
Fitness Tracking Application

MVC.html

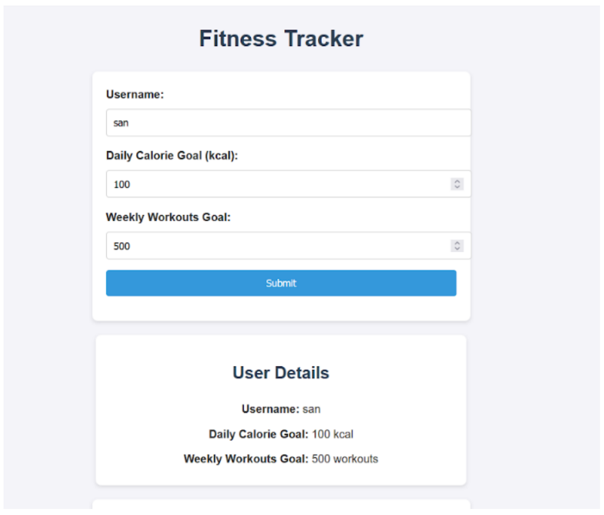
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Fitness Tracker</title>
</head>

<body>
    <h1>Fitness Tracker</h1>
    <!-- User details -->
    <form id="user-details-form">
        <label for="username">Username:</label>
        <input type="text" id="username" placeholder="Enter your name" required>

        <label for="daily-calories-goal">Daily Calorie Goal (kcal):</label>
        <input type="number" id="daily-calories-goal" placeholder="Enter daily calorie goal" required>

        <label for="weekly-workouts-goal">Weekly Workouts Goal:</label>
        <input type="number" id="weekly-workouts-goal" placeholder="Enter weekly workout goal" required>

        <button type="submit">Submit</button>
    </form>
    <div id="user-details-display"></div>
    <!-- Workout -->
    <form id="workout-form" style="display: none;">
        <label for="exercise">Exercise:</label>
        <select id="exercise">
            <option value="Running">Running</option>
            <option value="Cycling">Cycling</option>
            <option value="Yoga">Yoga</option>
            <option value="Weightlifting">Weightlifting</option>
        </select>
        <label for="duration">Duration (minutes):</label>
        <input type="number" id="duration" required>
        <button type="submit">Add Workout</button>
    </form>
    <!-- Display -->
    <h2>Workout History</h2>
    <ul id="history-list"></ul>
    <h2>Progress</h2>
    <p>Total Calories Burned: <span id="total-calories">0</span></p>
    <p>Total Workouts: <span id="total-workouts">0</span></p>
    <p id="goal-status"></p>
    <script src="mvc.js"></script>
</body>
</html>
```

MVC.jS

```
// Model
const user = {
    username: "",
    dailyCaloriesGoal: 0,
    weeklyWorkoutsGoal: 0,
    workoutHistory: [],
    totalCaloriesBurned: 0,
    totalWorkouts: 0,
};
function addWorkout(exercise, duration) {
    const caloriesPerUnit = getCaloriesPerUnit(exercise);
    const caloriesBurned = caloriesPerUnit * duration;

    const workout = {
        exercise: exercise,
        duration: duration,
        caloriesBurned: caloriesBurned
    };
    user.workoutHistory.push(workout);
    user.totalCaloriesBurned += caloriesBurned;
    user.totalWorkouts += 1;
}
function getCaloriesPerUnit(exercise) {
    const caloriesPerMinute = {
        Running: 10,
        Cycling: 8,
        Yoga: 4,
        Weightlifting: 6
    };
    return caloriesPerMinute[exercise] || 0;
}
// Controller
document.getElementById('user-details-form').addEventListener('submit', function(event) {
    event.preventDefault();
    const username = document.getElementById('username').value;
    const dailyCaloriesGoal = parseInt(document.getElementById('daily-calories-goal').value);
    const weeklyWorkoutsGoal = parseInt(document.getElementById('weekly-workouts-
goal').value);
    user.username = username;
    user.dailyCaloriesGoal = dailyCaloriesGoal;
    user.weeklyWorkoutsGoal = weeklyWorkoutsGoal;

