
# Sudoku

PROGETTO FINALE

Mobile Programming | 2021/2022

Simone Staccone | Simone Bauco

# Indice

# 1. Introduzione

## 1.1 Scopo del documento

In questo documento verrà descritta in dettaglio l'applicazione Sudoku, partendo dall'analisi dei requisiti, attraversando tutte le fasi di progettazione, per arrivare all'implementazione.

## 1.2 Descrizione dei requisiti

- Il progetto deve sviluppare una interfaccia per il gioco del Sudoku.
  - Vedi: https://it.wikipedia.org/wiki/Sudoku
- L'App dovrà prevedere:
  - Un generatore di schemi (vedi librerie su internet)
  - Una interfaccia utente (ben fatta!!!)
  - Una fase di gioco
  - Un database dei risultati precedenti
    - Partire giocate
    - Partite vinte (terminate)
    - Miglior Tempo
  - L'interfaccia dovrà prevedere il massimo degli aiuti possibili all'utente:
    - Suggerisci mossa (intelligente)
    - Appunti utente per ognuna delle 81 celle
    - https://www.wikihow.it/Risolvere-il-Sudoku
    - Fare riferimento ad app già esistenti
  - Gli schemi presentati all'utente dovranno essere
    - Automaticamente generati da un motore API
      - https://sugoku.herokuapp.com/board?difficulty=medium
    - Da un sito online sviluppato dal gruppo

## 1.3 Tecnologie utilizzate

- Linguaggio utilizzato: Kotlin
- Framework UI: Compose
- Librerie utilizzate: Room, Volley, Kotlin.Duration, GSON, ConstraintLayout

# 2. Analisi dei requisiti

## 2.1 User Stories

- **As a user, I want to play sudoku.**
- **As a player, I want to resume the previously started games.**
- **As a player, I want to keep track of my best time of resolution.**
- **As a beginner, I want a rules guide, so that I can learn how to play.**
- **As a beginner, I want to be helped with hints, so that I can avoid errors.**
- **As a player, I want to keep notes on each cell, so that I can keep track of a possible solution.**
- **As a player, I want to choose the difficulty of the sudoku.**
- **As a player, I want a scoring system, so that I can see my top scoring.**
- **As a player, I want the possibility to delete unfinished games.**
- **As a player, I want to customize the app, saving my preferences.**
- As a player, I want to share with my friend a screenshot of the completed sudoku.
- As a player, I want to choose the background of the sudoku.
- As a player, I want a daily challenge.

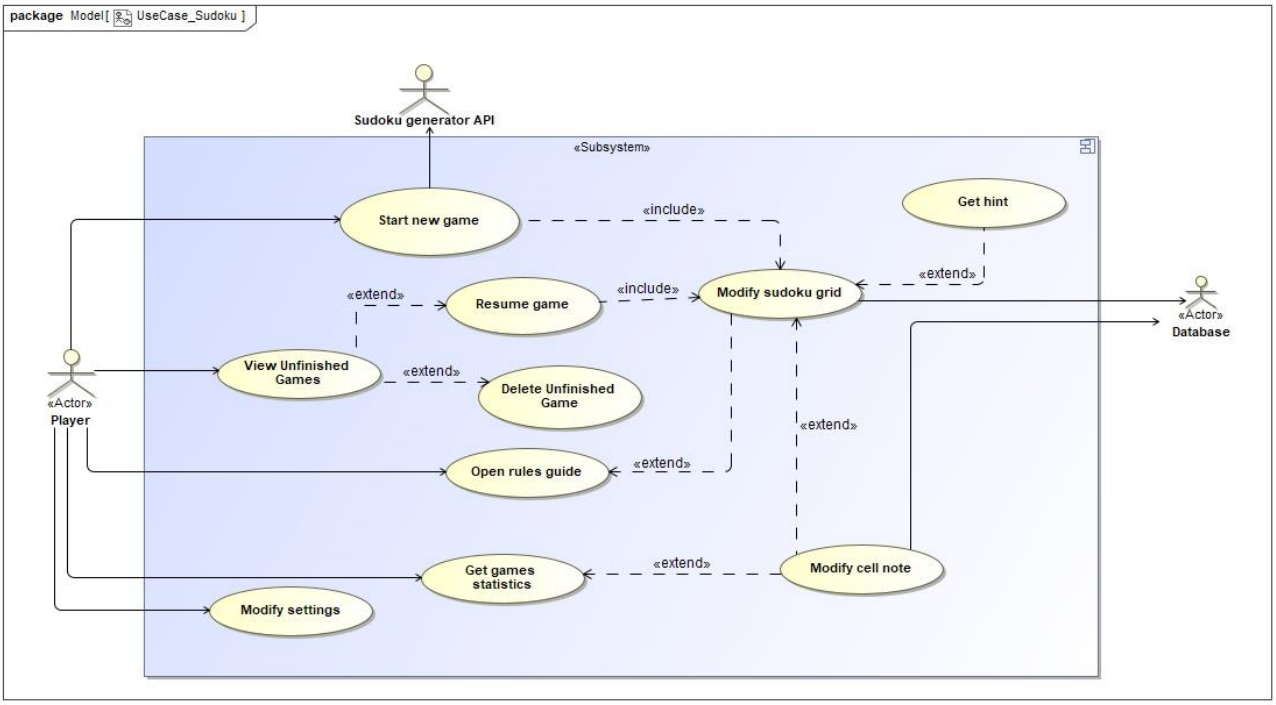
Le user stories non in grassetto sono idee iniziali, ma non implementate.

