Appendix A. Project Approach and Technology Stack Selection Template

1. Project Overview

1.1 Project Objectives

Our project, a cutting-edge Car Rental Platform, aims to revolutionize the way users rent cars for personal use. By leveraging the latest web technologies, we offer a seamless, intuitive, and comprehensive car rental experience. Users can effortlessly browse a wide selection of vehicles, read detailed reviews from previous customers, and choose the perfect car that suits their needs and preferences. Our secure and straightforward payment process ensures a hassle-free rental experience from start to finish.

1.2 Scope

Our scope is building a web application that would provide the service of renting a car. To develop a compelling middle-fidelity prototype following Agile Scrum methodology, utilizing GitHub for version control, bug tracking, task management, and continuous integration. The project duration is 10 weeks, with 4 iterations (sprints). The first 2 weeks of Sprint 1 are for training and setting up the development environment. We can define the scope of our project as follows :

Customers :

- Browse Vehicles for Rent: Access a catalog of vehicles, filtering by type, category, and price range.
- Start a Reservation: Specify rental details like location and dates to view available vehicles and add optional equipment.
- View/Modify/Cancel Reservation: Manage reservations with the flexibility to adjust or cancel as needed.
- Find a Branch: Locate the nearest rental branch using postal code or airport information.
- Rating and Review: Provide feedback on vehicles and the rental experience.

Customer Service Representatives (CSRs)

- Check-in Process: Assist customers with new or existing reservations, verify identification, review rental agreements, and process payments.
- Check-out Process: Inspect vehicles, ensure rental agreement compliance, process final billing, and confirm return completion in the system.

System Administrators

- CRUD Operations on Vehicles: Manage vehicle inventory.
- CRUD Operations on User Accounts: Handle user account management.
- CRUD Operations on Reservations: Oversee reservation records.

1.3 Target Audience

The application is intended to be used by those living in Montreal.

2. Project Approach

2.1 Development Methodology

Agile methodology is the best suitable development methodology for this project. Indeed, we chose Agile for its iterative nature which enables us to add features incrementally. Its incremental nature allows us to easily modify the code to respect evolving requirements. Indeed, our car rental website will be updated frequently with new features with each sprint. Agile is a methodology founded on team collaboration and open dialogue. Indeed, our use of GitHub allows us to cooperate on coding the features in small components then adding it together with our teammate's components. Since we have multiple sprints, Agile's risk management style based on addressing issues quickly is very advantageous. Also, our daily meetings and sprint planning works harmoniously with Agile's philosophy of partnership. Indeed, our small team size of 6 members allows for easy communication and collaboration.

2.2 Project Timeline

Date	**Description**
February 12th	Sprint 1 due date
March 11th	Sprint 2 due date
March 25th	Sprint 3 due date
April 10th	Sprint 4 due date

- Meetings every Monday and Friday

2.3 Collaboration and Communication

We mainly use discord as our communication channel. It provides real-time messaging and serves as the main platform for our team collaboration, discussions, and meetings. GitHub provides features like branching, merging, and pull requests for seamless collaboration and code review. It helps tracking code changes, enabling concurrent work without conflicts. GitHub Wikis serve as a centralized repository for additional team rules, including git rules for managing labels and repository. It promotes knowledge sharing and helps onboard new team members efficiently. Wikis can be edited collaboratively by team members, keeping documentation up to date. Also, the wikis hold information planning for upcoming sprints, information of each task with their objectives, timelines and dependencies.

3. Technology Stack

3.1 Backend Frameworks

- Django: Chosen for its comprehensive features and rapid development capabilities, ideal for our secure and scalable web application
- Node.js: Offers great performance for I/O intensive operations, suitable for real-time features.
- ASP.NET: Known for its robustness in enterprise scenarios, it provides strong security and performance.

3.1.1 Django

- Description: Django is a free open-source Python-based web framework that runs on a web server. It was developed to make the web development process fast and easy. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel.
- Rationale: Django was chosen for its robust set of features that enable building secure, scalable, and maintainable web applications quickly. Its built-in admin panel, ORM (Object-Relational Mapping), and class-based views are particularly beneficial for our project's backend development needs.
- Strengths : Django includes a comprehensive standard library, an automatic admin interface, and a tight security model to help developers avoid common security mistakes such as SQL injection, cross-site request forgery, and clickjacking. Its scalability and flexibility make it suitable for projects of any size.
- Weaknesses : Django's monolithic structure can be cumbersome for smaller projects or when a more microservices-oriented architecture is preferred. Its ORM may not be the most efficient for handling complex queries and massive datasets.
- Use Cases :
  - Rapid Development: Django's "batteries-included" approach for rapid development of features without the need for extensive setup.
  - Data-Driven Applications: With its powerful ORM, Django is well-suited for applications that require complex data processing and storage, such as car rental platforms.

