Praeses Technical Sample

Task

You will build a Blackjack game implemented as a Ruby on Rails web application. This project is an opportunity to demonstrate the caliber of your work by showcasing your ability to design and structure a functioning web app in Rails. The focus should be on the application's architecture and game logic rather than on visual design. A straightforward interface is acceptable, but the app should clearly demonstrate how you organize game logic, manage state, and structure the project within Rails.

Evaluation

Adherence to Game Rules (High)

Like most card games, Blackjack is heavily based on rules. As part of this task, you will need to identify rules for the core game and implement those in your submission. Bicycle Cards has a good set of rules. But please consider betting, splitting, and multi-player non-essential features. They may be included if you wish, but they are less important than the core of the game. The most important factors here are: "dealing" cards in the correct order, identifying win conditions, handling player actions appropriately, handling the value of aces.

Overall Impression (Medium)

A holistic impression of the submission. This includes, but is not limited to, completeness, polish, and how well the solution demonstrates your approach to problem-solving and coding practices.

Program Structure and Design (Medium-High)

This covers how well designed the application is. Is the code broken up into multiple files, classes, functions, etc? Is each piece of the application focused and cohesive?

Documentation and Readability (Medium)

This focuses on how easy the submission is to read and understand. Ensure that variables and functions are appropriately named. Code comments are not required, but feel free to add if needed to enhance clarity.

README(Required)

A short README should be included as part of your submission (a simple README.md file in the repository will suffice). It should indicate any additional features that you have implemented beyond the core game rules. Additionally, it should include instructions on how to build and run your application.

UI / Presentation (Low-Med)

The user interface for your Blackjack web application does not need to be highly polished or visually complex. A simple, functional presentation is sufficient. The primary expectation is that the application runs within Ruby on Rails and allows a user to play the game through a web interface. Focus on clarity and usability rather than advanced styling. What matters most is that the UI demonstrates how you integrate the game logic into a Rails web app and manage state across user interactions.

Additional Features (Low)

Additional consideration will be given for any features that are implemented beyond the core game, but these are considered supplementary to the core requirements and are not, themselves, required to complete this task. Some examples include multiplayer, betting, and splitting. These are just a few examples of features that you *could* add.