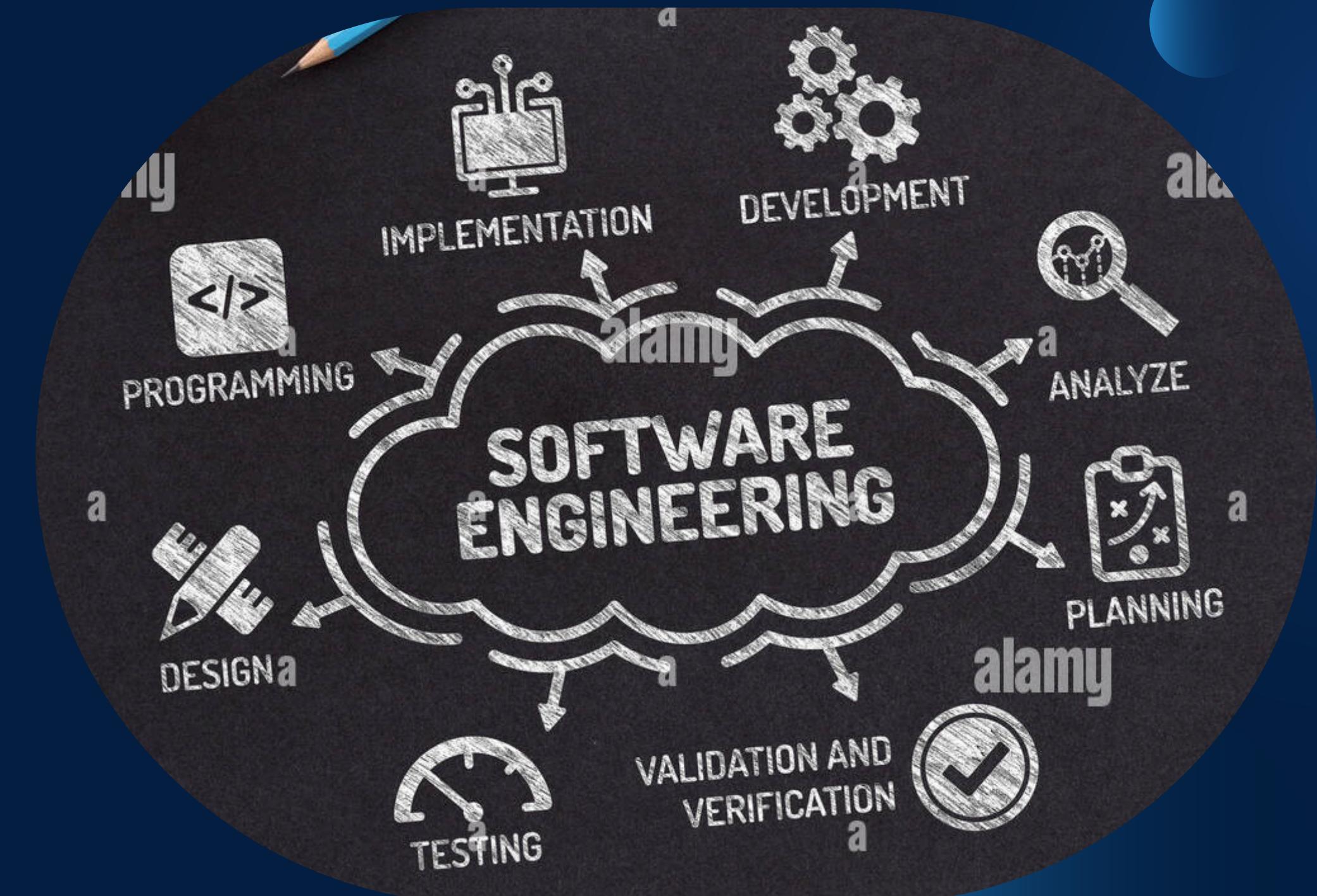


# SALA GIOCHI

## PROGETTO DI INGEGNERIA DEL SOFTWARE



Megna Matteo, Mondino Andrea, Scioscia Matilde

# OBIETTIVO

01

L'obiettivo era creare una piattaforma di intrattenimento funzionale applicando i principi di progettazione e sviluppo software alla creazione di un'applicazione desktop tramite un team working organizzato, prendendo spunto da strutture apprese durante il corso.

