**Todo List**

Why certain approaches were used, why others were not selected

* I chose Blazor Server App as template for this project because we can use C# for both client and server side development and it embraces SPA architecture. Blazor Web Assembly is supported by all popular browsers including mobile browsers. It faster and responsive because only the difference in HTML is posted back to server not the entire page over the SignalR connection.
* For Authorization I am using JWT Token, which is used to authenticate the user.
* For Data Access Layer, I am using Entity Framework Migrations. Migrations keep the database and our application data models in sync. For eg: if we want to add a column to table, we can create a migration in application for modifying the table which we will be reflected in database and reduce time efforts of dropping the table and re-creating it.
* I am using Swagger UI for WebApi interface.

Any design patterns used

* I have used Repository pattern for this project. Repository governs the data transfer between domain and data access layer.
* It abstracts the implementation of data access layer.
* By using Repository pattern, I reduce the redundancy of writing the query logic. Code is cleaner and easier to maintain.
* It decouples entity framework(EF) and application. We can change the persistence framework(EF) with minimal efforts for eg: Switching from SQL to Oracle Database.
* It facilitates dependency injection.

Anything extra you would have done give more time

* Notifications for tasks with upcoming expiry.
* Share lists to colleagues and friends.