Cairo University
Faculty of Computers and Artificial Intelligence

CS251
Introduction to Software Engineering

Toffee

Software Design Specifications

Version 0.0

ID	Name	Email	Mobile
20210499	Noor Eyad Eissa	nooreyadd39@gmail.com	01032750535
20210133	Rana Essam Ibrahim	ranaessam03@gmail.com	01145303111
20210437	Nourhan Abdallah	nourhanabdalla031@gmail.com	01014644331

CS251: Phase 2 – Toffee Team
Project: Toffee

Software Design Specification

April 2023

Contents

CS251: Phase 2 – Toffee Team
Project: Toffee

Software Design Specification

Team

ID	Name	Email	Mobile
20210499	Noor Eyad Eissa	nooreyadd39@gmail.com	01032750535
20210133	Rana Essam Ibrahim	ranaessam03@gmail.com	01145303111
20210437	Nourhan Abdallah	nourhanabdalla031@gmail.com	01014644331

Document Purpose and Audience

This is a software design specification document. It includes all the information related to the software design process, software design requirements, architecture, and others. It ensures that all the main software design requirements are met, and also helps streamline the workflow of the software development team by guiding them through the right steps to how to build the software.

This document targets the software development team and any of the stakeholders that may have a technical background and are interested to know the details of the implementation and how everything will be integrated together.

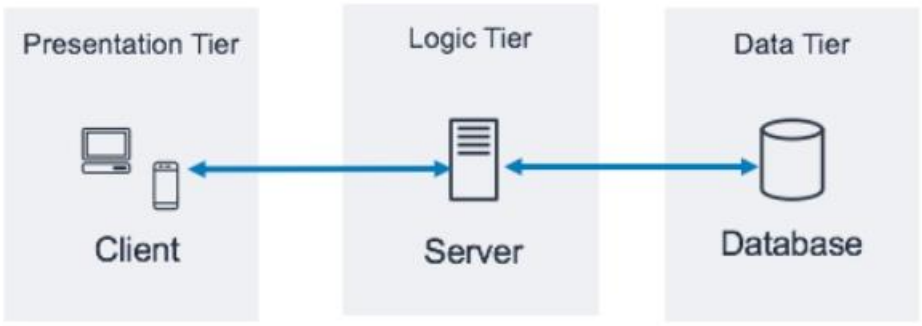
Software Design Specification

System Models

I. Architecture Diagram

Three Tier architecture

Three-tier architecture is a well-established software application architecture that organizes applications into three logical and physical computing tiers: the presentation tier, or user interface; the application tier, where data is processed; and the data tier, where the data associated with the application is stored and managed.



Presentation tier

The presentation tier is the user interface and communication layer of the application, where the end user interacts with the application. Its main purpose is to display information to and collect information from the user. This top-level tier can run on a web browser, as desktop application, or a graphical user interface (GUI), for example. Web presentation tiers are usually developed using HTML, CSS and JavaScript. Desktop applications can be written in a variety of languages depending on the platform.

In our application "TOFFEE" the presentation tier represents the interface of all the functions the user can do like create account, add to cart, checkout an order, views orders history and …..

Without this tire the user wouldn't be able to interact with any of the system's functions

Software Design Specification

Application tier

The application tier, also known as the logic tier or middle tier, is the heart of the application. In this tier, information collected in the presentation tier is processed - sometimes against other information in the data tier - using business logic, a specific set of business rules. The application tier can also add, delete or modify data in the data tier.

The application tier is typically developed using Python, Java, Perl, PHP or Ruby, and communicates with the data tier using API calls.

In our application "TOFFEE" this tier represents the tire which apply the main logic of the application, receives the function which will the user has chosen to apply which the system, then it interacts with the data base tier to execute some function by getting the required data to complete this function examples of functions that application tier applies-> send and get to and from the data base a request to verify credentials, send and receive from the GUI message about some functions and verify OTP of some user.

Data tier

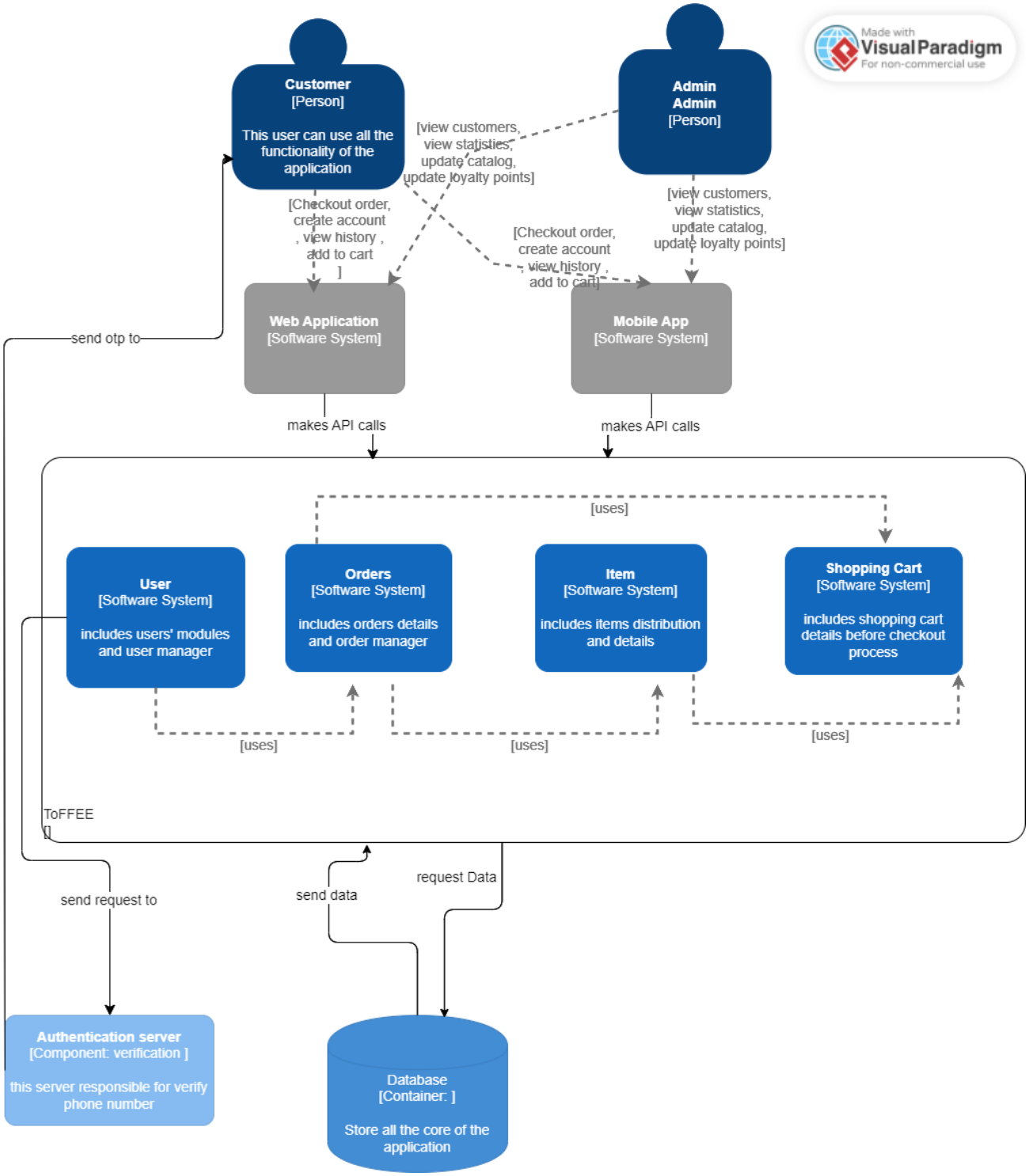
The data tier, sometimes called database tier, data access tier or back-end, is where the information processed by the application is stored and managed. This can be a relational database management system such as PostgreSQL, MySQL, MariaDB, Oracle, DB2, Informix or Microsoft SQL Server, or in a NoSQL Database server such as Cassandra, CouchDB or MongoDB.

