Introduction

The card game 'blackjack' has been played for many years and is quite well-liked. The goal of the game is to have a hand with a value of 21 or as close to 21 as possible without going over, it is also known as "21." A player and a dealer compete in the game by trying to outsmart the dealer's hand. Blackjack has changed throughout the years, and it is now one of the most popular games in both traditional and online casinos. We will create a blackjack game for use on a computer or mobile device as part of this project. The goal of this project is to develop a fully functional blackjack game that enables users to enjoy the thrill and excitement of the game in a virtual setting. The ability to place bets, get cards, hit, stand, and split will all be available in the game, along with all other conventional blackjack capabilities.In this project, we'll make a user-friendly and visually appealing interface for the game using contemporary programming languages and tools. We will use object-oriented programming techniques to design the blackjack game project, resulting in a modular and manageable codebase. The user interface will be made using a combination of HTML, CSS, and the server-side code will be made using SpringBoot. We'll make sure the game is performance-optimised and runs without a hitch on a variety of hardware setups.The essential components and specifications of the blackjack game project, as well as the development procedure and the tools and technologies that will be utilised to construct the game, will all be covered in the parts that follow.

The following essential characteristics and specifications will apply to the blackjack game project:

1.) User Interface: The game will have an intuitive user interface that makes it simple for players to gamble, receive cards, and make choices while playing. With regard to screen sizes and resolutions, the interface will be created using HTML and CSS.

2.) Play: The game will be played according to the conventional rules of blackjack, with the goal being for the player to have a hand value of 21 or as near to 21 as they can go without going over in order to beat the dealer's hand. The game will conclude when the player or dealer busts, whether the player wins or loses the hand, and the player will have the option to hit, stand, split, and double down.

3.) Betting : The game will contain a betting structure that enables players to wager on each hand. The user will be able to select how much they want to wager, and the game will record their wins and losses.

4.) Performance: The game will be speed-optimised so that it can run smoothly on a variety of hardware setups. The game will be built to respond rapidly to user interaction and to reduce the amount of data that needs to be sent between the server and the client.

The BlackJack project development was contributed by Stephen Dias, Aditya Limbekar and Atharva Salehittal of Spring 2023 batch.The project was started on 3rd April 2023 .The project's development will use an incremental and iterative approach. Starting with a simple game, we will progressively add extra features.

Game Rules

Aim : The goal of the game is to beat the dealer by having a hand that is worth more points than the dealer's hand, without going over 21.

Cards Value:The value of a hand is determined by the sum of the point values of the individual cards. Number cards (2-10) are worth their face value, face cards (J, Q, K) are worth 10 points each, and Aces can be worth 1 or 11 points.

GamePlay :

1. A wager is made by each player prior to the game beginning.
2. Each participant, including themselves, receives two cards from the dealer. The dealer's cards are dealt face-up and face-down, respectively.
3. The player has the option to stand (keep their current hand) or hit (take another card).
4. The player loses (goes broke) if their hand totals more than 21 points.
5. The dealer then displays their face-down card once each player has finished taking their turn.
6. Until their hand value is 17 or above, the dealer must hit.
7. The dealer loses (busts) if their hand totals more over 21 points.
8. The player wins if their hand is closer to 21 than the dealer's. The dealer wins if their hand is closer to 21 than the player's hand. It is a tie (push) if the hands have the same value.
9. Payouts are given in accordance with the table's betting guidelines.

Note :

Claim : The Blackjack project was created and put into place to guard against SQL injection flaws.

Justification :

1. PreparedStatement : The project handles all database queries using prepared statements, which guards against SQL injection threats by automatically escaping special characters in user input.
2. ParametrizedQueries : The project handles all dynamic database queries using parameterized queries, which enables the database server to correctly parse and carry out the query while also guarding against SQL injection attacks.
3. Validation of Input: The project verifies all user input to make sure that it contains only legal characters. Incorrect characters or commands will be ignored by the input validation process, preventing attackers from inserting harmful SQL code into the system.
4. Error Handling: To manage database errors that may arise while running queries, the project makes use of the proper error handling tools. By shielding attackers from detailed error messages that would divulge private information about the database or its structure, this helps to thwart SQL injection attempts.
5. Penetration testing: To find and fix any potential SQL injection vulnerabilities that might be present in the system, the project is subjected to routine penetration testing.

Conclusion:

On the basis of the above explanation, we can state with confidence that the Blackjack project was developed and put into use to guard against SQL injection vulnerabilities. In order to defend against SQL injection attacks, the project makes use of prepared statements, parameterized queries, input validation, and suitable error handling techniques. The project also goes through regular penetration testing to make sure it is protected against new threats. Players may relax and enjoy the game knowing that SQL injection assaults won't compromise their important data.

Claim : The blackjack project was developed and put into place to guard against cross-site scripting flaws, according to the claim.

Justification :

1. Validation of Input: The project verifies all user input to make sure that it contains only legal characters. Incorrect characters or commands will be ignored by the input validation process, preventing attackers from inserting malicious scripts into the system.
2. Encoding of Output : Before showing user-generated content to the user, the project applies the proper output encoding techniques to encode it. This makes sure that any potentially harmful scripts are neutralised and are unable to be run by the user's browser.
3. Secure Communication: To guarantee that all data communicated between the server and the user's browser is encrypted and secure, the project makes use of secure communication protocols (such as HTTPS). As a result, attackers are unable to intercept or alter data in transit, including scripts that might be dangerous.
4. Penetration testing: To find and fix any potential XSS vulnerabilities that might be present in the system, the project regularly undergoes penetration testing.

