SOEN 390

Software Architecture Document

Enterprise Resource Planning

Version 2.0

BicyclERP by team 11

Zhong Hao Kevin Li	40094223
Saebom Shin	40054234
Shijun Deng	40084956
Sunny Deng	40095407
Kim Chheng Heng	26809413
Yu Xiang Zhang	40009567
Omar Salahddine	40095462
Israt Noor Kazi	40029299
Elie Sabbagh	40021696

03/02/2021

Revision History

Version	Description of Versions / Changes	Responsible Party	Date
1.0	● Initial version ● Add definition of the contents ● Add use-case, sequence diagrams ● Add detailed view points and correspond diagrams(component and deployment diagrams)	Kevin.L Shijun.D Kimchheng.H Saebom.S	Feb.2.2021
2.0	● Revise diagrams(based on feedbacks): - 4+1 model view - Use cases ● Revise stakeholder section ● Revise Architecture representation section which maps with current project: Related Artifacts ● Add use cases, sequence diagrams according to sprint 2 plan	Kevin.L Shijun.D Saebom.S	Feb 24,2021

Table of Contents

List of Figures

List of Tables

1.0 Introduction

The document presented is intended to illustrate a high level overview of the enterprise resource planning (ERP) system of a company that specializes in the manufacturing of road and mountain bikes.

1.1 Purpose

The Software Architecture Document (SAD) can be considered to serve as a map of the bike manufacturing ERP system. In our project, BicyclERP system is being mapped based on the “4+1” view model of software architecture by Philippe Kruchten, which provides clear visualization of the system via diagrams and other visual representations.

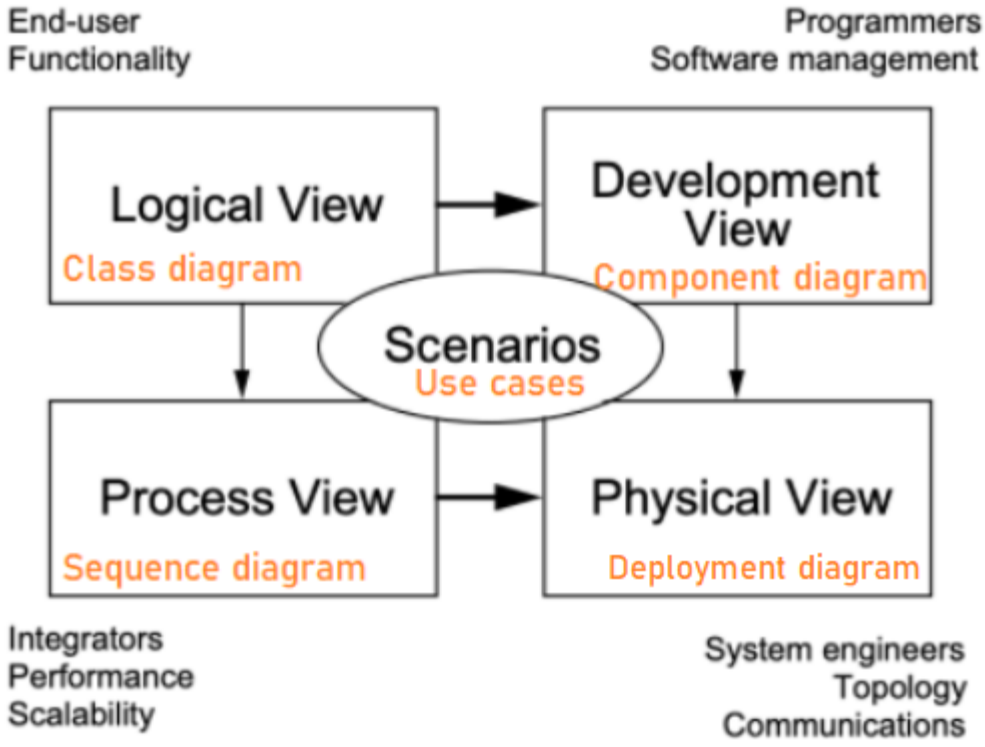

Figure 1. The “4+1” view model maps to BicyclERP system[1].

To address and deal with software-intensive systems, the use of multiple and concurrent views, as shown in the figure above, will allow stakeholders to address different concerns regarding the architecture and to handle its functional and non-functional requirements. To apply to our current BicycleERP system, a total of 5 types of visual representations have been made: use case, class, component, sequence, and deployment diagrams. A deeper explanation of how these diagrams model our system will describe in section 2.0 Architecture representation and throughout the document.

1.2 Scope

The scope of this SAD itself is to explain the architecture of the bike ERP system. On the other hand, due to time and resource constraints, the scope of the ERP system will be limited to road and mountain bikes. Nevertheless, many more types of bicycles can later be added to the system.

1.3 Definitions, Acronyms, Abbreviations

- **Spring Boot** – application framework and inversion of control container for the java platform, used to build the back end of the ERP web application.
- **React** – an open-source, front end, JavaScript library for building the user interfaces or UI components of the ERP web application.
- **MySQL** – is an open-source relational database management system used for the ERP web application.
- **WWW** – World Wide Web
- **UML** – Unified Modeling Language
- **User** – User who is registered on the ERP site. The user can be a client or admin.
- **SAD** – Software Architecture Document
- **CRUD** – Create, Read, Update, and Delete

1.4 References

[1]. Paper published in IEEE Software 12 (6) November 1995, pp. 42-50
http://web.ist.utl.pt/~fabio.ferreira/material/as/4+1view-architecture.pdf

1.5 Overview

To capture the architecture design and decision or the ERP system, the Software Architecture Document is split into various sections. A description of each of them is shown below.

Section 2: Architecture representation

Section 3: Architectural goals and constraints

Section 4: Stakeholders and concerns

Section 5: Viewpoints and views

Section 6: Architecture and rationales

2.0 Architecture Representation

This section gives an overview of how the architecture is broken down to answer stakeholder concerns using the Krutchen's 4+1 Architectural View Model. A more in-depth explanation can be found in section 4.

Use Case view / Scenarios

Audience: all the stakeholders of the system, mainly for the end-users.

Area: use-case view is a technique for analyzing user functional requirements, usually by using the UML to draw use cases. It describes the boundary of the system, and how the actors are related to the functional parts of the system to complete their respective tasks. This view represents the needs of the user in different scenarios and how the system utilizes different services to satisfy those needs.

Related Artifacts : Use Case Diagrams (see section 5.5)

Logical view

Audience: Analysts, Designers.

Area: It describes functionalities that the system can offer to the end-users. Without describing the relationship with the actors. It focuses on how functionalities work together within the system boundary. For a large system such as our ERP system, we can represent the logical view for different subsystems and integrate them together. We use UML diagrams such as class diagrams and state diagrams to represent the logical view.

Related Artifacts: Class diagram (see section 5.1 and Appendix I)

Process view

Audience: System Integrators.

Area: It describes the dynamic aspects of the system, and focuses on the information flow between different components. Usually, we use sequence diagrams to describe this view. Besides the flow of data, we also need to take concurrency and synchronization problems into consideration.

Related Artifacts: Sequence diagrams (see section 5.2)

Development / Implementation view

Audience: Programmers.

Area: It describes the components of the system in the perspective of programmers. Simply put, this view helps in software management. Based on it, the programmers can start to code. Usually, we use UML package diagrams to represent the development/implementation view.

Related Artifacts: Component diagram (see section 5.3)

Physical / Deployment view

Audience: System Engineers.

Area: It describes the hardware requirement or environment for each subsystem, and focuses on the topology of the system components on the hardware layer as well as the hardware connections. To represent this view, we use UML deployment diagrams or component diagrams.

Related Artifacts: Deployment diagram (see section 5.4)

3.0 Architecture Goals and Constraints

There are some key requirements and system constraints that have a significant bearing on the architecture that are to be considered:

1. One of the goals of architectural designs is to try and understand the end-user, all the while providing guidance for future architecture designers in understanding the system.
2. The ERP system is a web application which possesses 3 tier of patterns. They are the presentation layer, the domain layer and the data source layer. Each layer is driven by following chosen libraries or frameworks respectively; React, Spring and MySQL. Those 3 tier patterns are then packaged up inside a Docker to make them easier to deploy and run in any setup environment.
3. Since the ERP system contains sensitive data, certain sectors must be encrypted so that the system can be protected from any data leak.

4.0 Stakeholders and Concerns

A stakeholder is any individual, group or company that may have any type of direct or indirect relation with, responsibilities towards or interest in the project. Simply put, they are those who may affect or get affected by the outcome of the software project. Below, is a breakdown of stakeholders that have significant contribution in the success of the bicycle ERP system.

Table 1. Stakeholders and Concerns

Name	Description	Concerns
External Users (customers, suppliers, transport and shipping companies,)	External users are users outside the company, Their focus is on obtaining related information and trading via the system.	**Interests:** - Obtaining related information, such as product information (including product details, price, shipping, etc,.) for customers, orders to suppliers, shipping requirements for the transport and shipping companies - Feedback from the company, such as problem/complaint solution, service satisfaction for the suppliers, transport and shipping companies. - Trading, such as placing an order from the customer, obtaining an order via the ERP system. **Responsibilities:** - Following the instruction of the system - Providing required information accurately and timely - Providing the feedback of the services - Acting in accordance with the contracts
Internal Users (system admin,	These internal users are the people who work in the company and need to use the system in their	**Interests:** - Streamlined operations of bike manufacturing that is easily operated, to tracked and maintained

department managers, system operators)	daily works.	- Obtaining the correct information fast - Keeping all the processes in synchrony and controlled - Integrating all departments together in one system - Ensuring the consistency, safety, and reliability of data and its exchange **Responsibilities:** - Monitoring and maintaining the system to be reliable and accurate - Operating the system strictly according to the instruction - Reporting any problem when using the system - Responding to the system alarms and warnings on discovery
Project manager, architect, developers, tester, maintainer	This category of stakeholders mainly encompasses those who work hands on or closely with the ERP system that serves the production company.	**Interests:** - Provide ERP solutions. - Profit from companies using their ERP system. **Responsibilities:** - Ensure market demand. - Project funding (i.e. homegrown solutions), setting budgets, assigning developers. - Maintain product and monitor its progress.

		- Communicate with other stakeholders for use cases & system requirements. - Ensure product evolution and flexibility - connectivity etc.) to provide accurate and optimized devices.
Network, service and platform providers	These are the stakeholders that provide connectivity and allow communication to be integrated between devices and machinery.	**Responsibilities:** - Supply and handle network infrastructure and provide reliable connection between devices. - Provide hardware to support or enable connected services (i.e. internet connection) - Allow communication between devices via Bluetooth, NFC or LAN microchips.
Regulators	These may include policy makers, stakeholders who ensure standard of operation and domain experts in the government.	**Responsibilities & interests:** - Overseeing and provision of regulatory framework with appropriate targets and subsidies which gives a direction to entrepreneurialism and innovation towards carbon imprint/emission reduction and energy efficiency solutions. - Address ethical issues of technology. - Address working conditions of the assembly line.
Negative users	This category include hackers and other malicious users	**Concerns:** - Does the ERP system protect all forms of unauthorized access? - Is appropriate access given to the wrong type of users? - Is data properly encrypted?

5.0 Viewpoints and Views

5.1 Logical View

The logical view's main concern is with the functionality (i.e. functional requirements) that the ERP system provides to end-users. Here, the end-users are those who operate the ERP system within the bicycle manufacturing company.

More specifically, this view takes the problem domain, and decomposes it (the ERP system) via abstraction into objects or object classes. This allows for not only the functional analysis but also, to identify the design elements across various parts of the system. Typically the use of various types of representation such as UML, class or state diagrams are employed with an object-oriented (OO) style. The following diagram tries to illustrate a simple description of the system behavior while avoiding premature specialization of classes that are yet to be determined in the following sprints of the ERP system.

