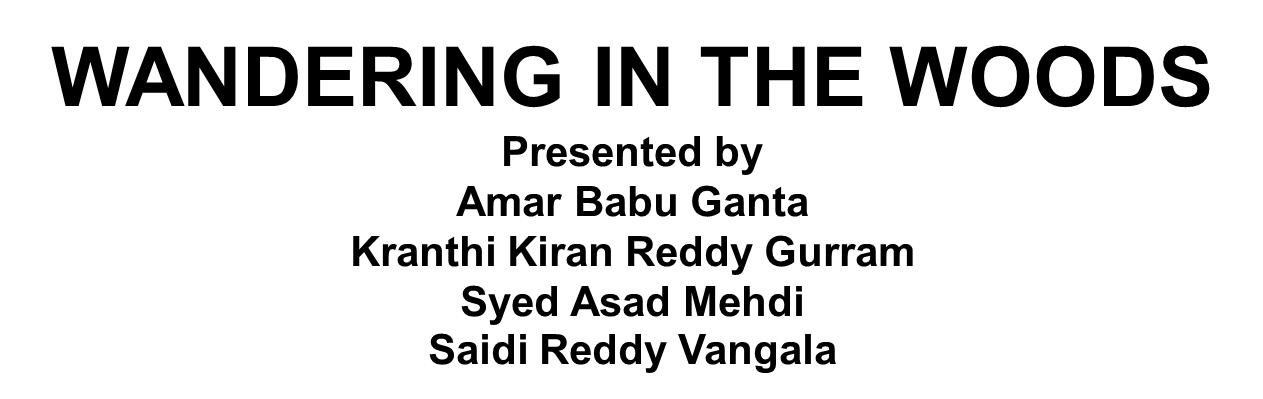
****

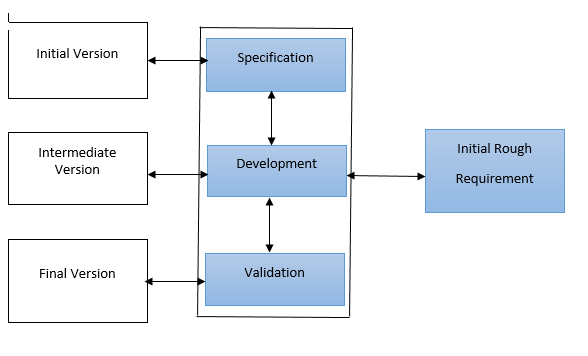
**Introduction:**

This paper has been made as a blueprint of the particular game that we are developing for the sake of the project. The suggested name of the game is “Lost in the Woods''. We are developing the game by using the Agile methodology. For developing the game the requirements for the case study have been verified and checked and after that a documentation has been prepared on the basis of the outcome. The agile design documentation is not constant and this may vary over time. In the design there are various subsystems as well as elements, it does not matter whether the elements are completed or not. In this document the required information has been put that the developers find important for the agile standard. This paper is not a final representation rather it is a blueprint or overall draft of the whole work. This is going to help the students to explore new ideas and thoughts on elementary computer reproduction or replication. This is as an introduction of computing fundamentals, mathematics and programming to the rudimentary students. Here we are going to provide a detailed explanation of the game “Lost in the Woods''.

The game is very simple and intends to introduce a little data to the school children as well as provide entertainment. The children of three to five have been introduced to data and learn to manage the data. The upper graded students are working with the datasets which may be largers as well as smaller, how to generate graphical presentations, making decisions etc. it is suggested to run the simulation with minimum two students who are sharing the same computer screen. This blueprint contains enough information, instructions and other measures that will ease the tasks of the students. This paper contains a detailed insight of this specific game “Lost in the Woods'' and the objectives as well as characteristics of the system.

**Process Model:**

Process models are the graphical representations of the processes related to a business or workflows. It is done to achieve the business objectives. Process models help the business organizations to envision their internal work processes for managing as well as optimizing the tasks in a better way. Most of the time this process models are done in agile methodology. The design of the game has been done by following the evolutionary process model. It is a combination of both the types of processes : iteration and incremental approaches. The process models provide documentation of the processes which already exists as well as provides the knowledge on the process. The evolutionary process model helps to develop a demanded product at a lower cost and minimal risks and all the increments and developments can be done on the product very fast here. The evolutionary model breaks down the models and implements the tasks in a manageable way. It also prioritizes the tasks on the basis of their importance to their clients. During the whole project a huge portion of the product has been delivered to their customers and during that the customers always receive qualitative products and services and hence it gives them more confidence. All the tasks here are very much manageable.



