

Blackjack core gameplay

Developers: Sleimon, George, Mark

Tester: Mark

Stats

Developers: Arya, Roopkiran, Sleimon

Tester: George

Chips and Virtual currency

Developers: Mark, Sleimon

Tester: Sleimon

User and Account Management

Developers: Arya, Roopkiran

Tester: George

Betting

Developer: Mark, Sleimon

Tester: Sleimon

Different Hands

Developers: George, Sleimon

Tester: Arya

Card and Deck

Developers: George, Sleimon

Tester: Mark

User Profile

Developers: Roopkiran

Tester: Arya

Fully functional server database

Developers: Arya

Tester: Roopkiran

Build

Developer: Arya

Tester: Roopkiran

Unfinished Tasks

Split

Main Menu


Multiplayer

Leaderboard

Friend System

Bugs

1. Bet bug: Whenever the bet button is hit, the chips text showing the amount of chips in the gui doesn't update. For example when the bet 50 button is hit, the chips text doesn't reflect that 50 chips have been removed from the player's chip total

 sleimon opened 2 hours ago · edited by sleimon Edits ▾ ...

Problem Report Number: 3

Reported By: Sleimon Naimi

Date Reported: March/24th/2025.

Program/Component Name:
Path: 2311BlackjackTeam4/src/main/java/com.blackjack/Models/Game.java.

Release Number: Released on March/7th/2025 at 1:58 P.M.

Version/Build Identifier: Git Hash: [82430e0](#)

Configurations:

Hardware:

- Device: IBUYPOWER Trace Mesh Gaming Desktop
- CPU: Intel(R) Core(TM) i7-14700F 2.10 GHz
- RAM: 32GB

Software:

- OS: Windows 11.
- Database: PostgreSQL on Supabase.
- Runtime Environment: Java JDK 21.

Report Type: Coding

Can Reproduce: Yes.

Severity: Low.

Priority: Low.

Problem Summary: Hitting the bet button doesn't immediately update the chips display


Key Words: Bet, Button, Chips, Display

Problem description and how to reproduce it:
In any playthrough, hit any of the bet buttons and it won't immediately update the chips display which should subtract the bet you made from the chips you have to signify you made a bet.

To reproduce it, just log in, start playing, hit any of the bet buttons (bet50, bet100, bet all) and then observe the chips display in the top left.

Suggested Fix: Taking a look at the updateStats() method

2. All the winning chips methods do not increase chip count. The purpose of blackjack is outsmarting or being luckier than the dealer and winning chips but whenever you win, you don't get the respective payout of chips

 sleimon opened 1 minute ago

Problem Report Number: 4

Reported By: Sleimon Naimi

Date Reported: March/24th/2025.

Program/Component Name:
Path: 2311BlackjackTeam4/src/main/java/com.blackjack/Models/Game.java.

Release Number: Released on March/7th/2025 at 1:58 P.M.

Version/Build Identifier: Git Hash: [82430e0](#)

Configurations:

Hardware:

- Device: IBUYPOWER Trace Mesh Gaming Desktop
- CPU: Intel(R) Core(TM) i7-14700F 2.10 GHz
- RAM: 32GB

Software:

- OS: Windows 11.
- Database: PostgreSQL on Supabase.
- Runtime Environment: Java JDK 21.

Report Type: Coding

Can Reproduce: Yes.

Severity: High

Priority: High

Problem Summary: Winning in any of the 5 ways of blackjack doesn't result in an increase of chips for the player

Key Words: Bet, Win, Increase, Chips

Problem description and how to reproduce it:
In any playthrough, try to win a hand in any of these 5 ways:

1. Winning a hand normally by getting 21, having a higher hand value than the dealer, or the dealer busting
2. Drawing a blackjack when the dealer doesn't have one, known as an instant blackjack
3. Hitting the double down button and winning
4. Predicting the dealer has 21 and correctly choosing insurance resulting in a win for you
5. A tie with the dealer

