hierarchy

```
HOTELIER/
│
├── src/
│   ├── client/
│   │   ├── HOTELIERCustomerClient.java
│   │   │   ├── void start()                          # Avvia il client e
l'interfaccia a linea di comando
│   │   │   ├── void register(String username, String password)   # Registra un
nuovo utente
│   │   │   ├── void login(String username, String password)      # Effettua il
login di un utente esistente
│   │   │   ├── void logout(String username)                      # Effettua il
logout dell'utente
│   │   │   ├── void searchHotel(String nomeHotel, String città)  # Cerca un
hotel per nome e città
│   │   │   ├── void searchAllHotels(String città)                # Cerca tutti
gli hotel di una città ordinati per ranking
│   │   │   ├── void insertReview(String nomeHotel, String nomeCittà, double
globalScore, Map<String, Integer> singleScores)  # Inserisce una recensione per
un hotel
│   │   │   ├── void showMyBadges()                               # Mostra i
distintivi dell'utente
│   │   │
│   │   ├── TCPConnectionHandler.java
│   │   │   ├── void connect(String serverAddress, int serverPort)    #
Stabilisce una connessione TCP con il server
│   │   │   ├── void sendRequest(String request)                     # Invia una
richiesta al server
│   │   │   ├── String receiveResponse()                             # Riceve una
risposta dal server
│   │   │   ├── void disconnect()                                    #
Disconnette dal server
│   │   │
│   │   ├── NotificationHandler.java
│   │       ├── void startListening(int port)                        # Inizia ad
ascoltare le notifiche dal server
│   │       ├── void handleNotification(String notification)         # Gestisce
una notifica ricevuta dal server
│   │
│   ├── server/
│   │   ├── HOTELIERServer.java
│   │   │   ├── void start()                                         # Avvia il
server
│   │   │   ├── void loadData()                                      # Carica i
```

```
dati degli utenti e degli hotel dai file JSON
|   |   |   ├── void saveData()                                        # Salva i
dati degli utenti e degli hotel in file JSON
|   |   |
|   |   ├── UserManagement.java
|   |   |   ├── String register(String username, String password)    # Registra
un nuovo utente
|   |   |   ├── String login(String username, String password)       # Effettua
il login di un utente esistente
|   |   |   ├── String logout(String username)                       # Effettua
il logout dell'utente
|   |   |   ├── User getUser(String username)                        # Ottiene i
dettagli di un utente
|   |   |
|   |   ├── HotelManagement.java
|   |   |   ├── void loadHotels(String filePath)                     # Carica le
informazioni sugli hotel da un file JSON
|   |   |   ├── List<Hotel> searchHotel(String nomeHotel, String città) # Cerca
un hotel per nome e città
|   |   |   ├── List<Hotel> searchAllHotels(String città)            # Cerca
tutti gli hotel di una città ordinati per ranking
|   |   |   ├── void addReview(String nomeHotel, String nomeCittà, Review review)
# Aggiunge una recensione per un hotel
|   |   |   ├── void updateRankings()                                # Aggiorna
il ranking degli hotel
|   |   |
|   |   ├── ReviewManagement.java
|   |   |   ├── void addReview(String nomeHotel, String nomeCittà, Review review)
# Aggiunge una recensione per un hotel
|   |   |   ├── List<Review> getReviews(String nomeHotel, String nomeCittà) #
Ottiene tutte le recensioni per un hotel
|   |   |
|   |   ├── RankingCalculator.java
|   |   |   ├── double calculateRankingScore(double quality, int quantity, double
recencyWeight) # Calcola il punteggio di ranking di un hotel
|   |   |   ├── void updateRankings(List<Hotel> hotels)              # Aggiorna
il ranking degli hotel
|   |   |
|   |   ├── NotificationService.java
|   |   |   ├── void sendNotification(String message)                # Invia una
notifica agli utenti registrati
|   |   |
|   |   ├── DataPersistence.java
|   |       ├── void saveUsers(Map<String, User> users, String filePath) # Salva
i dati degli utenti in un file JSON
|   |       ├── void saveHotels(List<Hotel> hotels, String filePath)  # Salva i
dati degli hotel in un file JSON
|   |       ├── Map<String, User> loadUsers(String filePath)         # Carica i
dati degli utenti da un file JSON
```

```
|   |       ├── List<Hotel> loadHotels(String filePath)             # Carica i
dati degli hotel da un file JSON
|
|   ├── models/
|   |   ├── User.java
|   |   |   ├── String getUsername()                                #
Restituisce il nome utente
|   |   |   ├── String getPassword()                                #
Restituisce la password (hash)
|   |   |   ├── int getReviews()                                    #
Restituisce il numero di recensioni dell'utente
|   |   |   ├── String getBadge()                                   #
Restituisce il distintivo dell'utente
|   |   |   ├── void incrementReviews()                             # Incrementa
