

**Computer Science and Engineering**

### \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Archie-ology, A Video Game**

**Project Management Plan**

### Version 1

Document Number: SPMP-001

Project Team Number Team A25

Project Team Members:

Jungho Kook jk5541

Jeffrey Qiu yq544

Juliet Ramdass jsr483

Sabrina Supapkooha sps450

**REVIEW AND APPROVALS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Printed Name and**  **Title** | **Function**  **(Author,**  **Reviewer,**  **Approval)** | **Date** | **Signature** |
| Jungho Kook | Moderator/Author | November 6, 2018 | Jungho Kook |
| Jeffrey Qiu | Recorder | November 6, 2018 | Jeffrey Qiu |
| Juliet Ramdass | Reader | November 6, 2018 | Juliet Ramdass |
| Sabrina Supapkooha | Inspector | November 6, 2018 | Sabrina Supapkooha |

**REVISION LEVEL**

|  |  |  |
| --- | --- | --- |
| **Date** | **Revision Number** | **Purpose** |
| November 6, 2018 | Version 1.0 | Initial Release |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

### TABLE OF CONTENTS

[1. INTRODUCTION 6](#_Toc527704657)

[1.1 Purpose 6](#_Toc527704658)

[2. SCOPE 6](#_Toc527704659)

[2.1 Identification 6](#_Toc527704660)

[2.2 Bounds 6](#_Toc527704661)

[2.3 Objectives 7](#_Toc527704662)

[3. OVERALL SYSTEM OVERVIEW 7](#_Toc527704663)

[3.1 Context Diagram 7](#_Toc527704664)

[3.2 Additional Descriptive Items 7](#_Toc527704665)

[4. DOCUMENT OVERVIEW 8](#_Toc527704666)

[5. REFERENCE DOCUMENTS 8](#_Toc527704667)

[6. BUSINESS REQUIREMENTS 8](#_Toc527704668)

[6.1 Technology 8](#_Toc527704669)

[6.2 Economics 9](#_Toc527704670)

[6.3 Regulatory and Legal 9](#_Toc527704671)

[6.4 Market Considerations 9](#_Toc527704672)

[6.6 Human Resources and Training 10](#_Toc527704673)

[7. SPECIFIC REQUIREMENTS (DESCRIPTIVE FUNCTIONAL REQUIREMENTS) 11](#_Toc527704674)

[7.1 Functional Descriptive Detailed Requirements 11](#_Toc527704675)

[7.2 Requirement Use Cases 12](#_Toc527704676)

[*7.3 Use Case Diagrams* 14](#_Toc527704677)

[8. NON-FUNCTIONAL DESCRIPTIVE DETAILED REQUIREMENTSNONFUNCTIONAL REQUIREMENTS DEFINITIONS 14](#_Toc527704678)

[9. ANALYSIS 14](#_Toc527704679)

[9.1 Component (Component/Package/Subsystem) Architecture 15](#_Toc527704680)

[9.2 Component Architecture Diagram 15](#_Toc527704681)

[9.3 Component Descriptions 15](#_Toc527704682)

[9.4 Class Diagrams 15](#_Toc527704683)

[9.4.1 Individual Class Diagrams 15](#_Toc527704684)

[9.5 Events 15](#_Toc527704685)

[9.6 Activity/State (Scenario) Section 15](#_Toc527704686)

[9.7 State logic 15](#_Toc527704687)

[9.8 Behavior 15](#_Toc527704688)

[10 SYSTEM TEST PLAN REQUIREMENTS 15](#_Toc527704689)

[11 QUALIFICATION PROVISIONS 16](#_Toc527704690)

[12 REQUIREMENTS TRACEABILITY 17](#_Toc527704691)

[13 EVOLUTION OF THE RAS 17](#_Toc527704692)

[14 RATIONALE 18](#_Toc527704693)

[15 NOTES 18](#_Toc527704694)

[16 APPENDICES 18](#_Toc527704695)

[Schedule Tracking 19](#_Toc527704696)

[Cumulative 19](#_Toc527704697)

[Defect Tracking 20](#_Toc527704698)

[Cumulative 20](#_Toc527704699)

# 1. Overview

## 1.1 Project Summery

The genre for metroidvania platformers are full of dark games such as Hollow Knight and Dead Cells. We want to create a game with a brighter atmosphere that embodies the grace and curiosity of a cat. Players that play this game should be able to feel what it’s like to explore the world and the beauty that it contains.

