Development Plan
ProgName

Team #, Team Name
Student 1 name
Student 2 name
Student 3 name
Student 4 name

Table 1: Revision History

Date	Developer(s)	Change
Date1	Name(s)	Description of changes
Date2	Name(s)	Description of changes
...	...	...

[Put your introductory blurb here. —SS]

1 Team Meeting Plan

Team Meeting objective(s):

- Discuss and delegate tasks for upcoming deliverable(s)
- Group work sessions on upcoming deliverable(s)

Team meetings are schedulied throughout the week, at the start of the week, via a discord poll. Each team member is expected to respond to each poll. If a team member confirms that they can attend, then they are expected to attend.

2 Team Communication Plan

All team communication is done through a WhatsApp group chat and a discord server. The discord has three text channels:

- General (anything not project-related is posted here)
- Important Updates (anything related to weekly meeting availability goes here)

- Locked In (anything related to project work goes here)

The discord also has 2 voice channels:

- Weekly Meeting (administrative meetings happen here)
- Locked In (collaboration of project development happens here)

All other communications happen through the WhatsApp group chat

3 Team Member Roles

- Ahmed: Meeting manager + scheduler, developer
- Ansel: Team liaison, developer
- Matthew: Developer
- Mohammed-Hassan: Developer
- Muhammad: Developer

4 Workflow Plan

- Git standards
  - Development branches will follow the "dev/*name*/*description*" naming convention
  - Documentation branches will follow the "docs/*name*/*description*" naming convention
  - A feature branch will be created for every major component of the project
  - Developers will base their branches off the feature branch they are contributing towards
  - Developers will create pull requests that concisely explain what the pull request is doing, with any useful information being written in the PR description
- Issue standards
  - Issues are created during usability testing
  - Issues must contain a comparison between what is expected of the code and what is actually seen by the user
  - If possible, issue creators should attach any log files they collect to the issue
  - The issue must be assigned to the developer who originally pushed the code that is causing the bug

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 Expected Technology

[What programming language or languages do you expect to use? What external libraries? What frameworks? What technologies. Are there major components of the implementation that you expect you will implement, despite the existence of libraries that provide the required functionality. For projects with machine learning, will you use pre-trained models, or be training your own model? —SS]

[The implementation decisions can, and likely will, change over the course of the project. The initial documentation should be written in an abstract way; it should be agnostic of the implementation choices, unless the implementation choices are project constraints. However, recording our initial thoughts on implementation helps understand the challenge level and feasibility of a project. It may also help with early identification of areas where project members will need to augment their training. —SS]

Topics to discuss include the following:

- Specific programming language
- Specific libraries
- Pre-trained models
- 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
- Tools you will likely be using?

7 Coding Standard

8 Project Scheduling

[How will the project be scheduled? —SS]

Appendix — Reflection

[Not required for CAS 741 —SS]

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. Why is it important to create a development plan prior to starting the project?
2. In your opinion, what are the advantages and disadvantages of using CI/CD?
3. What disagreements did your group have in this deliverable, if any, and how did you resolve them?