**CANTEEN MANAGEMENT SYSTEM**

**PROJECT REPORT**

**18CSC202J/ 18AIC203J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY**

**(2018 Regulation)**

**II Year/ III Semester**

**Academic Year: 2022 -2023**

By

**REETHIKA KOTHAMASU (RA2111026010298)**

**LOHIT JHAJHRIA(RA2111026010303)**

Under the guidance of

**Dr. M. Ferni Ukrit**

**Associate 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 **18CSC202J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY project report** titled “**CANTEEN MANAGEMENT SYSTEM”** is the bonafide work of **REETHIKA KOTHAMASU (RA2111026010298) LOHIT JHAJHRIA (RA2111026010303)**

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

**Signature of the Guide** **Signature of the II Year Academic Advisor**

Dr. M. Ferni Ukrit -------------------------

**Associate Professor** **Assistant Professor**

Department of CINTEL, Department of CINTEL

SRM Institute of Science and Technology SRM Institute of Science and Technology

**About the course:-**

18CSC202J/ 8AIC203J - Object Oriented Design and Programming are 4 credit courses with **L T P C as 3-0-2-4** (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

1. Construct programs using exceptional handling and collections
2. Create UML component diagram and deployment diagram
3. Create programs using object oriented approach and design methodologies

**Table 1: Rubrics for Laboratory Exercises**

(Internal Mark Splitup:- 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) | * **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 EXPERIMNENTS 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:**

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

**ABSTRACT**

An abstract must be a single paragraph in times new roman 14pt with a maximum  
of 300 words.

THE TOPIC CHOSEN: Canteen management system

**CANTEEN MANAGEMENT SYSTEM**

This mini project on the canteen management system depicts how a student/customer goes to the canteen, orders food online on the canteen site and the food gets served in the canteen.

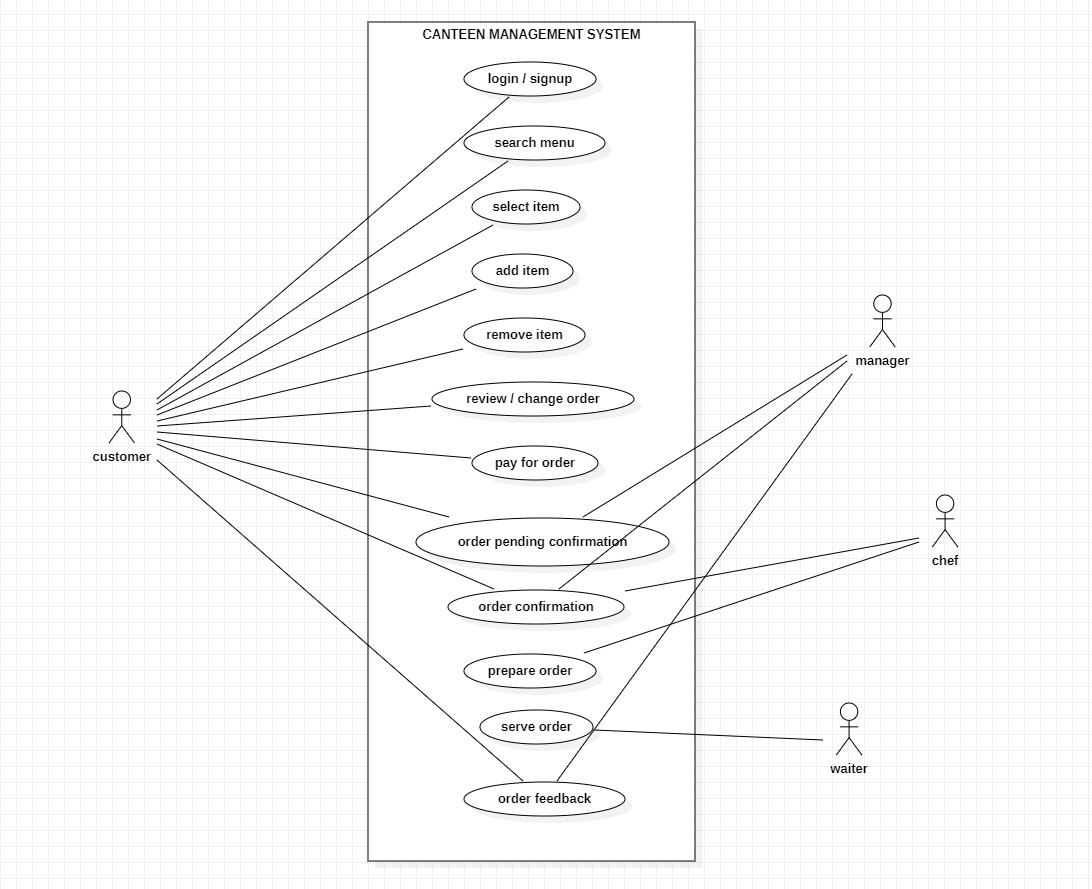
The benefits of this system are as follows:

1. An improved customer service
2. More time to browse the menu
3. More customer control over orders
4. Easier to place larger orders

Using the canteen management system as our topic for the project we drew some required UML diagrams as the Use case, Class, Sequence, Communication, State chart, Activity, Package, Component, and Deployment diagram.

