Development Plan
Software Eng

Team #11, Mac-AR
Student 1 Matthew Collard
Student 2 Sam Gorman
Student 3 Ethan Kannampuzha
Student 4 Kieran Gara

Table 1: Revision History

Date	**Developer(s)**	**Change**
2023/09/15	All	Initial addition of all sections
Date2	Name(s)	Description of changes
...	...	...

[Put your introductory blurb here. —SS]

1 Team Meeting Plan

Our Plan is to have a meeting every capstone lecture that is not Software Eng, or is not running. When we are behind on a deliverable, we will schedule a meeting during our time between classes to meet and go over what we will need to accomplish before the due date.

2 Team Communication Plan

For most issues, we will communicate over Discord with each other. Matthew is in charge of communicating with the profs/supervisor. In addition, we will also have a meeting with Irene Yuan every week during Irene Yuan's Office hours.

3 Team Member Roles

Role	Name	Responsibility
GitHub Administrator	Sam	Merge and maintain Github Branches
Final Revision Editor	Kieran	Last editor of requirements documents, makes sure we adhere to writing guidelines
Communication Director	Matthew	Communicates with the Supervisor/Prof and any stakeholders
Meeting Minute Writer	Ethan	Keeps track and writes down the meeting minutes
Lead Developer	Kieran	Leads the development process, makes sure we are on the right track to complete our goals at the agreed upon due dates
Lead UI Designer	Matthew	Makes sure the user interface is clear to the user, and functioning. Implements UI based code such as buttons
Functional Requirement Lead	Ethan	Makes sure every function requirement is met during the coding process
Non-Functional Requirement Lead	Sam	Makes sure every NFR is met during the coding process

4 Workflow Plan

- For Git, we will have a master branch that will always have a working code base, and up-to date documentation. We will have a second branch called develop, this is the development branch, and all our new changes get merged into this branch. It is meant to be unstable and most of our issues will pop up in this branch. Every time develop has a stable build, we will push the code into master. We will have branches off of develop that we are going to use as our feature branches, every new line of code gets written in these branches, and when the feature is done it will get merged into develop. This double buffer system ensures we always have working code easily accessible in master. Sam will be responsible for merging develop into master.
- Document changes are the only changes that can be merged directly from branch to main. All source code changes go through the development branch first.
- Changes into the dev and main branches will be done through formal pull requests. Reviewers will not be strictly necessary for these PRs, mainly being used as a way to track merges and make sure branches are up to date before merging in. Reviews from other team members are recommended when working on the same file/ directory.
- Issues will be created for each team meeting, lecture period, document revision, and feature change. Issues for meetings will be used to track who attended and a general overview of the agenda. Issues for document and dev work will be broken up by deliverable.

5 Proof of Concept Demonstration Plan

Main Risks:

- Some form of AR working
- Networking with multiple devices

The plan for the proof of concept is to have a simple AR program working that can allow two users to interact in some way. This will be used as a trial run for the technology and the actual game will be built off of the POC afterwards.

6 Technology

- C# (Unity)
- Linter: Microsoft Code Analysis extension in VS
- Test Framework: NUnit

- Code Coverage measuring tool: dotCover
- Specific plans for Continuous Integration (CI), or an explanation that CI is not being done ** Wait until tutorial **
- Specific performance measuring tools (like Valgrind), if appropriate
- Libraries: TBD based on project needs
- Tool/Framework: Unity

7 Coding Standard

- Variable name syntax: camelCase
- Function name syntax: PascalCase
- Class name syntax: PascalCase
- Brackets: Same line (Ex. If () {)
- Public/ Private: Use private when possible
- Indent: Tab

8 Project Scheduling

The team will use a Google Calendar to schedule goals for milestone progress and deliverable deadlines. As a general rule, the next deliverable will be started a minimum of one week before the previous deliverable is due for documentation. Code development will be ongoing throughout with deadlines set for various components.