# **Milestone 1 Scrum Report**

All students are expected to attend the scrum meetings and to participate. Failure to do so will result in greatly reduced grades.

**GROUP**: \_D\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Gurmehak Kaur Uppal | 4. Ajaypartap Singh Maan |
| 2. Arshnoor Kaur |  |
| 3. Manas Gandotra |  |

**Milestone 1 Tasks**

In this phase of the project you will:

* Setup teams of about 3-5 developers (6 is too large)
* Write and sign a team contract
* Create a GIT account
* Create a Jira account
* Add your professor to the GIT and Jira accounts
* Update Jira with the work performed and planned

**Deliverables due 4 days after your lab day:**

* Completed team contract.
* Fully initialized Git repository. **Be sure to send your professor the link to your GitHub repository and a screenshot of the GitHub users.**
* Fully setup Jira project. **Be sure to send your professor the link to your Jira Project.**
* Completed scrum report including reflection questions answered.

**Rubric**

|  |  |  |
| --- | --- | --- |
| **Individual** | Group participation | 80% |
| Teamwork | 20% |
| **Group** | Contract | 25% |
| Git repository | 25% |
| Jira project | 25% |
| Scrum report & reflections | 25% |
| **Deadline** | 20% deduction for each day you are late |  |
| **NOTE** | Both the individual and group marks are calculated separately. Each member of the group will have their mark calculated based on their contribution to the group work and their contributions to the team. The group participation is a percentage that your professor feels you contributed to the group work. This is multiplied by the weight of the group participation component to determine your grade. |  |

**Scrum Report**

**Summary of Tasks Completed or Delayed in the last week:**

Here you can list all of the tasks completed in the last week along with any tasks which could not be completed with a reason why they could not be completed.

|  |  |  |
| --- | --- | --- |
| **Member** | **Tasks Completed** | **Tasks Delayed/Blocked** |
| **Ajaypartap Singh Maan** | **Reflection** | **No** |
| **Arshnoor Kaur** | **Reflection** | **No** |
| **Gurmehak Kaur Uppal** | **Group Contract and Scrum Report** | **No** |
| **Manas Gandotra** | **Set up Jira and Github** | **No** |

**Summary of Meeting:**

A summary of the main points discusses in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Group Contract | Decided upon penalties for being late/not completing work or submitting sub- standard work. The meeting would be conducted once every week. Also, the individual component should be completed 2 days before the due date and the project should be submitted one day prior. | All group members agreed to the terms in the contract. |
| GitHub/Jira | Everyone should join Jira and GitHub and appropriate repositories to be created. | Jira and GitHub set-up was successful |

**Summary of Decisions Made:**

This will include major architecture and design decisions, testing decisions, prioritization of tasks, dealing with problems encountered and other major outcomes from the meeting.

|  |  |
| --- | --- |
| Decision | Rationale |
| Every individual deliverable should be complete 2 days prior to deadline. | In case the group member encounters any difficulty to complete their work, there is a time window left for other members to help. |
| Any possible delays in submission or inability to join meeting should be reported in advance. | There should always be an effective communication between group members. |

**Tasks Attempted During Meeting:**

Each member is assumed to participate in the scrum meeting and contribute to the completion of the scrum report and reflections. Since the scrum meeting will not take more than 20-30 minutes, there is lots of time left to undertake some of the actual work tasks. In the table below, each member should list what they did to complete the scrum report, the reflections, and 1-4 other tasks they completed during the class period. If a task cannot be completed, the student should indicate why this was not possible.

|  |  |  |  |
| --- | --- | --- | --- |
| Member | Task Attempted | Time Spent | Complete? |
| Gurmehak Kaur Uppal | **Understand the project thoroughly by reading the pdf.** | **40 mins** | **Yes** |
| Arshnoor Kaur | **Understand the project thoroughly by reading the pdf.** | **40 mins** | **Yes** |
| Ajaypartap Singh Maan | **Understand the project thoroughly by reading the pdf.** | **40 mins** | **Yes** |
| Manas Gandotra | **Understand the project thoroughly by reading the pdf.** | **40 mins** | **Yes** |

**Scrum Tasks Selected for Next Week**:

The tasks each member has selected to pursue for this class or the next week.

|  |  |
| --- | --- |
| Group Member | Task Description |
| Gurmehak Kaur Uppal | Scrum Report |
| Arshnoor Kaur | Analysis of problem and header file |
| Ajaypartap Singh Maan | Reflection |
| Manas Gandotra | Test Plan |

**Major Outcomes of Meeting:**

This is where you should highlight the major accomplishments of the class.

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Completed Milestone 1 work | **Successful submission of milestone 1 deliverables.** |
| Assigned work for Milestone 2 | **Milestone 2 would be completed in time.** |
| Understood the project requirements in detail | **Everyone knows what the project requires.** |

**Things That Went Well in This Meeting:**

Here you can highlight things which worked well. This indicates that the way you worked on these items is working and should be continued.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Success |
| Contract | **Everyone agreed on the terms in the contract without any conflict.** |
| GitHub and Jira | **No delays from anyone in joining the team on GitHub and Jira.** |
| Punctuality/Professionalism | **Everyone was punctual with the meeting time and demonstrated adequate level of professionalism throughout the discussion.** |

**Things That Did NOT go Well in This Meeting:**

This is where you can list things which did not go well in the class. You should analyze why this happened and suggest how you can improve it next time. This will lead to the goal of *continuous process improvement*.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Problem and How to do Better |
|  |  |
| Everything went well, as far as Milestone 1 is concerned. | |
|  |  |

**Reflections (to be answered by the group)**:

Answer the following questions using your own words. Make sure that each answer comprises a minimum of 100 words.

