Department of Information Sciences

University of Education, Lahore

(Multan Campus)

Online Food Ordering

By

Abdullah Ahsan

Barira Mahnoor

Chapter: #4

Semester: 3rd BSCS

Table of Contents

1. Introduction:

1.1. Purpose of the Document

The purpose of this document is to provide a comprehensive and detailed overview of the requirements and specifications for the development of the Food Ordering System. The document aims to serve as a reference guide for stakeholders involved in the development, testing, and maintenance of the system. By clearly defining the scope, objectives, and constraints of the project, this document will facilitate a shared understanding among all project participants.

1.2. Scope of the Food Ordering System

The Food Ordering System is designed to streamline and enhance the process of ordering food from various restaurants through an online platform. The system caters to both customers seeking a convenient and efficient way to browse, order, and track food deliveries and restaurant staff managing their menus and processing incoming orders. The scope includes the development of user interfaces for customers, restaurant staff, and administrators, as well as the backend processes that enable seamless order processing and delivery tracking.

2. Software Development Life Cycle

2.1. Software Development Life Cycle

The software industry includes many different processes, for example, analysis, development, maintenance and publication of software. This industry also includes software services, such as training, documentation, and consulting. Our focus here about software development life cycle (SDLC). So, due to that different types of projects have different requirements. Therefore, it may be required to choose the SDLC phases according to the specific needs of the project. These different requirements and needs give us various software development approaches to choose from during software implementation

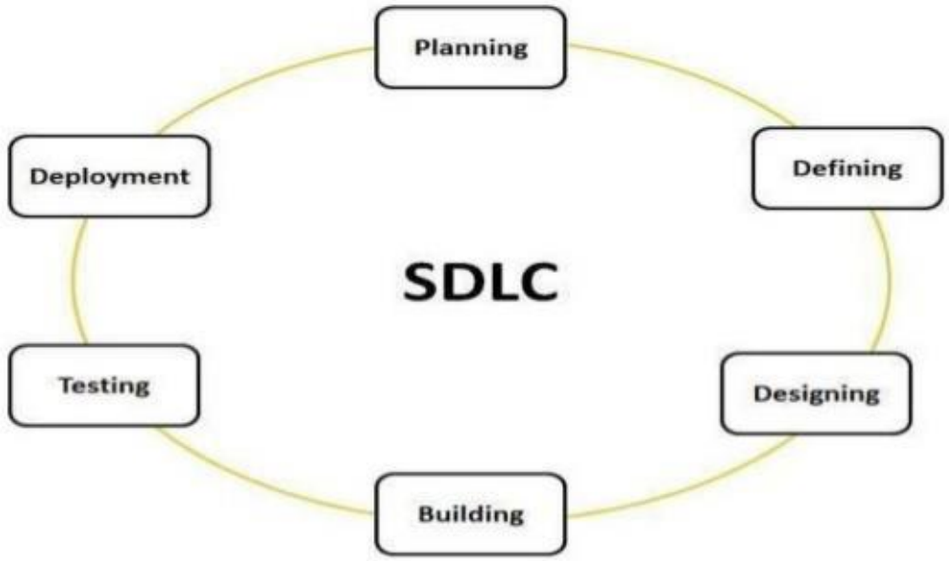

Figure 1 SDLC Phases

2.2. SDLC Models

There are various software development life cycle models defined and designed which are followed during software development process. These models are also referred as "Software Development Process Models". Each process model follows a Series of steps unique to its type, in order to ensure success in process of software development.

Following are the most important and popular SDLC models followed in the industry:

1. Waterfall model
2. V-Shaped model
3. Prototyping model
4· Spiral model
5. Iterative and Incremental model
6. Agile model
7. RAD model

2.3. Analysis of the Project Model

Selecting accurate model for developing of the software invention or request is very significant. Founded on the model the expansion and testing processes are accepted out. As

Waterfall Model is more traditional and easier to gather requirements and analyzing system, so we choose this model according to complete this project. This technique works well for big projects that may take numerous months to progress.

2.3.1. Waterfall Model

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development. The waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase is complete. In waterfall model phases do not overlap.

2.3.2. Waterfall Model Design

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is a diagrammatic representation of different phases of waterfall model.

