**Proposal for SUSTech Pokemon Project Part A**

1. **Project Title**

Pokémon: Special SUSTech Edition

1. **Executive abstract**

Design and implement a game centered on a pokemon combat system, which contains 6 pokemon evolution families and 16 pokemon categories. Pokemon properties include 6 basic statistics and different types, which decide the result of a combat together with chosen moves and activities. A pokemon growth system allows pokemons to gain particular experience from a combat to upgrade and evolve by unique paths, which improve their statistics and allow new moves. A store system provides product purchase function, where the player can use money gained by variable missions to purchase and use items with different effects. Wild pokemons are catchable in probabilistic to join the pokemon bench and Pokedex where to see information about pokemons. The game has a 3D art form and there is a explorable scene of SUSTech to trigger different game contents. PvE and local PvP modes are supported in the combat system.

1. **Team**

11910213 薛丁元

11910338 刘思岑

11911338 钟悦芸

11912627 孟宇阳

1. **Description**
2. **Motivation**

***The problems we met are but not limited to these:***

* It is hard for us to do a good art work.

We have had a long argument about art style. The problem was, if we chosen 2D, it was essential to make the picture beautiful and exquisite, which led a big problem to animation. If we chosen 3D, how to find applicable model and remake bones animation were also intractability. We thought needed figures that can easily change by ourselves, this mean we would most likely need to build our own models. Finally we got inspiration from Pokémon Quest, where the Q version of the models makes it easy to create and act, and their lovely appearance make the visual effect no less impressive. In the end, we decided our art style as Q version 3D.

* We had no basis for modeling.

We studied modeling everywhere. At first we tried the easier pixel modeling, but we thought it was not cute enough. Finally we chosen more difficult Blender as the modeling software for pokemons and Unity3D for scene, and for this purpose we took a long way to learn them.

* We had no basis for Unity software.

We also studied Unity everywhere. The group have taken more than 2 whole days to install, configure and repair the software and the environment. In fact, some of the team members still can not run Unity on their computers. We eventually decided to bypass this problem because the back end could be written directly with an external compiler so that this problem has little impact on our project schedule. We use JetBrains Rider to complete our rear-end coding.

* We had no knowledge of design patterns.

We started our project so early that the class hasn't reached the part of design patterns. We took some time to study it by reading Head First together. The good news was that the book is so well organized that the first three chapters cover almost all the design patterns we need.

* There are numerous pokemon moves, item functions and so on.

Consider the possibility of adding more Pokemon moves with items in the future, a flexible frame structure is essential. We use strategy pattern to solve this problem. And for the display of changing statistics, we found observer pattern matters.

* Game structure in Unity is seriously different from the Java we have used to.

At first, our rear-end framework has reserved for each interface switch, and for this changed the way some parameters passed in. Halfway through we realized that the page switching logic in Unity is independent of the rear-end. So we reset the underlying framework.

* We have a million questions about Unity's interface.

For example, we don't know how to get the world position of game objects, or how to find inactive game objects. For classes that inherit unit specific classes, we can't create instances through new, etc. These are all solved by CSDN

1. **Feature Description**

***User story (or the game process):***

The player can coontrol a pokemon model to walk in the SUSTech scene by WASD keys. When it arrives different places, it can choose to trigger different functions.

* The Student Dormitory

The player appears there when the game begins. Restore all pokemon status and save the game.

* The Activity Room and Mail Room

Collect rewards and Pokemon. Post tasks.

* Stores (contains Lychee Hill market, Lakeside market and Well)

Buy all kinds of items with money.

* Dining Halls (contains Lychee Hill dining hall, the second dining hall and center dining hall)

Buy all kinds of food that provide bonus attributes with money.

* Community Health Service Center

Buy medicine. Restore pokemon status with money.

* Teaching Buildings (contains Research Building, Teaching Building and Lychee Hill)

Choose PvE or PvP combat to join. When a combat start, player can choose 3 pokemons in the bag to the bench. Only one pokemon allows to be active at a time, and when a pokemon’s HP drops to 0 the next one will be active. Player can choose to use one move or use an item or escape from the combat per round. After that to the opponent's round. When one player’s 3 pokemons are all defeated, the combat over and show the results and rewards.

Change the moves that a pokemon carries.

* College of Engineering

Change a pokemon’s nature. Pokemon evolution. Complete tasks and work to gain money.

* Playground and baseball field

Fight with wild pokemons or catch them by pokeballs.

* Libraries (contains Lynn, Yidan and HanYong)

See Pokedex. See detail information about the game.

