Lean Frontend Coding Challenge

This challenge was created to test your ability to code in a React environment with Redux as the state manager.

> If you have any questions regarding the assignment, the scope we're grading or the dynamic of the challenge, please contact me **oscar@leantech.me**

Table of Contents

Getting Started

- `yarn` to get your dependencies (you'll have to have yarn installed).
- `yarn start` to fire up your webpack environment for viewing in your browser of choice.

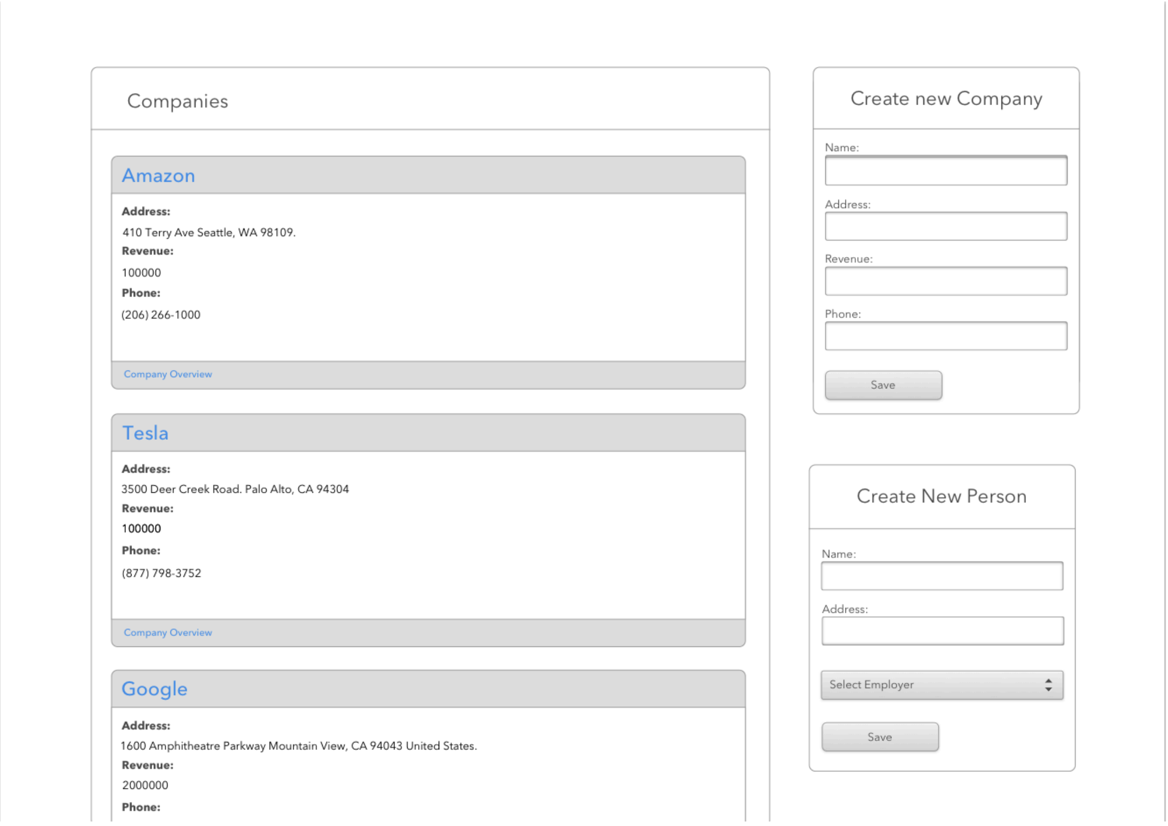
Challenge Motivations

- Show the ability to create components in JSX that follow a design mockup.
- Show proficiency in creating **reusable** components in React.
- Use React best practices in managing information.
- Use Redux to manage state at the application level.
- Use a CSS preprocessor (LESS is installed by default, however SASS or SCSS can be used)
- Use good judgement in deciding types of components (container, presentational, HOC)

Challenge Objectives

- The Numbers Page and bean counter widget were built solely as an example in architecting the directory structure and it's child files. You can choose to keep it or remove it.

- Create the widgets you see in the image below in the HomePage component.
- The widget titled *Create New Company* should have the behavior of adding new companies to the widget entitled *Companies*
- The widget titled *Create New Person* should add employees to a company's list of employees. If no companies are entered, this widget should be disabled or not visible, your choice.
- The widget title *Companies* should present all of the companies that have been entered, each with their own card.
- The company cards should provide the data entered from the *Create New Company* widget, and also have a link to the DetailsPage. The Details page will then display the respective companies information on click/key press.
- If no companies have been entered, a message saying: *There are currently no companies to review.* should be provided to the user.
- Validation should be provided in the following manner: **No fields should be empty or have falsy values**



Companies

Amazon

Address:
410 Terry Ave Seattle, WA 98109.
Revenue:
100000
Phone:
(206) 266-1000

Company Overview

Tesla

Address:
3500 Deer Creek Road. Palo Alto, CA 94304
Revenue:
100000
Phone:
(877) 798-3752

