### Junior Python Developer Skills Test

\*Objective\*: Assess the candidate&#39;s ability to develop a basic 3-tier web application using

Python, REST APIs, and SQLite.

---

\*Scenario\*:

You are tasked with creating a basic Book Management System. This system should allow

users to perform CRUD (Create, Read, Update, Delete) operations on books. The application

will consist of three tiers:

1. \*Database Tier\*: Using SQLite for data storage.

2. \*Application Tier\*: Using Python to handle business logic and data processing.

3. \*Presentation Tier\*: Using a simple web interface to interact with the system.

\*Requirements\*:

1. \*Database Tier\*:

- Create a SQLite database with a single table named books.

- The books table should have the following columns:

- id (Primary Key)

- title (String)

- author (String)

- published\_date (Date)

- isbn (String)

- pages (Integer)

2. \*Application Tier\*:

- Develop a REST API using Python and a web framework of your choice (e.g., Flask,

FastAPI).

- Implement the following endpoints:

- GET /books: Retrieve a list of all books.

- GET /books/&lt;id&gt;: Retrieve details of a specific book by ID.

- POST /books: Add a new book.

- PUT /books/&lt;id&gt;: Update an existing book by ID.

- DELETE /books/&lt;id&gt;: Delete a book by ID.

- Use SQLite for the database.

3. \*Presentation Tier\*:

- Create a simple web interface using HTML/CSS.

- The web interface should allow users to:

- View the list of books.

- Add a new book.

- Update an existing book.

- Delete a book.

\*Instructions\*:

1. \*Setup\*:

- Install necessary dependencies: choose an appropriate web framework and other

necessary libraries.

- Set up the SQLite database connection.

2. \*Database Creation\*:

- Write a script to create the books table in the SQLite database.

3. \*API Development\*:

- Create a Python application with the required REST API endpoints.

- Ensure proper error handling and validation.

4. \*Web Interface\*:

- Develop a simple web interface using HTML and CSS.

- Ensure the web interface communicates with the REST API to perform CRUD operations.

5. \*Submission\*:

- Package the project in a GitHub repository.

- Provide a README file with instructions on how to set up and run the application.

---

\*Assessment Criteria\*:

- \*Code Quality\*: Clean, readable, and well-documented code.

- \*Functionality\*: All specified endpoints and web interface features work correctly.

- \*Design\*: Proper use of a 3-tier architecture.

- \*Error Handling\*: Proper validation and error handling in API and web interface.

- \*Use of Technologies\*: Appropriate use of Python, chosen web framework, REST API

principles, and SQLite.

---

\*Additional Notes\*:

- Candidates may use any additional libraries or tools they deem necessary.

- Ensure that the application is thoroughly tested before submission.

- Focus on clearly understanding the 3-tier architecture and the specified technologies.

Good luck!