Chapter

2 Sprint 0 : Planning and Architecture

Introduction

In this chapter, we will focus on the planning and architecture phase of our project. We will start by capturing and analyzing requirements, which includes identifying actors, deriving functional and non-functional requirements, and creating a use case diagram to model the system functionalities. Project management will be executed using the Scrum methodology, including sprint planning. Additionally, we will evaluate the work environment, including hardware and software, and discuss the hardware and software architecture that will support our project.

2.1 *Capture and analysis of requirements*

2.1.1 Identifying the actors

Each computer system provides its actors with relevant information and features that enable them to interact with it. These actors can be classified, as indicated in Table 2.

Actors	
Super Administrator	Super Administrator holds the highest rank in the system's hierarchy and is responsible for setting up a company to start using the SEO Audit platform. He has the overall authority and control over the system and its functionalities.
SEO manager	The role of the SEO Manager in the SEO Audit Platform is to oversee and coordinate the work of SEO specialists to ensure that their efforts align with the goals of the business.
SEO specialist	This actor is responsible for analyzing website statistics, identifying technical issues, and making suggestions to improve the website's search engine visibility and rankings.

TABLE 2. *Identification of actors*

2.1.2 Identification of functional requirements

The functional requirements specification phase is essential to ensure that our platform's results align with the expectations of the Product Owner.

Thus, the different functionalities that we plan to implement in the framework of this project can be grouped into the following points :

- The platform should allow the SEO manager to create and manage user accounts for SEO specialists, with different roles and permissions depending on their level of access and responsibility.
- The platform should allow the SEO manager to manage user profiles, including updating user information.
- The platform should support collaboration and communication between SEO specialists, including the ability to share data, insights and assigning tasks.

- The platform should support notifications for upcoming tasks and allow users to manage their notification settings, including turning notifications on or off and reviewing previous notifications
- The platform should support integration with data sources such as Google Search Console (GSC) and Google Analytics (GA) to provide statistics.
- The platform should provide a tool to check website health and identify issues such as broken links, duplicate content, missing meta tags, slow page load times etc..
- The platform should allow SEO specialists to add various types of suggestions, such as keyword suggestions, content optimization suggestions, technical SEO suggestions, and backlink suggestions.
- The platform should allow super administrators to register new companies on the platform.
- The platform should allow super administrators to manage all accounts on the platform, including creating, modifying, and deleting accounts for SEO managers and SEO specialists.

2.1.3 Identification of non-functional requirements

Non-functional requirements play a crucial role in enhancing the software quality of a system. These requirements serve as constraints that must be carefully considered to ensure the implementation of an effective solution that meets the expected outcomes, while also preventing potential inconsistencies in the system. With that in mind, our application must comply with the following set of non-functional requirements :

Security : The platform should be designed with security in mind, including measures to prevent unauthorized access to sensitive data, protect against malicious attacks, and ensure the confidentiality and integrity of data.

Performance : To accommodate the large amount of data to be stored, the application must perform optimally in terms of response time, data processing, and loading of data.

Maintainability : The code should be well-structured, commented, and easily readable, enabling it to adapt to changes in requirements and evolve over time.

2.1.4 Modelling of requirements

The diagram shown in Figure 3 describes the different functional requirements of our application, each use case representing a functionality offered by the system to its users.

