

Nome: Simone Staccone  
Matricola: 0279340

# **PROGETTO ISPW**

**ANNO 2021/2022**

**Italian Ballerz**

## 1. Introduction

- 1.1. Aim of the document
  - 1.1.1. Overview of the defined system
  - 1.1.2. Operational settings
  - 1.1.3. Related systems (at least 2), Pros and Cons.
- 1.2. User Stories (3 per member)
- 1.3. Functional Requirements (3 per member)
- 1.4. Use Cases: Overview Diagram (1)

## 2. Storyboards

- 2.1. 2 screens per member, covering all the functionalities described in SRS, developed using Draw.io or similar

## 3. Design

- 3.1. Class diagram:
  - 3.1.1.1 VOPC per member.(analysis)
  - 3.1.2.1 design-level diagram per member (e.g. that includes patterns, or specific solutions that improve the engineering level of the system)
- 3.2. Design patterns: 1 different pattern per member. Possibly try to apply the pattern within the context of the project.
- 3.3. Activity diagram: 1 per member.
- 3.4. Sequence diagram: 1 per member.
- 3.5. State diagram: 1 per member.

## 4. Testing

- 4.1. Develop at least 3 test cases per person. In each test (class) file, please report (via Java comments) the name of the person in charge.

## 5. Code

- 5.1. Exceptions: at least 2 per member (do not just catch and back-propagate the exceptions, but properly handle them. Possibly define your own error logic by means of exceptions)

## 6. Analytics

- 6.1 Provide a Proces Control Chart as explained in the slides.

## 7. Video

- 7.1 A 1 to 2 minutes recorded video of the developed system performing the expected functionalities.

# 1. Introduction

## 1.1 Aim of the document

Lo scopo di questo documento è quello di esplicitare i vari passi della progettazione nella realizzazione del software Italian Ballerz, come progetto per l'esame di Ingegneria del software.

### 1.1.1 Overview of the defined system

Il software realizzato si occupa della gestione delle prenotazioni per campi da basket. Inoltre, si vuole far avere un account all'utente in grado di tener traccia delle proprie attività, come ad esempio permettere all'utente di salvare le proprie statistiche.

### 1.1.2 Operational settings

Per lo sviluppo del software ho usato:

- Figma.com per ottenere il codice HTML delle storyboard.
- Star UML per la realizzazione dei diagrammi relativi alla progettazione del sistema
- Scene Builder per la creazione di file FXML per realizzare l'interfaccia grafica (e quindi le view del MVC)
- JavaFX per gestire e ampliare i file FXML (controller grafico del MVC)
- IntelliJ e in particolare Java come IDE e ambiente di programmazione. Per utilizzare il software si deve estrarre la cartella ItalianBallerz dal file zip e aprirla come progetto in IntelliJ. Nella cartella è già presente il file .idea per configurare come fare il run dell'applicazione. Se questo non dovesse essere possibile la classe con la responsabilità di avviare l'applicazione è MainInterface.java .

### 1.1.3 Related systems

App correlate "CourtFinder" e "Pick Roll"

- CourtFinder
  - Pro: ItalianBallerz offre la possibilità di tener traccia delle proprie statistiche e di prenotare il posto al campo
  - Contro: CourtFinder fornisce una mappa interattiva da cui scegliere e permette di salvare i campi preferiti.
- Pick Roll
  - Pro: ItalianBallerz offre la possibilità di tener traccia delle proprie statistiche e di prenotare il posto al campo

- Contro: Pickle Roll permette di verificare la credibilità di un utente in base ad un sistema di valuta. Inoltre mostra immagini relative ai campetti.

