PRACTICAL 1

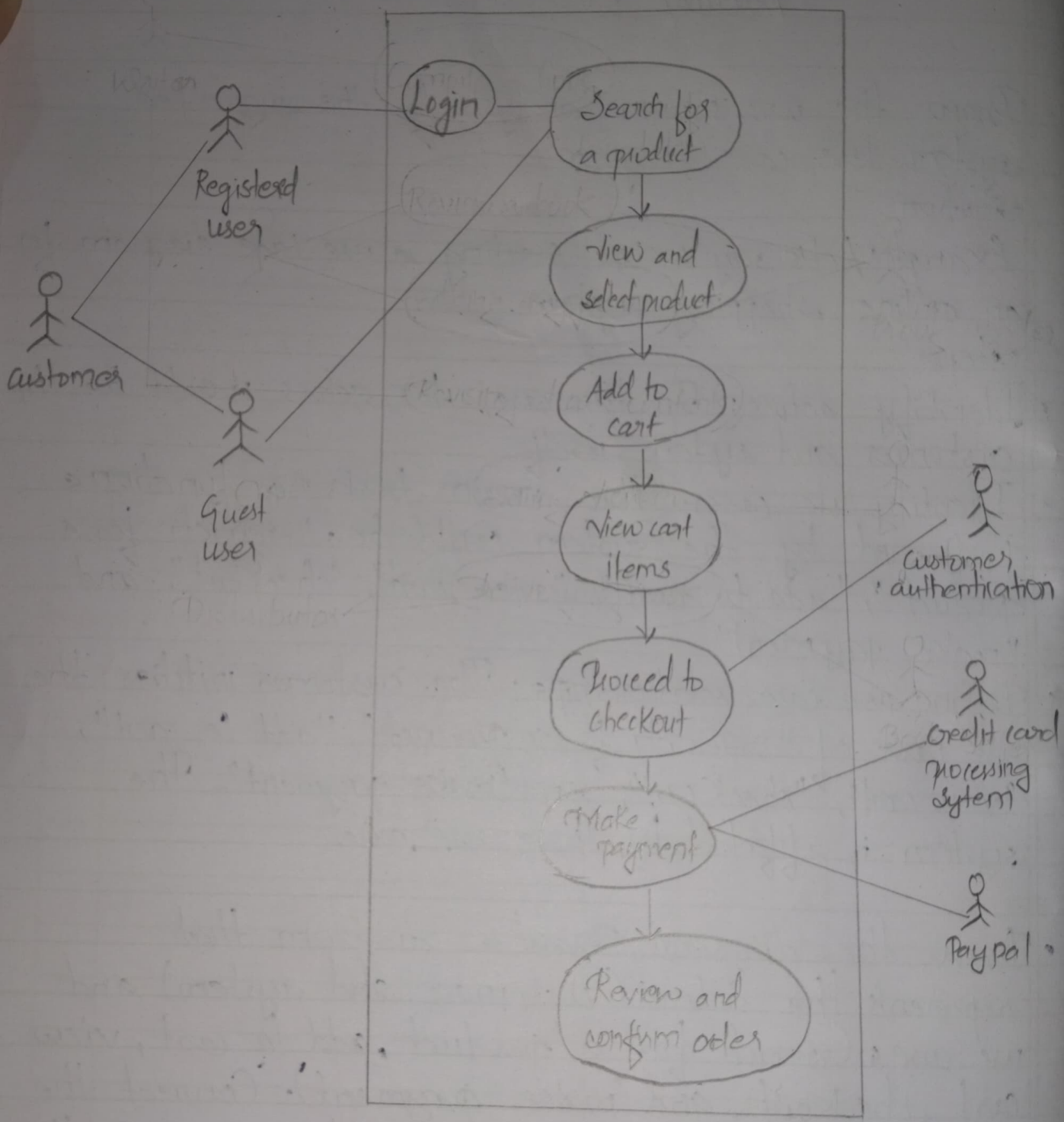
AIM:1. Draw the user's view analysis for the suggested system: Use case diagram

Solution

Example: Let's say we're creating a use case diagram for an online shopping system

Steps:

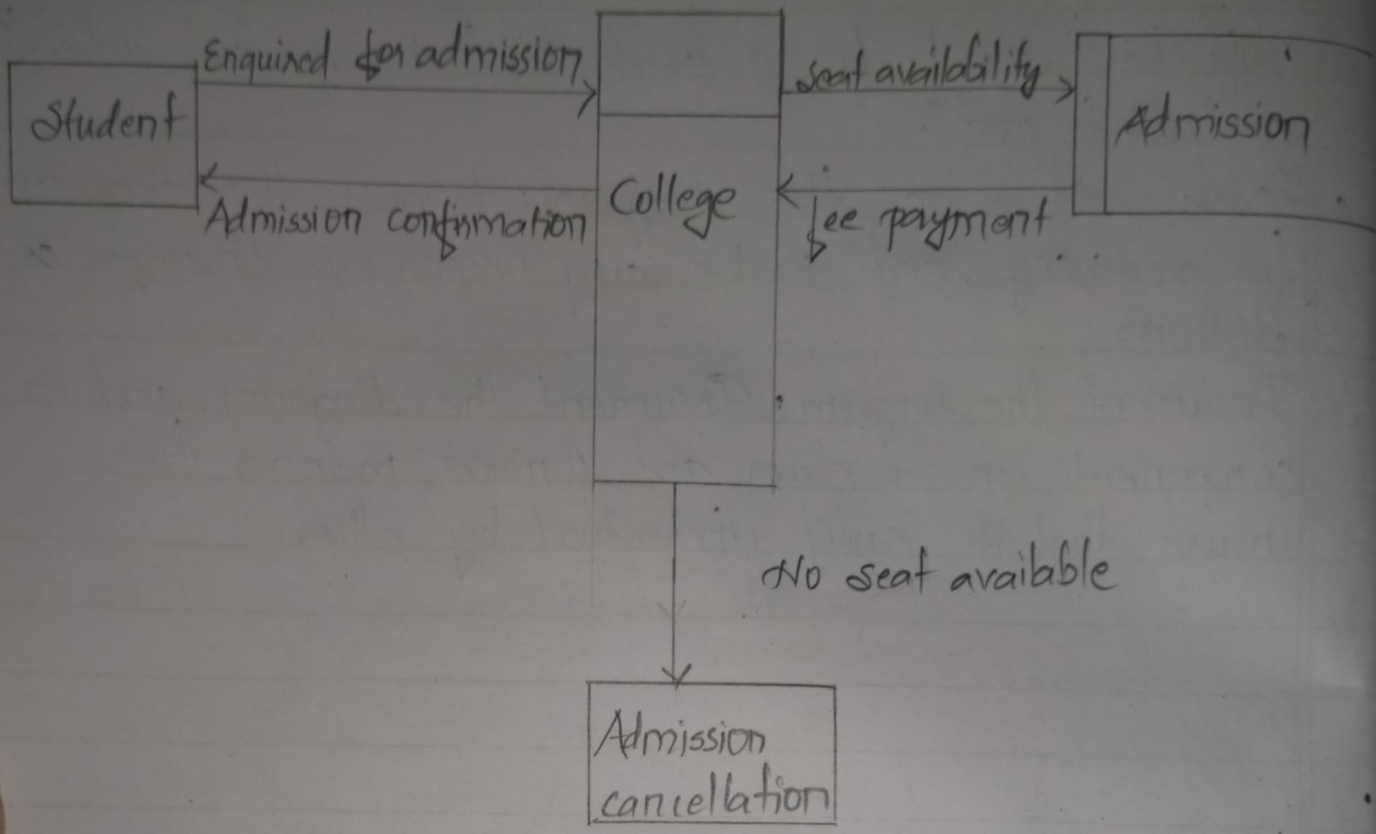
- Identify actors: In this case, the actors would be the customer and system itself.
- Identify use cases: The main tasks or functions performed by the system could be: "search for a product", "add to cart", "view cart", "checkout", and "make payment".
- Define use case relationships: The customer initiates the use case of "search for a product", "add to cart", "view cart", "checkout", and "make payment". The system is affected by these use case.
- Draw the diagram: Draw a diagram that represents the actors (customers and system) and use cases (search for a product, add to cart, view cart, checkout, and make payment). Connect the customer and use cases with lines to represent the relationships between them
- Add details to the use cases: Add additional detail to the use cases, such as the steps involved in each use case and any inputs or outputs. For example, the steps involved in the "search for a product" use case


