Department of Computer Engineering

Academic Term: First Term 2023-24

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

Practical No:	02
Title:	Implementing Project Using Scrum Method on JIRA Tool in Software Engineering
Date of Performance:	03/08/2023
Roll No:	9589
Team Members:	Mudabbir Bhat(9589), Muhammad Batliwala(9588), Nathan Dias(9597), Aqib Firdous(9614)

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: This lab experiment aims 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.

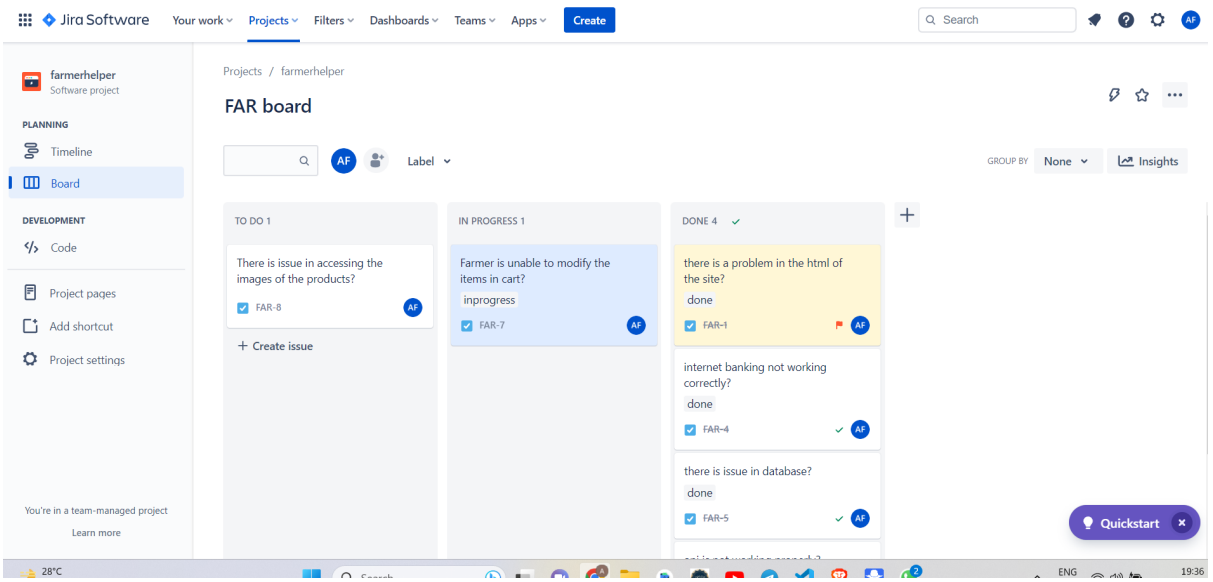
Introduction: Scrum is an agile project management methodology that promotes iterative development, collaboration, and continuous improvement. JIRA is a widely used tool that supports Scrum practices, provide teams with features to plan, track, and manage software projects effectively.

Lab Experiment Overview:

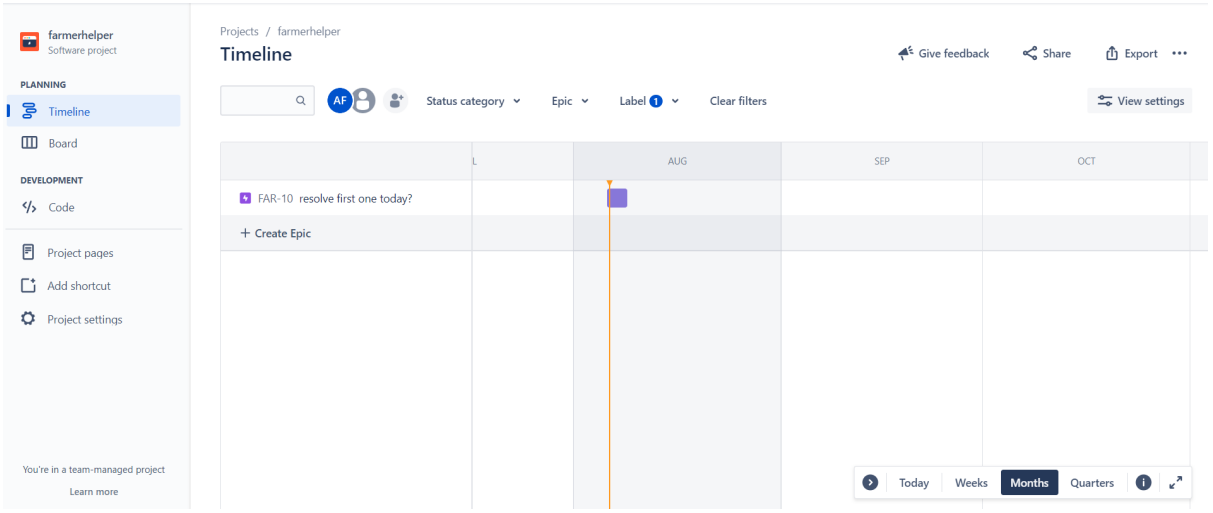
1. Introduction to Scrum: The lab session begins with an overview of the Scrum framework, including its roles (Product Owner, Scrum Master, and Development Team), events (Sprint Planning, Daily Standup, Sprint Review, and Sprint Retrospective), and artifacts (Product Backlog, Sprint Backlog, and Increment).
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, artifacts, and 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.