The aim is to develop and produce a challenging platformer that rewards the player for perseverance and creative thought while encouraging the player to plan ahead and prep for the next obstacle. This fast thinking and reaction time is inspired from the phrase “cat like reflexes” where a split second decision could be the difference in success or failure.

The purpose of this document is to give an overview of the project and all the processes involved with the project. This includes the project organization and management, as well as technical and supporting processes.

The audience of this document will be anyone that is working on the project. This includes the client, developer team, and anyone else who may be involved.

## 1.2 Purpose, Scope, and Objectives

The purpose of this project is to develop a game focused on item collection, exploration, and puzzle solving featuring a feline as the main protagonist. Many platformer games made in recent years have had a dark atmospheres or an underlying dark theme. In order to being new life to the genre, the solution is to create a game that will have a more pleasant atmosphere that is full of life and wonder. By doing this, the game will be able to attract a wider variety of consumers to the genre.

    The player will play as Archie, a cat who wishes to fill his owner’s museum with rare artifacts. In order to do this, Archie must venture into the grout outdoors. The outdoors will consist of a large map with many subsections. Upon discovering a new artifact, Archie will develop either a new skill that will allow him to traverse the map in a new way or a new aesthetic item that can be used to customize the character. The player will be able to change his clothing and other aesthetic aspects such as fur pattern. There will be an achievement system as a way to give the player goals to strive for. Finally, the game will have an user interface that will be easy to navigate.

Some features that may be excluded from the final game includes the number of artifacts that may be in the game. Due to the amount of effort that may be need to implement a new ability, the number of artifacts may be decreased.

    In order for this game to be a success upon release it is very important to market this game in places that gamers trust to get their games. This includes websites such as Steam, IndieGala, Humble Bundle and more. Due to the low production cost of the game as well as the lack of reputation of the developers, the game will be marketed at about five dollars to give as many people as possible a chance to play the game while also supporting the developers. The game will be promoted through YouTube. By giving a copy of the game to popular YouTubers for free, they can play the game, give a review, and then post a video about the game. As people watch the video, they may find the game interesting and worth playing.

## 1.3 Assumptions and Constraints

One of the primary assumptions that affect the success of the game is that the game will fit a niche that will attract gamers. Based on research, it appears that there are enough people interested in this genre to warrant creating the game, however, formal polls and questions were not asked so we are unsure how large our audience really is.

There are many constraints to this project. This project is limited to one semester during the school year so that means that the project must be worked of during free time between classes. There is also a budget constraint due to the fact that we are college students that can’t afford to invest too much money into a project. Since the game is primarily being developed in Unity, we are limited to features that can be produced using the Unity engine. The art is being done by NYU, Pratt Institute, and School of Visual Arts artists and the music will be created using free music software.

## 1.4 Project Deliverables

|  |  |  |  |
| --- | --- | --- | --- |
| Deliverable | Delivery Date | Format | Quantity |
| Project Proposal | 09/20/2018 | Word Document | 1 |
| Software Business Specification (RAS) | 10/04/2018 | Word Document | 1 |
| Software Requirement and Analysis Specification (RAS) | 10/19/2018 | Word Document | 1 |
| Software Project Management Plan (SPMP) | 11/06/2018 | Word Document | 1 |
| Software Analysis Specification – Final (RAS) | 11/20/2018 | Word Document | 1 |
| Software Design Document (SDD) | 11/29/2018 | Word Document | 1 |

## 1.3 Schedule and Budget Summary

|  |  |  |
| --- | --- | --- |
| Deliverable | Description | Delivery Date |
| Project Proposal | Document proposing the idea for the project. | 09/20/2018 |
| Software Business Specification (RAS) | First version of the RAS which focuses on the business side requirements. | 10/04/2018 |
| Software Requirement and Analysis Specification (RAS) | Second version of the RAS which adds in the functional requirements for the project. | 10/19/2018 |
| Software Project Management Plan (SPMP) | First version of the SPMP which lays out the plan for many of the project processes. | 11/06/2018 |
| Software Analysis Specification – Final (RAS) | Third version of the RAS which fixes all defects from previous documents. | 11/20/2018 |
| Software Design Document (SDD) | Document that describes the software product with the intention of giving it to the software development team. | 11/29/2018 |

## 1.6 Evolution of the Plan

The plan is scheduled to be updated after each playtesting milestone. The outline for the game will be updated with notes of what reviewers had to say about their experience with each set of features added. The ability to learn the game through playing it, the enjoyment it brings, and the smoothness of the controls will all be tested and changed or reapproved depending on the metrics created from the outcome.

