Trip Genie

CS 175 Final Report

Presented to

Prof. Yan Chen

Department of Computer Science

San Jose State University

In Partial Fulfillment

Of the Requirements

for the Class

Spring-2024: CS 175

By

Team 7

Bhargavi Chevva

Ketan Jadhav

Jovian Jaison

Deep Shah

Trip Genie is a travel app that allows users to effortlessly plan their next adventure. Our app has features that enable users to search among various tourist spots and book a package to get the best experience conveniently.

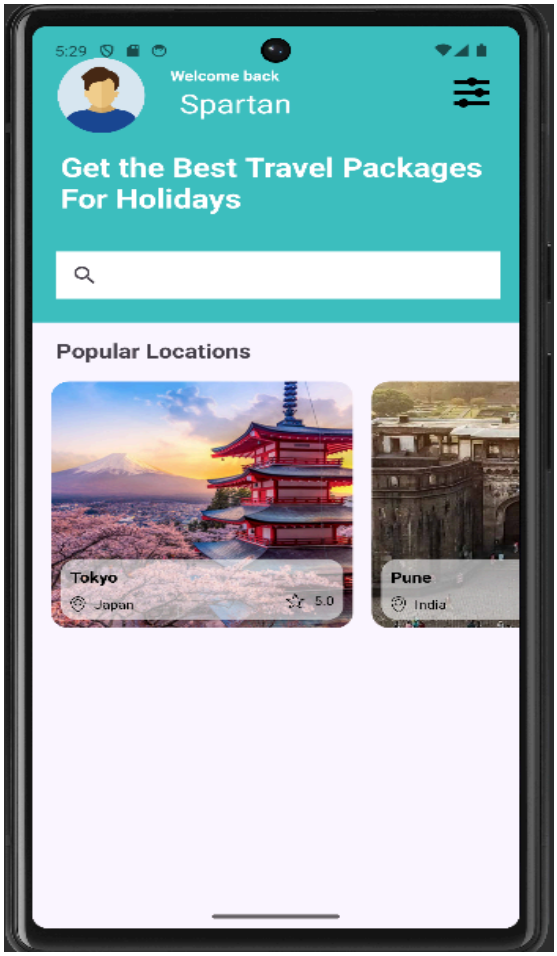
We have created a frontend in Android Native using Java and used APIs for all of our CRUD operations to the database.

Tech Stack:

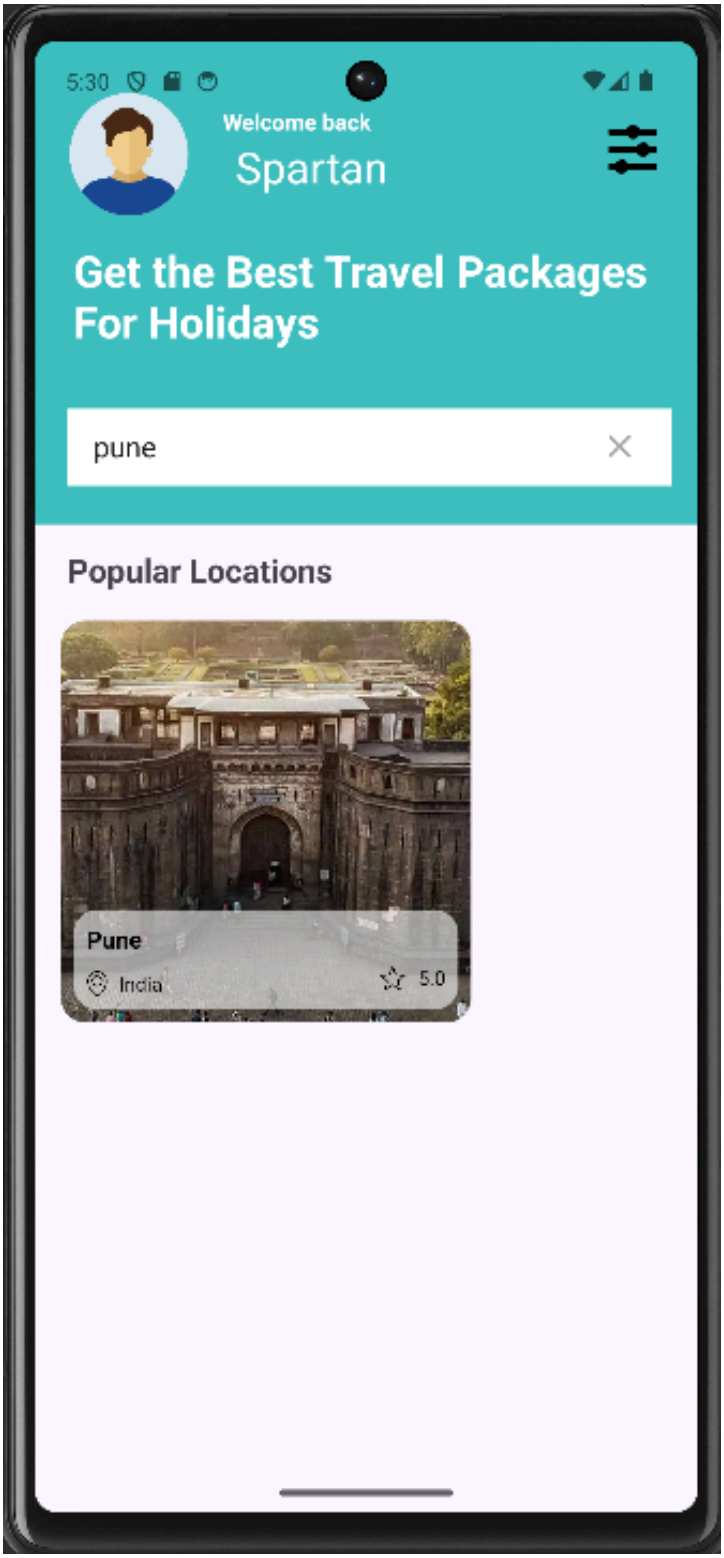
1. Java
2. MongoDB Atlas
3. Postman

Application Functionality:

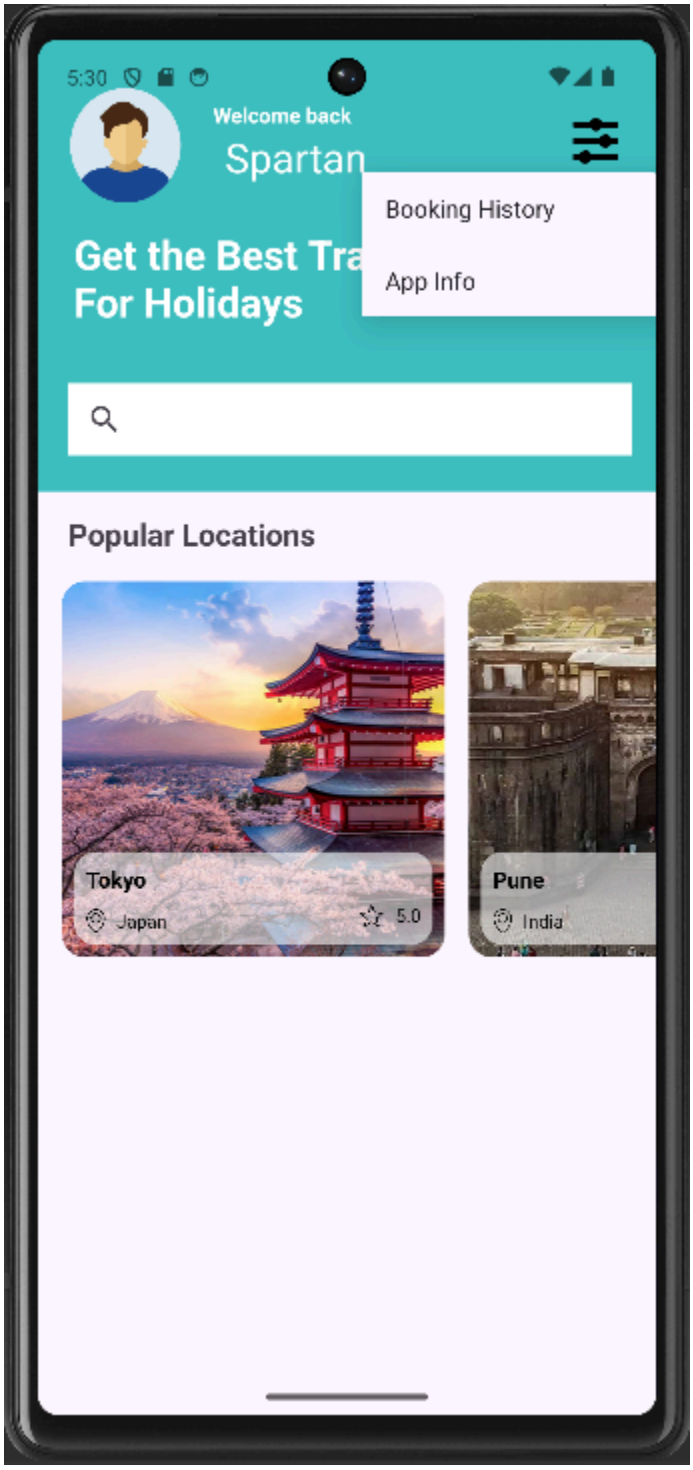
1. The home screen of our application displays the username, search bar, and various popular travel destinations to choose from.

