Cairo University

Faculty of Computers and Artificial Intelligence

CS251

Introduction to Software Engineering

Toffee System

Software Design Specification

Contents

Software Design Specification

Team

ID	Name	Email	Mobile
20210119	Habiba Ihab Mohamed	habibaihab73@gmail.com	01102881028
20210083	Aya Ali Hassan	ayaali95248@gmail.com	01013167571
20210380	Mariam Ayman	Mariamayman3131@gmail.com	01152812450

Document Purpose and Audience

The purpose: this document is to provide a comprehensive overview of the system design for the Toffee project, including the architecture diagram, class diagrams, class descriptions, sequence diagrams, and state diagram. It aims to provide a clear understanding of the different components and their relationships within the system, as well as the flow of interactions between these components.

Audience:

- Project Managers
- Developers
- System Administrators
- Business Owners
- Designers

System Models

I. Architecture Diagram

The suitable software architecture for toffee system is ***three-tier client-server architecture*** this consists of three main layers (presentation layer, the business logic layer, and the data storage layer).

1. ***Presentation layer:*** The primary responsibility of the Layer is to handle user input and display information in a visually appealing and user-friendly manner, It includes components such as web browsers, mobile apps, and other user-facing interfaces that allow users to interact with the system.

2. ***Business Logic Layer:*** It handles the processing of user inputs and communicates with the data storage layer to fetch and store data. It contains the business logic and processing logic of the system, including authentication, authorization, validation, and business rules.
3. ***Data Storage Layer:*** This layer is responsible for storing and managing the data used by the Toffee system. It can include databases, file systems, or any other data storage mechanisms.

This architecture suitable for toffee system due to its

scalability -> allows for easy expansion and adaptation to handle growing user demands.

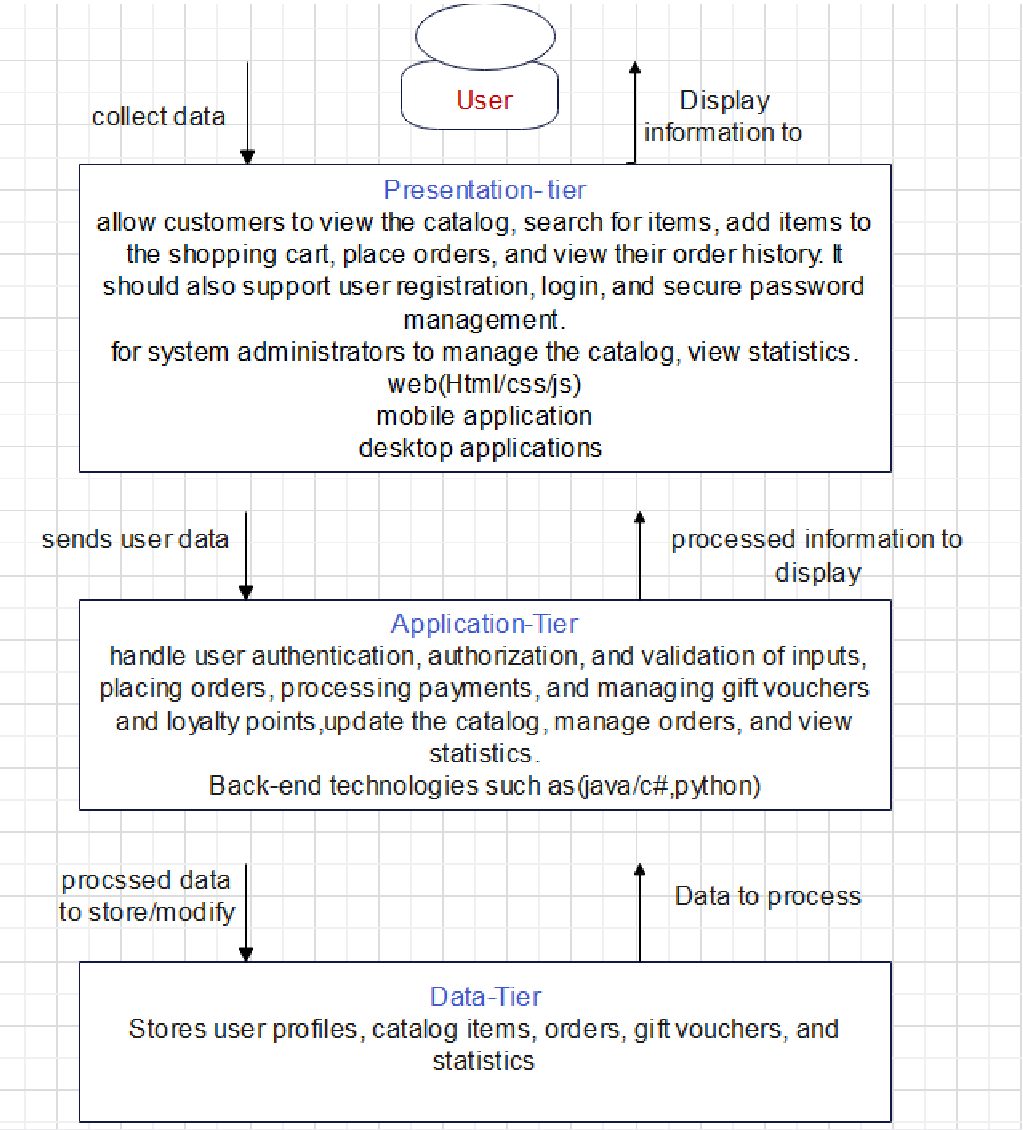
Modularity-> enables the system to be divided into distinct components, making it easier to develop, test, and maintain.

Maintainability-> aspect ensures that the system can be updated and enhanced without disrupting its overall functionality.

Security-> measures implemented in the architecture protect against potential vulnerabilities and ensure data privacy.

