SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

ANDROID KALASOVELLUS

TEKIJÄ: Rami Inberg

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma/Tutkinto-ohjelma Tietotekniikan koulutusohjelma	
Työn tekijä(t) Rami Inberg	
Työn nimi Android Kalasovellus	
Päiväys 23.09.2018	Sivumäärä/Liitteet 23
Ohjaaja(t) Jussi Koistinen, Keijo Kuosmanen	
Toimeksiantaja/Yhteistyökumppani(t) 3XPM	

Tiivistelmä

Opinnäytetyön tavoitteena oli tehdä 3XPM järjestölle Android sovellus, jolla voisi pitää lokia pyydystettyjen kalojen lajista, painosta, pituudesta sekä pyydystyspaikasta. Lisäksi toteutettiin sovelluksen ympärille yksinkertaiset nettisivut, jolla tallennettaisiin lokit. Lisäksi sovellus toimii alustana kilpailulle, jossa järjestettäisiin kilpailijat saatujen kalasaaliiden perusteella. Alkuaan olisi ainoastaan 3 kategoriaa, jotka ovat suurin ahven, kuha ja hauki. Myöhempiin versioihin tulee lisää kategorioita. Lopulliseen versioon tulisi myös mahdollisuus tehdä omia kilpailuja.

Työ aloitettiin suunnitelemmalla tietokannan rakenne ja yhteydet tietokannan ja sovelluksen välillä. Tietokannaksi valittiin MySQL aikaisempien kokemuksien perusteella. Tietokannalle lähetettävä pyynnöt toteutettiin PHP mokkulalla, jota kutsuttiin tarvittaessa Android sovelluksesta. Android sovellus tehtiin ja testattiin Android Studiolla, joka käytti Googlen karttapalvelua.

Työ oli odotettua vaikeampi, joten kaikkia ominaisuuksia ei saatu tehtyä opinnäytetyön aikana. Sovelluksella pystyy lisäämään, muokkaamaan tai poistamaan lokeja. Kuitenkaan omia kilpailuja ei pysty lisäämään halutulla tavalla ja sovelluksen tyylit eivät ehtineet valmiiksi kaikissa näkymissä. Sovelluksen on tarkoitus laittaa Googlen Play kauppaa ensi vuoden keväällä.

Avainsanat
Android, Joomla, MySQL, Google Maps

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author(s) Rami Inberg			
Title of Thesis Android Fish Application			
Date	23.9.2018	Pages/Appendices	23
Supervisor(s) Jussi Koistinen, Keijo Kuosmanen			
Client Organisation /Partners 3XPM			

Abstract

The purpose of this thesis was to make an Android application. This thesis was commissioned by the 3XPM organization so that they could keep track of the species, weight, length of the catches and the catching locations. The application serves as a platform for users to create their own fishing competitions. The biggest perch, pike and zander will be the main categories in the application, but later versions will add more.

The work started by making a structured database and connections between the database and the application. MySQL was chosen as the database based on personal experience. The Android application request from the database was implemented using the PHP module. An Android app was created and tested with Android Studio.

Making this thesis took more time than initially expected and all the features were not be completed during the process. As a result of this thesis, the basic functionalities of adding, editing and dropping records are ready. Also, there are scoreboards to watch the rankings of all users. However, the user's own competitions cannot be added as desired and the application styles did not complete in all views. The application will be published in Google Play store next spring.

Keywords
Android, Joomla, MySQL, Google Maps

SISÄLTÖ

1 JOHDANTO

Tämän opinnäytetyön tarkoituksena on suunnitella ja toteuttaa Android sovellus. Lisäksi tarvittiin Joomla moduuli, joka lähettää sovelluksen saamat tiedot MySQL serverille. Tämä opinnäytetyö on suunniteltu ja toteutettu järjestölle nimeltä 3xPM. Tässä opinnäytetyössä keskitytään Android sovellukseen.

Olen itse jäsenenä tässä järjestössä, jonka kautta tuli ehdotuksena tämä aihe opinnäytetyöhöni. Tarkoituksena oli tehdä kalastusloki sovellus, joka helpottaa osallistujien tuloksien ylläpitämistä ja toimii samalla käyttäjien päiväkirjana. Kysyttiin järjestön jäseniltä heidän käyttämiensä puhelimien käyttöjärjestelmästä ja ylivoimaiseksi voittajaksi muodostui Android. Yhdellä oli käytössä Windows Phone, eikä kenelläkään ollut käytössä Applea.

Käytännön mobiilisovellus toteutettiin Android Studion avulla. Tietokanta valittiin järjestön sivuston mukaan, jossa oli valmiksi asennettuna MySQL. Tämä opinnäytetyö lähti tyhjästä ja pyrittiin rakentamaan toimiva sovellus opinnäytetyön lopuksi.

2 KÄYTETYT TEKNIIKAT

Tämä opinnäytetyön tekniikat valittiin osaamiseni perusteella. Tekniikat jaettiin kahteen osaan, Android studiolla tehtiin varsinainen sovellus ja Joomlalla hoidettiin tietokantojen yhteydet.

