

FULL-STACK PROJECT

(2021-22)

Building and Deployment of Web application

TRAVELYAARI

PROJECT REPORT



Institute of Engineering & Technology

Submitted by: -

Satish Jha(191500728)

Ayushi Gupta (191500205)

Vartika Saxena(191500896)

Sneha Banga(191500813)

Supervised By: -

Mr. Mandeep Singh

Technical Trainer

Department of Computer Engineering & Applications

GLA University

Mathura- 281406,

INDIA



Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,

DECLARATION

I/we hereby declare that the work which is being presented in the B.Tech. Project “**TRAVELYAARI**”, in partial fulfillment of the requirements for the award of the Bachelor of Technology in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of Mr. Mandeep Singh.

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Sign_____

Name : Satish Jha

University Roll No.: 191500728

Sign_____

Name: Ayushi Gupta

University Roll No.:191500205

Sign_____

Name : Vartika Saxena

University Roll No.: 191500896

Sign_____

Name: Sneha Banga

University Roll No.:191500813

Date:



Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,

CERTIFICATE

This is to certify that the above statements made by the candidate are correct and true to the best of my knowledge and belief.

Signature of Supervisor:

Mr. Mandeep Singh

Assistant Professor

Institute of Engineering & Technology

GLA University

Date:

ACKNOWLEDGEMENT

We would like to express our deep and sincere gratitude to our college faculties for giving us this opportunity to do a full-stack project. We are extremely grateful to my mentor, Mr. Mandeep Singh, for his invaluable guidance throughout this project. His dynamism, vision, sincerity and motivation have deeply inspired us. He has guided us so well. It was a great privilege and honor to work and study under his guidance. We are extremely grateful for what he has offered us. I would also like to thank him for his empathy. We are extremely thankful to our friends and family for their acceptance and patience during this mini project. We are extremely grateful to our parents for their love, prayers, caring and sacrifices for educating and preparing us for my future.

Sign_____

Name : Satish Jha

University Roll No.: 191500728

Sign_____

Name: Ayushi Gupta

University Roll No.:191500205

Sign_____

Name : Vartika Saxena

University Roll No.: 191500896

Sign_____

Name: Sneha Banga

University Roll No.:191500813

ABSTRACT

This work deals with the development of web application for a travelling website TravelYaari. This application is developed for knowledge purpose allowing the user to choose the location he or she wants to travel and gather information about that place beforehand.

For the development of this project, data is collected from different sources about the places particularly in India. This application will make use of MERN stack technology.

The user will see images and information about the place on his screen and can choose the location on the basis of that information. All the locations are divided into six categories based on its type and are displayed accordingly to the user.

TABLE OF CONTENTS

DECLARATION.....	2
CERTIFICATE.....	3
ABSTRACT.....	5
1. INTRODUCTION.....	1
INTRODUCTION	1
OBJECTIVE.....	1
MOTIVATION.....	2
FEATURES	2
TECHONOLOGIES USED	2
MongoDB	2
Express JS.....	4
React.js	5
Node.js.....	6
WHAT IS A WEB APPLICATION (WEB APP).....	7
How a web application works.....	8
Introduction to Dependencies	9
SCOPE	10
2. SOFTWARE REQUIREMENT ANALYSIS	11
INTRODUCTION	11
Purpose	11
Document Convention	11
Intended Audience	11
Definitions, Acronyms, and Abbreviations.....	12
PERSPECTIVE	12
PRODUCT FUNCTIONS	12
USER CLASSES AND CHARACTERISCTICS.....	12
OPERATING ENIRNOMENT	12
DESIGN CONSTRAINS	13
ASSUMPTIONS AND DEPENDENCIES	13
PERORMANCE REQUIREMENTS.....	13

Hardware Requirements	13
Software Requirements	14
COMMUNICATION PROTOCOLS AND INTERFACES	14
3. SOFTWARE DESIGN.....	15
DATAFLOW DIAGRAM.....	15
DFD Level 0	15
DFD Level -1	15
DFD Level 2	16
DFD Level 2 User	17
USE CASE DIAGRAM.....	18
Sequence Diagram	19
CLASS DAGRAM	20
DATABASE DESIGN.....	21
User Database Collection.....	21
Categories Database Collection	22
Products Database Collection	23
Order Database Collection	25
4. IMPLEMENTATION AND USER INTERFACACES	26
TRAVELYAARI	26
TRAVELYAARI UI	29
5. CONCLUSION AND FUTURE WORK.....	39
Conclusion.....	39
Scope of Further Development	39
6. REFERENCES.....	40

LIST OF FIGURES

Figure 2:Overview of monodb	4
Figure 3: Overview of Node JS.....	7
Figure 4: Working of Web App	9
Figure 5: DFD Level 0.....	15
Figure 6:DFD Level 1 Admin Side	16
Figure 8: DFD Level 2 Admin.....	17
Figure 9: DFD Level 2 User.....	18
Figure 10: UseCase Diagram	19
Figure 11: Sequence Diagram.....	20
Figure 12: Class Diagram	21
Figure 13: Category Schema	22
Figure 14: Users Schema	23
Figure 15: Products Schema.....	24
Figure 16:Order Collection	25
Figure 17: Server Directory.....	26
Figure 18: Data Flow	28
Figure 19:Home Page Without Signin.....	29
Figure 20: Home page After Signin	29
Figure 21: SignIn Page.....	30
Figure 22: SignUp Page	30
Figure 23:Footer section	31
Figure 24:Admin Dashboard.....	31
Figure 25: Add new Category page.....	32
Figure 26:Add Place page	32
Figure 27: popular places	33
Figure 28: Manage places Page.....	33
Figure 29: User Dashboard	34
Figure 30: Places Page	34
Figure 31: related places	35
Figure 32:Filter places	35
Figure 33: Wishlist Page	36

Figure 34: Payment by Card.....	36
Figure 35: Payment Successful and Card Empty.....	37
Figure 36: Payment by paypal	37
Figure 37: paypal Integration.....	38

1. INTRODUCTION

INTRODUCTION

This project is designed to build a web-based application of a travelling website for the people, where someone can search and book the desirable he or she wants to visit in holidays.

It is a simple, knowledgeable website which is useful in gaining a lot of information about any destination user is planning to visit.

This mainly focuses on the destinations located in India. India is a beautiful country having lot of amazing places to visit and it is a place where each and every city has its own flavor and culture. There are a lot of things to do here.

India is a vibrant land of startling contrasts where both the traditional and modern worlds meet. The world's seventh largest nation by area and the second largest in terms of population, India boasts a rich heritage that's the result of centuries of different cultures and religions leaving their mark. Things to do for travelers include the opportunity to experience an array of sacred sites and spiritual encounters, while nature lovers will enjoy its sun-washed beaches, lush national parks, and exciting wildlife sanctuaries.

From the magnificent Taj Mahal in Agra to the holy sites of Harmandir Sahib in Amritsar and the Mecca Masjid mosque in Hyderabad, visitors to this exotic country will discover a trove of spiritual, cultural, and historical treasures.

OBJECTIVE

The main objective of the project is to create a website named “TRAVELYAARI” using MERN stack technologies to continuously provide enjoyable quality excursions/trips on time and on budget. It will also develop enthusiastically satisfied customers all of the time.

MOTIVATION

*“The World is a book and those who do not **travel** read only a page.”*

~ Saint Augustine.

The motivation behind building this web project is that almost all of us want to travel and visit new places but in so many cases either due to lack of information or a busy lifestyle we end up going nowhere and wasting a brilliant chance of having a brand-new experience.

So in order to let people find an easy get away from their hectic schedule, get a bit of freshness in the environment and uplift their travelling experience, this web application is built.

FEATURES

- Full featured Wishlist
- Product pagination
- User profile with orders
- Admin management
- Admin user management
- Admin Order details page
- Checkout process (payment method, availability, review ratings etc.)
- PayPal / credit card integration
- Database (places & users)

TECHONOLOGIES USED

MongoDB

MongoDB is source-available cross-platform document-oriented database program. Classified as NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server-Side Public License (SSPL).

MongoDB supports field, range query, and regular expression searches. Queries can return specific fields of documents and also include user-defined JavaScript functions. Queries can also be configured to return a random sample of results of a given size.

Fields in a MongoDB document can be indexed with primary and secondary indices.

MongoDB provides high availability with replica sets. A replica set consists of two or more copies of the data. Each replica-set member may act in the role of primary or secondary replica at any time. All writes and reads are done on the primary replica by default. Secondary replicas maintain a copy of the data of the primary using built-in replication. When a primary replica fails, the replica set automatically conducts an election process to determine which secondary should become the primary. Secondaries can optionally serve read operations, but that data is only eventually consistent by default.

If the replicated MongoDB deployment only has a single secondary member, a separate daemon called an *arbiter* must be added to the set. It has a single responsibility, which is to resolve the election of the new primary. As a consequence, an idealized distributed MongoDB deployment requires at least three separate servers, even in the case of just one primary and one secondary.

MongoDB can be used as a file system, called GridFS, with load balancing and data replication features over multiple machines for storing files. This function, called grid file system, is included with MongoDB drivers.

MongoDB exposes functions for file manipulation and content to developers. GridFS can be accessed using mongo files utility or plugins for Nginx and httpd. GridFS divides a file into parts, or chunks, and stores each of those chunks as a separate document.

JavaScript can be used in queries, aggregation functions (such as MapReduce), and sent directly to the database to be executed.



Figure 1: Overview of monodb

Express.JS

Express.js, or simply Express, is a backend 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.

The original author, TJ Holowaychuk, described it as a Sinatra-inspired server, meaning that it is relatively minimal with many features available as plugins. Express is the back-end component of popular development stacks like the MEAN, MERN or MEVN stack, together with the MongoDB database software and a JavaScript front-end framework or library.

Express.js was founded by TJ Holowaychuk. The first release, according to Express.js's GitHub repository, was on the 22nd of May, 2010. Version 0.12. In June 2014, rights to manage the project were acquired by StrongLoop. StrongLoop was acquired by IBM in September 2015; in January 2016, IBM announced that it would place Express.js under the stewardship of the Node.js Foundation incubator.

Features

- **Web Applications:** Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.
- **APIs:** With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy.

- **Performance:** Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love.
- **Frameworks:** Many popular frameworks are based on Express.

React.js

React (also known as React.js or ReactJS) is an open source, front-end, Javascript library for building user interfaces or UI components. 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. However, React is only concerned with state management and rendering that state to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality.

React is a JavaScript library and React applications built on it run in the browser, NOT on the server. Applications of this kind only communicate with the server when necessary, which makes them very fast compared to traditional websites that force the user to wait for the server to re-render entire pages and send them to the browser.

React is used for building user interfaces - what the user sees on their screen and interacts with to use your web app. This interface is split up into components, instead of having one huge page you break it up into smaller pieces known as components. In more general terms, this approach is called Modularity.

It's declarative: React uses a declarative paradigm that makes it easier to reason about your application.

It's efficient: React computes the minimal set of changes necessary to keep your DOM up-to-date.

And it's flexible: React works with the libraries and frameworks that you already know.

React is Declarative, for the most part, which means we are concerned more with what to do rather than how to do a specific task. Imperative code approach instructs JavaScript on how it should perform each step. With declarative code approach, we

tell JavaScript what we want to be done, and let JavaScript take care of performing the steps.

Declarative programming is a programming paradigm that expresses the logic of a computation without describing its control flow. Declarative programming comes with certain advantages such as reduced side effects (occurs when we modify any state or mutate something or make an API request), minimized mutability (as a lot of it is abstracted), enhanced readability, and fewer bugs.

React also has unidirectional dataflow. UI in React is actually the function of the state. This means that as the state updates it updates the UI as well. So our UI progresses as the state changes.

Node.js

Node.js is an open source, cross platform, Backend Javascript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web-page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts.

