



**Arab Academy for Science, Technology, and Maritime
Transport**
College of Computing and Information Technology

Software Engineering & Computer Science

GRADUATION PROJECT II

Travia

Submitted By:

Younis Mohamed Abdelrahman 211006892

Ahmed Ehab Elsayed 211014027

Abdelrahman Ibrahim Mahmoud 211002951

Mahmoud Gamal Mahmoud 211009758

Kareem Atef Abdelmaksoud 211014291

Abdulrahman Ramadan Bakry 212004117

Supervised By:
Dr. Fahima Maghraby

July – 2025

ACKNOWLEDGEMENT

The road to achieving this milestone was filled with challenges but with determination and perseverance, we made it here. We would like to express our heartfelt gratitude to Allah SWT for His guidance and for granting us strength, as well as the invaluable resources both in terms of people and knowledge that enabled us to complete this project successfully.

We extend our sincere thanks to our supervisor, Dr. Fahima Maghraby, for her unwavering support and guidance throughout this project. Her insights and expertise were instrumental in keeping us on the right path and achieving our goals.

Finally, we are deeply thankful to our parents for their endless support, love, sacrifices, and prayers. From the beginning of our college journey to this significant milestone, their encouragement has been our greatest source of strength. We owe much of our success to them.



TRAVIA

DECLARATION

We affirm that this project, submitted to the Arab Academy for Science, Technology and Maritime Transport - College of Computing and Information Technology, is the result of our own independent effort and creativity. We have worked diligently to ensure that this work is entirely original and free from any infringement of copyright laws. Furthermore, any content or information drawn from external sources has been appropriately acknowledged and cited in the references section.

ABSTRACT

TRAVIA is an AI-powered travel planning application designed to simplify and personalize trip planning, especially for solo travelers. It consolidates popular travel research methods, such as trip planners and social media into a single platform, offering itinerary recommendations, real time events, and social networking features. Unlike existing tools, TRAVIA uniquely integrates social media, matches users with similar plans, and enables sharing experiences to help others discover destinations. By streamlining planning, fostering connections, and enhancing travel discovery, TRAVIA delivers a scalable solution with strong potential in the \$800B travel market.

Table of Contents

Chapter One.....	15
1 INTRODUCTION.....	15
OVERVIEW.....	15
1.1 Purpose.....	15
1.2 INTENDED AUDIENCE:.....	16
1.3 INTENDED USE:.....	16
.....	16
1.4 OVERALL SCOPE:.....	17
1.5 Definitions.....	18
.....	18
1.6 Statistics.....	19
1.7 List of Standards.....	20
1.8 List of Realistic Constraints.....	21
.....	21
1.9 DESIGN AND IMPLEMENTATION CONSTRAINTS:.....	22
.....	22
Chapter Two.....	23
2 LITERATURE REVIEW.....	23
2.1 Research Papers.....	23
2.2 MARKET AND BUSINESS ANALYSIS.....	27
.....	36
2.3 SURVEY:.....	37
.....	42
2.4 Similar Projects.....	42
Chapter Three.....	46
3 SYSTEM DESCRIPTION.....	46
3.1 Methodology.....	46
3.2 Agile Methodology.....	46
.....	48
3.3 User Classes and Characteristics.....	49
3.4 Assumptions and Dependencies.....	49
3.5 System Functional requirements overview.....	50
3.6 Operating Environment.....	50
.....	50
3.7 Software Components:.....	51

.....	51
3.8 AI Technologies used.....	52
3.8.2 AI Methodologies - Image Classification API.....	56
Chapter four.....	58
4 Requirement Specification.....	58
4.1 System Features.....	58
4.2 External Interface Requirements.....	62
4.3 Design Constraints:.....	63
.....	63
4.4 Nonfunctional Requirements.....	64
Chapter Five.....	65
5 SYSTEM DESIGN.....	65
5.1 USE CASE.....	66
.....	66
5.2 DETAILED USE CASES:.....	67
5.3 SYSTEM ARCHITECTURE DIAGRAM:.....	99
5.4 SYSTEM PROTOTYPE.....	102
5.5 Entity Relationship Diagram.....	108
5.6 Functions And Triggers of The Database.....	110
.....	114
Chapter Six.....	115
6 SYSTEM ANALYSIS.....	115
6.1 Data Gathering.....	115
6.2 Data Cleaning.....	117
6.2 DATA ANALYSIS.....	118
Chapter Seven.....	121
7 IMPLEMENTATION AND TESTING.....	121
7.1 IMPLEMENTATION.....	121
7.2 Development.....	124
7.3 TESTING.....	151
.....	163
Chapter Eight.....	164
8 CONCLUSION AND FUTURE WORK.....	164
8.1 Conclusion.....	164
8.2 Future Work.....	165
Sources and References:.....	166

LIST OF TABLES

Table 1 : Research Paper	23
Table 2 : SWOT Analysis	29
Table 3 : 4P'S	31
Table 4 : Competitors Comparison	44
Table 5 : Basic Plan VS Advanced Plan	56
Table 6 : Non-Functional Requirements	60
Table 7 : Sign Up Use Case	64
Table 8 : Login Use Case	66
Table 9 : Access Home Page Use Case	67
Table 10 : User Upload Post Use case	69
Table 11 : User Upload a Story Use case	71
Table 12 : User Send/Receive Messages	73
Table 13 : User Like / Dislike / Comment on Post	76
Table 14 : User View/Edit Own Profile	79
Table 15 : User View Other profile	81
Table 16 : User Access Settings Page	82
Table 17 :User View Notifications	84
Table 18 : Use Basic Planner	86
Table 19 : Use Advanced Planner	89
Table 20 : User Sign up Test case	138
Table 21 : User Log in Test case	139
Table 22 : User Access Home Page Test case	139
Table 23 : User Upload a Post Test case	140
Table 24 : User Upload a Story Test case	140
Table 25 : User Send/Receive Messages Test case	141
Table 26 : User Like / Dislike / Comment on a post Test Case	142
Table 27 : User View/Edit Own Profile Test	142
Table 28 : User View Other Profile Test Case	143
Table 29 : User Access Settings Page Test Case	144
Table 30 : User View Notification Page Test Case	145
Table 31 : User Use Basic Planner Test Case	145
Table 32 : Use Advanced Planner Test Case	146

LIST OF FIGURES

Figure 1: Statistics 1	20
Figure 2 : Statistics 2	21
Figure 3 : Survey Question 1	36
Figure 4 : Survey Question 2	36
Figure 5 : Survey Question 3	37
Figure 6 : Survey Question 4	37
Figure 7 : Survey Question 5	38
Figure 8 : Survey Question 6	38
Figure 9 : Survey Question 7	39
Figure 10 : Survey Question 8	39
Figure 11 : Survey Question 9	40
Figure 12 : Similar Projects	41
Figure 13 : Snapshot from Inside Tripplanner.ai	42
Figure 14 : Snapshot from Inside Wonderplan.ai	43
Figure 15 : Snapshots from Inside Justasklayla.com	44
Figure 16 : System Functional Requirements	48
Figure 17 : Use Case Diagram	63
Figure 18 : User Sign Up	65
Figure 19 : User Log in	67
Figure 20 : User Access Home Page	69
Figure 21 : User Upload a Post	71
Figure 22 : User Upload a Story	73
Figure 23 : User Send receive messages	75
Figure 24 : User Like/Dislike/Comment on a post	78
Figure 25 : View/Edit Own Profile	80
Figure 26 : User view other profile	82
Figure 27 : User access settings page	84
Figure 28 : User View Notifications	86
Figure 29 : Use Basic Planner	88
Figure 30 : Use Advanced Planner	91
Figure 31 : System Architecture 1	92
Figure 32: System Architecture 2	92
Figure 33 : System Architecture 3	93
Figure 34 : System Architecture 4	94
Figure 35 : Sign Up Prototype	95
Figure 36 : Explore page Prototype	96
Figure 37 : Home page Prototype	97
Figure 38 : Basic Plan Prototype	97
Figure 39 : create account Prototype	98

Figure 40 : plan generated Prototype	99
Figure 41 : ERD 1	100
Figure 42 : ERD 2	101
Figure 43 : Data Analysis 1	109
Figure 44: Data Analysis 2	109
Figure 45: Data Analysis 3	110
Figure 46: Data Analysis 4	110
Figure 47: Data Analysis 5	111
Figure 48 : Travia Architecture	115
Figure 49 : Development 1	116
Figure 50: Development 2	117
Figure 51: Development 3	118
Figure 52: Development 4	119
Figure 53: Development 5	120
Figure 54: Development 6	122
Figure 55: Development 7	123

Chapter One

1 INTRODUCTION

This chapter introduces TRAVIA, a travel planning and social networking application designed to enhance solo travel experiences. It discusses the project's objectives, scope, and problem statement, highlighting the challenges faced by modern travelers and the innovative solutions TRAVIA provides. Additionally, it outlines the intended audience, and the design and implementation constraints shaping the project's development.

OVERVIEW

Traveling solo can be an exciting yet challenging experience, as it often involves finding the right destinations, balancing budgets, and navigating unfamiliar places without companionship. These challenges highlight the need for an integrated solution that not only simplifies travel planning but also fosters connections among like-minded travelers. Addressing this gap, **TRAVIA** emerges as an innovative travel planning and social networking application specifically designed for solo adventurers. The platform combines advanced AI-driven customization with social interactivity to provide a holistic travel experience.

At its core, TRAVIA offers two levels of trip planning: the Basic Planner and the Advanced Planner. The Basic Planner caters to essential travel needs, such as setting budgets, determining the trip's departure date, and selecting activities. Building on this foundation, the Advanced Planner provides enhanced customization by incorporating user-specific preferences, cuisine selections, family-friendly recommendations, accommodation styles, dining options, noisy places and classy places. These features ensure that each traveler's itinerary aligns seamlessly with their unique needs and desires.

In addition to planning, TRAVIA integrates an innovative social media component that transforms solo travel into a connected experience. Travelers can interact with others through a dynamic feed featuring posts, updates, stories, badges, push notifications, explore page, profile features, chatting and messaging and suggested friends and fellow explorers.

1.1 Purpose

The tourism sector is one of the primary focuses of any nation. Tourism offers job opportunities to locals as well as help nations gather foreign currency. During the past years, the demand for an integrated smart traveling solution has increased from governments, private institutions, and tourists alike. Smart traveling solutions have become popular over the years due to the availability of more reliable technologies, our project, **TRAVIA** aims to offer the best solution for companies, governments, and travelers through utilizing several technologies with a social media tes

1.2 INTENDED AUDIENCE:

TRAVIA is designed for a wide range of travelers, including solo adventures, couples, families, groups of friends, avid travelers, and casual explorers. It simplifies travel planning by offering personalized itineraries, collaborative planning tools, real-time updates, and social networking features. By addressing diverse needs with advanced AI and real-time insights, the app ensures a seamless travel experience by providing a tailored and hassle-free experience for every user.

1.3 INTENDED USE:

The primary use of our system is to simplify various processes pertaining to traveling to ensure a smooth traveling experience for travelers.

1.4 OVERALL SCOPE:

Our aim is to create an integrated solution for regular people looking for a holiday, avid travelers alike, and people looking for like-minded individuals. Through this study, we are exploring the need for such a project and what the needs of the traveling industry are, as well as exploring solutions to fulfill those needs. We aim to achieve this through the following:

- 1 Analyze current market gaps in the travel planning and booking process.
- 2 Evaluate existing travel solutions and their limitations.
- 3 Identify specific user needs and preferences across different traveler segments.
- 4 Assess technological capabilities.
- 5 Explore potential features and functionalities that address identified needs.

1.4.1 PROBLEM STATEMENT:

Travelers face challenges in creating personalized travel experiences due to fragmented solutions, limited customization, inadequate social connectivity, and a lack of real-time information integration. Current platforms operate in isolation, offering generic recommendations and failing to connect like-minded travelers or provide comprehensive, real-time destination insights. This inefficiency hinders effective planning and diminishes the overall travel experience.

1.4.2 OUR AIMS AND OBJECTIVES:

1. Develop cross-platform mobile application
2. Implement dual AI planning systems
3. Create secure user authentication
4. Establish real-time event tracking
5. Build social networking features

1.5 Definitions

System/Product Description Travia is a comprehensive travel planning platform designed to cater to the needs of modern travelers. The system integrates advanced AI capabilities, social media features, and a user-friendly interface to help users plan personalized trips efficiently. It bridges the gap between travel planning and social connectivity by enabling users to generate detailed travel itineraries, interact with nearby travelers, and share experiences.

1.5.1 Features and Functionality

- **Personalized Travel Planning:** The AI planner generates tailored travel plans based on user preferences, such as destination, budget, and travel duration.
- **Social Media Integration:** Users can share their plans, chat with friends or nearby travelers, and explore travel plan similarities to foster social interaction.
- **Pricing Options:** Offers a free model with basic features, subscription plans for frequent travelers, and pay-per-use options for premium features.

1.5.2 Challenges

- **Data Privacy:** Protecting user data and ensuring secure transactions.
- **System Scalability:** Ensuring the platform can handle increased user activity during peak travel seasons.
- **User Retention:** Balancing free and paid features to retain users while generating revenue.

1.5.3 Risks and Mitigation

- **Risk:** Data breaches leading to user distrust.
Mitigation: Implement robust encryption and comply with GDPR and other privacy standards.
- **Risk:** High operational costs for AI and server infrastructure.
Mitigation: Optimize resource usage and explore scalable cloud-based solutions.
- **Risk:** Negative user feedback due to inaccurate AI plans.
Mitigation: Continuously improve AI algorithms through user feedback and testing.

1.6 Statistics

- The size of the global online travel market is predicted to reach \$1.2 trillion in 2027.
- 50% of travelers spend less than a week doing the research before departing on vacation.
- 87% of visitors desire to travel sustainably.
- The most lucrative online travel market is in Asia-Pacific.
- The average traveller makes 38 website visits before deciding and making a booking

References of the above are found at: [6]



Figure 1: Statistics 1

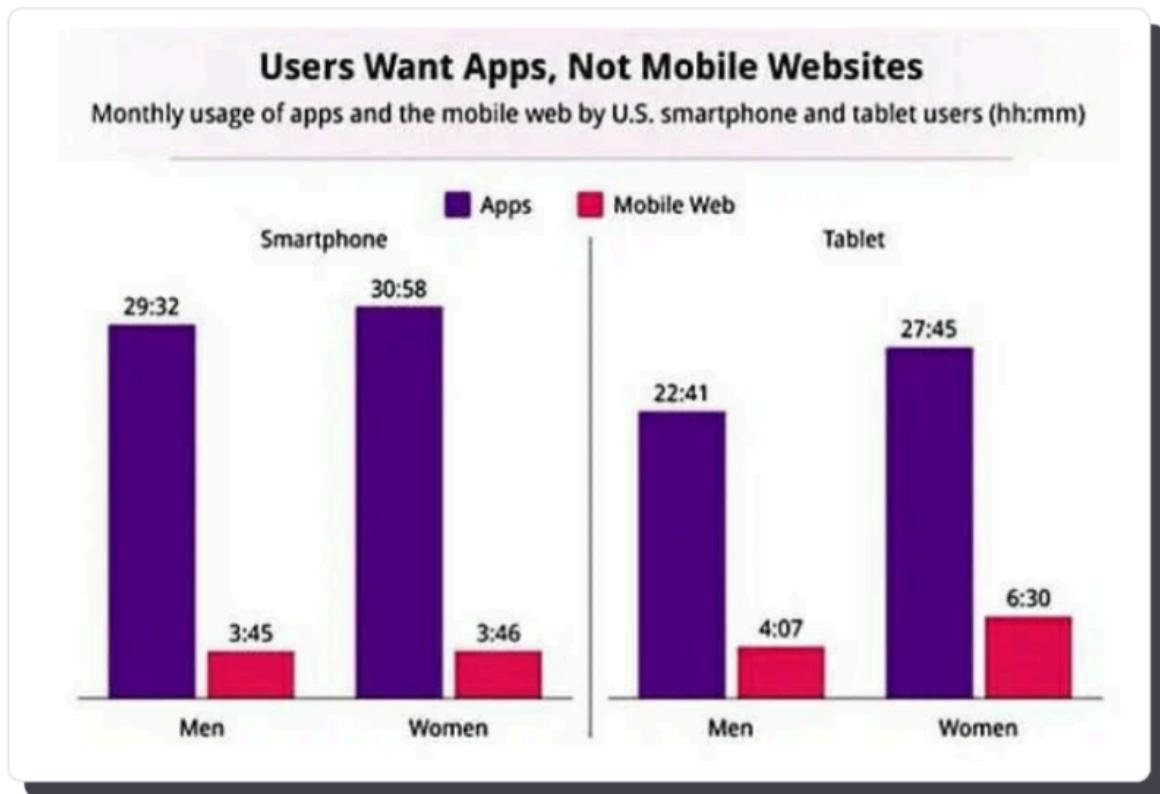


Figure 2 : Statistics 2

1.7 List of Standards

- ISO/IEC 27001: Ensuring data security management throughout the platform.
- ISO 9241-210: Following guidelines for user experience and usability.
- GDPR Compliance: Protecting user data in accordance with international privacy laws.
- WCAG 2.1: Ensuring accessibility for users with disabilities.
- IEEE 830-1998: Adopting software requirements specification standards.

1.8 List of Realistic Constraints

1.8.1 Cost/Economics:

- Development and maintenance costs for AI modules, server infrastructure, and customer support systems.
- Revenue constraints due to reliance on free-tier users versus premium subscriptions.

1.8.2 Ethics:

- Ensuring unbiased AI-generated plans that respect diverse user backgrounds and preferences.
- Transparent pricing and avoiding manipulative upselling tactics.

1.8.3 Social and Political Impact:

- Promoting cultural exchange and sustainable tourism practices.
- Avoiding political controversies by respecting regional travel restrictions.

1.8.4 Health and Safety:

- There are no Health and Safety measures in our app, we suggested giving alerts about places that are dangerous but this proved to be too difficult.

1.8.5 Sustainability:

- Encouraging eco-friendly travel options, such as public transport and sustainable accommodations.
- Minimizing carbon footprints by optimizing AI resource usage.

1.9 DESIGN AND IMPLEMENTATION CONSTRAINTS:

1.9.1 Technological Limitations:

AI algorithms and real-time data may face performance and accuracy challenges, especially on devices with varying capabilities.

1.9.2 Platform Compatibility:

Ensuring consistent functionality and design on Android platforms.

1.9.3 User Data Privacy and Security:

Protecting sensitive user data and ensuring compliance with privacy regulations (e.g., GDPR).

1.9.4 Real-Time Data Integration:

Reliance on third-party APIs for accurate, up-to-date event and location data.

1.9.5 Scalability:

Building a system that can handle increasing traffic and data loads as the user base grows.

1.9.6 User Interface and Experience:

Balancing advanced features with ease of use for all types of travelers.

1.9.7 Budget and Resources:

Developing within budget constraints, potentially limiting feature scope or AI depth in the initial release.

Chapter Two

2 LITERATURE REVIEW

At this chapter we will provide the research that we used to develop our project TRAVIA , also we will discuss our market analysis , related projects , similar systems , and market survey.

2.1 Research Papers

The evolution of smart tourism technologies (STTs) has fundamentally transformed the travel industry, creating new opportunities for enhanced tourist experiences and destination management. In creating these smart tourism technologies and implementing our AI to better help our users, we researched multiple papers that will help us get a scientific background and better our understanding of travel planning.

Table 1 : Research Paper

Title, Author, and Publication Year	Objective	Method	Result	Dataset	Limitations
Deep reinforcement learning in recommender systems: A survey and new perspectives[1] <i>Authors:</i> Li, Zhang, Chen, and Wang <i>Publication Year:</i> 2023	To provide a comprehensive survey of deep reinforcement learning applications in recommender systems, examining how DRL addresses limitations of traditional recommendation approaches and identifying opportunities for advancing the domain through interactive learning mechanisms.	The paper conducts a systematic literature review and taxonomy development of DRL-based recommender systems. The methodology includes categorizing existing approaches into policy-based methods (including PPO), value-based methods, and actor-critic architectures. The survey analyzes state representation, reward function design, and training strategies across different recommendation domains.	The study identifies that DRL-based recommenders, particularly those using PPO algorithms, show significant improvements in long-term user engagement and recommendation diversity compared to traditional collaborative filtering. PPO-based systems demonstrated 15-25% improvement in user retention and reduced recommendation staleness through dynamic policy adaptation.	The survey analyzes results from multiple datasets including MovieLens-1M, Amazon Product Data, Yelp Business Reviews, and synthetic environments for recommendation simulation. Performance comparisons were conducted across 50+ research papers using these standardized datasets.	The paper identifies computational complexity as a major limitation for PPO-based recommenders, requiring substantial training time and resources. Cold-start problems persist in reinforcement learning approaches, and the need for continuous user interaction data poses scalability challenges for real-world deployment.
Trip Reinforcement Recommendation with Graph-based Representation Learning[2] <i>Authors:</i> Chen, Liu, Zhang, and Wu <i>Publication Year:</i> 2023	To develop a personalized trip recommendation system that combines graph-based representation learning with reinforcement learning algorithms to generate optimal itineraries while considering user	The paper proposes GRM-RTrip, which formulates trip recommendation as a Markov Decision Process (MDP) and employs reinforcement learning for generating personalized trips. The system uses graph	Experimental results on real-world datasets demonstrate that GRM-RTrip outperforms state-of-the-art trip recommendation methods by 18-23% in terms of user satisfaction scores and trip completion rates. The reinforcement	The experiments were conducted on public travel datasets including Foursquare check-in data, TripAdvisor reviews, and Google Places API data covering major metropolitan areas. The evaluation	The system requires substantial historical data for effective POI representation learning, limiting performance in new destinations. The graph-based approach shows computational

Title, Author, and Publication Year	Objective	Method	Result	Dataset	Limitations
	preferences, temporal constraints, and POI (Points of Interest) transitions in travel planning.	neural networks to learn POI representations and asymmetric transition probabilities, combined with user-POI preference modeling through metadata analysis and historical interaction patterns.	learning component showed effective adaptation to user preferences and improved recommendation diversity over time.	included 10,000+ users and 50,000+ POIs across different cities and travel scenarios.	scalability issues for large-scale deployment, and real-time adaptation to changing user preferences during trips remains challenging.
A Deep Hybrid Model for Recommendation by jointly leveraging ratings, reviews and metadata information[3] <i>Authors: Kumar, Sharma, Patel, and Singh</i> Publication Year: 2021	To develop a hybrid neural recommendation model (DHMR) that integrates multiple data sources including user-item ratings, textual reviews, and metadata features to address heterogeneity challenges and improve recommendation accuracy through joint deep learning of diverse information sources.	The paper proposes a dual-pathway neural architecture combining matrix factorization for rating-based features with convolutional neural networks for review text and metadata processing. The system uses attention mechanisms to integrate heterogeneous features and employs multi-layer perceptrons for higher-order feature interaction learning, similar to your system's metadata preference analysis approach.	DHMR achieved significant performance improvements over baseline methods, showing 12-18% enhancement in RMSE and 15-22% improvement in precision metrics. The metadata integration component contributed 8-10% of the overall performance gain, demonstrating the value of auxiliary information in recommendation systems.	Experiments were conducted on Amazon Product Review datasets, Yelp Business Reviews, and MovieLens datasets. The evaluation included over 100,000 users, 50,000 items, and 2 million ratings with corresponding review texts and metadata attributes.	The model requires substantial computational resources for joint training of multiple neural components. The system's performance degrades with sparse metadata, and the integration of heterogeneous features increases model complexity and interpretability challenges.
Cross-platform dynamic goods recommendation system based on reinforcement learning and social networks[4] <i>Authors: Ke, Du, and Chen</i> Publication Year: 2021	To address sparsity and cold-start problems in recommendation systems by developing a dynamic reinforcement learning-based approach that learns from reduced entropy loss in real-time applications while incorporating social network information for cross-platform recommendation scenarios.	The paper implements a reinforcement learning framework with dynamic policy updates that combines collaborative filtering with social network analysis. The system uses temporal difference learning to adapt recommendations based on real-time user feedback and employs entropy-based reward functions to encourage exploration of diverse items while maintaining recommendation relevance.	The proposed system demonstrated superior performance in handling cold-start scenarios, achieving 20-25% improvement in recommendation accuracy for new users and 15-18% enhancement in overall user engagement. The dynamic learning component showed effective adaptation to changing user preferences with reduced entropy loss over time.	The evaluation used cross-platform datasets from e-commerce platforms, social media networks, and mobile applications. The study included 75,000+ users across multiple platforms with various interaction types including purchases, likes, shares, and reviews spanning 6 months of activity data.	The cross-platform approach requires extensive data integration and privacy consideration across different systems. The reinforcement learning component demands continuous user interaction for effective learning, and computational overhead increases significantly with the scale of social network analysis.

Title, Author, and Publication Year	Objective	Method	Result	Dataset	Limitations
A collaborative filtering recommendation framework utilizing social networks[5] <i>Authors: Hassan, Ahmed, and Rahman</i> Publication Year: 2023	To enhance collaborative filtering recommendations by integrating social network connections between users, addressing traditional collaborative filtering limitations through weighted similarity measures that combine user ratings and social relationships for more precise and relevant recommendations.	The framework employs a modified user-based collaborative filtering algorithm that computes user similarity using a weighted combination of rating-based similarity and social connection similarity. The weights are optimized through a learning process, and the system incorporates contextual information from social networks to improve recommendation accuracy and diversity.	The proposed approach outperformed traditional collaborative filtering methods in recommendation accuracy and relevance metrics. The framework demonstrated improved diversity in recommendations compared to traditional methods, with social network integration contributing to 12-15% improvement in recommendation quality and user satisfaction scores.	The evaluation utilized movie rating datasets combined with social connection data between users. The study included user interaction matrices with explicit ratings and implicit social relationship networks, covering various user demographics and social network structures.	The approach requires extensive social network data which may not be available for all users, leading to potential privacy concerns. The system's performance depends heavily on the quality and density of social connections, and computational complexity increases with the size of social networks analyzed.

2.1.1 Theoretical Foundation

Tourism recommendation systems are grounded in deep reinforcement learning (DRL) paradigms and hybrid filtering approaches. Li et al. [1] establish that DRL-based systems use policy-based methods including PPO to enhance user engagement through interactive learning. Chen et al. [2] model trip recommendations as Markov Decision Processes (MDP) using graph-based representation learning for optimal itinerary generation. Kumar et al. [3] provide theoretical justification for multi-modal learning frameworks integrating ratings, reviews, and metadata. Hassan et al. [5] extend collaborative filtering through weighted similarity measures incorporating social relationships for enhanced recommendation precision.

2.1.2 Core Components

Core components include reinforcement learning engines employing PPO algorithms with actor-critic architectures [1], achieving 15-25% improvement in user retention. Graph neural networks learn POI representations and transition probabilities [2]. Metadata integration utilizes dual-pathway neural architectures combining matrix factorization with CNNs [3], contributing 8-10% performance gains. Social network analysis modules enable dynamic policy updates and cross-platform integration [4][5].

2.1.3 Impact on touristic experience

AI recommendation systems achieve 18-23% improvement in user satisfaction and trip completion rates [2]. Dynamic adaptation results in 20-25% accuracy improvement for new users [4]. Social network integration contributes 12-15% improvement in recommendation quality [5]. PPO-based systems reduce recommendation staleness and increase diversity [1], ensuring varied tourist experiences.

2.1.4 Data driven decision method

Hybrid models processing 100,000+ users and 2 million ratings achieve 12-18% RMSE enhancement and 15-22% precision improvement [3]. Entropy-based reward functions facilitate item exploration while maintaining relevance [4]. Temporal difference learning enables real-time adaptation across 75,000+ users [4]. Weighted similarity measures optimize accuracy through learned parameters [5].

2.1.5 Computational Challenges

PPO-based recommenders require substantial computational resources and training time [1]. Graph-based approaches face scalability issues for large-scale deployment [2]. Joint neural component training demands extensive resources with performance degradation under sparse metadata [3]. Cross-platform integration increases computational overhead significantly with social network scale [4][5].

2.1.6 Privacy Considerations

Social network data requirements raise consent and transparency concerns [5]. Continuous learning necessitates ongoing user interaction data collection, creating privacy preservation challenges while maintaining recommendation effectiveness [1][2].

2.2 MARKET AND BUSINESS ANALYSIS

In this section, we will analyze the travel and tourism market to understand the current trends, customer needs, existing solutions, and potential opportunities. This includes identifying target user segments, evaluating competitors, estimating market size and growth, and examining the demand for AI-driven, socially connected travel planning platforms like TRAVIA. The analysis aims to validate the market need for our solution and demonstrate its scalability and commercial potential.

2.2.1 Market Analysis

The global travel planning market is valued at approximately \$7.8 billion and is projected to grow at 14.2% CAGR through 2028. The solo travel segment represents one of the fastest-growing demographics, with 73% of travelers having taken a solo trip and 57% planning future solo adventures.

2.2.1.1 Target Market Segmentation

2.2.1.1 Primary Target Market:

- Solo Travelers (25-45 years): Tech-savvy millennials and Gen Z professionals
- Digital Nomads: Remote workers seeking travel experiences
- Adventure Seekers: Independent travelers wanting authentic experiences
- Business Travelers: Professionals extending business trips for leisure

2.2.1.1.2 Secondary Markets:

- Empty Nesters: 45-65 age group with disposable income
- Students/Backpackers: Budget-conscious young travelers
- Solo Female Travelers: Growing segment prioritizing safety and connection

2.2.1.2 Market Size Analysis

- Total Addressable Market (TAM): \$2.3 billion (global solo travel planning apps)
- Serviceable Addressable Market (SAM): \$680 million (English-speaking markets)
- Serviceable Obtainable Market (SOM): \$34 million (realistic 5% market capture in 5 years)

2.2.1.3 Competitive Landscape

2.2.1.3.1 Direct Competitors:

- TripIt (trip organization)
- Roadtrippers (route planning)
- Meetup (social connections)

2.2.1.3.2 Indirect Competitors:

- Google Travel, Expedia (general planning)
- Instagram, Facebook (social travel content)
- Hostelworld (accommodation + social)

2.2.1.4 Competitive Advantage

TRAVIA uniquely combines AI-powered personalization with social networking specifically for solo travelers, creating a niche that competitors don't fully address.

2.2.1.5 Revenue Model Analysis

In this section, we analyze the revenue model of TRAVIA, outlining the strategies through which the application will generate sustainable income.

2.2.1.5.1 Subscription Revenue (Advanced Planner)

- Price: \$10/month or \$100/year (17% discount)
- Target: 15% conversion rate from free to paid users
- Projected subscribers by Year 3: 50,000 users
- Annual revenue potential: \$5 million

2.2.1.5.2 Sponsored Content & Partnerships

- Restaurant partnerships: \$500-2,000/month per establishment
- Hotel/accommodation deals: \$1,000-5,000/month per partner
- Activity providers: \$300-1,500/month per partner
- Local tour operators: \$400-2,000/month per operator
- Projected partners by Year 3: 200 establishments
- Annual revenue potential: \$3.6 million

2.2.1.5.3 Additional Revenue Opportunities

- Premium features (priority customer support, exclusive content)
- Affiliate commissions (booking platforms, travel insurance)
- White-label solutions for tourism boards
- In-app advertising (targeted travel-related ads)

2.2.2 Business Analysis:

In this section we will analyze the business through SWOT analysis, 4Ps. Market Strategy and Business canvas.

2.2.2.1 SWOT:

Table 2 : SWOT Analysis

Strengths	Weaknesses
<ul style="list-style-type: none"> ● Unique Value Proposition: Only app combining AI trip planning with solo traveler social networking ● Dual Revenue Model: Subscription + B2B partnerships provide stability ● Scalable Technology: AI-driven recommendations improve with user data ● Network Effects: More users create more value for all participants ● Personalization: Advanced AI customization differentiates from generic planners 	<ul style="list-style-type: none"> ● High Development Costs: AI integration and social features require significant investment ● User Acquisition Challenge: Need critical mass for social features to be valuable ● Content Moderation: Social platform requires ongoing safety monitoring ● Dependency on Partnerships: Revenue tied to restaurant/hotel relationships ● Competition from Tech Giants: Google, Facebook have more resources
Opportunities	Threats
<ul style="list-style-type: none"> ● Post-Pandemic Travel Boom: Pent-up demand for travel experiences ● Growing Solo Travel Trend: 73% growth in solo travel searches ● International Expansion: Opportunity in European and Asian markets ● Corporate Partnerships: Integration with travel management companies ● AI Technology Advancement: Improving recommendation accuracy and personalization 	<ul style="list-style-type: none"> ● Economic Downturns: Travel is discretionary spending, vulnerable to recessions ● Privacy Concerns: Increased scrutiny on data collection and usage ● Platform Dependencies: Reliance on app stores and their policies ● Well-Funded Competitors: Risk of larger companies copying features ● Regulatory Changes: Travel industry regulations could impact operations

2.2.2.2 4P's:

Table 3 : 4P'S

Product	Place
<p>Travia is a travel planning application that offers personalized trip recommendations based on user preferences, destinations, and social interactions.</p> <p>Core Features:</p> <ul style="list-style-type: none"> ● Basic Planner: Budget setting, itinerary creation, activity selection (free, limited to 3 plans) ● Advanced Planner: AI customization, cuisine preferences, accommodation matching (\$10/month) ● Social Features: Posts, stories, messaging, friend suggestions, community feed ● Discovery Features: Local recommendations, sponsored venue highlights <p>Product Development Strategy:</p> <ul style="list-style-type: none"> ● MVP launch with basic features ● Iterative improvement based on user feedback ● AI learning enhancement over time ● Regular feature updates and seasonal content 	<p>Travia operates as an online platform, accessible through:</p> <p>Primary Channels:</p> <ul style="list-style-type: none"> ● Google Play Store ● Direct website downloads ● Social media marketing and influencer partnerships ● Travel blog partnerships and content marketing <p>Distribution Strategy:</p> <ul style="list-style-type: none"> ● Global launch in English-speaking markets first ● Localization for key international markets (Europe, Australia) ● Partnership with travel agencies and tourism boards ● Integration with existing travel booking platforms

<p>Price</p> <p>The pricing strategy could follow:</p> <p>Freemium Strategy:</p> <ul style="list-style-type: none"> • Free tier: Basic planning tools (limited to 3 plans, 3 posts) • Premium tier: \$10/month or \$100/year • Pay per use: Travelers can pay for specific features like premium AI-generated plans. • Value proposition: Cost less than one restaurant meal for month of enhanced travel planning <p>Pricing Rationale:</p> <ul style="list-style-type: none"> • Competitive with fitness/productivity apps (\$9-15/month range) • Low enough to encourage trial, high enough to ensure quality users • Annual discount encourages longer commitment 	<p>Promotion</p> <p>Promotion strategies to increase awareness include:</p> <p>Digital Marketing Strategy:</p> <ul style="list-style-type: none"> • Content Marketing: Travel tips, destination guides, solo travel safety • Social Media: Instagram travel content, TikTok destination videos, Facebook community groups • Influencer Partnerships: Collaborate with solo travel bloggers and vloggers • App Store Optimization: Keyword optimization for "solo travel," "trip planner" • Referral Program: Free premium month for successful referrals <p>Launch Strategy:</p> <ul style="list-style-type: none"> • Beta testing with travel influencers and bloggers • PR campaign focusing on solo travel trend stories • Partnership announcements with initial restaurant/hotel sponsors • Contest and giveaway campaigns for early user acquisition
---	---

2.2.2.3 Marketing Strategy:

2.2.2.3.1 Brand Positioning

"TRAVIA: Where Solo Adventures Begin Together"

- Position as the essential companion for confident solo travelers
- Emphasize safety, connection, and personalized experiences
- Target modern, independent travelers who value both solitude and community

2.2.2.3.2 Customer Acquisition Strategy

Phase 1: Launch (Months 1-6)

- Target: 10,000 downloads, 1,000 active users
- Focus on travel influencer partnerships and PR
- Content marketing establishing thought leadership
- App store feature placement campaigns

Phase 2: Growth (Months 7-18)

- Target: 100,000 downloads, 15,000 active users, 1,500 premium subscribers
- Scaled digital advertising (Facebook, Instagram, Google)
- Partnership marketing with tourism boards
- User-generated content campaigns

Phase 3: Scale (Months 19+)

- Target: 500,000+ downloads, 75,000+ active users, 10,000+ premium subscribers
- International market expansion
- B2B partnership program
- Community-driven growth and referrals

2.2.2.3.3 Customer Retention Strategy

- Personalized push notifications about new destinations
- Seasonal travel inspiration and planning prompts
- Community challenges and travel badges
- Exclusive premium content and early access features
- Regular app updates with new AI recommendations

2.2.2.4 Business Model Canvas:

2.2.2.4.1 Key Partners

- **Technology Partners:** AI/ML service providers, cloud infrastructure (AWS, Google Cloud)
- **Content Partners:** Travel bloggers, local tourism boards, travel photographers
- **Business Partners:** Restaurants, hotels, activity providers, tour operators
- **Distribution Partners:** Travel agencies, corporate travel managers
- **Marketing Partners:** Travel influencers, tourism marketing agencies

2.2.2.4.2 Key Activities

- **Product Development:** AI algorithm improvement, feature development, user experience optimization
- **Community Management:** Content moderation, user engagement, customer support
- **Partnership Development:** Onboarding restaurants/hotels, negotiating sponsorship deals
- **Marketing & User Acquisition:** Digital marketing campaigns, content creation, PR activities
- **Data Analysis:** User behavior analysis, recommendation algorithm optimization

2.2.2.4.3 Key Resources

- **Technology Infrastructure:** AI recommendation engine, mobile app platform, cloud servers
- **Human Capital:** Development team, AI specialists, community managers, sales team
- **Data Assets:** User preference data, travel patterns, venue partnerships
- **Brand & IP:** Trademark, proprietary algorithms, user community
- **Financial Resources:** Investment capital, revenue streams, cash flow management

2.2.2.4.4 Value Propositions

2.2.2.4.4.1 For Solo Travelers:

- Personalized AI-driven travel recommendations
- Safe community of like-minded solo travelers
- Comprehensive trip planning in one platform
- Social connection without sacrificing independence
- Budget-friendly planning tools and exclusive deals

2.2.2.4.4.2 For Business Partners:

- Targeted marketing to high-value solo travelers
- Performance-based partnership model
- Direct access to engaged travel community
- Analytics and insights on customer preferences
- Co-marketing opportunities

2.2.2.4.5 Customer Relationships

- **Self-Service:** Intuitive app interface with minimal onboarding
- **Community-Driven:** User forums, peer recommendations, social features
- **Automated Assistance:** AI-powered recommendations and chatbot support
- **Personal Support:** Premium customer service for paid subscribers
- **Co-Creation:** User feedback integration, community-driven feature development

2.2.2.4.6 Channels

- **Digital Platforms:** Android app store
- **Direct Marketing:** Website, email campaigns, social media
- **Partnership Channels:** Travel blog integrations, tourism board collaborations
- **Word-of-Mouth:** Referral programs, user-generated content
- **Content Marketing:** SEO-optimized blog, travel guides, destination content

2.2.2.4.7 Customer Segments

2.2.2.4.7.1 Primary Segment: Tech-Savvy Solo Travelers (25-40)

- Disposable income of \$40,000+ annually
- Frequent social media users
- Value experiences over possessions
- Prioritize safety and authentic experiences

2.2.2.4.7.2 Secondary Segments:

- Digital nomads and remote workers
- Empty nesters exploring solo travel
- Business travelers extending trips
- Students and budget backpackers

2.2.2.4.8 Cost Structure

2.2.2.4.8.1 Technology Costs (40%)

- Development team salaries
- Cloud infrastructure and hosting
- AI/ML service fees
- App store fees (30% of subscription revenue)

2.2.2.4.8.2 Marketing & Sales (30%)

- Digital advertising spend
- Influencer partnerships
- Content creation
- Sales team compensation

2.2.2.4.8.3 Operations (20%)

- Customer support
- Community management
- Partnership management
- Legal and compliance

2.2.2.4.8.4 General & Administrative (10%)

- Office space and utilities
- Insurance and legal fees
- Accounting and financial services
- Executive compensation

2.2.2.4.9 Revenue Streams

2.2.2.4.9.1 Primary Streams:

- **Subscription Revenue (60%)**: \$10/month premium subscriptions
- **Sponsored Partnerships (35%)**: Restaurant, hotel, and activity sponsorships
- **Affiliate Revenue (5%)**: Booking commissions and travel service partnerships

2.2.2.4.9.2 Revenue Growth Strategy:

- Year 1: Focus on user acquisition and basic revenue
- Year 2: Scale partnerships and increase conversion rates
- Year 3+: International expansion and premium feature development

2.3 SURVEY:

We got 52 Responses from our survey, this will help us gain powerful insights on how people will preview the project.

