NATIONAL UNIVERSITY

A PROJECT REPORT ON
ONLINE BASED TRAVEL PARTNER & PACKAGE FINDER

Submitted for the course CSE-499

Department of Computer Science and Engineering
Tejgaon College, Dhaka

Submitted By

Name	Registration No.	Session
Tanmay Kumar	16502000676	2016-17
Md. Yakub Ali	16502000676	2016-17

Supervised By

Md. Moklesur Rahman
Lecturer
Department of CSE
Tejgaon College, Dhaka

DECLARATION

We hereby declare that, this report has been prepared by **Tanmay Kumar** and **Md. Yakub Ali** under the supervision of Md. Moklesur Rahman, Lecturer, Department of Computer Science and Engineering, Tejgaon College, Dhaka. We also declare that neither this report nor any part of this report has been submitted elsewhere for award of any degree.

Supervised by:

.....
Md. Moklesur Rahman
Lecturer,
Department of Computer Science and Engineering
Tejgaon College, Dhaka

Submitted by:

.....
Tanmay Kumar
Registration No: 16502000676
Session: 2016-17

.....
Md. Yakub Ali
Registration No: 16502000676
Session: 2016-17

APPROVAL OF ACCEPTANCE

A project report written by **Tanmay Kumar** (Reg: 16502000676) and **Md. Yakub Ali** (Reg: 1650200676) entitled "Online Based Travel Partner and Package Finder" is submitted to the Department of Computer Science and Engineering, Tejgaon College, Dhaka for partial fulfillment of the requirements for the degree of B.Sc. (Hon's) in Computer Science and Engineering. The project is done under the supervision of Md. Moklesur Rahman, Lecturer, Department of Computer Science and Engineering, Tejgaon College, Dhaka.

……..
1st Examiner

……..
2nd Examiner

Accepted by:

……..
Md. Moklesur Rahman
Lecturer,
Department of Computer Science & Engineering
Tejgaon College
Dhaka, Bangladesh

ACKNOWLEDGEMENT

This research project would not have been possible without the support of many people. We wish to express our gratitude to our supervisor Md. Moklesur Rahman, Lecturer, Department of Computer Science and Engineering, Tejgaon College, Dhaka, who offered us invaluable assistance, guideline and support. Our deepest gratitude to him without his support and assistance this study would not have been successful.

Thanks to our parents for their moral support. Special thanks to those people who provided invaluable assistance and advice.

ABSTRACT

This report is about the project "**Online Based Travel Partner and Package Finder**" that will provide us simple interface to find a suitable trip package or travel partner for one's dream vacation. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting relevant information may be very time consuming. All these problems are expected to be solved using this system.

This project will provide facilities to the travelers to find tour partner to minimize their tour cost and also the travel & tour companies to provide better service with a very interactive content management system to easily access the information etc.

This report covers all the designed development strategies of the project and also every module of the system is discussed with necessary diagrams, flowchart.

TABLE OF CONTENTS

CHAPTER-1
INTRODUCTION

The globalization as main effect of internationalization in business is widely recognized as we enter a new millennium. Commonly used term in the last ten to fifteen years is “web planning”. This domain was known prior as “personnel administration” and name change is not only question of terminology. Smart tourism as new term is a new buzzword applied to describe the increasing reliance of tourism destinations, their industries and their tourists.
Tourism is gradually being transformed to a new sphere of competitiveness globally. The creation of unique, meaningful and memorable experiences for consumers has been postulated as the key to generate added consumer value and competitive advantage. Currently, two major forces are however fundamentally transforming our understanding of how tourist experiences can be created. Several systems have been developed to help tourists organize a customized trip. We have analyzed them and detected some new aspects which could be considered in order to offer each tourist the most suitable itinerary when planning a trip involving a series of activities.

1.1 Travel Partner & Package Finder

With this the travelers will easily their suitable travel partners or packages to have a dream vacation the traveler expects. The purpose of the study is to design and develop a recommended system based on agent and web technologies, which utilizes a hybrid recommendation filtering for the smart tourism industry. A hybrid recommendation system based on agent technology is designed by considering the online communication with other sectors in the tourism industry, such as the tourism supply chain, agency etc. This system will open a new era of business opportunities to travel agents as they can showcase their packages with a competitive price and provide a better service.

1.2 Existing System

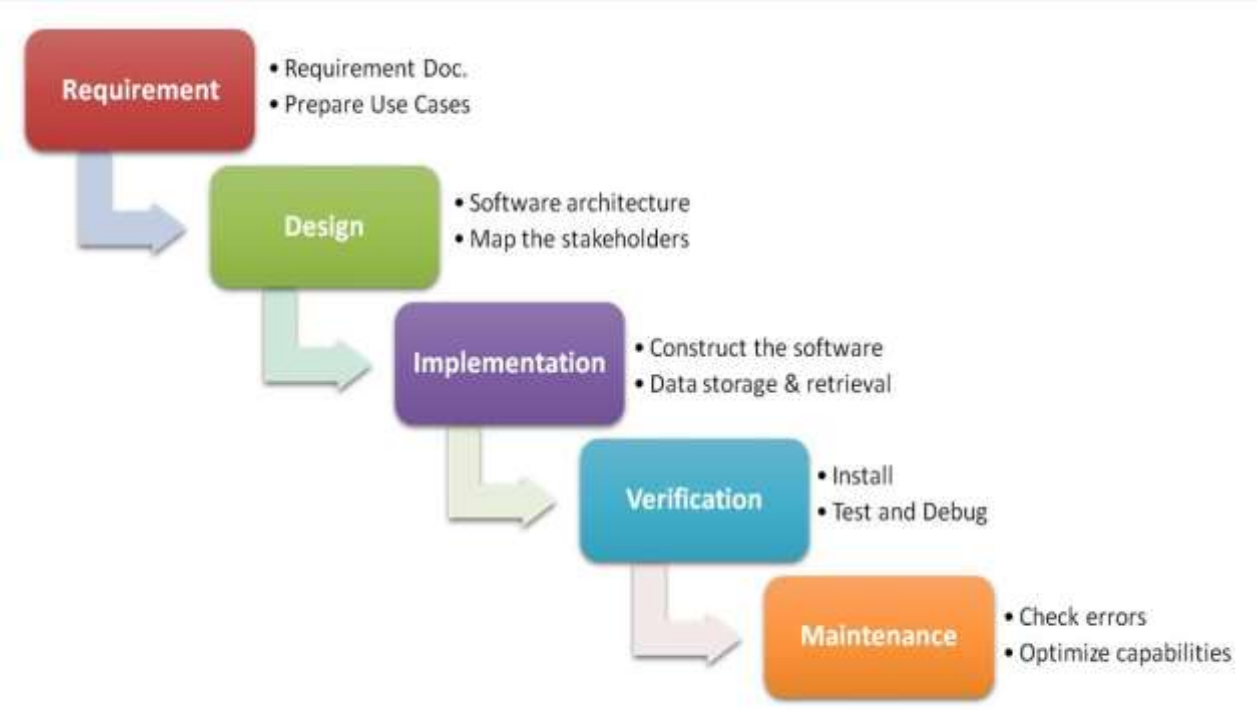
Trip planning with web and mobile applications is a popular service. When planning their travel, main problems of tourists are getting lost in the abundance of various tourism offers and losing a lot of time to find what you need. Travel, accommodation, sightseeing can all be organized by yourself, but it will take you unbelievable 45 days! just to find, choose and plan the trip, according to the researches. The modern internet user is daily faced with the problem of too much information, most of them being unimportant and useless.

1.3 Proposed Automated & Advance System

In our proposed automated system, everything will be much simpler and user friendly. A traveler can find suitable partners or suitable vacation plans & packages from the travel agents. The agencies can showcase their packages with a competitive price and provide a better service. The system is expected to be a client server architecture system. There will be a Node.js Server powered by Express.js framework. Which is the core building block of this whole infrastructure. It will provide a handsome amount of RESTful endpoint to make HTTP call like GET, POST, PUT, PATCH, DELETE. Node.js Server will communicate to the Mongo Database (which is a so-called no SQL Database) through Mongoose ORM (Object Relational Mapper).

1.4 Software Development Method

The method we are going to use to develop this routine management application is **Waterfall** method. The waterfall model is a traditional engineering approach applied to software engineering. A strict waterfall approach discourages revisiting and revising any prior phase once it is complete. This "inflexibility" in a pure waterfall model has been a source of criticism by supporters of other more "flexible" models. It has been widely blamed for several large-scale government projects running over budget, over time and sometimes failing to deliver on requirements due to the Big Design Up Front approach. Except when contractually required, the waterfall model has been largely superseded by more flexible and versatile methodologies developed specifically for software development.



The waterfall model is a sequential development approach, in which development is seen as flowing steadily downwards (like a waterfall) through several phases, typically:

- Requirements analysis
- Software design
- Implementation
- Testing
- Integration, if there are multiple subsystem
- Deployment
- Maintenance

The basic principles are:

- Project is divided into sequential phases, with some overlap and splash back acceptable between phases.
- Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.
- Tight control is maintained over the life of the project via extensive written documentation, formal reviews, and approval/signoff by the user and information technology management occurring at the end of most phases before beginning the next phase. Written documentation is an explicit deliverable of each phase.

1.5 Application Platform

In this expected web platform, we are going to host our Admin CMS. Also known as Web client application. The Web-Based Client will be used by the System Admins or in short Teachers to update & modify the class schedule. It's will be a pure React.js SPA (Single Page Application. It's going to use webpack under the hood to bundle all the asset like image, Compressed JavaScript bundle, Minified CSS with prefix etc. This React application will communicate to Server through AJAX (Asynchronous JavaScript and XML) call. It will be much more interactive UI because there will be no page loading related with actions & events. Like when button pressed, submitting form, navigation etc. It can have different admin to manage the information. They can be added and removed via going to profile.

1.6 Chapter Outline

Chapter one is introduction. Where we are going to have basic overview about the project and introductory information. It will give an idea about what we are trying to accomplish. In Chapter two we are going to analysis our proposed system and basic requirements that is required to complete our project. In Chapter three we are going to specify our requirements and present it in an understandable manner like a pictorial or diagram view. Chapter four explain how we are going to implement our design and how it is going to look and feel. In Chapter five we are going to discuss about the testing schedule and testing approach we are

going to adopt. In chapter six we are going to point out our current limitation and discuss about future enhancements. Chapter seven will be the conclusion of our project discussion.

1.7 Project Schedule

There is a six-month timeframe to implement class routine management system in BITC from the both side of the lecturers & students that the project commencement in time for spring 2020.

Month	Task
November	**System Analysis**
	1. Target Audience and Benefits. 2. Software Requirements and Specification 3. Use Cases
December	**Planning**
	1. Information Gathering 2. Analysis of existing system Database Infrastructure
January	**System Design**
	1. Tools Selection 2. UI & UX Design
February	**System Implementation**
	1. Developing the Backend 2. Developing the Frontend 3. Developing the Mobile Application
March	
April	**Testing**
	1. Unit Testing 2. Integration Testing 3. System Testing

CHAPTER-2
SYSTEM ANALYSIS

We are going to analysis the existing travel package offering websites to identify our requirements and set up our development strategies. System analysis will help us to keep track of our development progress. Also, it is going to help us to divide our projects in multiple steps so that the goal is easy to reach.

2.1 Background Study

The existing system involves lots and lots of paper work. The system involves that all user details will be taken on a static method. Current system of planning a tour is a very tedious process. In the present system a customer has to approach various agencies to find details of places and to book tickets. This often requires a lot of time and effort. A customer may not get the desired information from these offices and often the customer may be misguided. It is tedious for a customer to plan a particular journey and have it executed properly. The overview knowledge of the existing system,

i. **Loss of Data**: Lots of paper work are needed for the safe keeping of the details or information.
ii. **Time Wasting**: User time are wasted as a result of searching the routine data to the notice board.
iii. **Error Prone**: The existing system of operation is prone to error.
iv. **Processing Speed**: The processing speed is very low resulting into low output.

