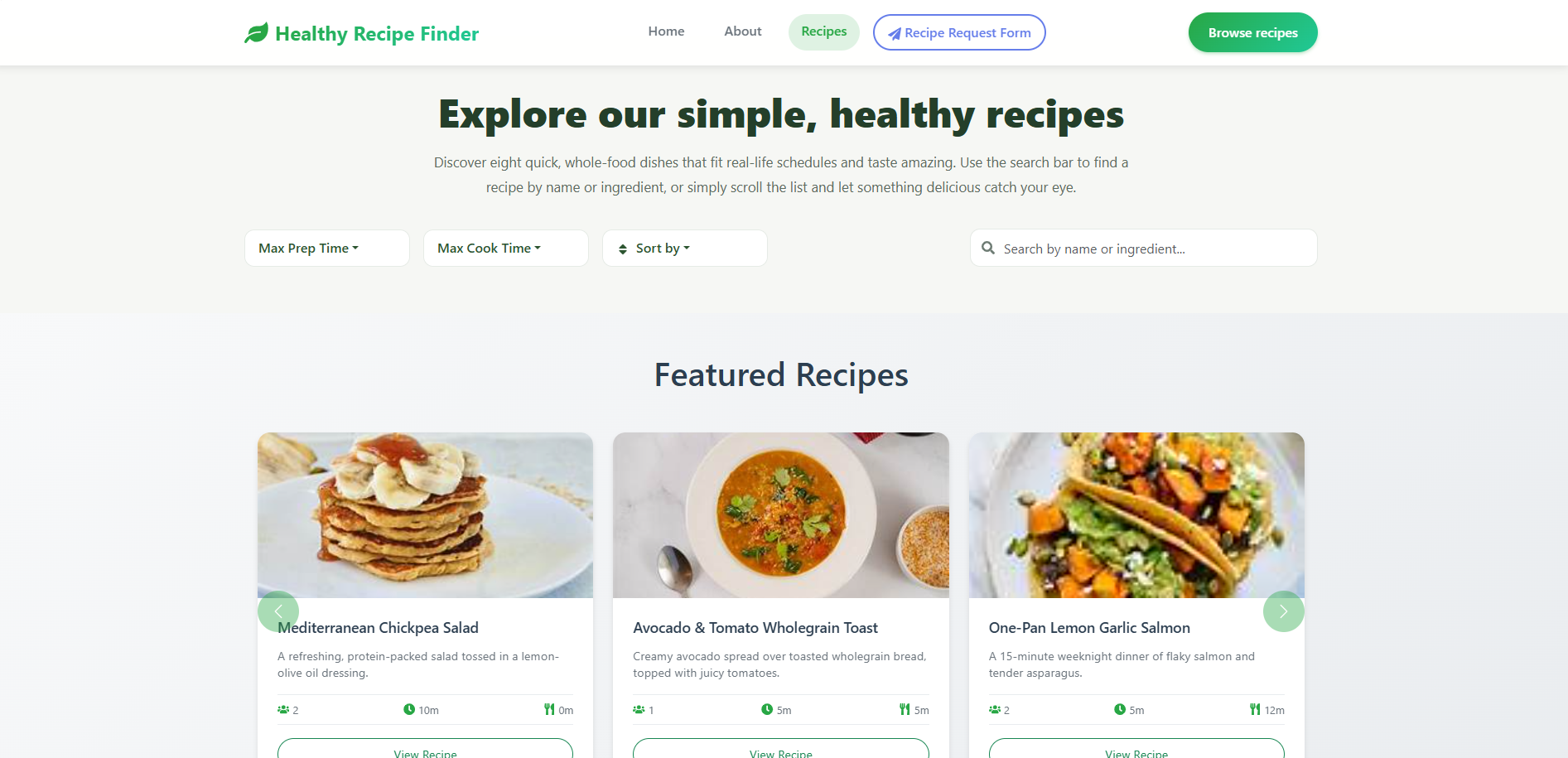
**LAB 1**

1. Carousels:

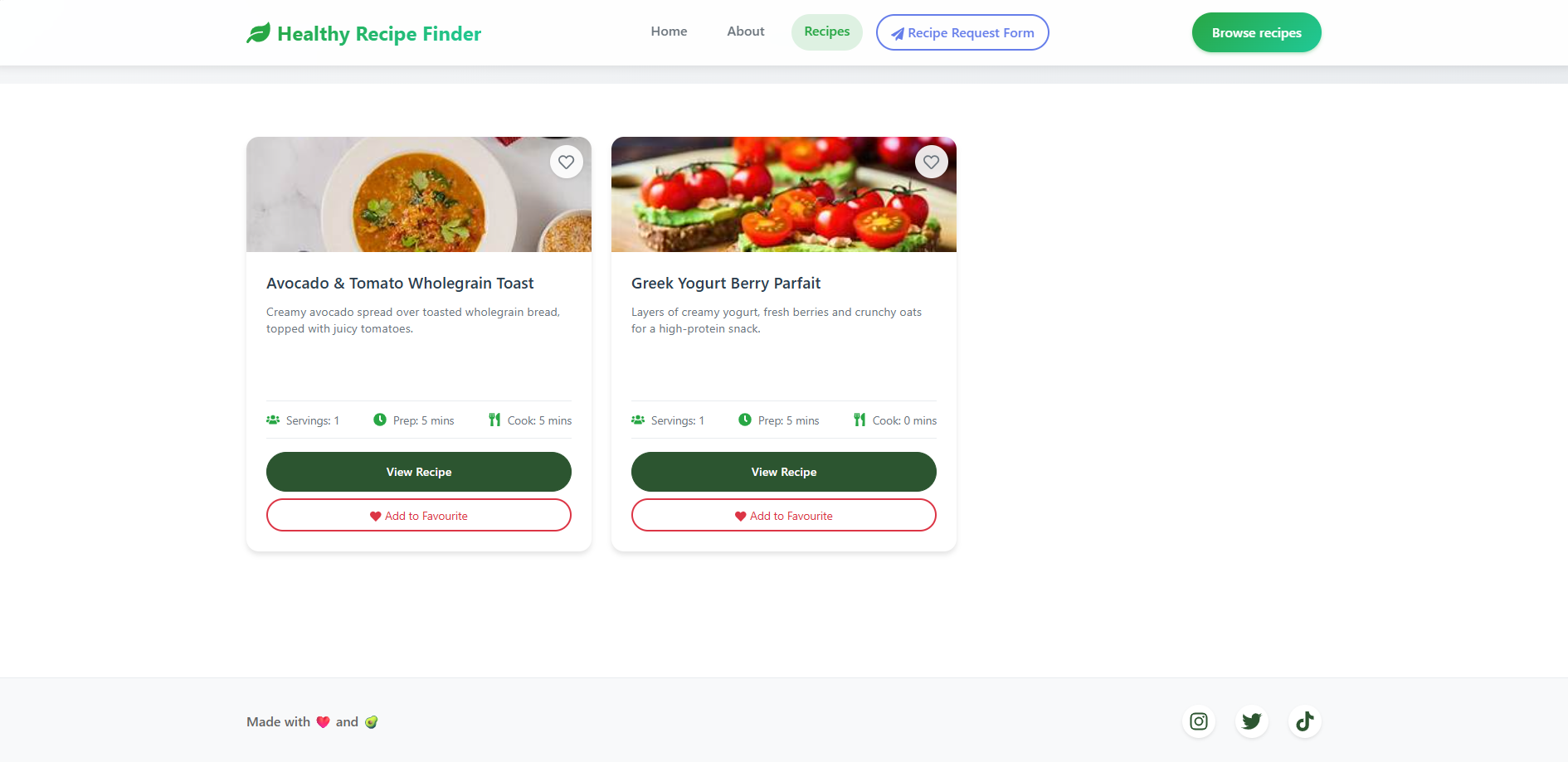
* Kết qủa:



* Code chương trình:
* import React from 'react';
* import { Carousel, Card, Button } from 'react-bootstrap';
* import { FaUsers, FaClock, FaUtensils } from 'react-icons/fa';
* import './RecipeCarousel.css';
* const RecipeCarousel = ({ *recipes*, *onViewRecipe* }) => {
* *// Chia recipes thành các nhóm 3 để hiển thị trong carousel*
* const groupedRecipes = [];
* for (let i = 0; i < recipes.length; i += 3) {
* groupedRecipes.push(recipes.slice(i, i + 3));
* }
* return (
* <section *className*="recipe-carousel-section">
* <div *className*="container">
* <h2 *className*="carousel-title">Featured Recipes</h2>
* <Carousel *interval*={5000} *className*="recipe-carousel">
* {groupedRecipes.map((*group*, *groupIndex*) => (
* <Carousel.Item *key*={groupIndex}>
* <div *className*="carousel-content">
* <div *className*="row">
* {group.map((*recipe*, *index*) => (
* <div *key*={index} *className*="col-md-4">
* <Card *className*="carousel-recipe-card">
* <div *className*="recipe-image-container">