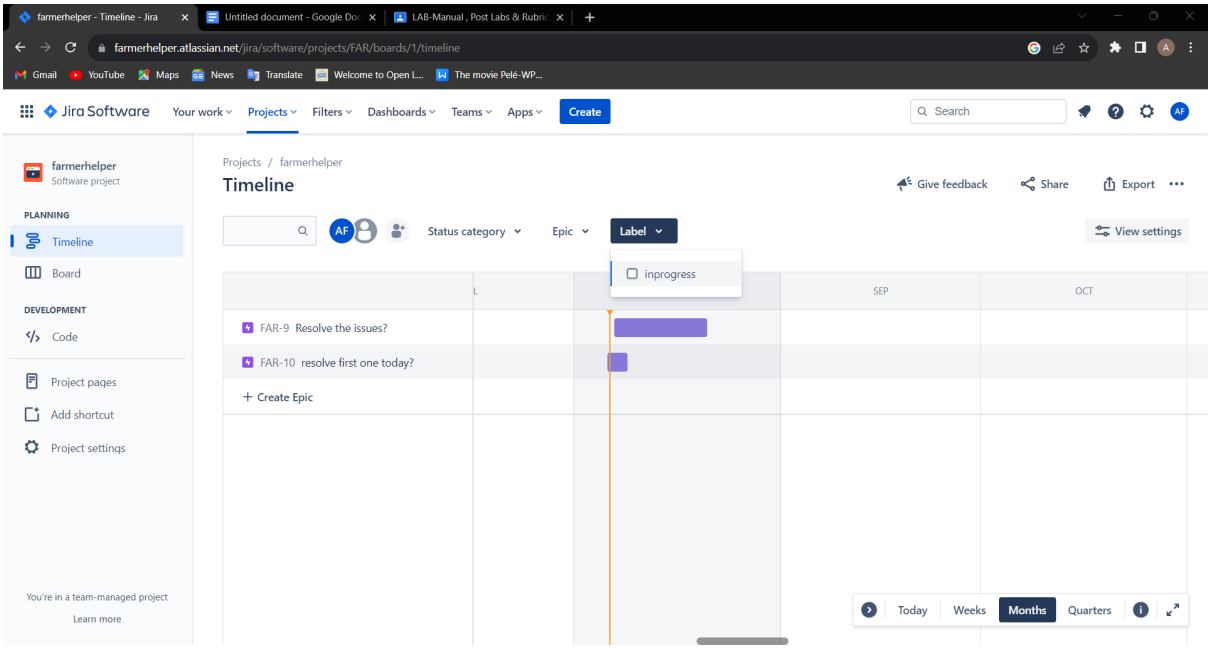
As the topic we had for our mini project was the farmer helper site so we tried it by taking issues that can be present and tried to resolve them with the help of scrum. As we know with the help of Jira we can assign a particular issue to a particular person, therefore it is a very important software tool. Some of the issues are mentioned below:



And I have tried resolving some issues and resolved some of these issues and others are in progress.



\



2. **Implementing the Project using the SCRUM method on JIRA Tool**:
a) Assess the effectiveness of the Scrum framework for managing software development projects compared to traditional project management

methodologies.

Ans: Assessing the effectiveness of the Scrum framework for managing software development projects compared to traditional project management methodologies involves considering various aspects such as project flexibility, team collaboration, adaptability, communication, and overall project success. Let's break down these aspects and evaluate how Scrum compares to traditional methods using the JIRA tool.
1. Flexibility and Adaptability:
2. Team Collaboration:
3. Communication
4. Customer Involvement
5. Risk Management

Using JIRA for Scrum: JIRA is a powerful tool for implementing Scrum, providing features for backlog management, sprint planning, task tracking, and reporting. JIRA's boards, backlogs, and sprint tracking can enhance Scrum practices by providing a digital platform for managing tasks and visualizing progress.

In conclusion, Scrum offers advantages in terms of adaptability, collaboration, communication, and customer involvement. It can be more effective for managing software development projects where requirements are subject to change. However, the choice between Scrum and traditional methods depends on the specific project's characteristics, team culture, and stakeholder preferences. Utilizing JIRA to

implement Scrum can further enhance its benefits by providing a comprehensive toolset for managing the Scrum process.

b) Analyse a Sprint Backlog in JIRA and identify any potential bottlenecks or issues that might hinder the team's progress during the sprint.
Ans: Analyzing a Sprint Backlog in JIRA for potential bottlenecks or issues requires a careful examination of the tasks, user stories, dependencies, priorities, and team capacity. Here's a step-by-step analysis to identify potential obstacles:

1. Task Breakdown: Look at how user stories have been broken down into tasks. If tasks are too large or unclear, they might lead to delays or confusion. Conversely, if tasks are too granular, it might indicate overcomplication.
Dependencies: Identify any task dependencies. If tasks are tightly interdependent and cannot be worked on concurrently, it can slow down progress. Also, if a critical task is dependent on another task that's not yet complete, it might become a bottleneck.