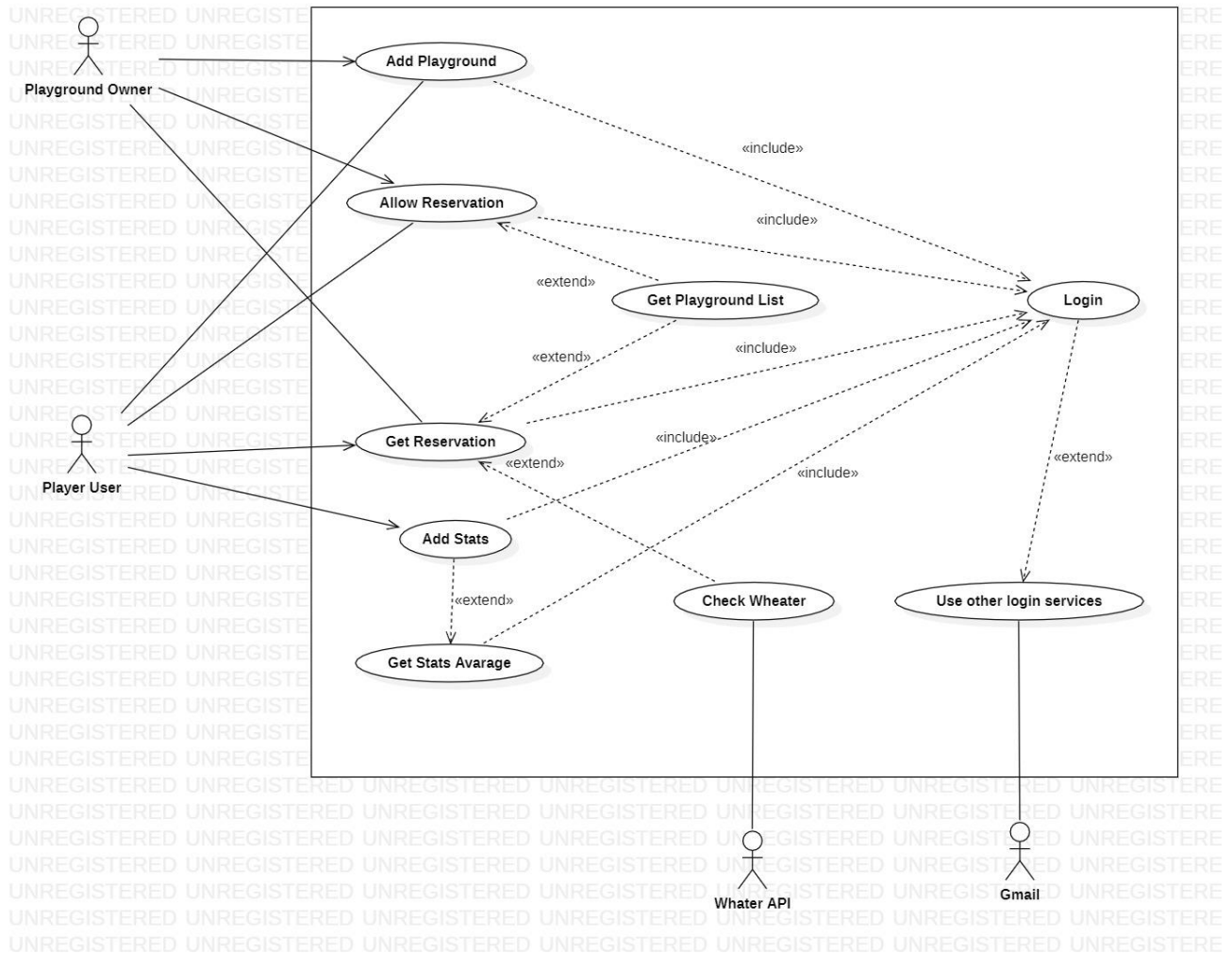
## 1.2 User stories

- **As a user, I want to search for basketball court next to me, so that I can play with other people. (Implementato)**
- **As a registered user, I want to track my basketball statistics, so that I can see if I'm improving or not. (Implementato)**
- **As a registered user, I want to log in, so that I can enter my profile. (Implementato)**
- As a user, I want to inform when I play in a specific court, so that everyone could know if the court is not empty and join me. (Non implementato)
- As a user, I want to review a court, so that everyone knows which court is better. (Non implementato)
- As a not so skilled basketball player user, I want to find other players at my same level, so that I will have fun while playing. (Non implementato)

## 1.3 Functional requirements

- **The system shall provide a list and a map on which are located the basketball courts in a city of choice. (Implementato tranne mappa)**
- **The system shall provide registration using an email, a password and a username. (Implementato)**
- **The system shall provide to insert the statistics of a registered user:**
  - a. **Points**
  - b. **Rebounds**
  - c. **Assists**
  - d. **Minutes****(Implementato)**
- The system shall provide log in using a password and a username. (Implementato)
- The system shall provide rated reviews, using a rating scale based stars, from 1 star up to 5. (Non implementato)
- The system shall provide a schedule for each registered court to be shown to both registered and guest users. (Implementato)

## 1.4 Use case overview diagram




## 2. Storyboard (Il file HTML si trova nella cartella StoryBoard)

HomeCampettiStatistiche


ITALIAN BALLER2

Trova il tuo campo!



Naviga e cerca tutti i campi nelle tue vicinanze. Controlla chi è già al campo e non perdere tempo, pensa solo a giocare!

Salva le tue statistiche!



Salva le tue statistiche, comparale con i tuoi amici e controlla i tuoi miglioramenti!