## 2.2 Functional Requirements

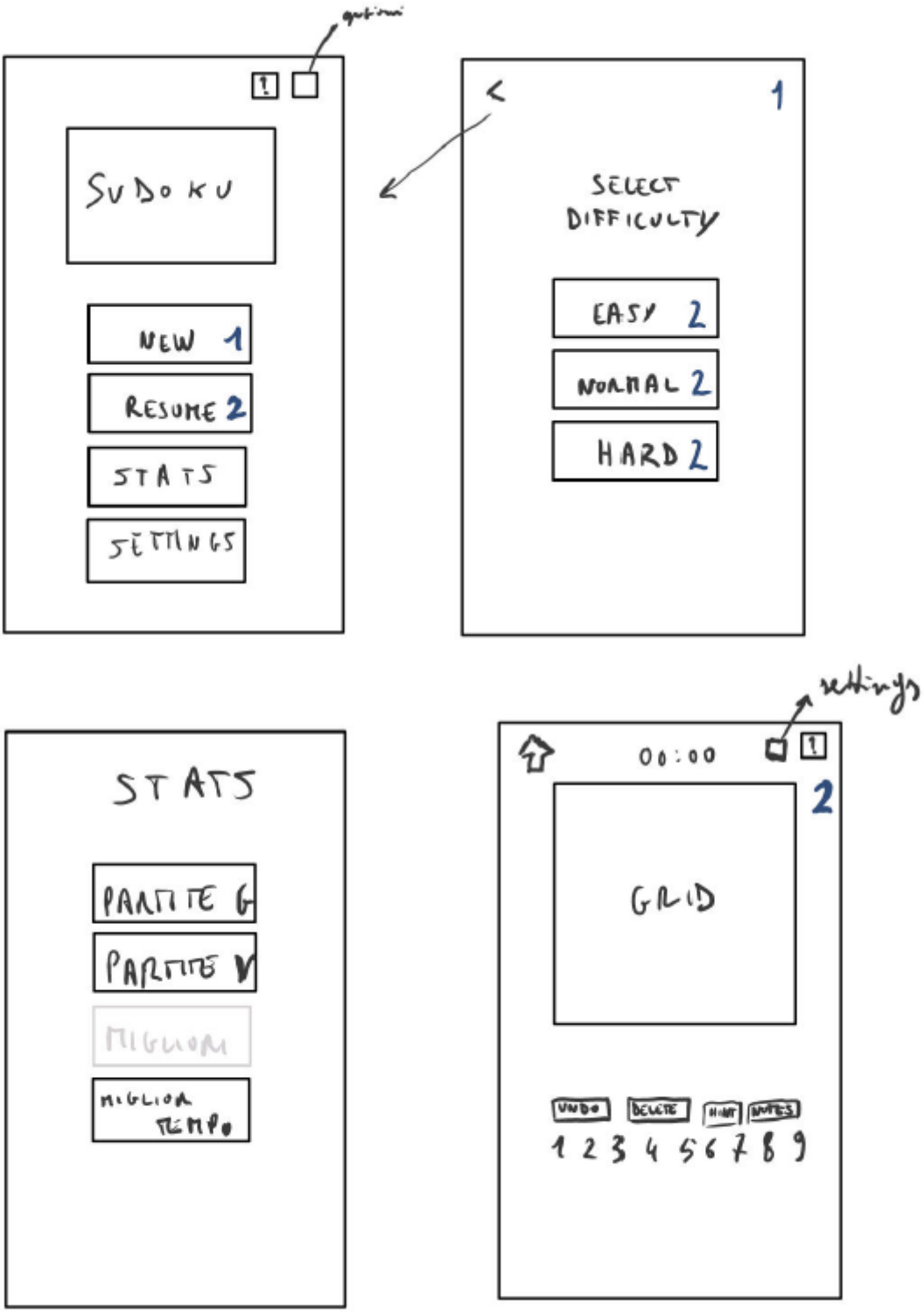
- **The system shall provide a gaming phase.**
- **The system shall keep track of all games:**
  - **Won (finished)**
  - **Started (started but not finished)**
- **The system shall resume unfinished games**
- **The system shall keep track of the best times of resolution.**
- **The system shall provide a section named "Rules", which shall contain a guide on how to play.**
- **The system shall be able to solve sudokus.**
- **The system shall provide hints (on request), showing the next number to be placed in a cell.**
- **The system shall register notes for each cell during the gaming phase.**
- **The system shall select different sudokus basing on the difficulty level.**
- **The system shall provide a scoring system.**
- **The system shall provide the possibility to delete unfinished games.**
- **The system shall keep track of the following user preferences, using settings:**
  - **Show/hide timer**
  - **Show/hide score**
  - **Enable/disable hints**
  - **Enable/disable dark mode**
- The system shall share finished games by taking an image of the grid.
- The system shall provide different backgrounds for the UI.
- The system shall provide a daily challenge system.
- The system shall provide a ranking board.

