

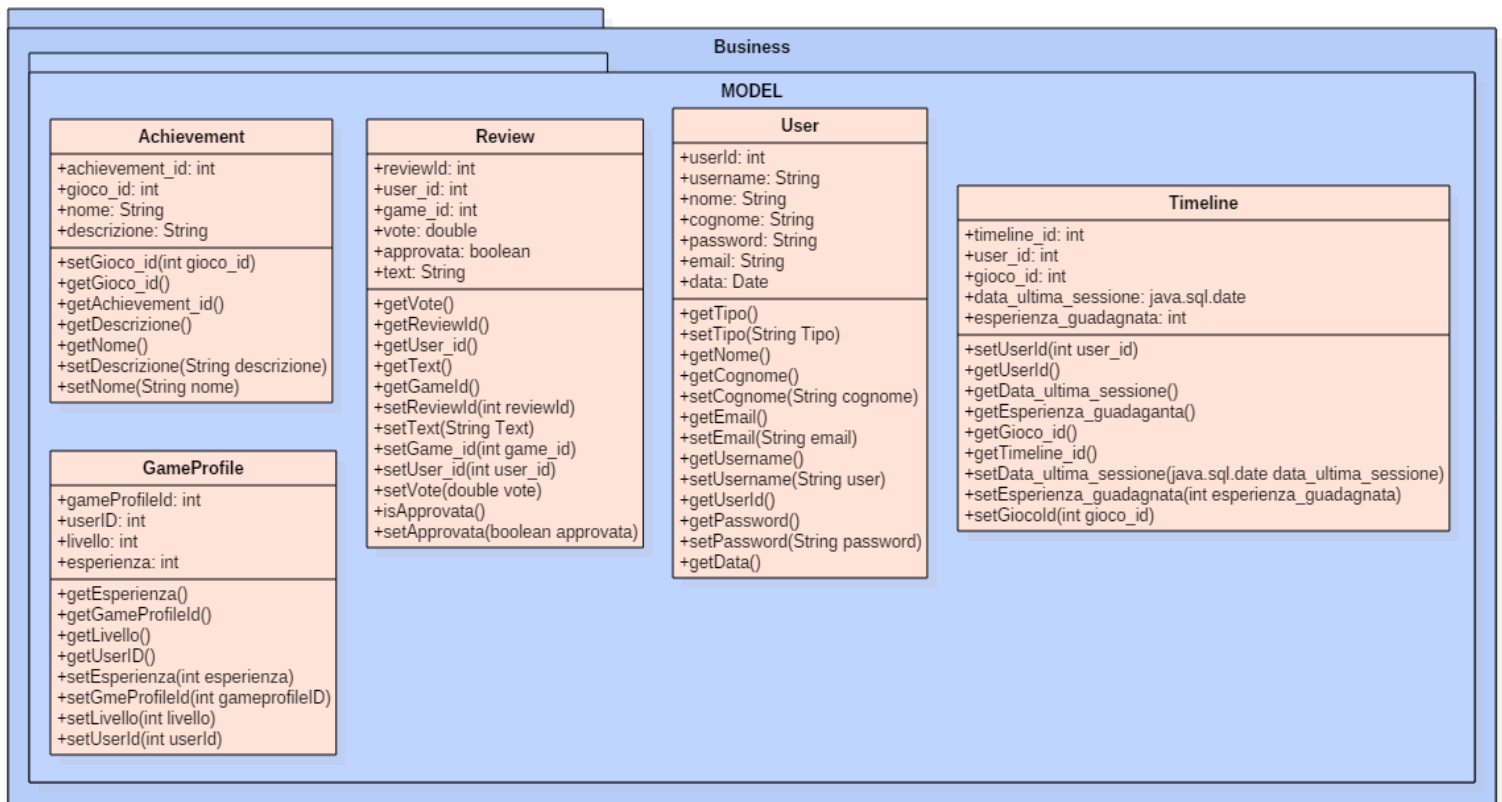
# CLASS DIAGRAM

L'elemento di modello principale del class diagram è la classe.

Una classe rappresenta una categoria di entità (istanze), nel caso particolare dette oggetti; il nome della classe indica la categoria di entità descritta in essa. Ogni classe è corredata da un insieme di attributi (che descrivono le caratteristiche o lo stato degli oggetti) e operazioni (che ne descrivono il comportamento).

