

Γ΄.2.1 Διακομιστές χαρτών

Το επόμενο βήμα μετά την δημιουργία της βάσης δεδομένων, ήταν η επιλογή του διακομιστή χαρτών (map server), που θα δημιουργεί τα layers με τα οποία θα τροφοδοτεί την εφαρμογή μας. Η ύπαρξη ενός τέτοιου διακομιστή που θα επιτελεί αυτό το έργο είναι απαραίτητη λόγω του ότι η εφαρμογή με αυτό τον τρόπο απαλλάσσεται από το βάρος να δημιουργήσει τα layers η ίδια, κάτι το οποίο θα είχε τεράστια και αρνητική επίδραση στην απόδοση της εφαρμογής. Η βιβλιοθήκη Openlayers που χρησιμοποιούμε στην εφαρμογή μας, έχει την δυνατότητα να δημιουργήσει features όπως ευθείες και σημεία πάνω στον χάρτη. Όμως έπειτα από κάποιο αριθμό features, η επίπτωση στην απόδοση είναι δραματική. Η λύση στο πρόβλημα αυτό την δίνει ένας διακομιστής χαρτών (map server) ο οποίος επιτελεί το έργο να δημιουργεί layers και να τα «σερβίρει» στην εφαρμογή.

Υπάρχουν πολλοί διακομιστές που δημιουργούν layers από γεωχωρικά δεδομένα (geospatial data) και χρησιμοποιούνται σε ανάλογες εφαρμογές. Υπάρχουν κάποιοι επί πληρωμή και κάποιοι που είναι δωρεάν, όπως ο Mapserver [29] και ο Geoserver.[30]

Για την δημιουργία των layers επιλέχθηκε ο Geoserver. Είναι ένας διακομιστής ανοιχτού κώδικα (open-source server) γραμμένος σε γλώσσα Java. Χρησιμοποιεί open standards που έχουν καθοριστεί από το Open Geospatial Consortium (OGC). Οι υπηρεσίες που μπορεί να προσφέρει ο Geoserver είναι οι Web Feature Service (WFS), Web Map Service (WMS), Web Coverage Standard (WCS). [30]

Γ΄.2.2 WMS

Το WMS [31] είναι ένα πρότυπο πρωτόκολλο, το οποίο αναπτύχθηκε από το OGC για την δημιουργία γεωαναφορικών (georeference) εικόνων. Οι εικόνες αυτές μπορούν να συσχετιστούν με ένα γεωχωρικό σύστημα συντεταγμένων. Το πρωτόκολλο WMS χρησιμοποιείται για την δημιουργία εικόνων με δεδομένα που λαμβάνονται από μια βάση δεδομένων.

Στην εφαρμογή μας θα χρησιμοποιήσουμε το WMS για την δημιουργία των layers. Τα δεδομένα θα λαμβάνονται από την βάση δεδομένων postgresQL που δημιουργήσαμε.

Γ'.2.3 Γραφικό Περιβάλλον Διαχείρισης

Ο Geoserver έχει δικό του γραφικό περιβάλλον διαχείρισης που φορτώνεται στον φυλλομετρητή μας. Μέσω του περιβάλλοντος αυτού, μπορούμε να πραγματοποιήσουμε μια σειρά από λειτουργίες, όπως να δούμε την κατάσταση (status) του διακομιστή, να δημιουργήσουμε ένα workspace μέσα στο οποίο θα ανήκουν τα layers μας, να δημιουργήσουμε τα layers που θέλουμε, να κάνουμε επισκόπηση και να δούμε πώς φαίνονται στον χάρτη μας και τέλος να δημιουργήσουμε styles με τα οποία θα «ντύσουμε» τα layers. Στην παρακάτω εικόνα βλέπουμε την αρχική σελίδα από το γραφικό περιβάλλον διαχείρισης του διακομιστή Geoserver.



Εικόνα Γ'2: Αρχική σελίδα του γραφικού περιβάλλοντος διαχείρισης του Geoserver

Γ'.2.4 Τρόπος Λειτουργίας

Ο Geoserver λαμβάνει τα γεωχωρικά δεδομένα από τη βάση δεδομένων PostgreSQL. Όπως είδαμε τα δεδομένα αυτά περιέχονται στους αντίστοιχους πίνακες και οι συντεταγμένες τους είναι αποθηκευμένες σε μορφή που υποστηρίζεται από την επέκταση postGIS. Στη συνέχεια ο Geoserver δημιουργεί τα layers ακριβώς όπως ορίσαμε μέσα από το γραφικό του περιβάλλον. Έπειτα παραδίδει τα layers με το πρωτόκολλο WMS, στην frontend εφαρμογή μας.

Γ'.2.5 Style Layer Descriptor

Ο Geoserver μας δίνει τη δυνατότητα να μορφοποιήσουμε επιπλέον το κάθε σχήμα χρησιμοποιώντας το κατάλληλο style το οποίο «ντύνει» το layer.

Ένα style Styled Layer Descriptor (S.L.D) έχει την μορφή xml αρχείου. Με την αντίστοιχη παραμετροποίηση του αρχείου αυτού μπορούμε να δημιουργήσουμε διαφορετικά styles ανάλογα με τις ανάγκες μας. Έτσι μπορούμε να αλλάξουμε το χρώμα και το πάχος της γραμμής, το χρώμα του γεμίσματος, αν πρόκειται για πολύγωνο, ή κύκλο και επίσης αν θα έχει διαφάνεια. Επίσης αν πρόκειται για σημείο να δημιουργήσουμε ένα σχήμα π.χ έναν κύκλο στην θέση του.

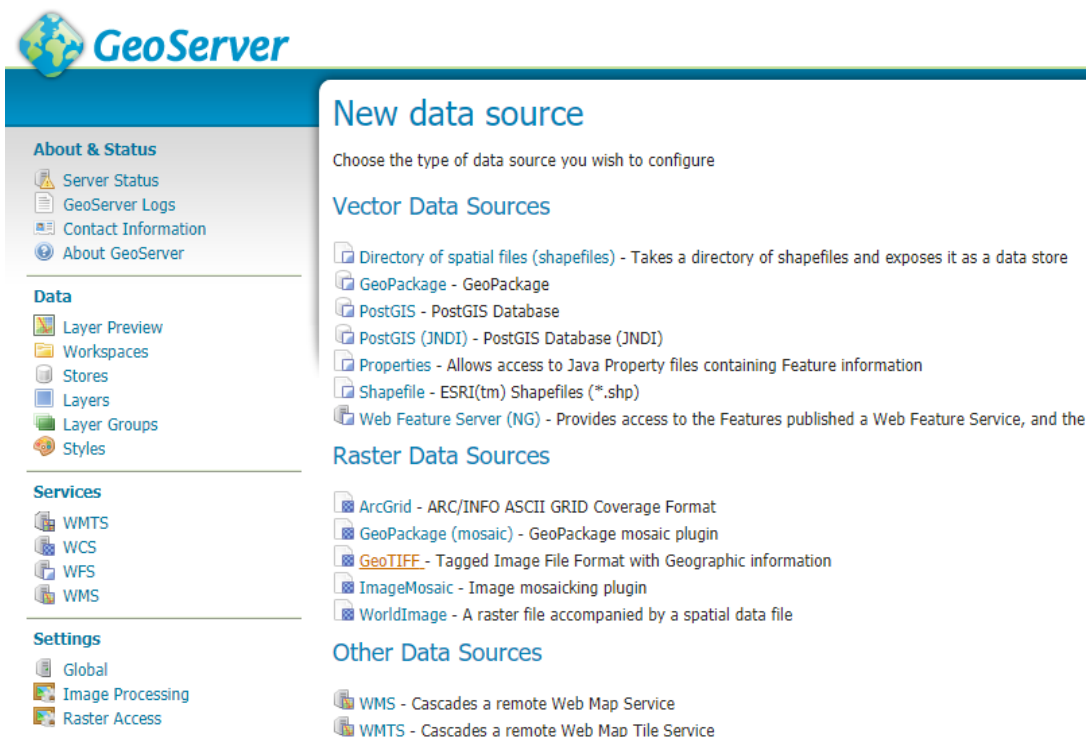
Επίσης παραμετροποιώντας ανάλογα το style, μπορούμε να απεικονίσουμε το όνομα του κάθε στοιχείου, το οποίο μας ενδιαφέρει σημαντικά αφού αυτό χρησιμοποιείται κατά κόρον στους θεματικούς αεροπορικούς χάρτες. Επίσης μπορούμε να καθορίσουμε τη θέση που θα τοποθετηθεί το label σε σχέση με το στοιχείο που αναφέρεται, αν θα έχει κάποιο offset, το χρώμα της γραμματοσειράς, το μέγεθος της γραμματοσειράς. Επιπλέον αν πρόκειται για αεροδιάδρομο, μας ενδιαφέρει πάρα πολύ το label να απεικονίζεται με κλίση παράλληλα με αυτόν. Αυτό επιτυγχάνεται επίσης με την κατάλληλη παραμετροποίηση του style. Ακόμη μας δίνεται η δυνατότητα να καθορίσουμε πόσο συχνά και σε πόση απόσταση θα επαναλαμβάνεται το label ενός αεροδιαδρόμου, στην περίπτωση που αυτός είναι μεγάλος σε μήκος.

Γ'.2.6 Δημιουργία Layer

Για να δημιουργήσουμε ένα layer στο Geoserver θα πρέπει πρώτα να δημιουργήσουμε ένα workspace. Στην συνέχεια, αφού δημιουργήσουμε το workspace, δημιουργούμε ένα νέο store στο οποίο καθορίζουμε το είδος των δεδομένων που θα λαμβάνει ο server μας. Στην συγκεκριμένη περίπτωση τα δεδομένα μας θα είναι Vector Data και θα λαμβάνονται από μια βάση δεδομένων, που περιέχει δεδομένα σύμφωνα με την επέκταση postGIS, όπως φαίνεται στην παρακάτω εικόνα.

Έπειτα καθορίζουμε διάφορα άλλα στοιχεία πώς θα επικοινωνεί ο Geoserver με την βάση δεδομένων, π.χ σε ποιο port, ο αριθμός των συνδέσεων κ.α

Έπειτα δημιουργούμε το layer που επιθυμούμε, δίνοντάς του το κατάλληλο όνομα και καθορίζουμε σε ποιο γεωχωρικό σύστημα συντεταγμένων θα αναφέρεται (στην εφαρμογή μας είναι το EPSG:4326). Στη συνέχεια, ο Geoserver υπολογίζει σύμφωνα με το εύρος των στοιχείων που λαμβάνει από την βάση δεδομένων, τα min X, min Y, max X, max Y, όπου X το γεωγραφικό μήκος και Y το γεωγραφικό πλάτος των δεδομένων εισόδου. Τέλος επιλέγουμε το style που θα «ντύνει» το συγκεκριμένο layer. Αφού κάνουμε όλα αυτά, στην συνέχεια κάνουμε publish το layer.



Εικόνα Γ'3: Καθορισμός του data source, το οποίο θα έχει το layer

Ο Geoserver μας προσφέρει τη δυνατότητα να κάνουμε προεπισκόπηση του layer που δημιουργήσαμε, σε διάφορα format όπως KML, JPEG, PNG, Openlayers.

Γ'.3 Διακομιστής Node.js

Γ'.3.1 Γενικά

Για την διασύνδεση του front end μέρους της εφαρμογής μας με τη βάση δεδομένων PostgreSQL, καθώς και την πρόσβαση της εφαρμογής μας σε εξωτερικούς διακομιστές από όπου η εφαρμογή μας θα λαμβάνει NOTAMs και METARs, επιλέξαμε τη χρήση του node.js server.

Το node.js [32] είναι ένα περιβάλλον ανάπτυξης λογισμικού cross-platform, δηλαδή μπορεί να εκτελεστεί σε διαφορετικές υπολογιστικές πλατφόρμες που χρησιμοποιείται σε εφαρμογές στο επίπεδο πρόσβασης δεδομένων (data access layer) backend. Αναπτύχθηκε το 2009 και είναι γραμμένο σε γλώσσες προγραμματισμού C, C++ και Javascript. Τρέχει πάνω στην V8 μηχανή (Javascript engine) η οποία αναπτύχθηκε για τον φυλλομετρητή Google Chrome. Προσφέρει την δυνατότητα να εκτελεστεί κώδικας javascript στην πλευρά του διακομιστή (server-side scripting) παράγοντας δυναμικές σελίδες web, πριν η σελίδα σταλεί στον φυλλομετρητή του χρήστη. Δεν βασίζεται στην πολυνηματικότητα (multithreading) αλλά στην αρχιτεκτονική event-driven. Λειτουργεί με την χρήση συμβάντων (callbacks) και επιτρέπει με αυτό τον τρόπο την ασύγχρονη επικοινωνία μεταξύ των υπολογιστικών πόρων. Αυτό μας δίνει το πλεονέκτημα ο κώδικάς μας να προχωρήσει στην εκτέλεσή του, χωρίς να χρειάζεται να περιμένει την ολοκλήρωση μιας άλλης λειτουργίας, π.χ την απάντηση από κάποιο διακομιστή σε κάποιο αίτημα.

Γ'.3.2 Βιβλιοθήκες του Node.js

Ένα σημαντικό πλεονέκτημα του node.js είναι η ανάπτυξη παρά πολλών βιβλιοθηκών, οι οποίες βοηθούν τους προγραμματιστές να ενσωματώσουν στις εφαρμογές που αναπτύσουν νέες δυνατότητες, όπως την διασύνδεση του φυλλομετρητή με βάσεις δεδομένων, όπως η mongodb, η postgresSQL, οι οποίες υποστηρίζουν δεδομένα σε μορφή JSON. Η εγκατάσταση των βιβλιοθηκών αυτών γίνεται με την χρήση του διαχειριστή πακέτων (package manager) npm.[32]

Βιβλιοθήκες οι οποίες έχουν αναπτυχθεί από την κοινότητα των προγραμματιστών για το node.js υπάρχουν στην ιστοσελίδα <https://www.npmjs.com/>. Η εφαρμογή μας χρησιμοποιεί τις παρακάτω βιβλιοθήκες node.js

- express.js (express)
Χρησιμεύει στη δημιουργία εφαρμογών web ή APIs. Με τη χρήση της βιβλιοθήκης express μπορούμε να χειριστούμε την όδευση αιτημάτων http [33] [34]
- body-parser

- Πρόκειται για ένα middleware, το οποίο είναι απαιτούμενο από τη βιβλιοθήκη express.js και το οποίο αναλύει (parse) το body και βλέπει αν αυτό πληροί κάποια κριτήρια, το επιστρέφει στους handlers, που έχουμε φτιάξει στον κώδικά μας για να το χειριστούμε παρακάτω.[35]
- @airport-discovery/notams
Χρησιμοποιεί στην ανάκτηση αεροναυτικών αγγελιών NOTAMs από την ιστοσελίδα notams.aim.faa.gov [36]
- @airport-discovery/metars-tafs
Χρησιμοποιεί στην ανάκτηση METARs από την ιστοσελίδα <https://www.aviationweather.gov/> [37]
- pg
Χρησιμοποιεί στη διασύνδεση του node.js με την βάση δεδομένων postgresQL. Υποστηρίζει callbacks, promises, async/await, connection pooling, prepared statements, cursors, streaming results, C/C++ bindings κ.λπ. [38]
- image-to-base64
Χρησιμοποιεί για την μετατροπή μιας εικόνας σε base64 string . Χρησιμοποιούμε αυτή τη βιβλιοθήκη για να μετατρέψουμε τις εικόνες των διαδικασιών S.I.D και ST.AR σε base64 strings, πριν τις αποστείλουμε στο front end μέρος της εφαρμογής μας. [39]
- fs
Χρησιμοποιείται για την πρόσβαση με σύγχρονο ή ασύγχρονο τρόπο στο file system

Γ'.4 Front End

Η σχεδίαση του επιπέδου front end δεν έγινε ανεξάρτητα, από το υπόλοιπο back end μέρος. Η προσέγγισή μας από την αρχή ήταν να αποφασίσουμε ταυτόχρονα για την επιλογή όλων των επιμέρους στοιχείων που θα διαλέξουμε ώστε να εξασφαλίσουμε την καλύτερη διασύνδεση μεταξύ τους πριν την ανάπτυξη της εφαρμογής.

Η επιλογή του στοιχείου για την οπτικοποίηση των δεδομένων πάνω στον χάρτη έγινε με κύριο κριτήριο το υπόβαθρο της εφαρμογής, δηλαδή ότι είναι web application και όχι desktop application. Δυστυχώς για την ανάπτυξη desktop εφαρμογής, οι διαθέσιμες επιλογές βιβλιοθηκών από γνωστές γλώσσες προγραμματισμού είναι εξαιρετικά λίγες και χωρίς πολλές δυνατότητες. Από την άλλη πλευρά, οι επιλογές για web εφαρμογές είναι πάρα πολλές αφού προσφέρονται ως βιβλιοθήκες ή frameworks της γλώσσας Javascript και αντιστοίχως με πολλές δυνατότητες.

Οι κυριότερες διαθέσιμες επιλογές βιβλιοθηκών Javascript που χρησιμοποιούνται σε GIS εφαρμογές είναι οι: Openlayers, Leaflet JS, Jvectormap, Amcharts, MapChart, ArcGIS κ.α.

Η Openlayers, η οποία ήταν η τελική μας επιλογή, είναι μια ανοιχτού κώδικα, βιβλιοθήκη της γλώσσας Javascript που χρησιμοποιείται στη χαρτογράφηση δεδομένων σε φυλλομετρητές web. Έχει δικό της Application Programming Interface (API), υποστηρίζει tiled layers, vector layers και πρότυπα OGC standards όπως το Web Map Service (WMS) και Web Feature Service (WFS) [40][41]

Επίσης στην αρχική της σελίδα διαθέτει πλήρες documentation (API docs), πολλά παραδείγματα για την υλοποίηση διαφόρων δυνατοτήτων που προσφέρει καθώς και tutorials.

Ακόμη στην εφαρμογή μας χρησιμοποιούμε την βιβλιοθήκη ol-ext που αποτελεί επέκταση της Openlayers. Η ol-ext έχει δικό της API και αρκετά παραδείγματα των επιπλέον δυνατοτήτων που προσφέρει.[42]

Κεφάλαιο Δ΄

Αρχιτεκτονική της Εφαρμογής

ΣΤΟ κεφάλαιο αυτό θα παρουσιάσουμε το μπλοκ διάγραμμα και το διάγραμμα ροής δεδομένων της εφαρμογής webGIS, τα μέρη από τα οποία αυτή αποτελείται, το πώς επικοινωνούν μεταξύ τους καθώς και οδηγίες εγκατάστασης του λογισμικού τους

Δ΄.1 Εφαρμογές G.I.S

Η εφαρμογή μας ανήκει στο είδος των Συστημάτων Γεωγραφικών Πληροφοριών (Σ.Γ.Π.), Geographic Information Systems (G.I.S.) [43] τα οποία είναι ολοκληρωμένα συστήματα τα οποία συλλέγουν και αναλύουν γεωχωρικά και γεωγραφικά δεδομένα. Γενικότερα, οι εφαρμογές GIS είναι υπολογιστικά εργαλεία τα οποία επιτρέπουν στους χρήστες να δημιουργήσουν διαδραστικά ερωτήματα (interactive queries), να επεξεργαστούν και να αποθηκεύσουν γεωχωρικά και μη γεωχωρικά δεδομένα, να αναλύσουν την πληροφορία εξόδου και να οπτικοποιήσουν τα αποτελέσματα πάνω σε χάρτες.

Οι εφαρμογές GIS χρησιμοποιούνται σε διάφορους επιστημονικούς κλάδους, όπως στην επιστήμη της Γεωγραφίας, αφού έχουν την δυνατότητα να ενσωματώνουν τα χωρικά με τα περιγραφικά δεδομένα, επιτρέποντας έτσι την διαδικασία της χωρικής ανάλυσης. [43]

Οι εφαρμογές GIS, αποτυπώνουν χωρικά δεδομένα σε σύστημα συντεταγμένων, το οποίο μπορεί να είναι γεωγραφικό, καρτεσιανό ή γεωγραφικό. Όπως είπαμε, τα χωρικά δεδομένα μπορούν να συσχετιστούν και με περιγραφικά δεδομένα. Για παράδειγμα π.χ. διάφορα σημεία που αναπαριστούν περιοχές του πλανήτη συνδέονται με ένα πίνακα που καταγράφει το ετήσιο ύψος βροχής και την μέση θερμοκρασία των περιοχών αυτών.

Η οπτικοποίηση των αποτελεσμάτων και η χαρτογραφική τους απόδοση στις εφαρμογές GIS λαμβάνει χώρα με μεθόδους που ανήκουν σε δύο κατηγορίες: τα ψηφιδωτά γραφικά (raster graphics) και τα διανυσματικά γραφικά (vector graphics).[44][45]

Με τη μέθοδο των ψηφιδωτών γραφικών οι χάρτες αποτελούνται από πλακίδια (tiles) εικόνων. Με τη μέθοδο των διανυσματικών γραφικών οι χάρτες σχηματίζονται από διανύσματα, σημεία, ευθείες και πολύγωνα.

Στα προηγούμενα κεφάλαια, είδαμε το επίπεδο της εφαρμογής το οποίο ανήκει στο back end μέρος (access data layer) και το οποίο αποτελείται από τη βάση δεδομένων PostgreSQL, τον διακομιστή Geoserver και τον διακομιστή node.js. Σε αυτό το κεφάλαιο θα ασχοληθούμε με το front end μέρος της εφαρμογής μας (presentation layer).

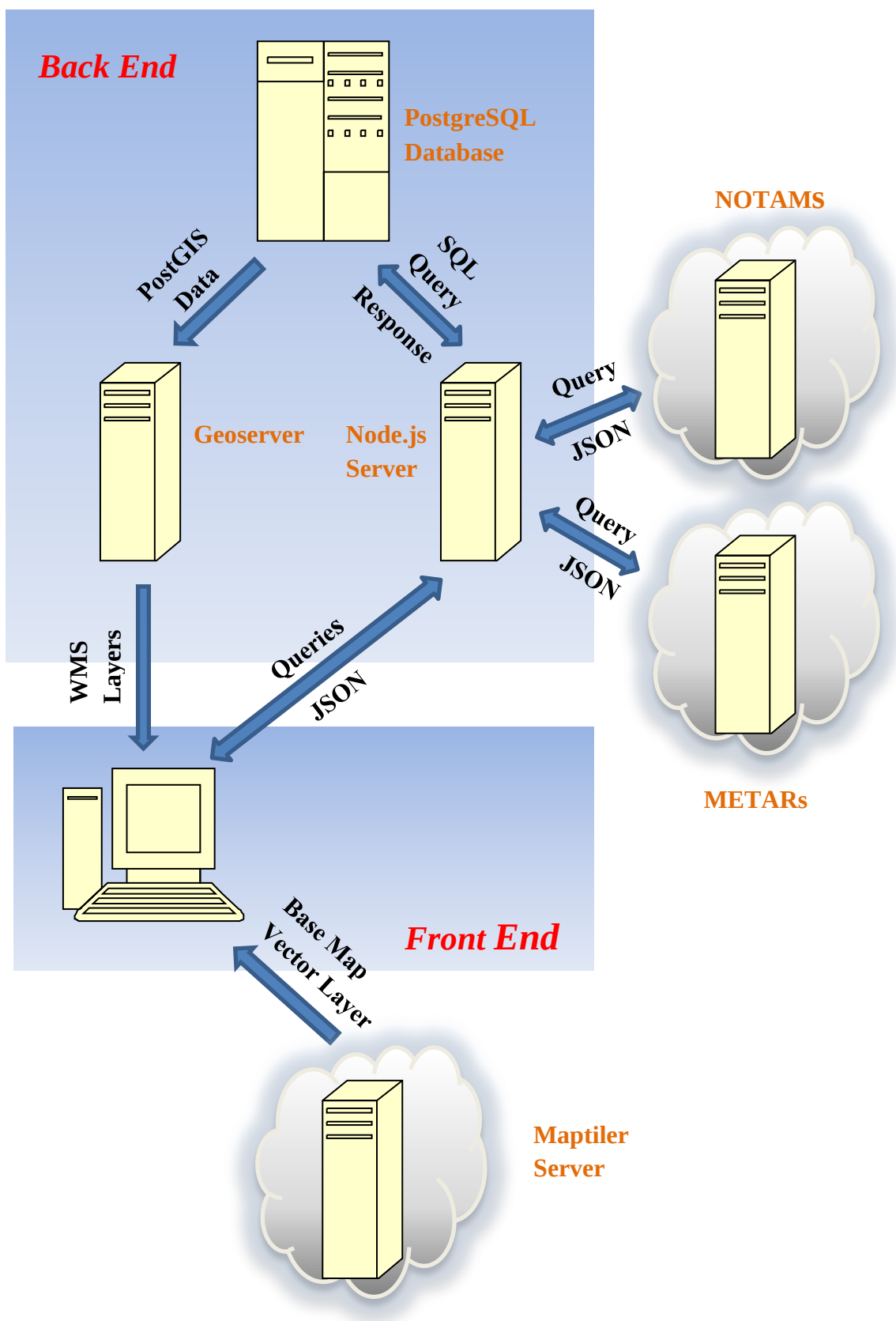
Δ'.2 Γενική Περιγραφή της Εφαρμογής

Η εφαρμογή αποτελείται από το back end και το front end μέρος. Το back end μέρος (data access layer) απαρτίζεται από τη βάση δεδομένων postgresQL, το διακομιστή Geoserver και το διακομιστή node.js. Το front end μέρος αποτελείται από το web μέρος της εφαρμογής, δηλαδή το αρχείο index.html, το αρχείο styles.css και τα αρχεία .js.

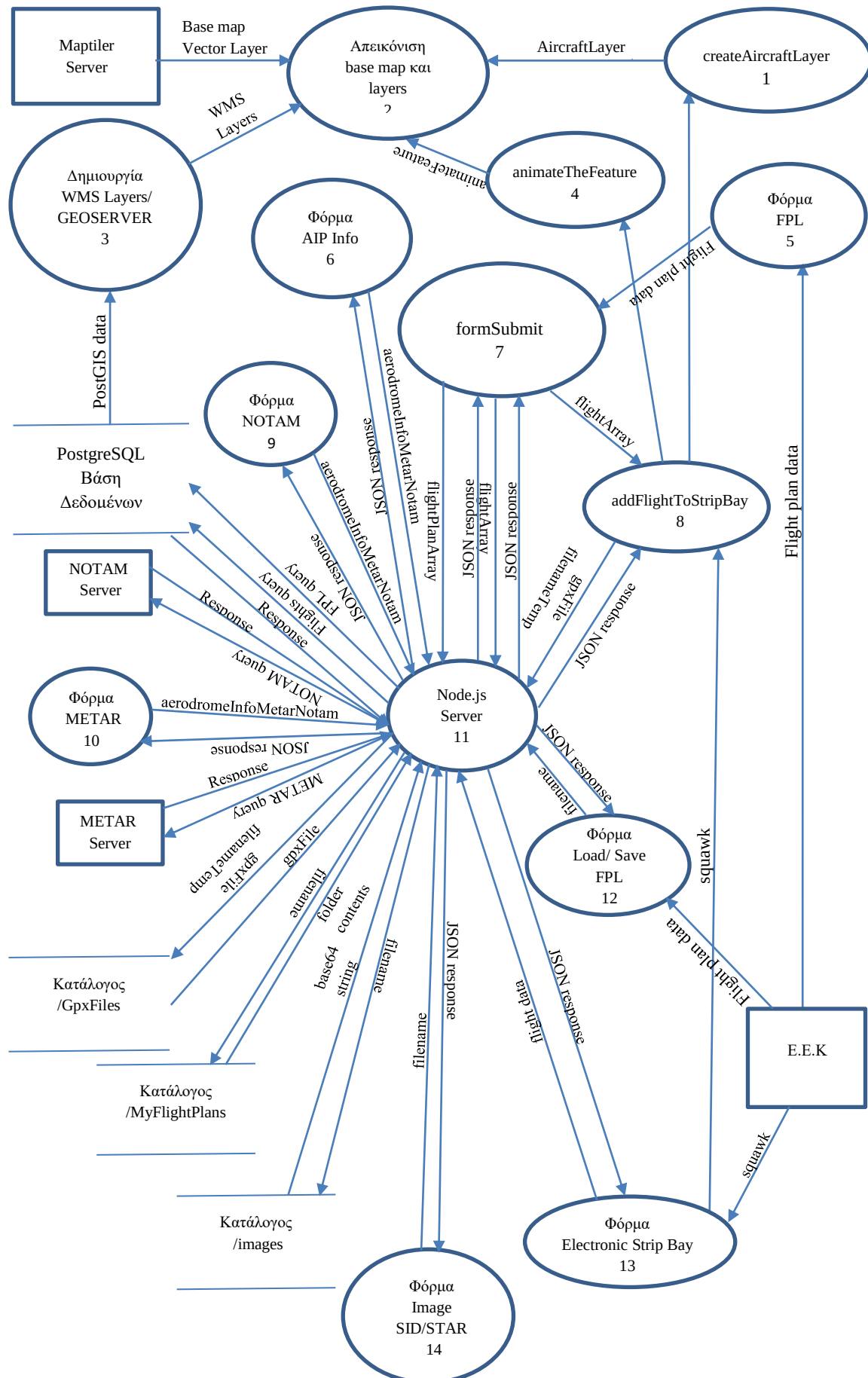
Δ'.3 Block Διάγραμμα και Διάγραμμα Ροής Δεδομένων

Στις δύο επόμενες εικόνες βλέπουμε το Block Διάγραμμα και το Διάγραμμα Ροής Δεδομένων (ΔΡΔ) Data Flow Diagram (DFD), της εφαρμογής webGIS.

Στο Διάγραμμα Ροής Δεδομένων βλέπουμε τις κυριότερες διεργασίες της εφαρμογής καθώς και τα ορίσματα που αποστέλλονται μεταξύ τους. Επίσης βλέπουμε την επικοινωνία της εφαρμογής με τους διακομιστές που βρίσκονται εκτός της εφαρμογής δηλαδή, τον Maptiler, τον NOTAM και METAR server.



Εικόνα Δ΄1: Block Διάγραμμα



Εικόνα Δ'2: Διάγραμμα Ροής Δεδομένων

Δ΄.4 Βάση Δεδομένων PostgreSQL

Δ΄.4.1 Πίνακας Waypoints_Layer

Ο πρώτος πίνακας που επιλέξαμε να δημιουργήσουμε, είναι ο πίνακας waypoints_layer. Ως waypoint ορίζουμε ένα νοητό σημείο στον αεροναυτικό χάρτη που έχει σταθερές γεωγραφικές συντεταγμένες και ένα όνομα. Τα waypoints είναι σημεία αναφοράς που βοηθούν τους πιλότους των αεροσκαφών να προσδιορίσουν με ακρίβεια την γεωγραφική θέση τους πάνω στον χάρτη. Επίσης χρησιμεύουν στο να οριστούν οι αεροδιάδρομοι, οι διαδικασίες απογείωσης Standard Instrument Departure (S.I.D), οι διαδικασίες προσγείωσης Standard Arrival (ST.AR) και τα όρια του Flight Information Region (FIR) κάθε χώρας.

Έχουμε δύο είδη waypoints. Τα waypoints που βρίσκονται στους θεματικούς αεροπορικούς χάρτες Instrument Flight Rules (IFR) και τα waypoints που βρίσκονται σε χάρτες Visual Flight Rules (VFR). Για αυτό τον λόγο στην αρχή δημιουργήθηκαν οι πίνακες waypoints και waypoints_vfr. Ο πρώτος πίνακας περιέχει τα σημεία που βρίσκονται στους IFR χάρτες ενώ ο δεύτερος τα σημεία που βρίσκονται στους VFR χάρτες.

Οι γεωγραφικές συντεταγμένες των waypoints του FIR Αθηνών ανακτήθηκαν από το εγχειρίδιο Αεροναυτικών Πληροφοριών Aeronautical Information Publication Greece (A.I.P). Αντίστοιχα, τα waypoints των FIR Ρώμης, Μπρίντεζι, Μιλάνου ανακτήθηκαν από το αντίστοιχο A.I.P της Ιταλίας. Τα VFR waypoints της Ελλάδας ανακτήθηκαν και αυτά από το A.I.P Greece. Λόγω μεγάλου όγκου των δεδομένων που χρησιμοποιεί η εφαρμογή μας, δεν περάστηκαν στη Βάση Δεδομένων τα VFR waypoints της Ιταλίας.

Ο κάθε πίνακας που περιέχει τα waypoints, έχει τις παρακάτω ιδιότητες: label που αντιστοιχεί στο όνομα του σημείου (το οποίο αποτελείται από 5 γράμματα για τα ifr waypoints και περισσότερα από 5 για τα vfr waypoints), coordinates που αντιστοιχεί στις γεωγραφικές συντεταγμένες του σημείου και fir που αντιστοιχεί στο FIR στο οποίο ανήκει το σημείο. Οι συντεταγμένες είναι αποθηκευμένες σύμφωνα με την επέκταση postGIS. Η σύνταξη της εντολής SQL που αποθηκεύει ένα σημείο στον αεροναυτικό χάρτη με όνομα name και συντεταγμένες longitude και latitude, είναι:

```
INSERT INTO WAYPOINTS_LAYER (LABEL, COORDINATES, FIR) VALUES  
(name, ST_SetSRID (ST_MakePoint (longitude, latitude), 4326), fir)
```

Η συνάρτηση ST_MakePoint της επέκτασης postGIS δημιουργεί ένα σημείο, λαμβάνοντας δύο ορίσματα που αντιστοιχούν στις συντεταγμένες του σημείου, γεωγραφικό μήκος και γεωγραφικό πλάτος, αντίστοιχα.

Η συνάρτηση ST_SetSRID χρησιμεύει για να προσδιορίσουμε σε ποιο γεωγραφικό σύστημα αναφοράς συντεταγμένων θα ανήκει το σημείο αυτό. Η εφαρμογή μας χρησιμοποιεί το γεωγραφικό σύστημα αναφοράς συντεταγμένων WGS 1984, EPSG:4326. Η συνάρτηση ST_SetSRID δέχεται δύο ορίσματα. Το πρώτο όρισμα είναι το σημείο που επιστρέφει η συνάρτηση ST_MakePoint και το δεύτερο όρισμα είναι ο αριθμός που αντιστοιχεί στο γεωγραφικό σύστημα αναφοράς που χρησιμοποιούμε.

Για παράδειγμα για το σημείο NEMES που βρίσκεται στην περιοχή της Νεμέας, η σύνταξη της εντολής με την οποία θα αποθηκευτεί στον πίνακα waypoints στη Βάση Δεδομένων είναι:

```
INSERT INTO WAYPOINTS_LAYER (LABEL, COORDINATES, FIR) VALUES  
('NEMES', ST_SetSRID (ST_MakePoint (22.5808, 37.7064), 4326), 'GREECE')
```

Συνολικά ο πίνακας waypoints_layer περιλαμβάνει 940 σημεία ενώ ο πίνακας waypoints_vfr_layer περιλαμβάνει 270 σημεία.