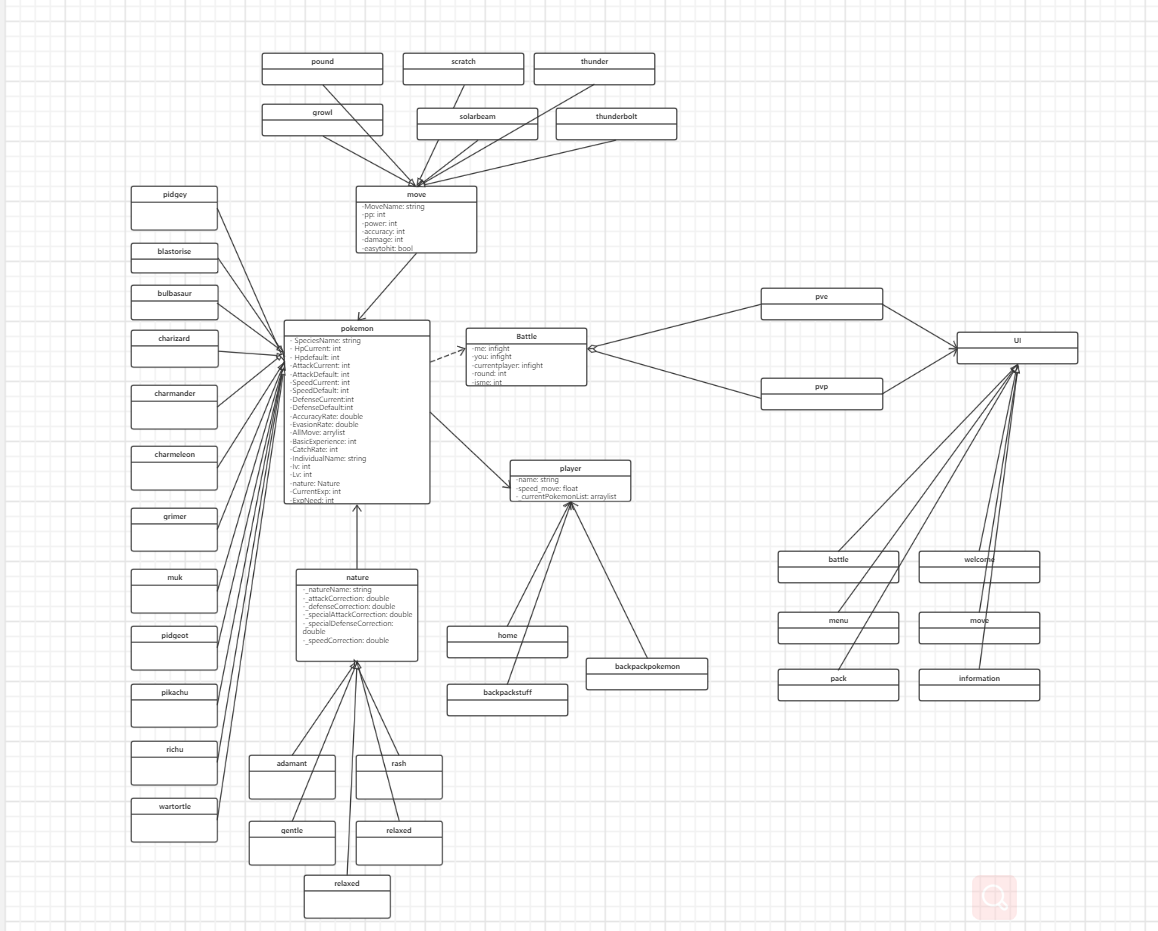
* The Event Center

See and Change the information about the player.

* Administration Building

Contact us.

***UML of rear-end:***



***Mockups:***

* For the scene model, we used Unity's own resource store to find a lot of good and beautiful models.
* For the models of Pokemon and Pokeball, we used Blender to make them one by one, very beautiful details.

1. **Requirements**

***The functional requirements are stated above.***

* We plan to support both PVE and PVP.
* And the character-controlled Pokemon can walk around the map and trigger the plot.
* We also support Pokemon to find wild pokemons on the map and capture them.
* We can use potions.
* We have special battle scenes and different pokemon properties.

***Non-functional requirements:***

* We are ready to support user archiving.
* We use The Unity engine to render the scene and keep the user's cost as low as possible.
* We are ready to package it into an .exe file and become a game APP.

1. **Design Document**

***Architecture:***

We have a lot of classes and a lot of inheritance, including UI interfaces. The class diagram of UML has been shown above, in *Feature Description.*

***Timeline：***

* Before 9.28 :

钟悦芸：Pinch the model, check the data

孟宇阳：Make the scene, check the data

刘思岑：Write code framework, check relevant information

薛丁元：Write code framework, check relevant information

* 9.28-10-13 :

钟悦芸：Write UI and Pokemon backpack

孟宇阳：Write combat scenes and optimize related scenes

刘思岑：Refine the code and complement the back-end code

薛丁元：Refine the code and complement the back-end code

* 10-14-10.22 :

All our members: Defense, writing reports, preparing related plot materials

* 10.22-11.18 :

钟悦芸：Optimize UI for multiple scenes and make story-related UI

孟宇阳：Optimize the combat scenario, optimize the connection to the back-end code

刘思岑：Optimize the code architecture and add story content to the game

薛丁元：Optimized backend code, optimized online and wild meet Pokemons content

* 11.18-mid-pre :

All our members: Optimize our game and prepare for our presentation

***APIs and services****：*

* Using unity as the game engine and C# as the back-end code can make the combination of the front and back end faster and improve the efficiency of our game.
* Use unity's built-in UI to write the UI, making the game easier to debug and run.
* In the back-end code, we write our own class and inheritance relationship, better combing the whole framework, making the game run more efficiently and better testing.

***Technologies:***

* Tools

Unity3D

Visual Studio - C#

JetBrains Rider - C#

Blender - modeling

PhotoShop

* Lirbaries

Various libraries required for front and back-end connections in Unity

Basic libraryies: System and so on

1. **Feasibility**

***We had no basis for modeling.***

We studied modeling everywhere. At first we tried the easier pixel modeling, but we thought it was not cute enough. Finally we chosen more difficult Blender as the modeling software for pokemons and Unity3D for scene, and for this purpose we took a long way to learn them.

***We had no basis for Unity software.***

We also studied Unity everywhere. The group have taken more than 2 whole days to install, configure and repair the software and the environment. In fact, some of the team members still can not run Unity on their computers. We eventually decided to bypass this problem because the back end could be written directly with an external compiler so that this problem has little impact on our project schedule. We use JetBrains Rider to complete our rear-end coding.

***We had no knowledge of design patterns.***

We started our project so early that the class hasn't reached the part of design patterns. We took some time to study it by reading Head First together. The good news was that the book is so well organized that the first three chapters cover almost all the design patterns we need.