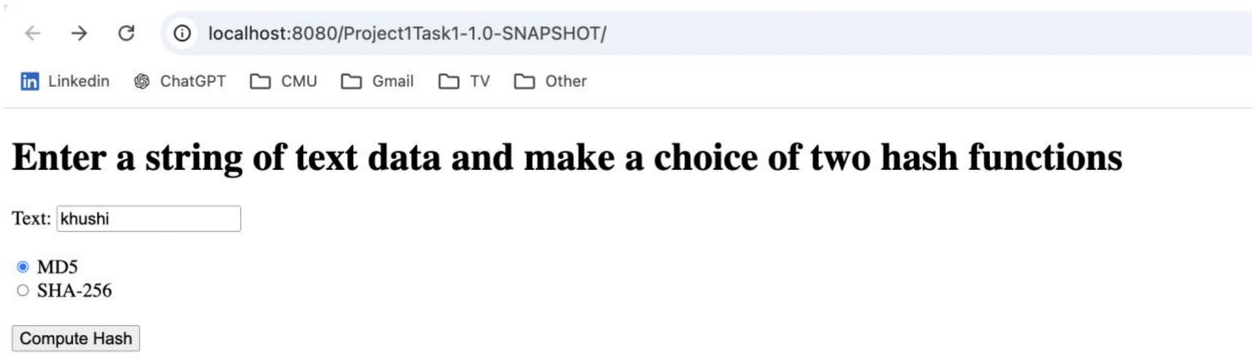
Khushi Bhuwania kbhuwani

PROJECT 1

Task 1

Screenshot:

MD5 Input (default):



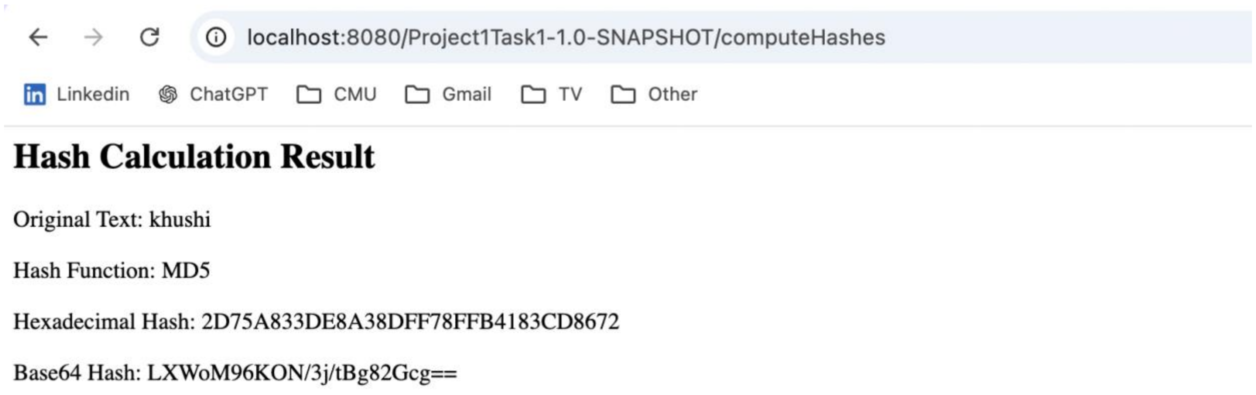
localhost:8080/Project1Task1-1.0-SNAPSHOT/

Enter a string of text data and make a choice of two hash functions

Text: khushi

[✓ MD5]
SHA-256

MD5 Output:



localhost:8080/Project1Task1-1.0-SNAPSHOT/computeHashes

Hash Calculation Result

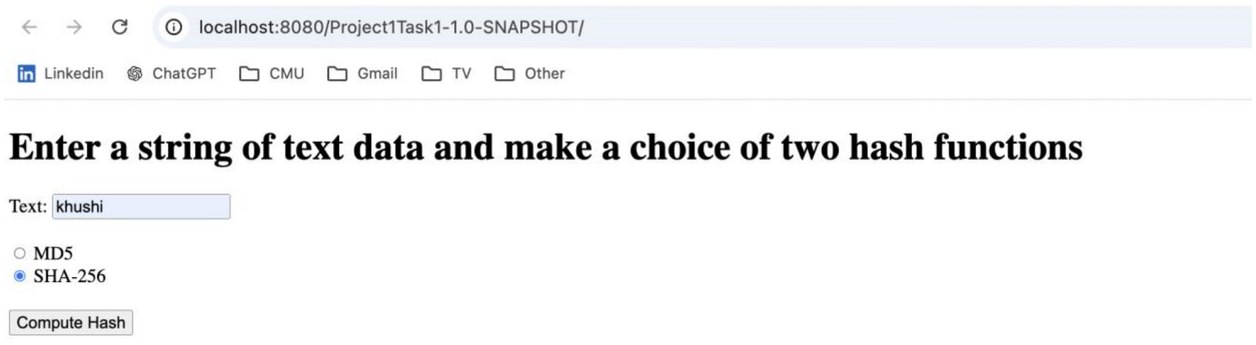
Original Text: khushi

Hash Function: MD5

Hexadecimal Hash: 2D75A833DE8A38DFF78FFB4183CD8672

Base64 Hash: LXWoM96KON/3j/tBg82Gcg==

SHA256 Input:



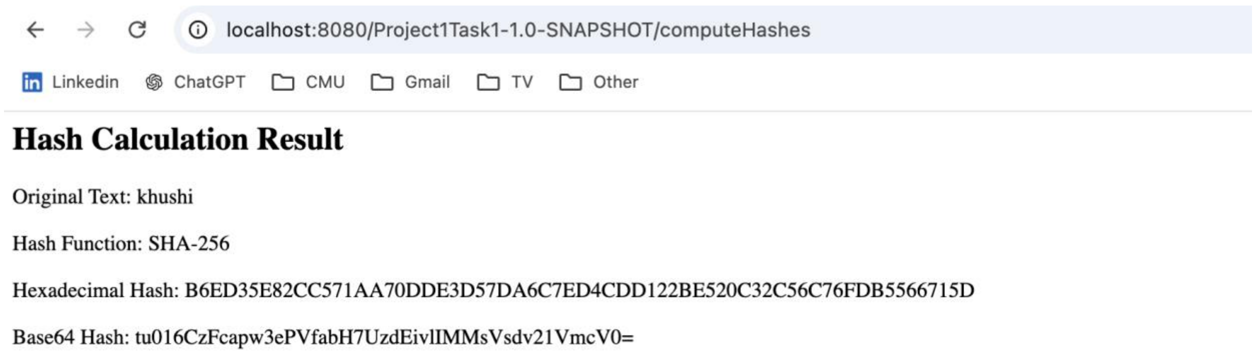
localhost:8080/Project1Task1-1.0-SNAPSHOT/

Enter a string of text data and make a choice of two hash functions

Text: khushi

MD5 [✓ SHA-256]

SHA-256 Output:



localhost:8080/Project1Task1-1.0-SNAPSHOT/computeHashes

Hash Calculation Result

Original Text: khushi

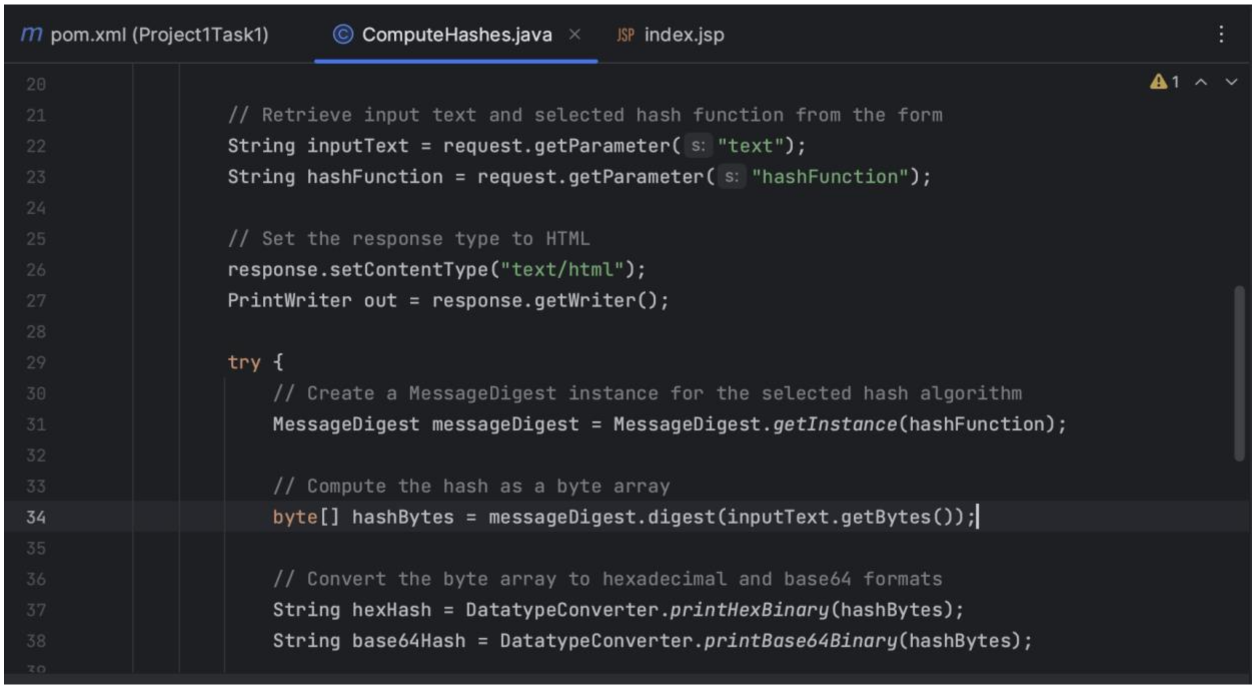
Hash Function: SHA-256

Hexadecimal Hash: B6ED35E82CC571AA70DDE3D57DA6C7ED4CDD122BE520C32C56C76FDB5566715D

