

P R O G E T T O   D I  
P R O G R A M M A Z I O N E  
D I   D I S P O S I T I V I  
M O B I L I

UNIVERSITA' DEGLI STUDI MILANO, BICOCCA  
A.A.2025/2026

# BinGo!

Gloria Invernizzi - 910243  
Elisabetta Locatelli - 914621  
Sara Maggioni - 909535  
Beatrice Pomi - 914386



# BinGo!



## INDICE

<b>BinGo!</b>	1
<b>Indice</b>	2
<b>Descrizione dell'applicazione</b>	3
<b>Mockup</b>	4
<b>Progettazione dell'interfaccia</b>	5
<b>Architettura</b>	6
<b>Diagramma Package</b>	7
<b>Analisi dei Requisiti</b>	8
<b>Diagramma dei Casi d'Uso</b>	9
<b>Caso d'Uso Formato dettagliato</b>	10
Registrazione e Login	10
Ricerca Prodotto	11
Aggiornamento Profilo Utente	12
<b>Diagramma di Navigazione</b>	14
<b>Funzionalità</b>	15
Welcome, Login e Registrazione	15
Home	16
Dove lo Butto?	15
Inserimento manuale	16
Scansione codice a barre	17
Risultato	18
Preferiti	18
Calendario	19
Famiglia	19
Profilo Utente	20
Impostazioni dell'account	21
Cambio email e password	22
<b>Web Services - Open Food Facts</b>	23
<b>Sviluppo Futuri</b>	24

# BinGo!



## Descrizione dell'applicazione

Questa guida fornisce una panoramica sull'utilizzo dell'applicazione Android **BinGo** per la gestione e lo smaltimento corretto dei rifiuti. L'app è progettata per aiutare gli utenti a comprendere come differenziare i rifiuti in modo efficace, riducendo l'impatto ambientale e promuovendo un **futuro più sostenibile**. L'applicazione agevola l'utente nella pianificazione dello smaltimento con la possibilità di inviare promemoria.

L'app BinGo è uno strumento prezioso per promuovere la **consapevolezza ambientale** e facilitare lo **smaltimento corretto dei rifiuti**. Utilizzando questa app, puoi contribuire attivamente alla salvaguardia del nostro pianeta e costruire un futuro più sostenibile per tutti. Scarica l'app oggi stesso e inizia a fare la differenza!

## Funzionalità Principali

- Guida alla differenziazione dei rifiuti: fornisce informazioni dettagliate su come smaltire correttamente vari tipi di rifiuti, identificando i vari prodotti tramite codice a barre.
- Notifiche e promemoria: Invia notifiche per ricordare agli utenti di smaltire i rifiuti nei giorni previsti, impostati manualmente dall'utente nel calendario.
- Interfaccia user-friendly: Design intuitivo e facile da usare per tutte le età.
- Supporto multilingue: Disponibile in più lingue per raggiungere un pubblico più ampio.
- Modalità offline-first: Salvataggio in locale delle informazioni per garantire l'utilizzo anche senza connessione internet.
- Modalità famiglia: Funzionalità per coinvolgere tutta la famiglia nel processo di smaltimento dei rifiuti.
- Supporto per codici a barre: Scansione dei codici a barre dei prodotti per ottenere informazioni sul corretto smaltimento o inserimento manuale.
- Personalizzazione: Opzioni per personalizzare l'aspetto e le funzionalità dell'app in base alle preferenze dell'utente.
- Modalità notturna: Tema scuro per un utilizzo più confortevole di notte.

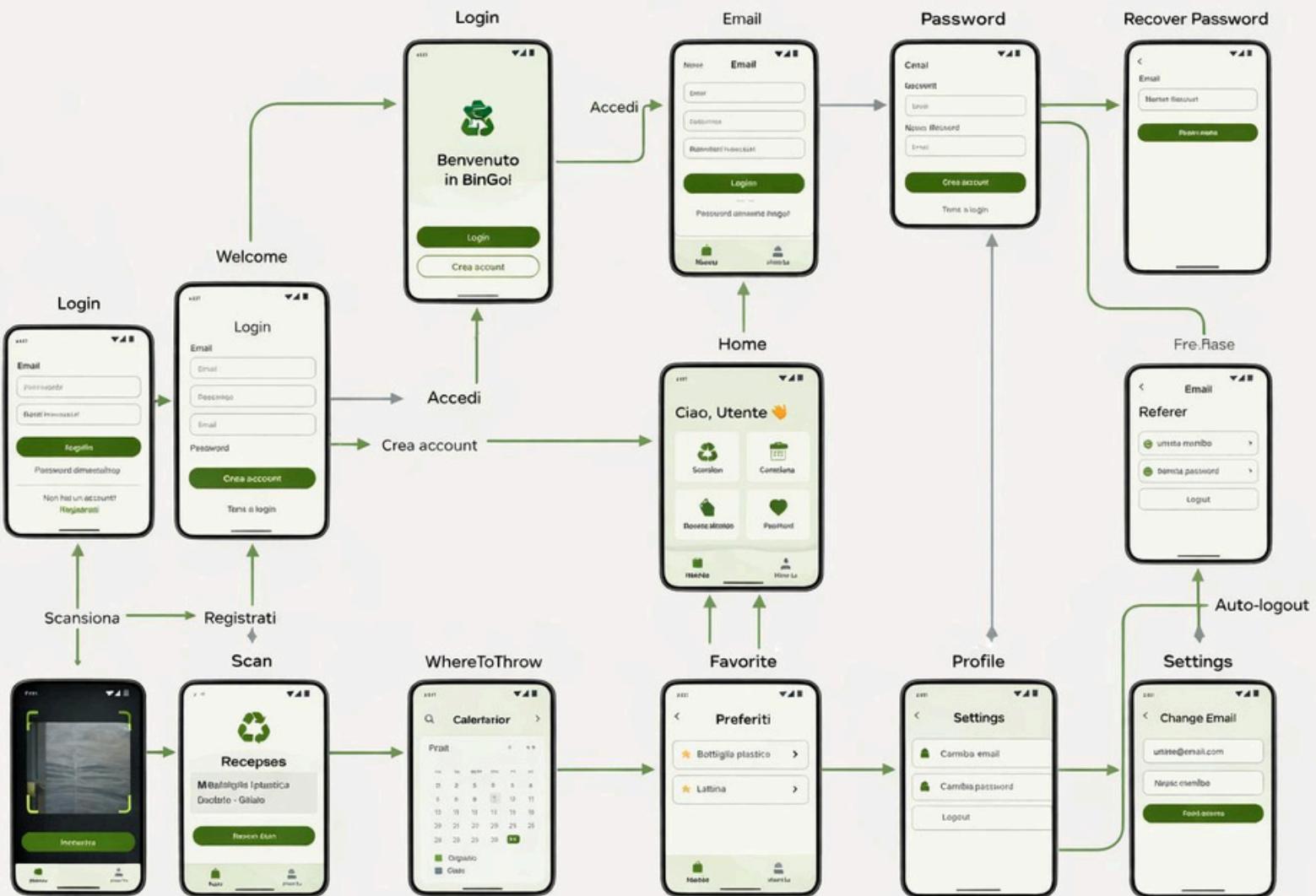
## Scelte implementative

- Linguaggio di programmazione: Java
- Ambiente di sviluppo: Android Studio
- Database: SQLite/ROOM/Firebase
- Librerie: CameraX e ML Kit per l'apertura della fotocamera e la scansione dei codici a barre, Retrofit per le chiamate API
- Material Design per l'interfaccia utente
- Architettura: MVVM (Model- View- ViewModel)
- Testing: JUnit, Espresso
- Strumenti di versionamento: Git/GitHub
- Gestione delle dipendenze: Gradle/Maven

# Come Utilizzare l'App

1. Download e Installazione: scarica l'app BinGo! da Google Play Store e installala sul tuo dispositivo Android.
2. Registrati o accedi al tuo account: salva il tuo calendario o unisciti al gruppo famiglia.
3. Riconoscimento del Rifiuto: utilizza la fotocamera o la ricerca manuale per cercare il rifiuto.

