Rentaroo: Car Rental Web Application
Project Approach and Technology Stack Selection

Team: Watermelons

Ikram Kamal

Hanine Tydrini

Yasmine Mouatif

Samuel Henderson

Uyi Oronsaye

Benjamin Ho

Outline

This document provides a comprehensive overview of our car rental project, covering its objectives, scope, and target audience. Additionally, it outlines our project approach, including development methodology, project timeline, and communication strategies. Moving on to the technological aspect, the document highlights the chosen technologies for both the backend and frontend, as well as the integration between them. Finally, it addresses third-party services and security considerations before providing a concise conclusion.

1. Project Overview:

Rentaroo aims to provide a user-friendly platform to facilitate the renting process for various types of vehicles. Acting as an interface between customers seeking to rent a vehicle and rental companies offering their services, the platform aims to simplify the process for both parties, making it more efficient and convenient.

1.1. Project Objectives

The main purpose of this project is to give us hands-on experience on the software development life cycle. By using Agile development as our approach, we hope to develop our teamwork and communication skills. In the following weeks, deliveries will continuously be made to release new features.

Our project revolves around the development of a car rental web application. This software application seeks to simplify the process of short-term vehicle rentals, accommodating durations between a few hours to a few weeks. Serving as the middleman, the application will connect clients to renowned car rental companies. Throughout the process, we wish to make the user’s experience as effortless as possible.

1.2. Scope

Since the time allowed for this project is rather short, we hope to deliver an application that could serve as the foundation to build a marketable product. The application will contain three types of users that will be able to use different functionalities:

1. User:
    a. Browser vehicles
    b. Reserve vehicles
        i. Allow users to provide a pickup location, booking dates, desired vehicle type, and price range
        ii. Allow users to add desired extra equipment
    c. View/Modify/Cancel reservations
    d. Find branches of car rental companies
    e. Rate and review their experience

2. Customer service representative:
    a. Check-in process
        i. If the user has a reservation, the representative will be able to confirm the user's reservation after verifying his reservation and identity. Ultimately leading to the rental's agreement and payment.
        ii. If the user does not have a reservation, the representative will be able to make one for him and lead him through the process mentioned previously.

3. System administrator
    a. CRUD operations on vehicles
    b. CRUD operations on user accounts
    c. CRUD operations on reservations

1.3. Target Audience

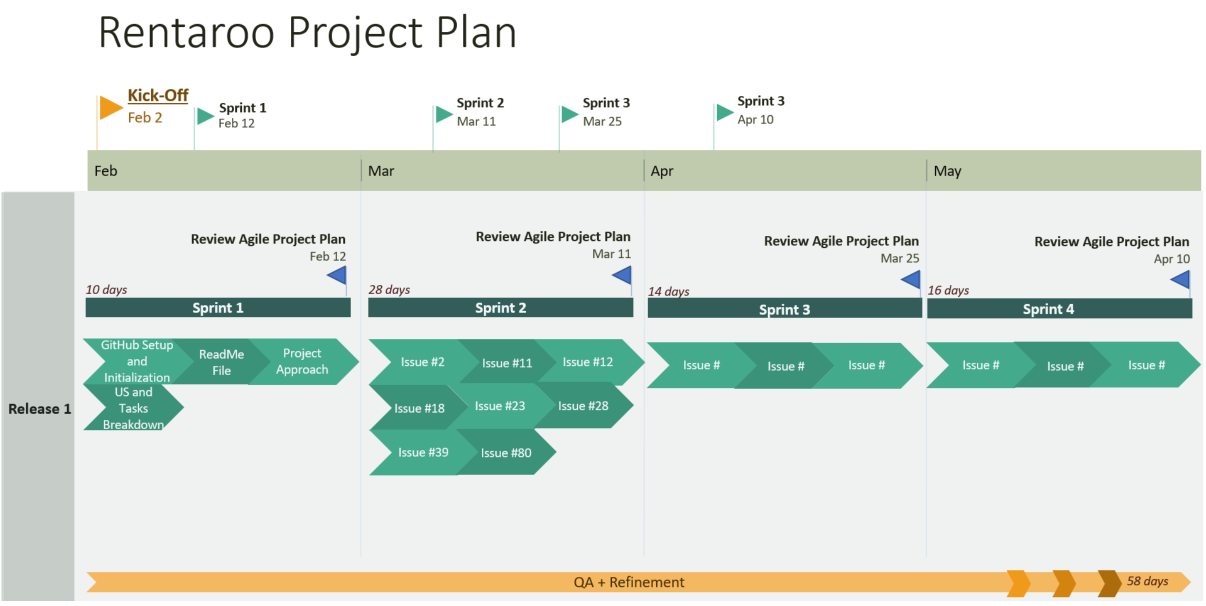
Our web application targets travelers and individuals in urgent need of a vehicle in Canada. By allowing them to freely select a pickup location, their booking dates, and their desired vehicle type, we ensure that their experience is as seamless and convenient as possible.

