UT-1 Solution

Q1. a. What is an Evolutionary model. Explain any one in detail.

Evolutionary model is a combination of Iterative and Incremental model of software development life cycle. Delivering your system in a big bang release, delivering it in incremental process over time is the action done in this model. Some initial requirements and architecture envisioning need to be done. It is better for software products that have their feature sets redefined during development because of user feedback and other factors. The Evolutionary development model divides the development cycle into smaller, incremental waterfall models in which users are able to get access to the product at the end of each cycle. Feedback is provided by the users on the product for the planning stage of the next cycle and the development team responds, often by changing the product, plan or process. Therefore, the software product evolves with time. All the models have the disadvantage that the duration of time from start of the project to the delivery time of a solution is very high.

Evolutionary models are iterative type models.

They allow to develop more complete versions of the software. Following are the evolutionary process

models.

1. The prototyping model
2. The spiral model
3. Concurrent development model

The Prototyping Model

Prototype is defined as first or preliminary form using which other forms are copied or derived.

- Prototype model is a set of general objectives for software.
- It does not identify the requirements like detailed input, output.
- It is software working model of limited functionality.

- In this model, working programs are quickly produced.

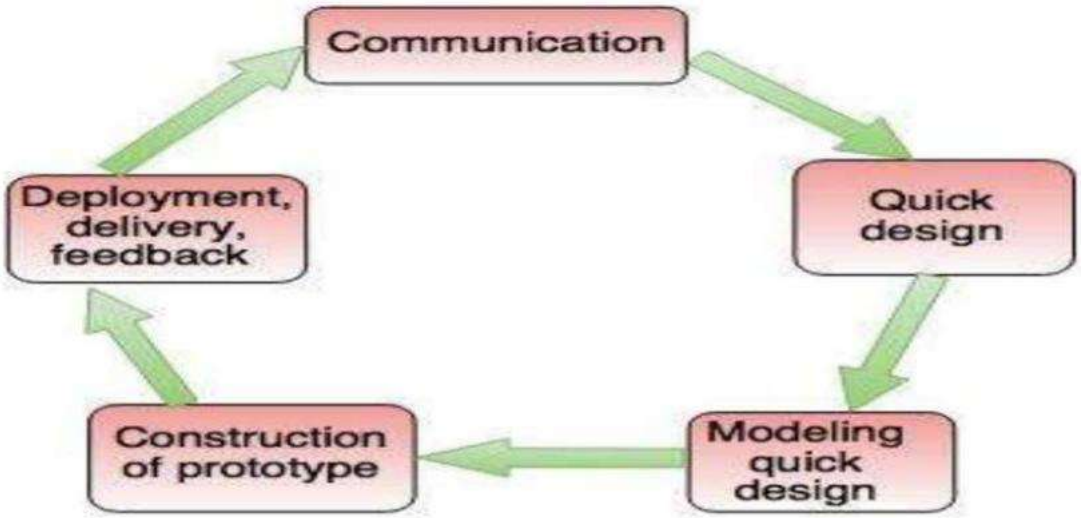


Fig. - The Prototyping Model

The different phases of Prototyping model are:

1. Communication

In this phase, developer and customer meet and discuss the overall objectives of the software.

2. Quick design

- Quick design is implemented when requirements are known.
- It includes only the important aspects like input and output format of the software.
- It focuses on those aspects which are visible to the user rather than the detailed plan.
- It helps to construct a prototype.

3. Modeling quick design

- This phase gives the clear idea about the development of software because the software is now built.
- It allows the developer to better understand the exact requirements.

4. Construction of prototype

The prototype is evaluated by the customer itself.

5. Deployment, delivery, feedback

- If the user is not satisfied with current prototype then it refines according to the

requirements of the user.

- The process of refining the prototype is repeated until all the requirements of users are met.
- When the users are satisfied with the developed prototype then the system is developed on the basis of final prototype.

Advantages of Prototyping Model

- Prototype model need not know the detailed input, output, processes, adaptability of operating system
- and full machine interaction.
- In the development process of this model users are actively involved.
- The development process is the best platform to understand the system by the user.

b. Explain Capability Maturity Model.

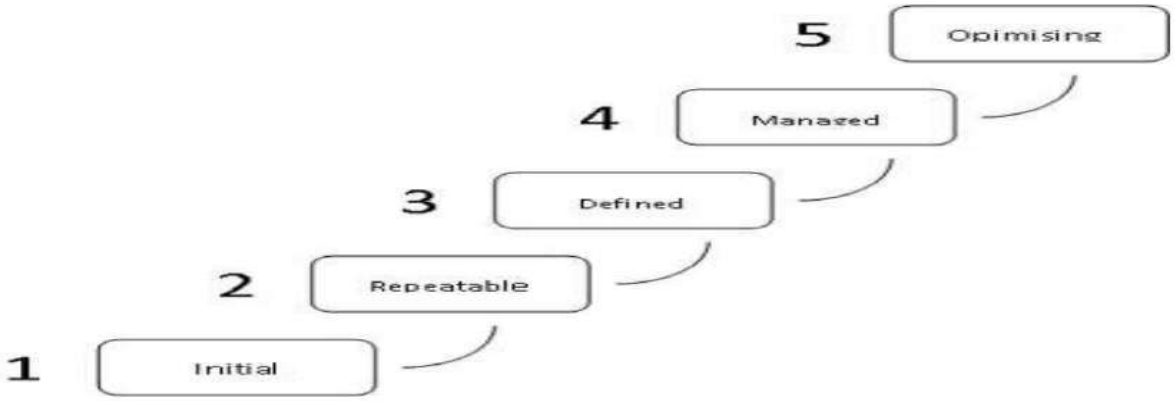
Capability Maturity Model (CMM)

- A maturity model is applied within the context of an SPI (Software Process Improvement). The intent of the maturity model is to provide an overall indication of the "process maturity" exhibited by a software organization. That is, an indication of the quality of the software process, the degree to which practitioner's understand and apply the process, and the general state of software engineering practice. This is accomplished using some type of ordinal scale.

