SHIKASTUDIO

GAME DESIGN DOCUMENT

CHICK CHICK

Field	Recommended
Product name	CHICK CHICK
Project version	Release
Company	ShikaStudio
Engine	Unity 6000.3.9f1
Source Scan	Assets/Scripts, Assets/Level, Assets/Prefabs, Assets/Scenes
Author	Shika

MỤC LỤC

1. GAME OVERVIEW **4**

1.1 Title and high-level concept 4

1.2 Core vision pillars 4

1.3 Genre and benchmark titles 4

1.4 Platforms 4

1.5 Target audience 4

1.6 Unique selling points 5

2. CORE GAMEPLAY MECHANICS **5**

2.1 Core loop 5

2.2 Player objectives 5

2.3 Player actions 5

2.4 Progression layers 5

3. NARRATIVE AND WORLD CONTEXT **6**

3.1 Narrative elevator pitch 6

3.2 World building elements 6

3.3 Tone and mood 6

4. SYSTEMS DESIGN **6**

4.1 Grid and level data architecture 6

4.2 Object and ground matrix 6

4.3 Interaction rules 7

4.4 Failure, reset and persistence 7

4.5 UI and menu flow 7

4.6 Audio hooks 7

5. VISUAL & AUDIO **7**

5.1 Art style reference 7

5.2 Readability requirements 7

5.3 Audio direction 8

6. LEVEL DESIGN **8**

6.1 Board and path design 8

6.2 Current level progression 8

6.3 Difficulty curve principles 8

7. MONETIZATION **9**

8. TECHNICAL REQUIREMENTS **9**

8.1 Current architecture 9

8.2 Estimated PC specs 9

8.3 Known technical debt 9

9. PRODUCTION PLAN **9**

9.1 Definition of done for vertical slice 9
9.2 Roadmap 10

1. GAME OVERVIEW

CHICK CHICK is a minimalist 3D grid-based puzzle game where players control a small chicken through compact maze levels to collect keys, gather coins, and unlock the exit door. Rather than relying on fast reflexes, the gameplay emphasizes reading the level layout, planning action sequences, pushing blocks, activating switches, and mastering the unique mechanics introduced in each stage. The current prototype includes a Main Menu, Level Select, and Gameplay Scene, along with a PlayerPrefs-based level unlock system and 12 LevelData assets. The implemented gameplay mechanics include grid movement, keys and exit doors, coins, block pushing, pressure plates with side doors, standard traps, timed traps, chasing traps, ice sliding, portals, and black/white color-switching ground mechanics.

1.1 Title and high-level concept

"A charming grid-based puzzle adventure with progressively deeper mechanics."

Players move one tile at a time, overcoming obstacles and hazards while solving compact puzzles to reach the exit through a series of thoughtful and precise decisions.

1.2 Core vision pillars

- **Readable Puzzles First**

Each level should clearly communicate its objective, possible routes, hazards, and interactive elements within the first few seconds, allowing players to understand the challenge before acting.

- **One New Idea at a Time**

Early levels introduce fundamental mechanics such as keys, doors, and basic traps. Later stages gradually expand the gameplay with block pushing, pressure plates, portals, ice sliding, chasing traps, and color-switching ground. Small environmental hints at the end of each level subtly foreshadow the mechanic introduced in the next one.

- **Cute Pressure, Not Punishment**

The visual style and audio maintain a lighthearted, playful atmosphere, while traps, locked doors, and pursuing enemies create enough tension to keep players engaged without becoming overly frustrating.

- **Compact Level Runs**

Each level is designed to be completed within a short play session, encouraging quick retries whenever players fail or choose an inefficient solution.

1.3 Genre and benchmark titles

CHICK CHICK is a grid-based puzzle adventure featuring elements of Sokoban, maze puzzles, and hazard avoidance. Its design draws inspiration from classic tile-based puzzle games, block-pushing games, casual maze adventures, and level-based mobile or web puzzle games that introduce new mechanics progressively.

1.4 Platforms

The primary target platforms are Mobile (Android & iOS). The game has been redesigned for a 1080 × 1920 (portrait) resolution, with the interface and gameplay optimized for touchscreen devices. The current build supports both mobile devices and desktop web browsers, although the intended player experience is focused on mobile platforms. The WebGL version serves as a convenient platform for testing, showcasing, and browser-based gameplay.

