CENG3004: Software Engineering

Stock Management System

Design Document

31.05.2023

Team Members:

Ali GÖKMEN-190709016
Gökhan MUTLU-190709056
Mamady III DIAKITE-180709719
Mehmet Reşit ÇAĞAN-190709045

DISCLAIMER

This document is produced based on Software Requirements Specification (SRS) that was delivered to and agreed by the customer. For the success of the project, it needs to be developed carefully tracing back to requirements as applicable, set design goals and address design goals thoroughly.

This page can be removed once read.

Contents

1 Overview

Provide an overview of the system. What is the system, what does it do, why do you do it, who is the customer?

Why do you do it?

We choose it because some complexity issue and connectionless between customer, stock, store, product and bill activities.

What is the system and what does it do?

It ensures the presence of products, customers and stocks in stores. Thus, it aims to reduce complexity in corporate companies.

- Handling flows between the activity of Customer, Stock, Store, Product and Bill.

- We have an admin that can search a specific customer also add, update and delete - customer. Admin can make changes on the bills like adding, editting and updating.

- User can request report of stock, product and store.

Who is the customer?

The end user of the stock management system who interacts with the system to place orders or check product availability.

2 Design Goals

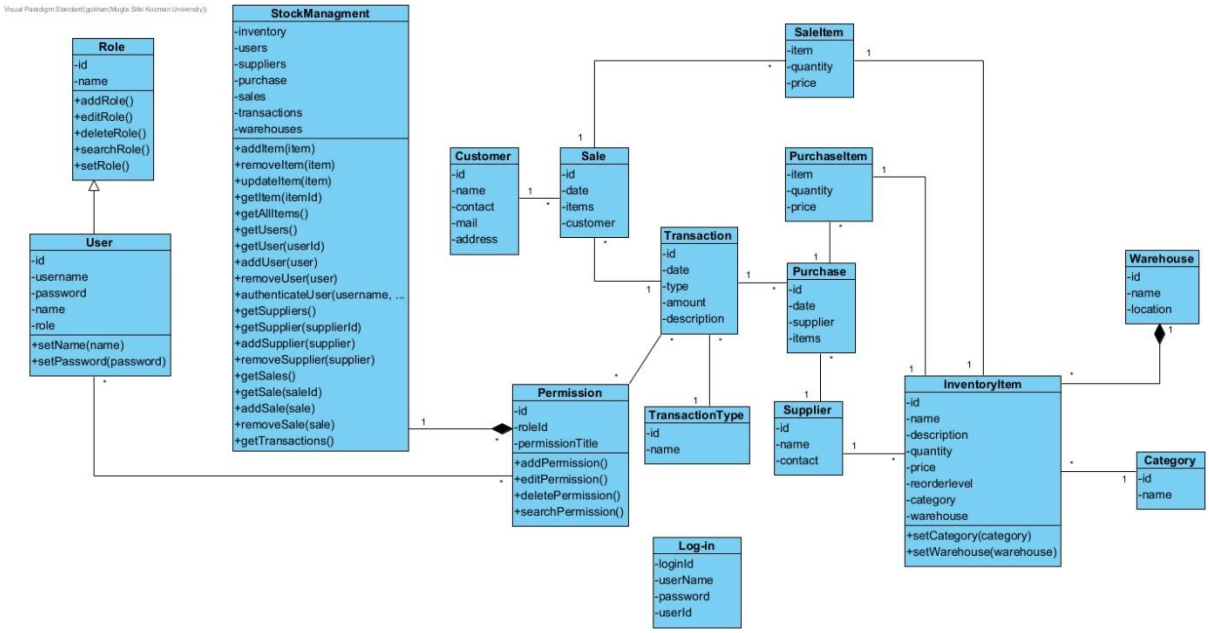
Goal's Concern[1]	**Related Requirement Identifier**[2]	**Description**
Performance	NR-1	The system should respond in 2 seconds to user requests.
Scalability	NR-2	The system should be able to handle increasing amountsda data such as handling 10.000 of inventory items in 3 seconds.

[1] Examples: Reliability, Modifiability, Maintainability, Understandability, Adaptability, Reusability, Efficiency, Portability, Traceability of Requirements, Fault Tolerance, Backward-Compatibility etc.. Check class slides for examples.
[2] The non-functional requirement identifiers from Analysis/Software Requirements Document

Usability	NR-3	The system's user interface should comply with WCAG 2.1 accessibility standards, ensuring accessibility for users with disabilities. It should also provide clear and intuitive navigation, with a maximum learning curve of 15 minutes for new users
Security	NR-4	The system should encrypt sensitive data, such as user credentials and financial transactions, using industry-standard encryption algorithms.

