1. **What is SDLC ?**

* SDLC is a Software Development Life Cycle.
* SDLC is a structure imposed on the development of a software product that defines the process for planning, implementation, testing, documentation, and ongoing maintenance and support.
* A Software Development Life Cycle is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of an application or piece of software.
* Without using an exact life cycle model, the development of a software product would not be in a systematic and disciplined manner.
* There are 6 SDLC Phases:
* Requirement gathering
* Analysis
* Design
* Implementation
* Testing
* Maintenance

1. **What is software testing?**

* Software testing is a process used to identify the correctness, completeness, and quality of developed computer software.
* Testing is the process of evaluating a system or its component with the intent to find that whether it satisfies the specified requirements or not.
* Testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements.
* It can be also be stated as the process of validating and verifying that a software program or application or product:
* Meets the business and technical requirements that guided it’s design and development.
* Work as expected.
* Can be implemented with the same characteristics.

1. **What is agile methodology?**

* Agile SDLC model is a combination of iterative and incremental process with focus on process adaptability and customer satisfaction by rapid delivery of working software product.
* Agile method break the product into small increment builds.
* These builds are provide in iterations.
* Each iteration typically lasts from about one to three weeks.
* Each iteration involves cross functional teams working simultaneously on various areas like planning, requirement analysis, design, coding, unit testing and acceptance testing.
* At the end of the iteration a working product is displayed to the customer and important stakeholder.
* In agile the tasks are divided to time boxes to deliver specific features for a release.
* Agile becoming popular with time due to it’s flexibility and adaptability.
* **Pros:**
* Little or no planning required.
* Easy to manage.
* Gives flexibility to developers.
* Suitable for fixed or changing requirements.
* Good model for environments that change steadily.
* **Cons:**
* Not suitable for handling complex dependencies.
* More risk of sustainability, maintainability and extensibility.
* An agile leader and agile PM (Project Manager) practise is a must without which it will not work.
* Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
* Transfer of technology to new team members may be quite challenging due to lack of documentation.

1. **What is SRS?**

* SRS stands for Software Requirement Specification.
* A Software Requirement Specification is a complete description of the behavior of the system to be developed.
* The SRS is a specification for a specific software product, program or set of applications that perform particular function in a specific environment.
* It includes a set of use cases that describe all of the interaction that user will have with the software.
* Use cases are also known as functional requirements and SRS also contain non-functional requirements.
* Requirements are categorized in several way, There are some common categorization of requirements that relate to technical management:
* Customer Requirements
* Functional Requirements
* Non-functional Requirements

1. **What is oops?**

* Oops is Object oriented Programming System.
* It is also known as a Black box testing and Functional testing.
* It is used to identifying object and assigning responsibilities to these object.
* Object communicate to other objects by sending messages.
* Messages are received by the methods of an object.
* An object is derived from abstract data type.

1. **Write Basic Concepts of oops.**

* Object
* Class
* Encapsulation
* Inheritance
* Polymorphism
* Overriding
* Overloading
* Abstraction

1. **What is object?**

* An object is an instance of an class.
* An object represents an individual, identifiable item, or entity real or abstract, with a well-defined role in the problem domain.
* An object is anything to which a concept applies.
* An object is used to create an memory of an class and also access the whole properties of an class except private.
* There are two parts of an object:
* Data
* Method (Function)
* This both data and method (function) operate on data are bundled as a unit called as object.
* Syntax:

class\_name object\_name = new class\_name();

1. **What is class?**

* Ans. Class is a blueprint for an object.
* A class is a collection of data member (variable) and member function (process, method) with its behaviors.
* A class represent an abstraction of the object and abstract the properties and behavior of that object.
* An object is a particular instance of a class which has actual existence and there can be many object for a class.
* There are two steps of Object Oriented Programming :
* Making classes: Creating, Extending or reusing abstract data types.
* Making object: Creating object from abstract data types and defining their relationships.
* **Syntax:**

class classname

{

Data member;

Member function;

}

* **Example:**
* In the case of car or laptop, there will be a blueprint or design created first and then the actual car or laptop will be built based on that.
* We do not actually buy these blueprint but the actual objects.

