Relazione per il progetto di informatica di Alessio Tommasi, Jhefferson Castro e Augustina Bulicanu

UTILIZZO GENERALE

Nella nostra scuola c’è la possibilità, durante l’intervallo, di comprare da una persona esterna la merenda, la prenotazione per ogni classe avviene nel seguente modo: entro le 10.00 ogni classe deve consegnare la lista (fogli prestampati che contengono la lista dei prodotti presenti) con le prenotazioni della classe, all’intervallo ogni rappresentante ritira i prodotti della propria classe.

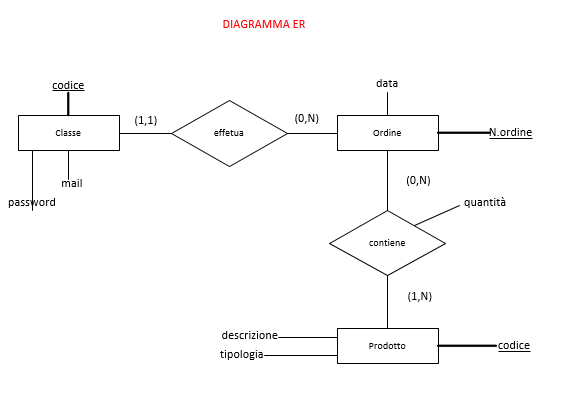
Si è pensato allora di sviluppare un’applicazione che permettesse di svolgere questo compito direttamente da un dispositivo mobile.

L’iniziativa dell’applicazione nasce con l’idea di agevolare sia gli studenti che la paninara, l’obbiettivo è quello di rendere più efficiente il processo di prenotazione e in modo da evitare lo spreco di carta.

L’applicazione finale deve “contenere” due interfacce differenti: un’interfaccia per gli studenti e una per la paninara.

L’ utilizzo dell’applicazione è stato pensato come segue: saranno solo i rappresentanti di classe ad installare l’applicazione e avranno loro la responsabilità delle prenotazioni.

Alla paninara, che avrà anche lei installato l’applicazione, arriveranno le prenotazioni da parte di ogni classe.

Nel diagramma ER viene rappresentata la realtà di interesse:

REGOLE DI LETTURA

Nello schema sono presenti 3 entità: Classe, Ordine, Prodotto.

Le associazioni fra le varie entità con la loro cardinalità e la loro partecipazione totale e parziale sono mostrate nel diagramma.

Sono degne di nota le seguenti osservazioni:

- Nell’entità Classe i 3 attributi sono stati pensati nel seguente modo:

A tutte le classi viene fornito un codice che al primo utilizzo viene usato sia come utente che come password, al primo utilizzo il programma obbliga l’utente a cambiare la password (questo avviene per motivi di sicurezza: se il codice, che è noto a tutto l’istituto, venisse utilizzato sia come utente che come password, alcuni studenti per semplice gioco potrebbero accedere agli ordini di altre classi e modificarli, questo comporterebbe problemi; ovviamente la password di ogni classe non deve essere divulgata).

L’email viene fornita per una questione di sicurezza: nel caso la password venga dimenticata.

- Nell’ entità Prodotto l’attributo tipologia serve ad identificare a quale delle 4 tipologie appartiene il prodotto (pizza, panino, piadina cotoletta).

Schema concettuale

(realizzato tramite il modello relazionale)

***CLASSE***

|  |  |  |  |
| --- | --- | --- | --- |
| **CAMPO** | **TIPO** | **PK/FK** | **NOTE** |
| Codice | char (8) | PK |  |
| Mail | Varchar (50) |  | Not null |
| Password | Varchar (50) |  | Non deve contenere meno di 8 caratteri |

***ORDINE***

|  |  |  |  |
| --- | --- | --- | --- |
| **CAMPO** | **TIPO** | **PK/FK** | **NOTE** |
| N.ordine | char (8) | PK |  |
| data | char (10) |  |  |
| FKcodice | Varchar (8) | FK | Si riferisce alla PK della tabella “ Classe” |

