# Online Course Platform

## Objective:

Build a RESTful API for an Online Course Platform where instructors can create courses, and students can enroll in and review courses. The application should include the following features:

1. **User Authentication with JWT Refresh Tokens:**
    - Implement user registration and login using email and password.
    - Implement JWT-based authentication with access and refresh tokens.
    - Provide an endpoint for refreshing tokens.
2. **MongoDB Integration:**
    - Use MongoDB to store users, courses, and enrollments.
    - Implement relationships where users can create and enroll in multiple courses.
3. **RESTful API Endpoints:**
    - **User Endpoints:**
        - `POST /register`: Register a new user.
        - `POST /login`: Log in an existing user.
        - `POST /refresh-token`: Refresh the JWT using the refresh token.
        - `POST /login/google`: Log in using Google.
        - `POST /login/facebook`: Log in using Facebook.
    - **Course Endpoints:**
        - `GET /courses`: Get all courses.
        - `GET /courses/:id`: Get a specific course by ID.
        - `POST /courses`: Create a new course (authenticated as an instructor).

- `PUT /courses/:id`: Update a course by ID (authenticated and authorized as an instructor).
- `DELETE /courses/:id`: Delete a course by ID (authenticated and authorized as an instructor).
- `POST /courses/:id/enroll`: Enroll in a course (authenticated).
- `POST /courses/:id/review`: Add a review to a course (authenticated).
- `GET /users/:id/courses`: Get all courses created or enrolled in by a specific user.

4. **File Uploads to Amazon S3:**
   - Allow instructors to upload course materials (e.g., videos, PDFs) to S3.
   - Store uploaded files in Amazon S3 and save the file URLs in MongoDB.
5. **Social Media Login Integration:**
   - Implement login functionality using Google and Facebook OAuth.
   - After successful social media login, generate JWT tokens for the user.

## Detailed Requirements:

1. **Authentication:**
   - Use `jsonwebtoken` to handle JWTs.
   - Store refresh tokens securely in MongoDB.
   - Implement middleware to protect routes that require authentication.
2. **MongoDB:**
   - Use Mongoose to define schemas for `User`, `Course`, and `Enrollment`.
   - Ensure that each `Course` is linked to an `Instructor` (User) and that `Enrollments` are linked to both `Courses` and `Students` (Users).
3. **Amazon S3:**
   - Use the `aws-sdk` or `@aws-sdk/client-s3` to upload files to an S3 bucket.
   - Ensure that only authenticated instructors can upload course materials.

- Store the S3 file URL in the `Course` document in MongoDB.

4. **Social Media Authentication:**
   - Use `passport.js` with `passport-google-oauth20` and `passport-facebook` strategies.
   - Handle OAuth callbacks to retrieve user data and log in or register the user.
   - Issue JWT tokens after successful social media login.

## Deliverables:

- Complete Node.js project with all the above features implemented.
- Postman collection demonstrating all API endpoints.
- Documentation on how to set up the environment, including MongoDB and S3 configuration, and OAuth credentials for Google and Facebook.

## Bonus:

- Implement a feature to track course progress for students.
- Add course search functionality with filters for category, difficulty level, or instructor.