- The Capability Maturity Model (CMM) provides a framework for organizing these evolutionary steps into five maturity levels that lay successive foundations for continuous process improvement. The five maturity levels define a scale for measuring the maturity of an organization's software process and for evaluating the capability of these processes. They also help an organization prioritize its improvement efforts.
- A maturity level is a well-defined evolutionary plateau toward achieving a mature software process. Each maturity level comprises a set of process goals that, when satisfied, stabilize an important component of the process. Achieving each level of maturity framework establishes a different component in the software process, resulting in an increase in the

process capability of the organization.

The five Software Capability Maturity levels have been defined as:



Level 5, Optimized—The organization has quantitative feedback systems in place to identify process weaknesses and strengthen them pro-actively. Project teams analyze defects to determine their causes; software processes are evaluated and updated to prevent known types of defects from recurring.

Level 4, Managed—Detailed software process and product quality metrics establish the quantitative evaluation foundation. Meaningful variations in process performance can be distinguished from random noise, and trends in process and product qualities can be predicted.

Level 3, Defined—Processes for management and engineering are documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization's standard software process for developing software.

Level 2, Repeatable—Basic project management processes are established to track cost, schedule, and functionality. Planning and managing new products is based on experience with similar projects.

Level 1, Initial—Few processes are defined, and success depends more on individual heroic efforts than on following a process and using a synergistic team effort.

C. Explain waterfall and incremental model. Differentiate between them.

Waterfall Model is also known as Classical/Traditional Model. Sometimes It is referred as linear-sequential life cycle model because all phases involve in this model completed one by one in linear fashion. In this model, we get software after completion of all coding phase. This model is basically used for small projects. There exist only one cycle in waterfall model.

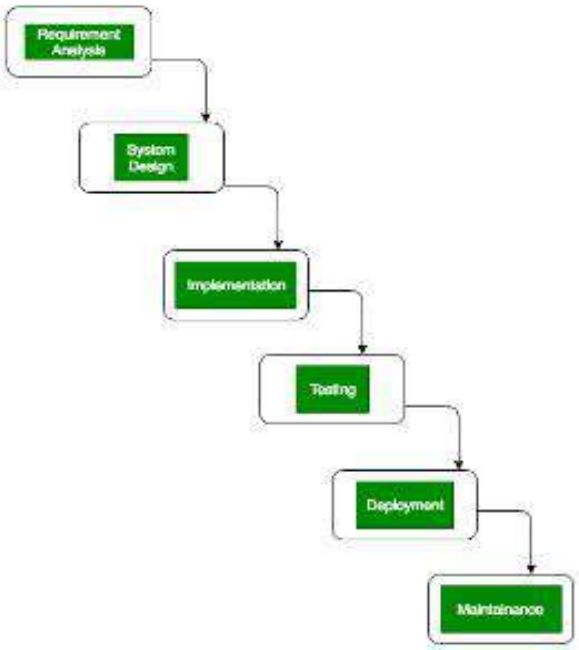


Figure – Waterfall Model

While in Incremental Model Multiple development cycles take place and these cycles are divided into more smaller modules. Generally a working software in incremental model is produced during first module Each subsequent release of the module adds function to the previous release. In incremental model, process continues till the complete system is achieved.

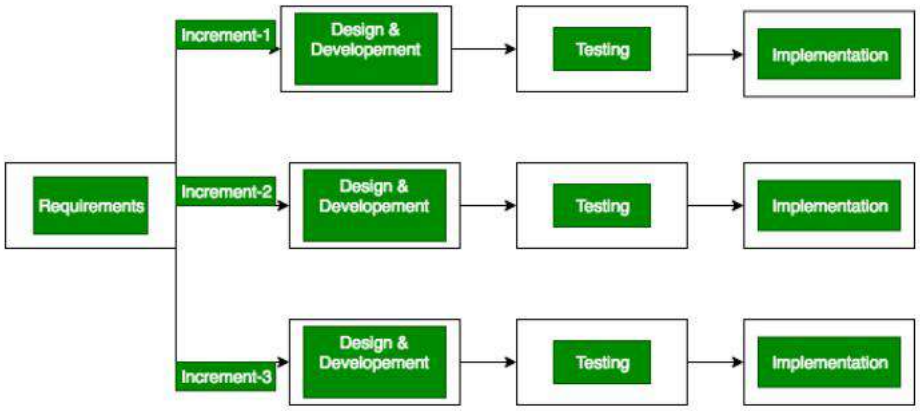


Figure – Incremental Model

Some differences between them are given below:

S. No.	Waterfall Model	Incremental Model
1.	Need for Detailed Documentation in the waterfall model is Necessary.	The need for Detailed Documentation in the incremental model is Necessary but not too much.
2.	In the waterfall model, early stage planning is necessary.	In an incremental model, early-stage planning is also necessary.
3.	There is a high amount of risk in the waterfall model.	There is a low amount of risk in the incremental model.
4.	There is a long waiting time for running software in the waterfall model.	There is a short waiting time for running software in the incremental model.
5.	The waterfall model can't handle large projects.	The incremental model also can't handle large projects.
6.	Flexibility to change in the waterfall model is Difficult.	Flexibility to change in incremental model is Easy.
7.	The cost of the Waterfall model is Low.	The cost of the incremental model is also Low.
8.	Testing is done in the waterfall model after the completion of the coding phase.	Testing is done in the incremental model after every iteration of the phase.

d. Write a short note on Agility Principles.

