Meats Game Platform Technical Report

1. system architecture and design

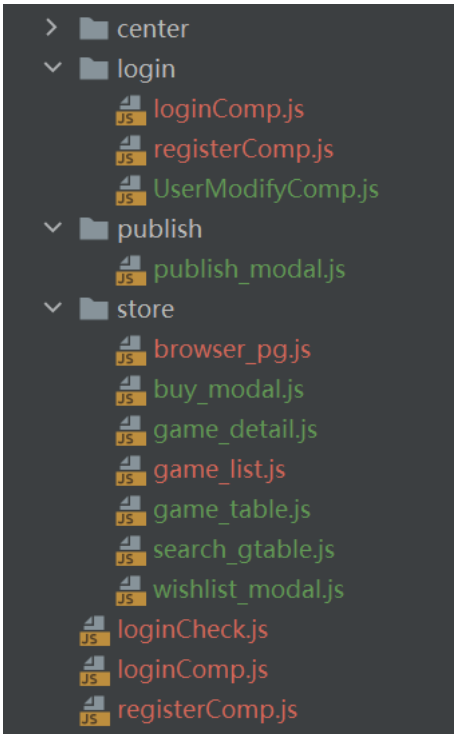
This project's frontend and backend are separated. The frontend framework is react.js and the backend framework is django. Axios is used for frontend-backend communicatioin. We use MySQL as our database and save data locally.

1.1 Frontend Design

For sake of simplicity and beauty of frontend interface, we use react semantic ui framework for UI presentation.

1.1.1 Architecture Design

We use the second mainstream react design, that is, the page function division, where a page function corresponds to a folder, and files used in page function, like the container, component, action and reducer, are placed in this folder. The following is the example of project structure based on the page function division.



1.1.2 File Tree Structure

1. **Layout component**: under folder /container/, there are homepage, community page, store page and login page. These js files only involves components of application UI interface, but not involves data requests and dynamic operation.

2. **Container component:** under folder /component/, it involves get and deal with data from backend.
3. **Display component**: uner folder /component/, it involves UI interface display.

1.1.3 Interaction with Backend

We use a HTTP package based on Promise – axios, which can be used in explorer and node.js. By encapsulating the post, get, etc. methods of axios in the action, we achieve front and back-end interaction.

2. Database Tables

1. Player，inherit from User class

id(generated automatically)	Username	Password	Email

2. Developer，inherit from User class

id(generated automatically)	Username	Password	Email

3. Game

Name	Grade	Price	Type	Developer	Pub_time	Avatar	N_comments

4. Play. Record a player's game progress

Player	Game	Progress	Rate

5.Dreamlist

Player	Game

6. Depository

Player	Game

7. Friend

Player1	Player2

3. Functions Implementation Details

3.1 user actions

（1） register

a) backend gets email、username、password、type (player or developer) from fronted;
b) ensures the username and the email are unused and register a new entry in the specified type.

（2） log in

a) Backend checks if the state is already login (if session user is not None),

if the current state is login, then returen fail;

b) check if the username, password, usertype from frontend are valid, if they are invalid then return fail;
c) save username and usertype into session.

（3） log out

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) executes auth.logout(), delete information in session.

（4） follow

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Player, if so get the current user from Player table according to username in session;
c) get target username from frontend and check Player table if the target user exists, check targetFriends table if the current user has followed the target user;
d) create a new entry, where player1 = user, player2 = target.

（5） unfollow

a) Backend checks if the current state is login (session user is not None), if it is not login then return fail;
b) check if usertype in session is Player, if so get the current user from Player table according to username in session;
c) get target username from frontend;
d) delete this entry from Friends table.

3.2 player-game operation

（1） add to wishlist

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Player, if so get the current user from Player table according to username in session;
c) get game name from frontend and check the Game table if the gamename exists;
d) check Dreamlist table if the combination of user, game exists, if so the entry can’t be added again.

（2） buy games

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Player, if so get the current user from Player table according to username in session;
c) get game name from frontend and check the Game table if the gamename exists;
d) check Depository table if the game has been bought before, if so the entry can’t

be added again;

e) add the entry in Depository table;
f) add the entry in Play tbale, the progress is 0.

（3） delete games from wishlist

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Player, if so get the current user from Player table according to username in session;
c) get game name from frontend and check the Game table if the gamename exists;
d) check Dreamlist table if user, game entry exists, then delete it.

（4） rate games

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Player, if so get the current user from Player table according to username in session;
c) get game name and user rate from frontend, filter the Depository table by username and check if the game name exists;
d) update rate in Playlist table to user_grade, and update grade in Game table to (grade+user_grade) / (n_commnets+1), n_comments+=1

（5） update progress

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Player, if so get the current user from Player table according to username in session;
c) get game name and progress update from frontend, filter the Play table by username and check if the game name exists;
d) update grade in the Play table entry

3.3 developer operation

（1） release games

a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Developer;
c) get game name, type, avatar and price from frontend;
d) check Game table if the current game name exists and if not, add the entry to Game table

（2） delete games

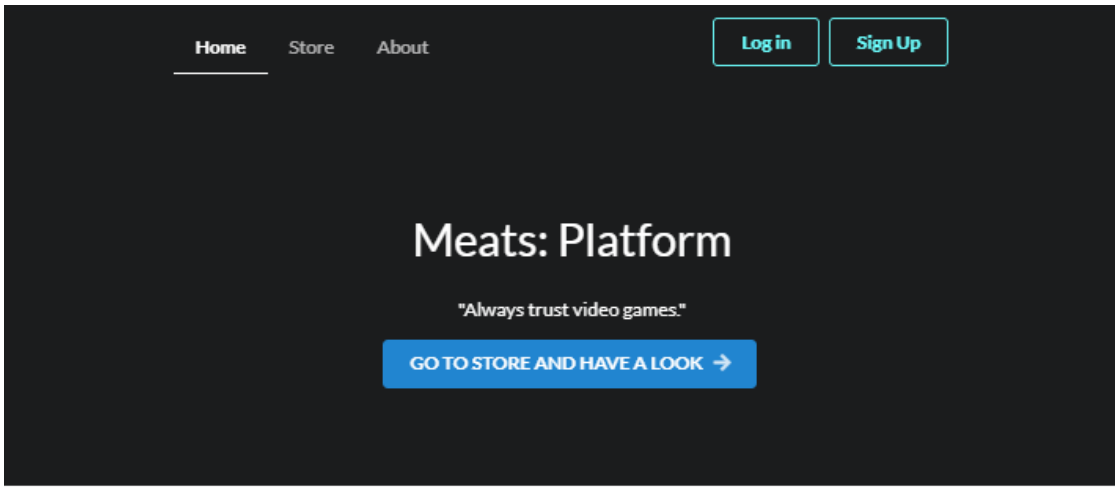
a) Backend checks if the current state is login (if session user is not None), if it is not login then return fail;
b) check if usertype in session is Developer;
c) get game name from frontend;

