ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«ЧЕРНІВЕЦЬКИЙ ПОЛІТЕХНІЧНИЙ КОЛЕДЖ»

Циклова комісія інженерії програмного забезпечення

**звіт**

з Навчальної практики 2

(зі спеціальних дисциплін програмування)

Студента ІІІ курсу 531 групи

спеціальність 121

«Інженерія програмного забезпечення»

Швеця Є. В.

Керівник Кравчук О. М.

Чернівці – 2020

**Зміст**

[вступ 3](#_Toc31531160)

[Розділ 1. технічний проект 4](#_Toc31531161)

[1.1 Дослідження предметної області 4](#_Toc31531162)

[1.2 Постановка задачі 5](#_Toc31531163)

[1.3 Моделювання програмного продукту 6](#_Toc31531164)

[Розділ 2. Робочий проект 8](#_Toc31531165)

[2.1 Керівництво програміста 8](#_Toc31531166)

[2.2 Інструкція користувачу 9](#_Toc31531167)

[Висновки 13](#_Toc31531168)

[перелік джерел 14](#_Toc31531169)

[ДодатОк а 15](#_Toc31531170)

# вступ

Розумний будинок (або smart-house) — це сучасний продукт діджиталізації, що працює на основі штучного інтелекту. Вважається, що проживання у будинку з подібними налаштуваннями робить життя простішим. Система “Розумний будинок” об’єднує всі ресурси житла в одне ціле: всі функції і сфери будинку працюють злагоджено і максимально ефективно. Вона розумно керує всім домашнім господарством, заощаджуючи ваші зусилля, час і гроші, піклуючись про вас та ваш комфорт.

Саме такий будинок і став об'єктом дослідженнянавчальної практики.

Розумний будинок – це високотехнологічна система, яка може об’єднати всі комунікації вашого дому, і керувати ними одним натисканням кнопки. Освітлення, опалення, полив вазонів, стеження за температурою та інші. Наприклад, система керування світлом дає змогу запрограмувати світлові сцени у будинку чи створити видимість присутності господаря вдома під час його відпочинку в іншій країні. Система поливу вазонів дозволятиме власнику забезпечити полив рослин в оселі, якщо його немає вдома. Мультирум – система, яка забезпечить можливість слухати улюблену музику в будь-якій кімнаті, не загромаджуючи приміщень апаратурою.

Предметом дослідженняє розгляд описаних вище функцій розумного дому.

Метою навчальної практики є засвоєння отриманих в процесі розробки програмного продукту знань щодо моделювання та розробки Android-додатку, створити програмний продукт, використовуючи методи об’єктно – орієнтованого програмування.

Завданням практики є створення проекту, що реалізує певні функції розумного дому.

Для розробки проекту було вибрано середовище розробки Android Studio.

Звіт складається з двох розділів. Перший розділ містить дослідження предметної області, постановку задачі та моделювання програмного продукту.

У другому розділі описано керівництво програміста та інструкція користувачу.

# Розділ 1. технічний проект

## Дослідження предметної області

“Розумний будинок” – це високотехнологічна система, яка може об’єднати всі комунікації вашого дому, і керувати ними одним натисканням кнопки.

Системи, якими може управляти “розумний будинок”:

* “розумне світло” – світло вмикається і вимикається згідно налаштувань користувача;
* мікроклімат – підтримується певна температура в будинку;
* полив квітів – в певний час або при певних показниках вологості грунту;
* енергоощадність – відповідно до температури зовні регулюється температура всередині будинку, батареї можуть вимикатися, коли користувач залишає домівку;
* безпека – система автоматично перекриває газові вентилі та воду при витіках, сигналізація сповіщує про незваних гостей.

В наш час існує чимало відомих та нових брендів, що реалізують “smart-house”. Серед них Amazon Echo, Nest Learning Thermostat, Apple HomeKit Netatmo, Meizu LifeKit, Google Home.

Серед українських компаній і стартапів також є такі, що займаються техногіями смарт-будинків.

Компанія CLAP займається розробкою усіх систем та їх деталей, з 2018 року почали встановлювати в кожній новобудові від Укрбуду.

Ecoisme — система для керування витратами енергії у помешканнях та офісах. Система із датчиків та центрального модуля дозволяє керувати енерговитратами та підлаштовувати режими енергоспоживання під повсякденні практики та найбільш поширені сценарії використання протягом дня. Вона також підтримує роботу із сонячними панелями.

