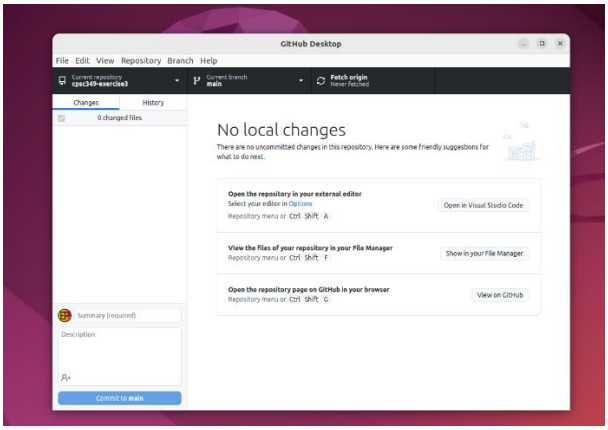
Edwin Peraza

Fall 2022, CPSC 349, Section 1

Exercise 3

Step 1:

I used the GitHub Desktop app to clone the repository via URL



After that I used the commands:

npm install

npm start

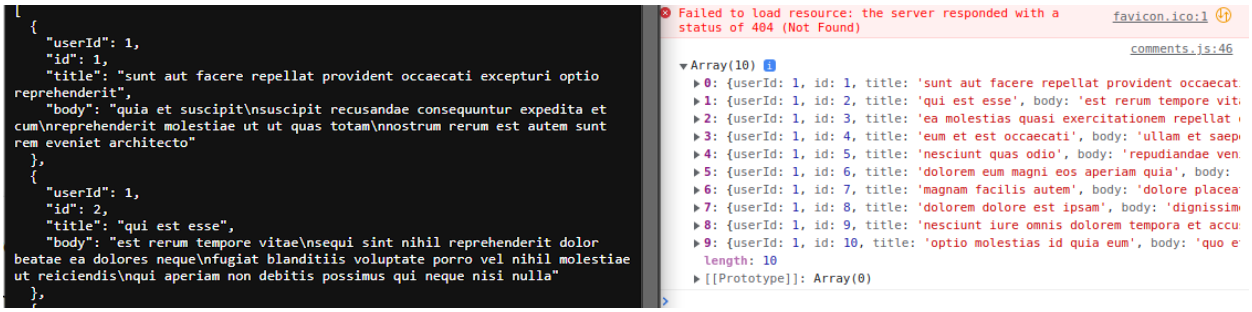
```
edwin@edwin-VirtualBox:~/Documents/GitHub/cpsc349-exercise3$ npm install

added 199 packages, and audited 200 packages in 5s

76 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
edwin@edwin-VirtualBox:~/Documents/GitHub/cpsc349-exercise3$ npm start

> cpsc349-exercise3@1.0.0 start
> browser-sync start --server --startPath /posts.html --files '*.html,*.css,*.js'
```

Step 2:

The three URLs have all the text parts of the html in post.html

Step 3:

The contents of the array are identical to the content that exist in the URL



Step 4:

For this step I created a new function called fillElements that received the JSON that was fetched from the URL in the function downladPosts. The function creates tags and fills them with the information in the JSON for each one of the array elements. The getUserName function is called to retrieve the author's name from a different JSON. I also created a function called newLine to replace the newline notation (\n) with
 tags for the content of the paragraph. I also added a toggle event listener for the details tag to whenever it is opened. The event listener creates an array with all the comments for the specified article id. The code that I used is the following:

```
function newLine(para) {
    para = para.replace(/(?:\n)/g, "<br>");
    return para;
}

// actual code
async function fillElements(obj) {
    const objects = obj;
    for (const objective of objects) {
        const myMain = document.querySelector("main");
        const art = document.createElement("article");
        art.setAttribute("data-post-id", objective.id);
        myMain.appendChild(art);
        const myH2 = document.createElement("h2");
        myH2.textContent = objective.title;
        art.appendChild(myH2);
        const myAside = document.createElement("aside");
        // add span for the author name
        const mySpan = document.createElement("span");
        mySpan.setAttribute("class", "author");
        mySpan.textContent = await getUserName(objective.userId);
        myAside.textContent = `by `;
        myAside.appendChild(mySpan);
        art.appendChild(myAside);
        // create paragraph
        const myPara = document.createElement("p");
        let myBody = objective.body;
        myPara.innerHTML = newLine(myBody);
        art.appendChild(myPara);
        // add detail section
        const det = document.createElement("details");
        myMain.appendChild(det);
        const mySum = document.createElement("summary");
        mySum.textContent = "See what our readers had to say...";
        det.appendChild(mySum);
        const mySec = document.createElement("section");
```

```
        det.appendChild(mySec);
        const myHeader = document.createElement("header");
        mySec.appendChild(myHeader);
        const myH3 = document.createElement("h3");
        myH3.textContent = "Comments";
        myHeader.appendChild(myH3);

        // add event listener
        det.addEventListener("toggle", async (event) => {
            if (det.open) {
                const asides = det.getElementsByTagName("aside");
                const commentsWereDownloaded = asides.length > 0;
                if (!commentsWereDownloaded) {
                    const articleId = getArticleId(det);
                    const comments = await downloadComments(articleId);
                    console.log(comments);
                }
            }
        });
    }
}
```

