



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

**Ανάπτυξη Εφαρμογής για Οργάνωση Ταξιδιών και Παροχή
Ταξιδιωτικών Προτάσεων**

Διπλωματική Εργασία

Μπάνου Φανή

Επιβλέπων: Βασιλακόπουλος Μιχαήλ

Σεπτέμβριος 2025



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

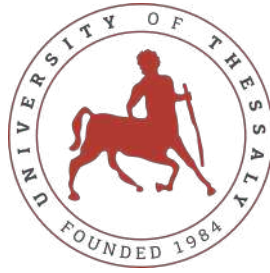
**Ανάπτυξη Εφαρμογής για Οργάνωση Ταξιδιών και Παροχή
Ταξιδιωτικών Προτάσεων**

Διπλωματική Εργασία

Μπάνου Φανή

Επιβλέπων: Βασιλακόπουλος Μιχαήλ

Σεπτέμβριος 2025



UNIVERSITY OF THESSALY
SCHOOL OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

**Development of an Application for Travel Planning and
Provisioning of Travel Suggestions**

Diploma Thesis

Banou Fani

Supervisor: Vasilakopoulos Michael

September 2025

Εγκρίνεται από την Επιτροπή Εξέτασης:

Επιβλέπων **Βασιλακόπουλος Μιχαήλ**

Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Μέλος **Θάνος Γεώργιος**

Μέλος Ε.ΔΙ.Π., Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Μέλος **Φεύγας Αθανάσιος**

Μέλος Ε.ΔΙ.Π., Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Ευχαριστίες

Πρώτα απ' όλα, ένα μεγάλο ευχαριστώ στον επιβλέποντα καθηγητή μου, κ. Βασιλακόπουλο Μιχαήλ, για την καθοδήγηση, τη βοήθεια και τις συμβουλές του καθ' όλη τη διάρκεια της εργασίας. Επίσης, θα ήθελα ομοίως να ευχαριστήσω τους καθηγητές Θάνο Γεώργιο και Φεύγα Αθανάσιο για τη συμμετοχή τους στην επιτροπή εξέτασης.

Ευχαριστώ επίσης την οικογένειά μου για την υποστήριξή της όλα αυτά τα χρόνια, την κατανόηση και την ενθάρρυνση σε κάθε μου βήμα. Τέλος, θέλω να ευχαριστήσω τους φίλους μου για την παρέα, τις ιδέες, και την υπομονή τους σε όλη αυτή τη διαδρομή.

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

«Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Δηλώνω επίσης ότι τα αποτελέσματα της εργασίας δεν έχουν χρησιμοποιηθεί για την απόκτηση άλλου πτυχίου. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής».

Η Δηλούσα

Μπάνου Φανή

Διπλωματική Εργασία

Ανάπτυξη Εφαρμογής για Οργάνωση Ταξιδιών και Παροχή Ταξιδιωτικών Προτάσεων

Μπάνου Φανή

Περίληψη

Η παρούσα διπλωματική εργασία ασχολείται με την ανάπτυξη μιας εφαρμογής κινητού που έχει στόχο να κάνει πιο εύκολο τον σχεδιασμό και την οργάνωση ταξιδιών από τους ίδιους τους χρήστες. Η ιδέα ήταν να δημιουργηθεί ένα εργαλείο που να λειτουργεί σαν προσωπικός ταξιδιωτικός οδηγός, δίνοντας τη δυνατότητα να ανακαλύπτει κανείς προορισμούς, να βρίσκει ξενοδοχεία, εστιατόρια και αξιοθέατα, αλλά και να φτιάχνει το δικό του πρόγραμμα ταξιδιού σύμφωνα με τις ανάγκες του.

Για την ανάπτυξη επιλέχθηκε το React Native, ώστε η εφαρμογή να μπορεί να τρέχει τόσο σε Android όσο και σε iOS συσκευές με κοινό κώδικα. Για τα γεωγραφικά δεδομένα και τις προτάσεις αξιοποιήθηκαν τα Google APIs (Places, Directions, Maps), τα οποία επιτρέπουν στον χρήστη να βλέπει κοντινά σημεία ενδιαφέροντος, να υπολογίζει διαδρομές και να έχει μια πιο διαδραστική εμπειρία με τον χάρτη. Επιπλέον, ενσωματώθηκαν εργαλεία τεχνητής νοημοσύνης, ώστε να προσφέρονται έξυπνες προτάσεις και αυτόματη δημιουργία ταξιδιωτικών προγραμμάτων. Στο backend χρησιμοποιήθηκαν Node.js, Express.js και MongoDB, για την αποθήκευση και διαχείριση των δεδομένων αλλά και για την επικοινωνία με το frontend.

Το τελικό αποτέλεσμα είναι μια φιλική προς τον χρήστη και λειτουργική εφαρμογή που προσφέρει όλα τα απαραίτητα εργαλεία για την οργάνωση ενός ταξιδιού, αξιοποιώντας έξυπνες τεχνολογίες και δεδομένα σε πραγματικό χρόνο.

Λέξεις-κλειδιά:

Ανάπτυξη εφαρμογής, Οργάνωση ταξιδιού, React Native, Express, Node, MongoDB

Diploma Thesis

Development of an Application for Travel Planning and Provisioning of Travel Suggestions

Banou Fani

Abstract

This thesis focuses on the development of a mobile application designed to simplify the process of travel planning and organization by the users themselves. The idea was to create a tool that functions as a personal travel guide, allowing users to discover destinations, find hotels, restaurants, and attractions, and create their own travel itinerary according to their needs.

The application was developed using React Native to ensure compatibility with both Android and iOS devices. For the collection of geographic data and recommendations, Google APIs (Google Places, Directions, Maps) were integrated, enabling the discovery of nearby points of interest, route calculation, and location visualization. Additionally, artificial intelligence tools were employed to generate smart recommendations and automate the creation of travel plans. The backend was implemented using Node.js, Express.js, and MongoDB to manage data and handle communication with the frontend.

The final outcome is a user-friendly and functional application that provides all the necessary tools for organizing a trip, utilizing smart technologies and real-time data.

Keywords:

Development of mobile application, Trip Planning, React Native, Express, Node, MongoDB

Πίνακας περιεχομένων

Ευχαριστίες	ix
Περίληψη	xii
Abstract	xiii
Πίνακας περιεχομένων	xv
Κατάλογος σχημάτων	xvii
Κατάλογος πινάκων	xxi
Συντομογραφίες	xxiii
1 Εισαγωγή	1
1.1 Αντικείμενο της Διπλωματικής	1
1.1.1 Συνεισφορά	2
1.2 Οργάνωση του τόμου	3
2 Σχετικές Εφαρμογές & Συστήματα	5
2.1 Ανάλυση Ταξιδιωτικών Εφαρμογών	5
2.2 Συστήματα και Κριτήρια Επιλογής	8
3 Τεχνολογίες & Εργαλεία	11
3.1 Front-end	11
3.2 Back-end	13
3.3 Βάση Δεδομένων	13
3.4 APIs και Διασύνδεση	14

3.5	Ασφάλεια και Πιστοποίηση	17
3.6	Λειτουργίες Τεχνητής Νοημοσύνης	18
4	Αρχιτεκτονική & Υλοποίηση	21
4.1	Βάση Δεδομένων	21
4.2	Backend	24
4.3	Frontend	34
5	Λειτουργίες & Χαρακτηριστικά	43
5.1	Σύνδεση (Sign In)	43
5.2	Εγγραφή (Sign Up)	46
5.3	Επαναφορά κωδικού (Reset Password)	48
5.4	Προφίλ Χρήστη (Profile)	50
5.5	Προβολή ταξιδιών (My Trips)	52
5.6	Πλάνο ταξιδιού (Trip Details)	56
5.7	Δημιουργία ταξιδιού (Create Trip)	57
5.8	Εξατομικευμένο ταξίδι (Custom Trip)	61
5.9	Σελίδα ανακάλυψης (Discover)	66
5.10	Αγαπημένα (Favorites)	70
6	Συμπεράσματα	73
6.1	Σύνοψη και συμπεράσματα	73
6.2	Μελλοντικές επεκτάσεις	74
	Βιβλιογραφία	77
	ΠΑΡΑΡΤΗΜΑΤΑ	81
A	Εγκατάσταση Εφαρμογής	83
A.1	Πηγαίος Κώδικας	83
A.2	Οδηγίες Εγκατάστασης	83
B	Έλεγχος και Δοκιμή σε Συσκευές	85

Κατάλογος σχημάτων

4.1	Σύνδεση του backend με τη MongoDB μέσω Mongoose	22
4.2	Δημιουργία φακέλου και αρχικοποίηση	24
4.3	Εγκατάσταση απαραίτητων βιβλιοθηκών μέσω npm	25
4.4	Ρύθμιση scripts για ανάπτυξη και παραγωγή στο package.json	25
4.5	Αρχείο .env με μεταβλητές περιβάλλοντος	25
4.6	Εκτέλεση του server σε ανάπτυξη και παραγωγή	26
4.7	Δομή αρχείων και φακέλων του backend	28
4.8	Ενεργοποίηση βασικών middleware στο server.js	28
4.9	Καταχώρηση των διαδρομών (routes)	29
4.10	Εκκίνηση του Express server	29
4.11	Χρήση του middleware για προστατευμένο endpoint	30
4.12	Παράδειγμα ορισμού μοντέλου User σε Mongoose	30
4.13	Παράδειγμα route για δημιουργία ταξιδιού	31
4.14	Λειτουργία του endpoint για σχεδιασμό ταξιδιού με AI	34
4.15	Δομή αρχείων και φακέλων του frontend	37
4.16	Δομή αρχείων και φακέλων του φακέλου app	37
4.17	Δομή αρχείων και φακέλων του φακέλου components	37
4.18	Παράδειγμα κώδικα για την ενημέρωση προφίλ χρήστη	41
4.19	Παράδειγμα κώδικα για την ανάκτηση πληροφοριών τοποθεσίας	42
5.1	Κεντρική οθόνη σύνδεσης (Sign In)	44
5.2	Παραδείγματα ελέγχων κατά τη σύνδεση (Sign In)	44
5.3	Παραδείγματα ενεργειών και μηνυμάτων κατά τη σύνδεση (Sign In)	45
5.4	Κεντρική οθόνη δημιουργίας λογαριασμού (Sign Up)	46
5.5	Παραδείγματα ελέγχων και μηνυμάτων κατά την εγγραφή (Sign Up)	47

5.6	Κεντρική οθόνη επαναφοράς κωδικού (Reset Password)	48
5.7	Παραδείγματα ελέγχων και μηνυμάτων στην επαναφορά κωδικού (Reset Password)	49
5.8	Κεντρική οθόνη προφίλ χρήστη (Profile)	50
5.9	Παραδείγματα ενεργειών και μηνυμάτων από το προφίλ χρήστη (Profile)	51
5.10	Κεντρική οθόνη ταξιδιών <i>My Trips</i>	52
5.11	Οθόνη προβολής ταξιδιών <i>My Trips</i>	53
5.12	Προσθήκης μέλους και μηνύματα κατάστασης	54
5.13	Παραδείγματα ελέγχων και μηνυμάτων	55
5.14	Σελίδα <i>Trip Details</i> που δημιουργείται από AI	56
5.15	Επιλογή προορισμού για την δημιουργία ταξιδιού	58
5.16	Βημάτα επιλογής στοιχείων ταξιδιού	59
5.17	Ενδεικτικά μηνύματα σφάλματος κατά την επιλογή στοιχείων ταξιδιού	60
5.18	Προβολή ταξιδιού με χρονοδιάγραμμα δραστηριοτήτων	62
5.19	Προβολή ταξιδιού χωρίς δραστηριότητες	62
5.20	Εναλλακτική προβολή δραστηριοτήτων σε ημερολόγιο	62
5.21	Επιβεβαίωση αποθήκευσης ταξιδιού	62
5.22	Παράδειγμα προσθήκης δραστηριότητας σε συγκεκριμένη ημέρα	63
5.23	Παράδειγμα επεξεργασίας υπάρχουσας δραστηριότητας	63
5.24	Παράδειγμα προβολής δραστηριοτήτων ανά ημέρα	64
5.25	Προβολή τοποθεσιών σε χάρτη	65
5.26	Φίλτρα αναζήτησης δραστηριοτήτων	65
5.27	Προβολή δραστηριότητας με δυνατότητα προσθήκης στο πλάνο	65
5.28	Προβολή τοποθεσιών σε χάρτη	67
5.29	Μενού ενεργειών (Search / Location / Filters)	67
5.30	Αναζήτηση τοποθεσίας	67
5.31	Φίλτρα τοποθεσιών	68
5.32	Κάρτα τοποθεσίας με λεπτομέρειες	68
5.33	Πλοήγηση με επιλογή Driving	68
5.34	Αναλυτικές οδηγίες διαδρομής	68
5.35	Πλοήγηση με επιλογή Walking	69
5.36	Αναλυτικές οδηγίες διαδρομής	69

5.37	Πλοήγηση με επιλογή Transit	69
5.38	Αναλυτικές οδηγίες διαδρομής	69
5.39	Παραδείγματα εμφάνισης αγαπημένων (Favorites)	70
5.40	Μηνύματα επιβεβαίωσης κατά τη διαγραφή αγαπημένων	71

Κατάλογος πινάκων

2.1	Συγκριτική ανάλυση ταξιδιωτικών εφαρμογών	7
3.1	Κύριες μέθοδοι HTTP και περιγραφή τους	15
3.2	Συχνότεροι HTTP Status Codes	15
4.1	Endpoints του auth.js	32
4.2	Endpoints του favorites.js	32
4.3	Endpoints του trips.js	33
4.4	Endpoint του generate-ai.js	33

Συντομογραφίες

βλπ	βλέπε
κ.λπ.	και λοιπά
κ.ο.κ	και ούτω καθεξής
cli	command line interface
npm	Node Package Manager
npv	Node Package eXecute
API	Application Programming Interface
REST	Representational State Transfer
CORS	Cross-Origin Resource Sharing
HTTP	Hypertext Transfer Protocol Secure
JSON	Javascript Object Notation
JWT	JSON Web Tokens
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
JSX	JavaScript XML
UI	User Interface
AI	Artificial Intelligence
DB	Database
SQL	Structured Query Language
NoSQL	Not only SQL

Κεφάλαιο 1

Εισαγωγή

Τα τελευταία χρόνια, ο τομέας του τουρισμού έχει εξελιχθεί σημαντικά, τόσο ως προς τον αριθμό των ταξιδιωτών όσο και ως προς την ποικιλία των εμπειριών που προσφέρονται. Η πρόοδος της τεχνολογίας και η ψηφιοποίηση πολλών υπηρεσιών έχουν επιτρέψει στους ταξιδιώτες να οργανώνουν μόνοι τους τα ταξίδια τους, με μεγαλύτερη ευκολία και προσαρμογή στις προσωπικές τους ανάγκες.

Ο σύγχρονος ταξιδιώτης δεν βασίζεται πλέον σε παραδοσιακά μέσα, όπως τα τουριστικά πρακτορεία ή οι ταξιδιωτικοί οδηγοί. Αντίθετα, αναζητά άμεση πληροφόρηση, εύκολη πρόσβαση σε κρατήσεις και εξατομικευμένες εμπειρίες με βάση τα ενδιαφέροντά του, το διαθέσιμο χρόνο και τον προϋπολογισμό του. Αν και οι εφαρμογές για κινητές συσκευές καλύπτουν αρκετές επιμέρους ανάγκες, συχνά δεν προσφέρουν μια ολοκληρωμένη λύση. Συχνά απουσιάζει η ουσιαστική προσαρμογή στις ανάγκες του χρήστη, ενώ τα δεδομένα που χρησιμοποιούνται δεν είναι πάντα έγκαιρα ή πλήρη. Επιπλέον, η εμπειρία χρήστη επηρεάζεται αρνητικά από περίπλοκες διεπαφές και την αδυναμία πολλών εφαρμογών να διαχειριστούν αποτελεσματικά πληροφορίες από διαφορετικές πηγές.

Παραμένει, λοιπόν, η ανάγκη για μια πιο ολοκληρωμένη λύση που να συνδυάζει λειτουργικότητα, ευχρηστία και έξυπνες προτάσεις σε ένα ενιαίο και αποδοτικό περιβάλλον χρήσης.

1.1 Αντικείμενο της Διπλωματικής

Αντικείμενο της παρούσας διπλωματικής εργασίας είναι η σχεδίαση και ανάπτυξη μιας σύγχρονης, διαδραστικής εφαρμογής για κινητές συσκευές, που λειτουργεί ως ολοκληρωμένος ψηφιακός ταξιδιωτικός οδηγός. Στόχος είναι να προσφέρει στους χρήστες τα απαραίτητα

εργαλεία για τον σχεδιασμό και την οργάνωση των ταξιδιών τους με ευκολία, ευελιξία και εξατομίκευση.

Η εργασία επικεντρώνεται στην επίλυση τριών βασικών προβλημάτων που εντοπίζονται σε υπάρχουσες εφαρμογές:

- **Διάσπαση λειτουργιών:** Οι ταξιδιώτες συχνά χρειάζονται πολλαπλές εφαρμογές για διαφορετικές ανάγκες (κρατήσεις, χάρτες, πλοήγηση, αξιοθέατα). Η παρούσα εφαρμογή συγκεντρώνει αυτές τις βασικές λειτουργίες σε μια ενιαία πλατφόρμα.
- **Έλλειψη εξατομίκευσης:** Πολλές εφαρμογές προσφέρουν γενικές προτάσεις, χωρίς να λαμβάνουν υπόψη τις πραγματικές ανάγκες του χρήστη. Η συγκεκριμένη εφαρμογή χρησιμοποιεί γεωγραφικά δεδομένα, φίλτρα προτιμήσεων και έξυπνους μηχανισμούς προτάσεων για πιο προσωποποιημένο περιεχόμενο.
- **Περιορισμένη εμπειρία χρήστη:** Συχνά παρατηρείται κακός σχεδιασμός διεπαφής, που δυσκολεύει τη χρήση. Η εφαρμογή αναπτύσσεται με στόχο ένα φιλικό, και απλό περιβάλλον διεπαφής, κατάλληλο για γρήγορη και ευχάριστη αλληλεπίδραση.

Συνολικά, η εργασία στοχεύει να καλύψει το κενό ανάμεσα στις δυνατότητες της τεχνολογίας και τις πραγματικές ανάγκες του ταξιδιώτη, προσφέροντας ένα εύχρηστο, ολοκληρωμένο εργαλείο που διευκολύνει τον προγραμματισμό ενός ταξιδιού.

1.1.1 Συνεισφορά

Η συνεισφορά της παρούσας διπλωματικής εργασίας συνοψίζεται ως εξής:

1. Πραγματοποιήθηκε μελέτη υπαρχόντων ταξιδιωτικών εφαρμογών, εντοπίζοντας βασικές ελλείψεις ως προς τη λειτουργικότητα, την εξατομίκευση και την εμπειρία χρήστη. Έτσι, καθορίστηκαν οι κύριες λειτουργίες που θα ενσωματωθούν στην εφαρμογή.
2. Εξετάστηκαν διάφορες τεχνολογίες ανάπτυξης και επιλέχθηκαν οι κατάλληλες (React Native, Node.js, Express.js και MongoDB), ώστε να εξασφαλιστεί αποδοτική λειτουργία, εύκολη διασύνδεση μεταξύ των επιμέρους συστημάτων και δυνατότητα μελλοντικής επέκτασης.

3. Αναπτύχθηκε το backend σύστημα για τη διαχείριση των προφίλ χρηστών, των ταξιδιωτικών προγραμμάτων και των αγαπημένων προορισμών, με χρήση των τεχνολογιών JWT για την αυθεντικοποίηση και bcrypt για την ασφαλή κρυπτογράφηση των ευαίσθητων δεδομένων.
4. Ενσωματώθηκαν στοιχεία τεχνητής νοημοσύνης, με αξιοποίηση του μοντέλου Gemini, για τη δυναμική δημιουργία ταξιδιωτικού προγράμματος, καθώς και για την αυτόματη αναζήτηση ξενοδοχείων και πτήσεων, βάσει των επιλογών και των προτιμήσεων του χρήστη.
5. Υλοποιήθηκε το frontend της εφαρμογής με χρήση της πλατφόρμας React Native, δημιουργώντας ένα δυναμικό και φιλικό προς τον χρήστη περιβάλλον διεπαφής για κινητές συσκευές, πλήρως συμβατό με Android και iOS.
6. Πραγματοποιήθηκε διασύνδεση με τα APIs της Google (Maps, Places, Directions) για την παροχή γεωγραφικών δεδομένων, προτεινόμενων τοποθεσιών και πληροφοριών πλοήγησης σε πραγματικό χρόνο.
7. Αξιολογήθηκε η λειτουργικότητα της εφαρμογής σε διαφορετικά στάδια ανάπτυξης, με στόχο τον εντοπισμό αδυναμιών και τη βελτίωση της συνολικής απόδοσης, μέσω της προσθήκης νέων λειτουργιών και της επίλυσης τεχνικών προβλημάτων.

1.2 Οργάνωση του τόμου

Ο τόμος της διπλωματικής εργασίας είναι οργανωμένος σε έξι κεφάλαια. Το Κεφάλαιο 2 εξετάζει τις σχετικές εφαρμογές και πλατφόρμες οργάνωσης ταξιδιών, ενώ παράλληλα παρουσιάζει τα κριτήρια βάσει των οποίων επιλέχθηκαν τα συστήματα της εφαρμογής. Το Κεφάλαιο 3 περιγράφει τις τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν κατά την ανάπτυξη της εφαρμογής, όπως React Native, Google APIs, Node.js και MongoDB. Στο Κεφάλαιο 4 αναπτύσσεται η αρχιτεκτονική του συστήματος και παρουσιάζονται λεπτομερώς τα βασικά υποσυστήματα της εφαρμογής, από το frontend και τη διασύνδεση με εξωτερικά APIs έως το backend και τη διαχείριση δεδομένων. Το Κεφάλαιο 5 επικεντρώνεται στη περιγραφή των λειτουργιών και την αναλυτική παρουσίαση της εφαρμογής. Τέλος, στο Κεφάλαιο 6 συνοψίζονται τα βασικά συμπεράσματα της εργασίας και διατυπώνονται προτάσεις για μελλοντικές βελτιώσεις και επεκτάσεις.

Κεφάλαιο 2

Σχετικές Εφαρμογές & Συστήματα

Το κεφάλαιο αυτό εξετάζει δημοφιλείς ταξιδιωτικές εφαρμογές και τις βασικές δυνατότητες που προσφέρουν. Μέσα από τη σύγκριση των πλεονεκτημάτων και των περιορισμών τους, εντοπίζονται τα σημεία που δεν καλύπτουν πλήρως τις ανάγκες των χρηστών. Στη συνέχεια, παρουσιάζονται οι τεχνολογίες που επιλέχθηκαν για την ανάπτυξη της παρούσας εφαρμογής, εξηγώντας πώς αυτές εξυπηρετούν καλύτερα τους στόχους της εργασίας και καλύπτουν τις αδυναμίες των υπάρχοντων λύσεων.

2.1 Ανάλυση Ταξιδιωτικών Εφαρμογών

Στο πλαίσιο της μελέτης των αναγκών των χρηστών, αναλύθηκαν αρκετές υπάρχουσες ταξιδιωτικές εφαρμογές που παρέχουν υπηρεσίες όπως προγραμματισμό ταξιδιών, κρατήσεις, πλοήγηση και αναζήτηση σημείων ενδιαφέροντος. Παρακάτω παρουσιάζονται ορισμένες από τις πιο διαδεδομένες:

TripIt

Το TripIt [1] είναι μια εφαρμογή που βοηθά τον χρήστη να οργανώσει το ταξίδι του αυτόματα, συγκεντρώνοντας πληροφορίες από επιβεβαιώσεις κρατήσεων μέσω email (πτήσεις, ξενοδοχεία, ενοικιάσεις αυτοκινήτων, κ.ά.). Δημιουργεί έτσι ένα ενιαίο χρονοδιάγραμμα που είναι προσβάσιμο από οποιαδήποτε συσκευή. Αν και είναι χρήσιμο στην οργάνωση του ταξιδιού, δεν υποστηρίζει δυνατότητες δημιουργίας εξατομικευμένου προγράμματος ή προσθήκης σημείων ενδιαφέροντος με βάση προτιμήσεις. Επιπλέον, δεν διαθέτει χάρτες ή εργαλεία πλοήγησης.

TripAdvisor

Το TripAdvisor [2] αποτελεί μία από τις πιο γνωστές πλατφόρμες για αξιολογήσεις ξενοδοχείων, εστιατορίων και αξιοθέατων, βασισμένες κυρίως σε εμπειρίες άλλων χρηστών. Προσφέρει επίσης δυνατότητες σύγκρισης τιμών και κρατήσεων μέσω συνεργαζόμενων υπηρεσιών. Ωστόσο, δεν λειτουργεί ως εργαλείο σχεδιασμού ταξιδιού με πρόγραμμα ή χάρτες. Ο χρήστης πρέπει να μεταβαίνει σε άλλες εφαρμογές για τον προγραμματισμό ή την πλοήγηση.

Sygie Travel

Η εφαρμογή Sygie Travel [3] παρέχει πολλές πληροφορίες για πόλεις και τουριστικά αξιοθέατα σε όλο τον κόσμο. Επιτρέπει στον χρήστη να οργανώσει το ταξίδι του με χρονολογικό πρόγραμμα, να δει διαδρομές και εκτιμώμενους χρόνους μετακίνησης, και να πλοηγηθεί σε διαδραστικό χάρτη — ακόμη και χωρίς σύνδεση στο διαδίκτυο. Παρόλα αυτά, το περιεχόμενο και οι προτάσεις δεν είναι εξατομικευμένες και δεν προσαρμόζονται στις προτιμήσεις του χρήστη, στον διαθέσιμο χρόνο ή στον προϋπολογισμό του.