2. Project Approach

2.1. Development Methodology

For this project, our team will be using the Agile methodology. Agile suits our needs, given its emphasis on iterative development and the requirement of delivering four sprints. The iterative nature of Agile allows us to adapt swiftly to changing requirements and implement necessary modifications. Agile puts significant focus on communication and collaboration. This includes frequent customer feedback and scrums. Since we are a small team, the scrums will serve as a way to update ourselves on the state of the project, as well as identifying and handling issues that arise during the development of the application.

2.2. Project Timeline



2.3. Collaboration and Communication

We will be communicating through a private forum on Moodle provided by our professor. As for our scrums, we will be using Discord. Github will be used to store and track the progress of our project.

3. Technology Stack

3.1. Backend Framework

3.1.1. Django

Django, a high-level Python web framework, stands out for its simplicity, rapid development capabilities, and adherence to the "don't repeat yourself" (DRY) principle. It offers a robust and scalable architecture, making it well-suited for building complex web applications.

Django's choice is justified by its extensive community support, which ensures ongoing updates, support, and a rich ecosystem of third-party packages. The built-in admin interface is a key advantage, streamlining CRUD operations on vehicles, user accounts, and reservations. Additionally, Django's adoption of the Model-View-Controller (MVC) architectural pattern provides a clear and modular structure for the car rental application.

Django's strengths lie in its user-friendliness, extensive documentation, and a large and active community. The "batteries-included" philosophy reduces the need for external libraries. However, some may consider it heavyweight for simpler projects, and its flexibility might be limited compared to microframeworks.

Django is an ideal choice for projects requiring a rapid development cycle and applications with complex database relationships. It is best in scenarios where a robust and modular framework is needed for scalable web applications.

3.1.2. Express.js (Node.js)

Express.js, a minimal and flexible Node.js web application framework, is designed to provide a robust set of features for web and mobile applications. Its focus on simplicity and performance makes it an attractive option for projects with specific requirements.

Express.js is chosen for its asynchronous nature, making it suitable for handling concurrent operations in a car rental system with multiple users. Being lightweight, it allows for quick setup and easy integration with various databases for efficient CRUD operations. The framework's performance is noteworthy, making it suitable for applications requiring high responsiveness.

Express.js is highly suitable for building RESTful APIs and single-page applications, with a large middleware ecosystem for extending functionality. However, it lacks built-in support for complex application structures and has a smaller standard library compared to other frameworks. Its flexibility allows developers to choose components according to project needs. Express.js is known for its minimalistic approach, making it ideal for microservices and rapid development.

Express.js is well-suited for projects requiring a lightweight and fast backend, especially those with real-time features. Its versatility makes it an excellent choice for applications where performance and simplicity are paramount.

3.1.3. Spring Boot

Spring Boot, a Java-based framework, simplifies the development of production-ready applications. It builds on the Spring framework with a focus on convention over configuration, aiming to streamline development processes.

The choice of Spring Boot is driven by its ease of integration with various databases, supporting seamless implementation of CRUD operations. Its comprehensive ecosystem, including Spring Data JPA, simplifies data access and manipulation. The framework's convention over configuration approach reduces boilerplate code, enhancing developer productivity.

Spring Boot excels in supporting the development of enterprise-level applications, with robust security features and extensive documentation. While it has a steeper learning curve for beginners, it benefits from a broad community support and a vast array of third-party libraries. Spring Boot emphasizes modularity and scalability, allowing developers to adopt features as needed.