* <Card.Img
* *variant*="top"
* *src*={*recipe*.image}
* *alt*={*recipe*.title}
* *className*="carousel-recipe-image"
* />
* </div>
* <Card.Body *className*="carousel-recipe-card-body">
* <Card.Title *className*="carousel-recipe-title">
* {*recipe*.title}
* </Card.Title>
* <Card.Text *className*="carousel-recipe-description">
* {*recipe*.description}
* </Card.Text>
* <div *className*="carousel-recipe-details">
* <div *className*="detail-item">
* <FaUsers *className*="detail-icon" />
* <span>{*recipe*.servings}</span>
* </div>
* <div *className*="detail-item">
* <FaClock *className*="detail-icon" />
* <span>{*recipe*.prep}m</span>
* </div>
* <div *className*="detail-item">
* <FaUtensils *className*="detail-icon" />
* <span>{*recipe*.cook}m</span>
* </div>
* </div>
* <Button
* *variant*="outline-success"
* *size*="sm"
* *className*="carousel-view-btn"
* *onClick*={() => onViewRecipe(*recipe*)}
* >
* View Recipe
* </Button>
* </Card.Body>
* </Card>
* </div>
* ))}
* </div>
* </div>
* </Carousel.Item>
* ))}
* </Carousel>
* </div>
* </section>
* );
* };
* export default RecipeCarousel;

