Department of Computer Engineering

Academic Term: First Term 2023-24

Class: T.E /Computer Sem – V / Software Engineering

Practical No:	**2**
Title:	**Implementing Project using SCRUM method on JIRA Tool**
Date of Performance:	**03/08/2023**
Roll No:	9590
Team Members:	

Rubrics for Evaluation:

Sr. No	Performance Indicator	Excellent	Good	Below Average	Total Score
1	On time Completion & Submission (01)	01 (On Time)	NA	00 (Not on Time)	
2	Theory Understanding(02)	02(Correct)	NA	01 (Tried)	
3	Content Quality (03)	03(All used)	02 (Partial)	01 (rarely followed)	
4	Post Lab Questions (04)	04(done well)	3 (Partially Correct)	2(submitted)	

Signature of the Teacher:

Lab Experiment 02

Experiment Name: Implementing Project Using Scrum Method on JIRA Tool in Software Engineering

Objective: The objective of this lab experiment is to introduce students to the Scrum framework and its implementation using the JIRA tool. Students will gain practical experience in managing a software project using Scrum principles and learn how to utilize JIRA as a project management tool to track and organize tasks, sprints, and team collaboration.

Introduction: Scrum is an agile project management methodology that promotes iterative development, collaboration, and continuous improvement. JIRA is a widely used tool that supports Scrum practices, providing teams with features to plan, track, and manage software projects effectively.

Lab Experiment Overview:

1. Introduction to Scrum: The lab session begins with an overview of the Scrum framework, including its roles (Product Owner, Scrum Master, and Development Team), events (Sprint Planning, Daily Standup, Sprint Review, and Sprint Retrospective), and artifacts (Product Backlog, Sprint Backlog, and Increment).
2. JIRA Tool Introduction: Students are introduced to the JIRA tool and its capabilities in supporting Scrum project management. They learn to create projects, epics, user stories, tasks, and sub-tasks in JIRA.
3. Defining the Project: Students are assigned a sample software project and create a Product Backlog, listing all the required features, user stories, and tasks for the project.
4. Sprint Planning: Students organize the Product Backlog into Sprints, selecting user stories and tasks for the first Sprint. They estimate the effort required for each task using story points.
5. Implementation in JIRA: Students use the JIRA tool to create a Sprint Backlog, add the selected user stories and tasks, and assign them to team members.
6. Daily Standup: Students conduct a simulated Daily Standup meeting, where they update the progress of their tasks and discuss any impediments they are facing.
7. Sprint Review and Retrospective: At the end of the Sprint, students review the completed tasks, demonstrate the implemented features, and gather feedback from their peers. They also conduct a Sprint Retrospective to identify areas of improvement for the next Sprint.
8. Continuous Iteration: Students continue implementing subsequent Sprints, repeating the Sprint Planning, Daily Standup, and Sprint Review & Retrospective events.
9. Conclusion and Reflection: At the end of the lab experiment, students reflect on their experience with Scrum and JIRA, discussing the advantages and challenges they encountered during the project.

Learning Outcomes: By the end of this lab experiment, students are expected to:

- Understand the Scrum framework and its principles in agile project management.

- Gain practical experience in using the JIRA tool for project management in a Scrum environment.
- Learn to create and manage Product Backlogs, Sprint Backlogs, and track progress using JIRA.
- Develop collaborative skills through Daily Standup meetings and Sprint Reviews.
- Gain insights into the iterative nature of software development and the importance of continuous improvement.

