TP0: Project proposal

**Project Description**

Stick figure badminton

This is a sport badminton game where you must get the first 7 points to win the game. You can play it by yourself or with your friend. There are two options for single player mode, the first is advanced mode and the second is fun mode. The advanced mode uses the keyboard to override (wasd, ↑↓←→), and the fun mode uses the mouse to control (move, click). For two-player mode, it is controlled by two players using wasd and ↑↓←→

**Similar projects**

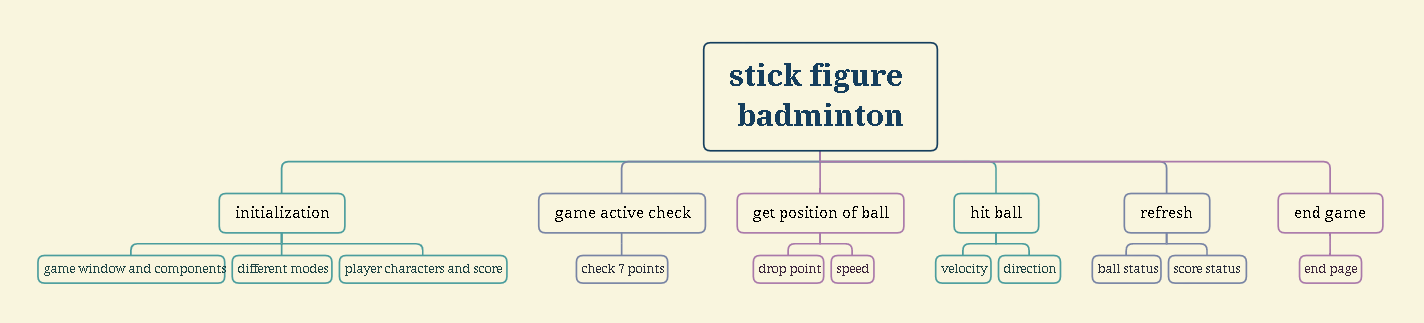
[1].https://www.crazygames.com/game/stick-figure-badminton

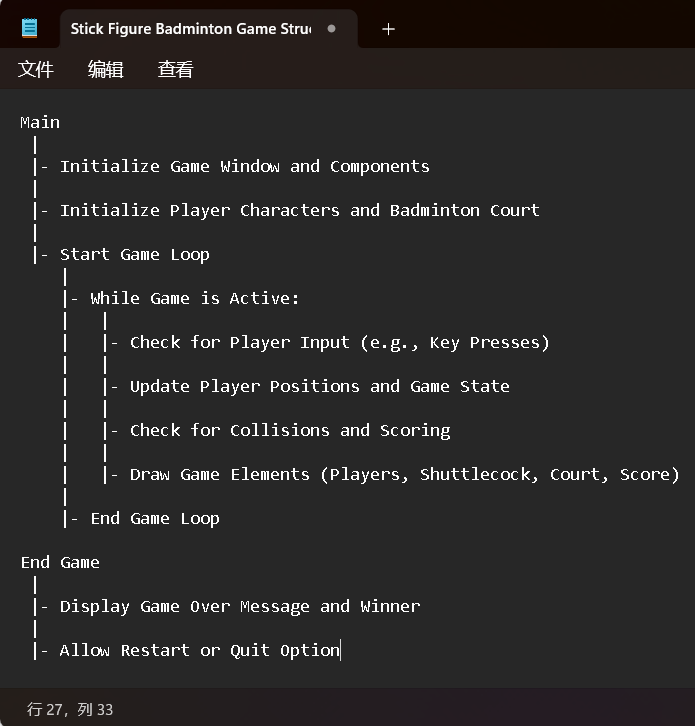
This game it's writing the code about badminton flying with gravity and air resistance in mind. This process need to implement a physics engine that calculates these effects in real-time can be complex and hard to realize . At the same time, there are red tracks behind the badminton in this game, which can make the game look more cool.

[2].https://www.4399.com/flash/46923.htm#search3-fa3f

This game provides large amount of stick figure with different costumes. This variety of options can pique the player's interest .At the same time, the game also has the option to input the name, which allows the player to input their own name into the game, and the experience is stronger when the user plays. However, i n my game, I make the sound effect which is more cool than the original one.

