Q1. Build a RESTful Service for a Library Borrowing System

Create a RESTful web service for managing Books, Members, and Borrowing Records.

The service must support the following:

1. Books
    - Add a new book
    - Update book details
    - List all books
    - Retrieve a specific book by ID
2. Members
    - Register a new member
    - Update member details
    - List all members
    - Retrieve a specific member by ID
3. Borrowing
    - A member can borrow a book if it is available
    - A book becomes unavailable while borrowed
    - A member can return a borrowed book
    - The service must record the borrow date and return date
4. The system must persist data in a relational database using EF Core.
5. When listing books, the API must support optional filtering by availability and by author.
6. Provide an endpoint that returns the complete borrowing history for a given member.
7. Include basic validation such as preventing multiple active borrows of the same book and preventing return of a book that is not currently borrowed.

Deliverables:

- Code
- Short explanation of architecture
- Instructions for running the service

Q2. Design a database schema for an online course platform.

Requirements:

- A Course contains one or more Modules.
- A Student can enroll in many Courses.
- An Enrollment must track the enrollment date.
- A Module must belong to exactly one Course.

Deliverables:

1. Draw or describe the tables with fields and relationships.
2. Provide SQL queries for the following:
    - List all courses with their module counts.
    - List all students enrolled in a given course.
    - List the top 3 courses with the highest number of enrollments.

    - For a given student, list all courses they are enrolled in along with the enrollment date.

Q3. Rate Limiter

Implement an in-memory rate limiter that restricts each user to X requests per minute.

Requirements:

- Support multiple users.
- Track requests per user.
- Enforce limits accurately for rolling one-minute windows.
- The implementation must be concurrency-safe.
- Provide an API endpoint that uses the rate limiter and returns whether the request is allowed or blocked.

Deliverables:

- Code
- Explanation of design and trade-offs

Q4. Implement a function that reads a list of URLs from a file and fetches the contents of all URLs concurrently.

The function must:

- Use asynchronous I/O for all network requests
- Limit the maximum number of concurrent requests to N
- Write the result of each request to an output file along with the URL and the HTTP status code
- Handle failures for individual URLs without stopping the entire processing