Note that the sample chosen for this survey consist of mainly university students and people between the ages of 18-24 and approximately two thirds of them are of male gender.

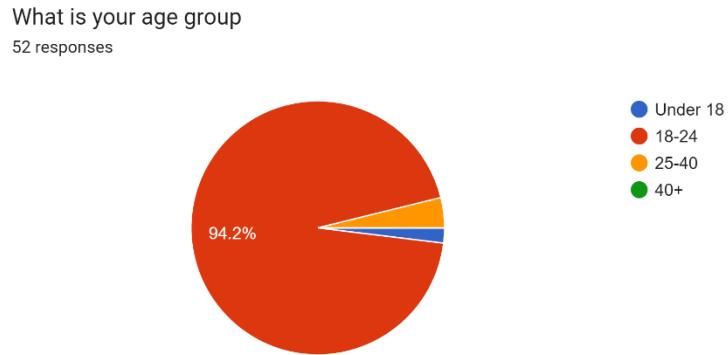


Figure 3 : Survey Question 1

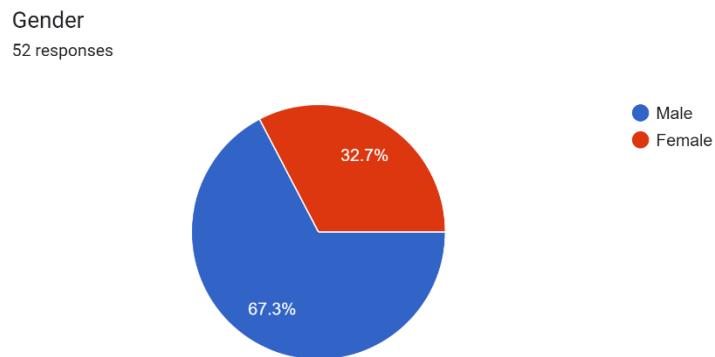


Figure 4 : Survey Question 2

The following figure shows the percentage of the sample's travelling habits.

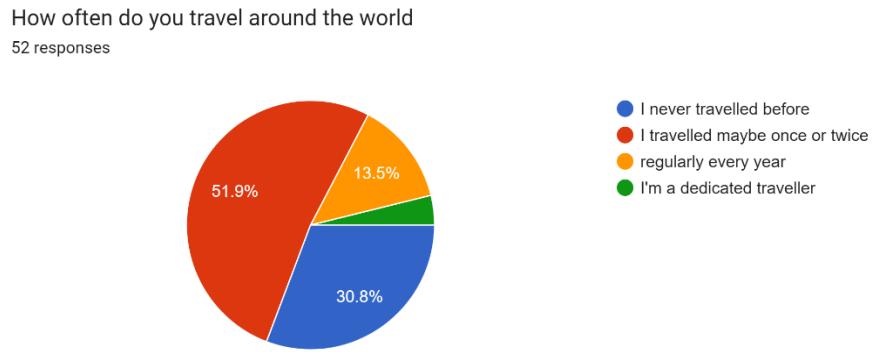


Figure 5 : Survey Question 3

Insights: We notice here that approx. 80% of the sample rarely travel or never do. This could hurt the popularity of our app among university students. One insight is university students and people of age 18-24 don't travel that much, possibly due to them being still young and unable to pay the costs of travelling. One last note that our app isn't tailored to travelers only, it could be used for outings in your own city.

The following two figures shows the percentage of the sample's way of finding recommendations and how successful are they:

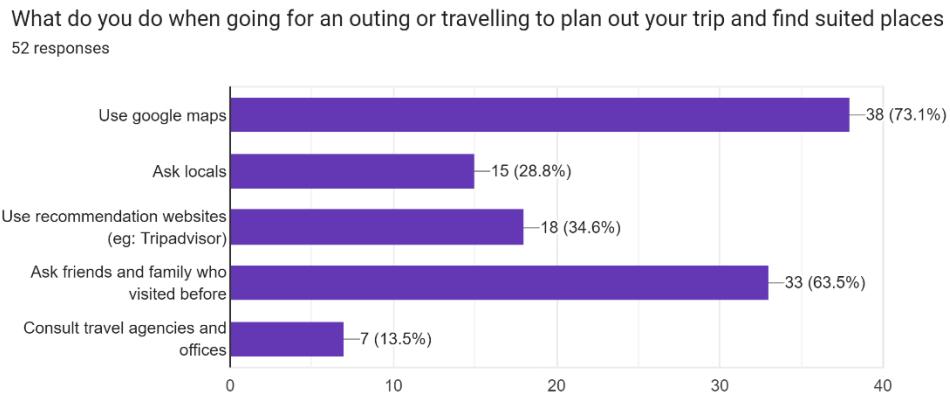


Figure 6 : Survey Question 4

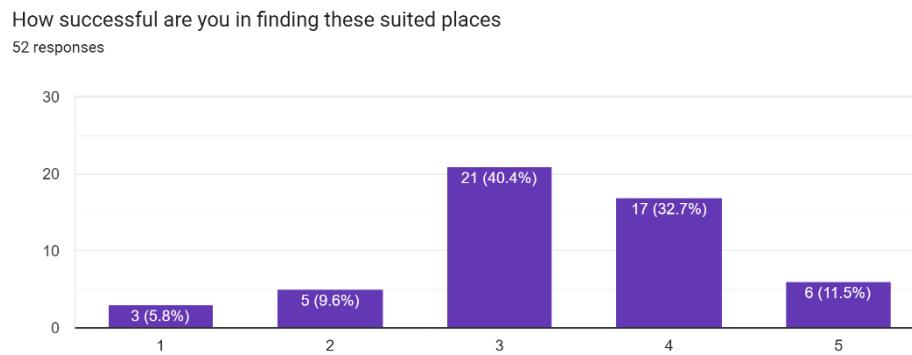


Figure 7 : Survey Question 5

Insights: The insight in the following is that people mainly use Google Maps and Ask relatives and friends for recommendations.

For the people that use Google Maps our app can vastly help ease the experience for them and find the best places possible.

And for the people who ask their friends or family that visited before, our app can help them find people who have experience in the city they want to visit.

Second figure shows an approximately normal curve of the sample finding the traditional methods for recommendations suited for them or not.

The following figure shows the percentage of the sample's ability to find suited partners for their travels and outings.

If you were to travel solo or go for an outing, how hard is it to find friends that have the same preferences as you

52 responses

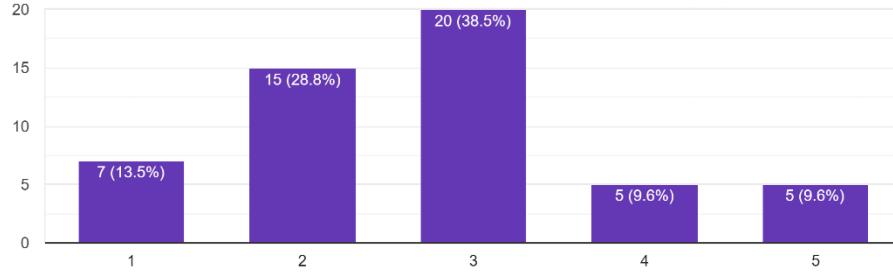


Figure 8 : Survey Question 6

Insights: Our app can help these people find their suited partner as most of the sample chose that they have a hard time of finding them.

The following figure shows a percentage of the sample's most important factors when planning a trip

What are the most important factors when planning a trip

52 responses

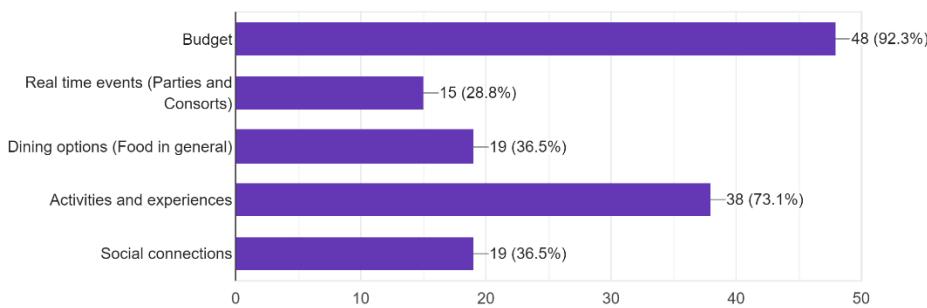


Figure 9 : Survey Question 7

Insights: This teaches us that most of the sample focus mainly on the budget and the activities are the most important factors that we need to consider.

The following figure shows us that the sample are comfortable with AI planning and managing their trip or outing.

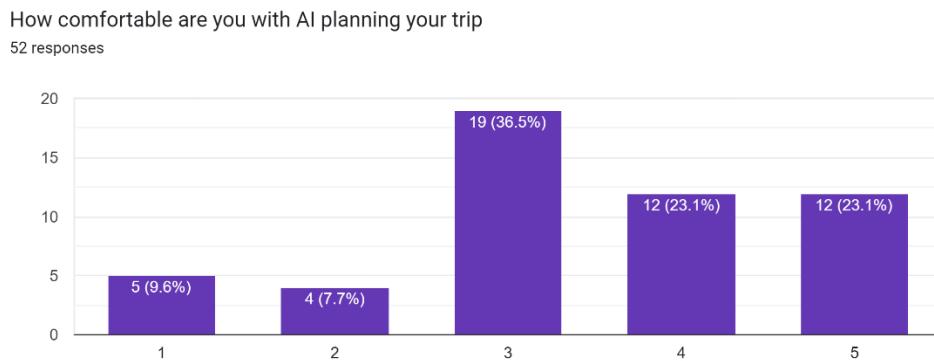


Figure 10 : Survey Question 8

Insights: The sample is mostly comfortable with AI planning their trip

The following figure shows us what are the biggest concerns of using our app.

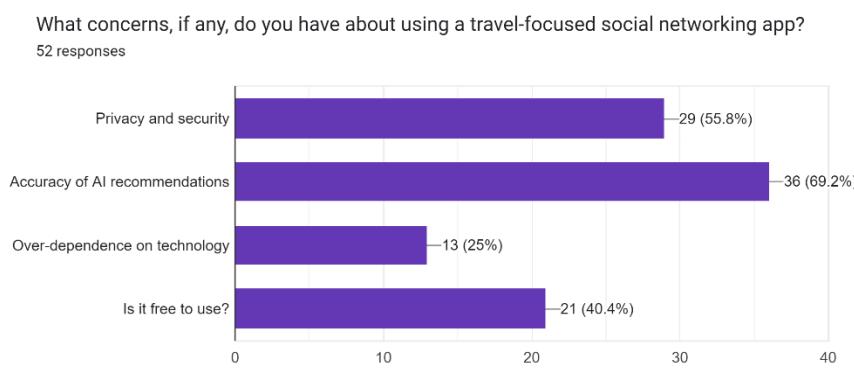


Figure 11 : Survey Question 9

Insights: this tells us that privacy, accuracy and pricing of the app are the most important factors.

2.4 Similar Projects

In this section, we will discuss the functionalities offered by the websites Tripplanner.ai, Wonderplan.ai, and Justasklayla.com as well as provide screenshots from within each website.



Figure 12 : Similar Projects

Figure 12 : From left to right: The logos of Tripplanner.ai, Wonderplan.ai, and Justasklayla.com

2.4.1 Tripplanner.ai

TripPlanner.ai is an AI-powered trip planning tool designed for creating detailed plans. It focuses on personalized travel suggestions and optimizing routes, but it does not offer social networking or real-time social engagement features.

TripPlanner.ai Strengths:

- Provides offline access through downloadable PDFs of itineraries.
- Focuses heavily on route optimization and detailed travel timelines.
- Includes robust AI recommendations for local cuisines and activities.

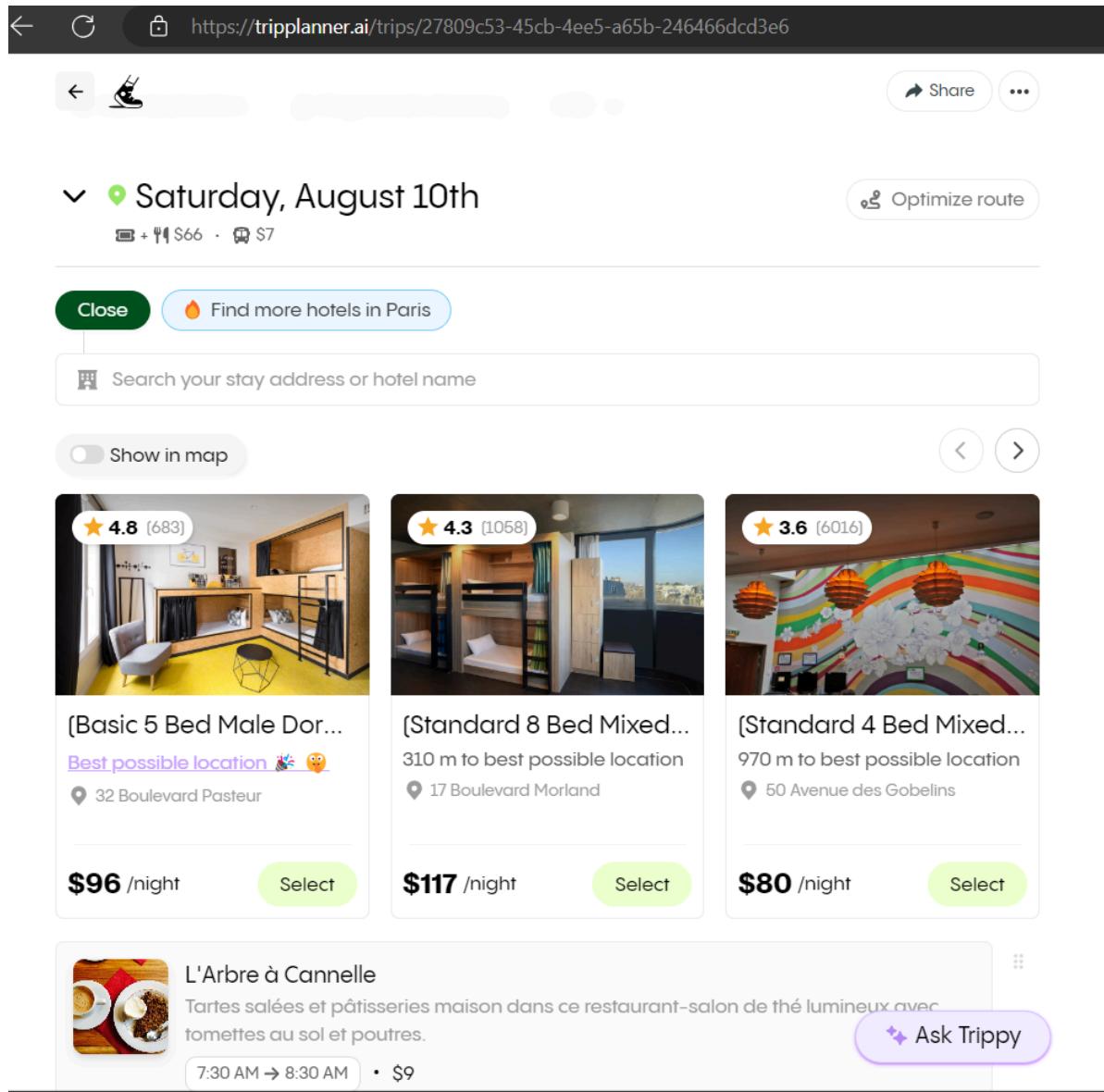


Figure 13 : Snapshot from Inside Tripplanner.ai

2.4.2 Wonderplan.ai

Wonderplan.ai is a comprehensive AI-driven trip planner that focuses on customizable plans tailored to user's preferences. It provides downloadable travel plans for offline use and a seamless user interface.

Wonderplan.ai Strengths:

- Offers robust itinerary customization with AI-generated suggestions.
- Includes offline plan access via downloadable PDFs.
- Allows users to quickly adjust plans based on budget or preferences.



▼ Day 2

Sun, 11 Aug



[Find a place to stay](#)

5

Santa Maria delle Grazie

Famous attraction for Leonardo da Vinci's 'The Last Supper' painting. Must-visit for art and history lovers.

● 120 min • Milan

6

Leonardo da Vinci Museum of Science and Technology

Explore the life and works of Leonardo da Vinci at this museum in Milan.

● 180 min • Milan

7

Pinacoteca di Brera

One of the most important art museums in Italy, with a collection of masterpieces from the 14th to the 20th century.

Figure 14 : Snapshot from Inside Wonderplan.ai

2.4.3 Justasklayla.com

Layla.ai is an interactive travel assistant leveraging AI chatbot technology for personalized trip planning. It simplifies the user experience with conversational planning but lacks comprehensive social or advanced customization features.

Layla.ai Strengths:

- Offers a highly interactive AI chatbot interface for quick and intuitive planning.
- Supports offline access with downloadable plans.
- Emphasizes ease of use and simplicity for users with basic travel needs.

Itinerary request

Hey, Layla here! Excited to help you with anything travel related. I can tell you where to go, what time of year to visit, what to do there...the list is endless. I'm on the edge of my seat, ask me anything.

build me an itinerary

Alright, let's get this party started! Do you have a specific destination in mind, or are you looking for some inspiration?

Beach destinations

Nice choice! Who doesn't love some sun, sand, and sea? How many days are you planning for this beach getaway?

Two weeks

Santorini looks amazing More suggestions please Love Bora Bora

give me the crime rate of every city

Itinerary

Day 1
Santorini

Day 2
Santorini

Day 3
Santorini

Discovering Akrotiri and Red Beach

Morning
Begin your second day with a visit to Akrotiri, an ancient Minoan city preserved under volcanic ash. Explore this fascinating archaeological site and learn about its history dating back to 1600 BC. For breakfast, stop by [The Good Heart](#) for some traditional Greek pastries and coffee before heading out.

Afternoon
After exploring Akrotiri, head to Red Beach nearby for some relaxation by the sea. The striking red cliffs create a unique backdrop against the turquoise waters. Spend some time swimming or sunbathing on this beautiful beach. For lunch, enjoy fresh seafood at [Taverna Glaros](#), located just steps away from Red Beach.

Evening
In the evening, return to Fira for more exploration. Visit [Santo Winery](#) where you can sample local wines while enjoying panoramic views of Santorini's vineyards and coastline. For dinner, savor authentic Greek dishes at [Naoussa Restaurant](#), known for its warm hospitality and delicious food.

Bedtime
Find amazing hotels in [Santorini](#).

For a balanced stay, [Andronis Boutique Hotel](#) choice offers a great mix of comfort and convenience. Those seeking an upscale experience can opt for the [Cellaria Estate Exclusive](#), featuring premium amenities and exceptional service. Budget-conscious visitors can find excellent value in a more affordable option like [Youth hostel Anna](#), providing basic comforts without breaking the bank.

Each type of accommodation ensures a pleasant stay, catering to different preferences and budgets.

Figure 15 : Snapshots from Inside Justasklayla.com

Table 4 : Competitors Comparison

Competitors → Features	Tripplanner.ai	JustAskLayla	Wonerplan.ai	Travia
Social Media	No	No	No	Yes
Tailored Recommendations	Limited	Yes	Yes	Yes
Real Time Events	No	No	No	Yes
Basic Planning Options	Yes	Yes	Yes	Yes
Multi Language Support	Yes	Yes	Yes	No
Advanced Planning Options	Limited	Limited	Limited	Yes
Customizable Budget	Yes	Yes	Yes	Yes
Ai Suggestions	Yes	Yes	Yes	Yes
Cross platform compatibility	Yes	Yes	Yes	Yes
Offers accommodations and tickets	Yes	No	Yes	Yes
Offline access to plans for later	Yes (PDF Download)	Yes (Custom Download)	Yes (PDF Download)	No

3 SYSTEM DESCRIPTION

TRAVIA is an essentially new System with several unique characteristics. **TRAVIA** consists primarily of a database as well as a mobile application serving as the interface.

3.1 Methodology

1. Gathering Data:

Explained in chapter 6

2. **Designing the mobile application:** We will design the mobile application using Figma, a mobile app prototyping tool, and ensure that the application meets key UI/UX design and HCI principles.
3. **Developing the mobile application:** We will develop the mobile application using flutter as to ensure compatibility and optimization with Android mobile phones.
4. **Testing the mobile application:** We will test the mobile through different types of application testing, which include but not limited to, performance, usability, compatibility, and functional testing.
5. **Integrating the system:** We will integrate all system components.
6. **Testing the system as a whole:** We will test the system as a whole to ensure all components are interconnected together and working flawlessly. This is done through performance, usability, compatibility, functional, and interruption testing.

3.2 Agile Methodology

We adopted a simplified Agile methodology to manage the iterative and evolving development of our TRAVIA application. Agile was chosen because it allows frequent feedback, continuous improvement, and flexibility to adjust to new ideas and changing requirements — especially important in a project that combines both travel planning and social networking features.

3.2.1 Key Agile Elements Used

3.2.1.1 Sprints

We divided the development into 4 major sprints, each lasting 2–3 weeks.

Each sprint focused on completing a set of functional requirements such as authentication, social media features, or planning modules.

3.2.1.2 Sprint Planning

At the start of each sprint, we selected the most important backlog items (features or fixes) to implement.

Tasks were assigned based on team members' roles and strengths (frontend/backend/database/AI).

3.2.1.3 Daily Communication (Stand-up Style)

While we didn't meet daily like in professional Agile, we communicated 2–3 times per week using:

WhatsApp for quick syncs

Google Meet/Zoom when needed for planning or debugging

3.2.1.4 Sprint Review

At the end of each sprint, we:

- Tested features
- Collected feedback from users/testers
- Demonstrated progress to our supervisor

3.2.1.5 Sprint Retrospective

We reflected on what worked, what didn't, and how we could improve in the next sprint.

3.2.1.6 Backlog and Prioritization

We maintained a Product Backlog using a shared Google Sheet.

We prioritized based on:

- Core app functionality
- Supervisor feedback
- Real user feedback from testing

3.2.1.7 Incremental Delivery

After each sprint, a functional version of the system was available.

This approach allowed us to test features in real scenarios and adapt quickly

3.2.1.8 Sprint Breakdown

Sprint	Main Goals
Sprint 1	Firebase Auth, Supabase setup, Sign Up/Login, User Profile
Sprint 2	Upload Post, Upload Story, Feed Display, Notifications
Sprint 3	Direct Messaging, Follow System, Explore by Country
Sprint 4	Basic/Advanced Planner, Real-Time Events, Final UI fixes

3.2.1.9 Benefits Gained from Using Agile

- Adapted quickly when feature priorities changed.
- Reduced delays by breaking the project into smaller parts.
- Delivered a real, working app within the project timeframe.
- Integrated AI and social features based on feedback and usability testing.

3.3 User Classes and Characteristics

Users of **TRAVIA** will be avid travelers and people looking for a holiday. We aim to help these people in their traveling through simplifying the processes related to traveling through our system.

Profiling the users:

- **Admin Users:**
 - Actively review system functionalities.
 - Review posts to ensure they meet the user guidelines standard.
 - Review feedback and reports submitted by users.
- **End Users:**
 - Avid Travelers
 - People looking for a holiday
 - People looking for like-minded individuals through our social media
 - People who want to go for an outing

3.4 Assumptions and Dependencies

- **Accuracy:** We assume our system functions accurately; we should perform comprehensive testing as well as software verification and validation to verify our assumption.
- **Integration and stability:** We assume our system is well integrated and functioning in a synchronized and consistent manner, we will also need to verify this through testing.

3.5 System Functional requirements overview

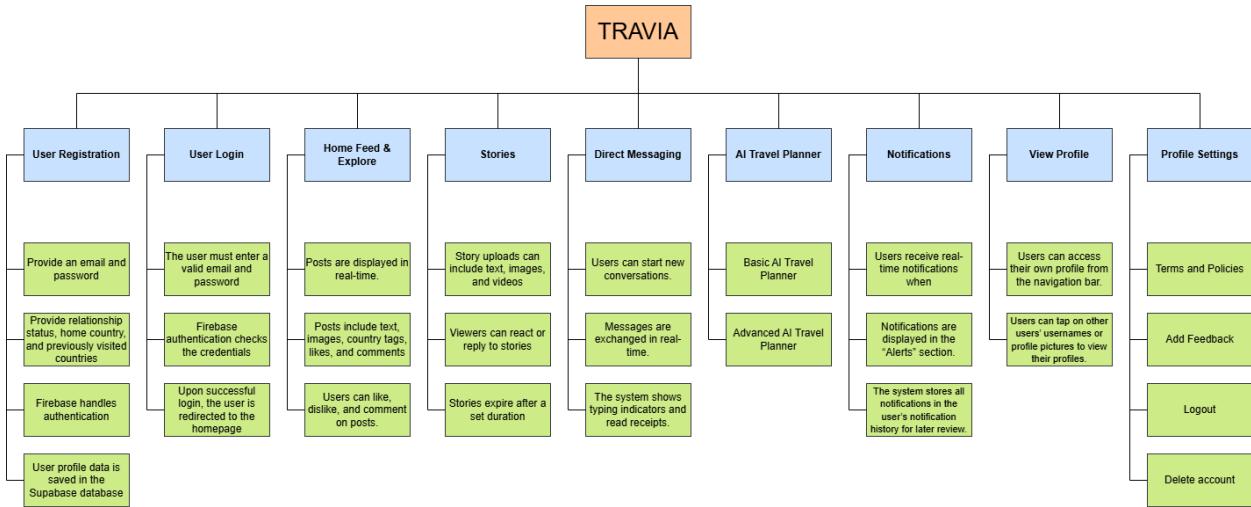


Figure 16 : System Functional Requirements

TRAVIA offers these basic functions:

- User Registration
- User Login
- Home Feed & Explore
- Stories
- Direct Messaging
- AI Travel Planner
- Notifications
- View Profile
- Profile Settings

3.6 Operating Environment

The interface component for **TRAVIA** will run as a mobile app with Android users as well as iPhone users.

3.7 Software Components:

We will mainly mention the software tools that will be used for the project.

3.7.1 Software tools overview:

- Fast API.
- Python.
- Supabase
- FireBase
- Android Studio
- Dart
- Flutter

3.7.2 Libraries:

- Numpy
- Pandas
- Sklearn
- Torch
- Blip
- YOLO
- Facebook/bart-large mnli ai

3.8 AI Technologies used

TRAVIA 2.0 integrates a sophisticated multi-modal AI architecture that combines reinforcement learning, content-based filtering, and social preference learning to deliver personalized travel recommendations. The system employs a hybrid approach designed to continuously learn from user interactions while maintaining recommendation diversity and accuracy.

3.8.1 Proximal Policy Optimization (PPO) Agent

3.8.1.1 Architecture Overview:

The core of TRAVIA's intelligent recommendation system is built on a PPO-based reinforcement learning agent that learns optimal recommendation policies through user feedback. The agent consists of two neural networks:

Actor Network (Policy Network):

- Input Layer: 22 features (combined user preferences and business attributes)
- Hidden Layers: Two layers with 128 neurons each, using ReLU activation
- Output Layer: 2 actions (like/dislike) with Softmax activation for probability distribution

Critic Network (Value Function):

- Input Layer: 22 features (same as actor)
- Hidden Layers: Two layers with 128 neurons each, using ReLU activation
- Output Layer: Single value output with linear activation

3.8.1.2 Learning Process

The PPO agent employs a sophisticated reward mechanism:

- **Base Rewards:** +2.0 for likes, -2.0 for dislikes
- **Cuisine Bonus:** +1.5 for preferred cuisine matches
- **Metadata Bonus:** +0.5 for ambience and attribute alignment
- **Training Epochs:** 5 epochs with PPO clipping ($\epsilon=0.3$) for stable learning

3.8.2 Content-Based Filtering System

3.8.2.1 Feature Engineering:

The system employs comprehensive feature extraction and engineering:

User Feature Vector (22 dimensions):

- Basic preferences (budget, travel days, accommodation type)
- Activity preferences (family-friendly, noise preference, ambience)
- Cuisine preferences (American, Italian, Mexican, Tapas, French, etc.)
- Social metadata preferences (romantic, classy, casual, good for kids)

Business Feature Vector (25 dimensions):

- Basic attributes (price range, ratings, review count)
- Business categorization (restaurant, cafe, bar, gym, shopping)
- Service capabilities (meal times, facilities, ambience)
- Encoded categorical features using one-hot encoding

3.8.2.2 Similarity Computation:

The content-based system uses cosine similarity and weighted feature matching to align user preferences with business characteristics, ensuring baseline compatibility before applying reinforcement learning optimization.

3.8.3 Social Preference Learning

3.8.3.1 Metadata Extraction Pipeline:

TRAVIA incorporates a novel social learning component that analyzes user interactions with social media content:

1. **Post Interaction Analysis:** Extract metadata from posts that users like/interact with
2. **Pattern Recognition:** Identify preferences for romantic venues, family-friendly locations, cuisine types
3. **Preference Scoring:** Calculate preference scores (0-1) based on interaction frequency
4. **Enhancement Boost:** Apply amplification for strong patterns ($>30\%$ frequency)
5. **Integration:** Incorporate learned preferences into the user feature vector

3.8.3.2 Metadata Attributes:

Binary attributes: romantic, good_for_kids, classy, casual

Temporal weighting: Recent interactions weighted more heavily

3.8.4 Hybrid Recommendation Algorithm

3.8.4.1 Multi-Modal Scoring System:

The recommendation engine employs a weighted combination approach:

- **RL Score (70% weight):** PPO agent prediction confidence
- **Content-Based Score (20% weight):** Feature similarity matching
- **Metadata Score (10% weight):** Social preference alignment

3.8.4.2 Balanced Variety Strategy:

To prevent recommendation echo chambers, the system implements a 70/30 strategy:

- **70% Preferred Businesses:** High-confidence matches based on learned preferences
- **30% Variety Businesses:** Diverse options to encourage exploration and discovery

3.8.5 Continuous Learning Framework

3.8.5.1 Training Pipeline

The system implements an adaptive learning pipeline:

- **Minimum Data Threshold:** Requires 2+ user interactions before training
- **Training Frequency:** Model updates after each feedback submission
- **Data Scope:** Uses last 50 interactions per user for training efficiency
- **Model Persistence:** Automatic saving/loading of trained models

3.8.5.2 Feedback Loop Integration

User Action → Database Storage → AI Processing → Model Update → Improved Recommendations

This creates a continuous improvement cycle where each user interaction enhances the system's ability to predict preferences accurately.

3.8.6 Technical Implementation

3.8.6.1 Deep Learning Framework:

PyTorch: Neural network implementation and training

Optimization: Adam optimizer with learning rate scheduling

Regularization: Gradient clipping and PPO clipping for stable training

3.8.6.2 Data Processing:

NumPy: Numerical computations and tensor operations

Pandas: Data manipulation and feature engineering

Scikit-learn: Text processing and feature extraction utilities

3.8.6.3 Performance Optimization:

Async Processing: Non-blocking model training and inference

Batch Processing: Efficient tensor operations for multiple recommendations

Model Caching: In-memory model storage for fast inference

Connection Pooling: Optimized database operations

3.8.7 Itinerary Generation Intelligence

3.8.7.1 Structured Planning Algorithm:

The AI system generates comprehensive daily itineraries:

- **Breakfast:** 2 AI-ranked options (cafes, breakfast restaurants)
- **Lunch:** 3 diverse options with cuisine variety
- **Dinner:** 3 higher-end evening options
- **Activities:** 1-4 options based on user preferences (gym, shopping, entertainment)
- **Dessert:** 2 options for sweet experiences

3.8.7.2 Contextual Awareness:

The recommendation engine considers:

- **Time-based preferences:** Different venues for different times of day
- **Location optimization:** Geographically logical itinerary flow
- **Preference diversity:** Balanced exposure to new experiences

3.8.2 AI Methodologies - Image Classification API

3.8.2.1 Overview

A multi-modal AI system that classifies venue attributes by combining three AI models: BLIP for scene understanding, YOLO for object detection, and BART for zero-shot classification.

3.8.2.2 Core AI Technologies

3.8.2.2.1 BLIP (Vision-Language Model)

- **Model:** Salesforce/blip-image-captioning-base
- **Purpose:** Generates natural language descriptions of images
- **Example Output:** "a restaurant with dim lighting and candles on tables"

3.8.2.2.2 YOLO (Object Detection)

- **Model:** YOLOv8n (You Only Look Once v8 - Nano variant)
- **Purpose:** Detects and counts specific objects in images
- **Example Output:** "contains 2 wine glasses, a candle, and 4 persons"
- **Why Nano:** Smallest YOLOv8 variant (3.2M parameters) for fast inference with minimal memory usage

3.8.2.2.3 BART-MNLI (Zero-Shot Classifier)

- **Model:** facebook/bart-large-mnli
- **Purpose:** Classifies text into categories without training
- **Accuracy:** 85-92% on unseen categories

3.8.2.3 Key Techniques

3.8.2.3.1 Multi-Modal Fusion

Combines three complementary information sources:

- **Visual Context** (BLIP): Overall scene understanding
- **Object Details** (YOLO): Specific items that indicate venue type
- **User Intent** (Caption): Context from the user's perspective

3.8.2.3.2 Zero-Shot Learning

- No training data required
- Uses Natural Language Inference (NLI) to understand semantic relationships
- Can classify into any category by understanding language meaning

3.8.2.3.3 Multi-Label Classification

- Each image can have multiple attributes
- Uses confidence thresholds (default: 0.5)
- Returns both boolean results and confidence scores

3.8.2.4 Classification Attributes

The system classifies venues into 4 attributes:

- ▀ **romantic**: Date-appropriate venues
- ▀ **good_for_kids**: Family-friendly locations
- ▀ **classy**: Upscale establishments
- ▀ **casual**: Relaxed, informal venues

4 Requirement Specification

Functional requirements define what a product must do and what its features and functions are. Nonfunctional requirements, also known as quality attributes, describe the general properties of a system. In this section we will thoroughly specify both our functional and nonfunctional requirements in detail.

