UNIVERSITÀ DELLA CALABRIA

DIPARTIMENTO DI INGEGNERIA INFORMATICA, MODELLISTICA, ELETTRONICA E SISTEMISTICA

DIMES

Corso di Laurea in
Ingegneria Informatica

Relazione progetto Sistemi Distribuiti e Cloud Computing: Food Pets

Professore

Prof. Domenico Talia
Ing. Loris Belcastro

Candidati

Ilaria Vrenna 224676

INTRODUZIONE

Food Pets è un'applicazione Android che vuole risolvere un problema che accomuna chiunque abbia un animale domestico che è quello di gestire al meglio le sue abitudini alimentari garantendogli una dieta sana ed equilibrata.

L'idea di base prevede di realizzare un'applicazione da cui è possibile comandare, con tecnologia bluetooth, una scheda elettronica Arduino che si occuperà di azionare l'erogatore delle crocchette, il quale rilascerà la giusta quantità di cibo. Oltre a ciò, all'interno dell'app, sono offerti diverse altri servizi, ossia:

- Uno spazio dedicato per ogni utente;
- Ricerca dei dispositivi bluetooth, connessione e trasferimento dati;
- Controllo degli ultimi pasti del proprio animale;
- Informazioni circa il livello di cibo ancora disponibile all'interno della mangiatoia.

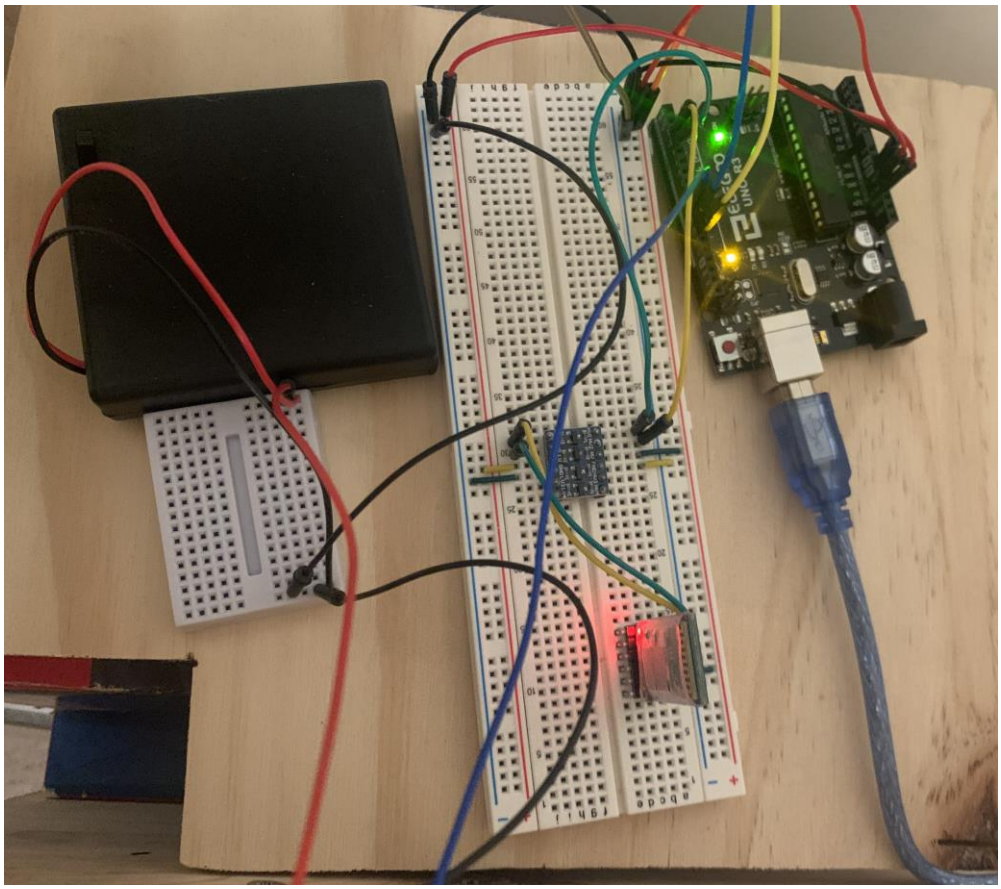
Ai fini progettuali sono stati adoperati l'IDE di Google (noto come Android Studio), Arduino IDE, il database MySQL e l'editor Visual Studio Code.

