PROBLEM STATEMENT

Aim:The aim of creating a registration form using CRUD (Create, Retrieve, Update, Delete) operations is to develop a user-friendly web application that allows users to manage their registration information efficiently. This application will enable users to:

- Create: Register by providing their personal information, such as name, email, password, and other relevant details.
- Retrieve: View their registration details and make any necessary changes.
- Update: Modify their registration information, ensuring that their profile remains accurate and up-to-date.
- Delete: Delete their registration information and associated data, should they choose to unregister from the system.

Problem Statement: Design and Implement a Registration Form with CRUD (Create, Retrieve, Update, Delete) Operations.

Description: You are tasked with developing a web-based registration system that allows users to register, retrieve, update, and delete their information. This system will serve as a basic user management system and should provide a user-friendly interface for performing these CRUD operations.

Requirements:

Registration Form (Create):

- Users should be able to access a registration form where they can enter their personal information, such as name, email, username, password, and any other relevant details.
- Validate user input to ensure data integrity (e.g., valid email format, strong password requirements).

Retrieve User Information (Read):

- Implement a feature that allows users to retrieve their information by searching for it using a unique identifier, such as their email or username.
- Display the retrieved information in a user-friendly format.

Update User Information (Update):

- Provide an interface that enables users to update their information. Users should be able to edit their details, including name, email, and password.
- Ensure that changes are validated and stored securely.

Delete User Account (Delete):

- Implement a feature that allows users to delete their accounts. When a user

chooses to delete their account, their information should be removed from the system.

User Authentication and Security:

- Implement user authentication to ensure that only registered users can access their information and perform CRUD operations on their accounts.
- Store user passwords securely by using encryption and hashing techniques (e.g., bcrypt).
- Protect against common web security vulnerabilities (e.g., SQL injection, cross-site scripting) to ensure the system's security.

User-Friendly Interface:

- Design a user-friendly and responsive web interface that is easy to navigate and interact with.
- Provide appropriate feedback messages for successful and unsuccessful CRUD operations.

Database Integration:

- Use a database system (e.g., MySQL, PostgreSQL, MongoDB) to store user information.
- Implement the necessary database queries and operations for CRUD functionality.

Error Handling:

- Implement robust error handling to gracefully handle unexpected situations, such as database connection failures or validation errors.

Logging and Monitoring:

- Implement logging to keep a record of important system events and user activities.
- Set up monitoring to detect and respond to system issues or security threats.

Testing:

- Perform comprehensive testing, including unit testing, integration testing, and user acceptance testing, to ensure the system's reliability and functionality.

Documentation:

- Provide clear and comprehensive documentation, including instructions for users on how to use the registration system and for developers on how to maintain and extend it.

Scalability and Performance:

- Ensure that the system can handle a growing number of users and data without significant performance degradation.

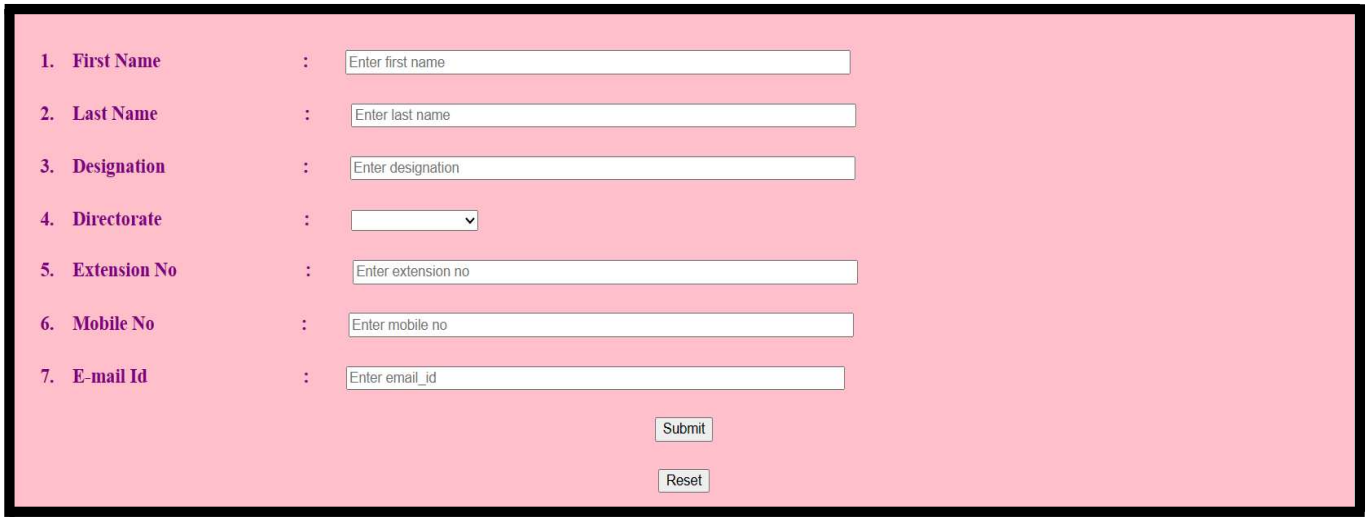

Figure: Final Outcome of our Project

The above picture is the registration form which we have developed in the problem statement.