Tops Technologies	**Name:** Patel Hardik Manharbhai **Assignment:** Software Testing
	Module – 1 (Fundamental)
Que: 1.	**What is SDLC?**
Ans.	Full form **Software Development life cycle.**
	The SDLC, or Software Development Life Cycle, is a process used to develop software from start to finish.
	It involves a series of steps, including planning, analysis, design, implementation, testing and
	maintenance, to ensure that the software is developed efficiently and effectively. Each phase produces
	specific deliverables, such as requirement specifications, design documents, and test plans, to guide the
	development process.
	These phases may vary from on organization to another, but purpose is almost all same, that is "Develop
	and Maintain Quality Software".
	• **SDLC Phases:**
	1) Requirement Gathering
	2) Analysis
	3) Design
	4) Implementation / Development
	5) Testing
	6) Deployment & Maintenance
	Software development process varies from one SDLC Model to another.
	• **SDLC MODELS:**
	1) Waterfall Model (Classical Software Cycle)
	2) V Model
	3) Iterative & Incremental Model
	4) Spiral Model
	5) Agile Model
	6) Use case
Que: 2.	**What is Software Testing?**
Ans.	**Software Testing** is process used to identify the correctness, completeness and quality of the developed
	software with respect to the client's expectation.
	Main intention is fulfilling client expectation, Defect free software, identify errors, gaps or missing
	requirements.
	It can also be stated as the **process of validating and verifying** that a software program or application or
	Product:
	1) Meets the business and technical requirements
	2) Works as expected
	3) Can be implemented with the same characteristic.

Que: 3.	**What is agile methodology?**
Ans.	Agile model is a combination of iterative and incremental process model with focus on process
	adaptability and customer satisfaction by rapid delivery of working software product.
	- Agile Methods break the product into small incremental builds.
	- In agile the task is divided to time boxes (small time frames) to deliver specific features for a release.
	- Agile thought process had started early in the software development and started becoming popular with
	time due to its flexibility and adaptability.
	- These builds are provided in iterations.
	- Each iteration typically lasts from about on to three weeks.
	- The Agile approach is designed to be adaptive and responsive to changing requirements, allowing teams
	to deliver high-quality software products quickly and efficiently. It involves breaking down the
	development process into smaller, more manageable chunks or iterations, known as sprints. Each sprint
	focuses on delivering a working product increment, which is then reviewed and refined in collaboration
	with stakeholders.
	- Some popular Agile methodologies include Scrum, Kanban, Extreme Programming (XP), and Lean
	software development.
Que: 4.	**What is SRS?**
Ans.	SRS stands for **Software Requirements Specification**, which is a document that describes the functional
	and non-functional requirements of a software system. In software testing, SRS plays a critical role in
	ensuring that the software meets the expectations of its users and clients.
	The SRS document typically includes information about the system's features, user interfaces,
	performance, reliability, and security and other technical details. The document serves as a blueprint for
	the software development team and provides a common understanding of what the software should do
	and how it should behave.
	During the software testing process, the SRS document is used as a reference point for creating test cases,
	Designing test scenarios, and validating the software against the documented requirements. Tester will
	Use the SRS to ensure that the software meets the functional and non-functional requirements specified
	In the documents.
	It includes a set of use cases also.
	• **Types of Requirements:**
	1) Customer Requirements
	2) Functional Requirements
	3) Non-Functional Requirements

Que: 5.	**What is OOPS?**
Ans.	OOPs stand for Object oriented programming, is a computer programming model that organizes software
	Design around data. Or abject, rather than function and logic.
	- identifying objects and assigning responsibilities to these objects.
	- Objects communicate to other objects by sending messages.
	- An object is like a black box.
	- The internal details are hidden.
	• **2 steps of OOP:**
	1) Making Classes: Creating, extending or reusing abstract data types.
	2) Making Objects interact: Creating objects from abstract data types and defining their relationships.
Que: 6.	**Write basic Concept of OOPs.**
Ans.	OOP is like designing software using objects. An object is like a blueprint that contains information about
	What an object.
	There is 4 Pillars of concepts of OOPs.
	1) Encapsulation
	2) Inheritance
	3) Polymorphism
	4) Data Abstraction
Que: 7.	**What is object?**
Ans.	- An object represents an individual, identifiable item, unit, or entity, either real or abstract, with well-
	defined role in the problem domain.
	- An “object” is anything to which a concept applies.
	- Two parts of an Object =Data + Methods
	- It is an instance of a class.
	That is both data and function that operate on data are bundled as a unit called an object.

