Software Design Document (SDD)

CS 465 Project Software Design Document

Version 1.0

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	<11/11/2025>	Nneka Hamilton	<Brief description of changes in this revision>

Instructions

Fill in all bracketed information on page one (the cover page), in the Document Revision History table, and below each header. Under each header, remove the bracketed prompt and write your own paragraph response covering the indicated information.

Executive Summary

The Travlr Getaways project is all about making full-stack web apps that are easy for tourists and the company's admin team to use. To make it quick, dependable, and simple to use. The MEAN stack includes MongoDB, Express, Angular, and Node.js.

The part of the site that customers see will let them look at vacation destinations, see trip information, and send booking requests. Staff at Travlr will be able to easily manage content with the admin single-page application. They won't have to update the coding on the website to add, change, or eliminate trips.

Node and Express handle the flow of data, while MongoDB keeps track of the trip information. The MEAN stack links everything together. This configuration keeps the information in the right order.

Design Constraints

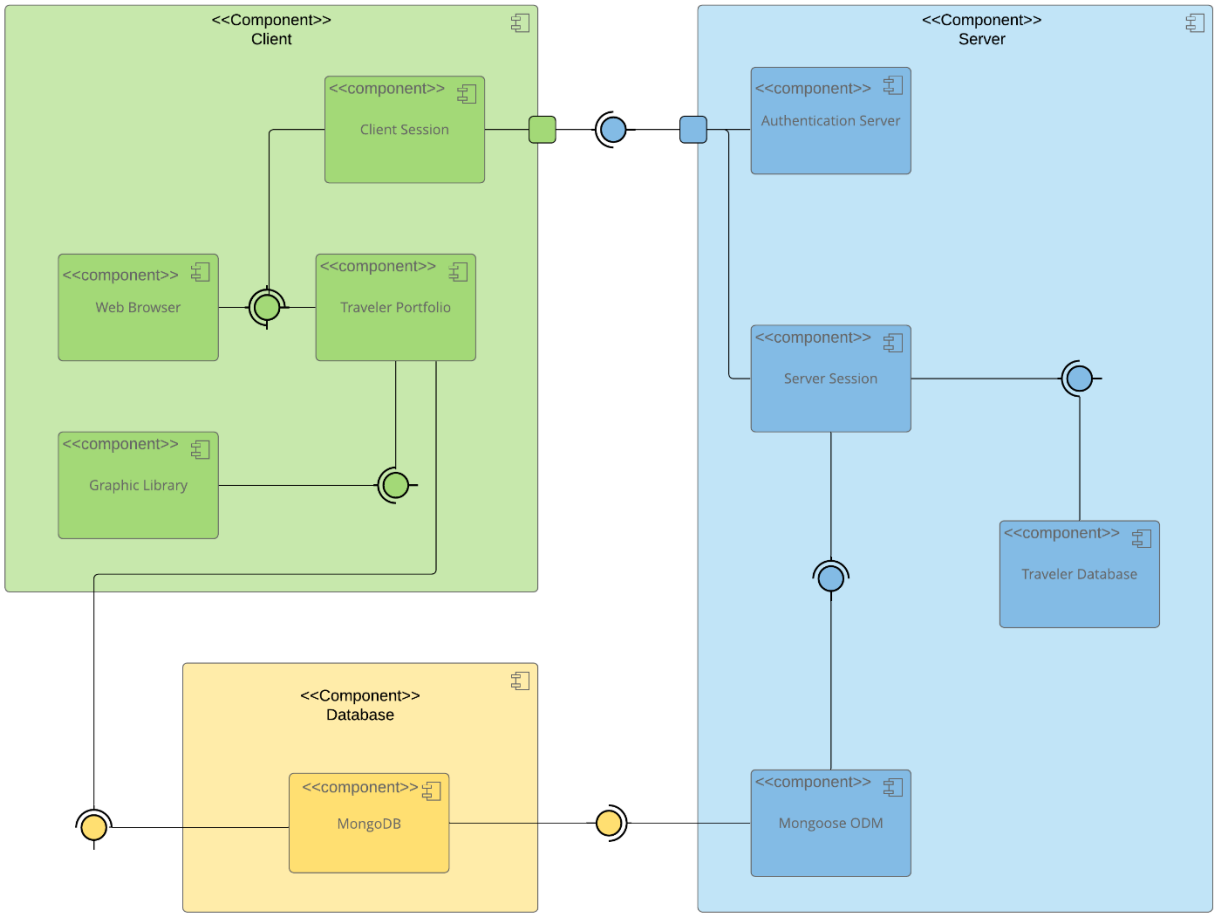
The MEAN stack shows that JavaScript runs all of the app's code. I can't utilize any other languages or tools, even if it means that everything stay the same. I have to make and test everything too so that the site looks excellent on phones, tablets, and PCs.