    displayUserDetails();
    document.getElementById('workout-form').style.display = 'block';
});
document.getElementById('workout-form').addEventListener('submit', function(event) {
    event.preventDefault();
    const exercise = document.getElementById('exercise').value;
    const duration = parseInt(document.getElementById('duration').value);
    addWorkout(exercise, duration);
    updateWorkoutHistory();
    updateProgress();
});
function displayUserDetails() {
    const displayDiv = document.getElementById('user-details-display');
    displayDiv.innerHTML = `
        <h2>User Details</h2>
        <p><strong>Username:</strong> ${user.username}</p>
        <p><strong>Daily Calorie Goal:</strong> ${user.dailyCaloriesGoal} kcal</p>
        <p><strong>Weekly Workouts Goal:</strong> ${user.weeklyWorkoutsGoal}
workouts</p>
    `;
}
function updateWorkoutHistory() {
    const historyList = document.getElementById('history-list');
    historyList.innerHTML = '';
    user.workoutHistory.forEach(workout => {
        const listItem = document.createElement('li');
        listItem.textContent = `${workout.exercise}: ${workout.duration} minutes,
${workout.caloriesBurned} calories burned`;
```

```
        historyList.appendChild(listItem);
    });
}
function updateProgress() {
    document.getElementById('total-calories').textContent = user.totalCaloriesBurned;
    document.getElementById('total-workouts').textContent = user.totalWorkouts;

    const goalStatus = document.getElementById('goal-status');
    if (user.totalCaloriesBurned >= user.dailyCaloriesGoal) {
        goalStatus.textContent = "You’ve reached your daily calorie goal!";
    } else {
        goalStatus.textContent = "Keep going! You haven’t reached your daily calorie goal
yet.";
    }
}
```

Output:



Fitness Tracker

Username: san

Daily Calorie Goal (kcal): 100

Weekly Workouts Goal: 500

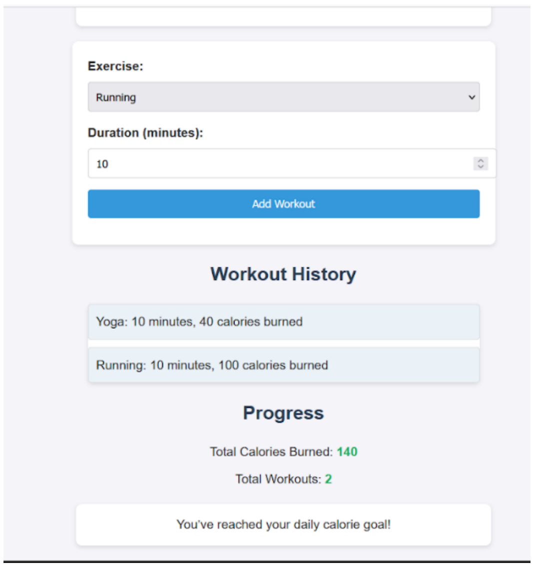
Submit

User Details

Username: san

Daily Calorie Goal: 100 kcal

Weekly Workouts Goal: 500 workouts



Exercise: Running

Duration (minutes): 10

Add Workout

Workout History

Yoga: 10 minutes, 40 calories burned

Running: 10 minutes, 100 calories burned

Progress

Total Calories Burned: 140

Total Workouts: 2

You've reached your daily calorie goal!

Online Quiz Application:

index.html (view):

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Online Quiz App</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="quiz-container">
        <h1>Online Quiz</h1>
        <div id="question-container">
            <h2 id="question"></h2>
            <div id="options"></div>
        </div>
        <button id="next-btn">Next</button>
        <p id="progress"></p>
    </div>
    <script src="model.js"></script>
    <script src="script.js"></script>
</body>
</html>
```