2.2 Target Audience and Benefits

2.2.1 Target Audience

With this system, the travelers will easily their suitable travel partners or packages to have a dream vacation the traveler expects. The purpose of the study is to design and develop a recommended system based on agent and web technologies, which utilizes a hybrid recommendation filtering for the smart tourism industry. A hybrid recommendation system based on agent technology is designed by considering the online communication with other sectors in the tourism industry, such as the tourism supply chain, agency etc. This system will open a new era of business opportunities to travel agents as they can showcase their packages with a competitive price and provide a better service.

2.2.2 Benefits of User

A traveler will be able to see partner requirement post and packages if they search for it. They can also review the package they bought through the system. This system will open a new era

of business opportunities to travel agents as they can showcase their packages with a competitive price and provide a better service. The agents will be able to post, update and promote their package. They will receive according to their provided service. From those reviews they can improve their service.

2.2.3 Environment

The system is web server that will run on a machine with 24*7 monitoring. User can use this system anywhere. This system can run on any machine. They do not need any high configuration machine. They need Pentium processor, 1024 MB RAM, 32-bit operating system, Mozilla Firefox and Google Chrome. That's the machine configuration but there is some software environment required like to run the whole server application we need a node.js environment. Because backend code or formally server code is written using node.js language. Node.js is an open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside of a browser.

2.2.4 Stakeholders

A traveler and travel planner agent can use this application via web protocol. Without this person no one can access this application. They can easily find their web pages. It actually a web app formally known as SPAs (Single page Application). Currently it's one of the best approaches to create a web app which is dynamic.

2.2.5 Performance and Capacity Needs

- 2,000 users can use the application at a time.
- It will be an open source web application.
- Use available Operating System.
- Admin CMS compatible with most common browser.
- Server should have a node.js runtime.
- Web application doesn't require too much power to run.
- Both serve most user-friendly user interface.

2.2.6 User Stories

We were travelling to Saint Martin Island of Bangladesh at the end of 2021. Before travelling there, we planned our trip in every way possible. But this was not enough. We felt the lack of a fully organized trip experience. Also, we felt like, if there were 2 more people in our group, the trip would have been so much budget friendly. So, that's we planned this project for both the travelers and the travel agents.

Requirement Interview

Number	Question & Answer	
1	Question	What are user interface requirements?
	Answer	1. Easy to operate. 2. Quick to response. 3. Effectively handling operational errors. 4. providing simple yet consistent user interface.
2	Question	What qualities do you want?
	Answer	1. Operational 2. Transitional 3. Maintenance 4. Usability 5. Efficiency 6. Security
3	Question	Probably how many users will use it concurrently?
	Answer	At least around 500 students at the same time.
4	Question	Do you want the admin content management system?
	Answer	Yes! It should be mobile responsive.

2.3 System Requirements Gathering

2.3.1 Functional Requirements

In software engineering and systems engineering, a functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between outputs and inputs. Functional requirements may involve calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describe all the cases where the system uses the functional requirements, these are captured in use cases. Functional requirements are supported by non-functional requirements (also known as "quality requirements"), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). Generally, functional requirements are expressed in the form "system must do <requirement>," while non-functional requirements take the form "system shall be <requirement>." The plan for implementing functional requirements is detailed in the system design, whereas non-functional requirements are detailed in the system architecture. As defined in requirements engineering, functional requirements specify particular results of a system. This should be contrasted with non-functional requirements, which specify overall characteristics such as cost and reliability. Functional requirements drive the application architecture of a system, while non-functional

requirements drive the technical architecture of a system. In some cases, a requirements analyst generates use cases after gathering and validating a set of functional requirements. The hierarchy of functional requirements collection and change, broadly speaking, is: user/stakeholder request → analyze → use case → incorporate. Stakeholders make a request; systems engineers attempt to discuss, observe, and understand the aspects of the requirement; use cases, entity relationship diagrams, and other models are built to validate the requirement; and, if documented and approved, the requirement is implemented/incorporated. All those things we can include in the functional requirements.

2.3.2 Non-Functional Requirements

A Non-Functional Requirement (NFR) defines the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to success of the software system. Example of nonfunctional requirement, how fast does the website load? Failing to meet non-functional requirements can result in systems that fail to satisfy user needs.

Non-functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users is > 10000. Description of non-functional requirements is just as critical as a functional requirement.

Types of Non-functional Requirement

- Usability requirement
- Serviceability requirement
- Manageability requirement
- Recoverability requirement
- Security requirement
- Data Integrity requirement
- Capacity requirement
- Availability requirement
- Scalability requirement
- Interoperability requirement
- Reliability requirement
- Maintainability requirement
- Regulatory requirement
- Environmental requirement

2.3.3 Environmental Requirements

Environmental requirements limit the effect that external environment (natural or induced) is to have on the system, and/o the effect that the system is to have on the external enveloping environment.

2.4 Use Cases

A use case diagram is a graphic depiction of the interactions among the elements of a system use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular Environment and related to a particular goal.

There are three types of users in this system. The first two are traveler and agents who are the main user of this system and the third type of user, the administrator, who is able to initially setup the system, modify this system and set their authorization level.

Travelers: This is a one of the main users of this system. This type of user able to see partner requirement post and packages if they search for it. They can also review the package they bought through the system.

Agents: This is also one of the main users of this system. This type of user also able to post,

update and promote their package. They will receive according to their provided service. From those reviews they can improve their service.

Admin: Finally, the system administrators are users who are able to setup the system from the initial installation and maintain the systems member accounts. They automatically have the functionality of authorized users within the normal operation of the system.

Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

For our better understand we also draw activity diagram for different users that include Teacher activity, Student activity and Admin activity that show below in figure no- 3.2, 3.3, 3.4.

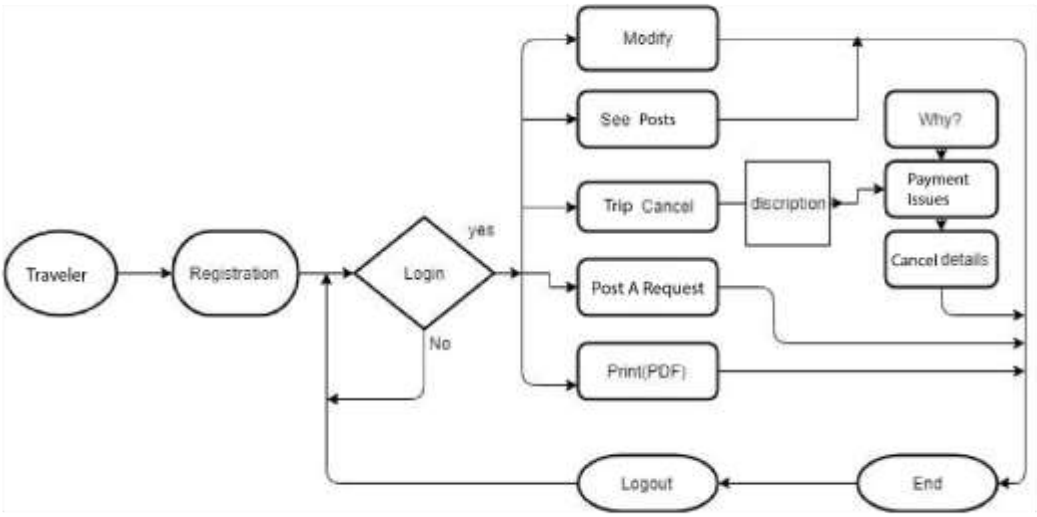


Figure 3.2 Activity diagram for Traveler

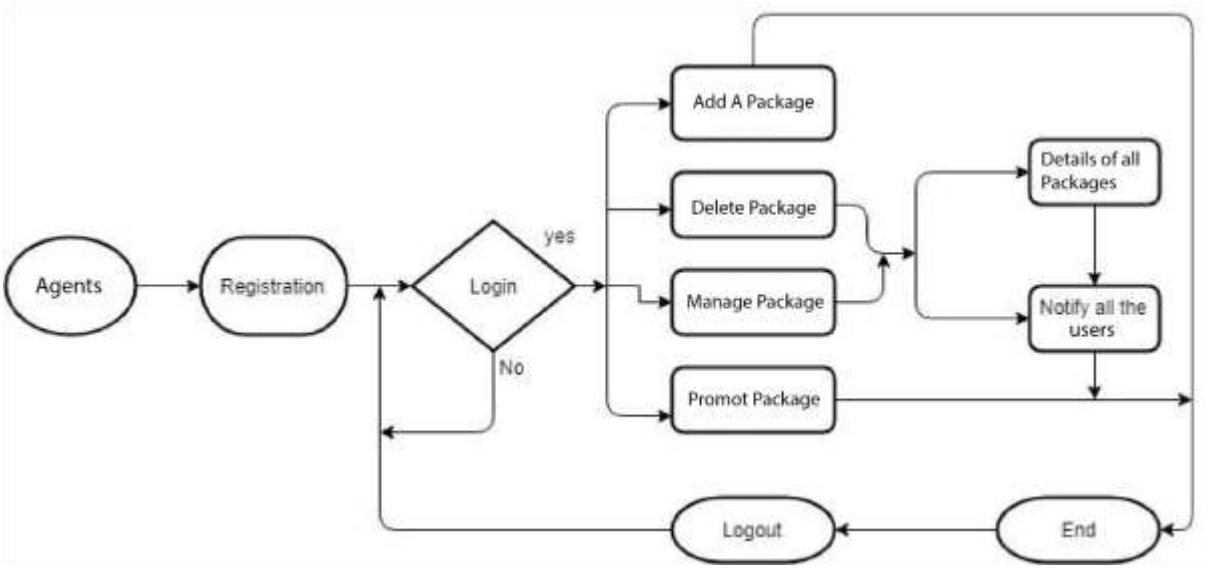


Figure 3.3 Activity diagram for Agent

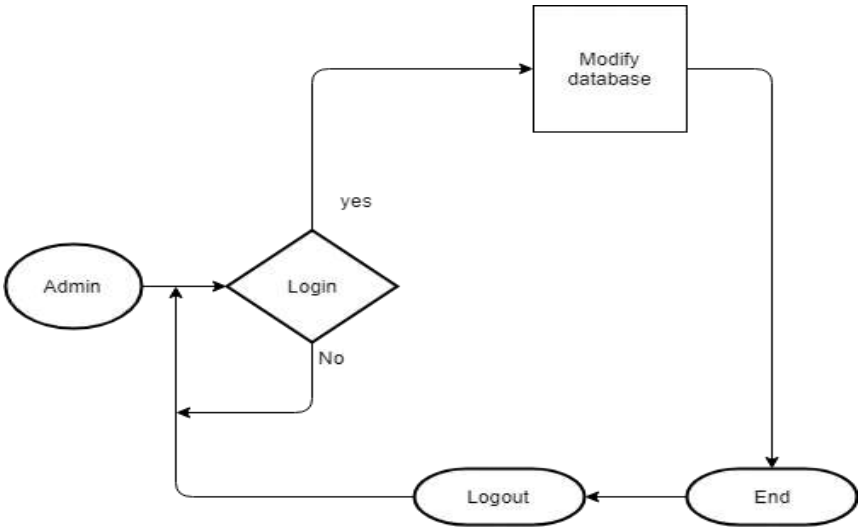


Figure 3.4 Activity diagram for admin

System Sequence Diagram

In software engineering, a system sequence diagram (SSD) is a sequence diagram that displays, for a specific situation of a use case, the proceedings that outside performers generate their instruction, and likely inter-system proceedings. Here we also draw system sequence diagram that helps us how they interact with this system that show in figure no-3.5, 3.6, 3.7.

