Iterazione 1

Andrea Belotti, Andrea Tintori, Sergio Fabiani

Novembre 2020

Contents

List of Figures

Listings

Capitolo 1

Introduzione

1.1 init

Pdf della documentazione 1 redatta il 20 novembre 2020 del progetto Kokokò, per il corso Informatica III.

1.2 Struttura e sezioni dell'elaborato

Nel Capitolo 2 - Casi d'uso, vengono descritti gli obiettivi raggiunti, i casi d'uso, le caratteristiche funzionali e non dell'iterazione 0 che sono state completate, quelle che mancano e i nuovi casi d'uso da produrre nell'iterazione corrente.
Nel Capitolo 3 - Architettura, viene descritta l'architettura dell'applicazione che è stata raggiunta al termine dell'iterazione precedente e quello che ci si aspetta al termine dell'attuale.
Nel Capitolo 4 - Algoritmi, vengono descritti gli algoritmi utilizzati all'interno del progetto, attraverso lo pseudo codice con la loro complessità.
Nel Capitolo 5 - Testing, viene descritto il metodo di testing dinamico e statico e vengono descritti i risultati.
Nel Capitolo 6 - Planning, viene descritto cosa si farà prima della prossima iterazione.

Capitolo 2

Casi d'uso

In questo capitolo si parla dei requisiti funzionali e non, dei diagrammi UML e scenari completati e di quelli a cui si vuole arrivare.

2.1 Requisiti raggiunti

Durante la riunione di brainstorming del 20 novembre 2020 è stato fatto il punto della situazione controllando i risultati raggiunti e i casi d'uso completati. Più nello specifico sono stati completati i seguenti casi d'uso:

- Giocare con l'applicazione.
- SignUp.
- LogIn.
- Modificare le opzioni di gioco.

Rimangono quindi da sviluppare e completare:

- Scegliere la modalità di gioco (dato che per ora si può solamente giocare off-line).
- Scegliere il livello nella modalità off-line (dato che per ora i livelli sono sequenziali e non è permesso scegliere livelli già completati).

Parlando invece di requisiti funzionali e non, si è potuto creare un'applicazione che funziona su più del 98% dei dispositivi Android, la dimensione attuale

dell'applicazione è di 9,5 MB e la percentuale di crash in ore di utilizzo è inferiore al 2% (per questa percentuale è stata fatta utilizzare l'applicazione a 60 persone per circa 10 minuti l'una). I livelli creati non sono ancora 10 e non sono ancora state create delle abilità per rendere il gioco più dinamico. Si è iniziato a pensare ad un algoritmo di match-making. La velocità di transizione tra un livello è l'altro è inferiore al decimo di secondo, non sono ancora stati fatti controlli sulla RAM utilizzata.

2.2 Requisiti Funzionali

I requisiti funzionali rimanenti sono quelli riguardanti il numero di livelli e la creazione di abilità così da rendere il gioco più dinamico per la parte off-line mentre riguardo la parte on-line rimane da fare tutto ciò che era stato detto nella scorsa iterazione ossia: "*si vuole arrivare ad avere un match-making equo che vada a considerare il livello dei singoli utenti e il luogo di provenienza così da rendere l'esperienza più piacevole ed avvincente*".

2.3 Requisiti non Funzionali

I requisiti non-funzionali, ossia i requisiti e vincoli offerti dal sistema che il cliente chiede tramite caratteristiche desiderate, che si vogliono raggiungere sono:

- **Affidabilità**: Vogliamo garantire la massima affidabilità ed efficienza per la gestione dei dati dei clienti, separando in due database i dati sensibili da quelli di gioco.

- **Portabilità**: Puntiamo a creare un'applicazione utilizzabile almeno dal 98% degli utenti Android. (*Raggiunto, da mantenere*)

- **Velocità**: Velocità di transizione tra i vari screen (1/10 di secondo) e velocità match-making (si punta ad un massimo di 30 secondi, anche se dipende dal numero di giocatori connessi in quel momento).

- **Memoria**: Essendo un gioco in pixel art fatto ex novo, si punta a non utilizzare troppa memoria (sicuramente sotto i 512MB) e a far in modo che la ram utilizzata dal dispositivo sia la minore possibile.

- **Facilità d'uso** : Si punta ad avere sia l'installazione che il gioco user-friendly e il più chiaro possibili.
- **Robustezza** : Puntiamo a non avere crash dell'applicazione e se per caso dovesse occorrere, faremo in modo che il tempo di risposta dell'applicazione sia il minore possibile(inferiore ai 5 secondi).

2.4 Elenco casi d'uso e descrizione (UML)

I casi d'uso non completati e i nuovi casi di test per l'applicazione sono:

- Scegliere la modalità di gioco, giocare quindi on-line o off-line.
- Scelgiere il livello nella modalità off-line.
- Creare una scoreboard così da permettere all'utente di vedere il suo punteggio.
- Rifinire le opzioni di gioco (attivare e disattivare l'audio, scegliere il nickname).
- Possibilità di fare il log-out.

Nome Requisito	id	Tipo	Prior.	Critic.	Rischio	Data	Descrizione	Fonte	Requisiti Figli
Scegliere mod. di gioco	1	funzion.	1	alta	basso	20/10/2020	deve permettere all'utente di scegliere la modalità di gioco	Riunione 20/10/2020	2
Scelta livello	2	funzion.	1	bassa	basso	20/10/2020	deve permettere all'utente di scegliere il livello	Riunione 20/10/2020	
Scoreboard	3	funzion.	1	bassa	basso	20/10/2020	deve permettere all'utente di accedere	Riunione 20/10/2020	
Modific. opzioni di gioco	4	funzion.	1	bassa	basso	20/10/2020	migliorare le opzioni di gioco	Riunione 20/10/2020	
Possibilità di Log-Out	5	funzion.	1	alta	alto	20/10/2020	deve permettere all'utente sloggarsi	Riunione 20/10/2020	

Figure 2.1: Diagramma UML

2.5 Scenari

Gli scenari descritti nella precedente iterazione (riportati qui sotto) sono stati pienamente "soddisfatti".

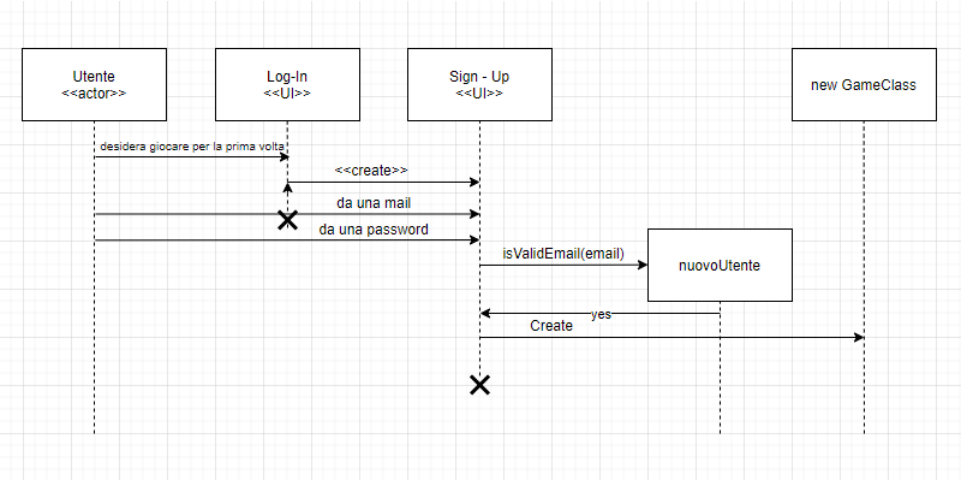


Figure 2.2: Scenario Sign-up (primo accesso)

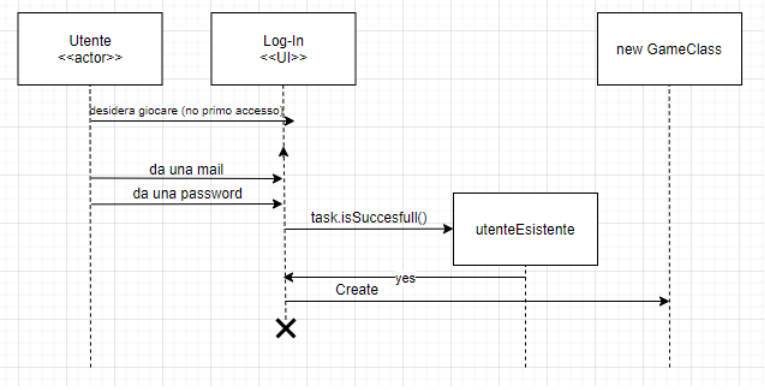


Figure 2.3: Scenario Log-in

Di seguito si portano altri scenari che si vogliono "completare" al termine di questa iterazione.