## Mockup



# BinGo!

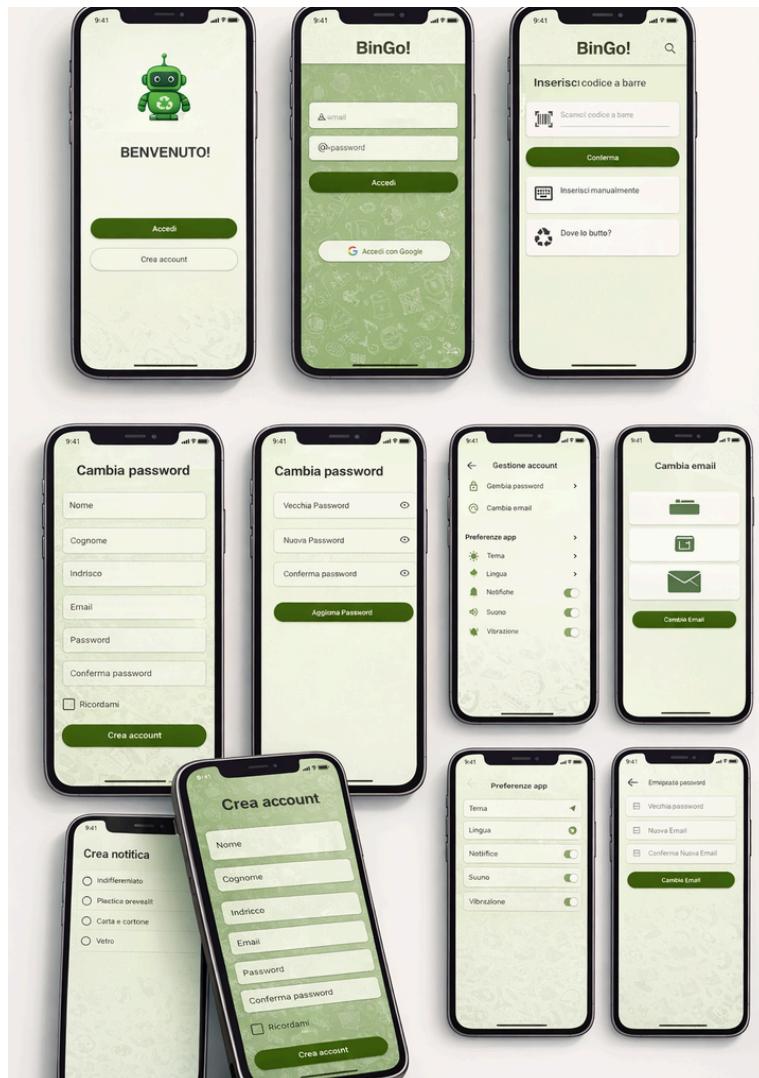


## Progettazione dell'interfaccia utente

L'interfaccia dell'app BinGo è stata progettata tramite wireframe low-fidelity, al fine di definire la struttura delle schermate e i flussi di navigazione prima dell'implementazione, permettendoci di concentrarci sull'usabilità e sulla semplicità dell'esperienza utente.

I wireframe evidenziano le principali funzionalità dell'applicazione: scansione dei codici a barre, consultazione delle informazioni di smaltimento e gestione del calendario di raccolta.

### Wireframe



# BinGo!

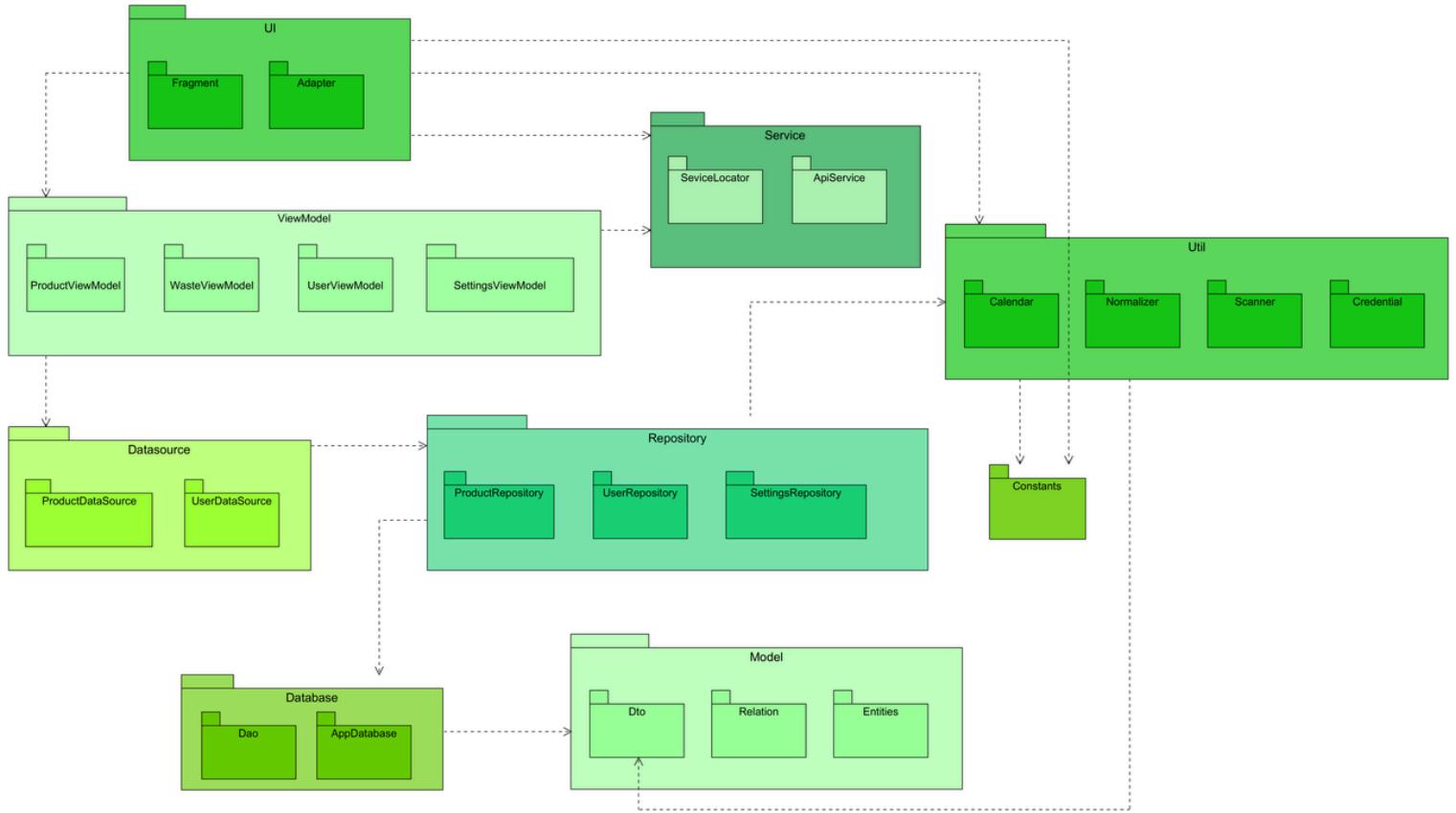


## Architettura

L'applicazione **BinGo!** è stata progettata seguendo un'architettura modulare, con l'obiettivo di garantire una chiara separazione delle responsabilità, una maggiore manutenibilità del codice e una migliore scalabilità dell'applicazione nel tempo.