Agile software engineering embraces a philosophy that encourages customer satisfaction, incremental software delivery, small project teams (composed of software engineers and stakeholders), informal methods, and minimal software engineering work products.

- Agile software engineering guidelines stress on-time delivery of an operational software increment over analysis and design
- Agile software engineering combines a philosophy and a set of deployment guidelines.
- The philosophy encourages customer satisfaction and early incremental delivery of software: small, highly motivated project teams; informal methods; minimal software engineering work products; and overall development simplicity.
- The development guidelines stress delivery over analysis communication between developers and customers.

Agility Principles:

- Highest priority is to satisfy customer through early and continuous delivery of valuable software
- Welcome changing requirements even late in development, accommodating change is viewed as increasing the customer's competitive advantage
- Delivering working software frequently with a preference for shorter delivery schedules (e.g., every 2 or 3 weeks)
- Business people and developers must work together daily during the project
- Build projects around motivated individuals, given them the environment and support they need, trust them to get the job done
- Face-to-face communication is the most effective method of conveying information within Working software is the primary measure of progress
- Agile processes support sustainable development, developers and customers should be

able

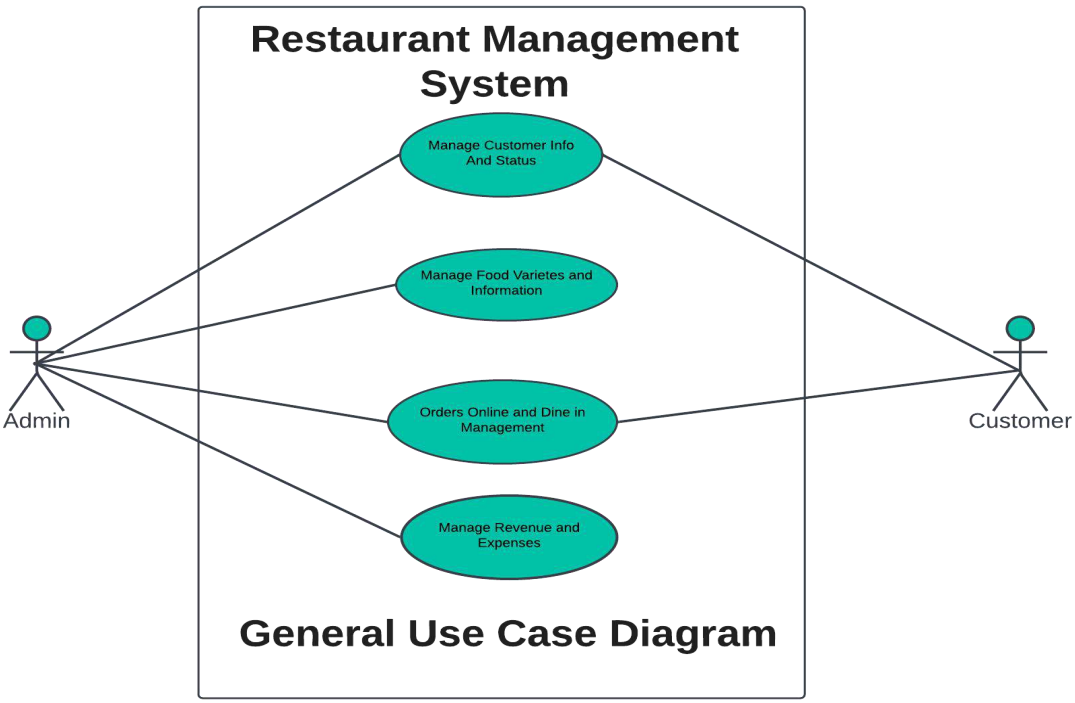
to continue development indefinitely

Q2. a. Draw use case diagram for Food Ordering System.

Continuous attention to technical excellence and good design enhances agility Use Case Diagram for the asked system will include:

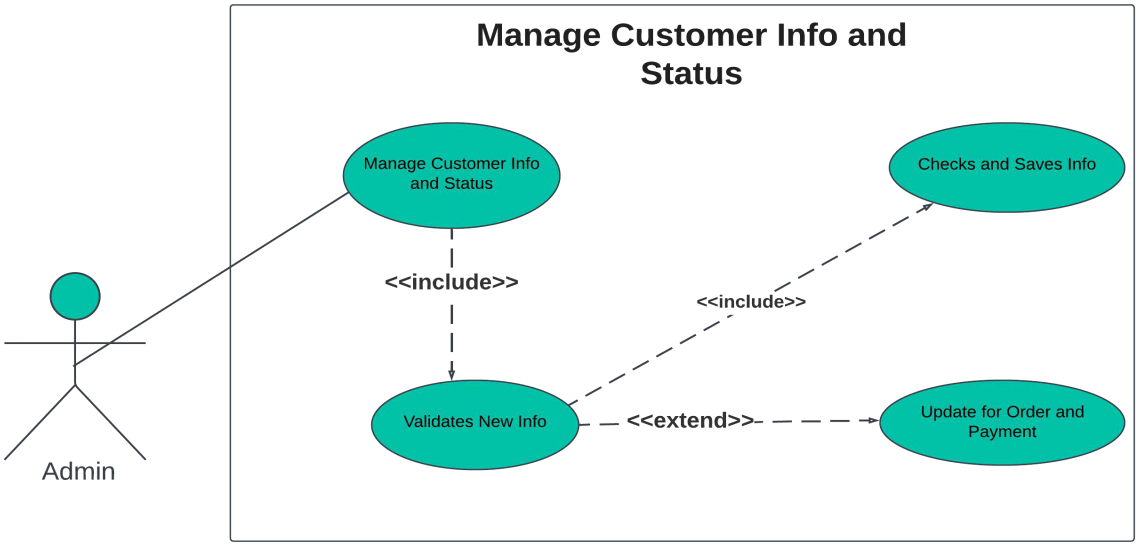
- General Use case diagram
- Monitor and Manage Customers' Information and Status
- Manage Food Information and Varieties
- Orders Online and Dine in Management
- Manage Revenue and Expenses

General Use case diagram: This diagram shows the general processes or function that the system could do which is based on the transactions done by the admin and customers. These activities involve managing restaurant activities as well as their customers' order and payment.



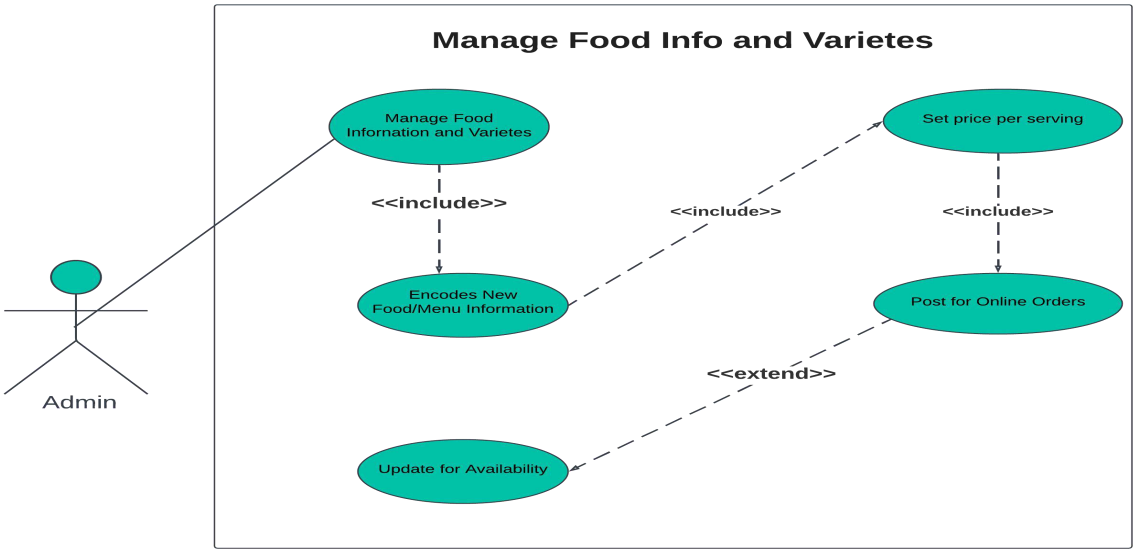
Monitor and Manage Customers' Information and Status

This is where the person in charge of the system could keep track of and manage the information and status of their customers. In this way, they were able to keep track of every order and payment made by their customers. These were very important to them when they did their inventory and summation of their income.