***Figure : 1 :*** *Evolutionary Process Model for System*

**Use Case:**

In the use cases the Users are generally referred to as persons who perform the specific tasks on a system. Use cases are the written description of the tasks that are performed by the user. This is mainly presented from the perspectives of a user. In each of the use cases the goals are set as the beginning point and the end of the case the particular goal is accomplished. It also describes the behavior of the system and helps to detect the wrong behavior. For measuring the cost as well as complexities of a system the use cases generally provide a table of objectives. After the determination of the required functions the developers can negotiate the functions that will be developed. The vuse cases are very easy to understand and are an excellent method for communicating with the clients and the users. The large projects can be handled efficiently with the use cases. In use cases there can be a single or multiple actors present. In businesses mainly the business processes are evaluated in the use cases. In use cases the design scope can be used for identifying the internal and external elements of a process boundary.

**Use Case: 1 : Start Game**

|  |  |
| --- | --- |
| **Use Case ID** | 1 |
| **Description** | In the use case the student is going to launch a game. The game is going to be navigated to the main page, where the user needs to choose the start game. |
| **Actor** | Student |
| **Pre - conditions** | The student needs to install the game first then launch it for starting. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. The game will begin. |
| **Post - Conditions** | The system needs to start the game after loading it in the system memory. |

**Use Case : 2 : Play Game**

|  |  |
| --- | --- |
| **Use Case ID** | 2 |
| **Description** | In this use case the student will play the game and will operate the grid page through various moves. |
| **Actor** | Student |
| **Pre - conditions** | The student needs to choose the play option from the game menu. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. He will choose the play option. 4. The game screen will be displayed by the system where the grid is displayed. 5. The student will start to play. |
| **Post - Conditions** | The system needs to show the grid page. |

**Use Case : 3 : Stage Selection**

|  |  |
| --- | --- |
| **Use Case ID** | 3 |
| **Description** | The student needs to choose a stage as there are multiple stages present in the game. |
| **Actor** | Student |
| **Pre - conditions** | The student should select the stage. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. The student needs to select a stage. 4. He will choose the play option. 5. The game screen will be displayed by the system where the grid is displayed. 6. The student will start to play. |
| **Post - Conditions** | The system needs to show the grid page on the basis of the chosen stage. |
| **Included use case** | Select stage |

**Use Case : 4 : Statistics View**

|  |  |
| --- | --- |
| **Use Case ID** | 4 |
| **Description** | The student will |
| **Actor** | Student |
| **Pre - conditions** | The student should play the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. The student needs to select a stage. 4. He will choose the play option. 5. The game screen will be displayed by the system where the grid is displayed. 6. The student will start to play. 7. The system will show the statistics. |
| **Post - Conditions** | The system needs to show statistics occasionally. |

**Use Case : 5 : Replay Game**

|  |  |
| --- | --- |
| **Use Case ID** | 5 |
| **Description** | The student will replay the game many times. |
| **Actor** | Student |
| **Pre - conditions** | The student must play the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. The student needs to select a stage. 4. He will choose the play option. 5. The game screen will be displayed by the system where the grid is displayed. 6. The student will start to play. 7. The system will show the statistics. 8. The student will replay the game |
| **Post - Conditions** | The system needs to reply to the game after getting the request from the student. |

**Use Case : 6 : Exit Game**

|  |  |
| --- | --- |
| **Use Case ID** | 6 |
| **Description** | The student will exit and finish the game. |
| **Actor** | Student |
| **Pre - conditions** | The student must be present inside the screen of the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option.  * The student will exit from the game.  1. The student will play the game.  * The student exits the game. |
| **Post - Conditions** | The system needs to exit the game and close the game window. |

