Department of Computer Engineering

Academic Term: First Term 2023-24

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

Practical No:	**1**
Title:	**Software Requirement Specification**
Date of Performance:	
Roll No:	
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:

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.

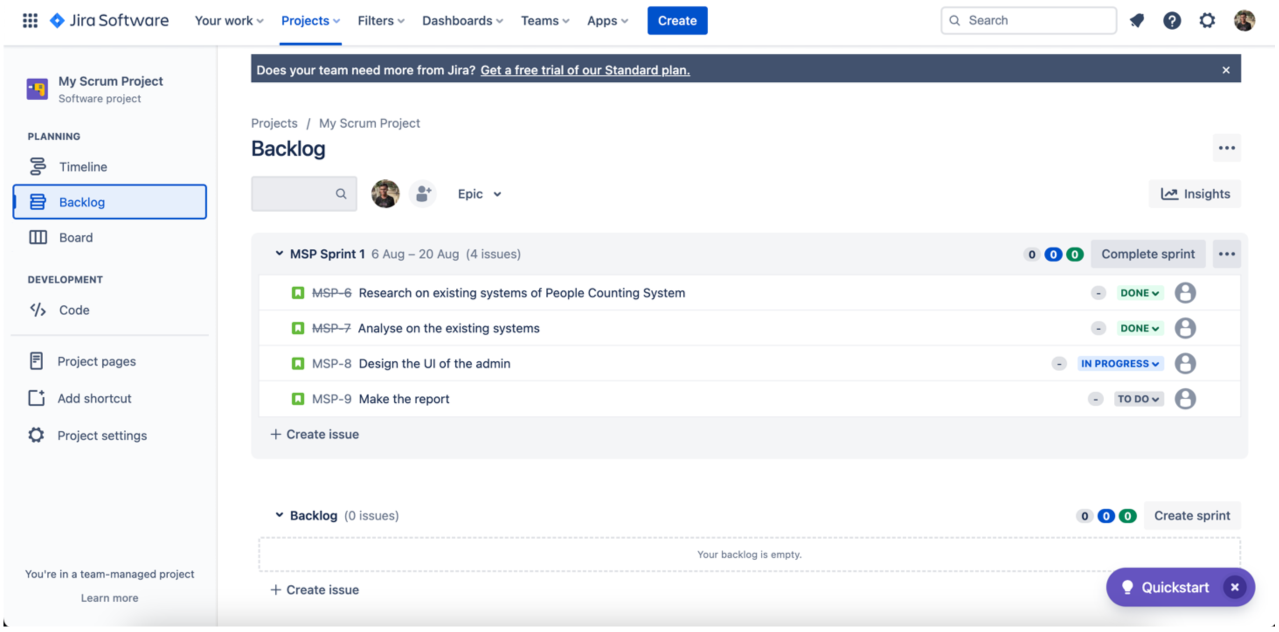
- Gain practical experience in using the JIRA tool for project management in a Scrum environment.
- Learn to create and manage Product Backlogs, Sprint Backlogs, and track progress using JIRA.
- Develop collaborative skills through Daily Standup meetings and Sprint Reviews.
- Gain insights into the iterative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



Jira Software   Your work   Projects   Filters   Dashboards   Teams   Apps   Create   Search

Does your team need more from Jira? Get a free trial of our Standard plan.

My Scrum Project
Software project

PLANNING
Timeline
Backlog
Board

DEVELOPMENT
Code

Project pages
Add shortcut
Project settings

You're in a team-managed project
Learn more

Projects / My Scrum Project

Backlog

Epic   Insights

MSP Sprint 1 6 Aug – 20 Aug (4 issues)   0 0 0   Complete sprint

~~MSP-6~~	Research on existing systems of People Counting System	-	DONE
~~MSP-7~~	Analyse on the existing systems	-	DONE
MSP-8	Design the UI of the admin	-	IN PROGRESS
MSP-9	Make the report	-	TO DO