Abbiamo deciso di strutturare il nostro sistema in tre package: Business, Listener e Presentation. Business a sua volta è diviso in Model e Implementation. In Model troviamo le classi principali, le quali rappresentano i protagonisti del sistema (Gli Oggetti) ovvero:

- Achievement;
- GameProfile;
- Review;
- User;
- Timeline;



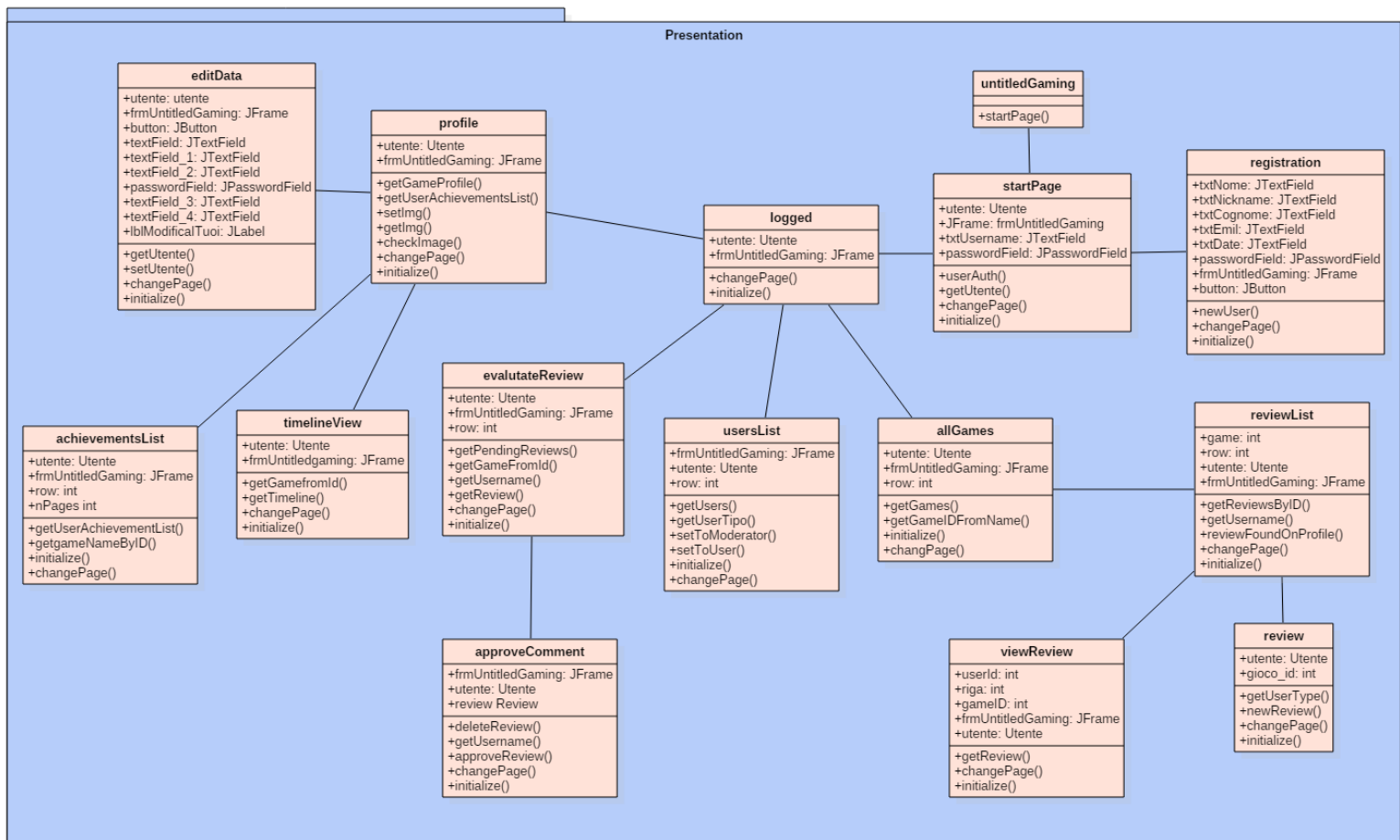
Per interfacciarsi al sistema ogni utente ha a disposizione una propria interfaccia grafica (GUI), la quale a sua volta è rappresentata da una classe.

