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

Detailed Design

COP 4331C, Fall 2015

Team Name: Project Pals

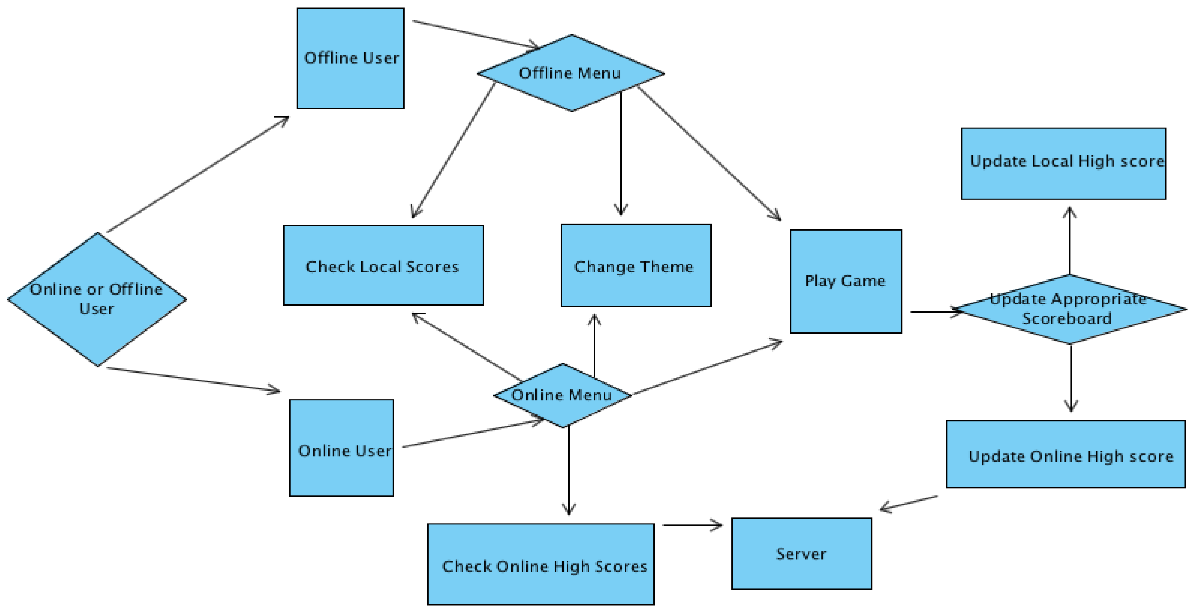
Team Members:

- 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	10/28/15	Clayton Cuteri	Initial Draft
V1.0	10/29/15	Connor Heckman	Proofread

Contents of this Document

High-level Architecture



- Online or Offline User – This is anybody who downloads the game and opens it to play the game.

- Offline User – This user does not have any way of connecting to the Internet to play.
- Offline Menu – This menu will be displayed to an Offline User.
- Online User – This user has an Internet connection when playing a game.
- Online Menu – This menu will be displayed to an Online User.
- Check Local Scores – This will be where the user accesses their personal top scores.
- Check Online Scores – This is where the user accesses the online top scores.
- Change Theme – All users will have the ability to choose between multiple themes to play the same game with a different aesthetic feel to it.
- Play Game – All users will launch the game to play with their desired theme.
- Update Appropriate Scoreboard – If the user is offline, the app will update the local scoreboard. If online, the app will update the overall scoreboard.
- Update Local High score – Local high scoreboard is updated when a score is beat.
- Update Online High score – Overall online high scoreboard is update when a score is beat.
- Server – Holds all of the High Score information and is accessed when someone needs the Online High Scoreboard

---

<u>Design Issues</u>

<u>Reliability</u>

A few issues regarding reliability could arise.

One issue could be upon updating the scoreboard. The online scoreboard may face issues when multiple online users access it simultaneously. We will implement a personal server for our application but if this prototype server is unable to handle the strain of multiple online players we will pursue a third party option for our networking needs.

Another reliability issue would arise with the accuracy of a tap. We need to assure that every portion of an enemy's sprite is considered a valid target area for the user.

<u>Reusability</u>

We will be creating our code from scratch so we have no concerns regarding reusability. The Source code created for the project has the potential to be reused for future game designs.

<u>Maintainability</u>

We are not worried about future updates thus after the conclusion of the class. We are not concerned with the maintainability associated with our application.

