

PANTRY TRACKER

Vai a fare la spesa e non ricordi se hai già del pane in cucina? La tua dispensa è un disastro e non sai quali prodotti contiene?

A tutto questo e di più ci pensa PANTRY TRACKER.

PANTRY TRACKER è una applicazione mobile sviluppata per dispositivi Android.

Questa applicazione è stata progettata per rendere la vita di tutti i giorni un po' più semplice, permettendo all'utente di tenere sott'occhio i prodotti all'interno della propria cucina.

COSA SI PUÒ FARE CON PANTRY TRACKER?

L'utilità di PANTRY TRACKER risiede nel poter gestire e controllare comodamente dal proprio cellulare i prodotti all'interno della propria cucina.

L'utente può quindi:

- Cercare un prodotto e aggiungerlo all'interno della propria dispensa virtuale
- Creare un nuovo prodotto (se non presente all'interno dell'applicazione) aggiungendo:
 - Nome del prodotto
 - Descrizione del prodotto – che può essere la marca proprietaria o una piccola didascalia che descriva l'articolo
 - Foto del prodotto – facoltativo. Per aggiungerla è necessario inserire il link della foto
 - **Data di scadenza del prodotto**
- Dare una votazione al prodotto cercato:
 - 1 - Se corrispondente all'articolo posseduto dall'utente, si dà una votazione di uno
 - 0 - Se non corrispondente, l'utente non ha bisogno di votare
- Eliminare un prodotto dalla propria dispensa virtuale

COME FUNZIONA PANTRY TRACKER?

La funzionalità dell'applicazione ruota attorno all'utilizzo di codici a barre.

Un codice a barre è una stringa numerica di lunghezza variabile che identifica uno e un solo prodotto.

Grazie al codice a barre è possibile identificare univocamente un articolo.

PANTRY TRACKER permette all'utente di inserire un codice barre che dovrà identificare un prodotto. Se questo è presente all'interno del database dell'applicazione, può essere aggiunto alla dispensa virtuale. Se non presente sarà responsabilità dell'utente creare un nuovo prodotto con il codice inserito.

IMPLEMENTAZIONE CODICE A BARRE

Un prodotto con il suo codice a barre è registrato all'interno di un database remoto che presenta un'interfaccia WebService. Quest'ultimo risponde a chiamate HTTP REST

È possibile far uso di tre operazioni:

1- GET PRODUCTS BY BARCODE

Una volta inserito il codice viene fatta richiesta al WebService di fornire i dettagli del prodotto. Questo risponderà fornendo una lista di prodotti con quel codice

2- POST PRODUCT DETAILS

Se il prodotto non fa parte della lista o la lista è vuota, l'utente deve manualmente inserire i dettagli del prodotto in modo che si crei una new entry all'interno del database remoto

3- POST PRODUCT PREFERENCE

Scelto il prodotto, l'utente può notificare il WebServer che il prodotto rispetta i criteri di ricerca.

Per implementare le chiamate al WebServer è stata utilizzata la libreria Volley

1- GET PRODUCTS BY BARCODE

```
public void getProducts(String barcode, String authorization) {  
  
    RequestQueue requestQueue= Volley.newRequestQueue(ProductActivity.this);  
  
    String URL= "https://lam21.iot-prism-lab.cs.unibo.it/products?barcode="+barcode;  
  
    StringRequest stringRequest= new StringRequest(Request.Method.GET, URL, new Response.Listener<String>() {  
        @Override  
        public void onResponse(String response) {  
            Intent intent = new Intent(ProductActivity.this, ProductListActivity.class);  
            intent.putExtra("response", response);  
        }  
    })
```

```
        intent.putExtra("barcode", barcode);
        intent.putExtra("auth", authorization);
        intent.putExtra("user", user);
        ProductActivity.this.startActivity(intent);
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Toast.makeText(getApplicationContext(), "C'è qualche problema, ritenta più tardi",
        Toast.LENGTH_SHORT).show();

    }
}){
    public Map<String, String> getHeaders(){
        Map<String, String> data= new HashMap<String, String>();
        data.put("Authorization", "Bearer " + authorization);
        return data;
    }
};

requestQueue.add(stringRequest);
}
```

Viene utilizzata una RequestQueue che permette di inserire una nuova richiesta tramite Volley. Viene poi utilizzata una StringRequest che prende il metodo della richiesta, l'URL del WebServer, la risposta alla richiesta e una risposta in caso di errore.