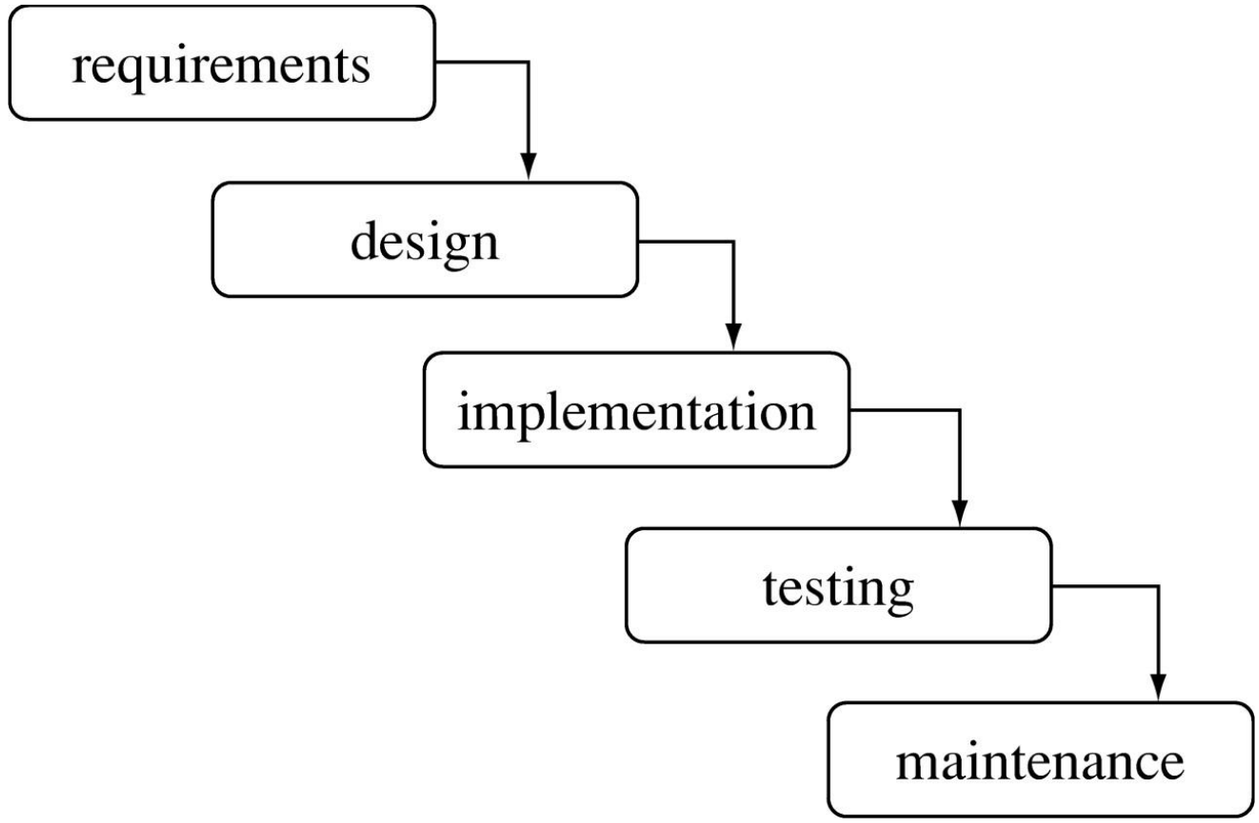

Figure 2 Waterfall Model

2.3.3. Waterfall Model Phases

The sequential phases in Waterfall model are:

1. Requirement Gathering and analysis: All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

2. System Design: The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

3. Implementation: With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

4. Integration and Testing: All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

5. Deployment of system: Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.

6. Maintenance: There are some issues which come up in the client environment. To fix those issues patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model phases do not overlap.

2.4. Advantages and Disadvantages of Waterfall Model

2.4.1. Advantages of Waterfall Model:

1) Waterfall model is simple to implement and also the amount of resources required for it are minimal.

2) In this model, output is generated after each stage (as seen before), therefore it has high visibility. The client and project manager gets a feel that there is considerable progress. Here it is important to note that in any project psychological factors also play an important role.

3) Project management, both at internal level and client's level, is easy again because of visible outputs after each phase. Deadlines can be set for the completion of each phase and evaluation can be done from time to time, to check if project is going as per milestones.

4) This methodology is significantly better than the haphazard approach to develop software. It provides a template into which methods of analysis, design, coding, testing and maintenance can be placed.

5) This methodology is preferred in projects where quality is more important as compared to schedule or cost.

2.4.2. Disadvantages of Waterfall Model:

1) Real projects rarely follow the sequential flow and iterations in this model are handled indirectly. These changes can cause confusion as the project proceeds.

2) It is often difficult to get customer requirements explicitly. Thus, specifications can't be freeze. If that case arises baseline approach is followed, wherein output of one phase is carried forward to next phase. For example, even if SRS is not well defined and requirements can't be freeze, still design starts. Now if any changes are made in SRS then formal procedure is followed to put those changes in baseline document.

3) In this model we freeze software and hardware. But as technology changes at a rapid pace, such freezing is not advisable especially in long-term projects.

4) This method is especially bad in case client is not IT-literate as getting specifications from such a person is tough.

5) Even a small change in any previous stage can cause big problem for subsequent phases as all phases are dependent on each-other.

6) Going back a phase or two can be a costly affair.

3. REQUIREMENT GATHERING/ANALYSIS

3.1. Requirement Analysis

Requirements analysis in systems engineering and software engineering, encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.

Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

3.2. System Requirement

Our system can be used in windows XP, windows 7, and windows 8 with 32-bits, and 64-bit operating system and also supported for another platform such as Linux OS X.

3.3. Software and Hardware Requirements

3.3.1. Software Requirements:

- Frontend design tool: Html5, CSS3, JavaScript, Word press, J Query.
- Server-side Scripting tool: Python Django.
- Web server software: Apache XAMPP, Local by Fly Wheel.
- Database tools: MYSQL.
- Compatible operating system: Windows, Mac.
- Software tools: WordPress CMS, Revolution Slider, Newsletter subscription,
- WooCommerce, Social sharing.

3.3.2. Hardware Requirements:

- Hardware recommend by all the software needed.
- RAM: 256MB or more
- Hard Drive: 10 GB or more
- Communication hardware to serve client request

3.4. User Requirements

To deliver the best service to the users we tried to find out the users' necessities which are below:

Administrator Aspect:

- Monitoring the whole system from admin panel.
- Taking back up of the database.
- Creating, deleting and modifying the records.
- Add users for the admin panel.
- Add customers and other staff.
- Keeping the customer's record like their details.
- Organizing their member registration system.
- Approve the notice to post.
- Monitoring the transaction system.

Customer Aspect:

- Signing in and signing up to the system.
- Changing their password.
- Resetting forgot password.

3.5. Functional Requirements:

In Software engineering and systems engineering, a functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behavior, and outputs.

Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases. Functional requirements are supported by non-functional requirements (also known as quality requirements), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). Generally, functional requirements are expressed in the form "system must do <requirement>", while non-functional requirements are "system shall be <requirement>". The plan for implementing functional requirements is detailed in the system design. The plan for implementing non-functional requirements is detailed in the system architecture.