Flexibility-> of the architecture enables future enhancements and modifications without significant rework and focus on user experience.

It provides a robust and scalable foundation for building a reliable and efficient system that meets the requirements of the Toffee project.



Presentation- tier

allow customers to view the catalog, search for items, add items to the shopping cart, place orders, and view their order history. It should also support user registration, login, and secure password management.

for system administrators to manage the catalog, view statistics.

web(Html/css/js)

mobile application

desktop applications

Application-Tier

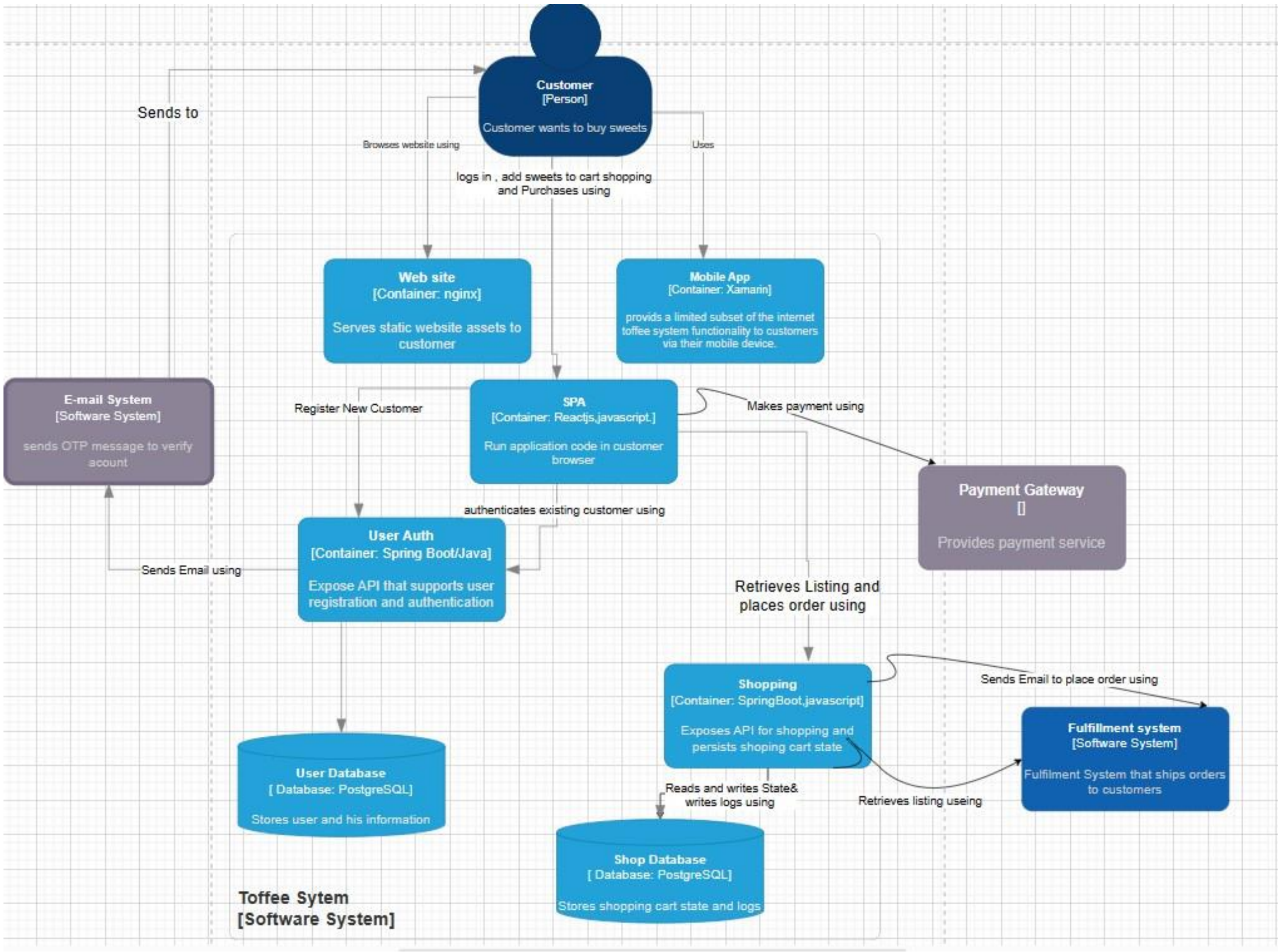
handle user authentication, authorization, and validation of inputs, placing orders, processing payments, and managing gift vouchers and loyalty points,update the catalog, manage orders, and view statistics.

Back-end technologies such as(java/c#,python)

