Development Plan
Software Engineering

Team #13, ARC
Avanish Ahluwalia
Russell Davidson
Rafey Malik
Abdul Zulfiqar

Table 1: Revision History

Date	Developer(s)	Change
2024-09-16	Russell Davidson	General ideas for Sections 4, 5, 6, 7, and 8
2024-09-18	Russell Davidson	Finished section 4
2024-09-21	Zulfiqar Abdul	Expected Technology, Coding Standard
2024-09-21	Russell Davidson	Finished sections 6, and 7
2024-09-23	Zulfiqar Abdul	Proof of Concept Demonstration Plan
2024-09-23	Russell Davidson	Finished sections 3, 5, and 8
2024-09-24	Everyone	Finished Appendix Reflection/Team Charter

1 Confidential Information?

This project will not have any confidential information to protect.

2 IP to Protect

This project will not have any IP to protect.

3 Copyright License

The Apache License 2.0 will be adopted for this project. The license can be found in the repo here.

4 Team Meeting Plan

Every week, there will be two in-person team meetings that can be extended in length if necessary:

- Mondays at 2:30pm-4:20pm
- Wednesdays 4:30pm-5:30pm

Before deliverables are due, additional meetings may be held virtually.

This team has no industry advisor so no meeting times are needed. If one is found in the future, meeting times will be scheduled according to their availability.

All meetings will start with the progress review of all action items from the previous meeting. Then, each item in the agenda section of the GitHub issue for the current meeting will be discussed. Notes will be added to the GitHub issue as topics are being discussed and action items will be made for each task that should be done before the next meeting. Before ending the meeting, a new GitHub issue for the next scheduled meeting will be created with an agenda.

Every meeting will have a chair starting with Rafey and rotating through members every week.

5 Team Communication Plan

Discord will be used for online meetings and asynchronous discussions. All scheduled meetings will be conducted as described in the previous section (4 Team Meeting Plan).

Communication with the TA will be done through MS Teams.

GitHub issues will be created for the following purposes:

1. Lecture attendance
   - Each lecture will have a corresponding GitHub issue where the attendance of team members and relevant questions related to the lecture content is recorded.
2. Team Meeting
   - Each team meeting will have a GitHub issue made in the previous meeting with attendance and a meeting agenda.
3. TA Meeting
   - All meetings with the TA will be recorded in a GitHub issue with both an agenda and questions to ask the TA.
4. Supervisor Meeting
   - If a supervisor for the team is found, meetings with them would be cataloged with this issue type. This will more likely be used for meetings with project stakeholders.
5. Project Tracking (bugs and features/documentation)
   - Project bugs/features/docs are to be tracked using GitHub issues for each desired change.

6 Team Member Roles

The team will have the following roles:

1. **Meeting Chair**
   - Make a Github issue for meeting notes and the agenda for the next meeting.
2. **Reviewer**
   - Look over GitHub pull requests to check changes are acceptable or suggest edits.
3. **Augmented Reality: Subject Matter Expert**
   - Works on, and render assistance to, parts of the project related to augmented reality (AR).
4. **Unity: Subject Matter Expert**
   - Works on, and render assistance to, parts of the project related to the Unity game engine.
5. **Team Liaison** *(Russell)*
   - Monitors email for team updates by course instructors and keeps the rest of the group up-to-date with the latest communications.

The meeting chair and reviewer will rotate on a weekly basis in alphabetical order of last name. Other roles will be assigned to members by October 1, 2024.

7 Workflow Plan

The team will use git for version control and store the project files remotely on GitHub. All branches will have a prefix to signify what type of changes are being made followed by a descriptive name that can be linked to an issue in the *GitHub Project*. When a change is finished, a new pull request will be created from a branch and contain a reference the *GitHub Issue*(s) being addressed by the commits within the branch.

The GitHub repo will have all the properties outlined below:

- **Default branch:** *main*
- **Branch Prefix Types:**
  - *feature/* (New features)
  - *fix/* (Fixing bugs in already existing code)
  - *release/* (Designated release)
  - *docs/* (Changes to documentation)
- **Pull Requests (PRs):**
  - Use the PR template in repo
  - Requires at least 1 reviewer for merge
  - All members will be automatically added as a reviewer to all PRs
- **Issues:**
  - Will be created as described above in the Team Communication Plan
  - PRs will reference the corresponding issue(s) being addressed
- **Releases:**
  - Commits for features that complete an epic will be tagged and made into a release

When not working directly with git, all work is to be added to the repo under one commit co-authored by all those who contributed. The work breakdown by each member will be described in the commit message.

