

Proposal

1. Project title

SUSTechMajiang

2. Executive abstract

With the development of the number of students in SUSTech, there is an urgent need to develop more kinds of recreational activities for SUSTech students to relax and have fun. If there is a recreation game where the background and some game elements come from SUSTech, not only can it enrich students' campus life, but also it enhances students' campus identity pretty well.

Our group is devoted to develop an Online Mahjong game with SUSTech elements and try to fulfil this mission. It is an online game combined with the basic Mahjong rules and SUSTech background. Our user are supposed to download the client-side (.exe file) first and play with other users on the platform we developed.

This product is mainly target for the faculties and students in SUSTech and those who are interested in SUSTech. Since SUSTech is a rapidly developing university with increasing ranking and popularity, it should have huge potential market.

3. Team

卞证 11710901 向昕昊 11712617 徐天元 11710208 林傲 11711035 孙斌贤 11712814

4. Description

Motivation

- *What is the problem?*
- *What is your vision for solving the problem?*
- *What are your "silver bullets"?*

In our project, the most difficulty is how to combine the mahjong rules with SUSTech elements, which is our product's core competition. The following are some big events we will implement in the project:

1. All the background imagines of the game (game hall, front and back of cards, buttons, etc.) will be in SUSTech element.
2. Our background music will combine mahjong and SUSTech style.
3. Besides the 144 basic cards in mahjong, we have 8 extra cards as special function cards.

Feature Description

Start with 2-4 user "stories"

Here are some users' stories:

PVP player:

He will first login, if it is the first time he plays, he will first create an account. Then he will in the game hall. Before to start a Majiang game, he may first browse his personal page and the shopping mall. When he is going to start a Majiang game, he can either create a room or join in room others create by input the room number. When the person who creates the room hit "start", a new Majiang game will start. After the game, he can see the result and some statistical data for the game. When he clicks "OK", he will come back to the game hall. He can play like this for many times until he hit "quit" in the game hall, in which the software will end.

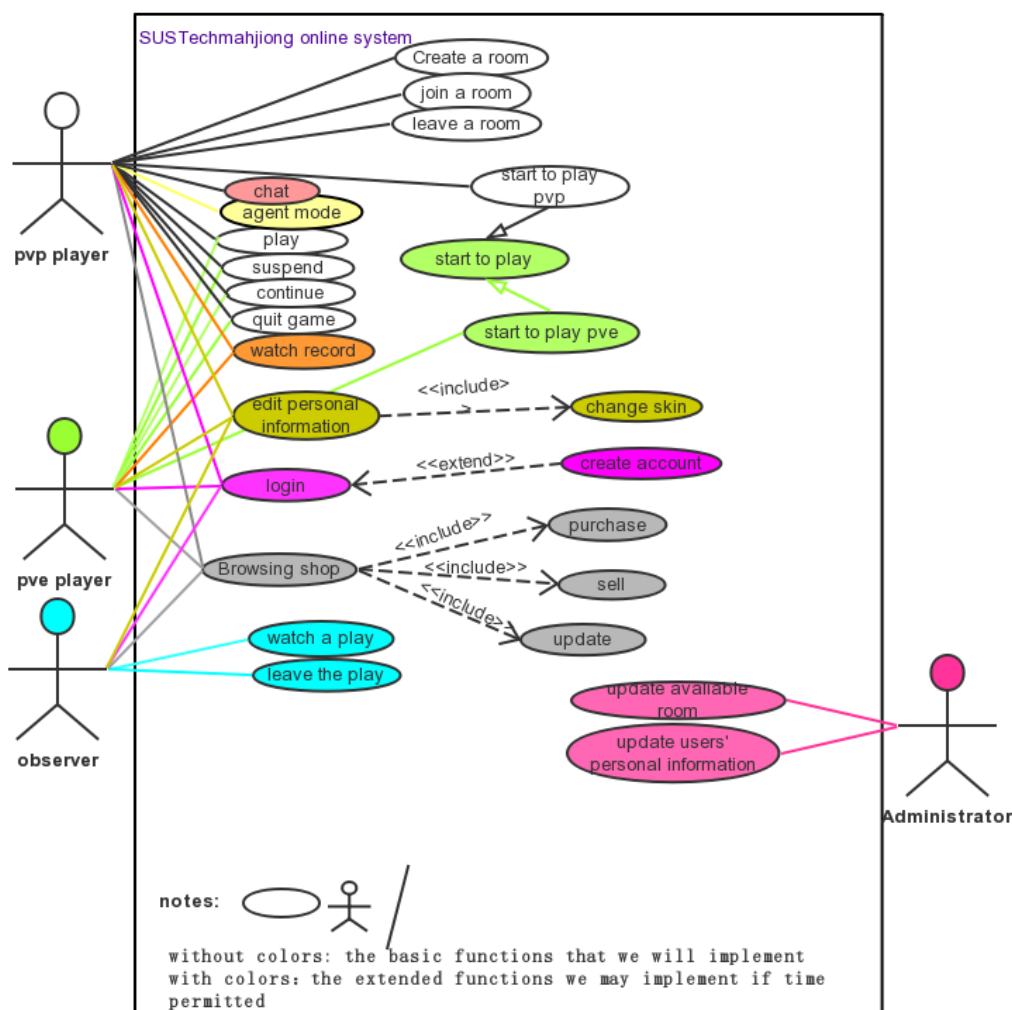
PVE player:

