**A.R.M.A.  
(Augmented Reality Mobile Application)**

**Detailed Design**

**COP 4331C, Fall 2015**

Team Name: Project Pals

Team Members:

* Eric Peralli - Eric.peralli@gmail.com
* Connor Heckman – Connor.heckman@me.com
* Clayton Cuteri – Cuteri.clayton@knights.ucf.edu

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Who | Comment |
| V0.0 | 10/04/15 | Connor Heckman | Original Draft |

**Contents of this Document**

SECTION 1 - Detailed Design Issues

* Reusability
* Reliability
* Maintainability
* Testability
* Performance
* Portability
* Security
* Safety

SECTION 2 - Detailed Design Information

* Class Diagrams
* Sequence Diagram
* Activity Diagram
* Detailed Command Sequence

SECTION 3 - Trace of Requirements to Design

* GUI
* Game Environment
* Themes
* Network
* Website

**SECTION 1 - Detailed Design Issues**

**Reusability**  
  
- The completed source code will be stored in the group git. This way, if a group member wishes to reuse code from the project on a future game design it will be readily available to them.

- We have also decided to publish the application’s source code on the developer site at the conclusion of the project. This way anyone with interest in creating a virtual reality application can reference it. Users of the game will also be able to update it and tweak its functionality if they wish.

**Reliability**

- Our personal server prototype was able to adequately handle being accessed by several users simultaneously. While we are not able to test the possible strain of a large number of players accessing our database, we assume our application will have a relatively small user base.

- We have chosen to utilize a personal server because it is a cheaper option to us than a third party server, and we wanted the networking experience for educational purposes.

- After testing a prototype scene of a user/enemy interaction in unity, we are confident that our ray casting method will provide an accurate means of targeting on screen enemies.

**Maintainability**

- Allowing our application’s source code to be available to our user base on long will allow for others to maintain the application if it is still being used.

**Testability**

- We will be able to test many aspects of our application as individual “scenes” in the unity engine, this will hopefully cut down on the time it will take to troubleshooting customization issues.

- We have be fortunate in that several of our friends have volunteered to help us test all the various test cases.

**Performance**

- We are able to manage our active enemies in the game environment through the use of unity’s rotational transform feature.

- This functionality allows us to have enemies active in the game environment, or in a particular “scene” through the use of separate rotational transforms. Enemies will only be displayed if their rotational transform overlaps with the user’s. Or in other words, enemies will only be displayed if the user is facing the direction they are in.

- By keeping enemies active even when off-screen, we create a sense of tension and chaotic action for the user, which enhances their gaming experience.

**Portability**

- We considered porting the game to apple devices as well as Android devices, but in the end decided to avoid a cross platform implementation due to the high cost of registering as an apple developer.

**Security**

- After conducting a brief survey of friends who we know to be mobile gamers, we observed that none of them believed they would deny the permissions our application will need to function properly.

- This is encouraging to hear, and assures us that the permissions our application needs to be granted by each user are reasonable in the eyes of seasoned mobile application gamers.

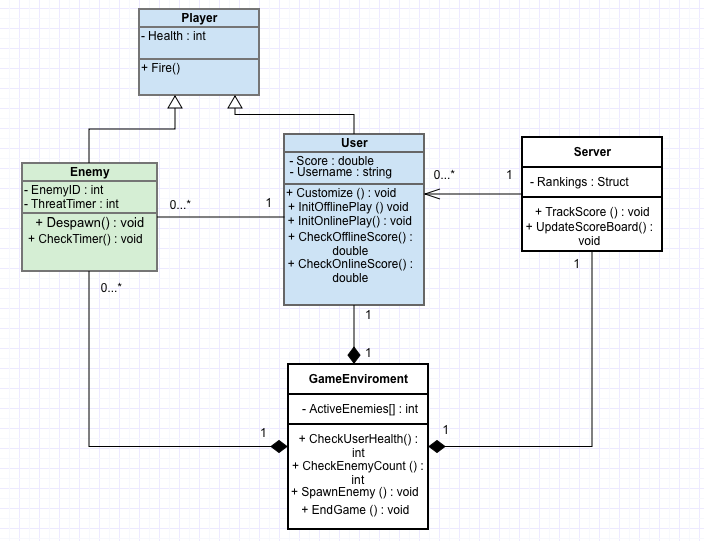
**Safety**

- We have concluded that our application will actually be safer to use than most mobile phone applications, since the “background” of our game is the user’s actual physical environment. This way the user will be aware of hazards ahead of them even when in game.

