**RUNNING THE PROJECT:**

Extract all the files from zip. Open the tourezee folder in a proper IDE or VS Code.

Run npm install to make sure all the required libraries mentioned in the package.json are installed.(Although this zip should contain all the required packages already)

To run the file execute command “ng serve” and wait for some time. On your local browser run <http://localhost:4200> to get the website running.

**OTHER INFORMATION:**

The website makes API calls to fetch information. All the components are made separately for easy reading and modifying in the Angular frontend. The website at moment uses firebase API for authentication whose API keys should be changed. The project has also made sure that it can be connected to backend through endpoints which are designed in backend file. It contains the design for further extension of other functionalities of site. At present the website fetches the required category information and details on pressing of search button.

The project contains sufficient comments for easy reading of code. The backend code is well accounted for and all the routes are labelled as well as their controllers. Their controller has functions that are well labelled and all the libraries that are imported are accompanied with required comments to understand their purpose.

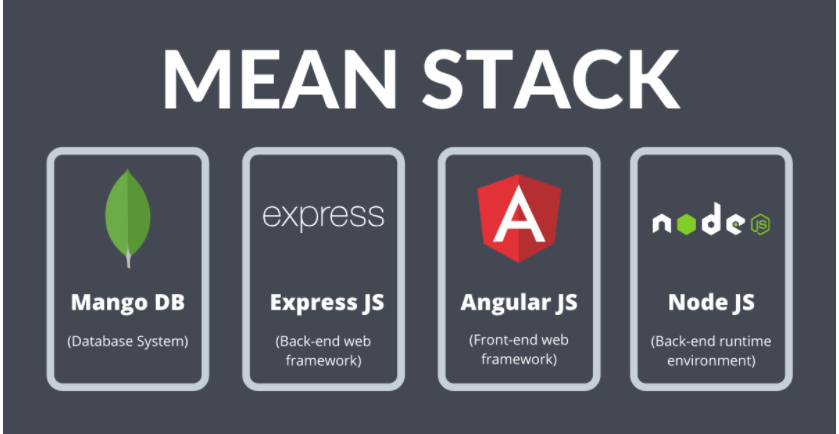
MEAN stack (MongoDB, ExpressJS, Angular, NodeJS) is used for the website and Flutter for the mobile app version of this website.

Node JS :- It is a runtime environment for JS and it also has http module with help of which any standalone application as well as websites and server side programming can be done.

MongoDB:- It is a good choice for database where number of writing and reading operations are higher and since node provides mongoose as a package it makes it easier to interact with the database. Unlike SQL mongo database stores information in forms of documents instead of tables.

Express JS:- It is a framework based on node which helps in simplifying the server side coding and logic. It provides a simpler interface for creating / requesting endpoints and cookies. Has features for making middlewares and routing table as well as dynamic rendering of HTML pages.

Angular(frontend):- A JS framework for developing single page applications, it comes with rxjs and other easy to use inbuilt libraries to make the rendering of elements easier once understood properly. It is an open source cross platform development tool provided by google.

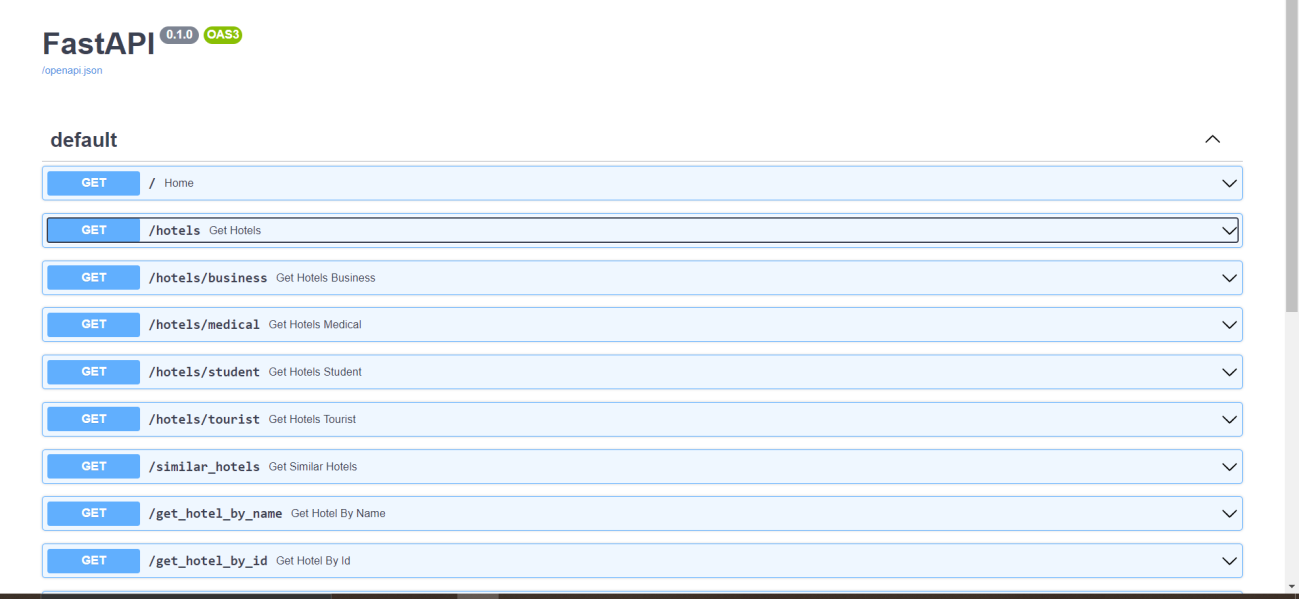


About APIs and RestAPIs

Since the ML team deployed their model on heroku in the form of API. The API is used to fetch the required data , therefore making the application use these APIs to render the right information to users on pressing of the buttons and based on their preferences. API (Application Program Interface) can be defined as set of rules or procedure via which two components communicate. In this case, it is our web app interface the receiver of those required information such as the hotel names and their attributes . API is like an intermediary software or code that allows us to put the required information on webapp by using http library provided in Angular. Just like a weather app fetches information from a server that gives information about weather. Usually data is transferred from a server in the JSON format or XML format or HTML in case of web pages.



RestAPI or Representation state transfer , as the name suggests it only transfers the state of object or instance of object of the information to the client that has requested the server using CRUD ( Create , Read, Update, Delete) commands which in http is actually GET, POST, PUT, DELETE, sometimes PATCH is also used.