Que: 8.	**What is class?**
Ans.	It contains data member and member functions with same behavioural changes.
	- When you define a class, you define a blueprint for an object.
	- This doesn't actually define any data, but it does define what the class name means, that is, what an
	Object of the class will consist of and what operations can be performed on such an object.
	- A class represents an abstraction of the object and abstracts the properties and behaviour of that
	Object.
	- Class can be considered as the blueprint or definition or a template for an object and describes the
	Properties and behaviour of that object, but without any actual existence.
	- An object is a particular instance of a class which has actual existence and there can be many objects
	For a class.
	- **Class Represents: a)** abstraction of the object.
	b) abstracts the properties.
	c) behaviour of that object.
Que: 9.	**What is encapsulation?**
Ans.	- It is wrapping up of data in single unit. E.g., Capsule
	- Encapsulation is the practice of including in an object everything it needs hidden from other objects. The
	- Internal state is usually not accessible by other objects.
	- Encapsulation = enclose **or** be enclosed in **or** if in a capsule(unit).
	- Encapsulation enables data hiding, hiding irrelevant information from the user of a class and exposing
	only the relevant details required by the user.
	- We can expose our operations hiding the details of what is needed to perform that operation.
Que: 10.	**What is inheritance?**
Ans.	- It is acquiring the property of base/parent class into derived/child class.
	There are 5 types: 1) Single Level Inheritance.
	2) Multi Level Inheritance.
	3) Multiple Level Inheritance.
	4) Hierarchical Level Inheritance.
	5) Hybrid Level Inheritance.
	- Inheritance means that one class inherits the characteristics of another class.
	- This is also called a "is a" relationship.
	- **Example:** A car is a vehicle, A dog is an Animal, A teacher is a person.

Que: 11. What is polymorphism?

Ans. - It is a combination of many forms.

Poly = Many

Morphism = Forms.

- It allows different objects to respond to the same message in different way, the response specific to the types of the object.

- The ability to change form is known as polymorphism.

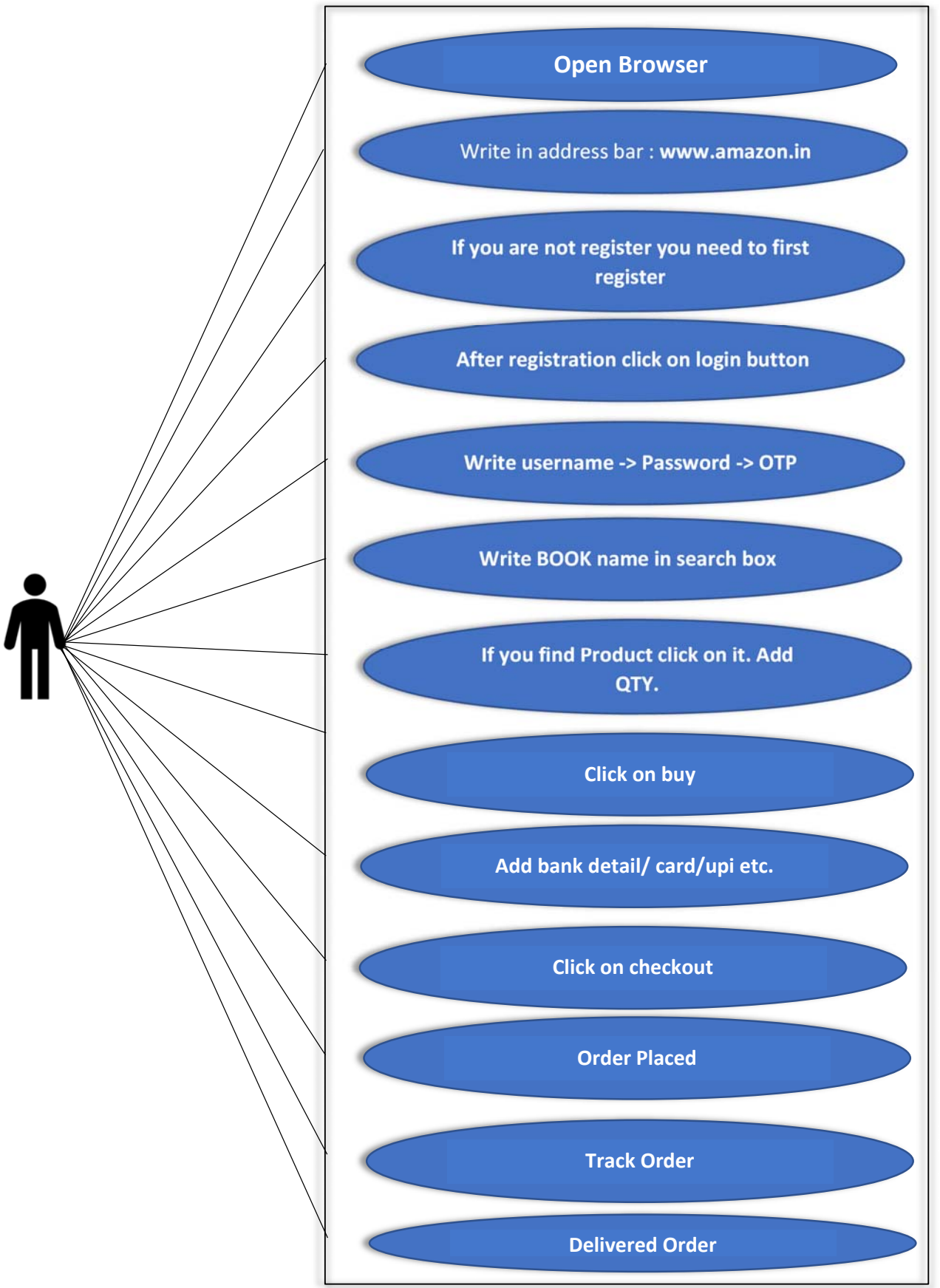
- **Two types of Polymorphism**: (a) **Compile**-time Polymorphism.

