**4.1 Plan Introduction**

This Software Development Plan provides the details of the planned development for the *Heartwired* application, a 2D puzzle-platformer video game that users will be able to download and play on both Mac/PC. In developing the game, each member of the team will be programming features in weekly sprints before coming together and consolidating the best possible version of the product. The developers will be using Windows PC computers and Unreal Engine Blueprints throughout the development of the project, in teamwork with assets created and imported from Maya 3D Software.

**4.1.1 Project Deliverables**

* Project Proposal Document (9/20)
* Requirements Specification (9/27)
* Software Development Plan Document (10/11)
* Preliminary Design Review Presentations (11/01)
* Software Design Description Document (Architecture Section) (11/01)
* Software Design Description Document (Detailed Section) (11/15)
* Critical Design Review (CDR) Presentations (11/15)
* Test Plan Document (11/29)
* AlphaBeta Project Presentations (11/29)
  + For the Alpha Project Presentation, we will be showing a playable tutorial level and a work in progress first world.
  + Both of these levels will be testable.
  + Many of the assets in these levels may be temporary, as the animation production schedule permits.
* User’s Manual (12/06)
* User’s Manual Final Updates (12/15)
* Final Project Presentation (12/16)
  + We will be presenting the game tutorial and first world at the final project presentations.
  + Both of these levels will have been rigorously tested.
  + Most of the assets in these levels will be at their complete stage.
  + Users will be able to play through both of these levels at their leisure.
* Oral Status Reports
* Written Status Reports in Project SDF

**4.2 Project Resources**

**4.2.1 Hardware Resources**

| **Operating System** | **CPU Name** | **CPU Speed** | **# of CPU Cores** | **Amount of RAM** | **Name of GPU** | **Amount of VRAM** | **Purpose** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Windows | Intel Xeon | 2.60 GHz | 20 cores | 64.0 GB | NVIDIA Quadro M400 | 8912 MB | Development |
| Windows | Intel Core i5 | 2.30 GHz | 5 cores | 8.0 GB | Intel HD Graphics 520 | 8060 MB | Development |
| Windows | Intel Core i7 | 2.70 GHz | 7 cores | 16.0 GB | Intel HD Graphics 620 | 8260 MB | Development |
| Windows | Intel Core i7 | 2.50 GHz | 7 cores | 8.0 GB | NVIDIA GeForce 940MX | 6067 MB | Development |

**4.2.2 Software Resources**

| **Resource** | **Development** | **Execution** |
| --- | --- | --- |
| Unreal Engine Blueprints | ✓ | ✓ |
| Github | ✓ |  |

**4.3 Project Organization**

Our method of assigning roles in the development of *Heartwired* is different from the typical software development plan. Rather than splitting the programming of each ability by specifc members of the team, every developer will be working concurrently on the features. We recognize that many aspects of the game code rely closely on one another; therefore, every person on the project must be able to understand and create the game from the ground up to avoid confusion and reliance on any one team member.

We approached our development plan with snowballing complexity: Kevin Metelus, Ryan Taus, Sasha Dmitrieva, and Zachary Fitzpatrick will each begin with programming the smallest working version of the game. Once we have finished, we will consolidate our work and determine which approach was the most efficient and satisfying in gameplay. Then, we will each program the next round of features, and so on. This schedule of increasing complexity is detailed in the next section.

Each member also participates in designing puzzles for the game's levels. In short, every member of the team has similar responsibilities in the development process.

**4.4 Project Schedule**

Since Heartwired will be built upon in increasing complexity, the development of the game will be split into weekly sprints, starting with basic functionality that will grow into complex features interacting and relying on each other. These sprints are our best projection of the time that it will take to program these features into the game; if the time required for these tasks does not align with our schedule, we will adjust as follows with our three weeks of buffer space.

