The Development Document of “掌上通通”(My Tongtong)

Group 4

Members: Zihan Zhou, Liang Deng, Junze Li, Zihan Xu,

Zicheng Xu, Rui Xing

2020.12

CONTENT

1. Demand Analysis

1.1 Background

In order to better understand the needs of college students, we took a look at the headaches of college life. Here are several popular answers, and these are also the problems around us: Where is the study room? What class am I having today? Where to say something in my heart without identification? and so on. These really become the pain point of college life. Therefore, a campus information platform that integrates multiple functions is in urgent need for students.

1.2 Goal

Our motivation is to use mobile development technology to develop a campus information platform to make our university life more convenient.

1.3 User characteristics

The main users of this application are college students. This user group is more proficient in the use of mobile applications and has high requirements for the appearance and interaction of the application.

1.4 Requirement

1.4.1 Function Requirement

The application mainly contains the following functions:

No.	Function	Description
1	Spare Classroom Query	Query the spare classroom of the building
2	Class Schedule	Get the classes you have of the week
3	Tree hole	Say something you want without identification
4	Step Count	Count your step and compare with your friends

Extra	User Management	Management your background, avatar and account info

1.4.2 Performance requirement

The data is required to be completely accurate, the data loading time under normal network conditions does not exceed 3 seconds, and the page switching and component click response time does not exceed 500 milliseconds.

1.4.3 Reliability requirement

Software network data transmission requires high reliability, and at the same time requires data accuracy and real-time. In addition, the software has the ability to handle errors and exceptions, and there is almost no software failure, ensuring the normal operation of the software and data storage.

1.4.4 Operating environment requirement

Android mobile phone, with Internet access, memory above 1GB, Android 9.0 and above recommended.

1.4.5 UI Requirement

The software is mainly based on a graphical interface, the theme is simple and clear, the layout is reasonable, and the user interaction effect is good. The user can complete all functions by clicking and keyboard input.

2. General Design

2.1 Introduction

2.1.1 Purpose

This general design is based on the requirement analysis of My Tongtong, and the outline design of My Tongtong is carried out. The general design of the software plays a guiding role in the development process of the software project, ensuring the project team to complete the project objectives on time and with good quality, facilitating the project team members to better understand the project situation, and making the actual development process of the project reasonable and orderly. Therefore, the overall design, interface design, operation design, data structure design and system error handling design of the software are recorded in the form of documentation, which serves as the consensus and agreement between project team members during the development process, as well as the basis for the project team to carry out and check the project work.

This general design specification is for all project developers and users, to provide the basis for the subsequent development of the system.

2.1.2 Background

Task proposer：Zihan Zhou, Liang Deng, Junze Li, Zihan Xu,
Zicheng Xu, Rui Xing
Project developer：Zihan Zhou, Liang Deng, Junze Li, Zihan Xu,
Zicheng Xu, Rui Xing
User：Teachers and students of Beijing Jiaotong University
The computing center of the software：The final form of this software is an APP running on the mobile Android platform, and the server of the software is supported by Aliyun.

2.1.3 Definition

Android: is a free and open source operating system based on Linux, mainly used in mobile devices.

C/S Architecture: A software architecture, that is, the client server structure.

JDBC: It is a Java API used to execute SQL statements. It can provide unified access to a variety of relational databases. It consists of a set of classes and interfaces written in Java language.

SQL: is a special purpose programming language, a database query and programming language, used to access data and query, update and manage relational database systems.

2.1.4 Reference

《Software Engineering A Practitioners Approach》Roger S.Pressman,Bruce R.Maxim ,2015
《Database System Concepts》 Abraham Silberschatz, Hennry F. Korth, S.

Sudarshan, 2012

2.2 Overall Design

2.2.1 Requirements

Name	Input	Output	Performance Requirements
Login	Account (email) and password	If the match is correct, enter the homepage, if the match fails, there will be a prompt animation	When the network is in good condition, the response time should not exceed 2 seconds
Registration	Email, user name (combination of 3~10 alphanumeric and underscore), password (6~18 digits and beginning with a letter), confirmation password and email verification code	Prompt whether the registration is successful	When the network is in good condition, the response time should not exceed 2 seconds

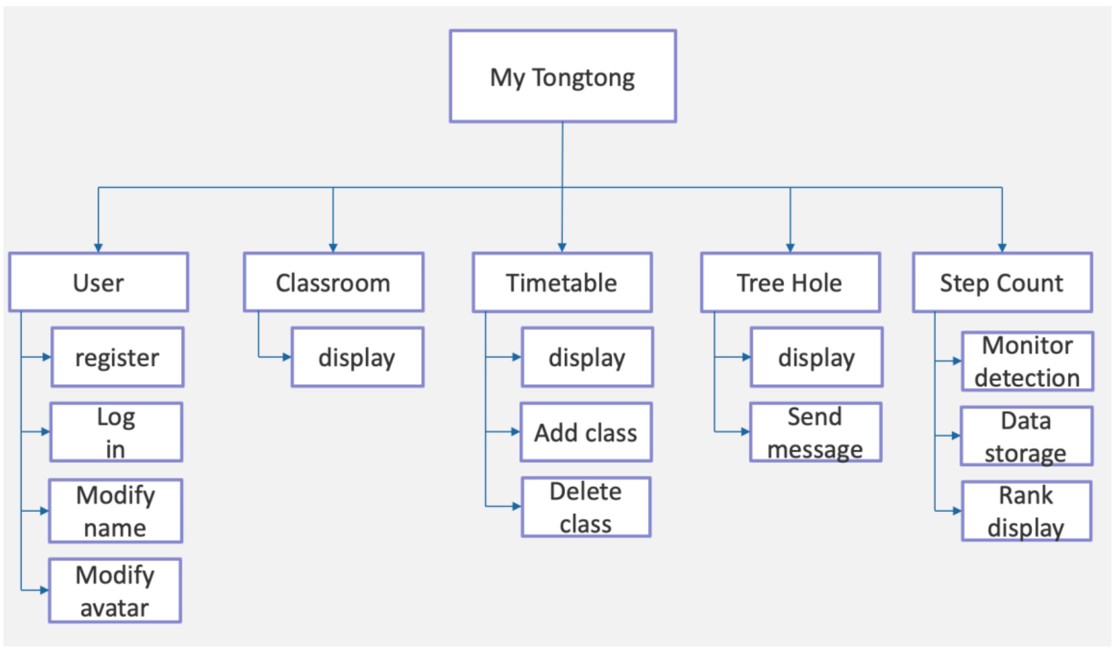
Forget Password	Email, password, confirm password and email verification code	Pop up whether the password is changed successfully	When the network is in good condition, the response time should not exceed 2 seconds
Course Schedule	Click to add a course or delete a course	Show added courses or remove deleted courses	When the network is in good condition, the response time should not exceed 2 seconds
Vacant Classroom	Time slots and academic buildings	Corresponding list of available classrooms	When the network is in good condition, the response time should not exceed 2 seconds
Hole	Tree hole information to be sent	Tree hole information is displayed in the tree hole list	When the network is in good condition, the response time should not exceed 2 seconds
Sport	Number of steps taken by the user	Display the step count of different users in the form of a leaderboard	When the network is in good condition, the response time should not exceed 2 seconds
Profile Setting	User avatar and background image	Change the user's avatar and background image	When the network is in good condition, the response time should not exceed 2 seconds
Newcomer Orientation	New user login information	Show orientation animation to new users	When the network is in good condition, the response time should not exceed 2 seconds

