OUR GITHUB REPO
OUR JIRA
LUCID CHART DIAGRAMS

SCRUM FLOWCHART CYCLE

1. **Product Backlog**
   a. Organization of project requirements and priorities
2. **Sprint Planning**
   a. Set clear goals and tasks for each upcoming sprint
3. **Sprint Execution**
   a. Implement tasks
4. **Daily Standups**
   a. Share progress, address challenges, etc. (in-class)
5. **Sprint Review**
   a. Review completed work and gather feedback
6. **Retrospective**
   a. Analyze sprint performance and plan for improvements / adjustments
7. **Back to step 1**

USER STORIES SHOULD TAKE ≤ 2 DAYS
TASKS SHOULD TAKE ≤ 2 HOURS

OUR TASK STORY POINT SYSTEM
1 point — ≤ 30min
2 points — ≤ 1 hour
3 points — ≤ 1.5 hours
5 points — ≤ 2 hours

MONTHLY FEEDBACK GRADE

Every month (every **project checkpoint**), we will receive a feedback grade on our Jira & Github project using the following metrics:

Each team member is graded **individually** based on two main areas:

1. AREA ONE: Completed Tasks

A task counts as **completed** if it meets all of the following criteria:

- 📝 **Properly** named, described, assigned to you, with story points & due date.

- 💻 **Includes at least one effective commit**:
    - Must add value to the project (adds new code, not just changes/removals).
    - Commit must be visible in **Jira**.
- 🌿 Task must include a **GitHub branch**, **code review**, and **successful merge** (all traceable in Jira).

Grading scale for completed story points (relative to team velocity):

- 🟢 **100%** → Within **25%** of the average story points completed by each team member.
- 🟡 **80%** → Within **50%** of the average story points completed by each team member.
- 🔴 **60%** → Within **75%** of the average story points completed by each team member.

2. AREA TWO: Code Reviews Performed for Teammates

- 📑 Must perform code reviews for **other team members** (visible on GitHub).
- You can refer to the textbook for details on how to write a code review.

Grading scale for reviews performed:

- 🟢 **100%** → At least **2 reviews per team member**
- 🟡 **80%** → At least **1 review per team member**
- 🔴 **60%** → **0 reviews per team member**

*See the calendar in Canvas for the Sprint Review dates (I'm assuming each sprint is due the night before each Sprint Review)

FIRST CHECKPOINT: WED OCT 1, 2025 11:59 PM

SECOND CHECKPOINT: OCT 31, 2025 11:59 PM

HOW TO USE JIRA

Some note about Jira:

- **User stories** (green bookmark) should take up to a **couple of days** and **tasks** (blue checkmark) should take up to a **couple of hours** to complete.

- Your product **backlog** should only have **user stories**. A user story comes from a PO with no knowledge of technical implementations or coding solutions.

- A **task** should translate to a **commit**. Doing research is not a task.

- Your tasks should be **descriptive**, anyone reading it should know exactly what needs to be done, and **specific**, there should be a clear way to verify its completion. If you say "set up the backend", what is the backend? If you say "create the database", how many tables? for what data?

- If you fail to complete all user stories assigned to a sprint, simply move them back to the backlog during the sprint retrospective.

- The sum of task story points completed at the end of a sprint represents your **team velocity**. It should be estimated at first, but it should stabilize after multiple sprints.

Populating the Product Backlog

- 👤 **The PO adds user stories**
  The Product Owner is responsible for adding user stories into the backlog. Stories should describe *needs* in plain English, not technical details, algorithms, or solutions.
- ✅ **Follow INVEST principles**
  Each story must be: **Independent, Negotiable, Valuable, Estimable, Small, Testable**.
- 📝 **Use proper story format**
  Write stories as:
  “As a [user role], I can [action].”
- 🗂️ **User story in Jira**
  Stories appear with the Jira story icon **a green bookmark**.
- 🎯 **Prioritize and estimate**
  In Jira, assign **priority** and **story points** (use *Planning Poker* for estimates).

HOW TO DO SPRINT PLANNING

During sprint planning, you should:

- 📊 **Determine team velocity**
  Estimate how many story points your team can complete in a sprint. This will be rough at first but gets more accurate after a few sprints.
- 🎯 **Select sprint stories**

Choose stories from the backlog equal to your team's velocity, based on the sprint goal.

- 📝 **Create the sprint in Jira**
  Push the **"Create Sprint"** button and move selected stories from the backlog to the sprint.
- 🔨 **Break down stories into tasks**
  Each task should be small, ideally taking only a few hours to complete.
- 🎴 **Estimate task effort**
  Assign story points to tasks (e.g., using *Planning Poker*).
- 👥 **Assign tasks to team members**
  Each team member sets a due date for their assigned tasks.
- 🔗 **Link tasks to stories**
  For each story: press **+** → **Link items** → **"is blocked by"**, and link the related tasks.
- ▶️ **Start the sprint**
  Press **"Start Sprint"** to begin work. This will take you to the Scrum board.

PROCESS FOR COMPLETING TASKS

1. 📋 **Pick your task**
   Go to Jira's **Scrum Board** and move the task you plan to work on to **In Progress**.
2. 🌿 **Create a branch**
   Click on the task, scroll down, and select **Create Branch**.
   This will automatically create a branch on **GitHub**.
3. 💻 **Switch to your IDE**
   Open your preferred IDE (e.g., **VS Code**, **GitHub Codespaces**).
   Make sure to switch to the newly created branch.
4. ⚙️ **Work on your task**
   Implement the changes and complete all tests.
5. ✅ **Move to Tested**
   On Jira's Scrum Board, move the task to **Tested** after completing all tests.
6. 💾 **Commit & Push**
   Commit and push your work.
   Include the **Jira commit code** in the commit message.
   (Find this code in Jira by clicking the task → scroll down → Commit Message.)
7. 🔀 **Create a Pull Request**
   In Jira, select the task, scroll down, and click **Create Pull Request**.
   This will take you to GitHub. Assign a **team member** to review your code.
8. 👀 **Code Review**
   After the code review is complete, move the task to **Reviewed** on Jira's Scrum

Board.

9. 🔗 **Merge Pull Request**
   On GitHub, merge the pull request into **main**.
   Resolve any conflicts if necessary.
10. 🏁 **Complete the task**
    On Jira's Scrum Board, move the task to **Done**.
    On GitHub, **delete the branch**.

PROJECT TIMELINE

OCTOBER
Sprint 2: Diagrams (due Sunday, Sep 28)
Sprint 3:

NOVEMBER
(Project must be completed in November because project presentations are first week of December)