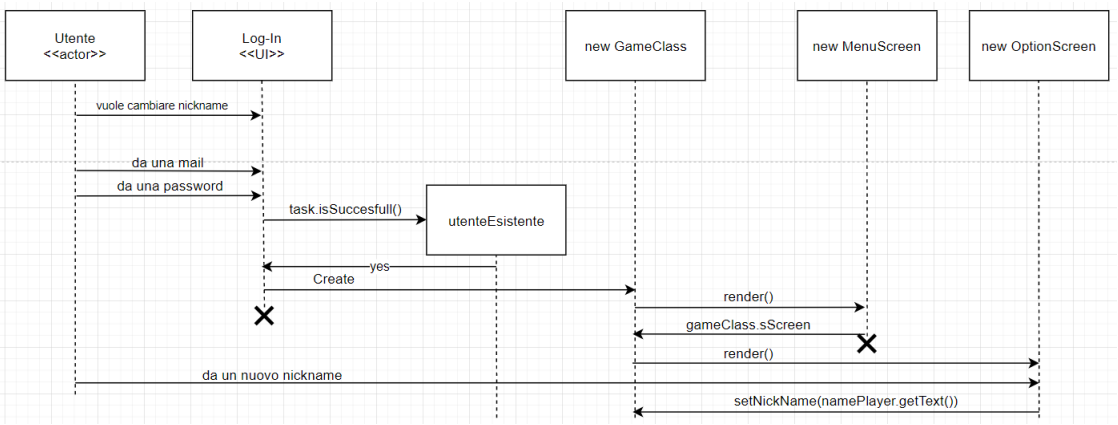


Figure 2.4: Scenario cambio del nickname

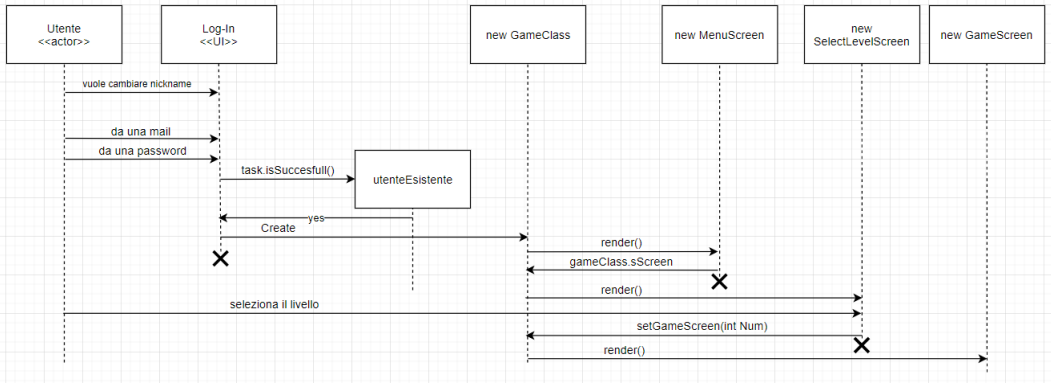


Figure 2.5: Scenario seleziona livello

Capitolo 3

Architettura

In questo paragrafo si propone l'architettura del progetto e alcune immagini che mostrano come si presenta l'applicazione programmata fino ad ora e l'architettura futura che si vuole raggiungere al termine di questa iterazione.

3.1 Design architettura attuale

Di seguito si riporta il design architetturale del progetto alla fine dell'iterazione 0 completata il 20/11/2020:

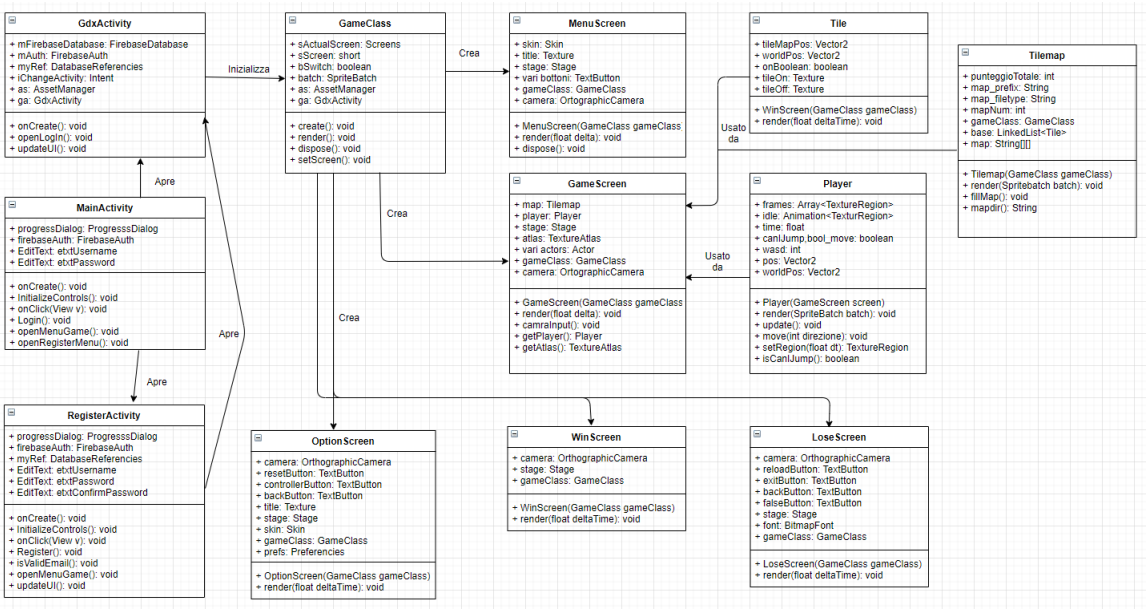
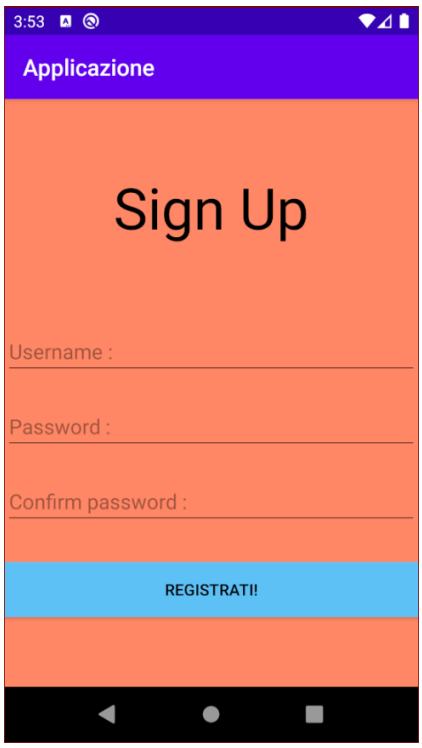
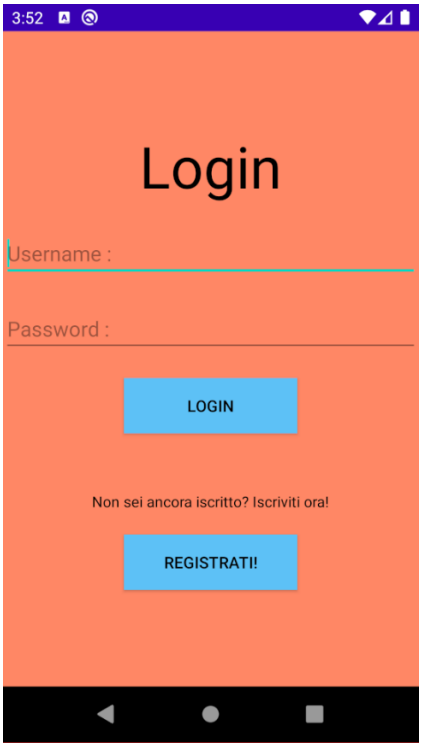


