**INTRODUCTION**

The game, “**Jack, The Savior!**” is a text-based game where the player needs to input certain text data to continue playing. The text data will be processed and according to the input further actions will decide by the game management system. The game is basically designed and developed using Unity game development platform and using C# for programming.

The text-based environment is made possible in game by creating questioning and answering session between the player and system. The player will have certain health while starting the game and will be reduced by certain amount when the player answered wrongly. The game will start by explaining game situation and then the player can start the game.

There are various approaches exists now for creating game applications, such as:

* Model View Controller (MVC)
* Model View Template (MVT)
* Model View View/Model (MVVM)

etc..

Each approach has their own strength and weakness towards the development. Here I am going to utilize Model View View/Model approach which is also known as MVVM architecture. More details regarding MVVM will be discussed in the later part.

The game is a 2-dimensional (2D) game which supports multiplayer facility. Multiple players can join the game and the winner will be those who finish the game first and with most health.

**MODEL-VIEW-VIEWMODEL ARCHITECTURE**

As the technology more emerge the need of effective and fast development process is inevitable. By making it possible it is also relevant to keep the system simple without being much complexity. To reduce the complexity of a system patterns are good and effective solution. Model-View-ViewModel or MVVM is one of the several patterns or approaches now using. Here the abstract flow is Model 🡨🡪ViewModel 🡨🡪View.

In traditional User Interface development, developer need to create a view using available window functionalities or similar process and will write all logical implementation. This makes the View large which then creates a strong dependency between UI and binding logic. This led to complex situation in working as team environment and keeping all code in one are leads to ineffective maintainability. This mainly happens because of the tight coupling between the view and logic which includes both business logic, event handling and data binding.

MVVM pattern usually takes advantages of device capabilities and thus it makes use of device memory to extend the application performance. Thus, it allows better user experience which then allows then to have the application on various devices which has varying screen size. MVVM enables the separation of graphical UI from business logic.

* Model: It represents the actual data or information that the application deals with. Here in the game the text which moves the story and game is stored using the game models.
* View: It is the most familiar part of any system. Here is where the end users really interact with. This visualizes data in a presentable way according to the nature or data and the way user needs it. The view has no knowledge or information about the model. It is fully controlled by a controller. It is where the player interacts with the game. Simple the game interface.
* ViewModel: It is what enables the view using the models available. It converts the data to presentable format instead of letting the model aware of user view. In game it is what which accepts the input from the user and then process it for further actions/moves.

**GAME DESCRIPTION**

“**Jack, The Savior!**” is a fun and best game for children since it doesn’t involve any violent actions or behaviors which will affect the mind of children. The questions designed for the game is similar to brain teaser which helps the children to think more and thus, they can develop their problem-solving skills.

The player has 2 option to play the game. Either the player can start a new game or join some random game. Joining some random game is like joining a game where other players are playing the game thus, those players may have more advantage than the later joined player since they joined earlier.

The game story is as follows.

Once there lived a bunny, Jack with his parents Will and Pink in a village very close to a jungle. In the deep jungle there also lived a fearsome monster in a castle who preyed on innocent animals. One day, Will and Pink along with their neighbor Tinku duck went to jungle in search of food and firewood. They couldn’t find enough food in the outer jungle, so they went deep into the forest. Suddenly they found a garden full of fruits and vegetables and they ran into the garden to pick some. They didn’t know that it was the monster’s garden. By that time the monster came back from his hunting. He found these three uninvited animals in his garden and got angry. He rushed to the garden and caught Will and Pick. Tinku somehow managed to escape from the monster and ran back to the village. He explained everything to Jack and told that only the ones who are blood related can go into the castle and save them. Jack took on oath that he will save his parents by any means.

The game starts here, the player will play the role of Jack. In-order to get inside of the castle successfully the player need to complete 5 levels. Each level will have certain questions to answer. Passing each level successfully will help the player move further inside of the castle and when all the levels are passed the player will be inside of the castle and the monster will release the parents. The player needs to complete all the tasks before he loses his all health.

When the player will start the game, 10 hearts are given as health. Each wrong answer in the game session will reduce 2 hearts. Hence 5 wrong answers will drain all the health and the player will lose the game.

The whole game is divided into 5 levels. Each level has 3 questions to answer. The difficulty level of each question will increase as the game progress.

The game consists of mainly 6 scenes including game room and certain dialog boxes to show information, warning, danger messages.

* Login scene
  + This scene is loaded first when the game application is loaded. It enables the user to login into the system.
* Register scene
  + This allows new users to create profile in the system.
* Dashboard scene
  + This provides the game story outline and options to start a new game or to join a random game.
* Character Interaction scene
  + This scene will load when the player starts playing the game where Jack and Tinku is exchanging their dialogues.
* Game Room
  + The actual game will run here in game room. The game room is where the game system interacts with the player.
* Co-player detail scene
  + This scene will provide the details of co-players, like their health and level details. The Co-player details scene will show when the player joined a random game and whenever the player needs he/she can open it through Game Room.

UI components used for designing the game are

* Image
* Text
* Input Field
* Scrollbar
* Buttons

Each component and buttons used in the design will be explained in the below storyboard.

The interactive session of the game is like a quiz where the system will ask questions and the player need to answer these. The player can move further only if he/she is able to answer correctly. Incorrect answer will reduce the health by 2 hearts. Hence it is very important to answer correctly.

**REFERENCES**

1. <https://www.sagitec.com/blog/the-5ws-of-mvvm>
2. <https://www.wintellect.com/model-view-viewmodel-mvvm-explained/>
3. <https://blogs.msdn.microsoft.com/msgulfcommunity/2013/03/13/understanding-the-basics-of-mvvm-design-pattern/>