American International University-Bangladesh (AIUB)

Department of Computer Science and Engineering

Faculty of Science &Technology (FST)

Fall 23-24

CSC 01488 SOFTWARE ENGINEERING

Section: A

Group NO: 11

Project Title: Missing Vowels

Submitted by:

Name	ID
1. MD SARAFAT ALI ADIR	20-41926-1
2. HASIN AABRAR KHAN	21-45297-2
3. MD TANVIR BIN ZOHA	21-45307-2

Submitted to:

DR. ABDUS SALAM
Assistant Professor, Computer Science

Project Description:

The Missing Vowels Game is a console-based word game that challenges users to guess words with missing vowels within a limited time of 2 minutes. The game includes features such as user registration and login, a timer-controlled gameplay mechanism, a scoring system, an admin panel with user management capabilities, and a leaderboard to track user performance. The project demonstrates modular programming, user authentication, real-time gameplay with a countdown timer, and persistent user score tracking.

Features Implemented:

User Registration

The user registration feature allows new users to sign up for the system by providing a unique username, full name, phone, email & password . If the username already exists, the user is notified. The implementation uses a simple text file **(user_data.txt)** for storing user information.

The **register_user()** function collects input for the username and password. It checks for duplicates, saves the new user data to a file, and initializes their score in the scores file.

```
def register_user():
    user_id = len(user_data) + 1
    username = input("Enter a new username: ")
    if username in user_data:
        print("Username already exists! Please try logging in.")
        return None
    password = input("Enter a new password: ")
    full_name = input("Enter your full name: ")
    email = input("Enter your email: ")
    phone = input("Enter your phone number: ")

    user_data[username] = {
        "id": user_id,
        "password": password,
        "full_name": full_name,
        "email": email,
        "phone": phone,
    }
    scores[username] = 0
    save_data(USER_DATA_FILE, user_data)
    save_data(SCORES_FILE, scores)
    print("Registration successful! Please log in.")
```

Login

The login system allows existing users to authenticate themselves using their credentials. Only users with valid credentials are allowed access to the game.

The **login_user()** function compares the entered username and password with data stored in **user_data.txt**. If valid, the user is granted access; otherwise, they are notified of invalid credentials.

```
def login_user():
    username = input("Enter username: ")
    password = input("Enter password: ")
    if username in user_data:
        if user_data[username]["password"] == password:
            print(f"Welcome, {username}!")
            return username
        else:
            print("Invalid password! Please try again.")
    else:
        print("Username not found! Please register first.")
    return None
```

Missing Vowels Game

The Missing Vowels Game challenges users to guess scrambled words with missing vowels. Players must guess as many correct words as possible within a 2-minute time limit. The game ends automatically after the timer runs out.

The **play_game()** function generates random words from a predefined dictionary **WORDS_WITHOUT_VOWELS**. It uses **time.time()** to monitor elapsed time and ends the game if 2 minutes have passed.

```python
def play_game(username):
    print("\nWelcome to the Missing Vowels Game!")
    words = random.sample(list(WORDS_WITHOUT_VOWELS.keys()), 5)
    correct_guesses = 0
    start_time = time.time()

    for word in words:
        elapsed_time = time.time() - start_time
        remaining_time = 120 - elapsed_time
        if remaining_time <= 0:
            print("\nTime's up! GAME OVER.")
            break

        print(f"\nTime remaining: {int(remaining_time)} seconds")
        scrambled = WORDS_WITHOUT_VOWELS[word]
        print(f"Word with missing vowels: {scrambled}")
        guess = input("Guess the word: ").strip().lower()

        if guess == word:
            print("Correct!")
            correct_guesses += 1
        else:
            print(f"Incorrect! The correct word was: {word}")

    scores[username] += correct_guesses
    save_data(SCORES_FILE, scores)
    print(f"\nYour total score: {scores[username]}\n")
```

Timer

A 2-minute timer is implemented during the gameplay. The timer monitors elapsed time in real time, and if 2 minutes are exceeded, the game terminates automatically.

The game uses **time.time()** to track elapsed time against a 120-second limit. If the elapsed time exceeds this limit, the game stops.

```python
for word in words:
    elapsed_time = time.time() - start_time
    remaining_time = 120 - elapsed_time
    if remaining_time <= 0:
        print("\nTime's up! GAME OVER.")
        break
```

Admin Panel

The admin panel allows:

- View the leaderboard (user scores).
- View All User Information
- Edit user Information.

- Logout

The **admin_operations()** function provides options to view the leaderboard, view & edit user information. Admin credentials are hardcoded as admin/admin.

```
def admin_operations():
    print("\nAdmin Panel")
    while True:
        print(
            "1. View All Users Information\n2. View Leaderboard\n3. Edit User Information\n4. Logout"
        )
        choice = input("Select an option: ")
        if choice == "1":
            print("\nAll Users Information:")
            for username, details in user_data.items():
                print(
                    f"ID: {details['id']}, Username: {username}, Full Name: {details['full_name']}, Email: {details['email']
                )
        elif choice == "2":
            print("\nLeaderboard:")
            sorted_scores = sorted(scores.items(), key=lambda x: x[1], reverse=True)
            for user, score in sorted_scores:
                print(f"Username: {user}, Score: {score}")
        elif choice == "3":
            user_id_to_edit = input("Enter the user ID to edit: ")
            for username, details in user_data.items():
                if str(details["id"]) == user_id_to_edit:
                    full_name = (
                        input(
                            f"Enter new full name (current: {details['full_name']}): "
                        )
                        or details["full_name"]
                    )
                    email = (
                        input(f"Enter new email (current: {details['email']}): ")
                        or details["email"]
                    )
                    phone = (
                        input(f"Enter new phone (current: {details['phone']}): ")
                        or details["phone"]
                    )

                    user_data[username]["full_name"] = full_name
                    user_data[username]["email"] = email
                    user_data[username]["phone"] = phone

                    save_data(USER_DATA_FILE, user_data)
                    print("User information updated successfully.")
                    break
            else:
                print("User ID not found!")
        elif choice == "4":
            print("Logging out from Admin Panel...\n")
            break
        else:
            print("Invalid choice. Try again.")
```

Leaderboard

The leaderboard displays all user scores in descending order. This functionality resides in the admin panel and dynamically fetches scores from the **scores.txt** file. Scores are loaded, sorted in descending order, and displayed to the admin using **sorted ()**.

```python
if choice == "1":
    print("\nLeaderboard:")
    sorted_scores = sorted(scores.items(), key=lambda x: x[1], reverse=True)
    for user, score in sorted_scores:
        print(f"Username: {user}, Score: {score}")
```