Δ'4.2 Πίνακας Nav aids_Layer

Στους θεματικούς αεροπορικούς χάρτες το waypoint αποτελεί βασικό στοιχείο. Ένα ακόμα βασικό στοιχείο στους χάρτες αυτούς είναι το ραδιοβοήθημα (navaid). Η συσκευή αυτή βοηθά ένα αεροσκάφος με ενεργητικό τρόπο να προσδιορίσει την θέση του στον χάρτη. Αποτελείται από ένα πομπό, μια κεραία ή συστοιχίες κεραιών, με τις οποίες εκπέμπεται ηλεκτρομαγνητικό σήμα προς ένα αεροπλάνο. Το σήμα αυτό περιλαμβάνει το αναγνωριστικό όνομα του ραδιοβοηθήματος, το οποίο συνήθως αποτελείται από τρία γράμματα (π.χ KRK για το ραδιοβοήθημα που βρίσκεται στην περιοχή Λευκίμμη της Κέρκυρας).

Η αποθήκευση των ραδιοβοηθημάτων στη βάση δεδομένων επιτυγχάνεται με την δημιουργία του πίνακα nav aids_layer. Στην πραγματικότητα ένα ραδιοβοήθημα απεικονίζεται στον χάρτη με τον ίδιο τρόπο όπως ένα waypoint, δηλαδή ως ένα σημείο. Γι αυτό τον λόγο ο πίνακας nav aids_layer έχει τις ίδιες ιδιότητες με τον πίνακα waypoints_layer, δηλαδή label, coordinates, fir. Οι συντεταγμένες του ραδιοβοηθήματος είναι αποθηκευμένες σύμφωνα με την επέκταση postGIS.

Παρόμοια η σύνταξη της εντολής SQL για την αποθήκευση στον πίνακα nav aids_layer ενός ραδιοβοηθήματος με όνομα name και συντεταγμένες longitude και latitude, είναι:

```
INSERT INTO NAVAIDS_LAYER (LABEL, COORDINATES, FIR) VALUES
(name, ST_SetSRID (ST_MakePoint (longitude, latitude), 4326), fir)
```

Για παράδειγμα για το ραδιοβοήθημα V.O.R που βρίσκεται εντός του αεροδρομίου της Κέρκυρας με όνομα GAR, η σύνταξη της εντολής με την οποία θα αποθηκευτεί στον πίνακα waypoints στη Βάση Δεδομένων είναι:

```
INSERT INTO NAVAIDS_LAYER (LABEL, COORDINATES, FIR) VALUES
('GAR', ST_SetSRID (ST_MakePoint (19.9094, 39.6064), 4326), 'GREECE')
```

Συνολικά ο πίνακας nav aids_layer περιλαμβάνει 217 ραδιοβοηθήματα.

Δ΄.4.3 Πίνακας Aerodromes_Layer

Στην συνέχεια δημιουργήθηκε ο πίνακας aerodromes_layer ο οποίος περιέχει 41 αεροδρόμια της Ελλάδος και 224 της Ιταλίας. Ένα αεροδρόμιο αναπαρίσταται στους θεματικούς αεροπορικούς χάρτες ως σημείο με συγκεκριμένες συντεταγμένες που είναι δημοσιευμένες στο εγχειρίδιο AIP της χώρας του αεροδρομίου. Στην εφαρμογή μας, οι συντεταγμένες ανακτήθηκαν από το AIP Ελλάδας και AIP Ιταλίας. Ο πίνακας aerodromes_layers έχει τις ιδιότητες label που αντιστοιχεί στο όνομα του αεροδρομίου όπως αυτό έχει οριστεί κατά ICAO και έχει μήκος 4 γράμματα, coordinates που αντιστοιχεί στις γεωγραφικές συντεταγμένες του αεροδρομίου δηλαδή στο γεωγραφικό πλάτος και γεωγραφικό μήκος σύμφωνα με την επέκταση του postGIS και fir που αντιστοιχεί στην χώρα που ανήκει το αεροδρόμιο.

Δ΄.4.4 Πίνακας Airways

Έχοντας φτιάξει τους πίνακες με τα waypoints ifr, waypoints vfr, ραδιοβοηθήματα nav aids και αεροδρόμια μπορούμε να προχωρήσουμε στο επόμενο σημαντικό στοιχείο των αεροπορικών θεματικών χαρτών, που είναι οι αεροδιάδρομοι. Οι αεροδιάδρομοι είναι νοητές ευθείες πάνω στον χάρτη οι οποίες απεικονίζουν πορείες που πρέπει να ακολουθήσουν τα αεροσκάφη όταν θέλουν να πετάξουν από ένα μέρος σε ένα άλλο. Ένας αεροδιάδρομος ορίζεται όπως μια ευθεία, ορίζεται δηλαδή από δύο σημεία. Τα σημεία αυτά μπορεί να είναι δύο waypoints, δύο ραδιοβοηθήματα nav aids ή συνδυασμός και των δύο. Επίσης ένας αεροδιάδρομος έχει ένα όνομα που αποτελείται από ένα γράμμα και έναν αριθμό. Δεν είναι απαραίτητο, ο αεροδιάδρομος να είναι συνεχόμενος από ένα αρχικό waypoint σε ένα τελικό waypoint. Πολλές φορές ο αεροδιάδρομος μπορεί να αποτελείται από μη συνεχόμενα τμήματα.

Η επεξεργασία των αεροδιαδρόμων έγινε με μη αυτοματοποιημένο τρόπο, καθώς δεν υπάρχουν ψηφιακά αποτυπωμένοι στο A.I.P με μορφή δεδομένων, παρά μόνο σε θεματικούς αεροπορικούς IFR χάρτες σε αρχεία pdf . Τα εγχειρίδια που χρησιμοποιήθηκαν είναι τα AIP Greece October 2020 και AIP Italy November 2020. Τα σημεία που ορίζουν έναν αεροδιάδρομο δεν υπάρχουν σε ηλεκτρονική ή έντυπη μορφή όπως υπάρχουν άλλα στοιχεία π.χ τα waypoints. Η διαδικασία που ακολουθήθηκε, ξεκίνησε αρχικά με την καταγραφή του κάθε αεροδιαδρόμου από θεματικό αεροναυτικό χάρτη των σημείων waypoints ή σημείων ραδιοβοηθημάτων nav aids από τα οποία αποτελείται. Έπειτα, δημιουργήθηκε ο βοηθητικός πίνακας airways, στην βάση δεδομένων, ο οποίος εκτός από τα σημεία που έχει ο κάθε αεροδιάδρομος, προστέθηκε επιπλέον η ιδιότητα segment, η οποία προσδιορίζει το τμήμα του κάθε αεροδιαδρόμου, στο οποίο αντιστοιχεί το συγκεκριμένο σημείο. Έτσι κατέστη εφικτό, ένας διάδρομος να έχει ξεχωριστά τμήματα, σε διαφορετικά μέρη του χάρτη.

Συνολικά ο πίνακας airways περιλαμβάνει στο σύνολο 342 segments, δηλαδή 342 τμήματα αεροδιαδρόμων και τα οποία αντιστοιχούν σε 1812 σημεία waypoints και σημεία nav aids.

Λόγω του μεγάλου όγκου των αεροδιαδρόμων που υπάρχουν στο FIR μιας χώρας δημιουργήθηκαν μόνο οι χάρτες της Ελλάδας και της Ιταλίας.

Δ'.4.5 Πίνακας Airways_Layer

Στη συνέχεια, δημιουργήθηκε στη βάση δεδομένων ο πίνακας airways_layer, του οποίου η κάθε εγγραφή αντιστοιχεί σε ένα τμήμα αεροδιαδρόμου που ενώνει δύο σημεία (waypoints ή nav aids σημεία). Η ιδιότητα coordinates του πίνακα airways_layer αποθηκεύει τις συντεταγμένες σύμφωνα με την επέκταση postGIS .

Η συνάρτηση ST_MakeLine της επέκτασης postGIS δέχεται δύο ορίσματα. Κάθε όρισμα είναι η τιμή που επιστρέφει η συνάρτηση ST_MakePoint και αντιστοιχεί σε ένα σημείο, όπως είδαμε στην δημιουργία του πίνακα waypoints_layer.

Η σύνταξη της εντολής SQL postGIS για ευθύγραμμο τμήμα που ενώνει δύο σημεία στον χάρτη είναι :

```
INSERT INTO AIRWAYS_LAYER (LABEL, COORDINATES) VALUES
(name, ST_MakeLine (ST_SetSRID (ST_MakePoint (longitude_pointA,
latitude_pointA), 4326), ST_SetSRID (ST_MakePoint (longitude_pointB,
latitude_pointB), 4326)))
```

Στο παράδειγμά μας, για το τμήμα του αεροδιαδρόμου A14 που ενώνει τα σημεία KRK και MALED είναι:

```
INSERT INTO AIRWAYS_LAYER (LABEL, COORDINATES) VALUES
('A14', ST_MakeLine (ST_SetSRID (ST_MakePoint (20.0706, 39.4431), 4326),
ST_SetSRID (ST_MakePoint (20.3033, 39.2208), 4326)))
```

Ο πίνακας `airways_layer` περιλαμβάνει συνολικά 1469 τμήματα αεροδιαδρόμων.

Δ΄.4.6 Πίνακας VFR_Layer

Στην συνέχεια ο πίνακας που δημιουργήσαμε είναι ο πίνακας `vfr_layer`. Η Περιοχή Πληροφοριών Πτήσης FIR (Flight Information Region) είναι μια περιοχή εναέριου χώρου στην οποία παρέχεται υπηρεσία Πληροφοριών Πτήσεων (Flight Information Service) για κάθε αεροσκάφος που πετά εντός της περιοχής αυτής.

Κάθε περιοχή FIR έχει ένα όνομα που αποτελείται από 4 γράμματα και το οποίο έχει δοθεί από τον Διεθνή Οργανισμό Πολιτικής Αεροπορίας ICAO. Για παράδειγμα η Ελλάδα έχει το ATHINA FIR ενώ η Ιταλία τα FIR Μπριντιζίου, Ρώμης, Μιλάνου. Τα αντίστοιχα ονόματα είναι LGGG, LIBB, LIRR, LIMM.

Η απεικόνιση των συγκεκριμένων περιοχών στον χάρτη γίνεται με την δημιουργία του πίνακα `vfr_layer`. Ο πίνακας αυτός έχει τις ιδιότητες `label`, που αντιστοιχεί στο όνομα του FIR, `coordinates`, που αντιστοιχεί στις συντεταγμένες της περιοχής και `fir` που αντιστοιχεί στη χώρα στην οποία ανήκει.

Οι συντεταγμένες είναι αποθηκευμένες σύμφωνα με την επέκταση `postGIS` και ορίζουν ένα πολύγωνο στο οποίο αντιστοιχεί η συγκεκριμένη περιοχή FIR.

Η σύνταξη της εντολής SQL για την αποθήκευση ενός πολυγώνου που αποτελείται από σημεία με συντεταγμένες $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ σύμφωνα με την επέκταση `postGIS` είναι:

```
INSERT INTO VFR_LAYER (LABEL, COORDINATES, FIR) VALUES
(label, ST_Polygon ('LINESTRING (longitude_x1 latitude_y1, longitude_x2
latitude_y2,...,longitude_xn latitude_yn) ::geometry,4326), fir)
```

Για παράδειγμα για το FIC Μπριντιζίου, η εντολή είναι :

```
INSERT INTO VFR_LAYER (LABEL, COORDINATES, FIR) VALUES
('BRINDISI FIR LIBB', ST_Polygon ('LINESTRING (19.0000 38.8856, 19.0000
```

```

39.2933, 19.0000 39.8167, 19.0000 40.0567, 19.0000 40.2447, 19.0000
40.5983, 19.0000 40.7572, 18.9517 40.8917, 18.8667 41.1336, 18.1292
41.5028, 17.2483 41.9283, 16.0489 42.5078, 15.7483 42.7039, 15.4375
42.9075, 14.7978 43.3117, 14.5069 43.5022, 13.3417 43.5192, 13.0853
42.9183, 13.5956 42.3489, 13.7375 42.1889, 14.3844 41.4511, 14.8069
41.3067, 15.1275 41.2017, 15.1714 41.1078, 15.4172 40.6906, 15.6186
40.3444, 15.8075 40.0161, 16.0392 39.6150 16.2544 39.2408, 16.3261
39.1622, 16.5272 38.9844, 17.1725 38.8850, 19.0000 38.8856)' ::
geometry, 4326), 'ITALY')

```

Δ'4.7 Πίνακες LGC_Layer, LGD_Layer και LGM_Layer

Ο επόμενος πίνακας στη βάση δεδομένων είναι ο πίνακας lgc_layer. Ο πίνακας αυτός περιέχει τις ελεγχόμενες περιοχές. Ελεγχόμενες περιοχές είναι δεσμευμένες περιοχές στις οποίες απαγορεύεται η πτήση αεροσκαφών άνευ αδείας. Οι περιοχές αυτές ορίζονται από σταθερά σημεία και έχουν την μορφή πολυγώνων. Κάθε χώρα έχει αναρτημένες τις περιοχές αυτές στο δικό της εγχειρίδιο αεροναυτικών πληροφοριών AIP.

Ο πίνακας lgc_layer έχει τις ιδιότητες label που αντιστοιχεί στο όνομα της περιοχής, coordinates που αντιστοιχεί στις συντεταγμένες της περιοχής και fir που αντιστοιχεί στο FIR της χώρας την οποία ανήκει.

Οι συντεταγμένες είναι αποθηκευμένες σύμφωνα με την επέκταση postGIS. Η σύνταξη της εντολής SQL είναι ίδια όπως και με τις περιοχές FIR.

Για παράδειγμα για την ελεγχόμενη περιοχή LGC-SIDARI η οποία βρίσκεται στα βόρεια του νησιού της Κέρκυρας είναι:

```

INSERT INTO LGC_LAYER (LABEL, COORDINATES, FIR) VALUES
('LGC-SIDARI', ST_Polygon ('LINESTRING (19.6253 39.7672, 19.8498
39.8337, 19.8663 39.8172, 19.7250 39.7175, 19.6253 39.7672)' ::
geometry, 4326), 'GREECE')

```

Οι πίνακες LGR_layer, LGD_layer, LGM_layer περιέχουν αντίστοιχα τις απαγορευμένες περιοχές LGR, τις επικίνδυνες περιοχές LGD και τις στρατιωτικές περιοχές LGM. Οι απαγορευμένες περιοχές LGR, επικίνδυνες περιοχές LGD και στρατιωτικές περιοχές LGM απεικονίζονται παρόμοια όπως και οι ελεγχόμενες περιοχές. Είναι και αυτές πολύγωνα και οι συντεταγμένες τους αποθηκεύονται με τον ίδιο τρόπο όπως οι ελεγχόμενες περιοχές στον πίνακα LGC.

Δ΄.4.8 Πίνακας MTMA_Layer

Στον πίνακα mtma_layer περιέχονται οι τερματικές περιοχές. Τερματικές περιοχές ονομάζονται οι περιοχές οι οποίες περικλείουν ένα αεροδρόμιο και αποτελούν την περιοχή προσέγγισής του. Κάθε αεροσκάφος που θα εισέλθει στην περιοχή αυτή οφείλει να επικοινωνήσει με τον ελεγκτή προσέγγισης που βρίσκεται στο αεροδρόμιο αυτό, ώστε να λάβει οδηγίες για την πορεία που θα πρέπει να ακολουθήσει στη συνέχεια. Η περιοχή αυτή είναι ένα πολύγωνο, το οποίο νοητά εκτείνεται προς τα πάνω μέχρι κάποιο συγκεκριμένο ύψος. Αν η τερματική περιοχή ανήκει σε πολιτικό αεροδρόμιο ονομάζεται απλά TMA, αν ανήκει σε στρατιωτικό ονομάζεται MTMA. Εδώ για λόγους συντομίας δημιουργήθηκε ο πίνακας MTMA_layer στον οποίο περιλαμβάνονται οι στρατιωτικές MTMA και πολιτικές TMA τερματικές περιοχές μαζί. Ο πίνακας MTMA_layer περιλαμβάνει 27 εγγραφές όσες είναι και το πλήθος των τερματικών περιοχών της Ελλάδας.

Κάθε εγγραφή έχει ιδιότητες όνομα που αντιστοιχεί στο όνομα της Τερματικής Περιοχής καθώς και το ύψος μέχρι το οποίο εκτείνεται η περιοχή αυτή, συντεταγμένες και το FIR της χώρας στην οποία ανήκει. Οι συντεταγμένες είναι αποθηκευμένες σύμφωνα με το πρότυπο postGIS. Η εντολή για την αποθήκευση των συνταγμένων είναι παρόμοια όπως και των Ελεγχόμενων περιοχών LGC.

Για παράδειγμα για την MTMA Νότιος Τομέας του αεροδρομίου της Ανδραβίδας η εντολή αποθήκευσης είναι:

```
INSERT INTO MTMA_LAYER (LABEL, COORDINATES, FIR) VALUES
('ANDRAVIDA MTMA SOUTH SECTOR UNL 1000 AGL', ST_Polygon ('LINESTRING
(19.6917 38.1500, 21.2758 37.9333, 21.9813 37.4583, 21.6333 37.4167,
21.3167 36.6833, 19.6667 37.0000, 19.6667 38.0000, 19.6917 38.1500)' ::
geometry, 4326), 'GREECE')
```

Δ΄.4.9 Πίνακας CTR_Layer

Στον πίνακα CTR_layer περιέχονται οι περιοχές CTR. Η περιοχή CTR (Controlled Traffic Region) είναι η περιοχή γύρω από ένα αεροδρόμιο, την οποία διέρχονται συνήθως όσα αεροσκάφη πρόκειται να προσγειωθούν ή να απογειωθούν από αυτό. Έχει συνήθως το σχήμα κύκλου ή πολυγώνου. Η περιοχή CTR εκτείνεται ως ένα ορισμένο ύψος, συγκεκριμένο για κάθε αεροδρόμιο. Όσα αεροσκάφη βρίσκονται εντός της CTR περιοχής επικοινωνούν με τον Πύργο Ελέγχου του αεροδρομίου και λαμβάνουν οδηγίες

Για την απεικόνιση της CTR περιοχής δημιουργήθηκε ο πίνακας CTR_layer του οποίου κάθε εγγραφή έχει τις ιδιότητες label, συντεταγμένες και FIR της χώρας στην οποία ανήκουν. Οι συντεταγμένες είναι αποθηκευμένες σύμφωνα με την επέκταση postGIS. Μια CTR έχει συνήθως το σχήμα του κύκλου ενώ κάποιες φορές μπορεί να είναι ένα πολύγωνο π.χ όταν εφάπτεται των εξωτερικών συνόρων του ελληνικού FIR.

Η σύνταξη SQL της εντολής αποθήκευσης όταν πρόκειται για πολύγωνο είναι ίδια με αυτές που έχουμε δει μέχρι τώρα. Όταν πρόκειται για κύκλο με ακτίνα radius, ενός αεροδρομίου με συντεταγμένες longitude και latitude τότε είναι:

```
INSERT INTO CTR_LAYER (LABEL, COORDINATES, FIR) VALUES
(label, ST_Buffer (ST_GeomFromGeoJSON ('{"type" : "Point", "coordinates"
: [longitude, latitude'] ) :: geography, radius, 'quad_segs = 64') ::
geometry, 'GREECE')
```

Για παράδειγμα για την CTR περιοχή του αεροδρομίου της Σούδας είναι:

```
INSERT INTO CTR_LAYER (LABEL, COORDINATES, FIR) VALUES
('LGSA', ST_Buffer (ST_GeomFromGeoJSON ('{"type" : "Point",
"coordinates" : [24.1511, 35.5314 ]}') :: geography, 18600, 'quad_segs
= 64') :: geometry, 'GREECE')
```

Δ'4.10 Πίνακας FPL

Ο επόμενος πίνακας που περιλαμβάνει η βάση δεδομένων μας είναι ο πίνακας FPL και δημιουργήθηκε για τον χειρισμό των σχεδίων πτήσης που κατατίθενται από τον ελεγκτή εναέριας κυκλοφορίας. Στην επόμενη εικόνα βλέπουμε το πρότυπο έντυπο κατά ICAO, για την κατάθεση ενός σχεδίου πτήσης και τα αντίστοιχα πεδία που περιλαμβάνει.

**Department of Transportation
Federal Aviation Administration**

International Flight Plan

PRIORITY <<= **FF** → **ADDRESSEE(S)**

FILING TIME → **ORIGINATOR** <<=

SPECIFIC IDENTIFICATION OF ADDRESSEE(S) AND/OR ORIGINATOR

3 MESSAGE <<= **(FPL** **7. AIRCRAFT IDENTIFICATION** **8 FLIGHT RULES** **TYPE OF FLIGHT** <<=

9. NUMBER **TYPE OF AIRCRAFT** **WAKE TURBULENCE CAT.** **10 EQUIPMENT** <<=

13 DEPARTURE AERODROME **TIME** <<=

15 CRUISING SPEED **LEVEL** **ROUTE**

16. DESTINATION AERODROME **TOTAL EET** **ALTN AERODROME** **2ND. ALTN AERODROME** <<=

18 OTHER INFORMATION

SUPPLEMENTARY INFORMATION (NOT TO BE TRANSMITTED IN FPL MESSAGES)

19 **ENDURANCE** **PERSONS ON BOARD** **EMERGENCY RADIO**

E/ **HR. MIN.** → **P/** **UHF** **VHF** **ELDA**

SURVIVAL EQUIPMENT → **S/** **P** **D** **M** **J** → **J/** **L** **F** **U** **V**

DINGHIES **NUMBER** **CAPACITY** **COVER** **COLOUR** <<=

A/ **REMARKS** <<=

C/ **PILOT-IN-COMMAND** <<=

FILED BY **ACCEPTED BY** **ADDITIONAL INFORMATION**

Εικόνα Δ΄3: Πρότυπο έντυπο κατά ICAO ενός σχεδίου πτήσης

Ο πίνακας FPL έχει ιδιότητες όλα τα πεδία που έχει μια φόρμα Flight Plan κατά το πρότυπο του ICAO.

Οι ιδιότητες του πίνακα είναι:

- aircraftID Χαρακτηριστικό κλήσης του αεροσκάφους
- flightRules Κανόνες πτήσης
- typeOfFlight Είδος πτήσης
- number Αριθμός αεροσκαφών

• typeOfAircraft	Τύπος αεροσκάφους
• wakeTurbulenceCat	Κατηγορία Αεροδινών
• equipment	Εξοπλισμός
• departureAerodrome	Αεροδρόμιο αναχώρησης
• time	Ώρα αναχώρησης
• cruisingSpeed	Ταχύτητα πτήσης
• level	Ύψος ή Επίπεδο πτήσης
• route	Διαδρομή
• destinationAerodrome	Αεροδρόμιο προορισμού
• totalEET	Συνολικός χρόνος πτήσης
• alternateAerodrome	Εναλλακτικό αεροδρόμιο
• secondAlternateAerodrome	Δεύτερο εναλλακτικό αεροδρόμιο
• otherInformation	Άλλες πληροφορίες
• endurance	Διάρκεια καυσίμων (εκφρασμένα σε χρονική διάρκεια)
• personsOnBoard	Αριθμός επιβαινόντων (πλήρωμα και επιβάτες)
• emergencyRadioUHF	Συσκευές επικοινωνίας UHF
• emergencyRadioVHF	Συσκευές επικοινωνίας VHF
• emergencyRadioELBA	Συσκευές επικοινωνίας ELT
• dinghies	Λέμβοι
• polar	Πολικές
• desert	Ερήμου
• maritime	Θάλασσας
• jungle	Ζούγκλας
• jackets	Σωσίβια
• light	Σωσίβια
• floures	Σωσίβια
• UHF	Σωσίβια
• VHF	Σωσίβια
• DinghiesSecond	(Δεν χρησιμοποιείται)
• numberOfDinghies	Αριθμός Λεμβών
• capacity	Χωρητικότητα λέμβου
• cover	Κάλυμα λέμβου
• color	Χρώμα λέμβου
• aircraftColorAndMarkings	Χρώμα αεροσκάφους
• remarks	Παρατηρήσεις

- pilotInCommand Ονοματεπώνυμο του Κυβερνήτη

Στον πίνακα FPL αποθηκεύονται τα σχέδια πτήσης που γίνονται submit από την φόρμα FPL. Ένα σχέδιο πτήσης μόλις γραφτεί ή φορτωθεί στην φόρμα FPL, και γίνει submit από τον χρήστη στέλνεται στην βάση δεδομένων μέσω του διακομιστή node.js ο οποίος το λαμβάνει από την εφαρμογή.

Δ΄.4.11 Πίνακας FLIGHTS

Ο επόμενος πίνακας της βάσης δεδομένων είναι ο πίνακας flights. Στον πίνακα αυτό αποθηκεύονται όλες οι πτήσεις οι οποίες είναι έτοιμες να απογειωθούν. Αφορά δηλαδή τα σχέδια πτήσης που είναι έτοιμα να ενεργοποιηθούν από τον ελεγκτή. Η ενεργοποίηση της πτήσης γίνεται μόλις ο ελεγκτής λάβει το squawk από το Κέντρο Ελέγχου Περιοχής Αθηνών (Κ.Ε.Π.ΑΘ) και το συμπληρώσει, τότε μπορεί να ενεργοποιήσει την πτήση με την απογείωσή της. Όταν μια πτήση τελειώσει, αφαιρείται η αντίστοιχη εγγραφή της από τον πίνακα flights. Η αποθήκευση του ενεργού σχεδίου πτήσης στον πίνακα flights και η διαγραφή του από τον πίνακα flights μόλις αυτό «κλείσει» (προσγειωθεί το αεροσκάφος) γίνεται μέσω του διακομιστή node.js.

Ο πίνακας flights έχει τις παρακάτω ιδιότητες:

- | | |
|------------------------|---------------------------------------|
| • aircraftID | Χαρακτηριστικό κλήσης του αεροσκάφους |
| • flightRules | Κανόνες πτήσης |
| • typeOfFlight | Είδος πτήσης |
| • number | Αριθμός αεροσκαφών |
| • typeOfAircraft | Τύπος αεροσκάφους |
| • wakeTurbulenceCat | Κατηγορία Αεροδινών |
| • equipment | Εξοπλισμός |
| • departureAerodrome | Αεροδρόμιο αναχώρησης |
| • time | Ωρα αναχώρησης |
| • cruisingSpeed | Ταχύτητα πτήσης |
| • level | Ύψος ή Επίπεδο πτήσης |
| • route | Διαδρομή |
| • destinationAerodrome | Αεροδρόμιο προορισμού |
| • depOrArr | Αναχωρηθέν ή αφικνούμενο αεροσκάφος |
| • active | Ενεργή πτήση |

Δ'4.12 Πίνακας AIP

Ο τελευταίος πίνακας της βάσης δεδομένων είναι ο πίνακας AIP. Ο πίνακας αυτό περιέχει πληροφορίες που αφορούν ένα αεροδρόμιο όπως το ύψος (elevation) στο οποίο βρίσκεται, τον προσανατολισμό του διαδρόμου προσγείωσης, το πλάτος του τροχόδρομου, την κατηγορία πυρόσβεσης, την διεύθυνση του στο Αεροναυτικό Δίκτυο Σταθερών Τηλεπικοινωνιών A.F.T.N, το είδος των καυσίμων που διαθέτει ως διευκολύνσεις, την διεύθυνση, τα τηλέφωνα, το fax και το e-mail των διοικητικών του υπηρεσιών. Οι πληροφορίες αυτές είναι δημοσιευμένες στο εγχειρίδιο Aeronautical Information Publication (AIP) το οποίο εκδίδεται από την Αρχή Πολιτικής Αεροπορίας της χώρας. Για λόγους συντομίας έχουν καταγραφεί οι πληροφορίες που αφορούν τα 41 αεροδρόμια μόνο της Ελλάδας από το AIP Greece.

Ο πίνακας AIP έχει τις παρακάτω ιδιότητες:

- | | |
|----------------------|--|
| • name | Όνομα του αεροδρομίου κατά ICAO |
| • title | Όνομα και τίτλος του αεροδρομίου |
| • elevation | Υψόμετρο |
| • address | Διεύθυνση |
| • telephone | Τηλέφωνο |
| • fax | FAX |
| • aftn | Διεύθυνση στο AFTN |
| • website | Ιστοσελίδα του αεροδρομίου |
| • email | Διεύθυνση ηλεκτρονικού ταχυδρομείου |
| • fuel | Είδος καυσίμων |
| • fuellingfacilities | Πληροφορίες για τις εταιρείες των καυσίμων |
| • firefighting | Κατηγορία πυρόσβεσης |
| • taxiwaywidth | Πλάτος τροχόδρομου |
| • runway | Διάδρομος προσγείωσης απογείωσης |

Δ΄.5 Διακομιστής Geoserver

Δ΄.5.1 Layers

Στο σύνολο ο Geoserver κατασκευάζει 12 layers τα οποία είναι:

- aerodromes_layer
- airways_layer
- waypoints_layer
- waypoints_vfr_layer
- nav aids_layer
- ctr_layer
- lgc_layer
- lgd_layer
- lgm_layer
- lgr_layer
- mtma_layer
- mfr_layer

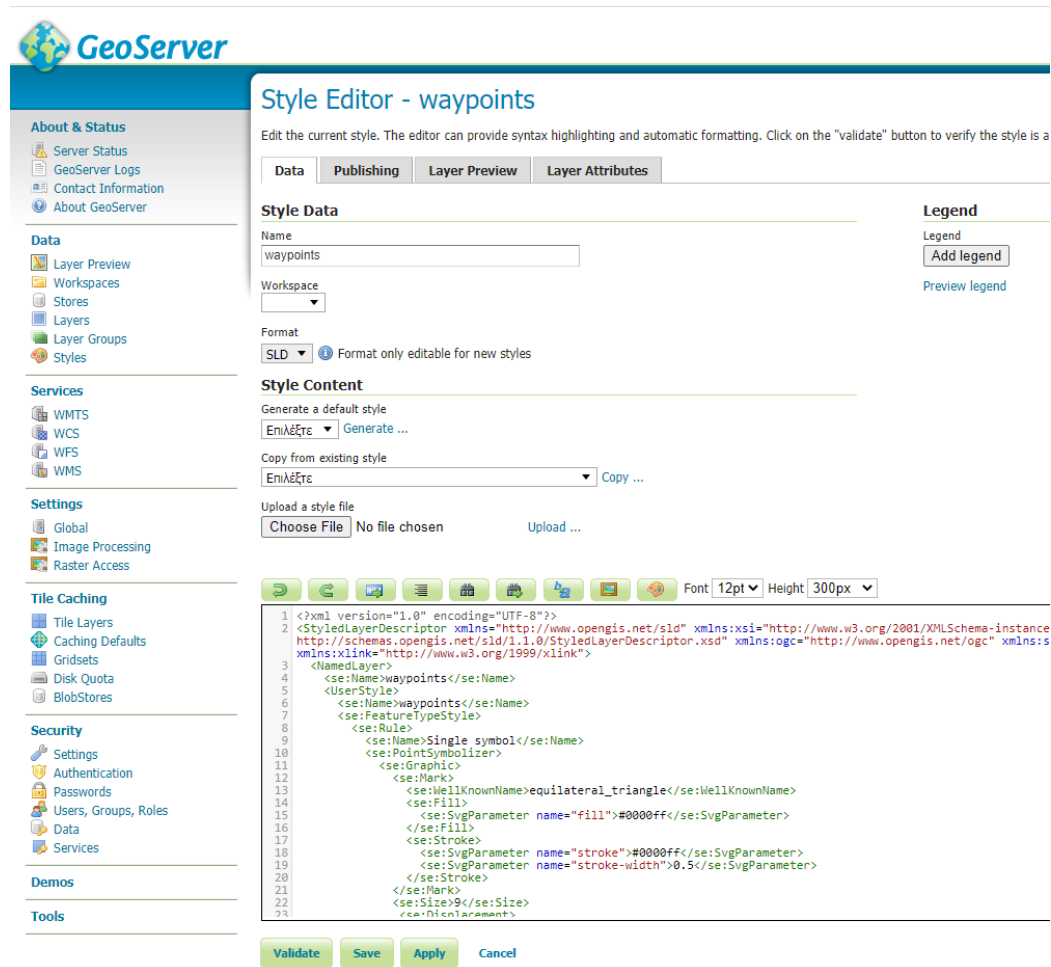
Τα παραπάνω layers έχουν τα ίδια ονόματα με τους πίνακες από τους οποίους τροφοδοτούνται με δεδομένα για λόγους ευκολίας.

Όπως είδαμε τα layers που δημιουργούνται από τον Geoserver έχουν διαφορετικά σχήματα. Π.χ οι αεροδιάδρομοι (airways_layer) είναι ευθείες ενώ οι τερματικές περιοχές αεροδρομίου είναι πολύγωνα. Το είδος του σχήματος καθορίζεται από την επέκταση postGIS κατά την αποθήκευση των δεδομένων. Με αυτό τον τρόπο ο Geoserver δημιουργεί το αντίστοιχο σχήμα. Μια εγγραφή ενός πίνακα περιλαμβάνει τις συντεταγμένες (coordinates) καθώς και το label το οποίο περιγράφει το όνομα του στοιχείου. Επίσης υπάρχει και το πεδίο fir το οποίο δείχνει σε ποια χώρα ανήκει το συγκεκριμένο στοιχείο (αεροδιάδρομος, waypoint κ.λπ)

Δ΄.5.2 Παραμετροποίηση S.L.D

Όπως αναφέραμε πιο πριν, ο Geoserver μας δίνει την δυνατότητα παραμετροποίησης του style προσφέροντάς μας αρκετές επιλογές ανάλογα με τις ανάγκες μας. Πριν ακολουθήσουμε την διαδικασία δημιουργίας ενός layer, την οποία περιγράψαμε με συντομία στην προηγούμενη παράγραφο, έχουμε ήδη δημιουργήσει το απαραίτητο style.

Στην επόμενη εικόνα βλέπουμε το εργαλείο Style Editor με το οποίο μπορούμε να παραμετροποιήσουμε ένα style.



Εικόνα Δ'4: Παραμετροποίηση ενός style με το Style Editor

Για το waypoints_layer δημιουργήσαμε ένα style το οποίο απεικονίζει ένα ισόπλευρο τρίγωνο για κάθε σημείο, (status), επειδή αυτό είναι το συνηθισμένο σχήμα που χρησιμοποιείται στους θεματικούς αεροπορικούς χάρτες. Στην συνέχεια βλέπουμε τα περιεχόμενα του style με όνομα waypoints.