La risposta alla richiesta utilizza al suo interno degli intent che passano dei dati attraverso "intent.putExtra". Questi dati sono rispettivamente stringhe per la risposta, per il codice a barre, per il token di accesso (auth / authorization) e per l'identificazione dell'utente.

Infine viene utilizzato un metodo per ottenere l'header della risposta e alla RequestQueue viene aggiunta la StringRequest appena creata.

2- POST PRODUCT DETAILS

```
private void postNewProduct(String token, String barcode, String name, String description, String authentication) {

    RequestQueue requestQueue= Volley.newRequestQueue(AddProductActivity.this);

    String URL= "https://lam21.iot-prism-lab.cs.unibo.it/products";

    JSONObject object= new JSONObject();

    try {
        object.put("test", false);
        object.put("token", token);
        object.put("name", name);
        object.put("description", description);
        object.put("barcode", barcode);
        if (TextUtils.isEmpty(insImage.getText())){
            object.put("img", null);
        }else{
            object.put("img", image);
        }
    }

    } catch (JSONException exception) {
```

```
exception.printStackTrace();
}

JsonObjectRequest request= new JsonObjectRequest(Request.Method.POST, URL, object,
    new Response.Listener<JSONObject>() {
        @Override
        public void onResponse(JSONObject response) {
            Toast.makeText(getApplicationContext(), "Prodotto inserito", Toast.LENGTH_SHORT).show();
            finish();
        }
    },
    new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText(getApplicationContext(), "C'è qualcosa che non va, ritenta più tardi",
                Toast.LENGTH_SHORT).show();
        }
    }
    ){
    public Map<String, String> getHeaders(){
        Map<String, String> data= new HashMap<String, String>();
        data.put("Authorization", "Bearer " + authentication);

        return data;
    }
    };

requestQueue.add(request);
}
```

A differenza di GET PRODUCTS BY BARCODE qui si utilizza un JSONObject in cui vengono inseriti i dati del prodotto, ovvero access token, nome, descrizione, codice a barre e immagine del prodotto. Dopodiché si fa una JSONObject request inserendo il metodo che in questo caso è POST, l'URL del metodo HTTP, l'oggetto JSON appena creato, la risposta in caso di successo e la risposta in caso di errore. Infine si richiedono gli Headers.

3- POST PRODUCT PREFERENCE

```
private void postPreference(String token, String id, String authorization){

    RequestQueue Readers= Volley.newRequestQueue(ChosenProductActivity.this);

    String URL= "https://lam21.iot-prism-lab.cs.unibo.it/votes";

    int voto= 1;

    JSONObject object= new JSONObject();
    try {
        object.put("token", token);
        object.put("rating", voto);
        object.put("productId", id);
    } catch (JSONException e) {
        e.printStackTrace();
    }

    JsonObjectRequest request= new JsonObjectRequest(Request.Method.POST, URL, object,
```

```
new Response.Listener<JSONObject>() {  
    @Override  
    public void onResponse(JSONObject response) {  
        Toast.makeText(getApplicationContext(), "Il tuo voto è stato inserito ", Toast.LENGTH_LONG).show();  
    }  
}, new Response.ErrorListener() {  
    @Override  
    public void onErrorResponse(VolleyError error) {  
        Toast.makeText(getApplicationContext(), "Errore, sicuro di non aver già votato?",  
Toast.LENGTH_SHORT).show();  
    }  
}){  
    public Map<String, String> getHeaders(){  
        Map<String, String> data= new HashMap<String, String>();  
        data.put("Authorization", "Bearer " + authorization);  
        return data;  
    }  
};  
requestQueue.add(request);  
}
```

