Dev Incubator Management System

Admin role

Actions	Admin
Watch the **Member's Manage Grid**	+
Add, edit, and delete a member on **Member's Manage Grid**	+
Watch the **Member's Progress grid**	+
Watch the **Tasks Manage Grid**	+
Add, edit, and delete a **New task**	+
Watch the **Member's Task Manage grid**	+
Set the **Member task's state** as Success or Fail	+
Watch the **Subtasks Manage Grid** of the current **Task**	-
Add, edit, and delete a **Subtasks** of the current Task	-

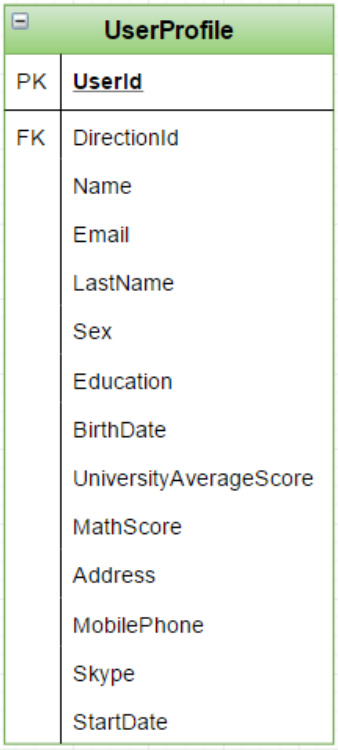
1st week:

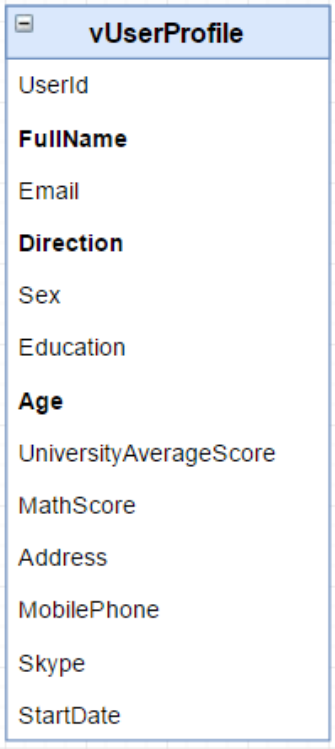
Death valley rewriting. You should rewrite you first project "Death Valley" following n-layer architecture pattern: database layer, business layer, presentational layer. Here is a detailed plan about what you should do. Additionally you should write unit tests to repositories and controllers. It will be super if you add also 2-3 integration tests. For the task you should create repository on github. During this week go deeper in learning of version control system like git. You can follow one of these sources githowto / git-scm / try-github to improve your skills for example and use one of the following GUI Source Tree or GitKraken

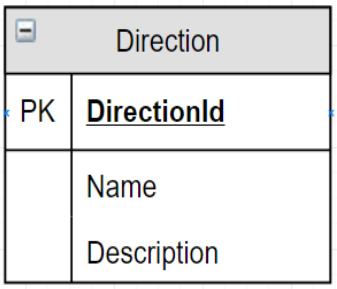
2nd week:

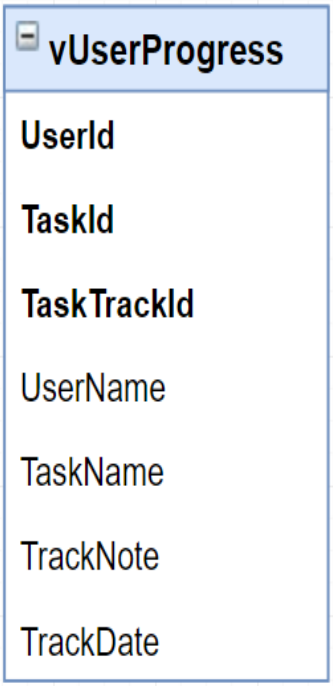
You should create the **UserProfile, Direction** tables, **vUserProfile & vUserProgers** views with all needed constrains (remember! you must use only the table level constraints not the column one).

Also you need to create the **DeleteUser** procedure, which gets the **UserId** as an input parameter and delete the **User** entry accordingly.


UserProfile
PK
UserId
FK
DirectionId
Name
Email
LastName
Sex
Education
BirthDate
UniversityAverageScore
MathScore
Address
MobilePhone
Skype
StartDate


vUserProfile
UserId
FullName
Email
Direction
Sex
Education
Age
UniversityAverageScore
MathScore
Address
MobilePhone
Skype
StartDate


Direction
PK
DirectionId
Name
Description


vUserProgress
UserId
TaskId
TaskTrackId
UserName
TaskName
TrackNote
TrackDate

Please, pay your attention to the **FullName,** and the **Age** fields, which are auto-calculative fields. The **Direction** field is taken via join with the **Direction** table.

Important note. All Foreign Keys constraints must be as **ON CASCADE DETELE / UPDATE.**

After creating all needed data base objects, you should update you model from database.