Login
Search for a product
View and select product
Add to cart
View cart items
Proceed to checkout
Make payment
Review and confirm order
Registered user
customer
Guest user
customer authentication
Credit card processing system
Paypal

could be: "Search for a product", "view product details", and "add to cart".

- Review and refine the diagram: Review the diagram for accuracy and make any neccesary changes or additions.
- Document the diagram: Document the diagram and its components in a clear and consice manner to ensure that its easily understood by others.

4


Student
Enquired for admission
Admission confirmation
College
Seat availability
Fee payment
Admission
No seat available
Admission cancellation

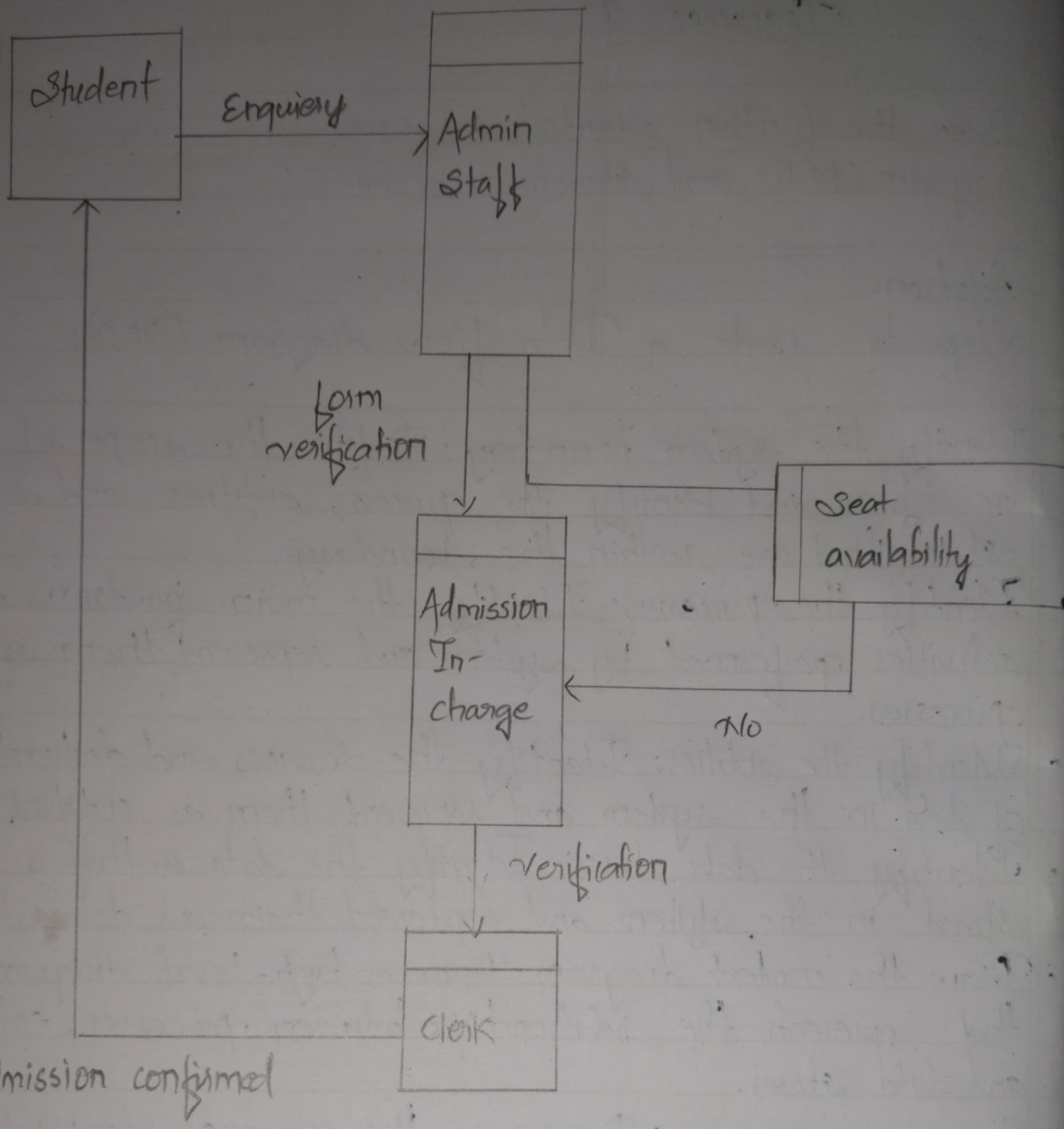
PRACTICAL 2

AIM: Draw the funtion oriented diagram: Data flow diagram (DFD) and structured chart.

