Development Plan
Flick Picker

Team 7, 7eam
Talha Asif - asift
Jarrod Colwell - colwellj
Madhi Nagarajan - nagarajm
Andrew Carvalino - carvalia
Ali Tabar - sahraeia

Table 1: Revision History

Date	**Developer(s)**	**Change**
Sept 21/22	Talha	Updating Workflow Plan
Sept 24/22	Talha	Updating Sections 1-3

Contents

This documentation details the entire development plan, from team meetings, to the workflow, all the way to the technology Flick Picker will use. 7eam fully understand their responsibilities and knows the flow our application will follow.

1 Team Meeting Plan

There will be an ad hoc meeting nearly weekly to ensure the team is on the same page, where a time is picked that works for all five team members. The reason for the meeting will also be clearly stated before meeting, and we will start as soon as everyone is in the call.

2 Team Communication Plan

7eam will utilize Discord as their main form of communication, a server has been created with all forms of communication in it currently. It is expected if something urgently needs a specific team member, they will be responsive within 24 hours.

3 Team Member Roles

Everyone shares responsibilities and is required to be flexible when needed, but there are areas of development each individual specializes in to keep work split evenly.

Team Developer Roles

Team Member	**Role**
Talha	DevOps, Full-Stack Developer
Jarrod	Back-End Developer
Madhi	Back-End Developer
Andrew	Front-End Developer
Ali	Front-End Developer

4 Workflow Plan

Git Workflow:

- *develop* branch will be the single source of truth, where the team reviews changes before they get merged. Thus the *develop* branch will have restricted permissions on it, preventing direct merges without admin overwrite, and only one developer will be the admin, Talha
- Any changes have to be on their branch, and a PR has to be cut with a full green checklist to get it merged into *develop*

- The checklist will grow as development on the application continues, currently, the checklist is a single item, where the PR must have two approvals from the team. The future checklist items are as planned:
  - Entire test run has to be successful to ensure *develop* is in a healthy state
  - Test coverage delta must not be reduced unless redundant tests are taken out
  - Spotbugs must pass, enforcing healthy code practices
  - Snyk checks must pass, ensuring the packages used do not have vulnerabilities
- Each PR must have at minimum a description filled out, and a relevant feature ticket must have a Jira issue attached along with it
- If a feature has an attached ticket, the PR title must start with [CAP-##] and then a title
- Every time a PR is made, automation will ping #pr-bot on Discord so the team is aware of changes being made

Issue Tracking - Jira:

- All development changes need a descriptive ticket cut before it is ready for code review. Automation will link the PR to the ticket and vice versa as well
  - Descriptive means the ticket must have a title and acceptance criteria (AC) to complete the ticket
- The status of the ticket must be updated on the board. Most importantly, if it is in "To Do" so multiple developers do not start working on the same feature
- Points will be arbitrary for the ticket, based on how much work the developer is working on it thinks it will be. It is going to be an indication of how complex the work is for the reviewers
- Utilize Jira's ticket types to have issue classifications:
  - Story: Ticket describing a new feature
  - Bug: Fixing existing code
  - Task: Small changes that do not fall in either of the above categories

4.1 Example Feature Workflow

An example developer workflow for a feature will be as follows:

1. Jira ticket is cut with a description and AC, assigned to a developer, then the ticket is moved to "In Progress" while it is developed
2. Branch is made for development and then marked "Ready for Review" when the owner thinks the AC is met
3. Reviewers ensure AC is met and healthy coding practices are followed
4. PR is merged, and the ticket is moved to "Done"

4.2 Example Documentation Workflow

An example documentation workflow will be as follows:

1. Asynchronous discussion on Discord is done to choose which sections of a documentation to update
2. Branch is made for updates and then marked "Ready for Review" when the owner thinks the documentation has been sufficiently written
3. PR is merged

5 Proof of Concept Demonstration Plan

What is the main risk, or risks, for the success of your project? What will you demonstrate during your proof of concept demonstration to convince yourself that you will be able to overcome this risk?

6 Technology

- Specific programming language
- Specific linter tool (if appropriate)
- Specific unit testing framework
- Investigation of code coverage measuring tools
- Specific plans for Continuous Integration (CI), or an explanation that CI is not being done
- Specific performance measuring tools (like Valgrind), if appropriate
- Libraries you will likely be using?
- Tools you will likely be using?

7 Coding Standard

8 Project Scheduling

[How will the project be scheduled? —SS]