HomeCampettiStatistiche

ITALIAN BALLER2

Nuovo Utente?  
Registrati

Username

Indirizzo e-mail

Password

Conferma Password

Annulla

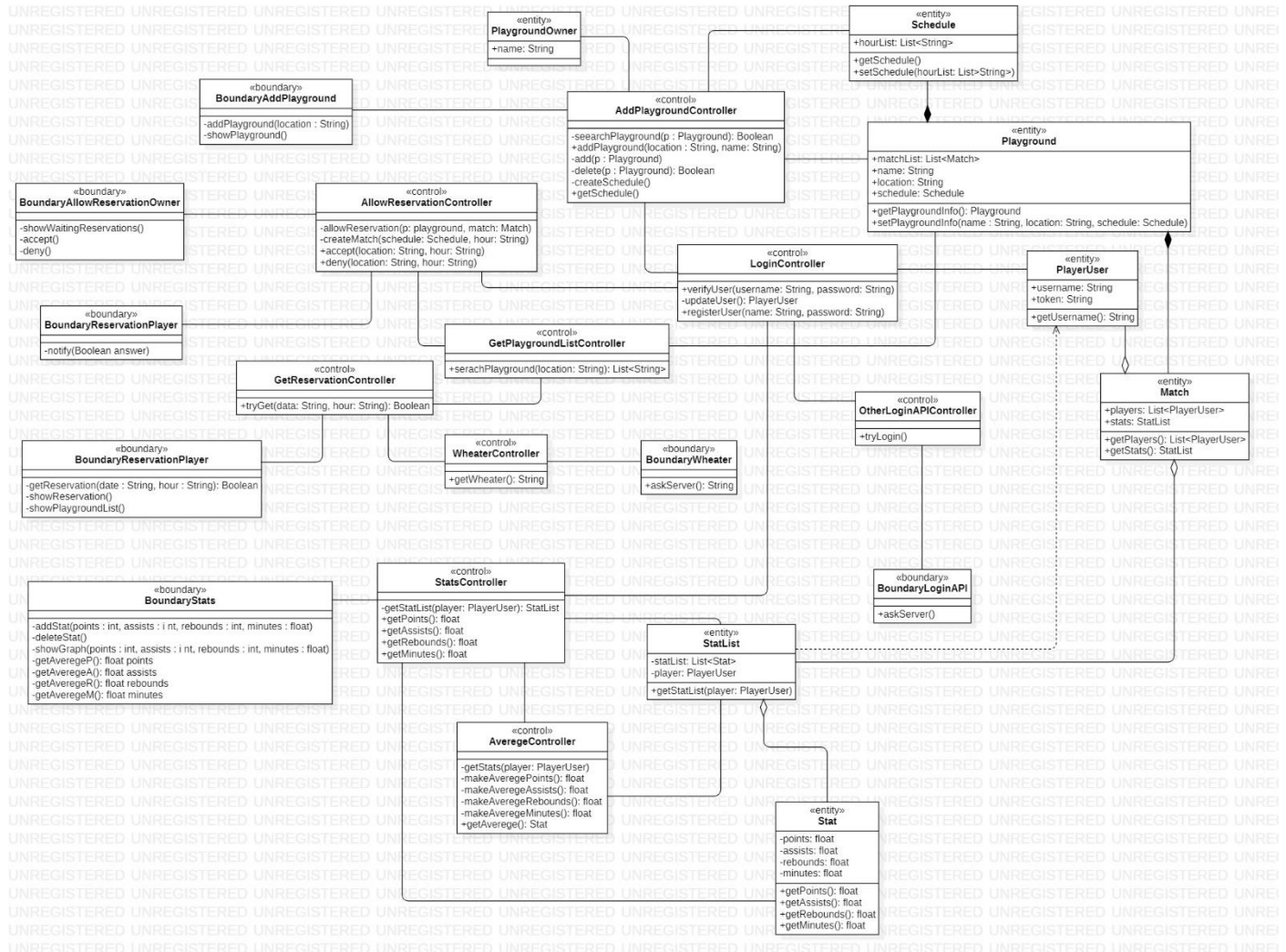
Registrati

Nome: Simone Staccone  
Matricola: 0279340

### 3. Design

### 3.1 Class diagram

### 3.1.1 VOPC (BCE class diagram)



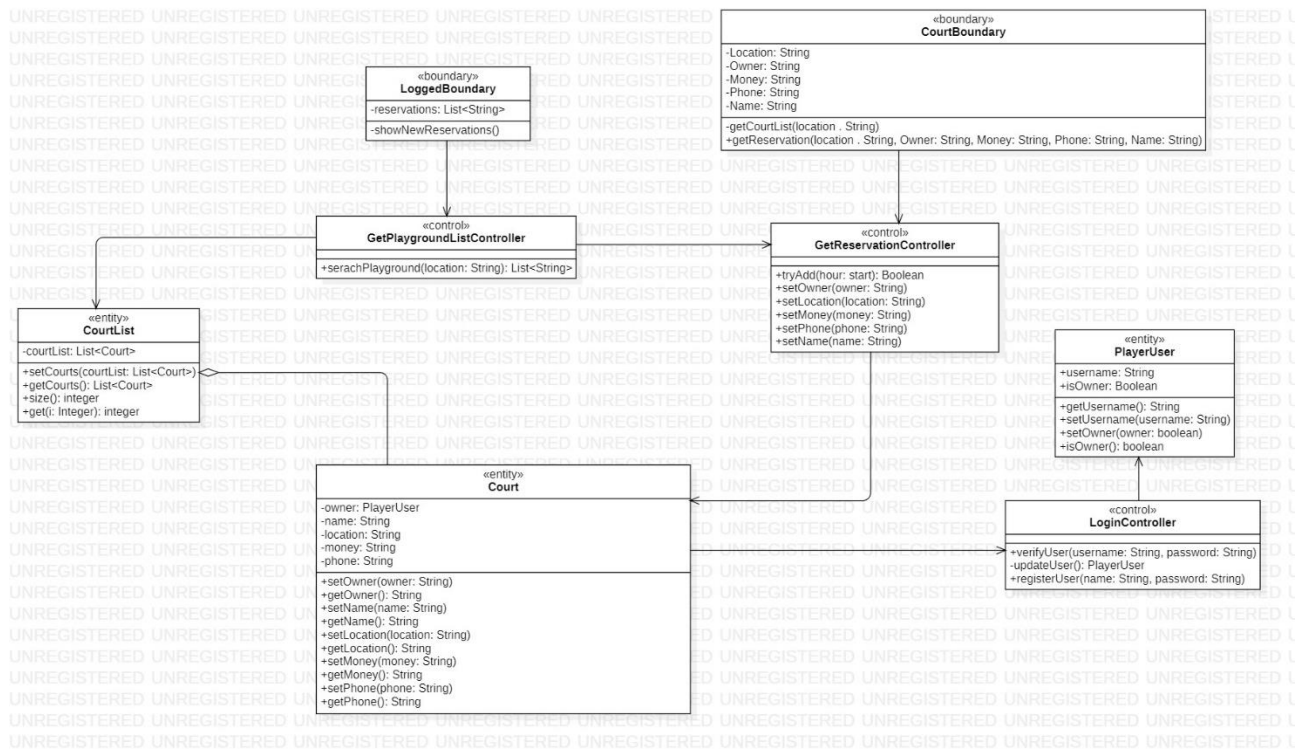
Questo class diagram a livello BCE è un class diagram realizzato prima dell’inizio della scrittura del codice, finalizzato ad avere una visione complessiva del sistema.

Durante la stesura del codice molte cose sono cambiate a livello di class diagram corrispondente, sia per il tempo disponibile, sia per la raffinazione del diagramma stesso in una visione di tipo MVC.

Per cui il progetto finale è simile a questo schema, ma non uguale.