il numero di recensioni dell'utente
|   |   |   ├── void updateBadge(String newBadge)                   # Aggiorna
il distintivo dell'utente
|   |   |
|   |   ├── Hotel.java
|   |       ├── String getName()                                    #
Restituisce il nome dell'hotel
|   |       ├── String getCity()                                    #
Restituisce la città dell'hotel
|   |       ├── double getGlobalScore()                             #
Restituisce il punteggio globale dell'hotel
|   |       ├── Map<String, Integer> getScores()                    #
Restituisce i punteggi specifici dell'hotel
|   |       ├── int getReviews()                                    #
Restituisce il numero di recensioni dell'hotel
|   |       ├── int getRank()                                       #
Restituisce il ranking dell'hotel
|   |       ├── void updateGlobalScore(double newScore)             # Aggiorna
il punteggio globale dell'hotel
|   |       ├── void updateScores(Map<String, Integer> newScores)   # Aggiorna i
punteggi specifici dell'hotel
|   |       ├── void incrementReviews()                             # Incrementa
il numero di recensioni dell'hotel
|   |       ├── void updateRank(int newRank)                        # Aggiorna
il ranking dell'hotel
|
|   ├── utils/
|   |   ├── JsonUtil.java
|   |       ├── String toJson(Object obj)                           # Converte
un oggetto in formato JSON
|   |       ├── <T> T fromJson(String json, Class<T> classOfT)      # Converte
una stringa JSON in un oggetto
|
|   ├── config/
|   |   ├── server_config.json                                      #
```

```
Configurazione del server (porta, intervallo aggiornamento ranking, etc.)
│   │   ├── client_config.json                                    #
Configurazione del client (indirizzo e porta del server)
│
├── tests/
│   ├── client/
│   │   ├── HOTELIERCustomerClientTest.java
│   │       ├── void testRegister()                               # Testa la
registrazione di un nuovo utente
│   │       ├── void testLogin()                                  # Testa il
login di un utente esistente
│   │       ├── void testLogout()                                 # Testa il
logout di un utente
│   │       ├── void testSearchHotel()                            # Testa la
ricerca di un hotel
│   │       ├── void testSearchAllHotels()                        # Testa la
ricerca di tutti gli hotel in una città
│   │       ├── void testInsertReview()                           # Testa
l'inserimento di una recensione
│   │       ├── void testShowMyBadges()                           # Testa la
visualizzazione dei distintivi
│   │
│   ├── server/
│   │   ├── HOTELIERServerTest.java
│   │       ├── void testStart()                                  # Testa
l'avvio del server
│   │       ├── void testLoadData()                               # Testa il
caricamento dei dati
│   │       ├── void testSaveData()                               # Testa il
salvataggio dei dati
│   │   ├── UserManagementTest.java
│   │       ├── void testRegister()                               # Testa la
registrazione di un nuovo utente
│   │       ├── void testLogin()                                  # Testa il
login di un utente esistente
│   │       ├── void testLogout()                                 # Testa il
logout di un utente
│   │   ├── HotelManagementTest.java
│   │       ├── void testLoadHotels()                             # Testa il
caricamento delle informazioni sugli hotel
│   │       ├── void testSearchHotel()                            # Testa la
ricerca di un hotel
│   │       ├── void testSearchAllHotels()                        # Testa la
ricerca di tutti gli hotel in una città
│   │   ├── ReviewManagementTest.java
│   │       ├── void testAddReview()                              # Testa
l'aggiunta di una recensione per un hotel
│   │       ├── void testGetReviews()                             # Testa
l'ottenimento delle recensioni per un hotel
```

```
│   │   ├── RankingCalculatorTest.java
│   │       ├── void testCalculateRankingScore()                    # Testa il
calcolo del punteggio di ranking di un hotel
│   │       ├── void testUpdateRankings()                           # Testa
l'aggiornamento dei ranking degli hotel
│
├── lib/
│   ├── gson-2.8.6.jar                                              # Libreria
Gson per la manipolazione JSON
│
├── docs/
│   ├── ProjectSpecification.pdf                                    # Specifiche
di progetto
│   ├── UserManual.pdf                                              # Manuale
utente
│
├── build/
│   ├── HOTELIERServer.jar                                          # File JAR
eseguibile per il server
│   ├── HOTELIERCustomerClient.jar                                  # File JAR
eseguibile per il client
│
├── README.md                                                       #
Descrizione del progetto e istruzioni per l'uso
├── LICENSE                                                         # Licenza
del progetto
```