- 1 `<?xml version = "1.0" encoding = "UTF-8"?>`
- 2 `<StyledLayerDescriptor xmlns = http://www.opengis.net/sld`
- 3 `xmlns:xsi = http://www.w3.org/2001/XMLSchema-instance`
- 4 `xsi:schemaLocation = "http://www.opengis.net/sld`
- 5 `http://schemas.opengis.net/sld/1.1.0/StyledLayerDescriptor.xsd"`
- 6 `xmlns:ogc = "http://www.opengis.net/ogc" xmlns:se = http://www.opengis.net/se`
- 7 `version = "1.1.0" xmlns:xlink = "http://www.w3.org/1999/xlink">`
- 8 `<NamedLayer>`

```

9      <se:Name> waypoints </se:Name>
10     <UserStyle>
11       <se:Name> waypoints </se:Name>
12       <se:FeatureTypeStyle>
13         <se:Rule>
14           <se:Name> Single symbol </se:Name>
15           <se:PointSymbolizer>
16             <se:Graphic>
17               <se:Mark>
18                 <se:WellKnownName> equilateral_triangle </se:WellKnownName>
19                 <se:Fill>
20                   <se:SvgParameter name = "fill"> #0000ff </se:SvgParameter>
21                 </se:Fill>
22                 <se:Stroke>
23                   <se:SvgParameter name = "stroke"> #0000ff </se:SvgParameter>
24                   <se:SvgParameter name = "stroke-width"> 0.5 </se:SvgParameter>
25                 </se:Stroke>
26               </se:Mark>
27             <se:Size> 9 </se:Size>
28             <se:Displacement>
29               <se:DisplacementX> 0 </se:DisplacementX>
30               <se:DisplacementY> -4 </se:DisplacementY>
31             </se:Displacement>
32           </se:Graphic>
33         </se:PointSymbolizer>
34       </se:Rule>
35     <se:Rule>
36       <se:TextSymbolizer>
37         <se:Label>
38           <ogc:PropertyName> label </ogc:PropertyName>
39         </se:Label>
40         <se:Font>
41           <se:SvgParameter name = "font-family"> MS Shell Dlg 2 </se:SvgParameter>
42           <se:SvgParameter name="font-size"> 10 </se:SvgParameter>
43         </se:Font>
44         <se:LabelPlacement>
45           <se:PointPlacement>

```

```

46      <se:AnchorPoint>
47      <se:AnchorPointX> 0.5 </se:AnchorPointX>
48      <se:AnchorPointY> 1 </se:AnchorPointY>
49      </se:AnchorPoint>
50      <se:Displacement>
51      <se:DisplacementX> 0 </se:DisplacementX>
52      <se:DisplacementY> -7 </se:DisplacementY>
53      </se:Displacement>
54      </se:PointPlacement>
55      </se:LabelPlacement>
56      <se:Fill>
57      <se:SvgParameter name="fill">#0000ff</se:SvgParameter>
58      </se:Fill>
59      </se:TextSymbolizer>
60      </se:Rule>
61      </se:FeatureTypeStyle>
62      </UserStyle>
63      </NamedLayer>
64      </StyledLayerDescriptor>

```

Στη γραμμή 18 καθορίζουμε ότι θα σχηματίζεται ένα ισόπλευρο τρίγωνο, στη γραμμή 20 καθορίζουμε το χρώμα του γεμίσματος, στη γραμμή 23 το χρώμα της γραμμής του τριγώνου, στη γραμμή 24 το πλάτος της γραμμής του τριγώνου, στη γραμμή 29 την μετατόπιση του σχήματος κατά τον άξονα του X, στη γραμμή 30 την μετατόπιση του σχήματος κατά τον άξονα του Y, στη γραμμή 38 καθορίζουμε από ποιο πεδίο του πίνακα θα λαμβάνεται το όνομα του στοιχείου, στη γραμμή 41 καθορίζουμε την γραμματοσειρά που θα έχει το όνομα του στοιχείου, στη γραμμή 42 καθορίζουμε το μέγεθος της γραμματοσειράς που θα έχει το όνομα του στοιχείου, στις γραμμές 47,48,51,52 καθορίζουμε την μετατόπιση του σημείου.

Για το `airways_layer` δημιουργήσαμε ένα style το οποίο απεικονίζει μια ευθεία γραμμή για κάθε τμήμα αεροδιαδρόμου που ενώνει δύο waypoints.

```

1      <?xml version="1.0" encoding = "UTF-8"?>
2      <StyledLayerDescriptor xmlns = http://www.opengis.net/sld
3      xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
4      xsi:schemaLocation="http://www.opengis.net/sld

```

```

5      http://schemas.opengis.net/sld/1.1.0/StyledLayerDescriptor.xsd"
6      xmlns:ogc="http://www.opengis.net/ogc" xmlns:se=http://www.opengis.net/se
7      version="1.1.0" xmlns:xlink="http://www.w3.org/1999/xlin">
8      <NamedLayer>
9          <se:Name>airways</se:Name>
10         <UserStyle>
11             <se:Name>airways</se:Name>
12             <se:FeatureTypeStyle>
13                 <se:Rule>
14                     <se:Name>Single symbol</se:Name>
15                     <se:LineSymbolizer>
16                         <se:Stroke>
17                             <se:SvgParameter name="stroke">#0000ff</se:SvgParameter>
18                             <se:SvgParameter name="stroke-width">0.2</se:SvgParameter>
19                             <se:SvgParameter name="stroke-linejoin">bevel</se:SvgParameter>
20                             <se:SvgParameter name="stroke-linecap">square</se:SvgParameter>
21                         </se:Stroke>
22                     </se:LineSymbolizer>
23                 </se:Rule>
24                 <se:Rule>
25                     <se:TextSymbolizer>
26                         <se:Label>
27                             <ogc:PropertyName>label</ogc:PropertyName>
28                         </se:Label>
29                         <se:Font>
30                             <se:SvgParameter name="font-family">MS Shell Dlg 2</se:SvgParameter>
31                             <se:SvgParameter name="font-size">12</se:SvgParameter>
32                         </se:Font>
33                         <se:LabelPlacement>
34                             <se:LinePlacement>
35                                 <se:PerpendicularOffset>10</se:PerpendicularOffset>
36                                 <se:Repeat>true</se:Repeat>
37                                 <se:Gap>214</se:Gap>
38                                 <se:GeneralizeLine>true</se:GeneralizeLine>
39                             </se:LinePlacement>
40                         </se:LabelPlacement>
41                     <se:Fill>

```

```

42      <se:SvgParameter name="fill">#0000ff</se:SvgParameter>
43      </se:Fill>
44      <se:VendorOption name="repeat">214</se:VendorOption>
45      </se:TextSymbolizer>
46      </se:Rule>
47      </se:FeatureTypeStyle>
48      </UserStyle>
49      </NamedLayer>
50      </StyledLayerDescriptor>

```

Στη γραμμή 15 καθορίζουμε ότι θα απεικονίζεται ευθεία γραμμή για κάθε αεροδιάδρομο, στις γραμμές 17-18 καθορίζουμε το χρώμα και το πάχος της γραμμής αυτής, στη γραμμή 27 καθορίζουμε ότι το όνομα του αεροδιαδρόμου θα λαμβάνεται από την ιδιότητα label του πίνακα `airways_layer` της βάσης δεδομένων, στις γραμμές 30-31 καθορίζουμε το μέγεθος και τη γραμματοσειρά με την οποία θα απεικονίζεται το όνομα του αεροδιαδρόμου, στη γραμμή 42 το χρώμα που θα έχει ως γέμισμα το όνομα του αεροδιαδρόμου, στη γραμμή 35 καθορίζουμε ότι το όνομα θα απεικονίζεται παράλληλα ως προς τον αεροδιάδρομο, στην γραμμή 36 καθορίζουμε ότι το όνομα του αεροδιαδρόμου θα επαναλαμβάνεται ανά τακτικά διαστήματα και στη γραμμή 37 καθορίζουμε το πόσο συχνά θα γίνεται αυτή η επανάληψη.

Δ'.6 Διακομιστής Node.js

Δ'.6.1 Δημιουργία του Διακομιστή

Χρησιμοποιώντας την βιβλιοθήκη `express` δημιουργήσαμε τον διακομιστή `node.js` με τον οποίο θα διασυνδέσουμε την `front end` εφαρμογή με τη βάση δεδομένων μας.

Ο διακομιστής `node.js` ακούει για αιτήματα (`requests`) στο `port 3000`. Τα αιτήματα που στέλνει η `front end` εφαρμογή μας στον διακομιστή, επιλέξαμε να γίνονται πάντα με την μέθοδο `get`. Επίσης δημιουργήσαμε τα κατάλληλα `path routes` στα οποία θα στέλνουμε τα αντίστοιχα `requests` με τα οποία θα εργαστούμε για να υλοποιήσουμε διάφορες λειτουργίες τις οποίες θα αναφέρουμε με συντομία σε αυτό το κεφάλαιο και θα αναλύσουμε περισσότερο στη συνέχεια.

Τα `requests` συνήθως είναι κάποια μεμονωμένα `queries` όταν είναι λίγα σε αριθμό, κάποιες φορές όμως λόγω του ότι αναγκάζομαστε να χρησιμοποιήσουμε πολλά `queries` σε ένα `request`, επιλέξαμε να στείλουμε στον διακομιστή τα `queries` με την μορφή ενός `query array`.

Παρακάτω βλέπουμε το JSON response που λαμβάνει η εφαρμογή όταν οι συντεταγμένες που στέλνουμε ως queries δεν αντιστοιχούν σε καμία περιοχή πλησίον κάποιου αεροδρομίου. Η ιδιότητα searchTerm επιστρέφει το γεωγραφικό πλάτος που στείλαμε ως query.

```
{  "status" : "notfound",
  "searchTerm" : "40.20"
}
```

Δ'.6.3 Route Path /coordinates

Το δεύτερο route path είναι το '/coordinates'. Το χρησιμοποιούμε για να δούμε αν ένα όνομα ανήκει σε ένα από τα παρακάτω: αεροδρόμιο, vfr waypoint, waypoint, ραδιοβοήθημα και αν ναι, να βρούμε τις γεωγραφικές του συντεταγμένες. Ο διακομιστής λαμβάνει ένα query με τη μέθοδο get το οποίο είναι το όνομα. Έπειτα δημιουργεί το ακόλουθο ερώτημα το οποίο αποστέλλεται στη βάση δεδομένων postgresSQL.

```
'SELECT ST_AsGeoJSON (COORDINATES) FROM (SELECT * FROM
AERODROMES_LAYER UNION SELECT * FROM WAYPOINTS_VFR_LAYER UNION
SELECT * FROM WAYPOINTS_LAYER UNION SELECT * FROM NAVAIDS_LAYER) AS
A WHERE LABEL = $1 '
```

Με παρόμοιο τρόπο με το προηγούμενο route path, η απάντηση αποστέλλεται στο front end μέρος της εφαρμογής από τον διακομιστή node.js σε μορφή JSON.

Παρακάτω βλέπουμε το URL με το οποίο στέλνουμε ως query το όνομα του ραδιοβοηθήματος 'KRK' που βρίσκεται στη Λευκίμμη της Κέρκυρας.

```
http://localhost:3000/coordinates/?q=KRK
```

Στη συνέχεια βλέπουμε το JSON response που λαμβάνει η εφαρμογή. Η ιδιότητα coordinates περιέχει τις γεωγραφικές συντεταγμένες του ραδιοβοηθήματος 'KRK'.

```
{  "status" : "ok",
  "results": "{  "type" : "Point",
    "coordinates" : [20.0706,39.4431]
  }"
}
```

Παρακάτω βλέπουμε το JSON response που λαμβάνει η εφαρμογή αν στείλουμε ως query το όνομα 'TEST'.

```
{  "status" : "notfound",
  "searchTerm" : "TEST"
}
```

Δ΄.6.4 Route Path /info

Το τρίτο route path, '/info', χρησιμοποιείται για να ανακτήσουμε από τον πίνακα AIP της βάσης δεδομένων τις πληροφορίες που αφορούν ένα αεροδρόμιο. Ο διακομιστής λαμβάνει το query που αντιστοιχεί στο όνομα του αεροδρομίου το οποίο αναζητούμε. Δημιουργείται το παρακάτω ερώτημα, το οποίο αποστέλλεται στη βάση δεδομένων postgresSQL.

```
'SELECT * FROM AIP WHERE NAME = $1'
```

Η απάντηση αποστέλλεται στην εφαρμογή μας σε μορφή JSON.

Ακολούθως βλέπουμε το URL με το οποίο στέλνουμε ως query το όνομα του αεροδρομίου της Κέρκυρας (LGKR).

<http://localhost:3000/info/?q=LGKR>

Παρακάτω βλέπουμε το JSON response που λαμβάνει η εφαρμογή. Η ιδιότητα results περιέχει σε εμφωλευμένο JSON τις δημοσιευμένες πληροφορίες του αεροδρομίου της Κέρκυρας που βρίσκονται στο εγχειρίδιο A.I.P.

```
{  "status" : "ok",
  "results" : {    "name" : "LGKR",
                  "title" : "KERKIRA - IOANNIS KAPODISTRIAS",
                  "elevation" : 5.74,
                  "address" : "Germanikis Scholis 10 15123 Maroussi GREECE",
                  "telephone" : "+30 698 5053 858",
                  "fax" : "+30 26610 45829",
                  "aftn" : "LGKRYDYX",
                  "website" : "www.cfu-airport.gr",
                  "email" : "kakkar@hcaa.gr",
                  "fuel" : "PF 100LL by GISCO TF JET A1 by EKO",
```

```

        "fuellingfacilities" : "EKO, GISCO From 1st APR to 31 OCT
H24",
        "firefighting" : "CAT 7",
        "taxiwaywidth": 23,
        "runway": "16/34"
    }
}

```

Δ'.6.5 Route Path /aerodrome

Το τέταρτο route path, '/aerodrome', χρησιμοποιείται για να βρούμε τις συντεταγμένες ενός αεροδρομίου από τον πίνακα aerodromes της βάσης δεδομένων. Το ερώτημα που δημιουργείται προς τη βάση δεδομένων είναι το παρακάτω:

```

'SELECT ST_AsGeoJSON (COORDINATES) FROM
AERODROMES_LAYER WHERE LABEL = $1 '

```

Παρόμοια η απάντηση αποστέλλεται στην εφαρμογή μας σε μορφή JSON.

Παρακάτω βλέπουμε το URL με το οποίο στέλνουμε ως query το όνομα του αεροδρομίου της Κέρκυρας καθώς και το αντίστοιχο JSON response που λαμβάνει η εφαρμογή.

<http://localhost:3000/aerodrome/?q=LGKR>

```

{
  "status" : "ok",
  "results" : "{    "type":"Point",
                  "coordinates": [19.9122, 39.6019]
                }"
}

```

Δ'.6.6 Route Path /airway

Το πέμπτο route path, '/airway', χρησιμοποιείται για να βρούμε αν δύο waypoints ανήκουν σε έναν διάδρομο και αν ναι, ποιο είναι το όνομα αυτού του διαδρόμου. Το χρησιμοποιούμε για να βρούμε αν ένα τμήμα μιας διαδρομής που αποτελείται από αυτά τα σημεία ανήκει σε έναν αεροδιάδρομο, με σκοπό να εμφανίσουμε το όνομα του πάνω από το τμήμα αυτό στην προεπισκόπηση του σχεδίου πτήσης, όπως θα δούμε στο επόμενο κεφάλαιο. Ο διακομιστής

λαμβάνει δυο queries τα οποία αντιστοιχούν στο 1^ο waypoint και στο 2^ο waypoint. Έπειτα δημιουργείται προς τη βάση δεδομένων το ακόλουθο ερώτημα.

```
'SELECT A.NAME FROM AIRWAYS AS A, AIRWAYS AS B WHERE A.WAYPOINT = $1 AND
B.WAYPOINT = $2 AND A.NAME = B.NAME'
```

Η απάντηση αποστέλλεται στο front end μέρος της εφαρμογής σε μορφή JSON.

Παρακάτω βλέπουμε το URL με το οποίο στέλνουμε ως queries τα waypoints VELOP και SOKR καθώς και το αντίστοιχο JSON response. Στην ιδιότητα name υπάρχει η τιμή B1, η οποία αντιστοιχεί στον ομώνυμο αεροδιάδρομο ο οποίος ορίζεται από αυτά τα σημεία.

<http://localhost:3000/airway/?q1=VELOP&q2=SOKRI>

```
{  "status": "ok",
  "results": {    "name": "B1"
  }
}
```

Ακολούθως βλέπουμε το JSON response που λαμβάνει η εφαρμογή όταν τα σημεία που στέλνουμε ως queries, δεν ανήκουν ταυτόχρονα σε κάποιον αεροδιάδρομο.

```
{  "status" : "notfound"

}
```

Δ΄.6.7 Route Path /FPL

Το έκτο route path, '/FPL', χρησιμοποιείται για εισάγουμε στον πίνακα FPL της βάσης δεδομένων PostgreSQL μια εγγραφή που αντιστοιχεί σε ένα σχέδιο πτήσης το οποίο έχει υποβληθεί με την φόρμα FPL. Στην περίπτωση αυτή ο τρόπος που χειριζόμαστε την αποστολή των queries από την εφαρμογή προς το διακομιστή node.js είναι διαφορετικός. Επειδή οι τιμές που πρέπει να υποβάλουμε είναι πάρα πολλές, δηλαδή 40, επιλέξαμε να χρησιμοποιήσουμε έναν πίνακα (array). Το array αυτό λαμβάνεται από τον διακομιστή και στη συνέχεια γίνεται διαχωρισμός των τιμών με την συνάρτηση split με delimiter χαρακτήρα το ','. Δημιουργείται η εντολή εισαγωγής, η οποία αποστέλλεται στη βάση δεδομένων.

```
'INSERT INTO FPL (AIRCRAFTID, FLIGHTRULES, TYPEOFFLIGHT, NUMBER,
TYPEOFAIRCRAFT, WAKETURBULENCECAT, EQUIPMENT,
DEPARTUREAERODROME, TIME, CRUISINGSPEED, LEVEL, ROUTE,
DESTINATIONAERODROME, TOTALEET, ALTERNATEAERODROME,
```

```

SECONDALTERNATEAERODROME, OTHERINFORMATION, ENDURANCE,
PERSONSONBOARD, EMERGENCYRADIOUHF, EMERGENCYRADIOVHF,
EMERGENCYRADIOELBA, DINGHIES, POLAR, DESERT, MARITIME,
JUNGLE, JACKETS, LIGHT, FLOURES, UHF, VHF, DINGHIESSECOND,
NUMBEROFDINGHIES, CAPACITY, COVER, COLOR,
AIRCRAFTCOLORANDMARKINGS, REMARKS, PILOTINCOMMAND) VALUES
($1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11, $12, $13, $14, $15, $16,
$17, $18, $19, $20, $21,
$22, $23, $24, $25, $26, $27, $28, $29, $30, $31, $32, $33, $34, $35,
$36, $37, $38, $39, $40) '

```

Αν η εισαγωγή είναι επιτυχής, αποστέλλεται απάντηση 'ok' σε μορφή JSON στην εφαρμογή.

Παρακάτω βλέπουμε το JSON response που λαμβάνει η εφαρμογή.

```

{
  "status" : "ok",
  "results" : "ok",
}

```

Δ'.6.8 Route Path /Flights

Το έβδομο route path, '/Flights', χρησιμοποιείται για την εισαγωγή στον πίνακα flights της βάσης δεδομένων PostgreSQL, εγγραφής η οποία αντιστοιχεί στο σχέδιο πτήσης που έχει ενεργοποιηθεί από τον ελεγκτή. Παρόμοια με το προηγούμενο route path, χρησιμοποιούμε ένα array για να στείλουμε τις 13 τιμές στο διακομιστή node.js. Επίσης χρησιμοποιούμε την συνάρτηση split για να διαχωρίσουμε τις τιμές από το array. Δημιουργείται η ακόλουθη εντολή εισαγωγής, η οποία στη συνέχεια αποστέλλεται στη βάση δεδομένων.

```

'INSERT INTO FLIGHTS (AIRCRAFTID, FLIGHTRULES, TYPEOFFLIGHT,
TYPEOFAIRCRAFT, WAKETURBULENCECAT, DEPARTUREAERODROME, TIME,
CRUISINGSPEED, LEVEL, ROUTE, DESTINATIONAERODROME, DEPORARR,ACTIVE)
VALUES ($1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11, $12, $13) '

```

Αν η εισαγωγή είναι επιτυχής, αποστέλλεται απάντηση 'ok' σε μορφή JSON στην εφαρμογή.

Ακολούθως βλέπουμε το JSON response.

```

{
  "status" : "ok",
  "results" : "ok",
}

```

Δ΄.6.9 Route Path /getRoute

Το όγδοο route path, '/getRoute', χρησιμοποιείται για την ανάκτηση της διαδρομής (route) από τον πίνακα flights μιας ενεργής πτήσης. Η αποστολή των τεσσάρων τιμών από την εφαρμογή στον node.js διακομιστή γίνεται με την βοήθεια ενός array. Οι τιμές αυτές προσδιορίζουν την συγκεκριμένη πτήση (aircraftId, departureaerodrome, time, destinationaerodrome). Έπειτα γίνεται με παρόμοιο τρόπο όπως είδαμε και πριν, ο διαχωρισμός των τιμών και η αποστολή του παρακάτω ερωτήματος στη βάση δεδομένων.

```
'SELECT A.ROUTE FROM FLIGHTS AS A WHERE A.AIRCRAFTID = $1 AND  
A.DEPARTUREAERODROME = $2 AND A.TIME = $3 AND  
A.DESTINATIONAERODROME = $4'
```

Αν η αναζήτηση είναι επιτυχής, η απάντηση στέλνεται στην εφαρμογή, σε μορφή JSON.

Παρακάτω βλέπουμε το URL με το οποίο στέλνουμε ως queries τα στοιχεία της πτήσης SXBAT η οποία αναχωρεί στις 09:50 ώρα UTC από το αεροδρόμιο της Κέρκυρας (LGKR) με προορισμό το αεροδρόμιο της Αθήνας (LGAV).

<http://localhost:3000/getRoute/?q=SXBAT,LGKR,0950,LGAV>

Στη συνέχεια βλέπουμε το JSON response, το οποίο περιέχει μέσα στην ιδιότητα route τη διαδρομή της συγκεκριμένης πτήσης.

```
{  "status": "ok",  
  "results": {    "route": "KRK IGOU PARNA IXONI PIKAD"  
  }  
}
```

Δ΄.6.10 Route Path /Reset

Το ένατο route path, '/Reset', χρησιμοποιείται για να διαγράψουμε τα περιεχόμενα του πίνακα flights, αρχικοποιώντας τον, κατά την εκκίνηση της εφαρμογής. Δημιουργείται η ακόλουθη εντολή διαγραφής, η οποία αποστέλλεται στη βάση δεδομένων.

```
'DELETE FROM FLIGHTS'
```

Αν η διαγραφή είναι επιτυχής αποστέλλεται απάντηση 'ok' στην εφαρμογή σε μορφή JSON.

Παρακάτω βλέπουμε το JSON response.

```
{  "status" : "ok",
```

```

    "results" : "ok",
  }

```

Δ'.6.11 Route Path /EndOfFlight

Το δέκατο route path, '/EndOfFlight', χρησιμοποιείται για την διαγραφή μιας πτήσης από τον πίνακα flights της βάσης δεδομένων, όταν η πτήση αυτή έχει ολοκληρωθεί. Στην περίπτωση αυτή έχουμε 4 τιμές που αποστέλλονται στον διακομιστή node.js από την εφαρμογή οι οποίες προσδιορίζουν την συγκεκριμένη πτήση και είναι οι εξής: aircraftId, departureaerodrome, time, destinationaerodrome. Στην περίπτωση αυτή χρησιμοποιούμε πάλι ένα array για την αποστολή των τεσσάρων τιμών. Η εντολή διαγραφής η οποία αποστέλλεται στη συνέχεια στη βάση δεδομένων είναι η ακόλουθη.

```

'DELETE FROM FLIGHTS AS A WHERE A.AIRCRAFTID = $1 AND
A.DEPARTUREAERODROME = $2 AND A.TIME = $3 AND
A.DESTINATIONAERODROME = $4 '

```

Στην περίπτωση επιτυχημένης διαγραφής αποστέλλεται απάντηση 'ok' στην εφαρμογή σε μορφή JSON.

Ακολούθως βλέπουμε το JSON response που λαμβάνει η εφαρμογή.

```

{
  "status" : "ok",
  "results" : "ok",
}

```

Δ'.6.12 Route Path /DeleteGPXFile

Το ενδέκατο route path, '/DeleteGPXFile', χρησιμοποιείται για την διαγραφή του αρχείου gpx που υπάρχει στον κατάλογο ./data/GPXFiles/ με την βοήθεια της βιβλιοθήκης fs που αντιστοιχεί σε μια πτήση που έχει ολοκληρωθεί. Το όνομα του αρχείου λαμβάνεται από τον διακομιστή και έπειτα γίνεται η διαγραφή του. Δεν αποστέλλεται απάντηση πίσω στην εφαρμογή.

Δ'.6.13 Route Path /GetMyFlightPlansContent

Το δωδέκατο route path, '/GetMyFlightPlansContent', χρησιμοποιείται για να ανακτηθούν τα περιεχόμενα του καταλόγου /MyFlightPlans/. Στον κατάλογο αυτό, περιέχονται όλα τα σχέδια πτήσης που έχει αποθηκεύσει ο ελεγκτής. Τα περιεχόμενα του καταλόγου εμφανίζονται στις

φόρμες 'Load FPL' και 'Save FPL'. Η ανάγνωση του καταλόγου γίνεται με την χρήση της βιβλιοθήκης fs. Ο διακομιστής node.js δεν λαμβάνει κάποιο query από την εφαρμογή καθώς δεν είναι απαραίτητο. Τα αποτελέσματα της ανάγνωσης του καταλόγου /MyFlightPlans/ αποστέλλονται στην εφαρμογή με τη μορφή JSON.

Παρακάτω βλέπουμε το URL με το οποίο η εφαρμογή λαμβάνει το αντίστοιχο JSON response.

<http://localhost:3000/GetMyFlightPlansContent/>

Ακολούθως βλέπουμε και το JSON response, το οποίο περιέχει στην ιδιότητα results τα περιεχόμενα του καταλόγου /MyFlightPlans.

```
{  "status" : "ok",
  "results": ["IMEJO.txt", "SXAJT.txt", "OAL054.txt", "SEH080.txt"]
}
```

Δ΄.6.14 Route Path /ReadFPL

Το δέκατο τρίτο route path είναι το '/ReadFPL'. Το χρησιμοποιούμε για να διαβάσουμε τα περιεχόμενα ενός αρχείου που βρίσκεται στο φάκελο /MyFlightPlans/. Η ανάγνωση του αρχείου γίνεται και εδώ με τη βοήθεια της βιβλιοθήκης fs. Το όνομα του αρχείου που πρόκειται να διαβαστεί λαμβάνεται ως query από τον διακομιστή node.js. Μετά την επιτυχή ανάγνωση, τα αποτελέσματα στέλνονται στο front end μέρος της εφαρμογής με τη μορφή JSON και χρησιμοποιούνται στη φόρμα 'Load FPL', όπως θα δούμε παρακάτω.

Παρακάτω βλέπουμε το URL με το οποίο η εφαρμογή στέλνει ως query το αρχείο SXBAT.txt στο διακομιστή node.js.

<http://localhost:3000/ReadFPL/?q=SXBAT.txt>

Στη συνέχεια βλέπουμε το JSON response που λαμβάνει η εφαρμογή και το οποίο επιστρέφει μέσα στην ιδιότητα results τα περιεχόμενα του αρχείου SXBAT.txt.

```
{  "status": "ok",
  "results": "(FPL-SXBAT-VG-01C172/L-SY/C-LGKR0950-N0120VFR KRK IGOU
  PARNA IXONI PIKAD-LGAV -DOF/210702)"
}
```

Δ'.6.15 Route Path /WriteFPL

Το δέκατο τέταρτο route path, '/WriteFPL', χρησιμοποιείται για την εγγραφή ενός αρχείου στον κατάλογο '/MyFlightPlans/'. Το αρχείο αυτό αντιστοιχεί σε ένα σχέδιο πτήσης που έχει σώσει ο ελεγκτής με την φόρμα 'Save FPL' προκειμένου να το επεξεργαστεί αργότερα ή να το έχει σαν template. Πολλές πτήσεις επαναλαμβάνονται τακτικά, όποτε είναι συνηθισμένη πρακτική κάποια σχέδια να σώζονται σαν templates ώστε να αποφεύγεται καθυστέρηση στην ηλεκτρολόγησή τους στο μέλλον. Η εγγραφή του σχεδίου πτήσης στο κατάλογο '/MyFlightPlans/' γίνεται με τη χρήση της βιβλιοθήκης fs. Λαμβάνονται δύο queries από τον διακομιστή node.js, τα οποία αντιστοιχούν στο όνομα του αρχείου και στα περιεχόμενά του. Μετά την επιτυχή εγγραφή του αρχείου αποστέλλεται απάντηση 'ok' στην εφαρμογή με την μορφή JSON.

Παρακάτω βλέπουμε το URL με το οποίο στέλνονται στο διακομιστή node.js ως queries το όνομα του αρχείου SXAJT.txt και τα περιεχόμενα τα οποία θα έχει μετά την εγγραφή του.

[http://localhost:3000/WriteFPL/?q1=SXAJT.txt&q2=\(FPL-SXAJT-VG-01C172/L-S/C-LGKR1200-N0100VFR KRK YNN-LGIO LGPZ-DOF/210702\)](http://localhost:3000/WriteFPL/?q1=SXAJT.txt&q2=(FPL-SXAJT-VG-01C172/L-S/C-LGKR1200-N0100VFR KRK YNN-LGIO LGPZ-DOF/210702))

Ακολούθως βλέπουμε το JSON response που λαμβάνει η εφαρμογή από το διακομιστή node.js.

```
{
  "status" : "ok",
  "data" : "ok",
}
```

Δ'.6.16 Route Path /WriteGPXFile

Το δέκατο πέμπτο route path είναι το '/WriteGPXFile'. Το χρησιμοποιούμε για την εγγραφή ενός αρχείου gpx στον κατάλογο ./data/GPXFiles/. Το αρχείο αυτό αντιστοιχεί σε μια πτήση η οποία μόλις ενεργοποιήθηκε. Η μέθοδος είναι παρόμοια με το προηγούμενο route path. Η εγγραφή γίνεται με τη βοήθεια της βιβλιοθήκης fs και λαμβάνονται δύο queries, το όνομα του αρχείου και τα περιεχόμενά του. Παρόμοια, μετά την επιτυχή εγγραφή του αρχείου αποστέλλεται απάντηση 'ok' στην εφαρμογή με την μορφή JSON.

Παρακάτω βλέπουμε το URL με το οποίο στέλνονται στο διακομιστή node.js ως queries το όνομα του αρχείου SXBAT-LGKR0950-LGAV.gpx και τα περιεχόμενα τα οποία θα έχει μετά την εγγραφή του.

[http://localhost:3000/WriteGPXFile/?q1=SXAJT.txt&q2=<?xml version="1.0" encoding="UTF-8"?><gpx xmlns=http://www.topografix.com/GPX/1/1](http://localhost:3000/WriteGPXFile/?q1=SXAJT.txt&q2=<?xml version=)

```

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" creator="WebGIS"
version="1.1" xsi:schemaLocation="http://www.topografix.com/GPX/1/1
http://www.topografix.com/GPX/1/1/gpx.xsd"><rte><name>SXBAT-LGKR0950-
LGAV</name><rtept lat="39.6019" lon="19.9122"><name>LGKR</name></rtept>
<rtept lat="39.4431" lon="20.0706"><name>KRK</name></rtept>
<rtept lat="39.5083" lon="20.2711"><name>IGOU</name></rtept>
<rtept lat="39.2622" lon="20.4303"><name>PARNA</name></rtept>
<rtept lat="38.315" lon="22.2322"><name>IXONI</name></rtept>
<rtept lat="38.0614" lon="22.6978"><name>PIKAD</name></rtept>
<rtept lat="37.9367" lon="23.9444"><name>LGAV</name></rtept></rte></gpx>

```

Ακολουθώς βλέπουμε το αντίστοιχο JSON response.

```

{
  "status" : "ok",
  "data" : "ok",
}

```

Δ΄.6.17 Route Path /metar

Το δέκατο έκτο route path, '/metar', χρησιμεύει για την ανάκτηση ενός μετεωρολογικού δελτίου METAR από την ιστοσελίδα <https://www.aviationweather.gov/> με τη χρήση της βιβλιοθήκης @airport-discovery/notams. Λαμβάνεται ένα query από τον διακομιστή node.js το οποίο αντιστοιχεί στο όνομα ενός αεροδρομίου, κατά ICAO. Στη συνέχεια μετά την επιτυχή λήψη του METAR, αυτό αποστέλλεται στο front end μέρος της εφαρμογής, σε μορφή JSON για να εμφανιστεί στην αντίστοιχη φόρμα.

Ακολουθώς βλέπουμε το URL με το οποίο στέλνεται ως query το όνομα του αεροδρομίου της Κέρκυρας.

<http://localhost:3000/metar/?q=LGKR>

Παρακάτω βλέπουμε το JSON response που λαμβάνει η εφαρμογή από το διακομιστή node.js και το οποίο περιέχει το METAR του αεροδρομίου της Κέρκυρας. Παρατηρούμε ότι ολόκληρο το METAR βρίσκεται στην ιδιότητα rawText.

```

{
  "status" : "ok",
  "results": {
    "rawText": "LGKR 151750Z 25006KT CAVOK 25/16
Q1016 NOSIG",
    "latitude": "39.62",
    "longitude": "19.92",

```

```

    "temperatureC": "25.0",
    "dewPointC": "16.0",
    "wind": { "directionDegrees": "250",
    "speedKt": "6"
    },
    "visibilityStatuteMi": "6.21",
    "altimInHg": "30.0",
    "qualityControlFlags": { "no_signal": "TRUE"
    },
    "skyCondition": { "skyCover": "CAVOK"
    },
    "flightCategory": "VFR",
    "metarType": "METAR",
    "elevationM": "2.0"
  }
}

```

Δ'.6.18 Route Path /notam

Το δέκατο έβδομο route path, είναι το '/notam'. Το χρησιμοποιούμε για την ανάκτηση των αεροναυτικών αγγελιών NOTAMs ενός αεροδρομίου. Η λειτουργία είναι παρόμοια με το προηγούμενο route path. Η ανάκτηση των NOTAMs γίνεται από την ιστοσελίδα notams.aim.faa.gov, με τη χρήση της βιβλιοθήκης @airport-discovery/notams. Λαμβάνεται από τον διακομιστή node.js ένα query το οποίο αντιστοιχεί στο όνομα ενός αεροδρομίου κατά ICAO. Μετά την επιτυχή λήψη των NOTAMs αποστέλλονται στην εφαρμογή σε μορφή JSON προκειμένου να εμφανιστούν στην αντίστοιχη φόρμα.