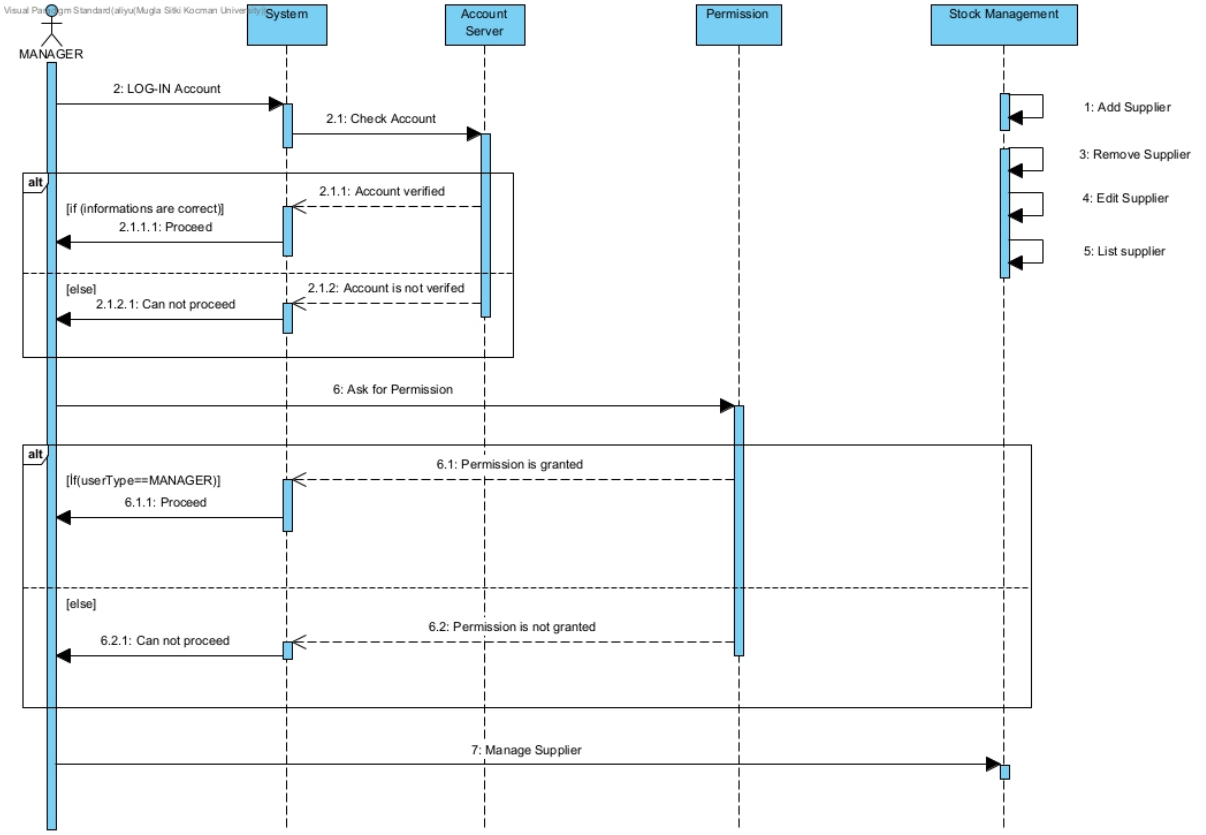
3 System Models

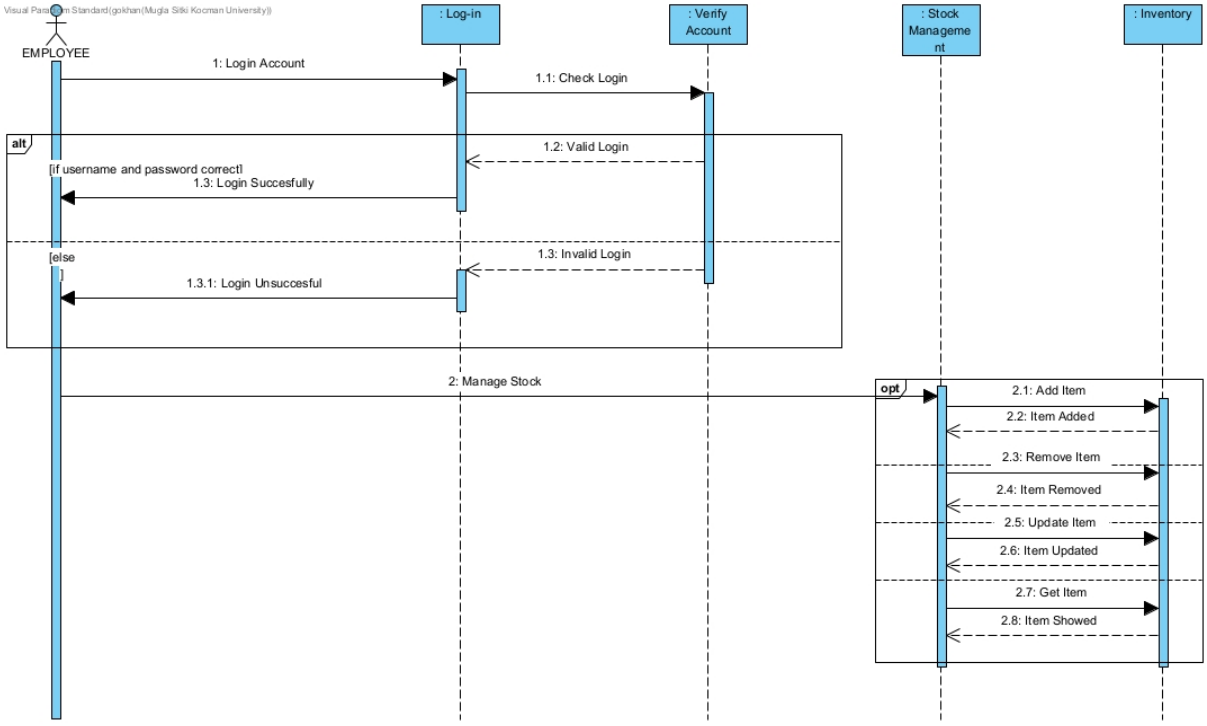
