AI Travel Companion

Project Team

Muhammad Taimoor Hassan	20P-0603
Muhammad Awais	20P-0107
Ammaar Naeem Laghari	20P-0180

Session 2020-2024

Supervised by

Mr. Shahzeb Khan

Department of Computer Science

National University of Computer and Emerging Sciences
Peshawar, Pakistan

June, 2024

Student's Declaration

We declare that this project titled "*AI Travel Companion*", submitted as requirement for the award of degree of Bachelors in Computer Science, does not contain any material previously submitted for a degree in any university; and that to the best of our knowledge, it does not contain any materials previously published or written by another person except where due reference is made in the text.

We understand that the management of Department of Computer Science, National University of Computer and Emerging Sciences, has a zero tolerance policy towards plagiarism. Therefore, We, as authors of the above-mentioned thesis, solemnly declare that no portion of our thesis has been plagiarized and any material used in the thesis from other sources is properly referenced.

We further understand that if we are found guilty of any form of plagiarism in the thesis work even after graduation, the University reserves the right to revoke our BS degree.

Muhammad Taimoor Hassan Signature: _____

Muhammad Awais Signature: _____

Ammaar Naeem Laghari Signature: _____

_____

Verified by Plagiarism Cell Officer

Dated:

Certificate of Approval

The Department of Computer Science, National University of Computer and Emerging Sciences, accepts this thesis titled *AI Travel Companion*, submitted by Muhammad Taimoor Hassan (20P-0603), Muhammad Awais (20P-0107), and Ammaar Naeem Laghari (20P-0180), in its current form, and it is satisfying the dissertation requirements for the award of Bachelors Degree in Computer Science.

Supervisor

Mr. Shahzeb Khan Signature: _____

Mr. Haroon Zafar

FYP Coordinator
National University of Computer and Emerging Sciences, Peshawar

---

Mr. Fazl-E-Basit

HoD of Department of Computer Science
National University of Computer and Emerging Sciences

Acknowledgements

We would like to express our deepest gratitude to all those who have supported and contributed to the completion of our project. We are delighted to acknowledge the valuable assistance and guidance we have received throughout this initiative.

First and foremost, we would like to express our sincere thanks to our supervisor, Mr. Shahzeb Khan for their expertise, unwavering encouragement, and unwavering support have been instrumental in shaping our work and enabling us to achieve our goals. We are eternally grateful for their advice, wisdom, and mentorship throughout this journey.

We would also like to thank Mr. Shahzeb Khan for their contributions to our research. Their experience and understanding of Machine Learning and Artificial Intelligence, have been invaluable resources in our research. We are deeply appreciative of their time, advice, and willingness to share their knowledge with us.

Lastly, we would like to express our gratitude to our institute for their continuous support and understanding. Their faith in our abilities has been a constant source of strength and motivation throughout this process.

Thank you all for your invaluable contributions to our project.

Muhammad Taimoor Hassan

Muhammad Awais

Ammaar Naeem Laghari

Abstract

Various sectors have been dominated by the Artificial Intelligence (AI), travel industry has yet to fully explore the potential of AI. To address the industry demands, we propose a solution to address this void by introducing "Jaaoo", an AI based travel planning mobile application. Our application offers users to create personalized travel plans through an interactive chatbot, it will provide a complete travel plan for various countries with best available options of flights, hotels, restaurants, tourist spots and everything you need for your traveling vacation. With complete plan, app also provide day to day latest news of the destinations to ensure the pleasant and comfortable vacation of our users. Users can also choose from a selection of pre-created travel plans for various destinations. Upon receiving the travel plan, seamless redirection to third-party applications like Airbnb facilitates swift and secure booking processes. The technicalities of our application rely on Flutter for mobile app, Firebase for efficient data management, and Pinecone database for integration of AI features.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Traveling is an important part of everyone life, it opens doors to discover yourself and the wonders of this amazing world. It provides an exposure to diverse cultures traditions, lifestyles and definitely helps in personal growth. It helps you to see the different perspective of life and bridges you to connect with nature. Traveling in groups strengthen the existing relationships, makes the bond everlasting. These qualities and our passion of traveling motivated us to simplify the life of travel enthusiasts like us.

Traveling is amazing and provides a learning experience but it is also a rollercoaster of joy and challenges. There are various problems related to traveling like booking tickets, find accommodations and many more. To tackle these problems, there are solutions available online in the form of ticketing and accommodations apps but they bring a new problem of endless options. There are many apps for different uses cases such as Airbnb for rooms and Booking.com for tickets. This is very much time consuming which results to be a unpleasant experience for most of the users.

Various sectors have been dominated by the Artificial Intelligence (AI), travel industry has yet to fully explore the potential of AI. To address the industry demands, we propose a solution to address this void by introducing "Jaaoo", an AI based travel planning mobile application. Our application offers users to create personalized travel plans through an

interactive chatbot, it will provide a complete travel plan for various countries with best available options of flights, hotels, restaurants, tourist spots and everything you need for your traveling vacation. With complete plan, app also provide day to day latest news of the destinations to ensure the pleasant and comfortable vacation of our users. Users can also choose from a selection of pre-created travel plans for various destinations. Upon receiving the travel plan, seamless redirection to third-party applications like Airbnb facilitates swift and secure booking processes. The technicalities of our application rely on Flutter for mobile app, Firebase for efficient data management, and Pinecone database for integration of AI features. AutoGPT, LangChain, and Falcon LLM enhance the chatbot's Natural Language Processing (NLP) capabilities.

1.2 Methodology

In the development of our AI travel app, we used a couple of methodologies.

1.2.1 Data Collection

One of the primary need of our project is to have a real time data. We have collected real-time data by using APIs of Airbnb and booking.com.

1.2.2 App Development

we will develope the mobile app interface with flutter along with backend services to power the chatbot, recommendation system and payment integrations through third parties. We will use firebase to store users related data and We will use pinecone a vector database to store the real time data that our agent will collect at run time.

1.2.3 Testing

User testing will be conducted to evaluate the chatbot conversations, app usability and end-to-end planning/booking. Feedback will be used to refine the system.

Chapter 2

Review of Literature

2.1 Introduction

This literature review provides an overview of recent research leveraging AI for developing conversational agents focused on travel planning. The goal is to synthesize key findings and limitations from current studies to identify opportunities and challenges in applying state-of-the-art AI to create an effective travel planning chatbot.

By reviewing existing work on AI travel assistants and conversational platforms, this review aims to inform the selection and application of appropriate NLP and ML methods. The goal is to facilitate personalized travel planning through conversations with an AI chatbot integrated into a mobile application.

2.2 Research papers

Sr. no	Year	Basic Idea	Methodologies	Results	Limitations
[1]	2017	Impacts of AI on the Travel and Tourism Industry.	AI impacts the travel industry by enhancing customer experiences through personalized recommendations and improving efficiency in areas like booking processes and customer service with the help of data analytics.	The adoption of AI driven tools like chatbots is crucial for travel industry for marking a positive future in the ongoing technological boom	AI-related solutions are still not widespread within the tourism sector.
[2]	2019	Technical Review of AI applications in travel industry.	AI has caused dramatic changes in the travel industry. Different AI bot like Customer Service bots, Facebook chatbots, Travel bots and robots provides vast new opportunities.	The use of computers and smart technology has made big changes in how tourism works. Robots and smart computer systems, like artificial intelligence, are helping tourism companies a lot.	Applications of AI can be introduced in the travel industry, but this subject has only been scratched on the surface.

Sr. no	Year	Basic Idea	Methodologies	Results	Limitations
[3]	2023	Researched on PDF chatbot system using Langchain, (LLM) and Pinecone for vector storage.	PDF extraction, semantic matching via cosine similarity, user query processing using a language model, text response generation, and user interaction through a Streamlit web app.	Successful creation of chatbot achieving high accuracy rate.	Improving the accuracy and fluency of the chatbot, and expanding the range of tasks.
[4]	2023	Benchmarking Auto-GPT styled agents for real-world decision-making tasks and introducing"Additional opinions algorithm".	evaluating Auto-GPT styled agents in web shopping and ALFWorld tasks, using task-specific prompts, and considering external expert opinions.	Auto-GPT approach, integrating large language models with external expert opinions, surpassed imitation learning-based models in intricate online decision-making tasks, with GPT-4 delivering top performance.	Bias in external expert models, and the limited scope of the evaluated tasks.

Sr. no	Year	Basic Idea	Methodologies	Results	Limitations
[5]	2024	Use of LLms in the field of tourism, automating customer services.	AI technologies, Educational impact	Investigating AI impacts on education, including benefits and potential risks.	Need for careful consideration of AI integration in education.
[6]	2018	Application of Firebase in Android app development	Google Firebase and Android development "kotlin"	A study on the application of Firebase in Android app development highlighting the practical uses and benefits.	The research is limited to the primary and fundamental Firebase functions and does not explore complex integrations.
[7]	2023	Implementation of generative artificial intelligence (such as ChatGPT) and similar tools in the hospitality and tourism industry.	Reviewing literature, setting a research agenda, and examining ChatGPT's implementation benefits and challenges.	Integrating ChatGPT and similar generative AI technologies can transform the hospitality and tourism industry, enhancing customer experiences and productivity.	Reliance on non-empirical data, potential lack of universal applicability, limited coverage of ethical complexities, trust and adoption challenges.

2.2.1 Summaries

The paper [1] discusses that Artificial intelligence is affecting the tourism industry in several ways. New technologies like chatbots and apps are used by travel companies. AI can personalize travel offers based on individual preferences. This allows for more efficient travel planning tailored to each customer. However, companies must adapt to keep up with changing consumer demands in this evolving industry.

The paper [2] discusses the rise of chatbots and artificial intelligence in the tourism industry. It covers different types of chatbots like customer service bots and Facebook chatbots. It also looks at how AI is being used in travel assistants and recommendations. Robots are also playing a bigger role in hotels, with some being staffed only by robots. In conclusion, AI is revolutionizing the tourism industry by providing personalized service and improving operations.

This paper [3] presents a PDF chatbot created using LangChain and an LLM model to answer questions about PDF documents. The chatbot is trained on a dataset of PDFs and uses Pinecone to store the document embeddings. React JS is used to build a frontend interface. The results show the chatbot can accurately answer questions about PDFs it has been trained on.

The paper [4] proposes using additional opinions from imitation learning models to help large language models in task completion. Experiments show this approach improved performance for models like GPT-3 and GPT-4 in web shopping and game tasks. The best results came from GPT-4, which was able to integrate suggestions while still outperforming imitation learning alone. Overall, the study demonstrates the effectiveness of this novel technique for adapting large language models to decision making.

This paper [5] focuses on the use of LLMs in tourism, specifically their role in automating customer service and support through chatbots and virtual assistants.

This Paper [6] explores the basics of Firebase and its use in Android app development, showcasing its benefits and functionalities

The paper [7] discusses the potential impact of ChatGPT on the hospitality and tourism industry. It explores opportunities for customer service improvements and job changes. Concerns around privacy, bias, and trust are also covered. OpenAI product offerings and

pricing strategies are areas for further research. Overall, ChatGPT may help improve customer experiences but also raise challenges to address.

2.3 Competitors

Our Final Year Project is product-based, built on new frameworks like crewAI, which is why fewer research papers are available but have some competitors in the market. Below is the table, showing our competitors and how we are different from them by filling the gaps in the market.

Comapny Name	**Founding Year**	**Company's Worth**	**Business Description**	**Limitations**
Airbnb	2007	$78 Billion	Online marketplace for short- and long-term homestays and experiences. The company acts as a broker and charges a commission from each booking.	• Millions of listing. • Scammable. • Less filter options.
Expedia	1996	$13 Billion	Online marketplace for hotels, air tickets, car rentals, etc	• Huge bunch of Listings. • No detailed filtering.
Tripadvisor	2000	$2.1 Billion	Online blogging+marketplace that advises and operates with online travel agencies and do comparisons	• Personal blogs. • Advices. • Aggregator & Reviews.

Company Name	**Founding Year**	**Company's Worth**	**Business Description**	**Limitations**
Booking.com	2005	$98 Billion	Largest online travel agency, marketplace for hotels	• Millions of hotels listed. • Less filter options.
Hostelworld	1999	$500 Million	Largest marketplace for hostels all around the world	• Thousand of hostels. • Less filter options. • Scammable.
Skyscanner	2001	$2 Billion - $5 Billion	The company lets people research and book travel options for their trips, including flights, hotels and car hire.	• Aggregate tickets & options.
NomadList	2021	$5 Million	Infoplace for digital nomads	• Information based. • Advices. • Community.

2.4 Conclusion

In conclusion, AI is powering travel planning chatbots by understanding language better. However, developing an effective chatbot like "Jaaoo" poses challenges in collecting destination data at scale. Continued research in AI used for travelling can help address this. Advancing techniques in NLP, ML and platforms like Flutter/Firebase can create engaging AI to simplify trip planning globally.

Chapter 3

Project Vision

This chapter should have the following information in it.

