-Login System with Registration and Login Attempt Tracking-

I. INTRODUCTION

In today's world, securing user credentials is of paramount importance. A key aspect of any secure system is its ability to properly authenticate users while ensuring their sensitive information, such as passwords, is kept secure. This project aims to build a simple **login system** where users can register and log in securely. The system hashes passwords before storing them in a MySQL database, making it resistant to unauthorized access even in the event of a data breach.

Additionally, the system logs the **login attempts** to track the number of successful and failed login attempts for each user, helping to detect any suspicious activity or potential brute-force attacks.

This project leverages:

- **Tkinter** for the graphical user interface (GUI),
- **bcrypt** for password hashing, and
- **MySQL** for database management.

II. Literature Review-

 Password Hashing and Security: One of the most crucial security practices in any authentication system is password hashing. Storing passwords as plain text in databases can lead to serious security breaches. In modern applications, it is a standard practice to use a cryptographic hashing algorithm to convert a password into a fixed-length string, which cannot be reversed. The **bcrypt** hashing algorithm is widely used because it is slow and computationally expensive, making it difficult for attackers to perform brute-force attacks on hashed passwords.

bcrypt provides a key strengthening mechanism and automatically handles the salting (adding random data to the password) to further enhance security.

MySQL in Authentication Systems: **MySQL** is one of the most popular relational database management systems (RDBMS). It is widely used for storing user credentials securely and efficiently. In this system, MySQL stores user data such as usernames and their hashed passwords. Additionally, a separate table is maintained to store **login attempt statistics**, which helps in tracking user behavior and detecting suspicious login activities.

Tkinter for GUI: **Tkinter** is the standard GUI (Graphical User Interface) library for Python. It is used for creating desktop applications with simple interfaces. In this project, Tkinter is used to create the login and registration forms, allowing users to interact with the system in an intuitive way.

Login Attempt Tracking: Tracking login attempts is important for monitoring the security of an authentication system. This can help prevent brute-force attacks, where attackers try to guess passwords by repeatedly attempting to log in. By logging every failed login attempt, administrators can take necessary actions, such as locking accounts after a certain number of failed attempts or triggering alerts.

.

III. PROJECT DESCRIPTION

The project consists of the following main components:

1. **User Registration**: Users can register by providing a **username** and **password**. The password is hashed using **bcrypt** before being stored in the MySQL database. This ensures that even if the database is compromised, the passwords cannot be easily retrieved.
2. **User Login**: Users can log in by entering their **username** and **password**. The system compares the entered password with the hashed password stored in the database. If the login is successful, the user is granted access to the system. Otherwise, a failed login attempt is logged in the database.
3. **Login Attempt Tracking**: Every login attempt, whether successful or failed, is logged in the **LoginAttempts** table. This table stores the user_id, status (success/failure), and timestamp of each login attempt. This data can be used to generate statistics, such as the number of successful and failed login attempts.
4. **View Login Stats**: Users can view their login statistics, which includes the number of **successful** and **failed** login attempts. This can help users understand their login history and monitor for any suspicious activity.
5. **Database Structure**:
    - **Users Table**: Stores user_id, username, and hashed_password.

- **Login Attempts Table**: Stores user_id, status (success/failure), and timestamp of the attempt.

6. **Graphical User Interface (GUI)**: The application uses **Tkinter** to provide a simple and user-friendly interface. The interface allows users to easily register, log in, and view login statistics with just a few clicks.

