Team: Orion

Names: Safaa Kentaoui (40228872), Jadon Games (40173613), Khaled Rezgui (40176606), Matia Komsic (40175388)

# **Appendix A. Project Approach and Technology Stack Selection Template**

1. Project Overview (Safaa)

1.1 Project Objectives

The main objective of the project is to create a car rental application that makes it easy for customers to rent vehicles by allowing them to browse and select car options that fit their needs. It will also allow customer service representatives to provide assistance for the customers and system administrators to manage the application’s data.

1.2 Scope

Our car rental website’s main features allow the customers to browse the available vehicles for rent. Customers will be able to filter the vehicles according to their needs; location, car model, car size, price range, and time period during which they wish to rent the car. They will be able to start, view, modify or cancel their reservations. Our website will also allow the customers to give public feedback on the rented vehicles to help other potential renters find what’s best for them. This application also includes functionalities for the customer service representatives, who will be able to make and confirm reservations for customers, as well as reviewing the rental agreement according to its terms and conditions. Lastly, the system administrators will be able to perform CRUD operations on vehicles, user accounts and reservations to ensure data coherence.

Deliverables for the project will include a fully functional car rental application that includes the aforementioned features and functionalities. The implementation of the application must be completed in around 10 weeks. Reports with information concerning the progress of the project accompanying each iteration of the project. The final website should be a compelling middle-fidelity prototype.

1.3 Target Audience

Our car rental application is intended for travelers who need a simple and interactive way of finding the cars that correspond to their needs. These customers will need a simple user interface and the ability to compare car models. Our web application will also cater to customer service representatives who need to assist the renters. Lastly, it will also be adapted for the system administrators who need to manage data.

2. Project Approach

2.1 Development Methodology

The development methodology that our group has come to an agreement on using is the Agile development methodology. Our team chose this methodology to compensate for the tight schedule that we have been given and to meet the requirements for the four Sprint deadlines. The agile methodology is also much faster in terms of creation of features and fixing bugs compared to the waterfall development methodology.

2.2 Project Timeline

The project timeline will follow the 4 sprint delivery deadlines that are given out during this semester. The timeline will be as follows:

Sprint 1(Feb 12)

* Set up a GitHub Repository
* Decide on project approach and technology
* Create 6 user stories backlog
* Breakdown the user stories in to tasks and assign those task to the team members

Sprint 2(Mar 11)

* Allocate story point to each user stories
* Implement 50% of the task given from the user stories
* Test the functionality of the features implemented
* Generate more user stories for optional features
* Breakdown those extra stories into tasks

Sprint 3(Mar 25)

* Implement all of the tasks given at the first sprint
* Test the functionality of the features implemented
* Implement 50% of the optional features task
* Test the optional features implemented

Sprint 4(Apr 10)

* Implement all of the optional features
* Test the functionality of the optional features implemented
* Write the final report and presentation

2.3 Collaboration and Communication

To be able to collaborate with each team member in the most efficient way possible we will all be using GitHub Repositories to share and upload code. Different branches will be assigned to each team member and assigned tasks as well. The branches will allow each member to have their own space then upload to the main branch and GitHub helps us organize ourselves to each task.

Our method of communication will be through the use of the Discord app. Through the app we will be able to meet and communicate frequently since it is online and we won’t have to always meet in person.

3. Technology Stack (Khaled)

3.1 Backend Frameworks:

3.1.1 Framework **Express (JavaScript)**:

* Description: Node.js-based backend framework for web application development.

Rationale:

* Community support, scalability, and integration ease are advantageous factors. The framework is very popular, so there are a lot of resources for help.

-

Qualitative Assessment:

* Strengths:

Widely adopted, large developer community.

Fast and lightweight for handling concurrent requests.

* Weaknesses:

Limited features compared to more opinionated frameworks.

Steeper learning curve for beginners.

* Use Cases:

Suitable for small to medium-sized projects requiring speed and simplicity.

3.1.2 Framework **Django (Python)**:

* Description: Python-based backend framework known for rapid development.

Rationale:

* Performance, security, and maintenance. Python is also a very popular and relatively easy to learn programming language.

Qualitative Assessment:

* Strengths:

High-level abstraction, enabling rapid development.

Built-in security features.