**Use Case : 7 : Move Diagonal**

|  |  |
| --- | --- |
| **Use Case ID** | 7 |
| **Description** | The student will move diagonally in the opposite direction. |
| **Actor** | K - 2 group Student |
| **Pre - conditions** | The student must be from the group of K2. The student must start the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. He will choose the play option. 4. The game screen will be displayed by the system where the grid is displayed. 5. The student will move diagonally in the opposite direction. 6. If two users or two students meet each other then happy graphics will be displayed on the screen. 7. The student will reset the game. |
| **Post - Conditions** | The system needs to calculate the moves as well as store the statistics. The system needs to show qualitative graphics and play the music. |

**Use Case : 8 : Move Random**

|  |  |
| --- | --- |
| **Use Case ID** | 8 |
| **Description** | The student will move randomly and can also place the characters of the game anywhere on the grid. |
| **Actor** | K-3 Group Student, K 6-8 Group Student |
| **Pre - conditions** | The student must be from the group of K-3 and K 6-8. The student must start the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. He will choose the play option. 4. The game screen will be displayed by the system where the grid is displayed. 5. The student can place the game anywhere on the grid. 6. The student will move in a random direction. 7. If two users or two students meet each other then happy graphics will be displayed on the screen. 8. The student will reset the game. |
| **Post - Conditions** | The system needs to calculate the moves as well as store the statistics.. The system needs to show qualitative graphics and play the music. |

**Use Case : 9 : Change Grid**

|  |  |
| --- | --- |
| **Use Case ID** | 9 |
| **Description** | The student will change the grid size. |
| **Actor** | K-3 Group Student, K 6-8 Group Student |
| **Pre - conditions** | The student must be from the group of K-3 and K 6-8. The student must start the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. He will choose the play option. 4. The game screen will be displayed by the system where the grid is displayed. 5. The student will change the grid size. 6. The student will move in a random direction. 7. If two users or two students meet each other then happy graphics will be displayed on the screen. 8. The student will reset the game. |
| **Post - Conditions** | The system needs to calculate the moves as well as store the statistics. The system needs to change the grid on the basis of the input of the students’. |

**Use Case : 10 : Play Challenges**

|  |  |
| --- | --- |
| **Use Case ID** | 10 |
| **Description** | The student will play the challenges provided by the system. |
| **Actor** | K 6-8 Group Student |
| **Pre - conditions** | The student must be from the group of K 6-8. The student must start the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. He will choose the play option. 4. The game screen will be displayed by the system where the grid is displayed. 5. The student will get new challenges in the game. 6. The student will change the grid size. 7. The student will move in a random direction. 8. If two users or two students meet each other then happy graphics will be displayed on the screen. 9. The student will reset the game. |
| **Post - Conditions** | The system needs to calculate the moves as well as store the statistics. |

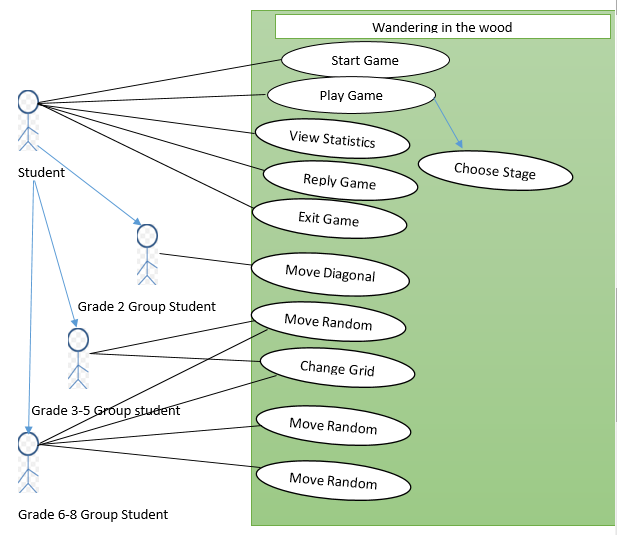
**Use Case : 11 : Variety of tests**

|  |  |
| --- | --- |
| **Use Case ID** | 11 |
| **Description** | The student will try as well as test different kinds of wandering methods for a short time. . |
| **Actor** | K 6-8 Group Student |
| **Pre - conditions** | The student must be from the group of K-3 and K 6-8. The student must start the game. |
| **Flow of Events** | 1. The student will click on the game on the screen. 2. He will choose the start game option. 3. He will choose the play option. 4. The game screen will be displayed by the system where the grid is displayed. 5. The student will use wandering methods. 6. The student will move in a random direction. 7. If two users or two students meet each other then happy graphics will be displayed on the screen. 8. The student will reset the game. |
| **Post - Conditions** | The system needs to calculate the moves as well as store the statistics. The system needs to display the wandering paths to the student. |

