Wilhelm Travers

Username Checker

This project is a Username Checker, which allows users to register a username and check the availability of a username. It aims to provide a simple and efficient solution for managing and validating usernames. By utilizing the username checker, users can avoid duplications and ensure the uniqueness of their chosen usernames. Additionally, instant feedback on username availability helps users make informed decisions and choose available usernames promptly.

Data Structures And Algorithms:

The username checker project utilizes the following data structures and algorithms:

- **ArrayList:** The project uses an "ArrayList" data structure to store the registered usernames. The "ArrayList" provides dynamic resizing and efficient element access, enabling efficient addition and retrieval of usernames.
- **Linear Search:** To check the availability of a username, the project employs the linear search algorithm. It iterates through the list of registered usernames, comparing each username with the provided input. This approach offers simplicity and effectiveness for small-scale username management.

Expected Outcomes:

The project aims to achieve the following outcomes:

- **Username Registration:** Users can register their desired usernames through the username checker program, ensuring that each username is unique within the system.
- **Username Availability Check:** The username checker allows users to quickly check the availability of a username. By entering a username, users receive instant feedback indicating whether the username is already taken or available for registration.

- **Efficient User Interaction:** The username checker provides a user-friendly interface through the Command Prompt, enabling users to interact with the program seamlessly. The system prompts users for input, validates it, and delivers clear messages about the status of the requested username.

Object-Oriented Design:

The username checker project adheres to an object-oriented design paradigm, employing classes and objects to structure and organize the codebase.

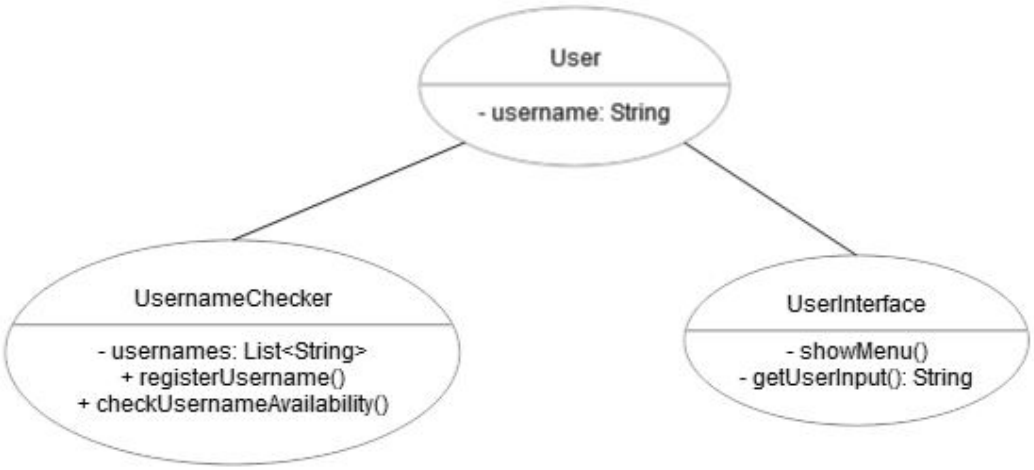
Objects/Entities:

- **User:** The "User" object represents an individual user interacting with the username checker. It encapsulates user-specific information and behaviors.
- **UsernameChecker:** The "UsernameChecker" object acts as the main class of the program. It orchestrates the functionality of the username checker, coordinating user interactions, username registration, and availability checks.

Classes:

- **UsernameChecker:** The "UsernameChecker" class contains the primary logic of the program. It includes methods for registering a username, checking username availability, and displaying the user interface. This class interacts with other objects and manages the flow of the username checker functionality.
- **ArrayList:** The "ArrayList" class is utilized to store the registered usernames in memory. It provides dynamic resizing and an efficient mechanism for adding and retrieving usernames.
- **Scanner:** The "Scanner" class facilitates user input reading from the Command Prompt. It allows the username checker to gather user inputs and process them accordingly.

Class Diagram:



User: Represents a user who interacts with the username checker. It has a private attribute username to store the user's chosen username.

UsernameChecker: The main class of the project. It manages the username registration and availability check functionalities. It contains a private attribute usernames, which is a list of strings storing the registered usernames. It also includes methods registerUsername() and checkUsernameAvailability() for registering usernames and checking their availability, respectively.

UserInterface: Represents the user interface of the username checker. It provides methods for displaying menus and getting user input. The showMenu() method displays the main menu to the user, and the getUserInput() method retrieves user input.

Additional Documentation:

How to run the Username Checker program:

- **Java Source Code Compilation:** Within Command Prompt, navigate to the directory with the "UsernameChecker.java" file and enter "javac UsernameChecker.java".
- **Program Execution:** Enter "java UsernameChecker" in Command Prompt.

Program Source Code:

Below is the source code for the Username Checker program, including comments providing brief explanations for each significant section of code. These comments aim to clarify the purpose and functionality of the respective code blocks, making it easier to understand the

logic of the Username Checker program:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class UsernameChecker {
    private static List<String> usernames; // List to store registered usernames

    public static void main(String[] args) {
        usernames = new ArrayList<>(); // Initialize the list of usernames
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("----- Username Checker -----");
            System.out.println("1. Register Username");
            System.out.println("2. Check Username Availability");
            System.out.println("3. Exit");
            System.out.print("Enter your choice: ");

            int choice = scanner.nextInt(); // Read user's choice
            scanner.nextLine(); // Consume the newline character

            switch (choice) {
                case 1:
```

```
                registerUsername(scanner); // Call method to register a username
                break;
            case 2:
                checkUsernameAvailability(scanner); // Call method to check username availability
                break;
            case 3:
                scanner.close();
                System.exit(0); // Exit the program
                break;
            default:
                System.out.println("Invalid choice! Please try again.");
        }
    }
}

private static void registerUsername(Scanner scanner) {
    System.out.print("Enter username: ");
    String username = scanner.nextLine(); // Read the username input

    if (usernames.contains(username)) {
        System.out.println("Username already taken. Please choose a different username.");
    } else {
        usernames.add(username); // Add the username to the list
```

```
                System.out.println("Username registered successfully.");
            }
        }

        private static void checkUsernameAvailability(Scanner scanner) {
            System.out.print("Enter username: ");
            String username = scanner.nextLine(); // Read the username input

            if (usernames.contains(username)) {
                System.out.println("Username is already taken.");
            } else {
                System.out.println("Username is available.");
            }
        }
}
```