* Weaknesses:

May be less performant for certain complex operations.

Can be restrictive for projects requiring extensive customization.

* Use Cases:

Ideal for projects with strict security requirements and rapid development needs.

3.1.3 Framework **Spring Boot (Java)**:

* Description: Java-based backend framework with a focus on convention over configuration.

Rationale:

* Community support, learning curve, and extensibility. Everyone in the team is already familiar with java, since it was the programming language choice for most of us.

Qualitative Assessment:

* Strengths:

Robust ecosystem and strong community support.

Simplifies configuration and development with convention over configuration.

* Weaknesses:

Initialization time can be slower.

Requires a deeper understanding of Java.

* Use Cases:

Well-suited for large-scale enterprise applications with a focus on maintainability.

3.2 Frontend Frameworks:

3.2.1 Framework **React**:

Description: JavaScript-based frontend framework that uses interactive components and a library for building interfaces.

Rationale:

* User interface capabilities, responsiveness, and cross-browser compatibility. Also, it allows interactivity for the users.

Qualitative Assessment:

* Strengths:

Virtual DOM for efficient updates.

Large and active community with extensive third-party libraries.

* Weaknesses:

Steeper learning curve for beginners.

Requires integration with additional tools for a complete development stack.

* Use Cases:

Best for building single-page applications with complex user interfaces.

3.2.2 Framework **Angular (JavaScript):**

Description: JavaScript-based frontend framework known for modularity and built-in tools.

Rationale:

* Modularity, performance, and community support. Some members of the team already have experience using angular as a front end framework.

Qualitative Assessment:

* Strengths:

Comprehensive framework with built-in tools.

Strong support for large-scale applications.

* Weaknesses:

Complexity may be excessive for smaller projects.

Steeper learning curve due to a range of concepts.

* Use Cases:

Suitable for large, enterprise-level applications with complex requirements.

3.2.3 Framework **Vue (JavaScript)**:

Description: JavaScript-based frontend framework known for its simplicity and incremental adoption.

Rationale:

* Ease of integration, component libraries, and developer experience.

Qualitative Assessment:

* Strengths:

Lightweight and easy to pick up.

Progressive adoption, can be integrated incrementally.

* Weaknesses:

Smaller community compared to React and Angular.

Limited resources and plugins compared to more established frameworks.

* Use Cases:

Ideal for small to medium-sized projects, or projects requiring incremental adoption.

4. Integration and Interoperability

4.1 Backend-Frontend Integration

We will most likely utilize APIs for data communication between the backend and frontend of the software. There is a possibility the usage of APIs won’t even be necessary for simplicity, so it might be avoided.

4.2 Third-Party Services

Payment: depending on whether the scope of the project will involve payment abilities for the car rental, there might be implementation of third-party payment tools. There are such services as Stripe that offer this capability.

User authentication can be done using third-party services.

User content can be stored on cloud storage services possibly. There are services such as AWS S3 for such purposes (Amazon S3).

MongoDB or mySQL database integration

5. Security Considerations

Implementation of a JWT token authentication system for validating inputs from users. Furthermore, depending on the type of user that is using the website, there must be role-access control measures implemented to protect certain features.

Use HTTPS for secure communication, with encryption protocols for internet communication. This is to ensure the protection of the data transferred.

Pro’s:

* Industry standard in terms of security, and most advanced security protocol, ensuring full confidentiality.

Con’s:

* Might slow down the website due to complex encryption algorithms operating.
* Requires keys and certificates to be up to date, which can cause errors.
* An SSL certificate costs money, which is required for HTTPS.

Use password protection hashing for ensuring.

Sanitation of user inputs to avoid malicious attacks on the website, such as XSS attacks.

6. Conclusion

In conclusion, we are going to use Agile development methodology to create this project, due to its flexibility as well as its ability to develop projects fast. Our project will include frontend and backend parts, with different team members focusing on either of those, and some focusing on both. Integrating frontend and backend together can be done using APIs, and the intervention of third party software can be used for ensuring full data privacy. For now, we tend to use ExpressJs for the backend, with NodeJs, and ReactJs for the front end. We also are investigating the use of a database system like MongoDb, highly inspired by the “MERN” stack, which already has a lot of resources that could help.