He will first login, if it is the first time he plays, he will first create an account. Then he will in the game hall. Before to start a Majiang game, he may first browse his personal page and the shopping mall. When he is going to start a Majiang game, he can hit “PVE mode” in the game hall, then a new game will start. In the game, he can suspend and continue at any time. After the game, he can see the result and some statistical data for the game. When he clicks “OK”, he will come back to the game hall. He can play like this for many times until he hit “quit” in the game hall, in which the software will end.

observer:

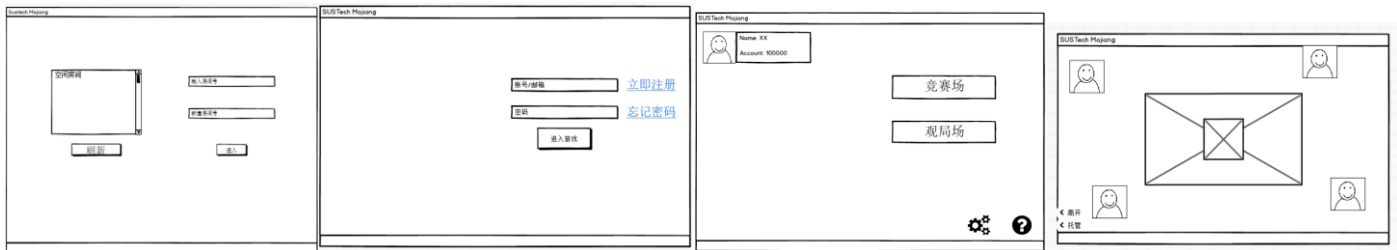
He will first login, if it is the first time he plays, he will first create an account. Then he will in the game hall. Before to start a Majiang game, he may first browse his personal page and the shopping mall. When he is going to observe a game, he can hit “observer mode” in the game hall and input the room number he wants to observe. Then he can observe. He can leave the room at any time by click “quit”. He will then come back to the game hall. He can do like this for many times until he hit “quit” in the game hall, in which the software will end.

Formalize with UML use cases



- **Mockups:** You could use balsamiq (or Google docs, or Adobe Fireworks, or Visio, or something of that sort)

>>>



These are the UI of the game hall, login, personal page, game playing in turns.

