|  |  |
| --- | --- |
|  | **ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ**  **ФАКУЛТЕТ КОМПЮТЪРНИ СИСТЕМИ И ТЕХНОЛОГИИ** |

**Курсова работа**

**Дисциплина: „Програмиране за мобилни устройства”**

**тема: Мобилно приложение – Познай държавата**

***Изготвил****:*

Ахмед Исмаил Гаваз

Фак. № 121221145

Група: 44

III курс, КСИ

Факултет: КСТ

Дата: 26.03.2024г.

**София, 2024**

Съдържание

[1.Увод 3](#_Toc159763709)

[2.Анализ на съществуващи разработки 3](#_Toc159763710)

[3.Проектиране 6](#_Toc159763711)

[4.Реализация 7](#_Toc159763712)

[5. Потребителскo ръководство 19](#_Toc159763713)

[6.Заключение 22](#_Toc159763714)

[7.Литература 23](#_Toc159763715)

# 1.Увод

Във века на глобализацията и свръхбързите комуникации, светът става все по-малък и достъпен. Но докато технологичният прогрес ни свързва посредством електронни устройства и социални мрежи, знанието за географията и културата на различните държави остава от съществено значение.

Географията, като наука, ни предлага вълнуваща панорама от информация за земята, нейните ландшафти, климат, народи, и култури. Изучаването на географията ни дава възможност да разберем разнообразието и богатството на света, да се запознаем с различните географски региони и техните особености, както и да оценим връзката между хората и техните екологични, икономически и културни среди.

Географията е наука, която е навсякъде около нас: от района, в който живеем, до картите, които имаме на мобилните си устройства. Познаването на географията е много важно, както за общата ни култура, така и за опознаването на света ни като цяло. Колкото повече знаем за географията, толкова по-добре познаваме нашата планета.

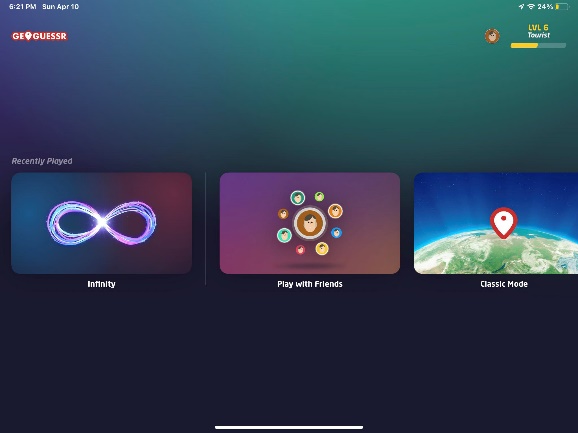
В този контекст реших да направя приложение за познаване на държавата, което се стреми да насърчи интереса и познанията по географията. Чрез иновативен подход и забавни игри, нашето приложение предоставя възможност за учене и развлечение, като потребителите се запознават с различни държави, техните флагове и столиците им. Нека заедно изследваме нашата планета, разгадавайки загадки, играейки игри и разглеждайки интересни факти за различните държави по света. Добре дошли в света на географията чрез нашето приложение за познаване на държавата!

# 2.Анализ на съществуващи разработки

Анализът на съществуващите разработки за приложения за познаване на държавите може да бъде извършен чрез преглед на различни приложения, които вече съществуват в магазините за приложения (като Google Play Store за Android или App Store за iOS). В следващите редове ще предоставя общ анализ, като ще се фокусирам на три различни приложения и ще изтъкна техните положителни и отрицателни страни:

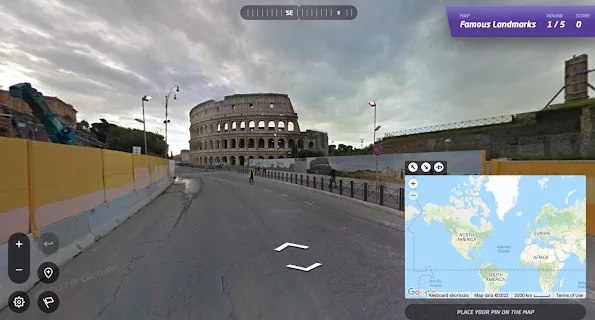
* GeoGuessr:
* Положителни страни:
  + Има интуитивен интерфейс, който е лесен за навигация от потребителите.
  + Предлага разнообразни режими на игра, включително използването на Google Street View за поставяне на потребителя на случайни места по целия свят.
  + Има активна общност от играчи и възможност за създаване на собствени предизвикателства и турнири.
* Отрицателни страни:
  + В някои случаи, използването на Google Street View може да доведе до появата на места с неподходящо съдържание или неприятни изненади за потребителите.
  + Наличието на различни режими на игра може да доведе до разреденост или загуба на фокус за някои потребители.

Изображения:



Фигура 1 Меню със скорошните модове на игра

Фигура 2 Режим на игра Street View



* World Geography Quiz Game:
* Положителни страни:
  + Предлага различни нива на трудност и разнообразни категории за игра, което прави приложението подходящо за играчи с различни нива на знания.
  + Има приятен дизайн и графика, които правят играта привлекателна за потребителите.
  + Включва режим за обучение, който позволява на потребителите да учат нови факти за различните държави.
* Отрицателни страни:
  + Някои отговори или въпроси могат да бъдат неправилно формулирани или недостатъчно ясни за потребителите, което може да доведе до объркване.
  + Може да липсва активност или възможност за състезание с други играчи, което може да намали интереса в дългосрочен план.