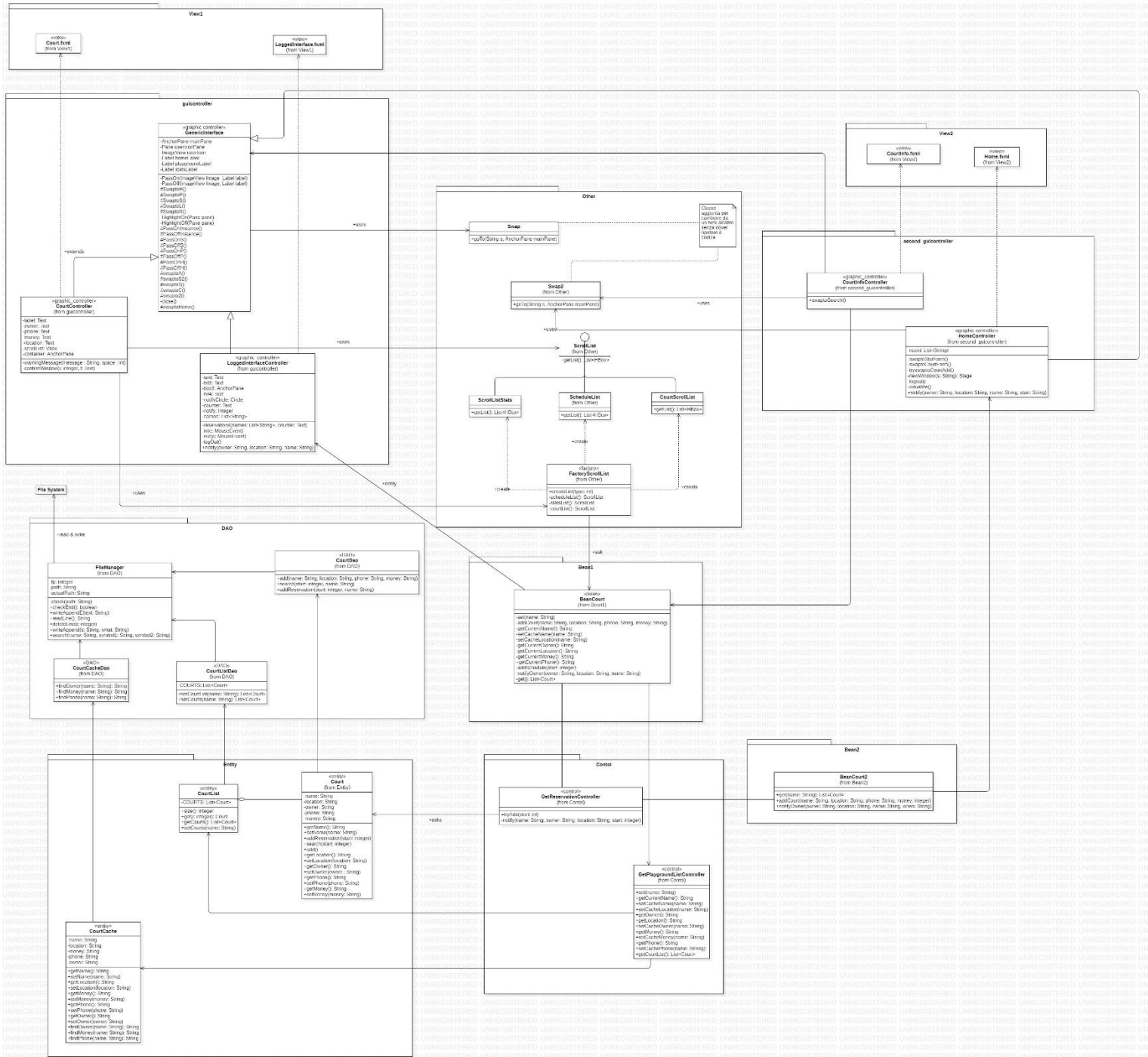
Di seguito riporto lo schema BCE dello use case relativo alla prenotazione di un campo (interessante per l'interazione tra due attori).