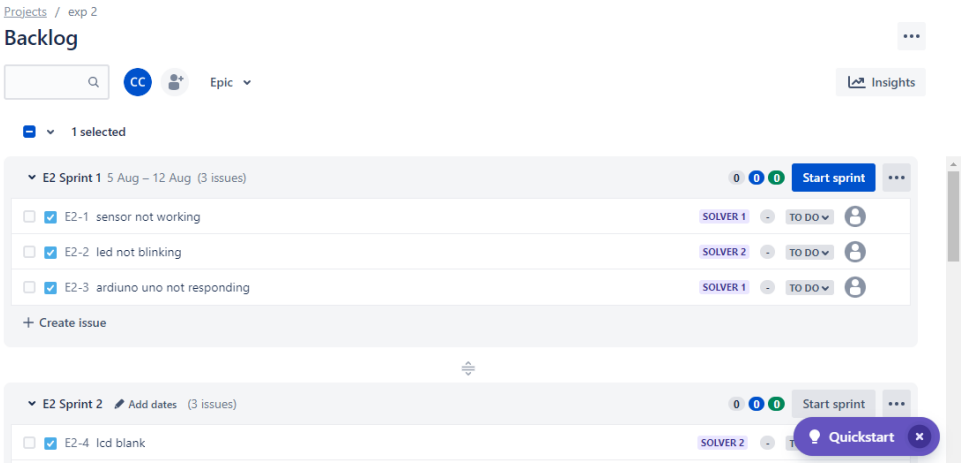
Pre-Lab Preparations: Before the lab session, students should familiarize themselves with the Scrum framework and the basics of the JIRA tool. They should review Scrum roles, events, and artifacts, as well as the features of JIRA relevant to Scrum implementation.

Materials and Resources:

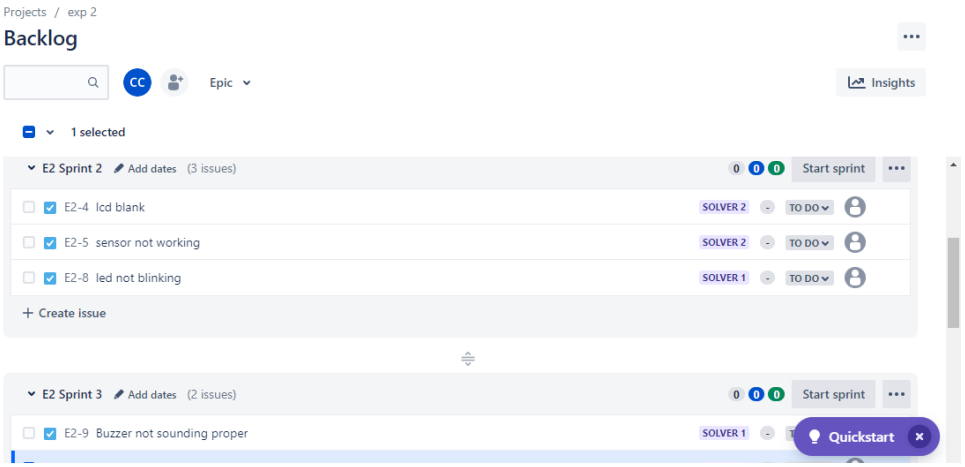
- Computers with internet access for accessing the JIRA tool
- Project brief and details for the sample software project
- Whiteboard or projector for explaining Scrum concepts

Conclusion: The lab experiment on implementing a project using Scrum on the JIRA tool offers students a hands-on experience in agile project management. By utilizing Scrum principles and JIRA's capabilities, students learn to collaborate effectively, manage tasks efficiently, and adapt to changing requirements. The practical exposure to Scrum and JIRA enhances their understanding of agile methodologies, equipping them with valuable skills for real-world software development projects. The lab experiment encourages students to embrace the agile mindset, promoting continuous improvement and customer-centric software development practices.

