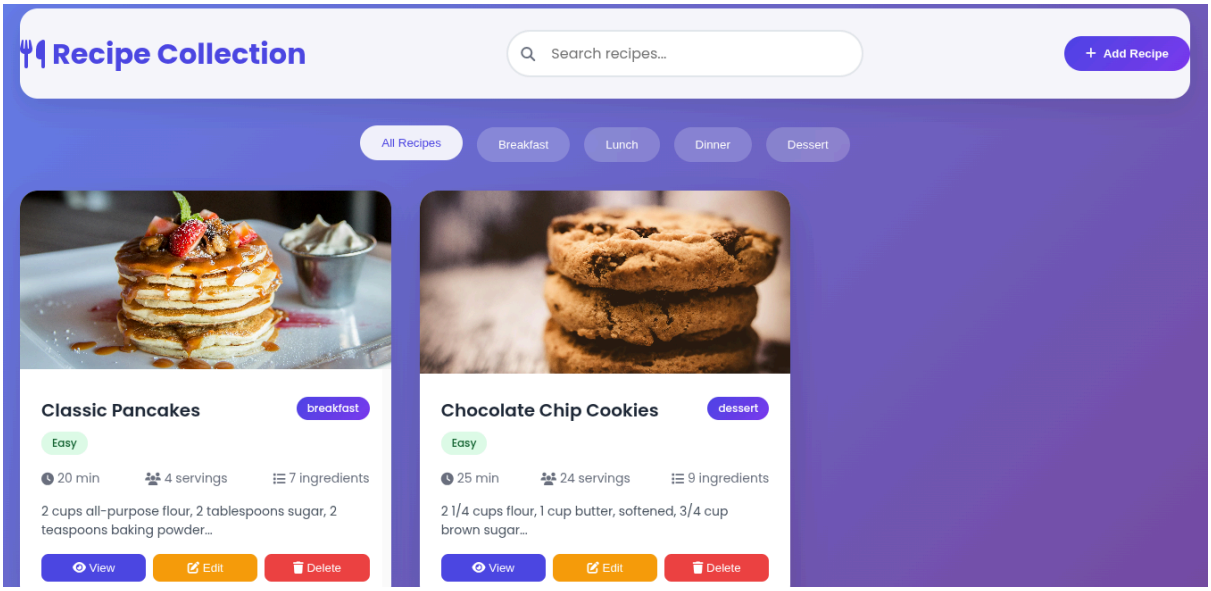
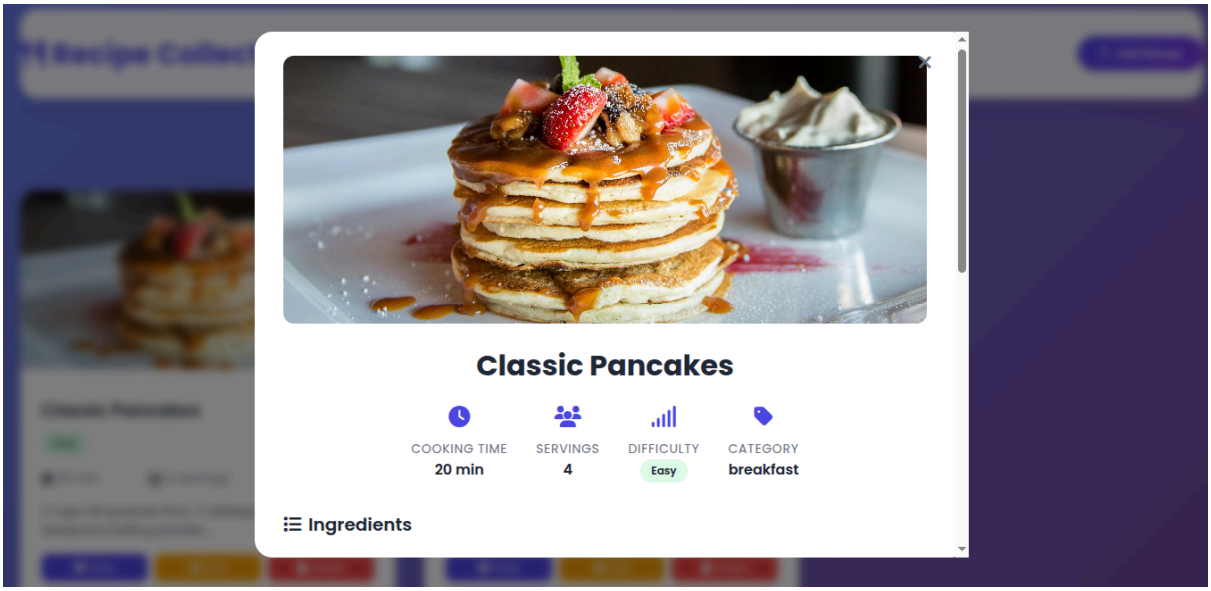