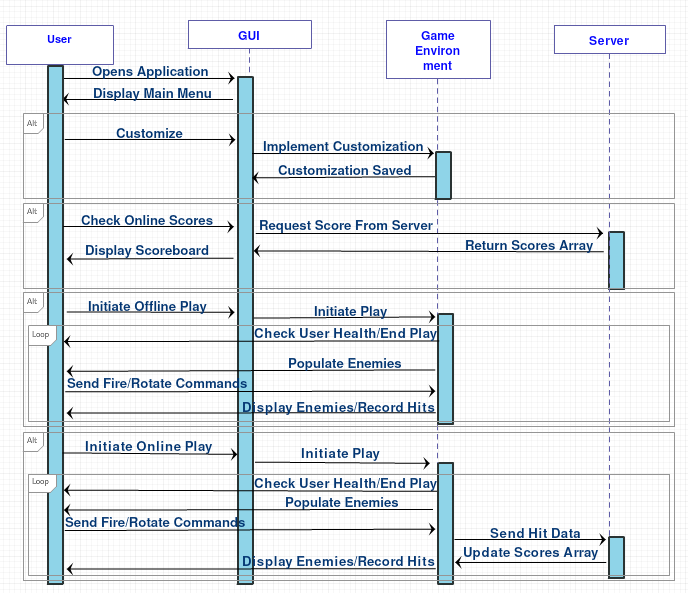
- Despite this assurance, we have decided to implement a warning at the application’s start that will caution users from attempting to play the game while in a potentially hazardous environment.

**SECTION 2 - Detailed Design Information**

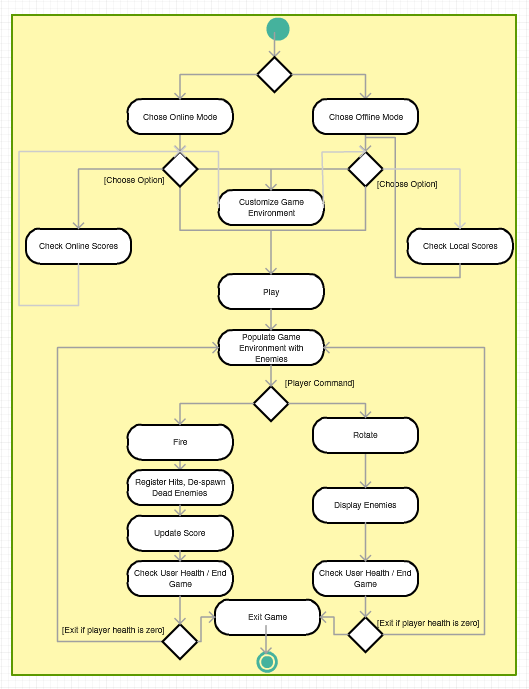
**Class Diagram**



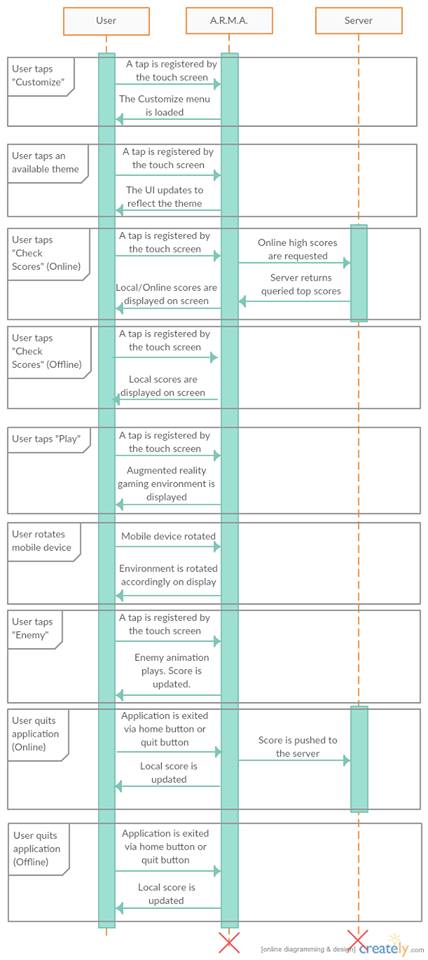
**Sequence Diagram**

****

**Activity Diagram**

****

**Detailed Command Sequence Diagram**

****

**SECTION 3 – Trace of Requirements to Design**

|  |  |
| --- | --- |
| **Requirement** | **Trace to Design** |
| GUI | The user interacts with our application’s main menu through the unity GUI. Here they can command the application to customize their gaming experience, interact with the server to receive their top scores, or initiate a gameplay environment. |
| Game Environment | Composed of two layers, the drawn elements on the camera representing the active enemies and aesthetic effects. And the reality layer of the camera API. While the user is in the gameplay environment, they are engaged in what is essentially a while loop that concludes when their health hits zero. While in this loop the send a command to the gameplay environment (Fire or Rotate) and the gameplay environment responds (De-spawn enemy, Display enemies in that direction). |
| Themes | The themes are comprised of several different databases of various PNGs and other design elements that can be mapped to the game environment to create a customized experience. The themes are selected using the GUI but saved by the Gameplay Environment. |
| Network | The server represents the Network, a personal computer interacting with players over the internet that updates users score while gameplay unfolds. The Gameplay environment calls its method, UpdateScore whenever the user registers a hit. |