Πιο κάτω βλέπουμε το URL με το οποίο στέλνεται ως query το όνομα του αεροδρομίου της Κέρκυρας.

<http://localhost:3000/notam/?q=LGKR>.

Παρακάτω βλέπουμε το JSON response που λαμβάνει η εφαρμογή από το διακομιστή node.js και το οποίο περιέχει ένα NOTAM του αεροδρομίου της Κέρκυρας. Στην πραγματικότητα συνήθως τα NOTAMs είναι περισσότερα από ένα και περιέχονται σε σειρά ως JSON objects στην ιδιότητα notamList.

```

{ "status": "ok",

```

```

"results": {
  "notamList": [{
    "facilityDesignator" : "LGKR",
    "notamNumber" : "A1486/21",
    "featureName" : "International",
    "issueDate" : "05/21/2021 0707",
    "startDate" : "05/21/2021 0707",
    "endDate" : "09/30/2021 2359",
    "source" : "USNS",
    "sourceType" : "I",
    "icaoMessage" : "A1486/21 NOTAMR A1473/21\n
                    Q)                                LGGG/QFAAP/IV/NBO/A
/000/999/3936N01954E005\n
A) LGKR B) 2105210707 C) 2109302359 \nE) FOR IFR/VFR GENERAL AND
BUSINESS FLIGHTS PRIOR PERMISSION\nREQUIRED (PPR).PLEASE REFER
TO:\n1.AIP LGKR AD 2\n
2.WWW.FRAPORT-GREECE.COM/ENG/OUR-EXPERTISE-AND-SERVICES/AVIATION/PPR-
\nPROCEDURE-AND-GUIDELINES.",
    "traditionalMessage" : "",
    "plainLanguageMessage" : "",
    "traditionalMessageFrom4thWord" : "FOR IFR/VFR
GENERAL AND BUSINESS FLIGHTS PRIOR PERMISSION\nREQUIRED (PPR).PLEASE
REFER TO:\n1.AIP LGKR AD 2\n2.WWW.FRAPORT-GREECE.COM/ENG/OUR-EXPERTISE-
AND-SERVICES/AVIATION/PPR-\n PROCEDURE-AND-GUIDELINES...",
    "icaoId" : "LGKR",
    "accountId" : "2198",
    "airportName" : "IOANNIS KAPODISTRIAS INTL",
    "procedure" : false,
    "userId" : 0,
    "transactionID" : 60586390,
    "cancelledOrExpired": false,
    "digitalTppLink": false,
    "status": "Active",
    "contractionsExpandedForPlainLanguage": false,
    "keyword": "INTERNATIONAL",
    "snowtam": false,
    "geometry": "POINT(5541619.64164954
                    5994482.9399985)",
    "digitallyTransformed": false,
    "messageDisplayed": "concise",
    "hasHistory": false,
    "moreThan300Chars": false,

```

```

    "showingFullText": false,
    "locID": 352353,
    "defaultIcao": false,
    "crossoverTransactionID": 0,
    "crossoverAccountID": "",
    "mapPointer": "POINT(19.91222222 39.60194444)",
    "requestID": 0
  }],

  "startRecordCount": 1,
  "endRecordCount": 5,
  "totalNotamCount": 5,
  "filteredResultCount": 0,
  "criteriaCaption": "      Location      search      on
location(s)              LGKR",
  "searchDateTime": "2021-09-15 18:37:33",
  "linkedLocationCaption": "",
  "error": "",
  "countsByType": [{ "name": "CONSTRUCTION",
    "value": 0
  },
  {
    "name" : "RNAV",
    "value": 0
  },
  {
    "name" : "GPS",
    "value" : 0
  },
  {
    "name" : "Ending this month",
    "value": 1
  },
  {
    "name" : "LTA",
    "value": 0
  },
  {
    "name" : "TFR",
    "value": 0
  },
  {
    "name" : "CARF",
    "value" : 0
  }
],

```

```
"requestID": 0}}}
```

Δ'.6.19 Route Path /STAR

Το δέκατο όγδοο route path, '/STAR', χρησιμεύει για την μετατροπή μιας εικόνας που αντιστοιχεί στη διαδικασία άφιξης (ST.AR) ενός αεροδρομίου σε base64 string. Στη συνέχεια το string αυτό αποστέλλεται στην εφαρμογή, όπου γίνεται η αντίστροφη μετατροπή του σε εικόνα η οποία εμφανίζεται στην αντίστοιχη φόρμα. Οι εικόνες ST.AR είναι οργανωμένες σε καταλόγους με το όνομα του αεροδρομίου κατά ICAO. Για παράδειγμα για το αεροδρόμιο της Κέρκυρας, ο κατάλογος webGIS/webserver/src/images/LGKR περιέχει τους καταλόγους STAR και SID στους οποίους βρίσκονται οι αντίστοιχες εικόνες που αφορούν το συγκεκριμένο αεροδρόμιο της Κέρκυρας. Μέσα στους καταλόγους αυτούς οι εικόνες είναι οργανωμένες ανά αύξοντα αριθμό. Με τον ίδιο τρόπο είναι οργανωμένες και οι εικόνες για τα υπόλοιπα αεροδρόμια. Με τον τρόπο αυτό, ο διακομιστής node.js λαμβάνει δύο queries τα οποία αντιστοιχούν στο όνομα του αεροδρομίου κατά ICAO και στο όνομα της εικόνας. Έτσι, σχηματίζεται path της εικόνας π.χ. '/wamp64/www/WebGIS/webserver/src/images/'+query1+'/STAR/'+query2+'.png' και το οποίο αντιστοιχεί στην συγκεκριμένη εικόνα που θέλουμε να ανακτήσουμε. Στη συνέχεια με την βοήθεια της βιβλιοθήκης imageToBase64 μετατρέπεται η εικόνα σε base64 string και αποστέλλεται στην εφαρμογή με τη μορφή JSON.

Πιο κάτω βλέπουμε το URL με το οποίο στέλνονται ως queries το όνομα του αεροδρομίου της Κέρκυρας και ο αριθμός 1 ο οποίος αντιστοιχεί στην πρώτη εικόνα STAR.

<http://localhost:3000/STAR/?q1=LGKR&q2=1>

Παρακάτω βλέπουμε το JSON response το οποίο λαμβάνεται από την εφαρμογή και περιέχει μέσα στην ιδιότητα results το base64 string της αντίστοιχης STAR εικόνας 1 του αεροδρομίου της Κέρκυρας. Για λόγους συντομίας το string αυτό έχει περικοπεί στο παράδειγμα.

```
{
  "status": "ok",
  "results": "iVBORw0KGgoAAAANSUgAACbAAAAbaCAYAAAAeRcPdAAAAACXBIWX
MAABcSAAAXEgFnn9JSAACAAElEQVR4nOydB7wVxfm//4kl9hpLYq+x994VC1ixix17F0Wwa2
wICCKISBNUFBEEG6BYsIDYC2LvRo09idEYEzX7v8/Ae35z5szu2XPvwr2X+30+n1Hu7p49u3
u2zCwz7/v//t//+3/JDjvskFx66aUqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKi
oqKioqKjO9bL/99gnu2v/jP3/...AAAAElFTkSuQmCC"
}
```

Δ'.6.20 Route Path /SID

Το δέκατο ένατο route path, '/SID', χρησιμεύει παρόμοια με το προηγούμενο, για την μετατροπή μιας εικόνας που αντιστοιχεί στη διαδικασία αναχώρησης (S.I.D) ενός αεροδρομίου σε base64 string. ο διακομιστής node.js λαμβάνει δύο queries τα οποία αντιστοιχούν στο όνομα του αεροδρομίου κατά ICAO και στο όνομα της εικόνας. Σχηματίζει το path της εικόνας με την βοήθεια των δύο queries και με τη χρήση της βιβλιοθήκης imageToBase64 imageToBase64 μετατρέπεται η εικόνα σε base64 string. Στη συνέχεια αποστέλλεται στο front end με τη μορφή JSON προκειμένου να εμφανιστεί στην αντίστοιχη φόρμα.

Αντίστοιχα με το προηγούμενο route ακολούθως βλέπουμε το URL με το οποίο στέλνονται ως queries το όνομα του αεροδρομίου της Κέρκυρας και ο αριθμός 1 ο οποίος αντιστοιχεί στην πρώτη εικόνα SID.

<http://localhost:3000/SID/?q1=LGKR&q2=1>

Παρακάτω βλέπουμε το αντίστοιχο JSON response που λαμβάνει η εφαρμογή.

```
{  "status": "ok",
  "results": "iVBORw0KGgoAAAANSUgAACbAAAAbaCAYAAAAeRcPdAAAAACXBIWXMAABcSAAAXEgFnn9JSAACAAE1EQVR4nOydBbgV1dQAF++1uxC7C7u7FQOVsLARDGwJCUFQQJQuCWnp7u4UARULBRMT43q9Xq9d69/vOnzj2uusmb334WzgwPc+z6ecPbNnz6xZU2u9863/+7//+z9zwQUXmCeeeEJDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ...QggghhBwAAAAASUVO RK5CYII="
}
```

Δ'.6.21 Route Path /GetSIDSTARContent

Το τελευταίο route path, '/GetSIDSTARContent', χρησιμοποιείται για να βρούμε το πλήθος των εικόνων που έχουν οι κατάλογοι STAR και SID. Αυτό γίνεται για να γνωρίζει η εφαρμογή μας το εύρος που θα ψάξει σε κάθε κατάλογο, για τις εικόνες STAR και SID, αυξάνοντας τον αριθμό της εικόνας από το 1 μέχρι το μέγιστο αριθμό που αντιπροσωπεύει το πλήθος τους. Ο διακομιστής node.js λαμβάνει ένα query το οποίο αντιστοιχεί στο όνομα του αεροδρομίου κατά ICAO και στη συνέχεια με τη χρήση της βιβλιοθήκης fs διαβάσει με τη σειρά τα περιεχόμενα των καταλόγων STAR και SID. Έπειτα αποστέλλει στην εφαρμογή το μήκος (length) των περιεχομένων, το οποίο αντιπροσωπεύει το πλήθος των εικόνων, με τη μορφή JSON ενσωματώνοντας τα πεδία sid και

star με τα αντίστοιχα πλήθη. Πιο κάτω βλέπουμε το URL με το οποίο στέλνεται ως query το όνομα του αεροδρομίου της Κέρκυρας.

<http://localhost:3000/GetSIDSTARContent/?q=LGKR>

Παρακάτω βλέπουμε το JSON response, το οποίο περιέχει τον αριθμό των SID και STAR εικόνων που έχει το αεροδρόμιο της Κέρκυρας.

```
{  "status": "ok",
  "sid": 7,
  "star": 6
}
```

Δ΄.7 Front End

Το front end επίπεδο της εφαρμογής μας περιλαμβάνει το αρχείο index.html, τα αρχεία main.js, menu.js, layers.js, controls.js, cursors.js, FPL.js, FDP.js, fly.js, log.js, splashes.js, timeUTC.js, το αρχείο style.css καθώς και τα απαραίτητα αρχεία της βιβλιοθήκης openlayers και ol-ext.

Το αρχείο index.html φορτώνει την σελίδα HTML στον φυλλομετρητή, όπου εμφανίζεται η αρχική σελίδα της εφαρμογής. Στο αρχείο αυτό, φορτώνονται στο head μέρος οι απαραίτητες βιβλιοθήκες openlayers, ol-ext, bootstrap, jquery, τα απαραίτητα fonts fontmaki2, fontawesomedef και τα αντίστοιχα style αρχεία css. Στο body μέρος φορτώνονται το bootstrap nav bar που θα χρησιμοποιήσουμε ως μενού της εφαρμογής με τις ανάλογες επιλογές, τα div της εφαρμογής, μαζί με τα απαραίτητα στοιχεία και forms που περικλείουν καθώς και τα αρχεία javascript που περιλαμβάνει η εφαρμογή.

Δ΄.7.1 Navigation Bar

Στο πάνω μέρος της οθόνης υπάρχει μπάρα περιήγησης (navigation bar) η οποία περιλαμβάνει τις επιλογές Home, AFTN, Map, AFIS, Options, Help.

Η πρώτη επιλογή είναι η επιλογή Home. Κάνοντας κλικ, φορτώνεται η αρχική σελίδα της εφαρμογής.

Η δεύτερη επιλογή είναι η επιλογή AFTN, η οποία εμφανίζει την οθόνη του τερματικού A.F.T.N. Αποτελεί μια προσπάθεια διασύνδεσης της εφαρμογής μας με το δίκτυο AFTN. Εδώ ο ελεγκτής

μπορεί να δει τα εξερχόμενα και εισερχόμενα σήματα από και προς το AFTN, κάποια από τα οποία παράγονται αυτόματα από την εφαρμογή.

Η τρίτη επιλογή, είναι η επιλογή Map, η οποία εμφανίζει έναν θεματικό αεροπορικό χάρτη με λειτουργίες και δυνατότητες που θα αναλύσουμε στη συνέχεια.

Η τέταρτη επιλογή είναι η επιλογή AFIS. Κάνοντας κλικ επάνω της, εμφανίζεται ένα drop down menu το περιλαμβάνει μια λίστα από μη ελεγχόμενα αεροδρόμια της Ελλάδας, στα οποία παρέχεται η υπηρεσία Πληροφοριών Πτήσεων - Αεροδρομίου Aerodrome Flight Information Service (AFIS). Η λίστα αυτή περιλαμβάνει τα αεροδρόμια Κυθήρων (LGKC) Αλέξανδρος Ωνάσης, Πάρου (LGPA), Νάξου (LGNX), Κάσου (LGKS), Μήλου (LGML), Κοζάνης (LGKZ), Καστοριάς (LGKA) και Ικαρίας (LGIK). Επιλέγοντας ο ελεγκτής κάποιο αεροδρόμιο από τα συγκεκριμένα, ο χάρτης μεταφέρει το κεντρικό του σημείο, στις συντεταγμένες του αεροδρομίου αυτού. Με αυτό τον τρόπο ο χρήστης δηλώνει στην εφαρμογή ποιο θα είναι το αεροδρόμιο εργασίας του, στο οποίο θα παρέχει υπηρεσία. Κατά την εκκίνηση, η εφαρμογή τοποθετεί το κέντρο του χάρτη στις συντεταγμένες του αεροδρομίου της Κέρκυρας (LGKR).

Η Πέμπτη επιλογή είναι η επιλογή Options. Με την επιλογή αυτή ο ελεγκτής μπορεί να διαλέξει αν το εικονίδιο του αεροσκάφους θα εμφανίζεται στον χάρτη ως μια πράσινη κουκίδα ή ως εικονίδιο αεροσκάφους.

Η έκτη επιλογή είναι η επιλογή Help. Με την επιλογή αυτή ο ελεγκτής μπορεί να δει στην οθόνη μια σύντομη επεξήγηση για το τι κάνει η εφαρμογή.

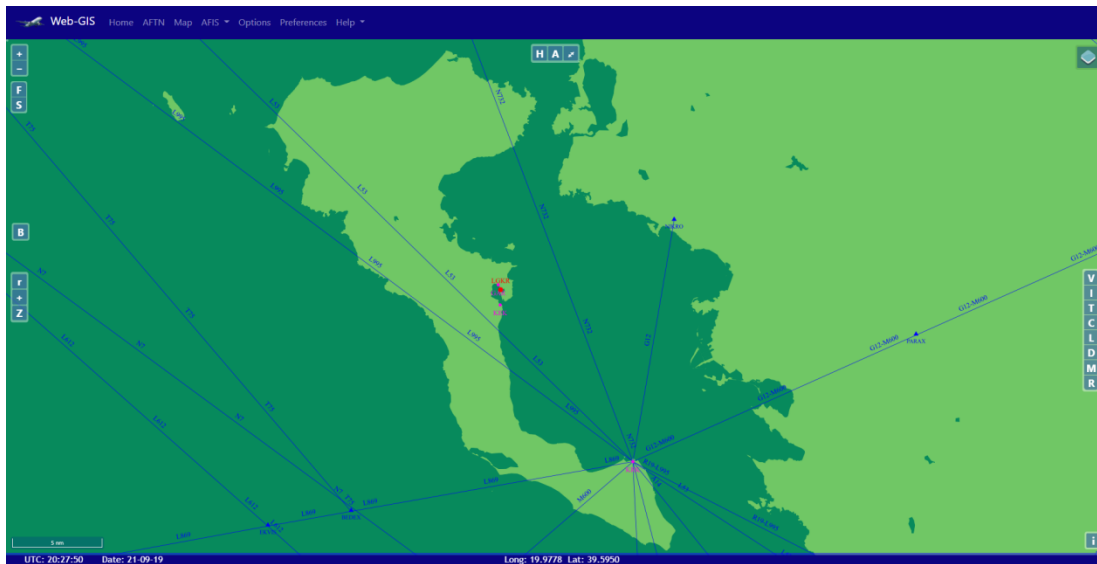
Δ'.7.2 Βασικός Χάρτης

Όλες οι εφαρμογές GIS, χρησιμοποιούν έναν χάρτη πάνω στον οποίο οπτικοποιούν τα δεδομένα τους και απεικονίζουν επιπλέον πληροφορίες. Ο χάρτης αυτός ονομάζεται βασικός χάρτης (base map) ή βασικό επίπεδο (base layer). Όλα τα άλλα επίπεδα (layers) που δημιουργούνται από μια GIS εφαρμογή φορτώνονται πάνω στον βασικό χάρτη. Ο βασικός χάρτης ανάλογα με τις προτιμήσεις του χρήστη μπορεί να είναι δορυφορικός (satellite), πολιτικός, γεωφυσικός κ.λπ. Επίσης ο βασικός χάρτης μπορεί να είναι είτε raster (ψηφιδωτός) ή vector (διανυσματικός).

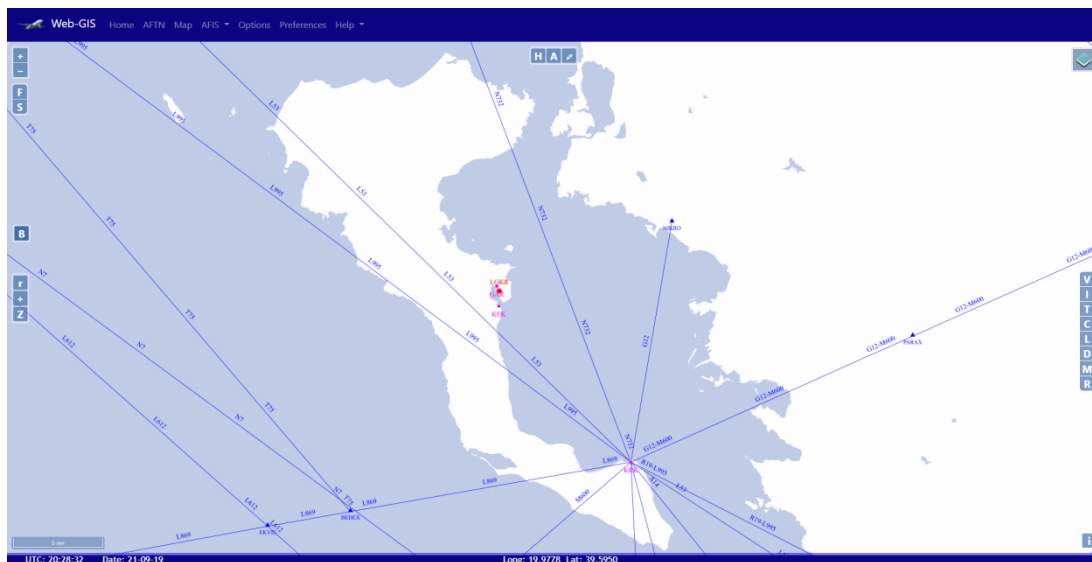
Η βιβλιοθήκη Openlayers δεν προσφέρει την δυνατότητα χρήσης ενός διανυσματικού χάρτη απ' ευθείας. Αυτό όμως μπορεί να επιτευχθεί με την χρήση της βιβλιοθήκης Mapbox GL-JS.[46]

Στην εφαρμογή μας χρησιμοποιούμε για βασικό χάρτη δυο διανυσματικούς χάρτες τους οποίους εισάγουμε στην βιβλιοθήκη Openlayers με τη βοήθεια της Mapbox GL-JS. Ο χάρτης που φορτώνεται με την εκκίνηση της εφαρμογής έχει πράσινη απόχρωση για τα ύδατα και πράσινο

ανοιχτό χρώμα για την ξηρά. Επιλέχθηκαν τα παραπάνω χρώματα έτσι ώστε ο χάρτης να είναι οικίος και να ομοιάζει με τον χάρτη που χρησιμοποιούν οι ελεγκτές εναέριας κυκλοφορίας στα υπάρχοντα συστήματα radar των αεροδρομίων. Ο δεύτερος βασικός χάρτης έχει γαλάζια ανοιχτή απόχρωση για τα ύδατα και άσπρο χρώμα για την ξηρά. Τα χρώματα αυτά επιλέχθηκαν έτσι ώστε ο χάρτης αυτός να ομοιάζει με τους θεματικούς αεροπορικούς χάρτες. Η αλλαγή μεταξύ των δύο βασικών χαρτών γίνεται με την χρήση του κουμπιού 'B' που υπάρχει στα αριστερά της οθόνης όπως θα δούμε και στη συνέχεια.



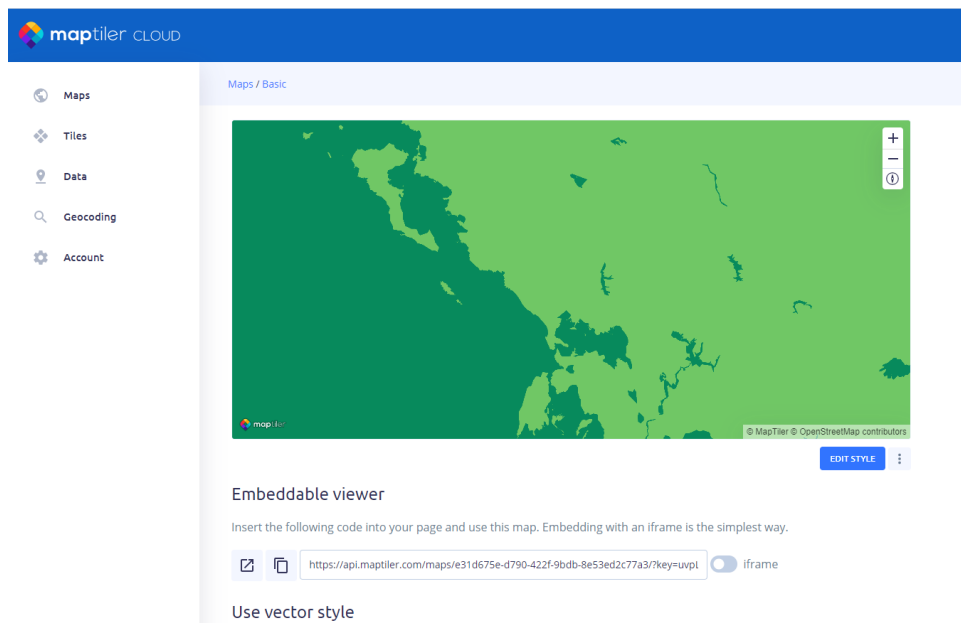
Εικόνα Δ΄5: Πρώτος βασικός χάρτης



Εικόνα Δ΄6: Δεύτερος βασικός χάρτης

Οι δύο χάρτες δημιουργήθηκαν στην ιστοσελίδα <https://www.maptiler.com/>. Η εταιρεία ανάπτυξης λογισμικού maptiler είναι ένας παγκόσμιος πάροχος υπηρεσίας χαρτών.[47] Αναπτύσσει το ανοιχτού κώδικα project OpenMapTiles, το οποίο στοχεύει στην δημιουργία παγκόσμιων χαρτών από ανοικτά δεδομένα. Αποτελείται από ένα σετ από εργαλεία, επιτρέποντας

σε όλους την δημιουργία προσωπικών διανυσματικών χαρτών από τα δεδομένα OpenStreetMap. Είναι διαθέσιμο σαν ελεύθερο λογισμικό, αλλά επίσης μπορεί κάποιος να το χρησιμοποιήσει στην ιστοσελίδα <https://www.maptiler.com/cloud/> εκεί όπου δημιουργήσαμε και τους χάρτες της εφαρμογής μας. [48] Κάνοντας login με το email μας μας δίνεται η δυνατότητα να χρησιμοποιήσουμε έτοιμους χάρτες τους οποίους μπορούμε να επεξεργαστούμε και να αφαιρέσουμε στοιχεία που δεν χρειαζόμαστε, όπως δρόμους, ονομασίες χωρών κ.λπ. Ο πρώτος βασικός χάρτης που δημιουργήσαμε, φαίνεται στην παρακάτω εικόνα.



Εικόνα Δ'7: Δημιουργία του πρώτου χάρτη στο maptiler

Το επόμενο βήμα αφού δημιουργήσουμε τον χάρτη, είναι να τον δημοσιεύσουμε. Στην συνέχεια με το key και το link που μας δίνει η maptiler, μπορούμε να τον χρησιμοποιήσουμε στην εφαρμογή μας καθώς αυτό θα αποτελεί το source των δεδομένων του χάρτη.

Για να δημιουργήσουμε τον βασικό χάρτη με την βιβλιοθήκη Openlayers, απαιτείται να δημιουργήσουμε ένα αντικείμενο ol.map (ol είναι η βασική βιβλιοθήκη της Openlayers), το οποίο έχει δύο ιδιότητες, την target και την view. Η ιδιότητα target προσδιορίζει το div στην HTML σελίδα μας, που θα φέρει τον χάρτη μας και η ιδιότητα view προσδιορίζει ποιες θα είναι οι συντεταγμένες του κέντρου του χάρτη μας κατά την εμφάνισή του. Επίσης στην ιδιότητα view καθορίζουμε και το επίπεδο zoom που θα έχει ο βασικός χάρτης κατά την πρώτη φόρτωσή του. [49]

Τα παραπάνω καθορίζονται στις γραμμές 5-17 στο αρχείο layers.js.

Στην συνέχεια χρησιμοποιούμε την συνάρτηση olms(map, styleJson) η οποία ανήκει στη βιβλιοθήκη mapbox GL-JS για να εισάγουμε τον διανυσματικό μας χάρτη στην βιβλιοθήκη

Openlayers. Το όρισμα `map` αντιστοιχεί στο αντικείμενο `map` της προηγούμενης παραγράφου ενώ το όρισμα `styleJson` αντιστοιχεί στο `link` με το `source` του χάρτη.

Μπορούμε να αλλάξουμε τον βασικό χάρτη, με την ίδια συνάρτηση, αλλάζοντας την τιμή του δεύτερου ορίσματος και κατά συνέπεια το `source` των δεδομένων του βασικού χάρτη με άλλον.

Δ΄.7.3 Layers

Κάθε άλλο επίπεδο `layer` που θα δημιουργήσουμε και θα εισάγουμε στην εφαρμογή μας, θα προστεθεί επάνω στο βασικό μας χάρτη. Στην συνέχεια δημιουργούμε τα επίπεδα που θα εμφανίζει η εφαρμογή. Τα επίπεδα αυτά τροφοδοτούνται από τον Geoserver με το πρωτόκολλο WMS και δημιουργούνται με την βιβλιοθήκη Openlayers.

Η διαδικασία που ακολουθούμε για την δημιουργία ενός επιπέδου που θα τροφοδοτείται με δεδομένα από τον Geoserver είναι η παρακάτω. Δημιουργείται ένα αντικείμενο `ol.layer.Image` στο οποίο καθορίζουμε το `source` από το οποίο θα τροφοδοτείται. [50]

Το `source` είναι κι αυτό ένα αντικείμενο, `source.ImageWMS`, με ιδιότητες που καθορίζουν:

το URL από το οποίο θα λαμβάνονται τα δεδομένα του `layer` (`'http://localhost:8080/geoserver/airGIS/wms'`), το όνομα του `layer` που στέλνει ο Geoserver (`{'LAYERS':'airGIS:aerodromes_layer'}`) και το είδος του διακομιστή από τον οποίο λαμβάνεται το `layer` (`'geoserver'`). Μόλις δημιουργηθεί το `layer` προστίθεται στον βασικό χάρτη με τη μέθοδο `addLayer` και όρισμα το αντικείμενο `layer` που δημιουργήσαμε. Δημιουργούνται με την σειρά τα παρακάτω επίπεδα: `aerodromesLayer`, `airwaysLayer`, `navaidsLayer`, `waypointsLayer`, `waypointsVFR`Layer, `VFR`Layer, `MTMA`Layer, `CTRL`Layer, `LGCL`Layer, `LGDL`Layer, `LGML`Layer, `LGR`Layer, `aerodromeCircleLayer`. Η δημιουργία των παραπάνω `layers` υπάρχει στο αρχείο `layers.js` στις γραμμές 30-226. [51]

Όλα τα επίπεδα που δημιουργήσαμε, εμφανίζονται πάνω στον βασικό χάρτη, όπως μια στοίβα. Ένα επίπεδο που βρίσκεται πιο κοντά στην κορυφή της στοίβας επικαλύπτει ένα επίπεδο που βρίσκεται στη βάση της στοίβας. Η θέση του επιπέδου στην στοίβα αυτή καθορίζεται από την τιμή της ιδιότητας `ZIndex` που έχει το κάθε αντικείμενο `layer`. Όσο μικρότερη είναι η τιμή που έχει η ιδιότητα `ZIndex` τόσο πιο κοντά στον βασικό χάρτη θα βρίσκεται το `layer`. Η αλλαγή του `ZIndex` γίνεται είτε κατά την στιγμή που δημιουργούμε το `layer`, είτε γίνεται με χρήση της μεθόδου `setZIndex`. Επίσης κάνοντας χρήση κι άλλων μεθόδων της βιβλιοθήκης Openlayers μπορούμε να αλλάξουμε και άλλες ιδιότητες ενός `layer`, όπως για παράδειγμα, να είναι ορατό ή μη ορατό πάνω στο βασικό χάρτη ή να είναι ημιδιαφανές ή όχι. Αυτά μπορούν να γίνουν με τη χρήση των μεθόδων `setVisible`

και `setOpacity`, δίνοντας αντίστοιχα ως ορίσματα για το `setVisible` μια boolean τιμή `true` ή `false` και για το `setOpacity` έναν αριθμό από το 0 έως το 1.

Όπως έχουμε αναφέρει σε προηγούμενο κεφάλαιο, το styling που έχει το κάθε layer καθορίζεται από τον Geoserver, μέσω του Style Layer Descriptor και δεν μπορεί να αλλάξει δυναμικά από την εφαρμογή μας. Η εφαρμογή μας μπορεί να επέμβει πάνω στα layers που σερβίρονται από τον Geoserver μόνο ως προς τη διαφάνεια και την ορατότητά τους.

Η διαφάνεια και η ορατότητα του κάθε layer, μπορεί να αλλάξει χειροκίνητα δηλαδή με τρόπο μη προγραμματιστικό, μέσα από την εφαρμογή μας, με τη χρήση του κουμπιού Layer Switcher που βρίσκεται πάνω δεξιά. Μόλις ο χρήστης κάνει κλικ επάνω στο κουμπί αυτό, αναπτύσσεται μια λίστα με όλα τα επίπεδα που υπάρχουν πάνω στον βασικό χάρτη. Στην συνέχεια επιλέγοντας όποιο layer θέλει μπορεί να αλλάξει την διαφάνειά και την ορατότητα του συγκεκριμένου layer. Το κουμπί αυτό ενσωματώνεται στην εφαρμογή μας μέσω της επέκτασης `ol-ext` στην οποία ανήκει. Η ενσωμάτωσή του καθορίζεται στις γραμμές 301-306 του αρχείου `controls.js`. Προγραμματιστικά στην εφαρμογή μας επεμβαίνουμε στο layer `airwaysLayer` και δίνουμε την τιμή 0.4 στο `opacity`, προκειμένου οι αεροδιάδρομοι να είναι πιο αχνοί και να μην κουράζουν και μπερδεύουν τον χρήστη.

Δ'.7.4 Κουμπιά

Στην συνέχεια αφού δημιουργήσαμε όλα τα layers και τα ενσωματώσαμε στον βασικό χάρτη `base map`, προσθέσαμε τα απαραίτητα κουμπιά `buttons` με τα οποία μπορεί ο ελεγκτής να αλληλεπιδράσει με την εφαρμογή.

Κάθε κουμπί είναι ένα αντικείμενο `ol.control.Button` το οποίο δημιουργείται με τη βιβλιοθήκη `ol-ext` και προστίθεται στον βασικό χάρτη με τη μέθοδο `addControl`. [52]

Ένα κουμπί μπορεί να είναι είτε αυτόνομο, όποτε προστίθεται στον βασικό χάρτη κατ' ευθείαν με την μέθοδο `addControl`, είτε να ανήκει σε μια ομάδα κουμπιών. Στην περίπτωση αυτή δημιουργείται ένα αντικείμενο `ol.control.Bar`, το οποίο είναι μια μπάρα κουμπιών την οποία προσθέτουμε στο βασικό χάρτη πάλι με τη μέθοδο `addControl`. Στην συνέχεια με την ίδια μέθοδο, προσθέτουμε κουμπιά όχι απ' ευθείας στον βασικό χάρτη αλλά στην μπάρα κουμπιών. Μπορούμε να καθορίσουμε τη θέση που θα έχει η μπάρα κουμπιών στην οθόνη, χρησιμοποιώντας την μέθοδο `setPosition` και δίνοντας ως όρισμα ένα string από τα επόμενα: `'top'`, `'left'`, `'bottom'`, `'right'` ή και συνδυασμό τους π.χ `'top-right'`[53]

Κάθε κουμπί έχει δύο καταστάσεις οι οποίες είναι ενεργό (active) και ανενεργό (inactive). Όταν πατήσουμε ένα κουμπί, αυτό αλλάζει χρώμα και γίνεται πράσινο που αντιστοιχεί στην κατάσταση ενεργό. Στην περίπτωση αυτή εμφανίζεται το αντίστοιχο layer που αντιπροσωπεύει το κουμπί.

