AeroAspire-SDE Intern

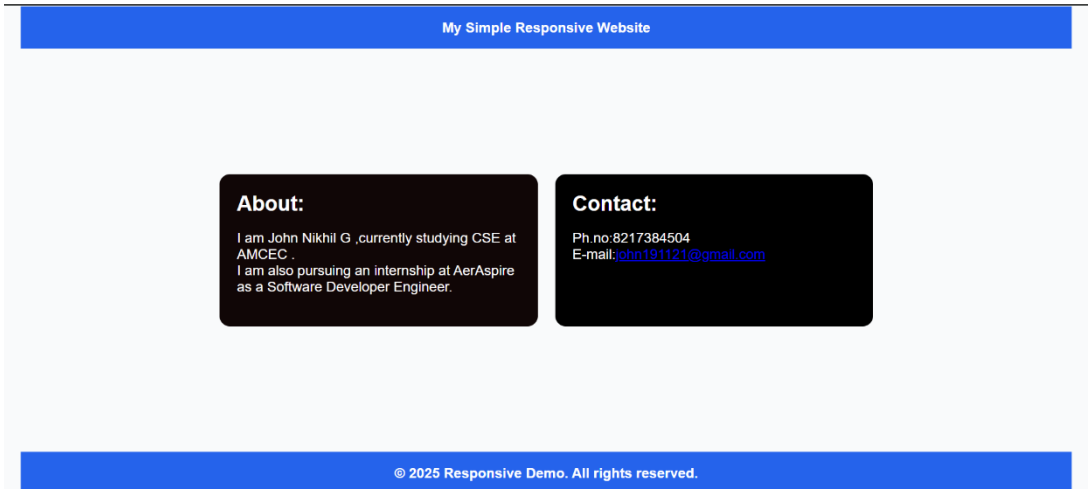
John Nikhil G

Week 1 – Day 4 (September 26)

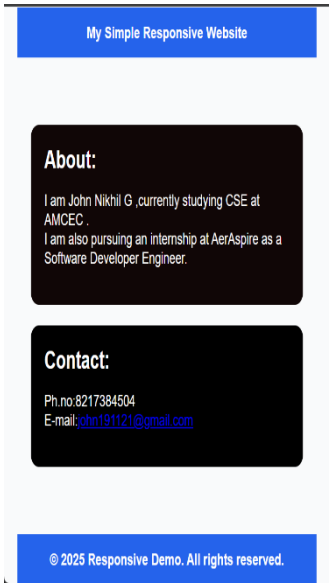
Task/Assignment :

Polish responsive layout;
deploy on GitHub Pages;
write README.

Desktop Layout :

