Department of Computer Engineering

Academic Term: First Term 2023-24

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

Practical No:	**2**
Title:	**Software Requirement Specification**
Date of Performance:	**03/08/2023**
Roll No:	**9592**
Team Members:	**Aston, Chhand, Slayde**

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:

Department of Computer Engineering

Academic Term: First Term 2022-23

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

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.

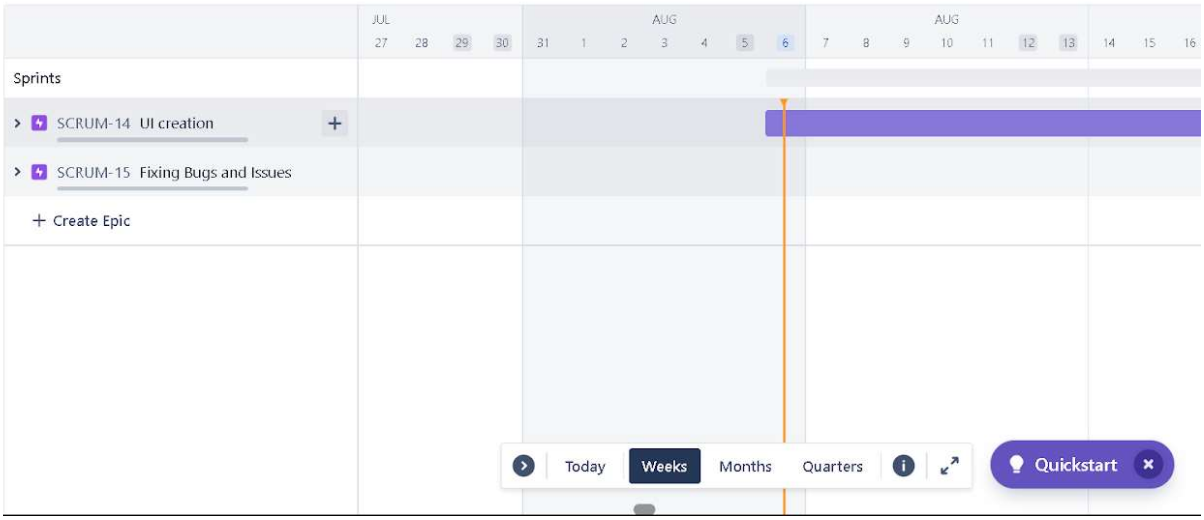
Materials and Resources:

- 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 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.