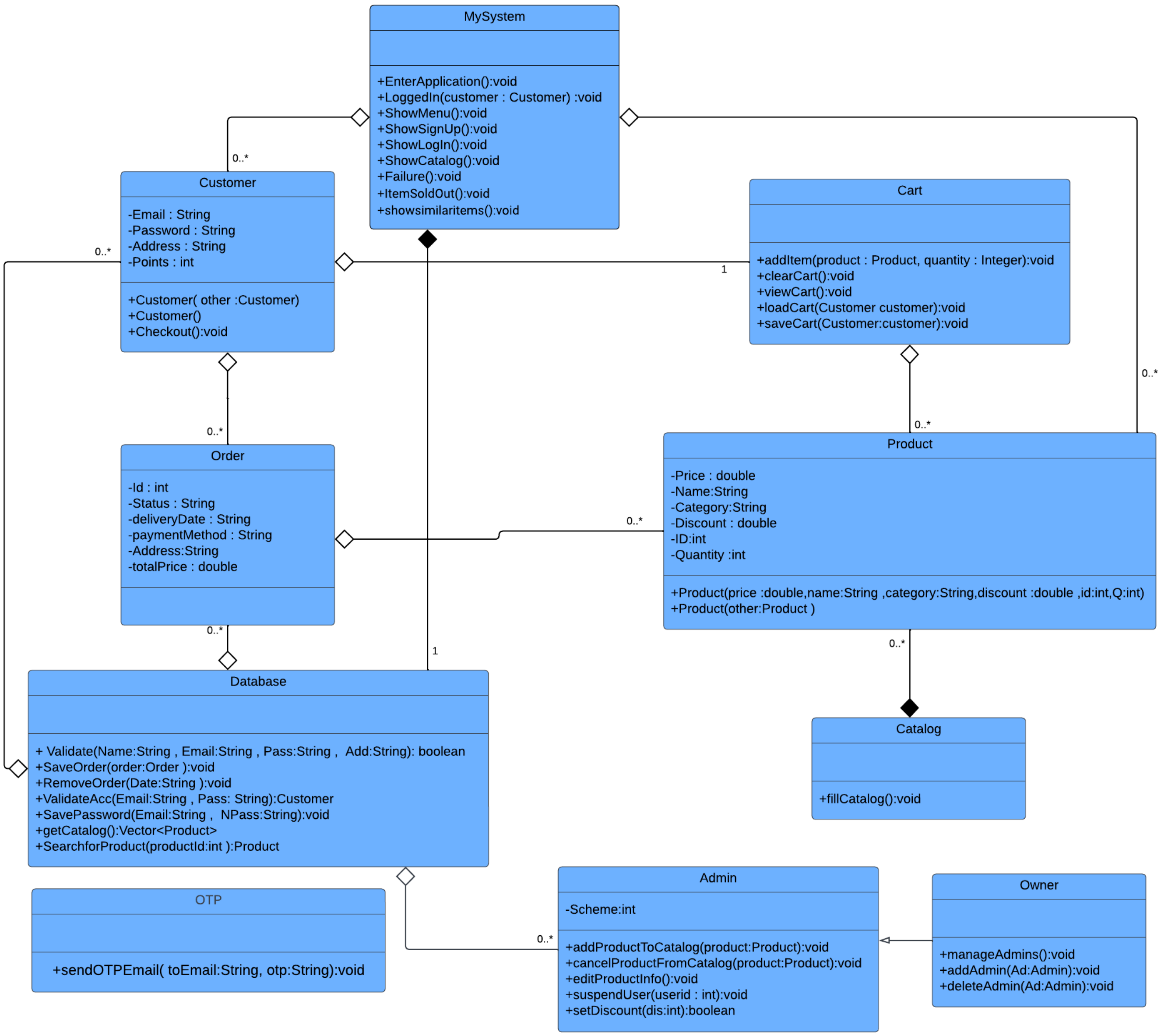
Data-Tier

Stores user profiles, catalog items, orders, gift vouchers, and statistics

C4 model for Toffee system


Customer
[Person]
Customer wants to buy sweets
Sends to
Browses website using
Uses
logs in , add sweets to cart shopping and Purchases using
Web site
[Container: nginx]
Serves static website assets to customer
Mobile App
[Container: Xamarin]
provids a limited subset of the internet toffee system functionality to customers via their mobile device.
E-mail System
[Software System]
sends OTP message to verify acount
Register New Customer
SPA
[Container: Reactjs,javascript.]
Run application code in customer browser
Makes payment using
Payment Gateway
[]
Provides payment service
authenticates existing customer using
User Auth
[Container: Spring Boot/Java]
Expose API that supports user registration and authentication
Sends Email using
Retrieves Listing and places order using
Shopping
[Container: SpringBoot,javascript]
Exposes API for shopping and persists shoping cart state
Sends Email to place order using
Fulfillment system
[Software System]
Fulfilment System that ships orders to customers
Reads and writes State& writes logs using
Retrieves listing useing
User Database
[Database: PostgreSQL]
Stores user and his information
Shop Database
[Database: PostgreSQL]
Stores shopping cart state and logs
Toffee Sytem
[Software System]

II. Class Diagram(s)


MySystem
+EnterApplication():void
+LoggedIn(customer : Customer) :void
+ShowMenu():void
+ShowSignUp():void
+ShowLogIn():void
+ShowCatalog():void
+Failure():void
+ItemSoldOut():void
+showsimilaritems():void
0..*
Customer
-Email : String
-Password : String
-Address : String
-Points : int
+Customer(other :Customer)
+Customer()
+Checkout():void
0..*
Cart
+addItem(product : Product, quantity : Integer):void
+clearCart():void
+viewCart():void
+loadCart(Customer customer):void
+saveCart(Customer:customer):void
1
0..*
0..*
Order
-Id : int
-Status : String
-deliveryDate : String
-paymentMethod : String
-Address:String
-totalPrice : double
0..*
Product
-Price : double
-Name:String
-Category:String
-Discount : double
-ID:int
-Quantity :int
+Product(price :double,name:String ,category:String,discount :double ,id:int,Q:int)
+Product(other:Product)
0..*
0..*
1
Database
+ Validate(Name:String , Email:String , Pass:String , Add:String): boolean
+SaveOrder(order:Order):void
+RemoveOrder(Date:String):void
+ValidateAcc(Email:String , Pass: String):Customer
+SavePassword(Email:String , NPass:String):void
+getCatalog():Vector<Product>
+SearchforProduct(productId:int):Product
0..*
Catalog
+fillCatalog():void
OTP
+sendOTPEmail(toEmail:String, otp:String):void
0..*
Admin
-Scheme:int
+addProductToCatalog(product:Product):void
+cancelProductFromCatalog(product:Product):void
+editProductInfo():void
+suspendUser(userid : int):void
+setDiscount(dis:int):boolean
Owner
+manageAdmins():void
+addAdmin(Ad:Admin):void
+deleteAdmin(Ad:Admin):void

