Traffic Racer

Members:
M. Raahim 24K-0543
Shobha Kumari 24K-1006
Abdul Saboor 24K-0536

Submission Date:
50 percent-> 24th April 2025
100 percent-> 11th -May-2025

1- Introduction:

The main idea of this project is creating a traffic racer game. It is a game designed on C++ language using SFML library. It focuses on object orientated programming concepts and real time game interaction.The goal is to demonstrate how OOP can be applied to create a real-time racing game that includes object behaviors, collision detection, score tracking. The project aims to simulate a traffic racing scenario where the player must avoid obstacles.The game is deemed over if the player collides with Police car while -2 is added to the score on collision with normal cars.

The objectives are:
1- Develop an interactive traffic racing game using C++ and SFML.
2- Apply core OOP concepts such as classes, inheritance, and encapsulation.
3- Demonstrate collision handling, sprite movement, and user input processing.
4- Implement a scoring and restart mechanism.

2- Scope:

Inclusions:
1- GUI will be integrated using SFML.
2- Vector library is included to manage list of obstacles.
3- A player controlled car to move left and right.
4- Obstacles like Normal cars and police cars.
5- Scoring system based on time.
6- Restart function by pressing R after the game is over.
7- Graphic Visuals by using SFML.

Exclusions:
1- No audio in the game.
2- No Multiplayer mode.
3- No movement of the car in forward or backward direction.
4- No saving of data/ No leaderboard feature.

3- Project Description:

Traffic Racer is a desktop-based game where players control a car driving along a busy road, avoiding other vehicles and a chasing police car. The player scores points by surviving and dodging cars. The game ends upon collision with the police car, offering a restart option.

Technical requirements:

1- IDE: Visual Studio Code
2- Graphics Library: SFML
3- Compiler: MinGW g++
4- Programming Language: C++
5- Assets: Images of car and background in PNG format and a font file.
6- Extra: Make for Windows[Used to ease the compiling].

Project phases:

1- Planning: Game mechanics and building up classes.
2- Background work: Setting up SFML and including its library in the project.
3- Design: Setting up sprites and building classes structure.
4- Development: Implementing game loop, movement, collision detection and scoring mechanism.
5- Finalization: Adding game over UI and setting up the Restart function.

4- Methodology:

Approach:

The initiative employs a repetitive and modular development strategy. Every game feature was divided into smaller tasks and addressed one after the other. The design began with basic class, then sprite rendering,then obstacle creation,player movements and then the game loop and working and in the end the restart function.

Responsibilities:

The responsibilities were distributed equally between all three members. The working on the project was done together during university hours and even on reserved day.
1-Raahim was responsible for creating sprites,building the SFML logic and implementing the core game loop and creating the proposal.
2-Shobha was responsible for creating classes,collision detection and restart mechanism.
3-Saboor was responsible for scoring mechanism and worked on report file.

5- Outcomes:

Deliverables:

1- A executable file (.exe file)
2- Source code with comments and organized functions.

Relevance:
This project showcases the practical application of OOP principles such as encapsulation and inheritance . It also includes ICT topics such as simple game development, event handling, data organization (scoring logic), and graphical integration using external libraries (SFML).

6- <u>Resources:</u>

Softwares Needed:
1- Visual Studio Code
3- SFML
4- Image assets for Cars and obstacles.

Other Resources:
1- YouTube Tutorials on how to setup SFML.
2- YouTube Tutorials on how to use SFML.
3- Chat GPT’ed some errors to fix.
4- Googled Sprites and fonts.