script.js(controller):

```
document.addEventListener("DOMContentLoaded", () => {
  const questionContainer = document.getElementById("question");
  const optionsContainer = document.getElementById("options");
  const nextButton = document.getElementById("next-btn");
  const progressContainer = document.getElementById("progress");

  function displayQuestion() {
      let currentQuestion = quizModel.questions[quizModel.currentQuestionIndex];
      questionContainer.innerText = currentQuestion.question;

      optionsContainer.innerHTML = "";
      currentQuestion.options.forEach(option => {
          const button = document.createElement("button");
          button.innerText = option;
          button.onclick = () => handleAnswer(option);
          optionsContainer.appendChild(button);
      });
      progressContainer.innerText = `Question ${quizModel.currentQuestionIndex + 1} of
${quizModel.questions.length}`;
  }
  function handleAnswer(selectedOption) {
      quizModel.checkAnswer(selectedOption);
      quizModel.nextQuestion();
      if (quizModel.isQuizOver()) {
          showResults();
```

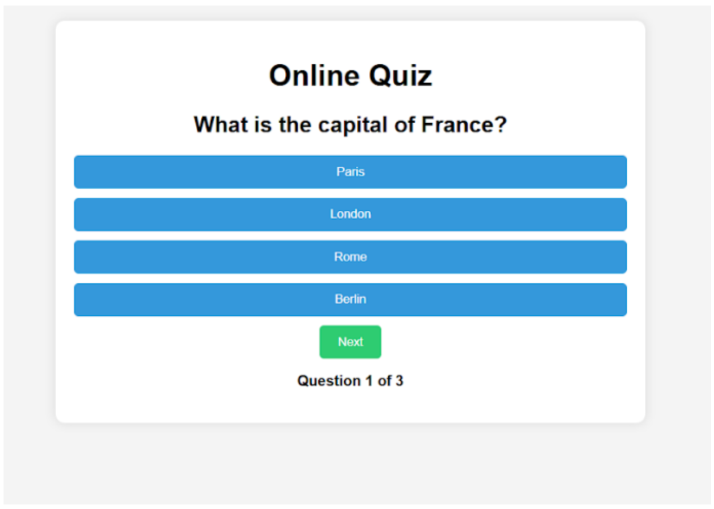
```
    } else {
      displayQuestion();
    }
  }
  function showResults() {
    questionContainer.innerText = `Quiz Completed! Your Score: ${quizModel.score} /
${quizModel.questions.length}`;
    optionsContainer.innerHTML = "";
    nextButton.style.display = "none";
  }
  nextButton.addEventListener("click", () => {
    if (!quizModel.isQuizOver()) {
      displayQuestion();
    }
  });
  displayQuestion();
});
```

model.js (Model):

```
const quizModel = {
  currentQuestionIndex: 0,
  score: 0,
  questions: [
    {
      question: "What is the capital of France?",
      options: ["Paris", "London", "Rome", "Berlin"],
      answer: "Paris"
    },
    {
      question: "Which language is used for web development?",
      options: ["Python", "Java", "HTML", "C++"],
      answer: "HTML"
    },
    {
      question: "What is 5 + 3?",
      options: ["5", "8", "10", "15"],
      answer: "8"
    }
  ],
  checkAnswer(selectedOption) {
    if (selectedOption === this.questions[this.currentQuestionIndex].answer) {
      this.score++;
    }
```

```
 },
 nextQuestion() {
    this.currentQuestionIndex++;
 },
 isQuizOver() {
    return this.currentQuestionIndex >= this.questions.length;
 }
};
```

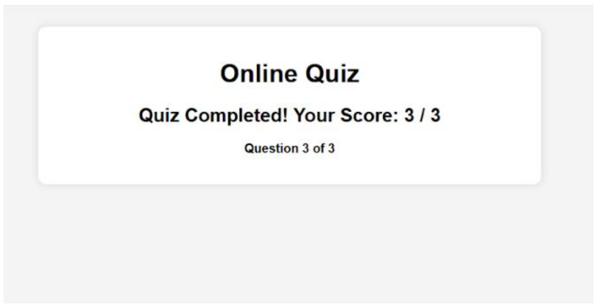
Output:



Online Quiz

What is the capital of France?