Κατά τη δημιουργία του αντικειμένου `ol.control.Bar` , με τις παρακάτω ιδιότητες δηλώνουμε:

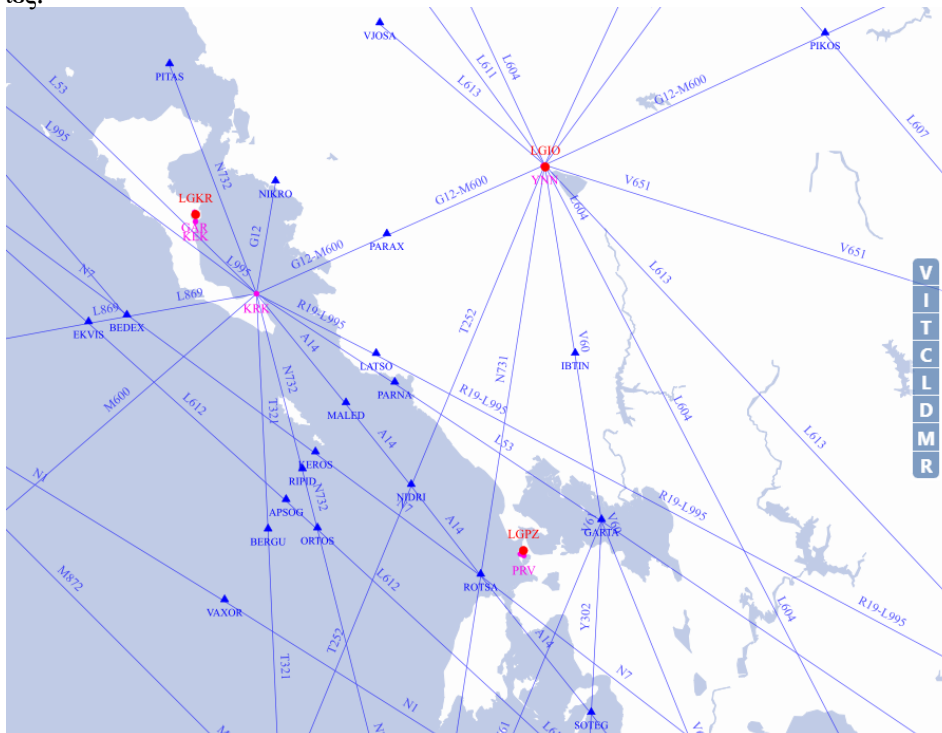
- `html`: τον χαρακτήρα που θα εμφανίζεται μέσα στο κουμπί
- `active`: αρχική κατάσταση, ενεργή ή μη ενεργή του κουμπιού
- `handleclick`: την συνάρτηση callback που θα εκτελεστεί με το click event.

Όταν ένα κουμπί πατηθεί τότε καλείται και η αντίστοιχη συνάρτηση callback. Κάνοντας χρήση αυτής της συνάρτησης μπορούμε να εμφανίσουμε ή να αποκρύψουμε το αντίστοιχο layer ή να εμφανίσουμε ή να αποκρύψουμε μια ομάδα layers μαζί.

Τα κουμπιά που υπάρχουν πάνω στο βασικό χάρτη της εφαρμογής μπορούμε να τα χωρίσουμε σε τέσσερις κατηγορίες.

Δ'.7.4.1 Κουμπιά V, I, T, C, L, D, M, R

Στην πρώτη κατηγορία ανήκουν τα κουμπιά V, I, T, C, L, D, M, R τα οποία βρίσκονται ομαδοποιημένα στην δεξιά πλευρά της οθόνης. Τα κουμπιά αυτά επιτελούν τις παρακάτω λειτουργίες:



Εικόνα Δ'8: Κουμπιά V, I, T, C, L, D, M, R

Το κουμπί V εμφανίζει τα layers VFR και waypoints VFR και αποκρύπτει τα layers airways και waypoints. Χρησιμοποιεί δηλαδή στο να δει ο ελεγκτής τον χάρτη VFR.

Το κουμπί I εμφανίζει τα layers airways και waypoints και αποκρύπτει τα layers VFR και waypoints VFR.

Το κουμπί T εμφανίζει το layer MTMA (στρατιωτικές τερματικές περιοχές MTMA).

Το κουμπί C εμφανίζει το layer CTR (Controlled Traffic Region).

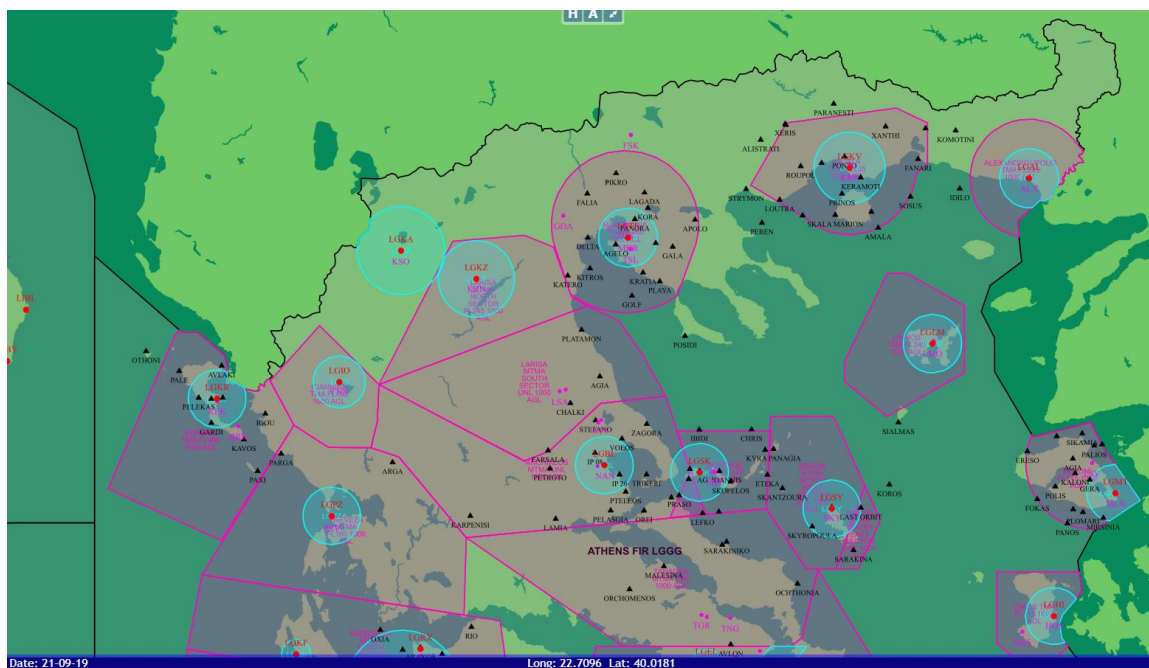
Το κουμπί L εμφανίζει το layer LGC (Ελεγχόμενες Περιοχές).

Το κουμπί D εμφανίζει το layer LGD (Επικίνδυνες Περιοχές).

Το κουμπί M εμφανίζει το layer LGM (Στρατιωτικές Περιοχές).

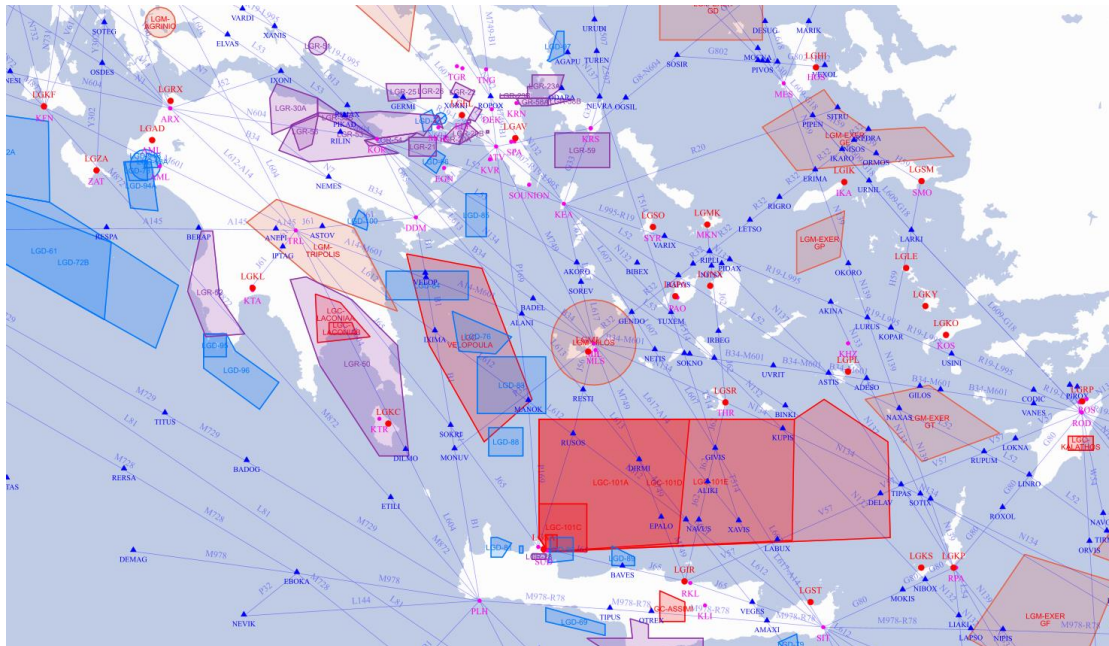
Το κουμπί R εμφανίζει το layer LGR (Απαγορευμένες Περιοχές).

Στην επόμενη εικόνα βλέπουμε τα layers MTMA, CTR, VFR.



Εικόνα Δ'9: MTMA, CTR, VFR Layers

Στην επόμενη εικόνα βλέπουμε τα layers LGC, LGD, LGM και LGR.



Εικόνα Δ΄10: LGC, LGD, LGM, LGR Layers

Δ΄.7.4.2 Κουμπιά F, S

Στη δεύτερη κατηγορία κουμπιών ανήκουν τα κουμπιά F και S τα οποία βρίσκονται ομαδοποιημένα στην πάνω αριστερά της οθόνης. Τα κουμπιά αυτής της κατηγορίας επιτελούν τις παρακάτω λειτουργίες:

Το κουμπί F εμφανίζει την φόρμα FPL με την οποία ο ελεγκτής μπορεί να δημιουργήσει ένα καινούργιο σχέδιο πτήσης και να το σώσει ή να φορτώσει ένα ήδη αποθηκευμένο σχέδιο πτήσης. Επίσης με τη φόρμα αυτή ο ελεγκτής μπορεί υποβάλλει ένα σχέδιο πτήσης, το οποίο αποστέλλεται στη βάση δεδομένων και στη συνέχεια περιμένει να ενεργοποιηθεί από την φόρμα Electronic Strip Bay. Θα αναλύσουμε παρακάτω περισσότερο την συγκεκριμένη λειτουργία.

Στην επόμενη εικόνα βλέπουμε την φόρμα FPL.

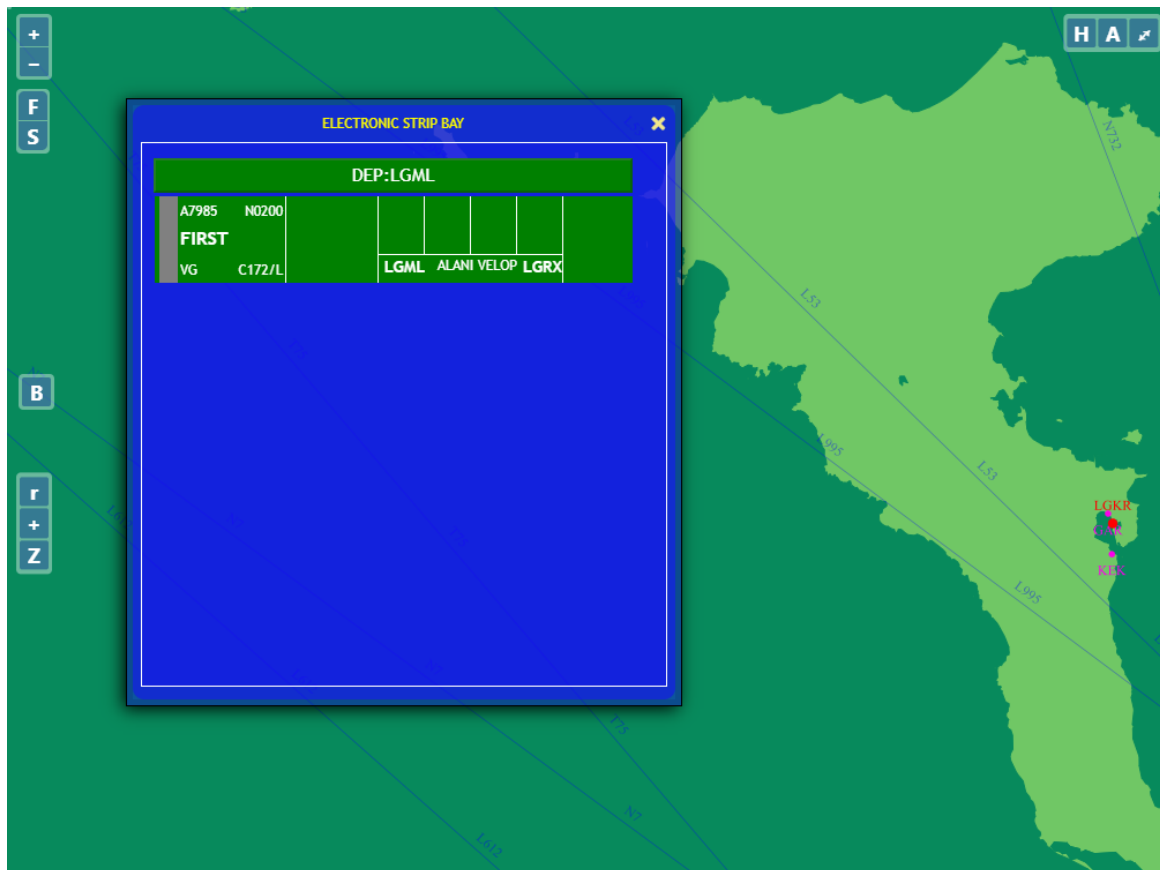


Εικόνα Δ'11: Φόρμα FPL

Με το κουμπί S μπορούμε να εμφανίσουμε την φόρμα Electronic Strip Bay. Η φόρμα αυτή περιέχει όλα τα σχέδια πτήσης που έχουν υποβληθεί και αναμένουν να απογειωθούν. Θα αναλύσουμε στη συνέχεια περισσότερο και αυτή τη λειτουργία.

Στην επόμενη εικόνα βλέπουμε την φόρμα Electronic Strip Bay

.



Εικόνα Δ΄12: Φόρμα Electronic Strip Bay

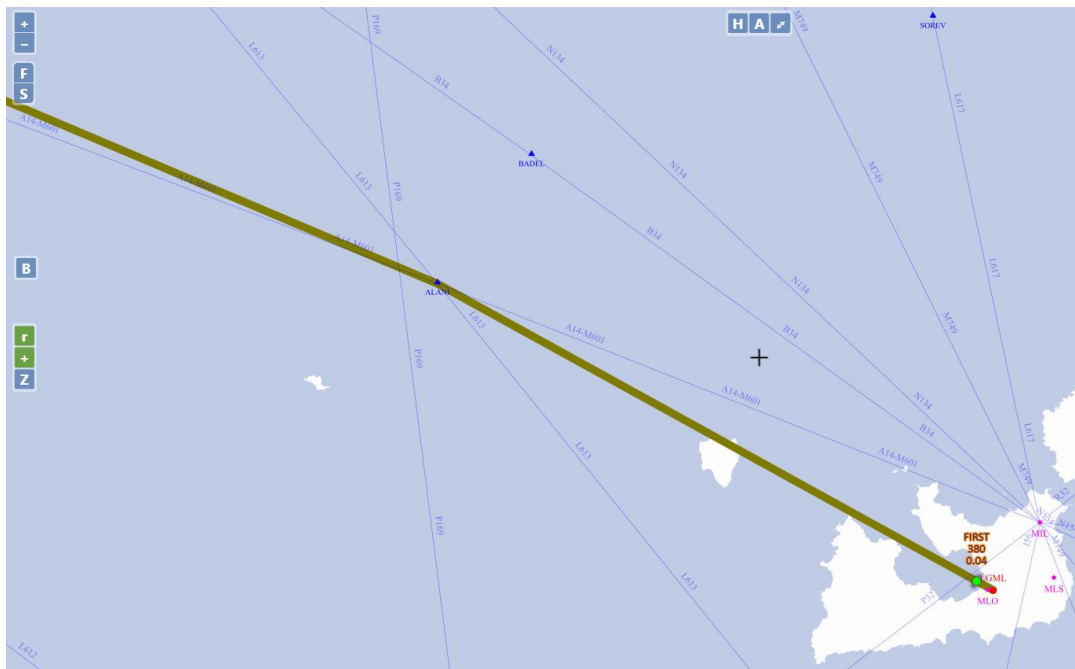
Και στις δυο περιπτώσεις, οι φόρμες FPL και Electronic Strip Bay εμφανίζονται με τη χρήση του JQuery, αλλάζοντας με τη μέθοδο css την ιδιότητα display σε block όπως μπορούμε να δούμε στις γραμμές 327-328 και 343-344 του αρχείου controls.js.

Δ΄.7.4.3 Κουμπιά r, +, Z, B

Στην τρίτη κατηγορία κουμπιών ανήκουν τα κουμπιά r, +, Z, B τα οποία βρίσκονται στην κάτω αριστερά πλευρά της οθόνης. Τα κουμπιά της τρίτης κατηγορίας επιτελούν τις παρακάτω λειτουργίες:

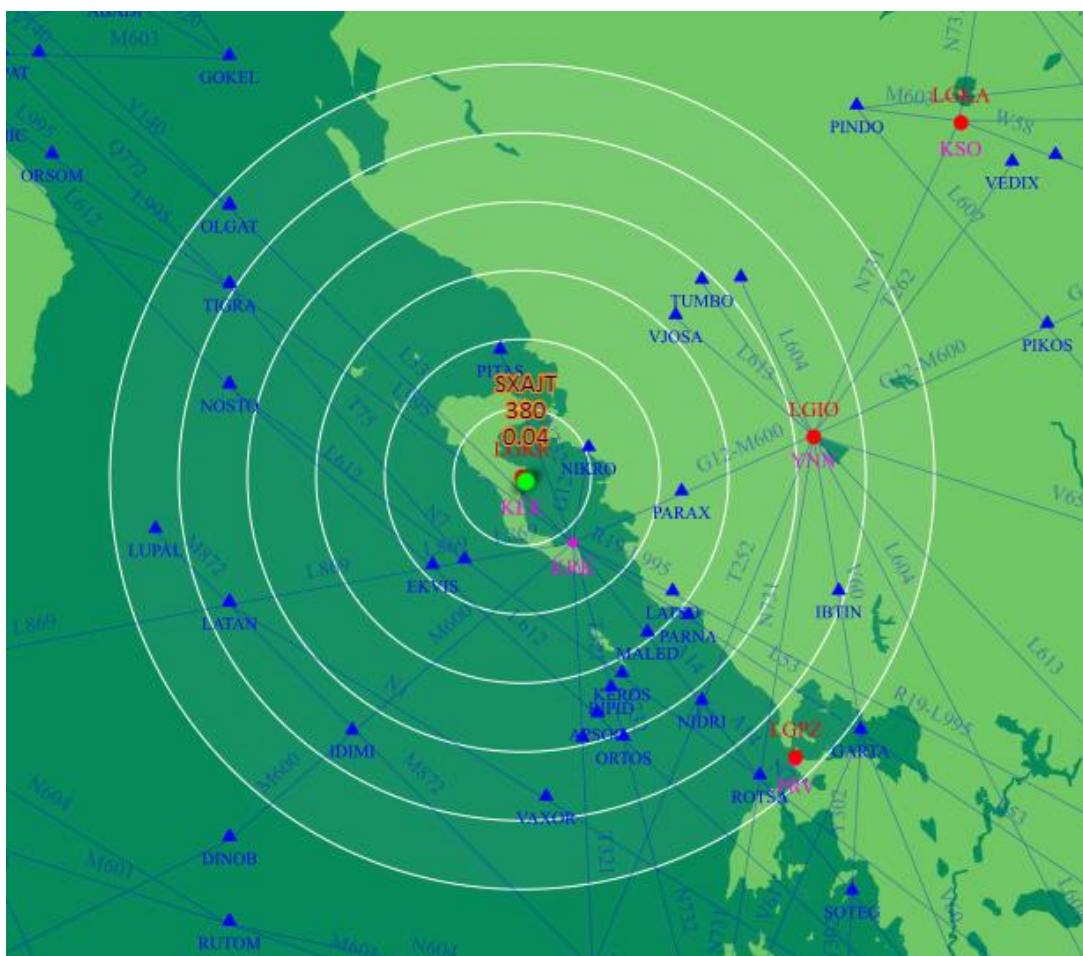
Με το κουμπί + ο ελεγκτής εναέριας κυκλοφορίας έχει την δυνατότητα να εμφανίσει έναν σταυρό ο οποίος προσδιορίζει το κέντρο του χάρτη. Επίσης στο κάτω μέρος της οθόνης απεικονίζονται οι συντεταγμένες που έχει ο σταυρός.

Στην επόμενη εικόνα βλέπουμε την λειτουργία από τα κουμπιά r και +.



Εικόνα Δ'13: Κουμπιά r και +

Στην επόμενη εικόνα βλέπουμε την λειτουργία του κουμπιού Z.



Εικόνα Δ'14: Κουμπί Z

Επειδή η εμφάνιση του σταυρού δεν γίνεται με κάποιο layer ή φόρμα, στην οποία ανήκει ο σταυρός αλλά πρόκειται για αντικείμενο ol.control, χρησιμοποιούμε έναν άλλο τρόπο για να τον εμφανίσουμε ή να τον αποκρύψουμε. Όλοι οι μηχανισμοί που προστίθενται στον βασικό χάρτη σχηματίζουν ένα array στο αντικείμενο controls. Έτσι χρησιμοποιούμε τις μεθόδους getControls και getArray για να βρούμε τον συγκεκριμένο μηχανισμό, ο οποίος είναι 27^{ος} και έπειτα να εφαρμόσουμε πάνω του την μέθοδο setVisible. Έτσι η εντολή αλλαγής της ορατότητας του συγκεκριμένου μηχανισμού θα είναι η εξής:

```
map.getControls().getArray()[26].setVisible(όρισμα)
```

όπου όρισμα είναι μια boolean τιμή true ή false.

Στην περίπτωση που μελλοντικά αφαιρέσουμε ή προσθέσουμε ένα νέο κουμπί εξυπακούεται ότι θα πρέπει ο αριθμός 27 να αλλάξει κατάλληλα.

Με το κουμπί r ο ελεγκτής έχει την δυνατότητα να εμφανίσει τις διαδρομές (route) των ενεργών πτήσεων, δηλαδή των πτήσεων που βρίσκονται στον αέρα. Όταν ένα σχέδιο πτήσης ενεργοποιηθεί, δηλαδή το αεροσκάφος ξεκινά να πετά, τότε δημιουργείται και το αντικείμενο layer που έχει ως ιδιότητα title, την τιμή 'Aircraft Route' και απεικονίζει το ίχνος όλης της διαδρομής του αεροσκάφους. Σε default κατάσταση το layer αυτό είναι μη ορατό. Έτσι, όταν ο ελεγκτής κάνει κλικ στο κουμπί r τότε η εφαρμογή αναζητά με ένα βρόγχο τα layers που είναι ενσωματωμένα στον βασικό χάρτη και έχουν ως ιδιότητα title την τιμή 'Aircraft Route' και τα κάνει με την μέθοδο setVisible ορατά. Αυτό επιτυγχάνεται με την παρακάτω συνθήκη

```
if (map.getLayers().getArray()[i].get('title') == 'Aircraft Route')
{
    map.getLayers().getArray()[i].setVisible(routeVisibility);
}
```

Με την μέθοδο getLayers έχουμε πρόσβαση σε όλα τα layers που είναι ενσωματωμένα στο αντικείμενο map και στην συνέχεια ψάχνουμε στο array που σχηματίζουν.

Ο χρωματισμός της διαδρομής γίνεται με τυχαίο χρώμα κατά την δημιουργία του layer 'Aircraft Route'.

Με το κουμπί Z η εφαρμογή μας προσφέρει στον ελεγκτή ένα εργαλείο το οποίο διαθέτουν και οι εφαρμογές radar των ελληνικών αεροδρομίων. Κάνοντας κλικ στο κουμπί αυτό εμφανίζονται λευκοί ομόκεντροι κύκλοι γύρω από το αεροδρόμιο εργασίας του ελεγκτή. Πατώντας το πλήκτρο Z συνεχόμενα, η εφαρμογή μεταβαίνει διαδοχικά σε τέσσερις καταστάσεις, των 12 κύκλων, των 6 κύκλων, των 2 κύκλων και χωρίς κύκλους. Το εργαλείο αυτό χρησιμεύει στον ελεγκτή εναέριας κυκλοφορίας στο να αναγνωρίζει οπτικά την απόσταση που απέχει ένα αεροσκάφος που έχει

απογειωθεί από το αεροδρόμιο ή που κατευθύνεται προς αυτό για να προσγειωθεί. Το χρώμα των κύκλων είναι λευκό για τον πρώτο default βασικό χάρτη με τις πράσινες αποχρώσεις και κόκκινο για τον δεύτερο βασικό χάρτη με τις γαλάζιες και λευκές αποχρώσεις. Η εξωτερική περίμετρος αντιστοιχεί πάντα στα 60 ναυτικά μίλια. Έτσι αν έχουμε δύο κύκλους, οι περίμετροί τους απέχουν μεταξύ τους 30 ναυτικά μίλια, αν έχουμε 6 κύκλους, απέχουν 10 ναυτικά μίλια και αν έχουμε 12 κύκλους απέχουν 5 ναυτικά μίλια.

Για να επιτύχουμε την κυκλική εναλλαγή μεταξύ των διάφορων καταστάσεων, χρησιμοποιούμε μια global μεταβλητή την `aerodromeCircleState` ενώ έχουμε ήδη δημιουργήσει το αντικείμενο layer `aerodromeCircleLayer` που περιέχει τους κύκλους. Όταν ξεκινά η εφαρμογή η default τιμή της `aerodromeCircleState` είναι 0 και του layer `aerodromeCircleLayer` είναι μη ορατό. Όταν ο ελεγκτής κάνει κλικ στο κουμπί Z, η callback συνάρτηση του κουμπιού ενεργοποιείται και ελέγχει αν η μεταβλητή είναι ίση με 3 και τότε την κάνει 0, θέτοντας παράλληλα το layer `aerodromeCircleLayer` στην κατάσταση μη ορατό με τη μέθοδο `setVisible`. Στην αντίθετη περίπτωση την αυξάνει κατά 1, θέτει το layer στην κατάσταση ορατό και καλεί την συνάρτηση `refreshCircleAerodrome()` η οποία βρίσκεται στη γραμμή 228 του αρχείου `layers.js` προκειμένου να ξανασχεδιάσει το layer αφού ο ελεγκτής θέλει να μεταβεί στην επόμενη κατάσταση.

Προκειμένου να σβήσουμε τους κύκλους και να τους ξανασχεδιάσουμε διαφορετικά, ακολουθούμε την παρακάτω διαδικασία μέσα στη συνάρτηση `refreshCircleAerodrome`:

Από το αντικείμενο του layer `aerodromeCircleLayer` καθαρίζουμε το `source`, με τη μέθοδο `clear`, με την εντολή:

```
aerodromeCircleLayer.getSource().clear();
```

Έπειτα δημιουργούμε ένα νέο αντικείμενο `ol.source.Vector` για το layer καθώς και νέο αντικείμενο `style` με χρώμα που αντιστοιχεί στη μεταβλητή `colorCircleAerodromeStroke`, η οποία έχει τιμή ανάλογη του βασικού χάρτη. Ένας κύκλος - όπως και άλλα γεωμετρικά σχήματα, π.χ σημεία, ευθείες - μπορεί να δημιουργηθεί με την μέθοδο `addFeature` η οποία δέχεται όρισμα ένα αντικείμενο `ol.Feature`. Το αντικείμενο αυτό `ol.Feature` έχει ιδιότητα `geometry` με τιμή ένα αντικείμενο `ol.geom.Circle`. Πάντα ένα αντικείμενο `ol.Feature` προστίθεται στο αντικείμενο `ol.source.Vector` όταν πρόκειται για διανυσματικό σχήμα, π.χ κύκλο, ευθεία, σημείο. Κάνοντας `clear` το αντικείμενο `ol.Source` σβήνονται και όλα τα σχήματα που είναι ενσωματωμένα σε αυτό.

Μπορούμε να προσθέσουμε πολλά τέτοια σχήματα (αντικείμενα `ol.Feature`) στο `source` αντικείμενο του αντικειμένου `ol.Layer`. Επειδή σε κάθε κατάσταση δημιουργούμε διαφορετικό αριθμό κύκλων, ο υπολογισμός τους γίνεται με τη βοήθεια του πίνακα `spaceOfCircle=[0,1,2,6]`. Ο πίνακας αυτός δίνει την ανάλογη τιμή `spaceOfCircle[aerodromeCircleState]`, κάθε φορά ανάλογα με τιμή της μεταβλητής `aerodromeCircleState`, στην μεταβλητή `numberOfCircles=(60/(5*spaceOfCircle[aerodromeCircleState]))` η οποία υπολογίζεται αναλόγως και μας δίνει τον αριθμό των κύκλων που θα ζωγραφιστούν. Η εξωτερική διάμετρος παραμένει

σταθερή και αντιστοιχεί σε 60 ναυτικά μίλια. Με έναν βρόγχο με μεταβλητή *i* δημιουργούνται πολλά αντικείμενα `ol.Feature` που έχουν κύκλους με την ανάλογη διάμετρο ίση με $11939 * \text{spaceOfCircle}[\text{aerodromeCircleState}] * i$, όπου *i* η μεταβλητή του βρόγχου και 11939 είναι η τιμή που αντιστοιχεί στο σύστημα συντεταγμένων στα 5 ναυτικά μίλια.

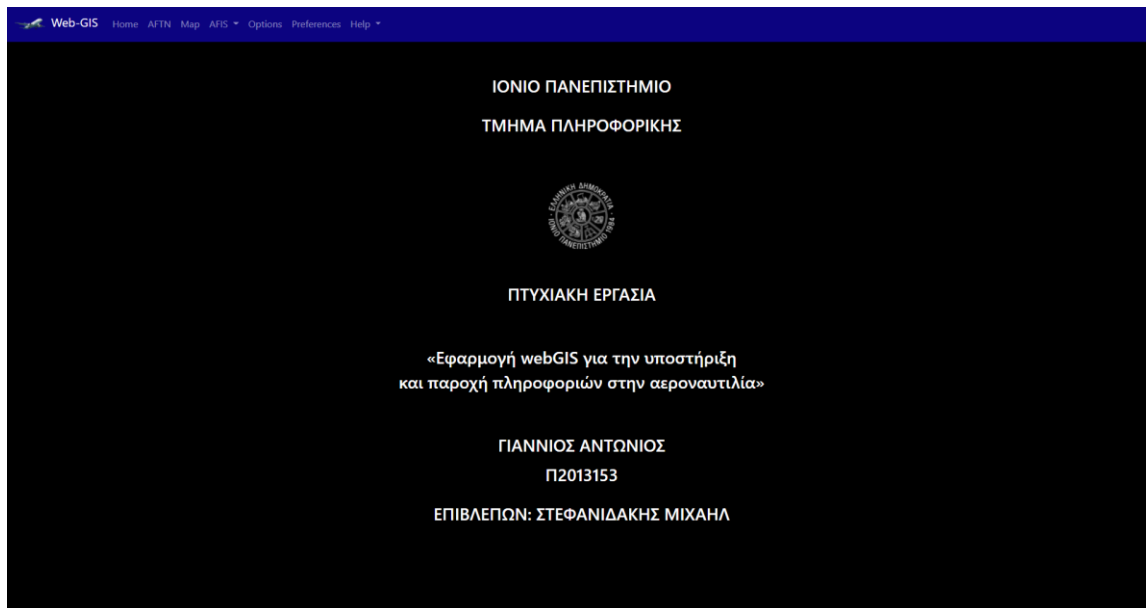
Με το κουμπί B, ο ελεγκτής μπορεί να αλλάξει τον βασικό χάρτη `base layer` χάρτη. Έχει την δυνατότητα να επιλέξει ανάμεσα στους δύο βασικούς χάρτες που έχουμε ήδη μιλήσει. Η εναλλαγή μεταξύ των δύο χαρτών γίνεται με την παρακάτω διαδικασία. Με κάθε κλικ στο κουμπί B δίνουμε στην μεταβλητή `styleJson` την τιμή που αντιστοιχεί στο `link` του `source` των δεδομένων του 1^{ου} χάρτη ή του 2^{ου} χάρτη αντίστοιχα. Έχουμε καταχωρήσει για ευκολία τις τιμές στις μεταβλητές `styleJson1` και `styleJson2` αντίστοιχα. Έπειτα βρίσκουμε την θέση που έχει το `base layer` μέσα στη στοίβα των `layers` που είναι ενσωματωμένα στο αντικείμενο `map`, το αφαιρούμε και έπειτα χρησιμοποιούμε την συνάρτηση `olms` που έχουμε ήδη δει, για να ενσωματώσουμε το νέο `base layer` στο αντικείμενο `map`. Η εύρεση της θέσης του `base layers` μέσα στη στοίβα των υπόλοιπων `layers` που έχει το αντικείμενο `map`, γίνεται με τη χρήση της συνάρτησης `getIndexOfBaseLayer` που υπάρχει στις γραμμές 452-463 του αρχείου `controls.js`. Σε αυτή την συνάρτηση ψάχνονται σειριακά όλα τα `layers` που έχει το αντικείμενο `map`, μέχρι να βρούμε αυτό το `layer` το οποίο έχει την ιδιότητα `mapbox-layers` ίση με την τιμή `'water'`. Αυτό είναι και το `base layer` το οποίο θα πρέπει να σβήσουμε.

Δ΄.7.4.4 Κουμπιά H, A, (full screen)

Στην τέταρτη κατηγορία ανήκουν τα τρία κουμπιά H, A, (full screen).

Με το κουμπί H μεταφερόμαστε στην αρχική σελίδα της εφαρμογής. Αυτό γίνεται με την χρήση του JQuery αποκρύπτοντας τα `div js-map` και `aftn-terminal` και εμφανίζοντας το `div main-page`. Στην επόμενη οθόνη βλέπουμε την αρχική οθόνη της εφαρμογής

.



Εικόνα Δ'15: Αρχική οθόνη της εφαρμογής webGIS

Με το κουμπί Α μεταφερόμαστε στην σελίδα AFTN της εφαρμογής. Επίσης και αυτό γίνεται με την χρήση του JQuery αποκρύπτωντας τα `div js-map` και `main-page` και εμφανίζοντας το `div aftn-terminal`.

