University of Dhaka
Department of Computer Science & Engineering

Project Report :

Fundamentals of Programming Lab(CSE-1211)

Project Name:

Bon Voyage

Team Members :

1. Fahmida Ara(SK-31)
2. Al Shahriar Rumel(FH-18)
3. Ahnaf Tahmid(FH-04)

1. Introduction :

“Bon Voyage" is an epic adventure game that takes a player on a journey to two most daunting, haunting and magnificently beautiful places. A player can choose to go on an adventure to the largest Mangrove forest Sundarban or can choose to visit the fantasy land Winterfell from the most coveted tv show of the century The Game of Thrones.

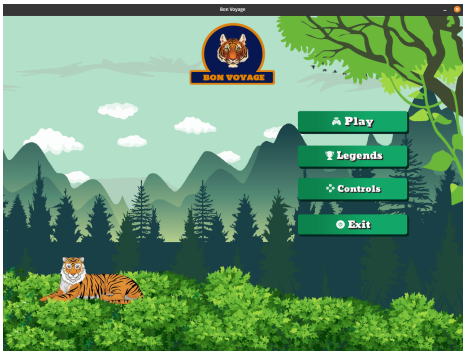
2. Objectives :

The main purpose of this project is to give a user a thrilling experience of playing a dynamic game. The gaming industry is gradually becoming bigger and younger people mostly spend a huge amount of their time playing games. We wanted to make a game that may attract a younger crowd to play this game for hours so we made it visually very appealing and also addicting. This game will not only give people joy but will show them some beautiful wonders of Bangladesh like the Sundarban or Curzon hall. It will also take them on a journey to everyone’s favourite Game of Thrones.

3. Features :

A. Welcome Window

An appealing and dynamic welcome window with the controls of the game and the scoreboard.



B. Choose Levels Window

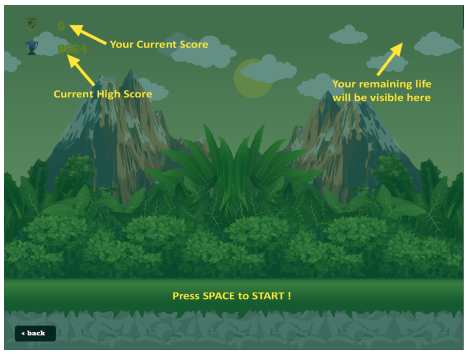
If the player chooses to play , the player is given Options to Play Different Levels

C. Sundarban Level GamePlay

If the player Chooses Sundarban :

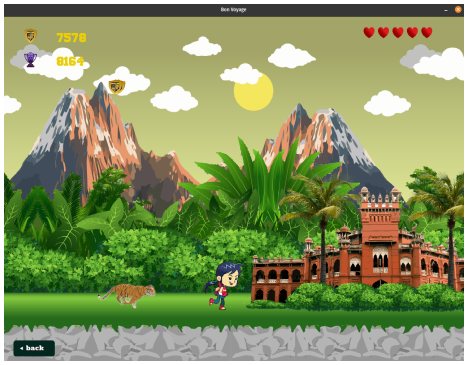
a. After Choosing a Level the Player is Asked to Enter His Handle(UserName) :

b. After Entering the Player Name , the Player is Given an initial Tutorial Window



c. After the player presses SPACE the Game Starts the bewitching level 1 that offers the views of the beautiful Sundarban. Here is a ferocious tiger chasing our amusing character, fascinating coins with CSEDU logo, menacing obstacles and losing lives when our character hits an obstacle. All the rewards and obstacles are random for a challenging feeling and there are sound effects for increasing the thrill.

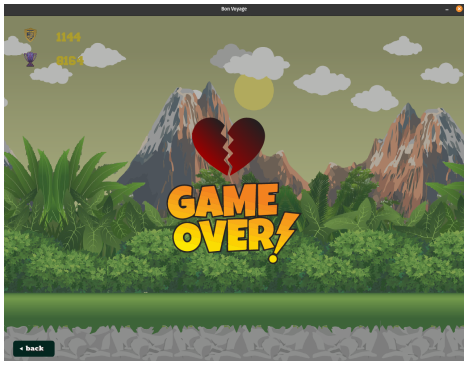
d. If the player reaches to Curzon hall then the level is completed



e. After the player made it to Curzon Hall , the Player is greeted with a game completion message and a Trophy

f. If the player scored more points than the current top five players , the player is given place at the legendary hall of fame

g. If the player fails to reach the Curzon hall , a Game Over window is displayed