As defined in requirements engineering, functional requirements specify particular results of a system. This should be contrasted with non-functional requirements which specify overall characteristics such as cost and reliability. Functional requirements drive the application architecture of a system, while non-functional requirements drive the technical architecture of a system.

1. **User Authentication and Authorization**
    a. Users should be able to register and log in
    b. User will provide their Name, Email, and Phone Number and set Password
    c. Differentiate between customer, restaurant staff, and admin roles
    d. Implement password recovery mechanisms
2. **Menu Management:**
    a. Restaurants should be able to perform CRUD (Create, Read, Update, Delete) operations on their menu items.
    b. Categorize menu items (appetizers, main courses, desserts, etc.).
    c. Include details like item name, description, price, and availability status.
3. **Ordering System:**
    a. Customers should be able to add items to their cart.
    b. Enable customers to customize their orders (e.g., toppings, special instructions).
    c. Support order review and confirmation.
4. **User Profiles:**
    a. Users should have profiles with personal information and order history.
    b. Provide an option for customers to save multiple delivery addresses.
5. **Shopping Cart and Checkout:**
    a. Maintain a shopping cart for each user session.
    b. Allow users to review their orders before confirming.
    c. Integrate a secure and user-friendly checkout process.
6. **Order Tracking:**
    a. Implement real-time order tracking for customers.
    b. Provide status updates (order received, preparation, out for delivery).
7. **Restaurant Management:**
    a. Allow restaurants to set their availability status (open/closed).
    b. Provide a dashboard for restaurants to view and manage orders.
8. **Admin Panel:**
    a. Admins should have access to manage users, restaurants, and overall system settings.
    b. Include reporting and analytics features for order trends and user behavior.
9. **Search and Filters:**
    a. Implement a robust search functionality for users to find restaurants and menu items.
    b. Include filters based on cuisine, price range, ratings, etc.
10. **Ratings and Reviews:**
    a. Allow customers to rate and review restaurants.
    b. Display average ratings for restaurants and menu items.

3.6. Non-Functional Requirements:

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design. The plan for implementing non-functional requirements is detailed in the system architecture, because they are usually Architecturally Significant Requirements.

1. **Performance:**
    a. Ensure fast loading times, even during peak hours.
    b. Handle a large number of simultaneous users and orders.
2. **Security:**
    a. Implement secure user authentication and data encryption.
    b. Protect against common web application vulnerabilities (e.g., SQL injection, cross-site scripting).
3. **Scalability:**
    a. Design the system to scale horizontally to accommodate increased user traffic.
4. **Reliability:**
    a. Ensure high availability and reliability of the system.
    b. Implement regular backups to prevent data loss.
5. **Usability:**
    a. Design an intuitive and user-friendly interface.
    b. Provide clear navigation and instructions for users.
6. **Compatibility:**
    a. Ensure compatibility with different web browsers and devices.
7. **Regulatory Compliance:**
    a. Comply with data protection and privacy regulations (e.g., GDPR).
8. **Integration:**
    a. Integrate with payment gateways for secure transactions.
    b. Integrate with map services for address verification and delivery tracking.
9. **System Maintenance:**
    a. Include features for system updates and maintenance without disrupting service.
10. **Documentation:**
    a. Provide comprehensive documentation for users, administrators, and developers.

3.7. Business Requirements

Business requirements in the context of software engineering or the software development life cycle, is about eliciting and documenting business requirements of business users such as customers, employees, and vendors early in the development cycle of a system to guide the design of the future system. Business requirements are often captured by business analysts, who analyze business activities and processes, and often study As-is process to define a target To-be process.

Business requirements often include:

- Business context, scope, and background, including reasons for change
- Key business stakeholders that have requirements
- Success factors for a future/target state
- Constraints imposed by the business or other systems
- Business process models and analysis, often using flowchart notations to depict either 'as-is' and 'to-be' business processes
- Logical data model and data dictionary references
- Glossaries of business terms and local jargon
- Data flow diagrams to illustrate how data flows through the information systems
- (different from flowcharts depicting algorithmic flow of business activities).

3.8. Data and Category Requirements

There are dissimilar classes of users namely admin, customer and other staff. Depending upon the category of users, the access rights are obvious. It means if the user is an administrator then he/she can be able to adjust the data delete, add etc. All other users expect the restaurant only has the rights to save the info about database. The database stores the detail of customer's proper time. Admin should be able to update restaurant records.

4. System Design (UML)

4.1. Use Case

In software and systems engineering, a use case is a list of actions or event steps, typically defining the interactions between a role and a system, to achieve a goal.

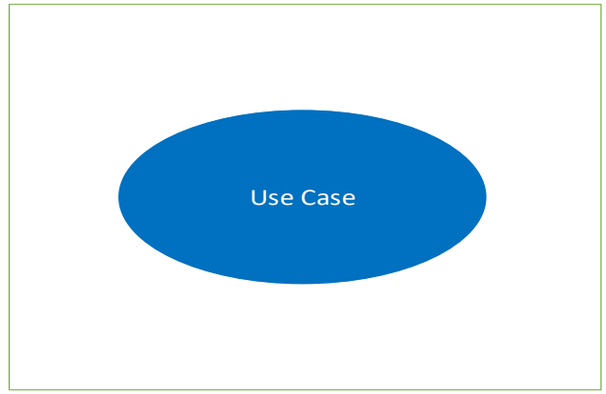

Figure 3 Use Case

4.1.1. Elements of Use Case Diagrams

Actor:

An actor in the Unified Modelling Language (UML) specifies a role played by a user or any other system that interacts with the subject.

