VN-UK
Institute for Research
& Executive Education

COMPUTER SCIENCE AND ENGINEERING

3D PROGRAMMING

Project Report

Marble Run

by

Vo Minh Nhut 20020005

I. Introduction

"Marble Run" is not just a game, it's an adventure that captivates the mind and tests the limits of agility and strategic planning. As players embark on this journey, they find themselves immersed in a world where each twist and turn brings new challenges. The essence of the game lies in navigating a marble through a labyrinth of ingeniously designed traps and obstacles, each meticulously crafted to offer a unique gaming experience.

The game's core objective is seemingly straightforward – guide your marble to the finish line within the allotted time. However, achieving this is anything but simple. Players are required to demonstrate exceptional control and foresight, anticipating the marble's trajectory and the consequences of every swift move. The sense of urgency is palpable as the clock ticks down, adding an exhilarating layer of pressure that keeps players engaged and on the edge of their seats.

One of the standout features of "Marble Run" is its stunning visual presentation. Built using Next.js and React-Three libraries, the game showcases a harmonious blend of advanced web technologies and 3D graphics. This choice of technology not only ensures smooth gameplay but also brings each level to life with breathtakingly realistic textures and lighting effects. The environment is not just a backdrop but an integral part of the game, with each setting offering a distinct atmosphere and a set of unique challenges.

The levels in "Marble Run" are designed to be progressively challenging, ensuring that players remain engaged and motivated. Each level introduces new traps and obstacles, ranging from simple hurdles to complex mechanisms that require quick thinking and precise control. This variety not only tests the players' dexterity but also their ability to adapt their strategies on the fly.

Moreover, the game's design encourages replayability. With multiple pathways and hidden secrets in each level, players are incentivized to explore different strategies to improve their time and score. This aspect of the game not only enhances its longevity but also fosters a sense of community as players share tips and compete against each other for the top spot on the leaderboard.

In addition to its engaging gameplay, "Marble Run" offers a user-friendly interface and intuitive controls, making it accessible to players of all ages and skill levels. Whether you're a seasoned gamer or new to the world of digital gaming, "Marble Run" offers an enjoyable and challenging experience.

In conclusion, "Marble Run" is more than just a game; it's a test of skill, a race against time, and a visual spectacle. It combines the thrill of arcade gaming with the complexity of strategy and puzzle-solving, all wrapped up in a visually stunning package. Whether you're playing to beat your personal best or competing with friends, "Marble Run" promises an engaging and rewarding experience for everyone.

II. Installation

The installation process for "React Three Run Game" is designed to be straightforward and user-friendly, catering to both seasoned developers and those new to using Git and Node.js. Here's a detailed guide to help you set up and run the game:

1. Prerequisites:

Before you begin, ensure you have the following installed on your computer:

1.1. Git:

This is a version control system for tracking changes in source code during software development. It's essential for downloading the game's repository.

1.2. Node.js and npm:

Node.js is a JavaScript runtime environment that lets you run the game's server-side code. npm (Node Package Manager) is included with Node.js and is used for managing JavaScript packages.

If you don't have these installed, visit the official websites for Git and Node.js, and follow the installation instructions for your operating system.

2. Installation Steps:

Once you have the prerequisites installed, follow these steps to install and run "React Three Run Game":

2.1. Clone the Repository:

First, you'll need to clone the game's repository from GitHub. This step involves downloading all the files you need to run the game.

Open your terminal or command prompt and enter the following command:

```
git clone https://github.com/nhut3110/react-three-run-game.git
```

This command creates a copy of the game's repository on your local machine.

2.2. Navigate to the Game Directory:

After cloning, move into the game's directory by typing:

```
cd react-three-run-game
```

This command changes your current directory to the newly cloned game repository.

2.3. Install Dependencies:

The game relies on various npm packages. To install these, run:

```
npm install
```

This command downloads and installs all the necessary dependencies defined in the game's `package.json` file. It might take a few minutes depending on your internet connection.

2.4. Launch the Game:

Finally, to start the game server, enter:

```
npm run dev
```

This command starts the development server. Once the server is running, it will provide a URL (usually `http://localhost:3000`) that you can open in your web browser to start playing the game.

3. Post-Installation:

After completing these steps, the game should be up and running on your local machine. You can now enjoy the immersive experience of "React Three Run Game" right from your browser. The development server allows for real-time updates, so any changes you make to the code will be immediately visible upon refreshing the game page.

4. Troubleshooting:

If you encounter any issues during installation, ensure you have the latest versions of Git, Node.js, and npm. Check your internet connection, and make sure there are no firewalls or antivirus programs blocking the installation. For specific errors, a quick search online or consulting the game's documentation or community forums can often provide solutions.

Enjoy your journey through the challenging and visually stunning world of "React Three Run Game"!