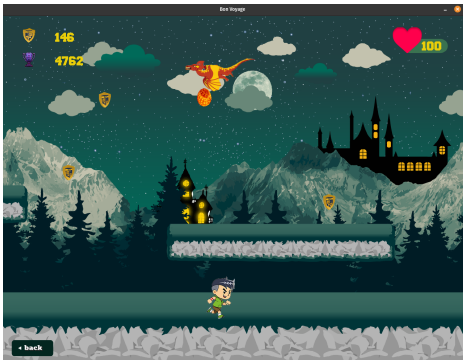
D. Winterfell Level GamePlay

If the player Chooses Winterfell :

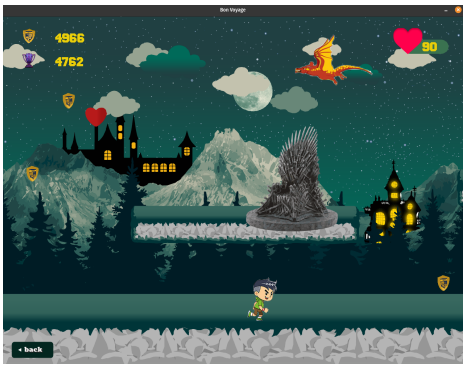
a. After Choosing a Level the Player is Asked to Enter His Handle(UserName) :

b. After Entering Player Name the player is given an initial Tutorial Window

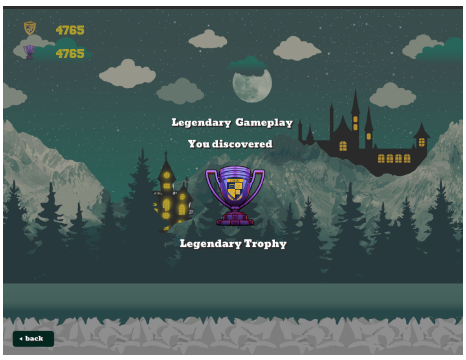
c. After the player presses SPACE the game starts. The mesmerising level 2 offers a captivating view of Winterfell. Here is a terrifying Dragon throwing fire balls, spinning coins,the character gaining lives and tracks coming continuously. All the lives,coins and the tracks are random and there are sound effects to make this playing experience more entertaining.



d. If the player reaches to the Iron Throne then the level is completed

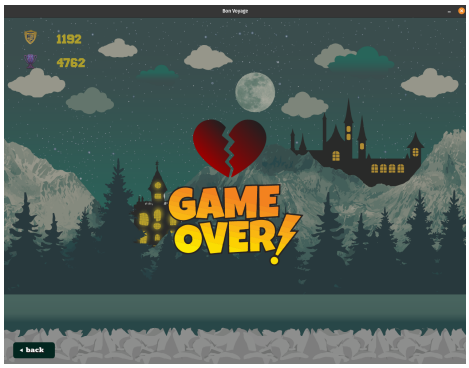


e. After the player made it to the Iron Throne , the player is Greeted with a game completion message and a Trophy



f. If the player scored more points than the current top five players , the player is given place at the legendary hall of fame

g. If the player fails to reach the Iron Throne a Game Over window is displayed.



E. ScoreBoard Window

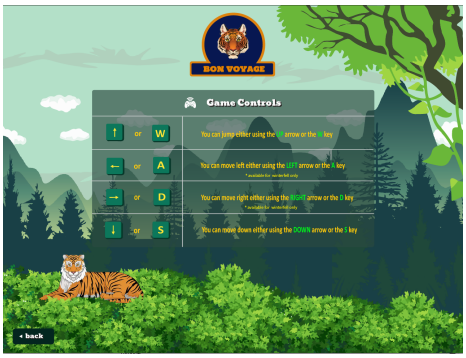
If the player wants to check the legendary LeaderBoard , the player can check the leaderboard by selecting the Legends menu and the player will be given two options to choose :

I. Legends of Sundarban

II. Legends of Winterfell

F. Game Controls Window

Controls tab that provides detailed information about game controls.