4.1 System Features

4.1.1 User Registration

Description:

The user creates an account on TRAVIA to access the app's features.

Requirement(s):

- The user must provide an email and password during registration.
- Additional data is required: relationship status, home country, and previously visited countries.
- The User can Register via his gmail account.
- Authentication is handled via Firebase.
- User profile data is saved in the Supabase database.

4.1.2 User Login

Description:

The user logs into the TRAVIA app using previously registered credentials.

Requirement(s):

- The user must enter a valid email and password.
- Firebase authentication checks the credentials.
- Upon successful login, the user is redirected to the homepage.

4.1.3 Home Feed & Explore

Description:

Users can view all public posts from others around the world and interact with them.

Requirement(s):

- Posts are displayed in real-time.
- Posts include text, images, country tags, likes, and comments.
- Users can like, dislike, and comment on posts.
 - Comments can have replies on it and also replies on the other replies.
- Explore section supports:
 - country-based filtering (e.g., Explore Egypt, Spain).
 - Search for Specific post from its content type like (a person smiling, a fruit, a pet, etc.).
- “Recommended for You” section appears based on post scores (likes, comments, dislikes, etc.).

4.1.4 Stories

Description:

Users can upload and view temporary stories.

Requirement(s):

- Story uploads can include text, images, and videos.
- Viewers can react or reply to stories.
- Stories expire after a set duration.

4.1.5 Direct Messaging

Description:

Users can send and receive private messages with others.

Requirement(s):

- Users can start new conversations.
- Messages are exchanged in real-time.
- The system shows if the user is online or offline.
- The system shows typing indicators and read receipts.
- Users can discover and message new people through suggestions.
- Users can create a group including his friends in one chat.

4.1.6 AI Travel Planner:

Table 5 : Basic Plan VS Advanced Plan

Basic AI Travel Planner	Advanced AI Travel Planner
<p>Description</p> <p>Users choose the basic AI travel planner to plan their trip.</p> <p>Requirement(s)</p> <ul style="list-style-type: none"> ● Basic Planner uses previous user interactions and preferences ● Basic Planner allows manual input of preferences: <ul style="list-style-type: none"> ○ Category (GYM, bar or night life, shopping, beauty and health care) ○ Location ○ Budget ○ Days ● AI learns user interests through tags and post engagement patterns. ● Multiple plans are generated with like/dislike buttons for refinement. ● User Can: <ul style="list-style-type: none"> ○ Find the best or cheapest hotels. ○ Search for Flights. ○ Check Real Time Events. 	<p>Description</p> <p>Users choose the advanced AI travel planner to plan their trip.</p> <p>Requirement(s)</p> <ul style="list-style-type: none"> ● Advanced Planner uses previous user interactions and preferences. ● Advanced Planner allows manual input of preferences (Category, Location, budget, days, classy sites and locations, Noisy places, Family friendly, good for kids, Cuisine type (Italian, Mexican, Café, Fast Food, etc.)) ● MCDA algorithm selects matching businesses/places based on user input ● Reinforcement learning (RL) agent improves suggestions based on post interactions and feedback on plans. ● Multiple plans are generated with like/dislike buttons for refinement. ● AI learns user interests through tags and post engagement patterns. ● User Can: <ul style="list-style-type: none"> ○ Find the best or cheapest hotels. ○ Search for Flights. ○ Check Real Time Events.

4.1.7 Notifications

Description:

The system notifies users about relevant social interactions on the platform.

Requirement(s):

- Users receive real-time notifications when:
 - Someone likes or dislikes their post or story.
 - Someone comments or reply on the user's comments on a post.
 - Someone starts following them.
 - Someone report him.
 - They receive a new message.
- Notifications are displayed in the “Alerts” section.
- The system stores all notifications in the user's notification history for later review.

4.1.8 View Profile

Description:

Users can view their own profile and the profiles of other users on the platform.

Requirement(s):

- Users can access their own profile from the navigation bar.
- Users can view their:
 - Posts
 - Saved posts
 - Liked posts
 - Viewed posts
 - Follower and following counts
 - Badges/tags
- Users can tap on other users' usernames or profile pictures to view their profiles.
- Other users' profiles show:
 - Username
 - Bio
 - Posts
 - Tags
 - Liked posts
 - Follower and following counts

4.1.9 Profile Settings

Description:

Users can access a settings menu from their profile page to manage additional app-related actions.

Requirement(s):

- The settings icon opens a list with the following options:
 - Terms and Policies
 - Add Feedback
 - Logout
 - Delete account
- Selecting “Logout” signs the user out and returns them to the login screen.
- “Add Feedback” allows users to submit feedback.
- “Delete account” deletes the stored user account from the app and the database.
- “Terms and Policies” displays the app’s legal/privacy policies.

4.2 External Interface Requirements

4.2.1 User Interfaces:

We will mention here the Interfaces used in the app

4.2.1.1 Login/Registration Interface:

- Simple, intuitive forms for account creation and login.
- Profile setup with fields for travel-related preferences.

4.2.1.2 AI Planner Interface:

- Dynamic interface for inputting preferences and viewing results.
- Toggle options for switching between Basic and Advanced planning modes.

4.2.1.3 Social Networking Interface:

- Feed-style interface displaying updates, posts, stories and events.
- Alert section for notifications.
- Direct message section for chatting.
- User Profile (whether the account owner or others).

4.2.2 Hardware Interfaces:

1. Compatible with smartphones and tablets.
2. Minimum specifications:
 - Android 5.0 (API Version 21) or later.
 - iPhone 7 or above
 - Internet connectivity (Wi-Fi or mobile data) is always important, since the app relies on a “stream provider” architecture for real time updates.

4.2.3 Software Interfaces:

1. Firebase for authentication.
2. Supabase for database, storage, real-time channels.
3. Torch, Sklearn, Pandas and Numpy for AI functionalities.
4. Flutter and Dart for cross-platform development.

4.2.4 Communication Interfaces:

1. RESTful APIs for accessing third-party real-time data.
2. WebSocket or Firebase Realtime Database for instant updates.
3. HTTPS and TLS protocols for secure data transmission.

4.3 Design Constraints:

1. **Cross-Platform Compatibility:**
Uniform experience across Android iPhone using Flutter.
2. **AI Efficiency:**
Optimized algorithms to work seamlessly on devices with varying computational power.
3. **Real-Time Updates:**
Dependence on third-party APIs may affect performance during downtimes or rate limits.
4. **Budget Constraints:**
Development within allocated resources may limit the depth of features at initial release.

4.4 Nonfunctional Requirements

Table 6 : Non-Functional Requirements

Performance	<ul style="list-style-type: none"> ✓ System Uptime: TRAVIA Depends on SupaBase for its services. The SLA on supabase states that its availability is 99.99% ✓ Response Time: All user actions should result in feedback within 1 second. ✓ Scalability: Designed to handle growing user traffic with efficient resource utilization, Supabase can handle 100 thousand active users and can reach 1 Billion on Enterprise plans
Safety	<ul style="list-style-type: none"> ✓ Error Handling: <ul style="list-style-type: none"> • Graceful error Popups for failed operations (e.g., network issues or API errors), and
Security	<ul style="list-style-type: none"> ✓ User Authentication: <ul style="list-style-type: none"> • Enforced password policies ✓ Data Protection: <ul style="list-style-type: none"> • Compliance with GDPR • Other users can't access each other's data by assigning roles.
Usability	<ul style="list-style-type: none"> ✓ The platform should allow users to create a travel plan with a few clicks. ✓ The system should be accessible and intuitive for users with basic technical skills. ✓ The UI must be easy to understand and intuitive to users.
Reliability	<ul style="list-style-type: none"> ✓ High fault tolerance with mechanisms to recover from failures.
Accessibility	<ul style="list-style-type: none"> ✓ The system should be inclusive as to create a design suitable for colorblind through choosing neutral colors. Also use visuals to help people with dyslexia and/or low reading skills use the application.

Chapter Five

5 SYSTEM DESIGN

This section focuses on the systematic analysis and design of the TRAVIA application. It presents use cases, diagrams, and prototypes that define the system's architecture and functionality. The chapter details how user requirements are translated into actionable designs, ensuring the system's effectiveness, scalability, and user-centric approach.

5.1 USE CASE

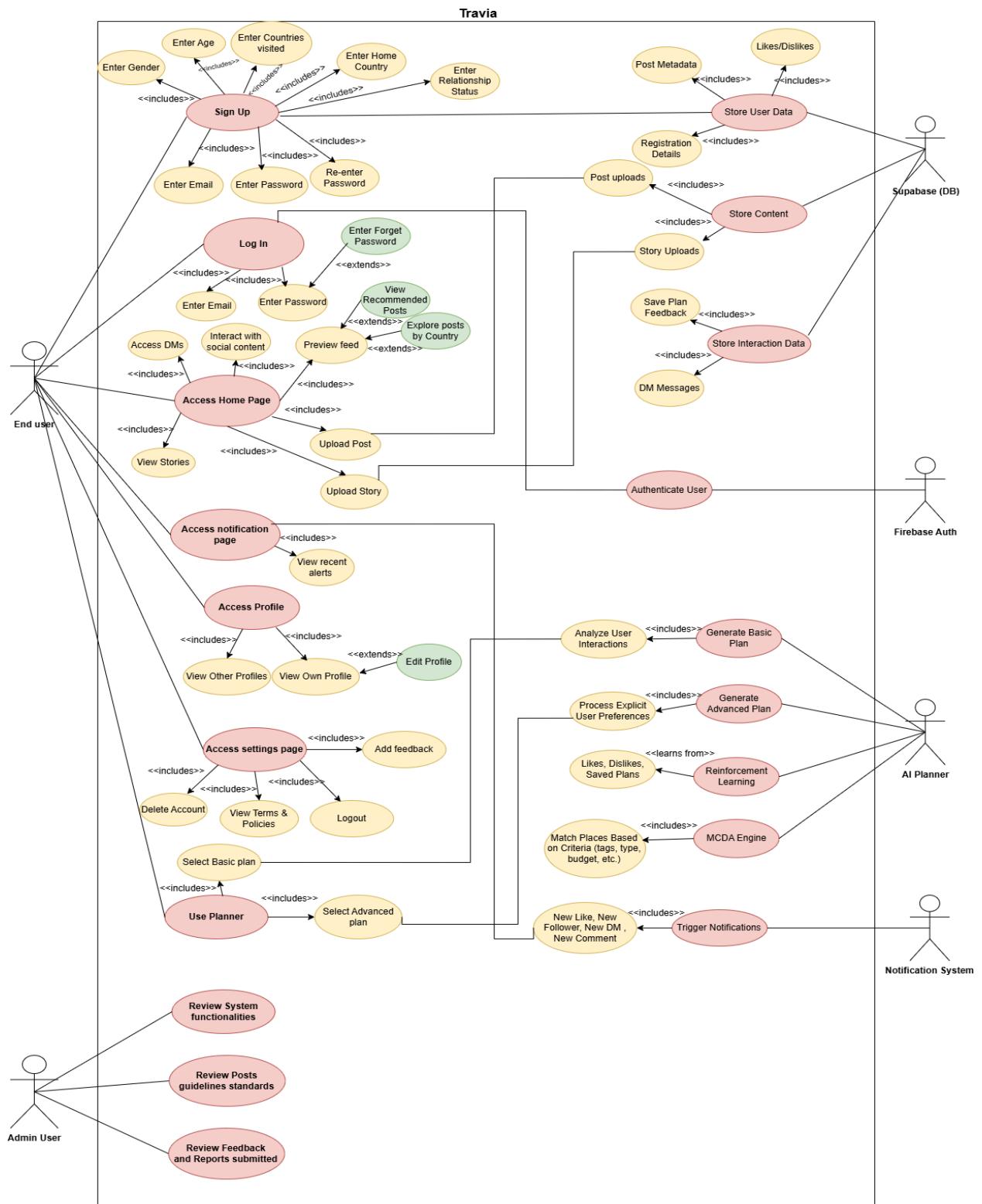


Figure 17 : Use Case Diagram

5.2 DETAILED USE CASES:

The detailed use cases offer expected actions and scenarios of a user using our application when performing specific tasks related to our functional requirements. These use cases are supplemented with sequence diagrams to show the interactions between objects in the sequential order that those interactions occur.

5.2.1 User Sign Up:

Table 7 : Sign Up Use Case

Title	User Sign Up
Primary actor	Traveler
Secondary actor	Firebase Auth, Supabase (DB)
Goal	To create a new user account and store user preferences.
Preconditions	The user has installed the TRAVIA app and launched it.
Postcondition(s)	The user is authenticated and their registration data is saved.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler opens the TRAVIA app. 2. Traveler selects Sign Up. 3. Traveler enters: <ul style="list-style-type: none"> o Email o Password & Re-enter Password 4. Traveler Chooses to Sign in with google account. 5. Traveler then gets a verification on the email he signed up with to confirm he owns it. 6. TRAVIA sends data to Firebase Authentication to register the email and password. 7. Firebase returns successful authentication. 8. Traveler goes to complete his profile by entering the following data: <ul style="list-style-type: none"> o Username o Display Name o Date of Birth o Gender o Relationship Status (Single, Married, Complicated) o Country of Residence o Previously Visited Countries 9. TRAVIA stores the rest of the data in Supabase (DB). 10. Traveler is redirected to the Home Page (logged in).
Alternative Flow	<ul style="list-style-type: none"> ● Email already registered: Firebase returns error : Show error: "This email is already in use." ● Weak password: Firebase returns error : Show hint: "Password should be at least 6 characters."

Exception(s)	<ul style="list-style-type: none"> -If the Traveler does not enter all required fields, the system displays an error. -If the email format is invalid, the system prompts the Traveler to correct it.

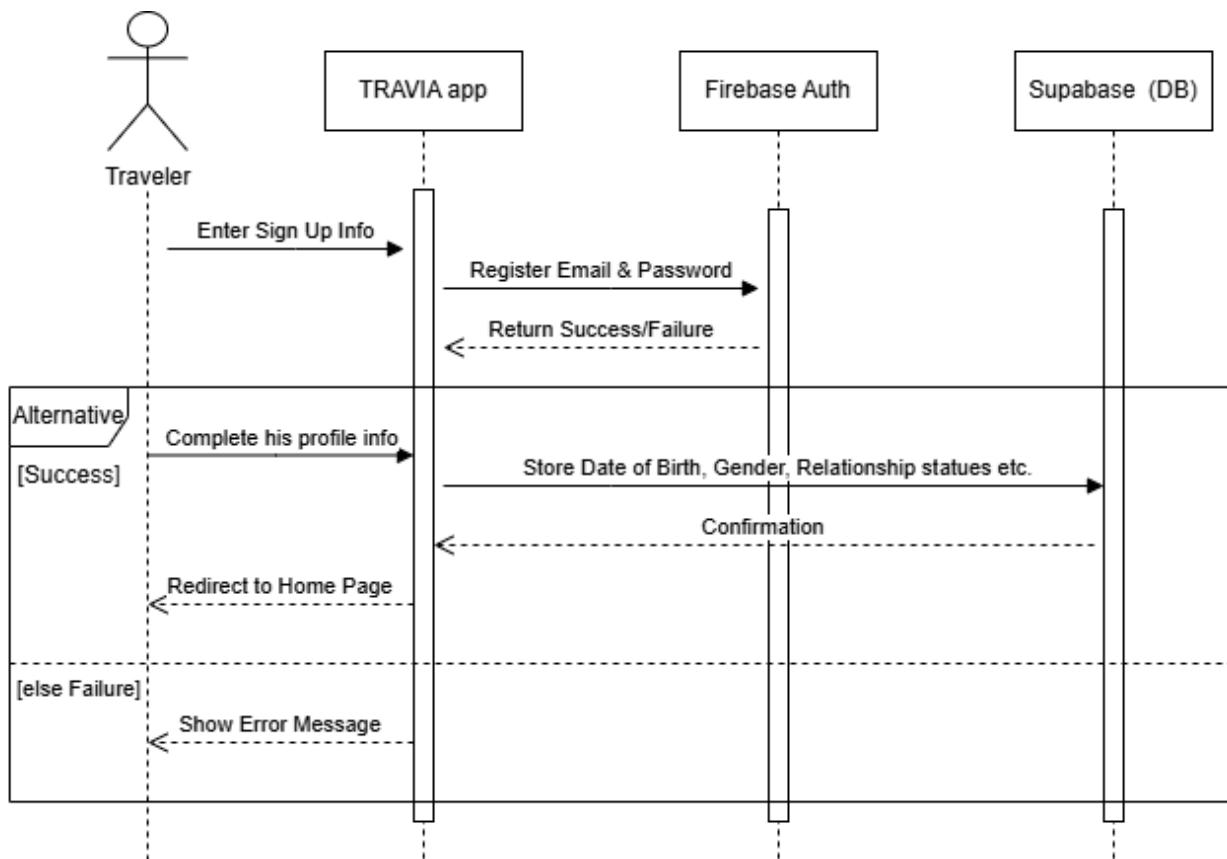


Figure 18 : User Sign Up

5.2.2 User Log in:

Table 8 : Login Use Case

Title	User Login
Primary actor	Traveler
Secondary actor	Firebase Auth
Goal	To authenticate an existing user and allow access to the app.
Precondition(s)	The user must have already created a TRAVIA account.
Postcondition(s)	The user is authenticated and redirected to the Home Page.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler opens the TRAVIA app. 2. Traveler selects Log In. 3. Traveler enters: <ul style="list-style-type: none"> o Email o Password 4. TRAVIA sends credentials to Firebase. 5. Firebase returns success (valid credentials). 6. Traveler is redirected to the Home Page.
Alternative Flow	<p>Forgot Password selected:</p> <ul style="list-style-type: none"> • Traveler clicks “Forgot Password?” link. • TRAVIA prompts for email. • Firebase sends password reset email to user. • Traveler receives confirmation.
Exception(s):	<ul style="list-style-type: none"> • Wrong email or password → Firebase returns error → Show “Incorrect email or password.” • Firebase service unavailable → Show error: “Unable to log in. Please try again later.”

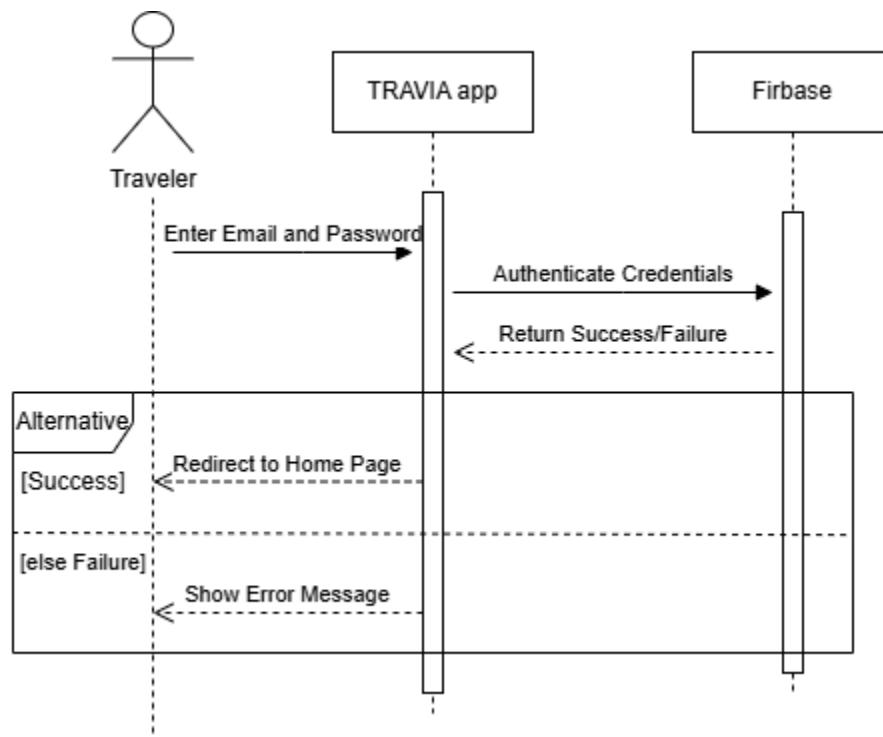


Figure 19 : User Log in

5.2.3 User Access Home Page:

Table 9 : Access Home Page Use Case

Title	Access Home Page
Primary actor	Traveler
Goal	To access the main interface of TRAVIA where the user can interact with content.
Precondition(s)	Traveler must be logged in successfully.
Postcondition(s)	Traveler views the home page feed with stories, posts, and recommendations.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler logs in and is redirected to the Home Page. 2. Home Page displays: <ul style="list-style-type: none"> o User Stories at the top. o Feed of Posts from different users. o Recommended Posts (based on post scoring algorithm). o Explore by Country posts. o Start Planning Trip option. 3. Traveler can: <ul style="list-style-type: none"> o View Stories. o Upload Story. o Like, Dislike, or Comment. o View Recommended Posts. o Explore posts by country. o View preview of the feed. o Go to dm. o Start planning a Trip. 4. The app records all interactions (likes, comments, etc.).
Alternative Flows	<ul style="list-style-type: none"> ● No stories or posts available: Show “No content to display” message or prompt user to follow others or upload. ● Network error: Show retry option or cached posts.
Exception(s):	<ul style="list-style-type: none"> ● Story or post fails to load → Show placeholder or retry icon. ● Upload failure → Notify user and allow retry.

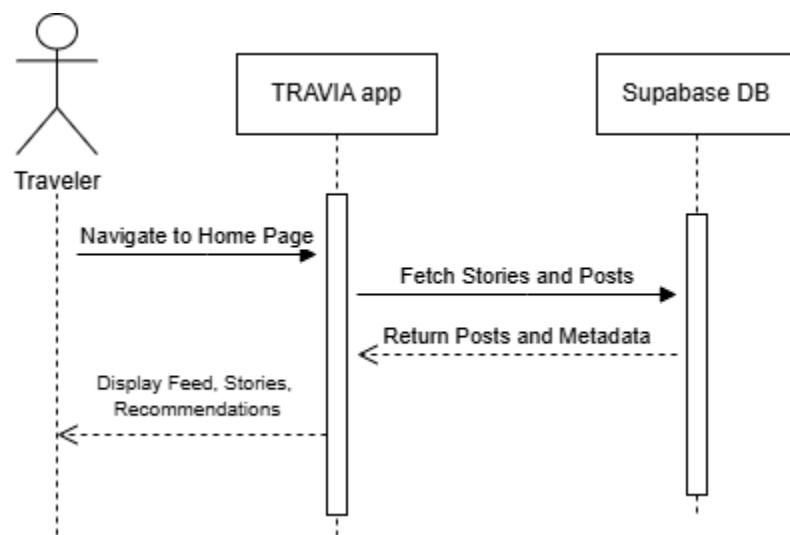


Figure 20 : User Access Home Page

5.2.4 User Uploads a post :

Table 10 : User Upload Post Use case

Title	Upload Post
Primary Actor	Traveler
Goal	To allow the traveler to share a new post on their profile which appears in the feed and explore section.
Precondition(s)	Traveler must be logged in and on the Home Page.
Postcondition(s)	The post is saved to the database and visible to followers and the public feed. Metadata is extracted for recommendation scoring.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler selects the upload post option from profile page 2. Travia asks user to access his device location to auto detect it. 3. Traveler chooses an image/video, writes a caption, and submits. 4. Travia allows users to choose only one media using single media picker. 5. Travia allows users for the media limit to 8Mb. 6. Traveler can edit image (e.g. Rotate, Flip, Reset). 7. Traveler can preview video before uploading it using video player. 8. Media gets sent to Inappropriate media recognition. 9. Media gets comprised. 10. The app extracts metadata from the caption (hashtags, keywords, etc.). 11. Media then gets sent to Image classification AI. 12. The post is uploaded and: <ul style="list-style-type: none"> o Stored in Supabase DB. o Metadata is associated with the post for interaction scoring. 13. Post becomes visible in: <ul style="list-style-type: none"> o Traveler's profile. o Follower feeds. o Explore by Country. o "Recommended for You" (if scoring is high).
Alternative Flows	<ul style="list-style-type: none"> ● Missing image or caption: Prompt user to provide required content. ● Poor internet connection: Retry option or store post temporarily.
Exception(s):	<ul style="list-style-type: none"> ● Upload fails (server error, file too large) → Show error message with retry. ● Invalid file type → Notify the user.

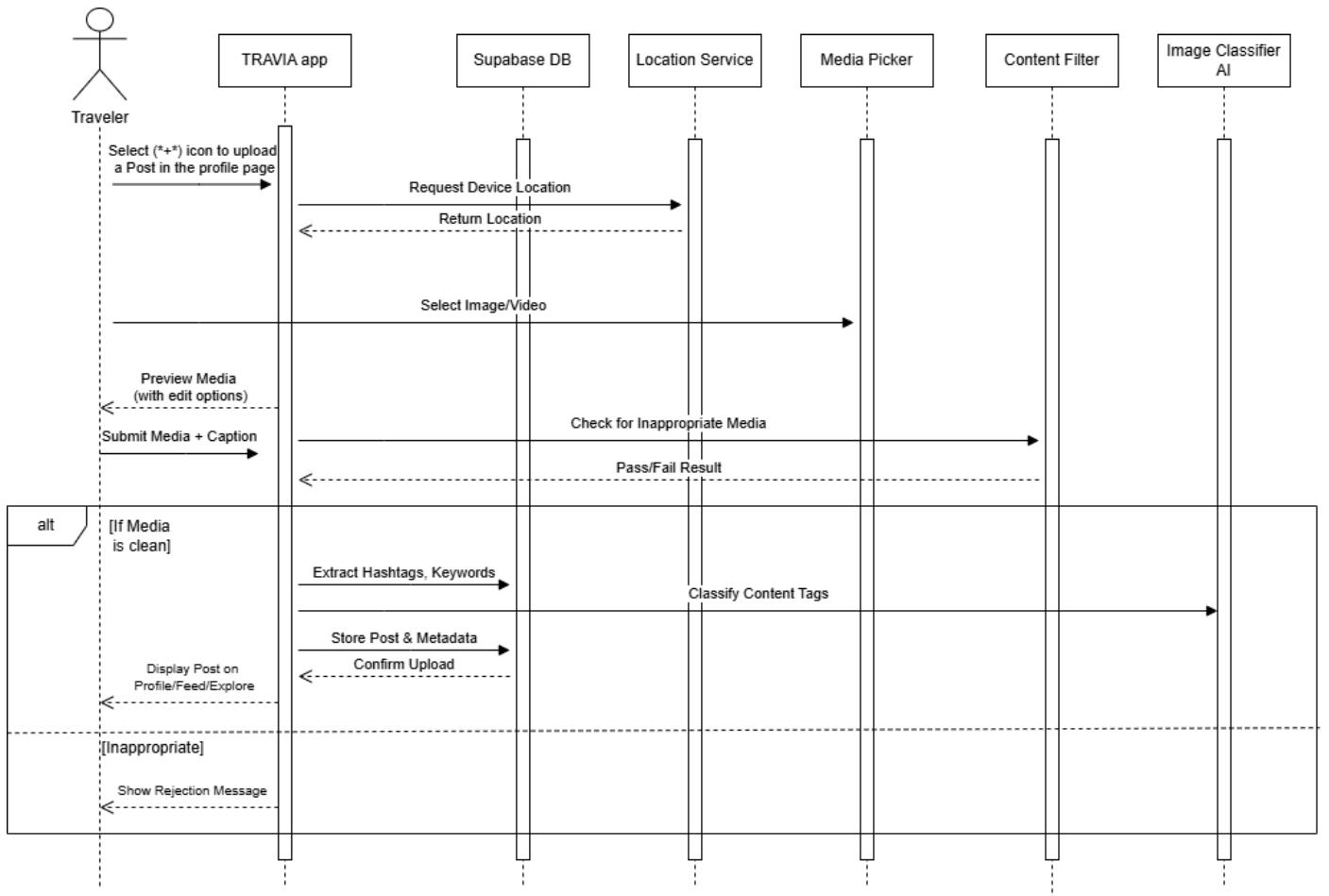


Figure 21 : User Upload a Post

5.2.5 User Uploads a Story:

Table 11 : User Upload a Story Use case

Title	Upload Story
Primary Actor	Traveler
Goal	To allow the traveler to share a temporary story (image/video) that appears for 24 hours.
Precondition(s)	Traveler must be logged in and on the Home Page.
Postcondition(s)	The story is uploaded and visible to followers via the story viewer. It expires after 24 hours.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler taps the “+” icon in the story section at Home Page. 2. Traveler selects media (image/video). 3. Travia allows user to select multiple media up to 5 media. 4. Traveler can edit the media he selected by the following options (paint, Text, Crop/Rotate, Tune option that has (brightness, contrast, etc..), Filter, Blur, Emoji). 5. Media gets sent to Inappropriate media recognition. 6. Travia allows users for the media limit to 8Mb. 7. Media gets compressed. 8. TRAVIA prepares and uploads the story content. 9. Story is stored in Supabase with a timestamp and expiration metadata. 10. Followers can now view the story from the Home Page story bar. 11. The story remains available for 24 hours.
Alternative Flows	<ol style="list-style-type: none"> 1. No media selected → Show error: “Please select a photo or video.” 2. Cancel during upload → Return to Home Page with no upload.
Exception(s):	<ol style="list-style-type: none"> 1. Upload failure due to poor network → Prompt retry or save draft. 2. Unsupported media type → Show error message.

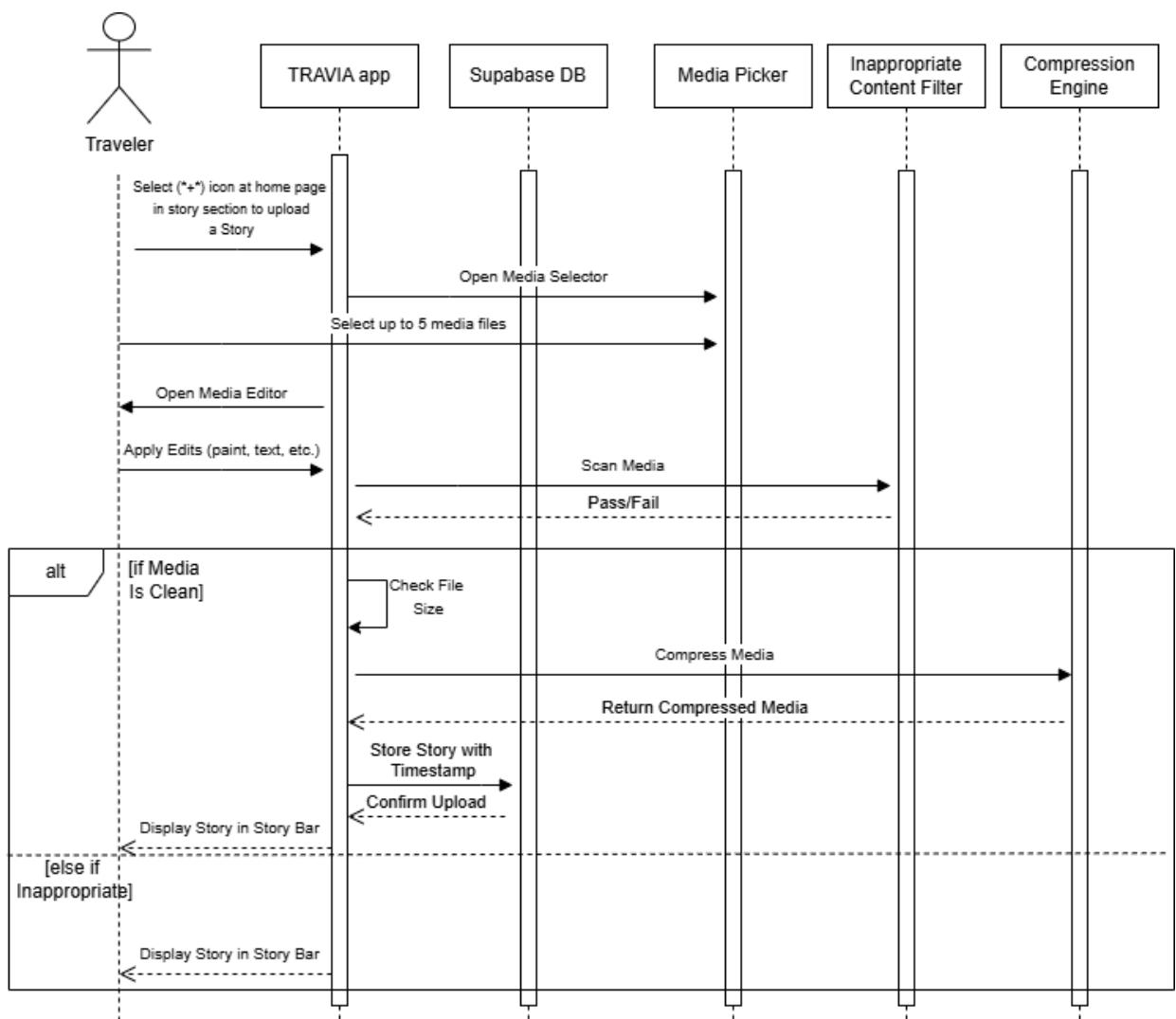


Figure 22 : User Upload a Story

5.2.6 User Send/Receive Messages:

Table 12 : User Send/Receive Messages

Title	Send/Receive Messages
Primary Actor	Traveler
Goal	To allow users to chat with each other in real-time through private messages.
Precondition(s)	Traveler must be logged in. The user must follow or find another user to start chatting.
Postcondition(s)	Messages are exchanged in real time and stored in the database for both users.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler opens the DM (Direct Messaging) section from the Home Page. 2. Traveler selects an existing chat or starts a new chat with another user. 3. Traveler can choose to create a group that contain multiple users. 4. Traveler will have the option of naming the group chat or to select the members he wants to include in the chat by searching for the other travelers. 5. Traveler can search for another user that doesn't exists in his followers list and start a new conversation with him. 6. Traveler can select an attachment image/video to send in the conversation chat with a limit of 5 media picker per message. 7. Traveler can record a voice message in the chat with the options of pausing the voice or deleting it or to swipe down to cancel the voice message. 8. Traveler types and sends a message. 9. Traveler can choose a specific message in chat to reply on it. 10. The message is sent to Supabase DB. 11. The receiver gets the message instantly in their inbox (real-time update). 12. Read receipts or typing indicators are displayed. 13. Traveler can see whether the member in the conversation is online or if he is offline. 14. Both users can continue the conversation.
Alternative Flows	<ul style="list-style-type: none"> ● Blocked user: If the recipient has blocked the traveler, the message is not sent. ● User is offline: Message is delivered when the user comes back online.
Exception(s):	<ul style="list-style-type: none"> ● If user goes offline → store messages locally and sync when reconnected.