3.1 Problem Statement

A lot of companies in the travel industry haven't fully taken advantage of what AI can do. There's a gap between what technology can offer and what's currently available for travelers. We aim to bridge this gap with our AI-powered travel app

3.2 Business Opportunity

In the travel world, there's a chance to make things way more easy and personalized for people. By using AI, we can create an app that designs travel plans tailored to individual preferences. This isn't just about tech; it's a business opportunity to provide better, smarter travel solutions.

3.3 Objectives

Our goal is to develop a user-friendly mobile app that uses AI to create travel plans. Whether you want a custom plan through chatting with a bot or just picking a ready-

made plan, we've got you covered. We want to make travel planning easy, enjoyable, and efficient.

3.4 Project Scope

Our focus is on building the AI travel app using Flutter, incorporating databases like Firebase and Pinecone, and tapping into APIs from popular platforms like Airbnb and Booking.com. We're keeping it practical, ensuring the app is useful and accessible to everyday users.

3.5 Target Audience

Our AI travel app targets travelers aged 20-50, including individuals, couples and families who like planning their own vacations. A lot of people in this age group are comfortable using apps and websites to figure out trips instead of travel agencies. personalized recommendations from chatbot would help busy professionals and families with kids who don't have hours to spend planning.

3.6 Constraints

There are some limitations of our AI travel app. We are covering top ten most visited countries to ensure high accuracy provided by our AI models.

3.7 Stakeholders Description

3.7.1 Stakeholders Summary

Our list of Stakeholders Include:

- Project supervisor : The head who'll be supervising the project.
- Project Team : The team responsible for designing, developing, and maintaining the application.
- End Users: Individuals Want to visit the world withouth worrying about the process and steps to be needed.
- Peer reviewers: reviewrs who'll review the project and provide feedback for correction purpose

3.7.2 Key High Level Goals and Problems of Stakeholders

Below are mentioned some of the High level Goals and problems related to the Stakeholders:

3.7.2.1 High Level Goals

- Project Success
- User Satisfaction
- Innovation and Growth
- Compliance and Risk Management

3.7.2.2 High Problems

- Budget Constraints

- Technical Challenges
- Communication Gaps
- Resource Constraints
- Schedule Delays

Chapter 4

Software Requirements Specifications

4.1 List of Features

4.1.1 Pre-created plans

Other than chatbot, our AI travel app have some pre-created plans as well. The user will have option to go with a pre-created plan as those pre-created plans would be the most common among people to choose or go with a customized plan.

4.1.2 Manual Listing

While we have chatbot to interact with through text prompts, we also provide a user friendly interface where a user just by selecting the options of country, budget, travel dates, accommodation and destination, can also get the travel plan rather than going for a text based chat with a chatbot.

4.1.3 bookings

Our AI travel app is one-stop platform. With the providing the itinerary plan, We also provide the option of booking their flights, hotels and rental cars through our platform. Our platform is connected with third parties through APIs which helps in bookings.

4.1.4 Personalized Notifications

The system provides the personalized notifications. It Utilizes AI to send personalized notifications and alerts to users, such as flight updates, weather forecasts, and recommendations for nearby attractions.

4.2 Functional Requirements

4.2.1 Authentication

Our AI travel app verifies the identity of a user before allowing access to the system, including entering usernames and passwords, biometric verification, or 2FA, ensuring the safety of data and access of data to authorized person.

4.2.2 Authorization levels

The system defines and controls the access levels of different users within a system. An admin has complete system access, while a regular user has limited access to certain features. ensuring a proper check and balance of our system.

4.2.3 Data processing

The system handles task very efficiently. It includes entry of the data. It assigns the data storage, also ensures the data validation and retrieval of the data as well.

4.2.4 User interface and user experience (UI/UX)

The UI/UX are essential elements of the system. A good User interface attracts users and they find using the app more pleasing. The goal is to create an experience that's not just easy but downright enjoyable and is also super user-friendly, meaning it feels natural and comfortable to use.

4.2.5 Natural Language Processing (NLP)

We are Using NLP capabilities for purpose of understanding and processing user queries in natural language. It enhances the conversational aspects our AI travel app. It makes the communication between the user and the app more smooth, making the app easier to use.

4.2.6 AI- based Behavioral Analysis

Our AI travel app uses AI-based behavioral analysis for User Insights, It analyzes the behaviour of user through the interaction and suggests the most suitable plan according to the behaviour of the user.

4.2.7 System integration

Our AI travel app will interact with other systems and connect to outside services, making everything run smoothly. This means our app can seamlessly interact with different technologies and use services from other companies to provide a good experience

4.2.8 Error handling and logging

The system specifies and it handles the errors, It also includes keeping a record of what goes wrong (logging), creating messages that explain errors, troubleshooting steps and maintaining logs for system activities.

4.2.9 Backup and recovery

The System ensures a proper backup of the data and also ensures the system recovery processes, ensuring the integrity of data and availability of system in case of failure.

4.3 Quality Attributes

4.3.1 Usability

it will be very easy for a user to learn and operate the system. simple to understand the interface, buttons and headings. Very less or no attempts needed for user to accomplish a particular task.

4.3.2 Security

Security requirements ensures that there is protection from unauthorized access to the system and its stored data. It has different levels of authorization and authentication across different user roles. Only the system's administrator can change access permissions for the particular system information and system's administrator can view everything. other normal users will have access to certain features.

4.3.3 Performance

Performance is an attribute that shows the system's responsiveness to various user interactions. Poor performance leads to a negative user experience. loading time of any page will not be more than 5 seconds for users that access the website using an 4G mobile connection.

4.3.4 Availability

New module deployment won't impact the front page, product pages, and checkout page availability and won't take longer than one hour. The rest of the pages that may experience problems will display a notification with a timer showing when the system is going to be up again.

4.3.5 Scalability

The system ensures app attendance limit will be scalable enough to support huge traffic at a time. It will smoothly up-to 200,000 users at a time, ensuring that everyone can seamlessly access the app without any hiccups.

4.4 Non-Functional Requirements

4.4.1 Compatibility

The system will ensures that it should be compatible with the various mobile platforms.

4.4.2 Efficiency

Efficiency will be achieved using NLP-based question answering.

4.4.3 Data Storage

The system will ensure the storage of the transaction data. It will ensure that your transaction history remains accessible and secure for an extended duration.

4.4.4 Regulatory

Personally identifiable user data is stored and processed according to local and international regulations.

4.4.5 Fault Tolerance

The system continues operating and prevents data loss during single points of failure through redundancy.