02

L'idea è stata quella di realizzare una sala giochi virtuale composta da un database che conservi gli account registrati e i punti totalizzati degli utenti iscritti all'interno della sala giochi.

03

La sala giochi è composta da 4 giochi: Tris , Battaglia navale, Lancio dei dadi e Roulette.



# DIFFICOLTA' INCONTRATE

## ► DEVICE ACCESS

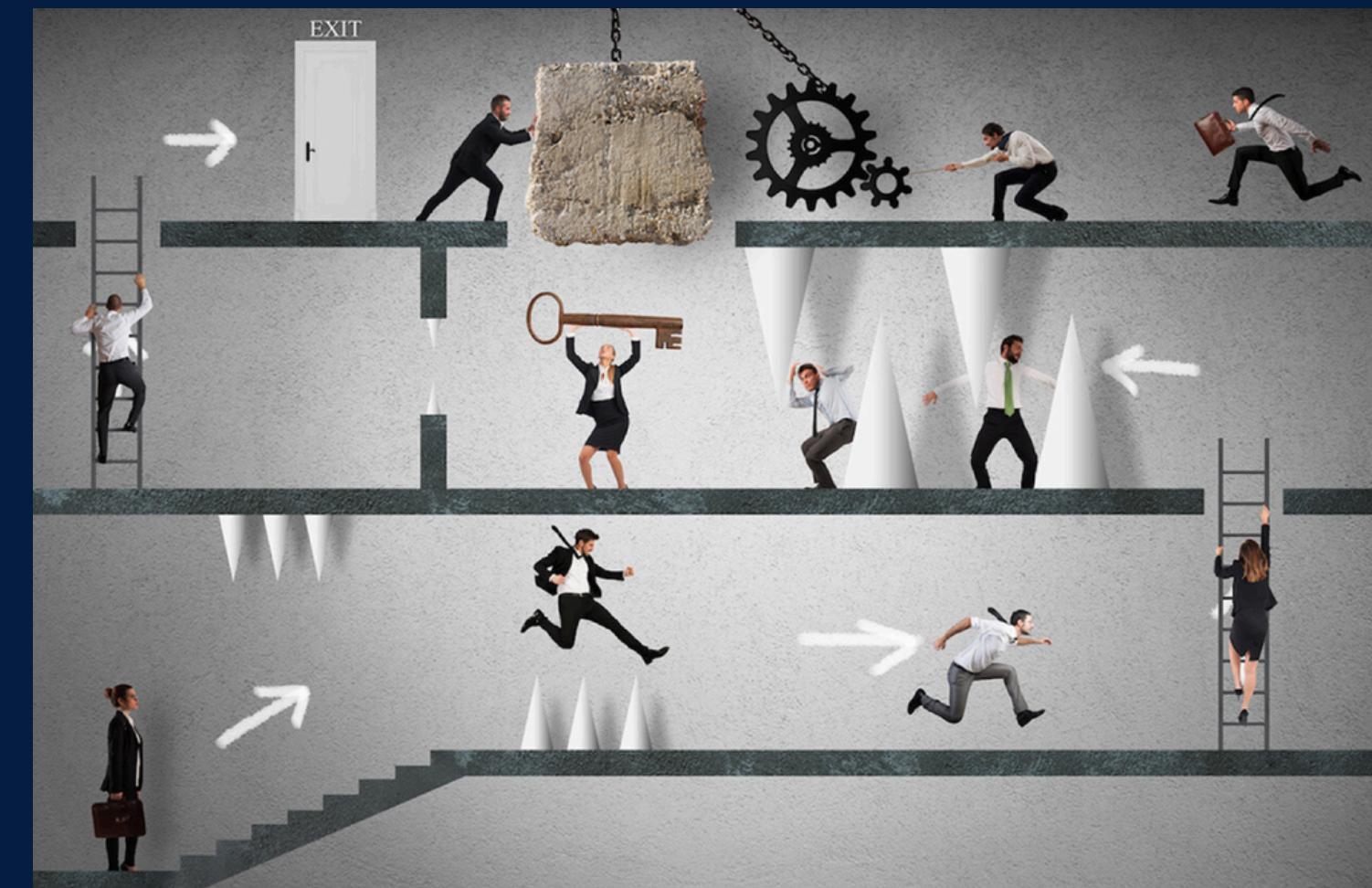
Connessione dei dispositivi personali all'interno del workplace

