

BFast



Progetto a cura di

01/04/2020, Rev 1.5

Guglielmo Strambini

Daniela De Pascali

Alexandru Rusei

Ither Khaza Rahman

Indice

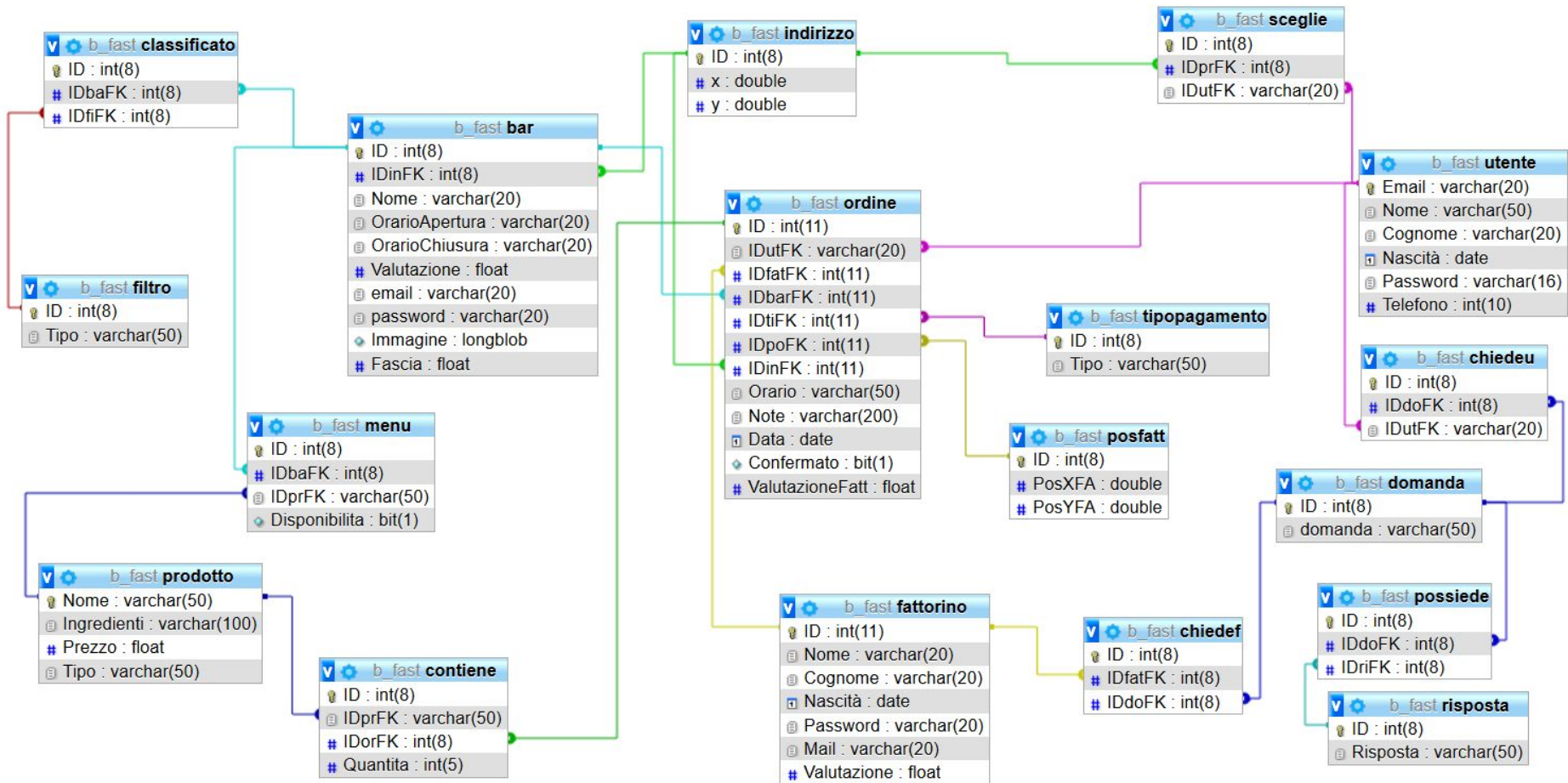
Indice

- *DB*
- *Parte Web*
 - *Schema*
 - *BackEnd*
 - *FrontEnd*
- *Applicazione*
 - *Schema*
 - *BackEnd*
 - *FrontEnd*

