Subject / Theme: Mitu Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

Name : Afsana Akter Mitu
ID : IT-21053

Answer to the Q^. NO-01

Scrum-Based E-commerce Project :

A. Product backlog (Breaking user stories into task)

User story 1: "As a usedr I want to log in securely so that I can access my account"

Task:
i) Design the log in page.
ii) Set up the database to store dataset
iii) Develop the backend for the login system
iv) Enerypt passwand
v) Valid inputs (check if username and passwand are correct.
vi) Add a forget passwand option.
vii) tests all features and fix bugs.

User story 2: As a user, I want to search for products by category to find

items easily"

Tasks:

i) Design the searchbar

ii) Create a database to store product information.

iii) Develop the backend for the search feature.

iv) Implement catagory based filtering

v) Optimize search for large database.

vi) Test the ~~f~~feature.

vii) Conduct user testing for the feature

B. Sprint Planning (setting priorities)

- Login Feature (user story 1):

→ Highest priority because users need secure access to their accounts.

- Search Feature (user story-2):

→ second priority because user can manually browse products if the search feature is not ready.

C. Scrum Board (for Task Tracking):

To Do	In Progress	Done
Design login page	Develop backend API	Database setup completed
Encrypted password	validate inputs	
Design search bar		

D. How the Scrum board works:

- To Do : Task that haven't started yet.
- In Progress: Task that are currently being worked on.
- Done : Tasks that are completed.

Task move step by step from "To Do" → "In Progress" → "Done"

The scrum master monitors Progress daily and helps solve any issues.

Answer to the Q: NO - 02

For a project with high risks and changing requirements choosing the right methodology

is important to manage risks and adopt to change effectively. Lets look at how each methodology helps.

1) Spiral methodology :

→ Risk management : The spiral model focuses on managing risk by constantly reviewing them throughout the project. It allows for quick identification of risks and fixked them early, with testing and prototyping in each phase.

→ Adaptability : The spiral model is flexible and iterative, allowing changes based on new informations or requirements. This is helpful when the project is complex or has many unknowns.

2) Agile methodology :

→ Risk management : Agile works in small steps called sprints, where the team checks and deals with risk regularly. This means risk can be addressed early and issues are found quickly.

→ Adaptability : Agile design to adaptable. The team can easily adjust to changes in client needs, as if involves constant feedback and changes after each sprint.

3) Extreme Programming (XP) :

→ Risk Management : XP uses continous testing and regular release which helps identify problems immediately. The team also work in pairs. which helps reduce errrors and risks in code.

→ Adaptability : XP is highly adaptable with short cycle and constant client feedback, so, it can quickly respond to changing needs.

* For this Project, Agile is the best choice, why Agile?

→ Risk Management : Agile's small, regular updates helps the team manage risk more easily.

→ Adaptability : Agiles flexibility

means the product can evolve as the clients needs changes

→ cost effectiveness : Agile allows the team to focous on the most important featones first saving time and money.

Answer to the Q: NO - 03

To choose the best development methodology for each project, we need to consider the characteristics of both Projects.

Project A : well defined requirement and a strickt deadline

Project B : Evolving requirements, uncertain timeline and continuous

customer feedback.

Lets compare the methodologies based on these characteristics.

1. Waterfall model :

→ Predictability : The waterfall model is linear and sequential. It works project will well defined requirements.

→ Customer collaborations : Minimal collaboration after the initial requirements phase. customer feedback is given only after the final product is delivered.

→ Risk management : It is less flexible to changes. If risks arise later, it becomes difficult to address them because changes require going back to earlier stages.

2. Agile model :

→ Predictability : Agile is iterative.

Delivering small pices of the product regularly. It is less predictable but more adaptable to changes.

→ Customer Collaboration : continuous customer involvement and feedback is key in Agile. The product evolves based on customer needs throughout the process

→ Risk Management : Agile helps in managing risk by working in small sprints. Risk are identified early and addressed immediately making it high flexible.

3. Extreme Programming (XP) :

→ Predictability : Similar to Agile. XP focuses on short cycles and continuous delivary which can make it less pre.

→ customer collaboration : XP emphasiz close collaboration with customes.

continuous feedback and regular adjustment to the product are center of XP.

→ Risk Mangement : XP uses practices like pair programming, continuous testing and frequent release to manage risks effectively and improve quality.

4. Spiral model :

→ Predictability : The spiral model combines iterative development with risk management.

→ Customer collaboration : It involves customer in regular reviews and the product evolves with their feedback.

→ Risk Management : The spiral models keys strength its focus on risk manage at every stage.

* Which methodology is best for each proje

For Project A :

Best : Waterfall or spiral :

→ Waterfall works well for clear requirements and strict deadline. the process of spinal model would be better as it includes risk management and allows adjustments along the way.

For Project B:

Best : Agile or XP:

→ since the requirement are evolving and there's continuous customer feedback, Agile would be the most suitable methodology. XP could also be considered as technical excellence frequent testing.

Answer to the Q:NO-04

Principle of Software Engineering Ethics:

1. Public Interest: Always prioritize the well-being of society, users

and the public over personal or company's benefits.

2. Quality of works: Deliver high quality software that meets user needs, work efficiency and avoid harm.

3. Honesty: Be truthful about what software can and can't do.

4. Privacy and Confidentiality: Protect user data and ensure confidentiality. Do not misuse information.

5. Competence: only take up work you are qualified for and continuously improve your skills to maintain professional standards.

Professional Responsibilty Issues:

1: Accountability: Software engineer are responsible for the impact of their works.