**MODULE DESCRIPTION**

**Use case diagram with explanation**

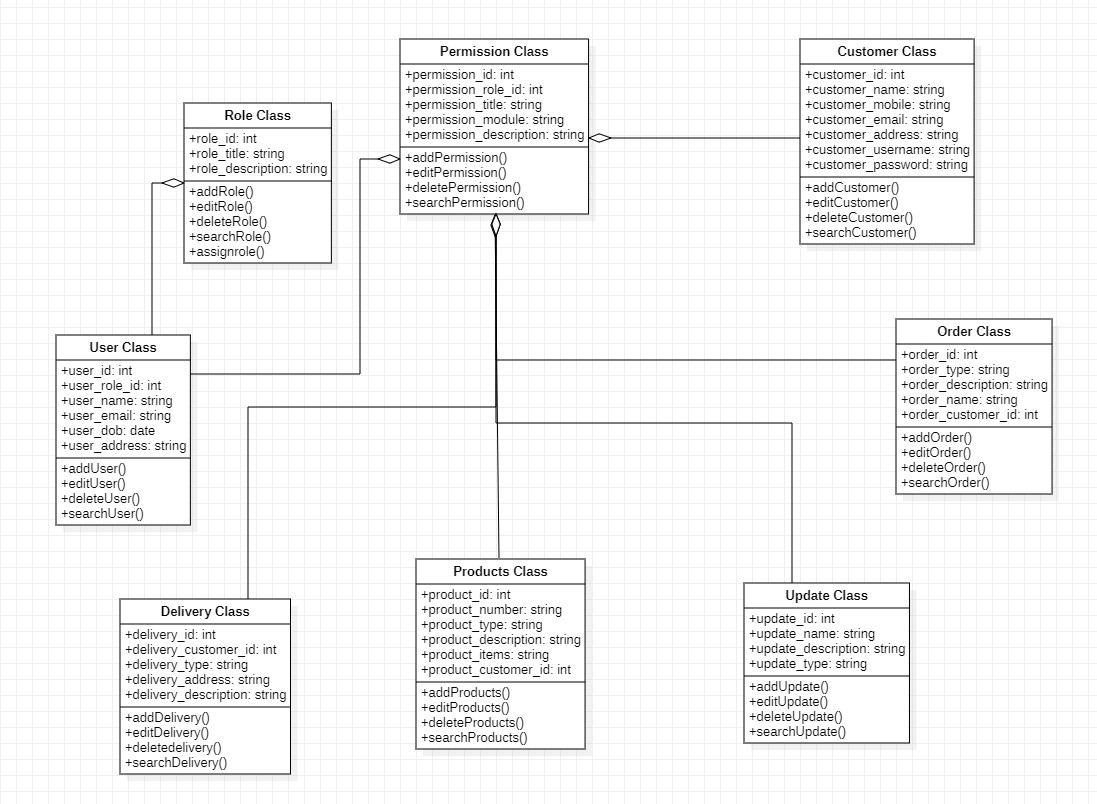
****

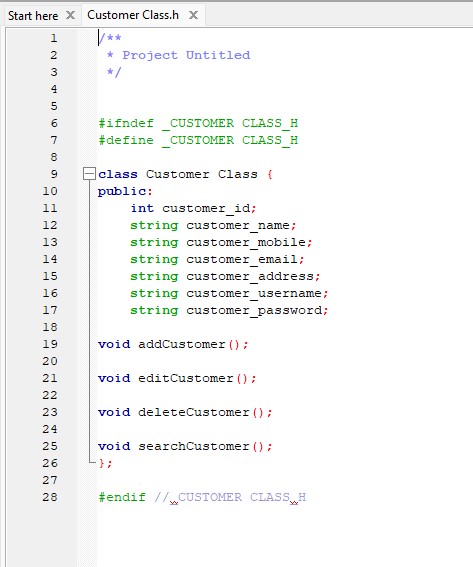
Use case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors.

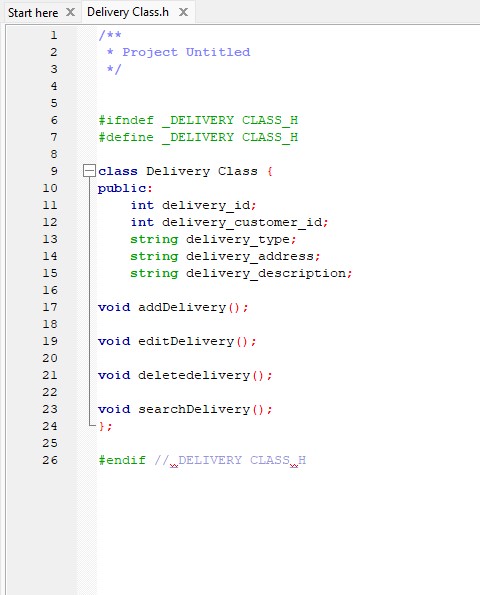
The actors in our diagram are the customer, manager, chef, and waiter. They perform different kinds of use cases such as login, search menu, select item, add item, remove item, review/change order, pay for order, order pending confirmation, order confirmation, prepare the order, serve the order, and order feedback. The actors are associated with the use cases as needed to understand their interactions.