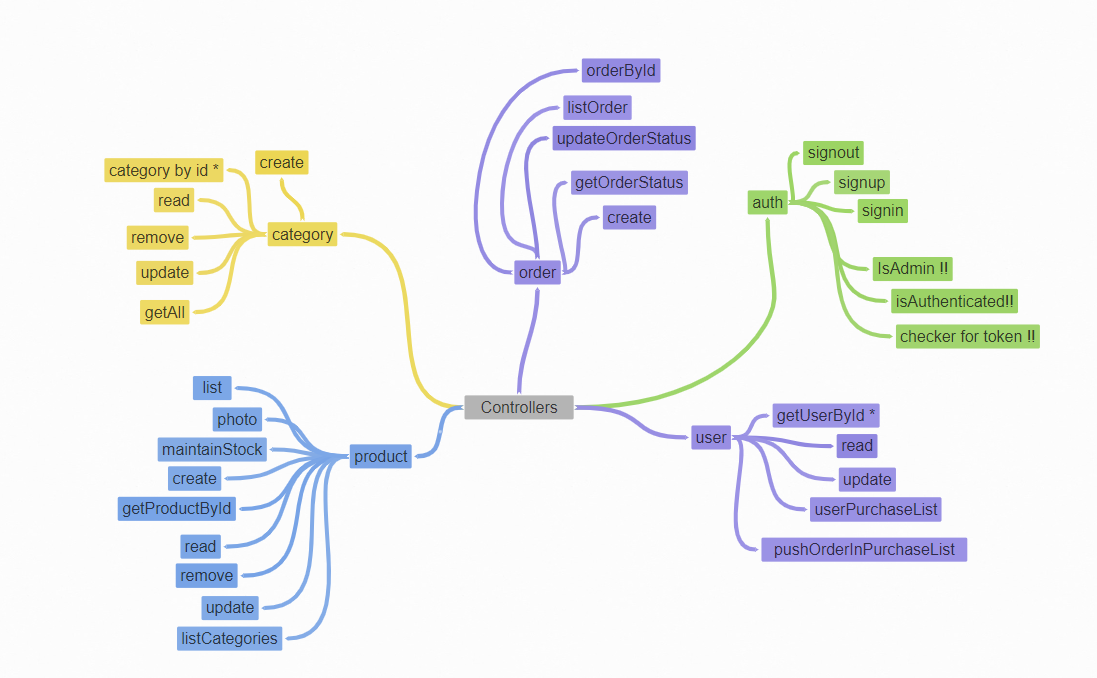


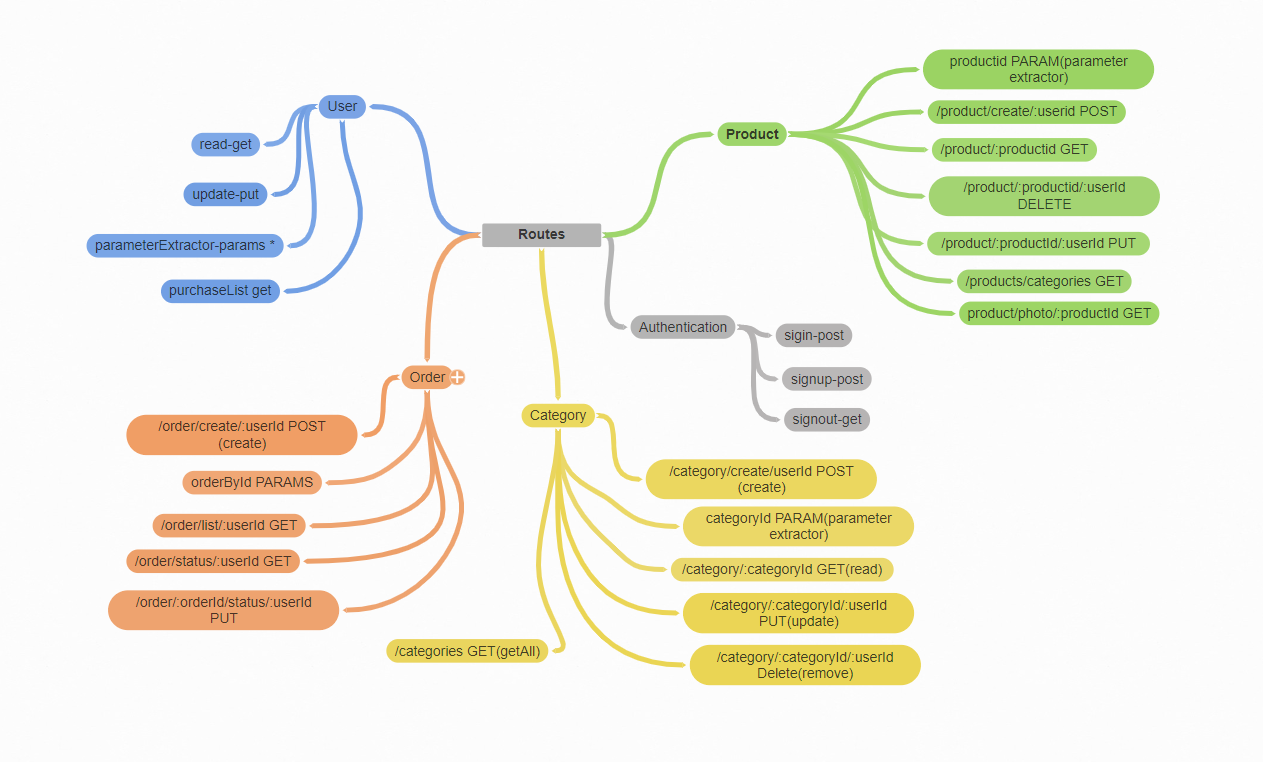
TOOLS AND METHODS

ROADMAP FOR PROJECT

Visual studio code was used as IDE for this project. Install the plugins required. Installing nodemon allowed development faster as any changes made in the code has to be verified by reloading the website again and again. nodemon solves it by allowing developers to make changes to websites without reloading it by just pressing ctrl+s it allowed automatic rendering of the said changes to the website.