1. Sort:

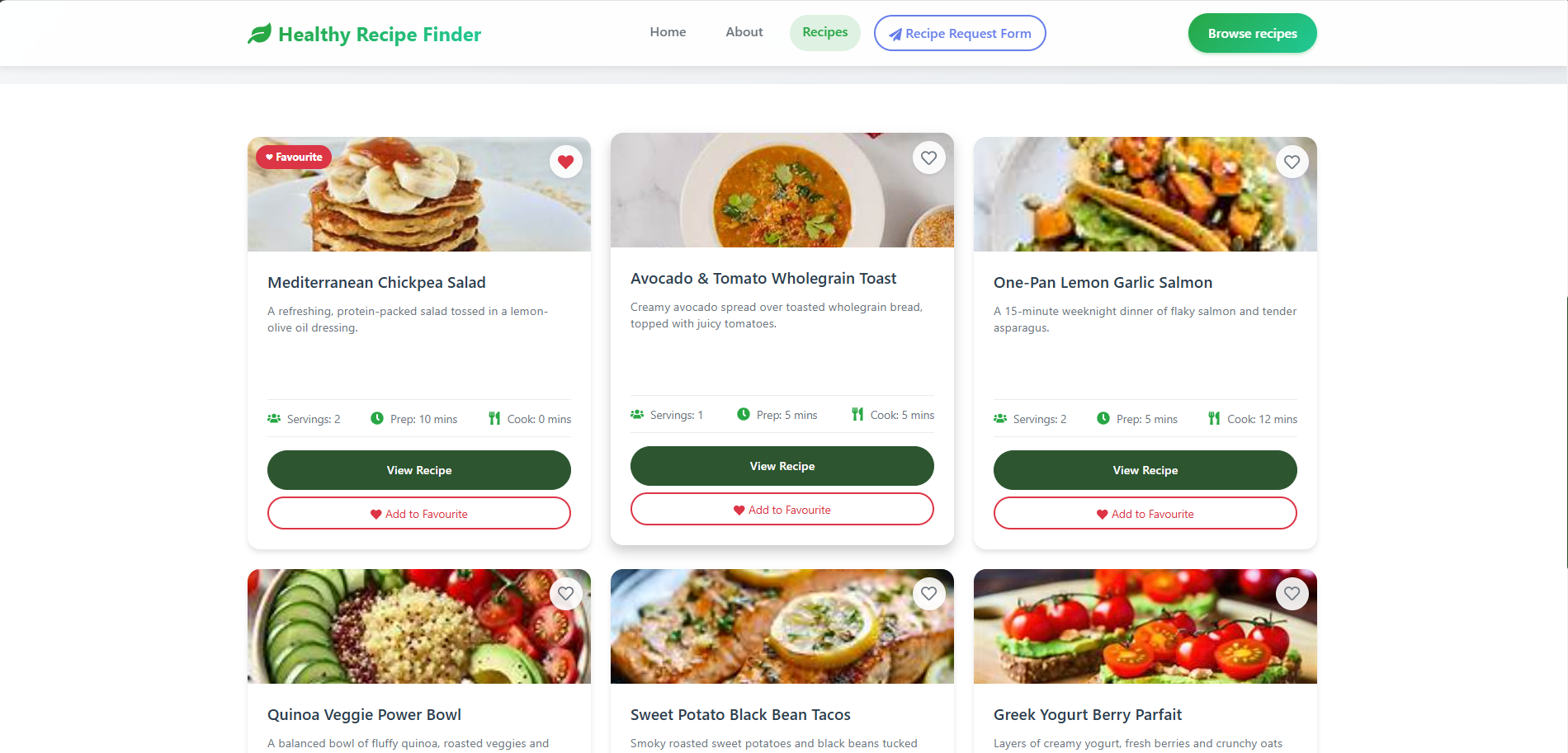
* Kết quả:



* Code chương trình:
* import React, { useEffect, useState } from 'react';
* import { Container, Row, Col, Form, Dropdown } from 'react-bootstrap';
* import { FaSearch, FaSort } from 'react-icons/fa';
* import './Hero.css';
* const Hero = ({ *onSearch*, *onFilterChange*, *onSortChange* }) => {
* const [searchTerm, setSearchTerm] = useState('');
* const [maxPrepTime, setMaxPrepTime] = useState('Max Prep Time');
* const [maxCookTime, setMaxCookTime] = useState('Max Cook Time');
* const [sortBy, setSortBy] = useState('Sort by');
* useEffect(() => {
* const handle = setTimeout(() => {
* onSearch(searchTerm);
* }, 300);
* return () => clearTimeout(handle);
* }, [searchTerm, onSearch]);
* const handlePrepTimeChange = (*time*) => {
* setMaxPrepTime(time);
* onFilterChange({ prep: time, cook: maxCookTime });
* };
* const handleCookTimeChange = (*time*) => {
* setMaxCookTime(time);
* onFilterChange({ prep: maxPrepTime, cook: time });
* };
* const handleSortChange = (*sortOption*) => {
* setSortBy(sortOption);
* onSortChange(sortOption);
* };
* return (
* <section *className*="hero-section">
* <Container>
* <Row *className*="justify-content-center text-center">
* <Col *lg*={10}>
* <h1 *className*="hero-title">
* Explore our simple, healthy recipes
* </h1>
* <p *className*="hero-description">
* Discover eight quick, whole-food dishes that fit real-life schedules and taste amazing.
* Use the search bar to find a recipe by name or ingredient, or simply scroll the list and
* let something delicious catch your eye.
* </p>
* </Col>
* </Row>
* <Row *className*="align-items-center g-3 hero-controls">
* <Col *xs*={12} *md*="auto" >
* <Dropdown>
* <Dropdown.Toggle *variant*="light" *className*="filter-dropdown">
* {maxPrepTime}
* </Dropdown.Toggle>
* <Dropdown.Menu>
* <Dropdown.Item *onClick*={() => handlePrepTimeChange('Max Prep Time')}>Any</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handlePrepTimeChange('5 mins')}>5 mins</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handlePrepTimeChange('10 mins')}>10 mins</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handlePrepTimeChange('15 mins')}>15 mins</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handlePrepTimeChange('20 mins')}>20 mins</Dropdown.Item>
* </Dropdown.Menu>
* </Dropdown>
* </Col>
* <Col *xs*={12} *md*="auto">
* <Dropdown>
* <Dropdown.Toggle *variant*="light" *className*="filter-dropdown">
* {maxCookTime}
* </Dropdown.Toggle>
* <Dropdown.Menu>
* <Dropdown.Item *onClick*={() => handleCookTimeChange('Max Cook Time')}>Any</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleCookTimeChange('5 mins')}>5 mins</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleCookTimeChange('10 mins')}>10 mins</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleCookTimeChange('15 mins')}>15 mins</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleCookTimeChange('20 mins')}>20 mins</Dropdown.Item>
* </Dropdown.Menu>
* </Dropdown>
* </Col>
* <Col *xs*={12} *md*="auto">
* <Dropdown>
* <Dropdown.Toggle *variant*="light" *className*="filter-dropdown">
* <FaSort *className*="me-2" />
* {sortBy}
* </Dropdown.Toggle>
* <Dropdown.Menu>
* <Dropdown.Item *onClick*={() => handleSortChange('Name A→Z')}>Name A→Z</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleSortChange('Name Z→A')}>Name Z→A</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleSortChange('Prep ↑')}>Prep ↑</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleSortChange('Prep ↓')}>Prep ↓</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleSortChange('Cook ↑')}>Cook ↑</Dropdown.Item>
* <Dropdown.Item *onClick*={() => handleSortChange('Cook ↓')}>Cook ↓</Dropdown.Item>
* </Dropdown.Menu>
* </Dropdown>
* </Col>
* <Col *xs*={12} *md* *className*="d-flex justify-content-md-end">
* <Form *className*="search-form">
* <div *className*="search-input-container">
* <FaSearch *className*="search-icon" />
* <Form.Control
* *type*="text"
* *placeholder*="Search by name or ingredient..."
* *value*={searchTerm}
* *onChange*={(*e*) => setSearchTerm(*e*.target.value)}
* *className*="search-input"
* />
* </div>
* </Form>
* </Col>
* </Row>
* </Container>
* </section>
* );
* };
* export default Hero;