2.1 Android Studio

Android Studio on virallinen integroitu kehitysympäristö Androidia käyttäville puhelimille. Google kehittämä Android studio julkaistiin vuonna 2014 korvaamaan Eclipsen Android käyttöjärjestelmien sovelluskehityksessä. Android Studiolla voi emuloida Android-sovelluksia eri laitteilla ja käyttöjärjestelmäversioilla. Android studiolla voi kehittää sovelluksia Androidia käyttäville puhelimille, kelloille ja televisioille. Android studio käyttää ohjelmointikielenä Javaa. (Android)

2.1.1 XML

XML on ohjelmistojen ja laitteista riippumaton merkintäkieli tietojen tallentamiseen ja siirtämiseen. Android studio käyttää XML näkymien luomiseen. (XML)

2.1.2 Java

Java on ohjelmointikieli ja tietojenkäsittelyalusta, jonka julkaisi Sun Microsystems vuonna 1995. Javaa käytetään laajasti sovelluksissa ja nettisivuissa. (Java)

2.1.3 Google Maps

Google karttapalvelu on Androidille tehty ohjelmistonkehitysalusta. Rajapinta käsittelee valmiiksi yhdistämisen Googlen servereille, tietojen lataamisen ja kartan kuuntelemisen. Lisäksi voit karttaa lisätä merkintöjä, monikulmioita, muuttaa kartan ulkoasua ja muuttaa käyttäjän näkymää tietyllä alueella kartassa. (Google)

2.1.4 Paint.net

Paint.net on ilmainen kuvankäsittely sovellus Windows käyttöjärjestelmille. Käytetty tässä opinnäytetyössä kuvien tekemiseen. (Paint)

2.2 Joomla

Joomla on ilmainen, avoimen lähdekoodin hallintajärjestelmä (CMS). Joomla on rakennettu MVC kehyksen ympärille. Joomlassa on sisäänrakennettu hallintajärjestelmä, jonka ansioista sinun ei tarvitse osata ohjelmoida käyttääkseen Joomla sivustoa. (Joomla)

2.2.1 Joomla moduuli

Joomla moduulit ovat pieniä laajennuksia, joita voidaan rakentaa Joomla-sivun sisälle. Näitä voi lisäillä ja poistaa dynaamisesti sivulta.

2.2.2 PHP

PHP on laajasti käytetty avoimen lähdekoodin ohjelmointikieli. Käytetään yleisesti nettikehityksessä ja PHP:n voi sisällyttää suoraan HTML sisään. PHP tarvitsee toimiakseen palvelimen, joka kuuntelee selainta. (PHP)

2.2.3 JSON

JSON on kevyt JavaScriptiin perustuva tiedonkäsittelyalusta. JSON on rakennettu ihmisille ja koneille helpoksi luettavaksi. Melkein kaikki modernit ohjelmointikielet tukevat JSONia. (JSON)

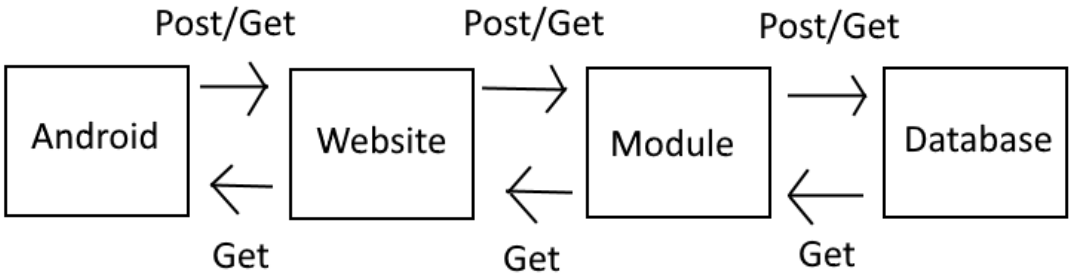
2.3 MySQL

MySQL on relaatiotietokantaohjelmisto. Eniten käytetään web-palveluiden tietokantana. Vuonna 2009 MySQL omistus siirtyi Oraclelle. (MySQL)

3 SUUNNITTELU

Tarkoitus oli alun perin tehdä opinnäytetyössä sivusto ja Android sovellus. Kuitenkin tehtävän laajuus oli liian suuri, joten sivuston tekeminen poistettiin opinnäytetyöstä. Sovellukseen toiminnan kannalta vaadittiin silti PHP-moduulin keskustelemaan serverin ja Android sovelluksen kanssa. Tietokannaksi valittiin MySQL. Projektissa oli varsin vapaat kädet suunnittelussa, joten käytettiin projektissa teknologioita, mitkä olivat entuudestaan tuttuja.

Sovelluksen saa tietonsa lähettämällä pyyntöjä web-serverille, joka tekee kyselyn tietokantaan. Sovellus lähettää tietokannalle tarvittavat tiedot POST pyynnöllä tai pyytää tietoa tietokannasta GET pyynnöllä.



Kuva 1. Sovelluksen ja tietokannan yhteyden rakennekuva.