Solution:

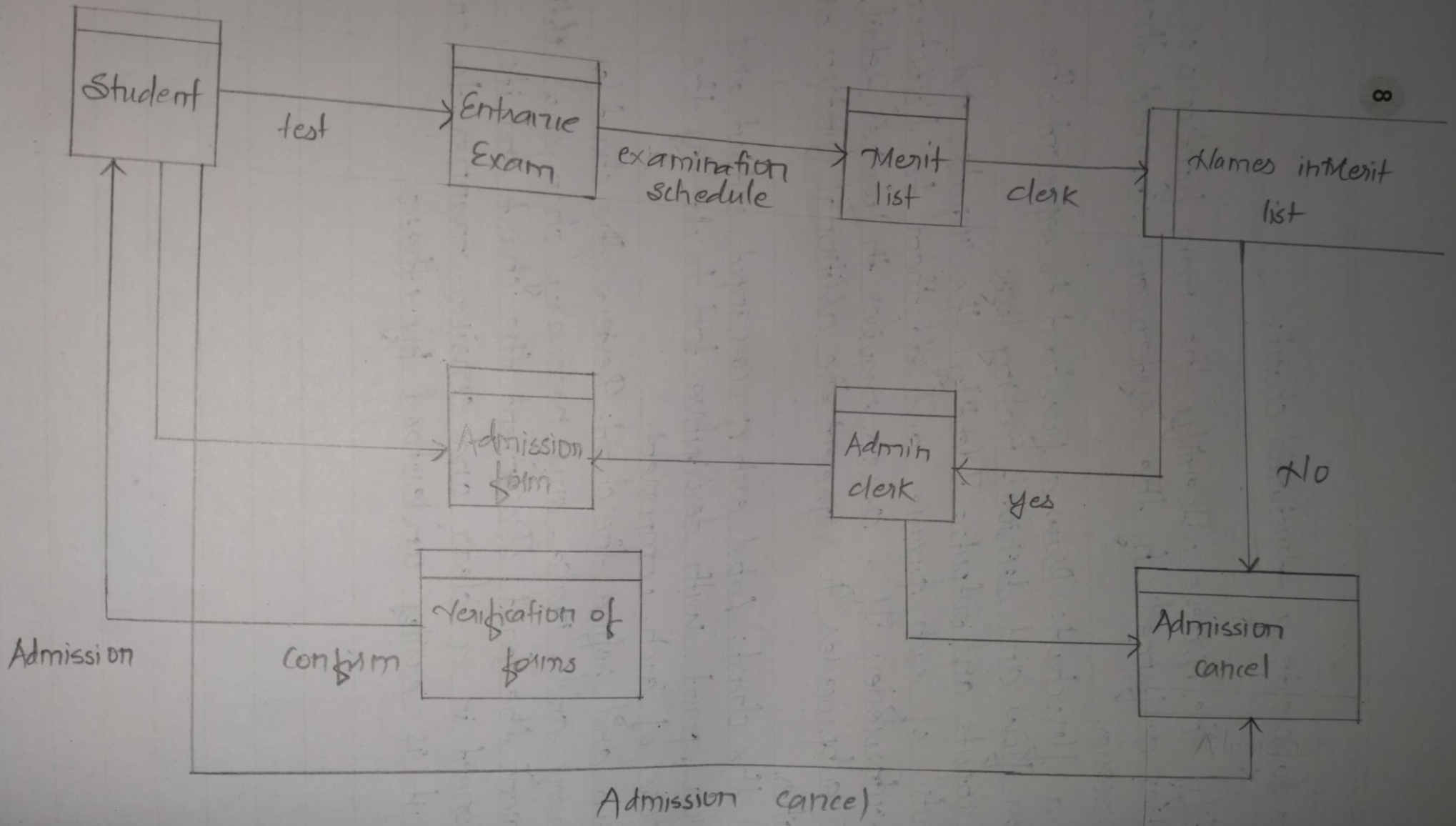
Steps to create a Data flow diagram (DFD):

- Identify the system boundary: Define the scope of the system and identify the process, entities, and data stores that are within the boundary.
- Identify the processes: Identify the main functions or activities performed by system and represent them as processes.
- Identify the entities: Identify the sources and destinations of data in the system and represent them as entities.
- Identify the data stores: Identify the data that is stored in the system and represent them as data stores
- Draw the context diagram: Draw a high-level diagram that represents the relationships between processes, entities and data stores.
- Decompose processes: Decompose the processes identified in step 2 into subprocesses and repeat the processes until each process is detailed enough.
- Draw the DFD: Draw a diagram for each process identified in step 6 and represent the data flows, data stores and external entities that interact with the processes.


Student
Enquiry
Admin Staff
Form verification
Seat availability
Admission In-charge
No
verification
Clerk
Admission confirmed

Steps to create a structured chart:

- Identify the processes: Identify the main functions or activities performed by the system and represent them as processes.
- Draw a flowchart: Draw a flowchart to represent the process flow and decision-making logic
- Add inputs and outputs: Identify the inputs and outputs of each processes and add them to flowchart
- Add subprocesses: If a process requires further detail, add subprocesses to represent the additional funtion or activities.
- Label the chart: Label each process, input, output and decision point with descriptive text to clarify the purpose of each component
- Review and refine the chart: Review the chart for accuracy and make any necessary changes or additions
- Document the chart: Document the chart and its component in a clear and consice manner to ensure that it is easily understood by others.


Student
test
Entrance Exam
examination schedule
Merit list
clerk
Names in Merit list
Admission form
Admin clerk
yes
No
Verification of forms
confirm
Admission
Admission cancel
Admission cancel

PRACTICAL 3

AIM: Draw the structural view diagram for the system: Class diagram, object diagram

Solution:

Example: Let's say we're creating a class diagram and object diagram for a simple library management system.

