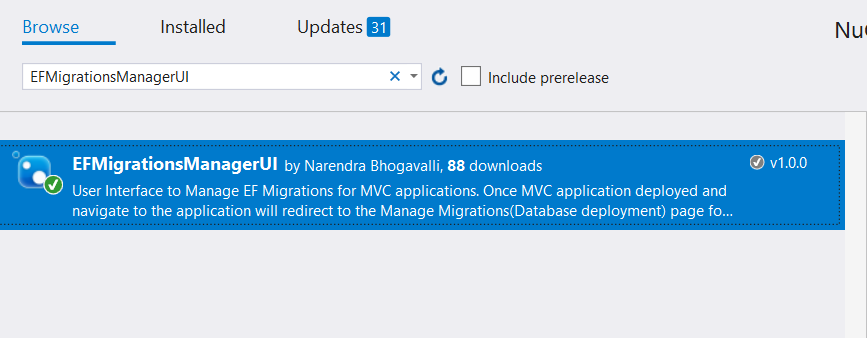
**Install EFMigrationsManagerUI:**

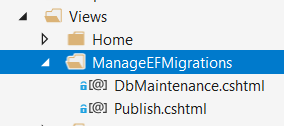
1. In Visual Studio, Open the target MVC project to enable Manage migrations user interface.
   * Right click on the MVC Project References and choose **Manage Nuget Packages.**
   * Search and add the EFMigrationsManagerUI Nuget package.



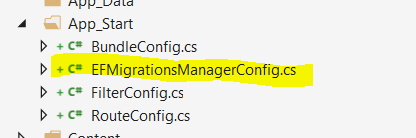
This Nuget package have dependencies with the following packages

* + EntityFramework
  + Microsoft.AspNet.Mvc
  + WebActivatorEx

1. Above Nuget package will add the below files in target project.
   * EFMigrationsManagerUI assembly reference.
   * In Controllers folder, Add EFMigrationsManagerController class.
   * In Views Folder, Adds EFMigrationsManager folder with below views
     1. Publish
     2. DbMaintenance



* + Under App\_Start folder, Adds EFMigrationsManagerConfig.cs file



1. AppSettings, Adds the below App setting entry
   * <add key="EFMigrationsManagerUI:AuthorizedUsers" value="Comma seperated admin user names to deploy EF migrations" />

**Configure the Application:**

1. Install the “EFMigrationsManagerUI” nuget package
2. Open the “EFMigrationsManagerConfig” class under App\_Start folder
   * Uncomment below line and pass the Entity Framework Configuration instance as parameter.

//EFMigrationsManagerSettings.SetEFConfiguration(new EFConfiguration());

* + 1. Search for Class in solution (Usually in models or data projects) that inherits from DbMigrationsConfiguration class.
    2. Replace EFConfiguration with Class found in above step.

1. Open the web.config and navigate to the appSettings section
   * Update the user names in below app setting section.

<add key="EFMigrationsManager:AuthorizedUsers" value="Comma separated admin user names to deploy EF migrations" />

Note:

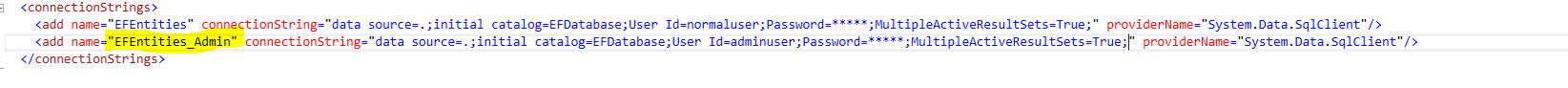
1. For multiple admins, enter comma separated **identity name**s.
2. For windows authentication, enter name as domain\username
3. For Forms authentication, enter name as [username@domain.com](mailto:username@domain.com) or [firstname.lastname@domain.com](mailto:firstname.lastname@domain.com)

**Advanced Configuration:**

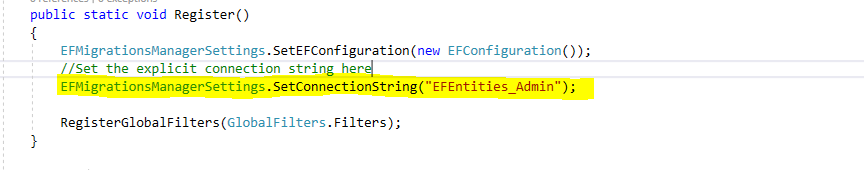
1. Recommendation: Its recommend using different connection string for EFMigrationsManager. Most of the scenarios, Entity Framework migrations (to update database) need permission for schema changes (Create/Alter/Drop table permissions).