COMPONENTI HARDWARE E CIRCUITO

Per lo sviluppo del progetto sono state impiegate le seguenti componenti:

- Scheda Arduino UNO;
- Servo motore;
- Modulo bluetooth HC-05;
- Convertitore di livello logico bidirezionale;
- Sensore di distanza ad ultrasuoni HC-SR04;
- Contenitore batteria AA;

Di seguito si mostra la struttura del circuito che è stata realizzata.

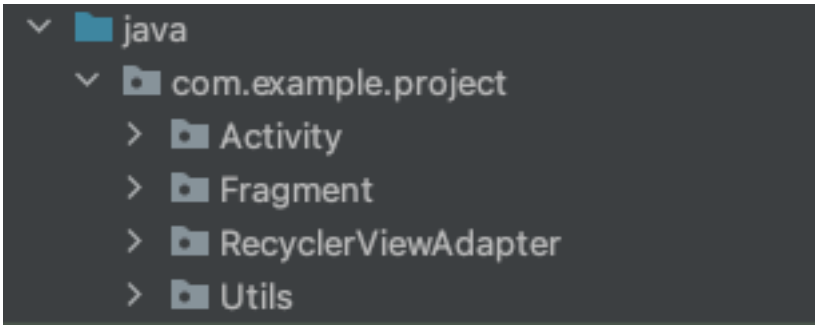



25kg

PROGETTAZIONE GENERALE

1. Divisione Package

In questa sezione si andrà a descrivere la divisione in package dell'applicazione *Food Pets*.



Il progetto è strutturato sulla base di quattro package principali: Activity, Fragment, RecyclerViewAdapter e Utils. Il package Activity contiene tutte le Activity che vanno a comporre l'applicazione e che gestiscono l'interazione con l'utente. All'interno del package Fragment sono presenti i Fragment che possono essere considerati una porzione dell'interfaccia utente, il cui ciclo di vita è strettamente legato al ciclo di vita dell'activity a cui sono associati. Il package RecyclerViewAdapter include le classi che estendono RecyclerView.Adapeter e che permettono di iterare gli elementi delle varie liste usate all'interno dell'applicazione. Infine, nel package Utils sono presenti le classi che portano utilità all'applicazione e gestiscono la trasmissione dati con i dispositivi bluetooth rilevati durante la fase di ricerca.

2. Il file AndroidManifest.xml

L'AndroidManifest è un documento XML contenente le principali informazioni dell'applicazione, come il nome e la versione minima di Android che deve essere presente sul dispositivo per il suo corretto funzionamento.

All’interno del Manifest è anche dichiarata la lista di tutte le classi che svolgono il ruolo di activity e i permessi necessari per poter usufruire delle funzionalità bluetooth e della connessione ad Internet utilizzata per l’accesso al database.

3. Cartella res

All’interno della cartella *res* del progetto Android sono presenti tutte le risorse che vengono utilizzate all’interno dell’applicazione. Le risorse sono dei file fondamentali per una app perché consentono di gestire diversi aspetti: immagini, layout, stringhe per internazionalizzare i testi delle interfacce, animazioni e tanto altro.

IMPLEMENTAZIONE

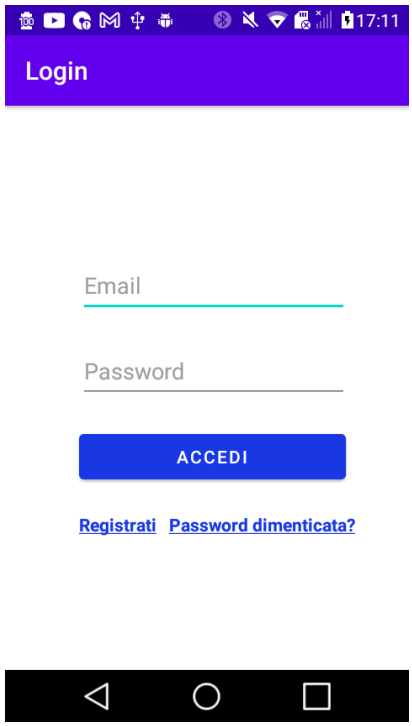
In questa sezione verranno descritte le implementazioni delle principali classi dell'applicazione per semplificarne la comprensione.

