The Legend of Harambe: Final Report

Team 20
20/20 Entertainment
Keegan Sotebeer, Ashna Guliani, Cory Cranford,
Souneth Ly, Andres Salinas, Ross Blassingame

December 14th, 2016

I. Project Reflection:

A. Project Management

In this project we used an array of software methods and tools to construct our product. Our methodology used in this project was the Agile method. This method worked well for the team because we were able to utilize Trello to help us keep track of the different pieces and requirements of the project, while allowing each member to work on those pieces assigned to them on their own. Our weekly stand-ups on Mondays allowed us to communicate about the things we had done, the things we were about to start working on for that week, and ask for any support we needed at that time with our areas of the project. The sprints we had were slightly more difficult to keep up with, because all of us had other classes and work to be doing, and we weren't always able to finish everything we wanted to or said we would in a sprint. But again, our Trello board helped us to keep track of the things we couldn't get done and had to move to the next sprint.

B. Team Communication

Our team utilized Slack as a way to communicate on a more day-to-day basis since it was not feasible for us to have a daily stand-up; this is why we had a weekly stand-up in person. But Slack was able to bridge that gap in in-person communication. Slack is a great tool and worked really well for us because we all downloaded the app on our phones so that we could all get notifications specifically from Slack and know that it had to do with our project, as opposed to something like facebook messenger where the notification could be from any one of our friends, and we might be more likely to ignore it.

C. Game Development Platform

We also used Unity Game developer as the platform for building the project. Unity is very in depth and definitely comes with a learning curve. But our team had a couple mini-hackathons where we all delved into doing tutorials on Unity and practiced with it so that we could overcome the learning curve as quickly as possible.

One thing we have to learn through Unity is the game animations. There are three steps to make a game animations. The first step is coding. In order to tell the player to do an animation, we need a variable that contain a reference to the animator component. Also, we need to define what is ground in the game. From there we can sets a grounded parameter of the aminator, which will trigger the animation. Furthermore, we can set jump and double jump animation from grounded parameter. After we done coding, the next step is to open the animation and animator tab on Unity. The animation tab is where we created all the animation by using the graphic of the player and set up the animation frame. In order to make the game animation smooth, there will be multiple graphics insert into the animation frame. Insert multiple graphics into the frame is one of toughest thing we have to do, because the sample number have to match the graphics and the speed of the game in order to have a smooth animation. The last thing we need to do is creating animation transitions. We can create animation transition in the animator tab. In the animator tab, there will be a list of all the animations we made in the animation tab. Also, there is an any state animation, which we use transition from running animation to a jumping animation. To make sure animations transit to the next animations when a certain thing happen in the game. For example, if we want to move from a running animation to a jumping animation, we have to create a parameter in the code and Unity. With the parameter, we can use a condition statement to when the running animation transit to the jumping animation and back to the running animation. The animator tab can get messy and confused because there three animations for each of the characters we have in the game. With these three step we created an awesome Harambe animations.

D. Version Control

For our version control, we used Github. Github worked well for us as keeping a remote area for all of our pieces to be dumped into so that we could all access it, as well as the TA's, however, it was not very helpful with our game development platform because Unity didn't integrate with it very well. Our Unity files did not merge successfully with each other due to the meta data and libraries created each time we build the project in Unity. Due to this, we often had to redo parts we had been working on multiple times. This led to a strict protocol for using Github. Whoever had researched and practiced integrating their piece on the outdated version of the project and was ready to add it to the final copy would communicate so with the team, pull the final version from the git, integrate their parts, by either importing the scripts and assigning them correctly or by reimplementing their piece manually, and then upload the new final version to the github for everyone to grab as their template for their new pieces. Then the next person could do their part.

There was also a very specific way of pulling and pushing the working project to and from Github. We couldn't have our Unity platform saving to our file within our local Git repo, so we had to save our work to a directory on our local machines, then when we were finished, we copied the whole game file, deleted the old game file from the local git repo, pasted the updated file, and then pushed that to the remote repo. Due to the added strain of the problems between Unity and Github, this might not have been the best pairing for version control, but we definitely made do.

D. Database and Backend

As far as the database and backend portion of the game, the tool used was phpMyAdmin. This is a tool that allows users to store a mySQL database on a web server and access it using php scripts. The web server used was hosted and maintained by bplaced, which is a tool that offers free webspace hosting along with easy database integration using phpMyAdmin. Overall, these tools worked well for the purposes of this project. The mySQL database was nice because of its familiarity and due to the fact that we are limiting the size of the database by only storing the top 100 scores. It was also pretty easy to integrate the online database with Unity. This was done by creating a new C sharp class and attaching it to the leaderboard scene. The script included functions to post data to the mySQL database and retrieve the top 5 scores for display on the screen. It was really convenient because the Unity scripting API offers a WWW class that is capable of taking a url and performing HTTP GET and POST requests over the server. By pointing the server to one of the two different php scripts stored on the bplaced server and using the WWW class, it was not that difficult to update the database or retrieve scores from it. Then, specifying what scores to retrieve or how to order scores, etc. was simply a matter of updating the mySQL code contained within the php scripts.