Nome: Simone Staccone  
Matricola: 0279340





### 3.1.2 Design-level diagram (MVC diagram)



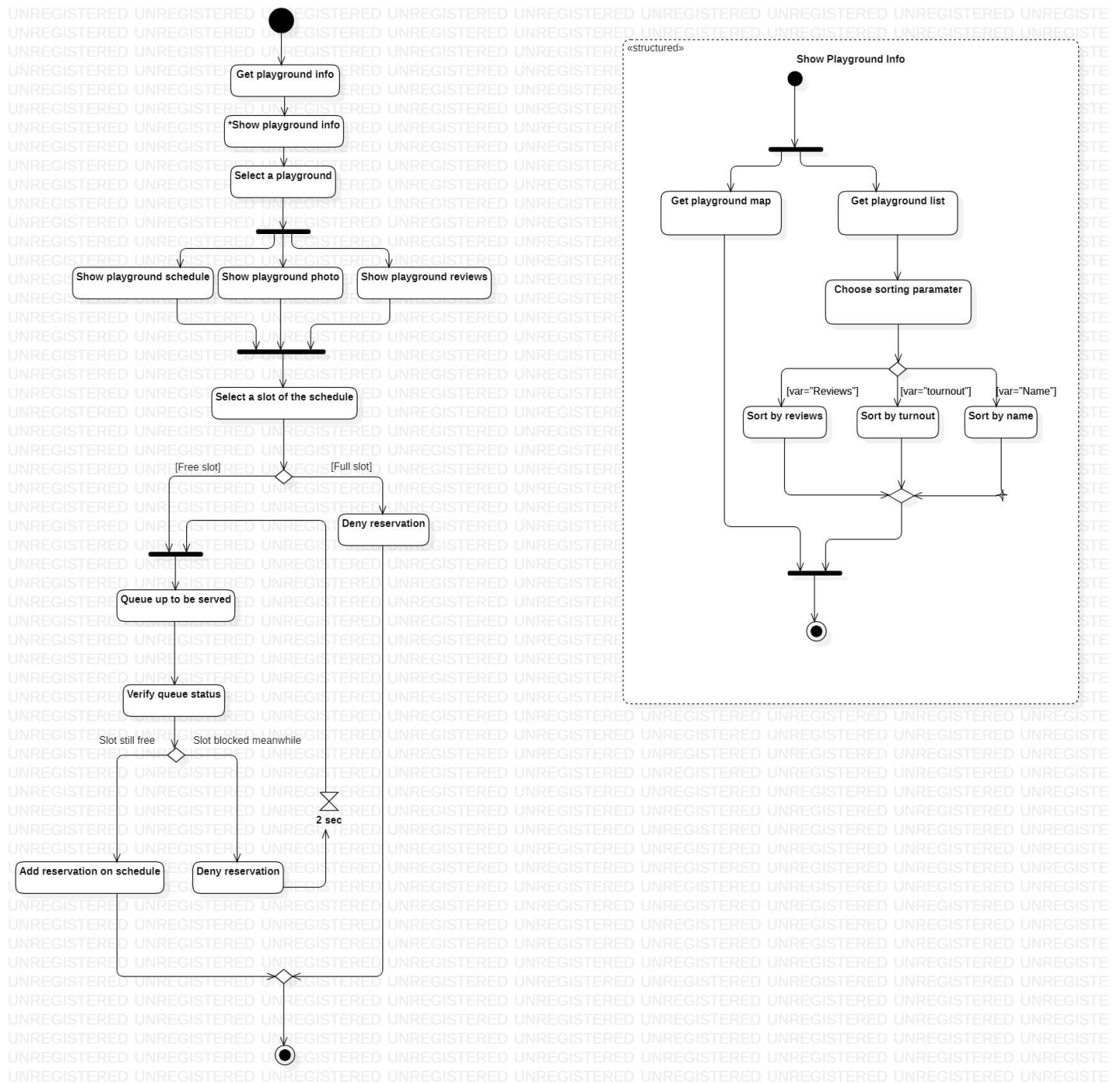
Questo è lo schema MVC completo relativo al progetto realizzato dello use case relativo alla richiesta di una prenotazione per un campo.

### 3.2 Design pattern

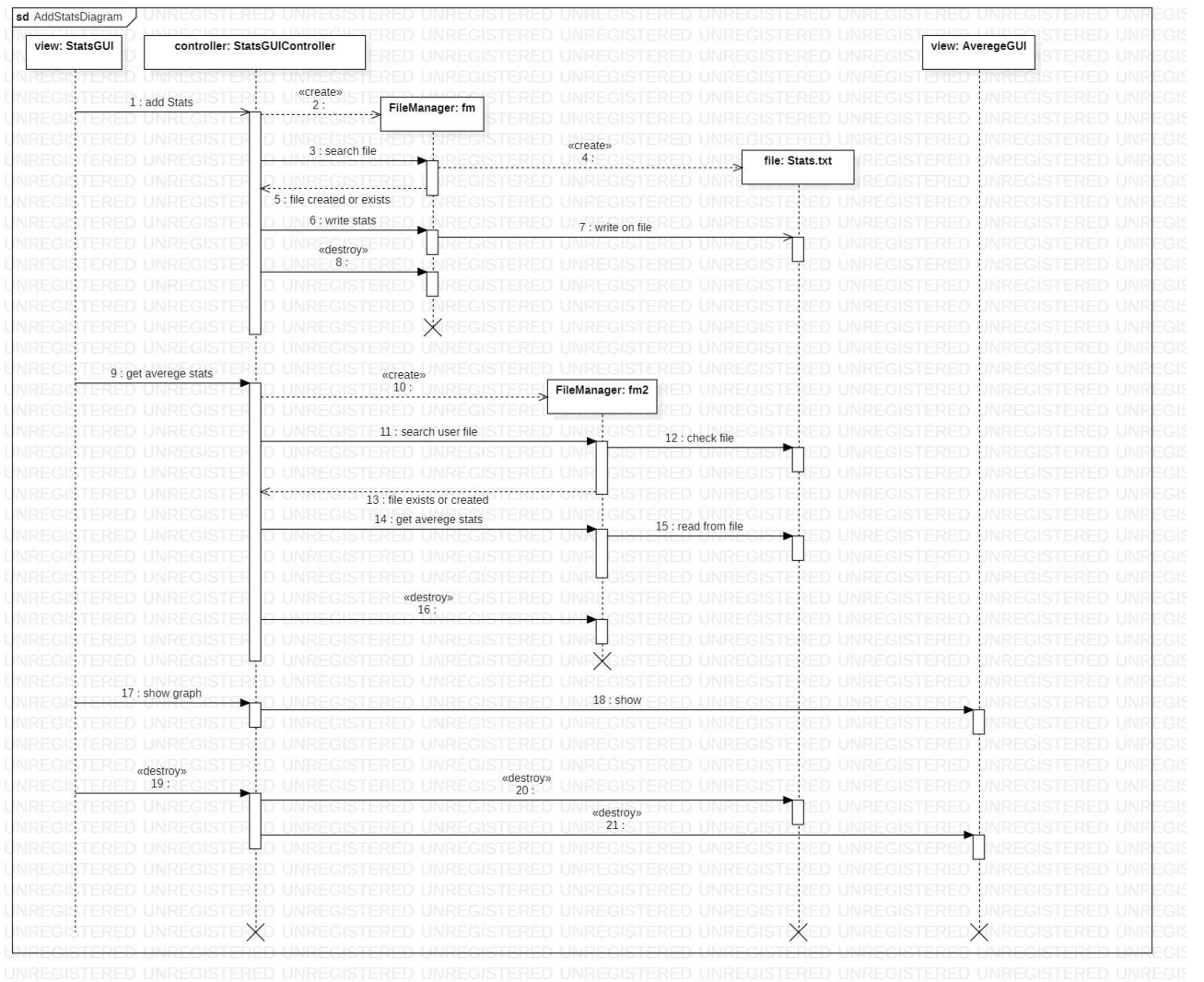
Nel Progetto ho utilizzato 2 design pattern.