**UML Model :**

**Use case Diagram :**

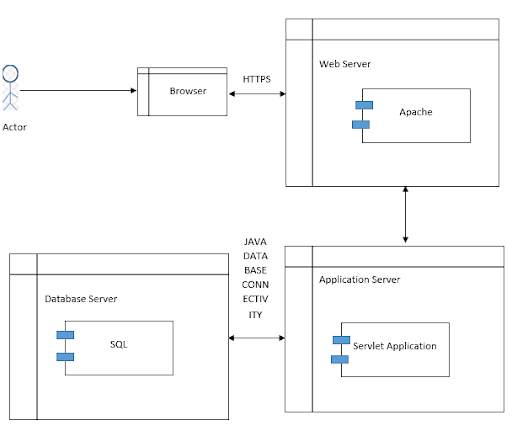
Use case diagrams can be used for capturing the dynamic aspects of the system. These diagrams are also used for evaluating different kinds of systems. This describes the interaction of the systems. The requirements of a system can be collected from the use case diagrams which includes both the influences internal as well as external. Design requirements are the most important among these. The functionalities of a system are combined together through the functions of a system and the recognition of the actors. These diagrams help to recognize the actors and to make interaction with each other. a use case diagram has been shown below :



***Figure : 2 :*** *Use Case Diagram for the System*

**Deployment Diagram :**

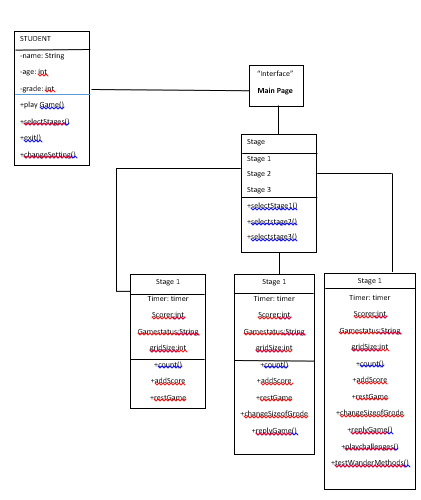
These diagrams are the visualization of the hardware system, their interactions and the software files which are kept on the hardware. UML deployment diagram is used for illustrating the physical architecture of a system. It shows the connection between the components of hardware and software in a system. Here we have developed the system by using Java, here Java connectivity has been used. Apache server will also be used for developing the game. Servlet application will be used by the application server and it will be connected to the SQL for the databases. Here data will remain stored.



***Figure : 3 :*** *Deployment Diagram for System*

**Class diagram :**

Class diagrams are known as static diagrams as well. Class diagrams represent the static views of an application. It acts as a software writing code, visual representation and an explanation for the system. In this “Lost in the Woods'' gamer three stages are there, for these reasons three classes have been developed, each class defines the functions. Here a class diagram has been shown :



***Figure : 4 :*** *Class Diagram for System*

**Class Descriptions :**

|  |  |
| --- | --- |
| **STUDENT** | |
| **ATTRIBUTES** | |
| Name | Student name in string |
| Age | Student age in integer |
| Grade | Student grade in integer for storing the enrolled grades of the students |
| **OPERATION** | |
| Playgame () | Function for playing game |
| Selectstages() | Function for choosing the stages in the game |
| Exit() | Function for exiting the game |
| Changesettings() | Function for changing the setting of the game |

|  |  |
| --- | --- |
| **STAGES** | |
| **ATTRIBUTES** | |
| Stage 1 | Student name in string |
| Stage 2 | Student age in integer |
| Stage 3 | Student grade in integer for storing the enrolled grades of the students |
| **OPERATION** | |
| SelectStage1() | Function for selecting stage 1 in the game. |
| Selectstage2() | Function for selecting stage 2 in the game. |
| SelectStage3() | Function for selecting stage 3 in the game. |