3.1 Class Diagrams

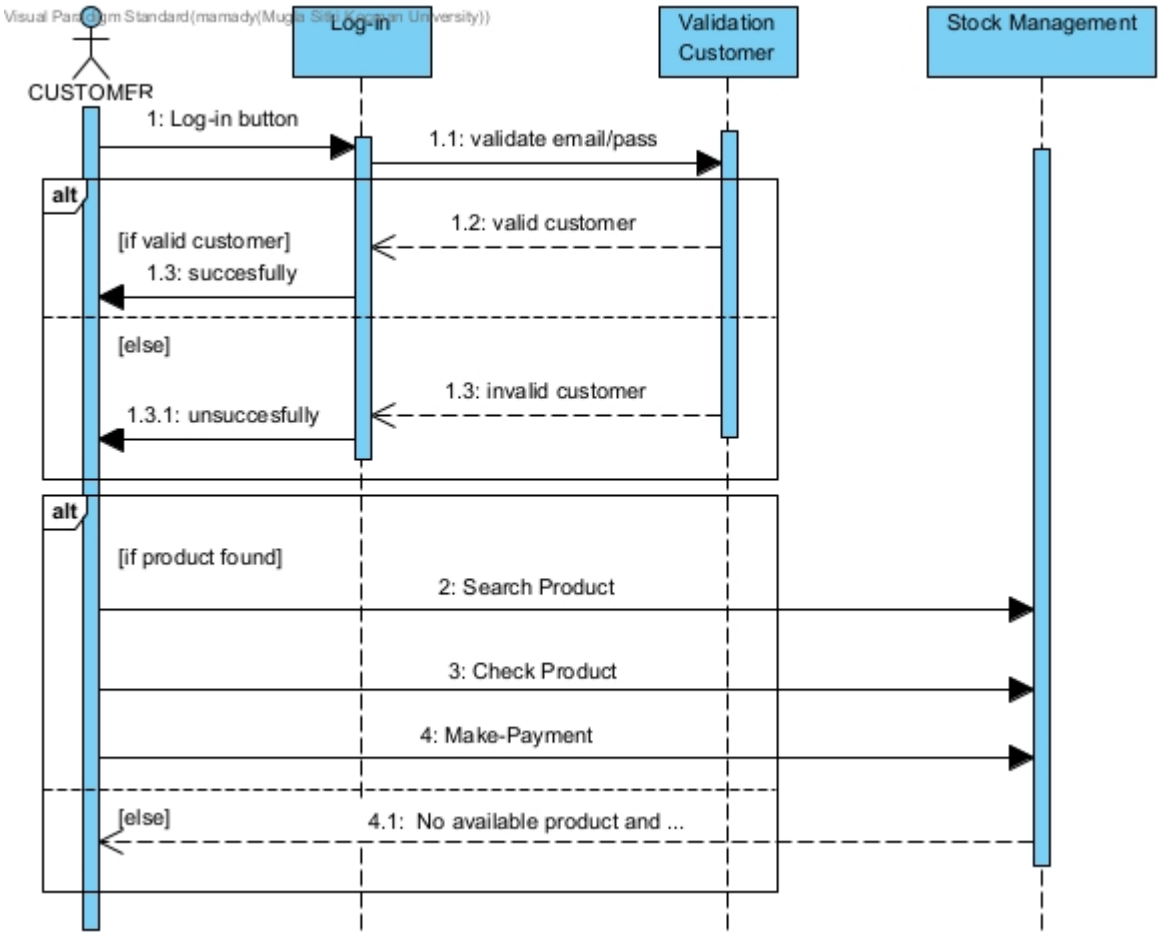


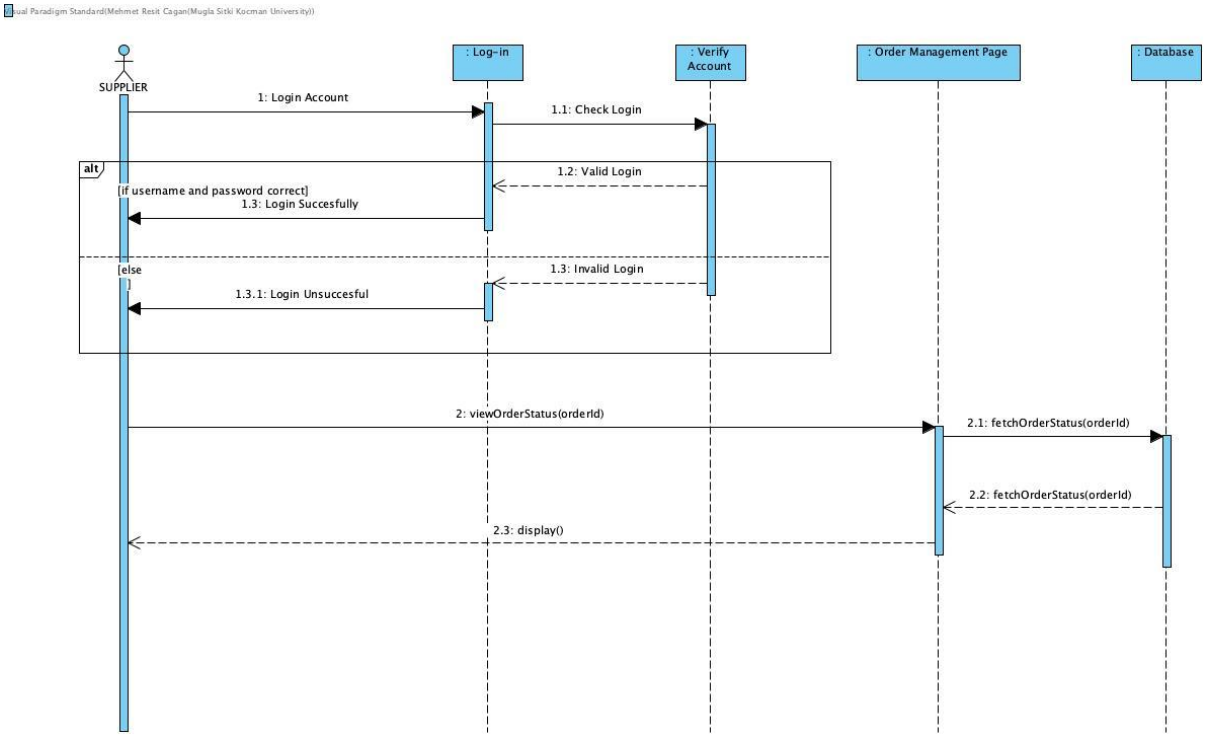
Revised/extended versions of the UML diagrams from the Analysis Document

