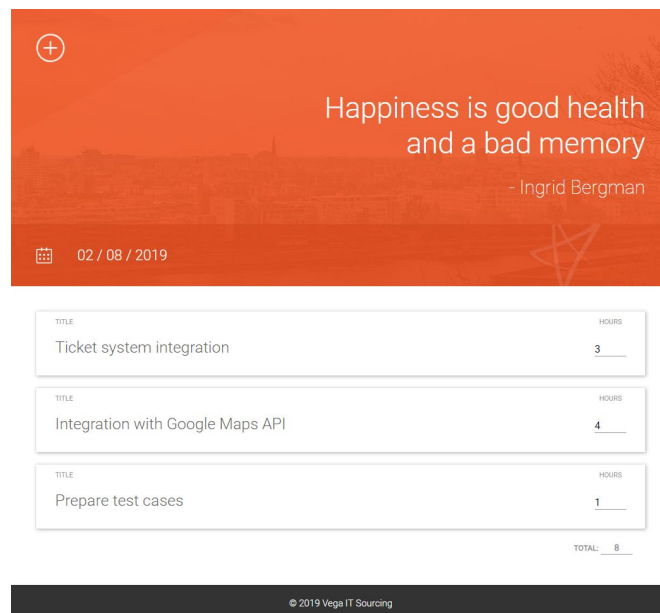


- We kindly suggest that you take some time and read this document thoroughly. You can find all the necessary details and possible answers in the text below. Good luck!
- **Note:** You are not allowed to publish your solution to a git repository. Zip the solution (without **bin** folders and **node modules**) and mail the Zip file to us, or put it on the Google Drive, and share us the link.

VEGA Timesheet application - specification

Description:

Timesheet application is a system for daily time record management. It enables users to track spent hours, which represent his/her daily tasks. Other than that, the system should also display a random motivational quote each time the user refreshes the page.

The screenshot shows the VEGA Timesheet application interface. At the top, there's a header with a calendar icon, a date "02 / 08 / 2019", and a motivational quote: "Happiness is good health and a bad memory" by Ingrid Bergman. Below the header, there's a table with three rows of tasks and their corresponding hours. The tasks are "Ticket system integration" (3 hours), "Integration with Google Maps API" (4 hours), and "Prepare test cases" (1 hour). The total hours are 8. At the bottom, there's a footer with the copyright notice "© 2019 Vega IT Sourcing".

TITLE	HOURS
Ticket system integration	3
Integration with Google Maps API	4
Prepare test cases	1
TOTAL: 8	

Requirements:

- **Create a timesheet item:** By clicking the add button (in the upper left corner), a popup for creating a new item should be displayed. “Title” and “Hours” fields are mandatory and the “Create” button should be disabled while the fields are empty. By clicking the “Create” button (if the fields are not empty), popup should disappear and new timesheet item should be added to the timesheet list.
- **Maximum hours per day:** It must **not** be allowed to exceed the maximum number of hours for a given day. The value that determines the maximum must be configurable.
- **Quotes:** List of quotes and their authors should be contained inside a **static** JSON file. Choose the quotes and authors yourself. When the page loads, a random quote is chosen from the list and displayed on the page, together with its author.
- **Current date:** On the dark orange track, near the calendar icon, you can find a date for which the daily timesheet items are displayed. The application should work for any date that a user provides through a URL parameter (e.g. <http://localhost:3000/13-5-2019>).
- **Readme.txt:** Create a readme.txt file with instructions on how to set up and run the project (e.g. how to compile React files, where to update the database connection string and etc.).

Assignments:

1. The application should be implemented through **ReactJS** and **Redux** for in-memory storage
2. The supporting backend API should be implemented using the **.NET** framework (**not .NET Core**). The API should support all the needed CRUD operations, required for the Timesheet application to function. At first, the data should be stored inside the **session** (no database is required still).
3. Create the needed requests towards the API, inside the Redux layer of the frontend app.
4. A **SQL Server** database with all the needed entities should be created and connected to the API application, over the **ADO.NET** library. Afterward, data on the API should not be stored in session anymore. It should go into the database instead.
5. API application should be implemented in three-layer architecture, with layers that you think are required.
6. Dependency Injection should be implemented inside the API application. You can choose the DI library yourself (Autofac, Castle Windsor, Unity, etc.).

Source code for the UI part of the application (HTML, CSS, and UI-related scripts) can be found on the following URL:

<https://drive.google.com/open?id=1bz3JydUaIVJf7p5fqNv0EK2i9Wzu1qEb>

Note: Assignments aren't mutually conditioned, so you don't need to do them sequentially, even though our recommended flow should be from task 1 to task 6, incrementally. Certain parts of some assignments make other parts of other assignments unnecessary, so plan well.