- Factory: utilizzato per creare liste nei controller grafici della prima interfaccia grafica. Grazie all'applicazione di questo pattern ho potuto riutilizzare il concetto di ScrfollList più volte. Le varie liste che genera la factory differiscono per il contenuto, ma non per la logica di presentazione, rendono riusabile il codice.
- Singleton: utilizzata per gestire l'istanza di PlayerUser relativa al login. Infatti per gestire se un utente è loggato o meno utilizzo la classe Singleton per verificare se l'istanza è nulla o meno.

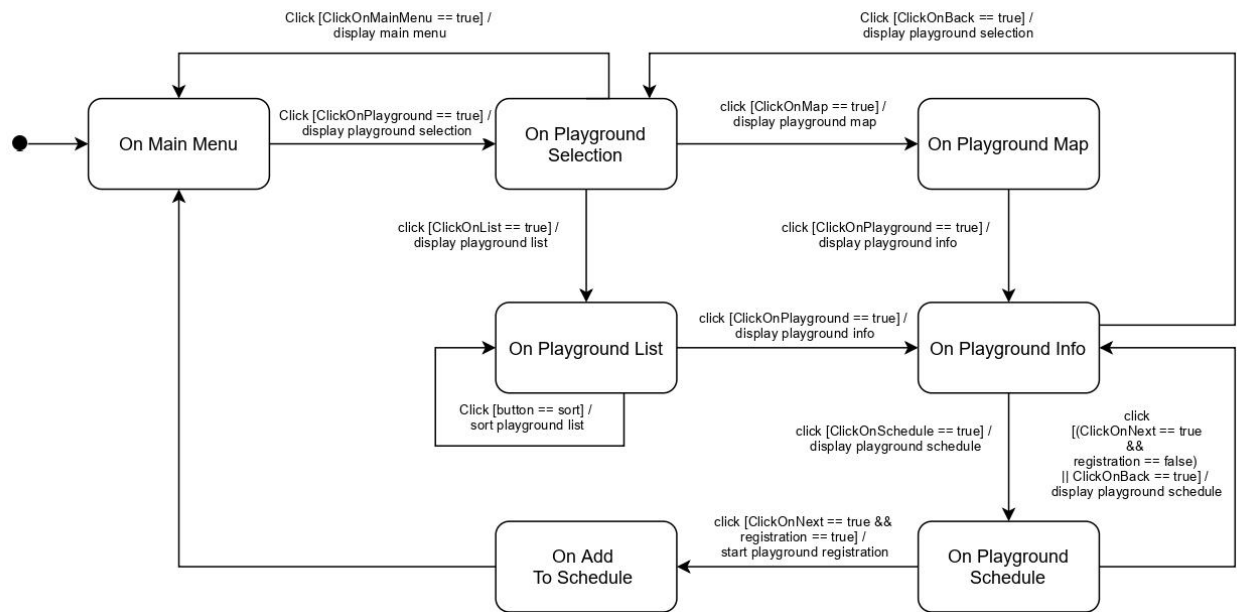
### 3.3 Activity diagram



### 3.4 Sequence diagram



### 3.5 State diagram



## 4. Testing

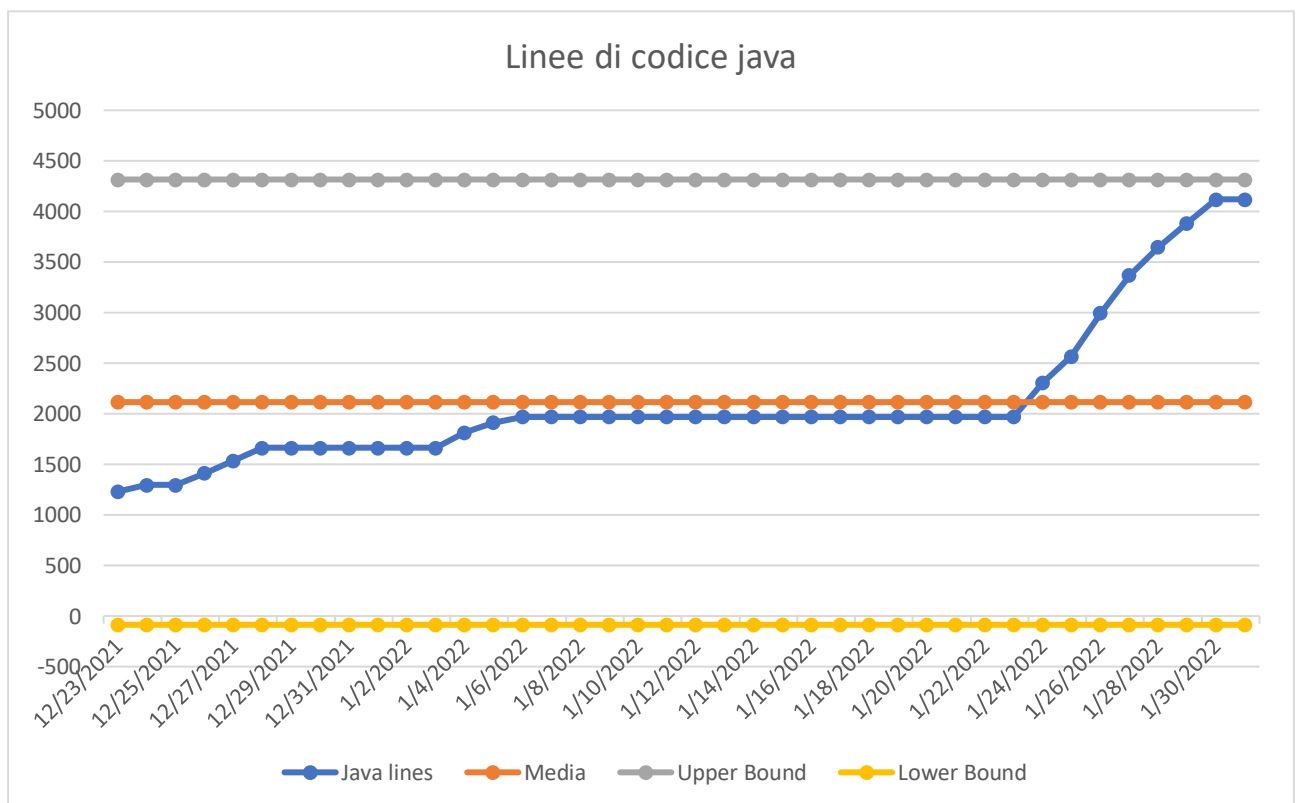
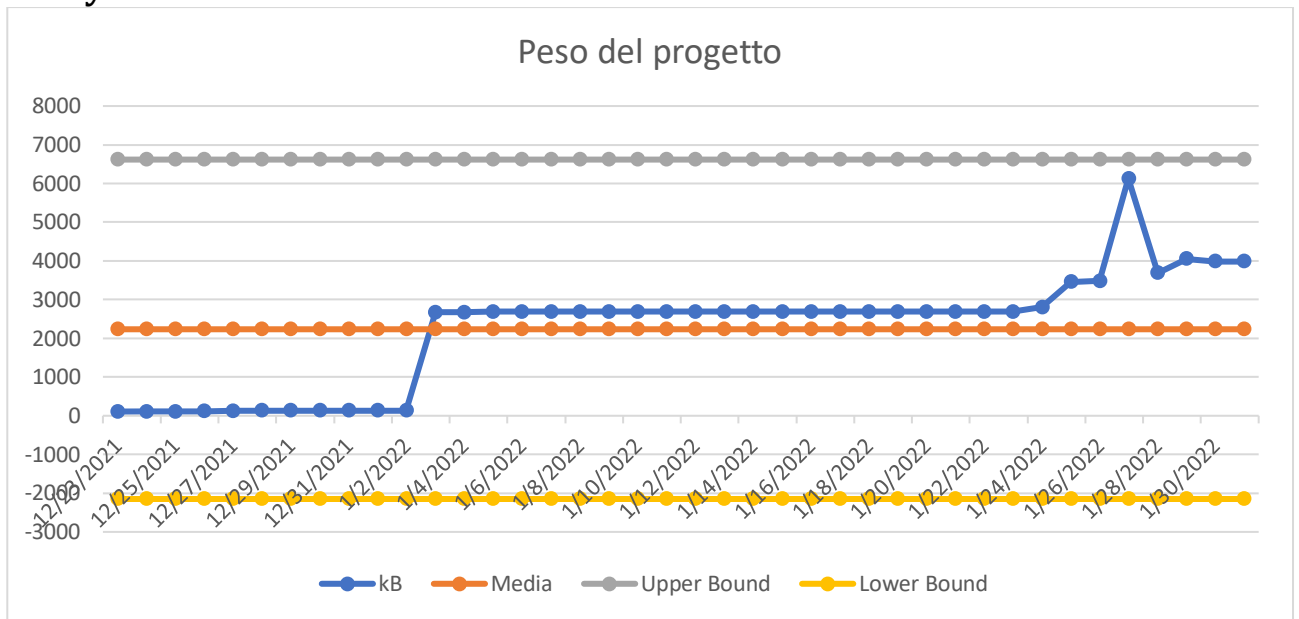
4.1 Ho sviluppato 3 test cases relative a 3 classi di tipo “controller applicativo” diverse, per verificare le funzionalità di : login/logout, ottenere la media delle statistiche, ottenere le informazioni di un determinato campetto di test. I test sono contenuti nel package `src.test.java_test`