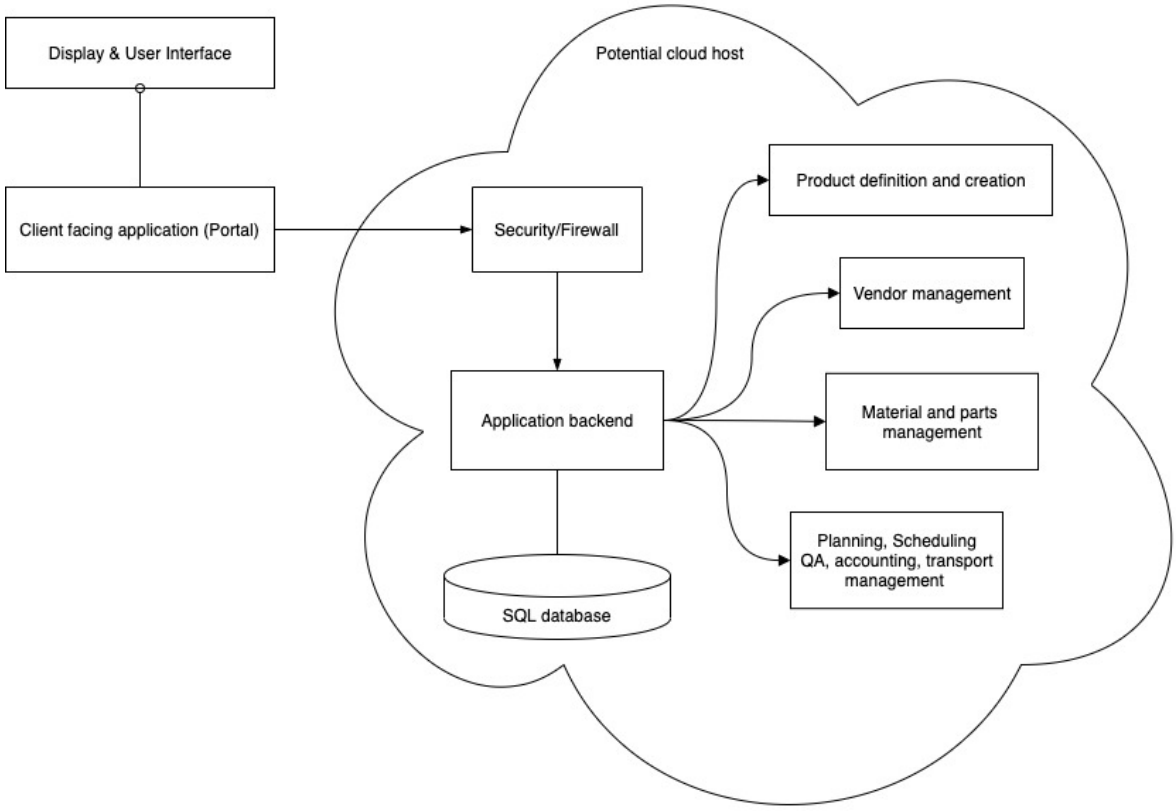

Figure 2. The logical view for the ERP system

- **Client facing application:** As either a web or mobile application, the portal utilizes display and UI to allow interaction with its users. The application consumes the ERP system's API.

- **Security/Firewall:** Protects the system against unwanted operations and network attacks. Users are given different access according to their title.

- **Application backend:** Business logic that allows for the creation of a bike all the way to it being shipped out to consumers. This includes multiple sub-categories of management systems separated and abstracted according to use cases.

- **Potential cloud host:** Possible service to use to host the bicycle ERP system in future iterations.

- **Database:** The database utilized for the ERP system is mySQL.

5.2 Process View

The process view focuses on the dynamic aspects of a given system. Here, it explains the run-time behavior of the bike ERP system and addresses issues such as concurrency. For each use case covered in the use-case view, there is a corresponding sequence diagram in the process view which illustrates the sequence of control, concurrency and passing of data within the ERP system.

5.2.1 Authentication

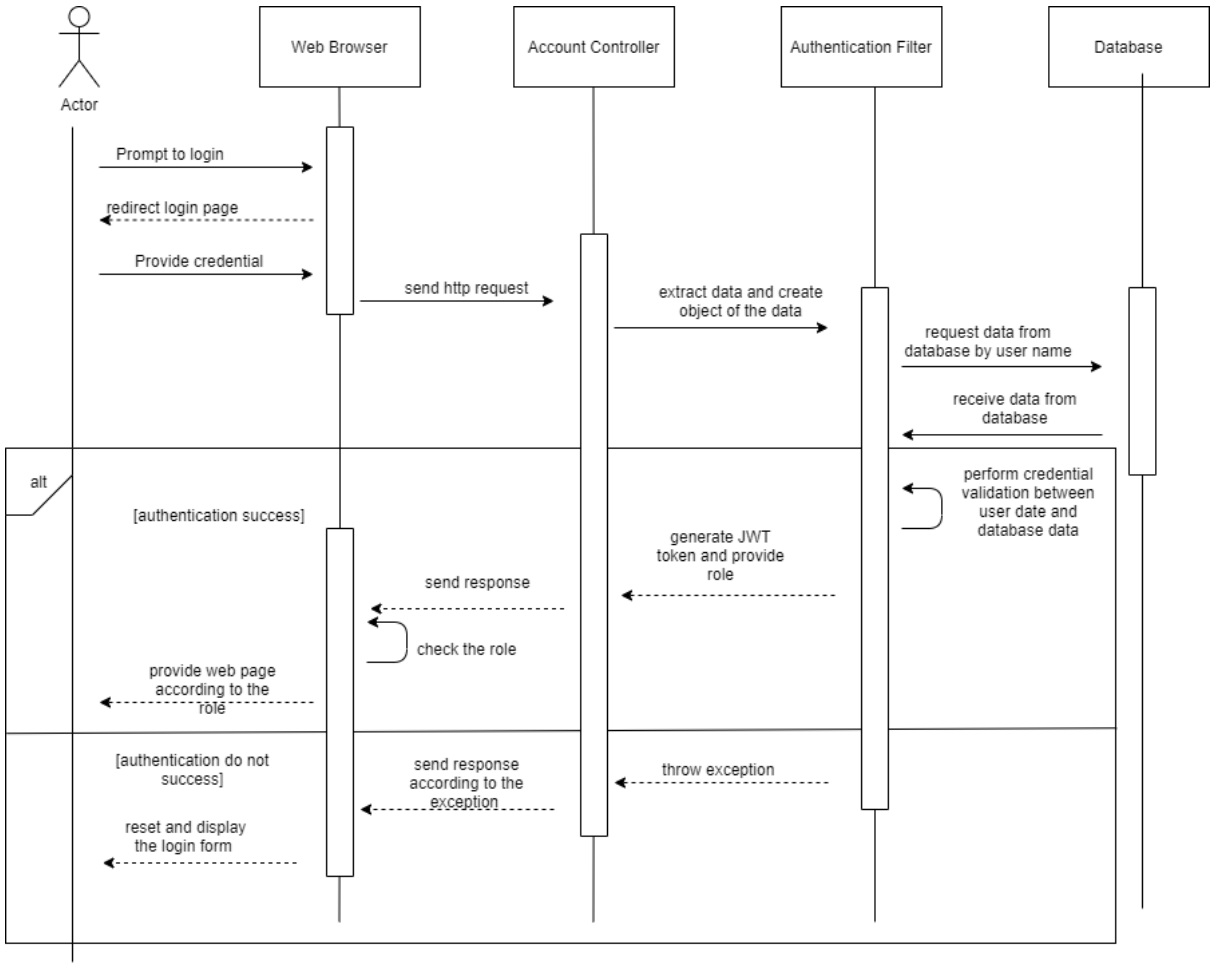

Figure 3. The process view of the login functionality

5.2.2 Signup

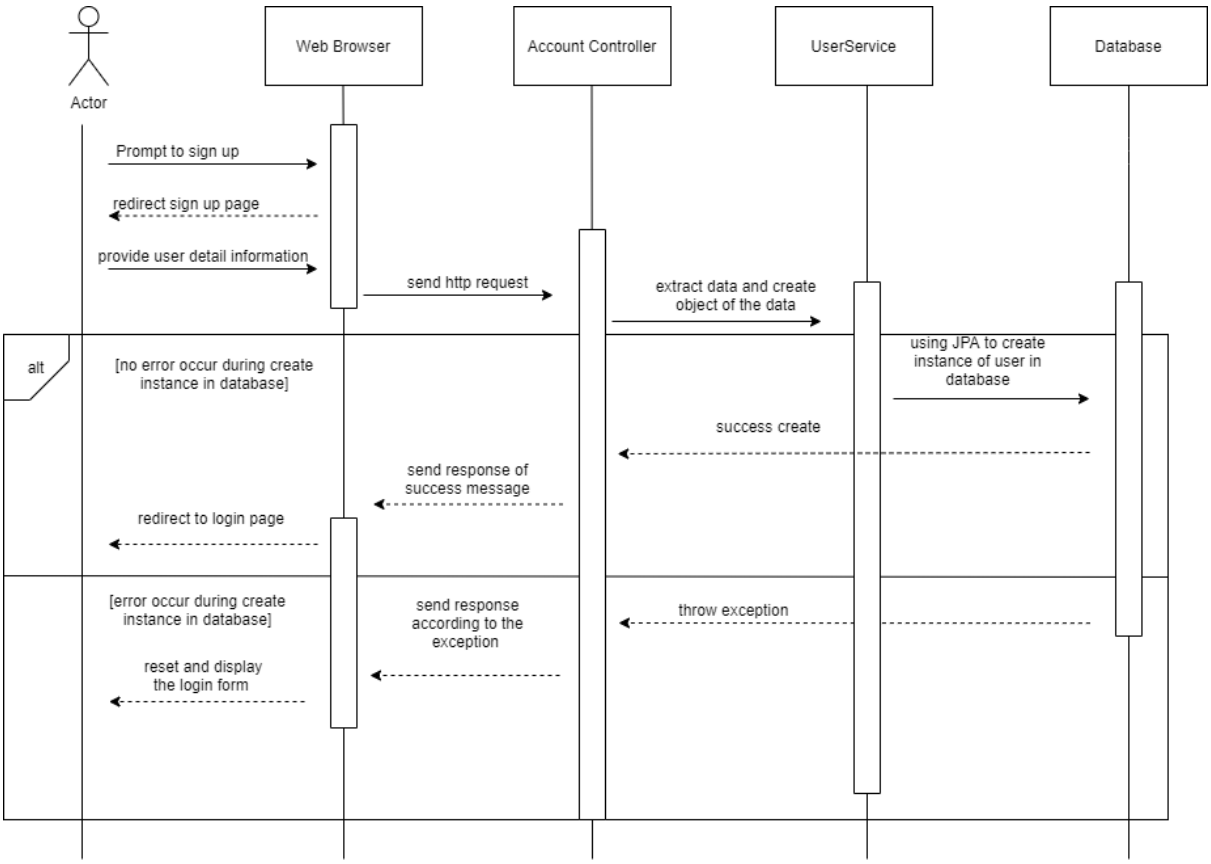

Figure 4. The process view for the signup functionality

5.2.3 General CRUD Operations

There are many CRUD operations in our system, such as product, materials, raw materials, user account management, etc,. Following are the general functionalities of these CRUD

Operations:

Create: create a new item based on the input info. This functionality will allow the authorized users to input the information of the new item, save it to the database, and display the updated information on screen.

Read: display the corresponding information of a list of items. This functionality will allow the authorized users to retrieve all the data of some item and display on screen.
Update: update the information of a certain item. This functionality will allow the authorized users to modify the information of a certain item, update the database with the modified information, and display the updated information on screen.

Delete: delete a certain item from the database. This functionality will allow the authorized user to choose an item, delete it from the database, and display the updated information on screen.

Following are the general process of CRUD Operations in the system.