Umuni — технічна платформа для організації діяльності енергосервісних компаній, мереж закладів, торгових і офісних центрів та інших приватних підприємств. Серед основних завдань програмного забезпечення та підключеного до нього інфраструктурного рішення — паспортизація будівель та енергоменеджмент.

Вони всі різні, тому що виконують різні функції, але тема “розумний будинок” та “інноваційність” об’єднує їх.

Провівши огляд, було виявлено перелік проблем, які є наданий час у сучасних системах управління для розумних будинків. Перша з них - це висока ціна, малий перелік готових функцій, відсутність україномовного та російськомовного розпізнавання голосових команд. Слід відзначити, що сучасні бренди поступово намагаються вирішити їх, але це складне завдання.

Отже, вивчивши предметну область, ми дізналися, що висока ціна перешкоджає купити цю систему звичайному користувачу, а розпізнавання української мови робить незвичну обстановку в будинку.

## Постановка задачі

Завданням навчальної практики є розробка програмного продукту, який повинен керувати пристроями будинку, а саме:

* лампами освітлення;
* розетками живлення;
* полив вазонів;
* температурний режим будинку;
* вологість.

Ця робота є частиною проекту “SmartHouse”. Вона присвячена розробці клієнтської частини Android-додатку цього проекту. Даний програмний продукт повинен надавати можливість клієнту обирати параметри «розумного будинку» та реалізувати зв'язок між клієнтом та сервером системи.

Програмний продукт повинен мати зручний для користувача інтерфейс для користувачів Android-платформи.

Вхідними даними користувача будуть ім’я, email, код підтвердження, пароль та схема будинку (план будинку та розміщені розумні об’єкти для керування будинком).

Вихідними даними будуть стани розумних пристроїв, локальна збережена статистика параметрів будинку.

## 1.3 Моделювання програмного продукту

### 1.3.1 Діаграма прецендентів

На діаграмі прецедентів (див. рис. 1) зображено 4 актори: користувач, додаток, сервер, Arduino.

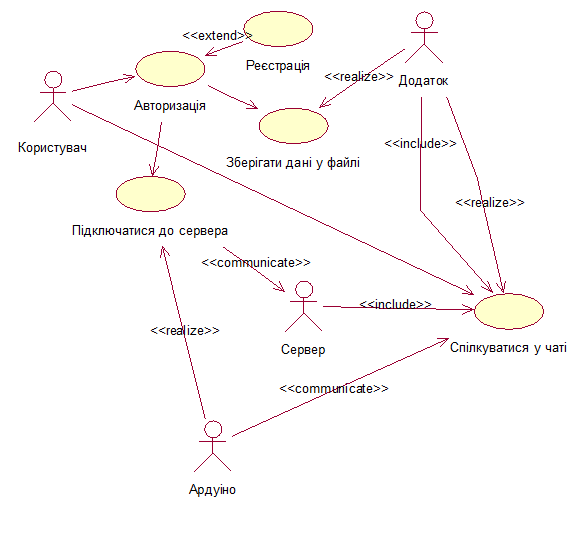


Рисунок 1 – Діаграма прецедентів проекту

На діаграмі прецедентів розміщено ввесь функціонал проекту на даний момент, у наступних версіях ця діаграма зміниться.

### 1.3.2 Діаграма класів

На рисунку 2 зображено діаграму класу “File”.



Рисунок 2 – Діаграма класу “File”

Діаграма класу “Function” зображена на рисунку 3.

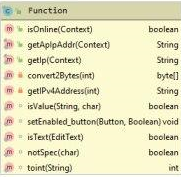


Рисунок 3 – Діаграма класу “ Function ”

На рисунку 4 зображено діаграму інтерфейсу “JsonApi”.

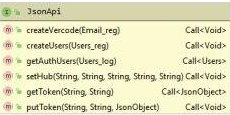


Рисунок 4 – Діаграма інтерфейсу “JsonApi”

На рисунку 5 зображено всі діаграми класів, інтерфейсів та вікон (activity).