Με το κουμπί (full screen) ο φυλλομετρητής μας μεταπίπτει σε κατάσταση πλήρους οθόνης εμφανίζοντας τον χάρτη. Η κατάσταση αυτή του χάρτη θα παραμείνει ακόμα και αν μεταβούμε στην αρχική οθόνη της εφαρμογής όποτε τότε γίνεται έξοδος από την πλήρη οθόνη αλλά μόλις επιστρέψουμε στον χάρτη, ο φυλλομετρητής θα επανέλθει πάλι σε πλήρη οθόνη μέχρι να πατήσουμε το κουμπί fullscreen ή να κάνουμε escape.

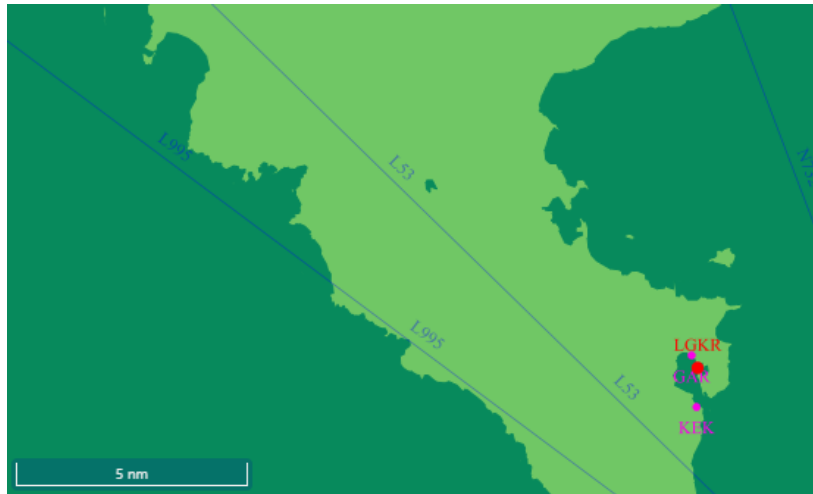
Δ'.7.5 Υπόλοιπες Λειτουργίες

Επίσης πάνω αριστερά στον χάρτη υπάρχουν τα κουμπιά +, - με τα οποία μπορούμε να κάνουμε zoom in και zoom out στον χάρτη.

Κάτω δεξιά υπάρχει κουμπί i, το οποίο εμφανίζει τους contributors της εφαρμογής, δηλαδή OpenStreetMaps και MapTiler.

Στην κάτω αριστερή πλευρά του χάρτη υπάρχει ενδείκτης κλίμακας του χάρτη, σε ναυτικά μίλια. Αυτό γίνεται με την δημιουργία ενός αντικειμένου `ol.control.ScaleLine` του οποίου η ιδιότητά του `units` έχει την τιμή 'nautical', όπως βλέπουμε στην γραμμή 405-406 του αρχείου `controls.js`. Στην συνέχεια ενσωματώνουμε το αντικείμενο αυτό στον χάρτη με τη μέθοδο `addControl`.

Στην επόμενη εικόνα βλέπουμε τον ενδείκτη κλίμακας του χάρτη.



Εικόνα Δ΄16: Ενδεικτής κλίμακας του χάρτη

Επίσης χρησιμοποιούμε ένα κρυφό μηχανισμό για να βρίσκουμε τις γεωγραφικές συντεταγμένες που έχει το ποντίκι μας πάνω στον χάρτη. Αυτό επιτυγχάνεται με την δημιουργία του αντικειμένου `ol.control.MousePosition`, και την ενσωμάτωσή του στο κρυφό `div mouse-position` προκειμένου να μην είναι ορατό στον χρήστη. Με τον τρόπο αυτό μπορούμε να διαβάζουμε τα περιεχόμενα του `div mouse-position` ώστε να γνωρίζουμε το γεωγραφικό σημείο στο οποίο ο χρήστης έχει τον κέρσορα του ποντικιού. Αυτό καθορίζεται στις γραμμές 308-315 του αρχείου `controls.js`.

Δ΄.7.6 Μηχανισμός Μετακίνησης Φορμών

Στο αρχείο `cursors.js` περιέχονται οι μηχανισμοί με τους οποίους ο ελεγκτής μπορεί να μετακινήσει τις φόρμες της εφαρμογής όπου αυτός επιθυμεί προκειμένου να μην τον εμποδίζουν στην ορατότητά του. Η μετακίνηση όλων των φορμών γίνεται ουσιαστικά με τον ίδιο μηχανισμό ο οποίος επαναλαμβάνεται για όλες τις φόρμες.

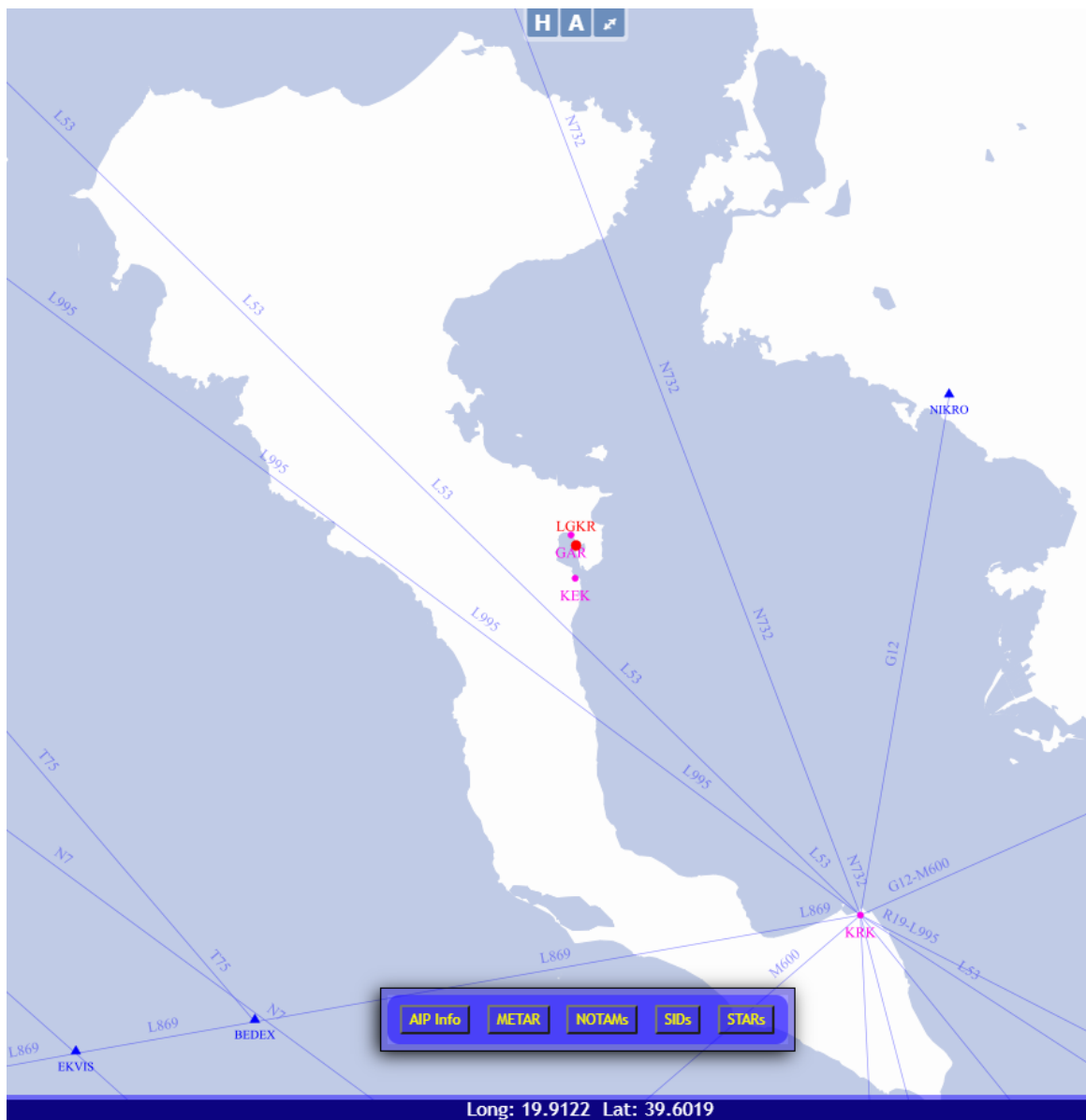
Σε κάθε φόρμα υπάρχει στην κορυφή της ένα `label` που έχει ως `class` την τιμή `move`. Όταν ο χρήστης κάνει `mousedown` στην περιοχή αυτή, τότε μια μεταβλητή για κάθε φόρμα γίνεται `true`. Π.χ για την φόρμα `FPL` η μεταβλητή `flpHeaderIsPressed` γίνεται `true`. Αντίστοιχα μόλις ο χρήστης κάνει `mouseup` τότε η μεταβλητή γίνεται `false`. Έτσι γνωρίζουμε αν ο χρήστης κρατάει πατημένο το αριστερό πλήκτρο του `mouse` πάνω στην περιοχή `move` με σκοπό την μετακίνηση της φόρμας. Έτσι με τη χρήση του `jQuery` κάνουμε έλεγχο αν ο χρήστης μετακινεί το ποντίκι του πάνω σε μια φόρμα και ταυτόχρονα έχει πατημένο το αριστερό κουμπί του `mouse` πάνω στην περιοχή `move` της φόρμας αυτής, τότε με κατάλληλη αλλαγή των ιδιοτήτων `left` και `top` του `CSS` της φόρμας, επιτυγχάνουμε την μετακίνησή της. Χρησιμοποιούμε μεταβλητές για να υπολογίσουμε την θέση

του ποντικιού στην οθόνη και παράλληλα όταν ο χρήστης ξεκινήσει να μετακινεί την φόρμα, αλλάζουμε την θέση του κέρσορα και τον τοποθετούμε στην κορυφή και στην μέση του πλάτους της φόρμας. Επίσης για αισθητικούς λόγους η κάθε φόρμα είναι διπλή, δηλαδή αποτελείται και από μια δεύτερη φόρμα που βρίσκεται πίσω από την κύρια και είναι ελάχιστα pixels μεγαλύτερη προς όλες τις διευθύνσεις. Γι αυτό το λόγο μετακινούμε και τις δύο φόρμες ταυτόχρονα. Π.χ η φόρμα FPL (class name: fpl-form) έχει στο πίσω μέρος της την φόρμα (class name: fpl-form-margin).

Δ'.7.7 Φόρμα Επιλογής

Ο ελεγκτής κάνοντας δεξί κλικ πάνω σε ένα αεροδρόμιο, με μια μικρή απόκλιση στις συντεταγμένες του γύρω από αυτό, όπως αναλύσαμε στο κεφάλαιο του διακομιστή node.js, εμφανίζεται μια μικρή φόρμα (φόρμα επιλογής) στην κάτω πλευρά της οθόνης η οποία περιλαμβάνει τις παρακάτω επιλογές: AIP Info, METAR, NOTAMs, SIDs, STARs.

Στην επόμενη εικόνα βλέπουμε την φόρμα επιλογής



Εικόνα Δ΄17: Φόρμα επιλογής

Αυτό επιτυγχάνεται με τη χρήση του JQuery, στις γραμμές 15-22, του αρχείου `splashes.js`. Με την μέθοδο `preventDefault` αποτρέπουμε τον φυλλομετρητή να εμφανίσει το default μενού για το δεξί κλικ. Στην συνέχεια λαμβάνουμε τις συντεταγμένες στις οποίες έκανε δεξί κλικ ο χρήστης, μέσω του κρυφού μηχανισμού που τις στέλνει στο κρυφό `div mouse-position`. Τις χωρίζουμε και τις εξάγουμε με την χρήση των μεθόδων `replace`, `split` και `parseFloat` τοποθετώντας τις μέσα σε δύο μεταβλητές και τις αποστέλλουμε ως `queries` στον διακομιστή `node.js` στο `route path /coordsToAerodrome`. Στη συνέχεια κάνουμε έλεγχο στο `response JSON` που επιστρέφει ο `node.js` και αν το πεδίο `status` είναι διάφορο του `'notfound'`, το οποίο σημαίνει ότι οι συντεταγμένες αντιστοιχούν σε κάποιο αεροδρόμιο, εμφανίζουμε την φόρμα επιλογής `class name: selection form`. Ταυτόχρονα αποκρύπτουμε τις φόρμες `METAR`, `AIP`, `NOTAM` και `IMAGE` στην

περίπτωση που ήταν ήδη ανοικτές από προηγούμενη επιλογή του χρήστη σε κάποιο άλλο αεροδρόμιο. Όλα τα παραπάνω γίνονται με τη χρήση του JQuery.

Στην συνέχεια αν ο χρήστης κάνει αριστερό κλικ σε οποιοδήποτε σημείο του χάρτη τότε αποκρύπτεται η φόρμα επιλογής. Αυτό έγινε με σκοπό να μην χρειαστεί να χρησιμοποιήσουμε εικονίδιο διαγραφής της φόρμας και έτσι να είναι πιο σύντομο για τον χρήστη να την ακυρώσει, σε περίπτωση που την εμφάνισε κατά λάθος. Η λειτουργία αυτή επιτελείται στις γραμμές 8-13, του αρχείου splashes.js.

Δ'.7.8 Φόρμα AIP Info

Ο χρήστης κάνοντας κλικ στην επιλογή AIP Info, αποκρύπτεται η φόρμα επιλογής και εμφανίζεται η φόρμα AIP Info. Σε αυτήν, ο χρήστης μπορεί να δει πληροφορίες οι οποίες είναι δημοσιευμένες στο εγχειρίδιο AIP και αφορούν το συγκεκριμένο αεροδρόμιο. Οι πληροφορίες αυτές είναι: Το όνομα και ο τίτλος του αεροδρομίου, το υψόμετρό του σε σχέση με το επίπεδο της θαλάσσης, η διεύθυνση, ο τηλεφωνικός αριθμός, ο αριθμός FAX, η διεύθυνση AFTN, η ιστοσελίδα του αεροδρομίου, το e-mail, το είδων των καυσίμων, πληροφορίες για τις εταιρείες των καυσίμων, η κατηγορία πυρόσβεσης, το πλάτος των τροχόδρομων και τα άκρα του διαδρόμου προσγείωσης απογείωσης . Η φόρμα αυτή όπως και όλες οι υπόλοιπες είναι μετακινήσιμη και διαθέτει εικονίδιο απόκρυψης πάνω δεξιά. Η εμφάνιση των πληροφοριών γίνεται με την κλήση της συνάρτησης showQuery. Οι παραπάνω λειτουργίες επιτελούνται στις γραμμές 43-49 του αρχείου splashes.js.

Στην επόμενη εικόνα βλέπουμε την φόρμα AIP.



Εικόνα Δ΄18: Φόρμα AIP Info

Η συνάρτηση showQuery, γραμμές 184-190, αρχείο splashes.js, αποστέλλει στον διακομιστή node.js, στο route path /info το όνομα του αεροδρομίου ως query και τις πληροφορίες που αποστέλλει πίσω ο διακομιστής τις περνάει ως όρισμα στην συνάρτηση tooltipHtml. Με τη σειρά της η συνάρτηση tooltipHtml, γραμμές 192-207 δημιουργεί έναν πίνακα, ο οποίος εμφανίζεται στο div air-info-text της φόρμας AIP με την μέθοδο html.

Δ΄.7.9 Φόρμα METAR

Ο ελεγκτής μόλις κάνει κλικ στην επιλογή METARs , αποκρύπτεται η φόρμα επιλογής και εμφανίζεται η φόρμα METARs (γραμμές 52-59, του αρχείου splashes.js). Στη συνέχεια καλείται η

συνάρτηση showMetar (γραμμές 164-170) η οποία αποστέλλει στο route path /notam του node.js το όνομα του αεροδρομίου ως query και στην συνέχεια λαμβάνει από αυτόν τα αποτελέσματα τα οποία τα εμφανίζει στο div metar-form-text με την μέθοδο html.

Στην επόμενη εικόνα βλέπουμε την φόρμα METAR.



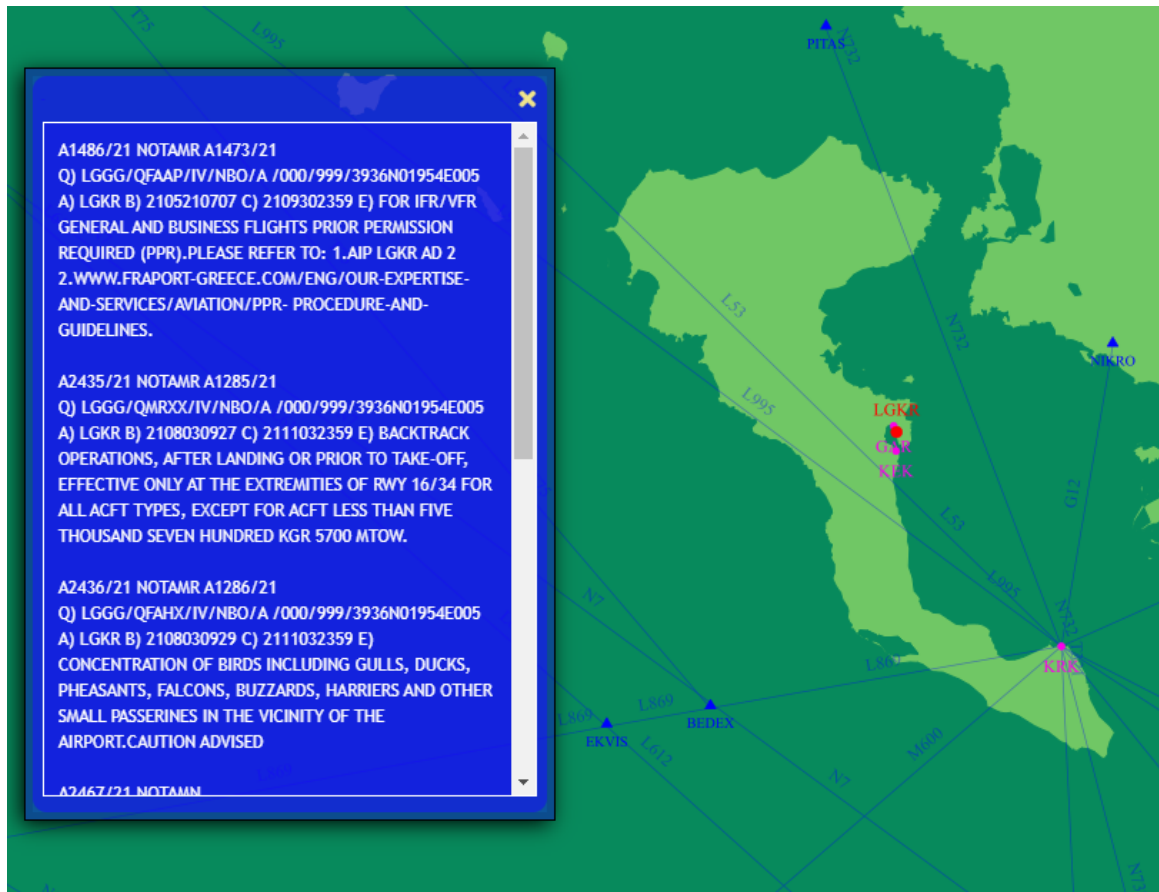
Εικόνα Δ'19: Φόρμα METAR

Δ'.7.10 Φόρμα NOTAM

Ο χρήστης κάνοντας κλικ στην επιλογή NOTAMs , παρόμοια με πριν, εμφανίζεται η φόρμα NOTAMs, όπως βλέπουμε στις γραμμές 61-68, του αρχείου splashes.js. Έπειτα καλείται η συνάρτηση showNotam η οποία βρίσκεται στις γραμμές 173-181 του ίδιου αρχείου. Η συνάρτηση showNotam αποστέλλει στο route path /notam του node.js το όνομα του αεροδρομίου ως query

και στην συνέχεια λαμβάνει από αυτόν τα αποτελέσματα τα οποία τα εμφανίζει στο div notam-form-text με την μέθοδο html. Εδώ χρησιμοποιούμε ένα τέχνασμα με το οποίο μέσω ενός βρόγχου σαρώνουμε σειριακά τα αποτελέσματα, αντικαθιστώντας τον χαρακτήρα \n με το html tag
 προκειμένου τα αποτελέσματα να εμφανιστούν σωστά έχοντας διπλό κενό μεταξύ δύο NOTAM ώστε να είναι πιο ευανάγνωστα.

Στην επόμενη εικόνα βλέπουμε τη φόρμα NOTAM.



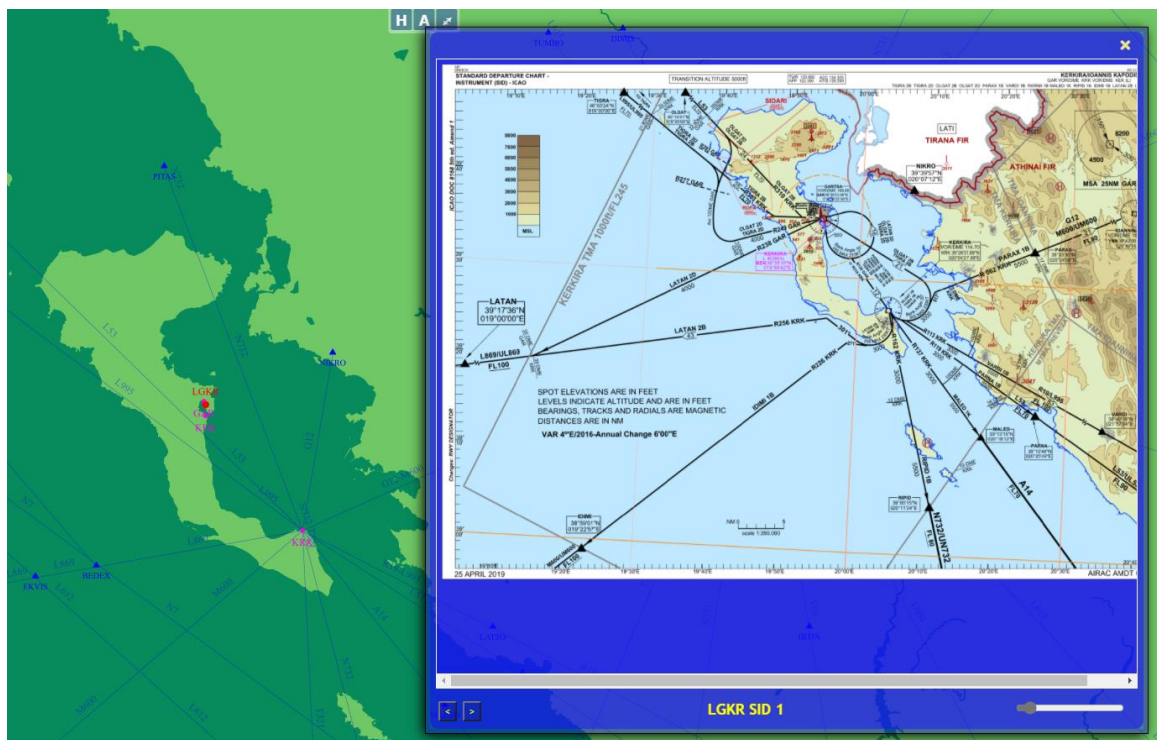
Εικόνα Δ΄20: Φόρμα NOTAM

Δ΄.7.11 Φόρμα IMAGE

Οι επόμενες δύο επιλογές SID και STAR λειτουργούν με παρόμοιο τρόπο, εμφανίζοντας την ίδια φόρμα IMAGE, μέσα στην οποία απεικονίζεται μια από τις διαδικασίες Standard Instrument Departure (S.I.D) ή Standard Arrival (ST.AR) οι οποίες είναι δημοσιευμένες στο εγχειρίδιο AIP και αφορούν το συγκεκριμένο αεροδρόμιο. Στο κάτω μέρος της φόρμας IMAGE υπάρχουν δύο κουμπιά < και > αντίστοιχα και ένα slider. Επίσης εμφανίζεται ο τίτλος της εικόνας στο μέσον της κάτω πλευράς της φόρμας IMAGE. Με αυτά τα δύο κουμπιά ο ελεγκτής μπορεί να μετακινηθεί από την ελάχιστη μέχρι την μέγιστη εικόνα. Επίσης το slider δίνει την δυνατότητα στον ελεγκτή να κάνει zoom in και zoom out στην εικόνα. Τα παραπάνω επιτελούνται στις γραμμές 70-82 του

αρχείου `splashes.js`. Η ανάκτηση των εικόνων γίνεται από τις συναρτήσεις αντίστοιχα `showSID` και `showSTAR` (γραμμές 129-161 του αρχείου `splashes.js`). Οι συναρτήσεις αυτές στέλνουν στον διακομιστή `node.js` ως `query` το όνομα του αεροδρομίου στα `route path /GetSIDSIDContent` και `/GetSIDSTARContent` αντίστοιχα και λαμβάνουν τα αποτελέσματα, καλώντας στη συνέχεια τη συνάρτηση `showImage` (γραμμές 112-126) περνώντας ως όρισμα τον αριθμό της τρέχουσας εικόνας που είναι αποθηκευμένος στην μεταβλητή `activeImage`. Έπειτα η συνάρτηση `showImage` ελέγχει την μεταβλητή `STARImages` για να διαπιστώσει για ποια εικόνα πρόκειται (`SID` ή `STAR`) και στη συνέχεια συνθέτει πλήρες `route path` μαζί με το όνομα του αρχείου που αποτελεί την εικόνα που θα ζητήσει από τον `node.js`. Στη συνέχεια στέλνει το όνομα του αρχείου ως `query` στο αντίστοιχο `route path` (το `route path` αντιπροσωπεύεται από την μεταβλητή `stringQuery`). Έπειτα λαμβάνει την εικόνα σε τύπο δεδομένων `base64 string` και την εμφανίζει στο `div image` κάνοντας την κατάλληλη μετατροπή σε αρχείο `png`. Επίσης στο `div image-label` εμφανίζει τον αριθμό που αντιπροσωπεύει την εικόνα.

Στην παρακάτω εικόνα βλέπουμε τη φόρμα IMAGE.



Εικόνα Δ'21: Φόρμα IMAGE

Δ'.7.12 Εμφάνιση Ώρας UTC και Χρονολογίας UTC

Ακόμη στο κάτω μέρος της οθόνης υπάρχει μια μπάρα πάνω στην οποία εμφανίζονται στο αριστερό της μέρος, η ημερομηνία και η ώρα σε χρόνο UTC, σε format κατά ICAO. Οι λειτουργίες αυτές επιτελούνται στο αρχείο `splashes.js` στις γραμμές 220-229, με τη χρήση

jQuery. Η ημερομηνία και η ώρα εμφανίζονται στα div utc-date και utc-time αντίστοιχα αφού υπολογιστεί πρώτα η τιμή τους από τις αντίστοιχες συναρτήσεις getUTCdateICAO και getUTCtime. Οι συναρτήσεις αυτές βρίσκονται στις γραμμές 1-36 του αρχείου timeUTC.js.

Μέσα στις συναρτήσεις αυτές γίνεται χρήση των μεθόδων getUTCHours, getUTCMinutes, getUTCSeconds, getUTCDate, getUTCMonth, getUTCFullYear του αντικειμένου Date για να υπολογιστούν οι τελικές τιμές. Επίσης χρησιμοποιούμε την συνάρτηση callibrateZero για να προσθέσουμε ένα μηδενικό στα μονά ψηφία, π.χ 5 15 η ώρα, θα γίνει 05 15 η ώρα. Επιπλέον κάνουμε χρήση της μεθόδου substring για να περικόψουμε την χρονολογία getUTCFullYear στα δύο τελευταία ψηφία. Π.χ το έτος 2021 θα γίνει 21.

Ακόμη με τη χρήση της συνάρτησης setInterval, ανανεώνονται η ώρα και η χρονολογία, έπειτα από διάστημα ενός δευτερολέπτου. (Γραμμή 228, αρχείο splashes.js)

Δ΄.7.13 Φόρμα FPL

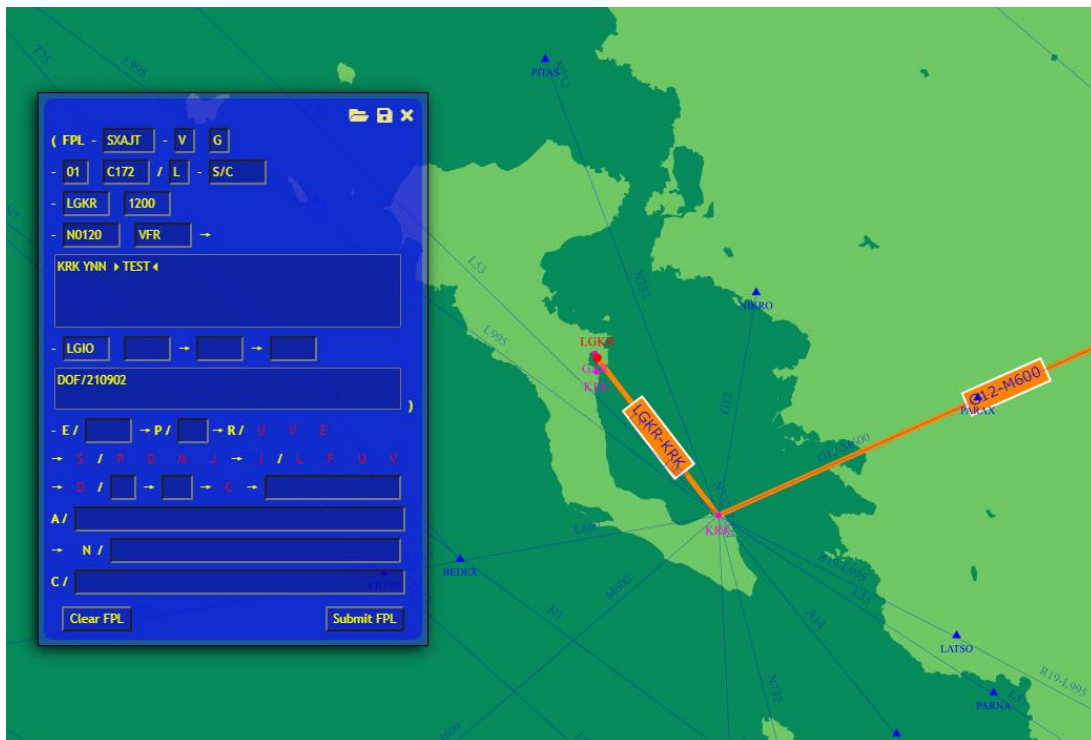
Δ΄.7.13.1 Έλεγχος Τιμών

Όπως αναφέραμε παραπάνω ο ελεγκτής κάνοντας κλικ στο κουμπί F, εμφανίζει την φόρμα FPL. Πάνω στη φόρμα αυτή υπάρχουν πεδία που αντιστοιχούν στα αντίστοιχα πεδία μιας φόρμας σύμφωνα με το έντυπο ICAO Doc 4444. [54] Ο ελεγκτής μπορεί να συμπληρώσει τα πεδία της φόρμας με ένα σχέδιο πτήσης. Το κάθε πεδίο έχει συγκεκριμένους περιορισμούς ως προς το πλήθος των χαρακτήρων που μπορεί να δεχθεί όπως και ως προς το ποιοι θα είναι αυτοί οι χαρακτήρες. Για παράδειγμα, το πεδίο type of flight μπορεί να δεχθεί μόνο ένα χαρακτήρα και αυτός μπορεί να είναι είτε I, είτε V. Η εφαρμογή μας κάνει έλεγχο τιμών σε συγκεκριμένα πεδία, όπως του πεδίου του αεροδρομίου αναχώρησης, του αεροδρομίου άφιξης, του εναλλακτικού αεροδρομίου και του δεύτερου εναλλακτικού αεροδρομίου. Αν ο ελεγκτής εισάγει ένα όνομα που έχει λιγότερο από 4 χαρακτήρες, όσοι δηλαδή πρέπει να είναι σύμφωνα με την ονοματοδοσία κατά ICAO, τότε το background χρώμα του πεδίου γίνεται κόκκινο. Το ίδιο συμβαίνει αν ο ελεγκτής δώσει ένα όνομα αεροδρομίου το οποίο δεν υπάρχει ως καταχώρηση στον πίνακα aerodromes της βάσης δεδομένων PostgreSQL. Η λειτουργία αυτή επιτελείται αντίστοιχα στις συναρτήσεις updateDepartureAerodromeField, updateDestinationAerodromeField και στις συναρτήσεις jQuery, των γραμμών 162-197. Στις συναρτήσεις updateDepartureAerodromeField και updateDestinationAerodromeField γίνεται έλεγχος αν το όνομα του πεδίου είναι μικρότερο από 4 χαρακτήρες οπότε σε αυτή την περίπτωση το background γίνεται κόκκινο. Αν δεν

συμβαίνει αυτό, αποστέλλεται το όνομα του πεδίου ως query στο route path aerodrome του διακομιστή node.js. Αν δεν ανήκει στον πίνακα aerodromes της βάσης δεδομένων τότε το background γίνεται επίσης κόκκινο. Όταν η τιμή του πεδίου είναι αποδεκτή, οι συντεταγμένες του αεροδρομίου, τις οποίες επιστρέφει ως απάντηση ο διακομιστής node.js καταχωρούνται σε array fplDestinationAerodromeCoordinates και fplDepartureAerodromeCoordinates αντίστοιχα, που θα χρησιμεύσουν για την εμφάνιση της προεπισκόπησης του σχεδίου πτήσης στον χάρτη. Έπειτα καλείται η συνάρτηση updatePath (γραμμές 274-340 του αρχείου FPL.js) η οποία σχεδιάζει την προεπισκόπηση του σχεδίου πτήσης πάνω στο χάρτη.

Η εφαρμογή εκτός από τα δύο προηγούμενα πεδία, κάνει έλεγχο τιμών στην ώρα αναχώρησης και στην διαδρομή route του σχεδίου πτήσης. Το πεδίο της διαδρομής ελέγχεται ως προς το αν τα εισαχθέντα σημεία waypoints είναι υπαρκτά αποδεκτά σημεία waypoints VFR ή IFR που υπάρχουν καταχωρημένα στους πίνακες waypoints και waypoints_VFR. Στην περίπτωση που κάποιο σημείο δεν είναι αποδεκτό μαρκάρεται με βέλη αριστερά και δεξιά για να δείξει στον ελεγκτή ότι θα πρέπει να αλλαχθεί. Η λειτουργία αυτή επιτελείται στην συνάρτηση updateRouteField στις γραμμές 203-260 του αρχείου FPL.js. Στη συνάρτηση αυτή η τιμή του πεδίου route χωρίζεται στα σημεία από τα οποία αποτελείται και δημιουργείται με αυτά ο πίνακας routeArray. Στη συνέχεια κάθε τιμή του πίνακα αυτού αποστέλλεται στο /coordinates route path για να βρεθεί αν κάθε σημείο είναι αποδεκτό και οι συντεταγμένες του. Αν κάποιο σημείο δεν είναι αποδεκτό τότε στον πίνακα routeArray η τιμή του αλλάζει και προστίθενται βέλη αριστερά και δεξιά της τιμής. Οι συντεταγμένες όλων των σημείων αποθηκεύονται στον πίνακα fplRouteCoordinates. Επειδή η αποστολή πολλαπλών τιμών στον node.js και η απάντηση από αυτόν, γίνεται με ασύγχρονο τρόπο, χρησιμοποιείται callback συνάρτηση για την σάρωση του πίνακα routeArray και επίσης μια χρονική καθυστέρηση στην ανασύνθεση της τιμής που θα εμφανιστεί στο πεδίο route του FPL με τις μαρκαρισμένες με βέλη τιμές. (γραμμές 245-257, αρχείο FPL.js)

Στην παρακάτω εικόνα βλέπουμε τον έλεγχο τιμών με τον οποίο μαρκάρεται με βέλη αριστερά και δεξιά το λανθασμένο σημείο TEST.