Figure 3.1: Design architettura classi UML

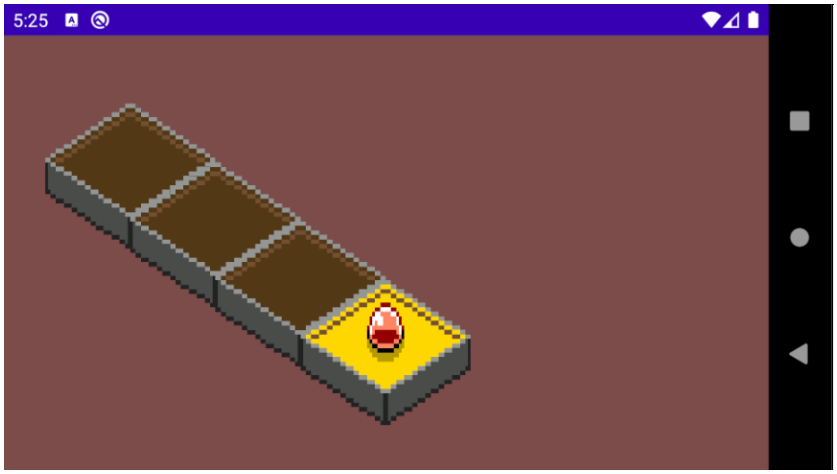
La classe principale da cui si avvia il gioco è la MainActivity, qui si crea la prima facciata, quella di log-in. Se non si è ancora registrati, allora si deve accedere alla classe RegisterActivity (tramite l'apposito pulsante). Una volta che è stato creato creato un account o che si è fatto l'accesso allora si accederà alla classe GdxActivity che permette di inizializzare la GameClass, la classe che va a renderizzare il gioco, da qui si creano i vari screen che vanno a definire il gioco (MenuScreen, GameScreen, OptionScreen, WinScreen e LoseScreen).



(a) RegisterActivity UI



(b) MainActivity UI



(c) GdxActivity UI

Figure 3.2: UI delle activity

3.2 Design architettura futura

L'architettura futura viene rappresentata attraverso un Deployment Diagram, ossia un diagramma che rappresenta in che modo sviluppato in concept a livello di codifica. I concetti fisici vengono rappresentati all'interno del ... utilizzando i nodi, ossia "parallelogrammi", mentre gli oggetti di tipo software vengono rappresentati tramite dei riquadri contenenti un piccolo riquadro in alto a destra con due rettangoli.

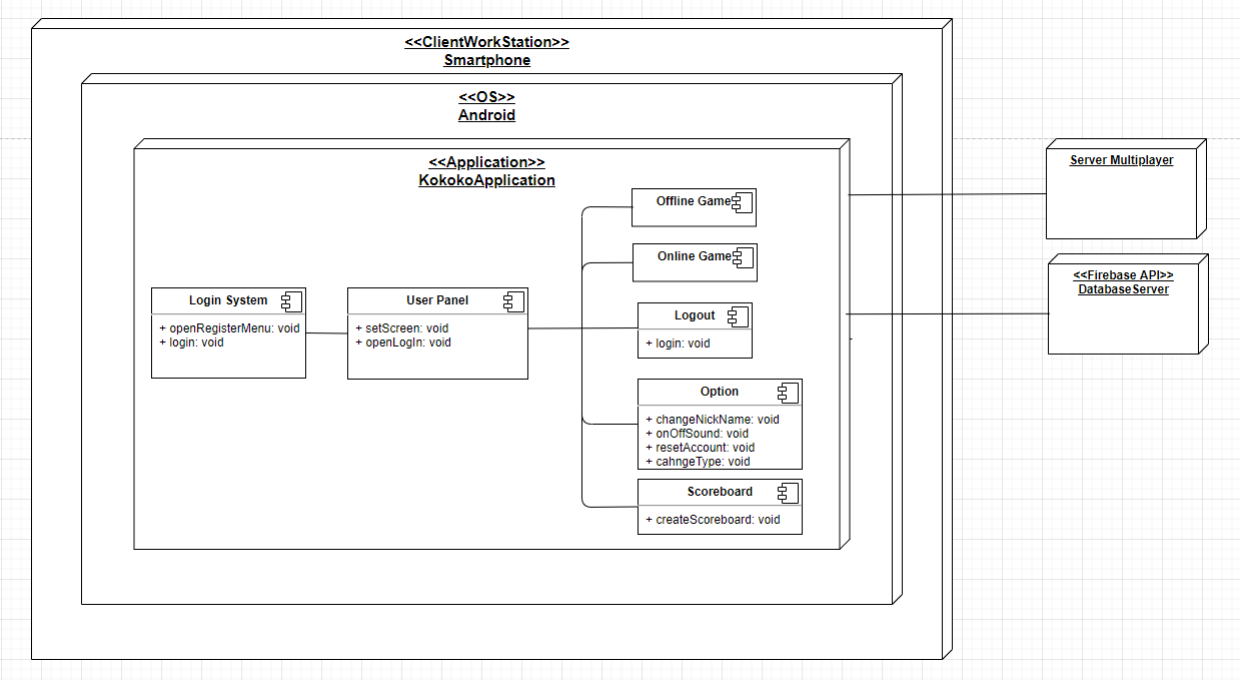


Figure 3.3: Deplyment Diagram ipotizzato del progetto al termine dell'terazione 1

Si può notare che i dispositivi che utilizzeranno l'applicazione saranno solamente Smartphone con Sistema Operativo Android e come si necessita di un database per contenere i dati degli utenti ed un server per gestire il mutiplayer.

Capitolo 4

Algoritmi

In questo capitolo si parla degli algoritmi inseriti all'interno del codice, in particolar modo si è pensato di inserire un algoritmo per la creazione della scoreboard. Dato che la classifica dovrà ordinare tutti i giocatori in base al punteggio l'idea avuta è quella di utilizzare un MergeSort modificato che ordini due vettori, il vettore dei Nomi e il vettore dei Punti, presi dal database in cui si salvano i dati dei vari utenti.

