

FOOD ORDERING SYSTEM

PROJECT REPORT

**18AIC203J - OBJECT ORIENTED DESIGN AND PROGRAMMING
LABORATORY**

(2018 Regulation)

II Year/ III Semester

Academic Year: 2022 -2023

By

**Viswesh(RA2111003010009), dharineesh(RA2111003010008) and
haasif(RA2111003010010)**

Under the guidance of

Dr. A.Pandian

Assistant Professor

Department of Computational Intelligence



FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

NOVEMBER 2022

BONAFIDE

This is to certify that 18AIC203J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY project report titled “FOOD ORDERING SYSTEM” is the bonafide work of Viswesh(RA2111003010009), dharineesh(RA2111003010008) and haasif(RA2111003010010)

who undertook the task of completing the project within the allotted time.

Signature of the Guide

Dr. A.Pandian

Assistant Professor

Department of CINTEL

SRM Institute of Science and Technology

Signature of the II Year Academic Advisor

Dr. R JAYA

Professor and Head

Department of CINTEL

SRM Institute of Science and Technology

About the course: -

18AIC203J - Object Oriented Design and Programming are 4 credit courses with **L T P C as3-0-2-4** (Tutorial modified as Practical from 2018 Curriculum onwards)

Objectives:

The student should be made to:

- Learn the basics of OOP concepts in C++
- Learn the basics of OOP analysis and design skills.
- Be exposed to the UML design diagrams.
- Be familiar with the various testing techniques

Course Learning Rationale (CLR): The purpose of learning this course is to:

- 1.Utilize class and build domain model for real-time programs
- 2.Utilize method overloading and operator overloading for real-time application development programs
- 3.Utilize inline, friend and virtual functions and create application development programs
- 4.Utilize exceptional handling and collections for real-time object-oriented programming applications
- 5.Construct UML component diagram and deployment diagram for design of applications
- 6.Create programs using object-oriented approach and design methodologies for real-time application development

Course Learning Outcomes (CLO): At the end of this course, learners will be able to:

- 1.Identify the class and build domain model
- 2.Construct programs using method overloading and operator overloading
- 3.Create programs using inline, friend and virtual functions, construct programs using standard templates
- 4.Construct programs using exceptional handling and collections
- 5.Create UML component diagram and deployment diagram
- 6.Create programs using object-oriented approach and design methodologies

Table 1: Rubrics for Laboratory Exercises

(Internal Mark Split-up: - As per Curriculum)

CLAP-1	5=(2(E-lab Completion) + 2(Simple Exercises)(from CodeZinger, and any other coding platform) + 1(HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-2	7.5=(2.0(E-lab Completion)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-3	7.5=(2.0(E-lab Completion(80 Pgms)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	2 Mark - E-lab Completion 80 Program Completion from 10 Session (Each session min 8 program) 2 Mark - Code to UML conversion GCR Exercises 3.5 Mark - Hacker Rank Coding challenge completion
CLAP-4	5= 3 (Model Practical) + 2(Oral Viva)	<ul style="list-style-type: none"> • 3 Mark – Model Test • 2 Mark – Oral Viva
Total	25	

COURSE ASSESSMENT PLAN FOR OODP LAB

S.No	List of Experiments	Course Learning Outcomes (CLO)	Blooms Level	PI	No of Programs in each session
1.	Implementation of I/O Operations in C++	CLO-1	Understand	2.8.1	10
2.	Implementation of Classes and Objects in C++	CLO-1	Apply	2.6.1	10
3.	To develop a problem statement. 1. From the problem statement, Identify Use Cases and develop the Use Case model. 2. From the problem statement, Identify the conceptual classes and develop a domain model with a UML Class diagram.	CLO-1	Analysis	4.6.1	Mini Project Given
4.	Implementation of Constructor Overloading and Method Overloading in C++	CLO-2	Apply	2.6.1	10
5.	Implementation of Operator Overloading in C++	CLO-2	Apply	2.6.1	10
6.	Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams and Collaboration diagrams	CLO-2	Analysis	4.6.1	Mini Project Given
7.	Implementation of Inheritance concepts in C++	CLO-3	Apply	2.6.1	10
8.	Implementation of Virtual function & interface concepts in C++	CLO-3	Apply	2.6.1	10
9.	Using the identified scenarios in your project, draw relevant state charts and activity diagrams.	CLO-3	Analysis	4.6.1	Mini Project Given
10.	Implementation of Templates in C++	CLO-3	Apply	2.6.1	10
11.	Implementation of Exception of Handling in C++	CLO-4	Apply	2.6.1	10
12.	Identify the User Interface, Domain objects, and Technical Services. Draw the partial layered, logical architecture diagram with UML package diagram notation such as Component Diagram, Deployment Diagram.	CLO-5	Analysis	4.6.1	Mini Project Given
13.	Implementation of STL Containers in C++	CLO-6	Apply	2.6.1	10
14.	Implementation of STL associate containers and algorithms in C++	CLO-6	Apply	2.6.1	10
15.	Implementation of Streams and File Handling in C++	CLO-6	Apply	2.6.1	10

LIST OF EXPERIMENTS FOR UML DESIGN AND MODELLING:

To develop a mini-project by following the exercises listed below.

1. To develop a problem statement.
2. Identify Use Cases and develop the Use Case model.
3. Identify the conceptual classes and develop a domain model with UML Class diagram.
4. Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams.
5. Draw relevant state charts and activity diagrams.
6. Identify the User Interface, Domain objects, and Technical services. Draw the partial layered, logical architecture diagram with UML package diagram notation.

