San Francisco State University
SW Engineering CSC 648/848
Milestone 1

October 2, 2024

Section 04 — Team 02

Members

Katy Lam	**Team Lead**
Arjun Singh Gill	**Back-end**
Matthew Aaron Weesner	**Back-end**
Niko Galedo	**Front-end**
Kevin Lam	**Front-end**
Kullathon “Mos” Sitthisarnwattanachai	**Git Master**
Arizza Cristobal	**Scrum Master**

Revision History

Version	Date	Summary
1.0	2024-10-02	First version submitted for instructor approval
0.2	2024-09-28	Update formatting and document style guide
0.1	2024-09-26	Initial draft proposed to Team Lead

Table of Contents

Executive Summary

With the rapid evolution of the gaming world, players are increasingly seeking games that offer engaging yet easy-to-learn mechanics and a casual, enjoyable experience. ***Click and Mortar*** is designed to provide players with exactly that, a fun, interactive, and accessible game that anyone can play without a steep learning curve. Built on the foundation of progression, the game ensures that players never feel left out, even if they take a break. Whether through active clicking or passive idling, players accumulate resources to upgrade their environment and further enhance their growth. Additionally, for those who thrive on competition, *Click and Mortar* offers leaderboards that display top performers across the server, fostering a sense of achievement for competitive players.

Click and Mortar is positioned to attract a wide range of players, from casual gamers seeking relaxation to competitive players striving for top ranks. What sets our game apart is its unique balance of playstyles, players can choose to engage casually or competitively without missing out on key content. The game fosters a strong community where players of all backgrounds can come together to strategize, share progress, and interact with each other. Unlike many games that focus solely on casual or competitive play, *Click and Mortar* provides a versatile experience that caters to both, creating a unified gaming community.

Our team is composed of passionate game developers, designers, and user experience experts, all driven by a love for creating immersive and enjoyable games. With diverse backgrounds in gaming, we aim to craft a product that captures a broad audience and appeals to all types of players. Whether through individual play or community-driven interaction, *Click and Mortar* promises to engage both players and viewers, making it a compelling project for investors.

Click and Mortar has the potential to reach millions of users in the gaming market by appealing to both casual and competitive gamers. Its unique approach ensures that no player is left behind, and its community-driven features foster engagement across the spectrum. Backed by a team with a passion for game development and a focus on user experience, *Click and Mortar* is positioned for growth and success in an ever-expanding industry.

Personas and User Stories

UP-1: Content Creator (Yuna, age 22)

P1: High

General Behavior

Enjoys playing games while interacting with her viewer. Candidate streams on Twitch at https://www.twitch.tv/yunenoms. Main game being Black Desert Online but has been moving towards streaming other games and other genres as well. Enjoys games with grinding aspects and goals in mind.

Interests

Progression and interaction with the audience, games where she can ask the audience for tips and suggestions but also offer her own challenges in terms of progression and how the game can be played. Wants to create a gaming persona for herself and expand to more variety while increasing viewer count.

Skills

Able to mod and stream her game at the same time. Adaptive to many types of genre and games and is quick to get used to gamerules and the challenges amongst the genres.

Pain Points

Her viewers only watch her for specific genres, and views can decline if she is not engaging enough with the content they are not used to.

User Story

As a growing streamer and content creator, she can only do so much with one game when expanding her viewers and who her audience are. Recently within the past couple months, she has been expanding and playing more casual games in order to interact with her viewers without needing to focus on the game too much. Having a more casual laid back style of gaming helps her with finding people who would watch her for her persona rather than her playstyle and game play but also a way for the audience to interact with her about games that she is playing.

UP-2: Casual Gamer (Simon, age 27)

P2: Moderate

General Behavior

Plays games as a way to relax while having videos playing in the background. Likes to make progression which does not need much thinking like a puzzle and much more straightforward playstyle.

Interests

A game that he can get back on and off without the fear of FOMO due to the game having limited time items, a game that he can pause at any time and get back on.

Skills

Able to adapt to any genre of a game. Retired competitive gamer, and can use strategic thinking to quickly advance in a grinding game than the average casual gamer.

Pain Points

Getting bored when the grind starts becoming harder and harder. As progression takes longer, it starts losing interest due to lack of content and repetitiveness.

User Story

From someone who played games competitively and requires a lot of grinding like Destiny 2, he has “retired” from the competitive grind and wants a game he can enjoy anytime without feeling like he is missing content when he is not on. Taking small steps and watching the progression overtime is what he likes to do to destress after a long day of working.