Ex: Overloading

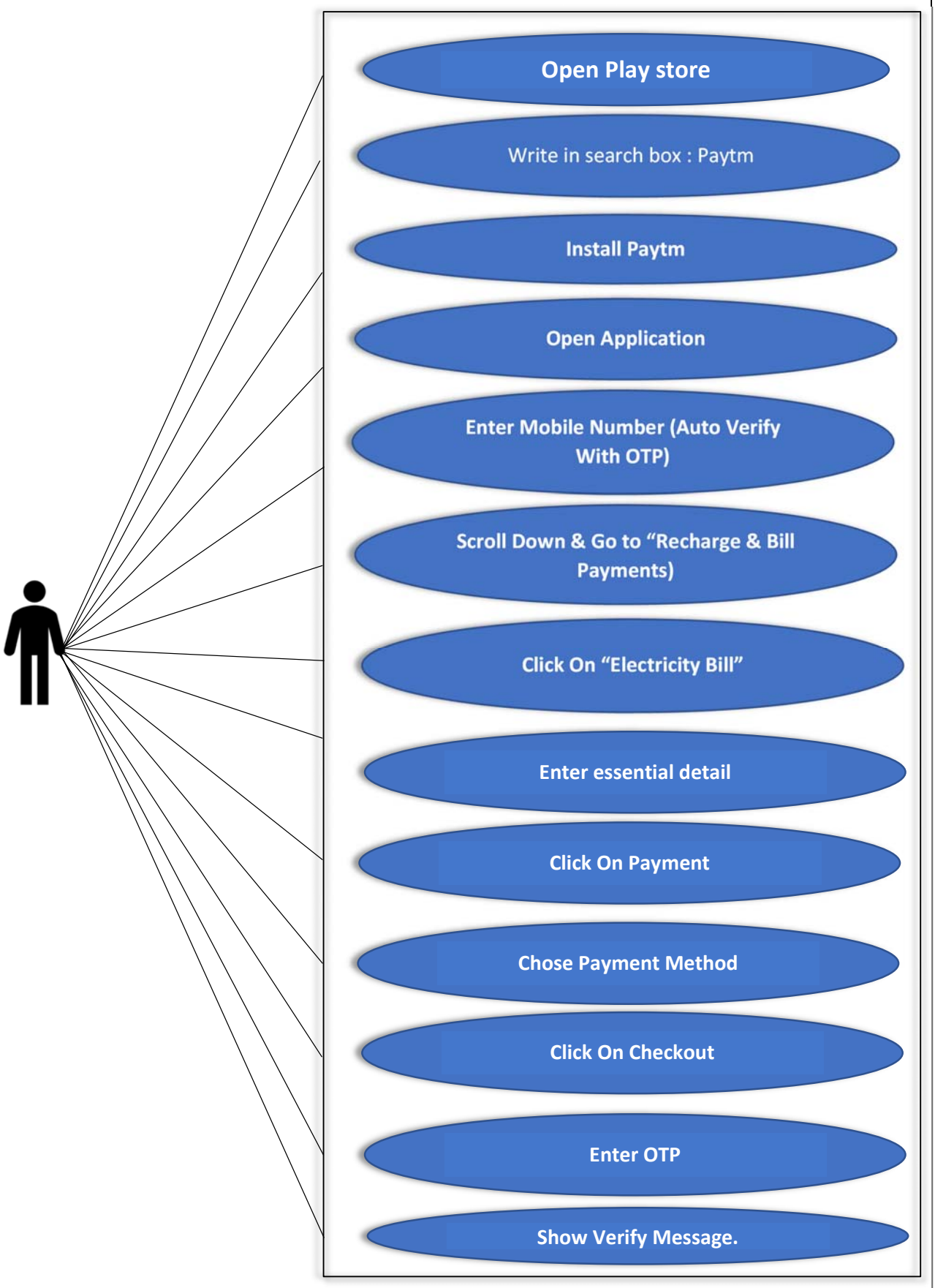
(b) **Run**-time Polymorphism.