2. Ethical Dilemmas: Sometimes engineers face tough choice, like balancing business goals with user safety.

Subject / Theme: Mita Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

Role of Acm/IEEE code of Ethies:

The ACM/IEEE code of ethies provides guidelines to help software engineers-

1. Act in public interest: Always prioritize safety, fairness and social good in software development.
2. Be honest and trustworthy: Avoid misleading stakeholders on users about software capabilities or limitations.
3. Respect Privacy: Protect user data and ensure its used responsibly.
4. Continuous learning: Stay updated with the latest knowledge and best practice
5. Work with integrity: Avoid bias, unfair treatment or harm in software processes.

BORNO

Subject / Theme: Mitu Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

Answer to the Q: NO-05

Functional Requirement:

1. Flight Booking: User must be able to search for flight, select one and book ticket.
2. User Registration and login: The system should allow user to create accounts and login security.
3. Payment processing: The system should support multiple payment method like credit cards, b debit cards and digital wallets.
4. Flight Reschedulingo or cancellation: User must be able to reschedule or cancel booking.
5. Real time updates: The system should provide real-time flight status update

Non Functional Requirements:

1. Performance: The system must handle up to 10,000 users concurrently without slowing down.

BORNO

2. Usabilities : The system should have a simple and intuitive interface to easy navigation.

3. Security : All transactions and used data must be encrypted

4. Scalability The system should support additional features or handle increased user traffic in the future

5. Availability : The system should be operational 99.9% of the time to minimize down time

Answer to the Q:NO-06

Defination : The v-model (varification & validation model) is a software development model that emphasizes the relationship between development phase and corresponding testing phase. The 'v' shape visually represents

the sequence of activities where the left side involves testing.

* Key features of the V model :

1. Development phase :

→ Requirement Analysis : Understanding what the customer wants.

test : Acceptance Testing ensures the final product meets user needs.

system Design : planning how the software will work as a whole

Architecture Design : Designing the modules, components to the software

Module Design : Detailed design of each module.

2. testing Phase :

→ Each development phase has a corresponding test phase to validat.

→ testing starts early with plans and progresses alongside development

Relationship Between Development and Testing :

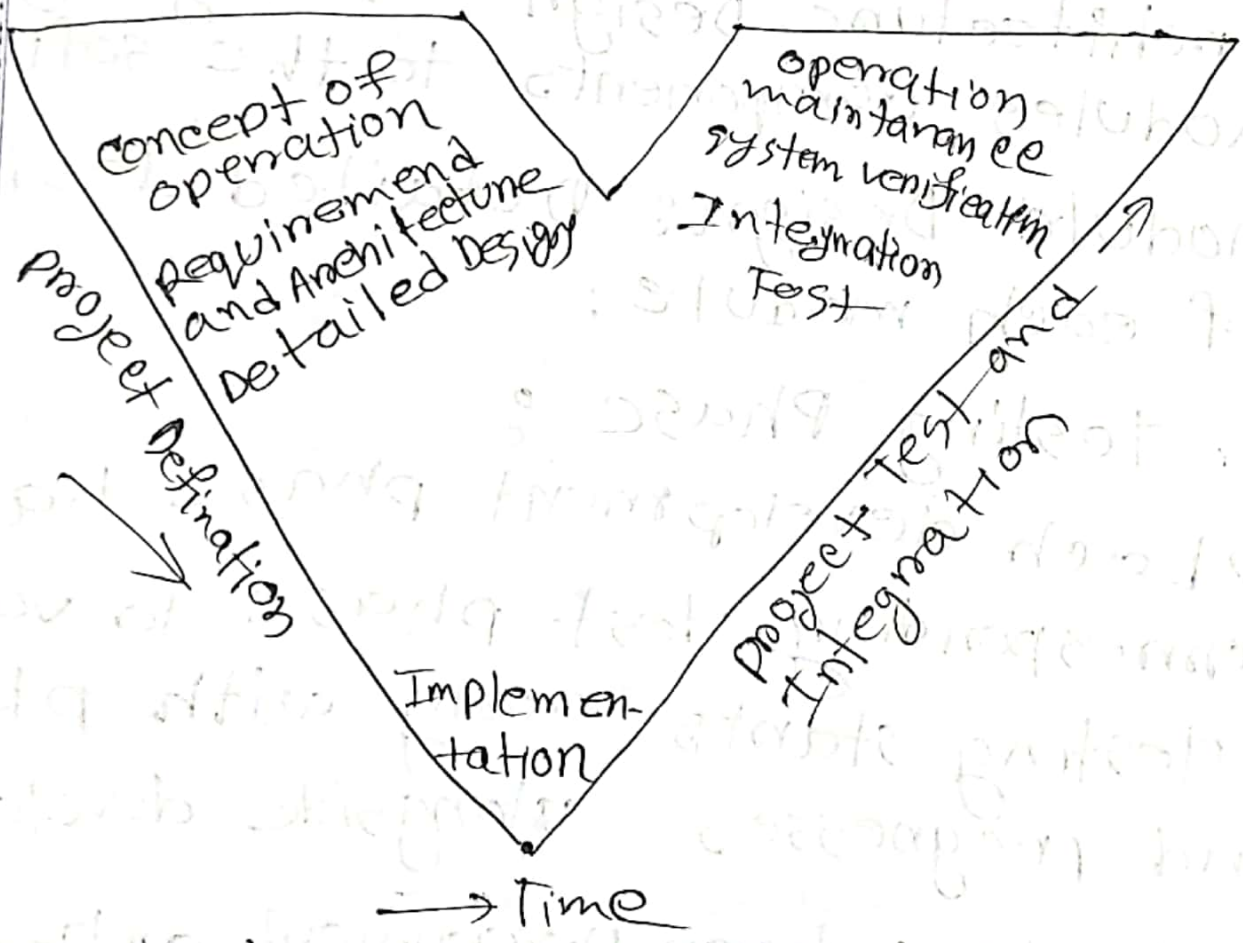
Subject / Theme: Milu Date: / / ☐Sat ☐Sun ☐Mon ☐Tue ☐Wed ☐Thu ☐Fri