## ► DB INTEGRITY

Rischio di corruzione o cancellazione dei file del DB durante lo sviluppo

## ► PROCESS MODELLING

Utilizzo di Papayrous



# PARADIGMI DI PROGETTAZIONE E TOOLS



# SOFTWARE CONFIGURATION MANAGEMENT

## Version Control

Git/GitHub - Manual Continuos Integration , con risoluzione immediata dei conflitti

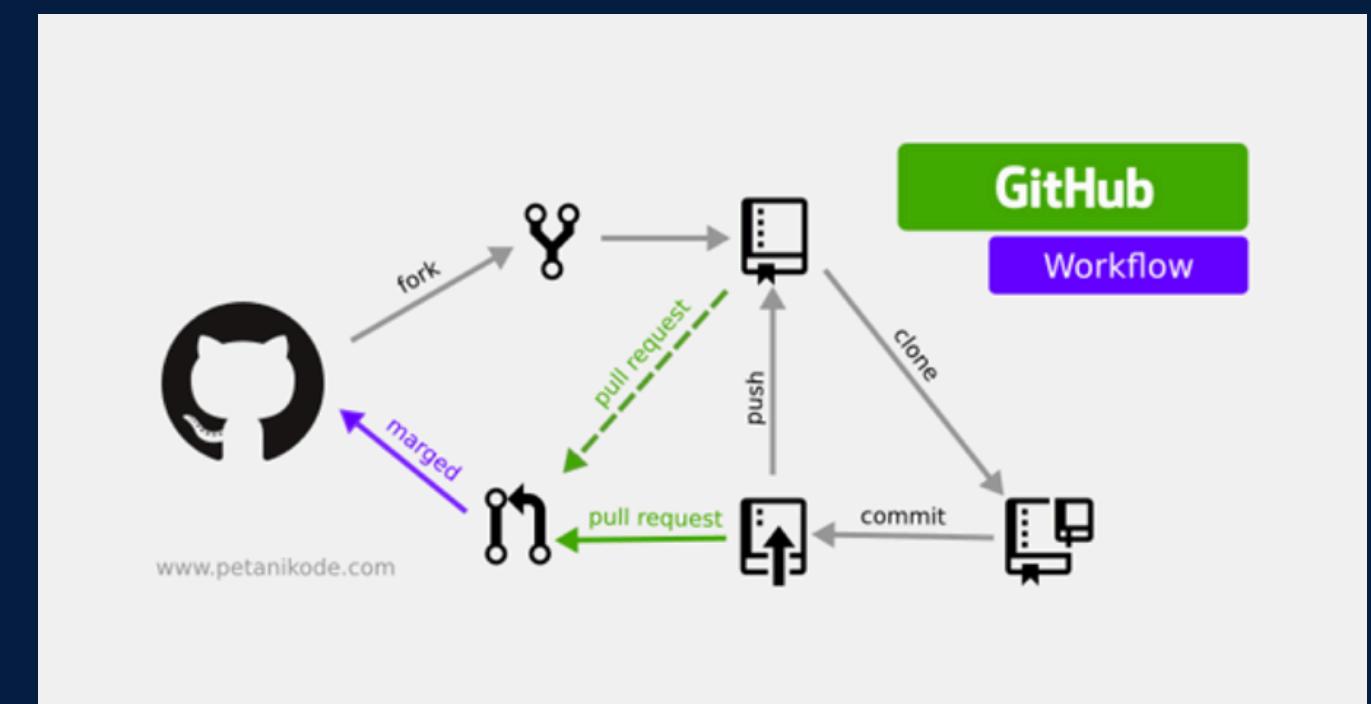


## Build Automation

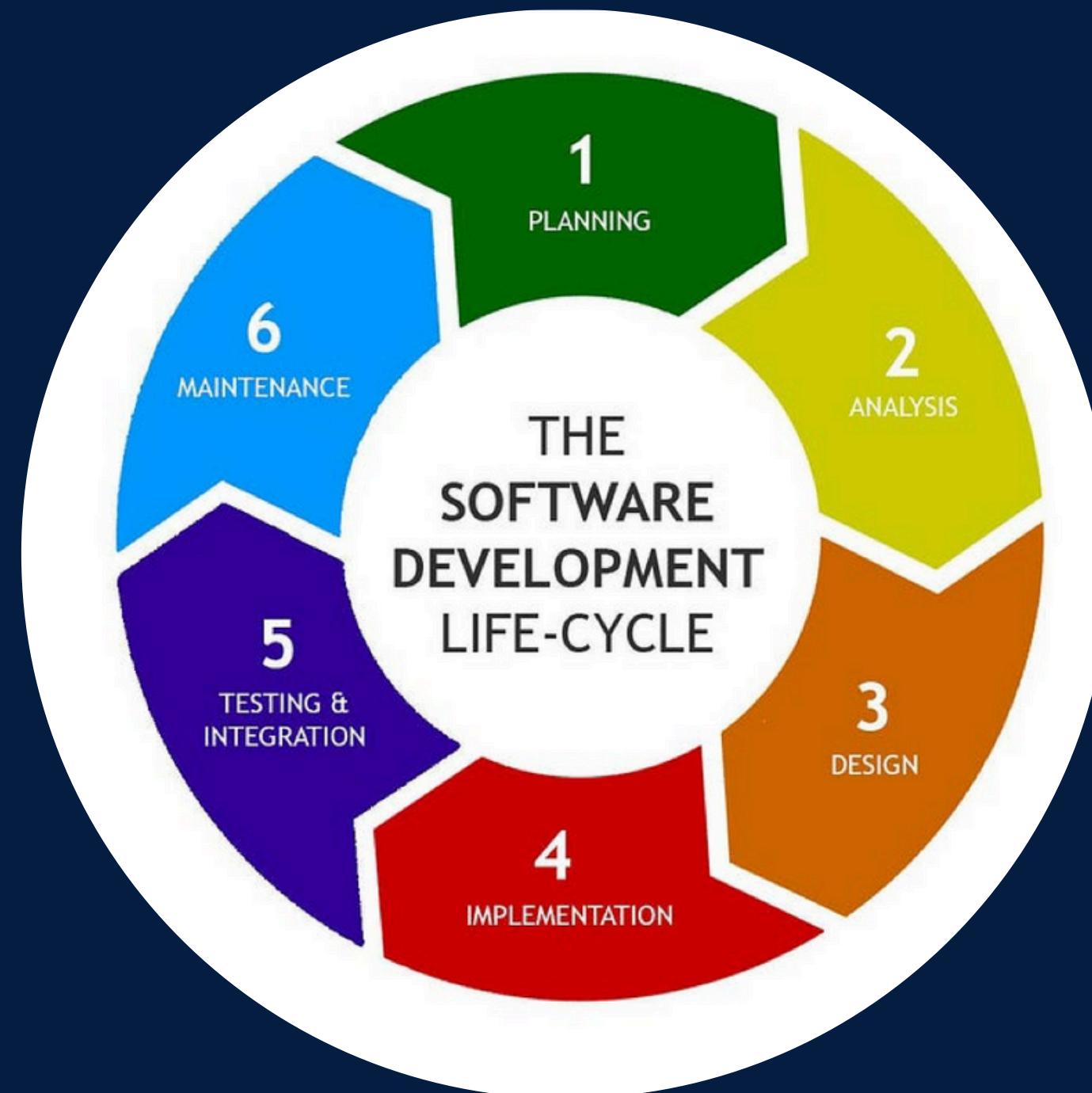
Maven - Integrazione delle dipendenze del progetto tramite pom.xml, per garantire l'uniformità delle versioni