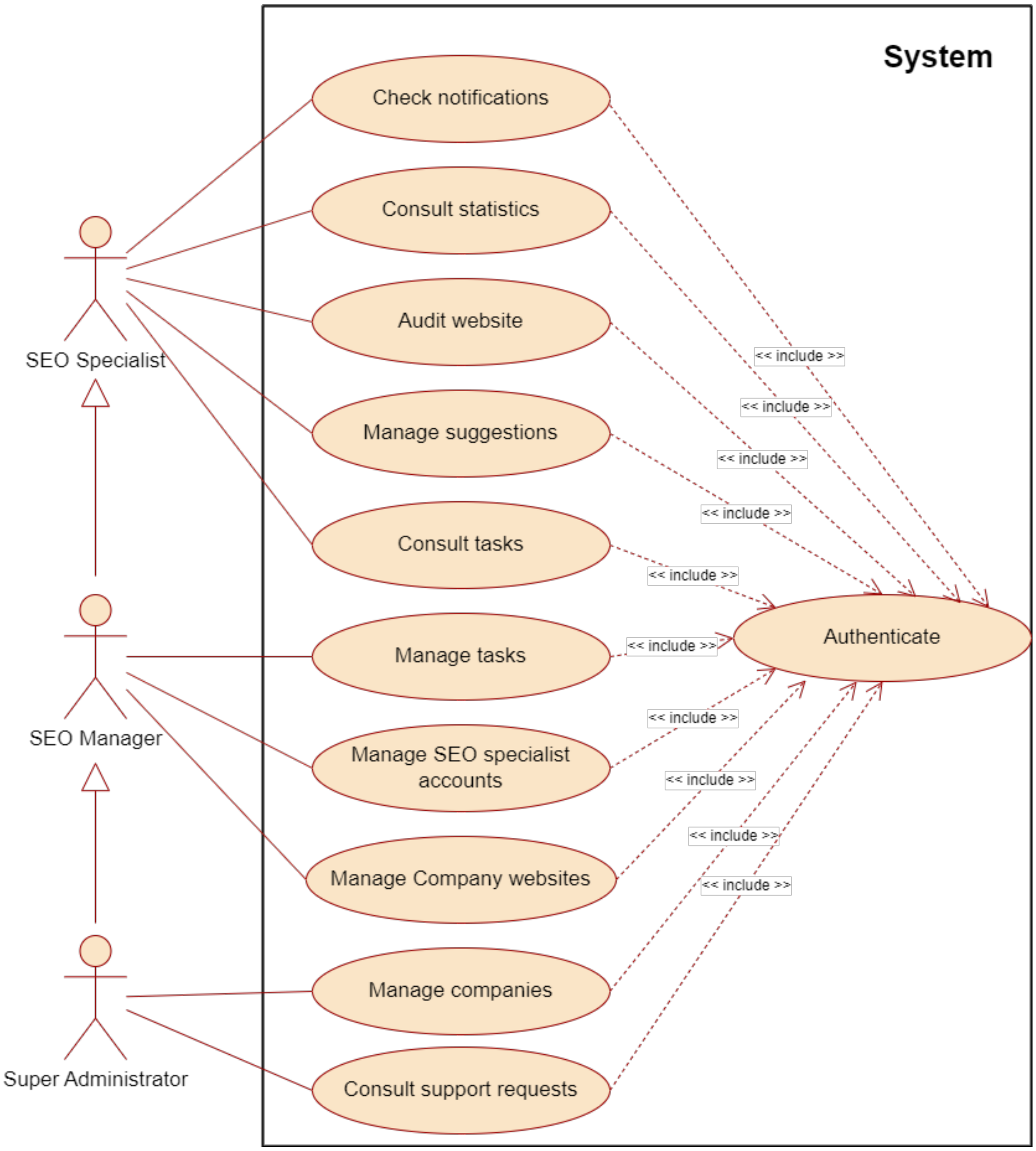

FIGURE 3. *Global use case diagram*

2.1.5 Product Backlog

The product backlog is the most important artifact in Scrum. It consists of a list of features or ”User Stories” that constitute the desired product. Each feature is characterized by an ID number, a theme, a priority, and an estimate of the effort required for a team to implement it. The estimation is represented using relative points based on the Fibonacci sequence to avoid confusion due to estimates that are too close together. The priority is represented using the ”MoSCoW” method, which stands for :

- **M :** ”Must have” - Non-negotiable product needs that are mandatory for the team.
- **S :** ”Should have” - Important initiatives that are not vital, but add significant value.
- **C :** ”Could have” - Nice to have initiatives that will have a small impact if left out.
- **W :** ”Will not have” - initiatives that are not a priority for this specific time frame.

Table 3 summarises the product backlog of our application.