After every push to the remote repo, a GitHub action will be run to build the project for both iOS and Android devices. The artifacts created in this process will be available to download. Additionally, automated tests may be added in the build process to guard against app breaking changes. This will function as the continuous integration (CI) for the project. Continuous delivery (CD) is not necessary for this project since builds will be installed on devices manually and will not require automated deployment.

8 Project Decomposition and Scheduling

GitHub Projects will be used to organize the sub-tasks needed to complete different milestones. These sub-tasks will be represented by *GitHub Issues* within the repo. Milestones will represent less granular tasks such as a major feature in the POC where many *GitHub Issues* could be attached to a single milestone. *GitHub Issues* will follow the project tracking templates mentioned in the Team Communication Plan. In the templates, there are checklists at the bottom for all the changes that need to be made. Once they are all checked off, a PR should be created using the PR template and linked to the corresponding issue. Three views will be used within the *GitHub Projects* to see the tasks and as a table, timeline, and kanban board.

The *GitHub Project* can be found here.

This project will be scheduled with broad milestones representing major features that are interwoven with course deliverables. The planned milestone start and completion times are scheduled based on the deadlines of the course deliverables they directly contribute to.

Table 2: Planned Milestone Scheduling

Release	**Milestone**	**Planned Start**	**Planned Completion**
	Problem Statement, POC Plan, Development Plan		2024-09-23
N/A	Learn Unity, C#, AR	2024-09-14	2024-10-01
0.0	Setup Project Infrastructure	2024-09-14	2024-10-01
	Requirements Document [Rev 0]		2024-10-09
	Hazard Analysis [Rev 0]		2024-10-23
	V&V Plan [Rev 0]		2024-11-01
0.1	Implement AR object placement	2024-10-01	2024-11-11
0.2	Implement 3D object scanning	2024-10-01	2024-11-11
	Proof of Concept		2024-11-11
0.3	Design AR object database	2024-10-15	2024-11-11
0.4	Implement user account system with cloud storage	2024-11-11	2024-11-20
0.5	Implement friends, groups, and object sharing	2024-11-11	2024-12-01
0.6	Implement tour system	2025-01-01	2025-01-14
1.0	Implement a variety of settings	2025-01-01	2025-01-14
	Design Document [Rev 0]		2025-01-15
1.1	User Documentation	2025-01-15	2025-02-01
1.2	Usability Testing	2025-01-15	2025-02-01
	Demonstration [Rev 0]		2025-02-05
	V&V Report [Rev 0]		2025-03-07
	Final Demonstration [Rev 1]		2025-03-24
	EXPO Demonstration		2025-04-01
	Final Documentation [Rev 1]		2025-04-01
1.3	Implement 3D object generation via prompts	2025-04-30	2025-12-31
1.4	Implement desktop tour editor	2025-04-30	2025-12-31

Note: Course defined milestones have a grey colored background

9 Proof of Concept Demonstration Plan

One of the biggest uncertainties of the project is the achievability of the precise spatial positioning of AR objects required by the user experience we envision for our product. One of the core ideas of our product is to create context specific AR experiences; this will not be possible if we cannot precisely recreate AR objects where they were originally placed.

The other major uncertainty in this project is the feasibility of allowing users to scan in their own 3D objects using only camera and sensor data available on a smartphone. This feature will be an important avenue for user generated content on the platform, without which, general users will primarily be limited to prebuilt 3D models provided by the platform.

Our POC demonstration will require both of these features, as well as the minimum UI required to implement the following user experience:

1. User scans object, obtaining a 3D model they can place
2. User virtually places said object in their surroundings
3. User leaves the area, returns, and can see the object they placed in the correct position

10 Expected Technology

From our investigation into AR technologies, Unity with AR Foundation (an extension built on ARCore and ARKit) is the most promising option to meet the needs of our project. Those needs being:

- Cross platform support
- Visual UI editor
- Integration with powerful AR libraries with capabilities such as
  - Surface detection
  - Lighting detection
  - Motion tracking
  - Depth measurement

As will be using Unity, our primary programming language will be C#.

The best testing tool to use will be the NUnit based Unity Test Framework due to its integration with unity, and NUnit's relative popularity and similarity to JUnit, which will make finding learning resources easy. This framework will provide both unit testing as well as code coverage metrics.

Application profiling will be done using Unity's built in profiler whenever necessary

Version control will be managed using git, and the remote repository will be hosted on GitHub as it is the industry standard, and has integration with CI and project management tools.

