3XA3 Development Plan
Rhythm Master

Team #16, Rhythm Masters
Almen Ng, nga18
David Yao, yaod9
Veerash Palanichamy, palanicv

February 5, 2021

Table 1: Revision History

Date	Developer(s)	Change
February 1, 2021	Almen Ng	Initial Document
		Introduction, Team Meeting Plan
February 2, 2021	Veerash Palanichamy	Technology: Programming Language
	David Yao	Communication Plan
February 3, 2021	Almen Ng	Team Meeting Plan: Agenda, Meeting Roles
	Veerash Palanichamy	Technology: IDE, Testing
	David Yao	Git Workflow Plan, Technology: Coding Style
February 4, 2021	Almen Ng	Member Roles, Proof of Concept Demonstration
	Veerash Palanichamy	Technology: Documentation
	David Yao	Project Schedule
April 10, 2021	Almen Ng	Added project title and Project Review for Revision 1

Contents

List of Tables

1 Introduction

This document is the Development Plan for the project, Rhythm Master, based on the open source project, Frets on Fire.

2 Team Meeting Plan

The following is the plan for when, where/frequency, and length of the team meetings:

Where	When	Length
MS Teams L01	Every Wednesday at 10:30AM & 7:30PM	2 Hours Each Session
Discord	Every Thursday at 9:00PM	2 Hours

Table 2: Team Meeting Plan

2.1 Rules for Agenda

The rules for the agenda for all team meetings are as follows:

1. An agenda shall be made and finalized before every meeting. The agenda shall have the following information:
   (a) Topics must be discussed in the form of questions
   (b) An estimate realistic time must be provided for each topic
   (c) A specified person must be identified for leading each topic
   (d) A process for addressing each item must be proposed
2. There shall only be one chair/facilitator at each meeting
3. Meeting minutes will be taken accordingly
4. Any conflicts and/or discussion on previous action items shall be addressed before the beginning of the current agenda
5. All members must be present during meetings unless otherwise mentioned
6. All meetings must be concluded with written statements on decisions made as well as tasks that must be fulfilled by the next meeting
7. All members must work and communicate respectfully to each other during the meetings

2.2 Meeting Roles

The following are the main roles that have been identified for every meeting:

Meeting Role	Name
Chair	Veerash Palanichamy
Recorder	Almen Ng
Timekeeper	David Yao
Participant	Everyone

Table 3: Meeting Roles

2.3 Meeting Role Descriptions

2.3.1 Chair

1. Leads the meeting in accordance to the agenda
2. Ensures equal speaking opportunities amongst all team members
3. Facilitates respectful discussion
4. Facilitates conflict resolution if needed
5. Communicates conclusions and next steps

2.3.2 Recorder

1. Records key decisions, conclusions, and action items
2. Compiles notes into a standard document
3. Distributes notes and conclusions

2.3.3 Timekeeper

1. Manages time limits set for each item

2.3.4 Participant

1. Understands the agenda and the purpose of each meeting
2. Contributes to the agenda items
3. Communicates and works respectfully and collaboratively to create a positive, safe atmosphere
4. Present ideas concisely while keeping in mind the time constraints

3 Communication Plan

Aforementioned meetings are to be conducted in voice calls over Discord or Microsoft Teams. Members should be in the appropriate voice channel at the meeting's start time. A Facebook group chat including all three members has been created for any communications outside designated meeting times. All members are expected to check this group chat for updates on a daily basis, or at their earliest convenience.

Urgent issues with project code, planning, or design must be relayed through the Facebook group chat as soon as possible. For more minor problems, creating a Git issue is sufficient. Table 4 contains contact information for each group member.

Name	**Discord**	**MACID**
Almen Ng	Mumbojumbo	nga18
Veerash Palanichamy	Calculess	palanicv
David Yao	d i n g u s	yaod9

Table 4: Member Contact Information

4 Member Roles

The following are the assignment of main roles of the team:

Name	**Role(s)**
Almen Ng	Scribe
	Documentation Expert
	Developer
	Tester
Veerash Palanichamy	Technology Expert
	Git Expert
	Developer
	Tester
David Yao	Team Leader
	LaTeX Expert
	Developer
	Tester

Table 5: Member Roles

5 Git Workflow Plan

The central repository is named 3XA3_L01_GR16_Project. This project will follow the feature-branch model. Project features and updates are to be pushed to branches of the main repository, which are to be merged only after proper testing to ascertain proper function. The master branch must always remain functional throughout the duration of the project. All commits must be labelled in detail in order to facilitate tracking of issue priority, revision history, and progress toward completion of given features. In addition, the team will use milestones to track progress towards weekly deliverable.

6 Proof of Concept Demonstration Plan

6.1 Potential Risks and Difficulties

6.1.1 Scope Size

When selecting Frets on Fire, there was uncertainty if it could be implemented by the end of the term as it included a lot of features and songs.

6.1.2 Implementation

The team is planning to use C# alongside the Unity game engine to develop the project. All members have prior experience with C#, but not Unity. There will be a learning curve associated with using this game engine to develop the game.

6.1.3 Testing

In terms of testing, a wide variety of types of testing will be used such as exploratory testing and play testing. Testing the game with exploratory testing becomes a complex and lengthy problem due to abundant number of scenarios the user can go through.