Node.js was written initially by Ryan Dahl in 2009, about thirteen years after the introduction of the first server-side JavaScript environment, Netscape's LiveWire Pro Web. The initial release supported only Linux and Mac OS X. Its development and maintenance was led by Dahl and later sponsored by Joyent.

Though `.js` is the standard filename extension for JavaScript code, the name "Node.js" doesn't refer to a particular file in this context and is merely the name of the product. Node.js has a event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/ outputs operations, as well as for real time Web applications (e.g., real-time communication programs and browser games).

The Node.js distributed development project was previously governed by the Node.js Foundation, and has now merged with the JS Foundation to form the Open JS foundation, which is facilitated by the Linux Foundation's Collaborative Projects program.

Node.js allows the creation of Web servers and networking tools using Javascript and a collection of "modules" that handle various core functionalities. Modules are provided for file system I/O, networking binary data (buffers), cryptography functions, data streams, and other core functions. Node.js's modules use an API designed to reduce the complexity of writing server applications.

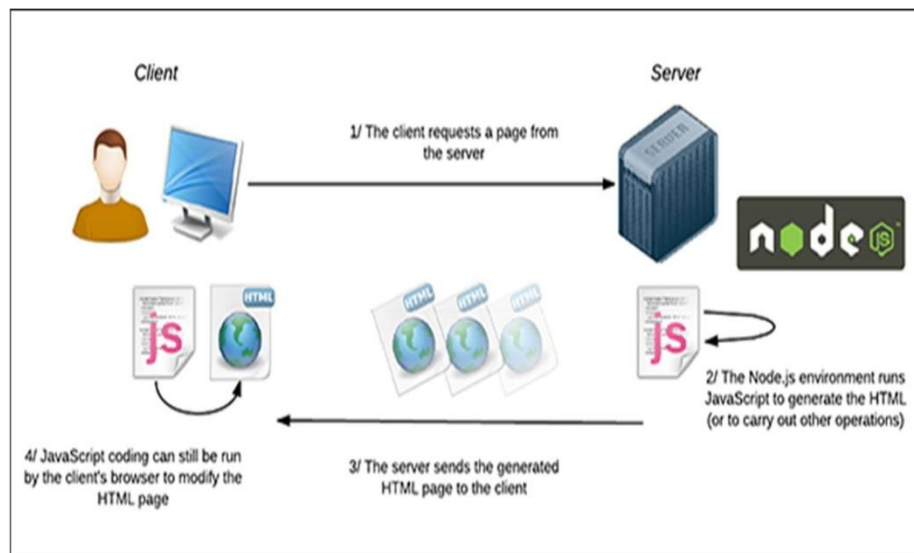


Figure 2: Overview of Node JS

WHAT IS A WEB APPLICATION (WEB APP)

A Web application (Web app) is an application program that is stored on a remote server and delivered over the Internet through a browser interface. Web services are Web apps by definition and many, although not all, websites contain Web apps. According to Web. App Storm editor Jarel Remick, any website component that performs some function for the user qualifies as a Web app. Web application can be designed for a wide variety of uses and can be used by anyone; from an organization to an individual for numerous reasons. Commonly used Web applications can include webmail, online calculators, or e-commerce shops. Some Web apps can be only accessed by a specific browser; however, most are available no matter the browser.

How a web application works

Web applications are usually coded in browser-supported language such as JavaScript and HTML as these languages rely on the browser to render the program executable. Some of the applications are dynamic, requiring server-side processing. Others are completely static with no processing required at the server.

The web application requires a web server to manage requests from the client, an application server to perform the tasks requested, and, sometimes, a database to store the information. Application server technology ranges from ASP.NET, ASP and ColdFusion, to PHP and JSP.

Here's what a typical web application flow looks like:

1. User triggers a request to the web server over the Internet, either through a web browser or the application's user interface
2. Web server forwards this request to the appropriate web application server.
3. Web application server performs the requested task – such as querying the database or processing the data – then generates the results of the requested data
4. Web application server sends results to the web server with the requested information or processed data
5. Web server responds back to the client with the requested information that then appears on the user's display.

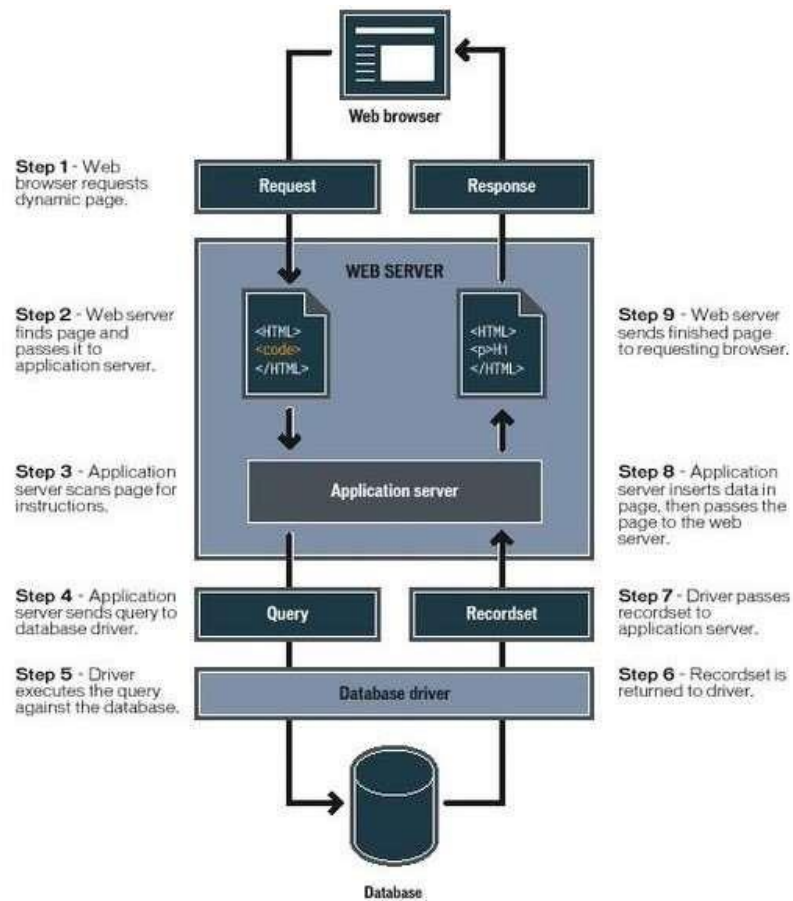


Figure 3: Working of Web App

Introduction to Dependencies

A brief description of each package and the function it will serve

[`bcryptjs`](#): used to hash passwords before we store them in our database

[`body-parser`](#): used to parse incoming request bodies in a middleware

[`concurrently`](#): allows us to run our backend and frontend concurrently and on different ports

[`express`](#): sits on top of `Node` to make the routing, request handling, and responding easier to write

[`is-empty`](#): global function that will come in handy when we use `validator`

[`jsonwebtoken`](#): used for authorization

[mongoose](#): used to interact with MongoDB

[passport](#): used to authenticate requests, which it does through an extensible set of plugins known as `strategies`

[passport-jwt](#): `passport` strategy for authenticating with a JSON Web Token (JWT); lets you authenticate endpoints using a JWT

[validator](#): used to validate inputs (e.g. check for valid email format, confirming passwords match)

[Nodemon](#): Nodemon is a utility that will monitor for any changes in code and automatically restart your server, which is perfect for development.

SCOPE

There is a great scope for providing a great experience to the people who want to have a good experience of travelling. The project that has been developed is a complete travel booking site where a user can easily search and choose where he or she wants to go. The aim of this project is to provide people of each type of taste in travelling with an interactive online platform to get all sort of information regarding their dream destinations. This web application can be great for those who want all sorts of places according to their choice and want all types of travelling facilities on a single platform.

2. SOFTWARE REQUIREMENT ANALYSIS

INTRODUCTION

The aim of this part is to gather and analyze and give an in-depth insight of the complete TRAVELYAARI PROJECT by defining the problem statement in detail. Nevertheless, it also concentrates on the capabilities required by stakeholders and their needs while defining high-level product features. The detailed requirements of the TRAVELYAARI PROJECT are provided in this document.

Purpose

The purpose of the document is to collect and analyze all assorted ideas that have come up to define the system, its requirements with respect to consumers. Also, we shall predict and sort out how we hope this product will be used in order to gain a better understanding of the project, outline concepts that may be developed later, and document ideas that are being considered, but may be discarded as the product develops.

In short, the purpose of this report document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. It defines how our client, team and audience see the product and its functionality. Nonetheless, it helps any designer and developer to assist in software delivery lifecycle (SDLC) processes.

Document Convention:

In this text, it will use font small 2 and overstriking for primary title, font small 3 for secondary title and font 4 for the content. And it will use the italic when mentions the name of the application TRAVELYAARI.

Intended Audience:

This SRS about TRAVELYAARI is for developers, mentors, users and testers. The article mainly introduces the overall description, external interface requirements system features and other non-functional requirements. I suppose mentor to read the

whole article carefully and user pay attention to overall description especially. Users and testers read the system features carefully.

Definitions, Acronyms, and Abbreviations.

Configuration	It means a product which is available / Selected from a catalogue can be customized.
FAQ	Frequently Asked Questions
CRM	Customer Relationship Management
RAID 5	Redundant Array of Inexpensive Disk/Drives

PERSPECTIVE:

A web application preferably using the MERN stack.

PRODUCT FUNCTIONS:

The web application has a simple interface with the products available on the home page with the arrival and top selling products. It has two roles: one for the admin and another for the user. Admin can create and add the product or modify the data whereas the user will only have the permission to view the product or buy the product.

In this application, User can search the product requiring its needs and can make the payment using the paypal or debit card method.

USER CLASSES AND CHARACTERISTICS:

The web application users interact with this application through a web browser. Other web applications are those applications like bots, third party application, can also use this application through its web Api. All requests should meet the specifications of application Api.

OPERATING ENVIRONMENT:

Our software is a multi-functional software system based on the windows platform. It

is compatible and can run on 64 - bit laptop or ordinary desktop as well as smartphones.

DESIGN CONSTRAINS:

Our application must accept the user login credentials whether it is the user or not. We must consider about the arrangement and beautification of the interface; Prioritization of processing operations and it deepens the difficulty of coding and testing. After, the verification the user can view the products according to its needs and after viewing the user can add to the cart and buy that product if its shipping available it will be delivered to their doorsteps.

ASSUMPTIONS AND DEPENDENCIES:

The people who manage the application should know about the knowledge of the MERN to manage the database and front-end.

TRAVELYAARI is a web app having MongoDB for backend interactions or third-party application interactions. It also uses react for frontend interactions.

List- MongoDB, Express, React, Node, Postman and CSS.

PERORMANCE REQUIREMENTS:

Since the system uses server client architecture, and use the large source information from distant server, it fetches data through internet. Internet bandwidth is

major performance parameter. As System uses GitHub repository as source of massive data, multiple request and response is needed to handle in QuickTime for instant result back to user. Large system queries are handled by the fast operating systems for both the mobile and the web-based process. Worker are used to provide faster HTTP Request handling.

Hardware Requirements:

The requests of the hardware for the web application are as followed:

- 64 bits laptop or desktop.
- TCP protocol.

Software Requirements:

To access a web portal of this application, its only need a PC/Laptop/Mobile with an integrated and updated web browser.

Desktop browser: Safari, Chrome, Firefox, Opera, IE9+.

Mobile browsers: Android, Chrome Mobile, iOS Safari.

On the server side: a PC/Web Server which meets these specifications:

- Window Operating System
- At least 2 GB RAM and 150 GB Free Space
- Redis Server Installed
- Python Compiler Installed

COMMUNICATION PROTOCOLS AND INTERFACES:

- A. TCP protocol.
- B. Secure transmission and encryption techniques.
- C. File transfer rate.