Steps to create a class diagram:

1. Identify classes: Identify the different types of objects or entities in the system and represent them as classes. In this case, the classes could be "Book", "Member" and "Loan"

2. Define class attributes: Define the attributes or characteristics of each class. For example, the "Book" class could have attributes such as "title", "author", and "ISBN"

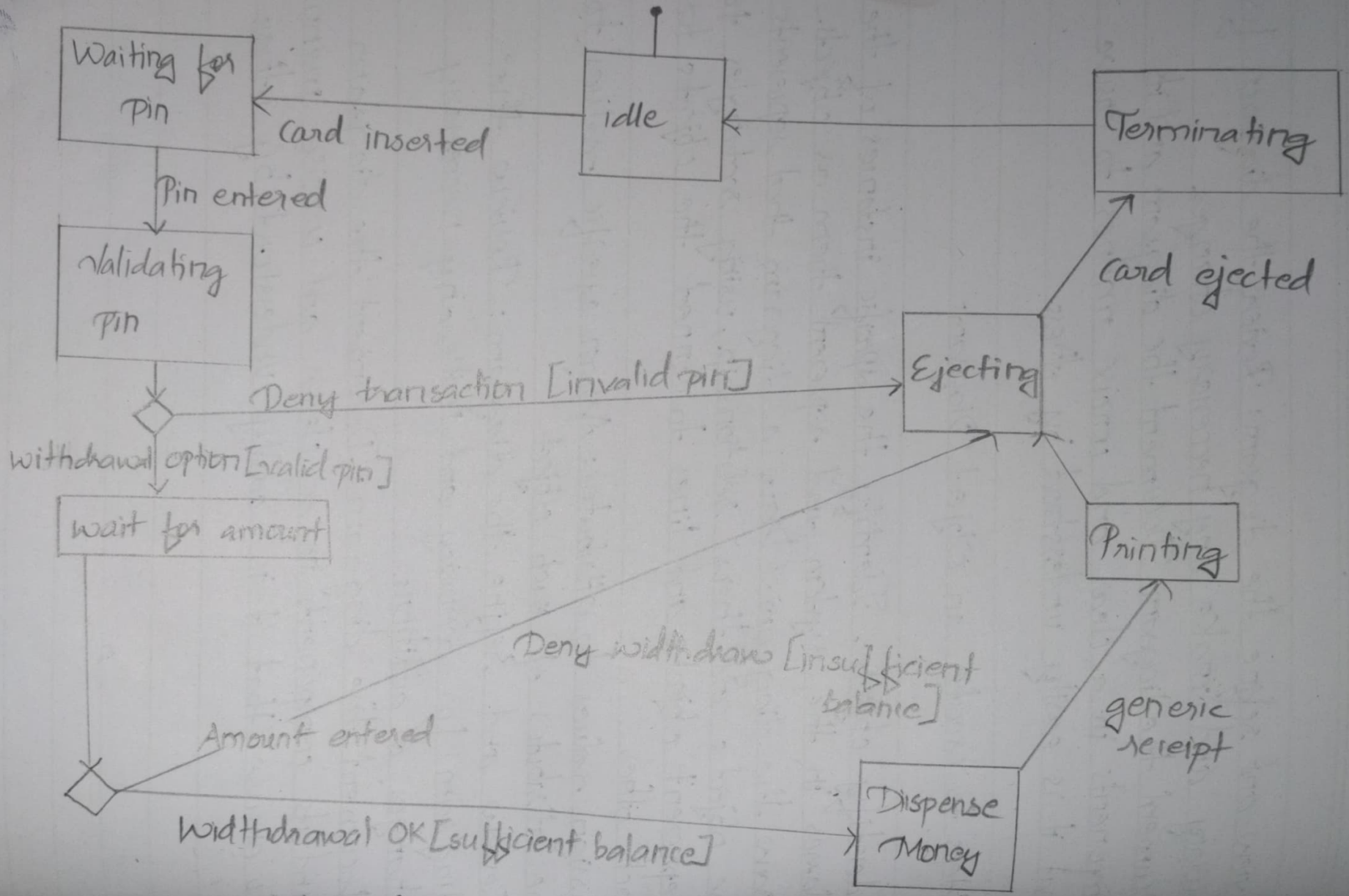
3. Define class methods: Define the methods or operations that can be performed on each class. For example, the "Book" class could have methods such as "checkAvailability" and "placeOnHold".

4. Draw the diagram: Draw a diagram that represents the classes and their attributes and methods, using rectangles to represent classes and lines to connect the classes to their attributes and methods.

5. Review and refine the diagram: Review the diagram for accuracy and make any neccesary changes.
6. Document the diagram: Document the diagram and its components in a clear and concise manner to ensure that it is easily understood by others.

Steps to create an Object Diagram:

1) Identify objects: Identify the specific instances of the clases in the system and represent them as objects.
2) Draw the diagram: Draw a diagram that represents the objects and their relationship, using rectangles to represent objects and lines to connect the objects to each other.
3) Assign values to attributes: Assign specific values to the attributes of each object
4) Review and Refine the diagram: Review the diagram for accuracy and make any neccesary changes or additions.
5) Document the diagram: Document the diagram and its components in a clear and concise manner to ensure that it is easily understood by others.


Waiting for Pin
idle
Terminating
Card inserted
Pin entered
Validating Pin
Card ejected
Ejecting
Deny transaction [invalid pin]
withdrawal option [valid pin]
wait for amount
Printing
Deny withdraw [insufficient balance]
Amount entered
generic receipt
Withdrawal OK [sufficient balance]
Dispense Money