## Analisi statica e qualità

SonarQube, PMD e stan4J - Revisione automatizzata



# SOFTWARE LIFE CYCLE



## SVILUPPO ITERATIVO E INCREMENTALE

- 01 CORE & INFRASTRUCTURE
- 02 UML & MODEL MAPPING
- 03 GAMEPLAY IMPLEMENTATION
- 04 REFINEMENT & FINALIZATION

# REQUISITI

## CODING STANDARD

- Convenzioni di denominazione standard di Java
- GitHub con branch principale main
- Aggiornamenti frequenti e sincronizzazione continua
- Progetto configurato come Maven project in Eclipse

## FUNZIONALI

- Registrazione
- Autenticazione
- Navigazione
- Gameplay
- Persistenza punteggi
- Classificazione
- Modalità anonima

## VINCOLI

- Portabilità
- Persistenza dati
- Interfaccia grafica
- Modellazione

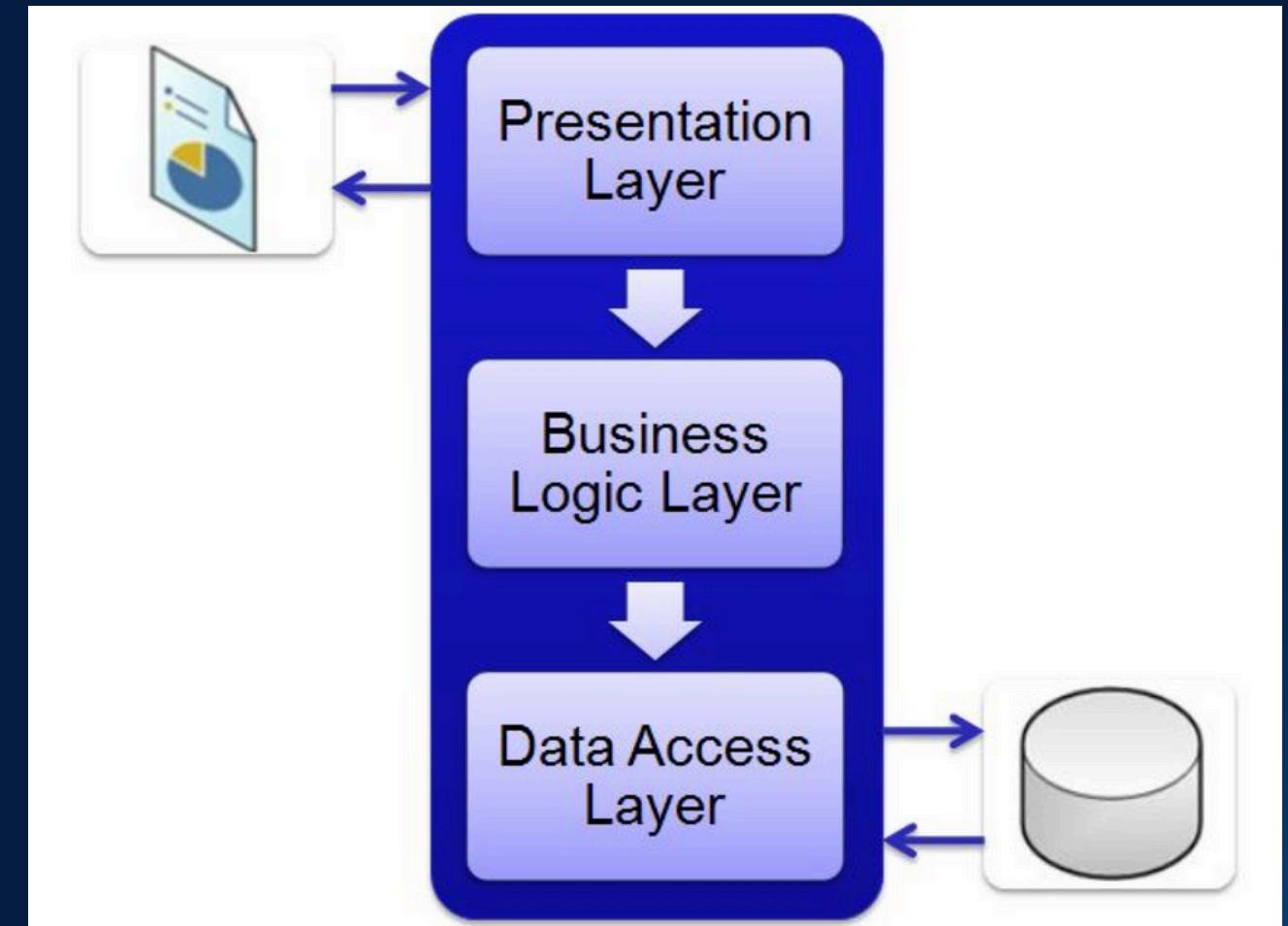
## OUT OF SCOPE

- Architettura distribuita
- Web deployment
- Multimedia e Adattabilità UI
- Target dispositivi mobili