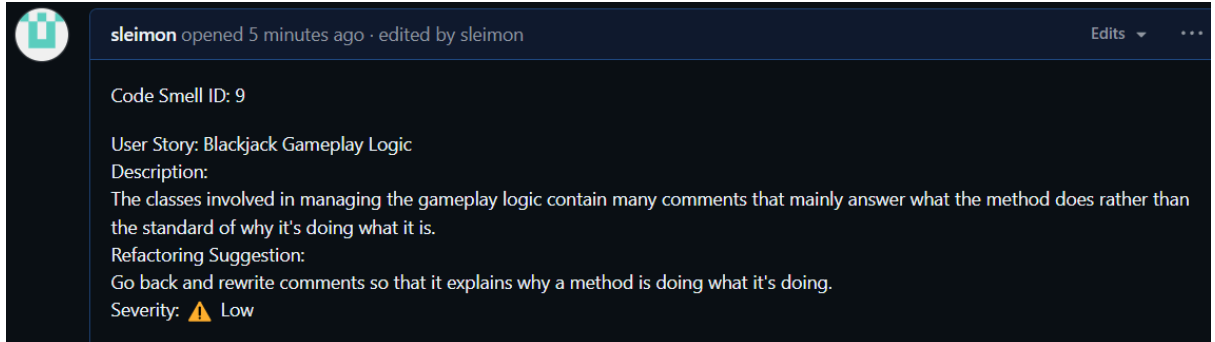
When any of these happen, you can observe that your chip count never goes up

Suggested Fix: Taking a look at the logic methods in game.java. Specifically `initialBlackjack()`, `dealerInitialBlackjack()`, `playerTurn()`, `checkPlayer21()`, `dealerTurn()`, `non21win()`, and `insurance()` methods

Code Review

Reported Smells:

1. **[Comments]** The comments in most of the classes don't answer why a method/class is doing what it's doing but rather how it works. This is not standard for proper coding.



sleimon opened 5 minutes ago · edited by sleimon

Code Smell ID: 9

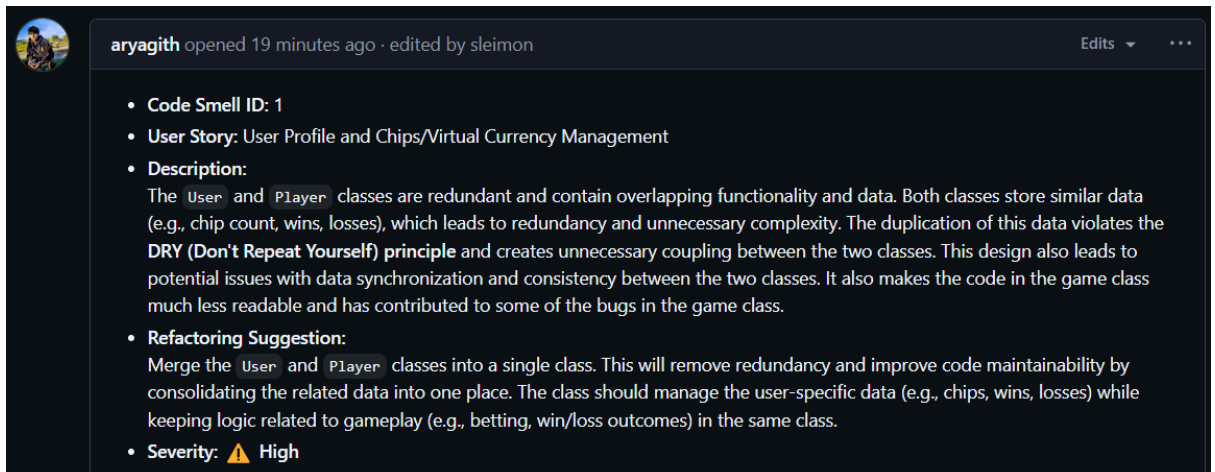
User Story: Blackjack Gameplay Logic

Description:
The classes involved in managing the gameplay logic contain many comments that mainly answer what the method does rather than the standard of why it's doing what it is.

Refactoring Suggestion:
Go back and rewrite comments so that it explains why a method is doing what it's doing.

