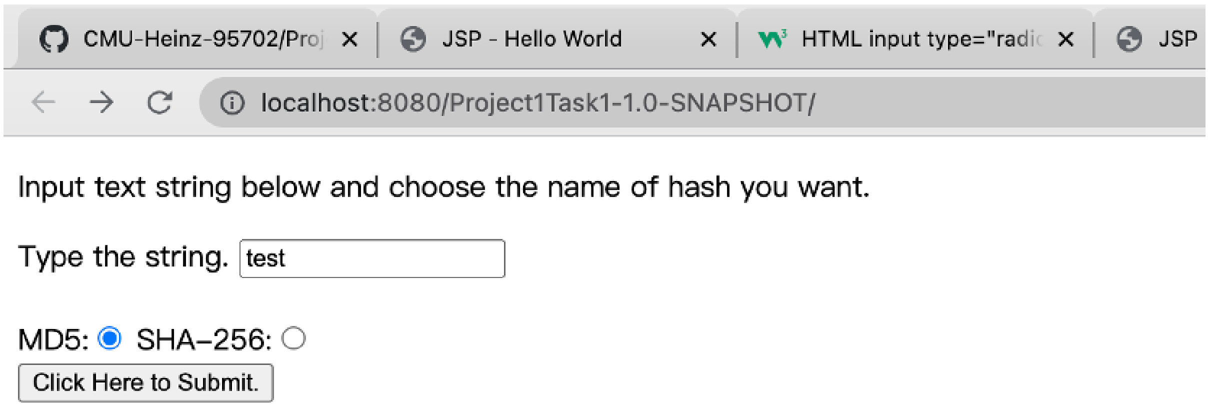
Project1

Task 1

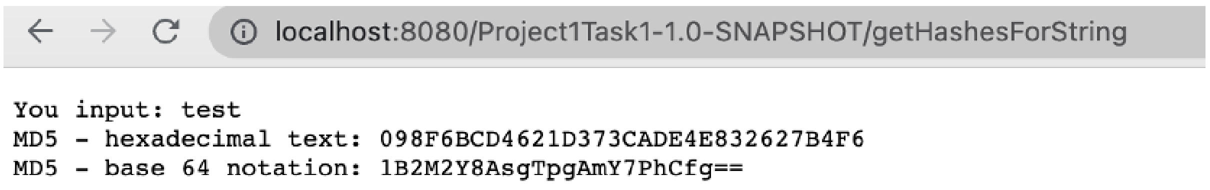


Input text string below and choose the name of hash you want.

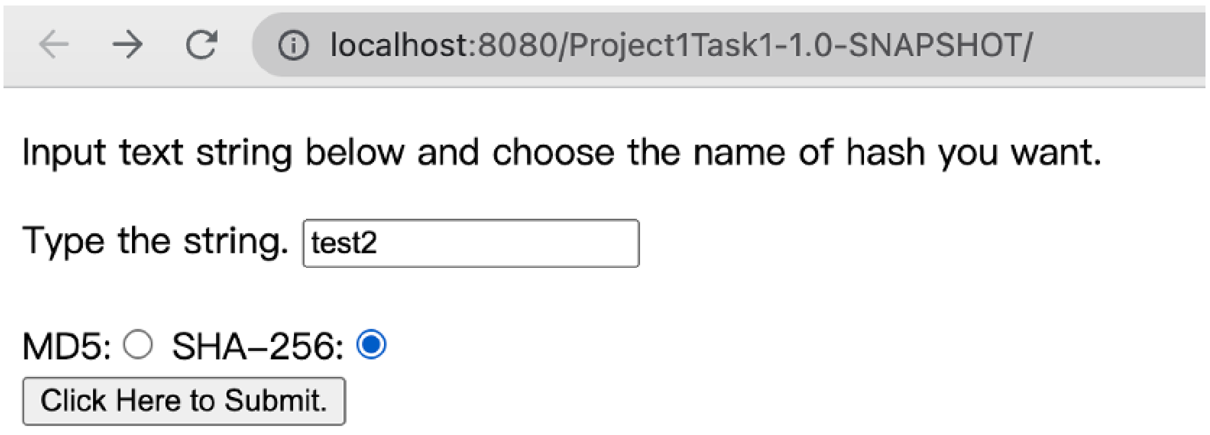
Type the string. test

MD5: [✓] SHA–256: [ ]

Click Here to Submit.


localhost:8080/Project1Task1-1.0-SNAPSHOT/getHashesForString

```
You input: test
MD5 – hexadecimal text: 098F6BCD4621D373CADE4E832627B4F6
MD5 – base 64 notation: 1B2M2Y8AsgTpgAmY7PhCfg==
```

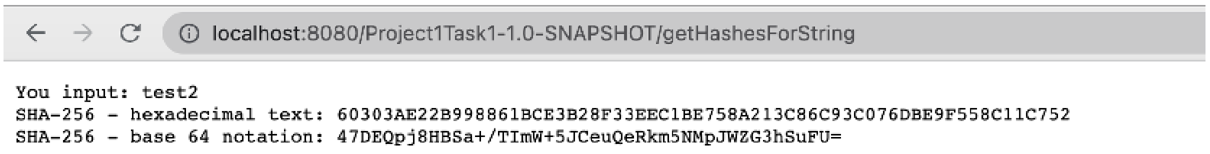


Input text string below and choose the name of hash you want.

Type the string. test2

MD5: [ ] SHA–256: [✓]

Click Here to Submit.


localhost:8080/Project1Task1-1.0-SNAPSHOT/getHashesForString

```
You input: test2
SHA-256 - hexadecimal text: 60303AE22B998861BCE3B28F33EEC1BE758A213C86C93C076DBE9F558C11C752
SHA-256 - base 64 notation: 47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=
```

- code for generating hex and base64 for md5 and sha-256:

```
@WebServlet(name = "ComputeHashes", value = "/getHashesForString")
public class ComputeHashes extends HttpServlet {
    /**
     * Get the string that user inputs and return its hash value(both hex and base64 e
xpression) based on hashing method.
     * @param request
     * @param response
     * @throws ServletException
     * @throws IOException
     */
    @Override
    public void doPost(HttpServletRequest request, HttpServletResponse response) throw
s ServletException, IOException {
        String searchWord = request.getParameter("searchWord");
        String hashMethod = request.getParameter("encrypt");
        String hex = "";
        String base64= "";
        if (hashMethod.equals("MD5")) {
            try {
                MessageDigest md = MessageDigest.getInstance("MD5");
                md.update(searchWord.getBytes());
                // hex text
                hex = DatatypeConverter.printHexBinary(md.digest());
                // 64 base notation
                base64 = DatatypeConverter.printBase64Binary(md.digest());
            } catch (NoSuchAlgorithmException e) {
                System.err.println("oops,MD5 is not a valid message digest algorith
m.");
            }
        } else if (hashMethod.equals("SHA-256")) {
            try {
                MessageDigest md = MessageDigest.getInstance("SHA-256");
                md.update(searchWord.getBytes());
                // hex text
                hex = DatatypeConverter.printHexBinary(md.digest());
                // 64 base notation
                base64 = DatatypeConverter.printBase64Binary(md.digest());
            } catch (NoSuchAlgorithmException e) {
                System.err.println("oops,SHA-256 is not a valid message digest algorit
hm.");
            }
        }
        PrintWriter printWriter = response.getWriter();
        String returnString = "You input: "+searchWord+"\n"+hashMethod+" - "+"hexadeci
mal text: "+hex+"\n"+hashMethod+" - "+"base 64 notation: "+base64+"\n";
```

