`

School of Engineering and Applied Science,
Ahmedabad University

CSE300 Software Engineering
Design Documentation Report

Guided By: Prof Khusru Doctor
TA Muskan Matwani
TA Anupama Nair

TRAVOYAGER

We are here to get you there

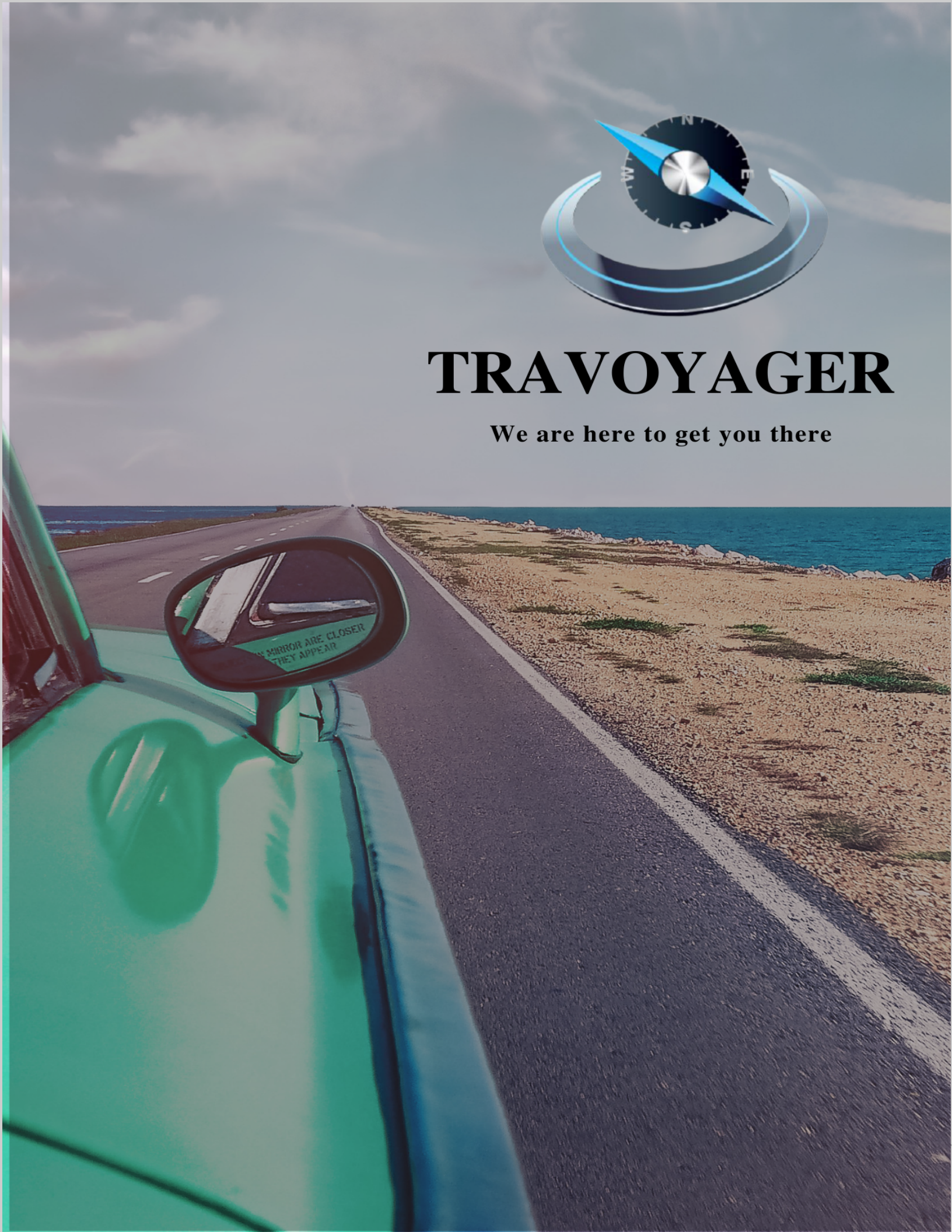


Table of Contents

1. Introduction

1.1. Purpose

It provides the details about how the system is designed in terms of user interface and user experience of the system. The design document should be able to completely explain the flow of the system and provide the understanding of what and how to build the system. By looking at the design document any person should be able to get an idea of how the system will function.

1.2. Identification

The system can be identified as developed using the django framework in the visual studio code environment. The database design is done completely in the django framework and the interface design is done using the HTML, CSS and Javascript coding language.

1.3. Intended Audience and Scope

This document is generated for the set of people who are fond of travelling and make use of travelling applications for their booking and travel purposes. The target set of audiences mainly includes people of any age group who wish to travel from their native place to another place and need help in developing a plan for travelling and visiting different places including their stay details.

It also includes the designing team, the developer team, the product owner, the stakeholders, the testing team, and all the clients developing or using the system. The client/user can utilise this document for verifying the flow of the designed system. The product owner can utilise this document to plan the flow of the system and can suggest the required modifications to the design team and can guide the developer team to build the system accordingly.

The design team can utilise this document to rebase the design of the system and make the design more modular and user friendly. The developer team can utilise this document as a foundation pillar for development of the system and can further link the requirements and design document to build the product to meet the expectations of the user. The testing team can utilise this document to design the test cases for every user interface design on the frontend and the backend side and check if the tests are done correctly or not.

1.4. Ideal stakeholders

The ideal stakeholders of the system can be the set of people who like to invest into travel related products like:

- Hodophiles
    - People who love to travel to different places and destinations.
- Itinerary generating sites
    - Site development companies or institutions who work on producing travel itineraries for their users.
- Travel agencies
    - The travel agencies who wish to make available online a set of itineraries to their users so that they can access it from their website also.

1.5. Tools and Technologies

Tools

- VS code
- SQLite
- draw.io
- Visual Paradigm
- Lucid chart
- Figma

Technologies

- Python(Django framework)
- HTML
- CSS
- Javascript

2. General Overview of the system

2.1. Overview

Currently for people who love travelling; planning for a trip can be a tedious task and one has to plan for the destination, mode of travel, hotel accommodation, what places to visit, how many days to spend etc. all in a decided budget. People may require to flip through numerous pages of brochures of travel agencies looking for the perfect plan for their trip. Many times the travel agencies packages seem costly to many people due to which they change their plans of travelling and decide to go on their own. Sometimes travelling on one's own self can be a difficult task for finding hotels, restaurants, places to visit etc.

Travyoger is a Travel recommendation system, which recomandate the best suitable places to visit in the destination city. The user can generate itineraries and parelley book the hotel. This will reduce unnecessary searches and further steps of booking the hotel.

2.2. Current scenario

As we go for a trip for relaxation purposes, the planning for that also needs to be pleasant. First searching for the best destination and then fetching the best place out of it is a very tedious task. Plus after reaching the destination, the user needs to find the suitable hotel to stay. So pre-booking is also needed. For this small task, users have to visit more than 4 or 5 sites and again there can be chances of unwanted failures.

2.3. Need for a solution

This need explains why the system is being developed, what purpose it serves, and why it is necessary. The system helps users to get all the details from one portal. The pre booking for the Hotel also can be done from the same site. This is a kind of organized way to manage the whole travel plan. The ease of use can also help users to have pleasant experiences.