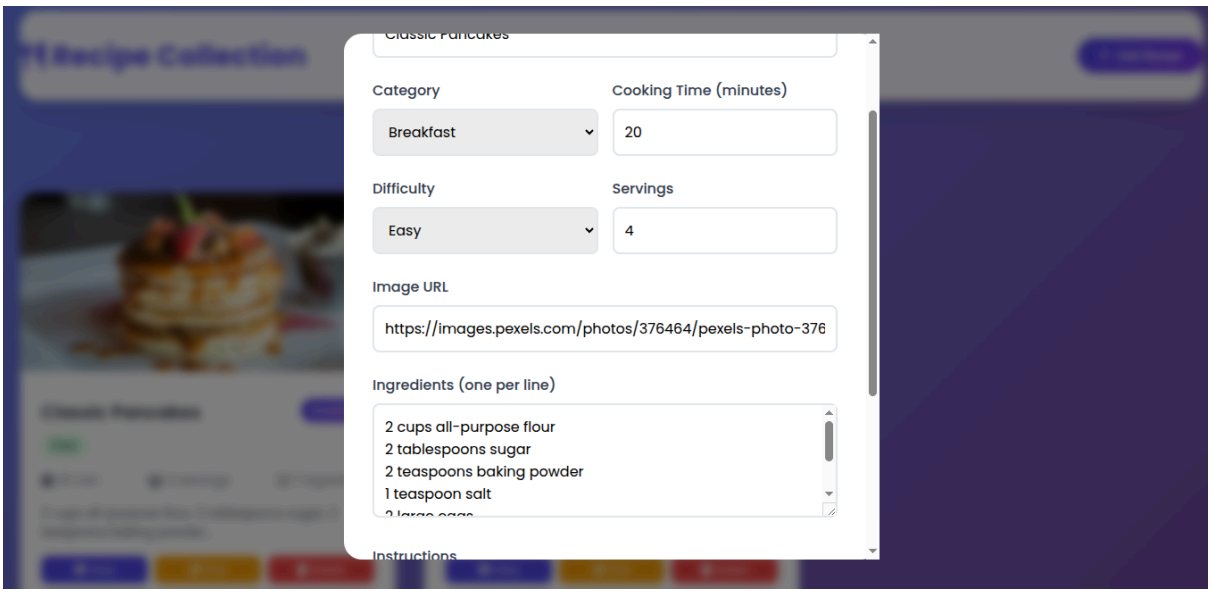
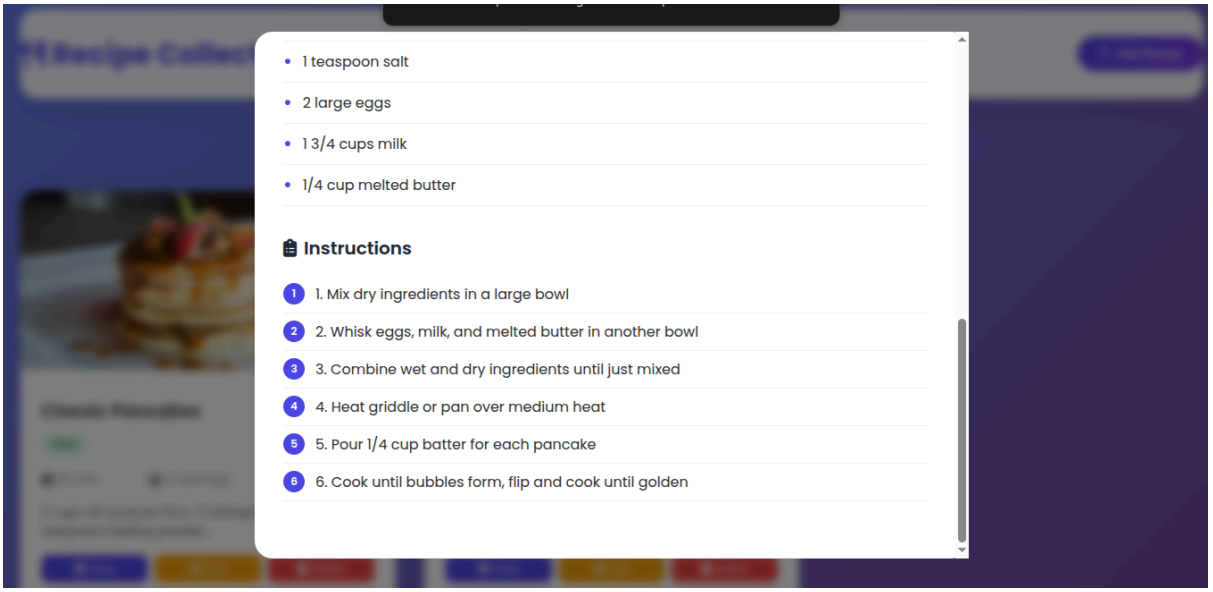