IV. PROJECT CODE AND IMPLEMENTATION

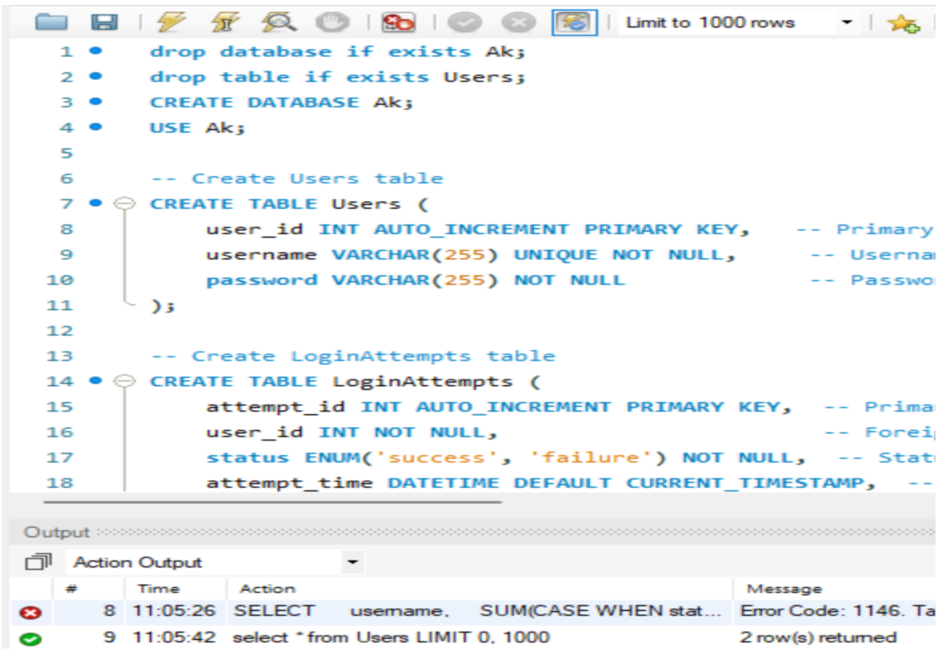
The project consists of several components, including the Python code for registration, login functionality, tracking login attempts, and interacting with a MySQL database. The code also uses Tkinter for the graphical user interface (GUI).

1. Database Setup

Before running the Python code, you need to set up the MySQL database. The structure consists of two tables:

1. **Users**: Stores user information (username and hashed password).
2. **LoginAttempts**: Tracks each login attempt (whether successful or failed).

SQL to create the tables:



```sql
drop database if exists Ak;
drop table if exists Users;
CREATE DATABASE Ak;
USE Ak;

-- Create Users table
CREATE TABLE Users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,    -- Primary
    username VARCHAR(255) UNIQUE NOT NULL,      -- Userna
    password VARCHAR(255) NOT NULL              -- Passwo
);

-- Create LoginAttempts table
CREATE TABLE LoginAttempts (
    attempt_id INT AUTO_INCREMENT PRIMARY KEY,    -- Prima
    user_id INT NOT NULL,                         -- Forei
    status ENUM('success', 'failure') NOT NULL,    -- Stat
    attempt_time DATETIME DEFAULT CURRENT_TIMESTAMP,   --
```