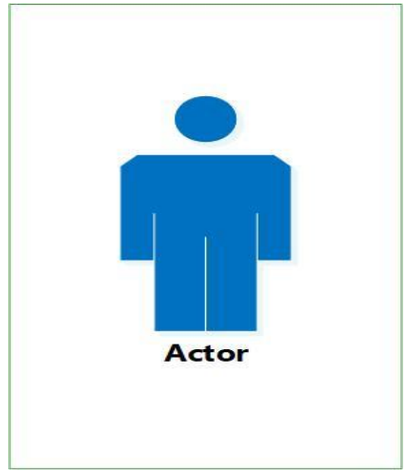

Figure 4 Actor

Association:
An association between an actor and a use case indicates that the actor and the use case somehow interact or communicate with each other.

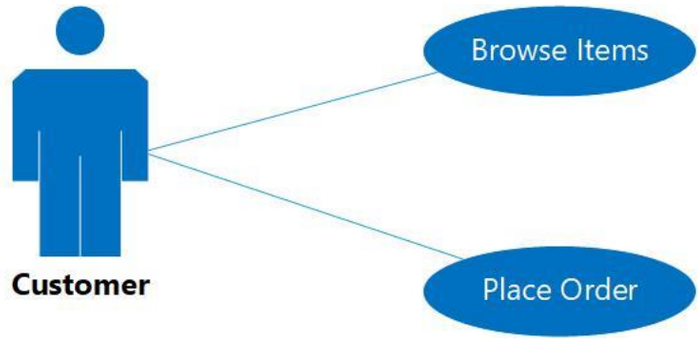

Figure 5 Associations in Use-Case

Includes
Include is used to extract use case fragments that are duplicated in multiple use cases. The included use case cannot stand alone and the original use case is not complete without the included one. This should be used sparingly an only in cases where the duplication is significant and exists by design (rather than by coincidence

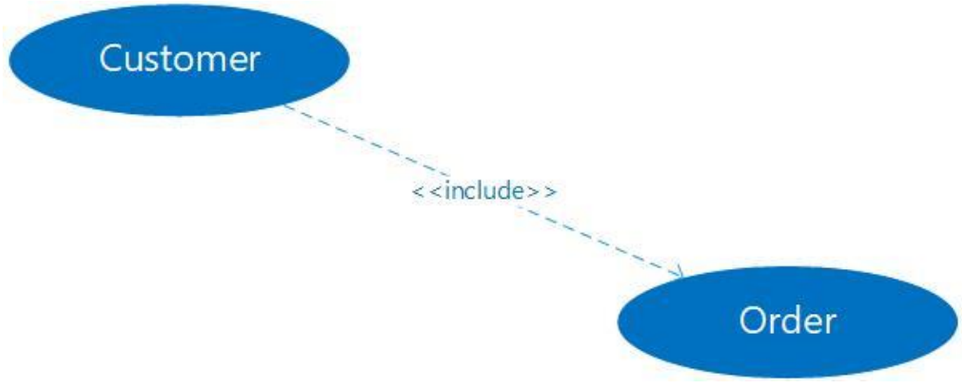

Figure 6 Include Relation in Use-Case

4.1.2. Actor Depiction

Actors are exterior entities that cooperate with the structure. Actor pledges system activities for the determination of finishing some task. Actors in this project are as follows:

Admin: Monitor the system, add product, add customer, add others staff etc.

Customer: Signing in and signing up to the system, changing their password and resetting forgot password.

4.1.3. Use-Case Diagram for Online Food Ordering System

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of a system and the various ways that they interact with the system.