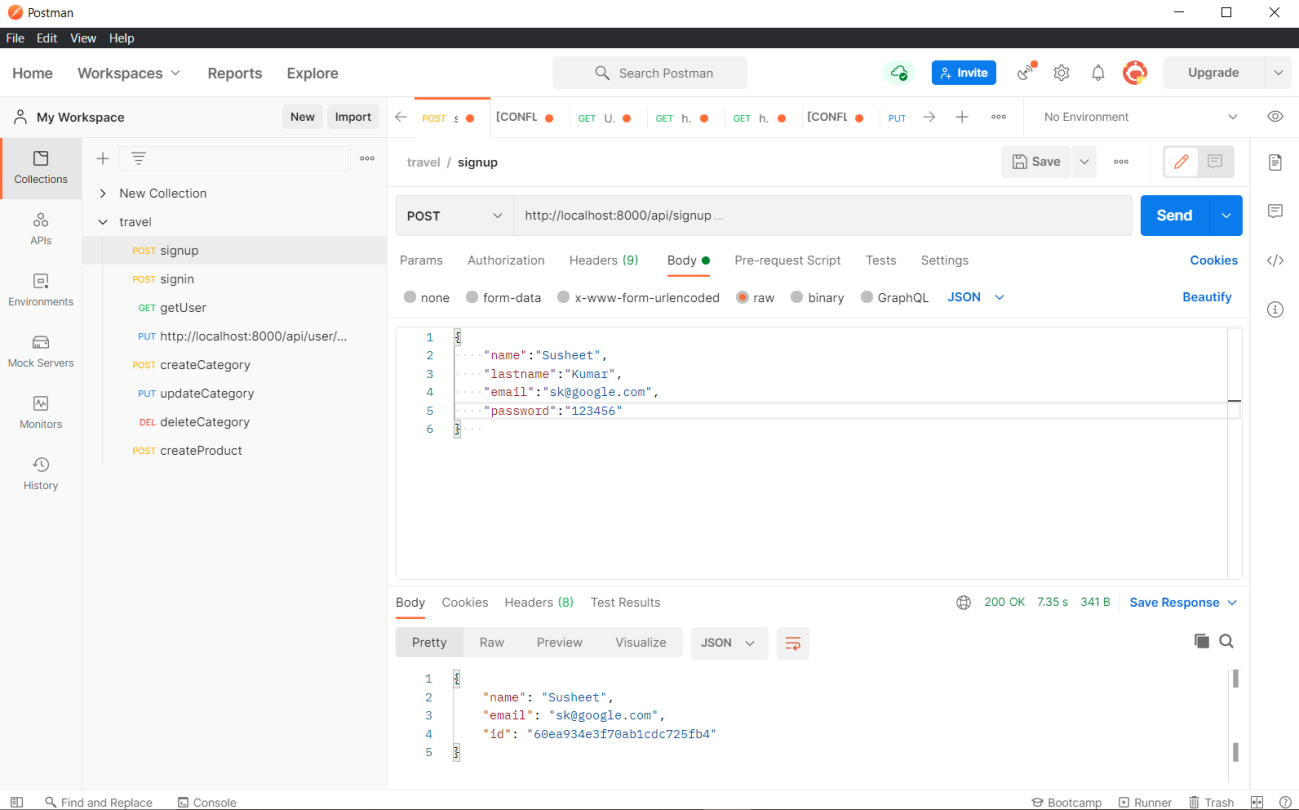
Following the MVC(model view controller) approach we designed routes that will be connected to frontend and the corresponding brainly logic for those routes that is the controllers.



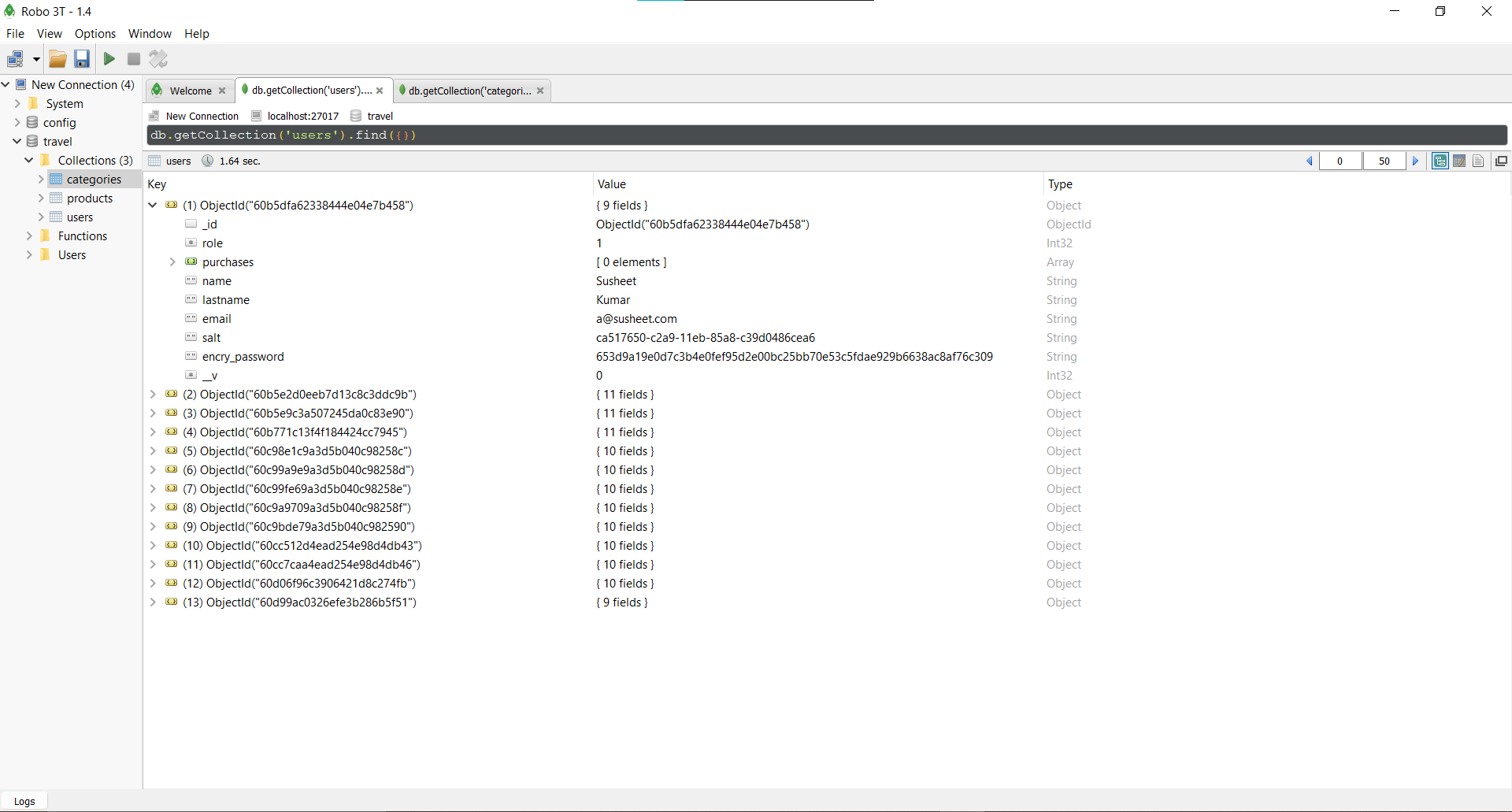


POSTMAN FOR ROUTE TESTING AND ROBO-3T FOR DATABSE

**Postman** was used for building, testing and modifying our API based approach that are to be passed on to frontend when frontend – backend connection happens. It is helpful in interacting with HTTP APIs.



