1. Introduction

1.1 Project Scope and Objectives The goal of this project is to design and implement a full-stack gambling platform, with games such as roulette, and coin flip, with a focus on creating a secure and user-friendly experience. It will involve building both backend and frontend components using modern frameworks and tools, while integrating DevOps practices for continues integration and deployment.

The scope includes backend architechture, API development, frontend design, security measures, and cloud deployment. However, it excludes marketing strategies, and legal compliance beyond technical implementation. The objective is to deliver a reliable platform with efficient DevOps integration and an engaging user interface.

1.2 Problem Statement and Solution Overview

Problem Statement: In this project several challenges had to be overcome, these include:

- **Scalability:** Making it possible for several different users to play simultaneously, while maintaining uninterrupted gameplay.
- **Security:** Preventing exploitation of tokens, to prevent potential threats.
- **Maintainability:** Creating a system that can be updated and improvied without any significant overhead.
- **Deployment:** Developing a deployment process with frequent updates without downtime

Addressing these challenges is vital to delivering a reliable and engaging platform.

Solution Overview: The project addresses these challenges the following ways:

- A **React frontend** using **Vite** and **TypeScript** to create a user-friendly interface.
- A **Quarkus-based backend** written in Java to implement core functionalities such as game logic, authentication, and data management. **PostgreSQL** is used as the database for data storage.
- **DevOps practices**:
  - **Containerization** with Docker to provide environments that are consistent across development, testing, and production.
  - **Version control and automation** using GitHub and GitHub Actions for continuous integration and deployment.
- An emphasis on **secure authentication and authorization**, by using **JWT tokens** to manage user sessions effectively.

These approaches both addresses the technical challenges, and ensures that the platform is secure, and scalable.

1.3 Methodology This project follows a Kanban methodology using GitHub Issues and Projects to emphasize continuous improvement and flexibility in task management. This methodology is based on a "pull system," where work items (Issues) are pulled from a backlog into a steady flow of work, rather than being pushed through the process.

Kanban is a methodology that focuses on visualizing work, limiting work-in-progress, and maximizing efficiency. It uses a Kanban board (GitHub Projects Board) to represent the workflow of our project. For our board, we chose to go with the columns: Backlog, Ready, In Progress, In Review and Done. Each Task/Issue is then represented as a card that can move through each column, providing a clear visual representation of the project's status.

Another key feature of Kanban is the limit you set on the In Progress column. This helps the team focus on completing tasks before starting new ones, reducing multitasking and improving efficiency. The size of the In Progress column is usually set to the teams size plus one. This allows each team member to focus on one task while providing a little flexibility for handling bottlenecks. Since Kanban does not prescribe specific roles or iterations, it is highly adaptable and fits well into our team structure.

Integrating Kanban with DevOps enhances workflow efficiency and flexibility by using a pull system and visual task management. This aligns with DevOps principles like CI/CD, ensuring a seamless workflow. This approach not only improves task management but also supports continuous improvement and adaptability, key aspects of both Kanban and DevOps.