كلية الحاسبات والذكاء الإصطناعي
Faculty of Computers & Artificial Intelligence

جامعة حلوان
Helwan University

HELWAN UNIVERSITY
Faculty of Computers and Artificial Intelligence
Information Systems Department

U GUIDE

A graduation project dissertation by:

Abdelrahman Mohamed Farraag Farag	**[201900436]**
Ahmed Mohamed Ibrahim Mohamed	**[202000063]**
Kareem Mohamed Sayed Abbas	**[202000672]**
Nancy Ehab Kamal Muhammed	**[202000981]**
Zeyad Magdy Elsayed Elmorsy	**[202000356]**
Yehia Hamdy Ahmed Mohamed	**[202001060]**

Submitted in partial fulfilment of the requirements for the degree of Bachelor of Science in Computers & Artificial Intelligence at the Information Systems Department, the Faculty of Computers & Artificial Intelligence, Helwan University.

Supervised by:

Dr. Helal Soliman Ahmed

June 2024

U GUIDE

EXPLORING EGYPT, WITH ONE CLICK

Acknowledgment

First and foremost, we would express our deepest gratitude and appreciation to our Advisor **Dr. Helal for** his support, outstanding guidance, and encouragement throughout our graduation project.

We would like to thank our families, especially our parents. We hope they are proud of us in our last year of education, we hope that we will start giving back to the community very soon! Thank you for supporting us andproviding us with the needed time, effort, encouragement, patience, and assistance over the years.

Finally, our faculty for providing us with the courses that guided us into the right direction of our lives and the help of all the professors that left agreat impact on our lives, **Thank you**.

Table of Contents

Chapter 1

Introduction

1.1 Overview

Welcome to U Guide, your premier destination for exploring the wonders of Egypt like never before! Whether you're an intrepid traveler or a curious explorer, our platform is your indispensable companion for unlocking the mysteries of this ancient land.

At U Guide, we seamlessly blend cutting-edge technology with timeless adventure to offer you an unparalleled experience. Our web and mobile application are meticulously crafted to guide you through every step of your journey.

Powered by innovative machine learning algorithms, our application boasts a range of features designed to enhance your exploration. Meet our friendly chatbot, your knowledgeable virtual guide ready to assist you 24/7. Whether you need recommendations for sightseeing, assistance with bookings, or insider tips on local customs, our chatbot is here to make your experience seamless and unforgettable.

But that's not all—prepare to be amazed by our Hieroglyphics Translator, a groundbreaking tool that allows you to decode the ancient symbols of Egypt with just a few taps. Whether you're standing before the majestic temples of Luxor or exploring the hidden tombs of the Valley of the Kings, our translator brings the secrets of the past to life, bridging the gap between ancient history and modern technology.

Navigate through Egypt's legendary landmarks with ease, from the iconic Pyramids of Giza to the bustling markets of Khan El-Khalili. Discover hidden gems off the beaten path and uncover the stories behind each stone, each carving, and each whispered legend.

At U Guide, we're not just guiding you to destinations—we're crafting unforgettable experiences that resonate long after your journey ends. Join us as we embark on a voyage through time and space, where every moment is infused with wonder, mystery, and the magic of Egypt.

Your adventure begins here. Welcome to U Guide, where the past meets the future, and every discovery is a journey unto itself. Let's explore Egypt together.

1.2 Objectives

Mainly U Guide focusing on traveling to Egypt offers a gateway to one of the most captivating destinations in the world. With its rich history, mesmerizing ancient sites, vibrant culture, and breathtaking landscapes, Egypt has always been a dream destination for travelers seeking adventure, exploration, and enlightenment. Let's delve into the features of your website, particularly the chatbot and hieroglyphics translator, and discuss their importance in enhancing the user experience and facilitating travelers' journeys to Egypt.

Chatbot Feature:
Imagine having a knowledgeable travel companion available 24/7 to assist you with every aspect of your trip to Egypt. That's precisely what the chatbot feature on your website offers. Whether users have questions about visa requirements, the best time to visit the Pyramids of Giza or recommendations for authentic Egyptian cuisine, the chatbot is there to provide instant, personalized assistance. This feature streamlines the trip planning process, alleviates concerns, and ensures that users feel supported and confident as they embark on their Egyptian adventure.

Hieroglyphics Translator Feature:
Egypt's ancient civilization has left behind a legacy of fascinating hieroglyphic inscriptions, which serve as a window into the past. Your website's hieroglyphics translator feature allows users to decode these ancient symbols, unlocking the secrets of Egypt's history and culture in an interactive and educational manner. Whether users are intrigued by the inscriptions on the walls of Karnak Temple or curious about the meaning behind King Tutankhamun's burial artifacts, the hieroglyphics translator empowers them to explore and understand Egypt's ancient past.

1.3 Purpose

The purpose of this project is to create an immersive and informative online platform that inspires and assists travelers in exploring the wonders of Egypt. By leveraging technology and interactive features, the project aims to provide a comprehensive resource for individuals interested in experiencing the rich history, culture, and landscapes of Egypt firsthand. Through practical tools such as a chatbot for personalized assistance and a hieroglyphics translator for cultural exploration, the project seeks to enhance the user experience and facilitate meaningful connections with Egypt's heritage. Ultimately, the purpose is to empower travelers to plan and embark on unforgettable journeys to Egypt, fostering a deeper appreciation for this extraordinary destination and creating lasting memories for visitors from around the world.

1.4 Scope

The scope of this project encompasses the development and implementation of an online platform dedicated to travel to Egypt, featuring a range of interactive features and resources aimed at enhancing the user experience and promoting engagement. The project includes the following key components:

1. Website Development: Designing and building a user-friendly website that serves as the central hub for all content related to travel to Egypt. The website will provide comprehensive information about Egypt's attractions, culture, history, and more.

2. Feature Integration: Implementing interactive features such as a chatbot and hieroglyphics translator to enrich the user experience. The chatbot will offer personalized assistance and recommendations to users, while the hieroglyphics translator will allow users to decode ancient symbols and explore Egypt's cultural heritage.

3. Content Creation: Developing high-quality content, including articles, guides, and multimedia resources, to educate and inspire travelers about Egypt. Content will cover various topics such as must-visit landmarks, and cultural traditions.

5. Mobile Optimization: Ensuring that the website is fully optimized for mobile devices to provide a seamless browsing experience for users accessing the platform on smartphones and tablets.

6. Search Engine Optimization (SEO): Implementing SEO best practices to improve the visibility and ranking of the website in search engine results, thereby attracting more organic traffic and increasing its reach.

Overall, the scope of this project is to create a comprehensive and engaging online platform that serves as a go-to destination for travelers interested in exploring Egypt, offering practical tools, inspiring content, and personalized assistance to enhance their journey and create memorable experiences.

1.5 General constraints

While the project aims to create a comprehensive and engaging online platform for travelers interested in exploring Egypt, there are several constraints that need to be considered:

1. Budget Limitations: The project's budget may be limited, constraining the allocation of resources for website development, feature implementation, content creation, marketing, and ongoing maintenance.

2. Time Constraints: There may be a tight timeline for the project, requiring efficient planning, execution, and delivery of various tasks within a specified timeframe.

3. Technical Limitations: Technical constraints such as compatibility issues, platform limitations, and software dependencies may impact the development and functionality of the website and its features.

4. Resource Availability: The availability of skilled personnel, including developers, content creators, designers, and marketing professionals, may be limited, affecting the project's execution and timeline.

5. Data Privacy and Security: Ensuring compliance with data privacy regulations and implementing robust security measures to protect user data and sensitive information are essential constraints that must be addressed throughout the project.

6. Content Accuracy and Legality: Ensuring the accuracy, legality, and authenticity of the content published on the website, including images, text, and multimedia materials, is crucial to avoid potential legal issues and maintain credibility.

7. Localization and Cultural Sensitivity: Adhering to cultural sensitivities and local regulations in Egypt, as well as considering language barriers and localization requirements, are important constraints to consider when developing content and features for the platform.

8. User Accessibility: Ensuring that the website is accessible to users with disabilities and adheres to accessibility standards and guidelines is essential to provide an inclusive experience for all users.

By addressing these constraints proactively and implementing effective strategies to mitigate risks, the project can overcome obstacles and achieve its objectives of creating a valuable and engaging online platform for travelers interested in exploring Egypt.

Chapter 2

Planning And Analysis

2.1 Project Planning

2.1.1 Feasibility Study

A Feasibility study is defined as the practical extent to which a project can be performed successfully and used to determine if a business or a specific project is achievable, to determine the achievability of our project we'll go deeper into the following points:

Market Feasibility:

After many searches, we found that the market needs a tourist community application as there is no application to help users deal with tourist statues and hieroglyphic inscriptions also the system is very easy to use, and there is no need for any training to know how to use the system. This will help in spreading our new application widely.

Technical Feasibility:

Project U Guide is a complete web-based application. The main technologies and tools that are associated with U Guide are.

- HTML CSS JavaScript
- Reacts
- Python
- Node Js

Each of the technologies is freely available and the technical skills required are manageable, Time limitations of the product development and the ease of implementing using these technologies are synchronized.

Initially, the website will be hosted in a free web hosting.
space, but for later. Implementations will be hosted in a
paid web hosting space with sufficient bandwidth required.
(his application is very low).

From these, it's clear that the project U Guide is technically feasible.

Resource and Time Feasibility

Resource feasibility

Resources that are required for the U Guide project include,

- Programming device (Laptop)
- Hosting space (freely online server)
- Programming tools (freely available)
- Programming individuals

Project U Guide has the required resource feasibility.

Social/Legal Feasibility

U Guide uses freely available development tools and provides a service for all people. Software libraries that are used in this system are free open-source libraries.

2.1.2 Estimated Cost

One-time Cost:

Requirements	Cost in (EGP)
Development Cost	50.000
Hardware Cost	50.000
Total	100.000

Recurring Cost:

Requirements	Cost in (EGP)
Software Maintenance	20.000
Incremental Data Storage	10.000
New/Update Hardware	25.000
Server cost	25.000
Total	80.000

2.1.3 Gantt Chart

Chapter Name	Start Date	End Date	Duration	Target Achievement
Chapter 1	01/09/2023	08/09/2023	8 Days	100%
Chapter 2	09/09/2023	16/09/2023	8 Days	100%
Chapter 3	17/09/2023	01/10/2023	14 Days	100%
Chapter 4	12/10/2023	30/03/2024	208 Days	100%
Chapter 5	01/04/2024	09/4/2024	9 Days	100%
Chapter 6	10/04/2024	21/4/2024	11 Days	100%
Chapter 7	23/04/2024	30/04/2024	8 Days	100%
Chapter 8	01/05/2024	08/05/2024	8 Days	100%

Start Date: 1/9/2023 End Date: 8/5/2024 Total Days: 250 days

2.2 Analysis and Limitation of existing system

The main limitations of the current system

The existing system, presumably referring to traditional methods of accessing information about travel to Egypt, likely has several limitations compared to the proposed online platform. Here's an analysis of the main limitations of the current system and factors contributing to its slow operation:

1. Limited Accessibility: Information about travel to Egypt may be scattered across various sources such as guidebooks, travel agencies, and government websites, making it challenging for users to access comprehensive and up-to-date information in one place.

2. Time-Consuming Research: Travelers need to spend considerable time researching multiple sources to gather relevant information about Egypt's attractions, accommodations, transportation, visa requirements, safety tips, and cultural insights, leading to inefficiency and frustration.

3. Lack of Personalization: The current system may not offer personalized assistance or recommendations tailored to individual preferences, travel interests, or budget constraints, resulting in a one-size-fits-all approach that may not meet users' specific needs or preferences.

4. Language Barriers: Information may be available only in certain languages, posing challenges for users who do not speak or understand those languages fluently, limiting their access to valuable resources and insights about traveling to Egypt.

5. Limited Interactivity: Traditional sources of information such as guidebooks or brochures are static and lack interactivity, preventing users from engaging with the content in a dynamic or immersive way and inhibiting their ability to explore Egypt's culture and history in depth.

6. Dependency on Human Assistance: Users may rely on travel agents or tour operators for assistance with trip planning, booking arrangements, and obtaining information, which can be time-consuming, costly, and may not always result in personalized or accurate recommendations.

Things that make the current system operate slowly.

1. Manual Processes: The reliance on manual processes for gathering, updating, and disseminating information about travel to Egypt can lead to delays and inefficiencies, particularly when information needs to be verified, consolidated, or distributed across multiple channels.

2. Limited Technology Adoption: Lack of adoption of modern technology and digital tools for information dissemination, communication, and transaction processing can hinder the speed and effectiveness of the current system, resulting in delays and bottlenecks.

Overall, the existing system's limitations stem from its reliance on traditional methods of information dissemination, lack of personalization and interactivity, language barriers, and inefficiencies in communication and processes. Addressing these limitations by transitioning to an online platform with interactive features, personalized assistance, multilingual support, and streamlined processes can significantly enhance the user experience and improve the efficiency and effectiveness of accessing information about travel to Egypt.

2.3 Need for the new system

Our project has new features that the old systems have that’s why our new system for exploring Egypt applies the following features:

- Voice Assistant:

  Using artificial intelligence (AI) software makes use of message applications, blogs, and telephones to simulate a conversation or a chat with a person. Voice Assistant is a program designed to automate a particular process.

- Hieroglyphics translator:

  You can unlock the mysteries of ancient Egypt with ease. Enter a hieroglyphic photo of any hieroglyphic inscription you encounter, and our advanced machine-learning algorithms will decode it in real-time, providing you with valuable insights into the rich history and culture of this fascinating civilization.

Beyond these innovative features, U Guide offers a wealth of resources and tools to enhance your journey. From curated recommendations for must-see attractions.

2.4 Analysis of the new system

2.4.1 User requirements

- This explains what the user will get through the site or application that you are using or the user what will do inside the system.

- Often referred to as user needs, describe what the user does with the system, such as activities that users should be able to do. User requirements are generally documented in the User Requirements Document (URD) using narrative text. User requirements are signed in Generic by the user and used as the primary input for creating system requirements.

- The system must have a user-friendly interface to make it easy for the user to use it.

- The registration form must be clear enough and understandable for The user to sign up easily.

- Security/Reliability/Maintainability/Availability.

Our Program will consist of Two users:

1- User:

- He can create an account for the platform by registering his basic data (Name - Email - Password - Nationality)
- He can then log in (using email - password)
- If he forgot the password, click on forget password.
- He can click on the boxes to learn about program.
- He can learn about the pyramids, Egyptian hieroglyphs, Khan El Khalil, the Cairo Tower, the Nile River, and the Red Sea.
- He can chat with a chatbot to get assistance as a tourist in Egypt.
- He can translate hieroglyphic characters to English with ease.
- He can learn about us (The Program) and contact us.
- He can logout anytime from the system.

2- Admin:

- Log in to the platform.
- See all users registered on the platform.
- He can add or delete admins.
- He can view any complaints for users.
- Can add or delete or update users.

Often referred to as user needs, describe what the user does with the system, such as what activities that users must be able to perform.

2.4.2 System Requirements

- This is considered the answer to the question - How will this system be built? What software, hardware, and network will you work on?

    1) I mean, what is the software that I use for my son to appreciate this system?
    2) My database is where I use it, for example, or where is it?
    3) What programming languages are used in my system?

- Browser: The application works on any browser ex.” Chrome, Microsoft edge, Internet explorer”

- Internet connection: This app uses a cloud server to store data, so you need to be connected to the internet to fetch and view it.

2.4.3 Domain Requirements

1. Travel Content Management:
   - Ability to manage and update information related to various destinations, and attractions in Egypt.
   - Integration with reputable travel APIs or databases to provide accurate and up-to-date information.

2. Language and Translation Services:
   - Integration of hieroglyphics translation functionality, allowing users to translate English text to hieroglyphics and vice versa.

3. User Authentication and Personalization:
   - Secure user registration and authentication mechanisms.

4. Customer Support and Assistance:
   - Implementation of a chatbot module for handling common inquiries, providing travel tips, and assisting users throughout their journey.
   - Option for live chat support with human agents for complex issues or personalized assistance.

2.4.4 Functional Requirements

1. Registration

- **User Registration (1.1):**
  - The system shall require the user to provide their full name, a valid Gmail address, a password meeting complexity requirement (minimum length, character types), password confirmation for verification, and nationality during registration.
  - The system shall display clear instructions and validation messages for each registration field (e.g., "Password must contain at least 8 characters, including uppercase, lowercase, numbers, and symbols").
- **User Addition (1.2):**
  - Upon successful completion of the registration process, the system shall add the user's information to the database.
  - The system shall display a confirmation message to the user indicating successful registration.

