System Requirements Specification

Project : OptiSync

Document: Software Requirement Specification

Doc. Ref. : SRS_v0.1.doc

Version : 0.1

Status : Drafted

Created by : AbhishekPatil,AbhinavDubey,KaranThakare,VenkateshwaraSutar,ShirishGaikwad

Date :25-August-2023

1. Introduction

The OPTISYNC project web application designed for managing everyday tasks in a manufacturing setup. It's a project aimed at simplifying operations using technologies like Spring Boot, React JS, and MySQL.

The project is centred around different roles, like managers, vendors, production teams, and more. It uses Spring Boot and React JS to create easy-to-use interfaces that help users navigate through tasks without any hassle.

For instance, the registration module helps users sign up into the application. The admin module, powered by Spring Boot and React JS, looks after essential tasks like logging in, handling passwords, managing requests, creating users, and more. It's all about keeping things secure and organized.

The setup module ensures everything is properly configured, while the login module makes it simple for authorized users to access the system. Managers find the manager module really handy; they can start new orders, keep an eye on orders, manage quantities, and give the go-ahead for password recovery. This module even handles final approvals for dispatching orders.

The store module is all about managing raw materials. It helps with updating material lists, ordering materials, tracking their status, and keeping tabs on quantities. With Spring Boot and React JS, information flows between departments like production, assembly, and vendors.

Vendors also have their space in the project. The vendor module helps in communication with stores, making sure materials are received and data is updated correctly. The production module, with Spring Boot and React JS, deals with material entries, daily task updates, and making sure the right parts are being produced.

When it comes to assembling and quality control, the assembly/quality module runs with similar function as of Production. It helps manage new parts, keep tasks on track, and ensure product quality. And finally, the dispatch module, powered by Spring Boot and React JS, manages new products, tracks tasks, and submits final drafts to managers. It's also in charge of making sure the products are dispatched as planned.

2. Functional Requirements

The Account part of OptiSync System has three modules which are divided 13 processes described as below.

No	BRS requirement	Description
2.1	**Registration**	
		Companies complete the registration process with required fields.
2.2	**Admin Module**	
		Login
		Accept new request
		Update process status
		Create new user
		Password recovery
		Check status of user
2.3	**Setup**	
		Sets Products
		Sets Part
		Sets Raw Material
		Sets Machine List
		Sets Critical Path Method
2.4	**Login Module**	
		Login Process
		Forget Password Process
2.5	**Manager Module**	
		Setup

		Registration
		Push New Order Process
		Status of Order Process
		Approval for Password Recovery Process
2.6	**Store Module**	
		Ordering Raw Material
		Check Status of Raw Material
		Forward material to Production/dispatch/assembly
2.7	**Production Module**	
		Receives material from Storage
		Update Daily Task Process
		Forward Part Quantity Process
2.8	**Assembly/Quality Module**	
		Receives Part from Production
		Update Daily Task Process
		Forward Product Quantity Process
2.9	**Dispatch Module**	
		New Products Entry Process
		Update Daily Task Process
		Final Dispatch Process

2.1 Registration Module

- **Company ID:** Each company in the system is assigned a unique identification number. This helps keep track of different companies using the system.
- **Company Name:** This is where the name of the company is stored. It helps identify the company easily.
- **Person Name:** The name of the person responsible for the company's registration is stored here.
- **Contact Number:** The contact number of the person or company is saved in this field.
- **Email:** The email address associated with the company is stored here.
- **Registration Date:** This field captures the date when the registration is made. It's automatically set to the current date.
- **Plan ID:** Each company selects a plan for using the system. This ID links to the chosen plan.
- **Payment Status:** This field indicates whether payment has been made. It's set to '0' initially and changes to '1' after payment.
- **Plan Start Date:** The start date of the chosen plan is stored here.
- **Plan End Date:** The end date of the selected plan is stored in this field.
- **Admin Approval:** This field represents whether the registration has been approved by the admin. It's set to '0' initially and changes to '1' after approval.
- **Mode of Transaction:** The method of transaction, such as "Online," is saved here. This indicates how the payment was made.