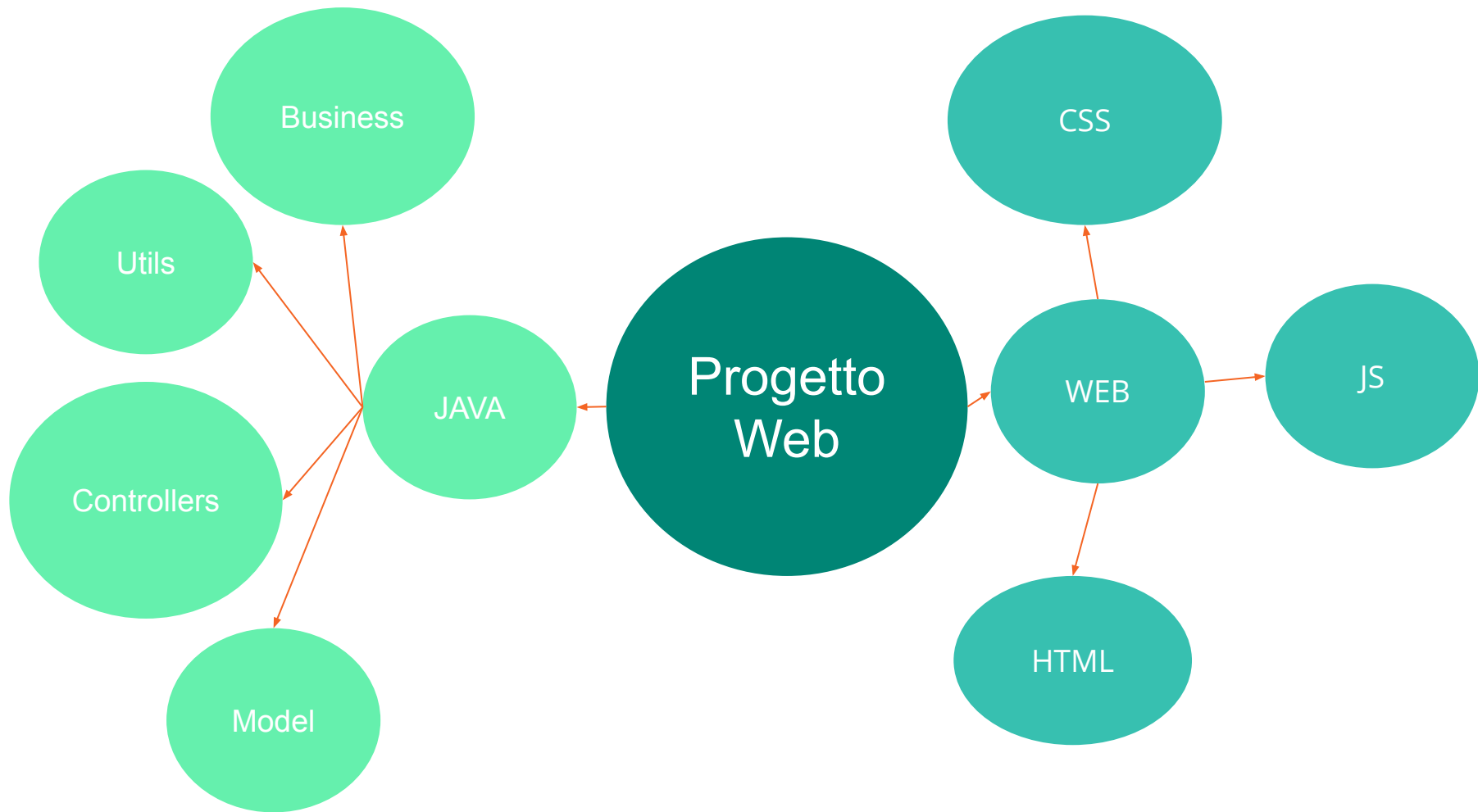
Data Base

Partendo dalla richiesta abbiamo iniziato a creare il diagramma E/R con un'entità principale ovvero quella degli ordini e da lì abbiamo creato le entità bar, utente e fattorino.

Infine in base alle iterazioni che uno dei tre utenti doveva svolgere, come fare domande oppure settare la via, e grazie alle regole di derivazione abbiamo creato il DataBase.



Parte web



Backend

Java

*Il Backend del nostro sito è stato
scritto in java, linguaggio di
programmazione ad alto livello,
orientato agli oggetti e a
tipizzazione statica, fu annunciato
ufficialmente nel 1995.*



JPA

Framework per il linguaggio di programmazione Java che si occupa della gestione della persistenza dei dati di un DBMS.

Maven

Maven usa un costrutto conosciuto come Project Object Model (POM); un file XML che descrive le dipendenze fra il progetto e le varie versioni di librerie necessarie nonché le dipendenze fra di esse.

Maven effettua automaticamente il download di librerie Java e plug-in

Utils Package

Nella Utils package ci sono tutte le operazioni che si ripetono più volte all'interno del codice e per facilitare il tutto le scriviamo una sola volta e le andiamo a chiamare

```
1 package utils;
2
3 import javax.persistence.EntityManagerFactory;
4
5
6 public class JPAUtil {
7
8     private EntityManagerFactory emf;
9
10    private static JPAUtil instance;
11
12    /**
13     *
14     */
15    private JPAUtil() {
16        this.emf = Persistence.createEntityManagerFactory("b.fast");
17    }
18
19    public static JPAUtil getInstance() {
20        if (instance == null)
21            instance = new JPAUtil();
22        return instance;
23    }
24
25    /**
26     * @return the emf
27     */
28    public EntityManagerFactory getEmf() {
29        return emf;
30    }
31
32
33 }
```

Model Package

*Nella model package ci sono tutte le
tabelle del db con tutti i campi
privati e i relativi getters and
setters*

```
13 @NamedQuery(name="Bar.findAll", query="SELECT b FROM Bar b")
14 public class Bar implements Serializable {
15     private static final long serialVersionUID = 1L;
16
17     @Id
18     private int id;
19
20     private String email;
21
22     private float fascia;
23
24     @Lob
25     private byte[] immagine;
26
27     private String indirizzo;
28
29     private String nome;
30
31     private String orarioApertura;
32
33     private String orarioChiusura;
34
35     private String password;
36
37     private float valutazione;
38
39     //bi-directional many-to-one association to Indirizzo
40     @ManyToOne
41     @JoinColumn(name="IDinFK")
42     private Indirizzo indirizzoBean;
43
44     //bi-directional many-to-one association to Classificato
45     @OneToMany(mappedBy="bar")
46     private List<Classificato> classificatos;
47
48     //bi-directional many-to-one association to Menu
49     @OneToMany(mappedBy="bar")
50     private List<Menu> menus;
51 }
```