Requirements

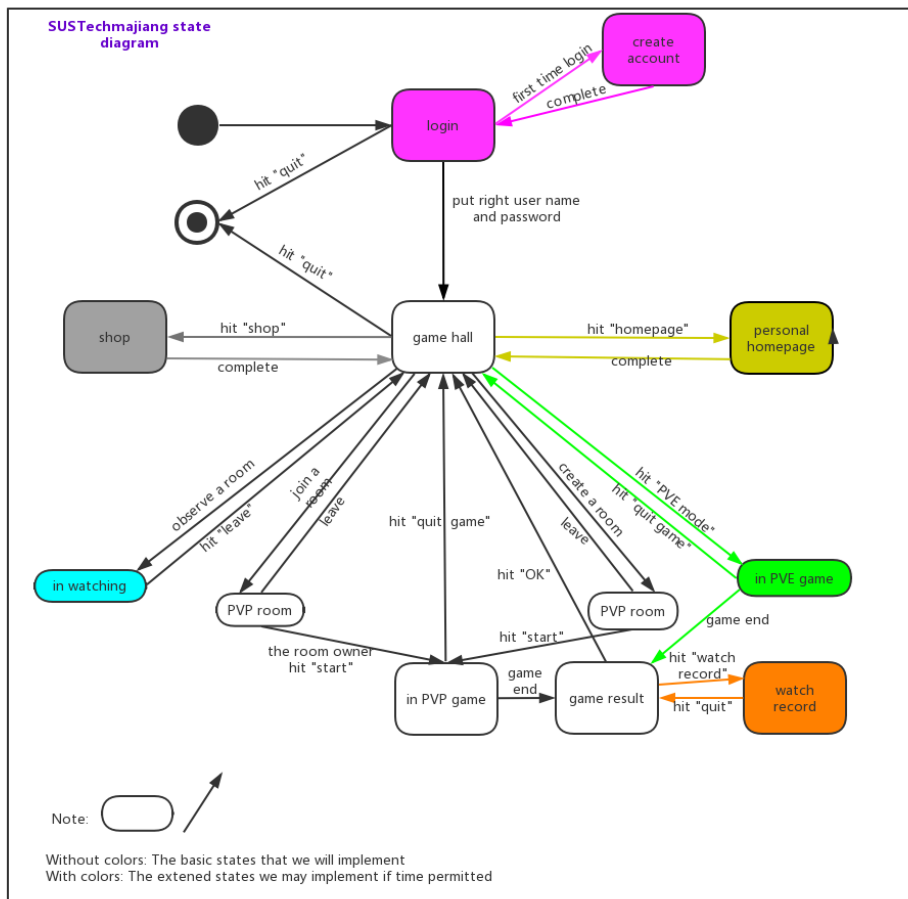
>>>

- The Functional requirements we should reach has been listed in the case diagram.
- We will focus on the Performance • when developing. E.g., response time, accuracy of results, etc.
- We will also focus on Storage requirements
- The Cost per user for deployment is not large.