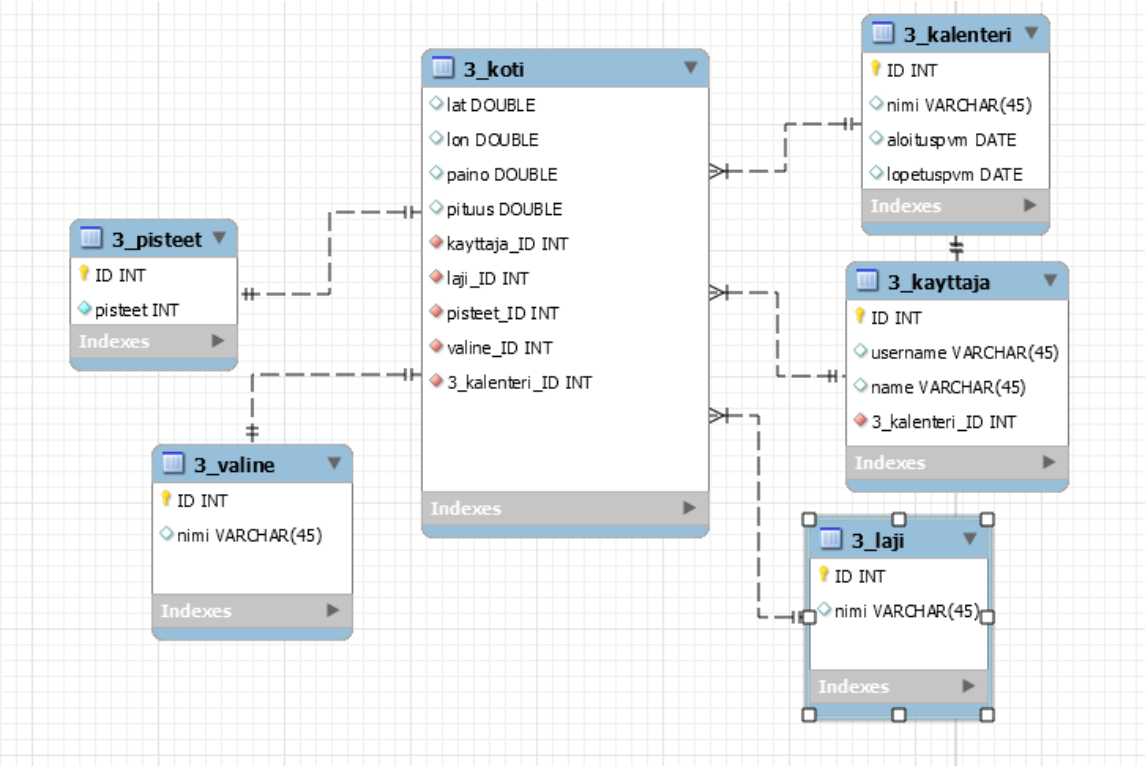
3.1 Tietokanta

Tietokannan valinta oli helppoa, kun järjestöllä valmiiksi ostetuissa nettisivuissa tuli mukana MySQL tietokanta. Lisäksi on tullut käytettyä MySQL tietokantoja ennenkin erilaisissa kouluprojekteissa ja omissa projekteissa. Tietokantana päädyttiin käyttämään tähtimallia, joka kokoaa kaikki tarvittavat tiedot yhteen tauluun.

3.1.1 MySQL Workbench

MySQL Workbenchiä on tietokantojen suunnittelussa käytetty sovellus, jolla pystytiin suunnittelemaan ja toteuttamaan tietokanta helposti. Tämän ansioista ei tarvittu kirjoittaa luonti koodeja tietokantaa. Ohjelmasta saatiin ulos nämä luontikoodit automaattisesti omien suunnittelujen pohjalta. Sovelluksen käyttämä tietokanta rakentui tähtimallin mukaiseksi. Tähtimaallisessa tietokannan rakenteessa on keskellä yksi tietokanta taulu, missä on viittauksia muihin tauluihin. Tähän tauluun kerätään kaikki tiedot muista tietokanta tauluista.

Alkuperäisiä suunnitelmia jouduttiin päivittämään muutamaan kerran projektin aikana, johtuen asioista mitä oli vaikeaa ottaa huomioon alkuperäisissä suunniteluissa. Workbenchin ansioista sai kuitenkin päivitettyä tehdyt muutokset kätevästi ja viedä ne tuotantoversioon.



Kuva 2. Tietokannan tähtimallin rakenne MySQL Workbenchillä.

3.2 Android sovellus

Android sovellus toteutettiin Android Studion avulla. Alkuvaiheen ongelma sovelluksen suunnittelussa oli hahmottaa, mitä kaikkea pystyttiin toteuttamaan tämän opinnäytetyön sisällä. Alkuaan oli paljon ideoita, jota jouduttiin karsimaan projektin aikana. Päädyttiin toteuttamaan yksinkertainen sovellus, jolla pystyi tallentamaan Google Maps avulla lokia omasta kalastauksesta ja lisäksi näyttämään muitten tulokset. Alkuaan valittiin kolme kalaa (ahven, kuha ja hauki), joita voisi lisätä sovellukseen. Tarkoitus olisi lisätä sitten lisää myöhemmin, kun muu sovellus on valmiina. Näistä tehtiin kolme pistetaulukkoa, jotka rankkaisivat käyttäjät saatujen kalojen painon mukaan. Pistetaulukkojen tuloksien mukaan laadittiin taulukko, joka sisältää kokonaistilanteen. Yksittäisen lajipistetaulukon voittaja saa tietyn määrä pisteitä ja se yhdistetään muitten pistetaulukkojen sijoituksiin. Näistä sitten määräytyisi kokonaistilanne.

Lisäksi lisättiin mahdollisuus tehdä omia kilpailuja käyttäjille, jolloin olisi mahdollista tehdä yksityisiä kilpailuja. Tämä ominaisuus toteutettaisiin projektin loppuvaiheessa. Tärkeintä oli saada toimimaan perusominaisuudet ensin. Kilpailut pystyivät myös kokonaan ottamaan pois käytöstä. Tämä antaisi mahdollisuuden käyttäjälle pitämään sovellusta ainoastaan henkilökohtaisena lokina.