Recipe Collection
Search recipes...
Add Recipe
All Recipes
Breakfast
Lunch
Dinner
Dessert
Classic Pancakes
breakfast
Easy
20 min
4 servings
7 ingredients
2 cups all-purpose flour, 2 tablespoons sugar, 2 teaspoons baking powder...
View
Edit
Delete
Chocolate Chip Cookies
dessert
Easy
25 min
24 servings
9 ingredients
2 1/4 cups flour, 1 cup butter, softened, 3/4 cup brown sugar...
View
Edit
Delete


Classic Pancakes
COOKING TIME
20 min
SERVINGS
4
DIFFICULTY
Easy
CATEGORY
breakfast
Ingredients





//HTML

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

```
    <link rel="stylesheet" href="style.css">
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&display=swap" rel="stylesheet">
    <link
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css" rel="stylesheet">
</head>
<body>
    <div class="container">
        <!-- Header -->
        <header class="header">
            <div class="header-content">
                <h1><i class="fas fa-utensils"></i> Recipe
Collection</h1>
                <div class="search-container">
                    <input type="text" id="searchInput"
placeholder="Search recipes..." class="search-input">
                    <i class="fas fa-search search-icon"></i>
                </div>
                <button class="add-recipe-btn"
id="addRecipeBtn">
                    <i class="fas fa-plus"></i> Add Recipe
                </button>
            </div>
        </header>

        <!-- Filter Tabs -->
        <div class="filter-tabs">
            <button class="tab-btn active"
data-category="all">All Recipes</button>
            <button class="tab-btn"
data-category="breakfast">Breakfast</button>
            <button class="tab-btn"
data-category="lunch">Lunch</button>
            <button class="tab-btn"
data-category="dinner">Dinner</button>
```

```
            <button class="tab-btn"
data-category="dessert">Dessert</button>
        </div>

        <!-- Recipe Cards Grid -->
        <div class="recipes-grid" id="recipesGrid">
            <!-- Recipe cards will be dynamically generated -->
        </div>

        <!-- Recipe Modal -->
        <div class="modal" id="recipeModal">
            <div class="modal-content">
                <span class="close-btn"
id="closeModal">&times;</span>
                <form id="recipeForm">
                    <h2 id="modalTitle">Add New Recipe</h2>

                    <div class="form-group">
                        <label for="recipeName">Recipe
Name</label>
                        <input type="text" id="recipeName"
required>
                    </div>

                    <div class="form-row">
                        <div class="form-group">
                            <label
for="recipeCategory">Category</label>
                            <select id="recipeCategory"
required>
                                <option value="">Select
Category</option>
                                <option
value="breakfast">Breakfast</option>
                                <option
value="lunch">Lunch</option>
```

```
                                <option
value="dinner">Dinner</option>
                                <option
value="dessert">Dessert</option>
                            </select>
                        </div>
                        <div class="form-group">
                            <label for="cookingTime">Cooking
Time (minutes)</label>
                            <input type="number"
id="cookingTime" required>
                        </div>
                    </div>

                    <div class="form-row">
                        <div class="form-group">
                            <label
for="difficulty">Difficulty</label>
                            <select id="difficulty" required>
                                <option value="">Select
Difficulty</option>
                                <option
value="easy">Easy</option>
                                <option
value="medium">Medium</option>
                                <option
value="hard">Hard</option>
                            </select>
                        </div>
                        <div class="form-group">
                            <label
for="servings">Servings</label>
                            <input type="number" id="servings"
required>
                        </div>
                    </div>
```

```
                    <div class="form-group">
                        <label for="imageUrl">Image URL</label>
                        <input type="url" id="imageUrl"
placeholder="https://example.com/image.jpg">
                    </div>

                    <div class="form-group">
                        <label for="ingredients">Ingredients
(one per line)</label>
                        <textarea id="ingredients" rows="4"
required placeholder="2 cups flour&#10;1 cup sugar&#10;3
eggs"></textarea>
                    </div>

                    <div class="form-group">
                        <label
for="instructions">Instructions</label>
                        <textarea id="instructions" rows="6"
required placeholder="1. Preheat oven to 350°F&#10;2. Mix dry
ingredients..."></textarea>
                    </div>

                    <div class="form-actions">
                        <button type="button"
id="cancelBtn">Cancel</button>
                        <button type="submit">Save
Recipe</button>
                    </div>
                </form>
            </div>
        </div>

        <!-- Recipe Detail Modal -->
        <div class="modal" id="recipeDetailModal">
            <div class="modal-content recipe-detail">
                <span class="close-btn"
id="closeDetailModal">&times;</span>
```

```
                <div id="recipeDetailContent">
                    <!-- Recipe details will be dynamically
generated -->
                </div>
            </div>
        </div>
    </div>

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

//css

```
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: 'Poppins', sans-serif;
    background: linear-gradient(135deg, #667eea 0%, #764ba2
100%);
    min-height: 100vh;
    color: #333;
}

.container {
    max-width: 1400px;
    margin: 0 auto;
    padding: 0 20px;
}

/* Header Styles */
```

```css
.header {
    background: rgba(255, 255, 255, 0.95);
    backdrop-filter: blur(10px);
    padding: 1.5rem 0;
    margin-bottom: 2rem;
    border-radius: 20px;
    margin-top: 2rem;
    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);
}

.header-content {
    display: flex;
    justify-content: space-between;
    align-items: center;
    flex-wrap: wrap;
    gap: 1rem;
}

.header h1 {
    color: #4f46e5;
    font-size: 2rem;
    font-weight: 700;
    display: flex;
    align-items: center;
    gap: 0.5rem;
}

.search-container {
    position: relative;
    flex: 1;
    max-width: 400px;
}

.search-input {
    width: 100%;
    padding: 0.75rem 1rem 0.75rem 3rem;
    border: 2px solid #e5e7eb;
```