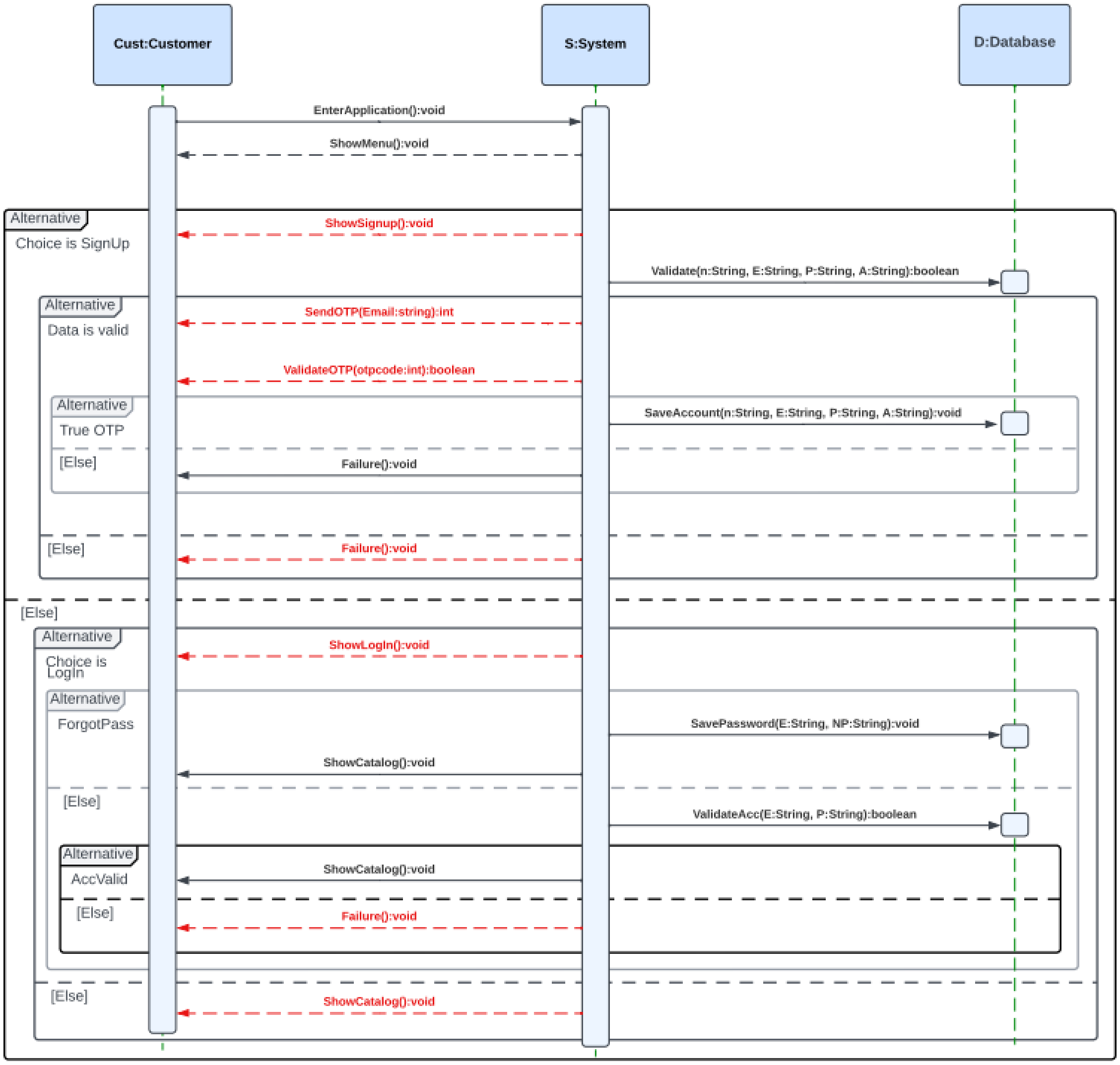
III. Class Descriptions

Class ID	Class Name	Description & Responsibility
C#01	Catalog	Managing the catalog of items like add item, remove item, update info of item, search of item by name or brand
C#02	Registered Customer	Manage user accounts include registration, login and his information.
C#02	Registered Customer	Validates email, password, and address during registration.
C#03	Cart	managing the shopping cart of registered user, provides methods to add items to the cart, view items in the cart, and remove items from the cart.
C#03	Cart	Calculates the total price of items in the cart and applies gift vouchers or loyalty points for payment.
C#04	Product	Represents an item in the catalog with properties such as name, category, ID, brand, price, and discount percentage. has methods to add item information.
C#05	Admin	Responsible for Set Loyalty Points Scheme, suspend user, Update catalog, cancel item, View all orders.
C#06	System	This class is responsible for managing the application to show the menu, Sign up. have method to validate the email, validate account, sends OTP messages and validate OTP messages.
C#07	Database	This class is responsible for storing the users accounts and their orders and information of the admins.
C#08	Owner	This class is responsible for adding and removing admins from the system.

