# YouTube Playlists

You are assigned to design and implement an application for **playlists containing YouTube videos**.

Users should be able to register into the application and create playlists.

**Users** have profiles with information about them – **first** and **last name**, **e-mail address (username = email)**, a **profile image** (URL) and links to their **Facebook** and **YouTube** accounts. The profile image is optional and should have a default one, if such is not provided. Facebook and YouTube accounts are optional too. Additionally users have **rating** which is accumulated from their playlists.

Playlists can be created only by registered users. Each playlist should have **title**, **description**, **video URLs** (separate entities), **category**, **creator** and **creation date**.

Playlist **categories** has **Name** and are added and edited only by administrators.

Each playlist can receive **rating** from 1 to 5. The total *average* rating is shown for each playlist. Additionally the users have rating – the average from their playlists.

Anonymous users can only browse the home page and view all the playlist details pages.

The system should be implemented using **ASP.NET Web Forms**. You are **not allowed** to use any external controls that are not part of the standard Web Forms suit. Your own user controls are allowed.

## Data Layer (21 points)

Use **Entity Framework** as ORM engine and **MS SQL Server** as database storage engine.

Your project should run after **"copy/paste" deployment** and after changing the connection string in the Web.config file with an appropriate value. For convenience use “*DefaultConnection*” as the name for your connection string.

You should use **code-first approach** to access your data from Entity Framework.

1 point

Design a simple data layer to hold **users**, **playlists, categories, ratings** and **videos**.

* **User** has mandatory **email**, **password, first name, last name** and optional **image URL**, **Facebook URL** and **YouTube URL**.
  + The password should be stored encrypted in the database.
* **Category** has **name** (unique and mandatory).
* **Playlist** has mandatory **title, description, category, creator** (user)and **creation date**.
* **Video** has **URL** and is linked to a playlist (one-to-many relationship).
* **Rating** has **UserId** (voter), **PlaylistId** and **Value** (from 1 to 5).

6 points

Fill the database with **sample data**.

* 1 role named “**Administator**”
* **1 administrator** with email (username) “**admin@site.com**” and password “**admin**”.
* **5 users** with email (username) “**user1@site.com**” and password “**user1**”, another with email (username) “**user2@site.com**” and password ”**user2**” and so on.
* At least **30** random **categories**
* At least **10 playlists** with **~3 real YouTube videos** each and **~5** random **ratings** each

6 points

Use the **ASP.NET Identity** system to keep the **users**, **roles** and their encrypted passwords.

1 point

**Data layer abstraction** – the data layer should be abstracted by using services and/or data repositories where applicable.

You are not required to use Ninject nor any other dependency container. You can create instances with the “new” keyword but the data queries **should not** be done through *DbContext* directly.

7 points

## Common Features (15 points)

* **Master Page** – design an ASP.NET Web Forms Master Pages to reuse the common page elements like headers, footers and navigation in all other pages in the project.

2 points

* In the **navigation (menu)** when user is not authorized add link only to all public parts (home page, register and login).

For registered users add all other links (create playlist, list of all playlists).

For administrators add all links for registered users and a link to the category administration page.

2 points

* **Configure the ASP.NET Identity System** to enable user management functionality (register / login / logout).
  + Disable password requirements for non-letter characters, lower case and upper case letters. The minimum required length of the password should be 5. “*admin*” should be a valid password.

2 points

* **Register user** – by email, password, first name and last name the system should be able to register a new user in the system. The fields for the image URL, Facebook URL and YouTube URL should be optional in the registration form. After successful registration, the user should be redirected to the home page.

2 points

* **Login user** – by email and password the system should be able to login an existing user. After a successful login, the user should be redirected to the home page.

1 point

* **Logout** – successfully logged in user should be able to logout from the system. After logout the user should be redirected to the home page.

1 point

* **Error handling** – in case of errors (e.g. DB connection lost, incorrect request, 404, etc.), an appropriate error message should be displayed. You are free to decide how exactly.

2 points

* **User interface (UI)** – the user interface should be usable enough. It is not needed to be beautiful. You may use Bootstrap if you want.

3 points

## Public Area (13 points)

* **Home page** – at the application start page display the top 10 most popular playlists (sorted by rating, descending) with their names, their authors and their ratings. Link each playlist to its details page. Link category to the “*list playlists by category*” page to visualize all playlists in a given category.

4 points

* + Load the data using AJAX.

2 points

* **Playlist details** page. Show information of a given playlist (title, description, ratings, category name, creator names (first + last), creation date, etc.) and list all videos as links directly to YouTube.

5 points

* + Bonus for those who **embed** the YouTube videos instead of linking them.

2 points

## User Area (41 points)

* **Profile page** – registered users (after login) should be able to see their profile info (email, names, image, URLs, etc.) and a list of all playlists created by them. Link each playlist to its details page.

4 points

* **Edit profile page** - registered users (after login) should be able change their information (first name, last name, image URL, Facebook URL, YouTube URL (without password and email)). Add validations.

4 points

* **List** the playlists as a table (available only for registered users). List the names of the playlists, their category name, the names of their creator and the number of videos in them.

4 points

* + Provide a way to **sort** the playlist by **date**, **name** or by **rating** (in ascending/descending order)

3 points

* + Use server side **paging** (page size is 10)

3 points

* + **Search** playlists by their name and description (text search) and list results. Search is available only for registered users. You may put the search form in the home page, the page with all playlists or anywhere in the navigation.

4 points

* + **List playlists by category** – when clicking on category names a new page with list of all playlists in that category should be visualized.

4 points

It would be clever if you use a technique for extracting common elements when listing playlists in order to reuse them when listing all playlists, playlists by category or when searching playlists.

* **Create a playlist** (video URLs can be added here or you can make another page where the user can add them separately). Adding video URLs later is not required it is only required to add videos when creating the playlist. Validate the input.

5 points

Users that gave up to 10 ratings to other playlists can add only up to 10 videos per playlist. Users that gave 10 or more ratings can add unlimited videos to their playlists

1 point

* **Delete videos** from playlist or the whole playlist. Only the creator is allowed to do that. You may implement this by adding **X** button next to the videos in the playlist details page. When deleting the whole playlist all videos in it should also be deleted.

4 points

* **Leave a rating (1-5)** for a playlist. Only registered users are allowed to do that. Provide 5 buttons with stars from 1 to 5 and clicking on any of them will cause a rating to be send. Users are allowed to change their ratings by clicking on the rating buttons more than once. For each user who rates a playlist you should have only 1 entry in the database.

5 points

## Administrator Area (10 points)

* **Category administration** page – Create a pageable (5 items per page) and sortable table with all categories where administrators can **create** and **edit** playlist categories.
  + The grid should have columns for the **Id**, **Name** and the **count** of the playlists in the category. Of course only the name is editable.
  + Don’t forget that the name should not be empty (add a validation for it).
  + Deleting is not required.

10 points

## Evaluation Criteria

The evaluation criteria include: correct and complete **fulfillment** of the requirements; good **technical design** and appropriate use of technologies; **high-quality code** (correctness, readability, maintainability).

Ensure your **UI behaves correctly** when the users enter invalid data, e.g. too long text in a text field or HTML special characters like "<br/>". **Validate the data** in your forms.

## Other Terms

During the exam you are allowed to use any teaching materials, lectures, books, existing source code, and other paper or Internet resources. Direct or indirect communication with anybody in class or outside is forbidden. This includes but does not limit to technical conversations with other people, using mobile phones, chat software (Skype, ICQ, etc.), email, forum posts, USB flash drives, etc.

## Exam Duration

Students are allowed to work up to **8 hours**.