Изображения:



Фигура 5 Ранглиста на играчите

Фигура 3 Основна информация

Фигура 4 Режим познаване на картата

* Flags Quiz:
* Положителни страни:
  + Специализира се в разпознаването на флагове на държави, което прави приложението идеално за потребители, които са заинтересовани предимно от тази област.
  + Има прост интерфейс и лесни за разбиране правила на игра.
  + Предлага възможност за учене и подобряване на знанията за флаговете на различните държави.
* Отрицателни страни:
  + Ограничената област на интерес (флагове на държави) може да направи приложението еднообразно за потребителите, които търсят по-широко географско изживяване.
  + Може да липсва разнообразие в режимите на игра или възможности за състезание с други играчи.

Изобраения:



Фигура 8 Показване на държавата на картата

Фигура 6 Избор на ниво на трудност

Фигура 7 Въпрос и възможни отговори

Общият анализ на тези три приложения показва, че успешните приложения за познаване на държавите се отличават с интуитивен интерфейс, разнообразни режими на игра и възможност за обучение и подобряване на знанията. Важно е също така да се отбележи, че голяма част от потребителския опит зависи от избора на правилно обхващане на темата, интересен дизайн и възможности за взаимодействие с други играчи.

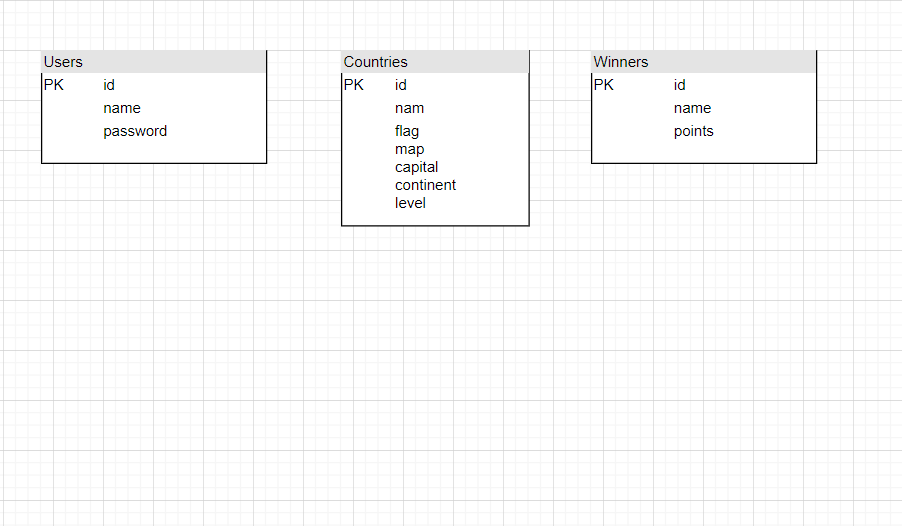
# 3.Проектиране

* Кой ще използва продукта?

Приложението е с образователна цел и иска да насърчи опознаването на държавите. Потенциалните потребители на приложението включват хора, които се интересуват от географията и искат да подобрят своите знания за държавите по света. Те могат да бъдат ученици, студенти, учители, любители на географията или просто хора, които искат да разширят своята културна осведоменост. Приложението е подходящо за всички възрастови групи.

* Какви данни ще се използват?

Приложението използва следните данни за държавите: наименованието им, знамената им, столиците им, формата им на картата и континента им. Изображенията на картите и формат на държавите се пазят в папка drawable и се зареждат от базата с данни. Освен данни за държавите се пазят и данни за потребителите и на победителите. Най-високите резултати се пазят локално на устройството на потребителя. Основните таблици в базата данни са:



Фигура 9 Основни таблици в базата данни

* Как ще бъдат достъпени функционалностите от потребителя?