1. SplashScreen

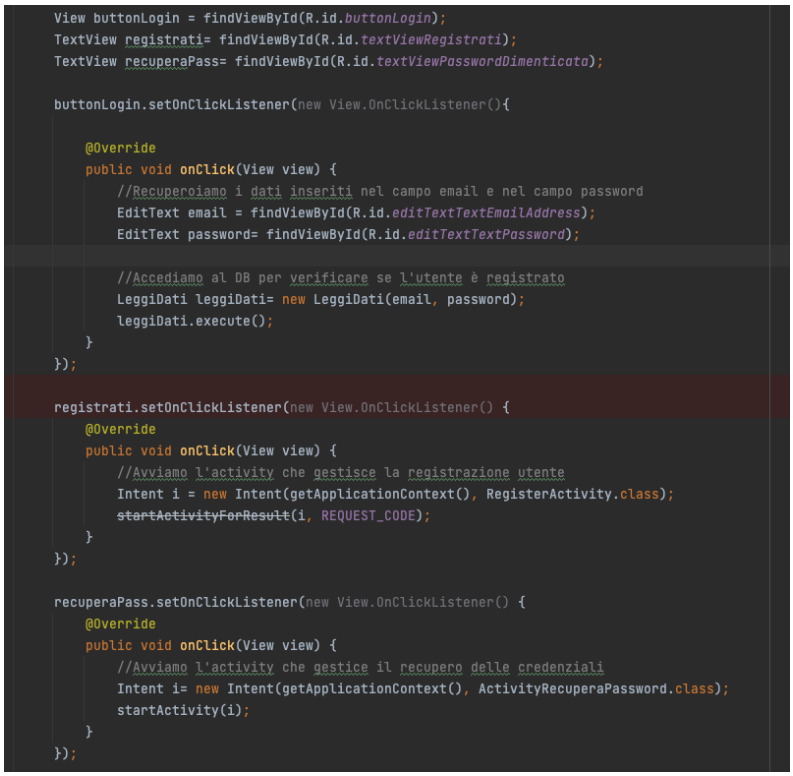
SplashScreen è l'activity principale che viene lanciata nel momento in cui l'utente avvia l'app per la prima volta e visualizzerà l'animazione, nel frattempo, che l'app sta completando la sua fase di caricamento, al termine della quale si avvierà l'activity successiva, chiamata LoginActivity.

2. LoginActivity

La LoginActivity implementa la funzione di login che consente a ciascun utente, inserendo la propria e-mail e password, di accedere alla propria area riservata da cui è possibile usufruire di tutte le funzionalità messe a disposizione dell'utente.



A partire dalle TextView, *Registrati* e *Password Dimentica*, si avvieranno le activity che gestiscono rispettivamente la registrazione e il recupero della password. L'onClick sul button *Accedi* va a verificare se i dati inseriti nei campi email e password siano corretti, altrimenti si visualizzerà un AllertDialog con il corrispondete codice di errore. Si illustra una piccola porzione di codice che mostra la come vengono gestiti gli eventi di onClick sulle View descritte.

