|  |  |
| --- | --- |
| A picture of a winding road and trees  **Software Testing Assignment**  **Module – 1 (Fundamental)** | Abstract  This assignment introduces key software development concepts, including SDLC, software testing, Agile methodology, SRS, and OOP principles. It also features practical use case diagrams for systems like online payments, banking, OTT platforms, and e-commerce, blending theory with real-world applications.  patelomex9201@gmail.com  Patel om r. |

**Software Testing Assignment**

**Module – 1 (Fundamental)**

1. **What Is SDLC?**

* **SDLC** Stands for **Software development life cycle**. It includes 6 Phases.
* It is a structured process used for developing software systems
* It has Different models like **Waterfall**, **Agile**, and **Spiral** apply SDLC in varied ways.

**🔄 SDLC Phases Overview**

| **Phase** | **Purpose** |
| --- | --- |
| **📋 Requirement Gathering** | Capture user needs, usage scenarios, functional & non-functional requirements. |
| **🔍 Analysis** | Define what the system must do; document clear, validated goals. |
| **🏗️Design** | Define how the system will achieve the goals; produce architecture & implementation plan. |
| **💻Implementation** | Build the system components; code development and integration. |
| **🧪 Testing** | Ensure quality through internal, unit, application, and stress testing. |
| **🔧 Maintenance** | Improve, adapt, and correct the software post-deployment. |

**2. What is Software Testing?**

**Software Testing** is the process of evaluating a software product to ensure it behaves as expected and meets all specified requirements. It involves checking whether the software performs correctly, is free from defects, and is reliable for end users.

**📌 Key Purpose of Testing:**

* To identify **errors, bugs, or missing functionalities**.
* To ensure the **quality, correctness, and completeness** of the software.
* To validate whether the system satisfies both **business** and **technical goals**.

**🧠 Why Testing Matters:**

* Testing is **not just about running code**. It includes planning, preparation, and evaluation.
* It helps **prevent defects early** in the life cycle rather than just fixing them later.
* Testing spans the entire **Software Development Life Cycle (SDLC)**—from reviewing requirements to post-release improvements.

**🧪 Types of Testing**

**📗 Static Testing**

**📘 Dynamic Testing**

**3. What Is Agile Methodology?**

**⚙️ What is Agile?**

Agile is a flexible and evolving method of developing software.  
It treats every project as unique and believes one-size-fits-all does **not** work!  
Work is divided into **small time frames** called **iterations** or **sprints**, and each sprint delivers a working piece of software.  
Each build adds new features, and the final build includes everything the customer needs.

Agile is **adaptive**, and encourages change even during development.

**✅ Pros of Agile**

* Functional software can be built and shown quickly
* Works well when requirements change frequently
* Supports parallel development and delivery
* Easy to manage and plan
* Gives developers flexibility

**⚠️ Cons of Agile**

* Long-term maintenance can be tricky
* Delivery deadlines may limit flexibility
* Heavy dependence on customer feedback
* Knowledge transfer can be challenging

**4. What Is SRS?**

**📄 Software Requirement Specification (SRS)**

* A **Software Requirements Specification (SRS)** is a complete description of how a software system should behave.
* It includes both **functional** and **non-functional** requirements.

**📌 Types of Requirements**

**🧑‍💼 Customer Requirements**

* Define the basic needs of the user or operator.
* Answer key questions like:
  + Where will the system be used?
  + What is the mission or purpose of the system?

**⚙️ Functional Requirements**

* Describe **what the system should do**.
* These are technical specifications and system behaviors.

**🛡️ Non-Functional Requirements**

* Define **how the system should perform**, not what it does.
* Categories include:
  + **Usability** – Easy to use
  + **Reliability** – Consistent performance
  + **Performance** – Fast and efficient
  + **Security** – Protected from threats

**5. What Is OOPS?**

**📄** OOPS Stands for **Object Oriented Programming**.

1. It is a way to write a code.
2. Reusability advantage.
3. Secure
4. Less code redundancy.

**6. Write Basic Concepts OOPS?**

**📄** OOPS Stands for **Object Oriented Programming**.

1. It is a way to write a code.
2. Reusability advantage.
3. Secure
4. Less code redundancy.

**📌 Concepts of class:**

1. **🧱 Class**
   * t is collection of data members & member functions.
2. **🏗️Object**
   * Object give permission to access functionality of class.
3. **🎁Encapsulation**
   * It used to wrap the data.
4. **🧠Inheritance**
   * it is used for drive attributes of some other class.
5. **🧬Polymorphism**
   * One name multiple form.
6. **🌀Abstraction**
   * Used to hiding details it shows only essential details.

**7. What is Object?**

An **object** represents a real or abstract thing with a well-defined role in a system.

Object gives permission to access functionality of class.

1. **🧠Example:** car, employee, contract, colour

**🧩 Object = Data + Methods**

* **Responsibility to Know → stores data (attributes)**
* **Responsibility to Do → performs operations (methods)**

**8. What is Class?**

🔄 A **Class** is a blueprint or template used to create objects in OOP.