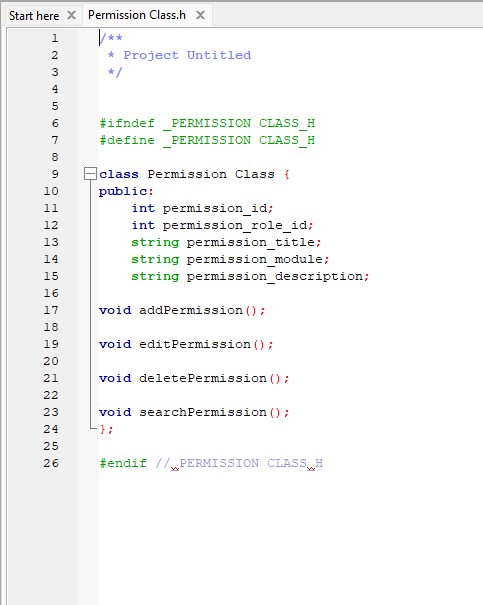
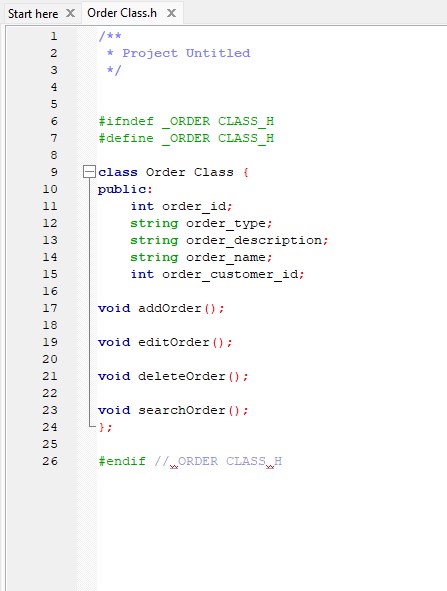
Hence the use case diagram of CANTEEN MANAGEMENT SYSTEM is done successfully.

