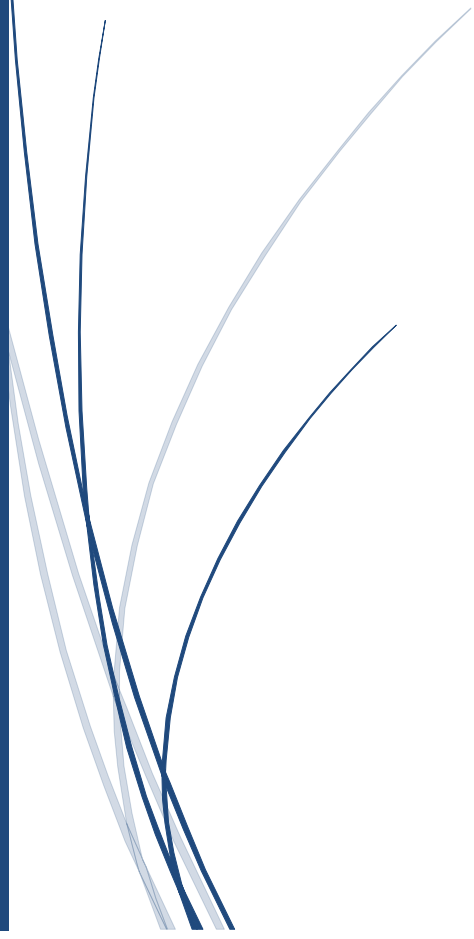
Autor: Krzysztof Małysa

CosmeticCare

Aplikacja mobilna rozpoznająca szkodliwe składniki w kosmetykach



1. Użyte biblioteki/technologie.

- Google Vision – biblioteka OCR,
- JSON (JavaScript Object Notation) - lekki format wymiany danych komputerowych. JSON jest formatem tekstowym, bazującym na podzbiorze języka JavaScript,
- Barcodescanner zxing – biblioteka do odczytywania kodów kreskowych,
- Volley - biblioteka od google do połączeń http, tworzy kolejki, dzięki którym nie można jednocześnie wysłać kilku zapytań do bazy,
- Android Studio - środowisko programistyczne (IDE) stworzone przez Google na bazie IntelliJ, które kierowane jest do developerów aplikacji na Androida. Pozwala ono wygodnie projektować, tworzyć i debugować własne programy na najpopularniejszą obecnie platformę systemową dla urządzeń mobilnych.
- PHP Storm - kompletne, multiplatformowe środowisko programistyczne, umożliwiające pracę z aplikacjami PHP. Posiada funkcje podpowiadania składni: klas, funkcji, metod, indeksów tablic oraz nazw zmiennych. Edytor obsługuje dokumentacje tworzone w formacie PHPDoc, oferuje możliwość refaktoryzacji kodu, w tym zmiany nazwy: plików, funkcji, stałych, klas, metod, parametrów czy zmiennych. W PhpStorm będzie można debugować zarówno aplikację (Zend Debugger oraz Xdebug) jak i JavaScript oraz przeprowadzić testy jednostkowe. Program posiada wsparcie dla narzędzi wiersza poleceń Zend Framework oraz Symfony, wspomagających zarządzanie projektami.
- MySQL - system zarządzania relacyjnymi bazami danych, nasza baza dostępna jest globanie na serwerze.

2. Opis aplikacji.

Aplikacja, dzięki użyciu najnowszych bibliotek i technologii pozwala na rozpoznanie szkodliwych substancji w kosmetykach poprzez zeskanowanie w czasie rzeczywistym etykiety bądź kodu kreskowego. Aplikacja w swojej bazie zawiera składniki, na które powinniśmy zwrócić uwagę. Substancje szkodliwe zostały podzielone na 3 kategorie:

- znikoma szkodliwość: substancja nie powoduje skutków ubocznych lub nie wywołuje żadnych negatywnych reakcji organizmu,
- średnia szkodliwość: należy zwrócić uwagę na ten składnik ponieważ potencjalnie może być on szkodliwy,
- wysoka szkodliwość: najlepiej unikać tego składnika, jest on szkodliwy i negatywnie wpływa na nasz organizm.