Testability

Testability will be relatively easy because our project does not have many variations or functions. The most time consuming portion of the testing phase will be assuring that all aspects of different theme implementations carry over to the gameplay environment. We will need to make sure that all themes respond to user touch in the same fashion.

Performance

The biggest concern we will encounter with performance is if the game will keep track of enemies off screen. When a user rotates the screen away from an active enemy, that enemy must remain a threat to the user's health bar even though they are not displayed on the screen currently. Thus, our gaming environment is larger than the scope of the user's camera.

Portability

There will be no portability issues. We are designing the game for Android only.

Security

Our application will require our user's to provide permissions to access their camera (to display the game environment) as well as their accelerometer (to monitor the rotation of their camera). These permissions are essential to our application and so it will be impossible to play the game without granting them. We will talk with our gamer friends to see if they would be comfortable with granting these permissions to an Android application and thus get a better understanding of the portion of users who may chose to refuse these permissions.

Safety

Since the gameplay environment reflects your surroundings, users will actually be less likely to run into real world objects. The user may actually play this game while walking anywhere and not worry about looking up every minute.

A safety issue could arise if a user turns too quickly to look behind him. This jerking action may not only be dangerous for his ligaments and personal well-being, but for others standing beside them.

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

- 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.

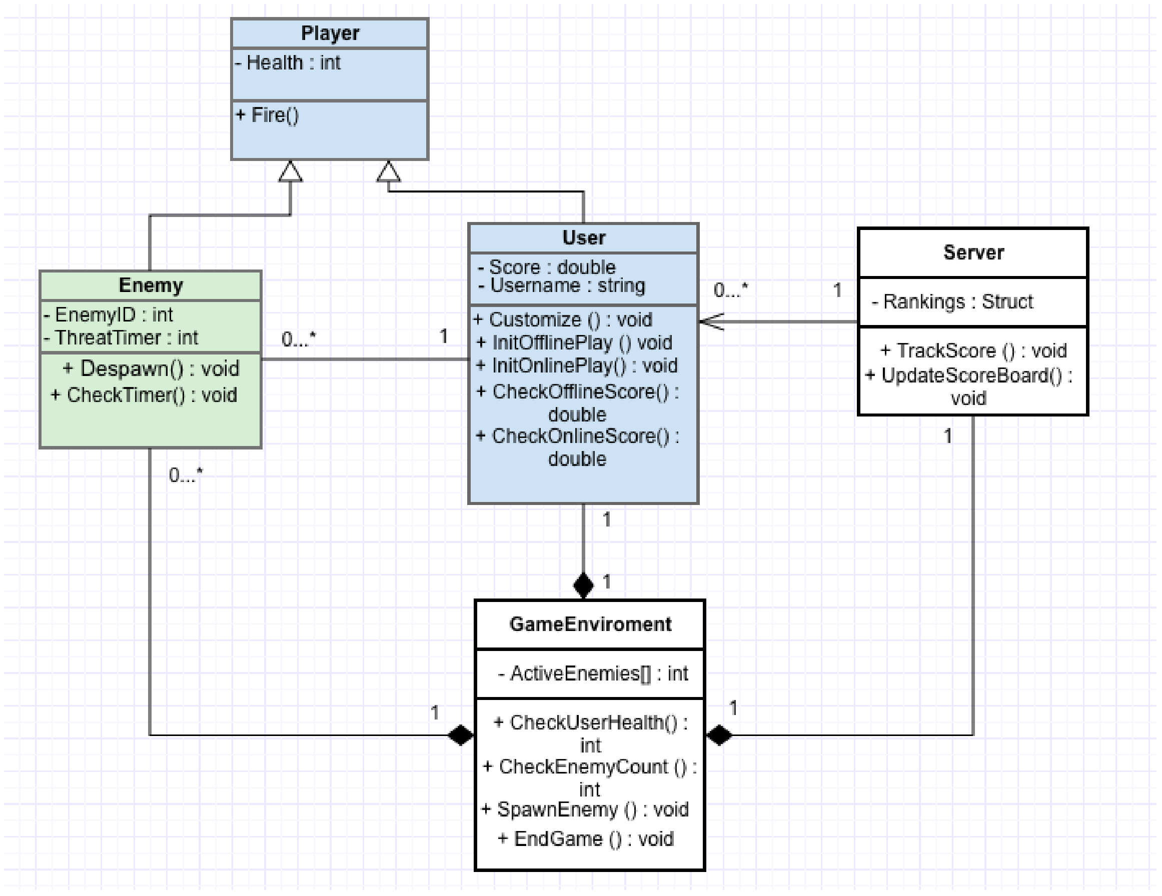
Safety

- 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.