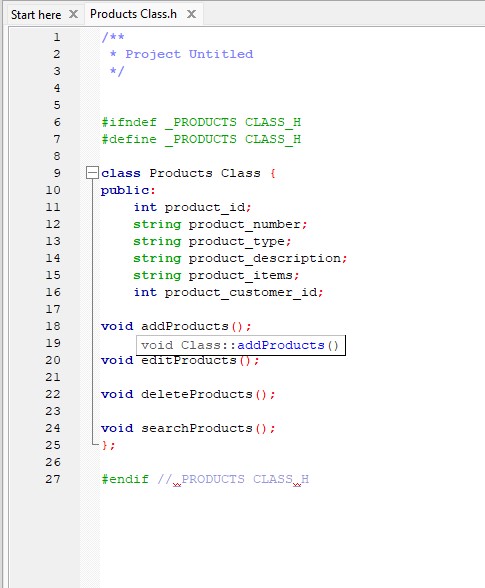
**Class Diagram with Explanation**

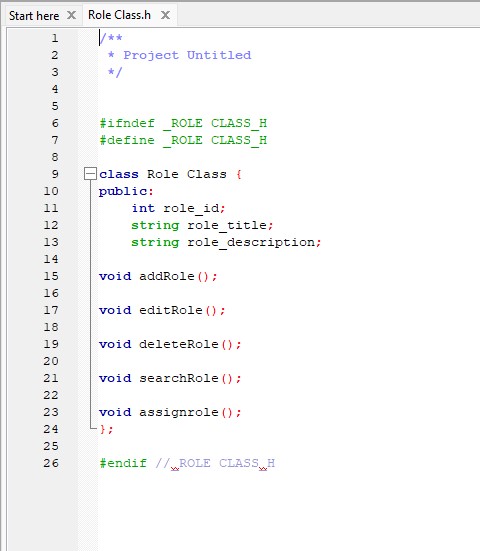
****

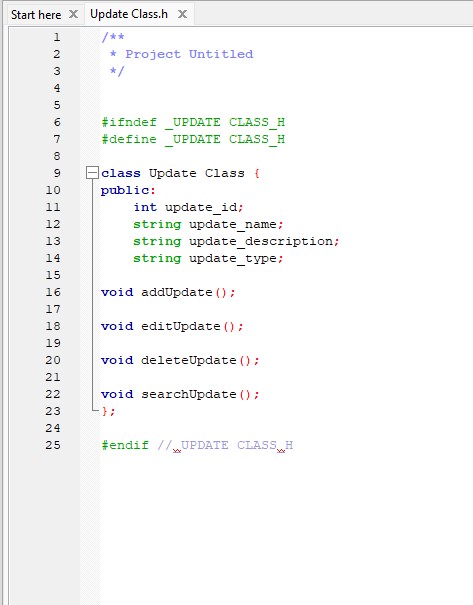


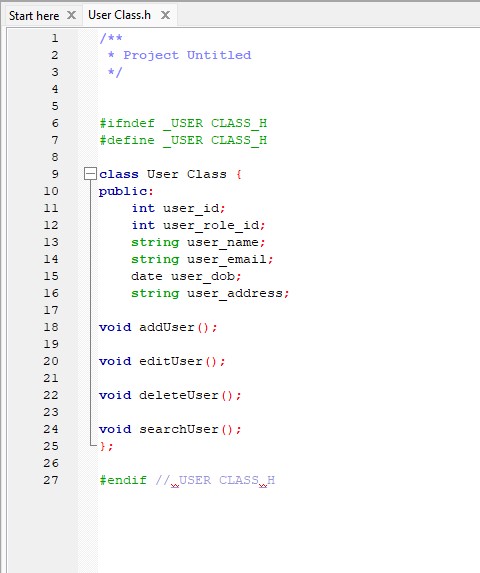












A class diagram is an illustration of the relationships and source code dependencies among classes in the unified modeling language(UML). In this context, a class defines the methods and variables in an object, which is a specified entity in a program or the unit of code representing that entity.

Our diagram illustrates the structure of a model by using attributes, operations, and signals.

In the class diagram, the class depicts information and behavior.

Consists of 3 sections

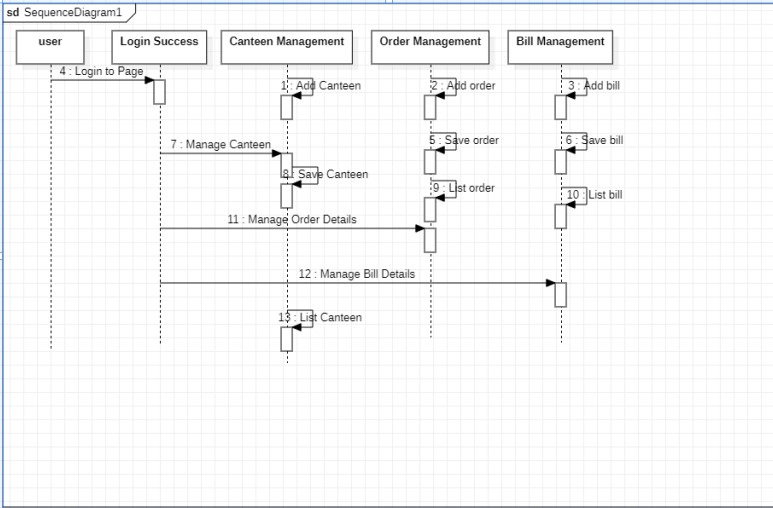
i)top: hold class name

ii)middle: attributes

iii)lower: holds operations or methods

Hence class diagram for the CANTEEN MANAGEMENT SYSTEM was successfully completed.

**Sequence Diagram with Explanation**

****

The sequence diagram is used primarily to show the interactions between objects in the sequential order that those interactions occur. The main purpose of a sequence diagram is to define event sequences that result in some desired outcome.

We have 5 lifelines here namely user, login success, canteen management, order management, and bill management.