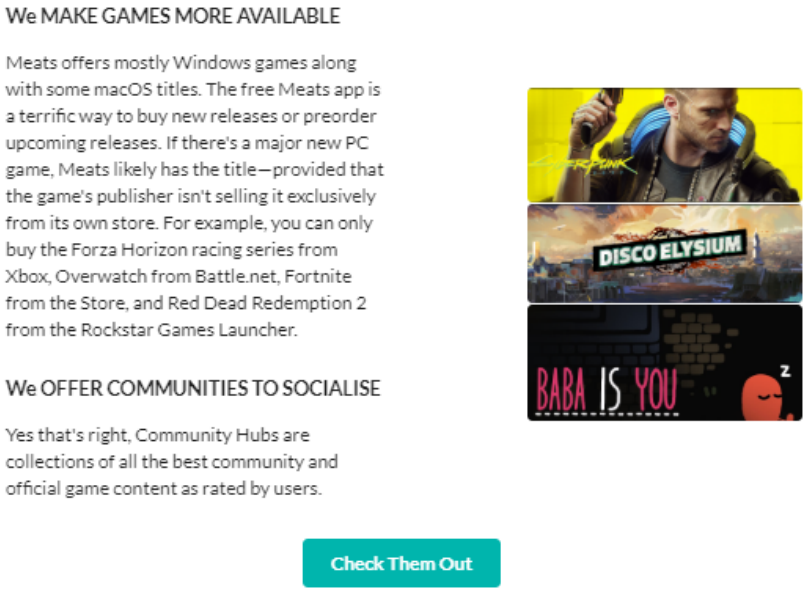
d) check Game table if the current game name exists and delete the entry from Game table