```
    border-radius: 50px;
    font-size: 1rem;
    transition: all 0.3s ease;
}

.search-input:focus {
    outline: none;
    border-color: #4f46e5;
    box-shadow: 0 0 0 3px rgba(79, 70, 229, 0.1);
}

.search-icon {
    position: absolute;
    left: 1rem;
    top: 50%;
    transform: translateY(-50%);
    color: #6b7280;
}

.add-recipe-btn {
    background: linear-gradient(135deg, #4f46e5, #7c3aed);
    color: white;
    border: none;
    padding: 0.75rem 1.5rem;
    border-radius: 50px;
    font-weight: 600;
    cursor: pointer;
    display: flex;
    align-items: center;
    gap: 0.5rem;
    transition: all 0.3s ease;
}

.add-recipe-btn:hover {
    transform: translateY(-2px);
    box-shadow: 0 10px 25px rgba(79, 70, 229, 0.3);
}
```

```css
/* Filter Tabs */
.filter-tabs {
    display: flex;
    justify-content: center;
    gap: 1rem;
    margin-bottom: 2rem;
    flex-wrap: wrap;
}

.tab-btn {
    padding: 0.75rem 1.5rem;
    border: none;
    background: rgba(255, 255, 255, 0.2);
    color: white;
    border-radius: 50px;
    font-weight: 500;
    cursor: pointer;
    transition: all 0.3s ease;
    backdrop-filter: blur(10px);
}

.tab-btn:hover,
.tab-btn.active {
    background: rgba(255, 255, 255, 0.9);
    color: #4f46e5;
    transform: translateY(-2px);
}

/* Recipe Grid */
.recipes-grid {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(320px,
1fr));
    gap: 2rem;
    margin-bottom: 2rem;
}
```

```css
/* Recipe Card */
.recipe-card {
    background: white;
    border-radius: 20px;
    overflow: hidden;
    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);
    transition: all 0.4s ease;
    cursor: pointer;
    opacity: 0;
    transform: translateY(20px);
    animation: fadeInUp 0.6s ease forwards;
}

.recipe-card:hover {
    transform: translateY(-10px);
    box-shadow: 0 20px 40px rgba(0, 0, 0, 0.15);
}

.recipe-card.removing {
    animation: fadeOutDown 0.4s ease forwards;
}

@keyframes fadeInUp {
    to {
        opacity: 1;
        transform: translateY(0);
    }
}

@keyframes fadeOutDown {
    to {
        opacity: 0;
        transform: translateY(20px) scale(0.9);
    }
}
```

```css
.recipe-image {
    width: 100%;
    height: 200px;
    object-fit: cover;
    transition: transform 0.4s ease;
}

.recipe-card:hover .recipe-image {
    transform: scale(1.05);
}

.recipe-content {
    padding: 1.5rem;
}

.recipe-header {
    display: flex;
    justify-content: space-between;
    align-items: flex-start;
    margin-bottom: 1rem;
}

.recipe-title {
    font-size: 1.25rem;
    font-weight: 600;
    color: #1f2937;
    margin-bottom: 0.5rem;
}

.recipe-category {
    background: linear-gradient(135deg, #4f46e5, #7c3aed);
    color: white;
    padding: 0.25rem 0.75rem;
    border-radius: 50px;
    font-size: 0.75rem;
    font-weight: 500;
}
```

```css
.recipe-meta {
    display: flex;
    justify-content: space-between;
    margin-bottom: 1rem;
    font-size: 0.875rem;
    color: #6b7280;
}

.meta-item {
    display: flex;
    align-items: center;
    gap: 0.25rem;
}

.recipe-description {
    color: #4b5563;
    font-size: 0.875rem;
    line-height: 1.5;
    margin-bottom: 1rem;
}

.recipe-actions {
    display: flex;
    gap: 0.5rem;
}

.action-btn {
    flex: 1;
    padding: 0.5rem;
    border: none;
    border-radius: 8px;
    cursor: pointer;
    font-weight: 500;
    transition: all 0.3s ease;
}
```

```css
.view-btn {
    background: #4f46e5;
    color: white;
}

.edit-btn {
    background: #f59e0b;
    color: white;
}

.delete-btn {
    background: #ef4444;
    color: white;
}

.action-btn:hover {
    transform: translateY(-1px);
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.15);
}

/* Difficulty Badge */
.difficulty {
    display: inline-block;
    padding: 0.25rem 0.75rem;
    border-radius: 50px;
    font-size: 0.75rem;
    font-weight: 500;
    text-transform: capitalize;
}

.difficulty.easy {
    background: #dcfce7;
    color: #166534;
}

.difficulty.medium {
    background: #fef3c7;
```

```
    color: #92400e;
}

.difficulty.hard {
    background: #fee2e2;
    color: #991b1b;
}

/* Modal Styles */
.modal {
    display: none;
    position: fixed;
    z-index: 1000;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    background: rgba(0, 0, 0, 0.5);
    backdrop-filter: blur(5px);
    opacity: 0;
    transition: opacity 0.3s ease;
}

.modal.show {
    display: flex;
    align-items: center;
    justify-content: center;
    opacity: 1;
}