***PRODOTTO***

|  |  |  |  |
| --- | --- | --- | --- |
| **CAMPO** | **TIPO** | **PK/FK** | **NOTE** |
| Codice | char (8) | PK |  |
| Descrizione | Varchar (50) |  |  |
| Tipologia | Varchar (50) |  | Not null |

***CONTENUTO***

|  |  |  |  |
| --- | --- | --- | --- |
| **CAMPO** | **TIPO** | **PK/FK** | **NOTE** |
| FKCodice | char (8) | FK  PK | Si riferisce alla PK della tabella “ Prodotto” |
| FKN.ordine | char (8) |  | Si riferisce alla PK della tabella “ Ordine” |
| Quantità | int |  |  |

Vincoli di integrità

1. **VINCOLI DI CHIAVE PRIMARIA (vincoli impliciti)**

Tutti i campi indicati con PK sono da intendere chiavi primarie, dunque NOT NULL e con valori univoci.

1. **VINCOLI DI DOMINIO (vincoli espliciti)**

Tutti i vincoli di dominio sono stati riportati nelle tabelle sopra nella colonna NOTE.

1. **VINCOLI DI ENNUPLA (vincoli espliciti)**

**/**

1. **VINCOLI DI INTEGRITA’ REFERENZIALE (vincoli impliciti ed inter-relazionali)**

Tutte le chiavi esterne indicate sopra con FK sono tali da assumere uno dei valori assunti dalla chiave primaria a cui si riferiscono.

FASI DI LAVORO

Il progetto è stato sviluppato in varie fasi. Prima di tutto è stata svolta un’analisi dell’obiettivo da raggiungere, cioè sono stati stabiliti la piattaforma e l’obiettivo cardine dell’attività.

Il progetto elaborato è stato sviluppato in linguaggio Java, l’ambiente di sviluppo utilizzato è Android Studio, versione Android Oreo 8.0.

L’IDE Android Studio è stato progettato specificatamente per lo sviluppo di applicazioni Android e quindi rende più efficiente il lavoro del programmatore inoltre facilita e rende più stimolante l’interfaccia utente.

Il progetto utilizza anche il metalinguaggio XML per definire e controllare gli elementi contenuti nei differenti layout.

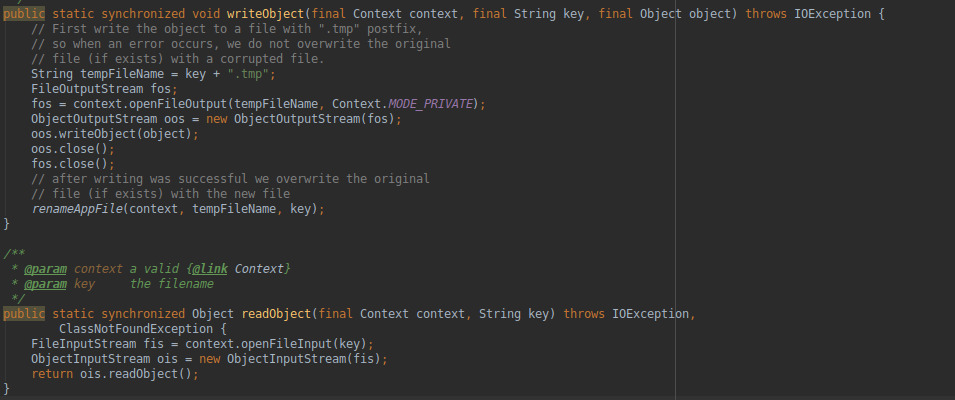
Il progetto è costituito da 10 classi: MainActivity.java, MD5.java, LoginActivity.java, DataModel.java, CustomAdapter.java, Carrello.java, PlaceholderFragment.java, SelectionsPagerAdapter.java, Values.java, paginaRegistrazione.java.