III. Usage

"Marble Run" is meticulously designed to offer a gameplay experience that is both intuitive and engaging, ensuring that players of all skill levels can dive into the game with ease while still finding it challenging and stimulating. Here is a detailed guide on how to control and interact with the game to maximize your enjoyment:

1. Basic Movement Controls:

- **Forward Movement (W Key):** Pressing the 'W' key propels the marble forward, allowing you to navigate through straight paths and towards upcoming obstacles.
- **Left Movement (A Key):** The 'A' key moves the marble to the left. This is essential for dodging obstacles on your right or steering through leftward turns.
- **Backward Movement (S Key):** Use the 'S' key to pull the marble backward. This can be particularly useful for slowing down or adjusting your position if you've overshot a turn or target.
- **Right Movement (D Key):** Pressing the 'D' key will move the marble to the right, vital for evading left-sided traps or navigating right turns.

2. Advanced Interaction:

- **Jump (Space Bar):** The Space bar is your tool for making the marble jump. This action is crucial for overcoming certain traps, hurdles, or gaps in the path. Timing your jumps with precision is key to advancing in the game.

3. In-Game Settings and Customization:

- Accessing Settings (Settings Button): Click on the settings button, conveniently located at the top-left corner of the screen, to open the game's settings menu.
- Adjust Trap Density: Inside the settings menu, you have the option to increase or decrease the number of traps in the game. The maximum limit is set to 100 traps. This feature allows players to tailor the difficulty level according to their preference, making the game suitable for both beginners and experienced gamers.
- Toggle Sound: You can choose to turn the game sound on or off. This flexibility is great for those who prefer playing in silence or for those who enjoy the immersive experience of game audio.
- Restarting the Game (Restart Button): If you find yourself needing to start over, simply click on the restart button located at the top-right corner of the

screen. This action resets the level, allowing you to try again from the beginning.

4. Gameplay Experience:

These controls and options are thoughtfully designed to give players complete control over their gameplay experience. "Marble Run" caters to a wide range of players by combining the straightforwardness of its controls with the depth of its gameplay mechanics. Whether you are meticulously navigating through the maze of traps or fine-tuning the game settings for a uniquely challenging experience, "Marble Run" ensures a dynamic and captivating experience for everyone.

Through its combination of intuitive controls and customizable settings, "Marble Run" stands out as a game that is both accessible to newcomers and deeply engaging for seasoned players. Each session promises not just a test of dexterity and strategic thinking, but also a journey through a beautifully crafted digital landscape filled with challenges and surprises. Enjoy your adventure through the captivating world of "Marble Run"!

IV. Code explanation

1. Libraries and Frameworks Used

In "Marble Run," the seamless integration of various libraries and frameworks plays a crucial role in creating an engaging and high-performance gaming experience. Each of these technologies contributes unique features that, when combined, result in a sophisticated and enjoyable game. Let's delve into the specifics of each library and framework used in the project:

1.1. React (v18.2.0):

- Purpose: React is a popular JavaScript library used for building user interfaces, particularly single-page applications. It's known for its efficiency and flexibility.
- Role in Marble Run: React is utilized for rendering the UI components of the game, managing the game's state, and responding to user interactions.

1.2. Next.js (v13.4.12):

- Purpose: Next.js is a powerful React framework that enables server-side rendering and generates static websites for React-based web applications.

- Role in Marble Run: It enhances the game’s performance by optimizing loading times and providing an efficient structure for the game's web application.

1.3. Three.js (v0.155.0):

- Purpose: This is a 3D library that creates and displays animated 3D computer graphics in a web browser using WebGL.
- Role in Marble Run: Three.js is central to rendering the 3D graphics of the game, including the intricate environments, obstacles, and the marble itself.

1.4. React Renderers for Three.js:

- @react-three/fiber: A reconciler that allows you to write Three.js scenes declaratively in React.
- @react-three/drei: A collection of useful helpers and abstractions for @react-three/fiber.
- @react-three/rapier: A physics engine integration for @react-three/fiber.
- Role in Marble Run: These libraries bridge the gap between React and Three.js, enabling the efficient creation and manipulation of 3D elements within the React framework.

1.5. Zustand:

- Purpose: A minimalistic state management solution that's simple, fast, and scalable.
- Role in Marble Run: Zustand manages the game's state, such as player scores, game settings, and level progression, ensuring a smooth and responsive gaming experience.

1.6. Framer Motion & Framer Motion 3D:

- Purpose: These libraries provide powerful and easy-to-use tools for animations in React applications.
- Role in Marble Run: They enhance the game's UI and 3D elements with fluid, dynamic animations, adding to the visual appeal and user interaction experience.

1.7. Tailwind CSS (v3.3.3):

- Purpose: A utility-first CSS framework for rapidly building custom designs.
- Role in Marble Run: Tailwind CSS is employed for styling the game’s interface, ensuring a sleek and responsive design across various devices.

1.8. TypeScript (v5.1.6):

- Purpose: A typed superset of JavaScript that compiles to plain JavaScript, offering optional static typing.
- Role in Marble Run: TypeScript is used to write more robust code, reducing errors and improving maintainability by adding type safety to the game's development process.