PRACTICAL 4

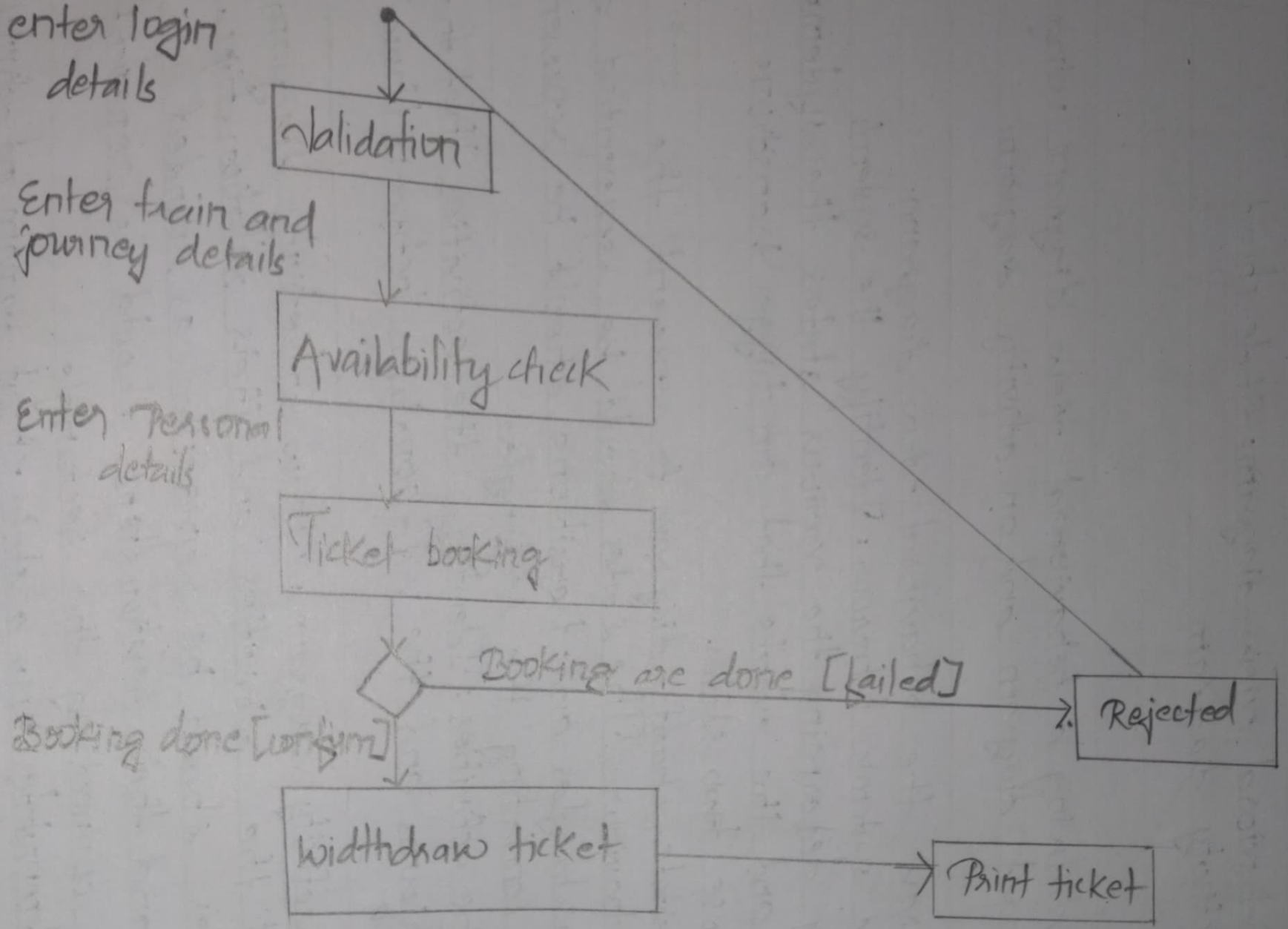
AIM: Draw the behavioral view diagram: State-chart diagram, Activity diagram.

Solution:
example of drawing a behavioural view diagram using a state-chart diagram and an activity diagram:

Steps to draw the behavioral view diagram:

1) Identify the system's behaviour: Identify the system's behavior by determing the various states the systems can be in and the events that can triggers transitions between those two states.
2) Draw the state-chart diagram to represent the system's behaviour. The states should be represented as rounded rectangles and transitions should be represented as arrows connecting the rectangles
3) Identify the activities: Identify the activities that the system performs and the relationships between those activities
4) Draw the activity diagram: Draw an activity diagram to represent the activities and their relationships. The activities should be represented as rounded rectangles and their relationships should be represented as arrows connecting the rectangle

The example demonstrates how a state-chart diagram and activity diagram can be used to represent the behavioural view of system. To ensure that the system is designed to behave as expected.


enter login details
validation
Enter train and journey details
Availability check
Enter Personal details
Ticket booking
Booking are done [failed]
Rejected
Booking done [confirm]
withdraw ticket
Print ticket

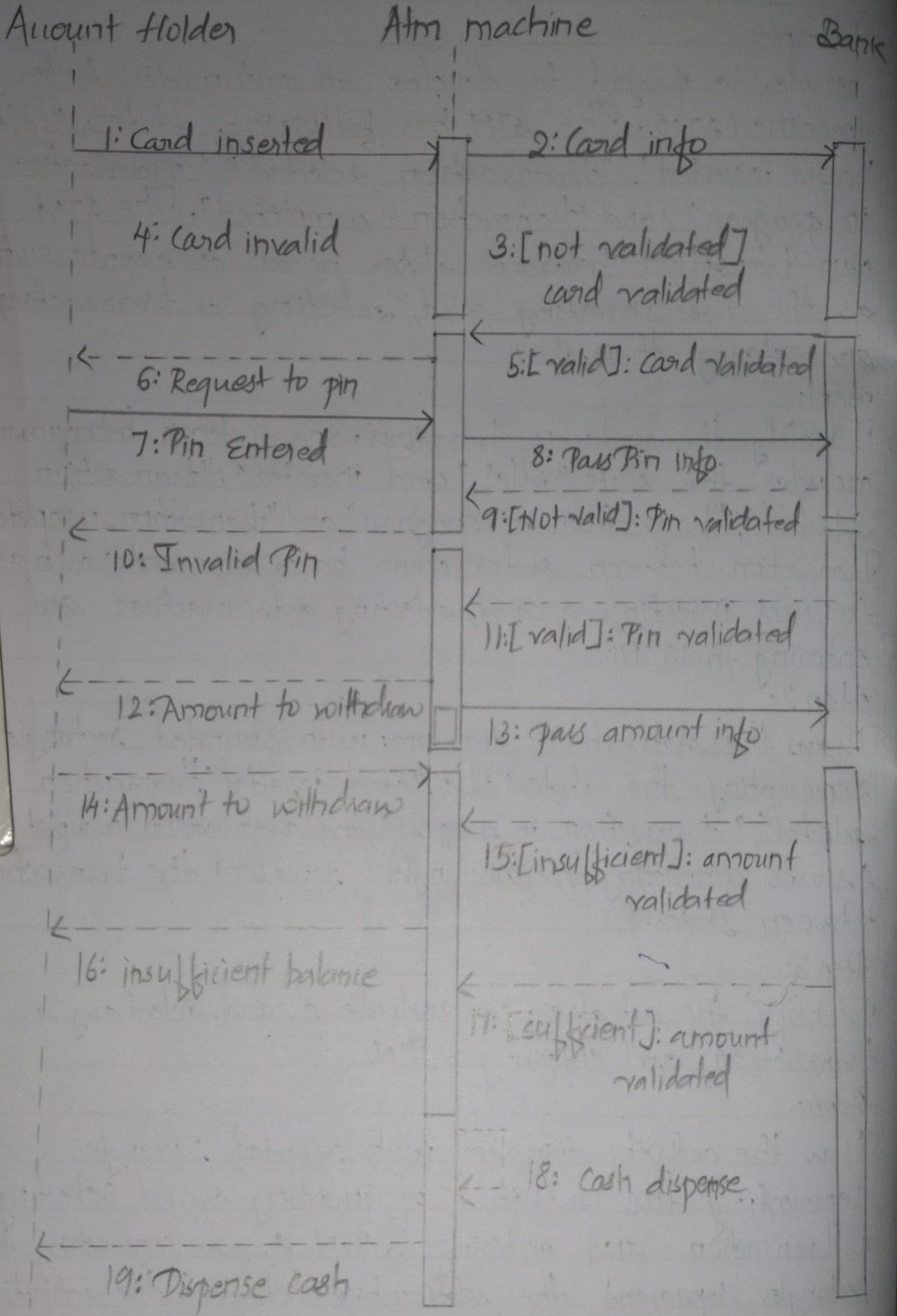
Example:
Consider a project to develop an automatic teller machine (ATM). The ATM has following states: "idle", "card inserted", "transaction selected", "transaction in progress" and "transaction completed". The ATM can transition between states based on events such as the user inserting card, selecting a transaction, and entering their PIN.