IMPLEMENTAZIONE ALTRI METODI

4- REGISTER

```
private void registerUser(String username, String email, String password) {
    RequestQueue requestQueue= Volley.newRequestQueue(RegisterActivity.this);
    String URL = "https://lam21.iot-prism-lab.cs.unibo.it/users";
    StringRequest stringRequest= new StringRequest(com.android.volley.Request.Method.POST, URL, new
    Response.Listener<String>() {
        @Override
        public void onResponse(String response) {

            startActivity(new Intent(RegisterActivity.this, LoginActivity.class));

        }
    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            String message= "C'è qualcosa che non va, ritenta più tardi ...";
            Toast.makeText(getApplicationContext(), message, Toast.LENGTH_LONG).show();

        }
    }) {
        @Nullable
        @Override
        protected Map<String, String> getParams() {
            Map<String, String> data= new HashMap<String, String>();
            data.put("username", username);
            data.put("password", password);
            data.put("email", password);
            return data;
        }
    };
    requestQueue.add(stringRequest);
}
}:
```

5- LOGIN

```
public void loginUser(String mail, String password) {
    RequestQueue requestQueue= Volley.newRequestQueue(LoginActivity.this);
    String URL= "https://lam21.iot-prism-lab.cs.unibo.it/auth/login";
    StringRequest stringRequest= new StringRequest(Request.Method.POST, URL, new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            response = response.substring(16);
            response = response.substring(0, response.length() - 2);
            switchActivity(response);
        }
    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            String message= "C'è qualcosa che non va, ritenta più tardi ...";
            Toast.makeText(LoginActivity.this, message, Toast.LENGTH_LONG).show();
        }
    }) {
        @Nullable
        @Override
        protected Map<String, String> getParams() throws AuthFailureError {
            Map<String, String> data= new HashMap<String, String>();
            data.put("email", password);
            data.put("password", password);

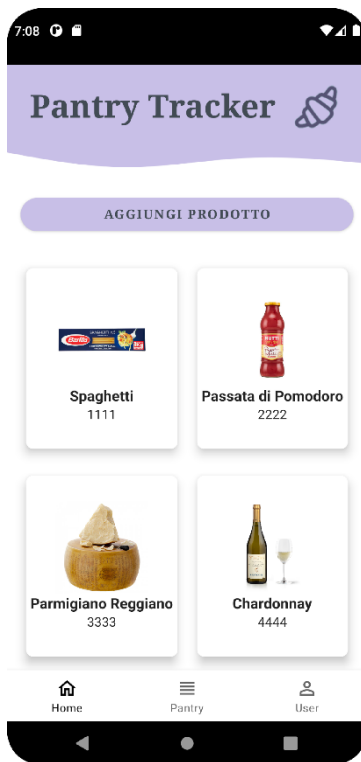
            return data;
        }
    };

    requestQueue.add(stringRequest);
}
```

COME È STRUTTURATA PANTRY TRACKER

Le varie parti di PANTRY TRACKER sono state divise tra più attività, ognuna con uno scopo preciso.

1- MAINACTIVITY



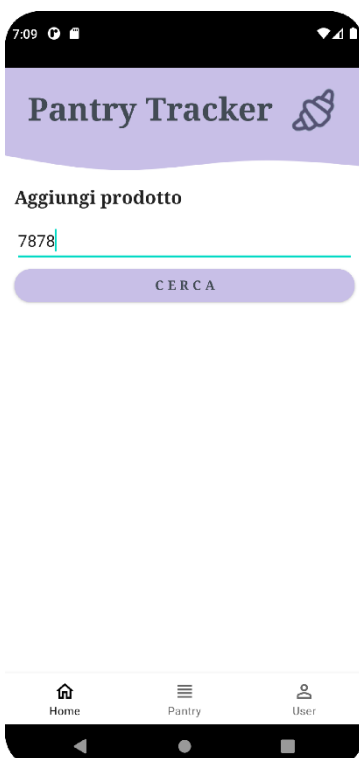
L'attività principale dell'applicazione.

Troviamo vari elementi, primo fra tutti un "Button" "AGGIUNGI PRODOTTO". Questo elemento reindirizzerà all'activity "ProductActivity".