Constraints in Database:

- **company_id:** Auto-incremented unique identifier for each company.
- **company_name:** Name of the company, required.
- **person_name:** Name of the person registering the company, required.
- **contact_no:** Contact number of the person, required.
- **email:** Email address of the person, required.
- **registration_date:** Date of registration, auto-generated to the current date.
- **plan_id:** ID of the selected plan, linked to the plans table, required.
- **payment_status:** Payment status, default set to 'Not Paid' (0).
- **plan_startdate:** Start date of the selected plan, optional.
- **plan_enddate:** End date of the selected plan, optional.
- **admin_approval:** Indicates admin approval status, initially set to 'Not Approved' (0).
- **mode_of_transaction:** Method of transaction, default set to 'Online'.

2.2 Admin Module

Admins play a crucial role in the OPTISYNC project, ensuring smooth operations and maintaining data accuracy. Their functionalities are focused on managing company-related information, user access, and overseeing various processes.

1. **Login**: Admin can log in using their username and password. The login credentials are stored in the **login** table.

   Constraints and References:

   - **username** should be unique.
   - **company_id** references the **company** table.
   - **role_id** references a table defining **roles**tables.

2. **Accept New Request**: Admin can review and accept new requests for company registration. The company details are stored in the **company** table.
   Constraints and References:

   - **plan_id** references a **plan** table
   - **company_id** is used as a foreign key in other tables.

3. **Update Process Status**: Admin can update the status of the pending processes, potentially related to different tables like **login** or **company**.

4. **Create New User**: Admin can create new users with different roles to access the system.
   Constraints and References:

   - **username** should be unique.
   - **company_id** references the **company** table.
   - **role_id** references a table defining **roles**

5. **Password Recovery**: Admin can initiate password recovery processes for users. The Manager is has the gate way when it comes to password recovery as before getting to the admin for password Manager request from its company members eg. If Store user wants password to be recovered first that company's manager will be notified.

6. **Check Status of User**: Admins can check the status of users with their package expiration status.

2.3 Setup Module

1. **Sets Products:** This function allows the manager to set up products for their company. A product can have a name, description, and is associated with a company.
   Constraints and References:
   - **product_name should be unique within a company.**
   - **company_id references the company table.**

2. **Sets Part:** This function enables the manager to define parts for the products. A part can have a name, description, and is linked to a product.
   Constraints and References:
   - **part_name should be unique within a product.**
   - **product_id references the product table.**

3. **Sets Raw Material:** This function lets the manager establish raw materials required for parts. A raw material can have a name, description, and is associated with a part.
   Constraints and References:
   - **name can be unique within a part.**
   - **part_id references the part table.**

4. **Sets Machine List:** This function allows the manager to create a list of machines. A machine can have a name, description, and is linked to a company.
   Constraints and References:
   - **machine_name should be unique within a company.**
   - **company_id references the company table.**

5. **Sets Critical Path Method (CPM):** This function lets the manager set up critical path method information. CPM includes various time intervals for different stages of production and dispatch. It is linked to a company, product, and part.
   Constraints and References:
   - **company_id references the company table.**
   - **product_id references the product table.**
   - **part_id references the part table.**

2.4 Login Module

- **Login Process:** This functionality allows users to log into the OPTISYNC system using their username and password.
- **Forget Password Process:** Users who have forgotten their password can initiate a password recovery process.
- **login_id**: Unique identification for each login entry.
- **username**: The username associated with the login.
- **password**: The password associated with the login.
- **company_id**: Links the login to a specific company's data.
- **role_id**: Associates the login with a specific role in the system.
- **forgetpass_status**: Indicates whether the forget password process has been initiated.
- **setup_status**: Reflects whether the setup process has been completed for the user.

Constraints and Considerations:

- The **company_id** associates the login with a particular company's data, ensuring data isolation between companies.
- The **role_id** links the login to a specific role, determining the user's permissions and access level.
- The **forgetpass_status** and **setup_status** flags help in tracking user actions and system setup status.

2.5 Manager Module

1. **Login:** Managers can log in using their username and password. The login credentials are stored in the login table.

   Constraints and References:

   - **username** should be unique.
   - **company_id** references the **company table.**
   - **role_id** references a table defining **roles** .