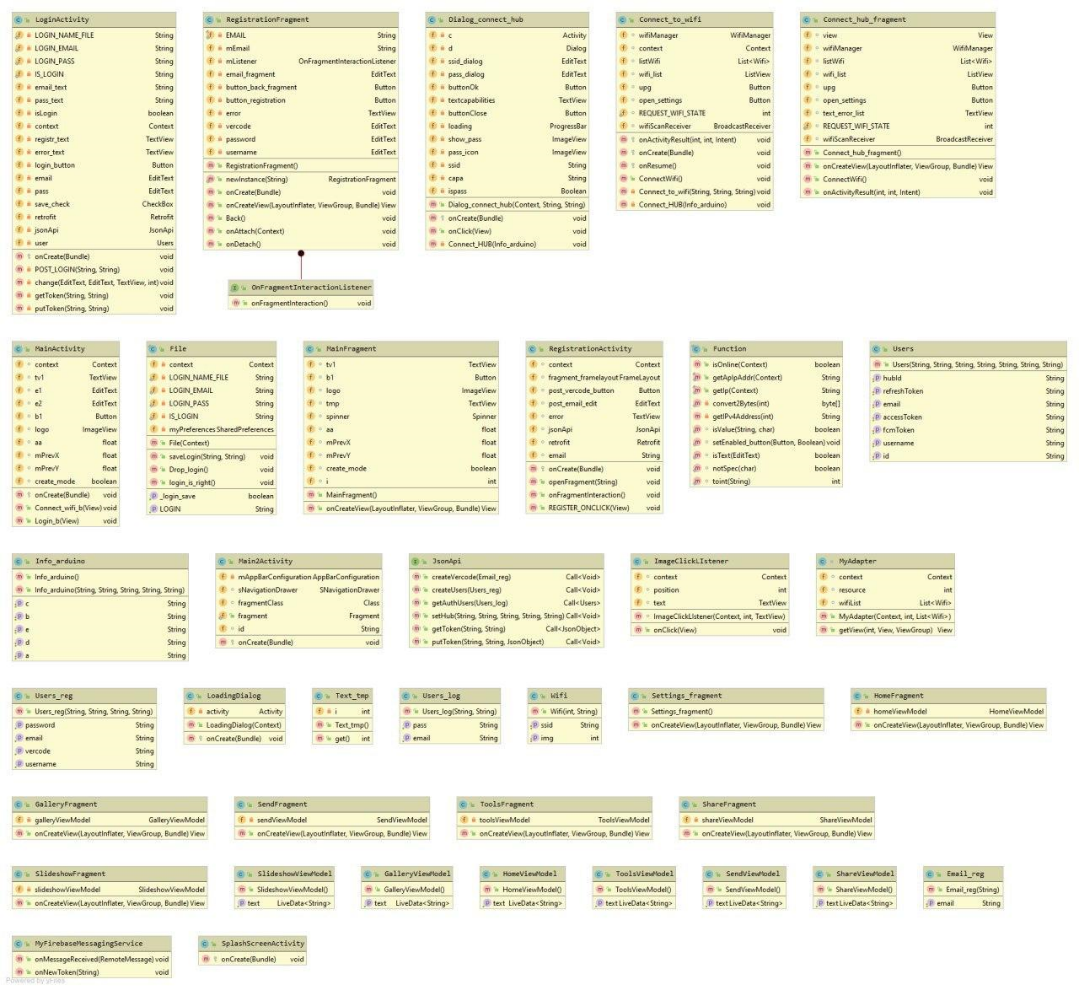


Рисунок 5 – Діаграма всіх класів

Отже, вище описано процес моделювання даних проекту.

# Розділ 2. Робочий проект

## 2.1 Керівництво програміста

Для зберігання даних про користувача був створений клас “File”, дані зберігаються у xml-файлах за допомогою “SharedPreferences” (дані захищені).

У класі існують статичні змінні для збереження ключів доступу до даних.

Методи класу:

* boolean is\_login\_save() – повертає, чи є збережені дані у файлі;
* void saveLogin() – зберігає логін та пароль у файл (параметри: логін та пароль);
* String getLOGIN() – повертає дані;
* void Drop\_login() – видаляє дані з файлу.

Для використання статичних функцій, які використовуються у всьому проекті, створено клас “Function”, у якому створено статичні методи:

* boolean isValue(String str, char ch) – перевіряє входження символу в рядку;
* void setEnabled\_button(Button button, Boolean bool) – встановлює анімаційну доступність кнопки, якщо bool – true, то прозорість – 0.5, інакше прозорість – 1;
* byte[] convert2Bytes(int hostAddress) – переводить у бінарне число;
* boolean isOnline(Context context) – перевіряє чи є доступ до інтернету.

Для створення запитів до сервера був розроблений інтерфейс “JsonApi”, в якому реалізовані прототипи для того чи іншого запиту. При цьому потрібно обрати метод запиту (GET, PUT, POST, DELETE) та вказати тип повернення.