Additionally, this setup was useful for testing and debugging because it allowed us to access our database on a web browser version of phpMyAdmin where we could easily see the contents of the database. This was convenient in debugging the php script that puts new scores in the database, which wasn’t working for a while. Although this turned out to be a nice tool and provided the simplest solution for our purposes this semester, there are a lot of things to be done that would improve the functionality of the backend and give users a better experience. This is detailed in the Future Plans section further on in the report.

E. Photoshop

Photoshop was really helpful to make possible the graphics used on the game. It made it simple enough to make graphics in .png format, which were used as sprites on the Unity platform. The only problem with photoshop is that it is pixelated based, not the best for games, it doesn't make it feel smooth enough. A better option would have been Adobe Illustrator since it is vector based, meaning that it doesn't matter if you resize the image it will always maintain quality, and make it look smoother. For example, if you look carefully at Harambe in the game, you may notice that he is a tiny bit pixelated since he had to be resized, but if the image had been done in Illustrator that wouldn't have been a problem

II. Project Report:

A. Project Accomplishments

Overall, this was a very successful project and a lot of work was accomplished in a short amount of time, especially considering the limited experience with game development that the team began with. When starting this project, the goal was to create and endless runner sidescrolling iPhone game with customization features, custom animations and graphics, and an online leaderboard that connected all our users together. In the end, most of this was in fact accomplished. Below is a more detailed list of the accomplishments made by 20/20 Entertainment this semester.

1. Unity Gameplay Development
    a. An endless running environment was developed so that the user plays as Harambe, who can run forever if the player is good enough.
    b. Random generation was implemented such that bananas and lasers are generated randomly throughout the game. Getting a banana gives you points and hitting a laser ends the game.
    c. Randomly generated biomes with smooth transitions were also implemented so that Harambe is running through an endlessly changing environment.
    d. Physics scripts were written such that Harambe had the ability to double jump, making the game more interesting and more addicting.
2. Graphics and Animations
    a. Custom graphics of Harambe were created in photoshop.
    b. The graphics involved different Harambe characters such as “Donald Trump Harambe” and “Skate Harambe.”
    c. For each character, the graphics package also included an extensive set of animation frames.
    d. Animations were implemented in Unity to give Harambe smooth running, jumping, and dieing motions.
3. Main Menu, Settings, and Leaderboard
    a. A fully functioning main menu with access to settings, leaderboard, and game play was developed in Unity
    b. The settings page allows users to scroll through the different customizeable Harambe characters, such as “Donald Trump Harambe” and “Skate Harambe.”
    c. The leaderboard allows users to view the top 5 Legend of Harambe scores of all time by pulling data from the online database
    d. Upon dying, users are prompted to enter a username, which adds their score to the online leaderboard.

B. Outstanding Issues

One of the main outstanding issues stems from the online database that stores user's scores. Originally, the design was to force every user to have a unique username. When a user enters their name after dying, the database would be checked to make sure that the username they typed hasn't been taken already. On top of that, the online database would only store each user's top score. So, the database would only be updated if a user beat their personal top score. Time constraints limited the addition of this functionality to the game, as it would require additional php scripting and server communication to check whether or not a username was taken. Furthermore, it would require a method of storing a user's top score on their local machine. This is definitely possible, but there is not an obvious way of doing it in Unity. Currently, the game uses the old-school arcade-like leaderboard, where the user can type any name they want after a game and it will go into the database with their score. This is not a bad implementation, but it could be frustrating for users if they were to see that the top 50 scores were all "John."

Another outstanding issue is simply the frequency and placement of the randomly generated objects. Getting the optimal placements and occurrence frequency for a challenging, yet manageable game is a highly iterative process. Currently, the random generation is working fine, but some of the parameters could be tweaked in order to give users the best experience.

C. Future Plans

In terms of gameplay, the future plans for Legend of Harambe are to continue testing both within Unity and on different iPhones in order to perfectly tweak the random generation and jump parameters to give the game the best possible balance so that it is challenging enough to make it fun and addictive, but not too challenging to where it is hopelessly frustrating. On top of that, we plan to implement more character customization options since the current version only has three. This is a not an overly difficult task, it is something that simply takes a lot of time. The reason being that sprites for the new characters first have to be created in PhotoShop and then a graphics set has to be made that includes all of the different animation phases involved in running, jumping and dying. Once that is complete, the animation can be added to the game and the option can be added to the settings menu.

As far as the backend and database goes, the future plan is to implement some of the functionality detailed in the previous section that we didn't have time for. Specifically, that means forcing users to create a unique username and then storing that username along with their top score locally. This way, the online database will only consist of each user's top score and users will only have to create a username when they first download the game. This makes it less annoying for the user because they don't have to type their name every time they die, and it also offers a better comparison between users so that they can see who the top 5 players are, not just the top 5 scores, which could all be from the same user.

Our plans also include submitting our game to the Apple App Store in the coming weeks.