Step 5:

For this step, I basically removed all the HTML within the main tags and the remaining code is the following:

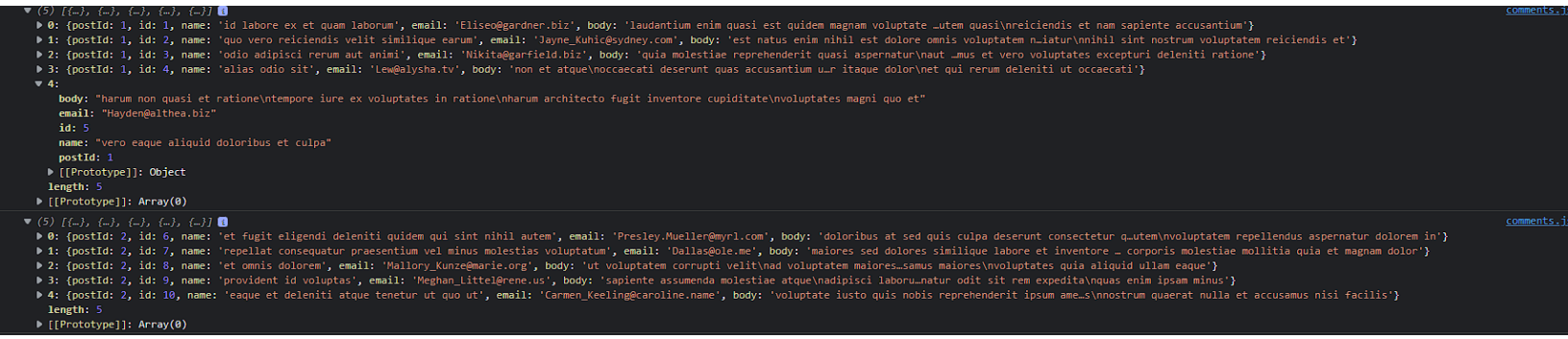
```
You, 1 second ago | 2 authors (You and others)
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <title>Lorem ipsum dolor sit amet</title>
        <link rel="stylesheet" href="https://unpkg.com/mvp.css@1.12/mvp.css" />
        <link rel="stylesheet" href="titles.css" />
    </head>

    <body>
        <header>
            <h1>Lorem Ipsum News</h1>
            <p>“All the news that’s fit to <em>consectetur adipiscing elit</em>”</p>
        </header>

        <main></main>

        <script type="module" src="comments.js"></script>
    </body>
</html>

```

The only difference in the output is that I previously had an extra article and details from the html that was included in the file, but the current output is as follows:

The 10 articles are being created dynamically, and when I toggle the details section an h3 with the word "Comments" appears.