Лістинг коду інтерфейсу розміщений у Додатку А.

## 2.2 Інструкція користувачу

Запустивши додаток, спершу відбувається ідентифікація користувача. Система зчитує дані з файлів і намагається авторизувати користувача. Якщо дані правильні, то даний користувач автоматично авторизується і переходить на головну сторінку додатку.

На рисунку 6 зображено завантаження додатку, яке створено з метою зчитування даних з файлів, щоб потім зберігати їх у динамічній пам’яті..



Рисунок 6 – Загрузка додатку

На рисунку 7 зображене вікно авторизації, кнопка авторизації неактивна, поки користувач не введе потрібні дані.



Рисунок 7 – Вікно авторизації

Якщо натиснути “Registration” у вікні авторизації, то користувачу потрібно буде зареєструватися (див. рис. 8), ввівши свою електронну пошту (в разі коректного введення на пошту висилається код підтвердження).

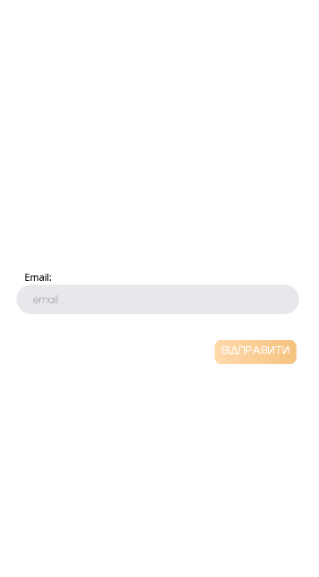


Рисунок 8 – Вікно реєстрації

Якщо пошта буде коректною, то можна продовжити процес реєстрації. Друге вікно реєстрації (див. рис. 9) створене для вводу даних для самої реєстрації. Коректно введені дані дадуть можливість успішно зареєструватися.

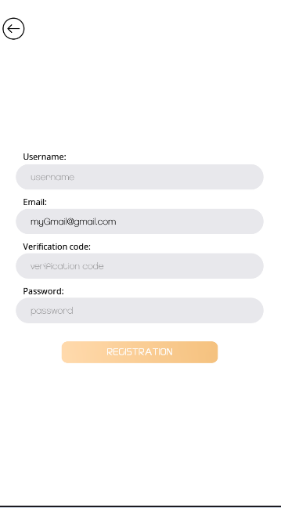


Рисунок 9 – Вікно реєстрації

Після проходження авторизації користувач переходить у головне вікно програми, що зображене на рисунку 10. У наступних версіях головне вікно буде змінюватись. На даний момент створений прототип для розміщення динамічно створених об’єктів для керування розумним будинком.



Рисунок 10 – Головне вікно

На рисунку 11 зображено структуру головного меню програми (створений прототип).

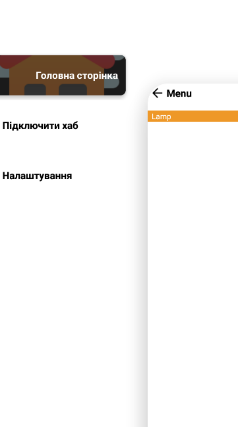


Рисунок 11 – Структура меню

Для підключення системи до сервера використовуємо пункт меню “Підключити хаб”. В результаті відкривається вікно, зображене на рисунку 12.

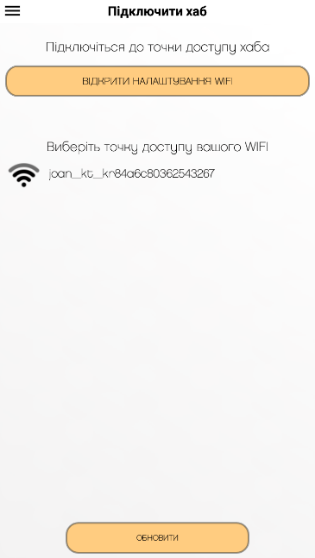


Рисунок 12 – Вікно підключення системи до сервера

На рисунку 13 зображено вікно “Налаштування”, в якому розміщені зчитані дані та кнопка видалення даних з файлу. Дані зберігаються у разі, якщо користувач обере у вікні авторизації галочку “Зберегти дані” (див. рис 7).



Рисунок 13 – Вікно налаштування

Отже, було започатковано розробку додатку для реалізації розумного будинку.

# Висновки

В ході виконання навчальної практики було проаналізовано різні системи розумних будинків та розроблено деякі функції для реалізації “розумного будинку”.