3. SOFTWARE DESIGN

Function Oriented Design for procedural approach and different diagram to show the designing of the application.

DATAFLOW DIAGRAM

DFD Level 0

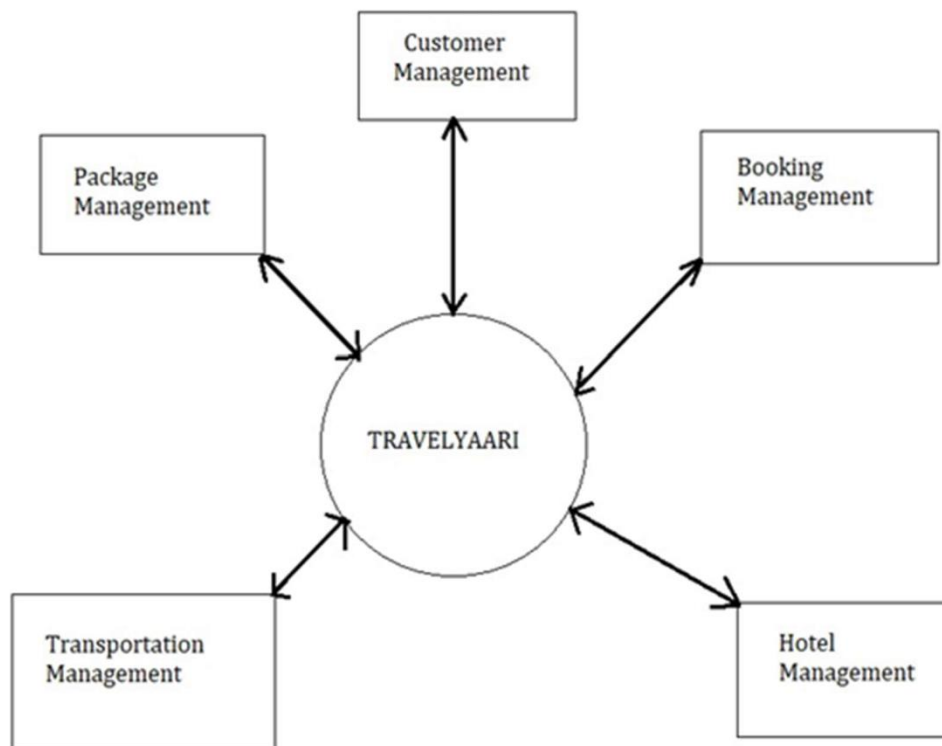


Figure 4: DFD Level 0

DFD Level -1

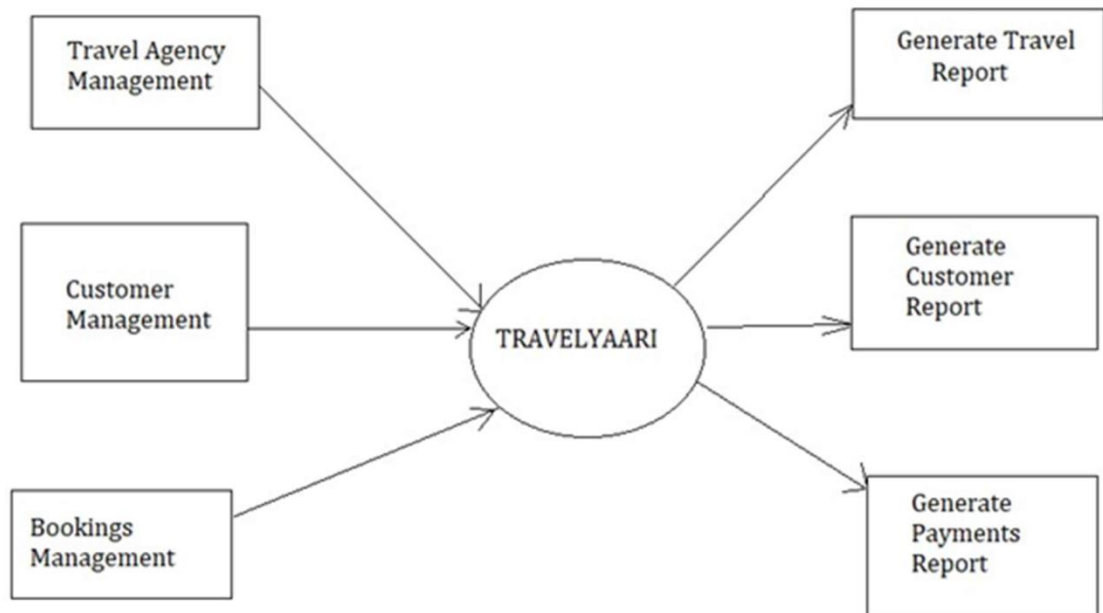


Figure 5:DFD Level 1 Admin Side

DFD Level 2

2nd Level Admin DFD

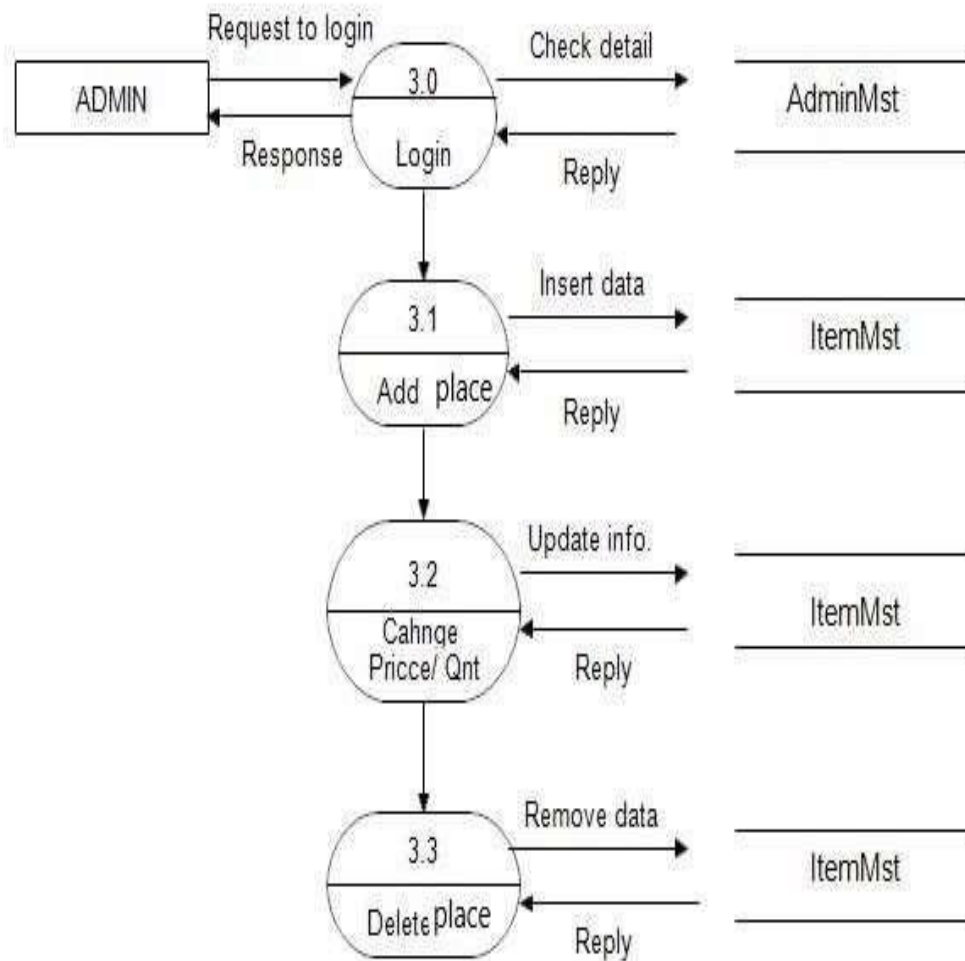


Figure 6: DFD Level 2 Admin

DFD Level 2 User

2st Level User DFD

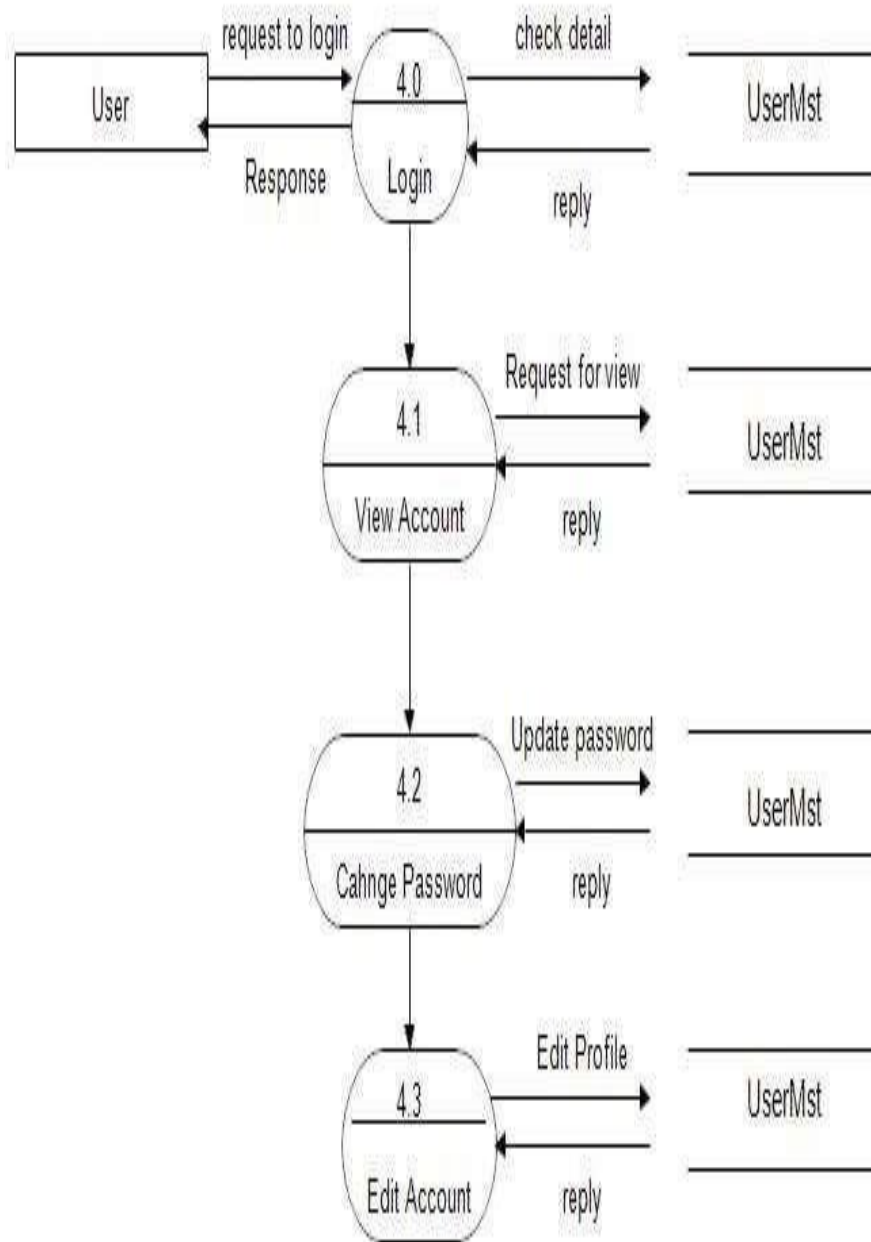


Figure 7: DFD Level 2 User

USE CASE DIAGRAM

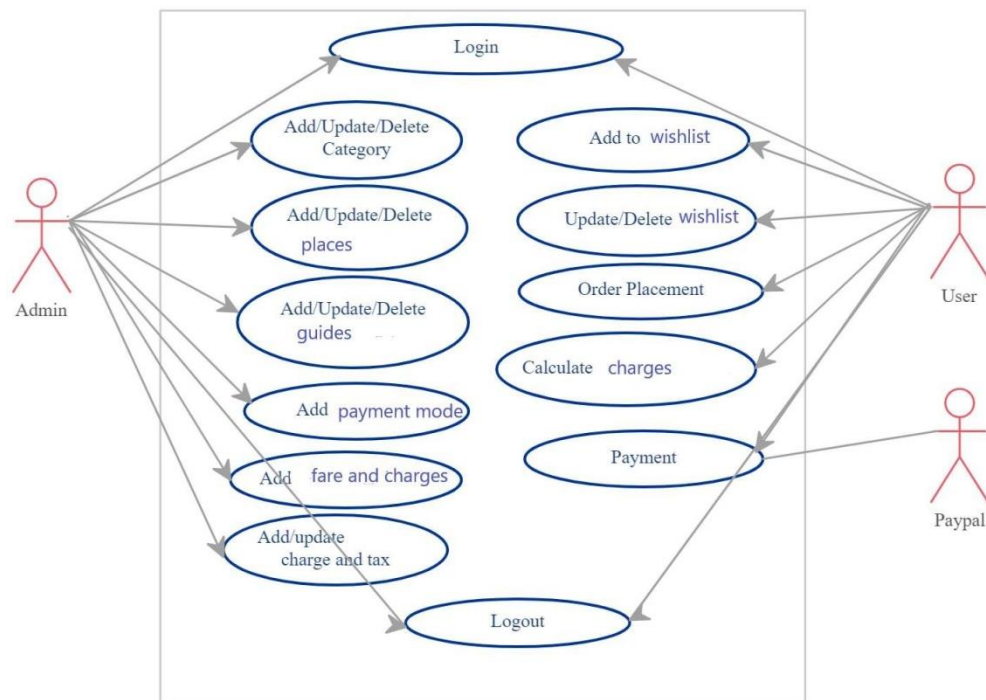


Figure 8: UseCase Diagram

Sequence Diagram

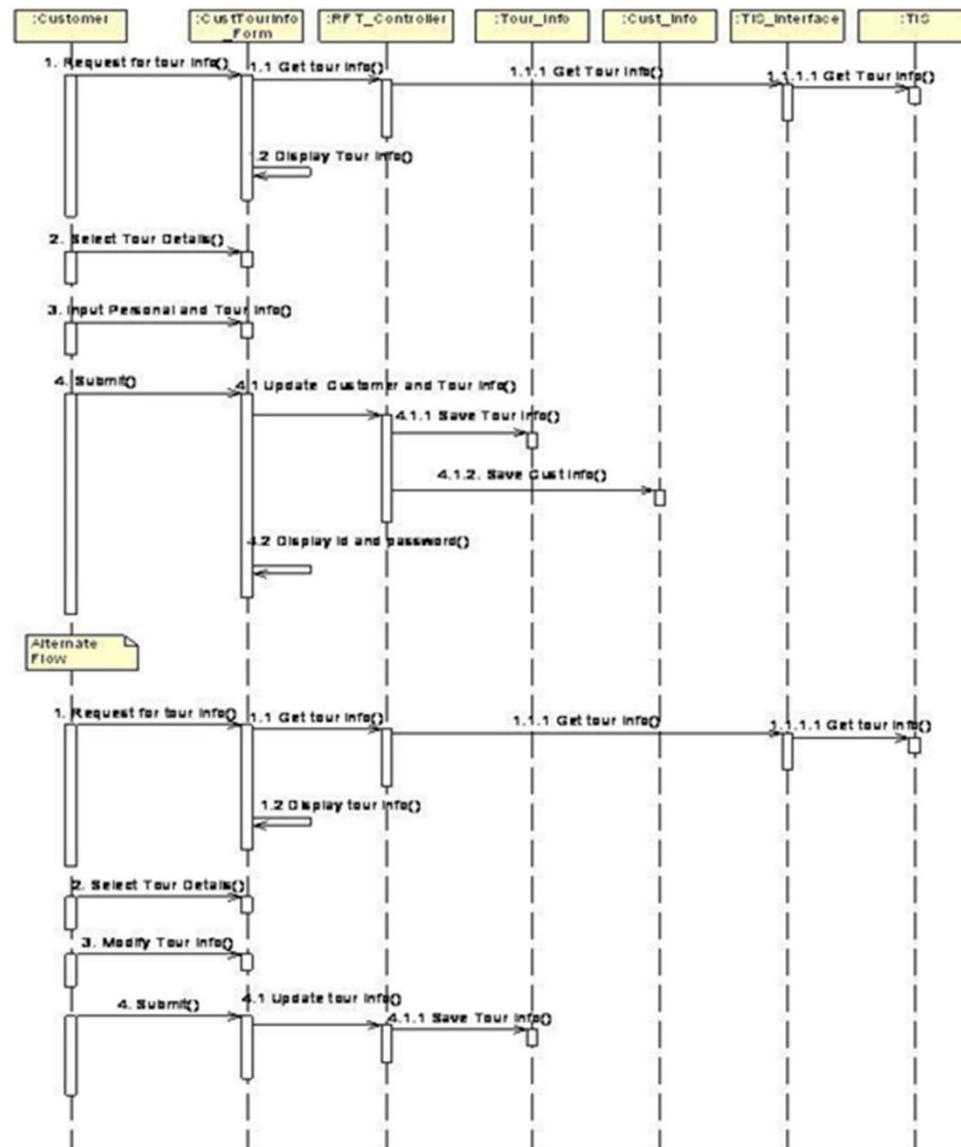


Figure 9: Sequence Diagram

CLASS DAGRAM

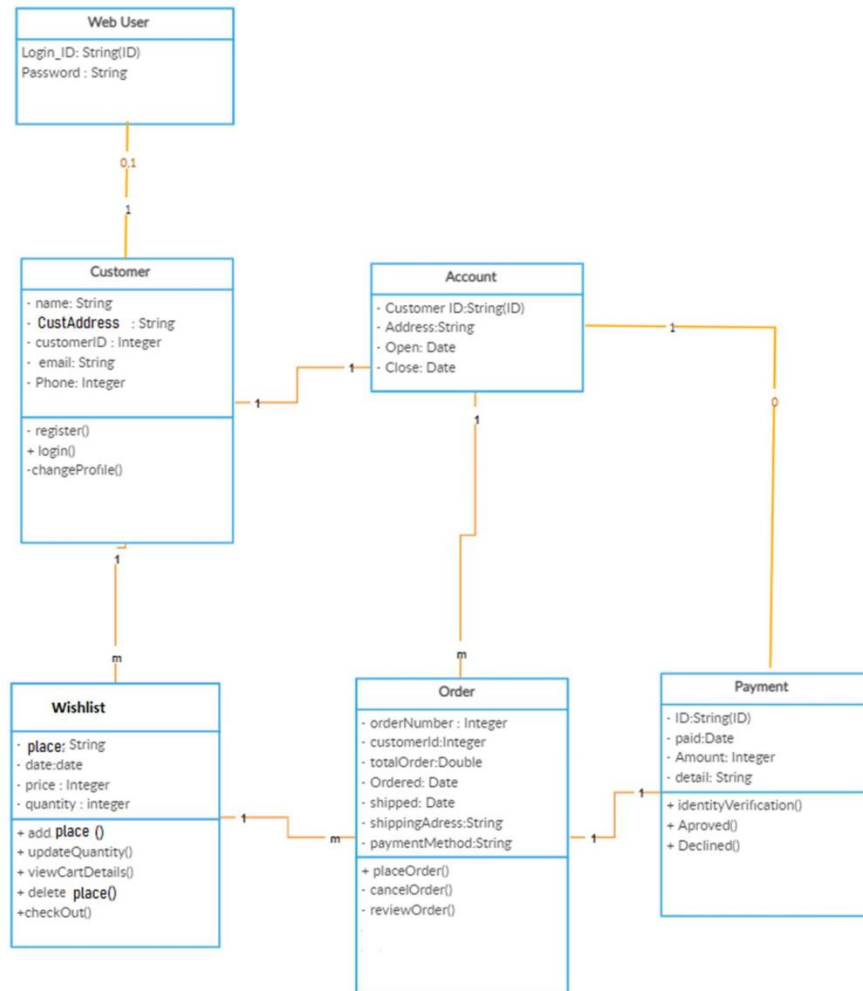


Figure 10: Class Diagram

DATABASE DESIGN

This project is based on NoSql database performed using MonoDb. Our Database named Ecommerce contains 4 collections that are used to store different results. The 4 Collections are as.

User Database Collection.

The user Collection contains the information regarding user and its role. It has 10 attributes namely –

1. _id – User id of the user
2. Role - Define user type

- a. Role 0 – For user
- b. Role 1 – For Admin
3. Name – Name of the user
4. Email – Email Id of the password.
5. Salt- Encryption method of password
6. Hashed Password – Encrypted Password.
7. History – User Buying History in form of array.
8. Id Created – Id creation date and time
9. Id update – Id Updating Date.
10. _ v – version history.

Categories Database Collection.

The Category collection contain information about product category, its name, created date and time and updated date and time. Category Schema is as –

```
const categorySchema = new mongoose.Schema(  
  {  
    name: {  
      type: String,  
      trim: true,  
      required: true,  
      maxlength: 32,  
      unique: true  
    }  
  },  
  { timestamps: true }  
);
```

Figure 11: Category Schema

```
const userSchema = new mongoose.Schema({
  name: {
    type: String,
    trim: true,
    required: true,
    maxlength: 32
  },
  email: {
    type: String,
    trim: true,
    required: true,
    unique: true
  },
  hashed_password: {
    type: String,
    required: true
  },
  about: {
    type: String,
    trim: true
  },
  salt: String,
  role: {
    type: Number,
    default: 0
  },
  history: {
    type: Array,
    default: []
  }
}, { timestamps: true });
```

Figure 12: Users Schema

