Backend Engineer Technical Assessment

Here's a technical assessment designed to evaluate the skills of a senior backend developer for your stack at Lumofy:

Objective: This assessment will evaluate the candidate's backend development skills, problem-solving abilities, and familiarity with our tech stack: Django, Django REST Framework, PostgreSQL, Kubernetes, and basic React.

Section 1: General Coding & Problem Solving (60 minutes)

1. **SQL and Database Design (30 minutes)**
   - **Problem**: Design a database schema for a Learning Management System (LMS) that supports:
     - Multiple courses
     - Students enrolling in courses
     - Teachers assigned to multiple courses
     - Each course having multiple lessons
     - Students tracking lesson completion status
   - **Tasks**:
     - Provide the database schema design (ER diagram optional).
     - Write SQL queries to:
       1. Retrieve all students enrolled in a given course.
       2. Get the progress of each student per course based on lesson completion.
       3. Retrieve the courses a teacher is assigned to.
   - **Tech Stack**: PostgreSQL
2. **Python/Django Problem (30 minutes)**

   Problem: You need to implement a feature where users can upload files and access them later.

   - Create a Django model that supports file uploads.

- Implement a basic Django REST API to:
    - Upload a file
    - List all uploaded files
    - Retrieve a specific file by ID

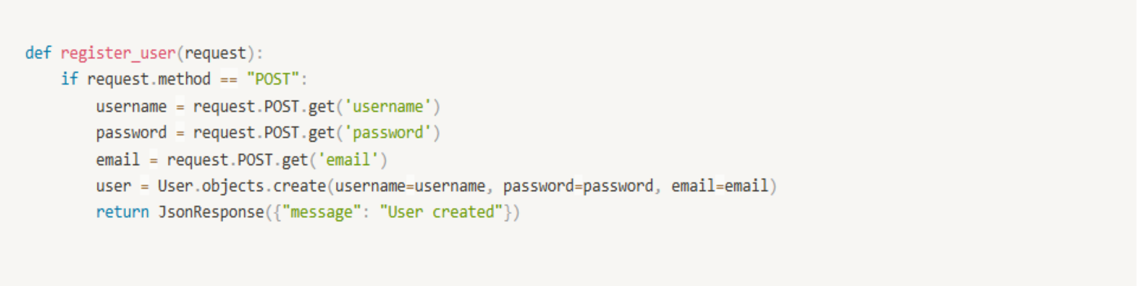
Bonus: Add file size validation and allow filtering files by type (e.g., PDF, image).

Section 2: API Design & Implementation (60 minutes)

1. **Django REST Framework API Task (60 minutes)**
    - **Scenario**: You're building an API for managing learning modules in an LMS. The API should support:
        - Creating, updating, deleting, and retrieving courses.
        - Adding/removing lessons from a course.
        - Tracking student progress for each lesson.
    - **Tasks**:
        - Implement a Django REST API for:
            1. Creating a course (POST).
            2. Retrieving the list of all courses (GET).
            3. Adding/removing lessons from a course (PUT).
            4. Tracking progress of students (GET).
        - Handle edge cases like invalid input, missing resources, and unauthorized access.
    - **Tech Stack**: Django, Django REST Framework

Section 3: Code Review & Engineering Mindset (45 minutes)

1. **Code Review Task (45 minutes)**
    - **Scenario**: You've been given a Django view that handles user registration. The code is functional but lacks optimization and has security vulnerabilities.
    - **Task**: Review the following Django code and provide feedback:
        - Highlight at least 3 potential improvements in terms of performance, security, and maintainability.
        - Refactor the code to address the issues you found.



```
def register_user(request):
    if request.method == "POST":
        username = request.POST.get('username')
        password = request.POST.get('password')
        email = request.POST.get('email')
        user = User.objects.create(username=username, password=password, email=email)
        return JsonResponse({"message": "User created"})
```

Section 4: React Integration (Optional, 30 minutes)

1. **Basic React Task (30 minutes)**
   - **Problem**: Implement a simple React component that fetches the list of courses from the Django REST API and displays them in a list format.
   - **Bonus**: Implement pagination to handle a large number of courses.

Assessment Submission Guidelines:

- Provide the code for each section in a GitHub repository or zip file.
- Ensure your code is modular, well-documented, and includes necessary tests.
- Mention any assumptions or design decisions you’ve made.

This assessment covers essential skills for your senior backend role, ensuring candidates are proficient in your stack and can handle real-world scenarios.