Dokumentacja projektu Mateusz Siwiec, Grzegorz Jagodziński

Opis projektu

Aplikacja ma na celu zapisywanie uczestników do jazdy na torach gokartowych. Możemy wybrać trudność toru, liczbę uczestników, czas startu oraz czas jazdy. Cena za wynajem będzie liczona na zasadzie liczba uczestników x czas trwania. Dla przykładu : dla 5 uczestników, tor na 20 minut będzie kosztował 100zł.

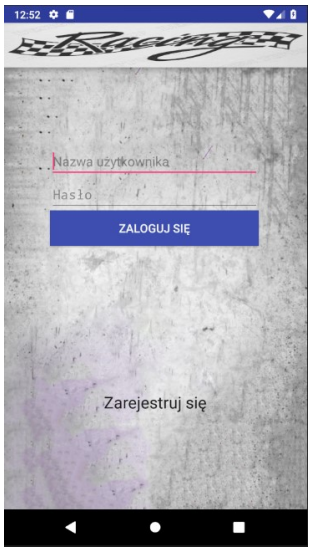
Zasada działania aplikacji

Do aplikacji jest podłączona zewnętrzna baza danych, która postawiona jest na darmowym hostingu (000webhost.com) i tam trzymamy pliki php, które po odwołaniu z kodu naszej aplikacji przekazują dane do bazy danych. Jest to komunikacja za pomocą metod POST i GET.

Na pierwszym etapie aplikacji mamy możliwość rejestracji. Po zarejestrowaniu możemy się już zalogować do aplikacji. Po zalogowaniu wyskakuje nam okienko. W tym okienku mamy napis przywitania oraz dwa przycisk. Jeden obsługuje dodanie nowej rezerwacji, a drugi obsługuje wyświetlanie obecnych rezerwacji użytkownika. Po wyświetleniu obecnych rezerwacji możemy kliknąć na konkretną i wtedy albo ją usunąć, albo edytować.

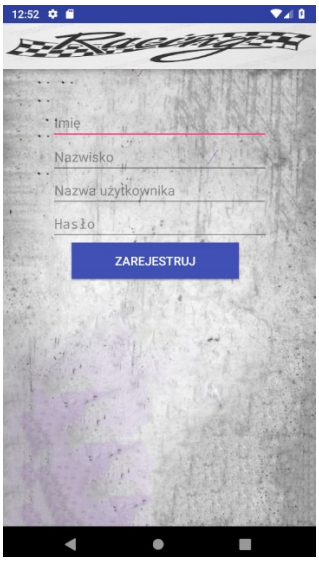
Opis poszczególnych ekranów w aplikacji

Ekran logowania

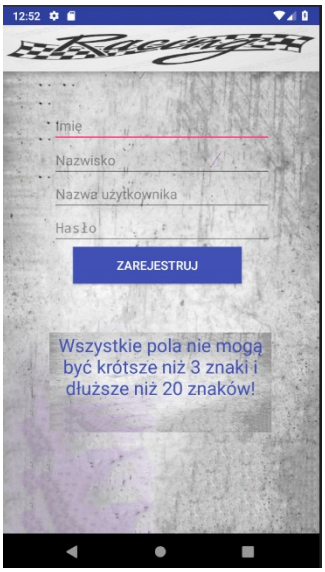


Jest to ekran na którym mamy możliwość zalogowania się poprzez wpisanie nazwy użytkownika oraz hasła. Poniżej również mamy przycisk do rejestracji, który przenosi nas na odpowiedni ekran rejestracji.

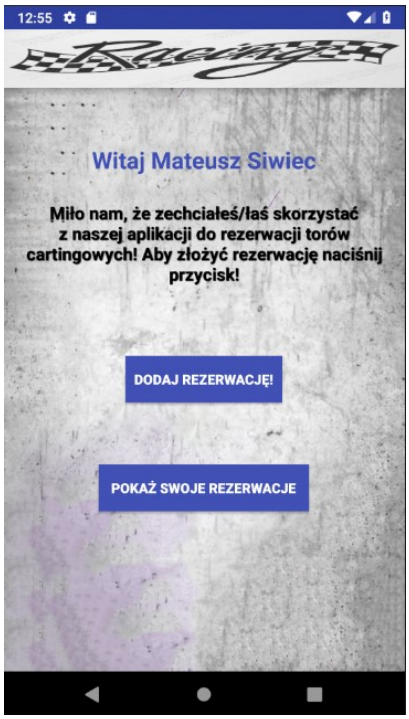
Ekran rejestracji



Na tym ekranie mamy możliwość zarejestrowania się do naszej aplikacji. Musimy podać imię, nazwisko, nazwe użytkownika oraz hasło. Wszystkie pola musza być dłuższe niż 3 znaki bo inaczej pojawi się błąd rejestracji.