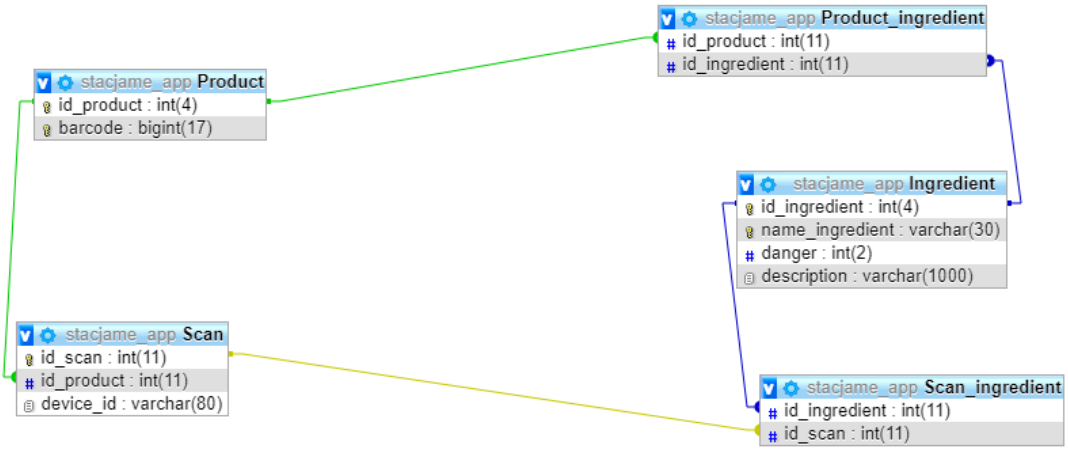
Aby rozpocząć sprawdzanie produktu należy kliknąć **rozpocznij skanowanie** i zeskanować kod kreskowy produktu. Następnie otrzymamy odpowiedź zwrotną od aplikacji (wynik skanu lub zostaniemy poproszeni o zeskanowanie składników produktu, gdyż produkt

nie znajduje się w bazie). Z menu głównego możemy przejść do listy składników, gdzie możemy zaznaczyć - poprzez dłuższe przytrzymanie na nim palca – składniki, które chcemy wyświetlać na początku listy wyników. Ponadto przez kliknięcie w wybrany składnik mamy możliwość podglądu krótkiego opisu.

Jeżeli nie ma kodu kreskowego, to system zapisuje go do bazy i otwiera nam skaner składników. Wtedy przez 15 sekund mamy możliwość skanowania substancji z etykiety. Po 4 skanach etykieta jest tworzona przez skrypt PHP, który wybiera te składniki, które się powtórzyły przynajmniej 2 razy i zapisuje je do gotowej etykiety. Każde urządzenie może dokonać skanu konkretnego produktu **tylko raz,** co chroni nas przed zaśmieceniem bazy oraz wprowadzaniu nieprawdziwych danych.

3. Działanie oraz wygląd aplikacji.

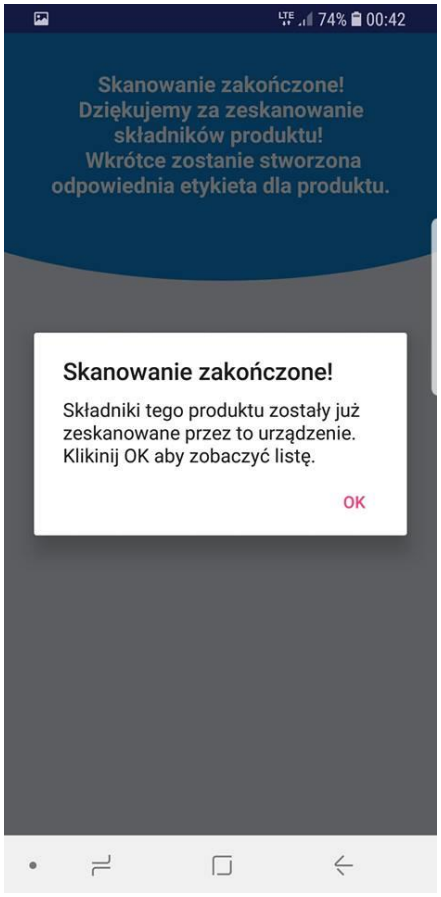
Schemat bazy danych:



Główne menu aplikacji:

Poniżej pokazany jest przykładowy skan kodów kreskowych:

Dalej przykładowy komunikat po zeskanowaniu kodu kreskowego:



Oraz lista składników z nazwą, opisem oraz kategorią:

4. Wybrane fragmenty kodu oraz ich opis.

Poniższy kod odpowiedzialny jest za połączenie z bazą danych oraz sprawdzenie, czy kod kreskowy ma przypisane składniki:

```
StringRequest stringRequest = new StringRequest(Request.Method.POST,
        Constants.URL_CHECKIFUSEDBARCODE,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {

                try {

                    JSONObject jsonObject = new JSONObject(response);
                    String temp = jsonObject.getString( name: "exist");
                    if (temp.contains("true")) inDb=true;

                    if (inDb){
                        Intent intent = new Intent( packageContext: BarcodeActivity.this,ResultActivity.class);
                        finish();
                        startActivity(intent);
                        //finish();
                        System.out.println(barcode);
                    } else if (!inDb) {
                        Intent intent = new Intent( packageContext: BarcodeActivity.this,MainActivity.class);
                        finish();
                        startActivity(intent);

                    }



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


            }
        },
        new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {

            }
        }){
    protected Map<String,String> getParams () throws AuthFailureError {
        Map<String,String> params = new HashMap<>();
        params.put( k: "barcode", barcode);
        return params;
    }
};

RequestHandler.getInstance(this).addToRequestQueue(stringRequest);
```