1. Requirement Analysis ↔ Acceptance testing.
→ Ensure the software satisfies user needs

2. System Design ↔ system testing
→ vanifies the complete system behavior

3. Architecture Design ↔ Integration Testing
→ Checks interaction between different modules

4. module Design ↔ unit testing
→ Ensures each small part work correctly



BORNO

Subject / Theme: Milu Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

Answer to the Q: NO. 07

Process of Prototype Development in software Engineering:

Key stages of Prototyping

1. Requirement Gathering
2. Quick Design:
3. Prototype Development.
4. User Evaluation
5. Refinement and Iteration.
6. Final Implementation.

Benefits of Prototyping Model:

→ User Feedback: Ensures the final product aligns with real user needs.

→ Risk reduction: Identifies design flaws early, reducing costly rework.

→ Iterative Development: Allows continuous refinement and adaptability.

By enabling early validation and continous improvement, prototyping enhances usability.

Subject / Theme: Mita Date: / / ☐Sat ☐Sun ☐Mon ☐Tue ☐Wed ☐Thu ☐Fri

8

Process improvement cycle in Software Engineering.

Key stages of process Improvement

1. Process Assesment: Identify inefficiencies and areas for improvement
2. Process modeling and Redesign: Define optimized workflows and best practices.
3. Implementation of changes. Apply modifications to the software process.
4. Monitoring and Evaluation: measure improvements using process metrics.
5. Continous improvement: commonly used process metrics

c

and their importance.

i) Defect density.

ii) cycle time.

iii) Lead time - measures the duration from request to delivery, improving project planning

iv) code churn → Evaluates the frequency of code changes, helping detect instability in development.

9

Five levels of CMM and their contribution

1. Initial (Level 1): Unstructured and Reactive.

→ Process are ad-hoc, unpredictable and poorly controlled

→ Improvement, Establishes basic project management.

2. Repeatable (Level 2) - manage at the project level.

Subject / Theme: Milu Date: / / ☐Sat ☐Sun ☐Mon ☐Tue ☐Wed ☐Thu ☐Fri

→ Basic project management and control exist.
→ Improvement : Ensures consistency in project execution.

3. Defined (Level 3) : Organization wide process standards.
→ Standardized, documented processe and accross the organization.
→ Improvement : Enhances process consistency and scalability.

4. Managed (Level 4) : Quantitative control.

5. Optimizing (Level 5) : continous process improvement.
→ Focuses on continaous learning and process enhancement.

BORNO

10

Agile is based on Agile Menifesto, emphasiging :

1. Customer collaboration - continuous user involvement for better alignment with needs.
2. Individuals and interactions - Teams over rigid processes for flexibility.
3. Working software - Deliver functional increements frequently.
4. Responding to change - Adaptability over strict planning

Application of Different Environments:

→ Startups
→ Regulated Industries
→ Large Enterprises

Benefits of Agile methods:

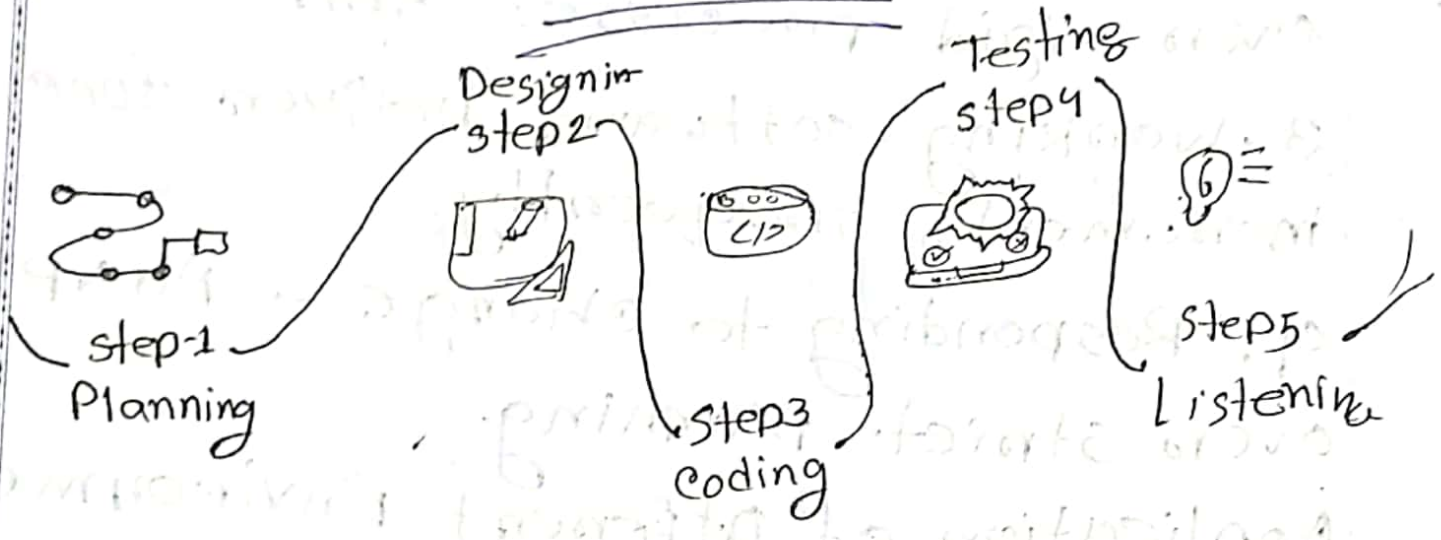
→ faster delivery and adaptability.
→ Higher customer satisfaction

→ continuous improvement and innovation.