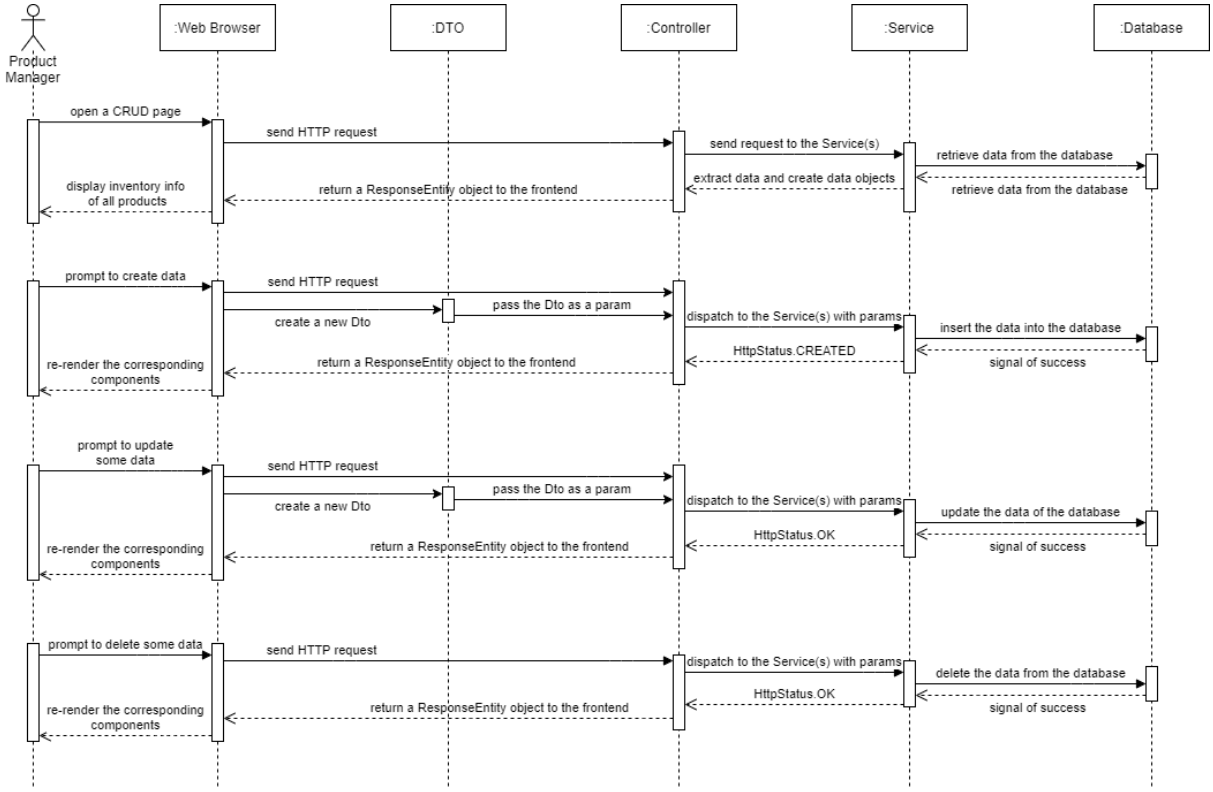

Figure 5. The sequence diagram for general CRUD operations

5.2.4 Product Access

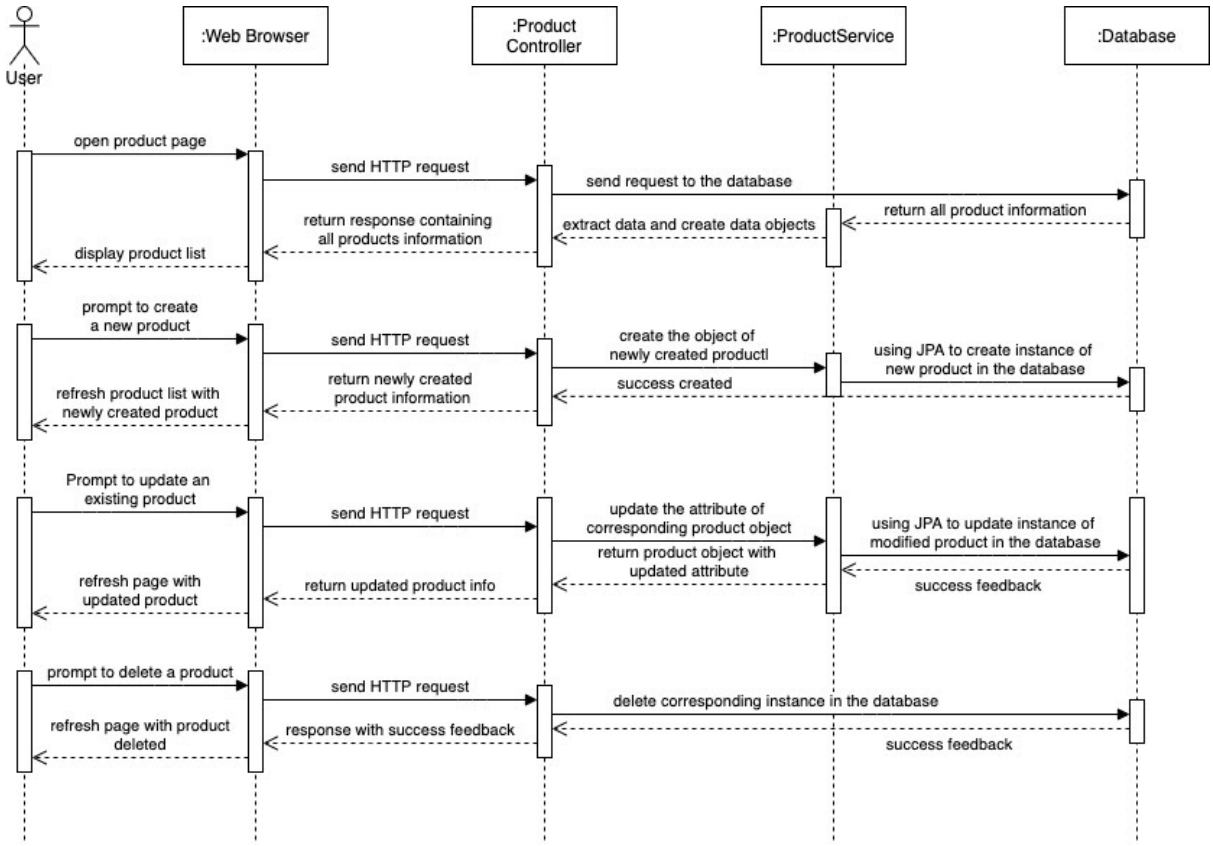

Figure 6. Product access sequence diagram

5.2.5 CRUD Material list

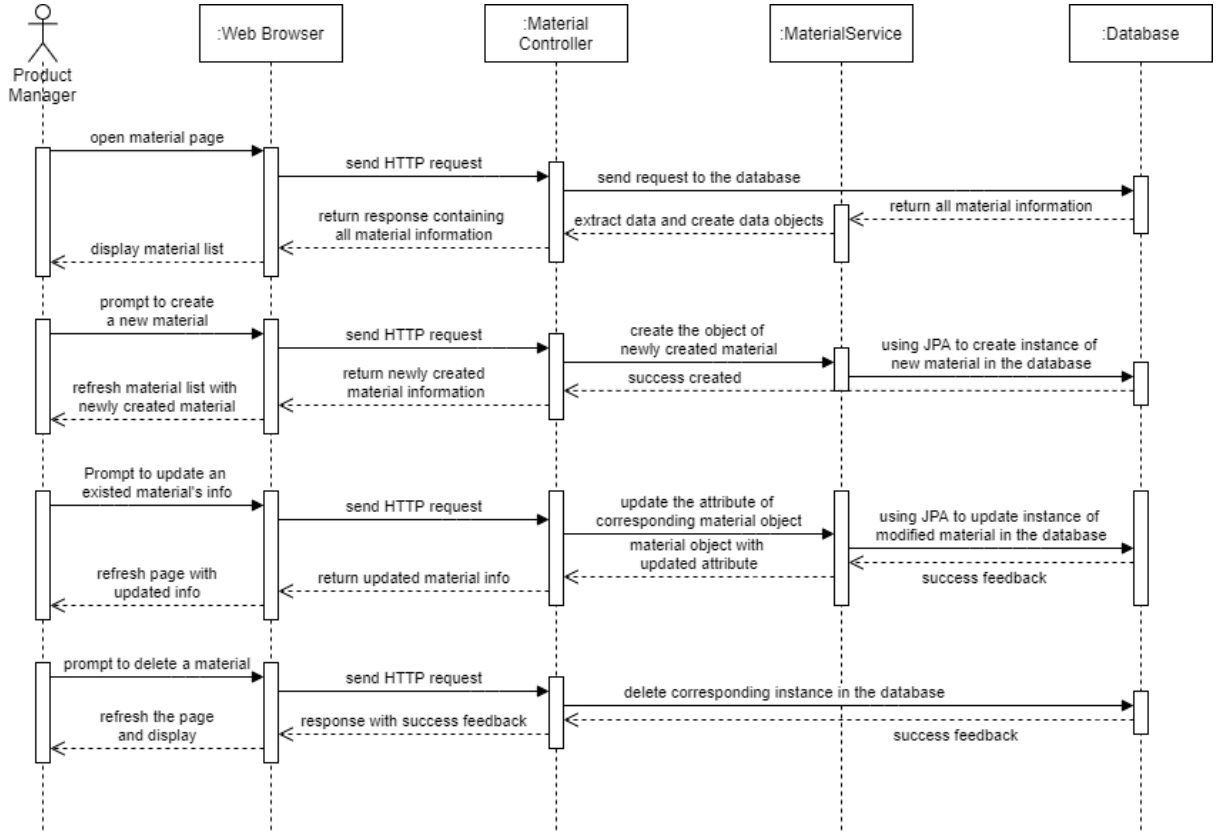

Figure 7. Sequence diagram of CRUD operations for material list.

5.2.6 CRUD Product Inventory

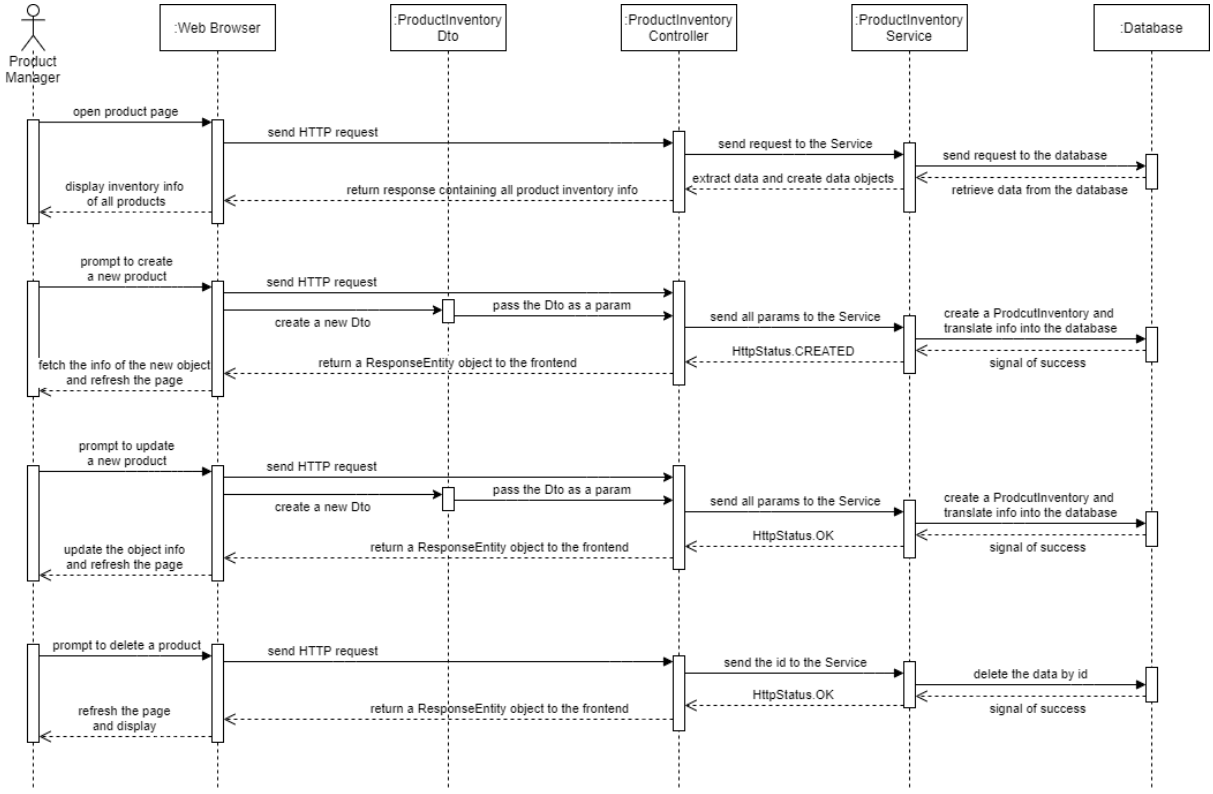

Figure 8. Sequence diagram of CRUD operations for product inventory.