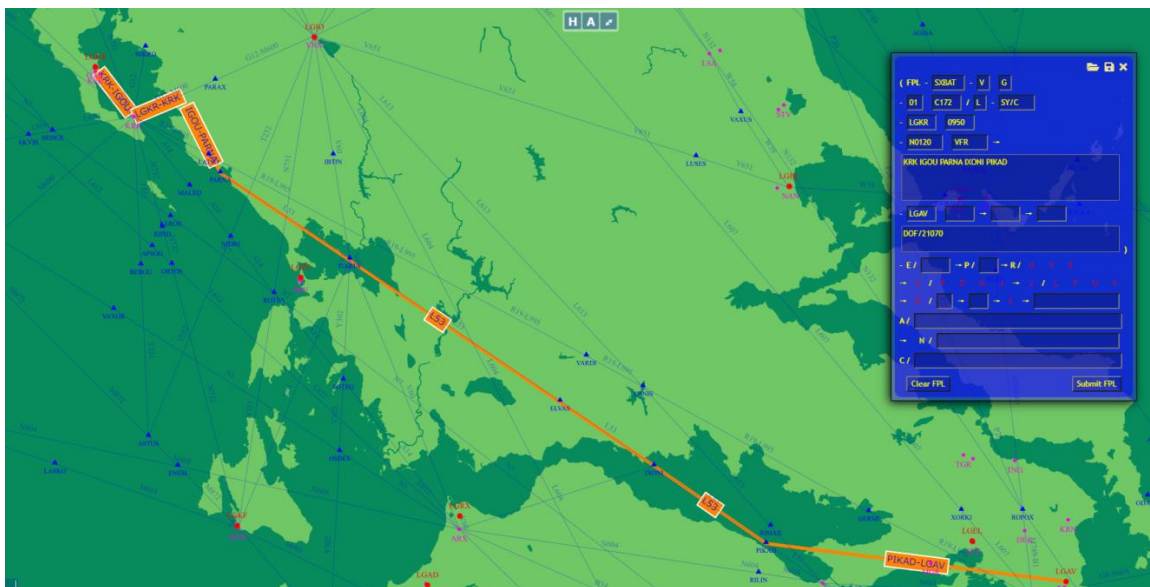
Εικόνα Δ΄22: Έλεγχος τιμών του πεδίου της διαδρομής

Η ώρα αναχώρησης ελέγχεται να ώστε να είναι μέσα στις αποδεκτές τιμές από 0000 έως 2359. Στην περίπτωση που η τιμή της ώρας δεν είναι αποδεκτή, το background χρώμα του πεδίου γίνεται κόκκινο. Η λειτουργία αυτή επιτελείται στις γραμμές 75-96 του αρχείου FPL.js με τη χρήση του JQuery, ελέγχοντας αν η τιμή του πεδίου είναι 4 χαρακτήρες και έπειτα αν είναι μέσα στα αποδεκτά όρια. Αν δεν συμβαίνει αυτό, τότε αλλάζουμε την CSS ιδιότητα background-color σε κόκκινο χρώμα, red.

Δ΄.7.13.2 Λειτουργία Προεπισκόπησης

Η εφαρμογή μας διαθέτει μια λειτουργία κατά την οποία όταν λάβει αρκετά στοιχεία, δηλαδή το αεροδρόμιο αναχώρησης, την ώρα αναχώρησης, την διαδρομή, το αεροδρόμιο προορισμού και τη διάρκεια ταξιδιού, τότε εμφανίζει μια προεπισκόπηση του σχεδίου πάνω στο χάρτη. Έτσι ο ελεγκτής μπορεί να δει, σχεδιασμένη την πλήρη διαδρομή που θα ακολουθήσει το αεροσκάφος και ενεργοποιώντας τα κατάλληλα layers του χάρτη να δει από ποιες τερματικές περιοχές, αεροδρόμια και ενδεχόμενα από ποιες επικίνδυνες, στρατιωτικές ή απαγορευμένες περιοχές θα περάσει. Ακόμη, μεταξύ των waypoints εμφανίζεται πινακίδα η οποία αναγράφει τα σημεία που ενώνει το τμήμα αυτό της διαδρομής. Η εμφάνιση της προεπισκόπησης του σχεδίου πτήσης επιτελείται στη συνάρτηση updatePath (γραμμές 274-340).

Για να εμφανίσουμε την προεπισκόπηση αυτή, η οποία θα αποτελείται από ευθείες και πινακίδες, είναι απαραίτητο να δημιουργήσουμε πρώτα ένα layer πάνω στο οποίο θα ενσωματώσουμε τα σχήματα αυτά. Το layer αυτό είναι ένα αντικείμενο `ol.layer.Vector` και δημιουργείται στις γραμμές 16-21, του αρχείου `FPL.js`. Επίσης όπως όλα τα layers προστίθεται στον βασικό χάρτη με τη μέθοδο `addLayer`. [55] Στην επόμενη εικόνα βλέπουμε την προεπισκόπηση του σχεδίου πτήσης του αεροσκάφους με χαρακτηριστικό κλήσης `SXBAT` από το αεροδρόμιο της Κέρκυρας (`LGKR`) στο αεροδρόμιο Ελευθέριος Βενιζέλος (`LGAV`). Παρατηρούμε ότι το τμήμα της διαδρομής `PARNA - GARTA - PIKAD` έχει label `L53`, αφού τα σημεία αυτά ορίζουν τον αεροδιάδρομο αυτό.



Εικόνα Δ'23: Προεπισκόπηση του σχεδίου πτήσης

Μέσα στη συνάρτηση `updatePath`, αρχικά ενώνουμε τους τρεις πίνακες που περιέχουν τις συντεταγμένες του αεροδρομίου αναχώρησης (`fplDepartureAerodromeCoordinates`), της διαδρομής (`fplRouteCoordinates`) και του αεροδρομίου προορισμού (`fplDestinationAerodromeCoordinates`) σε έναν πίνακα (`fplPathCoordinates`). Το ίδιο κάνουμε και με τους πίνακες που περιέχουν τα ονόματα των αντίστοιχων αεροδρομίων αναχώρησης και προορισμού και των σημείων που αποτελούν την διαδρομή, τα οποία τα ενώνουμε σε έναν πίνακα (`fplPathValidNames`). Στη συνέχεια στέλνουμε ανά ζεύγη τα ονόματα των waypoints που αποτελούν τη διαδρομή, ως queries, στο `airway route path` για να δούμε ποια από αυτά ανήκουν σε διάδρομο και ποιο το όνομα αυτού, ώστε να απεικονιστεί στο κατάλληλο label. Και εδώ, λόγω ασύγχρονης λειτουργίας, χρησιμοποιούμε συνάρτηση `callback` για την αποστολή των ονομάτων και χρονική καθυστέρηση για να εξασφαλίσουμε ότι έχουν επιστρέψει όλες οι απαντήσεις από τον `node.js`.

Παράλληλα αποθηκεύουμε τα ονόματα των διαδρόμων για τα αντίστοιχα ζεύγη σημείων στον πίνακα `fplAirways`. Ο πίνακας αυτός είναι ένας πίνακας αντικειμένων, όπου κάθε index του

πίνακα, είναι ένα αντικείμενο που έχει την ιδιότητα `name` που αντιστοιχεί στο όνομα του διαδρόμου, τις ιδιότητες `firstPoint`, `secondPoint` που αντιστοιχούν στις συντεταγμένες του πρώτου και του δεύτερου `waypoint` αντίστοιχα και την ιδιότητα `rotation` που αντιστοιχεί στην κλίση του διαδρόμου.

Στη συνέχεια σαρώνουμε τον πίνακα `fplAirways` και για κάθε `index` δημιουργούμε ένα αντικείμενο `ol.Feature` με ιδιότητα `geometry` η οποία έχει τιμή ένα αντικείμενο `ol.geom.LineString` που αντιστοιχεί σε ευθεία, που ενώνει τα δύο σημεία του `index` αυτού. Το κάθε αντικείμενο `ol.Feature` προστίθεται στο `source` αντικείμενο του `layer` που έχουμε δημιουργήσει για την προεπισκόπηση. (γραμμές 327-338 του αρχείου `FPL.js`). Το `label` που θα εμφανίζει το όνομα του κάθε διαδρόμου, δημιουργείται με την συνάρτηση `fplStyleFunction` (γραμμές 342-368 αρχείου `FPL.js`) η οποία δημιουργεί το `ol.style.Style` αντικείμενο με το οποίο θα «ντυθεί» το αντίστοιχο `ol.Feature` αντικείμενο. [56]

Μέσα στη συνάρτηση `fplStyleFunction` καθορίζουμε το χρώμα της γραμμής των αεροδιαδρόμων προεπισκόπησης, το χρώμα της γραμματοσειράς που θα έχει το `label` και το χρώμα γεμίσματος αυτής. Επίσης καθορίζουμε την κλίση του `label` ώστε να είναι παράλληλο με τον διάδρομο δίνοντας στην ιδιότητα `rotation` του αντικειμένου `ol.style.Text`, την τιμή `rotation` του `index` που αντιστοιχεί στο τμήμα αυτό του αεροδιαδρόμου. [57]

Επίσης για λόγους ευκρίνειας, η πινακίδα αυτή εμφανίζεται μόνο μετά από κάποιο επίπεδο `zoom` `in` στο χάρτη ώστε αν ο ελεγκτής θέλει μια μακρινή προεπισκόπηση να μην τον εμποδίζουν οι πολλές πινακίδες που θα υπάρχουν συνολικά. Αυτή η λειτουργία επιτελείται δίνοντας την τιμή `map.getView().getZoom() > 7 ? name1:"` στην ιδιότητα `text` του αντικειμένου `ol.style.Text`. Αυτή η τιμή σημαίνει ότι το `label` θα έχει κείμενο μόνο αν ο βασικός χάρτης έχει `zoom level` μεγαλύτερο από 7.

Δ΄.7.13.3 Υπόλοιπες Λειτουργίες

Σε ένα σχέδιο πτήσης οι χαρακτήρες είναι πάντα στα κεφαλαία. Η εφαρμογή μας αυτόματα μετατρέπει τους χαρακτήρες που πληκτρολογεί ο ελεγκτής σε κεφαλαία. Η λειτουργία αυτή επιτελείται στις γραμμές 24-36 του αρχείου `FPL.js` με τη χρήση του `JQuery` και των ιδιοτήτων `selectionStart`, `selectionEnd` για να μετακινήσουμε τον δρομέα προς τα πίσω και τη μέθοδο `toUpperCase` για να μετατρέψουμε τους χαρακτήρες σε κεφαλαία.

Επίσης στο πεδίο `other information` του σχεδίου πτήσης, ο ελεγκτής είναι απαραίτητο να γράψει την ημερομηνία πτήσης, με συγκεκριμένο `format`, δηλαδή `DOF/ημερομηνία` κατά `ICAO`. Π.χ `DOF/210805` η οποία αντιστοιχεί στην ημερομηνία 5 Αυγούστου 2021. Η εφαρμογή μας

αυτοματοποιεί την παραπάνω διαδικασία. Όταν ο ελεγκτής κάνει κλικ με το ποντίκι μέσα στο πεδίο other information, τότε δημιουργείται αυτόματα η εγγραφή με το συγκεκριμένο format για την τρέχουσα ημερομηνία, αφού τα περισσότερα σχέδια που εισάγονται αφορούν την ημερομηνία αυτή και σπανίως κάποια άλλη. Αυτή η λειτουργία επιτελείται με τη χρήση του JQuery στις γραμμές 266-271 του αρχείου FPL.js.

Εκτός από τα πεδία που ο ελεγκτής πρέπει να πληκτρολογήσει κάποια τιμή και να την εισάγει, πάνω στη φόρμα FPL υπάρχουν κάποιες τιμές που δεν χρειάζεται να πληκτρολογηθούν. Π.χ συσκευές επικοινωνίας UHF, συσκευές επικοινωνίας VHF, συσκευές επικοινωνίας ELT όπου αναπαρίστανται με τα γράμματα U V E. Σε αυτή την περίπτωση ο ελεγκτής πρέπει να κάνει κλικ πάνω στο γράμμα π.χ U, και τότε αυτό από κόκκινο χρώμα (που σημαίνει ότι δεν περιλαμβάνεται στο σχέδιο πτήσης) θα αλλάξει και θα γίνει κίτρινο (που σημαίνει ότι περιλαμβάνεται στο σχέδιο πτήσης). (γραμμές 116-124, αρχείο FPL.js)

Δ'.7.13.4 Κουμπιά Clear FPL και Submit FPL

Στο κάτω μέρος της φόρμας FPL υπάρχουν δύο κουμπιά, το Clear FPL και το Submit FPL.

Όταν ο ελεγκτής κάνει κλικ στο κουμπί Clear FPL, καθαρίζονται όλα τα πεδία του σχεδίου πτήσης από οποιαδήποτε εγγραφή που έχει κάνει ο ελεγκτής. Η λειτουργία αυτή επιτελείται με την συνάρτηση clearFPL στις γραμμές 575-631, του αρχείου FPL.js.

Όταν ο ελεγκτής κάνει κλικ στο κουμπί Submit FPL, τότε το σχέδιο πτήσης που υπάρχει πάνω στη φόρμα υποβάλλεται στην εφαρμογή και στέλνεται στη βάση δεδομένων και δημιουργείται μια εγγραφή στη φόρμα Electronic Strip Bay, όπως θα δούμε παρακάτω. Η λειτουργία αυτή επιτελείται με τη συνάρτηση formSubmit, που βρίσκεται στις γραμμές 387-471, του αρχείου FPL.js. Μέσα στη συνάρτηση αυτή δημιουργείται ο πίνακας flightPlanArray που περιλαμβάνει τις τιμές από όλα τα πεδία της φόρμας FPL. Επίσης δημιουργείται ο πίνακας flightArray ο οποίος περιλαμβάνει τις τιμές όλων των πεδίων από το πεδίο χαρακτηριστικό κλήσης μέχρι το πεδίο αεροδρόμιο προορισμού. Ο πίνακας flightPlanArray στέλνεται ως query στο /FPL route path του διακομιστή node.js ώστε να καταχωρηθεί το σχέδιο πτήσης στον πίνακα FPL της βάσης δεδομένων postgresQL. Στη συνέχεια ο πίνακας flightArray στέλνεται στο /Flights route path του διακομιστή node.js προκειμένου η πτήση να καταχωρηθεί στον πίνακα flights της βάσης δεδομένων. Ακόμη ο πίνακας flightArray περνάει ως όρισμα στη συνάρτηση addFlightToStripBay προκειμένου η πτήση να καταχωρηθεί στη φόρμα Electronic Strip Bay. Έπειτα κλείνει η φόρμα FPL και εμφανίζεται η φόρμα Electronic Strip Bay.

Πάνω στο πάνω δεξιό μέρος της φόρμα FPL, υπάρχουν τρία εικονίδια, με τα οποία ο ελεγκτής μπορεί αντίστοιχα να φορτώσει ένα αποθηκευμένο σχέδιο πτήσης, να σώσει ένα σχέδιο πτήσης που έχει πληκτρολογήσει ή να κλείσει τη φόρμα FPL.

Δ΄.7.14 Φόρμα Load FPL

Ο ελεγκτής κάνοντας κλικ στο 1^ο εικονίδιο αριστερά, ανοίγει τη φόρμα Load FPL. Στο πάνω μισό μέρος της φόρμας αυτής, εμφανίζονται τα περιεχόμενα του καταλόγου /MyFlightPlans. Ο κατάλογος αυτός περιέχει τα σχέδια πτήσης που έχει αποθηκεύσει ο ελεγκτής. Στο κάτω μισό μέρος της φόρμας, ο ελεγκτής μπορεί να δει σε προεπισκόπηση τα περιεχόμενα ενός αρχείου, μόλις το επιλέξει με το ποντίκι του, χωρίς να χρειαστεί να το φορτώσει. Όταν αποφασίσει ότι θέλει να το φορτώσει κάνει κλικ στο κουμπί Load FPL, η φόρμα κλείνει και το σχέδιο φορτώνεται στα πεδία της φόρμας FPL. Η φόρμα Load FPL, έχει εικονίδιο στο πάνω δεξιό μέρος της με το οποίο ο ελεγκτής μπορεί να την κλείσει. Οι παραπάνω λειτουργίες επιτελούνται στις γραμμές 492-516 του αρχείου FPL.js. Τα περιεχόμενα του καταλόγου /MyFlightPlans διαβάζονται από την συνάρτηση getMyFlightPlansContent που βρίσκεται στις γραμμές 474-490, του αρχείου FPL.js. Το αρχείο που επιλέγει ο ελεγκτής για να κάνει preview στέλνεται στο route path /ReadFPL του διακομιστή node.js. Τα αποτελέσματα φορτώνονται στο div load-fpl-preview. Μόλις ο χρήστης κάνει κλικ στο κουμπί Load FPL, τότε διαγράφονται τα περιεχόμενα της φόρμας FPL και φορτώνονται σε αυτήν τα περιεχόμενα του νέου σχεδίου πτήσης. Οι λειτουργίες αυτές επιτελούνται στις γραμμές 247-253 του αρχείου splashes.js και στις γραμμές 536-542, του αρχείου FPL.js, με την κλήση της συνάρτησης parseFPL, που βρίσκεται στις γραμμές 633-817, του ίδιου αρχείου. Στην συνάρτηση αυτή λαμβάνεται το σχέδιο πτήσης από το div load-fpl-preview της φόρμας Load FPL και με την χρήση της μεθόδου slice, τεμαχίζεται κατάλληλα, απομονώνονται οι επιμέρους τιμές ώστε να φορτωθούν στα αντίστοιχα πεδία της φόρμας FPL.

Στην επόμενη εικόνα βλέπουμε την φόρμα Load FPL.



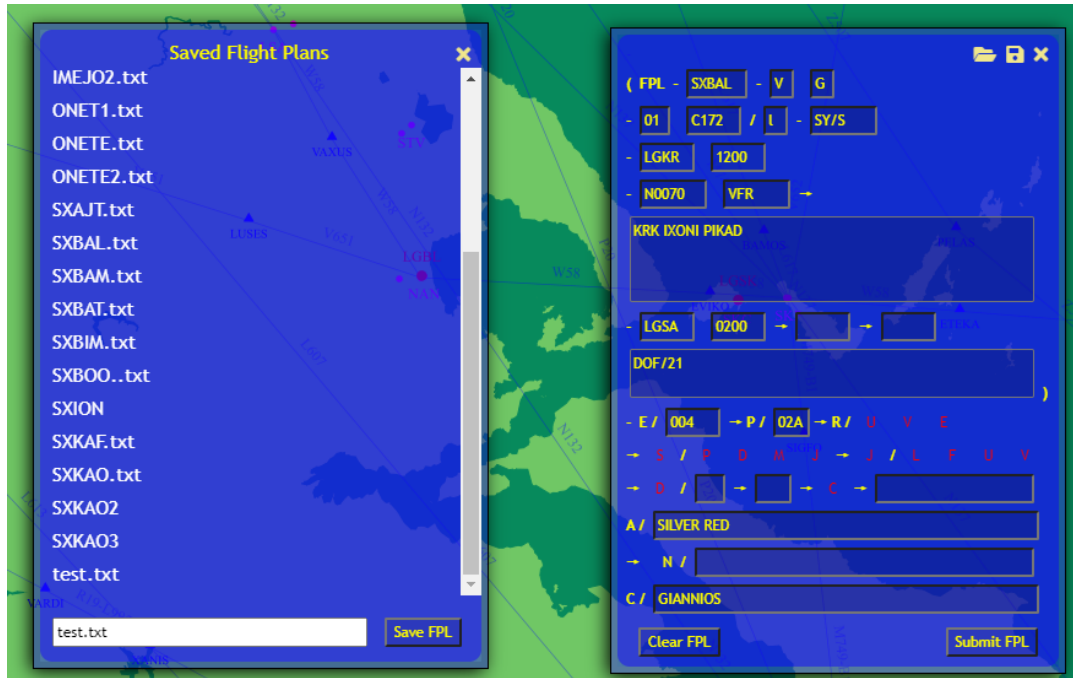
Εικόνα Δ'24: Φόρμα Load FPL

Δ'.7.15 Φόρμα Save FPL

Με τον ίδιο τρόπο, ο ελεγκτής μπορεί να σώσει ένα σχέδιο πτήσης που έχει πληκτρολογήσει, πριν το υποβάλλει με το κουμπί Submit, κάνοντας κλικ στο 2^ο εικονίδιο της φόρμας FPL. Στην περίπτωση αυτή εμφανίζεται η φόρμα Save FPL. Σε αυτή τη φόρμα ο ελεγκτής μπορεί να δει τα περιεχόμενα του καταλόγου /MyFlightPlans και να πληκτρολογήσει στο πεδίο που βρίσκεται στη βάση της φόρμας, ένα όνομα για το νέο σχέδιο πτήσης με το οποίο αυτό θα αποθηκευτεί. Η φόρμα Save FPL, διαθέτει και αυτή εικονίδιο με το οποίο ο ελεγκτής μπορεί να την κλείσει. Οι λειτουργίες αυτές επιτελούνται στις γραμμές 545-554 του αρχείου FPL.js και στις γραμμές 255-262, του αρχείου splash.js. Στην περίπτωση αυτή τα περιεχόμενα του καταλόγου διαβάζονται επίσης με την συνάρτηση `getMyFlightPlansContent /MyFlightPlans`. Μόλις ο ελεγκτής κάνει κλικ στο κουμπί Save FPL τότε καλείται η συνάρτηση `reverseParseFPL` με την οποία γίνεται η αντίστροφη διαδικασία με αυτή της συνάρτησης `parseFPL`. Στη συνάρτηση αυτή δημιουργείται το string `fplToBeSaved` από τις τιμές των επιμέρους τιμών της φόρμας FPL κατά το πρότυπο του ICAO για τα σχέδια πτήσης. Στην συνέχεια με τη χρήση του JQuery στέλνεται ως queries το

string στο /WriteFPL route path μαζί με το όνομα που έχει πληκτρολογήσει ο ελεγκτής προκειμένου να εγγραφεί το αρχείο στον κατάλογο /MyFlightPlans.

Στην επόμενη εικόνα βλέπουμε την φόρμα Save FPL.



Εικόνα Δ΄25: Φόρμα Save FPL

Με το 3^ο εικονίδιο ο ελεγκτής μπορεί να κλείσει την φόρμα FPL. Σε αυτή την περίπτωση όμως ότι έχει πληκτρολογήσει θα χαθεί. Η προεπισκόπηση του σχεδίου πτήσης πάνω στο χάρτη θα παραμείνει μέχρι ο ελεγκτής είτε να πληκτρολογήσει ένα νέο σχέδιο πτήσης είτε μέχρι να κάνει clear όλα τα πεδία της φόρμας.

Δ΄.7.16 Φόρμα Electronic Strip Bay

Όπως αναφέραμε παραπάνω ο ελεγκτής μόλις φορτώσει ή πληκτρολογήσει ένα σχέδιο πτήσης στη φόρμα FPL, υποβάλει το σχέδιο κάνοντας κλικ στο κουμπί Submit FPL. Μόλις γίνει αυτό, κλείνει η φόρμα FPL και ανοίγει η φόρμα Electronic Strip Bay στην οποία εμφανίζονται όλα τα σχέδια πτήσης των αεροσκαφών που είναι έτοιμα να ακολουθήσουν την διαδικασία εκκίνησης, τροχοδρόμησης και απογείωσης. Η φόρμα Electronic Strip Bay σχεδιάστηκε με τέτοιο τρόπο ώστε να είναι όμοια με την αντίστοιχη στα υπάρχοντα συστήματα radar των ελληνικών αεροδρομίων. Κάθε σχέδιο πτήσης που έχει υποβληθεί εμφανίζεται στη φόρμα Electronic Strip Bay ως ταινία (strip). Η ταινία αυτή είναι χωρισμένη σε κάθετα ορθογώνια, μέσα στα οποία αναγράφονται στοιχεία από το σχέδιο πτήσης, όπως ο τύπος του αεροσκάφους, το

χαρακτηριστικό κλήσης, το είδος της πτήσης, η κατηγορία αεροδινών, ο αριθμός squawk. Επίσης στην ταινία αυτή υπάρχει ένα οριζόντιο ορθογώνιο το οποίο αναγράφει τα πρώτα waypoints του σχεδίου πτήσης. Ακόμη υπάρχουν και άλλα κάθετα ορθογώνια τα οποία δεν τα χρησιμοποιούμε σε αυτή την εφαρμογή και παραμένουν κενά.

Η διαδικασία που ακολουθεί ένα αεροσκάφος που πρόκειται να απογειωθεί είναι η παρακάτω: Ο ελεγκτής λίγο πριν την εκκίνηση των κινητήρων του αεροσκάφους από τον πιλότο, επικοινωνεί με το Κέντρο Ελέγχου Περιοχής Αθηνών (Κ.Ε.Π.ΑΘ) και ζητάει από τον συνάδελφο του ελεγκτή εναέριας κυκλοφορίας να του εκδώσει άδεια απογείωσης (clearance). Ο ελεγκτής που βρίσκεται στο Κ.Ε.Π.ΑΘ βλέπει το σχέδιο πτήσης του αεροσκάφους που πρόκειται να αναχωρήσει και ενημερώνει τον ελεγκτή στο τοπικό αεροδρόμιο για τον αριθμό squawk που θα έχει η πτήση αυτή.

Έπειτα από λίγο ο πιλότος εκκινεί τους κινητήρες του και ζητάει άδεια για τροχοδρόμηση. Μόλις τροχοδρομήσει και λίγο πριν μπει στον διάδρομο για απογείωση, ο ελεγκτής εκφωνεί στον πιλότο την άδεια (clearance) που έχει λάβει από το Κ.Ε.Π.ΑΘ και τον ενημερώνει για τον αριθμό squawk που θα πρέπει να βάλει στον transponder του αεροσκάφους καθώς και για άλλα θέματα όπως π.χ την συχνότητα επικοινωνίας του αεροσκάφους μετά την απογείωση με την επόμενη μονάδα εναέριας κυκλοφορίας κ.α. Μόλις το αεροσκάφος απογειωθεί το δευτερεύον radar αναγνωρίζει το αεροσκάφος από τον αριθμό squawk και ενεργοποιεί την πτήση από το Electronic Strip Bay.

Στην εφαρμογή μας έχουμε σχεδιάσει τη δυνατότητα να εισάγει ο ελεγκτής τον αριθμό squawk όπως κάνει και στην πραγματικότητα. Επειδή στην εφαρμογή μας προφανώς η ενεργοποίηση της πτήσης δεν μπορεί να γίνει αυτόματα, έχουμε σχεδιάσει πάνω το κάθε strip ένα οριζόντιο κουμπί σε γκρι απόχρωση με την ένδειξη DEP (από την λέξη departure) το οποίο στην αρχή είναι ανενεργό. Μόλις ο ελεγκτής εισάγει τον αριθμό squawk το κουμπί αυτό γίνεται ενεργό.

Μετά την απογείωση ο ελεγκτής κάνοντας κλικ στο κουμπί αυτό, ενεργοποιεί την πτήση και το αεροσκάφος σχεδιάζεται να ξεκινά την πτήση του στον χάρτη. Το κουμπί με την ενεργοποίηση της πτήσης γίνεται πάλι ανενεργό και αλλάζει σε χρώμα πράσινο όπως είναι και η υπόλοιπη ταινία για να δηλώσει ότι η πτήση είναι πλέον ενεργή.

Ο ελεγκτής έχει την δυνατότητα να κλείσει τη φόρμα Electronic Strip Bay κάνοντας κλικ στο εικονίδιο που υπάρχει επάνω της. Αν επιχειρήσει πάλι να ανοίξει την φόρμα κάνοντας κλικ στο κουμπί S, τα περιεχόμενα της φόρμας παραμένουν τα ίδια και εμφανίζονται ξανά μαζί με την φόρμα όπως και πριν. Οι παραπάνω λειτουργίες επιτελούνται στο αρχείο FDP.js. Το αρχείο αυτό περιέχει την συνάρτηση addFlightToStripBay η οποία, όπως είδαμε πριν, λαμβάνει ως όρισμα μια σειρά από πεδία που αντιστοιχούν στην πτήση που αναμένει να απογειωθεί. Οι τιμές των πεδίων αυτών περνούν στα αντίστοιχα div των ορθογωνίων της ταινίας της συγκεκριμένης πτήσης.

(γραμμές 15-44, αρχείο FDP.js). Μόλις ο ελεγκτής εισάγει τον αριθμό squawk στο αντίστοιχο πεδίο, τότε γίνεται enabled το κουμπί πάνω από την ταινία. (γραμμές 46-51, αρχείο FDP.js)

Δ΄.7.17 Λειτουργία Απεικόνισης Πτήσης

Δ΄.7.17.1 Δημιουργία GPXFile

Στην συνέχεια μόλις ο ελεγκτής κάνει κλικ στο κουμπί που βρίσκεται πάνω από την ταινία τότε στέλνεται στο /getRoute route path του διακομιστή node.js ως query τα στοιχεία της συγκεκριμένης πτήσης (χαρακτηριστικό κλήσης, αεροδρόμιο αναχώρησης, ώρα αναχώρησης και αεροδρόμιο προορισμού) προκειμένου να ληφθεί η διαδρομή την οποία θα ακολουθήσει. Έπειτα καλείται η συνάρτηση createGPXFile με την οποία θα κατασκευαστεί το αρχείο .gpx που θα χρησιμοποιηθεί ως source για την πορεία που θα ακολουθήσει το εικονίδιο της πτήσης στον χάρτη. Ακολουθεί η κλήση των συναρτήσεων createAircraftLayer για την δημιουργία του layer που θα περιέχει την κίνηση του εικονιδίου της πτήσης και animateTheFeature για την δημιουργία του εικονιδίου και της κίνησής του αντίστοιχα.

Η συνάρτηση createGPXFile βρίσκεται στις γραμμες 81-143, στο αρχείο FDP.js. Δέχεται ως ορίσματα τον πίνακα με τα ονόματα των waypoints της διαδρομής, το χαρακτηριστικό κλήσης, το όνομα του αεροδρομίου αναχώρησης, την ώρα αναχώρησης και το όνομα του αεροδρομίου προορισμού. Δημιουργείται ο πίνακας gpxRoute ο οποίος περιέχει με την σειρά το όνομα του αεροδρομίου αναχώρησης, τα ονόματα των waypoints της διαδρομής και το όνομα του αεροδρομίου προορισμού. Στην συνέχεια ο πίνακας gpxRoute σαρώνεται σειριακά και το κάθε όνομα που περιέχεται στο κάθε index στέλνεται ως query στο /coordinates route path του διακομιστή node.js. Έτσι τα ονόματα μετατρέπονται στις αντίστοιχες συντεταγμένες και δημιουργείται ο πίνακας αντικειμένων έχει ιδιότητες name, που αντιστοιχεί στο όνομα του σημείου, lat και lon, που αντιστοιχούν στο γεωγραφικό πλάτος και μήκος. Έπειτα δημιουργείται το string gpxFile που συντίθεται κατά το πρότυπο ενός gpx αρχείου, Το αρχείο gpx είναι XML αρχείο το οποίο χρησιμεύει για την αποθήκευση δεδομένων σε μορφή γεωγραφικών συντεταγμένων. [58]

Μόλις δημιουργηθεί το αρχείο gpx, τότε στέλνεται ως query στο /WriteGPXFile για να αποθηκευτεί ως αρχείο στον κατάλογο /GPXFiles. Επίσης μαζί με αυτό, στέλνεται ως query και το όνομα του αρχείου που θα εγγραφεί και που αποτελείται από το χαρακτηριστικό κλήσης, το αεροδρόμιο αναχώρησης, την ώρα αναχώρησης και το αεροδρόμιο προορισμού. Στην συνέχεια

καλείται η συνάρτηση `activateFlight` που βρίσκεται στις γραμμές 284-287, του αρχείου `fly.js` η οποία τοποθετεί το χαρακτηριστικό κλήσης της πτήσης αυτής στον πίνακα `aircrafts` και το όνομα του αρχείου `gpx` στον πίνακα `routes`. Οι δύο αυτοί πίνακες χρησιμοποιούνται για να γνωρίζει η εφαρμογή ποιο θα είναι το αντίστοιχο αρχείο `gpx` που θα χρησιμοποιηθεί ως `source` για την συγκεκριμένη πτήση.

Δ'.7.17.2 Δημιουργία Layer Αεροσκάφους

Όπως είπαμε πριν, μετά την κλήση της συνάρτησης `createGPXFile`, ακολουθεί η κλήση της συνάρτησης `createAircraftLayer` που βρίσκεται στις γραμμές 118-142, του αρχείου `fly.js`. Στη συνάρτηση αυτή δημιουργείται το `layer` που θα χρησιμοποιήσουμε για να απεικονίσουμε την κίνηση του αεροσκάφους με εικονίδιο. Στην αρχή δημιουργούμε το αντικείμενο `ol.source.Vector` με το οποίο δηλώνουμε ποια δεδομένα θα χρησιμοποιήσουν όλα τα αντικείμενα `ol.Feature` που θα ενσωματώσουμε στο `layer` που θα έχει το συγκεκριμένο `source`.^[59] Τα δεδομένα αυτά αναφέρονται στην διαδρομή του αεροσκάφους. Το αντικείμενο `source` έχει δύο ιδιότητες: την `url` με την οποία καθορίζουμε το αρχείο που θα διαβάσει το `source` και είναι ο πίνακας `routes` με `index` το όρισμα του χαρακτηριστικού κλήσης του αεροσκάφους και την `format` με την οποία καθορίζουμε ότι πρόκειται για αρχείο `gpx`. Έπειτα δημιουργούμε το αντικείμενο `ol.layer.Vector`, το οποίο θα ενσωματώσει επάνω του το εικονίδιο του αεροσκάφους και όποια άλλα αντικείμενα `ol.Feature` χρειαστούν, π.χ τα `labels` με το χαρακτηριστικό κλήσης του αεροσκάφους, την ταχύτητα και το ύψος στο οποία πετά. Το `layer` αυτό έχει ιδιότητα `title` το `string` 'Aircraft' και `id` έναν αύξοντα αριθμό από το 100 και πάνω για την ευκολότερη διαχείρισή του. Επίσης στην ιδιότητα `style` δίνουμε την τιμή της μεταβλητής `style` η οποία καθορίζεται στην αρχή του αρχείου `fly.js` και με αυτό τον τρόπο καθορίζουμε πώς θα «ντυθούν» τα αντικείμενα `ol.Feature`. Η μεταβλητή `style` είναι ένα `array` αντικειμένων `ol.style.Style`. Το `style[0]` αντιστοιχεί στην σκιά που θα εμφανίζει το εικονίδιό μας από κάτω του, το `style[1]` αντιστοιχεί στο αντικείμενο `ol.style.FontSymbol` με το οποίο εμφανίζουμε ένα `font` που αντιστοιχεί σε αεροσκάφος και το `style[1]` αντιστοιχεί σε αντικείμενο `ol.style.RegularShape` με το οποίο εμφανίζεται μια πράσινη κουκίδα ως εικονίδιο του αεροσκάφους. Όπως βλέπουμε το `style` αντικείμενο ενός `layer` μπορεί να είναι ένα `array` από αντικείμενα `styles` και όχι υποχρεωτικά ένα μόνο αντικείμενο. Επίσης στην αρχή του αρχείου `fly.js` έχουμε φτιάξει και άλλες μεταβλητές που περιέχουν αντίστοιχα αντικείμενα `style` και οι οποίες θα χρησιμοποιηθούν για τα `labels` του εικονιδίου του αεροσκάφους.