Severity: ⚠ Low

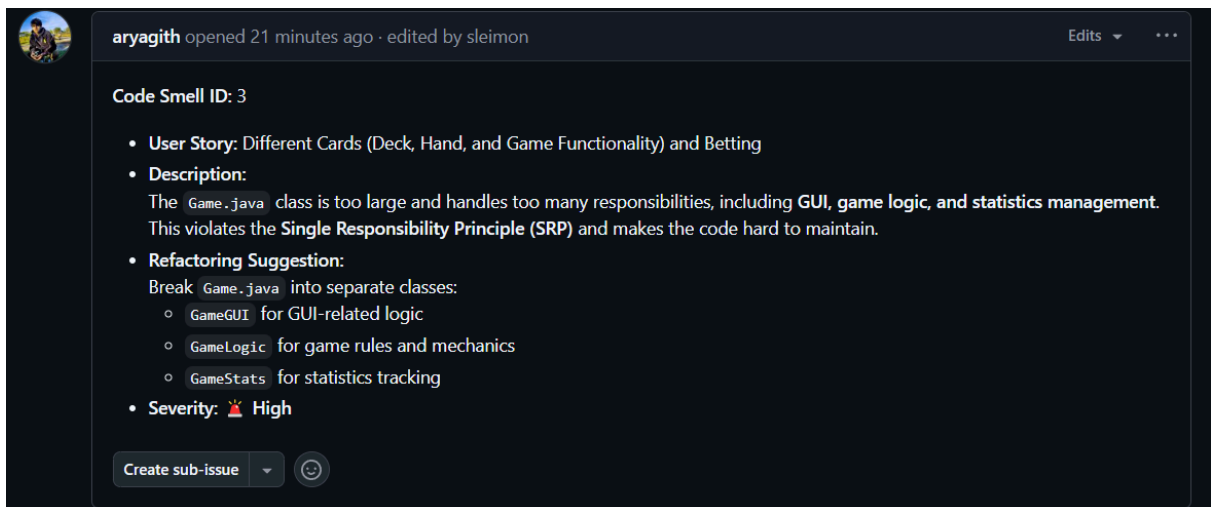
2. **[Duplicate Code]** The User class is completely redundant. It's duplicate code + a dead class frankly. Most of it has been duplicated from the player class and its use in the game class has made the code much less readable and has contributed to the chip currency bugs and the betting bugs



aryagith opened 19 minutes ago · edited by sleimon

- Code Smell ID: 1
- User Story: User Profile and Chips/Virtual Currency Management
- Description:
The `User` and `Player` classes are redundant and contain overlapping functionality and data. Both classes store similar data (e.g., chip count, wins, losses), which leads to redundancy and unnecessary complexity. The duplication of this data violates the **DRY (Don't Repeat Yourself) principle** and creates unnecessary coupling between the two classes. This design also leads to potential issues with data synchronization and consistency between the two classes. It also makes the code in the game class much less readable and has contributed to some of the bugs in the game class.
- Refactoring Suggestion:
Merge the `User` and `Player` classes into a single class. This will remove redundancy and improve code maintainability by consolidating the related data into one place. The class should manage the user-specific data (e.g., chips, wins, losses) while keeping logic related to gameplay (e.g., betting, win/loss outcomes) in the same class.
- Severity: ⚠ High

3. **[Class Too Large]** The game class is too large. It's 5-10x bigger than any other class. It stores all the logic and gui in one class which is bad code design.




aryagith opened 21 minutes ago · edited by sleimon

Code Smell ID: 3

- User Story: Different Cards (Deck, Hand, and Game Functionality) and Betting
- Description:
The `Game.java` class is too large and handles too many responsibilities, including **GUI, game logic, and statistics management**. This violates the **Single Responsibility Principle (SRP)** and makes the code hard to maintain.
- Refactoring Suggestion:
Break `Game.java` into separate classes:
 - `GameGUI` for GUI-related logic
 - `GameLogic` for game rules and mechanics
 - `GameStats` for statistics tracking
- Severity: 🚨 High

Create sub-issue ▾ 😊

4. **[Method Too Large]** The setupGUI method in the game class is too large. It encompasses nearly 1/5th of the class.



aryagith opened 21 minutes ago · edited by sleimon


Edits ▾ ⋮

Code Smell ID: 4

- **User Story:** Different Cards (Deck, Hand, and Game Functionality) and Chips/Virtual Currency Management
- **Description:**
The `setupGUI()` method in `Game.java` is too long and handles too many responsibilities, including button creation, event handling, and game statistics display.
- **Refactoring Suggestion:**
Split `setupGUI()` into smaller methods, each with a single responsibility, such as:
 - `createButtons()`
 - `addEventListeners()`
 - `displayGameStats()`
- **Severity:** ⚠ Low

Create sub-issue ▾ 😊

5. **[Dead Code]** There's some dead code in the game class either in the form of unused code or commented out code from a previous version.



aryagith opened 23 minutes ago · edited by sleimon


Edits ▾ ⋮

Code Smell ID: 5

- **User Story:** User Profile , Betting, Chips/Virtual Currency Management
- **Description:**
`Game.java` contains large segments of commented-out code that are no longer in use, cluttering the codebase and reducing readability.
- **Refactoring Suggestion:**
Remove all unused and commented-out code to enhance clarity and maintainability.
- **Severity:** ⚠ Low