1. **What is encapsulation?**

* Encapsulation is define as the wrapping up of data and information into a single unit.
* 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 enables data hiding, hiding irrelevant information from the users of a class and exposing only the relevant details required by the user.
* We can protect the internal state of an object by hiding its attributes from the outside world, and then exposing them through setter and getter method.

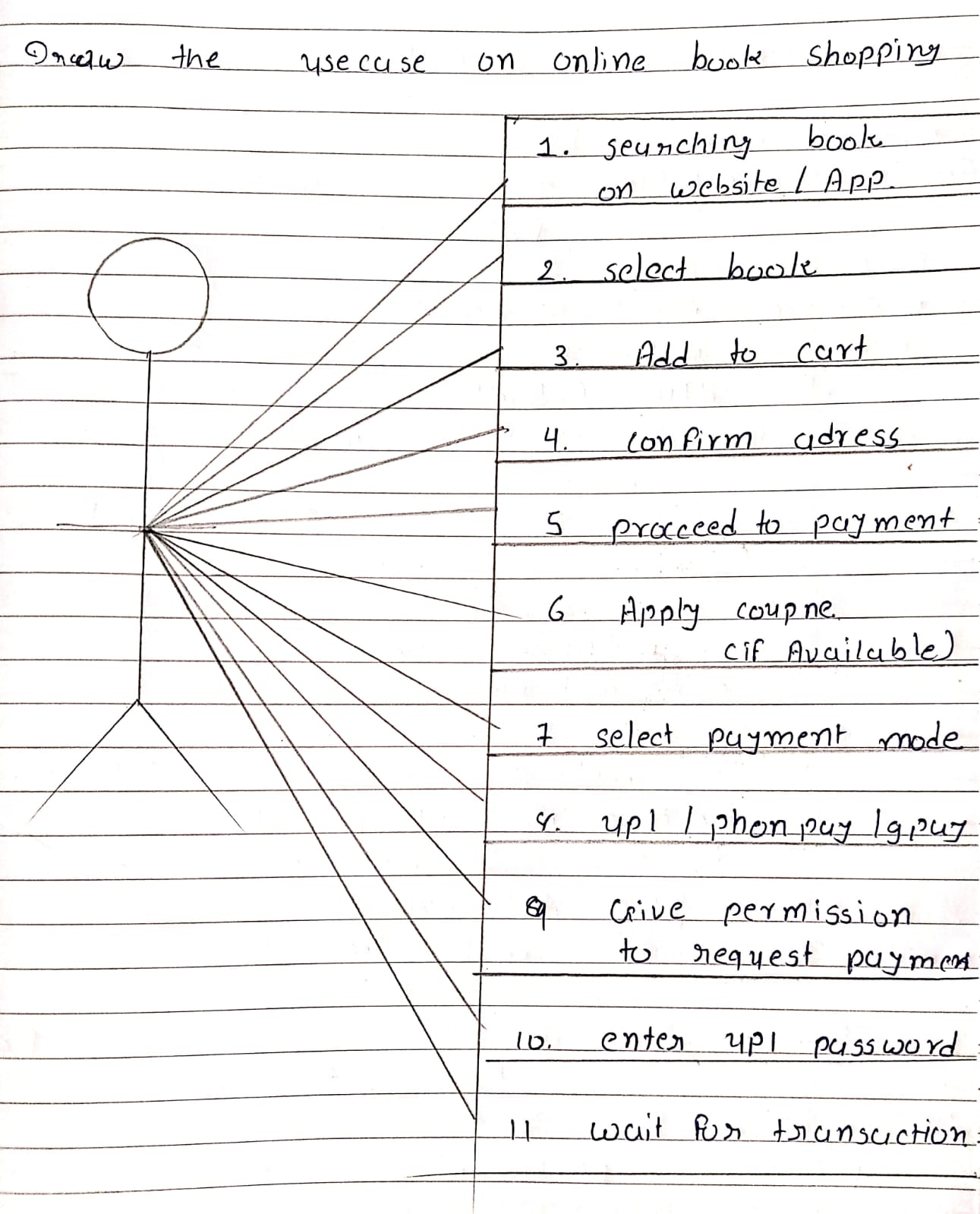
**10.What is inheritance?**

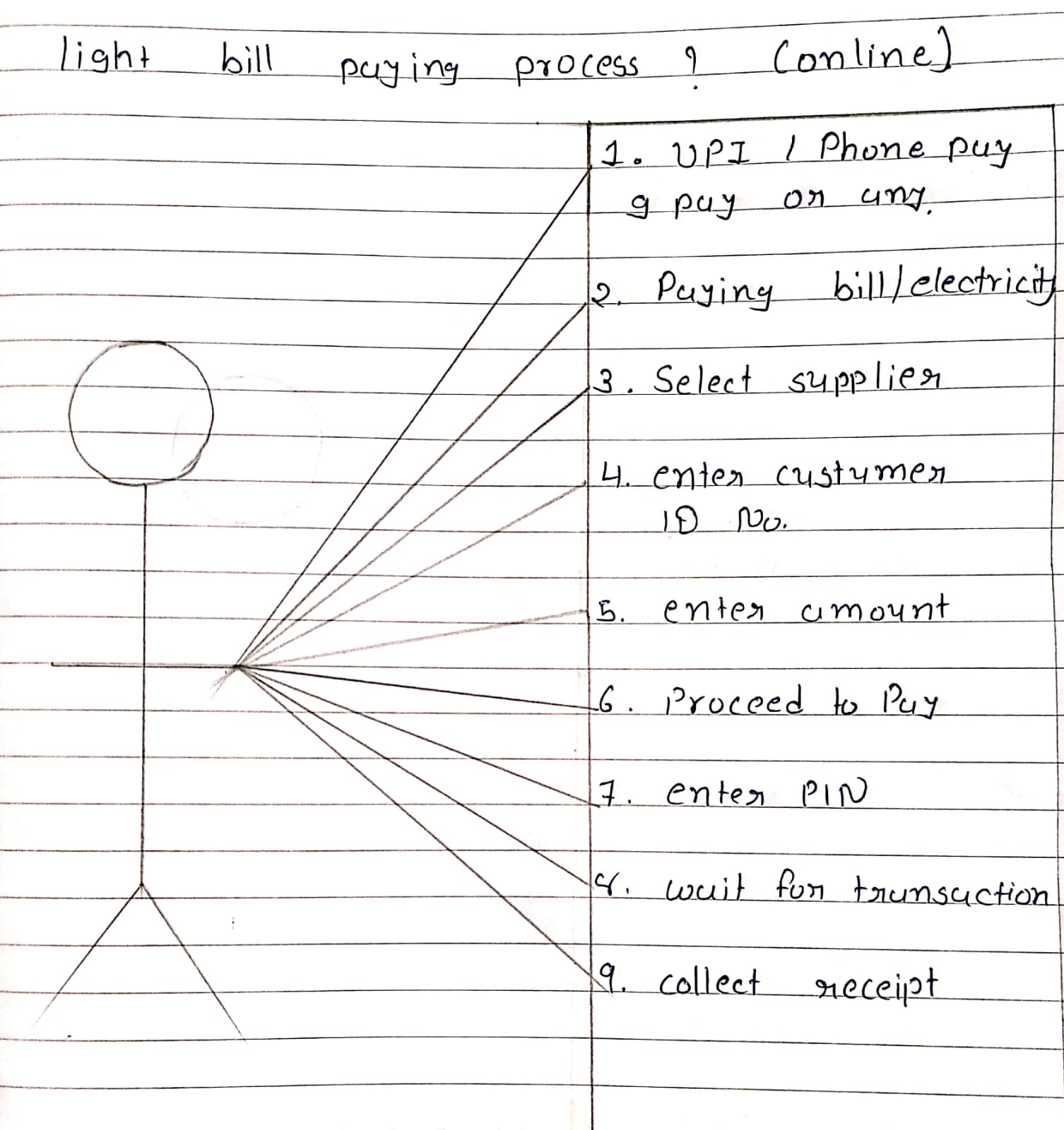
* inheritance is a process in which the properties of parent class extends into child class.
* Inheritance means that one class inherits the characteristics of another class. This is called “is-a” relationship.
* One of the most useful aspects of object-oriented programming is code reusability. As the name suggest inheritance is the process of forming a new class from an existing class called as base class, new class is formed called as derived class.
* Inheritance describes the relationship between two classes. A class can get some of its characteristics from a parent class and then add unique features of its own.
* There are mainly 5 types:
* Single inheritance
* Multilevel inheritance
* Hierarchical inheritance
* Multiple inheritance (java does not support)
  + java supports multiple inheritance only through interface.
* Hybrid inheritance (java does not support)