Ex: Overriding

Que: 12.	**Draw use case on online "book" shopping.**
Ans.	**Note: In this use case I am using platform for buy BOOK is Amazon.com. (Not using application)**
	For new or first-time user



Que: 13.	**Draw use case on online bill payment system. (Paytm)**
Ans.	**Note: Electricity bill using mobile application.**



Que: 14.	**Write SDLC phases with basic introduction.**
Ans.	Full form **Software Development life cycle.**
	The SDLC, or Software Development Life Cycle, is a process used to develop software from start to finish.
	It involves a series of steps, including planning, analysis, design, implementation, testing and
	maintenance, to ensure that the software is developed efficiently and effectively.
	• **SDLC Phases:**
	1) Requirement Gathering/Collection
	2) Analysis
	3) Design
	4) Implementation / Development
	5) Testing
	6) Maintenance
	1) Requirement Gathering/Collection: -
	• Establish Customer Needs
	• Features
	• Usage scenarios
	• Requirements will change!
	• User and business need change during the project.
	• Validation is needed throughout the software lifecycle, not only when the “Final System” is
	delivered.
	• Plan for change
	• Functional and Non-Functional
	• Requirements definitions usually consist of natural language, supplemented by diagrams and tables.
	❖ **Three types of problems can arise:**
	▪ Lake of clarity
	▪ Requirements confusion
	▪ Requirements Amalgamation
	❖ **Types of Requirements:**
	1) Functional Requirements: describe system services or functions.
	2) Non-Functional Requirements: are constraints on the system or the development process.
	2) Analysis Phase: -
	- The analysis phase defines the requirements of the system. Independent of how these requirements
	will be accomplished.
	- This analysis represents that “What” phase.
	- The requirement documentaries to capture the requirements from the customer’s perspective by
	defining goals.

	- The architecture defines the components, their interfaces and behaviours.
	- This phase represents the “How” phase.
	2) A
	3) Design Phase: -
	- Design Architecture Document
	- Implementation Plan
	- Critical Priority Analysis
	- Performance Analysis
	- Test Plan
	- The requirement document must guide this decision process.
	4) Implementation/Development Phase: -
	- In the implementation phase, the team builds the components either from scratch or by composition.
	- Given the architecture document from the design phase and the requirement document from the
	Analysis phase, the team should build exactly what has been requested, though there is still room for
	Innovation and flexibility.
	- The implementation phase deals with issues of quality, performance, baselines, libraries and debugging.
	- The end deliverable is the product itself.
	5) Testing Phase: -
	- Simply stated, Quality is very important.
	- Many companies have not learned that quality is important and deliver more claimed functionality but at
	a lower quality levels.
	- It is a much easier to explain to a customer why there is a missing feature than to explain to a customer
	why the product lacks quality.
	- A customer satisfied with the quality of a product will remain loyal and wait for new functionality in the
	Next version.
	• **Types of testing:** Regression testing, Internal testing, Unit testing, Application testing, Stress Testing.
	- The testing phase is a separate phase which is performed by a different team after the implementation is
	Completed.
	6) Maintenance Phase: -
	- Is the process of enhancing and optimizing deployed software, as well as fixing defects.
	- Software maintenance is also one of the phases in the SDLC, as it applies to soft. development. Is the
	phase which comes after deployment of the software into the field.
	- Configuration and Version Management
	- Updating all analysis, design and user documentation.

