

# **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 | 11/11/2024 | Selva path | Travel Gateway Software specification |

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

This travel gateway software application development will be built MEAN a technology stack used for building web applications, consisting of MongoDB (database), Express.js (backend framework), Angular (frontend framework), and Node.js (runtime environment); "framework includes for the database we are using MongoDB, for API postman, Angular, Angular is an open-source JavaScript framework used to create single-page web applications (SPAs). Scalability

Angular can scale from small projects to enterprise-level applications. Efficiency Agular’s consistent structure and built-in features can help developers create high-quality applications more efficiently. Node.js is a runtime environment that allows developers to use JavaScript to create server-side applications and tools for web development. Server-side programming also Node.js is used to create server-side web applications. It allows developers to use JavaScript for both client-side and server-side code. Dynamic websites and single-page applications Node.js and MongoDB, in conjunction with Handlebars, will be used to generate the static content that Express.js can serve. Handlebars is a templating language that can be used to dynamically populate a static page with content. MongoDB is the NoSQL (non-relational) database that will be used to store the static content to be displayed in the Handlebars templates.

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

A diagram of a stack architecture

Description automatically generated

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

Among many constraints in this Traveler Gateway software application designing and web development are limitations that will impact the creative and technical decisions made early during the design process. They can come from a variety of sources, such as customers, stakeholders, or external regulations. These are some examples of design. Technical constraints might occur. Device and operating system limitations, performance constraints, and integrations and APIs, Commercial constraints: Limited budget, set ETA, and deadlines also When end users access the application, there will be Functional constraints: Image formats, screen resolutions, and logic, Compatibility constraints: The software needs to interface with specific operating systems or databases. Security constraints: Certain encryption protocols, access controls, or data protection measures are required.

”

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

A diagram of a network

Description automatically generated

### Component Diagram



This application development is embedded in three components: client component, server component, and database server component. Each component server has a different client server, including client sessions and web browsers, travel data, and image libraries. The web browser and graphic library both provide an interface. The Web Browser and Graphic Library both provide an API interface. The client session and traveler portfolio require the Web browser interface. The Traveler Portfolio also requires the Graphic Library interface and the interface provided by the Database component. The client session interacts with a port on the client component to connect to the required server component interface. The server component had four components, including the authentication server, server

Session, traveler database, and Mongoose ODM components. The Mongoose ODM and Server

Sessions both provide an interface. The Traveler Database requires the Server Session interface.

and the Server session requires the Mongoose ODM interface.  The Mongoose ODM component

requires the interface provided by the Database component.  Both the Server Session and

Authentication Server components interact with a port on the Server component to provide an

interface. Lastly, the Database component has a single component for MongoDB which provides an

interface to the Client and Server components.

Sequence Diagram

A diagram of a company

Description automatically generated

A sequence diagram is a type of interaction diagram that shows how objects interact over time. It can be used to model the sequence of events that occur in an HTTP request, such as:

The web browser establishes a TCP connection with the web server

The web browser sends an HTTP POST

The web server responds with "HTTP 200 OK"

The web browser displays the return page to the user

Here are some key features of sequence diagrams:

Lifelines: Represent individual participants in the sequence diagram.

Messages: Arrows that show the interactions between objects. Messages can be synchronous or asynchronous, complete, lost, or found.

Vertical axis: Represents time, with the order of messages and calls shown top down.

Horizontal axis: Shows the object instances that the messages are sent to, left to right.

Activation boxes: Opaque rectangles that represent processes being performed in response to a message.

Sequence diagrams are used by software developers and business professionals to: understand requirements for a new system and document an existing process

A sequence of travel web development" diagram, typically represented as a UML sequence diagram, visually outlines the chronological order of interactions between different components (like user, database, airline API) involved in a travel booking process on a website, showing how data flows from the initial user request to the final booking confirmation, including steps like searching for flights, selecting dates, checking availability, payment processing, and sending booking details to the user.

Key elements of a travel web development sequence diagram:

Actors:

The primary participants involved, usually the "User" and the "Travel System" (which encompasses the website backend and external APIs).

Lifelines:

Vertical lines representing each actor, showing their active time during the interaction.

Messages:

Arrows connecting lifelines, signifying interactions like requests, responses, and data transfers between actors

## Class DiagramA diagram of a travel reservation system Description automatically generated

The Cruise Info, Flight Info, and Hotel Info classes all contain a name property and other fields

that are unique to each mode of travel. Each one also inherits the Trip Info class which contains

properties for the start and return data, as well as origin and destination locations.

Cruise Booking, Flight Booking, and Hotel Booking each have an association with them

corresponding Info class and the Traveler class. There are zero-to-many relationships

between the Booking classes and the Travel Agent class in both directions. The Travel Agent

class has also had associations with the Cruise Info, Flight Info, Hotel Info, and Traveler Info

classes and a one-to-many relationship with the Membership Admin class. The Traveler Info

class inherits the Member Account class. The Membership Admin class has an aggregate

relationship with the member Account class. The Itinerary class has an aggregate relationship

with the Cruise Info, Flight Info, and Hotel Info classes.

“

## [API](#_heading=h.2jxsxqh) Endpoints

An API endpoint diagram is a visual representation of how an API responds to requests and responses, and the relationships between different components of an API. API diagrams can help developers, product managers, and other stakeholders understand how an API system functions.

