Expt 9- Demo of Promise using Async,Await

**Theory: Demo of Promise using Async/Await**

**1. What is a Promise?**

A Promise in JavaScript is an object that represents the eventual completion (or failure) of an asynchronous operation and its resulting value. It can be in one of three states:

* **Pending**: The initial state, neither fulfilled nor rejected.
* **Fulfilled**: The operation completed successfully.
* **Rejected**: The operation failed.

Promises are used to handle asynchronous operations in JavaScript, allowing for cleaner and more manageable code compared to traditional callback methods.

**2. What is Async/Await?**

async and await are syntactic sugar built on top of Promises, making asynchronous code easier to read and write.

* **Async Function**: A function declared with the async keyword. It always returns a Promise. Inside an async function, you can use the await keyword to pause the execution until the Promise is resolved.
* **Await**: The await keyword is used to wait for a Promise to resolve or reject. It can only be used inside an async function.

Code:-

// fetchGitHubData.js

const axios = require('axios');

// Function to fetch GitHub user data

const fetchGitHubUserData = async (username) => {

try {

// Make a request to the GitHub API

const response = await axios.get(`https://api.github.com/users/${username}`);

// Extract user data

const { id, login, public\_repos } = response.data;

// Display the fetched data

console.log(`User ID: ${id}`);

console.log(`Username: ${login}`);

console.log(`Public Repositories: ${public\_repos}`);

} catch (error) {

// Handle any errors

console.error(`Error fetching data for user ${username}:`, error.message);

}

};

// Replace 'octocat' with any GitHub username you'd like to fetch data for

fetchGitHubUserData('octocat');