Question 1 of 3



Online Quiz

Quiz Completed! Your Score: 3 / 3

Question 3 of 3

Digital Library System

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Digital Library System</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <header>
      <h1>Digital Library System</h1>
   </header>
```

```
    <main>
        <section class="search-section">
            <input type="text" id="search-input" placeholder="Search books by title or author">
            <button id="search-button">Search</button>
        </section>

        <section class="books-section">
            <h2>Books Collection</h2>
            <div id="books-container"></div>
        </section>
        <section class="details-section" id="details-section" style="display: none;">
            <h2>Book Details</h2>
            <div id="book-details"></div>
            <button id="close-details">Close</button>
        </section>
    </main>
    <script src="model.js"></script>
    <script src="script.js"></script>
</body>
</html>
```

script.js(controller):

```
document.addEventListener("DOMContentLoaded", () => {
    const booksContainer = document.getElementById("books-container");
    const searchInput = document.getElementById("search-input");
    const searchButton = document.getElementById("search-button");
    const detailsSection = document.getElementById("details-section");
    const bookDetails = document.getElementById("book-details");
    const closeDetailsButton = document.getElementById("close-details");

    // Display all books
    const displayBooks = (books) => {
        booksContainer.innerHTML = "";
        books.forEach((book, index) => {
            const bookCard = document.createElement("div");
            bookCard.classList.add("book-card");

            bookCard.innerHTML = `
                <div>
                    <strong>${book.title}</strong><br>
                    Author: ${book.author}<br>
                    Genre: ${book.genre}<br>
                    Available: ${book.isAvailable ? "Yes" : "No"}
```

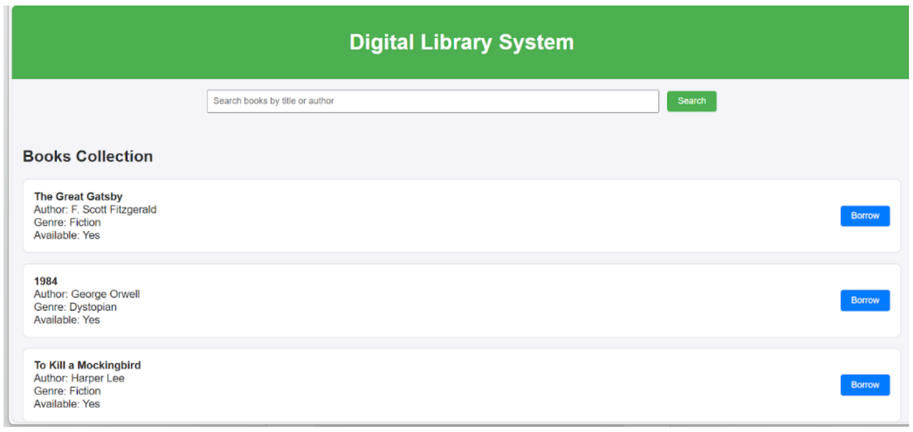
```
      </div>
      <button class="${book.isAvailable ? '' : 'borrowed'}"
          data-index="${index}"
          ${book.isAvailable ? "" : "disabled"}>
        ${book.isAvailable ? "Borrow" : "Unavailable"}
      </button>
    `;

    booksContainer.appendChild(bookCard);
  });
};

// Search books
const searchBooks = () => {
  const keyword = searchInput.value.toLowerCase();
  const filteredBooks = library.filter(
    (book) =>
      book.title.toLowerCase().includes(keyword) ||
      book.author.toLowerCase().includes(keyword)
  );
  displayBooks(filteredBooks);
};

// Borrow book
const borrowBook = (index) => {
  if (library[index].isAvailable) {
    library[index].isAvailable = false;
    alert(`You borrowed "${library[index].title}" successfully!`);
    displayBooks(library);
  } else {
    alert("Sorry, this book is unavailable.");
  }
};

// View book details
const viewDetails = (index) => {
  const book = library[index];
  bookDetails.innerHTML = `
    <h3>${book.title}</h3>
    <p><strong>Author:</strong> ${book.author}</p>
    <p><strong>Genre:</strong> ${book.genre}</p>
    <p><strong>Available:</strong> ${book.isAvailable ? "Yes" : "No"}</p>
  `;
```

```
        detailsSection.style.display = "block";
    };