I requisiti non in grassetto sono idee iniziali, ma non implementate.

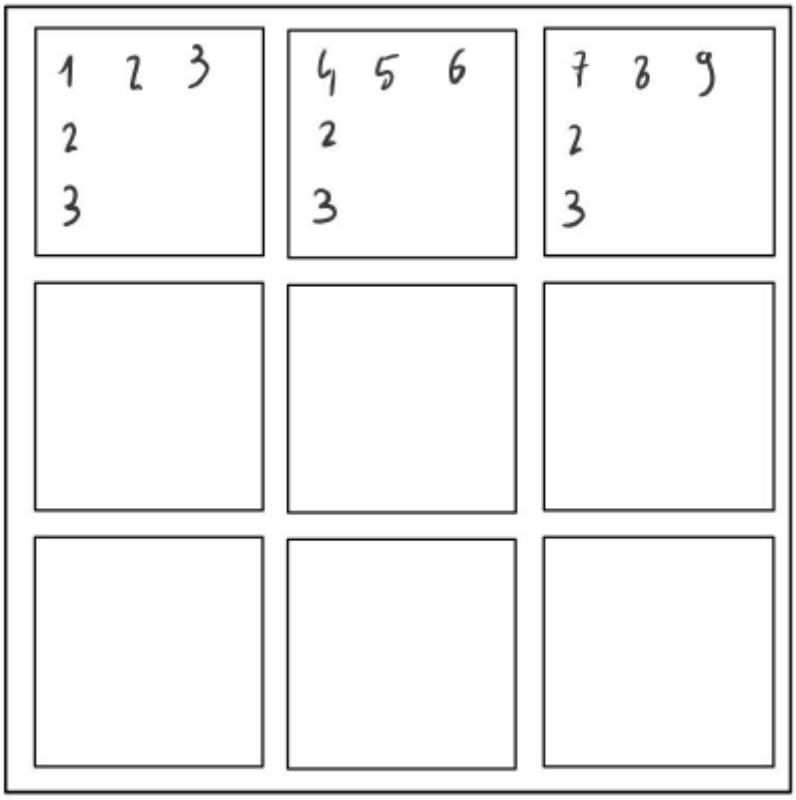
## 2.3 Use Case Diagram



## 3. Storyboards




Alcune storyboards iniziali (schermata home, selezione difficoltà, statistiche e di gioco)