RoutePerfect

Το RoutePerfect [4] βοηθά τον χρήστη να σχεδιάσει ταξίδια στην Ευρώπη, με δυνατότητα επιλογής πόλεων, ημερών, θεματικών φίλτρων (ρομαντικό, πολιτιστικό κ.ά.) και budget. Δημιουργεί ένα ενδεικτικό πρόγραμμα, συνδυάζοντας διαφορετικούς προορισμούς. Ωστόσο, δεν υποστηρίζει γεωεντοπισμό, προτάσεις βάσει της τρέχουσας τοποθεσίας ή ενσωματωμένο χάρτη. Επίσης, δεν περιλαμβάνει τεχνητή νοημοσύνη ή διασύνδεση με υπηρεσίες κρατήσεων μέσω API.

Stippl

Το Stippl [5] εστιάζει στον σχεδιασμό ενός ταξιδιού μέσα από ένα απλό περιβάλλον drag-and-drop. Είναι ιδανικό για συνεργασία, καθώς επιτρέπει την προσθήκη ημερομηνιών, σημειώσεων και εικόνων από πολλούς χρήστες. Ωστόσο, δεν υποστηρίζει χάρτες, διαδρομές ή έξυπνες προτάσεις βασισμένες σε εξωτερικά δεδομένα. Η χρήση του περιορίζεται κυρίως στην καταγραφή του προγράμματος και όχι στη δυναμική δημιουργία περιεχομένου.

Wanderlog

Το Wanderlog [6] είναι ένα εύχρηστο εργαλείο σχεδιασμού ταξιδιών που επιτρέπει τη δημιουργία προγράμματος με χάρτες, σημειώσεις και δυνατότητα εξαγωγής σε μορφή PDF. Υποστηρίζει συνεργασία μεταξύ χρηστών και ταξινόμηση προορισμών. Αν και φιλικό προς τον χρήστη, παραμένει στατικό. Δεν προσφέρει έξυπνες προτάσεις με βάση προτιμήσεις, τοποθεσία ή ώρα, ούτε χρησιμοποιεί τεχνητή νοημοσύνη για τη δυναμική δημιουργία περιεχομένου.

Συγκριτικός Πίνακας Εφαρμογών

Η παρακάτω συγκριτική παρουσίαση συνοψίζει τον στόχο, τις βασικές δυνατότητες και τις ελλείψεις καθεμιάς από τις δημοφιλέστερες εφαρμογές ταξιδιωτικού προγραμματισμού:

Εφαρμογή	Κύριος Σκοπός	Πλεονεκτήματα	Περιορισμοί
TripIt	Οργάνωση κρατήσεων από email	Αυτόματη δημιουργία χρονοδιαγράμματος	Δεν υποστηρίζει δημιουργία προγράμματος
TripAdvisor	Αξιολογήσεις και προτάσεις	Πλούσιο περιεχόμενο και κριτικές	Δεν παρέχει διαδραστικό σχεδιασμό ταξιδιού
Sygie Travel	Δημιουργία προγράμματος και διαδρομών	Offline χρήση, Διαδραστικός χάρτης	Μη εξατομικευμένες προτάσεις
RoutePerfect	Σχεδιασμός ταξιδιών στην Ευρώπη	Ευέλικτα φίλτρα	Δεν υποστηρίζει γεωεντοπισμό ή AI
Stippl	Καταγραφή και απεικόνιση προγράμματος	Διευκόλυνση συνεργασίας και σημειώσεων	Δεν ενσωματώνει χάρτες ή προτάσεις
Wanderlog	Προγραμματισμός διαδρομών με χάρτη	Συνεργασία, εξαγωγή προγράμματος	Περιορισμένη εξατομίκευση, απουσία AI

Πίνακας 2.1: Συγκριτική ανάλυση ταξιδιωτικών εφαρμογών

Η παραπάνω ανάλυση δείχνει ότι οι περισσότερες εφαρμογές καλύπτουν επιμέρους ανάγκες του χρήστη (π.χ. κράτηση, χάρτης, εμπειρίες), αλλά καμία δεν προσφέρει μια ολοκληρωμένη λύση που να καλύπτει ολόκληρη τη διαδικασία ενός ταξιδιού. Η εφαρμογή που αναπτύχθηκε στο πλαίσιο της παρούσας διπλωματικής ενσωματώνει τα πιο χρήσιμα στοιχεία των παραπάνω εφαρμογών και τα ενισχύει με λειτουργίες που απουσιάζουν, όπως προτάσεις για πτήσεις, ξενοδοχεία, σημεία ενδιαφέροντος και ημερήσιο πλάνο, βασισμένες σε TN (Gemini), καθώς και διαδραστικούς χάρτες με πλοήγηση. Όλα αυτά συγκεντρώνονται σε ένα απλό και φιλικό περιβάλλον χρήστη.

2.2 Συστήματα και Κριτήρια Επιλογής

Η ανάπτυξη της εφαρμογής στηρίχθηκε σε ένα σύνολο τεχνολογιών που επιλέχθηκαν με κριτήριο την ευχρηστία, την υψηλή απόδοση, καθώς και τη δυνατότητα διασύνδεσης με άλλα συστήματα και και μελλοντικής επέκτασης. Η τελική επιλογή των εργαλείων προέκυψε έπειτα από συγκριτική αξιολόγηση των διαθέσιμων λύσεων.

React Native έναντι Flutter, Swift, Kotlin

Για την ανάπτυξη του **front-end** της εφαρμογής χρησιμοποιήθηκε το **React Native**, καθώς επιτρέπει ανάπτυξη τη χρήση κοινής βάσης κώδικα για διαφορετικά λειτουργικά συστήματα, ενώ παρέχει native components για υψηλή απόδοση. [7]

Κριτήρια επιλογής

- Ενιαία βάση κώδικα μειώνοντας χρόνο και κόστος ανάπτυξης
- Μεγάλη και ενεργή κοινότητα υποστήριξης
- Συμβατότητα με υπάρχουσες βιβλιοθήκες της React

Σύγκριση με εναλλακτικές

- **Flutter:** Πολύ καλό για cross-platform αλλά απαιτεί χρήση Dart, μικρότερη κοινότητα σε σχέση με React Native
- **Swift/Kotlin:** Native ανάπτυξη με υψηλή απόδοση αλλά απαιτεί δύο ξεχωριστές εκδόσεις της εφαρμογής

Node.js και Express

Για την υλοποίηση του **backend** επιλέχθηκε το **Node.js** σε συνδυασμό με το framework **Express**. Πρόκειται για έναν γρήγορο και ευέλικτο συνδυασμό που χρησιμοποιείται ευρέως από προγραμματιστές σε όλο τον κόσμο. [8]

Κριτήρια επιλογής

- Πολύ καλή διαχείριση πολλαπλών αιτημάτων σε πραγματικό χρόνο
- Οργάνωση λειτουργιών μέσω Express (routes, APIs)
- Ενιαία γλώσσα με το front-end

Σύγκριση με εναλλακτικές

- **Django**: Ισχυρό framework Python, αλλά απαιτεί διαφορετική γλώσσα και μεγαλύτερη καμπύλη εκμάθησης
- **Laravel**: PHP framework με πλούσια χαρακτηριστικά αλλά λιγότερο αποδοτικό σε real-time εφαρμογές

MongoDB έναντι SQL

Για την αποθήκευση δεδομένων χρησιμοποιήθηκε η **MongoDB**, μια NoSQL βάση που δεν απαιτεί σταθερό σχήμα και είναι κατάλληλη για μεταβαλλόμενες δομές δεδομένων. [9]

Κριτήρια επιλογής

- Ευελιξία στη δομή (schema-less)
- Υψηλή ταχύτητα ανάκτησης
- Εύκολη ενσωμάτωση με Node.js μέσω Mongoose

Σύγκριση με εναλλακτικές

- **MySQL/PostgreSQL**: Σχεσιακές βάσεις που απαιτούν καθορισμένο σχήμα, και έχουν μικρότερη προσαρμοστικότητα σε μη τυποποιημένα δεδομένα

Google Maps APIs

Για τη διασύνδεση της εφαρμογής με χάρτες και σημεία ενδιαφέροντος αξιοποιήθηκαν τα **Google Maps APIs**. [10]

Κριτήρια επιλογής

- Ακρίβεια γεωγραφικών δεδομένων και παγκόσμια κάλυψη
- Πληρότητα τεκμηρίωσης και ευκολία ενσωμάτωσης
- Σταθερότητα και αξιοπιστία υπηρεσιών

Σύγκριση με εναλλακτικές

- **OpenStreetMap:** Δωρεάν και ανοικτό αλλά λιγότερο πλούσιο σε POIs και ακανόνιστη ενημέρωση δεδομένων
- **Mapbox:** Καλή προσαρμοστικότητα γραφικών, αλλά περιορισμένα δεδομένα σε σχέση με την Google

Κεφάλαιο 3

Τεχνολογίες & Εργαλεία

Στο κεφάλαιο αυτό παρουσιάζονται αναλυτικά οι τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής. Η ανάλυση οργανώνεται σε έξι κύριες κατηγορίες: **Front-end**, **Back-end**, **Βάση Δεδομένων**, **Διασύνδεση μέσω APIs**, **Ασφάλεια και Πιστοποίηση** και **Λειτουργίες Τεχνητής Νοημοσύνης**.

3.1 Front-end

Το front-end αποτελεί το τμήμα της εφαρμογής με το οποίο αλληλεπιδρά άμεσα ο χρήστης και αφορά την οπτική εμφάνιση, την πλοήγηση και τη συνολική εμπειρία χρήσης. Σχεδιάστηκε με στόχο να είναι γρήγορο, εύχρηστο και να προσαρμόζεται σε διαφορετικές συσκευές. Η ανάπτυξή του βασίστηκε στις εξής τεχνολογίες:

JavaScript

Η **JavaScript** [11] αποτέλεσε τη βασική γλώσσα προγραμματισμού που χρησιμοποιήθηκε για την ανάπτυξη του front-end. Επιτρέπει την άμεση αλληλεπίδραση με τον χρήστη και τη δυναμική αλλαγή στοιχείων στην οθόνη. Η ευρεία υποστήριξη από βιβλιοθήκες και frameworks, καθώς και η χρήση της τόσο στο front-end όσο και στο back-end (μέσω Node.js), μειώνουν την πολυπλοκότητα και διευκολύνουν την επαναχρησιμοποίηση κώδικα.

HTML και CSS

Η **HTML** [12] ορίζει τη δομή της διεπαφής χρήστη, ενώ η **CSS** [13] καθορίζει τη μορφοποίηση και το στυλ των στοιχείων. Παρότι η εφαρμογή αναπτύχθηκε σε περιβάλλον React Native, η γνώση HTML και CSS παραμένει σημαντική, καθώς η σύνταξη JSX και η λογική του styling βασίζονται σε αυτές. Η σωστή χρήση κανόνων CSS εξασφαλίζει συνέπεια στην εμφάνιση και ορθή προβολή σε διαφορετικές αναλύσεις.

JSX

Η **JSX** (JavaScript XML) [14] αποτελεί επέκταση της JavaScript, επιτρέποντας την ενσωμάτωση HTML κώδικα μέσα σε JavaScript συναρτήσεις. Με αυτόν τον τρόπο, η εμφάνιση και η λειτουργικότητα συνυπάρχουν στον ίδιο κώδικα, κάνοντάς τον πιο απλό και ευανάγνωστο. Στην εφαρμογή, η JSX χρησιμοποιήθηκε για την ανάπτυξη επαναχρησιμοποιήσιμων components, με καθαρή και κατανοητή δομή.

React Native

Το **React Native** [15] είναι ένα framework που επιτρέπει την ανάπτυξη εφαρμογών για Android και iOS, χρησιμοποιώντας JavaScript και React. Μετατρέπει τα components σε στοιχεία που αναγνωρίζει το ίδιο το λειτουργικό σύστημα, προσφέροντας υψηλή απόδοση και εμπειρία χρήσης αντίστοιχη με εφαρμογές που αναπτύσσονται σε γλώσσες όπως Java ή Swift. Η επιλογή του React Native επέτρεψε την ανάπτυξη της εφαρμογής από κοινή βάση κώδικα, μειώνοντας αισθητά τον χρόνο και το κόστος ανάπτυξης.

Expo

Το **Expo** [16] είναι ένα εργαλείο που συνεργάζεται με το React Native και διευκολύνει σημαντικά την ανάπτυξη εφαρμογών. Περιλαμβάνει έτοιμες βιβλιοθήκες για κοινές λειτουργίες, όπως πρόσβαση σε κάμερα, GPS και push notifications, χωρίς να χρειάζεται εγγενής κώδικας. Παρέχει αυτοματοποίηση στη διαδικασία build, ώστε η δημιουργία εκδόσεων της εφαρμογής να γίνεται εύκολα και γρήγορα. Επιπλέον, δίνει τη δυνατότητα άμεσης δοκιμής σε πραγματικές συσκευές, απλά σκανάροντας έναν κωδικό QR, χωρίς να απαιτούνται πολύπλοκες ρυθμίσεις ή σύνδεση με καλώδιο.

3.2 Back-end

To back-end είναι υπεύθυνο για την επιχειρησιακή λογική, την επεξεργασία δεδομένων, την επικοινωνία με τη βάση δεδομένων και την αλληλεπίδραση με εξωτερικά APIs. Η επιλογή των Node.js και Express βασίστηκε στην υψηλή απόδοση, την ευκολία επεκτασιμότητας και τη χρήση κοινής γλώσσας προγραμματισμού με το front-end, γεγονός που απλοποιεί την ανάπτυξη. Αναλυτικότερα:

Node.js

To **Node.js** [17] είναι ένα περιβάλλον που επιτρέπει την εκτέλεση JavaScript στην πλευρά του server. Η ασύγχρονη, event-driven αρχιτεκτονική του επιτρέπει την ταυτόχρονη διαχείριση πολλών αιτημάτων χωρίς καθυστερήσεις. Η χρήση της ίδιας γλώσσας με το front-end (JavaScript) προσφέρει ενιαία ανάπτυξη και μειώνει την ανάγκη για εκμάθηση πολλαπλών τεχνολογιών.

Express.js

To **Express.js** [18] είναι ένα ελαφρύ και ευέλικτο web framework για Node.js. Στην εφαρμογή χρησιμοποιήθηκε για τον ορισμό των routes, τη διαχείριση αιτημάτων και απαντήσεων, καθώς και την υλοποίηση των REST APIs. Η υποστήριξή του για middleware διευκολύνει την προσθήκη λειτουργιών, όπως καταγραφή ενεργειών (logging), διαχείριση ασφαλείας και έλεγχο πρόσβασης.

3.3 Βάση Δεδομένων

Για την αποθήκευση και ανάκτηση δεδομένων επιλέχθηκε μια λύση που συνδυάζει ευελιξία και δυνατότητα επέκτασης. Συγκεκριμένα, υιοθετήθηκε μια NoSQL προσέγγιση, η οποία προσφέρει γρήγορη πρόσβαση στις πληροφορίες και ευέλικτο χειρισμό δεδομένων που δεν ακολουθούν αυστηρά προκαθορισμένο σχήμα. Στο πλαίσιο αυτό, χρησιμοποιήθηκαν:

MongoDB

Η **MongoDB** [19] είναι μια NoSQL βάση δεδομένων που αποθηκεύει τις πληροφορίες σε έγγραφα (documents) μορφής JSON. Η απουσία αυστηρού σχήματος επιτρέπει την εύκολη

αποθήκευση διαφορετικών τύπων δεδομένων, κάτι ιδιαίτερα χρήσιμο σε εφαρμογές που εξελίσσονται συχνά. Η υποστήριξη για indexing και οι δυνατότητες ταχείας αναζήτησης συμβάλλουν σημαντικά στη βελτίωση της απόδοσης.

MongoDB Atlas

Το **MongoDB Atlas** [20] είναι η cloud εκδοχή της MongoDB. Προσφέρει έτοιμη υποδομή με αυτόματη προσαρμογή πόρων, δημιουργία αντιγράφων ασφαλείας και υψηλή διαθεσιμότητα. Έτσι αποφεύγεται η φυσική εγκατάσταση και διαχείριση server, μειώνοντας το κόστος και αυξάνοντας την αξιοπιστία της εφαρμογής.

3.4 APIs και Διασύνδεση

Η επικοινωνία της εφαρμογής με εξωτερικές υπηρεσίες και πηγές δεδομένων υλοποιείται μέσω προγραμματιστικών διεπαφών (*Application Programming Interfaces – APIs*). Η σωστή επιλογή και ενσωμάτωση αυτών των τεχνολογιών είναι σημαντική για την αξιοπιστία, την επεκτασιμότητα και την εμπειρία χρήσης της εφαρμογής. Τα βασικά εργαλεία και πρωτόκολλα που αξιοποιήθηκαν είναι:

HTTP και Status Codes

Το **HyperText Transfer Protocol (HTTP)** [21] αποτελεί το βασικό πρωτόκολλο επικοινωνίας και ανταλλαγής δεδομένων στον παγκόσμιο ιστό. Στην εφαρμογή αξιοποιείται ως το κύριο μέσο μεταφοράς όλων των αιτημάτων και αποκρίσεων, τόσο για την αλληλεπίδραση μεταξύ του front-end και του back-end, όσο και για την επικοινωνία με εξωτερικά APIs. Οι βασικές μέθοδοι που χρησιμοποιούνται είναι:

Μέθοδος	Λειτουργία
GET	Ανάκτηση δεδομένων από τον διακομιστή χωρίς να τροποποιούνται
POST	Αποστολή νέων δεδομένων για δημιουργία νέου πόρου
PUT	Ενημέρωση υπάρχοντος πόρου με πλήρη αντικατάσταση των δεδομένων του
PATCH	Μερική ενημέρωση υπάρχοντος πόρου
DELETE	Διαγραφή υπάρχοντος πόρου από τον διακομιστή

Πίνακας 3.1: Κύριες μέθοδοι HTTP και περιγραφή τους

Οι **HTTP Status Codes** (Κωδικοί Κατάστασης HTTP) [22] είναι τριψήφιοι αριθμοί που επιστρέφει ο διακομιστής ως απάντηση σε κάθε αίτημα, υποδεικνύοντας το αποτέλεσμα της επεξεργασίας του. Χωρίζονται σε κατηγορίες ανάλογα με το πρώτο ψηφίο και βοηθούν τον πελάτη να κατανοήσει αν η αίτηση ολοκληρώθηκε επιτυχώς, αν απαιτείται επιπλέον ενέργεια ή αν προέκυψε κάποιο σφάλμα. Παρουσιάζονται ορισμένοι από τους πιο συνηθισμένους κωδικούς κατάστασης:

Κωδικός	Περιγραφή
200 OK	Το αίτημα ολοκληρώθηκε επιτυχώς
201 Created	Δημιουργήθηκε νέος πόρος ως αποτέλεσμα του αιτήματος
400 Bad Request	Το αίτημα είχε μη έγκυρη μορφή ή ελλιπή δεδομένα
404 Not Found	Ο ζητούμενος πόρος δεν βρέθηκε
500 Internal Server Error	Εσωτερικό σφάλμα στον διακομιστή

Πίνακας 3.2: Συχνότεροι HTTP Status Codes

Η χρήση του HTTP, σε συνδυασμό με τις σαφείς μεθόδους και τους κωδικούς κατάστασης, παρέχει ευελιξία, συμβατότητα με πληθώρα υπηρεσιών και βιβλιοθηκών, καθώς και τυποποίηση στην επικοινωνία, γεγονός που διευκολύνει την ανάπτυξη, την ανίχνευση σφαλμάτων και τη συντήρηση της εφαρμογής.

REST APIs

Τα **RESTful APIs** [23] αποτελούν ένα πρότυπο αρχιτεκτονικής που ορίζει συγκεκριμένες αρχές για την ανταλλαγή δεδομένων μέσω του πρωτοκόλλου HTTP. Στην εφαρμογή, τα REST APIs χρησιμοποιούνται ως το κύριο μέσο επικοινωνίας μεταξύ του πελάτη (client) και του διακομιστή (server), με σαφή ορισμό πόρων και ενεργειών πάνω σε αυτούς. Η προσέγγιση REST προσφέρει:

- **Ευκολία ανάπτυξης** – Σαφής και καλά οργανωμένη δομή επικοινωνίας που απλοποιεί την υλοποίηση
- **Επεκτασιμότητα** – Δυνατότητα προσθήκης νέων λειτουργιών χωρίς εκτεταμένες αλλαγές στον υπάρχοντα κώδικα
- **Συμβατότητα** – Εύκολη ενσωμάτωση με άλλες εφαρμογές και υπηρεσίες

Axios

Το **Axios** [24] είναι μια δημοφιλής JavaScript βιβλιοθήκη για την εκτέλεση HTTP αιτημάτων, η οποία μπορεί να λειτουργήσει τόσο στον browser όσο και σε περιβάλλον Node.js. Στην εφαρμογή επιλέχθηκε λόγω της απλής και καθαρής σύνταξης, καθώς και των πρόσθετων δυνατοτήτων που προσφέρει, όπως:

- Υποστήριξη **Promises** για ασύγχρονη εκτέλεση και ευκολότερη διαχείριση της ροής δεδομένων
- **Interceptors** που επιτρέπουν την παρεμβολή λογικής στα αιτήματα και τις αποκρίσεις (π.χ. αυτόματη προσθήκη *authentication token*)
- **Ακύρωση αιτημάτων** σε περιπτώσεις που δεν απαιτείται πλέον η απόκριση (π.χ. αλλαγή αναζήτησης πριν ληφθεί το αποτέλεσμα)
- Αυτόματη μετατροπή των δεδομένων από και προς μορφή JSON

Η αξιοποίηση του Axios συνέβαλε στη μείωση της πολυπλοκότητας στη διαχείριση των δικτυακών κλήσεων, στην καλύτερη οργάνωση του κώδικα και στην αποτελεσματικότερη αντιμετώπιση σφαλμάτων.

Google APIs

Τα **Google APIs** [25] παρέχουν ένα σύνολο ισχυρών υπηρεσιών για χαρτογράφηση, εντοπισμό τοποθεσίας, αναζήτηση σημείων ενδιαφέροντος και υπολογισμό διαδρομών. Στην παρούσα εφαρμογή αξιοποιήθηκαν κυρίως οι υπηρεσίες:

- **Google Places API** – Προσφέρει αναλυτικές πληροφορίες για σημεία ενδιαφέροντος (*Points of Interest - POIs*), όπως αξιοθέατα, εστιατόρια και μουσεία. Περιλαμβάνει στοιχεία όπως ονομασία, περιγραφή, αξιολογήσεις χρηστών, φωτογραφίες και ωράρια λειτουργίας.
- **Google Directions API** – Υπολογίζει και προτείνει βέλτιστες διαδρομές μεταξύ δύο ή περισσότερων σημείων, παρέχοντας εκτίμηση χρόνου, απόστασης και εναλλακτικών επιλογών ανάλογα με το μέσο μεταφοράς.

Η ενσωμάτωση των Google APIs ενισχύει σημαντικά τη λειτουργικότητα και την εμπειρία χρήσης της εφαρμογής, καθώς προσφέρει:

- Ακριβή γεωγραφικά δεδομένα και εντοπισμό θέσης
- Πληροφορίες για POIs, εμπλουτισμένες με κριτικές και φωτογραφίες
- Υπολογισμό διαδρομών και χρόνων μετακίνησης
- Οπτική απεικόνιση σημείων και διαδρομών σε χάρτη

3.5 Ασφάλεια και Πιστοποίηση

Η διασφάλιση της προστασίας των δεδομένων και της ασφαλούς πρόσβασης των χρηστών αποτελεί κρίσιμο στοιχείο κάθε σύγχρονης εφαρμογής. Για την πιστοποίηση και εξουσιοδότηση των χρηστών χρησιμοποιήθηκε:

JWT (JSON Web Tokens)

Το **JWT** [26] είναι ένας ασφαλής τρόπος αποστολής πληροφοριών μεταξύ του client και του server σε μορφή JSON. Μετά τη σύνδεση του χρήστη, ο διακομιστής δημιουργεί ένα κρυπτογραφημένο *token* που περιέχει βασικές πληροφορίες, όπως το αναγνωριστικό χρήστη και

τη διάρκεια ισχύος. Το token αυτό αποθηκεύεται στον client και αποστέλλεται αυτόματα με κάθε επόμενο αίτημα, επιτρέποντας στον server να αναγνωρίζει τον χρήστη χωρίς να απαιτείται εκ νέου καταχώρηση στοιχείων σύνδεσης.

Τα κύρια πλεονεκτήματα της χρήσης JWT περιλαμβάνουν:

- **Ασφάλεια** – Δεν απαιτείται αποστολή κωδικών σε κάθε αίτημα
- **Ταχύτητα** – Η επαλήθευση του token γίνεται άμεσα, χωρίς πρόσθετα ερωτήματα σε βάση δεδομένων
- **Ευκολία** – Ιδανικό για εφαρμογές που αλληλεπιδρούν με πολλά και διαφορετικά συστήματα ή υπηρεσίες

3.6 Λειτουργίες Τεχνητής Νοημοσύνης

Για την εξατομίκευση των προτάσεων και τη δημιουργία αυτόματων ταξιδιωτικών προγραμμάτων, η εφαρμογή ενσωματώνει:

Gemini AI

Το **Gemini** [27] είναι ένα ισχυρό μοντέλο τεχνητής νοημοσύνης της Google, το οποίο αξιοποιήθηκε για την ανάλυση των προτιμήσεων του χρήστη και τη δημιουργία εξατομικευμένων ταξιδιωτικών προτάσεων. Η ικανότητά του να επεξεργάζεται φυσική γλώσσα και να προσαρμόζει δυναμικά τις απαντήσεις του σε πραγματικό χρόνο, προσδίδει στην εφαρμογή έναν διαδραστικό και “έξυπνο” χαρακτήρα.