# ARCHITETTURA

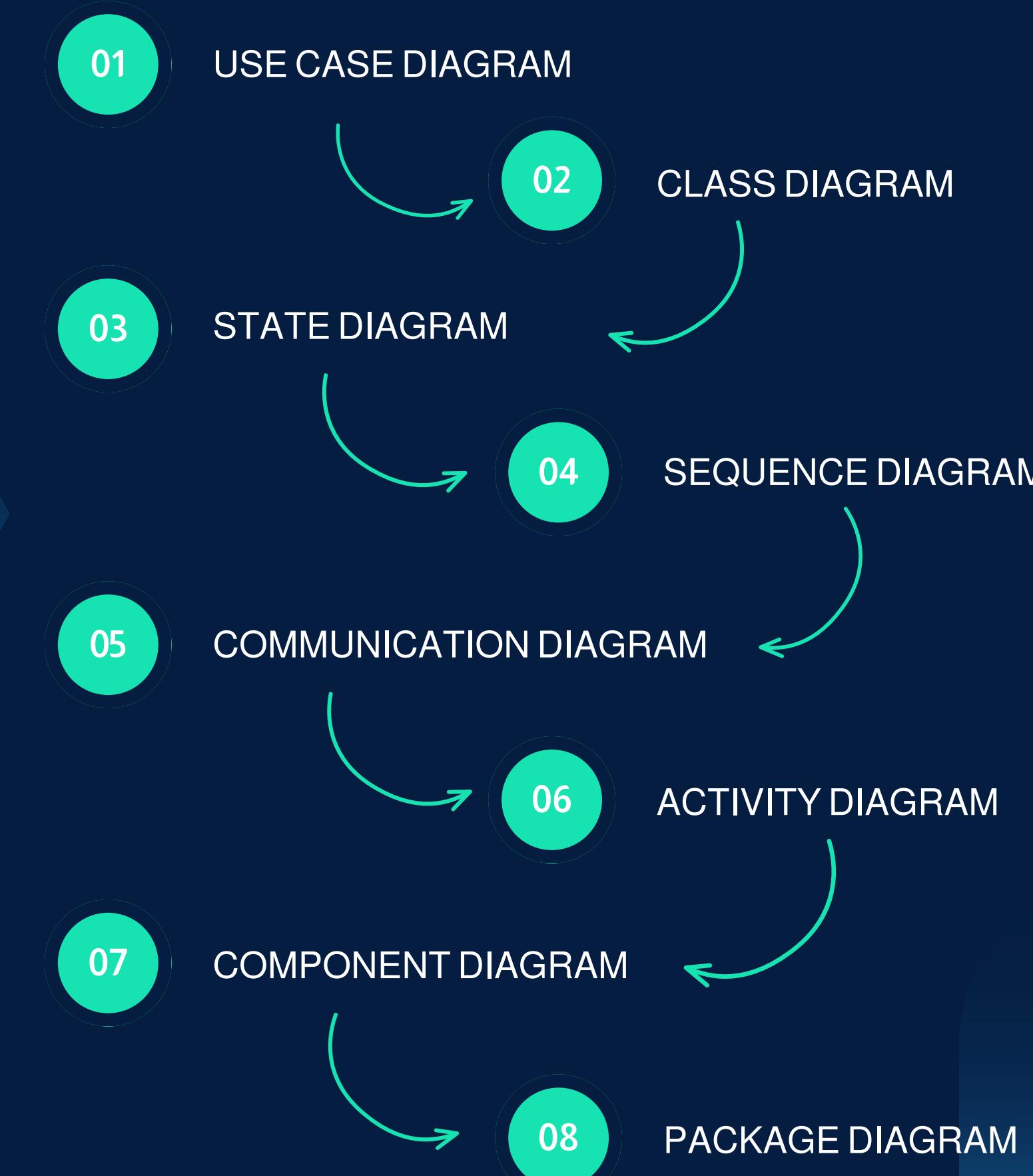
BASATA SU TRE LIVELLI:

- ▶ **PRESENTATION LAYER (GUI)**  
com.mondsciomegn.salagiochi.gui
- ▶ **LOGIC LAYER (GAME)**  
com.mondsciomegn.salagiochi.videogame
- ▶ **DATA ACCESS LAYER (DATABASE)**  
com.mondsciomegn.salagiochi.db





# MODELLAZIONE





## IMPLEMENTATION

# IMPLEMENTAZIONE

### ► MODELLO ORGANIZZATIVO IBRIDO

Chief Programmer Team:

- Chief Programmer
- Programmatori

Utilizzato un approccio agile per la modulazione

### ► MODALITA' OPERATIVA

- Pairing Verticale
- Pairing Orizzontale

### ► BENEFICI

- Condivisione continua delle conoscenze
- Code review integrata nel processo
- Migliore qualità del codice e riduzione dei bug

# TESTING



## MANUALE

- GRAFICA
- LOGICA
- IMPLEMENTAZIONE



## AUTOMATICO

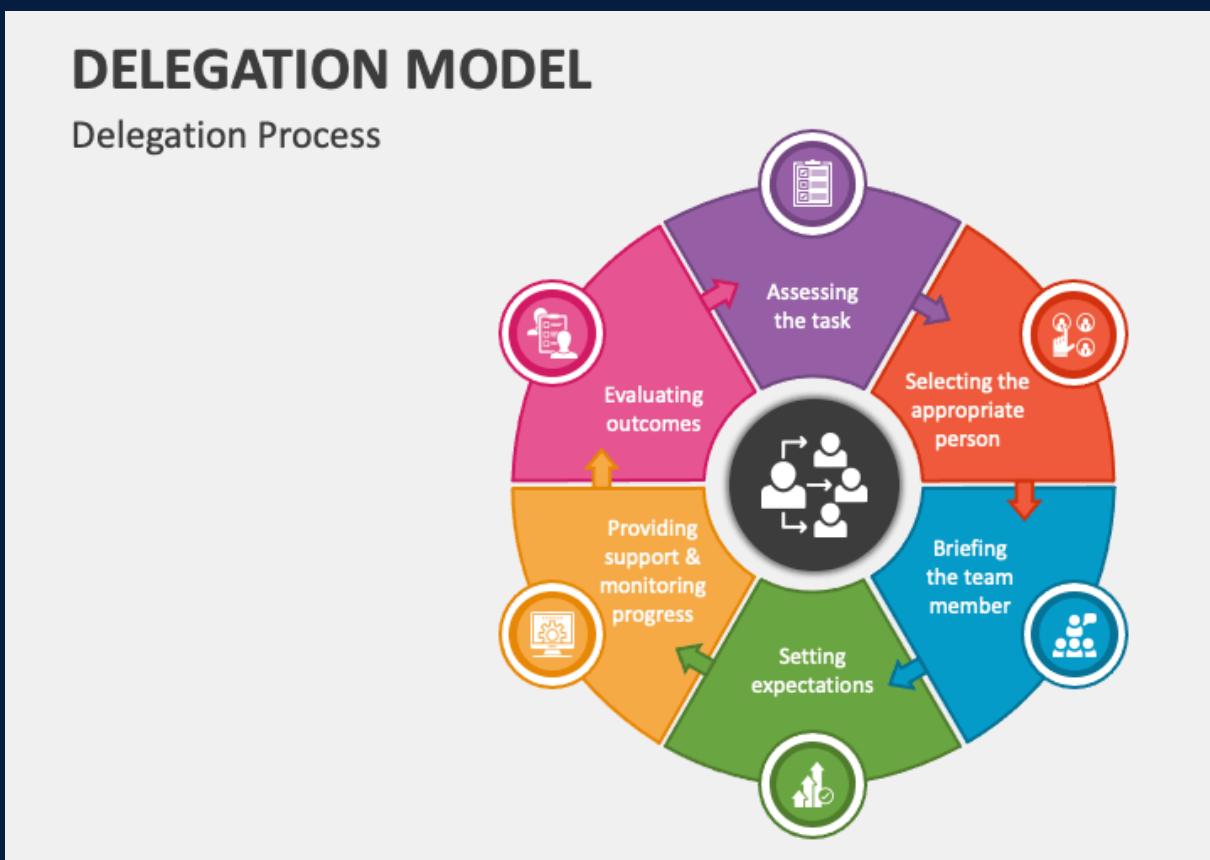
- JUNIT



# DESIGN PATTERN

## DELEGATION PATTERN

- Battleship delega la gestione della partita a due istanze di Player
- Player delega la gestione della griglia e dei colpi a Field
- Field delega la costruzione grafica a GuiGrid



## FACADE PATTERN

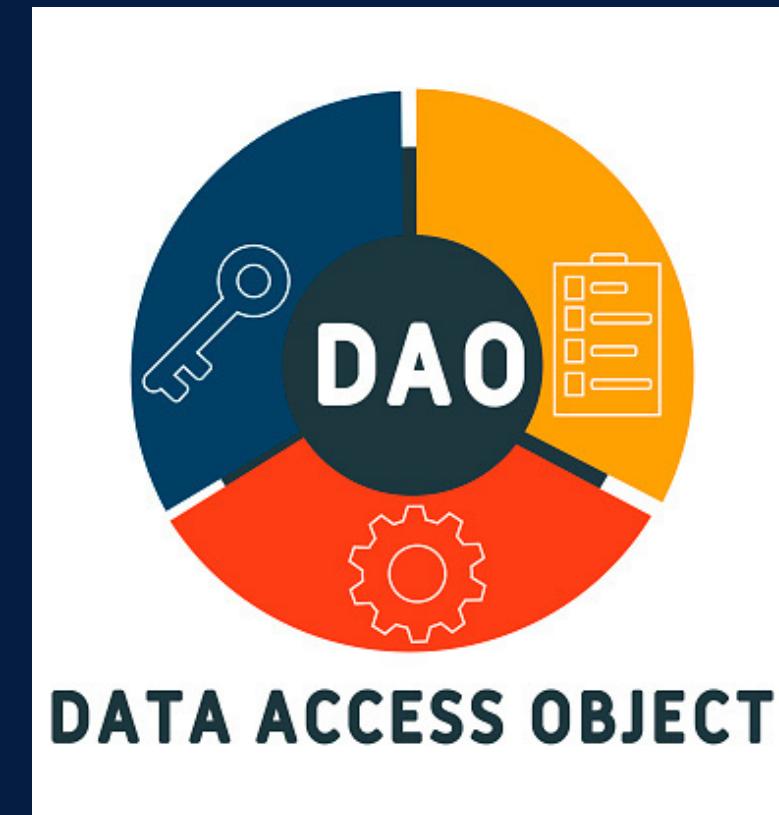
- La classe Player vuole solo sapere se può piazzare una nave o se ha colpito qualcosa. Non vuole preoccuparsi di calcolare le coordinate o gestire l'array di bottoni
- Field offre metodi semplificati ad alto livello (canPlaceShip, placeShip...) , mentre la logica complessa è gestita da altre classi, in modo da mantenere un'interfaccia semplice

## STRATEGY PATTERN

- Gestisce la complessità delle regole di scommessa e il calcolo delle vincite
- Attraverso un interfaccia (GameRules) si definisce il comportamento generale della logica.
- Classi concrete implementano le differenze delle regole per determinare la vincita di una scommessa
- L'aggiunta di una nuova regola è indipendente dalla classe Roulette grazie all'uso dell'interfaccia.

## DATA ACCESS OBJECT PATTERN

- RouletteDB fa da intermediario tra l'applicazione e il database
- La classe Roulette invoca metodi astratti per garantire che le modifiche al database non influiscano sulla logica di gioco.



# GRAZIE PER L'ATTENZIONE