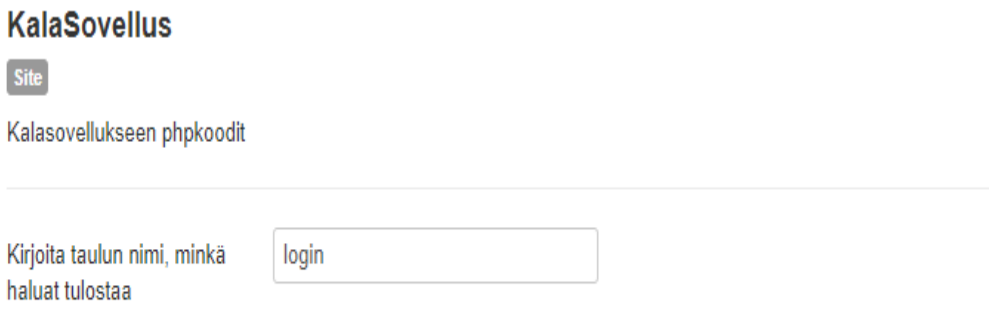
3.3 Joomla Moduuli

Joomla moduuleita käytettiin sovelluksen ja serverin keskusteluun. Tarkoitus oli tehdä mahdollisemman yksinkertainen moduuli. Sivuston ylläpitäjä tarvitsi vaan kirjoittaa mitä tietoja halutaan serveriltä ja serveriltä tulisi vastaus JSON muodossa. Sovellus taas lukisi tämän JSON tuloksen ja käsitte-

lisi tiedot halutulla tavalla. Tulevaisuudessa näitä moduuleilla saisi ulos pistetaulukon suoraan tietokannasta sivustolle. Tämä ei kuitenkaan ehtinyt valmistumaan opinnäytetyön aikana. Alla olevassa koodissa näkyy kuinka yksinkertaisesti saadaan haluttu tieto tietokannasta.

```
static function getData(&$params)
{
    $TableParams = array();
    $TableParams['txttable'] = $params->get( 'txttable' );
    return $TableParams;
}
```

Kuva 3. PHP koodi, joka palauttaa tekstilaatikon tekstin



Kuva 4. Joomlan näkymä Kuva 1, tekstilaatikolle

4 TOTEUTUS

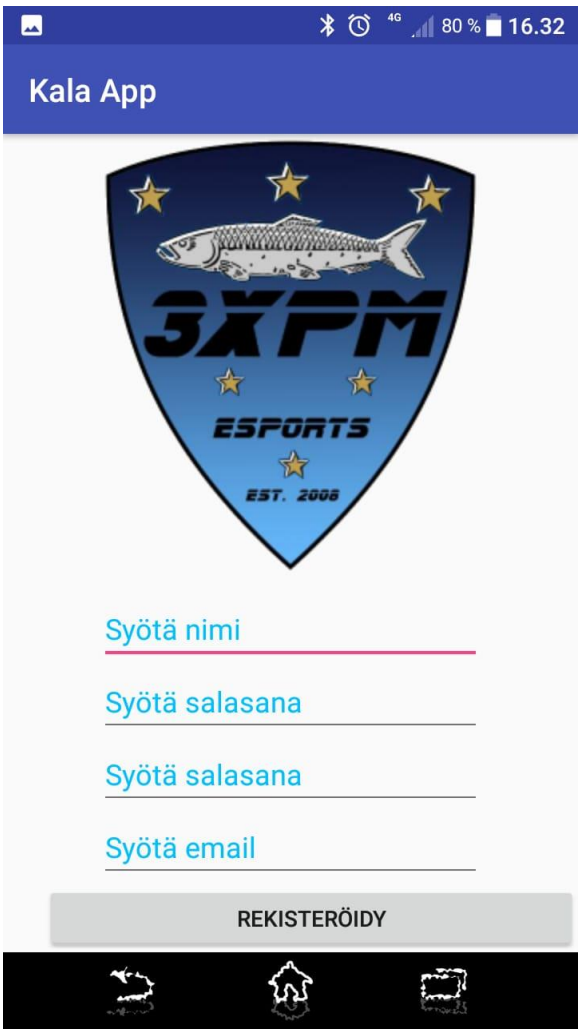
4.1 Kirjautuminen

Sovelluksen käynnistyessä ensimmäinen sivu on kirjautumisnäkymä. Tässä näkyy järjestön logo, mahdollisuus rekisteröityä uutena käyttäjänä tai kirjautua vanhana. Sovellus tarkistaa käyttäjätunnuksen ja salasanan oikeuden kirjautumisen yhteydessä ja avaa uuden näkymän päänäkymään. Tarkoitus olisi tähän lisätä myöhemmin tarkistuksen liian monesta kirjautumisyrityksessä.