Spring Boot is ideal for large-scale projects with complex business logic and applications requiring a high level of security. Its convention-based development model makes it suitable for teams familiar with Java, especially in enterprise environments where scalability, maintainability, and security are critical.

3.2. Frontend Frameworks

3.2.1. React

React is a Javascript library for building user interfaces. It used to create interactive and dynamic web applications following component based architecture. The user interface is divided into simple and independent components to compose together a complex UI.

React is chosen for its capacity to create independent and reusable UI components which facilitates the development and maintenance of complex interfaces. Additionally, one of the key features of React is its use of a virtual DOM which contributes to its responsiveness. Instead of directly manipulating the browser's DOM, React creates a lightweight virtual representation of the DOM in memory. When a component's state changes, React calculates the most efficient way to update the virtual DOM and then reconciles it with the actual DOM. By minimizing DOM manipulations and only updating components that have changed, React ensures that UI updates are fast and smooth, leading to a more responsive user experience. Finally, React offers cross platform ability which allows developers to create applications that can run seamlessly across different platforms.

React’s biggest strength is its component based architecture which allows breaking down complex UI making it easier to manage and maintain systems. Furthermore, React’s virtual DOM contributes to high performance and responsiveness, especially in applications with dynamic content. On the negative side, Reacts may be too complex for small projects. It can be overwhelming when developing a small platform compared to simpler alternatives.

React is ideal for building complex web applications with dynamic user interfaces. Its modular architecture and performance make it a popular and effective solution for building modern platforms.

3.2.2. Angular

Angular is a popular front-end framework designed to simplify the development of complex web applications by providing a structured framework for building dynamic and interactive UI.

The choice of Angular was based on its capability to offer a wide range of features and tools for building sophisticated user interfaces. Similar to React, its component-based architecture allows for the creation of reusable UI components, enabling developers to build complex UIs with ease. On the other hand, Angular prioritizes responsiveness by providing features such as lazy loading, Ahead-of-Time (AOT) compilation, and tree-shaking. These features help optimize performance and reduce load times, ensuring that Angular applications deliver a smooth and responsive user experience. Angular also offers cross platform ability ensuring consistent behavior and performance across different web browsers.

One of the best things about Angular is its two-way data binding. This feature helps to exchange data from component to view and from view to component which help the user to establish a bi-directionally communication. Additionally, since Angular breaks down an application into multiple modules, this framework offers huge advantages to the overall performance. Instead of loading all the application code at once in the browser which may cause more loading time and a slower initial render of the web page, it loads only what is necessary. On the downside, Angular can be a bit challenging for beginners. Angular is an opinionated framework which means that it has its own rules that developers need to learn and follow. Angular is often used for building large-scale enterprise applications with complex UI and business logic.

To sum up, Angular is a powerful front-end framework suitable for building modern web applications especially in large industries. While it has a unique philosophy, its extensive feature set and robust tooling make it a compelling choice for a wide range of use cases, especially for large-scale enterprise applications.

3.2.3. JQuery

jQuery is a fast, small, and feature-rich JavaScript library designed to simplify the use of Javascript on a website. It also provides an easy-to-use API for traversing and manipulating the DOM, handling events, making AJAX requests, and animating elements, among other tasks.

In terms of user interface capabilities, jQuery is chosen to provide powerful utilities for enhancing the user experience of web applications. Its simplified syntax and extensive library of plugins make it easy to create dynamic and interactive UI elements, such as dropdown menus and sliders. jQuery simplifies complex DOM manipulation tasks, allowing developers to update and manipulate HTML content with ease. Additionally, jQuery's built-in event handling capabilities enable developers to create responsive and interactive user interfaces that respond to user interactions in real-time.

jQuery is known for its excellent cross-browser compatibility, which is one of its key advantages. It abstracts away many browser-specific inconsistencies and provides a consistent API that works seamlessly across different browsers and versions. This ensures a consistent user experience for all users, regardless of the browser or device they are using. However, jQuery does have its weaknesses. While it simplifies common JavaScript tasks, it can introduce unnecessary overhead for simple projects or modern web applications that may not require its full feature set.

In summary, jQuery is a powerful and versatile JavaScript library that simplifies client-side scripting and enhances user interface capabilities. It is well-suited for rapid development and prototyping of user interfaces because of simplicity and ease of use.

