■ Day-10 – User Authentication Tasks (MongoDB Focused)

■ Person-2 (Backend – Routes & Auth Logic)

- Create User Routes (user_route.py): - POST /users/signup → Insert user with hashed password. - POST /users/login → Validate credentials, return JWT token. - GET /users/me → Fetch profile using JWT auth.
- Password Security: - Use bcrypt or werkzeug.security for password hashing. - Never store raw passwords.
- JWT Auth Setup: - Install and configure PyJWT or flask-jwt-extended. - Ensure tokens expire (e.g., 1h).
- Postman Testing: - Test signup, login, and me endpoints with valid & invalid data. - Share sample JSON responses with Person-1.

■ Person-1 (MongoDB Schema + Frontend Integration)

- MongoDB Setup (users collection): - Schema with username, email, password_hash, created_at, role. - Add unique index on email.
- Frontend (React/Streamlit): - Build Signup page → calls POST /users/signup. - Build Login page → calls POST /users/login. - Store JWT token in localStorage (React) or session_state (Streamlit). - Redirect to Dashboard after login.
- Protected Routes: - Ensure Summarizer, Quiz, Flashcards pages check for JWT before access. - Show 'Login Required' message if not authenticated.
- Testing & Demo: - Use dummy accounts (from Person-2 tests). - Try logging in → confirm UI unlocks restricted features. - Take screenshots of working signup/login flow.

■ End of Day-10 Deliverables

- users collection in MongoDB with unique email + hashed passwords.
- Working backend routes (signup, login, me) tested in Postman.
- JWT authentication fully functional.
- Frontend login/signup pages connected to backend.
- Screenshots of DB entries + login UI.