Kuva 5. Kirjautuminen sovelluksessa

```
try
{
    JSONObject jsonObj = new JSONObject(jsonlause);
    JSONArray items= jsonObj.getJSONArray( name: "items");
    //Jos serveriltä löytyy käyttäjätunnus.
    if (items.length() != 0)
    {
        //Muutetaan takaisin objectiksi, että voidaan verrata.
        JSONObject obj = items.getJSONObject( index: 0);
        //Luodaan uusi intentti, johon tallennetaan käyttäjäid, jotta muut activityt saa sen tiedon.
        Intent x = new Intent( packageContext: LoginActivity.this, MapsActivity.class);
        x.putExtra( name: "kayttajaID", obj.getInt( name: "ID"));
        startActivity(x);
    }
    //Jos käyttäjätunnus on väärin.
    else
    {
        Toast.makeText( context: LoginActivity.this,  text: "Käyttäjätunnus tai salasana on väärin",Toast.LENGTH_LONG).show();
    }
}
catch (Exception e)
{
    //Printataan JSON errorit
    e.printStackTrace();
}
```

Kuva 6. Käyttäjätunnuksen ja salasanan tarkistaminen



Kuva 7. Rekisteröityminen sovelluksessa

4.2 Ulkoasu

Käyttäjän todennettua oikeaksi käyttäjäksi hänet siirretään päänäkymään. Päänäkymä taustalla on Google Mapsin karttanäkymä ja kartassa näkyy käyttäjän saamat kalat ja niitten sijainnit. Tällä sivulla pystyt paikantamaan itsesi painamalla oikeassa yläkulmassa olevaa kuvaa, lisäämään uusia lokeja painamalla vihreää plus symbolia vasemmalta yläkulmasta, muokkaamaan vanhojen lokien sijaintia painamalla kalankuvaa ja vetämällä sen haluttuun sijaintii tai poistamaan kokonaan lokeja vetämällä lokin roskalaatikkoon, joka tulee näkyviin kalan valitsemisen jälkeen. Päätettiin toteuttaa paikallistaminen, niin ettei sovellus turhaan paikallistaisi, vaan paikallistaisi käyttäjän pyynnöstä. Painamalla vasemmasta yläkulmasta olevaa tähteä pääsee katsomaan kilpailuja ja tuloksia.

Kuva 8. Päänäkymä sovelluksessa, taustalla Google Maps

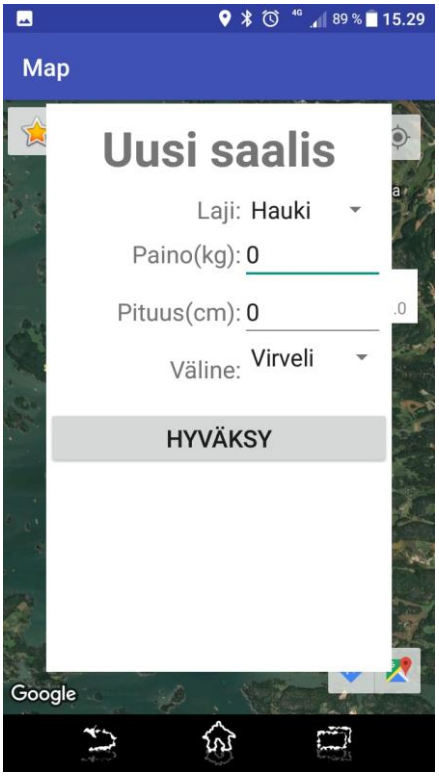
4.2.1 Lisäys

Käyttäjä pystyy lisäämään uusia lokeja painamalla yläkulmassa olevaan plus painiketta. Tämän jälkeen avautuu uusi ikkuna. Uuteen saaliiseen syötetään saamansa kalalaji, kalan paino, kalan pituus ja millä välineellä se on saatu. Välinettä kysytään kalastustavan takia. Mitä vaikeammalla tavalla kala pyydystetään sen isomman kertoimen saa. Suurin kerroin on kalan pyydystämisellä käsillä ja pienin kerroin on verkolla tai muulla pyydyksellä.

Tässä vaiheessa ei tarvitse laittaa mistä kyseinen kala on saatu. Sovellus katsoo valmiiksi Google Mapsista saaduilla arvoilla käyttäjän sijainnin. Käyttäjän on kuitenkin itse päivitettävä sijainti tarpeen vaatiessa. Sijainti tulee kartan viimeisellä paikannukselta.

```
326             JSONArray items= jsonObj.getJSONArray( name: "items");
327                 for (int i = 0; i < items.length(); i++)
328                 {
329                     JSONObject c = items.getJSONObject(i);
330                     if(c.getInt( name: "kayttaja_ID") == kayttajaID)
331                     {
332
333                         switch (c.getInt( name: "laji_ID"))
334                         {
335                             case 1: //ahven
336                                 LatLng ahvenpaikka = new LatLng(c.getDouble( name: "lat"), c.getDouble( name: "lon"));
337                                 Marker ahvenm = mMap.addMarker(new MarkerOptions().position(ahvenpaikka)
338                                         .title("Ahven").icon(BitmapDescriptorFactory.fromResource(R.mipmap.ic_ahven))
339                                         .draggable(true).snippet("paino: " + c.getDouble( name: "paino") + " pituus: "
340                                                 + c.getDouble( name: "pituus")));
341                                 ahvenm.setTag(c.getString( name: "ID"));
342                                 break;
343                             case 2: //kuha
344                                 LatLng kuhapaikka = new LatLng(c.getDouble( name: "lat"), c.getDouble( name: "lon"));
345                                 Marker kuham = mMap.addMarker(new MarkerOptions().position(kuhapaikka).title("Kuha")
346                                         .icon(BitmapDescriptorFactory.fromResource(R.mipmap.ic_kuha))
347                                         .draggable(true).snippet("paino: " + c.getDouble( name: "paino") + " pituus: "
348                                                 + c.getDouble( name: "pituus")));
349                                 kuham.setTag(c.getString( name: "ID"));
350                                 break;
351                             case 3://hauki
352                                 LatLng haukipaikka = new LatLng(c.getDouble( name: "lat"), c.getDouble( name: "lon"));
353                                 Marker haukim = mMap.addMarker(new MarkerOptions().position(haukipaikka).title("Hauki")
354                                         .icon(BitmapDescriptorFactory.fromResource(R.mipmap.ic_hauki))
355                                         .draggable(true).snippet("paino: " + c.getDouble( name: "paino") + " pituus: "
356                                                 + c.getDouble( name: "pituus")));
357                                 haukim.setTag(c.getString( name: "ID"));
358                                 break;
359                         }
360                     }
361                 }
```