2.4. Proposed System

Each system has some pros and cons. So by defining them, we can keep track of the limitations and future work related to the system.

2.4.1. Constraints

- This system supports only English language, so it doesn't have multi language support. The good internet connection is required because the server fetches the data from the database.
- The system can't generate more than 2 itinerary at same time for the same user. Means the user has to complete one itinerary before the generation of others.
- The trip starts from the same day of starting date, meaning the user can book from tomorrow's data till he wants.
- There are only 3 slots available for each day of the trip. There is no place that takes time to visit for more than 2 slots.
- The hotel room selection is not given to the user. The update of wallet balance through net banking is still pending.

2.4.2. Assumption

- The travel can happen from city to city only.
- The user can travel by his own from source to destination location and also to visit the places. The user has already reached the destination before the starting date of his/her trip
- The places are not sorted based on location.
- The user can afford all the travel expenses.
- Room capacity and quality of all the rooms of the hotel are the same. User hasn’t flexibility to choose the room at any specific location.
- The wallet balance update is pending.

2.4.3. Dependencies

The main dependencies are between:

- Itinerary generation and place fetch
- Hotel booking and Payment
- Completed trips and giving place review
- Place ratings and user given ratings
- Destination and place database

- Destination and hotel database
- User inputs and itinerary database
- Hotel and hotel booking database.

3. Design Considerations

3.1. Goals and Guidelines

The following goals were aimed to be fulfilled by the system proposed to solve the problem at hand.

1. **Architecture:**
    - The proposed solution to the problem must satisfy all the functional needs stated by the user, and is required to be developed considering all the non-functional requirements. It is supposed to be adaptive to further changes like support to additional features, functionalities and use cases.
2. **Development Environment:**
    - The application is made using the latest tools and technologies with backward compatibility with the older versions of those tools, and is consistent with the development environment.
3. **Extensibility:**
    - The system design is extensible, i.e. new features can be added to the application and it is designed to be compatible with its older versions.
4. **Ease of Use:**
    - The features and the entire functionality of the developing application must be easy to use and at the same time provide a strong and reliable user experience. New characteristics cannot impact the existing functionality from a user perspective.

3.2. Development Environment

- Django Framework - For developing the system and connecting backend and frontend
- HTML Development - For developing the frontend design
- CSS Stylesheets - For designing the frontend
- Figma - For preparing the UI of the frontend
- Git version control - For maintaining version of the system

- GitHub repository - For maintaining the changes in the system
- SQLite3 - For the database

3.3. Operational Goals

Functional goals of the proposed system include:

- Accurate functionality of the web application
- Coordinating data via a database model
- Ability to create the best possible itinerary based on given details
- Completing the one-click deployment model

3.4. Development Methodology

1. **Scalability**
    - Ensure that the architecture can be scaled horizontally, across multiple servers and across multiple regions. That means that once your traffic goes up, you should be able to add and remove new servers as the solution requires.
2. **Availability**
    - The architecture should support a high availability environment. Infrastructure redundancy is required. This ensures the solution is available if multiple servers or an entire data center fail. The current availability of the solution per the hosting providers service level agreement is 99.999% availability.
3. **Security**
    - Solution architecture should expose only the minimal amount of code possible. Most of the back-end pieces should be hidden away. In addition to that, security of each system should be multi-layered.
4. **Extensibility**
    - Architecture must be able to swap out modules, change layers, and add pieces to the application without having to worry about the underlying data contracts in place
5. **Separation of responsibility**
    - System should be modular enough that each piece of code has a set of responsibilities and not more. The back-end should not create front end code nor should the front-end code include business logic
6. **Restful Framework**
    - The reason for a Restful API is plain and simple flexibility. Framework does not want to be tied or dependent on a specific programming language and architecture. Architecture needs to be able to replace each layer independently and even use different languages that might be better suited for a certain layer
7. **Functionality**
    - The software is capable to provide functions which meet stated and implied needs when the software is used under specified conditions (what the software does to fulfil needs)
8. **Reliability**

- The software is capable to maintain its level of performance under stated conditions for a stated period of time

9. **Usability**
    - The software is capable to be understood, learned, used and attractive to the user, when used under specified conditions (the effort needed for use)
10. **Efficient**
    - The software is capable to provide appropriate performance, relative to the amount of resources used, under stated conditions
11. **Maintainability**
    - The software is capable of being modified. Modifications may include corrections, improvements or adaptations of the software to changes in the environment and in the requirements and functional specifications (the effort needed to be modified)
12. **Portability**
    - The software is capable of being transferred from one environment to another. The environment may include organizational, hardware or software environment

3.5. Architectural Strategies

3.5.1. Mobility

The term mobility has gained much importance in the current world. It has changed the way corporations do their businesses. Information technology is available to us with much better precision and accuracy than before. The term mobility and cloud together can make our life easier and faster in terms of development and deployment of the application.

3.5.2. Social

Social media has profoundly affected the way people work. Users can exchange information with various sources, and accountability in real-time. We need to have resources in a non-cloud setting based on a theoretical maximum peak conjecture. This may result in periods of idle expensive resources or in instances of insufficient efficiency.

3.5.3. Scalability

Scalability of the system should be in terms of horizontal as well as vertical scaling across different web-browsers and devices. So, as the new users get into the system, the system should be able to handle that much traffic with care. If the system cannot handle it, then it should report immediately to the administration for providing the solution.

3.5.4. Platform

The platform of the project should comprise tools for software creation, network access, application servers, database management, business service buses, analytics, etc. As a result, the existing program uses a network. This system has used:

- Python environment along with Django framework
- HTML, CSS and Javascript
- Figma
- SQLite database

3.6. Performance Engineering

Performance engineers may work as part of a team that aims to solve a specific problem; some people in this profession may work in an office on-site, while others may be able to interact.
The performance measures taken into consideration for this system are:

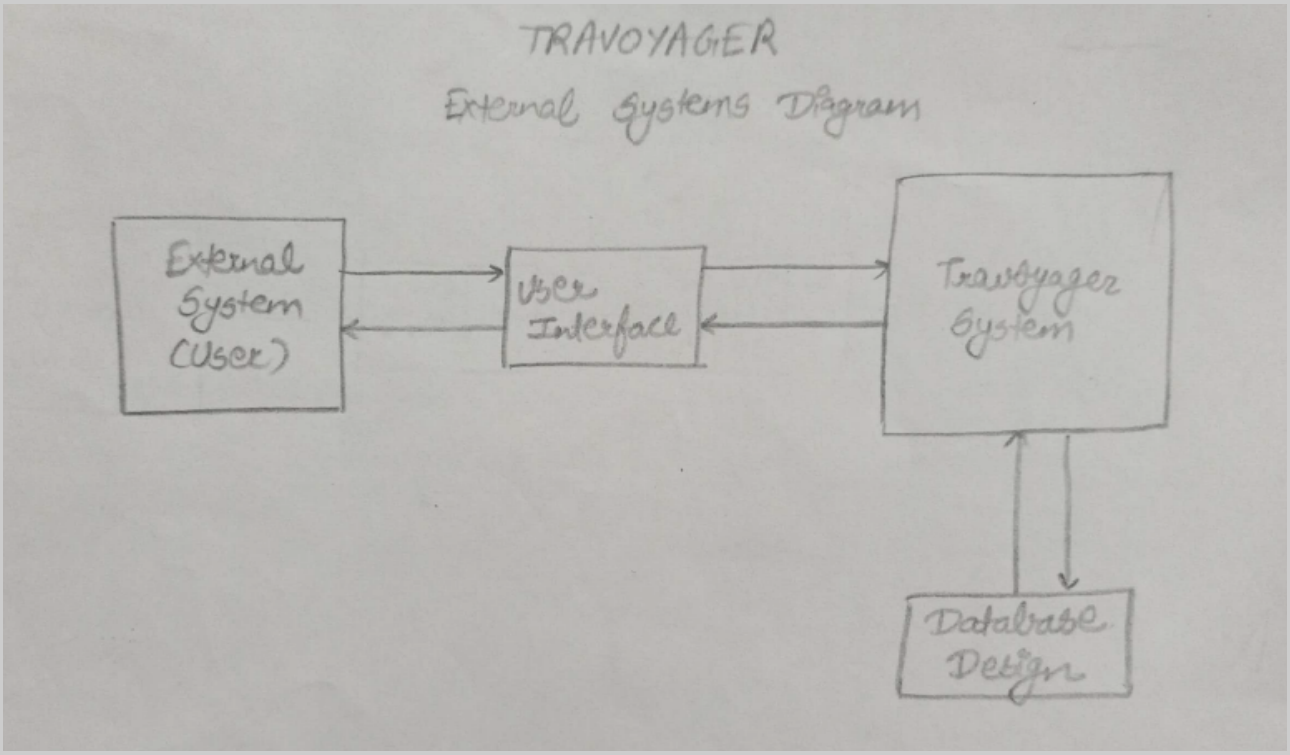
- Providing access to users to generate upto 2 new itineraries corresponding to 2 new trips until the previous ones are completed.
- Providing view access to users for viewing all the trip history after login.
- Providing booking confirmation details along with payment options to confirm the trip bookings.

4. Architecture Design

4.1. System Architecture

The system architecture can be explained in different ways such as the external representation of the system and the internal representation of the system.

4.1.1. External System Architecture

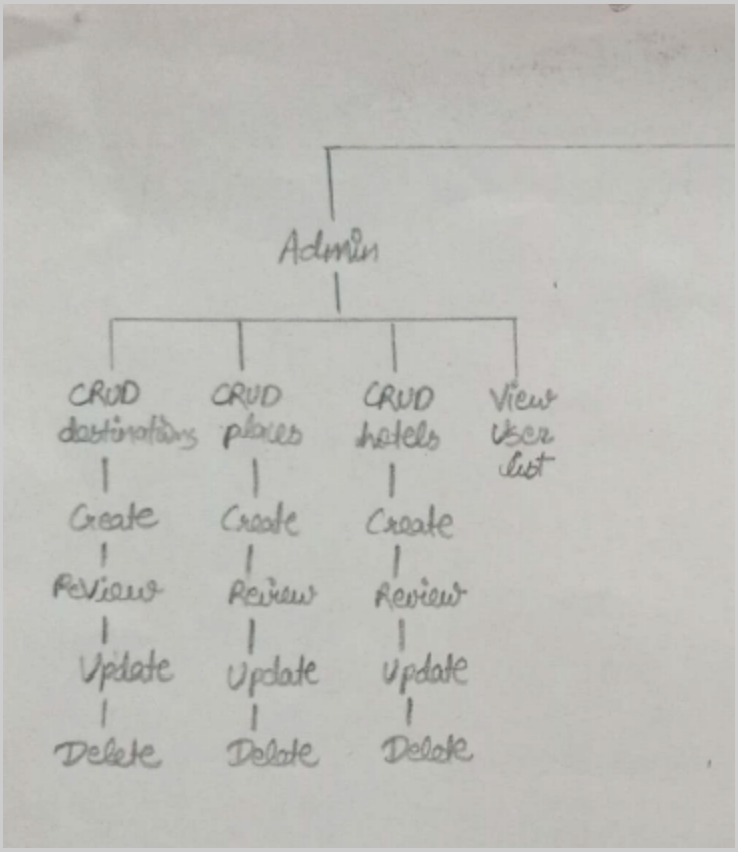


The architecture describes the working of the system when it comes in contact with the external system such as different types of users and how the external factors affect the system. This diagram can help to understand the scope of the project in terms of external requirements.

4.1.2. Website Architecture

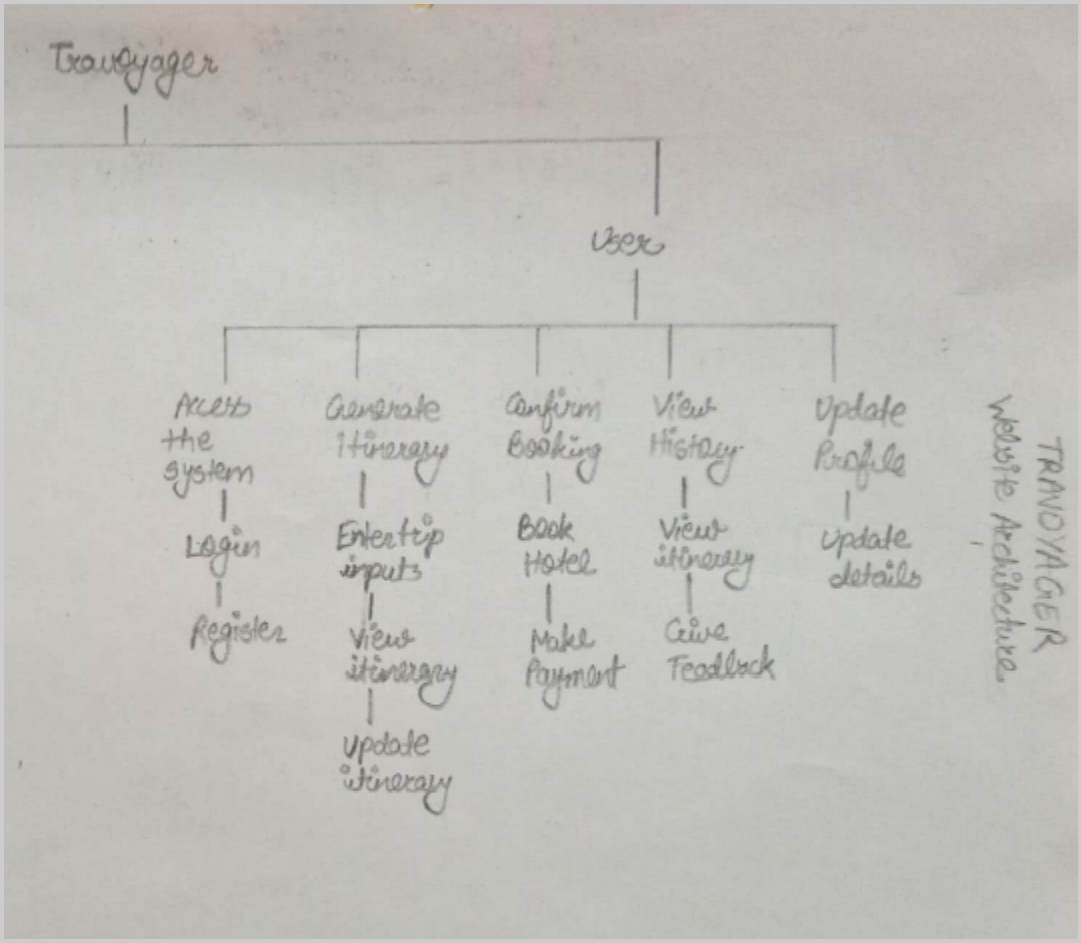
This architecture is described in a hierarchical format for deciding the internal structure representation of the system. It shows how the system will flow in the form of a website. The hierarchy represents the flow of the pages.

