**SOFTWARE DEVELOPMENT PLAN**

**<ARMAMENT>**



**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision # | Author | Revision Date | Comments |
| 1.0 | Jeremy Scott | Feb. 13, 2019 | initiated |
| 1.1 | Alex Cohn | Feb. 13, 2019 | Linked Gantt chart, minor additions/edits |
| 1.2 | Andrew Pitt | Feb 13, 2019 | System overview, Development env |
| 1.3 | Alex Cohn | Feb 14, 2019 | Minor additions |
| 1.4 | Keith Bosworth | Feb 14, 2019 | Gantt chart |
| 1.5 | Jeremy Scott | Feb 15, 2019 | Tasks |
| 1.6 | Ben Mankin | Feb 15, 2019 | Basic edits, Clean-up |
| 1.7 | Andrew Pitt | Feb 15, 2019 | Update dev env, formatting |
| 1.8 | Alex Cohn | Feb 15, 2019 | Edits, formatting |
| 2.0 | Keith Bosworth | March 9, 2019 | Header and System Overview updates |
| 2.1 | Keith Bosworth | March 9, 2019 | Gannt Updates, Task Assignment error fixes |
| 3.0 | Keith Bosworth | April 28, 2019 | System overview revisions, minor edits |
| 3.1 | Keith Bosworth | April 28, 2019 | Finalize |

**Table of Contents**

[System Overview 4](#_1fob9te)

Glossary 9

[Activities](#_3znysh7) 9

[Features](#_2et92p0) 10

[Demo 1 Features (03/25/2019)](#_tyjcwt) 10

[Demo 2 Features (04/08/2019)](#_3dy6vkm) 11

[Demo 3 Features (04/22/2019)](#_1t3h5sf) 12

[Demo 4 Features (04/29/2019)](#_4d34og8) 13

[Tasks 1](#_2s8eyo1)4

[Setup: Software and Hardware 1](#_17dp8vu)4

[Documentation 1](#_3rdcrjn)4

[Research 1](#_26in1rg)5

[Asset Development 1](#_lnxbz9)6

[UX Development](#_35nkun2) 18

[Game Logic Development](#_1ksv4uv) 19

[Networking](#_44sinio) 20

[Authentication](#_2jxsxqh) 21

[AI Development](#_z337ya) 22

[Testing and Verification](#_3j2qqm3) 23

[Bug Fixing](#_1y810tw) 27

[Platform](#_4i7ojhp) 31

[Optimization](#_2xcytpi) 32

[Schedule – Grouped by Feature](#_1ci93xb) 34

[Schedule – Grouped by Activity](#_3whwml4) 35

[Development Environment 3](#_2bn6wsx)6

[Version Control 3](#_qsh70q)6

[References 3](#_1pxezwc)6

## System Overview

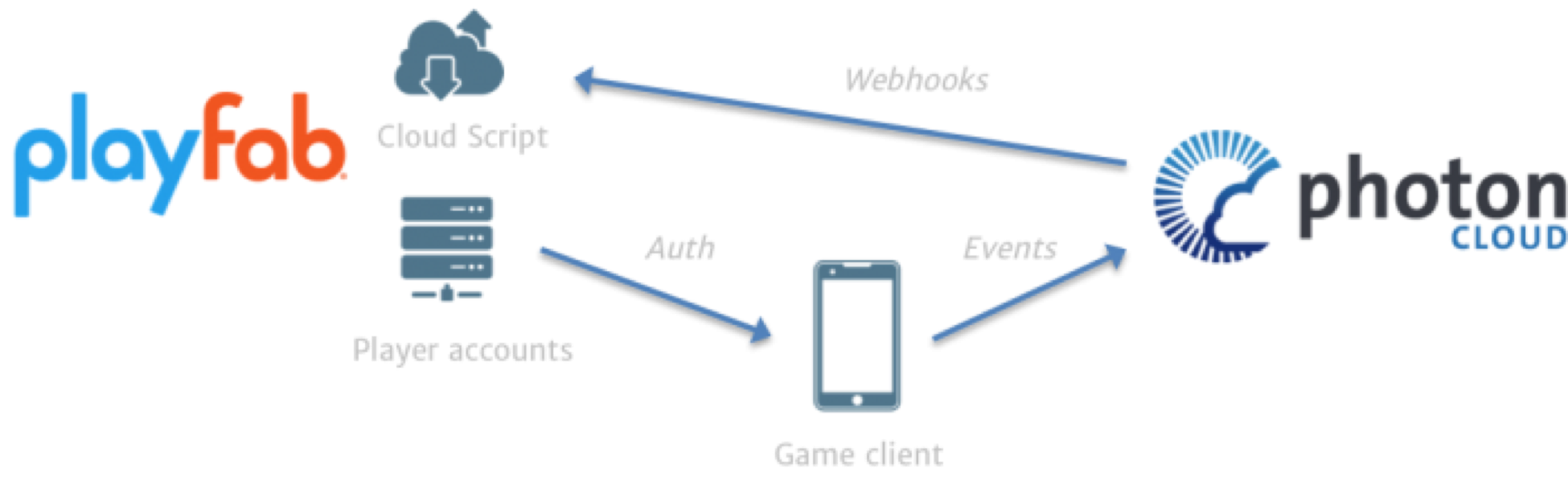
This section provides an overview of the Armament project: an original, networked, multiplayer, team-based video game built for PC, Mac, iOS, and Android platforms. Armament is built with the Unity game engine and API, Photon Networking servers and API, and PlayFab backend database and API.

Gameplay will consist of two stages played in succession: first, an *Armament* stage where players gather weapons and resources, and subsequently, a *Battle* stage where combatants fight for control of the arena using the resources they’ve acquired. Original sounds and art will be created for the project in addition to existing assets.