**11. What is polymorphism?**

* polymorphism means “ability to take one name having different forms”.
* Poly refers too many. That is a single function or an operator functioning in many ways different upon the usage is called polymorphism.
* It allows different objects to respond to the same message in different way, the response specific to the type of the object.
* The most important aspect of an object is its behavior (the things it can do).
* There are two types of polymorphism in java:
* Compile time polymorphism (function overloading).
* Runtime polymorphism (function overriding).

**12. Draw Use-case on Online book shopping.**

****

**13.Draw Use-case on online bill payment system (paytm)**

**14.Write SDLC phases with basic introduction.**

**1.Requirement gathering**: This phase involve gathering information about the software requirements from the client.

**2.Analysis**: In this phase, Model and specify the requirements “what”.

**3.Design**: In this phase, the software design is created, which includes the overall architecture of the software.

**4.Implementation:** In this phase, the design is then implemented in code.

**5.Testing:** The software is thoroughly tested to ensure that it meets the requirements and works correctly.

**6.Maintenance:** This phase includes ongoing support, bug fixes, and updates the software.

**14.Explain phases of the waterfall model.**

**1. Requirement gathering:** The aim of this phase is to understand the exact requirement of the customer and to document them properly.

* Both the customer and the software developer work together so as to document all the functions, performance of the software.
* **There are three types of problem can arise in this phase:**

**1.Lack of clarity:** It is hard to write document that are both precise and easy-to-read.

**2.Requirements confusion:** Functional and Non-functional requirements tend to be intertwined.

**3.Requirements amalgamation:** Several different requirements may be expressed together.

* **There are two types of Requirements:**

**1.Functional requirements:** It describe system services or function.

**2.Non-functional requirements:** it describe constraints on the system or development process.

**2. Analysis:** The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished.

* This phase defines the problem that the customer is trying to solve.
* The deliverable result at the end of this phase is requirement document.
* This analysis represent the “what” phase.
* The requirement documentaries to capture the requirements from the customer’s perspective by defining goals.
* The design may include the usage of existing components.

**3. Design:** It helps in specifying software and hardware requirements for product design.

* It also helps in the overall architecture of the system design.
* It is also helpful in transforming the SRS document into a functional design and development of the software product.
* So we can say that in designing phase, one makes the overall architecture for the development project.

**4. Implementation:** In the implementation phase, the team builds the components either from scratch or by the 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, tough there is still room for innovation and flexibility.
* Example: A component may be narrowly designed for this particular system, or the component may be made more general to satisfy a reusability guideline.

