Module 33-API, JSON, Data load, dynamic website

33-1 Module Introduction and What is an API

API=**Application Programming Interface**

https://www.somewhereinblog.net/blog/saumen0/30173919

33-2 Intro to JSON, JSON Structure, parse, stringify

JavaScript Object Notation (JSON)

stringify() method **converts a JavaScript object or value to a JSON string**,

parse() is **a crucial method for converting JSON data in string form into Javascript objects**.

```
// JavaScript Object Notation (JSON)
const user={id:191002012,name:"Hamid Hosen",job:"Programming"};
const stringified=JSON.stringify(user);
// console.log(user);
// console.log(stringified);

// stringify
const shop = {
  name: "Alia Store",
  address: "Ranbir road",
  profit: 15000,
  products: ["laptop", "mobile", "pepsi"],
  owner: {
    name: "Alia Bhatt",
    profession: "actor",
  },
  isExpensive: false,
};
const shopStringified=JSON.stringify(shop);
// console.log(shop);
console.log(shopStringified);
const converted=JSON.parse(shopStringified);
console.log(converted);
console.log(converted.owner.name);
```

33-3 JSON placeholder, GET data, display data on UI

The fetch() method in JavaScript is **used to request to the server and load the information in the webpages**.

```
function loadData(){
    fetch("https://jsonplaceholder.typicode.com/todos/1")
      .then((response) => response.json())
      .then((json) => console.log(json.title));
}

//   stringify JSON
// {
//   "userId": 1,
//   "id": 1,
//   "title": "delectus aut autem",
//   "completed": false
// }
```

33-4 Load more data, more APIs, send data to function

```
function loadUsers() {
  fetch("https://jsonplaceholder.typicode.com/users")
    .then((response) => response.json())
    // .then((json) => console.log(json));
    .then((json) => displayUsers(json));
}

function loadPosts() {
  fetch("https://jsonplaceholder.typicode.com/posts")
    .then((response) => response.json())
    .then((json) => console.log(json));
}

function displayUsers(data){
    console.log(data);
}
```

33-5 Dynamically display loaded data on your website

```
function displayUsers(data){
    const ul=document.getElementById('users');
    for(const user of data)
    {
```

```
        const li=document.createElement('li');
        li.innerText=`name : ${user.name} "............" email :
${user.email}`;
        ul.appendChild(li);
    }
}
```

33-6 Load posts and display on the website with CSS

```
function loadPosts() {
  fetch("https://jsonplaceholder.typicode.com/posts")
    .then((res) => res.json())
    .then((data) => displayPosts(data));
}
loadPosts();

function displayPosts(posts){
    const postContainer = document.getElementById("posts");
    for(const post of posts)
    {
        const div=document.createElement('div');
        div.classList.add("post");
        div.innerHTML = `
        <h3>${post.title}</h3>
        <p>${post.body}</p>
        `;
        postContainer.appendChild(div);
    }
}
```

33-7 GET, POST, PATCH, DELETE, CRUD, GET Vs POST

```
function addAPost() {
  fetch("https://jsonplaceholder.typicode.com/posts", {
    method: "POST",
    body: JSON.stringify({
      title: "My new post",
      body: "This is my posts",
      userId: 1,
    }),
    headers: {
      "Content-type": "application/json; charset=UTF-8",
    },
```

```
  })
    .then((response) => response.json())
    .then((json) => console.log(json));
}
addAPost();
```

Compare GET vs. POST

What is CRUD operation?

CRUD Meaning: CRUD is an acronym that comes from the world of computer programming and refers to the four functions that are considered necessary to implement a persistent storage application: **create, read, update and delete**.

What are CRUD operations in REST API?

Create, Read, Update, and Delete — CRUD — are the four major functions for interacting with database applications. CRUD functions often play a role in web-based REST APIs, where they map (albeit poorly) to the HTTP methods GET, POST, DELETE, PUT, and PATCH.

33-8 Debug API, Network tab,Status code, headers, bad API

33-9 Module Summary and two homeworks