A seguire vengono presentate quattro Cardview con all'interno dei prodotti espositivi. Una volta aver cliccato su uno dei prodotti si verrà reindirizzati alla pagina del prodotto selezionato "ChosenProductActivity".

Elemento comune a gran parte delle attività è la presenza di una "BottomNavbar" che è utile per spostarsi tra le varie attività.

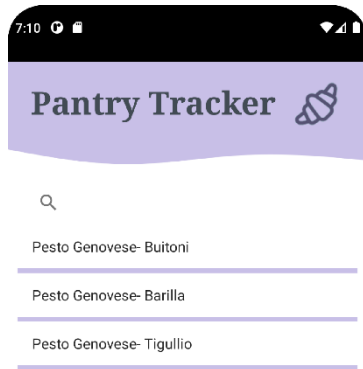
2- PRODUCTACTIVITY



All'interno di questa sezione è possibile inserire il codice a barre del prodotto. Il codice deve essere arbitrariamente numerico, ma può essere di lunghezza variabile.

Una volta inserito il codice a barre si andrà a effettuare la ricerca e si verrà reindirizzati all'attività contenente i risultati della ricerca – "ProductListActivity"

3- PRODUCTLISTACTIVITY



All' interno di questa attività vengono presentati i risultati della ricerca del codice a barre. Sono possibili due scenari:

- 1- La ricerca ha successo e viene presentata una lista contenente i prodotti con quel determinato codice a barre
- 2- La ricerca non ha successo e ci si presenta una schermata vuota.

Inoltre è possibile effettuare un'ulteriore ricerca all'interno di questa sezione, inserendo parole chiave per trovare il prodotto desiderato.

4- CHOSENPRODUCTACTIVITY



Viene presentato il prodotto scelto con:

1. Immagine del prodotto
2. Nome del prodotto
3. Descrizione del prodotto

È possibile effettuare tre azioni all'interno di questa schermata:

1. Aggiungere il prodotto alla propria dispensa virtuale
2. Votare in modo positivo il prodotto se rispetta i criteri di ricerca
3. Eliminare il prodotto – azione possibile solo se si hanno i permessi adeguati

5- ADDPRODUCTACTIVITY



7:11

Pantry Tracker

AGGIUNGI LE INFORMAZIONI DEL PRODOTTO

Nome

Descrizione

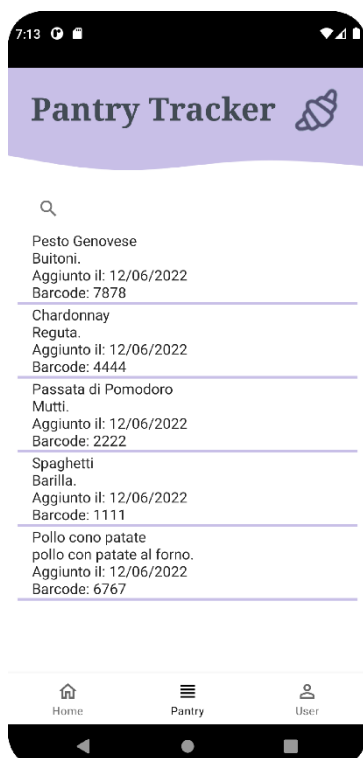
Link immagine

CREA

Nell'eventualità in cui il codice a barre inserito non dia i risultati sperati è possibile aggiungere autonomamente un prodotto. Questo prodotto non solo potrà essere inserito all'interno della propria dispensa virtuale, ma andrà a far parte della "Knowledge Base" condivisa dell'applicazione.

È necessario semplicemente inserire all'interno degli spazi appositi: nome, descrizione del prodotto. Eventualmente è possibile anche aggiungere il link di un'immagine relativa al prodotto – ma è completamente facoltativo.

6- PANTRYACTIVITY



7:13

Pantry Tracker

🔍

Pesto Genovese
Buitoni.
Aggiunto il: 12/06/2022
Barcode: 7878

Chardonnay
Reguta.
Aggiunto il: 12/06/2022
Barcode: 4444

Passata di Pomodoro
Mutti.
Aggiunto il: 12/06/2022
Barcode: 2222

Spaghetti
Barilla.
Aggiunto il: 12/06/2022
Barcode: 1111

Pollo con patate
pollo con patate al forno.
Aggiunto il: 12/06/2022
Barcode: 6767

Home Pantry User

PantryActivity presenta al suo interno la lista dei prodotti che sono stati aggiunti alla dispensa virtuale.