5.3 Implementation View

5.3.1 Purpose of the Implementation View

The implementation view helps to understand the architecture that binds the source code. It describes the components of the system in the perspective of programmers. Explicitly, the ERP system is built using the Model-View-Controller architectural pattern. This separates the application into three main logical components: the model, the view, and the controller. Each of these handle specific development aspects of the application.

5.3.2 Overall Implementation

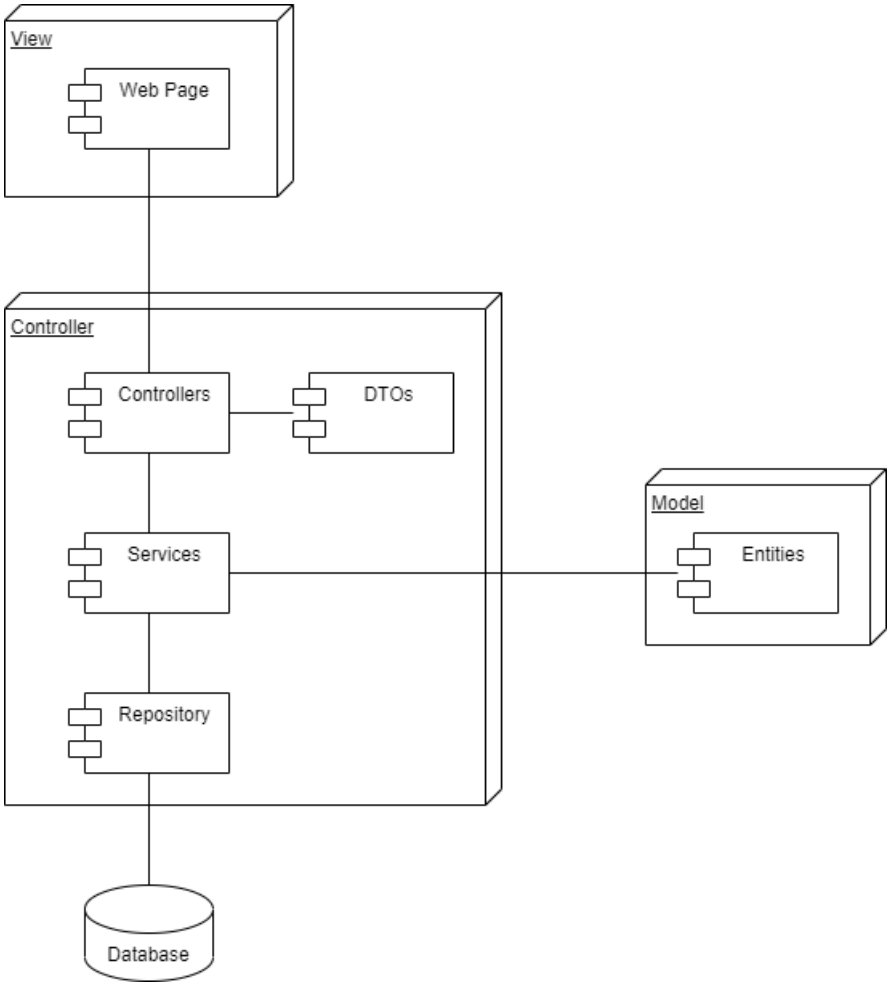

Figure 9. Overall development view.

Model: in this bike ERP system, the classic “model” is represented as “entity”. To name a few, Material, Part, Product and userAccount are only some of the entities currently present in the system. They represent data that can be transferred between the View and Controller components. As an example, the Product entity has its logic tied to the Part entity (i.e. a

product has many parts), and both are used to simulate real world entities and serve as data transfer objects that are quintessential to the logic of the ERP system. Moreover, a Product entity has its information in the database, and each Product may be created by an user action and have its attributes manipulated.

View: this component is used for all the UI logic of the application. The end users are to interact with this component in order to perform their tasks. For example, an administrator may interact with the buttons and icons on the production page of the ERP system to add a new product to put in production.

Controller: in short, controllers act as interfaces between the Model and View components. They contain and process all the business logic of actions from incoming requests and returned responses (i.e. RESTful Services). For instance, an administrator may send a request to update the parts that go into making a bike, and this will be handled by the controller.

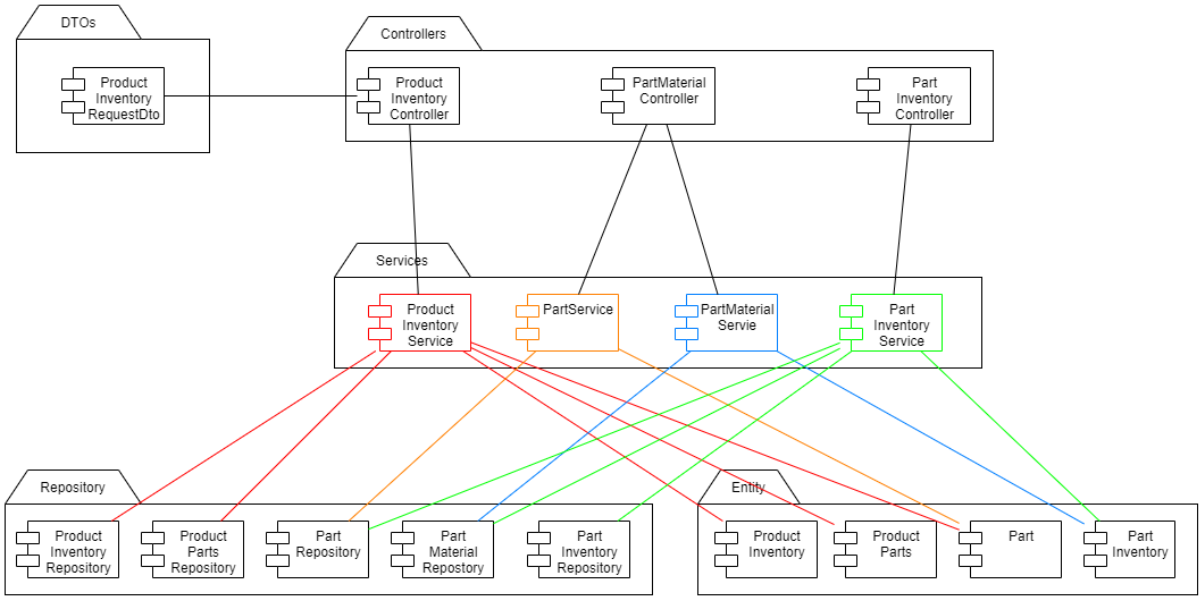

Figure 10. Backend implementation view.

5.4 Deployment View

5.4.1 Purpose of Deployment View

The purpose of the deployment view is multifolds. First, it helps display the components of the hardware and how these components communicate with each other. Then, it projects the software system into the hardware configuration it shall take. It also helps to design the software architecture, and finally, serves to document the deployment of the system for future reference.

5.4.2 Overall Deployment

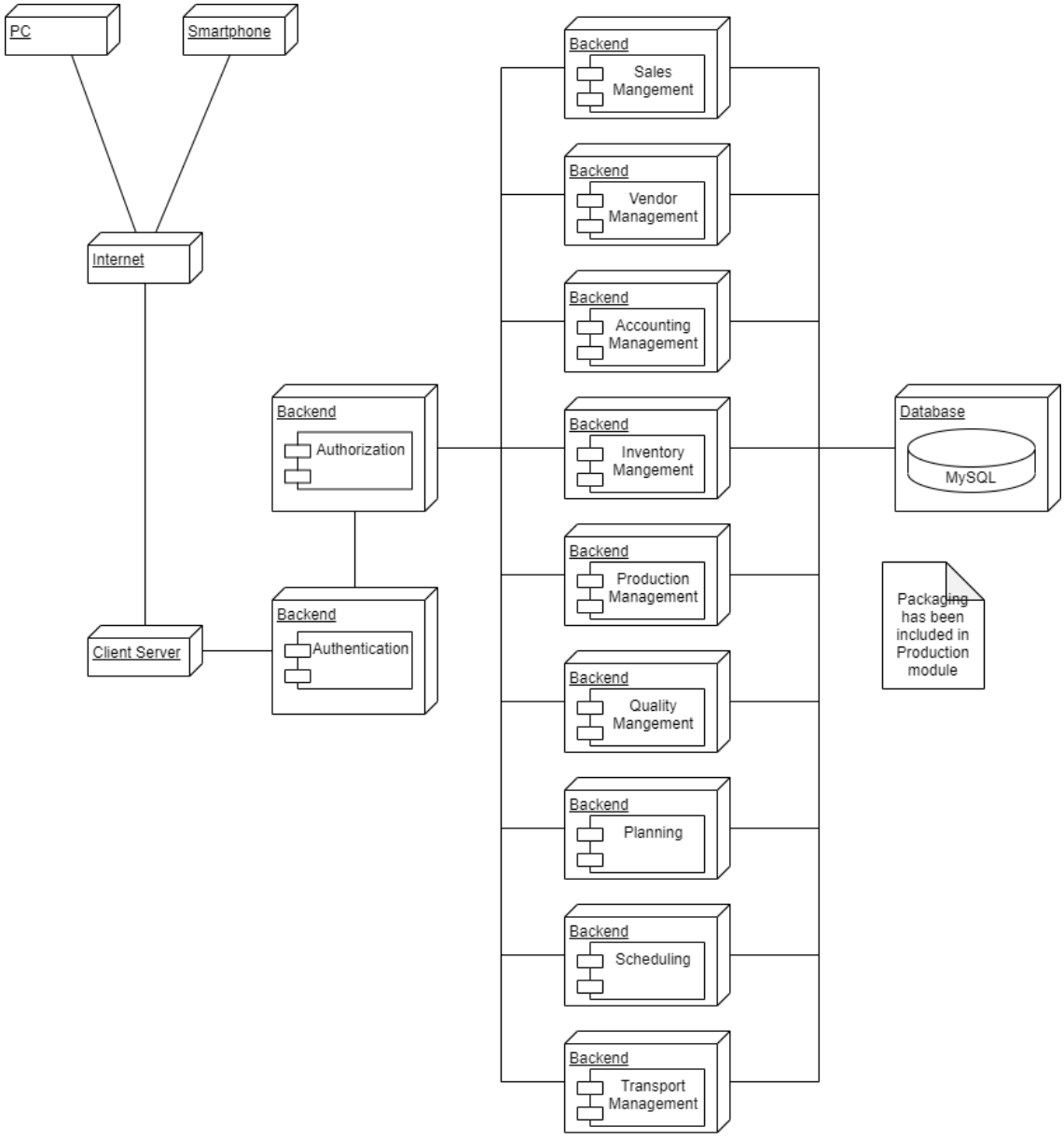

Figure 11. The deployment view for the ERP system.

5.5 Use Case View

In this section, the use-case view is shown. The use cases are a set of scenarios and actions experienced and performed by users that represent some significant central functionality of the ERP system. Currently, the SAD covers the authentication, product access, and material list manipulation use cases.

5.5.1 Authentication

- **Login:** Login is the basic step of the requirement in order to use the ERP system.
- **Authentication filter:** Filter confirms that if you are a valid or invalid user.
- **Generate JWT for authorization:** Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with the token.
- **Registration:** It gives users to have personality and recognized identity.
- **Checking valid email:** In our ERP system, it accepts email as of the user ID. Therefore, when a user registers a new account, the system must check that email is valid or not.