2.2.2 Operating Environment

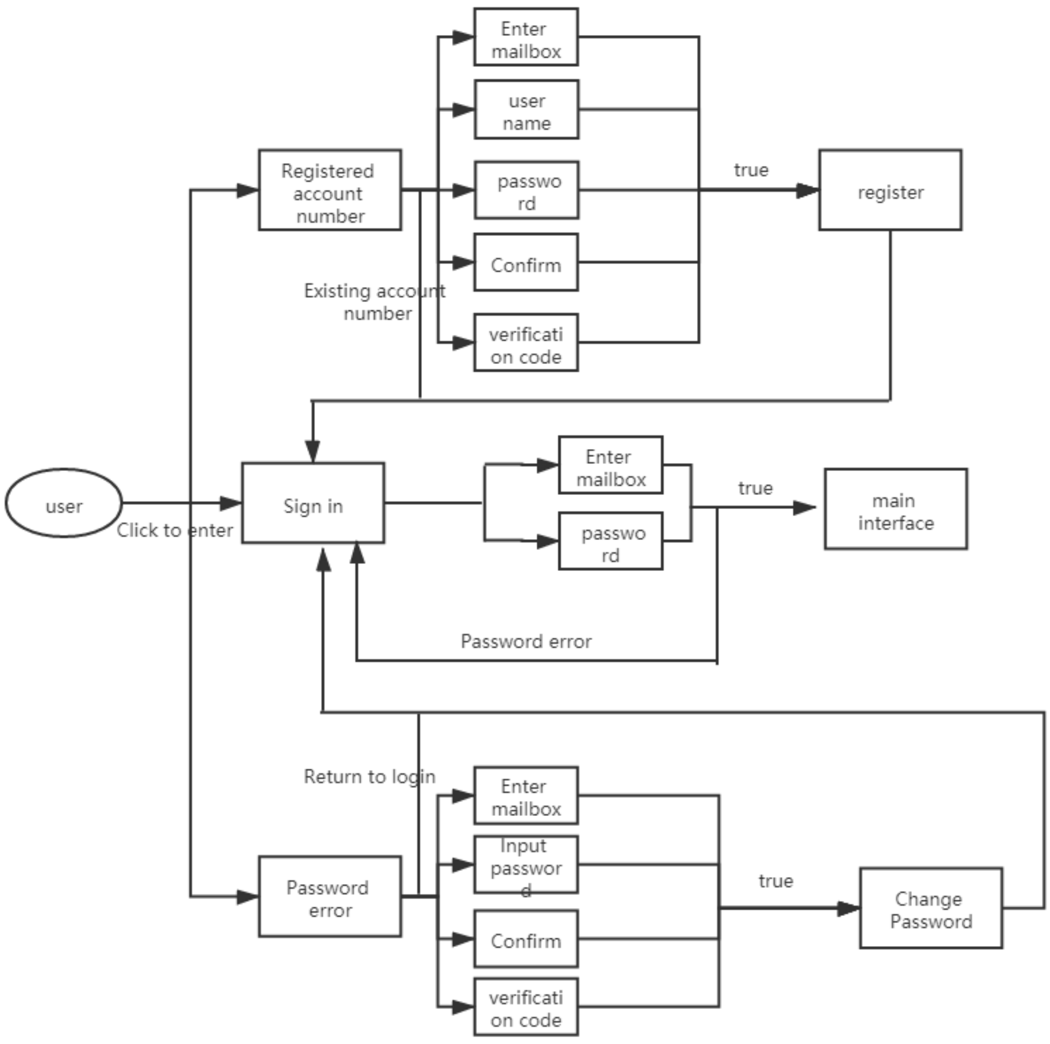
APP Operating environment: Android 8.0 system or above, running memory 2G or above
Backend Server: Ali cloud, single-core, 1M bandwidth, 2G memory
Server operation system: CentOS 7.3
Server Database: MySQL And Redis
Object Storage: Ali Cloud OSS

Support Software： Tomcat, Android studio, Docker

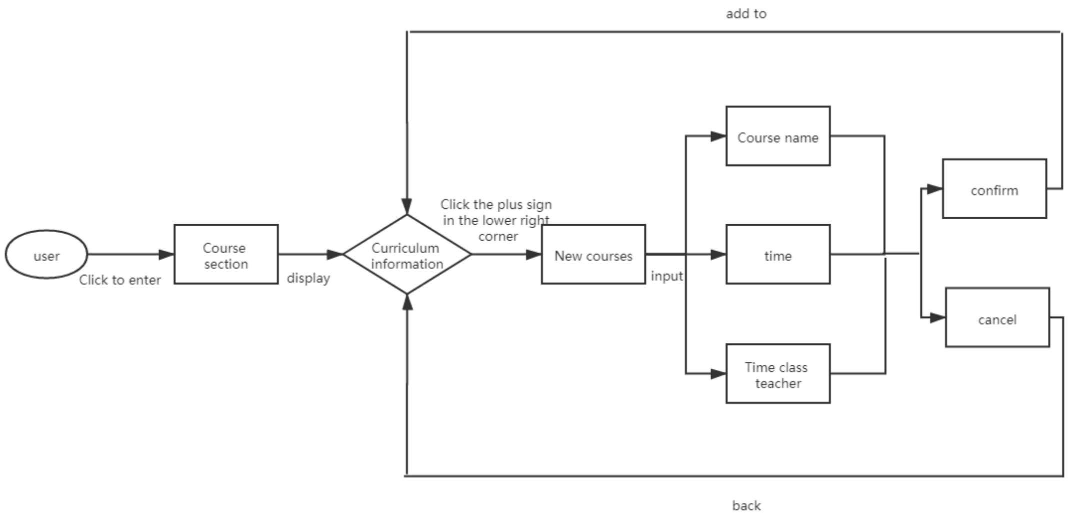
2.2.3 Basic Design Modules And Processes



