Treasure Can Dream: Magical Journey

11810417 田耕 11810112 姜润智 11910117 李浩博 11810418 田野 11910421 刘仝

2021 年 10 月 4 日

1 Abstract

Pokemon games are classic and fun, and we were very interested in making one. This project aims at completing a complete pokemon game, using Unity, Rider, PostgreSQL and other development tools, using C# and other programming languages for design. We have already designed UML use case diagrams, class diagrams, and other documents. We will continue fighting for a better project and make more efforts for this project.

2 Motivation

2.1 What is the problem?

How does C sharp work?
How does Unity work?
How does Unity and C sharp work together?
Where do we find the 3D model of Pokemon? (With the action
Which design patterns to use?

2.2 What is your vision for solving the problem?

Some problem can solved by searching online, and some can solved by reading textbook.

2.3 What are your solutions?

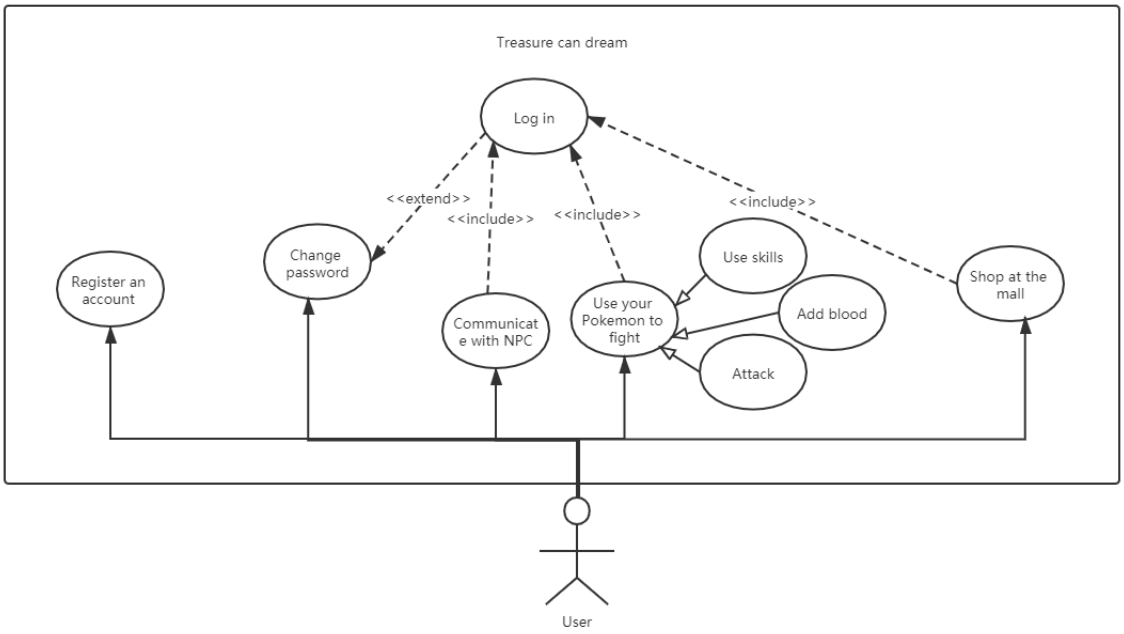
C sharp is very similar to Java, so it's easy to learn. We find some Unity tutorial on Bilibili.

We get some free 3D model of Pokemon in some websites.

We decided to use the observer mode, and we will exercise more.

3 Feature Description

3.1 Use case diagram



Use case diagram

3.2 Mockups

We use processon and datagrip to create our diagrams.

4 Requirements

4.1 Pokemon storage and cultivation

We will have a built-in Pokemon Home system to store and display the elves captured in battle. You can cultivate your own Pokemon in the garden, interact with them and give them minor stats.

4.2 Warchess turn-based combat

1V1 turn-based battles, complete with terrain and climate, pokemon can move around the six-square map and use skills to fight.