Ogni elemento della lista è caratterizzato dalla presenza del:

1. Nome del prodotto
2. Descrizione del prodotto
3. Data di aggiunta alla dispensa
4. Codice a barre di riferimento

Inoltre è possibile effettuare una ricerca all'interno della lista in modo tale da trovare facilmente l'articolo cercato.

STRUMENTI UTILIZZATI PER CREARE PANTRY TRACKER

ANDROID STUDIO

Android Studio è un ambiente di sviluppo integrato per lo sviluppo per la piattaforma Android.

JAVA

Java è un linguaggio di programmazione ad alto livello, orientato agli oggetti e a tipizzazione statica

VOLLEY

Volley è una libreria HTTP che rende più semplice e soprattutto più veloce il collegamento in rete delle applicazioni Android.

È stata utilizzata per effettuare le chiamate HTTP ai metodi dell'API

GLIDE

Glide è una libreria per il caricamento di immagini veloce ed efficiente per Android, incentrata sullo scorrimento fluido

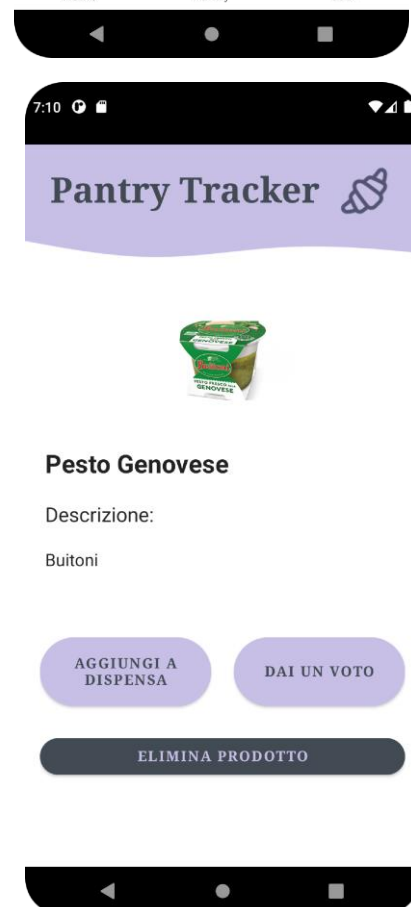
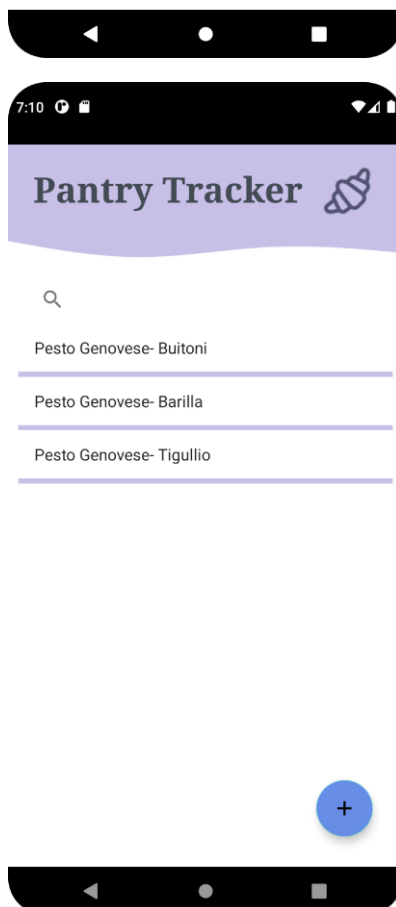
