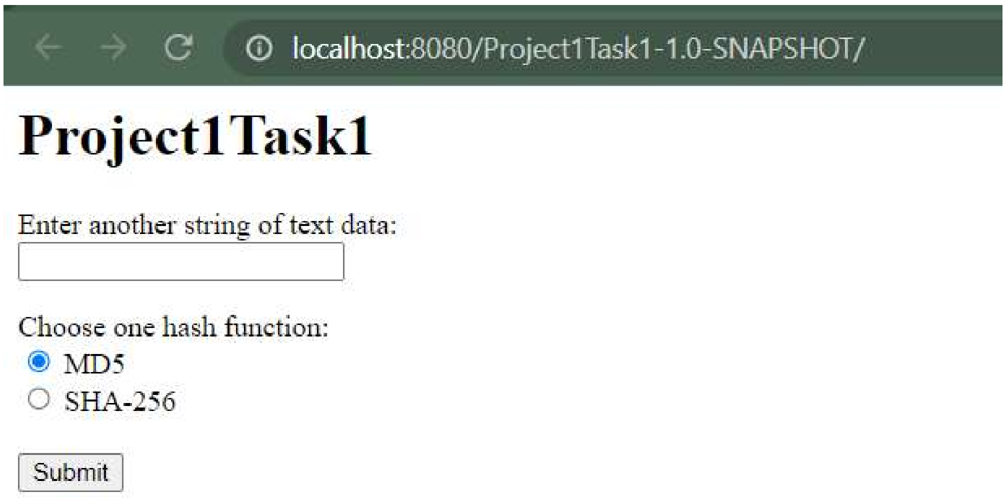
Project 1

Task 1:

1. **Screen shots** of input, MD5 and SHA-256 output, both in hex and base 64

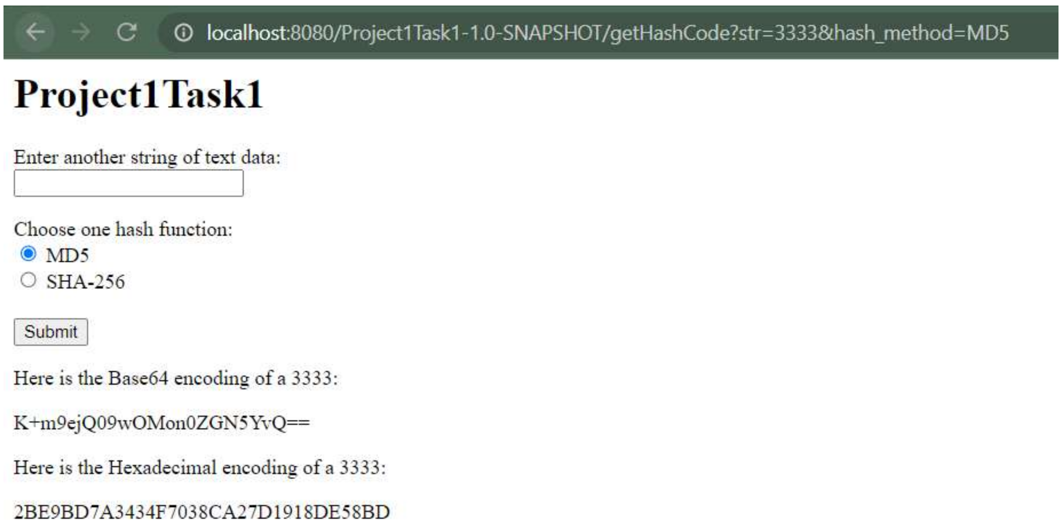


localhost:8080/Project1Task1-1.0-SNAPSHOT/

Project1Task1

Enter another string of text data:

Choose one hash function:
[✓ MD5]
SHA-256



localhost:8080/Project1Task1-1.0-SNAPSHOT/getHashCode?str=3333&hash_method=MD5

Project1Task1

Enter another string of text data:

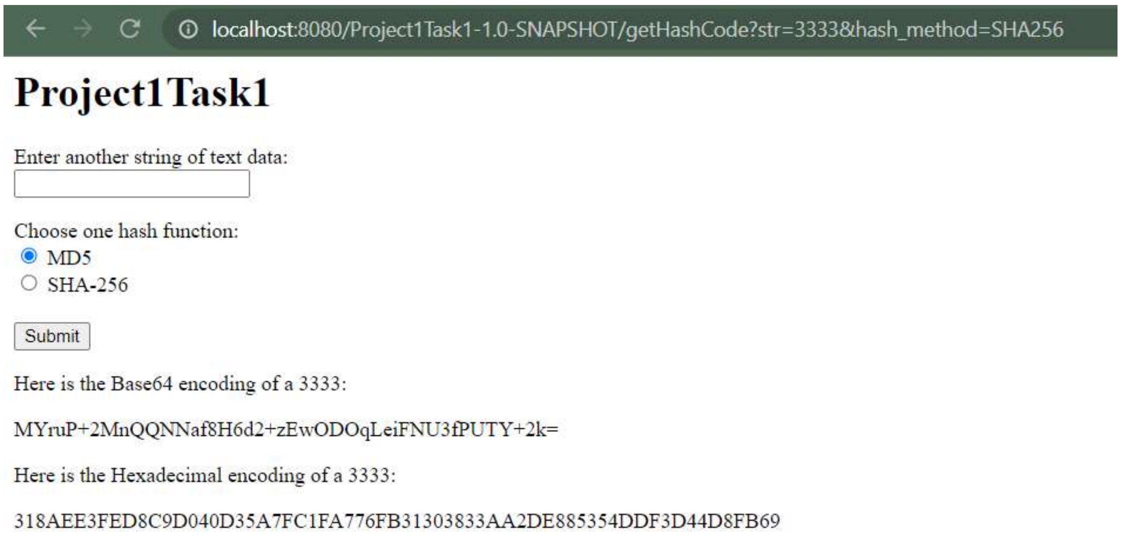
Choose one hash function:
[✓ MD5]
SHA-256

Here is the Base64 encoding of a 3333:

K+m9ejQ09wOMon0ZGN5YvQ==

Here is the Hexadecimal encoding of a 3333:

2BE9BD7A3434F7038CA27D1918DE58BD


localhost:8080/Project1Task1-1.0-SNAPSHOT/getHashCode?str=3333&hash_method=SHA256

Project1Task1

Enter another string of text data:

Choose one hash function:
[✓ MD5]
SHA-256

Here is the Base64 encoding of a 3333:

MYruP+2MnQQNNaf8H6d2+zEwODOqLeiFNU3fPUTY+2k=

Here is the Hexadecimal encoding of a 3333:

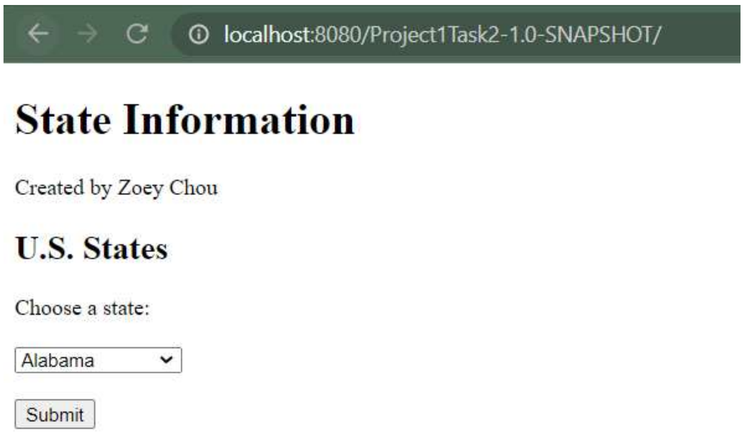
318AEE3FED8C9D040D35A7FC1FA776FB31303833AA2DE885354DDF3D44D8FB69