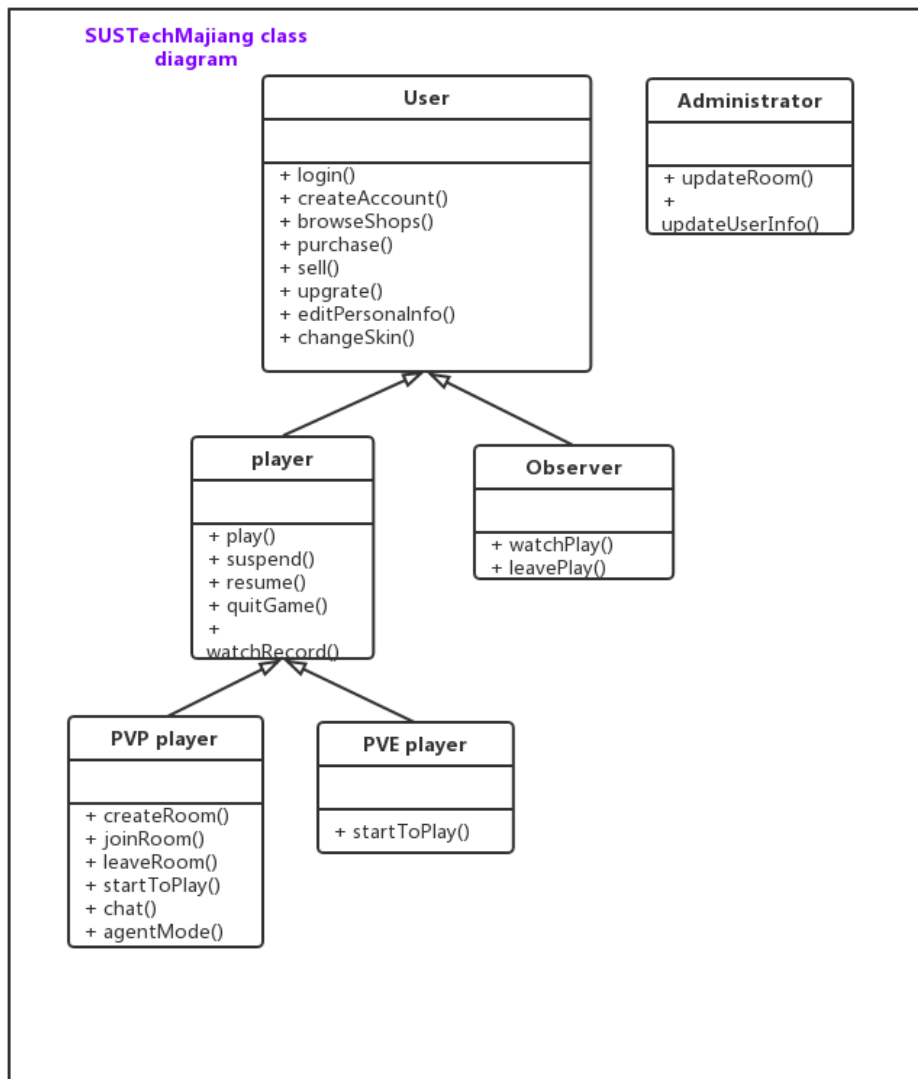
Design Document

Architecture: block diagram, flow charts, class diagram, database schema

State diagram:



Class diagram:



- **Timeline:** *key dates, effort required in number of hours, roles*

preparatory phase:

(before the opening report 2019.10.9)

Develop game rule, search and collect Art materials, learn the basic knowledge of APIs we will use and be familiar with them.

prophase:

(before 11.11)

Develop game logic, combine them with these art materials. Implement the server-side and client-side of game so that the whole game can run.

anaphase:

(before 12.11)

Testing and maintaining

Tasks for which team members are primarily responsible

林傲 (Lin Ao) : *Project Master planning*

Design the rules of the game

卞证 (Bian Zheng) : *Chief programmer*

Implement the code

孙斌贤 (Sun Binxian) : *Project Deputy Planning and Associated programmer*

Solving technical problems

Help Implement the code

徐天元 (Xu Tianyuan) : *Art Designer*

Provide art resource

Make cards with SUSTech element

Design UI in the anaphase

向昕昊 (Xiang Xinhao) : *Communicator (I/O)*

Keep in touch with “customers” (professor Zhang, teacher Zhu and SA)

Collect the latest information about the project and do manage (UML).

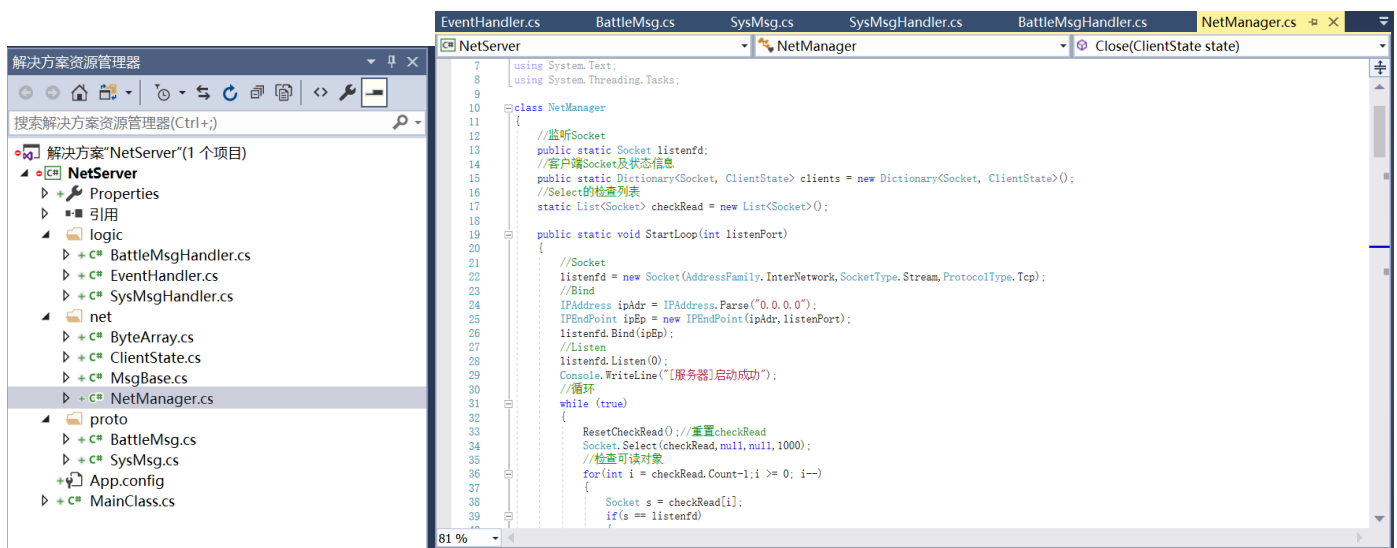
Write project report and do presentation