IV. Sequence diagrams

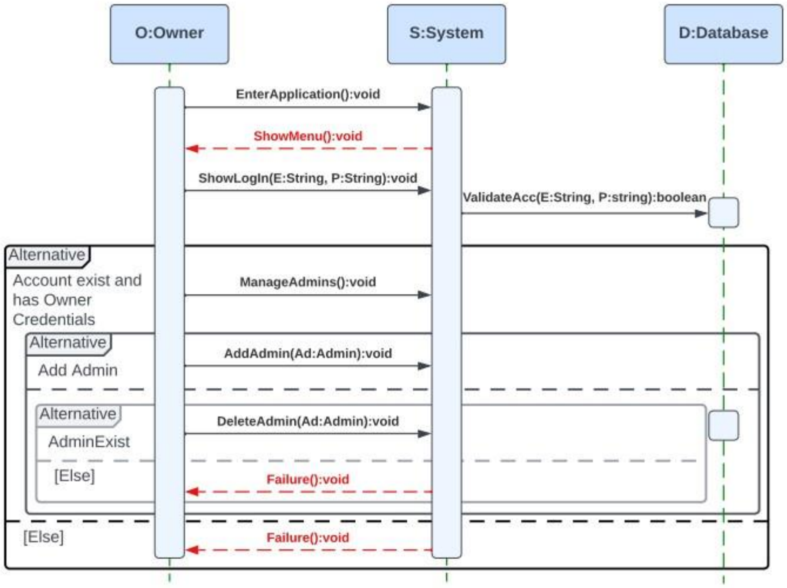
US #2 & #3 & #14

Sequence Diagram for Show Catalog + LogIn + SignUp



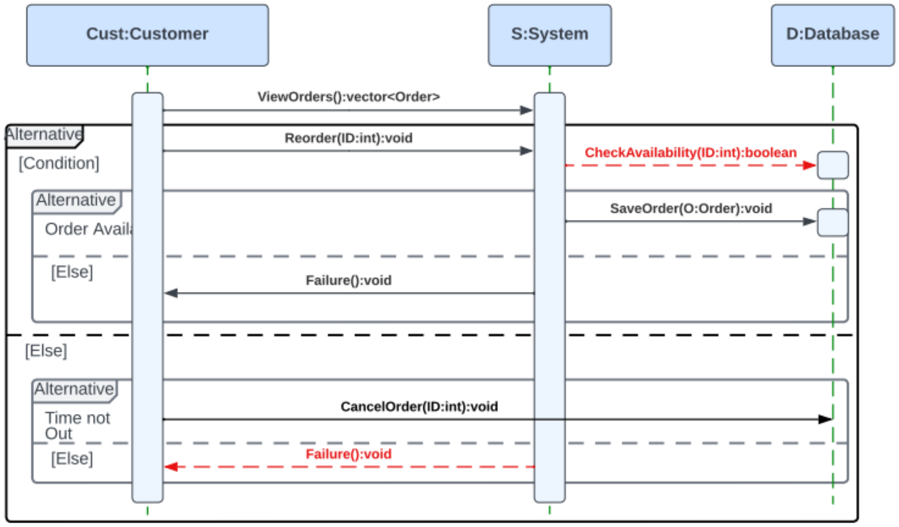
US #1

Sequence diagram for Adding and Removing Adimins



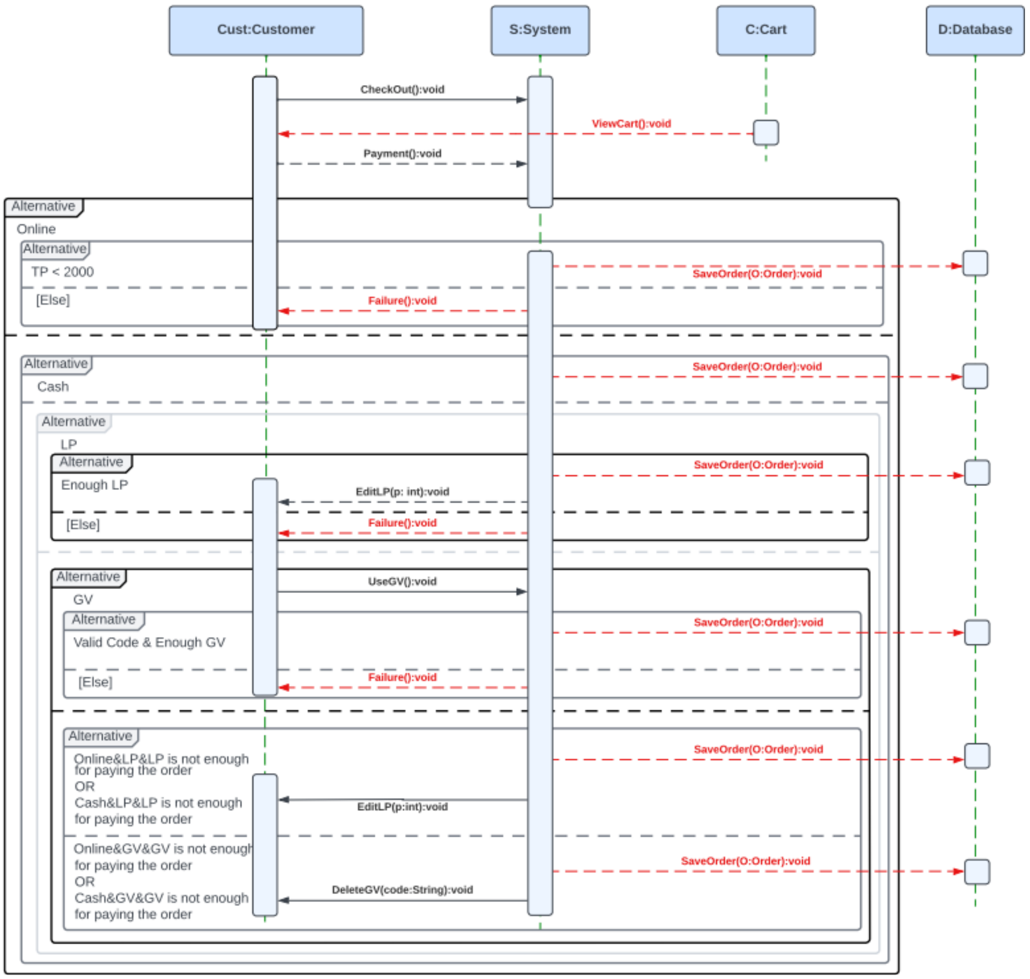
US# 6

Sequence Diagram for Reordering or canceling Order