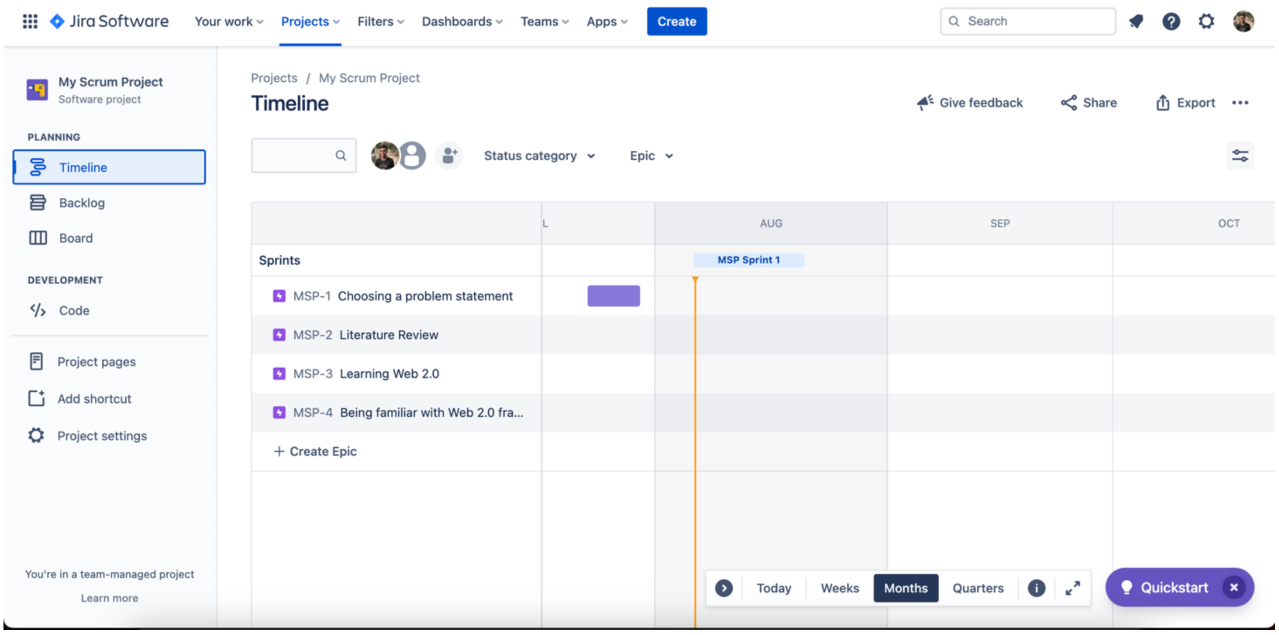
+ Create issue

Backlog (0 issues)   0 0 0   Create sprint

Your backlog is empty.

+ Create issue

Quickstart



Jira Software   Your work   Projects   Filters   Dashboards   Teams   Apps   Create   Search