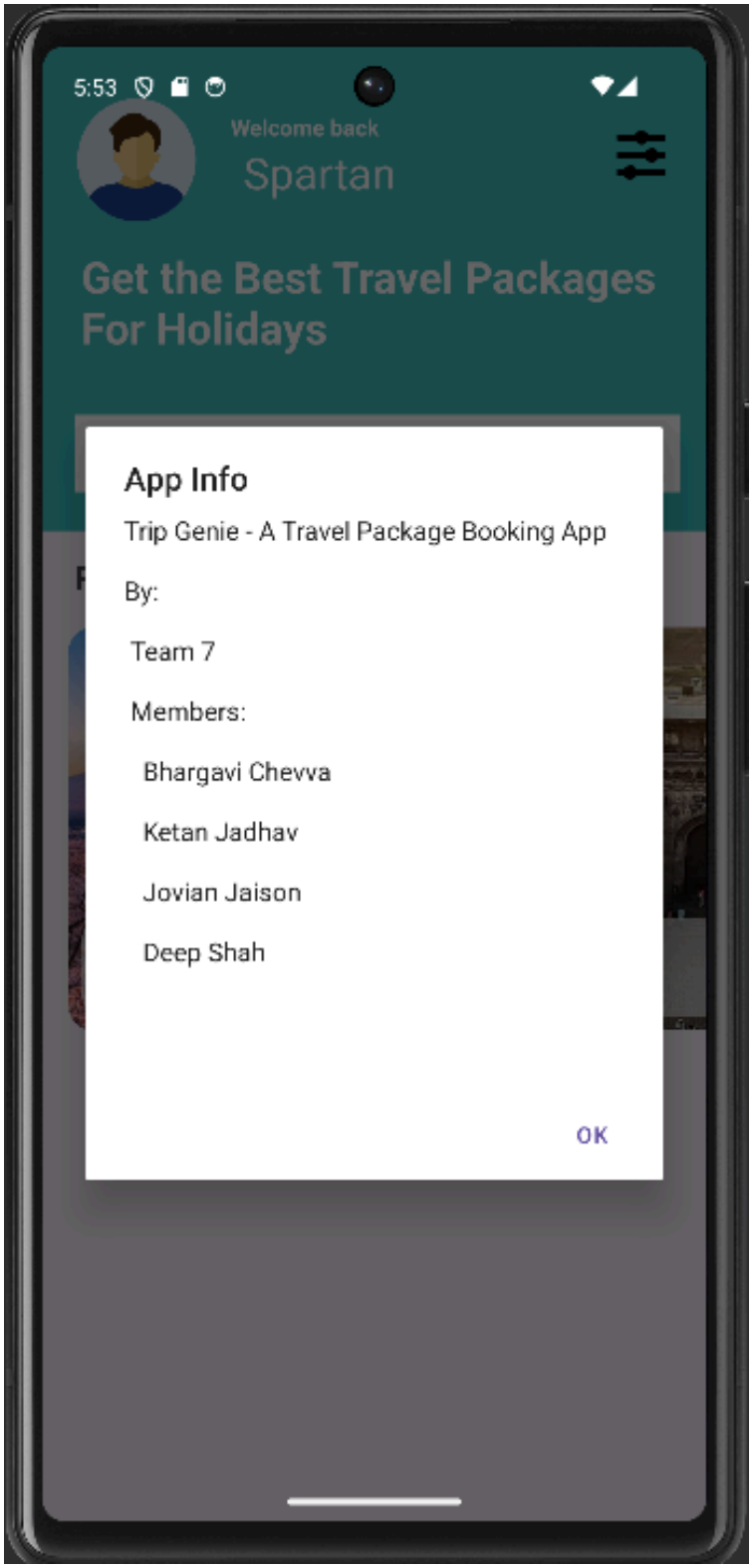


2. The search bar filters the destinations based on the user's query.

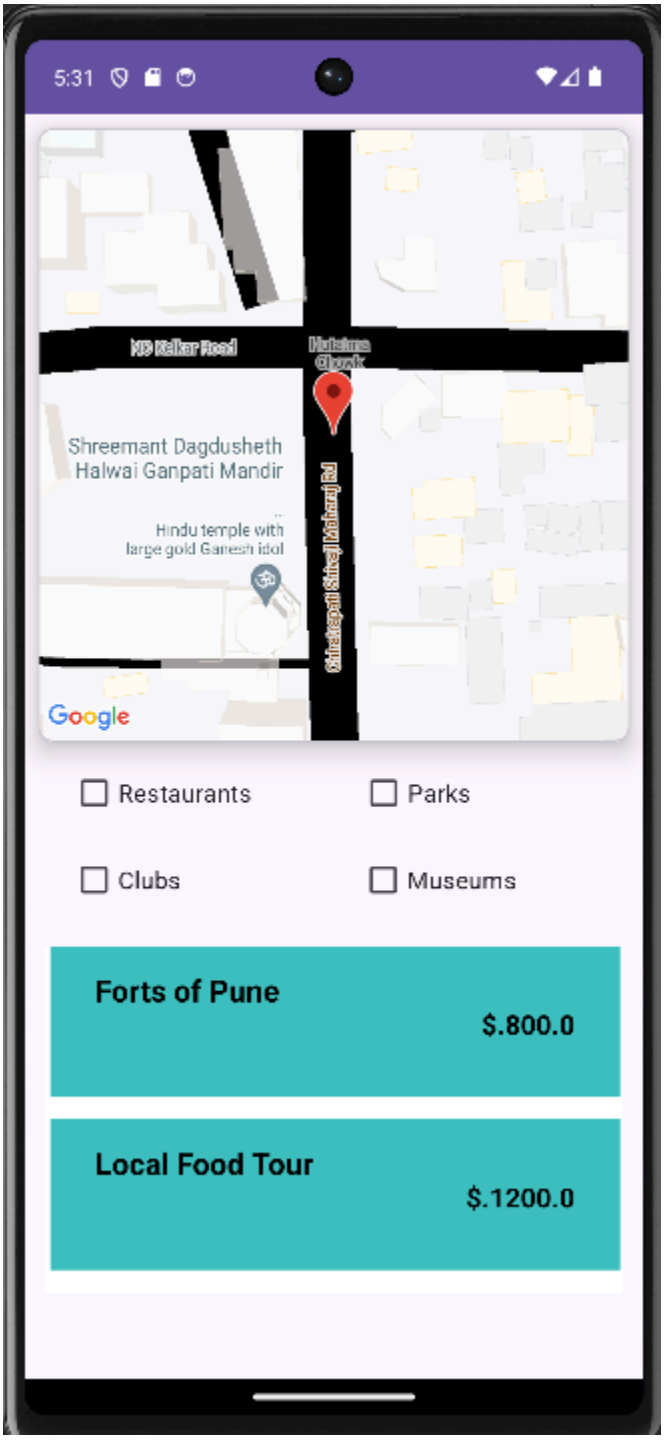


3. The home screen also consists of an expandable menu bar to view past bookings (maintained on MongoDB) and App information.