Το σύστημα τεχνητής νοημοσύνης χρησιμοποιείται για:

- Αυτόματη αναζήτηση και πρόταση ξενοδοχείων, εστιατορίων και σημείων ενδιαφέροντος
- Δημιουργία εξατομικευμένων ημερήσιων προγραμμάτων, προσαρμοσμένων στις προτιμήσεις του χρήστη (π.χ. διαθέσιμο budget, τύπος ταξιδιώτη, ενδιαφέροντα)
- Προσαρμογή της εμπειρίας ανάλογα με το προφίλ χρήστη και την τοποθεσία

Η ενσωμάτωση του Gemini AI δίνει στην εφαρμογή τη δυνατότητα να προσφέρει μια πιο έξυπνη, προσωποποιημένη και αυτοματοποιημένη ταξιδιωτική εμπειρία, καλύπτοντας ανάγκες που συχνά δεν εξυπηρετούνται από τις παραδοσιακές ταξιδιωτικές εφαρμογές.

Κεφάλαιο 4

Αρχιτεκτονική & Υλοποίηση

Στο κεφάλαιο αυτό παρουσιάζεται η αρχιτεκτονική και η υλοποίηση της εφαρμογής. Αρχικά παρουσιάζεται η **βάση δεδομένων**, καλύπτοντας την διαδικασία εγκατάστασης, παραμετροποίησης και σύνδεσης με το backend, καθώς και τη δομή που χρησιμοποιείται για την αποθήκευση και τη διαχείριση των δεδομένων. Στη συνέχεια, αναλύονται το **backend** και το **frontend**, δίνοντας έμφαση στη διαδικασία εγκατάστασης, στις βασικές βιβλιοθήκες και στη δομή αρχείων και φακέλων κάθε τμήματος. Η ενότητα αυτή παρέχει μια ολοκληρωμένη εικόνα της τεχνικής υλοποίησης και διευκολύνει την κατανόηση της συνολικής λειτουργίας του συστήματος.

4.1 Βάση Δεδομένων

Η βάση δεδομένων της εφαρμογής βασίζεται στη **MongoDB**, ένα NoSQL σύστημα αποθήκευσης εγγράφων. Αρχικά χρησιμοποιήθηκε η υπηρεσία **MongoDB Atlas**, η οποία προσφέρει cloud υποδομή για απομακρυσμένη πρόσβαση και εύκολη διαχείριση. Αργότερα, για τις ανάγκες ανάπτυξης και δοκιμών, το σύστημα μεταφέρθηκε σε τοπικό περιβάλλον, με την εγκατάσταση του **MongoDB Community Server** και τη χρήση του **MongoDB Compass** για οπτική διαχείριση των δεδομένων.

Setup και Διαμόρφωση

MongoDB Atlas

1. Δημιουργήθηκε λογαριασμός και νέο *cluster* στην πλατφόρμα MongoDB Atlas

2. Ορίστηκε *database user* με όνομα χρήστη και κωδικό πρόσβασης, και προστέθηκε IP whitelist για απομακρυσμένη σύνδεση
3. Δημιουργήθηκε η βάση δεδομένων *travelPlanner* και τα αρχικά *collections* (*users*, *favorites*, *trips*, *tripplans*)
4. Στο αρχείο ρυθμίσεων *.env* του backend ορίστηκε το connection string:

```
MONGODB_URI = mongodb+srv://<username>:<password>@<cluster>  
.mongodb.net/?retryWrites=true&w=majority
```

Τοπική εγκατάσταση

1. Εγκαταστάθηκε το **MongoDB Community Server** και το **MongoDB Compass** για γραφική διαχείριση και επεξεργασία εγγράφων
2. Ενημερώθηκε το *.env* του backend ώστε να χρησιμοποιεί το τοπικό connection string:

```
MONGODB_URI = mongodb://localhost:27017/travelPlanner
```

Διασύνδεση με Backend

Η επικοινωνία της βάσης με το backend υλοποιήθηκε μέσω της βιβλιοθήκης **Mongoose**, η οποία διευκολύνει την οργανωμένη διαχείριση των δεδομένων της MongoDB μέσα από το Node.js. Με τη Mongoose ορίζονται **schemas** και **models** που περιγράφουν τη δομή των collections, ενώ παρέχονται απλές μέθοδοι για δημιουργία, ανάγνωση, ενημέρωση και διαγραφή δεδομένων (CRUD). Υποστηρίζεται επίσης η σύνδεση και η αναφορά μεταξύ collections, διασφαλίζοντας τη συνοχή και την ακεραιότητα των δεδομένων.

Η σύνδεση στο MongoDB, γίνεται κατά την εκκίνηση του backend, όπως φαίνεται στο παρακάτω τμήμα κώδικα:

```
// MongoDB connection  
mongoose.connect(process.env.MONGO_URI)  
Tabnine | Edit | Test | Explain | Document  
.then(() => console.log('MongoDB Connected'))  
Tabnine | Edit | Test | Explain | Document  
.catch(err => console.error('DB Error:', err));
```

Σχήμα 4.1: Σύνδεση του backend με τη MongoDB μέσω Mongoose

Δομή της Βάσης Δεδομένων

Η βάση δεδομένων **travelPlanner** οργανώνεται σε τέσσερα κύρια collections, καθένα από τα οποία επιτελεί συγκεκριμένο ρόλο στην εφαρμογή:

1. **users** - Αποθηκεύει τα στοιχεία των χρηστών

- `email`: διεύθυνση email του χρήστη
- `password`: κωδικός πρόσβασης σε hashed μορφή για ασφάλεια
- `name`: ονοματεπώνυμο ή ψευδώνυμο του χρήστη

2. **favorites** - Περιέχει τις αγαπημένες τοποθεσίες των χρηστών

- `userId`: αναφορά στον χρήστη που αποθήκευσε την τοποθεσία
- `placeId`: μοναδικό αναγνωριστικό τοποθεσίας
- `name`: ονομασία τοποθεσίας
- `address`: ταχυδρομική διεύθυνση
- `rating`: βαθμολογία
- `photoReference`: σύνδεσμος σε φωτογραφία της τοποθεσίας
- `location`: γεωγραφικές συντεταγμένες (πλάτος και μήκος)

3. **trips** - Αποθηκεύει τα βασικά στοιχεία κάθε ταξιδιού που δημιουργεί ο χρήστης

- `userEmail`: email του δημιουργού του ταξιδιού
- `tripData`: πληροφορίες τοποθεσίας, φωτογραφίες και σύνδεσμοι
- `traveler`: τύπος ταξιδιού, περιγραφή και αριθμός ατόμων
- `startDate`: ημερομηνία έναρξης ταξιδιού
- `endDate`: ημερομηνία λήξης ταξιδιού
- `budget`: προϋπολογισμός ταξιδιού

- `activities`: λίστα δραστηριοτήτων
- `isFavorite`: ένδειξη αν το ταξίδι έχει προστεθεί στα αγαπημένα
- `members`: λίστα συμμετεχόντων στο ταξίδι

4. tripplans - Περιλαμβάνει το αναλυτικό πρόγραμμα ταξιδιών που έχουν δημιουργηθεί με τη βοήθεια AI

- `tripDetails`: συνοπτικές πληροφορίες (τοποθεσία, διάρκεια, προϋπολογισμός, θεματική)
- `flightDetails`: στοιχεία πτήσεων (αεροπορική εταιρεία, τιμή, σύνδεσμος κράτησης)
- `hotelOptions`: λίστα ξενοδοχείων με τιμές, βαθμολογίες και φωτογραφίες
- `dailyItinerary`: αναλυτικό ημερήσιο πρόγραμμα δραστηριοτήτων (τοποθεσίες, τιμές εισιτηρίων και διάρκεια)

4.2 Backend

Το backend της εφαρμογής αναπτύχθηκε με **Node.js** και **Express**. Παρέχει REST endpoints για εγγραφή και είσοδο χρηστών, διαχείριση ταξιδιών και αγαπημένων, καθώς και ένα endpoint που επικοινωνεί με το μοντέλο Gemini, ώστε να δημιουργεί προτάσεις ταξιδιών με ασφάλεια από την πλευρά του server.

Οδηγίες εγκατάστασης & εκτέλεσης

Για τη δημιουργία του backend ακολουθήθηκε η εξής διαδικασία:

1. Αρχικοποίηση έργου

Δημιουργήθηκε νέος φάκελος για το backend και εκτελέστηκε η εντολή αρχικοποίησης του npm:

```
mkdir backend && cd backend  
npm init -y
```

Σχήμα 4.2: Δημιουργία φακέλου και αρχικοποίηση

2. Εγκατάσταση βασικών πακέτων

Εγκαταστάθηκαν οι απαραίτητες βιβλιοθήκες για τη λειτουργία του server, τη σύνδεση με τη βάση δεδομένων, την αυθεντικοποίηση χρηστών και την επικοινωνία με το μοντέλο Gemini:

```
npm install express cors dotenv mongoose jsonwebtoken  
bcryptjs @google/generative-ai apple-signin-auth
```

Σχήμα 4.3: Εγκατάσταση απαραίτητων βιβλιοθηκών μέσω npm

3. Προσθήκη scripts στο package.json

Στο αρχείο `package.json` προστέθηκαν scripts για ευκολία κατά την ανάπτυξη (dev) και την εκτέλεση σε παραγωγή (start):

```
"scripts": {  
  "start": "node server",  
  "dev": "nodemon server.js"  
},
```

Σχήμα 4.4: Ρύθμιση scripts για ανάπτυξη και παραγωγή στο `package.json`

4. Δημιουργία αρχείου περιβάλλοντος .env

Δημιουργήθηκε αρχείο `.env` που περιλαμβάνει τις απαραίτητες μεταβλητές για τη σύνδεση με τη βάση και τα κλειδιά API:

```
PORT=5000  
MONGO_URI=<σύνδεσμος MongoDB>  
JWT_SECRET=<ισχυρό μυστικό>  
GEMINI_API_KEY=<κλειδί @google/generative-ai>  
APPLE_SERVICE_ID=<Apple Sign-In>
```

Σχήμα 4.5: Αρχείο `.env` με μεταβλητές περιβάλλοντος

5. Εκκίνηση του server

Εκτέλεση του server σε περιβάλλον ανάπτυξης ή παραγωγής:

```
npm run dev    # Ανάπτυξη με αυτόματη επανεκκίνηση  
npm start     # Παραγωγή
```

Σχήμα 4.6: Εκτέλεση του server σε ανάπτυξη και παραγωγή

Βασικές Βιβλιοθήκες

Το αρχείο `package.json` περιλαμβάνει τις κύριες βιβλιοθήκες που χρησιμοποιεί το backend, οι οποίες καλύπτουν την υλοποίηση του server, τη διαχείριση δεδομένων, την ασφάλεια και την ενσωμάτωση εξωτερικών υπηρεσιών. Πιο συγκεκριμένα:

express

Framework για την ανάπτυξη του server. Χρησιμοποιείται για τον ορισμό των διαδρομών (*routes*) που καλύπτουν λειτουργίες όπως η αυθεντικοποίηση, η διαχείριση ταξιδιών και αγαπημένων, καθώς και η επικοινωνία με το μοντέλο Gemini. [18]

cors

Ρυθμίζει την πολιτική Cross-Origin Resource Sharing (CORS), επιτρέποντας στο frontend της εφαρμογής να στέλνει αιτήματα στο backend ακόμη και όταν βρίσκονται σε διαφορετικά domains ή θύρες (ports). [28]

dotenv

Φορτώνει αυτόματα τις μεταβλητές περιβάλλοντος από το αρχείο `.env` και τις καθιστά προσβάσιμες στον κώδικα μέσω της `process.env`. Έτσι, ευαίσθητες ρυθμίσεις όπως κλειδιά API ή URI βάσης δεδομένων δεν αποθηκεύονται απευθείας στον κώδικα, βελτιώνοντας την ασφάλεια και διευκολύνοντας την αλλαγή τους όταν χρειάζεται. [29]

jsonwebtoken (JWT)

Δημιουργεί και επαληθεύει JSON Web Tokens για την αυθεντικοποίηση χρηστών. Στο backend χρησιμοποιείται για την έκδοση token μετά την επιτυχή είσοδο και για την επαλήθευσή του σε προστατευμένα endpoints μέσω του middleware αυθεντικοποίησης. [26]

mongoose

Αποτελεί ένα ODM (Object Data Modeling) εργαλείο για τη MongoDB. Επιτρέπει τον ορισμό σχημάτων και μοντέλων, τον έλεγχο εγκυρότητας δεδομένων και την εύκολη επικοινωνία με τη βάση μέσω JavaScript αντικειμένων. Χρησιμοποιείται για τα μοντέλα `User`, `Trip` και `Favorites`, διευκολύνοντας τις CRUD λειτουργίες με ασφάλεια και συνέπεια. [30]

bcryptjs

Χρησιμοποιείται για την ασφαλή αποθήκευση κωδικών πρόσβασης. Πριν καταχωρηθεί ένας κωδικός στη βάση δεδομένων, μετατρέπεται σε hash με τη βοήθεια της `bcryptjs`, κάνοντας πολύ δύσκολη την αποκωδικοποίησή του ακόμη και σε περίπτωση παραβίασης της βάσης. [31]

google/generative-ai

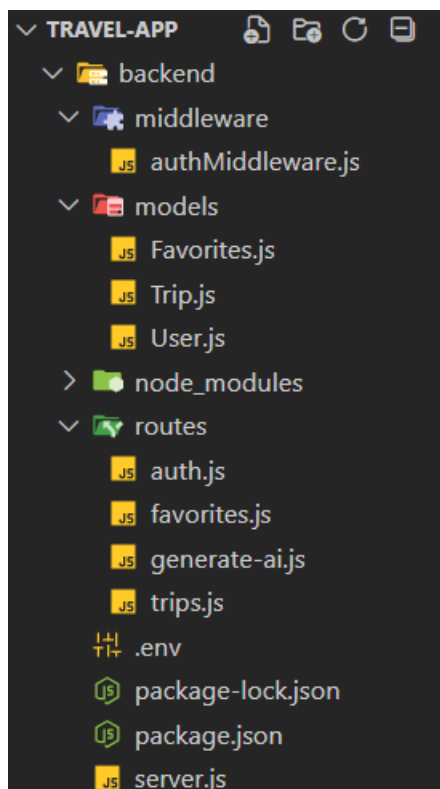
Επίσημη βιβλιοθήκη της Google για πρόσβαση στα μοντέλα Gemini από Node.js. Στο backend χρησιμοποιείται για την δημιουργία προτάσεων ταξιδιού με βάση παραμέτρους που δίνει ο χρήστης (π.χ. πόλη, μέρες, ενδιαφέροντα). Οι κλήσεις γίνονται αποκλειστικά από τον server ώστε να προστατεύεται το API key. [32]

apple-signin-auth

Βιβλιοθήκη για την υποστήριξη σύνδεσης χρηστών μέσω Apple ID. Επαληθεύει τα tokens που στέλνει το frontend κατά το Apple Sign-In και δημιουργεί τα απαραίτητα client secrets για την ολοκλήρωση της διαδικασίας. [33]

Δομή αρχείων και φακέλων

Η οργάνωση του backend ακολουθεί μια καθαρή και λογική δομή, ώστε κάθε κομμάτι της εφαρμογής να έχει σαφή ρόλο και να είναι εύκολο να εντοπιστεί και να τροποποιηθεί. Τα αρχεία χωρίζονται σε κατηγορίες σύμφωνα με τη λειτουργία τους, όπως το βασικό αρχείο εκκίνησης, τα μοντέλα δεδομένων, τα middleware και οι διαδρομές (routes). [34]



Σχήμα 4.7: Δομή αρχείων και φακέλων του backend

server.js

Αποτελεί το κεντρικό αρχείο εκκίνησης της εφαρμογής, όπου γίνεται η αρχικοποίηση και ρύθμιση του backend. Περιλαμβάνει:

- Δημιουργία του Express server
- Σύνδεση με τη βάση δεδομένων MongoDB μέσω Mongoose
- Ενεργοποίηση βασικών middleware, όπως CORS και επεξεργασία JSON αιτημάτων
- Καταχώρηση όλων των διαδρομών (routes) με κοινό prefix /api

```
const app = express();  
app.use(cors());  
app.use(express.json());
```

Σχήμα 4.8: Ενεργοποίηση βασικών middleware στο server.js

```
// Routes
app.use('/api/auth', require('./routes/auth'));
app.use('/api/trips', require('./routes/trips'));
app.use('/api/favorites', require('./routes/favorites'));
app.use('/api/generate-ai', require('./routes/generate-ai'));
```

Σχήμα 4.9: Καταχώρηση των διαδρομών (routes)

```
// Start server
Tabnine | Edit | Test | Explain | Document
app.listen(process.env.PORT, () => {
  console.log(`Server running on port ${process.env.PORT}`);
});
```

Σχήμα 4.10: Εκκίνηση του Express server

middleware

Το συγκεκριμένο middleware είναι υπεύθυνο για τον έλεγχο πρόσβασης στα προστατευμένα endpoints της εφαρμογής. Η λειτουργία του περιλαμβάνει τα ακόλουθα βήματα:

- Έλεγχο για ύπαρξη κεφαλίδας με μορφή `Authorization: Bearer <token>`

```
const token = req.header('Authorization')?.replace('Bearer ', '');
```

- Επαλήθευση του JWT με χρήση του `JWT_SECRET`

```
const decoded = jwt.verify(token, process.env.JWT_SECRET);
```

- Αναζήτηση του χρήστη στη βάση δεδομένων και αποθήκευση των στοιχείων του

```
req.user = await User.findById(decoded.id).select('-password');
```

- Απόρριψη της αίτησης αν το token λείπει, είναι άκυρο ή ο χρήστης δεν βρεθεί

```
if (!token) {
  return res.status(401).json({ msg: 'No token, authorization denied' });
}
```

```
if (!req.user) {
  return res.status(401).json({ msg: 'User not found' });
}
```

Παράδειγμα χρήσης του middleware:

```
router.get('/me', authMiddleware, (req, res) => {  
  // req.user is set in middleware  
  console.log('Found user: ', req.user);  
  res.json(req.user);  
});
```

Σχήμα 4.11: Χρήση του middleware για προστατευμένο endpoint

models

Περιλαμβάνει τα Mongoose μοντέλα της εφαρμογής, τα οποία ορίζουν τη δομή (schema) και τους κανόνες επικύρωσης των δεδομένων που αποθηκεύονται στη βάση MongoDB. Κάθε μοντέλο αντιστοιχεί σε μια συλλογή (collection) της βάσης:

- User: Στοιχεία λογαριασμού και διαπιστευτήρια χρήστη
- Trip και TripPlan: Πληροφορίες σχετικά με ταξίδια και λεπτομερή προγράμματα
- Favorite: Αγαπημένα μέρη που έχει αποθηκεύσει ο χρήστης

Παράδειγμα Mongoose μοντέλου:

```
const mongoose = require('mongoose');  
  
const UserSchema = new mongoose.Schema({  
  email: { type: String, required: true, unique: true },  
  password: { type: String, required: true },  
  name: { type: String },  
});  
  
module.exports = mongoose.model('User', UserSchema);
```

Σχήμα 4.12: Παράδειγμα ορισμού μοντέλου User σε Mongoose

routes

Περιέχει τα αρχεία που ορίζουν τα endpoints του backend, οργανωμένα σε θεματικές ενότητες. Κάθε αρχείο αντιστοιχεί σε ένα σύνολο λειτουργιών που σχετίζονται με συγκεκριμένο κομμάτι της εφαρμογής:

- `auth.js`: Διαχείριση χρηστών (εγγραφή, είσοδος, Apple Sign-In, ενημέρωση προφίλ, λογαριασμού, αλλαγή κωδικού πρόσβασης)
- `trips.js`: Δημιουργία, ανάγνωση, ενημέρωση και διαγραφή ταξιδιών, διαχείριση μελών, αγαπημένων τοποθεσιών και σχεδίων ταξιδιού
- `favorites.js`: Αποθήκευση, ανάκτηση και διαγραφή αγαπημένων τοποθεσιών
- `generate-ai.js`: Δημιουργία εξατομικευμένων προτάσεων ταξιδιού μέσω του μοντέλου Gemini

Παράδειγμα ορισμού route:

```
// Create new trip
Tabnine | Edit | Test | Explain | Document
router.post('/create', async (req, res) => {
  try {
    const { userEmail, tripData } = req.body;

    const trip = new Trip({
      userEmail: userEmail,
      tripData: tripData,
      createdAt: new Date()
    });

    await trip.save()
    console.log('New trip: ', trip)
    res.status(201).json({ success: true, tripId: trip._id })
  } catch (err) {
    res.status(500).json({ error: 'Failed to create trip' })
  }
})
```

Σχήμα 4.13: Παράδειγμα route για δημιουργία ταξιδιού

Διαδρομές Εφαρμογής

Τα routes ορίζουν τα διαθέσιμα REST endpoints της εφαρμογής και τη λογική που εκτελείται για κάθε αίτημα. Είναι οργανωμένα σε τέσσερα βασικά αρχεία (`auth.js`, `trips.js`, `favorites.js`, `generate-ai.js`), καθένα με σαφή θεματική και ξεχωριστό ρόλο στη λειτουργία του backend.

Διαδρομές Αυθεντικοποίησης (auth.js)

Αφορούν τη διαχείριση χρηστών, συμπεριλαμβανομένης της εγγραφής, της εισόδου, της ενημέρωσης στοιχείων προφίλ, καθώς και την υποστήριξη σύνδεσης μέσω Apple Sign-In.

Μέθοδος	Διαδρομή	Περιγραφή
POST	/api/auth/register	Εγγραφή νέου χρήστη
POST	/api/auth/login	Είσοδος χρήστη
DELETE	/api/auth/delete	Διαγραφή χρήστη και των ταξιδιών του
POST	/api/auth/apple-login	Είσοδος μέσω Apple
GET	/api/auth/me	Επιστροφή στοιχείων συνδεδεμένου χρήστη
PUT	/api/auth/update-profile	Ενημέρωση προφίλ χρήστη
POST	/api/auth/reset-password	Αλλαγή κωδικού χρήστη

Πίνακας 4.1: Endpoints του auth.js

Διαδρομές Αγαπημένων (favorites.js)

Αφορούν την αποθήκευση, ανάκτηση και διαγραφή αγαπημένων τοποθεσιών ενός χρήστη.

Μέθοδος	Διαδρομή	Περιγραφή
GET	/api/favorites/:userId	Λήψη όλων των αγαπημένων για συγκεκριμένο χρήστη
POST	/api/favorites	Προσθήκη νέας αγαπημένης τοποθεσίας
DELETE	/api/favorites/:userId/:placeId	Διαγραφή αγαπημένης τοποθεσίας

Πίνακας 4.2: Endpoints του favorites.js

Διαδρομές Ταξιδιών (trips.js)

Περιλαμβάνουν τα endpoints για τη διαχείριση ταξιδιών, όπως δημιουργία, προβολή, ενημέρωση και διαγραφή, καθώς και λειτουργίες για προσθήκη μελών και επισήμανση ταξιδιών ως αγαπημένα.

Μέθοδος	Διαδρομή	Περιγραφή
POST	/api/trips	Αποθήκευση νέου ταξιδιού απο AI
GET	/api/trips/:email	Λήψη όλων των ταξιδιών ενός χρήστη
GET	/api/trips/member/:email	Λήψη ταξιδιών όπου ο χρήστης είναι μέλος
POST	/api/trips/create	Δημιουργία βασικού ταξιδιού
POST	/api/trips/update	Ενημέρωση δραστηριοτήτων
DELETE	/api/trips/:tripId	Διαγραφή ταξιδιού
PATCH	/api/trips/favorite/:tripId	Προσθήκη ταξιδιού στα αγαπημένα
GET	/api/trips/favorites/:email	Λήψη αγαπημένων ταξιδιών χρήστη
PATCH	/api/trips/addUser	Προσθήκη μέλους σε ταξίδι (με email)

Πίνακας 4.3: Endpoints του `trips.js`**Διαδρομή AI (`generate-ai.js`)**

Η διαδρομή αυτή παρέχει πρόσβαση από τον server στο μοντέλο **Gemini** της Google, επιτρέποντας την παραγωγή εξατομικευμένων προτάσεων ταξιδιού με βάση δεδομένα που παρέχει ο χρήστης. Η κλήση του μοντέλου πραγματοποιείται αποκλειστικά από τον server για λόγους ασφαλείας, ώστε να προστατεύεται το API key και να εφαρμόζονται περιορισμοί και έλεγχοι.