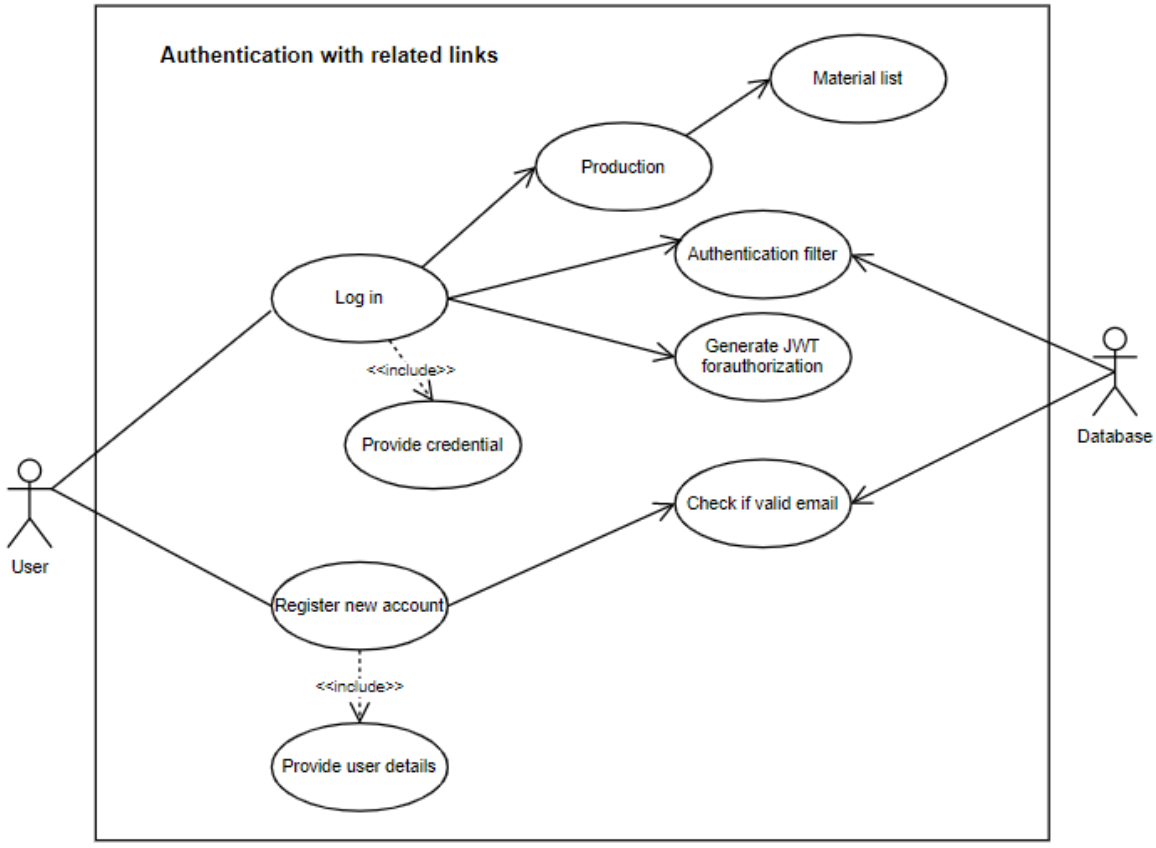

Figure 12. The use case diagram for Authentication functionality.

5.5.2 Product Access

- **Access products:** Registered users can access the catalog and detail of available products.
- **Search product list:** Any user can search the catalog of available products regardless of title.
- **Create new product:** Administrators can create new products and add it to the product list/catalog.

  Delete existing product: Administrators can delete existing products and thereby initiate a chain of command to remove production of a given model and dispose or transfer raw material and/or parts.
- **Modify existing product:** Administrators modify materials and parts that go in the making of existing products.
- **Input validation and user authorization:** The system validates user input and verifies them against database and availability (i.e. of raw material or parts). The system also ensures to give the right access privileges depending on the type of user account.

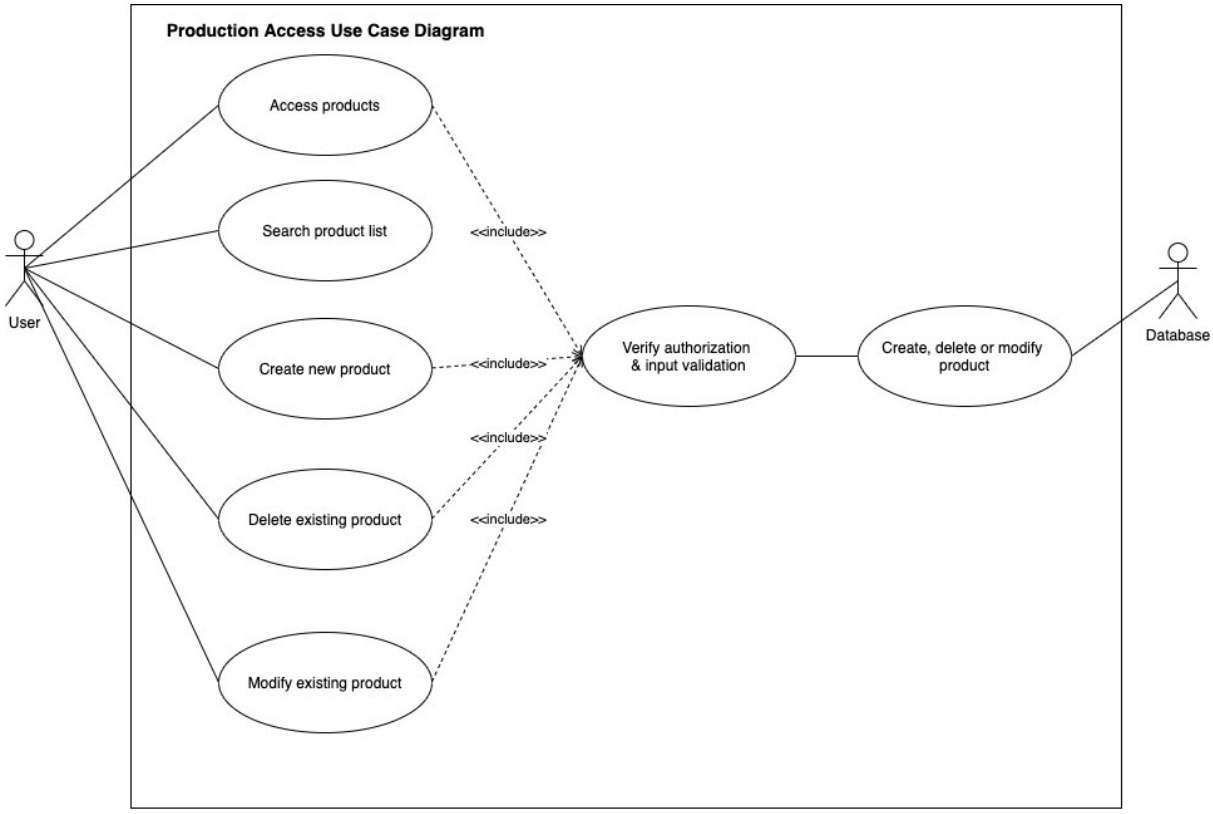

Figure 13. Product Access Use Case Diagram.

5.5.3 CRUD Material list

The Admin is allowed to do the CRUD operations to the material list. From the requirements, the system should allow the following:

- Authorize the product manager to manipulate the material list and prevent unauthorized users to access these functionalities
- Authorized users should be able to create, read, update and delete material list.
- Create, Update and Delete operations should be able to modify the corresponding data in the database, and the displayed material list should be refreshed after the operations have been done so that the users can see the correct record.
- Validate the inputs for creating and updating the material list.

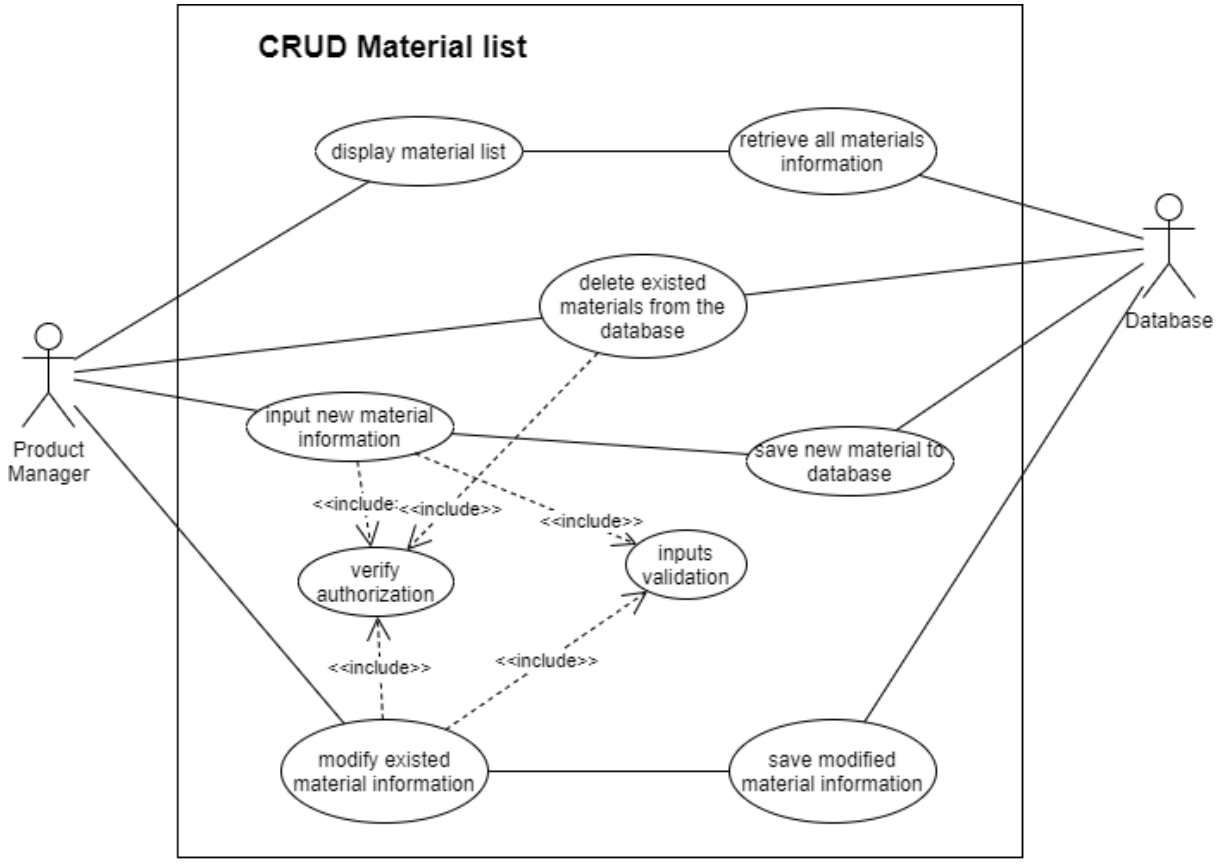

Figure 14. Use case diagram of CRUD operations for material list.

5.5.4 CRUD Product Inventory

The Admin is allowed to do the CRUD operations to the product inventory. From the requirements, the system should allow the following:

- Authorize the product manager to manipulate the product inventory and prevent unauthorized users to access these functionalities
- Authorized users should be able to create, read, update and delete product inventory.
- Create, Update and Delete operations should be able to modify the corresponding data in the database, and the displayed product inventory should be refreshed after the operations have been done so that the users can see the updated record.
- Validate the inputs for creating and updating the product inventory.