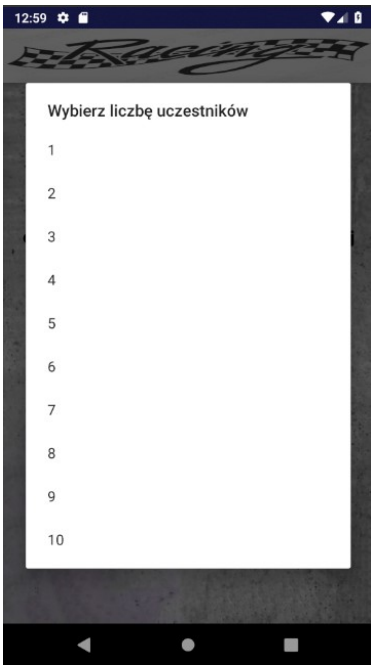
Po pomyślnym zalogowaniu pojawi się nam ekran powitalny w naszej aplikacji. Zawiera on tekst na powitanie oraz dwa przyciski. Pierwszy z nich umożliwa nam dodanie nowej rezerwacji, a drugi umożliwia podglądnięcie wszystkich rezerwacji użytkownika.

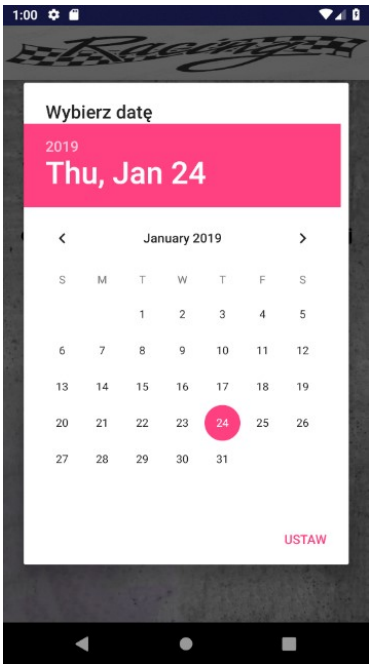


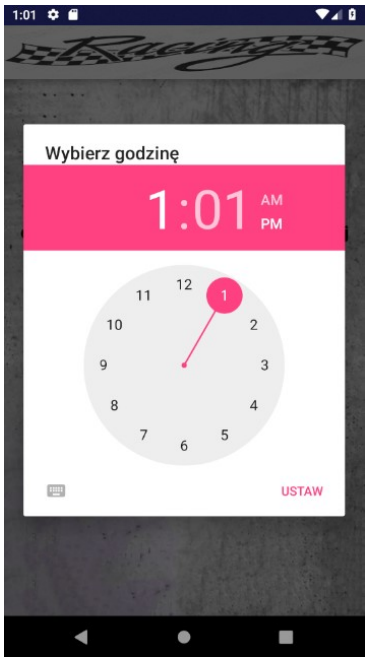
Tworzenie rezerwacji:

Po naciśnięciu przycisku „Dodaj rezerwację” pokazuje nam się okno dialogowe z informacją jakie dane będziemy musieli wypełnić, aby stworzyć rezerwację . Następnym krokiem jest wybranie trudności toru.

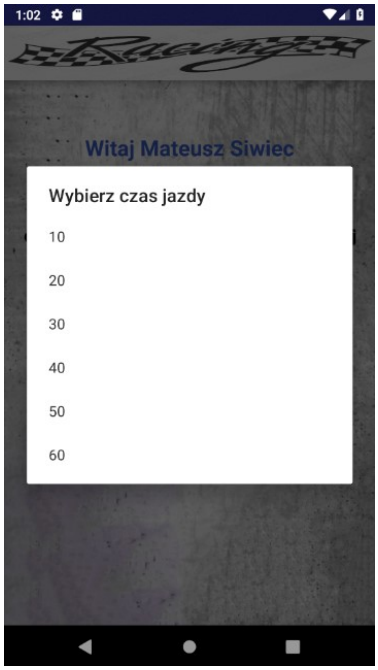
Kolejnym krokiem będzie wybranie liczby uczestników, następnie daty rezerwacji.