.modal-content {
    background: white;
    padding: 2rem;
    border-radius: 20px;
    width: 90%;
    max-width: 600px;
    max-height: 90vh;
```

```
    overflow-y: auto;
    transform: scale(0.9);
    transition: transform 0.3s ease;
    position: relative;
}

.modal.show .modal-content {
    transform: scale(1);
}

.close-btn {
    position: absolute;
    top: 1rem;
    right: 1.5rem;
    font-size: 2rem;
    color: #6b7280;
    cursor: pointer;
    transition: color 0.3s ease;
}

.close-btn:hover {
    color: #ef4444;
}

/* Form Styles */
.form-group {
    margin-bottom: 1.5rem;
}

.form-row {
    display: grid;
    grid-template-columns: 1fr 1fr;
    gap: 1rem;
}

label {
    display: block;
```

```
    margin-bottom: 0.5rem;
    font-weight: 500;
    color: #374151;
}

input,
select,
textarea {
    width: 100%;
    padding: 0.75rem;
    border: 2px solid #e5e7eb;
    border-radius: 8px;
    font-size: 1rem;
    transition: border-color 0.3s ease;
    font-family: inherit;
}

input:focus,
select:focus,
textarea:focus {
    outline: none;
    border-color: #4f46e5;
    box-shadow: 0 0 0 3px rgba(79, 70, 229, 0.1);
}

textarea {
    resize: vertical;
    min-height: 100px;
}

.form-actions {
    display: flex;
    gap: 1rem;
    justify-content: flex-end;
    margin-top: 2rem;
}
```

```css
.form-actions button {
    padding: 0.75rem 1.5rem;
    border: none;
    border-radius: 8px;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
}

.form-actions button[type="submit"] {
    background: #4f46e5;
    color: white;
}

.form-actions button[type="button"] {
    background: #6b7280;
    color: white;
}

.form-actions button:hover {
    transform: translateY(-1px);
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.15);
}

/* Recipe Detail Modal */
.recipe-detail {
    max-width: 800px;
}

.recipe-detail-header {
    text-align: center;
    margin-bottom: 2rem;
}

.recipe-detail-image {
    width: 100%;
    height: 300px;
```

```
    object-fit: cover;
    border-radius: 12px;
    margin-bottom: 1rem;
}

.recipe-detail-title {
    font-size: 2rem;
    font-weight: 700;
    color: #1f2937;
    margin-bottom: 1rem;
}

.recipe-detail-meta {
    display: flex;
    justify-content: center;
    gap: 2rem;
    margin-bottom: 2rem;
    flex-wrap: wrap;
}

.detail-meta-item {
    text-align: center;
}

.detail-meta-item .icon {
    font-size: 1.5rem;
    color: #4f46e5;
    margin-bottom: 0.5rem;
}

.detail-meta-item .label {
    font-size: 0.875rem;
    color: #6b7280;
    text-transform: uppercase;
    letter-spacing: 0.05em;
}
```

```css
.detail-meta-item .value {
    font-weight: 600;
    color: #1f2937;
}

.recipe-section {
    margin-bottom: 2rem;
}

.section-title {
    font-size: 1.25rem;
    font-weight: 600;
    color: #1f2937;
    margin-bottom: 1rem;
    display: flex;
    align-items: center;
    gap: 0.5rem;
}

.ingredients-list,
.instructions-list {
    list-style: none;
}

.ingredients-list li,
.instructions-list li {
    padding: 0.5rem 0;
    border-bottom: 1px solid #f3f4f6;
    display: flex;
    align-items: center;
    gap: 0.75rem;
}

.ingredients-list li:before {
    content: "•";
    color: #4f46e5;
    font-weight: bold;
```

```
    font-size: 1.2rem;
}

.instructions-list li {
    counter-increment: step-counter;
}

.instructions-list li:before {
    content: counter(step-counter);
    background: #4f46e5;
    color: white;
    border-radius: 50%;
    width: 24px;
    height: 24px;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 0.75rem;
    font-weight: 600;
    flex-shrink: 0;
}

.instructions-list {
    counter-reset: step-counter;
}

/* Empty State */
.empty-state {
    text-align: center;
    padding: 4rem 2rem;
    color: white;
}

.empty-state i {
    font-size: 4rem;
    margin-bottom: 1rem;
    opacity: 0.5;
```

```
}

.empty-state h3 {
    font-size: 1.5rem;
    margin-bottom: 0.5rem;
}

.empty-state p {
    opacity: 0.8;
}

/* Responsive Design */
@media (max-width: 768px) {
    .header-content {
        flex-direction: column;
        text-align: center;
    }

    .header h1 {
        font-size: 1.5rem;
    }

    .search-container {
        order: 1;
        max-width: none;
    }

    .add-recipe-btn {
        order: 2;
    }

    .filter-tabs {
        gap: 0.5rem;
    }

    .tab-btn {
        padding: 0.5rem 1rem;
```

```
        font-size: 0.875rem;
    }

    .recipes-grid {
        grid-template-columns: 1fr;
        gap: 1rem;
    }

    .form-row {
        grid-template-columns: 1fr;
    }

    .recipe-detail-meta {
        gap: 1rem;
    }

    .modal-content {
        width: 95%;
        padding: 1.5rem;
    }
}

/* Loading Animation */
.loading {
    display: inline-block;
    width: 20px;
    height: 20px;
    border: 3px solid rgba(255, 255, 255, 0.3);
    border-radius: 50%;
    border-top-color: #fff;
    animation: spin 1s ease-in-out infinite;
}