Responsabilità dettagliate di ciascun file/componente

`client` Package

- ### HOTELIERCustomerClient.java
  - `void start()` : Avvia il client e l'interfaccia a linea di comando.
  - `void register(String username, String password)` : Registra un nuovo utente.
  - `void login(String username, String password)` : Effettua il login di un utente esistente.
  - `void logout(String username)` : Effettua il logout dell'utente.
  - `void searchHotel(String nomeHotel, String città)` : Cerca un hotel per nome e città.
  - `void searchAllHotels(String città)` : Cerca tutti gli hotel di una città ordinati per ranking.
  - `void insertReview(String nomeHotel, String nomeCittà, double globalScore, Map<String, Integer> singleScores)` : Inserisce una recensione per un hotel.
  - `void showMyBadges()` : Mostra i distintivi dell'utente.
- ### TCPConnectionHandler.java

- `void connect(String serverAddress, int serverPort)` : Stabilisce una connessione TCP con il server.
- `void sendRequest(String request)` : Invia una richiesta al server.
- `String receiveResponse()` : Riceve una risposta dal server.
- `void disconnect()` : Disconnette dal server.

- **NotificationHandler.java**
  - `void startListening(int port)` : Inizia ad ascoltare le notifiche dal server.
  - `void handleNotification(String notification)` : Gestisce una notifica ricevuta dal server.

`server` Package

- **HOTELIERServer.java**
  - `void start()` : Avvia il server.
  - `void loadData()` : Carica i dati degli utenti e degli hotel dai file JSON.
  - `void saveData()` : Salva i dati degli utenti e degli hotel in file JSON.
- **UserManagement.java**
  - `String register(String username, String password)` : Registra un nuovo utente.
  - `String login(String username, String password)` : Effettua il login di un utente esistente.
  - `String logout(String username)` : Effettua il logout dell'utente.
  - `User getUser(String username)` : Ottiene i dettagli di un utente.
- **HotelManagement.java**
  - `void loadHotels(String filePath)` : Carica le informazioni sugli hotel da un file JSON.
  - `List<Hotel> searchHotel(String nomeHotel, String città)` : Cerca un hotel per nome e città.
  - `List<Hotel> searchAllHotels(String città)` : Cerca tutti gli hotel di una città ordinati per ranking.
  - `void addReview(String nomeHotel, String nomeCittà, Review review)` : Aggiunge una recensione per un hotel.
  - `void updateRankings()` : Aggiorna il ranking degli hotel.
- **ReviewManagement.java**
  - `void addReview(String nomeHotel, String nomeCittà, Review review)` : Aggiunge una recensione per un hotel.
  - `List<Review> getReviews(String nomeHotel, String nomeCittà)` : Ottiene tutte le recensioni per un hotel.
- **RankingCalculator.java**
  - `double calculateRankingScore(double quality, int quantity, double recencyWeight)` : Calcola il punteggio di ranking di un hotel.

- `void updateRankings(List<Hotel> hotels)` : Aggiorna il ranking degli hotel.