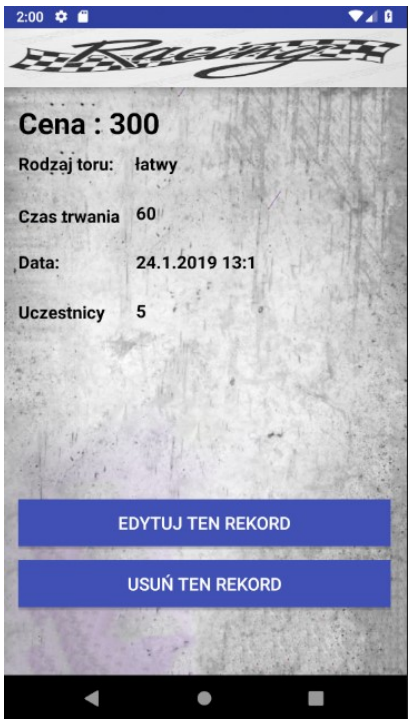


Ostatnią informacją jaką trzeba wypełnić będzie czas trwania jazdy.

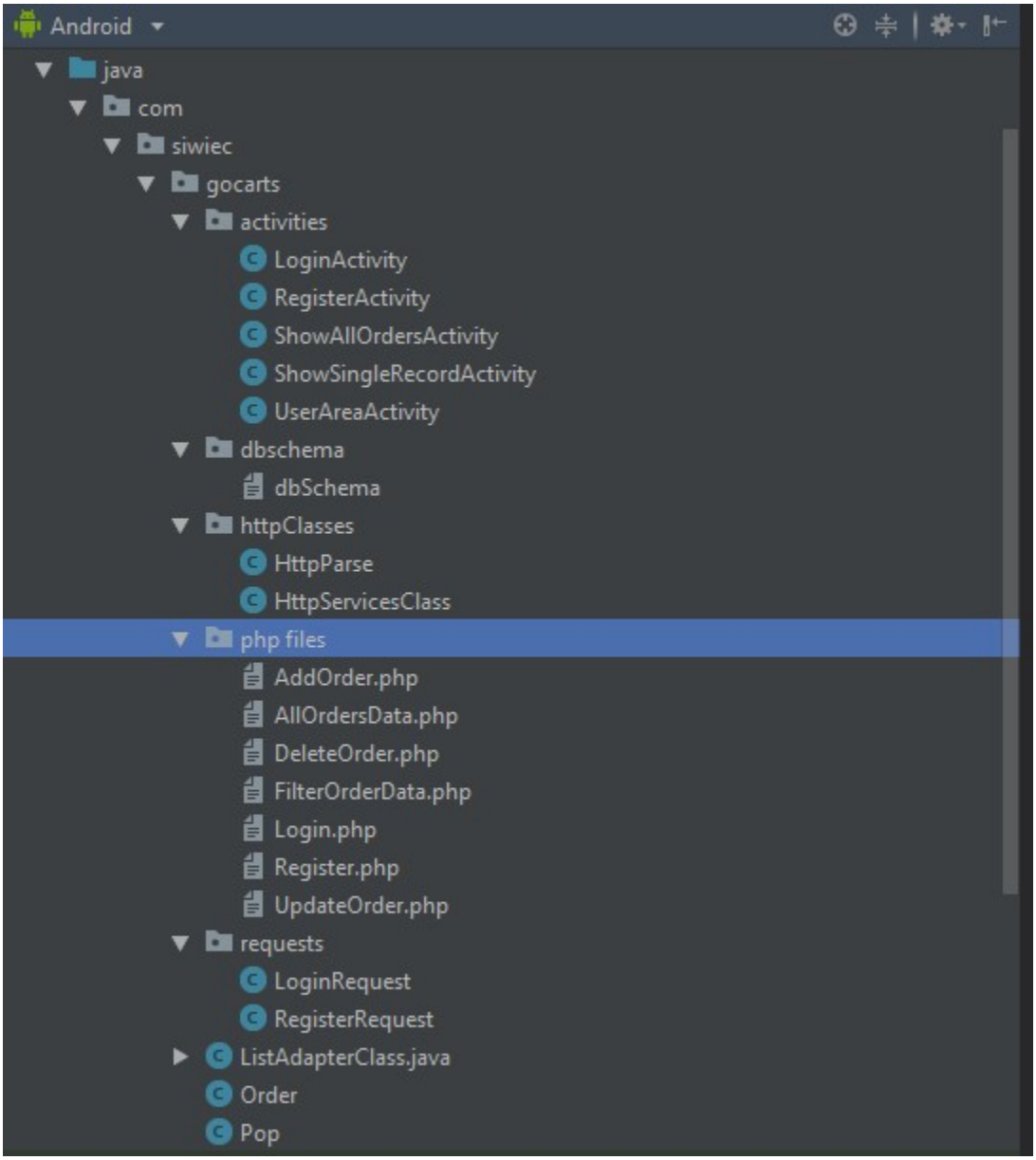


Po stworzeniu rezerwacji wracamy na ekran powitalny i możemy podglądnąć już nasze rezerwacje.

Teraz możemy kliknąć naszą rezerwację i wtedy pojawią się szczegółowe informacje na temat poszczególnej rezerwacji oraz dwie nowe opcje. Edytowanie obecnej rezerwacji oraz usunięcie jej.