There are issues with safety and performance. I need to make sure that my database queries and routes are as fast as possible so that I can manage more than one user at a time. Only admins who have permission should be able to update the security settings. I'll utilize HTTPS connections and authentication to keep the information safe.

The time frame isn't the biggest problem. I need to get my act together and focus on the most important things first because I have to do this by the end of the semester.

System Architecture View

Component Diagram



A text version of the component diagram is available: [CS 465 Full Stack Component Diagram Text Version].

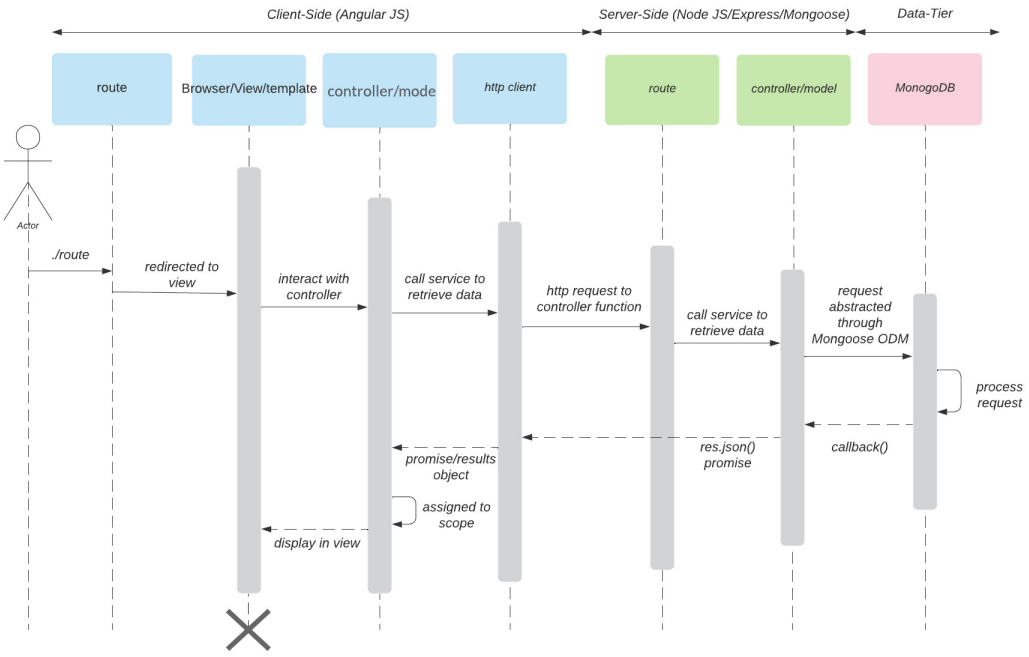
There are three parts to the Travlr Getaways online app:

Angular controls what users see on the front end. For instance, the text, pages, and buttons vary. This part is in charge of both the public travel site and the admin dashboard.

The server layer is powered by Express.js and Node.js. It gets all the requests from the front end and talks to the database.

MongoDB is the database layer that stores all the data regarding users' journeys, destinations, and activities.

Sequence Diagram



Caption

The sequence image shows how the Travelocity Getaways app handles data and requests. The flow in the Browser/View begins when someone goes to the Trips page or chooses a trip. The Route can be called by the browser using Angular services. Then, the Route sends the request to the right Controller.