4.1.2.1. Admin Side



The admin side mainly consists of performing the CRUD operations which involves creating, reviewing, updating and deleting entities such as destinations and its places and its hotels. It also includes viewing the user profiles registered into the system.

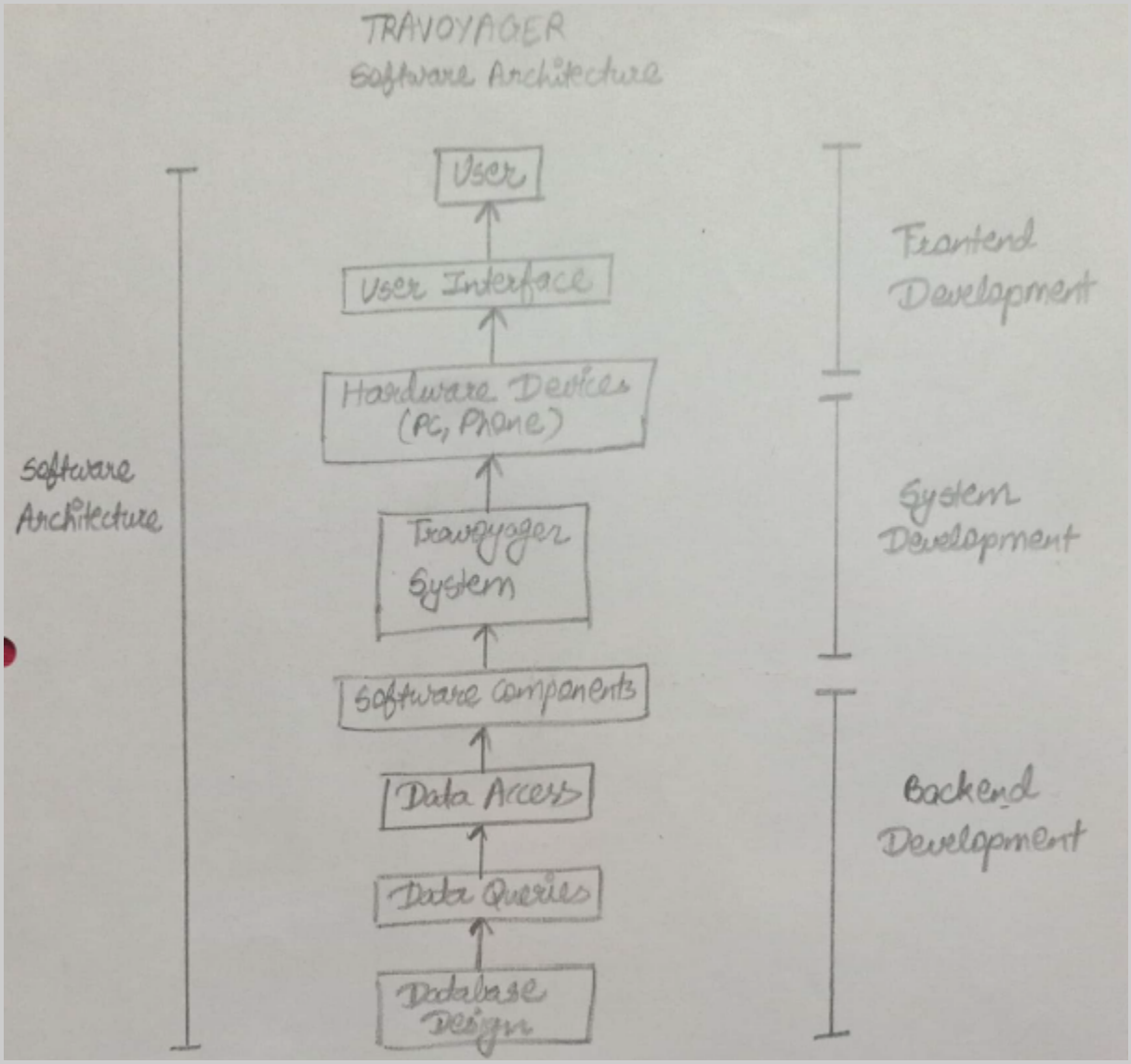
4.1.2.2. User Side



The user side describes the main flow of the system and how the user can utilise the system to get the desired output. It includes accessing the system, generating itinerary, confirm booking, viewing history, updating the profile etc.

4.2. Software Architecture

4.2.1. Software Architecture



The software architecture diagram can to visualise the overall system and can help the developer team know how the system will interact with the end users. The architecture design can be used to find patterns with different software and implement a similar architecture. Here, the architecture consists of connection of backend, system and the frontend development all together.

4.2.2. Software Elements

Name	Used for	Description
Python	Developing Environment	It can be used for other types of programming and software development besides web development. That includes back end development, software development, data science and writing system scripts among other things.
Django	Backend Development	Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel.
MySQL	Data Design	MySQL is a relational database management system based on SQL – Structured Query Language. The application is used for a wide range of purposes, including data warehousing, e-commerce, and logging applications. The most common use for mySQL however, is for the purpose of a web database.
HTML, CSS, JS	Frontend Development	HTML provides the basic structure of sites, which is enhanced and modified by other technologies like CSS and JavaScript. CSS is used to control presentation, formatting, and layout. JavaScript is used to control the behavior of different elements.
Figma	UI Design	Figma is a web-based graphics editing and user interface design app. You can use it to do all kinds of graphic design work from wireframing websites, designing mobile app interfaces, prototyping designs, crafting social media posts, and everything in between. Figma is different from other graphics editing tools

4.3. System Design

The system design gives an idea about the components required to build the system and further maintain the system. It describes how different components can be added to the system and provides security for maintaining the data privacy of the users.

4.3.1. Business Requirements

4.3.1.1. Accessibility

The system should be easily accessible by the users and the stakeholders. There should not be any kind of difficulty in accessing the system.

4.3.1.2. Data Privacy

The information about the users should be protected by providing authentication in form of signin and signup options in the system. This data should not be publicly visible to its users.

4.3.1.3. System Updates

The system should be updated from time to time by the developer team in order to solve the issues and the bugs present in the system to maintain proper flow of the system for its users.