Le classi implementano:

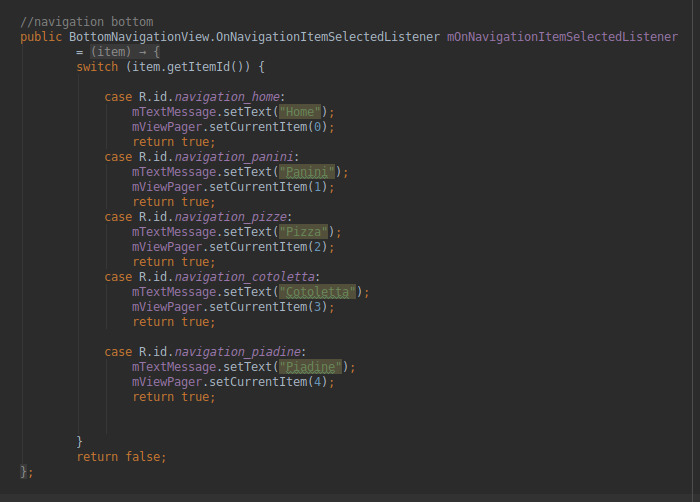
* ***MainActivity.java*** contiene il codice per l’avvio della schermata home, crea il Fragment per lo swipe utilizzando la classe play PlaceholderFragment.java.
* ***LoginActivity.java*** contiene il codice per il layout riguardante il login con i rispettivi controlli (i controlli verranno implementati dalla classe MD5.java).
* ***MD5.java*** contiene il codice per cifrare i dati del database.
* ***DataModel.java***  contiene il codice che definisce il modello di ogni riga del customListView del carrello.
* ***CustomAdapter.java*** contiene il codice che adatta la lista delle varie tipologie di prodotti per poter utilizzare il metalinguaggio XML.
* ***Carrello.java*** contiene il codice per gestire il layout del carrello inoltre contiene gli ascoltatori per il controllo del login (si possono aggiungere prodotti al carrello solo se prima è avvenuto il login all’applicazione, ovviamente l’operazione di login deve essersi conclusa con successo!)
* ***PlaceholderFragment.java***  contiene il codice che si occupa della gestione del coordinamento delle varie pagine.
* ***SelectionsPagerAdapter.java***  contiene il codice responsabile dell’ implementazione delle funzioni di gestione dei Fragment.
* ***Values.java*** contiene le funzioni predisposte per immagazzinare i dati prelevati dal database.
* ***paginaRegistrazione.java*** contiene il codice per il layout della pagina di registrazione e controlla che le informazioni inserite rispettino i vincoli imposti (es. password con minimo 8 caratteri) inoltre avvia l’activity di registrazione del layout activity\_pagina\_registrazione.xml.

Il compito dei diversi metodi nelle varie classi è facilmente intuibile dal codice, abbiamo però ritenuto necessario evidenziare alcuni blocchi di istruzioni poiché ci hanno dato particolare difficoltà o perché non sono facilmente comprensibili per chi non ha lavorato direttamente al progetto.

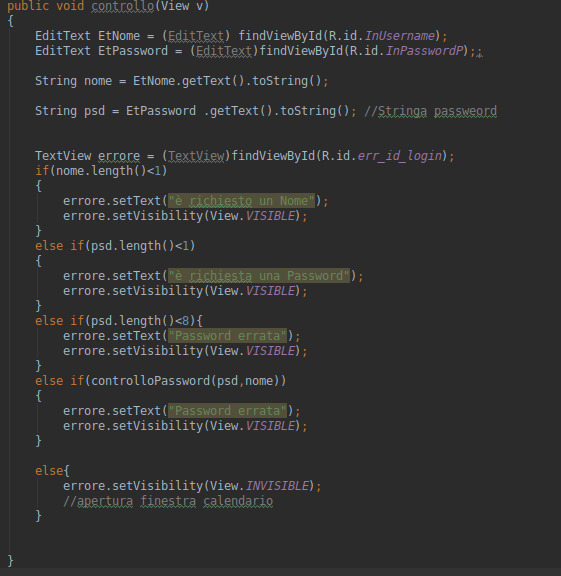
* Codice presente nel ***MainActivity.java,***  permette la scrittura e la lettura di oggetti da file.