Step1:
Identify the system's behaviour: The system's behaviour includes the state "idle", "card inserted", "transaction selected", "transaction in progress" and "transaction completed". Transaction between states occur based on events such as the user inserting a card, selecting a transaction and entering their PIN.

Step 2:
Draw the state-chart diagram: with rounded rectangles representing the states "idle", "card inserted", "transaction selected", "transaction in progress" and "transaction completed". Arrows connecting the rectangles represent the transaction between states.

Step 3:
Identify the activities; can include a card, selecting a transaction, and entering a PIN.

Step4:
Draw the activity diagram with rounded rectangles representing the activities of inserting a card, selecting a transaction and entering a PIN. Arrow connecting the rectangles represent the relationships between the activities.


Account Holder
Atm machine
Bank
1: Card inserted
2: Card info
4: Card invalid
3: [not validated] card validated
5: [valid]: card validated
6: Request to pin
7: Pin Entered
8: Pass Pin info
9: [Not valid]: Pin validated
10: Invalid Pin
11: [valid]: Pin validated
12: Amount to withdraw
13: pass amount info
14: Amount to withdraw
15: [insufficient]: amount validated
16: insufficient balance
17: [sufficient]: amount validated
18: Cash dispense
19: Dispense cash

PRACTICAL 5

Draw the behavioural view diagram for suggested system: Sequence diagram, collaboration diagram.

Solution:

example of drawing a behavioural view diagram using a sequence diagram and a collaboration diagram.

Steps to draw behavioral view diagram

1) Identify the system's behaviour: Identify the system's behavior by determining the various objects that interact with each other and message that are exchanged between them
2) Draw the sequence diagram to represent the behavior of objects and messages that are exchanged between them. The objects should be represented as rectangles and the messages should be represented as arrows connecting rectangle.
3) Identify the relationship between objects; such as inheritance and association.
4) Draw a collaboration diagram to represent the relationship between objects. The objects should be represented as rectangles and their relationships should be represented as

Example:

Consider a project to develope a banking system. The system includes a customer object, a bank object, and an account object by The Sendi customer object can interact with the bank object by sending messages such as "open account" and "deposit money". The bank object can interact with the account object by sending messages such as "update balance"

Step1:
Identify the system's behaviour. The systems behaviour includes the objects customer, bank and account, and messages that are exchanged between them

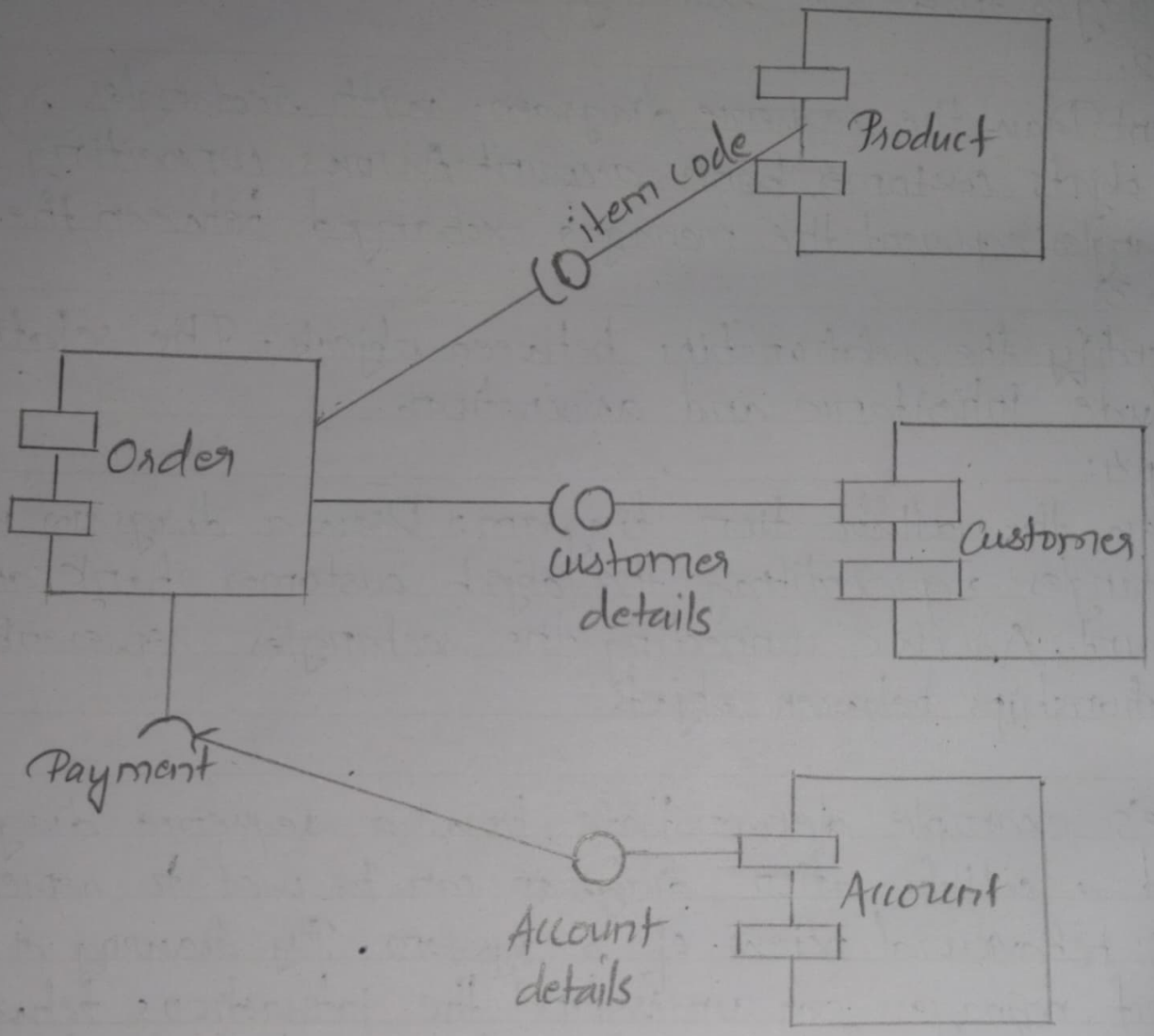
Step2:
Ident Draw the sequence diagram; with rectangles representing the objects customer, bank, account. Arrows connecting the rectangles represent the menages exchanged between the objects.