4.1 Pseudocodice

Di seguito è riportato lo pseudo-codice dell'algoritmo di MergeSort ridefinito per il progetto:

```
1  algoritmo MergeSort (A,S,p,r) => void
2      if (p < r) then
3          int q:= FLOOR ((p+r)/2);
4          MergeSort (A,S,p,q);
5          MergeSort (A,S,q+1,r);
6          Merge(A,S,p,q,r);
7      endif
8
9  algoritmo Merge (A,S,p,q,r) => void
10     int i := p
11     int j := q+1
12     array tmpA array di int
13     array tmpS array di string
14
15
```

```
16      while (i <= q && j<= r) do
17          if(A[i] >= A[j])
18              tmpA[k] = A[i];
19              tmpS[k] = S[i];
20              i := i+1;
21          else
22              tmpA[k] = A[j];
23              tmpS[k] = S[j];
24              j := j+1;
25          endif
26          k := k+1;
27      endwhile
28
29      while( i <= q ) do
30          tmpA[k] = A[i];
31          tmpS[k] = S[i];
32          i := i+1;
33          k := k+1;
34      endwhile
35
36      while( j <= r ) do
37          tmpA[k] = A[j];
38          tmpS[k] = S[j];
39          j := j+1;
40          k := k+1;
41      endwhile
42
43      for k := p to r do
44              A[k] = tmpA[k-p];
45              S[k] = tmpS[k-p];
46      endfor
```

Listing 4.1: pseudocode

4.2 Calcolo della complessità

La complessità di questo algoritmo si calcola usando il teorema di Master per l'equazione di ricorrenza T(n) che descrive lo pseudocodice riportato sopra. L'equazione di ricorrenza T(n) è la seguente:

$$T(n) = \begin{cases} c1 & \text{if } p == r \\ 2T(n/2) + \Theta(n) & \text{else} \end{cases}$$

4.2.1 Teorema di Master

Il Teorema di Master permette di analizzare algoritmi basati sulla tecnica del divide et impera:

- dividi il problema (di dimensione n) in a sotto-problemi di dimensione n/b con $a >= 1$ e $b > 1$.
- risolvi i sotto-problemi ricorsivamente.
- ricombina le soluzioni.

Sia $f(n)$ il tempo per dividere e ricombinare istanze di dimensione n. La relazione di ricorrenza è data da:

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ aT(n/b) + f(n) & \text{else} \end{cases}$$

Questa relazione ha soluzione :

- $T(n) = \Theta(n^{\log_b a})$ se $f(n) = O(n^{\log_b a - \epsilon})$ per $\epsilon > 0$.
- $T(n) = \Theta(n^{\log_b a} \log n)$ se $f(n) = \Theta(n^{\log_b a})$.
- $T(n) = \Theta(f(n))$ se $f(n) = \Omega(n^{\log_b a + \epsilon})$ per $\epsilon > 0$ e $af(n/b) \leq cf(n)$ per $c < 1$ e n sufficientemente grande.

4.2.2 Applicazione del Teorema di Master

Si applica quindi il teorema di master nel caso dello pseudo codice appena presentato

$$T(n) = \begin{cases} c1 & \text{if } p == r \\ 2T(n/2) + \Theta(n) & \text{else} \end{cases}$$

Ho che $a = b = 2$ ed $f(n) = \Theta(n)$.
Ora si calcola $g(n) = n(\log_b a) = n^1 = n$.

$$\lim_{n \to \text{inf}} \frac{f(n)}{g(n)} = \frac{n}{n} = cost.$$

Quindi ci troviamo nel secondo caso del Teorema di Master e quindi la complessità è $T(n) = \Theta(n \log n)$

4.3 Codice implementato

Il codice implementato all'interno del progetto è il seguente:

```
1  public void mergeSort(int[] punti, String[] nomi, int p, int
       r){
2          if(p < r){
3              int q = (p + r) / 2;
4              mergeSort(punti, nomi,  p,  q);
5              mergeSort(punti, nomi,  q+1,  r);
6              trueMerge(punti, nomi, p, q, r);
7          }
8       }
```

Listing 4.2: metodo mergeSort

```
1  public void trueMerge(int[] punti, String[] nomi, int p, int
       q ,int r){
2          int i = p;
3          int j = q+1;
4          int k = 0;
5          int[] tempPunti = new int[r-p+1];
6          String[] tempNomi = new String[r-p+1];
7
8          while(i <= q && j <= r){
9              if( punti[i] >= punti[j] ) {
10                 tempPunti[k] = punti[i];
```

```
                tempNomi[k] = nomi[i];
                i++;
            } else{
                tempPunti[k] = punti[j];
                tempNomi[k] = nomi[j];
                j++;
            }
            k++;
        }

        while( i <= q ){
            tempPunti[k] = punti[i];
            tempNomi[k] = nomi[i];
            i++;
            k++;
        }

        while( j <= r ){
            tempPunti[k] = punti[j];
            tempNomi[k] = nomi[j];
            j++;
            k++;
        }

        for(k = p; k <= r; k++ ){
            punti[k] = tempPunti[k-p];
            nomi[k] = tempNomi[k-p];
        }
    }
```

Listing 4.3: metodo trueMerge

Il codice riportato qui sopra è la traduzione dello pseudo-codice in Java. Questo codice semplicemente ordina i due vettori in modo decrescente rispetto al numero di punti fatti da ogni giocatore presente all'interno del database. Questa parte è stata inserita alla fine dell'iterazione 1.

Capitolo 5

Testing

In questo capitolo si descrivono i casi di test e la copertura raggiunta al termine della iterazione 1.

5.1 Analisi dinamica

L'analisi dinamica è il processo di valutazione di un sistema software o di un suo componente basato sull'osservazione del suo comportamento in esecuzione. La precondizione necessaria per poter effettuare un test è la conoscenza del comportamento atteso per poterlo confrontare con quello osservato. L'oracolo è quella componente che conosce il comportamento atteso per ogni caso di test e può essere umano, quindi basato sulla conoscenza delle specifiche e sul giudizio personale, oppure automatico, generato dalla definizione di specifiche formali.

La prima fase di un'analisi dinamica prevede l'esecuzione di tutte le procedure e dei casi di test, prendendo come input sia la configurazione del software che quella dei test. Il testing è stato svolto su tutti i metodi pubblici presenti all'interno del codice, evitando i metodi privati dato che sono richiamati all'interno di quelli pubblici, quindi testando i pubblici, sono stati testati anche loro.

Per fare il testing dinamico è stato utilizzato JUnit4, in particolar modo sono stati utilizzati i Mock e le Asserzioni per i metodi delle classi che non facevano parte delle Activity; per le classi facenti parte delle Activity, quindi aventi una GUI, il testing è stato un po' più complicato ed è stato utilizzato un framework chiamato Espresso che ha semplificato il lavoro di testing.