Remember! **The SQL views don't have PKs**, but Entity Framework requires the ones. You need to set the PK for your Views manually. After that you should implement the **Repositories** for your **Entities** and add them to the **UnitOfWork**.

3rd week:

Learn how to write unit-tests properly and write unit-tests for you repositories. Read about different approaches, **rule of "AAA"**, **right naming** and **code convention** relating unit testing.

Look at the following tools.
For test coverage of application we will use **dotCover**. Another popular libraries:

- **OpenCover**
- **ncrunch**
- **etc**

For unit tests writing we will use **NUnit**. Another popular libraries:

- **xUnit**
- **MSTest**

For fake/mock/stub we will use **moq**. Another popular libraries:

- **nsubstitute**
- **fakeiteasy**
- **Etc**

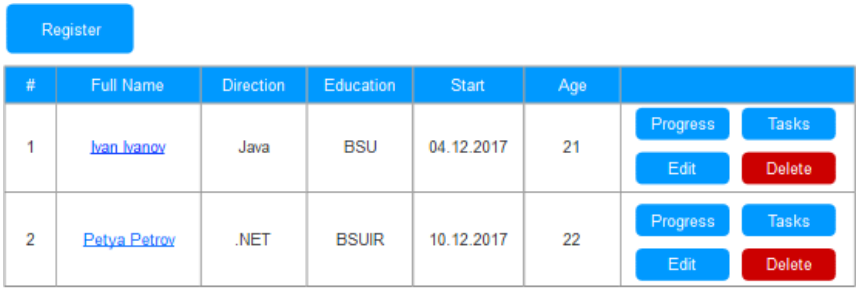
Moq library we will use later.

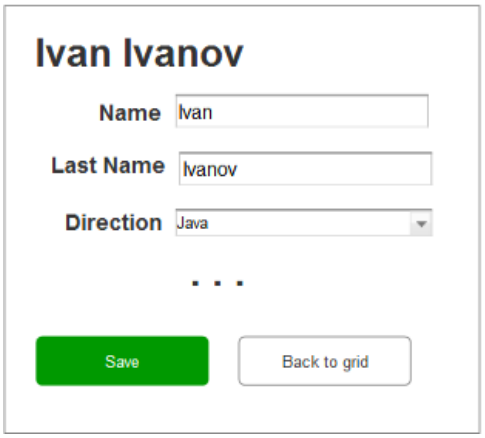
4th week:

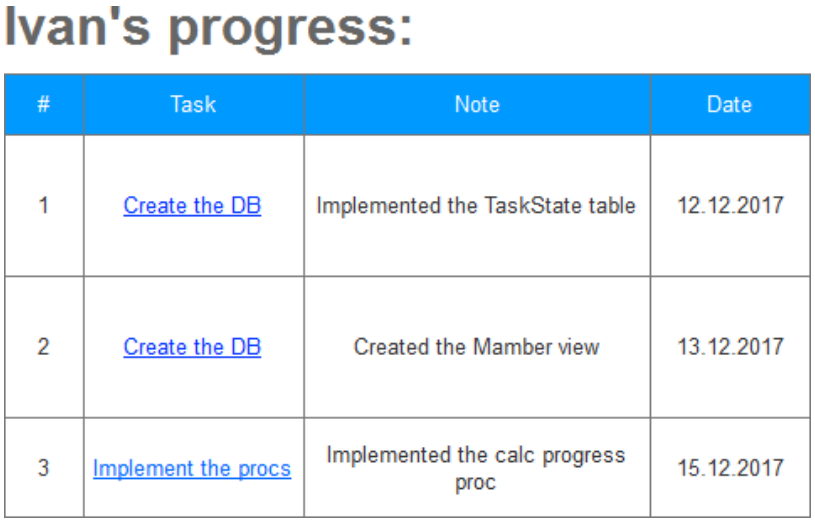
Members Manage Grid page & **Member's Progress Grid page** are the simple

grids with typical CRUD actions. You need to implement the Data Transfer Objects in you business logic layer.

Important note. When a new member is registered, one piece of data goes to Identity data base and another to the application data base. You should duplicate all needed data and when user login, you need to save the **UserId** in the Current Session object via retrieving the **UserId** by the Email from Identity database.







5th week:

Implement controllers with all needed actions, ViewModels, and Views. Look at existing SampleController, and do the same. The Create/Edit pages look similar, but Detail page needs to be as a read-only version. The Progress and Tasks buttons are the links to Progress grid and Member's Tasks grid. If this grids aren't implemented yet simple create this buttons without particular paths. When your partners make their piece of work add all needed links to this buttons.

6th week:

Complete your presentation layer. add nice styles. If you will work with front-end partner, provide him all routes and data that he can get from them accordingly.

7th week:

Put all parts together, add all logic related you main role. Write unit-tests for you controllers, business logic and 2 integration tests.

8th week:

Test your area of responsibility very intensively and fix all bugs. After this week you should provide a demo with description of business logic of your part in English.