Github Projects will be used for project management, the details of which are outlined in Project Decomposition and Scheduling.

Continuous integration will be done via GitHub actions. Unity testing and builds will be achieved using the Unity Builder and Unity Test Runner github actions provided by the GameCI open source project as a base. CI builds will be triggered on pushes to the dev branch and PRs to the main branch where changes have been made in the source code directory.

The project will primarily be developed using the Unity Engine, with VS-Code / VS as the preferred code editor to enforce code standards.

11 Coding Standard

All code must be formatted by the Microsoft C# formatter with default settings, and all suggestions from the Microsoft C# linter must be followed. In general we will follow Microsoft C# code standards.

Appendix — Reflection

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?

   It is important to create a development plan before starting the project because it can provide a clear roadmap of our project and give a high-level overview of what we need to do. Additionally, it allows us to time certain tasks accurately and create manageable milestones.

2. In your opinion, what are the advantages and disadvantages of using CI/CD?

   One of the advantages of CI/CD is that code goes through a rigorous review process, which can reduce errors and bugs. There is a level of consistency among each group member's work. It provides automation to check if the project builds successfully, following the addition of new features. The disadvantages are that it takes more time in setup and maintenance.

3. What disagreements did your group have in this deliverable, if any, and how did you resolve them?

   We disagreed on which features should be planned for the POC demonstration, what technology we should use (Unity or Unreal), how to categorize our planned features into main goals and stretch goals, and how we should decompose the project into milestones. All our disagreements were solved through team discussion, and compromise eventually leading to consensus.

Appendix — Team Charter

External Goals

We want to create a fun, high quality product that people would be interested in using. This can potentially pivot into a startup in the future. Additionally, we want to learn a variety of new skills in project management, CI/CD, as well as technical skills in Unity and C#. We want to add the project on our resumes and talk about it in interviews.

Attendance

Expectations

All team members are expected to be available for the full scheduled duration of a meeting as planned. Leaving early will be acceptable when the expected content for the remainder of the meeting relates to an aspect of the project that does not concern all members, or all items on the agenda have been completed early.

Acceptable Excuse

- Any illness that requires one to stay away from public spaces is a valid reason to attend an in-person meeting online in mild cases, and be absent from meetings in less mild cases.
- Any reasons that may make it difficult for a team member to attend a meeting that are known beforehand will ideally be resolved by rescheduling the meeting.
- Any other excuses will be evaluated on a case by case basis as a team.

In Case of Emergency

If a team member is incapable of attending scheduled meetings or completing assigned work, they will let the rest of the team know **as soon as possible**. Records of meetings are kept through *GitHub Issues* so the affected team member can look back to see what was discussed. An agreement will be made with the affected team member to divide their work among the other team members with the understanding that they will need to take on more work in future deliverables.

Accountability and Teamwork

Quality

- Between scheduled meetings it is expected that team members will attempt to complete all the action items assigned to them and they have viewed the agenda for the next meeting so everyone is on the same page.
- As for deliverables, every team member should put in their best effort when working on the deliverable parts assigned to them.
- All team members should follow the specified coding standards when adding features, making bug fixes or creating test cases.
- Team members should ensure their code passes all the tests in the testing-suite before creating a pull request.
- Changes in the source code should contain comments, to clarify to other group members the reason for the changes and its functionality.

Attitude

- Team members should respect any ideas that others may have, and consider them as a group as it pertains to the project.
- Team members are expected to follow the official McMaster University Code of Student Rights and Responsibilities.
- For conflict resolution, team members should try to solve the problem using the decision making process. If the problem persists and becomes a more serious issue, then we will raise the issue to Dr. Smith.

Stay on Track

The following will be the relative target metrics to meet for each team member:

- **Meeting attendance**: 90% of all planned meetings
- **Documentation**: Minimum contribution of 20% (based on total number of sections across all documents)
- **Code**: Minimum contribution of 20% (based on total number of issues resolved)
- **Issues authored**: Must have a minimum contribution of 15%
- **Lecture Attendance**: 80% of all software & general lectures

Team Building

- Celebrations after major milestones (POC, Rev 0, Rev 1)
- Every week on Monday for the in-person meetings, one team member buys coffee/cookies/muffins/etc. for the entire team. We alternate who pays every week.

Decision Making

Consensus whenever possible. When consensus is not achievable in the case of a large set of alternatives, approval voting will be used to narrow down the options. When there are two options and consensus cannot be achieved, we will decide by majority vote. When a majority cannot be achieved, we will flip a coin.