CPSC1520 – JavaScript 1 Assignment 4: Friend Finder [10%]

Introduction

This assignment builds on what you have already learned and demonstrated in class through the exercises and assessments to this point (i.e. up to and including async requests and JSON). You are responsible for adding the JavaScript that will bring this page to life.

Overview

The purpose of this assignment is to populate the page with data that has been obtained via asynchronous calls to a HTTP web server. There are two components to this assignment that you will need to carry out.

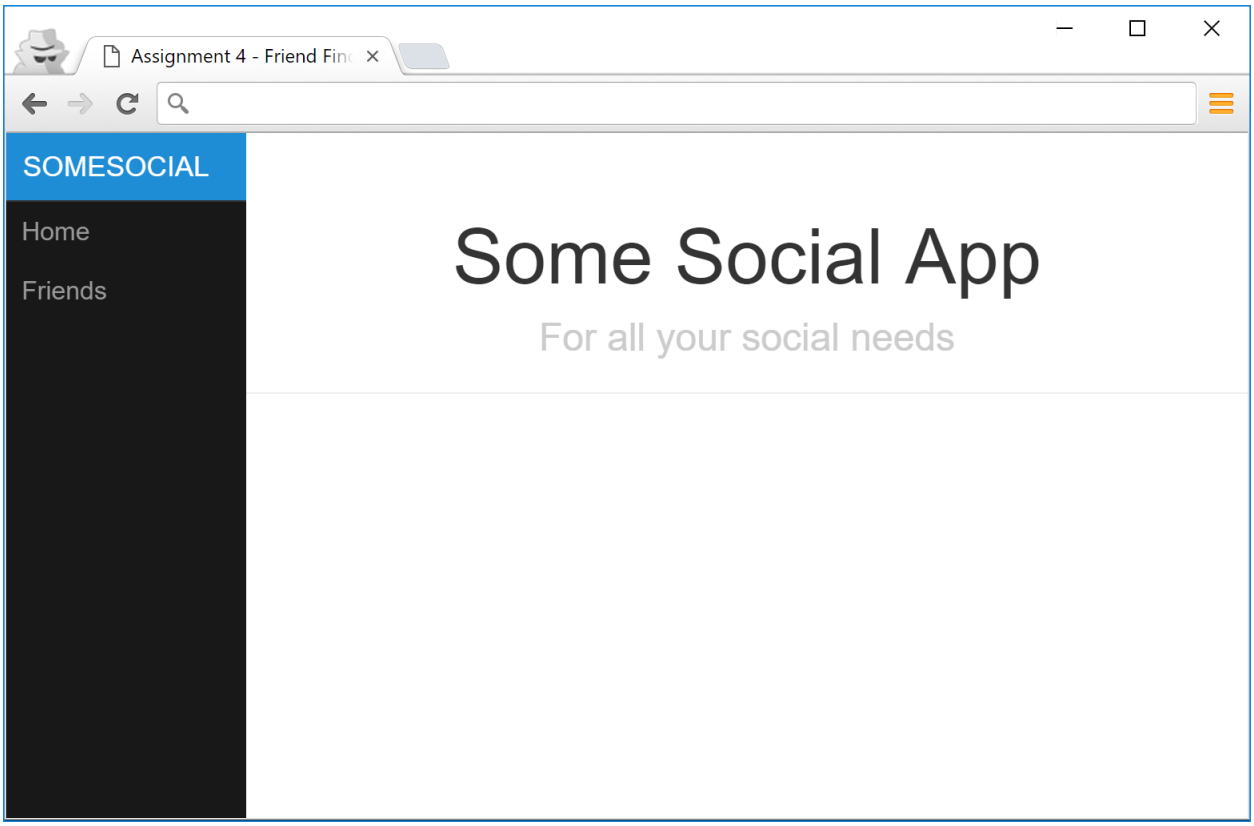

Figure 1. Initial user interface for the assignment

Fetching Data

The first component requires you to send async requests when additional data is required. There are two scenarios for fetching data: 1) fetching all short friend data (friends.json) and 2) fetching information for each individual friend's data (e.g. 1.json, 2.json, etc.).

Scenario 1 – Fetching friends list data

Scenario 1 involves the following:

- User clicks the **Friends** menu item, which triggers an async call for **friends.json**
- Upon successfully loading the response, the data in the response must be parsed and then rendered in the **div.content** element (see below)
    - The navigation menu Friends li element should also have the **pure-menu-selected** class added (highilights the menu item as shown below)

