(16 dec 2021)

ToDoList : Write-Up

Given Time span for project: **3 weak days**

API-REPO	:	https://github.com/KKutkarsh/ToDoListDeloitte
UI-REPO	:	https://github.com/KKutkarsh/DoListUI

Initial Though Process: Looking at the time constrain and the functionalities to be implemented, I chose to have Iterative approach:

1. Implement Happy Path First

2. Iterate over the solution

3. Update based on time in hand.

And to maintain code quality as much as possible. Also Main reason to Select JWT is to keep the performance in mind.

About Code Structure and Design:

The project structure is very much similar to **Domain Driven Design** (**DDD**). It is ensured that development closely follows the **SOLID principal**. **Repository Pattern** has been followed to interact with database. **Dependency injection** has been explicitly used across the application. Effort was taken to cover **Unit Tests** for most of the parts within the time constrain.

About Existing Application:

The application designed to run on IIS uses .netCore, EFCore and JWTToken on the server side. The client has been developed using Angular Framework.

Server-Side Security:

- Identity Provider along with JWT token has been used to validate and authorize users.
- The navigation of users are restricted based on authorization.
- Expiration Time of JWT is configurable in *appsettings.json*
- CROS is implemented to allow request from locally running Angular app on default Port: `4200` `URL:` `http://localhost:4200`.

Client-Side Security:

- **JWT Token** Library of Angular has been used on the client side.
- The JWT tokens on the client side is being stored in HTML local storage which is more secured than cookies.
- Angular Auth-Guard feature has been used to regulate users permissions to different URLs based on JWT token from server side.

(16 dec 2021)

Known Hot Vulnerabilities to be fixed if given more time:

- **Handle Situation when Token is compromised or logged in from multiple paces at same time** : Other than deleting token from client side we need to add the token in a BlockList (persistent) and need to validate token from every incoming request with the tokens in BlockList.
- **Getting ToDoList Items returns in link with the user as well:** Done this was for easy approach because of time constrain. Need to establish foreign key relationship between ToDoItem table and user table. Also we need to filter out only the required data in separate DTOs based on requirement. **Auto Mapper** for Model Extensions.
- **Mostly Happy Path Scenarios are checked:** Given time the null cased and other border around scenarios needs to be handled.
- **Remaining Unit Test Coverage:** Though Most of the code is covered by the unit test. There are still areas where we need to write unit test. Also the few of the existing unit tests can be further extended.
- **Code Structure:** Given time code can be better structured like using **Generic repository.** Adding **IUnitOfWork.**
- **Add Integration Test**

Security Issues to Handle:

- Rule for password has been compromised to accommodate default username and password.
- Multifactor Authentication can be added in future time.
- Sensitive data like JWT token etc. should be moved to Secure location like SSM in AWS or Azure Vault.
- Custom Authorization Filter is needed to validate and store JWT token on server side.

Structure Possible Updated:

- More Service layer can be added in the code for better modularity and separation of concern.
- However project is too small to be too much refactored.
- **Logging:** Need to integrate logging like **PagerDuty/Datadog**
- Refactor Client Side code.

Other Features that can be added:

- **Bulk Delete Option:** Can Implement checkbox to select multiple
- **Pagination:** We need pagination if ToDoList grows
- Display user detail after login.
- Better UI navigation and options.
- **Prioritize:** Adding feature to prioritize items in to do list
- **Sorting/Searching:** based on various fields like updated date, priority etc.
- **Reminder:** alert if work in todoList is more than certain time old.
- **Role Based Credentials:** Adding role based users and permissions.
- **Using Cloud services for alerts:**

(16 dec 2021)

- **Move data base from inMemory to SQL.**