La struttura architettonica adottata consente di isolare la logica di presentazione dall'interfaccia utente e dalla gestione dei dati, riducendo le dipendenze dirette tra i diversi componenti dell'applicazione.

### Diagramma dei package



Questa architettura garantisce **separazione delle responsabilità, testabilità, manutenibilità e scalabilità**.

# Pattern architetturale (MVVM)

L'applicazione adottato il pattern architetturale Model–View–ViewModel (MVVM) che consente di organizzare l'applicazione in tre componenti principali, ciascuno con responsabilità ben definite:

- Model: rappresenta i dati dell'applicazione e le entità di dominio;
- View: che si occupa esclusivamente della presentazione delle informazioni all'utente;
- ViewModel: che funge da intermediario tra View e Model, gestendo la logica applicativa e l'esposizione dei dati all'interfaccia.

## Model

Il livello Model comprende le classi che rappresentano i dati gestiti dall'applicazione, incluse le entità persistenti, gli **oggetti utilizzati per la comunicazione** con servizi remoti e le strutture dati necessarie al funzionamento dell'app (ad esempio User, Product, Packaging, Material, Notification...), usate per **modellare i dati** relativi all'utente, ai prodotti e alla gestione dei rifiuti. Il livello fornisce quindi una **rappresentazione coerente dei dati**, indipendente dall'interfaccia utente e dalla logica applicativa.

L'applicazione utilizza anche oggetti di trasferimento dati (DTO), impiegati per la comunicazione con i servizi remoti e per la **conversione dei dati ricevuti dalle API**. Questi oggetti non vengono persistiti localmente, ma svolgono un ruolo di supporto nello scambio di informazioni.

Sono presenti classi che definisco le **relazioni** tra le varie entità di dominio quali: ProductWithPackagings, PackagingWithTranslation.

## View

Il livello è costituito principalmente da Activity e Fragment, responsabili della **gestione dell'interfaccia grafica** e delle interazioni dell'utente. Questi componenti **osservano** i dati forniti dai ViewModel e **aggiornano** l'interfaccia di conseguenza, senza contenere logica di business complessa.

## UI

Il livello comprende i componenti responsabili dell'interfaccia utente e dell'interazione con l'utente. In BinGo! la UI è gestita dalla MainActivity, che coordina la navigazione tra i diversi Fragment dell'applicazione.

Tra i principali Fragment utilizzati vi sono WelcomeFragment, LoginFragment, RegisterFragment, HomeFragment, ScanFragment, ManualEntryFragment, WhereToThrowFragment, ProfileFragment, SettingsFragment e CalendarFragment, ciascuno dedicato a una specifica funzionalità.

I **Fragment** si occupano esclusivamente della **presentazione dei contenuti** e della gestione degli input dell'utente.

## ViewModel

Il livello rappresenta il **cuore della logica applicativa**: essa **recupera e aggiorna** i dati tramite i Repository, **elabora** le informazioni e le **rende disponibili** alla View in una forma pronta per essere visualizzata. In questo modo, i ViewModel fungono da **intermediari tra i Fragment e i Repository**, la View rimane indipendente dalle modalità di accesso ai dati e dalle logiche interne dell'applicazione e non vi sono accessi diretti alle sorgenti di dati.

Tra i principali ViewModel sono presenti: ProfileViewModel, SettingsViewModel, WasteViewModel, NotificationViewModel, ProductViewModel, ChangeEmailViewModel e ChangePasswordViewModel, ciascuno associato a una specifica funzionalità dell'applicazione.

## Database

L'applicazione segue un approccio **offline-first**, garantendo il funzionamento delle funzioni principali anche in assenza di connessione a Internet. La persistenza locale dei dati è affidata a **Room**, che memorizza le informazioni necessarie all'utilizzo dell'applicazione, come i dati dell'utente e contenuti relativi ai prodotti.

La gestione dell'autenticazione e delle credenziali è demandata a **Firebase**, che fornisce un sistema sicuro per l'accesso degli utenti. La **sincronizzazione** con i **servizi remoti** avviene solo quando la connessione è disponibile.

Per la memorizzazione delle impostazioni dell'applicazione, BinGo! utilizza inoltre **SharedPreferences**, impiegate per salvare preferenze come lingua, tema e configurazioni dell'utente.

## Repository

Il livello ha il compito di **gestire in modo centralizzato l'accesso ai dati**, fungendo da punto di collegamento tra le sorgenti di dati locali e remote e il resto dell'applicazione. Ogni Repository mantiene una chiara separazione delle responsabilità all'interno dell'architettura, evitando che ViewModel e View dipendano direttamente dai dettagli di implementazione.

Nell'applicazione troviamo: UserRepository, WasteRepository, NotificationRepository, ProductRepository e SettingsRepository, ciascuno, responsabile di una specifica area funzionale dell'applicazione, coordina l'accesso al database locale, ai servizi remoti e alle preferenze dell'utente.

## DataSource

Il livello gestisce le diverse sorgenti di dati dell'applicazione, sia locali che remote, per garantire la disponibilità e l'aggiornamento delle informazioni.

Nell'applicazione i dati dei prodotti e degli utenti vengono gestiti tramite DataSource specifici: quelli locali si occupano di conservare le informazioni necessarie per il corretto funzionamento dell'app anche offline (ProductLocalDataSource, UserLocalDataSource), mentre quelli remoti comunicano con servizi esterni per recuperare dati aggiornati (ProductRemoteDataSource, UserRemoteDataSource).

## Utilities

Il livello Utilities raccoglie classi di **supporto** utilizzate **trasversalmente** all'interno dell'applicazione BinGo!. Queste classi forniscono funzionalità comuni non legate a una specifica area dell'app, come il controllo della connettività (NetworkUtil), la gestione delle immagini e della fotocamera (PhotoHandler), il parsing delle informazioni ritornate dall'API e l'elaborazione di dati di supporto.

Le Utilities contribuiscono a **ridurre la duplicazione del codice** e a mantenere **separata la logica di supporto** dalle componenti principali dell'architettura.