In a three-tier application, all communication goes through the application tier. The presentation tier and the data tier cannot communicate directly with one another.

Why do we choose this pattern??

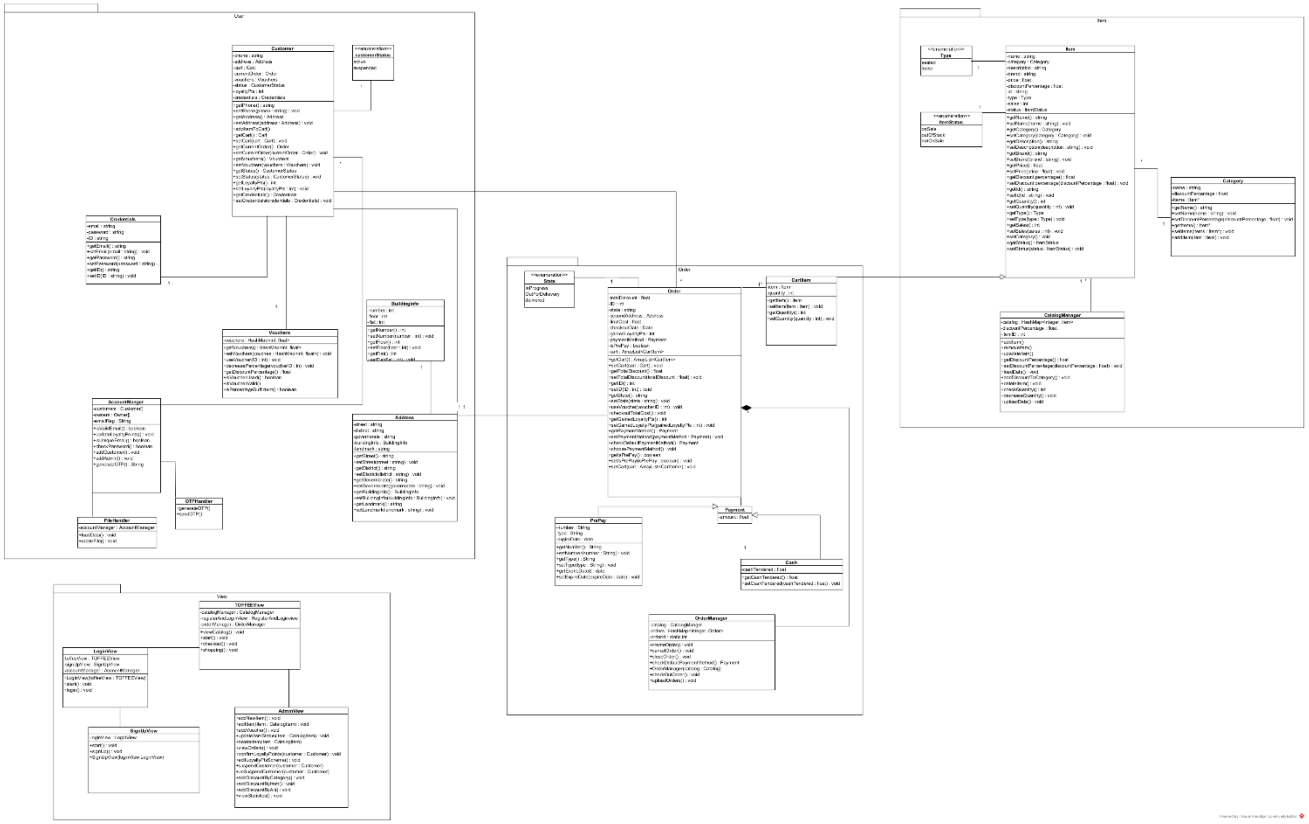
- **Faster development**: Because each tier can be developed simultaneously by different teams, an organization can bring the application to market faster, and programmers can use the latest and best languages and tools for each tier.
- **Improved scalability**: Any tier can be scaled independently of the others as needed.
- **Improved reliability**: An outage in one tier is less likely to impact the availability or performance of the other tiers.
- **Improved security**: Because the presentation tier and data tier can't communicate directly, a well-designed application tier can function as a sort of internal firewall, preventing SQL injections and other malicious exploits.