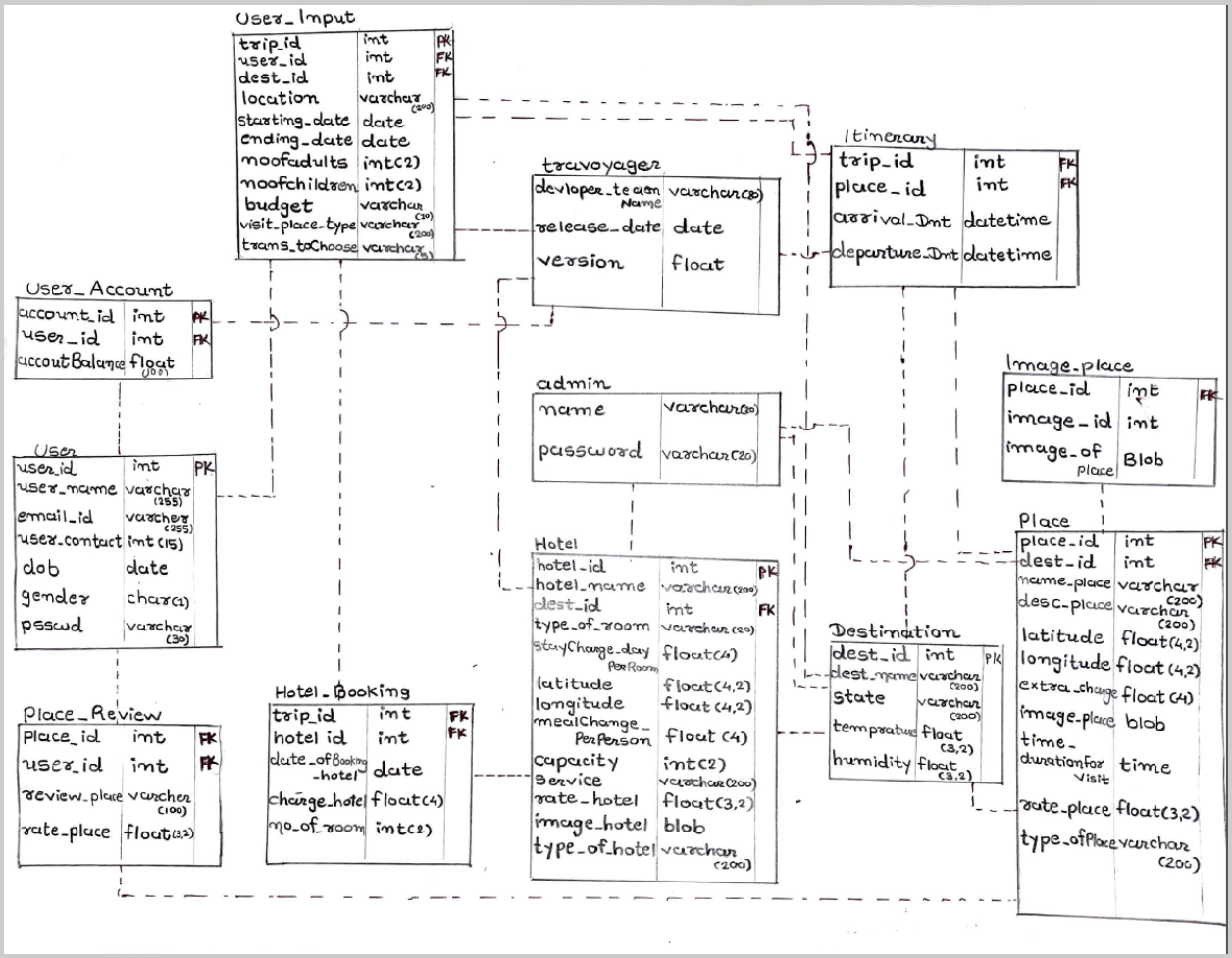
4.3.1.4. Integration

The stakeholders and the product owners should be able to access the system and its functionalities as a whole(Both as admin and user side)

4.3.1.5. User Satisfaction

The system should be designed keeping in mind the user demands and expectations. The user should be satisfied with the interface and the functionality of the system.