Products Database Collection

The Products collection contain information about each product which includes the following attributer names.

- `_id` - Id of the product
- Photo – Images of the product.
- Name – Name of the Product
- Description – Complete Description of the product
- Category – category of the product from category schema
- Price – Price of the product.
- Shipping – Shipping Status of the product.
- Quantity – Total Available Quantity of the product.
- Time Stamp – Created and updated status of the product.
- YouTube Link- Link of YouTube video .

The products schema is .

```
const productSchema = new mongoose.Schema(  
  {  
    name: {  
      type: String,  
      trim: true,  
      required: true,  
      maxlength: 32  
    },  
    description: {  
      type: String,  
      required: true,  
      maxlength: 2000  
    },  
    price: {  
      (property) trim: boolean  
      trim: true,  
      required: true,  
      maxlength: 32  
    },  
    category: {  
      type: ObjectId,  
      ref: "Category",  
      required: true  
    },  
    quantity: {  
      type: Number  
    },  
    sold: {  
      type: Number,  
      default: 0  
    },  
    photo: {  
      data: Buffer,  
      contentType: String  
    },  
    shipping: {  
      required: false,  
      type: Boolean  
    }  
  }  
)
```

Figure 13: Products Schema

Order Database Collection

Order Collection contains information about products orders . the schema of ordered products is –

```
const OrderSchema = new mongoose.Schema(  
  {  
    products: [CartItemSchema],  
    transaction_id: {},  
    amount: { type: Number },  
    address: String,  
    status: {  
      type: String,  
      default: "Not processed",  
      enum: ["Not processed", "Processing", "Shipped", "Delivered", "Cancelled"] // enum means string objects  
    },  
    updated: Date,  
    user: { type: ObjectId, ref: "User" }  
  },  
  { timestamps: true }  
);
```

Figure 14:Order Collection

4. IMPLEMENTATION AND USER INTERFACACES

TRAVELYAARI:

The Trvelyaari-API is the Node App and will manage the complete backend of our projects. this is created using Node, Express and MongoDB. This application contains the complete routing and modelling with all Api calls related to the projects. It includes Controllers, Models, Validators, and Routes. This application has dependency which includes sandgrid, bosy-parser, Braintree, cookie-parser, express, nodemon uuid and many more.