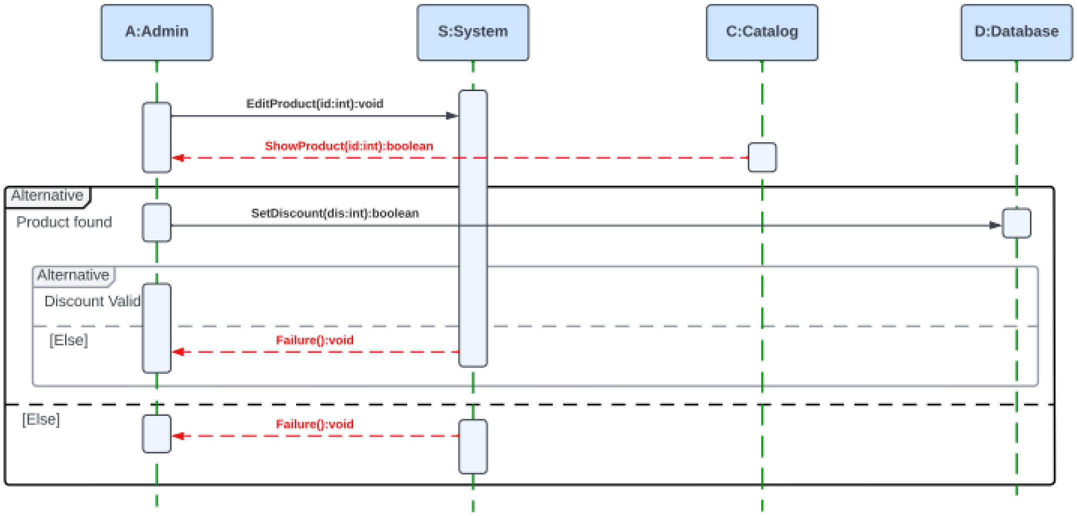
US#18

Sequnece Diagram
for Checking Out

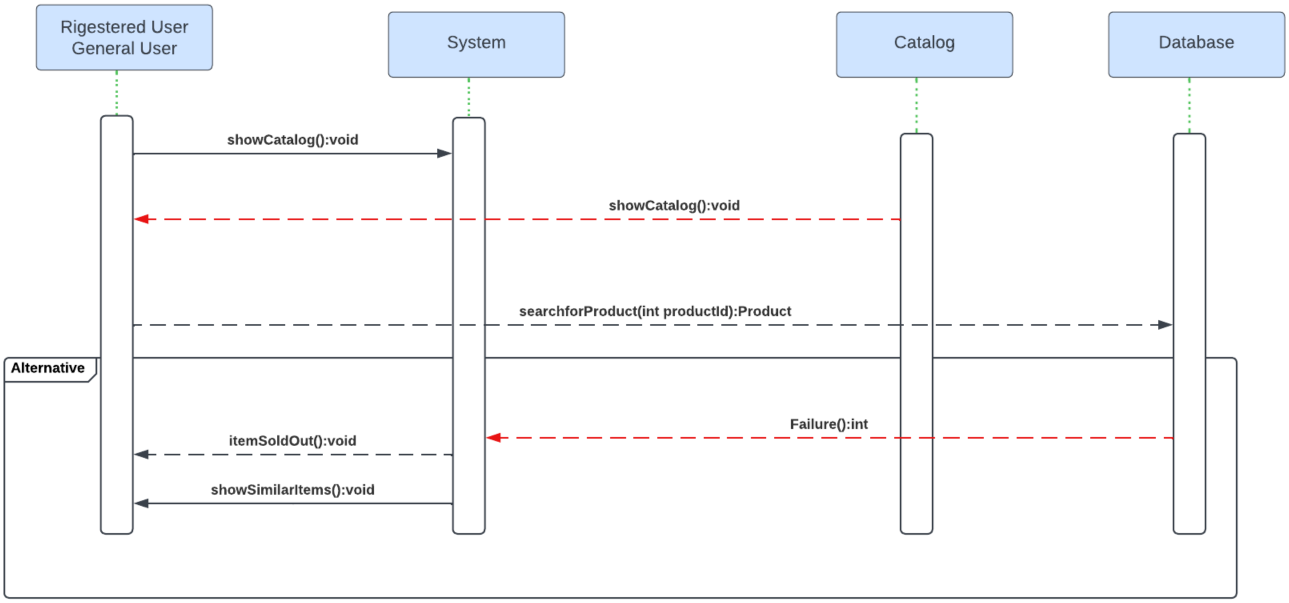


US#9

Sequence Diagram
For Set Discount



US#5


Rigestered User
General User
System
Catalog
Database
showCatalog():void
showCatalog():void
searchforProduct(int productId):Product
Alternative
Failure():int
itemSoldOut():void
showSimilarItems():void