Software Design Specification



Software Design Specification

II. Class Diagram(s)



NOTE: the image is included in the folder submitted as a PDF file

Software Design Specification

III. Class Descriptions

Package Name	Class ID	Class Name	Description & Responsibility
User	C01	BuildingInfo	This class represents the building info part of the address as there might be more than 1 customer with the same address but there cannot be more than 1 customer with the same building info.
User	C02	AccountManager	This class is responsible for validating emails and passwords, sending OTP for new account-registerers, loading customers information from csv file
User	C03	OTPHandler	This class handles OTP, it generates one and sends it to the customer
User	C04	FileHandler	This class is responsible for reading the customer csv file and inserting data into a hash-map.
User	C05	Customer	This class represents the customer, it holds information about him/her like phone number, loyalty Points, cart and his status.
User	C06	Credentials	This class represents important credentials that the user needs to enter in order to login/register like email and password.
User	C07	Address	This class represents the customers' address information and it has the
User	C08	CustomerStatus	This is an enumeration class to represent the status of the customer, either if he is suspended by the admin or active
User	C09	Vouchers	This class represents vouchers that a certain customer is allowed to use. the relationship between this class and the customer class is that the customer class composites the voucher class.
Order	C10	Order	This class represents how the customer will interact with the shopping cart and what will the shopping cart contain. It contains attributes like state of order,
Order	C11	Payment	This class represents the amount that the customer will have to pay in order to receiver his/her order, it's a parent class for both prepay class and cash class, which they both represent the 2 available ways of payment that the system offer.
Order	C12	CartItem	This class represents items inside the cart. This class has a relationship with the cart class that the cart composites cart item. It also inherits from the Item class (C8)
Order	C13	Cash	This class represent how the system will deal with the paid-on-delivery orders

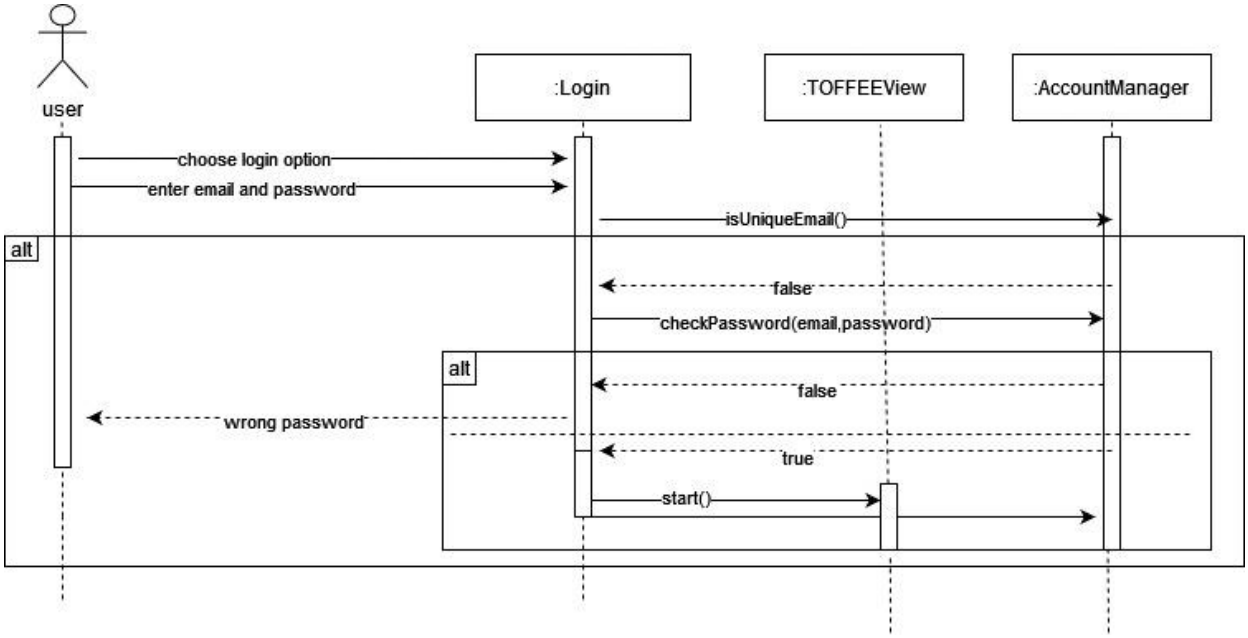
Software Design Specification

Package Name	Class ID	Class Name	Description & Responsibility
Order	C14	PrePay	This class represent how the system will deal with orders that have to be paid before they're received (those with total amount greater than 2000)
Order	C15	OrderManager	This class is responsible for all operations that will be performed on the order to create it, add item to cart, remove item from cart and close it.
Order	C16	State	This class is an enumeration for the state of the order, whether it is out for delivery, in progress or delivered.
Item	C17	ItemStatus	This class represents the state or the status of the item, either if it's on sale, not on sale or out of stock.
Item	C18	CatalogManager	This class is responsible for all the operations that will be performed on the items in the store, for example, add item, remove item, check quantity of the item and so on.
Item	C19	Category	This class represents categories of items in the store.
Item	C20	Type	This class is an enumeration for the type of items in the store (loose/solid)
Item	C21	Item	This class represents the entity item in the store system, it is responsible for all operations needed to create an item in the system.
View	C22	TOFFEEView	This class is responsible for displaying the catalog, checking out, start shopping
View	C23	AdminView	This class is responsible for all functions that the admin can do and all the changes that the admin can do to the system, for example, add new item, remove item, view orders and so on
View	C24	SighUpView	This class is responsible for signing up users that are new to the system and will create an account to shop on the system
View	C25	LoginView	This class is responsible for logging in users that already have account(s) on the system and are logging in to shop.