1. Add to Favourite:

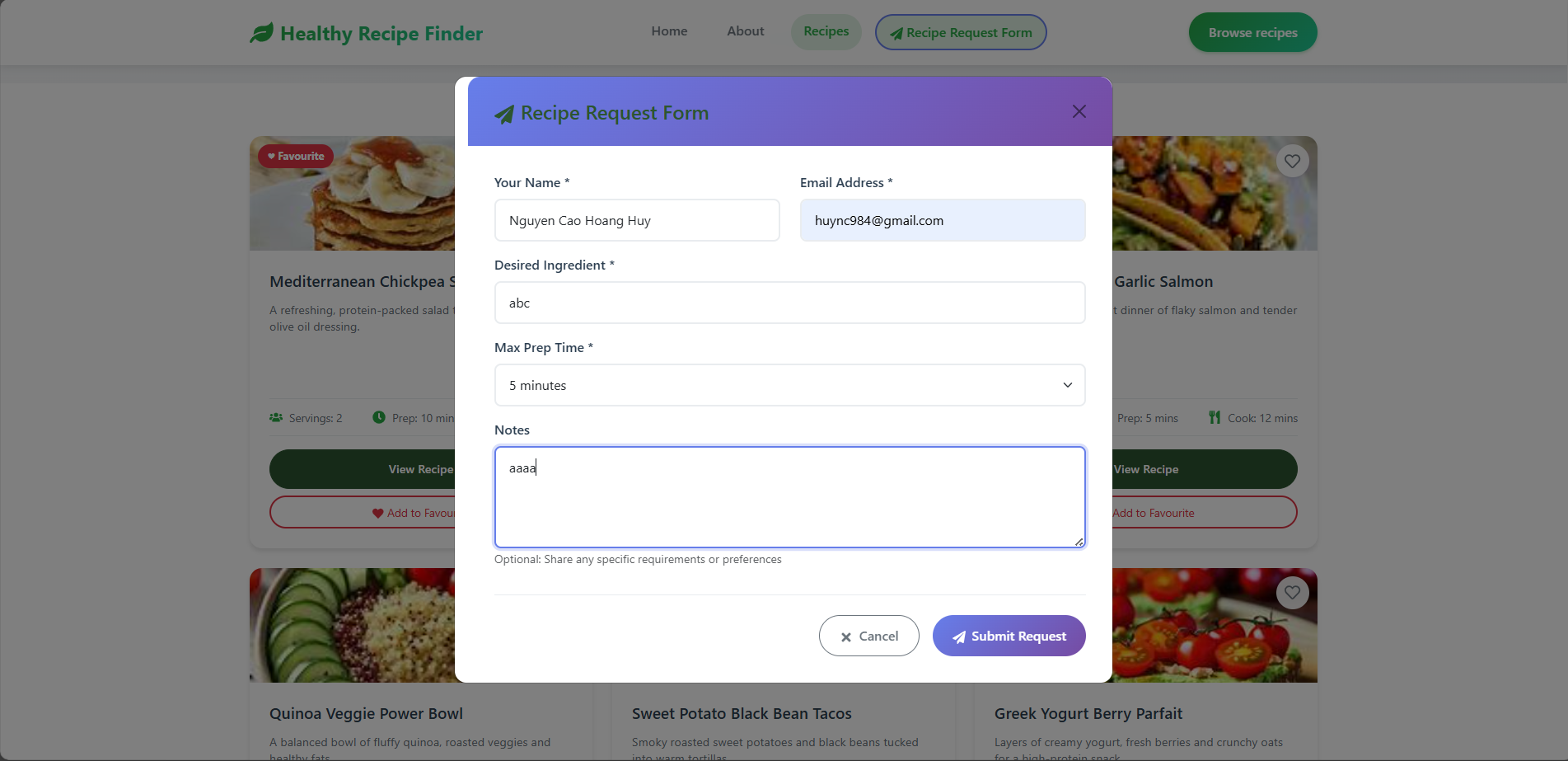
* Kết quả:



* Code chương trình:
* import React from 'react';
* import { Card, Button, Badge } from 'react-bootstrap';
* import { FaUsers, FaClock, FaUtensils, FaHeart, FaRegHeart } from 'react-icons/fa';
* import './RecipeCard.css';
* const **RecipeCard** = ({ *recipe*, *onViewRecipe*, *onAddToFavourite*, *isFavourite* }) => {
* return (
* <Card *className*="recipe-card">
* <div *className*="recipe-image-container">
* <Card.Img
* *variant*="top"
* *src*={*recipe*.image}
* *alt*={*recipe*.title}
* *className*="recipe-image"
* *loading*="lazy"
* />
* <div *className*="favourite-overlay">
* <Button
* *variant*="light"
* *size*="sm"
* *className*="favourite-btn"
* *onClick*={(*e*) => {
* *e*.stopPropagation();
* onAddToFavourite(*recipe*);
* }}
* >
* {*isFavourite* ? (
* <FaHeart *className*="heart-icon filled" />
* ) : (
* <FaRegHeart *className*="heart-icon" />
* )}
* </Button>
* </div>
* {*isFavourite* && (
* <Badge *bg*="danger" *className*="favourite-badge">
* ♥ Favourite
* </Badge>
* )}
* </div>
* <Card.Body *className*="recipe-card-body">
* <Card.Title *className*="recipe-title">{*recipe*.title}</Card.Title>
* <Card.Text *className*="recipe-description">
* {*recipe*.description}
* </Card.Text>
* <div *className*="recipe-details">
* <div *className*="detail-item">
* <FaUsers *className*="detail-icon" />
* <span>Servings: {*recipe*.servings}</span>
* </div>
* <div *className*="detail-item">
* <FaClock *className*="detail-icon" />
* <span>Prep: {*recipe*.prep} mins</span>
* </div>
* <div *className*="detail-item">
* <FaUtensils *className*="detail-icon" />
* <span>Cook: {*recipe*.cook} mins</span>
* </div>
* </div>
* <div *className*="recipe-actions">
* <Button
* *variant*="success"
* *className*="view-recipe-btn"
* *onClick*={() => onViewRecipe(*recipe*)}
* >
* View Recipe
* </Button>
* <Button
* *variant*="outline-danger"
* *size*="sm"
* *className*="add-favourite-btn"
* *onClick*={() => onAddToFavourite(*recipe*)}
* >
* <FaHeart *className*="me-1" />
* Add to Favourite
* </Button>
* </div>
* </Card.Body>
* </Card>
* );
* };
* export default RecipeCard;