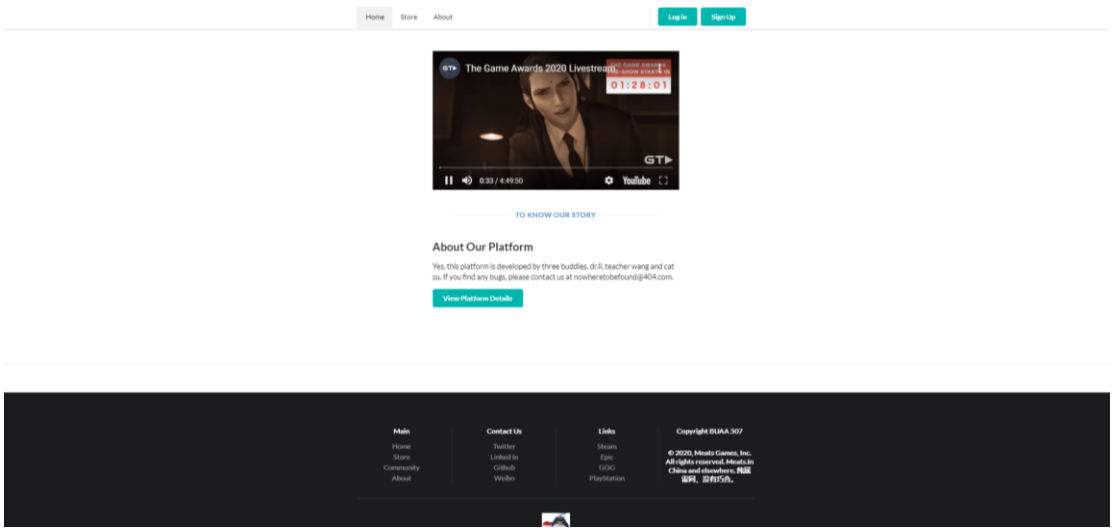
4. Result

1) UI design

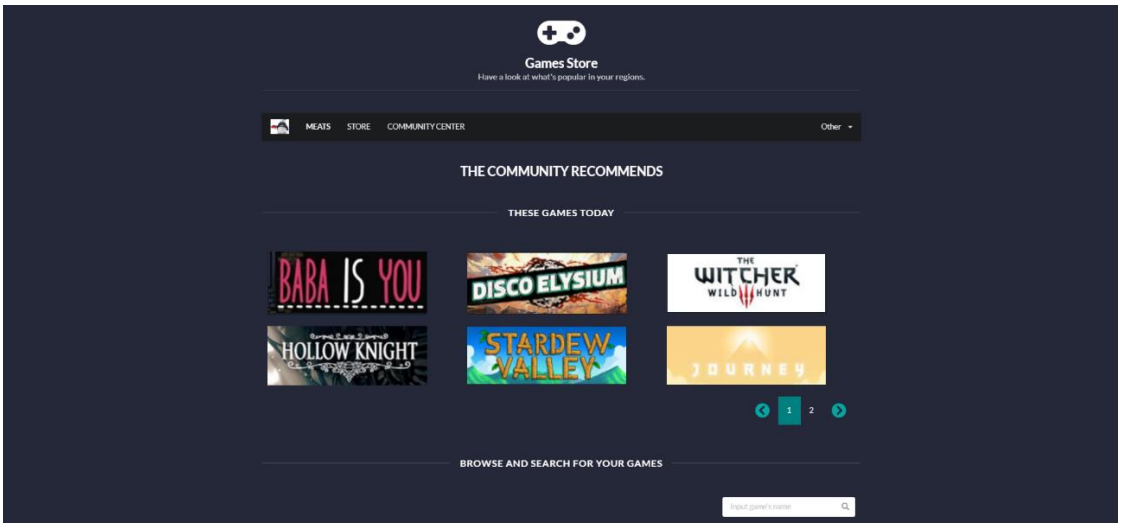
① homepage

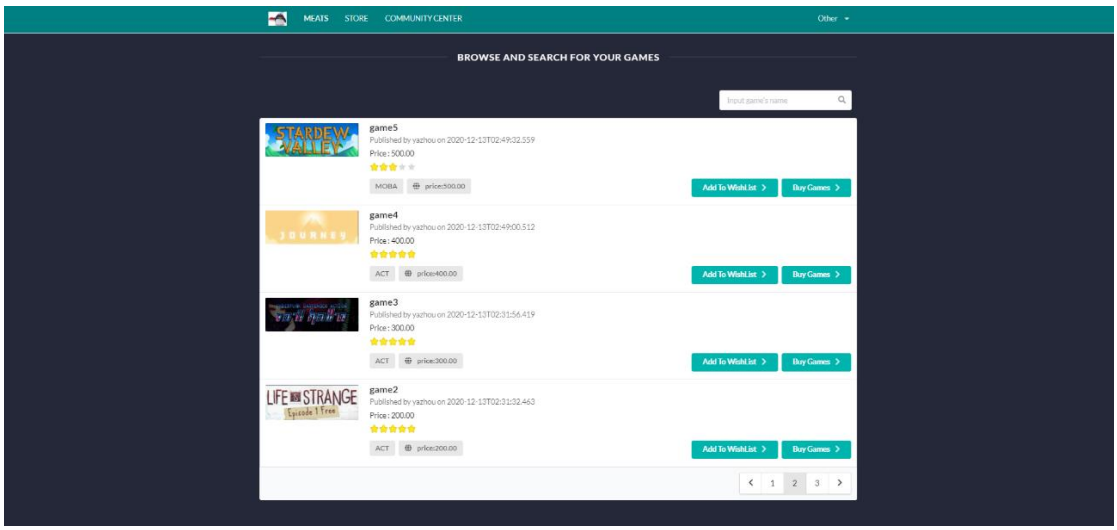




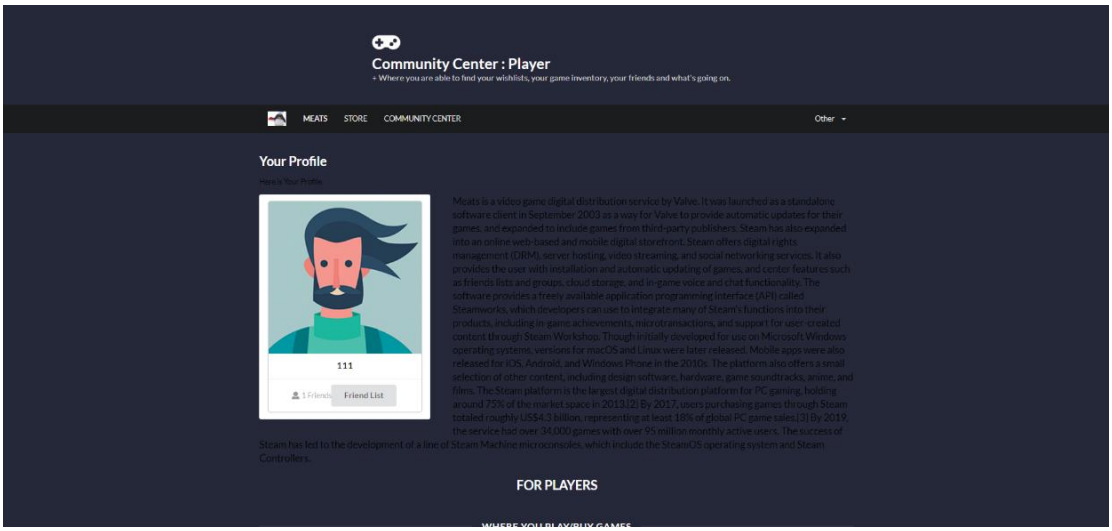


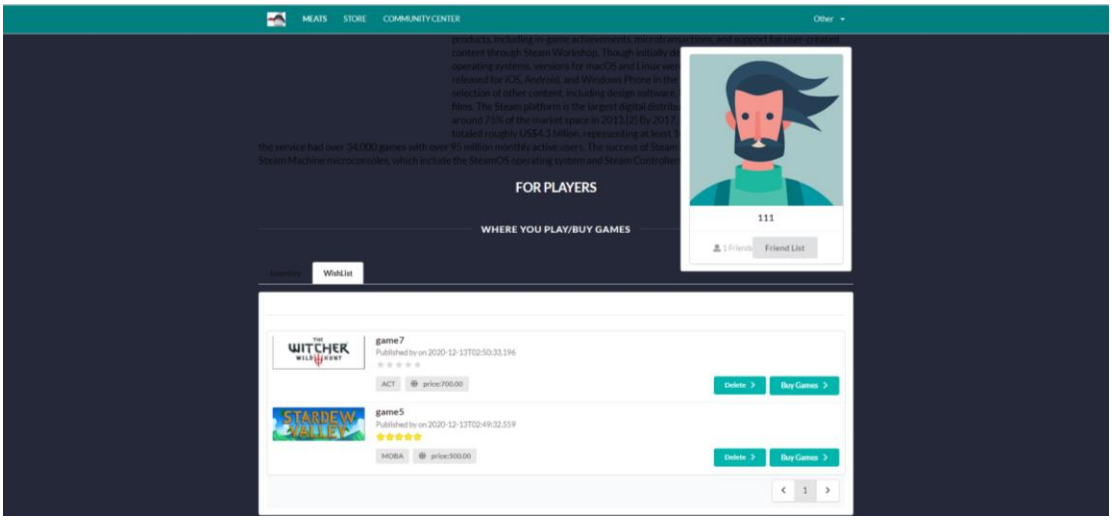
② store page

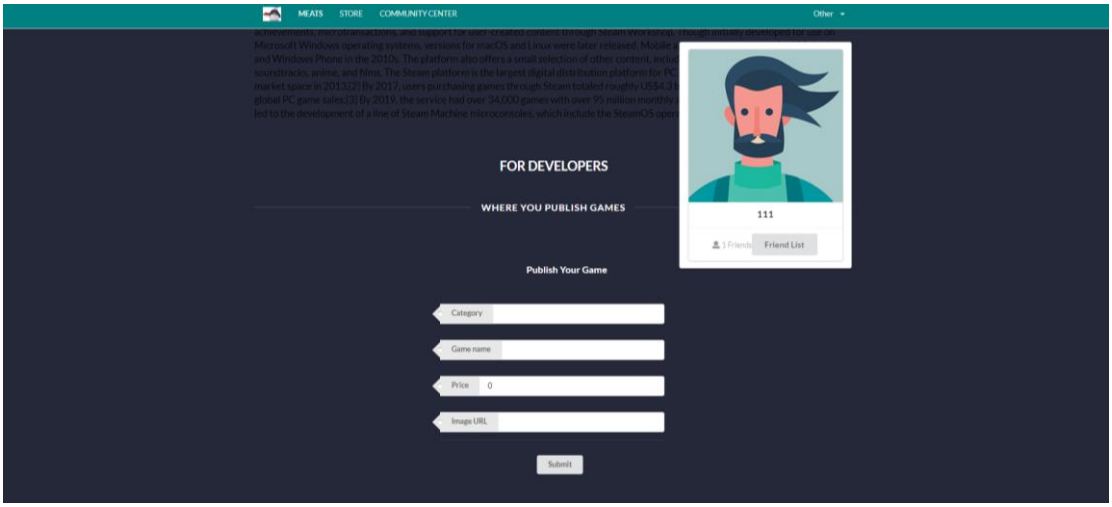


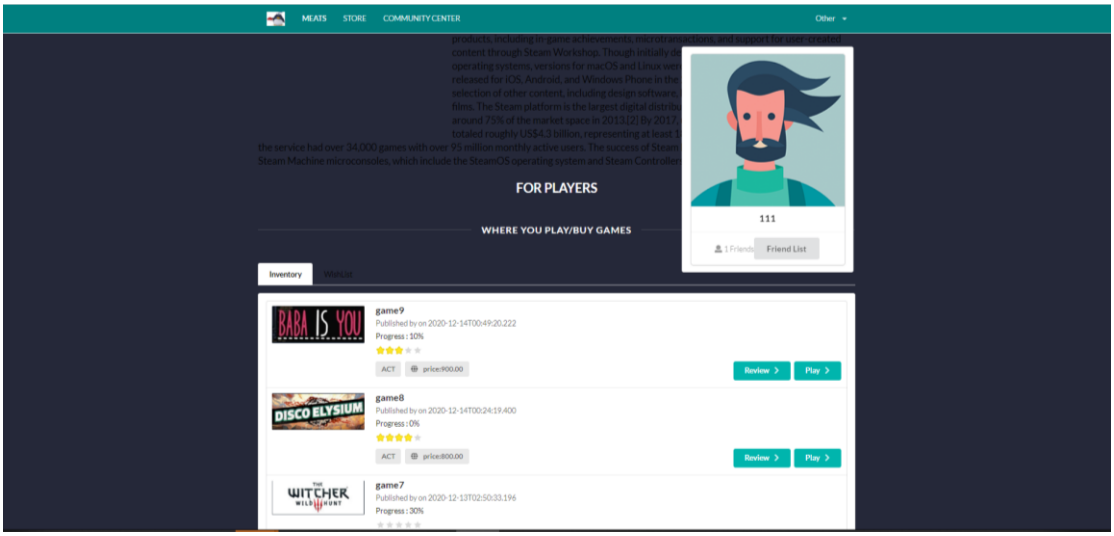


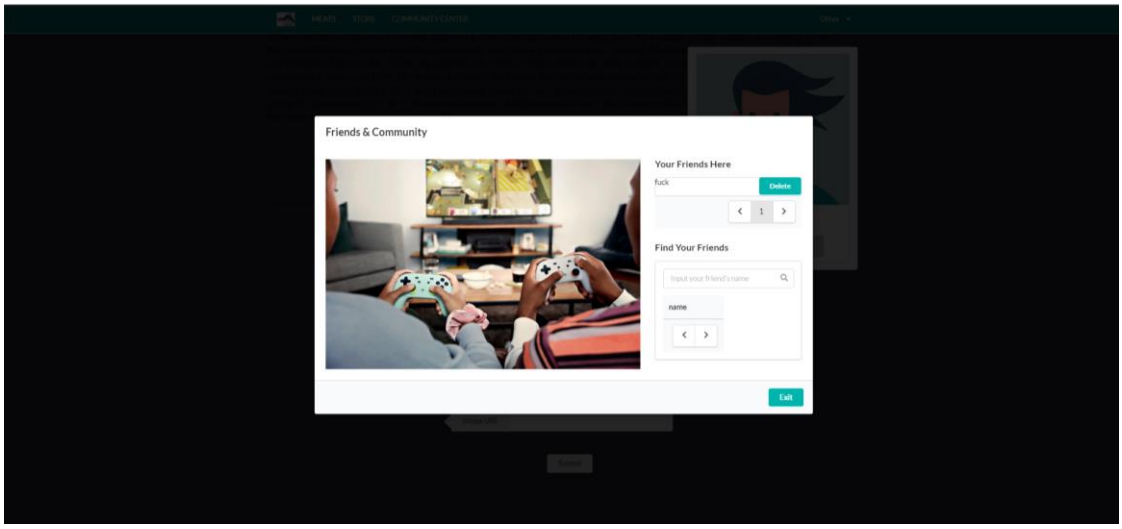
③ user center page


Community Center : Player
+ Where you are able to find your wishlists, your game inventory, your friends and what's going on.
MEATS
STORE
COMMUNITY CENTER
Other
Your Profile
Here's Your Profile
111
1 Friends
Friend List
Meats is a video game digital distribution service by Valve. It was launched as a standalone software client in September 2003 as a way for Valve to provide automatic updates for their games, and expanded to include games from third-party publishers. Steam has also expanded into an online web-based and mobile digital storefront. Steam offers digital rights management (DRM), server hosting, video streaming, and social networking services. It also provides the user with installation and automatic updating of games, and center features such as friends lists and groups, cloud storage, and in-game voice and chat functionality. The software provides a freely available application programming interface (API) called Steamworks, which developers can use to integrate many of Steam's functions into their products, including in-game