Step 3:
Identify the relationships between objects. The relationships include inheritance and association.

Step 4:
Draw the collaboration diagram: Draw a diagram with rectangles representing the object customer, bank and account. Arrows connecting the rectangles representing relationships between objects

The's example demonstrate how a sequence diagram and a collaboration diagram can be used to represent the behavioural view of a system. By drawing diagrams, project managers can understand the interactions between objects and ensure that system is designed to interact as expected.

Sample diagram:



PRACTICAL 6

Aim: Draw the implementation and envisonmental view diagram: Component diagram, deployment diagram

Solution

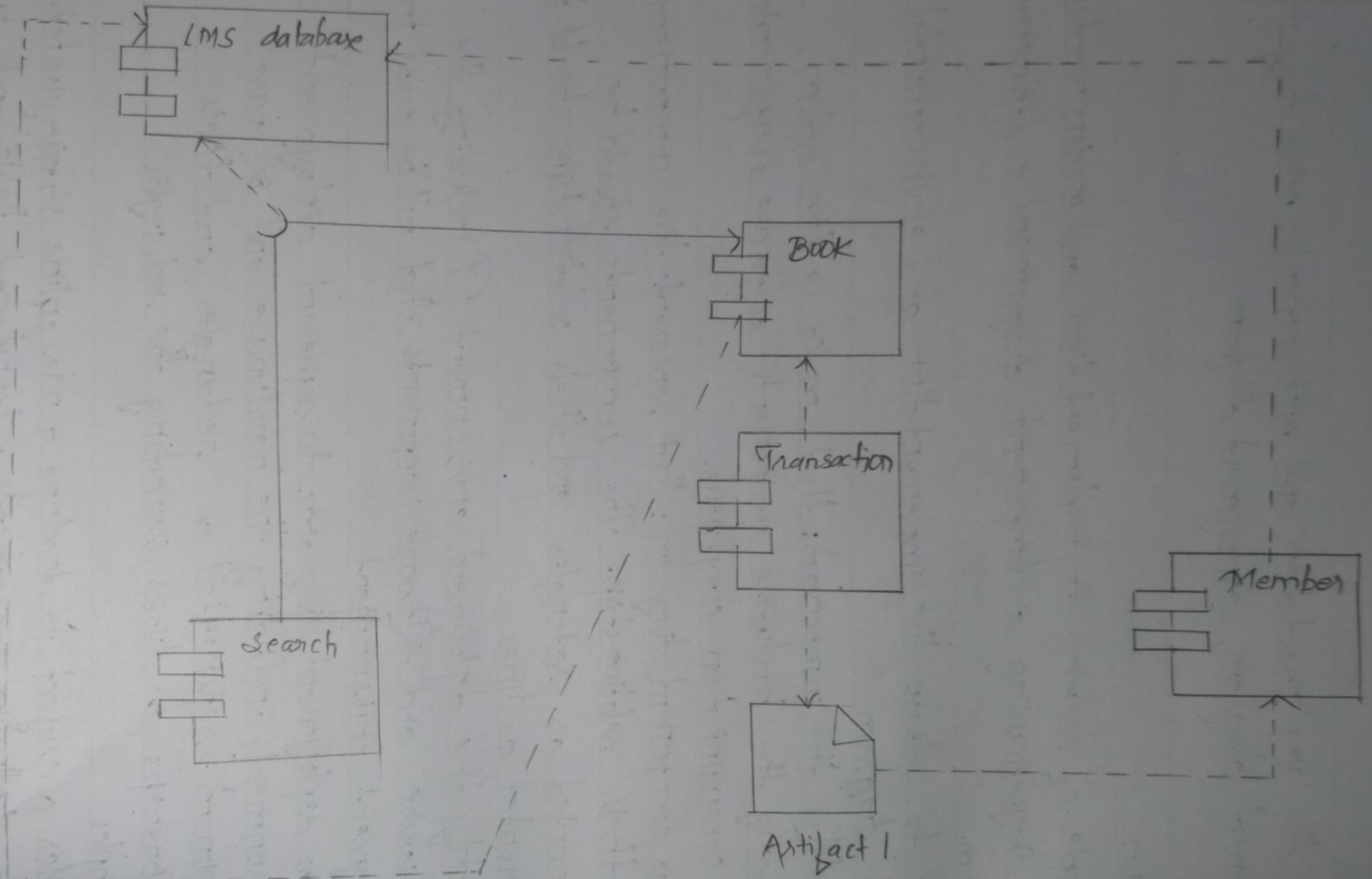
Example of drawing an implementation and envisonmental view diagram using a component diagram and a deployment diagram.

