🔐 Day-10 Authentication Implementation Summary

Project: AI Class Assistant

Date: August 20, 2025

Person-1 Tasks: MongoDB Setup + Frontend Integration

Person-2 Tasks: Backend Routes & Auth Logic ✅ Completed

📋 Tasks Completed Today

☑ MongoDB Setup (Person-1)

- **Enhanced existing `database.py`** with authentication support
- **Created unique index on email field** for user authentication
- **Verified schema compatibility** with Person-2's backend routes
- **Database ready** for user registration and login

☑ Frontend Authentication (Person-1)

- **AuthContext:** JWT token management and global auth state
- **Login Page:** Calls POST `/users/login` endpoint
- **Signup Page:** Calls POST `/users/signup` endpoint
- **Protected Routes:** JWT verification before accessing features
- **Navigation:** Dynamic user menu with logout functionality

☑ Backend Integration (Person-2)

- **Authentication Routes:** `/users/signup`, `/users/login`, `/users/me`
- **JWT Implementation:** 1-hour token expiry with secure hashing
- **Password Security:** bcrypt hashing, validation requirements

- **Database Integration:** User collection with proper schema

🛠 Technical Implementation

Database Schema

```
{
  "_id": ObjectId,
  "email": "string (unique)",
  "password_hash": "string (bcrypt)",
  "full_name": "string",
  "created_at": datetime
}
```

API Endpoints

- `POST /users/signup` - User registration
- `POST /users/login` - User authentication
- `GET /users/me` - Get current user (JWT protected)

Frontend Components

- `AuthContext.js` - Global authentication state
- `LoginPage.js` - User login form
- `SignupPage.js` - User registration form
- `ProtectedRoute.js` - Route protection wrapper

🔒 Security Features

Password Requirements

- Minimum 8 characters
- At least 1 uppercase letter
- At least 1 lowercase letter

- At least 1 number

JWT Token

- 1-hour expiration time
- Stored in localStorage
- Automatically included in API requests
- Verified on protected routes

Database Security

- Unique email constraint
- Password hashing with werkzeug
- No plain text password storage

🎯 Protected Features

After authentication, users can access:

- 🎯 **Flashcards** - Create and study flashcards
- 📝 **Quizzes** - Take quizzes and track progress
- 📊 **Statistics** - View learning analytics
- 📁 **File Upload** - Upload and process documents

☑️ Testing Results

Postman API Testing

- ✅ User signup successful (201 response)
- ✅ User login returns JWT token (200 response)
- ✅ Protected routes verify JWT properly
- ✅ Duplicate email validation working
- ✅ Password validation enforced

Frontend Integration

- ✅ Signup form connects to backend API
- ✅ Login form authenticates users successfully
- ✅ JWT tokens persist across browser refresh
- ✅ Protected routes redirect to login when needed
- ✅ Navigation shows user info when authenticated

📱 User Experience Flow

1. **New User:** Signup → Auto-login → Dashboard access
2. **Returning User:** Login → Dashboard with preserved session
3. **Protected Access:** Automatic redirect to login if not authenticated
4. **Session Management:** Logout clears token and redirects appropriately

🗂 File Structure

```
Frontend/src/
├── context/AuthContext.js        ✅ New
├── components/
│   ├── LoginPage.js              ✅ New
│   ├── SignupPage.js             ✅ New
│   ├── ProtectedRoute.js         ✅ New
│   └── AuthPages.css             ✅ New
├── App.js                        ✅ Enhanced
└── App.css                       ✅ Enhanced

Backend/
├── database.py                   ✅ Enhanced
├── routes/user_routes.py         ✅ Person-2
└── app.py                        ✅ Person-2 + JWT
```

🎉 Integration Success

Person-1 + Person-2 Collaboration

- ✅ Frontend perfectly integrates with Person-2's API
- ✅ Database schema matches backend expectations
- ✅ Error handling displays backend validation messages
- ✅ JWT token format compatible between frontend/backend

Preserved Functionality

- ✅ All existing features maintained
- ✅ Quiz system now requires authentication
- ✅ Flashcard system now user-specific
- ✅ Statistics tracking per authenticated user

📊 Deliverables Completed

- [x] MongoDB users collection with unique email index
- [x] Working backend authentication routes (Person-2)
- [x] JWT authentication fully functional
- [x] Frontend login/signup pages connected to backend
- [x] Protected routes ensuring feature access control
- [x] User session management and navigation
- [x] Successful integration testing completed

🏆 **Result:** Complete authentication system successfully implemented and tested. Users can now register, login, and securely access all AI Class Assistant features with proper session management and route protection.