achievements, microtransactions, and support for user-created content through Steam Workshop. Though initially developed for use on Microsoft Windows operating systems, versions for macOS and Linux were later released. Mobile apps were also released for iOS, Android, and Windows Phone in the 2010s. The platform also offers a small selection of other content, including design software, hardware, game soundtracks, anime, and films. The Steam platform is the largest digital distribution platform for PC gaming, holding around 75% of the market space in 2013.[2] By 2017, users purchasing games through Steam totaled roughly US$4.3 billion, representing at least 18% of global PC game sales.[3] By 2019, the service had over 34,000 games with over 95 million monthly active users. The success of Steam has led to the development of a line of Steam Machine microconsoles, which include the SteamOS operating system and Steam Controllers.
FOR PLAYERS
WHERE YOU PLAY/BUY GAMES


MEATS
STORE
COMMUNITY CENTER
Other
FOR PLAYERS
WHERE YOU PLAY/BUY GAMES
111
1 Friends
Friend List
Inventory
WishList
game7
Published by on 2020-12-13T02:50:33.196
ACT
price:700.00
Delete
Buy Games
game5
Published by on 2020-12-13T02:49:32.559
MOBA
price:500.00
Delete
Buy Games
1

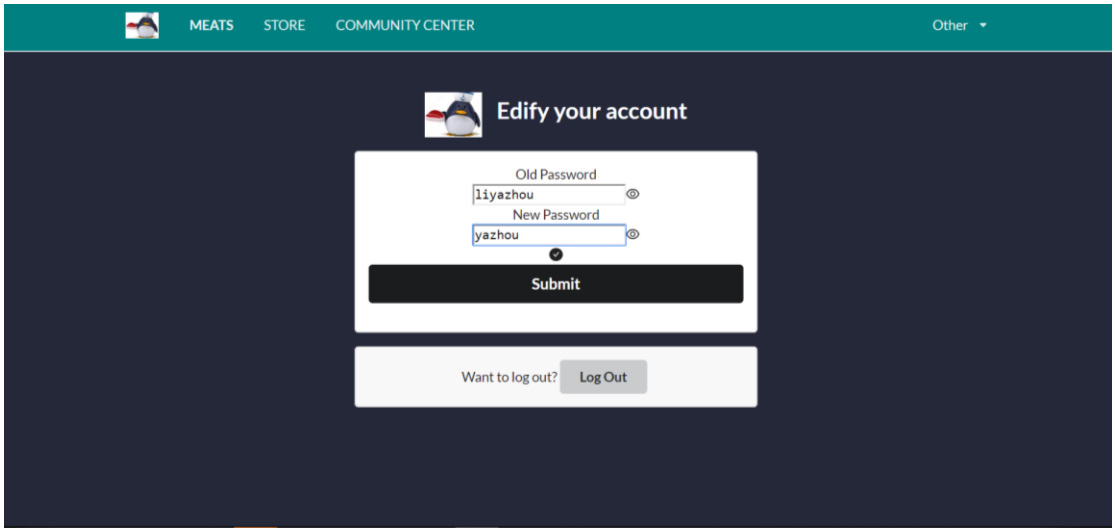

MEATS
STORE
COMMUNITY CENTER
Other
FOR DEVELOPERS
WHERE YOU PUBLISH GAMES
111
1 Friends
Friend List
Publish Your Game
Category
Game name
Price
0
Image URL
Submit