Steps to draw the implementation and envisonmental view diagram

1) Identify the components: that make up the system, including the hardware components, software components, and communication components.
2) Draw component diagram: that represent the components and their relationship. The components should be represented as rectangles and their relationships should be represented as lines.
3) Identify the deployment envisonment: Including the hardware and software components that will be used in deployment envisonment
4) Draw deployment diagram: to represent deployment envisonment, including the hardware and software components: represented as rectangles and their relationships as lines connecting the rectangles.

Example:

Consider a project to develope a new online shopping website. Having following components: front-end, back-end, database and payment gateway. The deployment envisonment


LMS database
Book
Transaction
Search
Member
Artifact 1

will be a server running Linux and a web server.

Step 1:

Identify the components: that are front-end, back-end, database and payment gateway.

Step 2:

Draw the component diagram: with rectangles representing the front-end, back-end, database, payment gateway components, lines connecting the rectangles representing the relationships between components.

Step 3:

Identify the deployment environment: that is a server running Linux and a web server.

Step 4:

Draw the deployment diagram: with rectangles representing the server running Linux and web server and lines connecting rectangles representing relationship between the server running Linux and web server

This example demonstrates how component and deployment diagram can be used to represent the implementation and environmental view of system. By drawing these diagram, project managers can ensure that system is designed correctly, and that components are correctly integrated and configured in deployment environment.

Task name	Duration	Planned start	Planned Finish	Assigned	Percent	Planned Hour	Money
Req. gathering	2 days	2/3/23	3/3/23	Sam	20%	20 Hours	
Coding	23 days	2/3/23	3/4/23	Mike	90%	200 Hours	13,500
Testing	5 days	6/4/23	12/4/23	Joan	60%		
Deployment	4 days	17/4/23	20/4/23	Jennifer	70%	25 hours	2000

PRACTICAL 7

Aim: Prepare time line chart / Gantt Chart / PERT chart

Solution:

example of preparing a Gantt Chart for a software dep development project

Consider a software development project with following tasks:

1) Requirement gathering and analysis
2) Design and prototyping
3) Implementing
4) Testing and debugging
5) Deployment and maintenance

Steps to prepare a Gantt chart:

1) Determine the duration of each task: This can be done by considering factors such as the complexity of task, the number of resources involved, and available time.
2) Determine the dependencies between tasks: Identify any tasks that must be completed before another task can begin. Example: the implementation task cannot begin until design and prototyping task is completed
3) Create a chart with the task duration: Create a chart with tasks listed vertically and the duration of each task listed horizontally. Each task should be represented by a horizontal bar, with the length of bar representing the duration of task.

4) Add the dependencies: Show dependencies between tasks by connecting the corresponding horizontal bars with arrows.
5) Add a timeline: to the chart, with each day or week represented by a vertical line.

This Gantt provides a visual representation of project timeline, including the duration of each task, dependencies between tasks, & overall project progress. The chart can be updated as the project progresses, providing a useful tool for monitoring and managing project timelines.

Example gantt chart

Task	Duration	Dependency
Requirement gathering and analysis	10 days	None
Design and prototyping	14 days	Requirement gathering and analysis
implementation	20 days	Design and prototyping
Testing and debugging	10 days	Implementation
Deployment and maintenance	5 days	Testing and debugging

PRACTICAL 8

Aim: Write down the problem statement for a suggested system of relevance.

Solution:

Problem statement:
The current method for searching for solutions to specific problems is time-consuming and often provides irrelevant results. The user often has to sort through multiple sources and pages of information to find practical solutions.

Proposed system:
A platform that provides step-by-step practical solutions to problem in a specific, organized and easy to understand format. The platform should use machine learning techniques to match the user's query to the most relevant solution and provide additional resources for further learning. The goal of system is to simply the process of finding practical solutions and reduce the time, effort required to find relevant information.

PRACTICAL 9

AIM: Perform requirement analysis and develop Software Requirement Specification Sheet (SRS) for suggested system.

Solution:

Software Requirement Specification (SRS)

Introduction:

1.1 Purpose: The purpose of this document is to describe the requirements for suggested system of relevance "Step by step Practical Solutions".

1.2 Scope: The system will provide step-by-step practical solutions to problems in specific, organized and easy-to understand format. Platform will use machine learning techniques to match users query to most relevant solution

1.3 Definations, Acronyms and Abbreviations: Not applicable

1.4 References: Not appliable

Overall Desription:

2.1 Productio prespective: The system will be a web-based platform accessible to users through a browsers.

2.2 Product funtions: The system will provide step-by-step solution to problems in various categories such as computer software, hardware and other problems. The platform will provide additional resources for further learning

2.3 User characteristics: The target users of the system are individuals seeking practical solutions to specific problems

The users should be having basic computer skills and knowledge to use internet.
2.4 Constraints: The system should be designed to be accessible on multiple devices and browsers.

Specific requirement:
3.1 Functional Requirements:
3.1.1 Users authentication: The system should allow users to sign up and log in to acess solutions.
3.1.2. Search funtionality: The system should provide a search bar to search for solutions based on keywords or specific problem statement
3.1.3 Solution categorization: The solutions should be categorized into different categories to make it easies for user to find relevant information.
3.1.4 Machine learning algorithm: The system should use machine learning algorithm to match the query to most relevant solution.
3.1.5 Step-by-Step Solution: The system should use provide detailed step-by-step solutions to problems in an organized and easy-to-understand format.
3.1.6 Additional resources: The system should provide additional resources such as videos, articles and links for further learning.
3.2 Non-Funtional Requirements:
3.2.1 Usability: The system should be user friendly and easy to use.
3.2.2 Performance: The system should be designed to handle high traffic and provide fast search results.

3.2.3 Security: The system should ensure the security of user data and protect against unauthorized access
3.2.4 Scalability: The system should be scalable to accommodate future growth.

Supporting Information

4.1 Assumptions and Dependencies : Not applicable
4.2 Appendices: Not applicable

This document serves as the initial Software Requirement Specification (SRS) for the "Step by Step Practical solutions" platform. The requirement described in this document may change as the project progresses and the final version will be updated to reflect to any changes.