Sprint 4: Refactoring and Final Touches

Sprint 4 Objectives

1) Project in GitHub (0 pts)
2) Features with tests (10 pts)
3) 80% Test coverage metrics (15 pts)
4) CI Pipeline
   (a) AI Code Review using OpenAI Platform for each Pull Request (2.5 pts)
   (b) Running automated tests for each commit (2.5 pts)
   (c) Reports test coverage metrics in console for each commit (2.5 pts)
   (d) PyLint and/or Flake 8: Linting scans in CI pipeline with reports (10 pts)
   (e) Dependabot: Dependency vulnerability scans in CI pipeline with reports (10 pts)
5) Deployed to a production environment
   (a) Custom domain name (no subdomains unless you own the top-level domain) (5 pts)
6) CD Pipeline that runs when code is merged into `master/main`
   (a) Continually deployed to production environment using automated actions (2.5 pts)
7) **NEW:** 100% of vulnerabilities and linter findings are addressed and fixed (20 pts)
8) **NEW:** 5-minute marketing video uploaded to YouTube showcasing the application (20 pts)
   (a) Loom offers free screen-recording options

Demo Expectations

1) Share the URL to your production deployment so we can follow along.
2) Tell us about your application and why you're making it.
3) Go through Zenhub / GitHub Projects and show us what work you completed based on user story.
4) Demo the work you completed on the production environment showing the feature in production and the CD pipeline and test cases used to test each requirement.
5) Get customer feedback.

Sprint Duration

Each sprint is two weeks long. Make sure that you work through both weeks of the sprint to achieve your tasks!

Common Sprint Requirements

Backlog Grooming and Prioritization

At the beginning of the sprint, meet with your group to groom the backlog and prioritize work. Mark stories that are MVPs (minimum viable product).

Planning and Story Division

Begin the sprint by reviewing your backlog. Add or divide stories as needed—this may include playing Planning Poker to determine story points. Record all changes in Zenhub / GitHub. Meet with your customer to identify the stories they want you to implement. Prioritize and confirm with them. Select as many stories as you hope to complete in the sprint. Balance workload between sprints; each person should have at least one story. At least two larger stories should be implemented and fully tested each sprint. Move the selected tasks from *Backlog* to *Sprint 1* (or the current sprint). Each group member should take ownership of **at least** one task. Keep the *FIRST* and *SMART* principles in mind. MVPs should be prioritized.

Testing and Development Process

All stories selected for a sprint should be assigned to a main person and include unit/integration tests. Follow the *Red-Red-Green-Green-Refactor* process (refactoring will be evaluated at the end of the semester). You are responsible for full-stack development (front end and back end) and associated integration and unit testing. Use branches and communicate clearly. Push, pull, and merge to `main` frequently. Branch protections are in place; approval is required for merges. **Work early** so your branch is submitted in time. As you finish tasks, move them along to completion in your project-management tool.

Submission

On Canvas (one team member)

- Submit your GitHub repository URL.
- Submit your application's production URL.
- Submit a short report of what was implemented and tested, noting any incomplete items and plans.
- Explain any changes to your plan since the previous submission.
- Document issues with tests, if any.
- Record your velocity (as reported in Zenhub) at the end of the document.

On GitHub

- Ensure all project files, tests, and code are committed and pushed to the `main` branch.
- All students must push to GitHub with correctly configured usernames.
- Tag the repository revision with `sprint<i>` where `<i>` is the sprint number.

```
git tag -a sprint<i>
git push --tags
```

- Verify that tags are available on GitHub before submission.

On Zenhub/GitHub Projects

Show your point assignments, assignees, and progress as described.

Discord

One team member must post to your `Dev_Cust` group detailing what was implemented and your app URL. Customers can respond there.

AI Documentation

Document in your project `README` where and when you received AI assistance, including transcript URLs for each instance.

Kritik Evaluations

Complete your Kritik evaluations. The first opens immediately after the sprint deadline; each activity has three stages. Check the schedule for deadlines.