* **Game Name**: **Labyrinth**
* **Concept** : Description - A gravity based game which controls the ball through a maze to reach a goal.



* **Work-flow :**   
  World Environment --> Plane with walls --> sphere --> gravity using CannonJS --> general UI to interact with the game
* **Technology**
  + **ThreeJS**
    - Creating the game environment and objects.
      * The plane, the walls and the sphere is created using ThreeJS. Also the skybox for the background
    - Lights, Camera and Renderer.
      * The Lights and the camera are created using this. Also the canvas in the html which displays the output is made using the renderer.
    - Textures.
      * Various textures for the objects that give it detailing and gives a real life look to them.
  + **CannonJS**
    - Physics of the game world.
      * Gravity and movement in the game is created using CannonJS.
  + **TypeScript**
    - Computation for the game.
    - Implementing the different libraries into one and making it into a single executable script.
  + **HTML and CSS**
    - Front-end and UI of the game.
* **Rules of the game :**
  + Use WASD to change the gravity (resembles tilting a board) to make the sphere roll over and navigate through the maze.
  + In case of mobile devices or devices inbuilt with a gyroscope, the sphere can be controlled by tilting the device. (experimental)
  + On escaping from one labyrinth, the user finds out that he/she is inside a chain of labyrinths and must escape it.
  + Use of wits and memory.
* **Estimate / Road-map**
  + Stage 1: Create Game World (3 days)
  + Stage 2: Create Game Physics (3 days)
  + Stage 3: Create Game Levels (different labyrinth for each level) (3 days per level)
  + Stage 4: Create Collider detection and game completion detection (3 days)
  + Stage 5: Make it compatible with various devices.
  + Stage 6: Additional bug-fixing and introducing mini features
  + Stage 7: Test Run

**STATE DIAGRAM**

**Start**

Deploy Ball into the maze and zoom in to the ball after 5 seconds

Manipulate gravity to move the ball in the respective direction

**Yes**

Ball crosses exit

**No**

End Screen and Score pops up

Escape button

**End**

Pause menu with Reset button

**CLASS DIAGRAM**

Main class or main window

scene

camera

renderer

main()

OnKeyDown()

zoomCamera()

animate()

Sphere

ThreeJS sphere

CannonJS Sphere

createSphere()

createBody()

Maze

ThreeJS Objects

CannonJS Objects

createMaze()