Suggested Software Tools for UML:

Star UML, Rational Suite, Argo UML (or) equivalent, Eclipse IDE and Junit

ABSTRACT

An Online Food Ordering System is proposed here which simplifies the food ordering process. The proposed system shows a user interface and update the menu with all available options so that it eases the customer work. Customer can choose more than one item to make an order and can view order details before logging off. The order confirmation is sent to the customer. The order is placed in the queue and updated in the database and returned in real time. This system assists the staff to go through the orders in real time and process it efficiently with minimal errors. One of the ways to reduce this expense is to use modern technology to replace some of the jobs done by human beings and make machines do the work. Here we propose an “Online Food Ordering System” that has been designed for Fast Food restaurant, Take-Out, or College Cafeterias. The system can also be used in any food delivery industry. This simplifies the process of food ordering for both the customer and the restaurant, as the entire process of taking orders is automated.

Using the application, the end users register online, read the E-menu card, and select the food from the e-menu card to order food online. Once the customer selects the required food item the chef will be able to see the results on the screen and start processing the food. This application nullifies the need of a waiter or reduces the workload of the waiter. The advantage is that in a crowded restaurant there will be chances that the waiters are overloaded with orders and they are unable to meet the requirements of the customer in a satisfactory manner. Therefore, by using this application, the users can directly place the order for food to the chef online. The system also enables the restaurant to know the items available in real time and make changes to their food and beverage inventory based on the orders placed and the orders completed.

INDEX

S.No	PROJECT CONTENT
1	Project Scope
2	Problem statement
3	Overall Description
4	Functional Requirements
5	Use Case Diagram
6	Class Diagrams with Description
7	Sequence Diagrams
8	Collaboration Diagrams
9	Package Diagram
10	Component Diagram
11	Deployment Diagram
12	Coding
13	Result and Output Screen
14	Conclusion
15	Reference

FOOD ORDERING SYSTEM

AIM:

To analyze, design and develop code for FOOD ORDERING SYSTEM using Star UML.

PROJECT SCOPE:

The proposed system allows customers to successfully order the food, can avoid the long queues at the counter, with a reasonable speed of execution and maximum throughput.

PROBLEM STATEMENT:

The main objective of “*MOVIE BOOKING SYSTEM*” is to provide a convenient way to the users to book the tickets for cinema hall online, through which they can book tickets anytime and anywhere.

Our app will also give the facility to let the user select any timing slot and they have the authority to choose any seat according to their convenience. This app is basically made for providing customer an anytime and anywhere service for booking the seat in the cinema hall and to gather information about the movies online. The user can easily be able to know about the movies released and then make the choices.

OVERVIEW

This app adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner. The software controls redundancy so that no two users can access the same seat at the same time and transactions should be independent.

PROJECT DESCRIPTION

Product Perspective

1. If the actor has a role of an admin then he/she would have access to accept user's request and make required updation in the database register, login, change password, payment and cancel ticket.
2. If the actor has a role of user then he/she would have access to register, login, change password, book movie ticket, order snack, payment and cancel ticket.

System Interfaces

This system does have one interface with online payment gateway of the existing systems.

System Specifications

- **Hardware Requirements –**

- Intel Pentium and Celeron class processor
- Processor Speed – 1.2 GHz or above
- RAM - 512 MB
- HDD - 40 GB
- Monitor-14"SVGA
- Printer
- Mouse- Normal
- Keyboard- Normal

- **Software Requirements –**

- **Front-end Tool:** - Microsoft ASP.NET 2.0
 - User friendly
 - Low Cost Solution
 - GUI feature
 - Better designing aspects
- **Back-end Tool:** - Microsoft SQL Server 2005
 - Security
 - Portability
 - Quality
 - Efficiency
 - Maintainability
- **Platform:** - Windows platform like: 2000 professional, XP & Vista // windows 10

DATA MODELING

ACTORS

- Customer
- Admin
- Restaurant Employee

USECASES:

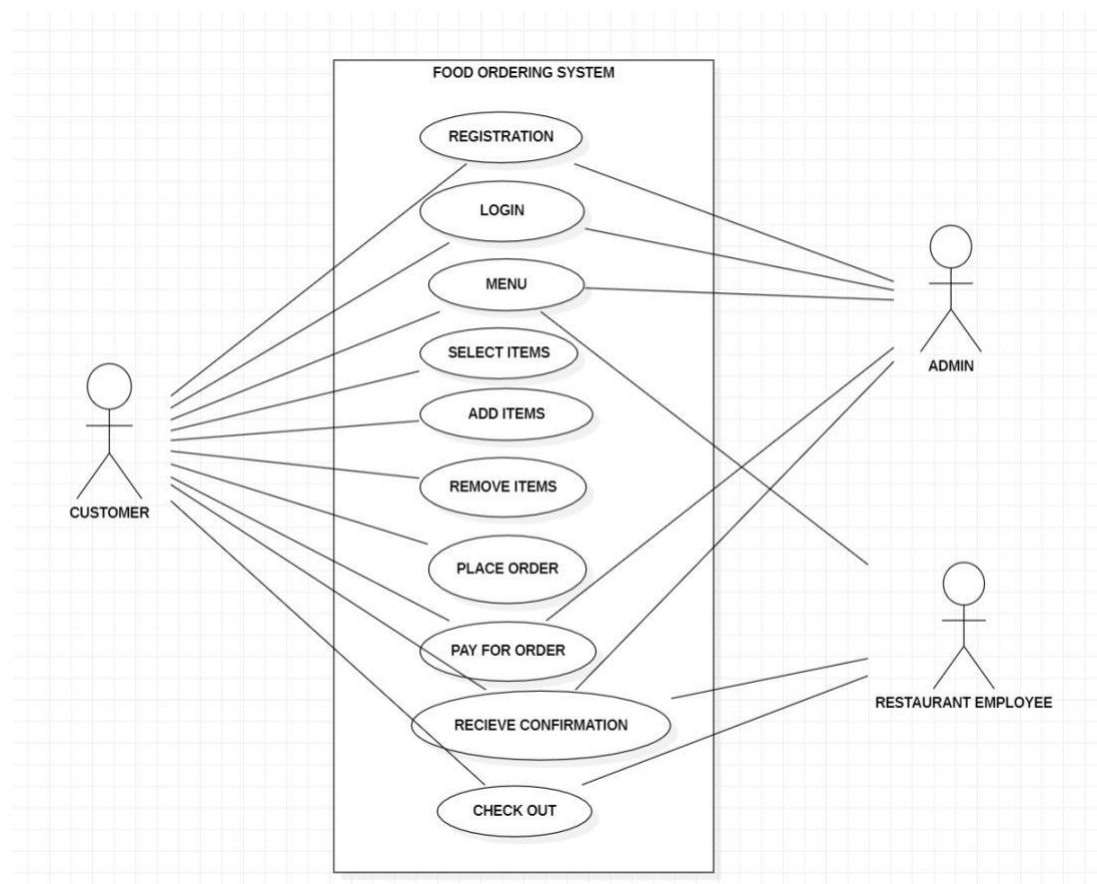
1. Registration
2. Login
3. Menu
4. Select items
5. Add items
6. Remove items
7. Place order
8. Pay for order
9. Receive confirmation
10. Check out