- If media exceeds size/limit → show error: “*You can only send up to 5 items per message.*”
- If voice recording fails → show “*Voice message failed. Please retry.*”
- If recipient no longer exists → show “*User not found or has deactivated their account.*”
- If the **referenced message was deleted** before sending the reply → Show: “*This message is no longer available. You can't reply to it.*”

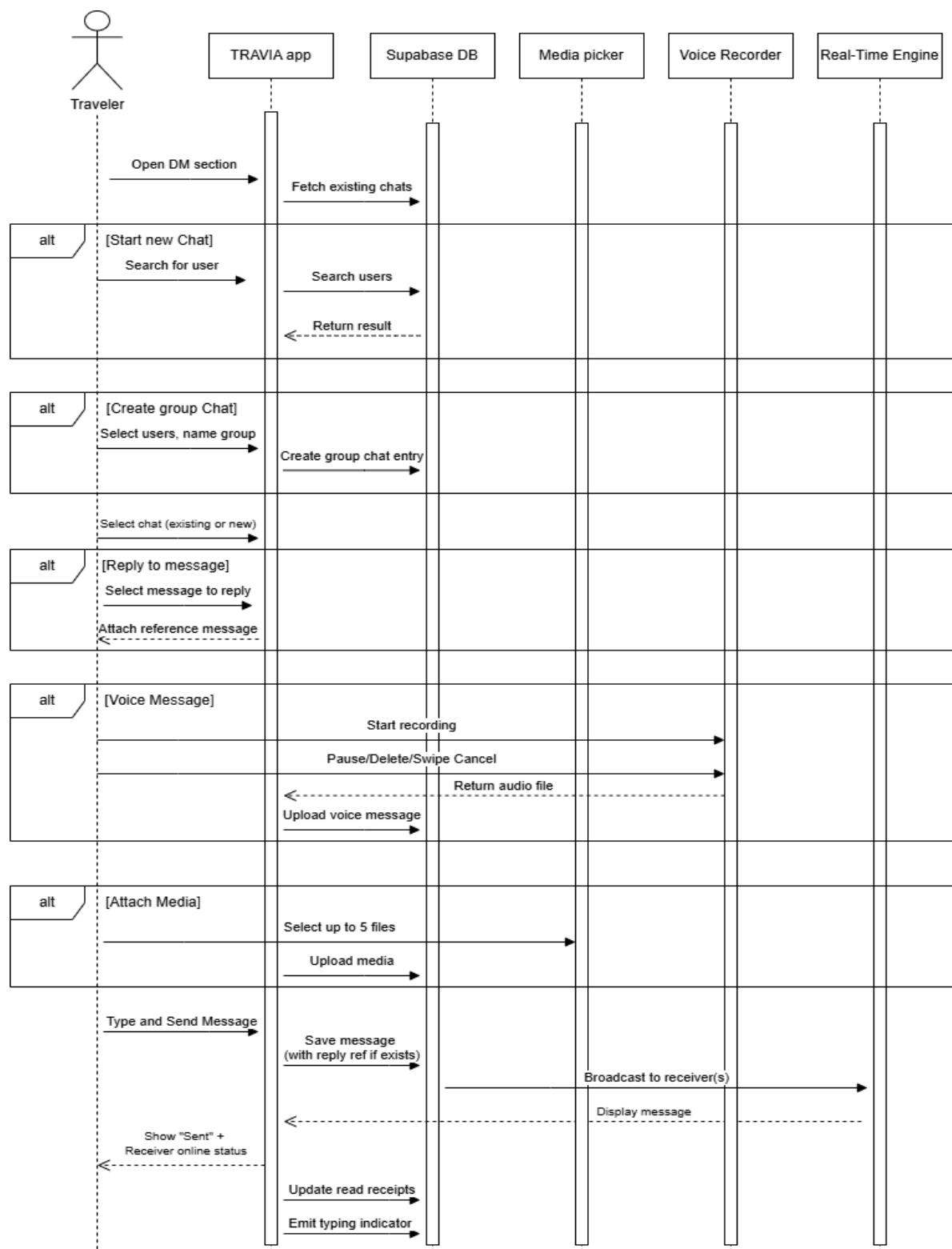


Figure 23 : User Send receive messages

5.2.7 User Like / Dislike / Comment on Post:

Table 13 : User Like / Dislike / Comment on Post

Title	Like / Dislike / Comment on Post.
Primary Actor	Traveler
Goal	To allow travelers to engage with posts by liking, disliking, or commenting.
Precondition(s)	Traveler must be logged in and viewing the Home Page or another user's post.
Postcondition(s)	The action is recorded, and the post's score is updated accordingly in the system.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler views a post from the feed or another user's profile. 2. Traveler chooses to: <ol style="list-style-type: none"> a. Like the post b. Dislike the post c. Comment on the post d. Share the post e. Save the post 3. Traveler in each comment can: <ol style="list-style-type: none"> a. Like the comment b. Reply on the comment c. Report the comment 4. The action is sent to Supabase. 5. The system: <ol style="list-style-type: none"> a. Updates the post's interaction count and metadata. b. Saves the comment and links it to the post and user. 6. The post owner receives a notification of the interaction.
Alternative Flows	<ul style="list-style-type: none"> ● User taps Like/Dislike again → Toggle state (remove previous reaction). ● Comment is empty or too long → Show validation message. ● Comment violates terms and policies → prevent user from sending it.

Exception(s):

- If user is **offline**, queue actions locally and sync when online.
- If Supabase write fails, show: "*Action failed. Try again.*"
- If user tries to **like and dislike a post at the same time**, show: "*Only one reaction is allowed.*"
- If comment is **reported multiple times**, flag it for moderation.

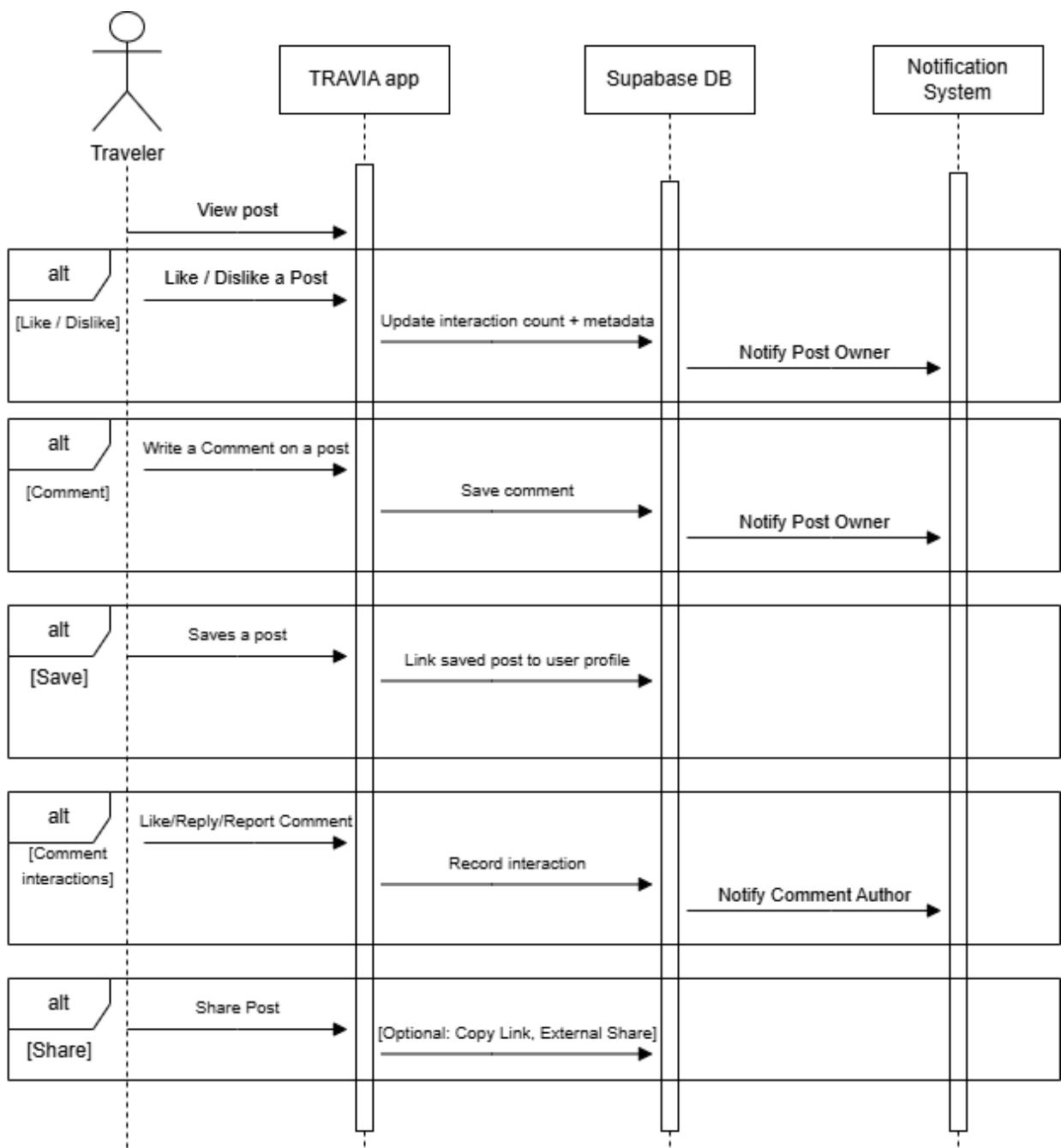


Figure 24 : User Like/Dislike/Comment on a post

5.2.8 User View/Edit Own Profile:

Table 14 : User View/Edit Own Profile

Title	View/Edit Own Profile
Primary Actor	Traveler
Goal	To allow users to view their own profile details and make edits if needed.
Precondition(s)	Traveler must be logged in.
Postcondition	Profile data is displayed and optionally updated in the database.
Basic Flow of events (View Profile)	<ol style="list-style-type: none"> 1. Traveler taps their profile icon from the navigation bar. 2. TRAVIA fetches user data from Supabase: <ul style="list-style-type: none"> o Profile picture o Followers / Following count o Display name o Username o Visited countries o Bio o Gender / age / relationship status o Badges gained by the user for his interaction o Posts, liked, saved 3. The app displays this data in the profile screen. 4. Traveler can gain badges like: <ol style="list-style-type: none"> a. Founder b. Poster c. Adventurer d. Trendy And more and more badges to gain to increase user interaction. 5. Traveler can scroll through their posts or switch between tabs (Saved, Liked, etc.).
Basic Flow of events (Edit Profile)	<ol style="list-style-type: none"> 1. Traveler taps the Edit Profile button. 2. Traveler updates any of the following: <ul style="list-style-type: none"> o Profile picture o Display Name o Username o Bio o Date of Birth

	<ul style="list-style-type: none"> o Gender o Relationship status o Visited countries o Radio button to show liked post or don't <p>3. TRAVIA sends the updated data to Supabase.</p> <p>4. Traveler sees a confirmation message.</p>
Alternative Flows	<ul style="list-style-type: none"> ● No internet connection → Show cached profile data and retry editing later. ● Edit canceled → Discard unsaved changes.
Exception(s):	<ul style="list-style-type: none"> ● If profile fetch fails → Show error: "<i>Unable to load profile. Try again later.</i>" ● If DB update fails during edit → Show: "<i>Update failed. Please try again.</i>" ● If invalid username or empty field → Show input error ● If traveler tries to update to a username already taken → Show: "<i>This username is already in use.</i>"

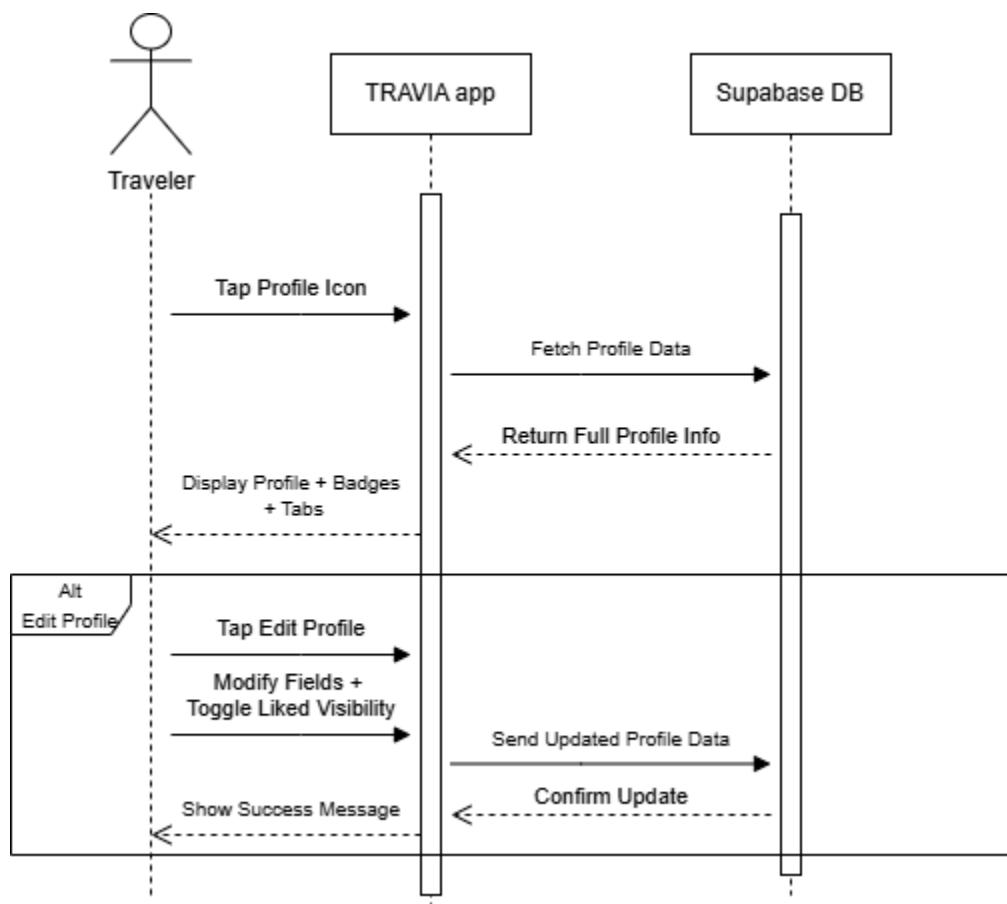


Figure 25 : View/Edit Own Profile

5.2.9 User View Other Profile:

Table 15 : User View Other profile

Title	View Other Profiles
Primary Actor	Traveler
Goal	To allow users to view the public profile of other users, including their posts and related info.
Precondition(s)	Traveler must be logged in and tap another user's name or avatar from a post or message.
Postcondition	The selected user's profile is displayed with public data and posts.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler taps on a username or profile picture (e.g., from a post or DM). 2. TRAVIA requests that user's public profile data from Supabase: <ol style="list-style-type: none"> a. Username b. Followers / Following counts c. Display name d. Profile picture e. Visited countries f. Bio g. Gender / Age / Relation ship status h. Badges i. Posts / liked 3. The app displays the selected user's profile. 4. Traveler can: <ol style="list-style-type: none"> a. View their posts b. Follow/unfollow them c. View Follower or Following list d. Initiate DM from the profile e. Block or Report User

Alternative Flows	<ul style="list-style-type: none"> User deleted their account: Show “User not found” message.
Exception(s):	<ul style="list-style-type: none"> Failed to load profile → Show retry or “profile unavailable” message. Network delay → Show loading placeholder/spinner.

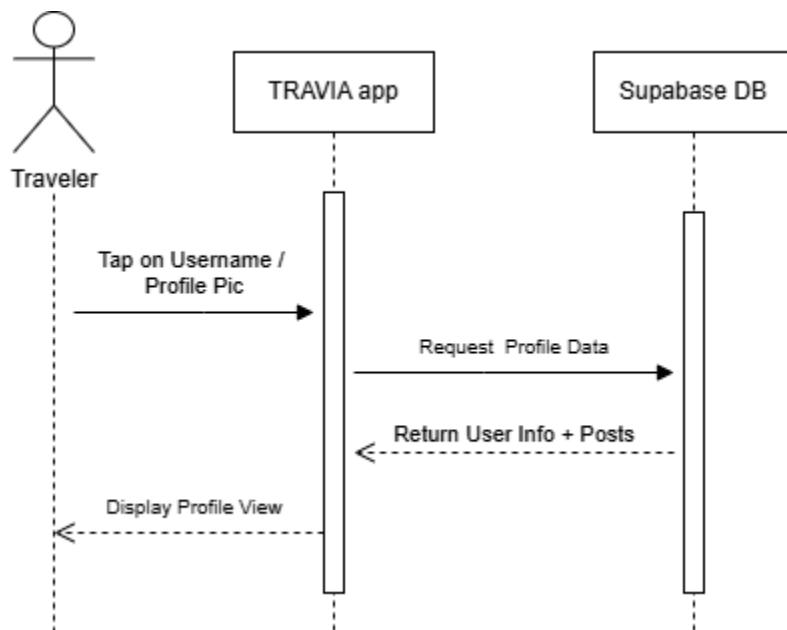


Figure 26 : User view other profile

5.2.10 User Access Settings Page:

Table 16 : User Access Settings Page

Title	Access Settings Page
Primary Actor	Traveler
Goal	To allow the traveler to manage additional options such as feedback, support, terms, and logout.
Precondition(s)	Traveler must be logged in and on their own profile page.
Postcondition	Appropriate action is performed based on the selected setting (e.g., logout, submit feedback, open external page).
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler opens their Profile Page. 2. Traveler taps the Settings (gear icon). 3. TRAVIA displays a list of settings options: <ol style="list-style-type: none"> a. Terms and Policies b. Add Feedback c. Logout d. Delete account 4. Traveler selects an option: <ol style="list-style-type: none"> a. Terms and Policies: Opens internal modal with legal content. b. Add Feedback: Opens a form to submit feedback. c. Logout: Logs the user out and returns to the login page. d. Delete account: Deletes User Account.
Alternative Flows	<ul style="list-style-type: none"> ● Back without action: Traveler returns to profile page without selecting anything. ● Invalid external link: Show error if support/feedback page fails to load.
Exception(s):	<ul style="list-style-type: none"> ● Logout fails due to Firebase session issue → Show error and retry option. ● Feedback form submission fails → Show offline save or retry option.

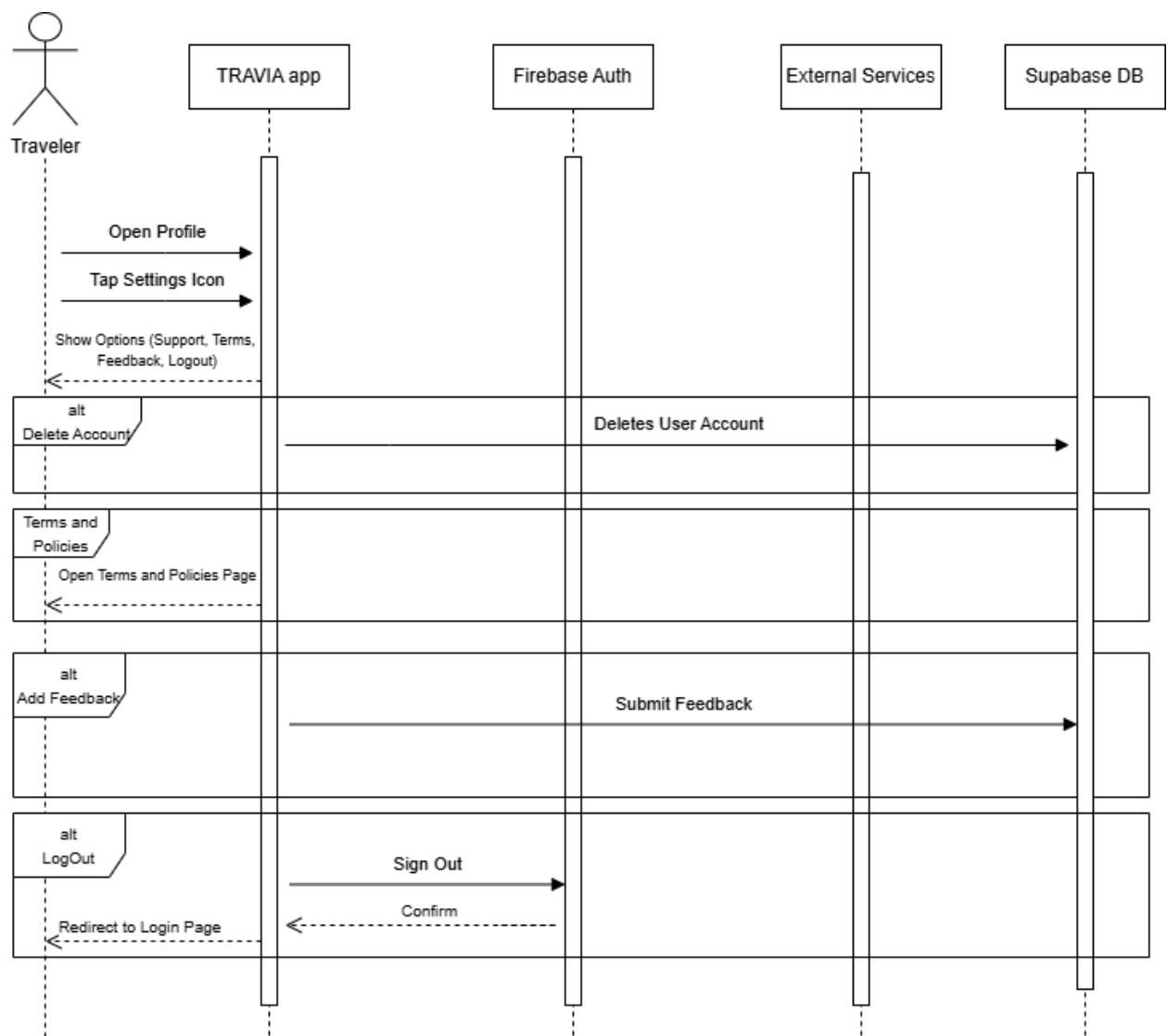


Figure 27 : User access settings page

5.2.11 User View Notifications page:

Table 17 :User View Notifications

Title	View Notifications page.
Primary Actor	Traveler
Secondary Actor	Notification System, Supabase
Goal	To allow the traveler to view social notifications such as likes, follows, and messages.
Precondition(s)	<ul style="list-style-type: none"> ● Traveler is logged in. ● Notification data exists in the database (e.g., from previous likes, follows, messages). ● Internet connection is available.
Postcondition	<ul style="list-style-type: none"> ● Notifications are displayed to the traveler. ● Opened notifications are marked as “read.” ● Redirect occurs if the traveler taps on a notification (e.g., to post, message, profile).
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler taps the Notification bell icon from the navigation bar. 2. TRAVIA app requests all unread and recent notifications from Supabase. 3. System displays a list of notifications: <ul style="list-style-type: none"> ○ [User] liked your post ○ [User] started following you ○ New message from [User] 4. Traveler can tap a notification to navigate to: <ul style="list-style-type: none"> ○ Post (if liked/disliked) ○ Chat (if message) ○ User profile (if follow) 5. Notifications are marked as “read.”
Alternative Flows	<ul style="list-style-type: none"> ● Traveler taps “Mark all as read” → all notifications update visually and in database.

Exception(s):

- If internet is disconnected → Show: “*Unable to load notifications. Please check your connection.*”
- If Supabase fails to respond → Show: “*Something went wrong. Try again later.*”
- If referenced content (post/message/user) was deleted → Show fallback: “*This content is no longer available.*”
- App shows “Check your internet connection” with a retry button.

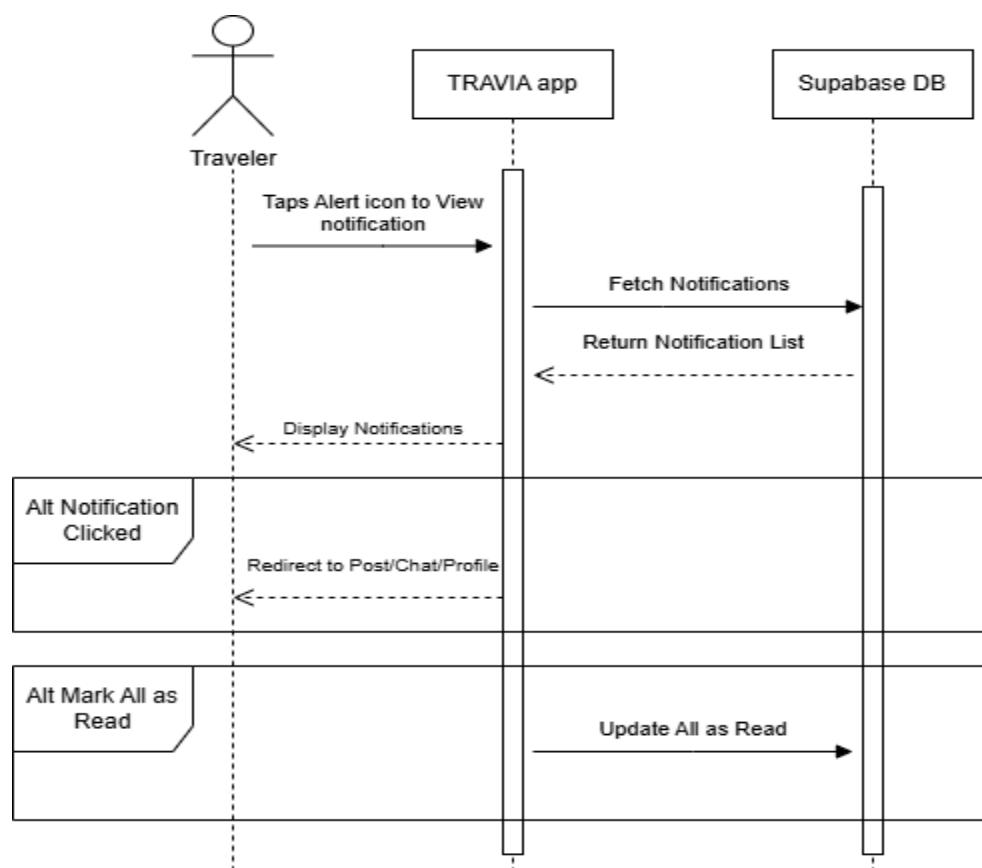


Figure 28 : User View Notifications

5.2.12 Use Basic Planner:

Table 18 : Use Basic Planner

Title	Use Basic Planner.
Primary Actor	Traveler
Goal	The traveler selects the Basic AI Planner to generate a personalized trip using previous social media interactions and optionally enters preferences such as destination, budget, and days, activities(gyms, bars, shopping, etc....).
Precondition(s)	<ul style="list-style-type: none"> ● Traveler is logged into the system. ● Traveler has interacted with posts (or can manually input preferences).
Postcondition	The user receives a trip plan based on preferences and interaction data.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler navigates to the Planner page. 2. Traveler selects the Basic Planner option. 3. System checks for previous interaction data (likes, post engagement). 4. Traveler optionally enters: <ol style="list-style-type: none"> a. Preferred Activities (e.g., gym, shop, night club) b. Location to visit c. Budget d. Number of days 5. System retrieves post metadata (tags, location, type). 6. AI Planner uses tag matching and interaction patterns to score places. 7. The system generates a trip plan with recommended places. 8. Traveler views the plan. 9. Traveler can : <ol style="list-style-type: none"> a. Like or dislike suggested plans. b. Can view the suggested Activities info: <ol style="list-style-type: none"> i. Call the place. ii. Visits the place website. iii. View the directions to the places through google maps. c. Find the best or cheapest Hotels. d. Search for deals on flight reservations.

	<p>e. Check the Real Time Events happening the plan location.</p> <p>10. The system stores feedback for future learning.</p> <p>11. Traveler can share his saved plans through DM and he can edit the plan name.</p> <p>12. The trip plan is saved in Supabase.</p>
Alternative Flow – Cold Start, If no interaction data exists:	<ul style="list-style-type: none"> ● The planner depends only on the manually entered preferences to generate the plan.
Exception(s):	<ul style="list-style-type: none"> ● If the user has no internet connection, the system displays an error message: <i>"Connection lost. Please try again later."</i> ● If the system fails to find relevant post metadata or tags, it shows a fallback message: <i>"No matches found. Try changing your preferences."</i> ● If the AI planner service is unavailable, the user is alerted with: <i>"We're unable to generate a plan right now. Try again soon."</i> ● If the system fails to fetch data from the database due to timeout or error, it retries once. If the retry fails, it shows: <i>"Something went wrong. Please restart the planner."</i>

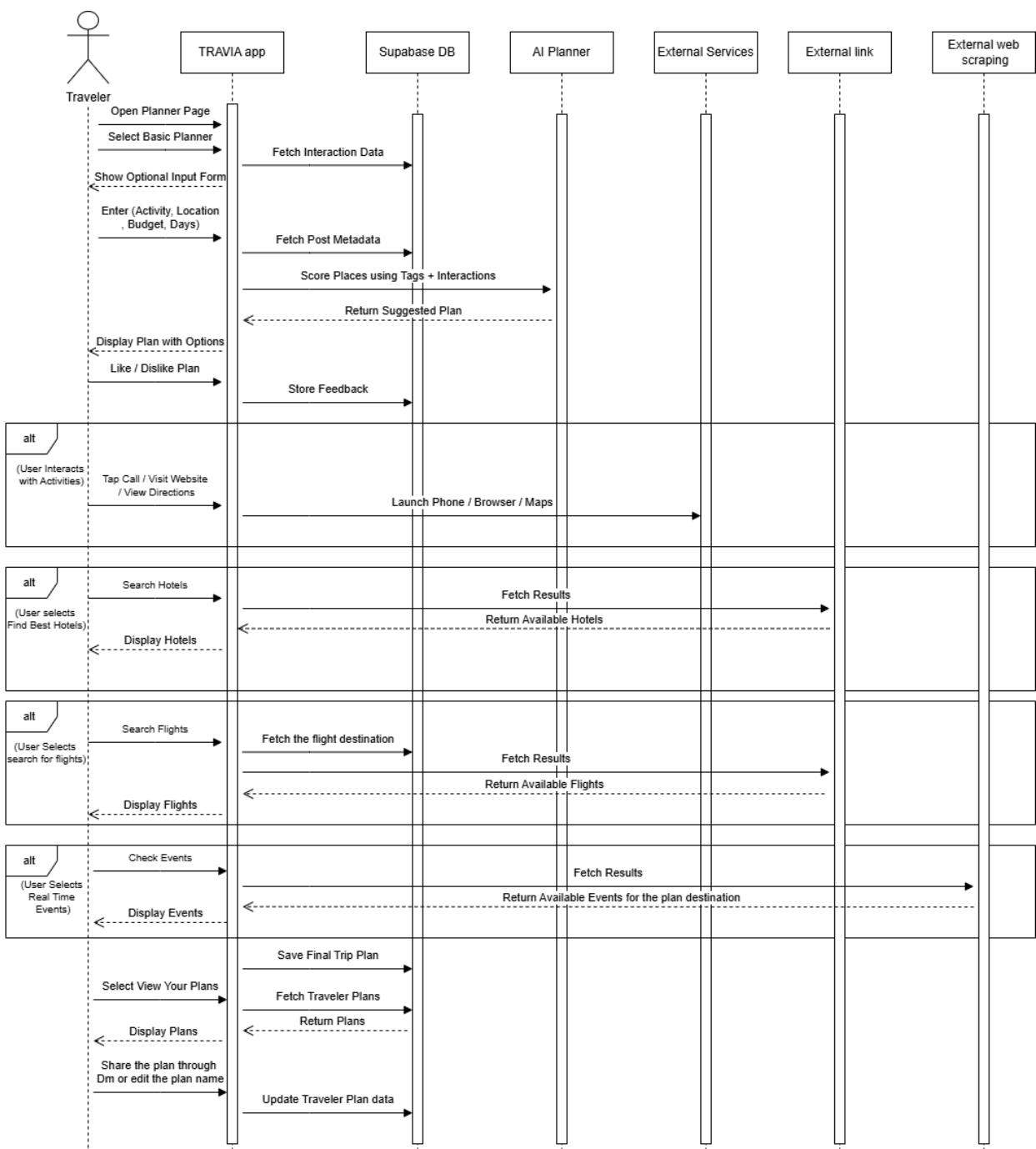


Figure 29 : Use Basic Planner

5.2.13 Use Advanced Planner:

Table 19 : Use Advanced Planner

Title	Use Advanced Planner.
Primary Actor	Traveler
Goal	To generate a highly personalized travel plan using both explicit user preferences and AI-driven decision-making.
Precondition(s)	<ul style="list-style-type: none"> ● User must be logged in. ● User has navigated to the Planner section. ● Firebase Authentication is active. ● Supabase is accessible to retrieve user interaction history.
Postcondition	<ul style="list-style-type: none"> ● AI learns from feedback and updates the RL model. ● Plans are saved and retrievable from the user's saved plans section.
Basic Flow of events	<ol style="list-style-type: none"> 1. Traveler selects Advanced Planner. 2. System prompts for explicit preferences: <ol style="list-style-type: none"> a. Travel category (e.g., night club, GYM, café). b. Location (city/country). c. Budget range. d. Number of days. e. Tags: noisy places, classy places, good places for kids, family-friendly places. f. Cuisine type (Italian, Mexican, Café, Fast Food). 3. Traveler inputs preferences. 4. System fetches user interaction data (likes/dislikes, story views, etc.). 5. AI Planner: <ol style="list-style-type: none"> a. Uses MCDA to match destinations/businesses based on explicit + learned preferences. b. Uses Reinforcement Learning (RL) to improve future suggestions. 6. System generates multiple trip plans. 7. Traveler can : <ol style="list-style-type: none"> a. Like or dislike suggested plans. b. Can view the suggested Activities info:

	<ul style="list-style-type: none"> i. Call the place. ii. Visits the place website. iii. View the directions to the places through google maps. <p>c. Find the best or cheapest Hotels.</p> <p>d. Search for deals on flight reservations.</p> <p>e. Check the Real Time Events happening the plan location.</p> <p>8. Traveler reviews and selects or provides feedback on the plans.</p> <p>9. System stores feedback for future learning.</p> <p>10. Trip plan is saved in Supabase.</p> <p>11. Traveler can share his saved plans through DM and he can edit the plan name.</p>
Alternative Flow	<ul style="list-style-type: none"> ● Traveler skips certain preferences → The system uses defaults or fills in values from past interactions. <i>(Example: if the user doesn't enter a budget, it assumes "no budget constraint.")</i> ● Traveler changes preferences after plan is generated → The system re-triggers the planning process with updated inputs, regenerating the trip suggestions. ● Traveler dislikes all generated plans → The system prompts: <i>"Would you like to adjust your preferences and try again?"</i> ● Traveler wants to start over mid-process → Traveler taps a "Reset" button, which clears current input and restarts the Advanced Planner setup.
Exception(s):	<ul style="list-style-type: none"> ● If internet is disconnected, the system shows: <i>"Please check your connection and try again."</i> ● If preferences are incomplete or invalid, display: <i>"Please complete all required fields to continue."</i> ● If AI services fail (MCDA/RL), show fallback message: <i>"Unable to generate plan right now. Please try again later."</i> ● If no matching destinations are found, show: <i>"No results found for current preferences. Try adjusting them."</i> ● If feedback submission fails, alert the user: <i>"Plan saved, but feedback couldn't be recorded."</i>