1.5 Target audience

- Casual players who enjoy short, accessible puzzle games with satisfying "aha!" moments.
- PC and web players looking for quick gameplay sessions that emphasize **show, don't tell** level design.
- Younger players and families, provided that the early-game visuals, audio, and difficulty curve remain approachable.

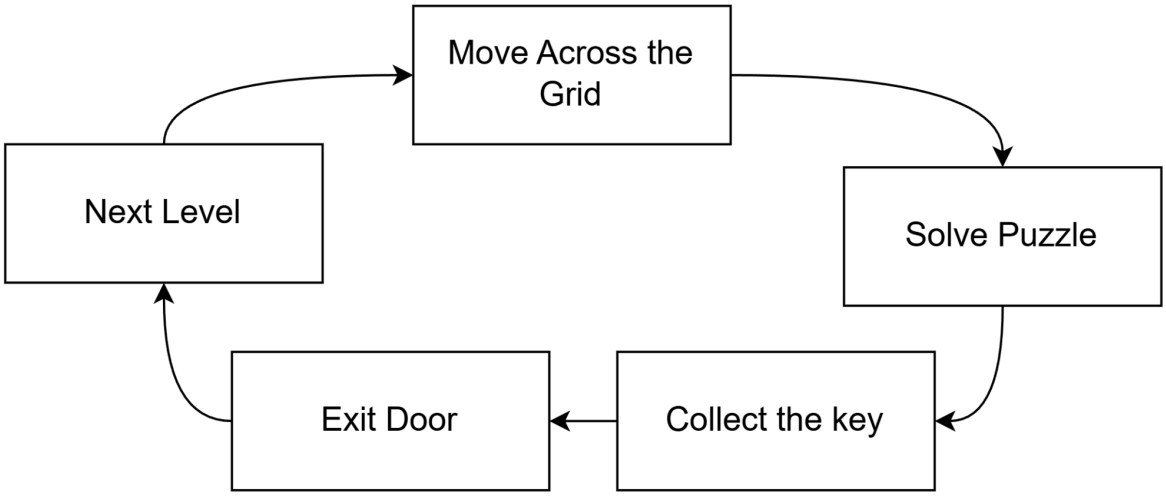
1.6 Unique selling points

The core strength of **CHICK CHICK** lies in combining a wide variety of puzzle mechanics within a simple grid-based movement system. Although players use only the WASD or Arrow Keys throughout the game, each group of levels encourages them to interpret the map differently.

One of the game's signature mechanics is the Black & White Ground System. By pressing Space, players switch between black and white states, allowing them to interact with corresponding tiles. This mechanic adds an additional layer of strategic thinking—players must consider not only where to move, but also which color state they should be in before stepping onto specific areas.

2. CORE GAMEPLAY MECHANICS

2.1 Core loop



1. The player selects an unlocked level from the Level Select scene.
2. The player uses WASD or the Arrow Keys to move across the grid.
3. The player interacts with the level by collecting keys and coins, pushing blocks, activating pressure plates, avoiding traps, and using mechanics such as portals, ice tiles, and black/white color-switching ground.
4. The ultimate objective is to collect the key and reach the exit door to complete the level.

2.2 Player objectives

- **Primary Objective**
  Collect the **key** and unlock the **main exit door** to complete the level.

- **Puzzle-Solving Objective**
  Determine the correct sequence of interactions to complete the level efficiently. For example, the player may need to push a block onto a pressure plate to open a side door before accessing the key or the exit.

2.3 Player actions

Action	Input	Design purpose
Move	WASD or Arrow Keys	Navigate the grid, explore the maze, and avoid hazards.
Hold Move	Hold a movement key	Allows continuous movement across long paths while preserving tile-by-tile navigation.
Push Block	Move into a block	Create puzzle solutions by repositioning blocks and activating pressure plates.

Switch Color Mode	Space	Cycle between Normal, White, and Black modes in levels featuring color-switching ground.
Pause / Menu	UI Button	Pause the game, resume gameplay, or return to the Main Menu.