2. **Code snippets** of computation of each hash

```java
String search = request.getParameter("str");
String hashMethod = request.getParameter("hash_method");
byte[] hashResult;
//determine hashcode method
if (search != null) {
    //hashcode method equals to MD5
    if (hashMethod.equals("MD5")){
        try {
            MessageDigest md = MessageDigest.getInstance("MD5");
            hashResult =md.digest(search.getBytes("UTF-8"));
            //print the base64 encoding
            base64 =
javax.xml.bind.DatatypeConverter.printBase64Binary(hashResult);
            //print hexadecimal encoding
            hex = javax.xml.bind.DatatypeConverter.printHexBinary(hashResult);
            //set base64 and hexadecimal encoding attribute back to the view
            request.setAttribute("base64",base64);
            request.setAttribute("hex",hex);
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
        //hashcode method equals to SHA-256
    } else if (hashMethod.equals("SHA256")) {
        try {
            MessageDigest sha = MessageDigest.getInstance("SHA-256");
            hashResult = sha.digest(search.getBytes("UTF-8"));
            //print the base64 encoding
            base64 =
javax.xml.bind.DatatypeConverter.printBase64Binary(hashResult);
            //print hexadecimal encoding
            hex = javax.xml.bind.DatatypeConverter.printHexBinary(hashResult);
            //set base64 and hexadecimal encoding attribute back to the view
            request.setAttribute("base64",base64);
            request.setAttribute("hex",hex);
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }
}
```

Task 2:

1. **Screen shots** of input page, drop-down menu, output page for Pennsylvania and New York.



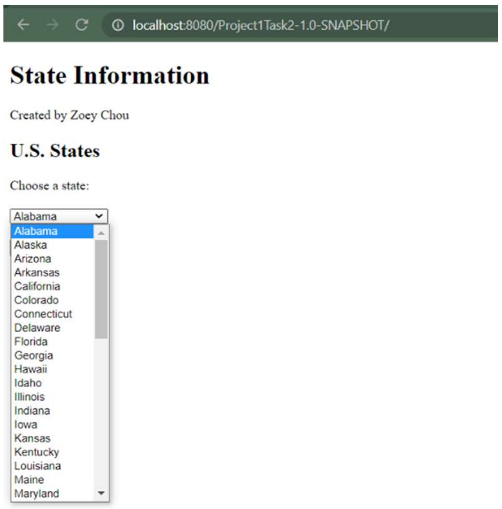
localhost:8080/Project1Task2-1.0-SNAPSHOT/

State Information

Created by Zoey Chou

U.S. States

Choose a state:



localhost:8080/Project1Task2-1.0-SNAPSHOT/

State Information

Created by Zoey Chou

U.S. States

Choose a state:

localhost:8080/Project1Task2-1.0-SNAPSHOT/getAnStateInformation?state=Pennsylvania&submit=Submit