    // Initialize event listeners
    searchButton.addEventListener("click", searchBooks);

    booksContainer.addEventListener("click", (e) => {
        if (e.target.tagName === "BUTTON") {
            const index = e.target.getAttribute("data-index");
            if (e.target.innerText === "Borrow") {
                borrowBook(index);
            } else {
                viewDetails(index);
            }
        }
    });

    closeDetailsButton.addEventListener("click", () => {
        detailsSection.style.display = "none";
    });

    // Initial display
    displayBooks(library);
});
```

model.js (Model):

```
class Book {
    constructor(title, author, genre, isAvailable) {
        this.title = title;
        this.author = author;
        this.genre = genre;
        this.isAvailable = isAvailable;
    }
}
const library = [
    new Book("The Great Gatsby", "F. Scott Fitzgerald", "Fiction", true),
    new Book("1984", "George Orwell", "Dystopian", true),
    new Book("To Kill a Mockingbird", "Harper Lee", "Fiction", true),
    new Book("The Catcher in the Rye", "J.D. Salinger", "Fiction", false),
];
```

OUTPUT:


Digital Library System

Search books by title or author

Search

Books Collection

The Great Gatsby
Author: F. Scott Fitzgerald
Genre: Fiction
Available: Yes

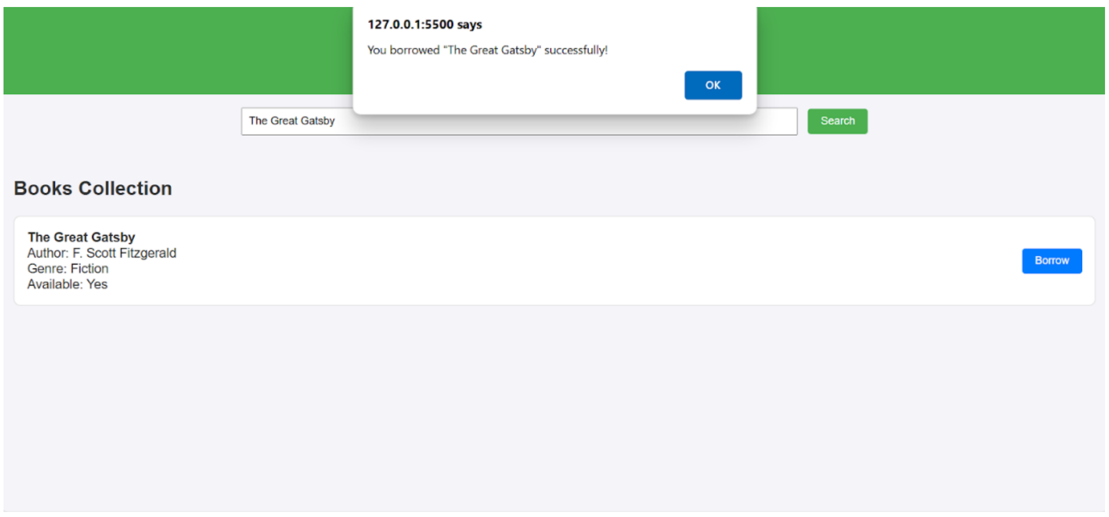
Borrow

1984
Author: George Orwell
Genre: Dystopian
Available: Yes

Borrow

To Kill a Mockingbird
Author: Harper Lee
Genre: Fiction
Available: Yes

Borrow


127.0.0.1:5500 says

You borrowed "The Great Gatsby" successfully!

OK

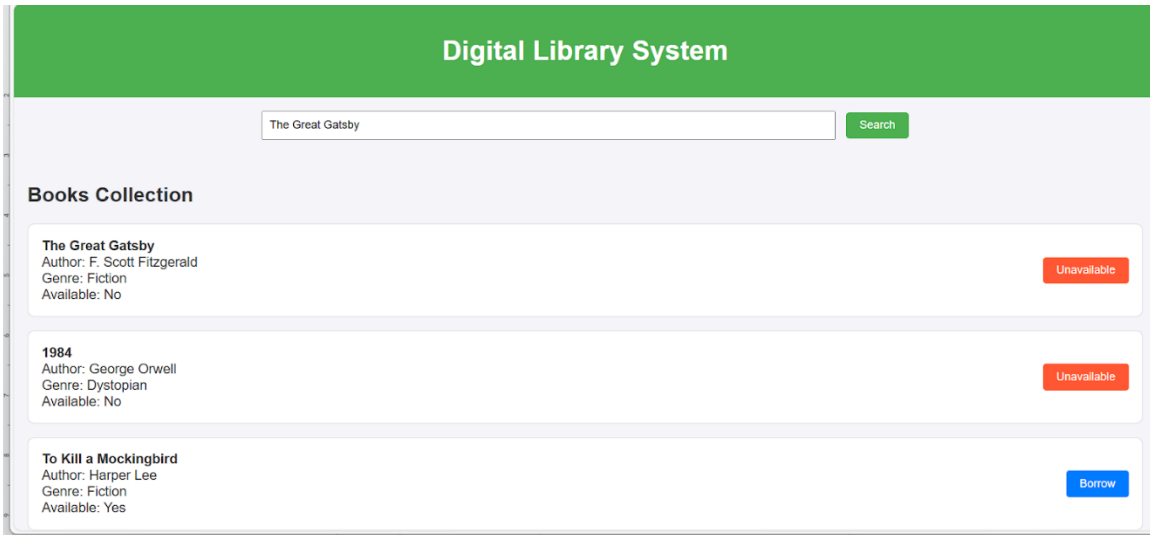