- Maintenance is the process of changing a system after it has been deployed.
    - **Corrective maintenance:** identifying and repairing defects
    - **Adaptive maintenance:** adapting new platforms.
    - **Perfective maintenance:** implement new requirements.
- In spiral model -> first prototype can be considered “Maintenance”

Que: 15. Explain Phases of the waterfall model.

Ans. **Waterfall Model = Classical Software Cycle**

- Waterfall Model Phases:

- Requirement must be “frozen” to early in the life cycle.
- Requirement are validated too late.
- It is a traditional model.

1) Requirements Collection:

- Done by Business Analysts and product analysts.
- Gathering requirements.
- Translates business language into software language.
- Ex., let us consider the example of a banking software.

2) Analysis:

- In this phase Business Requirements are converted as Software Requirements.
- This analysis phase defines the requirements of the system.
- This phase defines the problem that the customer is trying to solve.
- This analysis represents the “What” phase.
- This phase represents the “How” phase.
- The deliverable design document is the architecture.

3) Design:

- Design architecture document
- **There are 2 stages in design:**
    - i) HLD – High Level Design
    - ii) LLD – Low Level Design

HLD – gives the architecture of the software product to be developed and is done by architects and sr. dev.

LLD – Done by Sr. Developers. It describes how each and every feature in the product should work and how every component should work. Here, only the design will be there and not the code.

For ex. let us consider the example of building a house.

4) Implementation / Development / Coding:

- Done by developer
- This is the process where we start building the software and start writing the code the product.
- The implementation phase deals with issues of quality, performance, baselines, libraries and Debugging.
- The end deliverable is the product itself.

5) Testing:

- In this stage, system will be tested by testers, if the find any mismatch they report defects.

Developers/Programmers fix the defects and then testers close defects by performing confirmation testing (Regression Testing).

- Simply stated, Quality is very important.
- Many companies have not learned that quality is important and deliver more claimed functionality but at a lower quality level.
- It is a much easier to explain to a customer why there is a missing feature than to explain to a customer because the product lacks quality.

6) Release & Maintenance:

- Release team (consists of a few developers, testers and tech-support people etc..) install software

In customer environment and they consider below factors;