4.3 Shops and props

In order to diversify combat, we added shopping malls and item systems. First of all, you can buy or acquire various kinds of elf balls through events to capture elves, and then there are recovery drugs, antidotes and other combat AIDS to use in battle, and Pokemon possession items to improve their stats or change their combat disadvantages.

4.4 Character, attributes and state

Each Pokemon has its own unique attributes, personality and current status. There is a restraint relationship between different attributes, which can change the situation of disadvantage. Personality affects Pokemon's own attack and defense stats, and to a certain extent causes Pokemon not to listen to their trainers. The states, the negative states of intoxication, paralysis, parasitism or the positive states of inspiration, ascension, can have different effects.

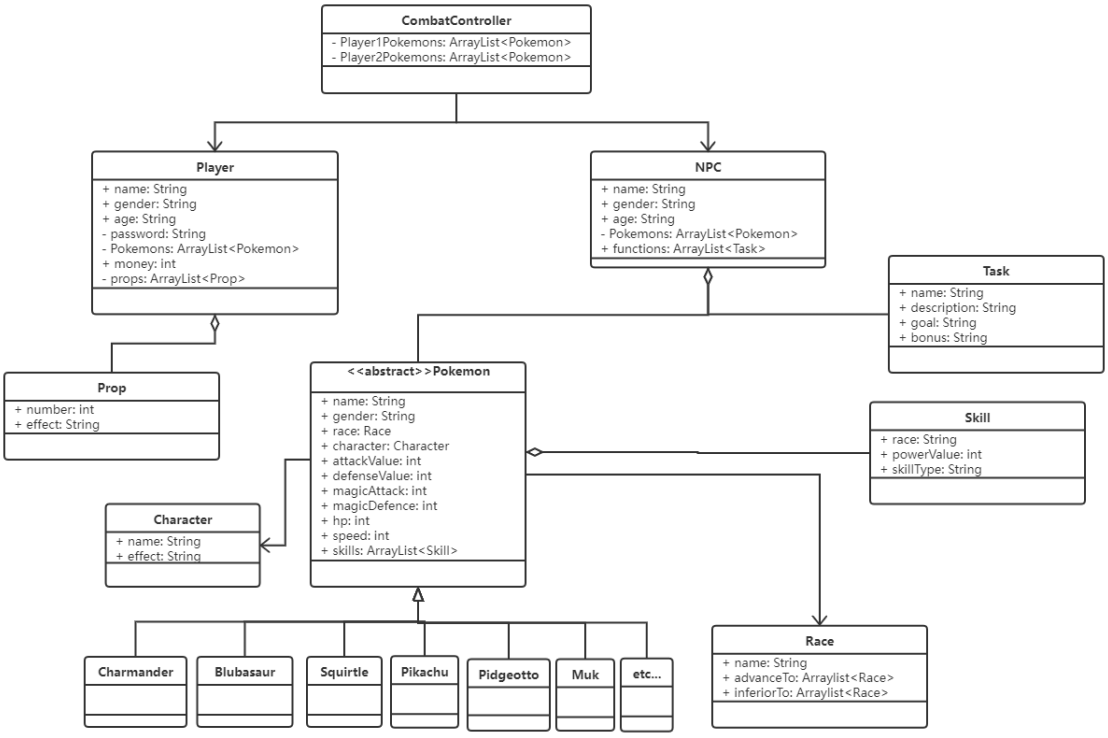
4.5 Multiplayer mode

Tired of fighting wild elves alone? Want to compete with your fellow players? We are launching an online battle system where you can accept invitations to duel with other players and play against their sprites in real time.