The Great Gatsby

Search

Books Collection

The Great Gatsby
Author: F. Scott Fitzgerald
Genre: Fiction
Available: Yes

Borrow



Digital Library System

The Great Gatsby

Books Collection

The Great Gatsby
Author: F. Scott Fitzgerald
Genre: Fiction
Available: No

1984
Author: George Orwell
Genre: Dystopian
Available: No

To Kill a Mockingbird
Author: Harper Lee
Genre: Fiction
Available: Yes

CONTENT MANAGEMENT SYSTEM (CMS)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Simple CMS</title>
  <style>
    body {
        font-family: Arial, sans-serif;
        margin: 20px;
    }
    .content-item {
        border: 1px solid #ccc;
        padding: 10px;
        margin: 10px 0;
    }
    .content-item button {
        margin-left: 10px;
    }
    form {
        margin-bottom: 20px;
    }
    input, select, textarea, button {
        display: block;
        margin: 10px 0;
```

```
      width: 100%;
      padding: 8px;
    }
  </style>

</head>
<body>
  <h2>Content Management System</h2>

  <form id="addContentForm">
    <label>Type:</label>
    <select id="contentType">
      <option value="article">Article</option>
      <option value="blog">Blog</option>
      <option value="media">Media</option>
    </select>
    <input type="text" id="contentTitle" placeholder="Title" required>
    <input type="text" id="contentAuthor" placeholder="Author" required>
    <textarea id="contentText" placeholder="Content" required></textarea>
    <button type="submit">Add Content</button>
  </form>

  <div id="contentList"></div>

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

SCRIPT.JS

MODEL