2. **Registration:** Here the manager is the entity which is registers on this web application . Refer Registration Module 2.1.

3. **Setup:** Managers can set up various components including products, parts, raw materials, machines, etc. Manager add all necessary data in the setup so that all the data will be available to all the roles purchased in that package. Refer Setup Module 2.3.

4. **Push New Order Process:**Managers can initiate a new order process by providing details such as the product, part, order quantity, etc.

   Constraints and References:

   - **order_id** should be unique.
   - **product_id** references the **product table.**
   - **part_id** references the **part table.**

5. **Status of Order Process:**Managers can check the status of ongoing order processes, including the achieved quantity, targeted value, dispatch status, etc.

   Constraints and References:

   - Depends on the specific **part_id**status being tracked and the **parts** tables.
   - Depends on the specific **product_id**status being tracked from **products** table.

6. **Approval for Password Recovery Process:**

   Managers can approve requests for password recovery, typically involving updates to the login table.

   Constraints and References:

   - Depends on the password recovery from particular user with **role_id** role's table belong to a company being implemented.

2.6 Store Module

1. **Login**: Store personnel can log in using their username and password. The login credentials are stored in the **login** table.
   Constraints and References:

   - **username** should be unique.
   - **company_id** references the **company** table.
   - **role_id** references a table defining **roles**.

2. **Ordering Raw Material**: Store personnel can initiate orders for raw materials required for production. The orders are stored in the **orders** table.
   Constraints and References:

   - **order_id** should be unique.
   - **client_id** references a **client** table.
   - **company_id** references the **company** table.
   - **product_id** references the **product** table.
3. **Check Status of Raw Material**: Store personnel check the current stock status of raw materials based on specific identifiers such as raw material ID, part ID, product ID, and order ID.
   Constraints and References:

   - The store personnel query the **raw_material** table to retrieve the stock information.
   - The identifiers (**raw_material_id**, **part_id**, **product_id**, and **order_id**) are used as references in the query to fetch the specific raw material stock details.

4. **Forward Material to Production/Dispatch/Assembly**: Store personnel can forward raw materials to various stages like production, vendor, dispatch, or assembly.
   Constraints and References:

   - **order_id** references the **orders** table.
   - **raw_material_id** references the **raw_materials** table.

2.7 Production Module

1. **Login:**Production personnel can log in to the system using their credentials to access the production module functionalities.
   Constraints and References:
   - **username** should be unique.
   - **company_id** references the **company** table.
   - **role_id** references a table defining **roles**.

2. **Receives Material from Store:**The production team receives raw materials from the store, which were previously forwarded from the store module.
   Constraints and References:
   - The**raw_materials**table provides information about raw materials, including**raw_material_id, name, description, and part_id.**

3. **Update Daily Task Process:** The production team updates the status and progress of daily tasks, ensuring transparency and effective communication within the production process.

Constraints and References:

- **machine_id** from **machine table**
- **order_id** from **order_table**.
- **Raw_material_id** from **raw_materials table**

4. **Forward Part Quantity Process:** The production team forwards the quantity of finished parts to subsequent modules, such as assembly and quality, ensuring proper distribution and coordination.

   Constraints and References:

   - **part_id** from **parts table**
   - **order_id** from **order table**

2.8 Assembly & Quality Module

1. **Receives Parts Process:** This process involves entering information about new parts that need to be assembled. It includes details such as the date, order ID, part ID, received part quantity, targeted value, and achievement status.
2. **Login:** Production personnel can log in to the system using their credentials to access the production module functionalities.
3. **Update Daily Task Process:** This process is about updating the daily tasks related to assembly and quality checks. It might involve marking tasks as completed, recording progress, or noting any issues.
4. **Forward Product Quantity Process:** This process entails forwarding the quantity of assembled products that have passed the quality checks. It's a crucial step toward managing inventory and distribution.