DESCRIPTIONS

USE CASE DIAGRAM

- Draw and drop the actors and use case from toolbox in the diagram window.
- Associate the Use cases and actors.

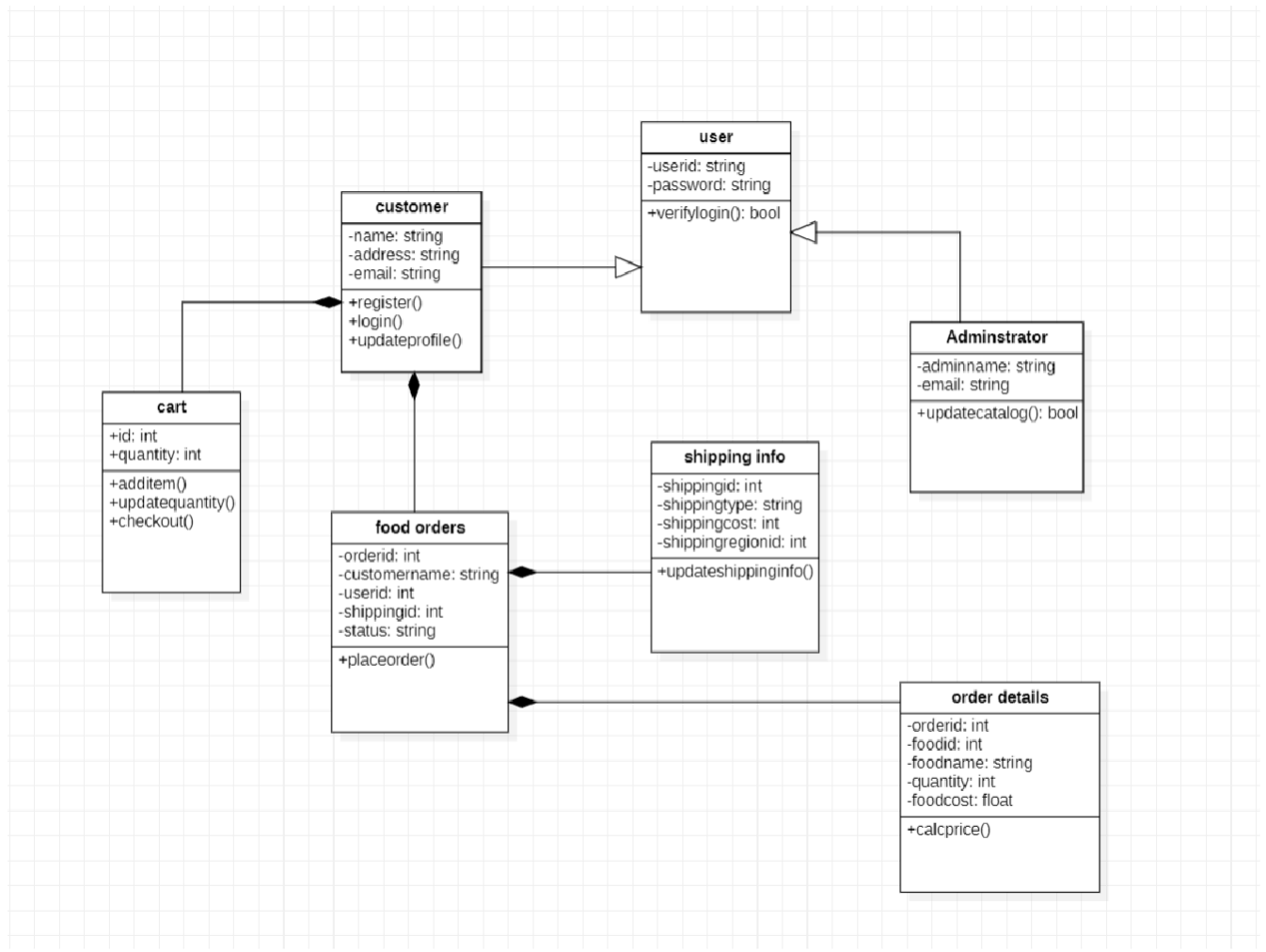
USE CASE DIAGRAM



CLASS DIAGRAM

- The class diagram shows a set of classes, interfaces and collaboration and their relationship.
- Class diagram shows the attributes and operation of a class and constraints that apply to the way the object are connected.