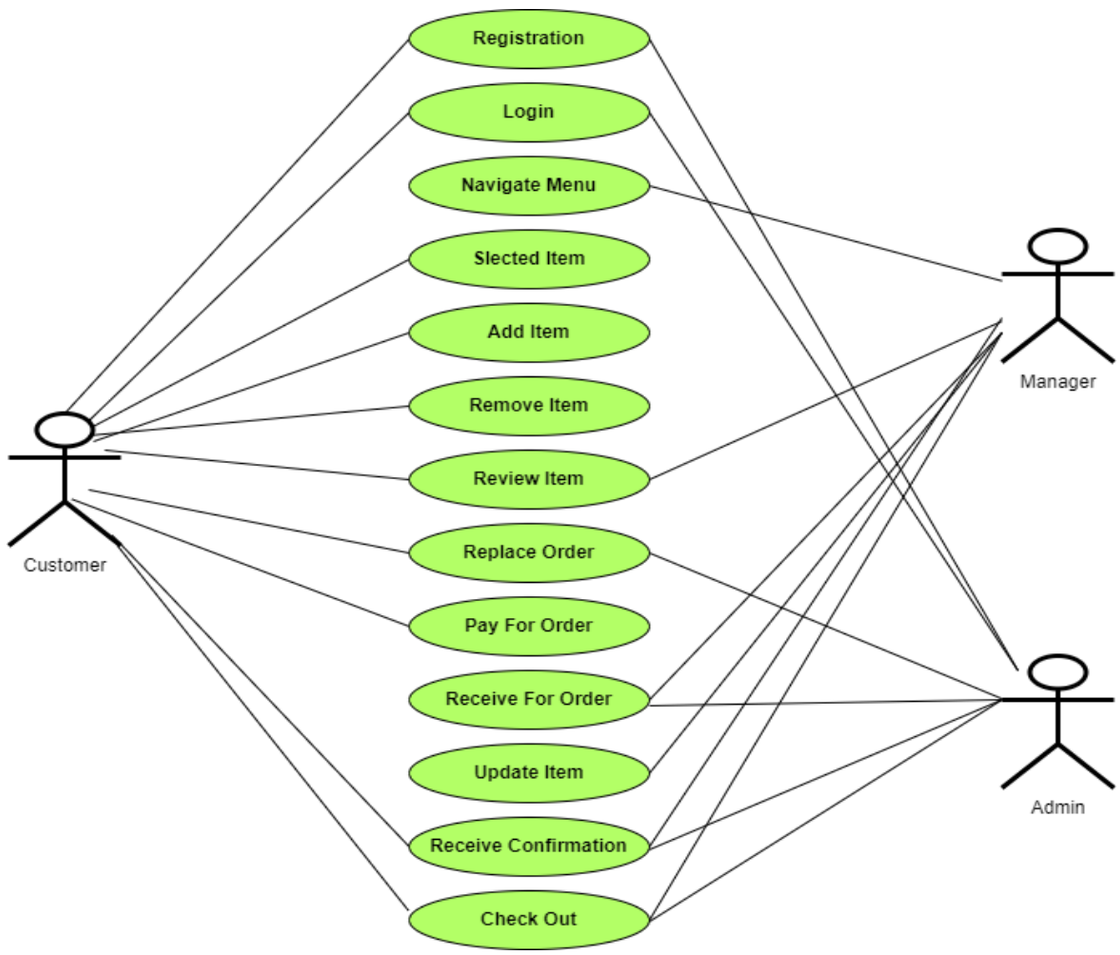

Figure 7 Use Case Diagram of Food Ordering System

4.2. Class Diagram of Online Food Ordering System

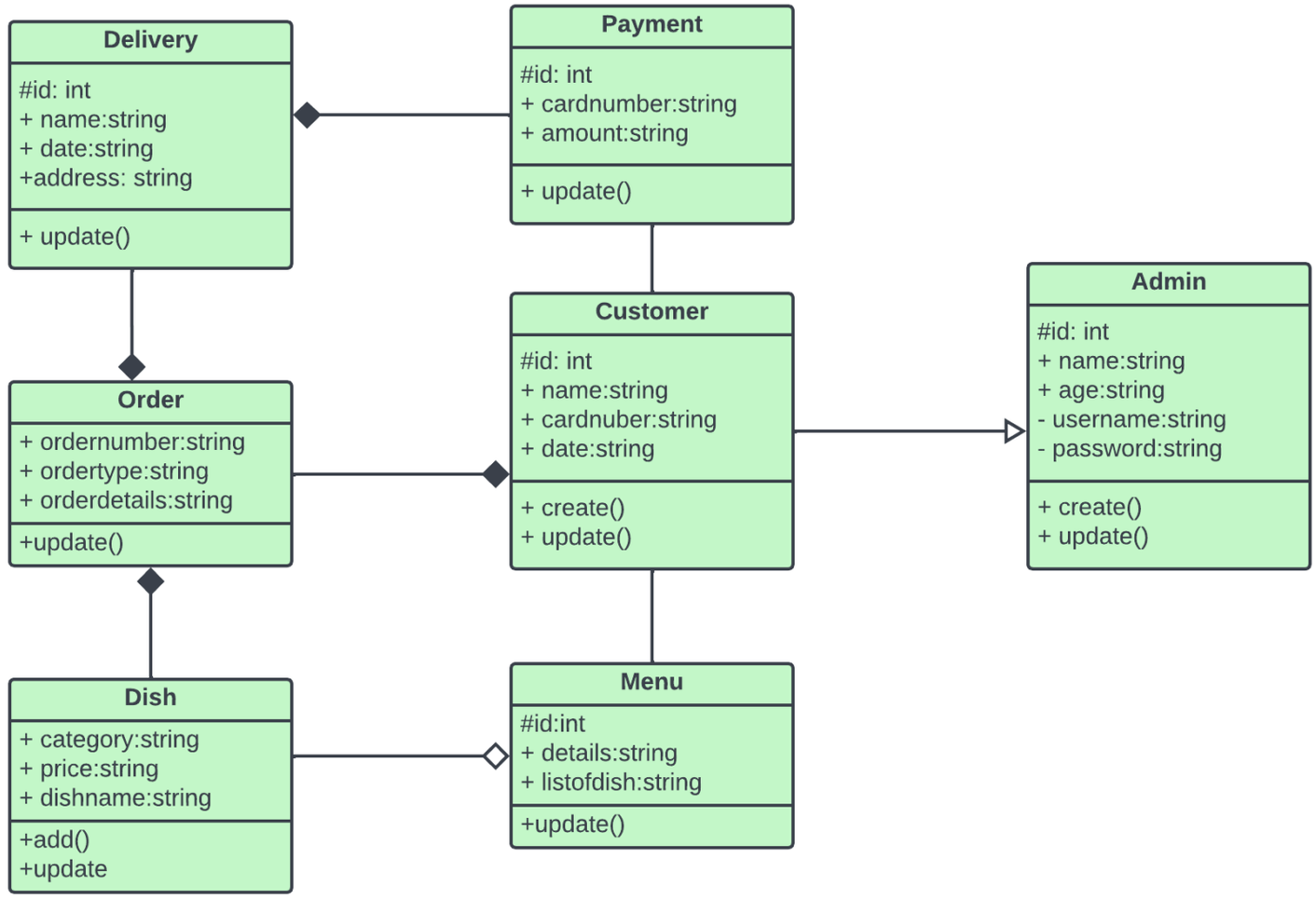

Figure 8 Figure: 4.6 Class diagram of Online Restaurant Management System

4.3. DFD of Online Food Ordering System

A food ordering system is essentially a software that enables restaurant managers to oversee and take orders that are placed in-person or over the Internet. Let's use a data flow diagram, or DFD, to better understand how the food ordering system operates. Below is the DFD for the food ordering system.

Various DFD levels, including Level 0 DFD, Level 1 DFD, Level 2 DFD, and Level 3 DFD, are displayed here for the Food Ordering System.