Software Design Specification

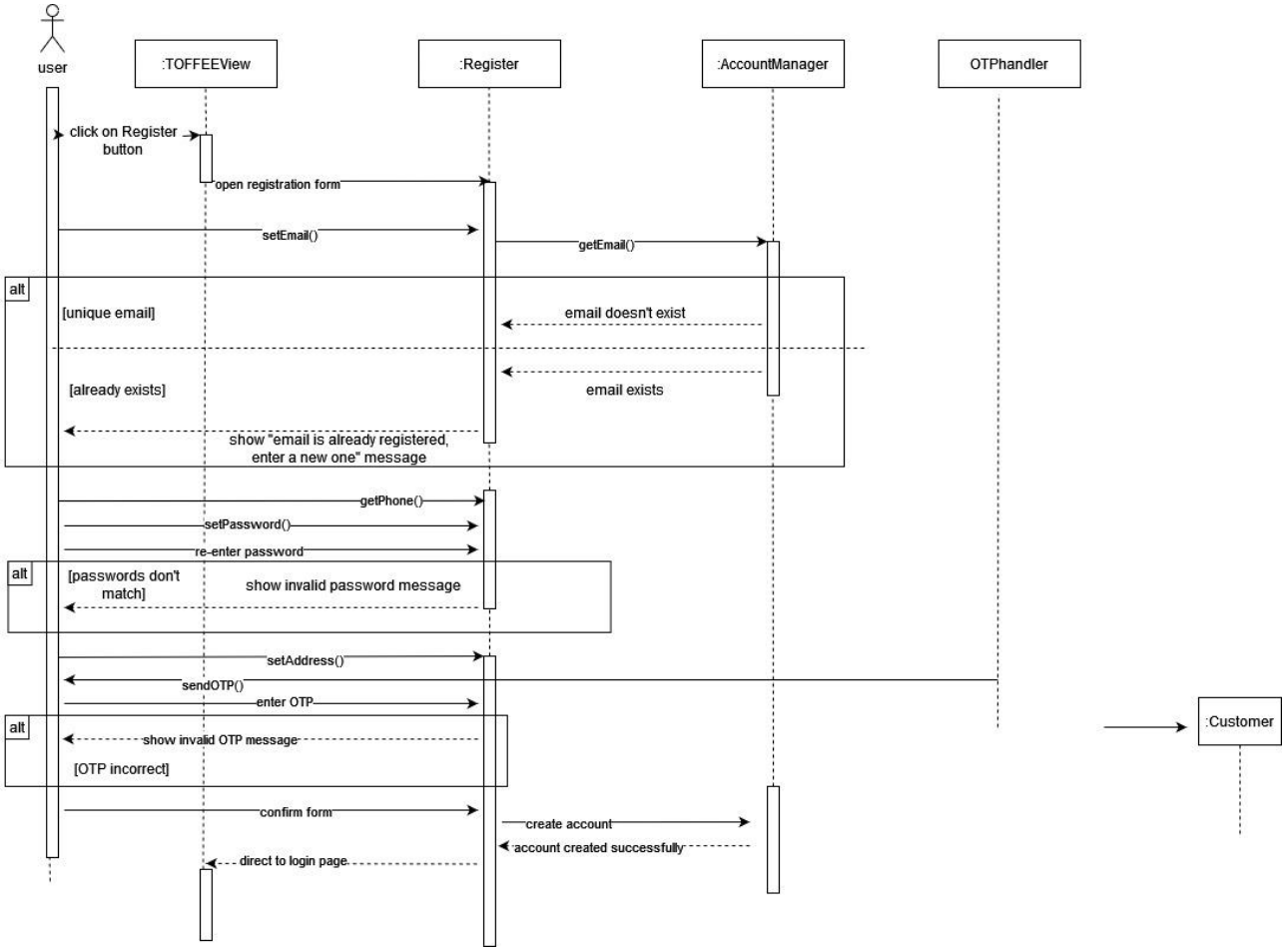
IV. Sequence diagrams

1. Log-in:



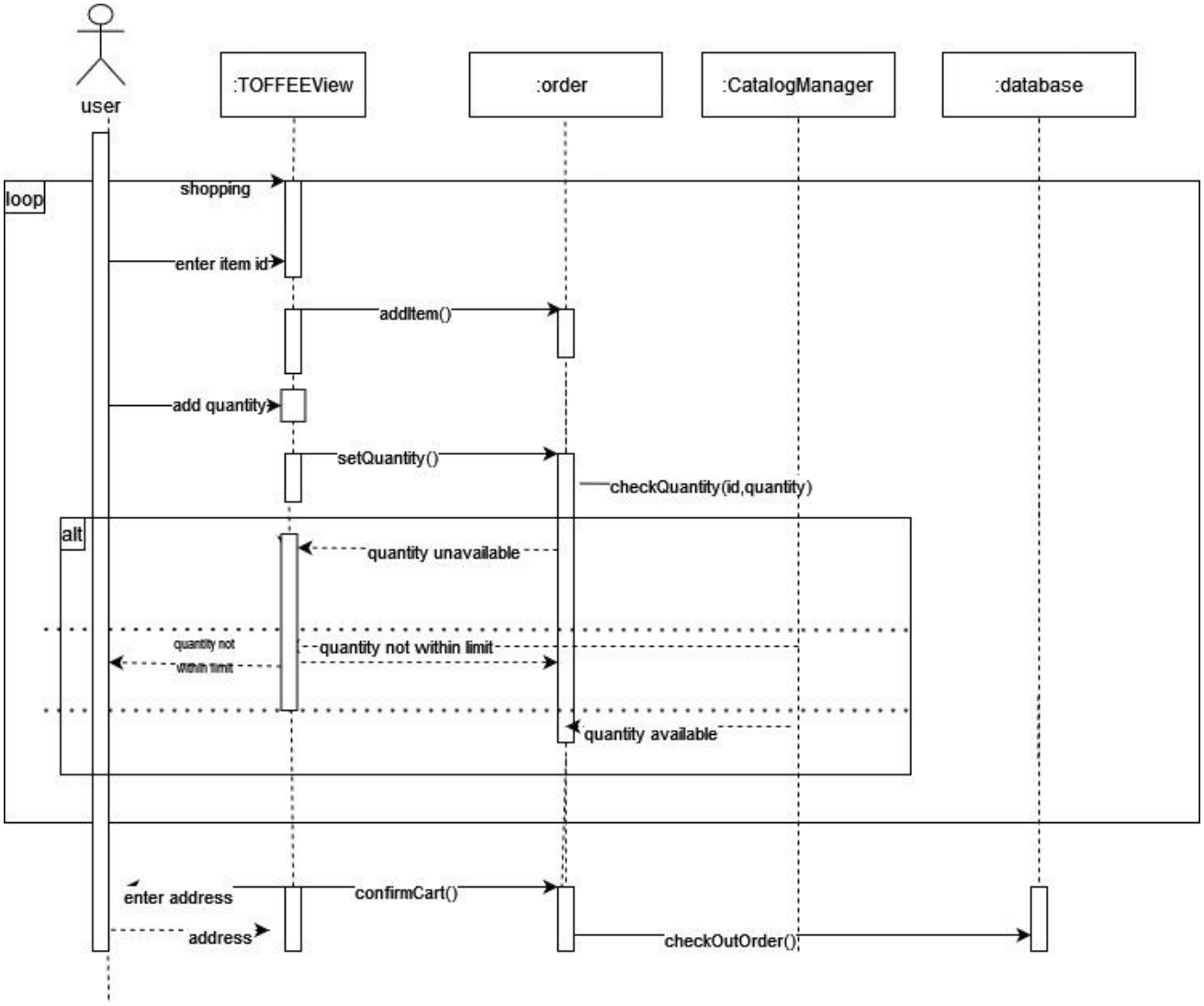
Software Design Specification

2. Register into the system:



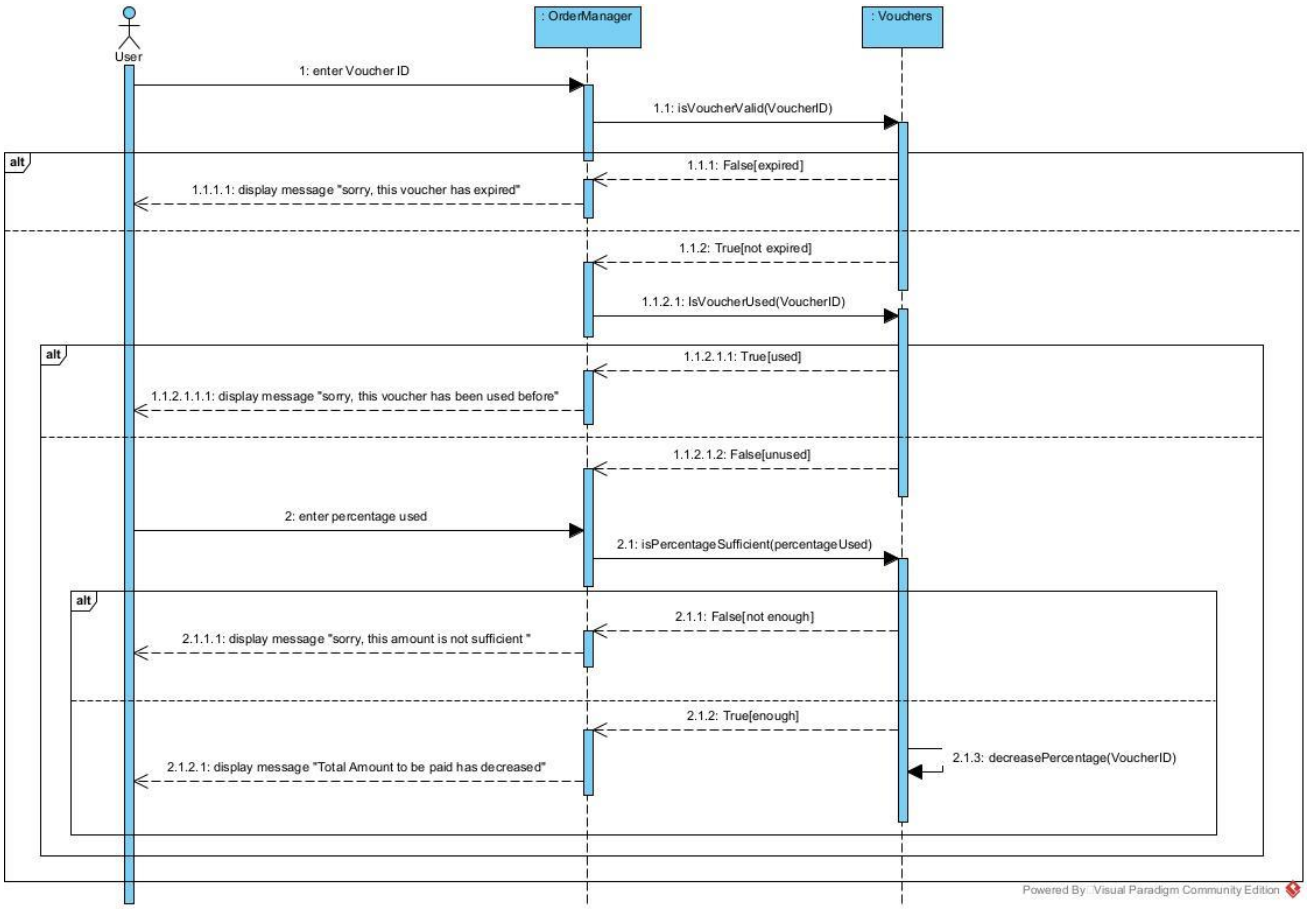
Software Design Specification

3. Add item to cart:



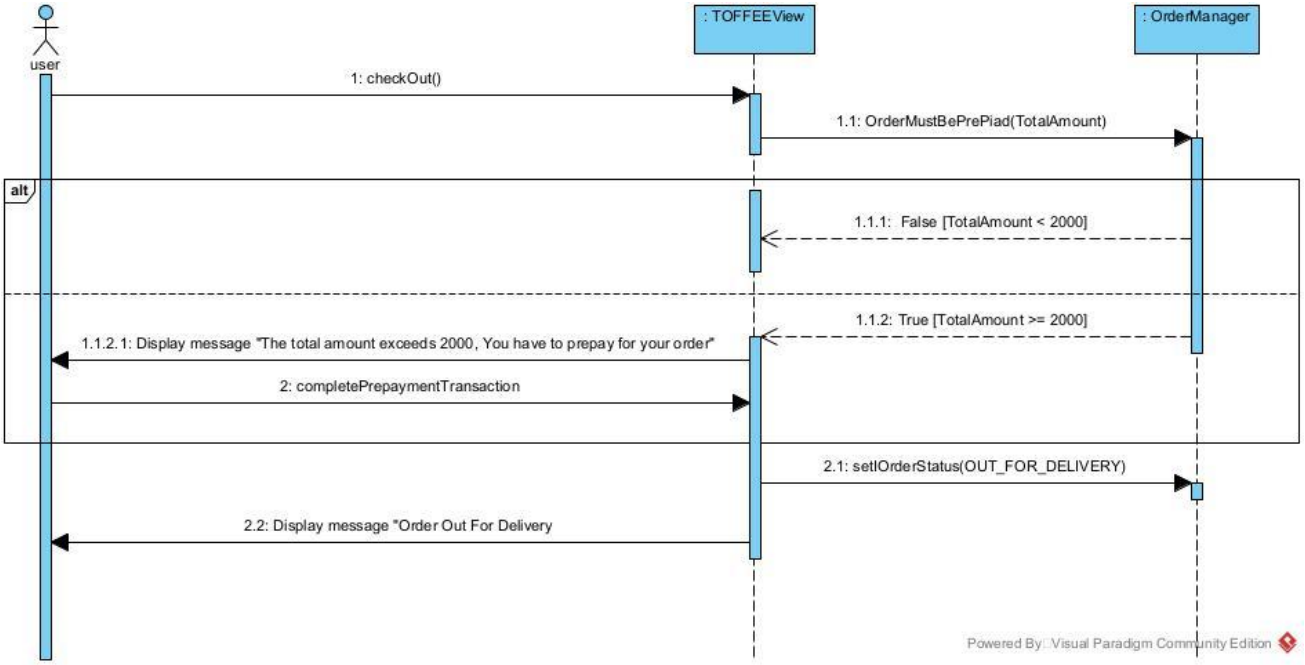
Software Design Specification

4. Using Vouchers:



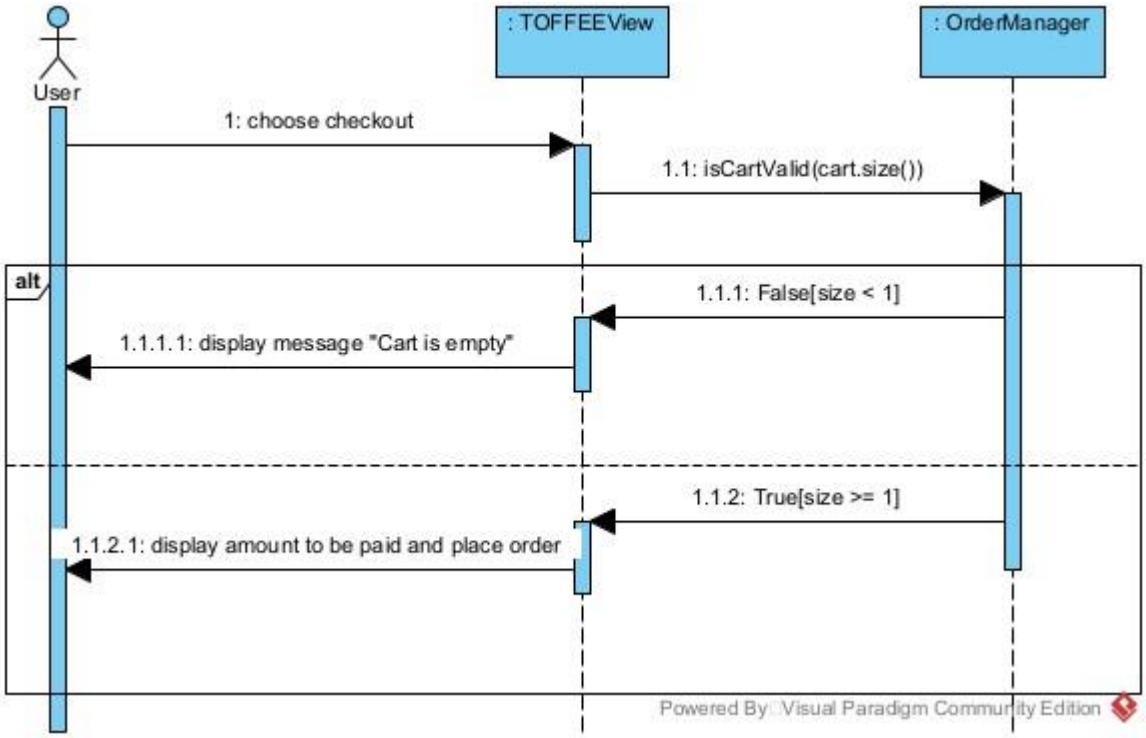
Software Design Specification