G. Game Exit

Player can simply quit the game by Selecting the EXIT button :

4. Project Modules :

We tried our best to make our whole code modular and efficient. For this reason our main.cpp file only contains approximately 10 lines of code just for function calls.The header files we used and their importance are written below:

1. Variables(for both level 1 and 2): Variables that were needed for animation effects are globally declared here and standard variable naming convention was followed.This header file allows us to use all the variables whenever we want just by importing it to the target header file.
2. Structures: Structures were made here for every component to create surface and texture. As a result,initialising and destroying the surfaces,textures were very easily done.
3. Reset file: This header was created to reset all the variables to what we needed at the beginning of the game.
4. Update Score: This header is needed whenever the scores should be updated.
5. Update Player: This header file is called whenever the game starts and then updates the leaderboard if the player is in the top 5 position.
6. Load Score: It's useful for storing the player names and high scores in the file system.
7. Preprocessors: Some preprocessors were declared here for window height width maintenance.
8. Initialising Headers: A lot of headers were declared in order to initialise the different windows and their aspects. For example, in the level one initialising file images for all the components like sky, ground, character, background were assigned.
9. Input header: This header file is needed to take all the mouse events and keyboard events and modify them.
10. Draw Header file: This maintains the speed of the elements and animation effects like colour changing, timing delays, special effects.All the rendering were also done here.
11. Collision Header: Detects collisions between elements and changes the effects and scores accordingly.

We used a common.mk file and added all the headers and cpp files there so that while running the game compiler automatically detects all the separate modules.

How to Use our Header files :

1. Creating New Component(i.e : new characters , levels) :
   To create a new component one simply has to add the name of that component to the structure *Component* in main.h file and after that it will be available throughout the whole modules and it can be used just by importing it to the desired header file by using *extern* keyword.
   For example : If I want to add a component name *newCharacter* ,
   In main.h I will have to write components newCharacter to declare it.

And now if I want to use it in LevelThree.cpp module I will have to import this component in LevelThree.h file by using :
extern component newCharacter

2. Creating New Module :
   If I want to add a new module called levelThree then I will have to add the object file *levelThree.o* in common.mk file for linking it with the rest of the modules.
3. Adding Global Header File :
   If I want to add a new header for all the variables of level Three in *levelThreeVariables.h* then I have to include it in *commonHeaders.h* file and then link it in the *common.mk* file as an object.

5. Team Member Responsibilities :

UI/UX :

The UI/UX of the game is designed by Al Shahriar Rumel.

Game Module :

Which team member implemented which module is written here:

- Al Shahriar Rumel :
  1. variableLevelOne.h
  2. variableLevelOne.cpp
  3. updateScore.h
  4. updateScore.cpp
  5. updatePlayerName.h
  6. updatePlayerName.cpp
  7. resetAll.h
  8. resetAll.cpp
  9. main.h
  10. main.cpp
  11. loadScorefie.h
  12. loadScorefile.cpp
  13. inputs.h
  14. inputs.cpp
  15. initiateWelcomeWindow.h
  16. initiateWelcomeWindow.cpp
  17. initiateScoreBoardHub.h

18. initiateScoreBoardHub.cpp
19. initiateScoreBoard.h
20. initiateScoreBoard.cpp
21. initiateNewGameWIndow.h
22. initiateNewGameWIndow.cpp
23. initiateControlsWindow.h
24. initiateControlsWindow.cpp
25. initialize.h
26. initialize.cpp
27. drawWelcomeWindow.h
28. drawWelcomeWindow.cpp
29. drawScoreBoardHub.h
30. drawScoreBoardHub.cpp
31. drawScoreBoard.h
32. drawScoreBoard.cpp
33. drawNewGameWindow.h
34. drawNewGameWindow.cpp
35. drawLevelOnePlayerName.h
36. drawLevelOnePlayerName.cpp
37. drawControlsWindow.h
38. drawControlsWindow.cpp
39. draw.h
40. draw.cpp
41. structures.h
42. commonHeaders.h

- Fahmida Ara :
    1. initiateLevelOneWindow.h
    2. initiateLevelOneWindow.cpp
    3. initiateObstacle.h
    4. intitiateObstacle.cpp
    5. initiateLife.h
    6. inititiateLife.cpp
    7. initiateLifeLoss.h
    8. initiateLifeLoss.cpp
    9. initiateLevelOneGameOver.h
    10. initiateLevelOneGameOver.cpp
    11. initiateLevelOneCompleted.h
    12. initiateLevelOneCompleted.cpp