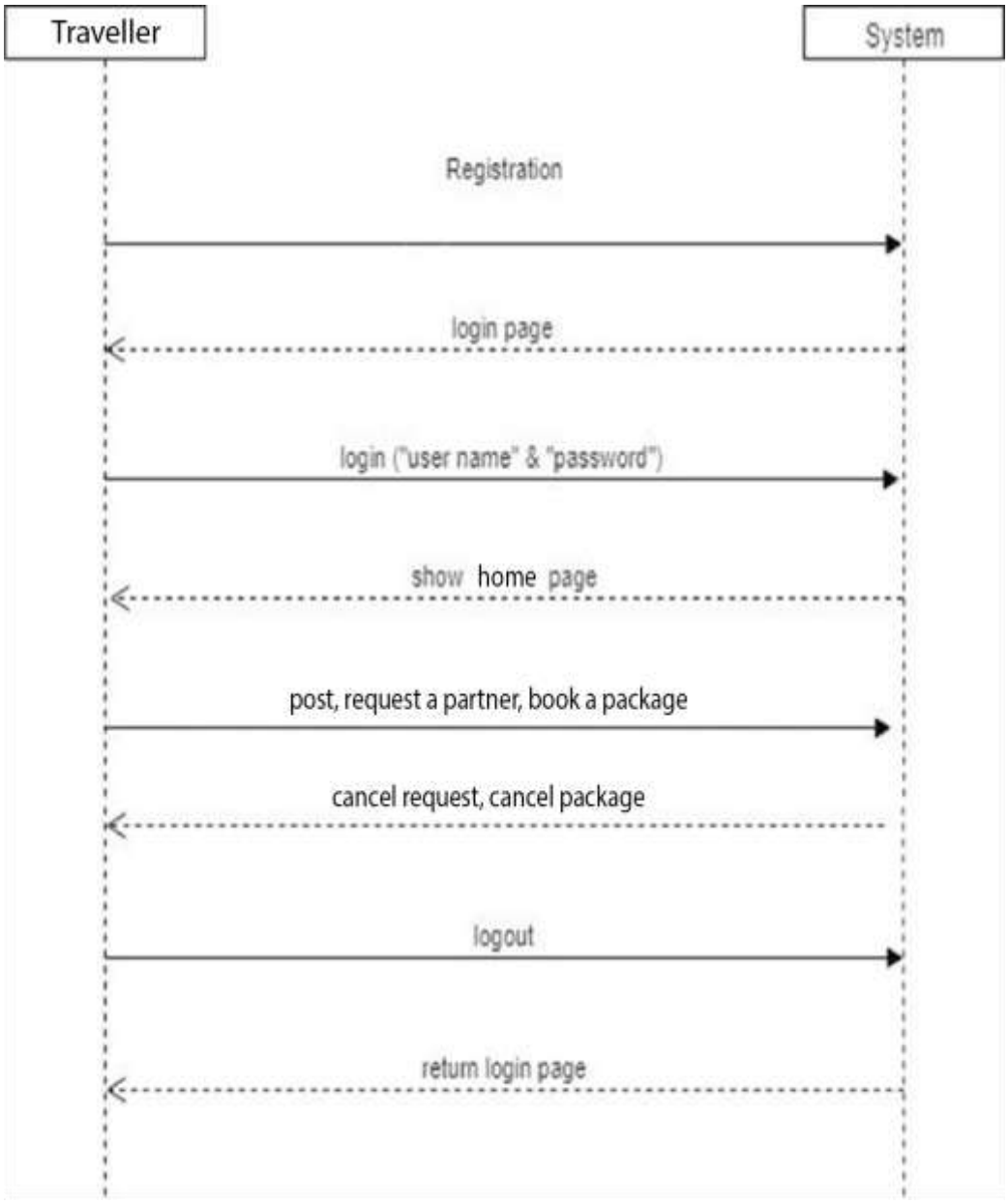


Figure 3.5 System Sequence diagram for teacher

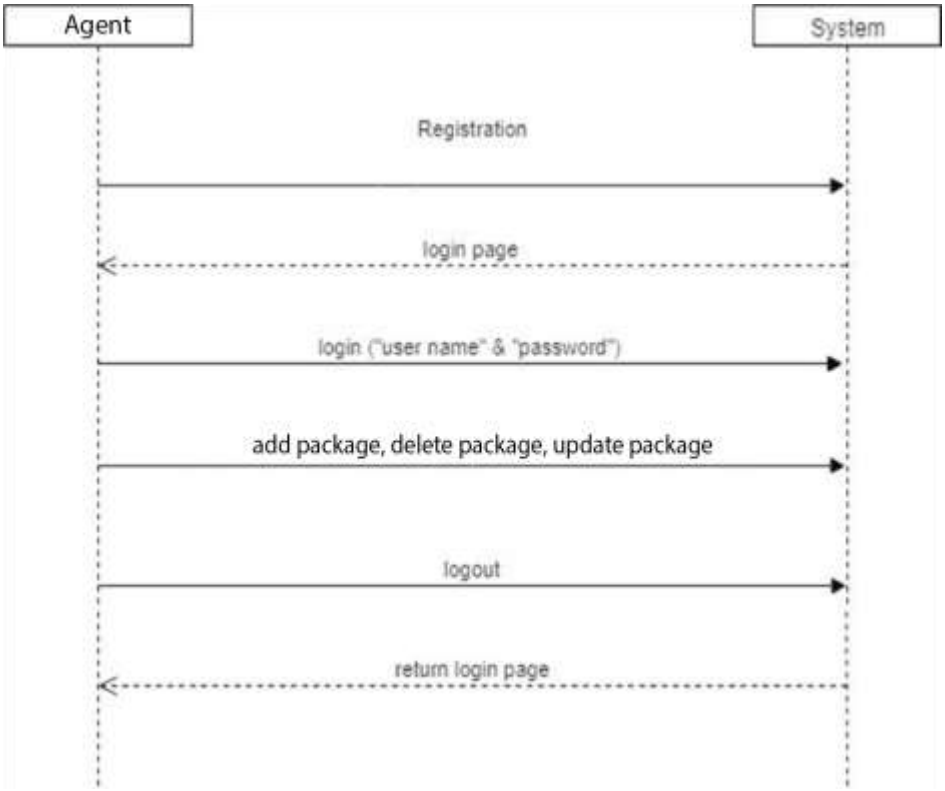


Figure 3.6 System Sequence diagram for student

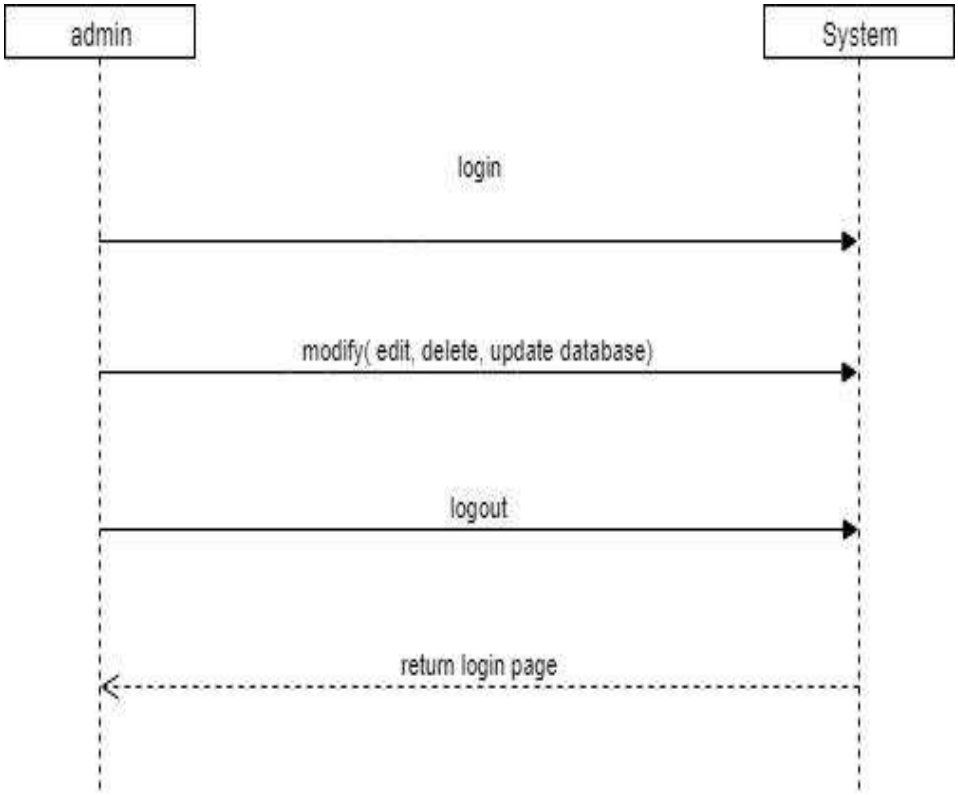


Figure 3.7 System Sequence diagram for admin

2.5 Information Gathering

Before setting up the system by software development tools, information will be gathered from the college system about the need for the users. In this stage the first goal will be decided by task analysis. Finally, the system will be finalized with the amendment on some problems of the user interface. In order to accurately and comprehensively specify the system, the software engineer gathers and analyzes information via various methodologies. This chapter discusses these methodologies as outlined below:

- Rationale for Information Gathering
- Interviews
- Questionnaires and Surveys
- Sampling and Experimenting
- Observation and Document Review
- Prototyping
- Brainstorming and Mathematical Proof
- Object Identification
- Summary and Concluding Remarks

Requirements gathering is a crucial part of any project, large or small. It is essential to understanding and fulfilling the needs of the customers. The process of requirement gathering include identifying and documenting the necessary requirements of customers, users, stakeholders etc. related to the project.

CHAPTER-3
DESIGN SPECIFICATION

3.1 Introduction

To develop a best fit system there are three stages. They are gathering information, design and implementation and final testing. Within these three sections, different tactics will be adopted so that we can design a system that can maintain high usability and accessibility. A design specification is a detailed document providing a list of points regarding a product or process. For example, the design specification could include required dimensions, environmental factors, ergonomic factors, aesthetic factors, maintenance that will be needed, etc. It may also give specific examples of how the design should be executed, helping others work properly (a guideline for what the person should do). Design Specifications describe how a system performs the requirements outlined in the Functional Requirements. Depending on the system, this can include instructions on testing specific requirements, configuration settings, or review of functions or code. All requirements outlined in the functional specification should be addressed; linking requirements between the functional requirements and design specification is performed via the Traceability Matrix. Good requirements are objective and testable. Design Specifications may include:

- Specific inputs, including data types, to be entered into the system.
- Calculations/code used to accomplish defined requirements.
- Outputs generated from the system.
- Explaining technical measures to ensure system security.
- Identify how the system meets applicable regulatory requirements.

System Requirements and verification of the installation process are usually tested in the Installation Qualification. Input, Processing, Output, and Security testing are usually tested in the Operational Qualification.