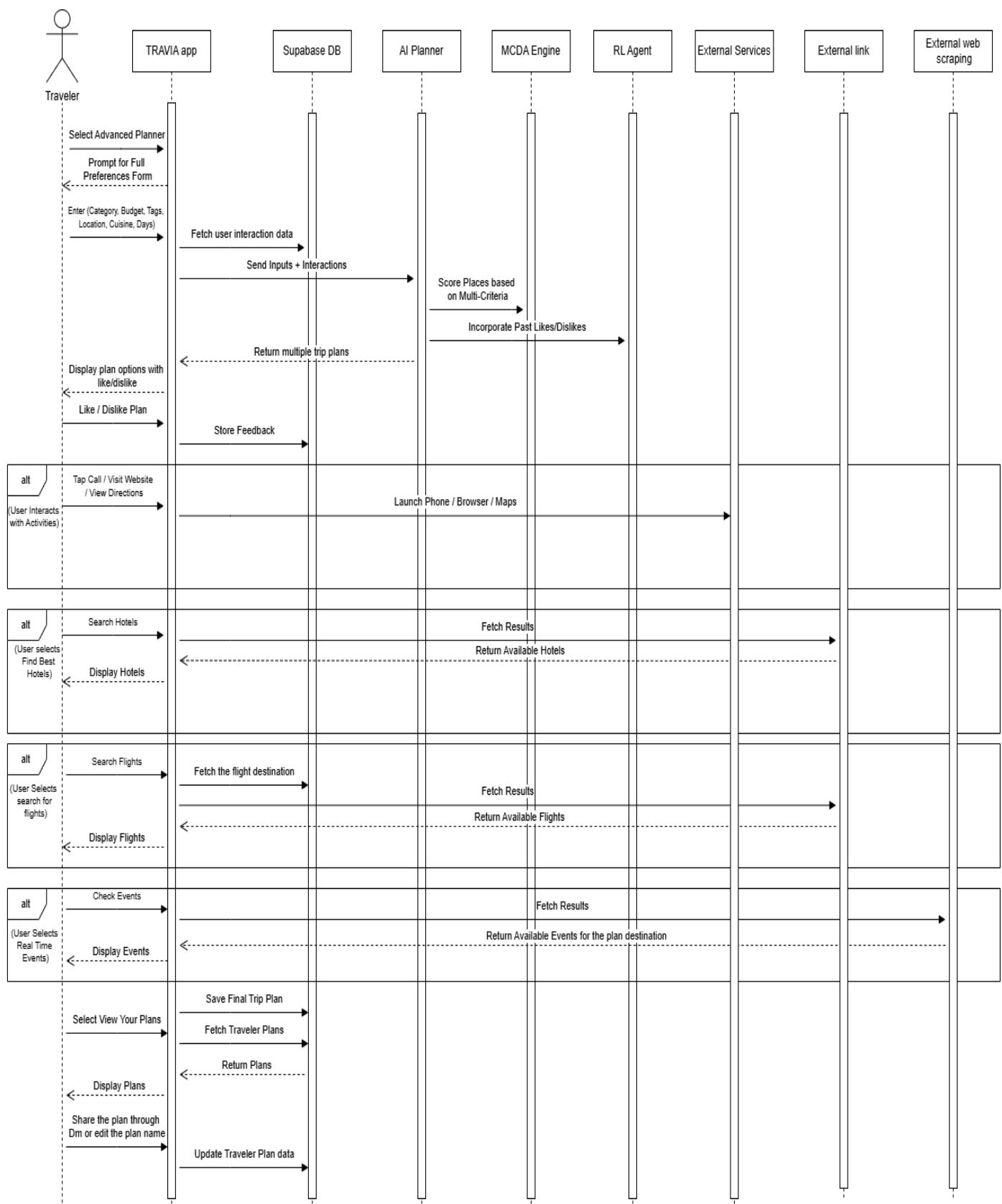


Figure 30 : Use Advanced Planner

5.3 SYSTEM ARCHITECTURE DIAGRAM:

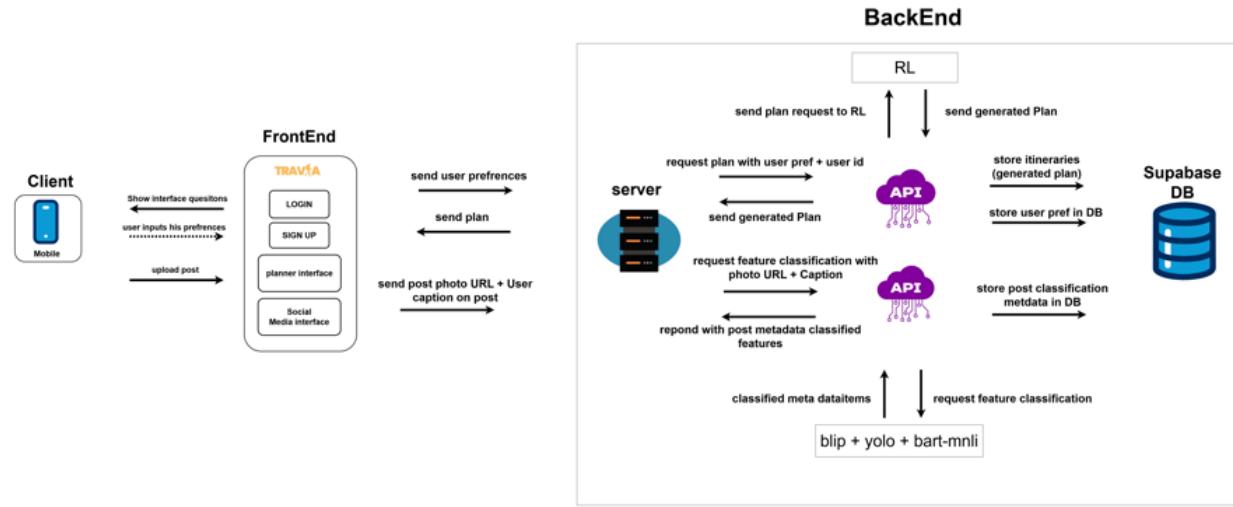


Figure 31 : System Architecture 1

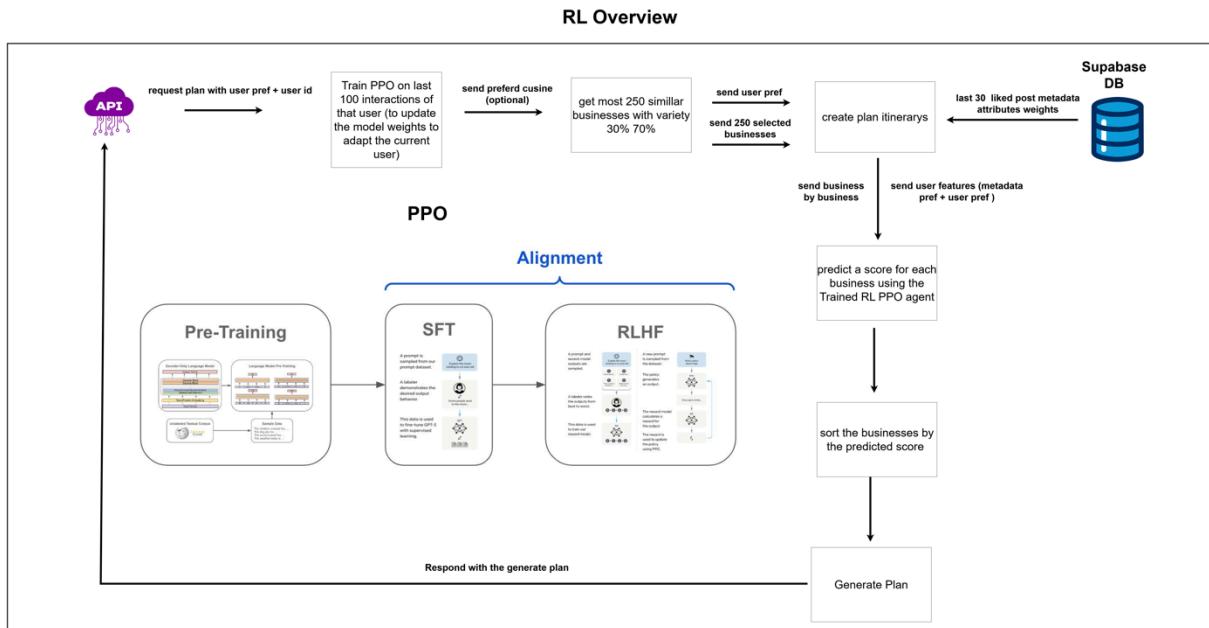


Figure 32: System Architecture 2

Classify Post Overview

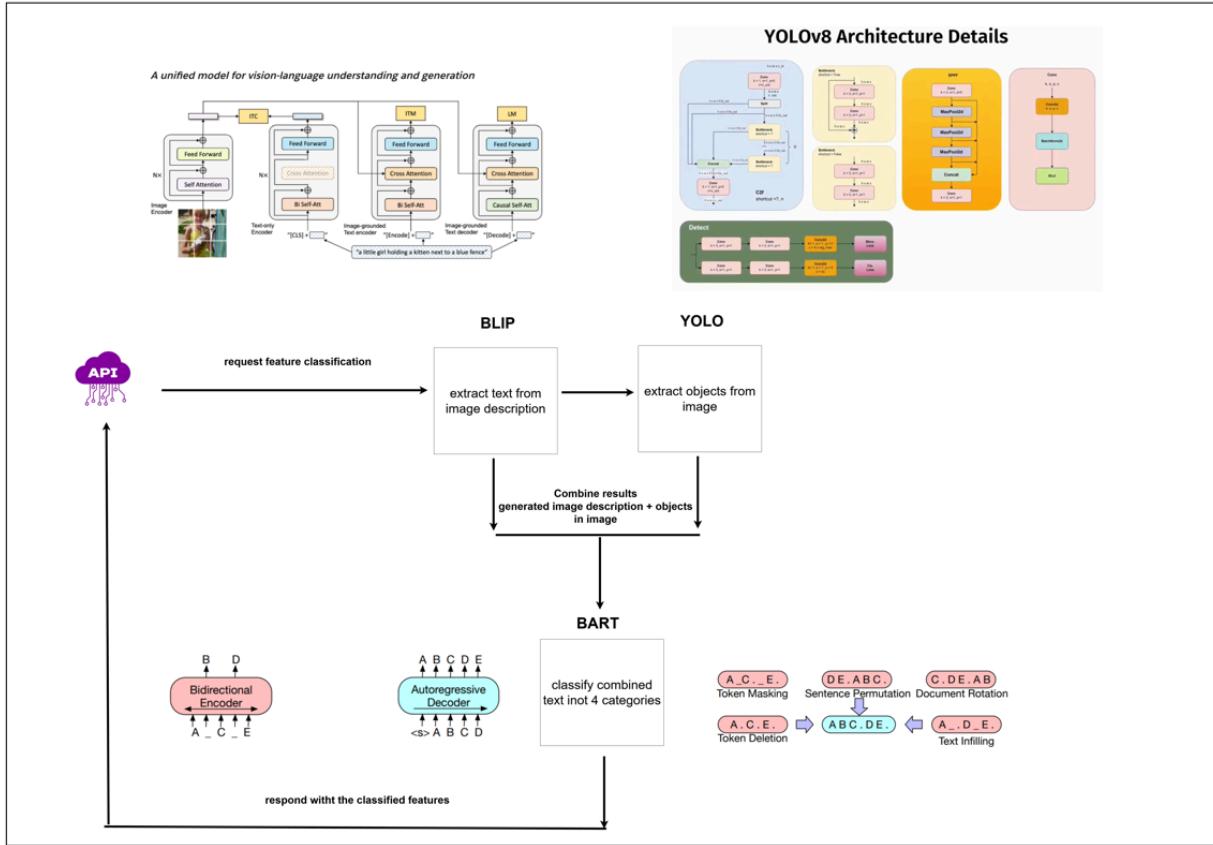


Figure 33 : System Architecture 3

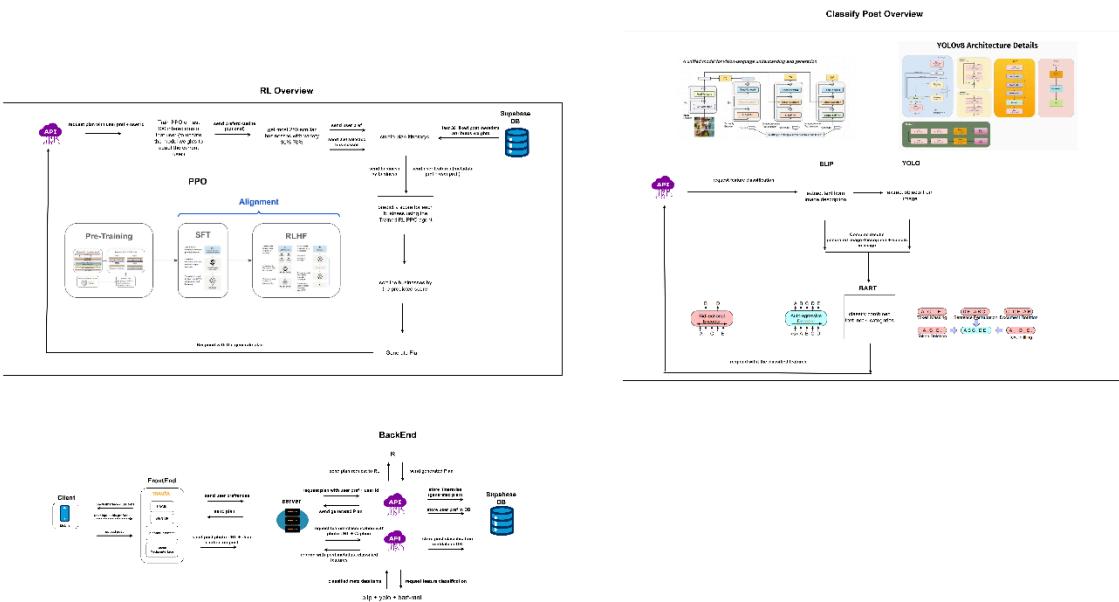


Figure 34 : System Architecture 4

5.4 SYSTEM PROTOTYPE

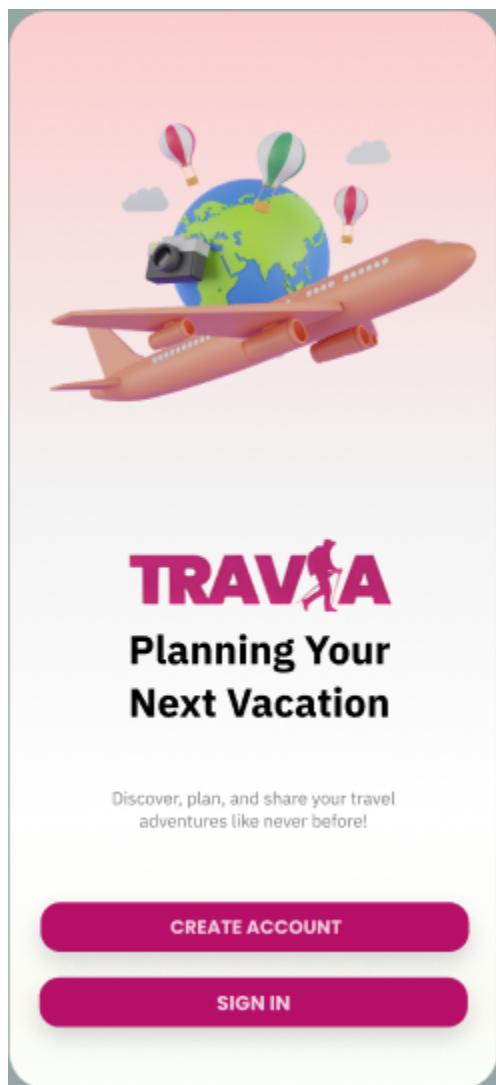


Figure 35 : Sign Up Prototype

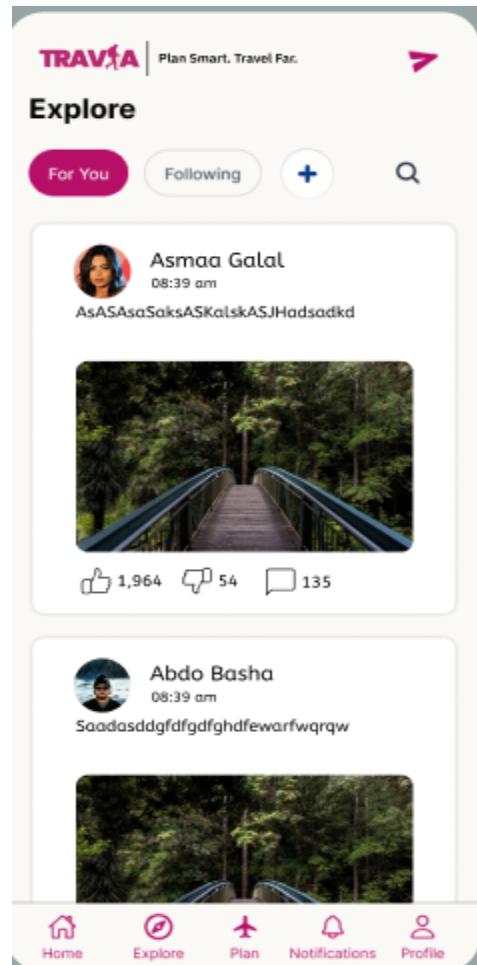
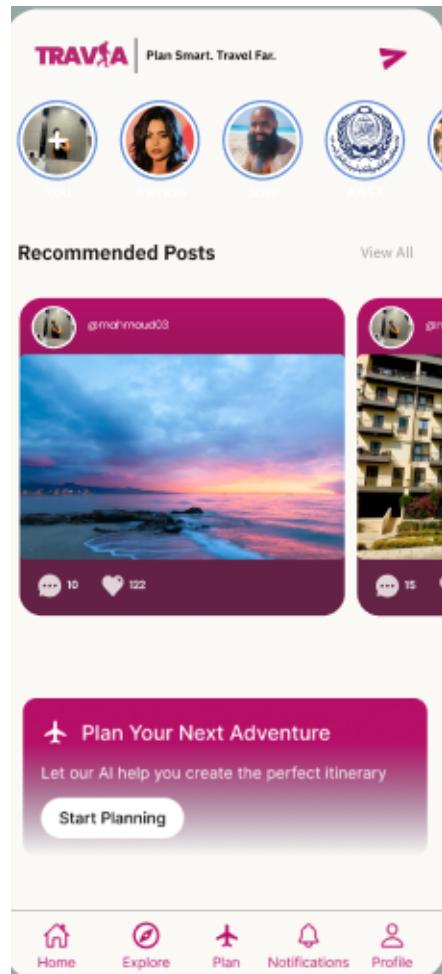


Figure 36 : Explore page Prototype



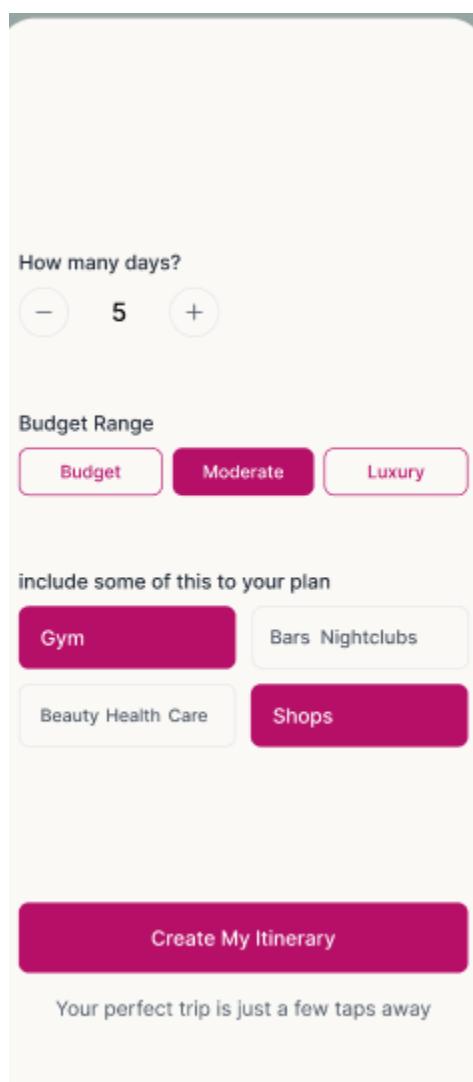


Figure 38 : Basic Plan Prototype

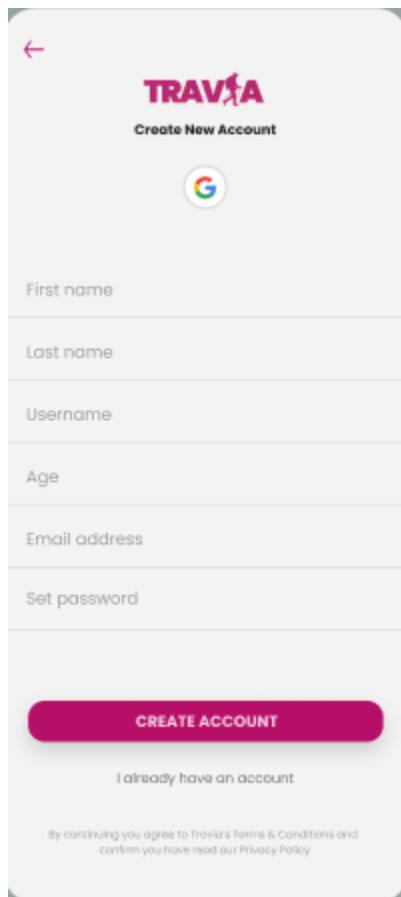


Figure 39 : create account Prototype

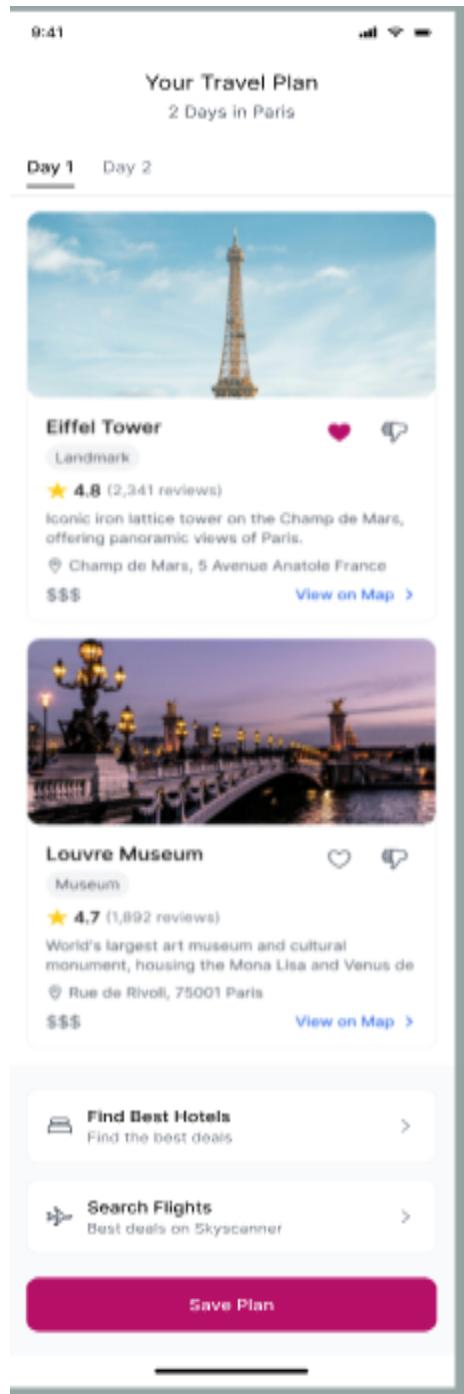


Figure 40 : plan generated Prototype

5.5 Entity Relationship Diagram

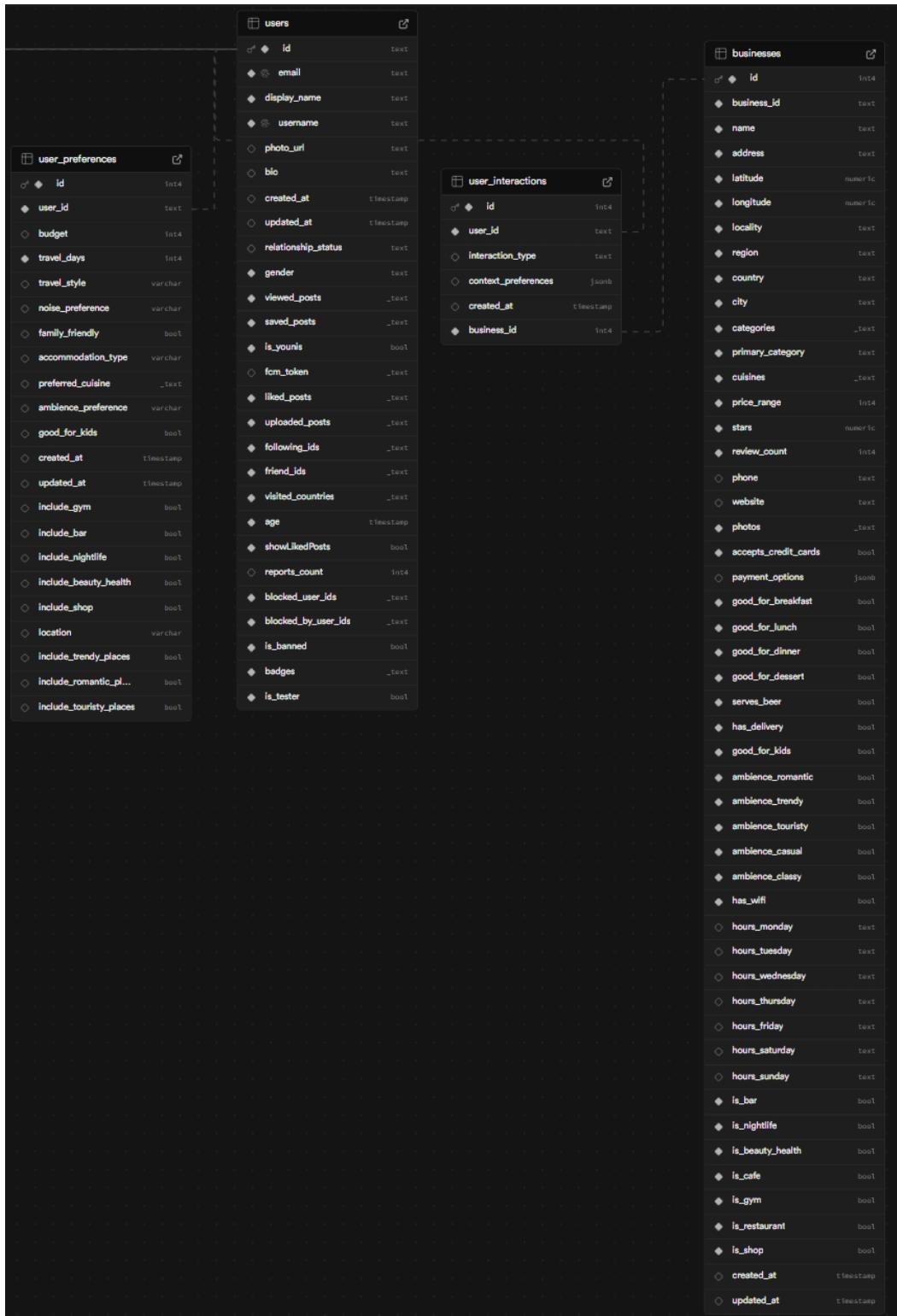


Figure 41 : ERD 1

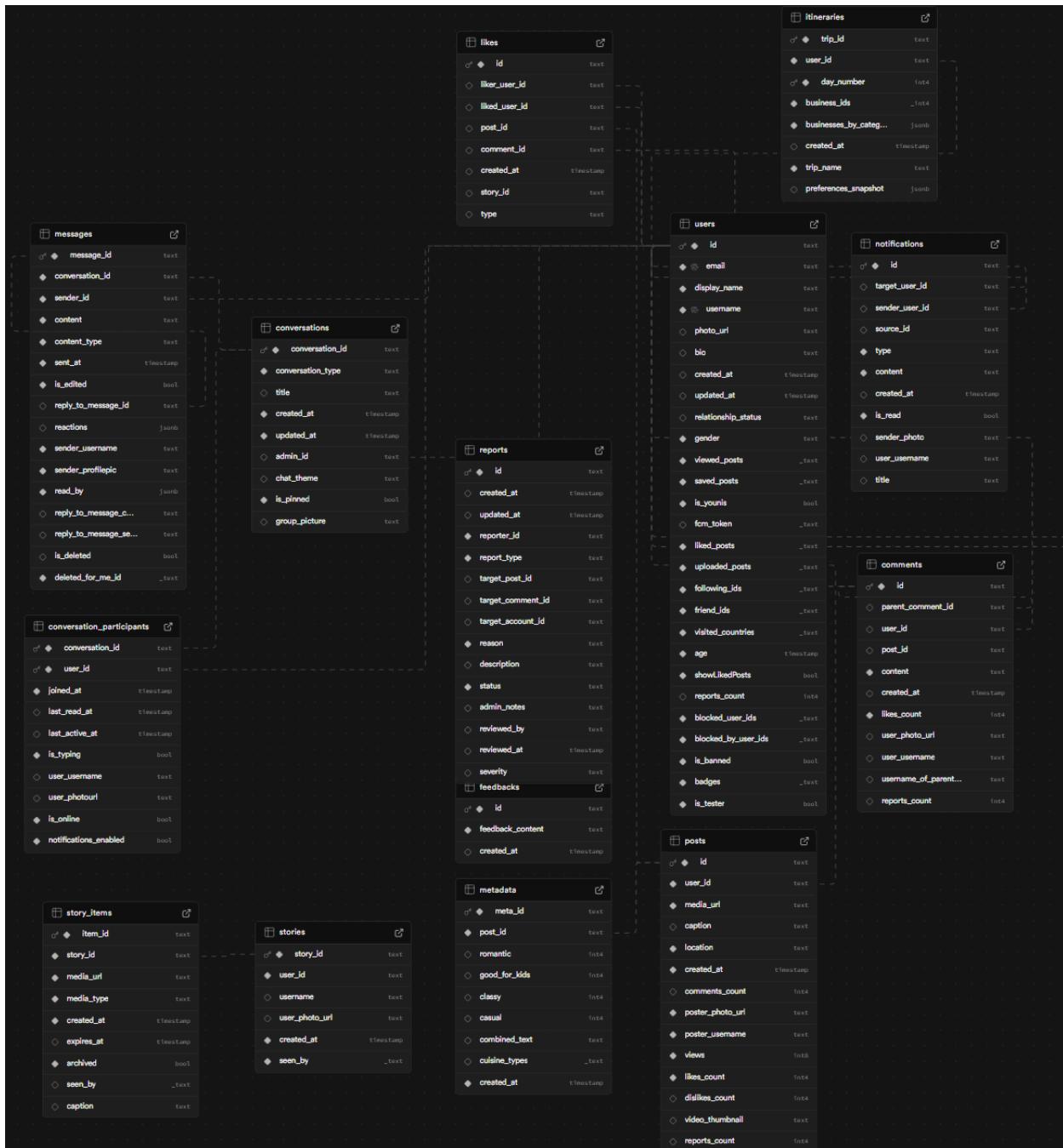


Figure 42 : ERD 2

5.6 Functions And Triggers of The Database

5.6.1 Functions:

5.6.1.1 Messaging & Conversations

1. **add_message**

Inserts a new message into a conversation and returns its ID.

2. **add_deleted_for_me_user**

Adds a user to the “deleted_for_me” list for a specific message.

3. **mark_messages_as_read**

Marks all messages as read in a given conversation.

4. **get_chat_details**

Retrieves full details about a chat, including group settings and participants.

5. **get_conversation_details**

Returns all active conversations for a user with metadata and UI state.

6. **set_message_sender_info (trigger function)**

Auto-fills sender information during message insertion.

7. **fill_reply_to_message_content (trigger function)**

Populates quoted reply content in case of message replies.

8. **update_reply_sender (trigger function)**

Sets reply sender metadata.

5.6.1.2 Posts, Likes, and Views

1. **add_saved_post / remove_saved_post**
Adds/removes a post from the user's saved list.
2. **add_viewed_post**
Marks a post as viewed to track engagement.
3. **update_post_likes_count (trigger function)**
Updates like count of a post when liked/unliked.
4. **update_post_views (trigger function)**
Increments post view count on new view.
5. **append_uploaded_post_id (trigger function)**
Adds new post ID to user's uploaded list.

5.6.1.3 Comments & Notifications

1. **update_comment_likes_count**
Updates the number of likes on a comment.
2. **update_comments_count**
Keeps track of number of comments per post.
3. **fill_comment_user_info (trigger function)**
Adds user metadata (photo, username) to comment automatically.
4. **fill_notification_sender_info (trigger function)**
Fills in sender details when notification is generated.
5. **modify_notification_title (trigger function)**
Auto-formats or adjusts notification title text.

5.6.1.4 User & Relationship

- 1. `follow_user / unfollow_user`**

Manages follower relationships between users.

- 2. `can_users_interact`**

Checks whether two users are allowed to interact (e.g., not blocked).

- 3. `remove_follower`**

Removes a follower from user's list manually.

- 4. `add_to_array / remove_from_array`**

Utility functions to modify array columns (e.g., followers list).

- 5. `refresh_user_badges (trigger function)`**

Evaluates and updates user badges based on activity.

- 6. `update_user_badges (trigger function)`**

Same as above, used after major actions (like post/comment).

- 7. `handle_user_deletion`**

Handles full cascade deletion logic when a user is removed.

- 8. `remove_deleted_user_references`**

Cleans all references to a deleted user.

5.6.1.5 Planning / Preferences

- 1. `update_user_interactions_timestamp`**

Updates the timestamp when the user interacts with planning module.

- 2. `set_limit / show_limit`**

Used for fuzzy scoring or preferences tuning.

- 3. `similarity / word_similarity / strict_word_similarity (and variants)`**

Functions used by the AI engine for text/tag similarity.

5.6.2 Triggers:

5.6.2.1 Messaging

1. messages_inserted → **update_conversations_timestamp**
2. trigger_set_message_sender_info → **set_message_sender_info**
3. auto_fill_reply_sender → **update_reply_sender**
4. auto_fill_reply_to_message_content → **fill_reply_to_message_content**

5.6.2.2 Posts

1. posts_before_insert_trigger → **fill_post_user_info**
2. trigger_append_uploaded_post_id → **append_uploaded_post_id**
3. trigger_clean_up_deleted_post → **clean_up_deleted_post**
4. trigger_update_likes_count → **update_post_likes_count**

5.6.2.3 Comments

1. comments_before_insert_trigger → **fill_comment_user_info**
2. update_comment_likes_count_trigger → **update_comment_likes_count**
3. update_comments_count_trigger → **update_comments_count**

5.6.2.4 Users

1. after_user_delete → **delete_notifications_for_user**
2. trigger_handle_user_deletion → **handle_user_deletion**
3. trigger_update_user_badges → **update_user_badges**
4. trigger_update_participant_user_info → **update_participant_user_info**
5. update_comments_on_user_change_trigger
6. update_notifications_on_user_change_trigger
7. update_posts_on_user_change_trigger
8. trigger_update_message_sender_info

5.6.2.5 Relationships & Follows

1. trigger_manage_liked_post_array
2. follow_user / remove_follower
3. can_users_interact

5.6.2.6 Notifications

1. notifications_before_insert_trigger → **fill_notification_sender_info**
2. zzz_notifications_title → **modify_notification_title**
3. on_notification-insert → **http_request**

5.6.2.7 Stories

1. trigger_populate_story_user_info
2. delete_expired_on_insert
3. delete_stories_on_user_delete_trigger
4. update_stories_user_info

5.6.2.8 Reports

1. auto_assign_report_severity
2. reports_increment_count / decrement_report_count
3. update_reports_updated_at

6 SYSTEM ANALYSIS

Systems analysis is the process of studying a procedure or business to identify its goal and purposes and create systems and procedures that will efficiently achieve them. We will discuss and analyze different Apps and see how they differentiate from us.

6.1 Data Gathering

To populate our global business database, we used the Foursquare Places API, a reliable and rich location-based service that provides detailed information about millions of businesses across the world. The API returns data about businesses such as their name, location, category, rating, price level, opening hours, photos, and even advanced features like amenities, payment methods, and service offerings.

6.1.1 Overcoming the Foursquare API Limitations

Foursquare imposes a limit on the number of results per request—a maximum of 50 businesses per API call. To gather a significantly larger and more representative sample of businesses in each city, we implemented a smart multi-radius search strategy:

For every city, we first retrieve its latitude and longitude coordinates.

Then, we simulate a grid of multiple search points around the city's center by slightly shifting the coordinates in various directions (north-east, south-west, etc.).

Each shifted point is queried with a specified search radius (e.g., 1500 to 4000 meters) to retrieve up to 50 businesses.

These requests are staggered with delays to respect rate limits and avoid overwhelming the API.

Each API response is parsed, and duplicate businesses are filtered out using their unique Foursquare ID.

This technique effectively increases data coverage and allows us to surpass the per-request limitations of the API, ensuring that we retrieve a wide variety of businesses spread across the entire urban area, not just the central location.

6.1.2 Cleaning and Enriching the Data

Once the raw business data is retrieved:

We extract important fields including name, address, geolocation, rating, price, website, phone, photos, and hours.

We use the API's feature data to classify businesses (e.g., romantic, trendy, casual, good for kids, serves beer, has delivery).

A custom classifier maps the primary category of each business into broader groups like restaurants, cafes, bars, gyms, shops, and health & beauty services.

Chain businesses such as McDonald's or Starbucks are limited per city to avoid over-representation.

6.1.3 Final Output and Export

The cleaned and enriched data is then:

Formatted to match our backend schema (compatible with Supabase).

Deduplicated and limited to unique businesses per city.

Exported to an Excel file with over 40 columns of structured data, ready for further processing, visualization, or ingestion into our database.

This system allows us to scale globally, automate updates, and maintain consistent business data across diverse regions and categories, forming a strong foundation for our travel planning and recommendation features.

6.2 Data Cleaning

6.2.1 Removing low quality entries

To ensure meaningful recommendations:

- We removed businesses with no rating or review count.
- Foursquare ratings are originally on a 0–10 scale, which we rescaled to 0–5 to match the standard out AI Uses.

6.2.2 Filtering Out Incomplete City Data

Cities with fewer than 200 valid businesses after filtering were excluded to ensure a minimum threshold of content necessary for itinerary generation and regional analytics.

6.2.3 Removing Unwanted Categories

To keep our dataset focused on leisure, dining, and entertainment, we excluded businesses in these non-relevant or inappropriate categories:

'Church', 'Mosque', 'Military Base', 'Non-Profit Organization', 'Office Supplies'
'Butcher', 'Grocery Store', 'Computer and Electronics Retail'

These types of businesses do not contribute to trip planning or social experiences.

6.2.4 Handling Missing Values

Columns with missing locality or region were filled by inheriting values from neighboring businesses in the same city (above or below in the data sequence).

Other missing fields such as phone numbers or photos were left blank or marked as null to maintain schema consistency.

6.2.5 Formatting Complex Fields

We formatted several nested or raw fields into AI-compatible JSON or string formats:
opening_hours, photos, and features were restructured to be more consistent and compact.
These fields were transformed to match our AI schema, making them ready for use in recommendation.

6.2.6 Marking touristic businesses

Foursquare doesn't directly flag whether a business is "touristy." To infer this:

We analyzed the category names of each business.

If a business's category included terms like "observation deck", "monument", "ski resort", "sunset cruise", etc., we flagged it as touristy.

