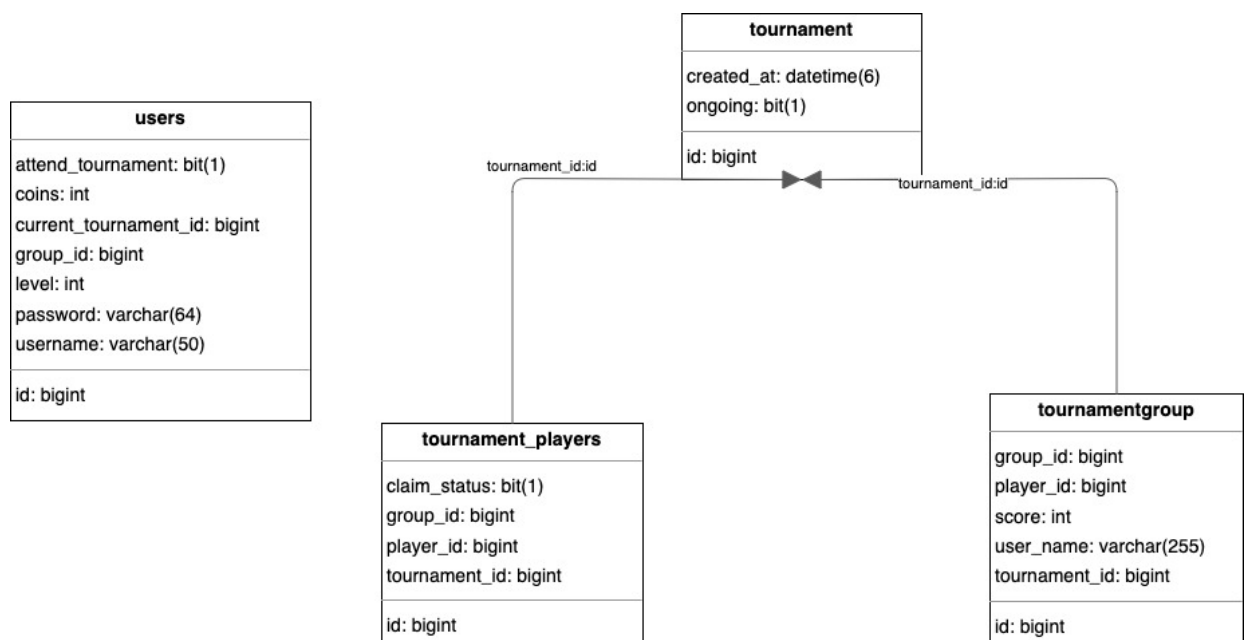# Design Documentation

Table of Contents

1. Introduction

1.1 Goal

The goal of our project is to develop a game API for a casual game. It is a case study for specifid tiem intervals.

2. Low-Level Design

2.1 Class Diagrams

There is one-to-one relationships between the tournament and tournament players and tournament and tournament group table.

2.2 Design Decisions

A tournament generates automatically with using java scheduling library.

2.2.1 User Side

- User creates firstly with request body username and password with request **users/createuser.** It will be created default as
1. coins -> 5000
2. level -> 1
3. attended_tournament -> false
4. current_tournament_id -> 0
5. group_id -> 0.

- User can play game without attending any tournament with request **updatelevel/{player_id}.**
- If user attend a tournament, it will be saved also into the player and group table of current tournament.
- If user already attened a tournament and it send request to "updatelevel", it will be find its group id and tournament id then updated its position in the own tournament group. Then level and coin are updated normally.

2.2.2 Tournament Side

- Tournament can be accessible just between the 00.00-20.00 hours. It can be sent request just between this time interval. Otherwise TournamentNotFound exception is thrown.
- If the player got the reward previous tournament and if it didn't claim the own reward before the enterTournament request, it cannot be added to new tournament.

- Player is assigned to tournameng group table with group id. Group id is calculated with remainder logic of division. If 20-100 level is divided 100, remainder equals the 0 and these level players assigned level % 100 + 1 group id. Others likewise..
- Until the tournament is over all changed datas are written to database and they read from always from database.
- When tournament is over claim status flags and reward coins updated. Program waits the 00.00.