1.9. Additional Libraries:

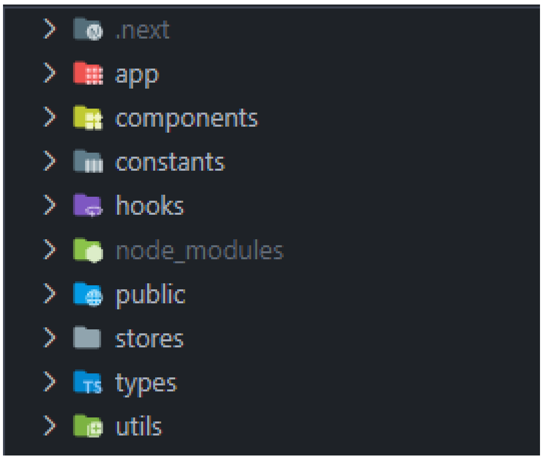
- For UI Components, Animations, and State Management: These additional libraries are integrated to further enhance the UI/UX, provide additional animation effects, and manage the game state more effectively.

In summary, the combination of these libraries and frameworks in "Marble Run" results in a robust, visually stunning, and interactive gaming experience. The use of React and Next.js provides a solid foundation for the UI and game structure, while Three.js and its related renderers bring the 3D world to life. Zustand, Framer Motion, and additional libraries contribute to dynamic state management and engaging animations. Finally, Tailwind CSS and TypeScript streamline the development process and enhance the overall design and performance of the game.

2. Code Structure and Explanation

The code structure of "Marble Run" is thoughtfully organized into several directories, each designated for a specific aspect of the game. This organization not only aids in maintaining a clean and modular codebase but also streamlines development and future scalability. Let's explore each of these directories and some key components in detail:

2.1. Directory Structure:



```
> .next
> app
> components
> constants
> hooks
> node_modules
> public
> stores
> types
> utils
```

2.1.1. `components/`:

- Purpose: This directory contains all the React components used in the game, which are crucial for both rendering the UI and managing the game logic.
- Key Components:
  - **`Player.tsx`**: Manages the marble's rendering and physics.
  - **`Lights.tsx`**: Responsible for the lighting in the game environment.
  - **`Timer.tsx`**: A component for the game timer.
  - **`GameContainer.tsx`**: Serves as the primary container for the game, integrating various components.

2.1.2. `public/`:

- Purpose: Stores all the static assets like textures, 3D models, and audio files, which are essential for creating a visually and auditorily immersive experience.

2.1.3. `stores/`:

- Purpose: Includes files for state management using Zustand. This directory is key to managing the game's dynamic state, such as player scores, settings, and level progression.

2.1.4. `utils/`:

- Purpose: Contains utility functions and helpers, crucial for handling common tasks and calculations throughout the game.
- Example: `meshProps.ts` for defining properties of 3D meshes.

2.1.5. `hooks/`:

- Purpose: Houses custom React hooks that provide reusable logic across components.
- Example: `useSmoothTransform.ts` for smooth transformations in the game.

2.1.6. `constants/`:

- Purpose: Maintains constant values used across the project, ensuring consistency and ease of modification.

2.1.7. `types/`:

- Purpose: Holds TypeScript type definitions and interfaces, crucial for ensuring type safety and clarity in the codebase.

2.2. Key Components Explained:

2.2.1 `useGame` (Custom Hook):

- Functionality: Manages the game's state using Zustand, including the marble's position, game timer, and interactions with traps. Centralized state management is vital for a responsive gaming experience.

2.2.2 Trap Components (`BlockAxe`, `BlockLimbo`, `BlockSpinner`):

- Variety in Gameplay: Each component represents a different type of trap, adding unique challenges.
  - `BlockAxe` features swinging axes.
  - `BlockLimbo` involves low-hanging obstacles.
  - `BlockSpinner` includes rotating elements.

2.2.3 `BlockStart` and `BlockEnd`:

- Define Game Boundaries: These components mark the starting and ending points of each level, outlining the player's journey.

2.2.4 `Bounds`:

- Structural Component: Creates the walls surrounding the map, defining the play area and keeping the gameplay contained within a specific zone.

2.2.5 `Experience` and `GameContainer`:

- Rendering and Environment:
  - `Experience` manages the 3D rendering and overall game experience.
  - `GameContainer` encapsulates the `Experience` component and other UI elements, acting as the main game container.

2.2.6 `Interface` and `Timer`:

- User Interaction:
  - `Interface` provides UI elements like score display.
  - `Timer` manages the countdown for each level, contributing to the game's pace.

2.2.7 `Level`:

- Dynamic Gameplay: Responsible for generating unique and random maps, essential for ensuring a varied and unpredictable gaming experience.

2.2.8 `Lights`:

- Atmospheric Component: Manages the game's lighting, crucial for setting the mood and ensuring visibility of game elements.

2.2.9 `Player`:

- Central Interaction: Contains the logic for the player's marble, including controls and interactions with the game world, making it a key component for player engagement.

In summary, the codebase of "Marble Run" is meticulously organized and each component is purposefully designed to contribute to a seamless and immersive gaming experience. This structure not only facilitates efficient development and maintenance but also enhances the game's functionality, aesthetics, and player interaction.