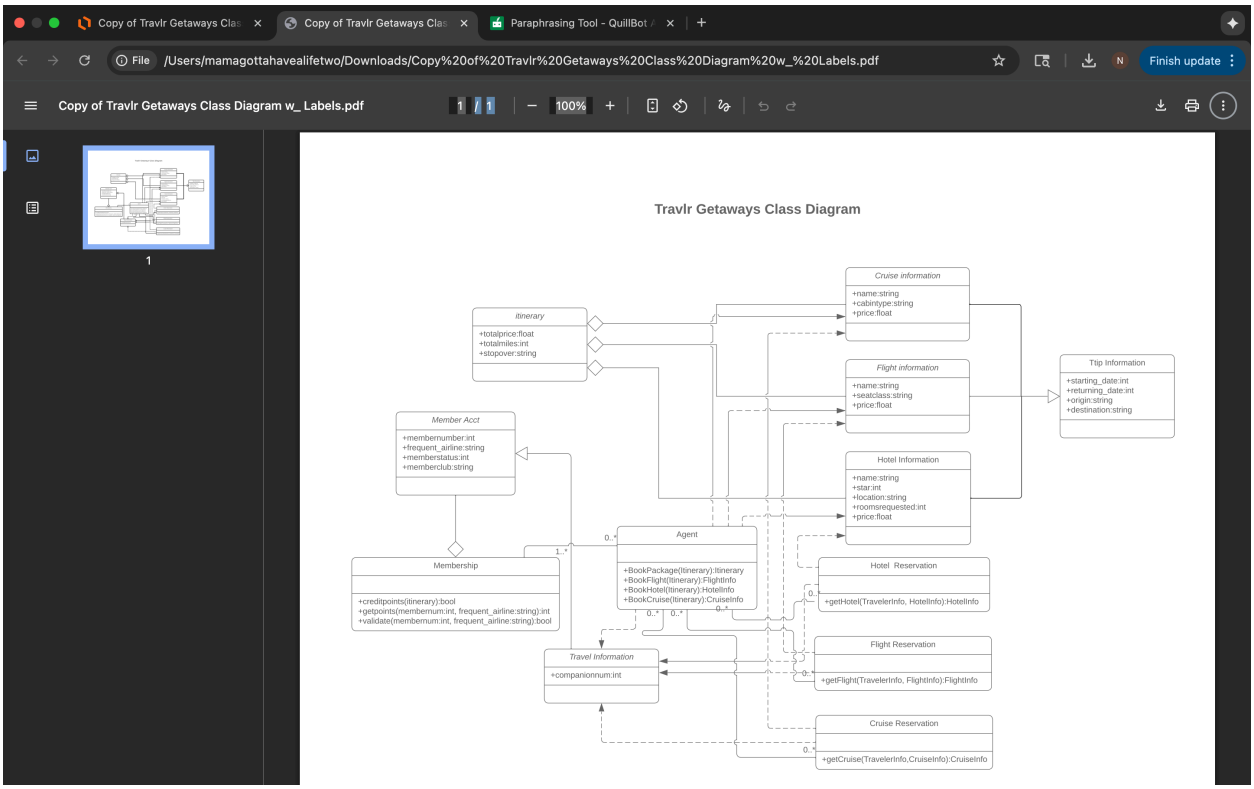
The Controller talks to the Model layer to obtain data from MongoDB. The Controller gets the trip, scheduling, or administrative information it asked for from the database.

After that, the Controller sends an answer back across the Route and into the Browser/View. After then, Angular changes the SPA in real time. You can use these procedures to check in, look at travels, and complete other administrative tasks. They make it clear that the layers of the software are not connected.

Class Diagram

The class diagram shows the data structures used in the app

- TripInfo – keeps track of crucial trip information, such as the destination, price, description, and dates.



- Itinerary – This document provides all the details concerning a trip's daily schedule.
- TravellerInfo – All of the information about clients who make reservations is kept in TravellerInfo.
- FlightInfo, HotelInfo, and CruiseInfo are all instances of travel and housing metadata.
- FlightBooking, HotelBooking, and CruiseBooking connect the information about travelers to the services they have booked.
- MemberAccount: This is where you store your login information, saved journeys, and settings.
- Membership_Admin – This shows the capabilities that are only available to system administrators.
- Travel_Agent – This is for employees who book trips or maintain track of trip details.