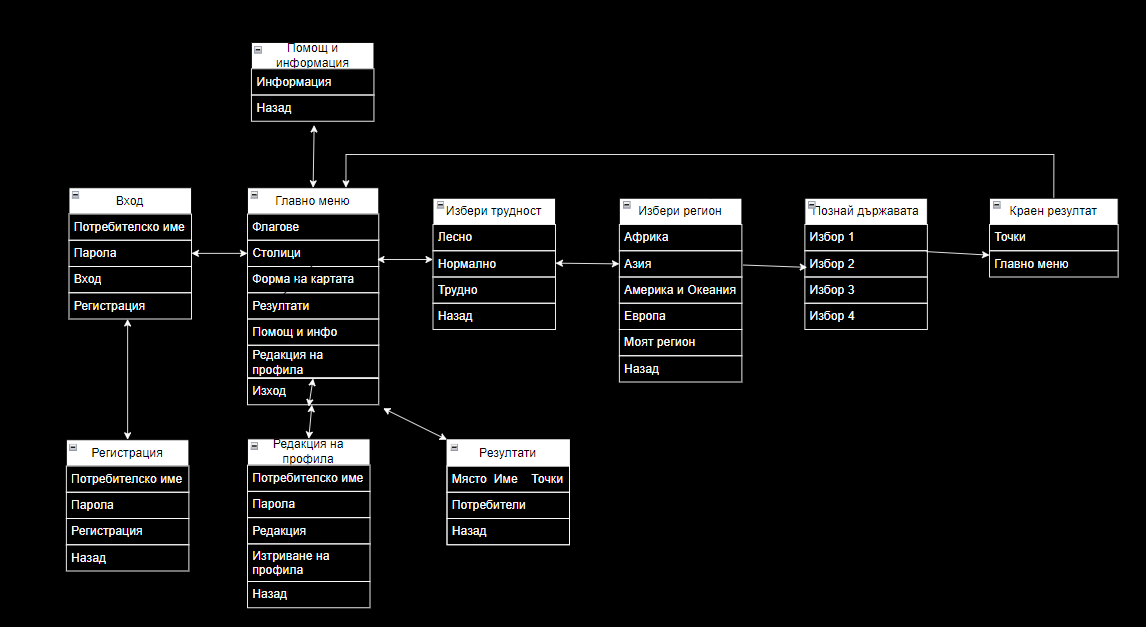
В началото потребителя трябва да влезе в профила си или да се регистрира ако няма профил. Основно тук потребителите трябва да познаят държавата по нейния флаг, нейната столица или формата й на картата. Списъците със столици и държави са взети от интернет и може да има разминаване с официално признатите държави или да има несъответствие между държава и столица. В приложението има над 150.

За създаване на приложението е използвана програмата Android Studio. За създаването на приложението са използвани основните му предимства, като например генериране на xml файлове при работа в Design Mode. Front-end частта е направена само с Android Studio и са използвани неговите бутони, шрифтове и помощни инструменти.

Знамената на държавите и формите им са поместени в папката drawable на проекта. Имената на държавите, столиците им и имената на файловете с флаговете на държавите се зареждат от базата данни. В класа TestScreen се намират основните методи за работа на изпитващата част на приложението.

Приложението има основно три режима: за познава на държава по флаг, столица или форма на картата. Основно държавите са разделени на четири географски региона: Азия, Африка, Европа и Америка и океания. Държавите също са разделени и по нивото на трудност, като за всяко ниво има средно по 15 държави. Основната структура на менюто е:

Фигура 10 Структура на менюто



# 4.Реализация

Приложението започва от формата за вход. При успешен вход се показва главното меню, където са разположени бутоните за избор на режима на игра, за резултатите, за помощ и информация, за редакция на профила и за изход. За всеки бутон е сложен метод, който указва какво да се случва при натискането му. Като при избор за режима на игра не само се сменя изгледа, но и се предава информация между менютата, за избрания режим, за нивото на трудно и за избрания регион. Реализацията на класа за главното меню и предаването на информация за избрания режим на игра:

public class MainMenu extends AppCompatActivity {

private Button flag, capital, help, map, results, exit, edit;

private String username;

private static final String TAG = "MainActivity";

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main\_menu);

Log.d(TAG, "onCreate: Started.");

Intent receiveMode = getIntent();

username = receiveMode.getStringExtra("User");

flag = findViewById(R.id.flag);

capital = findViewById(R.id.capital);

help = findViewById(R.id.help);

map = findViewById(R.id.map);

results = findViewById(R.id.results);

exit = findViewById(R.id.exit);

edit = findViewById(R.id.editProfile);

edit.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainMenu.this, EditProfile.class);

intent.putExtra("User", username);

startActivity(intent);

finish();

}

});

exit.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainMenu.this, Login.class);

startActivity(intent);

finish();

}

});

flag.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainMenu.this, Level.class);

intent.putExtra("Mode", "Flag");

intent.putExtra("User", username);

startActivity(intent);

finish();

}

});

capital.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainMenu.this, Level.class);

intent.putExtra("Mode", "Capital");

intent.putExtra("User", username);

startActivity(intent);

finish();

}

});

help.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainMenu.this, HelpScreen.class);

startActivity(intent);

finish();

}

});

map.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainMenu.this, Level.class);

intent.putExtra("Mode", "Map");

intent.putExtra("User", username);

startActivity(intent);

finish();

}

});

results.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainMenu.this, Result.class);

intent.putExtra("User", username);

startActivity(intent);

finish();

}

});

}

@Override

protected void onPause() {

super.onPause();

MyApplication.mediaPlayer.pause();

}

@Override

protected void onResume() {

super.onResume();

MyApplication.mediaPlayer.start();

}

}

След като сме предали режима на игра, ние трябва да го получим в следващото меню. Ето пример за получаването на режима на игра в менюто за нивото на трудност и предаването на няколко параметъра(режим и ниво на трудност) в менюто за трудност:

public class Level extends AppCompatActivity {

private static final String TAG = "SecondScreen";

private Button easy, medium, hard, mainMenu;

private String username;

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.level);

Log.d(TAG, "onCreate: Started.");

easy = findViewById(R.id.easy);

medium = findViewById(R.id.medium);

hard = findViewById(R.id.hard);

mainMenu = findViewById(R.id.backToMainMenu);

Intent receiveMode = getIntent();