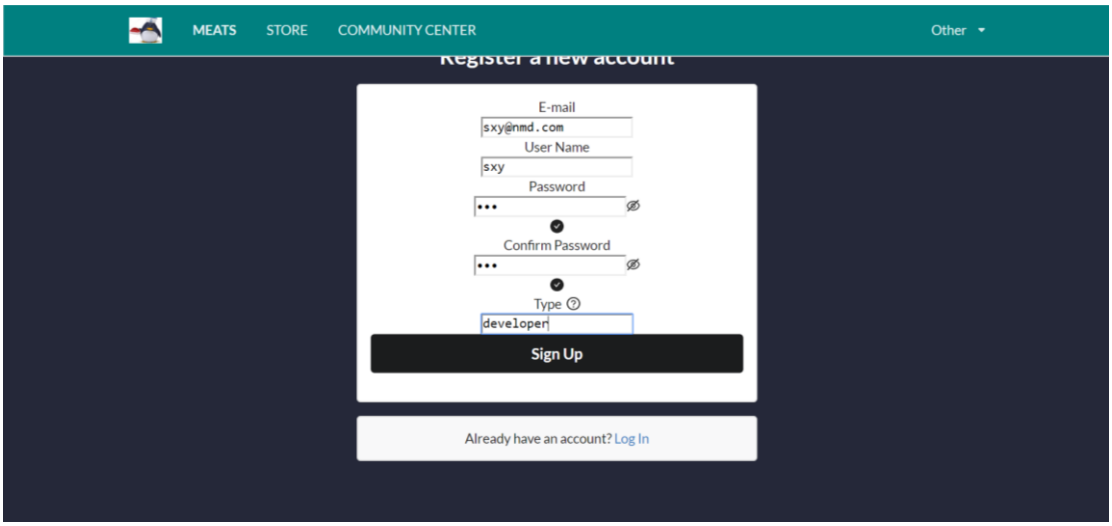


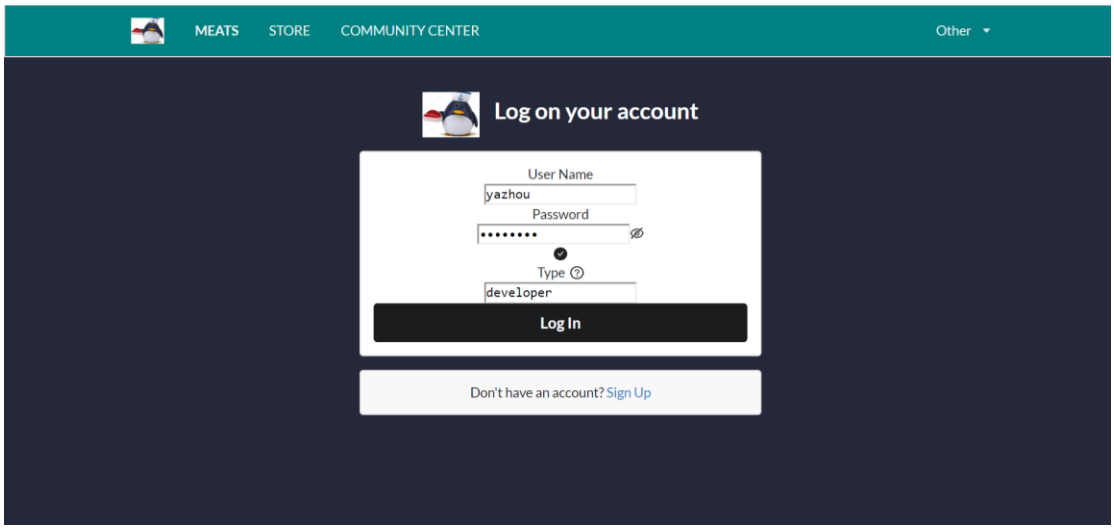


2) functions implementation

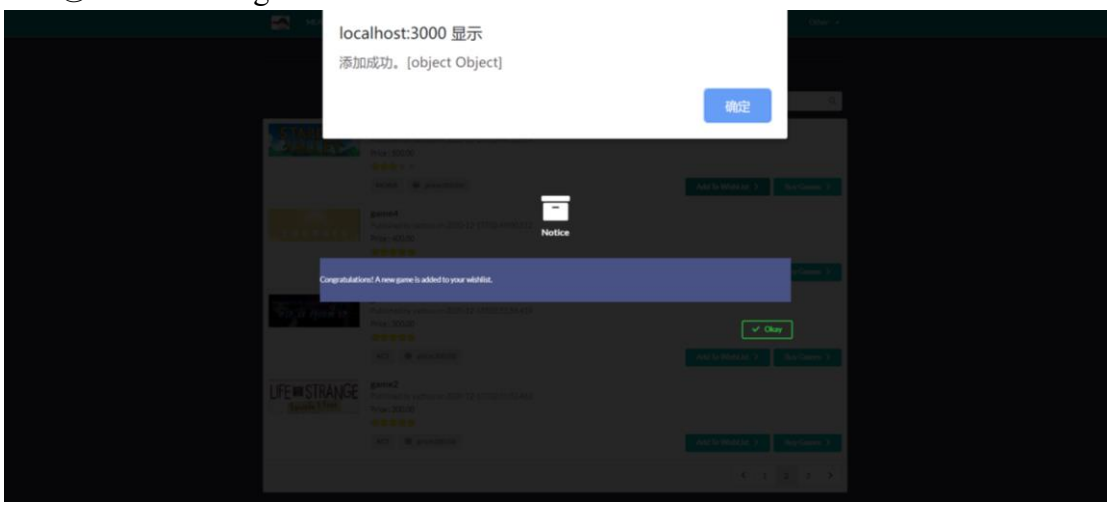
① log in/register/modify user information