2.4 Progression layers

The game's progression is **linear and level-based**. Player progress is stored using **PlayerPrefs**, where the highest unlocked level (`MAX_LEVEL`) is saved. In the **Level Select** scene, players can only access levels that have already been unlocked.

Difficulty increases progressively through both **larger map sizes** and **more complex gameplay mechanics**. Current **LevelData** assets range from compact **4×4** layouts to larger stages such as **12×12**, **15×12**, **20×11**, and **13×13**, allowing each new level to introduce greater spatial complexity and more advanced puzzle combinations.

3. NARRATIVE AND WORLD CONTEXT

3.1 Narrative elevator pitch

CHICK CHICK follows a curious little chicken that has wandered into a mysterious series of puzzle chambers. Each room presents a unique challenge filled with traps, portals, slippery ice, switches, and hidden paths. To escape, the chicken must collect the key, overcome the room's obstacles, and unlock the exit door leading to the next challenge. The narrative is intentionally lightweight, allowing gameplay and environmental puzzles to take center stage while giving players a simple motivation to continue progressing.

3.2 World building elements

Main Character

The protagonist is a charming low-poly cube-style chicken designed with a clean and minimalist appearance. Throughout the game, the character can switch between three visual states:

- Normal Mode
- White Mode
- Black Mode

These color states are directly tied to the gameplay through the Color Ground mechanic.

Puzzle Chambers

The game takes place inside a sequence of grid-based puzzle rooms. Every tile may contain different gameplay elements, including:

- Standard floor
- Walls
- Keys
- Coins
- Exit doors
- Side doors
- Pressure plates
- Pushable blocks
- Ice tiles
- Portals
- Standard and timed traps
- Chasing enemies
- Black and white color ground

Each room is designed as a self-contained puzzle that introduces or combines mechanics in increasingly complex ways.

Interactive Objects

- Keys unlock the main exit door.
- Pressure Plates activate side doors or other environmental mechanisms.
- Portals instantly transport the player between connected locations.
- Ice Tiles force continuous movement until an obstacle is reached.
- Color Ground can only be crossed while the player is in the matching color state.

3.3 Tone and mood

CHICK CHICK aims for a lighthearted, playful, and approachable atmosphere. The colorful environments, simple geometric art style, and cheerful sound effects create a welcoming experience for players of all ages.

While traps, environmental hazards, and chasing enemies introduce moments of tension, the game avoids creating a stressful or horror-like atmosphere. Instead, failure is treated as part of the learning process, encouraging players to experiment, retry, and discover better solutions.

The overall mood balances cute visuals with thoughtful puzzle-solving, making each completed level feel rewarding without overwhelming the player..

4. SYSTEMS DESIGN

4.1 Grid and Level Data Architecture

Each level is built on a tile-based grid defined by its x and y. Every tile stores both a ground layer and an object layer, allowing gameplay elements to coexist on the same position.

The level system is entirely data-driven, enabling designers to create and modify stages without changing gameplay logic. Objects and ground tiles are placed through level data assets, making it easy to introduce new puzzles while maintaining a consistent workflow.

4.2 Object and Ground Matrix

Type	Examples	Gameplay role
ObjectType	Wall, Key, MainDoor, Coin, Block, SideDoor	Occupy tiles, block movement, serve as collectibles, unlock progression, or interact with puzzle mechanics.
GroundType	Trap, PressurePlate, TimedTrap, ChasingTrap, Ice, Portal	Define tile behavior such as hazards, environmental triggers, movement modifiers, or teleportation.
Color Ground	WhiteGround, BlackGround	Can only be traversed while the player is in the corresponding color state.
Reserved enum	MovingTrapHorizontal, MovingTrapVertical	Reserved for future updates and currently not implemented in the prototype.

4.3 Interaction Rules

- Walls block movement completely.
- Blocks can be pushed if the destination tile is valid.
- Collecting a Key unlocks the ability to open the Main Exit Door.
- Pressure Plates activate linked mechanisms such as Side Doors when occupied by either the player or a movable block.
- Portals instantly teleport the player to their paired destination while preventing immediate teleport loops.
- Ice Tiles cause the player to continue sliding until reaching an obstacle or another valid stopping condition.
- Standard Traps defeat the player immediately upon contact.
- Timed Traps alternate between safe and dangerous states.
- Chasing Traps actively pursue the player, creating dynamic pressure during puzzle solving.