This rest Api will run on port 5000 of host and will act as a server for our application project. Below is the complete flow directory of our server application.

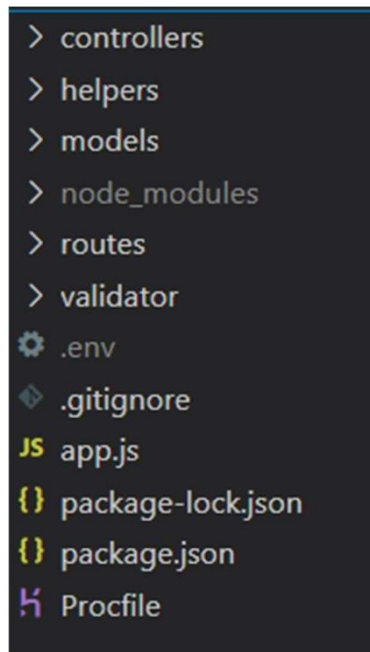


Figure 15: Server Directory

Models – Models are used to create Mongo DB schema and we have created 4 models namely Users, Products, Category and Orders. A Mongoose model is a wrapper on the Mongoose schema. A Mongoose schema defines the structure of the document, default values, validators, etc., whereas a Mongoose model provides an interface to the database for creating, querying, updating, deleting records, etc.

we defined *Mongoose* models to interact with the database, and used a (standalone) script to create some initial library records. We can now write the code to present that information to users. The first thing we need to do is determine what information we

want to be able to display in our pages, and then define appropriate URLs for returning those resources. Then we're going to need to create the routes (URL handlers) and views (templates) to display those pages.

Controllers - A controller's purpose is to receive specific requests for the application. The routing mechanism controls which controller receives which requests. Frequently, each controller has more than one route, and different routes can perform different actions.

In order to create a basic controller, we use classes and decorators. Decorators associate classes with required metadata and enable Nest to create a routing map (tie requests to the corresponding controllers). In our project we have created 6 Controllers namely –

- ❖ Auth – for authentications
- ❖ Braintree – For Braintree payment embedding.
- ❖ Category – For Controlling Products Categories.
- ❖ Order- For controlling orders flow
- ❖ Products- for controlling Products availability
- ❖ Users – For users Controlling.

Apart from that, "Routes" to forward the supported requests (and any information encoded in request URLs) to the appropriate controller functions. Controller functions to get the requested data from the models, create an HTML page displaying the data, and return it to the user to view in the browser. Views (templates) used by the co A route is a section of Express code that associates an HTTP verb (GET, POST, PUT, DELETE, etc.), a URL path/pattern, and a function that is called to handle that pattern.

There are several ways to create routes. For this project we have to use the express. Router middleware as it allows us to group the route handlers for a particular part of a site together and access them using a common route-prefix. We'll keep all our library-related routes in a "catalog" module, and, if we add routes for handling user accounts

or other functions, we can keep them grouped separately controllers to render the data.

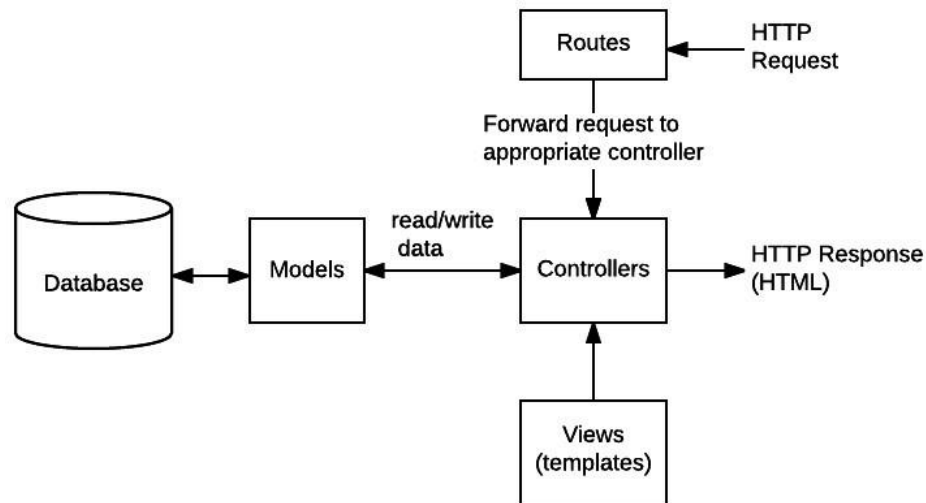


Figure 16: Data Flow

We also have helper function to handle database handling and validators for data validation.

We have following routes in our project

- / signup
- / Signin
- /signinout
- /userDashboard
- /products
- /products/products-search
- /products/related/:productId
- /products/categories
- /products/by/search
- /product/photo/:productId
- /order/:orderId/status/:UserId
- /order/status-values/:UserId
- /order/create/:UserId
- /order/list/:UserId
- /category/:categoryId

- /category/create/:UserId
- /category/:categoryId/:UserId
- /categories
- /braintree/getToken/:UserId
- /braintree/payment/:UserId

TRAVELYAARI UI –

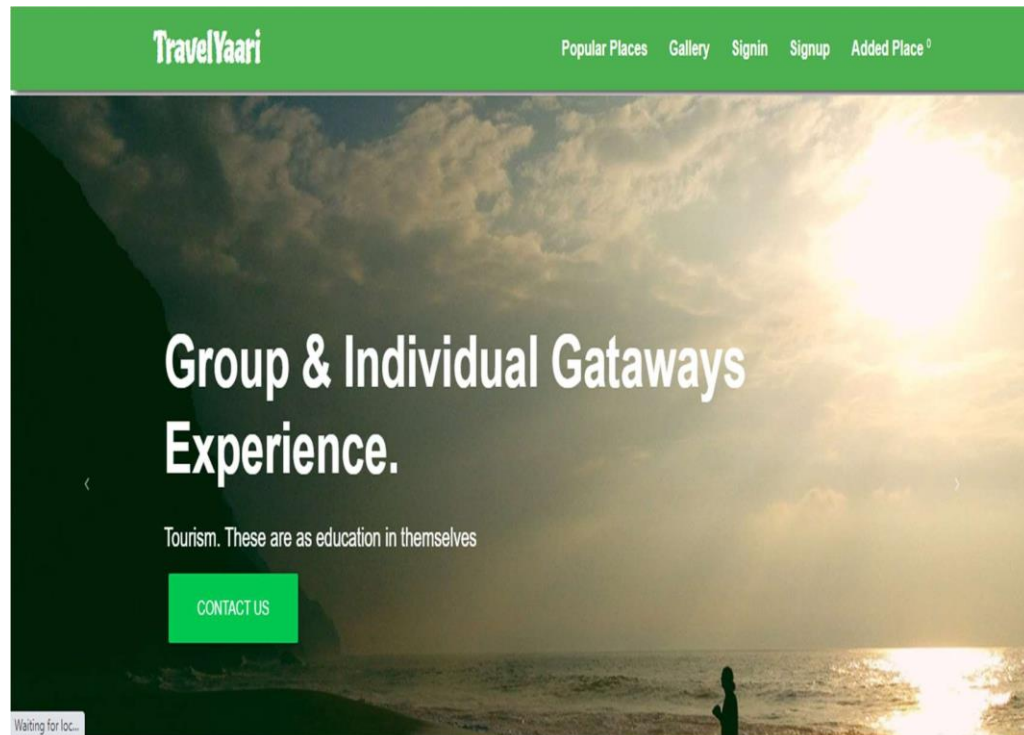


Figure 17: Home Page Without Signin

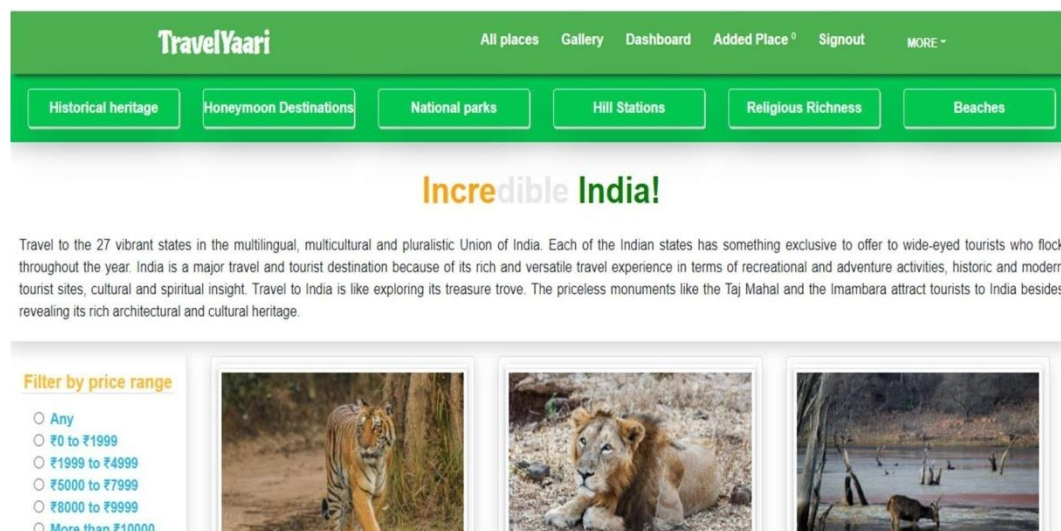
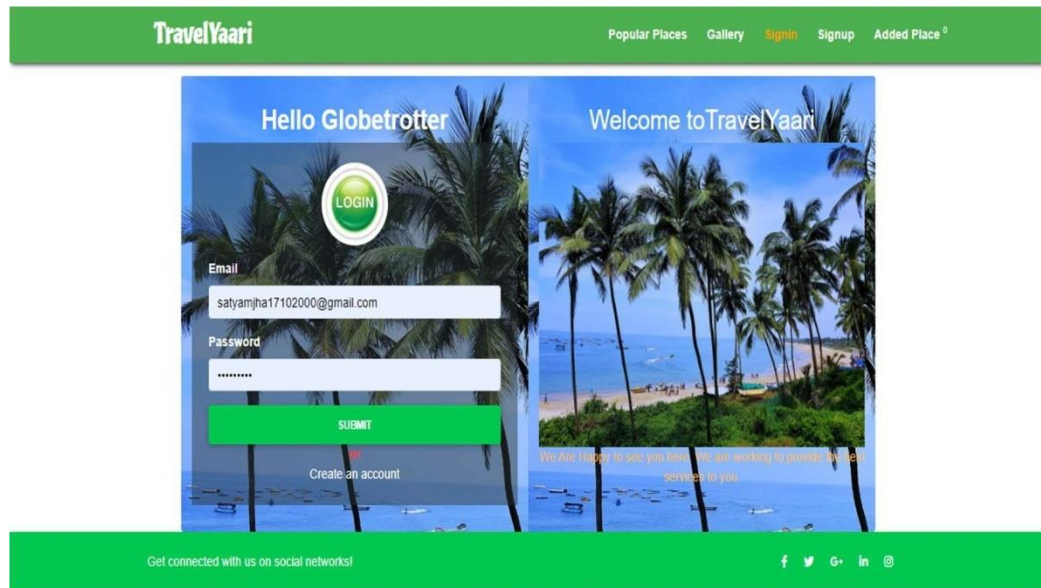
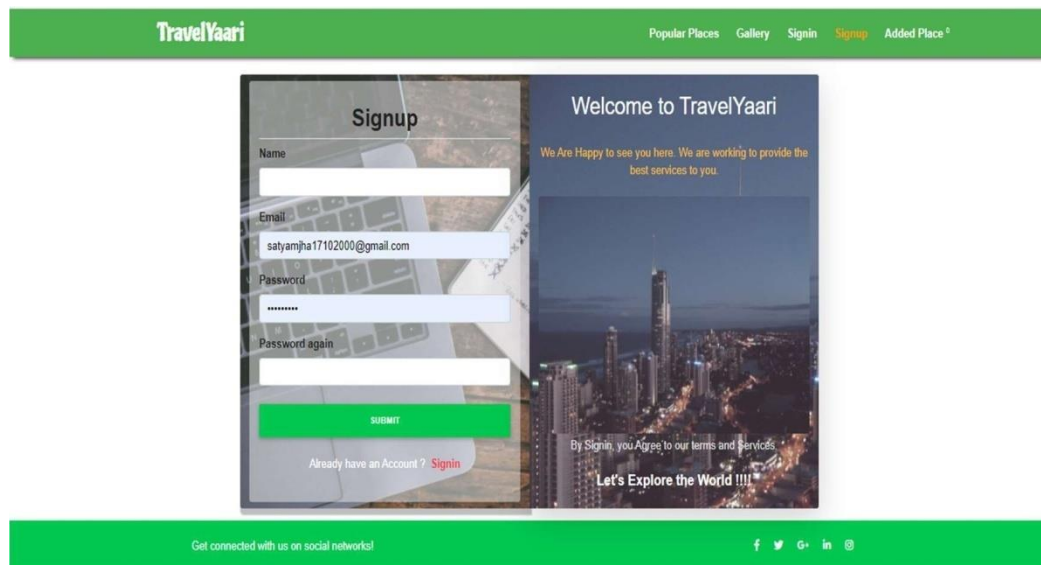