Challenges in Agile Adaption :

→ Difficult to scale in large structured organization.

→ Requires cultural and mindset shift.



Planning :

→ the development team and customer meet to discuss and approve the products features.

→ the planning game is mostly that helps guide the project

Iterations:
→ The XP life cycle iterates several times until the product is ready.
→ The development team assigns tasks for each iteration.

Death:
→ The product reaches the end of its life cycle.
→ XP is an agile methodology that uses practices like pair programming test driven development; and refactoring.

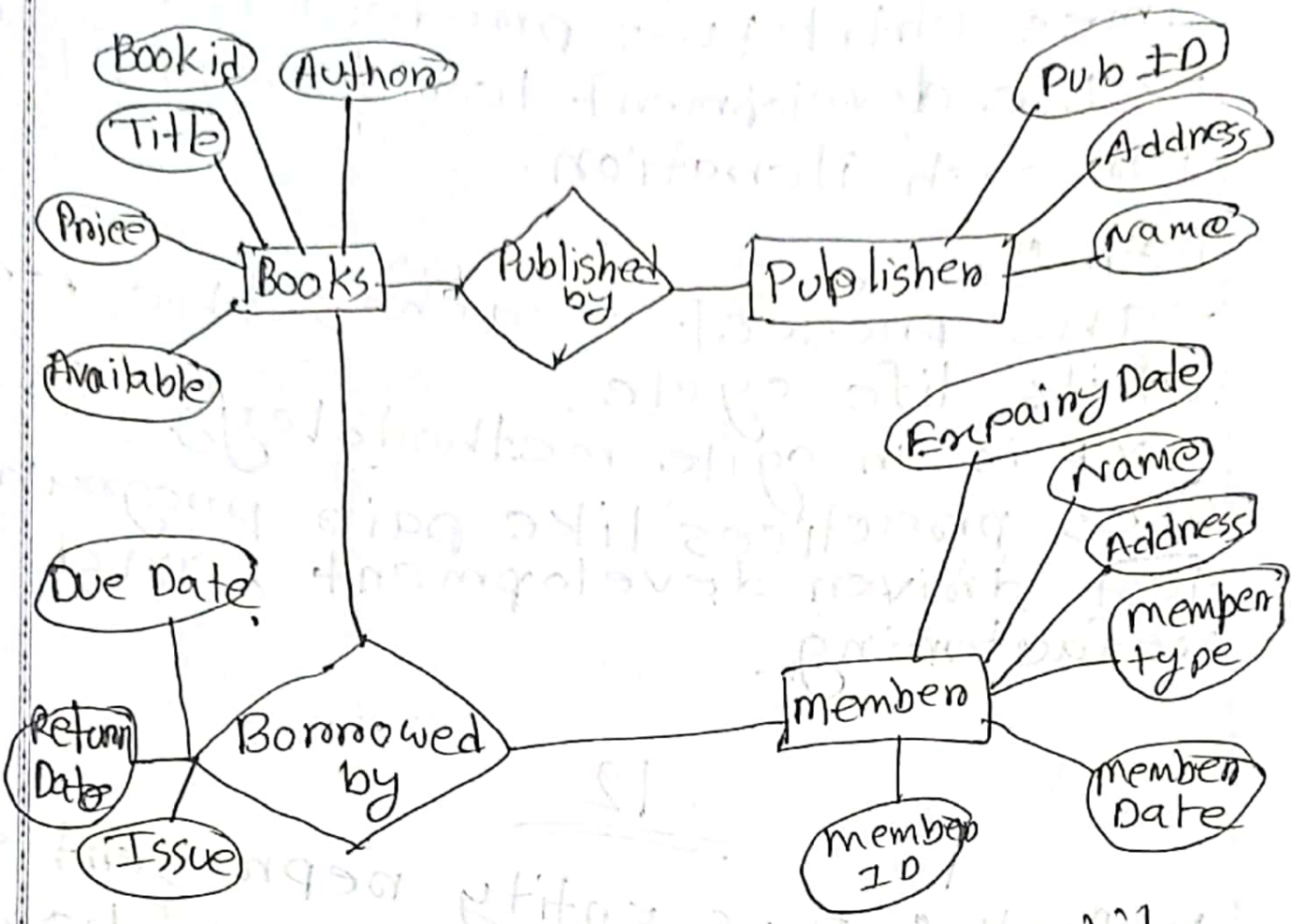
12

1. Book: this entity represents the library's collection of books. Each book has attribute like title, author, ISBN and genre
→ A book can be borrowed by multiple members

2. member: This entity represented library members who borrow books. Each member has attributes

Subject / Theme : Mitu Date : / / ☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

such as name membership, ID and contact Details.



3. Borrowing Activity: The entity tracks the borrowing details It records attributes like borrowing date, return due date and return status.