ID	Theme	ID	User story	Priority	Estimation
1	Authenticate	1.1	As a User, I would like to authenticate myself to access the platform.	M	8
2	Consult statistics	2.1	As an SEO specialist, I would like to consult the statistics.	S	3
2	Manage SEO specialist accounts	3.1	As an SEO manager, I would like to consult the list of SEO specialists.	M	2
		3.2	As an SEO manager, I would like to consult the profile of an SEO specialist.	S	2
		3.3	As an SEO manager, I would like to add an SEO specialist profile.	M	5
		3.4	As an SEO manager, I would like to modify the profile of an SEO specialist.	M	5

		3.5	As an SEO manager, I would like to delete an SEO specialist profile.	M	3
		3.6	As an SEO manager, I would like to filter the list of SEO specialists.	W	1
		3.7	As an SEO manager, I would like to search for an SEO specialist.	C	1
4	Manage tasks	4.1	As an SEO manager, I would like to add a task to the action steps list.	S	5
		4.2	As an SEO manager, I would like to edit a task from the action steps list.	S	5
		4.3	As an SEO manager, I would like to delete a task from the action steps list.	S	3
		4.4	As an SEO manager, I would like to archive a task.	C	2
5	Consult tasks	5.1	As an SEO specialist, I would like to consult the list of tasks.	S	2
		5.2	As an SEO specialist, I would like to consult the list of archived tasks.	C	2
6	Manage company websites	6.1	As an SEO manager, I would like to add a website.	M	5
		6.2	As an SEO manager, I would like to delete a website.	M	3
		6.3	As an SEO manager, I would like to edit a website.	M	5
		6.4	As an SEO manager, I would like to link a website with GSC and GA accounts.	C	8
7	Consult suggestions	7.1	As an SEO specialist, I would like to add a suggestion.	S	5

		7.2	As an SEO specialist, I would like to consult the list of suggestions.	S	3
		7.3	As an SEO specialist, I would like to consult the list of archived suggestions.	C	3
8	Audit Website	8.1	As an SEO specialist, I would like to crawl webpages from a website.	S	13
		8.2	As an SEO specialist, I would like to check the on-page issues.	S	13
		8.3	As an SEO specialist, I would like to consult the page importance list.	C	8
9	Check notifications	9.1	As an SEO specialist, I would like to check received notifications.	C	5
		9.2	As an SEO specialist, I would like to consult the list of history notifications.	C	8
10	Manage companies	10.1	As a Super administrator, I would like to create SEO manager accounts.	M	5
		10.2	As a Super administrator, I would like to consult the full list of Users.	M	2
		10.3	As a Super administrator, I would like to edit the User profile.	M	3
		10.4	As a Super administrator, I would like to delete a User.	M	5
		10.5	As a Super administrator, I would like to create a company account.	M	5
		10.6	As a Super administrator, I would like to delete a company's account.	M	3
		10.7	As a Super administrator, I would like to edit a company's account.	M	5

11	Consult support requests	11.1	As a Super administrator, I would like to consult the received requests	C	3

TABLE 3. *Project Backlog*

2.1.6 Sprint planning

The sprint planning meeting is a crucial Scrum event that establishes the tempo for the entire project and guarantees the seamless execution of each sprint. During this meeting, we discuss the planning of our work, which is scheduled over a three-month period with each sprint estimated to take between two to three weeks to complete.

Figure 4 shows the theoretical estimation of the different sprints.

