Graduate Capstone Project: Group 7
Platform (Web Application) for Indie Game Developers

Dmitriy Sizionov: 2054923
Guo Tang: 2056042
Jingchen Yang: 2085260

Fairleigh Dickinson University
24/VF CSCI 6806 V3 Computer Science Graduate Capstone Project
Dr. Sourav Mukherjee
September 20, 2024

Contents

1 Introduction

For the last decade video game industry has passed a long way from a niche-market with a few game studios to a multi-billion global market with global software companies. Game industry revenue might reach almost 307 billion by 2029 from 200 billion in 2022 (fortunebusinessinsights.com, 2024). Rapid growth is supported by the leaders in the industry for instance Microsoft, Nintendo, Sony, etc. but also small game studious. The video game industry is very competitive according to (MIT, 2022) with a low barrier to entering the market due to the simplification of game engines, availability of marketplaces, and possibility to communicate with players directly. However, the biggest challenge for small companies is establishing a community around the game in the early stage (Morris, 2020), since there is no common place for it. Game platforms are mostly focused on AAA titles from big companies since they are generating the biggest traffic of players to them. Small developers become lost and forgotten in the shadow from the big labels. Therefore, this project aims to solve this issue and provide a platform for small game developers to publish their works and build a community of fans around them.

Small indie game studious do not have a place where they can easily test and get feedback about the game from players in the pre-release stage. Since they have limited scale and budget, they need to start increasing their exposure early in the game development process to build a potential player base or to receive feedback and assistance from external sources. Currently, their options are mainly limited to setting up their own blogs or posting updates on personal social media platforms (Morris, 2020). Therefore, the platform, proposed in this project, aims to provide the possibilities for potential players to access the information about the indie projects they care about in one place and helping developers address issues encountered during development, so they are no longer isolated and unsupported.

The platform creation is a challenging project since the balance of interests between developers and players should persist. The aim is to make the platform easy to use for both players and developers in order to make it intuitive for browsing and interacting with. Therefore, the platform creation requires storing a content of developers and players, building packages for several interfaces, and integrate AI-generated analysis reports.

Despite the technical challenges, the project has significant benefits for users. Game studious will be able to upload test builds, publish promotional videos, trigger event notifications, distribute digital materials (such as character designs and soundtracks), get feedback about the game on the pre-released stage, and receive AI-generated analysis reports with a fan's community opinion. Potential players and Indie game enthusiasts can be involved into the whole process game development from a early edge. Developer can tweak their games details directly target to their fans preference, with high frequent communication between game developer and users. This really help player get their ideal indie-game come to reality.

2 Related Works

There are some well-known game distribution platforms like Steam, GOG and others. They place and promote indie games but these platforms focus more on promoting completed works. And also give ability to release there early-access products and facilitating community interaction. However, due to the large number of commercial AAA games on these platforms, the attention and support for indie games are often insufficient. In this situation, many game developers choose to use official social media accounts to promote their games like on Instagram/Discord. But in this case player targeting is a weak point since a social media platform are not specialized on this segment. Moreover, the biggest challenge to cooperate with players, game developers have to send direct personal invitations to the players to gather their feedback, analyze it, and incorporate changes after. While it requires a lot of resources from Game studio. In other hand, they might skip this development part with public tests and releasing the game without it. This decreases the chance for a game to be successful at the end.

Therefore, the creation of a place where small games studious and players can meet each other and share their ideas, discuss some features, and interact with each other is required. The players will familiarize themselves with the pre-released indie game info, interact with developers, and contribute to the work-in-process projects. Developers will be able to focus on games development, without spreading resources on gathering the players from various platforms. The platform will be able to provide a one-stop service for developers to upload game, test new features, and interact with potential players.

Besides, our platform emphasizes a more tightly Indie-game, ensuring quality while focusing on specific genres and themes align with niche market like narrative-driven game or retro style.

3 Approach

The initial project is a platform (web application) where game developers can share their games in the development stage for testing/feedback gathering. This web application will primarily serve independent game developers or small indie game studios. The main focus is on facilitating the close communication with community on the early-stage game development. Independent game developers will be able to create a homepage for their games. On this page, developers can post descriptions of the game, patch notes, playtest events, trailers, and more. Developers with needs can upload stage-specific game test builds for player testing, and quickly receive ratings and feedback from interested players about that stage's results, thus improving the final product more effectively. Players would have a chance to familiarize themselves and test games before game release. They would be able to browse and track the development progress of games they are interested in, participate in testing, and directly contact developers through the platform to leave suggestions that help improve the game. The platform will provide a summary for each game based on the player's feedback by gathering them and summarizing it via AI in a short article. That helps players to have a brief overview of the game. And also, it gives game developers a summary of feedback.