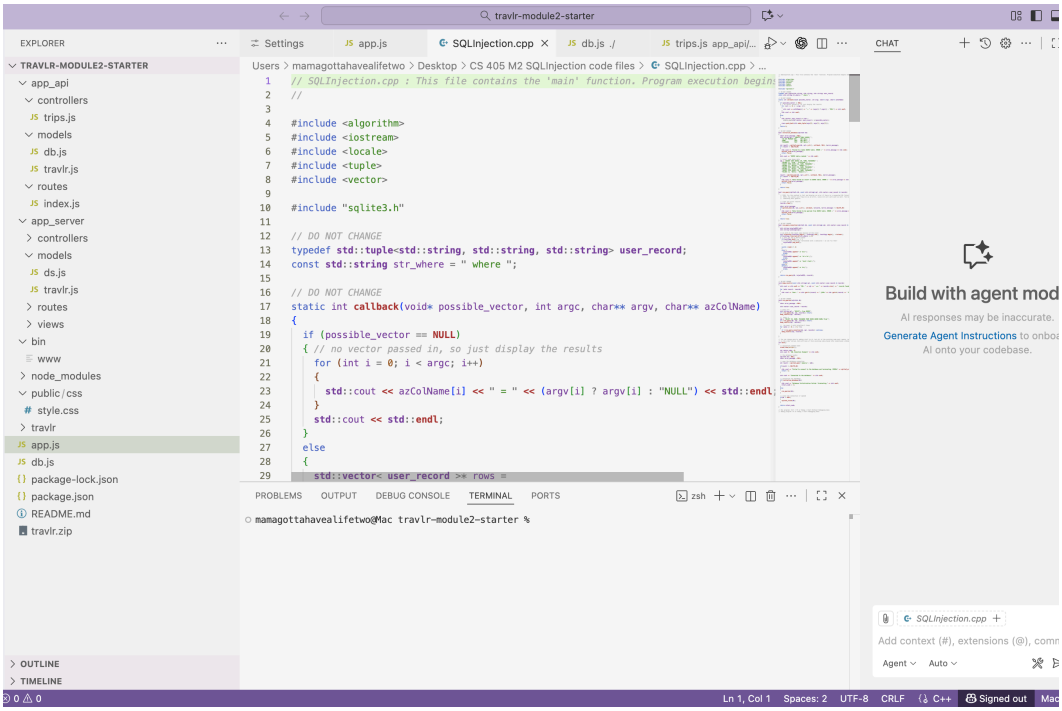
These classes give the Travlr app an organized, modular way to store and work with trip data.

API Endpoints

<Exposing RESTful endpoints is a design approach to enable an application to participate in a larger ecosystem. Document each endpoint in the table below, including the HTTP method, purpose, URL, and notes.>

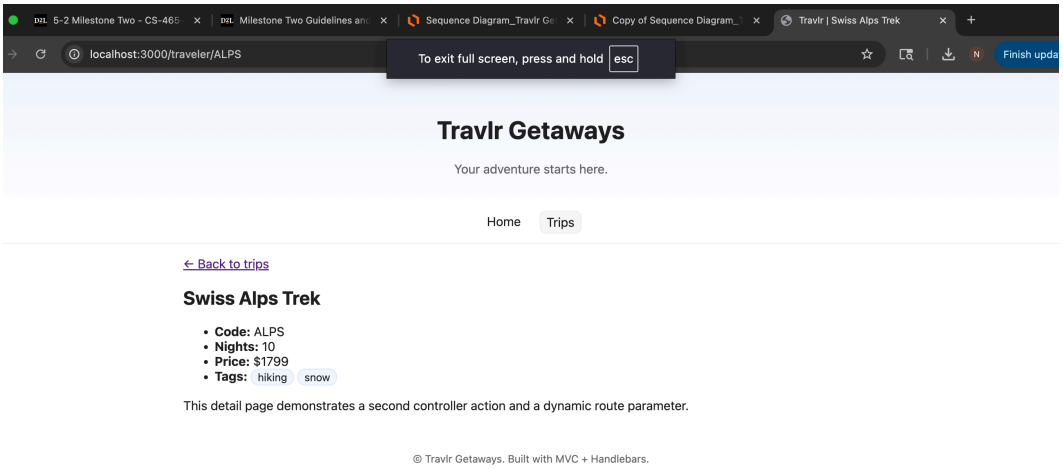
Method	Purpose	URL	Notes
GET	Retrieve all trips	/api/trips	Returns all active trips
GET	Retrieve single trip	/api/trips/:tripId>	Requires valid trip ID
Post	Create new trip	/api/trips	Admin only
PUT	Update existing trip	/api/trips/:tripId>	Validates ID
DELETE	Delete trip	/api/trips/:tripId>	Admin-restricted
GET	Retrieve all bookings	/api/bookings	All Service types
POST	New Booking	/api/bookings	Traveler and trip info
GET	Get flight option	/api/fights	Searching/filtering
GET	Hotel options	/api/hotels	Location filter
GET	Cruise options	/api/cruises	Cruise listings
GET	Member account	/api/members/:id	Req. Authorization
POST	Create new member	/api/members	Endpoint
PUT	Update member acc	/api/members/:id	Updates
DELETE	Delete member	/api/members/:id	Clean up