UP-3: Competitive/Completionist Gamer (Kyle, age 21)

P3: Low

General Behavior

Plays games that he can advance with and can show off his skills and achievements. Can get a bit aggressive and close minded with other people's ideas on how to play the game.

Interests

Games with a scoreboard that he can use to feel accomplished and look up towards when gaming. Likes progression features and being able to see big numbers.

Skills

Can learn every mechanic in the game and is flexible when finding his role and what to do. Loves to mid max anything and get the highest score possible.

Pain Points

If not given enough progression, can easily get bored. If the progression gets too easy, enough to lose his interest. Has to has some sort of way to compare stats.

User Story

As long as he's been gaming, he finds joy in being able to be competitive and flex his progression to other players. He enjoys finding meaning and accomplishments when he plays, so he can reflect back on how much he has advanced and how much he knows about the game. Likes to keep track of the game as much as possible to keep his name high up on the leaderboards.

Data Definitions

The following section outlines key data definitions for our game, focusing on high-level descriptions. As the project evolves, more detailed definitions will be added in future milestones.

Player

A single person who interacts with the game.

Usage

Manages resources and items, progresses through the game, and may have an associated user account for saving progress.

User Account

A profile associated with a player.

Usage

Stores player information such as username, progress, inventory, and achievements. Allows players to log in from different devices.

Resource

Basic elements that can be collected or generated.

Usage

Obtained through clicks or over time, displayed on cards, and used to create items or more advanced resources.

Card

Visual representation of a resource or item.

Usage

Displays information like resource type, quantity, and special attributes. Players interact with cards to manage resources.

Item

Objects created by combining resources.

Usage

Crafted using recipes, items can unlock new abilities, generate additional resources, or provide other benefits.

Recipe

A set of instructions for combining resources to create an item.

Usage

Defines the required resources and quantities needed for crafting specific items.

Inventory

The collection of resources and items a player possesses.

Usage

Allows players to view, manage, and utilize their collected resources and items.

Currency

In-game medium of exchange.

Usage

Earned through gameplay or achievements, used to purchase resources, items, or upgrades.

Achievement

Milestones or goals within the game.

Usage

Provides players with targets to aim for, rewarding them with resources, items, or currency upon completion.

Administrator

A user with elevated privileges.

Usage

Manages game content, user accounts, and system settings. Has access to administrative tools not available to regular players.

Click Action

The primary method of collecting resources.

Usage

Players perform click actions to gather resources displayed on cards.

Timer

A mechanism that tracks time-based events.

Usage

Regulates resource regeneration, cooldowns for actions, or timed bonuses within the game.

Upgrade

Enhancements that improve gameplay.

Usage

Purchased or earned to increase resource generation rates, unlock new features, or enhance existing capabilities.

Quest

Optional tasks or missions available to players.

Usage

Offers additional challenges and rewards, encouraging continued engagement with the game.

Level

A measure of a player’s progress.

Usage

Indicates experience or proficiency, often unlocking new content or abilities as higher levels are reached.

Energy

A resource that limits the number of actions a player can perform in a given time.

Usage

Consumed when performing certain actions, replenishes over time or through specific items.

Functional Requirements

FR-1: Player Progression System

P1: High

Description

Implement a comprehensive progression system that allows players to advance through levels or stages, unlocking new content and abilities as they progress.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22) UP-2: Casual Gamer (Simon, age 27) UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-2: Engaging Resource Mechanics

P1: High

Description

Develop resource collection mechanics that provide a satisfying grinding experience, enabling players to gather resources to create new items or unlock features.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22) UP-2: Casual Gamer (Simon, age 27)

FR-3: Customizable Avatars and Personas

P3: Low

Description

Allow players to create and customize their in-game avatars or personas, enhancing personal expression and helping content creators build their gaming persona.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22)

FR-4: Audience Interaction Features

P2: Moderate

Description

Integrate features that enable streamers to interact with their audience within the game, such as audience polls, suggestions, or challenges.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22)

FR-5: Pause and Resume Functionality

P4: Negligible

Description

Enable players to pause the game at any time and resume later without penalties, allowing for flexible gameplay schedules.

Relevant User Stories

UP-2: Casual Gamer (Simon, age 27)

FR-6: No Limited-Time Content

P1: High

Description

Design the game without limited-time items or events to prevent players from feeling pressured by time-limited content.

Relevant User Stories

UP-2: Casual Gamer (Simon, age 27)

FR-7: Dynamic Difficulty Adjustment

P1: High

Description

Implement a system that adjusts game difficulty and progression pace based on player performance to maintain engagement and prevent boredom.