Create sub-issue ▾ 😊

6. **[Duplicate Code]** There is some duplicate code being used across the game class



aryagith opened 19 minutes ago · edited by sleimon

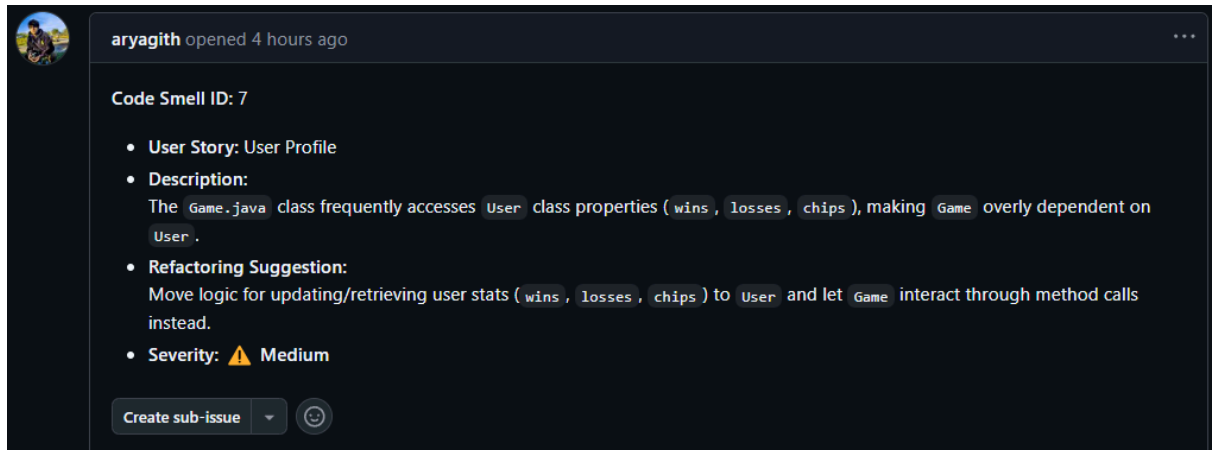
Edits ▾ ⋮

Code Smell ID: 6

- **User Story:** User Profile , Betting, Statistics
- **Description:**
The logic for updating user statistics (`wins` , `losses` , `pushes`) is repeated across multiple methods in `Game.java` , violating the DRY principle.
- **Refactoring Suggestion:**
Create a single method to handle all stat updates and call it wherever needed.
- **Severity:** ⚠ Medium

Create sub-issue ▾ 😊

7. **[Feature Envy]** The game class is overly dependant on the user class that we plan to get rid of anyways