Business Package

*Nella business package ci sono
tutte le operazioni che svolgiamo
con java e sono relative alla
comunicazione tra DB e Java*

```
1 package business;
2
3 import javax.persistence.EntityManager;
4
5
6
7
8
9 public class AutenticazioneUtente {
10
11     public Utente login(String mail, String password) {
12         Utente _return = null;
13         // cerco l'utente nel DB
14
15         EntityManager em = JPAUtil.getInstance().getEmf().createEntityManager();
16         _return = em.find(Utente.class, mail);
17         if (_return != null) {
18             // utente trovato; controllo la password
19             if (!password.contentEquals(_return.getPassword())) {
20                 _return = null;
21             }
22         }
23         em.close();
24         return _return;
25     }
26
27 }
28
```

Controller Package

Nella controller package ci sono tutte le operazioni che svolgiamo con java e sono relative alla comunicazione tra WEB e Java

```
14 import java.io.IOException;
15 @WebServlet("/login")
16 public class LoginControllerUtente extends HttpServlet {
17     private static final long serialVersionUID = 1L;
18     /**
19      * @see HttpServlet#HttpServlet()
20      */
21     public LoginControllerUtente() {
22         super();
23     }
24
25     /**
26      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
27      *      response)
28      */
29     protected void doGet(HttpServletRequest request, HttpServletResponse response)
30         throws ServletException, IOException {
31         HttpSession ses = request.getSession();
32         AutenticazioneUtente au = new AutenticazioneUtente();
33         Utente b = au.login(request.getParameter("mail"), request.getParameter("password"));
34         if (b == null) {
35             request.getRequestDispatcher("/").forward(request, response);
36         } else {
37             String id = request.getParameter("mail");
38             ses.setAttribute("ID", id);
39             request.getRequestDispatcher("/ok.html").forward(request, response);
40         }
41     }
42
43     /**
44      * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
45      *      response)
46      */
47     protected void doPost(HttpServletRequest request, HttpServletResponse response)
48         throws ServletException, IOException {
49         doGet(request, response);
50     }
51 }
```

Frontend

HTML

*Il FrontEnd del nostro sito è stato
scritto in HTML*

*Nato per la formattazione e
impaginazione di documenti
ipertestuali*



CSS

In informatica, è un linguaggio usato per definire la formattazione di documenti HTML e XML.

Inoltre per lo sviluppo CSS abbiamo utilizzato Bootstrap.

JavaScript

Linguaggio della programmazione orientato ad oggetti, per la creazione, in siti web e applicazioni web, di effetti dinamici interattivi tramite funzioni di script invocate da eventi innescati a loro volta in vari modi dall'utente sulla pagina web in uso

```
<nav id="sidebar">
  <div class="sidebar-header">
    <h3>Bar</h3>
  </div>
  <ul class="list-unstyled components">
    <li>
      <a> Home </a>
    </li>
    <br>
    <li>
      <a href="../GestioneIndirizzoOrario/SetPost.html">Gestisci orario e indirizzo</a>
    </li>
    <br>
    <br>
    <li>
      <a href="../GestioneUtente/index.html">Impostazioni</a>
    </li>
    <li>
      <a href="..">Logout</a>
    </li>
  </ul>
</nav>

<div id="content">
  <nav class="navbar navbar-expand-lg navbar-light">
    <div class="container-fluid">
      <button type="button" id="sidebarCollapse" class="btn">
        
      </button>
    </div>
  </nav>
```

```

65 <div id="content">
66   <nav class="navbar navbar-expand-lg navbar-light">
67     <div class="container-fluid">
68       <button type="button" id="sidebarCollapse" class="btn">
69         
70       </button>
71     </div>
72   </nav>
73   <h5 class="h1 text-center text-white" id="datiApp">DashBoard</h5>
74   <div class="align-middle d-md-flex w-100 justify-content-around">
75     <div class="login m-1 p-auto">
76       <div class="login-screen2">
77         <form action="Giorno" method="POST">
78           <div class="app-title">
79             <h1>Ordini giornalieri</h1>
80           </div>
81           <p id="Numerol"> </p>
82           <div class="login-form">
83             <input type="submit" onclick="validazione(event);" value="Visualizza" class="btn btn-primary btn-large btn-block">
84           </div>
85         </form>
86       </div>
87     <div class="login m-1 p-auto">
88       <form action="Mese" method="POST">
89     </div>
90   </div>
91   <div class="login m-1 p-auto">
92     <form action="Mese" method="POST">
93   </div>
94   <div class="login m-1 p-auto">
95     <form action="Mese" method="POST">
96   </div>
97   <div class="login m-1 p-auto">
98     <form action="Mese" method="POST">
99   </div>
100 </div>

