LIBRARIES AND MODULES USED IN THE PROJECT

Here is the detailed explaination of the libraries & modules in the project

1. **Flask**:
    - **Flask**: A lightweight web framework written in Python that allows you to build web applications. It provides tools and libraries for building web servers, routing requests, and managing sessions.
    - **Modules from Flask**:
        - Flask: The core class to initialize a Flask application.
        - jsonify: Converts Python dictionaries to JSON format to be sent as responses.
        - render_template: Renders HTML templates with dynamic content.
        - request: Accesses request data (e.g., form data, query parameters).
        - redirect: Redirects the user to a different endpoint.
        - url_for: Constructs a URL for a given endpoint.
        - g: Global request object for holding request-scoped data.
        - session: Manages server-side sessions, typically used for user authentication.
        - flash: Flashes a message that can be displayed to the user in the next request (e.g., success or error messages).
2. **pymysql**:
    - A MySQL database driver for Python that allows you to connect and interact with a MySQL database.
    - **Modules from pymysql**:

        - pymysql: Core module for connecting to the MySQL database.
        - pymysql.cursors: Provides different types of cursors to interact with database results, such as DictCursor, which returns query results as dictionaries rather than tuples.

3. **secrets**:
    - The secrets module is used for generating cryptographically secure random numbers or tokens. This can be useful for generating secure tokens for authentication, session management, or other security-sensitive tasks.
4. **werkzeug.security**:
    - A module from the Werkzeug library, which is a comprehensive WSGI web application library.
    - **Functions from werkzeug.security**:
        - generate_password_hash: Used to hash a password for secure storage.
        - check_password_hash: Used to check whether a provided password matches a previously hashed password.
5. **flask_mail**:
    - A Flask extension for sending emails from your Flask applications.
    - **Modules from flask_mail**:
        - Mail: Initializes the mail settings for your Flask app.
        - Message: Defines the structure of the email message (e.g., subject, sender, recipients, and body).
6. **dotenv**:
    - **load_dotenv**: Loads environment variables from a .env file into your application's environment. It is typically used for securely managing configuration values (e.g., database credentials, email settings).

7. **datetime**:
    - A built-in Python module used to work with dates and times.
    - **Classes and methods**:
        - datetime: The primary class for representing dates and times.
        - timedelta: Represents the difference between two dates or times. Useful for tasks such as expiring sessions or tokens after a period.

These modules and libraries form the core of a Flask-based application, enabling features like database interaction, secure password management, email functionality, session management, and time-based operations.