2. Task Assignment: Check if tasks are evenly distributed among team members. An uneven distribution can lead to some team members being overloaded while others are underutilized.

3. Priorities: Ensure that tasks are prioritized properly. If high-priority tasks are too complex and lower-priority tasks are simpler, it might lead to the team spending too much time on less impactful work.

4. Task Estimates: Review the task estimates. If tasks are consistently underestimated, the team might run out of time during the sprint. Overestimated tasks might lead to unnecessary delays.
Blocking Issues: Identify any blocking issues or impediments that might hinder the team's progress. These could include external dependencies, lack of resources, technical challenges, or waiting for approvals.

5. Capacity vs. Workload: Compare the team's capacity (available hours) with the total workload in the Sprint Backlog. If the workload exceeds the team's capacity, it could lead to burnout, lower-quality work, or incomplete tasks.

6. Emergent Work: Be prepared to accommodate emergent work that arises during the sprint. If the team is fully booked with planned tasks, they might struggle to handle unexpected tasks or urgent bug fixes.

7. Testing and Review: Ensure that tasks related to testing, quality assurance, and review are adequately represented. Neglecting these tasks can lead to a rush at the end of the sprint and compromise product quality.
8. Communication and Collaboration: Check if there's proper communication and collaboration among team members. If team members are working in isolation or not sharing progress, it can lead to misunderstandings and rework.

9. Scope Changes: Assess if there have been any scope changes during the sprint. Adding new tasks or user stories without adjusting priorities or extending the sprint can lead to overload.
10. Feedback Loops: Evaluate if there are mechanisms for obtaining feedback from stakeholders, Product Owners, and team members. Lack of feedback can result in misaligned expectations.

By thoroughly analyzing these aspects of the Sprint Backlog in JIRA, you can identify potential bottlenecks or issues that might hinder the team's progress during the sprint. Addressing these challenges proactively can help ensure a smoother and more successful sprint.

c) Evaluate the role of the Scrum Master in handling conflicts within the the development team and resolving impediments to maintain a smooth project flow.
Ans: The Scrum Master plays a crucial role in handling conflicts within the development team and resolving impediments to maintaining a smooth project flow in a Scrum environment. Here's an evaluation of the Scrum Master's role in these aspects:

Handling Conflicts:

1. Facilitator and Mediator: The Scrum Master acts as a neutral facilitator and mediator in conflicts. They create a safe environment for open communication and help team members express their concerns and viewpoints.
2. Conflict Identification: Scrum Masters are attentive to team dynamics and can identify conflicts early. They actively listen, observe interactions, and proactively address emerging conflicts.
3. Conflict Resolution Techniques: Scrum Masters use various conflict resolution techniques, such as active listening, coaching, and collaboration. They guide the team toward finding mutually agreeable solutions.
4. Coaching and Empowerment: Scrum Masters coach team members on effective communication and collaboration. They empower individuals to resolve conflicts directly when possible, promoting self-organization.
5. Escalation: If conflicts escalate beyond the team's capacity, the Scrum Master escalates the issue to higher management or stakeholders, ensuring timely resolution.

Resolving Impediments:

1. Identifying Impediments: Scrum Masters actively identify impediments that hinder the team's progress. They encourage the team to raise issues and collaborate on finding solutions.
Removing Obstacles: The Scrum Master takes ownership of removing

obstacles that the team cannot resolve themselves. This could involve coordinating with other teams, stakeholders, or management to resolve issues.
2. Problem-Solving: Scrum Masters facilitate problem-solving sessions where the team brainstorms solutions to impediments. They encourage creativity and innovation in overcoming challenges.
3. Tracking and Reporting: Scrum Masters keep track of impediments, their status, and their resolutions. This information is vital for retrospective discussions and continuous improvement.
4. Continuous Improvement: Scrum Masters use impediments as opportunities for learning and improvement. They analyze recurring issues and work towards preventing them in future sprints.
5. Prioritization: When multiple impediments arise, the Scrum Master helps the team prioritize based on the impact on project goals and sprint commitments.

Communication and Collaboration:

1. Stakeholder Management: The Scrum Master interacts with stakeholders to manage expectations and address any impediments that might be external to the team.
2. Team Alignment: Scrum Masters ensure the team is aligned with the Scrum framework and its principles. They facilitate discussions to maintain clarity and focus on sprint goals.
3. Transparency: Scrum Masters maintain transparency by making impediments visible to the entire team. This encourages collaboration in finding solutions.

In summary, the Scrum Master plays a pivotal role in handling conflicts within the development team and resolving impediments. By fostering effective communication, using conflict resolution techniques, and actively addressing impediments, the Scrum Master helps maintain a smooth project flow, enabling the team to stay focused, productive, and aligned with their goals.