State: Pennsylvania

Population: 13002700

Nickname: Keystone State

Capital: Harrisburg

Song: Pennsylvania

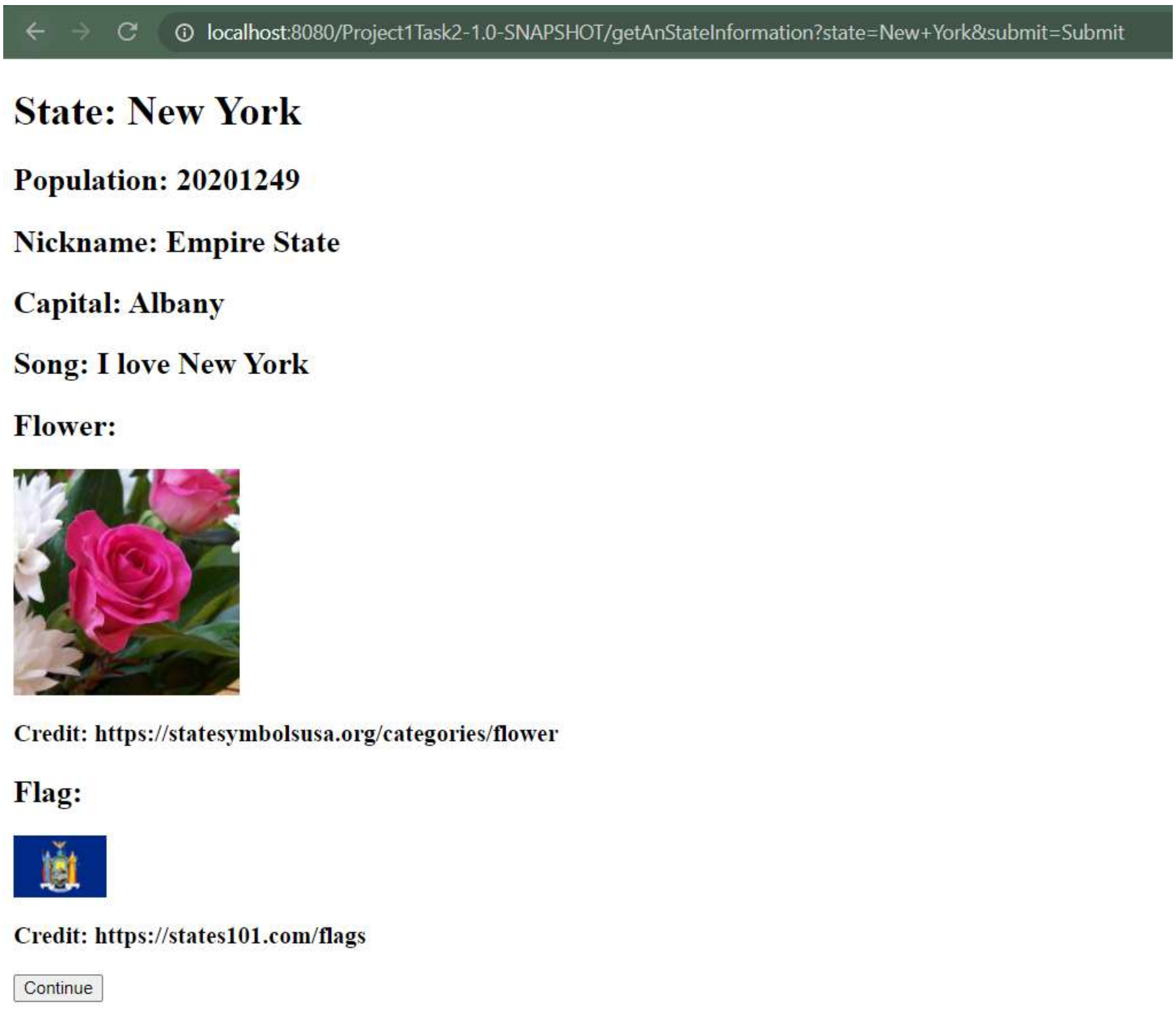
Flower:

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

Flag:

Credit: https://states101.com/flags

Continue



localhost:8080/Project1Task2-1.0-SNAPSHOT/getAnStateInformation?state=New+York&submit=Submit

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

2. **Code snippets** for:

- scraping of nickname

```
//Get NICKNAME data of the selected state
public String doNicknameSearch(String state) {
    String statename = "";
    String nickname = "";
    try {
        //fetch content from the web and parse them into document
        Document doc = Jsoup.connect("https://britannica.com/topic/List-of-
nicknames-of-U-S-States-2130544").get();
        Elements stateClass = doc.select("div.md-drag.md-table-wrapper tbody
tr");
        for (int i = 0; i < stateClass.size(); i++) {
            Element tr = stateClass.get(i);
            //get state name in web content
            statename = tr.getElementsByClass("md-crosslink").get(0).html();
            //check state name and get nickname in web content
            if (statename.equals(state)) {
                nickname = tr.getElementsByTag("td").get(1).html();
                break;
            }
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
```

```
        return nickname;
    }
```

- scraping of capital

```
//Get CAPITAL data of the selected state
public String doCapitalSearch(String state) {

        String capital = "";
        try {
            //fetch content from the web and parse them into document
            Document doc = Jsoup.connect("https://gisgeography.com/united-states-map-
with-capitals/").get();
            Elements stateClass = doc.select("#kt-layout-id_3ca99a-56 div div div
p");
            //get state name in web content
            for (int j = 0; j < stateClass.size(); j++) {
                String[] stateCapitalString = stateClass.get(j).html().split("<br>");
                for (int i = 0; i < stateCapitalString.length; i++) {
                    String[] stateCapital = stateCapitalString[i].replace(")",
"").split("\\(");
                    //check state name and get capital in web content
                    if (stateCapital[0].trim().equals(state)) {
                        capital = stateCapital[1].trim();
                        System.out.println(capital);
                    } else {
                        System.out.println(stateCapital[0].trim() + "," +
stateCapital[1].trim());
                    }
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return capital;
    }
```