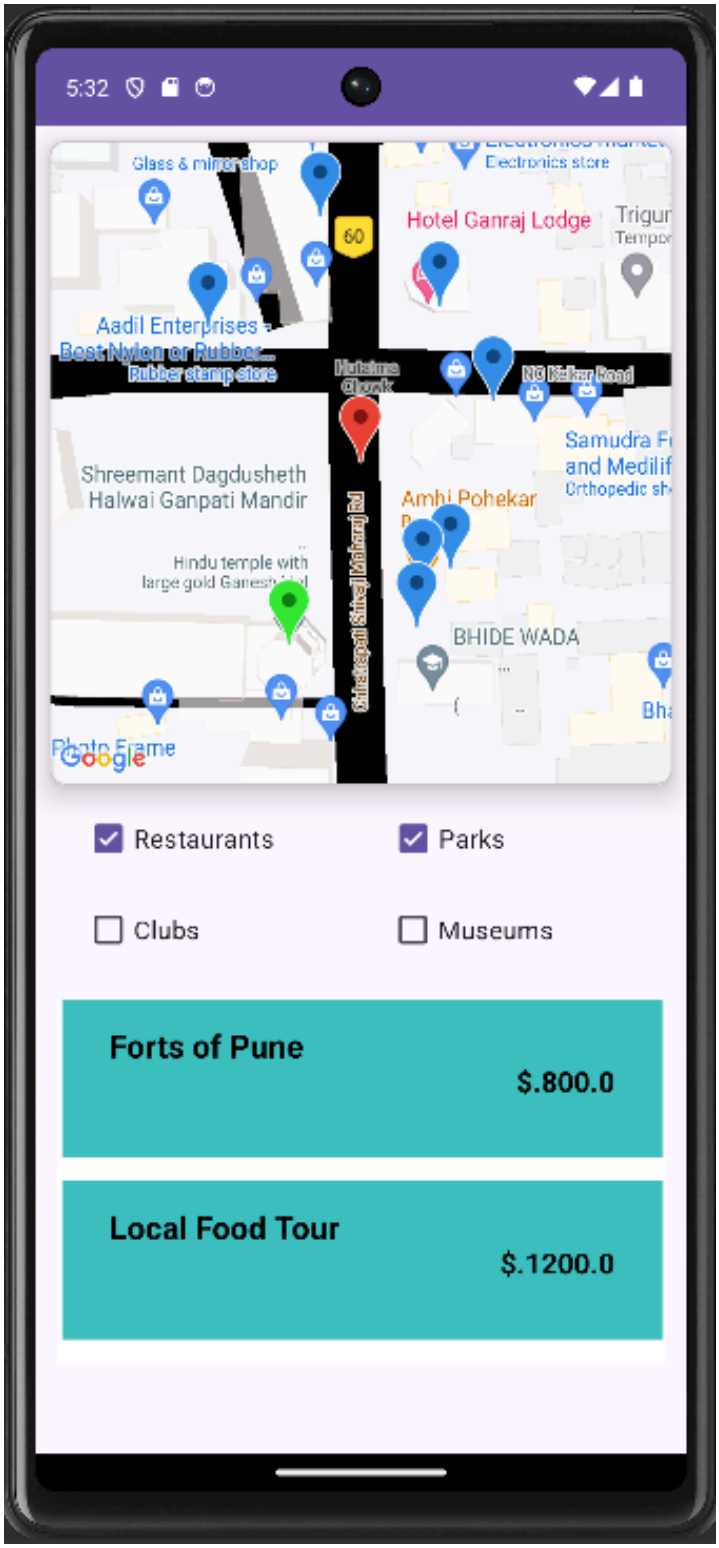

5:30
Welcome back
Spartan
Booking History
App Info
Get the Best Tra
For Holidays
Popular Locations
Tokyo
Japan
5.0
Pune
India


5:53
Welcome back
Spartan
Get the Best Travel Packages For Holidays
App Info
Trip Genie - A Travel Package Booking App
By:
Team 7
Members:
Bhargavi Chevva
Ketan Jadhav
Jovian Jaison
Deep Shah
OK

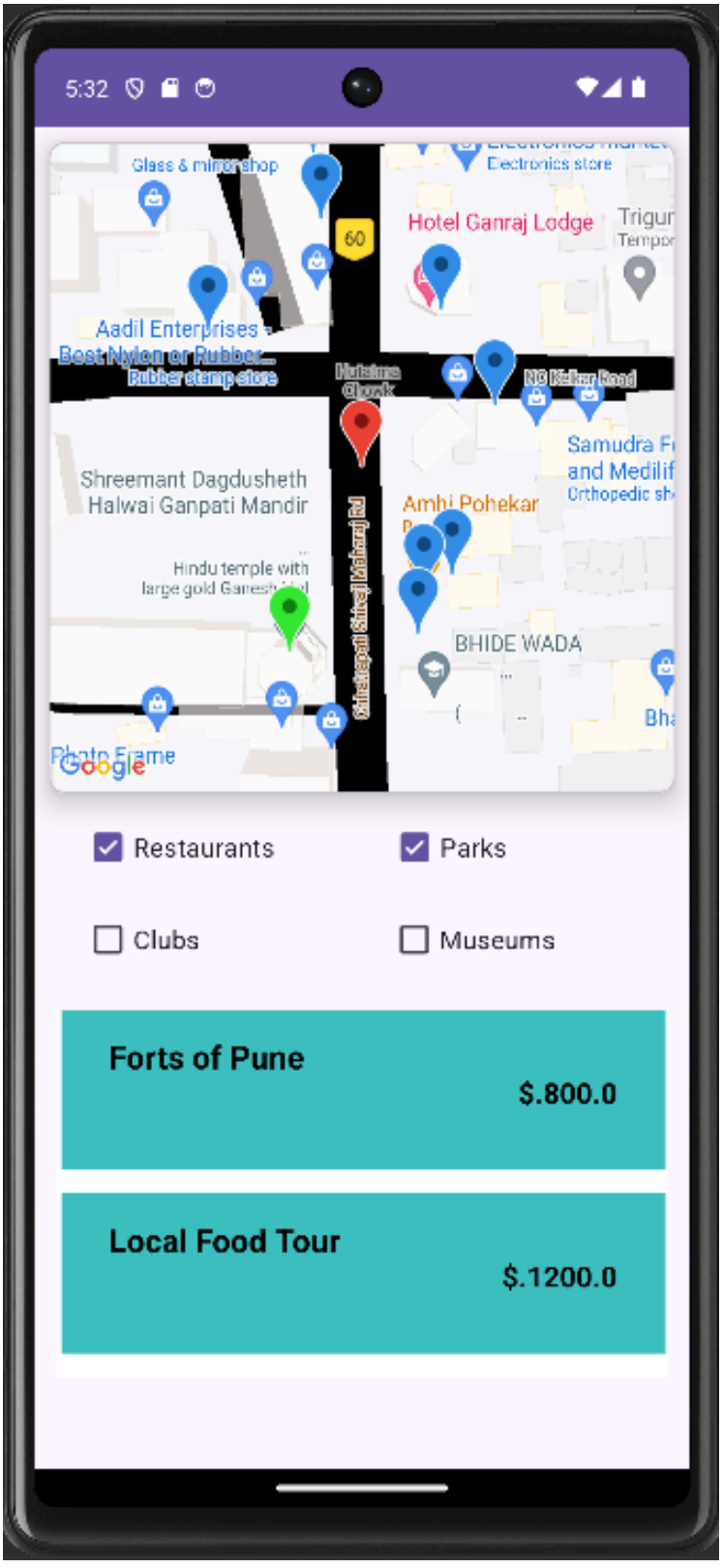
4. When a user clicks on a travel destination, the ‘Details’ page opens up on the screen. It consists of map view, check boxes to look for nearby locations, and a list of travel packages available for the chosen travel destination.