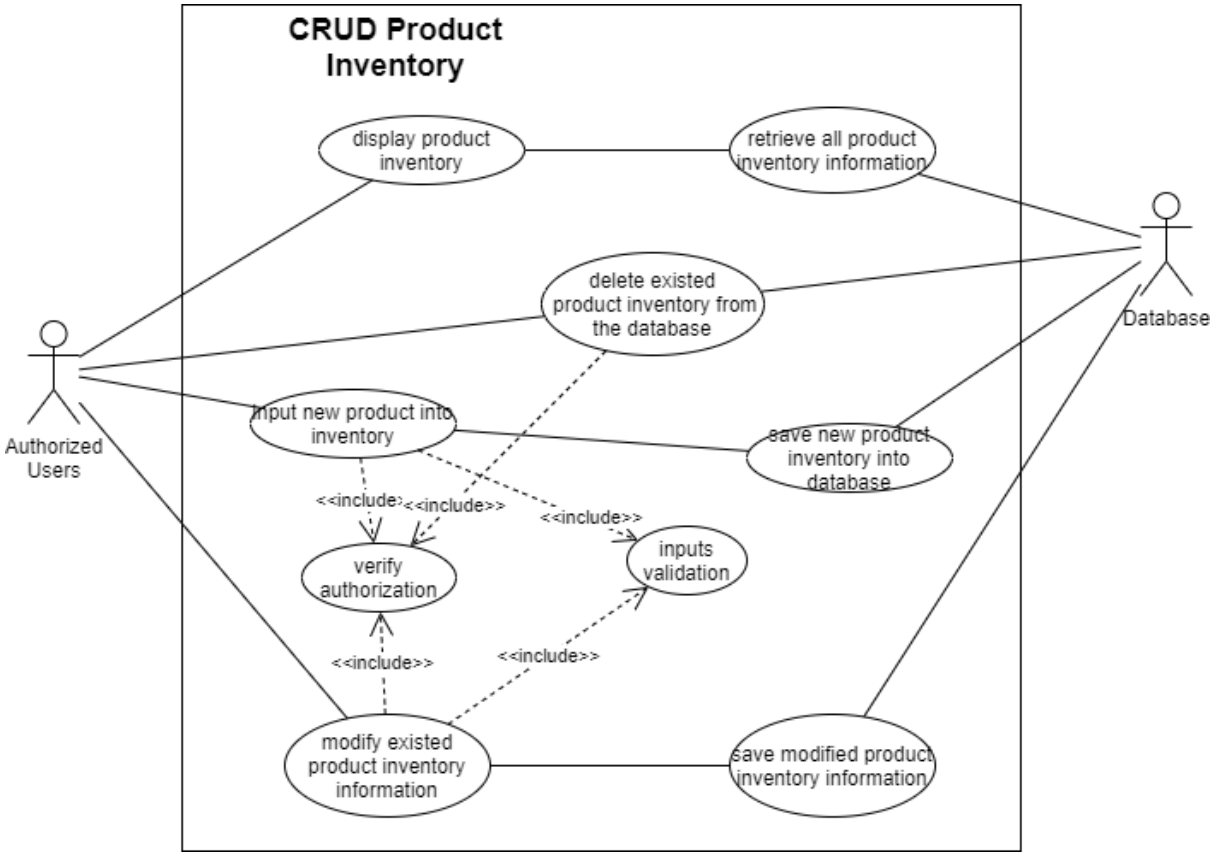

Figure 15. Use case diagram of CRUD operations for product inventory.

5.5.5 Product definition

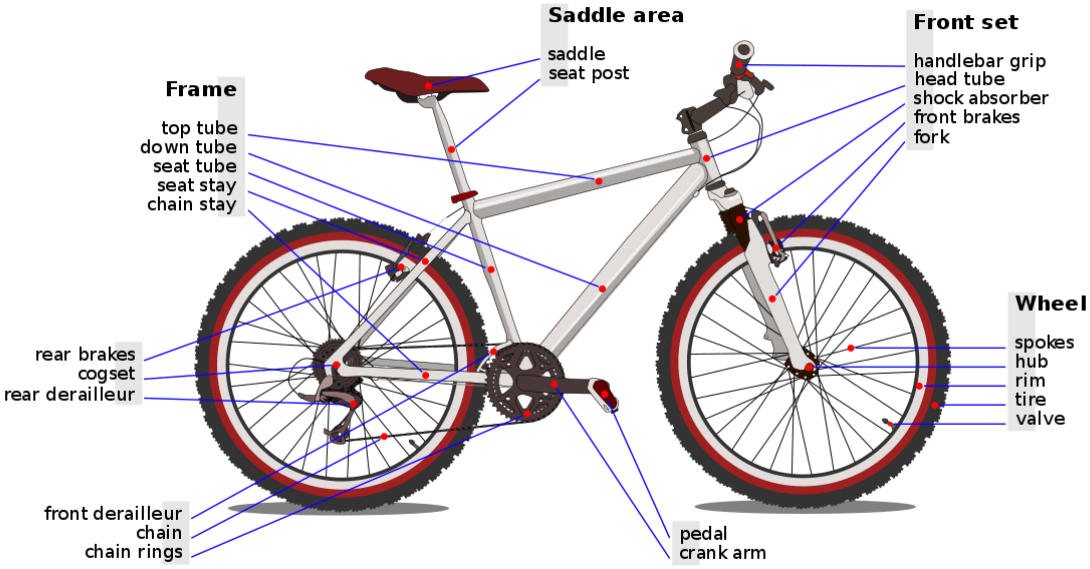

Figure 16. Product definition.

First, to better understand the flow of our operations, refer to the above bike figure to understand the definition of each subcategory that goes into building a bike. Each bike (product) is composed of parts (i.e. front set, saddle area etc.), which are themselves composed of materials (i.e. handlebar grip, head tube etc. for front set), which are also themselves composed of raw material (i.e. 100g of steel alloy, 6 screws, 2 springs).

The following diagram illustrates actions that the administrator accounts are able to perform. It is to be noted, however, that the backend code of some of these actions exist but are yet to be completely implemented in the following sprint (3). This is the case for the use cases that follow. Here, all the system functions will be listed, but the focus is on the product definition (i.e. the flow of “Define product”).

- **Define product:** an administrator may visit the production tab of the ERP system and create a new product. That is, they define a bike to be constructed and give it a name, a type, a size, a main color theme, its finish and its grade.
    - **Choose parts:** following the previous step, the administrator shall be able to choose the parts that go into the construction of the bike from a set of predefined parts.

    - **Generate material list:** now that the parts are chosen, the system will retrieve from the database the list of material that goes into making each part of the bike.
    - **Generate raw material list:** also, the system will retrieve from the database the list of raw material that goes into making each material. At this point, 3 tables will be generated (i.e. parts, material, raw material) and the user may choose to view these tables along with the quantity currently in inventory.
- **Edit product parts:** once an administrator completes the definition of a product, they may be allowed to swap out individual product parts for an alternative, but also pre-set one.
- **Make product:** once the product is defined, the user may decide to make the defined product (i.e. defined bike). To do so, the user will visit the parts table where the parts required to make the defined bike are listed. If the inventory column shows that there are enough of each part to produce the bike in question, the user may produce that bike. This action, then, decrements the number of each part in the inventory and increments the number of bikes to be produced accordingly.
- **Make part:** if the user wants to make parts or there are not enough parts to produce the wanted number of bikes, the user must visit the material table of the parts in question. Over there, once again, they may produce the part needed, given that there is enough material to perform such action.
- **Make material:** if the user wants to make materials or there is not enough material to produce the wanted number of a part, the user must visit the raw material table of the material in question. They may produce the material needed, given that there is enough raw material to perform such action.
- **Get raw material:** if the user wants to make raw materials or there is not enough raw material to produce the wanted number of a material, the user must order raw material from vendors inside of the raw material table.
- **Alternative:** if an user would like to create a bike, but there are missing parts, performing the action of making a bike will cascade into every “Make” level previously explained and automatically make/order the necessary amount of parts and/or materials and/or raw materials.
- **Show cost:** display cost of making the bikes.

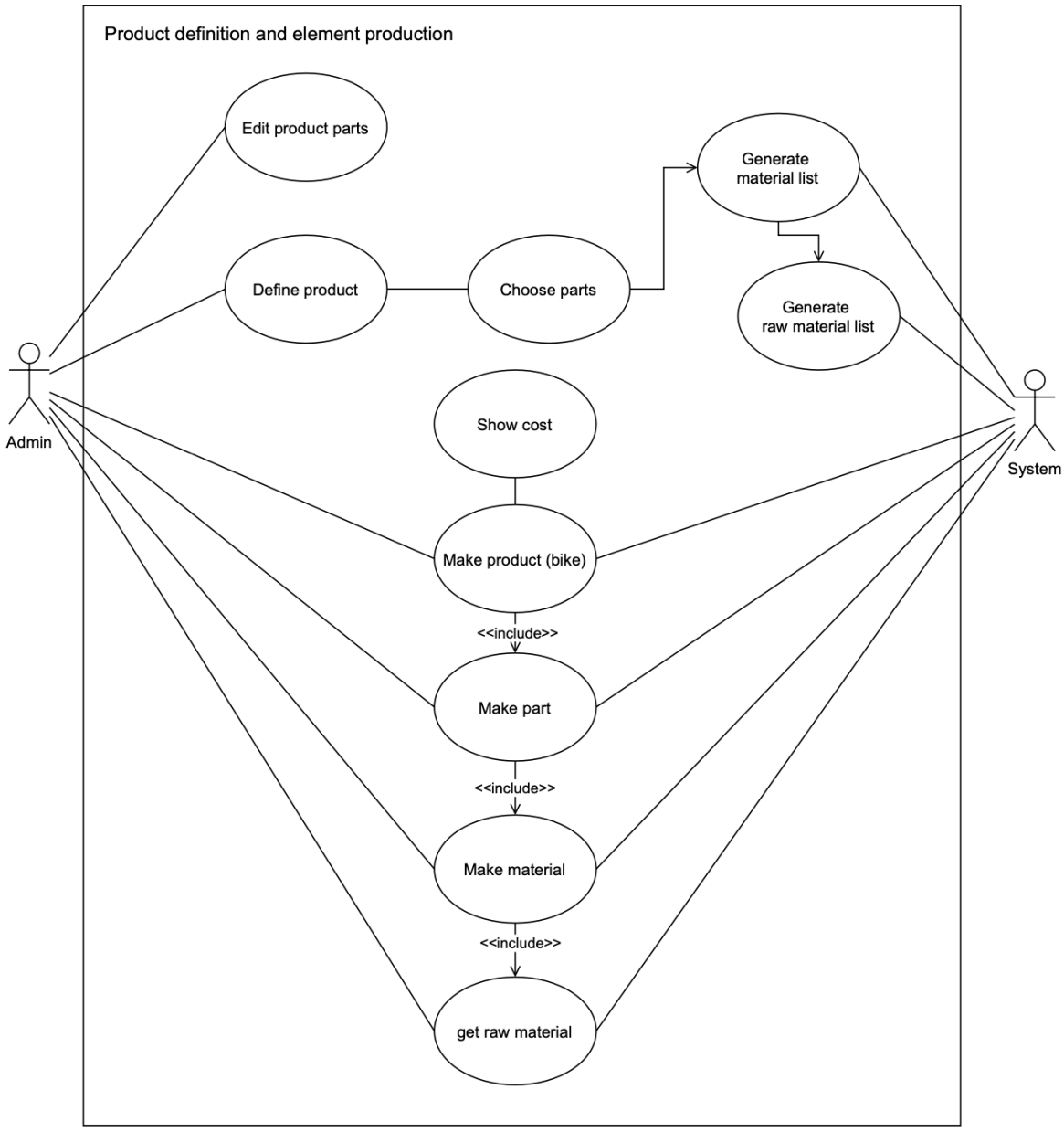

Figure 17. Use case diagram of Production definition and element production.

5.5.6 Make raw material (make order)

- **Log in:** user needs to be logged in as an admin to be given the privilege to make raw material orders.
- **Choose product:** pick a product given the existing product listed in Production tab of the main page of the ERP system.
- **Choose part:** pick a part on the parts page
- **Choose material:** pick a material on the materials page
- **Verify inventory:** the system will show the number of raw material that goes into making the material in the inventory column.
- **Order raw material**: user may now order raw material with selector vendors. This will increase the inventory of the specific raw material.