## Services

Il livello rappresenta i componenti dell'applicazione che forniscono funzionalità di supporto, come l'accesso ai dati e la comunicazione con servizi esterni. I ruolo principale del **ServiceLocator** non è creare logica applicativa, ma **preparare e fornire componenti già pronti** all'uso, in particolare i repository e i servizi di rete. In questo modo, i ViewModel possono ottenere facilmente i dati di cui hanno bisogno senza occuparsi di come vengono creati o configurati i componenti, **mantenendo l'app modulare e facilmente manutenibile**. In BinGo! questo ruolo è svolto dal ServiceLocator, che mette a disposizione istanze di ProductRepository e di ProductAPIService, il servizio per comunicare con le API esterne e ottenere informazioni sui prodotti.



## Analisi dei requisiti

Il sistema deve permettere ai cittadini di identificare correttamente la tipologia di rifiuto e la relativa modalità di smaltimento tramite scansione o ricerca, e fornire informazioni sui giorni di raccolta.

Di seguito è riportato un elenco chiaro di **cosa** che il sistema deve fare (funzionali) e **come** deve comportarsi (non funzionali).

### Requisiti funzionali

- **Registrazione** via email/password.
- Il sistema deve mantenere la **sessione** utente se richiesto ("Remember me").
- **Login** tramite credenziali, **Logout** e **Recupero password**.
- **Navigazione** tra schermate.
- Scansione **barcode** tramite fotocamera o inserimento manuale e **visualizzazione** delle **informazioni** disponibili.
- Aggiunta prodotto ai **preferiti**, in modo da agevolare l'accesso e la visualizzazione, nonchè garantire il principio di **offline-first**. **Ricerca testuale** del prodotto per nome, marca, codice.
- Il sistema deve mostrare un **calendario** della raccolta differenziata.
- Visualizzazione e **modifica profilo** (password, indirizzo, ...).
- Impostazioni **notifiche** e **permessi**.

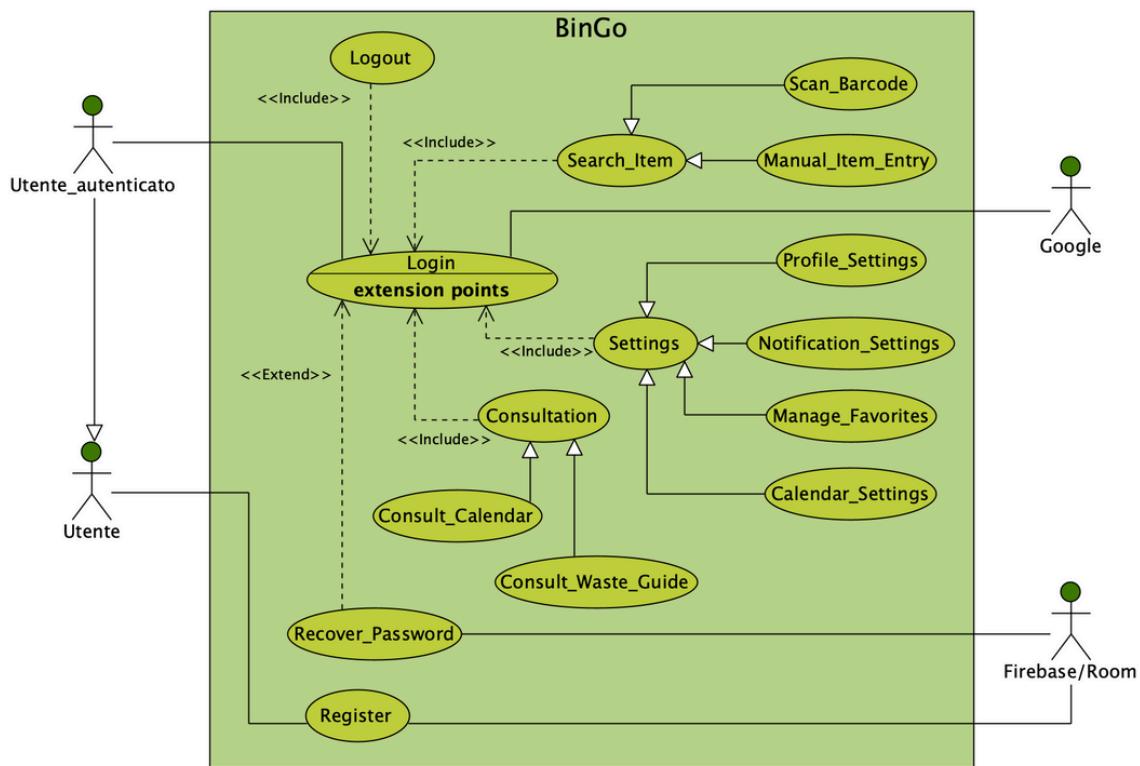
### Requisiti non funzionali

- **Sicurezza**: ID univoco generato automaticamente dalla console Firebase e uso di OAuth per Login con **Google**.
- **Prestazioni**: tempo risposta UI < 2s per query locali e < 5s per richieste remote.
- **Disponibilità**: funzionamento offline per dati cached (es. preferiti).
- **Performance**: riconoscimento del codice a barre deve avvenire in meno di 2 sec.
- **Usabilità**: supporto per accessibilità, testi in italiano, interfaccia intuitiva (sono sufficienti "pochi click" per utilizzare le funzionalità principali)
- **Compatibilità**: Android 8.0+ e utilizzo di librerie compatibili.
- **Privacy**: conformità GDPR per dati personali e Google/Firebase.

# BinGo!



## Diagramma dei Casi d'Uso



- **Registrazione**: attore non autenticato si registra creando un nuovo account con email/password (o Google) inserendo dati personali: nome, cognome, indirizzo, email, password.
- **Login**: Utente registrato effettua accesso tramite email/password o Google Sign-In per accedere alle funzionalità principali.
- **Recupero Password**: L'attore richiede il ripristino della password dimenticata tramite email (risposta automatica gestita da Firebase).
- **Scansione codice a barre**: Scansiona barcode e visualizza risultato.
- **Inserimento manuale**: L'utente inserisce manualmente il codice o il nome di un rifiuto.
- **Visualizza Home**: Accesso alla schermata principale con navigation a schede.
- **Consultazione Calendario**: L'utente visualizza i giorni di raccolta dei rifiuti.
- **Consulta guida di smaltimento "Dove lo butto?"**: L'utente consulta una guida o una lista per categorie di rifiuti.
- **Gestione Profilo**: L'utente visualizza e modifica i propri dati (foto, indirizzo, ...) o cambia la password.
- **Gestione Preferiti**: L'utente salva i risultati delle scansioni frequenti per un accesso rapido (aggiunge/rimuove prodotti dai preferiti).
- **Impostazioni/Notifiche**: Gestisce preferenze e logout.
- **Consultazione preferiti**: L'utente visualizza l'elenco dei preferiti, filtra la lista tramite query di ricerca, accede agevolmente ai prodotti contrassegnati come preferiti senza doverli riscansionare.