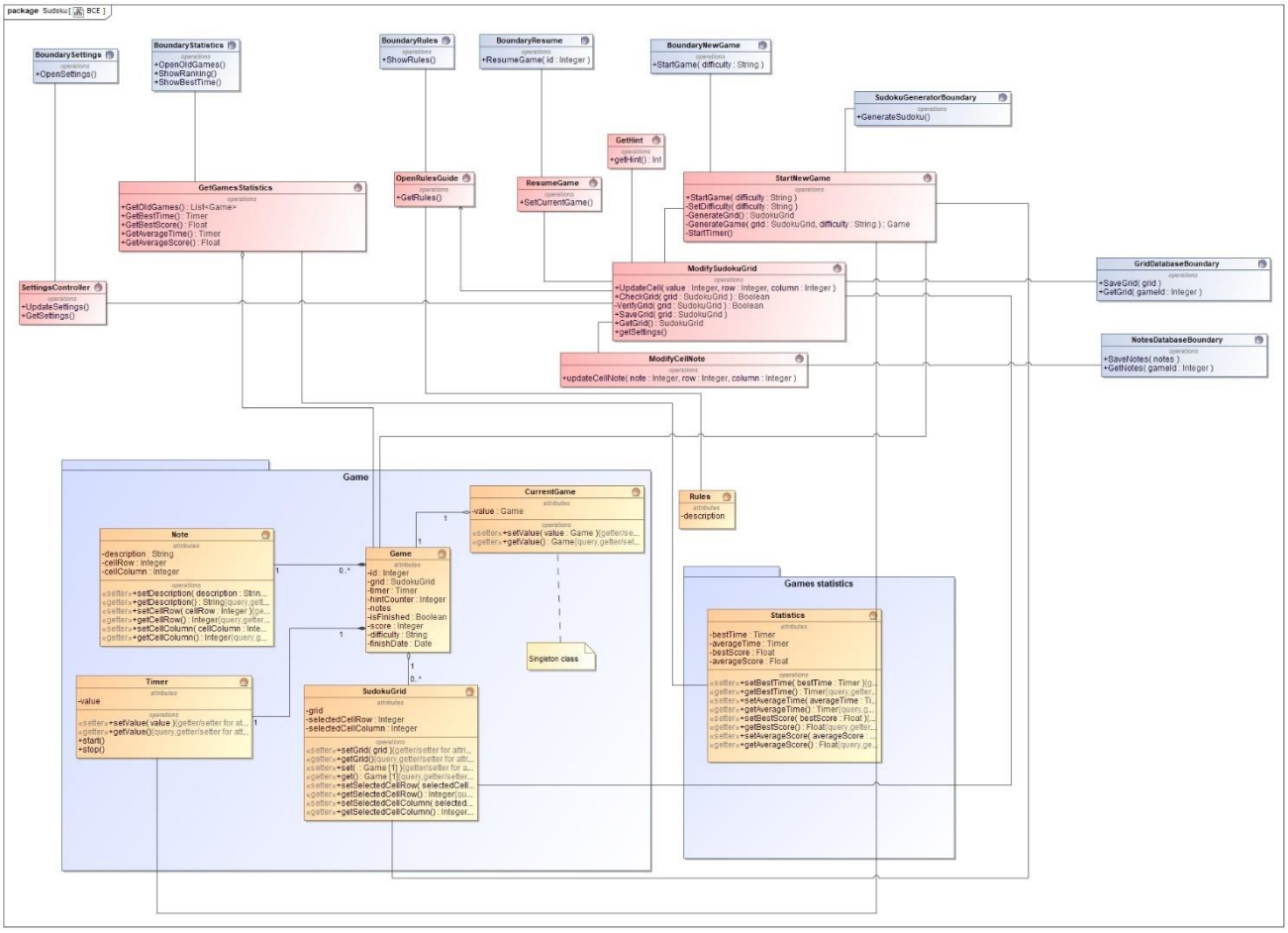



Schema della griglia

# 4. Design

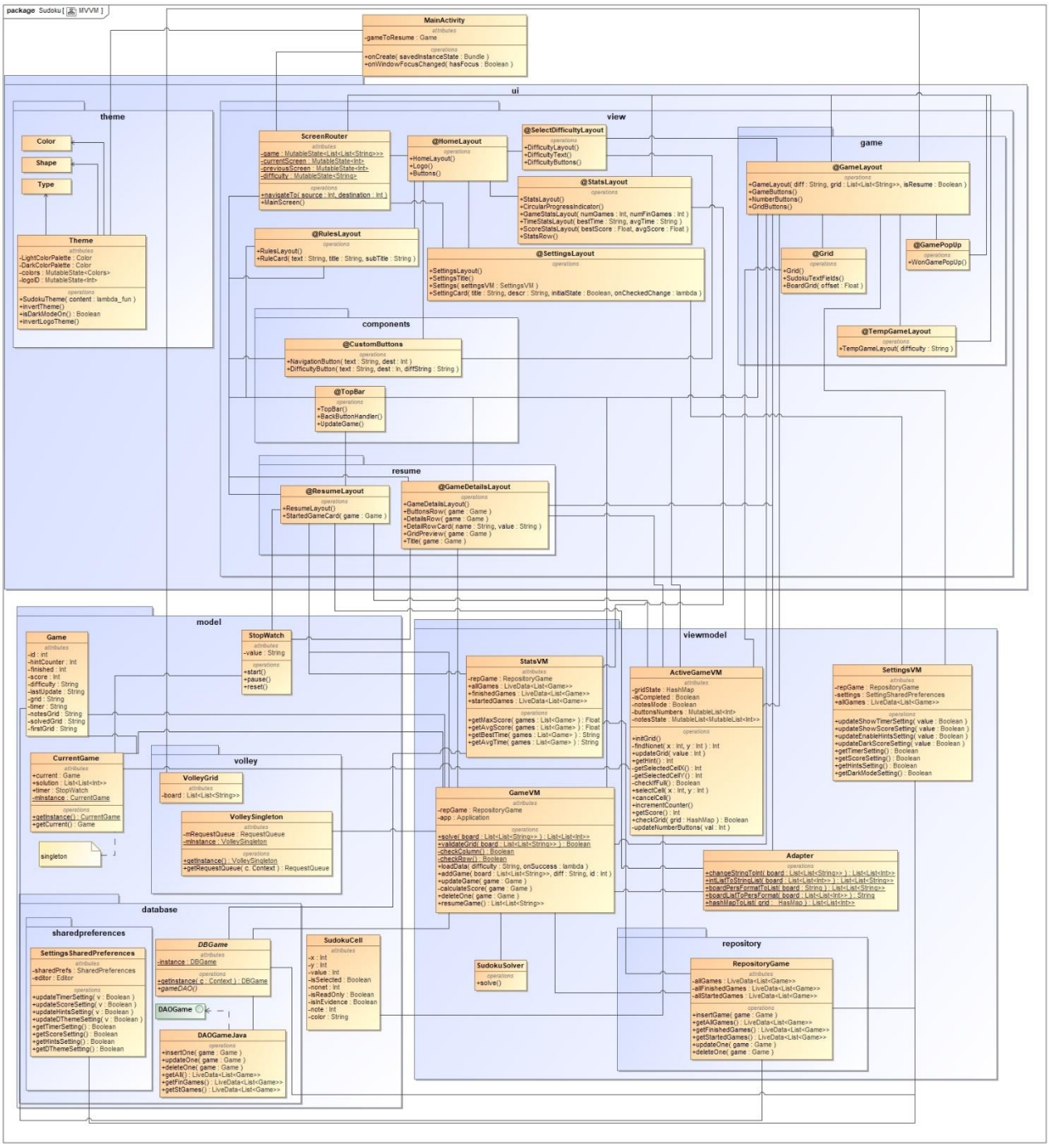
## 4.1 Class Diagram (paradigma BCE)

Da una prima analisi dello Use Case diagram, abbiamo prodotto uno schema di tipo Class Diagram adottando il paradigma Boundary Control Entity (BCE).



