**Unit Tests**

**1 – Two users with the same password should have their passwords hash to different values**

**Description:** This is testing the salting of the hashes. If the salting is done correctly, the identical passwords should hash to separate values. If the passwords hashed to the same value, an adversary would be able to use the information to perform attacks on the database. For example, the hacker could hash the password “cat” and instantly find all the usernames with that password if I did not use a salt correctly.

**Input:**

**1)** "python .\Enroll.py testUser1 testPassword"

**2)** "python .\Enroll.py testUser2 testPassword"

**Expected Outputs:**

**1)** Accepted

**2)** Accepted

**3)** Unencrypted database file should have entries with usernames "testUser1 " and "testUser2 " that each have unique hashed passwords.

**2 – Password of only numbers should return “Rejected”**

**Description:** This tests to see if the system rejects a type of “weak” password. Number only passwords reduces the amount of space that an adversary needs to consider when brute forcing a password.

**Input:**

**1)** "python .\Enroll.py testUser 123123123"

**Expected Outputs:**

**1)** “Rejected” with exit code -1

**3 – Password that is a word should return “Rejected”**

**Description:** This tests to see if the system rejects a type of “weak” password. A password that is only a single word reduces the amount of space that an adversary needs to consider when brute forcing a password. Single word passwords are also commonly found in the top 100 password list. For example, words like “cat”, “dog”, and “password” are common used as passwords.

**Input:**

**1)** "python .\Enroll.py testUser word"

**Expected Outputs:**

1. “Rejected” with exit code -1

**4 – Password of the form [WordNum] should return “Rejected”**

**Description:** This tests to see if the system rejects a type of “weak” password. A password with numbers at the end is similarly weak since it is easier for adversaries to brute force.

**Input:**

**1)** "python .\Enroll.py testUser word123"

**Expected Outputs:**

**1)** “Rejected” with exit code -1

**5 – Password of the form [NumWord] should return “Rejected”**

**Description:** This tests to see if the system rejects a type of “weak” password. A password with numbers at the start is similarly weak since it is easier for adversaries to brute force.

**Input:**

**1)** "python .\Enroll.py testUser 123word"

**Expected Outputs:**

**1)** “Rejected” with exit code -1

**6 – Trying to enroll with a username that is already in the database should return “Rejected”**

**Description:** This test makes sure that the data being added to the database is valid. If multiple users were enrolled with the same username, the ‘authenticate’ program would be unable to determine which user is trying to login.

**Input:**

**1)** "python .\Enroll.py userAlreadyInDatabase GoodPassword"

**Expected Outputs:**

**1)** “Rejected” with exit code -1

**Other Tests:**

* **Database should be encrypted and stay encrypted**
* **Edge cases should be handled (Max length strings for username and password)**