Base64 Hash: tu016CzFcapw3ePVfabH7UzdEivlIMMsVsdv21VmcV0=

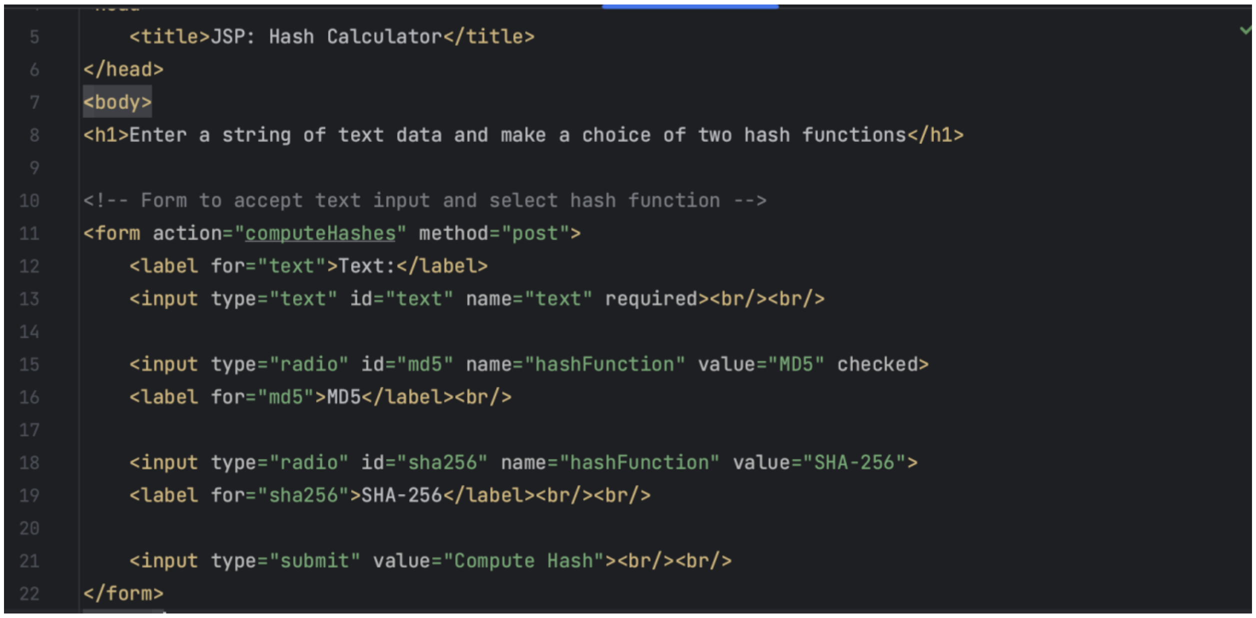
Khushi Bhuwania kbhuwani

Code Snippet:



```java
        // Retrieve input text and selected hash function from the form
        String inputText = request.getParameter( s: "text");
        String hashFunction = request.getParameter( s: "hashFunction");

        // Set the response type to HTML
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        try {
            // Create a MessageDigest instance for the selected hash algorithm
            MessageDigest messageDigest = MessageDigest.getInstance(hashFunction);

            // Compute the hash as a byte array
            byte[] hashBytes = messageDigest.digest(inputText.getBytes());

            // Convert the byte array to hexadecimal and base64 formats
            String hexHash = DatatypeConverter.printHexBinary(hashBytes);
            String base64Hash = DatatypeConverter.printBase64Binary(hashBytes);
```

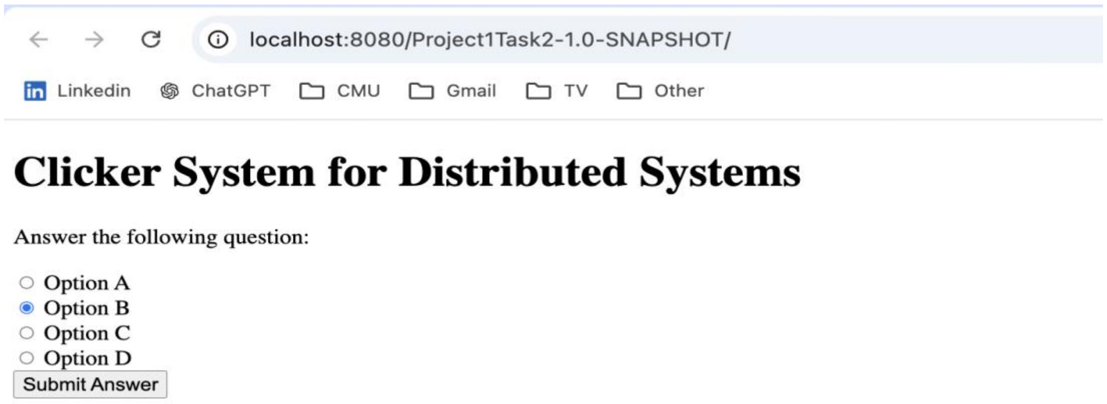
Index.jsp



```html
    <title>JSP: Hash Calculator</title>
</head>
<body>
<h1>Enter a string of text data and make a choice of two hash functions</h1>

<!-- Form to accept text input and select hash function -->
<form action="computeHashes" method="post">
    <label for="text">Text:</label>
    <input type="text" id="text" name="text" required><br/><br/>

    <input type="radio" id="md5" name="hashFunction" value="MD5" checked>
    <label for="md5">MD5</label><br/>

    <input type="radio" id="sha256" name="hashFunction" value="SHA-256">
    <label for="sha256">SHA-256</label><br/><br/>

    <input type="submit" value="Compute Hash"><br/><br/>
</form>
```

Task 2

Input page selection (When B is selected):

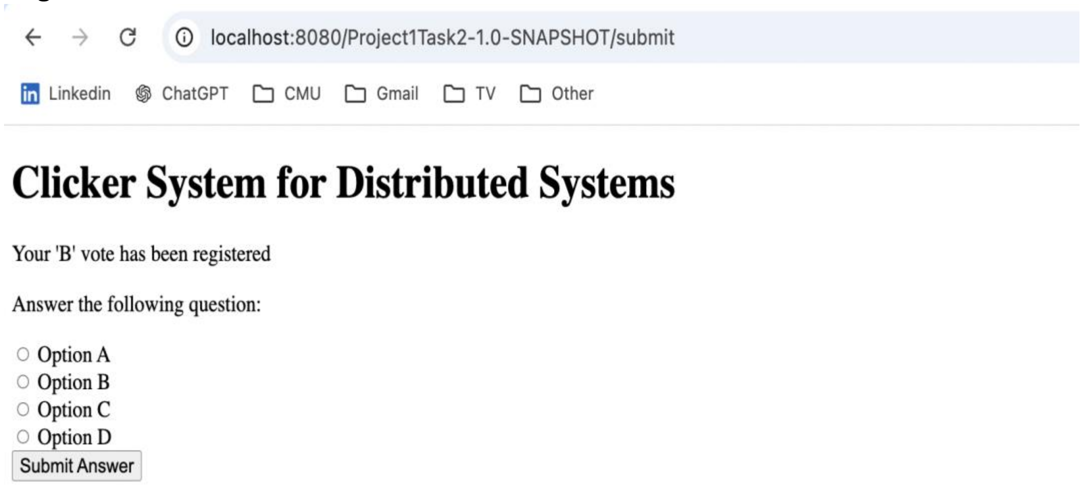

localhost:8080/Project1Task2-1.0-SNAPSHOT/

Clicker System for Distributed Systems

Answer the following question:

Option A
[✓ Option B]
Option C
Option D

Registered vote:


localhost:8080/Project1Task2-1.0-SNAPSHOT/submit

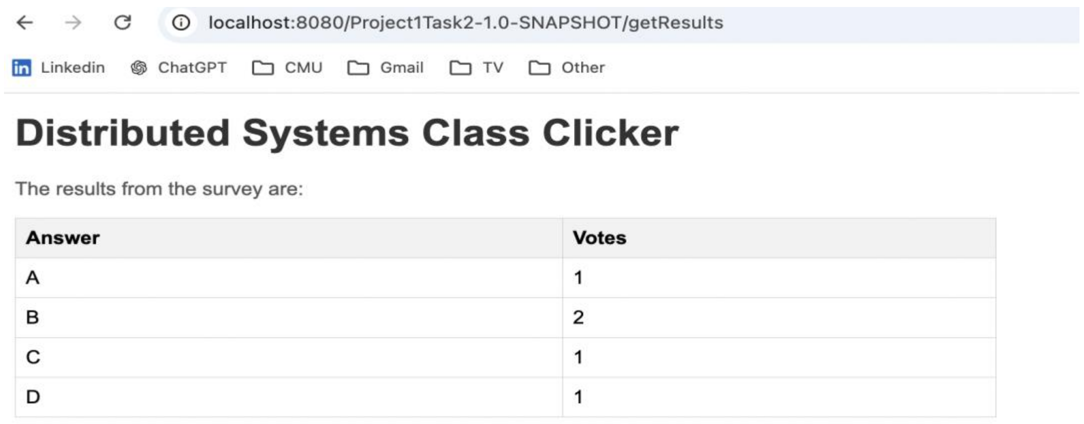
Clicker System for Distributed Systems

Your 'B' vote has been registered

Answer the following question:

Option A
Option B
Option C
Option D

Results (after few selections):

