Project Roadmap: Endless Runner Game

Game Overview

- **Endless runner game with 3 lanes**
- **Player moves left and right to avoid obstacles**
- **Collectible items (x) appear randomly, offering points**
- **Game ends upon collision with obstacles**
- **Score based on time survived and items collected**
- **High score tracking**

Key Features

- **3 lanes with downward movement**
- **Randomly spawned obstacles (single or double)**
- **Randomly spawned collectible items (x)**
- **Time-based scoring with item collection bonuses**
- **High score display**

Game Architecture Design

- **Game Loop**
    - **Responsible for updating and rendering the game state**
    - **Runs continuously until the game is exited**
    - **Typically implemented as a loop that repeats at a fixed rate (e.g., 60 times per second)**
    - **Handles:**
        - **User input**
        - **Game state updates**
        - **Collision detection**
        - **Scoring**
        - **Rendering**

Example game loop pseudocode:

```
while (gameRunning) {
  // Handle input
  handleUserInput();

  // Update game state
  updatePlayer();
  updateObstacles();
  updateCollectibles();

  // Check collisions
  checkCollisions();

  // Update score
  updateScore();

  // Render game state
  renderGame();

  // Cap frame rate
  delay(16); // 60 FPS
}
```

- **Game State**
  - **Manages the current status of the game**
  - **Includes:**
    - **Player data (position, velocity, score, etc.)**
    - **Obstacle data (positions, velocities, types)**
    - **Collectible data (positions, types)**
    - **Game mode (playing, paused, game over)**
    - **Score (current and high score)**
- **Player**
  - **Manages player movement and interactions**
  - **Includes:**
    - **Position and velocity**
    - **Collision detection**
    - **Scoring**
    - **Movement logic (left and right movement)**
- **Obstacles**
  - **Manages obstacle spawning and movement**
  - **Includes:**
    - **Spawning logic (random or predefined)**
    - **Movement logic (downward movement)**
    - **Collision detection**
    - **Types (different types of obstacles)**

- **Collectibles**
    - **Manages collectible item spawning and collection**
    - **Includes:**
        - **Spawning logic (random or predefined)**
        - **Collection logic**
        - **Scoring**
        - **Types (different types of collectibles)**
- **Graphics**
    - **Handles rendering of game elements using SplashKit**
    - **Includes:**
        - **Player rendering**
        - **Obstacle rendering**
        - **Collectible rendering**
        - **Background rendering**
        - **UI rendering (score, game over screen, etc.)**
- **Input**
    - **Handles user input (keyboard or mouse events)**
    - **Includes:**
        - **Player movement input**
        - **Pause/resume input**
        - **Restart input**

Tasks

1. **Set up SplashKit and create a basic game window**
2. **Implement game loop and game state management**
3. **Create player entity and implement movement**
4. **Implement obstacle spawning and movement**
5. **Add collision detection for player-obstacle and player-collectible interactions**
6. **Implement collectible item spawning and collection logic**
7. **Develop scoring system (time-based and collectible-based)**
8. **Create high score tracking system**
9. **Implement graphics rendering for game elements**
10. **Add user input handling for player movement**

Additional Considerations

- **Error handling and debugging**
- **Performance optimization**
- **Audio implementation (optional)**
- **Testing and iteration**