**Introduction to Razor Pages with Entity Framework : Workshop-1**

Welcome to the dynamic world of web development using Razor Pages and Entity Framework (EF). Razor Pages is a lightweight web framework in ASP.NET Core that enables you to build web applications with simplicity and flexibility.

**Razor Pages:**

- Emphasizes code simplicity.

- Utilizes a page-focused approach for building UI.

- Integrates seamlessly with ASP.NET Core.

**Entity Framework:**

- A powerful Object-Relational Mapping (ORM) framework.

- Simplifies database interactions in your application.

- Provides a convenient way to work with databases using C#.

**Why Combine Razor Pages with EF:**

- Rapid development with minimal boilerplate code.

- Seamless integration for building data-driven applications.

- Simplified syntax and structure for managing UI and data logic.

This powerful combination empowers developers to create efficient and scalable web applications with ease. Whether you are a beginner or an experienced developer, Razor Pages with EF offers a straightforward approach to building modern web solutions. Dive in and explore the endless possibilities of web development!

**Dear students,** for your hands-on learning experience with Razor Pages and Entity Framework, please create the following models with their respective primary and foreign key relationships:

**1. Applications:**

*Columns:*

- Id (uniqueidentifier) [Primary Key]

- UserId (uniqueidentifier) [Foreign Key: Users(Id)]

- JobId (uniqueidentifier) [Foreign Key: Jobs(Id)]

- CompanyId (uniqueidentifier) [Foreign Key: Companies(Id)]

- AppliedDate (date)

- Status (varchar(50))

**2. Companies:**

*Columns:*

- Id (uniqueidentifier) [Primary Key]

- Name (varchar(100))

- Email (varchar(50)) [Unique Constraint]

- Website (varchar(50))

- Phone (varchar(50))

- Logo (varchar(50))

- About (varchar(100))

- Vision (varchar(100))

- Mission (varchar(100))

- Location (varchar(50))

- Address (varchar(50))

- Status (varchar(50))

- CreatedDate (date)

**3. Interviews:**

*Columns:*

- Id (uniqueidentifier) [Primary Key]

- CompanyId (uniqueidentifier) [Foreign Key: Companies(Id)]

- JobId (uniqueidentifier) [Foreign Key: Jobs(Id)]

- JobseekerId (uniqueidentifier) [Foreign Key: Users(Id)]

- Date (date)

- Time (time(7))

- Location (varchar(50))

- Status (varchar(50))

- CreatedBy (uniqueidentifier) [Foreign Key: Users(Id)]

- CreatedDate (date)

*Constraints:*

- Foreign Key: CompanyId references Companies(Id)

- Foreign Key: JobId references Jobs(Id)

- Foreign Key: JobseekerId references Users(Id)

- Foreign Key: CreatedBy references Users(Id)

**4. Jobs:**

*Columns:*

- Id (uniqueidentifier) [Primary Key]

- Title (varchar(50))

- Description (varchar(50))

- Location (varchar(50))

- Experience (varchar(50))

- TypeOfWorkPlace (varchar(50))

- Salary (varchar(50))

- Responsibilities (varchar(50))

- JobType (varchar(50))

- VacanciesCount (int)

- AppliedCount (int)

- Status (varchar(50))

- CompanyId (uniqueidentifier) [Foreign Key: Companies(Id)]

- CreatedDate (date)

- CreatedBy (uniqueidentifier) [Foreign Key: Users(Id)]

*Constraints:*

- Foreign Key: CompanyId references Companies(Id)

- Foreign Key: CreatedBy references Users(Id)

**5. Qualifications:**

*Columns:*

- Id (uniqueidentifier) [Primary Key]

- UserId (uniqueidentifier) [Foreign Key: Users(Id)]

- Title (varchar(50))

- Mark (varchar(50))

- YearOfPassout (varchar(50))

- University (varchar(50))

- Status (varchar(50))

*Constraints:*

- Foreign Key: UserId references Users(Id)

**6. Skills:**

*Columns:*

- Id (uniqueidentifier) [Primary Key]

- UserId (uniqueidentifier) [Foreign Key: Users(Id)]

- Title (varchar(50))

- Status (varchar(50))

*Constraints:*

- Foreign Key: UserId references Users(Id)

**7. Users:**

*Columns:*

- Id (uniqueidentifier) [Primary Key]

- FirstName (varchar(50))

- LastName (varchar(50))

- Email (varchar(50)) [Unique Constraint]

- Gender (varchar(50))

- Location (varchar(50))

- Phone (varchar(50))

- Password (varchar(50))

- Role (varchar(50))

- About (varchar(50))

- Designation (varchar(50))

- CompanyId (uniqueidentifier) [Foreign Key: Companies(Id)]