There will be a list of questions produced to measure the quality of the player’s experience as well as the notes taken by a team member on the real time reactions. Testing will be done with playtesters of varying skills and familiarities with the game.

If a feature is found to be too unusable or out of place in the rest of the game, the outline will again be changed at whatever point this is determined. This would be an unscheduled change since all mechanics and features planned for in the outline are expected to be worth creating. These would be found either through playtesting by non-developers or through self quality checks of overall game flow. The outline of game progression and the layout of the world would both have to be adjusted to deal with this type of change.

Reviews of the flow of the game are scheduled to be done with each new feature developed. These will involve testing of the game so far by team members or outside people who have become familiar with the game through playtesting over the time of development.

Testing of individual features and their compatibility with the rest of the program will be done in a copy of the current game iteration before updating to the next version that includes these changes.

# 2. References

Project Proposal

Project Name: Archie-ology

Project Team: A25

Version 1.0

September 20, 2018

System Requirement and Analysis Specification

Project Name: Archie-ology

Project Team: A25

Version 1.1

October 19, 2018

# 3. Definitions

RAS – Requirement Analysis Specification

SPMP – Software Project Management Plan

# 4. Project Organization

## 4.1 External

Any part of the project that is developed in Unity will be considered to be a system entity. This will include code for the program and the user interface for the game. Items that are external to the system would include the art as well as the music for the game.

## 4.2 Internal Structure

The structure of the team is one in which each person belongs in each other groups with each person being a leader in one area. The four primary are the project leader, the quality assurance leader, the risk management leader, and the documentation leader. Each leader works with the other members of the development team to make sure each task is being completed with the highest quality. Because each person belongs in a team as well as a team, each team member contributes equally.

In order to keep up with configuration management, we will use Github to keep track of versions. However, before each version can be a new baseline, it must be approved by the entire team in a meeting. However this is necessary for large changes. For minor changes or bug fixes, it only needs to be approved by one other team member. Quality assurance is also managed in a similar way. Inspections are scheduled by the quality assurance leader and each member participates.

Due to the structure of the team, communication is key and every member of the team is expected to communicate when necessary. Any last minute planning will be done through messaging through a group chat.

## 4.2 Internal Structure

|  |  |
| --- | --- |
| Role | Responsibilities |
| Project Leader | Leading Meeting  Tracking Team Progress |
| Quality Assurance Leader | Scheduling Inspections  Assigning roles for inspections |
| Risk Management Leader | Tracking Risks  Planning around risks |
| Documentation Leader | Tracks Documentation  Ensure documentation is updated |

The table above lists the unique responsibilities given to each team leader. However, each member of the team have general responsibilities that need to be fulfilled as well. These include things such as updating documentation, participating in inspections, peer reviewing code and documentation, etc. So although there aren't many unique responsibilities each member is expected to fill in new roles when necessary.

# 5. Management Process

This section will include the planned project management processes for this project. This includes the plans for dealing with details needed to start the project, risk management throughout the project, a schedule of work on the program, and methods of measuring satisfaction and quality during construction and after the final iteration is produced.

## 5.1 Start-Up Plan

This section describes the details of the project that are required to be understood before software construction.

### 5.1.1 Estimation Plan

This section will not be included in this version of the document.

### 5.1.2 Staffing Plan

Software construction, quality testing, playtest administration and monitoring, and product distribution upon release will be handled by team members.

Sprite sheets and background images will be developed by two to three outside artists.

### 5.1.3 Resource Acquisition Plan

This section will not be included in this version of the document.

### 5.1.4 Training Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Training Type | Require Training | Entry Criteria | Exit Criteria | Training Method |
| Technical: Unity Skills | 4 | Automatic | Proof of competency | Online tutorials and practice |
| Technical: C++ Extensions for Unity Skills | 4 | Automatic | Proof of competency | Online tutorials and practice |
| Managerial: Communication and Useful Review Skills | 4 | Automatic | Approval of other team members | Online resources |
| Managerial: Playtesting Administration | 4 | Automatic | Testing with other products | Self developed process reviewed by mentors |

## 5.2 Work Plan

This section includes the work activities, resources, and estimated schedule of this project.