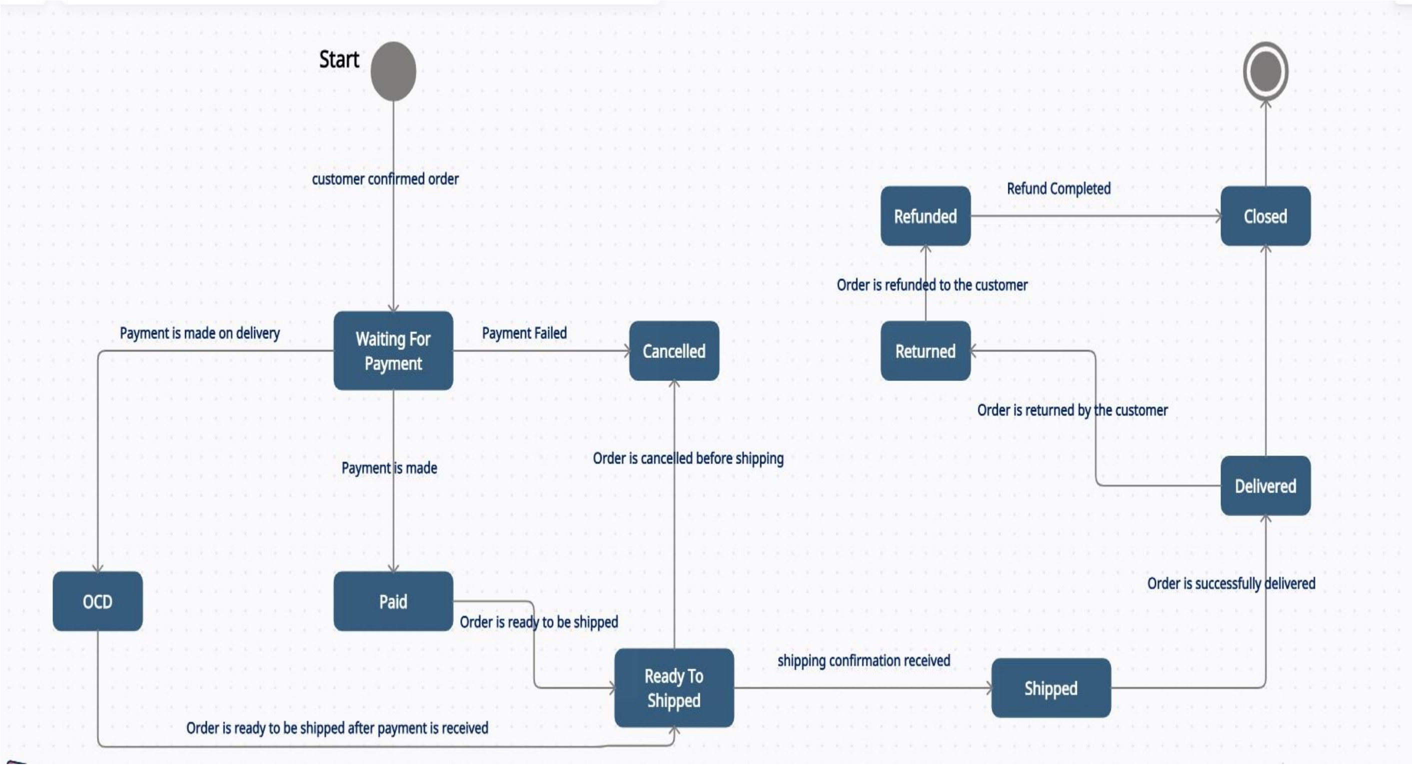
Class - Sequence Usage Table

Sequence Diagram	Classes Used	Methods
1. Show Catalog & Sign Up & Log In	Customer System Database	System: 1. Void EnterApplication() 2. Void ShowMenu() 3. Void Show SignUp() 4. Void ShowLogIn() 5. Int SendOTP(String Email) 6. Boolean ValidateOTP(int otpcode) 7. Void Failure() 8. Void ShowCatalog() Database: 1. Boolean Validate(String n, String E, String P, String A) 2. Void SaveAccount(String n, String E, String P, String A) 3. Void SavePassword(String E, String P) 4. Boolean ValidateAcc(String E, String P)
2.Check Out	Customer System Cart Database	Customer: 1. Void CheckOut() 2. Void Payment() System: 1. Void Failure() 2. Void EditLP(int p) 3. Void DeleteGV(String code) Cart: 1. Void ViewCart() Database: 1. SaveOrder(Order O)
3.Set Discount	Admin System Catalog Database	Admin: 1. Void EditProduct(int id) 2. Void SetDiscount(int dis) System: 1. Void Failure() Catalog: 1. Boolean ShowProduct(int id)
4.Adding & Removing Admins	Owner System Database	Owner: 1. Void ManageAdmins() 2. Void AddAdmin(Admin Ad) 3. Void DeleteAdmin(Admin Ad) System: 1. Void EnterApplication() 2. Void ShowMenu() 3. Void ShowLogIn() 4. Void Failure() Database: 1. Boolean ValidateAcc(String E, String P)

5.Reorder & Cancel Order	Customer System Database	Customer: 1. Vector<Orders> ViewOrders() 2. Reorder(int id) 3. CancelOrder(int id) System: 1. Void Failure() Database: 1. Boolean CheckAvailability() 2. Void SaveOrder(Order O)
6.Search for an item	Customer System Database Catalog	Customer: 1. showCatalog():void System: 1. itemSoldOut():void 2. showSimilarItems():void Database: 1. Product SearchForProduct(int productId) 2. int Failure() Catalog: 1. showCatalog():void

Software Design Specification

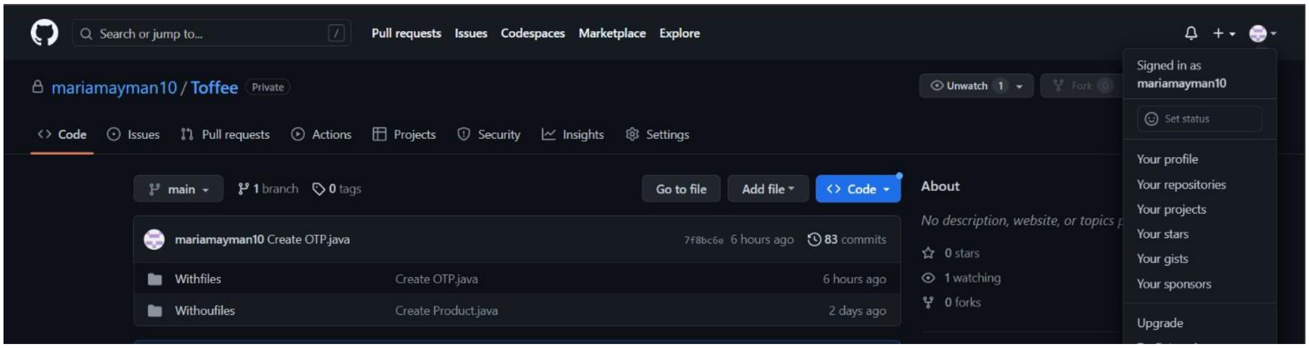
State Diagram


Start
customer confirmed order
Payment is made on delivery
Waiting For Payment
Payment Failed
Cancelled
Payment is made
Order is cancelled before shipping
OCD
Paid
Order is ready to be shipped
Ready To Shipped
shipping confirmation received
Shipped
Order is ready to be shipped after payment is received
Refunded
Refund Completed
Closed
Order is refunded to the customer
Returned
Order is returned by the customer
Delivered
Order is successfully delivered