Di seguito sono riportate una classe di testing fatto con Espresso ed una con i Mock:

```
1  package com.example.kokoko.android.activities;
2
3  import android.view.View;
4  import android.view.ViewGroup;
5  import android.view.ViewParent;
6  import com.example.kokoko.R;
7  import org.hamcrest.Description;
8  import org.hamcrest.Matcher;
9  import org.hamcrest.TypeSafeMatcher;
10 import org.junit.Rule;
11 import org.junit.Test;
12 import org.junit.runner.RunWith;
13 import androidx.test.espresso.ViewInteraction;
14 import androidx.test.filters.LargeTest;
15 import androidx.test.rule.ActivityTestRule;
16 import androidx.test.runner.AndroidJUnit4;
17 import static androidx.test.espresso.Espresso.onView;
18 import static androidx.test.espresso.action.ViewActions.click
       ;
19 import static androidx.test.espresso.action.ViewActions.
       closeSoftKeyboard;
20 import static androidx.test.espresso.action.ViewActions.
       pressImeActionButton;
21 import static androidx.test.espresso.action.ViewActions.
       replaceText;
22 import static androidx.test.espresso.assertion.ViewAssertions
       .matches;
23 import static androidx.test.espresso.matcher.RootMatchers.
       withDecorView;
24 import static androidx.test.espresso.matcher.ViewMatchers.
       isDisplayed;
25 import static androidx.test.espresso.matcher.ViewMatchers.
       withId;
26 import static androidx.test.espresso.matcher.ViewMatchers.
       withParent;
27 import static androidx.test.espresso.matcher.ViewMatchers.
       withText;
28 import static org.hamcrest.Matchers.allOf;
29 import static org.hamcrest.Matchers.is;
30 import static org.hamcrest.Matchers.not;
31
32
33 @LargeTest
```

```
34 @RunWith(AndroidJUnit4.class)
35 public class RegisterActivityTestDiffPass {
36
37     @Rule
38     public ActivityTestRule<MainActivity> mActivityTestRule =
       new ActivityTestRule<>(MainActivity.class);
39
40     @Test
41     public void mainActivityTestDiffPass() {
42         ViewInteraction button = onView(
43                 allOf(withId(R.id.btnRegister), withText("
       Registrati!"),
44                         childAtPosition(
45                                 childAtPosition(
46                                         withId(android.R.id.
       content),
47                                         0),
48                                 5),
49                         isDisplayed()));
50         button.perform(click());
51
52         ViewInteraction appCompatEditText = onView(
53                 allOf(withId(R.id.etxtUsername),
54                         childAtPosition(
55                                 childAtPosition(
56                                         withId(android.R.id.
       content),
57                                         0),
58                                 1),
59                         isDisplayed()));
60         appCompatEditText.perform(replaceText("giacomo@gmail.
       it"), closeSoftKeyboard());
61
62         ViewInteraction appCompatEditText2 = onView(
63                 allOf(withId(R.id.etxtUsername), withText("
       giacomo@gmail.it"),
64                         childAtPosition(
65                                 childAtPosition(
66                                         withId(android.R.id.
       content),
67                                         0),
68                                 1),
69                         isDisplayed()));
70         appCompatEditText2.perform(pressImeActionButton());
71
```

```
72          ViewInteraction appCompatEditText3 = onView(
73                  allOf(withId(R.id.etxtPassword),
74                          childAtPosition(
75                                  childAtPosition(
76                                          withId(android.R.id.
    content),
77                                          0),
78                                  2),
79                          isDisplayed()));
80          appCompatEditText3.perform(replaceText("giacomoo"),
    closeSoftKeyboard());
81
82          ViewInteraction appCompatEditText4 = onView(
83                  allOf(withId(R.id.etxtPassword), withText("
    giacomoo"),
84                          childAtPosition(
85                                  childAtPosition(
86                                          withId(android.R.id.
    content),
87                                          0),
88                                  2),
89                          isDisplayed()));
90          appCompatEditText4.perform(pressImeActionButton());
91
92          ViewInteraction appCompatEditText5 = onView(
93                  allOf(withId(R.id.etxtConfirmPassword),
94                          childAtPosition(
95                                  childAtPosition(
96                                          withId(android.R.id.
    content),
97                                          0),
98                                  3),
99                          isDisplayed()));
100         appCompatEditText5.perform(replaceText("giacomo"),
    closeSoftKeyboard());
101
102         ViewInteraction appCompatEditText6 = onView(
103                 allOf(withId(R.id.etxtConfirmPassword),
    withText("giacomo"),
104                         childAtPosition(
105                                 childAtPosition(
106                                         withId(android.R.id.
    content),
107                                         0),
108                                 3),
```

```
109                         isDisplayed()));
110         appCompatEditText6.perform(pressImeActionButton());
111 
112         ViewInteraction appCompatButton = onView(
113                 allOf(withId(R.id.btnRegister), withText("
    Registrati!"),
114                         childAtPosition(
115                                 childAtPosition(
116                                         withId(android.R.id.
    content),
117                                         0),
118                                 4),
119                         isDisplayed()));
120         appCompatButton.perform(click());
121 
122         onView(withText("Your passwords don't match"))
123                 .inRoot(withDecorView(not(mActivityTestRule.
    getActivity().getWindow().getDecorView())))
124                 .check(matches(isDisplayed()));
125 
126         ViewInteraction editText = onView(
127                 allOf(withId(R.id.etxtUsername), withText("
    giacomo@gmail.it"),
128                         withParent(withParent(withId(android.
    R.id.content))),
129                         isDisplayed()));
130         editText.check(matches(withText("giacomo@gmail.it")))
    ;
131 
132         ViewInteraction editText2 = onView(
133                 allOf(withId(R.id.etxtPassword), withText("
    giacomoo"),
134                         withParent(withParent(withId(android.
    R.id.content))),
135                         isDisplayed()));
136         editText2.check(matches(withText("giacomoo")));
137 
138         ViewInteraction editText3 = onView(
139                 allOf(withId(R.id.etxtConfirmPassword),
    withText("giacomo"),
140                         withParent(withParent(withId(android.
    R.id.content))),
141                         isDisplayed()));
142         editText3.check(matches(withText("giacomo")));
143 
```

```
144         ViewInteraction button2 = onView(
145                 allOf(withId(R.id.btnRegister), withText("
    REGISTRATI!"),
146                         withParent(withParent(withId(android.
    R.id.content))),
147                         isDisplayed()));
148         button2.check(matches(isDisplayed()));
149     }
150
151     private static Matcher<View> childAtPosition(
152             final Matcher<View> parentMatcher, final int
    position) {
153
154         return new TypeSafeMatcher<View>() {
155             @Override
156             public void describeTo(Description description) {
157                 description.appendText("Child at position " +
     position + " in parent ");
158                 parentMatcher.describeTo(description);
159             }
160
161             @Override
162             public boolean matchesSafely(View view) {
163                 ViewParent parent = view.getParent();
164                 return parent instanceof ViewGroup &&
    parentMatcher.matches(parent)
165                         && view.equals(((ViewGroup) parent).
    getChildAt(position));
166             }
167         };
168     }
169 }
```

Listing 5.1: testing Espresso register activity senza Password

```
1  package com.example.kokoko.libgdx;
2
3  import com.example.kokoko.Constant;
4  import com.example.kokoko.android.activities.MainActivity;
5
6  import org.junit.jupiter.api.Test;
7  import org.mockito.Mock;
8  import org.mockito.Mockito;
9
10 import static org.junit.jupiter.api.Assertions.*;
11
12 class GameClassTest {
13
14     private static final Constant.NumeroScreen GS = Constant.
    NumeroScreen.GAMESCREEN;
15     private static final Constant.NumeroScreen LS = Constant.
    NumeroScreen.LOSESCREEN;
16     private static final Constant.NumeroScreen MS = Constant.
    NumeroScreen.MENUSCREEN;
17     private static final Constant.NumeroScreen OS = Constant.
    NumeroScreen.OPTIONSCREEN;
18     private static final Constant.NumeroScreen WS = Constant.
    NumeroScreen.WINSCREEN;
19     @Mock
20     private GameClass gc = Mockito.spy(GameClass.class);
21
22     @Test
23     void testCreate(){
24         gc.create();
25         Mockito.verify(gc).create();
26     }
27
28     @Test
29     void testConstructor(){
30         assertNotNull(gc);
31     }
32
33     @Test
34     void setMenuScreen() {
35         Mockito.when(gc.setScreen()).thenReturn(Constant.
    NumeroScreen.MENUSCREEN);
36         Constant.NumeroScreen ris = gc.setScreen();
37         assertEquals(ris, MS);
38     }
39
```

```
40      @Test
41      void setGameScreen() {
42          Mockito.when(gc.setScreen()).thenReturn(Constant.
    NumeroScreen.GAMESCREEN);
43          Constant.NumeroScreen ris = gc.setScreen();
44          assertEquals(ris, GS);
45      }
46
47      @Test
48      void setLoseScreen() {
49          Mockito.when(gc.setScreen()).thenReturn(Constant.
    NumeroScreen.LOSESCREEN);
50          Constant.NumeroScreen ris = gc.setScreen();
51          assertEquals(ris, LS);
52      }
53
54      @Test
55      void setWinScreen() {
56          Mockito.when(gc.setScreen()).thenReturn(Constant.
    NumeroScreen.WINSCREEN);
57          Constant.NumeroScreen ris = gc.setScreen();
58          assertEquals(ris, WS);
59      }
60
61      @Test
62      void setOptionScreen() {
63          Mockito.when(gc.setScreen()).thenReturn(Constant.
    NumeroScreen.OPTIONSCREEN);
64          Constant.NumeroScreen ris = gc.setScreen();
65          assertEquals(ris, OS);
66      }
67
68  }
```

Listing 5.2: testing mock GameClass

La copertura raggiunta tramite il testing è del 70%. La percentuale è così bassa perché i metodi privati non sono stati testati dato che sono nascosti all'utente e quindi solitamente è buona norma ignorarli poiché vengono controllati quando si attuano i test dei metodi pubblici. La copertura data da test di Espresso invece ha raggiunto l'85% della copertura totale.