6.1.4 Library Installation

Library installation is not a concern as all the libraries are downloaded alongside Unity.

6.1.5 Portability

Portability is not a major concern because the project is intended to be compiled into an executable file, which should be able to run on macOS and Windows systems. The source code should also be viewable and editable on any IDE supporting C#. However, Unity does not support forward compatibility, meaning old versions of Unity cannot be used with projects made in a newer version.

6.2 Plan to Overcome Risks

6.2.1 Scope Size

The team has minimized the scope of the game to improve on one of the songs in Frets on Fire and implement the main features/settings in it.

6.2.2 Implementation

The team members will install Unity and go through various tutorials prior to starting the implementation of the game to grasp how to use Unity for game development.

For the Proof of Concept Demonstration, the most fundamental functionality will be demonstrated, given the time constraints. The plan is to demonstrate player input in response to game output displayed on the screen. Animations, music, and further features will be implemented later in the project, as more time becomes available.

6.2.3 Testing

To overcome the risk of not being able to use exploratory testing within the time frame, the team will use other forms of testing, such as unit testing, to ensure that each section/component of the software behaves as intended. Doing this will minimize the number of bugs that will surface during exploratory testing. Once all modules have been tested, exploratory testing can commence with all group members and some colleagues participating.

For the Proof of Concept Demonstration, some unit tests would already have been made to ensure that the demonstration is successful.

6.2.4 Library Installation

All team members should have collaboratively developed the demonstration which means Unity and the associated libraries should have been installed.

6.2.5 Portability

All programmers will coordinate on the version of Unity that is used to ensure compatibility.

7 Technology

7.1 Programming Language

The code will be written using a current stable version of C#. This code will be used within the Unity game engine. C# was chosen because of its ease of integration with the Unity game engine. Team members also have some experience with C#.

7.2 IDE

The IDE that will be used is Visual Studio Code. This IDE was selected for its ease of use and having an abundant set of extensions that can be used for documentation and static analysis.

7.3 Testing

Testing will be done with the help of the Unity Test Framework package which provides a standard test framework for Unity applications. Test cases will be written in C# scripts which will be run by the Unity Test Framework.

7.4 Documentation

This project will be documented using LaTeX, and will be made available in the project repository on GitLab. The code will be documented using Doxygen. Doxygen will auto-generate a PDF and HTML file with code documentation. Doxygen is simple to use and all the team members have past experience using it.

7.5 Coding Style

Code written should conform to the C# style laid out by Google, at: `https://google.github.io/styleguide/csharp-style.html`. Classes and methods are to be named using *PascalCase*, and variables and parameters are to be named using *camelCase*.
The goal is to maintain readability and ease of maintenance by other group members.

8 Project Schedule

The project schedule is organized as a Gantt chart, which can be viewed here: `https://gitlab.cas.mcmaster.ca/palanicv/3xa3__l01_gr16_project/-/blob/master/ProjectSchedule/Group16Gantt.pdf`.
The chart highlights deliverable deadlines, work distribution, and individual progress. It is to be updated with group member's progress continuously as the project continues.

9 Project Review

For this project, Group 16 has re-implemented and improved upon an existing open-source rhythm game project, Frets on Fire. Group 16, the development team, concluded that the project was a success. Problems with the original project were the graphics were unappealing, the programming language that was used to implement the game was outdated, and the testing of the game did not provide a lot of coverage. The team was able to solve these issues by implementing a system with updated graphics using free Unity 3D assets, developed the system in Unity Game Engine, a cross platform game engine that uses a modern programming language, and increasing the test coverage by implementing test

cases based on the non-functional and functional requirements the team has laid out in our **Software Requirements Specification**. On top of the improvements stated above, the source code is also modularized and well documented, a leaderboard was added, and the user-interface was improved.

The final product that was presented in the Final Demonstration fulfilled all of the requirements that were stated in the **Software Requirements Specification**; however, there were some features that the development team wished to add and implement but was unable to do due to the time constraint, including but not limited to an online leaderboard and creating and loading note maps. The team also had to remove some non-functional requirements as they required a lot of work that could not be done within the timeframe given. One major change that the team had to make was that the actual implementation includes a lot more modules. This is because of how Unity allows developers to create prefabs, which are GameObjects that can be reused throughout the project. Creating prefabs support the idea of low coupling, high cohesion, encapsulation, and reusability in the system; thus, the development team wished to create them and use it throughout the project. As a consequence, more modules had to be created to be placed within the GameObjects.

In terms of team dynamic, the development team worked very well together and were able to distribute the work that needed to be completed evenly amongst everyone. The team was able to communicate effectively through a Facebook Messenger group chat and through meetings done virtually on Discord. The team came to the consensus, however, that communication and working on the project would have been easier if the team were able to meet in person, but due to the COVID-19 pandemic, in-person meet-ups were not an option.

In conclusion, the developers of Rhythm Master believes the project was a success, though there are still a lot more features that will be further implemented after this course is finished. The team has learned a lot about the Software Development process, the documentation involved in software development, and collaborating in a team(with the challenge of everything being done virtually). The team is exceptionally proud with the product and look forward to improving it in the near future.