SRS DOCUMENT

Project: Python Casino Web Application (Blackjack)
Sprint Goal: Expand the MVP by improving UI clarity, data accuracy, and personalization to users while strengthening the wallet and funds functionality. In addition to the existing blackjack game, add a new slot machine game with persistent balance updates and incorporate preferred name display, player statistics, and polished money handling for a more complete casino experience.

Part 1 - Introduction

1.1 Purpose

The purpose of this project is to create a Python Casino Web Application that allows users to register, log in, and play the Blackjack and Slot Machine game through an interactive Flask-based interface. The system will manage user accounts, track balances, and store data in a JSON file to ensure persistence across all sections. The application also includes preferred names for user greetings and a basic colored UI to improve readability.

1.2 Scope

This software provides a complete, browser-based casino game for users. Users can

- Register and log in with a unique username and password
- Store a preferred name, which will be displayed on the home page and login
- See a grand total for money won, money lost, and the value along with username and current balance on a dashboard
- Deposit funds and track their balance
- Play a blackjack game against an automated dealer
- Play a simple slot machine game with three “reels” and symbol-based payouts
- Have all progress (balance, wins, losses) saved in a JSON file.

Part 2 - Overall Description:

2.1 User Characteristics

The target users are both professional and casual gamblers who want a simple and reliable way to check their game performance and balance through a virtual casino.

2.2 General Constraints

- The system must be implemented in Python using Flask
- All persistent data must be stored in a single JSON file (database.json)

- The app must run locally on a Flask server
- Each module (user.py, player.py, blackjack.py, main.py, slot.py) must remain modular and independent
- Passwords must remain private and not be displayed in the main text
- UI must include simple color styling to group content and improve readability
- The interface must show clear instructions and simple navigation
- Monetary values must display consistent decimal precision (two decimal places, to the cent) for balance and transaction views
- Error messages include a "Return to Home" button for invalid credentials during login
- Text content displays correct spellings

Part 3 - Specific Requirement

Feature 1: User Registration, Login, & Logout System

User story 1: As a forgetful gambler, I want a secure login method to store my information so that I will not lose track of my gambling history.

Acceptance criteria:

- When launching the app, users are prompted to create or enter a username and password, and an optional preferred name.
- If credentials do not match, an error message appears, and login is denied.
- On logout, the system saves player balances and game statistics (wins/losses) to the JSON file.

Feature 2: Casino Lobby & Dashboard Display

User story 2: As a professional and competitive gambler, I want to see a dashboard of my current balance so that I can track my progress

Acceptance criteria:

- The dashboard must display the gambler's username and current balance.
- The dashboard should have a simple color layout to see things more visually.
- Dashboard hides the initial user's balance from Homeview for privacy.
- The user must be able to navigate back to the Blackjack game or the deposit funds page.
- The display must automatically update when the balance changes.
- Routes /home and /start are displayed clearly and distinctly for a formalized UI layout.

Feature 3: Fund Deposits Tracker/Bank

User story 3: As a frequent gambler, I want to be able to track my wins (and losses) digitally and automatically so that I can eventually use this money for personal purchases.

Acceptance criteria:

- Users can fill out a deposit form and add more money to their balance.
- Deposits must reject negative or invalid amounts.
- Deposits over $1 billion are also automatically rejected to maintain realistic values.
- When a user wins, their winnings are added to the balance. And when they lose, the losses are subtracted.
- Balance is displayed and saved to a JSON file after each update.
- Balance and add-funds functionality are moved to a separate route from the home page.

Feature 4: Blackjack game

User story 4:
As an avid gambler, I want to be able to run the blackjack game smoothly so that I can have the best gambling experience.

Acceptance criteria:

- The system must include a playable blackjack game with standard rules and an automated dealer.
- Players must be prompted to enter a valid bet amount before starting each round.
- Wins, losses, and pushes must automatically update before starting each round.
- The total money won, total money lost, and net value of the user are displayed on the user dashboard.
- Game results and user progress (wins & losses) are updated and persisted in the JSON file.
- Monetary values are displayed with two decimal points (precise to the cent) for consistency and precision.

Feature 5: Slot Machine Game

User story 5:
As a casual gambler, I would enjoy playing a quick slot machine game where I can pull a lever and see if the three symbols align for a payout.

Acceptance criteria:

- The slot machine shows a bet input, a “Pull Lever” action, and three symbol positions.
- On lever pull, three symbols are chosen at random from a defined set.
- Payouts depend on symbol combinations
- A valid bet must be entered before each pull. The bet must be positive and not exceed the user’s balance.
- The round result and updated stats are added to the JSON file.
- The UI has a simple color layout, as the dashboard does.