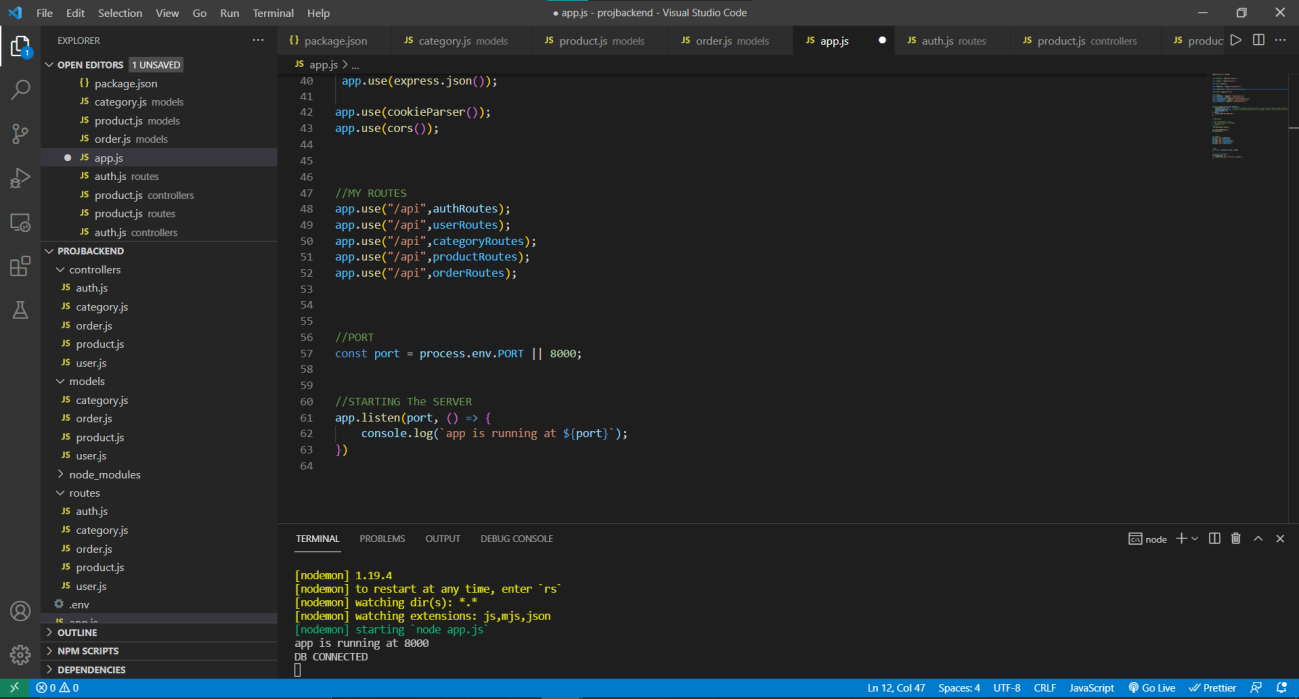
For working on database **Robo-3T**, a visual open source MongoDB GUI which helps in displaying the documents of information that is stored in database.



Apart from above mentioned tools , Google Chrome was used as browser to make the required changes and observe them through chrome’s developer tools that allowed easy inspection as well as seeing results in real time .

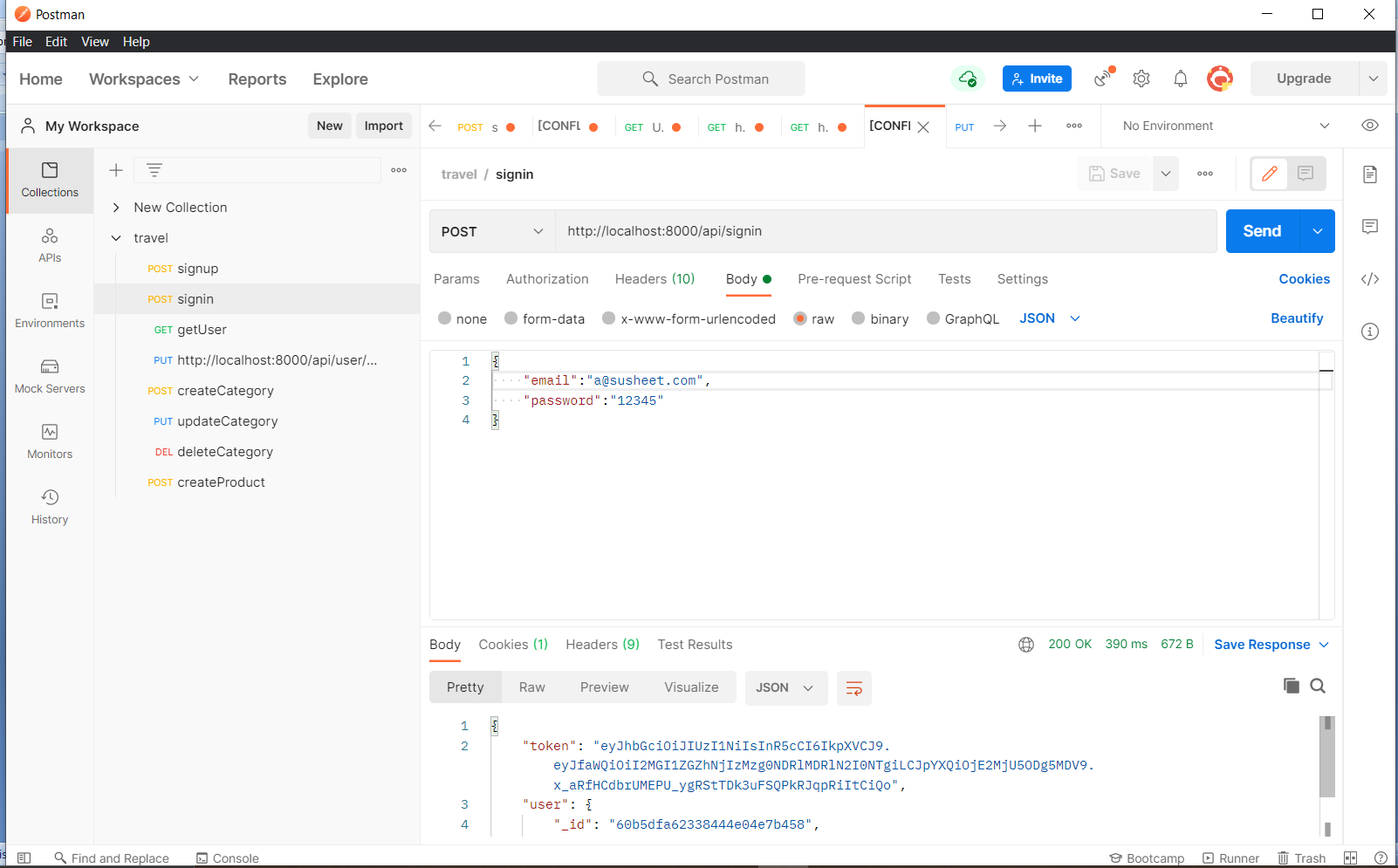
STARTING WITH BACKEND DEVELOPMENT

Following our approach of MVC, the first task was to design the database and hence using the mongoose library provided by expressJS , document design for users, products, orders and categories was made. In any e-commerce website, the product is associated with a category.