```

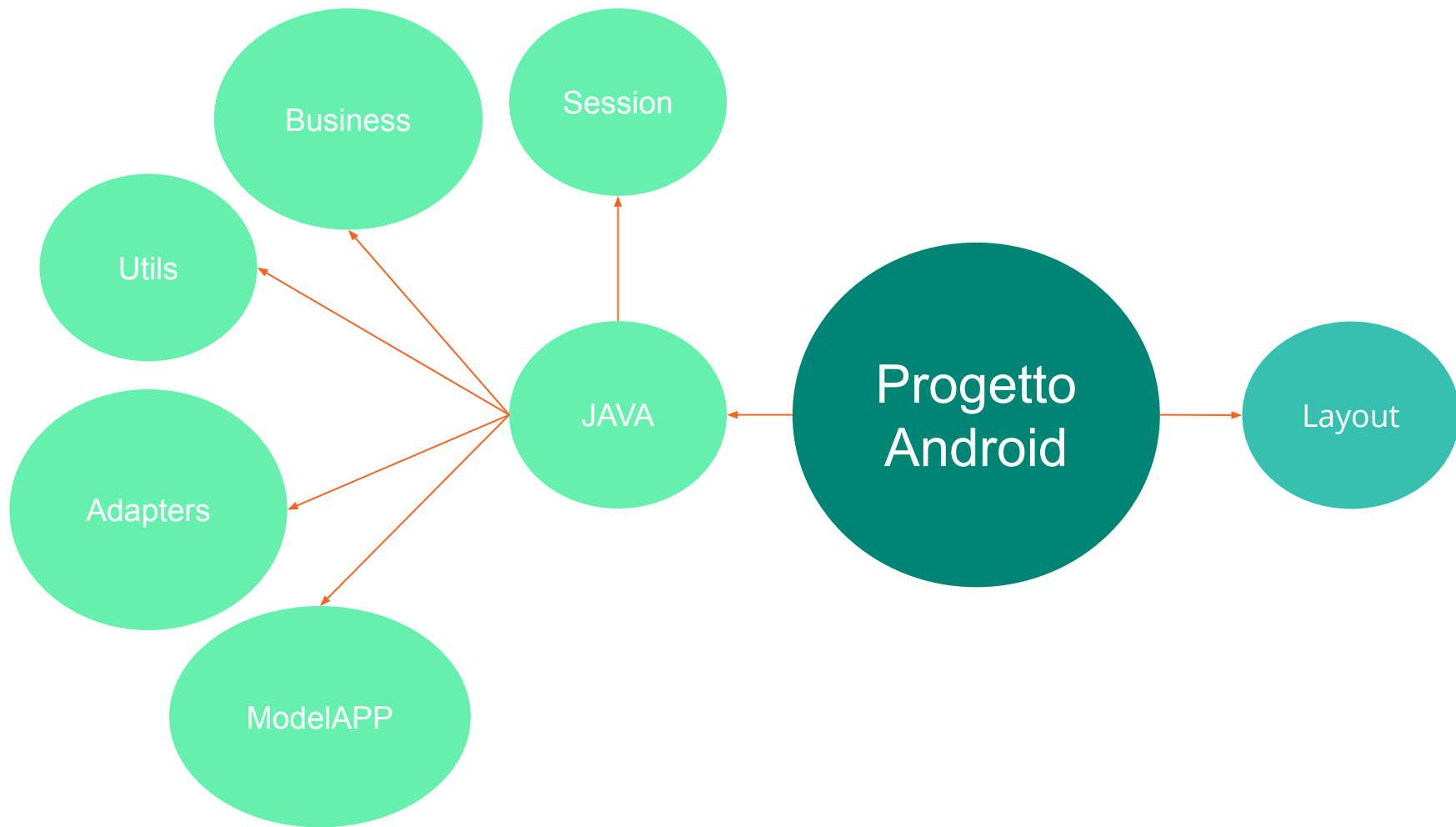
Applicazione

Android

*Sia per l'applicazione dell'utente e
del fattorino abbiamo utilizzato
android.*

*Android è un S.O. per i dispositivi
mobili sviluppato da Google nel 2007*





Backend

Utils Package

Nella Utils package ci sono tutte le operazioni che si ripetono più volte all'interno del codice e per facilitare il tutto le scriviamo una sola volta e le andiamo a chiamare

Retrofit

Retrofit è una libreria che consente di accedere ai servizi REST inserendo delle annotazioni ai metodi delle interfacce che rappresentano i servizi da utilizzare.


```
15 public class RetrofitUtils {
16     Gson gson = new GsonBuilder().registerTypeAdapter(Date.class, (JsonDeserializer) (json, typeOfT, context) → {
17         return new Date(json.getAsLong());
18     }).create();
19
20
21
22     private static RetrofitUtils instance = null;
23     public static final String BASE_URL = "http://192.168.1.27:8080/b fast/";
24     private BfastUtenteApi Bfast;
25     public static RetrofitUtils getInstance() {
26         if (instance == null) {
27             instance = new RetrofitUtils();
28         }
29         return instance;
30     }
31
32     private RetrofitUtils() { buildRetrofit(); }
33
34
35
36     private void buildRetrofit() {
37         Retrofit retrofit = new Retrofit.Builder()
38             .baseUrl(BASE_URL)
39             .addConverterFactory(GsonConverterFactory.create(gson))
40             .build();
41
42         this.Bfast = retrofit.create(BfastUtenteApi.class);
43     }
44
45     public BfastUtenteApi getBfastUtenteApi() { return this.Bfast; }
46
47
48 }
```

Rest

Per quanto riguarda lo sviluppo android, come prima cosa sono state definite le richieste web della nostra app, ossia le funzionalità rese disponibili in rete dai quali i nostri clienti e fattorini possono recuperare informazioni, richiedere elaborazioni o altro.