4.5 UML Diagrams

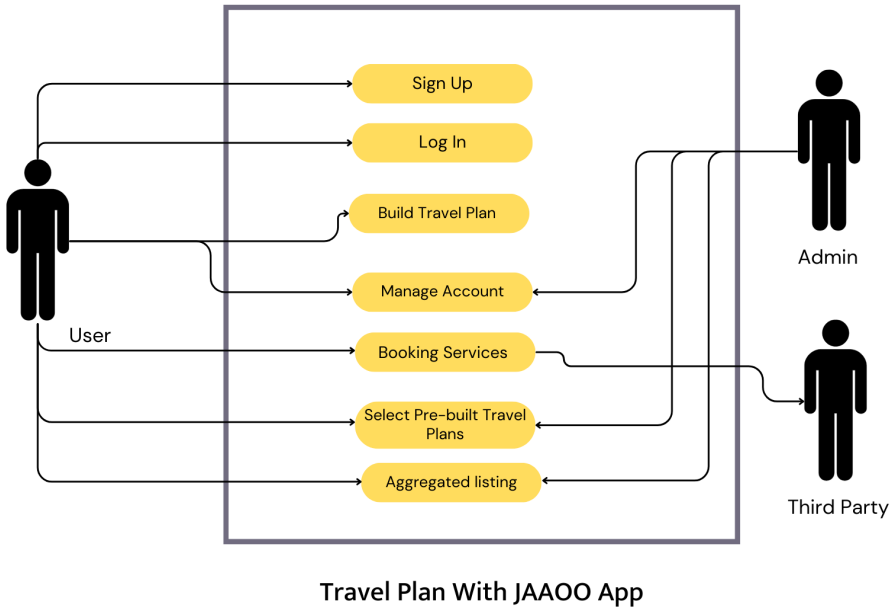


Figure 4.1: Use Case Diagram

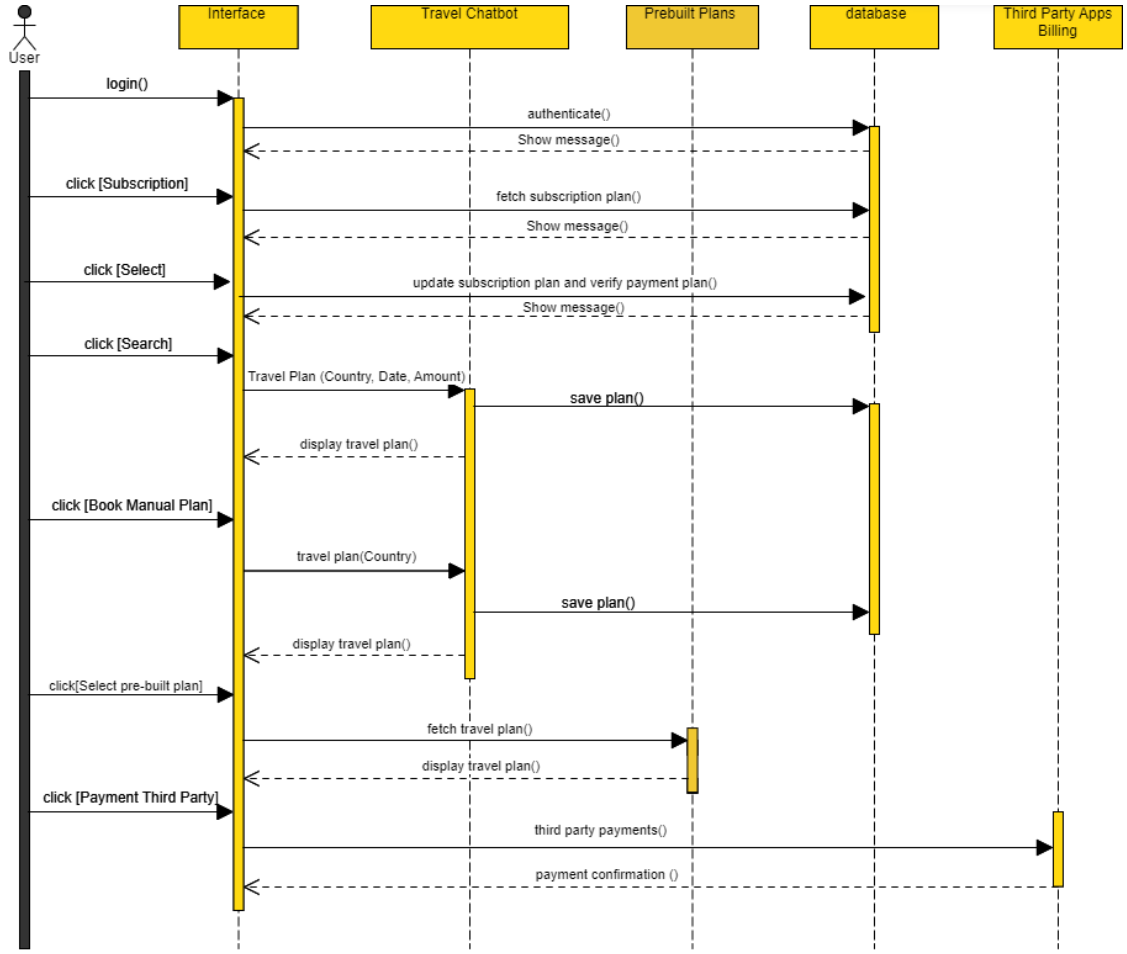


Figure 4.2: System Sequence Diagram

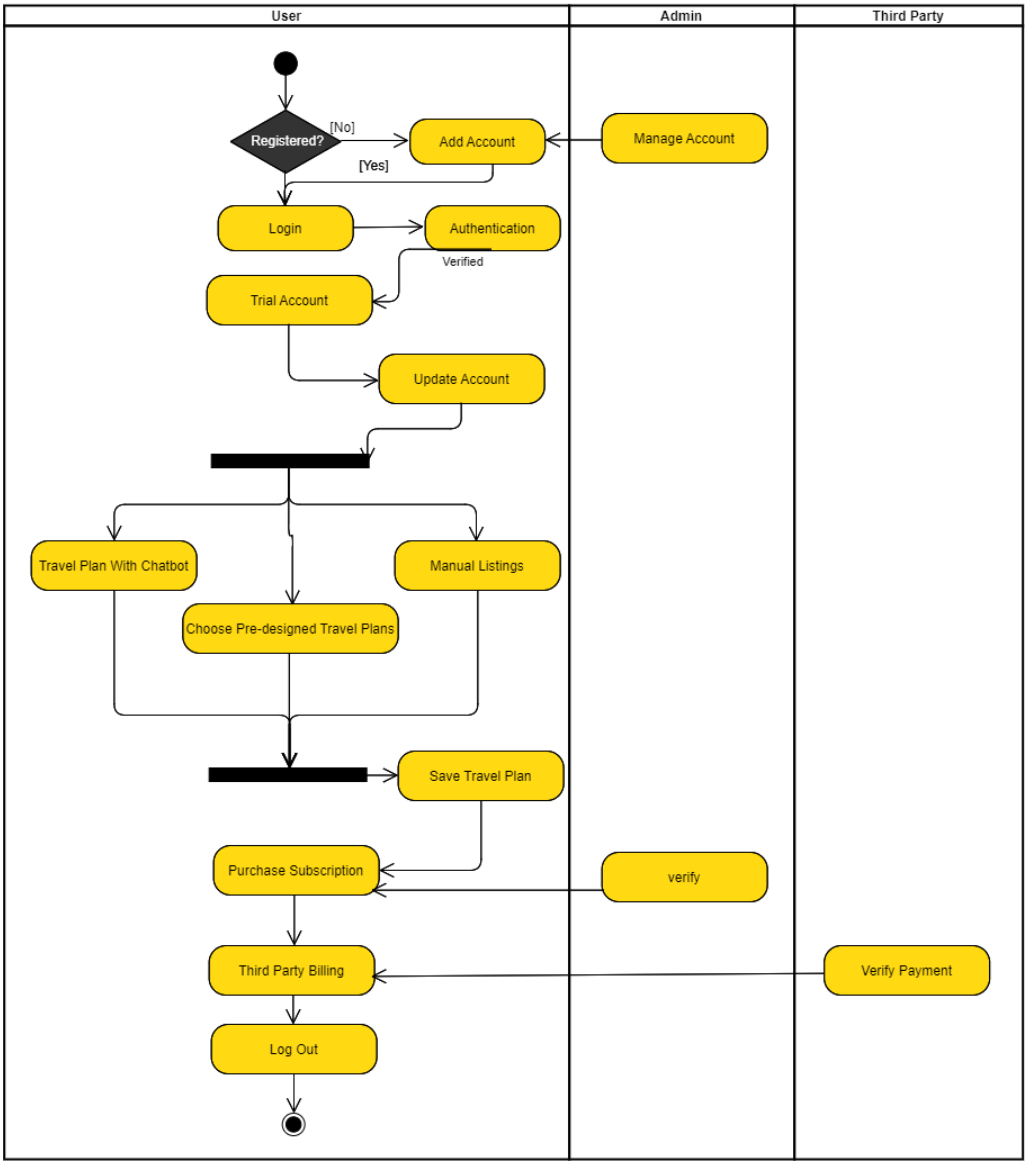


Figure 4.3: Swimlane Diagram

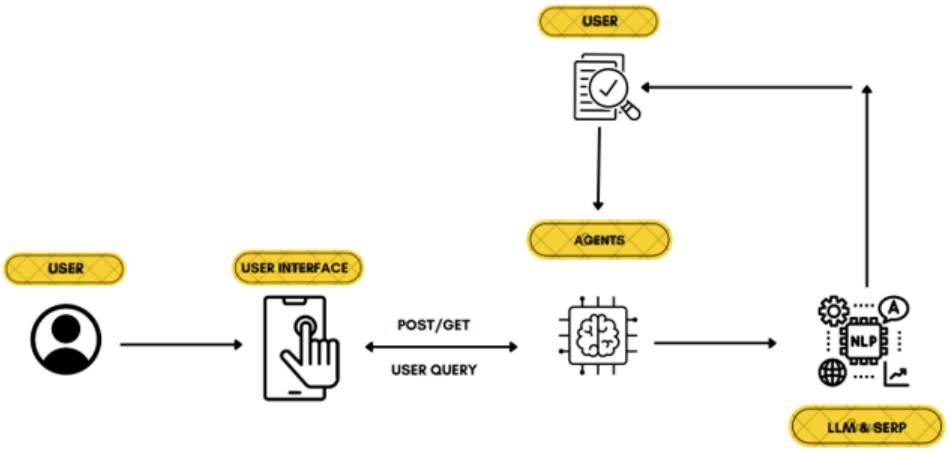


Figure 4.4: Backend Architecture Diagram

Chapter 5

Iteration 1

5.1 Midterm FYP 1

5.1.1 Tools

We have used the tools below to create the User Interface of the APP.

- Figma: Collaborative interface designing tool.
- Flutter: We implemented UX based on designed UI.

5.1.2 Screens

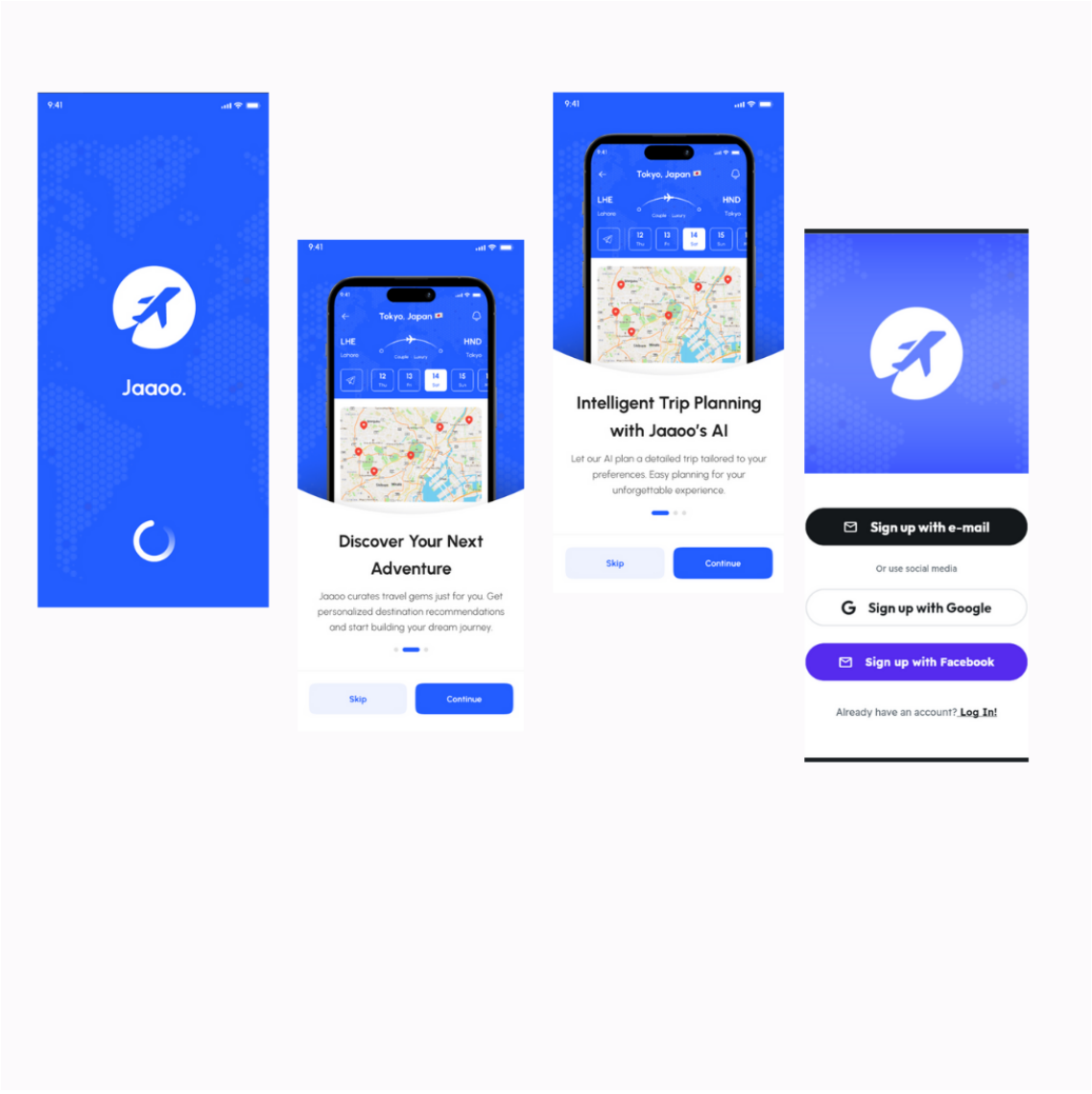


Figure 5.1: Onboarding Screens

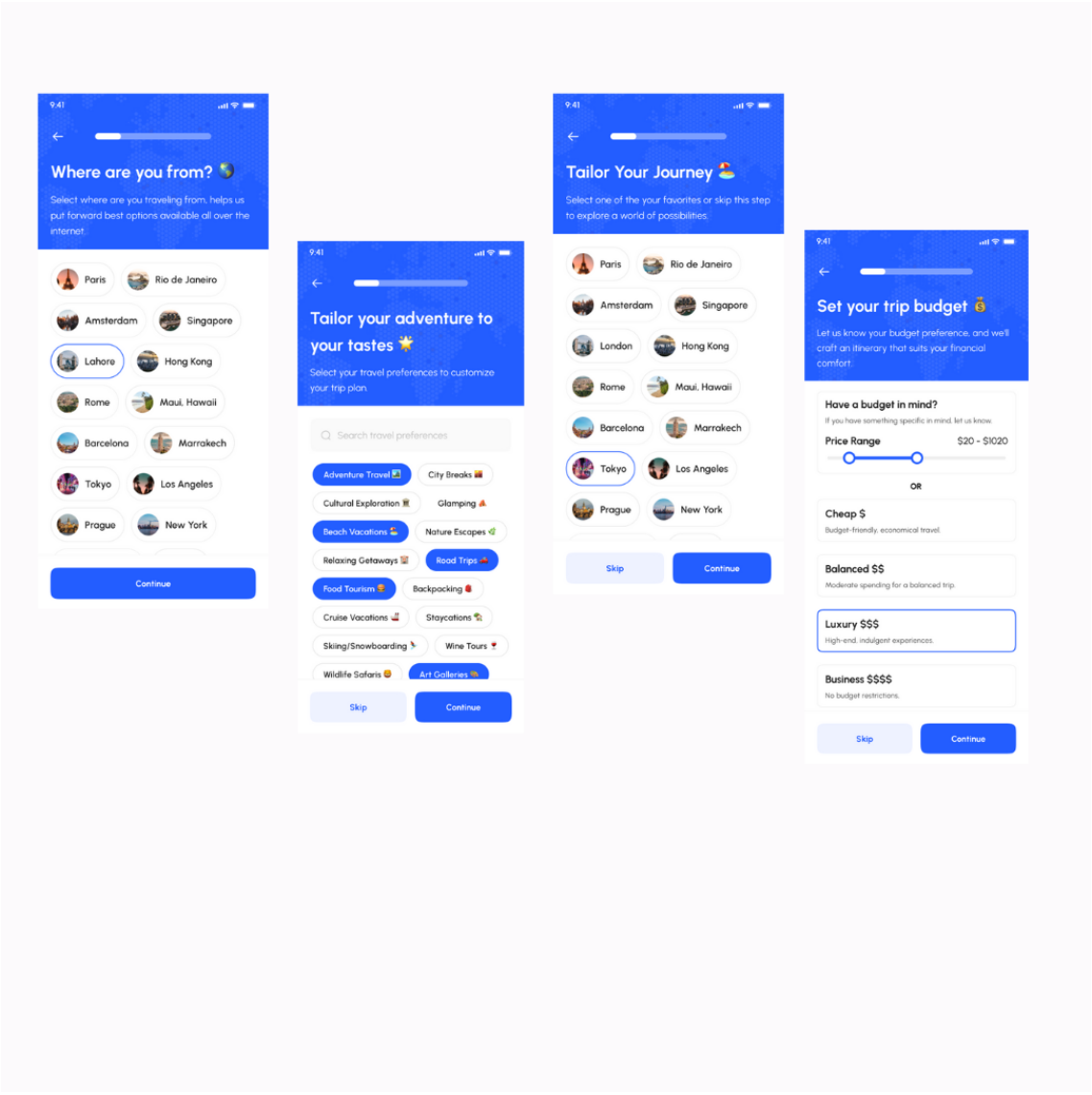


Figure 5.2: Journey Tailoring Screens

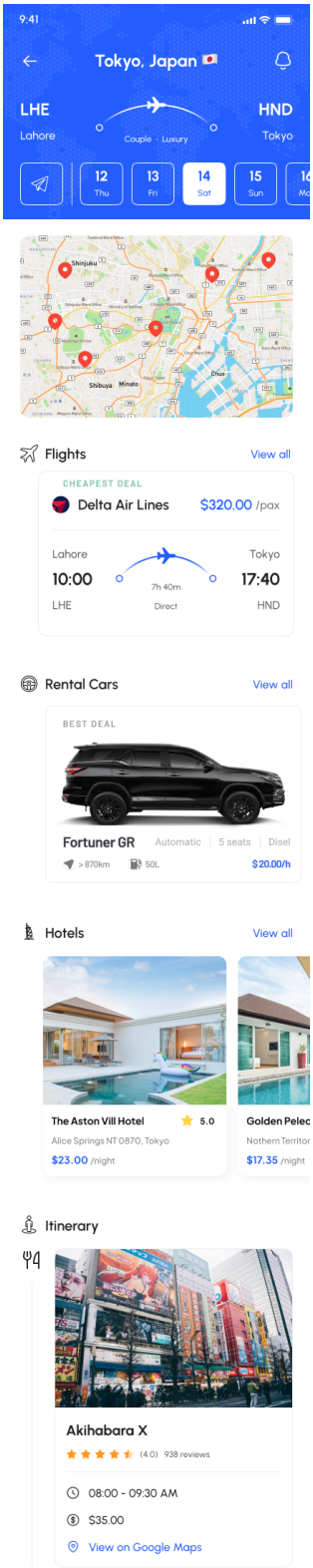


Figure 5.3: Itinerary & Booking Screen(1)