5 Design Document

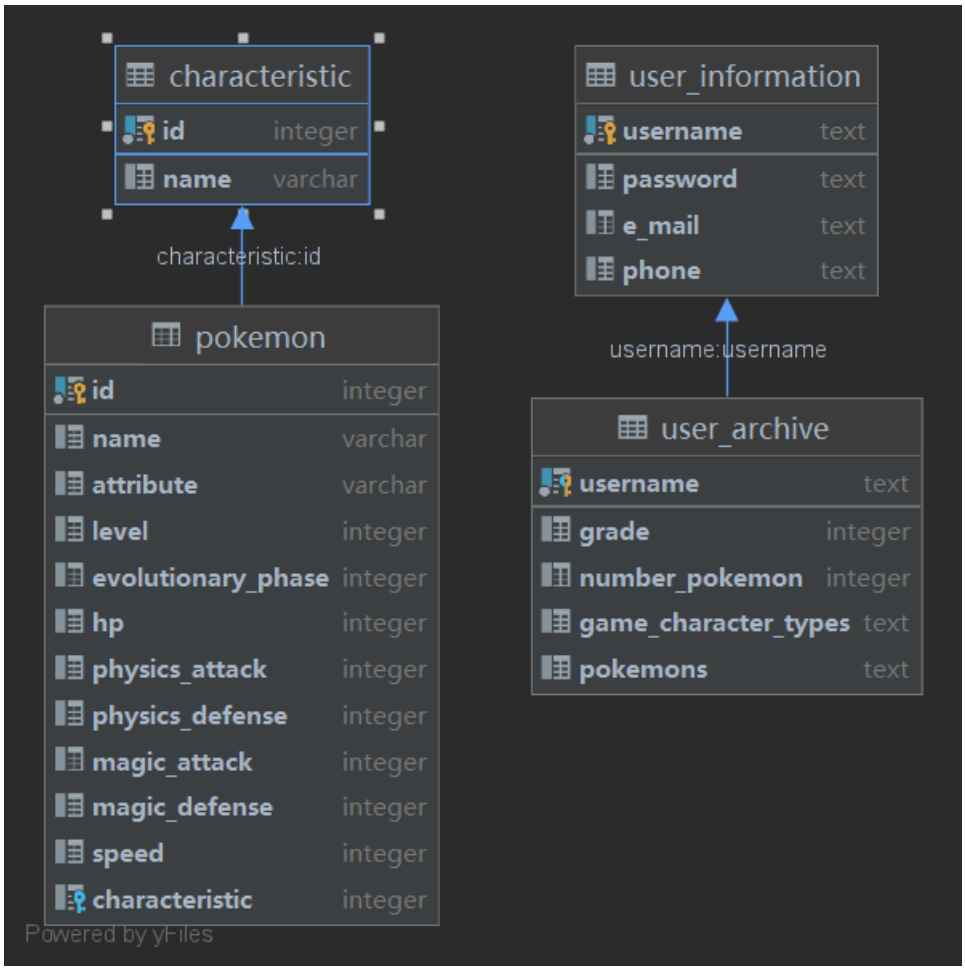
5.1 Architecture

5.1.1 Class diagram



Class diagram

5.1.2 Database schema



Class diagram

5.2 Timeline

10.1 10.7 holiday: looked for material, completed the opening report, created the project on Github

10.14: opening presentation

10.14 10.24: establishment of pokemons' models, design the combat process

10.25 11.7: design NPC and player system

11.7 half of November: enrich the game and debug

5.3 APIs, services

We use some packages that come with Unity, which have ready-made materials. These materials can help us build the game platform better and faster.

6 Feasibility

Many things will lead to the crash of the whole programme, just like the unmatchble modes and motions due to unfamiliar of the unity and the potential dangerous of database dropping. Unfamiliarity of APIs will lead to unreadable code and chaos. The lack of amount of pokemon will bring about incomplete combat system and world outlook which gives pity to whole project. Overly bloated code structure will lead to slow respond of computer which gives a high ping to combat process gives player very bad game experience. High concurrency will also be considered seriously because there may be several playser login and join in combat, we need to improve the robustness of the programmes in case huge amount of data crash the programme. In general, we need to get familiar with unity and other tools, and be proficient in using api, and accelerate our development process.