Μέθοδος	Διαδρομή	Περιγραφή
POST	/api/generate-ai/plan-trip	Δημιουργεί εξατομικευμένο πλάνο ταξιδιού από prompt μέσω Gemini

Πίνακας 4.4: Endpoint του `generate-ai.js`**Λειτουργία του endpoint:**

1. Ο client στέλνει αίτημα POST με το `prompt`
2. Ο server καλεί το Gemini API χρησιμοποιώντας τη βιβλιοθήκη `@google/generative-ai`
3. Το μοντέλο παράγει το πλάνο ταξιδιού και το επιστρέφει στον server
4. Ο server αποστέλλει την απάντηση στον client, όπου μπορεί να επεξεργαστεί (π.χ. με `JSON.parse`) για εμφάνιση ή αποθήκευση

```
Tabnine | Edit | Test | Explain | Document
router.post('/plan-trip', async (req, res) => {
  const { prompt } = req.body;

  if (!prompt) {
    return res.status(400).json({ error: 'Prompt is required' });
  }

  try {
    const result = await chatSession.sendMessage(prompt);
    const text = result.response.text();

    res.json({ result: text });
  } catch (error) {
    console.error('Error generating travel plan:', error);
    res.status(500).json({ error: 'Something went wrong', details: error.message });
  }
});
```

Σχήμα 4.14: Λειτουργία του endpoint για σχεδιασμό ταξιδιού με AI

4.3 Frontend

Το frontend της εφαρμογής αναπτύχθηκε με **React Native** και το framework **Expo**, το οποίο διευκολύνει την ανάπτυξη, τον έλεγχο και την εκτέλεση σε πραγματικές συσκευές ή εξομοιωτές. Η εφαρμογή επικοινωνεί με το backend μέσω REST API και παρέχει το περιβάλλον χρήστη για την αναζήτηση, οργάνωση και προβολή ταξιδιών.

Οδηγίες εγκατάστασης & εκτέλεσης

Για τη δημιουργία και ρύθμιση του frontend ακολουθήθηκε η εξής διαδικασία:

1. Δημιουργία νέου έργου με Expo

Δημιουργήθηκε φάκελος για το frontend και εκτελέστηκε η εντολή αρχικοποίησης μέσω Expo CLI:

```
npx create-expo-app <name>
```

2. Εκτέλεση της εφαρμογής

Εκκίνηση της εφαρμογής σε περιβάλλον ανάπτυξης με χρήση του Expo:

```
npx expo start
```

Στη συνέχεια, η εφαρμογή μπορεί να εκτελεστεί:

- **Σε πραγματική συσκευή:** Μέσω της εφαρμογής Expo Go [35], σκανάροντας τον QR code που εμφανίζεται
- **Σε εξομοιωτή:** Με χρήση Android Studio [36], εκκινώντας τον emulator από το τερματικό

Βασικές Βιβλιοθήκες

Το αρχείο `package.json` περιλαμβάνει τις βασικές βιβλιοθήκες που χρησιμοποιεί το frontend, οι οποίες καλύπτουν λειτουργίες όπως η πλοήγηση, η απεικόνιση χαρτών, η εμφάνιση εικονιδίων, η διαχείριση ημερομηνιών, η επικοινωνία με το backend και οι κύριες δυνατότητες του Expo. Οι σημαντικότερες από αυτές είναι:

react-native

Το βασικό πλαίσιο πάνω στο οποίο χτίζεται η εφαρμογή. Παρέχει τα απαραίτητα συστατικά για τη ανάπτυξη mobile εφαρμογών με JavaScript και React, μετατρέποντας τον κώδικα σε native στοιχεία διεπαφής χρήστη. Με αυτόν τον τρόπο, η εφαρμογή μπορεί να αξιοποιεί πλήρως τις δυνατότητες των συσκευών iOS και Android, όπως η απόδοση, οι αισθητήρες και οι λειτουργίες του συστήματος. [15]

expo

Το κύριο framework της εφαρμογής που βασίζεται στο React Native και περιλαμβάνει τη λειτουργικότητα του Expo. Προσφέρει έτοιμα APIs για λειτουργίες όπως τοποθεσία, εικόνες, αποθήκευση δεδομένων και πρόσβαση σε διάφορες δυνατότητες της συσκευής. Διευκολύνει την ανάπτυξη cross-platform εφαρμογών, αποφεύγοντας την ανάγκη για πολύπλοκες ρυθμίσεις. [16]

axios

Βιβλιοθήκη για την αποστολή και λήψη δεδομένων από το backend μέσω HTTP αιτημάτων, όπως GET και POST. Υποστηρίζει την αυτόματη προσθήκη headers (π.χ. *Authorization*) και διευκολύνει την ασφαλή και αποδοτική επικοινωνία με τον server. Στην εφαρμογή χρησιμοποιείται για την ανάκτηση πληροφοριών ταξιδιών, αγαπημένων τοποθεσιών και απαντήσεων

από το μοντέλο Gemini. [24]

react-navigation

Το πιο διαδεδομένο σύστημα πλοήγησης για εφαρμογές React Native, το οποίο υποστηρίζει πολλαπλούς τύπους πλοήγησης, όπως *stack*, *tabs* και *drawer*. Στην εφαρμογή χρησιμοποιείται σε συνδυασμό με το *expo-router* για την εναλλαγή οθονών, τη διατήρηση της κατάστασης πλοήγησης και την εξασφάλιση ομαλών και γρήγορων μεταβάσεων μεταξύ διαφορετικών μερών της εφαρμογής. [37]

expo-router

Σύστημα δρομολόγησης που βασίζεται στο *React Navigation* και επιτρέπει την οργάνωση των οθονών της εφαρμογής μέσω *file-based routing*. Κάθε αρχείο στον φάκελο *app/* αντιστοιχεί αυτόματα σε μια διαδρομή και υποστηρίζει λειτουργίες όπως *nested navigation*, *dynamic routes* και *deep linking*, μειώνοντας τον απαιτούμενο κώδικα και απλοποιώντας τη διαχείριση της πλοήγησης. [38]

react-native-maps

Βιβλιοθήκη που ενσωματώνει native χάρτες (Google Maps για Android και iOS, Apple Maps για iOS) στην εφαρμογή. Χρησιμοποιείται για την προβολή τοποθεσιών, σημείων ενδιαφέροντος και διαδρομών, επιτρέποντας στον χρήστη να πλοηγείται στον χάρτη και να ενημερώνεται για αποστάσεις και πορείες. [39]

moment

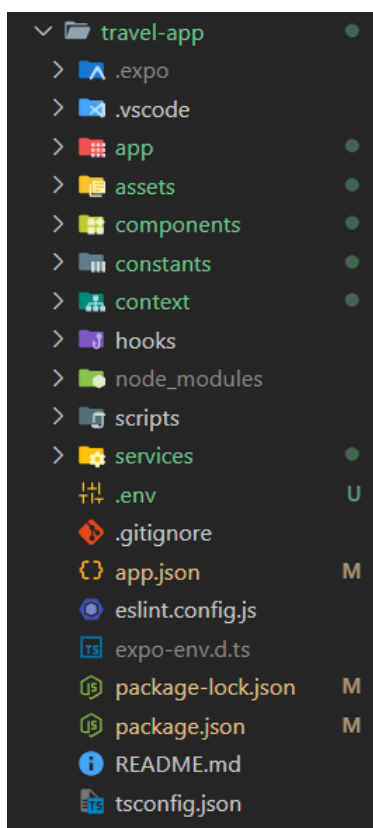
Βιβλιοθήκη για τη διαχείριση και μορφοποίηση ημερομηνιών και χρόνου. Στην εφαρμογή χρησιμοποιείται για την προβολή ημερομηνιών ταξιδιών σε φιλική μορφή, τον υπολογισμό χρονικών διαστημάτων (π.χ. διάρκεια ταξιδιού) και την εφαρμογή τοπικών ρυθμίσεων (π.χ. ελληνική μορφή ημερομηνιών). [40]

@expo/vector-icons

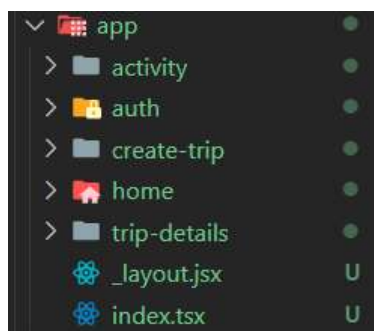
Συλλογή εικονιδίων ενσωματωμένη στο Expo, η οποία περιλαμβάνει δημοφιλή πακέτα όπως *Ionicons*, *Material Icons* και *FontAwesome*. Χρησιμοποιείται σε κουμπιά, καρτέλες και λίστες, βελτιώνοντας την εμπειρία χρήστη και προσφέροντας αναγνωρίσιμα σύμβολα. [41]

Δομή αρχείων και φακέλων

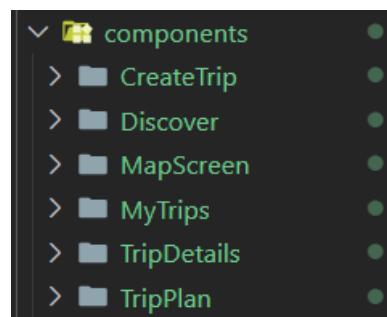
Η δομή των φακέλων της εφαρμογής έχει σχεδιαστεί ώστε να παρέχει καθαρή οργάνωση του κώδικα, σαφή διαχωρισμό των λειτουργικών ενότητων και ευκολία στη μελλοντική επέκταση. Κάθε φάκελος αντιστοιχεί σε συγκεκριμένη λειτουργική ενότητα της εφαρμογής.



Σχήμα 4.15: Δομή αρχείων και φακέλων του frontend



Σχήμα 4.16: Δομή αρχείων και φακέλων του φακέλου app



Σχήμα 4.17: Δομή αρχείων και φακέλων του φακέλου components

Φάκελος app (Routing & Screens)

Κύριος φάκελος πλοήγησης της εφαρμογής, σύμφωνα με την αρχιτεκτονική του expo-router.

Περιλαμβάνει υποφακέλους που αντιστοιχούν σε διαφορετικές ροές ή οθόνες, όπως:

- **activity** – Διαχείριση και προβολή δραστηριοτήτων ταξιδιού
 - `add-activity.js`: Προσθήκη νέας δραστηριότητας στο ταξίδι
 - `day-details.js`: Προβολή δραστηριοτήτων ανά ημέρα
 - `map-view.js`: Εμφάνιση δραστηριοτήτων σε χάρτη
- **auth** – Διαχείριση αυθεντικοποίησης χρήστη, με οθόνες για σύνδεση, εγγραφή και επαναφορά κωδικού πρόσβασης
- **create-trip** – Δημιουργία νέου ταξιδιού (AI-generated ή custom)
 - `search-place.jsx`: Αναζήτηση προορισμού
 - `select-dates.jsx`: Επιλογή ημερομηνιών ταξιδιού
 - `select-budget.jsx`: Καθορισμός προϋπολογισμού
 - `select-traveller.jsx`: Επιλογή αριθμού και τύπου ταξιδιωτών
 - `review-trip.jsx`: Τελική επισκόπηση πριν την αποθήκευση
 - `generate-trip.jsx`: Αυτόματη δημιουργία προτεινόμενου ταξιδιού
 - `custom-trip.js`: Δημιουργία προσαρμοσμένου ταξιδιού
- **home** – Κεντρική οθόνη και βασικές λειτουργίες εφαρμογής
 - `discover.jsx`: Ανακάλυψη τοποθεσιών και δραστηριοτήτων σε χάρτη
 - `favorites.jsx`: Λίστα αγαπημένων τοποθεσιών και ταξιδιών
 - `mytrip.jsx`: Προβολή αποθηκευμένων ταξιδιών χρήστη
 - `profile.jsx`: Προβολή και επεξεργασία προφίλ χρήστη
- **trip-details** – Αναλυτική παρουσίαση στοιχείων ταξιδιού
- **_layout.jsx** – Διάταξη και ροή πλοήγησης της εφαρμογής
- **index.tsx** – Κύριο σημείο εκκίνησης της εφαρμογής

Φάκελος components

Περιέχει επαναχρησιμοποιήσιμα στοιχεία διεπαφής (UI components), οργανωμένα σε φακέλους ανά λειτουργική ενότητα:

- **CreateTrip** – Επιλογή παραμέτρων για τη δημιουργία νέου ταξιδιού
- **Discover** – Ανακάλυψη τοποθεσιών, δραστηριοτήτων και αξιοθέατων
 - `FilterModal.js`: Παράθυρο φίλτρων αναζήτησης
 - `PlaceDetailsModal.js`: Παρουσίαση πληροφοριών τοποθεσίας
 - `PlaceMarker.js`: Marker τοποθεσίας στον χάρτη
 - `RouteManager.js`: Διαχείριση και εμφάνιση διαδρομών
 - `SearchBar.js`: Γραμμή αναζήτησης τοποθεσιών
- **MapScreen** – Εμφάνιση δραστηριοτήτων σε χάρτη
 - `DayTimePicker.js`: Επιλογή ημέρας/ώρας για προσθήκη δραστηριότητας
 - `FilterModal.js`: Φίλτρα για την προβολή χάρτη
 - `PlaceDetails.js`: Λεπτομέρειες σημείου στον χάρτη
- **MyTrips** – Προβολή λίστας αποθηκευμένων ταξιδιών του χρήστη
 - `AddUser.jsx`: Προσθήκη χρήστη σε ταξίδι
 - `StartNewTripCard.jsx`: Κάρτα δημιουργίας νέου ταξιδιού
 - `UserTripCard.jsx`: Κάρτα αποθηκευμένου ταξιδιού
 - `UserTripList.jsx`: Λίστα ταξιδιών του χρήστη
- **TripDetails** – Αναλυτική παρουσίαση προγράμματος ταξιδιού
 - `FlightInfo.jsx`: Πληροφορίες πτήσης
 - `HotelCard.jsx`: Κάρτα ξενοδοχείου
 - `HotelList.jsx`: Λίστα ξενοδοχείων
 - `PlaceCard.jsx`: Κάρτα τοποθεσίας
 - `PlannedTrip.jsx`: Πρόγραμμα δραστηριοτήτων

- **TripPlan** – Προβολή ημερήσιου προγράμματος ταξιδιού
 - `DayTimeline.js`: Χρονοδιάγραμμα δραστηριοτήτων ημέρας
 - `TripCalendar.js`: Ημερολόγιο ταξιδιού
 - `TripFooter.js`: Στοιχεία πλοήγησης και ενεργειών στο κάτω μέρος

Φάκελος assets

Περιέχει τα πολυμέσα της εφαρμογής, οργανωμένα ως εξής:

- **fonts** – Προσαρμοσμένες γραμματοσειρές για την εμφάνιση κειμένου
- **icons** – Εικονίδια εφαρμογής
- **images** – Γραφικά και φωτογραφίες για περιεχόμενο

Φάκελος constants

Περιέχει σταθερές τιμές και ρυθμίσεις που χρησιμοποιούνται σε όλη την εφαρμογή, όπως:

- `Colors.ts` – Παλέτα χρωμάτων για το UI
- `Options.js` – Προκαθορισμένες επιλογές ταξιδιωτών, προϋπολογισμού και πρότυπο για δημιουργία ταξιδιωτικού πλάνου μέσω AI
- `placeTypes.js` – Κατηγορίες τοποθεσιών (π.χ. εστιατόρια, μουσεία)

Φάκελος context

Περιέχει τα React Contexts για την κεντρική (global) διαχείριση της κατάστασης της εφαρμογής, όπως στοιχεία χρήστη και δεδομένα ταξιδιών. Συγκεκριμένα:

- `CreateTripContext.js` – Διαχειρίζεται δεδομένα που σχετίζονται με τη δημιουργία ταξιδιού και τα καθιστά διαθέσιμα σε όλα τα components που τα χρειάζονται
- `AuthContext.js` – Διαχειρίζεται την ταυτοποίηση και τα στοιχεία του χρήστη, παρέχοντας λειτουργίες όπως φόρτωση συνεδρίας, σύνδεση/εγγραφή, αποσύνδεση, ενημέρωση προφίλ, διαγραφή λογαριασμού και επαναφορά κωδικού, μέσω επικοινωνίας με το backend

Παράδειγμα από το AuthContext:

```
const updateUser = async (name, email) => {
  try {
    const res = await api.put('/auth/update-profile', { name, email }, {
      headers: { Authorization: `Bearer ${token}` },
    });

    if (res.data.success) {
      setUser(res.data.user);
      return { success: true };
    }

    return { success: false, message: res.data.message };
  } catch (err) {
    return { success: false, message: err.response?.data?.msg || 'Failed to update profile.' };
  }
};
```

Σχήμα 4.18: Παράδειγμα κώδικα για την ενημέρωση προφίλ χρήστη

Φάκελος services

Περιλαμβάνει τις συναρτήσεις και τα modules που διαχειρίζονται την επικοινωνία της εφαρμογής με εξωτερικές υπηρεσίες και APIs, όπως ο backend server και οι υπηρεσίες Google Maps/Places. Αναλυτικά:

- `api.js` – Δημιουργεί και ρυθμίζει ένα axios instance με προκαθορισμένο baseUrl για κλήσεις προς το backend
- `GooglePlaceApi.jsx` – Περιλαμβάνει εξειδικευμένες συναρτήσεις για την αλληλεπίδραση με τις υπηρεσίες Google Maps και Places, όπως:
 - **GetPhotoRef:** Αναζήτηση φωτογραφιών τοποθεσίας με βάση το όνομα
 - **fetchNearbyPlaces:** Εύρεση κοντινών σημείων ενδιαφέροντος βάσει φίλτρων (τύπος, απόσταση, τιμή, διαθεσιμότητα)
 - **fetchNextPage:** Ανάκτηση αποτελεσμάτων επόμενης σελίδας για αναζητήσεις
 - **fetchRoute:** Υπολογισμός διαδρομής μεταξύ δύο σημείων για διάφορα μέσα μετακίνησης
 - **geocodeAddress:** Μετατροπή διεύθυνσης σε γεωγραφικές συντεταγμένες
 - **fetchPlacePhotos:** Λήψη φωτογραφιών μιας τοποθεσίας
 - **fetchPlaceDetails:** Ανάκτηση αναλυτικών πληροφοριών για μια τοποθεσία

Παράδειγμα από το GooglePlaceApi:

```
export const fetchPlaceDetails = async (placeId, apiKey) => {  
  try {  
    const res = await axios.get(  
      'https://maps.googleapis.com/maps/api/place/details/json',  
      {  
        params: {  
          place_id: placeId,  
          key: apiKey,  
        },  
      },  
    );  
    return res.data.result;  
  } catch (err) {  
    console.error('Place details error:', err);  
    return null;  
  }  
};
```

Σχήμα 4.19: Παράδειγμα κώδικα για την ανάκτηση πληροφοριών τοποθεσίας

Αρχείο .env

Αρχείο περιβάλλοντος που αποθηκεύει ευαίσθητες μεταβλητές (π.χ. API keys, URIs), ώστε να μην εκτίθενται απευθείας στον κώδικα. Στην εφαρμογή χρησιμοποιείται για τον ορισμό του `GOOGLE_MAP_KEY`, που απαιτείται για την πρόσβαση στις υπηρεσίες Google Maps.

Αρχείο app.json

Κύριο αρχείο ρυθμίσεων του Expo, στο οποίο καθορίζονται στοιχεία όπως το όνομα της εφαρμογής, τα εικονίδια και τα απαραίτητα native permissions.

Αρχείο package.json

Περιέχει τις εξαρτήσεις της εφαρμογής και τις αντίστοιχες εκδόσεις τους, εξασφαλίζοντας συνεπή εγκατάσταση των πακέτων.

Κεφάλαιο 5

Λειτουργίες & Χαρακτηριστικά

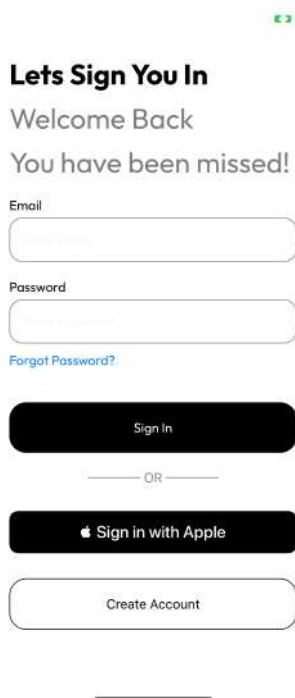
Το κεφάλαιο αυτό παρουσιάζει τις βασικές λειτουργίες της εφαρμογής και τα χαρακτηριστικά που προσφέρουν στον χρήστη. Σε κάθε ενότητα περιγράφεται ο τρόπος χρήσης, τα κύρια στοιχεία της διεπαφής και οι διαθέσιμες επιλογές. Με τη χρήση εικόνων και παραδειγμάτων γίνεται πιο κατανοητή η λειτουργία της εφαρμογής.

5.1 Σύνδεση (Sign In)

Η λειτουργία **Sign In** αποτελεί τον βασικό μηχανισμό αυθεντικοποίησης της εφαρμογής, επιτρέποντας στον χρήστη να αποκτήσει πρόσβαση στο προφίλ του και στις ταξιδιωτικές του πληροφορίες. Η διαδικασία σύνδεσης υλοποιείται μέσω του backend, το οποίο ελέγχει τα διαπιστευτήρια και επιστρέφει κατάλληλες απαντήσεις.

Κύρια Χαρακτηριστικά

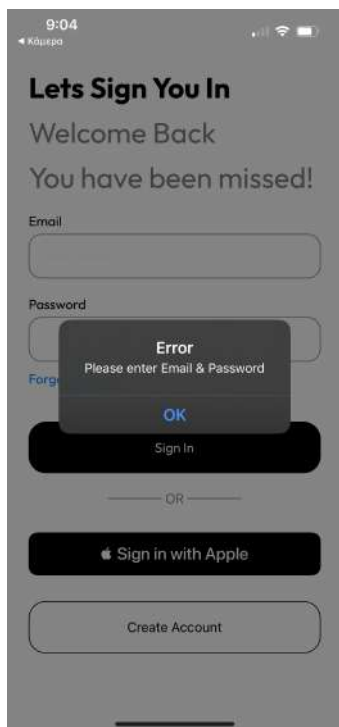
- Πεδία εισαγωγής **Email** και **Password**
- Κουμπί **Sign In** για εκκίνηση της σύνδεσης
- **Sign in with Apple** ως εναλλακτικός τρόπος σύνδεσης
- Επιλογή **Create Account** για νέους χρήστες
- Σύνδεσμο **Forgot Password** για επαναφορά κωδικού



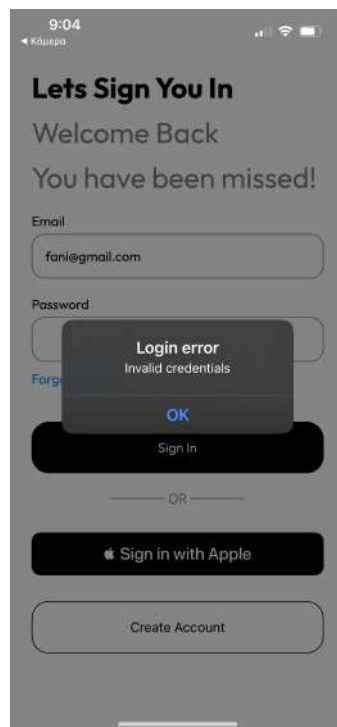
The main sign-in screen features a green status bar at the top. The text 'Lets Sign You In' is in bold, followed by 'Welcome Back' and 'You have been missed!'. There are two input fields: 'Email' and 'Password', both with placeholder text. A blue link 'Forgot Password?' is below the password field. A black 'Sign In' button is centered, followed by an 'OR' separator. Below this is a black 'Sign in with Apple' button with the Apple logo, and a white 'Create Account' button at the bottom.

Σχήμα 5.1: Κεντρική οθόνη σύνδεσης (Sign In)

Η διαδικασία συνοδεύεται από κατανοητά μηνύματα για κάθε περίπτωση. Ενδεικτικά παραδείγματα παρουσιάζονται στην εικόνα 5.2 και στην εικόνα 5.3.