- scraping of state song

```
//Get SONG data of the selected state
public String doSongSearch(String state)
            throws IOException {
        String song = "";
        String statename = "";

        //fetch content from the web and parse them into document
        Document doc = Jsoup.connect("https://www.50states.com/songs/").get();
        Elements stateClass = doc.select("div.rounded table tbody tr ul li");
        //get state name in web content
        for (int i = 0; i < stateClass.size(); i++) {
            Element dt = stateClass.get(i).select("dt").get(0);
            statename = dt.html();
            Elements dd = stateClass.get(i).select("dd a");
            if (dd.size() > 0) {
                song = dd.get(0).html();
            }
            //check state name and break from the loop, then return song name
            if (statename.equalsIgnoreCase(state)) {
                break;
            }
        }
        return song;
    }
```

- scraping of flower URL

```
//Get FLOWER data of the selected state
public String doFlowerSearch(String state) throws IOException {
    String flower = "";

    //fetch content from the web and parse them into document
    Document doc1 =
Jsoup.connect("https://statesymbolsusa.org/categories/flower").get();
    Elements stateClass1 = doc1.select("div.view-content div.item-list ul li");
    String stateName = "";
    String img = "";

    //get state name in web content
    for (int i = 0; i < stateClass1.size(); i++) {
        Element li = stateClass1.get(i);
        //get state flower image url
        if (li.getElementsByTag("a").size() >= 3) {
            stateName = li.getElementsByTag("a").get(1).html();
            img = li.getElementsByTag("img").get(0).attr("src");
        }
        //check state name and break from the loop, then return flower image url
        if (stateName.equalsIgnoreCase(state)) {
            flower = img;
            break;
        }
    }
    return flower;
}
```

- scraping of flag URL

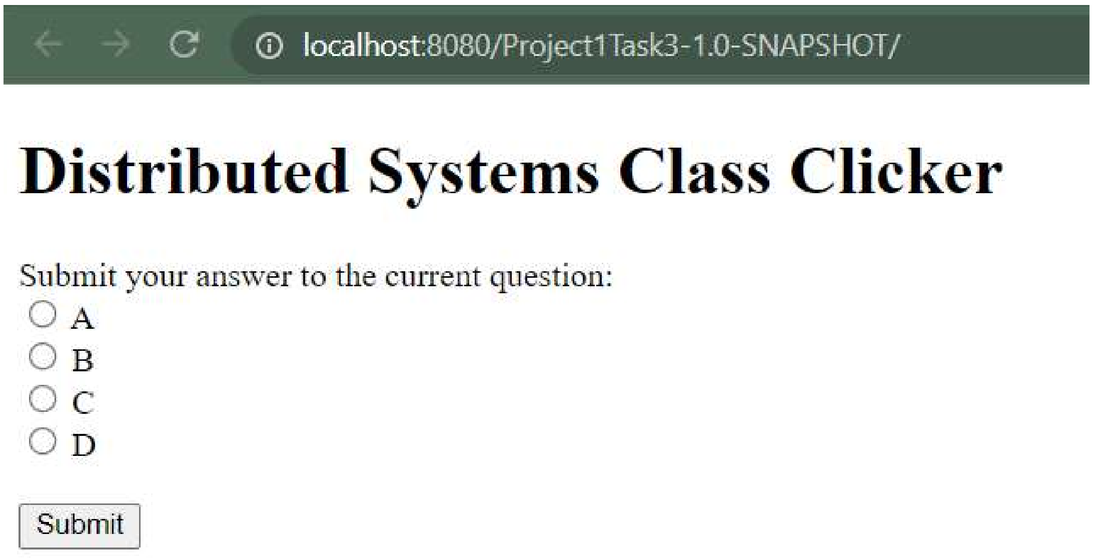
```
//Get FLAG data of the selected state
public String doFlagSearch(String state)throws IOException {
    String flag = "";

    //fetch content from the web and parse them into document
    Document doc2 = Jsoup.connect("https://www.states101.com/flags").get();
    Elements stateClass2 = doc2.select("div.row-fluid div.col-md-10 div.row
div");
    String stateName = "";
    String img = "";

    for (int i = 0; i < stateClass2.size(); i++) {
        Element div = stateClass2.get(i);
        //get state flag image url
        if (div.getElementsByTag("a").size() >= 0) {
            stateName = div.getElementsByTag("b").get(0).html().toString();
            img = div.getElementsByTag("img").get(0).attr("src");
        }
        //check state name and break from the loop, then return flag image url
        if (stateName.equalsIgnoreCase(state)) {
            flag = "https://www.states101.com" + img;
            break;
        }
    }
    return flag;
}
```

