**Game Design Document (GDD)**

**Button Press Game**

**1. Game Overview**

**1.1 Title**

Button Press Challenge

**1.2 Genre**

Casual, Reflex, Arcade

**1.3 Target Platforms**

PC (Windows, Mac, Linux)

Future porting to Web & Mobile

**1.4 Target Audience**

Casual gamers, students, and players of all ages seeking quick, engaging, reflex-based gameplay. Ideal for short gaming sessions or competitive play.

**1.5 Game Summary**

Floating Button Challenge is an engaging reflex game where players click a floating, animated cube that randomly relocates and changes color within a limited time. The game utilizes smooth animations, sound effects, and particle effects for feedback. The objective is to maximize the score within 30 seconds, challenging players’ timing and precision.

**2. Gameplay & Mechanics**

**2.1 Core Gameplay Loop**

* The game opens to a start menu with a "Play Now" button.
* When the player clicks "Play Now," a 30-second timer begins.
* A cube appears in the center, floating with a sinusoidal motion.
* Clicking the cube increments the score, changes its color, and repositions it randomly within a defined area.
* The game continues until time is up; then, a game over message appears with the final score.
* Players can restart by pressing "Enter."

**2.2 Player Controls**

* Mouse click: Click on the cube to earn points.
* Keyboard: Press "Enter" to start or restart the game.

**2.3 Objectives and Goals**

* Achieve the highest score possible within the 30-second window.
* Improve reaction time and mouse accuracy.

**2.4 Game Logic and Rules**

The cube moves vertically with a sine wave for visual dynamism.

Each successful click increases the score by one, changes the cube’s color, and moves it to a new random position.

The game ends when the timer expires, displaying the total score and a game over message.

**2.5 Difficulty & Progression**

The cube resets position and color randomly; speed remains constant, but you can add increasing speed or movement patterns in future updates for progression.

**3. Design and Visuals**

**3.1 Visual Style**

* Minimalist, colorful, with a focus on smooth animations.
* Background: A static scenic image ('3D Bg image.jpg') scaled to cover the screen.
* Interactive Cube: Bright, with changing colors for visual interest.
* Particle Effects: Bright spheres burst upon hits.

**3.2 UI Elements**

* Score Display: Top left, large, clear text.
* Timer: Top right, prominent, easy to read.
* Game Over Text: Centered, large, bold.
* Restart Instruction: Below the Game Over text: "Press Enter to Restart".

**3.3 Animation & Effects**

* Floating sinusoidal animation for the cube.
* Color change on successful click.
* Particle burst effects for visual feedback.

**4. Audio Design**

**4.1 Sound Effects**

Button Click: Plays sound ('Button click.wav') when the cube is clicked.

Background Music (planned): Loop ambient background for engagement.

Crash or failure sounds: Not needed here, but could be added for extended mechanics.

**4.2 Audio Implementation**

These sounds are triggered on specific events: clicks and game over.

Further audio enhancement could include pitch variation or spatial sound effects.

**5. Technical Specifications**

**5.1 Development Environment**

* Language: Python 3.x
* Engine: Ursina (OpenGL bindings)
* Additional libraries: Built-in Ursina functions, Python standard library modules.

**5.2 Core Components & Code Breakdown**

* Background Entity: Static background image.
* Cube Entity: Animated, clickable object.
* UI Texts: Score, Timer, Game Over messages.
* Audio: Click sound, background music.
* Game States: game\_started, timing, score variables.

**5.3 Main Loop & Update Logic**

* Update function manages:
* Floating animation via sine wave.
* Timer countdown.5.4 Performance Optimizations and Constraints
* The game is lightweight, relying on simple 3D objects and 2D overlay text.
* The sinusoidal floating animation is computationally inexpensive.
* Particle effects are short-lived and scaled for performance.
* The frame rate is capped at 60 FPS for smooth gameplay.

**5.5 Expandability & Future Features**

* Adding difficulty progression (increase float speed or spawn rate).
* Introducing multiple buttons or obstacles for added challenge.
* Implementing high score storage locally or online.
* Adding animations or skins for the cube and background.
* Extending to mobile with touch controls.

**6. User Interface & User Experience**

**6.1 Start Screen & Main Menu**

* Clear call-to-action button “Play Now” in a bright color.
* Background scene with game title and optional instructions.

**6.2 In-Game HUD**

* Score display: Top-left corner.
* Timer display: Top-right corner.
* Dynamic updates every second for the timer and with each score change.

**6.3 End Screen**

* Prominent "Game Over!" message.
* Final score display.
* Instruction: "Press Enter to Restart" or similar prompt.
* Optional: high score comparison if implemented.

**6.4 User Experience Considerations**

* Controls are responsive, with immediate feedback (color change, particles).
* Visual cues make it easy to see the cube’s position and state.
* Clear instructions for restart.

**7. Gameplay & Level Design**

**7.1 Difficulty Curve**

* Increasing the speed of the floating cube gradually (optional).
* Future levels could introduce more complex movements or multiple cubes.

**7.2 Challenges & Mechanics**

* Moving the mouse precisely to click the floating cube quickly.
* Reacting to quick position changes.

**7.3 Rewards & Punishments**

* Reward: Higher score, faster reactions, better eye-hand coordination.
* Punishment: Missed clicks do not count; collision detection isn’t needed here as the core mechanic is clicking.

**8. Monetization & Extensibility (Future scope)**

* In-game Ads: For mobile/Web versions.
* In-app Purchases: Skins, custom backgrounds, sound packs.
* Leaderboards: To compete with friends or globally.
* Achievements: Unlockable based on score milestones.

**9. Art & Asset List**

**Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Asset** | **Description** | **Usage** | **Notes** |
| Background Image | '3D Bg image.jpg' | Main background | Should be scenic, not distracting |
| Cube Model | 'cube' | Interactive object | Bright, visible colors |
| Button Click Sound | 'Button click.wav' | Feedback for clicks | Short, sharp sound |
| Music | 'background\_music.mp3' | Looping background music | Optional, for ambiance |
| Particle Effect | Sphere scaled to 0.1 | Visual feedback on click | Bright orange |

**10. Development Timeline & Roadmap**

**Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Tasks** | **Timeline** | **Notes** |
| Planning | Finalize game mechanics, art assets, sound assets | 1 week | Define scope and assets needed |
| Prototype | Implement basic interaction, UI, timer | 2 weeks | Focus on core mechanics |
| Alpha | Add particle effects, animations, sounds | 2 weeks | Test for bugs, performance issues |
| Beta | Polish visuals, UI, difficulty, testing | 2 weeks | Collect feedback, refine |
| Release | Final adjustments, deployment | 1 week | Publish to platform(s) |

**11. Testing & Quality Assurance**

**11.1 Testing Goals**

* Check for bugs (collision detection, timer accuracy).
* Verify controls are responsive and intuitive.
* Ensure performance stability at 60 FPS.
* Assess user engagement and challenge level.

**11.2 Playtesting & Feedback**

* Gather feedback on difficulty, visual clarity, sound cues.
* Adjust float speed, spawn rate, or scoring for balance.
* 11.3 Post-Launch Support
* Patch bugs and performance issues promptly.
* Add new features based on user feedback (e.g., themes, difficulty modes).

**12. Conclusion & Summary**

Floating Button Challenge is a simple yet addictive reflex game designed with engaging visuals and sound effects. Its core mechanic of clicking a floating, moving cube promotes quick reactions and hand-eye coordination.