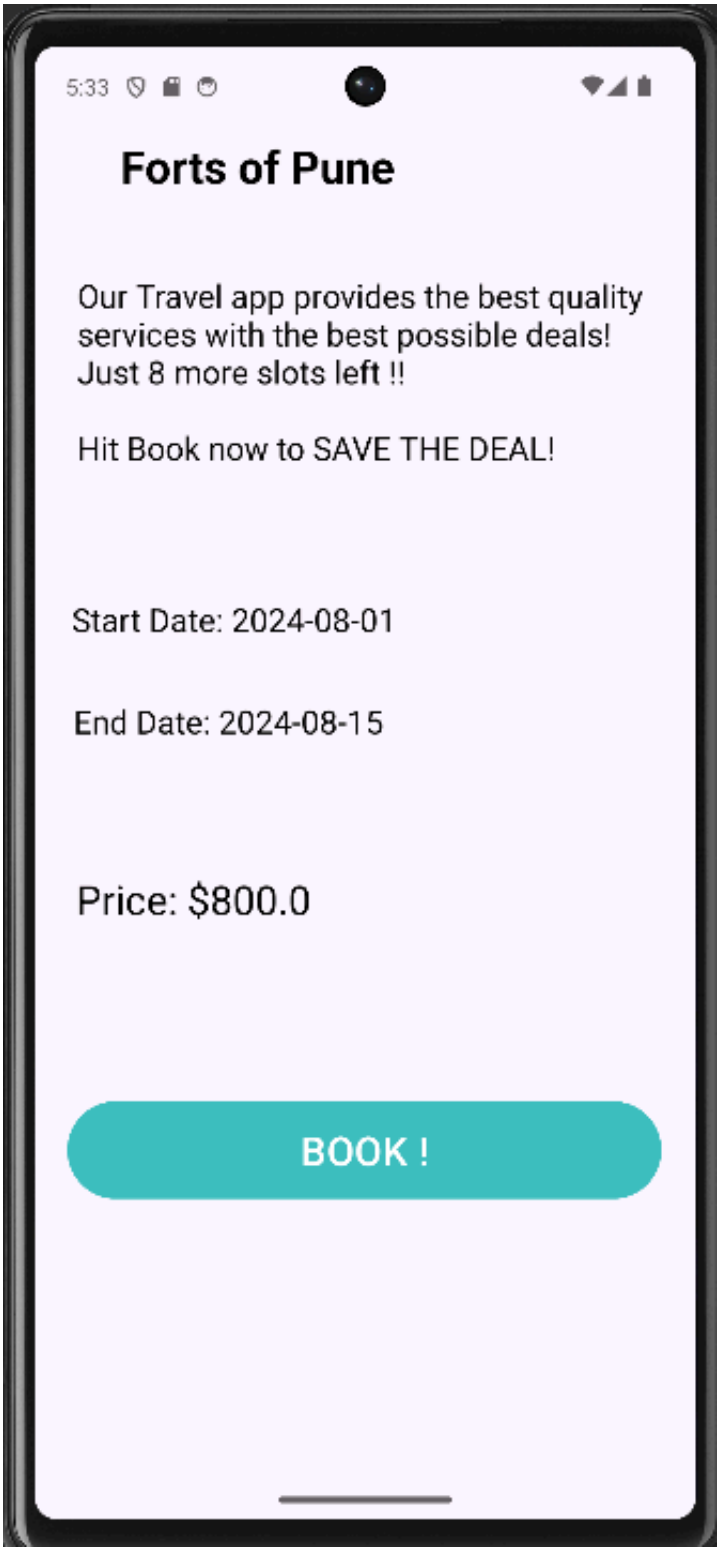
On choosing a checkbox, for example, restaurants, all nearby restaurants within a radius of 10 kilometers are marked on the screen. We color-coded markers for different queries. For example, markers for restaurants are in blue, parks are denoted by green markers as denoted in the below screenshot.