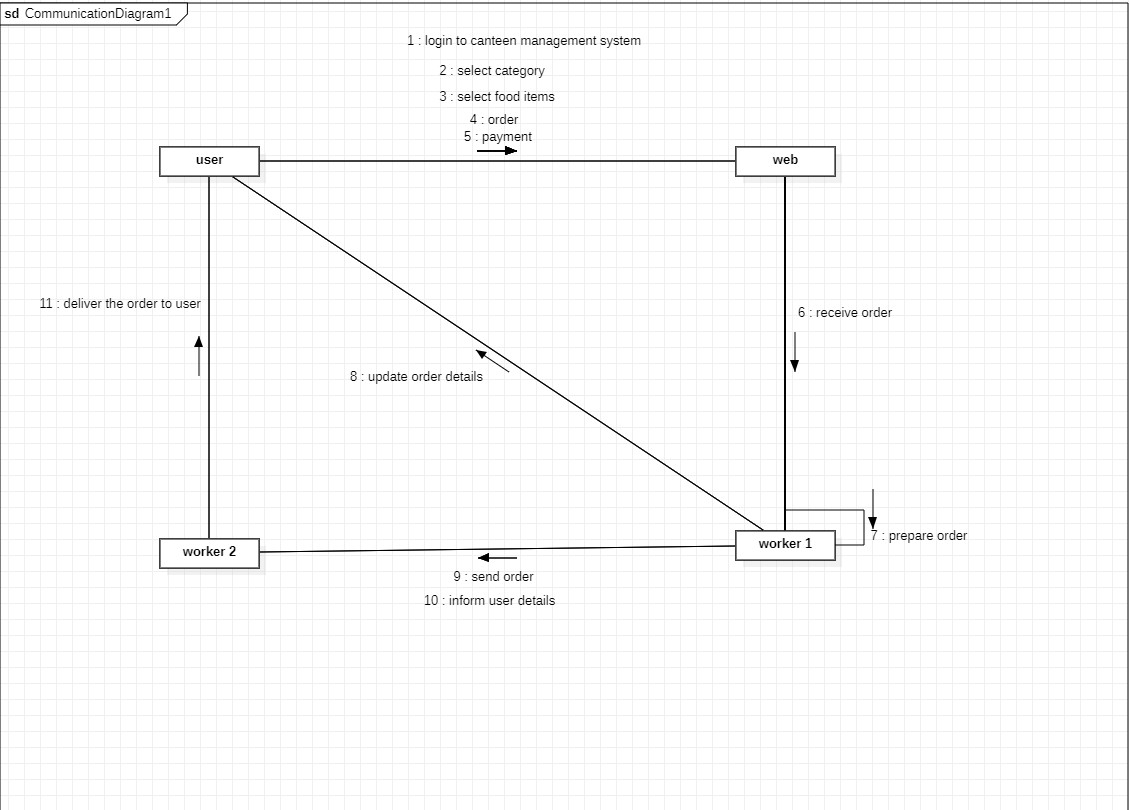
The sequence diagram shows how a user orders food online step by step. Like:

• Admin User can search Food Item, view the description of a selected Food Iten, add Food Item, update Food Item and delete Food Item. Its shows the activity flow of editing, adding, and updating of Canteen

• User will be able to search and generate reports of Bill Employees, Products, and Orders All objects such as (Food Items, Canteen, and Orders) are interlinked

Hence the sequence diagram for CANTEEN MANAGEMENT SYSTEM is successfully done.

**Communication Diagram with Explanation**

****

COLLABORATION DIAGRAM named as COMMUNICATION DIAGRAM between two objects how messages can be communicated can be clearly shown on a communication or collaboration diagram

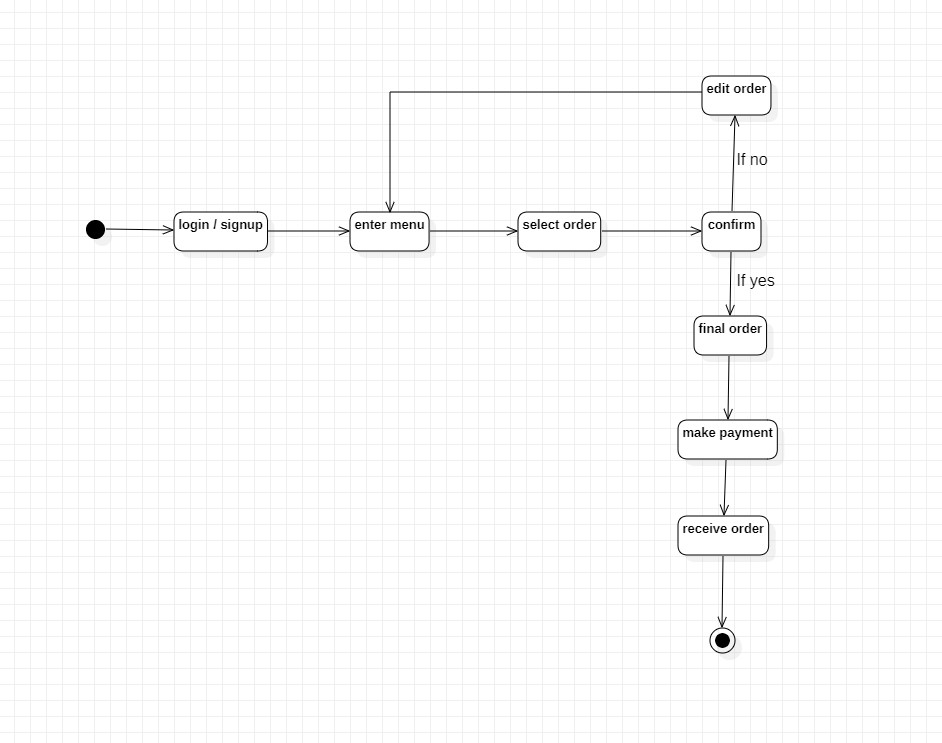
Collaboration diagrams messages are always coming with a number

In the communication diagram elements are organized with space.

Here in our diagram, we use some tools namely LIFELINE, Forward Message, and connector. The below collaboration diagram shows the interaction clearly between the user, web, worker1, and worker2. The collaboration diagram shows the behind steps of how a customer receives the order he orders online.