Figure 18: Home page After Signin



The image shows the 'SignIn' page of the TravelYaari application. The header is green with the 'TravelYaari' logo on the left and navigation links 'Popular Places', 'Gallery', 'Signin' (highlighted in orange), 'Signup', and 'Added Place 0' on the right. The main content area has a background image of palm trees and a beach. On the left, there is a 'Hello Globetrotter' greeting, a green circular 'LOGIN' button, and input fields for 'Email' (containing 'satyamjha17102000@gmail.com') and 'Password' (masked with dots). Below these is a green 'SUBMIT' button and a link 'Create an account'. On the right, there is a 'Welcome to TravelYaari' greeting and a message: 'We Are Happy to see you here. We are working to provide the best services to you.' The footer is green with the text 'Get connected with us on social networks!' and social media icons for Facebook, Twitter, Google+, LinkedIn, and Instagram.

Figure 19: SignIn Page



The image shows the 'SignUp' page of the TravelYaari application. The header is green with the 'TravelYaari' logo on the left and navigation links 'Popular Places', 'Gallery', 'Signin', 'Signup' (highlighted in orange), and 'Added Place 0' on the right. The main content area has a background image of a city skyline at night. On the left, there is a 'Signup' heading, input fields for 'Name', 'Email' (containing 'satyamjha17102000@gmail.com'), 'Password' (masked with dots), and 'Password again', followed by a green 'SUBMIT' button. Below the button is a link 'Already have an Account ? Signin'. On the right, there is a 'Welcome to TravelYaari' greeting and a message: 'We Are Happy to see you here. We are working to provide the best services to you.' Below this is a link 'By Signin, you Agree to our terms and Services' and the text 'Let's Explore the World !!!!'. The footer is green with the text 'Get connected with us on social networks!' and social media icons for Facebook, Twitter, Google+, LinkedIn, and Instagram.

Figure 20: SignUp Page



Figure 21:Footer section

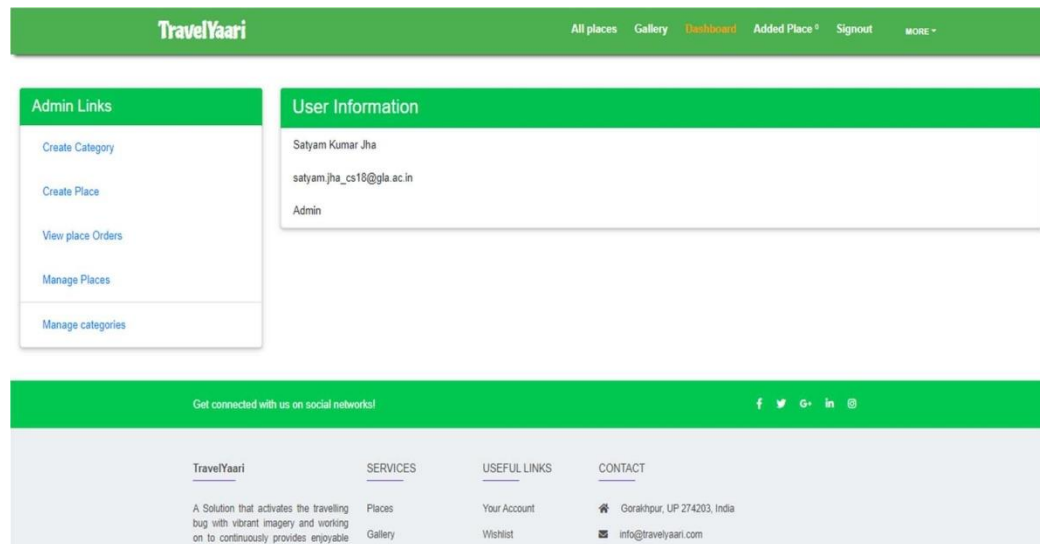
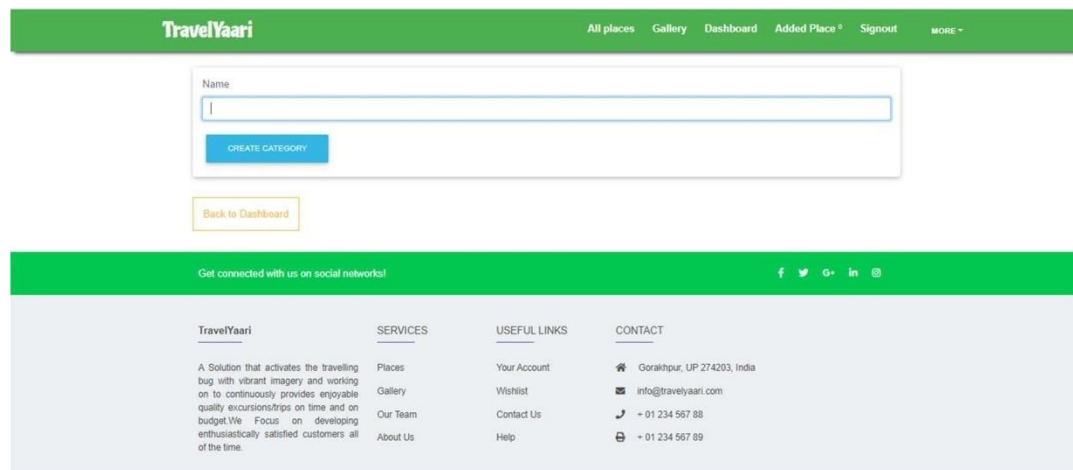


Figure 22:Admin Dashboard



TravelYaari

All places Gallery Dashboard Added Place 5 Signout MORE +

Name

CREATE CATEGORY

Back to Dashboard

Get connected with us on social networks!

TravelYaari

A Solution that activates the travelling bug with vibrant imagery and working on to continuously provides enjoyable quality excursions/trips on time and on budget. We Focus on developing enthusiastically satisfied customers all of the time.

SERVICES

Places
Gallery
Our Team
About Us

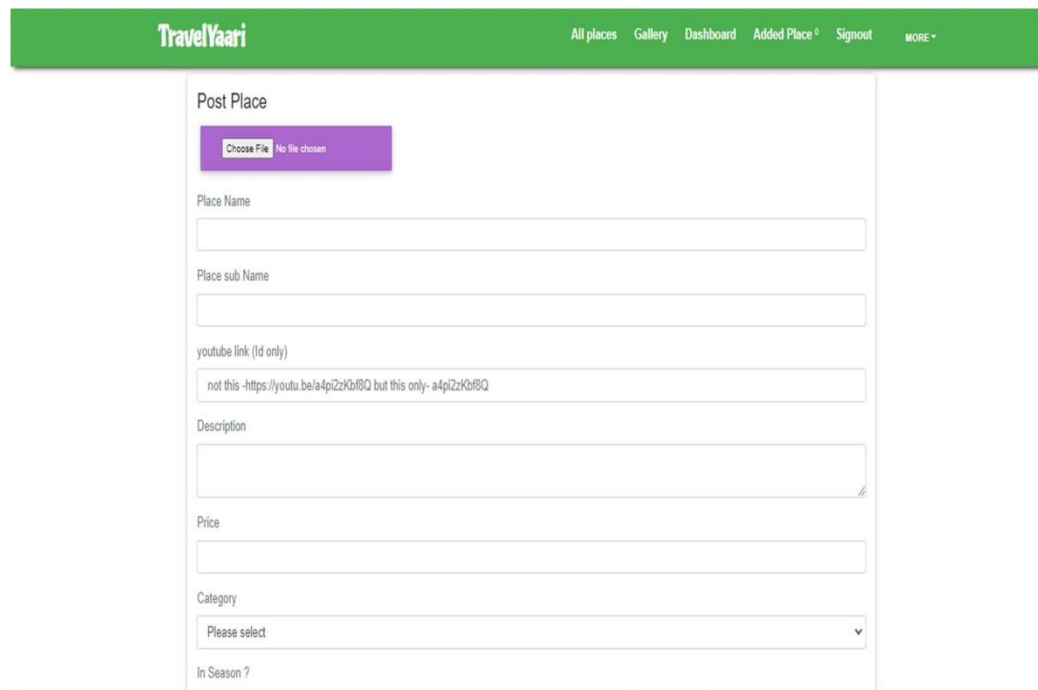
USEFUL LINKS

Your Account
Wishlist
Contact Us
Help

CONTACT

Gorakhpur, UP 274203, India
info@travelyaari.com
+ 01 234 567 88
+ 01 234 567 89

Figure 23: Add new Category page



TravelYaari

All places Gallery Dashboard Added Place 5 Signout MORE +

Post Place

Choose File No file chosen

Place Name

Place sub Name

youtube link (Id only)

not this -https://youtu.be/a4piZzKbR8Q but this only- a4piZzKbR8Q

Description

Price

Category

Please select

In Season ?

Get connected with us on social networks!

TravelYaari

A Solution that activates the travelling bug with vibrant imagery and working on to continuously provides enjoyable quality excursions/trips on time and on budget. We Focus on developing enthusiastically satisfied customers all of the time.