## 4.2 Class Diagram (paradigma MVVM)

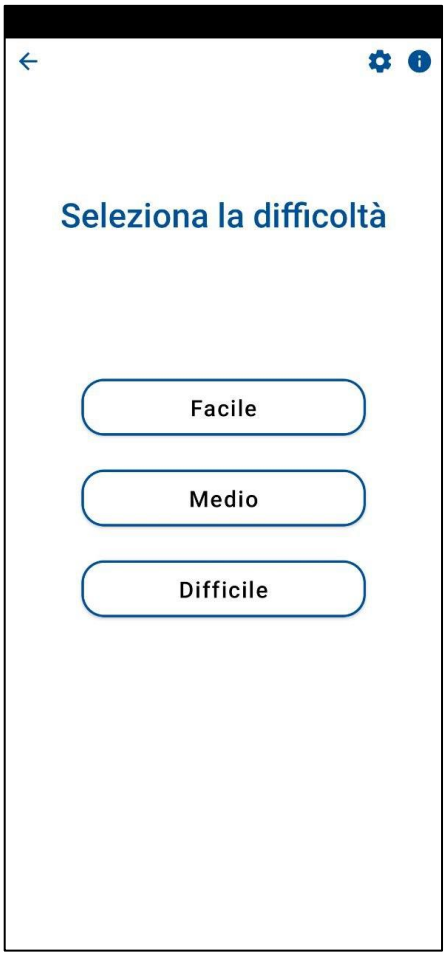
In una fase avanzata di progettazione, il modello precedente è stato raffinato per essere adattato al paradigma Model View ViewModel, tipico dell'ambiente Android.



# 5. Appendice A - Breve manuale d'uso

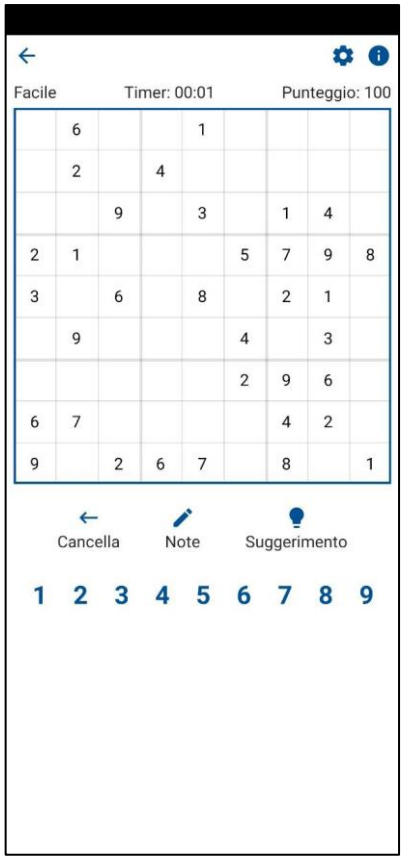

**Schermata Home**: qui sono presenti 6 bottoni:

- Nuova partita: permette di iniziare una nuova partita, apre la schermata "Seleziona difficoltà"
- Riprendi: visibile solo nel caso di partite iniziate, apre la schermata "Visualizza partite iniziate"
- Statistiche: apre la schermata "Statistiche"
- Impostazioni: apre la schermata "Impostazioni"
- Icona "ingranaggio": apre la schermata "Impostazioni"
- Icona "info": apre la schermata "Regole"




Schermata "**Seleziona la difficoltà**", presenta 6 bottoni

- Facile: consente di avviare una nuova partita a difficoltà "facile"
- Medio: consente di avviare una nuova partita a difficoltà "medio"
- Difficile: consente di avviare una nuova partita a difficoltà "difficile"
- Icona "ingranaggio": apre la schermata "Impostazioni"
- Icona "info": apre la schermata "Regole"
- Icona "indietro": permette di ritornare alla schermata precedente (Schermata Home)