Hence the collaboration or communication diagram for CANTEEN MANAGEMENT SYSTEM is successfully done.

**State Chart Diagram with Explanation**

****

State diagrams enable you to describe the behavior of objects during the entire life span. In addition, the different states and state changes as well as events causing transactions can be described.

It depicts the whole process of food order process in the canteen online.

starting with the initial state we created 8 states which perform various actions and there are connected with the control flow.

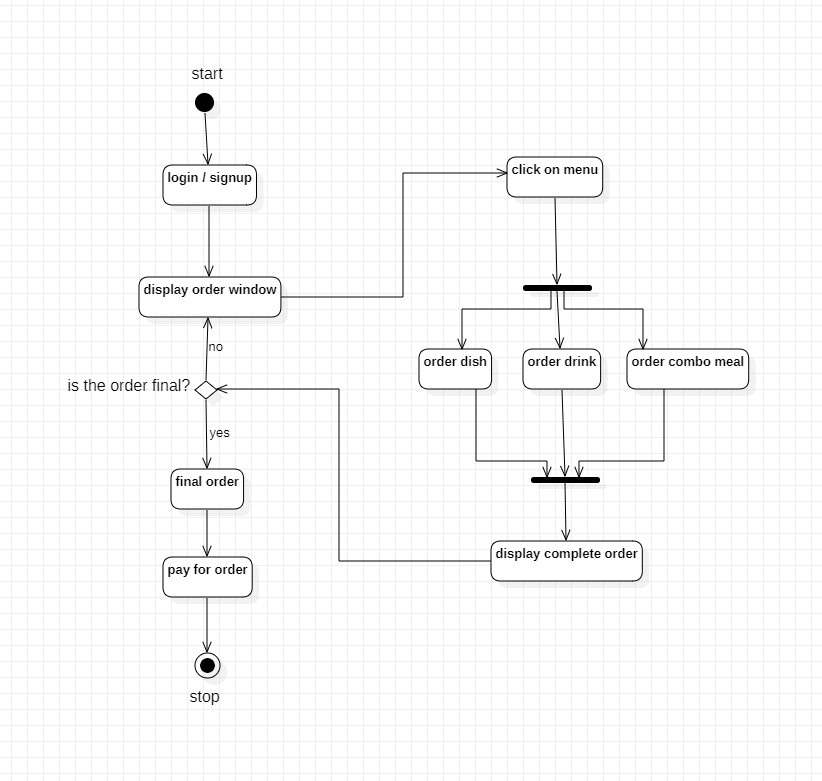
REPORT:

The above state chart diagram depicts the whole process of the food order process online in the canteen in a visual representation.

In general, illustrating use case scenarios in a business context describing how an object moves through various states within a lifetime

Hence we successfully performed STATE CHART DIAGRAM ON the CANTEEN management system using STAR UML.

