

Client-Side Web Technologies – Homework Assignment 5

Due: May 14th @ 11:59 PM

Value: 20% of total grade

Overview

In this assignment you will be creating a simple contact management application using AngularJS. To accomplish this, we will be making use of several 3rd party modules that we will include in our application via npm.

I have created a seed project for you that has the following:

- The npm package.json file that will install the lightweight HTTP server we used in class as well as all the packages you will need to complete this assignment already configured.
- An index.html file that has the necessary script tags for the packages you will use
- An app.js file that sets up some module configuration options, adds a file directive similar to the one demonstrated in class, and sets up routing
- Two empty view templates that correspond to the routes

This structure is contained in the seed.zip archive.

Similar to the GitHub repo code we looked at in class, you will just need to run `npm install` inside the `app` directory to get up and running. To start the HTTP server, run `npm start` and navigate to `http://127.0.0.1:8000` in your browser.

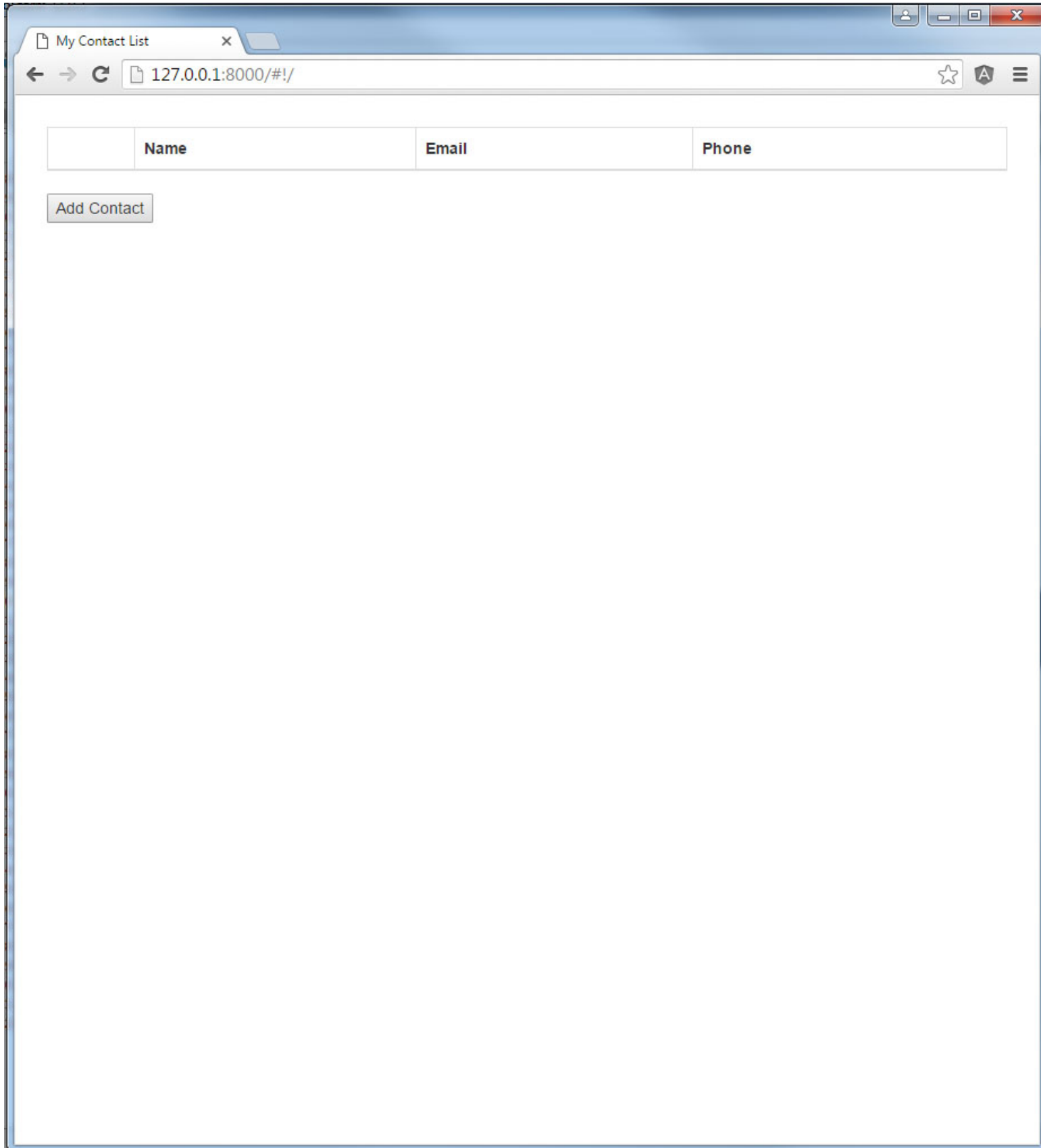
Resources

The following are resources that will be very useful:

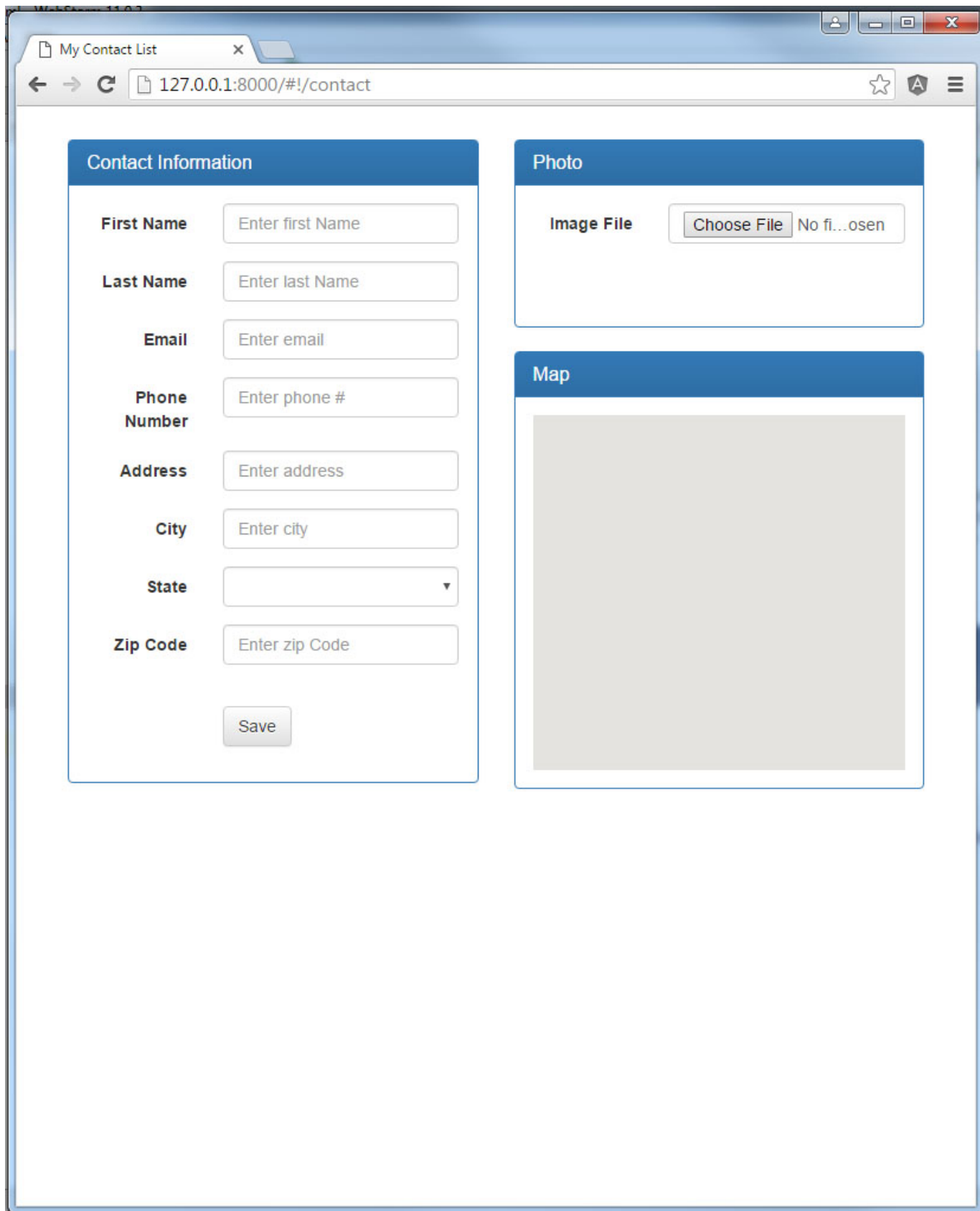
- http://www.w3schools.com/html/html5_webstorage.asp
- <https://github.com/grevory/angular-local-storage>
 - Specifically the `set`, `get`, `keys`, and `remove` methods. There are others that you may find useful.
- <https://github.com/allenhwkim/angularjs-google-maps>
 - Specifically this: http://ngmap.github.io/#/!marker_with_dynamic_address.html

Requirements

This will be a simple contact management application. When you first navigate to the application you will not have any contacts so there will just be an “Add Contact” button.