The project aims to adopt an architecture similar to MVC (Model View Controller) for building a platform. It's a software design pattern commonly used in web development by dividing the interface into 3 interconnected elements. According to Figure №1, it is planned to have 2 interfaces for Game Developers and Players. Additionally, it is planned to use one of the AI's API to process a Players input (ratings and comments) to generate data insights. To execute this architecture are developed a Class Diagram illustrated in Figure №2. Main actors will have their own entities with related methods. Furthermore, the project will include Database architecture represented in Figure № 3.

The project will be performed with the following Software stack:

Front-End will include BootStrap to manage a web application to implement a website's HTML and CSS programming languages. To enhance a web application usability the front end will include React.js which uses JavaScrip language for styling websites. The Back-End project we currently using C (ASP.NET) and planning to shift to java(SpringBoot) as a major programming language with the support of SQL to communicate with Database. A Database will be implemented in MySQL to store the data as a relational database. Additionally, the project will include external APIs to interact with AI to process data.

We are also planning to design various algorithm to make the WebApp more easy to use and meet more specific requirement for both game developers and players.

For the Game recommendation, we are currently anticipating to use "Content Based Collaborative Recommendations". This technique will make the webapp smartly identify players' preferences based on the genres and themes of the games the players have already followed or bought. These preferences could include game categories, producer, game type, art style, genre and so on. Then prioritizing the displaying of games that could be more attractive for certain field players.

Apart from this, The "Elastic Search" is ideal for games searching and ranking, that allows the search result displayed based on a variety of filed including game rating, reviews, price .etc.

We also have some optional algorithm choices like, in order to prompt users engagement, we will using "Gamification" strategy. Applying ranking algorithm to implement a leader board, and update it in real-time, with rewards(badges, credits, title) offered to top users.

For webpage performance, we can use catching algorithm, pre-store popular or frequently used content (e.g., popular game assets or pages) in memory to speed up loading process.

And to moderate the communication environment of our game forum, we can use Natural Language Processing Algorithms , which could contain language detection, Moderation.

The Railway database will serve as the backbone of the indie game platform, enabling secure and efficient management of user data, game content, and platform interactions. The platform will utilize the database to store and manage player and developer profiles, game metadata, reviews, purchases, and gameplay analytics. The separation of Players and Developers allows for tailored functionalities, such as game creation by developers and gameplay interaction by players, while maintaining relationships between these entities, as demonstrated in Figure 3. This architecture ensures that the platform can scale with increasing content, while also supporting key features like game reviews, purchases, and analytics tracking.

4 Deliverables

The project deliverables will include:

- Diagrams: Use Case, Class, State, and Sequence Diagrams.
- Database Architecture: Tables and an Access Matrix.
- Milestones: Mid-term Demo Version Submission and Final Project Submission.

A detailed timeline is provided below:

Date	Week	Task
Sep 4	Week 1	Project Idea Generation
Sep 11	Week 2	Project Concept Development
Sep 18	Week 3	Preparation of Documentation
Sep 25	Week 4	Establishment of Development Environment
Oct 2	Week 5	Creation of Main Entities, Methods, and Database Tables
Oct 9	Week 6	Simple Front-End Concept
Oct 16	Week 7	Demo Version Demonstration
Oct 23	Week 9	Development of advanced Features
Oct 30	Week 10	Enhanced Front-End Development
Nov 6	Week 11	Testing and Debugging
Nov 20	Week 13	Presentation and Final Report Submission

5 Evaluation

Evaluation of deliverable tasks is planned to be performed by conducting weekly team discussions and presentation of functionality. Additionally, it is planned to have 2 weeks of testing and debugging to achieve high-quality products at the end of the project. To achieve this the team plans to develop and perform auto-test for the web application in order to test functionality. Additionally, the team plans to conduct user testing gather feedback about usability and user product satisfaction. It is planned to assess the core web application functionalities for non-error conditions. And also the user interface should be simple and aesthetically pleasing, making it easy for new users to get started and for experienced users to navigate quickly.

The following metrics will be used for evaluation:

1. Game Browsing and Discovery Metrics:

- Game search results should be relevant, accurate, timely.
- Filters should be applied instantly, and sorting should correctly rank games by criteria such as popularity, release date, or rating.

2. Game Uploading for Developers Metrics:

- Games should be uploaded within a reasonable time frame
- Developers should be able to upload metadata (game title, description, screenshots, etc.) within 1 minute without errors.