Di conseguenza sono state create due tabelle, una per il cliente e una per il fattorino.

A	B	C	D	E
METODO	URL	ENDPOINT	PARAMETRI	ESEMPIO JSON
Post	www.bfast.it/registrazione	Registrazione	Nome-Cognome-Mail-Password Telefono-Data di Nascita	{ "Nome": "Mario", "Cognome": "Bianchi" { "Nome": "Lucia", "Cognome": "Rossi"
Post	www.bfast.it/login	Login	Mail-Password	{ "Mail": "strambg@ciko.it", "password": "123" { "Mail": "bianchim@gmail.com", "password": "456"
Post	www.bfast.it/modificamail	Modifica Mail	Nuova Mail-Conferma Mail	{ "Nuova mail": "alex@r.it" { "Nuova mail": "mario.b@gmail.com"
Post	www.bfast.it/confermamail	Conferma Mail	Conferma mail	{ "Conferma mail": "alex@r.it", "Confermo" { "Conferma mail": "mario.b@gmail.com", "Confermo"
Post	www.bfast.it/modificapassword	Modifica Password	Nuova Pass-Conferma Pass	{ "Nuova password": "321" { "Nuova password": "654"
Post	www.bfast.it/passworddimenticata	Password Dimenticata	Cambia Password	{ "Password dimenticata": "123", "Cambia password": "789" { "Password dimenticata": "456", "Cambia password": "963"
Post	www.bfast.it/selezionaprodotto	Seleziona Prodotto	Nome-Tipo	{ "Nome": "Caffe", "Tipo": "Macchiato" { "Nome": "Brioche", "Tipo": "crema"
Post	www.bfast.it/selezionaquantita	Seleziona Quantità	Numero	{ "Numero": "2" { "Numero": "4"
Post	www.bfast.it/selezionabar	Seleziona Bar	Nome Bar	{ "Nome Bar": "Alex's bar" { "Nome Bar": "Bar Ciko"
Post	www.bfast.it/selezionaposizione	Seleziona Posizione	Inserisci Posizione	{ "Inserisci posizione": "X,Y"
Post	www.bfast.it/confermaposizione	Conferma Posizione	Conferma Posizione	{ "Conferma posizione": "Confermo"
Post	www.bfast.it/carrello	Carrello/Oridini	Prodotto-Quantità	{ "Prodotto": "Caffe", "Quantità": "2" { "Prodotto": "Brioche", "Quantità": "1"
Post	www.bfast.it/scrividomanda	Scrivi Domanda	Domanda	{ "Scrivi domanda": "è possibile inserire un domicilio diverso?" { "Scrivi domanda": "in quanto tempo posso ricevere l'ordine?"
Post	www.bfast.it/scrivrisposta	Scrivi Risposta	Risposta	{ "Scrivi risposta": "si" { "Scrivi risposta": "no"
Get	www.bfast.it/valutazionefattorino	Valutazione Fattorino	Valutazione	{ "Valutazione Fattorino": "2" { "Valutazione Fattorino": "5"
Get	www.bfast.it/updatebar	Update Bar	Bar	{ "ID": "8", "Bar": "Ciko", "Orario Apertura": "6.00", "Orario Chiusura": "18.00", "Valutazione": "5", }
Get	www.bfast.it/updateprodotto	Update Prodotto	Prodotto	{ "Nome": "Brioche", "Ingredienti": "Cacao", "Prezzo": "2.00", "Tipo": "Dolce." }

METODO	URL	ENDPOINT	PARAMETRI	ESEMPIO JSON
Post	www.bfast.it/login	Login	ID, Password	{ "id": "1", "password": "123" } { "id": "155", "password": "321" }
Post	http://www.bfast.it/confermaid 	Registrazione	Nome, Cognome, Mail, Data	{ "Nome": "Mario", "Cognome": "Bianchi", "Mail": "mariob@bfast.it", "Data": "01.01.1991" } { "Nome": "Lucia", "Cognome": "verdi", "Mail": "lucyv@bfast.it", "Data": "15.06.1999" }
Post	www.bfast.it/confermaid	Conferma ID	ID	{ "Conferma ID": "123" } { "Conferma ID": "852" }
Post	www.bfast.it/cancellazionefattorino	Cancellazione Fattorino	ID	{ "ID": "12" } { "ID": "88" }
Post	www.bfast.it/cambiomail	Cambio mail	Nuova mail	{ "Mail": "mariob@bfast.it" } { "Nuova mail": "mbianchi@bfast.it" }
Post	www.bfast.it/cambiopassword	Cambio Password	Nuova password	{ "Password": "321" } { "Nuova Password": "456" }
Post	www.bfast.it/passworddimenticata	Password dimenticata	Cambia password	{ "Password dimenticata": "Cambia password" }
Post	www.bfast.it/scrividomanda	Scrivi Domanda	Domanda	{ "Scrivi domanda": "è possibile inserire un domicilio diverso?" } { "Scrivi domanda": "in quanto tempo posso consegnare l'ordine?" }
Post	www.bfast.it/scrivirisposta	Scrivi Risposta	Risposta	{ "Scrivi risposta": "si" } { "Scrivi risposta": "no" }
Post	www.bfast.it/confermaordine	Conferma Ordine	Conferma Ordine	{ "Conferma Ordine": "1" } { "Conferma Ordine": "6" }

```
public interface BfastUtenteApi {

    @POST("login")
    @FormUrlEncoded
    Call<Utente> login(@Query("mail")String mail, @Query("password") String pwd);

    @POST("registrazione")
    @FormUrlEncoded
    Call<Utente> registrazione(@Query("mail")String mail, @Query("password") String pwd,@Query("nome")String nome,
        @Query("cognome") String cognome, @Query("data") String data, @Query("telefono") String tel);

    @POST("CancellazioneUtente")
    @FormUrlEncoded
    Call<Utente> Cancellazione(@Query("mail")String mail);

    @POST("ConfermaMail")
    @FormUrlEncoded
    Call<Utente> ConfermaMail(@Query("mail")String mail);

    @POST("CambioMail")
    @FormUrlEncoded
    Call<Utente> CambioMail(@Query("mail")String mail);

    @POST("CambioPassword")
    @FormUrlEncoded
    Call<Utente> CambioPassword(@Query("password")String pass);
```