**BinGo!**



## Casi d'uso formato dettagliato

Caso d'uso	REGISTRAZIONE UTENTE
Attore Primario	Utente non registrato
Pre-condizioni	L'utente, passando dalla schermata Welcome, si trova ora nella schermata Register.
Post-condizioni	<ul style="list-style-type: none"><li>● Un nuovo account è creato su Firebase e nel DB locale (Room).</li><li>● L'utente può ora accedere alla schermata Home e usufruire delle funzionalità offerte da BinGo!</li></ul>
Scenario principale	<ol style="list-style-type: none"><li>1. L'utente seleziona "Create Account".</li><li>2. L'utente compila il form (Nome, Cognome, Indirizzo, Email, Password, Conferma Password).</li><li>3. L'utente accetta "Remember me" (opzionale).</li><li>4. Il sistema valida i dati (email valida, password &gt; 6 caratteri, corrispondenza password).</li><li>5. Il sistema registra l'utente su Firebase Auth.</li><li>6. Il sistema salva i dettagli utente nel database locale.</li></ol>
Scenari alternativi	<ul style="list-style-type: none"><li>● Email già già registrata.</li><li>● Dati mancanti.</li><li>● Connessione a Internet assente.</li><li>● Le password inserite non coincidono.</li><li>● Formato di email/password non valido.</li></ul>



**BinGo!**



## Casi d'uso formato dettagliato

Caso d'uso	LOGIN
Attore Primario	Utente registrato
Pre-condizioni	L'utente è già registrato e si trova nella schermata di login.
Post-condizioni	<ul style="list-style-type: none"><li>• L'utente può ora accedere alla schermata Home e usufruire delle funzionalità offerte da BinGo!</li></ul>
Scenario principale	<ol style="list-style-type: none"><li>1. L'utente inserisce Email e Password.</li><li>2. Il sistema verifica le credenziali.</li><li>3. In caso di successo, il sistema reindirizza alla Home.</li></ol>
Scenari alternativi	<ol style="list-style-type: none"><li>1. L'utente preme "Login with Google".</li><li>2. Il sistema mostra il selettore account Google.</li><li>3. L'utente seleziona l'account.</li><li>4. Il sistema autentica e crea l'utente nel DB locale se non esiste.</li></ol>



**BinGo!**



## Casi d'uso formato dettagliato

Caso d'uso	RICERCA PRODOTTO
Attore Primario	Utente autenticato
Pre-condizioni	L'utente dopo essersi autenticato vuole sapere come smaltire un prodotto.
Post-condizioni	<ul style="list-style-type: none"><li>● Il prodotto viene trovato nel database di open food facts (o Open Products facts)</li><li>● L'utente viene rimandato alla pagina di dettaglio del prodotto dalla quale può vedere tutte le informazioni relative allo smaltimento</li></ul>
Scenario principale	<ol style="list-style-type: none"><li>1. Dalla Home l'utente vuole inserire il codice a barre<ol style="list-style-type: none"><li>a. L'utente decide di inserire il codice a barre manualmente<ol style="list-style-type: none"><li>i. Digita nella barra di testo le cifre</li><li>ii. Clicca il bottone per avviare la ricerca</li><li>iii. L'applicazione avvia la richiesta all'API</li></ol></li><li>b. L'utente decide di scansionare il codice a barre<ol style="list-style-type: none"><li>i. Clicca su "Scansion codice a barre"</li><li>ii. Vengono richiesti i permessi</li><li>iii. L'utente approva i permessi e si apre la fotocamera</li><li>iv. L'utente inquadra il codice a barre</li><li>v. L'applicazione riconosce il codice a barre e avvia la richiesta all'API</li></ol></li><li>2. La richiesta ritorna il JSON contenente le informazioni necessarie</li><li>3. L'utente viene reindirizzato alla pagina di risultato</li><li>4. L'utente visualizza il nome del prodotto e il brand</li><li>5. L'utente può leggere i vari materiali che compongono l'imballaggio</li><li>6. L'utente può contrassegnare il prodotto come preferito<ol style="list-style-type: none"><li>a. L'utente decide di mettere il prodotto tra i preferiti</li><li>b. L'applicazione aggiorna la lista dei preferiti</li><li>c. L'utente navigando nella sezione dei preferiti vedrà il nuovo prodotto</li><li>d. L'utente può scorrere la lista e filtrarla a piacere per cercare quanto necessario</li></ol></li></ol></li></ol>

# BinGo!



## Casi d'uso formato dettagliato

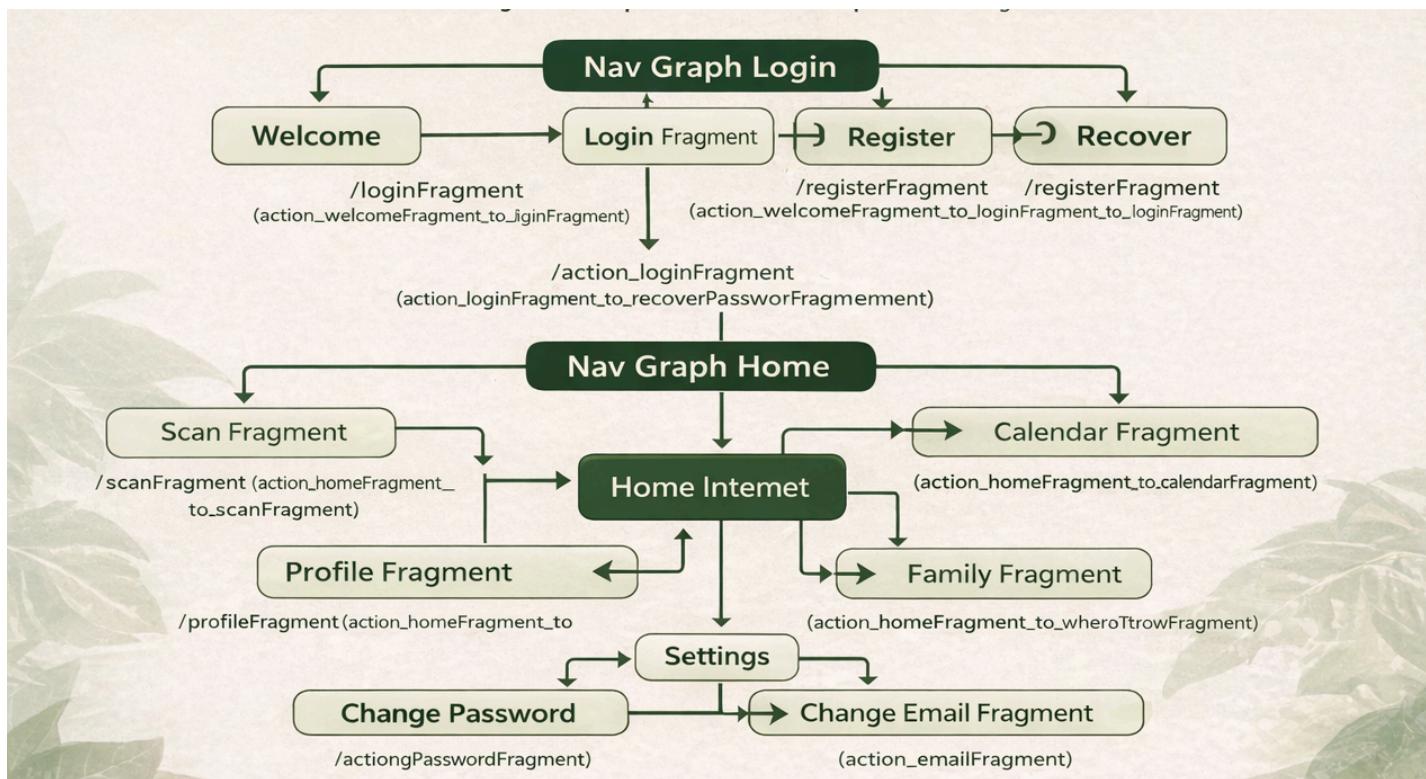
Caso d'uso	AGGIORNAMENTO PROFILO UTENTE
Attore Primario	Utente autenticato
Pre-condizioni	L'utente dopo essersi autenticato vuole aggiornare il suo profilo.
Post-condizioni	<ul style="list-style-type: none"><li>Le informazioni del profilo (nome, residenza e/o immagine del profilo) risultano aggiornate e salvate correttamente.</li><li>Le modifiche sono immediatamente visibili nella schermata del profilo.</li></ul>
Scenario principale	<ol style="list-style-type: none"><li>L'utente accede alla sezione Profilo tramite la barra di navigazione inferiore.</li><li>L'applicazione visualizza i dati principali dell'utente autenticato: nome, indirizzo email, residenza e immagine del profilo.</li><li>L'utente modifica il nome e/o la residenza utilizzando i campi di input presenti nella schermata.</li><li>L'utente, se lo desidera, seleziona l'opzione per modificare l'immagine del profilo.</li><li>L'applicazione consente all'utente di:<ol style="list-style-type: none"><li>selezionare un'immagine dalla galleria del dispositivo, oppure</li><li>scattare una nuova foto tramite la fotocamera.</li></ol></li><li>L'utente conferma le modifiche.</li><li>L'applicazione salva i dati aggiornati e mostra il profilo aggiornato.</li></ol>
Scenari alternativi	A1 – Modifica dell'indirizzo email non consentita <ul style="list-style-type: none"><li>Se l'utente tenta di modificare l'indirizzo email, l'applicazione non permette l'operazione, mostrando il campo come non modificabile.</li></ul> A3 – Annullamento delle modifiche <ul style="list-style-type: none"><li>L'utente abbandona la schermata del profilo senza salvare.</li><li>Nessuna modifica viene applicata ai dati esistenti.</li></ul>