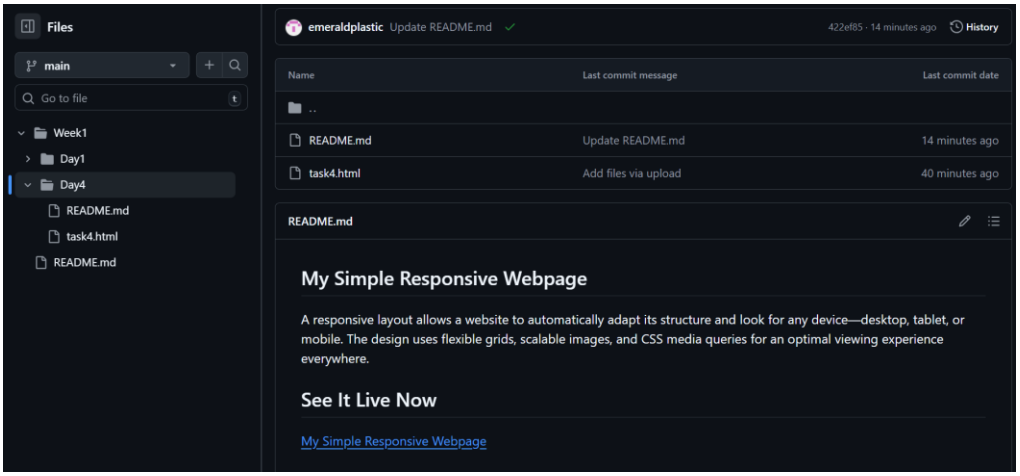


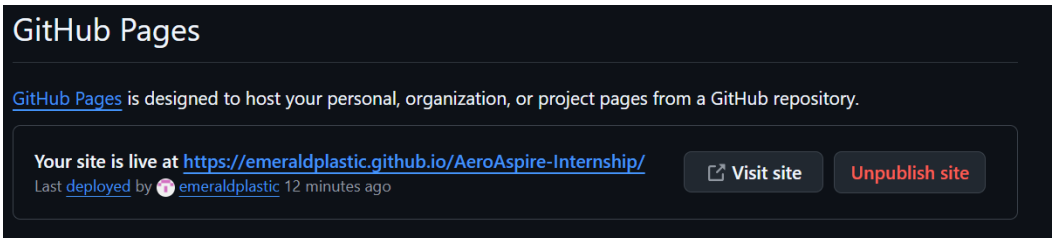
Mobile Layout :



Steps :

- **HTML Structure:-**
  - Created a simple webpage with **header, main content, and footer**.
  - Added **two content boxes**: *About* and *Contact*.
  - Used semantic tags like <header>, <main>, <footer>, and s for the boxes.
  - Each box has a **title (<h2>)** and placeholder for content (<p>), making it easy to add your own text later.
- **CSS Styling:-**
  - Styled the header, footer, and content boxes with **colors, padding, border-radius** to make it visually appealing.
  - Used **flexbox** and **grid** to align and organize content neatly.
  - Added **font styling** and **background colors** for better readability and aesthetics.
- **Media Queries / Responsiveness:-**
  - Added a **media query for screens ≤ 600px**.
  - The two boxes **stack vertically** on smaller screens to ensure a responsive layout.
  - Ensured the header, footer, and boxes resize and adjust properly for different screen widths.
- **GitHub Implementation:-**
  - Pushed the entire project to **GitHub** for version control and online access.
  - Created a **README** file explaining the project, its purpose, and how to use it.
  - The README also highlights the **responsive design and tech used** (HTML + CSS + media queries).





Questions/Reflections:

1. Steps I followed to deploy via GitHub Pages:

1. Created a GitHub Repository:

- Created a folder under week1/day4 called on GitHub.
- Added a professional description outlining my daily tasks and project objectives.
- Uploaded Project Files to GitHub:
- Initialized Git in my local project folder using git init.
- Staged and committed files with git add . and git commit -m "Initial commit".
- Pushed the code to GitHub with git push origin main.

2. Enabled GitHub Pages:

- Opened the repository settings on GitHub.
- Navigated to the *Pages* section under *Code and Automation*.
- Chose the main branch as the source and selected the root folder (/).
- Saved the configuration.

3. Deployed the Website:

- GitHub automatically built and published the project.
- The live website URL was generated, usually in the format: https://<username>.github.io/<repository-name>/.

4. Verified Deployment:

- Opened the live link on desktop, tablet, and mobile devices.
- Tested responsiveness, layout, and interactive features like the hamburger menu and contact form.
- Ensured that future updates reflected correctly when pushing new commits**.**

---

2. Challenges faced during deployment:

- Figuring out the GitHub Pages settings was tricky at first; the live site didn’t appear until I correctly selected the branch in the Pages configuration.
- These experiences helped me better understand branch conflicts, file paths, and deployment settings, making future deployments smoother.

---

3. How responsive breakpoints work (CSS media queries):

- CSS media queries (@media) allow you to apply styles based on device characteristics like width, height, or screen resolution.
- Common breakpoints include widths like 600px (mobile), 768px (tablet), 992px (desktop), and 1200px+ (large screens).
- Example:

```
@media only screen and (max-width: 600px) {
  body {
    background-color: lightblue;
  }
}
```

---

4. Importance of hover/focus/active states for UX:

- Hover (:hover) gives visual feedback when a user hovers over an element, improving discoverability and interaction.
- Focus (:focus) helps users, especially those navigating with a keyboard, see which element is currently active, improving accessibility.
- Active (:active) indicates when a user is interacting with an element, like pressing a button or link.
- These states enhance usability, guide user behavior, and ensure the interface is intuitive and accessible.