**Structural Plan**





**Explanation of Key Components**:

**Main**: This is the main function that serves as the entry point of the game. It initializes the game window, sets up the player characters, badminton court, and other game components. It then starts the game loop to manage game events and updates.

**Initialize Game Window and Components**: This step involves creating the graphics window, setting up the game environment, and initializing any necessary variables.

**Initialize Player Characters and Badminton Court**: This step sets up the stick figure player characters and the badminton court (net, boundaries, etc.).

**Start Game Loop**: The game loop runs continuously as long as the game is active. It handles player input, updates the game state, and draws game elements on the screen.

**Check for Player Input**: The game loop constantly checks for player input, such as key presses, mouse clicks. Player input affects the movement of stick figures and their actions (swinging the racket).

**Update Player Positions, badminton position and Game State**: Based on the player input, the game loop updates the positions of the stick figure players and the badminton. It also keeps track of the game's state, such as scores and player turns.

**Check for Collisions and Scoring**: The game loop checks for collisions between the badminton and the players, the badminton and the court boundaries, and if the badminton lands on the opponent's side. It then updates the score accordingly.

**Draw Game Elements**: The game loop draws the stick figure players, badminton, badminton court, and scores on the game window based on the updated game state.

**End Game**: When the game loop ends (e.g., reaching 7 scores), the game enters the end game phase. It displays the game over message, announces the winner, and provides options to restart or quit the game.

**Algorithmic Plan**

**Hit ball**

(Update Player Positions, badminton position)

1.**Real-time Movement**: In a real-time game like badminton, the badminton moves rapidly across the court, and its position changes continuously. Tracking the exact position of the badminton at any given moment requires precise and frequent updates, which can be computationally intensive.

**Solution**:use a real-time game loop that updates the position of the badminton at regular intervals (e.g., every frame). Within the game loop, I can calculate the new position of the badminton based on its velocity and direction. This way, the badminton will move continuously and smoothly across the court. To make the tracking more efficient, I can use vector-based calculations .

2.**Physics Simulation**: To accurately model the movement of the badminton, the game needs to simulate the laws of physics governing its trajectory, including factors like velocity, angle.

**Solution**: use physics equations to calculate the position and velocity of the badminton at each time step. Consider using numerical integration methods like Euler's method to approximate the continuous motion of the badminton in discrete time steps.

(Update Velocity:

Update the velocity of the badminton in both x and y directions using the current acceleration and time step:

vx\_new = vx + ax \* Δt

vy\_new = vy + ay \* Δt

Update Position:

Update the position of the badminton in both x and y directions using the updated velocity and time step:

x\_new = x + vx \* Δt

y\_new = y + vy \* Δt)

**Timeline Plan**

**TP0(Aug 1):**

**Game Concept and Design**: Finalize the concept and overall design of the game. Decide on the gameplay mechanics, rules, and objectives.

**Graphics and Assets**: Create or gather the necessary graphics and assets for the background, stick figures, badminton court, badminton, net, and other game elements.

**Initialize CMU Graphics**: create the game window

**Stick Figure Movement**: Implement basic movement controls for the stick figures using keyboard inputs. Allow them to move left and right.

**TP1(Aug5):**

**Stick Figure Animation**: Animate the stick figures' arms and legs to mimic basic running and jumping motions.

**badminton Movement**: Implement the badminton's movement logic. Make it move in a realistic trajectory when hit by the players.

**Scoring System**: Implement a scoring system that tracks points and declares the winner based on the game rules.

**TP2(Agu8):**

**Collision Detection**: Add collision detection for the badminton and the stick figures. Detect when the badminton is hit or when a player misses a shot.