A detailed heuristic list of keywords was used, covering attractions, natural landmarks, cultural experiences, and guided tours.

6.2.7 Identifying Nightlife Spots

Similarly, we labeled businesses as part of nightlife if their category included terms like: 'Hookah Bar', 'Night Club', 'Jazz Club', 'Speakeasy', 'Concert Hall', 'Social Club', or 'Country Dance Club'.

This allows our system to better categorize nightlife venues for users seeking evening or late-night activities.

6.2 DATA ANALYSIS

After getting the data from Foursquare, cleaning it and uploading to Supabase for our AI Models and interface, we need to do some basic data analysis to determine many things that will help us improve our model and help with our decision making.

6.2.1 Top 10 Cuisines

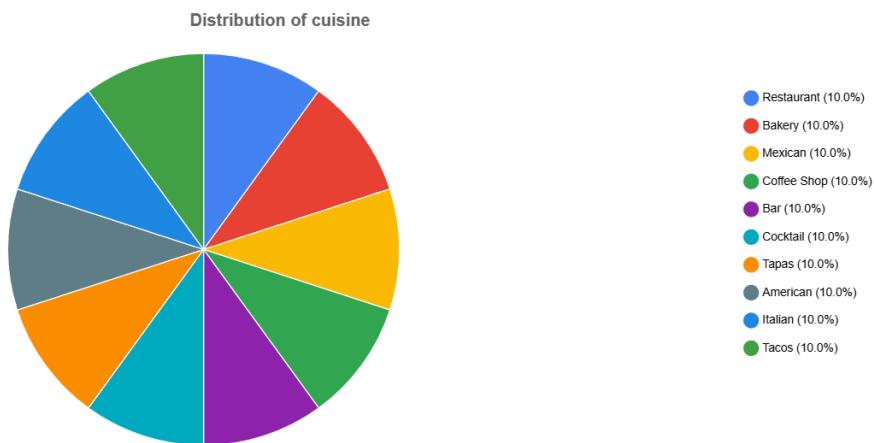


Figure 43 : Data Analysis 1

Insights: The graph shows that all cuisines have equal distribution so that no type of cuisine is preferred over the other by the application allowing freedom of choice for the user.

6.2.2 Businesses count for each city

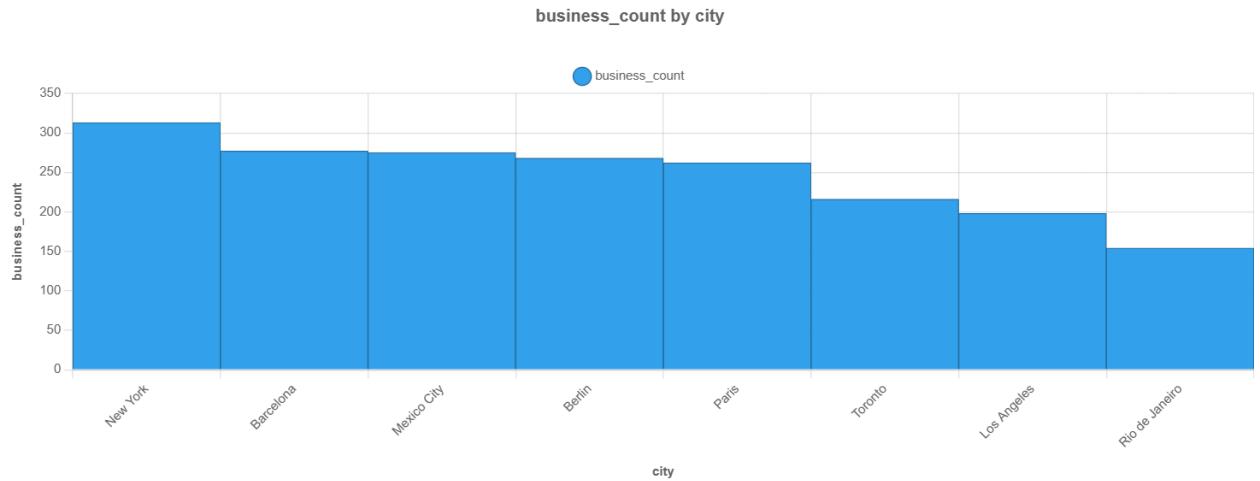


Figure 44: Data Analysis 2

Insights: The graph shows that New York has the most amount of businesses inferring that New York will have the most diversified plans.

6.2.3 Percentages of trendy places across cities

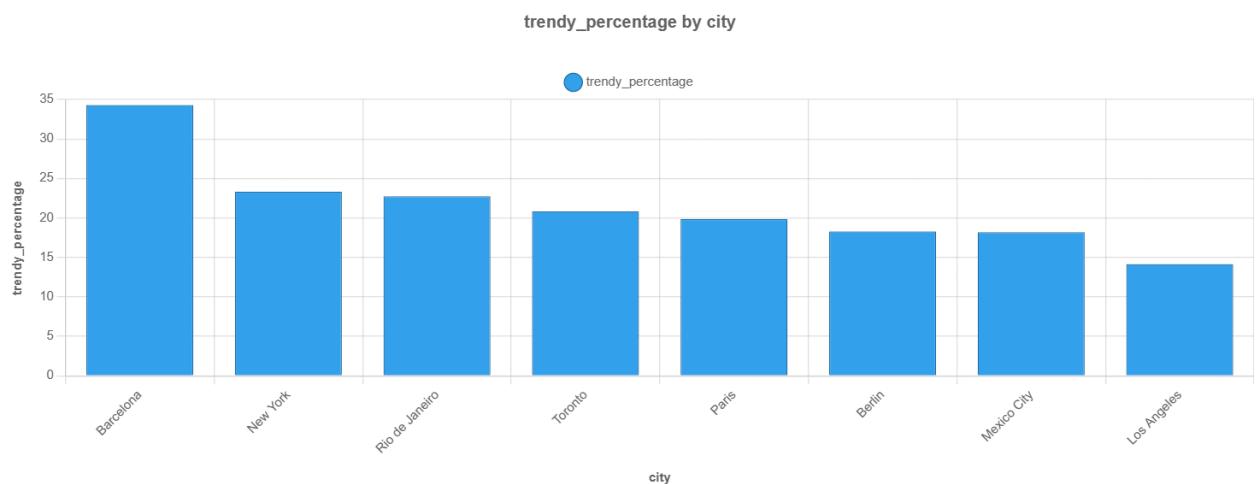


Figure 45: Data Analysis 3

Insights: The graph shows that Barcelona has the most amount of trendy places inferring that it will be the most likeable city for users seeking to visit trendy places

6.2.4 Ranking city with their number of restaurants

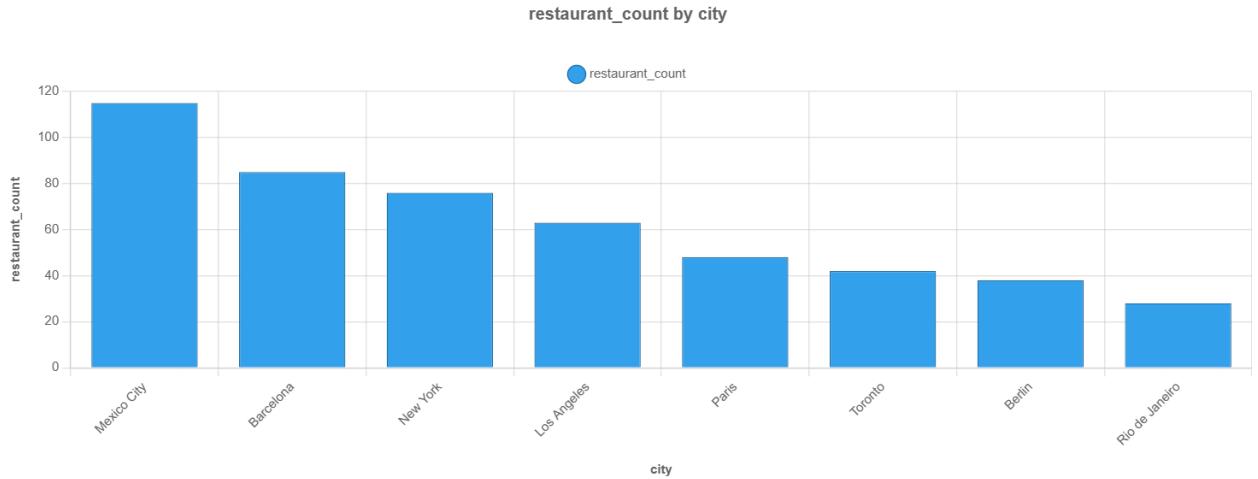


Figure 46: Data Analysis 4

Insights: The graph shows that Mexico City has the most amount of restaurants inferring that it will be the preferred destination for foodies and food bloggers

6.2.5 Number of businesses for every price range 1 -> 4

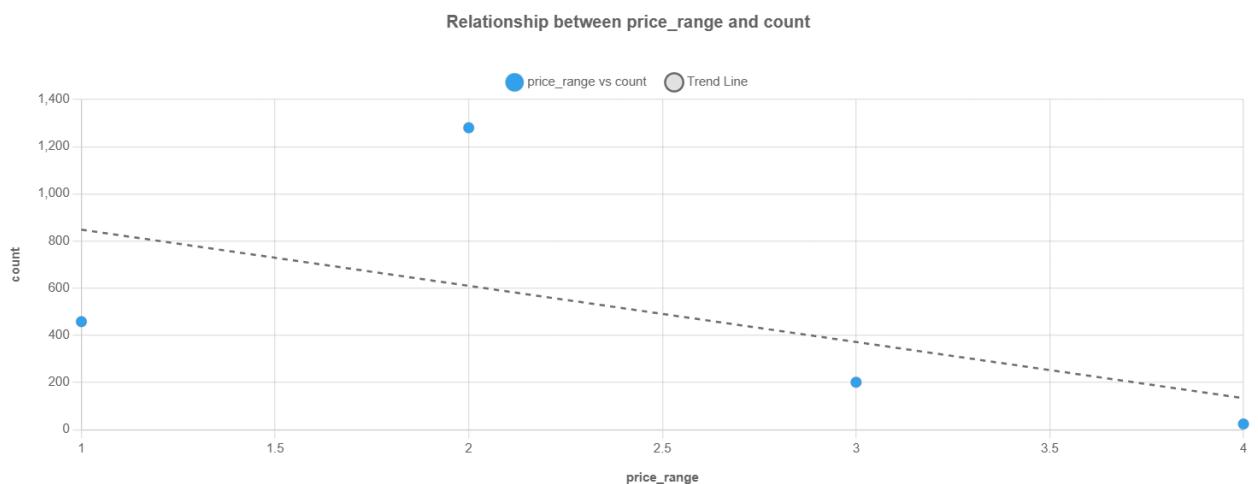


Figure 47: Data Analysis 5

Insights: This show us that price range 2 (Moderate pricing) contains the most businesses of the data with a high number of over 1200 businesses, other price ranges especially really classy places like 4 have the least amount of businesses inferring that the data set is budget friendly allowing for more middle class users

7 IMPLEMENTATION AND TESTING

7.1 IMPLEMENTATION

7.1.1 Technologies and tool utilized

7.1.1.1 Flutter

Description

Flutter is an open-source UI software development kit created by Google. It is used to develop cross platform applications from a single codebase for any web browser, Fuchsia, Android, iOS, Linux, macOS, and Windows.

Purpose

We will use Flutter to create the mobile application that will serve as an interface for our System.

7.1.1.2 Dart

Description

Dart is a programming language developed by Google. It can be used to develop web and mobile apps as well as server and desktop applications.

Purpose

We will use Dart with the flutter framework to create our mobile application.

7.1.1.3 Android Studio

Description

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.

Purpose

We will use Android Studio as our IDE to implement our Flutter application.

7.1.1.4 Python

Description

Python is a high-level, general-purpose programming language that is dynamically typed.

Purpose

We will use Python to implement our AI model.

7.1.1.5 VS Code

Description

VS Code is an integrated development environment developed by Microsoft for Windows, Linux, macOS and web browsers. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded version control with Git. Users can change the theme, keyboard shortcuts and preferences, as well as install extensions that add functionality.

Purpose

We will use VS Code as our IDE to implement our AI model using Python.

7.1.1.6 Web Scraping

Description

Web scraping is a form of copying in which specific data is gathered and copied from the web, typically into a central local database or spreadsheet, for later retrieval or analysis.

Purpose

We will use web scraping to obtain data from various sources in order to display to the user. We are considering web scraping google maps data for the selected city.

7.1.1.7 Fast API

Description

FastAPI is a high-performance web framework for building HTTP-based service APIs in Python 3.8+. It uses Pydantic and type hints to validate, serialize and deserialize data. FastAPI also automatically generates OpenAPI documentation for APIs built with it.

Purpose

We will use Fast API in order to establish a connection between our mobile application and the AI model.

7.1.1.8 Firebase

Description

Google Firebase is a set of cloud-based development tools that helps mobile app developers build, deploy and scale their apps.

Purpose

We will use Firebase for authentication and push notifications for our mobile application.

7.1.1.9 Artificial Intelligence

Description

Artificial intelligence (AI) is a set of technologies that enable computers to perform a variety of advanced functions, including the ability to see, understand and translate spoken and written language, analyze data, make recommendations, and more.

Purpose

We will use AI to produce a plan for the user based on their preferences and selected options.

7.1.1.10 Supabase

Description

Supabase is an open-source backend-as-a-service (BaaS) that provides developers with a set of tools to build scalable applications quickly. It offers a PostgreSQL database with real-time capabilities, user authentication and storage, all accessible through a modern API and dashboard.

Purpose

We will use supabase as cloud provider for our back-end, It will host our database, store our Images, videos and provide real time updates.

7.2 Development

In this section we will go through the development of the system and showcase of the actual system

7.2.1 Mobile App Development And System Integration:

This section outlines the core mobile app development strategy and how various system components are integrated to deliver a seamless user experience. TRAVIA was built using the Flutter framework to enable cross-platform deployment with a single codebase. The app combines multiple technologies and tools, including Riverpod for state management, Firebase and Supabase for authentication and backend services, and GoRouter for dynamic routing. All the following diagrams in the following sections were made by Mermaid JS, a platform to write diagrams with code.

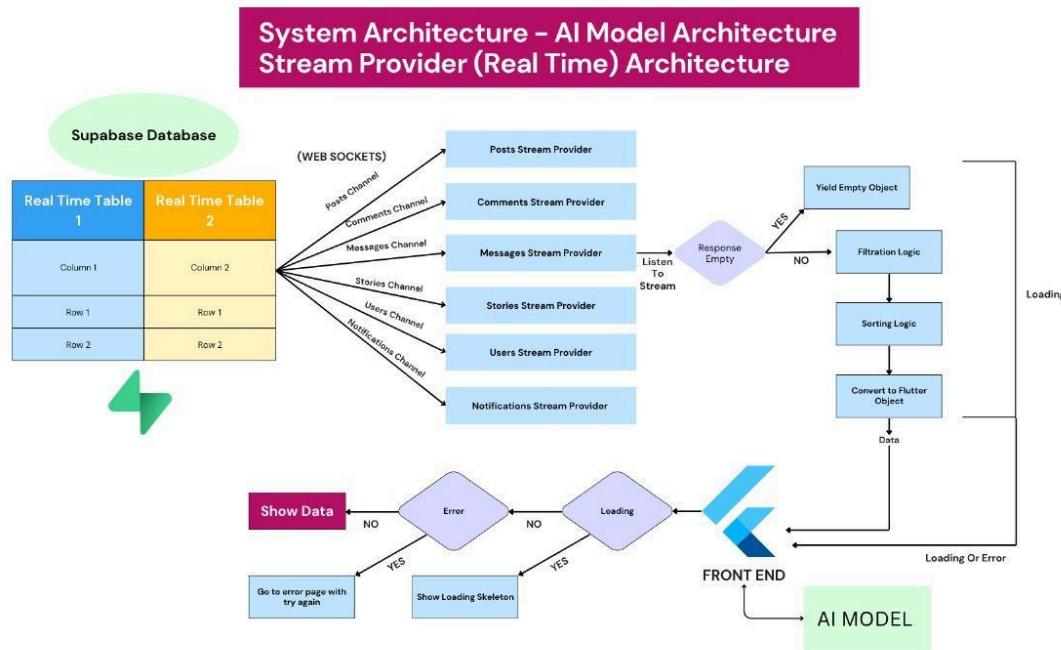


Figure 48 : Travia Architecture

7.2.1.1 State Management with Riverpod

Developing a social media platform with AI elements is not an easy task.

There are a lot of states moving all the time; users uploading posts all the time, sending messages, notifications, etc.

So we need a robust state management solution that links all this complexity together.

We decided to use riverpod for all the state management in the app.

Riverpod allows complex state management to make the app possible; it provides global state management, state invalidation, read states, watch states and more.

7.2.1.2 Back End Integration with Stream Provider Architecture and Real-Time Management

Our backend is hosted in Supabase which provides a real-time functionality to make our tables work with a stream.

Supabase enables WebSocket based subscriptions to database changes, allowing the frontend to react immediately to inserts, updates, and deletes across relevant tables to provide real time functionality.

To use Supabase's real time functionality with Flutter's UI layer, we use Riverpod's StreamProvider architecture. For each reactive feature in the app whether it's messaging, posts, notifications, chat rooms, trip plans, or user data; we define dedicated stream providers that listen to changes on the corresponding Supabase tables or views.

When a provider is instantiated, a dedicated Supabase channel is opened for that user only and filtered in a way that ensures only their data is processed and shown except for some general channels (e.g., posts are general, they have a dedicated channel but no filtration, direct messages have filtration.). This selective listening avoids unnecessary processing and bandwidth usage while preserving the real-time nature of the app.

Each StreamProvider exposes three core states to the UI layer:

Loading: while waiting for the initial data or stream to connect

Data: when new or updated data is received and ready to be rendered

Error: if the stream fails or data fetching encounters an exception

This model allows the UI to declaratively adapt to changes in backend state, enabling features such as:

- Automatically updating the feed when a new post is created
- Instantly showing new messages in a conversation
- Reflecting friend requests or trip invitation status in real-time
- Live syncing profile edits or user preferences

At the end of the life cycle, we handle cleanup state (e.g., cancelling Supabase subscriptions) when widgets are disposed of. This reduces the risk of memory leaks and stale data.

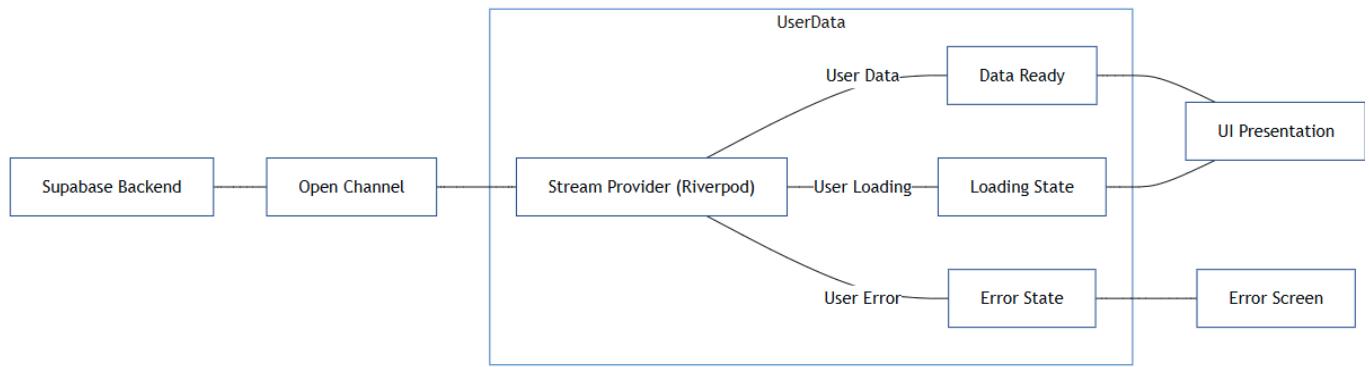


Figure 49 : Development 1

7.2.1.3 Splash Pre Loading

We have developed the splash screen to do work under the hood. Most of the application loads data in a non-blocking way, so all posts, conversations, messages, plans, authentication checks and other factors are loaded immediately with no loading.

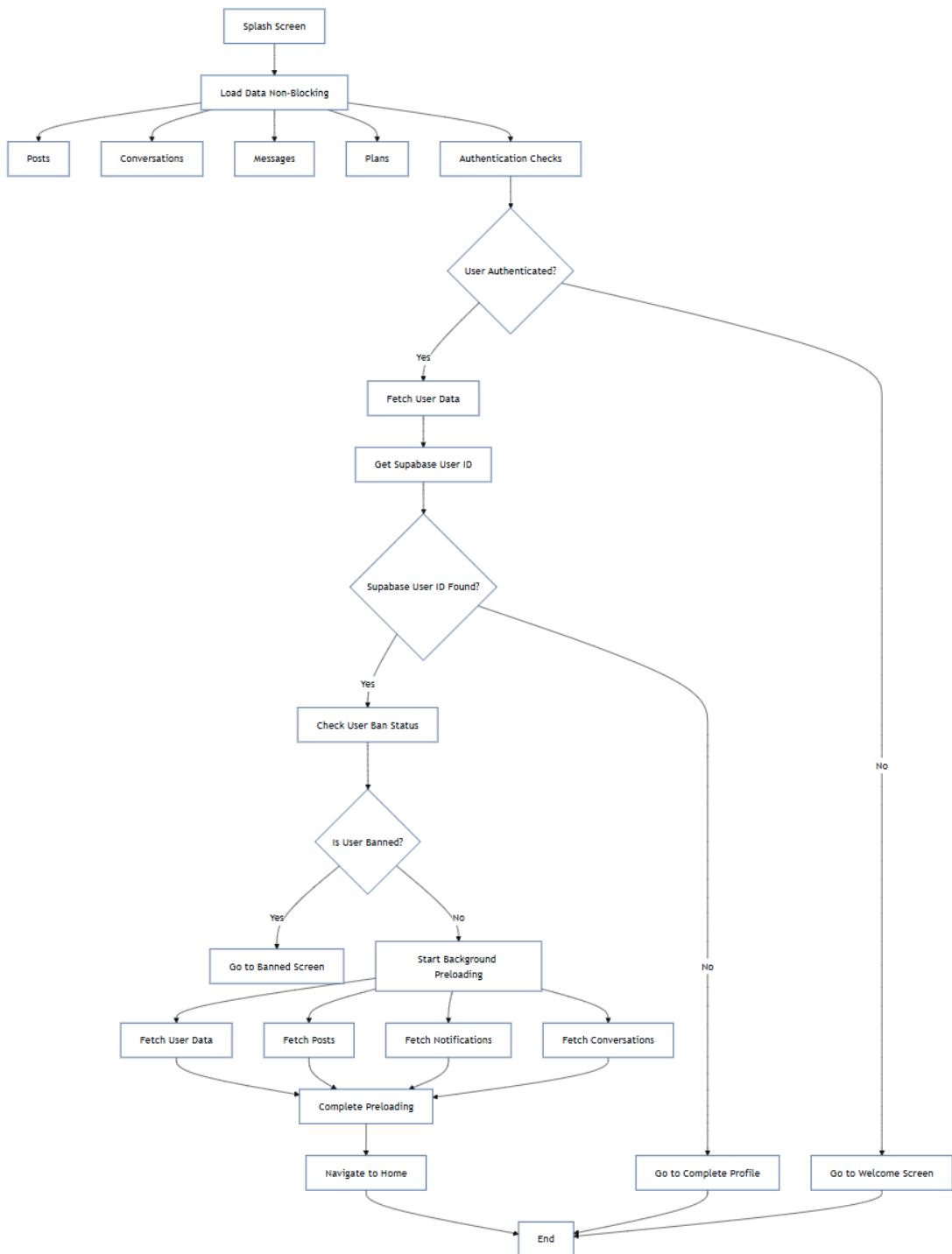


Figure 50: Development 2

7.2.1.4 Authentication

Our authentication is handled by the BaaS service Firebase. We chose Firebase for its ease of handling complex functionalities like security and verification logic, Google sign-in and storing the passwords hashed and salted with no effort from our side.

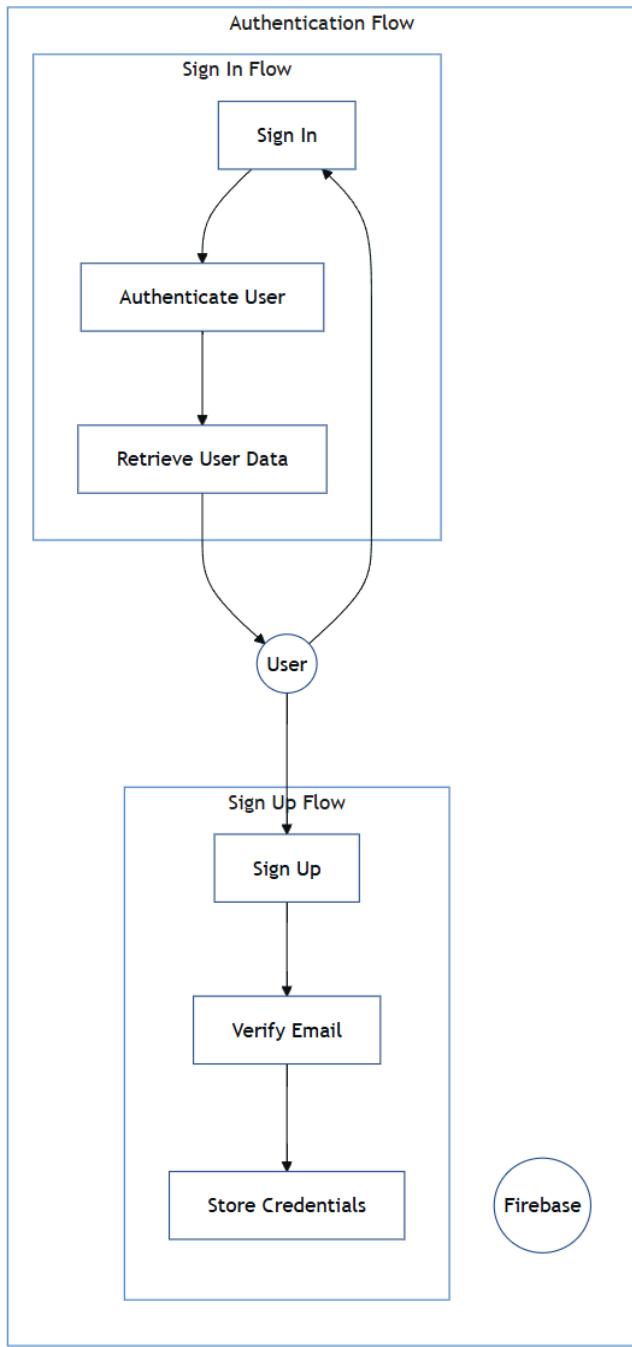


Figure 51: Development 3

7.2.1.5 Home Feed and Post Ranking

The Home Feed in TRAVIA is organized into multiple sections: a personalized Recommended section and additional sections grouped by the country where each post was created. Posts are ranked using a lightweight engagement-based scoring algorithm designed to promote high-quality, timely content while preventing older posts from dominating the feed indefinitely.

The ranking algorithm calculates a post's engagement score by assigning specific weights to different types of user interactions: likes contribute positively, comments have a higher impact, and dislikes reduce the score slightly. The formula also applies a time-decay factor, which gradually lowers a post's score as it ages by 5% per day down to a minimum of 50% of its original value. This ensures that newer posts have a chance to be discovered, while older posts naturally fade unless they continue to gain engagement. Negative scores are not allowed, so any post with a net negative engagement is simply assigned a score of zero.

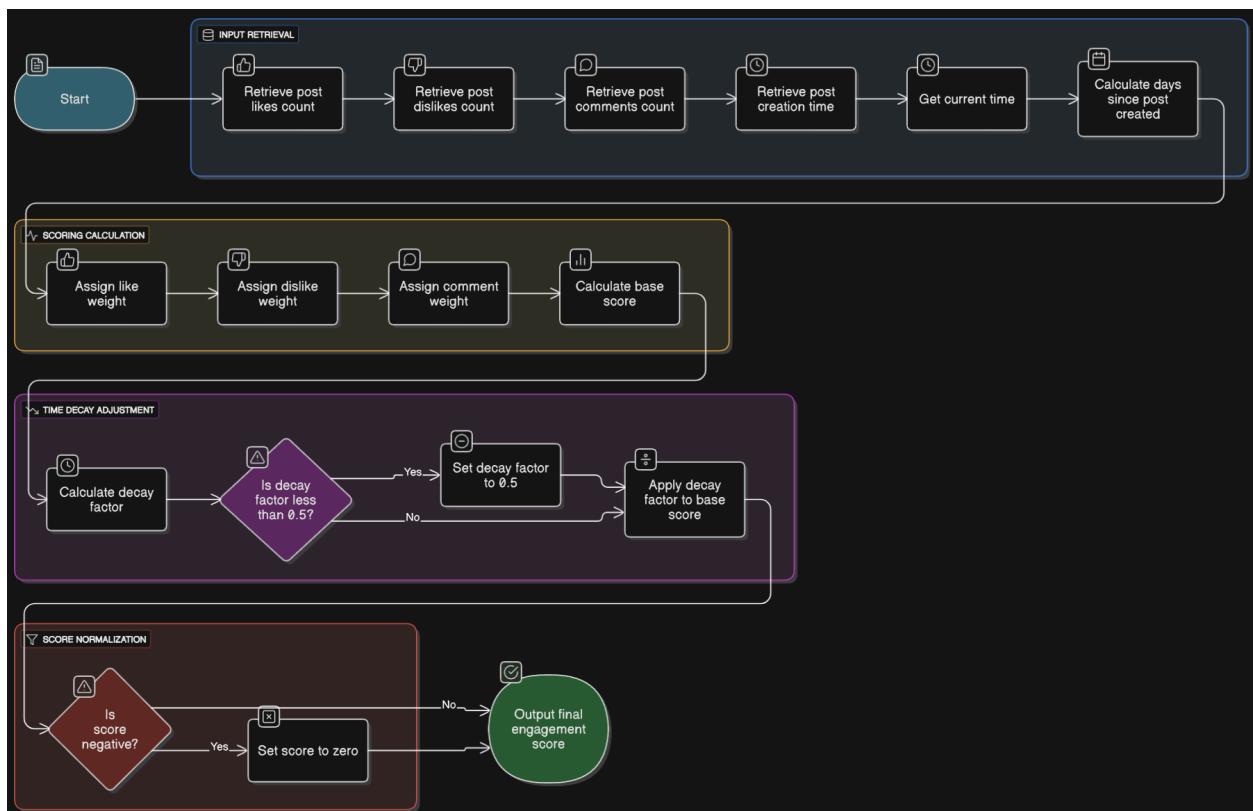


Figure 52: Development 4

7.2.1.6 Push Notifications

We have a complex notification architecture, since Supabase doesn't have a native way to manage push notifications unlike Firebase.

We made it so it connects to the Firebase service named FCM or Firebase cloud messaging.

When a notification is sent to the user either through a comment, message, etc.

A webhook is activated after detecting an insert in the notifications table, it will get the current user's FCM token saved in the users table and make an edge hook that calls the FCM service to send that push notifications.

Since we support logging to the same account with multiple phones, and FCM tokens are linked to the phone and not the user, the FCM tokens are saved in the users table as an array with tokens for each device the user is logged in with.

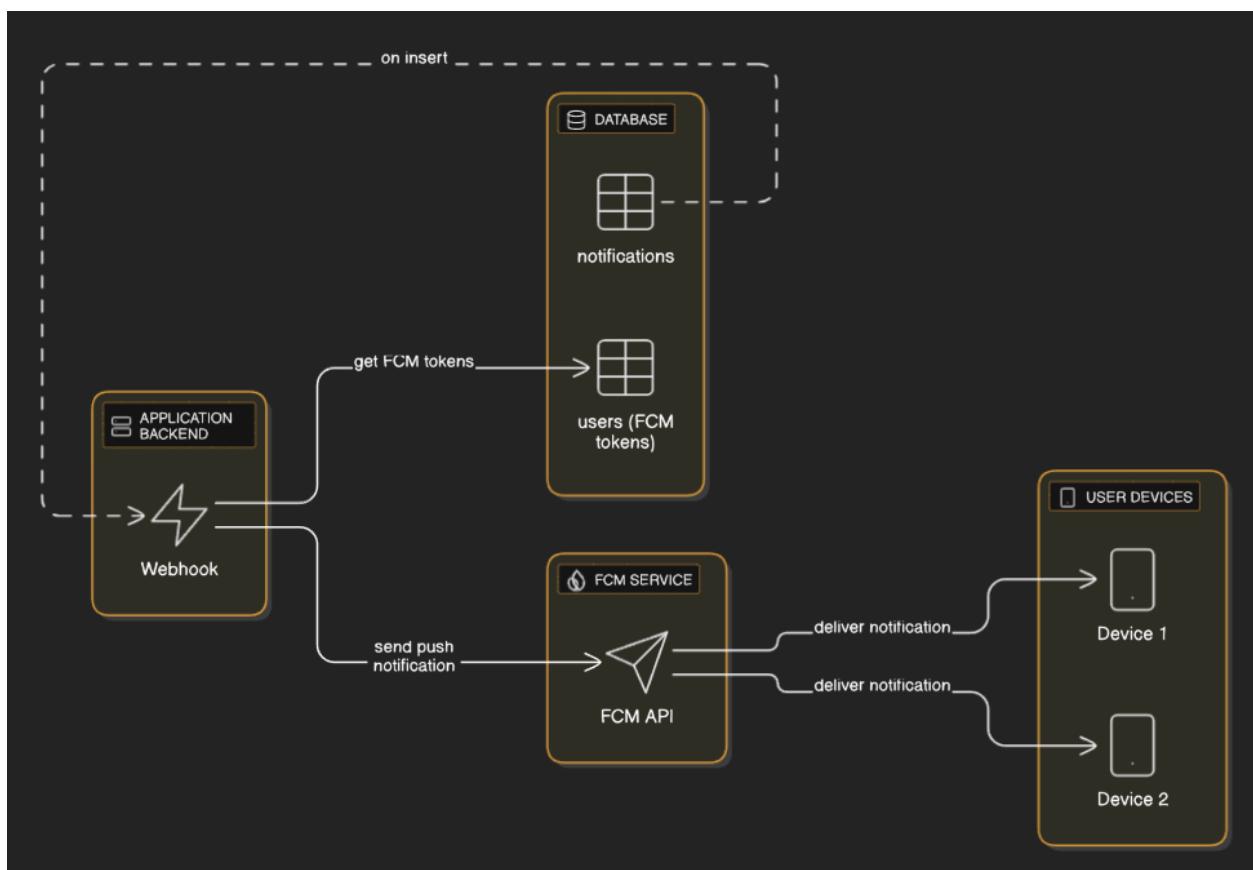


Figure 53: Development 5

7.2.1.7 Routing

We have a robust routing system in place, users get routed around the app using the go router. Go router makes it so every route corresponds to a URL.

The flutter default system for routing is pushing pages into the stack which is not an intelligent solution and introduces some problems:

- No deep link support, especially when users click on a push notification to go to a specific page
 - No web support, since we might scale later to a website; using go router solves that by routing users using links. For example: “/home/profile-page/\${userId}
 - No state driven route management since you have to call pop and push methods manually
- All this can be solved with Go Router.

7.2.1.8 Optimistic UI Updates

To give users immediate feedback on their actions we implemented optimistic ui updates. For any action that relates to asynchronous work like database insertions, deletions, updates; we update the UI immediately but do the heavy work on the background, for example if a user sends a message, the message will be sent immediately but will be treated as a placeholder until it's actually sent to the other participants, if the sending fails, it will flag the message as failed. Another example are likes, when a user likes a post or a story, an immediate ui update for the like button is shown, the like will try to register to the database behind the scenes so we can provide a seamless experience to the user.

7.2.1.9 Messaging Architecture

The messaging system in TRAVIA is designed to support real-time conversations between users, whether in direct (one-on-one) chats or group chats. It consists of three main tables:

conversations: stores overall chat info like title, type (direct or group) etc.

conversation_participants: tracks which users are part of each conversation, their status (typing, online), and when they joined

messages: stores individual messages, including sender info, content, timestamps, message type, replies, and reactions

Each message supports various content types such as text, images, videos, voice notes, and custom types like trip plans and story replies.

Real-time communication is powered by Supabase's stream subscriptions, allowing the app to receive new messages instantly. With Riverpod's StreamProvider, the frontend listens to changes in the messages table filtered by conversation ID, so users only get updates for conversations they're part of.

Additional features include read receipts (tracked with read_by), typing indicators, soft-deleted messages (hidden only for selected users), and emoji reactions (stored as JSON).