# BinGo!



## Diagramma di navigazione

Il sistema utilizza il Navigation Component di Android Jetpack con due grafi nidificati:





**BinGo!**



## Funzionalità principali

Di seguito vengono descritte le principali funzionalità dell'applicazione **BinGo!**

### Schermata di Welcome, Login e Registrazione

All'avvio dell'applicazione viene visualizzata una schermata di **welcome**, che ha lo scopo di introdurre l'applicazione e consentire all'utente di accedere o creare un nuovo account.

La schermata di welcome presenta due azioni principali:

- l'accesso tramite credenziali esistenti;
- la registrazione di un nuovo account.

Se l'utente seleziona l'opzione di accesso, viene mostrata una schermata di **login** per effettuare l'autenticazione tramite inserimento delle credenziali o via Google. L'interfaccia **guida l'utente nella compilazione** dei campi richiesti e fornisce un **feedback visivo** in caso di dati mancanti o non validi.

È inoltre disponibile una funzionalità di **recupero password**, accessibile direttamente dalla schermata di login.

Nel caso in cui l'utente scelga di creare un nuovo account, l'applicazione presenta una schermata di **registrazione** che consente l'inserimento delle informazioni necessarie.

Una volta completato correttamente il processo di accesso o registrazione, l'utente viene **reindirizzato** automaticamente alla **schermata principale** dell'applicazione, dalla quale può accedere a tutte le funzionalità offerte da BinGo!.



# BinGo!



## Home e navigazione principale

Dopo aver effettuato correttamente l'accesso o la registrazione, l'utente viene reindirizzato alla **schermata Home**, che rappresenta il centro operativo dell'interazione con il sistema.

La schermata Home consente un accesso immediato alle principali funzionalità dell'applicazione:

- **Inserimento manuale** del codice a barre di un prodotto nella schermata principale e successiva visualizzazione dei risultati relativi al prodotto.
- **Scansione codice a barre**
- **Dove lo butto?** Permette all'utente di inserire manualmente i dati di un prodotto nel caso in cui la scansione non sia disponibile.
- **Gestione del nucleo famiglia.**

La schermata è organizzata in modo **chiaro e intuitivo**, con tutte le principali azioni immediatamente accessibili.

Nella parte inferiore è presente una barra di navigazione, che permette di spostarsi rapidamente tra le diverse sezioni dell'applicazione.

## “Dove lo butto”

In questa schermata vengono mostrate in modo chiaro e strutturato le informazioni relative al prodotto analizzato, con particolare attenzione alle **indicazioni sul corretto smaltimento** dei materiali che lo compongono.

La lista di prodotti e relative informazioni può essere filtrata attraverso la ricerca testuale oppure selezionando la macrocategoria sulla tipologia di materiale.



# BinGo!



## Inserimento manuale

Questa schermata permette di utilizzare l'applicazione anche quando, per qualsiasi motivo la scansione dalla fotocamera non è disponibile.

Una volta inserito il codice, l'app elabora le informazioni necessarie e mostra automaticamente la schermata dei risultati, indicando le corrette modalità di smaltimento del prodotto o del materiale.

L'inserimento manuale è direttamente visibile nella Home perché rappresenta una funzionalità principale dell'applicazione, immediatamente accessibile all'utente senza passaggi aggiuntivi.

## Scansione del codice a barre

Questa schermata utilizza la **fotocamera del dispositivo** per l'acquisizione del codice a barre del prodotto.

Al primo utilizzo della funzionalità, l'applicazione richiede all'utente il **permesso di accesso alla fotocamera**. Nel caso in cui il permesso venga negato, l'utente viene informato dell'impossibilità di utilizzare la scansione e invitato a concedere l'autorizzazione per proseguire.

Una volta che il codice a barre viene riconosciuto, l'applicazione avvia automaticamente il recupero delle informazioni associate al prodotto. Durante questa operazione viene visualizzato un indicatore di caricamento, che segnala all'utente l'elaborazione in corso.

I dati del prodotto vengono ottenuti tramite **servizi remoti**; qualora la **connessione a Internet non fosse disponibile**, l'applicazione tenta di **recuperare** eventuali informazioni già presenti **localmente**. Se non sono disponibili dati sufficienti, l'utente viene notificato dell'impossibilità di completare la scansione.

Al termine dell'elaborazione, l'utente viene reindirizzato alla schermata dei risultati, nella quale sono visualizzate le informazioni relative al prodotto e le indicazioni per il corretto smaltimento dei materiali.

L'intero flusso di scansione è progettato per minimizzare i tempi di attesa e garantire un'esperienza utente fluida e intuitiva.

# BinGo!



## Risultato

Una volta rilevato il codice a barre viene attivato il ProductRepository affinché restituisca il prodotto correlato. Le informazioni relative al prodotto possono essere prese o dal **web service remoto** tramite richiesta API oppure localmente con una **query a database locale** con Room.

Una volta rilevato il prodotto tutte le informazioni quali nome, brand, **imballaggio** vengono mostrate dal ResultFragment. Le informazioni vengono **filtrate** e visualizzate **in base alla lingua** selezionata dall'utente nelle impostazioni.