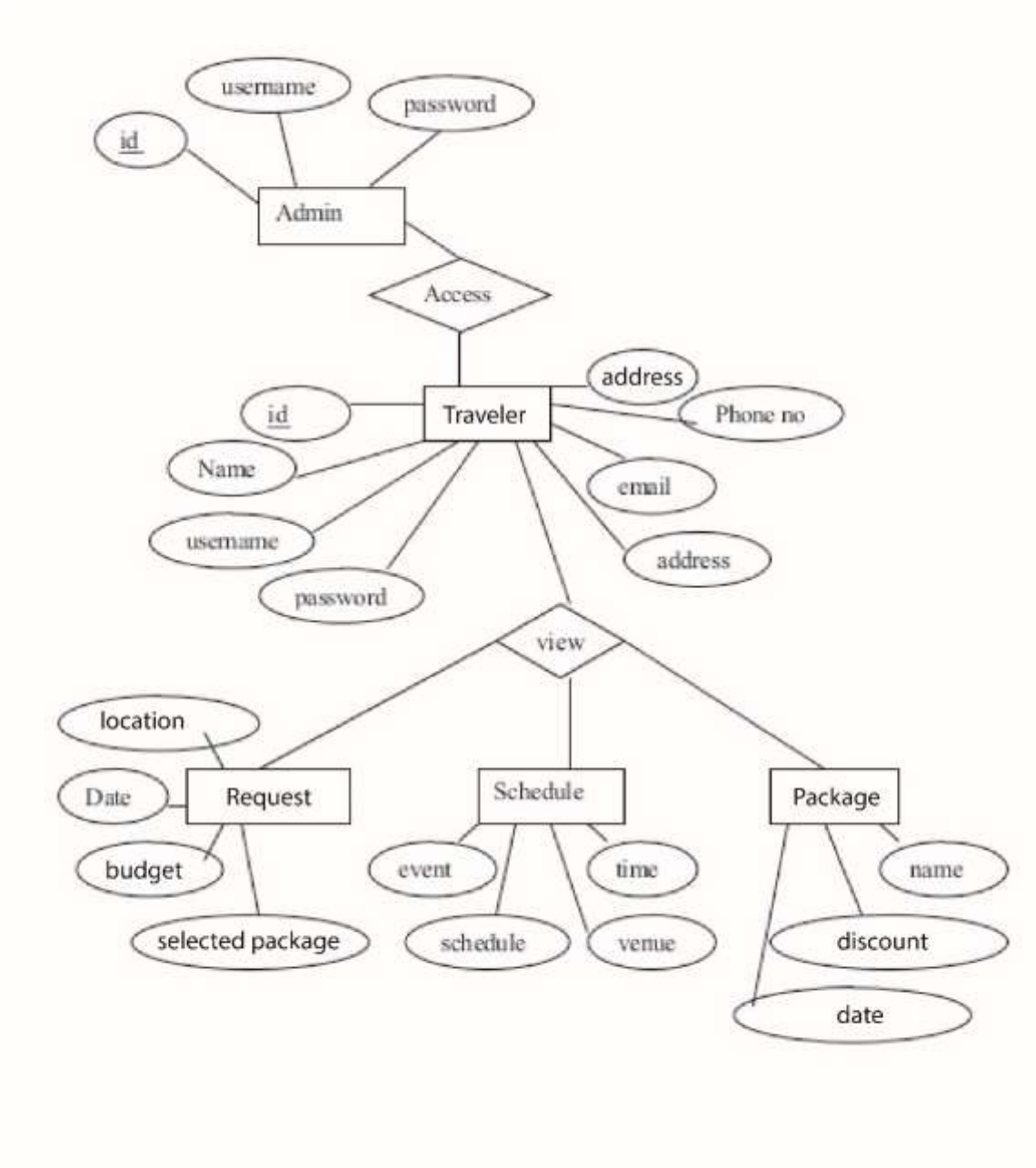
Due to the extremely technical nature of most design documents, there is currently some discussion in the industry about who needs to review the Design Specification. The Design Specification is reviewed and approved, at minimum, by the System Owner, System Developer, and Quality Assurance. Quality Assurance signs to ensure that the document complies with appropriate regulations and that all requirements were successfully addressed, but they do not necessarily need to review technical information.

Depending on the size and complexity of the program, the design specification may be combined with the functional requirements document.

3.2 Design and Implementation Methodology

The design methodology used in the proposed system is parallel as result of the face that parallel methods support the use of the proposed system side by side with the existing system in order to test for the system efficiency. Top down approach is used as will in the design because it allows the analysis of the system to be carried out one after the other. Next, the prototype of the system will be analyzed. Then a more complete prototype will be potential uses to collect feedback.

3.3 Data Flow Diagram



3.4 Sequence Diagram

Sequence diagrams are sometimes called event diagrams or event scenarios. Here sequence diagram represents how the system process work through the sequence in this system. In this sequence diagram we try to show all different users process through by the sequence diagram in below in figure no-4.1, 4.2 and 4.3.

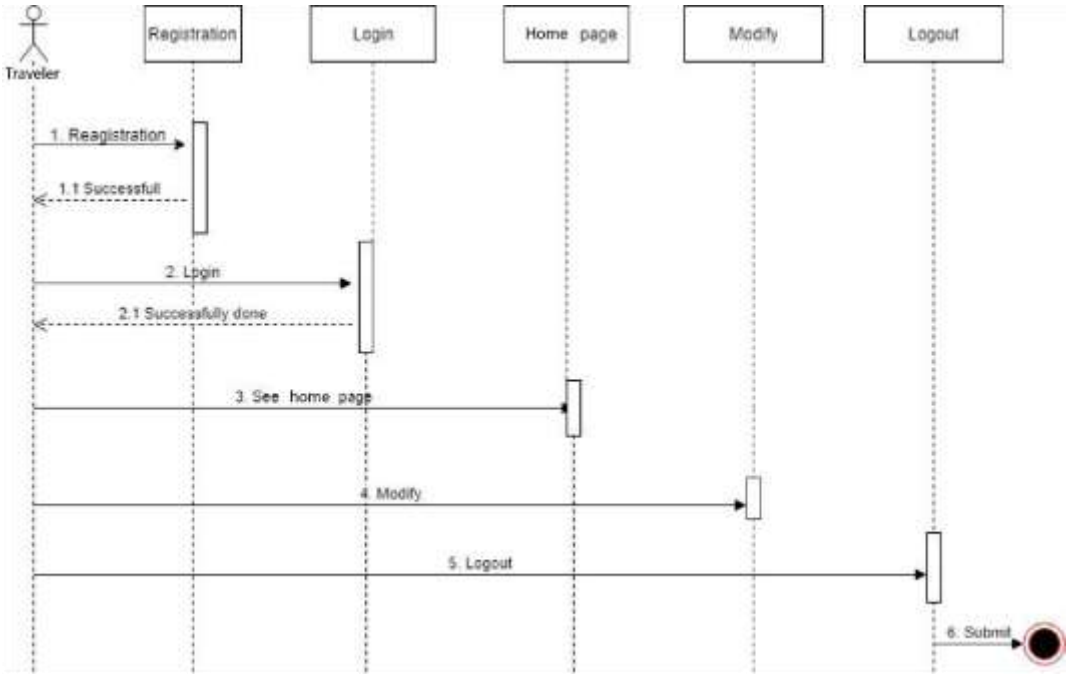


Figure 4.1: Sequence diagram for traveler

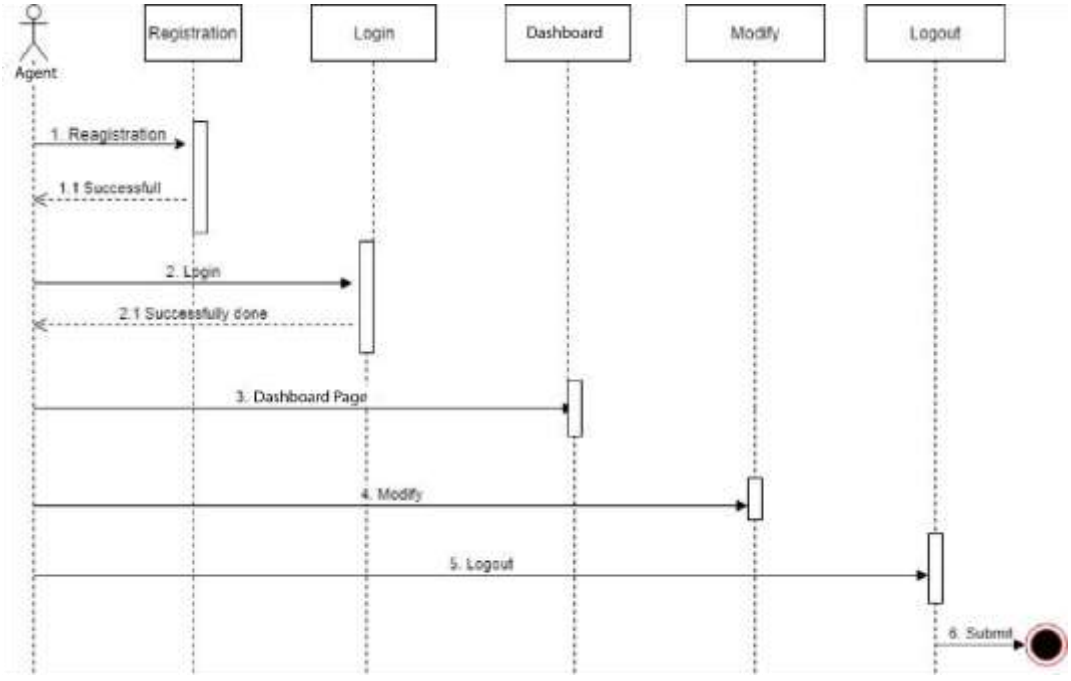


Figure 4.2: Sequence diagram for agent

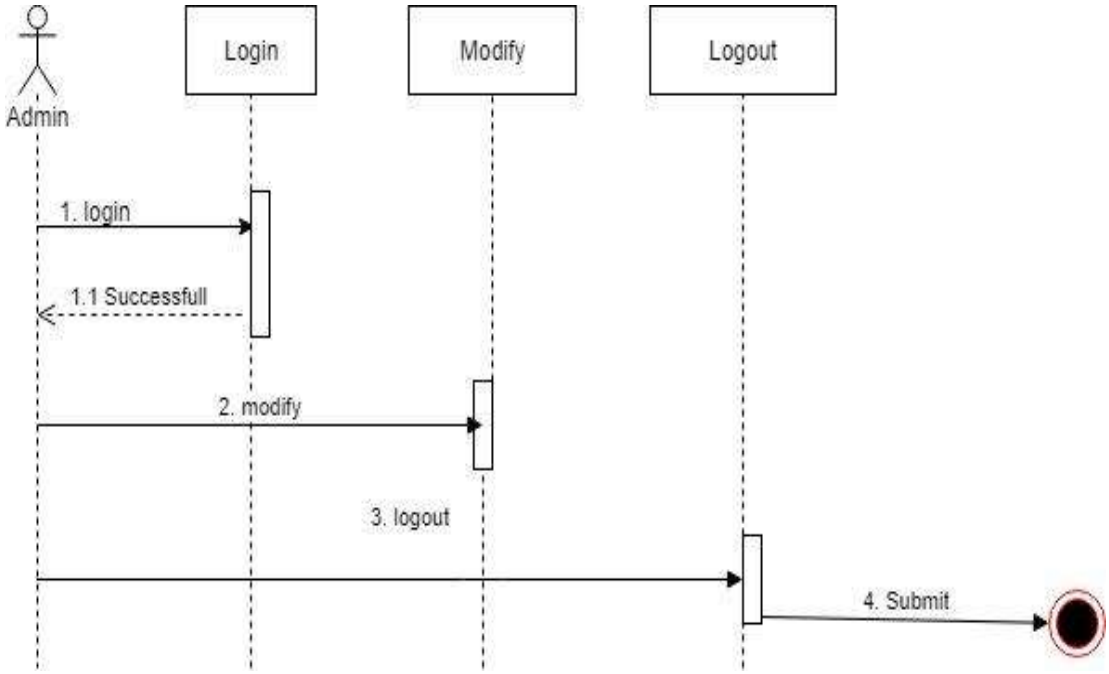


Figure 4.3: Sequence diagram for admin

3.5 ER-Diagram

An entity-relationship diagram, or ERD, is a chart that visually represents the relationship between database entities. ERDs model an organization's data storage requirements with three main components: entities, attributes, and relationships.

Entity: A data entity is anything real or abstract about which we want to store data. Entity types fall into five classes: roles, events, locations, tangible things or concepts. E.g. employee, payment, campus, book. Specific examples of an entity are called instances. E.g. the employee John Jones, Mary Smith's payment, etc.