String mode = receiveMode.getStringExtra("Mode");

username = receiveMode.getStringExtra("User");

easy.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(Level.this, Region.class);

intent.putExtra("Mode", mode);

intent.putExtra("User", username);

intent.putExtra("Level", "Easy");

startActivity(intent);

finish();

}

});

medium.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(Level.this, Region.class);

intent.putExtra("Mode", mode);

intent.putExtra("User", username);

intent.putExtra("Level", "Medium");

startActivity(intent);

finish();

}

});

hard.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(Level.this, Region.class);

intent.putExtra("Mode", mode);

intent.putExtra("User", username);

intent.putExtra("Level", "Hard");

startActivity(intent);

finish();

}

});

mainMenu.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(Level.this, MainMenu.class);

intent.putExtra("User", username);

startActivity(intent);

finish();

}

});

}

@Override

protected void onPause() {

super.onPause();

MyApplication.mediaPlayer.pause();

}

@Override

protected void onResume() {

super.onResume();

MyApplication.mediaPlayer.start();

}

}

По този начин предаваме информация между различните менюта(виж структурата на страница 7). Друго важно нещо в програмата са класовете за основната информация. В моя проект има три основни класа за прехвърляне на информация между базата данни и екраните. Това са класа за потребители, за държава и за победители.

Клас за потребители:

public class User {

private int id;

private String username;

private String password;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public User() {

}

public User(String Username, String \_pass) {

this.username = Username;

this.password = \_pass;

}

public int getID() {

return this.id;

}

public void setID(int id) {

this.id = id;

}

public String getUserName() {

return this.username;

}

public void setUserName(String name) {

this.username = name;

}

public String getPassword() {

return this.password;

}

public void setPassword(String \_pass) {

this.password = \_pass;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

}

Клас за държави:

public class Country {

private String name;

private String flag;

private String map;

private String capital;

private String continent;

private String level;

public String getContinent() {

return continent;

}

public Country(String name, String flag, String map, String capital, String continent, String level) {

this.name = name;

this.flag = flag;

this.map = map;

this.capital = capital;

this.continent = continent;

this.level = level;

}

public Country() {

}

public void setContinent(String continent) {

this.continent = continent;

}

public String getLevel() {

return level;

}

public void setLevel(String level) {

this.level = level;

}

public String getMap() {

return map;

}

public void setMap(String map) {

this.map = map;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getFlag() {

return flag;

}

public void setFlag(String flag) {

this.flag = flag;

}

public String getCapital() {

return capital;

}

public void setCapital(String capital) {

this.capital = capital;

}

}

Клас за победители:

public class Winner {

private int id;

private String username;

private int points;

public Winner(String username, int points) {

this.username = username;

this.points = points;

}

public Winner() {

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public int getPoints() {

return points;

}

public void setPoints(int points) {

this.points = points;

}

}

Друго важно нещо е класа играещ роля на база данни. Чрез него зареждаме информацията за държавите и потребителите, както и записваме най-високите резултати. Държавите се зареждат от конфигурационен файл в папката raw. Трябва да се обърне внимание, че при зареждането държавите се разделят по трудност и регион. Ето класа:

public class Database extends SQLiteOpenHelper {

private static final int DATABASE\_VERSION = 1;

private static final String DATABASE\_NAME = "USERMANAGER";

private static final String USERS = "USERS";

private static final String WINNERS = "WINNERS";

private static final String KEY\_ID = "id";

private static final String COUNTRIES = "COUNTRIES";

private static final String KEY\_CONTINENT = "continent";

private static final String KEY\_LEVEL = "level";

private static final String KEY\_FLAG = "flag";

private static final String KEY\_MAP = "map";

private static final String KEY\_CAPITAL = "capital";

private static final String KEY\_NAME = "username";

private static final String KEY\_PASSWORD = "password";

private static final String KEY\_POINTS = "points";

private Context context;

public Database(Context context) {

super(context, DATABASE\_NAME, null, DATABASE\_VERSION);

this.context=context;

}

@Override

public void onCreate(SQLiteDatabase db) {

String CREATE\_USERS\_TABLE = "CREATE TABLE " + USERS +

"("

+ KEY\_ID + " INTEGER PRIMARY KEY," + KEY\_NAME + " TEXT,"

+ KEY\_PASSWORD + " TEXT" + ")";

db.execSQL(CREATE\_USERS\_TABLE);

String CREATE\_WINNERS\_TABLE = "CREATE TABLE " + WINNERS +

"("

+ KEY\_ID + " INTEGER PRIMARY KEY," + KEY\_NAME + " TEXT,"

+ KEY\_POINTS + " INTEGER" + ")";

db.execSQL(CREATE\_WINNERS\_TABLE);

String CREATE\_COUNTRIES\_TABLE = "CREATE TABLE " + COUNTRIES +

"("

+ KEY\_ID + " INTEGER PRIMARY KEY," + KEY\_NAME + " TEXT,"

+ KEY\_FLAG + " TEXT," + KEY\_MAP + " TEXT," + KEY\_CAPITAL + " TEXT,"

+ KEY\_CONTINENT + " TEXT," + KEY\_LEVEL + " TEXT" + ")";

db.execSQL(CREATE\_COUNTRIES\_TABLE);