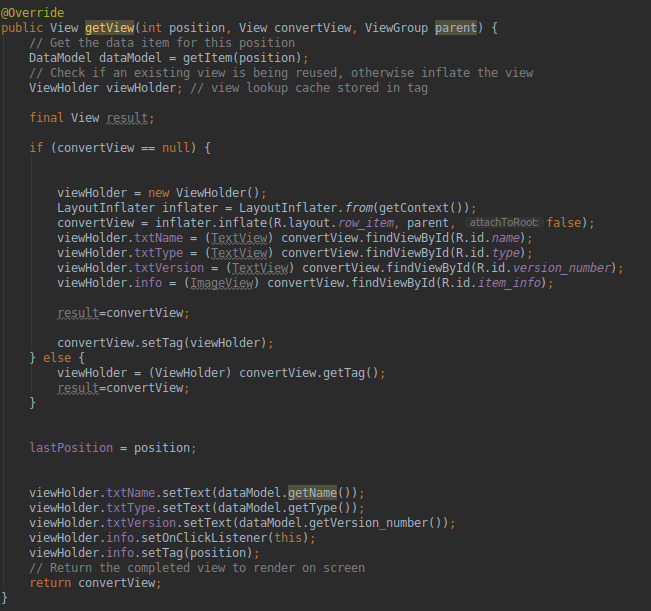
* Codice presente nel ***MainActivity.java,*** grazie al quale le pagine vengono settate.



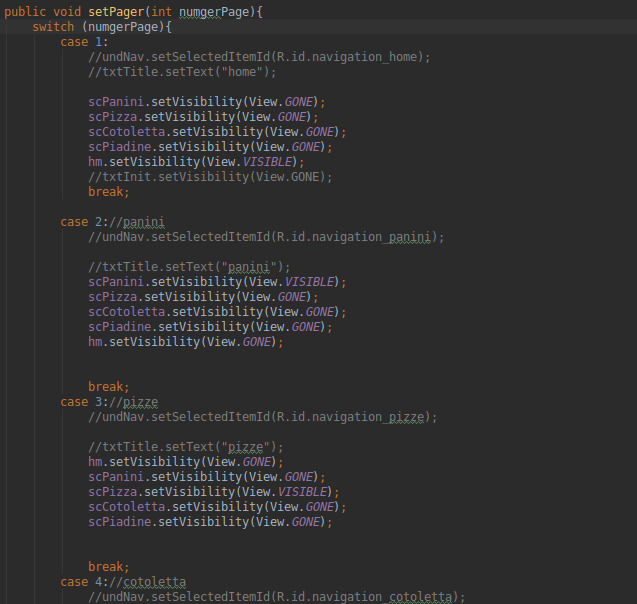
* Codice presente nel ***LoginActivity.java,*** che implementa i controlli necessari per effettuare il login.



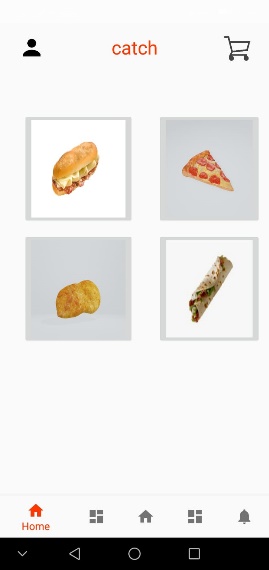
* Codice presente nel ***CustomAdapter.java,*** grazie al quale è possibile adattare l’elemento xml alla riga di codice, codice che poi verrà utilizzato per la CustomListView.



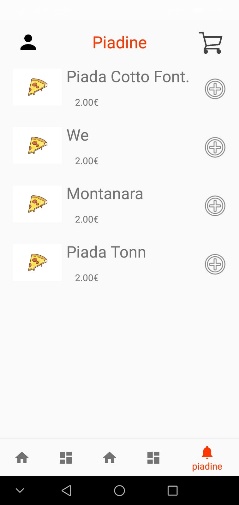
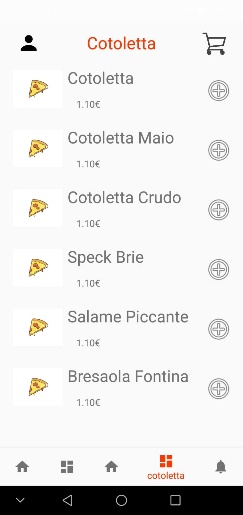
* Codice presente nel ***PlaceholderFragment.java,*** grazie al quale vengono settate le pagine quando si utilizza lo Swipe.