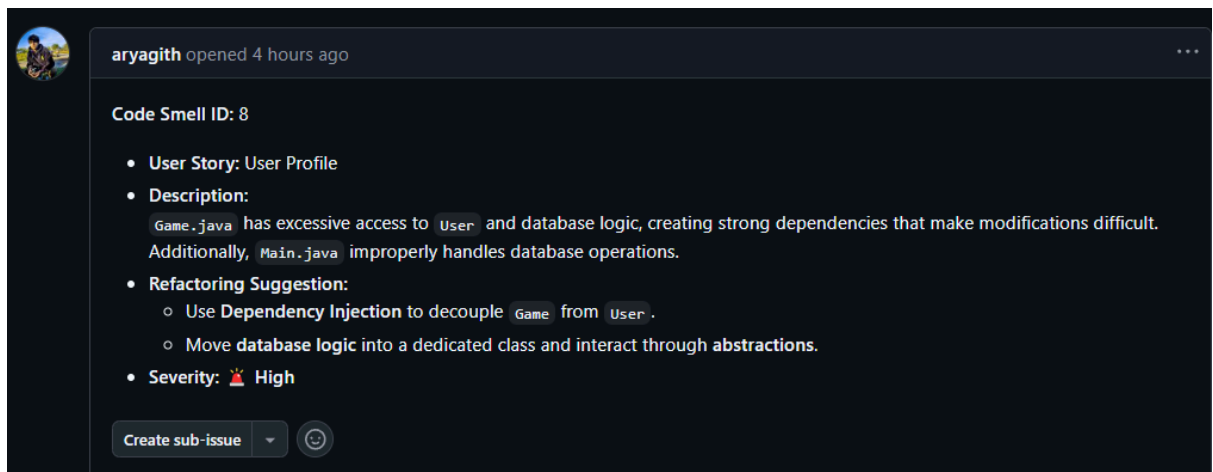
aryagith opened 4 hours ago

Code Smell ID: 7

- **User Story:** User Profile
- **Description:**
The `Game.java` class frequently accesses `User` class properties (`wins`, `losses`, `chips`), making `Game` overly dependent on `User`.
- **Refactoring Suggestion:**
Move logic for updating/retrieving user stats (`wins`, `losses`, `chips`) to `User` and let `Game` interact through method calls instead.
- **Severity:** ⚠ Medium

Create sub-issue

8. **[Inappropriate Intimacy]** The game class has way too much access to the user objects and the database making modifications very difficult



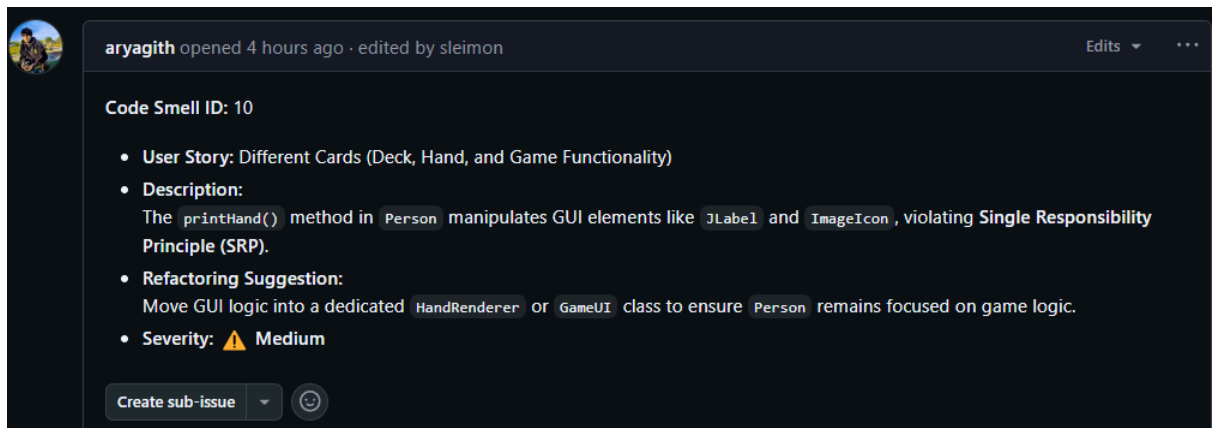
aryagith opened 4 hours ago

Code Smell ID: 8

- **User Story:** User Profile
- **Description:**
`Game.java` has excessive access to `User` and database logic, creating strong dependencies that make modifications difficult. Additionally, `Main.java` improperly handles database operations.
- **Refactoring Suggestion:**
 - Use **Dependency Injection** to decouple `Game` from `User`.
 - Move **database logic** into a dedicated class and interact through **abstractions**.
- **Severity:** 🔥 High

Create sub-issue

9. **[Feature Envy]** The printHand() method manipulates GUI elements when it shouldn't




aryagith opened 4 hours ago · edited by sleimon

Code Smell ID: 10

- **User Story:** Different Cards (Deck, Hand, and Game Functionality)
- **Description:**
The `printHand()` method in `Person` manipulates GUI elements like `JLabel` and `ImageIcon`, violating **Single Responsibility Principle (SRP)**.
- **Refactoring Suggestion:**
Move GUI logic into a dedicated `HandRenderer` or `GameUI` class to ensure `Person` remains focused on game logic.
- **Severity:** ⚠ Medium

Create sub-issue

10. **[Inappropriate Intimacy]** the `hit()` method in the `person` class is coded in a way that it reduces its ability to be flexible and be tested



aryagith opened 4 hours ago · edited by sleimon

Edits ▾ ⋮

Code Smell ID: 11

- **User Story:** Different Cards (Deck, Hand, and Game Functionality)
- **Description:**
The `hit()` method in `Person` class tightly couples it to `deckOfCards` and `discard`, reducing flexibility and testability.
- **Refactoring Suggestion:**
Delegate deck interactions to a `GameLogic` or `Dealer` class to separate concerns and improve maintainability.
- **Severity:** ⚠ Medium

Create sub-issue ▾ 😊