

# Testing

Versione	Data	Descrizione
1.0	19 Gennaio 2026	Casi di test e copertura codice
1.1	28 Gennaio 2026	Modifica copertura codice

## Casi di test di unità

I test di unità sono stati effettuati con il tool JUnit 5 e grazie all'aiuto della struttura del progetto Maven: abbiamo infatti una cartella apposta `src/test/java` dove poter creare i package e le classiTest corrispondenti alle rispettive classi che andremo a testare. Il testing avviene grazie alla dichiarazione di variabili globali e di un metodo `@BeforeEach` `setUpAll()` che viene eseguito prima dell'esecuzione di ogni caso di test.

## Package Card

In questo package è stata testato il metodo `checkRank()` fornito dalla classe `Hand` per calcolare il suo Rank, ovvero il punteggio della mano. Sono stati effettuati diversi casi di test, uno per ogni possibile punteggio del gioco: carta alta, coppia, doppia coppia, tris, scala, colore, full, poker, scala colore.

## Package Game

In questo package è stata testata la classe `Game`:

- metodo `checkFirstTurn()` che calcola l'id del primo turno sull'insieme dei soli giocatori attivi: è stato testato in modo che gli id fossero non incrementali e non consecutivi, e uno dei giocatori ha foldato.
- metodo `nextTurn()` che calcola l'id del turno successivo sull'insieme dei soli giocatori attivi. E' stato testato in modo che gli id fossero non incrementali e non consecutivi e che:
  - si passasse dall'ultimo del giro al secondo, poichè il primo ha foldato
  - si passasse da un giocatore che ha foldato a quello successivo
- metodo `splitPot()` che distribuisce il piatto tra i soli giocatori attivi

## Package Server

L'architettura client/server ci obbliga a scambiare messaggi, quindi è stata dichiarata una classe `FakeSocketSafe` per testare il codice senza effettivamente farlo.

In questo package è stato testato una possibile combinazioni di scelte dei giocatori accentuando alcuni elementi:

- azioni fuori turno
- azioni non valide
- all-in

## Copertura

Grazie all'analisi di copertura dei test JUnit abbiamo ottenuto il report che descrive il grado di copertura delle istruzioni nel nostro codice: nel nostro progetto è stato coperto circa il 44% delle istruzioni, un valore ben più che accettabile.

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
▼ poker	44,0 %	3.628	4.610	8.238
▼ src/main/java	33,4 %	2.306	4.607	6.913
> THRProject.gui	0,0 %	0	2.162	2.162
▼ THRProject.server	37,0 %	785	1.337	2.122
> Server.java	37,7 %	720	1.189	1.909
> ClientHandler.java	31,7 %	65	140	205
> ServerMain.java	0,0 %	0	8	8
> THRProject.client	0,0 %	0	590	590
> THRProject.database	18,7 %	38	165	203
▼ THRProject.game	79,4 %	548	142	690
> Game.java	76,1 %	442	139	581
> Pot.java	93,3 %	42	3	45
> GamePhase.java	100,0 %	64	0	64
▼ THRProject.player	59,9 %	154	103	257
> Player.java	36,2 %	54	95	149
> PlayerStatus.java	92,6 %	100	8	108
▼ THRProject.message	62,4 %	133	80	213
> ActionType.java	0,0 %	0	74	74
> Message.java	60,0 %	9	6	15
> ControlType.java	100,0 %	124	0	124
▼ THRProject.card.model	81,6 %	124	28	152
> Card.java	37,5 %	15	25	40
> Suit.java	94,9 %	56	3	59
> CardComparator.java	100,0 %	19	0	19
> Rank.java	100,0 %	34	0	34
▼ THRProject.card.logic	100,0 %	524	0	524
> Deck.java	100,0 %	88	0	88
> Hand.java	100,0 %	436	0	436
> src/test/java	99,8 %	1.322	3	1.325