Opis struktury plików w android studio

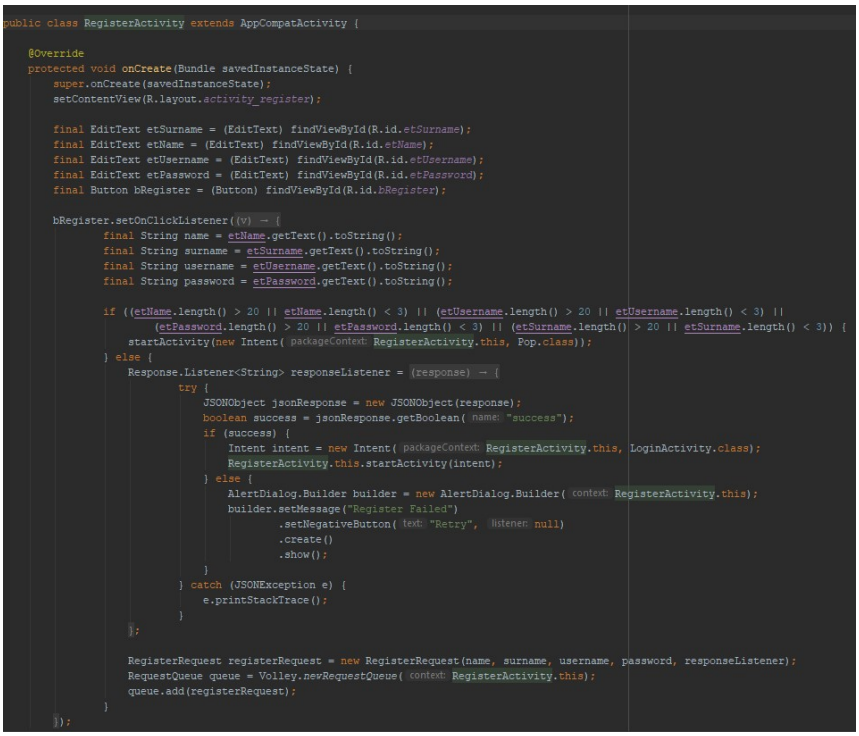


LoginActivity, LoginRequest – LoginActivity obsługuje proces logowania, przekazuje dane z logowania do LoginRequest po czym dostaje informację zwrotną czy udało się zalogować i jeżeli się udało zalogować przekazywane jest także id użytkownika.