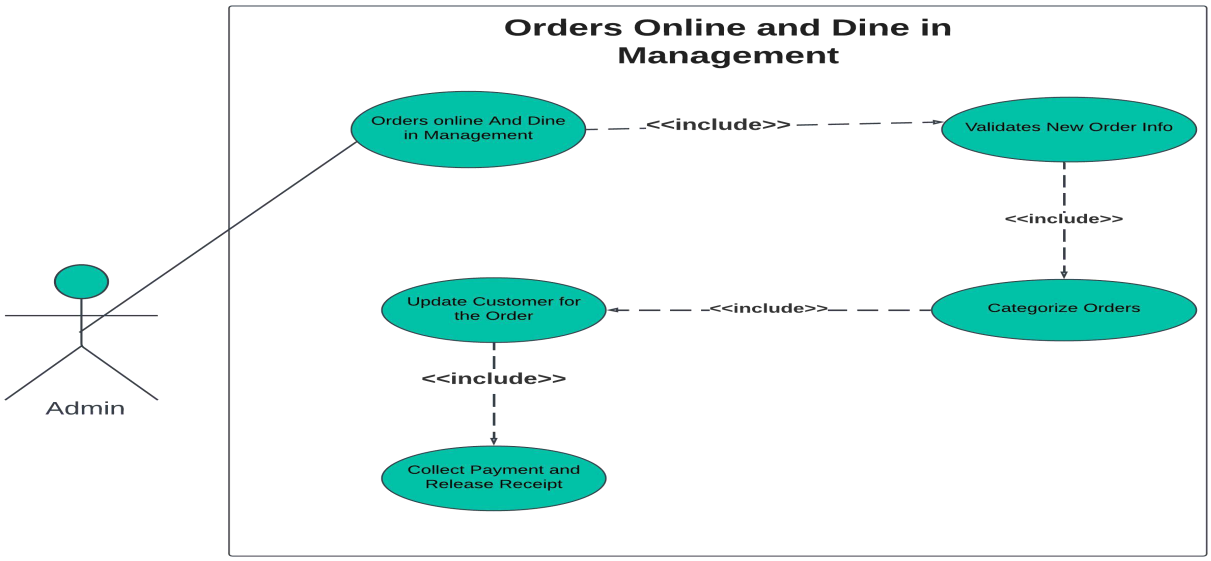
Manage Food Information and Varieties

It records the food they have and encodes the types of food they sell to their customers as part of the process. Prices and costs for each menu will also be managed by the admin.



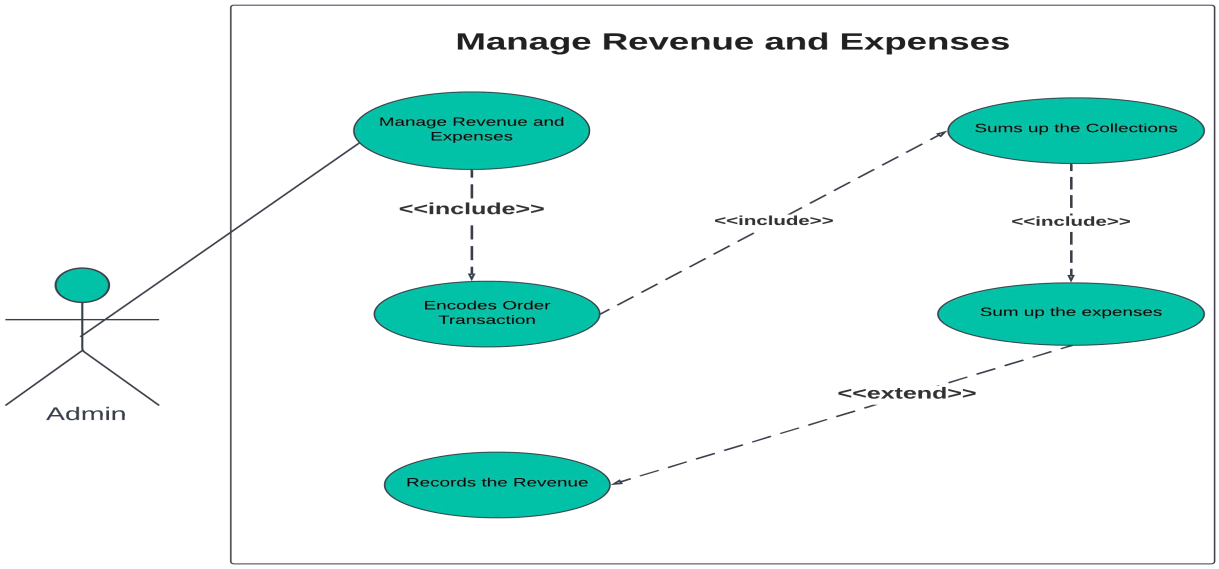
Orders Online and Dine in Management

Restaurant Management System administrators use this process to show how they use customer orders. These orders were put into groups based on whether they were ordered dine in or bought online. Also, this process will also keep track of the transactions that were made for inventory.



Manage Revenue and Expenses

Managing Revenue and Expenses describes how the system administrator handles the calculations and summing of money depending on data provided into the system. Therefore, this is an important role of the system that a programmer must prioritize.



- Simplicity (defined as maximizing the work not done) is essential
- The best architectures, requirements, and design emerge from self-organizing teams
- At regular intervals teams reflects how to become more effective and adjusts its behavior accordingly.

b. Draw the DFD up to Level 2 for Food Ordering System

Food Ordering System is actually a type of software that allows the manager of restaurants to manage and accept the placed orders over the Internet or in the restaurant.

Different levels of DFD are shown for Food Ordering System such as Level 0 DFD, Level 1 DFD, Level 2 DFD, and Level 3 DFD.

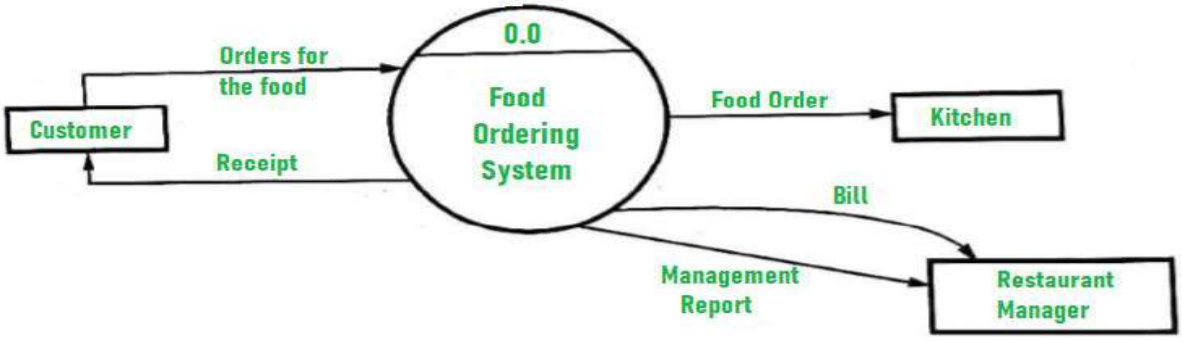
Level 0 DFD – At this level, the Input and Output of the system are shown. The system is designed and established across the world with input and output at this level.