```
class Content {
  constructor(id, type, title, author, createdAt, content) {
    this.id = id;
    this.type = type;
    this.title = title;
    this.author = author;
    this.createdAt = createdAt || new Date().toLocaleString();
    this.content = content;
  }
}

const contentDB = [
  new Content(1, "article", "Intro to CMS", "Alice", "2025-01-20", "This is an article
content."),
  new Content(2, "blog", "My Travel Blog", "Bob", "2025-01-21", "Welcome to my travel
blog."),
];

const ContentModel = {
  getAll: () => contentDB,
  getById: (id) => contentDB.find(item => item.id === id),
  add: (content) => contentDB.push(content),
  update: (id, newData) => {
    const index = contentDB.findIndex(item => item.id === id);
    if (index !== -1) contentDB[index] = { ...contentDB[index], ...newData };
  },
  delete: (id) => {
    const index = contentDB.findIndex(item => item.id === id);
    if (index !== -1) contentDB.splice(index, 1);
  }
};
```

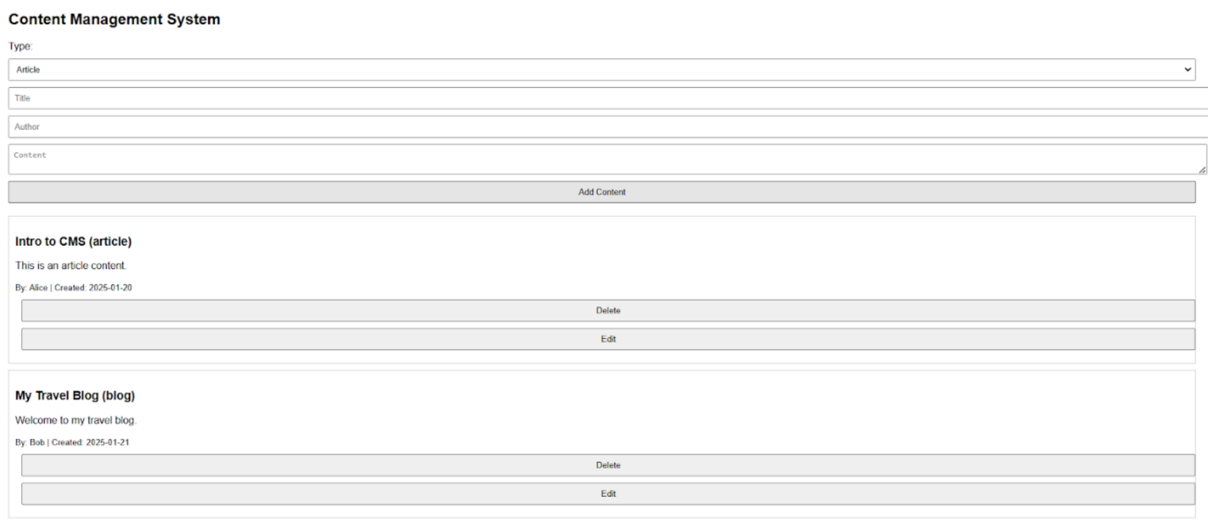
CONTROLLER

```
const ContentController = {
  loadContent: () => {
    const contentList = ContentModel.getAll();
    View.renderContentList(contentList);
  },
  createContent: (type, title, author, content) => {
    const newContent = new Content(Date.now(), type, title, author, new
Date().toLocaleString(), content);
    ContentModel.add(newContent);
    View.renderContentList(ContentModel.getAll());
  },
  editContent: (id, newData) => {
    ContentModel.update(id, newData);
    View.renderContentList(ContentModel.getAll());
  },

  deleteContent: (id) => {
    ContentModel.delete(id);
    View.renderContentList(ContentModel.getAll());
  }
};
```

VIEW

```
const View = {
  renderContentList: (contentList) => {
    const contentDiv = document.getElementById("contentList");
    contentDiv.innerHTML = "";

    contentList.forEach(item => {
      const div = document.createElement("div");
      div.className = "content-item";
      div.innerHTML = `
        <h3>${item.title} (${item.type})</h3>
        <p>${item.content}</p>
<small>By: ${item.author} | Created: ${item.createdAt}</small>
        <button onclick="ContentController.deleteContent(${item.id})">Delete</button>
        <button onclick="editContentPrompt(${item.id})">Edit</button>
      `;
      contentDiv.appendChild(div);
    });
  }
};
function editContentPrompt(id) {
  const newTitle = prompt("Enter new title:");
  const newContent = prompt("Enter new content:");
  if (newTitle && newContent) {
    ContentController.editContent(id, { title: newTitle, content: newContent });
  }
}

document.getElementById("addContentForm").addEventListener("submit", function (e) {
  e.preventDefault();
  const type = document.getElementById("contentType").value;
  const title = document.getElementById("contentTitle").value;
  const author = document.getElementById("contentAuthor").value;
  const content = document.getElementById("contentText").value;
  ContentController.createContent(type, title, author, content);
  e.target.reset();
});