// След като създадем таблиците, можем да добавим държавите

addCountries(context);

}

public void updateUser(String oldUsername, String newUsername, String newPassword) {

SQLiteDatabase db = this.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(KEY\_NAME, newUsername);

db.update(USERS, values, KEY\_NAME + " = ?", new String[]{oldUsername});

values.clear();

values.put(KEY\_PASSWORD, newPassword);

db.update(USERS, values, KEY\_NAME + " = ?", new String[]{newUsername});

db.close();

}

public void deleteUserByUsername(String username) {

SQLiteDatabase db = this.getWritableDatabase();

db.delete(USERS, KEY\_NAME + " = ?", new String[]{username});

db.close();

}

public void addCountries(Context context) {

SQLiteDatabase db = this.getWritableDatabase();

try {

InputStream inputStream = context.getResources().openRawResource(R.raw.config);

BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));

StringBuilder stringBuilder = new StringBuilder();

String line;

while ((line = reader.readLine()) != null) {

stringBuilder.append(line);

}

reader.close();

String json = stringBuilder.toString();

JSONArray countriesArray = new JSONArray(json);

for (int i = 0; i < countriesArray.length(); i++) {

JSONObject countryObject = countriesArray.getJSONObject(i);

String name = countryObject.getString("name");

String flag = countryObject.getString("flag");

String map = countryObject.getString("map");

String capital = countryObject.getString("capital");

String continent = countryObject.getString("continent");

String level = countryObject.getString("level");

ContentValues values = new ContentValues();

values.put(KEY\_NAME, name);

values.put(KEY\_FLAG, flag);

values.put(KEY\_MAP, map);

values.put(KEY\_CAPITAL, capital);

values.put(KEY\_CONTINENT, continent);

values.put(KEY\_LEVEL, level);

db.insert(COUNTRIES, null, values);

}

} catch (IOException | JSONException e) {

e.printStackTrace();

Log.e("Database", "An error occurred while adding countries", e);

} finally {

}

}

public List<Country> getCountriesByFlag(String flag) {

List<Country> countryList = new ArrayList<>();

SQLiteDatabase db = this.getWritableDatabase();

Cursor cursor = db.query(COUNTRIES, null, KEY\_FLAG + "=?",

new String[]{flag}, null, null, null);

if (cursor.moveToFirst()) {

do {

Country country = new Country();

country.setName(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_NAME)));

country.setFlag(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_FLAG)));

country.setMap(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_MAP)));

country.setCapital(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_CAPITAL)));

country.setContinent(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_CONTINENT)));

country.setLevel(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_LEVEL)));

countryList.add(country);

} while (cursor.moveToNext());

}

cursor.close();

db.close();

return countryList;

}

public User getUserByUsername(String username) {

SQLiteDatabase db = this.getReadableDatabase();

User user = null;

Cursor cursor = db.query(USERS, new String[]{KEY\_ID, KEY\_NAME, KEY\_PASSWORD},

KEY\_NAME + "=?", new String[]{username}, null, null, null);

if (cursor != null) {

cursor.moveToFirst();

if (cursor.getCount() > 0) {

user = new User();

user.setId(cursor.getInt(cursor.getColumnIndex(KEY\_ID)));

user.setUserName(cursor.getString(cursor.getColumnIndex(KEY\_NAME)));

user.setPassword(cursor.getString(cursor.getColumnIndex(KEY\_PASSWORD)));

}

cursor.close();

}

return user;

}

public List<Country> getCountriesByDifficultyAndContinent(String level, String continent) {

List<Country> countryList = new ArrayList<>();

SQLiteDatabase db = this.getWritableDatabase();

Cursor cursor = db.query(COUNTRIES, null, KEY\_LEVEL + "=? AND " + KEY\_CONTINENT + "=?",

new String[]{level, continent}, null, null, null);

if (cursor.moveToFirst()) {

do {

Country country = new Country();

country.setName(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_NAME)));

country.setFlag(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_FLAG)));

country.setMap(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_MAP)));

country.setCapital(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_CAPITAL)));

country.setContinent(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_CONTINENT)));

country.setLevel(cursor.getString(cursor.getColumnIndexOrThrow(KEY\_LEVEL)));

countryList.add(country);

} while (cursor.moveToNext());

}

cursor.close();

db.close();

return countryList;

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

db.execSQL("DROP TABLE IF EXISTS " + USERS);

db.execSQL("DROP TABLE IF EXISTS " + WINNERS);

db.execSQL("DROP TABLE IF EXISTS " + COUNTRIES);

onCreate(db);

}

public void onDeleteTable() {

SQLiteDatabase db = this.getWritableDatabase();

db.execSQL("DROP TABLE IF EXISTS " + USERS);

db.execSQL("DROP TABLE IF EXISTS " + WINNERS);

db.execSQL("DROP TABLE IF EXISTS " + COUNTRIES);

db.close();

}

public void addUser(User user) {

SQLiteDatabase db = this.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(KEY\_NAME, user.getUsername());

values.put(KEY\_PASSWORD, user.getPassword());

db.insert(USERS, null, values);

db.close();

}

public List<User> getAllUsers() {

List<User> userList = new ArrayList<>();

String selectQuery = "SELECT \* FROM " + USERS;