**1.Implementation – code**

**2.Critical Error Removal**

* The implementation phase deals with issues of quality, performance, baseline, libraries, and debugging.

**5. Testing:** Testing verifies that the product developed in the implementation phase fulfils the entire project’s requirements.

* Simply stated, quality is very important.
* It is much easier to explain to a customer why there is a missing feature than to explain to a customer why the product lacks quality.
* Quality is a distinguishing attribute of a system indicating the degree of excellence.
* 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 implementation is completed.

**6. Maintenance:** Software maintenance is one of the activities in software engineering, and is the process of enhancing and optimizing developed software, as well as fixing defect.

* The maintenance phase is the phase which comes after deployment of the software into the field.
* It’s also a phase to identify any errors you might have missed during the testing phase.
* Maintenance is the process of changing a system after it has been deployed.
  + - **Corrective maintenance:** identifying and repairing defects.
    - **Adaptive maintenance:** adapting the existing solution to the new platforms.
* **Perfective Maintenance:** implementing the new requirementsIn a spiral lifecycle, everything after the delivery and deployment of the first prototype can be considered “maintenance”!

**15.Write phases of spiral model.**

**1.Planning:** This phase includes understanding system requirements through ongoing communication between the customer and system analysts.

* At the end of the spiral, the product will be deployed in the identified market. This includes iteration cost, schedule, and resource estimates.
* This includes understanding system requirements for ongoing communication between system analysts and customers.

**2.Risk Analysis**: After the “plan” phase, the team prepares for the “risk” phase.

* During the “risk” phase, the team evaluates various aspects of the current state of the product, such as the state of its code, the state of its design, and the state of its prototype.
* The team then makes adjustments to the current state of the product based on the changes made in the “plan” phase, and then follows up with a “sales” phase to collect customer feedback.
* Risk analysis involves identifying, estimating and monitoring  technical feasibility and management risks.
* After testing the build, at the end of the first iteration, customers rate the software and provide feedback.

**3.Engineering:** In the next prototypes are built and tested. This step includes architectural design, module design, physical product design and final design. Convert the proposals made in the first two quadrants into usable software.

* This phase also includes the actual implementation of features in a project which are verified by performing testing.
* It performs the actual development and testing if the software takes place in this phase.

**4.Customer Evolution**: In this phase, the software is evaluated by the customer and feedback is given.

* The team prepares for the next phase of the planning process.
* The next phase of the planning process is known as the “spiral” phase.