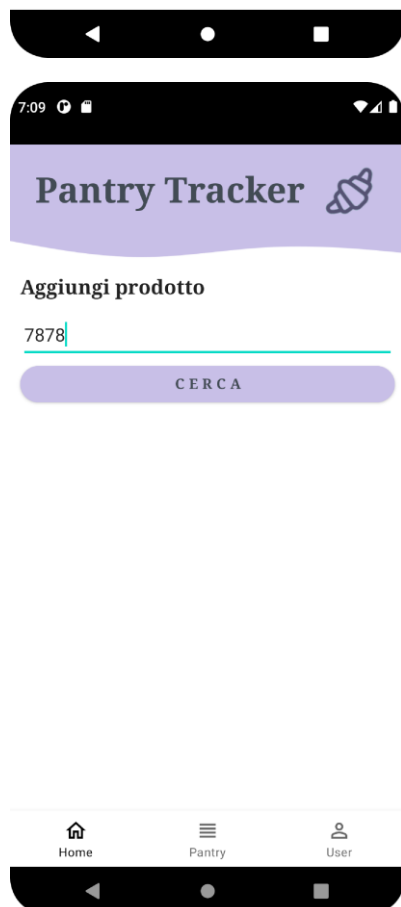
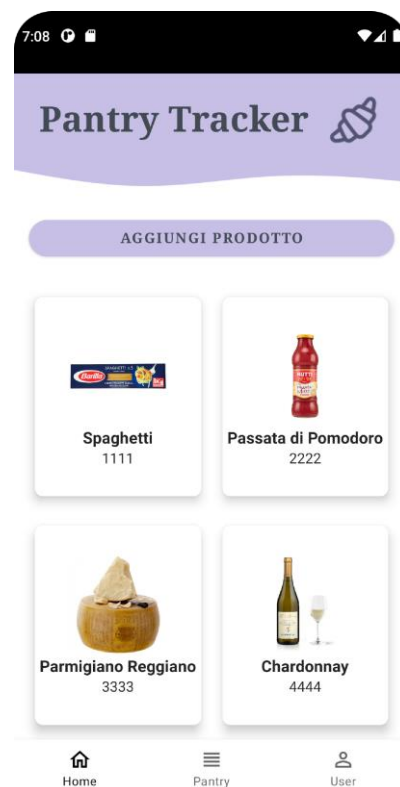
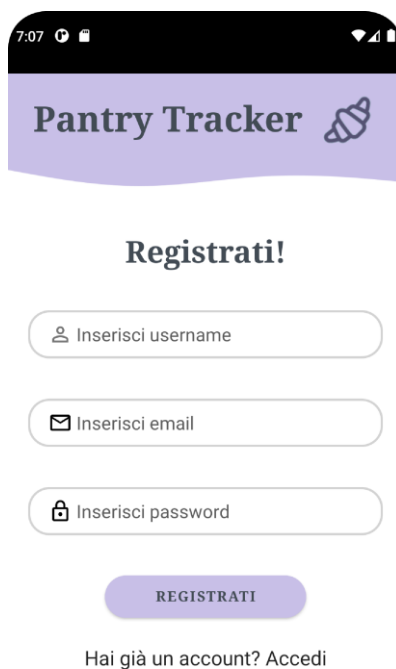
È stata utilizzata per caricare i link delle immagini dei prodotti di Pantry Tracker.

SQLITE

SQLite è un database che consente di salvare dati e informazioni di app Android sul file system del dispositivo.

È stata utilizzata per il salvataggio dei prodotti e la creazione della dispensa virtuale.

PANORAMICA DELL'APPLICAZIONE



7:11

Pantry Tracker

AGGIUNGI LE INFORMAZIONI DEL PRODOTTO

Nome

Descrizione

Link immagine

CREA

7:13

Pantry Tracker

🔍

Pesto Genovese
Buitoni.
Aggiunto il: 12/06/2022
Barcode: 7878

Chardonnay
Reguta.
Aggiunto il: 12/06/2022
Barcode: 4444

Passata di Pomodoro
Mutti.
Aggiunto il: 12/06/2022
Barcode: 2222

Spaghetti
Barilla.
Aggiunto il: 12/06/2022
Barcode: 1111

Pollo con patate
pollo con patate al forno.
Aggiunto il: 12/06/2022
Barcode: 6767

Home Pantry User

7:13

Pantry Tracker

LOGOUT

Home Pantry User