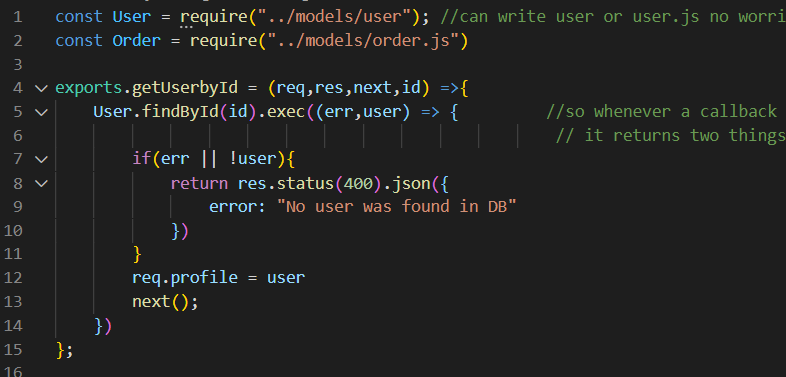
The first major part of our project was designing and implementing an authentication system. Any authentication system works on tokens, whenever a user logins into a system the website stores some cookies inside the browser of the user and whenever user tries to do something on website , the backend server runs checks for those tokens, verifying if the user is authorised to do so or not. Apart from this token system, various middlewares were also designed to keep the privileges of users in check based on their role that is either they are bookers, sellers or admin.

Token and Id generation on successful sign in by the user ( there is a concept of bearer that carries this token for authentication purpose)



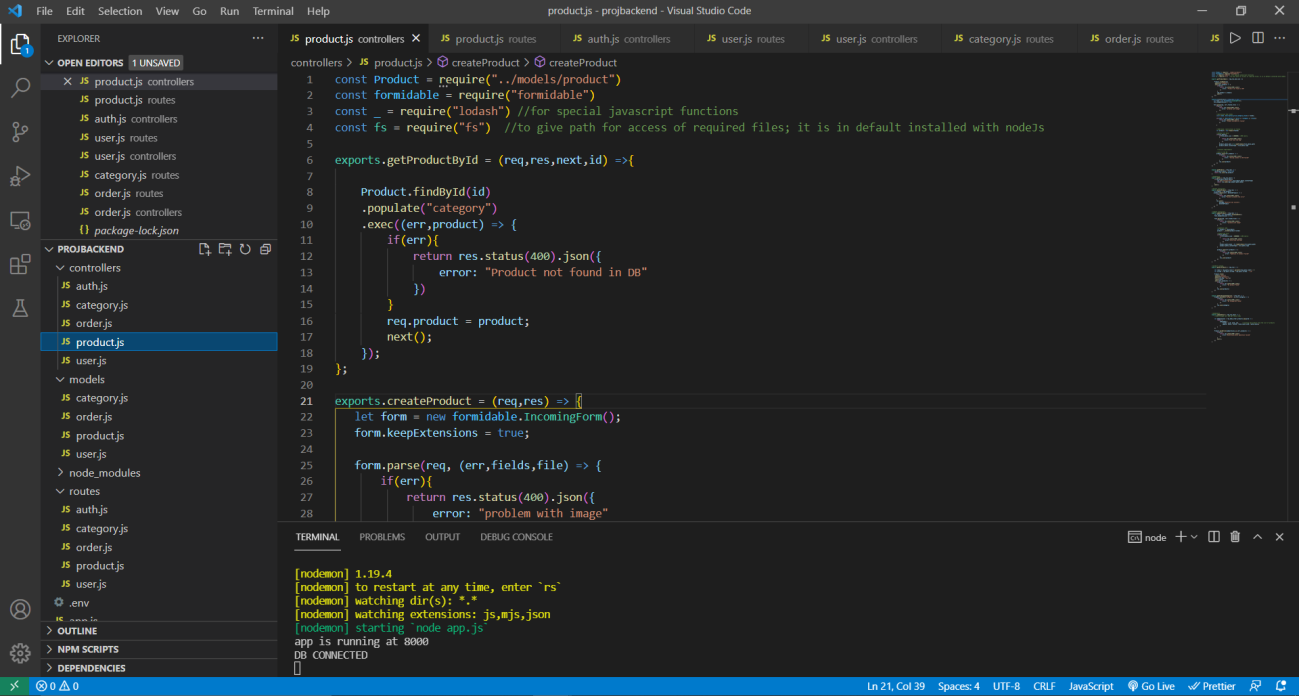
For this purpose various middlewares were made which is provided by express framework. Various third party libraries provided by npm (node’s package manager) such as validators and jsonwebtokens etc were used to make sure proper email , password are sent to server and then tokens are generated based on the ID of the user , the password was also stored in database in encrypted format with the help of another third party library crypto Node provided by node Js.

Interacting with database through backend required mongoose functions , and any time we interact with database though backend, the database returns two objects , one the object itself that is stored in database and other one is the error. This allowed to write easy code for checking if any error is there and also to ensure if the function was executed successfully or not. Various status codes were used at right places that helped in development process and also the customised error messages helped in identifying the problem and rectifying it. For example:-



And since we were targeting a MVP approach for our website, it was decided to use firebase authentication API for easy handling of signing in and signing out process. Firebase further provides the feature of connecting to the backend database and hence making the process easier by not worrying about further details and concentrating on the next part of the project.