**Game Logic**: Implement the core game logic, such as player turns, serving, and winning conditions.

**User Interface**: Design and implement a user interface that displays the score, player names, and game status.

**TP3(Aug10):**

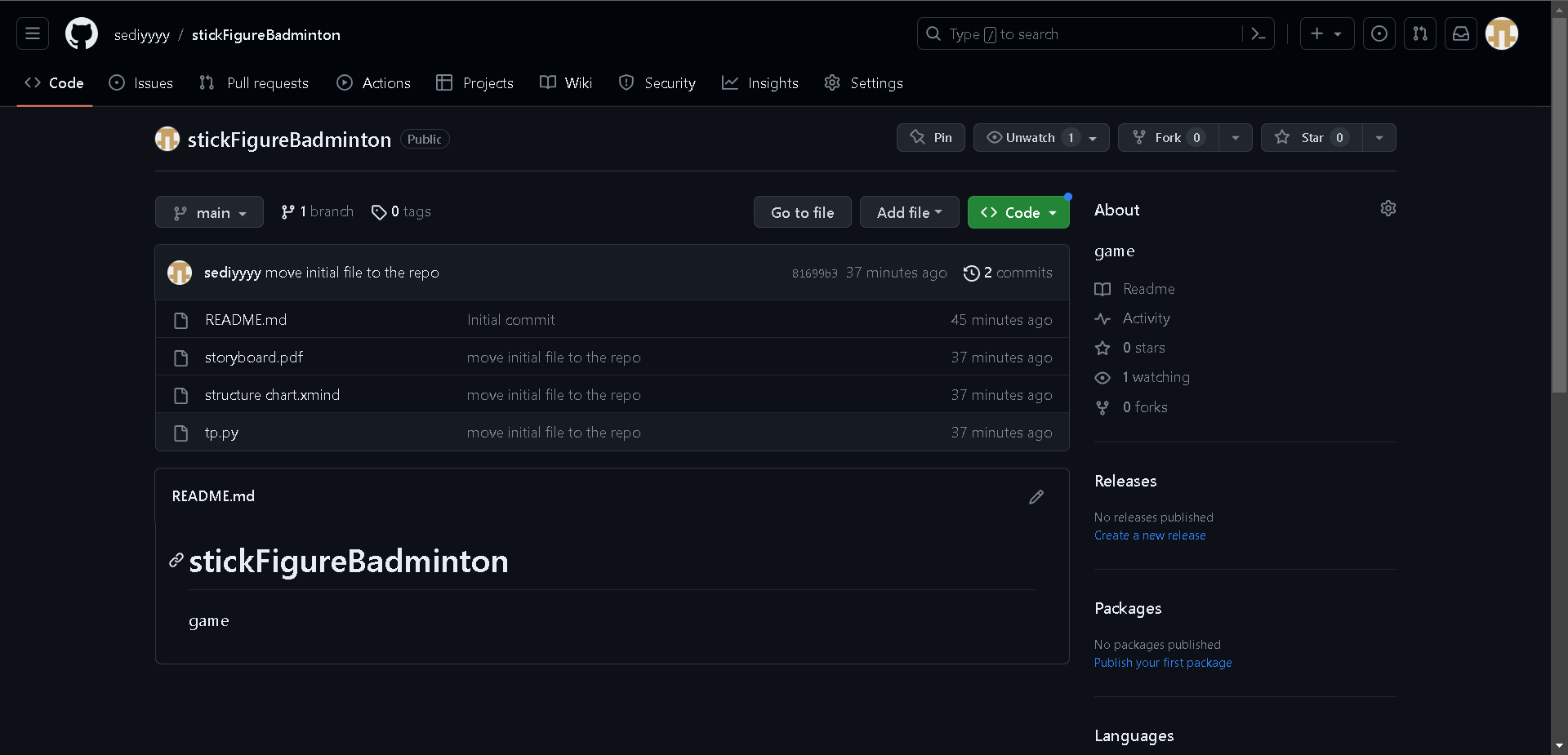
**Polish and Refinement**: Review the game for bugs, glitches, and inconsistencies. Fix any issues and refine the gameplay for a smoother experience.

**Sound Effects**: Add sound effects for hitting the badminton, scoring points, and other in-game actions to enhance the gaming experience.

**Testing and Debugging**: Conduct thorough testing on different devices and platforms to ensure the game runs smoothly and without errors.

**Version Control Plan**

github



**Module List**

none

**Story board**

separate document