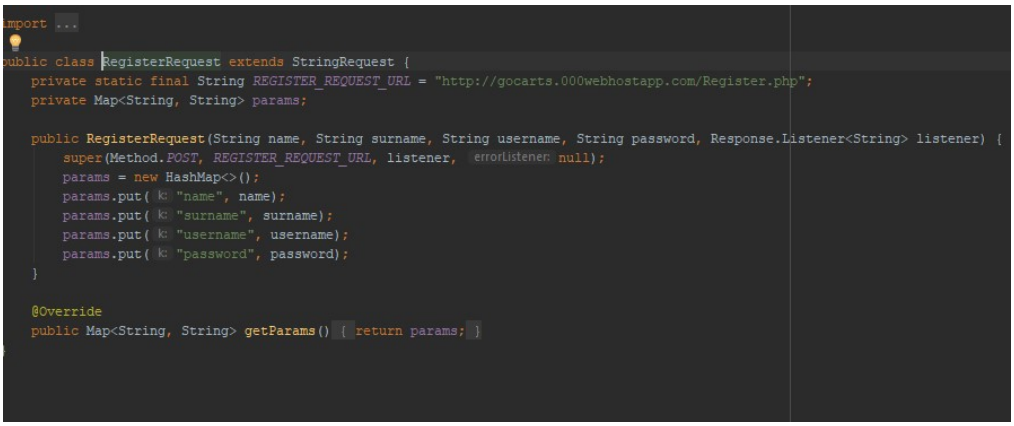
```
public class LoginActivity extends AppCompatActivity {
    public static String SHARED_PREFS = "sharedPrefs";
    public static String USER_ID = "sharedPrefs";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        final EditText etUsername = (EditText) findViewById(R.id.etUsername);
        final EditText etPassword = (EditText) findViewById(R.id.etPassword);
        final TextView tvRegisterLink = (TextView) findViewById(R.id.tvRegisterLink);
        final Button bLogin = (Button) findViewById(R.id.bSignIn);

        tvRegisterLink.setOnClickListener((v) → {
                Intent registerIntent = new Intent( packageContext: LoginActivity.this, RegisterActivity.class);
                LoginActivity.this.startActivity(registerIntent);
        });

        bLogin.setOnClickListener((v) → {
                final String username = etUsername.getText().toString();
                final String password = etPassword.getText().toString();

                // Response received from the server
                Response.Listener<String> responseListener = (response) → {
                        try {
                            JSONObject jsonResponse = new JSONObject(response);
                            boolean success = jsonResponse.getBoolean( name: "success");

                            if (success) {
                                String name = jsonResponse.getString( name: "name");
                                String surname = jsonResponse.getString( name: "surname");
                                String userId = jsonResponse.getString( name: "id");
                                saveData(userId);
                                Intent intent = new Intent( packageContext: LoginActivity.this, UserAreaActivity.class);
                                intent.putExtra( name: "name", name);
                                intent.putExtra( name: "surname", surname);
                                intent.putExtra( name: "username", username);
                                intent.putExtra( name: "id", userId.toString());
                                LoginActivity.this.startActivity(intent);
                            } else {
                                AlertDialog.Builder builder = new AlertDialog.Builder( context: LoginActivity.this);
                                builder.setMessage("Login Failed")
                                        .setNegativeButton( text: "Retry",  listener: null)
                                        .create()
                                        .show();
                            }
```

```
public class LoginRequest extends StringRequest {
    private static final String LOGIN_REQUEST_URL = "http://gocarts.000webhostapp.com/Login.php";
    private Map<String, String> params;

    public LoginRequest(String username, String password, Response.Listener<String> listener) {
        super(Method.POST, LOGIN_REQUEST_URL, listener,  errorListener: null);
        params = new HashMap<>();
        params.put( k: "username", username);
        params.put( k: "password", password);
    }

    @Override
    public Map<String, String> getParams() { return params; }
}
```