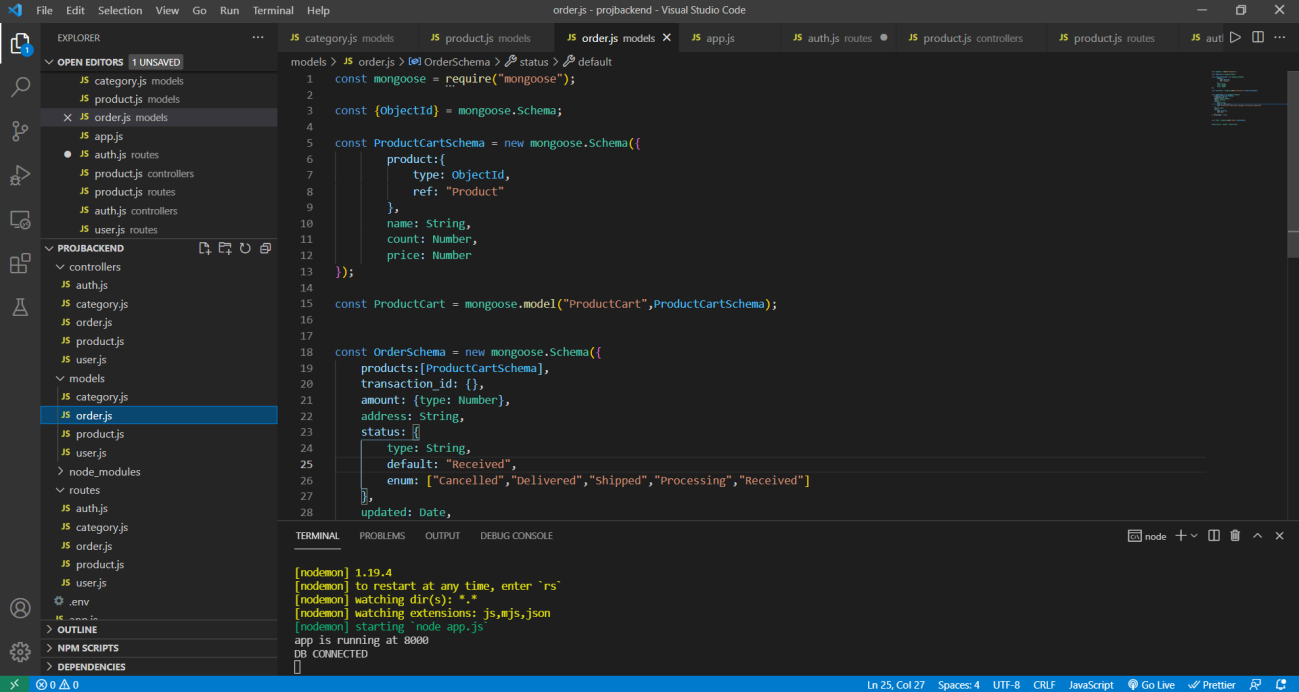
The next part of the this puzzle was to equip the backend for dealing with storing images in a compressed way , storing different form input that will be coming form frontend to backend . For this another third party library lodash a Javascript based library was used that allowed easy manipulation and modifying of objects. Formidable was used for storing form inputs that will be coming from frontend if the project takes into further stages, the website will be ready to do such things. For now our MVP approach was to make sure the website allowes authentication, displaying of the places to stay and attractions based on user preferences and their personas. But the backend was made ready to handle situations in case the website goes into further stage of development.



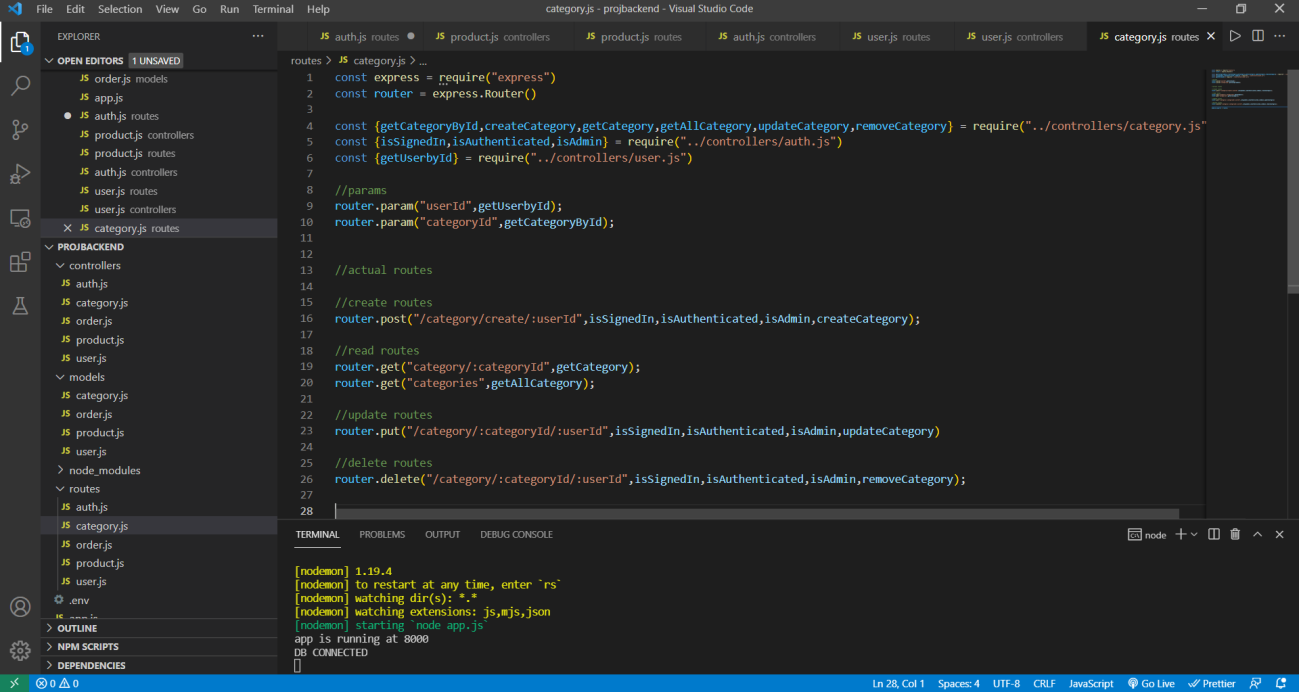
STRUCTURE OF THE WEBSITE

The backend part of the website is in accordance with the model view controller approach so , for example when a route for creating categories is written , a corresponding controller file is written for executing this in the controller folder. This similar pattern approach was followed for all CRUD operations and not just for category but also products, orders.

Another part of the problem was connecting the various documents in the database , such as categories need to be associated with a product, user need to be associated with order. And hence with the help of mongoose library a keyword called ref : object Id was used to connect these schema with each other.



After this the process was iterative , It was just inserting proper middlewares into the routes and making them secure and corresponding code for those routes were written in their controller files



In the controller files , first of the all the object was populated based on the request body that was send by frontend or in my case postman. Populating the req.body with required object and then required operation was performed depending on the routes accessed.