Food Ordering System has the following input:

- Food order is input as the customer's order for food.

Food Ordering System has the following output:

- Receipt of the order.
- For further processing the order, the food order is passed to the kitchen.
- The restaurant manager gets the report of Bill and Management.

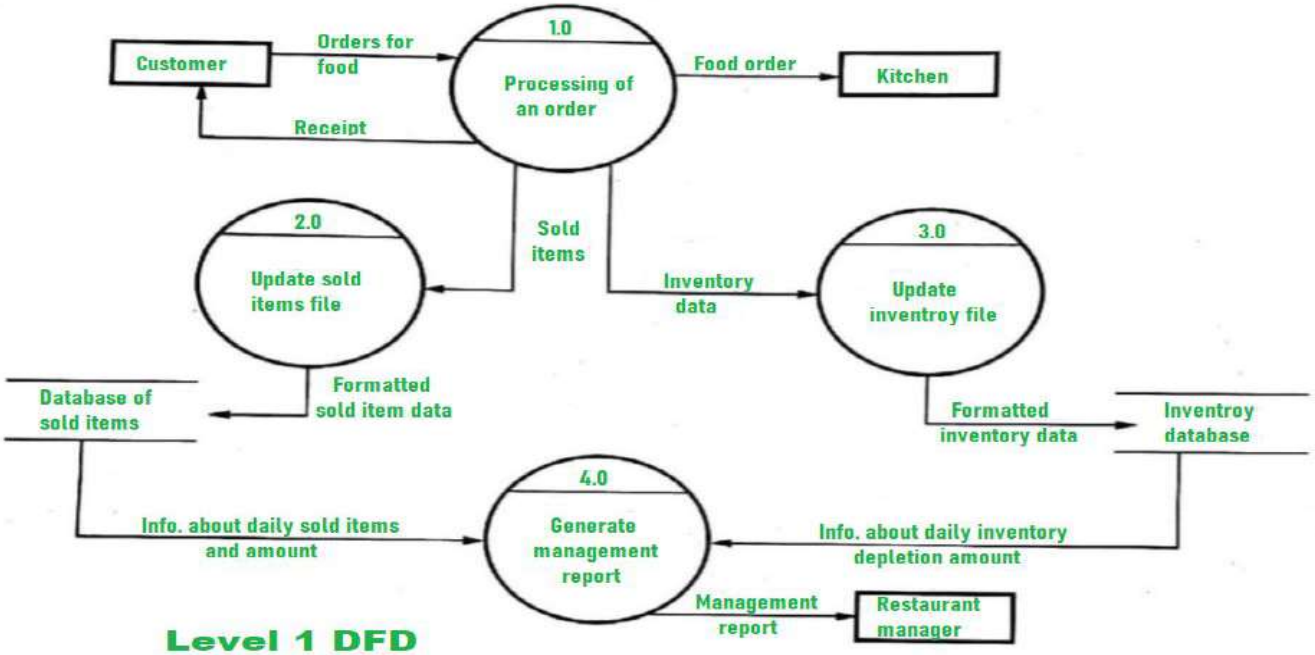


Level 0 DFD (Context Level

Level 1 DFD –

For processing the order, process 1.0 is responsible. For food, the housekeeping activities involved are represented by processes 2.0, 3.0, and 4.0. The detailed information about daily sold items should be available to create and report management and the list of items that are available 'in-stock' should be kept by maintaining the inventory data (describes the records of datasets such as their name, their content, source, many useful information, etc.) at the same time. Hence, two data stores are used in this level of DFD: Database of Sold items, Inventory database.

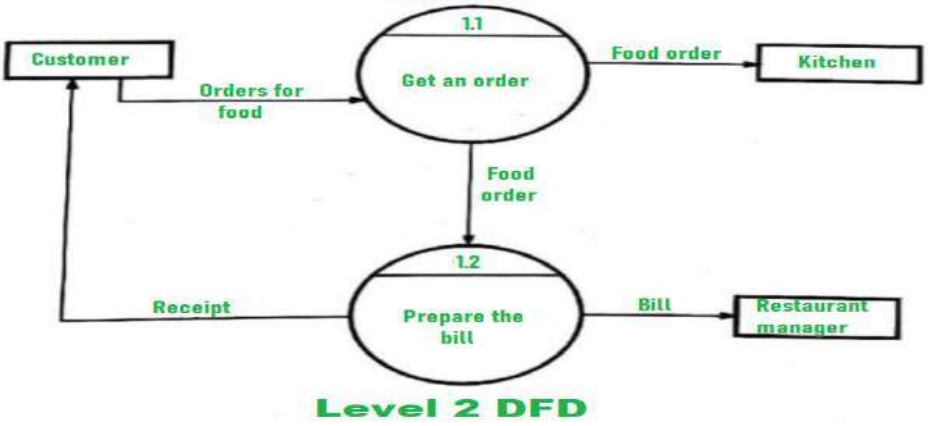
In the end, with the use of the amount of daily sold items and daily inventory depletion, it is easy to prepare a report of management. Further, the restaurant manager gets this report of management.



