# DevIncubator Management System

# Development Track #3

**1ˢᵗ week**: You should implement the **UserProfileView**. Use the **Task** table as base table. To get **UserId** and **Sate** join the **Task** table with **UserTask** table and **TaskState** table. As you can see, you have a lot of dependencies of your partners. If they don't implement all needed tables, please rush them!

```
┌──────────────────────────┐
│ ⊟    vUserTask           │
├──────────────────────────┤
│ UserId                   │
│                          │
│ TaskId                   │
│                          │
│ TaskName                 │
│                          │
│ Description              │
│                          │
│ StartDate                │
│                          │
│ DeadlineDate             │
│                          │
│ State                    │
└──────────────────────────┘
```

Also you should implement the **TaskState** table. It's a typical constant values table. You need to add three constant values to the **SeedData** which will be inserted all time in the publish process. They are **Active**, **Success**, **Fail.**

```
┌──────┬───────────────────┐
│ ⊟        TaskState       │
├──────┼───────────────────┤
│ PK   │ StateId           │
├──────┼───────────────────┤
│      │ StateName         │
└──────┴───────────────────┘
```

And you need to create the **SetUserTaskAsSuccess** and the **SetUserTaskAsFail** procedures, which take the UserId and TaskId as input parameters and change the **StateId** in the **UserTask** table to **Success** or **Fail** respectively.

**2<sup>nd</sup> week:** After creating all needed data base objects, you should refresh the Application Entity model by use the LLBLGen (please, look the Development Workflow). Remember! The SQL Views don't have PKs, but Entity Framework requires the ones. You need to set the PK for your View manually. After that you should implement the Repositories for your Entries and add this ones to the UnitOfWork.

**3<sup>rd</sup> week:** Please, look your piece of the UI very attentively. As you can see, this is a simple grid with typical CRUD actions. You need to implement the UserTask Data Transfer Object.

Hi, dear Ivan! This is your current tasks:

| # | Name | Start | Deadline | Status | | (Available only for Admin) |
|---|------|-------|----------|--------|-------|------------|
| 1 | Create the DB | 06.12.2017 | 12.12.2017 | Sucess | Track | Success / Fail |
| 2 | Implement the procs | 13.12.2017 | 20.12.2017 | Active | Track | Success / Fail |

**4<sup>th</sup> and 5<sup>th</sup> weeks:** You should implement the **UserTaskController** 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 **Track** is the links to Create Task Track page. If this page hasn't been implemented yet simple create this button without particular path. When your partner make their piece of work, you will add all needed links to this button.

**6<sup>th</sup> week:** Test your area of responsibility very intensively and fix all bugs which will you get. Your part of application doesn't have a lot of dependencies from another parts, and you can work very freely. After this week you should provide a video demo with description of business logic of your part on English, of course.