Armament will be designed and driven with the Unity engine and API, which contains numerous scripts and libraries that provide abstraction for the low-level details of physics rendering, graphics processing, animation, A.I, platform-specific builds, and system analytics.,

Player information is stored in a database provided by PlayFab. Users can register accounts and authenticate from both PC and mobile devices. Once logged in, players will see the Launcher where they can choose to play a game, or they can check the statistics stored in the database, which may be accessed through the leaderboard. If they choose to play, their statistics during that game will be updated to the leaderboard. Players can also add friends through the Launcher, which will allow them to invite those friends to private games. This feature gives players the ability to stay in touch with players they enjoyed playing with.

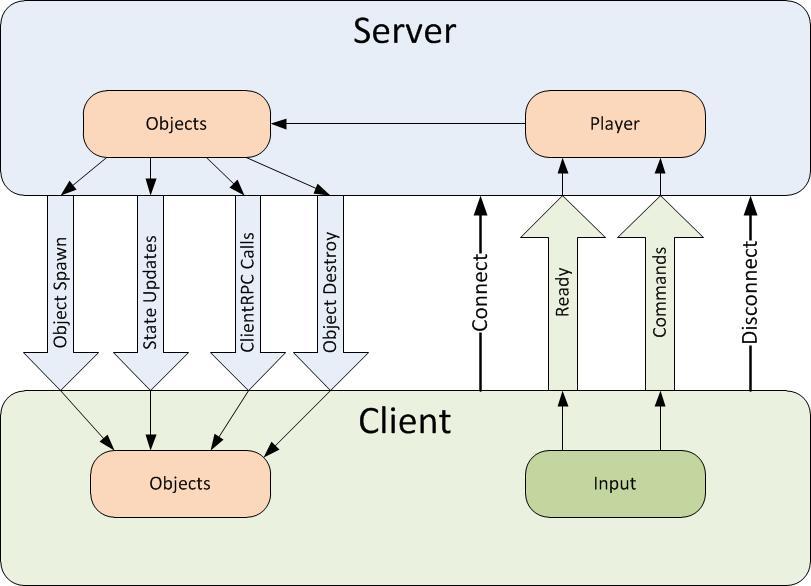
The interaction between Unity clients, the Photon cloud, and PlayFab can be seen below in the abstraction provided by **Figure 1.**

****

**Figure 1.** General overview of Unity client integration with Photon and PlayFab from a mobile perspective (similar for PC).

To play the role of the serving or host device, one client will assume the role of master client, which will synchronize remote clients by broadcasting RPC through Photon servers. A master client is a specialized type of local client that synchronizes remote clients using Photon servers as a means of communication, which sends and receives changes in specific game state based on client input. This implementation is headless in that it only passes RPC calls and choice data through Photon servers, as opposed to processing graphics and physics data within the Photon server space.

In order for Armament to synchronize across every connected device, copies of each object created in the game will exist in memory on both the master and remote clients. The master client device assumes the role of the authority to keep track of changes made to various GameObjects, as well as communicating those changes to the clients. The client/server relationship, RPC calls, data flow, GameObject storage, and state changes can be seen in **figure 2** below:



**Figure 2.** Overall data flow, object storage, and state changes from server (master client) to remote and local clients.

Data will be broadcasted from client to client using RCP and other modes of communication and data synchronization via the Photon server, and through the internet via UDP and TCP. The Photon networking framework provides a robust networking API built specifically for Unity projects in order to meet networking requirements. Clients send input, which is received by listening to various RPC events (for example OnMouseClick() called from within an RPC wrapper to listen to mouse clicks supplied from user input). The Photon Networking API provides abstraction of low-level socket code. **Figure 3** describes the user flow to begin playing online with other players.



**Figure 3.** User flow to begin playing Armament

When joining a game, players must use the in-game menu to connect to a Photon *name server*, which gives them access to a *master server*. Master servers are geographically located around the globe to provide low ping times to all clients, regardless of their location. Master clients will then place the clients in a *master server for matchmaking purposes.* . When a player finds a match they will communicate with other clients via a Game Server that is responsible for hosting the game room they are in.

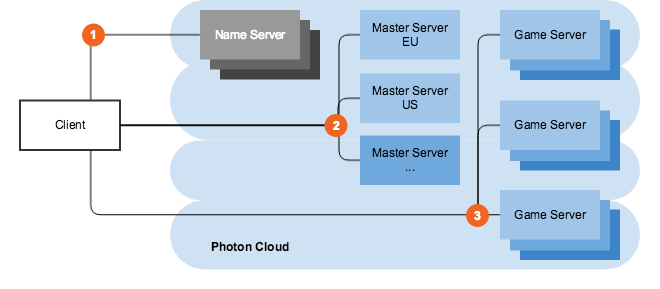
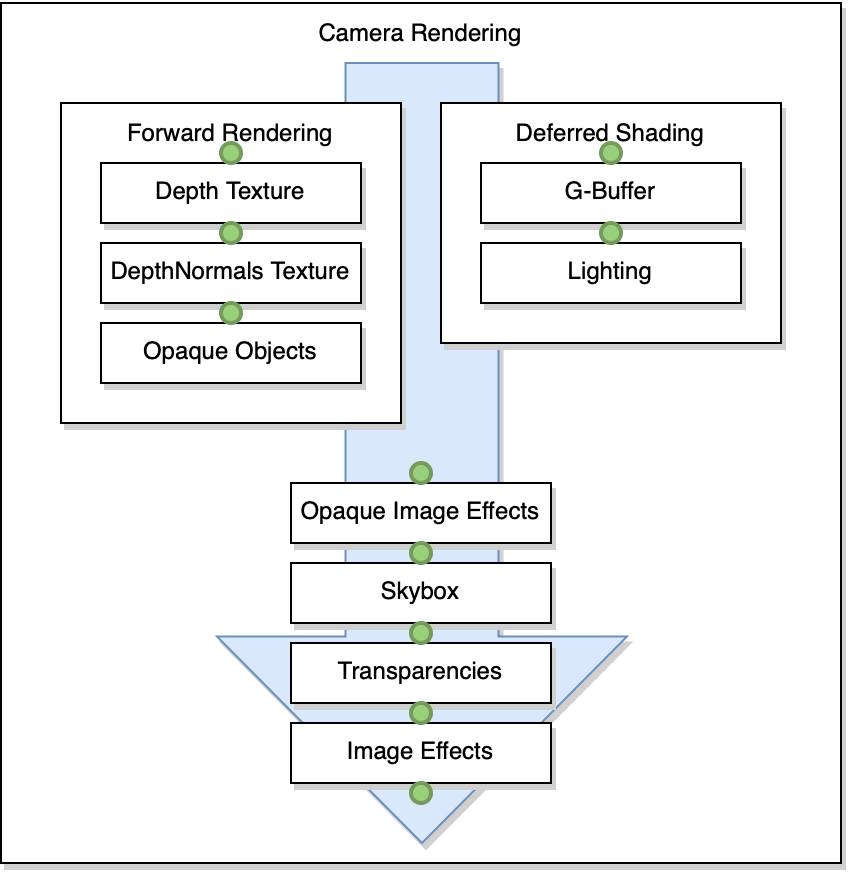


