Module 1-JavaScript

1.JavaScript Basics & Setup

CODE:

(i).index.html:

```
<!DOCTYPE html>
<html>
<head><title>Community Portal</title></head>
<body>
  <h1>Welcome to Our Community Portal</h1>
  <script src="main.js"></script>
</body>
</html>
```

(ii)main.js

```
console.log("Welcome to the Community Portal");
window.onload = () => alert("Page fully loaded");
```

OUTPUT:

Console: Welcome to the Community Portal
Alert: Page fully loaded

2. Syntax, Data Types, and Operators

CODE:

```
const eventName = "Local Art Fair";
const eventDate = "2025-06-30";
let availableSeats = 25;

console.log(`Event: ${eventName}, Date: ${eventDate}, Seats: ${availableSeats}`);
availableSeats--;
```

OUTPUT:

Event: Local Art Fair, Date: 2025-06-30, Seats: 25

3. Conditionals, Loops, and Error Handling

CODE:

```
const events = [
  { name: "Music Fest", date: "2025-07-01", seats: 0 },
  { name: "Art Show", date: "2025-06-25", seats: 10 }
];

const today = new Date();

events.forEach(event => {
  const eventDate = new Date(event.date);
  if (eventDate > today && event.seats > 0) {
    console.log(`Upcoming: ${event.name}`);
  }
});

function registerEvent(event) {
  try {
    if (event.seats <= 0) throw new Error("No seats available.");
    event.seats--;
    console.log("Registered successfully!");
  } catch (e) {
    console.error(e.message);
  }
}
```

OUTPUT:

Upcoming: Art Show

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

CODE:

```
function addEvent(name, category) {
  return { name, category, seats: 20 };
}

function registerUser(event) {
  if (event.seats > 0) {
    event.seats--;
    return true;
  }
```

```
  return false;
}

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

function registrationCounter() {
  let count = 0;
  return () => ++count;
}

const musicCounter = registrationCounter();
console.log(musicCounter()); // 1
console.log(musicCounter()); // 2
```

OUTPUT:

```
1
2
```

5. Objects and Prototypes

CODE:

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

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

const event1 = new Event("Tech Meetup", "2025-08-01", 10);
console.log(Object.entries(event1));
```

OUTPUT:

```
[["name", "Tech Meetup"], ["date", "2025-08-01"], ["seats", 10]]
```

6. Arrays and Methods

CODE:

```
let eventList = [
  { name: "Workshop on Baking", category: "Food" },
  { name: "Live Music", category: "Music" }
];

eventList.push({ name: "Dance Show", category: "Music" });

const musicEvents = eventList.filter(e => e.category === "Music");
const displayCards = eventList.map(e => `Event: ${e.name}`);

console.log(musicEvents);
console.log(displayCards);
```

OUTPUT:

```
[
  { name: "Live Music", category: "Music" },
  { name: "Dance Show", category: "Music" }
]
["Event: Workshop on Baking", "Event: Live Music", "Event: Dance Show"]
```

7. DOM Manipulation

CODE:

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

function renderEvent(event) {
  const card = document.createElement("div");
  card.textContent = `${event.name} - ${event.date}`;
  eventContainer.appendChild(card);
}
```

8. Event Handling

CODE:

```
document.querySelector("#registerBtn").onclick = () => {
  console.log("Register clicked");
};

document.querySelector("#categoryFilter").onchange = (e) => {
  console.log("Filtered by:", e.target.value);
};

document.querySelector("#searchInput").addEventListener("keydown", (e) => {
  if (e.key === "Enter") {
    console.log("Search term:", e.target.value);
  }
});
```

OUTPUT:

Console logs user actions when interacting with form elements.

9. Async JS, Promises, Async/Await

```
// Promise version
fetch("https://mockapi.io/events")
  .then(res => res.json())
  .then(data => console.log(data))
  .catch(err => console.error(err));
// Async/await version
async function loadEvents() {
  try {
    document.getElementById("spinner").style.display = "block";
    const res = await fetch("https://mockapi.io/events");
    const data = await res.json();
    console.log(data);
  } catch (e) {
    console.error(e);
  } finally {
    document.getElementById("spinner").style.display = "none";
  }
}
```

OUTPUT:

Logs fetched data.
Shows/hides spinner.

10. Modern JavaScript Features

CODE:

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

const { name, date } = { name: "Fair", date: "2025-09-15" };
const cloned = [...eventList];

greet();
greet("Tom");
```

OUTPUT:

Hello, Guest
Hello, Tom

11. Working with Forms

CODE:

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

  if (!name || !email) {
    document.querySelector("#error").textContent = "All fields required.";
  } else {
    console.log("Submitted:", name, email);
  }
});
```

OUTPUT:

Logs submitted data or shows error if fields are empty.

12. AJAX & Fetch API

CODE:

```
function submitRegistration(userData) {
  fetch("https://mockapi.io/register", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(userData),
  })
    .then(res => res.json())
    .then(() => {
      setTimeout(() => alert("Registration successful!"), 1000);
    })
    .catch(() => alert("Submission failed."));
}
```

OUTPUT:

A **POST request** is made to the mock API.
After 1 second:

- ✅ If successful: ✅ alert("Registration successful!")
- ❌ If failed: ❌ alert("Submission failed.")

13. Debugging and Testing

CODE:

```
function submitForm() {
  const userData = { name: "Jane", email: "jane@example.com" };
  console.log("Form submitted");
```

```
  console.log("Payload:", userData);

  fetch("https://mockapi.io/register", {
    method: "POST",
    body: JSON.stringify(userData),
    headers: { "Content-Type": "application/json" }
  });
}

submitForm();
```

OUTPUT:

```
Form submitted
Payload: { name: "Jane", email: "jane@example.com" }
```

14. jQuery and JS Frameworks

(i)html

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<button id="registerBtn">Register</button>
<div class="eventCard" style="display:none;">Music Night</div>
```

(ii)Script

```
$('#registerBtn').click(() => {
  alert("Registered via jQuery!");
});

$('.eventCard').fadeIn();  // Show event cards
setTimeout(() => $('.eventCard').fadeOut(), 3000);  // Fade out after 3s
```

OUTPUT:

click **Register** button → alert("Registered via jQuery!")
.eventCard smoothly fades in, then fades out after 3 seconds