Relationship: A data relationship is a natural association that exists between one or more entities. E.g. Employees process payments. Cardinality defines the number of occurrences of one entity for a single occurrence of the related entity. E.g. an employee may process many payments but might not process any payments depending on the nature of her job.

Attribute: A data attribute is a characteristic common to all or most instances of a particular entity. Synonyms include property, data element, field. E.g. Name, address, Employee Number, pay rate are all attributes of the entity employee. An attribute or combination of attributes that uniquely identifies one and only one instance of an entity is called a primary key or identifier. E.g. Employee Number is a primary key for Employee.

Uses of entity relationship diagrams:

Database design: ER diagrams are used to model and design relational databases, in terms of logic and business rules (in a logical data model) and in terms of the specific technology to be implemented (in a physical data model.) In software engineering, an ER diagram is often an initial step in determining requirements for an information systems project. It's also later used to model a particular database or databases. A relational database has an equivalent relational table and can potentially be expressed that way as needed.

Database troubleshooting: ER diagrams are used to analyze existing databases to find and resolve problems in logic or deployment. Drawing the diagram should reveal where it's going wrong.

Business information systems: The diagrams are used to design or analyze relational databases used in business processes. Any business process that uses fielded data involving entities, actions and interplay can potentially benefit from a relational database. It can streamline processes, uncover information more easily and improve results.

Business process re-engineering (BPR): ER diagrams help in analyzing databases used in business process re-engineering and in modeling a new database setup.

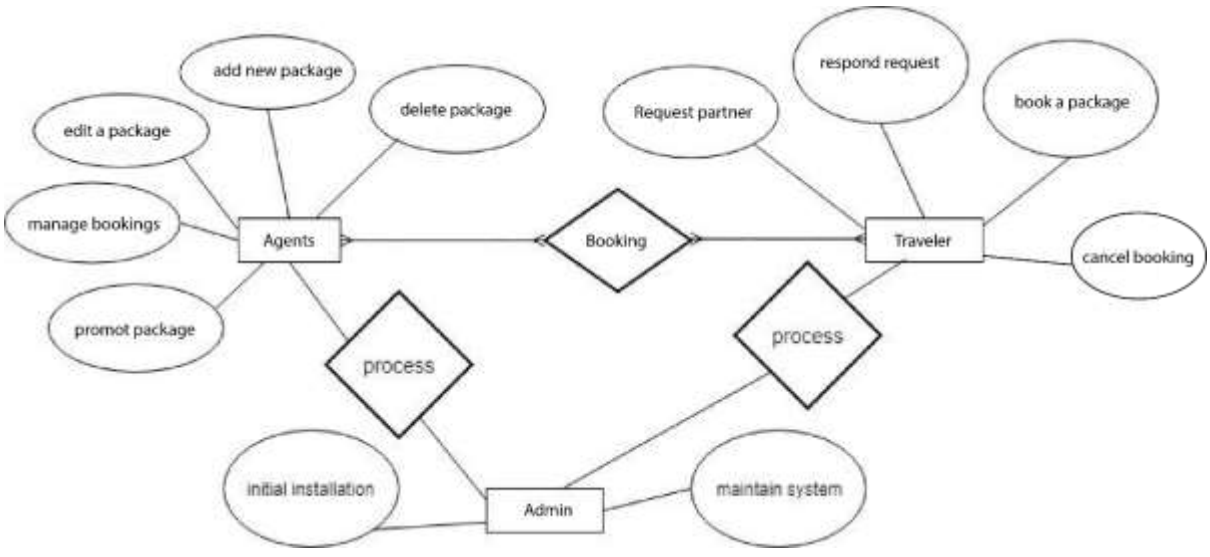


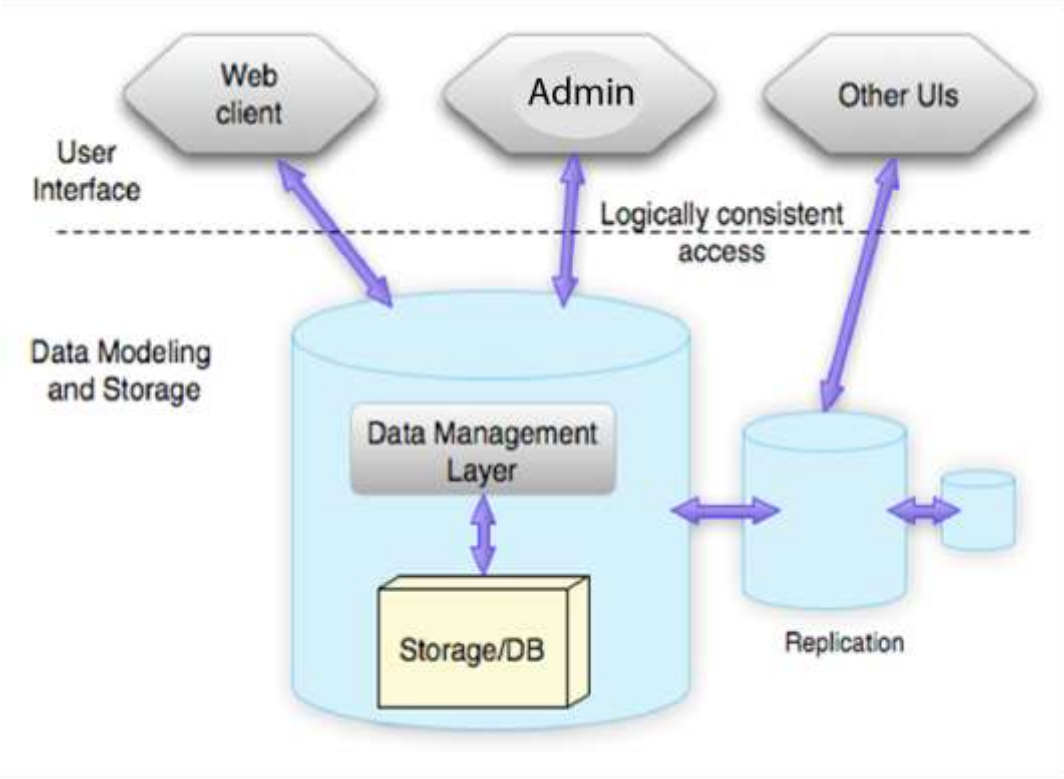
Figure: E-R diagram system

3.6 Database Design

Database design is the organization of data according to a database model. The designer determines what data must be stored and how the data elements interrelate. With this information, they can begin to fit the data to the database model. Database design involves mostly classifying data and identifying the interrelationships. A properly designed database provides you with access to up-to-date, accurate information. Because a correct design is essential to achieving your goals in working with a database, investing the time required to learn the principles of good design makes sense. In the end, you are much more likely to end up with a database that meets your needs and can easily accommodate change. Certain principles guide the database design process. The first principle is that duplicate information (also called redundant data) is bad, because it wastes space and increases the likelihood of errors and inconsistencies. The second principle is that the correctness and completeness of information is important. If your database contains incorrect information, any reports that pull information from the database will also contain incorrect information. As a result, any decisions you make that are based on those reports will then be misinformed.
The design process consists of the following steps:

- Determine the purpose of your database
- Find and organize the information required
- Divide the information into tables
- Turn information items into columns
- Specify primary keys
- Set up the table relationships
- Refine your design
- Apply the normalization rules

In our routine application we are using very popular and so-called No-SQL database ‘MongoDB’. MongoDB is an open-source NoSQL database. MongoDB is a document-based database. MongoDB is one of the leading NoSQL databases. NoSQL database is a type of non-relational database and it is capable of processing structured, semi-structured and unstructured data. MongoDB is a NoSQL database written in C++ language. Some of its drivers use the C programming language as the base. MongoDB is a document-oriented database where it stores data in collections instead of tables. The best part of MongoDB is that the drivers are available for almost all the popular programming languages.
In today's competitive technological world, every company has started hosting its enterprise applications over the cloud in order to expand the business globally, provide faster services and to personalize the customer's experience with the application and overall business. And NoSQL has become the first choice in database technology for developing such applications.
MongoDB is definitely a promising solution. It may be very useful in creating applications like bug tracking, discussion forums, advertisements, and the like. However, it may not be able to cater to all the needs that a relational database support. Joins are not possible; this is an advantage in relational DB. It requires proper analysis before making a decision.


Web client
Admin
Other UIs
User Interface
Logically consistent access
Data Modeling and Storage
Data Management Layer
Storage/DB
Replication

CHAPTER-4
DESIGN IMPLEMENTATION

4.1 Tools

JavaScript, React.js, Node.js, Express.js, MongoDB, React Native, RESTful API, Webpack, Mongoose ORM, Material UI, Heroku Hosting CLI, Expo.io, Moment.js and last but not least HTML5 & CSS3.

JavaScript

The most common use of JavaScript is to add client-side behavior to HTML pages, also known as Dynamic HTML (DHTML). Scripts are embedded in or included from HTML pages and interact with the Document Object Model (DOM) of the page. JavaScript, often abbreviated as JS, is a high-level, just-in-time compiled, multi-paradigm programming language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions

React.js

React is a JavaScript library for building user interfaces. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications.

Node.js

Node.js is an open-source and cross-platform JavaScript runtime environment. Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant. A Node.js app is run in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.

Express.js

Express.js, or simply Express, is a web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js.

MongoDB

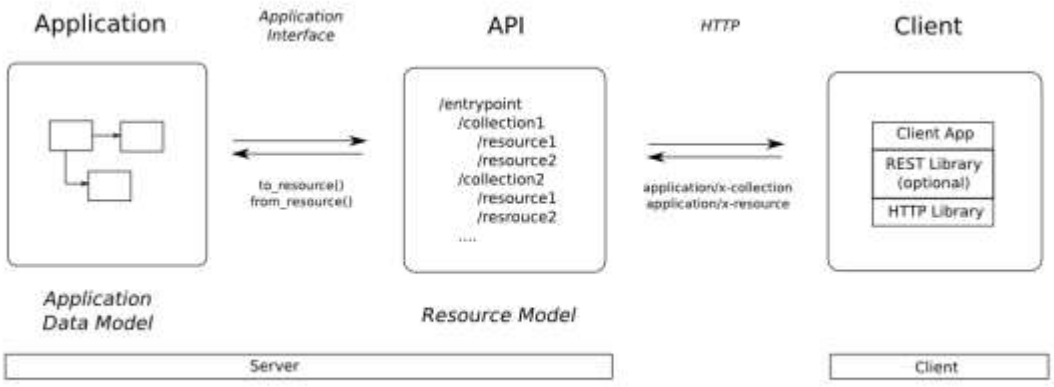
MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema. MongoDB is developed by MongoDB Inc. and licensed under the Server-Side Public License.

React Native (RN)

React Native is an open-source mobile application framework created by Facebook. It is used to develop applications for Android, iOS, Web and UWP by enabling developers to use React along with native platform capabilities. An incomplete port for Qt also exists.