Company Overview

Google

Address:
1600 Amphitheatre Parkway Mountain View, CA 94043 United States.
Revenue:
2000000
Phone:

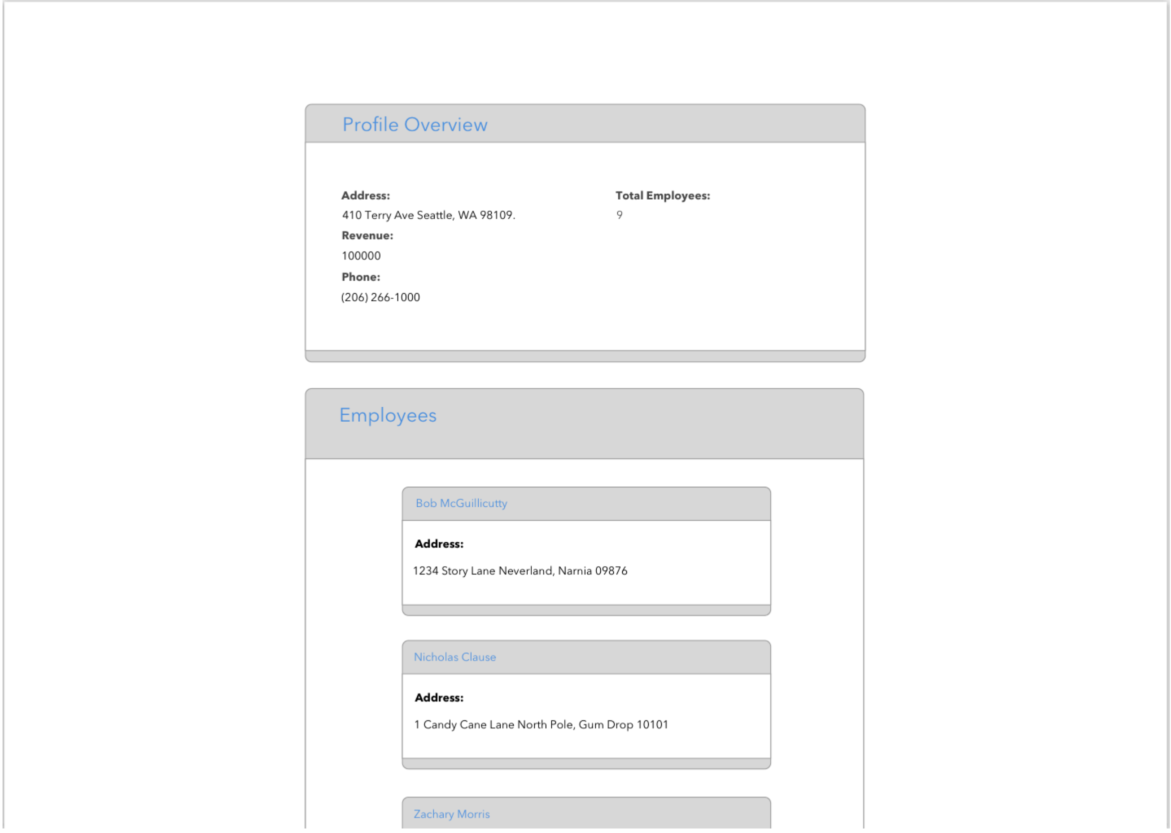
Create new Company

Create New Person

- Create the widgets you see in the image below in the DetailsPage component.

- The widget entitled *Profile Overview* should display a company's information entered from the Home page in addition to the number of employees entered in the HomePage *Create New Person* widget.

- The widget entitled *Employees* should display the employees that are affiliated with the company that was chosen from the drop down on the *Create New Person* widget from the Home page.



Profile Overview

Address:
410 Terry Ave Seattle, WA 98109.
Revenue:
100000
Phone:
(206) 266-1000

Total Employees:
9

Employees

Bob McGuillicutty

Address:
1234 Story Lane Neverland, Narnia 09876

Nicholas Clause

Address:
1 Candy Cane Lane North Pole, Gum Drop 10101

Zachary Morris

Extra Objectives

- The styling for this mockup is fairly bland, if you feel inspired, make it your own by enriching the design or use our color palette from below.
- Want to show off your understanding of asynchronous behavior in React/Javascript? Add an additional widget that pulls information from a sample API like JSON Placeholder!

Color Palette

In case you want to play around with the styles, here's a subset of our color palette.

- Primary Dark #081E25
- Primary Clear #F0F5F0
- Primary Accent #2BD675

- Primary Highlight #C5F0D7

Grading Criteria

Please keep in mind that our grading criteria is an internal guideline.

Criteria	Description
Usability	How easy it is to click and navigate between components, and interfaces.
Styling	Regardless of aesthetics, how well styling sheets and components work together.
Modularization	How well are components playing with each other, and how easy it is to reuse them.
Separation of Concerns	How good are separated stateful from stateless components, style sheets from markup, services from views.
Integration	How well integrated is the UI with the backend code, state or the REST Calls.
Documentation	How easy and complete it is to understand and read the documented code or the comments.