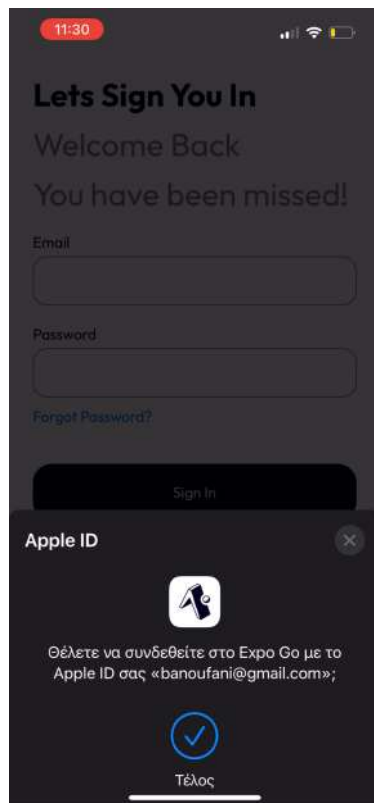


(α) Κενά πεδία

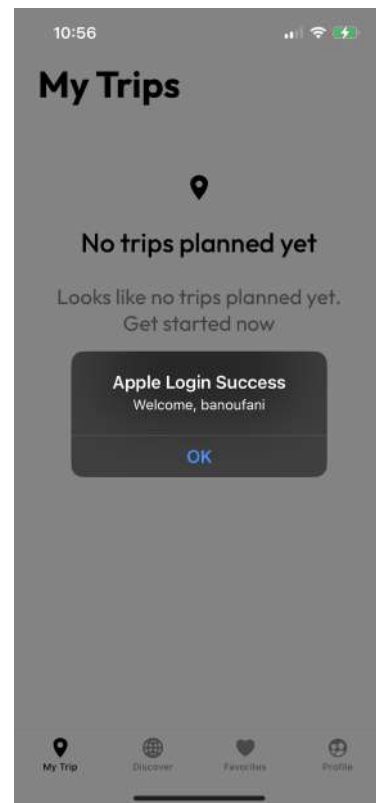


(β) Λανθασμένα στοιχεία

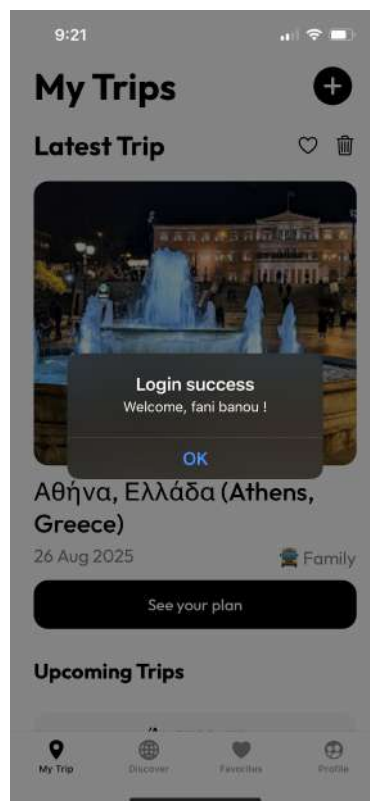
Σχήμα 5.2: Παραδείγματα ελέγχων κατά τη σύνδεση (Sign In)



(α) Σύνδεση με Apple ID



(β) Επιτυχής σύνδεση (Apple ID)



(γ) Επιτυχής σύνδεση

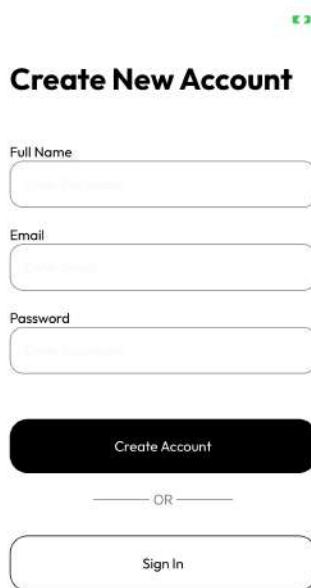
Σχήμα 5.3: Παραδείγματα ενεργειών και μηνυμάτων κατά τη σύνδεση (Sign In)

5.2 Εγγραφή (Sign Up)

Η λειτουργία **Sign Up** επιτρέπει τη δημιουργία νέου λογαριασμού χρήστη. Ο χρήστης εισάγει τα απαραίτητα στοιχεία (όνομα, email, κωδικό) και αποκτά πρόσβαση σε εξατομικευμένες λειτουργίες (π.χ. αποθήκευση ταξιδιών).

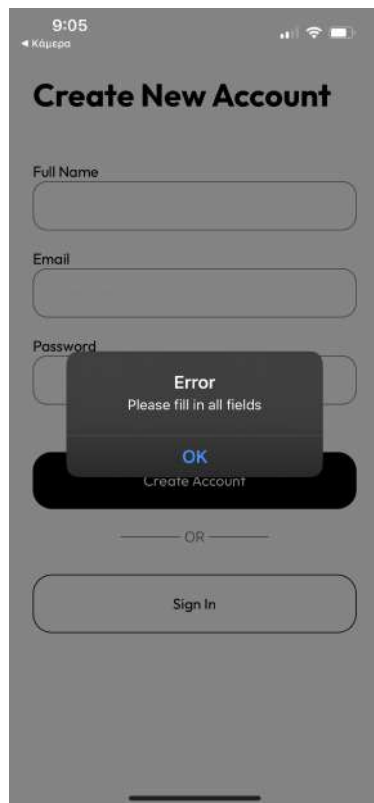
Κύρια Χαρακτηριστικά

- Πεδία για την εισαγωγή **Full Name**, **Email** και **Password**
- Κουμπί **Create Account** για την ολοκλήρωση της εγγραφής
- Επιλογή **Sign In** για χρήστες που διαθέτουν ήδη λογαριασμό

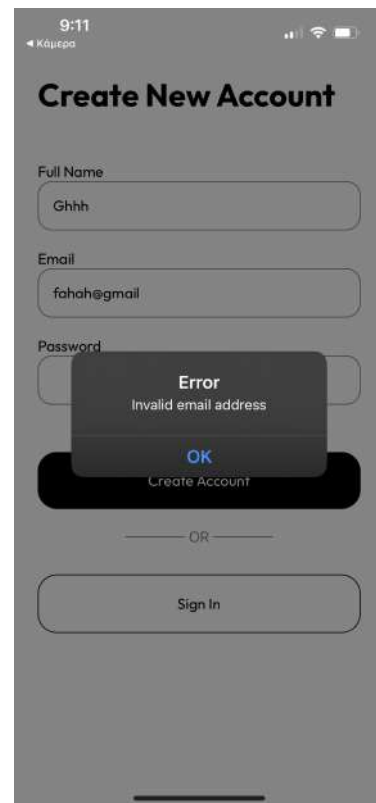


Σχήμα 5.4: Κεντρική οθόνη δημιουργίας λογαριασμού (Sign Up)

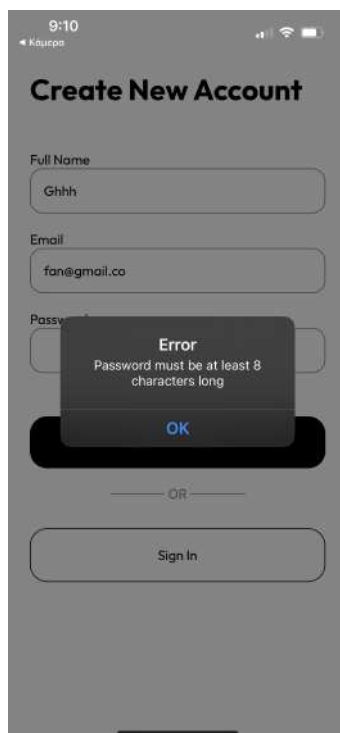
Η εφαρμογή ενημερώνει τον χρήστη με μηνύματα επιβεβαίωσης και σφάλματος κατά τη διαδικασία εγγραφής, όπως φαίνεται συγκεντρωτικά στην εικόνα 5.5.



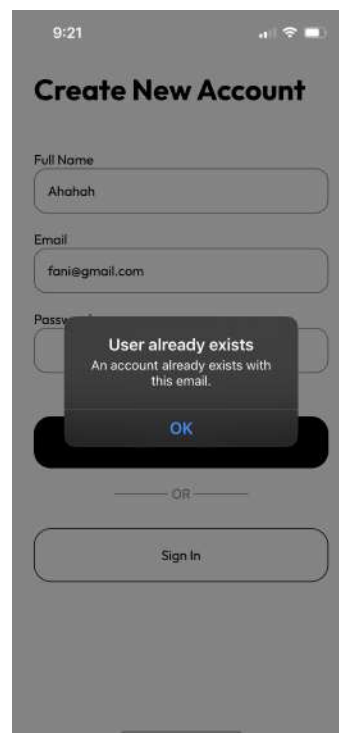
(α) Κενά πεδία



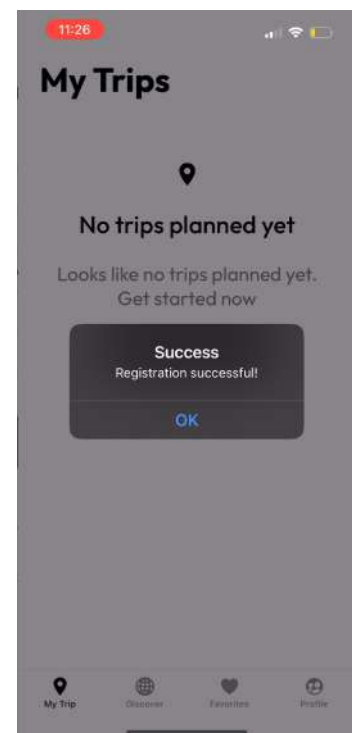
(β) Μη έγκυρο email



(γ) Αδύναμος κωδικός



(δ) Υπάρχων λογαριασμός



(ε) Επιτυχής εγγραφή

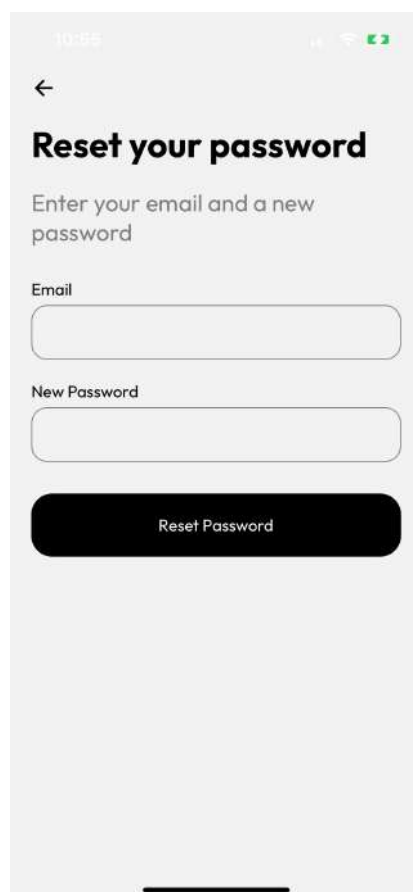
Σχήμα 5.5: Παραδείγματα ελέγχων και μηνυμάτων κατά την εγγραφή (Sign Up)

5.3 Επαναφορά κωδικού (Reset Password)

Η λειτουργία **Reset Password** επιτρέπει στον χρήστη να ορίσει νέο κωδικό σε περίπτωση απώλειας ή για λόγους ασφαλείας. Η διαδικασία γίνεται με εισαγωγή **Email** και **New Password** και έλεγχο από το backend.

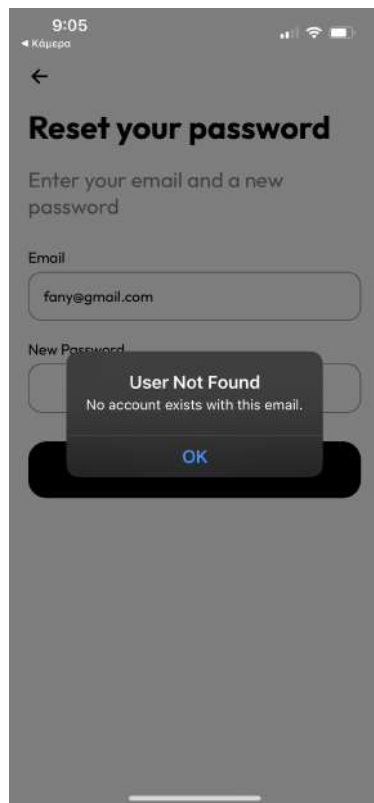
Κύρια Χαρακτηριστικά

- Πεδίο **Email** για αναγνώριση λογαριασμού
- Πεδίο **New Password**, όπου καταχωρείται ο νέος κωδικός
- Κουμπί **Reset Password** για ολοκλήρωση της διαδικασίας

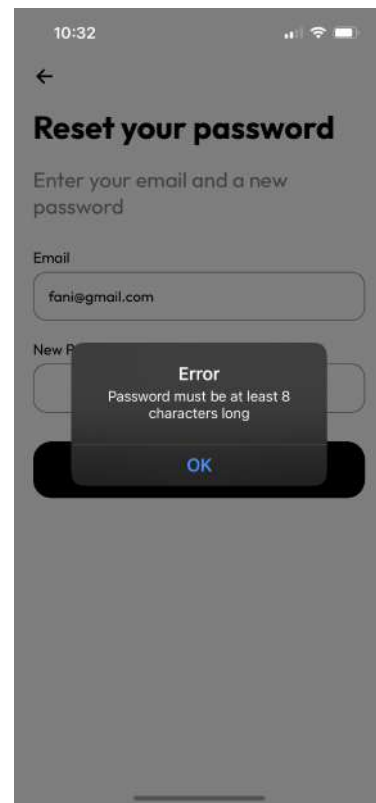


Σχήμα 5.6: Κεντρική οθόνη επαναφοράς κωδικού (Reset Password)

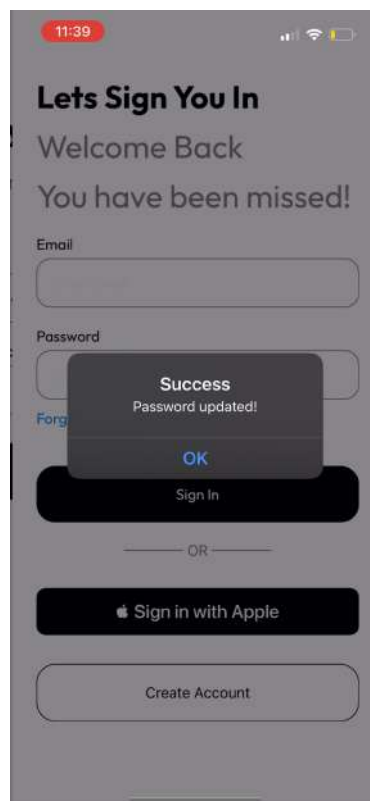
Κατά την επαναφορά κωδικού, παρέχονται σαφή μηνύματα επιτυχίας και αποτυχίας. Χαρακτηριστικά παραδείγματα φαίνονται στην εικόνα 5.7.



(α) Μη υπαρκτός χρήστης



(β) Ανεπαρκής ισχύς νέου κωδικού



(γ) Επιτυχής αλλαγή κωδικού

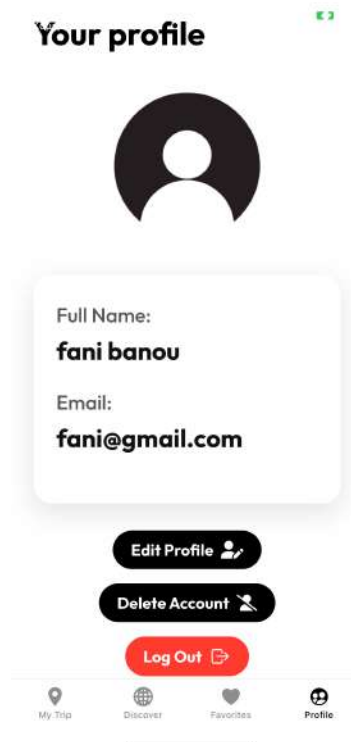
Σχήμα 5.7: Παραδείγματα ελέγχων και μηνυμάτων στην επαναφορά κωδικού (Reset Password)

5.4 Προφίλ Χρήστη (Profile)

Η λειτουργία **Profile** επιτρέπει στον χρήστη να διαχειρίζεται τα προσωπικά του στοιχεία και τις βασικές ρυθμίσεις λογαριασμού. Μέσα από την οθόνη του προφίλ, ο χρήστης μπορεί να δει τα στοιχεία του, να τα επεξεργαστεί αλλά και να πραγματοποιήσει ενέργειες, όπως διαγραφή λογαριασμού ή αποσύνδεση.

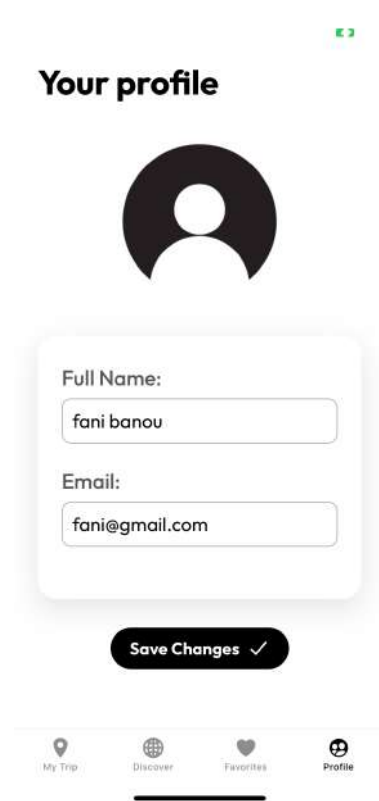
Κύρια Χαρακτηριστικά

- Προβολή **Full Name** και **Email**
- Δυνατότητα επεξεργασίας στοιχείων μέσω του **Edit Profile**
- Μόνιμη διαγραφή λογαριασμού με το **Delete Account**
- Αποσύνδεση από την εφαρμογή με το **Log Out**

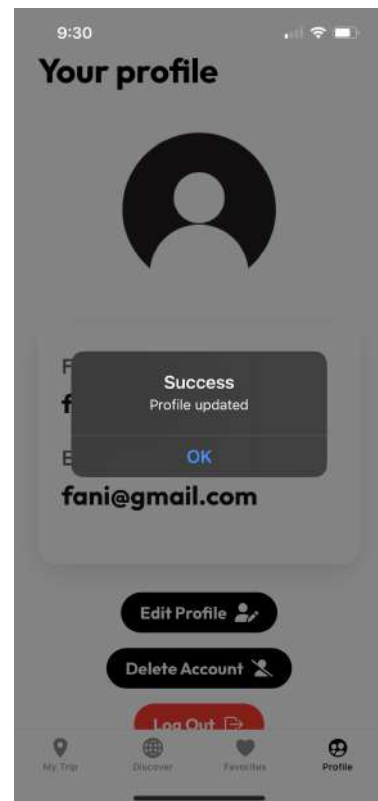


Σχήμα 5.8: Κεντρική οθόνη προφίλ χρήστη (Profile)

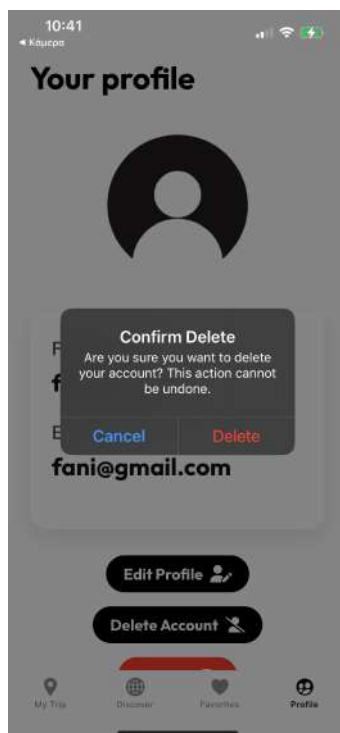
Η εφαρμογή παρέχει μηνύματα επιβεβαίωσης για όλες τις βασικές ενέργειες του προφίλ, όπως η επεξεργασία στοιχείων, η διαγραφή και η αποσύνδεση. Οι παραπάνω λειτουργίες φαίνονται συγκεντρωτικά στην εικόνα 5.9.



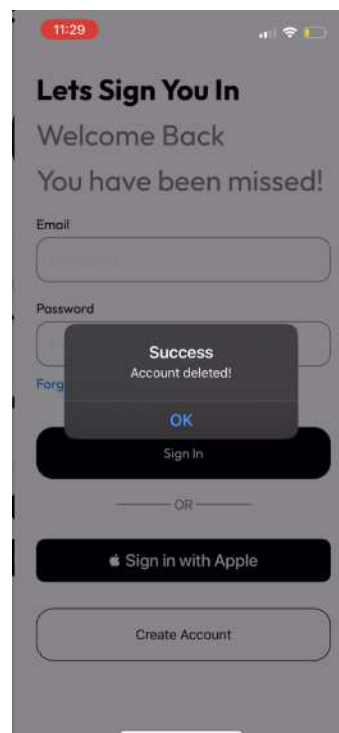
(α) Επεξεργασία στοιχείων



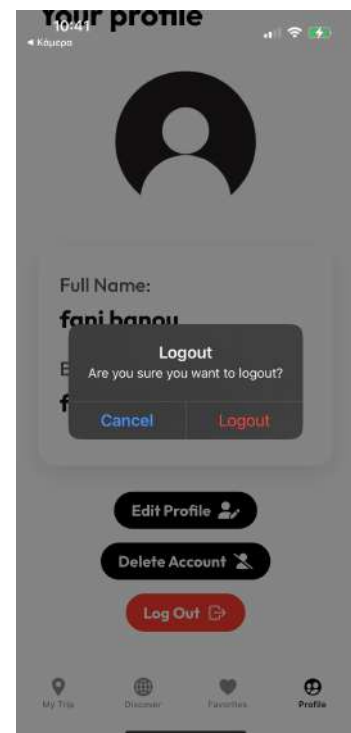
(β) Επιβεβαίωση επεξεργασίας



(γ) Επιβεβαίωση διαγραφής



(δ) Επιβεβαίωση διαγραφής



(δ) Επιβεβαίωση αποσύνδεσης

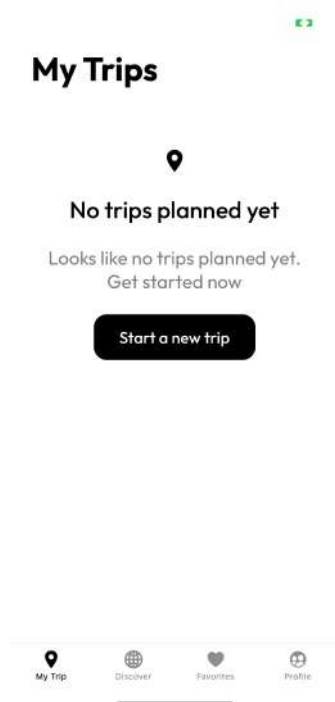
Σχήμα 5.9: Παραδείγματα ενεργειών και μηνυμάτων από το προφίλ χρήστη (Profile)

5.5 Προβολή ταξιδιών (My Trips)

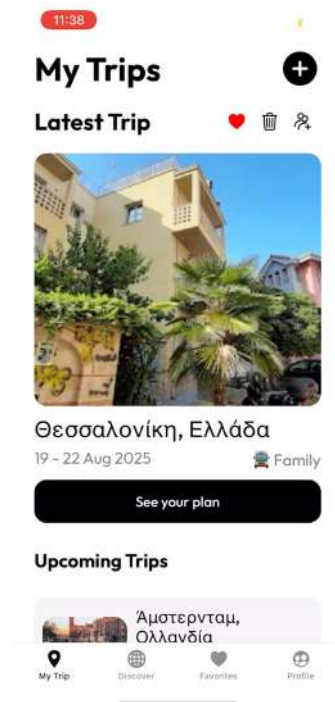
Η ενότητα **My Trips** συγκεντρώνει όλα τα ταξίδια του χρήστη και τα οργανώνει σε σαφείς κατηγορίες, επιτρέποντας γρήγορη πλοήγηση και βασικές ενέργειες (προβολή πλάνου, αγαπημένα, διαγραφή, προσθήκη μελών).