Kuva 9. Uuden saaliin lisäys Android Studiossa koodina



Kuva 10. Uuden saaliin lisääminen sovelluksessa

4.2.2 Muokkaus

Kuitenkin Google Maps ei ole täydellinen paikallistamiseen. Välillä se paikallistaa sinut väärin, mikä voi johtua satelliiteista, luonnon muodosta tai sinulla ei ole yksinkertaisesti nettiyhteyttä. Tämä ongelma ohitettiin antamalla käyttäjän itse muokata lokeja. Käyttäjä pystyy vetämään lokin haluamaansa paikkaan. Kalastuslokit näkyvät kartalla ahvenena, haukena tai kuhana. Riippuen lajista, jota siinä on pyydystetty. Painamalla yksittäistä lokia näet kyseisen kalan painon ja pituuden.

```
final String key = marker.getTag().toString();

final LatLng latlng = marker.getPosition();

//Tehdään string request post
StringRequest stringRequest = new StringRequest(Request.Method.POST,  url: [redacted]  new Response.Listener<String>(){
    @Override
    public void onResponse(String ServerResponse){

        Log.e( tag: "Server: ", ServerResponse);
        //Näyttää echo viestit, jotka tulee serveriltä.
        //Toast.makeText(MapsActivity.this, ServerResponse, Toast.LENGTH_LONG).show();
    }
},
        new Response.ErrorListener(){
            @Override
            public  void onErrorResponse(VolleyError volleyError){
                Log.e( tag: "Volley: ", volleyError.getMessage());
                //Näyttää echo viestit, jotka tulee serveriltä.
                Toast.makeText( context: MapsActivity.this, volleyError.toString(), Toast.LENGTH_LONG).show();
            }
        }) {
    @Override
    protected Map<String, String> getParams() {

        double doulat = latlng.latitude;
        String lat = Double.toString(doulat);

        double doulon = latlng.longitude;
        String lon = Double.toString(doulon);
        //Tehdään map string parametrit
        Map<String, String> params = new HashMap<>();
        params.put( k: "ID", key);
        params.put( k: "lat", lat);
        params.put( k: "lon", lon);

        return params;
    }
};

//Tehdään volley newRequestQueue
requestQueue = Volley.newRequestQueue( context: MapsActivity.this);

//Lisätään StringRequest objekti requestQueue
requestQueue.add(stringRequest);
```

Kuva 11. Koodina muokkauksen toiminnallisuus

4.2.3 Poisto

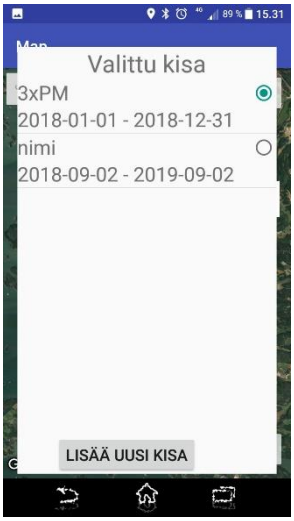
Lokin poisto tapahtuu samalla tavalla kuin muokkaaminen. Käyttäjä valitsee poistettavan lokin ja painaa sitä lokia pitkään, jonka jälkeen ilmestyy roskalaatikko yläkulmaan. Käyttäjä pystyy vetämään lokin suoraan roskalaatikkoon. Roskalaatikko vaihtaa väriä lokin oltaessa riittävän lähellä.

```
363     @Override
364     public void onMarkerDragStart(Marker marker) {
365         this.deleteMarker.setVisibility(View.VISIBLE);
366         origMarkerPos = marker.getPosition();
367     }
368
369     @Override
370     public void onMarkerDrag(Marker marker) {
371         final ImageView deleteMarker = this.deleteMarker;
372         Point markerScreenPosition = mMap.getProjection().toScreenLocation(marker.getPosition());
373         if(overlap(markerScreenPosition, deleteMarker))
374         {
375             deleteMarker.setImageResource(R.mipmap.ic_thrash_red);
376         } else {
377             deleteMarker.setImageResource(R.mipmap.ic_thrash);
378         }
379
380     }
381
382     private boolean overlap(Point point, ImageView imgview) {
383         int[] imgCoords = new int[2];
384         imgview.getLocationOnScreen(imgCoords);
385         boolean overlapX = point.x < imgCoords[0] + imgview.getWidth() && point.x > imgCoords[0] - imgview.getWidth();
386         boolean overlapY = point.y < imgCoords[1] + imgview.getHeight() && point.y > imgCoords[1] - imgview.getWidth();
387         return overlapX && overlapY;
388     }
```