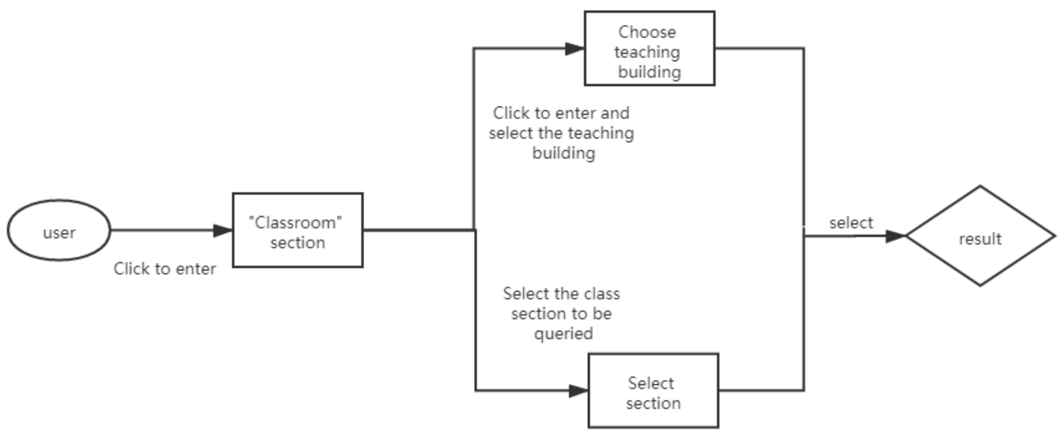
1.Functional Module



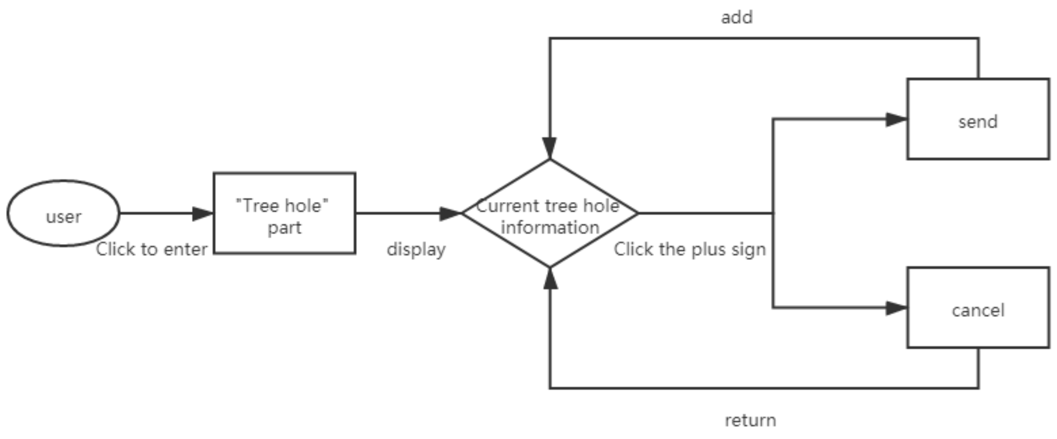
2.Register Login Forget



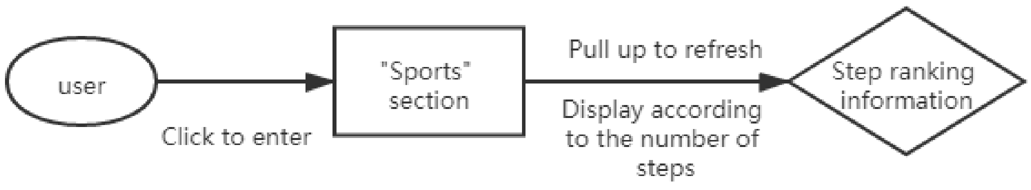
3.Class Schedule



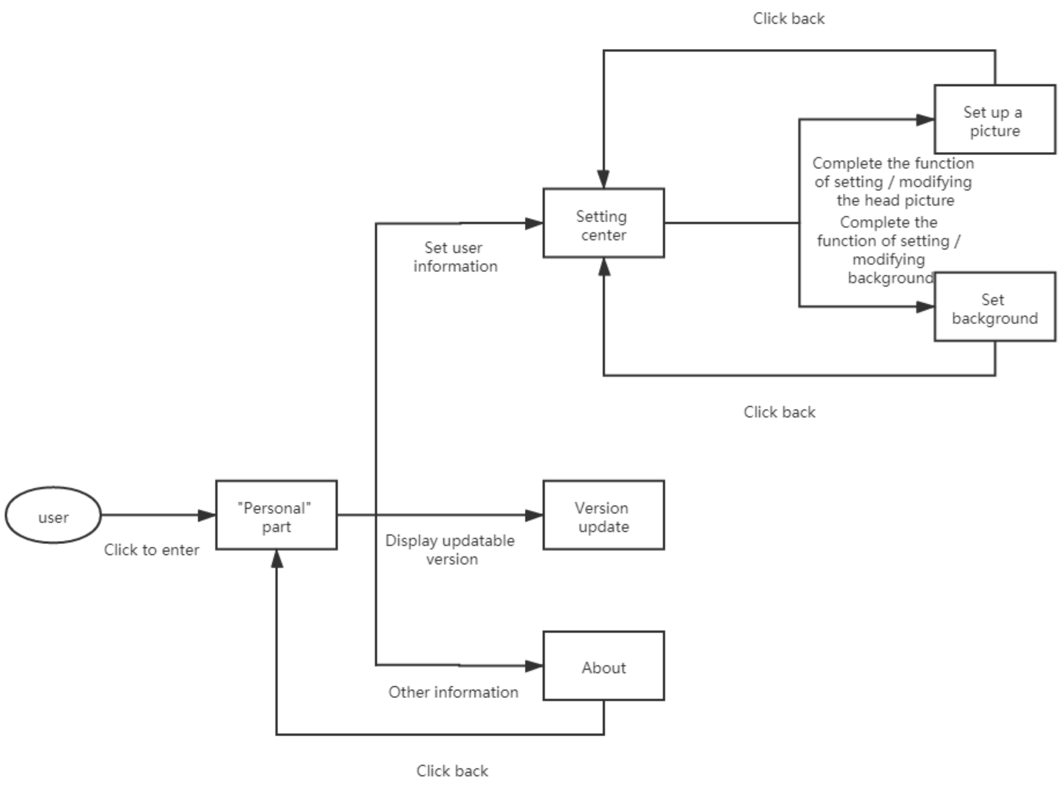
4.Spare Classroom Query



5.Tree Hole

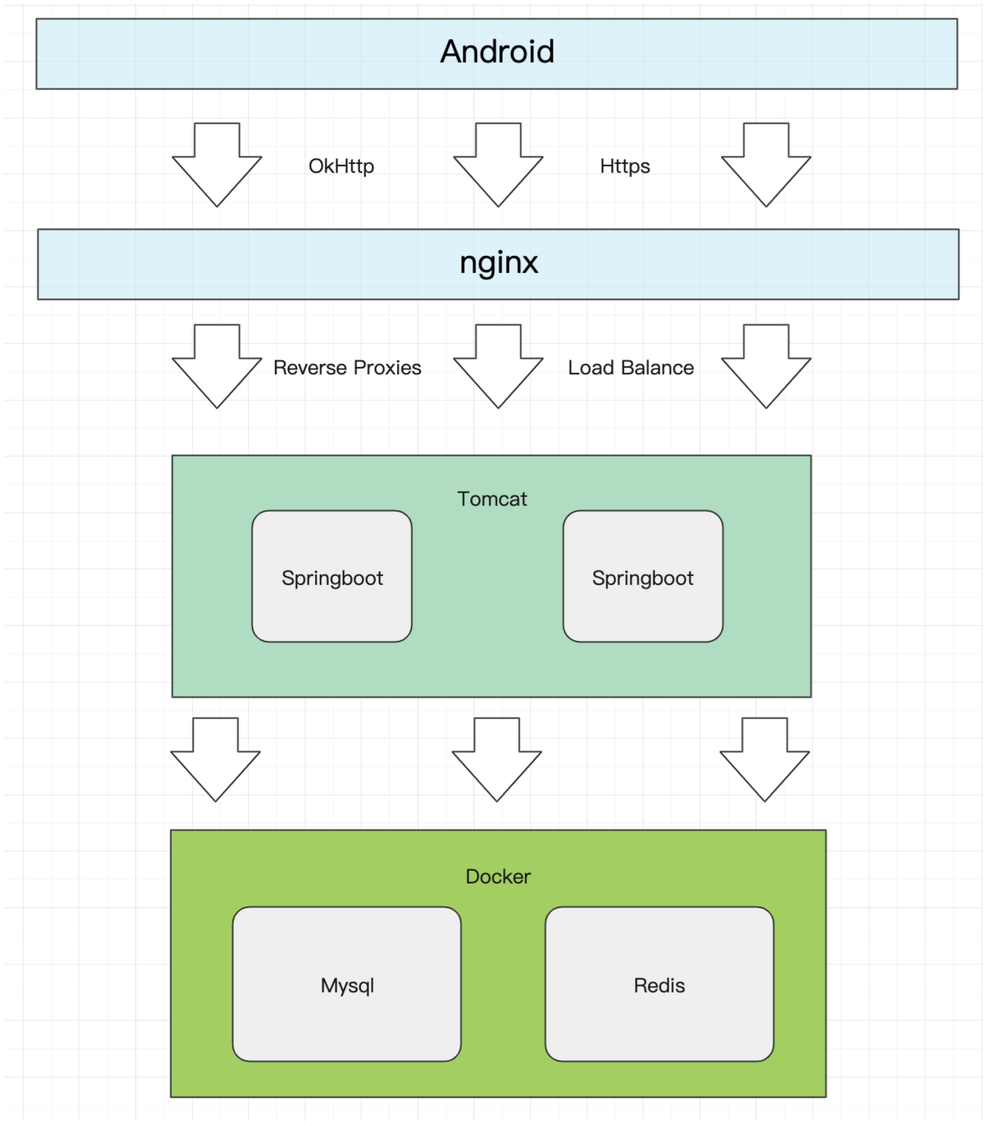


6.Steps



7.Personality

2.2.4 Architecture



2.2.5 Manual Treatment Process

The system will maintain the server database tables periodically during the later maintenance to ensure good data storage, reduce the pressure on the database and avoid data redundancy

Except for the above, the system does not require any manual processing.

2.2.6 Unresolved Issues

No unresolved issues at this time

2.3 Interface Design

2.3.1 User Interface

- Login interface: consists of mailbox input box, password input box and login button
- Registration interface: consists of email input box, user name input box, password input box, confirm password input box, email verification code input box and send verification code button.
- Schedule interface: there is a button to add a course, long press the added course to delete
- Free classroom interface: there are building, time slot selection box and query button
- Tree hole interface: there is a button to add a new tree hole
- Sports interface: you can slide down to refresh the ranking
- Personal information interface: you can set avatar, background image, view version information and software related information.

2.3.2 Internal Interfaces

This system is mainly based on the back-end database system, so the main interface of this system is to exchange data with the back-end Springboot service.

2.4 Operational Design

2.4.1 Combination Of Runtime Modules

The system is presented in the form of an APP, consisting of the main interface and each window interface. Any combination of running modules must include the login module, the registration module, and the main interface. The main interface can call each window interface to realize the combination of modules, and each window interface is independent of each other. In general, there is no need for data transfer between modules except for authentication information. Each interface implements a specific function.

2.4.2 Operation Control

Run control is implemented using function calls between modules.