Κύρια Χαρακτηριστικά

- **Κατηγορίες προβολής:** *Latest Trip*, *Upcoming Trips*, *Past Trips* και ταξίδια στα οποία ο χρήστης είναι μέλος
- **Κάρτες ταξιδιών** με τίτλο προορισμού, ημερομηνία, τύπο ταξιδιού και δράσεις
- **Ενέργειες ανά ταξίδι:**
 - *See your plan*: μετάβαση στις λεπτομέρειες/χρονοδιάγραμμα του ταξιδιού
 - *Favorite*: προσθήκη/αφαίρεση από τα αγαπημένα
 - *Delete*: διαγραφή ταξιδιού
 - *Add members*: προσθήκη μελών στο ταξίδι

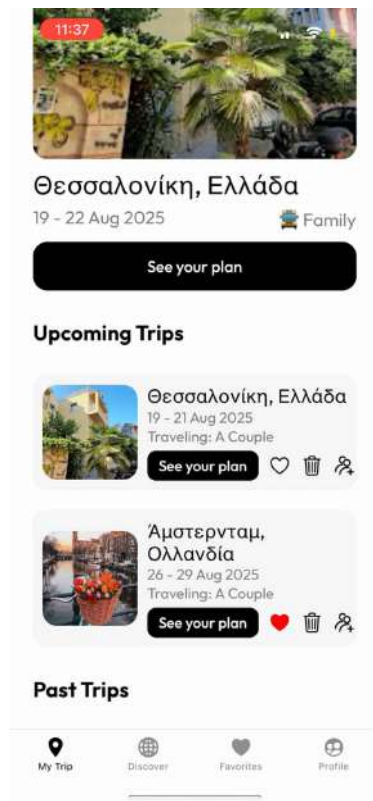


(α) Empty state

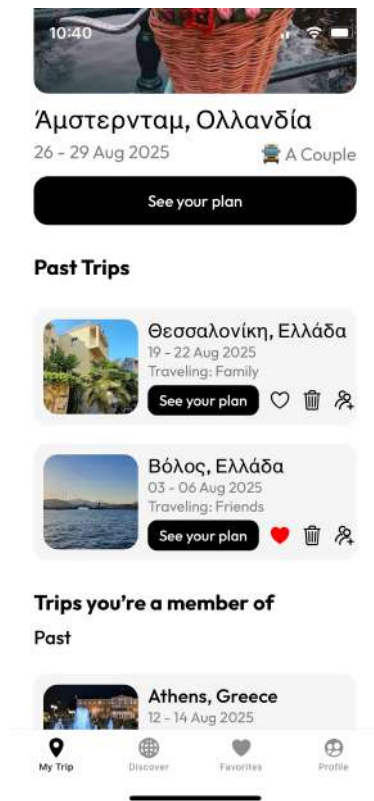


(β) Latest Trip

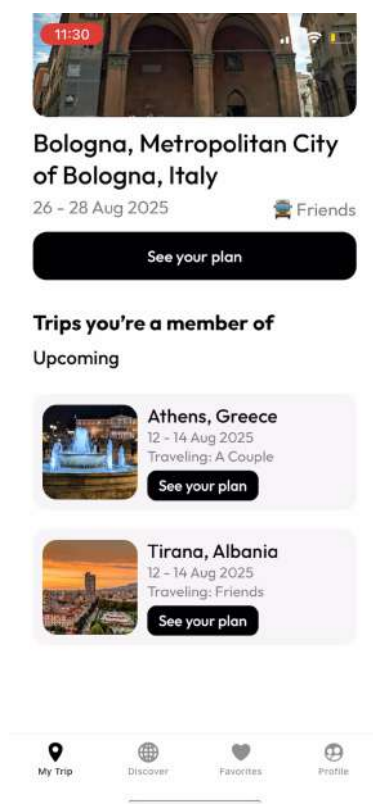
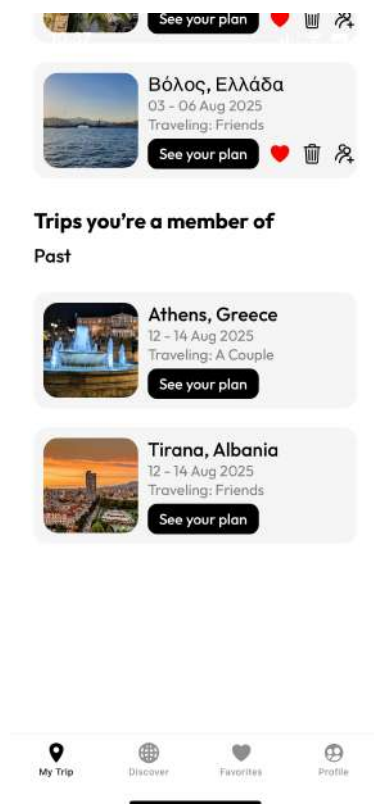
Σχήμα 5.10: Κεντρική οθόνη ταξιδιών *My Trips*

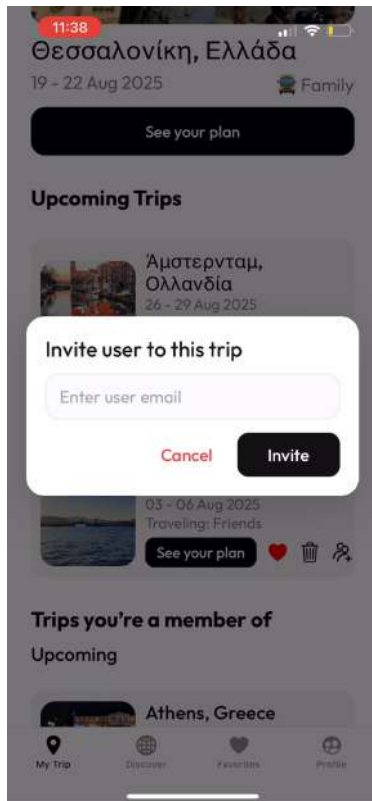


(α) Upcoming Trips

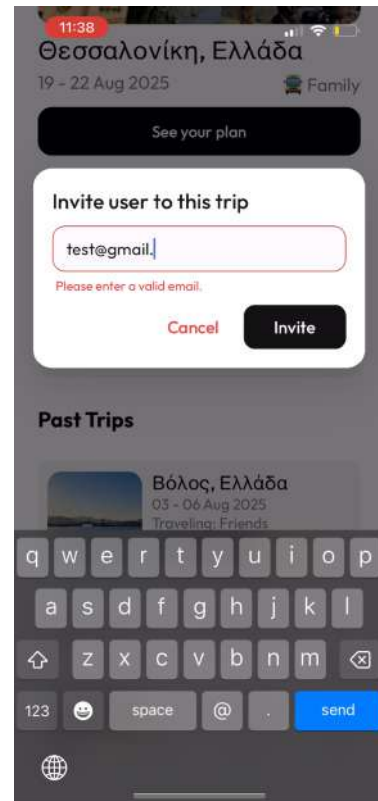


(β) Past Trips

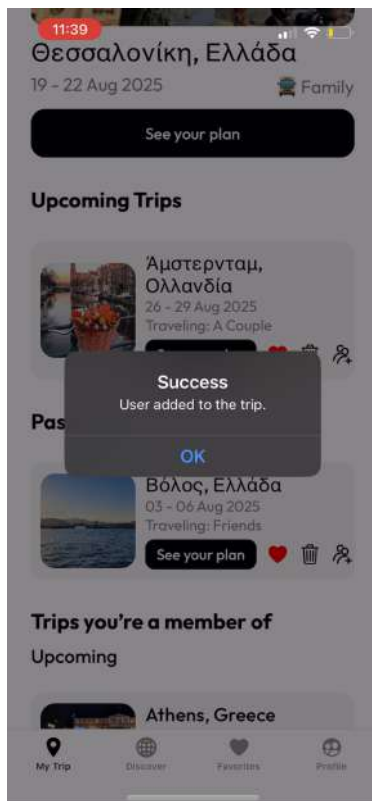
(γ) Ταξίδια που ο χρήστης είναι μέλος
(Upcoming Trips)(δ) Ταξίδια που ο χρήστης είναι μέλος
(Past Trips)Σχήμα 5.11: Οθόνη προβολής ταξιδιών *My Trips*



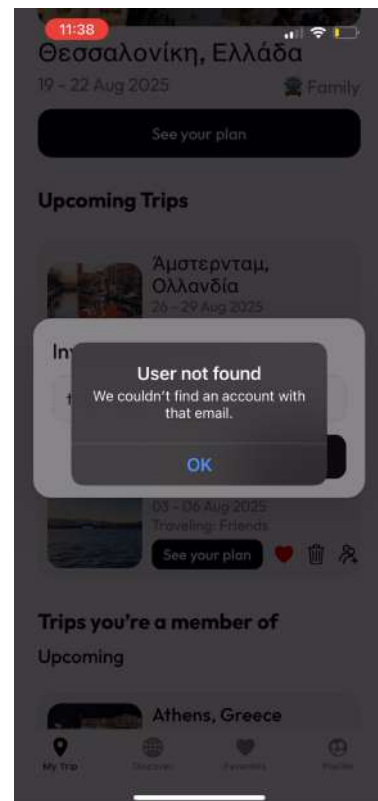
(α) Προσθήκης μέλους



(β) Έλεγχος εγκυρότητας email

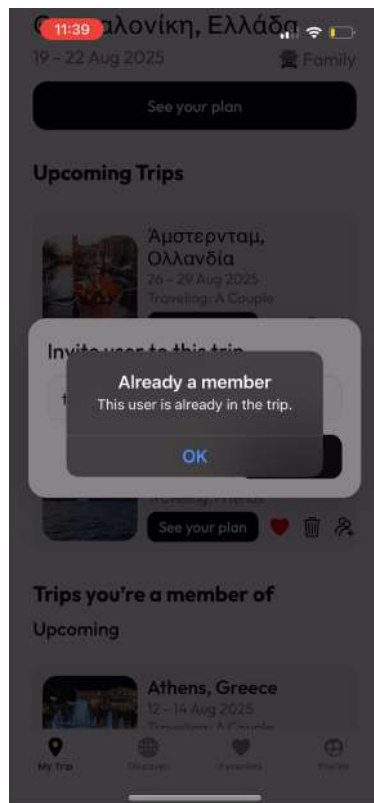


(γ) Επιτυχής προσθήκη μέλους

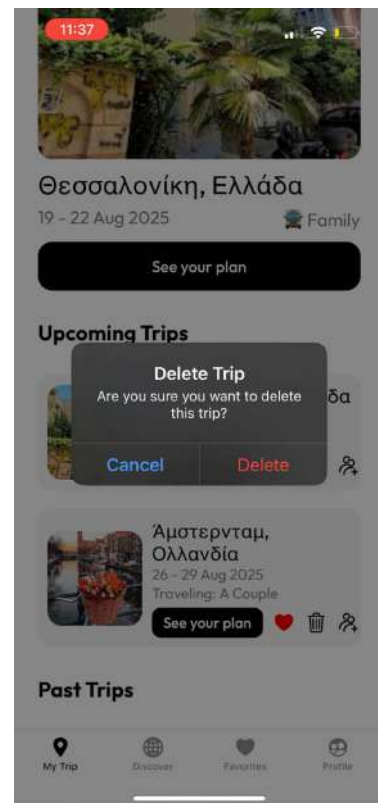


(δ) Μη υπαρκτός χρήστης

Σχήμα 5.12: Προσθήκης μέλους και μηνύματα κατάστασης



(α) Το μέλος υπάρχει ήδη στο ταξίδι



(β) Επιβεβαίωση διαγραφής ταξιδιού

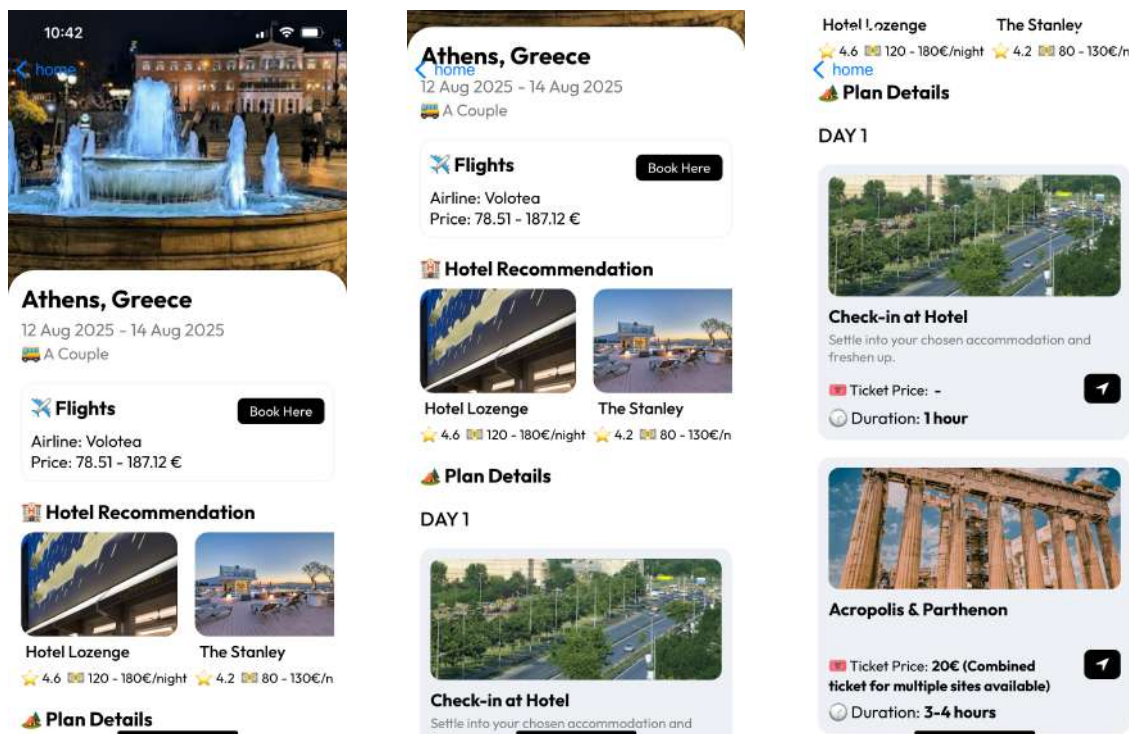
Σχήμα 5.13: Παραδείγματα ελέγχων και μηνυμάτων

5.6 Πλάνο ταξιδιού (Trip Details)

Η σελίδα **Trip Details** παρουσιάζει το αναλυτικό πλάνο που δημιουργείται αυτόματα από το AI, περιλαμβάνοντας προτάσεις για πτήσεις, διαμονή και ημερήσιο πρόγραμμα δραστηριοτήτων με εκτιμήσεις κόστους και διάρκειας. Το περιεχόμενο προσαρμόζεται δυναμικά ανάλογα με τον προορισμό, τις ημερομηνίες και τον τύπο του ταξιδιού, ώστε να ανταποκρίνεται καλύτερα στις ανάγκες του χρήστη.

Κύρια Χαρακτηριστικά

- **Επικεφαλίδα ταξιδιού:** προορισμός, ημερομηνίες, τύπος ταξιδιού
- **Flights card:** συνοπτική ενημέρωση (ενδεικτικές εταιρείες/εύρος τιμών) και κουμπί *Book Here* για κράτηση σε εξωτερική υπηρεσία
- **Hotel Recommendations:** λίστα με ξενοδοχεία, βαθμολογία και εύρος τιμών/διανυκτέρευση
- **Plan Details per day:** κάρτες δραστηριοτήτων με εικόνα, περιγραφή, κόστος εισιτηρίου, διάρκεια και κουμπί *directions* για άνοιγμα χάρτη/πλοήγησης



(α) Επικεφαλίδα & Flights

(β) Προτάσεις ξενοδοχείων

(γ) Plan Details

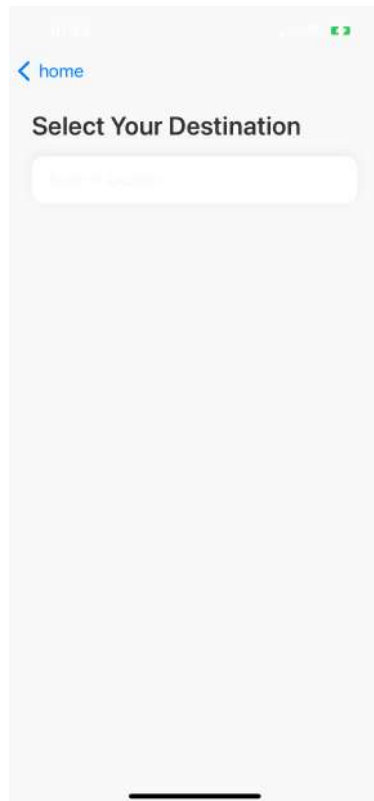
Σχήμα 5.14: Σελίδα *Trip Details* που δημιουργείται από AI

5.7 Δημιουργία ταξιδιού (Create Trip)

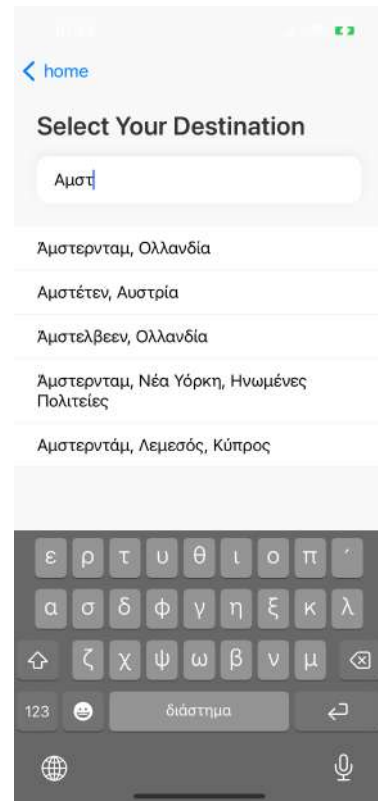
Για τη δημιουργία ενός ταξιδιού, ο χρήστης καθοδηγείται μέσα από πέντε διαδοχικά βήματα: επιλογή **προορισμού**, καθορισμό **τύπου ταξιδιωτών**, ορισμό **ημερομηνιών**, επιλογή **προϋπολογισμού** και τέλος, την **ανακεφαλαίωση** όλων των στοιχείων. Από το τελευταίο βήμα, ο χρήστης μπορεί να προχωρήσει στη δημιουργία του ταξιδιωτικού πλάνου με δύο εναλλακτικούς τρόπους. Με την επιλογή *Build with AI*, το backend, αξιοποιώντας το μοντέλο **Gemini**, παράγει δυναμικά προτάσεις για πτήσεις, ξενοδοχεία και ημερήσιο πλάνο, προσαρμοσμένες στις παραμέτρους που έχει ορίσει ο χρήστης. Αντίθετα, με την επιλογή *Build My Trip*, δίνεται η δυνατότητα στον χρήστη να σχεδιάσει το πλάνο του εξ ολοκλήρου χειροκίνητα, σύμφωνα με τις προσωπικές του προτιμήσεις.

Στάδια Δημιουργίας Ταξιδιού

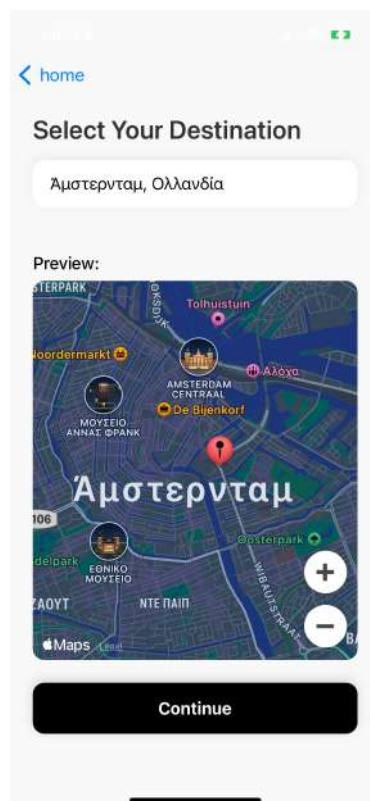
1. **Destination:** Αναζήτηση προορισμού με auto-complete και προεπισκόπηση χάρτη
2. **Who's Travelling:** Επιλογή τύπου ταξιδιωτών (Just Me, Couple, Family, Friends)
3. **Travel Dates:** Επιλογή εύρους ημερομηνιών μέσω ημερολογίου
4. **Budget:** Καθορισμός προϋπολογισμού (Cheap, Moderate, Luxury)
5. **Review:** Ανακεφαλαίωση επιλογών και δημιουργία πλάνου



(α) Destination



(β) Αναζήτηση προτάσεων



(γ) Προεπισκόπηση χάρτη

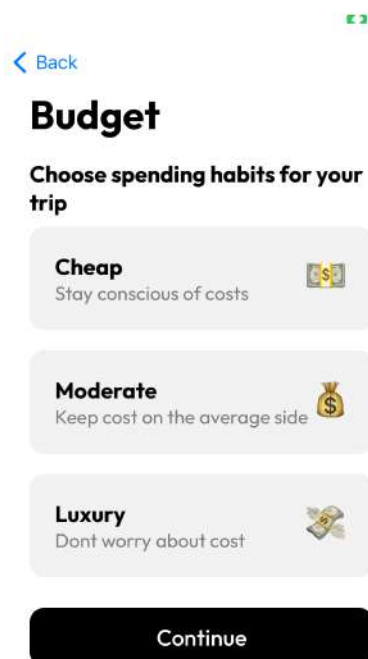
Σχήμα 5.15: Επιλογή προορισμού για την δημιουργία ταξιδιού



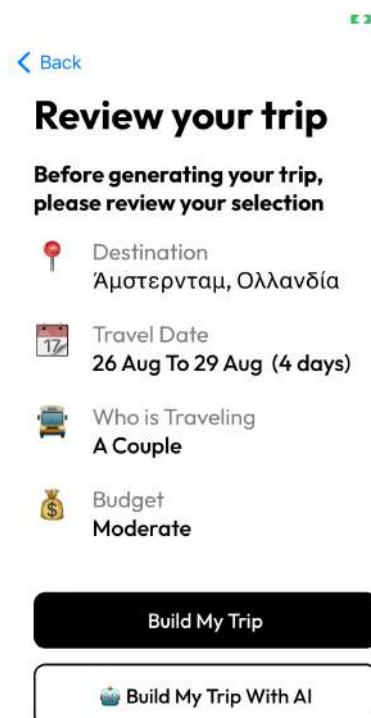
(α) Who's Travelling



(β) Travel Dates



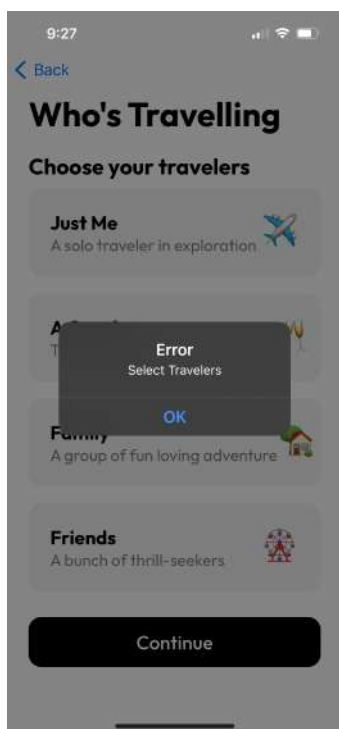
(γ) Budget



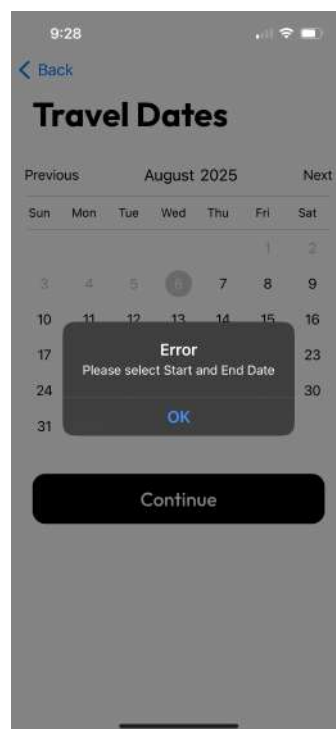
(δ) Review & Build

Σχήμα 5.16: Βημάτα επιλογής στοιχείων ταξιδιού

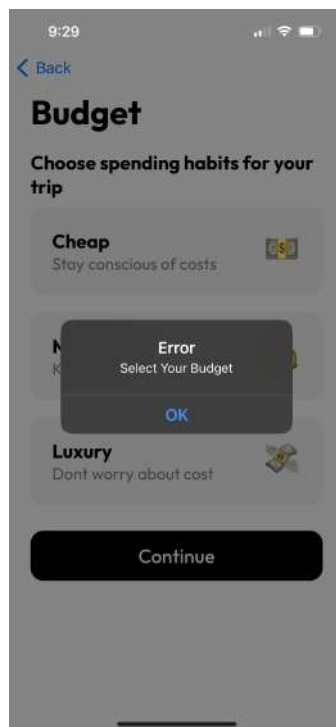
Κάθε στάδιο συνοδεύεται από ελέγχους εγκυρότητας και κατάλληλα μηνύματα σφάλματος, ώστε να αποφεύγονται ελλειπείς ή λανθασμένες καταχωρήσεις.



(α) Select Travelers



(β) Select Start & End Date



(γ) Select Your Budget

Σχήμα 5.17: Ενδεικτικά μηνύματα σφάλματος κατά την επιλογή στοιχείων ταξιδιού

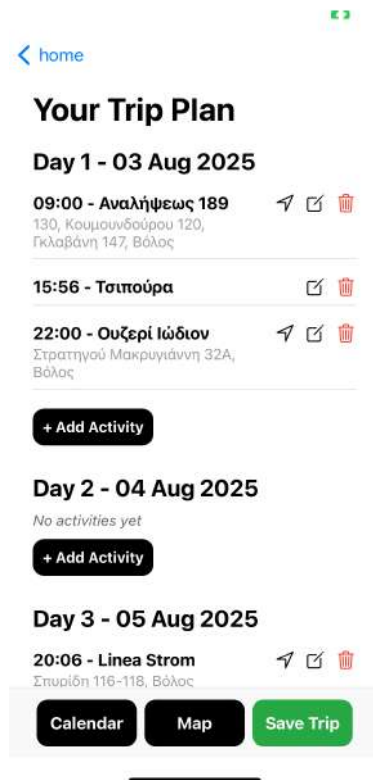
5.8 Εξατομικευμένο ταξίδι (Custom Trip)

Η σελίδα **Custom Trip** επιτρέπει στον χρήστη να δημιουργεί και να διαχειρίζεται εξατομικευμένα ταξίδια, προσθέτοντας δραστηριότητες ανά ημέρα και οργανώνοντάς τις σε χρονολογική ή ημερολογιακή μορφή. Επιπλέον, ενσωματώνει χάρτη με φίλτρα αναζήτησης τοποθεσιών και δυνατότητα προσθήκης στο πλάνο.