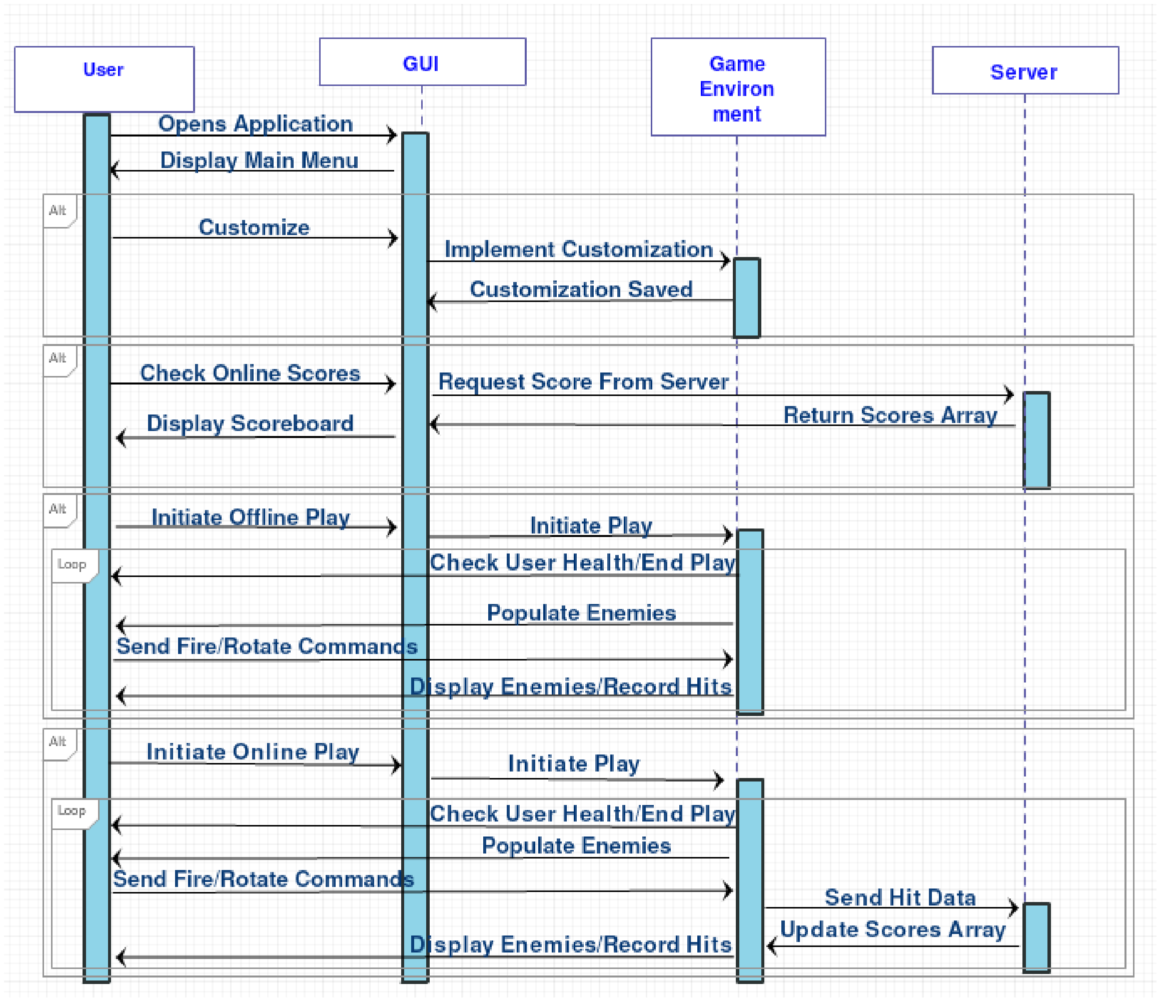
- Despite this assurance, we have decided to implement a warning at the application's start that will caution users from attempting to play the game while in a potentially hazardous environment.