The User Interface



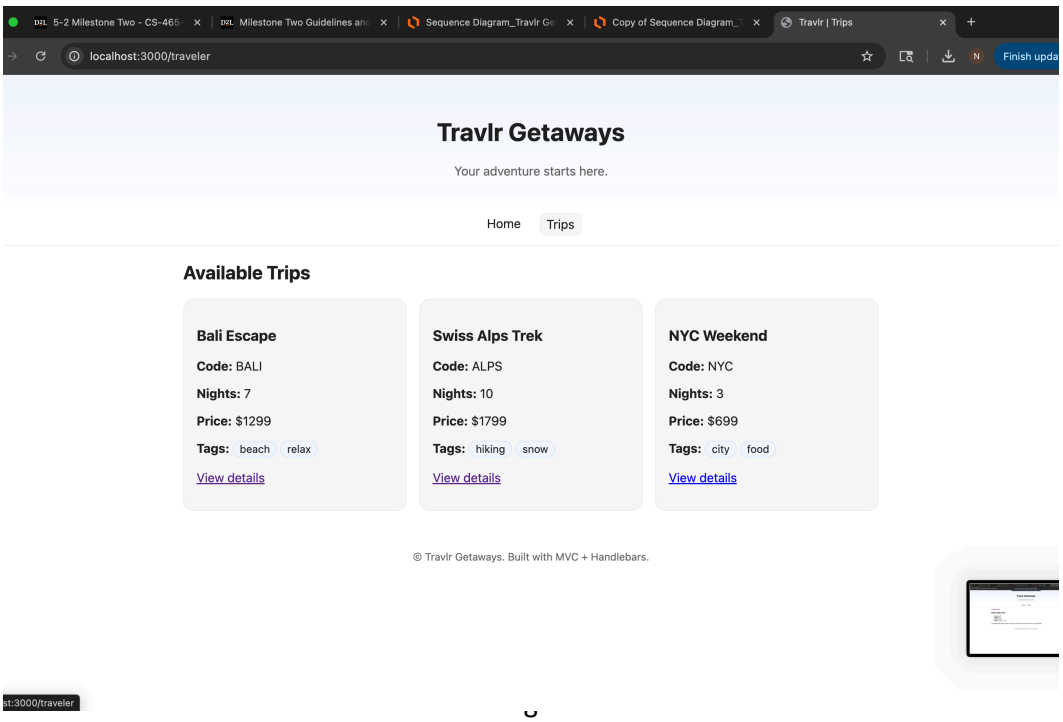
Caption

The Angular SPA is made up of components, modules, and services. These take care of routing, receiving



Caption

data, and talking to users. Angular offers more features than Express, which only provides simple



Caption

rendering on the server side. Angular, for instance, features reactive forms, dynamic site modifications, and service-based state management.

You need to check that Angular works with the API by sending GET requests to get trip data and PUT requests to alter trip details in order to test the SPA. If the communication is successful, MongoDB will always save and return the new information.