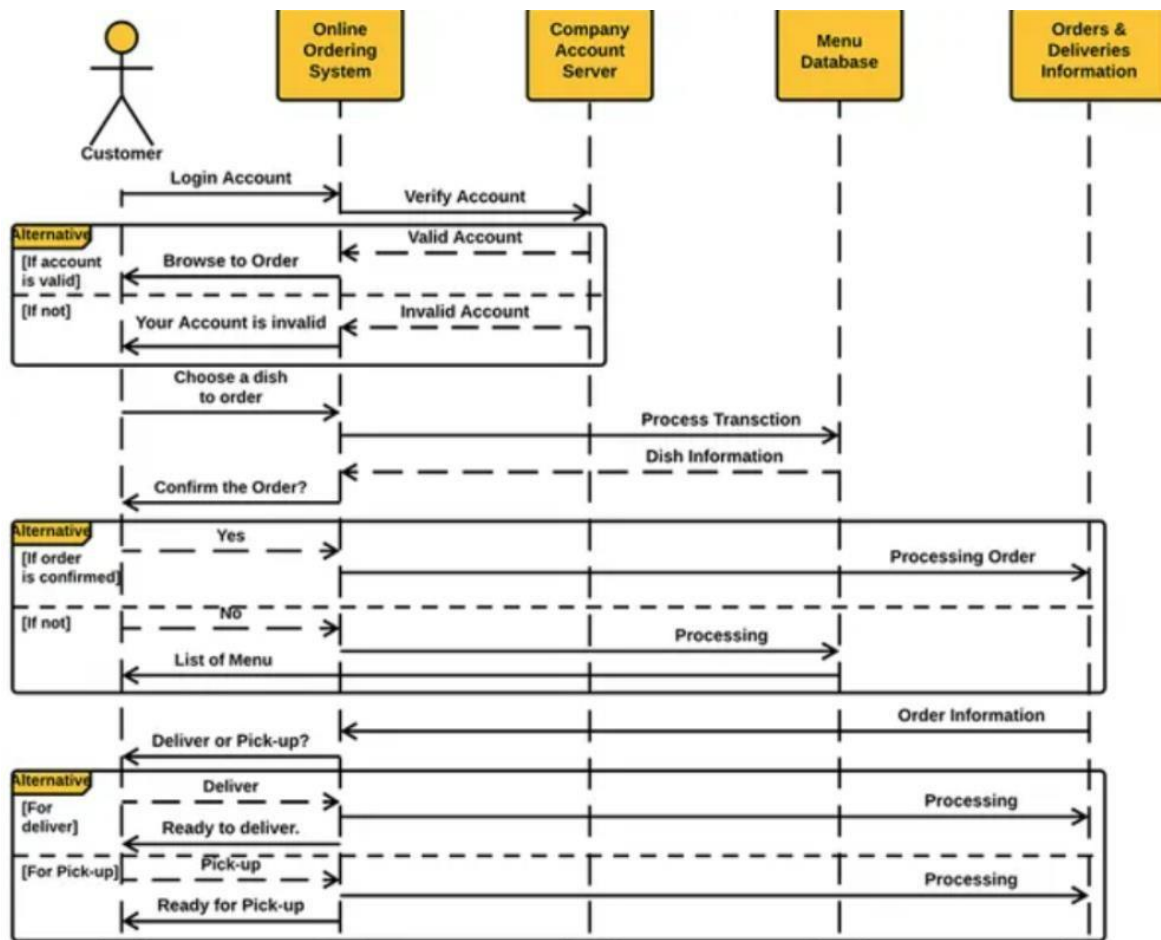
CLASS DIAGRAM



SEQUENCE DIAGRAM

- A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence – what happens first, what happens next.
- Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

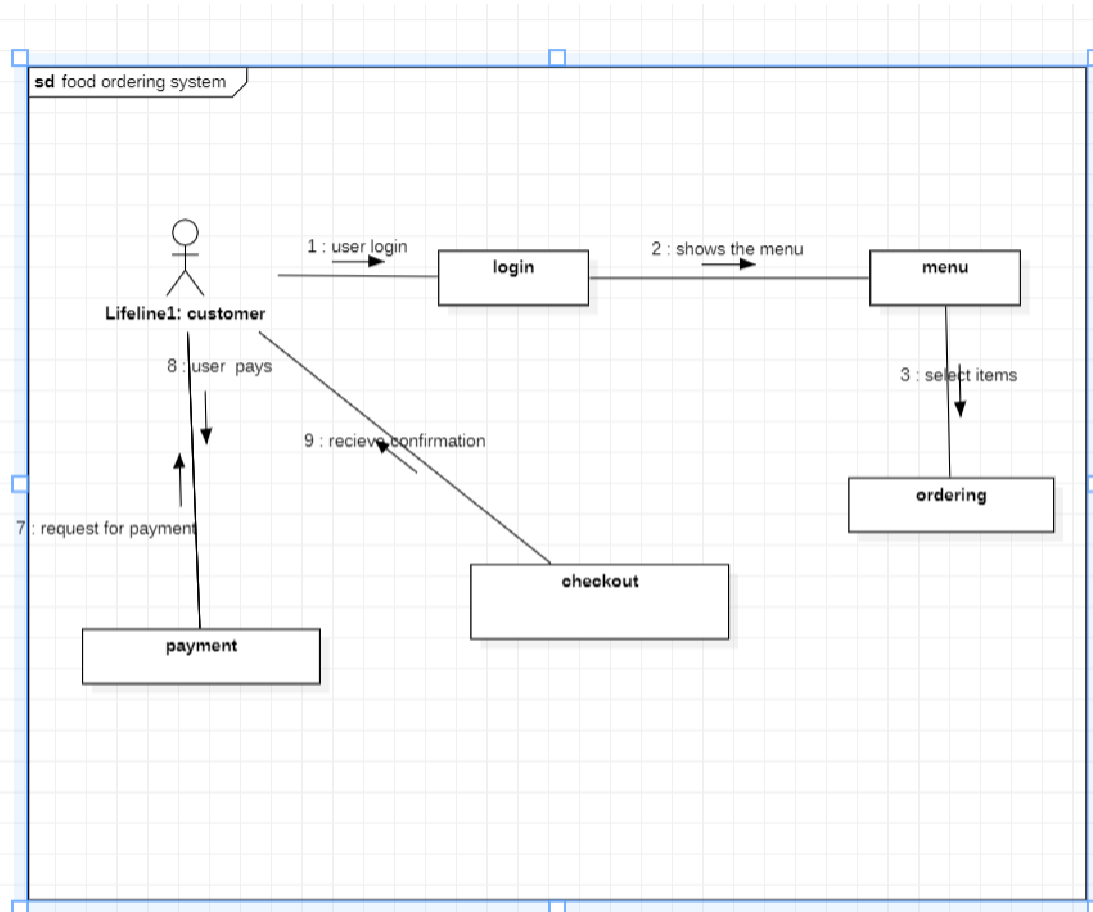
SEQUENCE DIAGRAM



COLLABARATION DIAGRAM

- The collaboration diagram can be drawn from the sequential diagram.
- The collaboration diagram shows the overall flow of central towards each phase of the class