**16.Write agile manifesto principles.**

1. Individual interaction

2. Working software

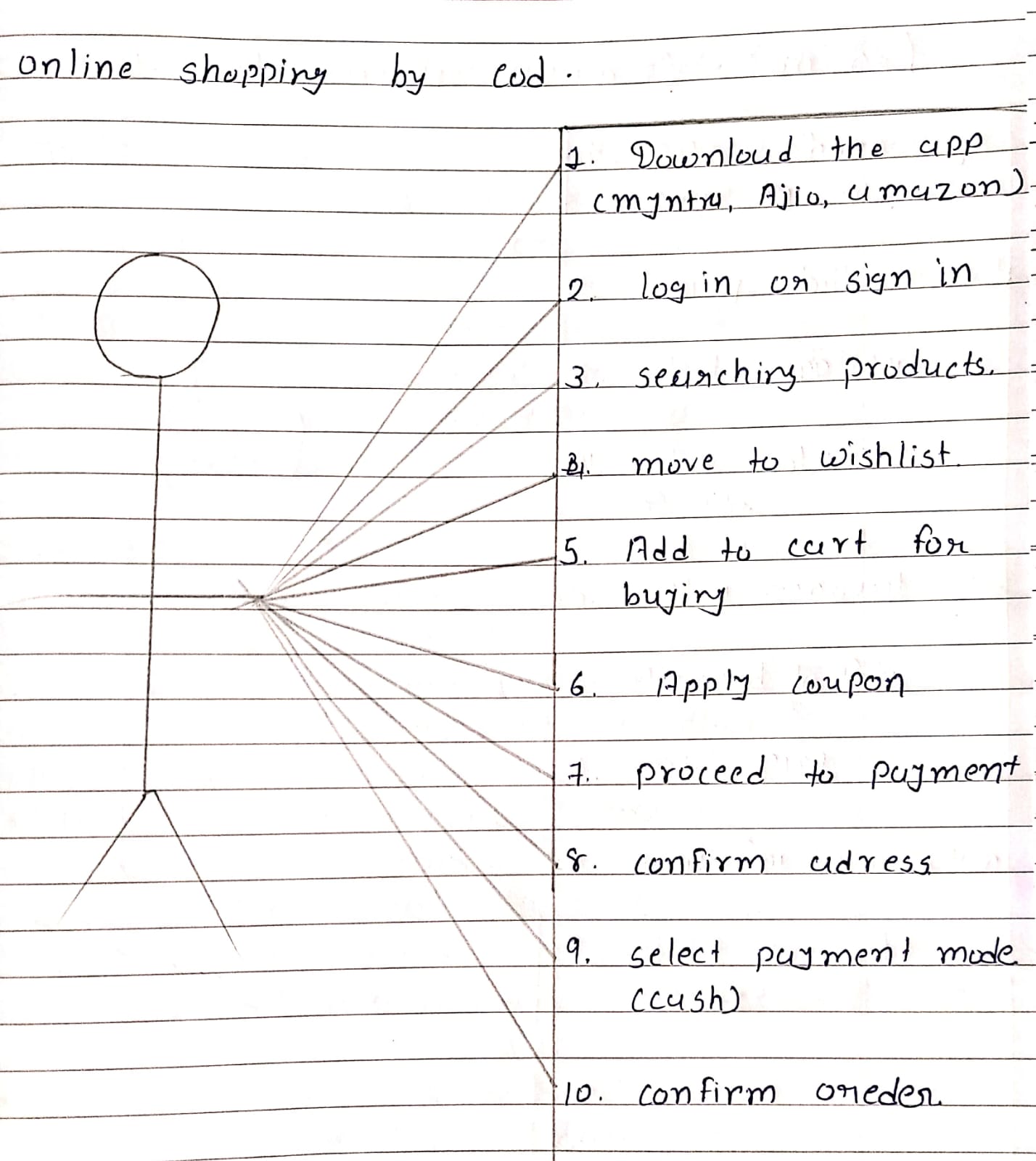
3. Customer collaboration