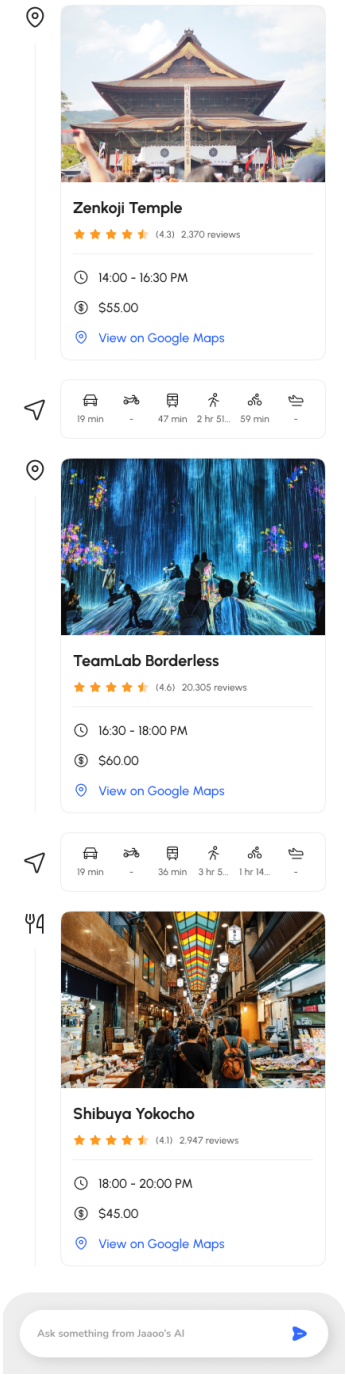


Figure 5.4: Itinerary & Booking Screen(2)

Chapter 6

Iteration 2

6.1 FYP 1 Final

6.1.1 Database Selection

We are using "Firebase" as the main database to store the users and their respective information along with the plan created by each user.

6.1.2 Tools

We are using the tools below,

- Crew AI: An AI framework to build custom agents.
- Flutter: Using Flutter to build a full-fledged application.
- Firebase: To store data and connect with the app to show the stored plans.
- Flask: Tool to connect python outputs with databases.