1. GIT is an example of a version control system. List and explain 3 benefits of using a version control system.

A: The 3 major benefits of using GIT are:

1. Version History: Version control in GIT comes in very handy in providing a detailed history of every change made to files over time. Every commit or change gets logged into the system with a message describing what changes are made and which line of code is added, removed or modified. This feature makes it easy for programmers to quickly check old versions of code and compare it with present code. It is also beneficial if the commit made is faulty and we need to switch back to the old version of the code.
2. Collaboration Features: GIT offers efficient collaboration that makes working with teams much easier. Multiple developers can work together on the same repository, by using features like branching, which allows them to work on the code without making permanent changes on the code. When the code is verified, they can merge the new code main code branch. This reduces conflicts and enhances productivity.
3. Decentralised System: GIT is an example of a distributed/decentralised version control system, which means that each developer can clone the code offline, work on it and after all the changes, they can sync those changes online using a pull request. This helps in reducing dependence on a central server and prevents data corruption during a network failure.
4. What is a version control system? Why does GitHub qualify as a version control system?  
   A: A version control system (VCS) is a tool in software that manages the source code along with a detailed history of changes made to the code, facilitates collaboration and provides a way to merge code changes. The primary purposes of VSC include- version tracking, Backup, Merging, and Branching features.

The reasons why GitHub fits the definition of VSC are:

1. GIT Integration: GitHub qualifies as a version control system because it is built upon GIT, which is a decentralised version control system and enhances features of version control capabilities like by tracking changes, supporting collaboration and managing complex workflows. By hosting a GIT repository, GitHub uses all the features of a version control system such as committing histories, branching, merging and reversing changes.
2. Collaboration: GitHub comes with collaboration features, so that multiple users can collaborate by cloning, and forking the repository and can work on the project offline. Features like pull requests can make the changes made on their device online and by doing so they get peer review before the integration process. They can also add comments on specific lines of code
3. Repository Hosting and Backup: GitHub allows developers to host their GIT projects online, making them accessible to everyone. Decentralised hosting of projects means they can get backed up and accessible even if the local copies get corrupted along with the history of all the changes made to the code.

1. What is Jira? How are we going to use Jira for this project?  
   **ANS**: JIRA is the best agile project management tool which is utilized to manage various tracking-related tasks. JIRA helps the developers in planning the developments made in the software efficiently from time to time. Being a software tool developed by Atlassian, it keeps a record of bugs faced while working on a project and provides extensive customization options through various developmental tools. JIRA helps streamline the project execution and results in a successful group project.

We have to use Jira to organize the workflow of the team and track issues in the milestones. Not only the JIRA support agile project management tools, example Kanban which helps to organize the tasks visually for the whole team. Firstly, the JIRA account set up by each member is followed by selecting the KANBAN board. The issues in the project will be recorded creating an issue on the Kanban board in Jira whenever a team member needs to write a function, test the code, or in order to fix a bug in the project. Also, the issues which have been created in the last step have to be added to the ‘to do’ column of the Kanban board. The files need to be moved across the Kanban board as the work which is assigned to each team member is in progress. The tag assigned to each issue will cater to the milestone to which it pertains. So, overall the issues should have a title, a short description of what needs to be done in the issue, and the connection that a particular issue has with others. In JIRA, the team members will organize their work and communicate with each other through collaborative work on JIRA as each of the members can see the same project details, issues, and associated boards such as Kanban.

1. Why is a Kanban board useful in software development? What are the advantages of using Kanban board?

**ANS:** As far as the Software Development field is considered, there are so many challenges that every software developers team has to face daily, be it on the project implementation methodologies or the continuous collaboration with the team members. They collectively need a tool where they can visualize the flow of tasks through various developmental stages, such as planning, designing, development, testing, deployment, and many more. Kanban comes into play at this point. Here are some of the advantages of using the Kanban board:

**1)**  Collaborative Work: First and foremost, when a team is using the Kanban board, not only is the teamwork and team participation maintained, but also the team members save time and manage the project efficiently. The visual representation gives a detailed analysis of the work being progressed and thus, team members can have an understanding of the group project. Not only this, the teams can collaborate virtually with each other, being at any place of the world.

**2)** Transparent Visualization: Kanban provides a wide view of tasks taking place currently, for example, the current status of the work (In Progress, Completed, To Do). With this, the team members can not only manage their tasks independently but also track other snippets of the project taking place by other team members.

**3)** Team Learning: In Kanban, the teams learn a lot from mistakes both made by them and made by other team members. This enforces quick learning as the project keeps on getting refined as the team moves collectively in the project from one milestone to another. Also, nobody faces an issue when the status of the work is updated, or if updations are made in any section of the project. This is because the Kanban board is adaptable to the changes made by each team member without disturbing the continuing project work.