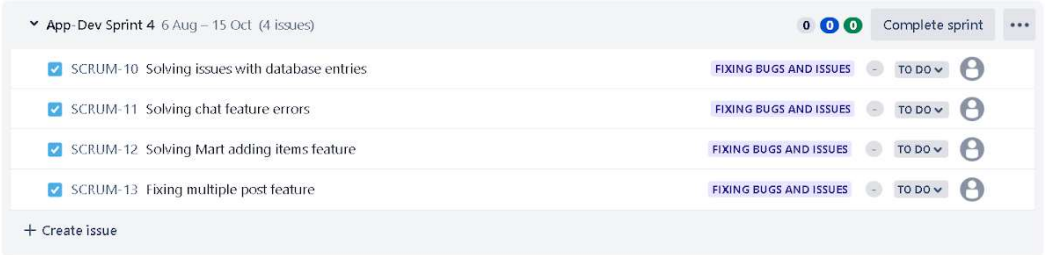
Outpt

EPIC

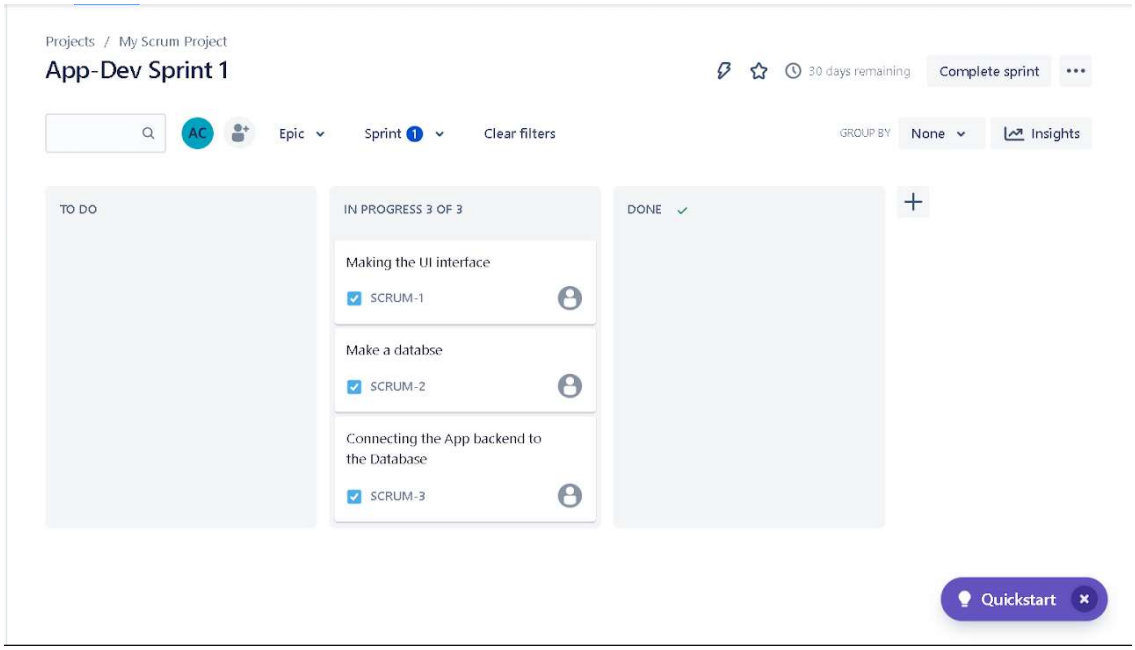


Backlog

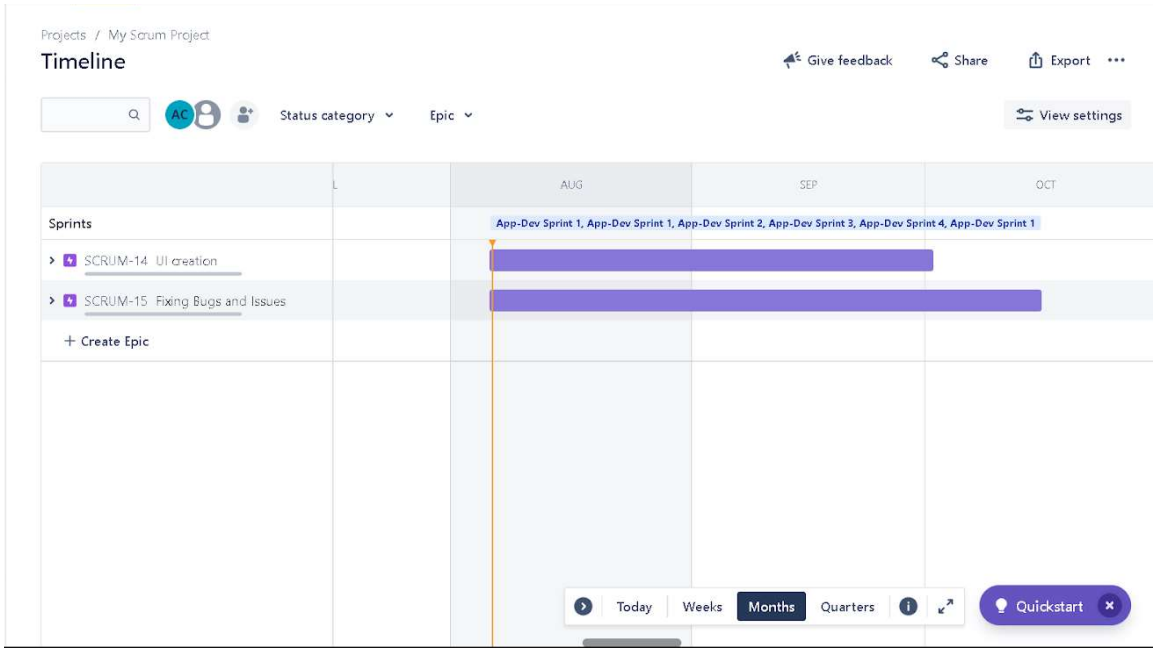
Sprint



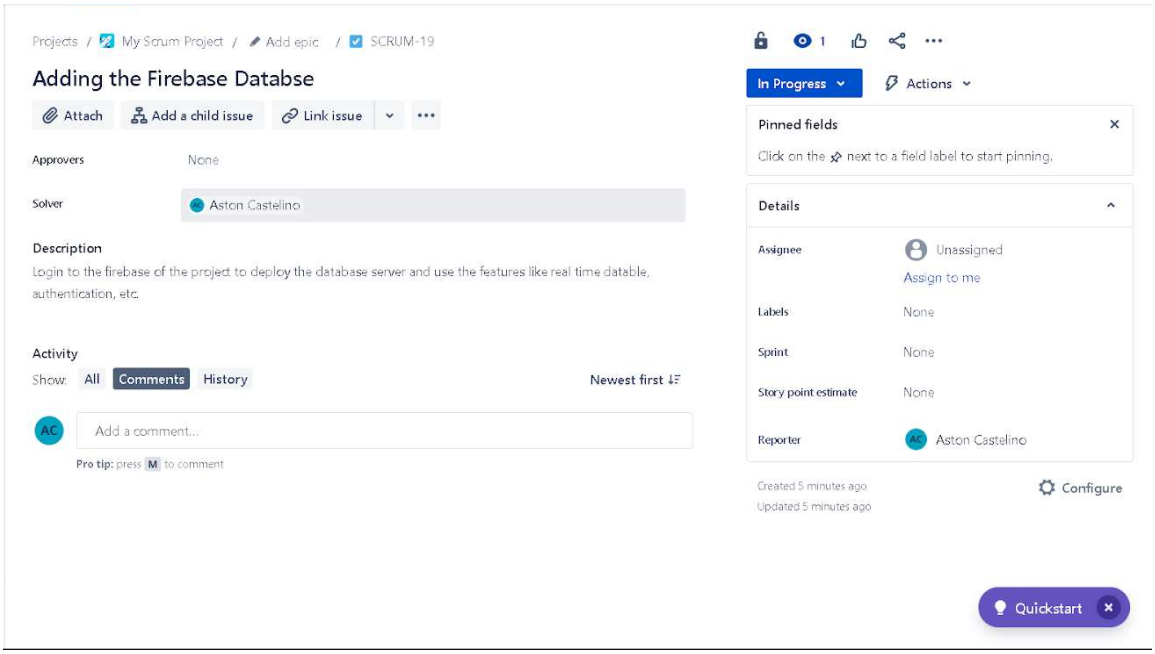
Sprint Board



Timeline



Tasks



PostLabs

a)
Scrum Framework:

Advantages:

Flexibility and Adaptability: Scrum is highly adaptable to changing requirements and priorities, making it well-suited for projects with evolving or unclear requirements.
Incremental Delivery: Scrum emphasizes delivering working increments of the software in short iterations (sprints), which allows for regular feedback and early value delivery.
Stakeholder Collaboration: Scrum encourages constant communication and collaboration between cross-functional teams and stakeholders, leading to better alignment with user needs.
Transparency: Scrum's ceremonies (daily stand-ups, sprint reviews, etc.) provide transparency into project progress and potential challenges.
Continuous Improvement: The retrospective process encourages teams to reflect on their work and make adjustments for continuous improvement.

Disadvantages:

Inexperienced Teams: Scrum relies on self-organizing teams, which might struggle if they lack experience or discipline.

Lack of Predictability: The focus on adaptability can sometimes lead to challenges in predicting project timelines and outcomes.
Minimal Documentation: Scrum values working software over comprehensive documentation, which might not align well with highly regulated industries.