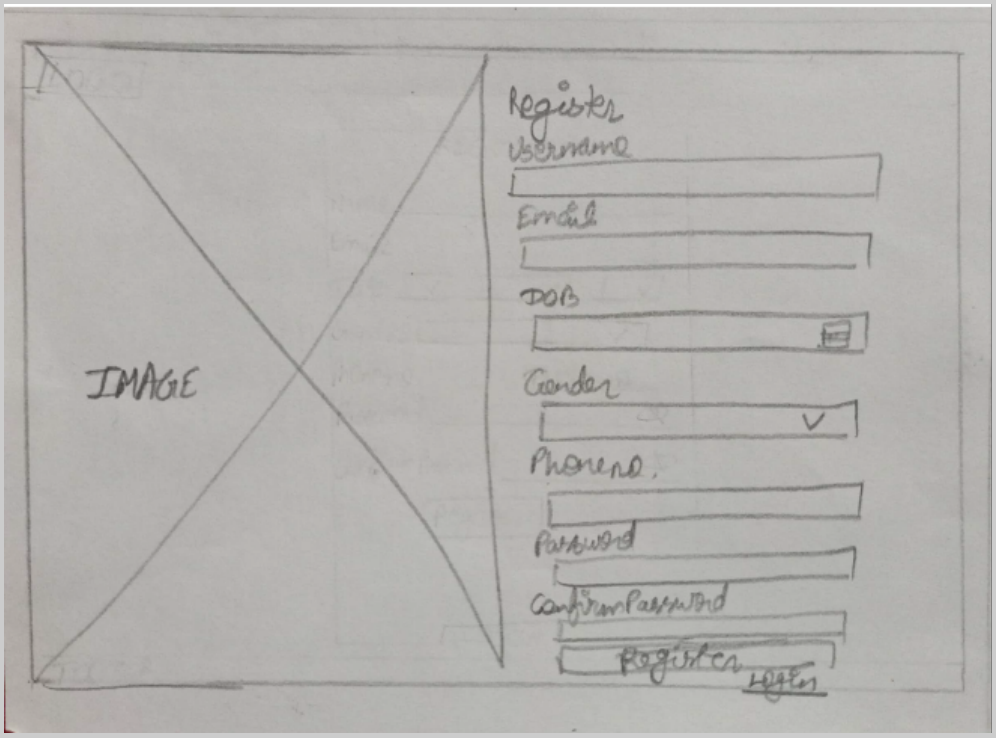
4.3.2. Data Design

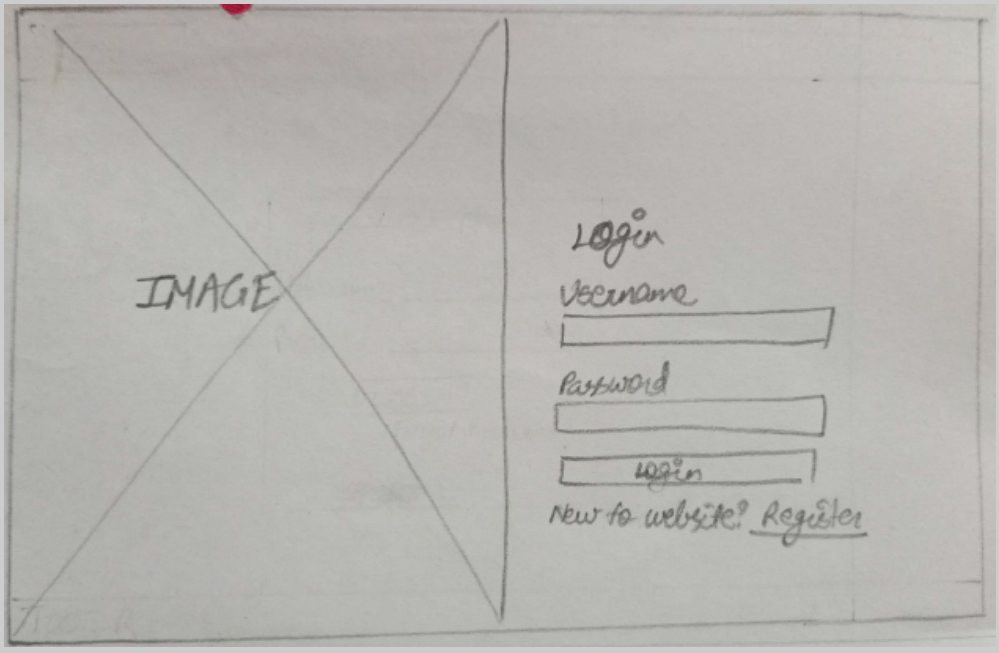


The data design describes how the data can be structured to use in designing the database for the system and maintaining the information about the different entities and its corresponding attributes.

