Idries Eagle-Masuak
6868228

Question 1

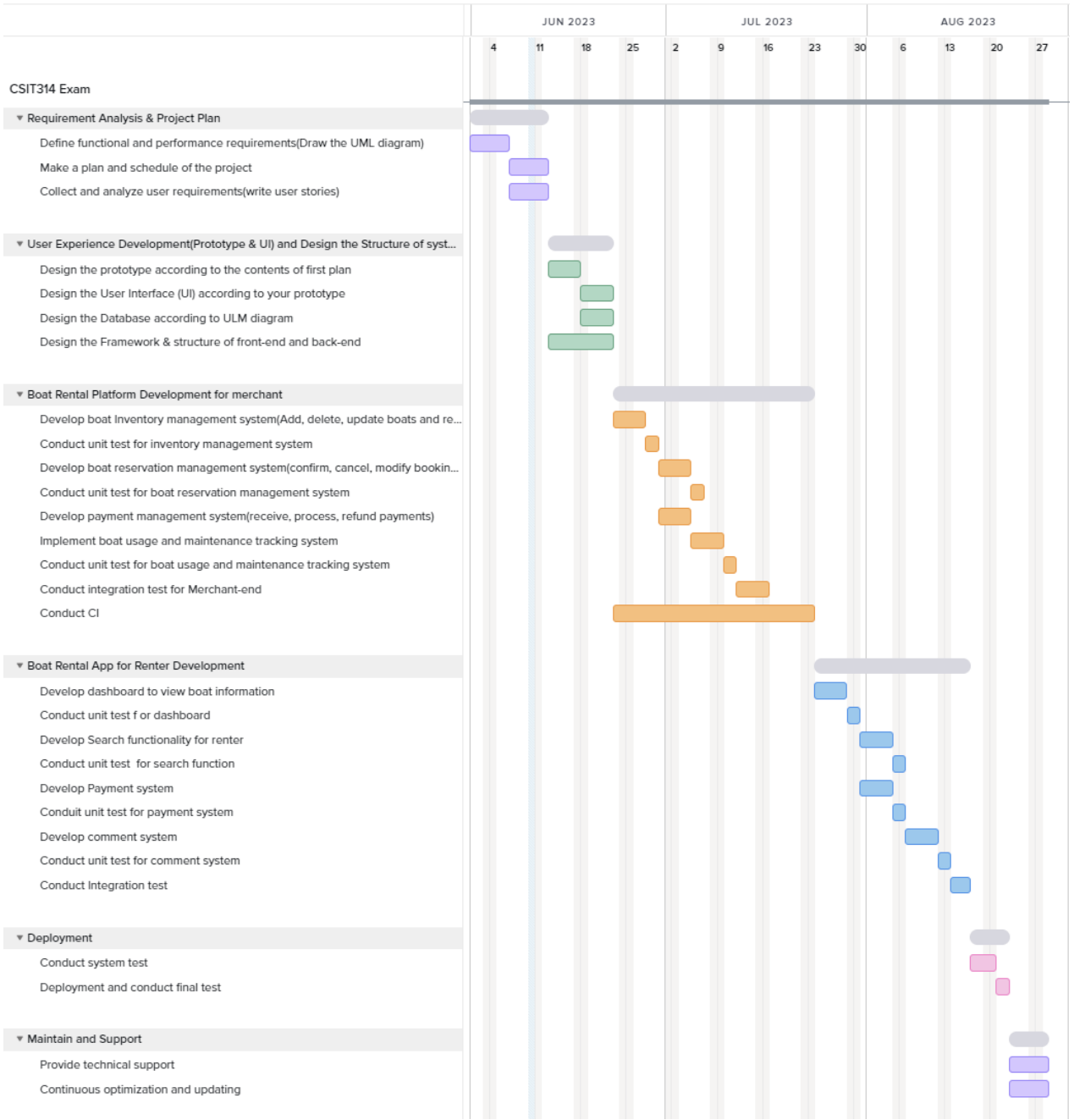
Assumption #1:

- There is a team of a size that's big enough to allow tasks to be completed at the same time

Issues with current plan:

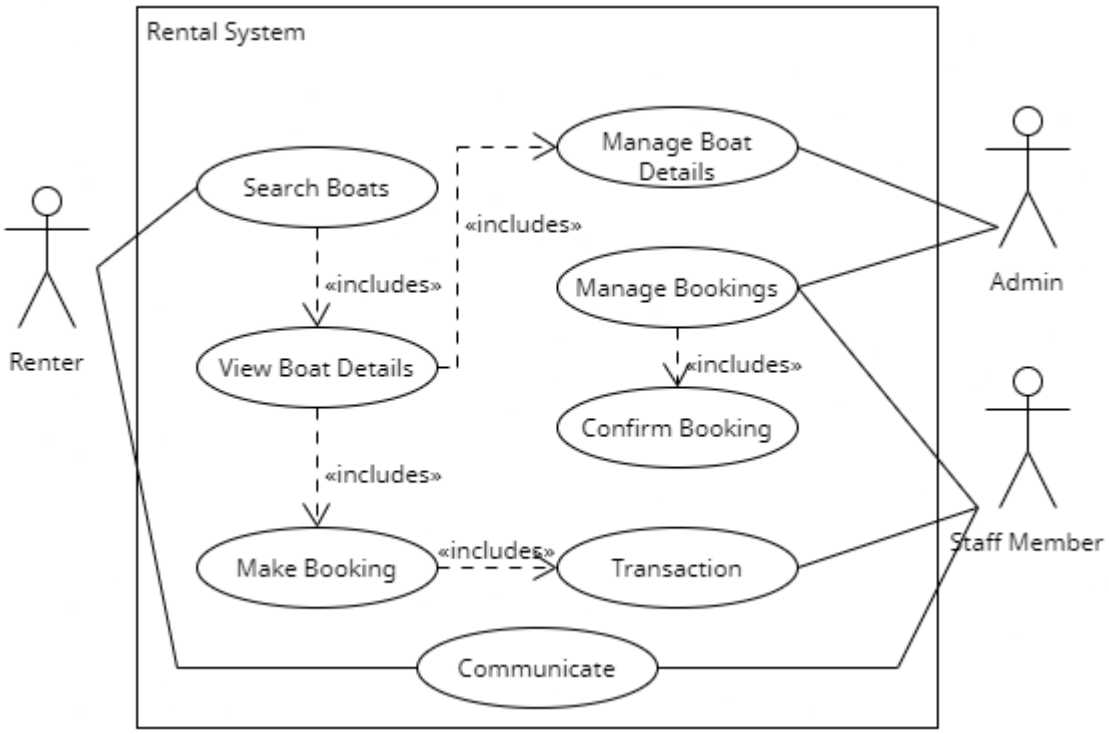
- BOAT-6
  - The total length of this part of the project doesn't equal the time taken BOAT-10, which means either BOAT-6 will need to take as long as BOAT-10, or BOAT-10 will need to take less time.
- BOAT-11
  - Same issue as Boat-6

Fixed Plan:



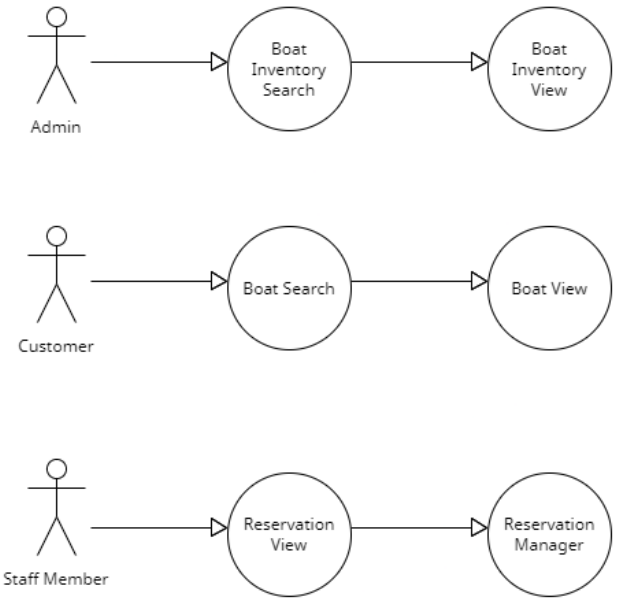
Idries Eagle-Masuak
6868228

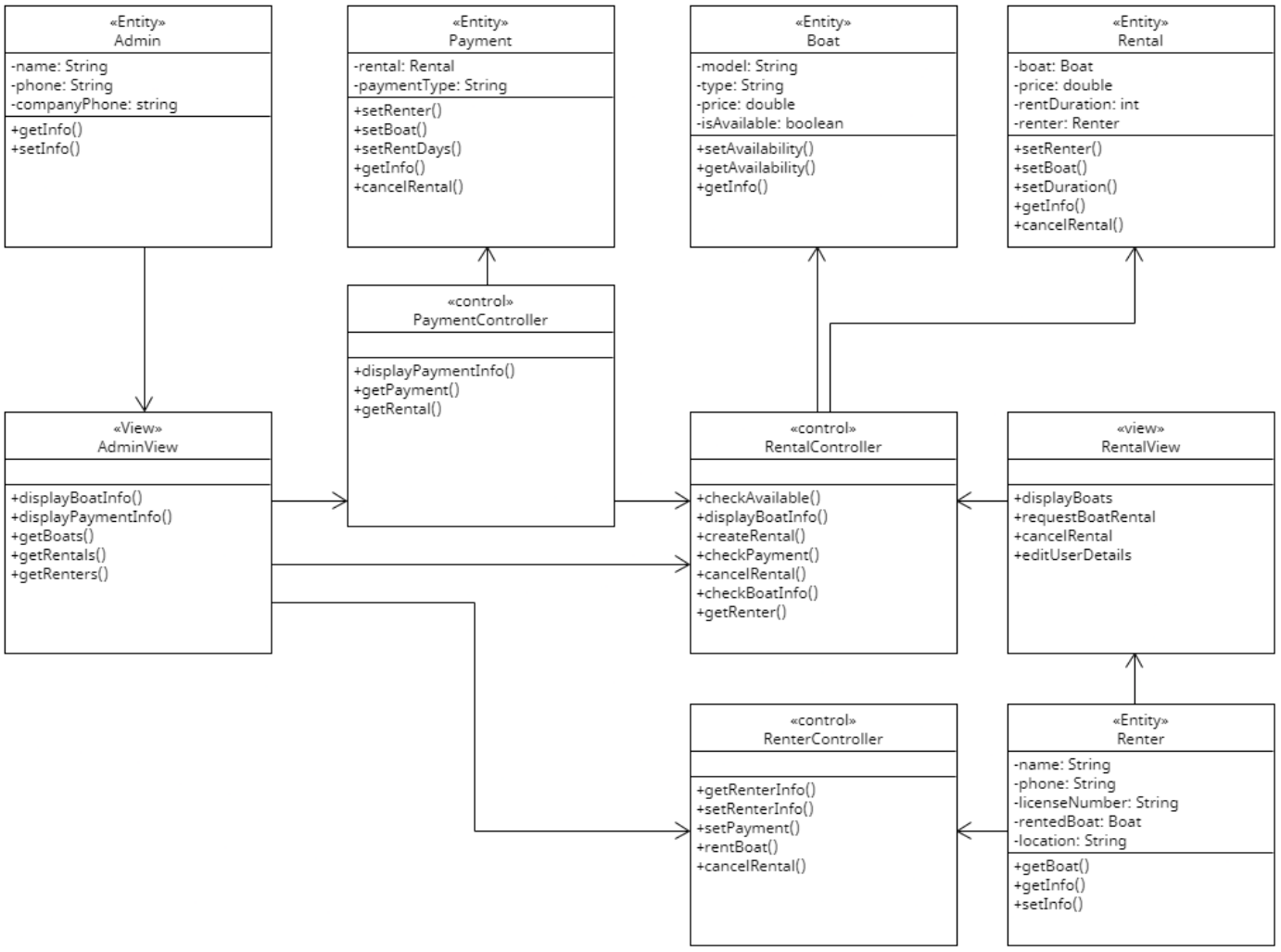
Question 2



Idries Eagle-Masuak
6868228

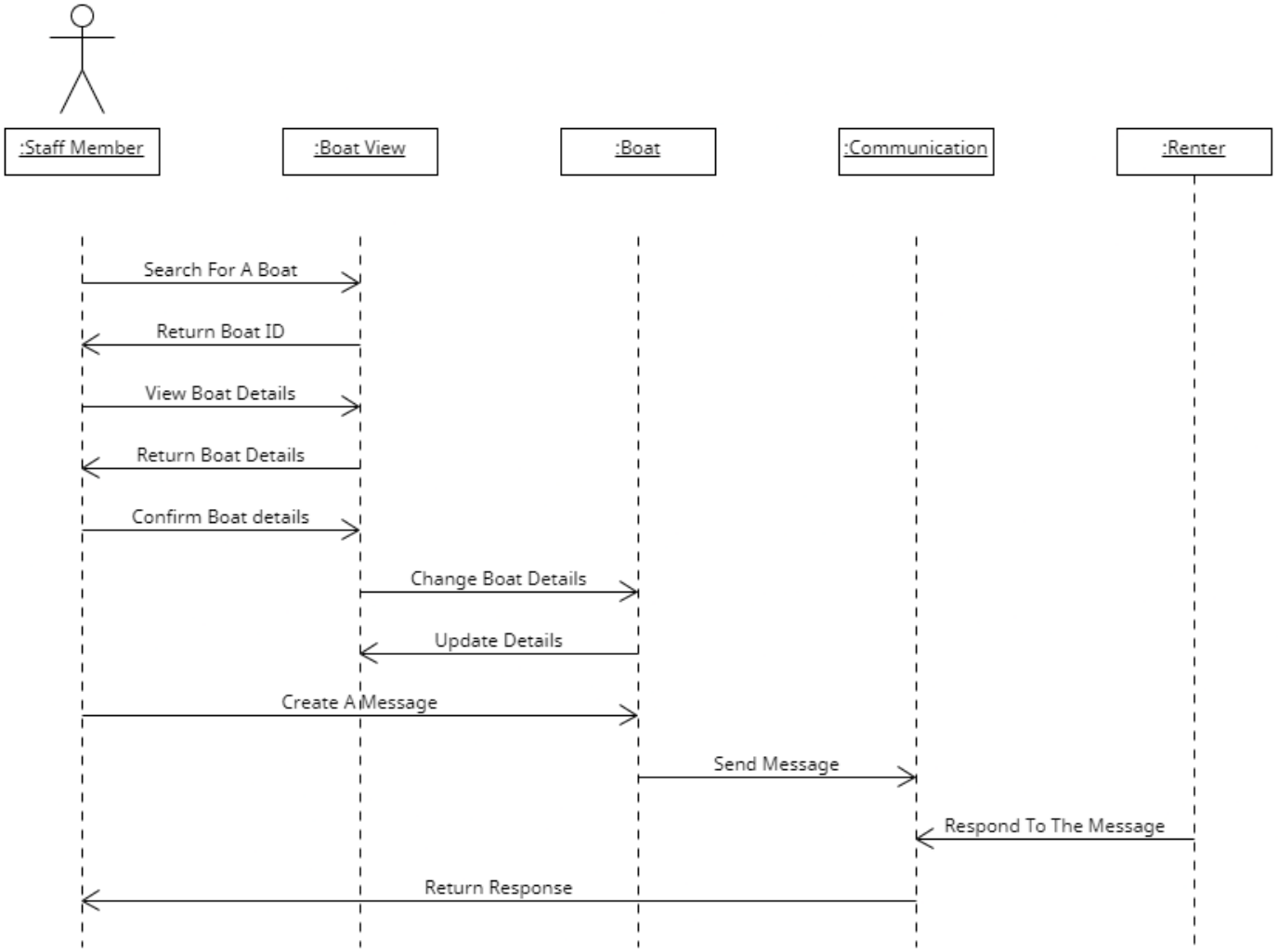
Question 3





Idries Eagle-Masuak
6868228

Question 4



Idries Eagle-Masuak
6868228

Question 5

1. Inception Phase
    - Objectives:
        - Understand the business case and scope of the project.
        - Identify stakeholders and their needs.
    - Activities:
        - Conduct initial requirements gathering and analysis.
        - Define the project vision and scope.
    - Outcomes:
        - Project vision document.
        - Initial use case model.
    - Milestones:
        - Lifecycle Objectives Milestone (LOM): Determines if the project is worth pursuing.

2. Elaboration Phase
    - Objectives:
        - Develop a solid architectural foundation for the project.
        - Mitigate risks and establish a stable development environment.
    - Activities:
        - Refine requirements and create a detailed use case model.
        - Define the system architecture and design.
    - Outcomes:
        - Software architecture document.
        - Detailed use case specifications.
    - Milestones:
        - Lifecycle Architecture Milestone (LAM): Determines if the architecture satisfies the requirements and is feasible.

3. Construction Phase
    - Objectives:
        - Build the system incrementally, focusing on feature implementation.