Traditional Project Management:

Advantages:

Predictability: Traditional methodologies (e.g., Waterfall) provide a clear plan and timeline from the start, making them suitable for projects with well-defined requirements.
Comprehensive Documentation: Detailed documentation at each phase ensures clear communication and is advantageous in industries with strict regulations.
Resource Allocation: Traditional methods allow for precise resource allocation and upfront planning.
Scope Control: Strict scope control reduces the risk of scope creep during the project.

Disadvantages:

Rigidity: Traditional methodologies can struggle to adapt to changing requirements, making them less suitable for projects with evolving needs. **Limited Stakeholder Involvement:** Traditional methods might involve stakeholders less frequently, potentially leading to misalignment with user needs. **Late Feedback:** Feedback is often received late in the process, which can result in significant changes or rework.
High Risk: Traditional methods can have higher risks due to late identification of problems or misunderstandings.

Effectiveness Comparison:

Scrum is more effective when:

- Requirements are uncertain or evolving.
- Collaboration and stakeholder involvement are crucial.
- Quick value delivery and continuous improvement are important.
- The project requires flexibility and adaptability.

Traditional methods are more effective when:

- Requirements are well-defined and unlikely to change.
- Comprehensive documentation is necessary.
- Predictability and strict control are priorities.
- The project is in a highly regulated industry.

b.)

Analysis of Potential Bottlenecks or Issues from Experiment 1 Project in JIRA:

Uneven Task Distribution: The task distribution is uneven, with some team members having multiple tasks while others have none. This might lead to a lack of balance in workloads and delays in completing specific features.
Dependency on Single Team Members: If one team member (e.g., Sarah) is assigned to critical tasks across different user stories, the team's progress could be affected if that person faces any delays or issues.
Inadequate Testing Coverage: There are no tasks specifically assigned for testing user stories. This could lead to inadequate testing, reduced quality, and potential issues later in the sprint.
Unstarted Tasks: Several tasks are marked as "Not Started." If these tasks are not initiated promptly, they might become bottlenecks, delaying the completion of user stories.

Lack of Task Breakdown: Some user stories (e.g., "Pet Listing") might require more detailed task breakdowns to ensure a clear understanding of what needs to be accomplished.
Missing Acceptance Criteria: If the user stories lack well-defined acceptance criteria, it could lead to misunderstandings about when a user story is truly complete.
Communication and Coordination: The team might face communication challenges if there's a lack of coordination among team members, especially if they're working on interdependent tasks.

c.)

The Scrum Master plays a critical role in handling conflicts within the development team and resolving impediments to maintain a smooth project flow in a Scrum framework. Their responsibilities encompass fostering a collaborative and productive environment while ensuring that the team can effectively follow the Scrum methodology. Here's an evaluation of the Scrum Master's role in conflict resolution and impediment removal:

Conflict Resolution:

Observation and Detection: The Scrum Master actively monitors team dynamics, communication patterns, and behaviors to identify any signs of conflict. This includes both explicit disputes and underlying tensions.
Facilitation: When conflicts arise, the Scrum Master facilitates open discussions among team members, allowing everyone to express their perspectives and concerns. They create a safe space for communication and ensure that all voices are heard.
Mediation: In situations where conflicts escalate, the Scrum Master acts as a neutral mediator, helping team members find common ground and guiding them toward mutually beneficial solutions.
Coaching and Empowerment: The Scrum Master coaches team members in conflict resolution techniques and helps them develop effective communication skills. They empower the team to address conflicts independently.
Conflict Prevention: The Scrum Master proactively works to prevent conflicts by encouraging transparency, fostering a culture of respect, and addressing issues before they escalate.

Impediment Removal:

Identifying Impediments: The Scrum Master identifies obstacles, bottlenecks, and impediments that hinder the team's progress. These can be related to processes, tools, communication, or external factors.
Prioritization: The Scrum Master helps the team prioritize impediments based on their impact on project progress and sprint goals.