DAY 10

Note: The names for my HTML, CSS, and JavaScript files (like DAYTHREE.html, DAYTHREE.css, DAYTHREE.js) are **not recommended** as they are considered **bad practice** in professional projects. However, since this is a **mini project**, I prefer using these names to **distinguish them from my other files**. For **larger or professional projects**, it's better to use standard and descriptive file names like index.html, style.css, and script.js.

DAYTEN.HTML

<head> – Page Settings

The <head> section contains the settings and links for your page:

- `<link rel="stylesheet" href="DAYTEN.CSS" />`
  Connects your **CSS file** (DAYTEN.CSS) for page styling.
- `<title> PROFILE SEARCHER </title>`
  Sets the title shown in the browser tab.
- `<link rel="icon" href="...">`
  Sets the **favicon** (the small icon shown in the tab).
- `<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>`
  Loads the **Axios library** (used for making API calls in JavaScript).

<body> – What's visible on the page

Inside <body>, you have the visible parts of the page:

1. Search Form

```
<form class="inputForm" id="userInput">
  <H1 id="heading">DAY 10</H1>
  <h1>PROFILE SEARCHER</h1>
  <input type="text" id="inputBox" autocomplete="off" placeholder="Search a Github User" />
</form>
```

- A form where the user can **type a GitHub username**.
- It has two headings (DAY 10 and PROFILE SEARCHER).
- One text input box for searching a user.

2. Main Section

```
<main id="main"></main>
```

- An empty section where the **searched user's profile info will appear** (filled using JavaScript).

DAYTEN.CSS

1. Google Fonts

```
@import url("https://fonts.googleapis.com/css2?family=Pacifico&display=swap");
```

This line imports a cool font called **"Pacifico"** from Google Fonts — you can use it in your text if needed.

2. Reset & Base Styles

```
* {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
}
```

This removes default spacing from all HTML elements, so your layout is consistent everywhere.

3. Body Styling

```
body {
  font-family: 'Segoe UI', ...;
  height: 100vh;
  background: radial-gradient(...);
  animation: gradientBG 15s ease infinite;
  color: #fff;
  padding: 20px;
}
```

- Sets the default font.
- Adds a **colorful animated gradient background** that slowly moves.
- Sets full height of the screen.
- Makes all text white (`color: #fff`).

Background Animation

```
@keyframes gradientBG {
  0% { background-position: 0% 50%; }
  50% { background-position: 100% 50%; }
  100% { background-position: 0% 50%; }
}
```

This animation smoothly moves the background colors left and right to create a **flowing effect**.

4. Form Styling

```
form {
  margin: auto;
  margin-top: 60px;
  max-width: 500px;
  ...
}
```

- Centers the form.
- Gives it a transparent black background with rounded corners and shadow.
- Adds padding to make it comfortable.

Heading Text

```
#heading {
  text-decoration: underline;
  color: #fff;
  font-size: 2.2rem;
}
h1 {
  font-size: 2rem;
}
```

- Makes the heading big, white, and underlined.

Input Box

```
.inputForm input {
  width: 80%;
  padding: 12px;
  ...
}
```

- A clean, white text box with shadows.
- Rounded corners.
- Focus effect: when clicked, it glows white.

5. GitHub User Card

This styles the user profile card after you search a GitHub user.

Main Container

```
.card {
  background-color: rgb(245, 210, 231); /* light pink */
  display: flex;
  border-radius: 20px;
  ...
}
```

- A large card with light pink background and shadow.
- Shows user's profile picture and info side by side.

Profile Picture

```
.avatar {
  border-radius: 50%; /* makes it a circle */
  border: 4px solid green;
  height: 150px;
  transition: transform 0.3s;
}
.avatar:hover {
  transform: scale(1.1); /* zooms when hovered */
}
```

Info Section

```
.user-info {
  padding: 1rem;
  background-color: rgba(255, 255, 255, 0.9);
  border-radius: 10px;
  ...
}
.user-info h2 {
  color: darkred;
}
```

- White background inside the card.
- Displays name, bio, and list of followers, repos, etc.