Software Design Specification

GitHub Snapshots

Mariam Ayman Taha - 20210380

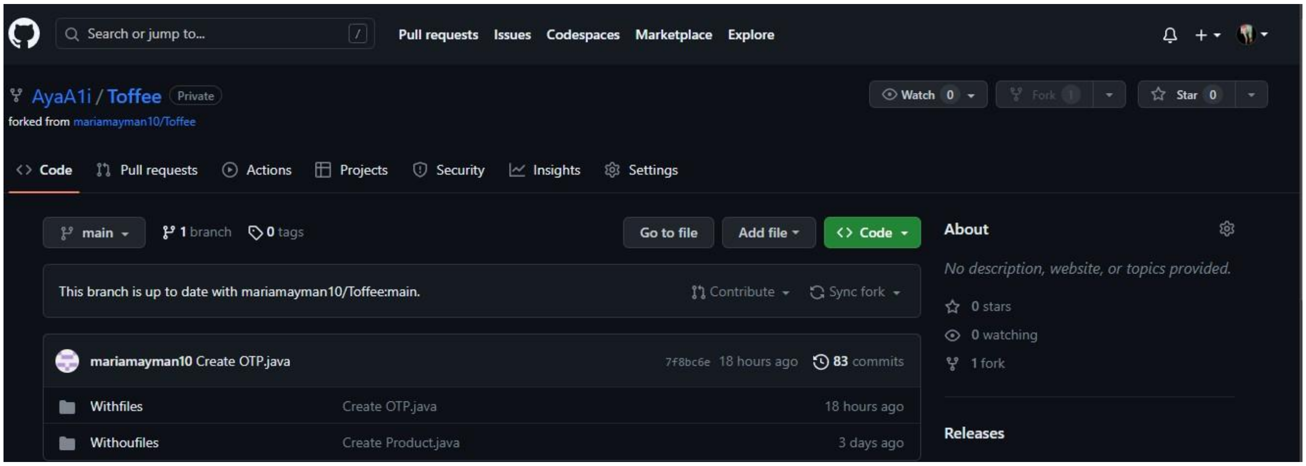


mariamayman10 Create OTP.java 7f8bc6e 6 hours ago 83 commits

Withfiles Create OTP.java 6 hours ago

Withoufiles Create Product.java 2 days ago

Aya Ali Hassan - 20210083



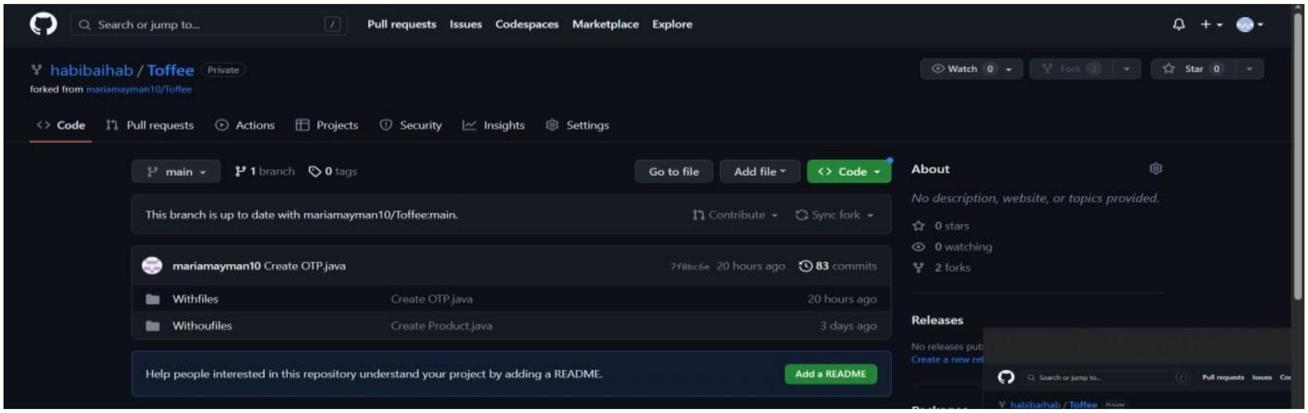
This branch is up to date with mariamayman10/Toffee:main.

mariamayman10 Create OTP.java 7f8bc6e 18 hours ago 83 commits

Withfiles Create OTP.java 18 hours ago

Withoufiles Create Product.java 3 days ago

Habiba Ihab - 20210119



This branch is up to date with mariamayman10/Toffee:main.

mariamayman10 Create OTP.java 7f8bc6e 20 hours ago 83 commits

Withfiles Create OTP.java 20 hours ago

Withoufiles Create Product.java 3 days ago

Help people interested in this repository understand your project by adding a README.

Software Design Specification

Tools

- Edrawsoft is used for Three-tier architecture.
- Lucid is used for sequence diagrams and class diagram.
- diagrams.net used for c4 model architecture.

Ownership Report

Item	Owners
Aya Ali Hassan	**Class Diagram** **Last sequence diagram** **Implemented 2 classes in java**
Mariam Ayman Taha	**First 5 Sequence diagrams** **Sequence usage table** **Implemented 3 classes in java**
Habiba Ihab Mohamed	**Class Architecture** **State Diagram** **Class responsibility table** **Implemented 2 classes in java**