5.2 Analisi statica

Questo tipo di test del software viene eseguito prima di mettere in azione il software. I test statici vengono eseguiti per cercare gli errori negli algoritmi, codici o documenti. Gli errori commessi durante la scrittura del software vengono controllati per la correzione utilizzando test statici.

Per questo codice è stato utilizzato Checkstyle - uno strumento di sviluppo per utile ai programmatori a scrivere codice Java che aderisce a uno standard di codifica. Automatizza il processo di controllo del codice Java per risparmiare agli umani questo compito noioso (ma importante). Ciò lo rende ideale per i progetti che desiderano applicare uno standard di codifica. Checkstyle può controllare molti aspetti del tuo codice sorgente. Può trovare problemi di progettazione di classi, problemi di progettazione di metodi. Ha anche la capacità di controllare il layout del codice e problemi di formattazione. I controlli vengono scritti in .xml e questo lo rende estremamente personalizzabile. Per poterlo utilizzare è stato creato un file checkstyle.xml inserito all'interno del progetto, in questo file sono stati inseriti diversi <module> che vanno a definire regole da seguire per avere uno standard comune nella programmazione del codice in Java.

Di seguito si riporta il file checkstyle.xml:

```
1  <?xml version="1.0"?>
2  <!DOCTYPE module PUBLIC
3            "-//Checkstyle//DTD Checkstyle Configuration 1.3//
       EN"
4            "https://checkstyle.org/dtds/configuration_1_3.dtd
       ">
5
6  <module name="Checker">
7    <module name="JavadocPackage"/>
8    <module name="TreeWalker">
9      <module name="ConstantName"/>
10     <module name="EmptyBlock"/>
```

```
11
12  <!-- Block Checks -->
13    <module name="AvoidNestedBlocks">
14      <property name="allowInSwitchCase" value="true"/>
15    </module>
16    <module name="EmptyCatchBlock"/>
17    <module name="LeftCurly"/>
18    <module name="NeedBraces"/>
19
20  <!-- Class Design -->
21
22    <module name="FinalClass"/>
23    <module name="InnerTypeLast"/>
24    <module name="InterfaceIsType"/>
25    <module name="MutableException"/>
26    <module name="OneTopLevelClass"/>
27    <module name="ThrowsCount">
28      <property name="max" value="2"/>
29    </module>
30
31
32  <!-- Coding -->
33    <module name="ArrayTrailingComma"/>
34    <module name="AvoidInlineConditionals"/>
35    <module name="CovariantEquals"/>
36    <module name="DeclarationOrder"/>
37    <module name="DefaultComesLast"/>
38    <module name="EmptyStatement"/>
39    <module name="EqualsAvoidNull"/>
40    <module name="EqualsHashCode"/>
41    <module name="ExplicitInitialization"/>
42    <module name="FallThrough"/>
43    <module name="FinalLocalVariable"/>
44    <module name="HiddenField">
45      <property name="ignoreConstructorParameter" value="true
     "/>
46      <property name="ignoreSetter" value="true"/>
47      <property name="setterCanReturnItsClass" value="true"/>
48    <property name="tokens" value="VARIABLE_DEF"/>
49    </module>
50    <module name="IllegalCatch">
51      <property name="illegalClassNames"
52                value="java.lang.Exception,
53                       java.lang.Throwable,
54                       java.lang.RuntimeException,
```

```
55                      java.lang.NullPointerException"/>
56     </module>
57     <module name="IllegalInstantiation">
58       <property name="classes"
59                 value="org.xml.sax.SAXException, org.xml.sax.
   SAXParseException,
60                      org.apache.commons.beanutils.
   ConversionException,
61                      org.antlr.v4.runtime.misc.
   ParseCancellationException,
62                      antlr.RecognitionException, antlr.
   TokenStreamException,
63                      antlr.TokenStreamRecognitionException,
    antlr.ANTLRException,
64                      java.lang.StringBuffer"/>
65     </module>
66     <module name="IllegalThrows"/>
67     <module name="IllegalToken">
68       <property name="tokens" value="LABELED_STAT"/>
69       <property name="tokens" value="LITERAL_NATIVE"/>
70       <property name="tokens" value="LITERAL_VOLATILE"/>
71       <property name="tokens" value="LITERAL_ASSERT"/>
72     </module>
73     <module name="IllegalTokenText">
74       <property name="tokens" value="STRING_LITERAL"/>
75       <property name="format" value="^(US-ASCII|ISO-8859-1|
   UTF-8|UTF-16BE|UTF-16LE|UTF-16)$"/>
76       <property name="ignoreCase" value="true"/>
77     </module>
78     <module name="IllegalType">
79       <property name="illegalClassNames"
80                 value="java.util.HashSet, HashSet, java.util.
   LinkedHashMap, LinkedHashMap,
81                      java.util.TreeMap, TreeMap, java.util.
   HashMap, HashMap,
82                      java.util.LinkedHashSet, LinkedHashSet
   , java.util.TreeSet, TreeSet,
83                      java.lang.StringBuffer, StringBuffer
   "/>
84     </module>
85     <module name="InnerAssignment"/>
86     <!-- <module name="MagicNumber"/> -->
87     <module name="MissingCtor">
88       <!--
89         we will not use that fanatic validation, extra code
```

```
    is not good
90        But this Check will exists as it was created by
    community demand.
91        -->
92        <property name="severity" value="ignore"/>
93      </module>
94    <module name="MissingSwitchDefault"/>
95      <module name="ModifiedControlVariable"/>
96      <module name="MultipleStringLiterals"/>
97      <module name="MultipleVariableDeclarations"/>
98      <module name="NestedForDepth">
99        <property name="max" value="2"/>
100     </module>
101     <module name="NestedIfDepth">
102       <property name="max" value="3"/>
103     </module>
104     <module name="NestedTryDepth"/>
105     <module name="NoArrayTrailingComma">
106       <!-- This Check is conflicting with ArrayTrailingComma
    -->
107       <property name="severity" value="ignore"/>
108     </module>
109     <module name="NoClone"/>
110     <module name="NoEnumTrailingComma">
111       <!-- This Check is conflicting with our vision of code
112            to be same as ArrayTrailingComma requires it -->
113       <property name="severity" value="ignore"/>
114     </module>
115     <module name="NoFinalizer"/>
116     <module name="OneStatementPerLine"/>
117     <module name="OverloadMethodsDeclarationOrder"/>
118     <module name="PackageDeclaration"/>
119     <module name="RequireThis"/>
120     <module name="ReturnCount">
121       <property name="max" value="1"/>
122       <property name="maxForVoid" value="0"/>
123     </module>
124     <module name="SimplifyBooleanExpression"/>
125     <module name="SimplifyBooleanReturn"/>
126     <module name="StringLiteralEquality"/>
127     <module name="SuperClone"/>
128     <module name="SuperFinalize"/>
129     <module name="UnnecessaryParentheses"/>
130     <module name="
    UnnecessarySemicolonAfterTypeMemberDeclaration"/>
```

```
131     <module name="UnnecessarySemicolonInEnumeration"/>
132     <module name="UnnecessarySemicolonInTryWithResources"/>
133     <module name="VariableDeclarationUsageDistance"/>
134
135   <!-- Imports -->
136     <module name="AvoidStarImport"/>
137     <module name="AvoidStaticImport"/>
138
139   <module name="IllegalImport"/>
140     <module name="RedundantImport"/>
141     <module name="UnusedImports"/>
142
143   <!-- Javadoc Comments -->
144     <module name="AtclauseOrder"/>
145     <module name="InvalidJavadocPosition"/>
146     <module name="JavadocBlockTagLocation">
147       <!-- default tags -->
148       <property name="tags" value="author, deprecated,
      exception, hidden, param, provides"/>
149       <property name="tags" value="return, see, serial,
      serialData, serialField, since, throws"/>
150       <property name="tags" value="uses, version"/>
151       <!-- additional tags used in the project -->
152       <property name="tags" value="noinspection"/>
153     </module>
154     <module name="JavadocContentLocation"/>
155     <module name="JavadocMethod">
156       <property name="validateThrows" value="true"/>
157     </module>
158   <module name="JavadocTagContinuationIndentation"/>
159     <module name="JavadocType">
160       <!-- avoid errors on tag '@noinspection' -->
161       <property name="allowUnknownTags" value="true"/>
162     </module>
163   <module name="MissingJavadocType">
164       <property name="scope" value="private"/>
165     </module>
166   <module name="NonEmptyAtclauseDescription"/>
167     <module name="SingleLineJavadoc"/>
168   <module name="WriteTag"/>
169
170
171    <!-- Metrics -->
172     <module name="BooleanExpressionComplexity">
173       <property name="max" value="7"/>
```

```
174     </module>
175     <module name="ClassDataAbstractionCoupling">
176       <!-- Default classes are also listed -->
177     <property name="max" value="9"/>
178       <property name="excludedClasses"
179                 value="boolean, byte, char, double, float,
    int, long, short, void,
180                        Boolean, Byte, Character, Double,
    Float, Integer, Long, Short, Void,
181                        Object, Class, String, StringBuffer,
    StringBuilder,
182                        ArrayIndexOutOfBoundsException,
    Exception, RuntimeException,
183                        IllegalArgumentException,
    IllegalStateException,
184                        IndexOutOfBoundsException,
    NullPointerException, Throwable,
185                        SecurityException,
    UnsupportedOperationException, List, ArrayList,
186                        Deque, Queue, LinkedList, Set, HashSet
    , SortedSet, TreeSet, Map,
187                        HashMap, SortedMap, TreeMap,
    DetailsAST, CheckstyleException,
188                        UnsupportedEncodingException,
    BuildException, ConversionException,