SQLiteDatabase db = this.getWritableDatabase();

Cursor cursor = db.rawQuery(selectQuery, null);

if (cursor.moveToFirst()) {

do {

User user = new User();

user.setID(Integer.parseInt(cursor.getString(0)));

user.setUserName(cursor.getString(1));

user.setPassword(cursor.getString(2));

userList.add(user);

} while (cursor.moveToNext());

}

return userList;

}

public boolean isUserExists(String username) {

SQLiteDatabase db = this.getReadableDatabase();

String query = "SELECT \* FROM " + USERS + " WHERE " + KEY\_NAME + "=?";

Cursor cursor = db.rawQuery(query, new String[]{username});

boolean exists = cursor.getCount() > 0;

cursor.close();

return exists;

}

public void addWinner(Winner winner) {

SQLiteDatabase db = this.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(KEY\_NAME, winner.getUsername());

values.put(KEY\_POINTS, winner.getPoints());

db.insert(WINNERS, null, values);

db.close();

}

public List<Winner> getAllWinners() {

List<Winner> winnersList = new ArrayList<>();

String selectQuery = "SELECT \* FROM " + WINNERS;

SQLiteDatabase db = this.getWritableDatabase();

Cursor cursor = db.rawQuery(selectQuery, null);

if (cursor.moveToFirst()) {

do {

Winner winner = new Winner();

winner.setId(Integer.parseInt(cursor.getString(0)));

winner.setUsername(cursor.getString(1));

winner.setPoints(Integer.parseInt(cursor.getString(2)));

winnersList.add(winner);

} while (cursor.moveToNext());

}

return winnersList;

}

}

Друг интересен момент, след като сме зареди държавите за съответния режим и регион, е алгоритъма за визуализация и избор на случайна държава, както и за зареждане на случайни държави в бутоните за възможни отговори. Следния алгоритъм се намира в метода setQuestionAndOptionsCapitals в класа TestScreen, който отговаря за същинската игра:

private void setQuestionAndOptionsCapitals(List<Country> countries, String mode, int pointsForMode) {

do {

Random random = new Random();

int index = random.nextInt(countries.size());

correctAnswer = countries.get(index);

answeredCountries.add(correctAnswer.getName());

if (mode.equals("Capital")) {

question.setText(countries.get(index).getCapital() + " е столица на коя държава?");

} else if (mode.equals("Flag")) {

question.setText("На коя държава принадлежи този флаг?");

ImageView img = findViewById(R.id.imageView5);

String drawableName = countries.get(index).getFlag();

int resourceId = getResources().getIdentifier(drawableName, "drawable", getPackageName());

img.setImageResource(resourceId);

} else if (mode.equals("Map")) {

question.setText("Коя държава има следната форма?");

ImageView img = findViewById(R.id.imageView5);

String drawableName = countries.get(index).getMap();

int resourceId = getResources().getIdentifier(drawableName, "drawable", getPackageName());

img.setImageResource(resourceId);

}

} while (askedCountries.contains(correctAnswer.getName()));

final int[] receivedPoints = {pointsForMode};

askedCountries.add(correctAnswer.getName());

List<Button> buttons = new ArrayList<>();

buttons.add(findViewById(R.id.first));

buttons.add(findViewById(R.id.second));

buttons.add(findViewById(R.id.third));

buttons.add(findViewById(R.id.fourth));

Collections.shuffle(buttons);

Button correctButton = buttons.get(0);

correctButton.setText(correctAnswer.getName());

List<String> options = new ArrayList<>();

for (Country country : countries) {

if (!country.equals(correctAnswer) && !answeredCountries.contains(country.getName())) {

options.add(country.getName());

}

}

Collections.shuffle(options);

for (int i = 1; i < 4; i++) {

try{

buttons.get(i).setText(options.get(i-1));

}catch (Exception e){

System.out.println();

}

}

View.OnClickListener answerClickListener = new View.OnClickListener() {

@Override

public void onClick(View v) {

Button clickedButton = (Button) v;

String selectedAnswer = clickedButton.getText().toString();

if (!selectedAnswer.equals(correctAnswer.getName()) && !wrongChoices.contains(clickedButton)) {

clickedButton.setBackgroundColor(Color.RED);

clickedButton.setTextColor(Color.RED);

errors++;

wrongChoices.add(clickedButton);

if (errors < 3) {

receivedPoints[0] = receivedPoints[0] / 2;

} else {

receivedPoints[0] = 0;

}

} else if (selectedAnswer.equals(correctAnswer.getName())) {

clickedButton.setBackgroundColor(Color.GREEN);

Intent intent = new Intent(TestScreen.this, TestScreen.class);

intent.putExtra("Mode", mode);

intent.putExtra("Level", level);

intent.putExtra("User", username);

intent.putExtra("Area", area);

questionCount++;

points += receivedPoints[0];

errors = 0;

startActivity(intent);

finish();

}

}

};

for (Button button : buttons) {

button.setOnClickListener(answerClickListener);

}

}

Освен тези методи имаме и клас за записване във json файл, локално на устройството на 10-те най-добре справили се потребителя. Ето го класа за запис и четене от този файл:

public class FileIO {

public static void writeUsersToFile(final Context context, final List<Winner> userList) {

try {

File path = context.getExternalFilesDir(null); // Get app-specific directory

File file = new File(path, "winners.json");