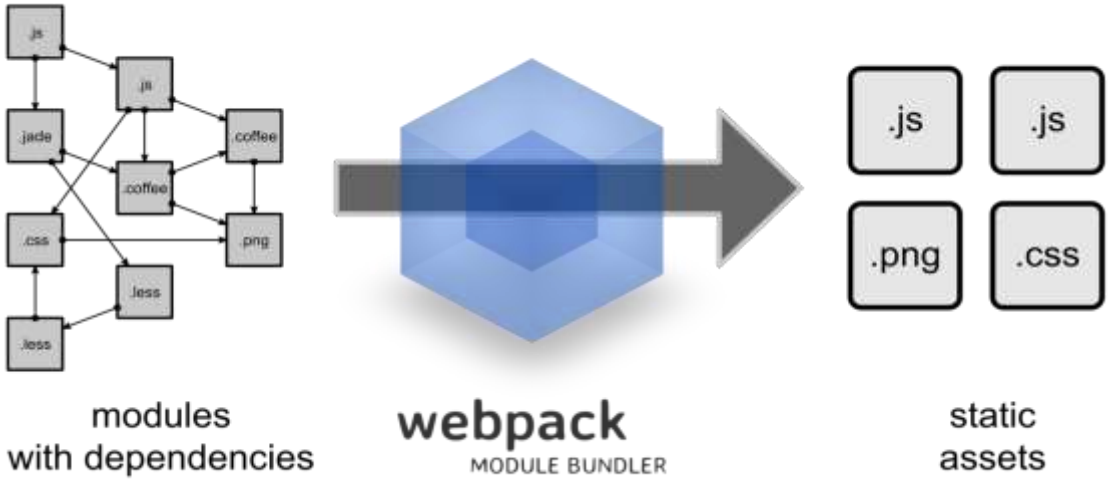
Restful API

A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data. A RESTful API -- also referred to as a RESTful web service or REST API -- is based on representational state transfer (REST) technology, an architectural style and approach to communications often used in web services development. REST technology is generally preferred to the more robust Simple Object Access Protocol (SOAP) technology because REST leverages less bandwidth, making it more suitable for internet usage. An API for a website is code that allows two software programs to communicate with each another. The API spells out the proper way for a developer to write a program requesting services from an operating system or other application. This video by Kevin Babcock details RESTful API design and related HTTP concepts. The REST used by browsers can be thought of as the language of the internet. With cloud use on the rise, APIs are emerging to expose web services. REST is a logical choice for building APIs that allow users to connect and interact with cloud services. RESTful APIs are used by such sites as Amazon, Google, LinkedIn and Twitter. How RESTful APIs work A RESTful API breaks down a transaction to create a series of small modules. Each module addresses a particular underlying part of the transaction. This modularity provides developers with a lot of flexibility, but it can be challenging for developers to design from scratch. Stateless components can be freely redeployed if something fails, and they can scale to accommodate load changes. This is because any request can be directed to any instance of a component; there can be nothing saved that has to be remembered by the next transaction. That makes REST preferred for web use, but the RESTful model is also helpful in cloud services because binding to a service through an API is a matter of controlling how the URL is decoded.



Webpack

webpack is an open-source JavaScript module bundler. It is a module bundler primarily for JavaScript, but it can transform front-end assets like HTML, CSS, and images if the corresponding loaders are included. webpack takes modules with dependencies and generates static assets representing those modules.



Mongoose ORM

Mongoose is similar to an ORM (Object-Relational Mapper) you would use with a relational database. Mongoose is an ODM that provides a straightforward and schema-based solution to model your application data on top of MongoDB's native drivers.

Material UI

React components for faster and easier web development. It also provides React Native component for productive React Native development. It has some useful pre-styled good-looking set of reusable components that can be easily integrate with any web or mobile user interface.

Heroku Hosting CLI

Heroku is a cloud platform as a service (PaaS) supporting several programming languages. ... server handles application repository pushes from permitted users. All Heroku services are hosted on Amazon's EC2 cloud-computing platform.

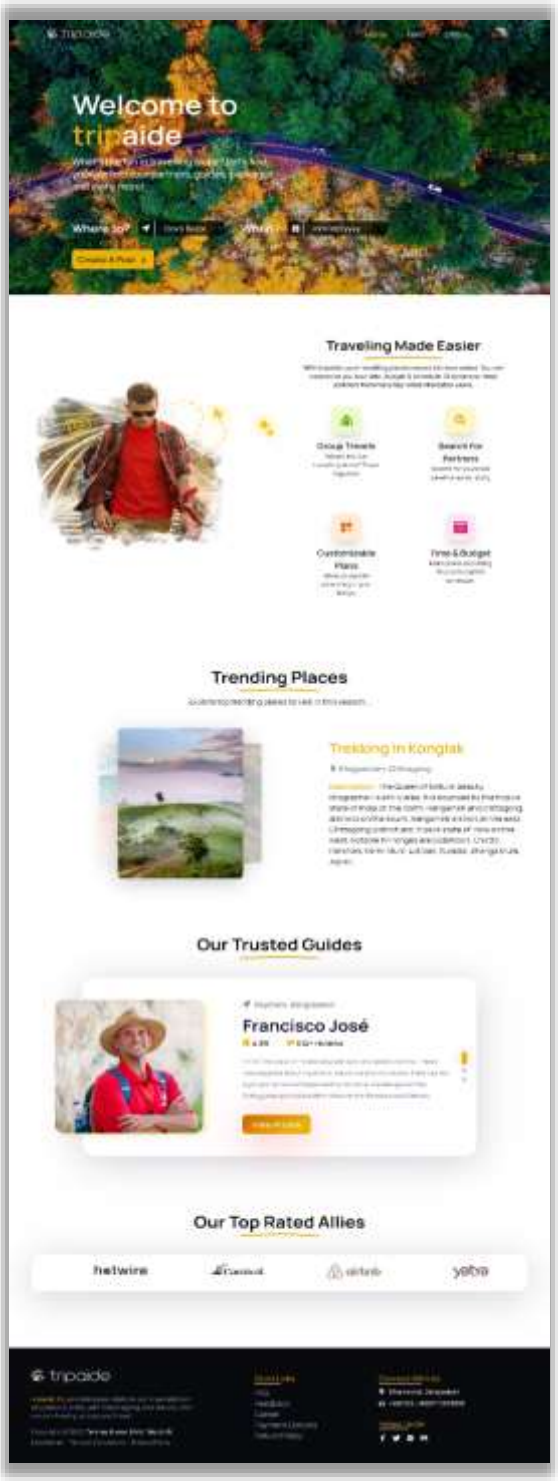
Expo.io

Expo is an open-source platform for making universal native apps for Android, iOS, and the web with JavaScript and React.

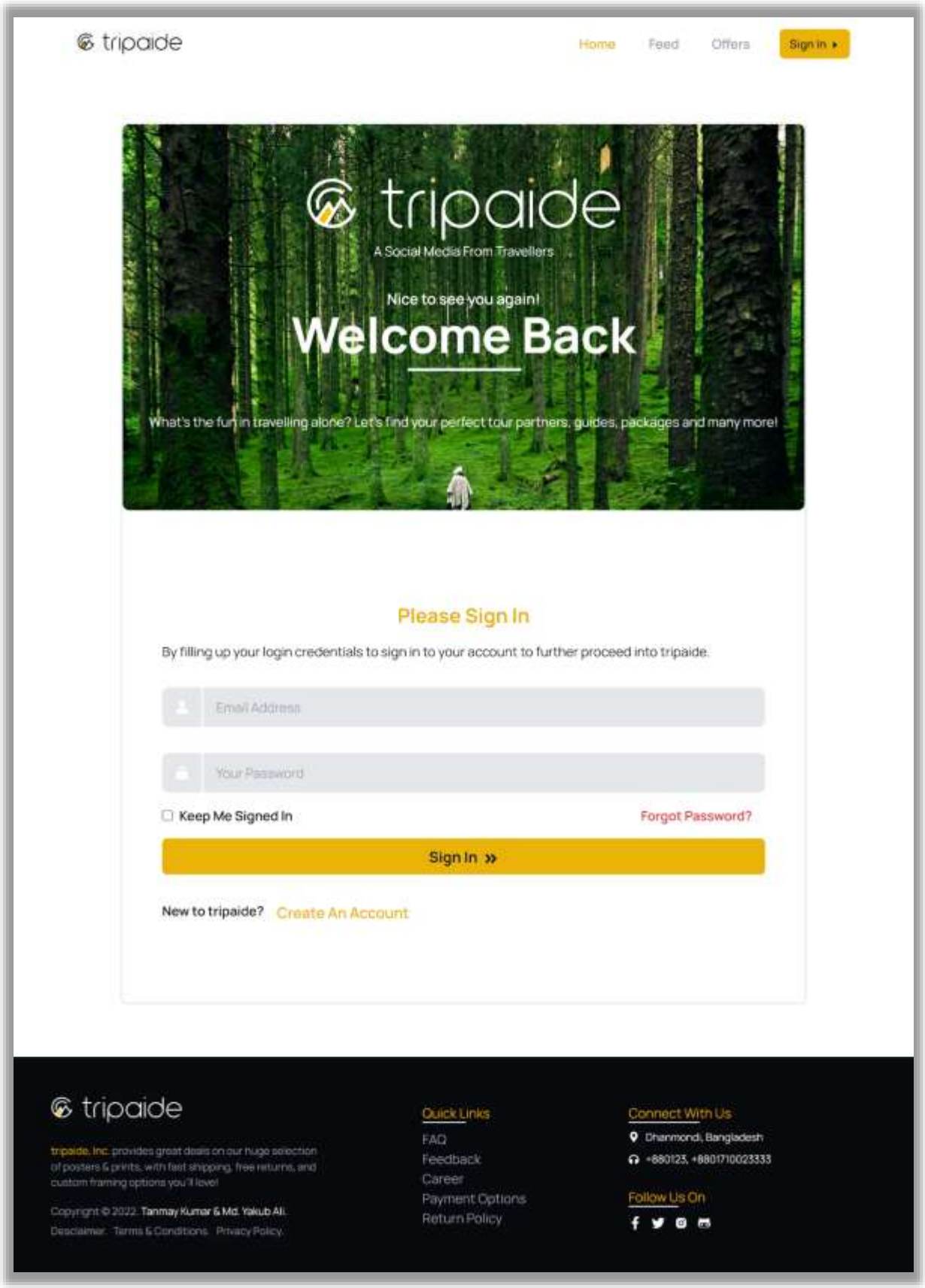
Moment.js

Moment.js is a free and open source JavaScript library that removes the need to use the native JavaScript Date object directly. The library is a wrapper for the Date object (in the same way that jQuery is a wrapper for JavaScript) making the object a whole lot easier to work with.

