ASSIGNMENT #2

Instructor: Mr. Waqar Ahmad

Course: Software Engineering

Submission: Requirements Engineering

Project: DeadPool Rush

Group Number: 9

Group Members: BSCS14022

BSCS14037

BSCS14040

BSCS14061

**Introduction:**

It’s a simple 2.5D running game in which a super hero character is running to find the villain and save his girlfriend from him. The character will face obstacles and enemies to on his way, along with guns and power ups on the ground to pick up and kill the enemies. The perspective of the game will be similar to the one shown in the figure below:



The screen will have following things to interact with:

* Fire Button
* Jump Button
* Score
* Powerup time
* Weapon in hand
* Special Attack Button

The character will keep on running to the right and the goal is to go as far as possible while avoiding obstacles and killing enemies. To avoid obstacles user will use Jump button so character can jump and avoid obstacles. To kill enemies user can use Fire button to shoot a bullet and if that bullet will collide with enemy it will die and score will be added. Different powers up will be provided throughout the game which user can use for different purposes. The game will end when character will collide with a obstacle or enemy .

**Techniques that will be used for requirement elicitation:**

We will use Scenarios and Use cases for requirements elicitation as scenarios are subset of use cases, with scenario-based requirements elicitation, we query the stakeholders for the kinds of things they want to be able to do. We gave them much of scenarios and ask them about what they feel about those scenarios (like what could go wrong in that scenario and what could be improved) .Consider following scenario for illustration of the concept.

*When the main character collides with an obstacle or gets hit by shot fired by enemies the game is over and player doesn’t get a second life*.

We could ask stakeholders about the scenario and they can give feedback like game should be over when character collides with obstacle twice in a row or the character should be given a second chance based on some criteria. We will create a complete set of scenarios and we will which we will document the scenarios and their responses .We then map their feedback into a system specification; the specification is represented as a set of actors and use cases.

**Requirements Grouping:**

Functional Requirements:

1- The game will display a game scene to the user.  
2- The user will be able to quit at any time.  
3- The user will be able to start a new game at any time.  
4- The user will be able to pause its game state.  
5- The user will be able to change its sound state(on/off).  
6- All the power ups will be shown with icons.  
7- User will be able to see its power and game assets at any time.  
8- The user can make its player jump, sit, swing etc.  
9- When the game will end, user will be asked to play again or quit.

10- The game will display a 2.5d environment

Non- Functional Requirements:

1- The game should load quickly  
(Performance requirements) (Speed).  
2- The game must have high quality graphics.  
3- The game must not drop frames at run time.  
(Performance requirements)( Reliability)   
4- The minimum frame rate allowed will be 30fps.  
(Usability requirements)( Reliability)   
5- The average time between button clicked and its response should be less than .5 seconds.  
(Performance requirements) (Speed)  
6- The game will cover most of the Android devices, i.e. Android 4.0 and OSs after it.(Portability)  
7- The game will not use extra memory at run time until it’s needed to use some and when the game will over program will flush and free the occupied memory.  
(Space requirements) (Size)  
8- The game will be reliable and resistant to crash states.  
(Performance requirements) (Reliability)  
9- Game must save users achievements and progress in permanent memory.  
(Space requirements) (Size)

**Following are the lens we’ve identified in our game:**

The Lens of emotion:

In super hero movies, there’s always a villain who hurts the hero emotionally by hurting hero’s loved ones. As our game is based on the movie of the character, therefore we’re putting the same story in the game as of movie to emotionally touch the audiences the way movie did; rescuing superhero’s girlfriend from villain.

The lens of surprise:

The superhero of our game is going to save his girlfriend from the villain. Sadly, all villains have army of bad people that blocks superhero’s way to villain and the hero is almost alone to kill them all. For that, he must have a lot of guns and ammo. But, thanks to DeadPool’s bad memory, he lost his duffle bag(ammunition bag) in Dophinder’s taxi. So, in the game, to put the element of surprise for user, Dophinder’s taxi will pass by randomly and drop 1 of the DeadPool’s guns which can be picked up to kill more enemies than his swords. In addition to that, right after we’ve successfully built the basic functionality of game, we’ll put some side-kick in the game, who’ll randomly appear for some seconds and help DeadPool. Lastly, the enemies will appear from right of the screen from random points to enhance the element of surprise for the user.

The lens of fun:

As we know from the movie and comics, DeadPool is the most hilarious superhero ever! Due to the fact that he’s the only super-hero that can do 4th wall break (4th wall break is a scenario in comics where comic characters directly talks to the viewer/reader), whenever the hero dies in the game, we’ll put some audios of DeadPool mocking the player for his negligence that caused his death.

The lens of curiosity:

The element of curiosity will be handled in our game in a way that the user will curious about the type of the next gun he’s going to get. DeadPool has a lot of guns in his duffle bag, but who know which of the gun will fall out of Dophinder’s taxi?

**Requirements Validation:**

* Consistency check:
  + There are no conflicts in the requirement s yet. If we come across any in the process of development, we’ll deal with them according to the scenario then.
* Realism Check:
  + We can build all the requirements we’ve documented yet with the available technology. We have not come across anything that we cannot build with the available technology yet.
* Verifiability Checks:
  + Requirement reviews:
    - We will present the requirements to some random people and take account to what they understand and will generalize the requirements accordingly. We will also present our viewers a paper version of this game(will draw different scenes of game on paper to give them a visual idea).
  + Prototyping
    - We will make prototypes of our game incrementally, module by module.
  + Test-case Generation
    - Once we make a prototype, we’ll make test cases of each module we make (incremental) and will test them. For example:
      * Fire Module Test Case
      * Jump Test Case and so on..