4. Responding to change

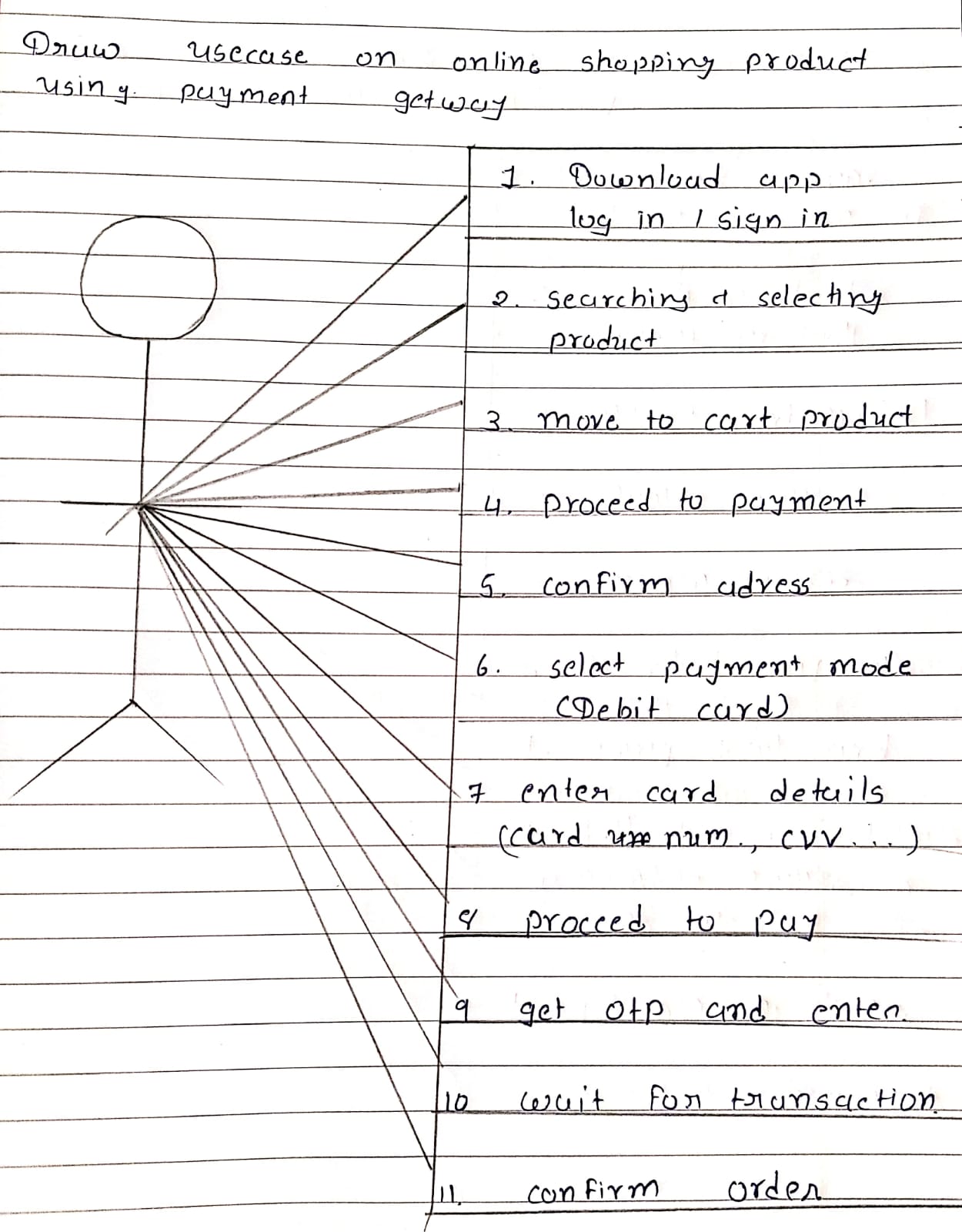
**17.Explain working methodology of agile model and also write pros and cons.**

