EdTech Assignment Tracker:

Name: Sharath M Talawar

Email ID: sharathmtalawar@gmail.com

Project Github Repository: https://github.com/sharath816/edtech-assignment-tracker

System Architecture:

The system is a **web-based assignment tracker** for teachers and students.

1. Architecture Components:

Component	Technology Used	Purpose
Frontend	HTML, CSS, JavaScript, Bootstrap	User interface for teachers & students
Backend	Python with FastAPI	RESTful API handling all core functionalities
Database	SQLite (can later migrate to PostgreSQL)	Stores user, assignment, and submission data
Authentication	JWT (JSON Web Token)	Secure login and protected API access
File Storage	Local uploads/ folder (S3 for prod)	Stores submitted assignment files

2. Core Entities & Relationships

ER Diagram (Tabular Format):

Entity	Attributes	Relationships
User	id, username, hashed_password, role (student/teacher)	One-to-Many with Assignments (if teacher), Submissions
Assignment	id, title, description, created_by (teacher ID)	One-to-Many with Submissions
Submission	id, assignment_id, submitted_by (student ID), file_path, timestamp	Belongs to one User and one Assignment

3. API Endpoints

3.1. Teacher Creates Assignment

- **Endpoint**: POST /assignments
- **Auth**: Required (teacher only)
- **Request**:

json

```
{
  "title": "DSA Assignment 1",
  "description": "Implement binary search and explain time complexity"
}
```

- **Response**:

json

```
{
  "message": "Assignment created successfully"
}
```

3.2. Student Submits Assignment

- **Endpoint**: POST /submit
- **Auth**: Required (student only)
- **Request**: multipart/form-data
    - assignment_id: integer
    - file: assignment file (PDF, DOCX, etc.)
- **Response**:

json

```
{
  "message": "Assignment submitted successfully"
}
```

3.3. Teacher Views Submissions

- **Endpoint**: GET /submissions/{assignment_id}
- **Auth**: Required (teacher only)
- **Response**:

json

```
[
  {
    "submission_id": 7,
    "assignment_id": 3,
    "student_id": 5,
    "student_username": "john_doe",
    "file_path": "uploads/...",
  },
```

]

3.4. Delete Assignment (New)

- **Endpoint**: DELETE /assignments/{assignment_id}
- **Auth**: Required (teacher only)
- **Function**: Removes assignment and associated submissions from DB

4. Authentication Strategy

- **JWT-based Authentication**: On login, a token is issued containing username and role
- **Role-based Access Control**:
    - student: can only submit assignments and view their submissions
    - teacher: can create assignments, view submissions, and delete their own assignments
- **FastAPI Dependency Injection** is used:

current_user: User = Depends(get_current_user)

5. Future Scalability Plans

Component	Current	Future Plan
Database	SQLite	Use PostgreSQL or MySQL with pooling
File Storage	Local uploads/ folder	Migrate to Amazon S3 or Azure Blob Storage
Auth	Basic JWT	Add refresh tokens, OAuth integration
Frontend	HTML + JS	Migrate to React.js or Vue.js for SPA
Backend	Monolithic FastAPI app	Split into microservices: Auth, Assignment, Submission

Component	Current	Future Plan
Deployment	Local	Deploy on cloud (e.g. AWS EC2 + RDS + S3)
Performance	N/A	Add Redis caching, background tasks for large files