5. Prepaid order:



Software Design Specification

6. Checkout:



Software Design Specification

V. Sequence Usage Table

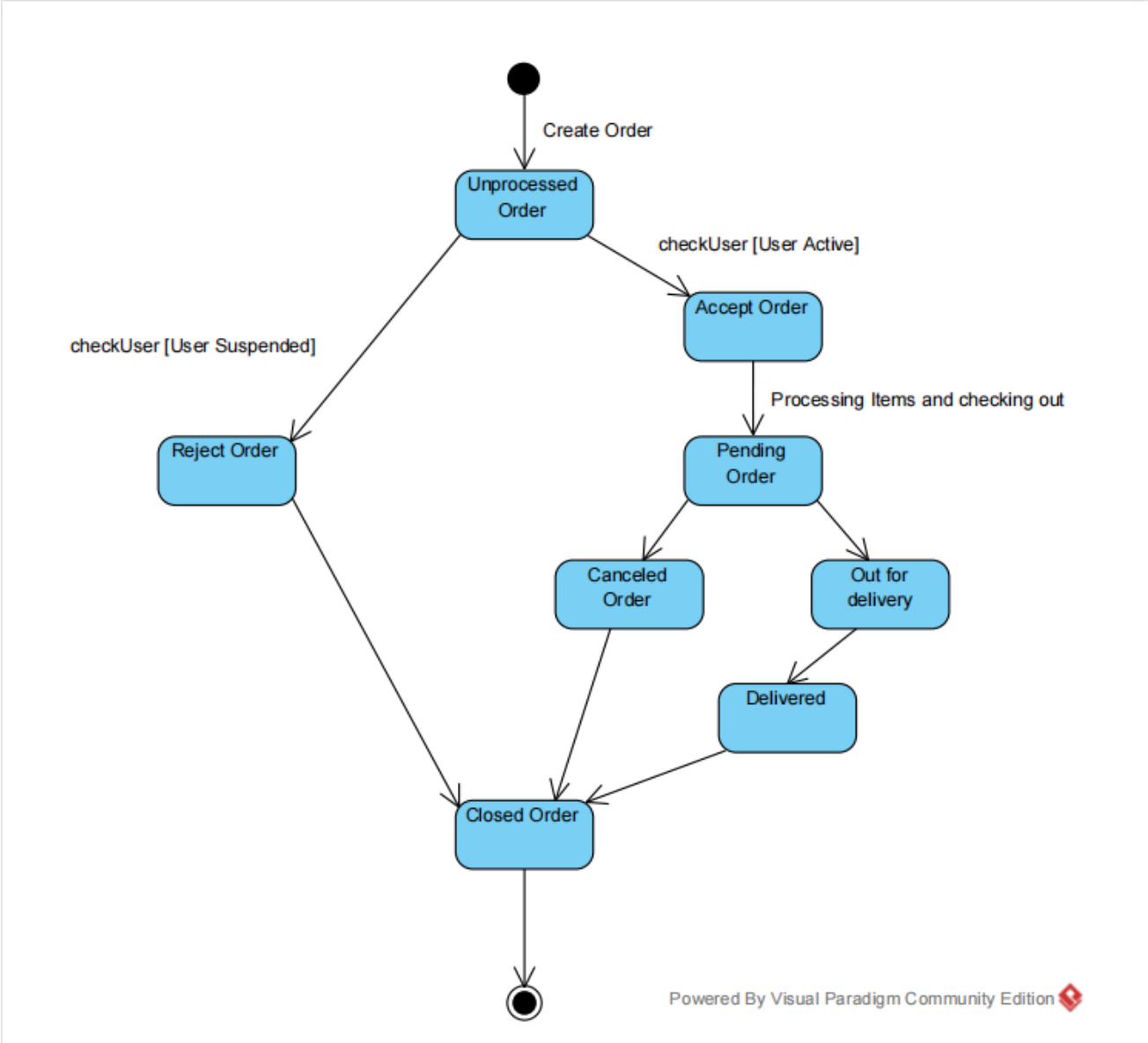
Sequence Diagram	Classes Used	All Methods Used
Log-in	Login	• Choose login option • enterEmailAndPassword(email, password) • output message when invalid
	TOFFEEView	• start()
	AccountManager	• isUniqueEmail(email) • checkPassword(password)
Register into the system	TOFFEEView	• choose register option • open registration form
	Register	• setEmail(email) • getEmail() • setPassword(password) • setAddress(address)
	AccountManager	• isUniqueEmail() • create account • getPhone()
	OTPHandler	• SendOTP()
Using Voucher	OrderManager	• Enter voucher ID
	Vouchers	• isVoucherValid(VoucherID) • isVoucherUsed(VoucherID) • isPercentageSufficient(percentage Used) • decreasePercentage(VoucherID)
Prepaid order	TOFFEEView	• checkout()
	OrderManager	• OrderMustBePrePaid(TotalAmount) • setOrderStatus(OUT_FOR_DELIVERY)