### 5.2.1 Work Activities

See Appendix I.

### 5.2.2 Schedule Allocation

See Appendix I.

### 5.2.3 Resource Allocation

See Appendix I.

### 5.2.4 Budget Allocation

This section will not be included in this version of the document.

## 5.3 Control Plan

### 5.3.1 Requirement Control and Traceability

Change control will be used to track and document the changes made to each iteration of the project. Any changes suggested will have been associated with a specific listed requirement of the project to be included. Large changes will be documented as being done by a specific team member. All team members will review the product not just for quality and correctness, but for relevance to the known requirements.

If there is a change to any requirements the current iteration will be re-reviewed to make sure it reflects this change.

### 5.3.2 Schedule Tacking and Adjustment

As each sub-goal is assigned to a team member to work on, that team member gives an estimate in how long they think it would take to do. This estimation is based on that member’s current programing level but is also based on the work that person has done in the past with the number of defects. If the difference between the team member’s estimated progress is too different from the planned process additional team members will step in to help complete the task. The hours spent by all team members on a milestone is recorded and tracked throughout the process in a table.

### 5.3.3 Budget Tracking and Adjustment

This section will not be included in this version of the document.

### 5.3.4 Quality Control

Each team member will complete training in the necessary skills needed for this project as described in Section 5.1.4. These trainings will be required and the mastery of the skills proven before construction of the product begins. This will be recorded as an assurance of quality by each team member.

Each feature of this product will be produced in an isolated environment. It will be reviewed by both team members responsible for constructing it. Once they give assurance that it is ready to be integrated into the main iteration, it will be tested in a copy of the iteration for compatibility and defects caused by interactions between features. All defects found during this process will be recorded and their status tracked until they are fixed. Only once the copy has been approved by all team members for quality and correctness will it be merged into the newest iteration.

Scheduled playtesting is included to ensure that product we are created is up to the standards and expectations of the users, perspective players. Their feedback will be recorded each time and compared to a standard of quality that we determine to be the minimum accepted amount. If it metrics reveal that we are below that standard the product shall be reworked for the next iteration along with scheduled changes.

### 5.3.5 Reporting Mechanisms

This section will not be included in this version of the document.

### 5.3.6 Metrics Collection Plan

Metrics will be collected upon the completion of each new feature before it is added to the current iteration and upon the review of the previous.

The metrics recorded for each feature will include a check for defects and a recording of the number of defects expected, found, and fixed. This will be recorded in a chart in the documentation for the current iteration once it reached the point where it can be merged. The iteration is then checked for defects in the same way and these cumulative metrics will be added to a separate chart in the documentation.

Each iteration will also have metrics collected from users who playtest. Team members administering the playtesting will record the enjoyment, frustration, and ease of learning that they observe players having during the process. This will be recorded using a one to five scale. There will then be a set of questions asked by the team members to the playtesters about the experience afterwards. These answers will also be converted into metrics. The metrics collected from each round of playtesting will be added to the iteration documentation and the appendix of the next iteration for easy reference.

### 5.4 Risk Management Plan

When risks are identified, they are sorted by the risk that has the highest impact on a scale of low, moderate, and high. In each of these subsections, they are sorted in the most likely frequency based on opinions. From there the risk management leader decides the priority of the risks how the best methods to either avoid or retire that risk. These are tracked in the risk documentation. The risk management leader may decide to meet with the team leaders to develop the best way to retire the more complex risks.

As new risks are found, it is the responsibility of the risk management leader to access the risk and retire it. This is done as quickly as possible and the risk is documented so that there is it doesn’t cause a problem in the future. As a risk occurs, it is up to the able team members to try to resolve the risk as quickly and smoothly as possible. I do not know how to find out how to do it. After the risk has been fully retired, we analyze how well the retiring process was and possible change the documentation for the risk so that it can be solved a better way in the future.

### 5.5 Post Implementation Plan

This section will not be included in this version of the document.