A detailed description of the sprint schedule follows:

Week 1:

* Ability to move and jump
* Collision
* Switching
* Camera transitioning between screens
* Menu system to load into menu

Week 2:

* Platforming: jump
* Follow at all points
* Permanently moving platforms
* Camera transition between screens and levels
* All menus

Week 3:

* Saving/Loading system
* Character follows unless told to hold
* Character switching
* Enemies present and detecting at a range and moving
* Begin adding in rough animations
* Pause Menu

Week 4:

* Collision is functional
* Linking animations
* Switches can be linked to certain types of environments
* Follow AI can sometimes detect when it cannot follow
* Health system
* Bear attacking
* Preliminary hookshot

Week 5:

* Checkpoints
* HUD system
* All types of hookshot independently
* Bear charge
* Sneak
* Enemy AI complete without fine-tuning

Week 6:

* All animations imported
* All animations smoothly transition
* Hookshot toggling and choosing correct destination
* Hookshot rope
* Menu animations
* Marker when other avatar is on other screen
* Toggling characters when on other screen

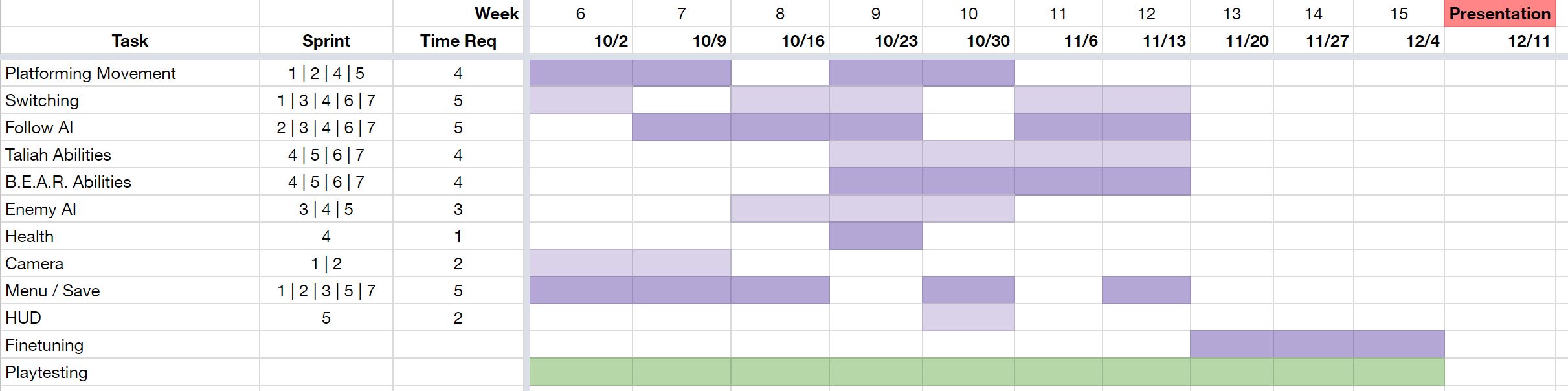
Week 7:

* Saving completed levels and not reloading them as they are
* Ability cooldowns
* Locking out switching on ability use
* World select

Understanding the importance of testing our game early and often, we have allocated time for playtesting throughout the semester. For more details on the abilities we will be tackling in our sprints, please see our [Requirements Specifications Document.](https://github.com/sashadmitrieva96/cyberwitch/blob/master/docs/requirements-specifications.md)

**4.4.1 PERT / GANTT Chart**

The GANTT Chart provides a visual representation of the pacing we are expecting to follow in programming each of these unique features into the game.

[](https://raw.githubusercontent.com/sashadmitrieva96/cyberwitch/master/docs/ganttchart.JPG)

**4.4.2 Task / Resource Table**

All of the programming for Heartwired will be done in the Unreal Engine Blueprints system:

| **Task** | **Hardware** | **Software** |
| --- | --- | --- |
| Platforming Movement | Windows PC | Unreal Blueprints |
| Switching | Windows PC | Unreal Blueprints |
| Follow AI | Windows PC | Unreal Blueprints |
| Taliah Abilities | Windows PC | Unreal Blueprints |
| B.E.A.R. Abilities | Windows PC | Unreal Blueprints |
| Enemy AI | Windows PC | Unreal Blueprints |
| Health | Windows PC | Unreal Blueprints |
| Camera | Windows PC | Unreal Blueprints |
| Menu / Save | Windows PC | Unreal Blueprints |
| HUD | Windows PC | Unreal Blueprints |

Very good job!

5 out of 5

Nothing else needed for this deliverable – good to go!