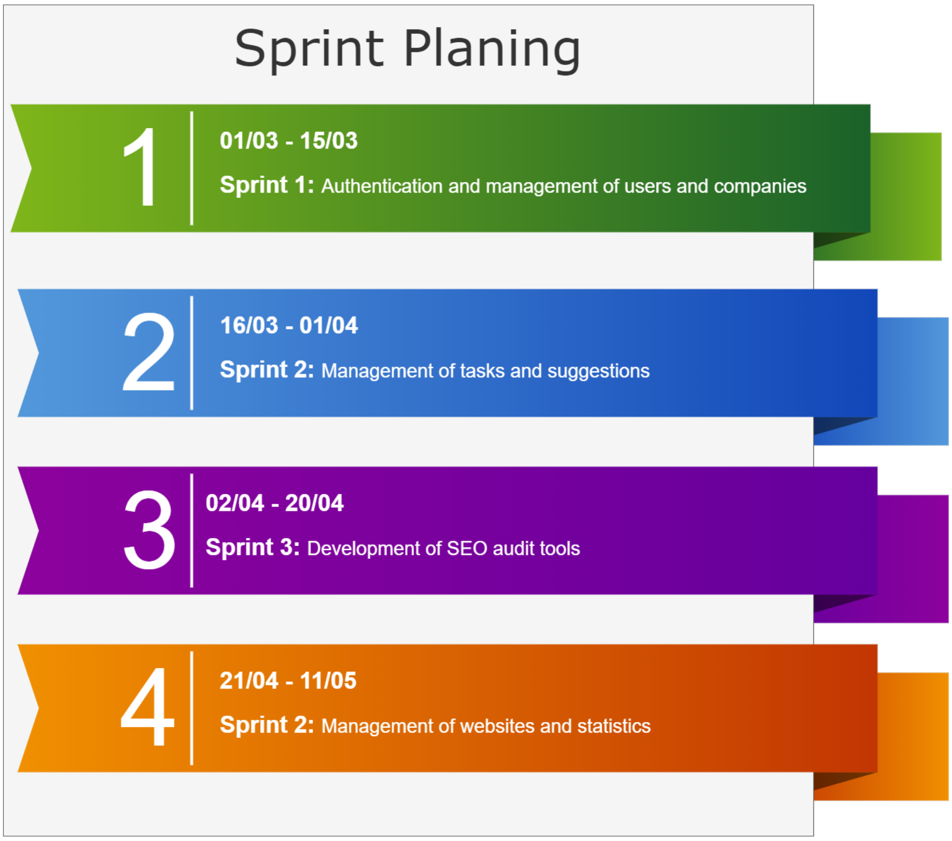
Sprint Planing
1
01/03 - 15/03
Sprint 1: Authentication and management of users and companies
2
16/03 - 01/04
Sprint 2: Management of tasks and suggestions
3
02/04 - 20/04
Sprint 3: Development of SEO audit tools
4
21/04 - 11/05
Sprint 2: Management of websites and statistics

FIGURE 4. *Theoretical sprint estimation.*

2.2 *Work Environment*

2.2.1 Hardware environment

For the realization of this project, we utilized a computer with the following specifications :

- **Processor :** AMD Ryzen™ 7 4800H
- **Operating System :** Windows 11, 64-bit version
- **Memory :** 16 GB DDR4-3200 MHz
- **Storage :** 512GB PCIe® 3.0 NVMe™ M.2 SSD
- **Graphics :** NVIDIA® GeForce® GTX 1660Ti, 6GB GDDR6

2.2.2 Software environment

Table 4 shows the software tools that will be used in the project.

Tool	Description
VSCode :	Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS, and Linux.[6]
Postman :	Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration so you can create better APIs faster. [7]
Draw.io :	diagrams.net (previously draw.io) is a free and open source cross-platform graph drawing software developed in HTML5 and JavaScript. Its interface can be used to create diagrams such as flowcharts, wireframes, and UML diagrams. [8]

Overleaf :	Overleaf is a collaborative cloud-based LaTeX editor used for writing, editing and publishing scientific documents. It partners with a wide range of scientific publishers to provide official journal LaTeX templates and direct submission links. [9]
Figma :	Figma is a collaborative web application for interface design, with additional offline features enabled by desktop applications for macOS and Windows.[10]
Github :	GitHub, Inc. is an Internet hosting service for software development and version control using Git. It provides the distributed version control of Git plus access control, bug tracking, software feature requests, task management, continuous integration, and wikis for every project.[11]

TABLE 4. *Software environment*

2.2.3 Development environment

Table 5 shows the development tools we will use in the project.