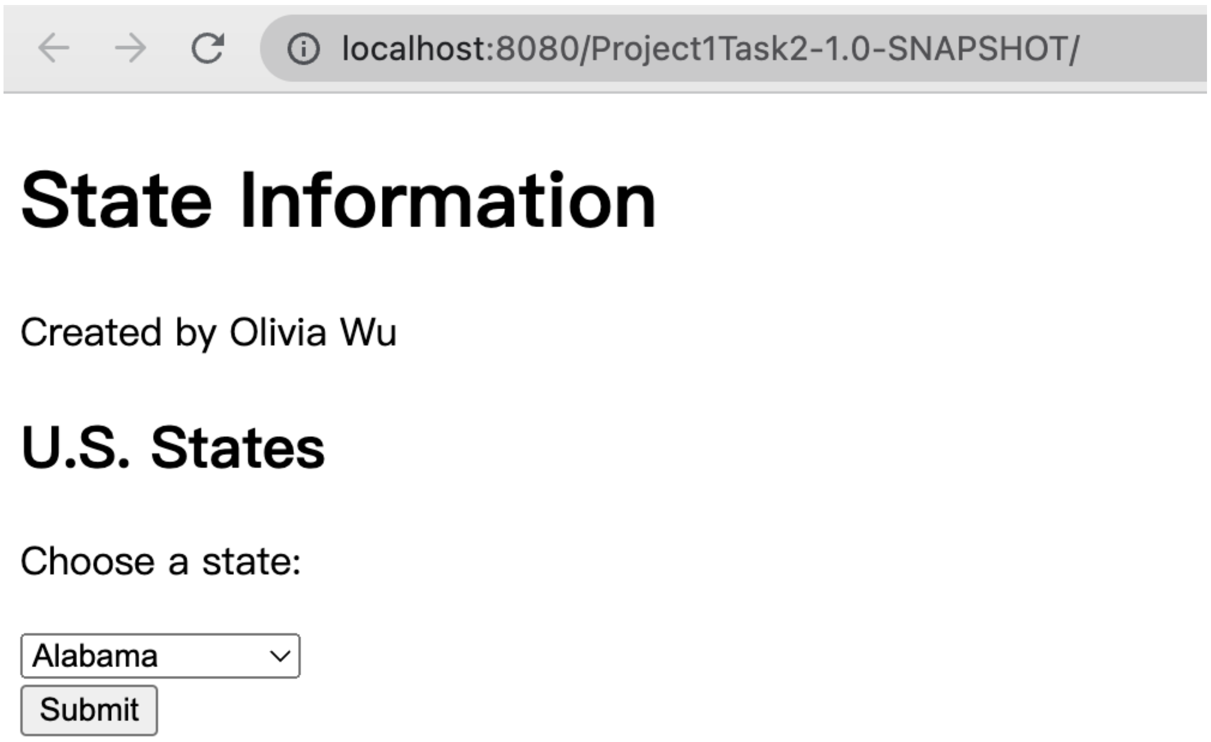
```
        printWriter.write(returnString);
    }
}
```

- index.jsp file:

```
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Task1 Compute Hashes</title>
    </head>
    <body>
        <p>Input text string below and choose the name of hash you want. </p>
        <form action="getHashesForString" method="POST">
            <label for="letter">Type the string.</label>
            <input type="text" name="searchWord" value="" /><br>
            <br>
            MD5:<input type="radio" name="encrypt" value="MD5" checked>
            SHA-256:<input type="radio" name="encrypt"  value="SHA-256">
            <br>
            <input type="submit" value="Click Here to Submit." />
        </form>
    </body>
</html>
```

Task 2

- input page and drop down menu:

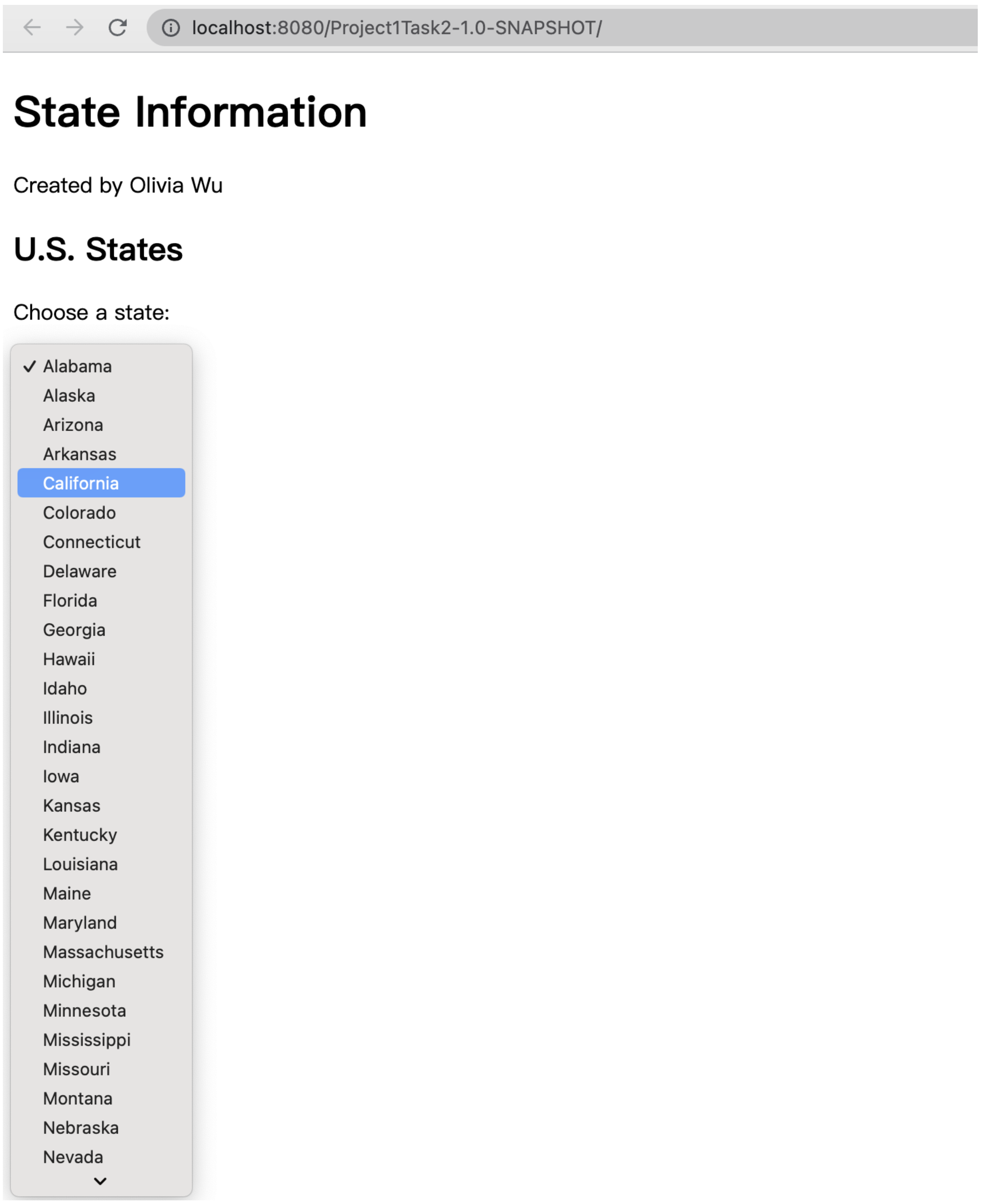

localhost:8080/Project1Task2-1.0-SNAPSHOT/

State Information

Created by Olivia Wu

U.S. States

Choose a state:



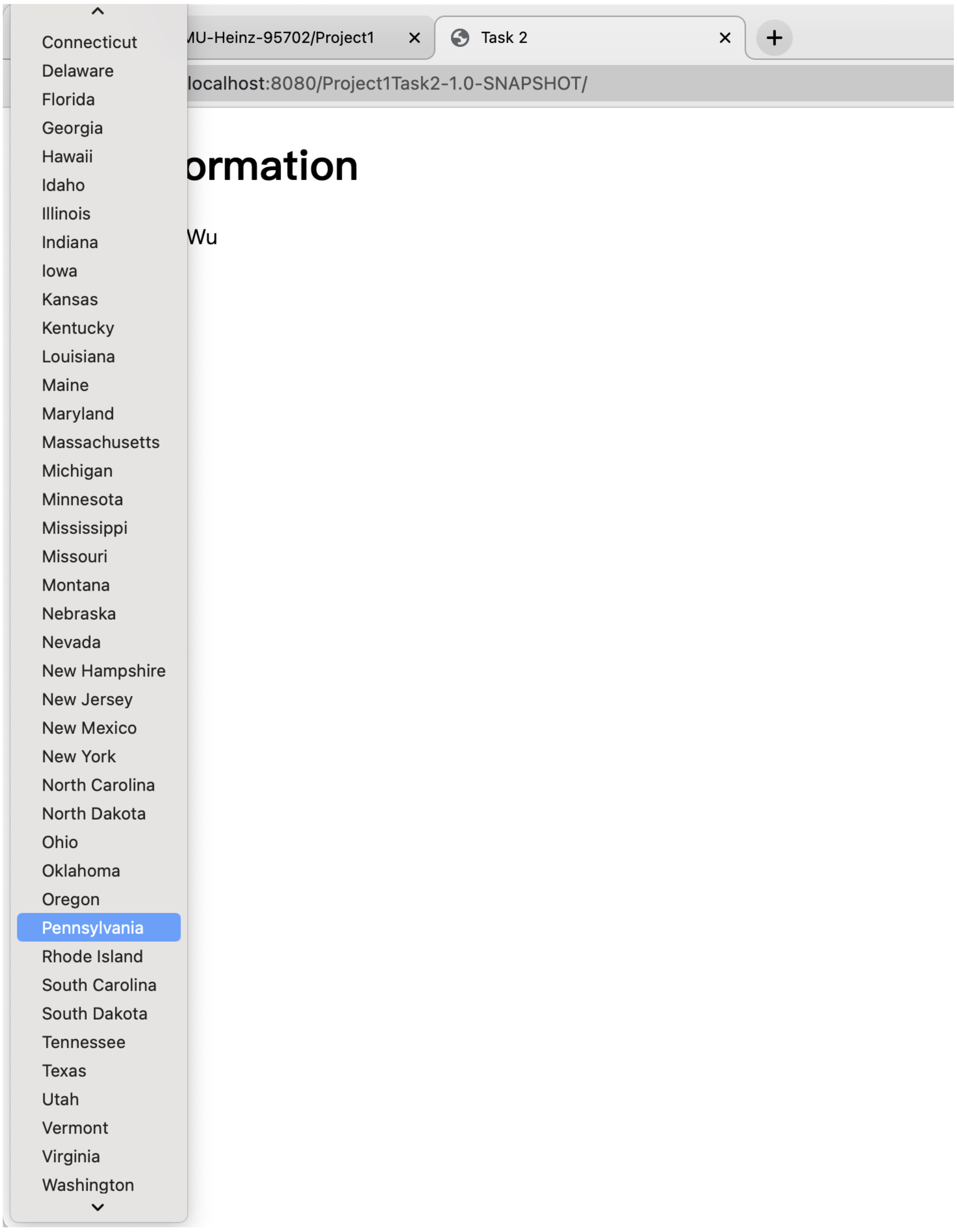
State Information

Created by Olivia Wu

U.S. States

Choose a state:

✓ Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky
Louisiana
Maine
Maryland
Massachusetts
Michigan
Minnesota
Mississippi
Missouri
Montana
Nebraska
Nevada



MU-Heinz-95702/Project1

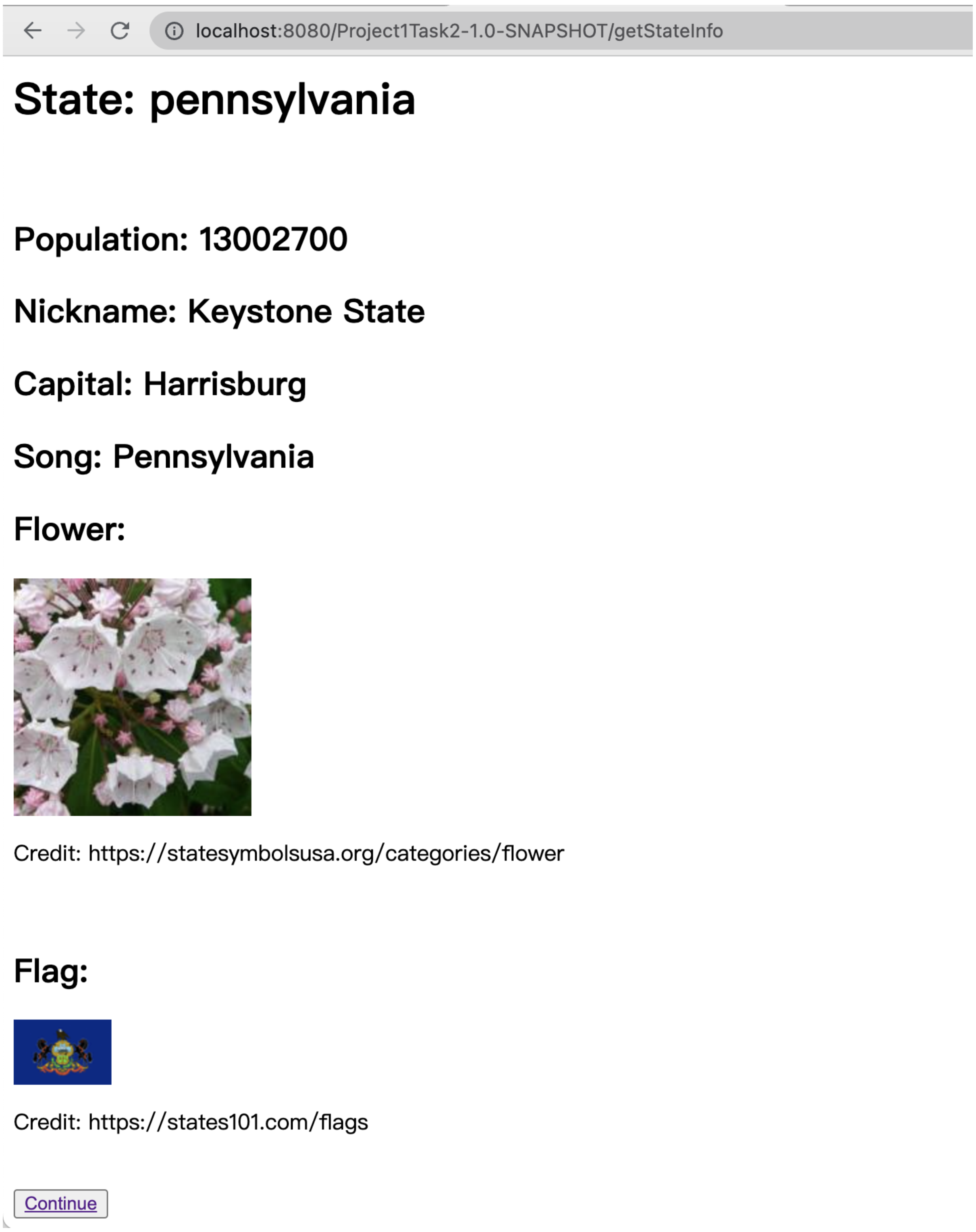
Task 2

localhost:8080/Project1Task2-1.0-SNAPSHOT/

ormation

Wu

- output page for Penn


localhost:8080/Project1Task2-1.0-SNAPSHOT/getStateInfo

State: pennsylvania

Population: 13002700

Nickname: Keystone State

Capital: Harrisburg

Song: Pennsylvania

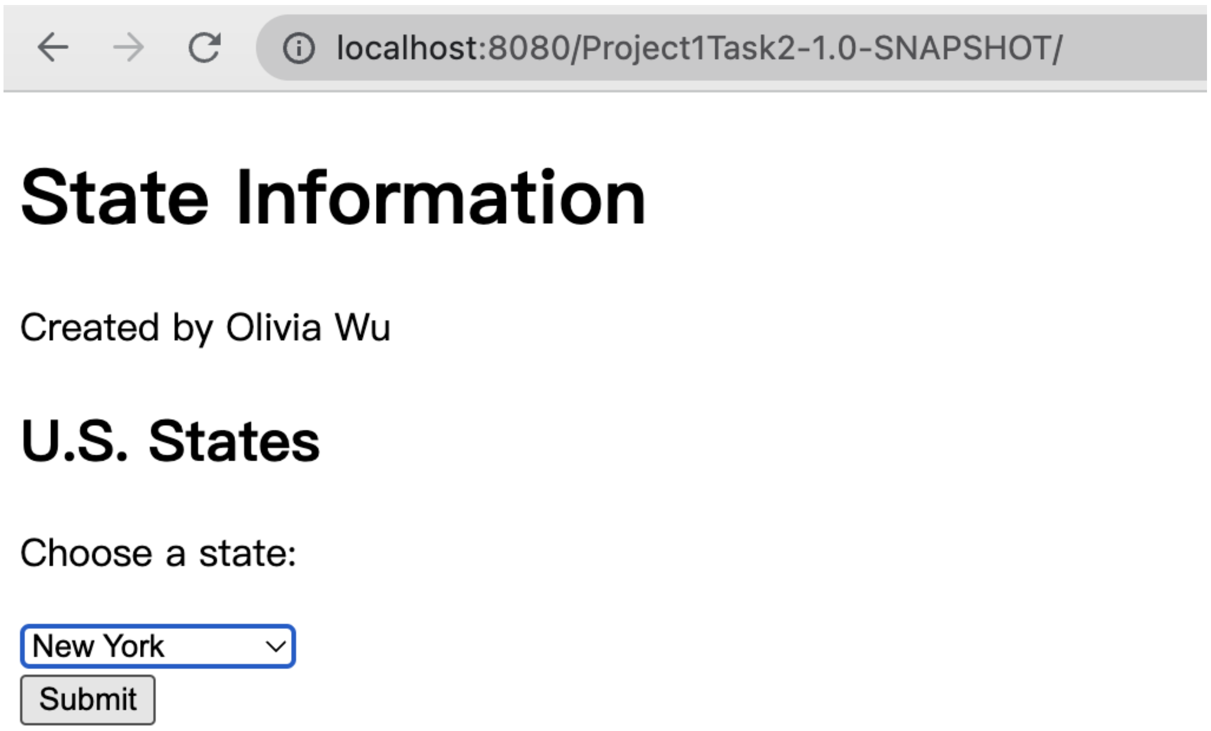
Flower:

Credit: https://statesymbolsusa.org/categories/flower

Flag:

Credit: https://states101.com/flags

- select New York and ouput page for it

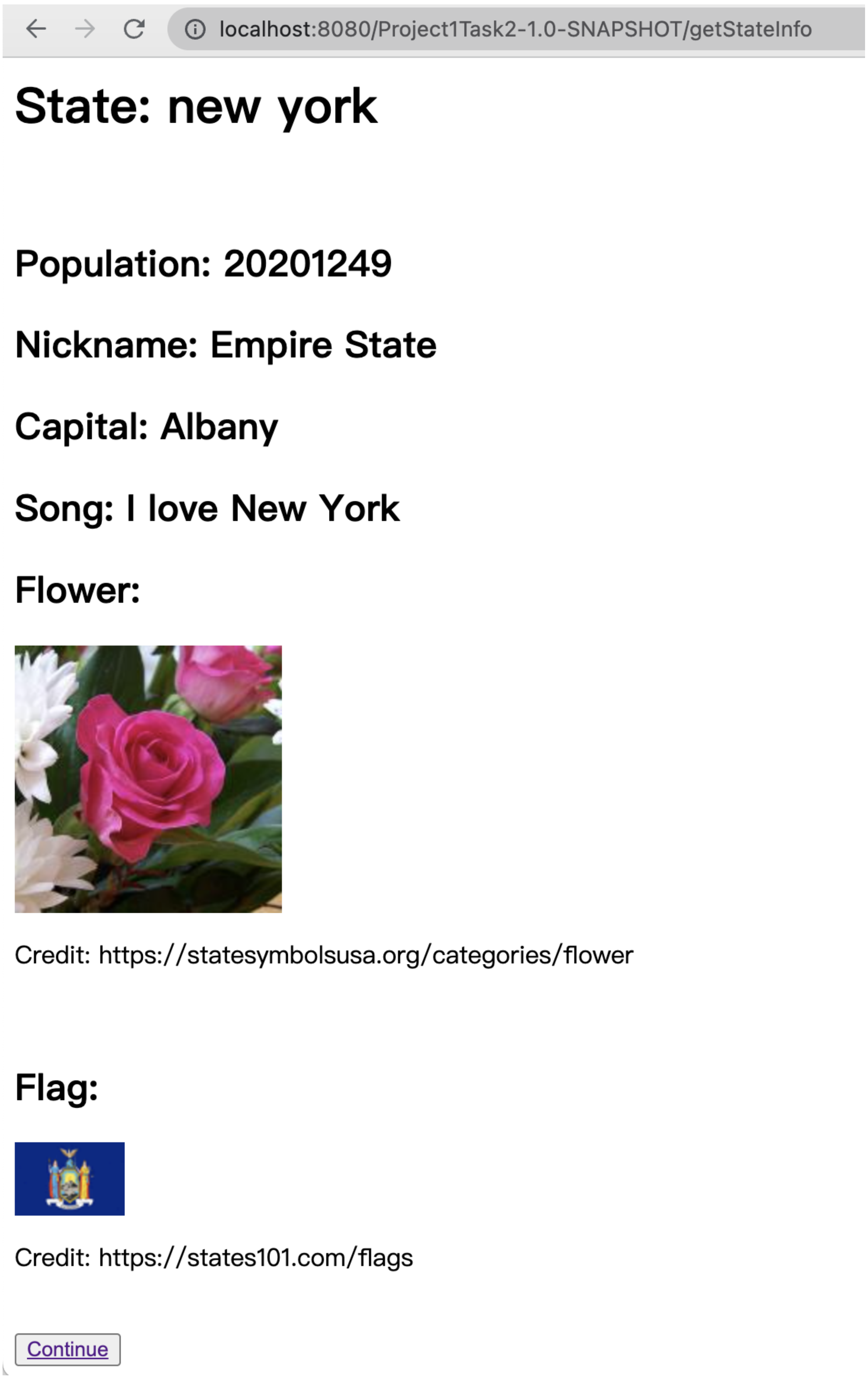

localhost:8080/Project1Task2-1.0-SNAPSHOT/

State Information

Created by Olivia Wu

U.S. States

Choose a state:



localhost:8080/Project1Task2-1.0-SNAPSHOT/getStateInfo

State: new york

Population: 20201249

Nickname: Empire State

Capital: Albany

Song: I love New York

Flower:

Credit: https://statesymbolsusa.org/categories/flower

Flag:

Credit: https://states101.com/flags

Continue

- doPost part in my StateInfoServlet.java code:

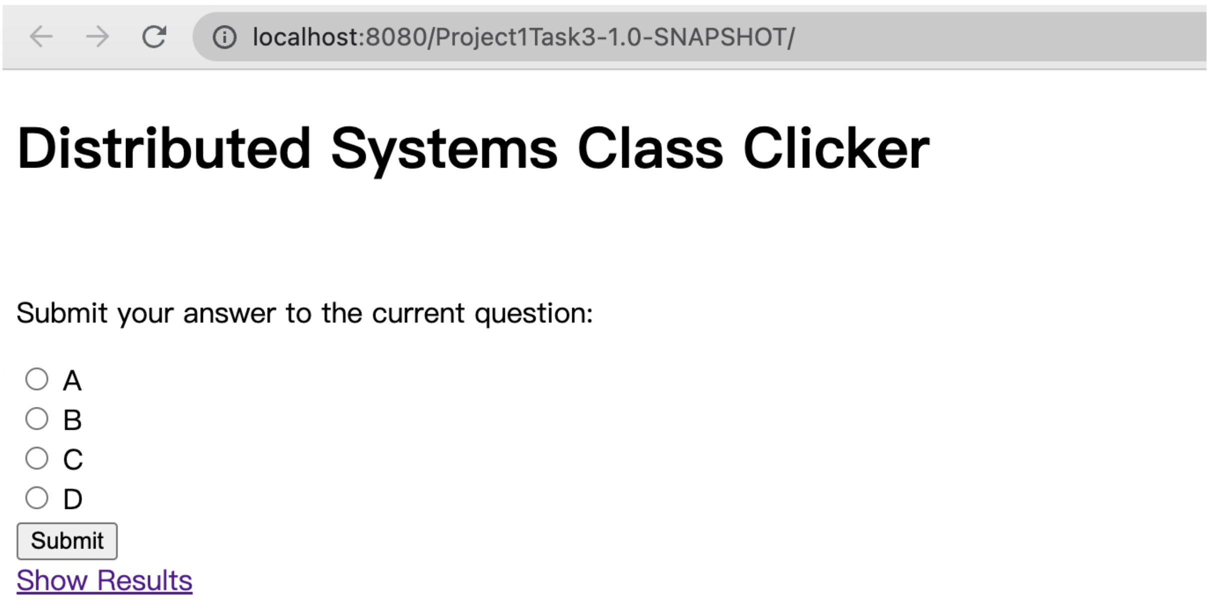
```
@Override
    public void doPost(HttpServletRequest request, HttpServletResponse response) throw
s ServletException, IOException {
        String state = request.getParameter("select");
        /*
        compatible to view on Android/iPhone
         */
        String ua = request.getHeader("User-Agent");
        boolean mobile;
        if (ua != null && ((ua.indexOf("Android") != -1) || (ua.indexOf("iPhone") != -
1))) {
            mobile = true;
            request.setAttribute("doctype", "<!DOCTYPE html PUBLIC \"-//WAPFORUM//DTD
 XHTML Mobile 1.2//EN\" \"http://www.openmobilealliance.org/tech/DTD/xhtml-mobile12.dt
d\">");
        } else {
            mobile = false;
            request.setAttribute("doctype", "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML
 4.01 Transitional//EN\" \"http://www.w3.org/TR/html4/loose.dtd\">");
        }
        String nextView;
        /*
         * Check if the search parameter is present.
         * If not, then give the user instructions and prompt for a search string.
         * If there is a search parameter, then do the search and return the result.
         */
        if (state != null) {
            String picSize = (mobile) ? "mobile" : "desktop";

            request.setAttribute("searchState", state);
            request.setAttribute("population", stateInfoModel.getStatePopulation(stat
e));
            request.setAttribute("nickname", stateInfoModel.getStateNickname(state));
            request.setAttribute("capital", stateInfoModel.getStateCapital(state));
            request.setAttribute("song", stateInfoModel.getStateSong(state));
            request.setAttribute("imgFlower", stateInfoModel.getStateFlowerImage(stat
e));
            request.setAttribute("imgFlag", stateInfoModel.getStateFlagImage(state));
            // Pass the user search string (pictureTag) also to the view.
            nextView = "result.jsp";
        } else {
            // no search parameter so choose the prompt view
            nextView = "index.jsp";
        }
        // Transfer control over the correct "view"
        RequestDispatcher view = request.getRequestDispatcher(nextView);
        view.forward(request, response);
    }
```