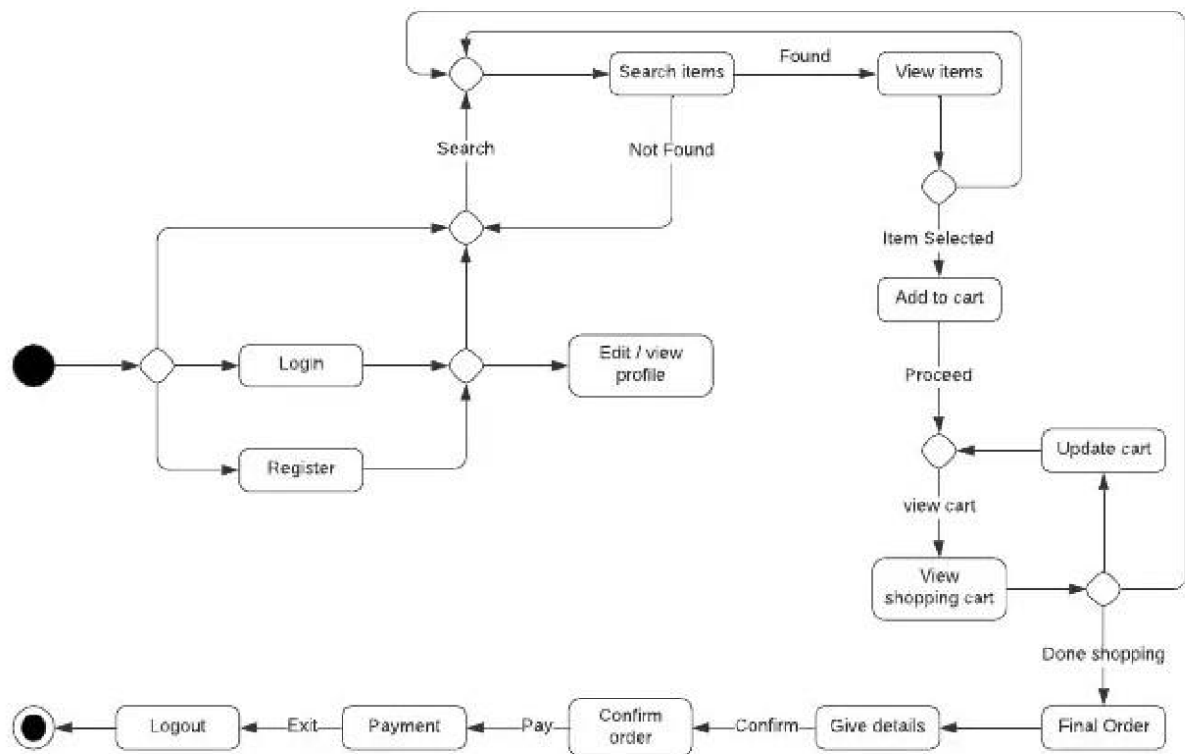
COLLABARATION DIAGRAM



STATECHART DIAGRAM

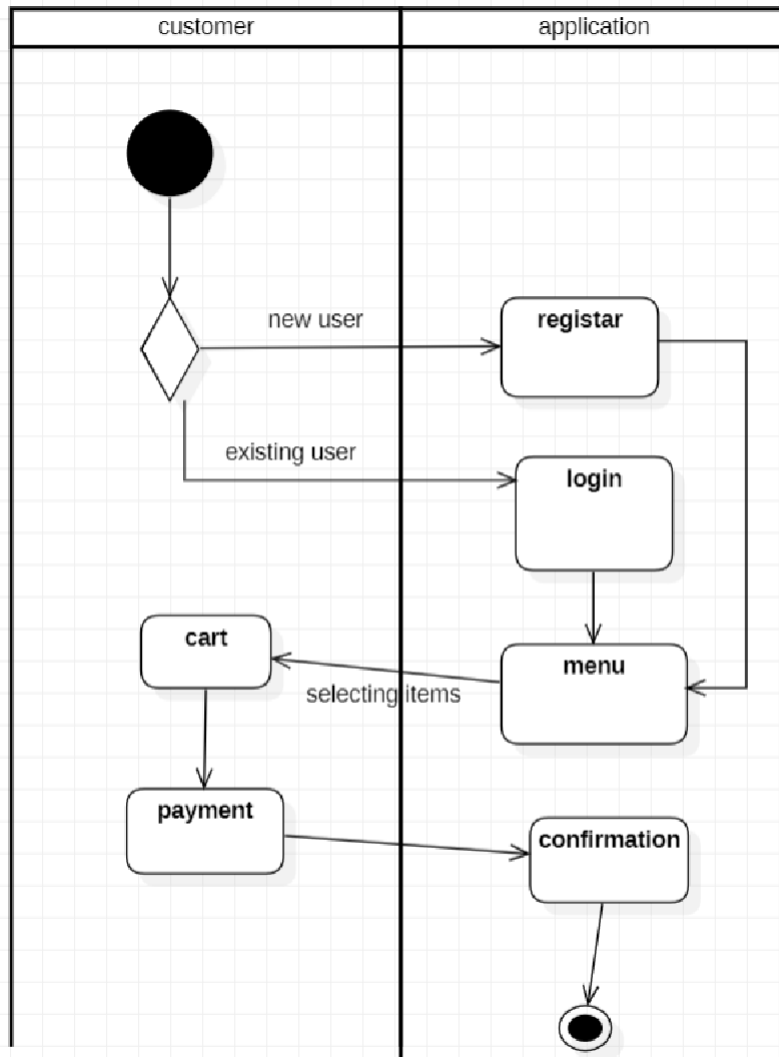
- The State chart diagram describes the whole process doing in the project.
- The state chart diagram shows the control of the full project.