Relevant User Stories

UP-2: Casual Gamer (Simon, age 27)
UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-8: Leaderboards and Scoreboards

P1: High

Description

Provide leaderboards where players can compare their stats, achievements, and progress with others to foster competitiveness.

Relevant User Stories

UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-9: Achievement System

P2: Moderate

Description

Introduce an achievement system that rewards players for reaching specific milestones, encouraging completionist behavior.

Relevant User Stories

UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-10: In-Depth Mechanics and Strategies

P2: Moderate

Description

Incorporate complex game mechanics and strategies that allow players to learn and master, appealing to those who like to optimize gameplay.

Relevant User Stories

UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-11: Mod Support

P3: Low

Description

Provide support for game modifications (mods), allowing players to customize and enhance their gameplay experience.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22)

FR-12: Variety of Game Content

P1: High

Description

Offer a wide variety of resources, items, and challenges to keep players engaged and prevent boredom due to repetitiveness.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22) UP-2: Casual Gamer (Simon, age 27) UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-13: Grinding Mechanics with Clear Goals

P1: High

Description

Design grinding aspects with clear, achievable goals to maintain player motivation and satisfaction during progression.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22) UP-2: Casual Gamer (Simon, age 27)

FR-14: Prominent Progression Metrics

P2: Moderate

Description

Display progression metrics and significant numerical indicators prominently to satisfy players who enjoy seeing tangible progress and big numbers.

Relevant User Stories

UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-15: Flexible Gameplay Styles

P1: High

Description

Allow for multiple gameplay styles to accommodate different player preferences, such as casual play, competitive play, or content creation.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22) UP-2: Casual Gamer (Simon, age 27)
UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-16: Optional In-Game Challenges and Events

P2: Moderate

Description

Offer optional challenges or events that players can participate in for extra rewards or achievements, without making them mandatory.

Relevant User Stories

UP-1: Content Creator (Yuna, age 22)
UP-3: Competitive/Completionist Gamer (Kyle, age 21)

FR-17: Accessibility Features

P2: Moderate

Description

Include features that make the game accessible to a wide audience, such as simple controls, clear instructions, and adjustable settings.

Relevant User Stories

UP-2: Casual Gamer (Simon, age 27)

FR-18: Balanced Progression Curve

P1: High

Description

Ensure that the game's progression curve is balanced to prevent it from becoming too easy or too difficult, maintaining player engagement.

Relevant User Stories

UP-2: Casual Gamer (Simon, age 27)
UP-3: Competitive/Completionist Gamer (Kyle, age 21)

Non-functional Requirements

Compatibility and Performance

NFR-1: Browser Support

The deliverable shall be supported on all modern browsers with emphasis on the desktop experience. The game must run smoothly on the latest versions of Chrome, Firefox, Safari, Edge regardless of the desktop operating system. A separate mobile-centric experience may be considered later on, such as a dashboard to monitor the player's progress.

NFR-2: Load Time

As a browser game, the application should not be too demanding and should be able to run on most modern computers. The game should load within fifteen (15) seconds on a standard broadband connection, with assets like textures and scripts optimized for size and lazy-loaded where applicable.

NFR-3: Concurrent Users

The game servers should be able to handle up to fifty (50) concurrent players per instance without significant performance degradation, ensuring smooth multiplayer gameplay.

Usability

NFR-4: Intuitive Controls

The game should provide easy-to-use controls, with support for mouse, keyboard, and touch inputs, ensuring accessibility for players on various devices.

NFR-5: Tutorials and Help

The game should offer in-game tutorials to ease new players into gameplay mechanics, with an option to skip or revisit them later.

Reliability

NFR-6: Session Persistence

In the event of a browser crash or accidental page reload, the game should allow users to continue from their last saved state, minimizing lost progress.

NFR-7: Server Stability

The back-end should guarantee high availability, with redundancy in place to prevent downtime affecting multiplayer game sessions.

NFR-8: Error Handling

The application should gracefully handle any unexpected errors, logging them and providing appropriate user feedback without crashing.

NFR-9: Fault Tolerance

If any service fails, the application should continue functioning with degraded performance but without crashing.

Security

NFR-10: Authentication and Authorization

User access should be controlled with role-based permissions, ensuring that only authorized users have access to sensitive data or features.

NFR-11: Cross-Site Scripting (XSS) Protection

Implement security measures to prevent malicious scripts from being injected through user input, such as chat or other interactive elements.

NFR-12: Cross-Origin Resource Sharing (CORS) Policy

Ensure that the game's assets and APIs are protected from unauthorized access through proper CORS policies.