RegisterActivity, RegisterRequest – RegisterActivity obsługuje proces rejestracji użytkownika, przekazuje dane do RegisterRequest po czym dostaje informację zwrotną o statusie rejestracji. Jeżeli udało się zarejestrować użytkownik powróci do ekranu logowania.



```
public class RegisterActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        final EditText etSurname = (EditText) findViewById(R.id.etSurname);
        final EditText etName = (EditText) findViewById(R.id.etName);
        final EditText etUsername = (EditText) findViewById(R.id.etUsername);
        final EditText etPassword = (EditText) findViewById(R.id.etPassword);
        final Button bRegister = (Button) findViewById(R.id.bRegister);

        bRegister.setOnClickListener((v) -> {
                final String name = etName.getText().toString();
                final String surname = etSurname.getText().toString();
                final String username = etUsername.getText().toString();
                final String password = etPassword.getText().toString();

                if ((etName.length() > 20 || etName.length() < 3) || (etUsername.length() > 20 || etUsername.length() < 3) ||
                        (etPassword.length() > 20 || etPassword.length() < 3) || (etSurname.length() > 20 || etSurname.length() < 3)) {
                    startActivity(new Intent( packageContext: RegisterActivity.this, Pop.class));
                } else {
                    Response.Listener<String> responseListener = (response) -> {
                            try {
                                JSONObject jsonResponse = new JSONObject(response);
                                boolean success = jsonResponse.getBoolean( name: "success");
                                if (success) {
                                    Intent intent = new Intent( packageContext: RegisterActivity.this, LoginActivity.class);
                                    RegisterActivity.this.startActivity(intent);
                                } else {
                                    AlertDialog.Builder builder = new AlertDialog.Builder( context: RegisterActivity.this);
                                    builder.setMessage("Register Failed")
                                            .setNegativeButton( text: "Retry",  listener: null)
                                            .create()
                                            .show();
                                }
                            } catch (JSONException e) {
                                e.printStackTrace();
                            }
                        };

                        RegisterRequest registerRequest = new RegisterRequest(name, surname, username, password, responseListener);
                        RequestQueue queue = Volley.newRequestQueue( context: RegisterActivity.this);
                        queue.add(registerRequest);
                    }
            });
```



```
import ...

public class RegisterRequest extends StringRequest {
    private static final String REGISTER_REQUEST_URL = "http://gocarts.000webhostapp.com/Register.php";
    private Map<String, String> params;

    public RegisterRequest(String name, String surname, String username, String password, Response.Listener<String> listener) {
        super(Method.POST, REGISTER_REQUEST_URL, listener,  errorListener: null);
        params = new HashMap<>();
        params.put( k: "name", name);
        params.put( k: "surname", surname);
        params.put( k: "username", username);
        params.put( k: "password", password);
    }

    @Override
    public Map<String, String> getParams() { return params; }
}
```

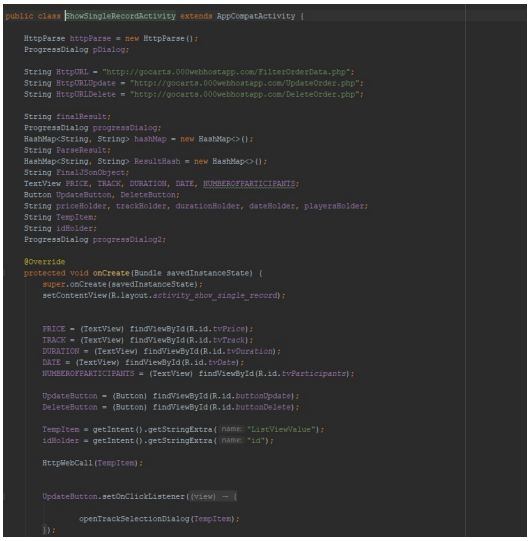
UserAreaActivity – obsługuję dodawanie rezerwacji oraz przekierowanie do listy rezerwacji.

```
public class UserAreaActivity extends AppCompatActivity {

    TextView tvWelcomeMsg;
    Button addOrderBtn, showOrders;
    ProgressDialog progressDialog;
    HashMap<String, String> hashMap = new HashMap<>();
    String finalResult;
    HttpParse httpParse = new HttpParse();
    String HttpURL = "http://gocarts.000webhostapp.com/AddOrder.php";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_user_area);

        Intent intent = getIntent();
        String name = intent.getStringExtra( name: "name");
        String surname = intent.getStringExtra( name: "surname");
        tvWelcomeMsg = (TextView) findViewById(R.id.tvWelcomeMsg);

        String message = "Witaj " + name + " " + surname;
        tvWelcomeMsg.setText(message);

        addOrderBtn = (Button) findViewById(R.id.addOrderBtn);
        showOrders = (Button) findViewById(R.id.buttonShow);

        final String userId = intent.getStringExtra( name: "id");

        addOrderBtn.setOnClickListener((view) -> { openHowToDialog(userId); });

        showOrders.setOnClickListener((view) -> {


                Intent intent = new Intent( packageContext: UserAreaActivity.this, ShowAllOrdersActivity.class);
                intent.putExtra( name: "id", userId.toString());
                UserAreaActivity.this.startActivity(intent);

        });
    }

    private void openHowToDialog(final String userId) {
        new AlertDialog.Builder( context: this).setTitle("Jak zarezerwować").setMessage("Po tej wiadomości otrzymasz listę wyborów twojej rezerwacji:" +
                " tor, liczbę uczestników oraz datę.").setCancelable(false)
                .setNeutralButton( text: "OK", (dialog, which) -> {
                        openTrackSelectionDialog(userId);
                }).show();
    }
```

ShowAllOrdersActivity – obsługuję listowanie wszystkich rezerwacji użytkownika.