STATE CHART DIAGRAM

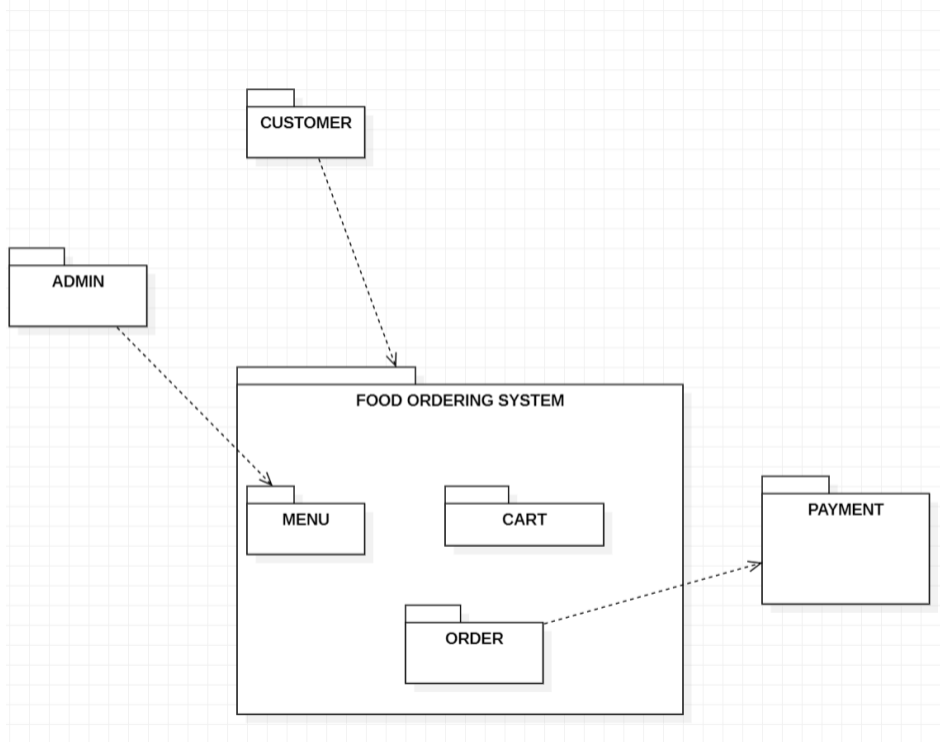


ACTIVITY DIAGRAM

- The activity diagram describes the sequencing of activities with support for both conditional and parallel behavior.
- The various controls are placed into the diagram window to create activity diagram.



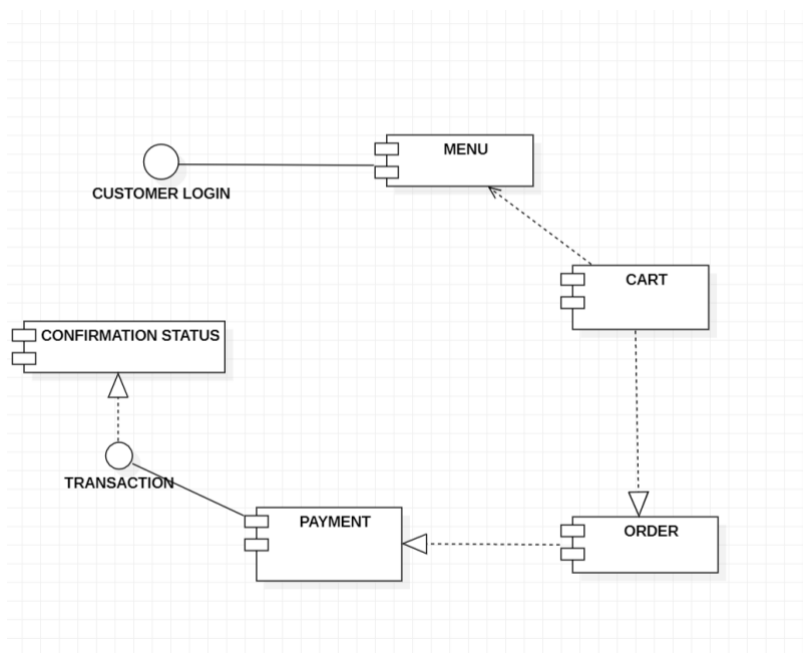
PACKAGE DIAGRAM



COMPONENT DIAGRAM

- It displays the high level packaged structure of the code itself.
- Dependencies among components are shown, including source code components, binary code components and executable components.