SECTION 2 - Detailed Design Information

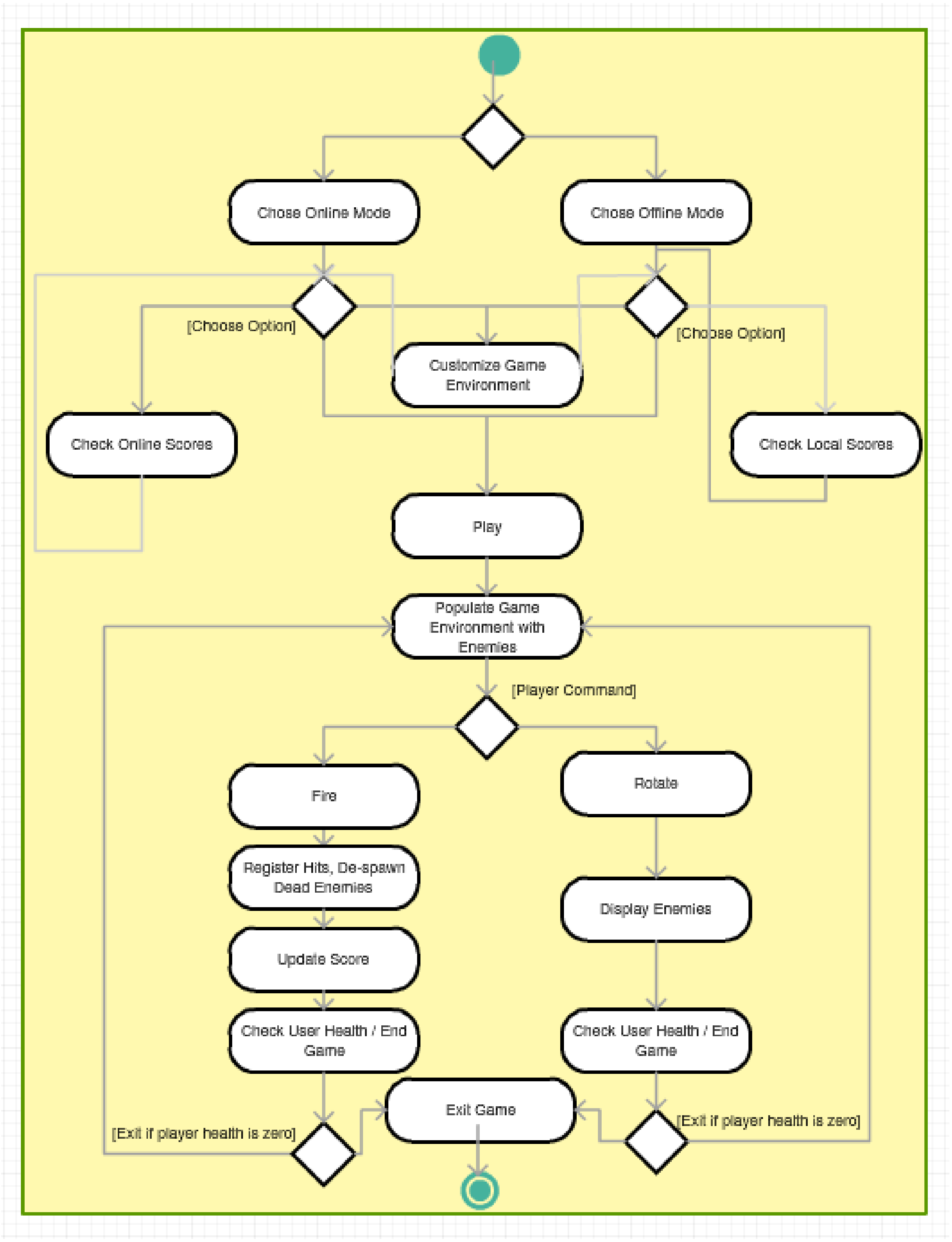
Class Diagram



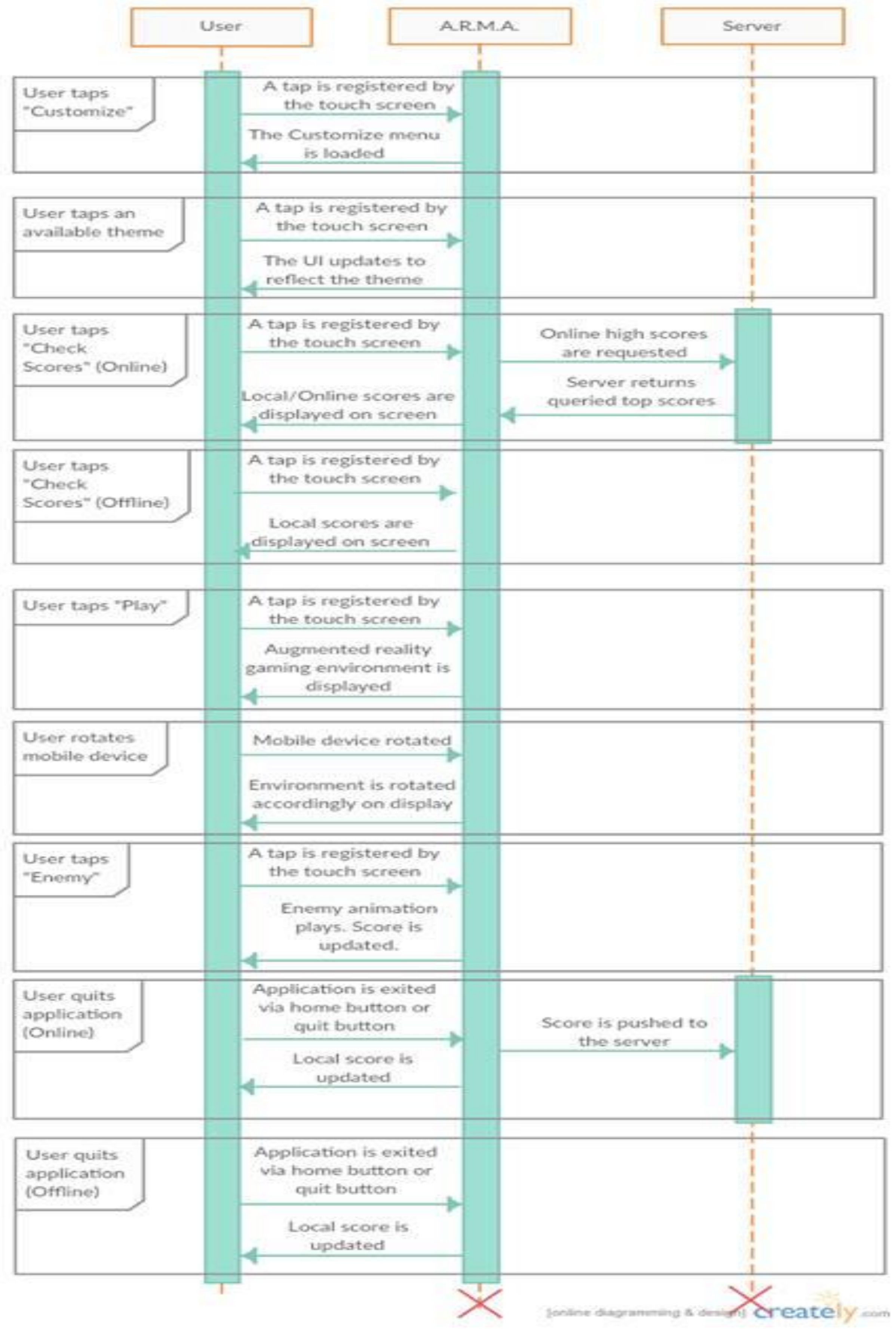
Sequence Diagram



Activity Diagram


Chose Online Mode
Chose Offline Mode
[Choose Option]
[Choose Option]
Customize Game Environment
Check Online Scores
Check Local Scores
Play
Populate Game Environment with Enemies
[Player Command]
Fire
Rotate
Register Hits, De-spawn Dead Enemies
Display Enemies
Update Score
Check User Health / End Game
Check User Health / End Game
Exit Game
[Exit if player health is zero]
[Exit if player health is zero]

Detailed Command Sequence Diagram


User
A.R.M.A.
Server
User taps "Customize"
A tap is registered by the touch screen
The Customize menu is loaded
User taps an available theme
A tap is registered by the touch screen
The UI updates to reflect the theme
User taps "Check Scores" (Online)
A tap is registered by the touch screen
Online high scores are requested
Server returns queried top scores
Local/Online scores are displayed on screen
User taps "Check Scores" (Offline)
A tap is registered by the touch screen
Local scores are displayed on screen
User taps "Play"
A tap is registered by the touch screen
Augmented reality gaming environment is displayed
User rotates mobile device
Mobile device rotated
Environment is rotated accordingly on display
User taps "Enemy"
A tap is registered by the touch screen
Enemy animation plays. Score is updated.
User quits application (Online)
Application is exited via home button or quit button
Score is pushed to the server
Local score is updated
User quits application (Offline)
Application is exited via home button or quit button
Local score is updated
[online diagramming & design] creately.com

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.