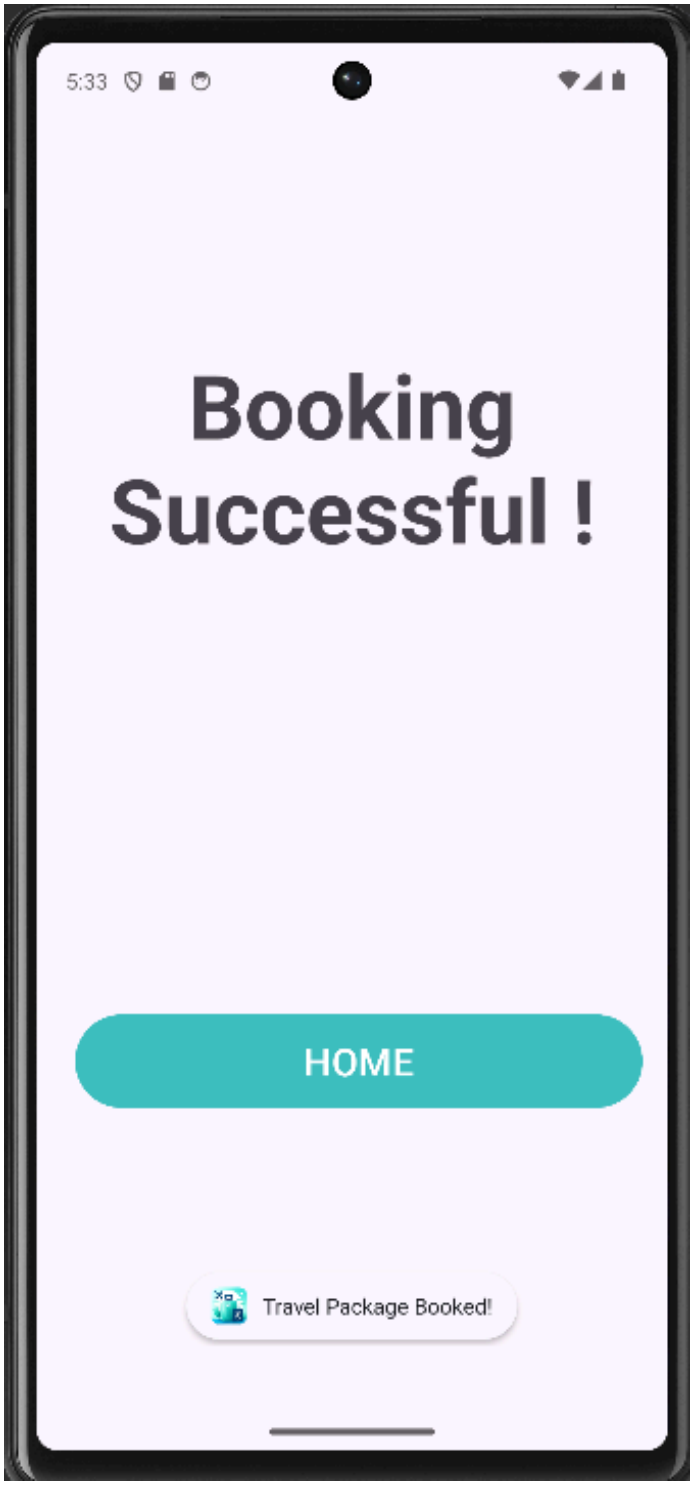
5. Below the map, various travel packages available for the travel destination appear on the screen along with their price.


5:32
Glass & mirror shop
Electronics store
Hotel Ganraj Lodge
60
Aadil Enterprises
Best Nylon or Rubber...
Rubber stamp store
Shreemant Dagdusheth
Halwai Ganpati Mandir
Hindu temple with
large gold Ganesh
Amhi Pohekar
BHIDE WADA
Google
Restaurants
Parks
Clubs
Museums
Forts of Pune
$.800.0
Local Food Tour
$.1200.0

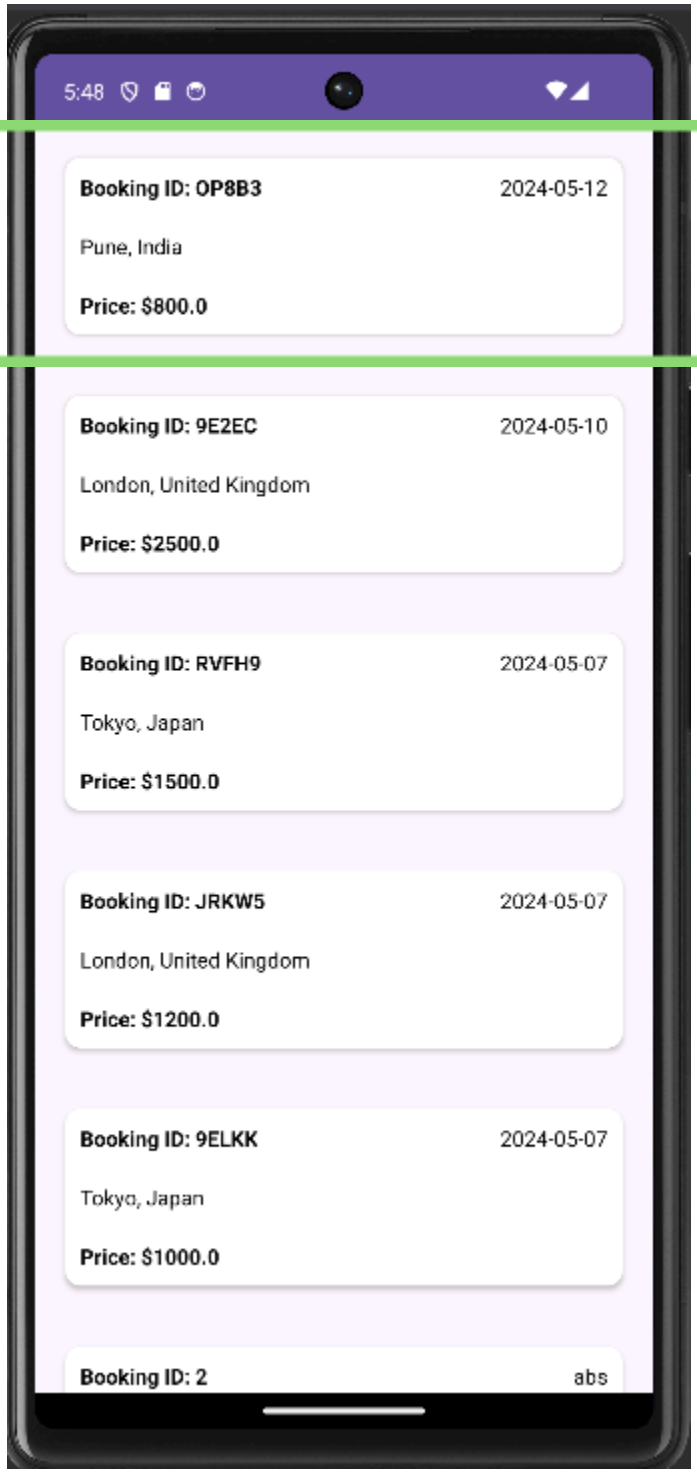
6. On choosing a travel package, further details about the travel package including their start date, end date, and price details appear on the screen. The user can click on the ‘BOOK’ button to book the travel package.



7. When the user books a travel package, the user's details along with the travel package details are added to the bookings table stored on the database. A button that enables the user to go back to the Home page also appears on the screen.