**Activity Diagram with Explanation**

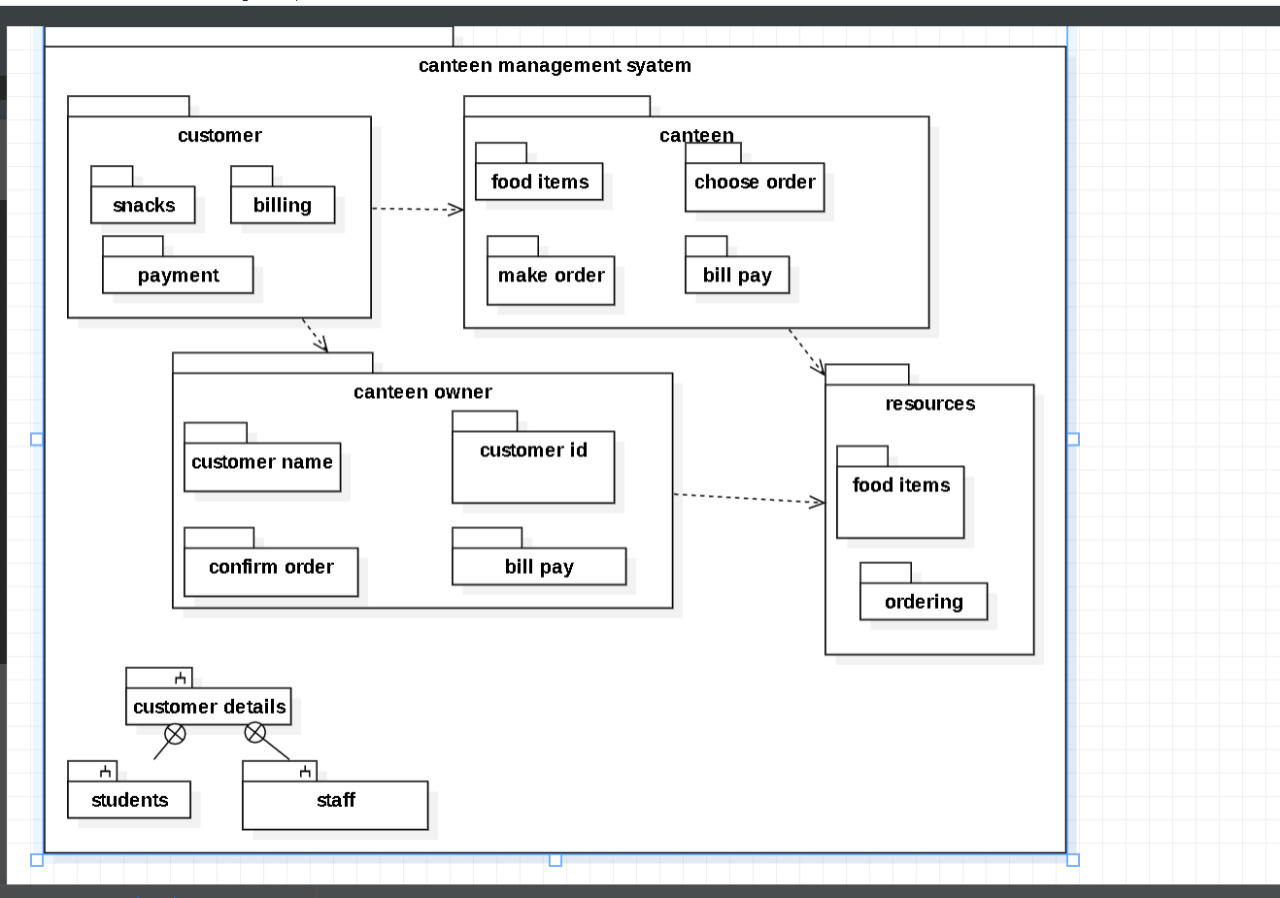
****

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram.

Starting from the initial we used actions that are connected with control flow and used the decision tool for taking decisions and the fork node to divide one action into two and join node to combine 2 actions from the fork node finally we ended the diagram with the Final tool.

The activity diagram describes the steps performed in a UML use case. Illustrate a business process or workflow between users and the system. Simplify and improve any process by clarifying complicated use cases. Model software architecture elements, such as method, function, and operation. We successfully performed and executed the ACTIVITY DIAGRAM for the CANTEEN MANAGEMENT SYSTEM.

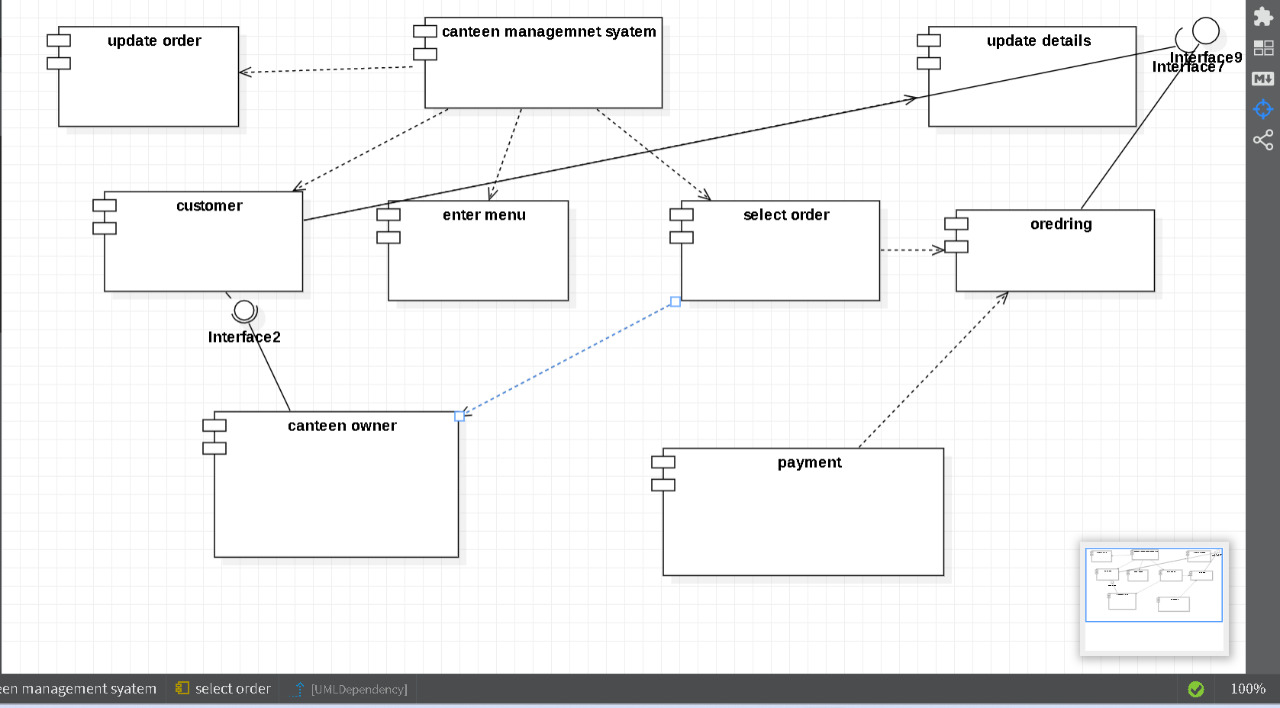
**Package Diagram with Explanation**

****

Package diagrams are structural diagrams used to show the organization and arrangement of various model elements in the form of packages. A package is a method of grouping related elements, such as diagrams, documents, classes, and packages in UML For each UML, a package can contain any element, that is, classes, interfaces, modules, nodes, use cases, diagrams, and other packages grouped into it.