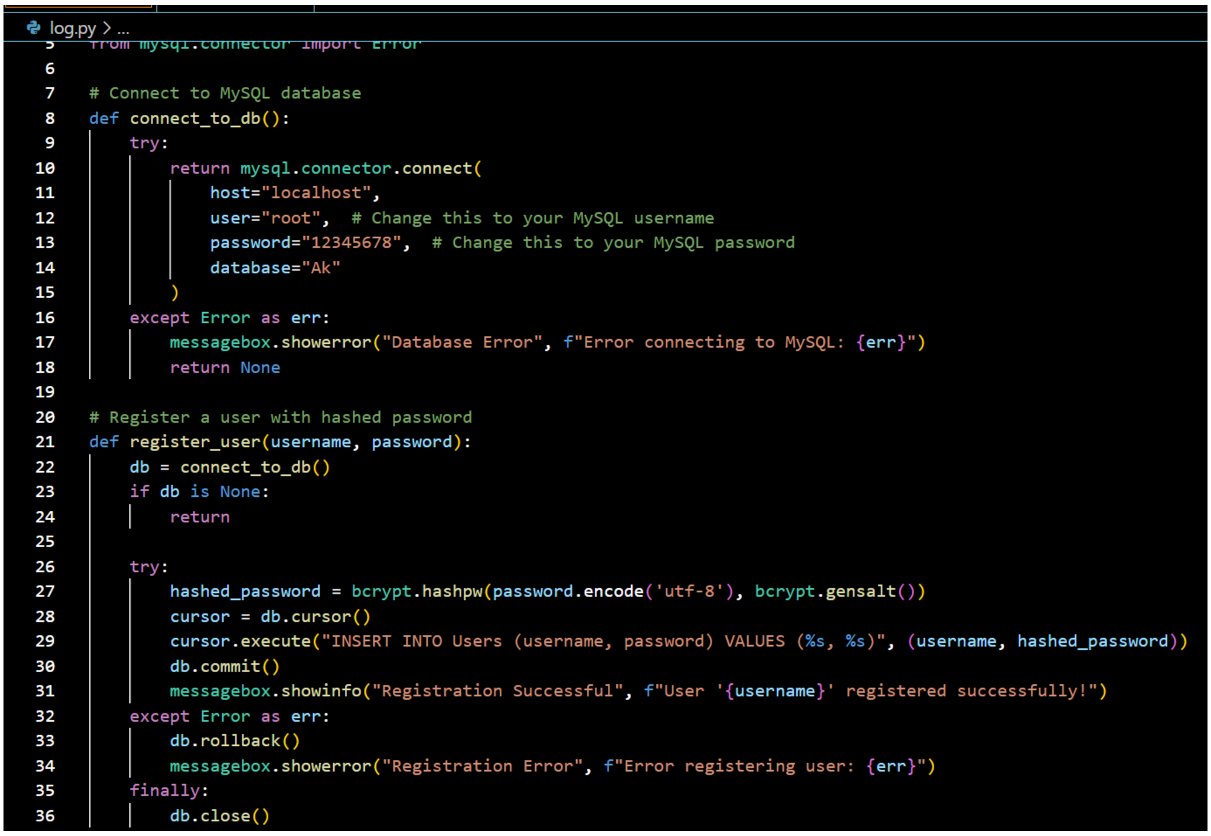
```
#   Time       Action                                      Message
8   11:05:26   SELECT    username,   SUM(CASE WHEN stat...  Error Code: 1146. Ta
9   11:05:42   select * from Users LIMIT 0, 1000            2 row(s) returned
```

This SQL code creates two tables in the database:

- The Users table stores the user_id, username, and password.
- The LoginAttempts table logs the user_id, login status (success or failure), and the timestamp of the attempt.

2. Python Code

Here's the **full Python code** to implement the login system, including registration, login, tracking login attempts, and the Tkinter GUI for user interaction.



```python
from mysql.connector import Error

# Connect to MySQL database
def connect_to_db():
    try:
        return mysql.connector.connect(
            host="localhost",
            user="root",  # Change this to your MySQL username
            password="12345678",  # Change this to your MySQL password
            database="Ak"
        )
    except Error as err:
        messagebox.showerror("Database Error", f"Error connecting to MySQL: {err}")
        return None

# Register a user with hashed password
def register_user(username, password):
    db = connect_to_db()
    if db is None:
        return

    try:
        hashed_password = bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())
        cursor = db.cursor()
        cursor.execute("INSERT INTO Users (username, password) VALUES (%s, %s)", (username, hashed_password))
        db.commit()
        messagebox.showinfo("Registration Successful", f"User '{username}' registered successfully!")
    except Error as err:
        db.rollback()
        messagebox.showerror("Registration Error", f"Error registering user: {err}")
    finally:
        db.close()
```

3. Explanation of Code Components

Database Functions

- connect_to_db(): This function establishes a connection to the MySQL database using the mysql.connector module. It handles any connection errors and returns a connection object or None if the connection fails.
- register_user(): This function registers a new user. It hashes the password using **bcrypt** before storing it in the database. It also handles exceptions and commits the user data to the Users table.
- login_user(): This function verifies if a user's credentials (username and password) match the stored data. If the login is successful, it logs the attempt as a "success"; otherwise, it logs the attempt as a "failure".
- log_attempt(): This function logs each login attempt in the LoginAttempts table. It records whether the attempt was successful or not, along with the user ID.
- get_login_stats(): This function retrieves the login statistics for a specific user, showing the number of successful and failed login attempts.