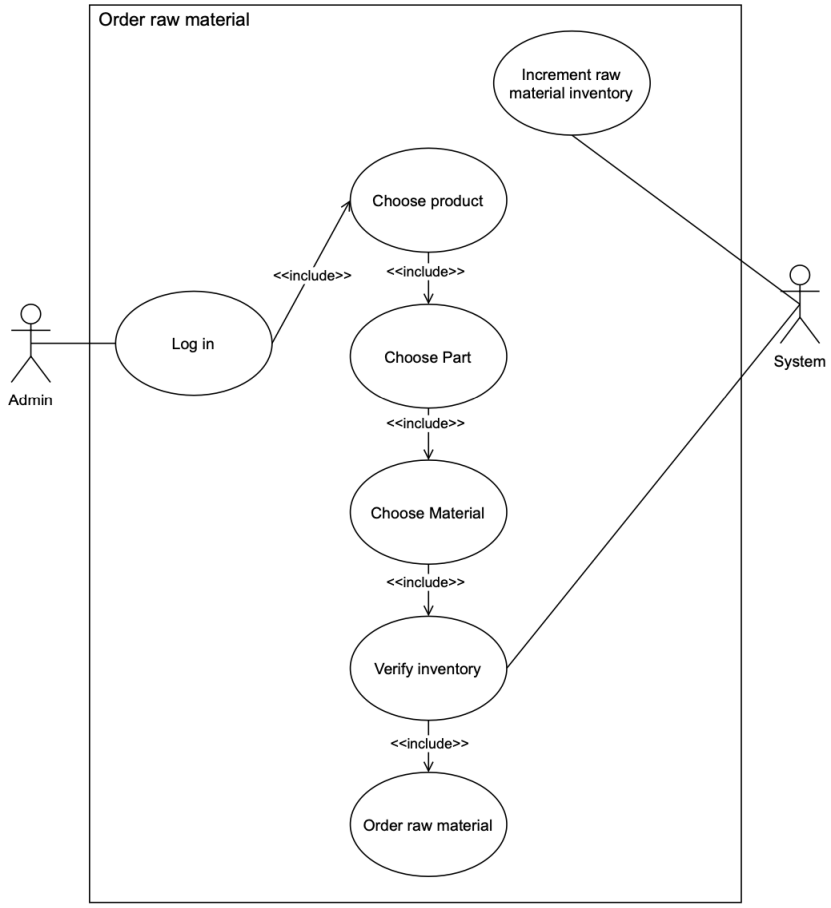

Figure 18. Use case diagram of Order raw material.

5.5.7 Make material

- **Log in:** user needs to be logged in as an admin to be given the privilege to make materials.
- **Choose product:** pick a product given the existing product listed in the Production tab of the main page of the ERP system.
- **Choose part:** pick a part on the parts page
- **Verify inventory:** the system will show the number of material that goes into making the part in the inventory column.
- **Make material**: user may make material if enough raw material is in inventory. This will increase the inventory of the specific material, and decrement all the raw materials that go into making the material.

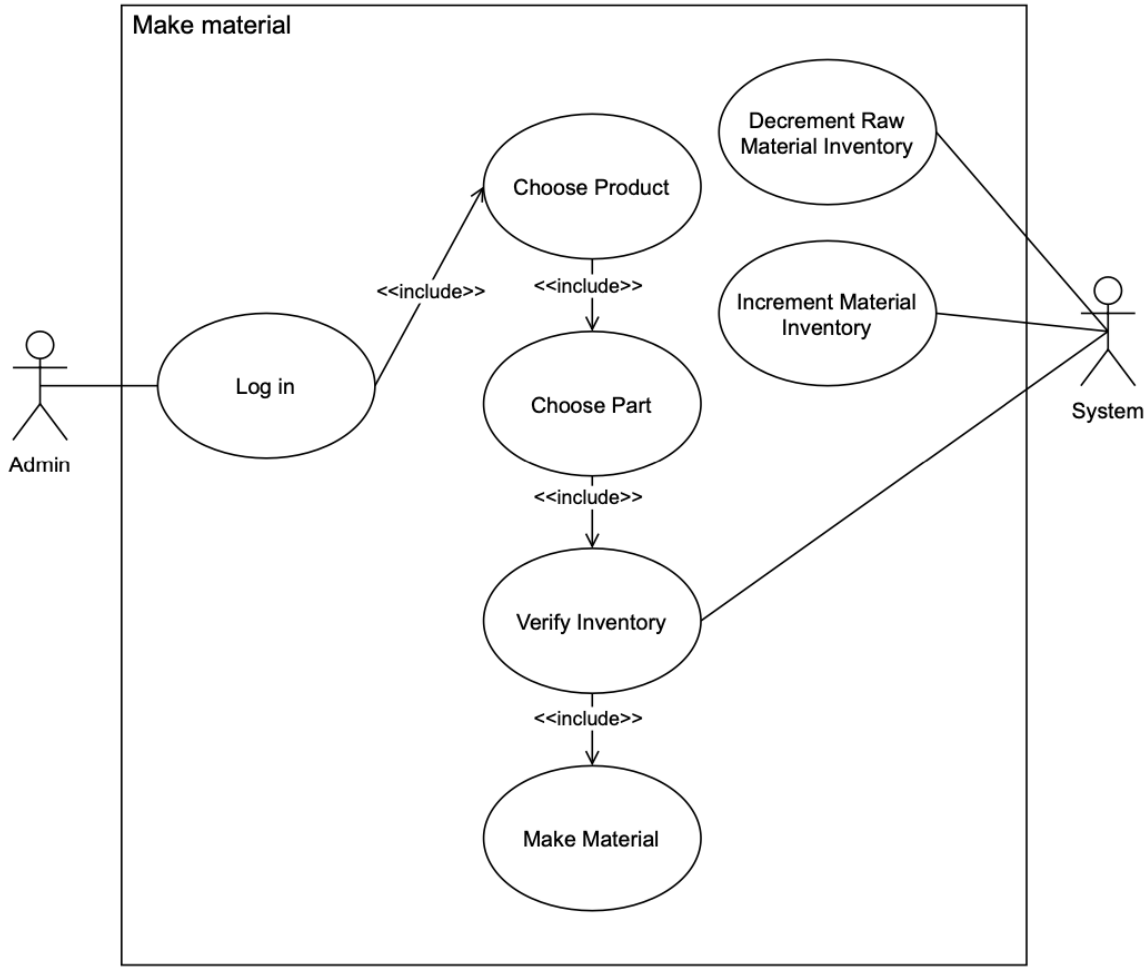

Figure 19. Use case diagram of Make material.

5.5.8 Make parts

- **Log in:** user needs to be logged in as an admin to be given the privilege to make parts.
- **Choose product:** pick a product given the existing product listed in the Production tab of the main page of the ERP system.
- **Verify inventory:** the system will show the number of parts that goes into making the bike in the inventory column.
- **Make part**: user may make the part if enough material is in inventory. This will increase the inventory of the specific part, and decrement all the materials that go into making the part.

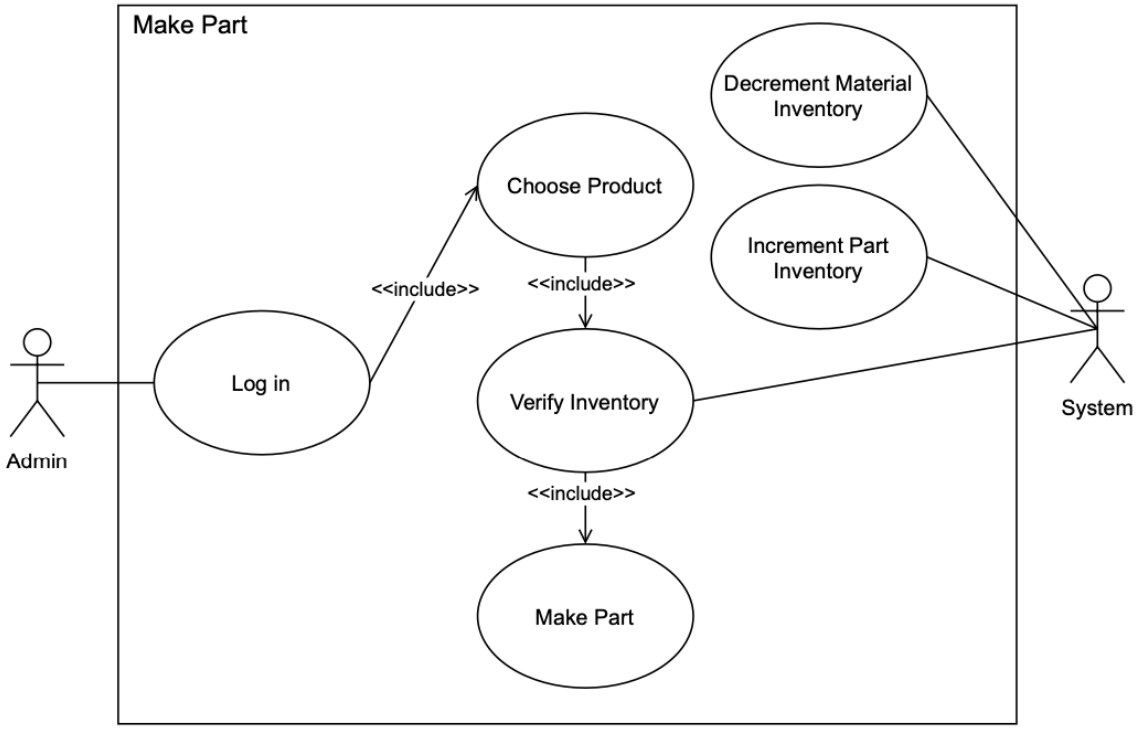

Figure 20. Use case diagram of Make Part.

5.5.9 Make bike

- **Log in:** user needs to be logged in as an admin to be given the privilege to make bikes.
- **Choose product:** pick a product given the existing product listed in the Production tab of the main page of the ERP system.
- **Verify inventory:** the system will show the number of parts that goes into making the bike in the inventory column.
- **Make bike**: user may make the bike if enough parts are in inventory. This will increase the inventory of the specific bike, and decrement all the parts that go into making the bike.

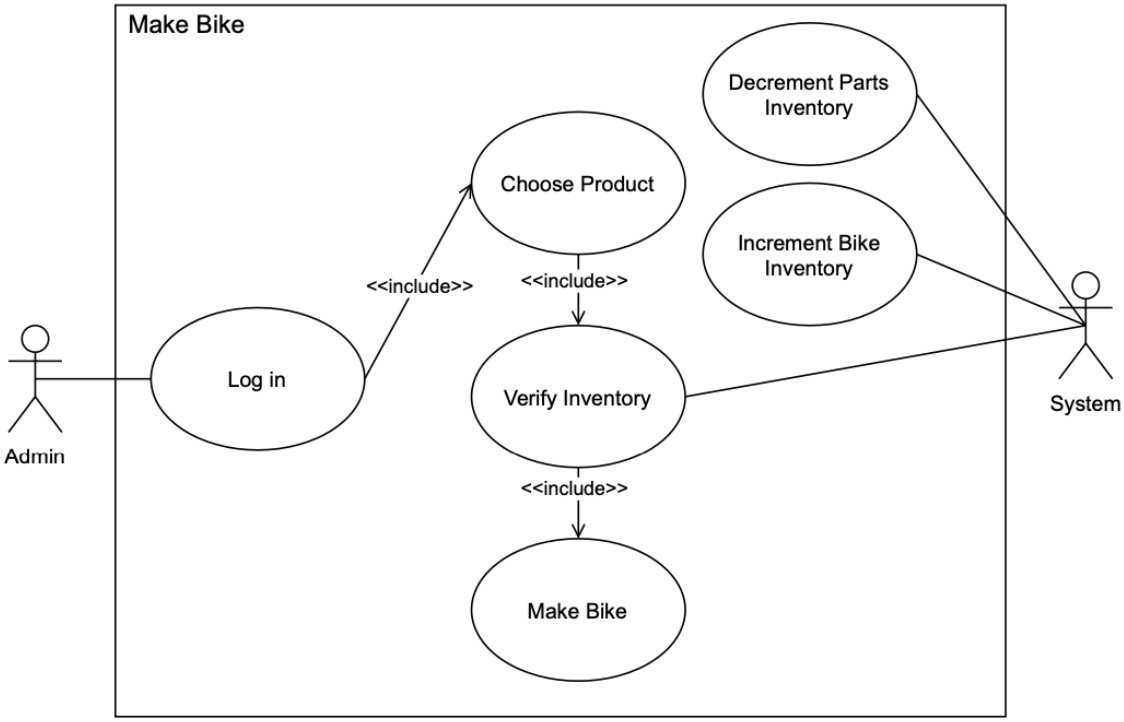

Figure 21. Use case diagram of Make Bike.