        - Conduct frequent testing and integration activities.
    - Activities:
        - Implement the system features based on the design.
        - Conduct unit testing and integration testing.
    - Outcomes:
        - Incremental releases of the software.
        - Test results and defect reports.
    - Milestones:
        - Initial Operational Capability (IOC): Demonstrates the system's initial functionality.

Idries Eagle-Masuak
6868228

4. Transition Phase
    - Objectives:
        - Prepare the system for deployment and user acceptance.
        - Conduct user training and finalize documentation.
    - Activities:
        - Perform system testing and user acceptance testing.
        - Prepare user manuals and training materials.
    - Outcomes:
        - Deployed system in the production environment.
        - User manuals and training materials.
    - Milestones:
        - Product Release Milestone: Marks the formal release of the software system.

Idries Eagle-Masuak
6868228

Question 6

Step 1: Define a Test Case

- Identify a specific functionality or behavior that you want to implement for the renter.
- Write a test case that verifies the expected outcome or behavior of that functionality.
- The test case should be written in a way that it can be automated and executed repeatedly.

Step 2: Run the Test Case

- Run the test case to confirm that it fails.
- Since there is no implementation yet, the test case should fail at this point.
- This step ensures that the test case is properly set up and ready for implementation.

Step 3: Write the Minimum Implementation

- Write the minimum code necessary to make the test case pass.
- Focus on implementing the simplest solution that fulfills the requirements of the test case.
- Avoid adding any unnecessary or complex code at this stage.
- The goal is to have a passing test case with the minimal implementation.

Step 4: Run All Test Cases

- Run all the previously written test cases, including the one you just implemented.
- Make sure that the new implementation does not break any existing functionality.
- This step ensures that the changes made to accommodate the new functionality do not introduce regressions.

Step 5: Refactor

- Refactor the code if necessary to improve its design, readability, or performance.
- This step aims to enhance the quality of the code without changing its functionality.
- Refactoring can include renaming variables, extracting methods, optimizing code, or applying design patterns.
- It is crucial to have a comprehensive set of test cases in place to catch any potential issues introduced during refactoring.

Step 6: Repeat the Process

- Repeat the above steps for each new functionality or behavior you want to implement.
- Start by defining a new test case, running it to fail, implementing the minimal code, and running all the test cases.
- Iterate through this process until all the desired functionalities for the renter have been implemented and tested.