- All Web scraping part for different information

```
public class StateInfoModel {

    /**
     * Get nickname for input state in https://www.britannica.com/topic/List-of-nickna
mes-of-U-S-States-2130544.
     * @param state
     * @return nickname string
     */
    public String getStateNickname(String state) {
        String nickNameURL = "https://www.britannica.com/topic/List-of-nicknames-of-U-
S-States-2130544";
        String response = fetch(nickNameURL, "TLSV1.3"); // SSL

        String nickname = "";
        Document doc = Jsoup.parse(response);
        // get the nickname
        Element table = doc.select("table").get(0); //select the first table.
        Elements rows = table.select("tr");

        //first row is the col names so skip it.
        for (int i = 1; i < rows.size(); i++) {
            Element row = rows.get(i);
            Elements cols = row.select("td");
            String key = cols.get(0).text();
            String value = cols.get(1).text();
            if (key.equalsIgnoreCase(state)) {
                nickname = value;
                break;
            }
        }
        return nickname;
    }

    /**
     * Get capital for input state by iterating through all elements in the url.
     * @param state
     * @return capital string
     */
    public String getStateCapital(String state) {
        String URL = "https://gisgeography.com/united-states-map-with-capitals/";
        String response = fetch(URL, "TLSV1.3"); // SSL

        String capital = "";
        Document doc = Jsoup.parse(response);
        // go through all elements to get capital for input state
        Elements divs = doc.getElementsByClass("kt-inside-inner-col");
        for (int index = 2; index < 4; index++){
            Element element1 = divs.get(index);
            Elements p = element1.select("p");
            String[] textSplitResult = p.first().html().split("<br>");
            for (String text:textSplitResult) {
                String key = text.substring(0,text.indexOf("(")-1);
                String value = text.substring(text.indexOf("(")+1, text.indexOf(")"));
                if (key.equalsIgnoreCase(state)) {
                    capital = value;
                    break;
                }
            }
        }
        return capital;
    }

    /**
     * Get state song via https://50states.com/songs/
     * @param state
     * @return song string
     */
    public String getStateSong(String state) {
        String URL = "https://50states.com/songs/";
        String response = fetch(URL, "TLSV1.3"); // SSL

        String song = "";
        Document doc = Jsoup.parse(response);
        // go through all elements to get capital for input state
        Element table = doc.select("table").get(0); //select the first table.
        Elements lists = table.select("li");
        // here start from index = 0, pay attention to nickname part.
        for (int i = 0; i < lists.size(); i++) { //first row is the col names so skip
 it.
            Element list = lists.get(i);
            String key = list.select("dt").text();
            String value = list.select("dd").text();
            if (key.equalsIgnoreCase(state)) {
                song = value;
                break;
            }
        }
        return song;
    }

    /**
     * Get population via API https://api.census.gov/data/2020/dec/pl?get=NAME,P1_001N
&for=state: [put the state FIPS code here]&key=[put your API key here]
     * and my key here is : 16ae7ef40a91c0902a387b0d94a7b73e89cded96
     * @param state
     * @return population string
     */
    public String getStatePopulation(String state) {
        String code = getStateCode(state);
        String URL = "https://api.census.gov/data/2020/dec/pl?get=NAME,P1_001N&for=sta
te:"+code+"&key=16ae7ef40a91c0902a387b0d94a7b73e89cded96";
        String response = fetch(URL, "TLSV1.3"); // SSL

        String population = response.split(",")[4];
        population = population.replaceAll("^\"|\"$", "");
        return population;
    }

    /**
     * Get image of state flower from https://statesymbolsusa.org/categories/flower.
     * @param state
     * @return url of flower image with proper size
     */
    public String getStateFlowerImage(String state) {
        String URL = "https://statesymbolsusa.org/categories/flower";
        String response = fetch(URL, "TLSV1.3"); // SSL

        String imgURL = "";
        Document doc = Jsoup.parse(response);
        // go through all elements to get capital for input state
        Elements imageElements = doc.select("img");
        Elements elements = doc.select("div.views-field.views-field-title-1 > span.fie
ld-content");
        for (int i = 0; i < elements.size();i++){
            String absoluteUrl = imageElements.get(i+2).absUrl("src");  //absolute URL
on src
            String key = elements.get(i).text();
            if (key.equalsIgnoreCase(state)){
                imgURL = absoluteUrl;
                break;
            }
        }
        return imgURL;
    }


    /**
     * Get image url of state flag from https://states101.com/flags
     * @param state
     * @return url of flag image with proper size
     */
    public String getStateFlagImage(String state) {
        String URL = "https://states101.com/flags";
        String response = fetch(URL, "TLSV1.3"); // SSL

        String imgURL = "";
        Document doc = Jsoup.parse(response);
        // go through all elements to get flag image
        Elements imageElements = doc.select("img");
        Elements elements = doc.select("div.col-md-3.col-sm-4.col-xs-6 > a");
        for (int i = 0; i < elements.size();i++){
            String absoluteUrl = imageElements.get(i+2).attr("src");  //absolute URL o
n src
            String key = elements.get(i).text();
            if (key.equalsIgnoreCase(state)){
                imgURL = "https://www.states101.com"+absoluteUrl;
                break;
            }
        }
        //imgURL = pictureSize(imgURL, picSize);
        return imgURL;
    }

    /**
     * Map state with its code as is shown in fips.csv.
     * @param state
     * @return state code as an integer
     */
    private String getStateCode(String state){
        HashMap<String, String> stateMap = new HashMap<>();
        stateMap.put("Alabama".toLowerCase(), "01");
        stateMap.put("Alaska".toLowerCase(), "02");
        stateMap.put("Arizona".toLowerCase(), "04");
        stateMap.put("Arkansas".toLowerCase(),"05");
        stateMap.put("California".toLowerCase(),"06");
        stateMap.put("Colorado".toLowerCase(),"08");
        stateMap.put("Connecticut".toLowerCase(),"09");
        stateMap.put("Delaware".toLowerCase(), "10");
        stateMap.put("Florida".toLowerCase(),"12");
        stateMap.put("Georgia".toLowerCase(), "13");
        stateMap.put("Hawaii".toLowerCase(),"15");
        stateMap.put("Idaho".toLowerCase(),"16");
        stateMap.put("Illinois".toLowerCase(),"17");
        stateMap.put("Indiana".toLowerCase(), "18");
        stateMap.put("Iowa".toLowerCase(),"19");
        stateMap.put("Kansas".toLowerCase(),"20");
        stateMap.put("Kentucky".toLowerCase(),"21");
        stateMap.put("Louisiana".toLowerCase(),"22");
        stateMap.put("Maine".toLowerCase(),"23");
        stateMap.put("Maryland".toLowerCase(),"24");
        stateMap.put("Massachusetts".toLowerCase(),"25");
        stateMap.put("Michigan".toLowerCase(),"26");
        stateMap.put("Minnesota".toLowerCase(),"27");
        stateMap.put("Mississippi".toLowerCase(),"28");
        stateMap.put("Missouri".toLowerCase(),"29");
        stateMap.put("Montana".toLowerCase(), "30");
        stateMap.put("Nebraska".toLowerCase(),"31");
        stateMap.put("Nevada".toLowerCase(),"32");
        stateMap.put("New Hampshire".toLowerCase(),"33");
        stateMap.put("New Jersey".toLowerCase(),"34");
        stateMap.put("New Mexico".toLowerCase(),"35");
        stateMap.put("New York".toLowerCase(),"36");
        stateMap.put("North Carolina".toLowerCase(),"37");
        stateMap.put("North Dakota".toLowerCase(),"38");
        stateMap.put("Ohio".toLowerCase(),"39");
        stateMap.put("Oklahoma".toLowerCase(),"40");
        stateMap.put("Oregon".toLowerCase(),"41");
        stateMap.put("Pennsylvania".toLowerCase(),"42");
        stateMap.put("Rhode Island".toLowerCase(),"44");
        stateMap.put("South Carolina".toLowerCase(),"45");
        stateMap.put("South Dakota".toLowerCase(),"46");
        stateMap.put("Tennessee".toLowerCase(),"47");
        stateMap.put("Texas".toLowerCase(),"48");
        stateMap.put("Utah".toLowerCase(),"49");
        stateMap.put("Vermont".toLowerCase(),"50");
        stateMap.put("Virginia".toLowerCase(),"51");
        stateMap.put("Washington".toLowerCase(),"53");
        stateMap.put("West Virginia".toLowerCase(),"54");
        stateMap.put("Wisconsin".toLowerCase(),"55");
        stateMap.put("Wyoming".toLowerCase(),"56");

        return stateMap.get(state);
    }


    /**
     * Open url without SSLHandshakeException.
     * @param searchURL
     * @param certType
     * @return response from the url
     */
    private String fetch(String searchURL, String certType) {
        try {
            // Create trust manager, which lets you ignore SSLHandshakeExceptions
            createTrustManager(certType);
        } catch (KeyManagementException ex) {
            System.out.println("Shouldn't come here: ");
            ex.printStackTrace();
        } catch (NoSuchAlgorithmException ex) {
            System.out.println("Shouldn't come here: ");
            ex.printStackTrace();
        }

        String response = "";
        try {
            URL url = new URL(searchURL);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();

            // Read all the text returned by the server
            BufferedReader in = new BufferedReader(new InputStreamReader(connection.ge
tInputStream(), "UTF-8"));
            String str;
            // Read each line of "in" until done, adding each to "response"
            while ((str = in.readLine()) != null) {
                // str is one line of text readLine() strips newline characters
                response += str;
            }
            in.close();
        } catch (IOException e) {
            System.err.println("Something wrong with URL");
            return null;
        }
        return response;
    }

    private void createTrustManager(String certType) throws KeyManagementException, No
SuchAlgorithmException{
        /**
         * Annoying SSLHandShakeException. After trying several methods, finally this
         * seemed to work.
         * Taken from: http://www.nakov.com/blog/2009/07/16/disable-certificate-valida
tion-in-java-ssl-connections/
         */
        // Create a trust manager that does not validate certificate chains
        TrustManager[] trustAllCerts = new TrustManager[] {new X509TrustManager() {
            public X509Certificate[] getAcceptedIssuers() {
                return null;
            }
            public void checkClientTrusted(X509Certificate[] certs, String authType) {
            }
            public void checkServerTrusted(X509Certificate[] certs, String authType) {
            }
        }
        };

        // Install the all-trusting trust manager
        SSLContext sc = SSLContext.getInstance(certType);
        sc.init(null, trustAllCerts, new java.security.SecureRandom());
        HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());

        // Create all-trusting host name verifier
        HostnameVerifier allHostsValid = new HostnameVerifier() {
            public boolean verify(String hostname, SSLSession session) {
                return true;
            }
        };
        // Install the all-trusting host verifier
        HttpsURLConnection.setDefaultHostnameVerifier(allHostsValid);
    }

}
```