@keyframes spin {
    to {
        transform: rotate(360deg);
    }
```

```
}
```

//script.js

```
class RecipeApp {
    constructor() {
        this.recipes = this.loadRecipes();
        this.currentEditId = null;
        this.currentFilter = 'all';

        this.initializeElements();
        this.bindEvents();
        this.renderRecipes();
    }

    initializeElements() {
        this.recipesGrid =
document.getElementById('recipesGrid');
        this.addRecipeBtn =
document.getElementById('addRecipeBtn');
        this.recipeModal =
document.getElementById('recipeModal');
        this.recipeDetailModal =
document.getElementById('recipeDetailModal');
        this.recipeForm =
document.getElementById('recipeForm');
        this.searchInput =
document.getElementById('searchInput');
        this.tabBtns = document.querySelectorAll('.tab-btn');
        this.closeModalBtns =
document.querySelectorAll('.close-btn');
        this.cancelBtn = document.getElementById('cancelBtn');
    }

    bindEvents() {
        this.addRecipeBtn.addEventListener('click', () =>
this.openModal());
```

```
        this.recipeForm.addEventListener('submit', (e) =>
this.handleSubmit(e));
        this.searchInput.addEventListener('input', (e) =>
this.handleSearch(e));
        this.cancelBtn.addEventListener('click', () =>
this.closeModal());

        this.closeModalBtns.forEach(btn => {
            btn.addEventListener('click', () =>
this.closeModal());
        });

        this.tabBtns.forEach(btn => {
            btn.addEventListener('click', (e) =>
this.handleFilter(e));
        });

        // Close modal on outside click
        [this.recipeModal,
this.recipeDetailModal].forEach(modal => {
            modal.addEventListener('click', (e) => {
                if (e.target === modal) this.closeModal();
            });
        });

        // Keyboard shortcuts
        document.addEventListener('keydown', (e) => {
            if (e.key === 'Escape') this.closeModal();
            if (e.key === 'n' && e.ctrlKey) {
                e.preventDefault();
                this.openModal();
            }
        });
    }

    loadRecipes() {
        const stored = localStorage.getItem('recipes');
```

```
        if (stored) {
            return JSON.parse(stored);
        }

        // Default recipes with sample data
        return [
            {
                id: 1,
                name: "Classic Pancakes",
                category: "breakfast",
                cookingTime: 20,
                difficulty: "easy",
                servings: 4,
                imageUrl:
"https://images.pexels.com/photos/376464/pexels-photo-376464.j
peg",
                ingredients: ["2 cups all-purpose flour", "2
tablespoons sugar", "2 teaspoons baking powder", "1 teaspoon
salt", "2 large eggs", "1 3/4 cups milk", "1/4 cup melted
butter"],
                instructions: "1. Mix dry ingredients in a
large bowl\n2. Whisk eggs, milk, and melted butter in another
bowl\n3. Combine wet and dry ingredients until just mixed\n4.
Heat griddle or pan over medium heat\n5. Pour 1/4 cup batter
for each pancake\n6. Cook until bubbles form, flip and cook
until golden"
            },
            {
                id: 3,
                name: "Chocolate Chip Cookies",
                category: "dessert",
                cookingTime: 25,
                difficulty: "easy",
                servings: 24,
                imageUrl:
"https://images.pexels.com/photos/230325/pexels-photo-230325.j
peg",
```

```
                ingredients: ["2 1/4 cups flour", "1 cup
butter, softened", "3/4 cup brown sugar", "1/2 cup white
sugar", "2 eggs", "1 tsp vanilla", "1 tsp baking soda", "1/2
tsp salt", "2 cups chocolate chips"],
                instructions: "1. Preheat oven to 375°F\n2.
Cream butter and sugars\n3. Beat in eggs and vanilla\n4. Mix
in flour, baking soda, and salt\n5. Stir in chocolate
chips\n6. Drop spoonfuls on baking sheet\n7. Bake 9-11 minutes
until golden"
            }
        ];
    }

    saveRecipes() {
        localStorage.setItem('recipes',
JSON.stringify(this.recipes));
    }

    generateId() {
        return Date.now() + Math.random();
    }

    openModal(recipe = null) {
        this.currentEditId = recipe ? recipe.id : null;
        const modalTitle =
document.getElementById('modalTitle');
        modalTitle.textContent = recipe ? 'Edit Recipe' : 'Add
New Recipe';

        if (recipe) {
            this.populateForm(recipe);
        } else {
            this.recipeForm.reset();
        }

        this.recipeModal.classList.add('show');
        document.body.style.overflow = 'hidden';
```