Once we have contacts they will be displayed in the table. You can show the table header as I have when there are no contacts or you can hide the table completely. It is up to you. When you click the “Add Contact” button you will be directed to the contact page.



The screenshot shows a web browser window with the title "My Contact List" and the address bar displaying "127.0.0.1:8000/#!/contact". The page contains a contact form with two main sections: "Contact Information" and "Photo".

Contact Information

First Name

Last Name

Email

Phone Number

Address

City

State

Zip Code

Photo

Image File No files open

Map

This will be a page that has the form inputs shown on the left as well as a “Save” button. On the right is a Photo section that uses the included `cswFileInput` directive. When a photo is uploaded it should immediately be displayed below the input as shown in the screenshot below. Below the photo section is a map section that uses the 3rd party map directive. When an address is entered the map should zoom in and mark the address as shown in the screenshot below.

My Contact List

127.0.0.1:8000/#!/contact

Contact Information

First Name

Jason

Last Name

Mussitsch

Email

jmusits@cs.cmu.edu

Phone Number

412-916-2734

Address

417 S. Craig St.

City

Pittsburgh

State

Pennsylvania

Zip Code

15213


Save

Photo

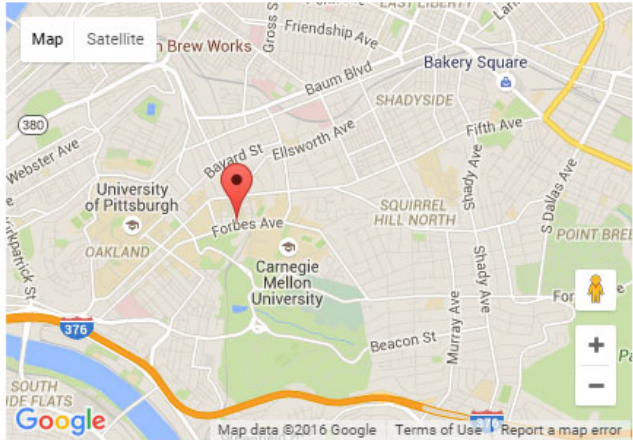
Image File

Choose File

Koala.jpg

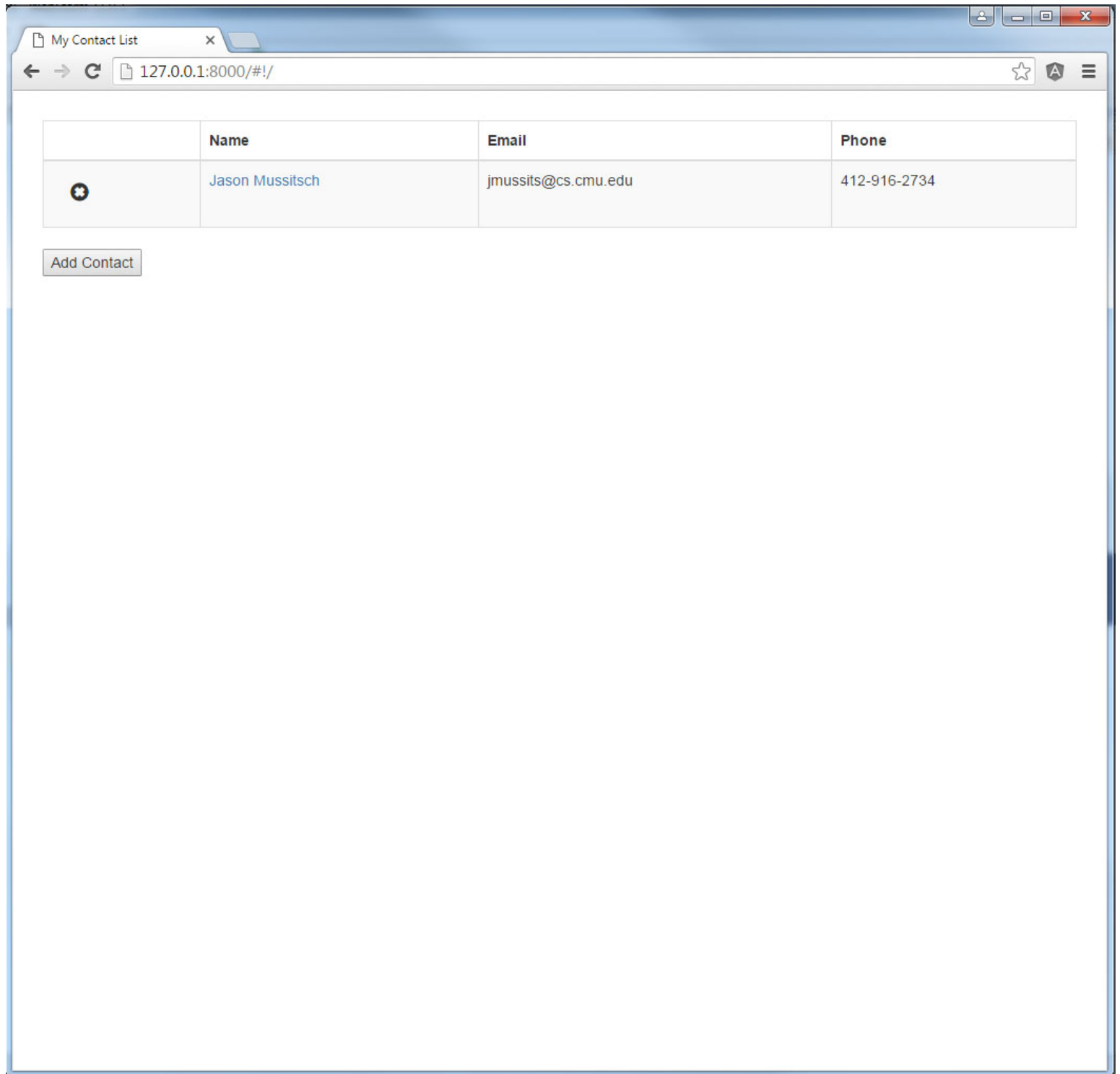


Map



Your layout and styles do not need to look exactly like shown. I used Bootstrap but you are not required to. This project does not have any CSS requirements.

When the “Save” button is clicked you will save all the contact information including the photo image to local storage using the 3rd party local storage service. Local Storage works by storing data using key/value pairs so you will need a key for each contact. You can generate some id or you can use first name and last name as the key. Your application will allow for editing of contacts so if you use the name as a key you must handle this either by not allowing name to be edited or by removing the old key and adding the new key when names change. Once the data is stored your application must switch back to the initial page that lists contacts. See the screenshot below to see how it should look when there are contacts.



Name, Email, and Phone should be displayed as well as a clickable X icon on the left that when clicked removes the contact from the table and from local storage.

The name column should show first name then last name and be clickable. When clicked, the contact page should load and be populated with all the contact details for that contact as shown below.