Help implement the code

Difficulties that we have been overcome by now (2019, 10, 9):

1. We have developed the whole network framework we need in this project so that the we can access the internet in our game now
2. We have succeeded in making the communication between the client-side and server-side
3. We have succeeded in solving “sticky package”, “half package” and other problems about the incomplete data transmission.
4. We have succeeded in sending complete data at the end of communication
5. We have achieved in doing interactive between the server and the corresponding database.

Here are some the screenshot of our achievement:



The screenshot displays the Visual Studio IDE. On the left, the 'Solution Explorer' shows the project structure for 'NetServer'. The 'net' folder contains files like 'ByteArray.cs', 'ClientState.cs', 'MsgBase.cs', and 'NetManager.cs'. The 'proto' folder contains 'BattleMsg.cs', 'SysMsg.cs', and 'App.config'. The 'MainClass.cs' file is also visible. The main editor window shows the 'NetManager.cs' file, which contains the following code:

```
using System.Text;
using System.Threading.Tasks;

class NetManager
{
    //监听Socket
    public static Socket listenfd;
    //客户端Socket及状态信息
    public static Dictionary<Socket, ClientState> clients = new Dictionary<Socket, ClientState>();
    //Select的检查列表
    static List<Socket> checkRead = new List<Socket>();
    public static void StartLoop(int listenPort)
    {
        //Socket
        listenfd = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
        //Bind
        IPAddress ipAddr = IPAddress.Parse("0.0.0.0");
        IPEndPoint ipEp = new IPEndPoint(ipAddr, listenPort);
        listenfd.Bind(ipEp);
        //Listen
        listenfd.Listen(0);
        Console.WriteLine("[服务器]启动成功");
        //循环
        while (true)
        {
            ResetCheckRead(); //重置checkRead
            Socket.Select(checkRead, null, null, 1000);
            //检查可读对象
            for (int i = checkRead.Count - 1; i >= 0; i--)
            {
                Socket s = checkRead[i];
                if (s == listenfd)
```

This is a Server-side code screenshot



This is the real game hall we have done. (just for testing, we will decorate it better for sure in the final.)

Some functions that can extend in the anaphase:

1. develop PVE mahjong game
2. develop the account system (insert some database in it)
3. develop shop system
4. develop section level system
5. develop skin system
6. develop “agent mode” in pvp play
7. develop “watch record” after a play
8. develop “chat” in pvp play
9. develop “observer” mode

• **APIs, services:** *what can you exploit to get done fastest*

Unity, .Net socket, UGUI

Feasibility

Things that may lead us to fail

- Lack of familiarity with APIs
- Unable to deliver on performance
- Cost excessive
- Too many features (prioritize)
- Third party APIs/service may not be reliable

However, we are still confident that we can complete the project, for the reason that:

1. We have clear division of every member's responsibility and scientific management system.
2. In the first state (preparatory phase), we have completed in time and have master the basic knowledge of the tools and APIs we will use.
3. There are no much cost, maybe the only cost is our time!
4. We will first develop the basic version and make sure it can run well, then will we consider developing other extensive function. Too many features will never happen.
5. All the APIs we used are sophisticated and legal software in China, lots of cases have proved that their service is reliable.

References

中国麻将竞赛规则 <http://www.xqbase.com/other/mahjongg.htm>

麻将基本术语中英文对照 http://www.xqbase.com/other/mahjongg_english.htm