3.2 Sequence Diagrams



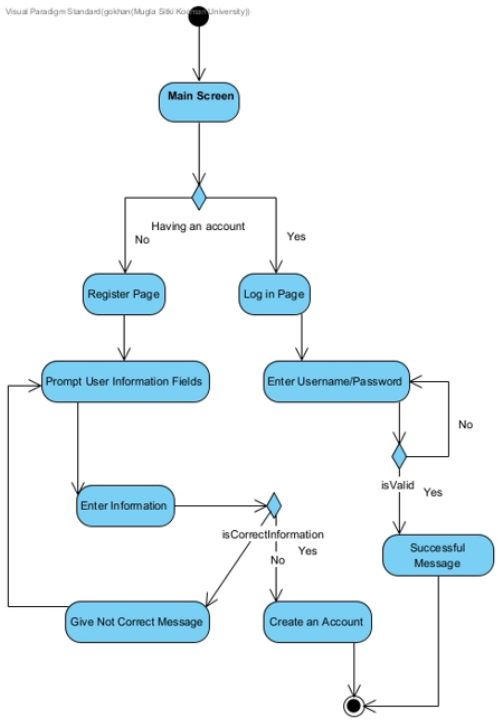


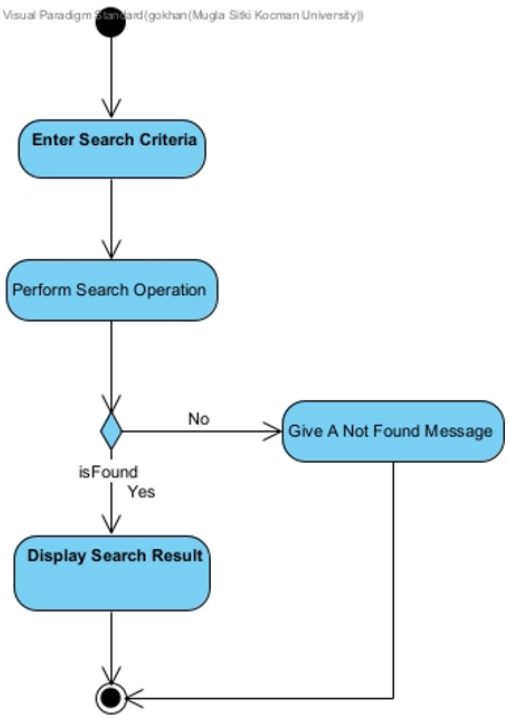


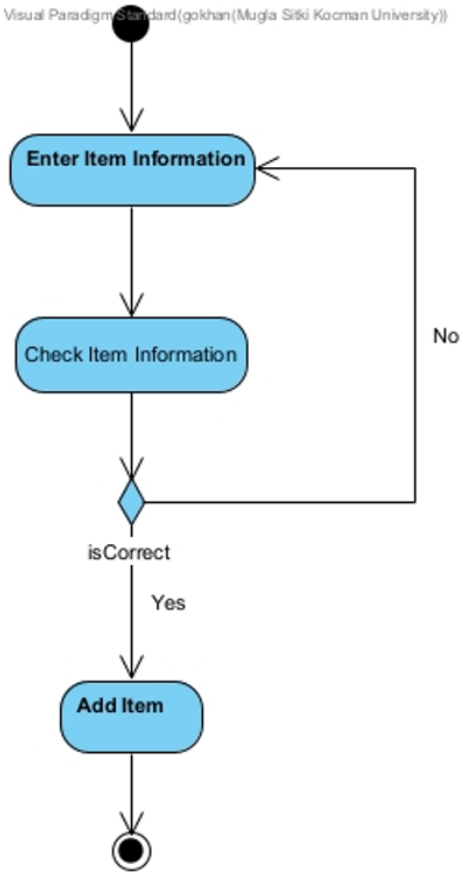


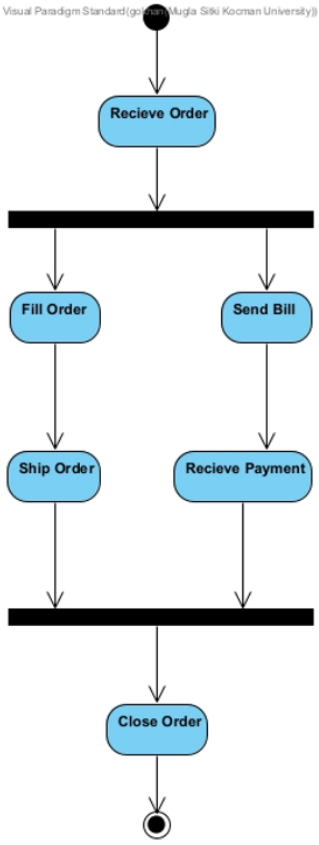
Revised versions and ,if needed, additional to and/or of UML diagrams from the Analysis Document