BORNO

Subject / Theme: Write Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

13

Testing is the process of Evaluating a system or component to ensure it behaves as expected and meets specified requirements.

Difference between validation and verification:

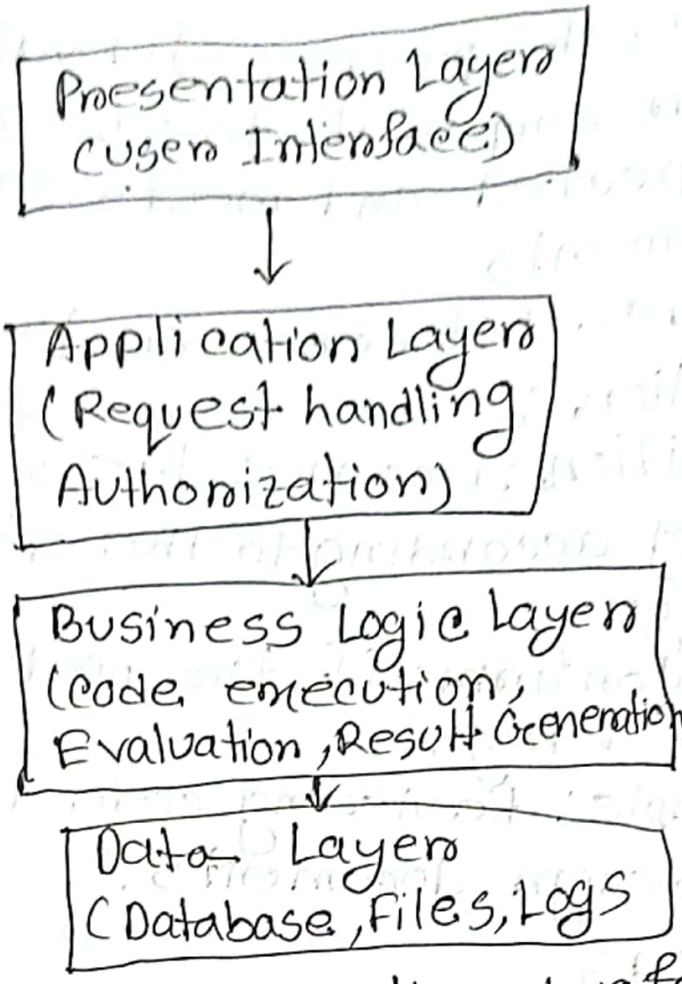
Verification:
→ Definition: Ensures the system is built correctly according to the specifications and design.
→ Focus: Confirms if the product was developed right.
→ Example: Reviewing code to ensure it meets design documents.

Validation:
→ Defination: Ensures the system meet the user's needs and requirements.
→ Focus: Confirms if the right product was built (meet user expectations)
→ Example: Testing if the application works as intended by the user.

BORNO

Mitu Date: / /
Subject / Theme:

14

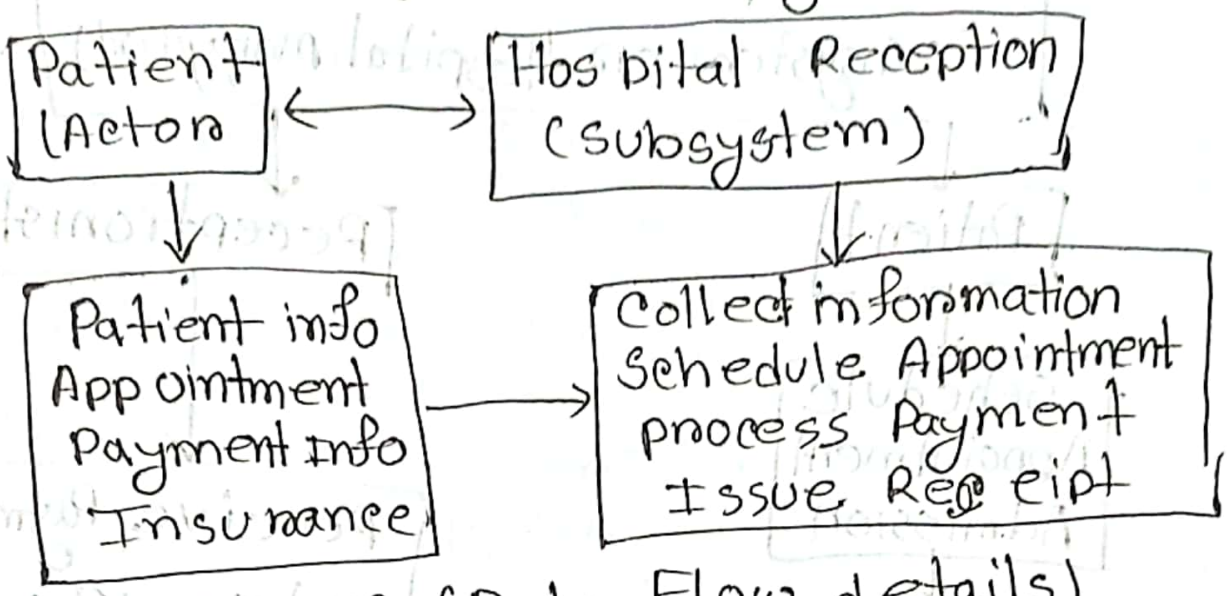


Explanation:

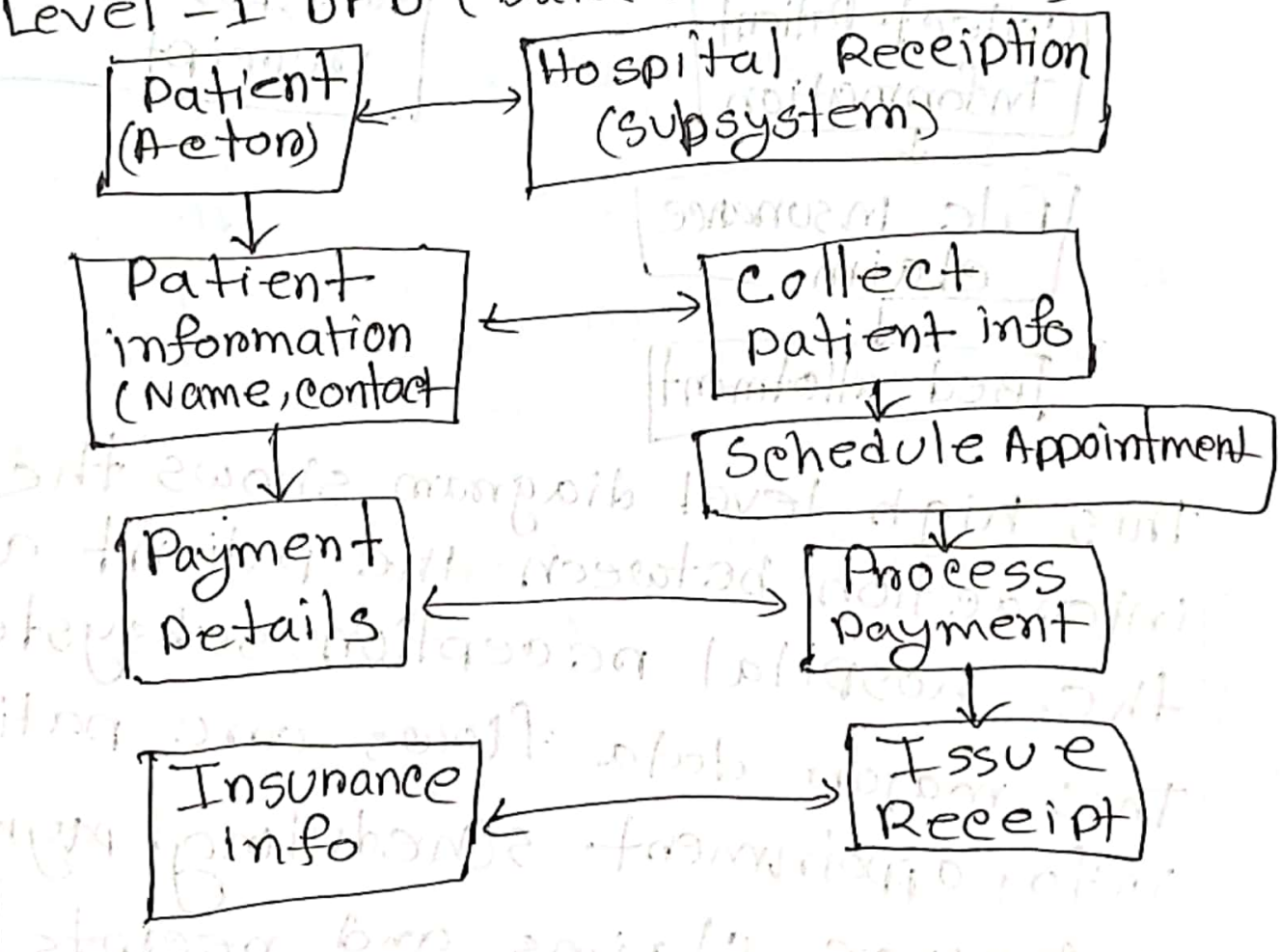
1. Presentation Layer: the interface where user interact. It collects inputs like problem sol: routing authorization.
2. Application Layer: Handles user request
3. Business Logic: Responsible for compiling executing code, cheking outputs and providing result.
4. Data Layer: stores all persistant data..

15

Level 0 DFD (Context Diagram):