* It is collection of data, data member, data function.
* It defines the **structure** and **behaviour** of objects, but doesn’t hold actual data itself.

**📌 Key Points:**

* A class is an **abstraction** of an object.
* It shows how an object should behave, but **does not exist physically**—only its instances (objects) do.

**9. What is Encapsulation?**

Encapsulation is the concept of **wrapping of data and functions**.

**📌 Key Points:**

* An object hides its internal details from other objects.
* It includes everything it needs—**privately contained** inside—only exposing necessary parts.
* In **Java**, encapsulation is implemented using **classes** or **interfaces**.

**10. What is Inheritance?**

**🧬 Inheritance**

Inheritance is where **one class inherits the properties and behaviours of another class**.

**📌 Key Points:**

* Allows creation of a new class (**child class**) from an existing class (**parent class**).
* Promotes **code reusability** by sharing common logic.
* Reduces redundancy and improves maintenance.

**11. What is Inheritance?**

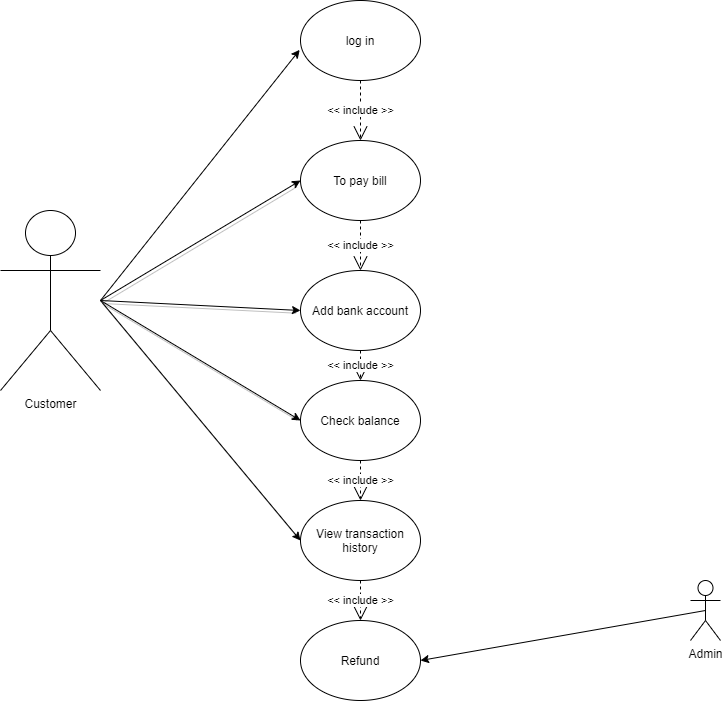
**Polymorphism** means "having many forms."  
It allows objects of different classes to respond to the **same message** in **different ways**, depending on their type.

**📌 Key Points**

* The term **poly** means “many” and **morph** means “form”.
* Polymorphism lets a single function, operator, or method act differently depending on the context.

**12. Use-case on online bill payment system?**

**📌 Paytm Use-case:**



**13.Use-case on banking system for customers.?**

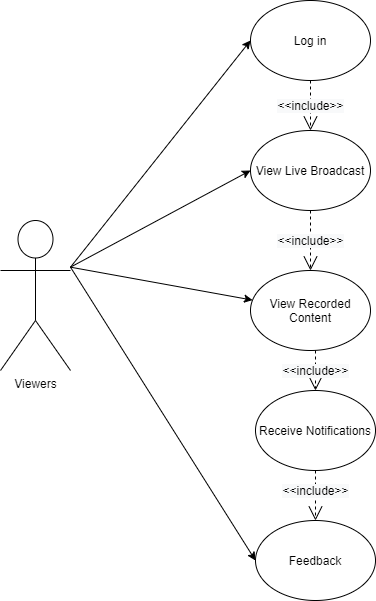
**📌 Banking System Use-case:**

A screen shot of a black and white screen

AI-generated content may be incorrect.

**14.Use-case on broadcasting system?**

**📌 Broadcast System Use-case:**



**15.Write SDLC phases with basic introduction?**

* **SDLC** Stands for **Software development life cycle**. It includes 6 Phases.