- api call for the population

```
//Get POPULATION data of the selected state
public String doPopulationSearch(String state)
        throws UnsupportedEncodingException {
    state = URLEncoder.encode(state, "UTF-8");
    String population = "";
    String stateNum = "";
    try {
        CloseableHttpClient httpClient = HttpClients.createDefault();

        //read state code file
        StringBuilder fileContent = new StringBuilder();
        URL url = this.getClass().getClassLoader().getResource("fips.csv");
        Scanner input = new Scanner(new File(url.getFile()));
        while (input.hasNextLine()) {
            fileContent.append(input.nextLine() + "\n");
        }
        input.close();

        //get state code
        String[] rows = fileContent.toString().split("\n");
        for (int i = 0; i < rows.length; i++) {
            String[] column = rows[i].split(",");
            String st = column[0];
            if (st.equalsIgnoreCase(state)) {
                stateNum = column[1];
                break;
            }
        }

        //set population url with new state code
        String httpUrl =
"https://api.census.gov/data/2020/dec/pl?get=NAME,P1_001N&for=state:" + stateNum
+ "&key=e0accfa0b2d9291776cc07ab926bab1494d291ea";

        //get population data via json type
        HttpGet request = new HttpGet(httpUrl);
        CloseableHttpResponse response = httpClient.execute(request);
        HttpEntity entity = response.getEntity();
        if (entity != null) {
            // return it as a String
            String result = EntityUtils.toString(entity);
            //store as json type
            JsonNode json = new ObjectMapper().readTree(result);
            //extract population data
            int statepopulation = json.get(1).get(1).asInt();
            System.out.println(statepopulation);
            population = String.valueOf(statepopulation);
        }
        response.close();
        httpClient.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return population;
}
```

Task 3:

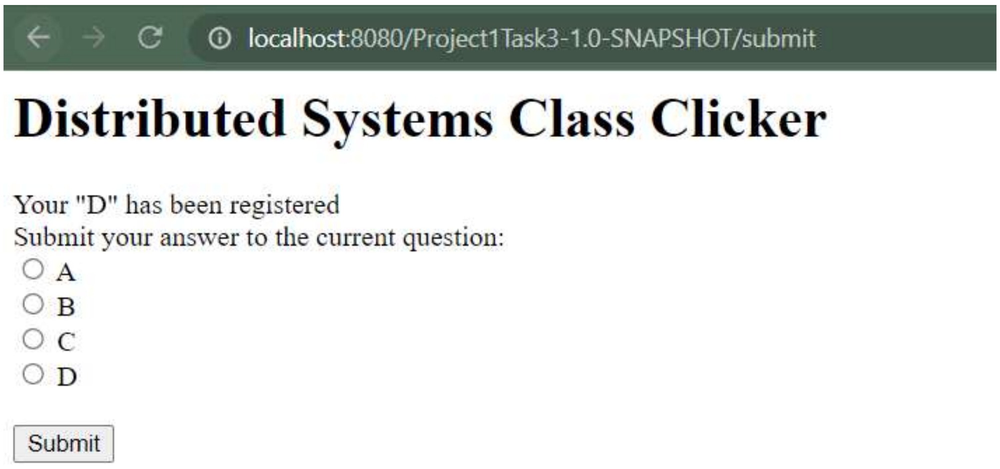
1. **Screen shots** of the input page, output page (one vote), results page