6.1.3 Coding Structure

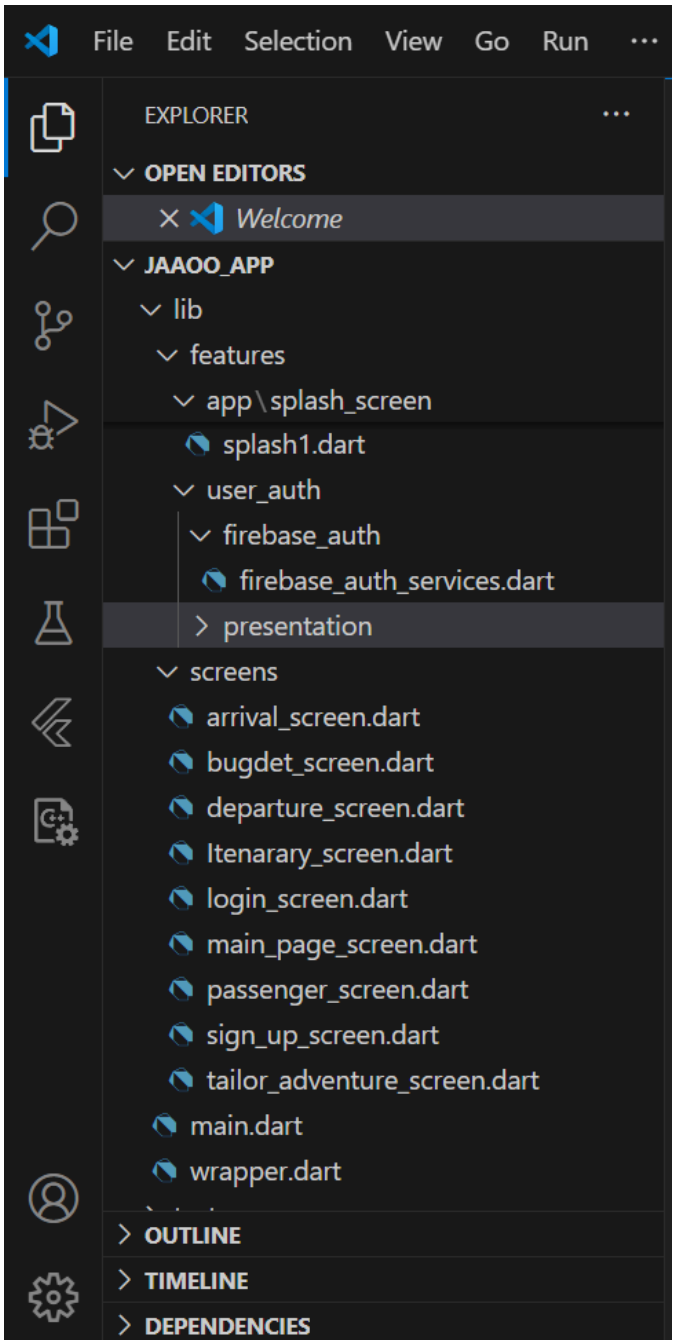


Figure 6.1: Flutter Structure

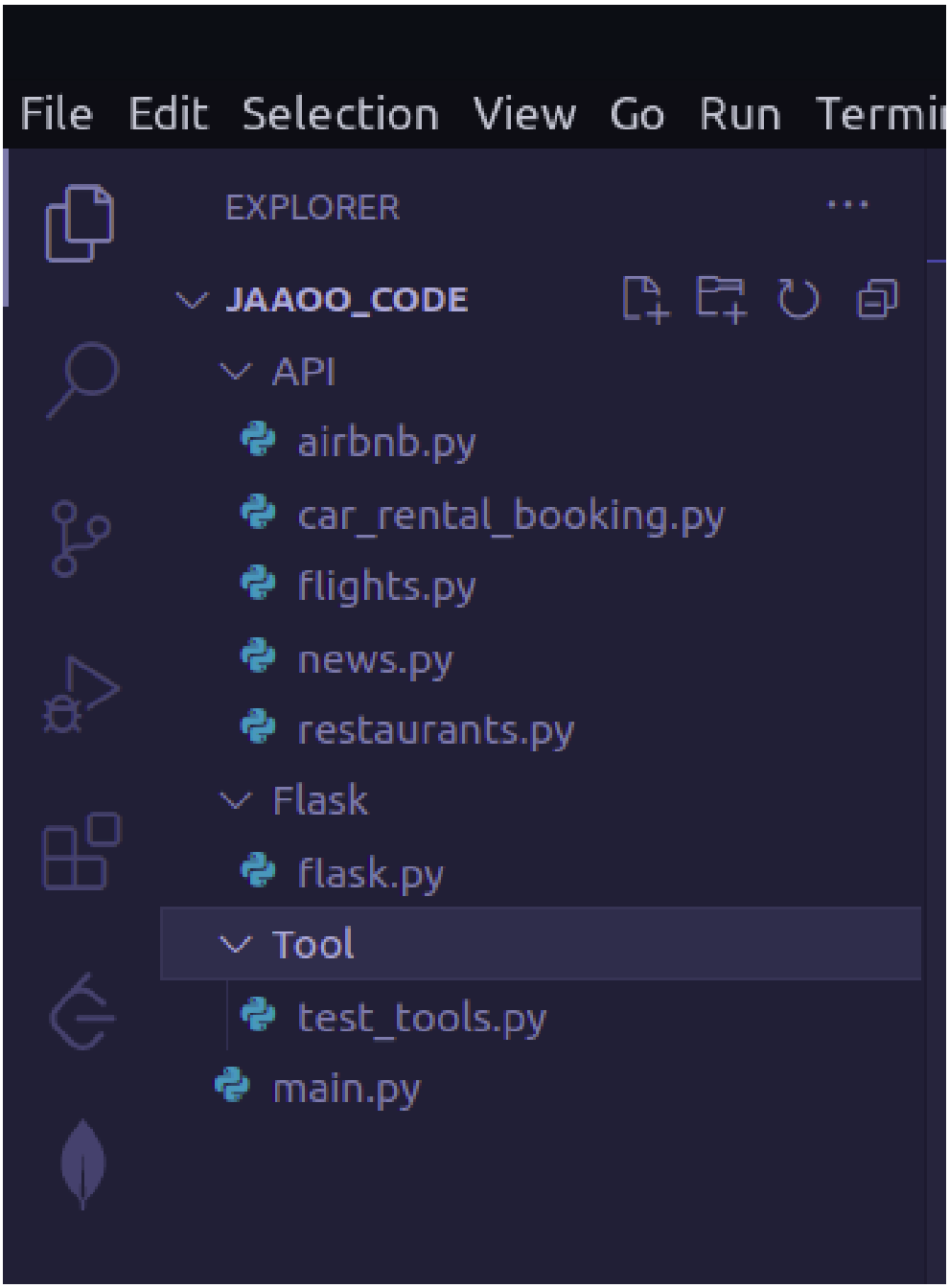


Figure 6.2: Core Structure

Chapter 7

Iteration 3

7.1 Midterm FYP 2

7.1.1 Backend System

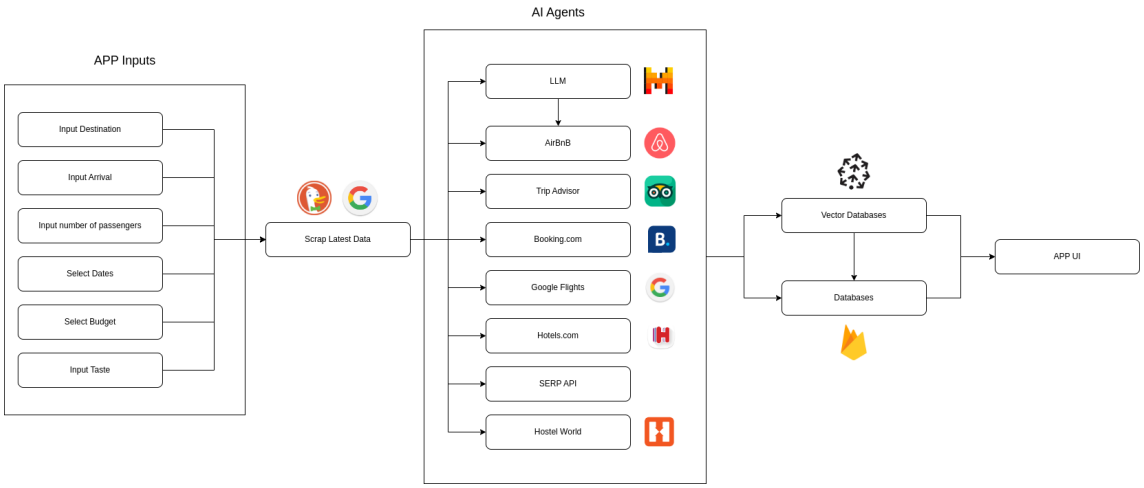


Figure 7.1: App Structure

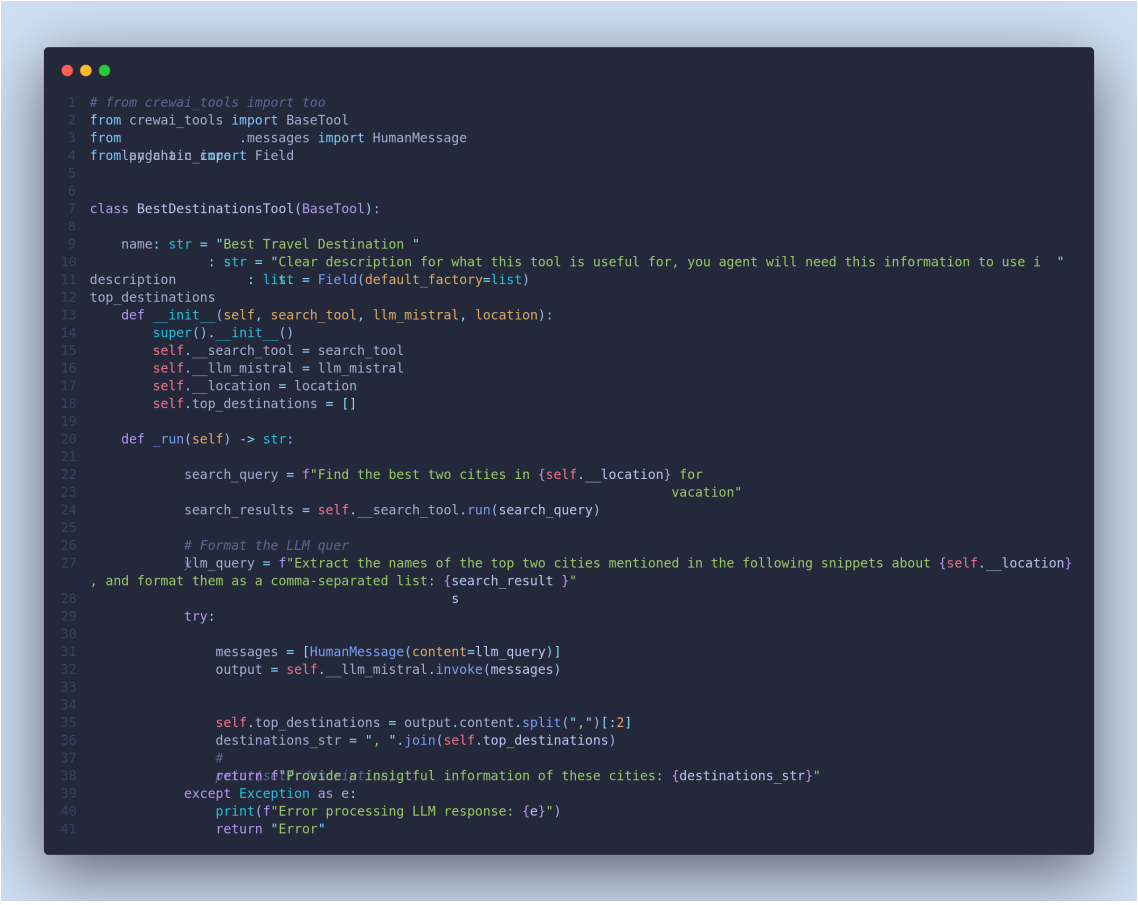
7.1.2 Code

```
    # Create
    Agencoach = Agent(
            role='Travel      ',
            goal=fAgdnbest_destination (loc),
            backstory="
You're an experienced travel agent with a keen eye for finding the perfect destination for any travele  ",
r.          verbose=True,
            allow_delegation=False,
            # tools=[search_too
            llm=llm_mistral_coach
        )

        influencer = Agent(
            role='Travel Planen ',
            goal="Craft detailed travel plans based on selected destinantions with emoji filled bullet  ",
            backstory="
You're a renowned travel planner known for your ability to turn any destination into an unforgettable adventure. With a deep
 understanding of culture, cuisine, and hidden gems, you create personalized itineraries that cater to the interests and desi
res of every traveler. Your plans are not just about places to visit; they're about experiences that last a lifetime.
",
            verbose=True,
            allow_delegation=True,
            llm=llm_mistral
        )

        critic = Agent(
            role="Travel Experience Enhance ",
            goal="Provide two hundred  words insightful feedback to refine and elevate the planned travel experience ",
            backstory="
With years of globetrotting under your belt and a keen eye for detail, you've become an expert in identifying what makes a go
od trip great. Your experiences have taught you the importance of cultural immersion, sustainable travel, and creating moment
s that resonate on a personal level. Now, you lend your expertise by reviewing travel plans, suggesting enhancements that pro
mise to deepen the traveler's connection with each destination and ensure a more memorable journey.
",
            verbose=True,
            allow_delegation=True,
            llm=llm_mistral
        )

        hotel = Agent(
            role = "Hotel Marketing Agen ",
            goal = "Provide insightful and hotel information to elevate the user experience " + main(),
            backstory = "
As a specialized AI agent programmed to fetch hotel information from the TripAdvisor API, you're equipped with advanced algor
ithms to extract and interpret data about various hotels.
",
            verbose=True,
            allow_delegation=True,
            # tools=
            llm=llm_mistral
        )

```

Figure 7.2: Custom Agents

```python
import os
import requests


def GetAirbnb(location, checkin, checkout, adults):

    url = "https://airbnb13.p.rapidapi.com/search-locatio "
          n
    rapid_api_key = os.environ.get("RAPID_API_KEY")

    headers = {
        "X-RapidAPI-Ke ":              ,
        "X-RapidAPI-    "rapidapi.co ",
        "User-agent": "AirBnb"

    }

    querystring = {"location": location, "checkin": checkin, "checkout": checkout, "adults": adults,
                   "children": "0", "infants": "0", "pets": "0", "page": "1", "currency": "USD"}

    response = requests.get(url, headers=headers, params=querystring)

    response_json = response.json()

    # print(response_jso
    #)Filter listings with a rating greater than 4.5 and extract the desired detail
    filtered_listings = []
    # Initialize a counter for the entrie
    entries_count = 0

    # Iterate over the results in the JSON respons
    for listing in                ['results']:
        # Check if response_json
        5   # Add the listing's details to the filtered listings lis
        if 'rating' in listing and listing['rating'] > 4.5:
            filtered_listing .append({
s               'name': listing['name'],
                'bathrooms': listing['bathrooms'],
                'beds': listing['beds'],
                'bedrooms': listing['bedrooms'],
                'persons': listing['persons'],
                'reviewsCount': listing['reviewsCount'],
                'rating': listing['rating'],
                'type': listing['type'],
                'address': listing['address'],
                'previewAmenities': listing['previewAmenities'],
                # Assuming total price is require
                'price': listing['price']['total'],
                'url': listing['url']
            })
            # Increment the counte
            entries_count += 1
            # Check if we have reached 5 entrie
            if entries_count == 5:
                # Break the loop if 5 entries are collecte
                break
        else:
            continue

    print(f"\nTop Five Airbnb Listings in {location}: \n")
    # Display the filtered listing
    for listing in filtered_listings:
        print(f"🏠 Name: {listing['name']}\n🛁 Bathrooms: {listing['bathrooms']}\n🛏 Beds: {listing['beds']}"
            f"\n🚪 Bedrooms: {listing['bedrooms']}\n👥 Persons: {listing['persons']}\n📝 Reviews Count: {listing['
reviewsCount']}"
            f"\n⭐ Rating: {listing['rating']}\n🏡 Type: {listing['type']}\n📍 Address: {listing['address']}"
            f"\n🔑 Preview Amenities: {', '.join(listing['previewAmenities'])}\n$ Price:  {listing['price']}"
            f"\n🔗 URL: {listing['url']}\n" + "-"*80)                          $
```

Figure 7.3: AirBnB Data Scrapping



```
# from crewai tools import too
from crewai_tools import BaseTool
from              .messages import HumanMessage
fromlpydahin_impor Field


class BestDestinationsTool(BaseTool):

    name: str = "Best Travel Destination "
              : str = "Clear description for what this tool is useful for, you agent will need this information to use i  "
description         : list = Field(default_factory=list)
top_destinations
    def __init__(self, search_tool, llm_mistral, location):
        super().__init__()
        self.__search_tool = search_tool
        self.__llm_mistral = llm_mistral
        self.__location = location
        self.top_destinations = []

    def _run(self) -> str:

            search_query = f"Find the best two cities in {self.__location} for
                                                                    vacation"
            search_results = self.__search_tool.run(search_query)

            # Format the LLM quer
            llm_query = f"Extract the names of the top two cities mentioned in the following snippets about {self.__location}
, and format them as a comma-separated list: {search_result }"
                                                s
            try:

                messages = [HumanMessage(content=llm_query)]
                output = self.__llm_mistral.invoke(messages)


                self.top_destinations = output.content.split(",")[:2]
                destinations_str = ", ".join(self.top_destinations)
                #
                return f"Provide insigtful information of these cities: {destinations_str}"
            except Exception as e:
                print(f"Error processing LLM response: {e}")
                return "Error"
```

Figure 7.4: Scrape Best Destination

```python
import os
import requests


def GetCarRentals(location):

    rapid_api_key = os.environ.get("RAPID_API_KEY")

    url = "https://booking-com15.p.rapidapi.com/api/v1/cars/searchDestinatio "
    querystring = {"query": location}

    # Headers including your RapidAPI key and hos
    headers = {
        "X-RapidAPI-Ke ":              ,
        "X-RapidAPI-    "rapidapi_key,
        "User-agent": "Car_Rental"

    }

    # Make the GET request to the AP
    response = requests.get(url, headers=headers, params=querystring)

    # Parse the JSON
    data = response.json()

    #
    # Check if the API call was successfu
    if data['status'] and data['message'] == "Success":
        # Extract names of the car rental location
        locations = [location['name'] for location in data['data']]

        # Prepare the output messag
        output_message = "🚗 Car rentals are available at these locations:   \n" + \
            "\n".join(f"- {location}" for location in locations)

        print(output_messag )
    else:     e
        print("Failed to retrieve car rental location  ")
              s.
```

Figure 7.5: Car Rental

```python
import requests
import serpapi
import os
from serpapi import GoogleSearch

rapid_api_key = os.environ.get("RAPID_API_KEY")
google_flights_key = os.environ.get("G_FLIGHTS_KEY")



g_latitude, g_longitude = "", ""


def get_airport_code(city, headers = 0):
    """
    Fetch the airport code for a given city using TripAdvisor AP
I.  """
    global g_latitude, g_longitude
    url = "https://tripadvisor16.p.rapidapi.com/api/v1/flights/searchAirpor "
    querystring = {"query": city}
    response = requests.get(url, headers=headers, params=querystring)
    data = response.json()
    #
    if data['status'] and 'data' in data and len(data['data']) > 0:
        # Assuming the first entry in 'data' is the relevant airpor
        airport_data = data['data'][0]
        if 'coords' in airport_data:
            coords = airport_data['coords']
                    , g_longitude = map(str, coords.split(","))
g_latitude  'airportCode' in airport_data:
            return airport_data['airportCode']
    return ""


# def
#etLocation(
#:        return latitude, longitud
#     else:
#         return "51.51924", "-0.09665
4"

def search_flight (departure_cod , arrival_code, adate, ddate):
    """           e
    Search for flights from departure_code to arrival_code on a given dat
e.  """


    params = {
    "engine": "google_flight ",
    "departure_id":             ,
    "arrival_id": adeparturecode
    "outbound_date": adate,
    "return_date": ddate,
    "currency": "USD",
    "hl": "en",
    "api_key":                  ,
    }         google_flights_key

    search = GoogleSearch(params)
    results = search.get_dict()
    return str(results)
    # Execute the search using the too
    l
def GetFlights(departure_cit , arrival_city, adate, ddate) -> str:
    headers = {y
        "X-RapidAPI-Ke ":             ,
        "X-RapidAPI-    "rapidapi-key
        "User-agent": "GetFlights"

    }

    departure_code = get_airport_code(departure_cit , headers)
    arrival_code = get_airport_code(arrival_city, headers)

    # departure_code = "IS
    #"arrival_code = "LH
    E"
    print(departure_cod , arrival_code)
    if notedeparture_code or not arrival_code:
        print("Failed to get airport code  ")
    try:       s.
        flights = search_flight (departure_cod , arrival_code, adate, ddate)
        return flights          e
    except Exception as e:
        # Handle any exceptions that might occur during function executio
        return f"."
```

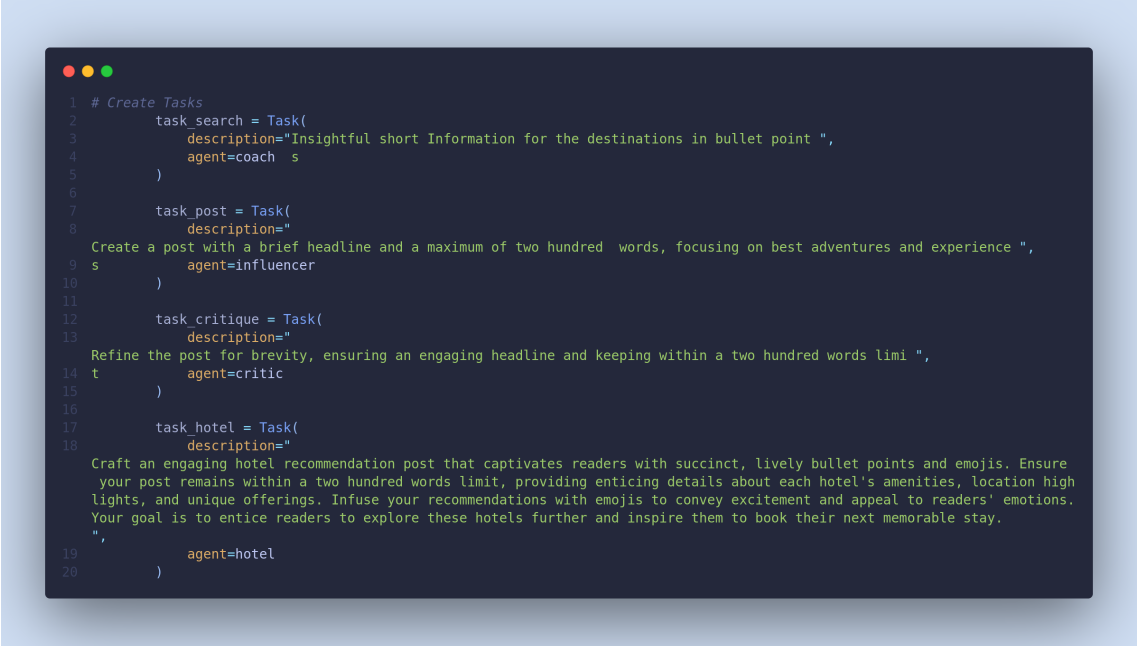
Figure 7.6: Flights

```python
import os
from serpapi import GoogleSearch

def GetNews(location):
    google_flights_key = os.environ.get("G_FLIGHTS_KEY")

    params = {
    "engine": "google_news",
    "q": location,
    "api_key":
    }         google_flights_key

    search = GoogleSearch(params)
    results = search.get_dict()
    # print(results)
    news_results = results["news_results"][8:]

    titles_and_links = [{"title": news["title"], "link": news["link"]} for news in news_results]

    # Displaying the extracted informatio
    for news in                    :
        print(ftitles_and_links[
```