ContentController.loadContent();
```

OUTPUT



Content Management System

Type:

Article

Title

Author

Content

Add Content

Intro to CMS (article)

This is an article content.

By: Alice | Created: 2025-01-20

Delete

Edit

My Travel Blog (blog)

Welcome to my travel blog.

By: Bob | Created: 2025-01-21

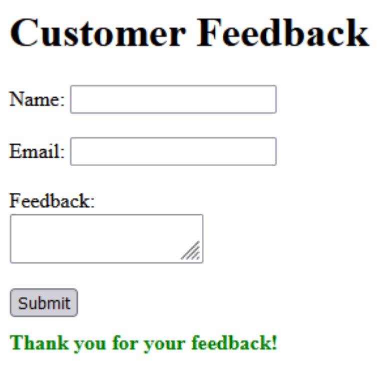
Delete

Edit

CUSTOMER FEEDBACK FORM USING MVC

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Customer Feedback Form</title>
  <style>
        #responseMessage {
        margin-top: 10px;
        font-weight: bold;
        }
        .error {
        color: red;
        }
        .success {
        color: green;
        }
  </style>
</head>
<body>
  <h1>Customer Feedback</h1>

  <form id="surveyForm">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required><br><br>

        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required><br><br>

        <label for="feedback">Feedback:</label><br>
        <textarea id="feedback" name="feedback" required></textarea><br><br>
        <button type="submit">Submit</button>
</form>

<div id="responseMessage"></div>

<script>

        class SurveyModel {
        constructor() {
        this.responses = JSON.parse(localStorage.getItem('surveyResponses')) || [];
        }

        saveResponse(response) {
        this.responses.push(response);
        localStorage.setItem('surveyResponses', JSON.stringify(this.responses));
        }
        }

        class SurveyView {
        constructor() {
        this.form = document.getElementById('surveyForm');
        this.messageElement = document.getElementById('responseMessage');
        }

        getFormData() {
        const name = document.getElementById('name').value;
        const email = document.getElementById('email').value;
        const feedback = document.getElementById('feedback').value;
        return { name, email, feedback };
        }

        displayMessage(message, type) {
        this.messageElement.textContent = message;
        this.messageElement.className = type;
        }

        resetForm() {
        this.form.reset();
        }
        }


        class SurveyController {
```

```
        constructor(view, model) {
        this.view = view;
        this.model = model;
        }

        validateInput(name, email, feedback) {
        return name && email && feedback;
        }

        submitForm(name, email, feedback) {
        if (this.validateInput(name, email, feedback)) {
        const response = { name, email, feedback, timestamp: new Date().toISOString() };
        this.model.saveResponse(response);
        this.view.displayMessage('Thank you for your feedback!', 'success');
        this.view.resetForm();
        return true;
        } else {
        this.view.displayMessage('Please fill in all the fields correctly.', 'error');
        return false;
        }}
        }
        const surveyModel = new SurveyModel();
        const surveyView = new SurveyView();
        const surveyController = new SurveyController(surveyView, surveyModel);

        surveyView.form.addEventListener('submit', function(event) {
        event.preventDefault();

        const { name, email, feedback } = surveyView.getFormData();
        surveyController.submitForm(name, email, feedback);
        });
  </script>
</body>
</html>
```


Customer Feedback

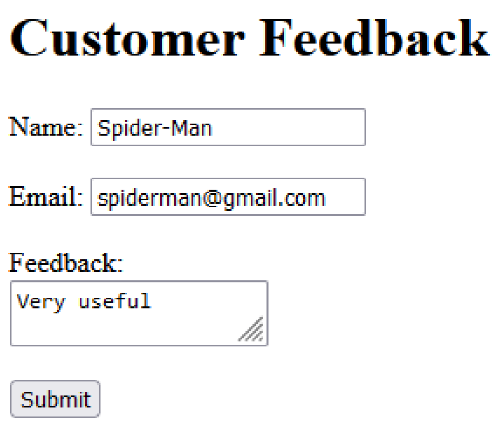
Name:

Email:

Feedback:

Submit

Thank you for your feedback!


Customer Feedback

Name: Spider-Man

Email: spiderman@gmail.com

Feedback:

Very useful

Submit

Problem Identification (5)	Execution (5)	Time management (5)	Viva (5)	Total (20)

Result:

Thus the above mvc programs run successfully and verified.