Model APP Package

*Nella modelAPP package ci sono
tutte le tabelle del db con tutti i
campi privati e i relativi getters
and setters*

Adapter Package

*Nella controller Package ci sono la
creazione e lo sviluppo dei DB
interni*

```

7 public class DatabaseHelper extends SQLiteOpenHelper {
8     private static final String DB_NAME = "database.db";
9     private static final int DB_VERSION = 1;
10    private static final String DB_CREATE = "create table Utente (email text primary key , nome text not null unique, " +
11        "password text not null, cognome text not null, telefono int not null,nascita Date not null);";
12    private static final String DB_CREATE2 = "create table Ordine (id int primary key autoincrement, orario text not null unique, " +
13        "note text not null, data Date not null, confermato bit not null,valutazioneFatt float not null," +
14        " FOREIGN KEY (\"+idUser+\") REFERENCES \"+Utente+\\"(\"+email+\"), FOREIGN KEY (\"+idBar+\") REFERENCES \"+Bar+\\"(\"+id+\")," +
15        "FOREIGN KEY (\"+idTipoPagamento+\") REFERENCES \"+Pagamento+\\"(\"+id+\\"));";
16    private static final String DB_CREATE3 = "create table Domanda (id integer primary key autoincrement, testo text not null unique);";
17    private static final String DB_CREATE4 = "create table Risposta (id integer primary key autoincrement, testo text not null unique);";
18    private static final String DB_CREATE5 = "create table Bar (id int primary key autoincrement, nome text not null unique, " +
19        "password text not null, orarioApe text not null,orarioChi text not null, fascia float not null,valutazione float not null,email text not null," +
20        "FOREIGN KEY(\"+idIndirizzo+\") REFERENCES \"+Indirizzo+\\"(\"+id+\\"));";
21    private static final String DB_CREATE6 = "create table Possiede (id integer primary key autoincrement," +
22        "FOREIGN KEY(\"+idDom+\") REFERENCES \"+Domanda+\\"(\"+id+\\"), FOREIGN KEY(\"+idRis+\") REFERENCES \"+Risposta+\\"(\"+id+\\"));";
23    private static final String DB_CREATE7 = "create table Contiene (id integer primary key autoincrement," +
24        "FOREIGN KEY(\"+idOrd+\") REFERENCES \"+Ordine+\\"(\"+id+\\"), FOREIGN KEY(\"+idPro+\") REFERENCES \"+Prodotto+\\"(\"+id+\\"));";
25    private static final String DB_CREATE8 = "create table Prodotto (nome text primary key ," +
26        "ingrediente text not null, costo float not null,tipo text not null);";
27    private static final String DB_CREATE9 = "create table Indirizzo (id integer primary key autoincrement,x double not null ," +
28        "y double not null);";
29    private static final String DB_CREATE10 = "create table Pagamento (id int primary key autoincrement, tipo text not null unique);";
30
31    public DatabaseHelper(Context context) { super(context, DB_NAME, factory: null, DB_VERSION); }
32
33    @Override
34    public void onCreate(SQLiteDatabase db) {
35        db.execSQL(DB_CREATE);
36        db.execSQL(DB_CREATE2);
37        db.execSQL(DB_CREATE3);
38        db.execSQL(DB_CREATE4);
39        db.execSQL(DB_CREATE5);
40        db.execSQL(DB_CREATE6);
41