Figure 7.7: News Scrapping

```python
import requests
import serpapi
import os
from serpapi import GoogleSearch


def GetRestaurant (location):
    s
    result_key = os.environ.get("G_FLIGHTS_KEY")

    url = f"https://serpapi.com/locations.json?q {location}&limit=5"
          =
    headers = {
        "User-agent": "Get_Location"
    }

    response = requests.get(url, headers=headers)

    response_json = response.json()

    zoom = "15.1z"
    # Extracting the first set of GPS coordinate
    s        , latitude =            [0]['gps']
longitude               response_json
    # Formatting the string as specifie
    ll = f"@{latitude},{longitude},{zoom}"

    params = {
    "engine": "google_maps",
    "q": "resturants",
    "ll": ll,
    "type": "search",
    "api_key": result_key
    }

    search = GoogleSearch(params)
    results = search.get_dict()

    # Extracting meaningful information for restaurant
    local_results = results['local_results']

    # Prepare the announcement message for restaurant
    announcement = "🍽️ Restaurants available in your area:   \n"
                    🍽️
    for result in             :
        announcement += f"{i}. 🏠 {result['title']} - {result['type']}\n"
                        f"⭐ Rating: {result.get('rating', 'N/A')} (📝 {result.get('reviews', 'N/A')} reviews)\n"
                        f"📍 Address: {result['address']}\n\n")

    # Print the announcemen
    print(announcement)
```

Figure 7.8: Restaurants



```
# Create Tasks
        task_search = Task(
            description="Insightful short Information for the destinations in bullet point ",
            agent=coach  s
        )

        task_post = Task(
            description="
Create a post with a brief headline and a maximum of two hundred  words, focusing on best adventures and experience ",
s           agent=influencer
        )

        task_critique = Task(
            description="
Refine the post for brevity, ensuring an engaging headline and keeping within a two hundred words limi ",
t           agent=critic
        )

        task_hotel = Task(
            description="
Craft an engaging hotel recommendation post that captivates readers with succinct, lively bullet points and emojis. Ensure
 your post remains within a two hundred words limit, providing enticing details about each hotel's amenities, location high
lights, and unique offerings. Infuse your recommendations with emojis to convey excitement and appeal to readers' emotions.
Your goal is to entice readers to explore these hotels further and inspire them to book their next memorable stay.
",
            agent=hotel
        )
```

Figure 7.9: Final Tasks

Chapter 8

Iteration 4

8.1 Final FYP 2

The main objective of our FYP-2 final was to deliver a full fledge working app "Jaaoo".

8.1.1 Backend System

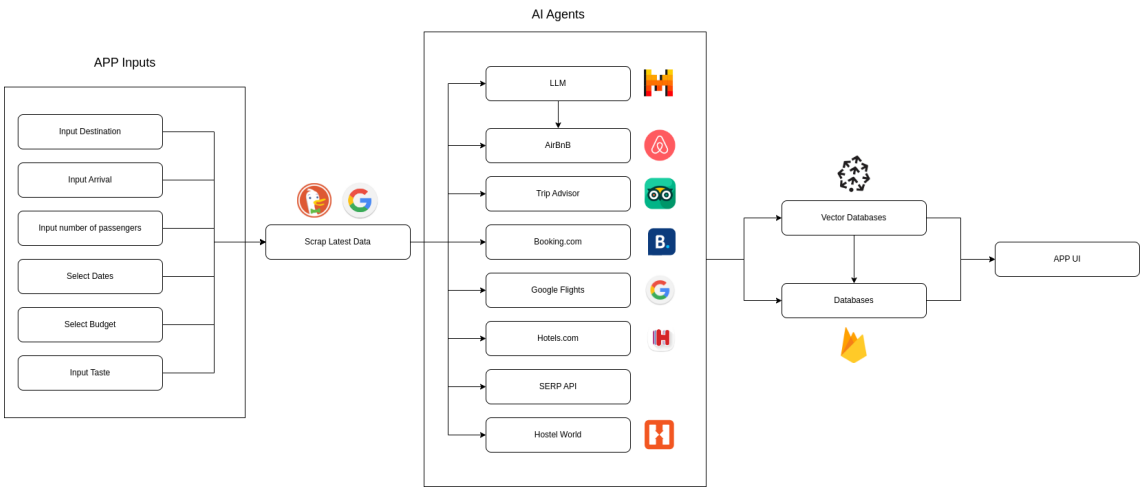


Figure 8.1: App Structure

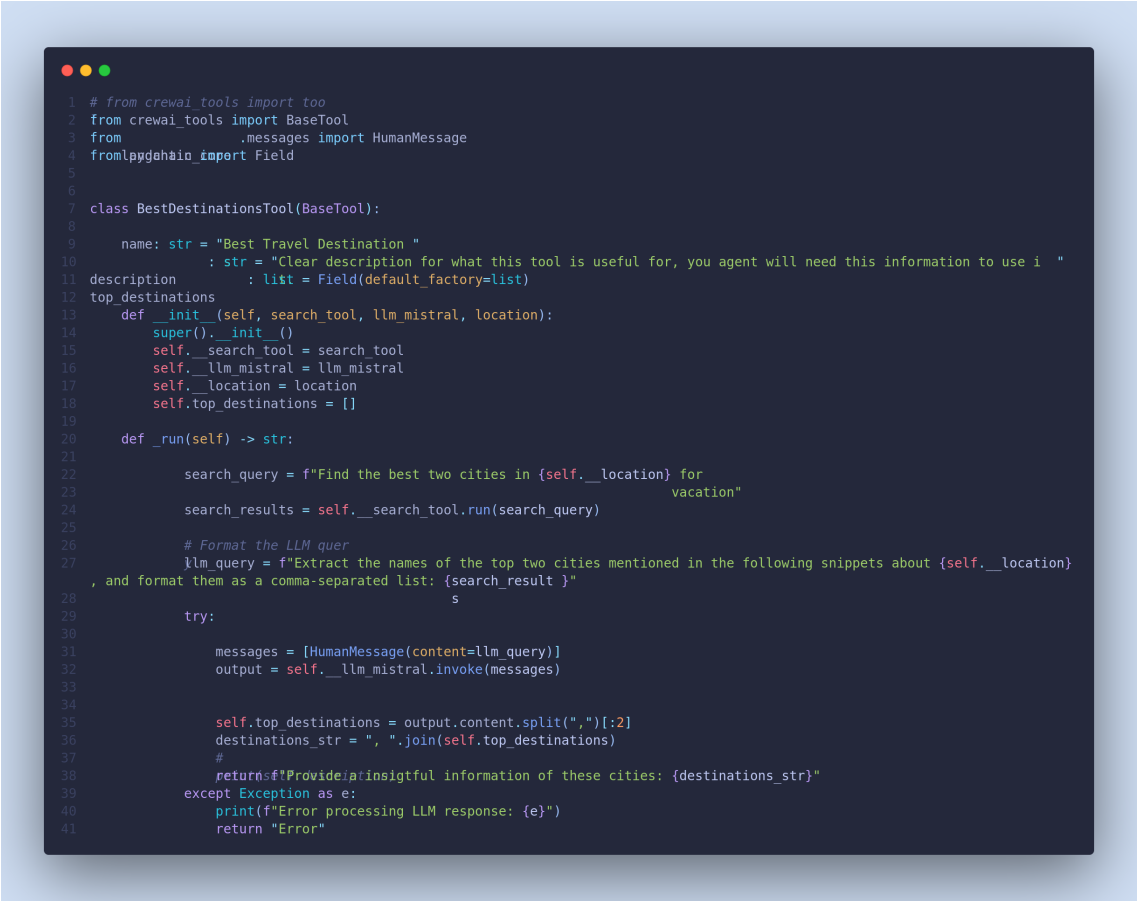
8.1.2 Code

```
    # Create
    Agencoach = Agent(
            role='Travel      ',
            goal=fAgdnbest_destination (loc),
            backstory="
You're an experienced travel agent with a keen eye for finding the perfect destination for any travele  ",
r.          verbose=True,
            allow_delegation=False,
            # tools=[search_too
            llm=llm_mistral_coach
        )

        influencer = Agent(
            role='Travel Planen ',
            goal="Craft detailed travel plans based on selected destinantions with emoji filled bullet  ",
            backstory="
You're a renowned travel planner known for your ability to turn any destination into an unforgettable adventure. With a deep
 understanding of culture, cuisine, and hidden gems, you create personalized itineraries that cater to the interests and desi
res of every traveler. Your plans are not just about places to visit; they're about experiences that last a lifetime.
",
            verbose=True,
            allow_delegation=True,
            llm=llm_mistral
        )

        critic = Agent(
            role="Travel Experience Enhance ",
            goal="Provide two hundred  words insightful feedback to refine and elevate the planned travel experience ",
            backstory="
With years of globetrotting under your belt and a keen eye for detail, you've become an expert in identifying what makes a go
od trip great. Your experiences have taught you the importance of cultural immersion, sustainable travel, and creating moment
s that resonate on a personal level. Now, you lend your expertise by reviewing travel plans, suggesting enhancements that pro
mise to deepen the traveler's connection with each destination and ensure a more memorable journey.
",
            verbose=True,
            allow_delegation=True,
            llm=llm_mistral
        )

        hotel = Agent(
            role = "Hotel Marketing Agen ",
            goal = "Provide insightful and hotel information to elevate the user experience " + main(),
            backstory = "
As a specialized AI agent programmed to fetch hotel information from the TripAdvisor API, you're equipped with advanced algor
ithms to extract and interpret data about various hotels.
",
            verbose=True,
            allow_delegation=True,
            # tools=
            llm=llm_mistral
        )

```

Figure 8.2: Custom Agents

```python
import os
import requests


def GetAirbnb(location, checkin, checkout, adults):

    url = "https://airbnb13.p.rapidapi.com/search-locatio "
          n
    rapid_api_key = os.environ.get("RAPID_API_KEY")

    headers = {
        "X-RapidAPI-Ke ":              ,
        "X-RapidAPI-    "rapidapi.co ",
        "User-agent": "AirBnb"

    }

    querystring = {"location": location, "checkin": checkin, "checkout": checkout, "adults": adults,
                   "children": "0", "infants": "0", "pets": "0", "page": "1", "currency": "USD"}

    response = requests.get(url, headers=headers, params=querystring)

    response_json = response.json()

    # print(response_jso
    #)Filter listings with a rating greater than 4.5 and extract the desired detail
    filtered_listings = []
    # Initialize a counter for the entrie
    entries_count = 0

    # Iterate over the results in the JSON respons
    for listing in                ['results']:
        # Check if response_json
        5   # Add the listing's details to the filtered listings lis
        if 'rating' in listing and listing['rating'] > 4.5:
            filtered_listing .append({
s               'name': listing['name'],
                'bathrooms': listing['bathrooms'],
                'beds': listing['beds'],
                'bedrooms': listing['bedrooms'],
                'persons': listing['persons'],
                'reviewsCount': listing['reviewsCount'],
                'rating': listing['rating'],
                'type': listing['type'],
                'address': listing['address'],
                'previewAmenities': listing['previewAmenities'],
                # Assuming total price is require
                'price': listing['price']['total'],
                'url': listing['url']
            })
            # Increment the counte
            entries_count += 1
            # Check if we have reached 5 entrie
            if entries_count == 5:
                # Break the loop if 5 entries are collecte
                break
        else:
            continue

    print(f"\nTop Five Airbnb Listings in {location}: \n")
    # Display the filtered listing
    for listing in filtered_listings:
        print(f"🏠 Name: {listing['name']}\n🛁 Bathrooms: {listing['bathrooms']}\n🛏 Beds: {listing['beds']}"
            f"\n🚪 Bedrooms: {listing['bedrooms']}\n👥 Persons: {listing['persons']}\n📝 Reviews Count: {listing['
reviewsCount']}"
            f"\n⭐ Rating: {listing['rating']}\n🏡 Type: {listing['type']}\n📍 Address: {listing['address']}"
            f"\n🔑 Preview Amenities: {', '.join(listing['previewAmenities'])}\n$ Price:  {listing['price']}"
            f"\n🔗 URL: {listing['url']}\n" + "-"*80)                          $
```