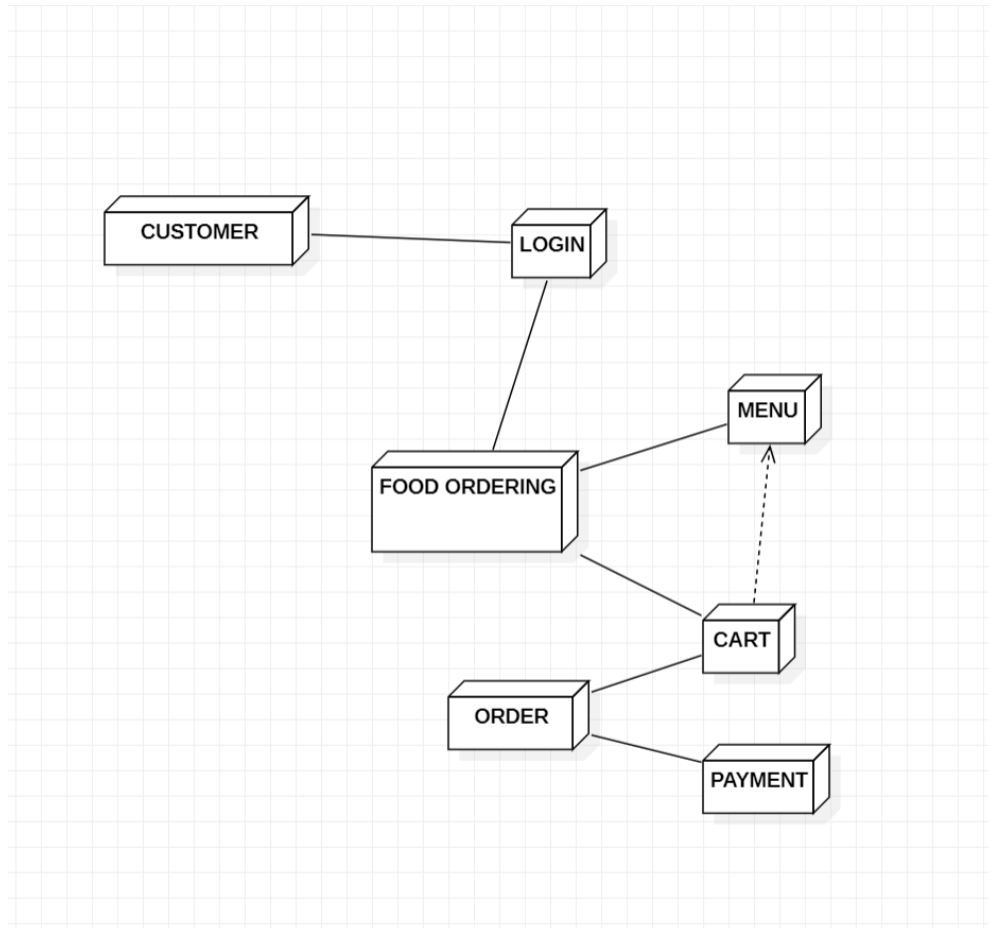
COMPONENT DIAGRAM



DEPLOYMENT DIAGRAM

- A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.
- A deployment diagram in the [Unified Modeling Language](#) models the *physical* deployment of [artifacts](#) on [nodes](#).

DEPLOYMENT DIAGRAM



CODE EXTRACTION FROM UML:

```
/**  
 * Project FOOD ORDERING SYSTEM  
 */
```

```
#include "Adminstrator.h"
```

```
/**  
 * Adminstrator implementation  
 */
```

```
/**  
 * @return bool  
 */
```

```
bool Adminstrator::updatecatalog() {  
    return false;  
}
```

```
#include "App user.h"
```

```
/**  
 * App user implementation  
 *
```

```
 * fsds  
 */
```

```
#include "cart.h"
```

```
/**  
 * cart implementation  
 */
```

```
void cart::additem() {
```

```
}
```

```
void cart::updatequantity() {
```

```
}
```

```
void cart::checkout() {
```

```

}

#include "Class1.h"
#include "customer.h"
/**
 * customer implementation
 */
void customer::register() {
}
void customer::login() {
}
void customer::updateprofile() {
}
#include "food cart.h"
/**
 * food cart implementation
 */
void food cart::additem() {
}
void food cart::updatequantity() {
}
void food cart::checkout() {
}
#include "food orders.h"

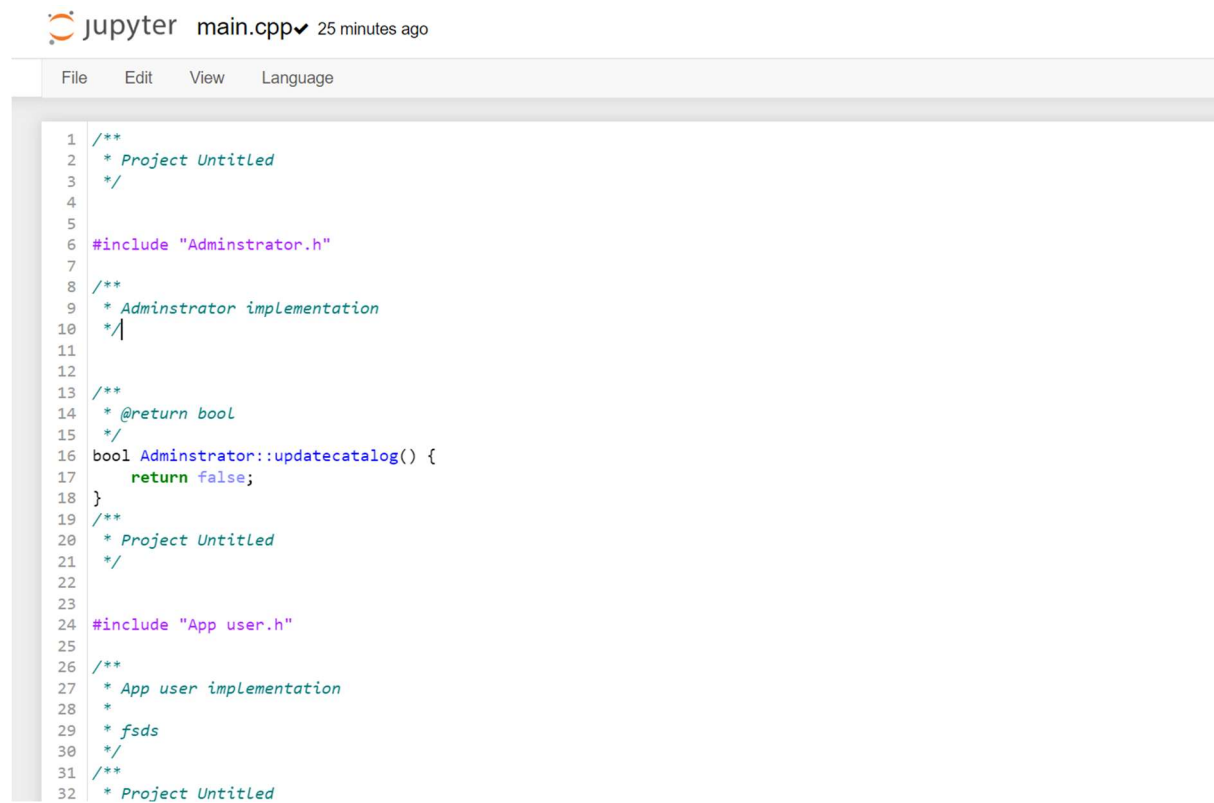
/**
 * food orders implementation
 */
void food orders::placeorder() {
}
#include "order details.h"
void order details::calcprice() {
}
#include "shipping info.h"
void shipping info::updateshippinginfo() {