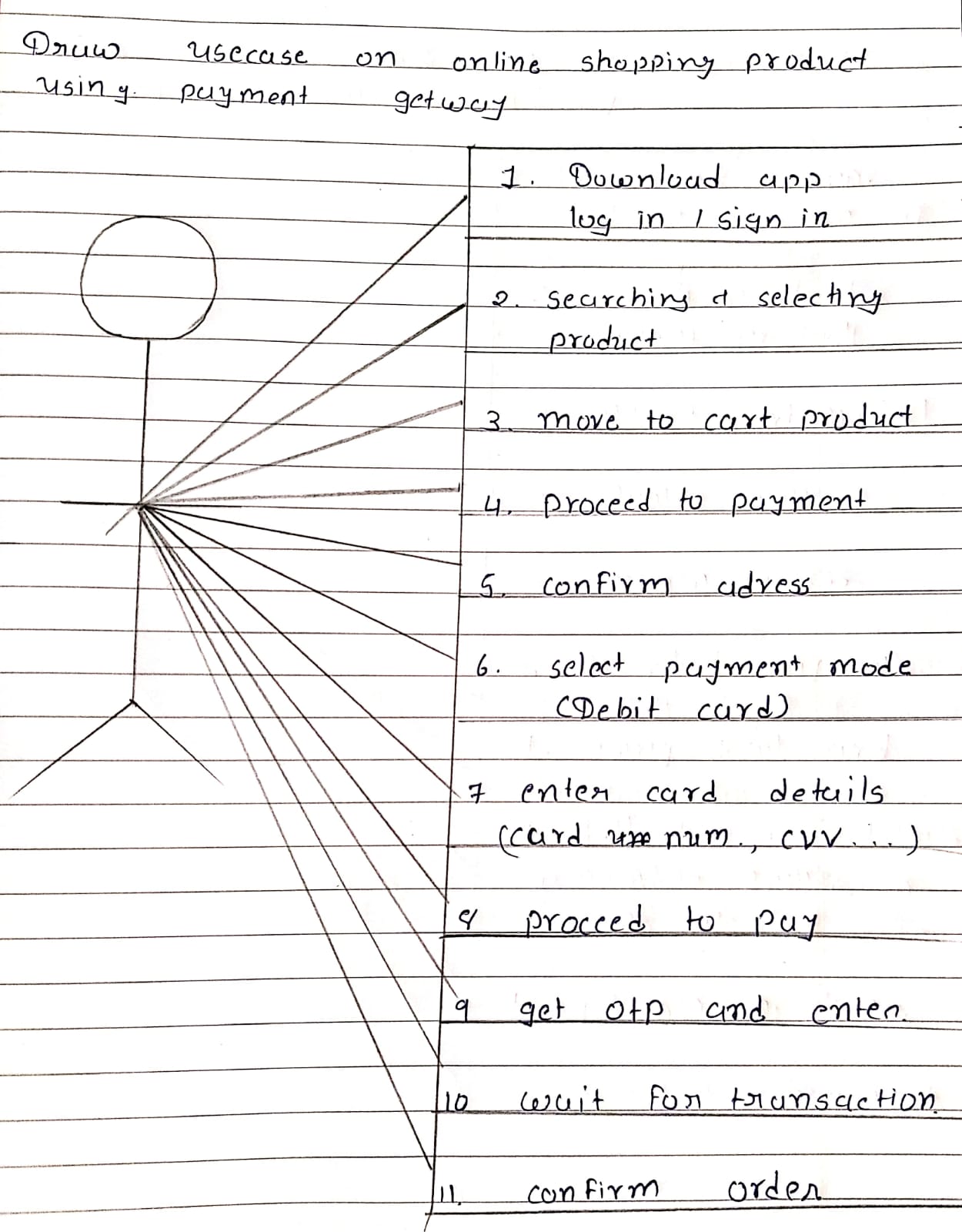
* Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.
* Agile Methods break the product into small incremental builds.
* These builds are provided in iterations.
* Each iteration typically lasts from about one to three weeks.
* Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.
* At the end of the iteration a working product is displayed to the customer and important stakeholders.
* **Pros:**
* Is a very realistic approach to software development
* Promotes teamwork and cross training.
* Functionality can be developed rapidly and demonstrated.
* Resource requirements are minimum.
* Suitable for fixed or changing requirements.
* Delivers early partial working solutions.
* Good model for environments that change steadily.
* Minimal rules, documentation easily employed.
* Enables concurrent development and delivery within an overall planned context.
* Little or no planning required.
* Easy to manage.
* Gives flexibility to developers.
* **Cons:**
* Not suitable for handling complex dependencies.
* More risk of sustainability, maintainability and extensibility.
* An overall plan, an agile leader and agile PM practice is a must without which it will not work.
* Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
* Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
* There is very high individual dependency, since there is minimum documentation generated.
* Transfer of technology to new team members may be quite challenging due to lack of documentation.

**18.Draw use-case on Online shopping product using COD.**

****

**19.Draw use-case on Online shopping product using payment gateway.**

****

****