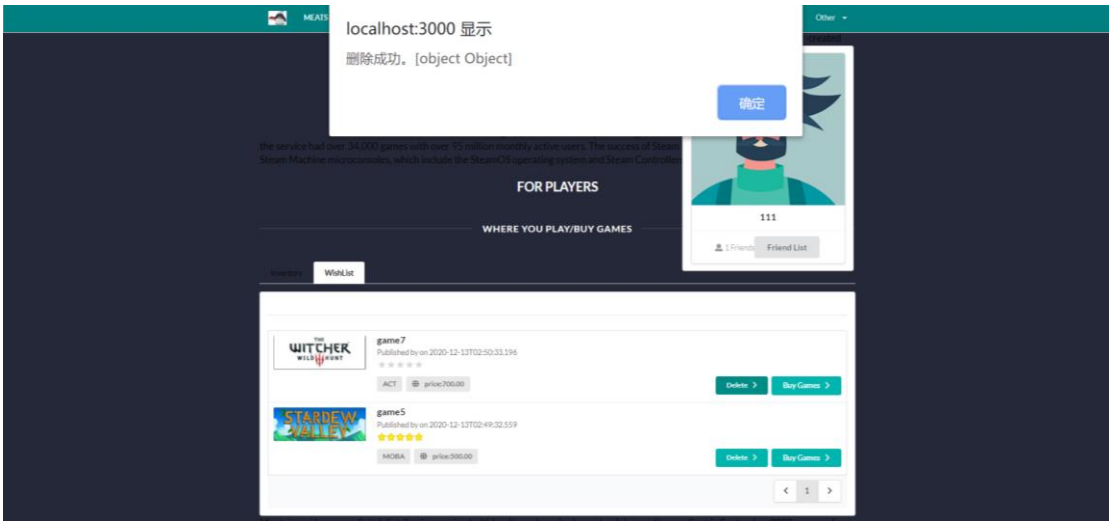


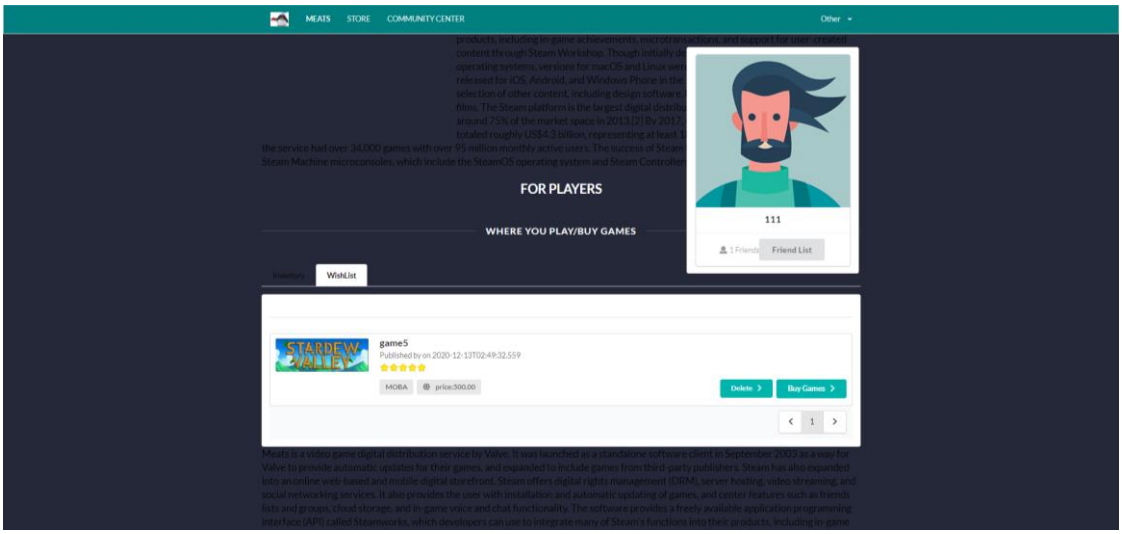




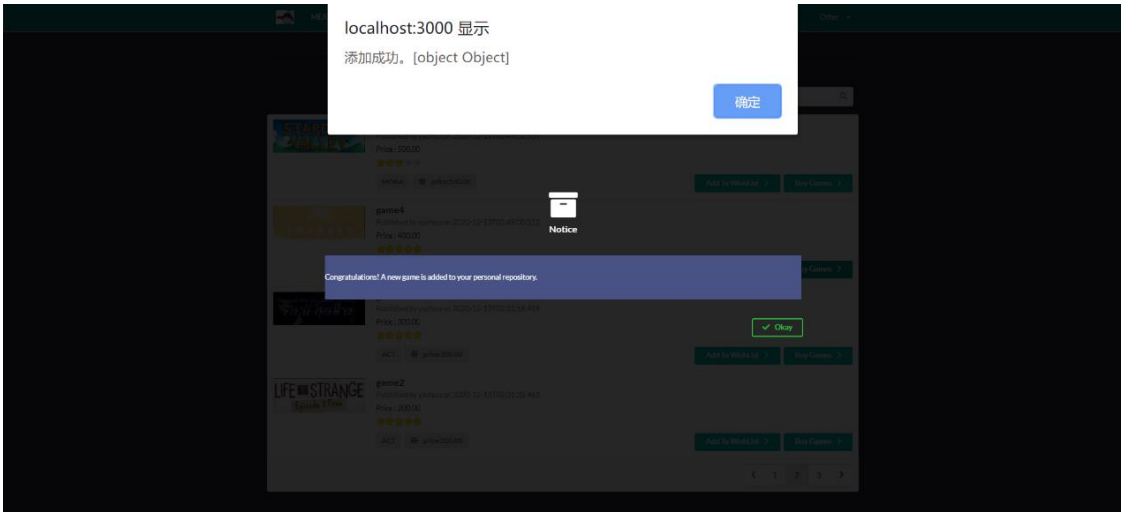
② add/delete games from wishlist

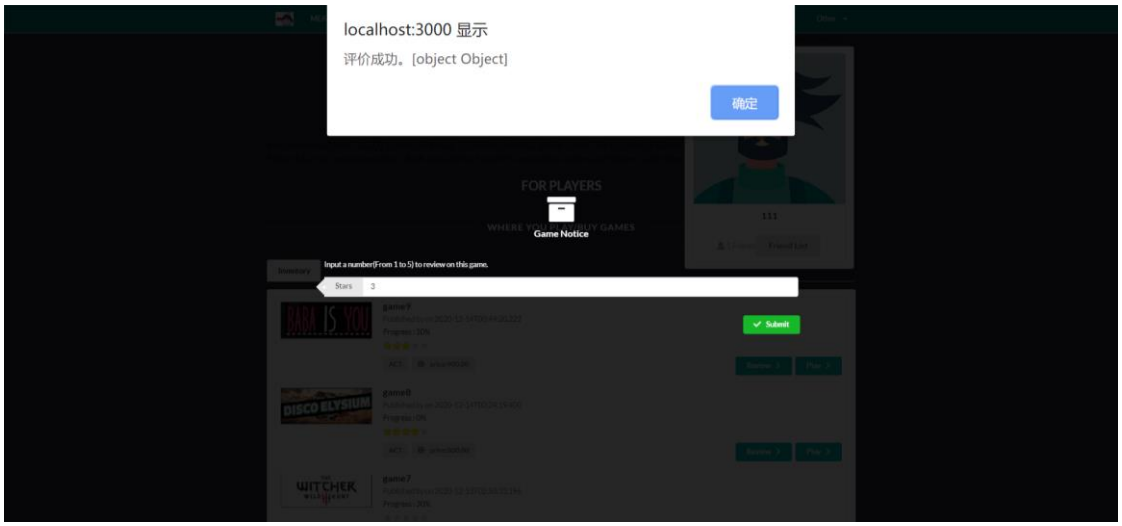


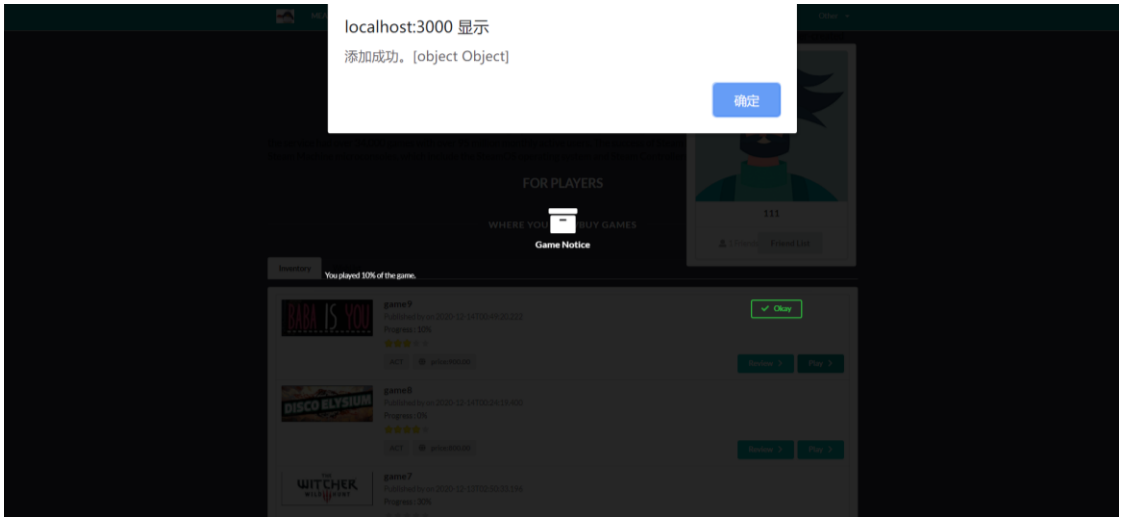




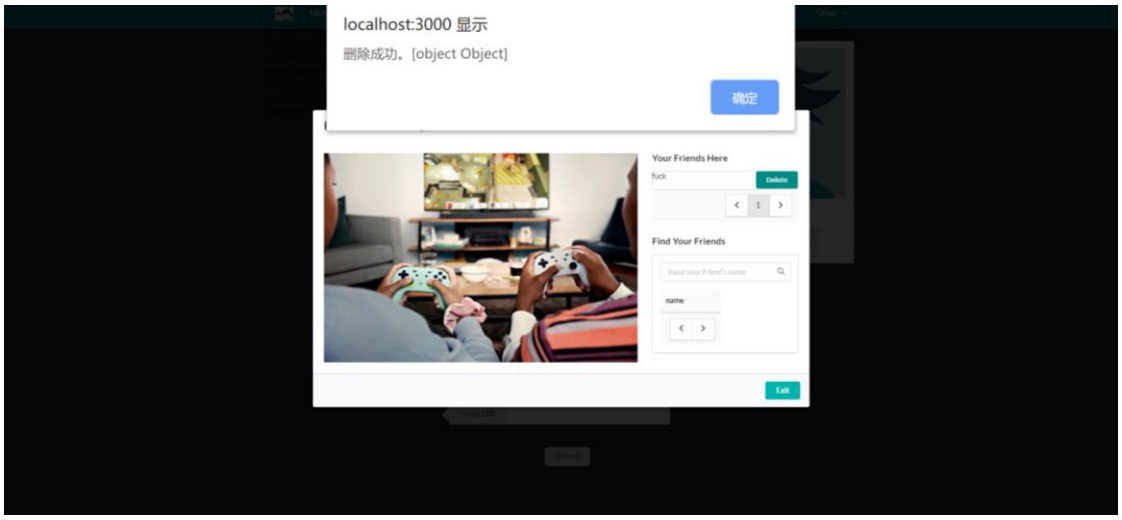
③ search/buy/review/play games

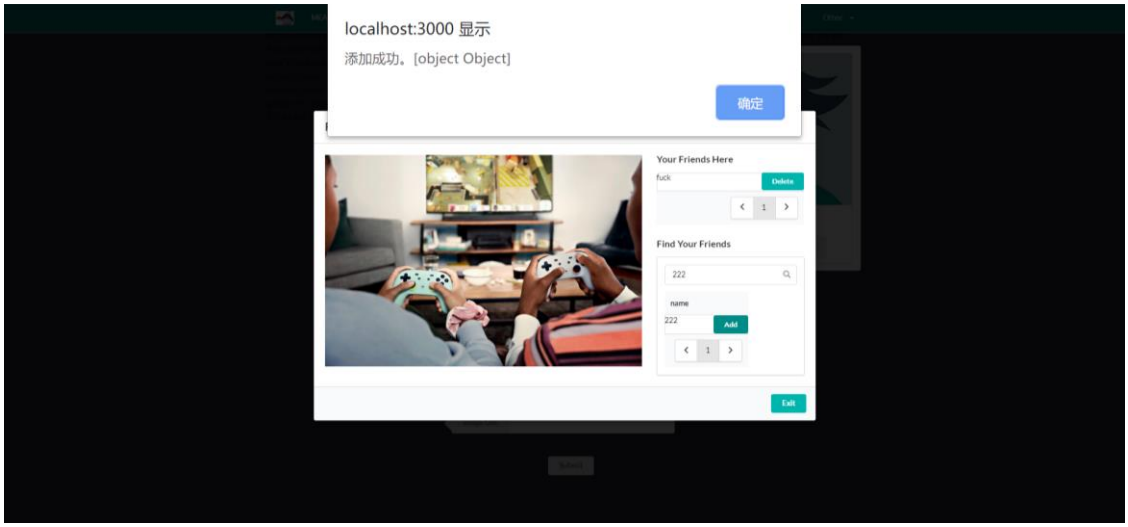




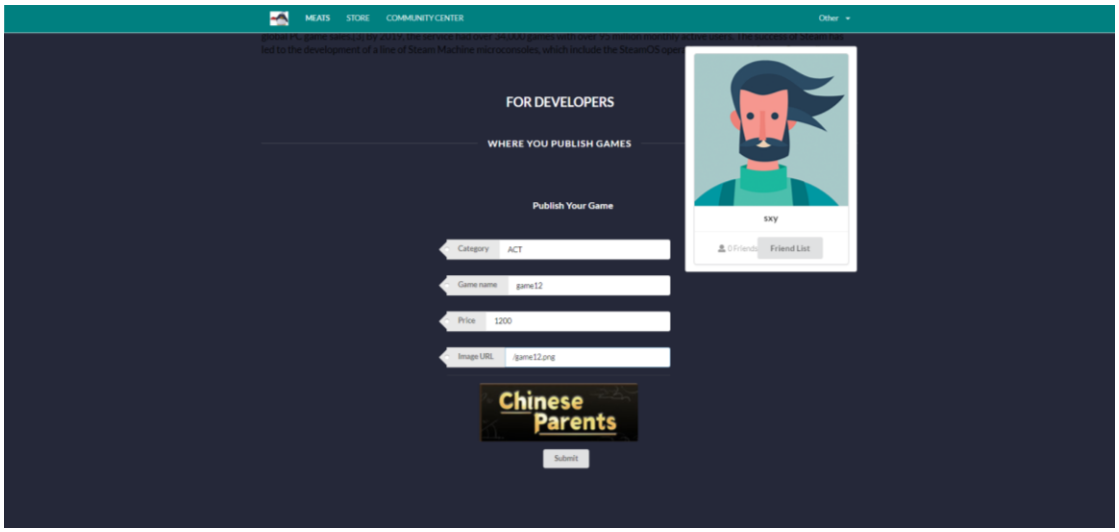