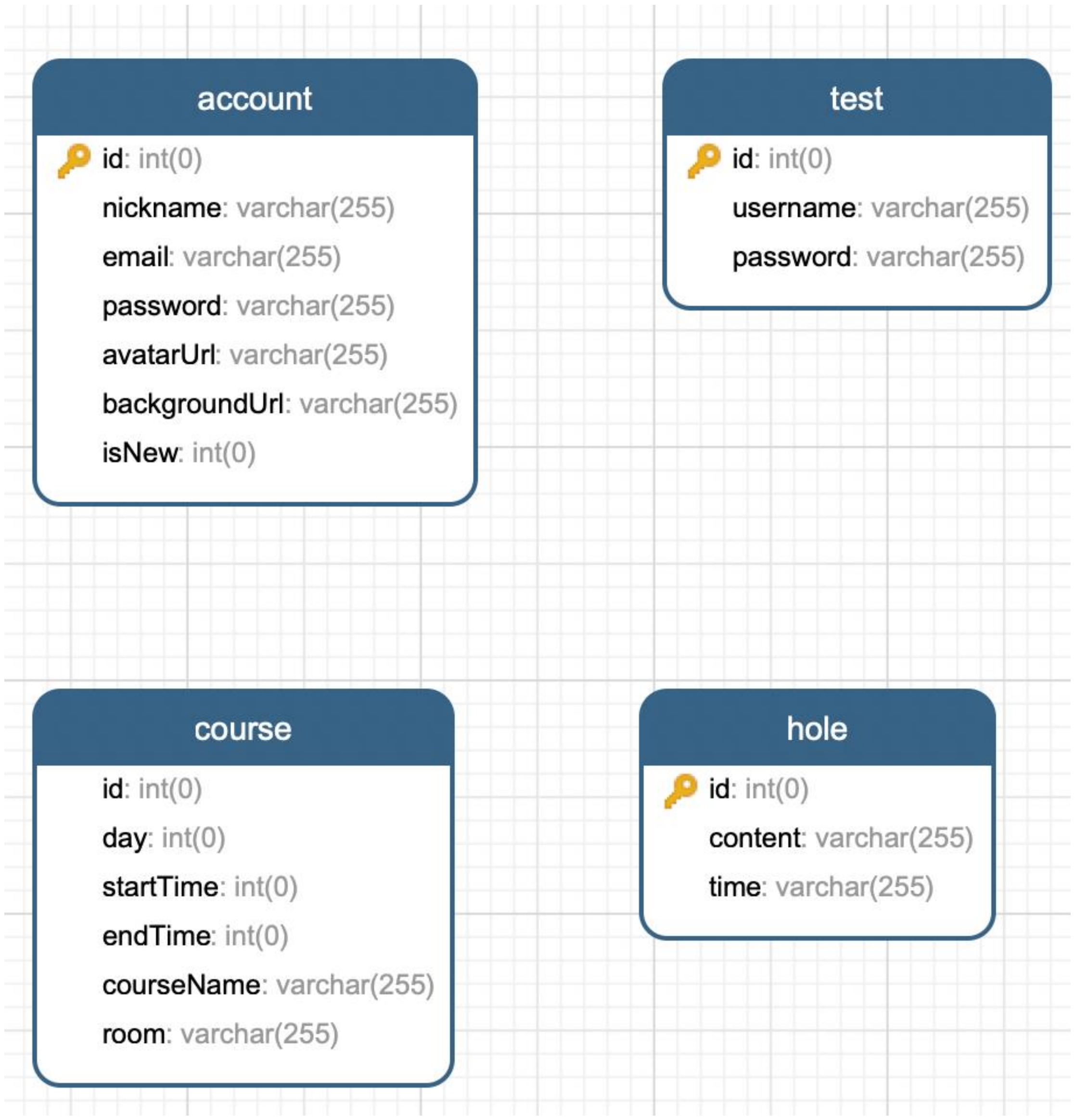
When the client interacts with the server, i.e., network transmission, the client sends a request to the server, the server receives the data, interacts with the database, performs add, delete, change, and check operations, and then wraps the results and returns them to the client, and the client receives the data, processes the data, and then returns to the interface to present the data.

2.4.3 Running Time

The timing of each module combination cannot be determined, depending on the user's operation, the network conditions, and the size of the actual data being operated.

2.5 System Data Structure Design

2.5.1 Logical Structure Design Points



2.5.2 Physical Structure Design Points

The user password data items are stored in the data base table using Blowfish encryption, and all data items except the password data items are stored in the data base table in plaintext.

The access method of data is provided by JDBC connection.

The access unit of the database is determined according to the data type of the data items.

2.5.3 Relationship Between Data Structures And Programs

The data structure of the database base table is defined at the time of defining the base table, and the data items are defined. The program uses SQL statements through JDBC connection and calls JDBC program execution to modify the database data structure and data items.

2.6 System Error Handling Design

2.6.1 Error Messages

Fault Name	Output Form	Processing
Server is not accessible	Server output error message	Perform a reboot of the server
Database connection failure	Server output error message	Check if the database service is started
User cannot connect to the server	APP prompts connection failure	Check for network problems
User input is not legitimate	APP prompts for correct input	User operation cannot be performed, re-enter

2.6.2 Remedial Measures

Backup server WEB code for program recovery or project redeployment in case of program problems. Backup database files, in the form of SQL files, for timely recovery in case of data file errors or data file loss.

2.6.3 System Maintenance Design

The system is currently not designed to maintain a dedicated maintenance module for system maintenance, but only for timely maintenance of server-side database information.

3. Detailed Design

3.1 Overview

There are 5 functional modules, which are user module, tree hole module, user curriculum module, free classroom module, step counting module, and data cache and OSS object storage modules.

The cache module has the function of caching the avatar and background image.

The OSS object storage module has cloud storage image background images and provides the function of obtaining the URL of the resource.

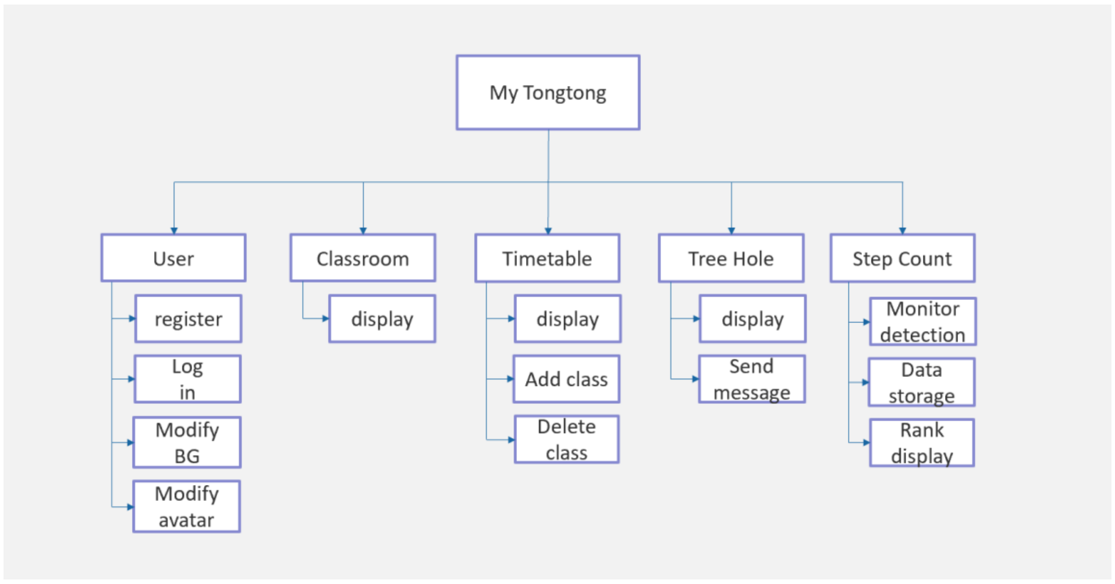
the user module has the functions of logging in, registering, modifying the avatar, and modifying the background image.

The free classroom module has the function of querying free classrooms.

The tree hole module has the functions of displaying tree holes and sending tree holes.

The course schedule module has the functions of displaying courses, adding courses and deleting courses.

The step-counting module has the functions of step-counting and ranking.