# 6. TECHNICAL PROCESSES

## 6.1 Process Model

|  |  |  |
| --- | --- | --- |
| Milestone Name | Description | Date |
| Game Outline | A written and organized outline of how the game should progress is produced. | 02/07/2019 |
| World Layout Finalized | The projected layout of the maneuverable world is finalized and approved by all team members. | 02/15/2019 |
| Physics | The basic physics of the game is coded and compatible with the appropriate locations in the world design. | 02/25/2019 |
| Playtesting | The game has been playtested by individuals outside the project and adjusted until standards are satisfactory. | 03/01/2019 |
| Additional Mechanics | The first set of special game mechanics is developed and compatible with the rest of the program. | 03/21/2019 |
| Playtesting | The game has been playtested by individuals outside the project and adjusted until standards are satisfactory. | 03/25/2019 |
| Additional Mechanics | The second set of mid game mechanics is developed and compatible with the rest of the program. | 04/01/2019 |
| Art Production | The final completed versions of all sprites and visuals have been produced and provided to our team. | 04/05/2019 |
| Graphics Incorporation | The aspects of the game are now linked to their final sprites and graphics. | 04/15/2019 |
| Playtesting | The game has been playtested by individuals outside the project and adjusted until standards are satisfactory. | 04/20/2019 |
| Additional Mechanics | The final set of late game mechanics is developed and compatible with the rest of the program. | 04/25/2018 |
| Playtesting | The game has been playtested by individuals outside the project and adjusted until standards are satisfactory. | 05/10/2019 |
| Project Completion and Final Review | All aspects of the project are fully combined and approved for quality to ensure it is ready for release. | 05/20/2019 |

Major Milestones and their timings are displayed above.

The project was divided into several milestones as a way to spitting the project up into understandable sections where there is an achievable goal each time. Upon the completion of a milestone, a new baseline is generated. Because of this, it is extremely important that the whole team approves of the new baseline before moving onto the next milestone. In order to achieve each milestone, each feature that needs to be added becomes its own sub goal. These goals treated as smaller milestones and also needs approval of a peer before completing that mini milestone. These objectives are identified at the beginning of each milestone by the entire team.

## 6.2 Methods, Tools, and Techniques

All programming will be done in the Unity Engine. This means that the scripts for the game will be developed in the C# or C++ programing language. GitHub will also be used to keep track of version control. In order to keep code readable even though many different people are coding, some standards will be put in place. One of these standards will be to keep variable names descriptive and camel cased so that anyone can look at a variable and understand what it means.

The game itself will be developed in an object oriented fashion. So all the classes will be documented so that anyone can understand what each class means and what it does.

## 6.3 Infrastructure Plan

Each member of the team will need to install Unity and GitHub on to a computer that they have access to. When working in Unity, all the changes are local so each team member has freedom over their local copy of the game. However, when working with the GitHub repository, each team member must work in their own branch in order to retain the integrity of configuration management. When testing the game, each team member will download the latest safe version of the game for testing.

## 6.4 Product Acceptance and Migration Plan

This section is not covered in this version of the document.

# 7. SUPPORTING PROCESSES PLANS

## 7.1 Configuration Management Plan

Configuration management is done using GitHub as our primary tool. By using GitHub, we are able to view look at each release and potential future releases. Tracking the current status of changes being made can be done by checking in on the person seeing what progress that person made. There should also be a new branch on the repository that corresponds to the change being made. This would make peer reviews easier because everyone on the team has access to the GitHub repository.

The process for submitting changes is quite simple. For smaller releases, the code will go through inspection and a peer review before it becomes added to the main repository. After the document passes the peer review and the inspection, there is a pull request on GitHub stating who is making what changes and who else approves of these changes. This is used to keep track of baselines and who is making what changes.

For the general milestone pull requests, the entire team is assembled to look at all the changes that occurred during a milestone and to re-review everything before creating a new baseline

## 7.2 Qualification Plan

Development and qualification activities are depended on each other to a certain extent. Because there is a strict time deadline, the qualification activities will no matter how much progress had been made on the development side. Ideally, we can finish a sub-goal before having reviewing the code, but the lack of time prevents us from having that luxury. So code reviews and inspections happen no matter how much progress is made then the schedule is adjusted based on the progress.

During review methods, no matter how many people attend one, each person is given a set of roles. This allows people to know which areas they should be focusing on during the review process. Each person can take turns being the reader but the author will assume the scribe role while the other person will focus on finding defects.