3.1.2 Node.js

- Description: Node.js is an open-source, cross-platform JavaScript runtime environment that allows the execution of JavaScript code server-side, allowing for the development of backend applications using the same language familiar to many developers from frontend development. It is built on Google Chrome's V8 JavaScript Engine.Brief overview of Framework B.
- Rationale: Node.js is known for building efficient, scalable applications because of its non-broken, event-driven architecture making it highly scalable and suitable for real-life applications. Due to its simple integration with web technologies, this JavaScript-based framework was chosen and will be a great fit for our project. Additionally, It is also worth mentioning that all the developers in the project are well-familiarized with this language. Qualitative Assessment:
- Strengths : Node.js possesses multiple strengths that make it a preferred choice for the development of web applications. First of all, Node.js can handle multiple operations in the background without

blocking the main thread due to its asynchronous architecture. This model is a perfect match for web applications that require a lot of I/O operations, thus increasing the throughput of the system. Unlike other traditional processing models, I/O operations don't halt the execution of subsequent operations. While using Node.js, developers can maintain unity within their frontend and backend scripting structures of a single language, JavaScript. Such a unified environment reduces the workflow and learning curve of both front and back teams. Hence, The development process gains greater efficiency and organization.

- Weaknesses : Overall, Node.js is a powerful tool for web development, but it has some weaknesses that make it less suitable for certain projects. As explicitly said, Node.js operates on a single-threaded loop. This is great for I/O tasks, but not for CPU-intensive operations. The single-threaded nature can become a bottleneck when it has to deal with these operations. As the event loop is frequently busy operating I/O requests, other incoming requests get delayed, which reduces the overall performance of the system. Another weakness is the extensive nesting of callback functions, necessary for any asynchronous architecture.
- Use Cases :
    - Collaborative Tools: Node.js provides extensive tools for collaborative applications such as document editing platforms which let multiple users collaborate and edit the same document in real-time. Node.js handles WebSockets which facilitates real-time event base communication.
    - IoT Applications: The Node.js non-blocking I/O model is a perfect match for IoT Applications, those that handle concurrent connections from various devices at the same time. The ability to process a large amount of data aligns well with the lightweight data-intensive requirements of IoT models.

3.1.3 ASP.NET

- Description: ASP.NET is an open-source web framework used for building the most modern dynamic web applications and services with .Net by Microsoft. It accommodates different programming styles like Web Forms, MVC, and Web API. ASP.NET enhances the .NET platform with components for building specific types of apps. It offers a new comprehensive framework that processes web requests in languages like C# and F#.
- Rationale: ASP.NET offers developers a seamless integration with other Microsoft technologies, like Visual Studio, SQL Server, and Azure. ASP.NET is known for providing a rich set of features and libraries for building web applications and also includes support for model-view-controller architecture. Caching, web APIs, authentication, and authorization are examples of more technologies that ASP.NET offers.
- Strengths : ASP.NET is known for its high performance nature. It uses compiled code which is faster than interpreted code used in many other web applications frameworks because of features like caching, JIT compilation, and native optimization.
- Weaknesses : The main weaknesses of using ASP.NET come from its environment. While being great for Microsoft technologies, using ASP.NET with a non-Windows environment can be relatively

challenging. Furthermore, ASP.NET could present a difficulty in the learning curve for those who are not familiar with .Net languages such as C#, F#, and VB.NET.

- Use Cases :
  - Web Applications: ASP.NET is widely used for building dynamic web applications ranging from small websites to large-scale enterprise applications. These applications can include e-commerce platforms, content management systems (CMS), customer relationship management (CRM) systems, and more.
  - E-commerce Platforms: ASP.NET is suitable for building robust e-commerce platforms that handle various functionalities such as product catalog management, shopping cart management, order processing, payment integration, and user account management.

3.2 Frontend Frameworks

- Vue.js: Selected for its ease of learning and integration, Vue.js enables us to develop a dynamic UI efficiently.
- React: Offers a vast ecosystem and high performance, ideal for complex UIs.
- Angular: A full-fledged framework providing a wide range of features, suitable for enterprise-level applications.

3.2.1 Vue.js

- Description: Vue.js is a JavaScript framework for building interactive web interfaces. It focuses on the view layer and provides a simple API. It's very approachable since its syntax is a blend of HTML and JavaScript. Vue.js is known for its vast ecosystems of libraries and tools making it a popular framework for both small and big projects alike.
- Rationale: Vue.js was chosen due to its simple nature. When we considered our front-end developers' technological background, we found that Vue.js 's ease of integration and incremental approach were the right fit for our team. Indeed, we are more familiar with HTML, CSS, and JavaScript thus Vue.js similar syntax is a great asset. Also, its component-based architecture and incremental philosophy enable us to modularize the code which helps us scale harmoniously.
- Strengths : As previously mentioned, Vue.js has a simple syntax and HTML-based templates, making it friendly to developers already comfortable with HTML, CSS, and JavaScript. Vue.js, through its virtual DOM, efficiently updates the DOM when a data property is changed in a Vue component. Also, Vue.js provided reactive features such as computed properties and watchers which allow faster responses to changing data. Vue.js encourages component-based architecture. Thus, the encapsulated components promote easy communication between parent-child, code reuse, modularity, and maintainability.
- Weaknesses : Vue.js is a relatively newer framework, thus it faces challenges in enterprise adoption compared to its older peers such as React and Angular. Indeed, due to the lack of enterprise adoption, Vue.js doesn't have the same level of studies and documentation. Also, its ecosystem of libraries and tools may not be as vast as React or Angular. Some developers may not find a library or a set of tools specific to their unique needs. Furthermore, Vue.js official tooling named Vue CLI is very limited.