189                        FileNotFoundException, TestException
    "/>
190     </module>
191     <module name="ClassFanOutComplexity">
192       <property name="max" value="25"/>
193       <!-- Default classes are also listed -->
194       <property name="excludedClasses"
195                 value="boolean, byte, char, double, float,
    int, long, short,
196                        void, Boolean, Byte, Character, Double
    , Float, Integer,
197                        Long, Short, Void, Object, Class,
    String, StringBuffer,
198                        StringBuilder,
    ArrayIndexOutOfBoundsException, Exception,
199                        RuntimeException,
    IllegalArgumentException, IllegalStateException,
200                        IndexOutOfBoundsException,
    NullPointerException, Throwable,
201                        SecurityException,
```

```
    UnsupportedOperationException, List, ArrayList,
202                       Deque, Queue, LinkedList, Set, HashSet
    , SortedSet, TreeSet, Map,
203                       HashMap, SortedMap, TreeMap,
    DetailsAST, CheckstyleException,
204                       UnsupportedEncodingException,
    BuildException, ConversionException,
205                       FileNotFoundException, TestException,
    Log, Sets, Multimap,
206                       TokenStreamRecognitionException,
    RecognitionException,
207                       TokenStreamException, IOException,
    Override, Deprecated, SafeVarargs,
208                       SuppressWarnings, FunctionalInterface
    "/>
209     </module>
210   <module name="CyclomaticComplexity">
211       <property name="switchBlockAsSingleDecisionPoint" value
    ="true"/>
212     </module>
213     <module name="JavaNCSS"/>
214     <module name="NPathComplexity"/>
215
216
217
218   <!-- Misc -->
219     <module name="ArrayTypeStyle"/>
220     <module name="AvoidEscapedUnicodeCharacters">
221       <property name="allowIfAllCharactersEscaped" value="
    true"/>
222     </module>
223     <module name="CommentsIndentation"/>
224     <module name="DescendantToken"/>
225     <module name="FinalParameters">
226       <!--
227         we will not use that fanatic validation, extra
    modifiers pollute a code
228         it is better to use extra validation(Check) that
    argument is reassigned
229         But this Check will exists as it was created by
    community demand.
230       -->
231       <property name="severity" value="ignore"/>
232     </module>
233     <module name="Indentation">
```

```
234       <property name="basicOffset" value="4"/>
235       <property name="braceAdjustment" value="0"/>
236       <property name="caseIndent" value="4"/>
237       <property name="throwsIndent" value="8"/>
238     </module>
239
240
241   <!-- Modifiers -->
242     <module name="ClassMemberImpliedModifier">
243       <!-- effectively the opposite of RedundantModifier, so
    output must be ignored -->
244       <property name="severity" value="ignore"/>
245     </module>
246     <module name="InterfaceMemberImpliedModifier">
247       <!-- effectively the opposite of RedundantModifier, so
    output must be ignored -->
248       <property name="severity" value="ignore"/>
249     </module>
250     <module name="ModifierOrder"/>
251     <module name="RedundantModifier"/>
252
253
254 <!-- Naming Conventions -->
255     <module name="AbbreviationAsWordInName">
256       <property name="ignoreFinal" value="false"/>
257       <property name="allowedAbbreviationLength" value="0"/>
258       <property name="allowedAbbreviations" value="AST"/>
259     </module>
260     <module name="AbstractClassName"/>
261     <module name="ConstantName"/>
262     <module name="InterfaceTypeParameterName"/>
263     <module name="LocalFinalVariableName"/>
264  <module name="LocalVariableName">
265       <property name="format" value="^(id)|([a-z][a-z0-9][a-
    zA-Z0-9]+)$"/>
266       <property name="allowOneCharVarInForLoop" value="true
    "/>
267     </module>
268     <module name="MemberName">
269       <property name="format" value="^(id)|([a-z][a-z0-9][a-
    zA-Z0-9]+)$"/>
270     </module>
271     <module name="MethodName"/>
272     <module name="MethodTypeParameterName"/>
273     <module name="PackageName"/>
```

```
274     <module name="ParameterName">
275       <property name="format" value="^(id)|([a-z][a-z0-9][a-
    zA-Z0-9]+)$"/>
276       <property name="ignoreOverridden" value="true"/>
277     </module>
278     <module name="LambdaParameterName">
279       <property name="format" value="^(id)|([a-z][a-z0-9][a-
    zA-Z0-9]+)$"/>
280     </module>
281     <module name="CatchParameterName">
282       <property name="format" value="^(ex|[a-z][a-z][a-zA-Z
    ]+)$"/>
283     </module>
284     <module name="StaticVariableName">
285       <property name="format" value="^(id)|([a-z][a-z0-9][a-
    zA-Z0-9]+)$"/>
286     </module>
287     <module name="TypeName"/>
288
289   <!-- Regexp -->
290     <module name="Regexp"/>
291     <module name="RegexpSinglelineJava"/>
292     <module name="RegexpSinglelineJava">
293       <property name="format" value="[^\p{ASCII}]"/>
294       <property name="ignoreComments" value="true"/>
295     </module>
296
297   <!-- Size Violations -->
298     <module name="AnonInnerLength"/>
299     <module name="ExecutableStatementCount">
300       <property name="max" value="30"/>
301     </module>
302     <module name="MethodCount">
303       <property name="maxTotal" value="34"/>
304     </module>
305     <module name="MethodLength"/>
306     <module name="OuterTypeNumber"/>
307     <module name="ParameterNumber"/>
308
309   <!-- Whitespace -->
310     <module name="EmptyForInitializerPad"/>
311     <module name="EmptyForIteratorPad"/>
312     <module name="EmptyLineSeparator">
313       <property name="allowNoEmptyLineBetweenFields" value="
    true"/>
```

```
      <property name="
    allowMultipleEmptyLinesInsideClassMembers" value="false"/>
    </module>
    <module name="GenericWhitespace"/>
    <module name="MethodParamPad"/>
    <module name="NoLineWrap"/>
    <module name="NoWhitespaceAfter">
      <property name="tokens" value="ARRAY_INIT"/>
      <property name="tokens" value="AT"/>
      <property name="tokens" value="BNOT"/>
      <property name="tokens" value="DEC"/>
      <property name="tokens" value="DOT"/>
      <property name="tokens" value="INC"/>
      <property name="tokens" value="LNOT"/>
      <property name="tokens" value="UNARY_MINUS"/>
      <property name="tokens" value="UNARY_PLUS"/>
      <property name="tokens" value="ARRAY_DECLARATOR"/>
      <property name="tokens" value="INDEX_OP"/>
      <property name="tokens" value="METHOD_REF"/>
    </module>
    <module name="NoWhitespaceBefore"/>
    <module name="NoWhitespaceBefore">
      <property name="tokens" value="DOT"/>
      <property name="tokens" value="METHOD_REF"/>
      <property name="allowLineBreaks" value="true"/>
    </module>
    <module name="OperatorWrap">
      <property name="tokens" value="QUESTION"/>
      <property name="tokens" value="COLON"/>
      <property name="tokens" value="EQUAL"/>
      <property name="tokens" value="NOT_EQUAL"/>
      <property name="tokens" value="DIV"/>
      <property name="tokens" value="PLUS"/>
      <property name="tokens" value="MINUS"/>
      <property name="tokens" value="STAR"/>
      <property name="tokens" value="MOD"/>
      <property name="tokens" value="SR"/>
      <property name="tokens" value="BSR"/>
      <property name="tokens" value="GE"/>
      <property name="tokens" value="GT"/>
      <property name="tokens" value="SL"/>
      <property name="tokens" value="LE"/>
      <property name="tokens" value="LT"/>
      <property name="tokens" value="BXOR"/>
      <property name="tokens" value="BOR"/>
```

```
358       <property name="tokens" value="LOR"/>
359       <property name="tokens" value="BAND"/>
360       <property name="tokens" value="LAND"/>
361       <property name="tokens" value="TYPE_EXTENSION_AND"/>
362       <property name="tokens" value="LITERAL_INSTANCEOF"/>
363       <property name="tokens" value="METHOD_REF"/>
364       <property name="option" value="nl"/>
365     </module>
366     <module name="OperatorWrap">
367       <property name="tokens" value="ASSIGN"/>
368       <property name="tokens" value="DIV_ASSIGN"/>
369       <property name="tokens" value="PLUS_ASSIGN"/>
370       <property name="tokens" value="MINUS_ASSIGN"/>
371       <property name="tokens" value="STAR_ASSIGN"/>
372       <property name="tokens" value="MOD_ASSIGN"/>
373       <property name="tokens" value="SR_ASSIGN"/>
374       <property name="tokens" value="BSR_ASSIGN"/>
375       <property name="tokens" value="SL_ASSIGN"/>
376       <property name="tokens" value="BXOR_ASSIGN"/>
377       <property name="tokens" value="BOR_ASSIGN"/>
378       <property name="tokens" value="BAND_ASSIGN"/>
379       <property name="option" value="eol"/>
380     </module>
381     <module name="ParenPad"/>
382     <module name="SeparatorWrap">
383       <property name="tokens" value="DOT"/>
384       <property name="tokens" value="AT"/>
385       <property name="tokens" value="METHOD_REF"/>
386       <property name="option" value="nl"/>
387     </module>
388     <module name="SeparatorWrap">
389       <property name="tokens" value="COMMA"/>
390       <property name="tokens" value="RBRACK"/>
391       <property name="tokens" value="ARRAY_DECLARATOR"/>
392       <property name="tokens" value="ELLIPSIS"/>
393       <property name="tokens" value="SEMI"/>
394       <property name="option" value="EOL"/>
395     </module>
396     <module name="SingleSpaceSeparator">
397       <property name="validateComments" value="false"/>
398     </module>
399     <module name="TypecastParenPad"/>
400     <module name="WhitespaceAfter"/>
401     <module name="WhitespaceAround"/>
402
```

```
403
404   </module>
405 </module>
```