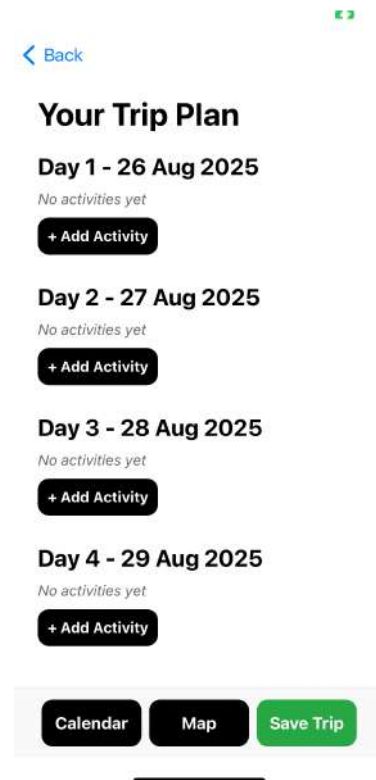
Κύρια Χαρακτηριστικά

- **Timeline view:** λίστα δραστηριοτήτων οργανωμένων ανά ημέρα και ώρα (Εικ. 5.18)
- **Calendar view:** ημερολογιακή απεικόνιση με επισήμανση ημερών που έχουν δραστηριότητες (Εικ. 5.20)
- **Day details:** λίστα δραστηριοτήτων ανά ημέρα (Εικ. 5.24)
- **Route preview:** προεπισκόπηση ενδεικτικής διαδρομής μεταξύ δραστηριοτήτων της ίδιας ημέρας (Εικ. 5.24)
- **Add Activity:** φόρμα δημιουργίας νέας δραστηριότητας με ώρα, τίτλο και προαιρετική περιγραφή (Εικ. 5.22)
- **Edit & Delete:** δυνατότητα επεξεργασίας και διαγραφής δραστηριοτήτων (Εικ. 5.23)
- **Map view:** εμφάνιση σημείων ενδιαφέροντος σε χάρτη (Εικ. 5.25)
- **Place details:** προβολή λεπτομερειών τοποθεσίας (φωτογραφία, αξιολόγηση, διεύθυνση, απόσταση) και άμεση προσθήκη στο πλάνο (Εικ. 5.27)
- **Filters:** φίλτρα αναζήτησης με βάση τον τύπο δραστηριότητας (π.χ. μουσεία, πάρκα), τη βαθμολογία, την απόσταση και την τιμή (Εικ. 5.26)

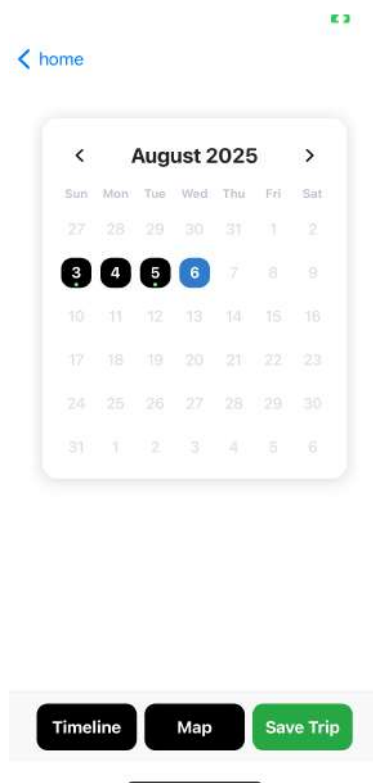
Η δημιουργία δραστηριότητας γίνεται με την επιλογή της ημέρας και προαιρετικά της ώρας, ενώ ο χρήστης εισάγει τον τίτλο και λεπτομέρειες. Κατά την επεξεργασία, τα πεδία εμφανίζονται προσυμπληρωμένα ώστε να επιτρέπουν εύκολες αλλαγές. Η εφαρμογή παρέχει επίσης τη δυνατότητα εισαγωγής δραστηριοτήτων από τον χάρτη.



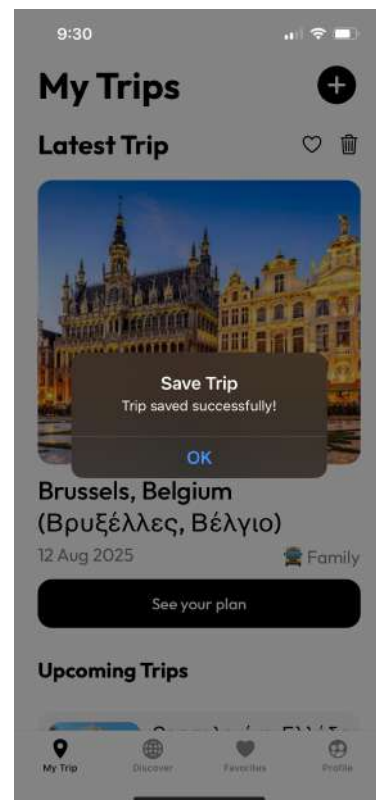
Σχήμα 5.18: Προβολή ταξιδιού με χρονο-διάγραμμα δραστηριοτήτων



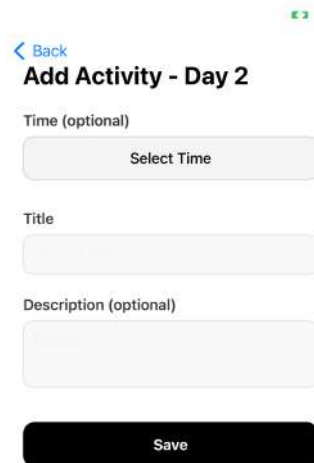
Σχήμα 5.19: Προβολή ταξιδιού χωρίς δραστηριότητες



Σχήμα 5.20: Εναλλακτική προβολή δραστηριοτήτων σε ημερολόγιο



Σχήμα 5.21: Επιβεβαίωση αποθήκευσης ταξιδιού



< Back

Add Activity - Day 2

Time (optional)

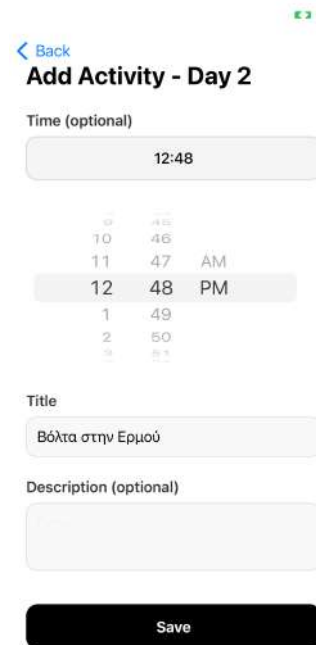
Select Time

Title

Description (optional)

Save

(α) Προσθήκη δραστηριότητας



< Back

Add Activity - Day 2

Time (optional)

12:48

10 46
11 47 AM
12 48 PM
1 49
2 50
3 51

Title

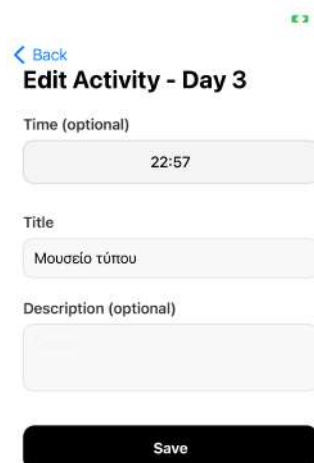
Βόλτα στην Ερμού

Description (optional)

Save

(β) Προσθήκη δραστηριότητας με επιλεγμένη ώρα

Σχήμα 5.22: Παράδειγμα προσθήκης δραστηριότητας σε συγκεκριμένη ημέρα



< Back

Edit Activity - Day 3

Time (optional)

22:57

Title

Μουσείο τύπου

Description (optional)

Save

(α) Επεξεργασία δραστηριότητας



< Back

Edit Activity - Day 3

Time (optional)

10:57

7 55
8 56
9 57
10 57 AM
11 58 PM
12 59
1 00

Title

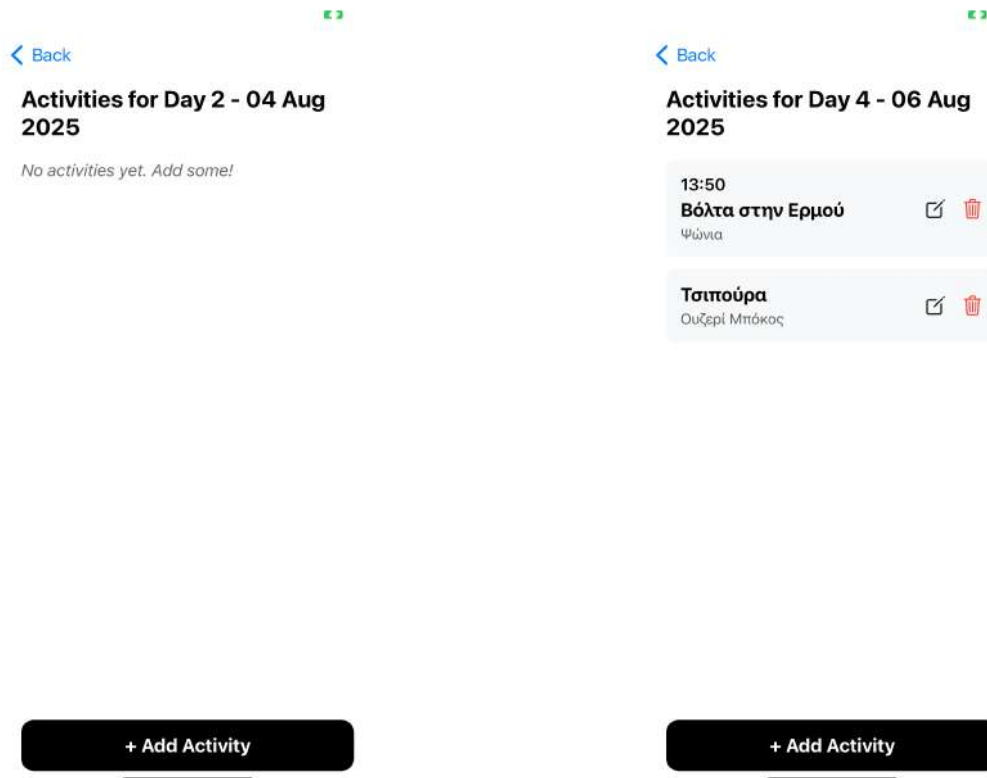
Μουσείο τύπου

Description (optional)

Save

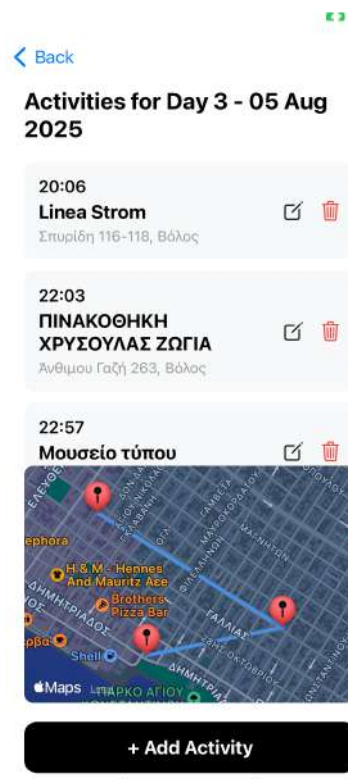
(β) Επεξεργασία δραστηριότητας με επιλεγμένη ώρα

Σχήμα 5.23: Παράδειγμα επεξεργασίας υπάρχουσας δραστηριότητας



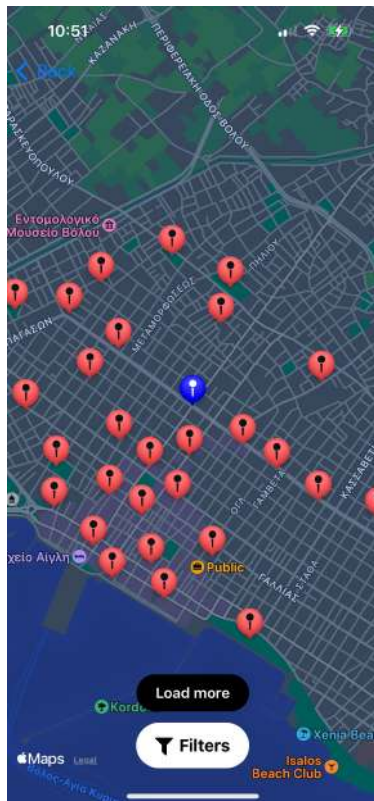
(α) Ημέρα χωρίς δραστηριότητες

(β) Ημέρα με δραστηριότητες

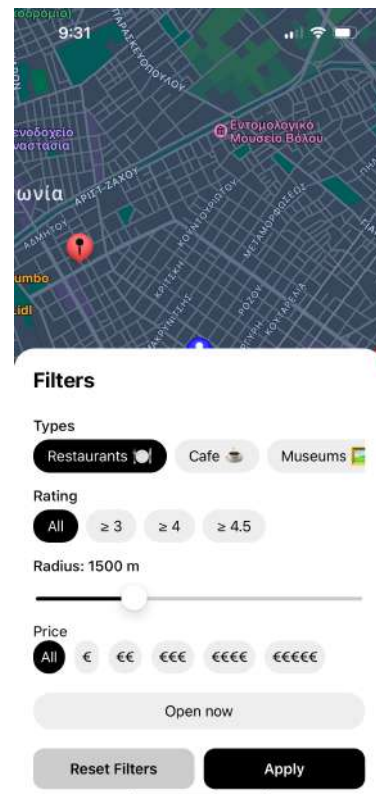


(γ) Προεπισκόπηση διαδρομής μεταξύ δραστηριοτήτων

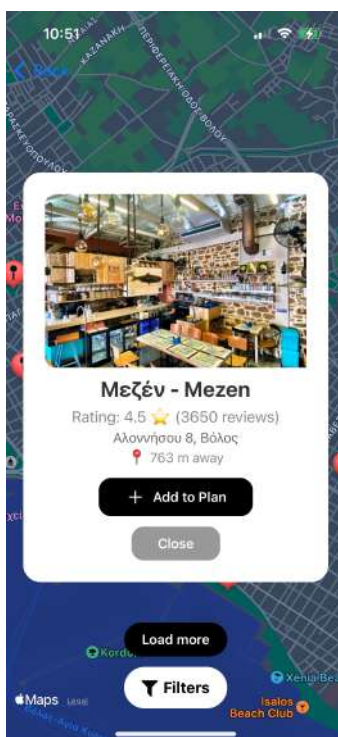
Σχήμα 5.24: Παράδειγμα προβολής δραστηριοτήτων ανά ημέρα



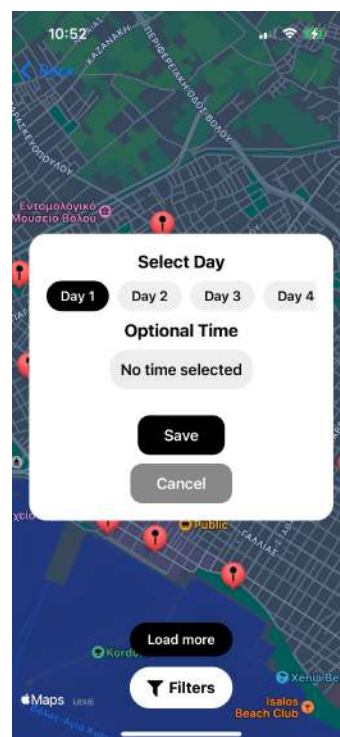
Σχήμα 5.25: Προβολή τοποθεσιών σε χάρτη



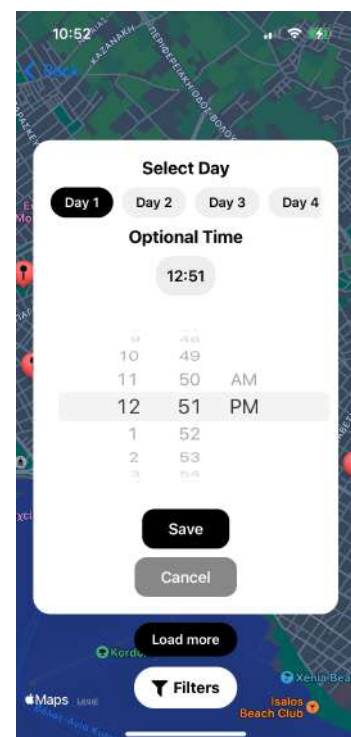
Σχήμα 5.26: Φίλτρα αναζήτησης δραστηριοτήτων



(α) Κάρτα λεπτομερειών τοποθεσίας



(β) Προσθήκη δραστηριότητας στο πλάνο



(γ) Προσθήκη δραστηριότητας με επιλογή ώρας

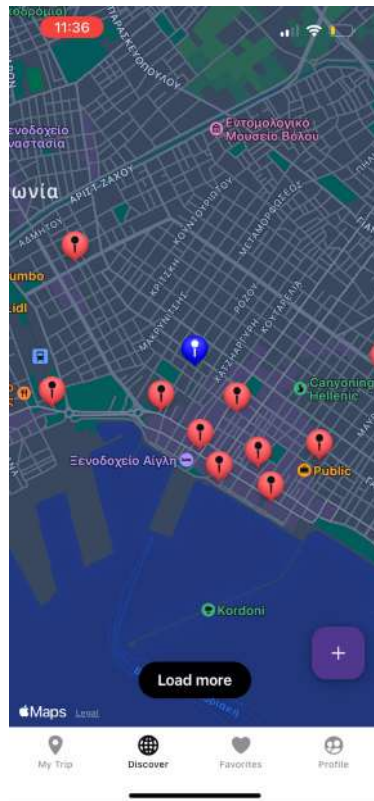
Σχήμα 5.27: Προβολή δραστηριότητας με δυνατότητα προσθήκης στο πλάνο

5.9 Σελίδα ανακάλυψης (Discover)

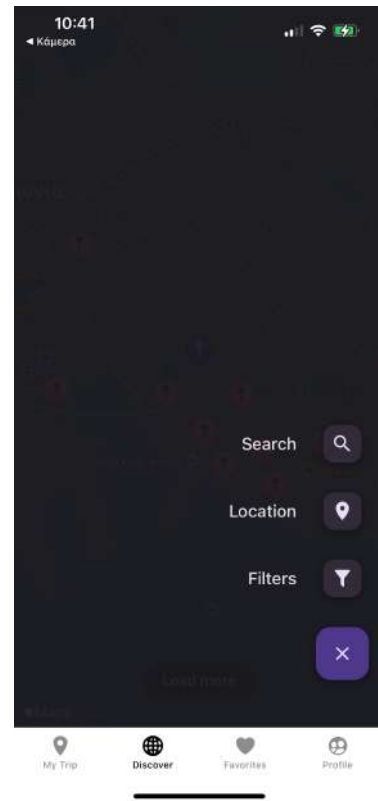
Η σελίδα **Discover** αποτελεί το κεντρικό σημείο για την ανακάλυψη νέων τοποθεσιών και δραστηριοτήτων. Μέσα από έναν διαδραστικό χάρτη με δυνατότητες φίλτρων, αναζήτησης και οδηγίες πλοήγησης, ο χρήστης μπορεί εύκολα να εξερευνήσει διάφορα μέρη για να τα ενσωματώσει μελλοντικά στο ταξιδιωτικό του πρόγραμμα.

Κύρια Χαρακτηριστικά

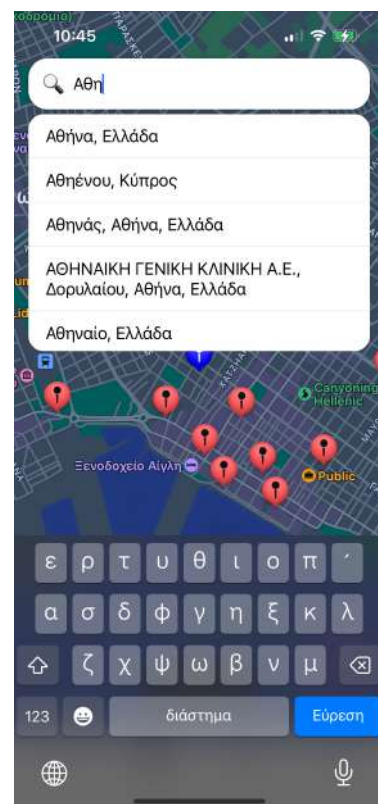
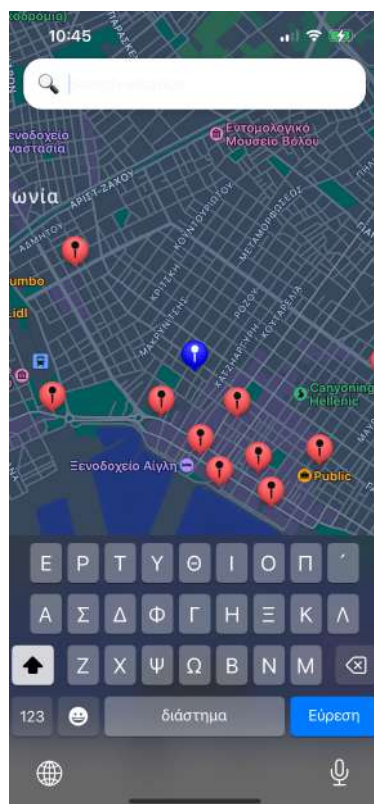
- **Χάρτης με markers:** εμφάνιση σημείων ενδιαφέροντος (Εικ. 5.28)
- **Search:** αναζήτηση προορισμών με προτάσεις autocomplete (Εικ. 5.30)
- **Filters:** φιλτράρισμα αποτελεσμάτων με βάση τύπο (μουσεία, πάρκα, αξιοθέατα), αξιολόγηση, απόσταση και τιμή (Εικ. 5.31)
- **Location button:** εύρεση της τρέχουσας θέσης του χρήστη (Εικ. 5.29)
- **Κάρτα τοποθεσίας:** εμφάνιση λεπτομερειών (όνομα, διεύθυνση, φωτογραφία, κριτικές) και άμεσες ενέργειες όπως αποθήκευση στα αγαπημένα ή πλοήγηση με επιλογή τρόπου μετακίνησης (οδήγηση, περπάτημα, μέσα μεταφοράς) (Εικ. 5.32)
- **Navigation:** εμφάνιση διαδρομής στον χάρτη με τον επιλεγμένο τρόπο μετακίνησης (Εικ. 5.33, Εικ. 5.35, Εικ. 5.37)
- **Route Instructions:** αναλυτικές οδηγίες διαδρομής βήμα προς βήμα, ανάλογα με τον τρόπο μετακίνησης (Εικ. 5.34, Εικ. 5.36, Εικ. 5.38)



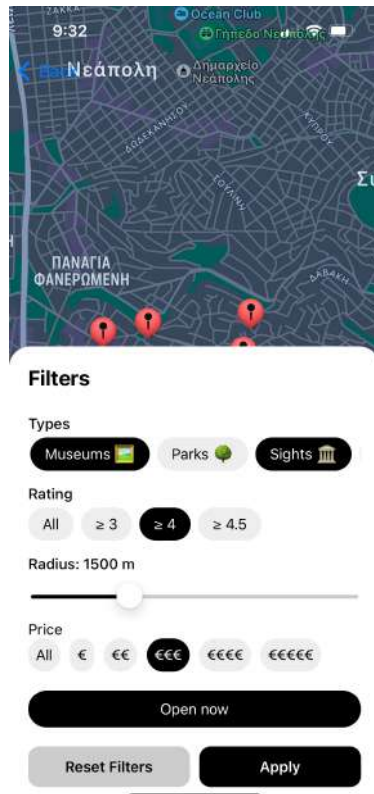
Σχήμα 5.28: Προβολή τοποθεσιών σε χάρτη



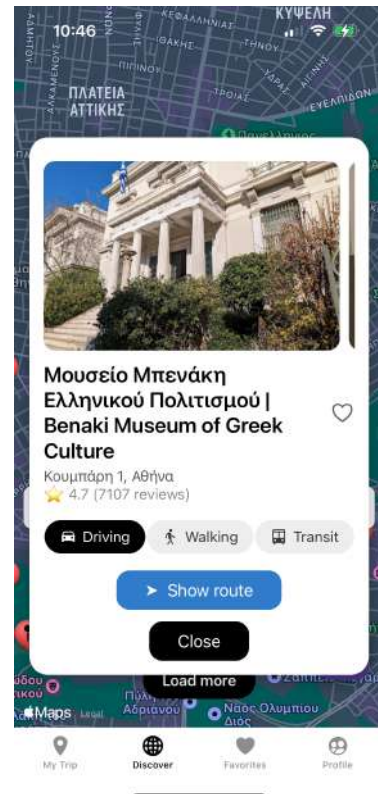
Σχήμα 5.29: Μενού ενεργειών (Search / Location / Filters)



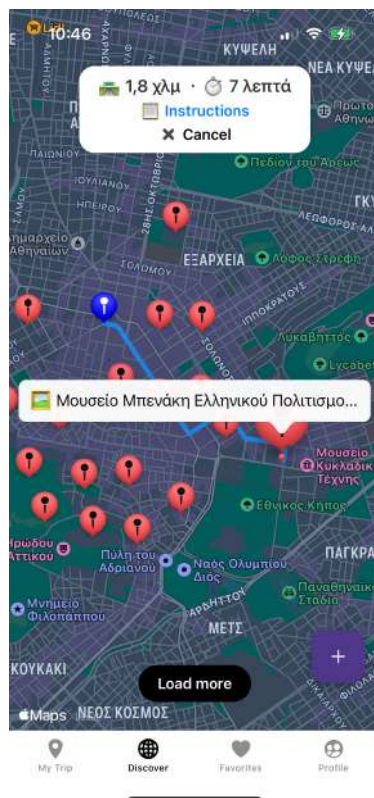
Σχήμα 5.30: Αναζήτηση τοποθεσίας



Σχήμα 5.31: Φίλτρα τοποθεσιών



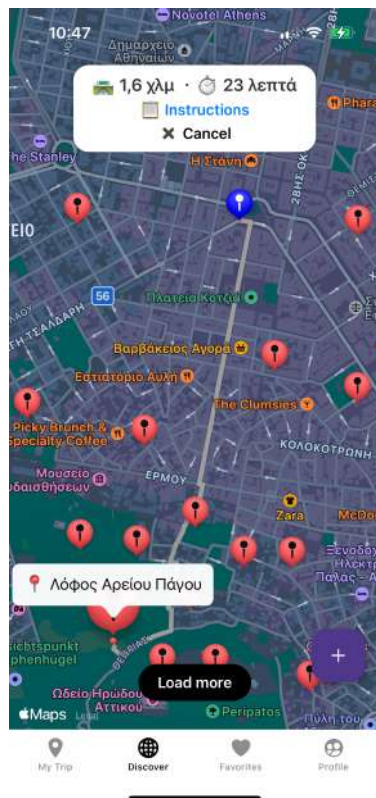
Σχήμα 5.32: Κάρτα τοποθεσίας με λεπτομέρειες