Le classi Swing sono definite nel pacchetto javax.swing, il cui nome javax indica estensione standard a Java, ogni interfaccia è stata progettata per risultare il più possibile user-friendly.

Queste classi sono contenute nel package “Presentation”. Ogni oggetto grafico è predisposto ad essere sollecitato in qualche modo dall’utente e ad ogni sollecitazione genera eventi che vengono inoltrati ad appositi ascoltatori, che reagiscono agli eventi secondo i desideri del programmatore.

In “Listener” troviamo la classe Listener-Eventi, nella quale tutti gli eventi generati dall’utente mediante interfaccia grafica (ad esempio click di un bottone, inserimento di testo ecc...) sono catturati e gestiti opportunamente, permettendo la comunicazione con le altre componenti del sistema.

| Listener_Eventi   |
|---|
| +newUser(String username, String password, String nome, String cognome, String email, String dateString, int img_value, String tipo)<br>+setToNull(int userId)<br>+getUtente(String username)<br>+setUtente(Utente utente, String Nome, String cognome, String date, String email, String password, String username)<br>+newReview(Utente utente, String Review, int giocold, double vote, boolean approved)<br>+getTimeline(int userId)<br>+addTimeline(int userId, int gioco_id, java.sql.Date dataUltima, int esperienzaGuadagnata)<br>+getReview(int user_id, int game_id)<br>+getPendingReviews()<br>+getApprovedReviews()<br>+approveReview(Review review)<br>+addXP(Utente utente, int xP)<br>+addLivello(gameProfile gameProfile, int livello)<br>+checkLivello(gameProfile gameProfile)<br>+getGames()<br>+getGameProfile(int user_id)<br>+updateTimeline(int user_id, java.sql.Date dataUltima, int esperienzaGuadagnata, int gioco_id)<br>+checkAchievement(gamProfile gameProfile)<br>+AchievementFoundOnProfile(int achievements_id, int user_id)<br>+getAllReviews()<br>+reviewFoundOnProfile(int user_id, int achievement_id)<br>+insertAchievementToProfile(int user_id, int achievement_id)<br>+deleteReview(Review review)<br>+getUserAchievementsList(int UserId)<br>+setToModerator(String username)<br>+setToUser(String username)<br>+setToAdministrator(String username)<br>+getGameNameById(int gameId)<br>+updatePassword(int userId, String password)<br>+setImg(int userID, File Imgfile)<br>+getImg(int userID, String filePathName)<br>+checkImage(int userID)<br>+getUserID(String user)<br>+getUserTipo(String user)<br>+getUsername(int user_id)<br>+getGameFromId(int Game_id)<br>+userAuth(String username, String password)<br>+getGame()<br>+getUsers()<br>+getGameIDFromName(String gameName)<br>+getReviewsById(int gameId)<br>+cangePage(String page, Utente utente) |



La classe “Listener-Eventi” chiama dei metodi implementati nel package “Implementation”, dove troviamo le classi che servono a interagire con il database:

- **“ReviewManagement”**: adibita a gestire i metodi riguardanti le recensioni. Tra le altre funzioni è possibile aggiungere o eliminare un oggetto di tipo “recensione” e gestirne i vari aspetti.

- **“UserManagement”**: essa gestisce i metodi riguardanti l’utenza e quindi l’aggiunta di un nuovo utente, la promozione o retrocessione di esso, la visualizzazione della timeline relativa all’utente, ecc.

- **“DBManager”**: è una classe che serve a instaurare una connessione tra il database e il sistema. La connessione è effettuata tramite un oggetto Connection dichiarato *static*, in modo da avere un’unica connessione al database per tutto il sistema. All’interno della classe sono presenti altre funzioni relative al check di password (tramite verifica degli hash), username (per evitare che vengano registrati 2 utenti con lo stesso username) ed email (per far sì che venga inserito un indirizzo email valido).

In ogni chiamata viene gestita un’eccezione, tramite la creazione di un nuovo oggetto della classe **“BusinessException”**, che si trova nel package Business. Questa classe si occupa di

avvertire, tramite un messaggio di errore, l'utente nel caso si verifichi un qualsiasi problema nell'applicazione.