13. updateLevelTwoPlayerName.h
14. updateLevelTwoPlayerName.cpp
15. initiateCoinsEffect.h
16. initiateCoinsEffect.cpp
17. initiateCoins.h
18. initiateCoins.cpp
19. inputs.h
20. inputs.cpp
21. drawLevelOneWindow.h
22. drawLevelOneWindow.cpp
23. drawObstacle.h
24. drawObstacle.cpp
25. drawLife.h
26. drawLife.cpp
27. drawLifeLoss.h
28. drawLifeLoss.cpp
29. drawCoinsEffect.h
30. drawCoinsEffect.cpp
31. drawCoins.h
32. drawCoins.cpp
33. drawLevelOneGameOver.h
34. drawLevelOneGameOver.cpp
35. drawLevelOneCompleted.h
36. drawLevelOneCompleted.cpp
37. collisionLevelOne.h
38. collisionLevelone.cpp
39. structures.h
40. commonHeaders.h

- Ahnaf Tahmid :
    1. variablesLevelTwo.h
    2. variablesLevelTwo.cpp
    3. initiateLevelTwoTracks.h
    4. initiateLevelTwoTracks.cpp
    5. initiateLevelTwoLife.h
    6. initiateLevelTwoLife.cpp
    7. initiateLevelTwoCoins.h
    8. initiateLevelTwoCoins.cpp
    9. initiateLevelTwoBombandDragon.h

10. initiateLevelTwoBombandDragon.cpp
11. initiateLevelTwoScoreBoard.h
12. initiateLevelTwoScoreBoard.cpp
13. initiateLevelTwoCompleted.h
14. initiateLevelTwoCompleted.cpp
15. inputs.h
16. inputs.cpp
17. initiateLevelTwoWindow.h
18. initiateLevelTwoWindow.cpp
19. drawLevelTwoWindow.h
20. drawLevelTwoWindow.cpp
21. drawLevelTwoTracks.h
22. drawLevelTwoTracks.cpp
23. drawLevelTwoScoreBoard.h
24. drawLevelTwoScoreBoard.cpp
25. drawLevelTwoPlayerName.h
26. drawLevelTwoPlayerName.cpp
27. drawLevelTwoLife.h
28. drawLevelTwoLife.cpp
29. drawLevelTwoCoins.h
30. drawLevelTwoCoins.cpp
31. drawLevelTwoBombandDragon.h
32. drawLevelTwoBombandDragon.cpp
33. drawLevelTwoGameOver.h
34. drawLevelTwoGameOver.cpp
35. drawLevelTwoCompleted.h
36. drawLevelTwoCompleted.cpp
37. collisionLevelTwo.h
38. collisionLevelTwo.cpp
39. structures.h
40. commonHeaders.h

6. Platform, Library,Tools :

This project is a terminal based game developed for linux based operating systems (i.e ubuntu,popos,mint etc). The language this game is written in is C. The library and the supporting libraries and their installation process is written below.

- SDL/SDL2 : *$ sudo apt-get install libsdl2-image-dev*
- Image rendering :*$ sudo apt-get install libsdl2-image-dev*
- TTF for text rendering : $ *sudo apt-get install libsdl2-ttf-dev*
- Mixer : $ *sudo apt-get install libsdl2-mixer-2.0-0*
- Graphics Creating Tool : *Adobe Illustrator*

7. Limitations :

The main limitation of our game is that it is only available in linux based operating systems. And lastly we couldn't make a *.exe* file for our game for easier installation in different OS.

8. Conclusions :

This project has taught us the real life applications of all the basic components of C language that we learnt from the very beginning of first year.We expected to learn a lot doing this project and we definitely did. We got to see how each component works and changes according to the alterations of our code. The debugging part was very difficult but it taught us how to be patient and how to be relentless in solving a problem. We learnt how to work in a team and how important communication skills are. We learnt about ui/ux and how changing some little details can make a game more acceptable to a user. Making our code modular was tremendously challenging but we learnt it while doing and working so it was a blessing in disguise. This is our first project and it showed us the way how projects are done in a professional work environment and what requirements and rubrics should be maintained. Moreover, it taught us the importance of deadlines.

9. Future Plan :

Our future plan would be to make this game available for all the platforms and make more advanced levels to make the game more difficult to finish.

Repositories :

Github repository : https://github.com/Bon-Voyager/BonVoyage
Youtube video : https://www.youtube.com/watch?v=jwL_E4QpZxk

References :

1. *https://wiki.libsdl.org/*
2. *https://lazyfoo.net/tutorials/SDL/index.php*
3. *https://stackoverflow.com/*