(Function diagram)

3.2 User module

3.2.1 Design of data storage structure of user module

Field name	Type	constraint	Description
avatarUrl	string		Avatar Url
background Url	string		Background Image Url
code	string		Check code
email	string	not null unique	Email address
id	int	unique	Id
isNew	int		Is it a new user
nickname	string	not null	nickname
password	string	not null	password

3.2.2 Interface design of user module

3.2.2.1 User login interface

Request Methods: POST URL: /account/login

Request Parameters:

Filed Name	Description	Type	Remarks	Required
avatarUrl	avatarUrl	String	avatat Url	False
backgroundUrl	backgroundUrl	String	background Url	False
code	Check code	String	Return null	False
email	Email	String	Email	True
id	Id	int	Id	False
isNew	Is new user	int	New:1,NO:0	False
nickname	Nickname	String	Nickname	False
password	password	String	Password	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	Account	User Info	True
msg	Interface message	String	Return message	True

data:

Filed Name	Description	Type	Remarks	Required
avatarUrl	avatarUrl	String	avatat Url	True
backgroundUrl	backgroundUrl	String	background Url	True
code	Check code	String	Return null	True
email	Email	String	Email	True
id	Id	int	Id	True
isNew	Is new user	int	New:1,NO:0	True
nickname	Nickname	String	Nickname	True
password	password	String	Password	True

3.2.2.2 User register interface

Request Methods: POST URL: /account/register

Request Parameters:

Filed Name	Description	Type	Remarks	Required
avatarUrl	avatarUrl	String	avatat Url	False
backgroundUrl	backgroundUrl	String	background Url	False
code	Check code	String	Return null	True
email	Email	String	Email	True
id	Id	int	Id	False
isNew	Is new user	int	New:1,NO:0	False
nickname	Nickname	String	Nickname	True
password	password	String	Password	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.2.2.3 Change avatar interface

Request Methods: POST URL: account/changeAvator

Request Parameters:

Filed Name	Description	Type	Remarks	Required
avatarUrl	avatarUrl	String	avatat Url	True
backgroundUrl	backgroundUrl	String	background Url	False
code	Check code	String	Return null	False
email	Email	String	Email	False
id	Id	int	Id	True
isNew	Is new user	int	New:1,NO:0	False
nickname	Nickname	String	Nickname	False
password	password	String	Password	False

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.2.2.4 Change background interface

Request Methods: POST URL: account/changeBackground

Request Parameters:

Filed Name	Description	Type	Remarks	Required
avatarUrl	avatarUrl	String	avatat Url	False
backgroundUrl	backgroundUrl	String	background Url	True
code	Check code	String	Return null	False
email	Email	String	Email	False
id	Id	int	Id	True
isNew	Is new user	int	New:1,NO:0	False
nickname	Nickname	String	Nickname	False
password	password	String	Password	False

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True

data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.2.2.5 User forget password interface

Request Methods: POST URL: account/forget
Request Parameters:

Filed Name	Description	Type	Remarks	Required
avatarUrl	avatarUrl	String	avatat Url	False
backgroundUrl	backgroundUrl	String	background Url	False
code	Check code	String	Return null	True
email	Email	String	Email	True
id	Id	int	Id	False
isNew	Is new user	int	New:1,NO:0	False
nickname	Nickname	String	Nickname	False
password	password	String	Password	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.2.2.6 Send code to email when forgetting password interface

Request Methods: Get URL: /account/getForgetCode/{email}
Request Parameters:

Filed Name	Description	Type	Remarks	Required
email	Email	String	Email	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required

code	Interface status code	Number	correct ： 0 ， error：1	True
data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.2.2.7 Send code to email when registering interface

Request Methods：Get URL：/account/getRegisterCode/{email}

Request Parameters：

Filed Name	Description	Type	Remarks	Required
email	Email	String	Email	True

Response Parameters：

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct ： 0 ， error：1	True
data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.2.2.8 Judge unique email interface

Request Methods：Get URL：/account/judgeEmail/{email}

Request Parameters：

Filed Name	Description	Type	Remarks	Required
email	Email	String	Email	True

Response Parameters：

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct ： 0 ， error：1	True

data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.2.2.9 Judge unique nickname interface

Request Methods: Get URL: /account/judgeNickname/{email}
Request Parameters:

Filed Name	Description	Type	Remarks	Required
nickname	nickname	String	Nickname	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	String	Return empty String	True
msg	Interface message	String	Return message	True

3.3 Free classroom Module

3.3.1 Design of data storage structure of free classroom module

Field name	Type	constraint	Description
building	string	not null	Building
course	String	not null	Course time period

3.3.2 Interface design of free classroom module

3.3.2.1 Query free classroom interface

Request Methods: POST URL: /queryClass/freeroom

Request Parameters:

Filed Name	Description	Type	Remarks	Required
bilding	Building	String	building	True
course	Course	String	Course time period	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	Data	List<String>	Class info	True
msg	Interface message	String	Return message	True

3.4 User class list module

3.4.1 Design of data storage structure of user class list module

Field name	Type	constraint	Description
Id	Int	not null	Account ID
courseName	String	not null	Course time period
day	Int	not null	Day
room	String	not null	class room
startTime	Int	not null	Start time
endTime	Int	not null	End time

3.4.2 Interface design of class list module

3.4.2.1 Load class interface

Request Methods: GET URL: /classlist/{id}

Request Parameters:

Filed Name	Description	Type	Remarks	Required
id	account id	int	account id	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	List<course>	course Info	True
msg	Interface message	String	Return message	True

data:

Filed Name	Description	Type	Remarks	Required
Id	Id	int	Building	True
courseName	Course name	String	Course time period	True
day	Day	int	Day	True
room	Class room	String	class room	True
startTime	Course start time	int	Start time	True
endTime	Course end time	int	End time	True

3.4.2.2 Add class interface

Request Methods: POST URL: /classlist/add

Request Parameters:

Filed Name	Description	Type	Remarks	Required
Id	Id	int	Account id	True
courseName	Course name	String	Course time period	True
day	Day	int	Day	True
room	Class room	String	class room	True
startTime	Course start time	int	Start time	True
endTime	Course end time	int	End time	True

Response Parameters：

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct ： 0 ， error： 1	True
data	data	String	Return empty message	True
msg	Interface message	String	Return message	True

3.4.2.3 Delete class interface

Request Methods： POST URL： /classlist/delete
Request Parameters：

Filed Name	Description	Type	Remarks	Required
Id	Id	int	Account id	True
courseName	Course name	String	Course time period	False
day	Day	int	Day	True
room	Class room	String	class room	False
startTime	Course start time	int	Start time	True
endTime	Course end time	int	End time	False

Response Parameters：

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct ： 0 ， error： 1	True
data	data	String	Return empty message	True
msg	Interface message	String	Return message	True

3.5 Step counter module

3.5.1 Design of data storage structure of step counter module

Use the hash structure of redis to store the number of steps of each user. Each user name is a field, and the number of steps is the number of steps of the corresponding user.

3.5.2 Interface design of step counter module

3.5.2.1 Get all steps interface

Request Methods: POST URL: /step/getStep

Request Parameters:

Filed Name	Description	Type	Remarks	Required
nickname	Nickname	String	Current nickname	True
steps	Steps	String	Current steps	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	List<Step>	course Info	True
msg	Interface message	String	Return message	True

data:

Filed Name	Description	Type	Remarks	Required
nickname	Nickname	String	Nickname	True
steps	Steps	String	Steps	True
avatarUrl	Avatar URL	String	Avatat url	True

3.6 Tree hole module

3.6.1 Design of data storage structure of tree hole module

Field name	Type	constraint	Description
id	Int	not null	Hole ID
content	String	not null	Content
time	String	not null	Date of hole published

3.6.2 Interface design of tree hole module

3.6.2.1 Add hole interface

Request Methods: POST URL: /hole/addHole
Request Parameters:

Filed Name	Description	Type	Remarks	Required
id	Hole id	int	Hole id	True
content	Hole content	String	Hole content	True
time	Date of hole published	String	Date of hole published	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	String	Return empty message	True
msg	Interface message	String	Return message	True

3.6.2.2 Load more hole interface

Request Methods: GET URL: /hole/holeLoadMore/{id}
Request Parameters:

Filed Name	Description	Type	Remarks	Required
id	hole id	int	account id	True

Response Parameters:

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct : 0 , error: 1	True
data	data	List<Hole>	Hole Info	True
msg	Interface message	String	Return message	True

data:

Filed Name	Description	Type	Remarks	Required
id	Hole id	int	Hole id	True
content	Hole content	String	Hole content	True
time	Date of hole published	String	Date of hole published	True

3.6.2.3 Refresh hole interface

Request Methods：GET　URL：/hole/holeRefresh/{id}
Request Parameters：

Filed Name	Description	Type	Remarks	Required
id	hole id	int	account id	True

Response Parameters：

Filed Name	Description	Type	Remarks	Required
code	Interface status code	Number	correct ： 0 ， error：1	True
data	data	List<Hole>	Hole Info	True
msg	Interface message	String	Return message	True

data:

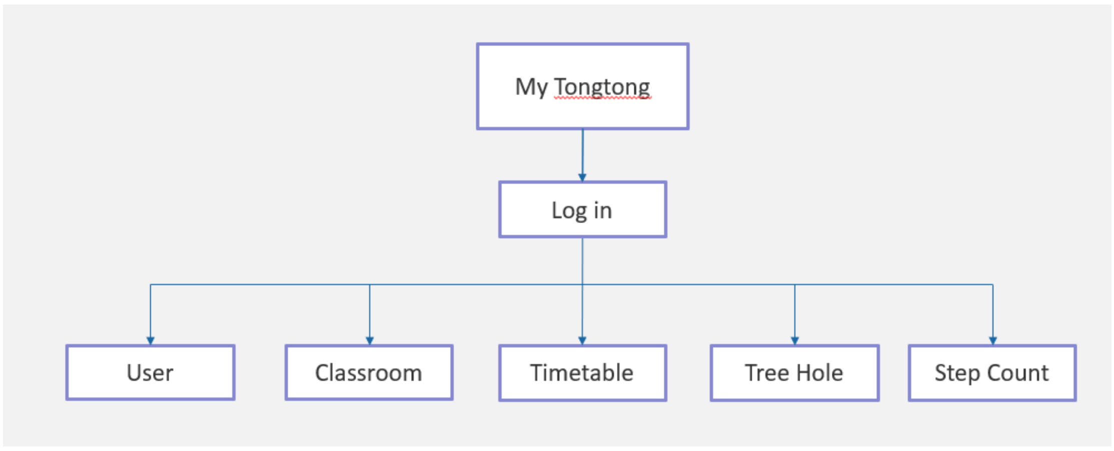
Filed Name	Description	Type	Remarks	Required
id	Hole id	int	Hole id	True
content	Hole content	String	Hole content	True
time	Date of hole published	String	Date of hole published	True

4 Implementation

4.1 Summary

To implement all the functions we design, our project has six major module: Log in module, Class schedule module, Spare classroom module, Tree hole module, steps ranking interface and personality module.

The structure of our project is like following: log in module is the main entrance after our application runs. After user successfully log in, rest modules will be accessible to user, displayed at the same level.



4.2 Log In Module

4.2.1 Functions

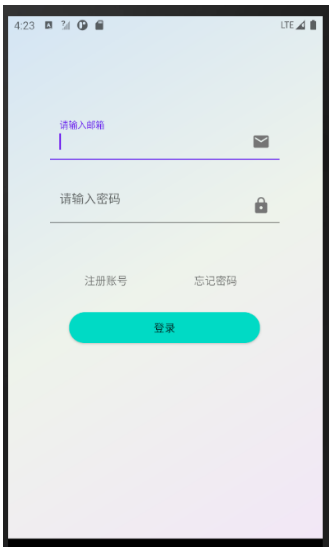
Log in module is responsible for three functions: register, log in and reset password.

4.2.2 Interface Implement

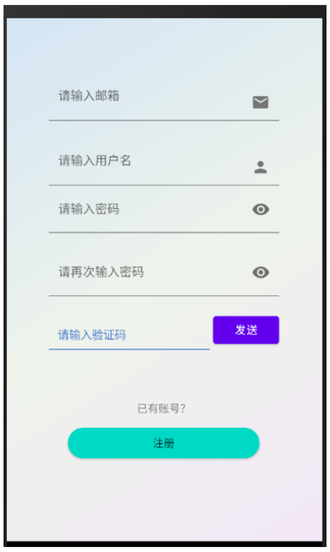
To log in, user need to enter the mailbox and the password, which will show the log in animation if the password mailbox match. If this is the first time for user to log in, our application will show a guide interface.

For new user, log in interface allow user to register a new account. It requires user to enter some information: mailbox, user name and password. Our application will check the form of mailbox, duplication of user name, require twice enter of password, and require the confirmation code sent to the mailbox.

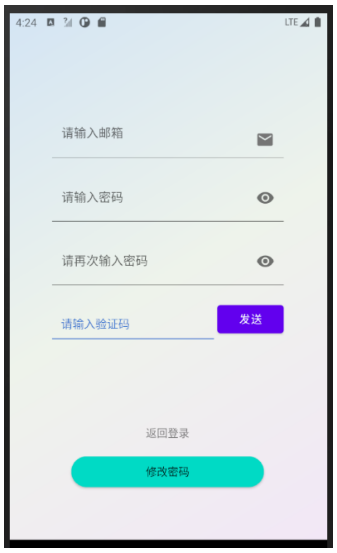
If the user forgets his password, he can reset them in the password reset. To reset the password, user need to enter the mailbox, new password, and a confirmation code sent to the mailbox.



Log in

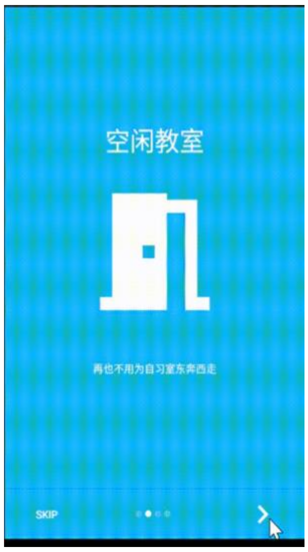


create account



reset password

After user log in, our system will check if this is the first time for this account to enter our system. If this is true, we will show a guide interface to introduce all the functions in our application. This guide include five sub interface describing the function of each individual interface.



4.2.3 Network Interface

interface Name	Input form	Output form	URL	function
login	Mailbox, password	Error code Data message	/login	Check user login info

register	Mailbox, password, Confirmation code, nickname	Error code Data message	/register	Register a new account
Send confirmation code	email	Error code Data message	/getRegisterCode/{email}	Send confirmation code to target email for register
Send confirmation code	email	Error code Data message	/getForgetCode/{email}	Send confirmation code to target email for reset password
Reset password	Mailbox, password, Confirmation code,	Error code Data message	/forget	Reset password of a target account
Check uniqueness of email	Email	Error code Data message	/judgeEmail/{email}	Check uniqueness of email
Check uniqueness of nickname	nickname	Error code Data message	/judgeNickname/{nickname}	Check uniqueness of nickname

4.2.4 Server Implement

In Log in module System server will interreact with MySQL database and Redis database, account table. It will answer to following request:

1. log in request. Server will receive request for user to log in. After receive account’s email and password, it will search in the database to check whether there is a match. If there is, it will send back the login permission. If there isn’t, it will send back the error code suggest that user’s input is not legitimate.
2. Register request. Server will receive request for user to create a new account. After receive the request with account data and email confirmation code, server will first check the integrity of the confirmation code. If it expires or mis entered, the register will be denied and feedback is sent back to server. Otherwise, a new account will be

created.

3. Send confirmation code for register. Server will automatically generate a 6 digit confirmation code to de entered email. This code will be stored in Redis with an expire time of 5 minutes.
4. Send confirmation code for register. Operation is similar to the previous one.
5. Reset password. Server will receive request from user to reset account password. After receive the request with account data and email confirmation code, server will first check the integrity of the confirmation code. If it expires or mis entered, reset operation will be denied and feedback is sent back to server. Otherwise, a new account will be created.
6. Check uniqueness of email. Server will check whether this email address already exists in the database. After receive the request, server will search all the data in the account table to see whether this email address is already used. Then a feed-back will be sent back to the server.
7. Check uniqueness of nickname. The operation is similar to the previous one.

4.3 Class Schedule Module

4.3.1 Functions

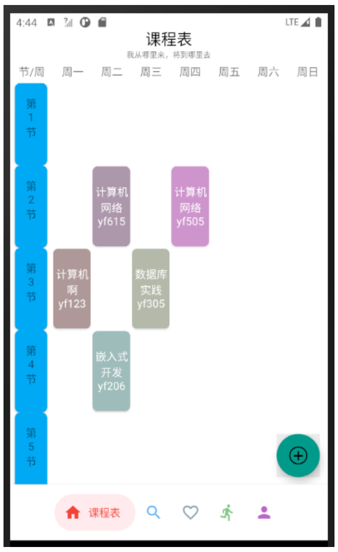
In this module, user is able to check his own class schedule. This module implements the design function of schedule display, class adding and class deleting.

4.3.2 Interface Implement

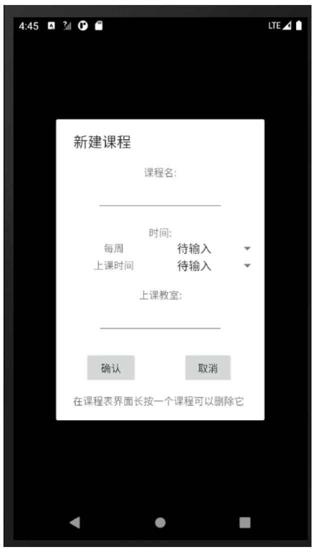
To display the class, our system uses cardview to display each class. Each class view will be given a random color and will be add a listener that will show an interface to delete it from database if it is pressed and hold.