6 Risks

The project major risks are the dis-synchronization within the team, the complexity of the software stack, and the size of the project. To mitigate those risks the team plan to implement the following measures: The main risks include:

- **Team Synchronization**: The team plans to develop documentation. In order to have a clear plan what is the scope of the project and how it is going to be delivered. This approach aims to provide clear project structure with the ability to nominate a specific task to each team member and set deadline for delivery.
- **Stack Complexity**: Risk of stack complexity is planned to be mitigated by team members experience in various topics. The team has experience in web development and database maintenance. All are familiar with Python programming language and have knowledge of front-end syntaxes. Also, the team has hands-on experience with SQL in Databases.
- **Project Scope**: The size of the project is also a risk for the project completion, but the mitigation measure is to outline the project scope in detail before the project kickoff. And to be flexible during the project to adjust and allocate team resources in required direction.

7 Conclusion

The project aims to solve a real-world problem in the video game industry. In situation where game studious face a problem attracting players' attention to their projects on the early stage of game development. Since it requires a lot of resources from developers that might be directed to the game creation activity. Current platform mainly neglects small indies studious due to the insufficient size of the community around their projects. Meantime, players have a difficulty to find new interesting projects with fresh view on games. And don't have efficient channels for direct communication with game developers.

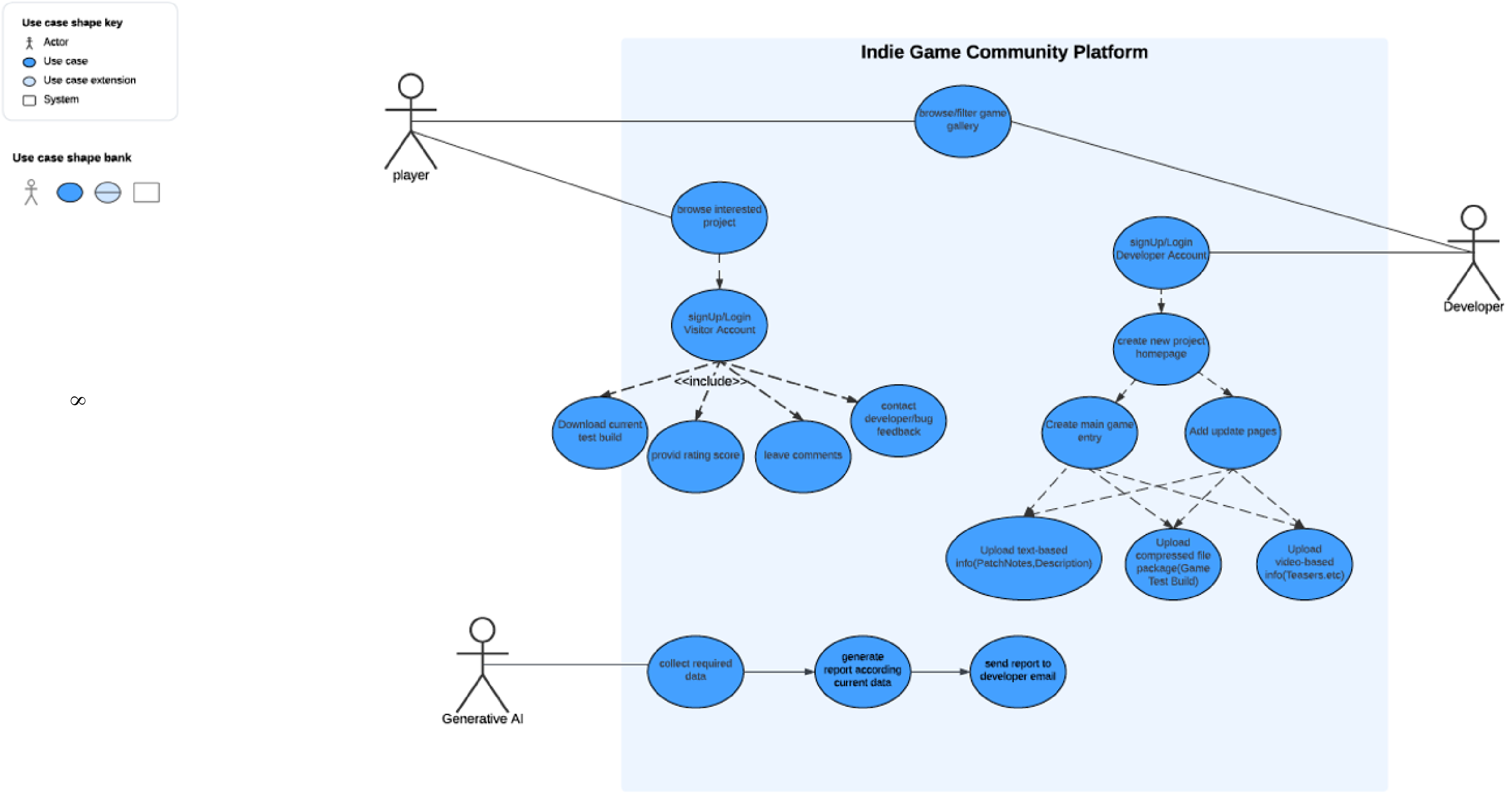
The project aims to develop a platform (web application) in accordance with industry standards and course deadline. That platform should be able to help developers to publish their game content and establish communication with the players community. It also should provide possibility for players to post their feedback about the games and interact with developers.