GUI (Tkinter)

- The graphical interface allows users to input their username and password for registration or login. It also provides a button to view login statistics. The GUI uses Tkinter for creating windows, labels, text fields, and buttons.
- **Buttons**:
  - **Login Button**: Triggers the login process.
  - **Register Button**: Triggers the registration process.
  - **View Stats Button**: Displays the login statistics of the user.

4. How to Run the Project

1. **Set up MySQL Database**: Create the Users and LoginAttempts tables in your MySQL database (use the provided SQL queries).
2. **Install Dependencies**: Install the required libraries by running:
   - pip install mysql-connector-python bcrypt tk
3. **Run the Python Code**: Execute the Python file in your terminal or IDE. The Tkinter GUI will appear, allowing you to interact with the system.
4. **Use the Application**:
   - Register a new user.
   - Log in with the registered username and password.
   - Track your login attempts and view stats.

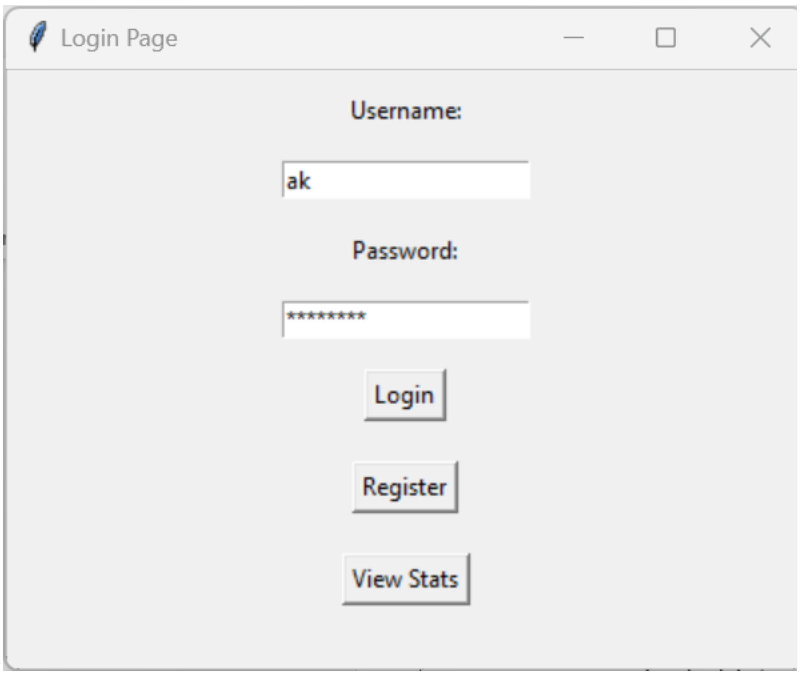
V. RESULT AND OUTPUT

The final system allows users to:

Register an Account: When a user registers, their password is hashed using **bcrypt** and stored securely in the database.

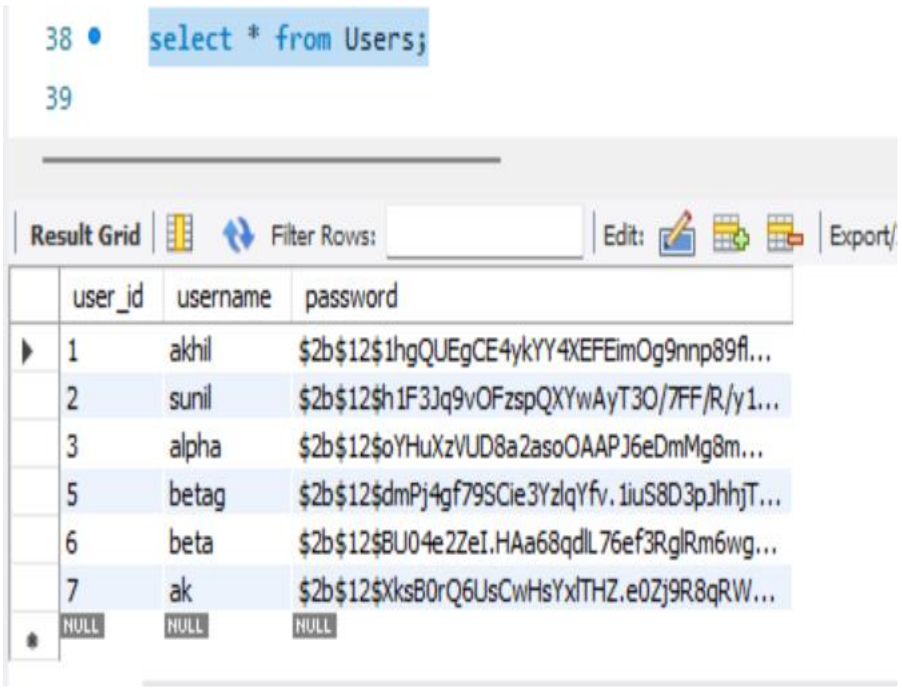
Login to the System: Users can log in by entering their username and password. If the entered password matches the stored hash, they are successfully logged in. Failed attempts are recorded.

