Good Morning TAL Team,

I have used these technologies:

1. ReactJS for client side.
2. React features like – styled-components, controlled forms, redux, redux-saga is used, Routing etc.
3. Visual Studio 2019
4. .NET Core 3.1 API
5. Swagger for API testing
6. SOLID patterns, Repository pattern.
7. NUnit the unit testing. <WIP… due to time constraint. >
8. NLog for logging the exception in the text file.

# Improvements on the API side:

Swagger is used for the API testing.

API end point for getting the occupation list:

**https://localhost:44320/api/PremiumCalculator/GetOccupationList**

Premium calculator API endpoint:

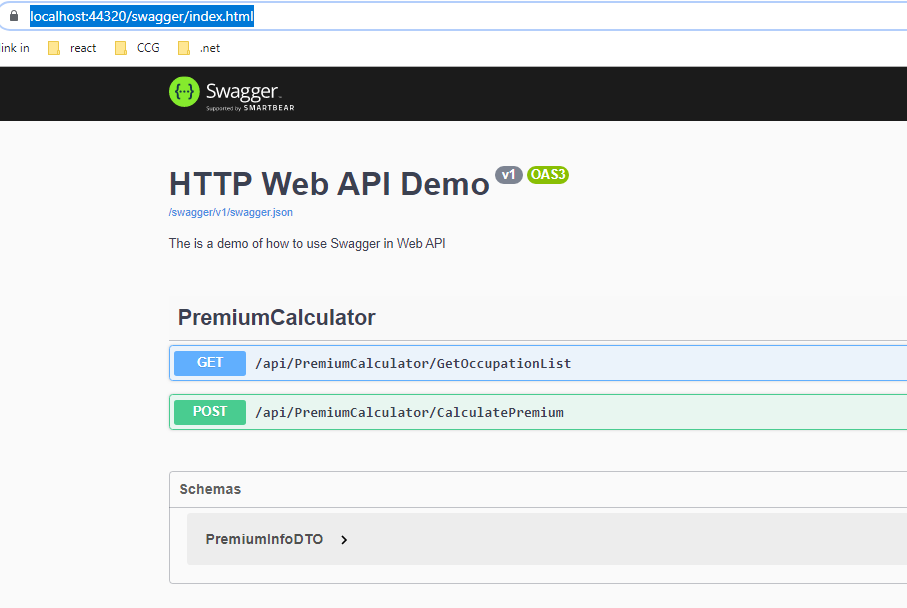
**https://localhost:44320/api/PremiumCalculator/CalculatePremium**

In case you need to change the port number of the API for the client App – update the file:

~\ClientApp\tal-premium-calculator\src\**config\index.js**

API endpoints are available on the below url:

<https://localhost:44320/swagger/index.html>

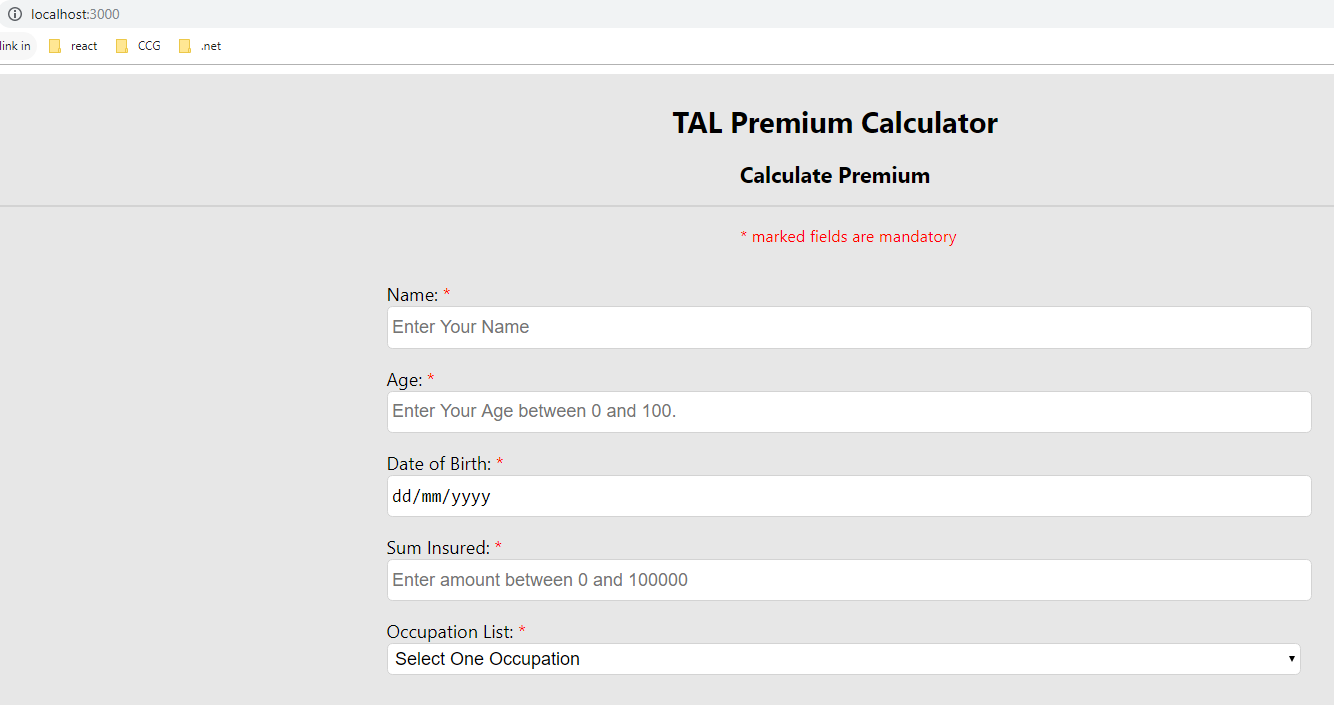


Run the ReactJS application from the below path:

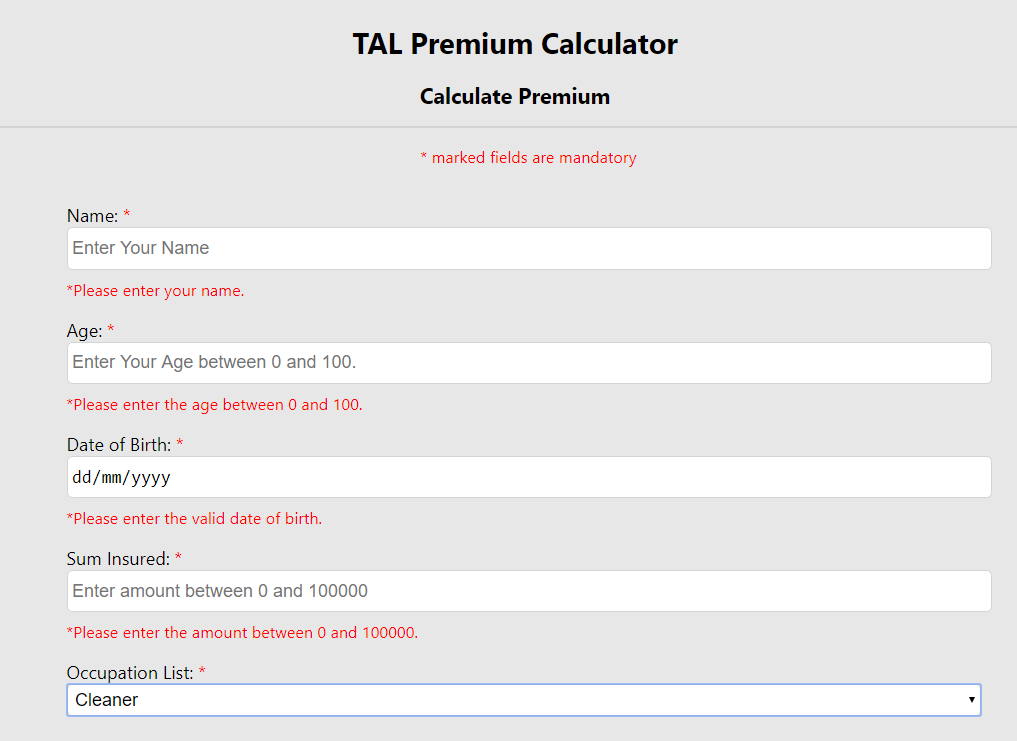
~\TALPremiumCalculator\ClientApp\tal-premium-calculator

Install the npm packages. ClientApp should build and run “npm start” to view the home page.