Dalla schermata di risultato è possibile **inserire** e **rimuovere** un prodotto dai **preferiti**. Quando un prodotto viene messo nei preferiti ProductRepository lo salva a database.

Tutti i prodotti che sono stati **almeno una volta inseriti nei preferiti** vengono **memorizzati localmente** e rimangono disponibili finché l'applicazione è installata sul dispositivo. Questo **insieme di prodotti** costituisce la base per un approccio **offline-first**, garantendo l'accesso alle informazioni anche in assenza di connessione Internet.

## Preferiti

Un'altra sezione facilmente accessibile dalla Home, tramite la barra di navigazione inferiore, è la lista dei preferiti, nella quale ogni elemento **mostra i dettagli di un prodotto corredati dalla relativa immagine**. Ogni volta che un prodotto viene aggiunto o rimosso dai preferiti la lista viene aggiornata automaticamente.

Questa funzionalità semplifica notevolmente la ricerca delle informazioni, consentendo all'utente di **selezionare rapidamente** un prodotto senza dover inserire manualmente o scansionare il codice a barre. La selezione è ulteriormente **agevolata** dalla presenza di una **barra di ricerca** che permette di **filtrare l'elenco** in base al testo inserito.

La combinazione tra immagini, nomi dei prodotti e ricerca testuale garantisce un'**esperienza di navigazione efficiente**, intuitiva e rapida, migliorando l'usabilità complessiva dell'applicazione.

# BinGo!



## Calendario

La funzionalità è accessibile direttamente dalla barra di navigazione inferiore e consente all'utente di visualizzare e gestire gli eventi e i promemoria legati allo smaltimento dei rifiuti.

All'apertura della schermata, viene mostrato un **calendario interattivo** in cui la data corrente è selezionata automaticamente, permettendo all'utente di avere subito sotto controllo le notifiche più rilevanti per il giorno corrente.

Se l'utente seleziona un giorno specifico, la lista sottostante viene aggiornata dinamicamente, mostrando tutte le notifiche associate a quella data. Ogni notifica contiene informazioni sul tipo di rifiuto, l'orario di raccolta e, se presente, la periodicità settimanale del promemoria.

Nella parte inferiore della schermata è presente un **pulsante di azione “+”**, che consente all'utente di **aggiungere manualmente una nuova notifica**.

L'applicazione permette di inserire le informazioni necessarie e di associare la notifica alla data selezionata, rendendola immediatamente visibile all'interno della lista.

## Famiglia

La funzionalità è accessibile direttamente dalla schermata Home e permette agli utenti di coordinare la gestione domestica dei rifiuti attraverso la collaborazione tra più membri. All'apertura della schermata, l'applicazione verifica automaticamente lo stato dell'utente: se non appartiene ancora a un nucleo, viene presentata un'interfaccia intuitiva che offre la possibilità di creare un nuovo gruppo o di unirsi a uno esistente.

Scegliendo di creare una famiglia, il sistema genera un codice identificativo univoco di otto caratteri, che funge da **chiave d'accesso** per gli altri componenti del nucleo. Se invece l'utente possiede già un codice fornito da un familiare, può inserirlo nell'apposito campo per essere integrato istantaneamente nel gruppo, aggiornando dinamicamente la propria configurazione locale.

Una volta all'interno di una famiglia, la schermata mostra il codice del gruppo e una lista aggiornata di tutti i partecipanti. Questa visualizzazione consente di avere sempre sotto controllo chi sta collaborando alla **gestione domestica**, facilitando la divisione dei compiti. Nella parte inferiore della schermata, l'utente ha la possibilità di abbandonare il gruppo in qualsiasi momento, eliminando l'associazione al nucleo e tornando alla visualizzazione individuale. Tale struttura trasforma l'applicazione in uno strumento di cooperazione sociale, rendendo la gestione del riciclo un'attività condivisa e meglio organizzata.



# BinGo!



## Profilo Utente

L'applicazione mette a disposizione dell'utente una sezione dedicata al **profilo**, accessibile tramite la barra di navigazione inferiore, che gli consente di **visualizzare e modificare alcune informazioni personali**, mantenendo separata la gestione delle credenziali di accesso, che viene invece demandata alla sezione delle impostazioni.

All'interno della schermata del profilo vengono mostrati i **dati principali dell'utente autenticato, tra cui nome, indirizzo email e residenza**.

Il nome e la residenza possono essere modificati direttamente dall'utente tramite i campi presenti nella schermata. L'interfaccia è progettata per rendere l'aggiornamento di questi dati semplice e immediato, consentendo di salvare le modifiche senza dover accedere a schermate aggiuntive.

L'indirizzo email, invece, è visibile ma non modificabile, poiché le credenziali di accesso sono gestite in una sezione separata dedicata alle impostazioni dell'account.

La schermata del profilo consente inoltre di **visualizzare e modificare l'immagine del profilo**.

L'utente può selezionare una foto dalla galleria o da altre applicazioni, oppure scattare una nuova foto tramite fotocamera.

Dalla schermata del profilo, è possibile accedere rapidamente alla **sezione delle impostazioni** tramite un'icona dedicata posizionata nella parte superiore dell'interfaccia. Questo consente di mantenere una chiara **separazione** tra la gestione dei **dati personali** e quella delle **impostazioni dell'account**.

È inoltre presente un pulsante dedicato al **logout**, visivamente separato dalle altre azioni, che permette all'utente di terminare la sessione mantenendo comunque i dati salvati localmente.



# BinGo!



## Impostazioni dell'account

La schermata, accessibile dalla sezione profilo, consente di gestire le **informazioni dell'account** e le **preferenze di utilizzo** dell'applicazione.

La **schermata** delle impostazioni è **suddivisa** in modo chiaro in **due aree** principali:

- **Gestione account**
- **Preferenze dell'app**

### Gestione account

Tale sezione raccoglie le funzionalità legate alla **modifica delle credenziali di accesso** dell'utente, nonché, informazioni sensibili dell'account, come l'indirizzo email e la password.

Le **singole operazioni** di modifica non avvengono direttamente nella schermata Impostazioni, ma sono **demandate a schermate specifiche**, così da rendere esplicita l'azione che l'utente sta per compiere e migliorare la sicurezza e la chiarezza dell'esperienza utente.

### Preferenze dell'app

Tale sezione consente all'utente di **personalizzare il comportamento dell'applicazione** in base alle proprie esigenze, configurando alcune impostazioni che riguardano l'esperienza d'uso complessiva.

Tra le preferenze configurabili rientrano:

- **Gestione delle notifiche:** consente di abilitare o disabilitare la ricezione di avvisi e promemoria.
- **Lingua dell'applicazione:** permette di modificare la lingua visualizzata nell'interfaccia.
- **Tema grafico:** consente di scegliere tra tema chiaro o scuro, personalizzando l'aspetto visivo dell'app.
- **Vibrazione:** possibilità di attivare o disattivare la vibrazione.
- **Suoni:** possibilità di attivare o disattivare i suoni dell'app.

Le **preferenze** vengono **salvate** e **applicate** durante l'utilizzo dell'app, garantendo un comportamento coerente nel tempo.