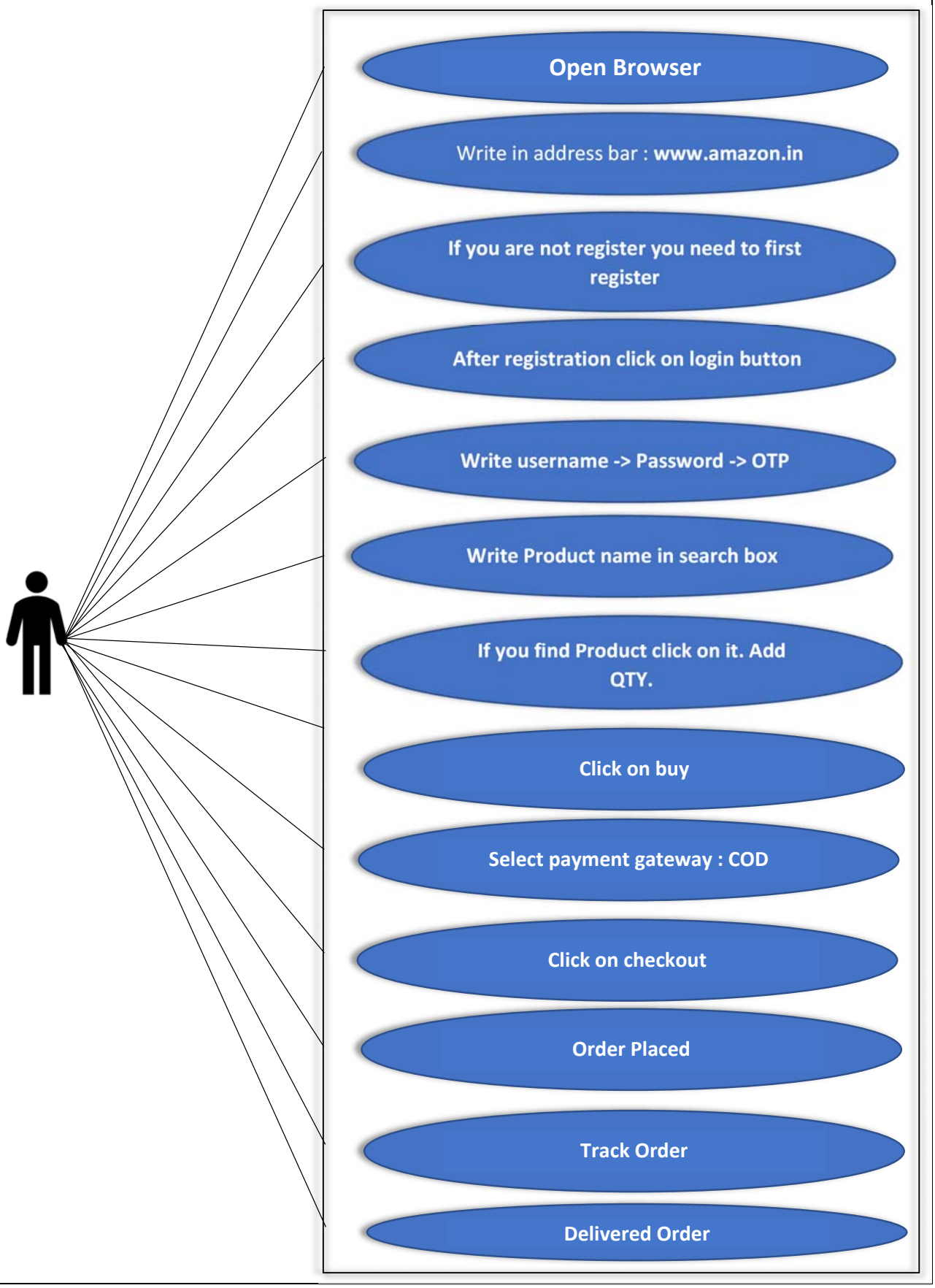
Correct and complete installation, user management, services, management, handling of input and Output devices, and handling of secondary storage devices.

- Maintenance team process customer issues based on service agreements.
- Configuration and Version Management
- Updating all analysis, design and user documentation.

Que: 16.	**Write phases of spiral model.**
Ans.	Spiral Model is very widely used in the software industry as it is in synch with the natural development
	Process of any product.
Spiral Model Phases	
Que: 17.	**Write agile manifesto principles.**
Ans.	**Four Manifest in Agile Model:**
	1) Individual and interaction
	2) Working / Demo Software
	3) Customer / Client Collaboration
	4) Responding to change. (Always Ready for changes)

Que: 18.	**Explain working methodology of agile model and also write pros and cons.**
Ans.	Agile model is a combination of iterative and incremental process model with focus on process
	adaptability and customer satisfaction by rapid delivery of working software product.
	- Agile Methods break the product into small incremental builds.
	- In agile the task is divided to time boxes (small time frames) to deliver specific features for a release.
	- Agile thought process had started early in the software development and started becoming popular with
	time due to its flexibility and adaptability.
	- These builds are provided in iterations.
	- Each iteration typically lasts from about on to three weeks.
	- Agile methodology encourages frequent communication and collaboration between team members and
	stakeholders, including customers, developers, and testers. It values the ability to respond to changing
	requirements and feedback quickly, allowing teams to deliver value to customers more efficiently.
	Pros: 1) easy to manage
	2) Is a very realistic approach to software development
	3) Good model for environments that change steadily.
	4) Resource requirements are minimum.
	5) Give flexibility to developers.
	6) little or no planning required.
	7) Delivers early partial working solutions.
	8) Promotes teamwork and cross training.
	Cons: 1) Not suitable for handling complex dependencies.
	2) More risk of sustainability, maintainability and extensibility.
	3) There is very high individual dependency, since there is minimum documentation generated.
	4) An overall plan, an agile leader and agile PM practice is a must without which it will not work.
	5) Transfer of technology to new team members may be quite challenging due to lack of
	documentation.

Que: 19.	**Draw use case on Online shopping product using COD.**
Ans.	**Note: In this use case I am using platform for buy "Product" is Amazon.com. (Not using application)**
	For new or first-time user



Que: 20.	Draw use case on Online shopping product using payment gateway.
Ans.	Note: In this use case I am using platform for buy “Product” is Amazon.com. (Not using application)
	For new or first-time user