My Scrum Project
Software project

PLANNING
Timeline
Backlog
Board

DEVELOPMENT
Code

Project pages
Add shortcut
Project settings

You're in a team-managed project
Learn more

Projects / My Scrum Project

Timeline

Give feedback   Share   Export

Status category   Epic

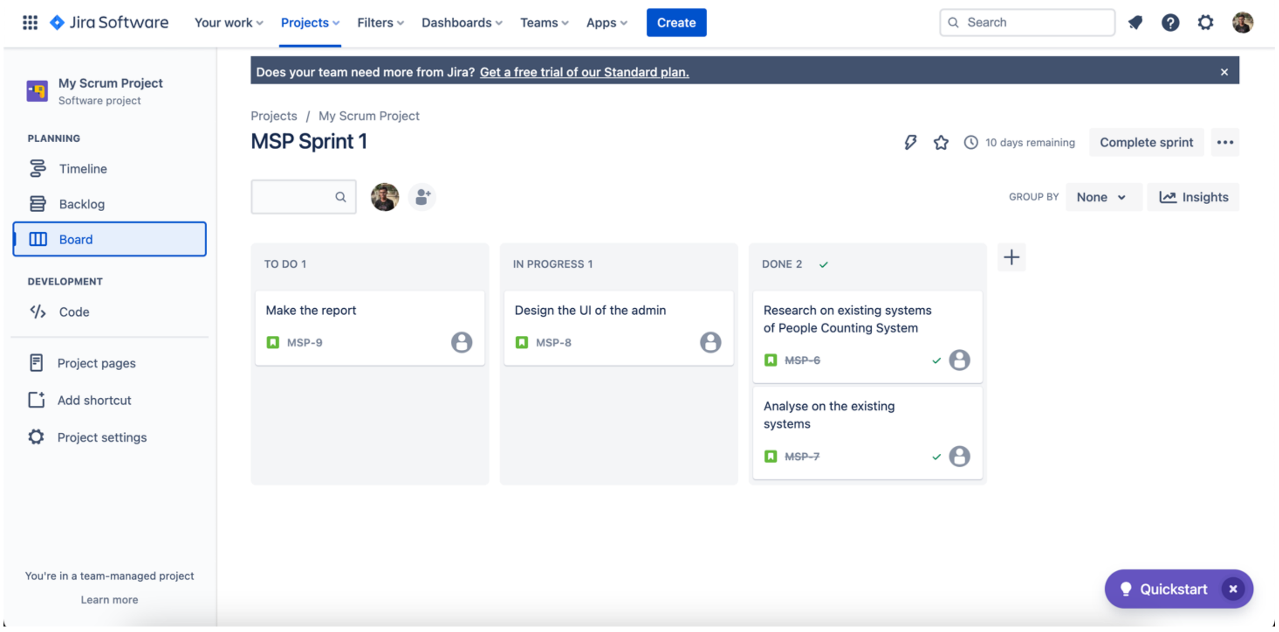
L   AUG   SEP   OCT

Sprints   MSP Sprint 1

MSP-1 Choosing a problem statement
MSP-2 Literature Review
MSP-3 Learning Web 2.0
MSP-4 Being familiar with Web 2.0 fra...
+ Create Epic

Today   Weeks   Months   Quarters

Quickstart



Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / My Scrum Project

MSP Sprint 1

10 days remaining · Complete sprint

GROUP BY None · Insights

TO DO 1
- Make the report — MSP-9

IN PROGRESS 1
- Design the UI of the admin — MSP-8

DONE 2
- Research on existing systems of People Counting System — ~~MSP-6~~ ✓
- Analyse on the existing systems — ~~MSP-7~~ ✓

Post-lab Experiment 02

a) **Assessing the effectiveness of the Scrum framework for managing software development projects compared to traditional project management methodologies requires considering various factors. Let's examine some key aspects to compare the two approaches:**

1. Flexibility and Adaptability:
- Scrum: Scrum is known for its flexibility and adaptability. It embraces changing requirements and encourages iterative development. The regular sprint reviews and retrospectives allow teams to continuously improve the product and process based on feedback.
- Traditional: Traditional project management methodologies, like Waterfall, follow a sequential approach with fixed requirements. Changes during development can be challenging and may require significant effort and cost.

2. Customer Collaboration:
- Scrum: Scrum emphasizes close collaboration with customers and stakeholders. Frequent demonstrations of working software during sprint reviews allow for early and continuous feedback, ensuring the product meets customer expectations.
- Traditional: Traditional methodologies often involve limited customer involvement during development, leading to potential misalignment with customer needs.

3. Time-to-Market:
- Scrum: Scrum's iterative approach and time-boxed sprints allow for frequent releases of functional software. This can lead to quicker time-to-market and faster delivery of value to customers.
- Traditional: Traditional methodologies may take longer to deliver a complete product, as development and testing are typically done in sequential phases.

4. Risk Management:
- Scrum: Scrum promotes early risk identification and mitigation through incremental development and regular retrospectives. Risks are addressed continuously, reducing the impact of potential issues.
- Traditional: Traditional methodologies may have limited risk management during the development process, leading to higher risks during later stages.

5. Team Collaboration and Empowerment:
- Scrum: Scrum fosters self-organizing and cross-functional teams. Team members have a high level of ownership and responsibility for their work, leading to increased collaboration and motivation.
- Traditional: Traditional methodologies may have a more hierarchical structure, where decisions are made by managers, potentially leading to reduced team empowerment.

6. Documentation and Planning:
- Scrum: Scrum focuses on delivering working software over comprehensive documentation. While essential information is captured in artifacts like the Product Backlog and Sprint Backlog, the emphasis is on communication rather than excessive documentation.
- Traditional: Traditional methodologies often require detailed documentation throughout the project, which can be time-consuming and may not always reflect the evolving requirements accurately.

7. Project Visibility and Control:
   - Scrum: Scrum provides excellent project visibility through various ceremonies (e.g., Daily Stand-ups, Sprint Reviews, Sprint Retrospectives). It allows stakeholders to monitor progress continuously and make informed decisions.
   - Traditional: Traditional methodologies may have less frequent project visibility, as progress is assessed at specific milestones.

In conclusion, the Scrum framework can be highly effective for managing software development projects, particularly in dynamic and complex environments where requirements are likely to change. Its iterative and customer-centric approach, coupled with strong team collaboration, promotes faster development, higher customer satisfaction, and better risk management. However, traditional project management methodologies can still be suitable for certain projects with stable and well-defined requirements, where predictability and extensive documentation are critical. The choice between Scrum and traditional methods depends on the specific project's characteristics, the organization's culture, and the preferences of stakeholders involved. Many organizations adopt a hybrid approach, tailoring methodologies to fit their unique needs and project contexts.

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.
An example of a Sprint Backlog in JIRA for a team working on developing a web application. Assume the team is using Scrum to manage their development process.