Kolejny fragment pokazuje dodanie składnika do listy ulubionych poprzez chwilowe przytrzymanie na nim palca(metoda onItemLongClick):

```
ingredientsList.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> arg0, View arg1,
                                   int pos, long id) {

        boolean add = true;
        if (datasList.contains(rowItems.get(pos).getId())){

                db.deleteIngredient(rowItems.get(pos).getId());

                add = false;

                Intent intent = new Intent( packageContext: ListActivity.this,ListActivity.class);
                finish();
                startActivity(intent);
        }

        if (add){
            db.addIngredient(rowItems.get(pos).getId());

            Intent intent = new Intent( packageContext: ListActivity.this,ListActivity.class);
            finish();
            startActivity(intent);
        }

        return true;
    }
});
```

Dalej parsowanie obiektu Json-a :

```
public static Product fromJson(JSONObject jsonObject) {
    Product b = new Product();
    // Deserialize json into object fields
    try {
        b.setId_product(jsonObject.getInt( name: "id_product"));
        b.setBarcode(jsonObject.getString( name: "barcode"));

    } catch (JSONException e) {
        e.printStackTrace();
        return null;
    }
    // Return new object
    return b;
}
```

Oraz otrzymywanie listy z Json-a:

```
public static ArrayList<Product> fromJsonList(JSONArray jsonArray) {
    JSONObject productsJson;
    ArrayList<Product> products = new ArrayList<~>(jsonArray.length());
    // Process each result in json array, decode and convert to business object
    for (int i=0; i < jsonArray.length(); i++) {
        try {
            productsJson = jsonArray.getJSONObject(i);
        } catch (Exception e) {
            e.printStackTrace();
            continue;
        }

        Product prod = Product.fromJson(productsJson);
        if (prod != null) {
            products.add(prod);
        }
    }

    return products;
}
```

Kolejne fragmenty to kod PHP, poniższy screen pokazuje w jaki sposób sprawdzamy, czy barcode posiada przypisane składniki:

```
function checkIfHasIngr($barcode){
    $stmt = $this->con->prepare("SELECT Product_ingredient.id_product FROM Product_ingredient,Product WHERE Product_ingredient.id_product=Product.id_product AND Product.barcode=?");
    $stmt->bind_param("s",$barcode);
    $stmt->execute();
    $stmt->store_result();
    return $stmt->num_rows>0;
}
```

Następny warty uwagi fragment pokazuje w jaki sposób odbywa się dodawanie nowego kodu kreskowego:

```php
function addProductIngredient($id_ingredient, $id_product){
    $stmt = $this->con->prepare("INSERT INTO Product_ingredient(id_ingredient,id_product) VALUES (?,?)");
    $stmt->bind_param("ss",$id_ingredient,$id_product);

    if($stmt->execute()){

        return true;

    } else {
        return false;
    }
}
```

Oraz pokazanie w jaki sposób wywołuje się funkcję dodającą nowy kod kreskowy:

```php
<?php

    require_once '../includes/DbOperations.php';
    $response = array();

    if($_SERVER['REQUEST_METHOD']=='POST'){

        if (
            isset($_POST['barcode'])){

                $db = new DbOperations();
                if ($db->createProduct(
                $_POST['barcode']
                )){
                    $response['error'] = false;
                    $response['message'] = "Product added successfully";
                }else {
                    $response['error'] = true;
                    $response['message'] = "Some error occurred please try again";

                }


            }else{
                $response['error'] = true;
                $response['message'] = "Required fields are missing";
            }

    } else {

        $response['error'] = true;
        $response['message'] = "Invalid Request";

    }

    echo json_encode($response);


?>
```

Ostatni fragment pokazuje przypisanie do widoku listy obrazków, nazwy składnika oraz opisu:

```
@Override
public View getView(int i, View convertView, ViewGroup parent) {


    View v = convertView;

    if (v == null) {
        LayoutInflater vi;
        vi = LayoutInflater.from(context);
        v = vi.inflate(R.layout.list_ingredients,  root: null);
    }

    //Item p = getItem(position);
    RowIngredient p = (RowIngredient) getItem(i);

    if (p != null) {
        TextView tt1 = (TextView) v.findViewById(R.id.tvName);
        TextView tt2 = (TextView) v.findViewById(R.id.tvDesc);
        ImageView tt3 = (ImageView) v.findViewById(R.id.imageEmoticon);
        ImageView tt4 = (ImageView) v.findViewById(R.id.imageStar);

        if (tt1 != null) {
            tt1.setText(p.getIngredientName());
        }

        if (tt2 != null) {
            tt2.setText(p.getIngredientDesc());
        }

        if (tt3 != null) {
            tt3.setImageResource(p.getImageEmoticon());
        }

        if (tt4 != null) {
            tt4.setImageResource(p.getImageStar());
        }
    }

    return v;
```