1. JavaScript Basics & Setup

HTML:

html

CopyEdit

```html
<script src="main.js"></script>
```

main.js:

javascript

CopyEdit

```javascript
console.log("Welcome to the Community Portal");

window.onload = function() {
  alert("Page fully loaded");
};
```

---

2. Syntax, Data Types, and Operators

javascript

CopyEdit

```javascript
const eventName = "Community Yoga Session";
const eventDate = "2025-06-10";
let seatsAvailable = 25;

console.log(`${eventName} on ${eventDate} has ${seatsAvailable} seats.`);

// On registration
seatsAvailable--;
```

---

3. Conditionals, Loops, and Error Handling

javascript

CopyEdit

```
const events = [
  { name: "Yoga", date: "2025-06-10", seats: 5 },
  { name: "Coding Workshop", date: "2024-05-01", seats: 0 }
];

events.forEach(event => {
  const today = new Date().toISOString().split("T")[0];
  if (event.date >= today && event.seats > 0) {
    console.log(`${event.name} is available`);
  } else {
    console.log(`${event.name} is not available`);
  }
});

function register(event) {
  try {
    if (event.seats <= 0) throw "No seats available";
    event.seats--;
  } catch (err) {
    console.error("Registration error:", err);
  }
}
```

---

4. Functions, Scope, Closures, Higher-Order Functions

javascript

CopyEdit

```
function addEvent(name, category) {
  return { name, category };
}

function registerUser(eventName) {
  console.log(`User registered for ${eventName}`);
}

function filterEventsByCategory(events, category) {
  return events.filter(e => e.category === category);
}

// Closure example
function createCategoryCounter() {
  let count = 0;
  return function () {
    count++;
    return count;
  };
}
const musicCounter = createCategoryCounter();
```

---

5. Objects and Prototypes

javascript

CopyEdit

```
function Event(name, date, seats) {
  this.name = name;
  this.date = date;
  this.seats = seats;
```

```javascript
}

Event.prototype.checkAvailability = function() {
  return this.seats > 0;
};

const event1 = new Event("Music Fest", "2025-07-01", 10);

console.log(Object.entries(event1));
```

---

6. Arrays and Methods

javascript

CopyEdit

```javascript
let eventList = [
  { name: "Music Night", category: "music" },
  { name: "Book Club", category: "literature" }
];

eventList.push({ name: "Jazz Evening", category: "music" });

const musicEvents = eventList.filter(e => e.category === "music");

const formatted = eventList.map(e => `Event: ${e.name}`);
console.log(formatted);
```

---

7. DOM Manipulation

javascript

CopyEdit

```
const container = document.querySelector("#eventContainer");

function displayEvent(event) {
  const card = document.createElement("div");
  card.className = "event-card";
  card.textContent = event.name;
  container.appendChild(card);
}

function updateUIOnRegister() {
  document.querySelector("#registerBtn").textContent = "Registered!";
}
```

---

8. Event Handling

javascript

CopyEdit

```
document.querySelector("#registerBtn").onclick = () => alert("You are registered!");

document.querySelector("#categorySelect").onchange = function () {
  console.log("Selected category:", this.value);
};

document.querySelector("#searchInput").addEventListener("keydown", e => {
  console.log("Searching for:", e.target.value);
});
```

---

9. Async JS, Promises, Async/Await

javascript

CopyEdit

```
// Using fetch with promises
fetch("events.json")
  .then(res => res.json())
  .then(data => console.log(data))
  .catch(err => console.error(err));

// Using async/await
async function loadEvents() {
  document.getElementById("loader").style.display = "block";
  try {
    const res = await fetch("events.json");
    const data = await res.json();
    console.log(data);
  } finally {
    document.getElementById("loader").style.display = "none";
  }
}
```

---

10. Modern JavaScript Features

javascript

CopyEdit

```
const getEventDetails = ({ name, date }) => `${name} is on ${date}`;

let original = [{ name: "Yoga", seats: 10 }];
let clone = [...original]; // Spread operator

function greet(name = "Guest") {
  console.log(`Hello, ${name}`);
}
```

---

11. Working with Forms

javascript

CopyEdit

```
document.querySelector("#regForm").addEventListener("submit", function(e) {
  e.preventDefault();
  const name = this.elements["name"].value;
  const email = this.elements["email"].value;
  const event = this.elements["event"].value;

  if (!name || !email || !event) {
    document.getElementById("error").textContent = "All fields required!";
    return;
  }

  console.log("Registered:", { name, email, event });
});
```

---

12. AJAX & Fetch API

javascript

CopyEdit

```
function submitForm(data) {
  fetch("https://mockapi.io/submit", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(data)
  })
    .then(res => res.json())
    .then(() => alert("Submitted successfully!"))
    .catch(() => alert("Submission failed."));

  setTimeout(() => {
    console.log("Simulated delay done");
  }, 2000);
}
```

---

13. Debugging and Testing

- Use console.log() at each step (e.g., form data, API request).
- Open Chrome DevTools → Console tab for logs and errors.
- Use **Network tab** to verify fetch requests and payloads.
- Use **Sources tab** to add breakpoints and watch variables.

---

14. jQuery and JS Frameworks

javascript

CopyEdit

```
// jQuery usage
$("#registerBtn").click(() => alert("Registered via jQuery"));

$(".event-card").fadeIn();
$(".event-card").fadeOut();

// Framework benefit
// React makes UI component-based, reusable, and state-driven.
```