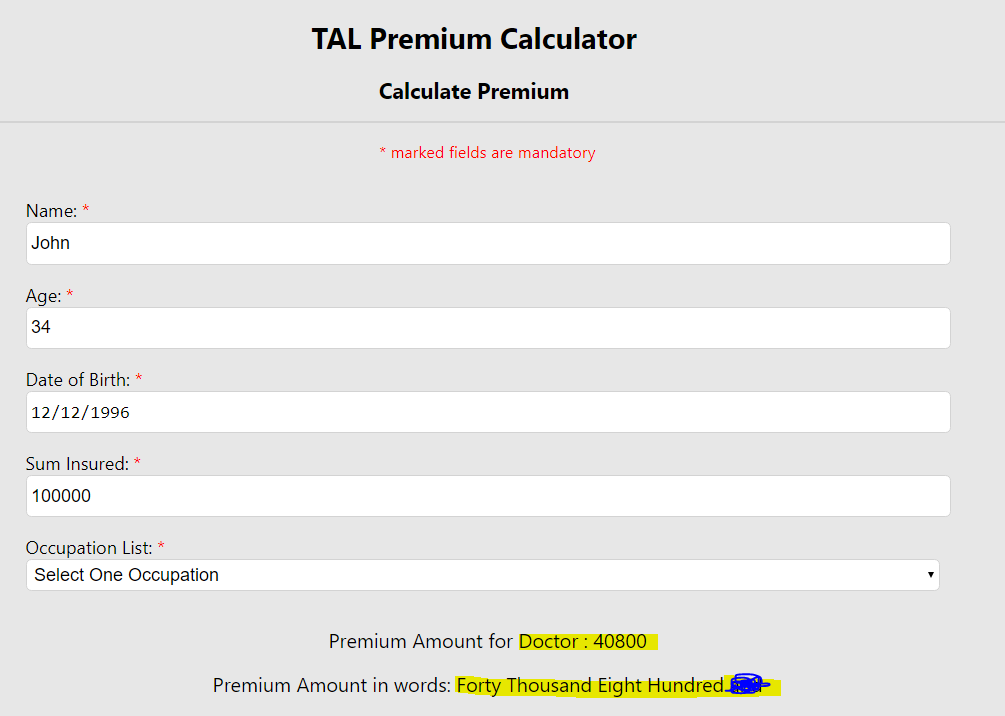
Home page of the calculator:

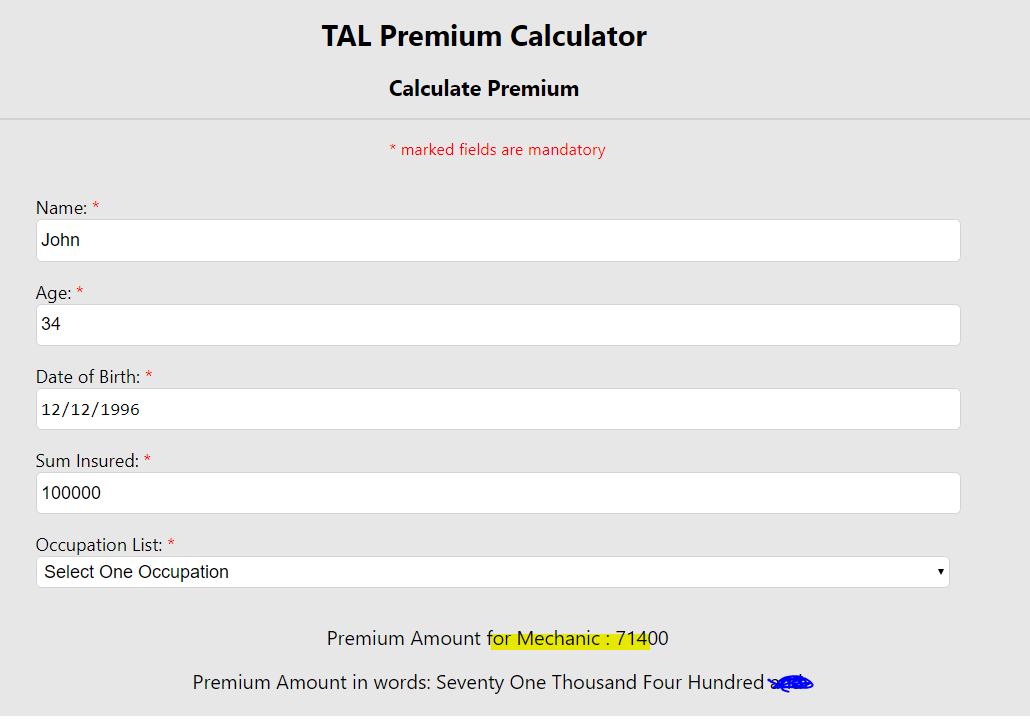


Form validations on selecting the occupation:

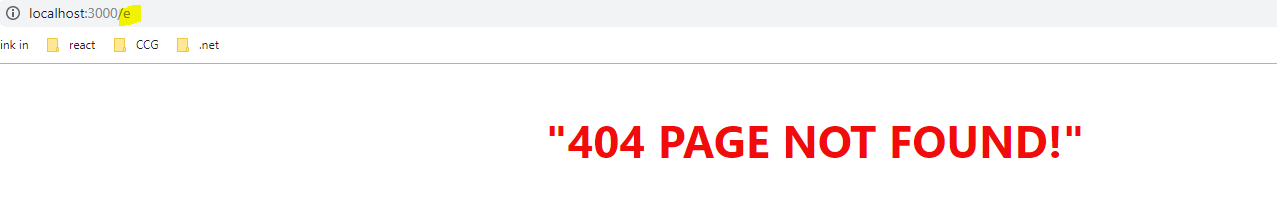


Premium calculation on filing the data and selecting the occupation:



New premium gets calculated on changing the occupation.

On non-existent route the application will show error page:



Assumption:

* No database is setup in this task.
* API should be developed by following the TDD approach.
* Few Unit test cases are added for the API