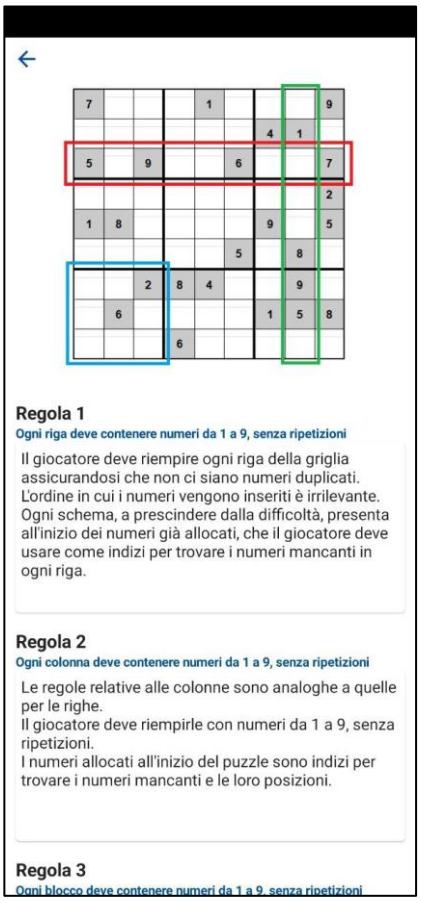



Schermata di **gioco**, contiene i seguenti elementi:

- Icona “ingranaggio”: apre la schermata “Impostazioni”;
- Icona “info”: apre la schermata “Regole”;
- Icona “indietro”: permette di ritornare alla schermata precedente (Schermata Home);
- Informazioni sulla partita corrente (difficoltà, timer e punteggio);
- Griglia di gioco: tutte le celle vuote all’inizio della partita sono modificabili, le celle iniziali non sono selezionabili;
- Icona “Cancella”: permette di eliminare il contenuto della cella selezionata;
- Icona “Note”, permette di entrare in modalità Note (i valori inseriti nelle caselle saranno salvati come note);
- Icona “Suggerimento”, permette di conoscere il contenuto della cella selezionata (ogni volta che si richiede un suggerimento, il punteggio diminuisce);
- Numeri da 1 a 9, bottoni che consentono di inserire un valore nella cella selezionata

Schermata **Regole**, contiene i seguenti elementi:

- Icona “indietro”: permette di ritornare alla schermata precedente (Schermata precedente);
- Colonna scorrevole contenente le principali regole del sudoku

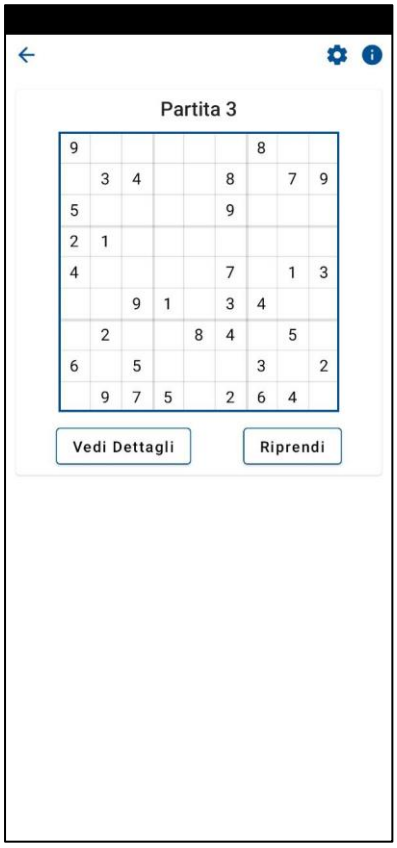



Schermata **Impostazioni**, contiene i seguenti elementi:

- Impostazioni di gioco: consentono di modificare le proprie preferenze riguardo timer, punteggio e suggerimenti. In particolare, l'utente può scegliere se visualizzare o no timer e punteggio e se abilitare o meno i suggerimenti;
- Icona "indietro": permette di ritornare alla schermata precedente (Schermata precedente);
- Impostazione tema: consente di abilitare o disabilitare il tema scuro.