```
    }

    populateForm(recipe) {
        document.getElementById('recipeName').value =
recipe.name;
        document.getElementById('recipeCategory').value =
recipe.category;
        document.getElementById('cookingTime').value =
recipe.cookingTime;
        document.getElementById('difficulty').value =
recipe.difficulty;
        document.getElementById('servings').value =
recipe.servings;
        document.getElementById('imageUrl').value =
recipe.imageUrl || '';
        document.getElementById('ingredients').value =
Array.isArray(recipe.ingredients)
            ? recipe.ingredients.join('\n')
            : recipe.ingredients;
        document.getElementById('instructions').value =
recipe.instructions;
    }

    closeModal() {
        this.recipeModal.classList.remove('show');
        this.recipeDetailModal.classList.remove('show');
        document.body.style.overflow = '';
        this.currentEditId = null;
    }

    handleSubmit(e) {
        e.preventDefault();

        const formData = new FormData(this.recipeForm);
        const recipe = {
            id: this.currentEditId || this.generateId(),
```

```
            name: formData.get('recipeName') ||
document.getElementById('recipeName').value,
            category: formData.get('recipeCategory') ||
document.getElementById('recipeCategory').value,
            cookingTime: parseInt(formData.get('cookingTime')
|| document.getElementById('cookingTime').value),
            difficulty: formData.get('difficulty') ||
document.getElementById('difficulty').value,
            servings: parseInt(formData.get('servings') ||
document.getElementById('servings').value),
            imageUrl: formData.get('imageUrl') ||
document.getElementById('imageUrl').value,
            ingredients: (formData.get('ingredients') ||
document.getElementById('ingredients').value)
                .split('\n')
                .filter(ingredient => ingredient.trim())
                .map(ingredient => ingredient.trim()),
            instructions: formData.get('instructions') ||
document.getElementById('instructions').value
        };

        if (this.currentEditId) {
            const index = this.recipes.findIndex(r => r.id ===
this.currentEditId);
            if (index !== -1) {
                this.recipes[index] = recipe;
            }
        } else {
            this.recipes.push(recipe);
        }

        this.saveRecipes();
        this.renderRecipes();
        this.closeModal();

        // Show success message
        this.showNotification(
```

```
            this.currentEditId ? 'Recipe updated successfully!'
: 'Recipe added successfully!',
            'success'
        );
    }

    handleSearch(e) {
        const searchTerm = e.target.value.toLowerCase();
        this.renderRecipes(searchTerm);
    }

    handleFilter(e) {
        this.tabBtns.forEach(btn =>
btn.classList.remove('active'));
        e.target.classList.add('active');
        this.currentFilter = e.target.dataset.category;
        this.renderRecipes();
    }

    renderRecipes(searchTerm = '') {
        let filteredRecipes = this.recipes;

        // Apply category filter
        if (this.currentFilter !== 'all') {
            filteredRecipes = filteredRecipes.filter(recipe =>
                recipe.category === this.currentFilter
            );
        }

        // Apply search filter
        if (searchTerm) {
            filteredRecipes = filteredRecipes.filter(recipe =>
                recipe.name.toLowerCase().includes(searchTerm)
||
                recipe.ingredients.some(ingredient =>

ingredient.toLowerCase().includes(searchTerm)
```

```
                )
            );
        }

        if (filteredRecipes.length === 0) {
            this.recipesGrid.innerHTML = `
                <div class="empty-state">
                    <i class="fas fa-utensils"></i>
                    <h3>No recipes found</h3>
                    <p>${searchTerm ? 'Try adjusting your
search terms' : 'Start by adding your first recipe!'}</p>
                </div>
            `;
            return;
        }

        this.recipesGrid.innerHTML = filteredRecipes
            .map((recipe, index) =>
this.createRecipeCard(recipe, index))
            .join('');

        // Bind card events
        this.bindCardEvents();
    }

    createRecipeCard(recipe, index) {
        const defaultImage =
"https://images.pexels.com/photos/1640777/pexels-photo-1640777
.jpeg";
        const imageUrl = recipe.imageUrl &&
recipe.imageUrl.trim() ? recipe.imageUrl : defaultImage;

        return `
            <div class="recipe-card" style="animation-delay:
${index * 0.1}s">
                <img src="${imageUrl}" alt="${recipe.name}"
class="recipe-image"
```

```
                    onerror="this.src='${defaultImage}'">
                <div class="recipe-content">
                    <div class="recipe-header">
                        <div>
                            <h3
class="recipe-title">${recipe.name}</h3>
                            <span class="difficulty
${recipe.difficulty}">${recipe.difficulty}</span>
                        </div>
                        <span
class="recipe-category">${recipe.category}</span>
                    </div>

                    <div class="recipe-meta">
                        <div class="meta-item">
                            <i class="fas fa-clock"></i>
                            <span>${recipe.cookingTime}
min</span>
                        </div>
                        <div class="meta-item">
                            <i class="fas fa-users"></i>
                            <span>${recipe.servings}
servings</span>
                        </div>
                        <div class="meta-item">
                            <i class="fas fa-list"></i>
                            <span>${recipe.ingredients.length}
ingredients</span>
                        </div>
                    </div>

                    <p class="recipe-description">
                        ${recipe.ingredients.slice(0,
3).join(', ')}${recipe.ingredients.length > 3 ? '...' : ''}
                    </p>