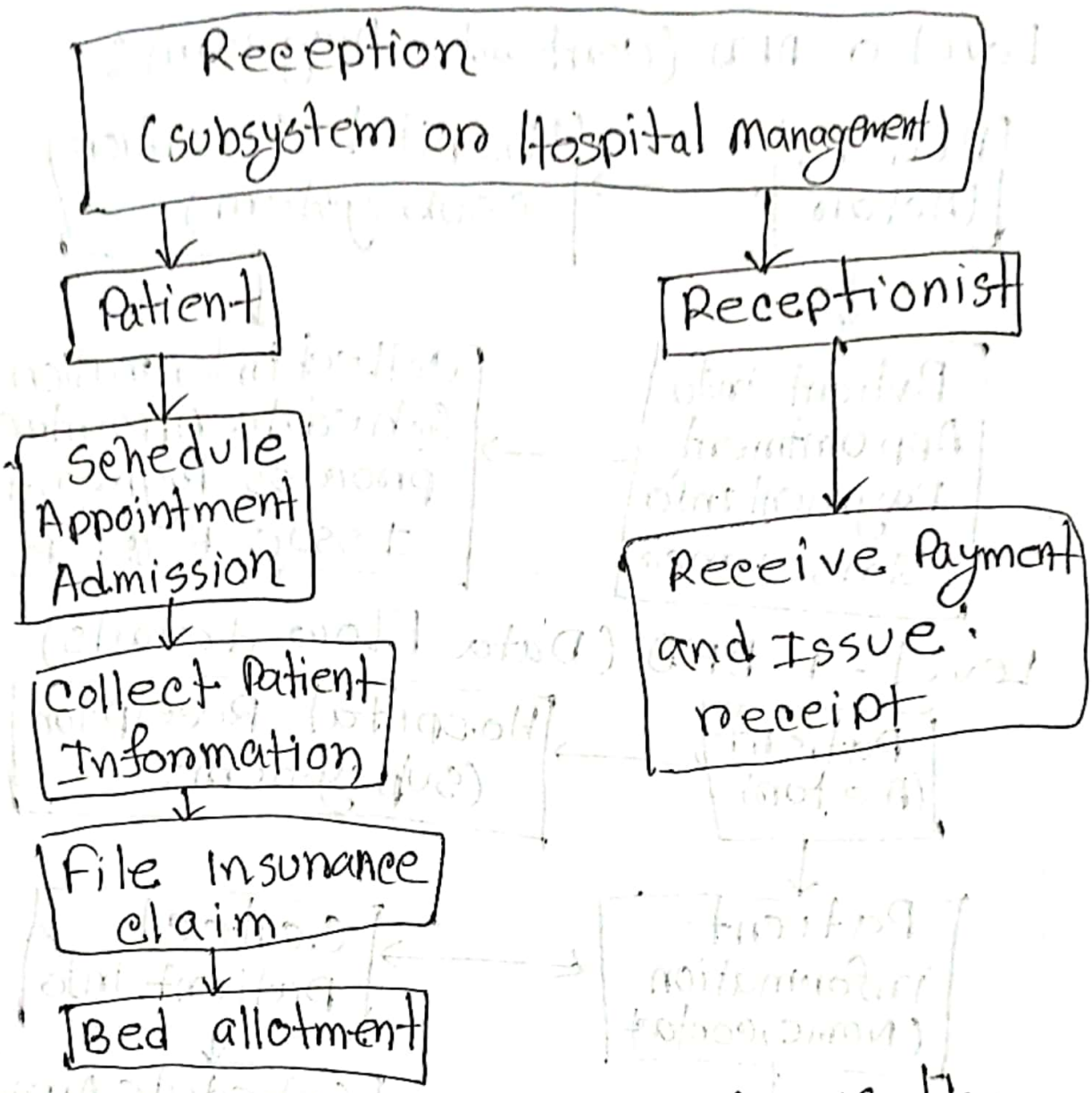


Level -1 DFD (Data Flow details)



Subject / Theme: Mila Date: / / Sat Sun Mon Tue Wed Thu Fri

UML use case Diagram for Reception



This high level diagram shows the interaction between the patient and the hospital reception subsystem. The major data flows are patient info, appointment scheduling, payment insurance claims and receipts.

BORNO

16

Based on the UML sequence Diagram you provided, we can design a UML class diagram to represent the relationship between the key classes involved in this system -

Identified classes from the sequence Diagra

1. Student : UserID, password
methods : clickloginbutton(), viewclassList()

2. Login screen :
→ methods : validateUser(UserID, Password)

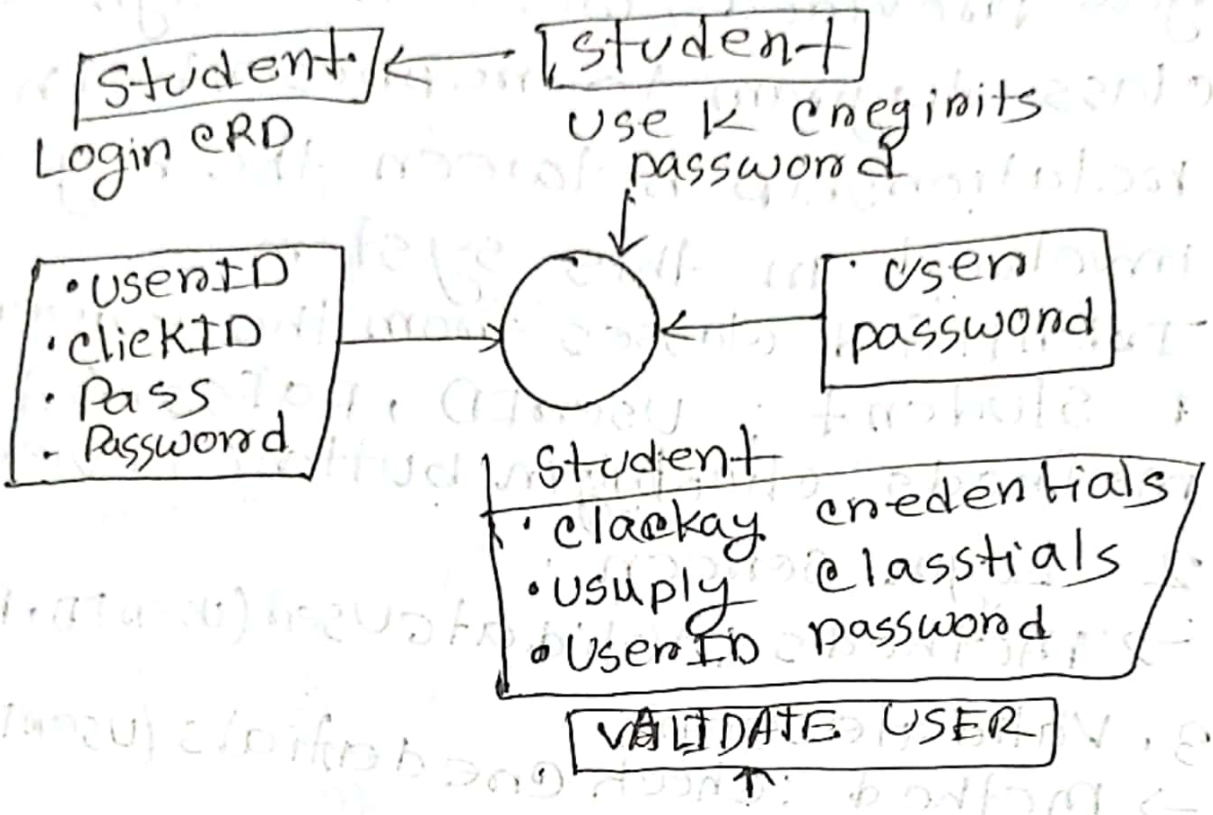
3. validateUser :
→ method : check credentials (UserID, passwor