Sprint Backlog in JIRA:
1. User Story: As a user, I want to be able to sign up for an account on the website.
   - Tasks:
     - Design the sign-up form UI.
     - Implement back-end logic for user registration.
     - Write unit tests for user registration.
     - Conduct user acceptance testing.
   - Status: In Progress

2. User Story: As a user, I want to be able to log in to my account on the website.
   - Tasks:
     - Design the login form UI.
     - Implement back-end logic for user login.
     - Write unit tests for user login.
     - Conduct user acceptance testing.
   - Status: In Progress

3. User Story: As a user, I want to see a personalized dashboard after logging in.
   - Tasks:
     - Design the dashboard UI.
     - Implement back-end logic for fetching user data.
     - Integrate user data with the dashboard UI.
     - Write unit tests for dashboard functionality.
     - Conduct user acceptance testing.

  - Status: To Do

4. User Story: As a user, I want to be able to update my profile information.
  - Tasks:
    - Design the profile update form UI.
    - Implement back-end logic for updating user profile.
    - Write unit tests for profile update functionality.
    - Conduct user acceptance testing.
  - Status: To Do

Potential Bottlenecks and Issues:

1. Blocked Task: The "Implement back-end logic for user login" task (from User Story 2) is blocked because it requires integration with an external authentication service, and the API documentation for that service is not yet available.

2. Scope Creep: During the sprint, a new user story is added to implement two-factor authentication for account security. This addition was not planned during sprint planning and could impact the team's capacity.

3. Lack of Task Breakdown: The task "Design the dashboard UI" (from User Story 3) is too broad and needs to be broken down into smaller tasks, such as designing individual components.

4. Communication Issue: The team faces communication challenges between the front-end and back-end developers, causing delays in the completion of tasks that require collaboration.

To address these issues:
- The Scrum Master can work with the team to resolve the blocked task by reaching out to the external service provider or seeking an alternative solution.
- The team should evaluate the new user story's impact and discuss its inclusion with the Product Owner to ensure it aligns with the sprint goals.
- The team can break down the "Design the dashboard UI" task into smaller components and assign them to the respective team members.
- The Scrum Master can facilitate better communication channels and ensure that team members collaborate effectively.

By addressing these potential bottlenecks and issues, the team can improve their progress during the sprint and deliver high-quality results.

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

Ans.

The role of the Scrum Master in handling conflicts within the development team and resolving impediments is crucial to maintaining a smooth project flow in Scrum. The Scrum Master serves as a facilitator and servant leader, focusing on promoting collaboration, continuous improvement, and the overall success of the team.

1. Facilitating Communication: The Scrum Master ensures effective communication among team members. By actively listening to concerns and facilitating open discussions, they create a safe environment for team members to express their viewpoints and resolve conflicts.

2. Identifying Conflicts Early: The Scrum Master is attentive to signs of conflicts within the team, whether they are related to work processes, technical issues, or personal interactions. By identifying conflicts early, the Scrum Master can address them before they escalate and impact the team's productivity.

3. Mediation and Conflict Resolution: When conflicts arise, the Scrum Master acts as a mediator to facilitate discussions between involved parties. They help team members understand each other's perspectives, find common ground, and work towards a resolution that benefits the team and the project.

4. Promoting Collaboration: The Scrum Master encourages a collaborative culture within the team, fostering an atmosphere where individuals work together towards shared goals. This minimizes potential conflicts arising from individual silos or misunderstandings.

5. Removing Impediments: The Scrum Master proactively identifies impediments that hinder the team's progress and removes them promptly. Impediments can range from resource constraints, external dependencies, technical issues, or conflicts. By addressing impediments, the Scrum Master helps maintain the team's productivity and project momentum.

6. Coaching and Mentoring: The Scrum Master provides coaching and mentoring to team members, helping them develop better communication and conflict resolution skills. This empowers team members to address minor conflicts independently and promotes self-organizing behaviour.

7. Supporting Continuous Improvement: The Scrum Master facilitates retrospectives, where the team reflects on the sprint and identifies areas for improvement, including how to handle conflicts more effectively. They assist the team in implementing actionable changes to enhance collaboration and prevent future conflicts.

8. Working with Stakeholders: The Scrum Master collaborates with stakeholders, including Product Owners and management, to address external impediments that may affect the team's performance. This can include negotiating scope changes, managing expectations, and advocating for the team's needs.

9. Promoting Emotional Intelligence: The Scrum Master encourages emotional intelligence among team members, helping them manage emotions and respond constructively during challenging situations. This contributes to a positive team dynamic and reduces the likelihood of conflicts.

Overall, the Scrum Master plays a pivotal role in fostering a harmonious and productive environment within the development team. By proactively addressing conflicts and impediments, they ensure that the team can focus on delivering value and meeting project goals, ultimately leading to a smooth project flow and successful outcomes.