```



```

49 public long addUser (String mail, String password, String nome, String cognome,String telefono,String nascita) {
50     ContentValues values = createContentValues(mail, password, nome, cognome,telefono,nascita);
51     return database.insertOrThrow( table: "user", nullColumnHack: null, values);
52 }
53
54 public Cursor getUserLogin(String mail) {
55     Cursor cursor = database.query( distinct: true, table: "user", new String[] { KEY_MAIL, KEY_PASSWORD},
56     selection: KEY_MAIL + "= '" + mail + "'", selectionArgs: null, groupBy: null, having: null, orderBy: null, limit: null);
57     return cursor;
58 }
59
60
61 public boolean updateUser(String mail, String password, String nome, String cognome, int telefono, long nascita) {
62     String tel = Integer.toString(telefono);
63     String nas = Integer.toString((int) nascita);
64     ContentValues updateValues = createContentValues(mail, password, nome, cognome,tel,nas);
65     return database.update( table: "user", updateValues, whereClause: KEY_MAIL + "=" + mail, whereArgs: null) > 0;
66 }
67
68 public boolean deleteUserByUsername(String s) {
69     return database.delete( table: "user", whereClause: KEY_MAIL + "= '" + s + "'", whereArgs: null) > 0;
70 }
71
72 public Cursor fetchUsers() {
73     return database.query( table: "user", new String[]
74     {KEY_MAIL, KEY_NOME, KEY_PASSWORD, KEY_COGNOME, KEY_TELEFONO,KEY_NASCITA},
75     selection: null, selectionArgs: null, groupBy: null, having: null, orderBy: null);
76 }
77
78 }

```

Session Package

*Nel Session Package ci sono tutte
le variabili che servono durante
tutto il ciclo di vita del programma
e che quindi salviamo in apposite
classi*

```
1 package com.ifts.bfastutente.Sessioni;
2
3 import android.content.Context;
4 import android.content.SharedPreferences;
5 import android.preference.PreferenceManager;
6
7 public class SessionUte {
8
9     private SharedPreferences prefs;
10
11     public SessionUte(Context cntx) {
12         // TODO Auto-generated constructor stub
13         prefs = PreferenceManager.getDefaultSharedPreferences(cntx);
14     }
15
16     public void setMailUt(String mail) { prefs.edit().putString("mail", mail).commit(); }
17
18
19
20     public String getMailUt() {
21         String mail = prefs.getString( key: "mail", defValue: "");
22         return mail;
23     }
24
25 }
```

Business Package

*Nella business class ci sono tutte le
operazioni che svolgiamo con java ,
ovvero la comunicazione tra
Layout, Java e DB*

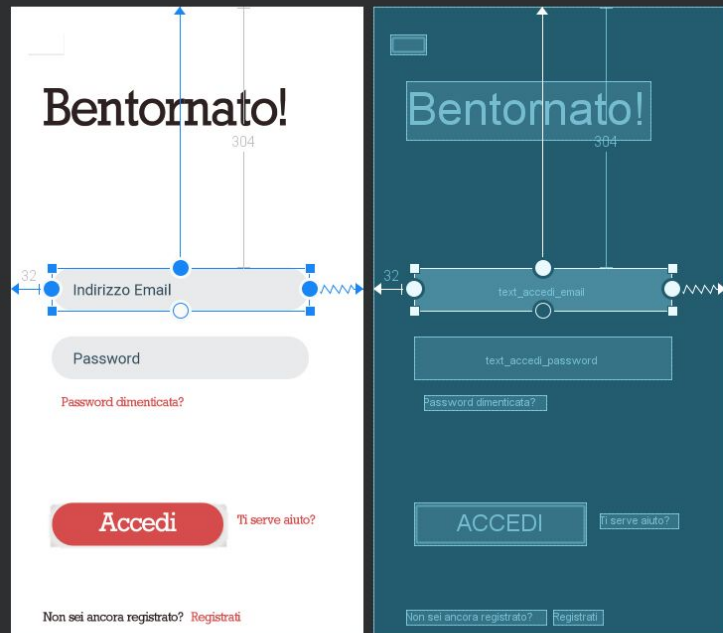
```

25 public class LoginActivity extends AppCompatActivity {
26     final UserDBAdapter udb = new UserDBAdapter( context: this);
27     private Button b1;
28     private TextView t1;
29     private TextView t2;
30     private SessionUte session;
31     BfastUtenteApi apiService = RetrofitUtils.getInstance().getBfastUtenteApi();
32
33     @Override
34     protected void onCreate(Bundle savedInstanceState) {
35         super.onCreate(savedInstanceState);
36         b1 = findViewById(R.id.BtnLogin);
37         t1 = findViewById(R.id.textView2);
38         t2 = findViewById(R.id.textView3);
39         b1.setOnClickListener((v) -> {
42             final EditText text1 = findViewById(R.id.ETmail);
43             EditText text2 = findViewById(R.id.ETpass);
44             Call<Utente> call = apiService.login(text1.toString(), text2.toString());
45             call.enqueue(new Callback<Utente>() {
46                 @Override
47                 public void onResponse(Call<Utente> call, Response<Utente> response) {
48                     Utente u = (Utente) udb.getUserLogin(text1.toString());
49                     session.setMailUt(text1.getText().toString());
50                     Intent log = new Intent( packageContext: LoginActivity.this, MapActivity.class);
51                     startActivity(log);
52                 }
53
54                 @Override
55                 public void onFailure(Call<Utente> call, Throwable t) {
56                     Toast.makeText( context: LoginActivity.this, text: "Errore nel login", Toast.LENGTH_LONG).show();
57                 }
58             });
59         });

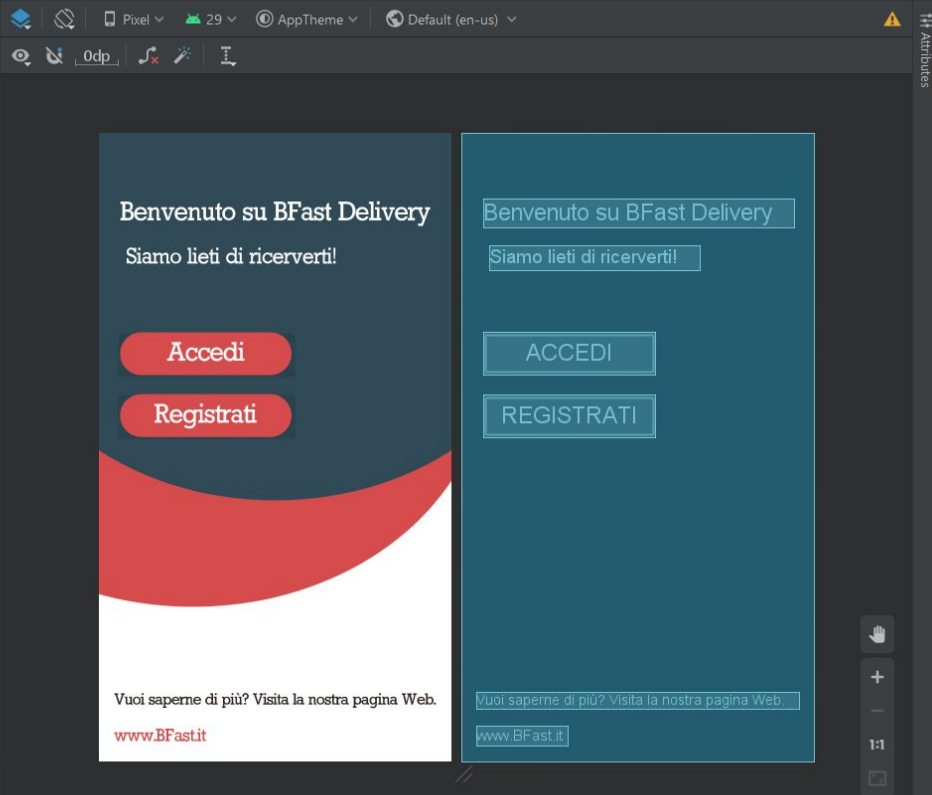
```

