**Project Overview**

This Django-based system is designed to assist lecturers in marking assignments and exams. It leverages AI models to analyze and evaluate student responses against a rubric provided by the lecturer, generating both scores and explanations for each question. The system will utilize the LangChain framework for language model capabilities and the Groq API for efficient AI processing.

**Key Features and Requirements**

**1. User Roles and Authentication**

* **Lecturer**: Can create, upload, and manage rubrics and assignments/exams for marking.
* **Admin**: Can manage system settings and user accounts.
* **Authentication**: Use Django’s built-in authentication system to manage user login and permissions.

**2. Assignment and Exam Management**

* **Assignment/Exam Creation**: Lecturers can upload assignments and exams as PDF files.
* **Rubric Upload**: Lecturers upload the rubric as a PDF, which defines the criteria and marks for each question.

**3. AI-Driven Marking System**

* **Processing of PDFs**: Use OCR or PDF parsing libraries (like PyMuPDF or PDFplumber) to extract text from student responses and rubric PDFs.
* **AI Model Integration**:
  + Integrate **LangChain** to interpret the rubric and analyze student answers based on evaluation criteria.
  + **Marking Algorithm**: Implement an AI model that compares student responses with rubric requirements, assigns marks to each question, and generates a justification for each mark given.
  + **Justification and Feedback**: Use natural language generation (NLG) to create feedback based on what aspects of the rubric the student met or missed.
* **Groq API**: Utilize the Groq API for efficient and accelerated AI model inference, especially during batch processing or complex rubric analysis.

**4. Interface and User Experience**

* **Dashboard for Lecturers**:
  + View a list of assignments and exams with options to upload rubrics and student submissions.
  + Display overall marks and detailed feedback for each student submission.
* **Student Dashboard**:
  + View graded assignments and exams, along with detailed feedback on each question.
* **Bootstrap 5 Frontend**:
  + Create a responsive and user-friendly interface with Bootstrap 5.
  + Use components like cards, modals, and tables to display assignments, submissions, and feedback.

**5. Feedback and Justification System**

* **Feedback Generation**:
  + The system should clearly outline why each mark was given, detailing any rubric criteria that were met or missed.
  + Justifications should be auto-generated using the AI model, based on a comparison between rubric requirements and the student’s response.
* **Highlight Missed Rubric Points**: Identify rubric elements the student missed, and provide actionable suggestions on how they could improve.

**6. Detailed Reports and Analytics**

* **Per-Assignment Summary**: Display a breakdown of how students performed overall and on individual questions.
* **Mark Distribution**: Visualize mark distribution to help lecturers assess overall student performance.
* **Exportable Reports**: Generate PDF reports for individual student marks and feedback or summary reports for the whole class.

**Technical Requirements**

**Backend (Django)**

* **Models**:
  + Assignment: Stores assignment/exam information, submission deadlines, etc.
  + Rubric: Holds rubric details for each question, including criteria and marks distribution.
  + Submission: Stores student submissions and links them to assignments.
  + Feedback: Generated feedback for each question in a student submission.
* **Views and Logic**:
  + Create views to handle file uploads (assignments, rubrics, and submissions) and display marks and feedback.
  + Incorporate business logic for AI-driven marking and feedback generation.
* **Integration with LangChain and Groq API**:
  + Use LangChain for processing rubric language, evaluating student responses, and generating feedback text.
  + Use Groq API to optimize model performance and processing times, particularly when handling multiple student submissions.

**Frontend (Bootstrap 5)**

* **Responsive Design**: Ensure the interface is mobile-friendly and easily navigable with Bootstrap 5.
* **User Flow**:
  + Dashboard: Provide a clear, user-friendly dashboard for both students and lecturers.
  + Marking Results: Use tables or cards to present student marks and AI-generated feedback.
* **File Uploads and Feedback Display**:
  + Implement forms for lecturers to upload assignments, rubrics, and student submissions.
  + Display the feedback and marks for each question in a structured format that’s easy to understand.

**AI Model Workflow**

1. **Rubric Analysis**:
   * Use LangChain to interpret rubric content and criteria, breaking it down into individual marking criteria per question.
2. **Student Answer Evaluation**:
   * Process student responses to match rubric criteria.
   * Generate a score for each question based on how well the answer meets rubric points.
3. **Feedback and Justification Generation**:
   * Use LangChain to generate a narrative explanation for each mark, highlighting what was done well and what was missed.
   * Streamline feedback generation through Groq API to improve response time and efficiency.

Cursor says

Here's the complete step-by-step guide for marking:

 **Login Process:**

 Go to http://127.0.0.1:8000

 Click "Login" in the top navigation

 Enter your lecturer credentials

 Click "Login"

 **Access Marking Dashboard:**

 Click "Marking" in the top navigation

 Select "Marking Dashboard" from the dropdown

 You'll see all assignments and pending submissions

 **View Submissions:**

 Find the assignment you want to mark

 Click "View Submissions" on the assignment card

 You'll see a list of all student submissions

 **Mark a Submission:**

 Find the student's submission you want to mark

 Click "Mark Now" (or "Review Mark" if already marked)

 You'll see the marking interface with:

 Student's submitted work

 Rubric criteria

 Scoring fields

 Comment boxes

 **Complete the Marking:**

 Enter scores for each criterion

 Add comments for each criterion

 Provide overall feedback

 Click "Submit Marks"

 **Review Marks:**

 Return to the submissions list

 Marked submissions will show:

 Green border

 Score displayed

 "Review Mark" option instead of "Mark Now"

The UI includes visual cues:

 Green borders for marked submissions

 Clear status indicators

 Easy-to-use scoring interface

 Quick access to student work

Would you like me to:

 Add more specific styling for better visual hierarchy?

 Include additional features like batch marking?

 Add filtering and sorting options for submissions?

 Create a marking progress tracker?