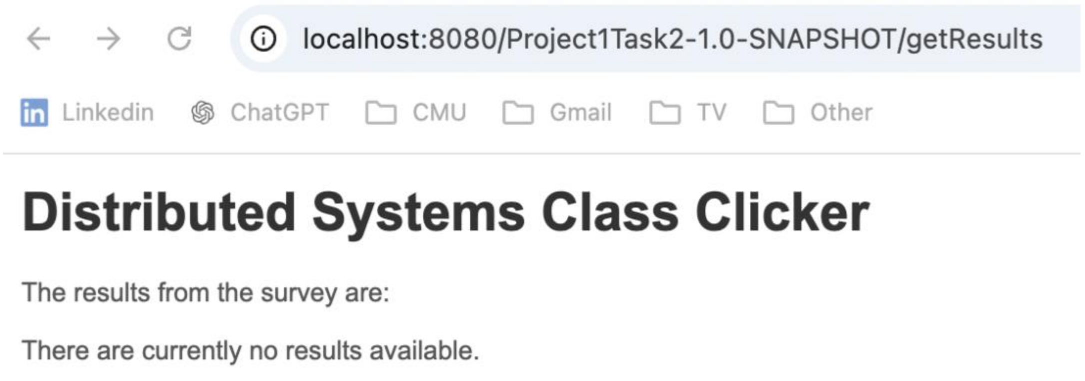

localhost:8080/Project1Task2-1.0-SNAPSHOT/getResults

Distributed Systems Class Clicker

The results from the survey are:

Answer	Votes
A	1
B	2
C	1
D	1

After refreshing, the results disappear

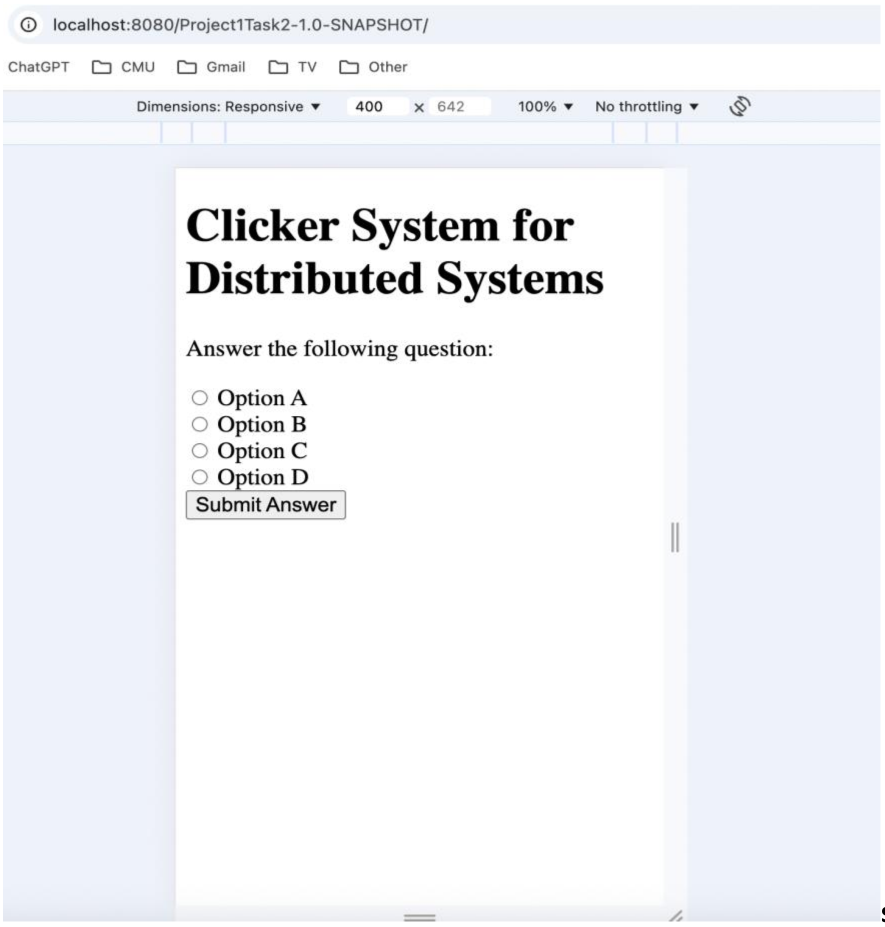


Distributed Systems Class Clicker

The results from the survey are:

There are currently no results available.

Mobile screenshots:

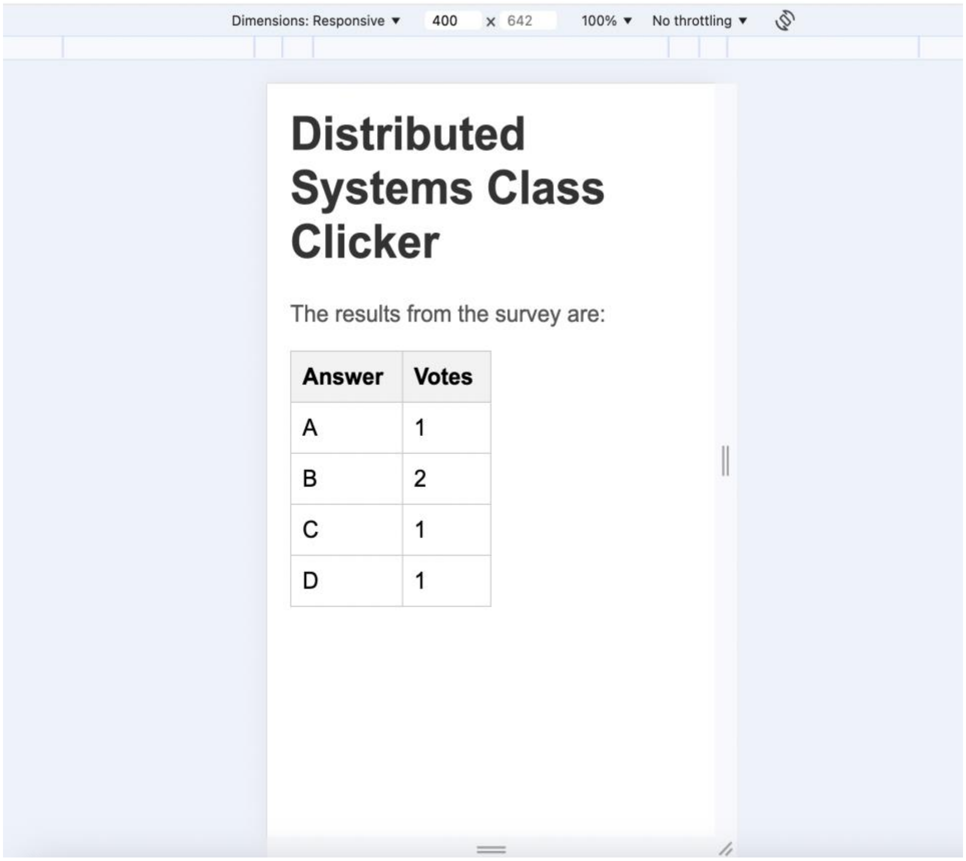


Clicker System for Distributed Systems

Answer the following question:

- ○ Option A
- ○ Option B
- ○ Option C
- ○ Option D

s:

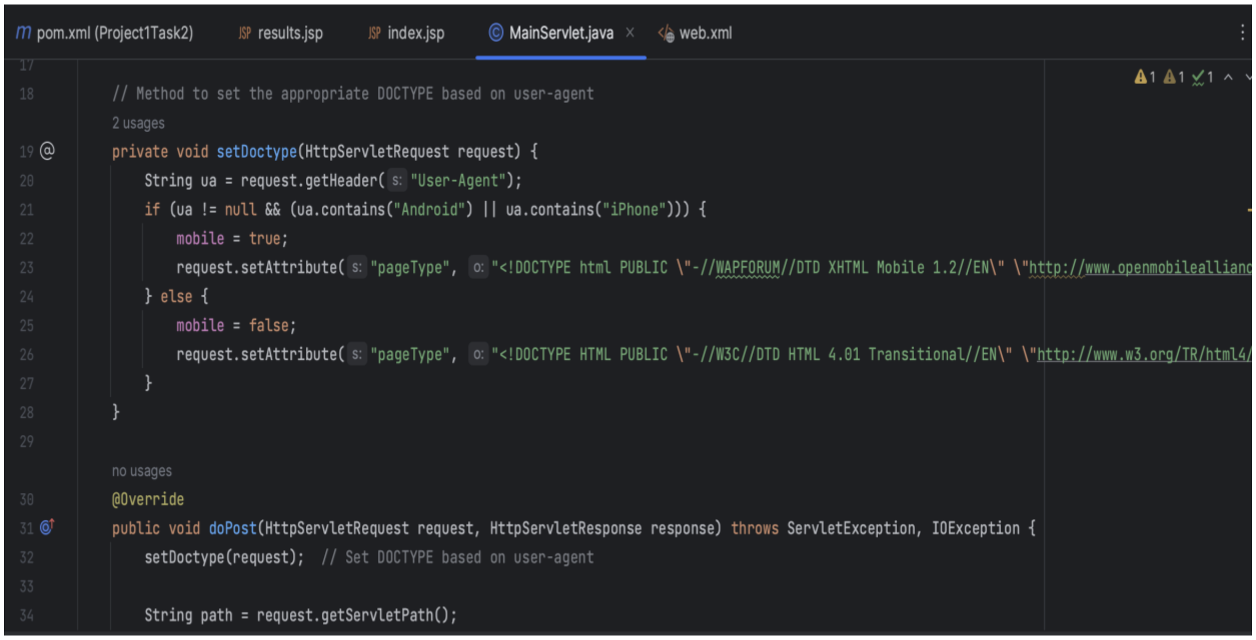


Distributed Systems Class Clicker

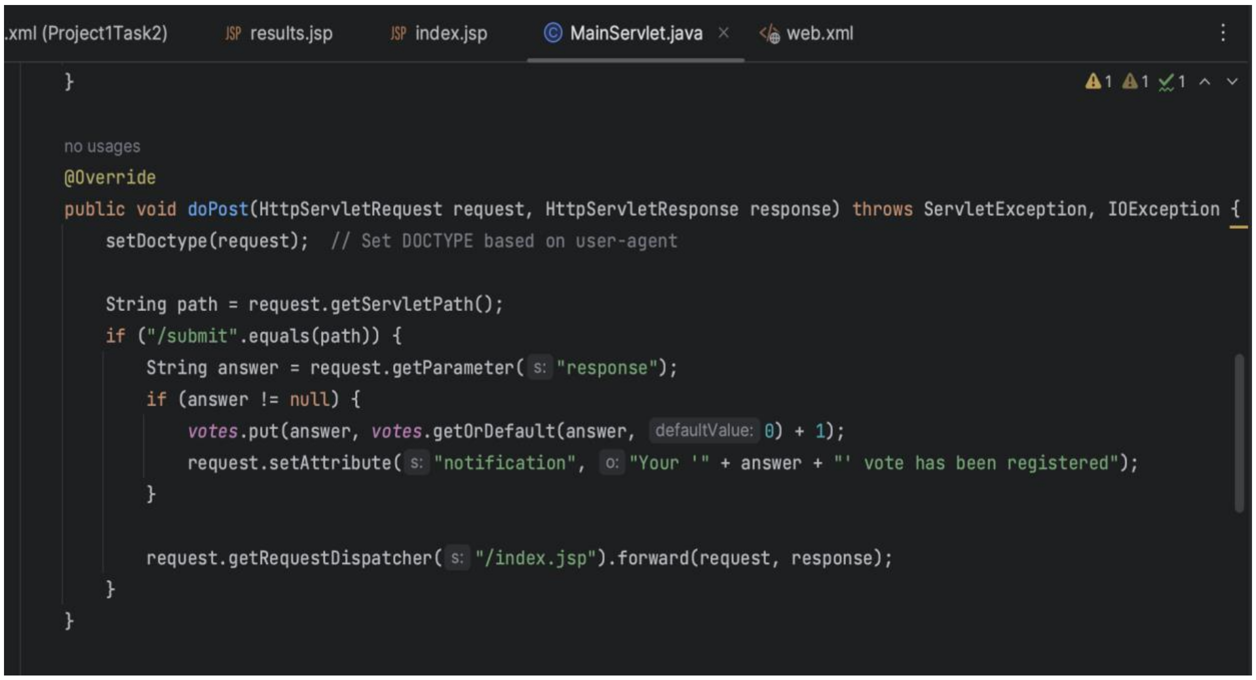
The results from the survey are:

Answer	Votes
A	1
B	2
C	1
D	1

Code snippets:

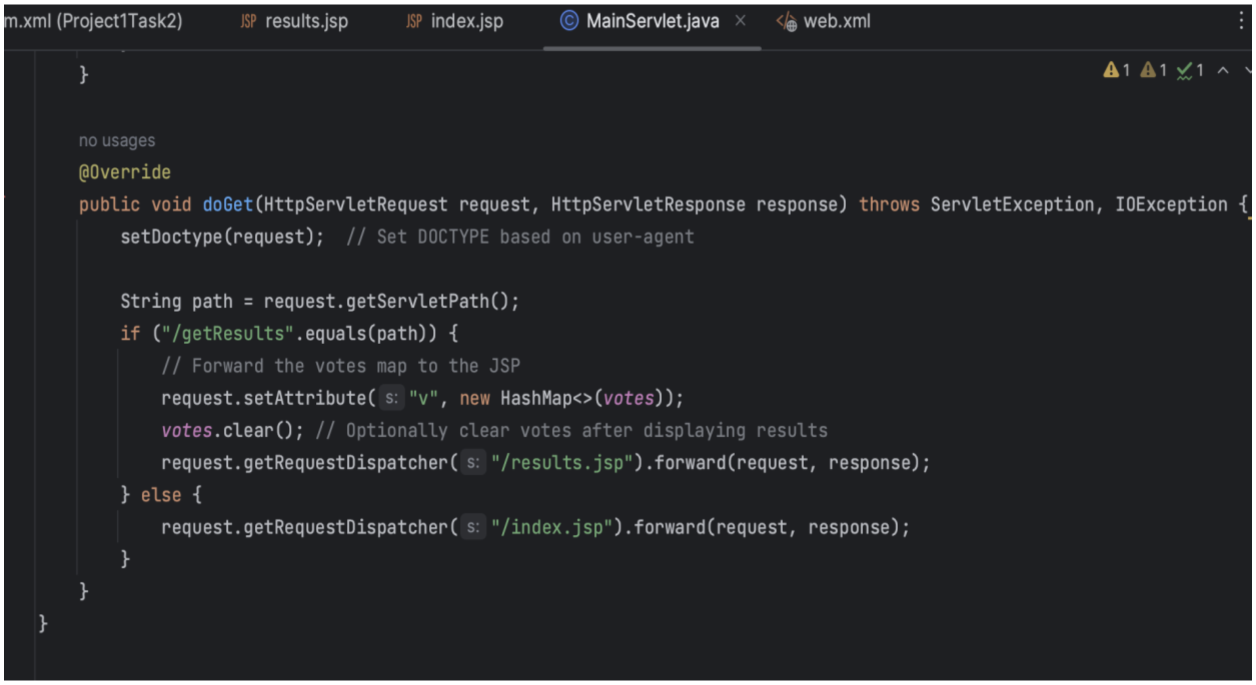


```java
    // Method to set the appropriate DOCTYPE based on user-agent
    2 usages
    private void setDoctype(HttpServletRequest request) {
        String ua = request.getHeader( s: "User-Agent");
        if (ua != null && (ua.contains("Android") || ua.contains("iPhone"))) {
            mobile = true;
            request.setAttribute( s: "pageType",  o: "<!DOCTYPE html PUBLIC \"-//WAPFORUM//DTD XHTML Mobile 1.2//EN\" \"http://www.openmobilealliance
        } else {
            mobile = false;
            request.setAttribute( s: "pageType",  o: "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\" \"http://www.w3.org/TR/html4/
        }
    }

    no usages
    @Override
    public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        setDoctype(request);  // Set DOCTYPE based on user-agent

        String path = request.getServletPath();
```



.xml (Project1Task2) | results.jsp | index.jsp | MainServlet.java | web.xml

```
    }

    no usages
    @Override
    public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        setDoctype(request);  // Set DOCTYPE based on user-agent

        String path = request.getServletPath();
        if ("/submit".equals(path)) {
            String answer = request.getParameter( s: "response");
            if (answer != null) {
                votes.put(answer, votes.getOrDefault(answer,  defaultValue: 0) + 1);
                request.setAttribute( s: "notification",  o: "Your '" + answer + "' vote has been registered");
            }

            request.getRequestDispatcher( s: "/index.jsp").forward(request, response);
        }
    }
```

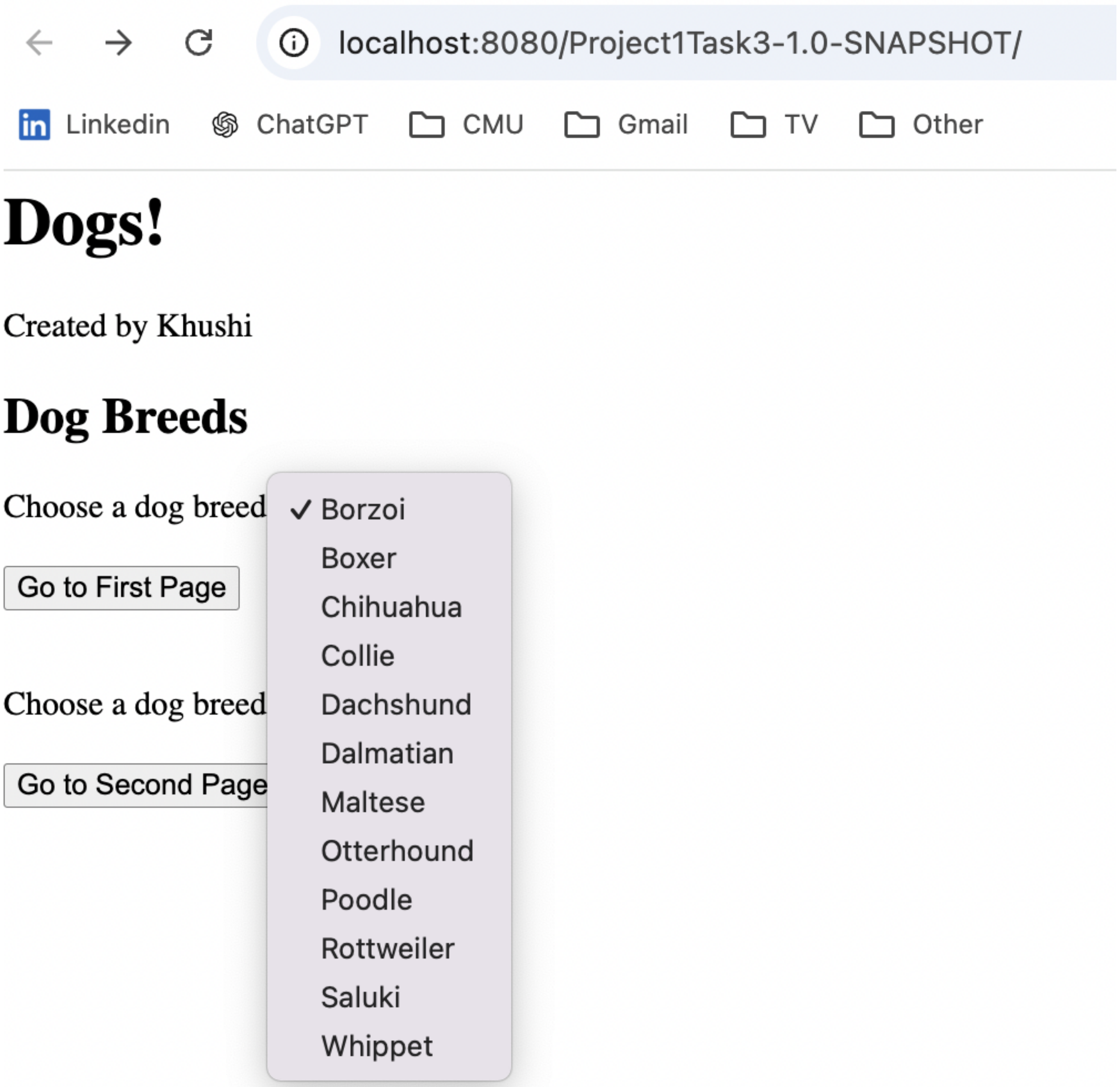


m.xml (Project1Task2) | results.jsp | index.jsp | MainServlet.java | web.xml

```
    }

    no usages
    @Override
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        setDoctype(request);  // Set DOCTYPE based on user-agent

        String path = request.getServletPath();
        if ("/getResults".equals(path)) {
            // Forward the votes map to the JSP
            request.setAttribute( s: "v", new HashMap<>(votes));
            votes.clear(); // Optionally clear votes after displaying results
            request.getRequestDispatcher( s: "/results.jsp").forward(request, response);
        } else {
            request.getRequestDispatcher( s: "/index.jsp").forward(request, response);
        }
    }
}
```

Task 3

Screenshots



Dogs!