3.3 Activitiy Diagrams



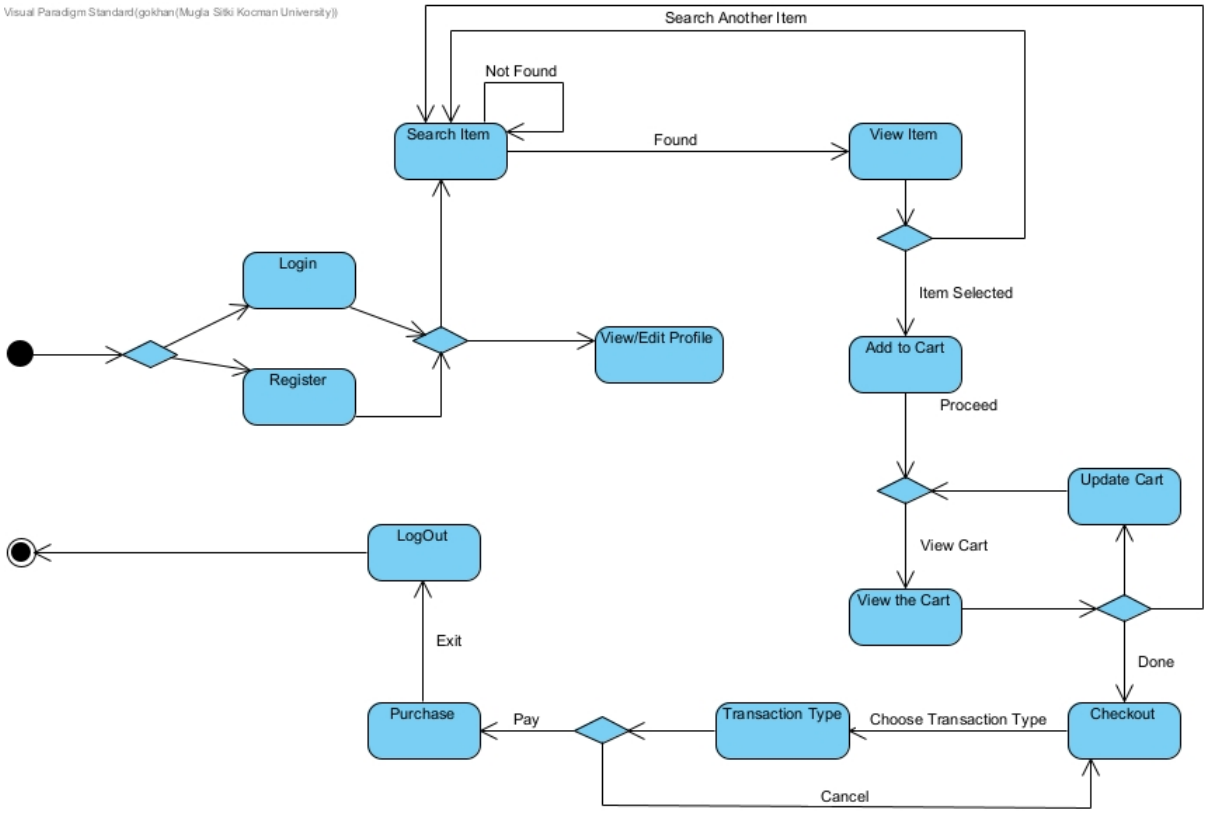






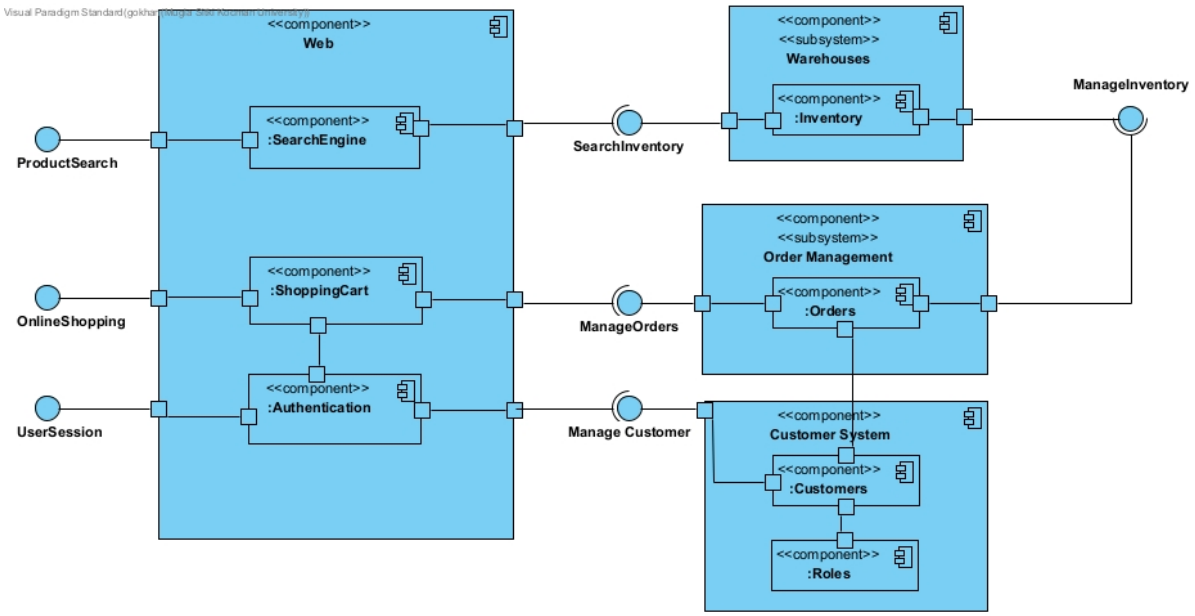
Revised versions and, if needed, additional to and/or of UML diagrams from the Analysis Document