A float button is used for the adding of classes. Press it and a new intent will be start, enable user to enter the content info of the class he wants to add. This intent will check the integrity of the content, and send the request to the server if it checks out. After the message is sent, it will reload the display.

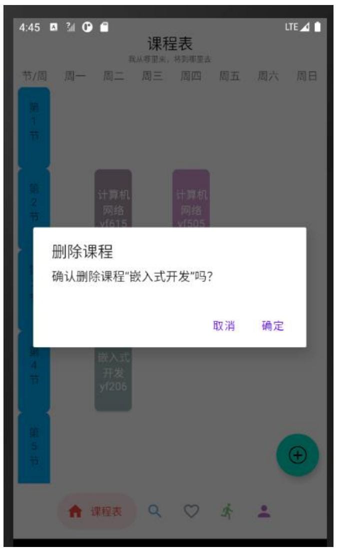
After holding the button for two second, an alter dialog will be shown. It will ask user to confirm his operation on deleting select class. If user confirms, it will send the request to the server and reload this interface.



Class schedule



add class



delete class

4.3.3 Network Interface

interface Name	Input form	Output form	URL	function
Add course	Class name Start time End time place	Error code Data message	/add	Add one course to account's schedule
Get course	id	Error code Data message	/{id}	Get account's schedule
Delete course	Class name Start time End time place	Error code Data message	/delete	Delete one course from account's schedule

4.3.4 Server Implement

In class schedule module System server will interreact with MySQL database. It will answer to following request:

1. Get course. Server will receive request from user to get all the classes. After receive account's id, it will search all the course in the course table that match the given id. All the match result then will be sent to the application.
2. Add course. Server will receive request from user to add a new course to this account. Server will first check if there is a conflict course for this account. If

there isn’t a new course will be added to the database.

3. Delete course. Server will receive request from user to delete a course. After receiving the course data, server will remove that course from database.

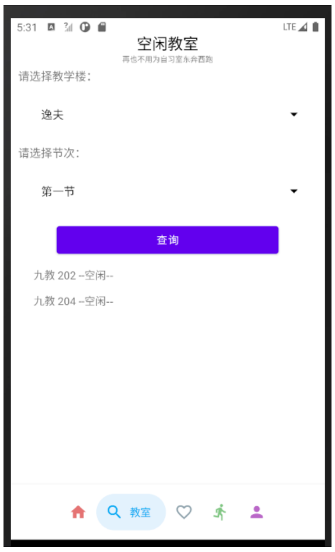
4.4 Spare Classroom Module

4.4.1 Summary

In this module, user is able to check the available class room for self-study. This module implements the design function of spare room check.

4.4.2 Interface Implement

This interface contains two select list and a pressable button, To search the available room, user need to select the target building and level to search, and the available room will be listed below the select button.



九教 202 --空闲--

九教 204 --空闲--

4.4.3 Network Interface

interface Name	Input form	Output form	URL	function
Check free room	Building level	Error code Data message	/freeroom	Check free room available

4.4.4 Server Implement

In Log in module System server will interreact with MySQL database. It will answer to following request:

1. Get course. Server will receive request from user to get free classroom. After receive building name and level, it will search all the classroom in the classroom table that is available at current time. All the match result then will be sent to the application.

4.5 Tree Hole Module

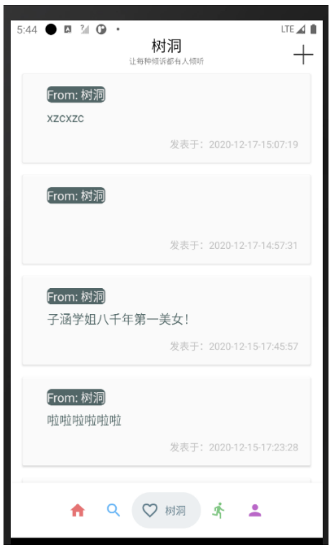
4.5.1 Summary

In this module, our system will display all the anonymous information sent to the server. This module implements the design function of Tree hole.

4.5.2 Interface Implement

To display the message, our system uses cardview to display each message. Each message will be shown with a time tag and a source sender. Initially there will be most recently ten messages to be displayed. User can slide down to get another ten, or slide up for refresh. This is implemented with BGA refresh layout.

To send a new message, user can press the plus mark on the right upper conner. A view will be shown for user to input the message. After enter the full message, user can press the send button and the request will be send to the server.

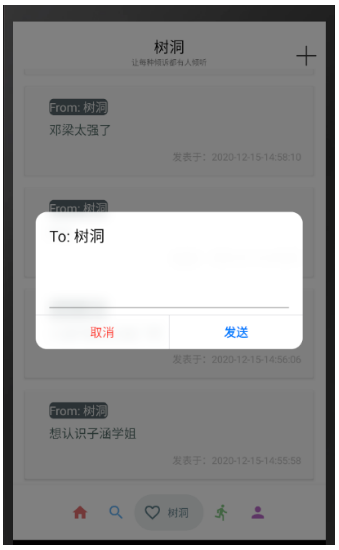


From: 树洞
xzcxzc
发表于：2020-12-17-15:07:19

From: 树洞
发表于：2020-12-17-14:57:31

From: 树洞
子涵学姐八千年第一美女！
发表于：2020-12-15-17:45:57

From: 树洞
啦啦啦啦啦啦
发表于：2020-12-15-17:23:28



From: 树洞
邓梁太强了
发表于：2020-12-15-14:58:10

To: 树洞

发表于：2020-12-15-14:56:06

From: 树洞
想认识子涵学姐
发表于：2020-12-15-14:55:58

4.5.3 Network Interface

interface Name	Input form	Output form	URL	function
Add hole	Id Content time	Error code Data message	/addHole	Add one message to treehole
Hole Refresh	id	Error code Data message	/holeRefresh/{id}	Reload tree hole message
Hole more	id	Error code Data message	/holeLoadMore/{id}	load more tree hole message

4.5.4 Server Implement

In tree hole module System server will interreact with MySQL database. It will answer to following request:

1. Add hole. Server will receive request from user to add a new message. A new message will be insert into the message table.
2. Hole refresh. Server will receive request from user to resent all messages. All the messages will be sent right away.
3. Hole more. Server will receive request from user to load more message. Operation is similar to the previous one.

4.6 Sports Module

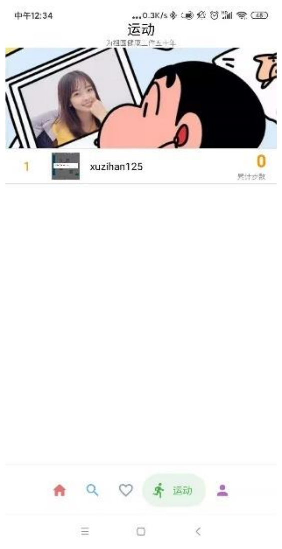
4.6.1 Summary

In this module, our system will display the step ranking list of all users. This interface implements the design function of sports.

4.6.2 Interface Implement

To detect where user is walking and count the step, our system use sensor. TYPE_STEP_COUNT sensor to detect, and store the info in Redis database. All the data will be expired on 0 o’clock every day.

After all data is detected, a ranking list will be displayed. Our system use listview, baseAdapter and headView to display. In order to display user’s avartar with a quick responds speed, we use OKHttp protocol to transfer image file, and use cache for local storage.



4.6.3 Network Interface

interface Name	Input form	Output form	URL	function
Get step	Nickname, Step count,	Nickname, Step count, Avatar url	/getStep	Post current footstep count of this user, get the rank list of all users

4.6.4 Server Implement

In tree hole module system server will interreact with MySQL and Redis database. It will answer to following request:

1. Get step. Server will receive request from user to get step count rank list. After receive the request, server will first update the record content in Redis. Then, it will retrieve all content, sort them by step count, and send them back to application in order.

4.7 Personality Module

4.7.1 Summary

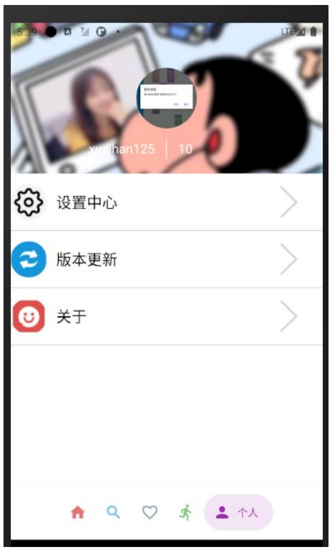
In this module, our system will allow user to change personal settings, including changing avatar and background, update application, and display application information. This module implements the design function of profile setting.