Idries Eagle-Masuak
6868228

Test Cases:

Test Case 1: Test getBoat()

```
// Define the test case for getBoat()
testGetBoat() {
  // Create a sample renter object
  Renter renter = new Renter();

  // Set up the rented boat information
  renter.setBoatInfo("Boat1");

  // Call the getBoat() method and assert the expected result
  assert(renter.getBoat() == "Boat1");
}
```

Test Case 2: Test getInfo()

```
// Define the test case for getInfo()
testGetInfo() {
  // Create a sample renter object
  Renter renter = new Renter();

  // Set up the renter's information
  renter.setInfo("John Doe");

  // Call the getInfo() method and assert the expected result
  assert(renter.getInfo() == "John Doe");
}
```

Test Case 3: Test setInfo()

```
// Define the test case for setInfo()
testSetInfo() {
  // Create a sample renter object
  Renter renter = new Renter();

  // Set the renter's information using the setInfo() method
  renter.setInfo("Jane Smith");

  // Call the getInfo() method and assert the expected result
  assert(renter.getInfo() == "Jane Smith");
}
```

Idries Eagle-Masuak
6868228

Question 7

Based on the unique characteristics of the team members, the Scrum development model would be more appropriate for the boat rental system. Scrum is an agile framework that emphasizes iterative and incremental development, frequent collaboration, and flexibility in adapting to changing requirements. Here's the justification for selecting Scrum:

1. Team Characteristics:
    - Team Size: With 10 software engineers, Scrum provides a suitable structure for larger teams to collaborate effectively.
    - Cross-Functional Skills: Scrum teams are typically cross-functional, meaning team members have diverse skill sets. This can be beneficial for a project that involves multiple aspects such as frontend development, backend development, database management, and user interface design.
    - Adaptive and Collaborative: Scrum encourages close collaboration and adaptability, which can be advantageous when working with a team that has different perspectives and skills. It allows for regular feedback and adjustments throughout the development process.

Implementation Strategies and Anticipated Outcomes:

1. Formation of Scrum Team:
    - Assign a Scrum Master: Appoint a team member who will act as the Scrum Master, responsible for facilitating Scrum events and ensuring the team follows Scrum principles.
    - Define Roles: Clearly define the roles and responsibilities of each team member, such as Product Owner, Development Team, and Scrum Master.

2. Product Backlog and Sprint Planning:
    - Collaborative Backlog Refinement: Conduct regular backlog refinement sessions to prioritize and refine user stories and requirements. Involve all team members in the process to ensure a shared understanding.

    - Sprint Planning: Plan and prioritize the user stories for each sprint in collaboration with the Product Owner. Consider the team's capacity and estimated effort required for each user story.

3. Sprint Execution:
    - Time-Boxed Sprints: Divide the development process into time-boxed iterations called sprints, typically ranging from 1-4 weeks. Encourage the team to deliver a potentially shippable increment of the product by the end of each sprint.
    - Daily Scrum Meetings: Conduct daily stand-up meetings to provide updates on progress, discuss any impediments, and plan the day's work.

4. Continuous Integration and Testing:
    - Continuous Integration: Implement a continuous integration process to integrate code changes frequently and detect any integration issues early.
    - Automated Testing: Encourage the use of automated testing to ensure the quality of the software and facilitate faster feedback.

5. Sprint Review and Retrospective:
    - Sprint Review: Conduct sprint review meetings at the end of each sprint to showcase the completed work to stakeholders and gather feedback.

- Sprint Retrospective: Hold sprint retrospective meetings to reflect on the sprint and identify areas for improvement.

Anticipated Outcomes:

- Improved Collaboration: Scrum promotes regular collaboration among team members, facilitating knowledge sharing and problem-solving.
- Increased Flexibility: The iterative nature of Scrum allows for flexibility in accommodating changing requirements or priorities.
- Early and Frequent Feedback: Scrum's emphasis on regular feedback helps identify and address issues early in the development process.
- Higher Customer Satisfaction: The involvement of the Product Owner and regular sprint reviews ensure alignment with customer needs and expectations.
- Continuous Improvement: The sprint retrospectives provide an opportunity to reflect on the team's performance and identify areas for improvement, leading to increased productivity over time.