Step 6:

Since I added an event listener on toggle, when I click on "See what our readers had to say…" the comments for the corresponding id are stored in an array. The same way as the array of articles.




```
[
  {
    "postId": 1,
    "id": 1,
    "name": "id labore ex et quam laborum",
    "email": "Eliseo@gardner.biz",
    "body": "laudantium enim quasi est quidem magnam voluptate ipsam eos\ntempora quo necessitatibus\ndolor quam autem quasi\nreiciendis et nam sapiente accusantium"
  },
  {
    "postId": 1,
    "id": 2,
    "name": "quo vero reiciendis velit similique earum",
    "email": "Jayne_Kuhic@sydney.com",
    "body": "est natus enim nihil est dolore omnis voluptatem numquam\net omnis occaecati quod ullam at\nvoluptatem error expedita pariatur\nnihil sint nostrum voluptatem reiciendis et"
  },
  {
```

Step 7:

For this step, I struggled for a while. I basically used the same approach as for the previous part. Use a loop to iterate through the array of comments, create an aside for each comment and fill the paragraphs with the content of the JSON. However, I made a typo in my loop and my browser was not detecting the issue. I thought that my problem was somewhere inside the loop. To be honest, I still do not understand much what was going on since the elements were being created, but the textContent was not being updated. I inspected the resulting HTML, and all the asides were created at the moment of toggling the section without content. The loop was working in some ways, but not everything inside the loop was being perform.

I had “for (const comment in comments)” instead of “for (const comment of comments)”

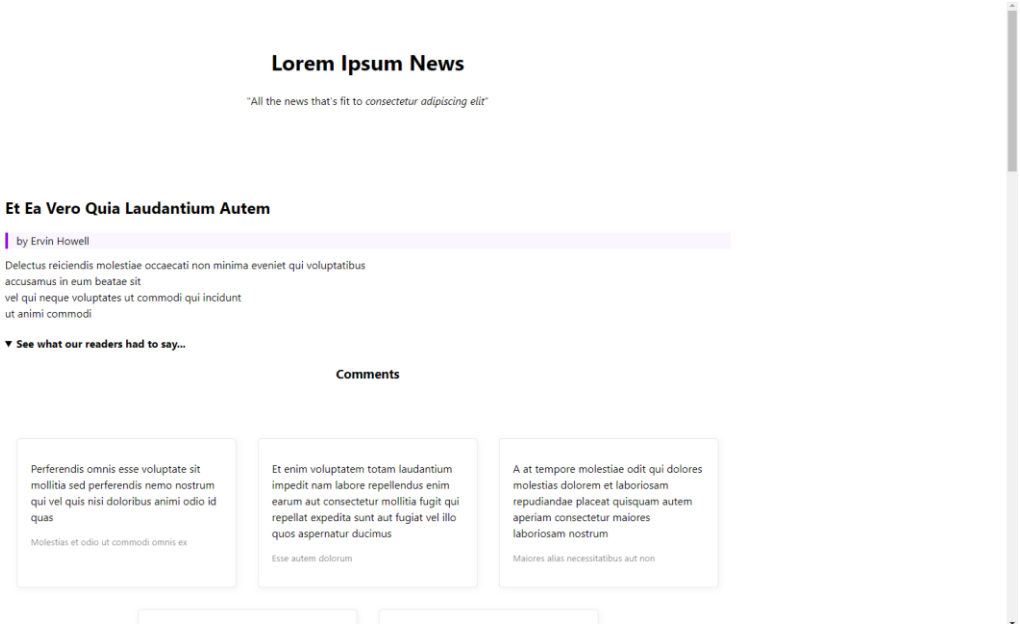
Once I found the error, everything worked as intended. I already had a separate function to replace the “\n” with “
”, so I just called it.

The code that I used is the following:

```
// add event listener
        det.addEventListener("toggle", async (event) => {
            if (det.open) {
                const asides = det.getElementsByTagName("aside");
                const commentsWereDownloaded = asides.length > 0;
                if (!commentsWereDownloaded) {
                    const articleId = getArticleId(det);
                    const comments = await downloadComments(articleId);
                    console.log(comments);
                    // add code to create asides
                    for (const comment of comments) {
                        const comAside = document.createElement("aside");
                        mySec.appendChild(comAside);
                        const description = document.createElement("p");
                        console.log(comment.body);
                        description.textContent = comment.body;
                        comAside.appendChild(description);
                        const second = document.createElement("p");
                        const s = document.createElement("small");
                        s.textContent = comment.name;
                        second.appendChild(s);
                        comAside.appendChild(second);
                    }
                }
            }
        });
```

Step 8:

The content of the page are still being created dynamically when I change downloadPost() to downloadPost(2) with the contents of the second page.



Everything is being populated as intended. I had no issues with this step.