4.3.1. Level 0 DFD –

At this level, the Input and Output of the system are shown.

- 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.

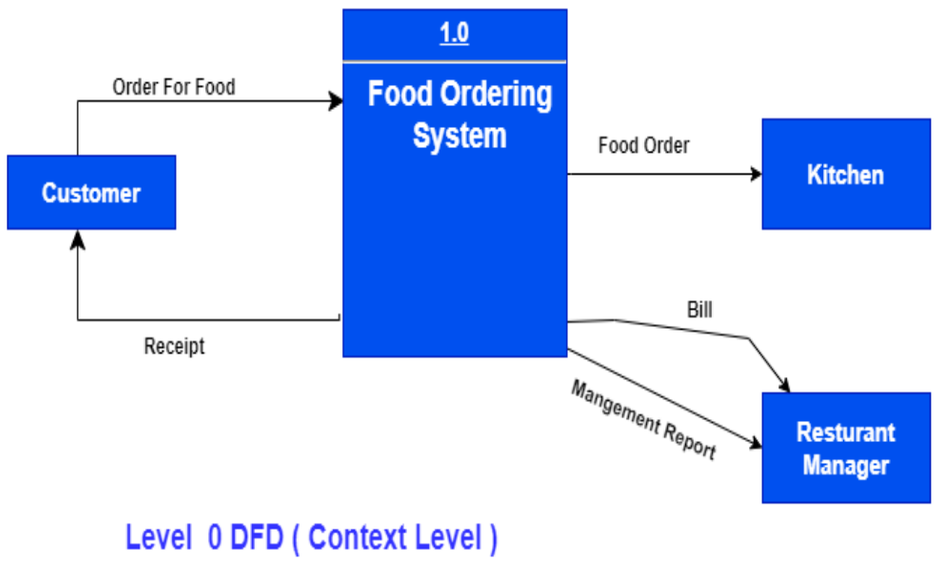

Figure 9 Level 0 DFD

4.3.2. Level 1 DFD –

Process 1.0 is in charge of processing the order. The associated housekeeping tasks for food are symbolized by processes 2.0, 3.0, and 4.0. The inventory data (which describes the records of datasets such as their name, their content, their source, many useful information, etc.) should be maintained at the same time as the detailed information about daily sold items is made available to create and report management.

Therefore, the DFD level shown below uses the following two data stores:

- Database of goods sold
- Database for inventory

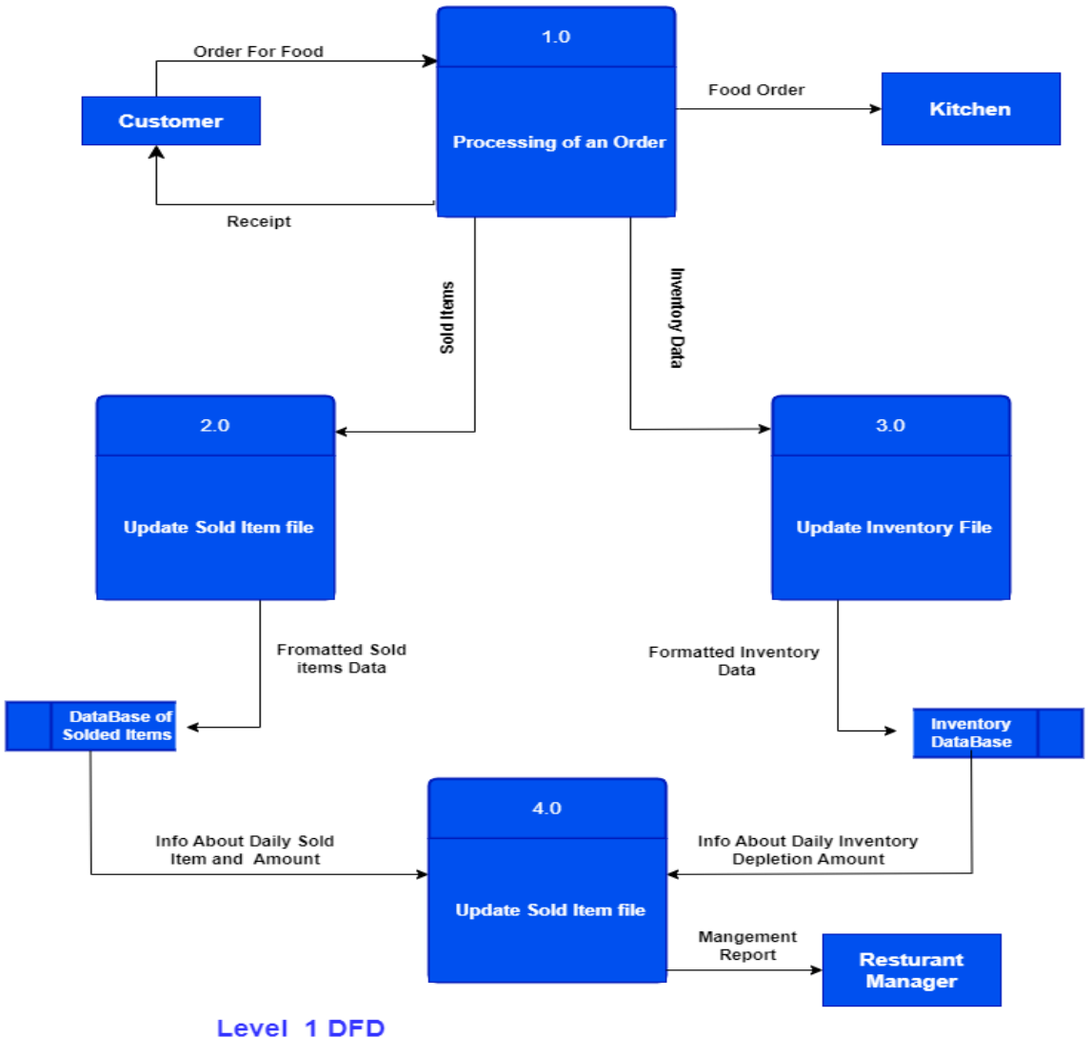

Figure 10 Level 1 DFD

4.3.3. Level 2 DFD –

Below is detailed information regarding "Processing of an Order":