SERVICES

Places
Gallery
Our Team
About Us

USEFUL LINKS


Your Account
Wishlist
Contact Us
Help

CONTACT

Gorakhpur, UP 274203, India
info@travelyaari.com
+ 01 234 567 88
+ 01 234 567 89


Figure 24: Add Place page

Popular Places




Ooty, Tamil Nadu
Queen of the Nilgiris ★★★★★
Price : ₹ 8000

Nestled amidst Nilgiri hills, Ooty, also known as Udagamandalam, is a hill station in Tamil Nadu which serves as a top-rated tourist destination. Once regarded as the summer headquarters of the East India Company, the Queen of the hills is a pictures que getaway. Dotted with tea gardens, serene waterfalls, winding country lanes, and charming colonial




Auli, Uttarakhand
Skiing destination of India ★★★★★
Price : ₹ 6000

Dotted with the apple orchards, old oaks, and pine trees there is no dearth of natural beauty in Auli. Apart from skiing you can also go for numerous treks in the hills of Garhwal Himalayas and enjoy the spellbinding views of the snow-draped mountains. It is a popular hill resort in the Himalayan range dating back to the 8th Century AD. The Garhwal Mandal



Leh Ladakh, Jammu & Kashmir
India's Own Moonland ★★★★★
Price : ₹ 9000

Ladakh is a union territory in the Kashmir region of India. Formerly falling in the state of Jammu & Kashmir, Ladakh was administered a union territory on 31st October 2019. Extending from the Siachen Glacier to the main Great Himalayas, Ladakh is a land like no other. Ladakh is an adventure playground



Manali, Himachal Pradesh
India's Honeymoon capital ★★★★★
Price : ₹ 10000

Nestled in between the snow-capped slopes of the Pir Panjal and the Dhauladhar ranges, Manali is one of the most popular hill stations in the country. With jaw-dropping views, lush green forests, sprawling meadows carpeted with flowers, gushing blue streams, a perpetual fairy-tale like mist lingering in the air, and a persistent fragrance of pines -

Figure 25: popular places

Total 70 Places

Leh Ladakh, Jammu & Kashmir	Delete	Delete
Ooty, Tamil Nadu	Delete	Delete
Manali, Himachal Pradesh	Delete	Delete
Auli, Uttarakhand	Delete	Delete
Srinagar, Jammu & Kashmir	Delete	Delete
Shimla, Himachal Pradesh	Delete	Delete
Mussoorie, Uttarakhand	Delete	Delete
Darjeeling, West Bengal	Delete	Delete
Mcleodganj, Himachal Pradesh	Delete	Delete
Munnar, Kerala	Delete	Delete
Coorg, Karnataka	Delete	Delete
Kasol, Himachal Pradesh	Delete	Delete
Taj Mahal, Agra	Delete	Delete
Agra Fort, Agra	Delete	Delete
Jhansi Fort, Jhansi	Delete	Delete
Red Fort, Delhi	Delete	Delete
Qutub Minar, Delhi	Delete	Delete
Hawa Mahal, Jaipur	Delete	Delete
Amer Fort, Rajasthan	Delete	Delete
Sanchi Stupa, Madhya Pradesh	Delete	Delete
Qasim Fort, Madhya Pradesh	Delete	Delete
Konark Temple, Odisha	Delete	Delete
Char Minar, Hyderabad	Delete	Delete
The Ruins of Hampi, Karnataka	Delete	Delete

Figure 26: Manage places Page

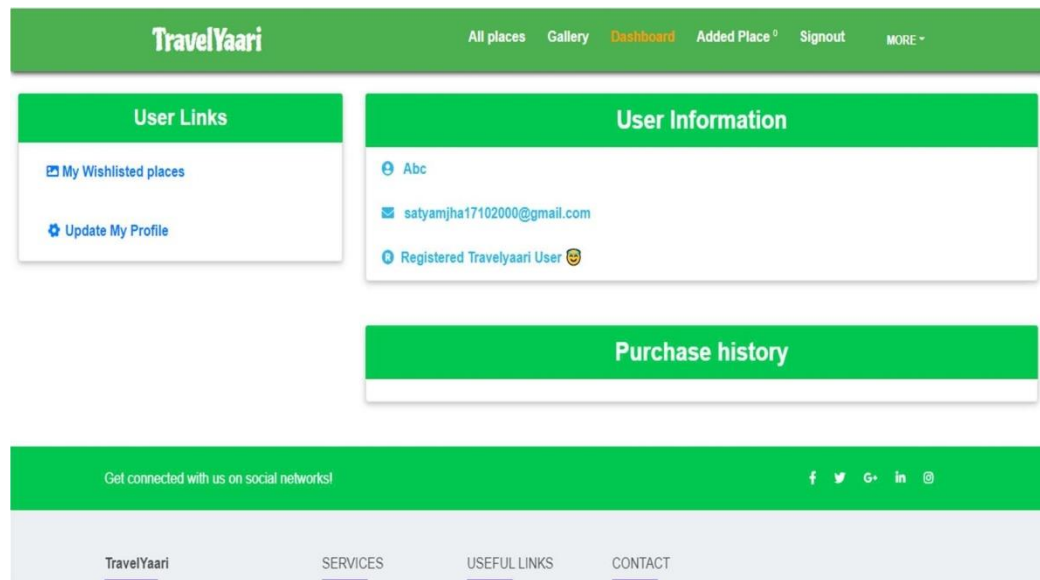


Figure 27: User Dashboard

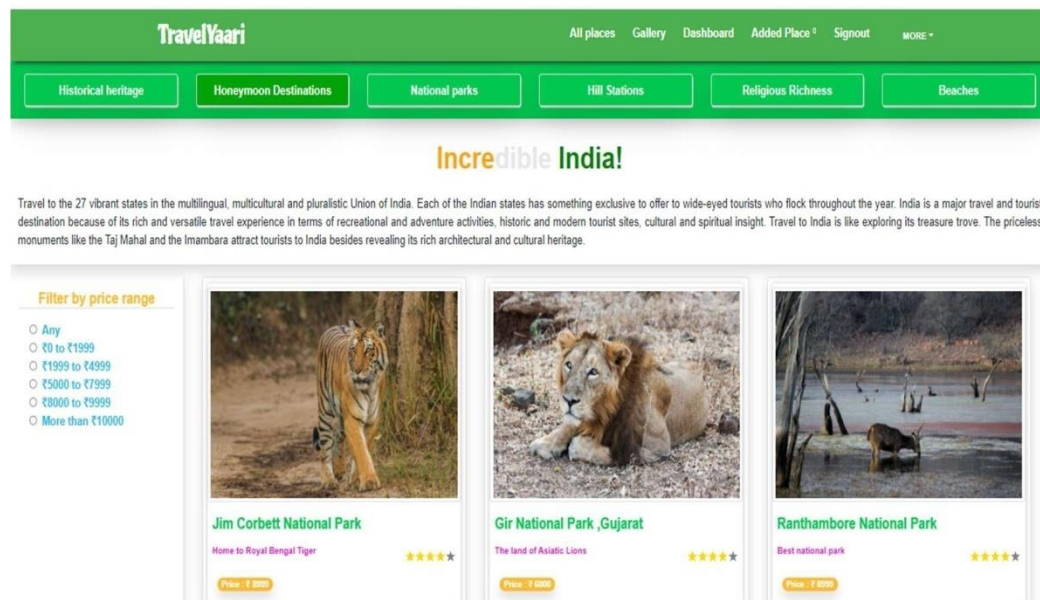


Figure 28: Places Page

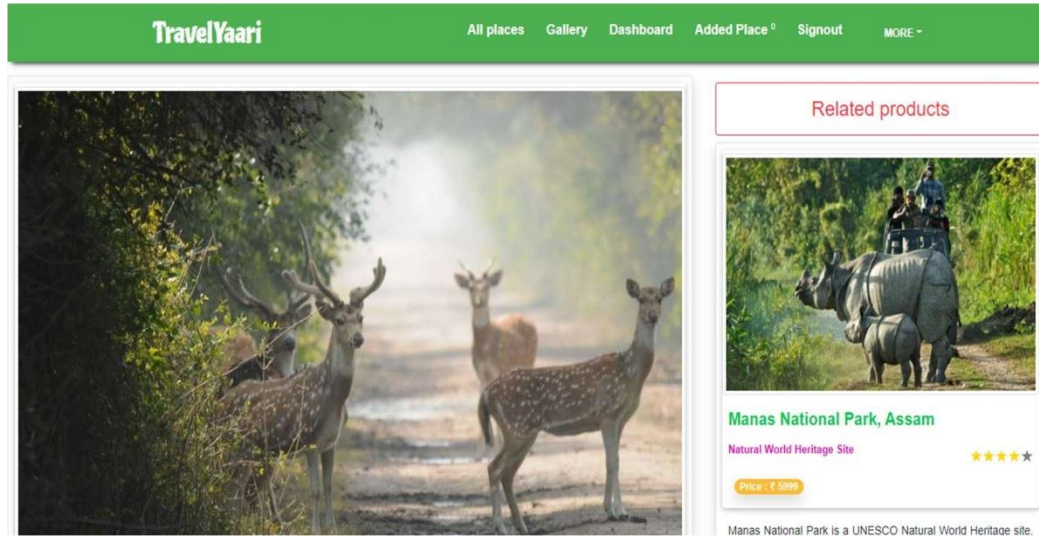


Figure 29: related places

Travel to the 27 vibrant states in the multilingual, multicultural and pluralistic Union of India. Each of the Indian states has something exclusive to offer to wide-eyed tourists who flock throughout the year. India is a major travel and tourist destination because of its rich and versatile travel experience in terms of recreational and adventure activities, historic and modern tourist sites, cultural and spiritual insight. Travel to India is like exploring its treasure trove. The priceless monuments like the Taj Mahal and the Imambara attract tourists to India besides revealing its rich architectural and cultural heritage.