User Stats

```
.user-info ul {
  list-style: none;
  display: flex;
  justify-content: space-between;
}
.user-info ul li strong {
  color: darkred;
}
```

- A clean, horizontal list of stats (followers, following, etc.)

6. Repository Tags

```
.repo {
  background-color: aquamarine;
  color: black;
  ...
}
.repo:hover {
  background-color: turquoise;
  transform: scale(1.05);
}
```

- These are buttons/tags that show user's repositories.

- They change color and grow slightly when hovered.

7. Responsive Design

```
@media (max-width: 768px) {
  .card {
    flex-direction: column;
    ...
  }
  .inputForm input {
    width: 100%;
  }
}
```

- Makes layout **mobile-friendly** by stacking elements vertically.
- Adjusts input width for small screens.

DAYTEN.JS

const apiURL = "https://api.github.com/users/";

- This is the **GitHub API link** that gives us user information.
- We'll add the username to the end of this link to get data about that user.

Getting HTML Elements

```
const main = document.getElementById("main");
const form = document.getElementById("userInput");
const input = document.getElementById("inputBox");
```

- These lines connect your JavaScript to parts of your HTML:
  - `main`: Where the user card will show up.
  - `form`: The form where you enter the username.
  - `input`: The input box where you type the username.

1. Show Loading Message

```
function showLoading() {
   main.innerHTML = "<h2>Loading...</h2>";
}
```

- When someone searches, it shows "Loading..." to let the user know it's working.

2. Show User Profile Card

```
function showUserCard(user) {
```

- Takes GitHub user data and shows it as a **beautiful profile card**.

```
  const userName = user.name || user.login;
```

- If user has a real name, use it. Otherwise use their GitHub username.

```
  const bio = user.bio ? `<p>${user.bio}</p>` : "";
```

- If the user has a bio, show it. If not, skip it.

```
  const cardHTML = `...`;
  main.innerHTML = cardHTML;
```

- Creates a styled card (with avatar, name, followers, etc.) and adds it to the page.

3. Show Error Message

```
function showError(message) {
  main.innerHTML = `
    <div class="card">
      <h2>${message}</h2>
    </div>
  `;
}
```

- If the username is wrong or there's an error, show a message like **"User not found"**.

4. Show Top 5 Repositories

```
function showRepos(repos) {
```

- Takes a list of repos and displays only the **top 5**.

```
  const reposDiv = document.getElementById("repos");
  reposDiv.innerHTML = "";
```

- Clears any previous repo list.

```
  repos.slice(0, 5).forEach(repo => {
    ...
  });
```

- Only shows the **first 5 repositories** using a loop.
- Creates a clickable link for each repo.

5. Fetch GitHub User Data

```
function fetchUser(username) {
  showLoading();
```

- When you search a name, it first shows **"Loading…"**.

```
  axios.get(apiURL + username)
```

- This line **calls GitHub API** and gets user data.

```
.then(res => {
  showUserCard(res.data);
  fetchRepos(username);
})
```

- If successful:
  - Shows the user card.
  - Also fetches their repos.

```
.catch(err => {
  ...
});
```

- If there's an error:
  - Show “User not found” or “Something went wrong”.

6. Fetch Repositories

```
function fetchRepos(username) {
  axios.get(`${apiURL}${username}/repos?sort=created`)
    .then(res => showRepos(res.data))
    .catch(() => showError("Error loading repositories."));
}
```

- Gets the user’s **repositories** and shows them.
- If it fails, show an error.

7. Form Submit Event

```
form.addEventListener("submit", (e) => {
  e.preventDefault();
```

- This listens for when the form is submitted (Enter key or click).
- Stops the page from refreshing.

```
  const username = input.value.trim();
```

- Gets the text from the input box.

```
  if (username) {
    fetchUser(username);
    input.value = "";
  }
});
```

- If the input is not empty:
  - Call `fetchUser` to get GitHub data.
  - Then **clear the input box**.