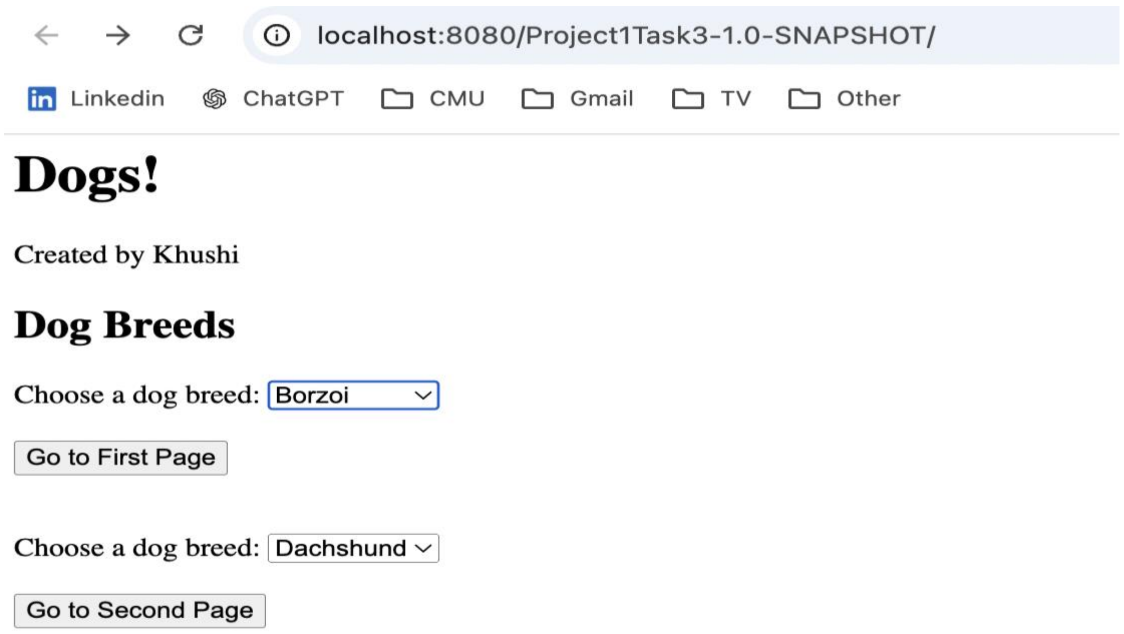
Created by Khushi

Dog Breeds

Choose a dog breed

Choose a dog breed

For First page:
Input 1 (selected Borzoi):



localhost:8080/Project1Task3-1.0-SNAPSHOT/

Dogs!

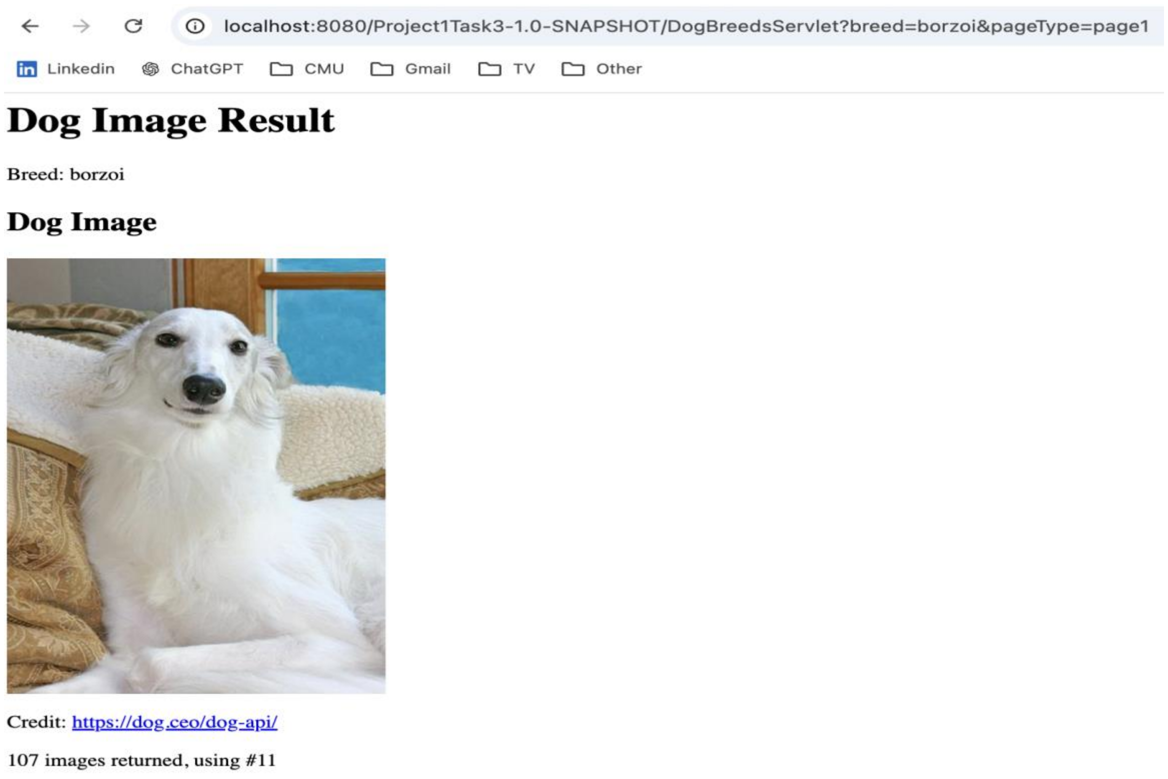
Created by Khushi

Dog Breeds

Choose a dog breed: Borzoi

Choose a dog breed: Dachshund

Output 1:



localhost:8080/Project1Task3-1.0-SNAPSHOT/DogBreedsServlet?breed=borzoi&pageType=page1

Dog Image Result

Breed: borzoi

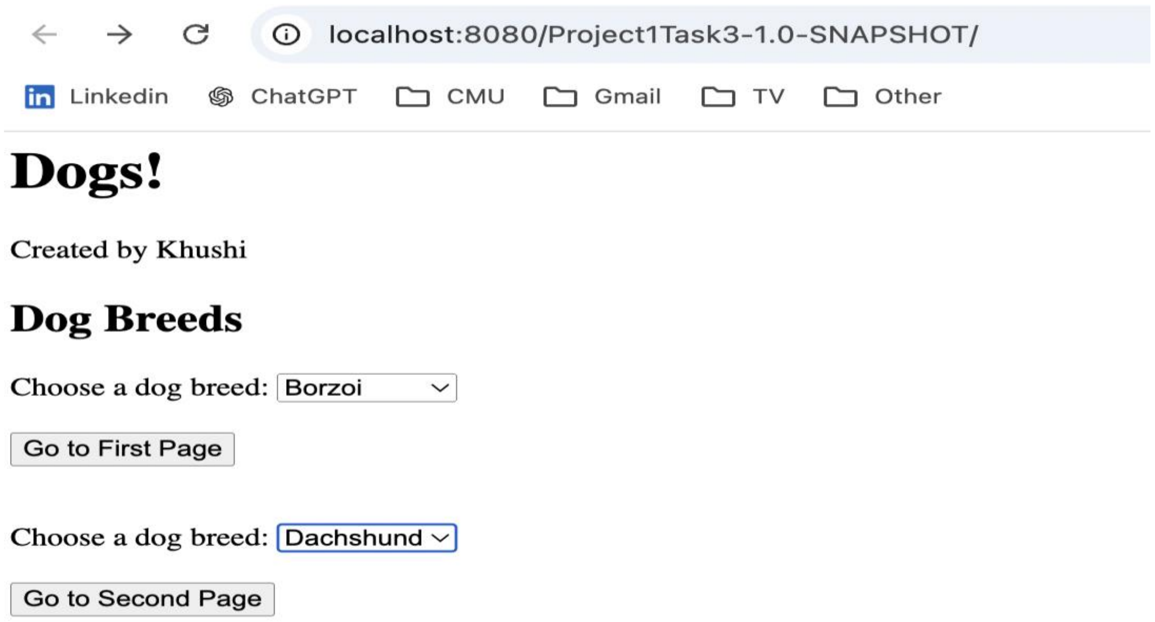
Dog Image

Credit: https://dog.ceo/dog-api/

107 images returned, using #11

For Second page:

Input 1 (selected Dachshund):



Dogs!