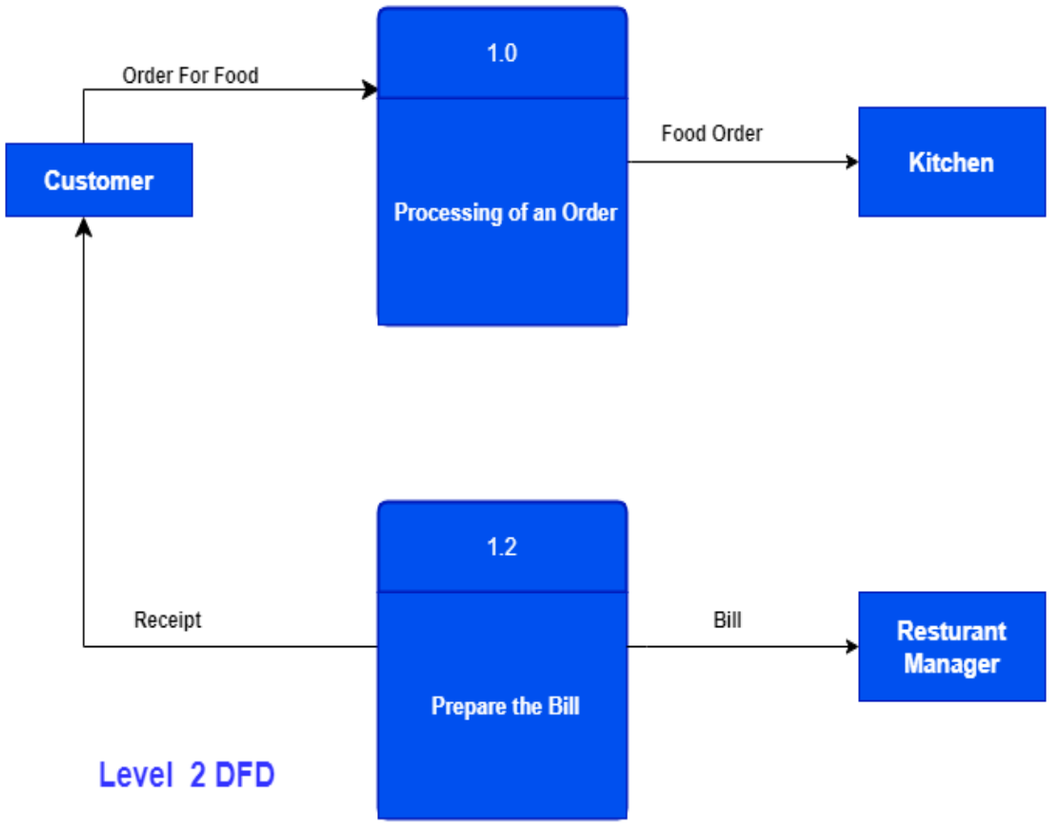

Figure 11 Level 2 DFD

4.3.4. Level 3 DFD –

At this DFD level, more specific details regarding the "Generate Management Report" activity are provided. To generate a management report, it is necessary to have access to inventory and sold item data. The management report that is generated from the aforementioned computations should then be given to the restaurant manager once the data for both solid goods and inventory have been combined.