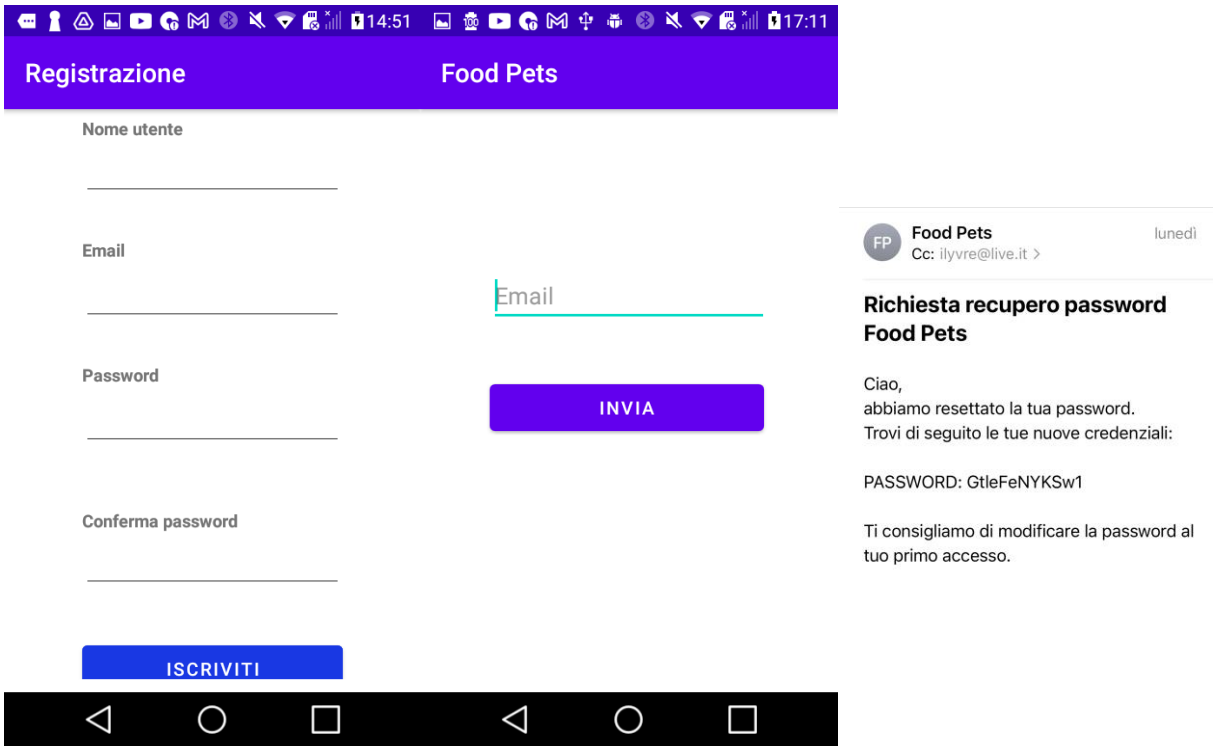


```
View buttonLogin = findViewById(R.id.buttonLogin);
TextView registrati= findViewById(R.id.textViewRegistrati);
TextView recuperaPass= findViewById(R.id.textViewPasswordDimenticata);

buttonLogin.setOnClickListener(new View.OnClickListener(){

    @Override
    public void onClick(View view) {
        //Recuperoiamo i dati inseriti nel campo email e nel campo password
        EditText email = findViewById(R.id.editTextTextEmailAddress);
        EditText password= findViewById(R.id.editTextTextPassword);

        //Accediamo al DB per verificare se l'utente è registrato
        LeggiDati leggiDati= new LeggiDati(email, password);
        leggiDati.execute();
    }
});

registrati.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        //Avviamo l'activity che gestisce la registrazione utente
        Intent i = new Intent(getApplicationContext(), RegisterActivity.class);
        startActivityForResult(i, REQUEST_CODE);
    }
});

recuperaPass.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        //Avviamo l'activity che gestice il recupero delle credenziali
        Intent i= new Intent(getApplicationContext(), ActivityRecuperaPassword.class);
        startActivity(i);
    }
});
```

LeggiDati è una sottoclasse che estende AsyncTask e si occupa di gestire la comunicazione con il database. La classe AsyncTask permette, con i suoi metodi, di effettuare operazioni di rete in maniera facile ed efficiente.

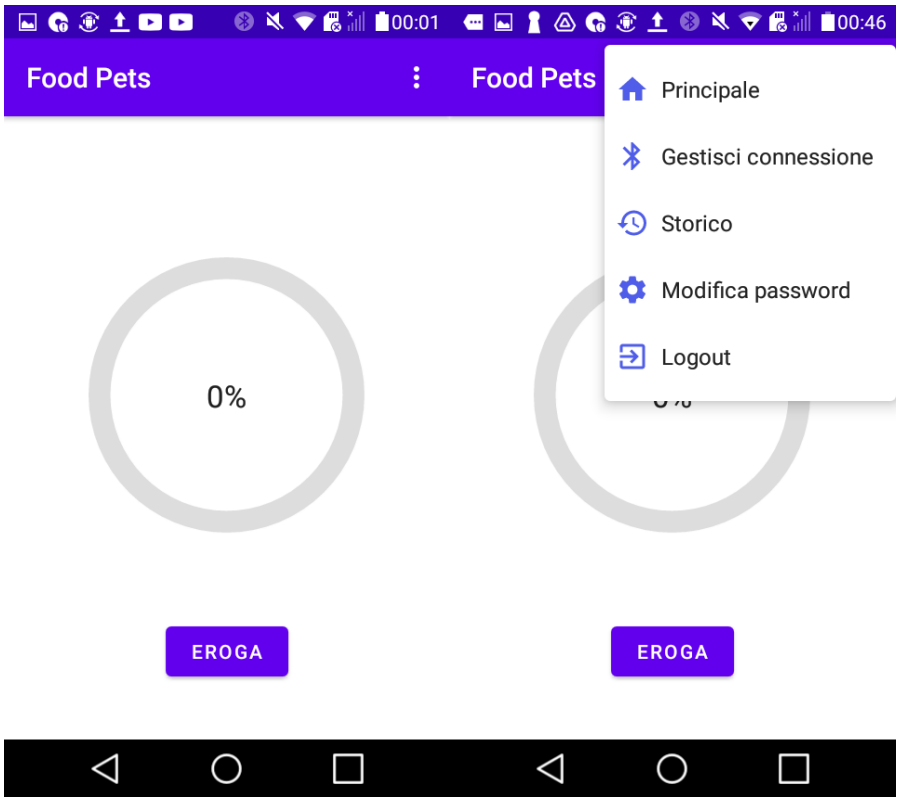
3. RegistratiActivity e ActivityRecuperaPassword