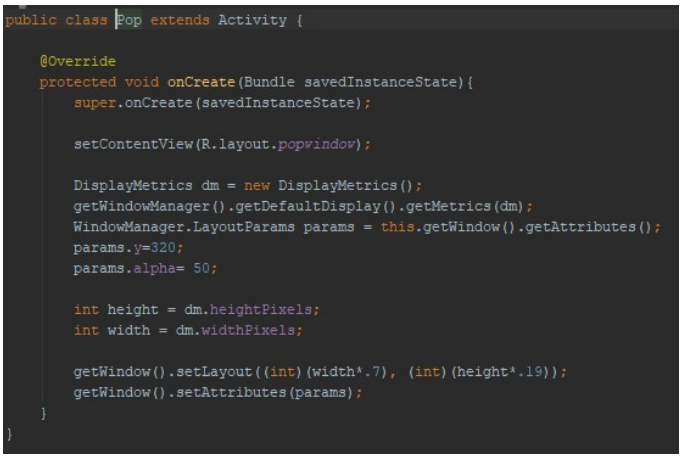
```
public void save_user_id(String id) { userId = id; }

private class GetHttpResponse extends AsyncTask<Void, Void, Void> {

    String HttpUrl = "http://gocarts.000webhostapp.com/AllOrdersData.php?userId=" + userId;

    public Context context;

    String JSonResult;

    List<Order> orderList;

    public GetHttpResponse(Context context) { this.context = context; }

    @Override
    protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Void doInBackground(Void... arg0) {
        HttpServicesClass httpServicesClass = new HttpServicesClass(HttpUrl);
        try {
            httpServicesClass.ExecutePostRequest();

            if (httpServicesClass.getResponseCode() == 200) {
                JSonResult = httpServicesClass.getResponse();

                if (JSonResult != null) {
                    JSONArray jsonArray = null;

                    try {
                        jsonArray = new JSONArray(JSonResult);

                        JSONObject jsonObject;
```

ShowSingleRecordActivity – obsługuję wyświetlanie rezerwacji, jej treści oraz usuwanie rejestracji. To tutaj także odbywa się edycja rezerwacji (w takich samych krokach jak w momencie rezerwacji).



```
public class ShowSingleRecordActivity extends AppCompatActivity {

    HttpParse httpParse = new HttpParse();
    ProgressDialog pDialog;

    String HttpURL = "http://gocarts.000webhostapp.com/FilterOrderData.php";
    String HttpURLUpdate = "http://gocarts.000webhostapp.com/UpdateOrder.php";
    String HttpURLDelete = "http://gocarts.000webhostapp.com/DeleteOrder.php";

    String finalResult;
    ProgressDialog progressDialog;
    HashMap<String, String> hashMap = new HashMap<>();
    String ParseResult;
    HashMap<String, String> ResultHash = new HashMap<>();
    String FinalJsonObject;
    TextView PRICE, TRACK, DURATION, DATE, NUMBEROFPARTICIPANTS;
    Button UpdateButton, DeleteButton;
    String priceHolder, trackHolder, durationHolder, dateHolder, playersHolder;
    String TempItem;
    String idHolder;
    ProgressDialog progressDialog2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_show_single_record);


        PRICE = (TextView) findViewById(R.id.tvPrice);
        TRACK = (TextView) findViewById(R.id.tvTrack);
        DURATION = (TextView) findViewById(R.id.tvDuration);
        DATE = (TextView) findViewById(R.id.tvDate);
        NUMBEROFPARTICIPANTS = (TextView) findViewById(R.id.tvParticipants);

        UpdateButton = (Button) findViewById(R.id.buttonUpdate);
        DeleteButton = (Button) findViewById(R.id.buttonDelete);

        TempItem = getIntent().getStringExtra( name: "ListViewValue");
        idHolder = getIntent().getStringExtra( name: "id");

        HttpWebCall(TempItem);


        UpdateButton.setOnClickListener((view) -> {

            openTrackSelectionDialog(TempItem);
        });
```