localhost:8080/Project1Task3-1.0-SNAPSHOT/

Distributed Systems Class Clicker

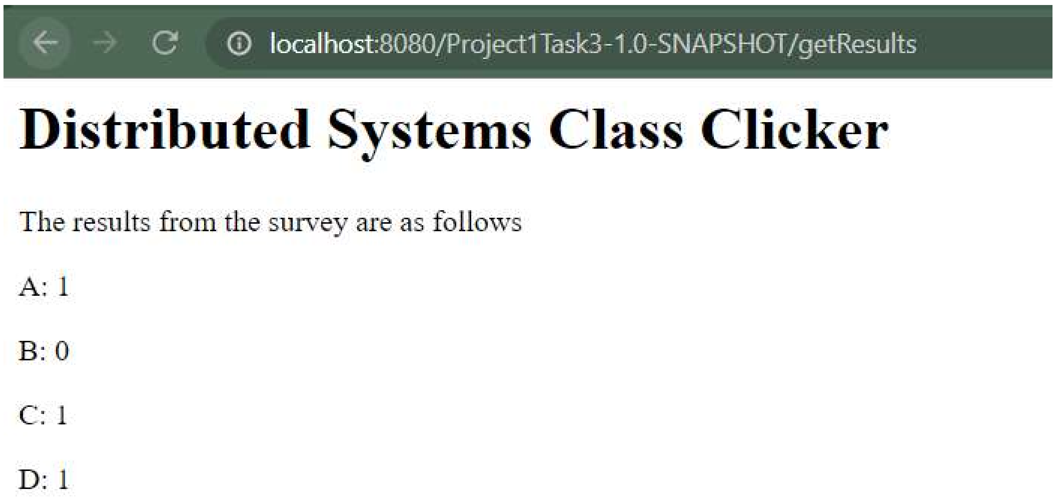
Submit your answer to the current question:



localhost:8080/Project1Task3-1.0-SNAPSHOT/submit

Distributed Systems Class Clicker

Your "D" has been registered
Submit your answer to the current question:



localhost:8080/Project1Task3-1.0-SNAPSHOT/getResults

Distributed Systems Class Clicker

The results from the survey are as follows

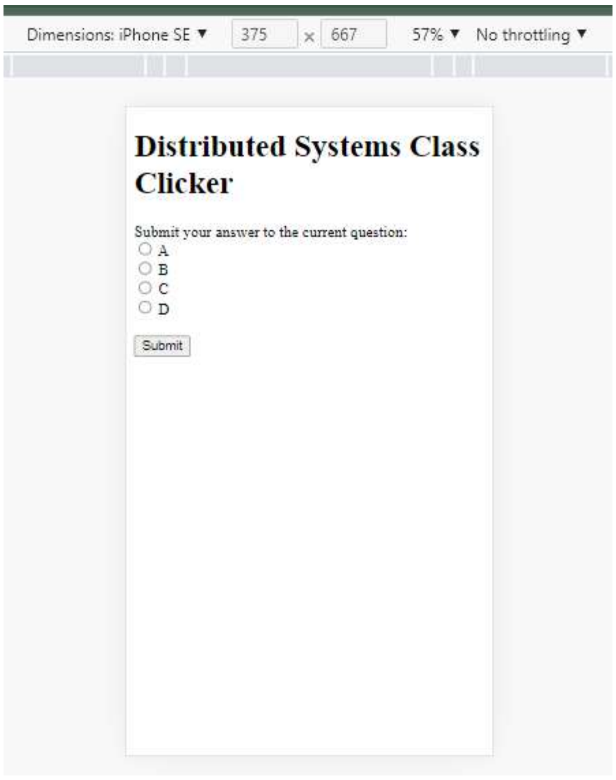
A: 1

B: 0

C: 1

D: 1

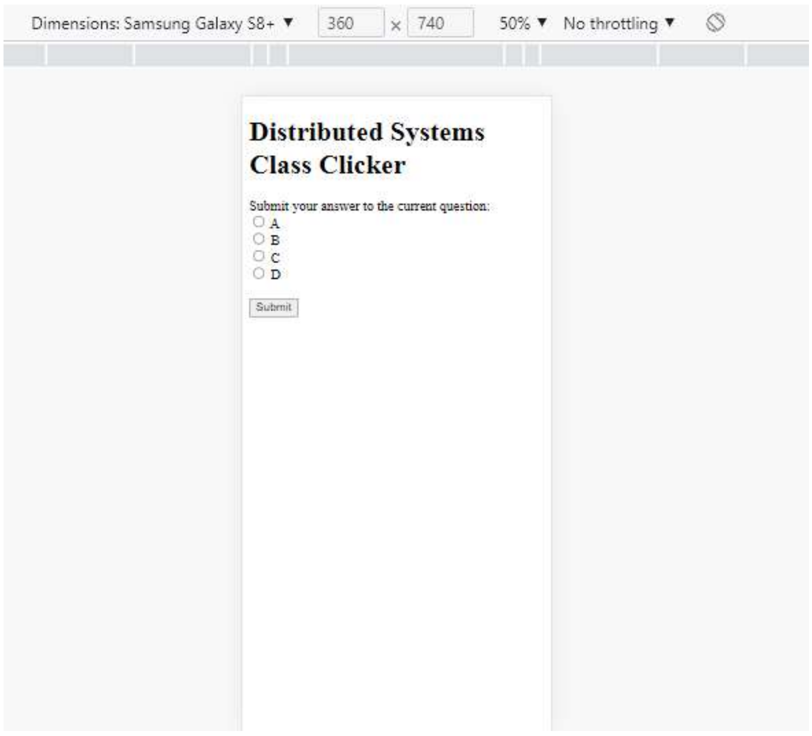
Mobile(iPhone device)