4.3.3. Interface Design

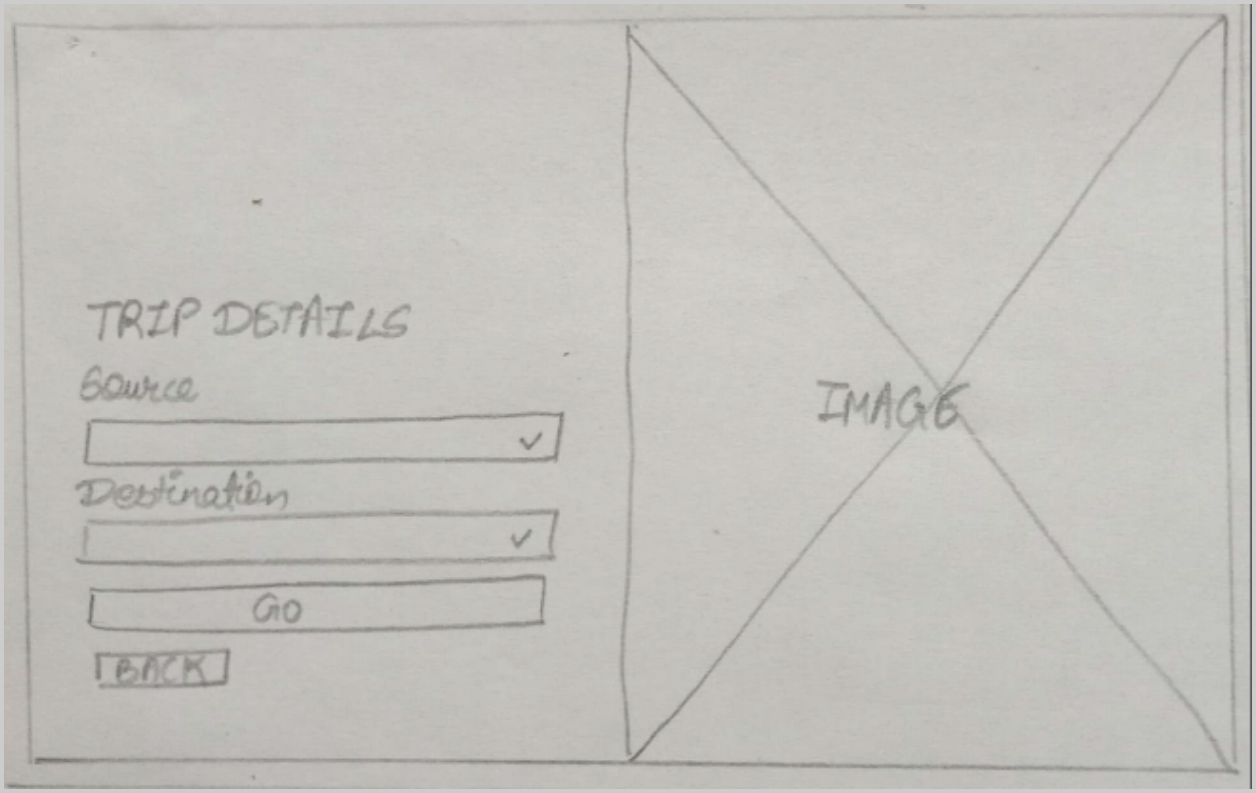
4.3.3.1. Sign in/Sign up interface



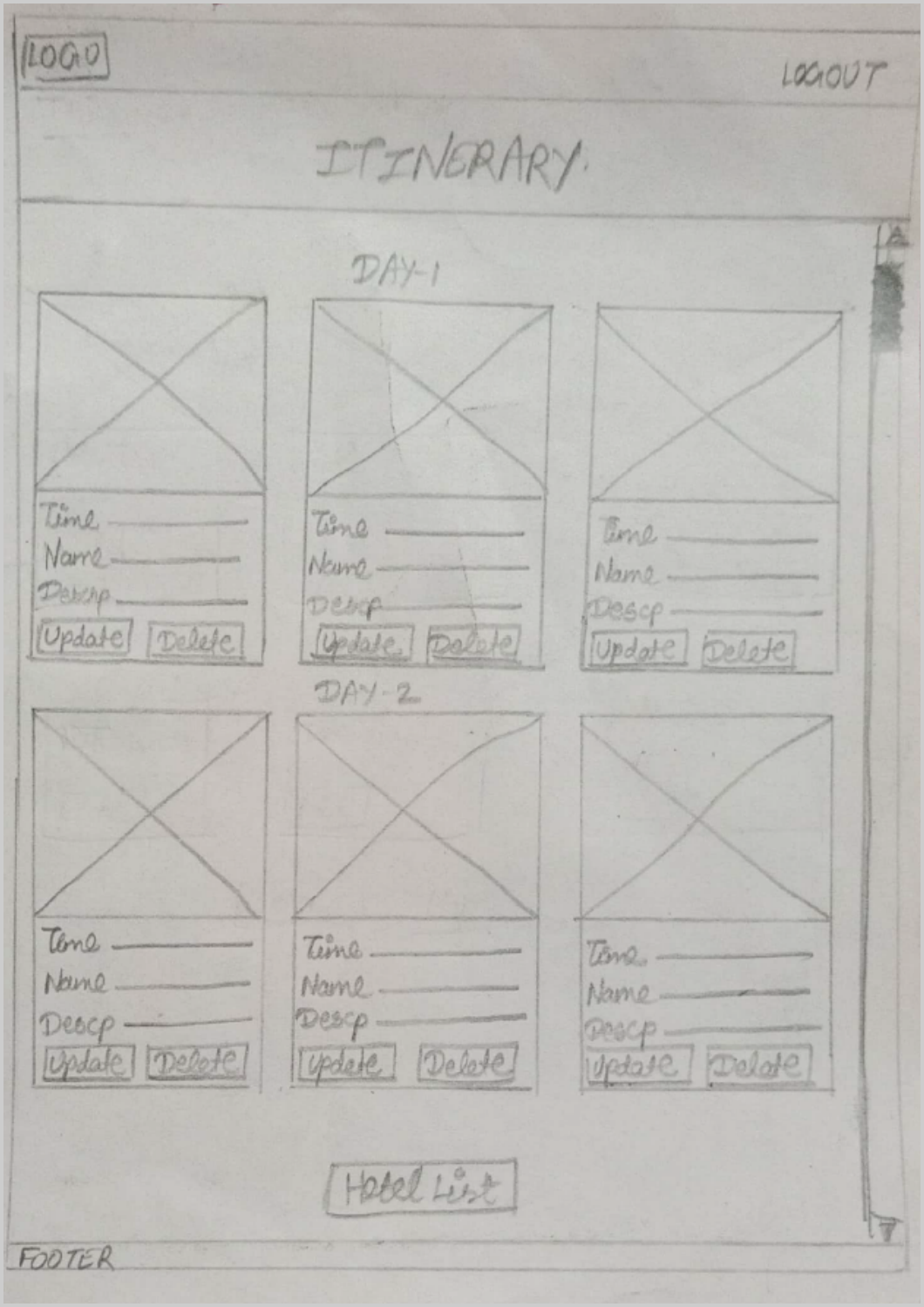


4.3.3.2. Home interface

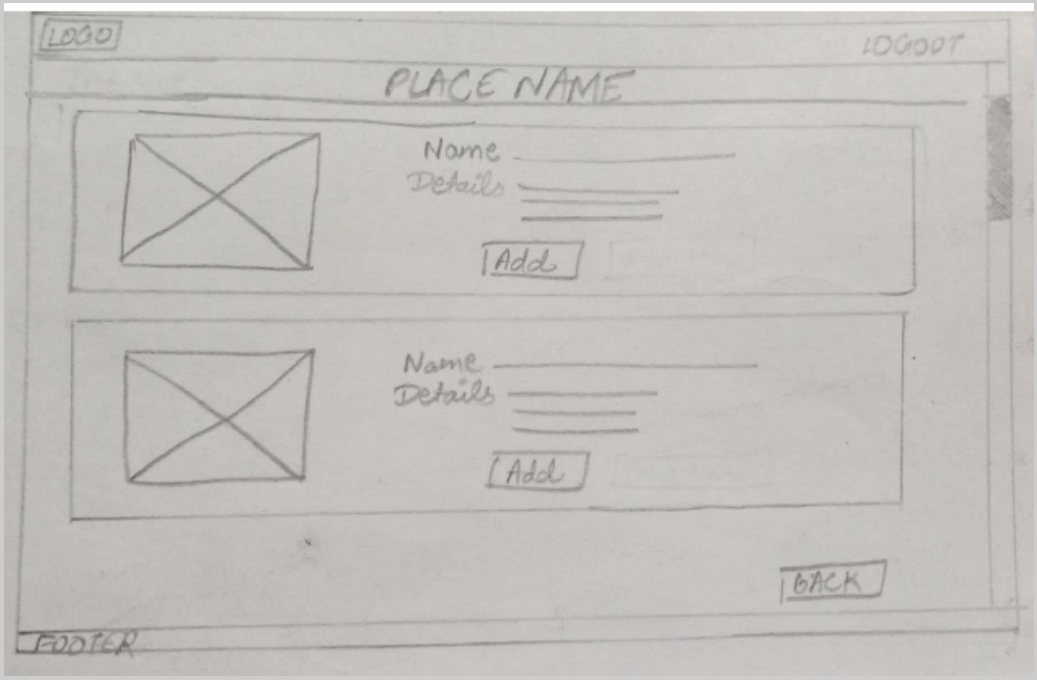
4.3.3.3. Trip input interface



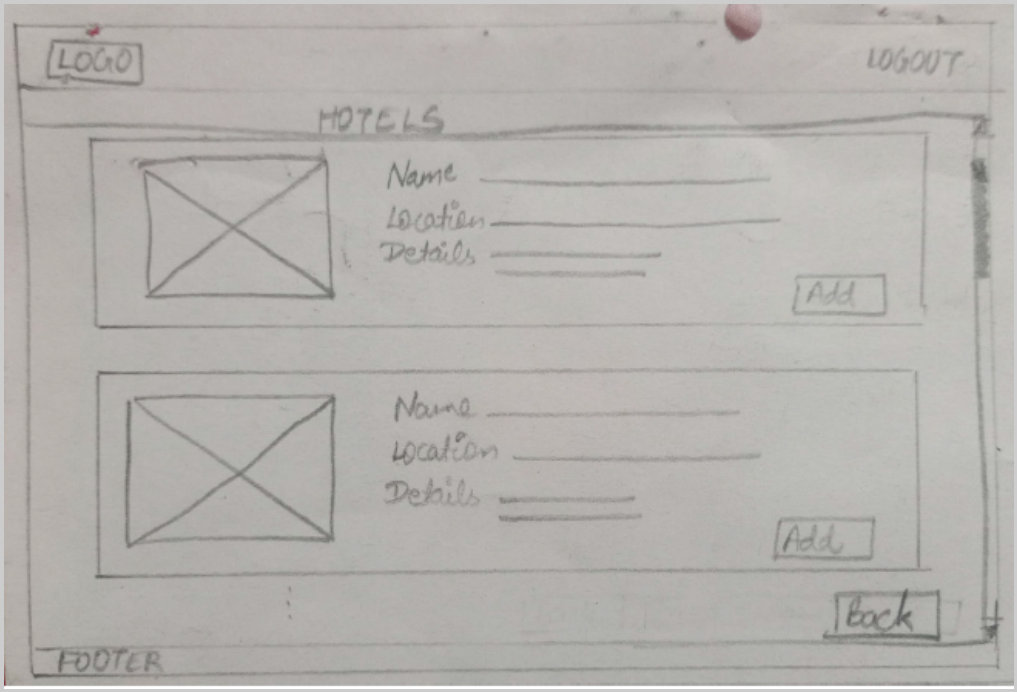
4.3.3.4. Itinerary interface



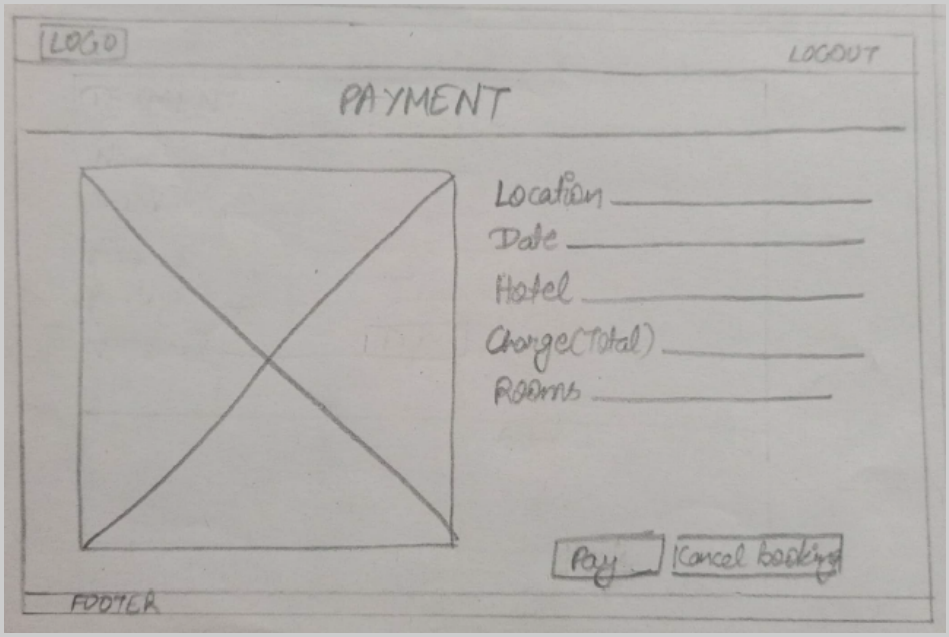
4.3.3.5. Add/Update Place interface



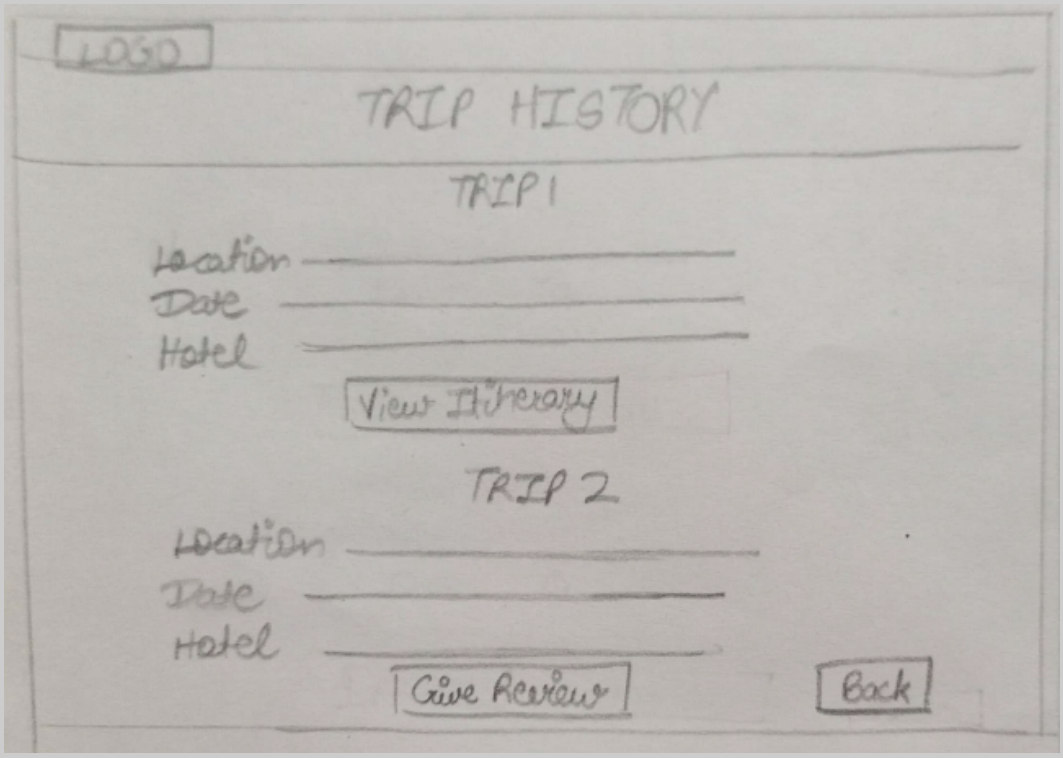
4.3.3.6. Hotel Booking interface



4.3.3.7. Payment interface



4.3.3.8. Trip History interface



5. System Risks Involved

5.1. Software Risks

This risk tells about the chances of occurrence of uncertain events and how these events incur loss within the organisation/company. Managing such risks is an important part of all the companies to continue to provide better services in terms of technology and user-friendly environment. The software risk can be viewed as a combination of performance, security, efficiency, robustness and risk present within the system. These risks can be avoided by proper consideration of system vulnerabilities, compliance issues, stability problems, performance degradation, and security flaws.

5.2. Risk Identification

The identification of the risk within the system needs to be done as early as possible such that the impact of risk can either be eradicated or minimised, thus making the system effective. The system can have multiple risks varying from the risks in requirement to the risk in deployment. So, it becomes necessary to categorise the risk and then resolve them. The risks can be identified into major categories as:

- Requirements Risk
    - Risks that assume from the changes to the customer requirement and the process of managing the requirements change.
- Design Risks
    - Risks that assume from the changes to the UI design and customer desires for accessing the system and product owner desires in the development design change.
- Tools and Technology Risk
    - Risks that assume from the software or hardware technologies that are used to develop the system.
- Budget and Resources Risk
    - Risks that assume from the management estimates of the resources required to build the system
- Operational Risk
    - Risks which happen in day-to-day operational activities during project development

5.3. Risk Projection/Analysis

Risk Type	**Risk Issue**
Requirements Risk	Requirements of the user and stakeholders are not clearly defined
	Requirements are not completely understood by

	the team or described by the user
	Requirements are ignored by the team considering it as not required, inconsistent