Figure 4.Master server and game server layout..

In the case that the host connection quality terminates or becomes suboptimal, the Photon API offers a host migration service which is called in order to move server identity to the next available device. Sending RPC through the Photon server adds one layer of security for clients in that the IPs of remote clients are managed by the Photon server instead of seen directly by the master.

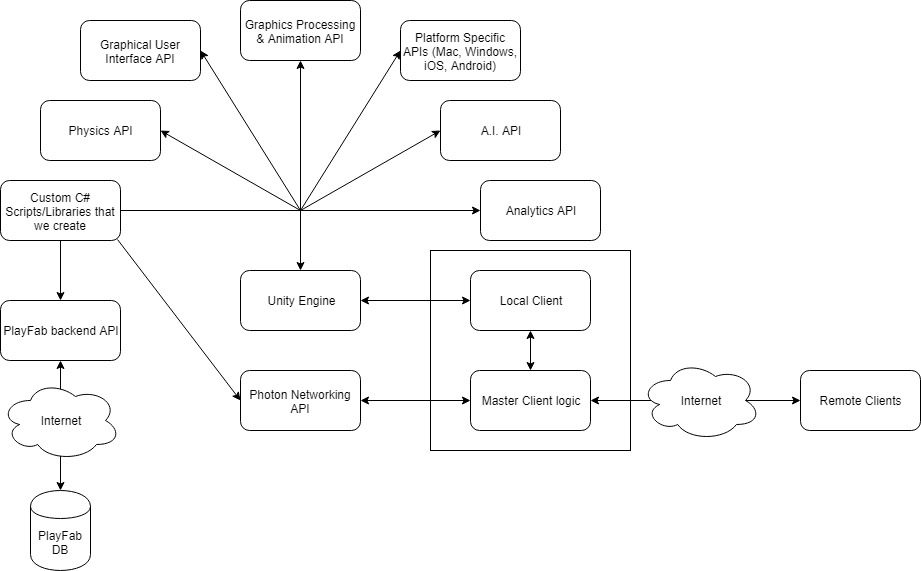
## The onus of graphics processing is placed on the client devices, which may flex the master client’s processing capabilities somewhat, but overall is not an impediment considering how picky our master client is in terms of choosing what to broadcast to clients through Photon servers. On the client side, the Unity engine will render a camera placed in the virtual environment, resulting in the data flow represented in figure 5 below. After graphics processing, certain state changes to the GUI will be communicated as attributes via various synchronization mechanisms provided by photon to all clients over the Internet.



**Figure 5**. Camera rendering data flow

## During gameplay, players have the option to toggle an A.I. controller. When this controller activated the game automatically takes control over the player avatar’s movement and actions. If the A.I. controller is activated during the *Armament* stage, the A.I. player will target (i.e., go to) known gun spawn points in order to pick up guns. Along the way, the A.I. may recognize that a gun, which it is not currently targeting for pickup and not yet picked up by another player, has come into view. . When it sees a valid gun target, it will run towards it and attempt to pick it up. During the Battle stage, the A.I. will wait for opponents (i.e., players on the other team) to come into view or shoot at it. Either event will trigger the A.I. player to target the opponent. The A.I. player will pursue its targeted opponent even if the player tries to run away. If the A.I. player gets the target in its crosshairs, it will immediately shoot..The A.I. player always calculates shortest path to its destination whether or not its target’s position changes. A high-level representation of the interlocking systems is described below in figure 4.

## System Block Diagram



**Figure 4.** A high-level view of the components in Armament.

## 

## Glossary

* **Master client**: the client that is designated to act as a pseudo server for all other clients. The master client becomes responsible for making decisions and coordinating actions that would typically be the responsibility of a server in a server-client model. Any client that joins a game room can potentially become the master client at some point. By default, the master client is chosen in the order of who entered the game room first.
* **Remote client**: all clients that are not currently the master client.
* Name server: the first server that every client contacts, which provides the list of available regions.
* **Master server**: every region has a completely separate master server for matchmaking.Game server: hosts game rooms
* **Launcher**: the first scene presented to the user upon starting the game. In this scene, a user has the ability to log in to their account, choose their gameplay options, and enter a game room to begin playing the game.

## Activities

Activities describe the type of work being accomplished for a given task. The list below summarizes every category of activity that Armament tasks may be classified as.

* *Setup*: downloading and installing development environment software.
* *Research*: gaining a deeper understanding of unknowns.
* *Asset Development*: acquiring or designing and developing unity assets.
* *Authentication*: implementing user authentication features using secure 3rd party APIs.
* *Game Logic Development*: developing rules for how the game should flow.
* *Networking*: allowing multiple computers to send and receive game data.
* *AI Development*: implementing a machine learning library to train and integrate an AI player.
* *Documentation*: producing documents describing the game and the work to be done.
* *Platform*: building the unity project for multiple platforms.
* *Integration*: merging branches in github as we go.
* *Testing & Verification*: comparing requirements with performance. One part of testing will entail exploration within the virtual environment from the perspective of the user. The other part will consist of automated batches of tests.
* *Debugging*: removing errors from existing code.
* *Optimization*: analyzing performance, and refactoring to meet speed and space requirements.