My Contact List

127.0.0.1:8000/#!/contact/Jason_Mussitsch

Contact Information

First Name

Jason

Last Name

Mussitsch

Email

jmussts@cs.cmu.edu

Phone Number

412-916-2734

Address

417 S. Craig St.

City

Pittsburgh

State

Pennsylvania

Zip Code

15213


Save

Photo

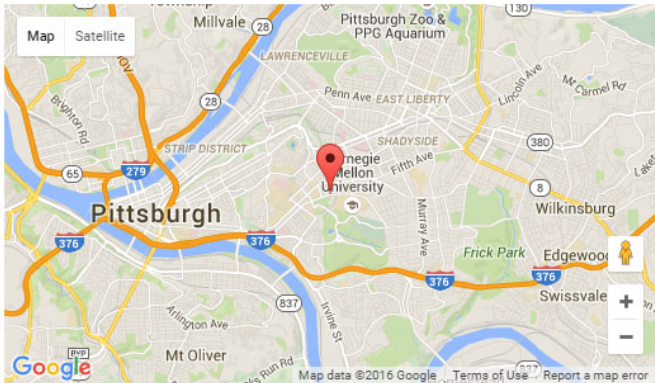
Image File

Choose File

No file chosen



Map



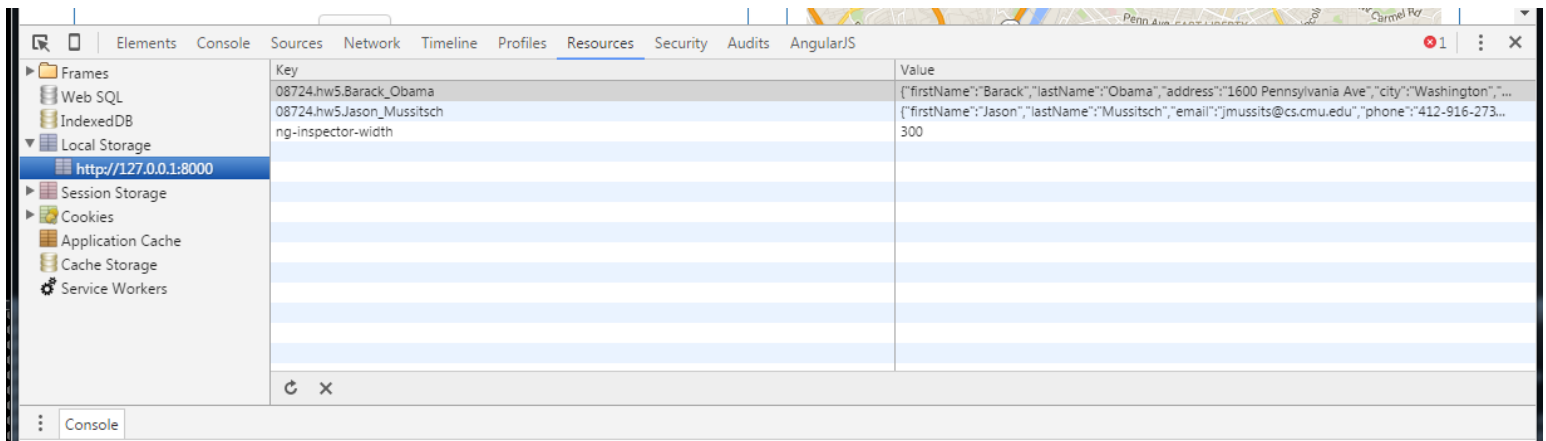
Making changes and clicking “Save” should update the contact in local storage and direct back to the list page.

Viewing Local Storage

Using Chrome's dev tools we can look at what is in local storage. Go to the Resources tab and then expand the Local Storage node on the left. Click on the domain for your server and you will see all the key/values currently stored by the browser. Local Storage saves the data even after you completely close the browser. So you should be able to close your browser and reopen to your application and see all the saved contacts.

Most browsers enforce limits on the amount of data a domain can store in local storage (I think around 5 MB is typical). Consider also that since we are base-64 encoding the images that the amount of storage needed per image will be a bit larger than the actual binary size of the original file. Therefore, you will run into issues if you try to test your application with large image files. I would stick to files in the 100 KB range or less.

With that said, it obviously creates a severe limitation in the application. That is OK since this is just an academic exercise. If we wanted to modify the design to not have this issue, then we would need to upload the image files to a server (or maybe cloud storage like Amazon S3) and save the URI that points to the actual image. Since we are not dealing with server-side code in this course I did not want to do that for this assignment.



Using the cswFileInput Directive

Similar to the example in class, this directive does not work with an array of objects but just binds to a single variable (or object property) that will contain the data URL string. You can use it like this:

```
<input type="file" csw-file-input file-data="someObject.someProperty">

```


Submission

To submit your assignment, add your files to a ZIP archive, name the ZIP file ***Homework5_[andrewid].ZIP***, and upload to Blackboard under Assignment 5 by the due date/time above. For example, my ZIP file name would be ***Homework5_jmussits.ZIP***.

Do NOT include your node_modules folder. I do not need these and they will make your submission too large.

To grade your application I should be able to run from your unzipped directory npm install and then npm start and then navigate to <http://127.0.0.1:8000>.

Grading Rubric

This assignment is worth 100 points (20% of your total grade). The following is how the assignment will likely be scored:

- Functionality works as described above **[60 points]**
- AngularJS is used correctly (e.g. no DOM access unless in directives, \$scope used properly, injection patterns followed correctly, etc.) **[40 points]**