1. **View Login Statistics**: The system displays the number of successful and failed login attempts for each user. For example, the output for a user may look like:
2. **Error Handling**: If a user tries to log in with an incorrect password, an error message is displayed:



The system ensures that:

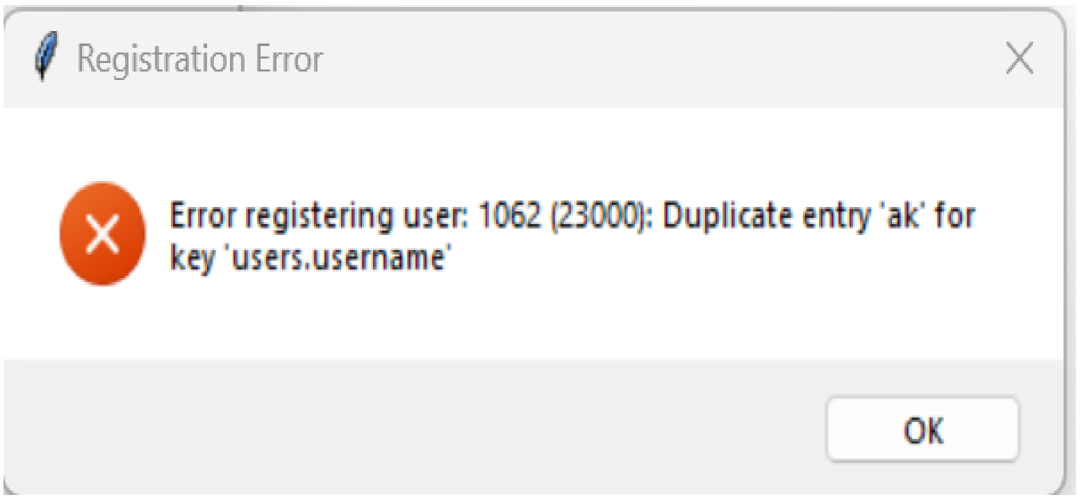
- **Passwords are stored securely** using hashing.
- **Login attempts are tracked**, providing insights into user login behavior.
- **Users can easily interact** with the system via a simple Tkinter interface.



```
select * from Users;
```

user_id	username	password
1	akhil	$2b$12$1hgQUEgCE4ykYY4XEFEimOg9nnp89fl...
2	sunil	$2b$12$h1F3Jq9vOFzspQXYwAyT3O/7FF/R/y1...
3	alpha	$2b$12$oYHuXzVUD8a2asoOAAPJ6eDmMg8m...
5	betag	$2b$12$dmPj4gf79SCie3YzlqYfv.1iuS8D3pJhhjT...
6	beta	$2b$12$BU04e2ZeI.HAa68qdlL76ef3RglRm6wg...
7	ak	$2b$12$XksB0rQ6UsCwHsYxlTHZ.e0Zj9R8qRW...
NULL	NULL	NULL

.



Registration Error

```
Error registering user: 1062 (23000): Duplicate entry 'ak' for
key 'users.username'
```

VI. REFERENCES

1. *Bcrypt documentation:- https://pypi.org/project/bcrypt/*
2. MySQL documentation: - *https://dev.mysql.com/doc/*
3. *Tkinter documentation*:- https://docs.python.org/3/library/tkinter.html
4. Python documentation:- https://docs.python.org/