Assessment Overview

Objective: Evaluate the candidate's proficiency in FastAPI, along with their ability to integrate it with other Python features and external systems.

Duration: 4-6 hours (to be completed within a day of sharing)

# Part 1: API Development Task (30% of Total Score)

Build a RESTful API using FastAPI for a hypothetical book review system.

### 

Requirements:

1. Endpoints:

- Add a new book (title, author, publication year).

- Submit a review for a book (text review, rating).

- Retrieve all books with an option to filter by author or publication year.

- Retrieve all reviews for a specific book.

2. Data Validation: Implement data validation using Pydantic models.

3. Documentation: Comments.

4. Error Handling: Implement proper error handling for invalid requests.

### 

# Part 2: Integration with Database (20% of Total Score)

Objective: Enhance the API from Part 2 to persist data using a database (e.g., SQLite, PostgreSQL).

Requirements:

1. Database Integration: Use SQLite to integrate with a database.

2. CRUD Operations: Implement CRUD (Create, Read, Update, Delete) operations for books and reviews.

3. Data Modeling: Design database schema appropriate for the book review system.

### 

# Part 3: Advanced Features and Testing (20% of Total Score)

1. Background Task: Implement a background task for sending a confirmation email (simulated) after a review is posted.

2. Testing: Write tests for the API endpoints using FastAPI's test client.

# Part 4: Theoretical Questions (30% of Total Score)

* - Question 1: Explain how FastAPI handles asynchronous requests and its benefits over synchronous code in Python.
* - Question 2: Describe how dependency injection works in FastAPI and give an example of its practical use.
* - Question 3: Code walkthrough

### 

# Evaluation Criteria

* - Code Quality: Readability, use of Pythonic idioms.
* - Functionality: How well the API meets the specified requirements.
* - Error Handling and Robustness: Graceful handling of edge cases and invalid inputs.
* - Testing: Coverage and effectiveness of tests.
* - Documentation: Clarity and completeness of the API documentation.

### 

# Submission Instructions

Candidates should submit a link to a GitHub repository containing:

* - Source code.
* - README file with setup and usage instructions.
* - Any other necessary documentation.

### 

This assessment will allow us to gauge the candidate's proficiency in FastAPI, their understanding of RESTful API principles, database integration, and their general coding skills and practices.