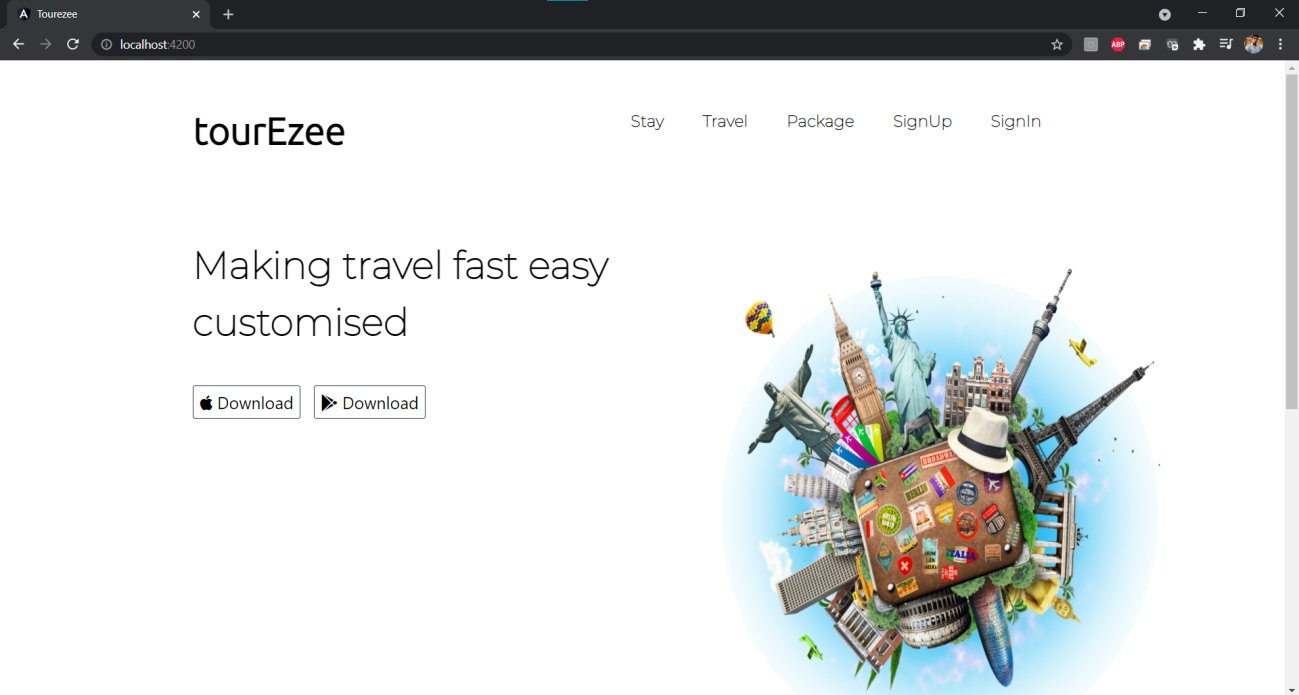


Similar approach was followed for all other backend components such as user, order and product.

2.4.1 The Frontend (Angular’s entry)

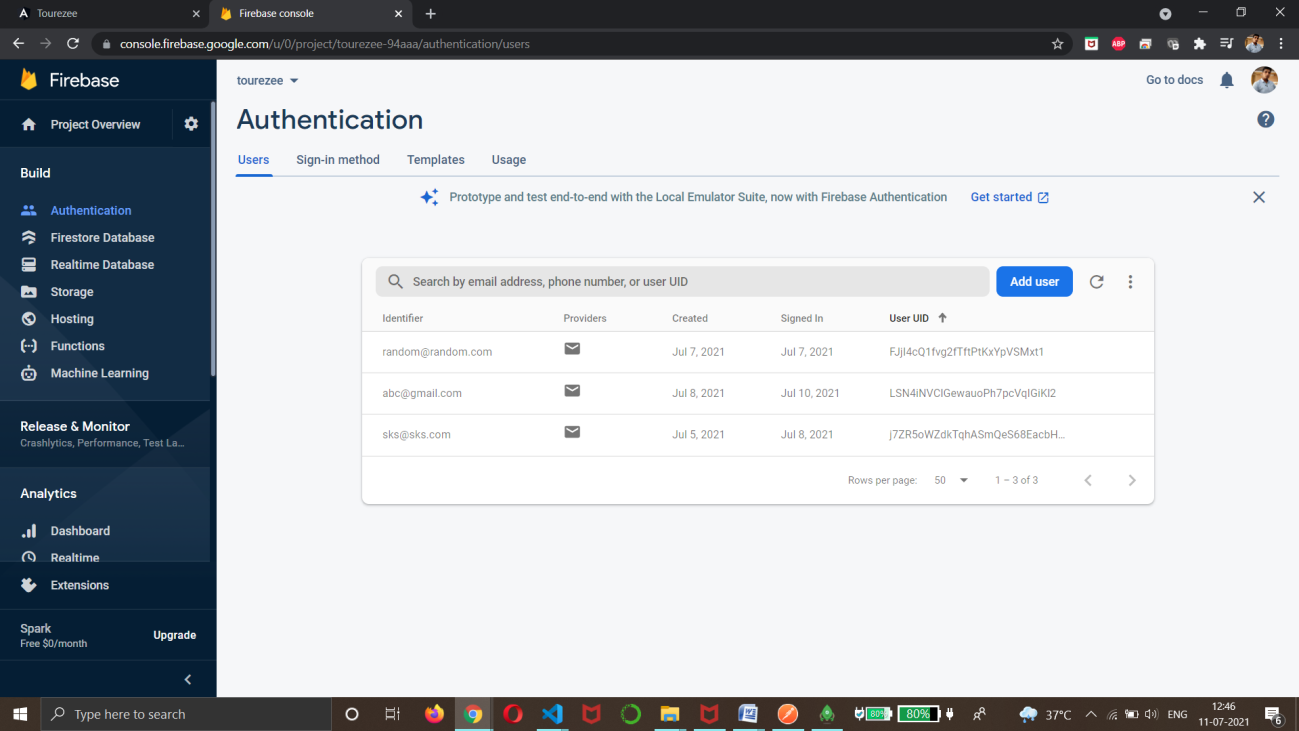
The next part of the project was designing the frontend as well as connecting the frontend to backend. Angular at default runs on port 4200 with a default page showing buttons for generating components or services. Following our mentor’s approach for app first, the home page of website consists of links for downloading the app for both android as well as iOS along with a header that shows options for signup signin and stay buttons that fetches the information from APIs and displaying that information in an aesthetic way . Since angular provides inbuilt support for routing and routing table, no third party library was used . Apart from regular default packages that are provided by angular some additional libraries were also installed using npm in this angular frontend part of the project.