Figure 8.3: AirBnB Data Scrapping



```
# from crewai tools import too
from crewai_tools import BaseTool
from              .messages import HumanMessage
fromlpydahin_impor Field


class BestDestinationsTool(BaseTool):

    name: str = "Best Travel Destination "
             : str = "Clear description for what this tool is useful for, you agent will need this information to use i  "
description         : list = Field(default_factory=list)
top_destinations
    def __init__(self, search_tool, llm_mistral, location):
        super().__init__()
        self.__search_tool = search_tool
        self.__llm_mistral = llm_mistral
        self.__location = location
        self.top_destinations = []

    def _run(self) -> str:

            search_query = f"Find the best two cities in {self.__location} for
                                                                    vacation"
            search_results = self.__search_tool.run(search_query)

            # Format the LLM quer
            llm_query = f"Extract the names of the top two cities mentioned in the following snippets about {self.__location}
, and format them as a comma-separated list: {search_result }"
                                                s
            try:

                messages = [HumanMessage(content=llm_query)]
                output = self.__llm_mistral.invoke(messages)


                self.top_destinations = output.content.split(",")[:2]
                destinations_str = ", ".join(self.top_destinations)
                #
                return f"Provide insigtful information of these cities: {destinations_str}"
            except Exception as e:
                print(f"Error processing LLM response: {e}")
                return "Error"
```

Figure 8.4: Scrape Best Destination

```python
import os
import requests


def GetCarRentals(location):

    rapid_api_key = os.environ.get("RAPID_API_KEY")

    url = "https://booking-com15.p.rapidapi.com/api/v1/cars/searchDestinatio "
    querystring = {"query": location}

    # Headers including your RapidAPI key and hos
    headers = {
        "X-RapidAPI-Ke ":              ,
        "X-RapidAPI-    "rapidapi_key
        "User-agent": "Car_Rental"

    }

    # Make the GET request to the AP
    response = requests.get(url, headers=headers, params=querystring)

    # Parse the JSON
    data = response.json()

    #
    # Check if the API call was successfu
    if data['status'] and data['message'] == "Success":
        # Extract names of the car rental location
        locations = [location['name'] for location in data['data']]

        # Prepare the output messag
        output_message = "🚗 Car rentals are available at these locations:   \n" + \
            "\n".join(f"- {location}" for location in locations)

        print(output_messag )
    else:     e
        print("Failed to retrieve car rental location  ")
              s.
```

Figure 8.5: Car Rental

```python
import requests
import serpapi
import os
from serpapi import GoogleSearch

rapid_api_key = os.environ.get("RAPID_API_KEY")
google_flights_key = os.environ.get("G_FLIGHTS_KEY")



g_latitude, g_longitude = "", ""


def get_airport_code(city, headers = 0):
    """
    Fetch the airport code for a given city using TripAdvisor AP
I.  """
    global g_latitude, g_longitude
    url = "https://tripadvisor16.p.rapidapi.com/api/v1/flights/searchAirpor "
    querystring = {"query": city}
    response = requests.get(url, headers=headers, params=querystring)
    data = response.json()
    #
    if data['status'] and 'data' in data and len(data['data']) > 0:
        # Assuming the first entry in 'data' is the relevant airpor
        airport_data = data['data'][0]
        if 'coords' in airport_data:
            coords = airport_data['coords']
                    , g_longitude = map(str, coords.split(","))
g_latitude
            if 'airportCode' in airport_data:
                return airport_data['airportCode']
    return ""


# def
#GetLocation(
#:        return latitude, longitud
#     else:
#         return "51.51924", "-0.09665
4"

def search_flight (departure_cod , arrival_code, adate, ddate):
    """
    Search for flights from departure_code to arrival_code on a given dat
e.  """


    params = {
    "engine": "google_flight ",
    "departure_id":             ,
    "arrival_id": departure_code
    "outbound_date": adate,
    "return_date": ddate,
    "currency": "USD",
    "hl": "en",
    "api_key":                  ,
    }         google_flights_key

    search = GoogleSearch(params)
    results = search.get_dict()
    return str(results)
    # Execute the search using the too
    l
def GetFlights(departure_cit , arrival_city, adate, ddate) -> str:
    headers = {y
        "X-RapidAPI-Ke ":             ,
        "X-RapidAPI-    "rapidapi-key
        "Host-agent": "GetFlights"


    }

    departure_code = get_airport_code(departure_cit , headers)
    arrival_code = get_airport_code(arrival_city, headers)

    # departure_code = "IS
    #"arrival_code = "LH
    E"
    print(departure_cod , arrival_code)
    if notedeparture_code or not arrival_code:
        print("Failed to get airport code  ")
    try:       s.
        flights = search_flight (departure_cod , arrival_code, adate, ddate)
        return flights          e
    except Exception as e:
        # Handle any exceptions that might occur during function executio
        return f"."
```

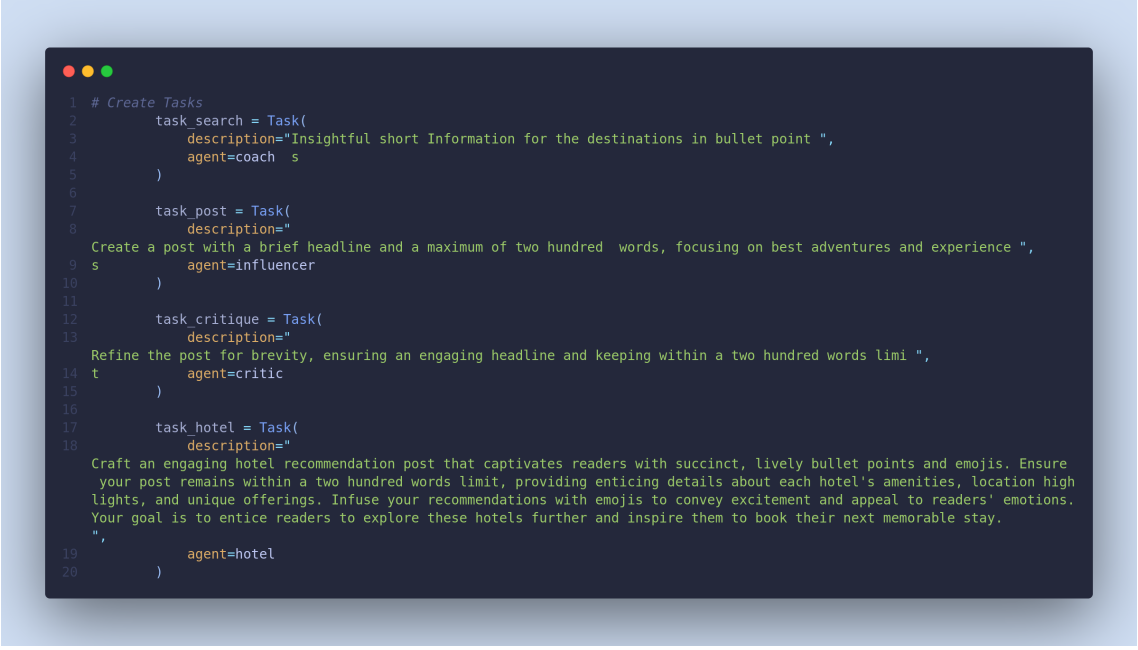
Figure 8.6: Flights

```python
import os
from serpapi import GoogleSearch

def GetNews(location):
    google_flights_key = os.environ.get("G_FLIGHTS_KEY")

    params = {
    "engine": "google_news",
    "q": location,
    "api_key":
    }         google_flights_key

    search = GoogleSearch(params)
    results = search.get_dict()
    # print(results)
    news_results = results["news_results"][8:]

    titles_and_links = [{"title": news["title"], "link": news["link"]} for news in news_results]

    # Displaying the extracted informatio
    for news in                    :
        print(ftitles_and_links['title']}\nLink: {news["link"]}\n')
```

Figure 8.7: News Scrapping

```python
import requests
import serpapi
import os
from serpapi import GoogleSearch


def GetRestaurant (location):
    s
    result_key = os.environ.get("G_FLIGHTS_KEY")

    url = f"https://serpapi.com/locations.json?q {location}&limit=5"
          =
    headers = {
        "User-agent": "Get_Location"
    }

    response = requests.get(url, headers=headers)

    response_json = response.json()

    zoom = "15.1z"
    # Extracting the first set of GPS coordinate
    s        , latitude =            [0]['gps']
longitude               response_json
    # Formatting the string as specifie
    ll = f"@{latitude},{longitude},{zoom}"

    params = {
    "engine": "google_maps",
    "q": "resturants",
    "ll": ll,
    "type": "search",
    "api_key": result_key
    }

    search = GoogleSearch(params)
    results = search.get_dict()

    # Extracting meaningful information for restaurant
    local_results = results['local_results']

    # Prepare the announcement message for restaurant
    announcement = "🍽️ Restaurants available in your area:   \n"
                    🍽️
    for result in             :
        announcement += f"{i}. 🏠 {result['title']} - {result['type']}\n"
                        f"⭐ Rating: {result.get('rating', 'N/A')} (📝 {result.get('reviews', 'N/A')} reviews)\n"
                        f"📍 Address: {result['address']}\n\n")

    # Print the announcemen
    print(announcement)
```

Figure 8.8: Restaurants



```
# Create Tasks
        task_search = Task(
            description="Insightful short Information for the destinations in bullet point ",
            agent=coach  s
        )

        task_post = Task(
            description="
Create a post with a brief headline and a maximum of two hundred  words, focusing on best adventures and experience ",
s           agent=influencer
        )

        task_critique = Task(
            description="
Refine the post for brevity, ensuring an engaging headline and keeping within a two hundred words limi ",
t           agent=critic
        )

        task_hotel = Task(
            description="
Craft an engaging hotel recommendation post that captivates readers with succinct, lively bullet points and emojis. Ensure
 your post remains within a two hundred words limit, providing enticing details about each hotel's amenities, location high
lights, and unique offerings. Infuse your recommendations with emojis to convey excitement and appeal to readers' emotions.
Your goal is to entice readers to explore these hotels further and inspire them to book their next memorable stay.
",
            agent=hotel
        )
```

Figure 8.9: Final Tasks

Chapter 9

Implementation Details

9.1 Programming Languages & Technologies

9.1.1 Fronted System

The Mobile application is developed using the Flutter framework, which use Dart for frontend development.

9.1.2 Backend System

The backend of application is implemented in python, libraries like langchain and flask, auto agents frameworks such as crewai are used.

9.1.3 Database

AI Travel Companion uses Firebase for the database, as it stores all live fecthed data from agents and as well as pre-created travel plan templates are stored over it.

9.1.4 Security

The protection of user data is of utmost importance. We prioritize the implementation of robust security measures to ensure the confidentiality and integrity of user data. Sensitive information, such as passwords, is handle with care and protected from unauthorized access through secure communication channels and data encryption techniques. This approach is in line with industry best practices, and our commitment to maintaining the privacy and trust of our users.

9.1.5 User Experience

we prioritize the creation of an intuitive and user-friendly interface for our users. We achieve this by incorporating a range of UX elements, including responsive design, interactive feedback, and clear instructions. These features are tailored to improve the overall user experience and promote ease of use for all of our users.

Chapter 10

Test Cases

10.1 Backend Testing

Test Case 1: Firebase Authentication

Objective: The user should be signed in and all their sessions restored.

Method: FirebaseAuth instance to authenticate the user with the correct credentials. Upon successful authentication.

Expected Outcome: The user is authenticated successfully and the user session is stored by fetching their data from Firebase.

Test Case 2: Integration and Data Fetch with External APIs

Objective: Ensure the external APIs (e.g., TripAdvisor, Airbnb) retrieves the data.

Method: Fetching data by GET request to External APIs (e.g. Rapidapi's TripAdvisor API)

Expected Outcome: 200 status code with the data in JSON format.

Test Case 3: API Call with No Results

Objective: Verifying the application behavior when API call has no result for specific parameters.

Method: Fetching data by GET request to External APIs and handling it with python conditionals

Expected Outcome: `200` status code with the user-friendly response of `no results` found to the user.

