1 Homework 4

Due Date: CS 580W: Dec 5; CS 480W: Dec 7
No late submissions

To be turned in on paper in class.

Important Reminder: As per the course *Academic Honesty Statement*, cheating of any kind will minimally result in receiving an F letter grade for the entire course.

Please remember to justify all answers.

You are encouraged to use the web or the library but are required to cite any external sources used in your answers.

1. The course discussed two approaches for maintaining state across multiple HTTP requests:

   (a) Include the state as part of the URL for each request, usually as a query parameter.

   (b) Use cookies.

   Usually the state maintained in a client is some kind of id which identifies state stored on the server.

   Discuss the tradeoffs between these two approaches. *10-points*

2. John is a non-technical proprietor for a small business. He hires a consultant to develop a web site where customers can place orders. The consultant successfully delivers a slick looking web site based on express.js and sets up the express.js server using a web hosting company. The web site works well and orders are pouring in.

   (a) Unfortunately, after a few days of successful operation John gets a call from the web hosting company informing him that the memory footprint of the express.js server has increased tremendously.

   (b) John has no idea what that means and places a call to the consultant requesting assistance.

   (c) Before the consultant can get back to him, John receives another call from the web hosting company informing him that his server crashed and they have restarted it.

   (d) The web site is working well with the server having a small memory footprint. However, John starts receiving complaints from customers saying that they have lost the contents of their shopping carts.

   Give a possible reason for this sequence of events. *5-points*

3. One disadvantage of the PRG Post-Redirect-Get pattern is difficulty sending information from a successful `POST` to the `GET`. Discuss alternative solutions to this problem? *10-points*

4. You are hired to help an ecommerce web site evaluate two alternatives for allowing users to checkout their shopping carts:

   (a) A checkout flow:

      i. A shipping page where the user provides shipping information for the order.

      ii. A billing page where the user provides a credit card number to pay for the order.

      iii. A page where the user can review all the order information before deciding whether to place the order.

      iv. An invoice page for the order after it has been successfully placed.

   (b) A single checkout page on which the user provides all necessary information like shipping and billing. When an order is placed, the checkout page is replaced by an invoice when an order is sucessfully placed.

   The web site may need to support the following:

   - Allow the user to specify promotional offers like coupons during the checkout. These offers may affect the total cost of the order.
   - Allow a discounted (possibly free) shipping based on the total cost of the order.

   Provide a technical evaluation for the two alternatives. Your evaluation should include the following:

   - For the checkout flow, suggest possible URLs for the different pages as well as the HTTP methods used.
   - The information flow between the user and the application.
   - Requests sent between the browser and server.

   You may assume that the server stores all details of the shopping cart and current state of the checkout flow with cookies being used to link browser requests to the server state. However, for security reasons, credit card numbers are not permitted to ever be stored on the server. *15-points*

5. A code base uses `XMLHttpRequest` to provide an `insertContent()` function which asynchronously inserts content from a specified `url` into the DOM element specified by an `id`:

```
function insertContent(id, url) {
  const req = new XMLHttpRequest();
  req.onreadystatechange = () => {
```

```
    if (this.readyState == 4 &&
        this.status == 200) {
      //insert into page
      document.getElementById(id).
        innerHTML = req.responseText;
    }
  };
  req.open("GET", url);
  req.send();
}
```

This function is used to insert messages into a page:

```
const messages = [
  { id: 'intro', url: 'http://ex.com/intro.txt', },
  { id: 'descr', url: 'http://ex.com/desc.txt', },
  ...
];

messages.forEach(m => insertContent(m.id, m.url));
```

Since the messages are received asynchronously, they will be inserted into the DOM in an arbitrary order. Describe how you would modify the above code so as to insert the messages into the DOM in the same order as they are specified in the `messages[]` array. *10-points*

6. Jane, who is a programmer working at a university lab, is learning about web services and builds a simple web service running on her PC at `http:¬ //jane.lab.university.edu/fortune` which returns a random fortune. Jane embeds this service into her personal home page on her PC by using the `insertContent()` function from the previous question with the following code snippet:

```
insertContent('fortune', '
                'http://jane.lab.university.edu/fortune')
```

The lab director is impressed by this web service and instructs the lab's intranet programmer to embed the fortune into the lab's intranet home page at `https://lab.university.edu` by:

   (a) Adding a suitable `<div id="fortune"></div>` to its HTML.

   (b) Modify its JavaScript to include the `insertContent()` function as well Jane's code snippet which calls `insertContent()`.

This does not work. Give possible reasons why. *10-points*

7. Given a reactjs component `Component`:

```
class Component extends React.Component {
  constructor(props) {
    super(props);
    this.id = props.id;
    //no redefinition of this.handleChange
  }

  handleChange() {
    //references this.id
    ...
  }
}
```

   (a) What is wrong with the following `render()` method for the above `Component`?

```
render() {
  return (
  <input type="text" onChange={this.handleChange()}/>
  );
}
```

   (b) What is wrong with the following `render()` method for the above `Component`?

```
render() {
  return (
    <input type="text"
       onChange={function() {this.handleChange()}}/>
  );
}
```

   (c) Why will the following closely related `render()` method for the above `Component` work?

```
render() {
  return (
    <input type="text"
         onChange={() => this.handleChange()}/>
  );
}
```

   15-points

8. You need to call two independent remote web services fronted using two `async` functions `async function service1()`, and `async function ser-`

`vice2()`. Since these are remote web services, it is likely that the calls take appreciable time (milliseconds) which is not predicable beforehand. Your code needs to block until both web service calls have completed. It is trivial to use `await` to write code which calls both web services sequentially but that has the disadvantage of taking time which is the sum of the times taken by each service. How would you set things up so as to call both services but to block only for a time corresponding to the slowest service? *10-points*

9. You have a working promise chain:

```
promiseGen().
  then(function(v1) { ... return v2; }).
  then(function(v2) { ... return v3; }).
  then(function(v3) { ... return v4; });
```

Due to changes in your code, you need to change the `function()` in the third `then()` so that its body has access to the value `v1` returned to the first `then()` by the promise generator `promiseGen()`. However, `v1` is out-of-scope for the third `then()`. How can you make this happen? *15-points*