For optimized loading, SQL procedures (get_chat_details, get_conversation_details) return ready-to-use data for each conversation, including participant info, avatars, and other important information to complete the chatting interface

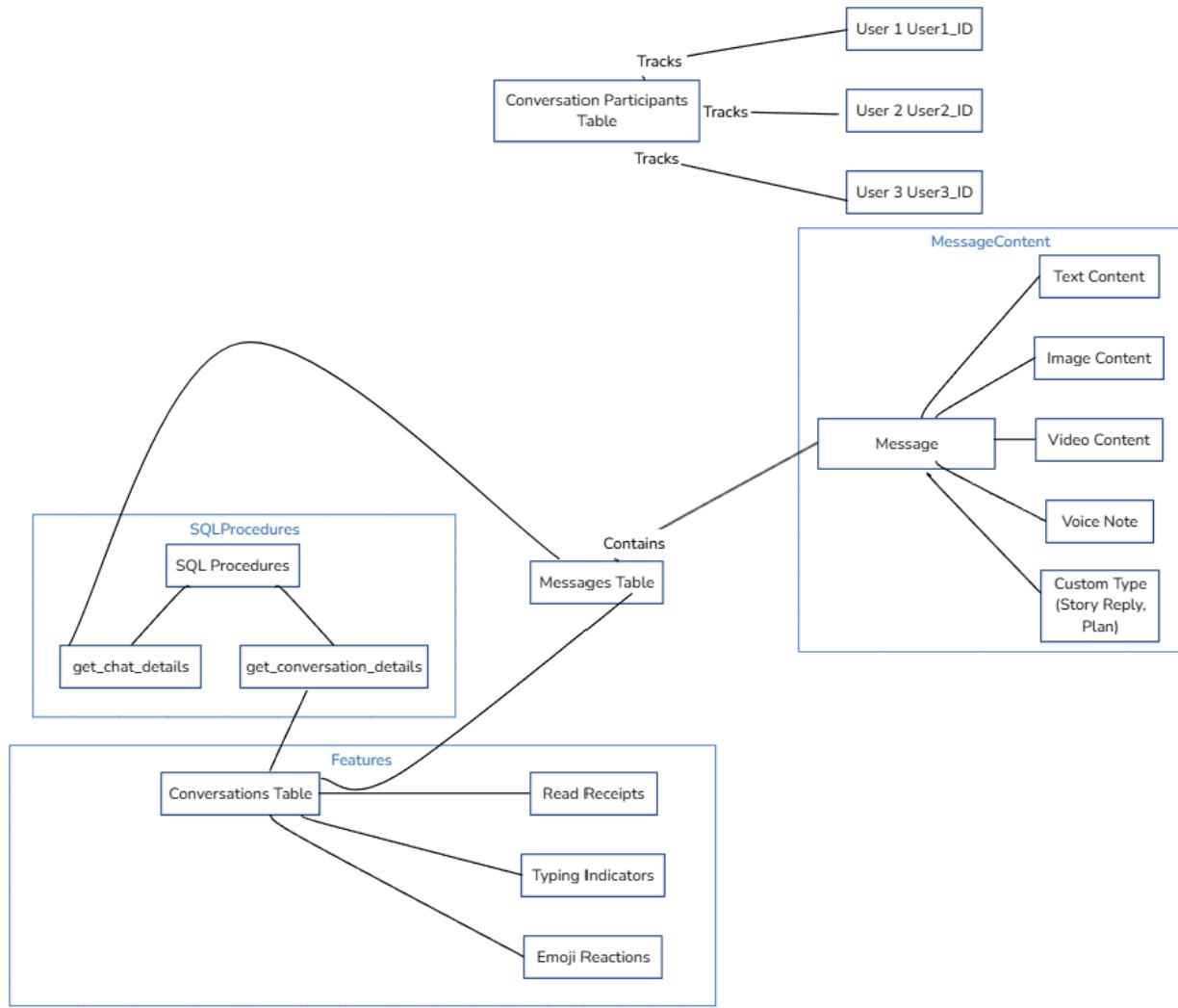


Figure 54: Development 6

7.2.1.10 Post - Message - Story Uploads:

To support media sharing across posts, messages, and stories, we developed a centralized MediaUploadService responsible for handling media compression, validation, uploading to Supabase Storage, and saving references in the database.

The service supports both images and videos, with optional compression to optimize file size before uploading. It automatically validates that media does not exceed 8 MB, notifies the user if it does, and generates a unique storage path using the user's ID and a timestamp.

For audio messages, a separate method handles uploading .mp3 files to a dedicated bucket. After a successful upload, the service returns a public URL for use in the app.

Once the media is uploaded, another method handles saving post metadata (e.g., caption, location, video thumbnail) directly to the posts table.

This unified system streamlines the media handling process across multiple features, resulting in a consistent and optimized user experience when posting content, sending messages, or sharing stories.

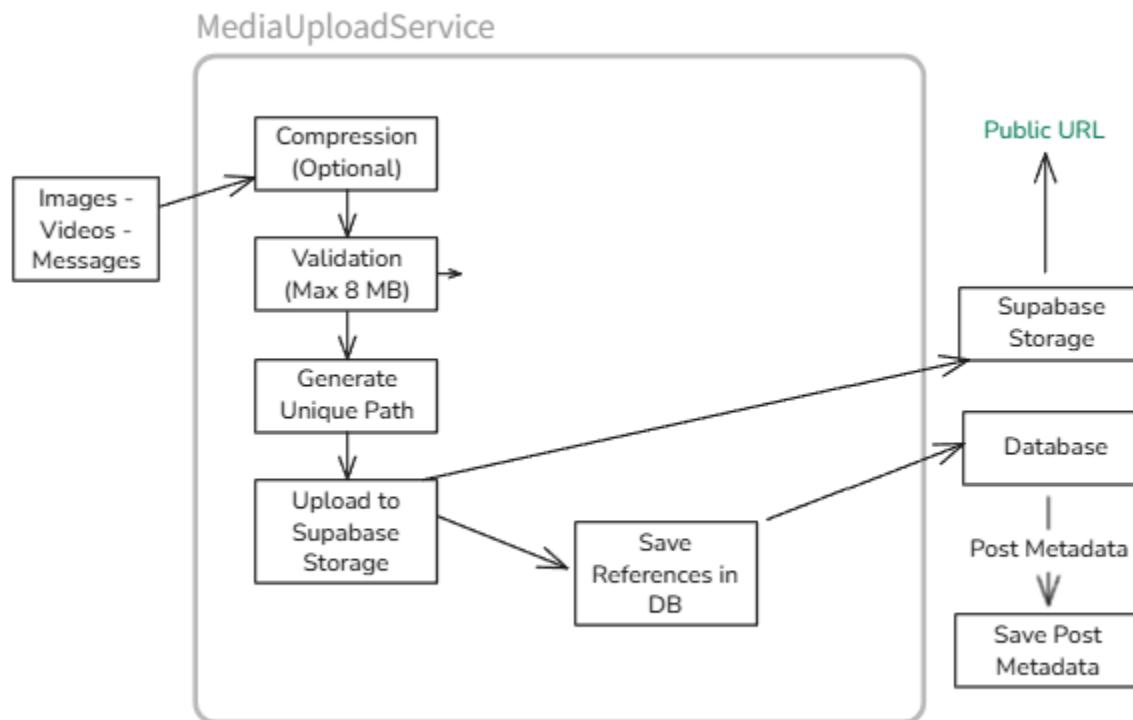


Figure 55: Development 7

7.2.1.11 Geolocation auto detection

To make a better UX experience, we have automatic detection for the current country the user is in. Useful for faster uploads since the post upload requires you to pick the country you are uploading the post from. And also useful for flight reservations to detect where the user is flying from.

To implement this feature, we used the geolocator and geocoding packages along with Riverpod state management. When the feature is triggered, a FutureProvider checks for location service availability and permission status. If permissions are granted, it retrieves the device's current latitude and longitude with high accuracy.

Using these coordinates, the system performs reverse geocoding to obtain the user's country name and ISO code. It then matches this information with a predefined country list to auto-select the user's current country. A dedicated StateProvider manages the detection state (e.g., detecting, success, failed, permission denied), allowing the UI to respond accordingly with appropriate feedback or fallbacks.

This architecture ensures that the country is automatically filled in without user input and enhances the user experience.

7.2.1.12 Comments and Replies

The comment system in TRAVIA supports threaded discussions with levels of nested replies, designed using a recursive rendering structure. Each comment can act as a parent to a list of replies, forming a comment tree.

To build this tree, we implemented a recursive function “_buildCommentTree” that takes a Comment object, a threaded comment map, and a depth parameter to control indentation. Each comment is rendered using a reusable CommentCard widget. Replies are dynamically toggled using a replyExpandProvider, which manages expansion state for each comment thread via Riverpod.

If replies exist, a "Show replies" or "Hide replies" toggle is shown. Upon interaction, the reply section animates the opening and closing and shows the replies or hides them accordingly, which provides a smooth and responsive user experience.

The recursive approach ensures scalability and handles deeply nested conversations while maintaining clean UI organization. It also minimizes rebuilds through fine-grained state control, resulting in efficient performance even in large comment trees.

7.2.1.13 Real time events Web-Scraping

To enhance the user experience with localized event suggestions, we implemented real-time web scraping to fetch upcoming events from Eventbrite based on the user's city.

The system uses a predefined mapping of popular cities to their corresponding Eventbrite URLs. When a user selects a city, the app sends a request to the Eventbrite page and parses the HTML to extract relevant data such as the event title, time, location, image, and link.

To ensure reliability, the scraper uses custom logic to filter out irrelevant elements (e.g., promotional tags like "selling fast") and extract meaningful content even when HTML structures vary. The implementation also handles edge cases like missing location or time data gracefully.

This lightweight scraping architecture allows us to deliver fresh, city-specific events directly inside the app without needing direct API access, which Eventbrite limits or restricts for public use.

7.2.1.14 Follow Friends Logic

To manage the "follow" feature between users in our social network, we developed a reactive architecture using Riverpod, Supabase, and FirebaseAuth.

We implemented a FollowController that handles follow/unfollow actions optimistically; meaning the UI updates instantly before the backend confirms the change as explained above. We also track the loading state per user to prevent repeated taps or race conditions.

The follow logic relies on:

A followStateProvider for caching each follow relationship (userA -> userB)

A followLoadingProvider for managing UI loading indicators

A FollowService that wraps Supabase RPCs to update the database and send notifications

A FollowStateNotifier that fetches and updates follow states per user efficiently

This system ensures a single source of truth across the app while gracefully handling blocked users, rollback on failures, and real-time visual feedback.

7.2.1.15 Online Status and Last Active

To display accurate online presence and last active timestamps across chats, we developed a background-aware online status system using app lifecycle tracking.

The `UserPresenceService` monitors whether a user is signed in, active, or backgrounded:

When the app is resumed, the user is marked as online.

When paused or closed, the user is marked offline, and their `last_active_at` timestamp is updated.

A heartbeat runs every 90 seconds to keep `last_active_at` fresh while the user is active.

We also use Riverpod providers to:

Stream live online status for any user (`userOnlineStatusProvider`)

Fetch the last active timestamp (`userLastActiveProvider`)

This design ensures reliable and real-time visibility of user availability within conversations while minimizing performance overhead.

7.2.1.16 Message Encryption

Our messages are encrypted for user safety and to ensure that even platform administrators cannot read private chats. We achieve this using AES (Advanced Encryption Standard) symmetric key encryption, which encrypts and decrypts messages using a conversation-specific key.

When a user sends a message, the system generates a unique encryption key derived from the conversation ID and a secure application salt. This salt is stored securely in environment variables, not in the codebase, to ensure protection against reverse engineering.

Here's a summary of the encryption process:

A 32-character encryption key is generated using a SHA-256 hash of the conversation ID combined with the hidden salt.

Each message is encrypted with AES encryption in CBC mode, using a randomly generated IV (Initialization Vector) for every message.

The encrypted message is stored in the format: iv:encryptedContent (both base64 encoded).

On retrieval, the system uses the same key and IV to decrypt the message content safely and accurately.

This approach guarantees:

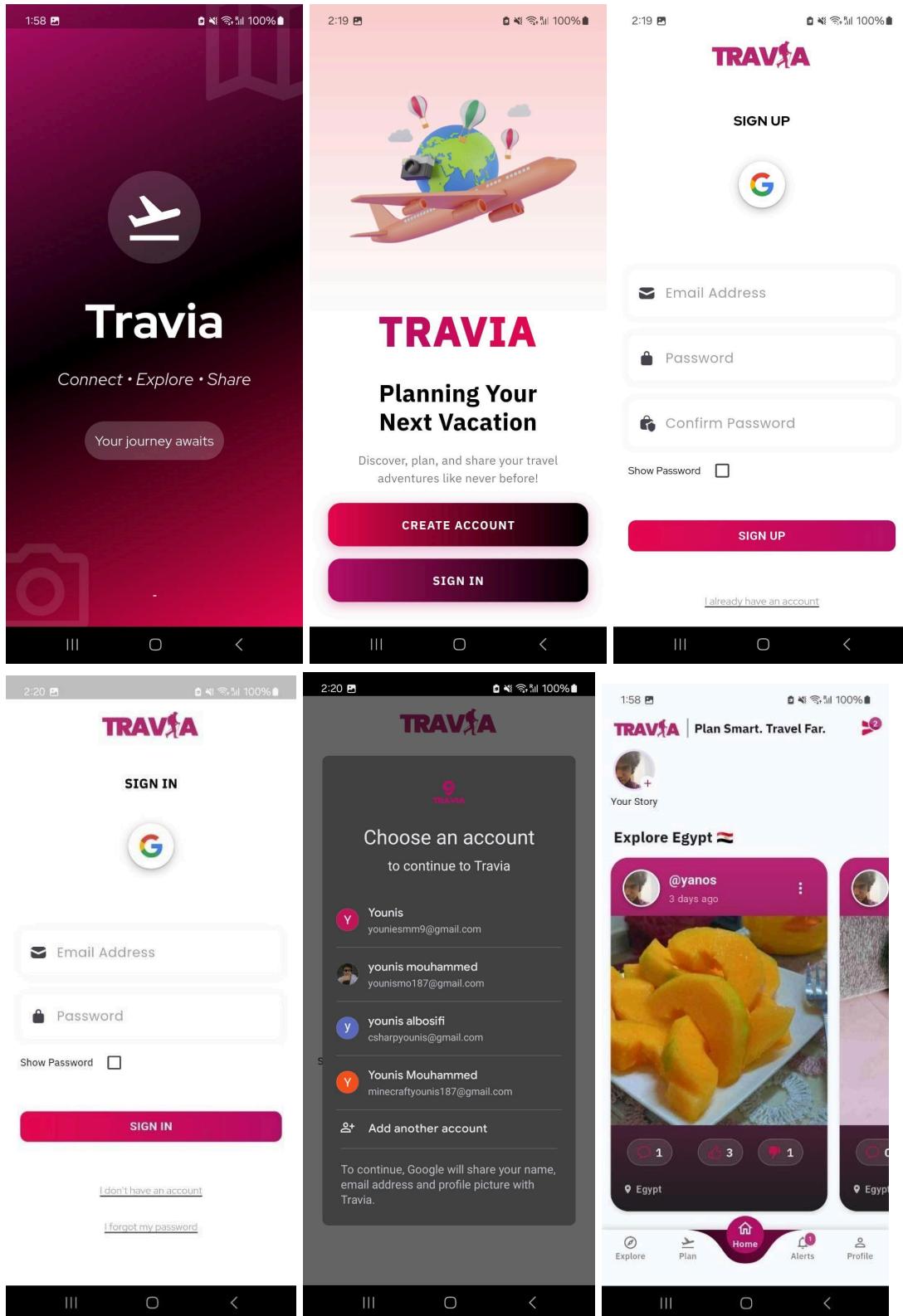
End-to-database encryption: Messages are encrypted before being sent to the server and decrypted only upon retrieval.

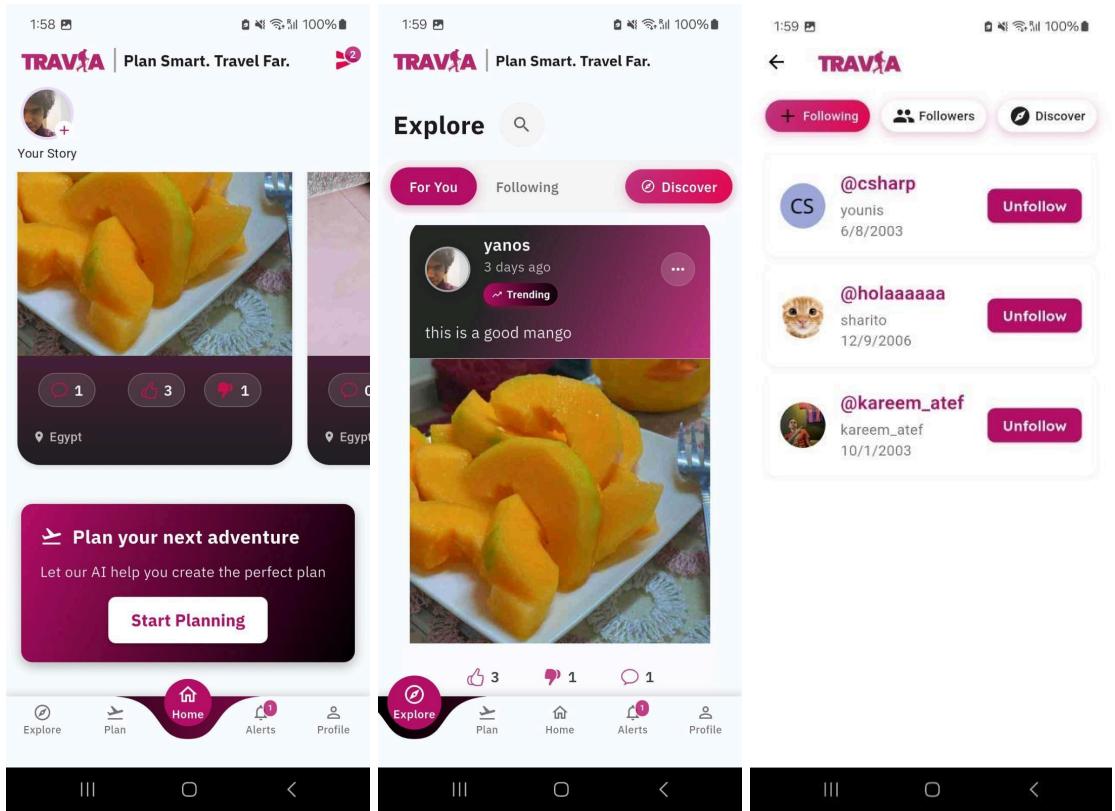
Admin-inaccessible messages: Even database access does not reveal actual message content.

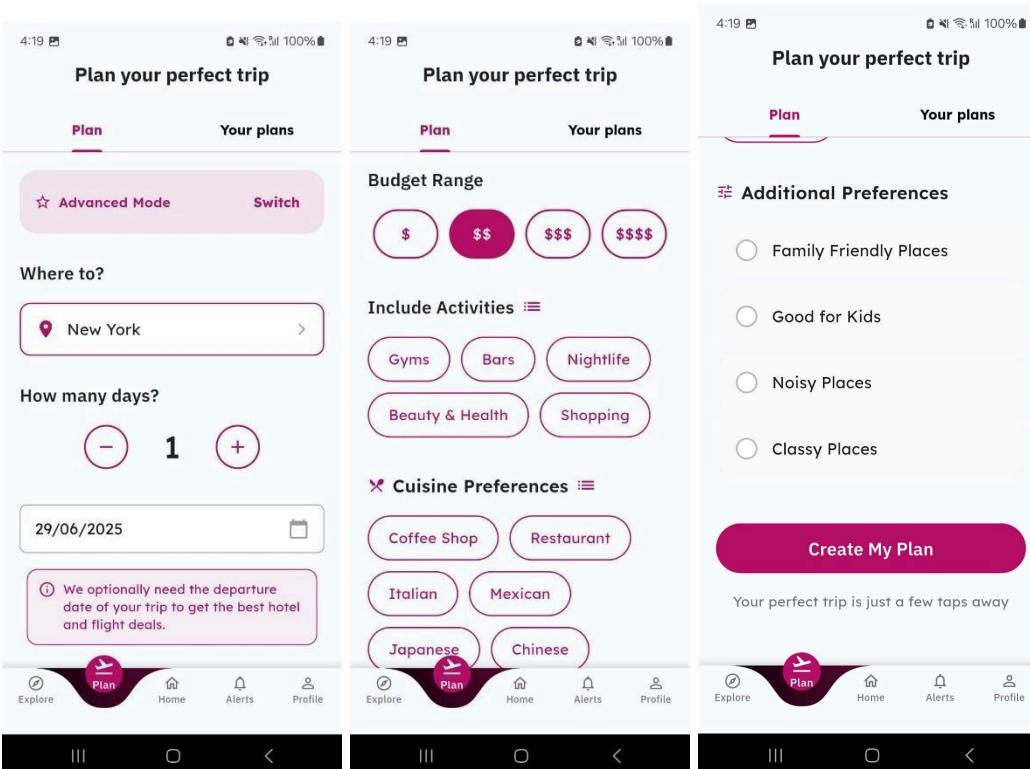
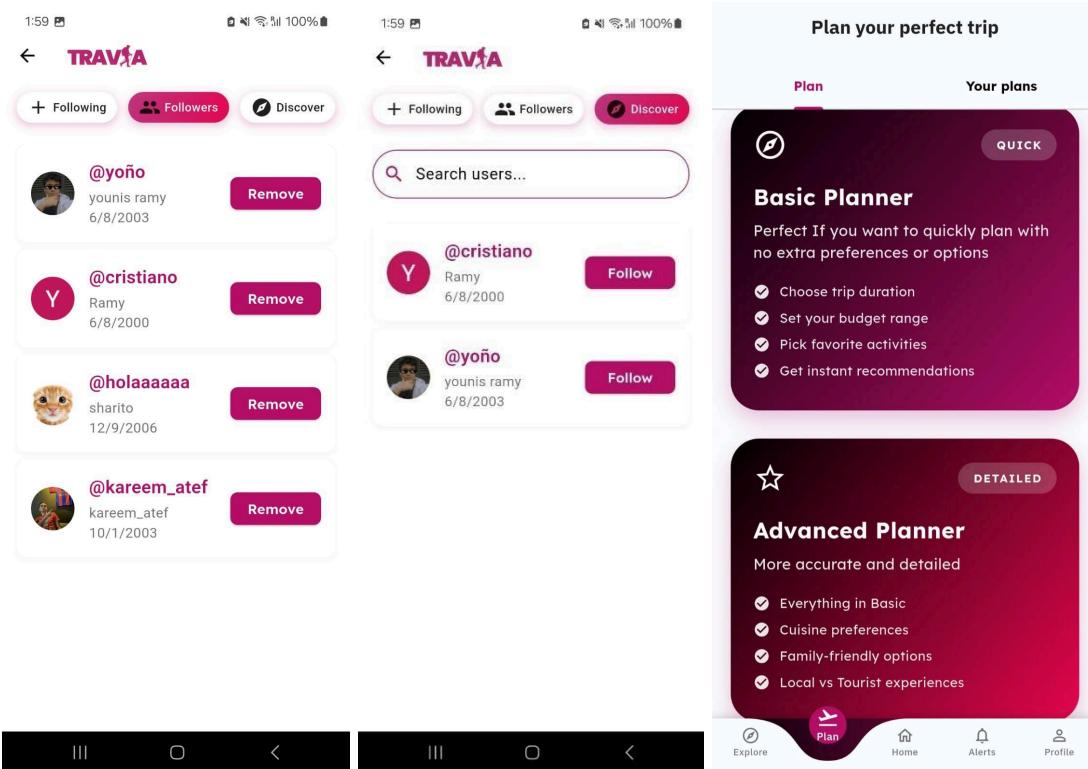
Conversation-specific isolation: Each conversation has its own encryption key, enhancing privacy.

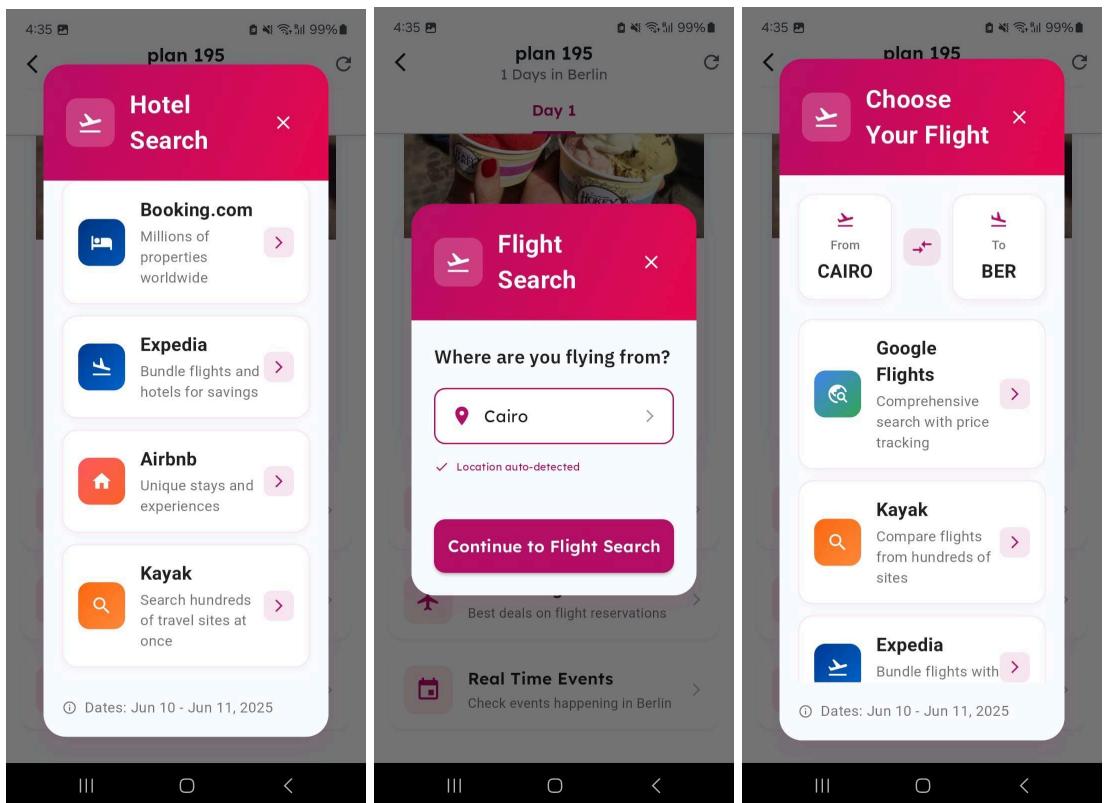
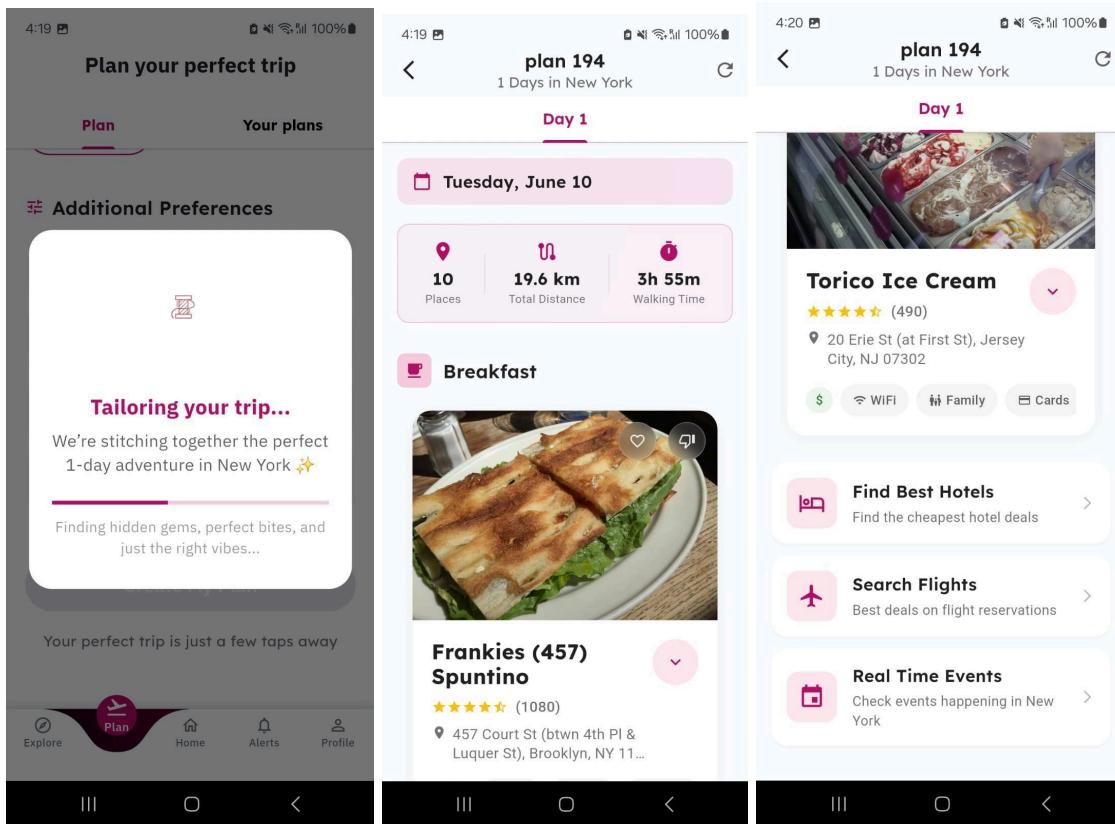
This setup lays the groundwork for transitioning to full end-to-end encryption (E2EE) in future work.

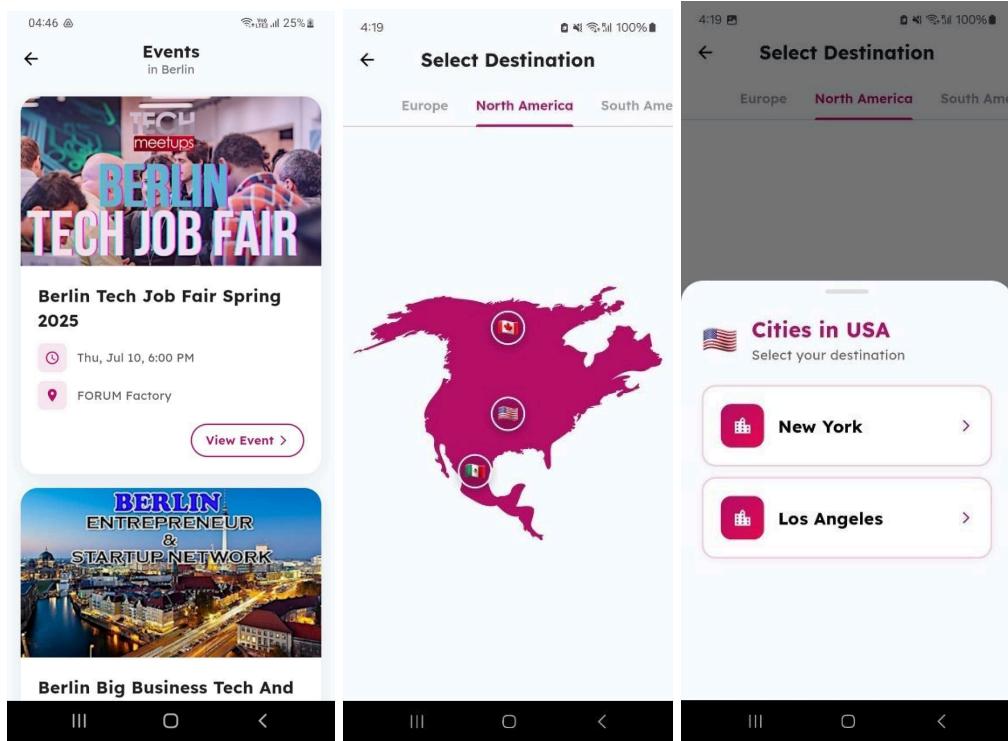
7.2.2 Finished Mobile App Showcase

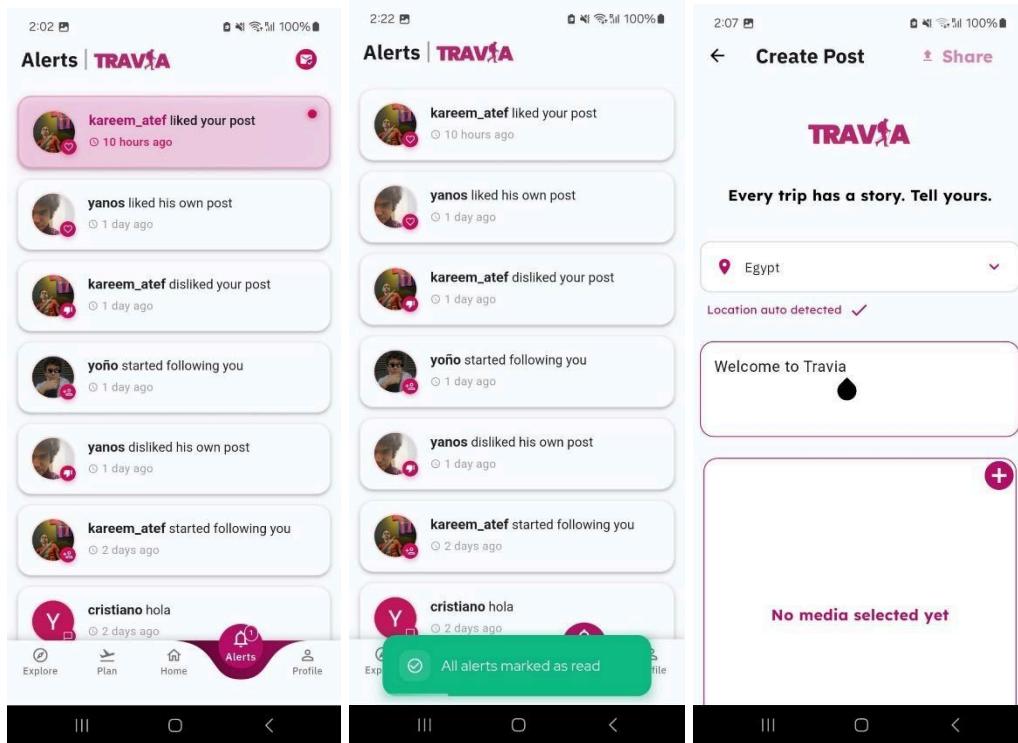
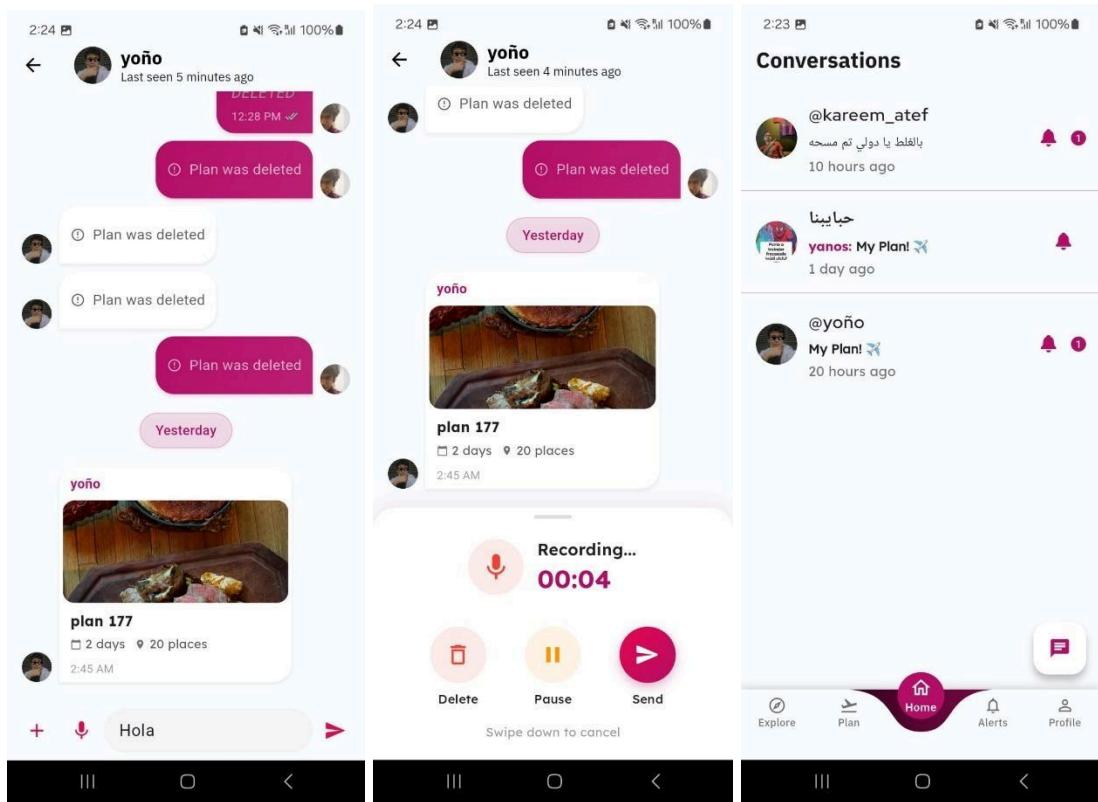