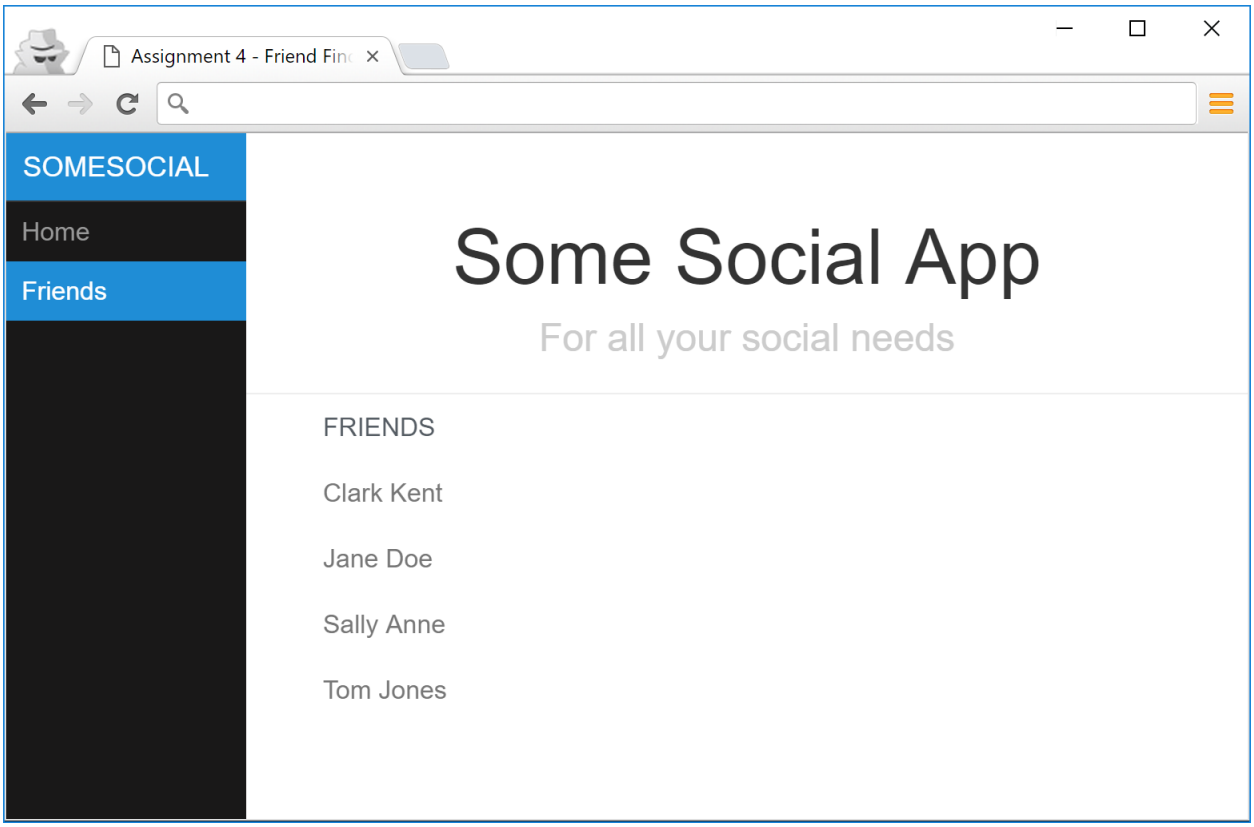

Figure 2. Rendering after friends.json data has been fetched

The markup for the friends display is provided in a comment in the index.html file.

Scenario 2 – Fetching Individual friend data

Scenario 2 involves the following:

- User clicks a friend in the friends list, which triggers an async call for the associated friend data file
    - The friend's id value (found in the **data-id** attribute) should be used to fetch its associated friend data file. For example, if the friend's data-id is 1, then the **1.json** data file should be fetched.
- Upon successfully loading the response, clear the **div.content** element and render the individual friend data