## Features

Features describe some characteristic or functionality within Armament. Ideally, a feature represents capability in the system that is demonstrable to a user.

### Demo 1 Features (03/25/2019)

1. Game executable can be accessed in mac or windows environment. The game environment can be successfully loaded.

Tasks

1. Basic game skeleton developed 02/18/2019 – 03/03/2019 - Jeremy
2. Game arena is accessible for play.

Tasks

1. Design game arena 03/04/2019 – 03/10/2019 - Ben
2. Develop game arena 03/11/2019 – 03/17/2019 - Andrew
3. Test game arena 03/18/2019 – 03/24/2019 - Andrew
4. Game weapons usable for play.

Tasks

1. Design weapons and items 03/04/2019 – 03/10/2019 - Keith
2. Develop weapons 03/11/2019 – 03/17/2019 - Alex
3. Test weapons 03/18/2019 – 03/24/2019 - Alex
4. Game items usable for play.

Tasks

1. Design weapons and items 03/04/2019 – 03/10/2019 - Keith
2. Develop items 03/11/2019 – 03/17/2019 - Jeremy
3. Test items 03/18/2019 – 03/24/2019 - Jeremy
4. HUD is usable, readable, and responsive to in-game actions.

Tasks

1. Design HUD 03/04/2019 – 03/10/2019 - Jeremy
2. Develop HUD 03/11/2019 – 03/17/2019 - Alex
3. Test HUD 03/18/2019 – 03/24/2019 - Ben

### Demo 2 Features (04/08/2019)

1. Game events synchronized across network for multiplayer sessions

Tasks

1. Establish Photon server 03/04/2019 – 03/10/2019 - Alex
2. Connect to server 03/11/2019 – 03/17/2019 - Ben
3. Synchronize Gameobjects on server 03/18/2019 – 03/24/2019 - Andrew
4. Test network synchronization 03/25/2019 – 03/31/2019 - Ben
5. User can create account, log in to account, and remove account all in-game.

Tasks

1. Setup Auth Database 03/11/2019 – 03/17/2019 - Jeremy
2. Implement CRUD for Users 03/18/2019 – 03/24/2019 - Alex
3. Implement Login/Logout 03/18/2019 – 03/24/2019 - Ben
4. Test CRUD 03/25/2019 – 03/31/2019 - Keith
5. Test Login/Logout 03/25/2019 – 03/31/2019 - Andrew

1. Main menu, pause menu, and additional submenus implemented.

Tasks

1. Design menu pages and flow 03/18/2019 – 03/24/2019 - Ben
2. Develop menu pages 03/25/2019 – 03/31/2019 - Andrew
3. Test menu pages 04/01/2019 – 04/07/2019 - Jeremy
4. User can create a game with static AI opponents.

Tasks

1. Implement Static AI 03/25/2019 – 03/31/2019 - Keith
2. Test Static AI 04/01/2019 – 04/07/2019 - Ben
3. Player statistics are synchronized after match and aggregated across matches.

Tasks

1. Implement Player Stat Sync 03/25/2019 – 03/31/2019 - Andrew
2. Test Player Stat Sync 04/01/2019 – 04/07/2019 - Andrew

### Demo 3 Features (04/22/2019)

1. Multiplayer arena has interactable barrier with dynamic timer. User can cause barrier to collapse and initiate a transition from collection to battle stage.

Tasks

1. Implement stage transition 03/25/2019 – 03/31/2019 - Keith
2. Implement barrier timer & collapse 04/01/2019 – 04/07/2019 - Jeremy
3. Test stage transition 04/01/2019 – 04/07/2019 - Alex
4. Test barrier 04/08/2019 – 04/14/2019 - Keith
5. Game has custom sound & music assets

Tasks

1. Develop sound & music assets 04/08/2019 – 04/14/2019 - Ben
2. Test sound & music assets 04/15/2019 – 04/21/2019 - Jeremy
3. User can create a game with basic AI opponents that will recognize the player and attempt to fight them.

Tasks

1. Implement Basic AI 04/08/2019 – 04/14/2019 - Jeremy
2. Test Basic AI 04/15/2019 – 04/21/2019 - Alex
3. User can use matchmaking to find ongoing game lobbies to join.

Tasks

1. Implement matchmaking 04/08/2019 – 04/14/2019 - Keith
2. Test matchmaking 04/15/2019 – 04/21/2019 - Ben

### Demo 4 Features (04/29/2019)

1. User can access game on a mobile (iOS, Android) platform. User has access to all existing functionality on mobile platform.

Tasks

1. Add default mobile configurations 04/15/2019 – 04/21/2019 - Ben
2. Finalize multi-platform builds 04/15/2019 – 04/21/2019 - Andrew
3. Test default config for mobile 04/22/2019 – 04/28/2019 - Keith
4. Optimize networking 04/22/2019 – 04/28/2019 - Alex
5. User can change graphics settings on computer platform, and/or benchmark their computer to produce auto-configured graphics settings.

Tasks

1. Add auto-benchmarking 04/15/2019 – 04/21/2019 - Alex
2. Finalize multi-platform builds 04/15/2019 – 04/21/2019 - Andrew
3. Optimize graphics 04/22/2019 – 04/28/2019 - Ben
4. User can create game with intelligent AI opponents that will act similar to a human player.

Tasks

1. Implement Smart AI 04/22/2019 – 04/28/2019 - Alex

## Tasks

Tasks describe a medium sized (roughly one week) portion of work in a given activity that also works towards the completion of a feature.