Prototyping and simulations will be programmed in Unity so that any ideas or innovations developed can be transferred into the final project more easily. Testing will be done through various methods. One method is having the developer play the game. Through this method, developers and try to find specific bugs or certain things that he or she is worried about. The second testing method is by asking the public to play the game. By hosting game testing sessions we can receive feedback from real players who play the game and adjust some of the requirements based on their feedback.

## 7.3 Documentation Plan

The documentation leader would assume leadership over the documentation for the project. This includes tracking which documents need to be generated, which one needs to be changed and which ones are relevant at the time. While this is managed by the documentation leader, he or she is also able to assign tasks to the other development members to aid with keeping documentation up to date. Team members may need to write documentation or review documentation.

The document will be generated and reviewed by all the members of the development team. The document leader will approve and date it for an initial baseline version. The documentation leader is also in charge of distributing the documentation for review.

## 7. 4 Quality Assurance Plan

In order to ensure that the software we are producing is of the highest quality, there will be many reviews and inspections to make sure that the project has minimal defects and follows the requirements. At the beginning of every milestone, the list of items to implement for the milestone comes from the requirements. This is done to ensure that the customer receives a product with the feature that they are looking for.

If the developer wants to add a feature to the product, then it can be done by pitching the idea to the rest of the development group and the client. If everyone agrees that the change will be beneficial to the product as a whole the change is add and the documentation is changed to reflect the change.

## 7. 5 Reviews and Audits

There are many ways to review the software. During this section we will cover management reviews, developer peer reviews, technical reviews, walkthroughs, inspections, and audits.

Management reviews will require the entire development team to prepare to give a short overview of the current status of the project to the professor. Feedback will be written down but fixed during this session.

Peer Review will be more involved than desk checks. The author will pair up with one other member of the team. Together they will review the document as if it was a small informal inspection. The peer will read the document out loud as they track the defects.

During technical reviews, the code will be read by at least one other developer to ensure that the coding guidelines are being follows as well as to make sure that the code is being implemented in a way that makes sense instead of a way that is very inefficient.

Walkthroughs are more involved peer reviews due to the fact that it requires a team meeting to walkthrough the document. However they are not as formal as inspections. The moderator for the walkthrough tends to be the author of the document. The document is reviewed together while tracking defects.

We also have scheduled inspection sessions. The purpose of these inspections is to identify defects in the document. Defects include anything that makes the document incorrect, inconsistent, or unstable. Defects will not be fixed during these inspections. During inspections, each member of the team will assume a different role so that each member has a category to focus on. Role assignments will be given based on what is under inspection.

With audits the team will meet together with the person who is performing the audit. The team will give a short presentation of the code, similar to management reviews, and ask for feedback which will be written down for future review.

## 7. 6 Problem Resolution Plan

If a software problem occurs, the first step requires whoever is experiencing the error to do some research into why the error is occurring and to see what they can do to fix it. If nothing fixes the issue, then another developer can be asked to help. Another option is to post the problem in to the developer group message to see if anyone else had the problem before and possibly has a solution.

If these steps fail, then GitHub is used to download the latest safe version of the software. If this fixes the problem, the developer would just need to remake his or her changes. However, if this doesn’t fix the problem, Unity customer support would need to be emailed to see if there is something wrong with Unity. Once a solution to the problem is found, the solution is posted to the group chat so that the rest of the developers would know what to do if they ever encounter the problem.

## 7. 7 Environment Management Plans

This section will not be included in this version of the document.

## 7. 8 Process Improvement Plan

This section will not be included in this version of the document.

# 8. Additional Plans

None at this time.