Constraints and Considerations:

- **Username:** should be unique.
- **company_id:** references the **company** table.
- **role_id** references a table defining **roles**.
- **assembly_entry_no**: This field serves as the primary key, ensuring each entry has a unique identification number.
- **date**: Represents the date of the assembly entry, providing a timeline reference.
- **order_id**: Refers to the order associated with the assembly, connecting the entry to a specific order.
- **part_id**: Points to the part being assembled, linking the entry to a particular part.
- **received_part_qty**: Indicates the quantity of parts received for assembly, aiding in tracking materials.
- **targeted_value**: Denotes the expected quantity of parts to be assembled, offering a benchmark for comparison.
- **achieve**: Represents the actual quantity achieved through assembly, providing insight into performance

2.9 Dispatch

1. New Products Entry Process: This process involves entering details about new products that are ready to be dispatched. It includes information such as the product name, description, and quantity.

1. **Login:** Dispatch personnel can log in to the system using their credentials to access the production module functionalities.

2. **Update Daily Task Process:** This process focuses on updating daily tasks related to product dispatch. This might involve tasks like preparing products for shipment, verifying quantities, and ensuring proper packaging.

3. **Final Dispatch Process:** This process includes the actual dispatch of products to clients. It ensures that the right products, quantities, and details are sent to the clients.

Constraints and Considerations:

- **username** should be unique.
- **company_id** references the **company** table.
- **role_id** references a table defining **roles**.
- **assembly_entry_no**: Primary key for the assembly entry, ensuring unique identification.
- **date**: Date of the assembly entry, providing a timeline reference.
- **order_id**: Connects the assembly entry to a specific order for tracking.
- **part_id**: Links the assembly entry to a particular part being assembled.
- **dispatch_qty**: Quantity of the product that has been dispatched.
- **company_id**: Primary key for the company table, ensuring each company has a unique identification.
- **product_id**: Primary key for the product table, ensuring each product has a unique identification.

3. Use Cases

Admin:

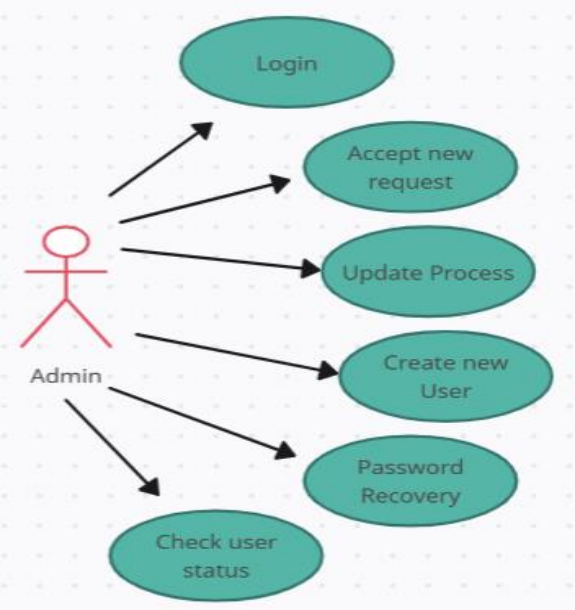

Fig. Use case diagram for admin

Setup:

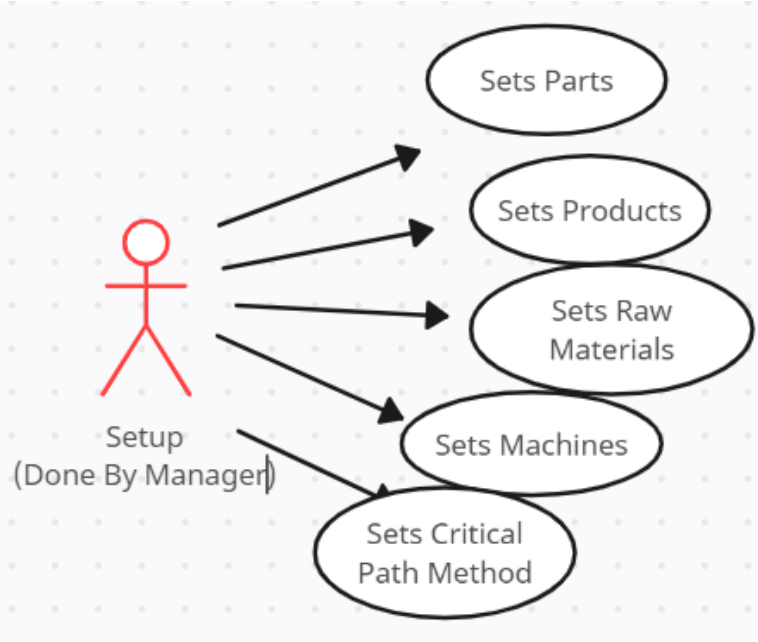

Fig. Use case diagram for setup

Manager:

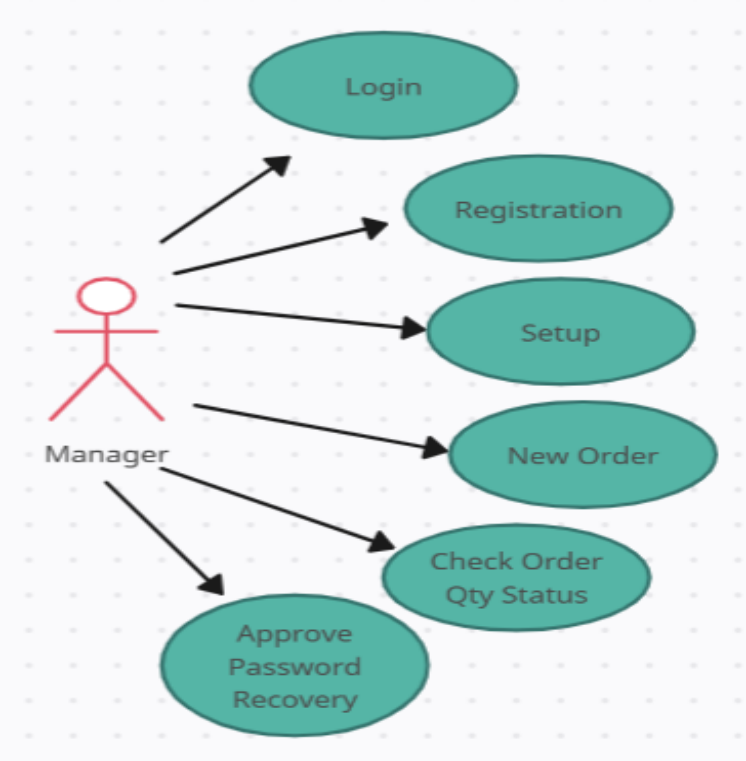

Fig. Use case diagram for Manager

Store:

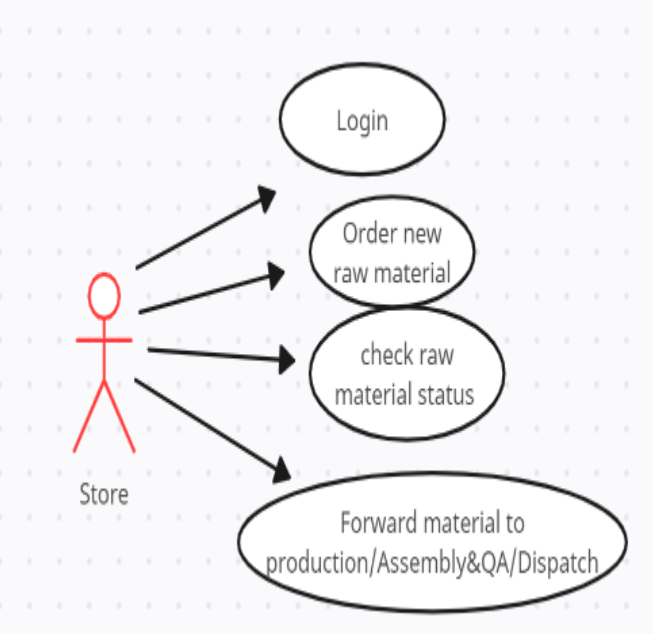

Fig. Use case diagram for Store

Production:

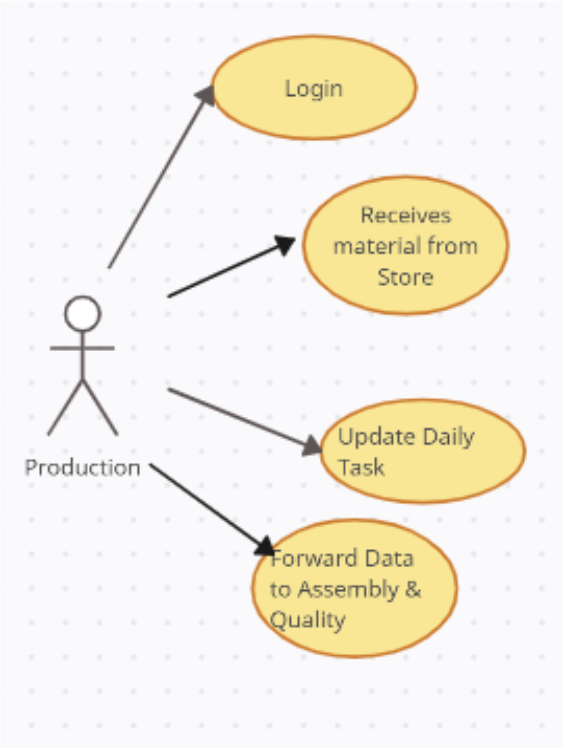

Fig. Use case diagram for Production

Assembly&Quality:

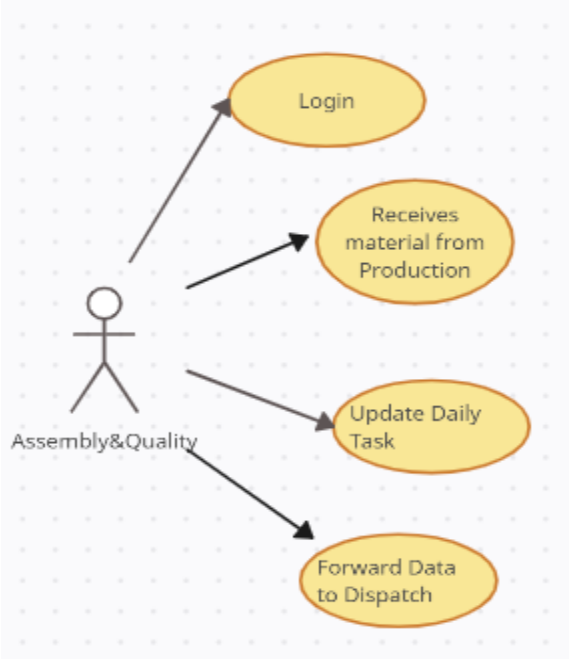

Fig. Use case diagram for Assembly&Quality

Dispatch:

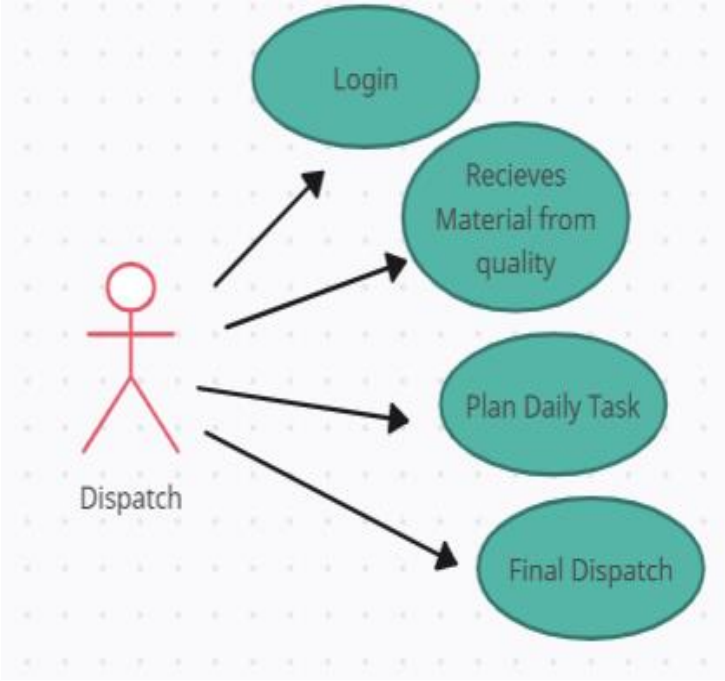

Fig. Use case diagram for Dispatch