1. Forms

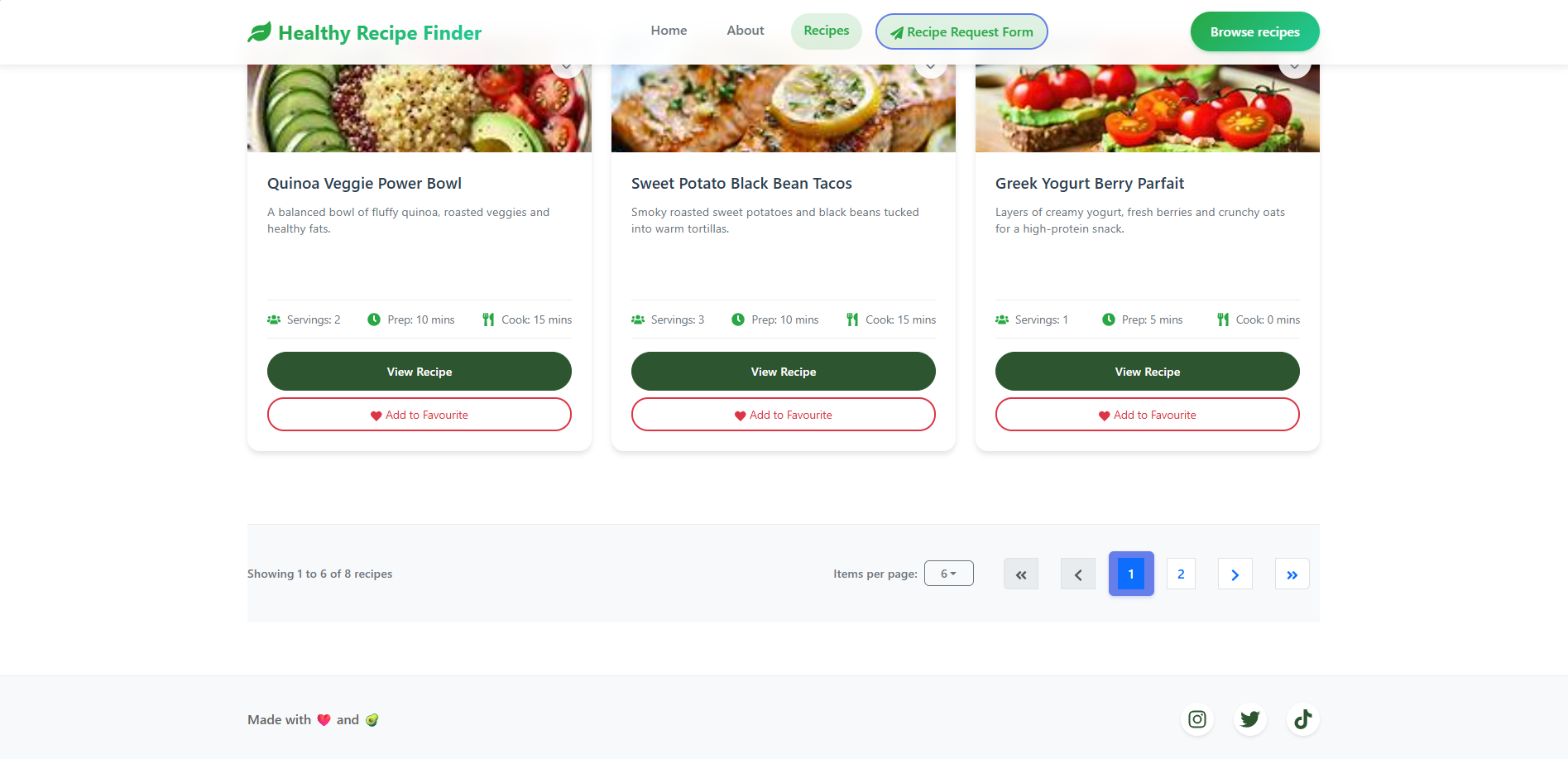
* Kết quả:



* Code chương trình:
* import React, { useState } from 'react';
* import { Modal, Form, Button, Row, Col } from 'react-bootstrap';
* import { FaPaperPlane, FaTimes } from 'react-icons/fa';
* import './RecipeRequestForm.css';
* const RecipeRequestForm = ({ *show*, *onHide* }) => {
* const [formData, setFormData] = useState({
* name: '',
* email: '',
* ingredient: '',
* prepTime: '',
* notes: ''
* });
* const [showFeedback, setShowFeedback] = useState({
* name: false,
* email: false,
* ingredient: false,
* prepTime: false,
* notes: false
* });
* const handleInputChange = (*field*, *value*) => {
* setFormData(*prev* => ({
* ...prev,
* [field]: value
* }));
* *// Show feedback when user starts typing*
* if (value.trim() === '') {
* setShowFeedback(*prev* => ({
* ...prev,
* [field]: false
* }));
* } else {
* setShowFeedback(*prev* => ({
* ...prev,
* [field]: true
* }));
* }
* };
* const handleSubmit = (*e*) => {
* e.preventDefault();
* console.log('Form submitted:', formData);
* *// Here you would typically send the data to a server*
* alert('Recipe request submitted successfully!');
* onHide();
* };
* const handleClose = () => {
* setFormData({
* name: '',
* email: '',
* ingredient: '',
* prepTime: '',
* notes: ''
* });
* setShowFeedback({
* name: false,
* email: false,
* ingredient: false,
* prepTime: false,
* notes: false
* });
* onHide();
* };
* return (
* <Modal *show*={show} *onHide*={handleClose} *size*="lg" *centered* *className*="recipe-request-modal">
* <Modal.Header *closeButton* *className*="modal-header">
* <Modal.Title *className*="modal-title">
* <FaPaperPlane *className*="me-2" />
* Recipe Request Form
* </Modal.Title>
* </Modal.Header>
* <Modal.Body *className*="modal-body">
* <Form *onSubmit*={handleSubmit}>
* <Row>
* <Col *md*={6}>
* <Form.Group *className*="mb-3">
* <Form.Label>Your Name \*</Form.Label>
* <Form.Control
* *type*="text"
* *placeholder*="Enter your name"
* *value*={formData.name}
* *onChange*={(*e*) => handleInputChange('name', e.target.value)}
* *className*={showFeedback.name && formData.name.trim() === '' ? 'is-invalid' : ''}
* />
* {showFeedback.name && formData.name.trim() === '' && (
* <Form.Control.Feedback *type*="invalid">
* Please enter your name
* </Form.Control.Feedback>
* )}
* </Form.Group>
* </Col>
* <Col *md*={6}>
* <Form.Group *className*="mb-3">
* <Form.Label>Email Address \*</Form.Label>
* <Form.Control
* *type*="email"
* *placeholder*="Enter your email"
* *value*={formData.email}
* *onChange*={(*e*) => handleInputChange('email', e.target.value)}
* *className*={showFeedback.email && !formData.email.includes('@') ? 'is-invalid' : ''}
* />
* {showFeedback.email && !formData.email.includes('@') && (
* <Form.Control.Feedback *type*="invalid">
* Please provide a valid email
* </Form.Control.Feedback>
* )}
* </Form.Group>
* </Col>
* </Row>
* <Form.Group *className*="mb-3">
* <Form.Label>Desired Ingredient \*</Form.Label>
* <Form.Control
* *type*="text"
* *placeholder*="e.g., Fresh salmon, organic quinoa, seasonal vegetables..."
* *value*={formData.ingredient}
* *onChange*={(*e*) => handleInputChange('ingredient', e.target.value)}
* *className*={showFeedback.ingredient && formData.ingredient.trim() === '' ? 'is-invalid' : ''}
* />
* {showFeedback.ingredient && formData.ingredient.trim() === '' && (
* <Form.Control.Feedback *type*="invalid">
* Please specify your desired ingredients
* </Form.Control.Feedback>
* )}
* </Form.Group>
* <Form.Group *className*="mb-3">
* <Form.Label>Max Prep Time \*</Form.Label>
* <Form.Select
* *value*={formData.prepTime}
* *onChange*={(*e*) => handleInputChange('prepTime', e.target.value)}
* *className*={showFeedback.prepTime && formData.prepTime === '' ? 'is-invalid' : ''}
* >
* <option *value*="">Select prep time</option>
* <option *value*="5">5 minutes</option>
* <option *value*="10">10 minutes</option>
* <option *value*="15">15 minutes</option>
* <option *value*="30">30 minutes</option>
* </Form.Select>
* {showFeedback.prepTime && formData.prepTime === '' && (
* <Form.Control.Feedback *type*="invalid">
* Please select a prep time
* </Form.Control.Feedback>
* )}
* </Form.Group>
* <Form.Group *className*="mb-3">
* <Form.Label>Notes</Form.Label>
* <Form.Control
* *as*="textarea"
* *rows*={4}
* *placeholder*="Any additional notes, dietary restrictions, or preferences..."
* *value*={formData.notes}
* *onChange*={(*e*) => handleInputChange('notes', e.target.value)}
* />
* <Form.Text *className*="text-muted">
* Optional: Share any specific requirements or preferences
* </Form.Text>
* </Form.Group>
* <div *className*="form-actions">
* <Button
* *variant*="outline-secondary"
* *onClick*={handleClose}
* *className*="cancel-btn"
* >
* <FaTimes *className*="me-2" />
* Cancel
* </Button>
* <Button
* *variant*="primary"
* *type*="submit"
* *className*="submit-btn"
* >
* <FaPaperPlane *className*="me-2" />
* Submit Request
* </Button>
* </div>
* </Form>
* </Modal.Body>
* </Modal>
* );
* };
* export default RecipeRequestForm;

