**Mail Management System**

**Introduction**The **Mail Management System** is a web application developed using Angular, aimed at improving the management of email communications within an organization. The application provides a structured platform for composing, organizing, and tracking emails, ensuring both efficiency and security.

**Application Overview**

**Key Features:**

1. **User Authentication:**

- The application includes secure user registration and login functionalities.

- Passwords are encrypted for security, ensuring that sensitive information remains protected.

2**. Dashboard:**

- After logging in, users are taken to the dashboard, which displays existing email records.

- The dashboard is designed to be user-friendly, allowing users to view, update, or delete emails easily.

3. **Mail Component:**

- Users can add new emails by providing details such as Name, Mail Type, Detail, Date, and Tab Type.

- The Mail Type options include Work, Business, Friend, and Important, while the Tab Type includes Inbox, Sent, and Spam.

- Deleted emails are moved to a "Deleted" tab, where they can be reviewed or permanently deleted.

4. **CRUD Operations:**

- The application allows users to Create, Read, Update, and Delete email records.

- It also supports multiple deletions, making it easier to manage large volumes of emails.

**Project Details**

- **Application Name:** The project was named `mail-app-studentname`, in line with the Angular application.

- **Application Title:** The title, `Mail Management System`, is consistently used across the application.

- **Technology Stack**: The project was built using Angular, with Angular Material employed for the user interface components.

**Project Structure**

1 . **Component-Based Architecture:**

* + The application is structured into components, each responsible for a specific part of the UI or functionality.
  + Components include LoginComponent, DashboardComponent, and MailComponent, among others.

2. **Routing:**

* + Angular routing is used to navigate between different views, such as the login page, dashboard, and email creation form.
  + Lazy loading is implemented to optimize the loading time of the application.

**Components Developed:**

1. **Login Component:**

- A login page was created to authenticate users, with fields for username and password.

- Passwords are hidden during input for added security.

2. **Dashboard Component:**

- This component displays five existing email records and provides navigation options to add, update, or delete records.

3. **Mail Component:**

- The mail creation form includes fields for various details and dropdowns for Mail Type and Tab Type.

**Conclusion**

Completing this Angular application has been an important step in my journey as a developer. The project has provided hands-on experience with key Angular concepts, including:

- Setting up and structuring an Angular project.

- Implementing data binding and managing user interactions.

- Creating responsive and intuitive user interfaces with Angular Material.

- Performing CRUD operations and implementing form validation.

- Utilizing Angular routing for efficient navigation.

This project not only demonstrates the technical skills I have acquired but also highlights my ability to design and implement a functional application from scratch. The knowledge gained from this project will be invaluable as I take on more complex projects in the future.