Assignment 2: APIs

Objective: This assignment focuses on creating and consuming RESTful APIs. Students will learn to design APIs, integrate them with front-end applications, and consume third-party APIs to fetch and display data.

Part 1: Designing RESTful APIs

1. **Set Up the Environment:**
   - Initialize a Node.js project.
   - Install necessary packages: `express`, `mongoose`, `dotenv`, and `body-parser`.
2. **Create a Simple API:**
   - Design an API for a **Book Management System** with the following endpoints:
     - `GET /books`: Fetch all books.
     - `POST /books`: Add a new book.
     - `PUT /books/:id`: Update a book's details.
     - `DELETE /books/:id`: Delete a book.
   - Book Schema:

```
javascript
CopyEdit
{
  title: String,
  author: String,
  year: Number,
  genre: String,
}
```

3. **API Features:**
   - Implement validation for input data (e.g., ensure "title" and "author" are not empty).
   - Include error handling for scenarios like invalid book IDs or missing fields.

Part 2: Consuming Third-Party APIs

1. **Select a Public API:**
   - Use a third-party API like OpenWeather, NewsAPI, or The Movie Database (TMDb).
   - Register for an API key if required.
2. **Integrate with Your Application:**
   - Create an endpoint (e.g., `GET /weather/:city`) that fetches data from the third-party API.
   - Parse and display the response in a user-friendly format.

- For example, for a weather API:

```
json
CopyEdit
{
  "city": "New York",
  "temperature": "15°C",
  "condition": "Cloudy"
}
```

3. **Enhance with Parameters:**
   - Allow users to specify additional parameters like:
     - City or genre for filtering.
     - Date range for fetching news.

Part 3: API Documentation

1. **Document Your API:**
   - Use a tool like Swagger or Postman to create interactive API documentation.
   - Include the following details:
     - Endpoint URL and HTTP methods.
     - Required parameters.
     - Sample request and response bodies.
   - Example:

```
vbnet
CopyEdit
Endpoint: GET /books
Description: Fetch all books.
Response:
[
  {
    "id": "1",
    "title": "1984",
    "author": "George Orwell",
    "year": 1949,
    "genre": "Dystopian"
  }
]
```

Part 4: Deployment

1. **Host the API:**
   - Deploy your API using platforms like Heroku, Render, or Vercel.
   - Provide the live link for accessing the API.
2. **Testing:**
   - Test your endpoints using Postman or similar tools.

- Include screenshots of successful API responses in your submission.

Bonus Tasks (Optional):

- Implement **authentication** for your API using JWT or API keys.
- Add **rate limiting** to protect your API from abuse.
- Cache third-party API responses for improved performance.
- Create a **front-end application** to consume your API (e.g., a simple weather dashboard).

Submission Guidelines:

1. **Deliverables:**
   - Source code of your API.
   - API documentation (Postman collection or Swagger JSON).
   - Screenshots of API responses (both your API and third-party API).
   - Deployment link (if hosted).
2. **Submission:**
   - Upload a ZIP file with all deliverables on the Learning Management System (Moodle) under Assignment 2.
   - Ensure your documentation and setup instructions are included.
3. **Deadline:** On moodle.

Evaluation Criteria:

Criterion	Weight
API Design and Functionality	40%
Well-structured endpoints, CRUD operations, and validation.	
Third-Party API Integration	30%
Fetching and displaying external data.	
Documentation and Deployment	20%
Detailed and clear documentation with successful deployment.	
Bonus Features	10%
Advanced functionalities or front-end integration.	