1. Phân trang Pagination:

* Kết quả:



* Code chương trình:
* import React from 'react';
* import { Pagination, Dropdown, Row, Col } from 'react-bootstrap';
* import { FaChevronLeft, FaChevronRight, FaAngleDoubleLeft, FaAngleDoubleRight } from 'react-icons/fa';
* import './RecipePagination.css';
* const **RecipePagination** = ({
* *currentPage*,
* *totalPages*,
* *itemsPerPage*,
* *totalItems*,
* *onPageChange*,
* *onItemsPerPageChange*
* }) => {
* const **getPageNumbers** = () => {
* const pages = [];
* const maxVisiblePages = 5;
* if (*totalPages* <= maxVisiblePages) {
* for (let i = 1; i <= *totalPages*; i++) {
* pages.push(i);
* }
* } else {
* if (*currentPage* <= 3) {
* for (let i = 1; i <= 4; i++) {
* pages.push(i);
* }
* pages.push('...');
* pages.push(*totalPages*);
* } else if (*currentPage* >= *totalPages* - 2) {
* pages.push(1);
* pages.push('...');
* for (let i = *totalPages* - 3; i <= *totalPages*; i++) {
* pages.push(i);
* }
* } else {
* pages.push(1);
* pages.push('...');
* for (let i = *currentPage* - 1; i <= *currentPage* + 1; i++) {
* pages.push(i);
* }
* pages.push('...');
* pages.push(*totalPages*);
* }
* }
* return pages;
* };
* const startItem = (currentPage - 1) \* itemsPerPage + 1;
* const endItem = Math.min(currentPage \* itemsPerPage, totalItems);
* return (
* <div *className*="recipe-pagination-section">
* <Row *className*="align-items-center">
* <Col *md*={6}>
* <div *className*="pagination-info">
* Showing {startItem} to {endItem} of {totalItems} recipes
* </div>
* </Col>
* <Col *md*={6} *className*="d-flex justify-content-md-end">
* <div *className*="pagination-controls">
* <div *className*="items-per-page-container">
* <span *className*="items-label">Items per page:</span>
* <Dropdown>
* <Dropdown.Toggle *variant*="outline-secondary" *size*="sm" *className*="items-dropdown">
* {itemsPerPage}
* </Dropdown.Toggle>
* <Dropdown.Menu>
* <Dropdown.Item *onClick*={() => onItemsPerPageChange(6)}>6</Dropdown.Item>
* <Dropdown.Item *onClick*={() => onItemsPerPageChange(9)}>9</Dropdown.Item>
* <Dropdown.Item *onClick*={() => onItemsPerPageChange(12)}>12</Dropdown.Item>
* </Dropdown.Menu>
* </Dropdown>
* </div>
* <Pagination *className*="pagination-custom">
* {*/\* First Page \*/*}
* <Pagination.First
* *onClick*={() => onPageChange(1)}
* *disabled*={currentPage === 1}
* *className*="pagination-btn"
* >
* <FaAngleDoubleLeft />
* </Pagination.First>
* {*/\* Previous Page \*/*}
* <Pagination.Prev
* *onClick*={() => onPageChange(currentPage - 1)}
* *disabled*={currentPage === 1}
* *className*="pagination-btn"
* >
* <FaChevronLeft />
* </Pagination.Prev>
* {*/\* Page Numbers \*/*}
* {getPageNumbers().map((*page*, *index*) => (
* <Pagination.Item
* *key*={index}
* *active*={page === currentPage}
* *onClick*={() => typeof page === 'number' && onPageChange(page)}
* *disabled*={page === '...'}
* *className*={`pagination-btn ${page === '...' ? 'ellipsis' : ''}`}
* >
* {page}
* </Pagination.Item>
* ))}
* {*/\* Next Page \*/*}
* <Pagination.Next
* *onClick*={() => onPageChange(currentPage + 1)}
* *disabled*={currentPage === totalPages}
* *className*="pagination-btn"
* >
* <FaChevronRight />
* </Pagination.Next>
* {*/\* Last Page \*/*}
* <Pagination.Last
* *onClick*={() => onPageChange(totalPages)}
* *disabled*={currentPage === totalPages}
* *className*="pagination-btn"
* >
* <FaAngleDoubleRight />
* </Pagination.Last>
* </Pagination>
* </div>
* </Col>
* </Row>
* </div>
* );
* };
* export default RecipePagination;