3.4 Statechart Diagrams



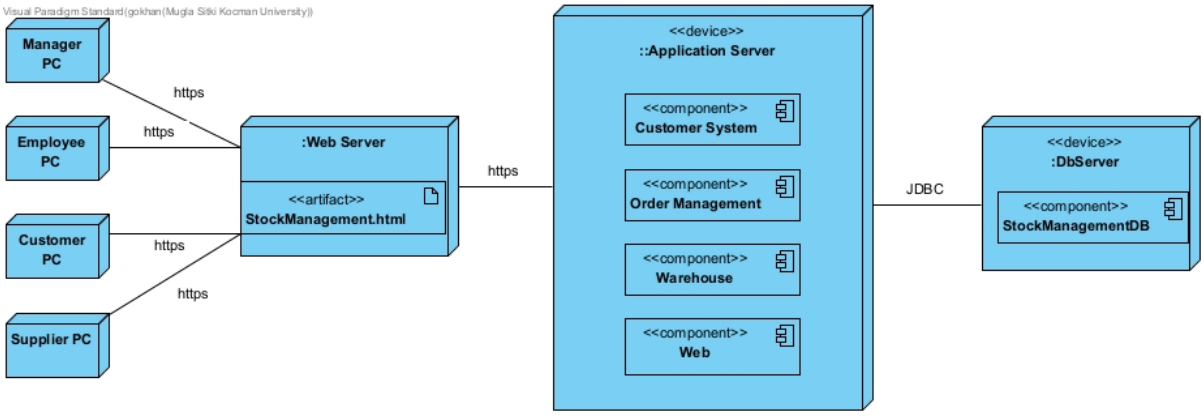
Revised versions and ,if needed, additional to and/or of UML diagrams from the Analysis Document

4 Subsystem Decomposition



Provide the UML Component Diagram and list of all classes per component.

5 Hardware / Software mapping



Provide the UML Deployment Diagram. Refer to subsystems in section 4 as needed.

6 Other Design Concerns (use relevant subsections)

6.1 Concurrency

Identify threads that simultenously. Report race conditions. Refer to Sequence Diagrams in Section 3.2 or Analysis document.

Does the system provide access to multiple users?

Yes, the stock management system is designed to handle multiple users simultaneously. Different users can perform actions such as managing inventory, making purchases, recording sales, and accessing reports concurrently.

Which entity objects of the object model can be executed independently from each other?

In the stock management system, several entity objects can be executed independently, allowing for potential concurrency such as inventory item(update, delete etc.), purchase etc.

Can a single request to the system be decomposed into multiple requests?

Yes, in certain scenarios, a single request to the system can be decomposed into multiple requests that can be executed concurrently. For example:

Concurrent sales: When multiple sales transactions occur simultaneously, each transaction can be processed independently, allowing for parallel processing.

Can these requests be handled in parallel?

Yes, the stock management system can handle concurrent requests and process them in parallel when appropriate. For instance:

Concurrent inventory updates: When multiple users or subsystems update the inventory simultaneously, the system can handle these updates concurrently to maintain data consistency and minimize contention.