RegistratiActivity è la classe a partire dalla quale l'utente può effettuare la sua registrazione. Per poter completare la registrazione, a ciascun utente è richiesto di compilare i campi: Nome utente, E-mail, Password e Conferma password. Su ciascuno di questi campi sono effettuati opportuni controlli per garantire la validità delle informazioni inserite. Mentre, ActivityRecuperaPassword è l'activity che offre la possibilità a ciascun utente di recuperare l'accesso qualora avesse smarrito la propria chiave; in particolare, riceverà sull'email, usata in fase di registrazione, una password temporanea che potrà usare per l'accesso.



4. MainActivity

La MainActivity è l'activity principale dell'applicazione ed è quella che permette l'utilizzo di tutte le funzionalità offerte. L'utente, dopo aver inserito, a partire dall'interfaccia di login, l'e-mail e la password corretta, avrà accesso a questa activity, all'interno della quale è presente un OptionMenu che si compone di più voci: *Principale*, *Gestisci connessione*, *Storico*, *Modifica password* e *Logout*. Selezionando ciascuna di queste opzioni verranno visualizzate delle componenti grafiche e nascoste delle altre.

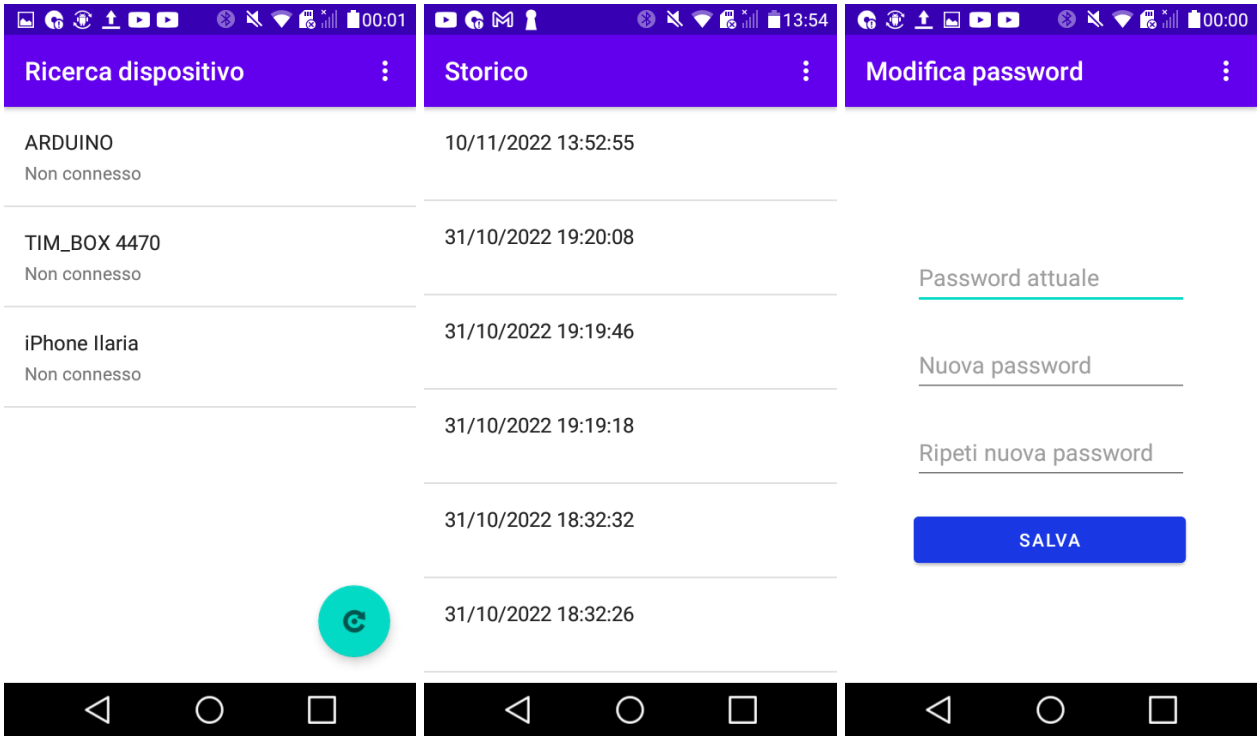


Più dettagliatamente, selezionando la voce di menu *Gestisci connessione* si visualizzerà il button che avvia la ricerca dei dispositivi bluetooth presenti nelle vicinanze e la lista dei dispositivi rilevati.

Selezionando la voce *Storico* dal menu, l'utente potrà controllare quali sono stati gli ultimi pasti erogati, in particolare l'applicazione tiene traccia delle ultime dieci erogazioni fatte e per ciascuna di queste è indicata la data e l'ora.

A partire da *Modifica password*, è possibile modificare la propria chiave d'accesso. Mentre, con *Logout* si implementa la disconnessione; in tal caso l'utente verrà rindirizzato sull'interfaccia di login.

L'opzione *Principale* riconduce l'utente sull'interfaccia di partenza che contiene una ProgressBar, in cui, dopo aver instaurato la connessione con Arduino, verrà mostrata la percentuale di carico della mangiatoia, e il button *EROGA*, dal quale verrà avviata l'erogazione delle crocchette.



5. Sketch Arduino