Level 1 DFD

Level 2 DFD –

Detailed information about "Processing of an Order" is shown below:



Level 2 DFD

C. Draw use case diagram for ATM banking system.

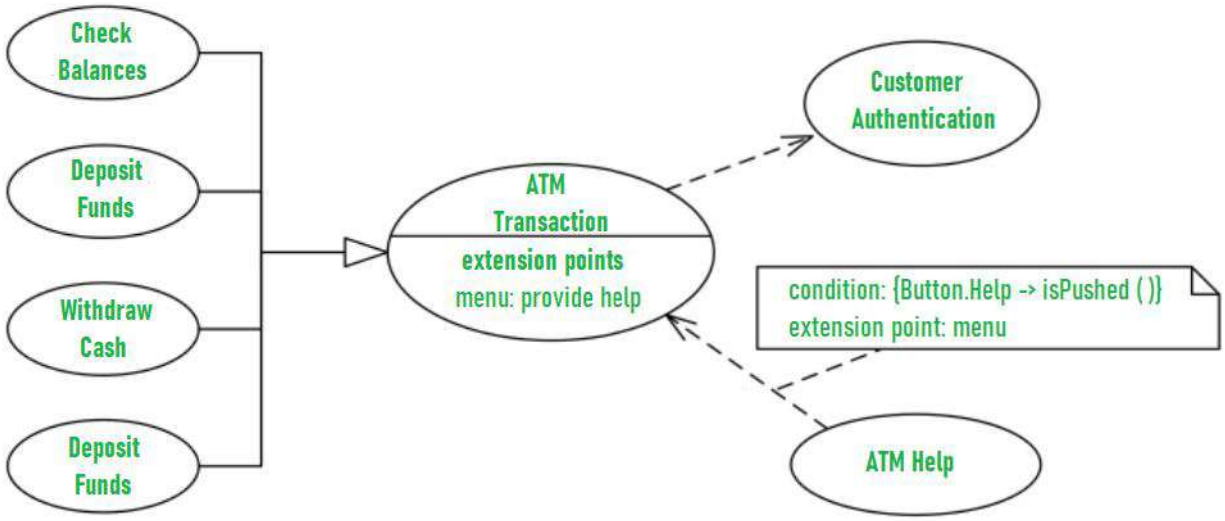
Automated Teller Machine (ATM) also known as ABM (Automated Banking Machine) is a banking system. This banking system allows customers or users to have access to financial transactions. These transactions can be done in public space without any need for a clerk, cashier, or bank teller. Working and description of the ATM can be explained with the help of the Use Case Diagram.

Let us understand about designing the use case diagram for the ATM system. Some scenarios of the system are as follows.

Step-1:

The user is authenticated when enters the plastic ATM card in a Bank ATM. Then enters the user's name and PIN (Personal Identification Number). For every ATM transaction, a Customer Authentication use case is required and essential. So, it is shown as include relationship.

Example of use case diagram for Customer Authentication is shown below:

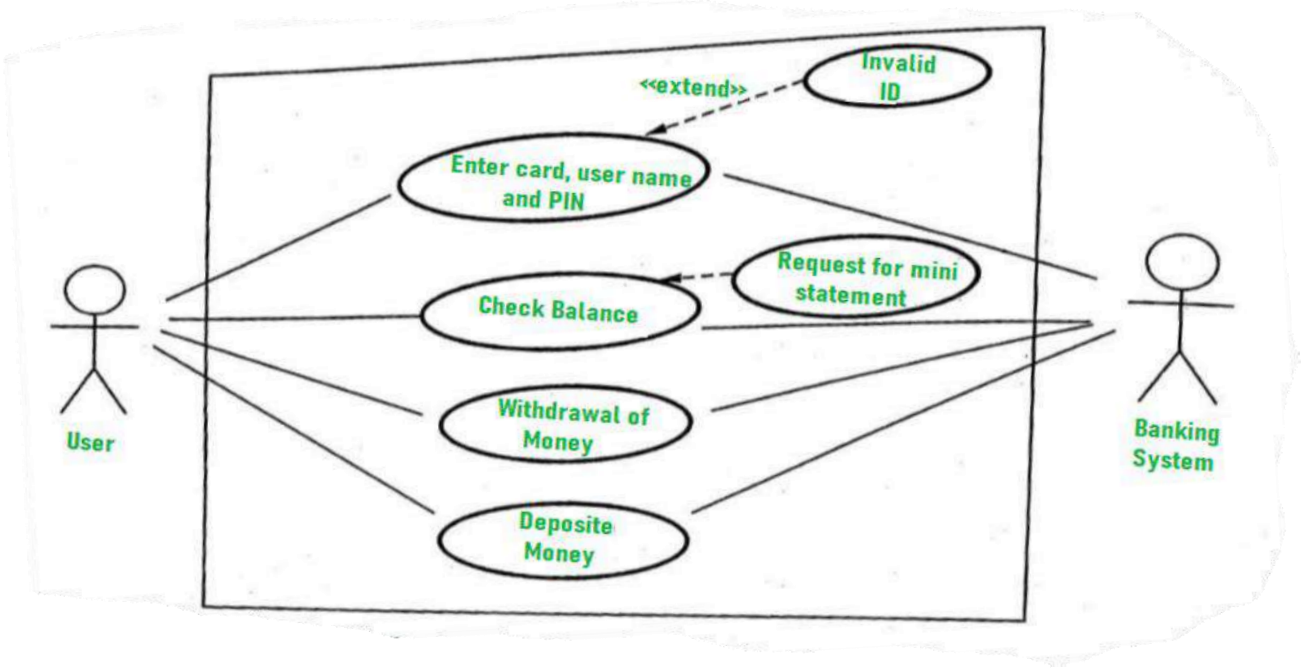


Use Case Diagram for Customer Authentication

Step-2:

User checks the bank balance as well as also demands the mini statement about the bank balance if they want. Then the user withdraws the money as per their need. If they want to deposit some money, they can do it. After complete action, the user closes the session.

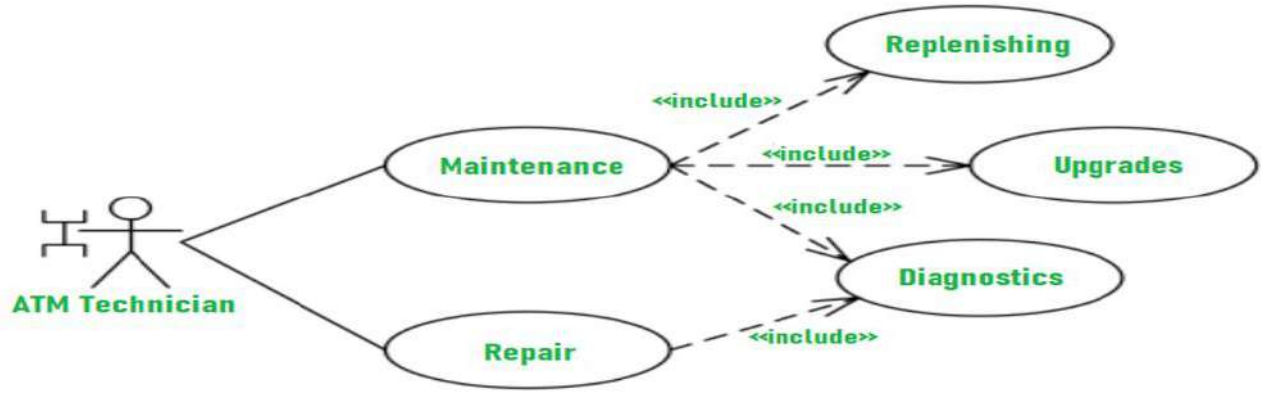
Example of the use case diagram for Bank ATM system is shown below:



Step-3:

If there is any error or repair needed in Bank ATM, it is done by an ATM technician. ATM technician is responsible for the maintenance of the Bank ATM, upgrades for hardware, firmware or software, and onsite diagnosis.

Example of use case diagram for working of ATM technician is shown below:



3 a. Given the following values, compute function point when all complexity adjustment factor (CAF) and weighting factors are average. User Input = 50, User Output = 40, User Inquiries = 35, User Files = 6, External Interface = 4

User Input = 50

User Output = 40

User Inquiries = 35

User Files = 6

External Interface = 4

Explanation:

- **Step-1:**
  TDI = 14 * scale
  Scale varies from 0 to 5 according to character of **Value Adjustment Factor** (VAF).
  Below table shows scale:

  0 - No Influence

  1 - Incidental

  2 - Moderate

  3 - Average

  4 - Significant

  5 - Essential

As Value adjustment factor is average (given in question), hence scale = 3.
TDI = 14 * 3 = 42

- **Step-2:**
  VAF = 0.65 + (0.01 * 42) = 1.07
- **Step-3:** As weighting factors are also average (given in question) hence we will multiply each individual function point to corresponding values in TABLE.

Function type	Simple	Average	Complex
External Inputs	3	4	6
External Output	4	5	7
External Inquiries	3	4	6
Internal Logical Files	7	10	15
External Interface Files	5	7	10

UFP = (50*4) + (40*5) + (35*4) + (6*10) + (4*7) = 628

- **Step-4:**
  Function Point = 628 * 1.07 = 671.96

  This is the required answer.

b)Given the following values, compute function point value for a project with the following information domain characteristics. User Input = 32 User Output = 60 User Inquiries = 24 User Files = 8 External Interface = 2. Assume that all complexity adjustment value are average and ∑fi=40

1. Given the following values, compute function point when all complexity adjustment factor (VAF) and weighting factors are average.

User Input = 32

User Output = 60

User Inquiries = 24

User Files = 8

Solution: External Interface = 2

Explanation:

• Step-1:

TDI = 14 * scale

Scale varies from 0 to 5 according to character of Value Adjustment Factor (VAF). Below table

shows scale:

0 - No Influence

1 - Incidental

2 - Moderate

3 - Average

4 - Significant

5 - Essential

As Value adjustment factor is average and summation is given.

TDI = ∑(fi)=40

• Step-2:

VAF = 0.65 + (0.01 * 40) = 1.05

• Step-3: As weighting factors are also average (given in question) hence we will multiply each individual function point to corresponding values in TABLE.

UFP = 4*32+60*5+24*4+8*10+7*2=618

• Step-4:

Function Point = 618 * 1.05 = 648.9

This is the required answer.