	Requirements are clashing among the different types of users or user and owner
Design Risk	Design is not as per the user requirements
	Design is too much complex to understand by its users
	Design contains many unnecessary elements in the system
	Design is incomplete or unclear due to lack of understandability between user and designer
Tools and Technology Risk	Technology for development selected not upto to the mark for developing the system
	Tools are not able to integrate the modules of the system
	Constant change in technology due to change in requirements and design
	Technology and tools used are complex to understand for the user
Budget and Resources Risk	Lack of budget and resources for developing the system
	Overshot of resources and budget while creating the system
	Unexpected changes in system leading to frequent changes in budget
Operational Risk	Limited amount of resources
	Lack of planning of the developer team in development process
	Miscommunication between the product owner and the developer
	Lack of management of the tasks and limited number of people in developer team

5.4. Risk Management/Resolution

It is mainly the process of achieving the desired or the required goals by managing the risks within the system. After identifying and analysing the risks, the project should include all the likely risks to occur. Every risk needs to have a different solution for itself and thus needs ingenuity of the project manager and the developing team for tackling the risk. The main methods to perform risk management are:

- Avoid the risk as much as possible
    - This may take several ways such as discussing with the client to change the requirements to decrease the scope of the work, giving incentives to the engineers to avoid the risk of human resources turnover, etc.
- Transfer the risk
    - This method involves getting the risky element developed by a third party, like payment gateway etc.
- Reduce the risk
    - This means a planning method to include the loss due to risk. For instance, if there is a risk that some famous places are not there in the list, then the admin side should add them to satisfy user needs.

6. References

[1] https://www.geeksforgeeks.org/different-types-of-risks-in-software-project-development/
[2] https://www.javatpoint.com/software-engineering-risk-management-activities
[3] https://www.castsoftware.com/research-labs/software-risk
[4] https://www.lucidchart.com/blog/how-to-create-software-design-documents
[5] https://xd.adobe.com/ideas/principles/web-design/design-documentation/