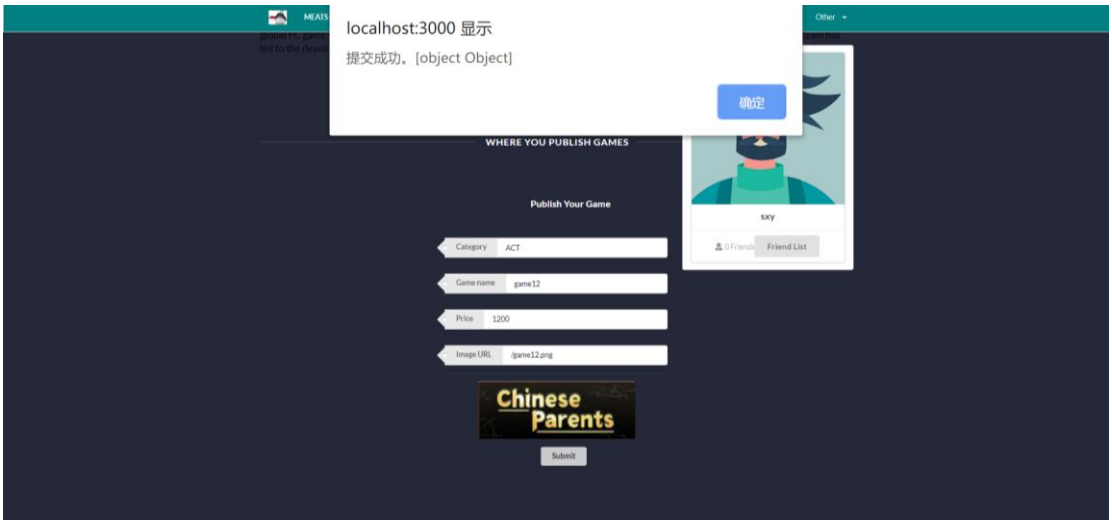
④ add/delete/search friends



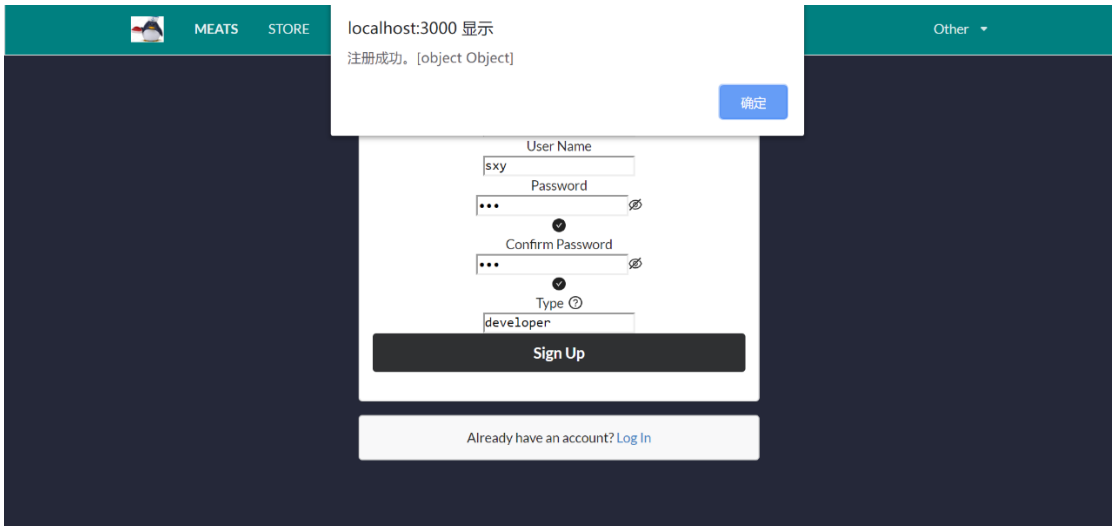


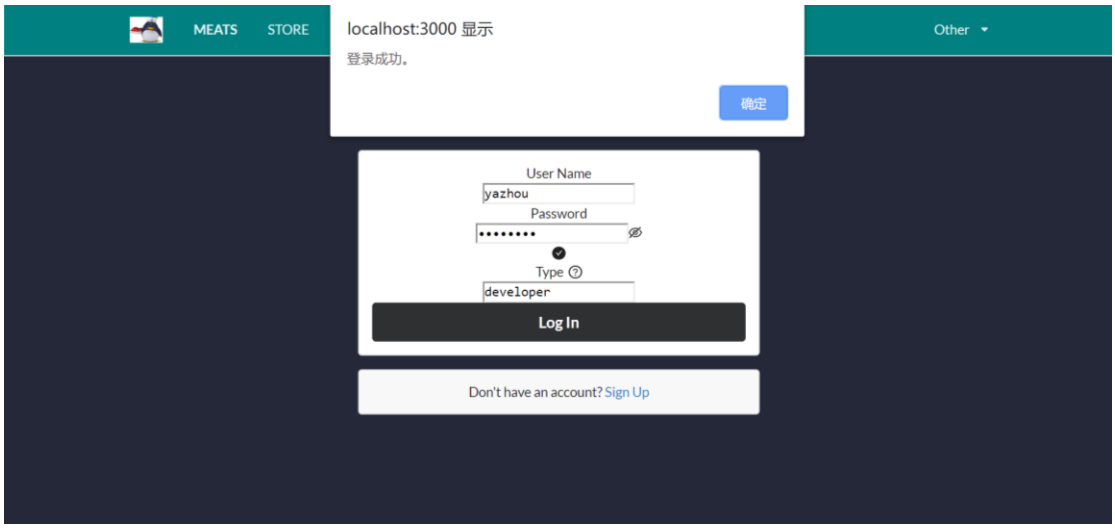
⑤ release games

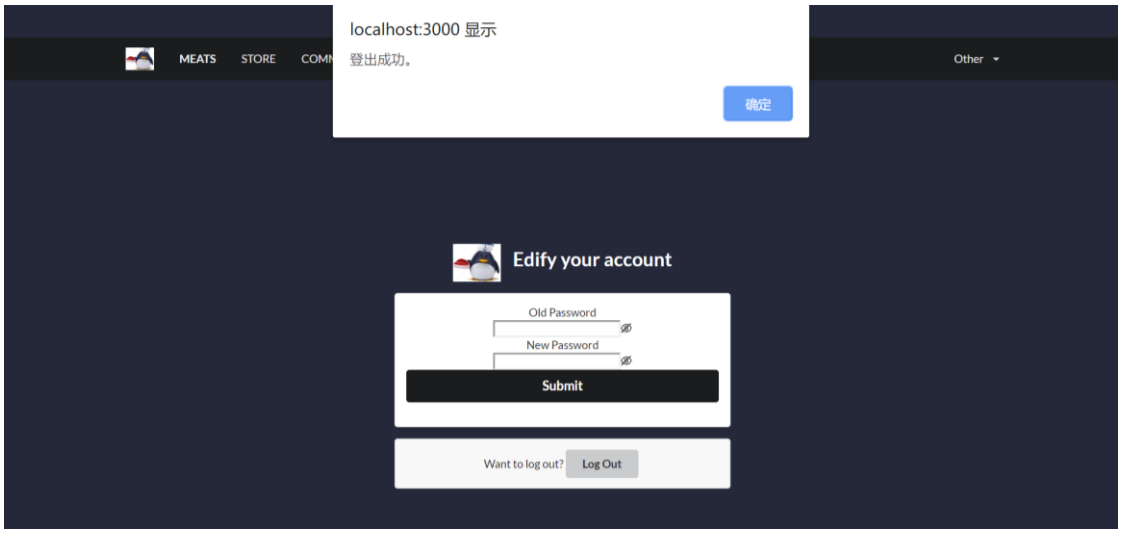


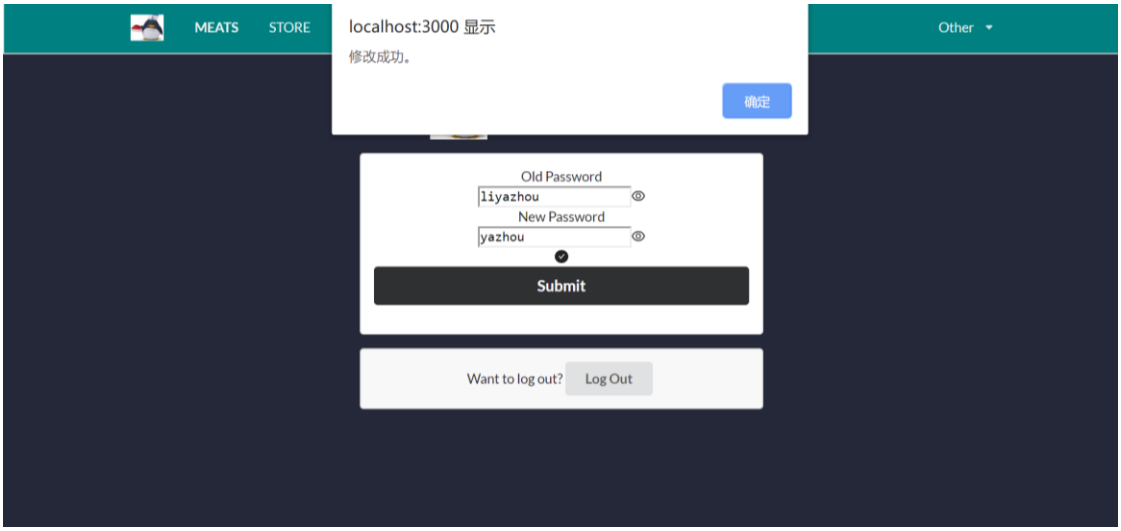


⑥ register/log in/log out/ revise information


MEATS
STORE
Other
localhost:3000 显示
注册成功。[object Object]
确定
User Name
sxy
Password
Confirm Password
Type
developer
Sign Up
Already have an account? Log In


MEATS
STORE
Other
localhost:3000 显示
登录成功。
确定
User Name
yazhou
Password
Type
developer
Log In
Don't have an account? Sign Up


MEATS
STORE
Other
localhost:3000 显示
登出成功。
确定
Edify your account
Old Password
New Password
Submit
Want to log out?
Log Out


MEATS
STORE
Other
localhost:3000 显示
修改成功。
确定
Old Password
liyazhou
New Password
yazhou
Submit
Want to log out?
Log Out