Here are some benefits of API endpoint diagrams:

Method Purpose URL Notes

POST Login a user /api/login Authenticates a user and

returns a JWT

POST Register a user /api/register Add a new user to the

database and returns a JWT

GET Retrieve list of meals /api/meals Returns all meals

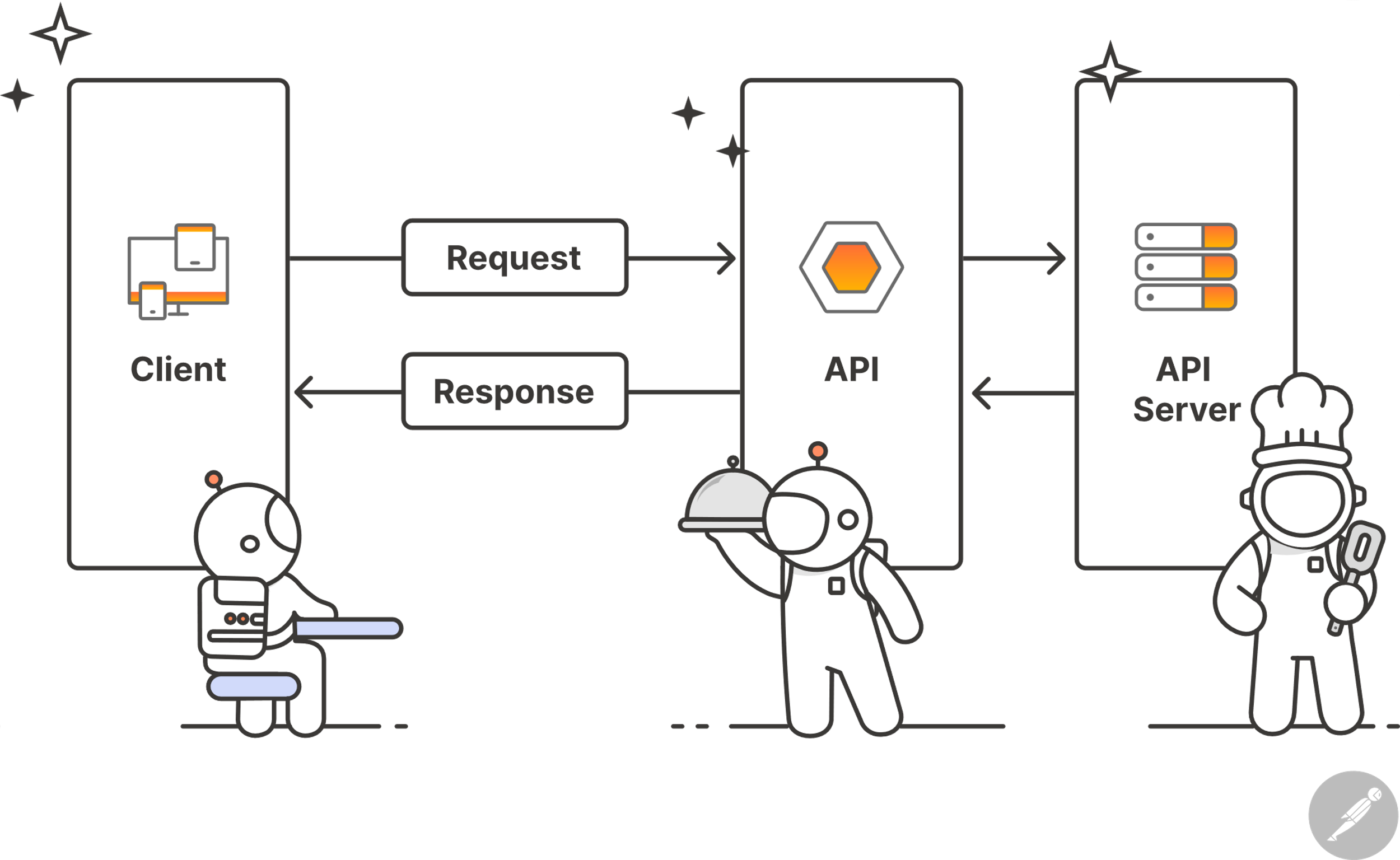
GET Retrieve single meal /api/meals/:mealCode Returns single meal,

identified by the meal code

at end of URL

GET Retrieve list of news /api/news Returns all news content

GET Retrieve sing



| **Method** | **Purpose** | **URL** | **Notes** |
| --- | --- | --- | --- |
| **GET** | Users enter login | Travel web | Verify user |
| **GET** | Create a user account | Register in the travel page | Crate a new user in the database |

## The User Interface

UI Summary

Angular is a frontend framework with the views rendered on the client side, whereas Express is a

backend framework with the views rendered on the server and sent to the client. The Angular

project is made up of models, services, routes, and components. The Express site is made up of

views, controllers, and routes. Angular consists of a single HTML page that dynamically

updates the views and page content on the client. Angular uses reusable components to make up

the parts of the site. Express uses a templating engine, in this case handlebars, to dynamically

generate content on the server before sending to the client. Both use APIs to retrieve or send

data.

Some advantages of SPA functionality include:

• Reduced servers load due to only needing to send the initial page.

• SPAs can create very interactive sites with lots of functionality.

• Faster users experience due to eliminating full page reloads and only necessary data is

retrieved from the server.

Some disadvantages of SPA functionality include:

• A longer initial load time due to needing to retrieve the entire JavaScript application.

• They can be difficult to optimize for SEO.

Additional SPA functionality of simple web applications include:

• Client-side routing which removes the need for additional server requests and provides a

smooth transition between views.

Additional images for page

A room with a bed and a window

Description automatically generatedA coral reef with fish swimming in the water

Description automatically generatedA close-up of a room

Description automatically generatedA room with a bed and a canopy

Description automatically generated

A plate of food on a table

Description automatically generated