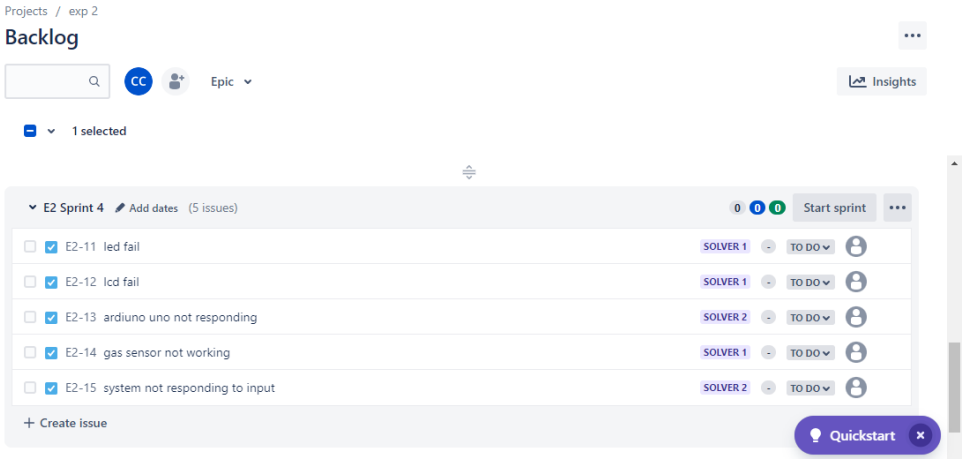
1. Creating the Problems/ Backlog



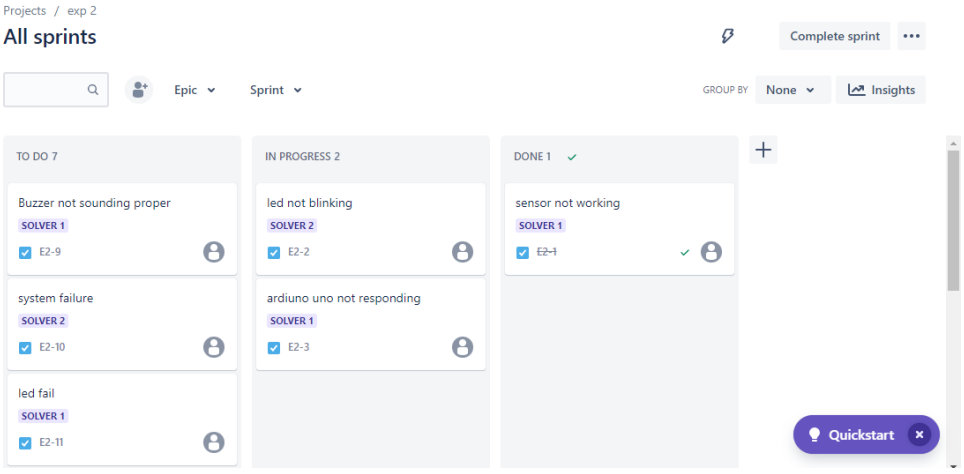
2. Assigning the EPIC



3. Starting the Sprint



4. Checking the board, To do, in progress, done.



Postlab Questions:

a) Assess the effectiveness of the Scrum framework for managing software development projects compared to traditional project management methodologies.

Assessing the effectiveness of the Scrum framework for managing software development projects compared to traditional project management methodologies requires considering various factors and the specific context of the project. Let's compare the two approaches based on key aspects:

1. Flexibility and Adaptability:

Scrum: Scrum is highly flexible and adaptive, allowing teams to respond to changing requirements and priorities quickly. It follows an iterative and incremental approach, where work is divided into short sprints, typically two to four weeks long. The team regularly reviews and adapts the product backlog, ensuring that the most valuable features are delivered first.

Traditional Project Management: Traditional methodologies, such as Waterfall, are more rigid and linear. They involve detailed planning upfront and a fixed scope. Changes in requirements during development can be challenging and costly to accommodate.

2. Customer Focus:

Scrum: Scrum emphasizes continuous feedback from stakeholders, including the product owner and end-users. The product backlog prioritization is based on customer needs, ensuring that the most valuable features are delivered early in the development process.

Traditional Project Management: Traditional methodologies may not prioritize customer feedback as prominently. Requirements are often defined at the beginning of the project and may not be revisited until the end.

3. Transparency and Visibility:

Scrum: Scrum promotes transparency through various ceremonies, such as daily stand-ups, sprint reviews, and sprint retrospectives. Progress, challenges, and achievements are visible to the entire team.