5.5.10 Viewing inventory

- **View product/material/raw material inventory and location:** user may view the number of each product, part, material, raw material along with their location in the production center.
- **Display product inventory:** the ERP system will display the product inventory and its location.
- **Display material inventory:** the ERP system will display the materials inventory and its location.
- **Display raw inventory:** the ERP system will display the raw materials inventory and its location.

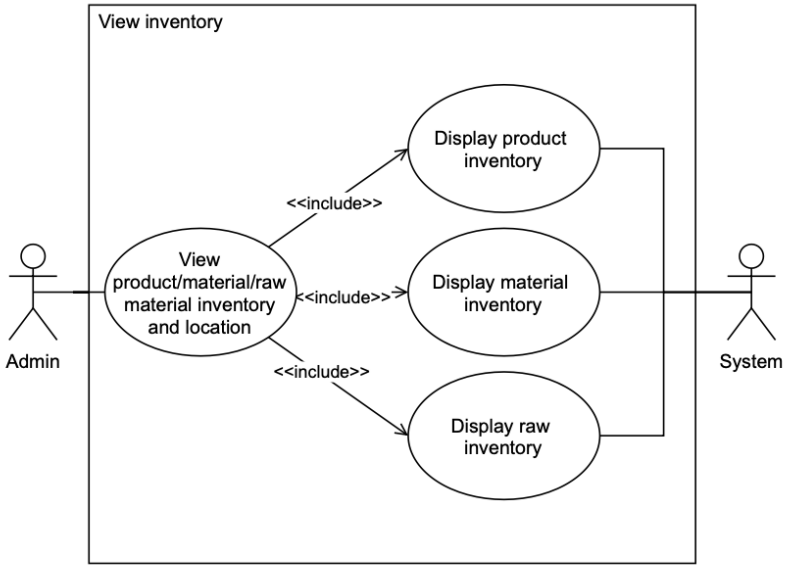

Figure 22. Use case diagram of View inventory.

The following figure is a high level sketch that demonstrates that each table will show the required number of elements needed to create its parent object.

Item → Mountain bike <requirement 1 of each> [Make Button]

Parts	Inventory	See material
1) Saddle	100	>>
2) Frame	160	>>
3) Wheels	67	>>
4) Front set	99	>>
5) Pedal Unit	277	>>

*if make Button clicked → display if bike was made successfully, but to add machinery logic too

Part → Front set <requirement 1 of each> [Make Button]

Material	Inventory	See raw
1) Handlebar grip	99	>>
2) Head tube	32	>>
3) Shock absorber	44	>>
4) Front brakes	99	>>
5) Fork	222	>>

Material → Handlebar grip <requirement 20g alloy steel, 5g rubber, 4 screws> [Make Button]

Raw	Inventory	Order from vendor
1) Stainless alloy steel	55	Vendor1 + - [Change vendor]
2) Rubber	23	Vendor2 + - [Change vendor]
3) Screws	96	Vendor3 + - [Change vendor]

Figure 23. High level sketch of element.

6.0 Architecture Decisions and Rationales

Table 2. Architecture decision 1 - Overall architecture

Related Requirement	The system shall be mobile friendly web.
Alternative	Mobile Application(iOS and android), Desktop application(JavaFx, C# etc)
Constraints	Online web application, rely on the internet to communicate between client and server.
Assumption	All devices have the internet access and installed browser.
Architecture Decision	
Identifier	#1
Description	ERP solution will be a web application.
Rationale	● The client does not have to install application ● Responsive web page access for all devices

Table 3. Architecture decision 2- MVC design pattern, front-end language and back-end language

Related Requirement	The system shall follow the MVC design pattern.
Alternative	MVVM(Model view view model), MVP(Model View Presenter)
Constraints	The developer must be familiar with MVC design patterns. The developers have basic knowledge about java and javascript.
Assumption	All browsers are javascript enabled and could run ECMAScript 5 or higher.
Architecture Decision	
Identifier	#2
Description	ERP solution will separate front-end from back-end. By using asynchronous Javascript and JSON to exchange the data. ERP would use Java(Spring boot) and Javascript(React).
Rationale	● MVC would make an independent between front end and back end. Therefore, changing or update one side does not affect another side ● SpringBoot is chosen for the back-end. ● Restful API would be used for the controller. ● React is implemented for the view. ● Some models would be represented by a simple java class some would be used as java beans.

Table 4. Architecture decision 3- MySQL database and JPA

Related Requirement	The system must allow defining, ordering, and tracking raw material and final products.
Alternative	NoSQL, JDBC(Java Database Connectivity) and POJO(Plain old java object)
Constraints	The developers have basic knowledge about databases and some web design patterns..
Assumption	The developers have installed needed software like MySQL. There will not be a lot of data(Gigabytes)
Architecture Decision	
Identifier	#3
Description	ERP solution will use MySQL database to store the data, JPA for data mapping and table data pattern for CRUD repository.
Rationale	● MySQL, which is a relational database, helps to access the data faster and efficiently. ● JPA follows the data mapping pattern of enterprise application architecture. JPA helps mapping the java class to a table in the database. ● Table data patterns are used to separate the query to the database from the resource or data itself.

Table 5. Architecture decision 4- Spring Security and JWT

Related Requirement	Sensitive data must be encrypted (credential details, personal information)
Alternative	MD5(message-digest), RSA(Rivest–Shamir–Adleman)
Constraints	Hash function encryption is one way, there is no way to decrypt. The spring security framework has to be used to confirm the encryption.
Assumption	The developer has some basic knowledge about cryptography.
Architecture Decision	
Identifier	#4
Description	ERP solution will use Spring security for encryption the password and authentication.
Rationale	● Spring security using hashing and salting to encrypt the password. Hashing is one way function. Therefore storing the encryption password in the database is secure.

Table 6. Architecture decision 5 - Testing

Related Requirement	The system shall follow the MVC design pattern and at least 50% test coverage for Controllers classes.
Alternative	TestNG , NUnit , openClover
Constraints	Testing could only identify if there is a present of the bug in the software but it cannot prevent the bug.
Assumption	The developer has basic knowledge about testing.
Architecture Decision	
Identifier	#5
Description	ERP solution would use Junit testing and Jacoco.
Rationale	● Junit would be unit testing for the software. ● Jacoco would be used for test coverage of line, method, class etc. ● Could perform Spring testing if it is necessary.

Table 7. Architecture decision 6- SMTP

Related Requirement	The system sends e-mails to certain roles
Alternative	N/A
Constraints	The developer has basic knowledge about electronic mail and user agents.
Assumption	The email that is stored in the database is a valid email.
Architecture Decision	
Identifier	#6
Description	ERP solution will use SMTP(Simple Mail Transfer Protocol) to send email.
Rationale	● With help of the spring library, ERP could deliver the message to the receiver server.

Appendix I Class diagram

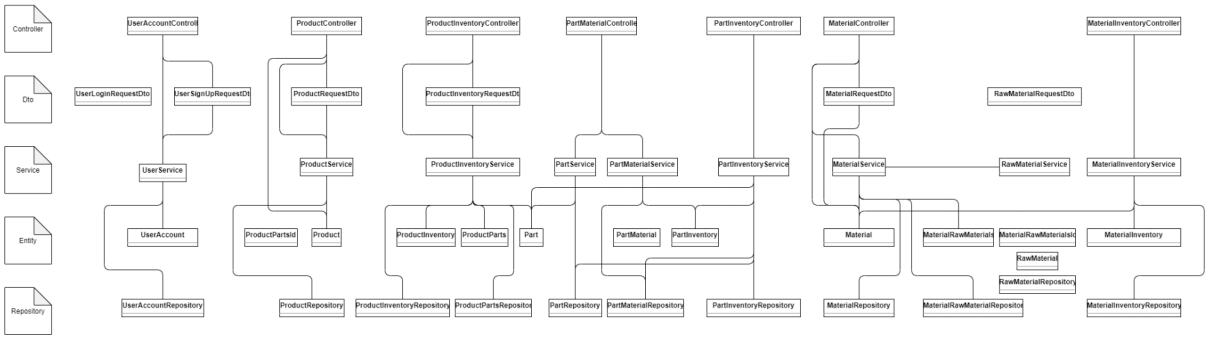

Figure 24. Class diagram of current release.

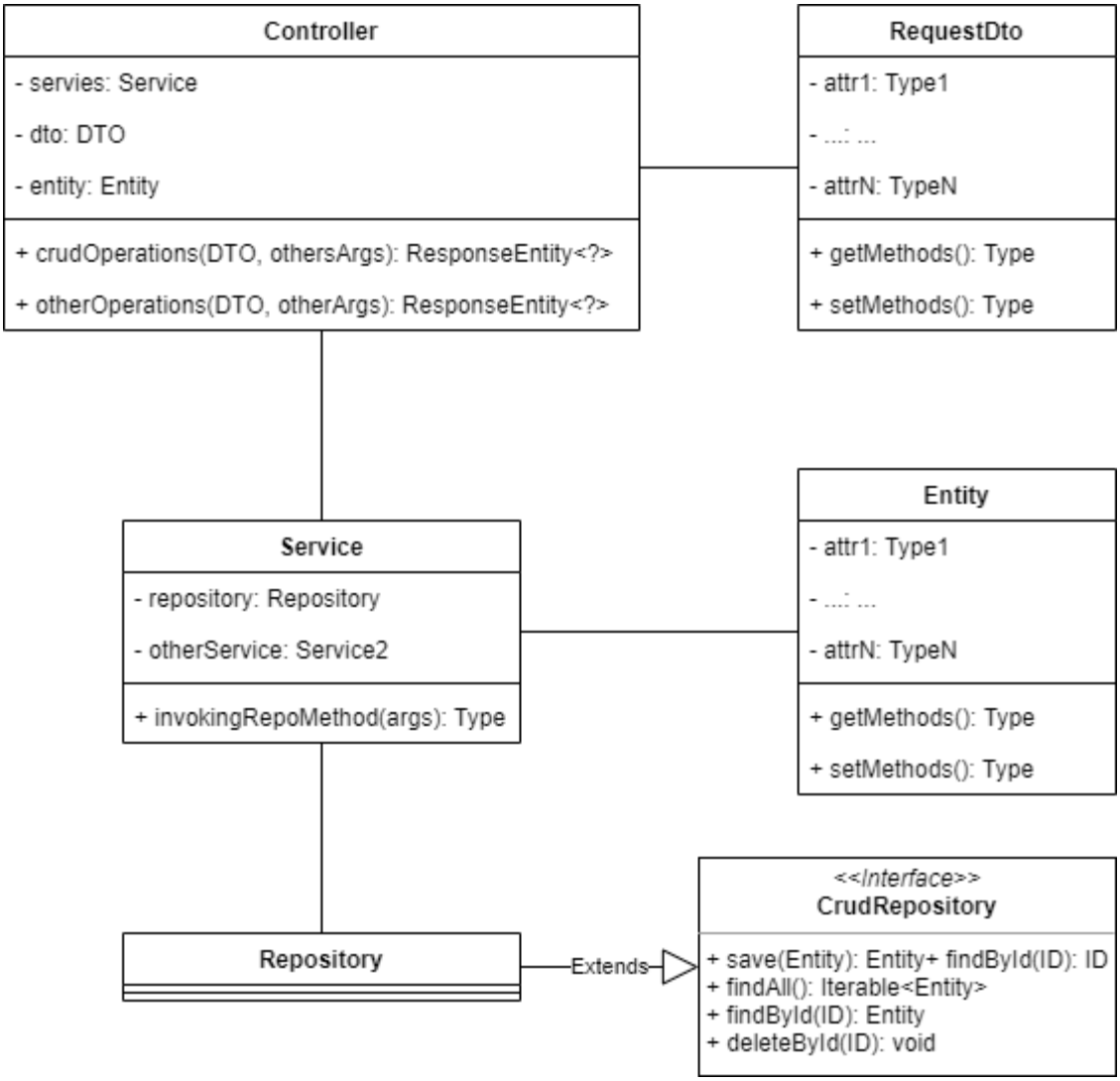

Figure 25. Class diagram for overall architecture.