Dimensions: iPhone SE ▼ 375 × 667 57% ▼ No throttling ▼

Distributed Systems Class Clicker

Submit your answer to the current question:
○ A
○ B
○ C
○ D

Mobile(Android device)


Dimensions: Samsung Galaxy S8+ ▼ 360 × 740 50% ▼ No throttling ▼

Distributed Systems Class Clicker

Submit your answer to the current question:
○ A
○ B
○ C
○ D

2. **Code snippets** from the Java code that produces the output page and the results page.

Servlet

```
@Override
protected void doGet(HttpServletRequest request,
                     HttpServletResponse response)
        throws ServletException, IOException {
```

```
    // determine what type of device our user is
    String ua = request.getHeader("User-Agent");

    /*
     *reference:lab2 practice code, @author Joe Mertz
     */
    boolean mobile;
    // prepare the appropriate DOCTYPE for the view pages
    if (ua != null && ((ua.indexOf("Android") != -1) || (ua.indexOf("iPhone") !=
-1))) {
        mobile = true;
        /*
         * This is the latest XHTML Mobile doctype. To see the difference it
         * makes, comment it out so that a default desktop doctype is used
         * and view on an Android or iPhone.
         */
        request.setAttribute("doctype", "<!DOCTYPE html PUBLIC \"-//WAPFORUM//DTD
XHTML Mobile 1.2//EN\" \"http://www.openmobilealliance.org/tech/DTD/xhtml-
mobile12.dtd\">");
    } else {
        mobile = false;
        request.setAttribute("doctype", "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML
4.01 Transitional//EN\" \"http://www.w3.org/TR/html4/loose.dtd\">");
    }
    /*
     *reference end
     */

    String nextView;
    //change view depends on servlet path
    String path = request.getServletPath();
    if (path.equals("/getResults")) {
        //get map data from model
        Map<String, Integer> newMap = vm.getVoteCount();
        //set map data to view
        request.setAttribute("voteCount", newMap);
        //show "result.jsp" view for result
        nextView = "result.jsp";
    } else {
        //show "prompt.jsp" view for initial page and vote submit
        nextView = "prompt.jsp";
    }
    // Transfer control over the correct "view"
    RequestDispatcher view = request.getRequestDispatcher(nextView);
    view.forward(request, response);
}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
    //get vote value from view
    String vote = req.getParameter("vote");
    //based on vote value update vote result in model
    vm.doVoteCount(vote);
    //set vote value(A, B, C, or D) to view
    req.setAttribute("voteMessage", String.format("Your \"%s\" has been
registered", vote));
    //show "prompt.jsp" view
    RequestDispatcher view = req.getRequestDispatcher("prompt.jsp");
    view.forward(req, resp);
}
```

Model

```java
public Map doVoteCount(String nvote) {
    map.put(nvote, map.get(nvote) + 1);
    return map;
}

//get vote result as treemap
public Map getVoteCount() {
    int sum = 0;
    //check if there's no results
    for (int i : map.values()) {
        sum += i;
    }
    if (sum == 0) {
        return null;
    }
    //after showing the result, the stored results are cleared so that a new
question can be posed
    Map<String, Integer> result = new TreeMap<>();
    for (Map.Entry m : map.entrySet()) {
        result.put((String) m.getKey(), (Integer) m.getValue());
        map.put((String) m.getKey(), 0);
    }
    return result;
}
```