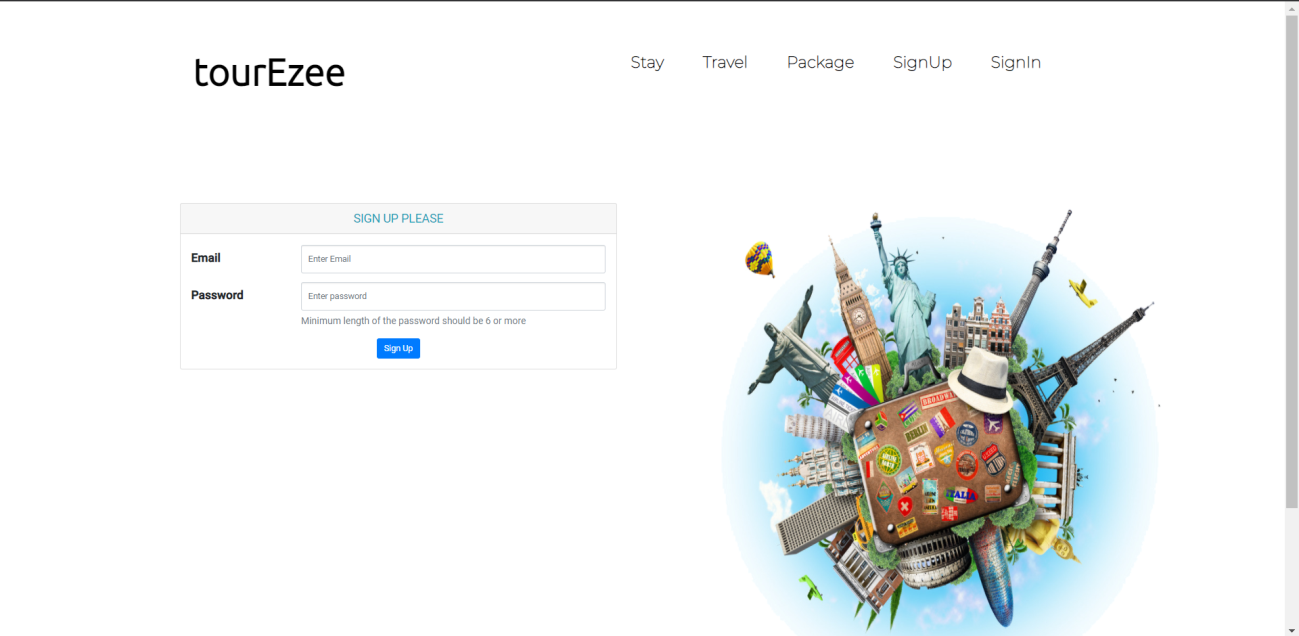
These include bootstrap (this library provides attractive and beautiful css designs and buttons and easy designing of components and services), ngx-toastr to show the messages such as successful or failed.



The front page of the website

For the sign in and sign out using google’s firebase , @angular/fire was used that took care of the authentication part of the website. This also worked like an API, by storing the API key in environments folder(provided by default by angular) of the angular project and writing the required functions. Google provides sparks which is a free plan as well as it provides a lot of analytics features for analysing the data .



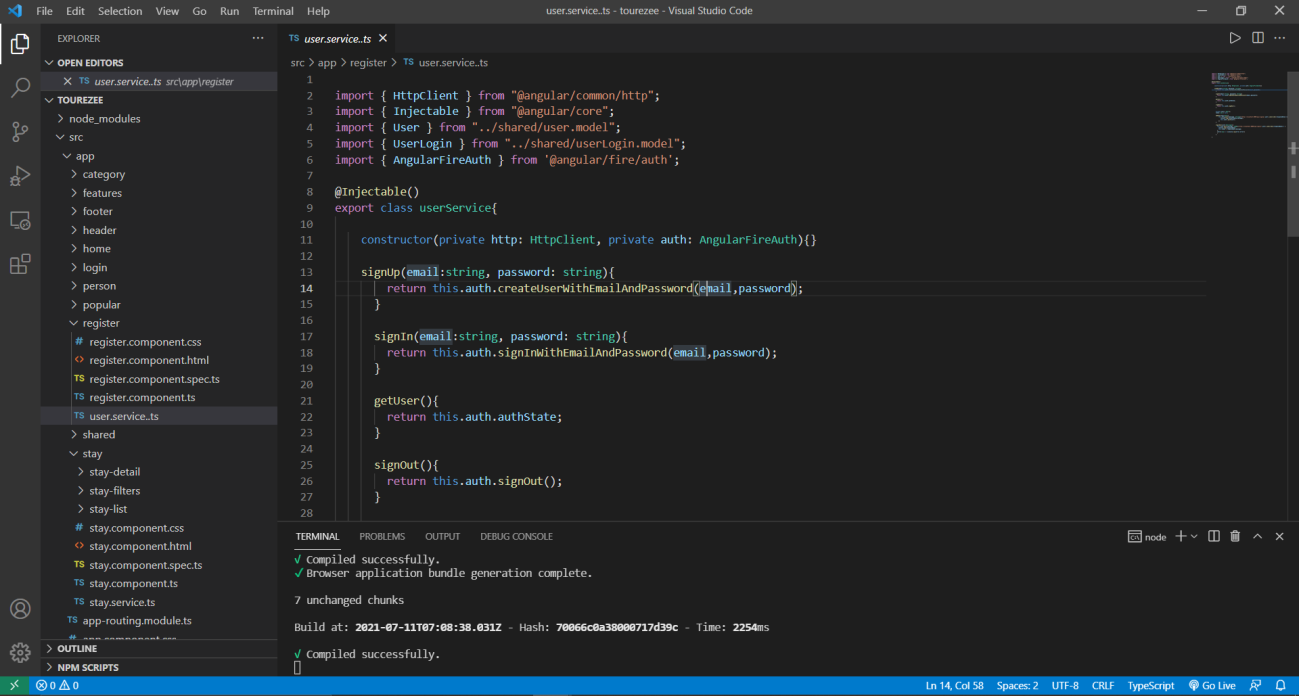


A simple sign up and sign in page that required only email and password(following our MVP way of keeping things to minimal)

Connecting frontend to backend

(further stage of development)

Since the goal was to go for a mvp product that does minimalistic things, but the need for connecting frontend with backend will eventually arrive. Hence services components were generated in required components file structure of angular so that from there http calls can be made to backend to execute the required function.



At last with more time and with better experience and tools , this project can become a full fledged website.