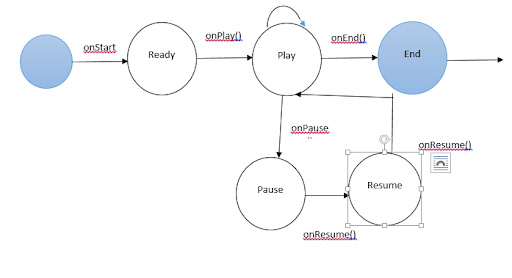
|  |  |
| --- | --- |
| **STAGE 1** | |
| **ATTRIBUTES** | |
| timer | Student name in string |
| scorer | Student age in integer |
| gamestatus | For storing the game status as there are different stages like play, exit etc. |
| gridsize |  |
| **OPERATION** | |
| Count() | Function for calculating the moves of the student |
| Addscore() | Function for adding the student’s scores in each move. |
| Resetgame() | Function to reset the game. |

|  |  |
| --- | --- |
| **STAGE 2** | |
| **ATTRIBUTES** | |
| timer | Student name in string |
| scorer | Student age in integer |
| gamestatus | For storing the game status as there are different stages like play, exit etc. |
| gridsize |  |
| **OPERATION** | |
| Count() | Function for calculating the moves of the student |
| Addscore() | Function for adding the student’s scores in each move. |
| Resetgame() | Function to reset the game. |
| Changesizeof grid() | Function for changing the grid size. |
| replaygame() | Function to replay the game. |

|  |  |
| --- | --- |
| **STAGE 3** | |
| **ATTRIBUTES** | |
| timer | Student name in string |
| scorer | Student age in integer |
| gamestatus | For storing the game status as there are different stages like play, exit etc. |
| gridsize |  |
| **OPERATION** | |
| Count() | Function for calculating the moves of the student |
| Addscore() | Function for adding the student’s scores in each move. |
| Resetgame() | Function to reset the game. |
| Changesizeof grid() | Function for changing the grid size. |
| replaygame() | Function to replay the game. |
| playchallenges() | Function for showing the new challenges to the K 6-8 group students. |
| testwandermethods() | Function for trying and testing different kinds of Wandering methods. |

**State Diagram :**

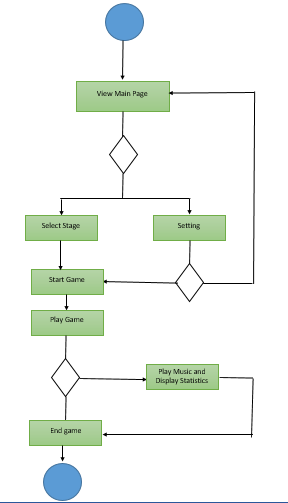
The state diagrams are used for showing the states of a system as well as a part of a system over the time. The behavior of a system is presented through the transitions of the finite state. The state diagram is also referred to as a state machine or the state chart diagram. These diagrams show the responses of a class to the changes of an external stimuli. There are states present in all the classes but we do not use the diagrams for modeling the classes. The states need to be modeled with a minimum of three states. There we have shown a state diagram of the game. There are three states in the diagram READY, PLAY and END. The student will start the game then the game will be in READY state, when the student will choose the option to play the game then it will be converted to PLAY state. If the student PAUSE or RESUME the game then the state will change again. When the student will exit the game then the state will change into the END state.



***Figure : 5 :*** *State Diagram for System*

**Activity Diagram :**

A system’s dynamic aspects are displayed through an activity diagram. It is a latest form of the flowchart that describes the flows of the activities. When the student begins the game a page will show the options and the user needs to choose an option and choose the stage. The user can get back to the main page anytime. If the user prefers to play music and graphics then the system will display qualitative graphics and play music as well. Here an activity diagram has been shown which shows the flow of the activities.



***Figure : 6 :*** *Activity Diagram for System*

**Customer Journey Map :**

The customer journey map is also known as user journey maps. It shows the interaction process with the brand work. These maps can be used for viewing the businesses from the perspectives of the customers. The developers use this for learning about the expectations of the customers and how their expectations can be fulfilled.

**Personas :**

Personas are the standard of the users that can be used by the designers and the developers. Personas help to recognize the characters of the users. After doing all the analysis the developers have found out a user who is an 8th grader student and has some weakness in some specific subjects. He is fond of playing games. This game will help him a lot. He may face some challenges in the game but there are some hints present in the game that will help to solve the challenges.