8. The booked packages appear on the Bookings option in the menu bar on the Home screen as displayed in the figure below. Our recently booked Pune package (highlighted in green) is visible now.

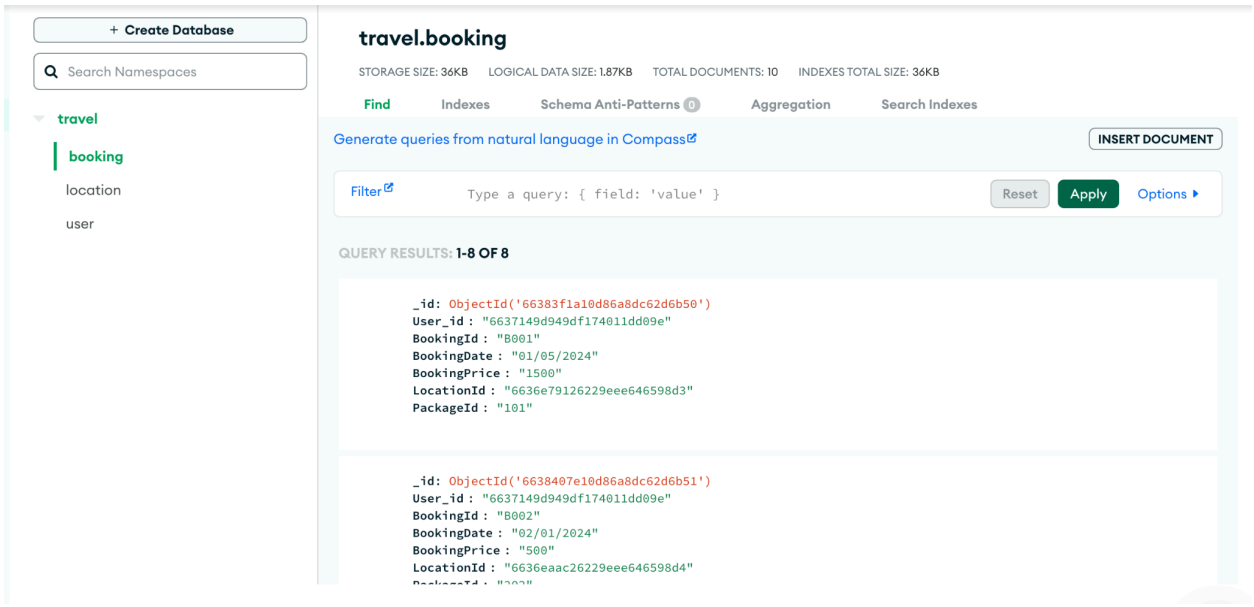


Data Model:

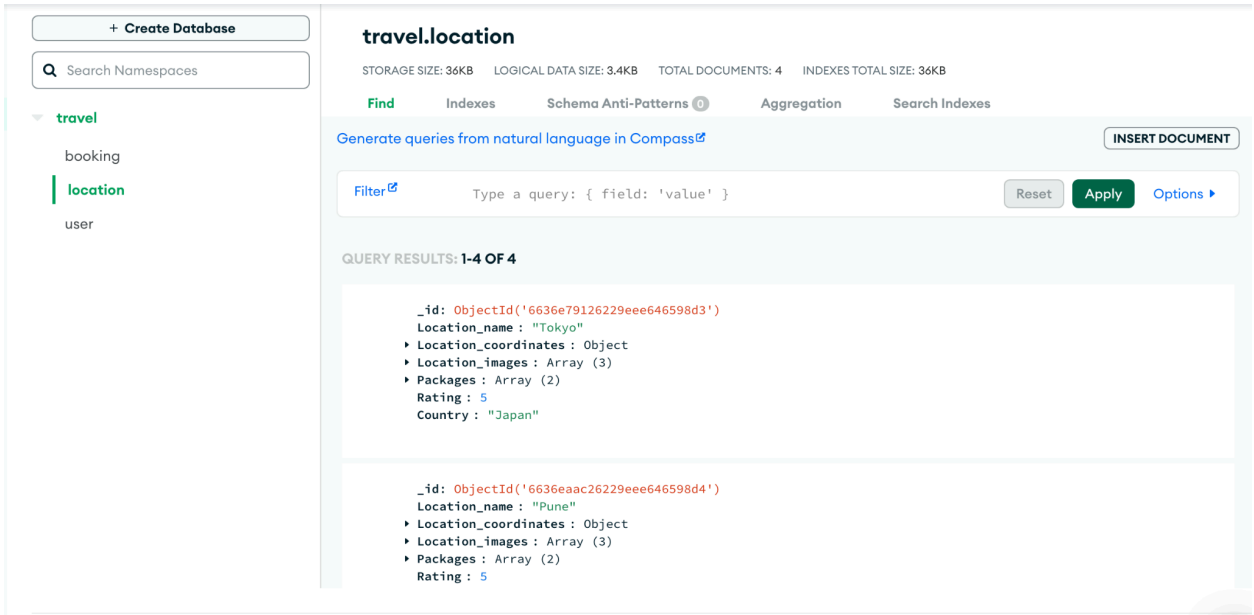
We used MongoDB's Atlas as our cloud-based NoSQL database.