Schermata di **Statistiche**, contiene i seguenti elementi:

- Icona "indietro": permette di ritornare alla schermata precedente (Schermata precedente);
- Indicatore della percentuale di partite vinte;
- Statistiche relative alle partite: indicano il numero di partite giocate (partite iniziate + partite terminate) e il numero di partite vinte (terminate);
- Statistiche relative al tempo: indicano il miglior tempo e il tempo medio, calcolati su tutte le partite terminate;
- Statistiche relative al punteggio: indicano il miglior punteggio e il punteggio medio, calcolati su tutte le partite terminate.




Schermata **Visualizza partite iniziate**, contiene:

- Una card, per ogni partita, contenente l'id della partita, la griglia e due bottoni, uno che apre la schermata dei dettagli della partita e uno che permette di riprendere la partita;
- Icona "ingranaggio": apre la schermata "Impostazioni";
- Icona "info": apre la schermata "Regole";
- Icona "indietro": permette di ritornare alla schermata precedente (Schermata precedente).

Schermata **Dettagli della partita**, contiene:

- Icona "ingranaggio": apre la schermata "Impostazioni";
- Icona "info": apre la schermata "Regole";
- Icona "indietro": permette di ritornare alla schermata precedente (Schermata precedente);
- Id della partita;
- Griglia in sola lettura;
- Informazioni riguardanti la difficoltà, la data di ultima modifica, il punteggio e il timer della partita in questione.
- Un'icona "cestino" che elimina la partita;
- Un bottone che permette di riprendere la partita.

## 6. Appendice B – Gradle

```
plugins {
    id 'com.android.application'
    id 'org.jetbrains.kotlin.android'
    id "kotlin-parcelize"
    id "kotlin-kapt"
}

android {
    compileSdk 32
    defaultConfig {
        applicationId "mp.sudoku"
        minSdk 28
        targetSdk 32
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
        vectorDrawables {
            useSupportLibrary true
        }
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-
optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
    buildFeatures {
        compose true
    }
    composeOptions {
        kotlinCompilerExtensionVersion compose_version
    }
    packagingOptions {
        resources {
            excludes += '/META-INF/{AL2.0,LGPL2.1}'
        }
    }
}

dependencies {
    implementation 'androidx.activity:activity-ktx:1.4.0'
    implementation 'androidx.core:core-ktx:1.7.0'
    implementation "androidx.compose.ui:ui:$compose_version"
    implementation "androidx.compose.material:material:$compose_version"
    implementation "androidx.compose.ui:ui-tooling-preview:$compose_version"
    implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.4.1'
```

```
    implementation 'androidx.activity:activity-compose:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
    androidTestImplementation "androidx.compose.ui:ui-test-
junit4:$compose_version"
    debugImplementation "androidx.compose.ui:ui-tooling:$compose_version"
    implementation "androidx.room:room-runtime:$room_version"
    kapt "androidx.room:room-compiler:$room_version"
    implementation "androidx.room:room-ktx:$room_version"
    annotationProcessor "androidx.room:room-compiler-processing:$room_version"
    implementation "androidx.constraintlayout:constraintlayout-compose:1.1.0-
alpha02"
    implementation "androidx.compose.runtime:runtime-livedata:$compose_version"
    implementation 'com.android.volley:volley:1.2.1'
    implementation 'com.google.code.gson:gson:2.8.6'
    implementation "androidx.compose.material:material-icons-
extended:$compose_version"
}
```

file Build.Gradle (module)

```
buildscript {
    ext {
        compose_version = '1.1.1'
        room_version = '2.4.2'
    }
}// Top-level build file where you can add configuration options common to all
sub-projects/modules.
plugins {
    id 'com.android.application' version '7.1.2' apply false
    id 'com.android.library' version '7.1.2' apply false
    id 'org.jetbrains.kotlin.android' version '1.6.10' apply false
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

file Build.Gradle (project)