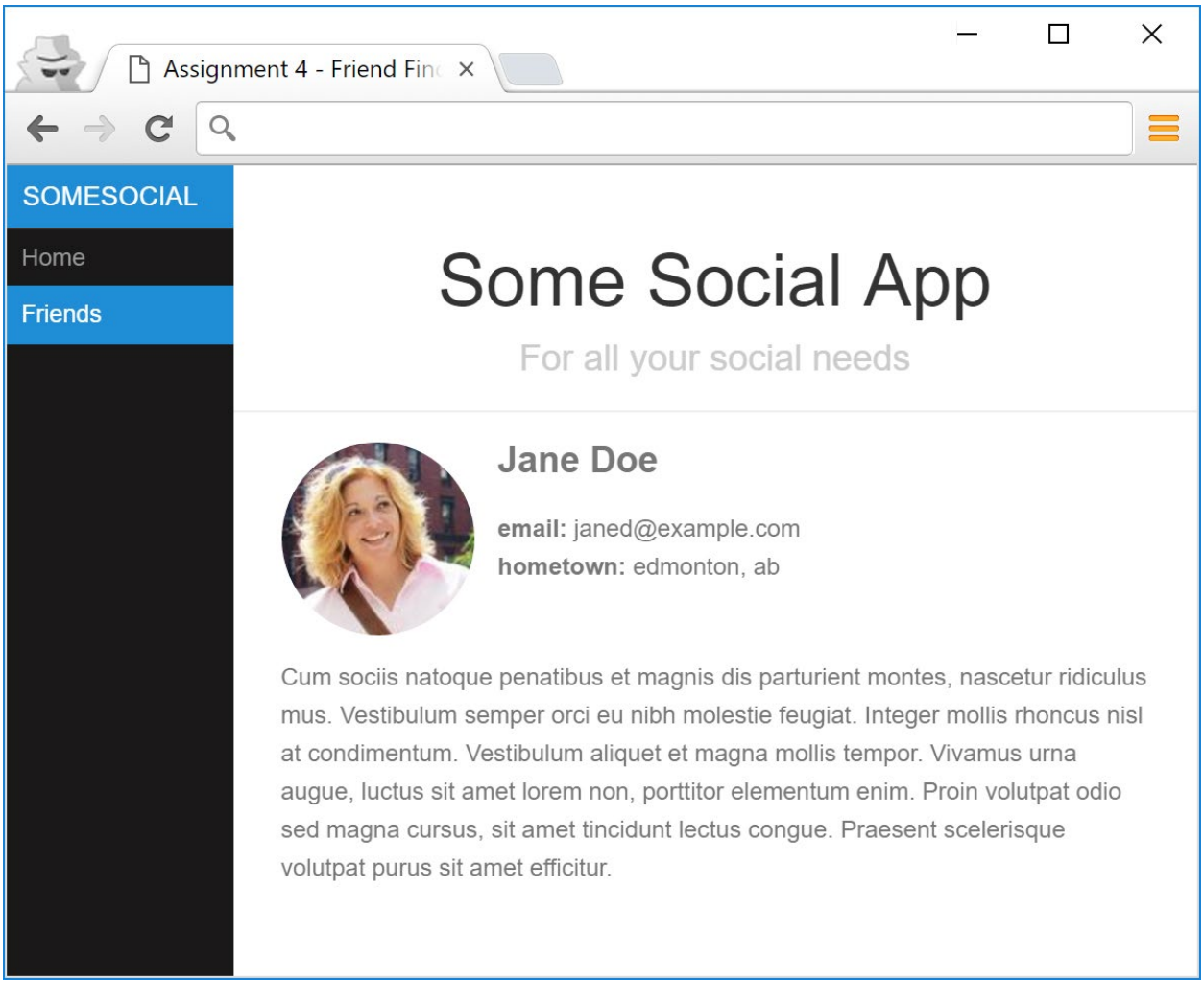

Figure 3. Individual friend render

The markup for the individual friend display is provided in a comment in the index.html file.

Composing JSON Data

The second component requires you to update the JSON data files for each friend. Currently, the **bio**, **email**, and **hometown** properties are missing. Update each of the individual friend data files so that they contain the required properties.

Ensure that these properties are properly rendered as shown in the image above.

Required Tasks

The following is a list of requirements for this assignment:

- An event listener is required for clicking the Friends main menu item
    - The Friends menu li should have the class pure-menu-selected added
    - An async request should be sent for the friends.json file
    - The data should be parsed and rendered as shown in the HTML friends-list comment in index.html
- An event listener is required for clicking on individual friends in the displayed friends list
    - An async request should be sent for the individual friend's data file
        - The data-id attribute contains the id of the friend's data to fetch
            - Corresponding file (e.g. data-id=1 -> 1.json)
    - The div.content should be cleared
    - The data should be parsed and rendered as shown in the HTML friend comment in the index.html
- Update each of the individual friend data files (1.json, 2.json, etc.) with the required missing properties:
    - bio, email, hometown
- If you are unsure of any of the requirements, **be sure to ask your instructor for clarification**.

References

1. http://uifaces.com/

Marking Key

Tasks	Grade	Marks	Total
Friend Menu Listener • Event listener added • Async request • Friends list properly rendered • Friend menu item updated		 **1** **5** **3** **1**	
Individual Friend Listener • Event listener added • Async request • Friend properly rendered		 **1** **3** **3**	
JSON Update • Add the required JSON properties		**3**	
Code Formatting and Style		**-3**	

Comments	SCORE	GRADE
	20	

Marking Rubric

Marks	5 Marks Criteria
5	**Task was completed with the highest of proficiency adhering to best practices and followed subject matter guidelines all tasks were completed to a professional standard.**
4	**Task was completed well some minor mistakes. Well above average work shows good understanding of the task and high degree of competence**
3	**Satisfactory work some features missing or incorrectly implemented. Show a moderate level of understanding in the task with room for improvement.**
2	**Below average work. Task was poorly complete. Show understanding of the task and the requirements to implement but implementation was poorly executed.**
1	**Some of the task was completed. Showed a lack of understanding in the subject matter and very poorly executed**
0	**Not completed.**

Marks	**3 Marks Criteria**
3	**Proficient shows a high degree of competence in completing task.**
2	**Capable above average degree of competence in completing task**
1	**Satisfactory shows a satisfactory degree of competence in completing task.**
0	**Shows a limited degree of competence in completing task.**

Marks	**1 Marks Criteria**
1	**Task Completed satisfactorily**
0	**Task was not**