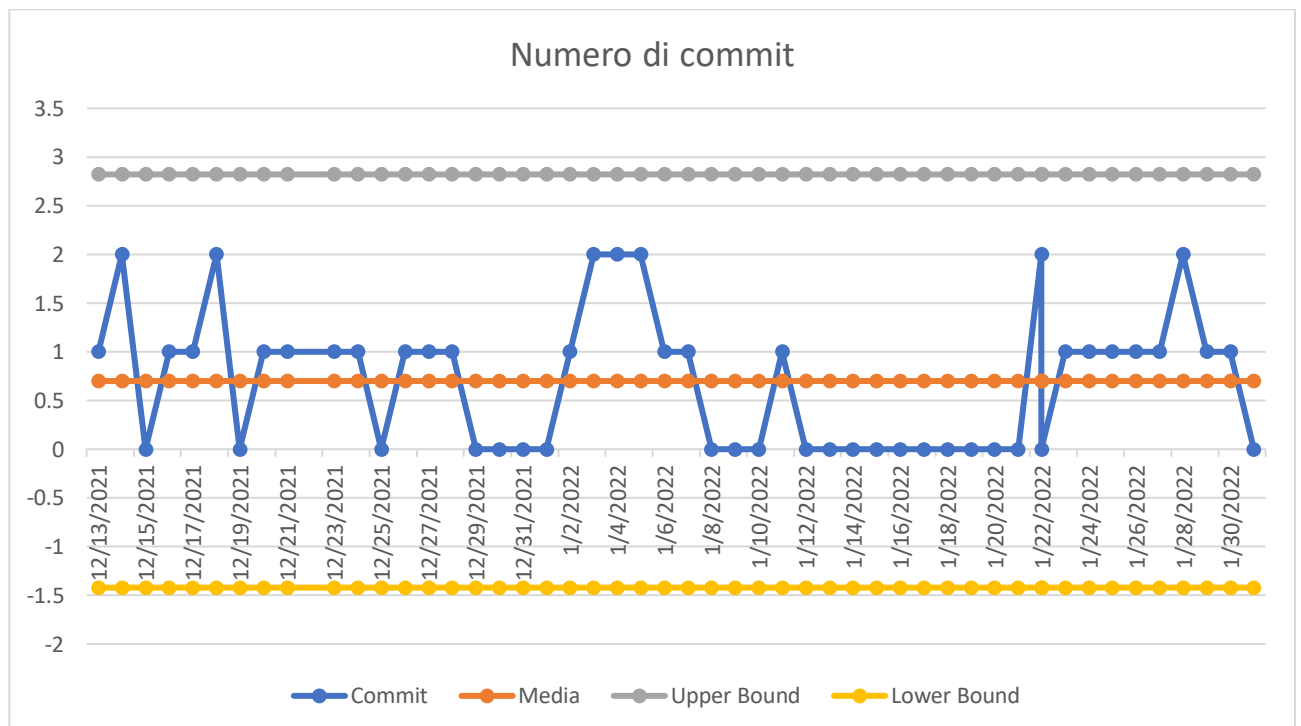
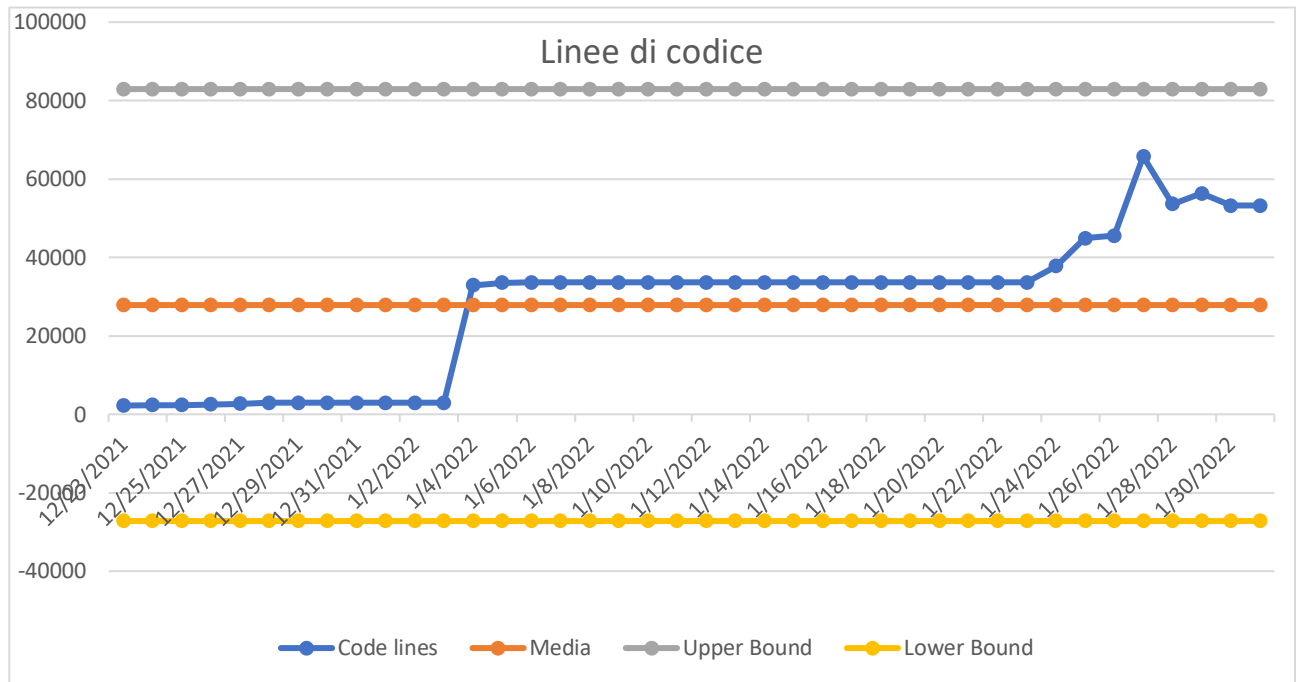
## 5. Code

5.1 Le due eccezioni che ho gestito per intero sono:

- Eccezione dovuta ad utente che si è già prenotato per un certo orario per un determinato campo : nei controller grafici si gestisce l’eccezione di tipo `AlreadyReserved` creando una finestra per avvisare che lo slot dello schedule è già stato prenotato. L’eccezione viene generata dalla classe dal metodo `search` della classe `CourtDao` che la propaga all’indietro fino ad arrivare al controller grafico.
- Eccezione dovuta per la richiesta di fare la media delle statistiche di un utente che ancora non ha alcuna statistica : l’eccezione è gestita nei controller grafici mostrando un messaggio di errore nel primo caso e aggiungendo il messaggio di errore ad una `ListView` nel secondo caso. L’eccezione è generata nella classe `FileManager` , sotto forma di `FileNotFoundException`, e viene propagata dalla classe `StatsDao` a ritroso fino alla classe `AverageController` che ne metodo `average` la converte in un’eccezione di tipo `MyException` fino a propagarla al controller grafico.

## 6. Analytics





## 7. Video

Nel file zip sono presenti due video, ognuno relative ad un'interfaccia grafica mostrando le stesse funzionalità.