Δ΄.7.17.3 Δημιουργία Layer Διαδρομής

Αφού φτιάξαμε το layer για το εικονίδιο του αεροσκάφους, μέσα στην ίδια συνάρτηση `createAircraftLayer` φτιάχνουμε στη συνέχεια και ένα επιπλέον αντικείμενο `ol.layer.Vector` το οποίο θα το χρησιμοποιήσουμε για να απεικονίσουμε την πλήρη διαδρομή του αεροσκάφους ως ένα σύνολο από γραμμές πάνω στο χάρτη. Αυτό το layer θα εμφανίζεται μόνο όταν ο ελεγκτής κάνει κλικ στο κουμπί `r`. Δίνουμε στην ιδιότητα `title`, του layer, την τιμή 'Aircraft Route' και στο layer το ανάλογο style με το οποίο θα το «ντύσουμε» και το οποίο είναι ένα αντικείμενο `ol.style.Style` που αντιστοιχεί σε γραμμή με τυχαίο χρώμα. Η τιμή του τυχαίου χρώματος δημιουργείται με την συνάρτηση `getRandomColor` που βρίσκεται στις γραμμές 275-282, του αρχείου `fly.js`.

Πάνω από το αεροσκάφος υπάρχουν τρία labels όπως υπάρχουν και στα πραγματικά radar, που αναφέρουν την ταχύτητα πτήσης του αεροσκάφους, το ύψος στο οποίο πετά και το χαρακτηριστικό κλήσης του. Η ταχύτητα πτήσης είναι αυτή που αναφέρεται στο σχέδιο πτήσης.

Δ΄.7.17.4 Δημιουργία Features

Στη συνέχεια όπως είπαμε πριν, καλείται η συνάρτηση `animateTheFeature` με την οποία θα επιτευχθεί η κίνηση όλων των αντικειμένων `ol.Feature` που δημιουργήσαμε στο layer με ιδιότητα 'Aircrafts'. Μέσα στη συνάρτηση `animateTheFeature`, αρχικά ελέγχουμε ότι το αντικείμενο `ol.source.Vector` είναι σε κατάσταση 'ready', δηλαδή έχει φορτωθεί κανονικά. (γραμμή 146). Έπειτα δημιουργούμε τα αντικείμενα `ol.Feature` που αντιστοιχούν στη σκιά (`shadowFeature`), στο εικονίδιο του αεροσκάφους (`aircraftFeature`), στο label με το χαρακτηριστικό του αεροσκάφους (`callSignLabelFeature`), στο label με το ύψος του αεροσκάφους (`altitudeLabelFeature`) και στο label με την ταχύτητα του αεροσκάφους (`velocityLabelFeature`). Τα αντικείμενα αυτά τα επενδύουμε με τα αντίστοιχα styles, `style[0]` για το `shadowFeature`, `style[1]` ή `style[2]` για το `aircraftFeature` ανάλογα με την επιλογή που έχει κάνει ο ελεγκτής από το μενού Options, `callSignLabelStyle` για το `callSignLabelFeature`, `altitudeLabelStyle` για το `altitudeLabelFeature` και `velocityLabelStyle` για το `velocityLabelFeature`.

Δ΄.7.17.5 Κίνηση Features

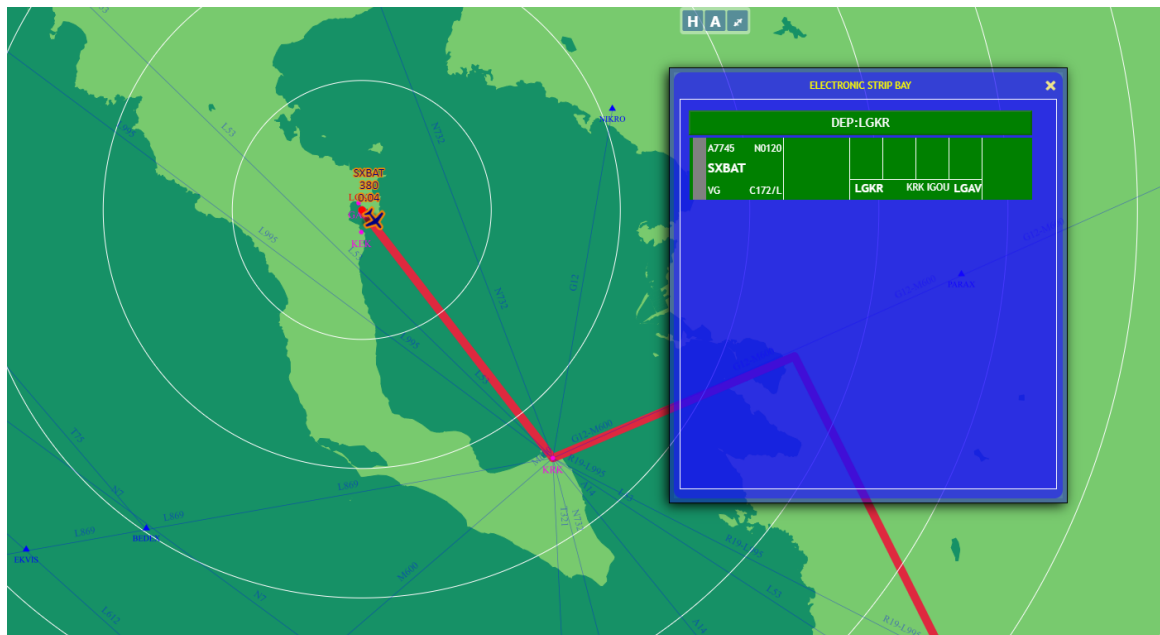
Στη συνέχεια, δημιουργούμε τα αντικείμενα `ol.featureAnimation.Path` με τα οποία θα θέσουμε το εικονίδιο του αεροσκάφους, μαζί με τα υπόλοιπα αντικείμενα `ol.Features` σε κίνηση. [60] Το κάθε αντικείμενο `ol.featureAnimation.Path` έχει:

- την ιδιότητα `path` με την οποία καθορίζουμε την διαδρομή που θα ακολουθήσει το αντικείμενο `ol.Feature` πάνω στο χάρτη
- την ιδιότητα `rotate`, με την οποία το αντικείμενο περιστρέφεται, όταν η ιδιότητα έχει την τιμή `true` (την θέτουμε `true` μόνο για την σκιά και το εικονίδιο του αεροσκάφους)
- την ιδιότητα `easing` που της δίνουμε την τιμή `'linear'` για γραμμική κίνηση του αντικειμένου
- την ιδιότητα `speed` που έχει τιμή από 0 έως 1 και καθορίζει την ταχύτητα με την οποία θα κινηθεί το αντικείμενο και της οποίας δίνουμε τιμή ανάλογη με την τιμή στο πεδίο ταχύτητα πλεύσης του σχεδίου πτήσης

Αφού δημιουργήσουμε όλα τα αντικείμενα `ol.featureAnimation.Path` δηλαδή τα αντικείμενα `animationFeatureShadow`, `aircraftAnimationFeature`, `callSignLabelAnimationFeature`, `altitudeLabelAnimationFeature`, `velocityLabelAnimationFeature` που αντιστοιχούν σε όλα τα αντικείμενα `ol.Feature`, τότε τα θέτουμε σε κίνηση με τη μέθοδο `animateFeature` (γραμμές 233-238, του αρχείου `fly.js`). Επίσης χρησιμοποιούμε ένα `event listener` στις γραμμές 251-270, του αρχείου `fly.js` ώστε όταν η κίνηση μιας πτήσης φτάσει στο τέλος, τότε αφαιρούνται τα `layers` που αντιστοιχούν σε αυτήν ('Aircrafts' και 'Aircraft Route'). Η αφαίρεση αυτή γίνεται με τη μέθοδο `removeLayer` που εφαρμόζεται στο αντικείμενο `map` με όρισμα το `layer` που θέλουμε να αφαιρέσουμε. Ακόμη, αφαιρείται η συγκεκριμένη πτήση από τον πίνακα `flights` της βάσης δεδομένων `postgreSQL` στέλνοντας τα στοιχεία της πτήσης ως `query` στο `/EndOfFlight route path` του διακομιστή `node.js`, όπως διαγράφεται επίσης και το αντίστοιχο αρχείο `gpx` από τον κατάλογο `/GPXFiles`, στέλνοντας το όνομα του αντίστοιχου αρχείου στο `/DeleteGPXFile route path` του διακομιστή `node.js`.

Ο ελεγκτής έχει την δυνατότητα πριν την απογείωση του αεροσκάφους να διαλέξει μέσα από την επιλογή `Options` του μενού της εφαρμογής, αν το `sprite` που απεικονίζει το αεροσκάφος θα είναι μια τελεία σε πράσινο ανοιχτό χρώμα ή θα είναι ένα εικονίδιο αεροσκάφους.

Στην επόμενη εικόνα βλέπουμε την απεικόνιση της πτήσης ενός αεροσκάφους στον χάρτη.



Εικόνα Δ΄26: Απεικόνιση πτήσης στο χάρτη

Δ΄.8 Εγκατάσταση Λογισμικού

Στη συνέχεια θα παρουσιάσουμε εν συντομία τη διαδικασία εγκατάστασης του λογισμικού που απαιτείται για να τρέξει η εφαρμογή webGIS. Το περιβάλλον στο οποίο τρέχει η εφαρμογή μας είναι το λειτουργικό σύστημα Windows 10 Professional 64-bit Build 19041.vb_release.191206-1406 το οποίο τρέχει σε hardware HP 255 G7 Notebook, Ryzen 5, 8GB RAM, 500GB SSD. Οι διαδικασίες εγκατάστασης αναφέρονται στο παραπάνω λειτουργικό σύστημα. Επειδή η εφαρμογή μας είναι web application μπορεί να τρέξει και σε άλλα λειτουργικά συστήματα, αρκεί όμως να ακολουθηθούν οι ανάλογες διαδικασίες εγκατάστασης για τη βάση δεδομένων PostgreSQL και το διακομιστή Geoserver.

Δ΄.8.1 Βάση Δεδομένων PostgreSQL

Για να εγκαταστήσουμε την βάση δεδομένων PostgreSQL, επιλέγουμε από το URL <https://www.enterprisedb.com/downloads/postgres-postgresql-downloads> την έκδοση που επιθυμούμε, καθώς και το λειτουργικό σύστημα στο οποίο θα εγκατασταθεί. Στην εφαρμογή μας χρησιμοποιούμε την έκδοση 13.

Έπειτα κάνουμε download το αντίστοιχο .exe αρχείο, το οποίο τρέχουμε με δικαιώματα administrator. Ο installer μας ενημερώνει σε ποιο φάκελο θα κάνει εγκατάσταση τη βάση δεδομένων μας, καθώς και ποια επιμέρους προγράμματα θα εγκαταστήσει. Εγκαθιστά τον διακομιστή postgresSQL, το γραφικό περιβάλλον διαχείρισης pgAdmin 4, Command Line tools και το εργαλείο διαχείρισης πακέτων (packet manager) Stack Builder

Κατά την διάρκεια της εγκατάστασης ο installer θα μας ζητήσει να δημιουργήσουμε λογαριασμό database superuser δίνοντας ένα password της αρεσκείας μας.

Στη συνέχεια θα μας ζητηθεί από τον διαχειριστή πακέτων Stack Builder να επιλέξουμε ποια πρόσθετα της βάσης δεδομένων θέλουμε να εγκαταστήσει. Στην εφαρμογή μας χρησιμοποιούμε το πρόσθετο PostGIS, και το οποίο το εγκαθιστούμε επιλέγοντας Categories->Spatial Extensions->PostGIS Bundle for PostgreSQL.

Έπειτα ξεκινούμε το γραφικό περιβάλλον διαχείρισης pgAdmin και επιλέγουμε create Server για να δημιουργήσουμε το διακομιστή της βάσης δεδομένων μας. Δίνουμε ένα όνομα που επιθυμούμε π.χ PostgreSQL 13 και στη συνέχεια το port στο οποίο θα επικοινωνεί, στην περίπτωση μας είναι το port 5432 καθώς και τα credentials που φτιάξαμε κατά την εγκατάσταση.

Χρησιμοποιώντας το γραφικό περιβάλλον pgAdmin μπορούμε να φτιάξουμε τη βάση δεδομένων μας π.χ airGISdb δίνοντας το πλήρες σετ εντολών SQL που θα βρούμε στο URL https://github.com/p13gian1/WebGIS/tree/master/postgreSQL_files και το οποίο αντιστοιχεί στο repository Github που βρίσκεται η εφαρμογή webGIS. Μέσα στο σετ περιέχονται έτοιμα όλα τα δεδομένα που χρησιμοποιεί η εφαρμογή.

Δ'.8.2 Διακομιστής Geoserver

Πριν προχωρήσουμε στη διαδικασία εγκατάστασης του διακομιστή Geoserver είναι απαραίτητο να έχουμε ήδη εγκατεστημένο το Java Runtime Environment (JRE) έκδοση 8 ή 11.

Στη συνέχεια από το URL <http://geoserver.org/download/> επιλέγουμε την έκδοση του Geoserver την οποία επιθυμούμε να εγκαταστήσουμε. Στην εφαρμογή μας χρησιμοποιούμε την έκδοση 2.19.0 stable.

Εκτελώντας τον installer από το παραπάνω URL, θα μας ζητηθεί να φτιάξουμε credentials δίνοντας ένα username και ένα password και θα ενημερωθούμε σε ποιο κατάλογο θα γίνει η εγκατάσταση. Έπειτα θα μας ζητηθεί να δώσουμε το port στο οποίο θα επικοινωνεί ο διακομιστής

Geoserver. Στην εφαρμογή μας χρησιμοποιούμε το port 8080

Μόλις ολοκληρωθεί η διαδικασία από τον installer, μπορούμε να εκκινήσουμε τον διακομιστή εκτελώντας το αρχείο startup.bat που βρίσκεται στον κατάλογο /bin ο οποίος βρίσκεται στο path της εγκατάστασης.

Εκκινούμε το γραφικό περιβάλλον διαχείρισης του διακομιστή Geoserver από το URL <http://localhost:8080/geoserver/web/> δίνοντας τα credentials που φτιάξαμε κατά την εγκατάσταση.

Τα styles που δημιουργήσαμε και χρησιμοποιούμε με το διακομιστή Geoserver βρίσκονται στο παρακάτω URL <https://github.com/p13gian1/WebGIS/tree/master/GeoserverStyles>.

Δ΄.8.3 Διακομιστής Node.js

Η διαδικασία εγκατάστασης του διακομιστή node.js ξεκινά από την επιλογή της έκδοσης που θέλουμε να εγκαταστήσουμε από το URL <https://nodejs.org/en/>. Στην εφαρμογή μας χρησιμοποιούμε την έκδοση 14.6.1. Αφού εκτελέσουμε τον installer επιλέγουμε τον κατάλογο στον οποίο θα γίνει η εγκατάσταση.

Στη συνέχεια μέσα από το command line των Windows δίνουμε την εντολή

```
powershell -Command "Start-Process cmd -Verb RunAs"
```

Μέσα από το παράθυρο powershell το οποίο θα ανοίξει, δίνουμε την εντολή

```
npm install -g windows-build-tools
```

Για να εκκινήσουμε το διακομιστή node.js, εκτελούμε με την εντολή node το αρχείο javascript, στο οποίο υπάρχει ο κώδικας του server, μέσα από το command line των Windows. Στην περίπτωση μας το αρχείο μας είναι το app.js και η εντολή είναι

```
node app.js
```

Εξυπακούεται βεβαίως ότι working directory θα είναι ο κατάλογος που περιέχει το αρχείο του node server.

Δ'.8.4 Front End

Τα αρχεία που χρειάζονται για να τρέξει το front end μέρος βρίσκονται στο URL <https://github.com/p13gian1/WebGIS> που αντιστοιχεί στο repository Github της εφαρμογής webGIS. Ο κατάλογος WebGIS περιλαμβάνει τον κατάλογο webserver ο οποίος με τη σειρά του περιλαμβάνει τους καταλόγους src και public. Στον κατάλογο public περιλαμβάνονται όλα τα αρχεία του front end μέρους της εφαρμογής, html, css, js καθώς και οι απαραίτητες βιβλιοθήκες jquery, ol και ol-ext.

Στον κατάλογο src υπάρχει το αρχείο app.js που αντιστοιχεί στο διακομιστή node.js. Επίσης στον ίδιο κατάλογο περιλαμβάνονται οι εικόνες SID και STAR όλων των ελληνικών αεροδρομίων.

Δ'.9 Δεδομένα

Όπως αναφέραμε και παραπάνω, τα δεδομένα που χρησιμοποιούμε στην εφαρμογή μας, ανακτήθηκαν είτε από το εγχειρίδιο A.I.P Greece October 2020, είτε δημιουργήθηκαν έπειτα από επεξεργασία θεματικών αεροπορικών χαρτών.

Συγκεκριμένα από τα εγχειρίδια AIP Ελλάδας και Ιταλίας ανακτήθηκαν τα δεδομένα για τα waypoints, waypoints_vfr, navaids, aerodromes, AIP info. Οι γεωγραφικές συντεταγμένες ήταν της μορφής μοίρες, λεπτά της μοίρας και δευτερόλεπτα της μοίρας. Έτσι τα δεδομένα αυτά πριν να εισαχθούν στη βάση δεδομένων, έπρεπε να συγκεντρωθούν σε φύλλο Excel και να μετατραπούν σε δεκαδική μορφή.

Τα υπόλοιπα δεδομένα από τους πίνακες VFR_Layer, MTMA_Layer, LGC_Layer, LGR_Layer, LGM_Layer, LGD_Layer και CTR_Layer δημιουργήθηκαν από θεματικούς αεροπορικούς χάρτες καταγράφοντας από σημείο σε σημείο τις γεωγραφικές συντεταγμένες των ορίων των περιοχών αυτών, διαδικασία η οποία αποδείχθηκε επίπονη και χρονοβόρα.

Τέλος η διαδικασία που ακολουθήθηκε για την δημιουργία του πίνακα airways_Layer περιγράφηκε στην παράγραφο Δ'.4.4.

Κεφάλαιο Ε΄

Συμπεράσματα – Μελλοντικές

Επεκτάσεις

ΤΕΛΕΙΩΝΟΝΤΑΣ την παρουσίαση της εφαρμογής webGIS, θα αναφερθούμε στα αποτελέσματα της υλοποίησής της καθώς και στις μελλοντικές επεκτάσεις που θα μπορούσαν να γίνουν.

Η εφαρμογή webGIS, αποτέλεσε μια προσπάθεια να οπτικοποιηθεί η κυκλοφορία των αεροσκαφών σε μη ελεγχόμενα αεροδρόμια και να υποστηριχθεί το έργο των υπαλλήλων της Υπηρεσίας Πολιτικής Αεροπορίας που παρέχουν έργο A.F.I.S. Ο σχεδιασμός και η υλοποίηση της εφαρμογής στηρίχθηκαν πάνω στις επιλογές που μας προσφέρουν οι βιβλιοθήκες openlayers και ol-ext. Ενδεχομένως αν χρησιμοποιούσαμε κάποια άλλη βιβλιοθήκη π.χ leaflet να μπορούσαμε να πετύχουμε καλύτερα αποτελέσματα. Ακόμα όμως και σε αυτά τα πλαίσια στα οποία κινηθήκαμε η εφαρμογή webGIS έχει ικανοποιητικές επιδόσεις για τον σκοπό τον οποίο σχεδιάστηκε.

Κατά τη διάρκεια της υλοποίησης της εφαρμογής αντιμετωπίσαμε διάφορα προβλήματα, όπως η εισαγωγή του διανυσματικού βασικού χάρτη από τον διακομιστή maptiler στο αντικείμενο map της βιβλιοθήκης openlayers, ώστε να μπορεί να γίνει manipulation όπως τα υπόλοιπα layers, η σωστή απεικόνιση των ονομάτων των αεροδιαδρόμων πάνω στο βασικό χάρτη ώστε να εμφανίζονται παράλληλα με αυτούς, η όσο το δυνατόν σωστή απεικόνιση της κίνησης του αεροσκάφους όπως στις πραγματικές εφαρμογές radar σε σχέση με τις δυνατότητες που προσφέρουν οι βιβλιοθήκες openlayers και ol-ext και τέλος η αναγκαία μετατροπή των

γεωδαιτικών δεδομένων σε δεκαδική μορφή από την προηγούμενη πρωτότυπη μορφή τους όπως δέκατα και εκατοστά της μοίρας ή λεπτά και δευτερόλεπτα της μοίρας.

Υπάρχουν και άλλες δυνατότητες οι οποίες δεν πρόλαβαν να υλοποιηθούν λόγω χρόνου, πολλές από αυτές είναι πιο εύκολες στην υλοποίηση και άλλες πιο δύσκολες. Η εφαρμογή μας θα μπορούσε να επικοινωνεί με το σταθερό δίκτυο αεροναυτικών επικοινωνιών AFTN και να λαμβάνει από εκεί κατευθείαν τα σχέδια πτήσης που θα υποβληθούν στο σύστημα, χωρίς να χρειαστεί να τα πληκτρολογήσει ο χρήστης. Η εφαρμογή μας θα μπορούσε επιπλέον να απεικονίζει προειδοποίηση ότι ένα σχέδιο VFR είναι εκτός ορίων VFR πτήσης, δηλαδή ο ήλιος πρόκειται να δύσει πριν την άφιξη του αεροσκάφους στον προορισμό του και ως εκ τούτου δεν πρέπει να αναχωρήσει. Θα μπορούσε ακόμα να εμφανίζει τον προτεινόμενο διάδρομο σε σχέση με τον παρόντα άνεμο αεροδρομίου, δηλαδή ποιο άκρο του διαδρόμου είναι το προτεινόμενο για να προσγειωθεί το αεροσκάφος, ώστε ο άνεμος να έχει αντίθετη φορά από την κίνηση του αεροσκάφους. Τέλος η εφαρμογή webGIS θα μπορούσε να εμφανίζει προειδοποιήσεις στον χάρτη για επικίνδυνο καιρό με βάση METARs ή TAFs που εκδίδει η Εθνική Μετεωρολογική Υπηρεσία (Ε.Μ.Υ) στο ελληνικό F.I.R.

Βιβλιογραφία - Δικτυογραφία

- [1] Geographic coordinate system, https://en.wikipedia.org/wiki/Geographic_coordinate_system
- [2] Reference ellipsoid, https://en.wikipedia.org/wiki/Reference_ellipsoid
- [3] World Geodetic System, https://en.wikipedia.org/wiki/World_Geodetic_System
- [4] EPSG Geodetic Parameter Dataset,
https://en.wikipedia.org/wiki/EPST_Geodetic_Parameter_Dataset
- [5] EPSG 31254 Austria GK West, <https://epsg.io/31254>
- [6] Coordinated Universal Time, https://en.wikipedia.org/wiki/Coordinated_Universal_Time
- [7] International Civil Aviation Organization,
https://en.wikipedia.org/wiki/International_Civil_Aviation_Organization
- [8] Aeronautical Information Publication,
https://en.wikipedia.org/wiki/Aeronautical_Information_Publication
- [9] Atmospheric pressure, https://en.wikipedia.org/wiki/Atmospheric_pressure
- [10] Pitot tube, https://en.wikipedia.org/wiki/Pitot_tube
- [11] Pitot-static system, https://en.wikipedia.org/wiki/Pitot-static_system
- [12] Flight information region, https://en.wikipedia.org/wiki/Flight_information_region
- [13] ICAO: Aerodrome Flight Information Service (AFIS),
<http://aviationknowledge.wikidot.com/aviation:aerodrome-flight-information-service-afis>
- [14] Flight information service, https://en.wikipedia.org/wiki/Flight_information_service

- [15] Airspace Volumes & Sectorisation Module 13 – Activity 9,
https://www.icao.int/MID/Documents/2014/PBN%20Workshop-Tunis/13%20%20EUR%20PBN%20Airspace%20Workshop_Designing%20Volumes%20+%20Sector-s-vJUL2013%20.pdf
- [16] Control Zone, https://en.wikipedia.org/wiki/Control_zone
- [17] Terminal Control Area - Terminal Manoeuvring Area,
https://en.wikipedia.org/wiki/Terminal_control_area
- [18] Aerodrome Traffic Zone,
[https://www.skybrary.aero/index.php/Aerodrome_Traffic_Zone_\(ATZ\)](https://www.skybrary.aero/index.php/Aerodrome_Traffic_Zone_(ATZ))
- [19] Primary Radar, https://en.wikipedia.org/wiki/Primary_radar
- [20] Secondary Surveillance Radar, https://en.wikipedia.org/wiki/Secondary_surveillance_radar
- [21] Transponder, [https://en.wikipedia.org/wiki/Transponder_\(aeronautics\)](https://en.wikipedia.org/wiki/Transponder_(aeronautics))
- [22] List of transponder codes, https://en.wikipedia.org/wiki/List_of_transponder_codes
- [23] Flight Management System, https://en.wikipedia.org/wiki/Flight_management_system
- [24] Aeronautical Fixed Telecommunication Network,
https://en.wikipedia.org/wiki/Aeronautical_Fixed_Telecommunication_Network
- [25] Notice to Airmen NO.T.AM, <https://el.wikipedia.org/wiki/NOTAM>
- [26] METAR, <https://en.wikipedia.org/wiki/METAR>
- [27] PostgreSQL, <https://en.wikipedia.org/wiki/PostgreSQL>
- [28] PostGIS, <https://en.wikipedia.org/wiki/PostGIS>
- [29] Mapserver, <https://en.wikipedia.org/wiki/MapServer>
- [30] GeoServer, <https://en.wikipedia.org/wiki/GeoServer>
- [31] Web Map Service, https://en.wikipedia.org/wiki/Web_Map_Service
- [32] Node.js, <https://en.wikipedia.org/wiki/Node.js>
- [33] Express.js, <https://en.wikipedia.org/wiki/Express.js>
- [34] express, <https://www.npmjs.com/package/express>

- [35] body-parser, <https://www.npmjs.com/package/body-parser>
- [36] @airport-discovery/notams, <https://www.npmjs.com/package/@airport-discovery/notams>
- [37] @airport-discovery/metars-tafs, <https://www.npmjs.com/package/@airport-discovery/metars-tafs>
- [38] Pg - node-postgres, <https://node-postgres.com/>
- [39] image-to-base64, <https://www.npmjs.com/package/image-to-base64>
- [40] Openlayers, <https://openlayers.org>
- [41] OpenLayers, <https://en.wikipedia.org/wiki/OpenLayers>
- [42] ol-ext, <https://github.com/Viglino/ol-extboot>
- [43] GIS, https://en.wikipedia.org/wiki/Geographic_information_system
- [44] Raster graphics, https://en.wikipedia.org/wiki/Raster_graphics
- [45] Vector graphics, https://en.wikipedia.org/wiki/Vector_graphics
- [46] Mapbox, <https://en.wikipedia.org/wiki/Mapbox>
- [47] MapTiler, <https://wiki.openstreetmap.org/wiki/MapTiler>
- [48] maptiler, <https://www.maptiler.com>
- [49] ol.Map, https://openlayers.org/en/v6.0.0/apidoc/module-ol_Map-Map.html
- [50] ol.layer.Image, https://openlayers.org/en/v6.0.0/apidoc/module-ol_layer_Image-ImageLayer.html
- [51] ol.source.ImageWMS, https://openlayers.org/en/v6.0.0/apidoc/module-ol_source_ImageWMS-ImageWMS.html
- [52] ol.control.Button, <https://viglino.github.io/ol-ext/doc/doc-pages/ol.control.Button>
- [53] ol.control.Bar, <https://viglino.github.io/ol-ext/doc/doc-pages/ol.control.Bar.html>
- [54] Flight Plan, https://en.wikipedia.org/wiki/Flight_plan
- [55] Vector Layer, https://openlayers.org/en/v6.0.0/apidoc/module-ol_layer_Vector-VectorLayer.html
- [56] ol.style.Style, https://openlayers.org/en/v6.0.0/apidoc/module-ol_style_Style.html

- [57] ol.style.Text, https://openlayers.org/en/v6.0.0/apidoc/module-ol_style_Text.html
- [58] GPX – GPS exchange format, <https://wiki.openstreetmap.org/wiki/GPX>
- [59] ol.source.Vector, https://openlayers.org/en/v6.0.0/apidoc/module-ol_source_Vector.html
- [60] ol.featureAnimation, <https://viglino.github.io/ol-ext/doc/doc-pages/ol.featureAnimation.html>

Συντμήσεις

ACID	Atomicity Consistency Isolation Durability
AFIS	Aerodrome Information Service
AFTN	Aeronautical Fixed Telecommunication Network
AGL	Above Ground Level
AIP	Aeronautical Information Publication
API	Application Programming Interface
ATZ	Aerodrome Traffic Zone
CTR	Controlled Traffic Region
DFD	Data Flow Diagram
DME	Distance Measuring Equipment
EPSG	Geodetic Parameter Dataset
FDP	Flight Data Processing
FIR	Flight Information Region
FL	Flight Level
FPL	Flight Plan
GIS	Geographic Information Systems
GMT	Greenwich Mean Time
GPS	Global Positioning System
ICAO	International Civil Aviation Organization
IFR	Instrument Flight Rules
METAR	Meteorological Terminal Aviation Routine Weather Report

MSL	Mean Sea Level
MTMA	Military Terminal Manoeuvring Area
NDB	Non Directional Beacon
NM	Nautical Mile
NOTAM	Notice To Airmen
QNH	Regional Or Airfield Pressure Setting
PSR	Primary Surveillance Radar
SID	Standard Instrument Departure
SLD	Style Layer Descriptor
SSR	Secondary Surveillance Radar
STAR	Standard Arrival
TAF	Terminal Aerodrome Forecast
TMA	Terminal Manoeuvring Area
VFR	Visual Flight Rules
VHF	Very High Frequency
VOR	Very High Frequency Omni Directional Radio Range
UHF	Ultra High Frequency
UTC	Coordinated Universal Time
WCS	Web Coverage Standard
WFS	Web Feature Service
WGS	World Geodetic System
WMS	Web Map Service

Γλωσσάρι Ξενικών Όρων

Access	Πρόσβαση
Aeronautical	Αεροναυτικός
Aircraft	Αεροσκάφος
Altitude	Υψόμετρο
Application	Εφαρμογή
Apron	Χώρος Στάθμευσης Αεροσκαφών
Array	Πίνακας
Arrival	Άφιξη
Beacon	Φάρος
Broken	Σπασμένος
Credentials	Διαπιστευτήρια
Data	Δεδομένα
Database	Βάση Δεδομένων
Departure	Αναχώρηση
Directional	Κατευθυντικός
Distance	Απόσταση
Feature	Λειτουργία
Few	Λίγος
Fixed	Σταθερός
Flight	Πτήση
Flight Level	Επίπεδο Πτήσης

Flight Plan	Σχέδιο Πτήσης	Geoid	Γεωειδές
Instrument	Όργανο Πλοήγησης		
Interactive	Διαδραστικός		
Label	Ετικέτα		
Latitude	Γεωγραφικό Πλάτος		
Layer	Επίπεδο		
LineString	Ευθεία		
Longitude	Γεωγραφικό Μήκος		
Management	Διαχείριση		
Map	Χάρτης		
Multithreading	Πολυνηματικότητα		
Network	Δίκτυο		
Opacity	Διαφάνεια		
Overcast	Νεφελώδης, εδώ έχει την σημασία της πλήρους κάλυψης του ουρανού από σύννεφα		
Permanent	Μόνιμος		
Point	Σημείο		
Polygon	Πολύγωνο		
Query	Ερώτημα		
Raster	Ψηφιδωτός		
Reference	Ελλειψοειδές Αναφοράς		
Report	Αναφορά		
Response	Απάντηση		
Rule	Κανόνας		
Runway	Διάδρομος		
Satellite Navigation	Δορυφορική Πλοήγηση		
Scattered	Διασκορπισμένος		
Server	Διακομιστής		

Status	Κατάσταση
Surveillance	Παρακολούθηση
Taxiway	Συνδετήριος Διάδρομος
Telecommunication	Τηλεπικοινωνίες
Terminal	Τερματικός
Traffic	Κυκλοφορία
Vector	Διανυσματικός
Visibility	Ορατότητα
Visual	Εξ Όψεως

Ευρετήριο

αεροδιάδρομοι, 9

βασικός, 70

χάρτης, 70

βάση, 21

δεδομένων, 21

γεωγραφικό, 4

μήκος, 4

πλάτος, 4

γεωδαιτικό σύστημα, 4

εγκατάσταση, 107

λογισμικό, 107

λειτουργία,

απεικόνιση, 103

πτήσης, 103

προεπισκόπηση, 95

ραδιοβοήθημα, 7

σχέδιο, 15

πτήσης, 15

συντεταγμένες, 4

φόρμα,

επιλογής, 86

AIP Info, 88

METAR, 89

NOTAM, 90

IMAGE, 91

FPL, 93

Load FPL, 99

Save FPL, 100

Electronic Strip Bay, 101

AIP, 8

airways, 9

Coordinated Universal Time, 5

EPSG, 5

FIR, 10

Geoserver, 23

GIS, 31

ICAO, 6

indicator, 6

latitude, 4

location, 6

longitude, 4

METAR, 18

node.js, 27

postGIS, 21

postgreSQL, 21

SLD, 25

Squawk, 14

waypoints, 8

World Geodetic System, 4