### Setup: Software and Hardware

PC with an internet connection (Mac, Windows)

Effort: Very Low

Mobile device (iOS, Android)

Effort: Very Low

Unity Engine & API

Effort: Very Low

Unity Testing Suite

Effort: Medium

Visual Studio

Effort: Low

Photon Services (Server Hosting)

Effort: Medium

Playfab (Authentication)

Effort: Low

Audio/Visual Software

Effort: High

### Documentation

Proposal Document

Effort: Medium

ETA: Week 3 - 02/01/2019

Responsible individual: Ben

Requirements Document

Effort: Medium

ETA: Week 4 - 02/08/2019

Responsible individual: Jeremy

Schedule Document

Effort: High

ETA: Week 5 - 02/15/2019

Responsible individual: Andrew

Design Document 1

Effort: High

ETA: Week 6 - 02/22/2019

Responsible individual: Alex

Design Document 2

Effort: Medium

ETA: Week 7 - 03/01/2019

Responsible individual: Keith

Weekly Status – Team

Effort: Low

ETA: Weeks 8-15

Responsible individual: Andrew

Weekly Status – Individual

Effort: Low

ETA: Weeks 8-15

Responsible individual: N/A

User Manual

Effort: Medium

ETA: Week 11 – 03/29/2019

Responsible individual: Ben

Test Procedures

Effort: Medium

ETA: Week 12 - 04/05/2019

Responsible individual: Jeremy

Test Report

Effort: Low

ETA: Week 15 – 04/26/2019

Responsible individual: Andrew

### Research

Explore Unity Features and API

Effort: Medium

ETA: Week 6 – 02/24/2019

Responsible individual: N/A

Predecessor tasks:

* + - N/A

Successor tasks:

* + - Build Basic Game Skeleton in Unity

Research Authentication (Firebase or Alternative)

Effort: Low

ETA: Week 6 – 02/24/2019

Responsible individual: N/A

Predecessor tasks:

* + - N/A

Successor tasks:

* + - Setup Player Authentication Database

Research Photon Multiplayer Networking APIs/Hosting Services

Effort: Medium

ETA: Week 6 – 02/24/2019

Responsible individual: N/A

Predecessor tasks:

* + - N/A

Successor tasks:

* + - Establish Photon Server

Build Basic Game Skeleton in Unity

Effort: Low-Medium

ETA: Week 7 – 03/03/2019

Responsible individual: Jeremy

Predecessor tasks:

* + - Explore Unity Features and API

Successor tasks:

* + - Testing

### Asset Development

Select and Import Premade Assets for Testing

Effort: Very Low

ETA: Week 6 – 02/24/2019

Responsible individual: Alex

Predecessor tasks:

* + - N/A

Successor tasks:

* Tuning the look & feel of the game towards something visually pleasing
* Custom animations for graphic models that trigger upon input

Design Game Arena

Effort: Low

ETA: Week 8 – 03/10/2019

Responsible individual: Ben

Predecessor tasks:

* N/A

Successor tasks:

* Develop Game Arena

Develop Game Arena

Effort: Medium

ETA: Week 9 – 03/17/2019

Responsible individual: Andrew

Predecessor tasks:

* Design Game Arena

Successor tasks:

* N/A

Design Items and Weapons

Effort: Medium

ETA: Week 6 – 02/24/2019

Responsible individual: Keith

Predecessor tasks:

* N/A

Successor tasks:

* Develop Item Assets
* Develop Weapon Assets

Develop Item Assets

Effort: Medium

ETA: Week 7 – 03/03/2019

Responsible individual: Jeremy

Predecessor tasks:

* Design Items and Weapons

Successor tasks:

* N/A

Develop Weapon Assets

Effort: Medium

ETA: Week 7 – 03/03/2019

Responsible individual: Alex

Predecessor tasks:

Design Items and Weapons

Successor tasks:

* N/A

Develop Sound and Music Assets

Effort: Medium

ETA: Week 13 – 04/14/2019

Responsible individual: Ben

Predecessor tasks:

* N/A

Successor tasks:

* N/A

### UX Development

Design a Control Scheme

Effort: Low

ETA: Week 6 – 02/24/2019

Responsible individual: Andrew

Predecessor tasks:

* N/A

Successor tasks:

* Develop Control Scheme

Develop Control Scheme

Effort: Medium

ETA: Week 7 – 03/03/2019

Responsible individual: Keith

Predecessor tasks:

* Design a Control Scheme

Successor tasks:

* N/A

Design a HUD (Heads Up Display)

Effort: Low

ETA: Week 8 – 03/10/2019

Responsible individual: Jeremy

Predecessor tasks:

* N/A

Successor tasks:

* Develop a HUD (Heads Up Display)

Develop a HUD (Heads Up Display)

Effort: Medium

ETA: Week 9 – 03/17/2019

Responsible individual: Alex

Predecessor tasks:

* Design a HUD (Heads Up Display)

Successor tasks:

* Design Menu Flow and Pages

Design Menu Flow and Pages

Effort: Low

ETA: Week 10 – 03/24/2019

Responsible individual: Ben

Predecessor tasks:

* Develop a HUD (Heads Up Display)

Successor tasks:

* Develop Menus

Develop Menus

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Andrew

Predecessor tasks:

* Design Menu Flow and Pages

Successor tasks:

* N/A

### Game Logic Development

Implement Stage Transition

Effort: High

ETA: Week 11 – 03/31/2019

Responsible individual: Keith

Predecessor tasks:

* N/A

Successor tasks:

* Implement Barrier Timer and Collapse Override

Implement Barrier Timer and Collapse Override

Effort: Medium

ETA: Week 12 – 04/07/2019

Responsible individual: Jeremy

Predecessor tasks:

* Implement Stage Transition

Successor tasks:

* N/A

### Networking

Establish Photon Server

Effort: Medium

ETA: Week 8 – 03/10/2019

Responsible individual: Alex

Predecessor tasks:

* N/A

Successor tasks:

* Successfully connect to remote server

Successfully Connect to Remote Server

Effort: Medium

ETA: Week 9 – 03/17/2019

Responsible individual: Ben

Predecessor tasks:

* Establish Photon Server

Successor tasks:

* Synchronize GameObjects Across Network

Synchronize GameObjects Across Network

Effort: High

ETA: Week 10 – 03/24/2019

Responsible individual: Andrew

Predecessor tasks:

* Successfully Connect to Remote Server

Successor tasks:

* Implement Matchmaking

Implement Matchmaking

Effort: High

ETA: Week 13 – 04/14/2019

Responsible individual: Keith

Predecessor tasks:

* Synchronize GameObjects Across Network

Successor tasks:

* N/A

### Authentication

Setup Player Authentication Database

Effort: Medium

ETA: Week 9 – 03/17/2019

Responsible individual: Jeremy

Predecessor tasks:

* Establish Photon Server

Successor tasks:

* Implement CRUD for Users
* Implement Login/Logout Functionality
* Implement Player Stat Updates

Implement CRUD for Users

Effort: Low

ETA: Week 10 – 03/24/2019

Responsible individual: Alex

Predecessor tasks:

* Setup Player Authentication Database

Successor tasks:

* N/A

Implement Login/Logout Functionality

Effort: Low

ETA: Week 10 – 03/24/2019

Responsible individual: Ben

Predecessor tasks:

* Setup Player Authentication Database

Successor tasks:

* N/A

Implement Player Stat Updates

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Andrew

Predecessor tasks:

* Setup Player Authentication Database

Successor tasks:

* N/A

### AI Development

Implement Static AI

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Keith

Predecessor tasks:

* N/A

Successor tasks:

* Implement Dumb AI

Implement Basic AI

Effort: High

ETA: Week 13 – 04/14/2019

Responsible individual: Jeremy

Predecessor tasks:

* Implement Static AI

Successor tasks:

* Implement Smart AI

Implement Smart AI

Effort: Very High

ETA: Week 15 – 04/28/2019

Responsible individual: Alex

Predecessor tasks:

* Implement Dumb AI

Successor tasks:

* N/A

### Testing and Verification

Test HUD

Effort: Medium

ETA: Week 10 – 03/24/2019

Responsible individual: Ben

Predecessor tasks:

* Develop HUD

Successor tasks:

* Bugfix HUD

Test Arena

Effort: Medium

ETA: Week 10 – 03/24/2019

Responsible individual: Andrew

Predecessor tasks:

* Develop Game Arena

Successor tasks:

* Bugfix Arena

Test Controls

Effort: Low

ETA: Week 10 – 03/24/2019

Responsible individual: Keith

Predecessor tasks:

* Develop Control Scheme

Successor tasks:

* Bugfix Controls

Test Items

Effort: Medium

ETA: Week 10 – 03/24/2019

Responsible individual: Jeremy

Predecessor tasks:

* Develop Item Assets

Successor tasks:

* Bugfix Items

Test Weapons

Effort: Medium

ETA: Week 10 – 03/24/2019

Responsible individual: Alex

Predecessor tasks:

* Develop Weapon Assets

Successor tasks:

* Bugfix Weapons

Test Network Sync

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Ben

Predecessor tasks:

* Synchronize GameObjects Across Network

Successor tasks:

* Bugfix Network Synchronization

Test Login/Logout

Effort: Very Low

ETA: Week 11 – 03/31/2019

Responsible individual: Andrew

Predecessor tasks:

* Implement Login/Logout Functionality

Successor tasks:

* Bugfix Login/Logout

Test CRUD

Effort: Low

ETA: Week 11 – 03/31/2019

Responsible individual: Keith

Predecessor tasks:

* Implement CRUD For Users

Successor tasks:

* Bugfix CRUD

Test Menus

Effort: Low

ETA: Week 12 – 04/07/2019

Responsible individual: Jeremy

Predecessor tasks:

* Develop Menus

Successor tasks:

* Bugfix Menus

Test Stage Transition

Effort: Medium

ETA: Week 12 – 04/07/2019

Responsible individual: Alex

Predecessor tasks:

* Implement Stage Transition

Successor tasks:

* Bugfix Stage Transition

Test Static AI

Effort: Medium

ETA: Week 12 – 04/07/2019

Responsible individual: Ben

Predecessor tasks:

* Implement Static AI

Successor tasks:

* Bugfix Static AI

Test Player Stats Synchronization

Effort: Medium

ETA: Week 12 – 04/07/2019

Responsible individual: Andrew

Predecessor tasks:

* Implement Player Stat Updates

Successor tasks:

* Bugfix Player Stats Synchronization

Test Barrier

Effort: Medium

ETA: Week 13 – 04/14/2019

Responsible individual: Keith

Predecessor tasks:

* Implement Barrier Timer and Collapse Override

Successor tasks:

* Bugfix Barrier

Test Sound and Music Assets

Effort: Low

ETA: Week 14 – 04/21/2019

Responsible individual: Jeremy

Predecessor tasks:

* Develop Sound and Music Assets

Successor tasks:

* Bugfix Sound and Music Assets

Test Basic AI

Effort: High

ETA: Week 14 – 04/21/2019

Responsible individual: Alex

Predecessor tasks:

* Implement Dumb AI

Successor tasks:

* Bugfix Dumb AI

Test Matchmaking

Effort: Medium

ETA: Week 14 – 04/21/2019

Responsible individual: Ben

Predecessor tasks:

* Implement Matchmaking

Successor tasks:

* Bugfix Matchmaking

Test Auto Benchmark

Effort: Medium

ETA: Week 15 – 04/28/2019

Responsible individual: Andrew

Predecessor tasks:

* Benchmarking/Auto Graphics Settings

Successor tasks:

* Bugfix Auto Benchmark

Test Default Configuration for Mobile

Effort: Medium

ETA: Week 15 – 04/28/2019

Responsible individual: Keith

Predecessor tasks:

* Default Configuration for Mobile Platform

Successor tasks:

* Bugfix Default Configuration for Mobile

### Bug Fixing

Bugfix HUD

Effort: Low

ETA: Week 11 – 03/31/2019

Responsible individual: Jeremy

Predecessor tasks:

* Test HUD

Successor tasks:

* N/A

Bugfix Arena

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Alex

Predecessor tasks:

* Test Arena

Successor tasks:

* N/A

Bugfix Controls

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Ben

Predecessor tasks:

* Test Controls

Successor tasks:

* N/A

Bugfix Items

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Andrew

Predecessor tasks:

* Test Items

Successor tasks:

* N/A

Bugfix Weapons

Effort: Medium

ETA: Week 11 – 03/31/2019

Responsible individual: Keith

Predecessor tasks:

* Test Weapons

Successor tasks:

* N/A

Bugfix Network Synchronization

Effort: Medium

ETA: Week 12 – 04/07/2019

Responsible individual: Jeremy

Predecessor tasks:

* Test Network Synchronization

Successor tasks:

* N/A

Bugfix Login/Logout

Effort: Low

ETA: Week 12 – 04/07/2019

Responsible individual: Alex

Predecessor tasks:

* Test Login/Logout

Successor tasks:

* N/A

Bugfix CRUD

Effort: Low

ETA: Week 12 – 04/07/2019

Responsible individual: Ben

Predecessor tasks:

* Test CRUD

Successor tasks:

* N/A

Bugfix Menus

Effort: Low

ETA: Week 13 – 04/14/2019

Responsible individual: Andrew

Predecessor tasks:

* Test Menus

Successor tasks:

* N/A

Bugfix Stage Transition

Effort: Low

ETA: Week 13 – 04/14/2019

Responsible individual: Keith

Predecessor tasks:

* Test Stage Transition

Successor tasks:

* N/A

Bugfix Static AI

Effort: Medium

ETA: Week 13 – 04/14/2019

Responsible individual: Jeremy

Predecessor tasks:

* Test Static AI

Successor tasks:

* N/A

Bugfix Player Stats Synchronization

Effort: Medium

ETA: Week 13 – 04/14/2019

Responsible individual: Alex

Predecessor tasks:

* Test Player Stats Synchronization

Successor tasks:

* N/A

Bugfix Barrier

Effort: Medium

ETA: Week 14 – 04/21/2019

Responsible individual: Ben

Predecessor tasks:

* Test Barrier

Successor tasks:

* N/A

Bugfix Sound and Music Assets

Effort: Medium

ETA: Week 15 – 04/28/2019

Responsible individual: Andrew

Predecessor tasks:

* Test Sound and Music Assets

Successor tasks:

* N/A

Bugfix Basic AI

Effort: High

ETA: Week 15 – 04/28/2019

Responsible individual: Keith

Predecessor tasks:

* Test Dumb AI

Successor tasks:

* N/A

Bugfix Matchmaking

Effort: Medium

ETA: Week 15 – 04/28/2019

Responsible individual: Jeremy

Predecessor tasks:

* Test Matchmaking

Successor tasks:

* N/A

### Platform

Benchmarking/Auto Graphics Settings

Effort: Medium

ETA: Week 14 – 04/21/2019

Responsible individual: Alex

Predecessor tasks:

* N/A

Successor tasks:

* Test Auto Benchmark

Default Configuration for Mobile Platform

Effort: Medium

ETA: Week 14 – 04/21/2019

Responsible individual: Ben

Predecessor tasks:

* N/A

Successor tasks:

* Test Default Configuration for Mobile

Finalize Build for Multi-Platform

Effort: Medium

ETA: Week 14 – 04/21/2019

Responsible individual: Andrew

Predecessor tasks:

* N/A

Successor tasks:

* N/A

### Optimization

Optimize Assets

Effort: High

ETA: Week 14 – 04/21/2019

Responsible individual: Jeremy

Predecessor tasks:

* N/A

Successor tasks:

* N/A

Optimize Networking

Effort: High

ETA: Week 14 – 04/21/2019

Responsible individual: Alex

Predecessor tasks:

* N/A

Successor tasks:

* N/A

Optimize Graphics

Effort: Medium

ETA: Week 14 – 04/21/2019

Responsible individual: Ben

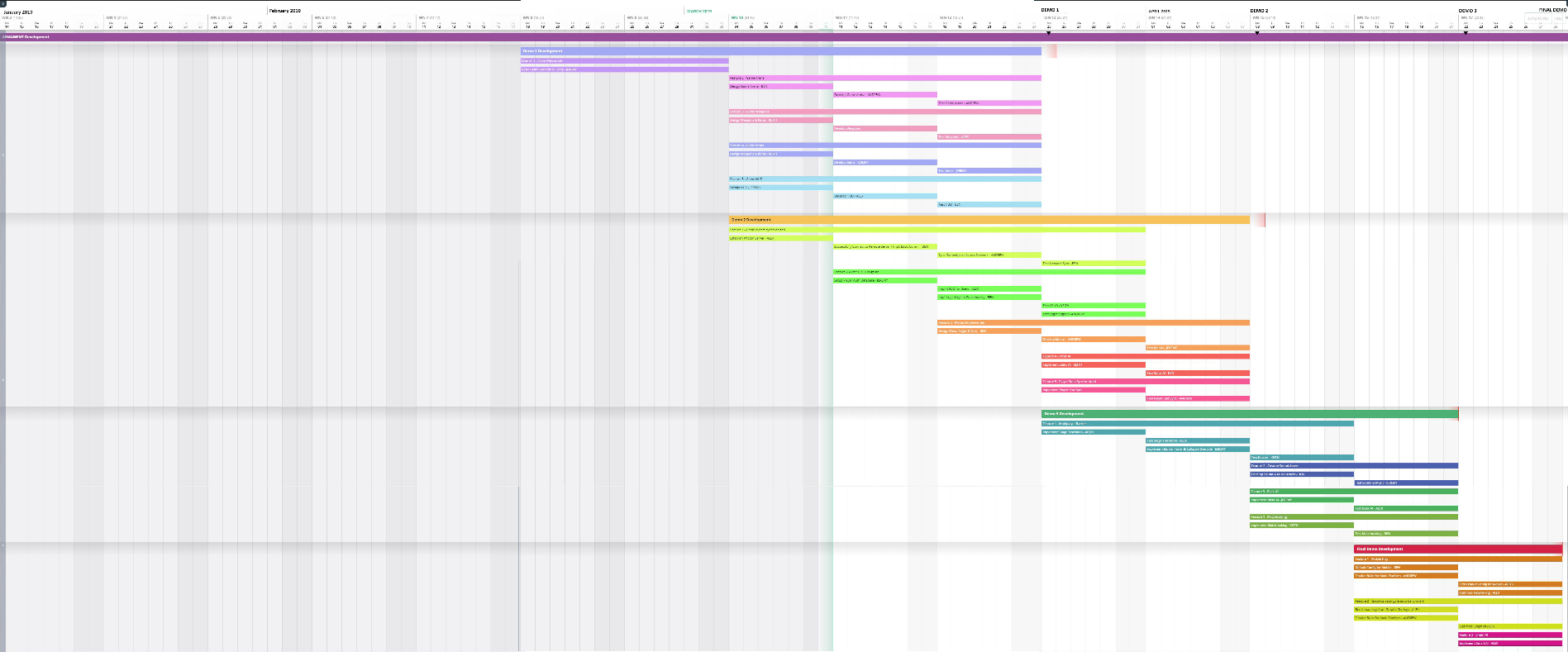
Predecessor tasks:

* N/A

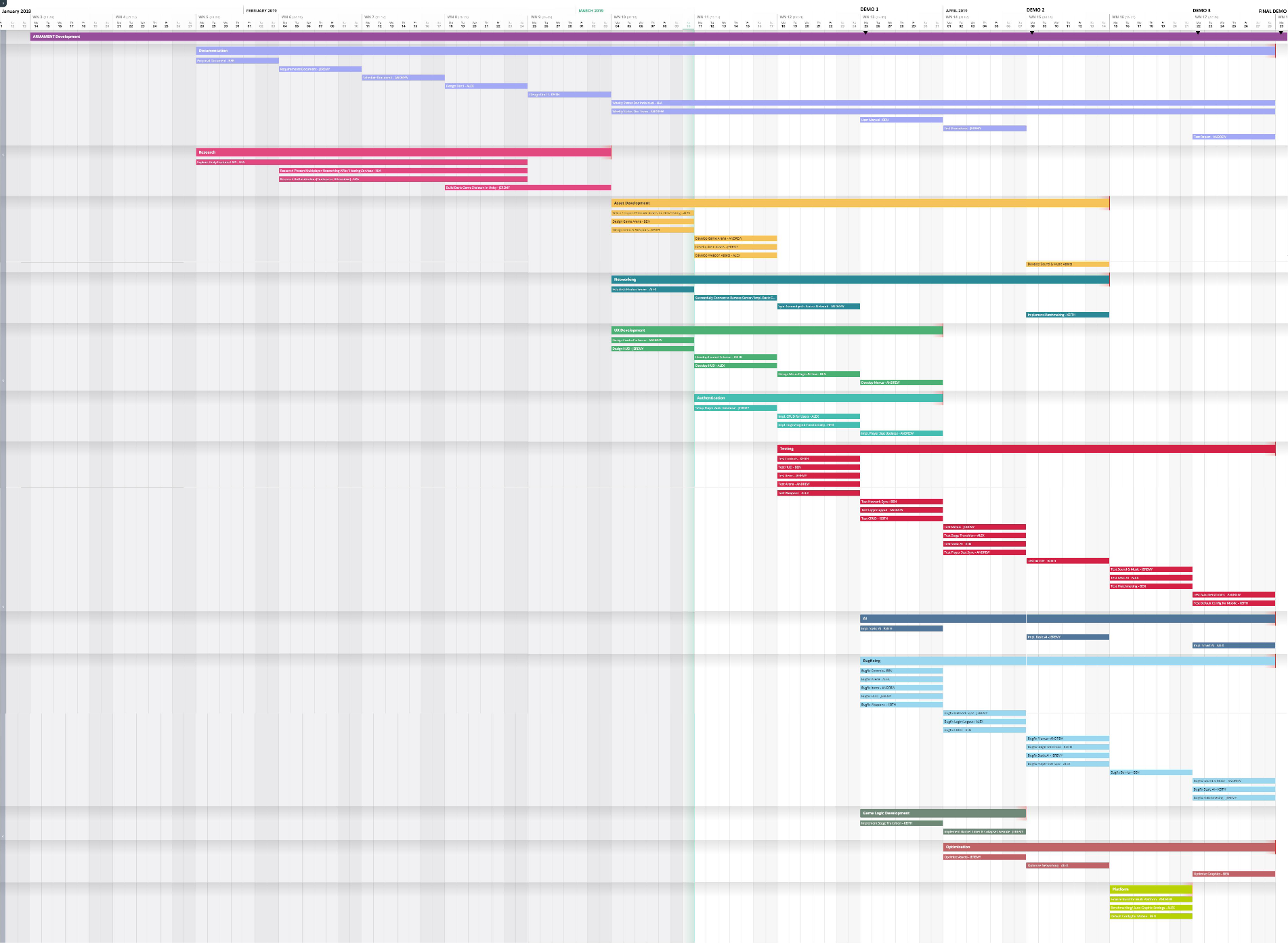
Successor tasks:

* N/A

## Schedule – Grouped by Feature



## Schedule – Grouped by Activity



## Development Environment

* PC with an internet connection (Mac, Windows)
* Mobile device (iOS, Android)
* Unity Engine
* Unity API
* Unity testing suite
* Visual studio
* Photon services (server hosting)
* Google Firebase (authentication)
* Optional Audio/Visual Software:
  + Maya (for visual assets)
  + Photoshop (for visual assets)
  + GIMP (for visual assets)
  + Audacity (for audio assets)
  + Logic Pro (for audio assets)
  + Serum VST (for audio assets)
  + Analog sound “‘studio” (for audio assets)

## Version Control

* git for version control
* GitHub for remote hosting of git repo
  + Repo located at https://github.com/molongjoe/CapstoneProject
* GitKraken for local repo management

Procedure: maintain one release-level branch, one development-level branch, and various downstream branches. Downstream branches will be merged often to the development branch in order to eliminate weakly-connected features.

## References

* Unity user Manual 2018.3 <https://docs.unity3d.com/Manual/index.html>
* Photon documentation <https://doc.photonengine.com/en-us/pun/v2/getting-started/pun-intro>
* Playfab documentation <https://api.playfab.com/docs/general-getting-started>
* Git branching model https://nvie.com/posts/a-successful-git-branching-model/