Indeed, it doesn't offer as many features as React's Create React App or Angular's Angular CLI. For example, Vue CLI does not offer built-in support for server-side rendering while React has Next.js and Angular has a built-in option.

- Use Cases :
  - In-app Widgets: Vue.js is lightweight and flexible enough to be embedded into existing applications. It can easily be integrated into larger applications due to its modular and incremental nature.
  - Interactive User Interfaces: Vue.js 's fast response rate due to its virtual DOM and responsive features suits applications with real-time data updates like dashboards.

3.2.2 React

- Description: React is a front-end framework. It is an open-source JavaScript library developed and created by Meta Platforms formerly known as Facebook. It's a component-based framework meaning developers can create reusable self-contained code that defines a specific UI component. React's component-based architecture, virtual DOM, and JSX syntax streamline UI development, offering fast rendering and easy state management. Its large ecosystem provides extensive libraries and community support, making it ideal for building scalable and maintainable web applications.
- Strengths : As previously mentioned, React has a component architecture that makes it easier to manage large and complex applications due to its encapsulated nature. React has a virtual DOM which is a lighter version of the actual DOM in the memory. Thus, React has faster and more efficient DOM updates and Browser re-renders since it updates the virtual DOM and then optimizes when updating the real DOM. React is the most in-demand web framework as such it has a large and ever-growing community and provides an abundance of resources such as tutorials, forums, and online communities.
- Weaknesses : React is based on a flux architecture which can prove hard to learn for developers. New concepts such as Actions, Dispatchers, Views, and Stores are hard to understand for beginners. Indeed, the Flux architecture can introduce scalability challenges due to a growing set of dependencies between the aforementioned concepts. Also, React's flux architecture forces developers to write many repetitive codes known as boilerplate code to set up the Flux pattern.
- Use Cases :
  - React Native allows developers to build mobile applications using JavaScript and React, thus enabling code sharing between the web and mobile platforms.
  - Large-Scale Web Applications: React's modular architecture and vast tools like Redux and React Router are great assets for building large-scale web applications with multiple UI requirements.

3.2.3 Angular

- Description: Angular is an open-source framework used to develop web applications created by Google. It is a TypeScript based framework, which is an extended version of JavaScript. Its main use is to build single-page web applications.
- Rationale: Angular provides a wide-ranging toolkit and may be too complex for our car rental web application. Furthermore, considering the front-end developers of our team, Angular is not the best option due to its unfamiliarity.
- Strengths : Angular provides a wide variety of tools, libraries, and third-party integrations, which can facilitate and accelerate the development of the web application. In other words, it has more reusable components, which requires less coding. It has a reduced need for manipulating the DOM manually, due to its data binding feature, which simplifies the synchronization between the data model and the UI. Additionally, Angular is still being updated regularly by its large community of developers, meaning that there is a vast amount of resources to provide support to the user.
- Weaknesses : If not optimized properly, Angular's features may lead to performance overhead, which could affect the loading time and the responsiveness of the website, especially for smaller applications. It is also very complex and not ideal for simple and small web applications, since it has a steeper learning curve.
- Use Cases :
    - Single-page applications: Angular provides the necessities for building dynamic and interactive UX without page reloads.
    - Real-time applications: Due to Angular's data binding and reactive programming features, it can be used to build applications such as chat applications, dashboards, etc.

4. Integration and Interoperability

Our car rental platform leverages the robust backend capabilities of Django with the dynamic and responsive frontend framework Vue.js, creating a seamless and efficient user experience. Django serves as the powerful backend, utilizing its ORM for database abstraction and its migration system for secure database changes, while Vue.js enhances the frontend with its component-based architecture and reactive data binding. This integration is facilitated through RESTful API endpoints created with Django, which Vue.js accesses using Axios for smooth data exchange. Together, Django and Vue.js enable a modular, scalable, and maintainable architecture, ensuring our platform is both user-friendly and technically sound.

5. Security Considerations

For security considerations in our car rental platform, we prioritize safeguarding user data and interactions. By integrating Auth0, we implement robust authentication and authorization processes, utilizing JWT (JSON Web Tokens) for secure communication between our Django backend and Vue.js frontend. This setup ensures that sensitive operations and data access are strictly reserved for authenticated users. Furthermore, we enforce HTTPS protocols for encrypted data transmission and adhere to CORS policies in Django to manage cross-origin requests safely. These measures, alongside diligent input validation and regular security updates, form a comprehensive security strategy that upholds the integrity and confidentiality of user information and interactions across our platform.

6. Conclusion

The project team has opted to adopt the Agile methodology, given the manageable size of the team. In line with this decision, Vue.js and Django have been selected for front-end and back-end development, respectively. This choice stems from the team members' familiarity with each framework and their comprehensive feature sets, which align well with the desired product objectives.