Using the package tool we drew the required packages and using dependency we depicted the relationship between the packages like the operations that are going to be performed. Hence we successfully completed the package diagram for the CANTEEN MANAGEMENT SYSTEM.

**Component Diagram with Explanation**

****

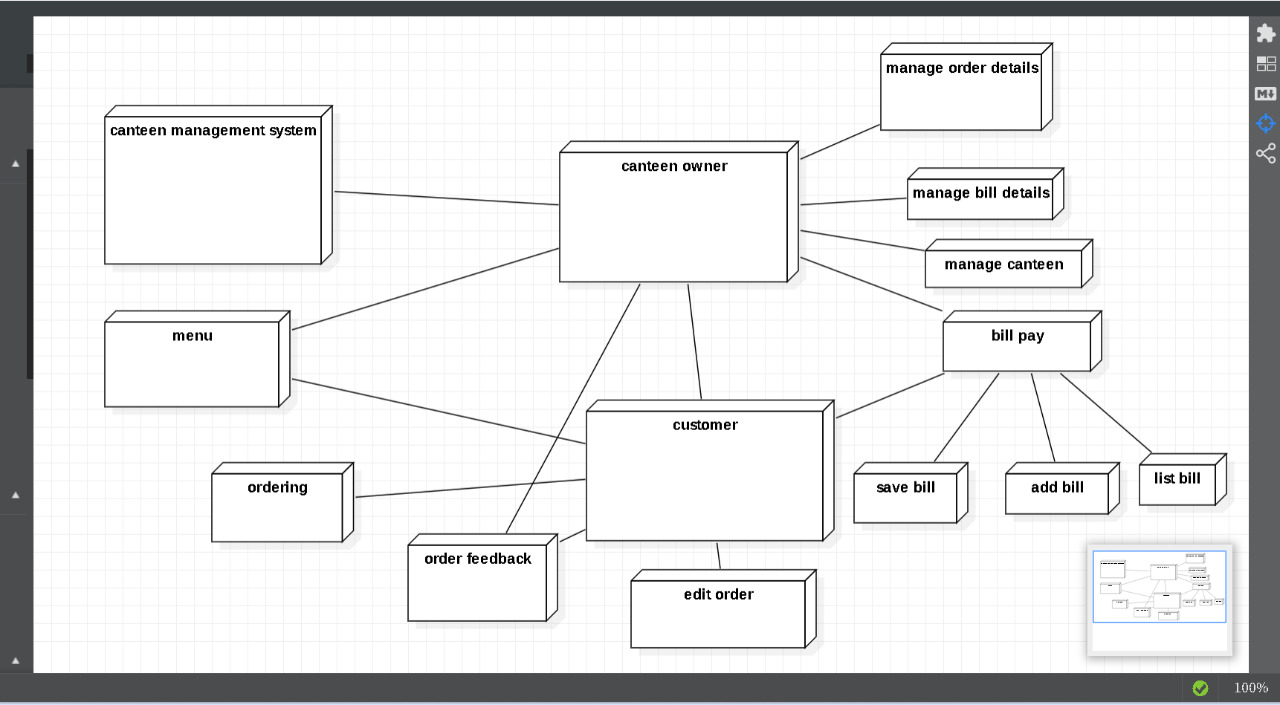
A component diagram, also known as a UML component diagram, **describes the organization and wiring of the physical components in a system**. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.

The dependency between each of the components are shown using dependency arrows.

The purpose of a component diagram is **to show the relationship between different components in a system**.

Hence the component diagram for the CANTEEN MANAGEMENT DIAGRAM is done successfully.

**Deployment Diagram with Explanation**

****

A deployment diagram is **a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them**. Deployment diagrams are typically used to visualize the physical hardware and software of a system.

The communication between each node is depicted using association. The deployment diagram models the physical deployment of artifacts on nodes. It is a kind of structure diagram used in modeling the physical aspects of an object-oriented system. It falls under the structural diagramming family. It is similar to the component diagram whereas the only difference between them is that we use nodes in deployment but components in the component diagram.

**CONCLUSION**

Thus, CANTEEN management system is represented in all forms of UML diagrams both static and dynamic modeling.

**REFERENCES**

<https://creately.com/blog/diagrams/uml-diagram-types-examples/>

<https://www.tutorialspoint.com/uml/uml_standard_diagrams.htm>