4. Integration and Interoperability

4.1. Backend-Frontend Integration

In the process of developing our car rental web application, the integration of both backend and frontend technologies is crucial for proper functionality, and user experience. Our strategy revolves around the use of a MERN (MongoDB, ExpressJS ,React, Node.js) stack to achieve this goal.

Integration Approach:

RESTful API design: We will need to design a RESTful API that clearly defines our endpoints in order to grant access to backend functionalities such as user authentication for security, vehicle inventory management system, booking services available, as well as payment processing.

Asynchronous form of communication: We can use asynchronous communication through HTTP requests, the frontend can interact with the backend to fetch some data, submit user input data, and perform some operations without interfering with user access to the UI.

JSON Web Tokens (JWT): WE can use this to secure communication between the frontend and backend, utilizing the header, payload and signature to create a JWT-based authenticator. the backend would potentially issue a token that the frontend can include in other API requests.

We agreed on using the MERN stack

4.2. Third-Party Services

After settling our frontend and backend technologies, and integrating them successfully we will also integrate a couple of third party services and APIs to enhance the functionality and the user experience and increase ease of access of our car rental application.

Geolocation Services(Google Maps): For this as mentioned in the class as a recommendation we would utilize the google maps API, to provide services for locating rental stations for pickup, and dropoffs, calculate the distance, and estimate the travel times. This integration could grant a good user experience.

Email Notification system(Twilio SendGrid): for this we can utilize TwilioSendGrid, a free API, that handles email notifications for booking confirmations, cancellation confirmation, appointment reminders, and general updates. This would help enhance communication with users, and improve user experience, more importantly customer experience.

Vehicle information providers(Edmunds): We can also integrate edmuds to provide our users with vehicle data for cars in our catalog. it would include details such as specifications, pricing, and availability. The data enhances the transparency and credibility of our rental inventory, aiding users to make a good choice when selecting a car.

5. Security Considerations

To ensure security measures are properly met, many considerations will be made for the backend and frontend of our application.

As for measurements that involve the front and backend, we will implement a secure and hidden password, and add an MFA system. The secure password will be ensured by recommending a strong password that is at least 8 characters, contains a number, a capitalised letter, and a special character. When the user types it in, it will also be masked behind asterisks with an option to view the actual characters if the user feels as though they are in a safe enough space. As for the MFA system, upon logging in, customers, customer service representatives and system administrators will have to enter a 6-digit code into an input that will have been sent to their email.This will be executed using Twilio Authy, a free API for customer authentication. Implementing this API will allow us to send and verify OTPs (One-time passwords) sent to users.

For considerations that strictly involve the backend, we will limit the number of login attempts for a certain account, use JWT tokens, and also encrypt the passwords at rest within the database. User's will have 5 attempts to login for a specific account. This will help protect against attack cases that would involve trying as many different combinations as possible to login. We will also make use of JSON web tokens which will serve as a user's proof of authentication across the server. Additionally, this will also allow us to set expiration times and have customers be kicked out after 30 minutes. It will lower the risk of having an undesired individual use an account while a device might have been left unattended, or a customer might have forgotten to sign out. Our final backend consideration will be to use the bcryptjs library to encrypt passwords in the database. With this library we can hash passwords used for account creation. When a user attempts to login with their created account, that same password will be hashed and compared to the hashed password associated with that account. If the encrypted password provided by the user matches the encrypted password stored in the database, the user will be granted access to their account page.

To sum up, considerations made for the front and backend will be to implement a secure and hidden password, and add an MFA system with Twilio Authy. In terms of backend considerations, we will restrict the number of login attempts for a specific account, utilise JWT tokens, and also encrypt account passwords within the database using the bcryptjs library.

6. Conclusion

Ultimately, using Agile development, we will produce a functional car rental application. It will be created with proper security considerations and be capable of simplifying the process of short-term vehicle rentals, accommodate durations between a few hours to a few weeks, and serve as the middleman between clients and car rental companies. To develop the application, we will utilize the MERN stack: React for our frontend, Express.js and Node.js for our backend and MongoDB for our database. After extensive research, these frameworks were carefully chosen for their ability to seamlessly integrate with Agile development. By harnessing React's component-based architecture and leveraging Express.js' performance and lightweight backend, we ensure a cohesive and responsive system all while keeping our data safe in a MongoDB database.