2. Login as User

- **User Login (2.1):**
  - The system shall allow the user to login using their registered email address or username (if username functionality is implemented).
  - The system shall prompt the user for their password upon entering their email address/username.
  - The system shall implement secure password hashing for user authentication.
- **Login Notification (2.2):**
  - Upon successful login, the system shall display a clear notification message to the user, indicating they are logged in.
- **Password Reset (2.3):**
  - The system shall provide a functionality for users to reset their forgotten password.
  - The system shall offer options like sending a password reset link to the user's registered email address or answering security questions (if implemented).
  - The system shall guide the user through the password reset process with clear instructions and feedback messages.

3. Login as Admin

- **Admin Login (3.1):**
  - The system shall require the admin to use their registered email address and password to log in.
  - The system shall implement secure password hashing for admin authentication (separate from user authentication).

4. Admin Dashboard Access

- **View Users (4.1):**
  - The system shall display a list of all registered users on the platform within the admin dashboard.
  - The list shall provide relevant user information such as name, email address, and registration date.
- **Access User Complaints (4.2):**
  - The system shall allow the admin to access and review user-submitted complaints within the dashboard.
  - The system shall display details associated with each complaint, including the user who submitted it, the nature of the complaint, and any relevant screenshots or attachments.
- **User Account Management (4.3):**
  - The system shall grant the admin the ability to add, delete, or update user accounts.
  - The system shall provide clear functionalities for each action within the dashboard.
  - The system shall prompt for confirmation before permanent deletion of a user account.
- **Admin Account Management (4.4):**
  - The system shall allow for the creation or deletion of additional admin accounts, managed by the primary admin.
  - The system shall enforce security measures for admin account management, such as two-factor authentication (if applicable).
- **Informative Content for Users:**

  Informative Cards/Flip Cards:

  5.1 Cards shall be provided for users to click on and access information about tourist places.

  5.1.1 Detailed descriptions of tourist places shall be available to be viewed by users.

  5.2 Photos shall be available for users to swipe through and discover popular destinations.

6. Application provides tourism U Guide assistant:

- The user will be able to submit a request to the site to the U Guide assistant to learn more about the culture in Egypt.

6.1 The user can talk with a U Guide assistant model and text.

6.2 The U Guide assistant response with inquiries in text and voice.

7. Application provides Hieroglyphics Translator model:

- The application will provide translation for uploaded hieroglyphics code.

7.1 The user can upload a photo with Hieroglyphics language.

7.2 users can look at the information that appears to them.

7.3 This information is stored in a database.

2.4.5 Non-Functional Requirements (NFRs)

A. Look and Feel

1. The system shall use a limited color palette (two or three colors of the same shade) for a visually comfortable experience.
2. The user interface shall be designed with eye comfort in mind, promoting a relaxing and non-straining experience.
3. The system shall utilize clear and easily recognizable icons for intuitive navigation.

B. Usability & Humanity

4. The system shall provide well-designed icons that are self-explanatory, facilitating easy understanding and use regardless of language comprehension.
5. Users shall have easy access to lists (e.g., opening a blocking list or dialer) for efficient management.
6. Editing, removing, adding, and selecting functionalities within lists (e.g., settings for a blocking list) shall be readily accessible and logically organized.
7. The system shall offer clear and understandable tips for first-time users to navigate the platform effectively.
8. Users shall have the option to submit complaints about bugs or errors to ensure a smooth user experience.
9. Users shall be able to easily update and edit their profiles for better personalization.

C. Performance

- **Response Time:**
  - Website and application: Respond promptly with minimal latency.
  - Chatbot: Provide quick and accurate responses within 2 seconds.
  - Hieroglyphics Translator: Translate text efficiently, rendering results within 5 seconds.

D. Reliability

- **Uptime:**
    - Website and application: Be available 24/7 with a minimum uptime of 99.9%.
    - Chatbot and Hieroglyphics Translator: Be reliable and stable with minimal downtime.

E. Scalability

- The system should handle increasing user traffic without performance degradation.
- The chatbot and Hieroglyphics translator should scale horizontally to accommodate a growing user base.

F. Security

- **Data Protection:** Encrypt user data (personal information & chat transcripts) during transmission and storage.
- **Authentication:** Implement secure mechanisms to prevent unauthorized access.
- **Vulnerability Management:** Conduct regular security audits and updates to mitigate potential vulnerabilities.

G. Compatibility

- **Devices and Browsers:** Support a wide range of devices (desktops, laptops, tablets, smartphones) and browsers.
- **Platforms:** Ensure compatibility with popular messaging platforms and operating systems for chatbot and translator.

H. Maintainability

- **Codebase:** Maintain well-documented and organized code for website, application, chatbot, and translator.
- **Version Control:** Utilize version control systems for effective codebase management.

I. Accessibility

- **WCAG Compliance:** Ensure the website and application comply with accessibility standards like WCAG for users with disabilities.

2.5 Advantages of the new system

- Enhanced User Experience: Users will enjoy an interactive and engaging experience on your platform, thanks to features like the chatbot for instant assistance and the Hieroglyphics translator for cultural exploration.

- Convenience and Accessibility: Users can access travel-related information, support, and cultural resources conveniently through your website and application, enhancing accessibility for travelers planning to visit Egypt.

- Educational Value: The Hieroglyphics translator adds educational value by allowing users to learn about ancient Egyptian writing systems and culture while engaging with the content.

- Improved Customer Service: The chatbot provides instant responses to user queries, improving customer service by addressing concerns and aiding in real time, even outside of regular business hours.

- Differentiation and Competitive Edge: Offering unique features like a Hieroglyphics translator sets your platform apart from competitors, attracting users who are interested in exploring Egypt's rich cultural heritage.

- Increased Engagement and Retention: Interactive features encourage users to spend more time on your platform, increasing engagement and fostering a sense of loyalty among visitors.

- Data Insights: The chatbot can gather valuable insights into user preferences, frequently asked questions, and areas of interest, which can inform future content creation and marketing strategies.

- Brand Building: By providing innovative and valuable features, your platform can establish itself as a trusted resource for travelers interested in Egypt, contributing to brand building and reputation enhancement.

2.6 Risk and Risk Managements

1. Technical Risks:

- Integration Issues: There might be compatibility issues when integrating the chatbot and Hieroglyphics translator with your website and application.

- Performance Challenges: Ensuring the smooth performance of these features, especially under high user loads, could be challenging.

Risk Management:

- Conduct thorough testing during the development phase to identify and address integration issues early. Implement performance monitoring and scaling strategies to handle increased traffic.

2. Data Security Risks:

- Data Breaches: Storing user data, including chat transcripts, poses a risk of data breaches if adequate security measures are not in place.

- Privacy Concerns: Users might be concerned about the privacy of their personal information shared through the chatbot.

Risk Management:

- Implement robust data encryption techniques to protect sensitive user data.

3. Cultural Sensitivity Risks:

- Misinterpretation: The Hieroglyphics translator might inadvertently convey incorrect or culturally insensitive translations.

- Offensive Content: Users may generate or encounter offensive content while using the chatbot or Hieroglyphics translator.

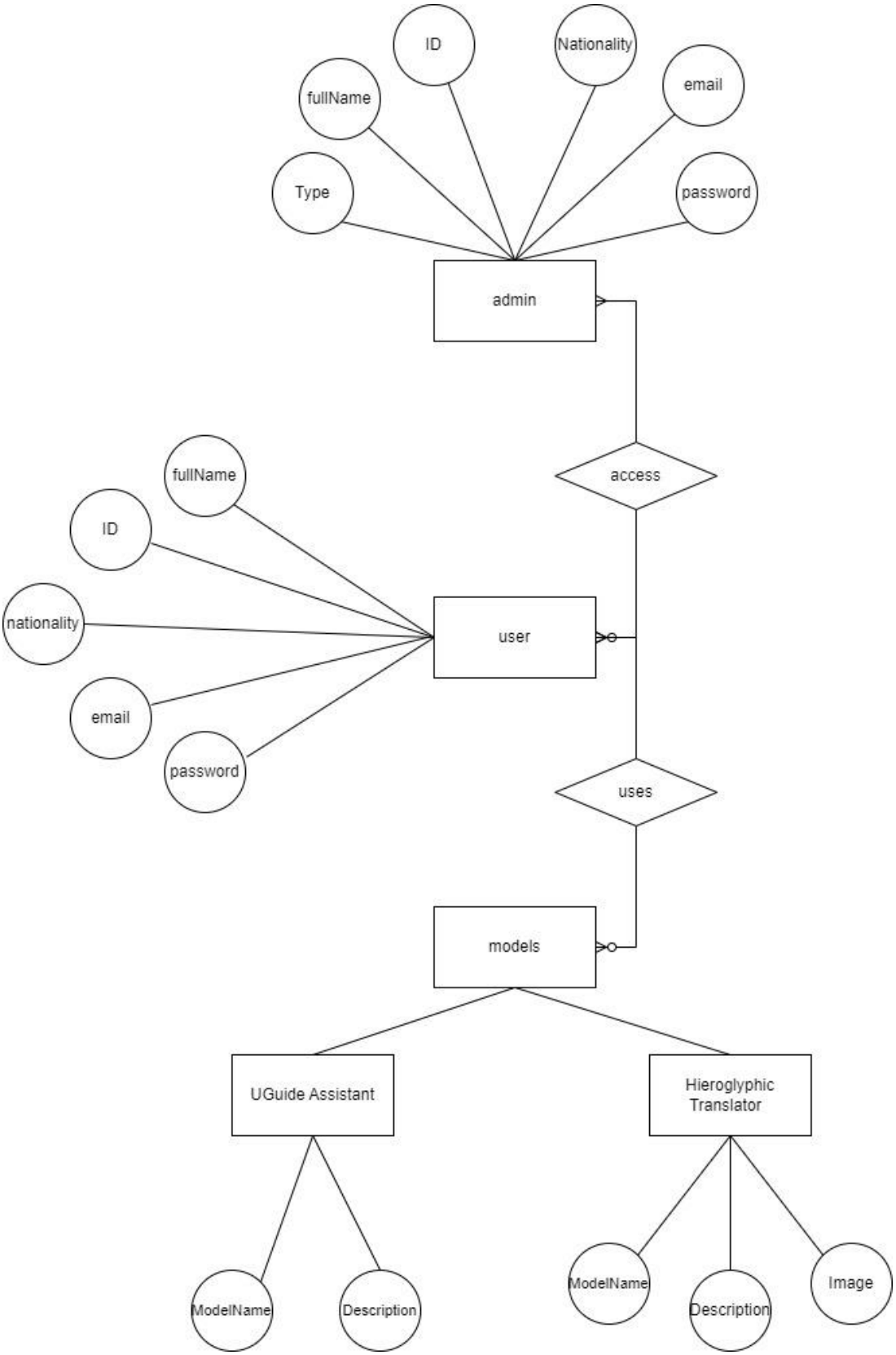
Risk Management:

- Implement filters and moderation mechanisms to prevent the generation and dissemination of offensive content.

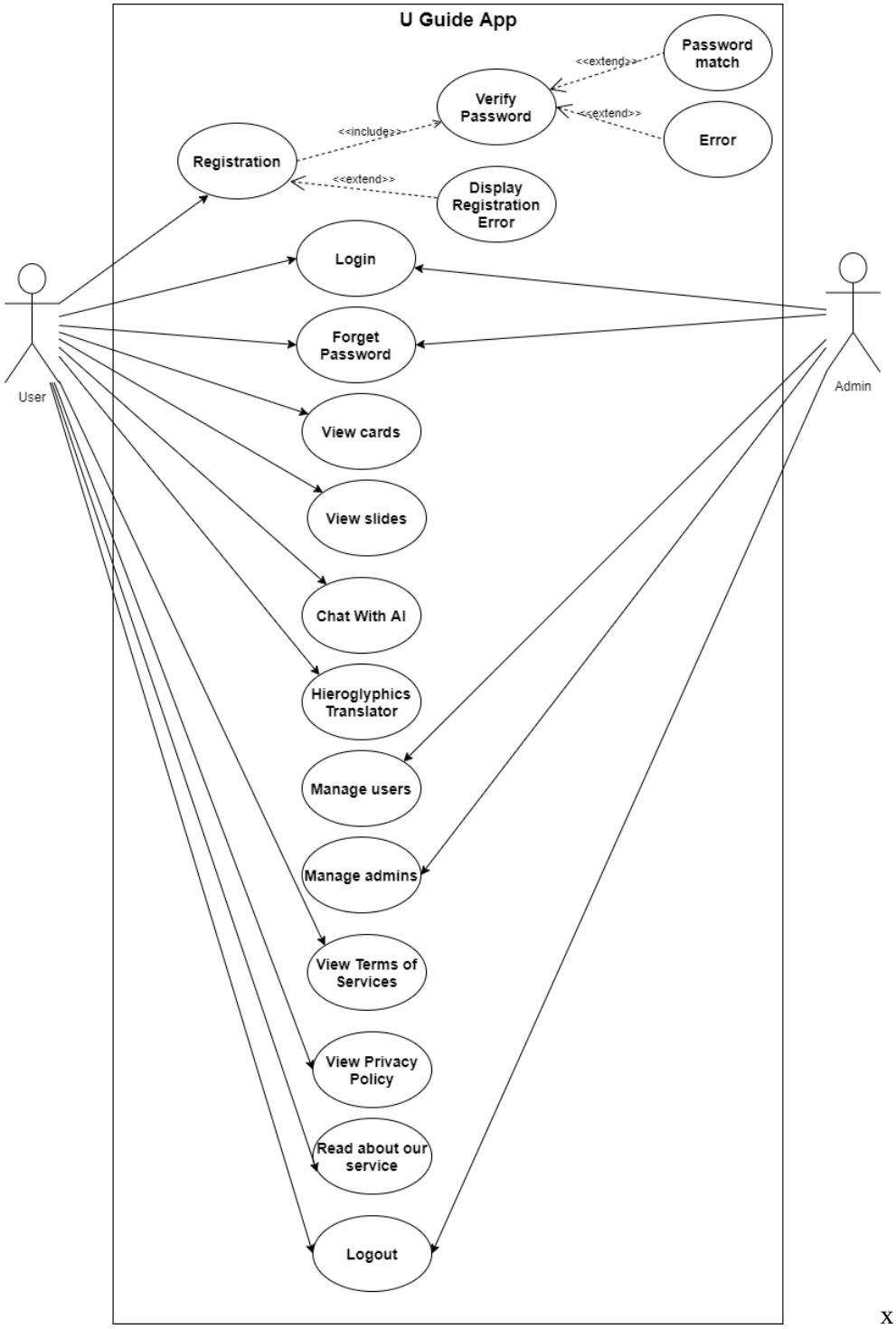
Chapter 3

SYSTEM DESIGN

3.1 ERD Diagram



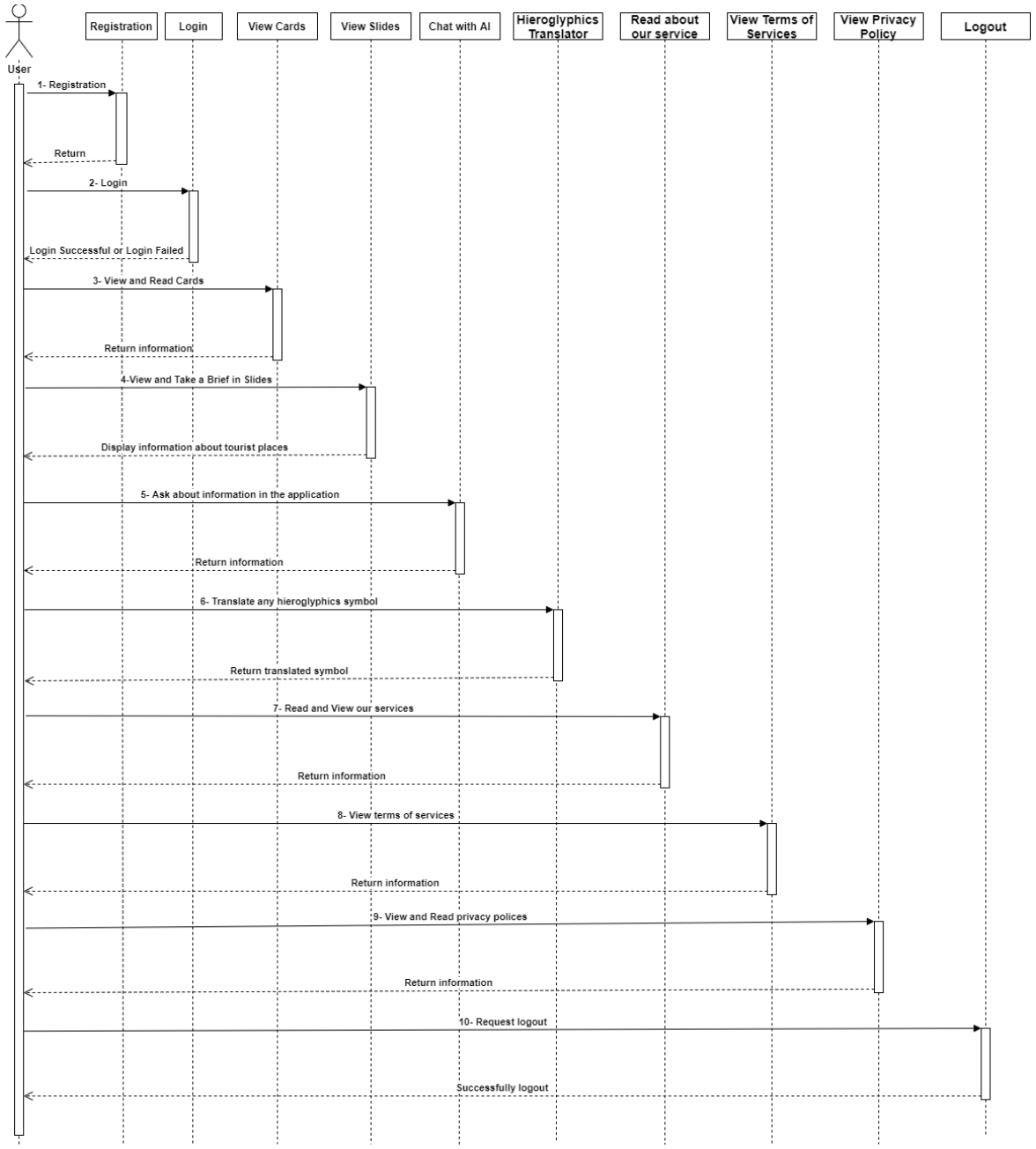
3.2 Use Case Diagram:



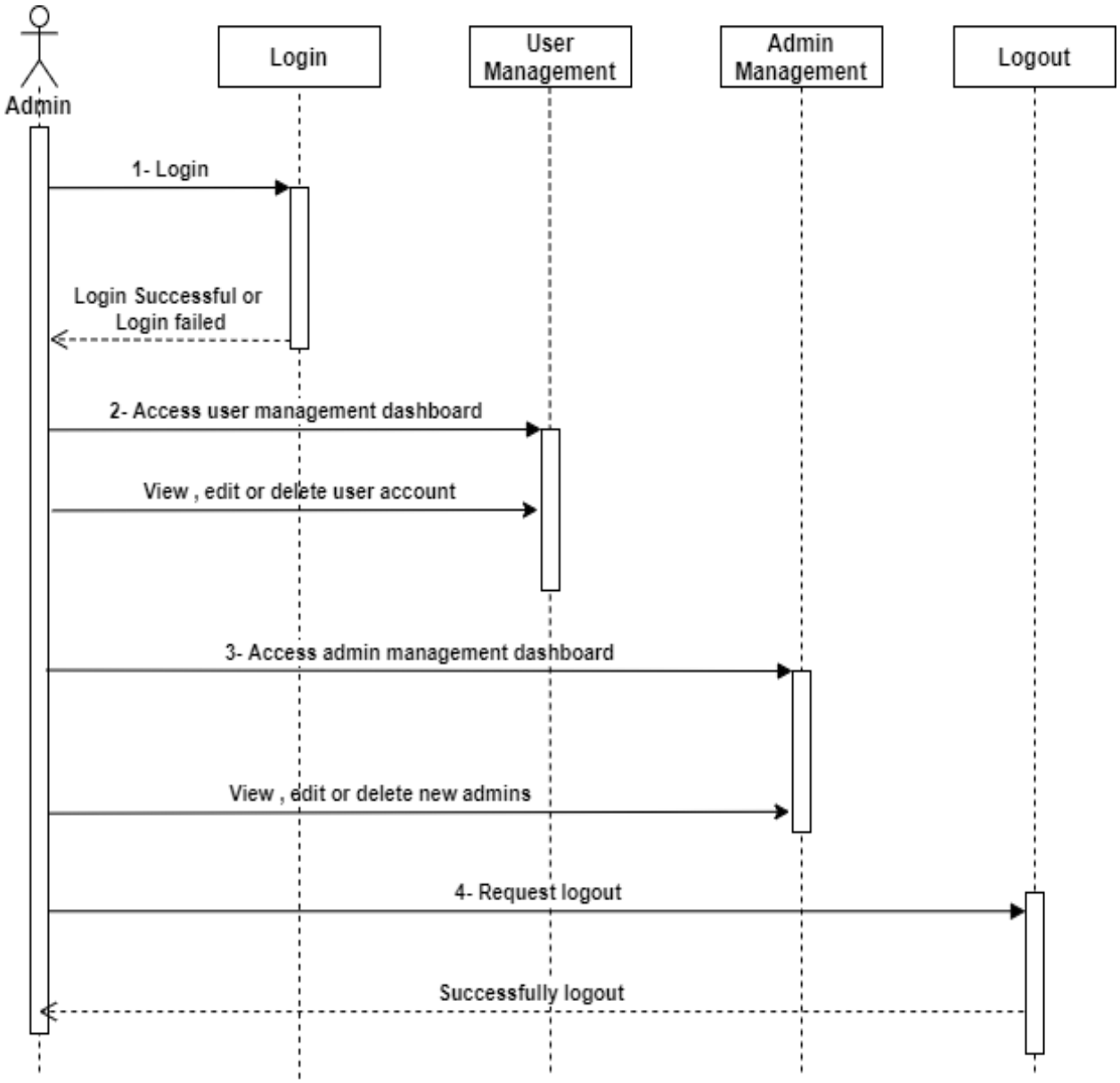
x

3.3 Sequence Diagram:

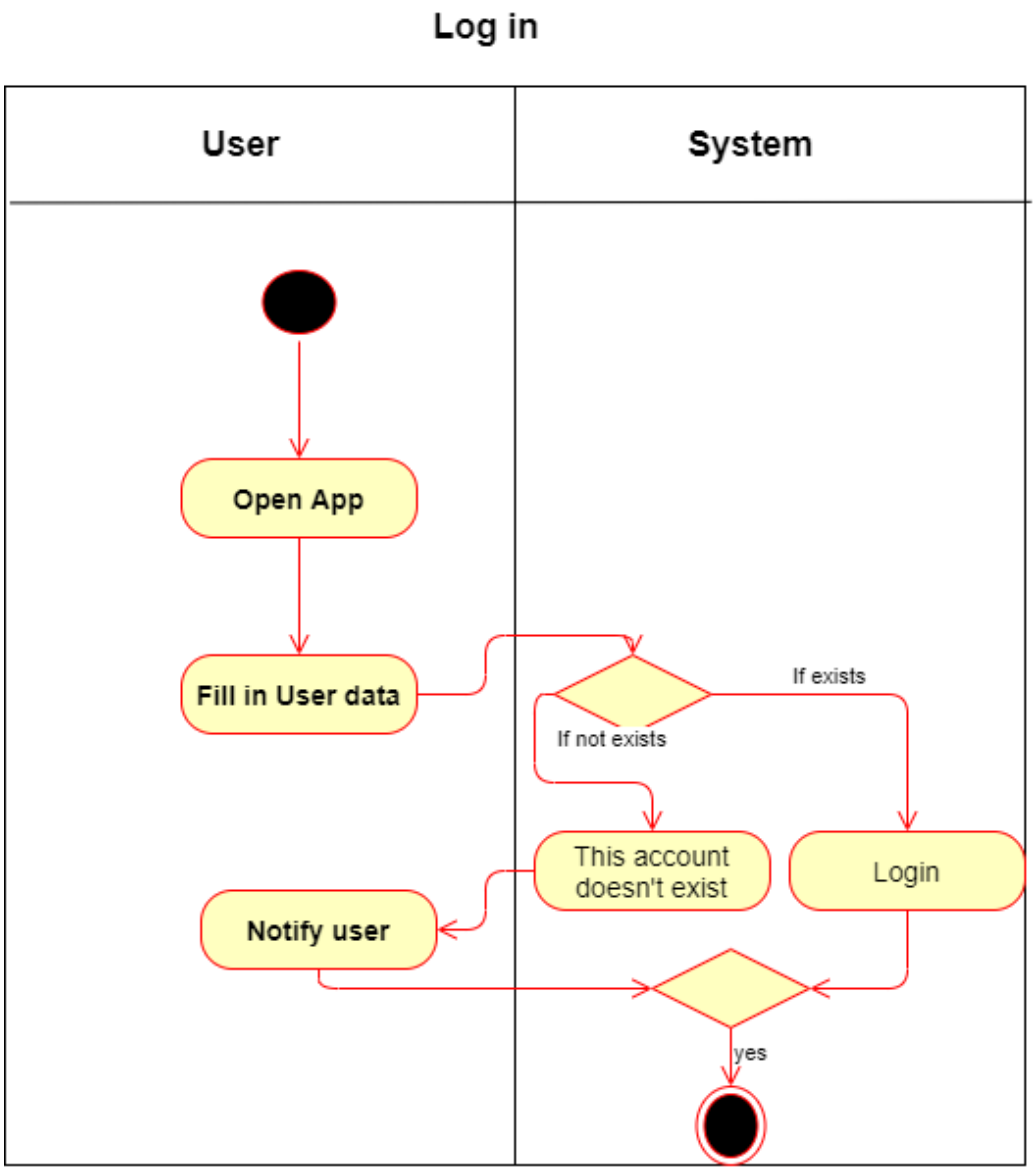
1- User



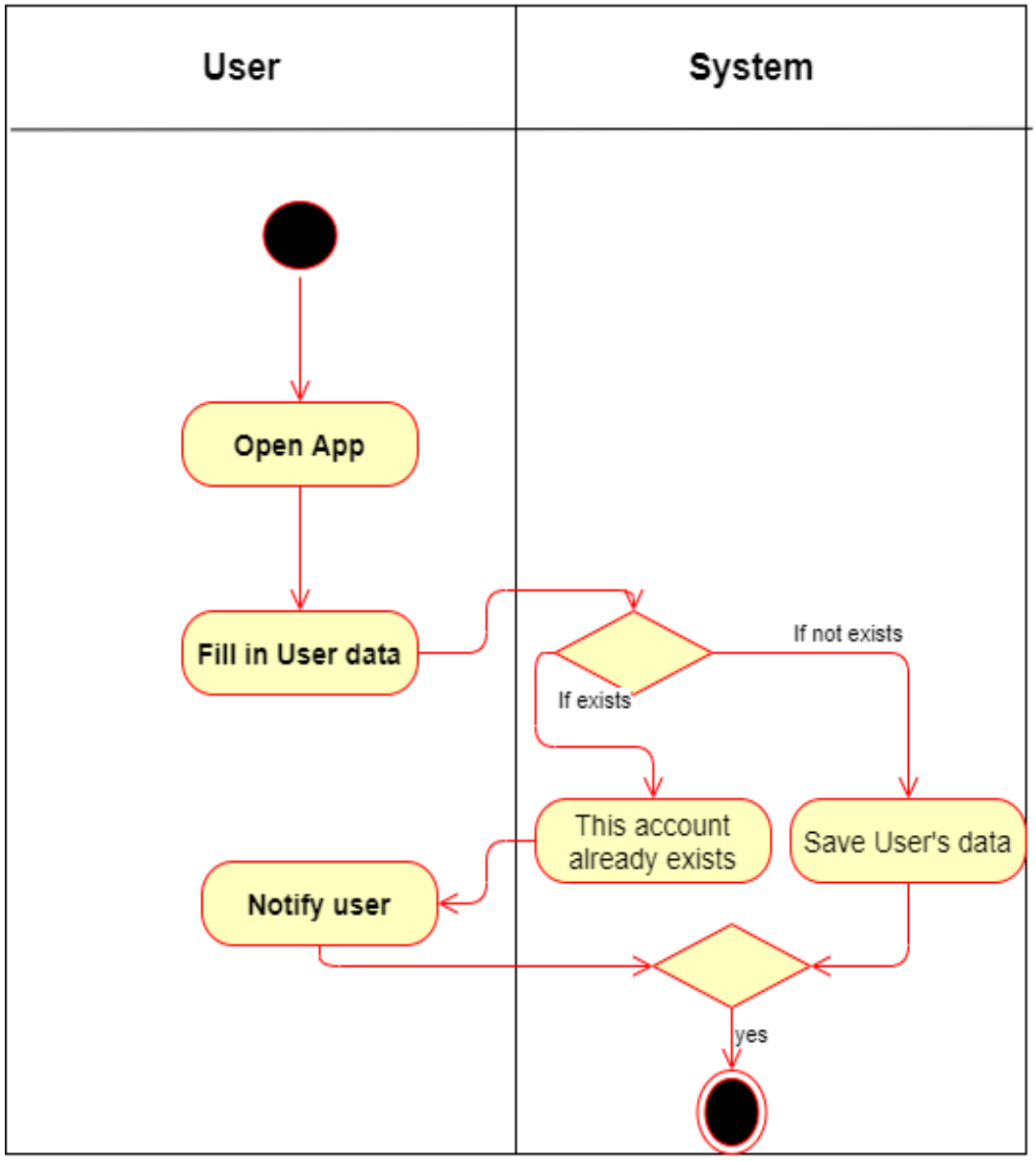
2- Admin


Admin
Login
User Management
Admin Management
Logout
1- Login
Login Successful or Login failed
2- Access user management dashboard
View , edit or delete user account
3- Access admin management dashboard
View , edit or delete new admins
4- Request logout
Successfully logout

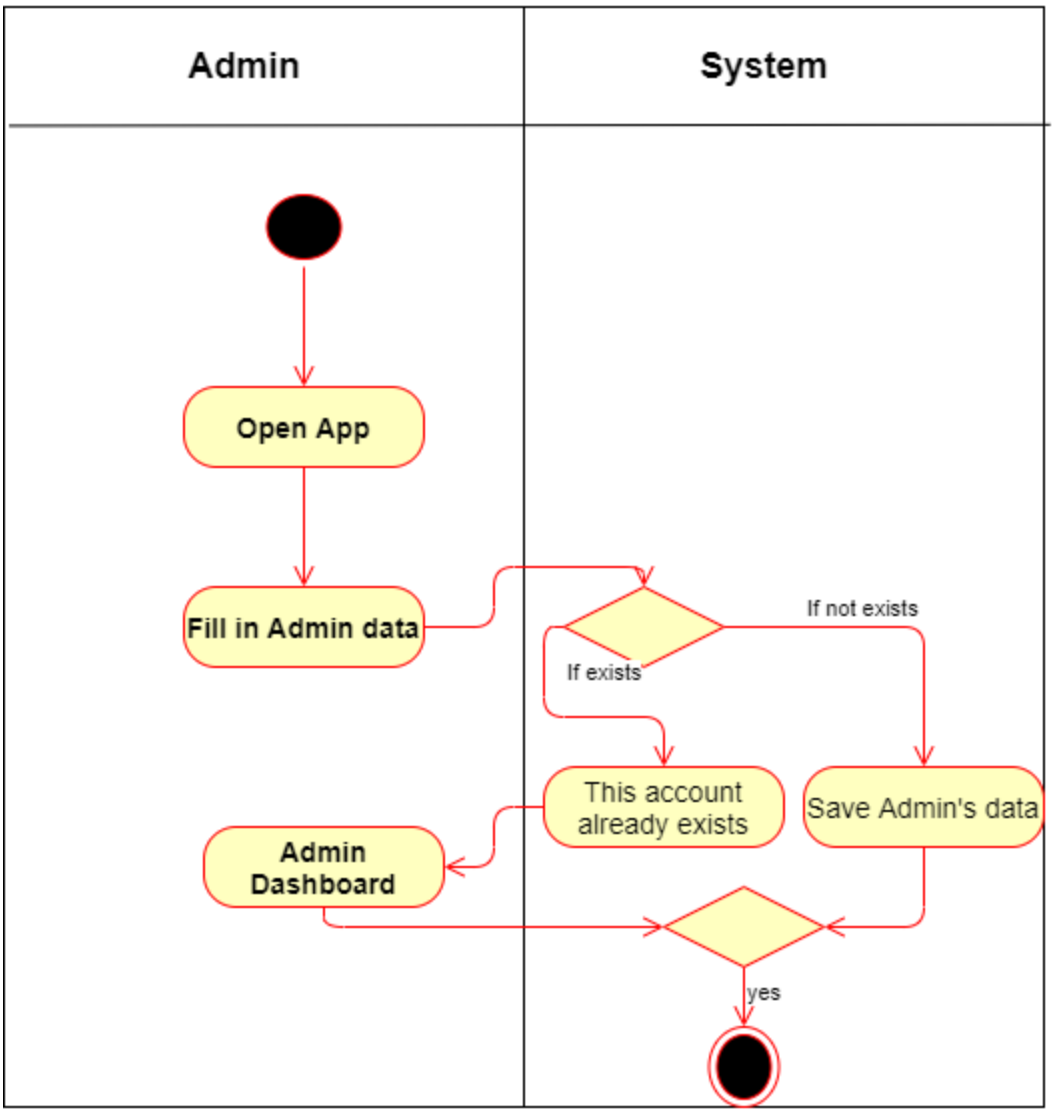
3.4 Activity Diagram:



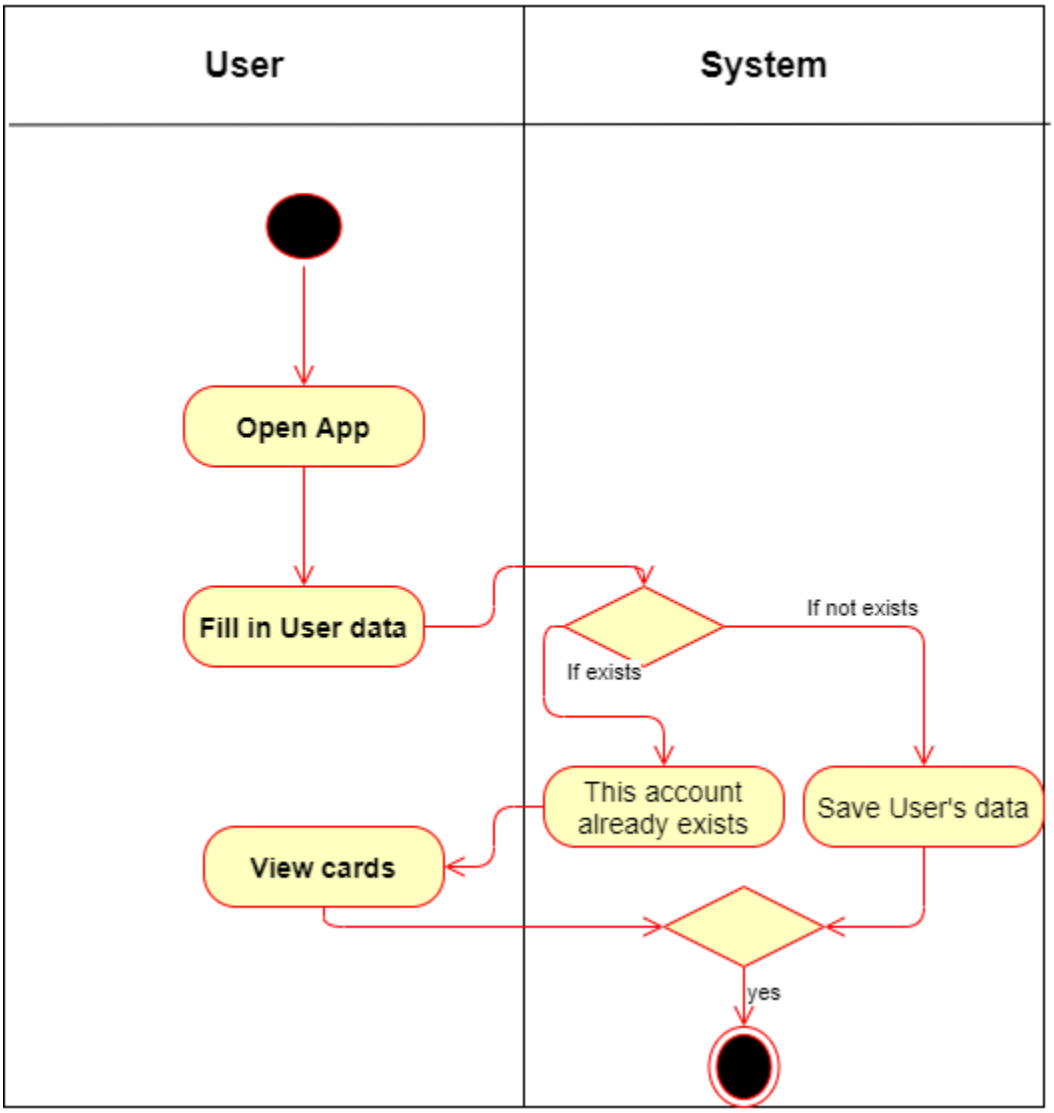
Registration



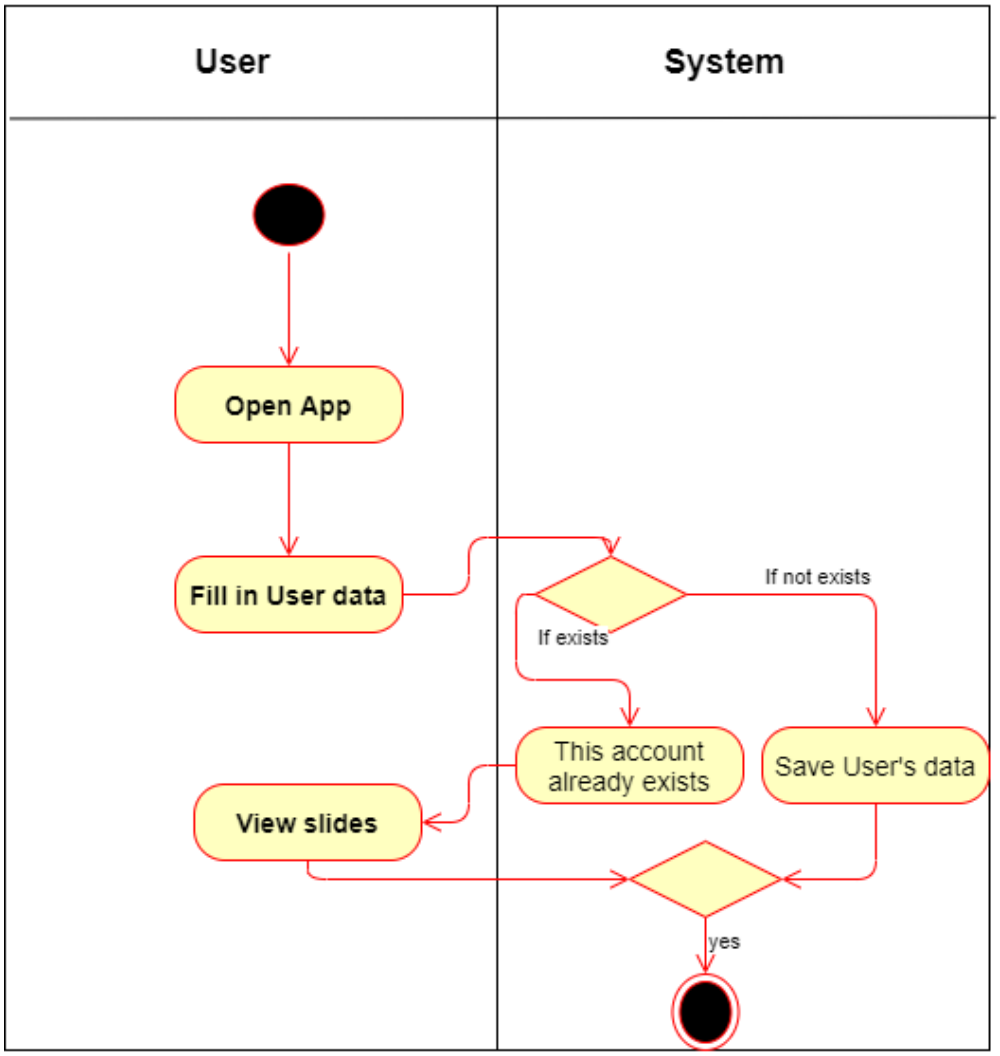
Admin Dashboard



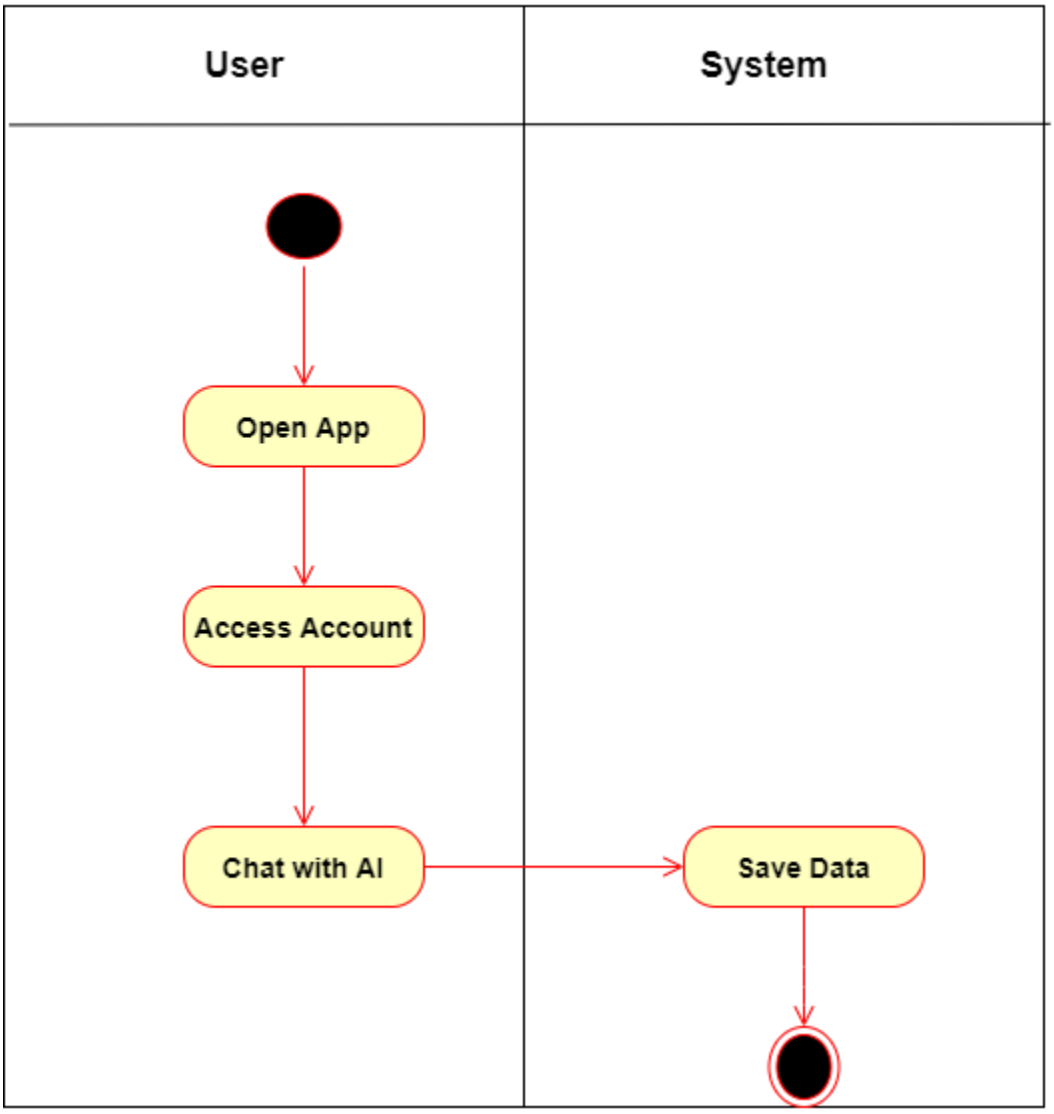
View Cards

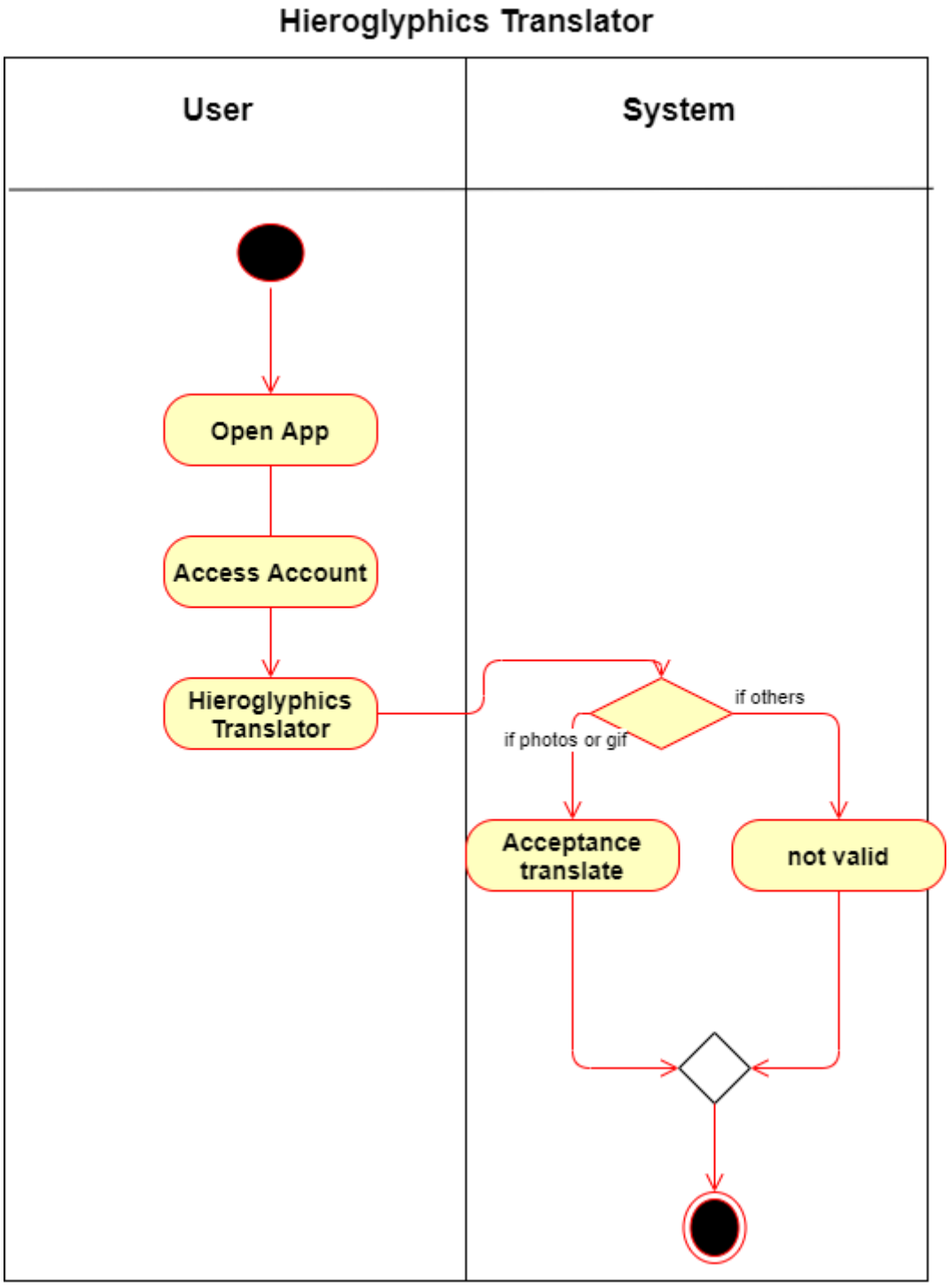

User
System
Open App
Fill in User data
If not exists
If exists
This account already exists
Save User's data
View cards
yes

View Slides

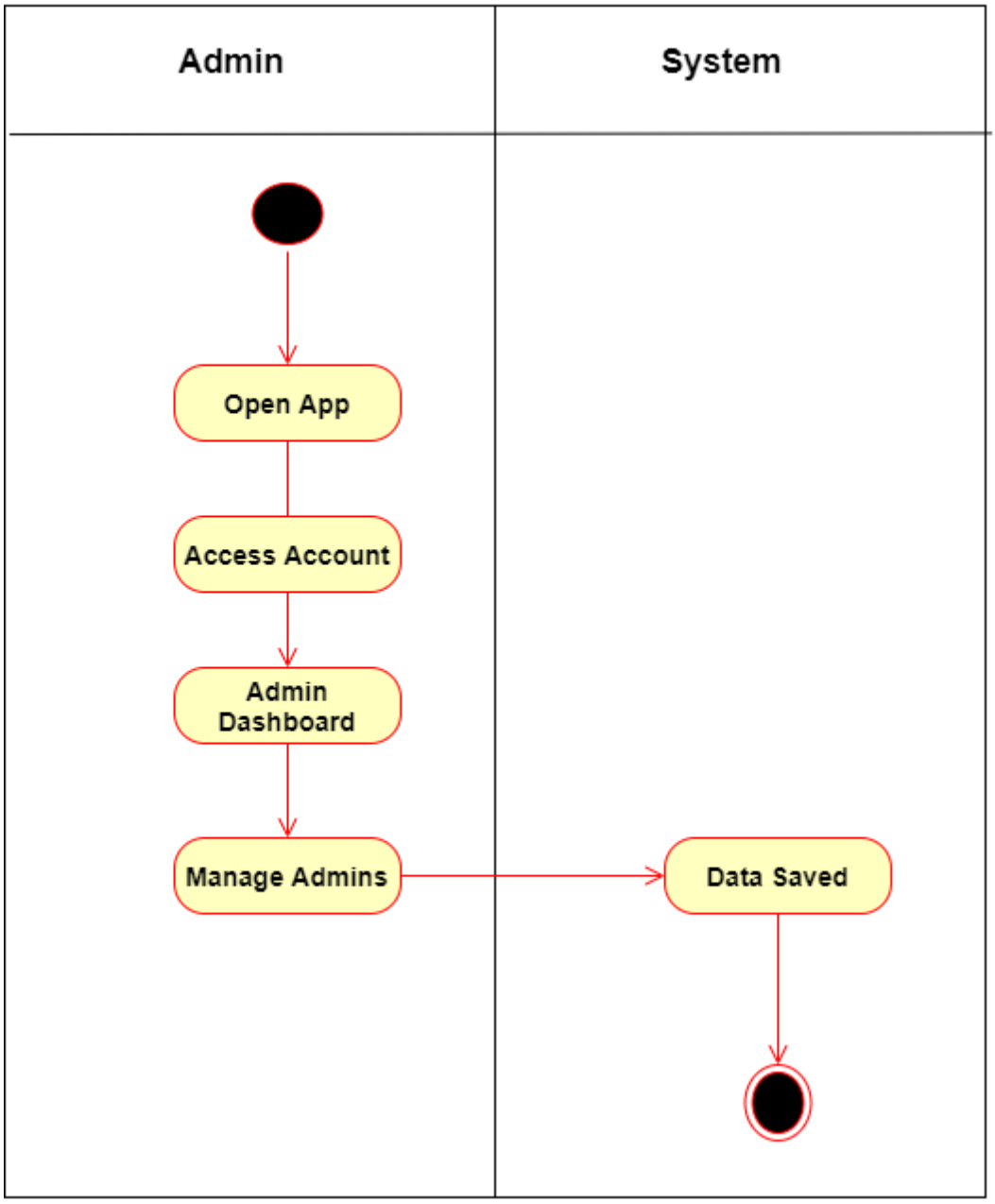

User
System
Open App
Fill in User data
If not exists
If exists
This account already exists
Save User's data
View slides
yes