Frontend

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:id="@+id/accedi"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:background="@drawable/background_accedi"
9     tools:context=".Accedi">
10
11
12 <TextView
13     android:id="@+id/view_benvenuto2"
14     android:layout_width="wrap_content"
15     android:layout_height="wrap_content"
16     android:layout_marginStart="16dp"
17     android:layout_marginTop="32dp"
18     android:fontFamily="@font/rockwell"
19     android:text="Bentornato!"
20     android:textColor="@color/blackgray"
21     android:textSize="60dp"
22     app:layout_constraintEnd_toEndOf="parent"
23     app:layout_constraintHorizontal_bias="0.2"
24     app:layout_constraintStart_toStartOf="parent"
25     app:layout_constraintTop_toBottomOf="@+id/backBtn_Accedi" />
26
27 <Button
28     android:id="@+id/backBtn_Accedi"
29     android:layout_width="41dp"
30     android:layout_height="23dp"
31     android:layout_marginStart="2dp"
32     android:layout_marginTop="32dp"
33     android:background="@drawable/back"
34     app:layout_constraintEnd_toEndOf="parent"
35     app:layout_constraintHorizontal_bias="0.05"
36     app:layout_constraintStart_toStartOf="parent"
37     app:layout_constraintTop_toTopOf="parent" />
38
39 <EditText
40     android:id="@+id/text_accedi_email"
41     android:layout_width="300dp"
```



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:id="@+id/benvenuto"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:background="@drawable/background"
9     tools:context=".MainActivity">
10
11     <TextView
12         android:id="@+id/viewTesto_Benvenuto"
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content"
15         android:layout_marginStart="8dp"
16         android:layout_marginTop="296dp"
17         android:layout_marginEnd="8dp"
18         android:fontFamily="@font/pockwen"
19         android:text="Vuoi saperne di più? Visita la nostra pagina Web."
20         android:textColor="@color/blackgray"
21         android:textSize="18dp"
22         app:layout_constraintEnd_toEndOf="parent"
23         app:layout_constraintStart_toStartOf="parent"
24         app:layout_constraintTop_toBottomOf="@+id/registraBtn_Benvenuto" />
25
26     <TextView
27         android:id="@+id/viewRegistraBtn_Benvenuto"
28         android:layout_width="wrap_content"
29         android:layout_height="wrap_content"
30         android:layout_marginTop="20dp"
31         android:fontFamily="@font/pockwen"
32         android:text="www.BFast.it"
33         android:textColor="@color/red"
34         android:textSize="20dp"
35         app:layout_constraintEnd_toEndOf="parent"
36         app:layout_constraintHorizontal_bias="0.059"
37         app:layout_constraintStart_toStartOf="parent"
38         app:layout_constraintTop_toBottomOf="@+id/viewTesto_Benvenuto" />
39
40     <TextView
41         android:id="@+id/view_benvenuto"
```



Layout

Un layout definisce la struttura per un'interfaccia utente nell'app.

Tutti gli elementi nel layout sono creati usando una gerarchia di View e ViewGroup.