Software Design Specification

Sequence Diagram	Classes Used	All Methods Used
Checkout	TOFFEEView	• choose checkout
	OrderManager	• isCartValid(cart.size())

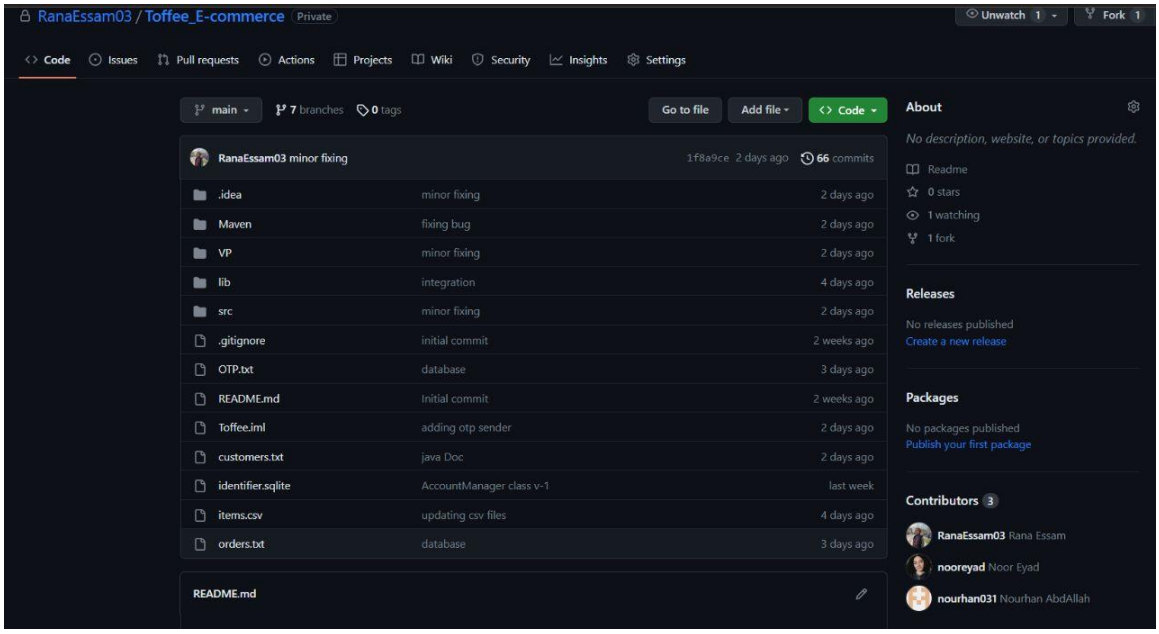
Software Design Specification

VI. State Diagram

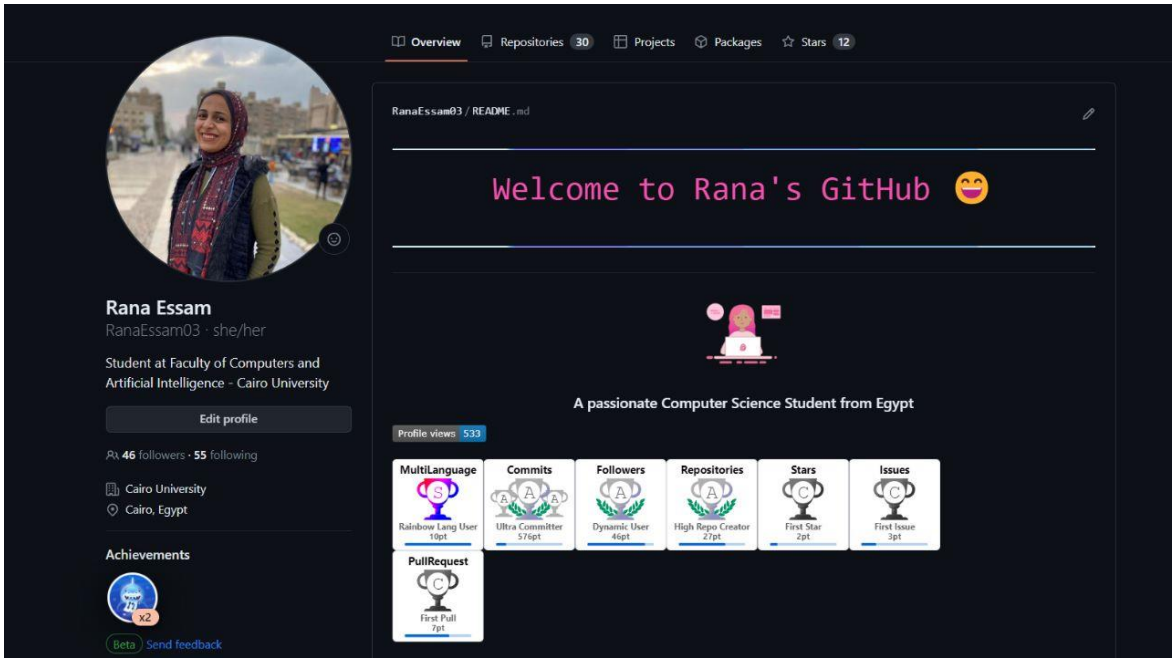


Software Design Specification

VII. GitHub Activity Screenshots:

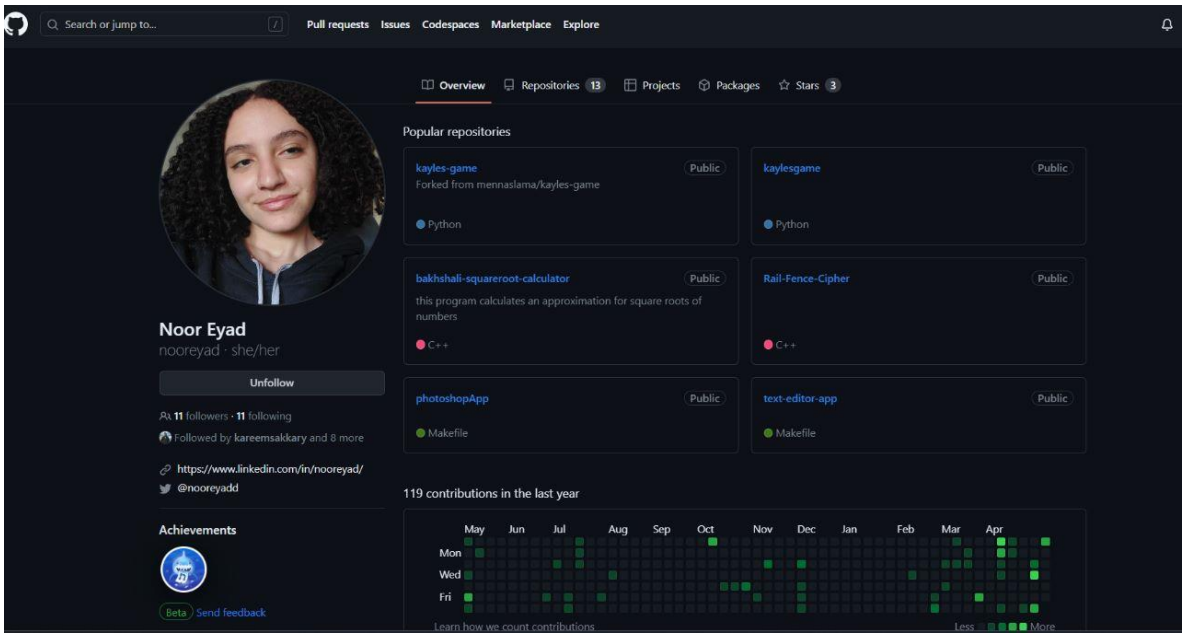


Rana Essam:

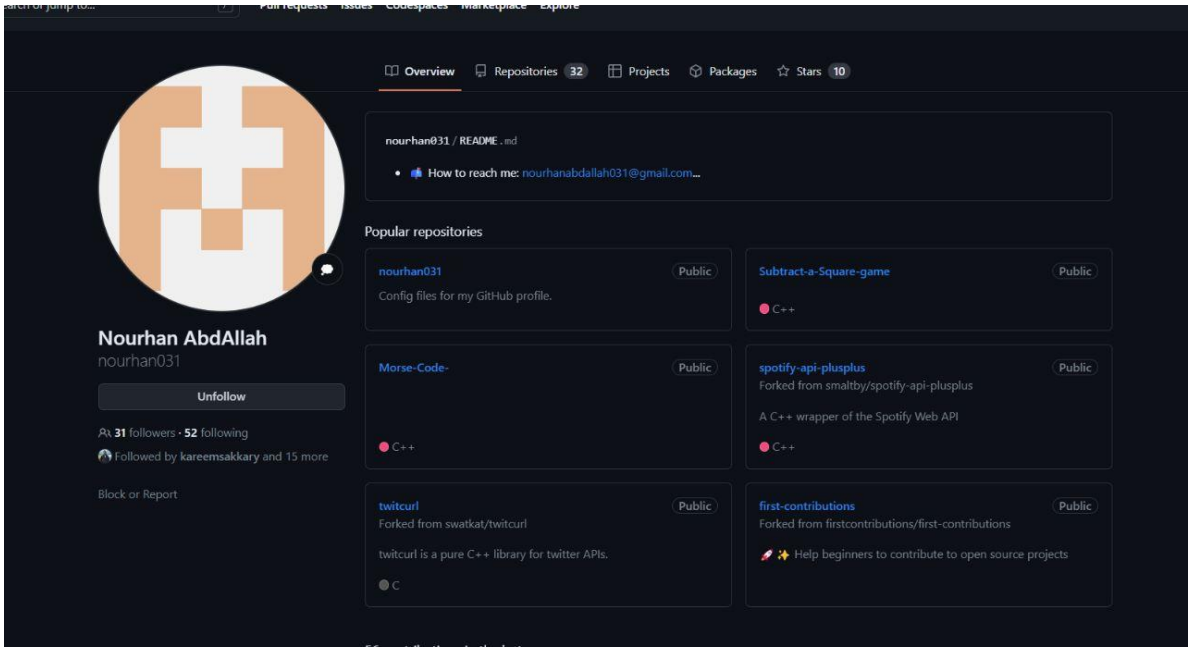


Software Design Specification

Noor Eyad:



Nourhan Abdallah:



CS251: Phase 2 – Toffee Team
Project: Toffee

Software Design Specification

Tools

1. Visual Paradigm
2. Draw.io

Ownership Report

Team member	Owns
Noor Eyad Eissa	• Class diagram • Class description table • Sequence Usage Table • 4, 5, 6 sequence diagram • State diagram
Rana Essam Ibrahim	• Class diagram • Architecture diagram • Sequence Usage Table • 1, 2, 3 sequence diagram • Class description table
Nourhan Abdallah	• 1, 2, 3 Sequence diagrams • Class diagram • Sequence Usage Table • Class description table