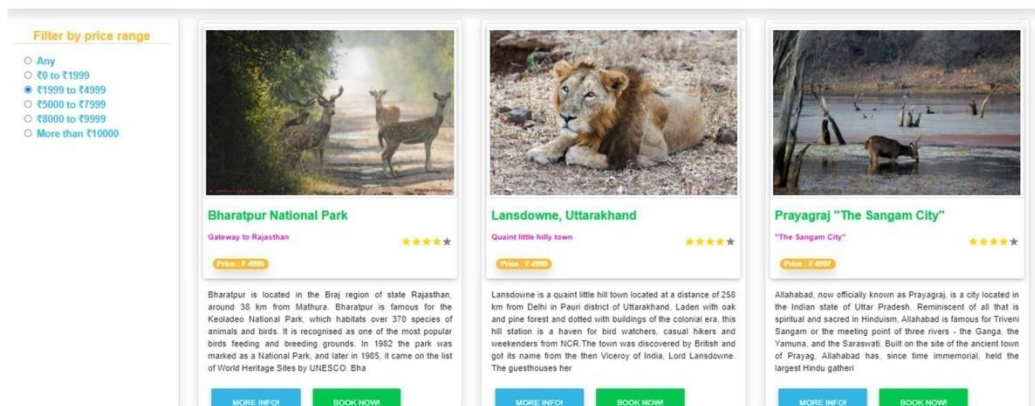


Figure 30: Filter places

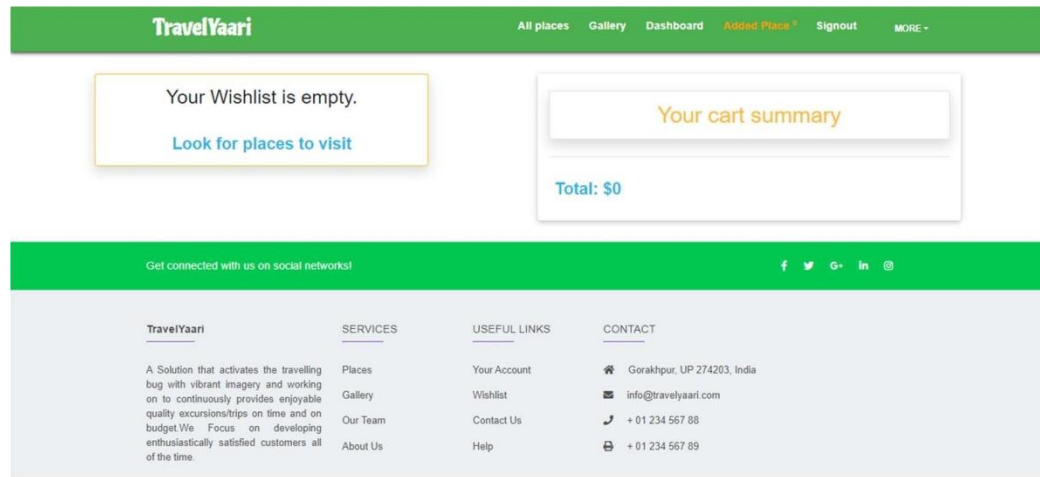


Figure 31: Wishlist Page

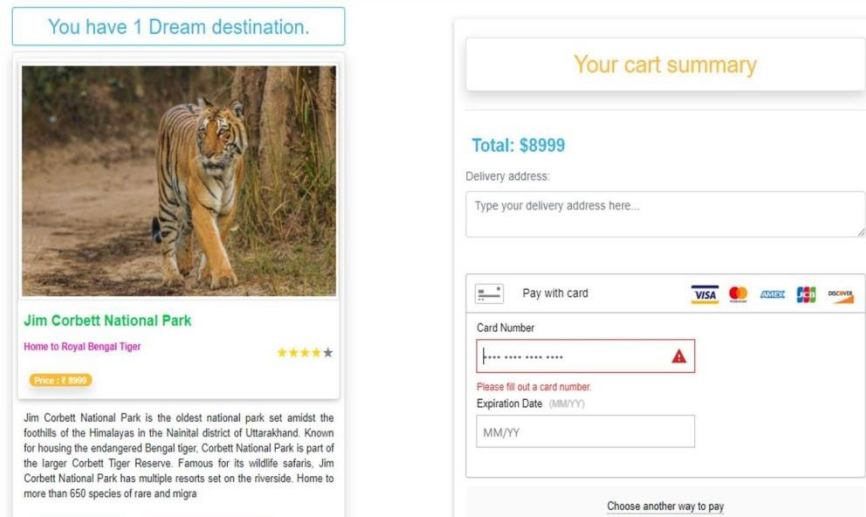


Figure 32: Payment by Card

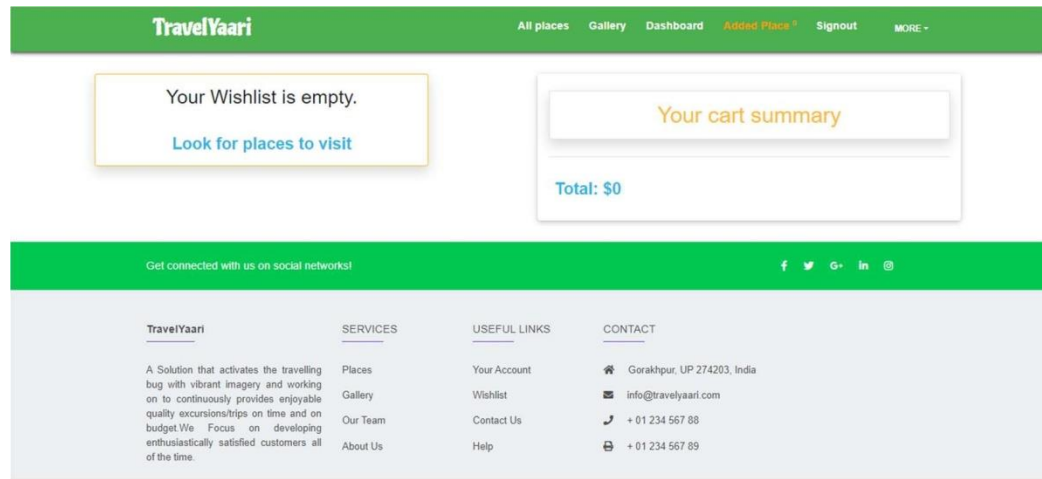


Figure 33: Payment Successful and Card Empty

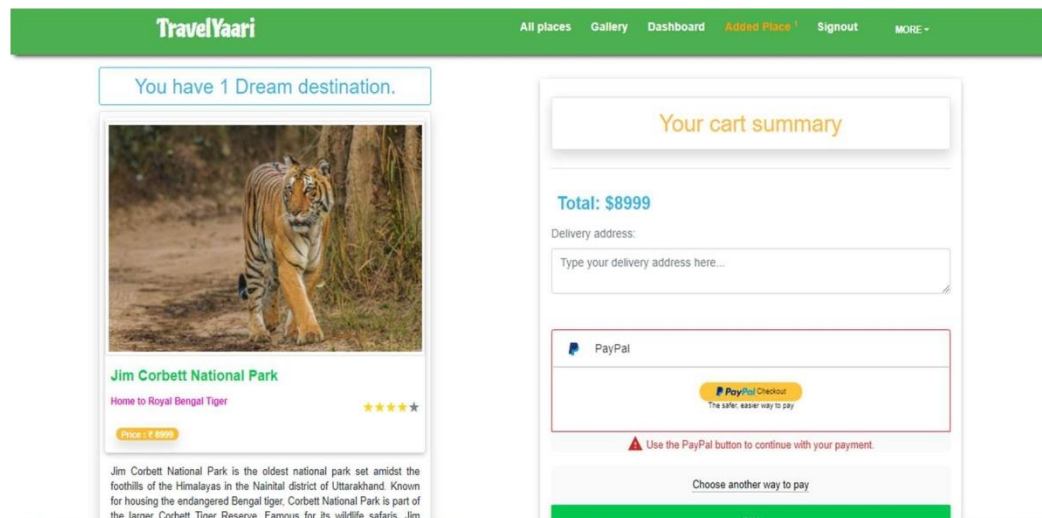
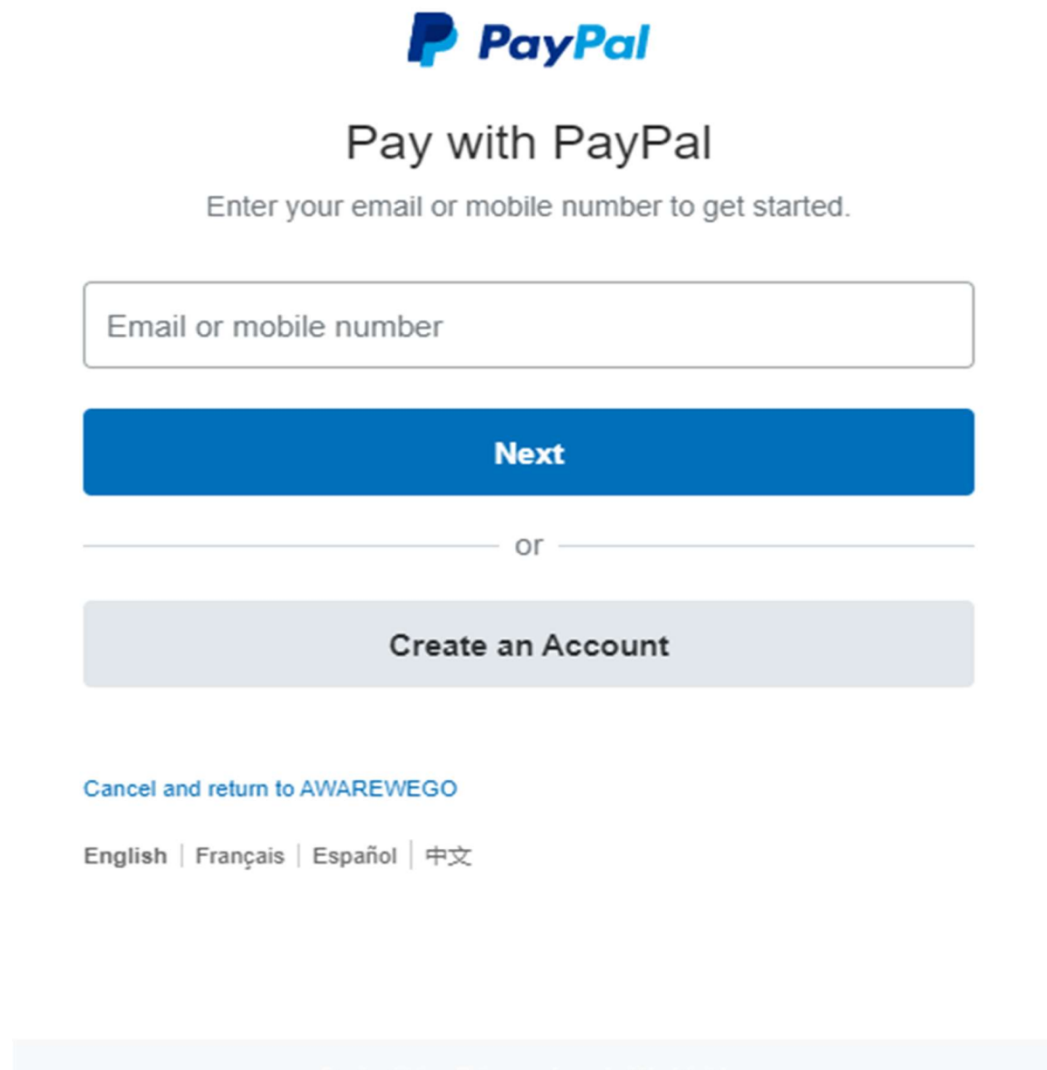


Figure 34: Payment by paypal

A screenshot of the PayPal payment interface. At the top is the PayPal logo. Below it is the heading "Pay with PayPal" and a subtext "Enter your email or mobile number to get started." There is a text input field with the placeholder "Email or mobile number". Below the input field is a blue button labeled "Next". Underneath the button is a horizontal line with the word "or" in the center. Below the line is a light gray button labeled "Create an Account". At the bottom left, there is a link "Cancel and return to AWAREWEGO" and a row of language options: "English | Français | Español | 中文".

PayPal

Pay with PayPal

Enter your email or mobile number to get started.

Next

or

Create an Account

[Cancel and return to AWAREWEGO](#)

English | Français | Español | 中文

Figure 35: paypal Integration

5. CONCLUSION AND FUTURE WORK

Conclusion

The main purpose of our project is to develop an application that offers new aspects of learning and improving knowledge in educational area. Most of the available apps are entertainment-based, which mostly do not contribute to the academic enhancement of the students. The main purpose of our project is to develop an application that offers new aspects of learning and improving knowledge in educational area. Most of the available apps are entertainment-based, which mostly do not contribute to the academic enhancement of the students.

The theme of this project is to provide user a quality experience when it comes to travelling experiences and give some prior information to the user before actually visiting a particular destination.

Through this project, we have learnt a lot MERN stack technologies. We have found that the difficult part was to gather all the information to be used as content and making the pages responsive and interactive from user point of view. We hope that other developers will take advantage from our experience/from our development.

Scope of Further Development

Web-based application development is a crucial topic for a travelling management site. People can browse, choose the places, and order towards the company, and the company will be providing the convenience within a few days. We have a plan to change and bring a few changes to this website. Besides, we have another plan to add a few functionalities in this website which are given below:

- ❖ Add the chatting system.
- ❖ Have a launch android app
- ❖ Add a greater number of online payment system other than paypal and Card.

6. REFERENCES

- [1] <https://nodejs.org/en/docs/> , accessed on April 2022.
- [2] <https://reactjs.org/docs/getting-started.html> , accessed on April 2022.
- [3] <https://react-bootstrap.github.io/> , Accessed on April 2022.
- [4] <https://www.braintreepayments.com/sandbox> , Accessed on May 2022.
- [5] <https://developer.paypal.com/home>, Accessed on May 2022.
- [6] <https://expressjs.com/en/starter/installing.html> , accessed on April 2022.
- [7] https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/routes ,accessed on May 2022.
- [8] <https://www.udemy.com/course/create-nodejs-app-with-express-socket-io-and-mongodb/learn/lecture/24058868?start=0#overview>, Accessed on April 2022.
- [9] <https://www.udemy.com/course/react-the-complete-guide-incl-redux/learn/lecture/8090870?start=0#overview> , Accessed on April 2022.
- [10] <https://www.heroku.com/nodejs> , accessed on May 2022.
- [11] <https://www.mongodb.com/cloud/atlas>, accessed on April 2022.
- [12] <https://docs.mongodb.com/> , accessed on April 2022.