Conclusion : On the basis of the above explanation, we can state with confidence that the Blackjack project was created and put into place to guard against cross-site scripting vulnerabilities. To defend against XSS attacks, the project makes use of input validation, output encoding, secure communication protocols and routine penetration testing. Players may relax and enjoy the game knowing that XSS assaults won't compromise their sensitive data.

Claim : The blackjack project was developed and put into action to guard against vulnerabilities caused by hard-coded credentials.

Justification :

1. Password Policy: The project imposes a strong password policy that calls for users to generate a distinctive password that complies with certain complexity specifications, including a minimum length and the use of special characters, numerals, and upper- and lowercase letters. This rule aids in preventing users from selecting passwords that are weak or simple to decipher and could be used by attackers.
2. Passwords Stored in Encrypted Format: The project employs a safe hashing algorithm to save all user passwords in encrypted format. By doing this, it is ensured that even if an attacker has access to the database that stores user credentials, they will not be able to see the actual passwords.
3. Secure Coding Techniques: The project follows secure coding techniques, such as not hard-coding any passwords or other sensitive information into the code or configuration files. All sensitive data is safely stored outside of the code and may only be accessed using secure ways, including database connection strings, API keys, and other credentials.
4. Penetration testing: The project goes through regular penetration testing to find and fix any potential vulnerabilities caused by hard-coded credentials that might exist in the system.
5. Access Controls: To restrict access to sensitive portions of the system, the project implements stringent access controls and role-based permissions. This makes it harder for unauthorised individuals to access or change important information, such as user credentials.

Conclusion : On the basis of the above justification, we can state with confidence that the Blackjack project has been developed and put into use to guard against vulnerabilities related to hard-coded credentials. The project enacts access controls and maintains a robust password policy to guard against credential-based attacks. It also follows secure coding guidelines and stores passwords securely using encryption. Knowing that their sensitive information is shielded from misuse or illegal access, players may enjoy the game with confidence.

Claim : The Uncontrolled Resource Consumption vulnerabilities have been addressed by the design and implementation of the Blackjack project.

Justification :

1. Resource Caps: The project establishes sensible resource caps for several system elements like memory utilisation, CPU usage, and network bandwidth. By restricting the resources that a single user or request can use, this helps to prevent resource exhaustion attacks.
2. Validation of Input: To make sure that only legitimate input is allowed, the project validates all user input. This stops attackers from sending requests that need too many resources, like those that have huge payloads or demand computationally expensive actions.
3. Penetration testing : Regular penetration testing is conducted on the project to find and fix any potential resource exhaustion vulnerabilities that might be present.

Conclusion : With the above explanation, we can state with confidence that the Blackjack project was developed and put into use to guard against Uncontrolled Resource Consumption vulnerabilities. To defend against resource depletion assaults, the project establishes appropriate resource constraints, performs input validation and undergoes routine penetration testing. Because the system is guarded against resource exhaustion attacks, players may relax and enjoy the game.

Claim : Blackjack was created and put into place to guard against improper authentication flaws.

Justification :

1. Strong Password Policy: The project imposes a strong password policy that calls for users to develop a distinct password that complies with certain complexity specifications, including a minimum length and the use of special characters, numerals, and upper- and lowercase letters. This rule aids in preventing users from selecting passwords that are weak or simple to decipher and could be used by attackers.
2. Encryption: The project encrypted sensitive data in transit and at rest, including user passwords and session tokens. This stops attackers from intercepting or stealing private data that may be used to enter user accounts without authorization.
3. Penetration testing: To find and fix any possible Improper Authentication vulnerabilities that might exist in the system, the project is regularly subjected to penetration testing.

Conclusion : On the basis of the above justification, we can state with confidence that the Blackjack project was created and put into place to guard against Improper Authentication vulnerabilities. To defend against authentication-based attacks, the project enforces a strict password policy, encrypted sensitive data, and is subjected to routine penetration testing. As their user accounts are secured from illegal access, players can enjoy the game with confidence.

Claim : Blackjack was developed and put into use to guard against Command Injection flaws.

Justification :

1. Input Validation: To make sure that only expected and secure characters are received, the project thoroughly validates all user input. This stops hackers from using user input to introduce harmful commands into the system.
2. Parameterized Queries : The project makes use of parameterized queries to guard against SQL injection attacks, which have the potential to run arbitrary instructions. By doing this, you can be guaranteed that any user input is correctly escaped and sanitised before being utilised in commands.
3. Principle of Least Privilege : The project adheres to the notion of least privilege by making sure that all commands run by the system are carried out with the least amount of privilege required to complete the task. This stops attackers from using command injection attacks to access private system resources.
4. Penetration testing: To find and fix any possible Command Injection vulnerabilities that might exist in the system, the project is regularly subjected to penetration testing.

Conclusion : On the basis of the above explanation, we can state with confidence that the Blackjack project was developed and put into use to guard against Command Injection vulnerabilities. The project employs parameterized queries, adheres to the idea of least privilege, thoroughly validates, sanitises, and encodes input, adheres to the principle of least privilege, and regularly undergoes penetration testing to guard against command injection attacks. Knowing that the system is secure from command injection attacks allows players to relax and enjoy the game.