The Platform should work without technical errors, has a high usability for users and be intuitive in explanation.

8 List of Sources

- MIT OpenCourseWare. (n.d.). What's possible for the gaming industry in the next dimension? EY. https://www.ey.com/en_us/insights/media-entertainment/what-s-possible-for-the-gaming-industry-in-t
- Morris J. (2020 December 1). How to Build an Audience for Your Game as an Indie Developer on Social Media. Medium. https://mister-morris-games.medium.com/how-to-build-an-audience-for-your-game-as-an
- Video Games Market Size Industry Share Forecast 2032. Fortune Business Insights. https://www.fortunebusinessinsights.com/video-game-market-102548

A Appendix I: List of Figures


Use case shape key
Actor
Use case
Use case extension
System
Use case shape bank
Indie Game Community Platform
player
browse/filter game gallery
browse interested project
signUp/Login Visitor Account
<<include>>
Download current test build
provid rating score
leave comments
contact developer/bug feedback
signUp/Login Developer Account
create new project homepage
Create main game entry
Add update pages
Upload text-based info(PatchNotes,Description)
Upload compressed file package(Game Test Build)
Upload video-based info(Teasers.etc)
Developer
Generative AI
collect required data
generate report according current data
send report to developer email

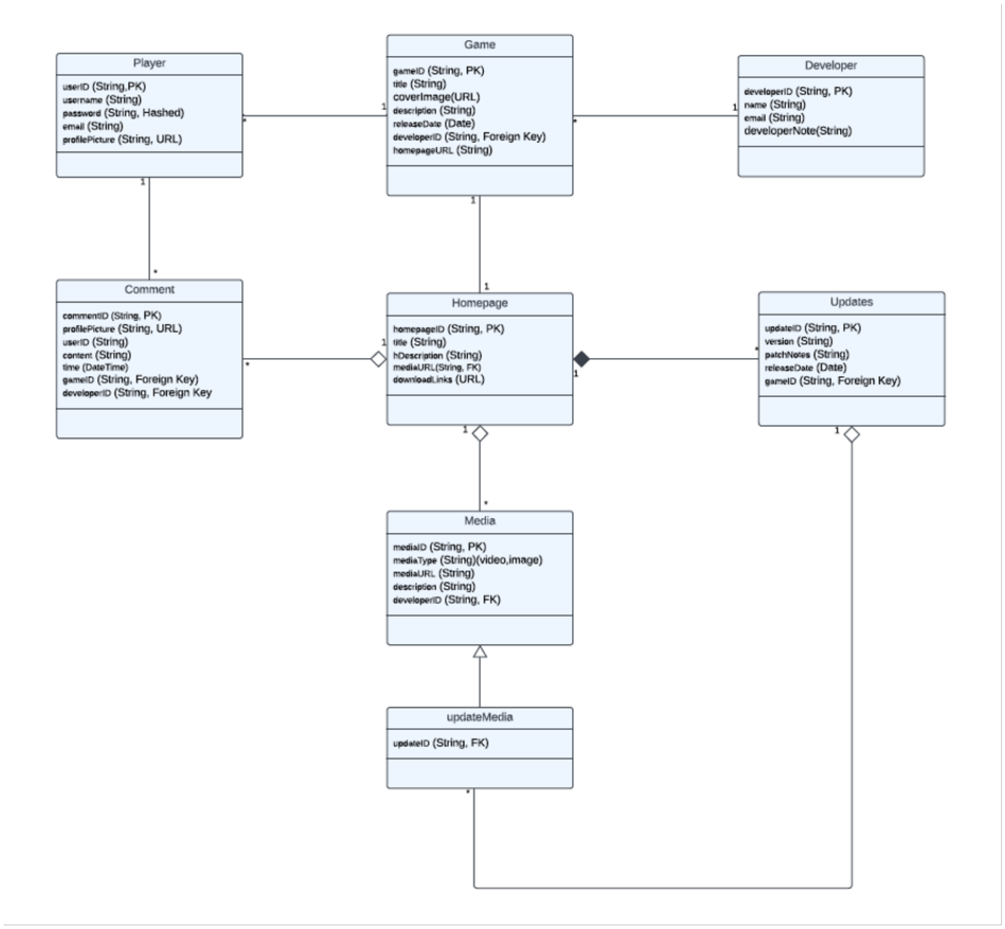
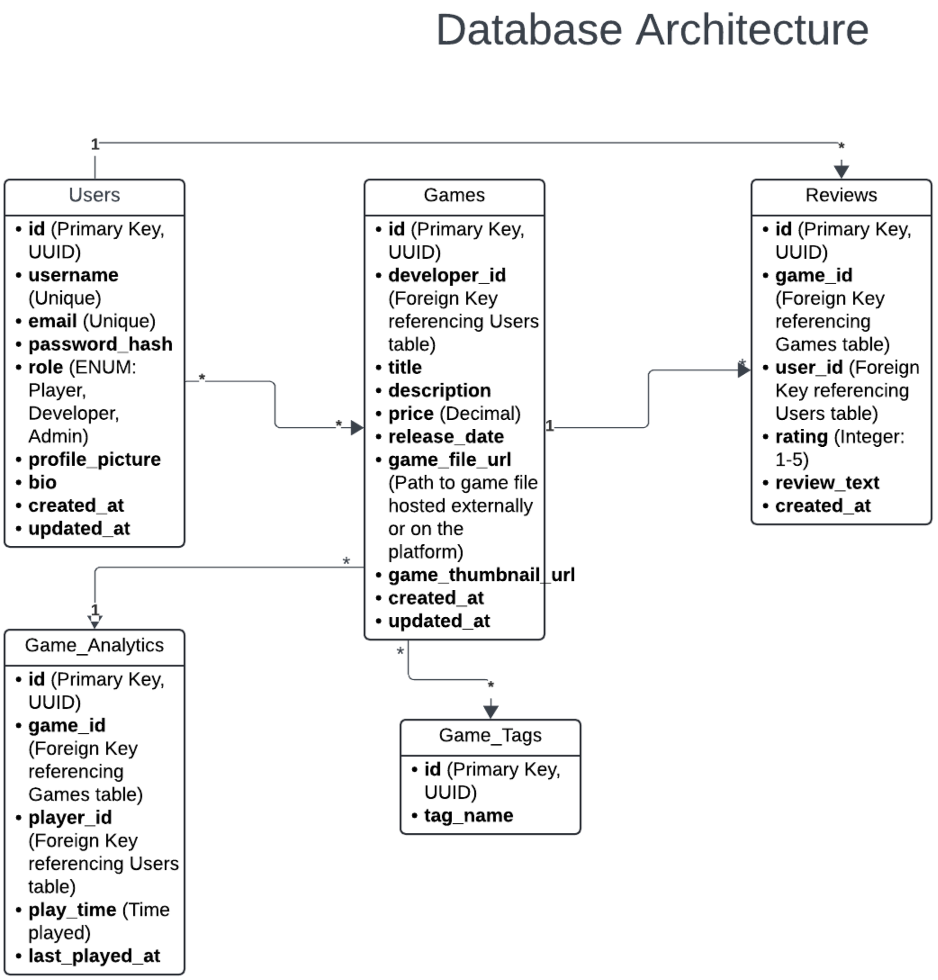


Figure 2: Class (Entity) Diagram



Users

- **id** (Primary Key, UUID)
- **username** (Unique)
- **email** (Unique)
- **password_hash**
- **role** (ENUM: Player, Developer, Admin)
- **profile_picture**
- **bio**
- **created_at**
- **updated_at**

Games

- **id** (Primary Key, UUID)
- **developer_id** (Foreign Key referencing Users table)
- **title**
- **description**
- **price** (Decimal)
- **release_date**
- **game_file_url** (Path to game file hosted externally or on the platform)
- **game_thumbnail_url**
- **created_at**
- **updated_at**

Reviews

- **id** (Primary Key, UUID)
- **game_id** (Foreign Key referencing Games table)
- **user_id** (Foreign Key referencing Users table)
- **rating** (Integer: 1-5)
- **review_text**
- **created_at**

Game_Analytics

- **id** (Primary Key, UUID)
- **game_id** (Foreign Key referencing Games table)
- **player_id** (Foreign Key referencing Users table)
- **play_time** (Time played)
- **last_played_at**

Game_Tags

- **id** (Primary Key, UUID)
- **tag_name**

Figure 3: Database Architecture