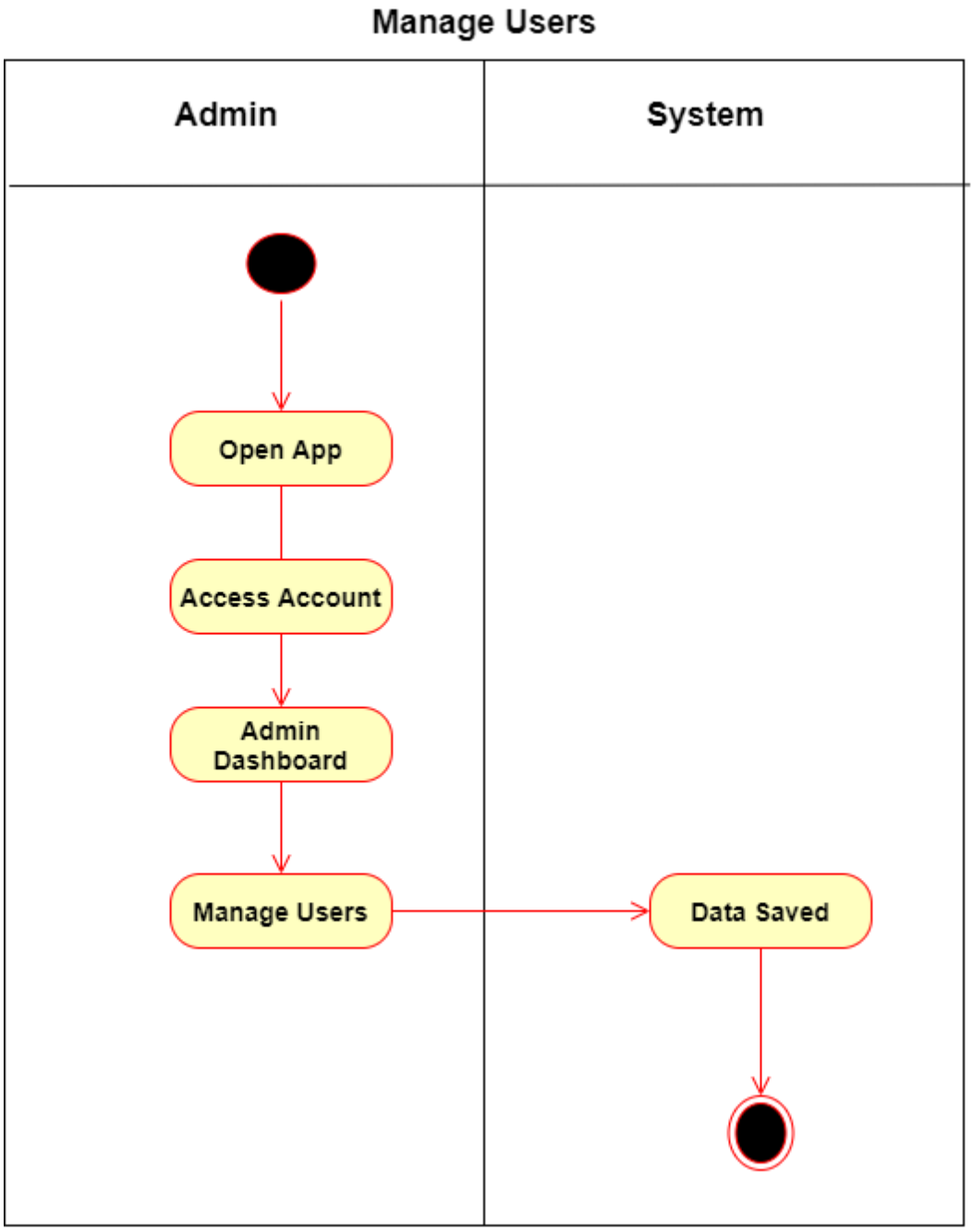
Chat With AI




Hieroglyphics Translator
User
System
Open App
Access Account
Hieroglyphics Translator
if others
if photos or gif
Acceptance translate
not valid

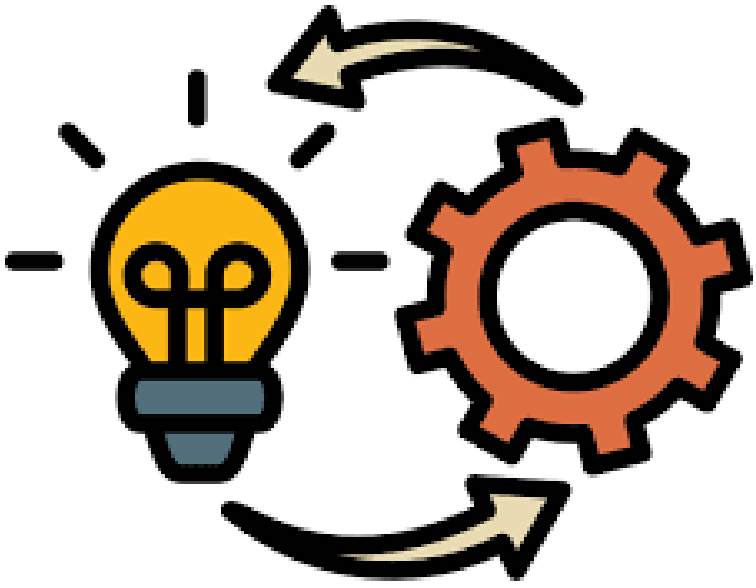
Manage Admins




Manage Users
Admin
System
Open App
Access Account
Admin Dashboard
Manage Users
Data Saved

Chapter 4
Implementation

In this chapter we will show the architecture of the application and snapshots of the main functions. We will also mention the technologies we used.



4.1 Software tools and Technologies

Front-End: React

React is a powerful JavaScript library for building user interfaces. It allows developers to create reusable components that update dynamically based on user interaction or data changes. React is known for its speed, efficiency, and large community, making it a popular choice for building modern web applications.

Back-End: Node.js

Node.js is an open-source runtime environment built on Chrome's JavaScript engine. It allows developers to write server-side applications using JavaScript, creating a more unified development experience. Node.js is known for its asynchronous nature, making it efficient at handling multiple requests simultaneously.

Machine Learning: Python

Python is a general-purpose programming language that has become a favorite for machine learning due to its readability, extensive libraries, and large community. Popular libraries like TensorFlow and PyTorch provide tools for building and training machine learning models. Python's clear syntax makes it easier to learn and understand even for beginners.

Fast API

Fast API is a Python framework for building APIs (Application Programming Interfaces). It simplifies the process of creating APIs by providing automatic data validation, dependency injection, and automatic documentation generation. Fast API allows developers to focus on the core logic of their API while the framework handles the boilerplate code.

Visual Studio

Visual Studio is a powerful and versatile Integrated Development Environment (IDE) created by Microsoft. It provides a comprehensive set of tools and features for building software of all types. Visual Studio supports many programming languages, including those mentioned here, making it a popular choice for developers across various disciplines.

Postman

Postman is a popular API client used for testing, debugging, and documenting APIs. It allows developers to send HTTP requests, view responses, and manage different API endpoints. Postman is a valuable tool for ensuring APIs function as expected before integrating them into applications.

MongoDB

MongoDB is a NoSQL document database that stores data in JSON-like documents. It offers flexibility and scalability compared to traditional relational databases, making it a good choice for storing and managing large amounts of unstructured or semi-structured data.

JavaScript (JS)

JavaScript is a high-level, interpreted programming language primarily used for web development. It allows developers to create dynamic and interactive web pages. JS plays a crucial role in both front-end (user interface) and back-end (server-side) development using Node.js.

HTML

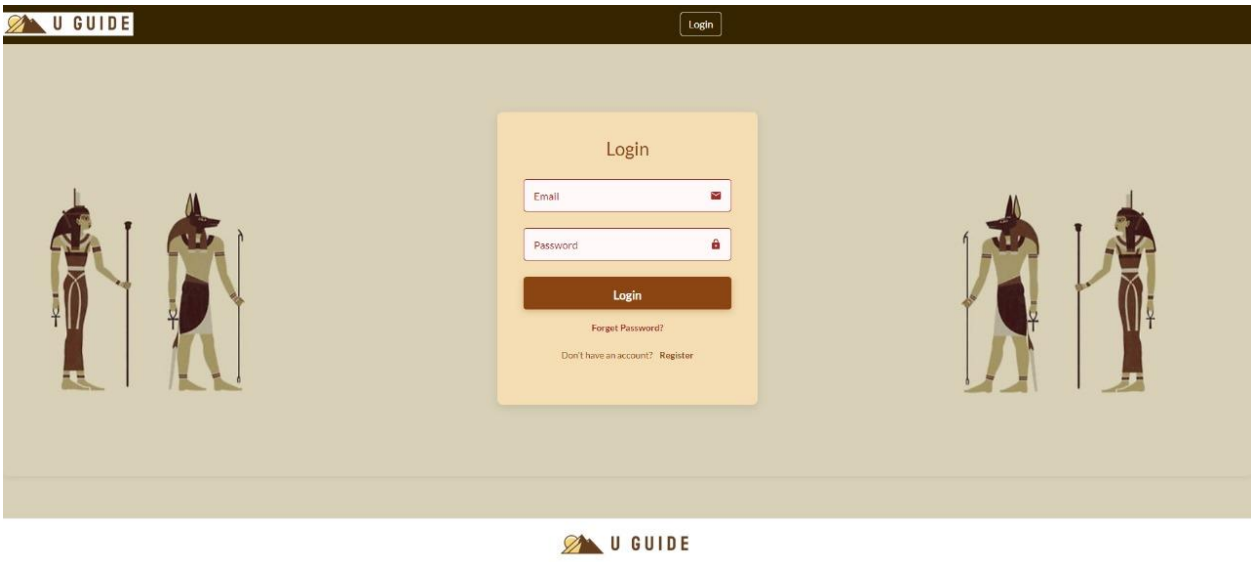
HTML (HyperText Markup Language) is the standard markup language for creating web pages. It defines the structure of a web page, including elements like headings, paragraphs, images, and forms. HTML provides the foundation for all web pages and is essential for building web applications.

CSS

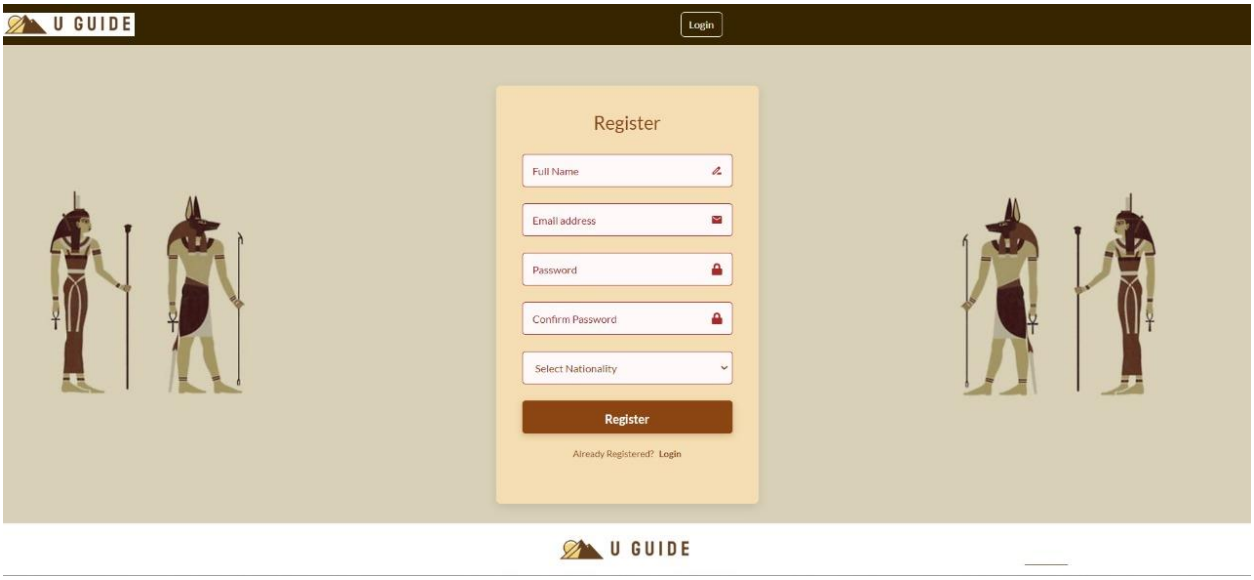
CSS (Cascading Style Sheets) defines the presentation of a web page, including layout, colors, fonts, and other visual elements. CSS allows developers to style HTML elements and create a visually appealing and consistent user interface. Together, HTML and CSS form the core building blocks of web development.

Front End Implementation:

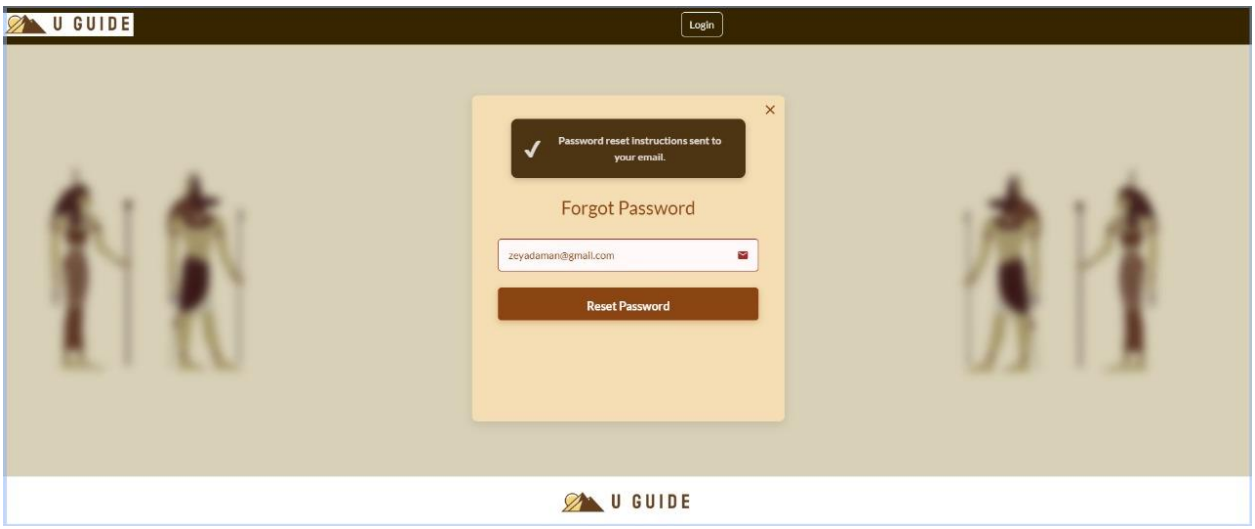
1.Login Page



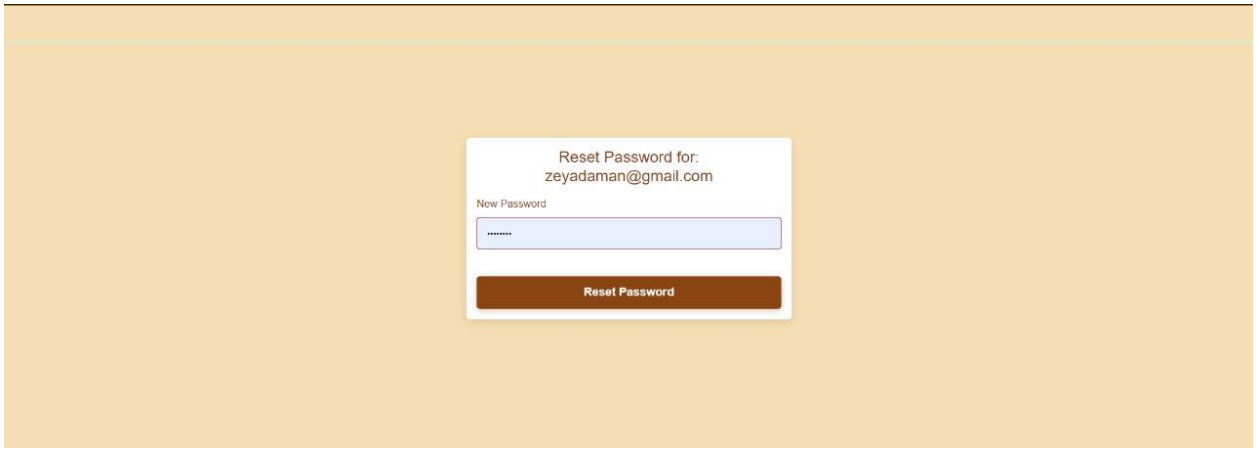
2. Registration



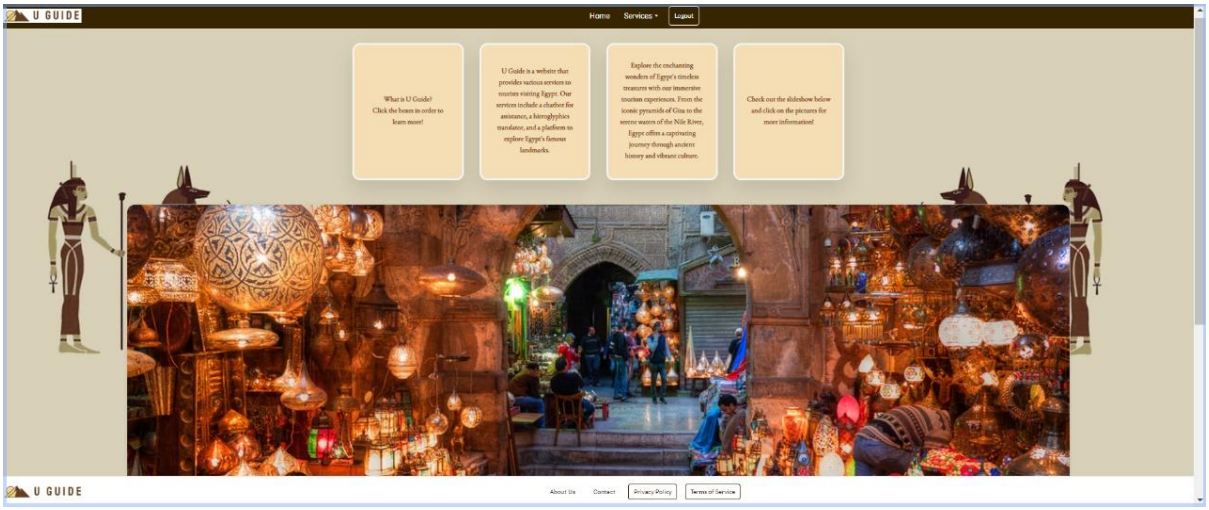
3.Forget Password

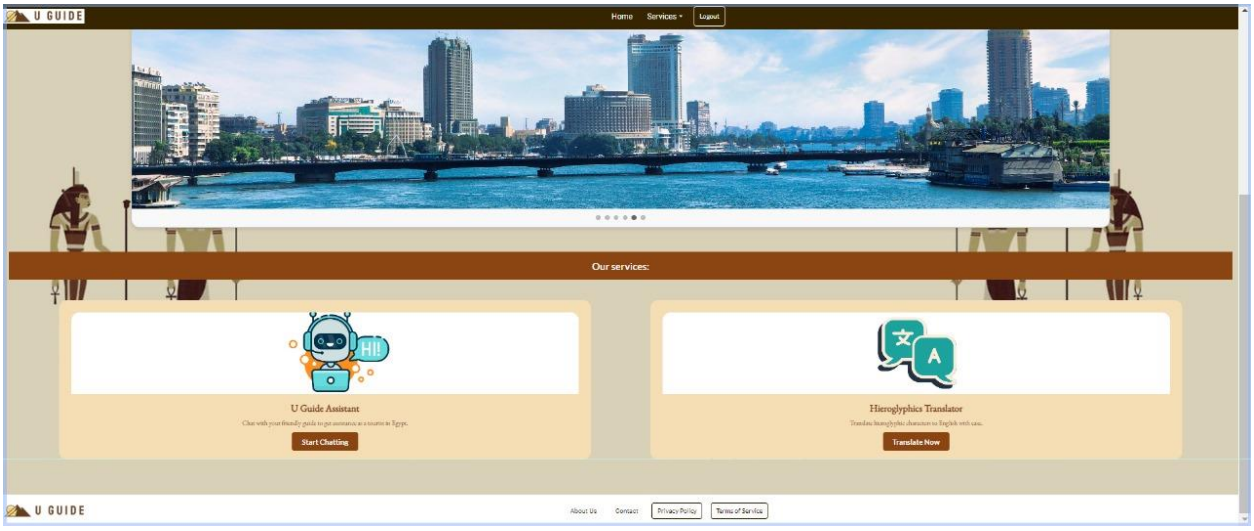


3.1 Reset Password

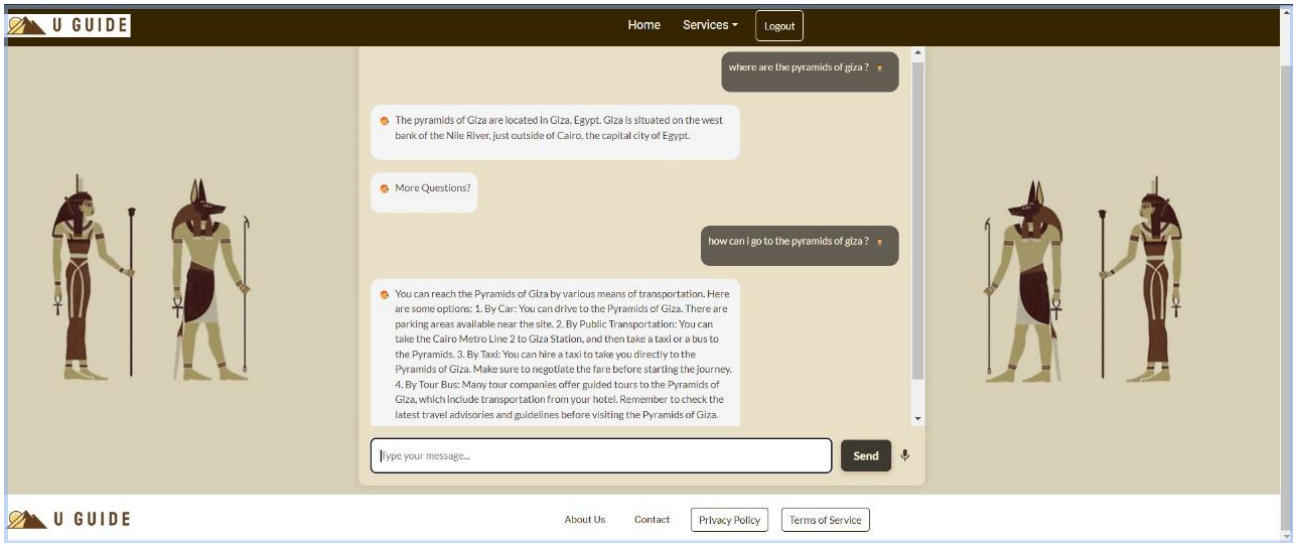


4. Landing Page

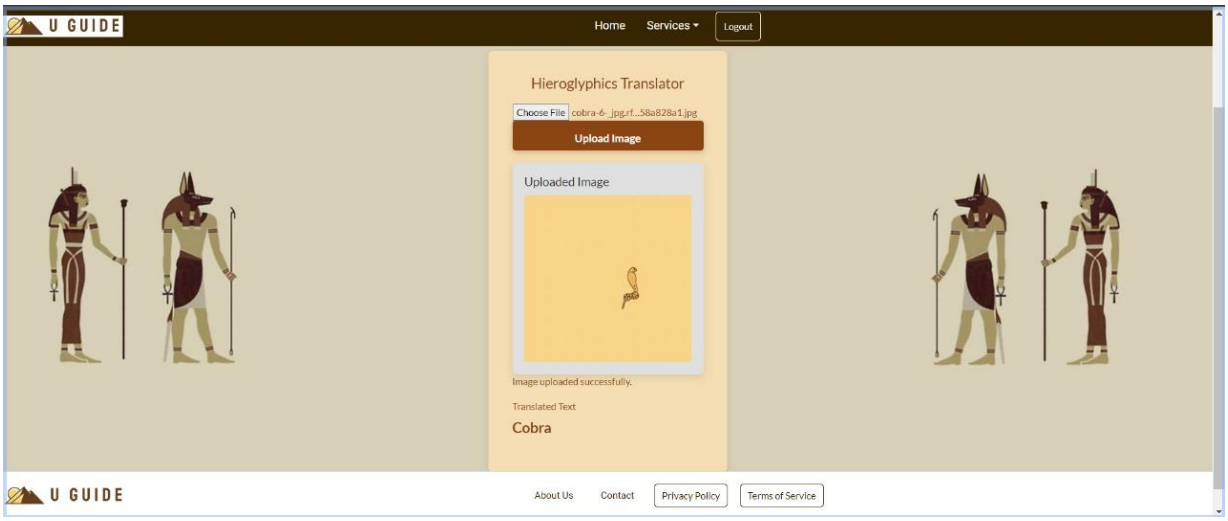




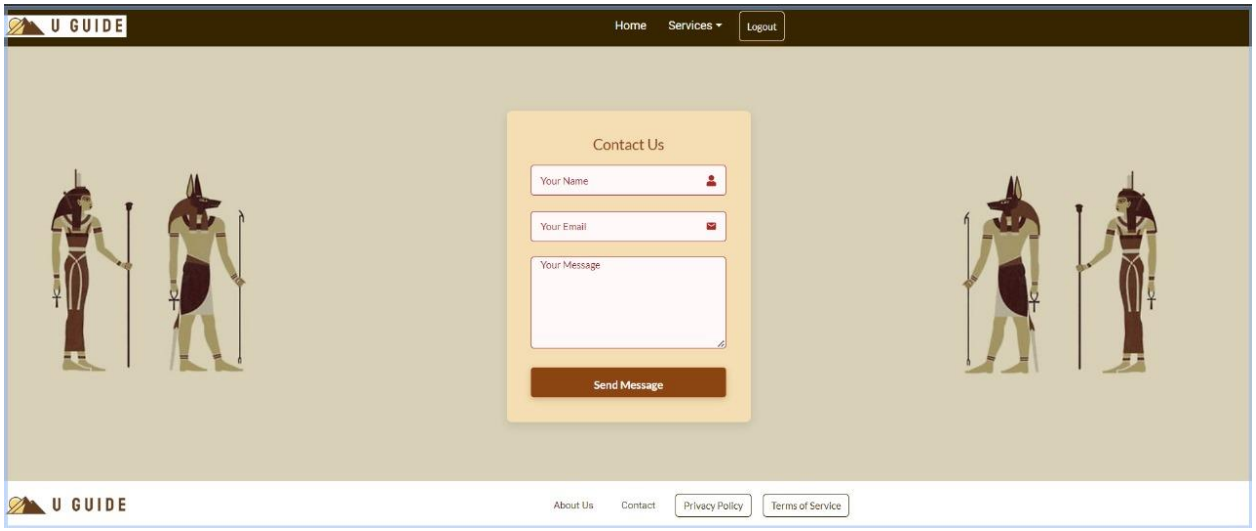
5. U Guide Assistant



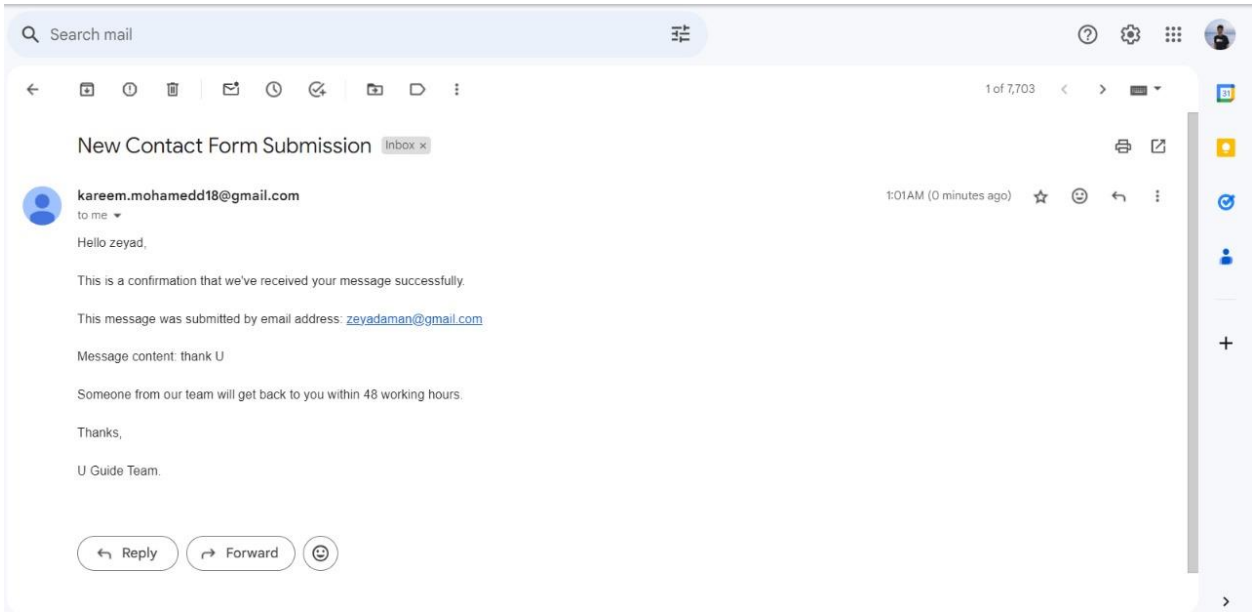
6. Hieroglyphics Translator



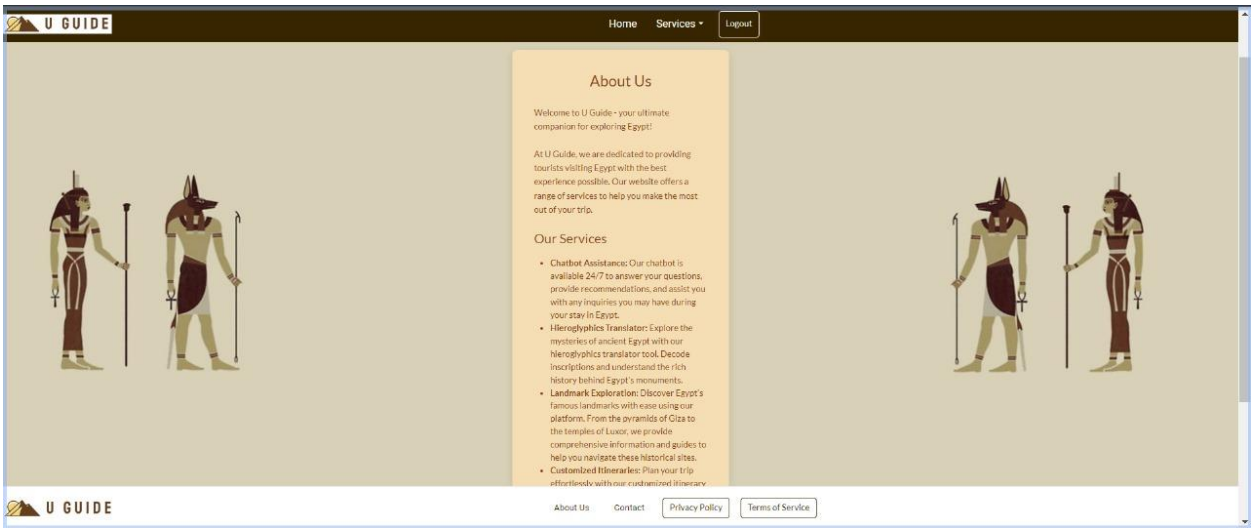
7. Contact us



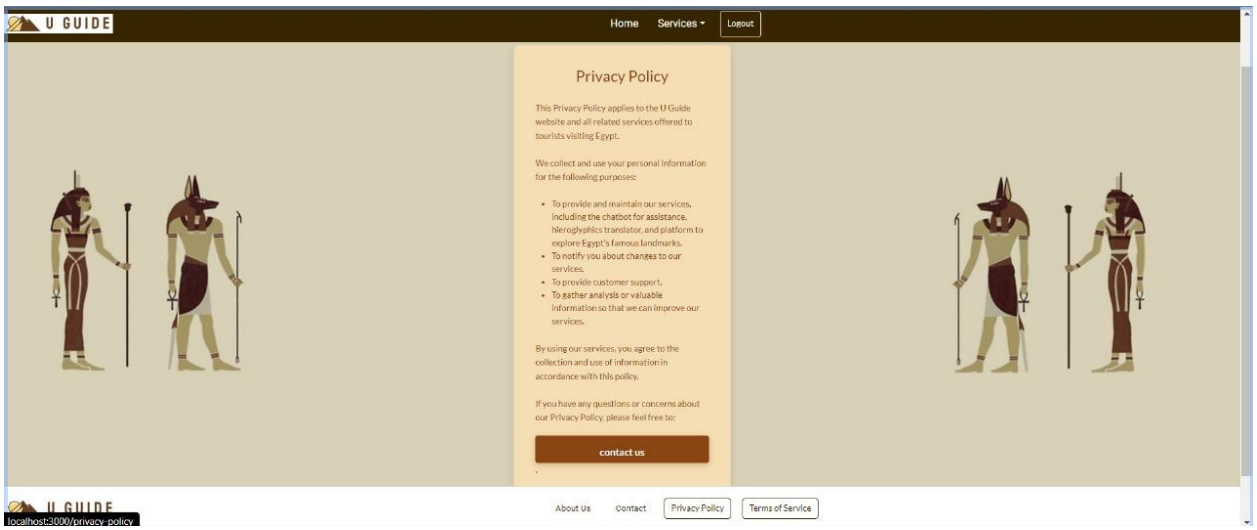
7.1 Receiving messages



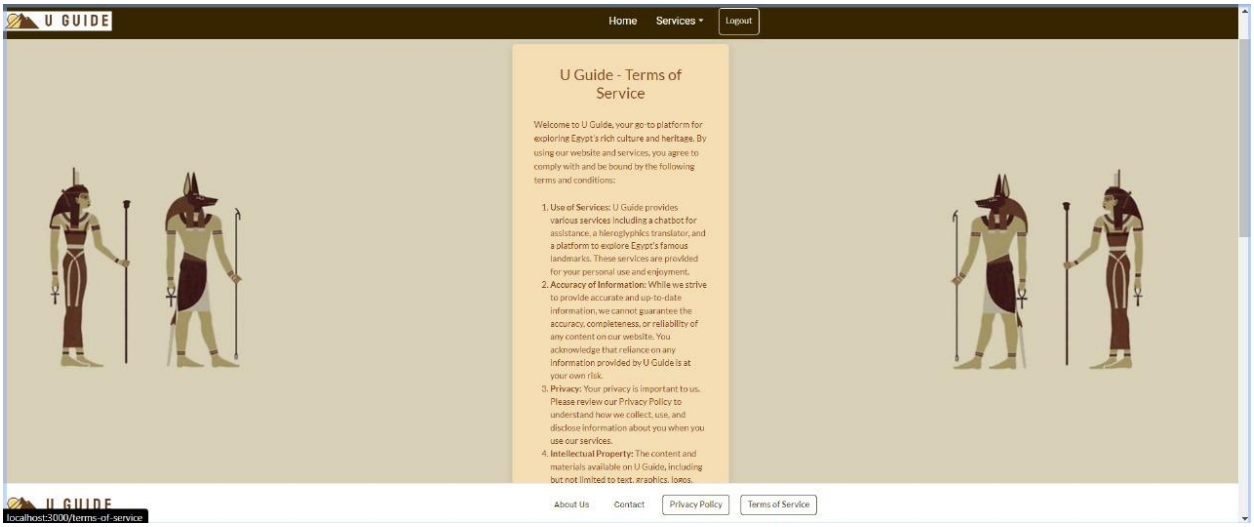
8. About us



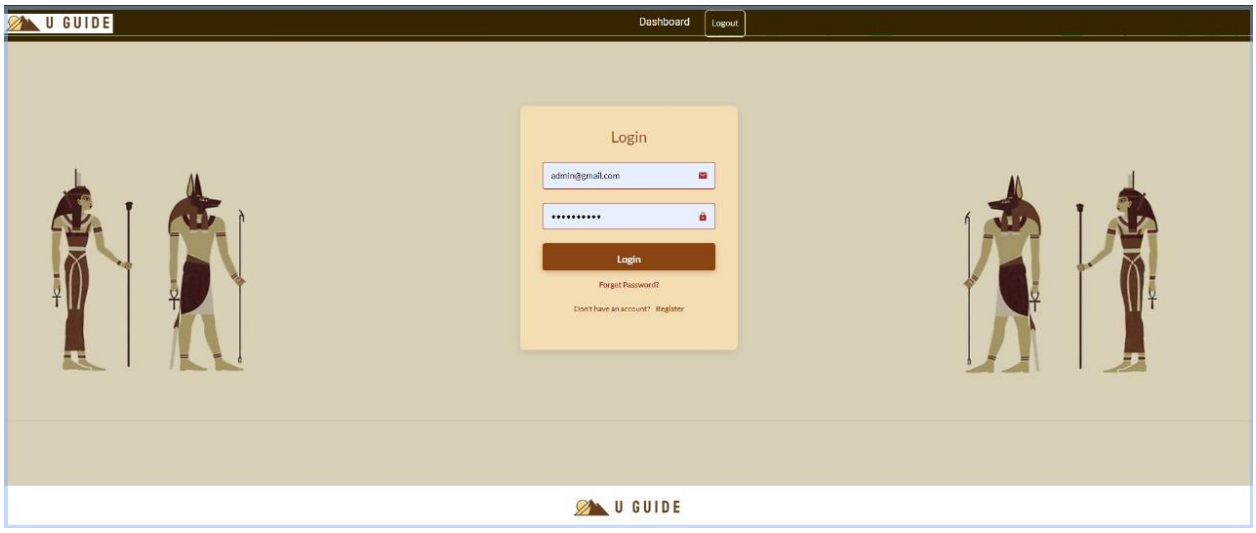
9.Privacy Policy



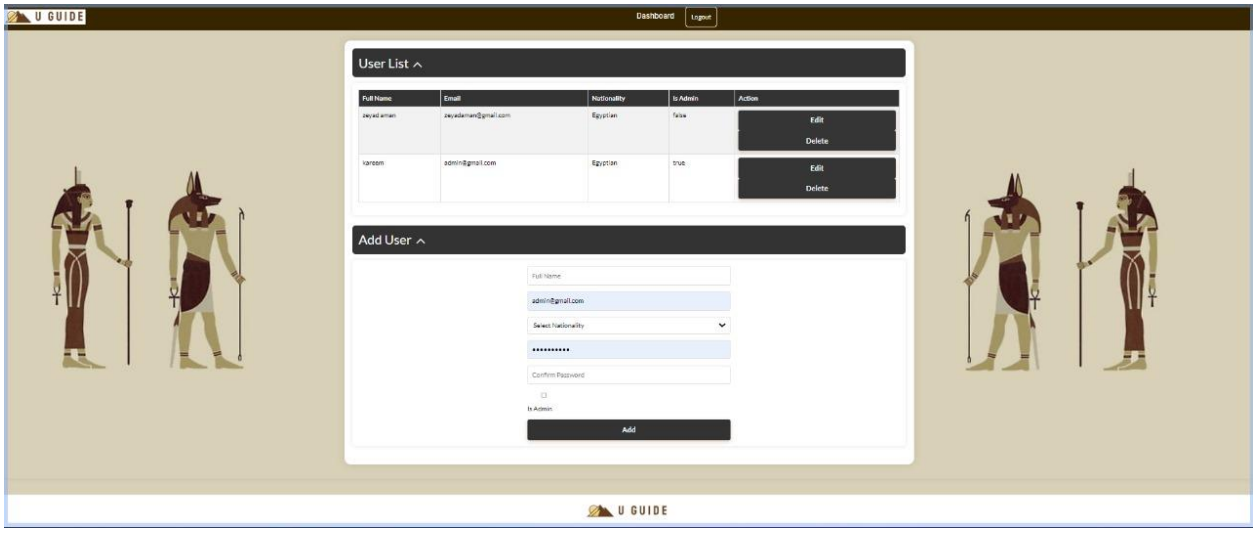
10. Terms of Service



11.Admin login



12.Admin Dashboard



Back-End Implementation:

1.Registration

```
/**
 * @desc    Register New User
 * @route   /api/auth/register
 * @method  POST
 * @access  public
 */
module.exports.register = asyncHandler(async (req, res) => {
  const { error } = validateRegisterUser(req.body);
  if (error) {
    return res.status(400).json({ message: error.details[0].message });
  }
  let user = await User.findOne({ email: req.body.email });
  if (user) {
    return res.status(400).json({ message: "This user is already registered" });
  }
  if (req.body.password !== req.body.confirmPassword) {
    return res.status(400).json({ message: "Passwords do not match" });
  }
  const salt = await bcrypt.genSalt(10);
  req.body.password = await bcrypt.hash(req.body.password, salt);
  req.body.confirmPassword = await bcrypt.hash(req.body.confirmPassword, salt);
  user = new User({
    fullName: req.body.fullName,
    email: req.body.email,
    password: req.body.password,
    confirmPassword: req.body.confirmPassword,
    nationality: req.body.nationality,
    isAdmin: req.body.isAdmin || false
  });
  const result = await user.save();
  const token = jwt.sign({ id: user._id, isAdmin: user.isAdmin }, process.env.JWT_SECRET_KEY, { expiresIn: '24h' });
  const { password, ...other } = result._doc;
  res.status(201).json({ ...other, token });
});
```

2.Login

```
41  /**
42   * @desc    Login User
43   * @route   /api/auth/login
44   * @method  POST
45   * @access  public
46   */
47  module.exports.login = asyncHandler(async (req, res) => {
48    const { error } = validateLoginUser(req.body);
49    if (error) {
50      return res.status(400).json({ message: error.details[0].message });
51    }
52    let user = await User.findOne({ email: req.body.email });
53    if (!user) {
54      return res.status(400).json({ message: "Invalid email or password" });
55    }
56    const isPasswordMatch = await bcrypt.compare(
57      req.body.password,
58      user.password
59    );
60    if (!isPasswordMatch) {
61      return res.status(400).json({ message: "Invalid email or password" });
62    }
63    const token = jwt.sign({ id: user._id, isAdmin: user.isAdmin }, process.env.JWT_SECRET_KEY, { expiresIn: '24h' });
64    const { password, ...other } = user._doc;
65    res.status(200).json({ ...other, token });
66  });
```

3.Forget password

```
/**
 * @desc  Get Forgot Password View
 * @route /password/forgot-password
 * @method GET
 * @access public
 */
module.exports.getForgotPasswordView = asyncHandler((req, res) => {
    res.render('forgot-password');
});
```

```
/**
 * @desc  Send Forgot Password Link
 * @route /password/forgot-password
 * @method POST
 * @access public
 */
module.exports.sendForgotPasswordLink = asyncHandler(async (req, res) => {
    const user = await User.findOne({ email: req.body.email });
    if (!user) {
        return res.status(404).json({ message: "User not found!" });
    }

    const secret = process.env.JWT_SECRET_KEY + user.password;
    const token = jwt.sign({ email: user.email, id: user.id }, secret, {
        expiresIn: '10m'
    });

    const link = `http://localhost:5000/password/reset-password/${user._id}/${token}`;

    const transporter = nodemailer.createTransport({
        service: "gmail",
        auth: {
            user: process.env.USER_EMAIL,
            pass: process.env.USER_PASS,
        }
    });

    const formattedDate = moment().format('MMMM Do YYYY, h:mm:ss a');

    const mailOptions = {
        from: process.env.USER_EMAIL,
        to: user.email,
        subject: "U Guide Password Reset",
        html: `<div>
            <p>Hi ${user.email},</p>
            <p>We've received your request to reset your password on ${formattedDate}.</p>
            <h4>Please click on the link below to reset your password:</h4>
            <p>${link}</p>
            <p>Kind Regards,</p>
            <p>U Guide Team</p>
        </div>`
    };

    transporter.sendMail(mailOptions, function (error, success) {
        if (error) {
