**1. Einleitung**

Die Authentifizierung ist ein zentraler Bestandteil moderner Webanwendungen. Sie gewährleistet, dass nur autorisierte Benutzer Zugriff auf bestimmte Ressourcen erhalten. In der vorliegenden Implementierung wird der Registrierungs- und Login-Prozess mithilfe des Flask-Frameworks, SQLAlchemy als ORM, Flask-Mail zur E-Mail-Kommunikation und Flask-Login für das Session-Management realisiert. Ziel dieser Ausarbeitung ist es, den Ablauf der Registrierung, E-Mail-Bestätigung sowie des Logins und Logouts im Detail zu erläutern.

**2. Registrierungsprozess**

**2.1 Benutzererfassung und Validierung**

Beim Aufruf der Registrierungsroute (/auth/register) wird zunächst ein Formular angezeigt, in dem der Benutzer seine E-Mail-Adresse und ein Passwort eingeben kann. Die E-Mail-Adresse dient als Basis für den Benutzernamen, da aus der Adresse der Teil vor dem „@“-Symbol extrahiert wird. Anschließend erfolgt eine Validierung, bei der überprüft wird, ob bereits ein Benutzer mit dem gleichen Benutzernamen oder der gleichen E-Mail-Adresse existiert. Diese Prüfungen werden über entsprechende Datenbankabfragen (SQLAlchemy-Query) realisiert.

**2.2 Passwort-Handling und Persistierung**

Falls die Validierung erfolgreich verläuft, wird ein neues Benutzerobjekt erzeugt. Vor dem Speichern in der Datenbank wird das eingegebene Passwort mit Hilfe der werkzeug.security-Funktionen gehasht. Dadurch wird sichergestellt, dass das Passwort nicht im Klartext in der Datenbank abgelegt wird. Anschließend wird der neue Benutzer persistiert, indem er der Session hinzugefügt und die Änderungen mittels db.session.commit() dauerhaft gespeichert werden.

**2.3 E-Mail-Bestätigung**

Um sicherzustellen, dass die angegebene E-Mail-Adresse auch tatsächlich dem Benutzer gehört, wird ein Bestätigungsmechanismus implementiert. Es wird ein Token generiert, das mit der Bibliothek **itsdangerous** erstellt wird. Dieser Token beinhaltet verschlüsselt die E-Mail-Adresse und wird mit einem Salt (hier: 'email-confirm-salt') versehen. Anschließend wird über Flask-Mail eine Bestätigungs-E-Mail an den Benutzer versendet, die einen Link enthält. Dieser Link verweist auf die Route /auth/confirm/<token>, die im nächsten Abschnitt erläutert wird.

**3. E-Mail-Bestätigung**

Die Route /auth/confirm/<token> dient der Verifikation des Bestätigungslinks. Beim Aufruf dieser Route wird der Token mittels serializer.loads() wieder entschlüsselt. Hierbei wird zusätzlich ein Zeitlimit (max\_age) definiert, um zu verhindern, dass alte oder kompromittierte Token unbegrenzt verwendet werden können.

* **Token-Überprüfung:**  
  Sollte der Token nicht validiert werden können (zum Beispiel, weil er abgelaufen ist oder manipuliert wurde), erhält der Benutzer eine Fehlermeldung.
* **Aktualisierung des Benutzerstatus:**  
  Bei erfolgreicher Verifikation wird in der Datenbank das Feld confirmed des entsprechenden Benutzers auf True gesetzt. Damit signalisiert die Anwendung, dass der Benutzer seine E-Mail-Adresse verifiziert hat und somit vertrauenswürdig ist.

**4. Login-Prozess**

**4.1 Eingabe und Identifikation**

Beim Login gibt der Benutzer seine Anmeldedaten (entweder E-Mail oder Benutzernamen sowie Passwort) in ein Formular ein. Die Implementierung erlaubt beide Varianten der Authentifizierung. Entsprechend wird anhand des Eingabewerts ermittelt, ob eine Suche über die E-Mail oder den Benutzernamen erfolgen soll.

**4.2 Passwort-Validierung**

Nach erfolgreicher Identifikation des Benutzers erfolgt die Überprüfung des eingegebenen Passworts. Hierbei wird die zuvor gehashte Version des Passworts mit dem eingegebenen Wert verglichen. Nur wenn das Passwort korrekt ist, wird der Benutzer als authentifiziert betrachtet.

**4.3 Verwendung von Flask-Login**

Anstelle der manuellen Verwaltung der Session übernimmt die Bibliothek **Flask-Login** das Session-Management. Durch den Aufruf der Funktion login\_user(user) wird der Benutzer in die Session eingeloggt. Flask-Login überprüft dabei automatisch wichtige Eigenschaften wie is\_active, is\_authenticated und ruft die Methode get\_id() auf, um den Benutzer eindeutig zu identifizieren. Dies erleichtert nicht nur die Verwaltung, sondern sorgt auch für ein sichereres und standardisiertes Handling der Benutzer-Sessions.

**4.4 Feedback und Weiterleitung**

Nach erfolgreichem Login wird dem Benutzer eine Bestätigungsmeldung angezeigt. Anschließend erfolgt eine Weiterleitung auf eine geschützte Startseite oder das Dashboard, was signalisiert, dass der Benutzer nun Zugriff auf weitere Funktionen der Anwendung hat.

**5. Logout-Prozess**

Für den Logout wird ebenfalls auf Flask-Login zurückgegriffen. Die Funktion logout\_user() entfernt den Benutzer aus der aktiven Session. Dies stellt sicher, dass der Benutzer nach dem Logout keine geschützten Seiten mehr aufrufen kann. Auch hier wird dem Benutzer ein entsprechendes Feedback angezeigt und er wird zur Login-Seite weitergeleitet.

**6. Fazit**

Die Implementierung des Registrierungs- und Login-Prozesses in der vorliegenden Flask-Anwendung illustriert, wie verschiedene Bibliotheken (SQLAlchemy, Flask-Mail, itsdangerous und Flask-Login) zusammenwirken, um eine sichere und benutzerfreundliche Authentifizierung zu gewährleisten. Die Verwendung von Token-basierten E-Mail-Bestätigungen erhöht die Sicherheit, indem sie sicherstellt, dass nur verifizierte E-Mail-Adressen zur Registrierung genutzt werden können. Gleichzeitig erleichtert Flask-Login das Session-Management und reduziert die Komplexität der Codebasis. Insgesamt zeigt dieser Ansatz, wie moderne Webanwendungen Authentifizierungsmechanismen implementieren können, um sowohl Sicherheit als auch Benutzerfreundlichkeit zu gewährleisten.

Diese Ausarbeitung bietet eine umfassende Übersicht über die technischen Details und den Ablauf der Benutzerregistrierung und -authentifizierung und ist somit geeignet, als Teil einer Bachelor-Arbeit oder als fundierte Dokumentation einer Flask-Anwendung herangezogen zu werden.