4.4 Failure, Reset And Persistence

When the player is defeated, the current level resets to its initial state. All puzzle objects, traps, and collectibles return to their original positions, while any stars collected during the failed attempt are restored to the level's starting value. Player progression is saved by unlocking completed levels, allowing future attempts to begin from the highest unlocked stage rather than restarting the entire game.

4.5 UI and Menu Flow

The prototype consists of three primary scenes:

- **Main Menu** – Start the game or access the level selection screen.
- **Level Select** – Displays all levels and allows players to choose any unlocked stage.
- **Game Scene** – Contains the gameplay interface, including the coin counter, restart button, and pause menu with Resume and Main Menu options.

4.6 Audio Hooks

The game uses dedicated sound effects for player movement, block pushing, key collection, coin pickup, portals, ice sliding, traps, locked doors, level completion, and chasing enemies.

To reduce repetition, multiple sound variations can be assigned to the same action, with the system selecting clips randomly while avoiding consecutive repeats. Background music loops continuously to maintain a lighthearted and consistent atmosphere throughout gameplay.

5. VISUAL & AUDIO

5.1 Art Style Reference

The current prototype uses simple geometric prefabs, solid-colored materials, and toon-style shaders to achieve a clean, readable visual style.The intended art direction is a minimalist low-poly 3D aesthetic with an isometric/top-down camera. Every gameplay element should have a distinct silhouette and color to ensure instant recognition.

Recommended color language:

Element	Visual Identity
Trap	Red / Black
Ice	Cyan
Portal	Purple
Pressure Plate	Green
Key	Gold
Coin	Gold or Cyan
Side Door	Green
Main Door	Brown / Wooden

Player	Bright Yellow

5.2 Readability requirements

Visual clarity is a core design principle throughout the game.

- The player character must remain highly visible against the floor and never blend with blocks or collectibles.
- The Main Exit Door should clearly communicate its two states: Locked and Unlocked after the player collects the key.
- Timed Traps require obvious activation and deactivation animations so players can accurately judge safe timing.
- White Ground and Black Ground must clearly indicate which player color state is required, especially in the later levels where the mechanic becomes central to puzzle solving.
- Chasing Traps should have clear movement feedback so players understand that they are actively pursuing them through valid paths.
- Interactive objects should remain recognizable at a glance, allowing players to understand puzzle layouts within the first few seconds of entering a level.

5.3 Audio direction

The audio style should be light, playful, and responsive, reinforcing player actions without becoming distracting.

- Movement and block pushing should provide rhythmic feedback that complements the grid-based gameplay.
- Coin and key pickups should feel rewarding with bright, satisfying sound effects.
- Portals and ice tiles should have unique audio identities so players can immediately recognize these mechanics.
- Traps and player death should deliver clear feedback while avoiding harsh or overly dramatic sounds that conflict with the game's cheerful tone.
- Background music should maintain a casual, upbeat atmosphere, supporting concentration during puzzle solving while remaining subtle enough not to overpower gameplay sound effects.

The overall audio experience should reinforce the game's approachable, family-friendly personality while providing clear feedback for every important gameplay interaction.

6. LEVEL DESIGN

6.1 Board and path design

Each level is built on a 1×1 tile-based grid, where every tile serves a deliberate gameplay purpose. Smaller 4×4 layouts introduce the core gameplay loop, while larger stages gradually expand into more complex puzzles featuring branching paths, linked mechanisms, and layered objectives.

Because player movement is tile-based, every obstacle should contribute meaningfully to the puzzle by:

- Blocking or redirecting movement.
- Encouraging exploration of alternative routes.
- Introducing or reinforcing a gameplay mechanic.
- Creating meaningful risk-versus-reward decisions.

Levels should avoid unnecessary empty space, ensuring that every tile supports the intended puzzle experience.