Oltre alle classi il progetto è costituito anche da 8 layout: activity\_main.xml, activity\_login.xml, activity\_carrello.xml, activity\_pagina\_registrazione.xml, content\_main.xml, elenco.xml, fragment\_main.xml, row\_item.xml.

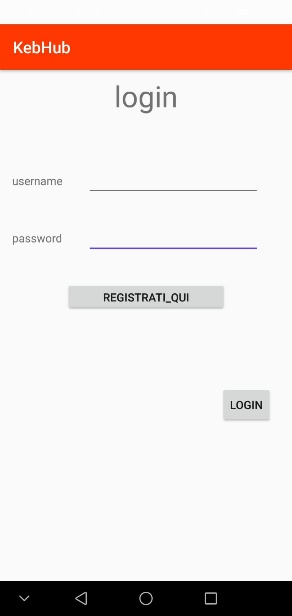
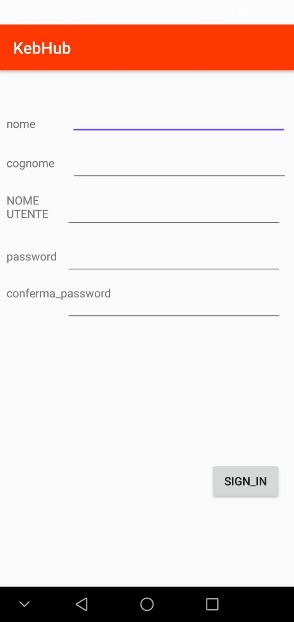


L’immagine mostra la schermata principale dell’applicazione, da qui si può accedere all’area di login, al carrello e in diversi modi (grazie allo swipe o cliccando sulle immagini dei quadrati) all’ elenco delle diverse tipologie di prodotto (come mostra l’immagine le tipologie sono 4: panino, pizza, cotoletta e piadina; ognuno di loro ha una lista di sottotipi (immagine sottostante)).

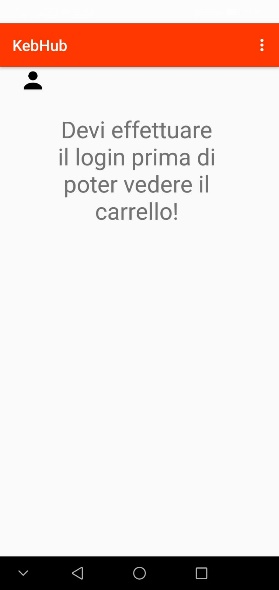


I layout sottostanti corrispondono all’area dedicata al login: la prima immagine è il layout che apparre quando si clicca sul login, qui gli utenti già registrati possono inserire il proprio username e la password; nel caso in cui

un utente deve registrarsi dovrà premere il bottone “registrati qui” e si troverà la schermata della seconda immagine dove dovrà inserire i dati richiesti per eseguire con successo la registrazione. Come spiegato nelle regole di lettura del diagramma ER a tutte le classi viene fornito un codice che al primo utilizzo viene usato sia come utente che come password, al primo utilizzo il programma obbliga l’utente a cambiare la password per motivi di sicurezza.

L’ultimo layout riguarda il carrello, i prodotti selezionati vengono aggiunti in quest’area e dopo la conferma l’ordine viene spedito alla paninara; per poter aggiungere prodotti al carrello è necessario aver prima effettuato il login nell’apposita area.



CONCLUSIONI

Alla fine di questo progetto viene redatta questa relazione per documentare il lavoro svolto e il codice.

Durante la fase finale di testing l’applicazione sembra funzionare dapprima con qualche errore, ma dopo alcune modifiche al codice sorgente essa funziona, nonostante persistono difficoltà con lo swipe.

Per lo sviluppo del progetto sono stati consultati alcuni forum per la risoluzione di vari errori, in particolare sono state riscontrate difficoltà soprattutto per quanto riguarda la gestione dei Fragment, per questo è stato richiesto l’aiuto del professore.

Le competenze raggiunte tramite questa attività sono: sviluppo di un’applicazione tramite l’ambiente di sviluppo Android Studio.