- **NotificationService.java**
  - `void sendNotification(String message)` : Invia una notifica agli utenti registrati.
- **DataPersistence.java**
  - `void saveUsers(Map<String, User> users, String filePath)` : Salva i dati degli utenti in un file JSON.
  - `void saveHotels(List<Hotel> hotels, String filePath)` : Salva i dati degli hotel in un file JSON.
  - `Map<String, User> loadUsers(String filePath)` : Carica i dati degli utenti da un file JSON.
  - `List<Hotel> loadHotels(String filePath)` : Carica i dati degli hotel da un file JSON.

`models` Package

- **User.java**
  - `String getUsername()` : Restituisce il nome utente.
  - `String getPassword()` : Restituisce la password (hash).
  - `int getReviews()` : Restituisce il numero di recensioni dell'utente.
  - `String getBadge()` : Restituisce il distintivo dell'utente.
  - `void incrementReviews()` : Incrementa il numero di recensioni dell'utente.
  - `void updateBadge(String newBadge)` : Aggiorna il distintivo dell'utente.
- **Hotel.java**
  - `String getName()` : Restituisce il nome dell'hotel.
  - `String getCity()` : Restituisce la città dell'hotel.
  - `double getGlobalScore()` : Restituisce il punteggio globale dell'hotel.
  - `Map<String, Integer> getScores()` : Restituisce i punteggi specifici dell'hotel.
  - `int getReviews()` : Restituisce il numero di recensioni dell'hotel.
  - `int getRank()` : Restituisce il ranking dell'hotel.
  - `void updateGlobalScore(double newScore)` : Aggiorna il punteggio globale dell'hotel.
  - `void updateScores(Map<String, Integer> newScores)` : Aggiorna i punteggi specifici dell'hotel.
  - `void incrementReviews()` : Incrementa il numero di recensioni dell'hotel.
  - `void updateRank(int newRank)` : Aggiorna il ranking dell'hotel.

`utils` Package

- **JsonUtil.java**
  - `String toJson(Object obj)` : Converte un oggetto in formato JSON.

- `<T> T fromJson(String json, Class<T> classOfT)` : Converte una stringa JSON in un oggetto.

`config` Package

- **server_config.json**: Configurazione del server (porta, intervallo aggiornamento ranking, etc.).
- **client_config.json**: Configurazione del client (indirizzo e porta del server).

`tests` Package

- **HOTELIERCustomerClientTest.java**
  - `void testRegister()` : Testa la registrazione di un nuovo utente.
  - `void testLogin()` : Testa il login di un utente esistente.
  - `void testLogout()` : Testa il logout di un utente.
  - `void testSearchHotel()` : Testa la ricerca di un hotel.
  - `void testSearchAllHotels()` : Testa la ricerca di tutti gli hotel in una città.
  - `void testInsertReview()` : Testa l'inserimento di una recensione.
  - `void testShowMyBadges()` : Testa la visualizzazione dei distintivi.
- **HOTELIERServerTest.java**
  - `void testStart()` : Testa l'avvio del server.
  - `void testLoadData()` : Testa il caricamento dei dati.
  - `void testSaveData()` : Testa il salvataggio dei dati.
- **UserManagementTest.java**
  - `void testRegister()` : Testa la registrazione di un nuovo utente.
  - `void testLogin()` : Testa il login di un utente esistente.
  - `void testLogout()` : Testa il logout di un utente.
- **HotelManagementTest.java**
  - `void testLoadHotels()` : Testa il caricamento delle informazioni sugli hotel.
  - `void testSearchHotel()` : Testa la ricerca di un hotel.
  - `void testSearchAllHotels()` : Testa la ricerca di tutti gli hotel in una città.
- **ReviewManagementTest.java**
  - `void testAddReview()` : Testa l'aggiunta di una recensione per un hotel.
  - `void testGetReviews()` : Testa l'ottenimento delle recensioni per un hotel.
- **RankingCalculatorTest.java**
  - `void testCalculateRankingScore()` : Testa il calcolo del punteggio di ranking di un hotel.
  - `void testUpdateRankings()` : Testa l'aggiornamento dei ranking degli hotel.