6.2 Current Level Progression

Level Design (Canva): https://canva.link/zv9wdw2e0r8z8fd

Level Group	Implemented Focus	Design Purpose
LV1 – LV4	Keys, Main Door, Walls, Coins, Traps, introductory Block, Ice, and Portal mechanics	Small, straightforward levels that teach movement, objectives, and the core gameplay loop.
LV5 – LV8	Pressure Plates, Side Doors, multiple linked mechanisms, larger layouts	Introduce sequential puzzle-solving through block placement, environmental interactions, and branching paths.
LV9 – LV10	Portals, Ice Tiles, Timed Traps, extended layouts	Shift the challenge toward spatial reasoning, route planning, and movement timing.
LV11 – LV12	Black & White Ground, Color Switching, Chasing Traps, combined mechanics	Advanced puzzle levels that require players to combine multiple mechanics while managing dynamic hazards. Strong visual guidance is essential.

6.3 Difficulty Curve Principles

The game's difficulty should increase through mechanic complexity, not simply larger maps or additional hazards.

- Every new mechanic should first appear in a safe environment where players can learn it without significant punishment.

- More advanced levels should combine previously introduced mechanics rather than introducing multiple unfamiliar systems simultaneously.
- Larger maps should naturally divide into distinct puzzle sections, such as obtaining the key, activating mechanisms, and returning to the exit.
- Black & White Ground should only appear after players fully understand the color-switching mechanic through clear UI and visual feedback.
- Chasing Traps should be used sparingly. Their purpose is to introduce time pressure and transform otherwise static puzzles into dynamic decision-making challenges, making them most effective as occasional highlights rather than constant obstacles.

7. MONETIZATION

The game follows a free-to-play model supported by advertisements.

- Rewarded Ads – Players can voluntarily watch an advertisement to receive additional coins or other optional rewards.
- Interstitial Ads – Displayed between completed levels or after several gameplay sessions to minimize disruption.
- Banner Ads (Optional) – Can be shown on non-gameplay screens such as the Main Menu or Level Select.

8. TECHNICAL REQUIREMENTS

8.1 Current Architecture

System	Current implementation	Risk / Next step
Engine	Unity 6000.3.9f1, URP 17.3.0	Stable for PC, Mobile, and WebGL builds. Additional testing is required on target devices.
Input	Keyboard controls (WASD / Arrow Keys) with mobile touch controls	Continue refining the touch interface and consider abstracting the input layer using Unity's Input System for future scalability.
Levels	ScriptableObject-based LevelData assets	Add editor validation tools to detect duplicate cells, invalid object placement, or mismatched linked IDs.
Persistence	PlayerPrefs for unlocked levels and selected level	Suitable for the prototype. A structured save system should be considered for future content expansion.
Enemy AI	Grid-based BFS pathfinding for Chasing Traps	Performs well on current maps. Larger levels or multiple enemies may require further optimization.

8.2 Estimated System Requirements

Because **CHICK CHICK** uses a lightweight low-poly 3D art style, tile-based gameplay, and a limited number of active objects, the hardware requirements are modest.

Minimum Requirements

- OS: Windows 10 or later
- Processor: Dual-core CPU (Intel Core i3 or equivalent)
- Memory: 4 GB RAM
- Graphics: Integrated graphics with WebGL 2.0 / DirectX 11 support
- Storage: Approximately 200 MB available space

Recommended Requirements

- Processor: Intel Core i5 or equivalent
- Memory: 8 GB RAM
- Graphics: Dedicated GPU or modern integrated graphics
- Target Frame Rate: 60 FPS

The game is also designed for 1080 × 1920 portrait displays on mobile devices and supports WebGL-compatible desktop browsers.

8.3 Known Technical Debt

- Moving Trap (Horizontal) and Moving Trap (Vertical) are defined but have not yet been implemented.
- Some level assets contain duplicate tile placements, making debugging more difficult. An automated validation tool should be added to detect conflicting objects and ground tiles.
- The input system is still partially coupled to desktop controls and requires further refinement for mobile devices.
- Certain class and scene naming conventions may become ambiguous as the project grows and should be standardized.
- Final verification is required for gameplay visuals, including Main Door state transitions, Side Door activation, and Black & White Color Mode feedback to ensure consistency across all builds.