                    <div class="recipe-actions">
```

```
                        <button class="action-btn view-btn"
onclick="app.viewRecipe(${recipe.id})">
                            <i class="fas fa-eye"></i> View
                        </button>
                        <button class="action-btn edit-btn"
onclick="app.editRecipe(${recipe.id})">
                            <i class="fas fa-edit"></i> Edit
                        </button>
                        <button class="action-btn delete-btn"
onclick="app.deleteRecipe(${recipe.id})">
                            <i class="fas fa-trash"></i> Delete
                        </button>
                    </div>
                </div>
            </div>
        `;
    }

    bindCardEvents() {
        const cards =
document.querySelectorAll('.recipe-card');
        cards.forEach(card => {
            card.addEventListener('mouseenter', () => {
                card.style.transform = 'translateY(-10px)';
            });

            card.addEventListener('mouseleave', () => {
                card.style.transform = 'translateY(0)';
            });
        });
    }

    viewRecipe(id) {
        const recipe = this.recipes.find(r => r.id === id);
        if (!recipe) return;
```

```
        const defaultImage =
"https://images.pexels.com/photos/1640777/pexels-photo-1640777
.jpeg";
        const imageUrl = recipe.imageUrl &&
recipe.imageUrl.trim() ? recipe.imageUrl : defaultImage;


document.getElementById('recipeDetailContent').innerHTML = `
            <div class="recipe-detail-header">
                <img src="${imageUrl}" alt="${recipe.name}"
class="recipe-detail-image"
                     onerror="this.src='${defaultImage}'">
                <h2
class="recipe-detail-title">${recipe.name}</h2>
                <div class="recipe-detail-meta">
                    <div class="detail-meta-item">
                        <div class="icon"><i class="fas
fa-clock"></i></div>
                        <div class="label">Cooking Time</div>
                        <div
class="value">${recipe.cookingTime} min</div>
                    </div>
                    <div class="detail-meta-item">
                        <div class="icon"><i class="fas
fa-users"></i></div>
                        <div class="label">Servings</div>
                        <div
class="value">${recipe.servings}</div>
                    </div>
                    <div class="detail-meta-item">
                        <div class="icon"><i class="fas
fa-signal"></i></div>
                        <div class="label">Difficulty</div>
                        <div class="value difficulty
${recipe.difficulty}">${recipe.difficulty}</div>
                    </div>
                    <div class="detail-meta-item">
```

```
                        <div class="icon"><i class="fas
fa-tag"></i></div>
                        <div class="label">Category</div>
                        <div
class="value">${recipe.category}</div>
                    </div>
                </div>
            </div>

            <div class="recipe-section">
                <h3 class="section-title">
                    <i class="fas fa-list"></i>
                    Ingredients
                </h3>
                <ul class="ingredients-list">
                    ${recipe.ingredients.map(ingredient =>
`<li>${ingredient}</li>`).join('')}
                </ul>
            </div>

            <div class="recipe-section">
                <h3 class="section-title">
                    <i class="fas fa-clipboard-list"></i>
                    Instructions
                </h3>
                <ol class="instructions-list">

${recipe.instructions.split('\n').filter(step =>
step.trim()).map(step => `<li>${step.trim()}</li>`).join('')}
                </ol>
            </div>
        `;

        this.recipeDetailModal.classList.add('show');
        document.body.style.overflow = 'hidden';
    }
```

```
    editRecipe(id) {
        const recipe = this.recipes.find(r => r.id === id);
        if (recipe) {
            this.openModal(recipe);
        }
    }

    deleteRecipe(id) {
        if (confirm('Are you sure you want to delete this
recipe?')) {
            const card =
document.querySelector(`[onclick="app.deleteRecipe(${id})"]`).
closest('.recipe-card');
            card.classList.add('removing');

            setTimeout(() => {
                this.recipes = this.recipes.filter(r => r.id
!== id);
                this.saveRecipes();
                this.renderRecipes();
                this.showNotification('Recipe deleted
successfully!', 'success');
            }, 400);
        }
    }

    showNotification(message, type = 'info') {
        const notification = document.createElement('div');
        notification.className = `notification ${type}`;
        notification.style.cssText = `
            position: fixed;
            top: 20px;
            right: 20px;
            background: ${type === 'success' ? '#10b981' :
'#3b82f6'};
            color: white;
            padding: 1rem 1.5rem;
```

```
            border-radius: 8px;
            box-shadow: 0 10px 30px rgba(0, 0, 0, 0.2);
            z-index: 10000;
            transform: translateX(400px);
            transition: transform 0.3s ease;
        `;
        notification.textContent = message;

        document.body.appendChild(notification);

        setTimeout(() => {
            notification.style.transform = 'translateX(0)';
        }, 100);

        setTimeout(() => {
            notification.style.transform = 'translateX(400px)';
            setTimeout(() => {
                document.body.removeChild(notification);
            }, 300);
        }, 3000);
    }
}

// Initialize app when DOM is loaded
document.addEventListener('DOMContentLoaded', () => {
    window.app = new RecipeApp();
});

// Service Worker for offline functionality (optional)
if ('serviceWorker' in navigator) {
    window.addEventListener('load', () => {
        navigator.serviceWorker.register('/sw.js')
            .then(() => console.log('SW registered'))
            .catch(() => console.log('SW registration
failed'));
    });
}
```