BUBT

PROJECT PROPOSAL

ON

DEVELOPING A COMPUTER GRAPHICS PROJECT USING OPENGL

SHADOW NEMESIS

NAME	**ID**	**INTAKE/ SECTION**
ABDULLAH AL MAHMUD JOY	21225103506	49/7
MD. FAIZER ISLAM	21225103466	49/7
MD. SUNZID ISLAM	21225103212	49/7
MIRZA NIAZ MORSHED	21225103467	49/7
ARAFAT HOSSAIN FAHIM	21225103456	49/7

SUBMITTED TO:
MD. SAKHAWAT HOSSAIN
LECTURER, DEPARTMENT OF CSE

MAY 4, 2025

i) Proposal Number: Proposal 1

ii) Proposal Title
Speed Rush: A 2D Top-Down Racing Challenge

iii)Abstract
Speed Rush is a 2D top-down racing game where players control a car speeding along a scrolling highway while dodging obstacles such as oncoming vehicles and roadblocks. The goal is to survive for as long as possible while the game's difficulty increases over time. The project showcases core computer graphics principles like object modelling, real-time transformations, and collision detection, making it an engaging and educational experience in game development.

iv)Justification
This project was chosen for its balance between creativity and technical depth. It offers a fun and visually interactive way to apply essential computer graphics concepts such as object rendering, animation, and collision logic. Developing this game will strengthen skills in graphics programming, logical structuring, and user interaction design, all of which are vital for both academic and industry-level software development.

v) Features

- Top-down 2D car racing gameplay
- Continuously scrolling road background
- Dynamic obstacle generation (other cars, roadblocks)
- Increasing difficulty with progressive speed
- Collision detection for game-over scenarios
- Score tracking based on survival time
- Basic animation for moving objects
- Optional: Texture mapping for car and road visuals
- Restart and exit options from the game menu

vi) Software Toolchain Programming Language:
C++ — for performance, control over memory, and wide compatibility with graphics libraries.

- Hardware Requirement (Minimum):
  - Core i3 processor-based computer
  - 4 GB RAM
  - 256 HDD/128 SSD
- Software Requirement (Minimum):
  - Windows operating system (7 or above)
  - Code: Blocks
- Graphics Library:
  OpenGL (specifically 2D rendering) — to implement graphics primitives, transformations, and the rendering pipeline.

i) Proposal Number: Proposal 2

ii) Proposal Title
Brick Blaster: A Classic Arcade Remake

iii)Abstract
Brick Blaster is a 2D arcade-style game where the player controls a horizontal paddle to keep a bouncing ball in play, aiming to break all bricks arranged at the top of the screen. The ball ricochets off the paddle and bricks, and the game ends if the ball falls below the paddle. The project showcases core computer graphics and interactive programming concepts, such as 2D transformations, collision detection, and event-driven input.

iv)Justification
This project was selected due to its simplicity, popularity, and strong relevance to foundational graphics and game design principles. Building this game reinforces understanding of motion physics, collision response, real-time input handling, and level progression. It also encourages modular design and game state management, which are critical skills for more advanced interactive applications.

v) Features

- Real-time ball and paddle movement using 2D transformations
- Collision detection between ball, paddle, bricks, and screen borders
- Brick breaking logic with score update
- Multiple levels or increasing difficulty (e.g., ball speed increases, more bricks)
- Game-over and restart functionality
- Paddle movement using keyboard input
- Sound effects for ball hits and brick breaking (optional)
- Visual effects: color-changing bricks, disappearing animations (optional)
- Score display and lives counter

vi) Software Toolchain Programming Language:
C++ — for performance, control over memory, and wide compatibility with graphics libraries.

- Hardware Requirement (Minimum):
    - □ Core i3 processor-based computer
    - □ 4 GB RAM
    - □ 256 HDD/128 SSD
- Software Requirement (Minimum):
    - □ Windows operating system (7 or above)
    - □ Code: Blocks
- Graphics Library:
  OpenGL (specifically 2D rendering) — to implement graphics primitives, transformations, and the rendering pipeline..

i) Proposal Number: Proposal 3

ii) Proposal Title
Sky Strike: 2D Fighter Jet Shooter

iii)Abstract
Sky Strike is a 2D side-scrolling shooter game in which the player pilots a fighter jet capable of moving vertically and firing bullets to eliminate enemy aircraft and obstacles. The game features real-time controls, object collisions, and continuous animations. Players score points by destroying enemies while avoiding collisions and enemy fire. The project illustrates fundamental graphics concepts like object modelling, 2D transformations, animation, and interactive input handling.

iv)Justification
This project was chosen for its blend of action and technical depth. It offers a fun way to apply graphics programming techniques such as collision detection, movement physics, and real-time input. Creating multiple enemy patterns and shooting mechanics challenges problem-solving and game logic design. This project is ideal for enhancing skills relevant to game development, including sprite animation and screen-bound object management.

v) Features

- Player-controlled fighter jet with vertical movement
- Bullet firing mechanism and enemy collision detection
- Animated enemies flying from right to left
- Score system based on enemies destroyed
- Life system or health bar (optional)
- Keyboard-based input control
- Explosion or particle effect animations (optional)
- Background scrolling to simulate flight (optional)
- Multiple enemy types or increasing difficulty levels

vi) Software Toolchain Programming Language:
C++ — for performance, control over memory, and wide compatibility with graphics libraries.

- Hardware Requirement (Minimum):
    - Core i3 processor-based computer
    - 4 GB RAM
    - 256 HDD/128 SSD
- Software Requirement (Minimum):
    - Windows operating system (7 or above)
    - Code: Blocks
- Graphics Library:
  OpenGL (specifically 2D rendering) — to implement graphics primitives, transformations, and the rendering pipeline.