Traditional Project Management: Traditional approaches may not provide the same level of transparency, leading to potential communication gaps and reduced visibility into project progress.

4. Team Collaboration:

Scrum: Scrum encourages collaborative, cross-functional teams. The roles of Scrum Master, Product Owner, and Development Team foster a shared sense of responsibility and ownership.

Traditional Project Management: Traditional methodologies may follow a more hierarchical approach, leading to potential communication barriers and less emphasis on team collaboration.

5. Predictability and Planning:

Scrum: Scrum provides a predictable development cadence through sprint iterations, allowing stakeholders to plan and prioritize upcoming releases.

Traditional Project Management: Traditional methodologies often involve detailed upfront planning, but changes during development can impact project predictability.

b) Analyse a Sprint Backlog in JIRA and identify any potential bottlenecks or issues that might hinder the team's progress during the sprint.

Some common issues that might hinder the team's progress during the sprint:

1. Overloaded Sprint Backlog: If the Sprint Backlog contains too many user stories or tasks, it can lead to overcommitment and make it difficult for the team to complete everything within the sprint timeline. This can lead to a decrease in overall productivity and quality.

2. Unclear or Unrefined User Stories: User stories that lack clarity or are not well-defined can cause confusion and lead to delays in development. The team may spend extra time seeking clarification or reworking the implementation.

3. Lack of Task Breakdown: If user stories are not broken down into smaller, manageable tasks, it can be challenging for team members to collaborate effectively, estimate efforts accurately, and track progress efficiently.

4. Dependencies and Blocked Items: Dependencies between user stories or tasks can create bottlenecks if the completion of one task is contingent on another. Similarly, blocked items (tasks that cannot progress due to external factors) can slow down the team's overall progress.

5. Insufficient Test Coverage: Lack of adequate testing or testing delays can lead to the accumulation of bugs, reducing the team's capacity to work on new items.

6. Communication and Collaboration Issues: Inadequate communication and collaboration between team members can lead to misunderstandings, duplication of efforts, and decreased productivity.

c) Evaluate the role of the Scrum Master in handling conflicts within the development team and resolving impediments to maintain a smooth project flow.

The Scrum Master plays a crucial role in handling conflicts within the development team and resolving impediments to maintain a smooth project flow in a Scrum environment. Their primary responsibility is to facilitate and support the Scrum process and ensure that the team can work effectively and efficiently. Here's how the Scrum Master's role is essential in managing conflicts and impediments:

1. Conflict Resolution:

Facilitation: The Scrum Master acts as a neutral facilitator during team interactions and helps team members communicate effectively. They create a safe and open environment where conflicts can be addressed constructively.

Mediation: When conflicts arise between team members, the Scrum Master mediates the situation, encouraging active listening and helping to find mutually acceptable resolutions.

Conflict Management: The Scrum Master helps the team understand that conflict is a natural part of collaboration and can lead to growth and improvement when managed properly. They guide the team towards constructive conflict resolution.

2. Impediment Removal:

Identifying Impediments: The Scrum Master actively seeks out impediments that are hindering the team's progress. These impediments can be related to tools, processes, team dynamics, or external factors.

Team Empowerment: The Scrum Master empowers the team to identify impediments and encourages them to take ownership of resolving issues that are within their control.

Escalation: For impediments that are beyond the team's control, the Scrum Master escalates the issues to the appropriate stakeholders or management to ensure they are addressed promptly.

3. Team Collaboration:

Collaboration Facilitation: The Scrum Master promotes collaboration and effective communication within the team, helping to build a strong sense of teamwork and trust.

Cross-Functional Cooperation: They encourage cross-functional cooperation and help team members understand each other's roles and contributions.

4. Removing Obstacles:

Organizational Support: The Scrum Master collaborates with stakeholders and management to address organizational impediments that affect the team's productivity.

Resource Allocation: They work to ensure that the team has the necessary resources, tools, and support to deliver the project successfully.