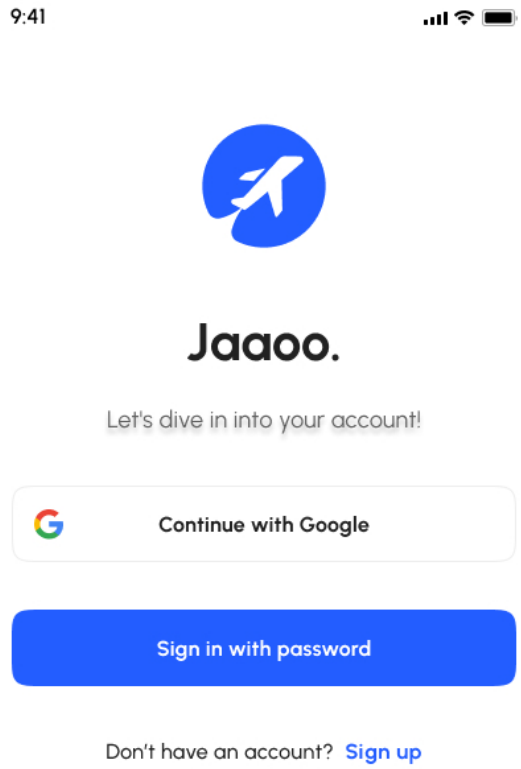
Chapter 11

User Manual

11.1 User Manual Guide

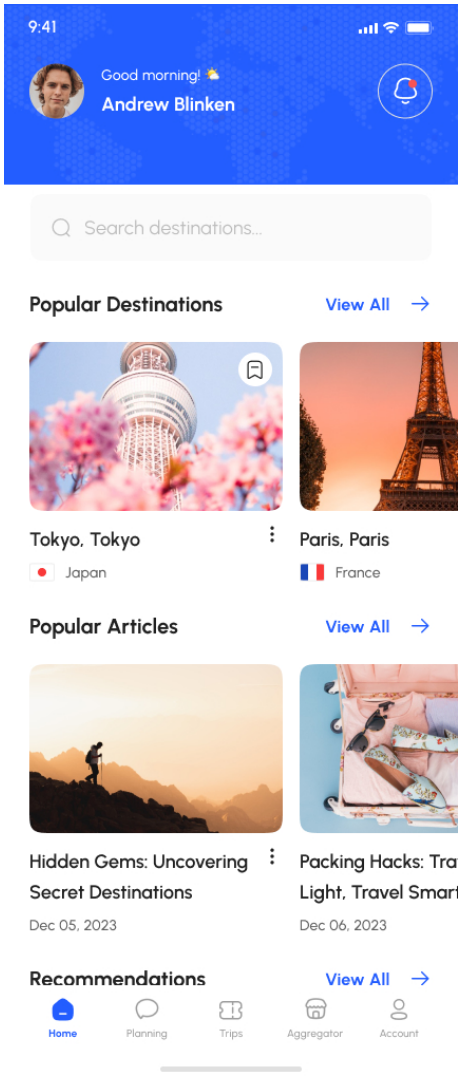
11.1.1 Sign Up or Log In

- New to our app! create your account/sign up.
- Already a user of our, login to your account.



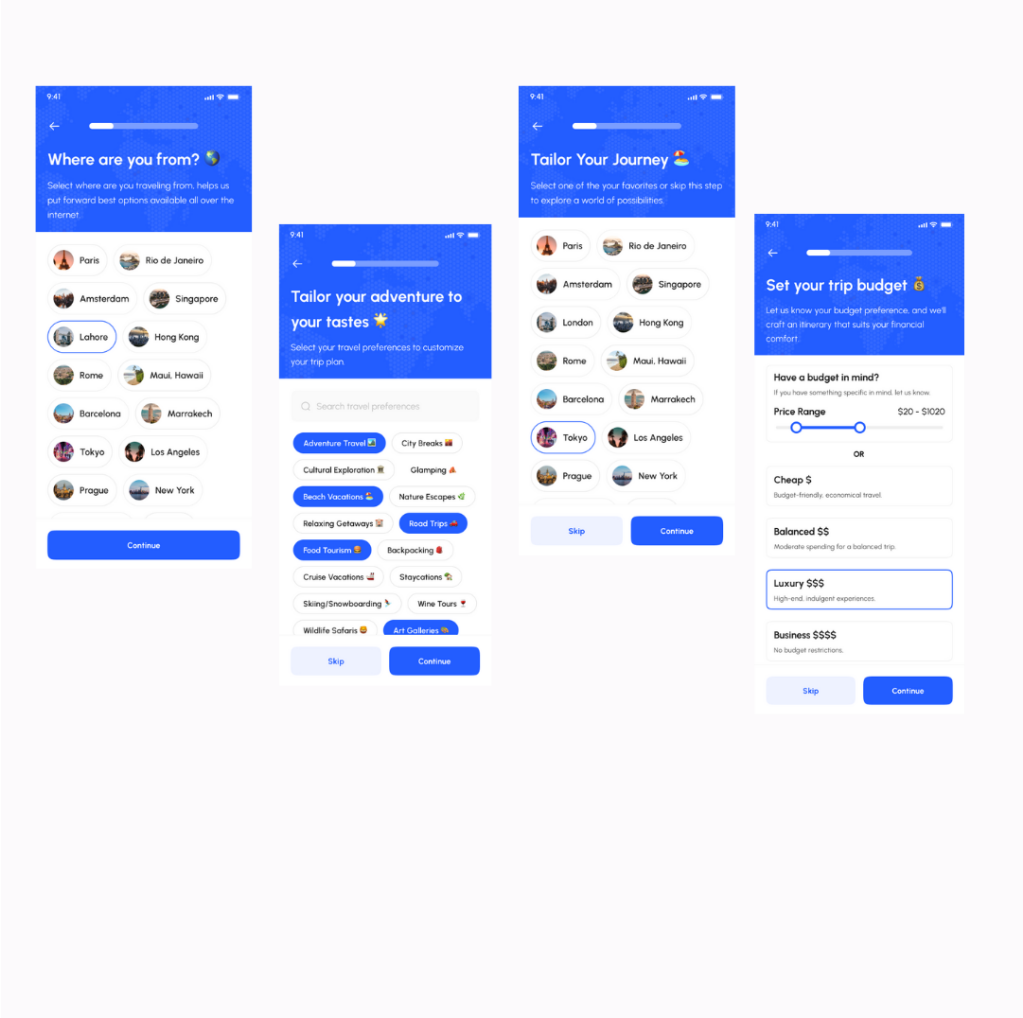
11.1.2 Home Screen

- Look at some amazing and popular places to visit, review the news and Examine the reviews and feedback from visitors.
- check your notifications
- Start generating itinerary for your travel.



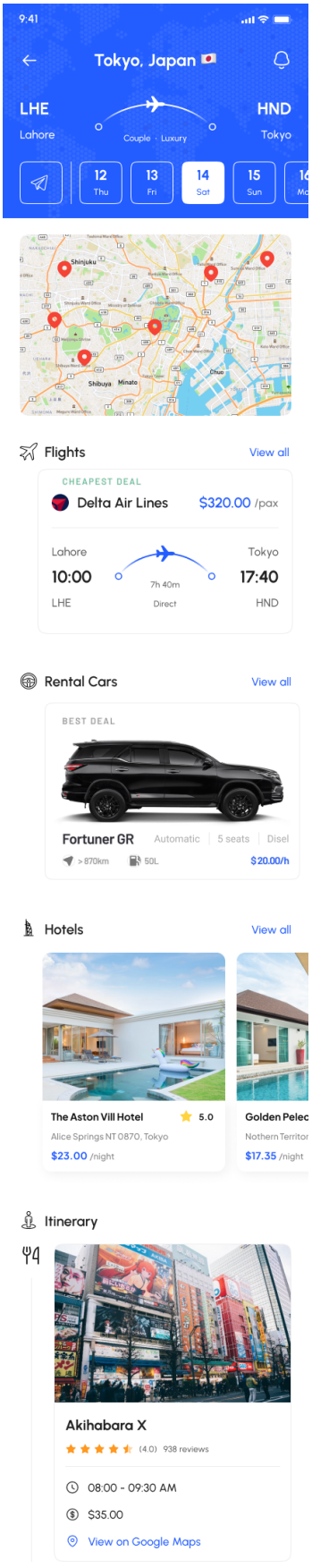
11.1.3 Manual listing selection Screens

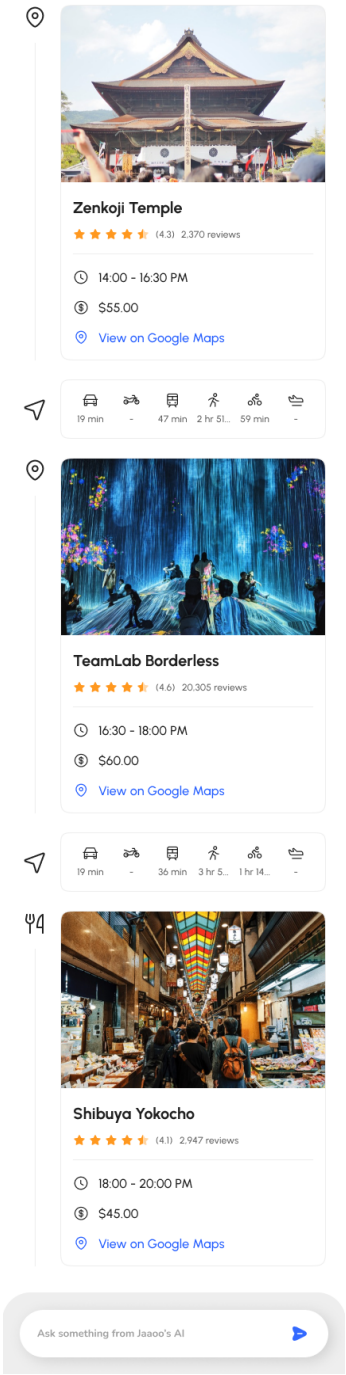
- Select from where you are, your destination and with whom you are going.
- Set a budget that is more suitable and friendly to you
- Generate itinerary for your travel according to your selection.



11.1.4 Itinerary Generated & Booking Screen

- Have look at the generated itinerary for you destination.
- Book the best cheapest possible options for your flights, hotels, rental cars.
- Wanna customize your itinerary a bit. write what you want to change in your itinerary and it will customize itinerary according to your need.


9:41
Tokyo, Japan
LHE
Lahore
Couple · Luxury
HND
Tokyo
12 Thu
13 Fri
14 Sat
15 Sun
Shinjuku
Shibuya
Minato
Chuo
Flights
View all
CHEAPEST DEAL
Delta Air Lines
$320.00 /pax
Lahore
10:00
LHE
7h 40m
Direct
Tokyo
17:40
HND
Rental Cars
View all
BEST DEAL
Fortuner GR
Automatic | 5 seats | Disel
>870km
50L
$20.00/h
Hotels
View all
The Aston Vill Hotel
5.0
Alice Springs NT 0870, Tokyo
$23.00 /night
Golden Pelec
Nothern Territor
$17.35 /night
Itinerary
Akihabara X
(4.0) 938 reviews
08:00 - 09:30 AM
$35.00
View on Google Maps


Zenkoji Temple
(4.3) 2,370 reviews
14:00 - 16:30 PM
$55.00
View on Google Maps
19 min
-
47 min
2 hr 51...
59 min
-
TeamLab Borderless
(4.6) 20,305 reviews
16:30 - 18:00 PM
$60.00
View on Google Maps
19 min
-
36 min
3 hr 5...
1 hr 14...
-
Shibuya Yokocho
(4.1) 2,947 reviews
18:00 - 20:00 PM
$45.00
View on Google Maps
Ask something from Jaaoo's AI

Chapter 12

Conclusion & Future Directions

12.1 Accomplishments

12.1.1 Successful Android & IOS App Development

An android app, having a user friendly interface to interact and create travel itinerary for your tour. With this it also provides the best cheapest possible options for the bookings of flights, hotels and rental cars.

12.2 Future Works

12.2.1 Improving User Experience

With the passage of time, we will continue to further enhance the user experience and make it more attractive and engaging for the users.

12.2.2 Addition of More countries

Currently the app provides the travel Itinerary options for top 10 most visited countries, but in future we will include more countries as well.

12.2.3 New Features

With the advancement of technology and with the passage of time, new features will be added to our app according to the feedback of users.

Bibliography

[1] M. Zsarnoczky, "How does artificial intelligence affect the tourism industry?" *Vadyba Journal of Management*, vol. 31, no. 2, pp. 85–90, 2017.

[2] J. P. Sonja Zlatanov, "Current applications of artificial intelligence in tourism and hospitality," *International Scientific Conference On Information Technology and Data Related Research*, 2019.

[3] A. R. T. Arjun Pesaru, Taranveer Singh Gill, "Ai assistant for document management using lang chain and pinecone," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 05, 2023.

[4] Y. H. Hui Yang, Sifu Yue, "Auto-gpt for online decision making: Benchmarks and additional opinions," *arxiv*, 2023.

[5] S. Gu, "A survey of large language models in tourism (tourism llms)," p. 31, 2024.

[6] C. Khawas and P. Shah, "Application of firebase in android app development-a study," *International Journal of Computer Applications*, vol. 179, no. 46, pp. 49–53, 2018.

[7] W. C. Yogesh K. Dwivedi, Neeraj Pandey, "Leveraging chatgpt and other generative artificial intelligence (ai)-based applications in the hospitality and tourism industry: practices, challenges and research agenda," *International Journal of Contemporary Hospitality Management*, p. 19, 2023.