Tool	**Description**
React js :	React.js, more commonly known as React, is a free, open-source JavaScript library. It works best to build user interfaces by combining sections of code (components) into full websites. [12]
Node js :	Node.js is a cross-platform, open-source server environment that can run on Windows, Linux, Unix, macOS, and more. Node.js is a back-end JavaScript runtime environment, runs on the V8 JavaScript Engine, and executes JavaScript code outside a web browser.[13]

HTML :	HTML, or Hypertext Markup Language, is a markup language for the web that defines the structure of web pages. [14]
Tailwind css :	Tailwind is a CSS framework that provides us with single-purpose utility classes which are opinionated for the most part, and which help us design our web pages from right inside our markup or .js/.jsx/.ts/.tsx files.[15]
phpMyAdmin :	phpMyAdmin is a free software tool written in PHP that is intended to handle the administration of a MySQL or MariaDB database server. [16]
MySQL :	MySQL is a relational database management system (RDBMS) developed by Oracle that is based on structured query language (SQL).[17]
JSON :	JSON is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays.[18]

TABLE 5. *Development environment*

2.2.4 Hardware architecture

The hardware architecture adopted for this project is the client-server architecture, a popular pattern widely used in the development of modern software applications. This architecture pattern is composed of two primary components, the client component, and the server component, that interact with one another to provide a complete application experience to the end-user. The client component is responsible for providing the user interface and transmitting user requests to the server component, while the server component processes these requests, executes business logic, and interacts with data storage systems.

To implement the client component in this project, the React.js library was chosen, providing a highly modular and customizable front-end framework that is widely used in industry and academia alike. This allowed for the rapid development of an intuitive and responsive user interface, including features such as forms, interactive charts, and dynamic content loading. The server component, on the other hand, was implemented using the Node.js runtime, providing a highly scalable and performant platform for handling network requests and executing server-side logic. This allowed for the efficient and reliable processing of client requests, including the retrieval and storage of data using a MySQL database management system and phpMyAdmin as a web-based interface for managing the database.

The choice of the client-server architecture for this project was motivated by several factors, including its flexibility, scalability, and maintainability. By separating the user interface and server logic, this architecture pattern allowed for the development of highly modular and extensible code that could be easily maintained and updated over time. Furthermore, by using established and widely adopted technologies such as React.js and Node.js, this architecture pattern ensured the long-term viability and maintainability of the application.

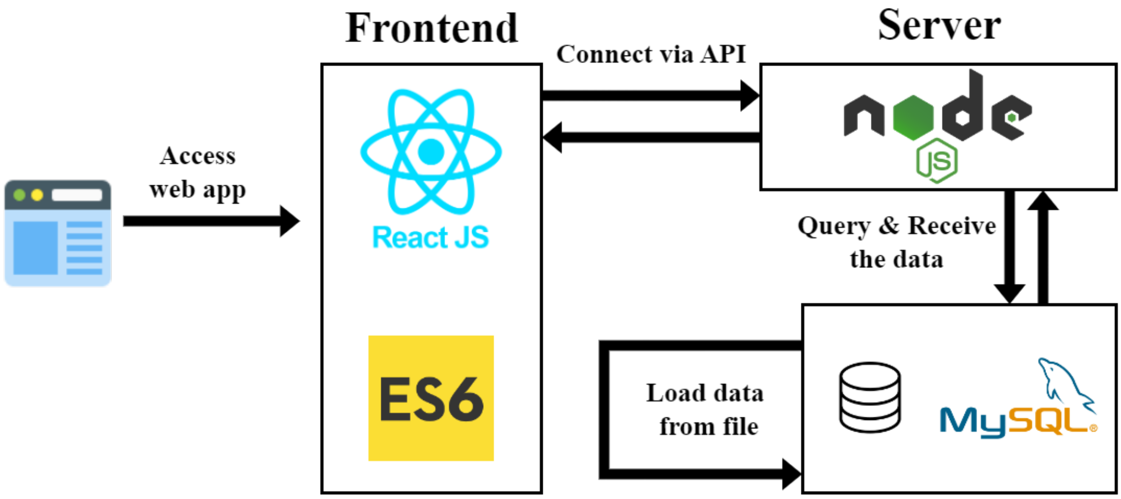

FIGURE 5. *The client-server architecture of our application*

2.2.5 Software architecture

In the process of implementing an application, one of the critical steps is to choose a design pattern that can facilitate communication, save time, and improve the overall quality of the application.