Kuva 12. Koodina merkitsemisen tunnistaminen vedettäessä

4.3 Kilpailut

Kilpailut valikoista pystyy valitsemaan oman kilpailu. Tällä hetkellä pystyy osallistumaan vain yhteen kilpailuun. Tulevaisuudessa olisi tarkoitus korjata radionapit valintaruuduilla, joka mahdollistaisi osallistumaan useampaan kisaan samaan aikaan. Tällä hetkellä ei pysty vaihtamaan kilpailua ja joutuu osallistumaan oletuskilpailuun. Tämä kuitenkin on tarkoitus korjata ennen kuin sovellus julkaistaan play kauppaan. Kisan lisääminen toimii, mutta näkymän tyyli jäi vähän keskeneräiseksi.

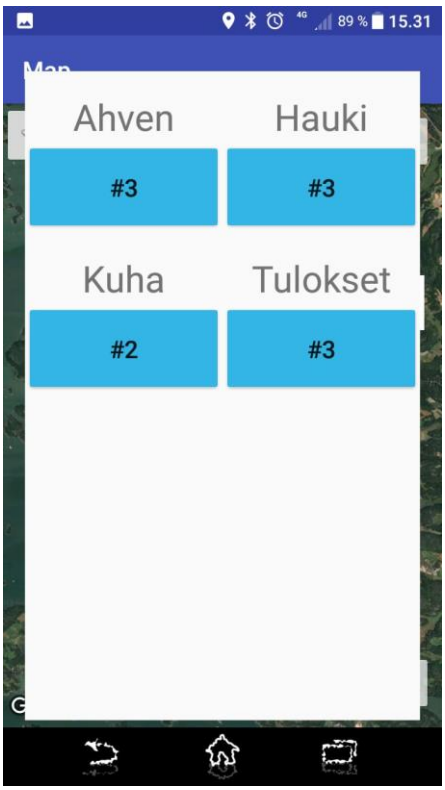


Kuva 13. Kisalistan näkymä sovelluksessa

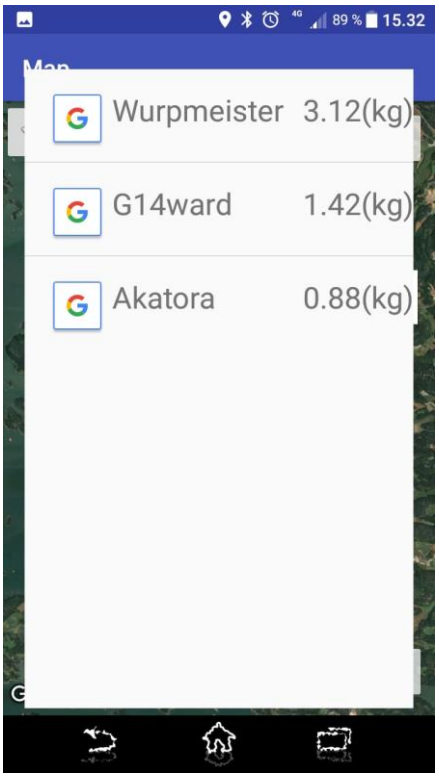
Kuva 14. Kisan lisääminen

4.4 Tulokset

Tulokset näkymästä näkyy oma sijoitus kaikista kategorioista. Painamalla haluttua kategoriaa näkee oman tuloksen ja muitten tuloksen valitussa kategoriassa. Tulevaisuudessa tähän lisätään mahdollisuus kustomoida oman profiiliin.



Kuva 15. Tulokset näkymä sovelluksessa



Kuva 16. Ahvenen tulokset sovelluksessa

```
public void afterTextChanged(Editable s) {
    counter++;
    Helper h = new Helper();
    Map<String, Integer> ahvenpoints = h.getPoints(ahvenlist);
    points = h.finalpoints(ahvenpoints, points);
    alllist = h.getAllPoints(ahvenlist, alllist);

    if(counter == 3)
    {
        points = h.sortByValue(points);
        alllist = h.sortJsonArray(alllist);
        alllist = h.reverseOrder(alllist);

        try{

            int tmp = points.size();
            int sij, x = 0;
            String kayttaja = Integer.toString(kayttajaID);
            Iterator it = points.entrySet().iterator();
            while (it.hasNext())
            {
                Map.Entry pair = (Map.Entry)it.next();

                sij = tmp - x;
                x++;
                it.remove();
                String testi = pair.getKey().toString();
                if(testi.equals(kayttaja))
                {
                    btnPoints.setText("#" + sij);
                    btnPoints.setEnabled(true);
                }
            }
            if(btnPoints.getText() == "Loading...")
            {
                btnPoints.setText("Ei dataa");
                btnPoints.setEnabled(true);
            }

        }catch (Exception e)
        {
            e.printStackTrace();
        }

        Log.e( tag: "TAG",  msg: "afterTextChanged: " + points);
    }
```

Kuva 17. Ahven painikkeen sijoituksen määrittäminen

5 JATKOKEHITTELY

5.1 Tulevaisuuden toiminnot

Tähän sovellukseen on olemassa niin paljon ideoita tulevaisuudessa. Tässä projektissa loppui aika kesken, eikä ehditty toteuttamaan kaikkea haluttuja ideoita.