```

```
}  
#include "user.h"  
/**  
 * @return bool  
 */  
bool user::verifylogin() {  
    return false;  
}
```

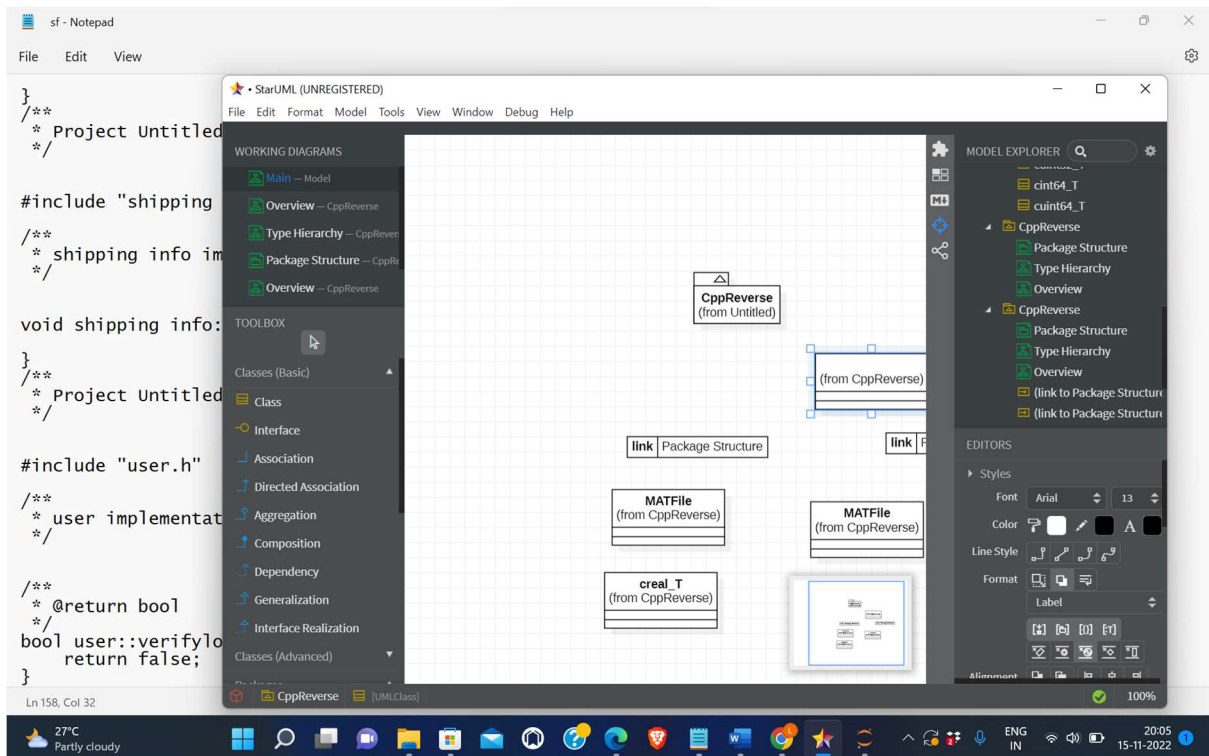
C++ Code to UML Conversion

CODE :



The image shows a Jupyter Notebook interface with a file named `main.cpp` opened 25 minutes ago. The interface includes a menu bar with 'File', 'Edit', 'View', and 'Language' options. The code editor displays the following C++ code:

```
1  /**
2   * Project Untitled
3   */
4
5
6  #include "Adminstrator.h"
7
8  /**
9   * Adminstrator implementation
10  */
11
12
13  /**
14   * @return bool
15   */
16  bool Adminstrator::updatecatalog() {
17      return false;
18  }
19  /**
20   * Project Untitled
21   */
22
23
24  #include "App user.h"
25
26  /**
27   * App user implementation
28   *
29   * fsds
30   */
31  /**
32   * Project Untitled
```



Output Screen

```
***** WELCOME TO FOOD ORDERING SYSTEM *****

-----START YOUR ORDER-----

*****
[1] Veggie Supreme (Rs.480)
[4] Crispy Chicken (Rs.520)
[7] Tetrazzini (Rs.420)
[10] Ham Burger (Rs.390)
[13] Fajitas (Rs.335)
*****

*****
[2] Exotica Pizza(Rs.440)
[5] Spaghetti (Rs.350)
[8] Double Cheese (Rs.540)
[11] Margherita Pizza(Rs.525)
[14] Tempura (Rs.324)
*****

*****
[3] Chicken Sizzler (Rs.580)
[6] Country Feast (Rs.400)
[9] Makiushi (Rs.548)
[12] Fish 'n' Chips (Rs.425)
[15] Hot Dog (Rs.360)
*****

[16] EXIT

*****

ENTER YOUR ORDER (one order at a time): _
```

RESULT

Thus, the UML diagrams for a food ordering system were created and the corresponding code is generated.

Conclusion

By using star UML, we were able to create UML diagrams and generate code from it thus creating a food ordering system

References

1. <https://www.swiggy.com/> {reference}
2. <https://staruml.io>
3. <https://www.youtube.com/watch?v=rTVVyjq-ozM>
4. <https://code-projects.org/food-ordering-system-in-c-with-source-code/>
5. <https://www.javatpoint.com/>
6. <https://www.lawinsider.com/dictionary/project-results>