Task 3

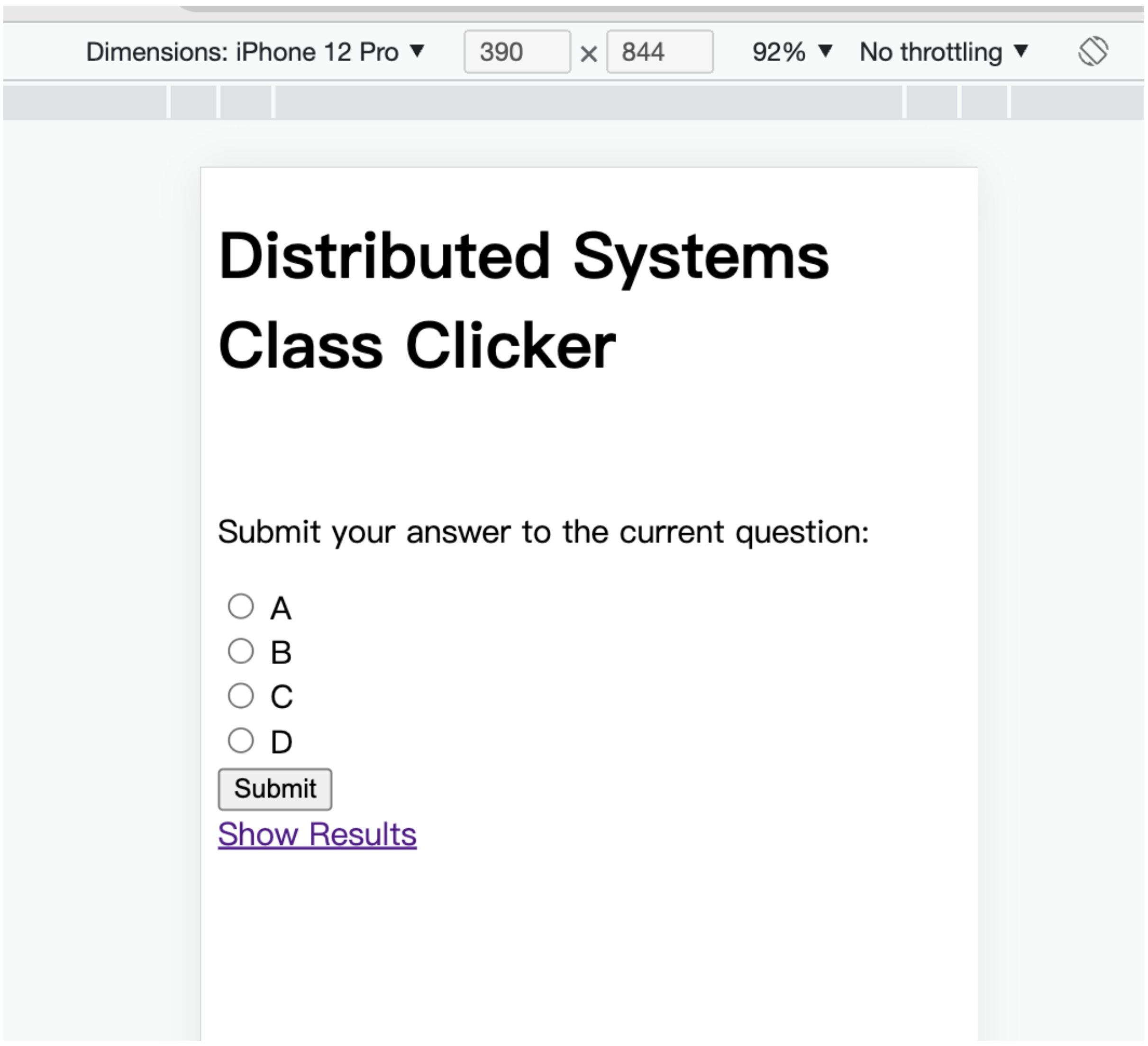
- Input Page in both pc mode and iPhone mode



Distributed Systems Class Clicker

Submit your answer to the current question:

○ A
○ B
○ C
○ D


Dimensions: iPhone 12 Pro ▼ 390 × 844 92% ▼ No throttling ▼

Distributed Systems Class Clicker

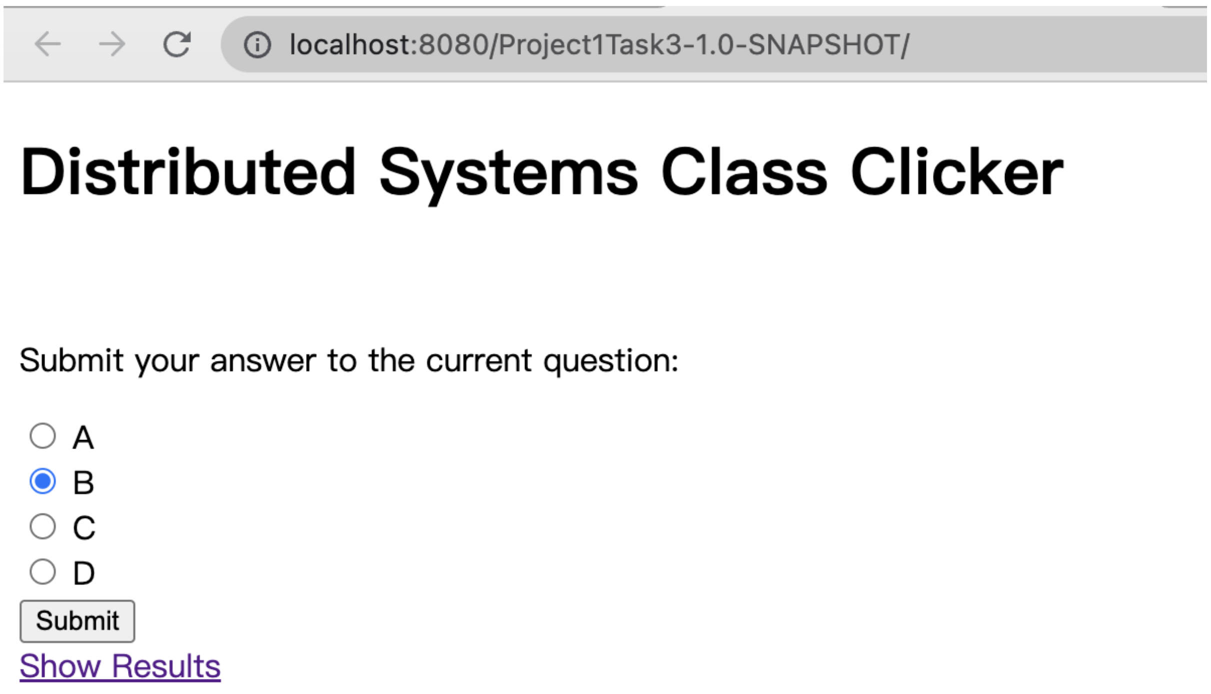
Submit your answer to the current question:

- ◯ A
- ◯ B
- ◯ C
- ◯ D

Submit

Show Results

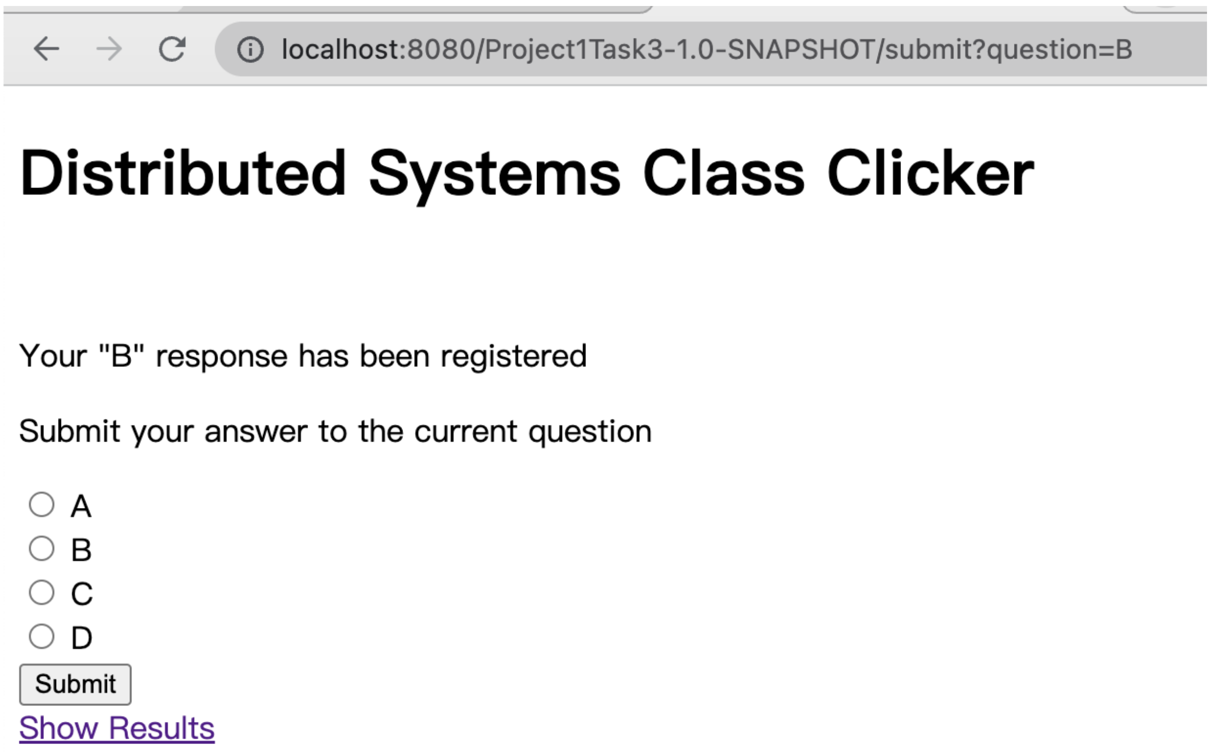
- output page (one vote) in both PC and iPhone mode



Distributed Systems Class Clicker

Submit your answer to the current question:

- ( ) A
- [✓ B]
- ( ) C
- ( ) D

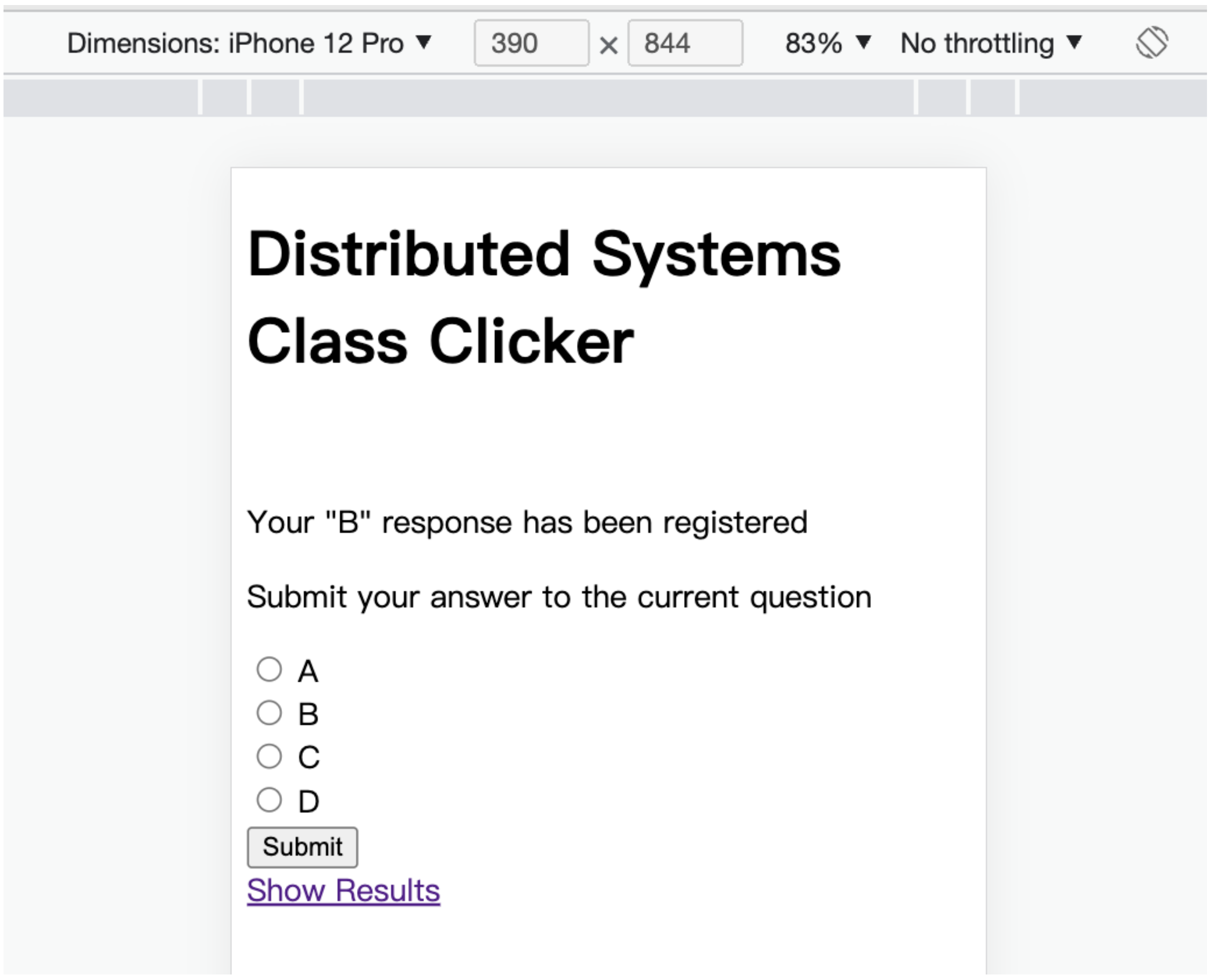


Distributed Systems Class Clicker

Your "B" response has been registered

Submit your answer to the current question

- ( ) A
- ( ) B
- ( ) C
- ( ) D

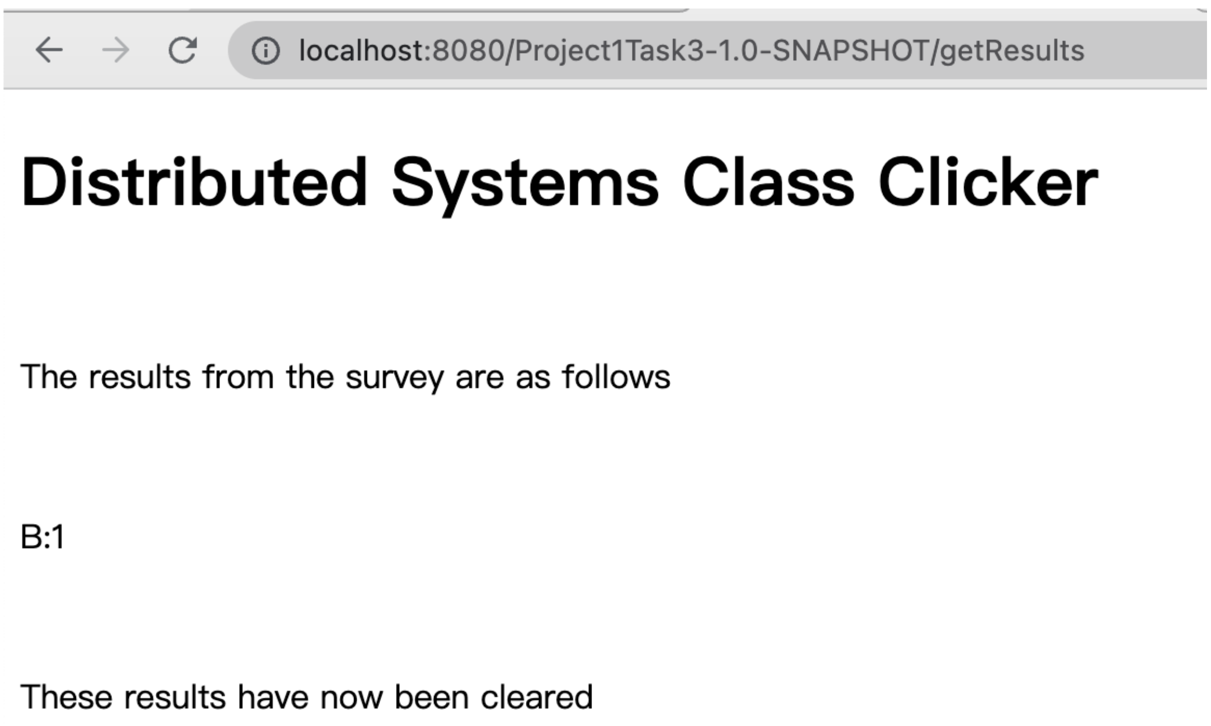

Dimensions: iPhone 12 Pro ▼ 390 × 844 83% ▼ No throttling ▼

Distributed Systems Class Clicker

Your "B" response has been registered

Submit your answer to the current question

- results page in both modes

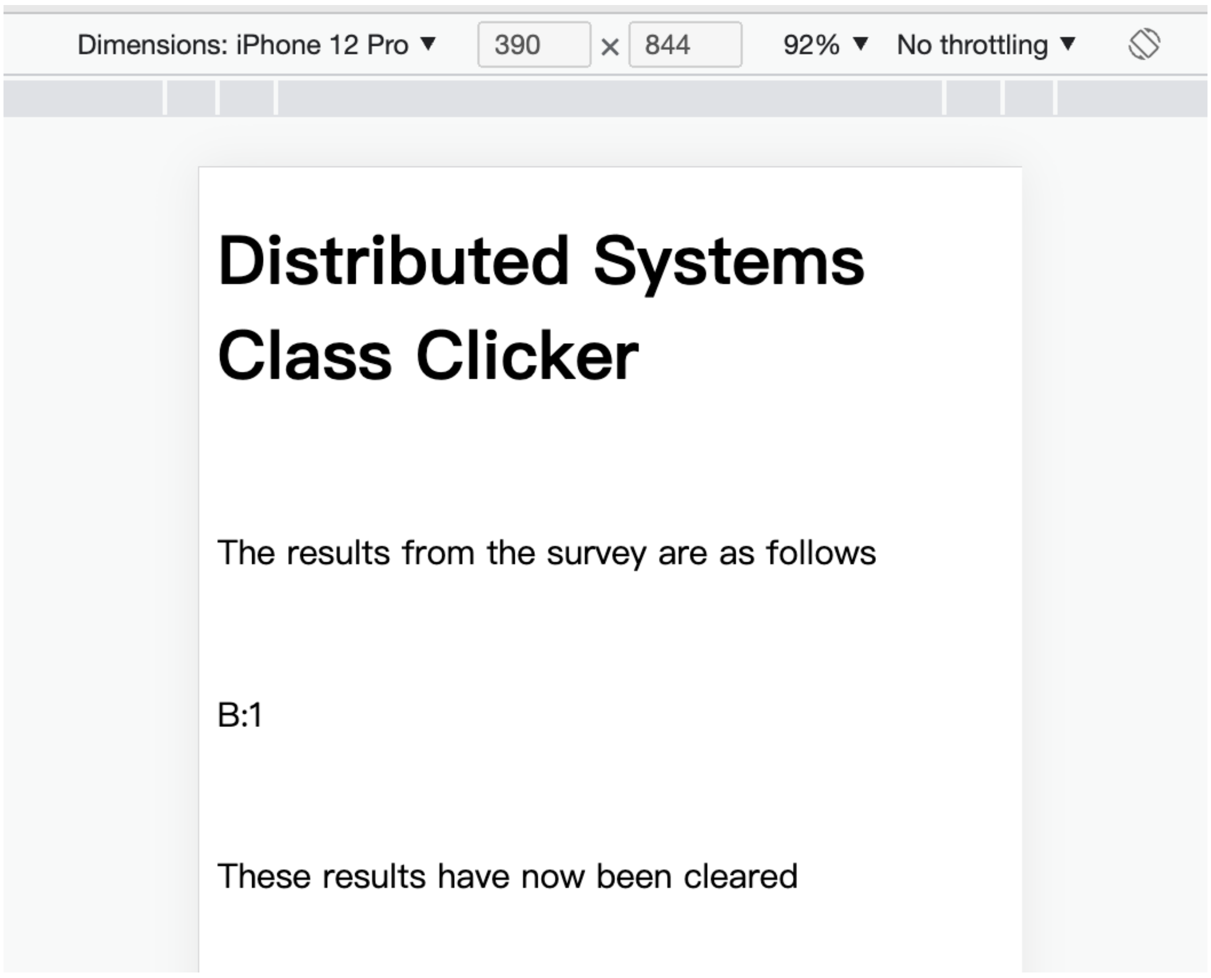

localhost:8080/Project1Task3-1.0-SNAPSHOT/getResults

Distributed Systems Class Clicker

The results from the survey are as follows

B:1

These results have now been cleared


Dimensions: iPhone 12 Pro ▼ 390 × 844 92% ▼ No throttling ▼

Distributed Systems Class Clicker

The results from the survey are as follows

B:1

These results have now been cleared

- code snippets: Java code that produces the output page and the results page

```
//ClickerModel.java
package ds.project1task3;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.stream.Collectors;
public class ClickerModel {
    private HashMap<String, Integer> hashMap = new HashMap<>();
    // save answer distribution to the hashmap
    public void updateResults(String answer){
        if (hashMap.containsKey(answer)) {
            hashMap.put(answer, hashMap.get(answer)+1);
        } else {
            hashMap.put(answer, 1);
        }
    }
    // get results
    public HashMap<String, Integer> getResults() {
        return hashMap;
    }
    // clear the results
    public void clear() {
        this.hashMap = new HashMap<>();
    }
}
```

```
//ClickerServlet
package ds.project1task3;

import java.io.*;
import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;

@WebServlet(name = "helloServlet", urlPatterns = {"/submit", "/getResults"})
public class ClickerServlet extends HttpServlet {
    private ClickerModel clickerModel = null;
    @Override
    public void init() {
         clickerModel = new ClickerModel();
    }
    @Override
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException,IOException {
        System.out.println(request.getServletPath());
        //submit page
        if(request.getServletPath().equals("/submit")) {
            String answer = request.getParameter("question");
            clickerModel.updateResults(answer);
            request.setAttribute("answer",answer);
            RequestDispatcher view = request.getRequestDispatcher("index2.jsp");
            view.forward(request, response);
        }
        //getResults page
        if(request.getServletPath().equals("/getResults")) {
            HashMap<String, Integer> results = clickerModel.getResults();
            clickerModel.clear();
            request.setAttribute("result", results);
            RequestDispatcher view = request.getRequestDispatcher("result.jsp");
            view.forward(request, response);
        }
    }

}
```