

Praeses Technical Sample

Task

You will create a program that plays the card game Blackjack using web technologies with a foundation in Ruby on Rails. This is an opportunity for you to show us the caliber of work that you produce.

You may implement this as a console app or as a web app. A console app is acceptable and will allow you to demonstrate core programming and design skills. However, a well-executed web app will showcase broader capabilities in building user-facing applications, which is particularly relevant for candidates pursuing more senior-level positions.

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 focus here is on the user experience of the application. While a graphical interface is not required, a console-based experience should still feel intentional and usable. A well-designed web app will demonstrate higher-level skills—such as front-end development, integration with frameworks, and user experience considerations—and is a strong signal of readiness for more senior-level roles.

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.