            console.log(error);
        } else {
            console.log("Email sent:" + success.response);
        }
    });

    res.render("link-send");
});
```

```
/**
 * @desc  get reset password view
 * @route /password/reset-password/:userid/:token
 * @method GET
 * @access public
 */
module.exports.getResetPasswordView = asyncHandler(async (req, res) => {
    const user = await User.findById(req.params.userId);
    if (!user) {
        return res.status(404).json({ message: "User not found!" });
    }

    const secret = process.env.JWT_SECRET_KEY + user.password;
    try {
        jwt.verify(req.params.token, secret);
        res.render('reset-password', { email: user.email });
    } catch (error) {
        console.log(error);
        res.json({ message: "Error" });
    }

});

/**
 * @desc  get reset password view
 * @route /password/reset-password/:userid/:token
 * @method POST
 * @access public
 */
module.exports.resetThePassword = asyncHandler(async (req, res) => {
    // todo : validation
    const user = await User.findById(req.params.userId);
    if (!user) {
        return res.status(404).json({ message: "user not found" });
    }

    const secret = process.env.JWT_SECRET_KEY + user.password;
    try {
        jwt.verify(req.params.token, secret);
        const salt = await bcrypt.genSalt(10);
        req.body.password = await bcrypt.hash(req.body.password, salt);
        user.password = req.body.password;
        await user.save();
        res.render('success-password');
    } catch (error) {
        console.log(error);
        res.json({ message: "Error" });
    }
});
```

Hieroglyphics Translator

app.js upload.js Users.js .env

routes > upload.js > router.post("/") callback > PyResponse

```
const express = require("express");
const router = express.Router();
const multer = require("multer");
const path = require("path");
const FormData = require("form-data");
const fs = require('fs');
const axios = require("axios"); // Added axios import

const storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, path.join(__dirname, "../images"));
  },
  filename: function (req, file, cb) {
    cb(null, new Date().toISOString().replace(/:/g, "-") + file.originalname);
  }
});

const upload = multer({
  storage,
  fileFilter: function (req, file, cb) {
    if (!file.originalname.match(/\.(jpg|jpeg|png|gif)$/)) {
      return cb(new Error('Only image files are allowed!'));
    }
    cb(null, true);
  }
});

// /api/upload
router.post("/", upload.single("image"), async (req, res) => { // Added async keyword
  try {
    if (!req.file) {
      // If no file is received in the request
      return res.status(400).json({ message: "No file uploaded" });
    }

    // If file upload is successful
    const form = new FormData();
    form.append('image', fs.createReadStream(req.file.path)); // Fixed the file path
    const PyResponse = await axios.post("http://127.0.0.1:8000/upload-image", form); // Corrected the URL
    // Send response
    res.status(200).json({ message: "Image uploaded", response: PyResponse.data });
  } catch (error) {
    // If an error occurs during the upload process
    console.error("Error uploading image:", error.message);
    res.status(500).json({ message: error.message });
  }
});

module.exports = router;
```

Contact Us

```
app.js  upload.js  contactController.js  .env
controllers > contactController.js > ...
const nodemailer = require('nodemailer');
const Contact = require('../models/contact');

const transporter = nodemailer.createTransport({
    service: 'gmail',
    auth: {
        user: 'kareem.mohamedd18@gmail.com',
        pass: 'mjux izhg phwd dnvy'
    }
});

const sendContactMessage = async (req, res) => {
    try {
        const { name, email, message } = req.body;

        console.log('Received contact form submission:');
        console.log('Name:', name);
        console.log('Email:', email);
        console.log('Message:', message);

        // Save the contact message to the database
        const newContactMessage = new Contact({ name, email, message });
        await newContactMessage.save();

        await transporter.sendMail({
            from: 'kareem.mohamedd18@gmail.com',
            to: email,
            subject: 'New Contact Form Submission',
            text: `Hello ${name}, \n\nThis is a confirmation that we've received your message successfully. \n\nThis message was submitted by email address: ${email}\n\nMessage content: ${message} \n\nSomeone from our team will ge
        });

        res.status(200).json({ message: "Message received successfully!" });
    } catch (error) {
        console.error('Error processing contact form submission:', error);
        res.status(500).json({ message: "Internal server error" });
    }
};

module.exports = { sendContactMessage };
```

