**📘 Lecture Handout – Week 2**

**🔹 Topic 4: What Is an API and How Do Websites Talk to Each Other?**

**👨‍🏫 What You Need to Know**

* **API** means "Application Programming Interface." It helps websites and apps talk to each other.
* **MockAPI.io** is a free online tool that allows users to create fake REST APIs for testing and development purposes without the need for a real backend. It lets you set up endpoints such as GET, POST, PUT, and DELETE, and manage fake data like lists of users, books, or tasks. The responses are returned in JSON format, making it ideal for frontend developers who want to test their apps, build prototypes, or practice working with APIs. It's simple to use and perfect for classroom activities, project demos, or learning how APIs work.

**🔑 Basic Terms:**

* **GET** – Ask for data (e.g., show all books)
* **POST** – Send new data (e.g., add a book)
* **PUT** – Update something (e.g., edit a book)
* **DELETE** – Remove something (e.g., delete a book)

**🌟 Example:**

Website for a bookstore:

* GET /books → Show all books
* POST /books → Add a new book
* DELETE /books/3 → Delete book with ID 3

**✍️ Activity 4: Make Your Own Fake API**

**Instructions**:

1. Go to [https://mockapi.io](https://mockapi.io/).
2. Make a new fake project for a library.
3. Create these:
   * List of books (GET)
   * Add book (POST)
   * Update book (PUT)
   * Delete book (DELETE)
4. Take a screenshot of your setup.

**What to Submit**:

* Screenshot of your fake API
* One example of a POST request and response in JSON

**✅ Instructions**

**1. Go to** [**https://mockapi.io**](https://mockapi.io)

* Open the website in your browser.
* Sign up or log in using your email or GitHub account.

**2. Create a New Fake Project**

* Click **“Create New Project”**.
* Name it something like **"Library Management"** or **"Fake Library API"**.

**3. Set Up Your Resources**

You need to create one resource: **Books**.

**To do that:**

1. Inside your project, click **“Create resource”**.
2. Name it **“books”**.
3. Add the following fields (example schema):
   * title (string)
   * author (string)
   * yearPublished (number)
   * genre (string)
   * available (boolean)

This automatically gives you the following endpoints:

* **GET /books** – List of books
* **POST /books** – Add a new book
* **PUT /books/:id** – Update a book
* **DELETE /books/:id** – Delete a book

**4. Take a Screenshot of Your Setup**

* Once your resource is created and visible (showing fields and endpoints), take a screenshot.
* Make sure it shows the **Books** resource with the list and endpoint URL.

[

{

"id": "1",

"title": "1984",

"author": "George Orwell",

"yearPublished": 1949,

"genre": "Dystopian",

"available": true

},

{

"id": "2",

"title": "Pride and Prejudice",

"author": "Jane Austen",

"yearPublished": 1813,

"genre": "Romance",

"available": true

},

{

"id": "3",

"title": "The Great Gatsby",

"author": "F. Scott Fitzgerald",

"yearPublished": 1925,

"genre": "Classic",

"available": false

},

{

"id": "4",

"title": "To Kill a Mockingbird",

"author": "Harper Lee",

"yearPublished": 1960,

"genre": "Fiction",

"available": true

},

{

"id": "5",

"title": "The Hobbit",

"author": "J.R.R. Tolkien",

"yearPublished": 1937,

"genre": "Fantasy",

"available": false

}

]

**🔹 Topic 5: Getting Started with React (A JavaScript Tool for Websites)**

**👨‍🏫 What You Need to Know**

* **React** is a JavaScript tool used to build modern websites easily.
* It lets you make parts of a page (like buttons, forms) reusable and smart.
* **CodeSandbox.io** is an online code editor and development environment that lets you build and run web applications directly in your browser—no downloads or installations needed. It supports popular web technologies like **React**, **Vue**, **Angular**, **JavaScript**, **HTML**, and **CSS**, allowing you to create full-stack projects in real-time. CodeSandbox automatically sets up your project structure, provides live preview of your app, and allows you to share your work instantly through a URL. It’s widely used for **prototyping**, **learning**, **collaboration**, and **testing code** quickly. In short, it's like having a ready-to-use coding playground in your browser.

**💡 Simple React Ideas:**

* **Components**: Little building blocks like “Header,” “Footer,” “Task List.”
* **JSX**: A mix of JavaScript and HTML.
* **State**: React remembers things like a counter or a list of tasks.