**🔄 SDLC Phases Overview**

| **Phase** | **Purpose** |
| --- | --- |
| **📋 Requirement Gathering** | Capture user needs, usage scenarios, functional & non-functional requirements. |
| **🔍 Analysis** | Define what the system must do; document clear, validated goals. |
| **🏗️Design** | Define how the system will achieve the goals; produce architecture & implementation plan. |
| **💻Implementation** | Build the system components; code development and integration. |
| **🧪 Testing** | Ensure quality through internal, unit, application, and stress testing. |
| **🔧 Maintenance** | Improve, adapt, and correct the software post-deployment. |

**16.Explain Phases of the waterfall model?**

💧 Phases of the Waterfall Model

The **Waterfall Model** is a linear and sequential approach to software development. Each phase begins only after the previous one is completed.

| **Phase** | **Description** |
| --- | --- |
| 📝 Requirement Analysis | Collect and document what the system should do. Output is the SRS document. |
| 🧩 System Design | Plan system architecture, modules, and data flow based on requirements. |
| 💻 Implementation | Developers write code as per design specifications. |
| 🧪 Testing | Identify bugs and verify that the software meets all requirements. |
| 🚀 Deployment | Deliver the final product to users; install and configure. |
| 🔧 Maintenance | Post-deployment support, updates, and issue resolution based on user feedback. |

**17.Explain Phases of the spiral model?**

**🔄 Spiral Model Phases – Summary Table**

| **Phase** | **Description** |
| --- | --- |
| 🔍 **Identification** | Define system requirements through consultation with stakeholders. Includes business needs, objectives, and constraints. |
| 🧠 **Design** | Create architectural designs and module-level specifications based on the refined requirements. |
| ⚙️ **Construction** | Develop and test the system incrementally. Prototype evaluation plays a key role here. |
| 🔁 **Evaluation & Risk Analysis** | Assess risks (technical, cost, schedule), get stakeholder feedback, and decide next steps. |

**18. Write agile manifesto principles?**

**📜 Agile Manifesto – Principles**

| **No.** | **Principle** | **Description** |
| --- | --- | --- |
| 1️⃣ | Customer Satisfaction | Deliver valuable software early and regularly to meet customer needs. |
| 2️⃣ | Welcome Change | Embrace changing requirements even late in development for competitive edge. |
| 3️⃣ | Frequent Delivery | Provide working software in short, regular intervals—preferably weekly. |
| 4️⃣ | Collaboration | Maintain daily cooperation between business and developers. |
| 5️⃣ | Build Around Motivated People | Trust and support skilled individuals to ensure successful outcomes. |
| 6️⃣ | Face-to-Face Communication | Encourage direct conversation for better understanding and faster decisions. |

**19. write agile model pros and cons?**

| **✅ Advantages** | **⚠️ Disadvantages** |
| --- | --- |
| Flexible to changing needs | Less detailed documentation |
| Rapid and continuous delivery | High dependency on team skills |
| Customer collaboration | Time-consuming meetings |
| Iterative development | Risk of scope creep |
| Early and frequent testing | Not ideal for all project types |
| Transparent progress | May lack long-term predictability |
| Supports constant improvement | Difficult in fixed-budget projects |

**20. Draw use-case on OTT Platform?**

**📌 OTT Platform Use-case:**

**A black screen with white circles and white text

AI-generated content may be incorrect.**

**21. Draw use-case on E-Commerce Platform?**

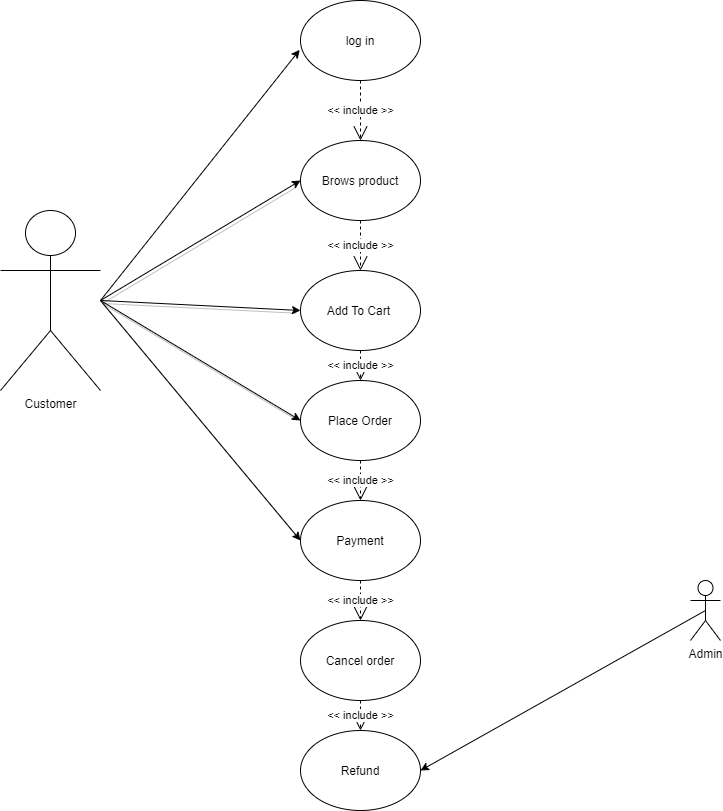
**📌 E-Commerce Platform Use-case:**

**A screen shot of a phone

AI-generated content may be incorrect.**

**22. Use-case on Online shopping product using payment gateway?**

**📌 online shopping Platform Use-case:**

****