In this project, we opted for the Model-View-Controller (MVC) design pattern. The MVC pattern is a widely used architectural pattern that provides a clear separation of concerns by dividing the graphical interface of a program into three distinct entities - the model, the view, and the controller - each with a specific role in processing information. [19]

- **Model :** The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. It can add or retrieve data from the database. It responds to the controller's request because the controller can't interact with the database by itself. The model interacts with the database and gives the required data back to the controller.
- **View :** The View component is used for all the UI logic of the application. It generates a user interface for the user. Views are created by the data which is collected by the model component but these data aren't taken directly but through the controller. It only interacts with the controller.
- **Controller :** The Controller is the component that enables the interconnection between the views and the model so it acts as an intermediary. The controller doesn't have to worry about handling data logic, it just tells the model what to do. It processes all the business logic and incoming requests, manipulates data using the Model component, and interacts with the View to render the final output.

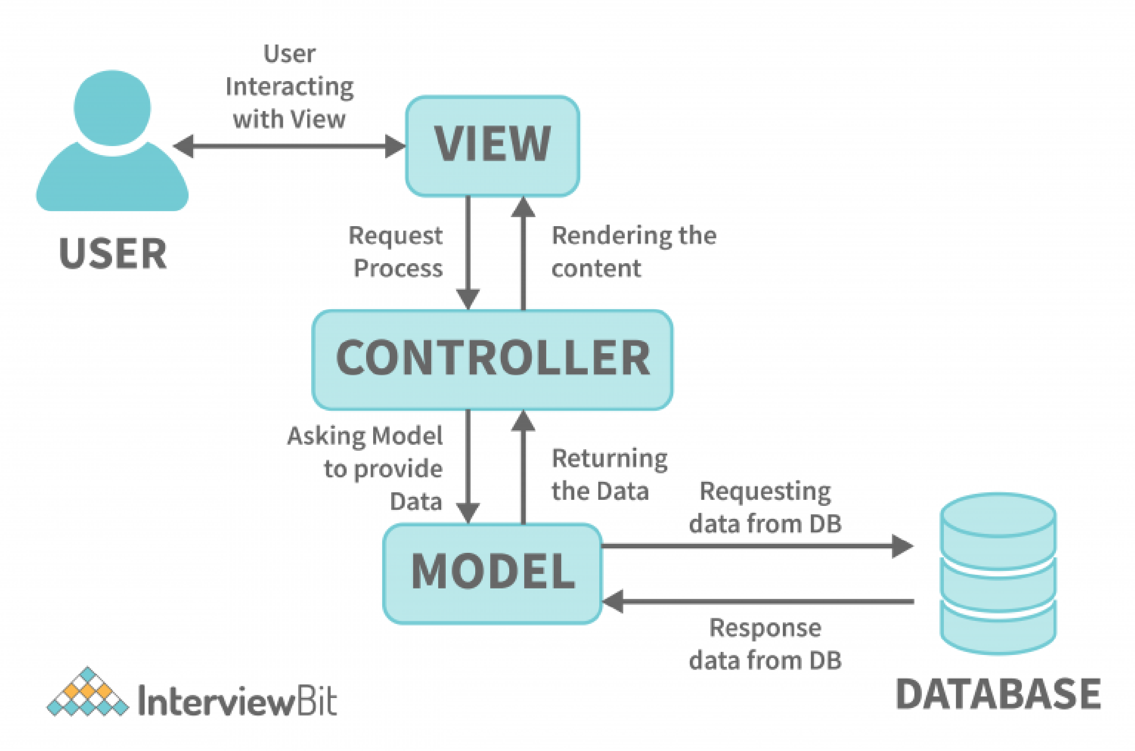

FIGURE 6. *MVC architecture [20]*

Conclusion

In this chapter, we have successfully completed the planning and architecture phase of our project. We have identified the actors and captured their requirements, which will ensure the effectiveness of our solution. We have also created a use case diagram to model the system functionalities and evaluated the hardware and software environment. Moreover, we have discussed the hardware and software architecture that will serve as the foundation of our project. In the next chapter, we will move into the first sprint of our project.