4.2 Homepage



4.3 Users Login Process



4.4 Users Registration Process

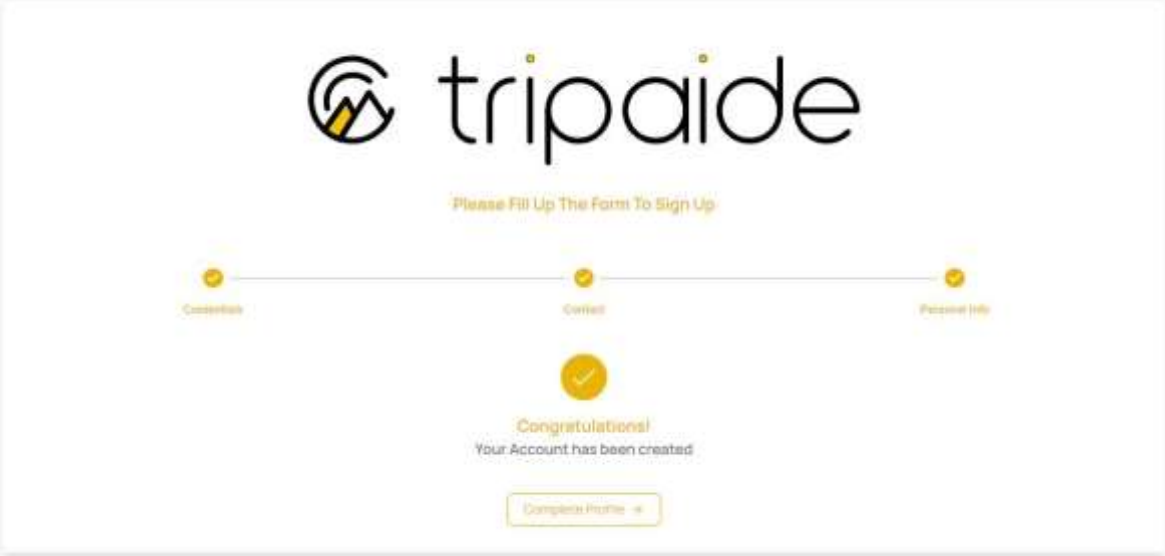
Step 1

Step 2

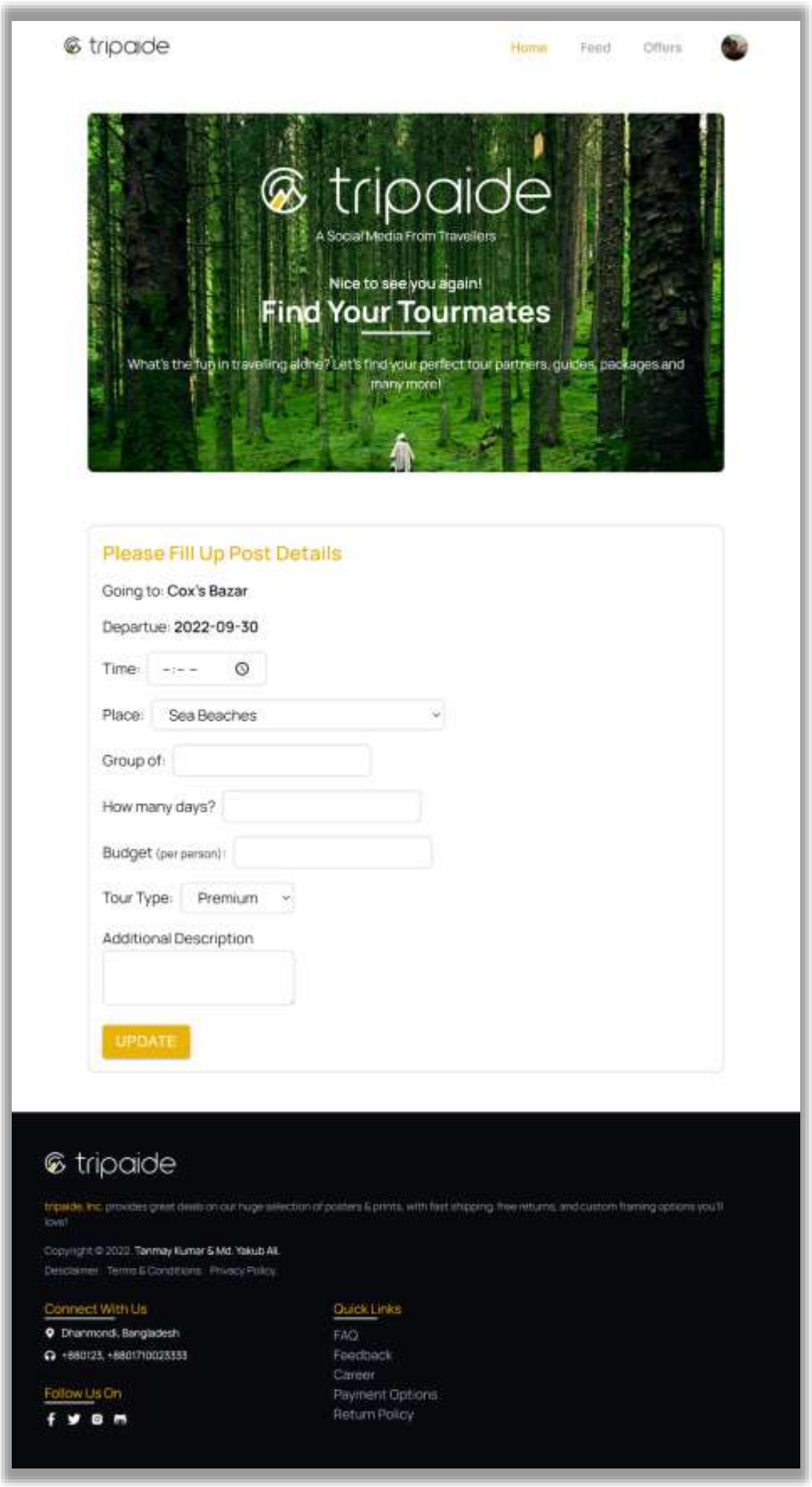
Step 3


tripaide
Please Fill Up The Form To Sign Up
1234567890
07/21/1999
1234

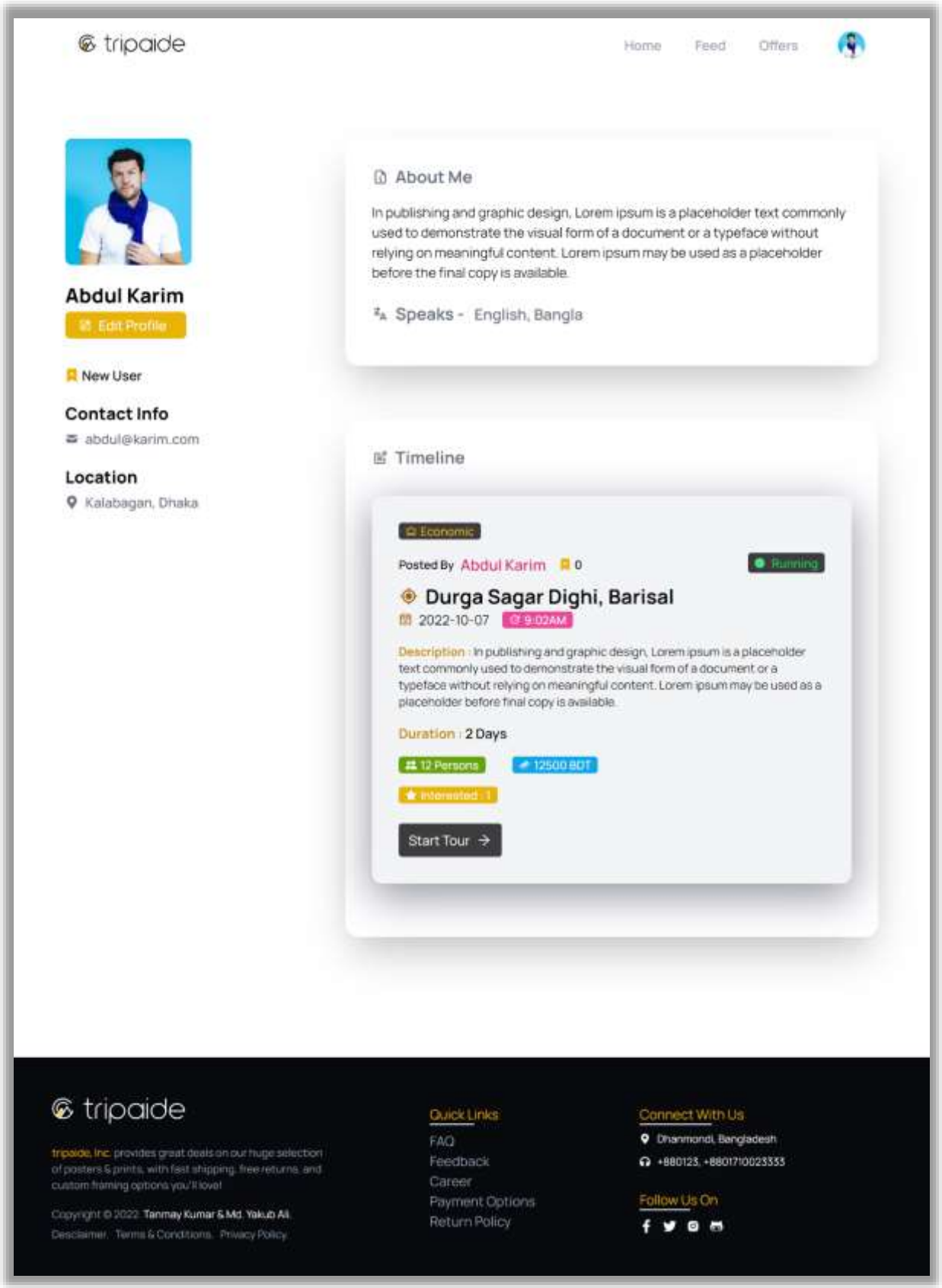
Step 4


tripaide
Please Fill Up The Form To Sign Up
Congratulations!
Your Account has been created