JSONArray jsonArray = new JSONArray();

for (Winner winner : userList) {

JSONObject jsonObject = new JSONObject();

jsonObject.put("username", winner.getUsername());

jsonObject.put("points", winner.getPoints());

jsonArray.put(jsonObject);

}

FileWriter writer = new FileWriter(file);

writer.write(jsonArray.toString());

writer.close();

} catch (JSONException | IOException e) {

System.err.println("Error writing users to JSON file: " + e.getMessage());

}

}

public static List<Winner> readUsersFromFile(Context context) {

List<Winner> winners = new ArrayList<>();

if (ContextCompat.checkSelfPermission(context, Manifest.permission.READ\_EXTERNAL\_STORAGE)

!= PackageManager.PERMISSION\_GRANTED) {

// Handle permission denied

}

try {

File file = new File(context.getExternalFilesDir(null), "winners.json");

if (file.exists()) {

BufferedReader reader = new BufferedReader(new FileReader(file));

StringBuilder json = new StringBuilder();

String line;

while ((line = reader.readLine()) != null) {

json.append(line);

}

reader.close();

JSONArray jsonArray = new JSONArray(json.toString());

for (int i = 0; i < jsonArray.length(); i++) {

JSONObject jsonObject = jsonArray.getJSONObject(i);

String username = jsonObject.getString("username");

int points = jsonObject.getInt("points");

winners.add(new Winner(username, points));

}

}

} catch (JSONException | IOException e) {

System.err.println("Error reading users from JSON file: " + e.getMessage());

}

return winners;

}

}

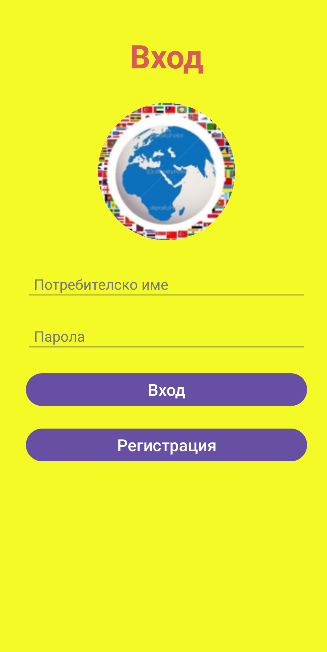
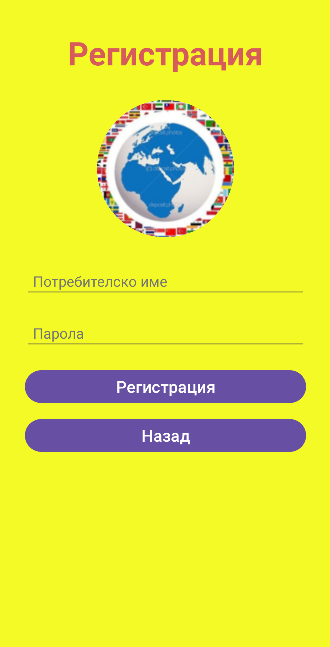
# 5. Потребителскo ръководство

При стартиране на приложението се отваря менюто за вход. Ако потребител няма регистрация, може да си направи с натискане на бутона за регистрация, при което потребителя трябва да въведе своето потребителско име и парола. При успешен вход на потребителя се показва главното меню на приложението. В него има седем бутона: три за избор на режима на игра, един за най-високите резултати, един за информация за приложението, един за редакция на профила и един за изход.

При избор на бутона за най-високи резултати се извежда списък с имената и точките на играчите с най-високи изкарани точки. При избор на бутона за информация за приложението се извежда текста основна информация за приложението, целта на приложението и информация за това как се използва приложението. При избор на бутона за редакция на профила се извеждат полета с потребителското име и паролата на потребителя, които той би желал да промени. Трябва да се обърне внимание, че при промяна на потребителското име, то трябва да бъде такова, че да няма друг потребител с такова име. Също от това меню може да се изтрие и профила на потребителя. Освен това има и бутони за връщане към главното меню при менютата за резултатите, информацията и редакцията на профила. При избор на опцията за режим на игра: флаг, столица или форма на картата, се извежда друго меню с нивото на трудност. В менюто има бутони за избор на нивото на трудност (лесно, средно и трудно), както и бутон за връщане към главното меню. При избор на нивото на трудност, се извежда и друго меню с избор за съответния регион на държавите (Африка, Азия, Европа, Америка и Океания и Моят регион). При избор на опцията Моят регион потребителя трябва да даде съгласието си за използване на местоположението му, защото тази опция използва местоположението на телефона и на база местоположението се определя региона за игра. След избора на регион започва играта: Потребителя ще бъде попитан 8 въпроса в зависимост от режима на игра, който е избрал. При задаване на въпроса се избира на случаен принцип държавата, за която трябва да се познае флага или столицата. И осемте въпроса са за различни държави. При грешен отговор, натиснатия бутон става червен. При избор на правилния отговор, натиснатия бутон става зелен и се преминава към следващия въпрос. След отговарянето на последния въпрос, се извеждат и спячелените точки от потребителя. Трябва да се обърне, че всяко ниво на трудност и всеки режим на играта имат различни точки. Те са както следва:

* Флагове:
  + Лесно – 100 точки
  + Средно – 200 точки
  + Трудно – 300 точки
* Столици:
  + Лесно – 120 точки
  + Средно – 240 точки
  + Трудно – 360 точки
* Форма на картата на страната:
  + Лесно – 160 точки
  + Средно – 320 точки
  + Трудно - 480 точки

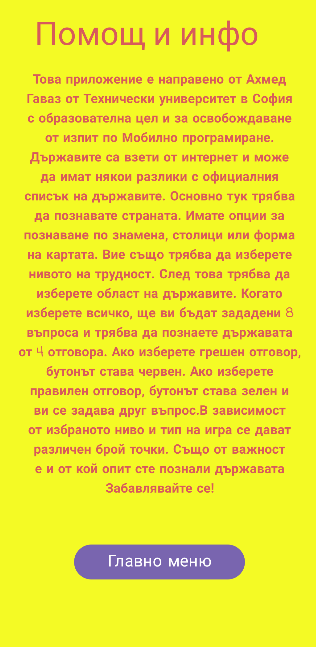
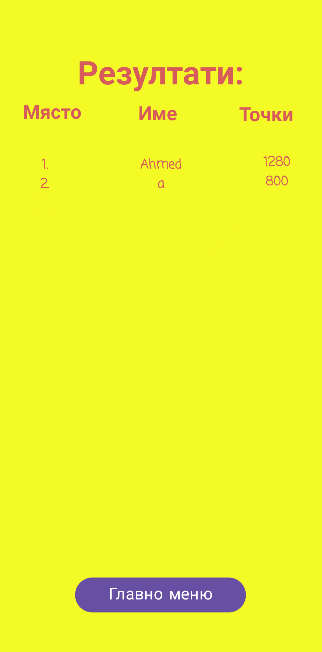
Също се взима в предвид и от кой опит е позната държавата. Ако се отговори от първия път се вземат 100% от точките, ако се познае от втория път се дават 50% от точките, ако се познае от третия път се дават 25% от точките и ако се познае от четвъртия опит не се дават точки. След играта се извеждат точките, спечелени от потребителя и след това се извежда отново главното меню. Ето изображение на менютата:



Фигура 13 Главно меню

Фигура 11 Меню за вход

Фигура 12 Меню за регистрация



Фигура 14 Меню за редакция на профила

Фигура 16 Резултати

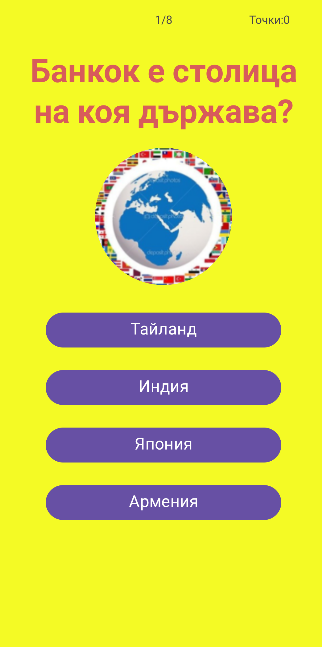
Фигура 15 Помощ и информация



Фигура 19 Познаване на държавата по флаг

Фигура 18 Меню за избор на трудност

Фигура 17 Меню за избор на регион



# 

Фигура 22 Край на играта

Фигура 21 Познаване на държавата по форма на картата

Фигура 20 Познаване на държавата по столица

# 6. Заключение

Приложението за познаване с държавите е създадено с цел да предложи на потребителите интересен и образователен начин за запознаване с географията на света. Чрез иновативен подход и разнообразни режими на игра, приложението предоставя възможност за учене и развлечение, като потребителите имат възможност да се запознаят с различни държави, техните флагове, столици и форми на картите им.

Анализът на съществуващите разработки показа важните аспекти за успешното функциониране на подобно приложение, включително интуитивния интерфейс, разнообразните режими на игра и възможностите за обучение. Приложението е проектирано с оглед на тези фактори, като е взето предвид разнообразието от потребители - от ученици до любители на географията.

Създаденото потребителско ръководство предоставя ясни инструкции за използване на приложението, като подчертава функционалностите и начините за достъп до тях. Потребителите могат лесно да се ориентират в приложението и да се насладят на играта, докато разширяват своите знания за държавите по света.

В крайна сметка, приложението за познаване на държавите представлява успешно съчетание между забавление и образование, което допринася за повишаване на географската осведоменост и интереса към различните култури и региони на света.

# 7.Литература

* <https://developer.android.com>
* <https://simple.wikipedia.org/wiki/List_of_countries_by_continents>
* <https://github.com/hampusborgos/country-flags>
* <https://play.google.com/store/apps/details?id=com.geoguessr.app&hl=en_US>
* <https://play.google.com/store/apps/details?id=com.age.wgg.appspot&hl=en_US>
* https://play.google.com/store/apps/details?id=guess.country.flag&hl=en\_US
* <https://mdl.fcst.tu-sofia.bg/course/view.php?id=105>
* Линк към проекта в github: https://github.com/ahmedgavaz/Android-app