Maintainability

NFR-13: Modularity

The game's code should be organized into modular components (e.g., UI, game mechanics, networking), allowing for easy updates and the addition of new features.

NFR-14: Code Readability

Code should follow consistent style guidelines and be commented adequately, making it easy to understand for future developers.

NFR-15: Enforced Coding Style

All code must adhere to the project's coding style guidelines, which will be automatically enforced when pushed to the remote repository.

NFR-16: Automated Testing

Unit tests shall be implemented, with automated test suites run on every commit to catch regressions early.

NFR-17: Containerization

The system should be capable of running within Docker containers, ensuring smooth transitions between the development, staging, and production environments.

Competitive Analysis

The inspiration for our project comes from games like *Clash of Clans*, *Satisfactory*, and *Factorio*. Our planned product distinguishes itself by combining the dynamic resource management and open-world factory building from these games.

Clash of Clans

Feature	**Clash of Clans**	**Our Planned Product**
Visiting Other Players' Base	Yes, visit for attacks and defense	Yes, visit and interact with other players' bases
Gameplay Pace	Fast-paced, season-based	Player-paced, no season pass
Account Integration	Facebook integration	No external account sync, in-game accounts for interaction
Base Expansion	Fixed grid, limited by space	Continuously expanding grid with endless base building
Competitive Play	PvP attacks on bases	PvP with a focus on base functionality comparison
Animation Style	2.5D animation	Heavy focus on 2D animation

Satisfactory

Feature	***Satisfactory***	**Our Planned Product**
Visiting Other Players' Base	No	Yes, visit and interact with other players' bases

Gameplay Pace	Continuous, open-ended	Player-paced, no season pass
Account Integration	No integration	No external account sync, in-game accounts for interaction
Base Expansion	Expansive, open-world	Continuously expanding grid with endless base building
Competitive Play	No, single-player	PvP with a focus on base functionality comparison
Animation Style	3D	Heavy focus on 2D animation

Factorio

Feature	***Factorio***	**Our Planned Product**
Visiting Other Players' Base	No	Yes, visit and interact with other players' bases
Gameplay Pace	Continuous, open-ended	Player-paced, no season pass
Account Integration	No integration	No external account sync, in-game accounts for interaction
Base Expansion	Large, open-world	Continuously expanding grid with endless base building
Competitive Play	Limited multiplayer	PvP with a focus on base functionality comparison

Animation Style	2D	Heavy focus on 2D animation

Summary

Our product stands out with its peaceful, creative gameplay that focuses on 2D animation and allows players to build and expand their base indefinitely without external account integration or seasonal pressure. Unlike *Clash of Clans*, which is heavily PvP-focused, our game emphasizes base functionality comparison without combat. While *Satisfactory* and *Factorio* are known for automation and open-ended play, our game offers continuous base growth combined with a more accessible player-paced progression and non-combative multiplayer interaction.

High-level System Requirements

Please note that this list reflects updates made to the previously planned infrastructure outlined in Milestone 0.

Front-end

- React for building the user interface.
- Chakra UI for the front-end components.
- Phaser.io for the game framework.

Back-end

- FastAPI for the back-end.
- SQLModel for the Python SQL database interactions (ORM).
- Pydantic for data validation and settings management.
- PostgreSQL as the SQL database.

Deployment Platform

- Docker for containerization of the application.
- Amazon Web Services (AWS) EC2 for hosting both the front- and back-end server.
- AWS S3 for static asset storage (e.g., textures, scripts).
- AWS CloudFront for content delivery and caching.

Build and CI Tools

- Pytest for Python unit tests.
- Playwright for end-to-end tests.
- ESLint, Prettier, and Black for enforcing code style and formatting.
- GitHub Actions for all CI and CD.

Supported Browsers

Per NFR-1: Browser Support, the deliverable shall support all major browsers, namely:

- Google Chrome (latest stable version)
- Mozilla Firefox (latest stable version)
- Microsoft Edge (latest stable version)
- Safari (latest stable version for macOS and iOS)

Team

Members

Per Milestone 1 specification, the list of team member names and their roles are repeated here:

Member	Role
Katy Lam	Team Lead
Arjun Singh Gill	Back-end
Matthew Aaron Weesner	Back-end
Niko Galedo	Front-end
Kevin Lam	Front-end
Kullathon “Mos” Sitthisarnwattanachai	Git Master
Arizza Cristobal	Scrum Master

Study Plan

Learning Plan: Phaser.io (Front-end Team)

- **Goal**: Understand the basics of Phaser.io for building 2D games.