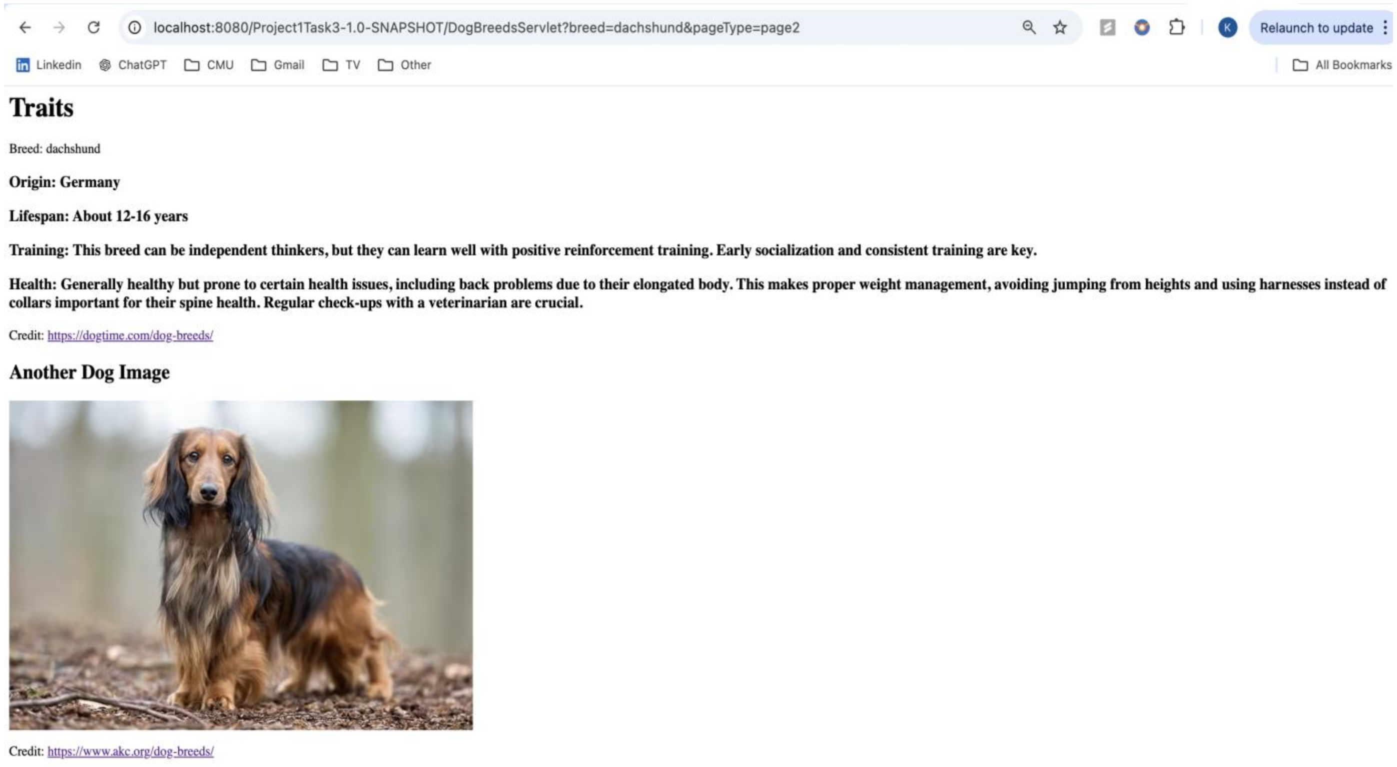
Created by Khushi

Dog Breeds

Choose a dog breed: Borzoi

Choose a dog breed: Dachshund

Output 1:


localhost:8080/Project1Task3-1.0-SNAPSHOT/DogBreedsServlet?breed=dachshund&pageType=page2
Relaunch to update
Linkedin
ChatGPT
CMU
Gmail
TV
Other
All Bookmarks

Traits

Breed: dachshund

Origin: Germany

Lifespan: About 12-16 years

Training: This breed can be independent thinkers, but they can learn well with positive reinforcement training. Early socialization and consistent training are key.

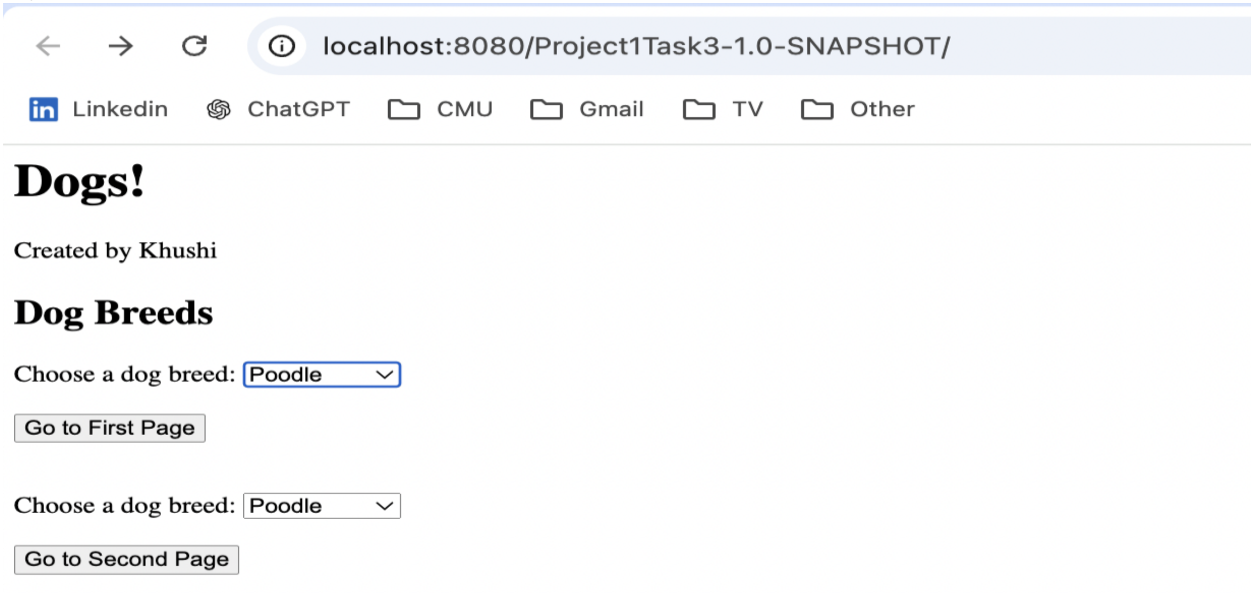
Health: Generally healthy but prone to certain health issues, including back problems due to their elongated body. This makes proper weight management, avoiding jumping from heights and using harnesses instead of collars important for their spine health. Regular check-ups with a veterinarian are crucial.

Credit: https://dogtime.com/dog-breeds/

Another Dog Image

Credit: https://www.akc.org/dog-breeds/

First page:
Input 2 (selected Poodle):



localhost:8080/Project1Task3-1.0-SNAPSHOT/

Linkedin ChatGPT CMU Gmail TV Other

Dogs!

Created by Khushi

Dog Breeds

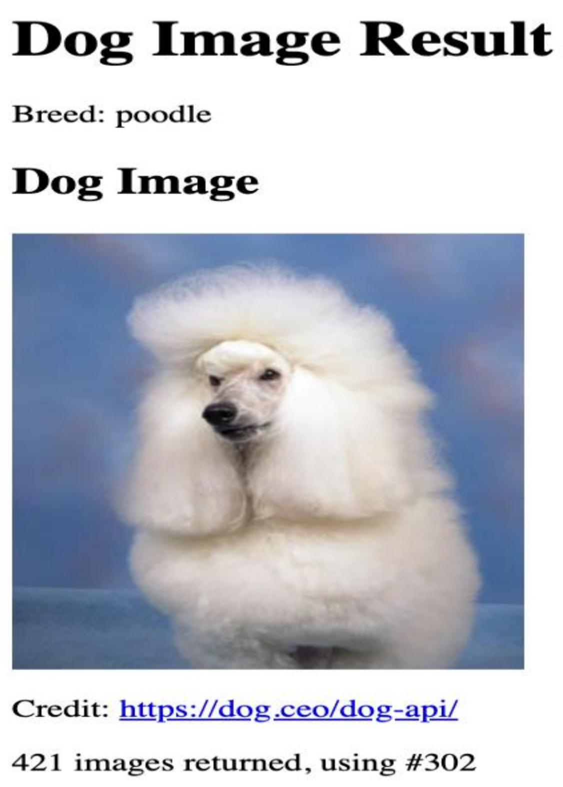
Choose a dog breed: Poodle

Go to First Page

Choose a dog breed: Poodle

Go to Second Page

Output 2:



Dog Image Result

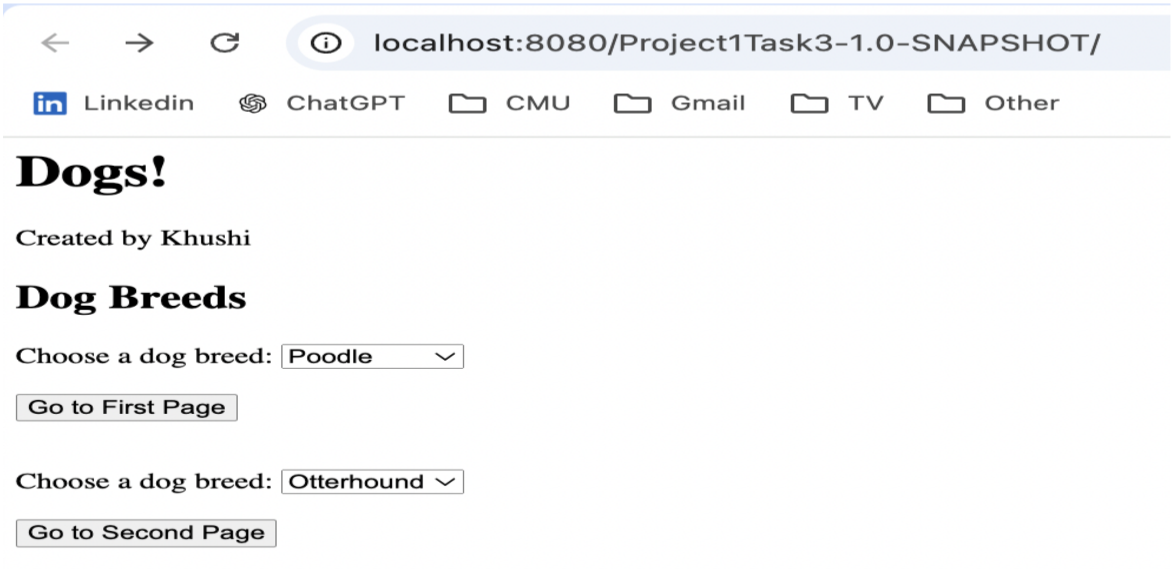
Breed: poodle

Dog Image

Credit: https://dog.ceo/dog-api/

421 images returned, using #302

Second page:
Input 2 (selected Otterhound):



Dogs!