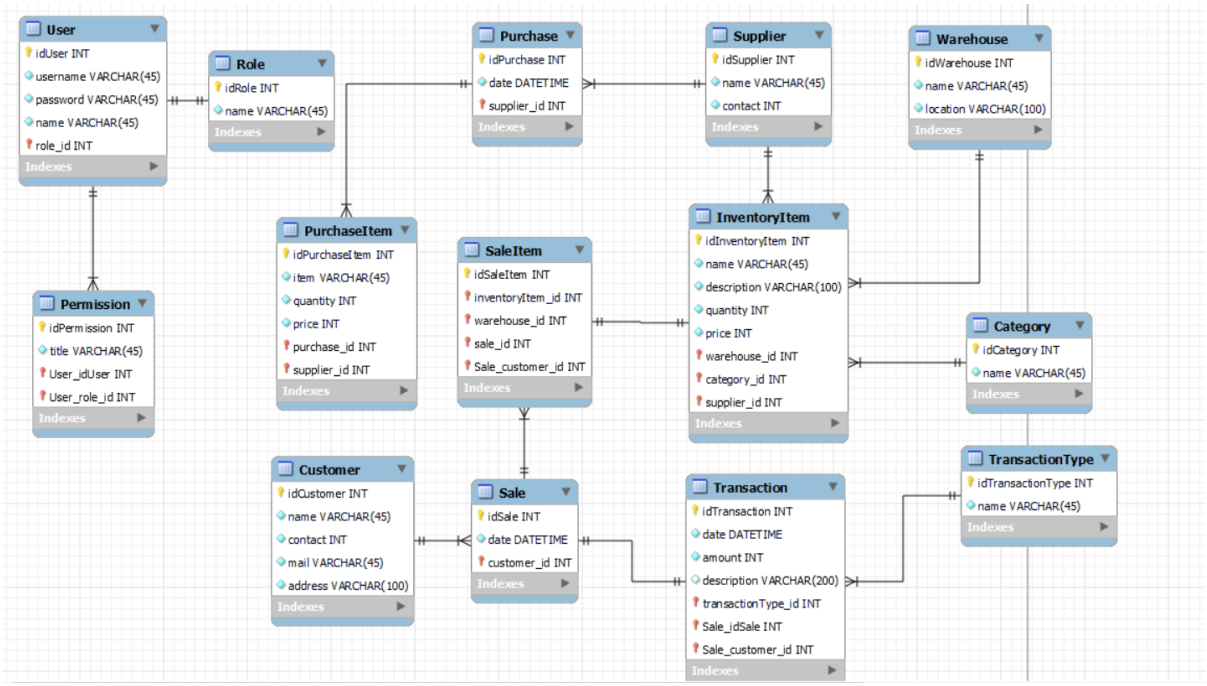
RACE CONDITION

Inventory Item Update: If multiple threads or processes try to update the same inventory item simultaneously, there may be a race condition. For example, if two employees attempt to increase the quantity of an item in the inventory at the same time, the final quantity may not be accurate due to interleaved updates.

6.2 Data Management

Identify the storage (e.g. RDBMs vs File system) you plan to use. Ideally provide an ER diagram.

We plan to use MySQL 8.0.33 for this system. it offer structured data storage, ACID (Atomicity, Consistency, Isolation, Durability) compliance, and efficient querying capabilities.



6.3 Global Resource Handling

Discuss authentication/authoritization mechanisms. Provide an “Access Matrix” using factors should correspond to actors in uses cases from Analysis Document..

Authentication Mechanisms:

Username and Password: This is a traditional method where users provide a unique username and password combination to authenticate themselves.

Authorization Mechanisms:

Role-Based Access Control: It assigns roles to users and grants permissions based on those roles. For example, roles like "Customer," "Manager," and "Employee" may have different levels of access to the stock management system.

Class/Actor	Manager	Customer	Employee
User[3]	setName(name) setPassword(password)	setName(name) setPassword(password)	setName(name) setPassword(password)
StockManagement	addItem(item) removeltem (item) updateltem(item) getItem (itemid) getAllItem() getUsers() getUser(userld) addUser(user) removeUser(user) authenticateUser(username,password) getSuppliers() getSupplier(supplierid) addSupplier(supplier) remove Supplier(supplier) getSales() getSale(saleld) addSale(sale) removeSale (sale) getTransactions()	getItem (itemid) getAllItem() getTransactions()	addItem(item) removeltem (item) updateltem(item) getItem (itemid) getAllItem() getSuppliers() getSupplier(supplierid) getSales() getSale(saleld) addSale(sale) removeSale (sale) getTransactions()
InventoryItem	setCategory(category) setWarehouse(warehouse)		setCategory(category) setWarehouse(warehouse)
Role	addRole() editRole() deleteRole() searchRole() setRole()		searchRole()
Permission	addPermission() editPermission() deletePermission() searchPermission()		searchPermission()

[3] Classes should correspond to classes in Section 3.1.

6.4 Boundary Conditions

Discuss boundary conditions initialization, termination and failure. See below for some questions relevant:

Initialization

- What data need to be accessed at startup time?

  At startup time, the system needs to access the configuration data such as database connection details, system settings, and user preferences.

- What services have to be registered?

  Services that need to be registered include logging services, authentication services, and any external system integrations.

- What does the user interface do at start up time?

  The user interface will display a welcome screen and login screen with relevant information.

Termination

- Are single subsystem is allowed to terminate?

  Each subsystem within the system is allowed to terminate independently. For example, the UserInterface subsystem, DatabaseInterface subsystem, and ExternalSystem subsystem can be terminated separately.

- Are subsystems notified if a single subsystem terminates?

  If a single subsystem terminates, other subsystems do not necessarily need to be notified. The system can handle the termination gracefully by ensuring that any pending operations or resources are properly closed or released within the terminating subsystem.

- How are updates communicated to the database?

  Updates to the database are typically communicated through the DatabaseInterface subsystem.

Failure

- How does the system behave when a node or communication link fails?

  If a node fails, the system will detect the failure and initiate appropriate actions. For example, reassigning tasks or responsibilities to other available nodes, redirecting requests to backup nodes, or notifying system administrators or operators about the failure.

  If a communication link fails, the system has a mechanisms in place to handle the loss of connectivity. For example, retrying the communication, using alternate communication paths or protocols, or buffering and queuing data until the connection is restored.

- How does the system recover from failure?

  The system will ensure data integrity and consistency even in the event of failures. It will involve using transactional mechanisms for database operations, will be implementing backup and recovery procedures for critical data, and performing regular backups to prevent data loss.

7 Glossary

8 References

"Stock Management System Basics" Microsoft, https://dynamics.microsoft.com/en-gb/field-service/inventory-management-system/#:~:text=The%20purpose%20of%20stock%20management,are%20never%20out%20of%20stock.

"Stock Management System Details " lovelycoding, https://www.lovelycoding.org/stock-management-system-project-for-final-year/

"What is stock management?" mecalux.com, https://www.mecalux.com/blog/what-is-stock-management

9 Appendix