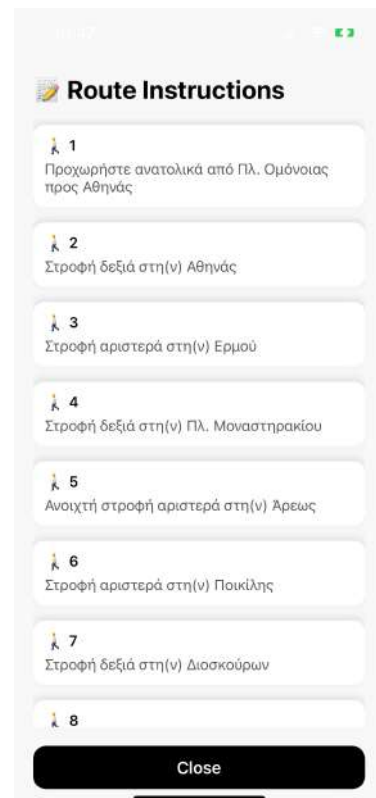
Σχήμα 5.33: Πλοήγηση με επιλογή Driving



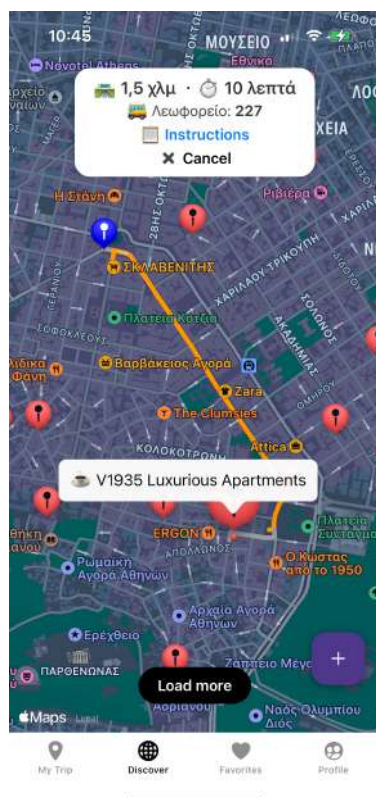
Σχήμα 5.34: Αναλυτικές οδηγίες διαδρομής



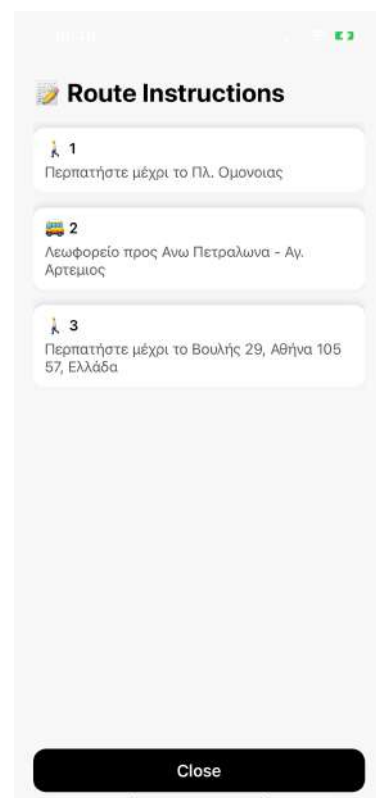
Σχήμα 5.35: Πλοήγηση με επιλογή Walking



Σχήμα 5.36: Αναλυτικές οδηγίες διαδρομής



Σχήμα 5.37: Πλοήγηση με επιλογή Transit



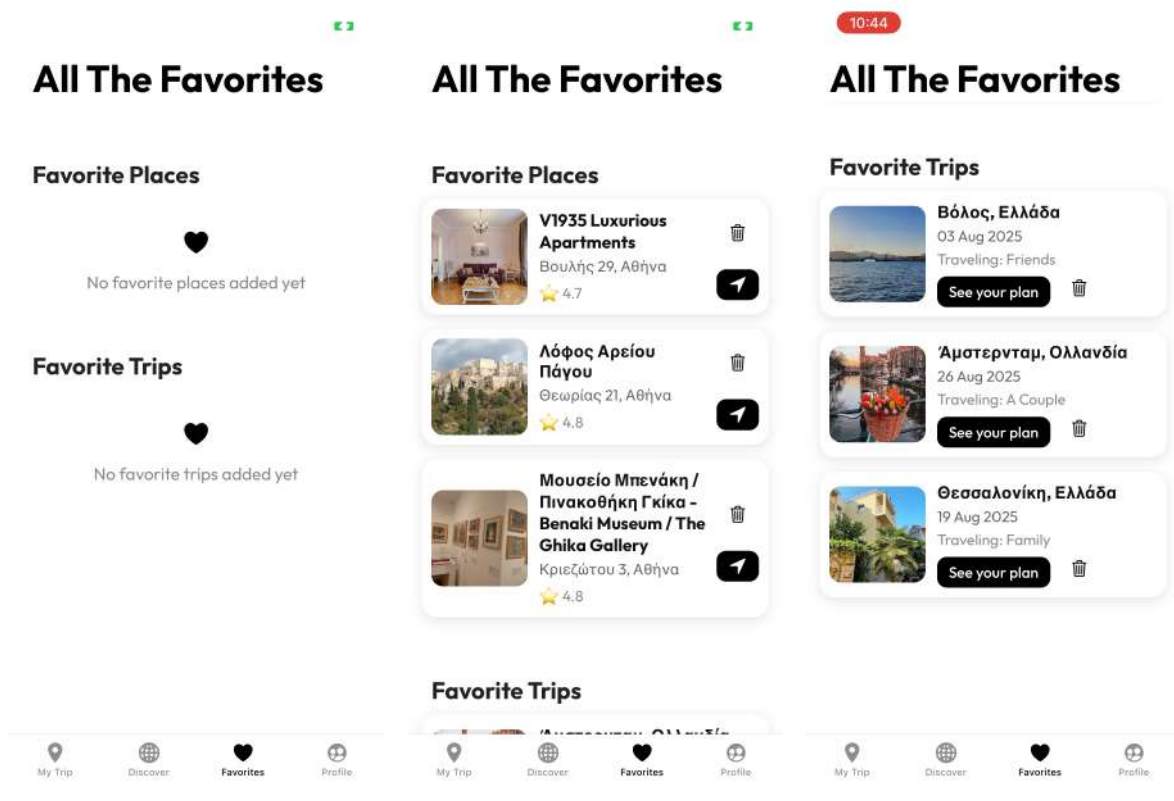
Σχήμα 5.38: Αναλυτικές οδηγίες διαδρομής

5.10 Αγαπημένα (Favorites)

Η λειτουργία **Favorites** επιτρέπει στον χρήστη να αποθηκεύει και να διαχειρίζεται τα αγαπημένα του σημεία (τοποθεσίες) και ταξίδια. Η σελίδα αυτή χωρίζεται σε δύο ενότητες: τα **Favorite Places** και τα **Favorite Trips**. Αν δεν έχουν προστεθεί αγαπημένα, εμφανίζεται σχετικό μήνυμα ενημέρωσης.

Κύρια Χαρακτηριστικά

- Προβολή αγαπημένων τοποθεσιών με εικόνα, όνομα, διεύθυνση και βαθμολογία
- Δυνατότητα άμεσης πλοήγησης προς αγαπημένη τοποθεσία
- Προβολή αγαπημένων ταξιδιών με εικόνα, ημερομηνία, τύπο ταξιδιού και δυνατότητα εμφάνισης του πλάνου (**See your plan**)
- Δυνατότητα διαγραφής αγαπημένων



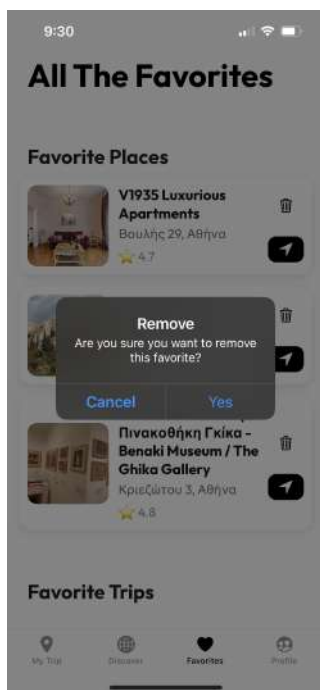
(α) Κενή λίστα

(β) Αγαπημένες Τοποθεσίες

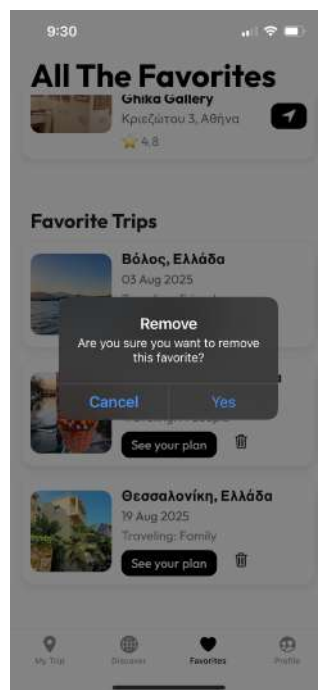
(γ) Αγαπημένα Ταξίδια

Σχήμα 5.39: Παραδείγματα εμφάνισης αγαπημένων (Favorites)

Επιπλέον, κατά την επιλογή διαγραφής εμφανίζεται μήνυμα επιβεβαίωσης ώστε να αποφευχθούν τυχόν λάθη.



(α) Επιβεβαίωση διαγραφής τοποθεσίας



(β) Επιβεβαίωση διαγραφής ταξιδιού

Σχήμα 5.40: Μηνύματα επιβεβαίωσης κατά τη διαγραφή αγαπημένων

Κεφάλαιο 6

Συμπεράσματα

Στο κεφάλαιο αυτό παρουσιάζονται συνοπτικά οι βασικές λειτουργίες της εφαρμογής και ο ρόλος της στη δημιουργία ταξιδιωτικών πλάνων. Στη συνέχεια προτείνονται μελλοντικές βελτιώσεις για την περαιτέρω ενίσχυσή της.

6.1 Σύνοψη και συμπεράσματα

Η παρούσα διπλωματική εργασία είχε ως στόχο την ανάπτυξη μιας εφαρμογής για κινητά, που διευκολύνει τον προγραμματισμό ταξιδιών. Μέσα από την εφαρμογή, ο χρήστης μπορεί να αναζητήσει προορισμούς, να βρει ξενοδοχεία, εστιατόρια και αξιοθέατα, να χρησιμοποιήσει τον χάρτη για πλοήγηση και να δημιουργήσει το δικό του εξατομικευμένο πλάνο ταξιδιού. Η εφαρμογή υλοποιήθηκε σε *React Native*, ώστε να λειτουργεί τόσο σε iOS όσο και σε Android, ενώ για τα δεδομένα τοποθεσίας και τις διαδρομές αξιοποιήθηκαν τα *Google Places*, *Directions* και *Maps APIs*. Το backend αναπτύχθηκε με *Node.js/Express* και *MongoDB*, αναλαμβάνοντας την αυθεντικοποίηση των χρηστών, τη διαχείριση ταξιδιών και αγαπημένων, καθώς και τη δημιουργία προτάσεων με χρήση τεχνητής νοημοσύνης.

Πιο συγκεκριμένα, η εφαρμογή προσφέρει τις εξής βασικές λειτουργίες:

- **Εγγραφή και Σύνδεση** (*Sign up / Sign in*): δημιουργία νέου λογαριασμού ή είσοδος σε υπάρχοντα
- **Προβολή ταξιδιών** (*My Trips*): εμφάνιση αποθηκευμένων ταξιδιών σε κατηγορίες (Latest Trip, Upcoming Trips, Past Trips και ταξίδια μελών)

- **Δημιουργία ταξιδιού (Create Trip):** επιλογή προορισμού, καθορισμός τύπου ταξιδιωτών, ορισμός ημερομηνιών και προϋπολογισμού
- **Πλάνο ταξιδιού (Trip Details):** αναλυτικό πλάνο με προτάσεις AI για πτήσεις, διαμονή και ημερήσιες δραστηριότητες
- **Εξατομικευμένο ταξίδι (Custom Trip):** διαχείριση πλάνου μέσω *Timeline* και *Calendar*, με δυνατότητα προσθήκης δραστηριοτήτων, προεπισκόπησης διαδρομών και εισαγωγής σημείων από τον χάρτη
- **Ανακάλυψη (Discover):** διαδραστικός χάρτης με αναζήτηση, φίλτρα, εντοπισμό θέσης, οδηγίες πλοήγησης και λεπτομέρειες τοποθεσιών
- **Αγαπημένα (Favorites):** αποθήκευση και οργάνωση σημείων ενδιαφέροντος και ταξιδιών
- **Προφίλ χρήστη (Profile):** προβολή και επεξεργασία προσωπικών στοιχείων, καθώς και δυνατότητα αποσύνδεσης ή διαγραφής λογαριασμού

Η εργασία καταλήγει στην ανάπτυξη μιας εφαρμογής που συγκεντρώνει όλες τις βασικές λειτουργίες ταξιδιωτικού προγραμματισμού σε ένα εύχρηστο περιβάλλον, καλύπτοντας ολόκληρη τη διαδικασία οργάνωσης ταξιδιού. Με την ενσωμάτωση του *Gemini AI*, η εφαρμογή μπορεί να δημιουργεί αυτόματα ταξιδιωτικά πλάνα, προσαρμοσμένα στις προτιμήσεις του χρήστη και σε περιορισμούς όπως οι ημερομηνίες ή ο διαθέσιμος προϋπολογισμός. Έτσι, προσφέρει μία ολοκληρωμένη εμπειρία που ξεκινά από την αναζήτηση και την εξερεύνηση και φτάνει έως τη δημιουργία και την αποθήκευση του τελικού πλάνου.

6.2 Μελλοντικές επεκτάσεις

Με βάση την τρέχουσα υλοποίηση, προτείνονται οι παρακάτω επεκτάσεις:

- **Ημερολόγιο & Timeline:** μετακίνηση δραστηριοτήτων με *drag & drop*, αντιγραφή/επικόλληση ημερών και αυτόματος εντοπισμός συγκρούσεων (*conflict detection*)
- **Έξυπνη βελτιστοποίηση πλάνου:** αυτόματη διάταξη δραστηριοτήτων με βάση ωράρια, δημοτικότητα/ουρές, καιρικές συνθήκες και εκτιμώμενους χρόνους μετακίνησης

- **Δημόσιες μεταφορές:** ενσωμάτωση ζωντανών δρομολογίων και ενημερώσεων καθυστερήσεων
- **Ανοικτά APIs:** σταδιακή μετάβαση σε *OpenStreetMap* [42], *OpenTripMap* [43] και *OpenRouteService* [44]
- **Caching & offline:** αποθήκευση χάρτη, βασικών στοιχείων POIs και πλάνων για λειτουργία εκτός σύνδεσης
- **Καιρός στο πλάνο:** πρόγνωση ανά ημέρα/δραστηριότητα και προτάσεις προσαρμογής
- **Συνομιλιακός σχεδιασμός:** συνομιλία με *chatbot* και αυτόματη ενημέρωση του *Timeline* σε πραγματικό χρόνο
- **Μάθηση προτιμήσεων:** το σύστημα μαθαίνει από τις επιλογές του χρήστη και βελτιώνει την κατάταξη (*ranking*) των προτάσεων
- **Έξυπνες ειδοποιήσεις:** παρακολούθηση τιμών/εισιτηρίων, προειδοποιήσεις καιρού και εναλλακτικές προτάσεις
- **Κοινή επεξεργασία:** συνεργασία σε πραγματικό χρόνο, σχόλια ανά δραστηριότητα και ιστορικό αλλαγών
- **Κοινή χρήση & εξαγωγές:** *shareable links*, εξαγωγή σε PDF και συγχρονισμός με Google/Apple Calendar
- **Έξοδα & προϋπολογισμός:** καταγραφή δαπανών, πολλαπλά νομίσματα και επιμερισμός (*split*) λογαριασμών
- **Αξιολόγηση:** δείκτες ικανοποίησης και υιοθέτησης πλάνων

Με τις παραπάνω επεκτάσεις, η εφαρμογή μπορεί να εξελιχθεί σε μια πλήρη πλατφόρμα ταξιδιωτικού προγραμματισμού, ικανή να υποστηρίξει ποικίλα σενάρια, με υψηλή αυτοματοποίηση, αξιοπιστία και ευχρηστία, ενώ κρατά χαμηλό κόστος χάρη στα ανοικτά δεδομένα και τη βελτιωμένη αρχιτεκτονική.

Βιβλιογραφία

- [1] Tripit. <https://www.tripit.com/web>. Ημερομηνία πρόσβασης: 25-08-2025.
- [2] Tripadvisor. <https://www.tripadvisor.com.gr/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [3] Sygic travel. <https://www.sygic.com/travel>. Ημερομηνία πρόσβασης: 25-08-2025.
- [4] Routeperfect. <https://www.routeperfect.com/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [5] Stippl. <https://www.stippl.io/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [6] Wanderlog travel planner. <https://wanderlog.com/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [7] React native vs flutter vs swift/kotlin for mobile app development. <https://colaninfotech.com/blog/react-native-vs-flutter-vs-swift-kotlin-for-mobile-app-development/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [8] Node.js vs django vs laravel. <https://dianapps.com/blog/nodejs-vs-django-vs-laravel/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [9] Mongodb vs sql. <https://www.knowi.com/blog/mongodb-vs-sql/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [10] Mapbox vs google maps vs openstreetmap. <https://relevant.software/blog/choosing-a-map-amapbox-google-maps-openstreetmap/>. Ημερομηνία πρόσβασης: 26-08-2025.

- [11] Javascript. https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/What_is_JavaScript. Ημερομηνία πρόσβασης: 26-08-2025.
- [12] Html. <https://html.com/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [13] Css. https://www.w3schools.com/css/css_intro.asp. Ημερομηνία πρόσβασης: 26-08-2025.
- [14] Introducing jsx. <https://legacy.reactjs.org/docs/introducing-jsx.html>. Ημερομηνία πρόσβασης: 26-08-2025.
- [15] React native documentation. <https://reactnative.dev>. Ημερομηνία πρόσβασης: 25-08-2025.
- [16] Expo documentation. <https://docs.expo.dev>. Ημερομηνία πρόσβασης: 25-08-2025.
- [17] Node.js documentation. <https://nodejs.org/en/docs>. Ημερομηνία πρόσβασης: 25-08-2025.
- [18] Express.js guide. <https://expressjs.com/en/guide/routing.html>. Ημερομηνία πρόσβασης: 25-08-2025.
- [19] Mongodb manual. <https://www.mongodb.com/docs/manual/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [20] Mongodb developer data platform. <https://www.mongodb.com/products/platform>. Ημερομηνία πρόσβασης: 26-08-2025.
- [21] Hypertext transfer protocol (http). <https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [22] Http status codes. <https://umbraco.com/knowledge-base/http-status-codes/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [23] Rest api. <https://www.geeksforgeeks.org/node-js/rest-api-introduction/>. Ημερομηνία πρόσβασης: 26-08-2025.

- [24] Axios http client. <https://axios-http.com/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [25] Google maps platform documentation (maps, places, directions). <https://developers.google.com/maps/documentation>. Ημερομηνία πρόσβασης: 25-08-2025.
- [26] Json web tokens. <https://www.jwt.io/introduction#what-is-json-web-token-structure>. Ημερομηνία πρόσβασης: 26-08-2025.
- [27] Gemini & ai for developers. <https://ai.google.dev/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [28] Cors. <https://www.geeksforgeeks.org/node-js/npm-cors/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [29] dotenv. <https://www.npmjs.com/package/dotenv>. Ημερομηνία πρόσβασης: 26-08-2025.
- [30] Mongoose documentation. <https://mongoosejs.com/docs/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [31] Password hashing using bcrypt. https://medium.com/@bhupendra_Maurya/password-hashing-using-bcrypt-e36f5c655e09. Ημερομηνία πρόσβασης: 26-08-2025.
- [32] Gemini api overview. <https://ai.google.dev/gemini-api/docs>. Ημερομηνία πρόσβασης: 25-08-2025.
- [33] Implementing user authentication with sign in with apple. <https://developer.apple.com/documentation/authenticationservices/implementing-user-authentication-with-sign-in-with-apple>. Ημερομηνία πρόσβασης: 26-08-2025.
- [34] Node.js project structure. <https://medium.com/@jayjethava101/node-js-project-structure-best-practices-and-example-for-clean-code-3e1f5530fd3b>. Ημερομηνία πρόσβασης: 26-08-2025.
- [35] Expo go. <https://expo.dev/go>. Ημερομηνία πρόσβασης: 26-08-2025.

- [36] Android studio. <https://developer.android.com/studio/intro>. Ημερομηνία πρόσβασης: 25-08-2025.
- [37] React navigation docs. <https://reactnavigation.org/docs/getting-started>. Ημερομηνία πρόσβασης: 25-08-2025.
- [38] Expo router documentation. <https://expo.github.io/router/docs>. Ημερομηνία πρόσβασης: 25-08-2025.
- [39] Mapview. <https://docs.expo.dev/versions/latest/sdk/map-view/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [40] Moment.js. <https://momentjs.com/docs/#/-project-status/>. Ημερομηνία πρόσβασης: 26-08-2025.
- [41] @expo/vector-icons. <https://www.npmjs.com/package/@expo/vector-icons>. Ημερομηνία πρόσβασης: 26-08-2025.
- [42] Openstreetmap. <https://wiki.openstreetmap.org/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [43] Opentripmap. <https://opentripmap.io/docs>. Ημερομηνία πρόσβασης: 25-08-2025.
- [44] Openrouteservice. <https://openrouteservice.org/documentation/>. Ημερομηνία πρόσβασης: 25-08-2025.
- [45] Ldplayer android emulator. <https://www.ldplayer.net/>. Ημερομηνία πρόσβασης: 25-08-2025.

ΠΑΡΑΡΤΗΜΑΤΑ

Παράρτημα Α

Εγκατάσταση Εφαρμογής

A.1 Πηγαίος Κώδικας

Ο πηγαίος κώδικας της εφαρμογής οργανώνεται σε δύο βασικά μέρη: **backend** (Node.js/Express για API και βάση δεδομένων) και **frontend** (React Native με Expo για το περιβάλλον χρήστη). Κατεβάστε τα αρχεία από τον παρακάτω σύνδεσμο :

<https://github.com/fbanou/travel-planner>

A.2 Οδηγίες Εγκατάστασης

Προαπαιτούμενα

- Node.js
- MongoDB (τοπικά ή μέσω cloud, π.χ. Atlas)
- Εφαρμογή **Expo Go** σε κινητό ή Android/iOS emulator

Backend

1. Μεταβείτε στον φάκελο:

```
cd backend
```

2. Εγκαταστήστε τις βιβλιοθήκες:

```
npm install
```

3. Ρυθμίστε το αρχείο `.env` με τις απαραίτητες παραμέτρους
(π.χ. `MONGO_URI`, `GEMINI_API_KEY`)

4. Εκκινήστε τον server:

```
npm start
```

Frontend

1. Μεταβείτε στον φάκελο:

```
cd frontend
```

2. Εγκαταστήστε τις βιβλιοθήκες:

```
npm install
```

3. Εκκινήστε την εφαρμογή μέσω Expo:

```
npx expo start
```

4. Σκανάρετε το QR code με την εφαρμογή **Expo Go** ή τρέξτε σε emulator

Σύνδεση Frontend – Backend

Η επικοινωνία πραγματοποιείται μέσω REST API. Στο αρχείο `services/api.js` ορίζεται το `baseUrl`, το οποίο πρέπει να αντιστοιχεί στη διεύθυνση IP του υπολογιστή που εκτελεί το backend. Η μορφή του είναι η εξής:

```
baseUrl: 'http://<IP_ADDRESS>:<PORT>/api'
```

Παράρτημα Β

Έλεγχος και Δοκιμή σε Συσκευές

Για την αξιολόγηση και τον έλεγχο της εφαρμογής χρησιμοποιήθηκαν τόσο φυσικές συσκευές όσο και εξομοιωτές. Ο συνδυασμός φυσικής συσκευής iOS και διαφορετικών Android εξομοιωτών επέτρεψε την επαρκή αξιολόγηση της εφαρμογής σε πολλαπλά περιβάλλοντα, εξασφαλίζοντας τη σωστή λειτουργία τόσο σε πραγματικές συσκευές όσο και σε εικονικά περιβάλλοντα ανάπτυξης.

iOS Συσκευή

Η εφαρμογή δοκιμάστηκε σε πραγματική συσκευή iOS (iPhone), μέσω της εφαρμογής **Expo Go** [35]. Η χρήση φυσικής συσκευής εξασφαλίζει την πιο αξιόπιστη εικόνα για την απόδοση, την αλληλεπίδραση με τον χρήστη και τη συμβατότητα με τα χαρακτηριστικά του λειτουργικού συστήματος.

Android Συσκευές & Εξομοιωτές

Για τις δοκιμές στο λειτουργικό **Android** ακολουθήθηκαν δύο προσεγγίσεις:

- **Android Studio Emulator**

Αρχικά χρησιμοποιήθηκε ο επίσημος εξομοιωτής του **Android Studio** [36]. Πρόκειται για εργαλείο της Google που προσομοιώνει πλήρως συσκευές Android, επιτρέποντας την εκτέλεση και αποσφαλμάτωση εφαρμογών χωρίς φυσική συσκευή. Παρότι είναι ιδιαίτερα ακριβής, απαιτεί σημαντικούς πόρους συστήματος και σε ορισμένες περιπτώσεις παρουσιάζει καθυστερήσεις στην απόδοση.

- **LDPlayer**

Στη συνέχεια δοκιμάστηκε ο **LDPlayer** [45], ένας ελαφρύτερος εξομοιωτής Android για Windows. Είναι περισσότερο προσανατολισμένος σε gaming, ωστόσο προσφέρει γρήγορη απόκριση και σταθερή λειτουργία για εφαρμογές React Native μέσω Expo. Αποδείχθηκε πιο πρακτικός για καθημερινές δοκιμές σε σχέση με το Android Studio.