В першому розділі був проведений аналіз об’єкту автоматизації, розглянуті сучасні комп’ютеризовані системи управління розумним будинком. Розроблена постановка задачі, створена діаграма прецедентів та діаграма класів, які були реалізовані на даному етапі створення програмного продукту.

В другому розділі було описано структуру програми та функції роботи з даними. Також описано інструкцію використання програмного продукту для користувача.

Отже, на даному етапі розроблено реєстрацію/авторизацію та підключення додатку до серверу. В подальшому планується доповнити програмний продукт функціями для реалізації системи “розумний будинок”.

# перелік джерел

1. <https://developer.android.com/>
2. <https://reqbin.com/>
3. <https://material.io/design/material-theming/#using-material-theming>
4. <https://startandroid.ru/ru/>
5. <https://github.com/>

# ДодатОк а

клас “File”

package com.yevhen.project;  
import android.app.Application;  
import android.content.Context;  
import android.content.SharedPreferences;  
  
import static android.content.Context.*MODE\_PRIVATE*;  
  
public final class File extends Application {  
 public File(Context context) {  
 this.context = context;  
 }  
 private Context context;  
 private static String *LOGIN\_NAME\_FILE* = "LOGIN";  
 private static String *LOGIN\_EMAIL* = "LOGIN\_EMAIL";  
 private static String *LOGIN\_PASS* = "LOGIN\_PASS";  
 private static String *IS\_LOGIN* = "LOGIN\_SAVE";  
 private SharedPreferences myPreferences;  
  
 public boolean is\_login\_save(){  
 myPreferences = context.getSharedPreferences(*LOGIN\_NAME\_FILE*,*MODE\_PRIVATE*);  
 return myPreferences.getBoolean(*IS\_LOGIN*,false);  
 }  
 public void saveLogin(String email,String pass){  
 myPreferences = context.getSharedPreferences(*LOGIN\_NAME\_FILE*,*MODE\_PRIVATE*);  
 SharedPreferences.Editor editor = myPreferences.edit();  
  
 editor.putBoolean(*IS\_LOGIN*,true);  
 editor.putString(*LOGIN\_EMAIL*,email);  
 editor.putString(*LOGIN\_PASS*,pass);  
 editor.apply();  
 }  
 public String getLOGIN(){  
 myPreferences = context.getSharedPreferences(*LOGIN\_NAME\_FILE*,*MODE\_PRIVATE*);  
 if(is\_login\_save()){  
 String e,p;  
 e= myPreferences.getString(*LOGIN\_EMAIL*,"");  
 p = myPreferences.getString(*LOGIN\_PASS*,"");  
 return e+";"+p;  
 }else{  
 return "EROOR";  
 }  
 }  
 public void Drop\_login() {  
 myPreferences = context.getSharedPreferences(*LOGIN\_NAME\_FILE*, *MODE\_PRIVATE*);  
 SharedPreferences.Editor editor = myPreferences.edit();  
 editor.clear();  
 editor.commit();  
 }  
}

клас “Function”

package com.yevhen.project;  
import android.app.Application;  
import android.content.Context;  
import android.net.ConnectivityManager;  
import android.net.DhcpInfo;  
import android.net.NetworkInfo;  
import android.net.wifi.WifiInfo;  
import android.net.wifi.WifiManager;  
import android.widget.Button;  
import android.widget.EditText;  
import java.math.BigInteger;  
import java.net.InetAddress;  
import java.net.UnknownHostException;  
  
public final class Function extends Application {  
 public boolean isOnline(Context context) {  
 ConnectivityManager cm =  
 (ConnectivityManager) context.getSystemService(context.*CONNECTIVITY\_SERVICE*);  
 NetworkInfo netInfo = cm.getActiveNetworkInfo();  
 return netInfo != null && netInfo.isConnectedOrConnecting();  
 }  
  