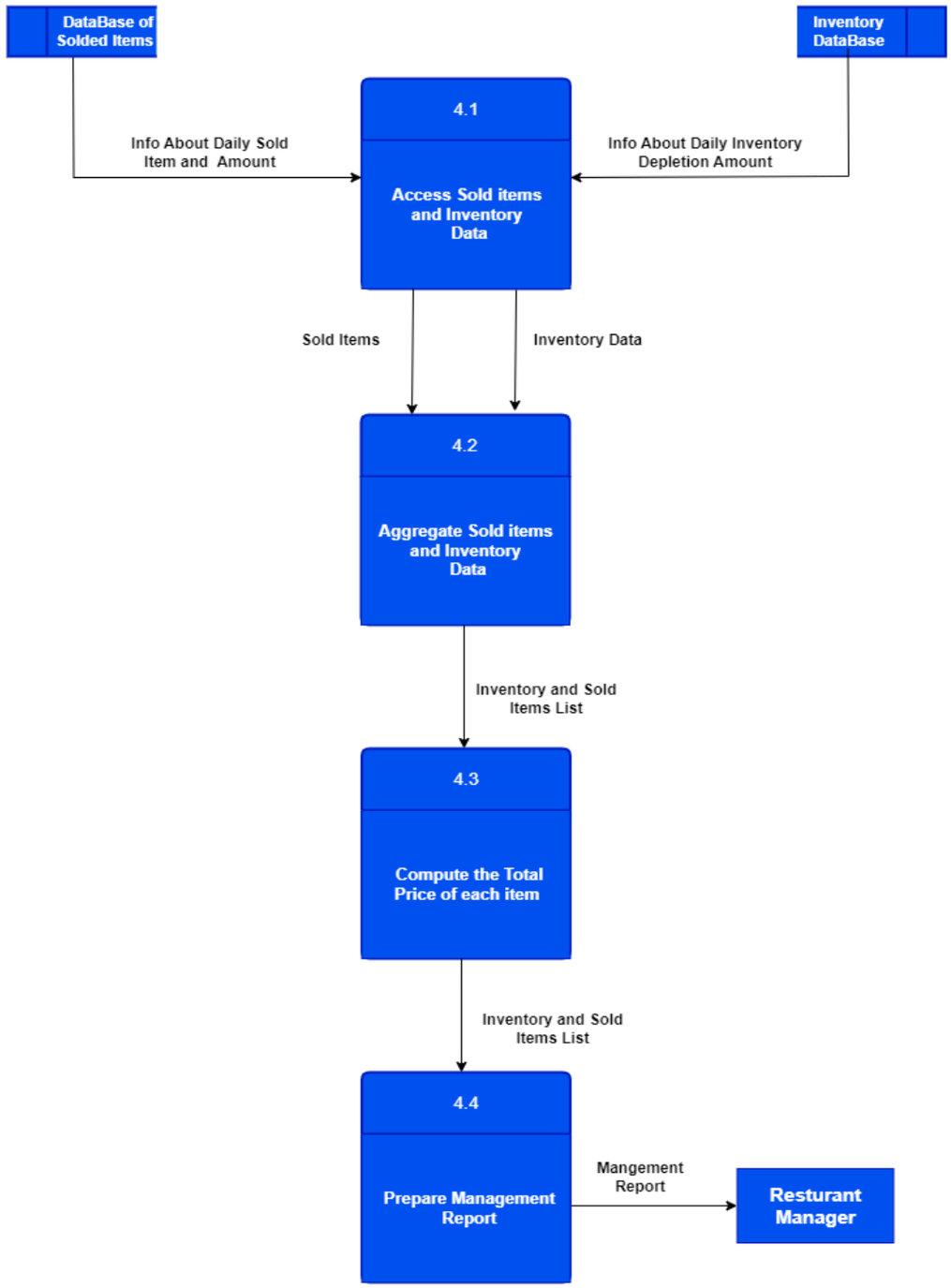

Figure 12 Level 3 DFD

5. Reference

5.1. [Online]. Available:

DESIGN_AND_IMPLEMENTATION_OF_ONLINE_FOOD_ORDERING_SYSTEM.pdf. Accessed: Nov. 19, 2023.

5.2. [Online]. Available:

"Software development life cycle models and methodologies,"
[Online]. Available: https://melsatar.wordpress.com/2012/03/15/software-development-life-cyclemodels-and-methodologies. Accessed: Oct. 19, 2023

5.3. [Online]. Available:

Available: https://www.tutorialspoint.com/sdlc/sdlc_tutorial.pdf. Accessed: Oct. 19,2023.

5.4. [Online]. Available:

"Waterfall model: Advantages and disadvantages of waterfall model,". [Online]. Available: http://www.ianswer4u.com/2011/11/advantages-and disadvantagesof.html#axzz4M06BvkOA. Accessed: Oct. 19, 2023.

5.5. [Online]. Available:

https://en.wikipedia.org/wiki/Requirements_analysis. [Accessed: 19-Nov- 2023].

5.6. [Online]. Available:

https://en.wikipedia.org/wiki/Non-functional_requirement. [Accessed:19- Nov- 2023].

5.7. [Online]. Available:

Available: https://en.wikipedia.org/wiki/Software_maintenance. [Accessed: 19-Nov-2023].

5.8. [Online]. Available:

Association use case image - Google Search", Google.com, 2016. [Online]. Available: https://www.google.com/search?q=association+use+case+image&biw=1517&bih=654&source=lnms&sa=X&ved=0ahUKEwjxnNvB29TPAhVFro8KHa22Dw4Q_AUIBSgA&dpr=0. [Accessed: 19- Nov- 2023].

5.9. [Online]. Available:
"Business requirements", En.wikipedia.org, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Business_requirements. [Accessed: 19- Nov- 2023].

5.10. [Online]. Available:
"What's is the difference between include and extend in use case diagram?", Stackoverflow.com, 2016. [Online]. Available: http://stackoverflow.com/questions/1696927/whatsis-the-difference-between-include-and-extend-in-use-case-diagram. [Accessed: 19- Nov- 2023].

5.11. [Online]. Available:
Entity relationship model - Google Search", Google.com, 2016. [Online]. Available: https://www.google.com/search?sourceid=chrome&ie=utf-8&oe=utf-8&aq=t&hl=&q=entity%20relationship%20model. [Accessed: 19- Nov- 2023].

5.12. [Online]. Available:
Sequence diagram definition - Google Search", Google.com, 2016. [Online]. Available: https://www.google.com/search?sourceid=chrome&ie=utf-8&oe=utf-8&aq=t&hl=&q=sequence%20diagram%20defination. [Accessed: 19- Nov- 2023].

5.13. [Online]. Available:
Google Search", Google.com, 2016. [Online]. Available: https://www.google.com/search?sourceid=chrome&ie=utf-8&oe=utf8&aq=t&hl=&q=usecase. [Accessed: 19- Nov- 2023].

5.14. [Online]. Available:
Google Search", Google.com, 2016. [Online]. Available: https://www.google.com/search?q=usecase&biw=1517&bih=654&source=lnms&sa=X&ved=0ahUKEwiO7Lno0dTPAhUMQY8KHS5KB24Q_AUIBygA&dpr=0.9#q=actor+use+case+diagram. [Accessed: 19-Nov- 2023].

5.15. [Online]. Available:
https://www.google.com/search?sourceid=chrome&ie=utf-8&oe=utf-8&aq=t&hl=&q=entity%20relationship%20model#q=entity-relationship+diagram+definition

5.16. [Online]. Available:
https://en.wikipedia.org/wiki/Software_maintenance