# 9. Index

1. Overview ………………………………………………………………………………………………………………………….6

2. References ……………………………………………………………………………………………………………………….8

3. Definitions ……………………………………………………………………………………………………………………….9

4. Project Organization ………………………………………………………………………………………..………………9

5. Management Processes …………………………………………………………………………………….………….10

6. Technical Processes ……………………………………………………………………………………………………….14

7. Supporting Processes ………………………………………………………………………………………….………….16

8. Additional Plans ………………………………………………………………………………………………..……………19

9. Index……………………………………………………………………………………………………………….……………..19

10. Rationale ………………………………………………………………………………………………………………………19

11. Notes…………………………………………………………………………………………………………………………….19

12. Appendix……………………………………………………………………………………………………………………..20

# 10 Rationale

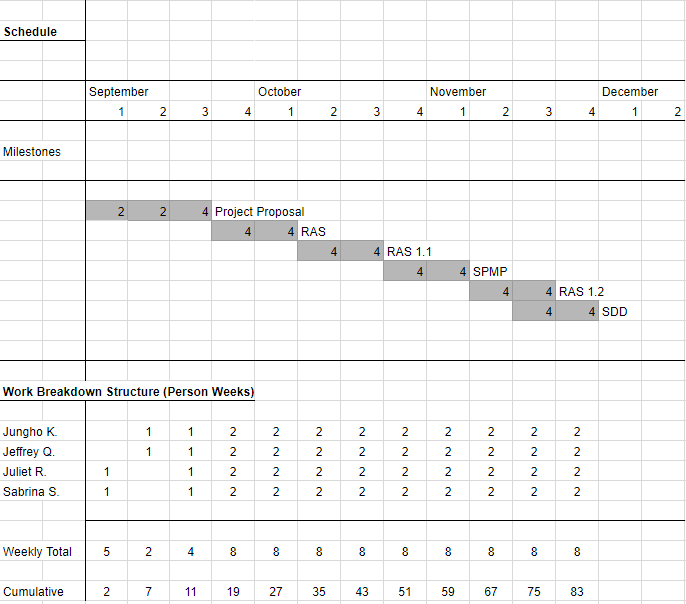
None at this time.

# 11 NOTES

None at this time.

# 12 APPENDIX

## Gantt Chart



## Schedule Tracking

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or Team) | Estimated (Hours) | Actual (Hours) | Difference |
| RAS |  |  |  |  |
|  | Juliet Ramdass | 2.5 | 3 | 0.5 |
|  | Jeffrey Qui | 1 | 2 | 1 |
|  | Sabrina Supapkooha | 2 | 2.5 | 0.5 |
|  | Jungho Kook | 2 | 2 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or Team) | Estimated (Hours) | Actual (Hours) | Difference |
| RAS 1.1 |  |  |  |  |
|  | Juliet Ramdass | 0.5 | 1 | 0.5 |
|  | Jeffrey Qiu | 1 | 1 | 0 |
|  | Sabrina Supapkooha | 1 | 1.5 | 0.5 |
|  | Jungho Kook | 0.5 | 1 | 0.5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or Team) | Estimated | Actual | Difference |
| SPMP |  |  |  |  |
|  | Juliet Ramdass | 3.5 | 5.5 | 2 |
|  | Jeffrey Qui | 4 | 5.5 | 1.5 |
|  | Sabrina Supapkooha | 3 | 4.5 | 2.5 |
|  | Jungho Kook | 2.5 | 4.5 | 2 |

**Cumulative**

|  |  |  |  |
| --- | --- | --- | --- |
| Who (individual or Team) | Estimated | Actual | Difference |
| Team for RAS | 7.5 | 9.5 | 2 |
| Team for RAS 1.1 | 3 | 4.5 | 1.5 |
| Team for SPMP | 12 | 20 | 8 |

## Defect Tracking

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or Team) | Estimated | Actual | Difference |
| RAS | Juliet Ramdass | 15 | 23 | 8 more |
|  | Jungho Kook | 10 | 15 | 5 more |
|  | Jeffrey Qui | 13 | 20 | 7 more |
|  | Sabrina Supapkooha | 18 | 19 | 1 more |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or Team) | Estimated | Actual | Difference |
| RAS 1.1 | Juliet Ramdass | 5 | 8 | 3 more |
|  | Jeffrey Qiu | 4 | 8 | 4 more |
|  | Sabrina Supapkooha | 7 | 5 | 2 less |
|  | Jungho Kook | 2 | 5 | 3 more |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or Team) | Estimated | Actual | Difference |
| SPMP | Juliet Ramdass | 20 | 23 | 3 more |
|  | Jungho Kook | 25 | 26 | 1 more |
|  | Jeffrey Qui | 22 | 23 | 1 more |
|  | Sabrina Supapkooha | 20 | 24 | 4 more |

**Cumulative**

|  |  |  |  |
| --- | --- | --- | --- |
| Who (individual or Team) | Estimated | Actual | Difference |
| Team for RAS | 54 | 67 | 13 more |
| Team for RAS 1.1 | 20 | 26 | 6 more |
| Team for SPMP | 87 | 96 | 9 more |