Per programmare la scheda Arduino è stato scritto un piccolo sketch che è essenzialmente diviso in due parti principali: *setup ()* e *loop()*. Nella prima parte è posto il codice di inizializzazione che formatta tutte le impostazioni e le operazioni di configurazione delle porte di ingresso/uscita delle componenti hardware e software. Questo gruppo di istruzioni, di fatti, viene eseguito una sola volta al momento di accensione, prima che il ciclo principale abbia inizio. Nella seconda parte è contenuto il codice principale, che è formato da una sequenza di istruzioni che vengono eseguite una dopo l'altra quando Arduino ha terminato di eseguire la parte di setup. In particolare, all'interno del metodo loop si va a verificare se sul canale bluetooth ci siano dei caratteri da leggere: se il carattere coincide con "a" si avvia il servomotore che compiendo una rotazione di 180° rilascia la giusta quantità di crocchette; mentre, se il carattere ricevuto è "b", si aziona il sensore ad ultrasuoni, i cui dati ricevuti vengono adoperati per calcolare

la percentuale di cibo presente all'interno della mangiatoia, a tale scopo si è scritta la funzione *InviaDati().*

6. File php

Dal lato server, per comunicare con il database sono stati scritti diversi script php:

- *Insert_user.php*: gestisce la registrazione di nuovi utenti.
- *Select_list.php*: a partire dall'identificatore univoco associato a ciascun utente recupera la lista delle erogazioni, espressa sotto forma di stringa.
- *Select_user.php*: verifica che le credenziali di accesso fornite dell'utente in fase di login siano corrette, altrimenti ritorna un messaggio di errore.
- *Send_email.php*: implementa il recupero password; in particolare, va a verificare se l'e-mail fornita sia presente all'interno del DB, costruisce il messaggio inserendo la password temporanea e lo spedisce all'indirizzo di posta elettronica specificato.
- *Update_lista.php*: aggiorna la lista delle ultime erogazioni.
- *Update_password.php*: aggiorna il campo password con la password fornita dall'utente.

CODICE SORGENTE PROGETTO

Di seguito è possibile prendere visione del codice sorgente del progetto sviluppato:

FIGURA 1: [LINK TO GITHUB REPOSITORY]