

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 | March 16, 2023 | Eric Wallace |  |

## [Executive Summary](#_heading=h.35nkun2)

To develop the Travlr Getaways web application, the MEAN stack (MongoDB, Express.js, AngularJS, and Node.js) provides a SPA (single page application) to the client’s customers which will allow them to find and book vacation getaways. The application will also provide the client with the ability to manage data for the getaways, meals, rooms, and news seen on the application.

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

Below is a list of some of the constraints/requirements for the Travlr Getaways web application. This is not a complete list as I am sure there are a few that have been missed.

* Web application
* Authentication
* Authorization
* Time
* Budget
* Browser Compatibility
* Responsive
* Compliance
* SSL/TLS
* User Accounts
* Data management

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

### Component Diagram

The component diagram shown above makes use of the MVC design pattern and makes use of a client-side application, server-side application, database server and API. The following are components used in the application and an overview of each:

* Client-side application - Used for rendering information to the user and handles user interactions. It communicates with the API server to request data from the system.
* Server-side application - Used to handle the logic of the application, it receives requests from the API server, queries the database server for information, processes the information received from the database server and then sends it to the API server.
* Database server/component - Stores data related to getaways, meals, room, and news. It handles queries that are received from the server and returns the result to the server.
* API server - Facilitates the communication between the server-side application and the client-side application, so in other words, it is the middleman. It receives requests from the client and sends those requests to the server, and then it sends the data received from the server back to the client.



**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.>