### Sicurezza e Privacy

Tale sezione consente all'utente di **eliminare** il proprio account. Prima di procedere, viene richiesta una conferma, trattandosi di un'azione irrevocabile.



## Gestione Account: Cambio Email e Password

All'interno della sezione **Gestione account** delle Impostazioni, l'applicazione offre due funzionalità fondamentali per mantenere aggiornate le credenziali dell'utente: il **cambio** dell'indirizzo **email** e il cambio della **password**. Queste operazioni, sebbene l'applicazione sia progettata per funzionare offline-first, richiedono una connessione per garantire la **sicurezza** e la **coerenza** con il server remoto.

### Cambio Email

Quando l'utente seleziona l'opzione di cambio email, viene reindirizzato a una schermata dedicata in cui è possibile inserire il nuovo indirizzo di posta elettronica.

Prima di confermare la modifica, l'applicazione richiede la **password attuale** per autenticare la richiesta, aumentando la sicurezza dell'operazione.

L'aggiornamento avviene prima sul server remoto (Firebase). In questa fase, viene inviata una **email di verifica** alla nuova casella: l'utente deve aprire il messaggio e seguire il link di conferma per completare la modifica. Solo dopo il successo dell'operazione remota, i dati vengono aggiornati localmente (Room e PrefsManager), garantendo coerenza tra server e dispositivo.

### Cambio Password

La funzionalità di cambio password segue un flusso simile: l'utente accede a una schermata dedicata, dove può inserire la **vecchia password**, la **nuova password** e la relativa **conferma**.

Prima di inviare la richiesta, l'applicazione esegue alcune **validazioni**: verifica che tutti i campi siano compilati, che la nuova password corrisponda alla conferma e che l'utente non sia un account Google (per i quali il cambio password non è consentito).

Solo se tutte queste condizioni sono rispettate, viene **effettuata la re-authentication su Firebase** utilizzando la vecchia password. La nuova password viene quindi aggiornata sul server remoto. Una volta confermato l'aggiornamento, i dati locali vengono **sincronizzati**, garantendo coerenza tra server e dispositivo.

Durante l'intero processo, l'utente riceve feedback immediato su eventuali problemi, come campi mancanti, password non corrispondenti, assenza di connessione o errori di autenticazione.

# BinGo!



## Web Services - Open Food Facts

The screenshot shows a product page for "Acqua minerale naturale – San Benedetto – 2 litri". At the top, there's a navigation bar with links for "Prodotti", "I tuoi criteri", "Salute", "Ambiente", "Segnala un problema", and "Contributo". A search bar and a "SCARICA L'APP" button are also present. Below the navigation, there's a section for "Questa pagina prodotto non è completa. Puoi contribuire a completarla modificandola e aggiungendo altri dati dalle foto in nostro possesso, oppure scattando altre foto tramite l'app per Android o iPhone/iPad. Grazie!".

Key data points highlighted with red boxes:

- Codice a barre: 8053259800282 (EAN / EAN-13)
- Quantità: 2 litri
- Packaging: in bottiglia per...

Below these, smaller text includes: Marche: San Benedetto; Categorie: Bevande, Acque, Acque di sorgente, Acque minerali; Paesi di vendita: Italia.

[Esempio di chiamata API](#)

## OPEN FOOD FACTS

**Open Food Facts** è un **database collaborativo open source** dedicato ai **prodotti alimentari**, accessibile gratuitamente tramite interfaccia web e API. Qualora il codice a barre richiesto non corrisponda a un prodotto alimentare, Open Food Facts effettua automaticamente un **reindirizzamento** della richiesta verso l'endpoint **World Open Products Facts** (<https://world.openproductsfacts.org>), restituendo comunque una risposta in formato **JSON** conforme allo **standard** utilizzato, **anche per prodotti non alimentari**.

Il **JSON** restituito dalle API viene **elaborato** dal servizio di utility **JsonDeserializer**, che sfrutta la libreria **Gson** per convertire i dati in un oggetto **DTO** della classe **ProductWithPackagingWithTranslation**.

Ogni prodotto può essere associato a uno o più **imballaggi**, in base alle diverse tipologie di confezionamento in cui è disponibile. Inoltre, per la natura dell'applicazione, viene garantita **un'esperienza utente multilingua**, che richiede il **salvataggio** delle **descrizioni** del prodotto nelle **diverse lingue supportate** dal sistema.

L'endpoint di Open Food Facts restituisce, per ciascun imballaggio, una **taxonomia**, ovvero un **codice** identificativo che **rappresenta il materiale utilizzato**. A tal fine, viene impiegato un secondo servizio di utility, **MaterialParser**, che si occupa di **convertire** il codice della **taxonomia** nella corrispondente classe **Material**, contenente il nome e la descrizione del materiale. La **documentazione e l'elenco** delle taxonomie sono forniti direttamente da Open Food Facts in formato JSON e sono disponibili al seguente indirizzo:

[https://static.openfoodfacts.org/data/taxonomies/packaging\\_materials.json](https://static.openfoodfacts.org/data/taxonomies/packaging_materials.json)

Poiché Open Food Facts è un database **collaborativo** in **continua evoluzione**, a ogni richiesta di un prodotto viene comunque effettuata una chiamata al servizio remoto, anche nel caso in cui il prodotto sia già presente nel database locale. Questa scelta consente di garantire la sincronizzazione e l'aggiornamento costante delle informazioni.

© Open Food Facts — Dati disponibili sotto licenza Open Database License (ODbL) 1.0

Questa applicazione è un progetto indipendente realizzato utilizzando dati di Open Food Facts.



# BinGo!



## Sviluppi Futuri

L'applicazione BinGo! presenta diverse possibilità di estensione e miglioramento, che potrebbero essere sviluppate in versioni future al fine di arricchire l'esperienza utente e aumentare l'utilità complessiva dell'applicazione.

### Integrazione di API per mappe e punti di raccolta

Un possibile sviluppo futuro riguarda l'integrazione di **API di mappe**, che permetterebbero di visualizzare i **punti di raccolta dei rifiuti** in base alla posizione dell'utente.

Questa funzionalità consentirebbe di mostrare sulla mappa i centri di raccolta, le isole ecologiche o i punti di conferimento più vicini, facilitando ulteriormente il corretto smaltimento dei rifiuti.

### Accessibilità

Un altro possibile sviluppo riguarda il miglioramento dell'accessibilità dell'applicazione, con particolare attenzione agli utenti con disabilità.

In questo contesto, potrebbe essere prevista l'integrazione con **assistenti vocali** o sistemi di lettura vocale, che consentano di utilizzare alcune funzionalità dell'app attraverso comandi vocali o feedback audio.

### Feedback degli utenti

Un possibile sviluppo futuro riguarda l'introduzione di un sistema di feedback da parte degli utenti, pensato non solo per la segnalazione di problemi o suggerimenti sull'applicazione, ma anche per la condivisione di informazioni utili sul territorio.

In particolare, gli utenti potrebbero segnalare punti di raccolta non ancora presenti, errori nelle informazioni fornite o aggiornamenti relativi alle modalità di smaltimento dei rifiuti. Queste segnalazioni potrebbero contribuire a mantenere i contenuti dell'applicazione più accurati e aggiornati nel tempo.