5.1.1 Profiilisivu

Oma profiilisivu täydentämään omia asetuksia. Pitäisi pystyä muokkaamaan omaa kuvaa, nimimerkkiä ja salasanaa sovelluksen sisällä. Pystyisi asetuksissa myös muuttamaan tyyliä ja pitäisi pystyä asettamaan, ettei osallistu mihinkään kisaan.

5.1.2 Ystävälistat

Pystyisi lisäämään käyttäjille ystäviä. Tämän ansioista ei tarvitsisi välttämättä tehdä erikseen kilpailua, vaan olisi aina ystävän kanssa kilpailussa. Asetuksista pystyisi kieltämään tämän ominaisuuden.

Asetuksista pystyisi myös hyväksymään jaetun lokin ystävän kanssa. Tämän ansioista pystyy suoraan jakamaan käyttäjän lokin kaverien kanssa.

5.1.3 Kuvasta tunnistaminen

Tätä ominaisuus on vasta alkuvaiheessa. Käytettäisiin samaa kirjastoa, mitä Facebook käyttää tunnistamaan ihmisten naamoista. Voi tulla teknisiä vaikeuksia lajien tunnistamista johtuen lajien samanlaisuudesta. Teknisesti tämä kuitenkin olisi mahdollista.

5.2 Web sivusto

Sivuston tarkoitus olisi näyttää järjestön omia asioita, sekä taulukoita, kokonaissijoituksia ja historiaa mobiilisovelluksen keräämästä datasta. Tänne voisi laittaa historiaa kalojen suuruudesta, viime kausien voittajista tai parhaista kalastusalueista.

5.3 RKTL

Riista ja kalatalouden tutkimuslaitos määrittää kalakiintiöt tietyille alueille ja suojelutarpeet tietyille lajeille. Tarkoitus olisi lähettää anonyymista dataa sovelluksen saadun datan perusteella. Kalastuksesta on vaikea kerätä dataa, kun osa kalastuksen vuosittaisesta saalista tulee vapaa-ajan kalastajilta. Sovellus voisi auttaa samaan tätä dataa.

6 POHDINTA

Opinnäytetyöstä aloitettu projektia tullaan vielä jatkamaan monia vuosia. Tälle sovelluksella on varmasti kysyntää isoimmillakin markkinoilla. Sovelluksen perusrakenteen ollessa valmis pystyisi tätä laajentamaan muihinkin eläimiin, kasveihin tai sieniin. Muitakin tämän tapaisia sovelluksia on tullut play kauppaan. Kuitenkaan itse en ole vielä löytänyt toimivaa sovellusta, joka toimisi tämän sovelluksen tavoin.

Ongelmia tässä opinnäytetyön suorittamisessa oli useampia. Ei enää riittänyt aika tehdä tätä sovellusta töitten lomassa. Sain nykyisen työpaikkani tämän sovelluksen ansiosta ja työkuvani on tällä hetkellä suunnitella ja toteuttaa Android sovelluksia. Tästä johtuen olen kehittynyt ohjelmoijana ja sovelluksesta käytetyssä koodissa on muutamia ongelmakohtia, jotka pitäisi korjata. Joomla tuntuu olevan vähän liian raskas ja eikä niin helposti käytettävä, kuten alkuaan ajattelin. Sovelluksia pitää päivittää jatkuvasti teknologioiden kehittyessä, joten olisi tärkeää hienosäätää perusteet vankaksi, jotta päivittäminen olisi mahdollisimman helppoa.

Olen tämän kesän kokeillut sovellusta ja se toimii enimmäkseen halutulla tavalla. Kaloja tallentavat lokit toimivat, mutta se ei jostain syystä enää päivitä reaaliajassa laittaessa uuden lokin. Silloin pitää kirjautua ulos ja käynnistää sovellus uudestaan. Lisäksi Googlen navigointia mahdollistavat satelliitit eivät toimi Turun saaristossa kauhean hyvin, tämä voi myös johtua nettiyhteydestä. Tämä nyt ei ole niin kamala ongelma, kun lokeja voi päivittää myös kalastuksen jälkeenkin. Välillä saattaa olla myös niin huono ilma, ettei ole mahdollisuutta käyttää sovellusta. Sovellukseen pitäisi lisätä mahdollisuus toimia myös ilman nettiyhteyttä ja data lähetettäisiin silloin kuin kännykkä löytää nettiyhteyden. Ensi kesänä olisi tarkoitus testata järjestön jäsenillä ja saada sieltä tietoa, miten tätä sovellusta voitaisiin parantaa.

7 LÄHDELUETTELO

Java. [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://www.java.com/en/download/faq/whatis_java.xml

Google. Google Maps [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://developers.google.com/maps/documentation/android-sdk/intro

Paint.net. Wikipedia [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://en.wikipedia.org/wiki/Paint.net

XML. W3Schools [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://www.w3schools.com/xml/xml_whatis.asp

Joomla. [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://www.joomla.org/about-joomla.html

PHP. [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
http://php.net/manual/en/intro-whatis.php

JSON. [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://www.json.org/

MySQL. Wikipedia [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://fi.wikipedia.org/wiki/MySQL

Android Studio. Wikipedia [Verkkoaineisto] [Viitattu: 2018-05-07.] Saatavissa:
https://fi.wikipedia.org/wiki/Android_Studio