**Assignment Submission Portal - Backend Documentation**

**Overview**

This is a backend system for an assignment submission portal where users can upload their assignments and admins can review, accept, or reject them. The system uses Express.js in Node.js, with MongoDB as the database, and TypeScript was used to enhance type safety and maintainability.

**Key Features:**

1. **User Registration & Login**: Users can register, log in, and upload assignments.
2. **Admin Management**: Admins can register, log in, and view assignments tagged to them. Admins can accept or reject assignments.
3. **Assignment Upload & Review**: Users can upload assignments and admins can accept or reject them.

**System Setup**

**Prerequisites**

To run this project, ensure that you have the following installed:

* **Node.js**
* **MongoDB**
* **Git**

**STEPS**  
  
Clone the repository from GitHub

Install the required dependencies

Create a .env file in the root directory and add the following:

MONGO\_URI=<Your MongoDB Connection String>

JWT\_SECRET=<A random secret key for JWT authentication>

PORT=5000

**API Endpoints**

**User Endpoints**

**1. POST /register - Register a new user**

**2. POST /login - User login**

**3. POST /upload - Upload an assignment**

**4. GET /admins - Fetch all admins**

**Admin Endpoints**

**1. POST /register - Register a new admin**

**2. POST /login - Admin login**

**3. GET /assignments - View assignments tagged to the admin**

**4. POST /assignments/:id/accept - Accept an assignment  
  
5. POST /assignments/:id/reject - Reject an assignment**

**Database Schema**

**User Schema**

import { Schema, model, Document } from 'mongoose';

const userSchema = new Schema({

username: { type: String, required: true },

password: { type: String, required: true },

});

export interface IUser extends Document {

username: string;

password: string;

}

export const User = model<IUser>('User', userSchema);

**Assignment Schema**

import { Schema, model, Document } from 'mongoose';

const assignmentSchema = new Schema({

userId: { type: String, required: true },

task: { type: String, required: true },

admin: { type: String, required: true },

timestamp: { type: Date, default: Date.now },

status: { type: String, enum: ['pending', 'accepted', 'rejected'], default: 'pending' },

});

export interface IAssignment extends Document {

userId: string;

task: string;

admin: string;

timestamp: Date;

status: 'pending' | 'accepted' | 'rejected';

}

export const Assignment = model<IAssignment>('Assignment', assignmentSchema);

**Authentication and Authorization**

* **Authorization**: Admins have access to specific routes like viewing and managing assignments, while users only have access to upload and view their own assignments.

**Validation**

* **User Input Validation**: All incoming requests (e.g., registration, login, upload) are validated for required fields, valid formats (e.g., email), and password strength.
* **Error Handling**: Clear error messages are returned for invalid inputs, unauthorized access, or any other issues.

**How to Run the Project**

1. **Clone the repository**
2. **Install dependencies**
3. **Set up MongoDB**
4. **Start the server**
5. **Access the API**

**Conclusion**

This system provides a simple, scalable backend for assignment submissions with user authentication and assignment management for admins. The modular structure ensures that the code is easy to maintain and extend for future features.