Whereas applications connection string doesn’t need those admin permissions and it’s not recommendable to provider admin permissions.

* Create another connection string like below



* + Open the “EFMigrationsManagerConfig” class under App\_Start folder and navigate to the **Register** method. Call the EFMigrationsManagerSettings.SetConnectionString method with admin connection string. Please check the below image for quick reference.



Now application page will use connection string with read/write permissions based on the application requirement and EFMigrationManagerUI(Database deployment pages) will use admin connection string.

1. If any specific application page is not required to check Migrations are up to date or not, then use the below attribute at Controller/Action level



Example: Application Error pages. If application is throwing some database exception then application will redirect default error pages. If again error pages are hitting the database then application might throw another error in error pages. These kinds of situations skip the verification by using above attribute.

1. Disable auto detection of pending migrations and redirect to the deployment page:

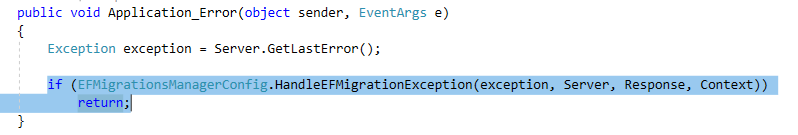
By default, when install nuget package, AutoDetect feature is enabled. To disable that feature, remove below line from EFMigrationsManagerConfig class.



1. Alternative approach to Auto Detection: Default configuration is with MVC action filter on above step. MVC action filter will run for every MVC page request. To avoid checking the migrations are up to date on every page request,
   * Follow the above step. (Remove the below line from EFMigrationsManagerConfig class)



* + Call the below highlighted method in Application\_Error. This method will check if exception is related to the entity framework and EF migrations context is mismatched with the database then redirect to the database deployment page.



Note: Entity framework will throw the Exceptions on database call if context is mismatched. If no database call on any specific pages then user will see the requested page instead of database deployment/maintenance pages even database is not up to date.

**Examples/Demos:**

Created below samples for MVC based applications and hosted in GitHib repository.

1. MVC application with Windows authentication:

Follow the below steps to setup the working sample.

* + Download/Clone the source code from the below GitHub URL

<https://github.com/naren-b/EFMigrationsManagerUI_Sample_WindowsAuthentication>

* + Open the project in visual studio
  + Set ‘Sample\_WindowsAuthentication’ as startup project.
  + Open web.config file and update the below appsetting entry with admin usernames.

<add key="EFMigrationsManager:AuthorizedUsers" value="domain\username" />

Note: For multiple admins, enter comma separated names.

* + Update the below connection string if required to test on different database.

<connectionStrings>

<add name="EFEntities" connectionString="data source=.;initial catalog=EFDatabase;Integrated Security=SSPI;MultipleActiveResultSets=True;App=EntityFramework" providerName="System.Data.SqlClient" />

</connectionStrings>

1. MVC application with Forms based authentication

Created sample Web application integration with azure active directory.

To know more about the Azure Active directory, please follow the below link.

<https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-integrating-applications>

Follow the below steps to setup the working sample.

* + Download/Clone the source code from the below GitHub URL

<https://github.com/naren-b/EFMigrationsManagerUI_Sample_FormsAuthentication>

* + Open the project in visual studio
  + Set ‘FormsAuthentication’ project as startup.
  + Open web.config file and update the below appsetting entry with admin usernames.

<add key="EFMigrationsManager:AuthorizedUsers" value="username@domain.com" />

Note: For multiple admins, enter comma separated names.

* + Update the below connection string if required to test on different database.

<connectionStrings>

<add name="EFEntities" connectionString="data source=.;initial catalog=EFDatabase;Integrated Security=SSPI;MultipleActiveResultSets=True;App=EntityFramework" providerName="System.Data.SqlClient" />

</connectionStrings>

* + Update the below azure active directory integration related settings

<add key="ida:ClientId" value="" />

<add key="ida:ClientSecret" value="" />

<add key="ida:Domain" value="" />

<add key="ida:TenantId" value="" />

<add key="ida:PostLogoutRedirectUri" value="" />

<add key="EFMigrationsManager:AuthorizedUsers" value="https://localhost:44351/" />