

Travlr Getaways Web Application

# **CS 465 Project Software Design Document**

Version 1.0

## Table of Contents

[**CS 465 Project Software Design Document** 1](#_Toc36198462)

[Table of Contents 2](#_Toc36198463)

[Document Revision History 2](#_Toc36198464)

[Instructions 2](#_Toc36198465)

[Executive Summary 3](#_Toc36198466)

[Design Constraints 3](#_Toc36198467)

[System Architecture View 3](#_Toc36198468)

[Component Diagram 3](#_Toc36198469)

[Sequence Diagram 4](#_Toc36198470)

[Class Diagram 4](#_Toc36198471)

[API Endpoints 4](#_Toc36198472)

[The User Interface 4](#_Toc36198473)

## [Document Revision History](#_heading=h.lnxbz9)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 03/16/22 | Luke Raghoo | Wrote executive summary, design constraints, and system architecture view. |

## 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](#_heading=h.35nkun2)

<Describe the appropriate architecture of the web application based on your client’s software requirements. Be sure to reference your use of the MEAN stack for development. Explain both the customer-facing side of the application and the administrator single-page application (SPA).>

This web application utilizes something called the MEAN stack. Each letter in the acronym represents a software that is used: MongoDB, Express, Angular, and Node.js. All the software work together and scale up to support as many connections as possible in a short time frame. Travlr Getaways is trying to create a travel booking site for their customers to book travel packages. They need to be able to create an account, search for travel packages by location and price, and book reservations. Customers are supposed to be able to visit the website on a regular basis before their trip to view itineraries. This would be the customer-facing side of the application. The administrator single-page application is where Travlr Getaways administrators are able to maintain the customer base, available trip packages, and pricing for each item and package.

## [Design Constraints](#_heading=h.1ksv4uv)

<Identify the design constraints for developing the web-based Travlr Getaways application. Explain the implications of the design constraints on the application development.>

One very common design constraint that these types of applications usually have would be the budget. Companies have a certain amount of money that they are willing to spend on a project. The budget is something that will influence the result of the project. Building the website is also done through the Node.js server and the Express framework. Depending on what the client wants to achieve this may or may not be the best way to go about doing it.

## [System Architecture View](#_heading=h.44sinio)

### Component Diagram



A text version of the component diagram is available: [CS 465 Full Stack Component Diagram Text Version](https://learn.snhu.edu/d2l/lor/viewer/view.d2l?ou=6606&loIdentId=24342).

<Describe the overall system architecture of the web application by referring to the component diagram above. Identify the significant components that will be used and their relationships to one another.>

In the diagram above there are three component boxes. The first component box is the client component box in the top left corner. Within that box there are four components: the web browser, client session, traveler portfolio, and graphic library. The traveler portfolio is something that is stored in the database component. The database component (located in the bottom left) is also connected to Mongoose ODM in the server component box. Mongoose ODM is an object data modeling library for MongoDB and Node.js. It manages relationships between data and the representation of objects in MongoDB. The server component box is on the right side which includes three other components besides Mongoose ODM. It contains authentication server, server session, and traveler database.

### Sequence Diagram

<Illustrate the flow of logic in a web application by completing a sequence diagram. Insert an image of the sequence diagram here.>

<Describe the flow of logic in the web application based on the sequence diagram. Be sure to describe the interactions between the layers, or tiers, of the full stack application. It will be helpful to include significant processes such as Sign In, Trips, and Admin interactions when referring to the sequence diagram.>

## Class Diagram

<Illustrate the JavaScript classes of the web application by completing a class diagram for the web application. Insert an image of the class diagram here.>

<Describe the JavaScript classes of the web application based on the class diagram.>

## [API](#_heading=h.2jxsxqh) 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 list of things> | </api/things> | <Returns all active things> |
| **GET** | <Retrieve single thing> | </api/things/:thingId> | <Returns single thing instance, identified by the thing ID passed on the request URL> |

## The User Interface

<Insert screenshots from the development of the SPA development to show the following: (1) a unique trip, added by you, (2) the Edit screen, and (3) the Update screen.>

<Summarize the Angular project structure and how it compares to the Express project structure. Be sure to describe the rich functionality provided by the SPA compared to a simple web application interaction. Describe the process of testing to make sure the SPA is working with the API to GET and PUT data in the database.>