Hieroglyphics Translator Model

```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import cv2
import seaborn as sns
import random

from sklearn.metrics import classification_report, confusion_matrix,average_precision_score

from sklearn.preprocessing import OrdinalEncoder
from tensorflow.keras.layers import Input, Dense, GlobalAveragePooling2D,BatchNormalization, Activation,Dropout, Conv2D, MaxPooling2D, Flatten
from tensorflow.keras.callbacks import EarlyStopping
tf.random.set_seed(10)



train_data = pd.read_csv('C:\\Users\\DELL\\Desktop\\pp\\pp\\archive (2)\\train\\_annotations.csv')
val_data = pd.read_csv('C:\\Users\\DELL\\Desktop\\pp\\pp\\archive (2)\\valid\\_annotations.csv')
test_data = pd.read_csv('C:\\Users\\DELL\\Desktop\\pp\\pp\\archive (2)\\test\\_annotations.csv')
```

```
oe = OrdinalEncoder()

# Encoding class labels in the train set
main_path_train = 'C:\\Users\\DELL\\Desktop\\pp\\pp\\archive (2)\\train/'
train_data['filename'] = train_data['filename'].apply(lambda x: main_path_train + x)

oe.fit(np.array(train_data['class']).reshape(-1,1))
train_data['class'] = oe.transform(np.array(train_data['class']).reshape(-1,1))

# Encoding class labels in the validation set
main_path_val = 'C:\\Users\\DELL\\Desktop\\pp\\pp\\archive (2)\\valid/'
val_data['filename'] = val_data['filename'].apply(lambda x: main_path_val + x)
val_data['class'] = oe.transform(np.array(val_data['class']).reshape(-1,1))


# Encoding class labels in the test set
main_path_test = 'C:\\Users\\DELL\\Desktop\\pp\\pp\\archive (2)\\test/'
test_data['filename'] = test_data['filename'].apply(lambda x: main_path_test + x)
test_data['class'] = oe.transform(np.array(test_data['class']).reshape(-1,1))
```

DenseNet Model

```
pretrained_model = tf.keras.applications.DenseNet169(weights='imagenet',include_top=False ,input_shape=[250,250, 3])
pretrained_model.trainable = True

model = tf.keras.Sequential([
    pretrained_model,
    GlobalAveragePooling2D(),

    Dense(256),
    BatchNormalization(),
    Activation('relu'),
    Dropout(0.1),

    Dense(128),
    BatchNormalization(),
    Activation('relu'),
    Dropout(0.1),

    Dense(len(oe.categories_[0]), activation = 'softmax') #nclasses = 95

])

model.compile(optimizer  = tf.keras.optimizers.Adam(learning_rate = 0.00001), loss = 'sparse_categorical_crossentropy', metrics = ['accuracy'])

model.summary()
```

Evaluation

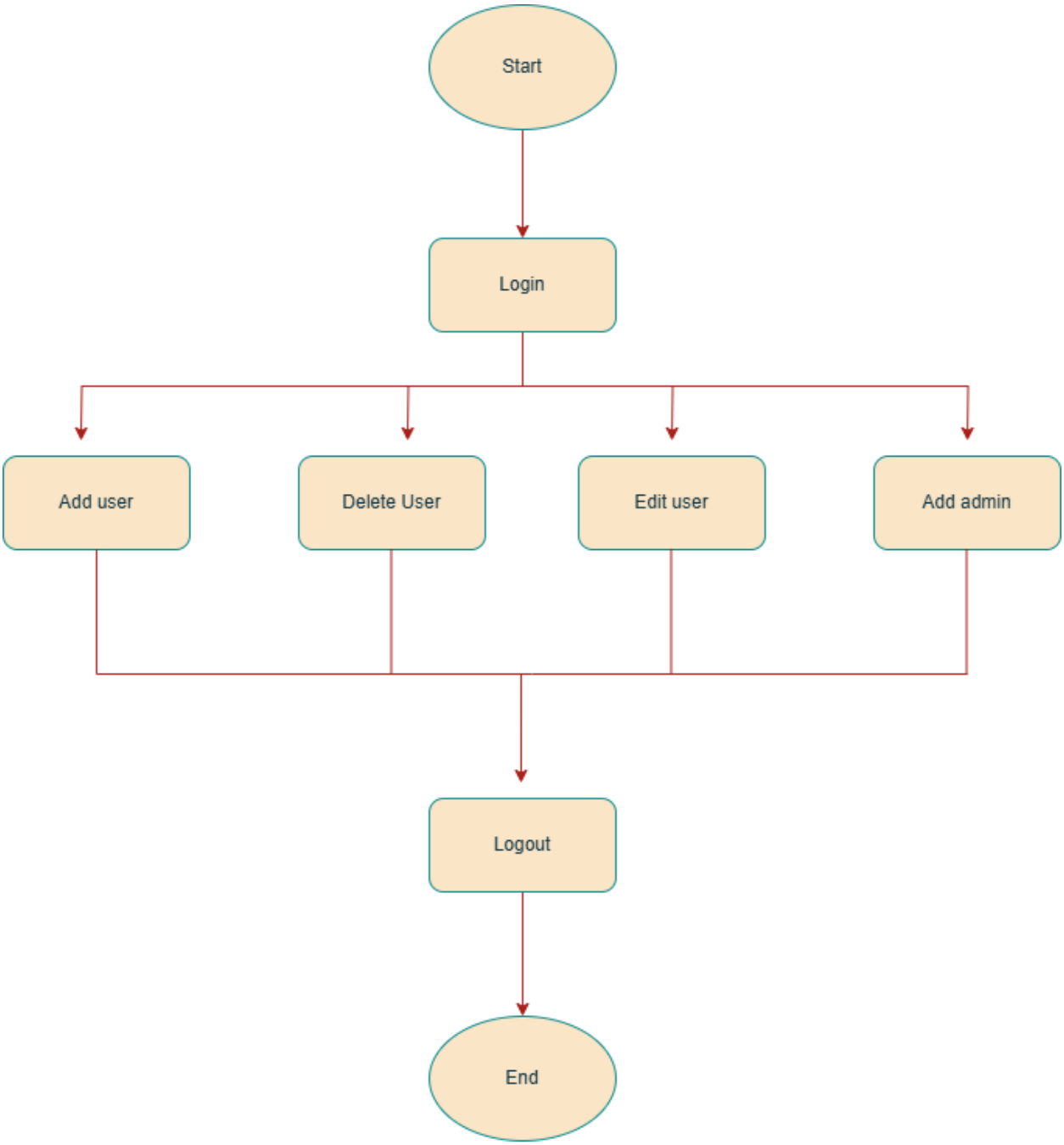
```
test_eval = model.evaluate(test_dataset)
print('test accuracy : {0:.3f} %'.format(test_eval[1]*100))
```

Prediction

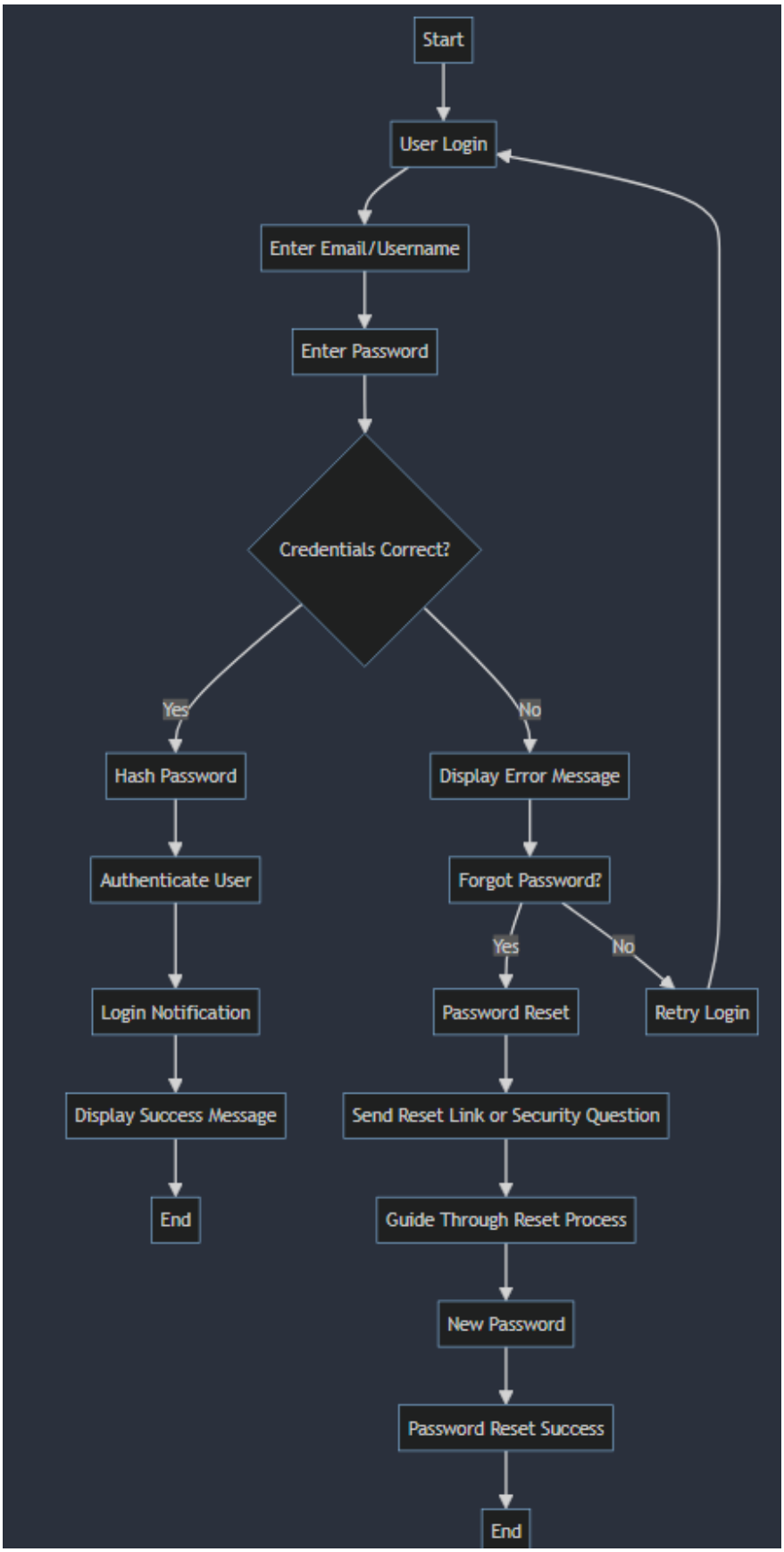
```
pred = model.predict(test_dataset)
pred_ = np.argmax(pred, axis = 1)



# Classification report

report = classification_report(np.array(test_data['class']), pred_)
print(report)
```

4.2 Pseudocode, Flowchart or workflow

1.Admin Flowchart



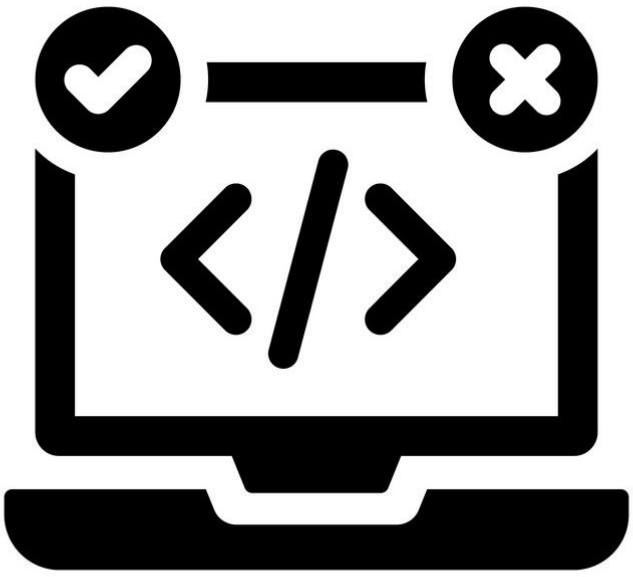
2.User Flowchart



Chapter 6

Testing

In this chapter we will talk about the testing stages we have done to our project to ensure that our project was functioning as we need, the system is free of bugs and errors and that it achieved the desired level of quality and the defined specifications.



6.1 Functional Testing

6.1.1 Unit testing

We have done test cases during the development process to test each individual unit in isolation from other units such as modules, classes, programs, objects, functions, etc. to ensure that these individual units are working as expected.

6.1.2 Integration testing

In this level of testing, we have done testing on groups of these modules to make sure that they work together and interact properly, validate how the software behaves when a transfer of data or control occurs and that the software components and system interface work together.

6.1.3 System testing

In system testing we validated the complete and fully integrated software product, tested the whole behavior of the software as a complete entity.

6.2 Non-Functional Testing

6.2.1 Performance testing

In performance testing we evaluated the speed, responsiveness and stability of a computer, network, software program or device under a workload.

6.2.2 Load testing

We tested the system performance when the workloads increase, workload could mean concurrent users or transactions.

6.2.3 Security testing

We tested how the system behaves against unauthorized internal or external access.

Chapter 7
Results

In this chapter we will discuss the expected results for our project and the actual results and why our actual result failed to be the expected results.

6.1 Results

6.1.1 Expected results.

 Registration:

- User completes registration form with all required information.
- Clear instructions and validation messages guide the user.
- User receives a confirmation message upon successful registration.

 Login as User:

- User logs in with email address or username (if applicable).
- Secure password hashing ensures user authentication.
- Login notification confirms successful access.
- Password reset functionality allows recovery with clear instructions.

 Login as Admin (Separate from User Login):

- Admin logs in with email address and password.
- Secure password hashing ensures separate admin authentication.

 Admin Dashboard Access:

- View a list of all registered users (name, email, registration date).
- Add, delete, or update user accounts with confirmation prompts.
- Create or delete additional admin accounts with security measures (like 2FA).

 Informative Content for Users:

- Click cards/flip cards to access detailed descriptions of tourist places.
- View photos to discover popular tourist destinations through a swipeable gallery.

6.1.2 Actual results

 Registration: Streamline user account creation with clear instructions and validation for a smooth onboarding process.
 Login: Provide secure and user-friendly login options (email/username) with password reset functionality for forgotten credentials.
 User Interface (UI) Design:

- Informative cards or flip cards should visually present detailed descriptions of tourist places.
- A swipe able photo gallery should allow users to discover popular tourist destinations in an engaging way.

 Admin Dashboard: Empower admins with functionalities to:

- View a comprehensive list of registered users.
- Manage user accounts through adding, deleting, or updating them.
- Create or delete additional admin accounts with robust security measures.

6.2 Discussion

As a conclusion for the previous points, we have managed to meet most of the expectations we planned for except for some points such as security questions for registration, also saving the registration time of the user in the system.

And due to shortage of time and limitations, we couldn't make a real time scan for the Hieroglyphics translator that can instantly detect the symbols and translate them instead, we provided uploading saved images from your device to help you translate it.

Chapter 7

Conclusion

This chapter will include a summary of our report and our recommendations on how to enhance our project.

8.1 Conclusion:

Finally, here we are writing the very last words and putting the last lines of our story and adventure at FCAI-HU, working in this project was really different from any other projects we worked on during the last 4 years, this project was full of the feeling of responsibility towards our society and ourselves, that is why we try to do our best in it, regardless of the poor resources we had and the limited time.

We believe this project is a tribute to all our professors and teacher assistance and to our country, attempting to improve tourism in Egypt through what we learned and learned was a genuine challenge, a great challenge that was worth trying.

We hope that U Guide will play its role in society as we are expecting.

We are looking forward to being the first local solution to improve tourism more and more and let the world know our heritage and the heritage of our ancestors.

8.2 Recommendations to enhance our project:

The application is still under enhancements, The performance of the application is continuously improved to meet the expectations of the users and provide the best possible experience.

The user interface will always be in continuous enhancement and modification to always match the user's expectations and reach the highest possible level of being a user-friendly mobile application, also we will try to improve the existing features and add new features.

Chapter 8

Future Work

In this chapter we will introduce our future to improve our application as we are looking forward to extending it and adding new smart features to it, so our journey will be continued.

8.1 Future Plans

Expanding on existing chatbot functionality, you can integrate advanced natural language processing (NLP) capabilities to provide users with more personalized and intelligent assistance.

1. Customer Service AI Call assist: There will be a dedicated number provided to call for more assistance from our AI chatbot in Tourism that can process Natural language to support a variety of people specially blind people and others with sight invariabilities.

2. Multi-Language Support: In addition to translating hieroglyphics, the chatbot can support multiple languages to cater to a diverse range of users. Whether someone speaks English, Arabic, French, or any other language, they can interact with the chatbot in their preferred language for a seamless experience.

3. Personalized Recommendations: By analyzing user preferences, browsing history, and demographic information, the chatbot can offer personalized travel recommendations tailored to everyone. For example, it can suggest specific tour packages, activities, or cultural experiences based on the user's interests, budget, and travel dates.

4. Real-Time Assistance: Implementing real-time assistance features, such as live chat support with human agents, can enhance the chatbot's capabilities. When the chatbot encounters complex inquiries or situations that require human intervention, it can seamlessly transfer the conversation to a human agent for further assistance while maintaining continuity.

By enhancing the chatbot with advanced NLP capabilities, multi-language support, personalized recommendations, and real-time assistance features, you can provide users with a more efficient, engaging, and tailored travel experience tailored to their needs and preferences.