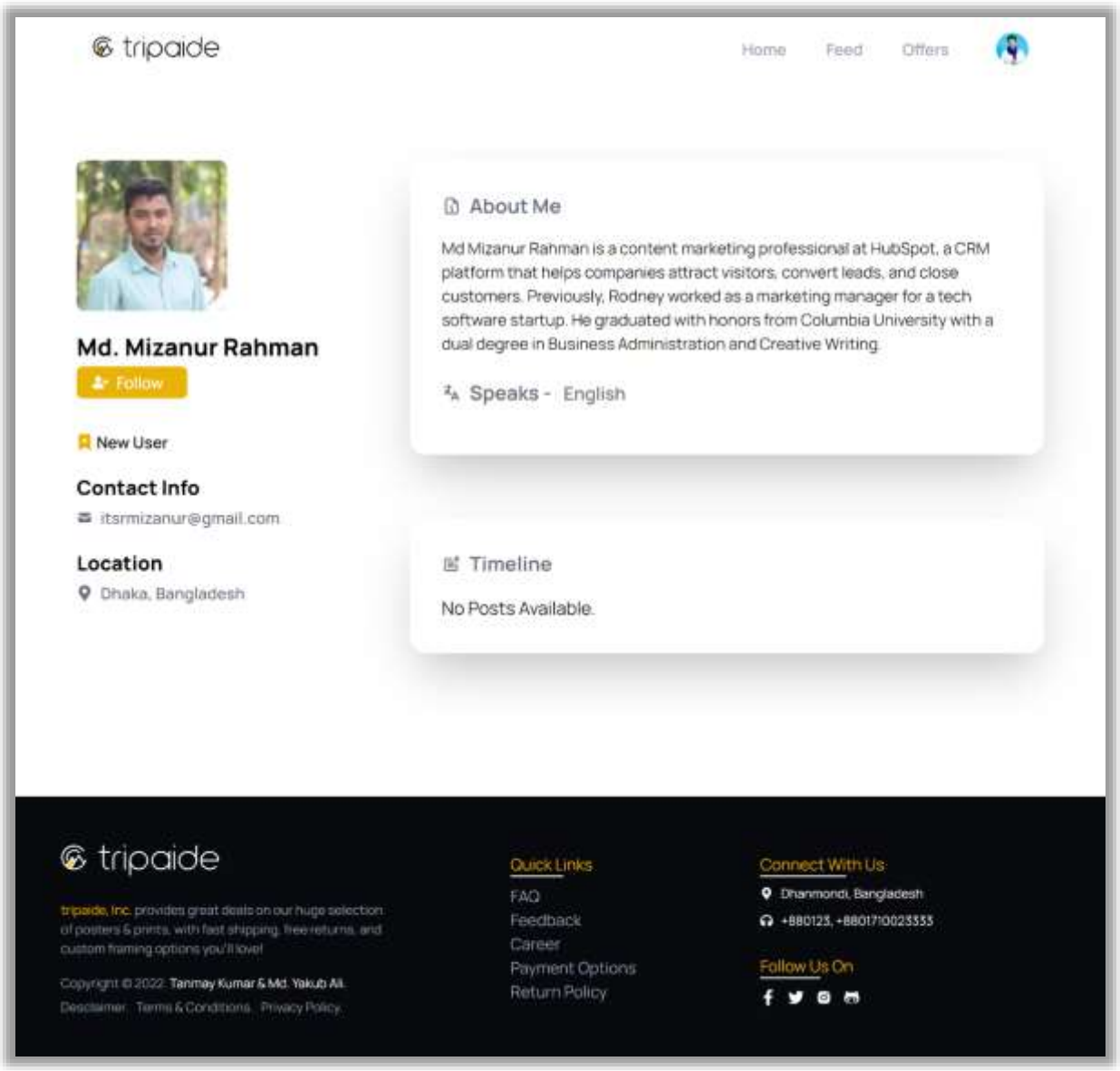
4.5 Users Information Management



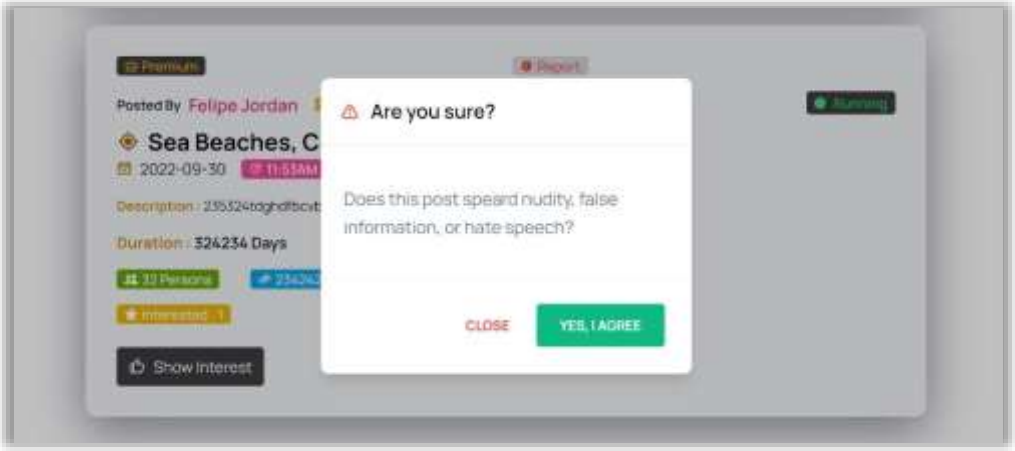
4.6 Users Information Management



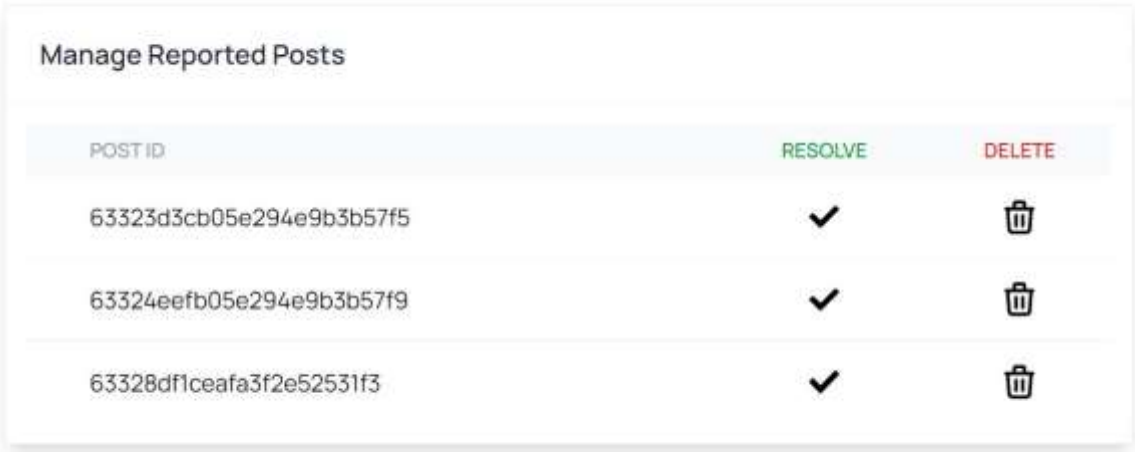
4.7 Other User's Profile



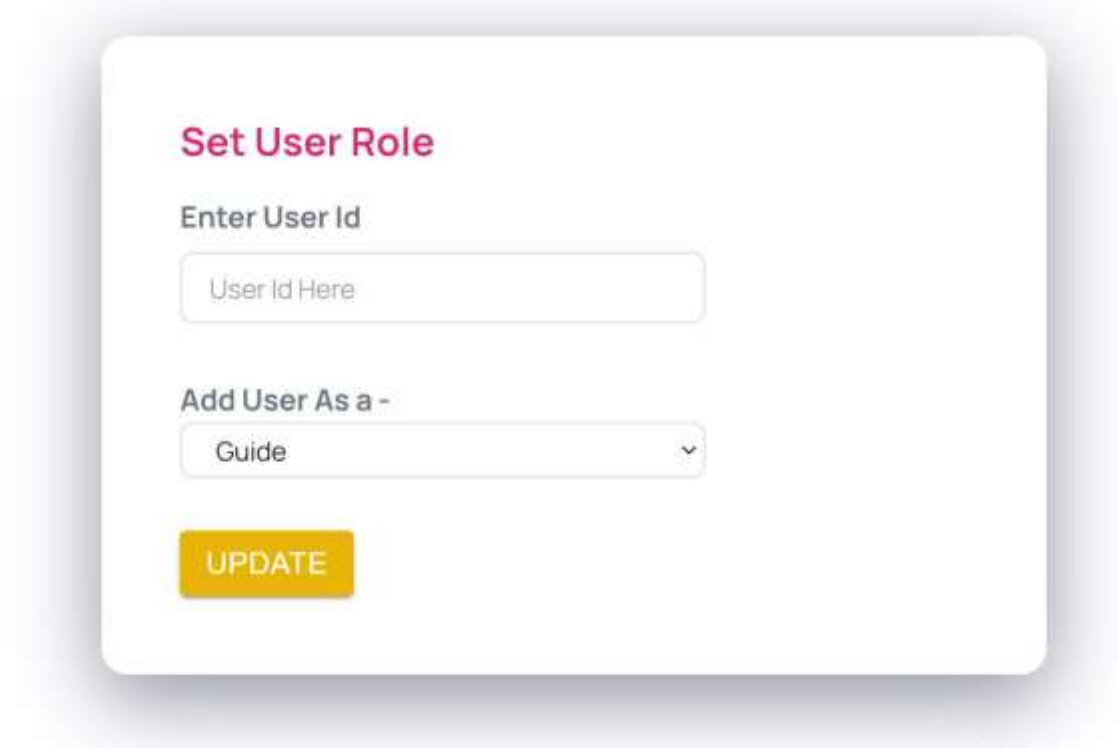
4.8 Post Reporting to Admin



4.9 Reported Post Management



4.10 User Role Change



CHAPTER-5
SYSTEM TESTING PLANNING

5.1 Testing Features

A Software feature can be strong as the changes made in the system to add new functionality or adapt the current functionality. All features are supposed to have characteristics that are designed to be useful, intuitive and effective.

In realism, a new test set is shaped for testing that feature consistent to that cycle of that announcement. The tremendously significant and usually used new features must to be tested methodically in each build of that release and also reversion testing should be done pertinent to those areas.

5.2 Features to be tested

Features	Priority	Description
Registration	1	Get all service from this system, it is required to be registered.
Modify	3	Edit the information when need.
Create Post	2	Create a new post of their travel plan.
Show Details	1	All posted information will show in the page.
Login	1	Login as authenticated user.
Logout	1	Logout from the system.
Report Post	2	A post can be report if it contains nasty contents.
Error Message	1	It is important for all to get the proper error message.
Technological Features		
Database	1	Entry to database is often wanted process. So this technical feature should be firmly in control from management system.

5.3 Testing Strategy

A test strategy is a plan that defines the testing method of the software development cycle. It is created to notify project managers, testers, and developers about some important subjects of the testing process. They are created based on development design documents.

5.4 Testing Approach

Test approach is the test plan application of a project, describes how testing would be approved available.

Black Box Testing

Black box testing also called functional testing that ignores the internal mechanism of a system or component and focuses on the outputs generated in response to selected inputs and execution conditions. We have decided to perform equivalence partitioning and Boundary value analysis for this system.

White Box Testing

White box testing is a software testing method in which the internal structure or implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential.

5.5 Pass/Fail Criteria

The arrival criteria for all stage of testing must be encountered beforehand the following stage can begin. Currently the criteria for pass and fail are given below.

- Rendering to the assumed situation the predictable result need to take place then the situation will be careful as pass then that standards should be failed.
- If an item tested 10 times, 9 times flawlessly worked and single time do not work correctly then it will reflect as fail case.
- System crash will be careful as fail case.
- Afterward succumbing inquiry in the system, if predictable page won't seem then it will be careful as fail case.

5.6 Testing Schedule

This section will describe testing schedule.

- Stipulate test signs.
- Stipulate entirely article transmitted proceedings.
- Approximation time obligatory to do each testing mission.
- Schedule entirely testing tasks and test signs
- For each testing supply, stipulate its retro of use.

Test Phase	Time
Test Plan Creation	5 days
Test specification creation	5 days
Unit Testing	During Development time
Component testing	3 days
Test Phase	**Time**
Integration Testing	5 days
Use case validation	2 days
User interface testing	3 days
Load testing	2 days
Performance Testing	3 days
Release to Production	2 days

5.7 Testing Environment

Testing environment is a arrangement of software and hardware for the testing teams to perform test cases. In additional words, it ropes test implementation with hardware, software and network configured.

For test environment, key part to set up comprises,

- System and applications
- Test data
- Database server
- Front end running environment
- Client operating system
- Browser
- Hardware includes Server Operating system
- Network
- Documentation required like reference, documents, configuration guides, installation guides and user manuals.

CHAPTER-6
LIMITATION & FUTURE WORK

6.1 Limitation

There are some limitations in the current design of our system. Some of them are listed below,

- There is no system admin feature who can control whole infrastructure.
- No mail service is integrated.
- No module to manage student information.
- There is no customizable feature or user interface.
- Currently teacher can modify his/her profile except media content.
- There is no fallback if system failure occurs.

Also, in our ER-Diagram we added attendance module but in the design section it is not implemented yet.

6.2 Future Enhancements

Such a desktop-based platform has immense potential for future growth not only for students but also for faculty as well as department concerned in improving upon their efficiency of work with no or less burden. Some of the areas where future scope lies and if implemented would help a great deal to the students at large.

In the Future,

- New improvement on the mobile client application in terms of user interface.
- New functionality in the admin administration page will be added.
- Admin content management system will be mobile responsive.
- Mailing system module or functionality will be added so that when the routine data change students will get a mail.
- There will a history system so that admin can revert back the changes.
- Security will be improved.
- We will add a new module to maintain attendance data.

CHAPTER-7
CONCLUSION

This report is about the project "**Online Based Travel Partner and Package Finder**" also can be called as **"tripaide"** that provides us simple interface to find a suitable trip package or travel partner for one's dream vacation. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting relevant information may be very time consuming. All these problems are expected to be solved using this system.

This project provides facilities to the travelers to find tour partner to minimize their tour cost and also the travel & tour companies to provide better service with a very interactive content management system to easily access the information etc.

REFERENCES

8.1 Book Reference

1. Express.js in Action, Manning Publications, April 2016.
2. Getting MEAN with Mongo, Express, Angular, and Node, Manning Publications, early 2015.
3. Getting MEAN with Mongo, Express, Angular, and Node, Second Edition, Manning Publications, April 2017.
4. Pro Express.js: Master Express.js: The Node.js Framework For Your Web Development, Apress, December 2014.
5. Mastering Web Application Development with Express, Packt Publishing, September 2014.
6. Web Development with Node and Express, O'Reilly Media, July 2014.
7. Node.js in Action, Manning Publications, October 2013.
8. Express Web Application Development, Packt Publishing, June 2013.

8.2 Website Reference

1. https://nodejs.org/docs/latest-v11.x/api/
2. https://reactjs.org/
3. https://facebook.github.io/react-native/
4. https://expo.io/
5. https://www.mongodb.com/
6. https://www.heroku.com
7. https://momentjs.com/
8. https://expressjs.com/en/4x/api.html