**🌟 Example:**

function Hello() {

return <h1>Hello, World!</h1>;

}

**✍️ Activity 5: Make a Simple To-Do List App**

**Instructions**:

1. Go to [https://codesandbox.io](https://codesandbox.io/).
2. Make a new React app.
3. Add:
   * A box to type a task
   * A button to add it
   * A list to show your tasks

**What to Submit**:

* Link to your CodeSandbox
* Screenshot of your app

**✅ Step-by-Step Instructions**

**🖥 1. Go to CodeSandbox**

* Visit <https://codesandbox.io>
* Click **“Create Sandbox”**
* Choose **React** (the one with the React logo)

**🧩 2. Build the To-Do List App**

Here’s the code you’ll need:

**App.js**

import React, { useState } from "react";

export default function App() {

const [task, setTask] = useState("");

const [tasks, setTasks] = useState([]);

const handleAdd = () => {

if (task.trim()) {

setTasks([...tasks, task]);

setTask("");

}

};

return (

<div style={{ padding: "2rem" }}>

<h2>📝 My To-Do List</h2>

<input

type="text"

placeholder="Enter a task..."

value={task}

onChange={(e) => setTask(e.target.value)}

/>

<button onClick={handleAdd} style={{ marginLeft: "1rem" }}>

Add Task

</button>

<ul>

{tasks.map((t, index) => (

<li key={index}>{t}</li>

))}

</ul>

</div>

);

}

**📤 What to Submit**

**🔗 Link to your CodeSandbox**

* Click **“Share”** on the top right
* Copy and submit the **public link**

**📸 Screenshot of your app**

* Press **PrtScn** (Windows) or **Cmd+Shift+4** (Mac)
* Save the image and submit it with your assignment

**🔹 Topic 6: What Is Git and Why Do We Use GitHub?**

**👨‍🏫 What You Need to Know**

* **GitHub** is a web-based platform used to store, manage, and share code. It is built around Git, a version control system that tracks changes in code over time. GitHub allows developers to collaborate on projects by creating repositories, managing branches, submitting pull requests, and tracking issues. It is widely used for both personal and professional software development, enabling teams to work together efficiently, maintain version history, and contribute to open-source projects. GitHub also offers tools for project management, documentation, and code deployment, making it a powerful hub for modern software development.

**💻 Easy Git Commands:**

| **Command** | **What It Does** |
| --- | --- |
| git init | Start a new project |
| git add . | Save your changes |
| git commit -m "message" | Record the saved changes |
| git push | Send changes to GitHub |
| git status | See what’s going on |
| git log | See history of changes |

**✍️ Activity 6: Create Your First GitHub Project**

**Instructions**:

1. Go to [https://github.com](https://github.com/) and make a free account.
2. Create a new public repository called week2-practice.
3. Add a file called README.md.
4. Use Git to push your code (or upload it manually).
5. Make at least one commit.

**What to Submit**:

* Link to your GitHub repo
* Screenshot showing your commit history

**✅ Instructions:**

**🟢 1. Go to** [**https://github.com**](https://github.com/)

* Sign up for a **free account** if you don’t already have one.

**🟢 2. Create a New Public Repository**

* After logging in, click the **“+” icon** in the top-right corner > **New repository**
* Set the **Repository name** to: week2-practice
* Choose **Public**
* (Optional) Check “Initialize this repository with a README” if you want it added automatically
* Click **Create repository**

**🟢 3. Add a File Called README.md**

If not added yet:

* Click **“Add file” > “Create new file”**
* Name the file: README.md
* Add some text (e.g., “# Week 2 Practice”)
* Scroll down and click **Commit new file**

**🟢 4. Push Code via Git (or Upload Manually)**

**🔧 Option A: Using Git (if installed)**

1. Open your terminal or command prompt
2. Run the following commands:

git clone https://github.com/your-username/week2-practice.git

cd week2-practice

echo "# Week 2 Practice" > README.md

git add README.md

git commit -m "Add README"

git push origin main

**🖱 Option B: Upload Files Manually**

* Go to your repo on GitHub
* Click **Add file > Upload files**
* Choose files from your computer and upload
* Click **Commit changes**

**🟢 5. Make at Least One Commit**

* If you added or edited a file and saved it with a message, that counts as a commit.
* You can view your commits under the **“Commits” tab** of your repository.

**📤 What to Submit:**

* 🔗 **Link to your repository:**  
  Copy the URL (e.g., https://github.com/your-username/week2-practice)
* 📸 **Screenshot showing your commit history:**
  + Go to your repo > Click on **“X commits”** > Screenshot that page

**📋 Summary Table – Week 2**

| **Topic** | **What You Learn** | **Activity** | **Example** | **What You Turn In** |
| --- | --- | --- | --- | --- |
| 4. API Basics | How websites talk to each other | Fake API for library | Bookstore API | Screenshot + one JSON request |
| 5. Intro to React | Build simple interactive pages | To-Do List with React | React component | CodeSandbox link + Screenshot |
| 6. Git & GitHub Basics | Save code and work with others | Make a GitHub repo | Git commands | Repo link + commit screenshot |

**✅ Key Takeaways**

* APIs let different parts of a web app communicate.
* React makes it easy to build reusable parts of a website.
* Git and GitHub help you track your code and work with others.