Listing 5.3: XML

Sono stati creati diversi casi di analisi statica per il progetto svolto. I principali controlli fatti riguardano:

- Block Checks
- Class Design
- Coding
- Headers
- Imports
- Javadoc Comments
- Metrics
- Miscellaneous
- Modifiers
- Naming Conventions
- Regexp
- Size Violation
- Whitespace

Il risultato raggiunto è l'assenza di errori nell'analisi statica all'interno del progetto, dopo aver modificato parti del codice. I principali problemi riscontrati riguardano gli Whitespace e gli Import.

```
<?xml version="1.0" encoding="UTF-8"?>
<checkstyle version="8.27">
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\Constant.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\android\activities\GdxActivity.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\android\activities\MainActivity.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\android\activities\RegisterActivity.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\android\activities\package-info.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\AbstractScreens.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Entity.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\GameClass.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Player.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Screen\GameScreen.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Screen\LoseScreen.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Screen\MenuScreen.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Screen\OptionScreen.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Screen\WinScreen.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Screen\package-info.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Tile.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\Tilemap.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\libgdx\package-info.java">
</file>
<file name="D:\ProgettoKokoko\ProgettoInfo3\app\src\main\java\com\example\kokoko\package-info.java">
</file>
</checkstyle>
```

Figure 5.1: File di output degli errori dell'analisi statica

Capitolo 6

Planning

6.1 Passi da intraprendere

Ora che le basi dell'applicazione e del gioco sono state implementate bisogna pensare a come aggiungere tutti i casi d'uso descritti. Per la scoreboard sarà necessario implementare una nuova classe in cui utilizzare l'algoritmo descritto precedentemente; per il log-out sarà sufficiente inserire un tasto che permetta all'utente di tornare al menù d'accesso scollegandolo dall'applicazione. Per la scelta di modalità di gioco bisognerà aggiungere qualche schermata in cui dare all'utente la possibilità di scegliere cosa desidera fare. Oltre a questo bisogna iniziare ad implementare l'algoritmo di match-making e decidere dove inserirlo.

Glossary

algoritm è una strategia che serve per risolvere un problema ed è costituito da una sequenza finita di operazioni (dette anche istruzioni), consente di risolvere tutti i quesiti di una stessa classe. 9, 12, 23, 25, 51

architettura l'insieme dei criteri di progetto in base ai quali è progettato e realizzato un computer, oppure un dispositivo facente parte di esso. Per estensione, descrivere l'architettura di un dispositivo significa, in particolare, elencarne le sottoparti costituenti ed illustrarne i rapporti interfunzionali. 9

casi d'uso tecnica usata nei processi di ingegneria del software per effettuare in maniera esaustiva e non ambigua, la raccolta dei requisiti al fine di produrre software di qualità. 9, 11, 13

crash indica il blocco o la terminazione improvvisa, non richiesta e inaspettata di un programma in esecuzione (sistema operativo o applicazione), oppure il blocco completo dell'intero computer. 12, 13

database un insieme di dati strutturati ovvero omogeneo per contenuti e formato, memorizzati in un computer, rappresentando di fatto la versione digitale di un archivio dati o schedario. 12

iterazione l'atto di ripetere un processo con l'obiettivo di avvicinarsi a un risultato desiderato. 1, 9, 17

match-making designa l'unione di due individui per affinità. 12

off-line usato quale sinonimo della locuzione non in linea o fuori linea. 11–13

on-line usato quale sinonimo della locuzione in linea. 12, 13

planning termine usato per prevedere in linea di massima quando compiere un'attività e/o una serie di attività. 9

testing indica un procedimento, che fa parte del ciclo di vita del software, utilizzato per individuare le carenze di correttezza, completezza e affidabilità delle componenti software in corso di sviluppo.. 9

Acronyms

UML Unified Modeling Language. 5, 11, 15