We have a single database called 'travel' with 3 collections inside it:

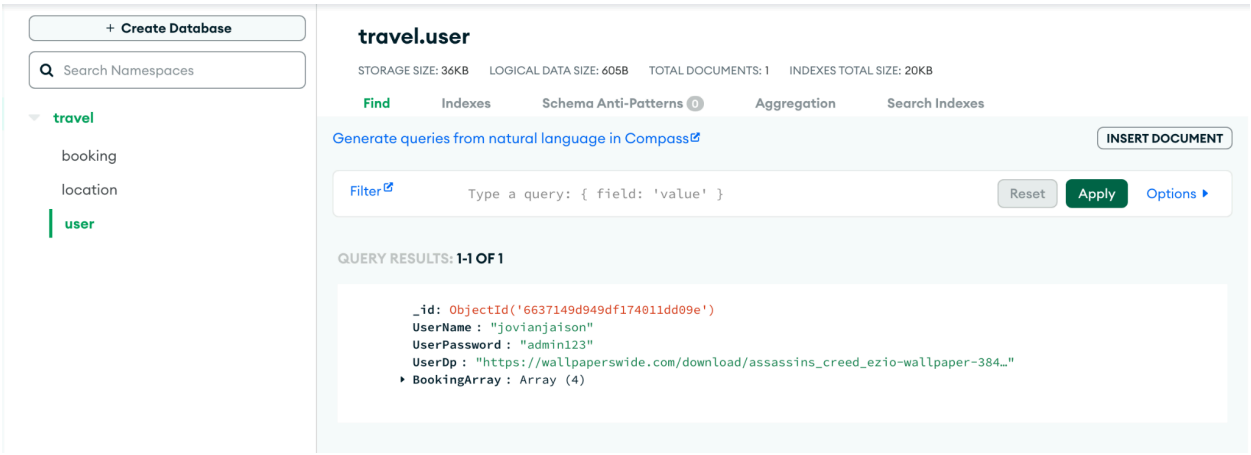
1. booking



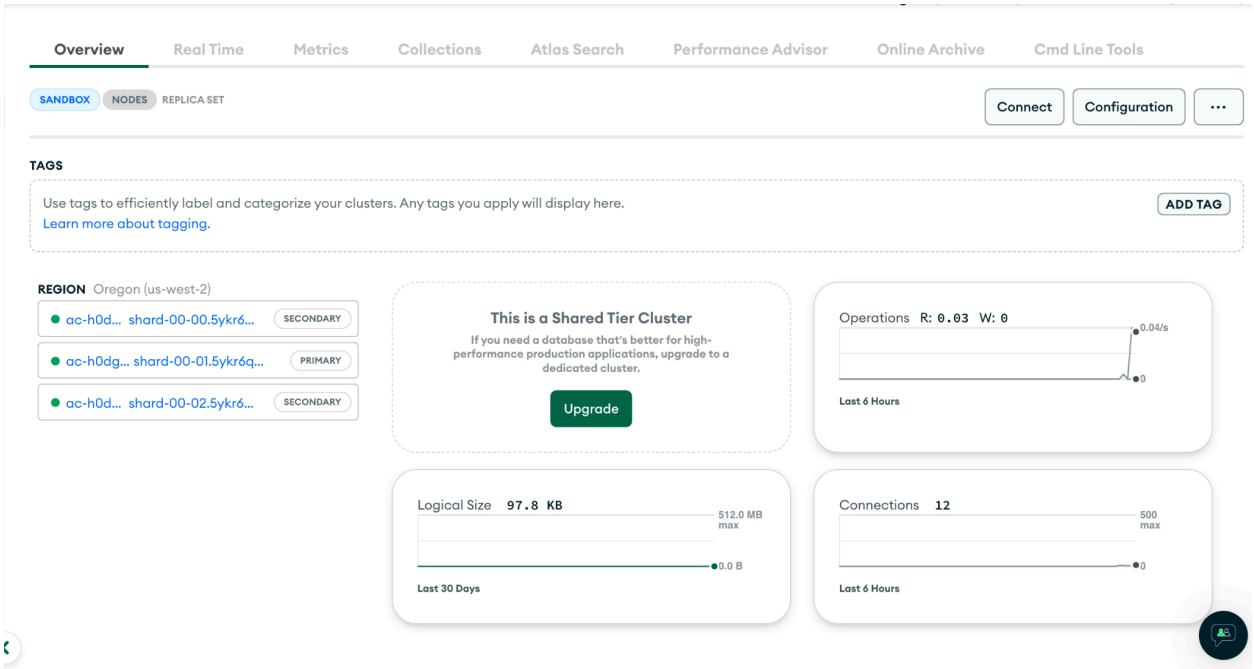
2. location



3. user



We have ensured that our database is sharded, this allows for better scalability and fault tolerance.



Backend:

We utilized MongoDB Atlas's new App Services with Custom HTTPS Endpoints and Functions to create our APIs. These are as follows:

Custom HTTPS Endpoints

HTTPS Endpoints | Data API

Add An Endpoint

Search Endpoints: Search for an endpoint

Search Endpoints: Any HTTP Method

Endpoint Route	HTTP Method	Linked Function	Status	Last Modified	Actions
/booking	GET	getBookingForUser	Enabled	05/06/2024 02:39:54 UTC	•••
/location	GET	getLocationById	Enabled	05/05/2024 03:24:36 UTC	•••
/location/all	GET	getAllLocations	Enabled	05/05/2024 03:37:27 UTC	•••
/booking	POST	putBooking	Enabled	05/05/2024 05:37:59 UTC	•••

Functions

Functions | Dependencies

Create New Function

Search for a function

Function Name	Last Modified	Actions
getAllLocations	05/05/2024 04:27:32 UTC	•••
getBookingForUser	05/06/2024 02:48:24 UTC	•••
getLocationById	05/05/2024 03:34:15 UTC	•••
putBooking	05/06/2024 04:07:29 UTC	•••

getAllLocations

No Changes •••

Function Editor | Settings

Add Dependency

```
// This function is the endpoint's request handler.
exports = function({ query, headers, body}, response) {
    const { _id } = query;
    const result = context.services.get("mongodb-atlas").db("travel").collection("location").find();
    return result;
};

```

getBookingForUser

No Changes •••

Function Editor | Settings

Add Dependency

```
// This function is the endpoint's request handler.
exports = async function({ query, headers, body}, response) {
    const { _id } = query;
    const result = await context.services.get("mongodb-atlas").db("travel").collection("booking").find({ User_id: _id });
    return result;
};

```

Future Work:

For the simplicity of the application, we did not include user authentication and payments. We can add them in the future as an extension to the application.