Project: ProgrammingTutorForKids Platform with ASP.NET Core

Objective:

Build a scalable, interactive online learning platform designed to help kids learn programming concepts through tutorials, quizzes, and achievements. The platform will include an admin panel for managing users, tutorials, progress, and achievements. It will follow NTier architecture and leverage design patterns like Repository Pattern and Unit of Work for clean separation of concerns and efficient development.

Design Patterns to Use:

1. **NTier Architecture:** Separate concerns into Presentation (UI), Business Logic (Services), and Data Access (Repository/Entity Framework) layers.
2. **Repository Pattern:** Abstract database operations, providing a clean API for querying and persisting data.
3. **Unit of Work Pattern:** Ensure that multiple changes (such as creating a user, adding progress, assigning achievements) are grouped into a single transaction.
4. **Dependency Injection:** Inject services (repositories, business logic) into controllers for loose coupling and improved testability.

Technologies to Use:

- **ASP.NET Core MVC**
- **Entity Framework Core**
- **ASP.NET Identity (Authentication)**
- **JQuery, Bootstrap**
- **SignalR (for real-time notifications like achievement unlocks)**
- **Microsoft Azure (Deployment)**
- **DataTables (for Admin Dashboard)**
- **Toastr.js (for UI notifications)**

Week 1: Initial Setup, User Authentication, and Tutorial Listings

NTier Architecture Setup:

- **Presentation Layer:** ProgrammingTutor.Web - User-facing ASP.NET MVC application for displaying views, controllers, and client-side logic.
- **Business Logic Layer:** ProgrammingTutor.Services - Handles business rules, tutorial progress, and achievement operations.
- **Data Access Layer:** ProgrammingTutor.Data - Implements Repository and Unit of Work patterns for database interactions (tutorials, users, achievements, feedback).

Database Design:

- Define database schema using **Entity Framework Core**, covering tables like Tutorials, Users, Progress, Achievements, and Feedback.
- Implement the **DbContext** as the Unit of Work, and create repositories for data access (e.g., TutorialRepository, UserRepository, ProgressRepository).

User Authentication:

- Set up **ASP.NET Identity** for user registration, login, and role management (Admin, Student).

Tutorial Listings:

- Implement the tutorial listing page with search and filtering functionalities (based on categories, difficulty levels, etc.).

Deliverables:

- Complete NTier Architecture setup.
- User authentication integrated with **ASP.NET Identity**.
- Tutorial listing page with search and filtering.
- **Entity Framework Core** database design with repositories and Unit of Work pattern.

Week 2: Progress Tracking, Admin Panel, and Role-Based Access Control

Progress Tracking:

- Implement a **progress system** where users (kids) can track their progress through tutorials.
- Use **Session State** to temporarily save tutorial progress and store it in the database upon completion.

Admin Dashboard:

- Build an admin panel to manage tutorials, users, progress, and achievements. Use **DataTables** for sorting, filtering, and pagination.

- Use the **Repository Pattern** to ensure all updates (tutorials, users) go through the **Unit of Work**.

Role-Based Access Control (RBAC):

- Define roles for **Admin** and **Student** using **ASP.NET Identity**. Restrict access to the admin panel and allow students to only access tutorials and their progress.

Deliverables:

- Fully functional **progress tracking system**.
- **Admin dashboard** to manage tutorials, users, and progress.
- Role-based access control with restricted features based on roles.

Week 3: Quiz System, Achievements, and Real-Time Notifications

Quiz System:

- Implement a quiz system for each tutorial. After completing a tutorial, the user must pass a quiz to move forward. Store quiz results using the **Repository Pattern**.

Achievement System:

- Add an achievement system that tracks when users complete certain milestones (e.g., finish five tutorials, complete a quiz perfectly).
- **Real-Time Notifications:** Use **SignalR** to send real-time notifications when users unlock new achievements.

Deliverables:

- **Quiz system** integrated with tutorials.
- **Achievement system** with unlockable badges.
- **Real-time notifications** for achievements using **SignalR**.

Week 4: UI Enhancements, Final Testing, and Deployment

UI Enhancements:

- Refine the UI using **Bootstrap** for a responsive, modern design. Ensure it's intuitive for children.
- Use **Toastr.js** for smooth, non-intrusive notifications (e.g., tutorial completion, achievement unlock).

Admin Dashboard with DataTables:

- Integrate **DataTables** into the admin dashboard to allow for better management of tutorials, progress, and users (sorting, filtering, pagination).

Final Testing and Deployment:

- Perform thorough testing of the entire system, including tutorial navigation, progress tracking, and admin operations.
- Deploy the project to **Microsoft Azure** or another hosting provider for live usage.

Deliverables:

- Responsive and polished UI for both users and admin panel.
- **DataTables** integrated for managing tutorials and users.
- Live deployment of the **ProgrammingTutorForKids** platform.
- Complete documentation, including architecture, setup guide, and user manual.

Optional Additions (Stretch Goals):

1. **Gamification:**
    - Add a **leaderboard** for achievements or quiz scores to encourage healthy competition among users.
2. **Live Code Editor:**
    - Implement a simple **code editor** within the tutorials where users can practice writing code and get instant feedback.
3. **Subscription Plans:**
    - Implement subscription tiers (free, premium) to allow students to unlock additional tutorials, quizzes, and features.
4. **Parent Dashboard:**
    - Allow parents to sign up and monitor their child's progress, see achievements, and encourage learning.
5. **API Layer:**
    - Build a REST API that allows external developers to build apps around the platform's data (tutorials, progress).