Created by Khushi

Dog Breeds

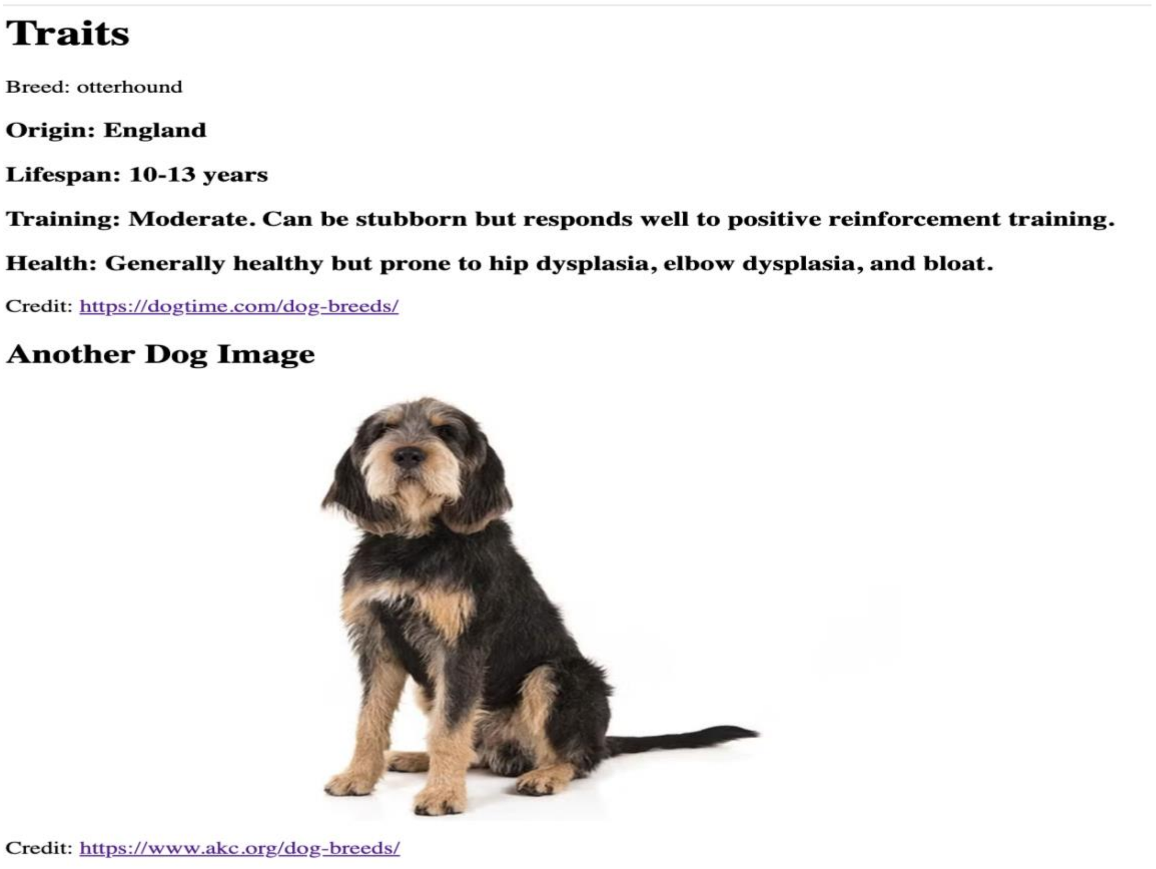
Choose a dog breed: Poodle

Go to First Page

Choose a dog breed: Otterhound

Go to Second Page

Output 2:



Traits

Breed: otterhound

Origin: England

Lifespan: 10-13 years

Training: Moderate. Can be stubborn but responds well to positive reinforcement training.

Health: Generally healthy but prone to hip dysplasia, elbow dysplasia, and bloat.

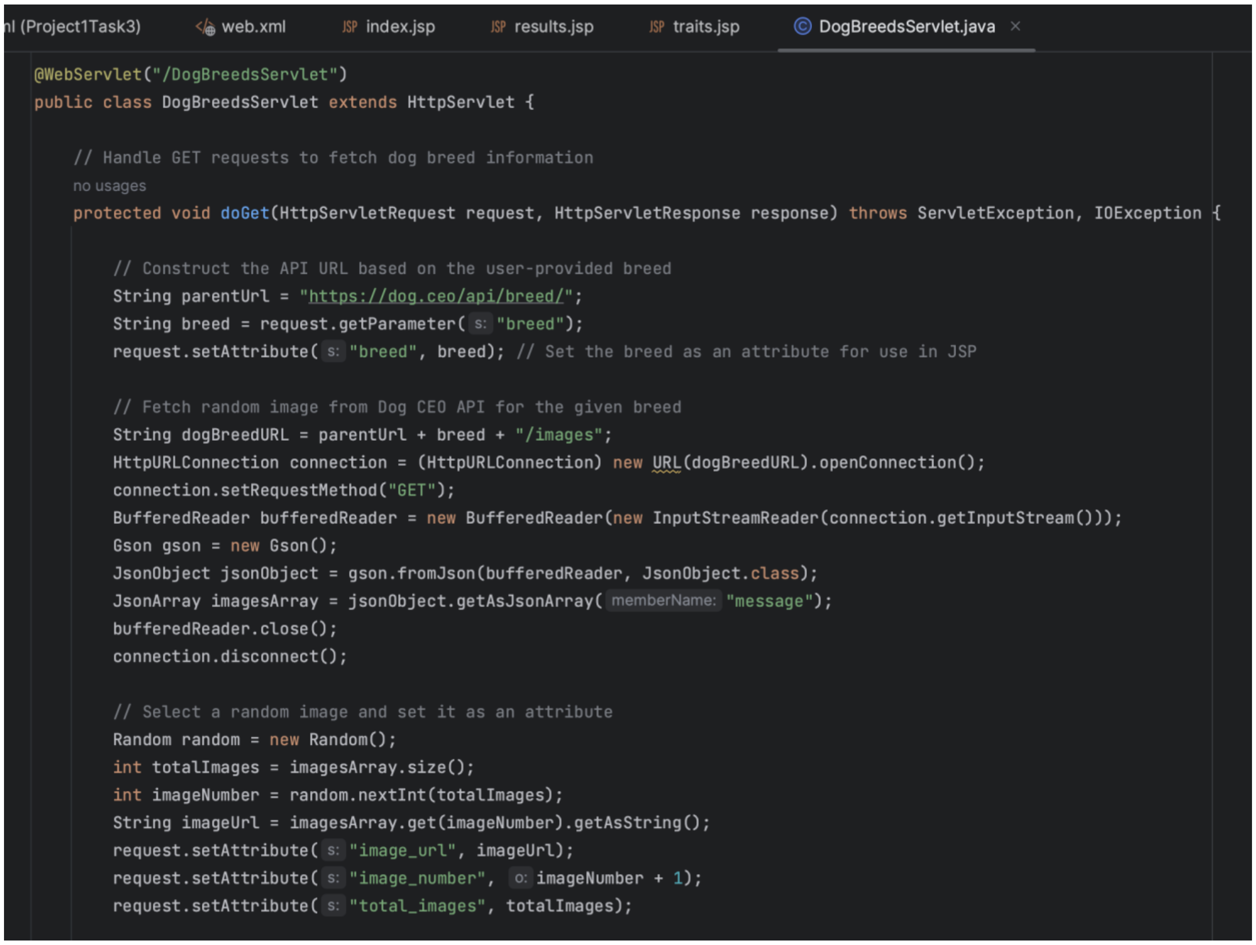
Credit: https://dogtime.com/dog-breeds/

Another Dog Image

Credit: https://www.akc.org/dog-breeds/

Code Snippets:

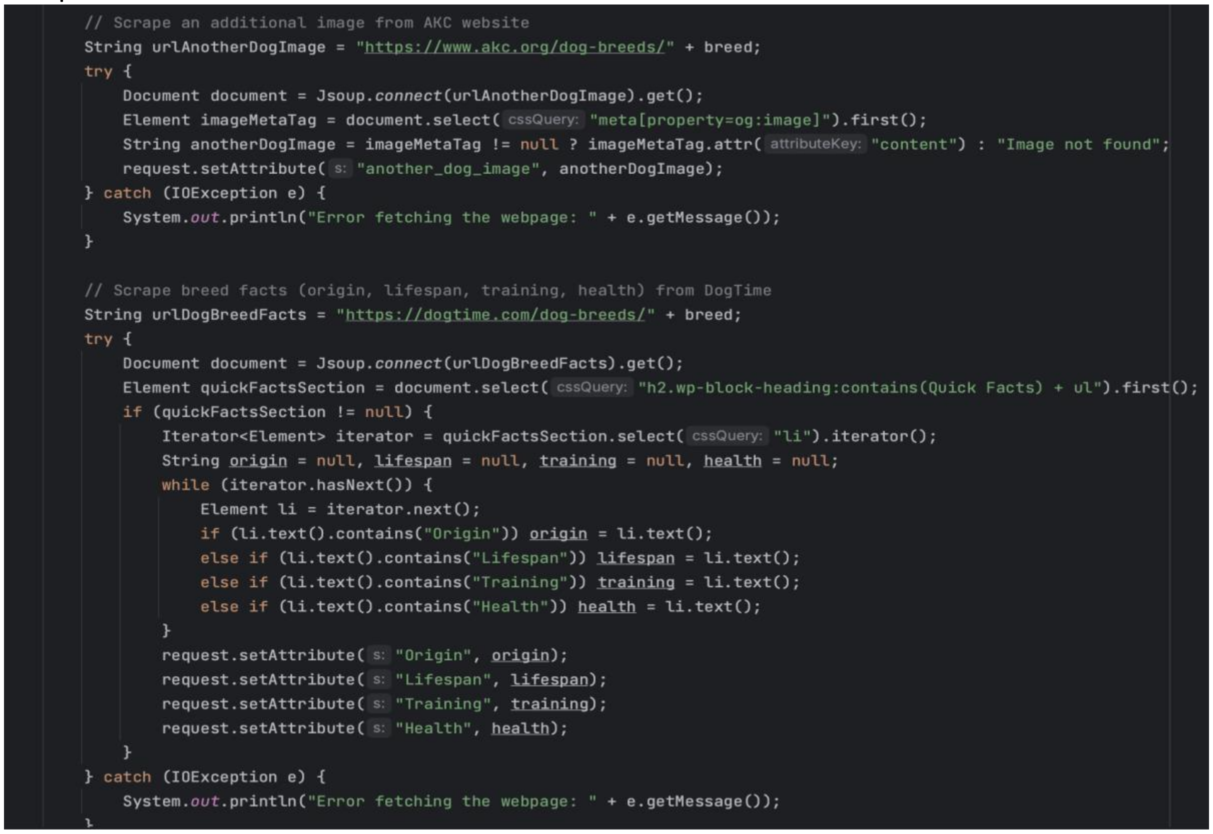
API Querying:



```java
@WebServlet("/DogBreedsServlet")
public class DogBreedsServlet extends HttpServlet {

    // Handle GET requests to fetch dog breed information
    no usages
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

        // Construct the API URL based on the user-provided breed
        String parentUrl = "https://dog.ceo/api/breed/";
        String breed = request.getParameter( s: "breed");
        request.setAttribute( s: "breed", breed); // Set the breed as an attribute for use in JSP

        // Fetch random image from Dog CEO API for the given breed
        String dogBreedURL = parentUrl + breed + "/images";
        HttpURLConnection connection = (HttpURLConnection) new URL(dogBreedURL).openConnection();
        connection.setRequestMethod("GET");
        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(connection.getInputStream()));
        Gson gson = new Gson();
        JsonObject jsonObject = gson.fromJson(bufferedReader, JsonObject.class);
        JsonArray imagesArray = jsonObject.getAsJsonArray( memberName: "message");
        bufferedReader.close();
        connection.disconnect();

        // Select a random image and set it as an attribute
        Random random = new Random();
        int totalImages = imagesArray.size();
        int imageNumber = random.nextInt(totalImages);
        String imageUrl = imagesArray.get(imageNumber).getAsString();
        request.setAttribute( s: "image_url", imageUrl);
        request.setAttribute( s: "image_number", o: imageNumber + 1);
        request.setAttribute( s: "total_images", totalImages);
```

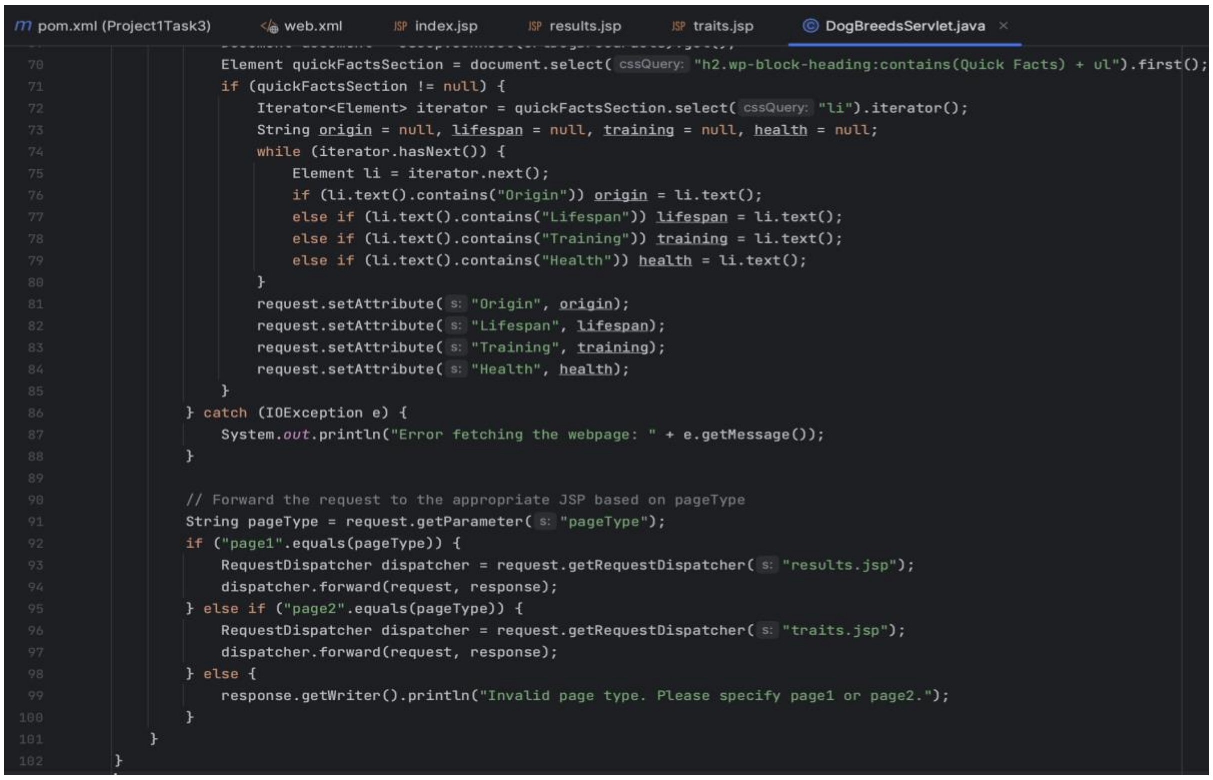
Khushi Bhuwania kbhuwani

Scraper:



```
// Scrape an additional image from AKC website
String urlAnotherDogImage = "https://www.akc.org/dog-breeds/" + breed;
try {
    Document document = Jsoup.connect(urlAnotherDogImage).get();
    Element imageMetaTag = document.select( cssQuery: "meta[property=og:image]").first();
    String anotherDogImage = imageMetaTag != null ? imageMetaTag.attr( attributeKey: "content") : "Image not found";
    request.setAttribute( s: "another_dog_image", anotherDogImage);
} catch (IOException e) {
    System.out.println("Error fetching the webpage: " + e.getMessage());
}

// Scrape breed facts (origin, lifespan, training, health) from DogTime
String urlDogBreedFacts = "https://dogtime.com/dog-breeds/" + breed;
try {
    Document document = Jsoup.connect(urlDogBreedFacts).get();
    Element quickFactsSection = document.select( cssQuery: "h2.wp-block-heading:contains(Quick Facts) + ul").first();
    if (quickFactsSection != null) {
        Iterator<Element> iterator = quickFactsSection.select( cssQuery: "li").iterator();
        String origin = null, lifespan = null, training = null, health = null;
        while (iterator.hasNext()) {
            Element li = iterator.next();
            if (li.text().contains("Origin")) origin = li.text();
            else if (li.text().contains("Lifespan")) lifespan = li.text();
            else if (li.text().contains("Training")) training = li.text();
            else if (li.text().contains("Health")) health = li.text();
        }
        request.setAttribute( s: "Origin", origin);
        request.setAttribute( s: "Lifespan", lifespan);
        request.setAttribute( s: "Training", training);
        request.setAttribute( s: "Health", health);
    }
} catch (IOException e) {
    System.out.println("Error fetching the webpage: " + e.getMessage());
```

Controller:



pom.xml (Project1Task3) web.xml index.jsp results.jsp traits.jsp DogBreedsServlet.java

```
            Element quickFactsSection = document.select( cssQuery: "h2.wp-block-heading:contains(Quick Facts) + ul").first();
            if (quickFactsSection != null) {
                Iterator<Element> iterator = quickFactsSection.select( cssQuery: "li").iterator();
                String origin = null, lifespan = null, training = null, health = null;
                while (iterator.hasNext()) {
                    Element li = iterator.next();
                    if (li.text().contains("Origin")) origin = li.text();
                    else if (li.text().contains("Lifespan")) lifespan = li.text();
                    else if (li.text().contains("Training")) training = li.text();
                    else if (li.text().contains("Health")) health = li.text();
                }
                request.setAttribute( s: "Origin", origin);
                request.setAttribute( s: "Lifespan", lifespan);
                request.setAttribute( s: "Training", training);
                request.setAttribute( s: "Health", health);
            }
        } catch (IOException e) {
            System.out.println("Error fetching the webpage: " + e.getMessage());
        }

        // Forward the request to the appropriate JSP based on pageType
        String pageType = request.getParameter( s: "pageType");
        if ("page1".equals(pageType)) {
            RequestDispatcher dispatcher = request.getRequestDispatcher( s: "results.jsp");
            dispatcher.forward(request, response);
        } else if ("page2".equals(pageType)) {
            RequestDispatcher dispatcher = request.getRequestDispatcher( s: "traits.jsp");
            dispatcher.forward(request, response);
        } else {
            response.getWriter().println("Invalid page type. Please specify page1 or page2.");
        }
    }
}
```

P.s. Used chatgpt for the reference for JSoup code in task3