4. Database :
Attributes : UserID, password, classList.
methods : fetch User Details(), returnclasslist()

Relationships :
Student interact with login screen
→ Login screen communicates with Validate User
→ Validate user fetches data from Database.

BORNO

→ Database stores user details and classlist. Now, let me generate this UML class diagram based on this structure



17

Quality Assurance focuses on improving and ensuring the processes involved in production, aiming to prevent defects by optimizing workflows and systems.

Subject / Theme: Mitu Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

Quality control involves inspecting and testing the final product to ensure it meets specified standards.

Differences:

→ Focus: QA is on process, QC is on product.

→ Approach: QA is proactive, QC is reactive.

→ Objective: QA aims to improve system QC aims to find and fix defects.

Impediments:

→ QA impediments: Lack of clear processes, resistance to change, insufficient training and inadequate resources.

→ QC impediments: Lack of proper testing tools time constraints, poor communication and inconsistant standards.

BORNO

Subject / Theme: MPlca Date: / /
☐Sat ☐Sun ☐Mon ☐Tue ☐Wed ☐Thu ☐Fri

18

No, the goal of QA is not just to find bugs early, but to ensure the overall quality of the process, product and outcome by preventing defects through systematic improvements.

Role of QA in SDLC:

1. Requirement phase review requirements for clearity and testability.
2. Design phase: Ensure design meets requirements and is feasible for testing.
3. Development phase: Establish coding standards and conduct code reviews
4. testing phase: Perform detailed testing
5. Deployment phase: verify deployment readiness and conduct final checks.
6. maintenance Phase: Monitor post release and ensure quality in updates and patches.

BORNO

Subject / Theme: Mitu Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

19

Rapid Action Development (RAD) is a software development model focused on quick development and itenation through user feedback, typically using prototypes.

Key Phases:

1. Requirements Planning: Gather and define high-level requirements from stakeholders.
2. User Design: Collaborative design where users interact with developers to create prototypes.
3. Construction: Iterative development of the software with continuous feedback and improvements.
4. Cutover: Finalizing the system, data migration and transitioning to live usage.

Key Principle:

→ Users involvement: Active participation

throughout development.

→ Prototyping: Frequent delivery of prototypes for user testing and feedback

→ Iteration: Rapid cycles of development and refinement.

Advantage:

→ Faster development due to parallel and iterative work.

→ High user satisfaction from direct involvement and feedback.

20

Here a Java version of the JUnit test class based on the login in the given python code.

The Java code follows the save principles ~~test~~ the decisions in my provided code.

Decision Table:

Decision	x input	y input
y == 0	5	0
x == 0	0	3
i) y == 0 (loop)	4	2
ii) y != 0 (loop)	4	3
Edge case : y negative	1	-2
Edge cas : x negative	5	2
	-3	

Junit test class in Java :

```
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import java.util.ArrayList;
import java.util.list;
public class DecisionTest {
private List<String> output;
public void setup() {
output = new ArrayList<>(); { }
private void println(String message){
  output.add(message);
  }
  private void process(int x, int y){
    if (y==0){
      println("y is zero"); }
```

```
else if (x == 0) {
  println ("x is zero");}
else{
   for (int i =1; i<=x; i++){
     if (i % y ==){
    println (string .valueof (i));
}}}

public void test-y-is-zero() {
    output.clear();
    process (5,0);
  assert Equals (List.of ("y is zero"), output);
public void test-edge-case-y-negative(){
output.clear();
   process (5, -2);
assert Equals (List.of ("2", "4"), output);
public void test-edge-case-x-negative(){
      output.clear();
      process (-3, 2);
 assert True (output.is Empty());
    }
}
```

Subject / Theme: Mike Date: / /
☐ Sat ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

2.1

To solve this problem using JUnit 4, we will develop test cases that demonstrate the application of exception, setup functions and timeout rule.

Steps to Approach:

→ Setup Function: We'll use the @Before annotation to setup common objects or states needed before each other test.

→ Exception Testing: Well use the @Test (expected = Exception class) annotation to test whether the expected exception is thrown.

→ Timeout Rule: The @ Rule annotation can be used with the Timeout class to ensure that tests do not run longer than a specified duration.

Example Production Code:

```
public class calculator{
```

```
public int divide(int numerator,
    int denominator) throws Arithmatic
        Exception {
    if (denominator == 0) {
throw new ArithmeticException("can-
    not divide by zero");
    return numerator/denominator;
    }
public int slowOperation() throw
    InterruptedException() {
        Thread.sleep(3000);
        return 42;
    }
```

Key Benifits:

→ Early error Detection
→ Test setup
→ Time out management.