Week 1: Phaser Basics

- **Objective**: Familiarize with the Phaser library and environment setup.
- **Tasks**:
    - Install Phaser and set up a basic game environment.
    - Explore the Phaser documentation and follow introductory tutorials (e.g., creating a simple scene, handling sprites, physics, and basic input).
    - Learn how to load assets (images, sounds) and render them in the game.
    - Experiment with creating interactive elements (buttons, movement).

Week 2: Game Mechanics

- **Objective**: Start implementing game mechanics relevant to the project.
- **Tasks**:
    - Learn how to handle collision detection and player movement.

- Implement basic UI elements like health bars, scores, and in-game menus.
- Experiment with Phaser's physics system (arcade physics).
- Begin integrating with the front-end (login/shop features, interactions with game objects).

Week 3: Advanced Phaser Concepts

- **Objective**: Dive deeper into more complex Phaser features.
- **Tasks**:
  - Study game states and scene transitions.
  - Implement animations and advanced user interactions.
  - Learn about multiplayer setup (if applicable).
  - Optimize performance, especially for mobile devices.

Learning Plan: SQL (Back-end Team)

- **Goal**: Understand SQL for database management and queries, with a focus on integration with the back-end.

Week 1: SQL Fundamentals

- **Objective**: Learn basic SQL syntax and commands.
- **Tasks**:
  - Set up a local SQL server and create a simple database.
  - Learn to create tables, insert data, and retrieve data using basic `SELECT` queries.
  - Understand primary keys, foreign keys, and relationships between tables.

Week 2: Intermediate SQL Concepts

- **Objective**: Work with more complex queries and database optimization.
- **Tasks**:
  - Study joins (`INNER`, `LEFT`, `RIGHT` joins) to retrieve data from multiple tables.
  - Learn how to use indexes for optimizing query performance.

    - Practice updating and deleting data using `UPDATE` and `DELETE` commands.
    - Learn about transactions and rollback for handling data integrity.

Week 3: SQL Integration with Back-end

- **Objective**: Integrate SQL database with back-end applications.
- **Tasks**:
    - Learn how to connect the back-end (Node.js/Express or equivalent) with the SQL database.
    - Write API endpoints that interact with the database for user data (login, registration, profiles).
    - Implement database security practices, such as SQL injection prevention.

Meeting Plan

- **Wednesday (All Hands, In-Person)**: 2:30 PM Sync Meeting
    - **Agenda**:
        - Front-end and back-end teams present progress.
        - Discuss blockers and plan for upcoming tasks.
        - Align team priorities and timelines, especially on integration between front-end and back-end.
        - Arjun provides guidance on using Jira and tracks team issues/tasks.
        - Mos assists the team with Git functionality, including creating pull requests (PRs) and managing issues.
- **Thursday (Front-end, Remote)**: Weekly Meeting
    - **Agenda**:
        - Discuss progress on Phaser.io and any front-end features.
        - Share learning experiences and coordinate tasks (e.g., UI styling, game mechanics, user interactions).
        - Review Niko, Ari, and Kevin's work on Phaser and CSS.

        - Plan tasks for the next week, focusing on integration with the back-end as needed.

- **Sunday (Back-end, Remote)**: Weekly Meeting
    - **Agenda**:
        - Discuss SQL progress (Matt) and back-end infrastructure (Mos, Arjun).
        - Matt and Mos sync on API development and database integration.
        - Plan next steps for integrating login/registration with SQL.
        - Identify blockers and request help from front-end or infrastructure support.

Project Management Tools

In addition, we will also be hosting recurring sessions to help educate the team using Jira and Git.

Jira

- Arjun will lead sessions on how to set up and manage Jira boards, tickets, and sprint planning.
- The team will learn to use Jira to track progress and assign tasks effectively.

Git/GitHub

- Mos will hold short Git workshops on how to manage repositories, create PRs, review code, and resolve conflicts.
- The team will regularly use Git for collaboration, submitting PRs and resolving issues through issues tracking.

9. Checklist

The following checklist is included to comply with the Milestone 1 specification.

Item	Status
Team found time to meet outside of class	DONE
Scrum Master shares meeting minutes with everyone after each meeting	DONE
Git Master chosen	DONE
Everyone sets up their local development environment from the team's git repo	DONE
Team decided and agreed together on using the listed SW tools and deployment server	DONE
Team ready and able to use the chosen back/front-end frameworks	DONE
Team lead ensured that all team members read the final M1 and agree/understand it before submission	DONE