Pop – obsługuję wyświetlanie wiadomości o błednie wpisanych danych podczas rejestracji.



```
public class Pop extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);

        setContentView(R.layout.popwindow);

        DisplayMetrics dm = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(dm);
        WindowManager.LayoutParams params = this.getWindow().getAttributes();
        params.y=320;
        params.alpha= 50;

        int height = dm.heightPixels;
        int width = dm.widthPixels;

        getWindow().setLayout((int)(width*.7), (int)(height*.19));
        getWindow().setAttributes(params);
    }
}
```

Pliki PHP

Login.php – obsługuje logowanie do aplikacji, zwraca inofrmacje w formacie JSON o userze po zalogowaniu

Register.php – obsługuje rejestracjie użytkownika do bazy danych, zwraca informację w formacie JSON o pomyślnym zarejestrowaniu.

AllOrdersData.php – wyświetla wszystkie rezerwacje użytkownika.

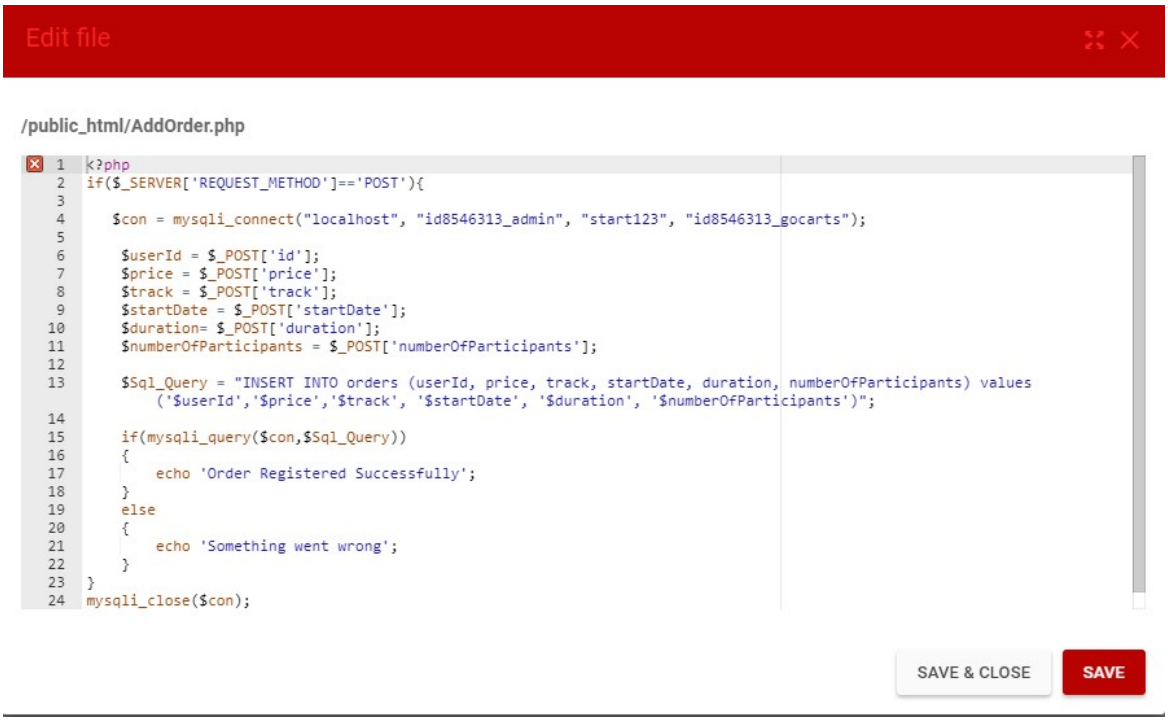
FilterOrderData.php – wyświelta wybraną przez użytkownika rezerwację z listy.

UpdateOrder.php – edytuje rezerwację w bazie z wcześniej otrzymanych danych POSTem.

AddOrders.php- dodaje zamówienie.

DeleteOrder.php – usuwa wybrane zamówienie.

Przykładowy plik do dodawania zamówienia do bazy



Projekt wykonał Mateusz Siwiec oraz Grzegorz Jagodziński