4.7.2 Interface Implement

This interface begins with three pressable button, where each button will lead to a new activity. Setting center button will lead to user profile modification interface, where user can reset their background and avatar. To do so, our application will need to access device’s photo file.

Update button will update the application. It will check the version number of current application. If the application need update, it will download the newest APK, and if it is not, it uses toast to say “no update is needed”.

About button will display the application info interview. This will display the information about this application.



4.7.3 Network Interface

interface Name	Input form	Output form	URL	function
Change avatar	Mailbox, password,	Error code Data message	/changeAvatar	Change avatar of an account

Change background	Mailbox, password,	Error code Data message	/changeBackground	Change background of an account

4.7.4 Server Implement

In personality module System server will interreact with MySQL and Redis database. It will answer to following request:

1. Change avatar. Server will receive request from user to reset avatar to a new image. After receive the request, server will set the corresponding avatar URL to the new one and receive image transition from user.
2. Change background. The operation is similar to the previous one.

5 Test

5.1 Log In Module

5.1.1 Log In

Test name	Test content	result
Empty email	email:	Toast
Empty password	email:xuzihan125@163.com password:	Toast
Correct log in	email:xuzihan125@163.com password:xzh111	Enter main interface
Wrong log in	email:xuzihan125@163.com password:111111	Toast

5.1.2 Register

Test name	Test content	result
Empty email	email:	Red warning
Incorrect email form	email:xuzihan125	Red warning

Duplicate email	email:xuzihan125@163.com	Red warning
Empty nickname	email:xuzihan@163.com nickname:	Red warning
Duplicate nickname	email:xuzihan@163.com nickname:xuzihan	Red warning
Empty password	email:xuzihan@163.com nickname:xuzihan password:	Red warning
Mismatch password	email:xuzihan@163.com nickname:xuzihan password:xzh111 password:123123	Red warning
Empty Confirmation code	email:xuzihan@163.com nickname:xuzihan password:xzh111 password:123123 confirmation code:	Red warning
Mismatch Confirmation code	email:xuzihan@163.com nickname:xuzihan password:xzh111 password:123123 confirmation code:801745	Red warning
Correct log in	email:xuzihan@163.com nickname:xuzihan password:xzh111 password: xzh111 confirmation code:847383	Return to home page

5.1.3 Reset Password

Test name	Test content	result
Empty email	email:	Red warning
Incorrect email form	email:xuzihan125	Red warning
Empty password	email:xuzihan@163.com password:	Red warning
Mismatch password	email:xuzihan@163.com password:xzh111 password:123123	Red warning

Empty Confirmation code	email:xuzihan@163.com password:xzh111 password:123123 confirmation code:	Red warning
Mismatch Confirmation code	email:xuzihan@163.com password:xzh111 password:123123 confirmation code:801745	Red warning
Correct reset password	email:xuzihan@163.com password:xzh111 password: xzh111 confirmation code:847383	Return to home page

5.2 Class Schedule Module

Test name	Test content	result
0 Class display		In line with expectations Display empty table
With Class display		In line with expectations Display table with classes
Delete class		Class deleted
Add class: Empty class name	Class name:	Toast
Add class: unchosen time	Class name: c++ Time:	Toast
Add class: empty classroom	Class name: c++ Time:Mon. 2nd class Classroom:	Toast
Correct reset password	Class name: c++ Time:Mon. 2nd class Classroom: yf510	Back to main interface

5.3 Spare Classroom Module

Test name	Test content	result
Check classroom		In line with expectations Display list of classes

5.4 Tree Hole Module

Test name	Test content	result
display		In line with expectations Display list of message
reload	Slide down	In line with expectations reload list of message
Get more message	Slide up	In line with expectations Get 10 more message
Get more message	Slide up	In line with expectations Get 10 more message
Reach bottom	Slide up	In line with expectations Toast
Add message	Content:test1234	In line with expectations Message add to display

5.5 Steps Ranking Interface

Test name	Test content	result
display		In line with expectations Display rank list of step count
reload	Slide down	In line with expectations reload rank list of step count
Check sensor	Walk with phone	In line with expectations Step count increase

5.6 Personality Module

Test name	Test content	result
Reset avatar	Chose another picture	In line with expectations Avatar changed
Reset background	Chose another picture	In line with expectations Background changed
update		In line with expectations
about		In line with expectations

6 Cooperation

6.1 Members

The composition of our team is as follows:

No.	Name	ID	Role
1	Zihan Zhou (周子涵)	18301060	Project Manager, Development Engineer
2	Liang Deng (邓梁)	18221221	System Architect, Development Engineer
3	Junze Li (李俊泽)	18301044	Development Engineer
4	Zihan Xu (徐子涵)	18301055	Development Engineer
5	Zicheng Xu (徐紫程)	18301054	Development Engineer
6	Rui Xing (邢睿)	18301052	Test Engineer

6.2 Task Allocation

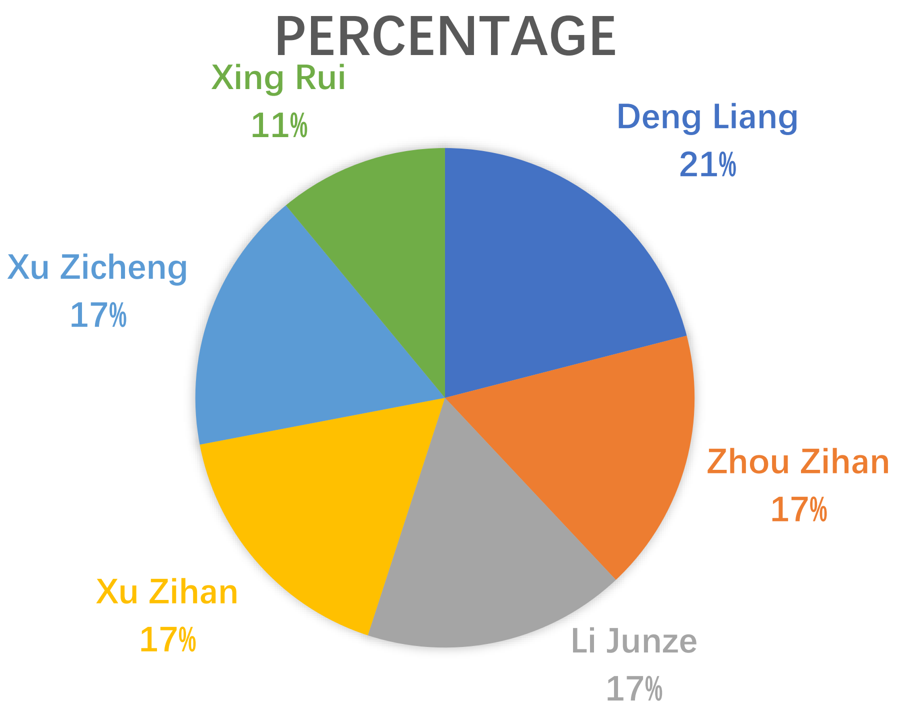
The allocation of the main tasks of the project is as follows:

No.	Name	Task
1	Zihan Zhou (周子涵)	Project management, Tree hole function implementation, Spare classroom query function implementation, Project testing, document writing
2	Liang Deng (邓梁)	Architecture design, Back-end server implementation, Database management, Network transmission encapsulation, Registration and login function implementation, Project testing, document writing
3	Junze Li (李俊泽)	Step count function implementation, Local cache function implementation, Project testing, document writing
4	Zihan Xu (徐子涵)	Class schedule function implementation, Spare classroom query layout design, Project testing, document writing
5	Zicheng Xu (徐紫程)	User information interface implementation, Project testing, document writing
6	Rui Xing (邢睿)	Project testing, document writing

6.3 Rank & Percentage

According to the actual workload during the development of the project, the actual contribution of each person to the project is as follows:

NO.	Name	Percentage
1	Liang Deng (邓梁)	21%
2	Zihan Zhou (周子涵)	17%
3	Junze Li (李俊泽)	17%
4	Zihan Xu (徐子涵)	17%
5	Zicheng Xu (徐紫程)	17%
6	Rui Xing (邢睿)	11%



And the rank of each person’s work is as follows:

NO.	Name	Rank
1	Liang Deng (邓梁)	A
2	Zihan Zhou (周子涵)	A
3	Junze Li (李俊泽)	A
4	Zihan Xu (徐子涵)	A
5	Zicheng Xu (徐紫程)	A
6	Rui Xing (邢睿)	A-