 public final static String getApIpAddr(Context context) {  
 WifiManager wifiManager = (WifiManager) context.getSystemService(Context.*WIFI\_SERVICE*);  
 DhcpInfo dhcpInfo = wifiManager.getDhcpInfo();  
 byte[] ipAddress = *convert2Bytes*(dhcpInfo.serverAddress);  
 try {  
 String apIpAddr = InetAddress.*getByAddress*(ipAddress).getHostAddress();  
 return apIpAddr;  
 } catch (UnknownHostException e) {  
 e.printStackTrace();  
 }  
 return null;}  
 public final static String getIp(Context context) {  
 WifiManager wifiManager = (WifiManager) context.getSystemService(Context.*WIFI\_SERVICE*);  
 WifiInfo wifiInfo = wifiManager.getConnectionInfo();  
 byte[] ipAddress = *convert2Bytes*(wifiInfo.getIpAddress());  
 try {  
 String apIpAddr = InetAddress.*getByAddress*(ipAddress).getHostAddress();  
 return apIpAddr;  
 } catch (UnknownHostException e) {  
 e.printStackTrace();  
 }  
 return null;  
 }  
 private static byte[] convert2Bytes(int hostAddress) {  
 byte[] addressBytes = { (byte)(0xff & hostAddress),  
 (byte)(0xff & (hostAddress >> 8)),  
 (byte)(0xff & (hostAddress >> 16)),  
 (byte)(0xff & (hostAddress >> 24)) };  
 return addressBytes;  
 }  
  
 private String getIPv4Address(int ipAddress) {  
 *// convert integer ip to a byte array* byte[] tempAddress = BigInteger.*valueOf*(ipAddress).toByteArray();  
 int size = tempAddress.length;  
 *// reverse the content of the byte array* for(int i = 0; i < size/2; i++) {  
 byte temp = tempAddress[size-1-i];  
 tempAddress[size-1-i] = tempAddress[i];  
 tempAddress[i] = temp;  
 }  
 try {  
 *// get the IPv4 formatted ip from the reversed byte array* InetAddress inetIP = InetAddress.*getByAddress*(tempAddress);  
 return inetIP.getHostAddress();  
 } catch (UnknownHostException e) {  
 e.printStackTrace();  
 }  
 return "";  
 }  
 static boolean isValue(String str, char ch){  
 for(int i=0;i<str.length();i++){  
 if(str.charAt(i)==ch) return true;  
 }  
 return false;  
 }  
 static void setEnabled\_button(Button button, Boolean bool){  
 if(button ==null) return;  
 else{  
 if(bool == true) {  
 button.setEnabled(true);  
 button.setAlpha(1f);  
 }else{  
 button.setEnabled(false);  
 button.setAlpha(0.6f);  
 }} }  
 static boolean isText(EditText editText){  
 if(editText==null) return false;  
 else{  
 if(editText.length()>0) return true;  
 }  
 return false;  
 }  
 static boolean notSpec(char a){  
 switch (a){  
 case '1':  
 case '2':  
 case '3':  
 case '4':  
 case '5':  
 case '6':  
 case '7':  
 case '8':  
 case '9':  
 case '0': return false;  
 }  
 return true;  
 }  
 static int toint(String a){  
 int zn = new Integer(a);  
 return zn;  
 }  
}

інтерфейс “JsonApi”

package com.yevhen.project;  
import com.google.android.gms.common.internal.safeparcel.SafeParcelable;  
import com.google.gson.JsonObject;  
import com.yevhen.project.Class.Email\_reg;  
import com.yevhen.project.Class.Users;  
import com.yevhen.project.Class.Users\_log;  
import com.yevhen.project.Class.Users\_reg;  
import retrofit2.Call;  
import retrofit2.http.Body;  
import retrofit2.http.Field;  
import retrofit2.http.GET;  
import retrofit2.http.Header;  
import retrofit2.http.Headers;  
import retrofit2.http.Multipart;  
import retrofit2.http.PATCH;  
import retrofit2.http.POST;  
import retrofit2.http.PUT;  
import retrofit2.http.Part;  
import retrofit2.http.Path;  
import retrofit2.http.Query;  
  
public interface JsonApi {  
 @POST("users/sendVerificationCode")  
 Call<Void> createVercode(@Body Email\_reg email);  
 @POST("users")  
 Call<Void> createUsers(@Body Users\_reg users);  
 @Headers("Content-Type: application/json")  
 @POST("users/auth")  
 Call<Users> getAuthUsers(@Body Users\_log users);  
 @POST("options")  
 Call<Void> setHub(@Query("a")String a,  
 @Query("b")String b,  
 @Query("c")String c,  
 @Query("d")String d,  
 @Query("e")String e);  
 *//@PUT* @GET("users/{id}/fcmToken")  
 Call<JsonObject> getToken(@Path("id") String id, @Query("access\_token") String access\_token);  
 @PUT("users/{id}/fcmToken")  
 Call<Void> putToken(@Path("id") String id, @Query("access\_token") String access\_token, @Query("fcmToken") JsonObject token);  
  
}