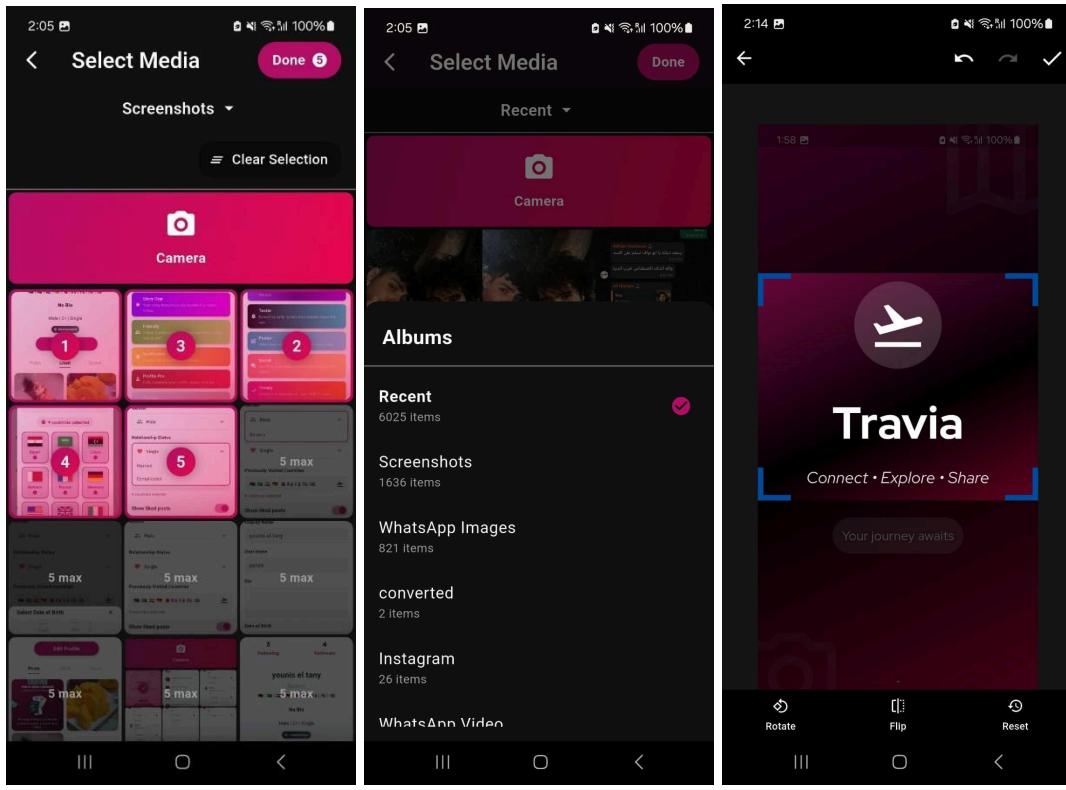


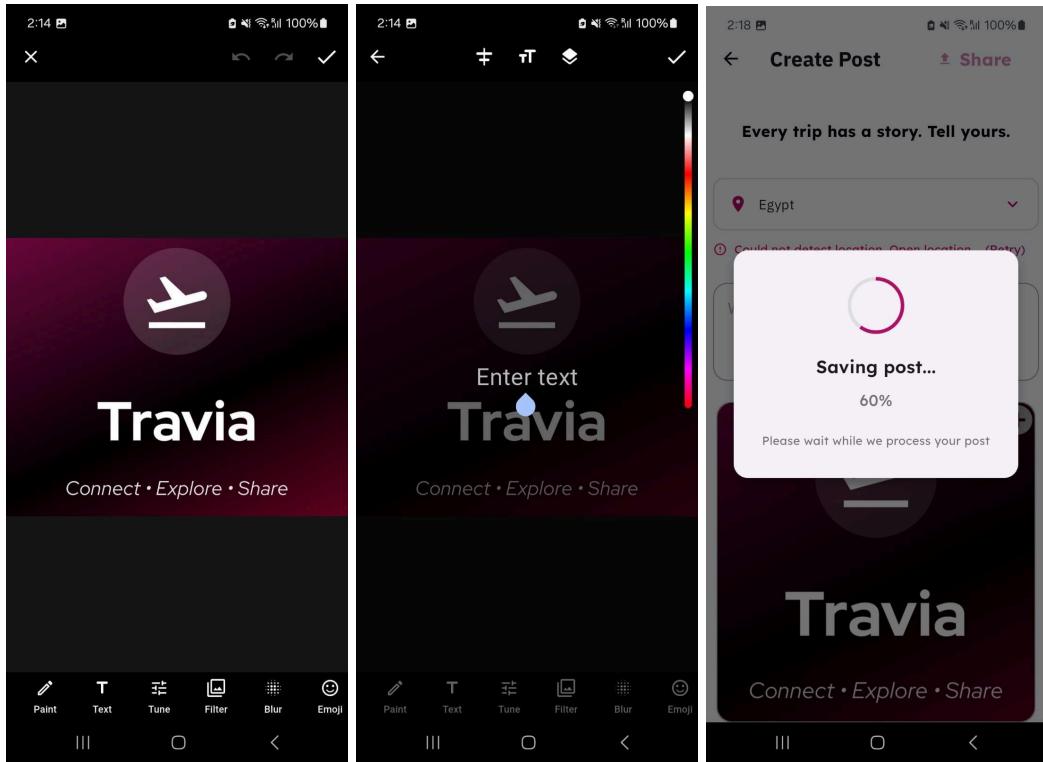




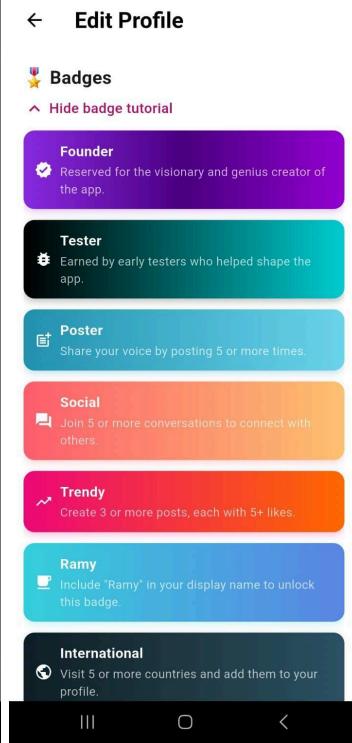
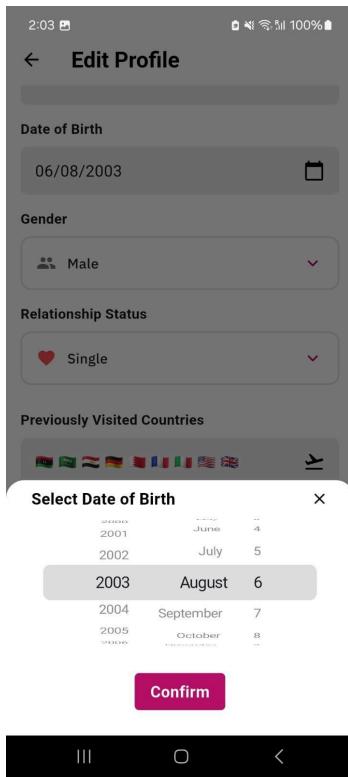


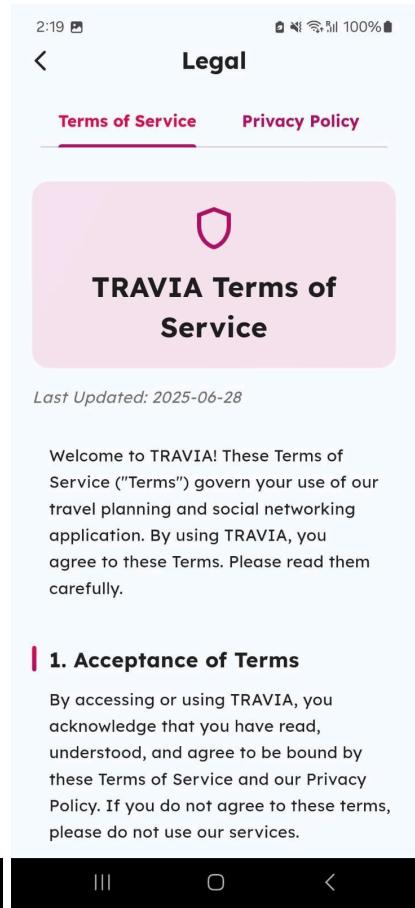
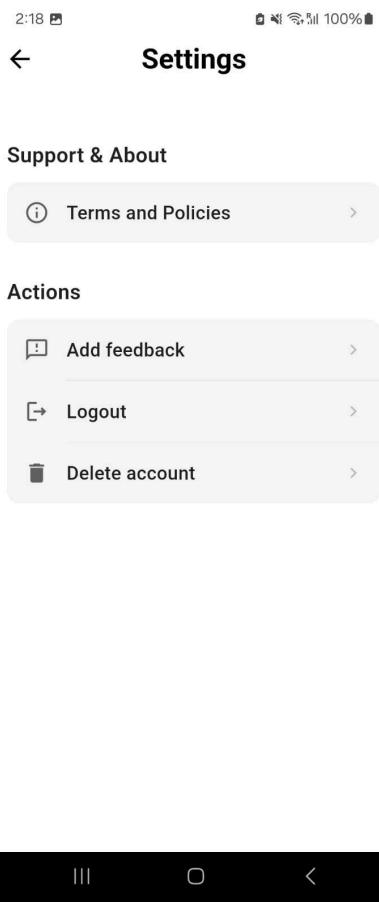
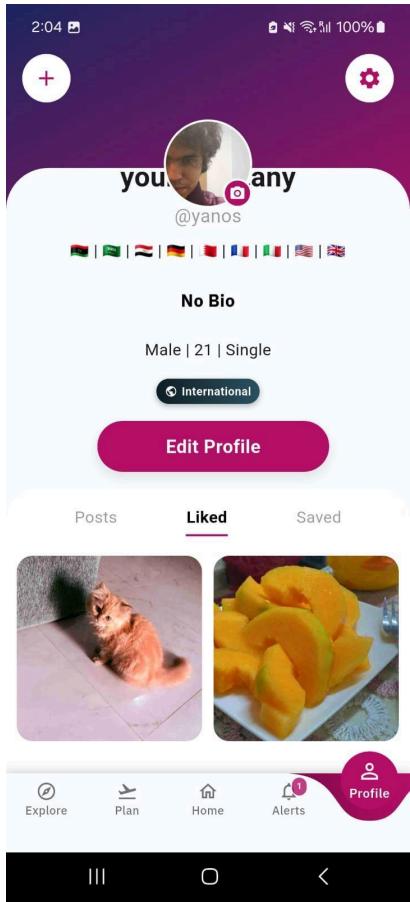


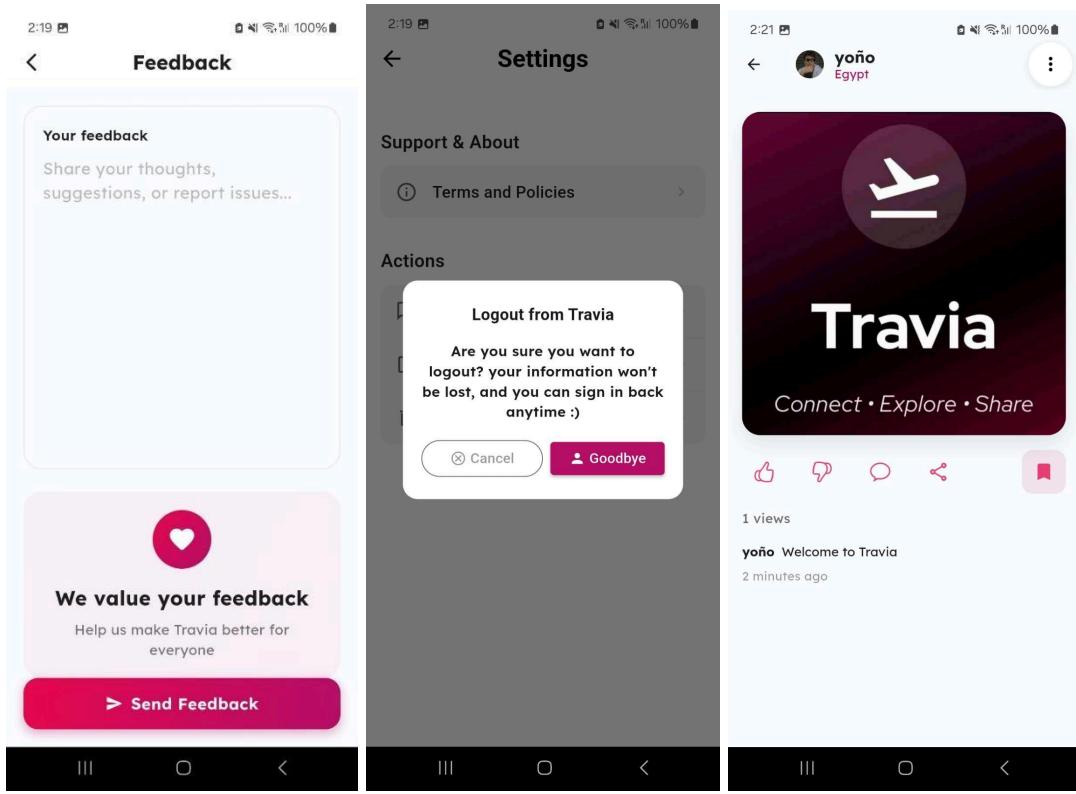


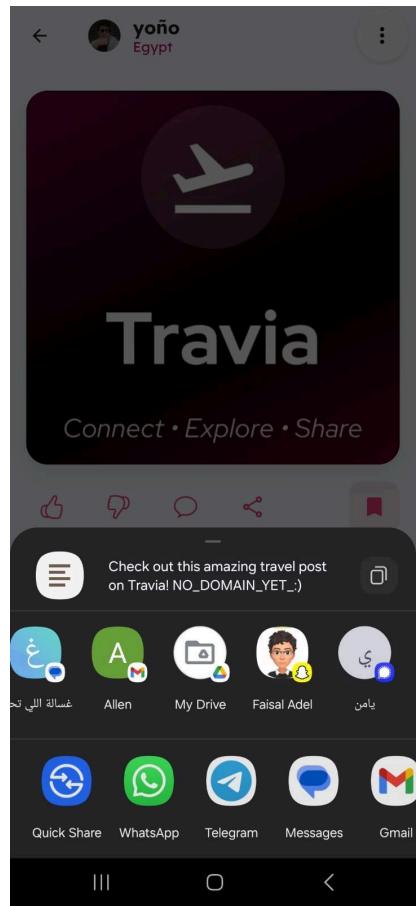


The left screenshot shows a user profile for "younis el tany" (@yanos). It displays 3 Following and 4 Followers. The bio section says "No Bio". Below it, the user is listed as "Male | 21 | Single" and "International". There is a prominent "Edit Profile" button. The bottom navigation bar includes "Posts" (selected), "Liked", "Saved", "Explore", "Plan", "Home", and "Alerts". The right screenshot shows the "Edit Profile" screen. It includes fields for "Display Name" (younis el tany), "User name" (yanos), "Bio" (empty), "Date of Birth" (06/08/2003), "Gender" (Male), "Relationship Status" (Single), "Previously Visited Countries" (with a list of 9 countries and a flight icon), "Show liked posts" (toggle switch), "Badges" (with a link to "How to earn badges"), and a "Save changes" button. The right side of this screenshot shows a comparison profile for "younis el tany" with the same details.









7.3 TESTING

7.3.1 Test App Vision:

The goal of testing in this project was to identify major functional and usability issues before launch. Instead of relying on automated test frameworks, we focused on practical real world feedback by involving users early. This approach allowed us to uncover edge-case bugs and performance bottlenecks that might not have been caught through traditional methods alone.

7.3.1.1 Our Test Methodology:

Our testing strategy was exploratory and feedback-driven. It consisted of:

Manual testing by the developer during development (often referred to as "monkey testing") to quickly check feature behavior, navigation, and UI layout.

Real-user testing with over 30 users who received the APK file directly and were encouraged to explore the app freely, report issues, and suggest improvements.

This informal but broad testing approach was valuable in identifying a large number of bugs related to:

Crashes on edge cases (e.g., uploading large videos)

UI responsiveness and layout inconsistencies

Navigation logic flaws

Inconsistent media handling across devices

Unexpected behavior in the planner and messaging features

7.3.1.2 Our Test results:

Testing revealed several critical bugs and areas for improvement, thanks to the feedback from early users. Common issues included:

Unresponsive buttons or broken navigation paths

Performance issues during media uploads or scrolling

Logic bugs in features like following, comments, and geolocation

Missing validations in authentication flows

Planner instability on certain devices

Despite these challenges, testing helped us drastically improve stability and usability before a wider release. Most reported issues were documented and resolved iteratively during development.

7.3.2 Testing the System:

To ensure that our project meets all functional requirements and operates as intended, we will thoroughly test detailed use cases. These use cases include: (User Sign Up, User Log in, User Access Home Page, User Upload a post, User Upload a Story, User Send/Receive Messages , User Like / Dislike / Comment on a post, User View/Edit Own Profile , User View Other Profile , User Access Settings Page , User View Notification Page, User Use Basic Planner, User Use Advanced Planner)

7.3.2.1 User Sign Up Test Case:

Table 20 : User Sign up Test case

Test Scenario	Expected Output	Actual Output	Result
Enter valid credentials	Sign Up Successful (Redirect to Home page)	Sign Up Successful (Redirect to Home page)	Pass
Enter already used E-mail	Error message (E-mail address already used)	Error message (E-mail address already used)	Pass
Enter mismatching passwords	Error message (Passwords do not match)	Error message (Passwords do not match)	Pass
Leave field(s) empty	Error message(s)	Error message(s)	Pass

7.3.2.2 User Log in Test Case:

Table 21 : User Log in Test case

Test Scenario	Expected Output	Actual Output	Result
Enter valid email and password	Login Successful (Redirect to Home Page)	Login Successful (Redirect to Home Page)	Pass
Enter incorrect password	Error message (Incorrect password)	Error message (Incorrect password)	Pass
Enter unregistered email	Error message (Account not found)	Error message (Account not found)	Pass
Leave email or password field empty	Error message (Field required)	Error message (Field required)	Pass

7.3.2.3 User Access Home Page Test Case:

Table 22 : User Access Home Page Test case

Test Scenario	Expected Output	Actual Output	Result
Access Home Page after successful login	Home Page loads with story bar, posts feed, and navigation bar visible	Home Page loads with story bar, posts feed, and navigation bar visible	Pass
Click "View Stories"	User's own and followed users' stories appear	User's own and followed users' stories appear	Pass
Click "Upload Story" icon	Media picker opens (max 5 media selection)	Media picker opens (max 5 media selection)	Pass
Click "Upload Post" icon	User redirected to upload post screen	User redirected to upload post screen	Pass
Interact with "Recommended for You" section	Recommended posts appear based on interaction score	Recommended posts appear based on interaction score	Pass
Tap "Explore by Country" tab	Posts filtered and displayed based on selected country	Posts filtered and displayed based on selected country	Pass
Click "DM" icon	User navigated to Direct Messages section	User navigated to Direct Messages section	Pass
Pull to refresh feed	Home feed refreshes with latest posts and stories	Home feed refreshes with latest posts and stories	Pass
Access Home Page without login	Redirect to login screen or show unauthorized access message	Redirect to login screen	Pass

7.3.2.4 User Upload a post Test Case:

Table 23 : User Upload a Post Test case

Test Scenario	Expected Output	Actual Output	Result
Select a single valid media and caption, then upload	Post uploaded successfully and visible in feed/profile	Post uploaded successfully and visible in feed/profile	Pass
Select unsupported media format	Error message (Unsupported media format)	Error message (Unsupported media format)	Pass
Skip media selection	Error message (No media selected)	Error message (No media selected)	Pass
Media flagged by inappropriate media recognition	Upload blocked with alert message	Upload blocked with alert message	Pass
Metadata not extracted properly from caption	Warning (Metadata extraction failed, post still uploaded)	Warning (Metadata extraction failed, post still uploaded)	Pass
Network loss during upload	Upload fails with retry option	Upload fails with retry option	Pass

7.3.2.5 User Upload a Story Test Case:

Table 24 : User Upload a Story Test case

Test Scenario	Expected Output	Actual Output	Result
Select up to 5 media files	Media editor opens with selected files	Media editor opens with selected files	Pass
Try to select more than 5 media	Error message: "You can only select up to 5 items."	Error message: "You can only select up to 5 items."	Pass
Apply edits to media (paint, crop, etc.)	Edits applied and preview shown before upload	Edits applied and preview shown before upload	Pass
Upload media > 8MB	Error message: "Media exceeds 8MB limit."	Error message shown	Pass
Upload valid media after compression	Story is uploaded and visible in Story bar	Story is uploaded and visible in Story bar	Pass
Inappropriate content detected	Upload blocked with warning: "This media violates our guidelines."	Upload blocked with warning	Pass
Network disconnect during upload	Upload fails, user sees retry option	Upload fails, user sees retry option	Pass
Story disappears after 24 hours	Story is automatically removed from Story bar and Supabase	Story is removed after 24 hours	Pass

7.3.2.6 User Send/Receive Messages Test Case:

Table 25 : User Send/Receive Messages Test case

Test Scenario	Expected Output	Actual Output	Result
Open DM screen from home page	Chat list loads with last messages & unread counts.	Chat list loads with last messages & unread counts.	Pass
Search for a user outside follower list	Search results show matching usernames; “Start chat” button enabled.	Results displayed; “Start chat” enabled.	Pass
Create new 1-to-1 chat and send a text	Message appears in own thread; recipient receives it in real time.	Message delivered instantly to both users.	Pass
Create group chat (select 3 users, name group)	Group created; all members listed; system sends “Group created” system message.	Group created, system message shown.	Pass
Send more than 5 media files in one message	Error: “You can attach up to 5 files per message.”	Error displayed.	Pass
Send ≤ 5 images/videos	Media thumbnails appear in chat; recipients receive them.	Media displayed & delivered.	Pass
Record voice message, pause, then resume & send	Timer pauses; resumes; final clip sent and playable.	Works as expected.	Pass
Swipe down to cancel voice recording	Recording discarded; no audio sent.	Recording cancelled; no audio sent.	Pass
Reply to a specific message in the thread	New message shows quoted preview of referenced message for all participants.	Quoted preview displayed.	Pass
Recipient goes offline during send	Message stored; shows single-tick “sent” status; delivers when recipient reconnects.	Stored; delivered on reconnect.	Pass
View typing indicator from another user	“User is typing...” banner appears until message sent or stopped.	Indicator appears / disappears correctly.	Pass
View read receipt after recipient opens message	Double-tick/“seen” badge updates in real time.	Badge updates in real time.	Pass
View online / offline status	Green dot (online) / grey dot (offline) reflects real-time presence.	Status icons accurate.	Pass
Attempt to message a user who has blocked traveler	Error: “Cannot send message – you’re blocked by this user.”	Error displayed.	Pass
Network loss while sending	“Failed to send” badge shown with retry option.	Failure badge + retry shown.	Pass

7.3.2.7 User Like / Dislike / Comment on a post Test Case:

Table 26 : User Like / Dislike / Comment on a post Test Case

Test Scenario	Expected Output	Actual Output	Result
Like a post	Post gets liked; like count increases; like icon updates.	Post liked; count increases; icon updates.	Pass
Dislike a post	Post gets disliked; dislike count increases; like icon deselects.	Post disliked successfully.	Pass
Like a post that's already liked	Like is removed (toggle off); like count decreases.	Like toggled off; count decreases.	Pass
Dislike a post that's already disliked	Dislike is removed (toggle off); dislike count decreases.	Dislike toggled off; count decreases.	Pass
Comment on a post	Comment appears instantly under post; stored in DB.	Comment posted and visible.	Pass
Like a comment	Comment like count increases; icon updates.	Like count increases.	Pass
Reply to a comment	Reply nested under original comment.	Reply displayed under parent comment.	Pass
Report a comment	Comment is flagged; moderation system notified.	Comment reported successfully.	Pass
Share post to another user	Share menu opens; post sent or link copied.	Post shared or link copied successfully.	Pass
Save post	Post added to user's saved posts list.	Post saved.	Pass
Save post again (already saved)	Save toggles off; post removed from saved list.	Post unsaved.	Pass
Try to like & dislike simultaneously	Only one is allowed; previous is removed.	Toggle switches automatically.	Pass
Comment with empty content	Error message: "Comment cannot be empty."	Error displayed.	Pass
Network loss during action	Error shown with retry or auto-retry on reconnect.	Action failed; retry shown.	Pass

7.3.2.8 User View/Edit Own Profile Test Case:

Table 27 : User View/Edit Own Profile Test

Test Scenario	Expected Output	Actual Output	Result
Open profile page from navigation bar	Profile loads with user data: photo, username, bio, posts, badges, etc.	Profile loads with full data as expected.	Pass
Scroll through "Posts" tab	User's uploaded posts are displayed.	Posts shown correctly.	Pass
Switch to "Saved" tab	Saved posts displayed.	Saved posts visible.	Pass
Switch to "Liked" tab	Liked posts shown if visibility toggle is enabled.	Liked posts appear.	Pass
Toggle off liked posts visibility	Liked posts tab becomes hidden from other users.	Liked tab hidden from public view.	Pass
View badges (e.g., Founder, Adventurer)	Badges appear based on interaction milestones.	Badges displayed accurately.	Pass
Tap "Edit Profile"	Edit screen opens with editable fields.	Edit screen opens.	Pass
Update display name, username, bio	Fields updated and saved.	Changes reflected immediately.	Pass
Change profile picture	New image uploaded and displayed.	Profile picture updated.	Pass
Update gender, DOB, relationship status	Values updated and stored.	Changes saved correctly.	Pass
Update visited countries	New countries saved and shown in profile.	Data stored and displayed.	Pass
Leave a required field blank	Error message shown (e.g., "Username is required").	Field validation error displayed.	Pass
Choose a taken username	Error message: "This username is already in use."	Username conflict handled properly.	Pass
Save edits with valid data	Confirmation message: "Profile updated successfully."	Success message shown.	Pass
Network disconnected during edit	Save fails, error displayed: "Unable to update profile. Check connection."	Error shown with retry.	Pass

7.3.2.9 User View Other Profile Test Case:

Table 28 : User View Other Profile Test Case

Test Scenario	Expected Output	Actual Output	Result
Tap on a user's profile image or username	Redirects to selected user's public profile	Profile loads correctly	Pass
View posts on other user's profile	Traveler can scroll through public posts	Posts are visible	Pass
View user's profile picture, bio, badges, country info	Displayed in profile header section	Information appears correctly	Pass
Tap on "Follow" button	Traveler follows the user; count updates	Follow successful; count updated	Pass
Tap "Unfollow" button	Traveler unfollows the user; count updates	Unfollow successful	Pass
View visited countries and interaction badges	Public country list and badges are shown if not hidden	Country list and badges visible	Pass
Attempt to view profile of deactivated user	Message: "User not found or profile unavailable"	Error displayed	Pass
Try to access private profile (if implemented)	Message: "This profile is private" or follow request sent	Access restricted as expected	Pass
Search for another user and open profile	Matching user appears in search; profile opens	Search and profile load correctly	Pass
Network disconnected while viewing	Error: "Unable to load profile. Please check your connection."	Error shown	Pass

7.3.2.10 User Access Settings Page Test Case:

Table 29 : User Access Settings Page Test Case

Test Scenario	Expected Output	Actual Output	Result
Tap settings icon from profile page	Settings menu opens with options (Support Us, Add Feedback, Terms & Policies, Logout)	Menu displayed correctly	Pass
Tap “Support Us” option	Redirects to external donation/support link	External page opens	Pass
Tap “Add Feedback”	Opens feedback form or external link to submit feedback	Feedback form shown	Pass
Tap “Terms & Policies”	App displays full terms in a scrollable view or webview	Terms content opens	Pass
Tap “Logout”	User is logged out, session cleared, redirected to login screen	Logout successful, redirect works	Pass
Attempt to access settings when offline	Content shows or cached, but external links fail to load with message	Error message shown for failed external content	Pass
Attempt to logout when Firebase is down	Error: “Unable to logout at this time. Please try again.”	Error displayed	Pass
Exit settings screen without action	Returns to previous screen (profile page)	Navigation works as expected	Pass

7.3.2.11 User View Notification Page Test Case:

Table 30 : User View Notification Page Test Case

Test Scenario	Expected Output	Actual Output	Result
Tap notification icon from navigation bar	Notification list loads (likes, follows, messages, etc.)	Notification list displayed correctly	Pass
Tap on a “like” notification	Redirects to the post that was liked	Post opens as expected	Pass
Tap on a “new follower” notification	Redirects to follower’s profile page	User profile loads successfully	Pass
Tap on a “message” notification	Redirects to the relevant conversation in chat	Chat opens correctly	Pass
Tap on “mark all as read”	All notifications marked read visually and in DB	Notifications updated successfully	Pass
Swipe to delete a notification (if implemented)	Notification disappears from the list	Notification removed	Pass
Open notification list with no notifications	Message: “No notifications available.”	Message shown	Pass
Try to open deleted/invalid content via notification	Message: “Content no longer available.”	Error shown	Pass
Network disconnected while loading	Message: “Unable to load notifications. Please check your connection.”	Error displayed	Pass

7.3.2.12 User Use Basic Planner Test Case:

Table 31 : User Use Basic Planner Test Case

Test Scenario	Expected Output	Actual Output	Result
Select Basic Planner	Basic planner interface opens with input form	Interface loads	Pass
Enter optional preferences (category, budget, etc.)	Preferences accepted, used in generation	Inputs stored and passed to planner	Pass
No interaction data + no preferences	Prompt to add preferences or switch to Advanced Planner	Prompt shown	Pass
Receive trip plan	Plan with scored activities generated and shown	Plan shown	Pass
Tap activity: call, website, or directions	Opens phone app, browser, or Google Maps	External action executed	Pass
Search for nearby hotels	Hotels fetched via external API	Results displayed	Pass
Search flight deals	Flights loaded via integration	Data shown	Pass
View real-time events in location	Events list appears based on destination	Events shown	Pass
Like or dislike a plan	Feedback saved for RL model	Feedback saved	Pass
Save plan	Plan saved in Supabase	Confirmed save	Pass
Network failure during plan generation	Error shown: "Unable to generate plan. Try again later."	Error displayed	Pass

7.3.2.13 Use Advanced Planner Test Case:

Table 32 : Use Advanced Planner Test Case

Test Scenario	Expected Output	Actual Output	Result
Open Advanced Planner	Full preference form is displayed	Form appears	Pass
Fill all fields and submit	System generates multiple personalized trip plans	Multiple plans returned	Pass
Enter invalid values or leave required fields blank	Error: "Please complete all required fields."	Error shown	Pass
Submit without interaction history	Plans generated using only explicit preferences	Plans returned	Pass
Tap call, website, or directions for a suggested activity	External service opens as expected	Action completed	Pass
Search for hotels in trip location	Nearby hotels fetched via API	Hotels displayed	Pass
Search for flight deals	Flights returned from external source	Flights shown	Pass
Check real-time events in location	Events list loads correctly	Events appear	Pass
Like/dislike a generated plan	Feedback saved and used by RL model	Feedback saved	Pass
Save final plan	Trip plan stored in Supabase	Plan saved	Pass
Server or AI failure	Error: "Unable to generate plan right now."	Error displayed	Pass

8 CONCLUSION AND FUTURE WORK

8.1 Conclusion

The TRAVIA project represents a significant step forward in addressing the evolving needs of modern solo travelers by seamlessly integrating advanced AI technologies and social connectivity into a user-friendly travel planning application. Throughout the development journey, we explored innovative approaches to tackle existing gaps in the travel industry, such as fragmented planning processes, limited customization, and lack of social engagement.

TRAVIA's AI planner provides tailored solutions that accommodate diverse traveler preferences, budgets, and interests. By leveraging machine learning models, generative AI, and clustering techniques, the platform ensures personalized and efficient itinerary generation, dynamically adapting to real-time updates and user feedback. The integration of reinforcement learning for adaptive decision-making has enhanced the system's ability to refine recommendations continuously, ensuring alignment with user expectations.

The unique social media component elevates TRAVIA beyond traditional planning tools, fostering connections among travelers through shared itineraries, and compatibility metrics. This innovative feature transforms solo travel into a socially enriched experience, enabling users to form meaningful connections with like-minded explorers.

Extensive research and implementation of robust preprocessing techniques ensured the accuracy and efficiency of our AI models. We successfully navigated challenges related to data privacy, system scalability, and real-time performance by adhering to international standards and leveraging cloud-based technologies.

As we conclude this phase, TRAVIA stands as a comprehensive travel solution poised to redefine solo travel. It bridges the gap between travel planning and social networking, empowering users to plan, connect, and explore with ease. The insights gained during this project pave the way for future advancements, including enhanced personalization, improved real-time adaptation, and expansion into new markets. We are confident that TRAVIA will not only meet but exceed user expectations, offering an unparalleled travel experience for the modern explorer.

8.2 Future Work

- Deploying AI Models and back end on AWS or any other cloud platform for large scalability
- End to End encryption on messages
- Interactive Map that shows the nearby friends and the percentage of similarity of their plan and the user
- Use APIs for booking flights and hotels instead of links
- Use APIs for real time events for more accurate data
- Membership monetization and content creators monetization
- Making monolith recommendation systems for recommending social media posts
- Turning our planning service into an API Service to lease it to other systems for a subscription

Sources and References:

1. S. Li, Y. Zhang, H. Chen, and X. Wang, "Deep reinforcement learning in recommender systems: A survey and new perspectives," Engineering Applications of Artificial Intelligence, vol. 123, pp. 106-118, 2023, doi: 10.1016/j.engappai.2023.106118
2. L. Chen, H. Liu, S. Zhang, and Z. Wu, "Trip Reinforcement Recommendation with Graph-based Representation Learning," ACM Transactions on Knowledge Discovery from Data, vol. 17, no. 3, pp. 1-25, 2023, doi: 10.1145/3564609
3. A. Kumar, R. Sharma, D. Patel, and N. Singh, "A Deep Hybrid Model for Recommendation by jointly leveraging ratings, reviews and metadata information," Knowledge-Based Systems, vol. 214, pp. 106-120, 2021, doi: 10.1016/j.knosys.2020.106120
4. G. Ke, H. L. Du, and Y. C. Chen, "Cross-platform dynamic goods recommendation system based on reinforcement learning and social networks," Applied Soft Computing, vol. 104, pp. 107213, 2021, doi: 10.1016/j.asoc.2021.107213
5. M. Hassan, S. Ahmed, and T. Rahman, "A collaborative filtering recommendation framework utilizing social networks," Information Processing & Management, vol. 60, no. 3, pp. 103-118, 2023, doi: 10.1016/j.ipm.2023.103288

الملخص

TRAVIA هو تطبيق لخطيط السفر مدحوم بالذكاء الاصطناعي، تم تصميمه لتيسير وتحسين خطط الرحلات، خاصة للمسافرين الفردية. يقوم بدمج

أساليب البحث الشائعة عن السفر، مثل مخططات الرحلات ووسائل التواصل الاجتماعي، في منصة واحدة، ويقدم توصيات لمسارات الرحلات، وأحداث في الوقت

الفعلي، وميزات للتواصل الاجتماعي. على عكس الأدوات الحالية، يدمج TRAVIA وسائل التواصل الاجتماعي بشكل فريد، ويطابق المستخدمين ذوي

الخلفية المشابهة، ويتيح مشاركة التجارب لمساعدة الآخرين في اكتشاف الوجهات. من خلال تيسير الخطط، وتعزيز الروابط، وتحسين اكتشاف الرحلات،

يقدم TRAVIA حلًا قابلًا للتتوسيع يمتلك بإمكانات قوية في سوق السفر الذي تبلغ قيمته 800 مليار دولار.



الأكاديمية العربية للعلوم والتكنولوجيا والنقل البحري
كلية الحاسوب و تكنولوجيا المعلومات

علوم الحاسوب و هندسة البرمجيات

مشروع التخرج 2

Travia

مقدم من

كريم عاطف عبدالمقصود 211014291
يونس محمد عبدالرحمن 211006892
محمود جمال محمود 211009758
عبدالرحمن ابراهيم محمود 211002951
أحمد ايهاب السيد 211014027
عبدالرحمن رمضان بكري 212004117

تحت إشراف
د. فهيمة مغربي

يوليو-2025