- Status (varchar(50))

- Image (varchar(50))

- CreatedDate (date)

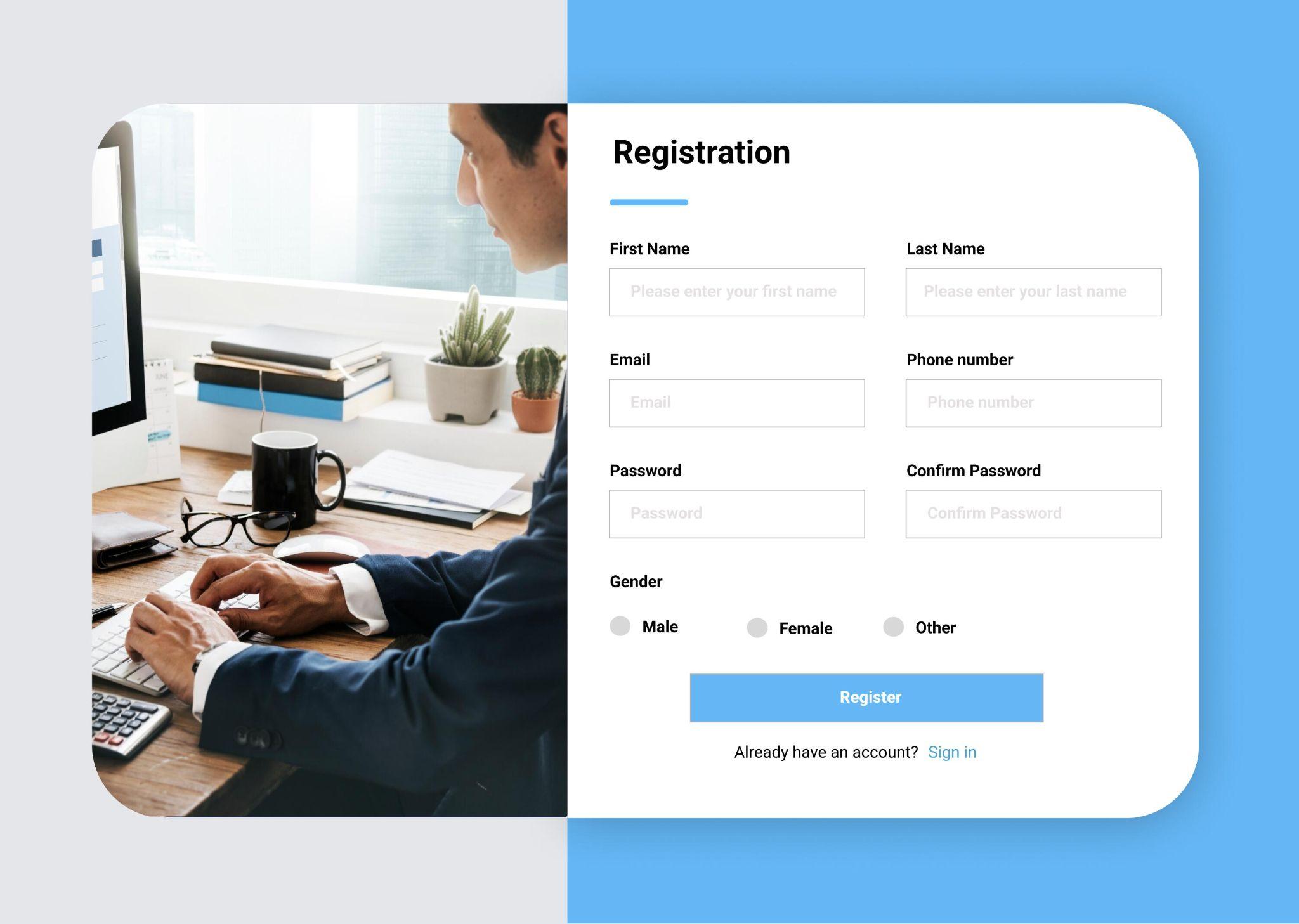
*Constraints:*

- Foreign Key: CompanyId references Companies(Id)

Feel free to dive in and create these models to strengthen your understanding of building database structures for web applications.

**Task: Create a Registration Screen**

Your next challenge is to implement a registration screen using Razor Pages and Entity Framework. Follow these steps:



1. Design a registration form with fields for a user's first name, last name, email, password, and any other relevant information.

2. Implement the backend logic to handle form submissions and store user data in the database.

3. Ensure that the registration process includes proper validation for user input.

4. After a user successfully registers, check the database to verify that the user's information has been correctly stored.

Feel free to leverage the Entity Framework to interact with the database. This exercise will not only test your Razor Pages skills but also reinforce your understanding of database interactions.

**Happy coding 😀**