**MAST POE:**

**CHANGELOG:**

1.Implement lecturer feedback:

-Fixed layout on the home page.

-Added error handling for invalid menu inputs.

2.Added the average price feature to the home screen.

3.Created a separate screen for managing menu items (add/remove)

4.Added a guest filtering feature to filter menu by course

5.Refractored code:

-Created reusable components (MenuItems, AddMenu)

Refactor Navigation Structure:

The navigation structure was updated to include several new screens: Home, ChefLogin, ChefPage, AddMenu, FilterMenu, and Menu.

App.js was updated to implement React Navigation with stack navigators, linking the various screens appropriately.

Home Screen Updates:

A new functionality to show the average price of the menu items broken down by different courses was added to the Home screen.

This allows the user (or chef) to view the pricing details at a glance.

ChefLogin Screen Added:

A new ChefLogin screen was created where the chef can log in to the app before accessing the other features. This screen handles user authentication and ensures chefs can securely access their page.

ChefPage Screen Added:

The ChefPage screen was added where chefs can manage their menu items and other actions that only chefs should have access to.

AddMenu Screen Added:

A separate AddMenu screen was created, where chefs can add new menu items and manage them. This screen is distinct from the Home screen and provides a more focused space for menu management.

FilterMenu Screen Added:

A new FilterMenu screen was created, allowing users to filter menu items by course (starter, main, dessert, etc.). This enhances the usability of the app, especially for guests browsing the menu.

Menu Screen Added:

The Menu screen was introduced to display the complete list of menu items. This screen is now dedicated to showcasing the entire menu to the guests.

Refactored Code into Multiple Files:

The code was organized into separate files to improve maintainability. Each screen now exists in its own file, reducing clutter in App.js and making the codebase easier to manage.

Added Loops for Better Data Handling:

For Loops: Used for iterating over menu items to calculate averages and filter courses.

While Loop: Applied for removing items from the menu based on a price threshold.

Improved Code Organization:

Each feature is split into a dedicated screen component to improve readability and maintainability.

Functions are used to encapsulate reusable logic, making the app scalable.

Added Features:

Averages calculation and display on the Home screen.

Filtering menu items by course on the FilterMenu screen.

Dynamic menu item removal with a while loop on the AddMenu screen.

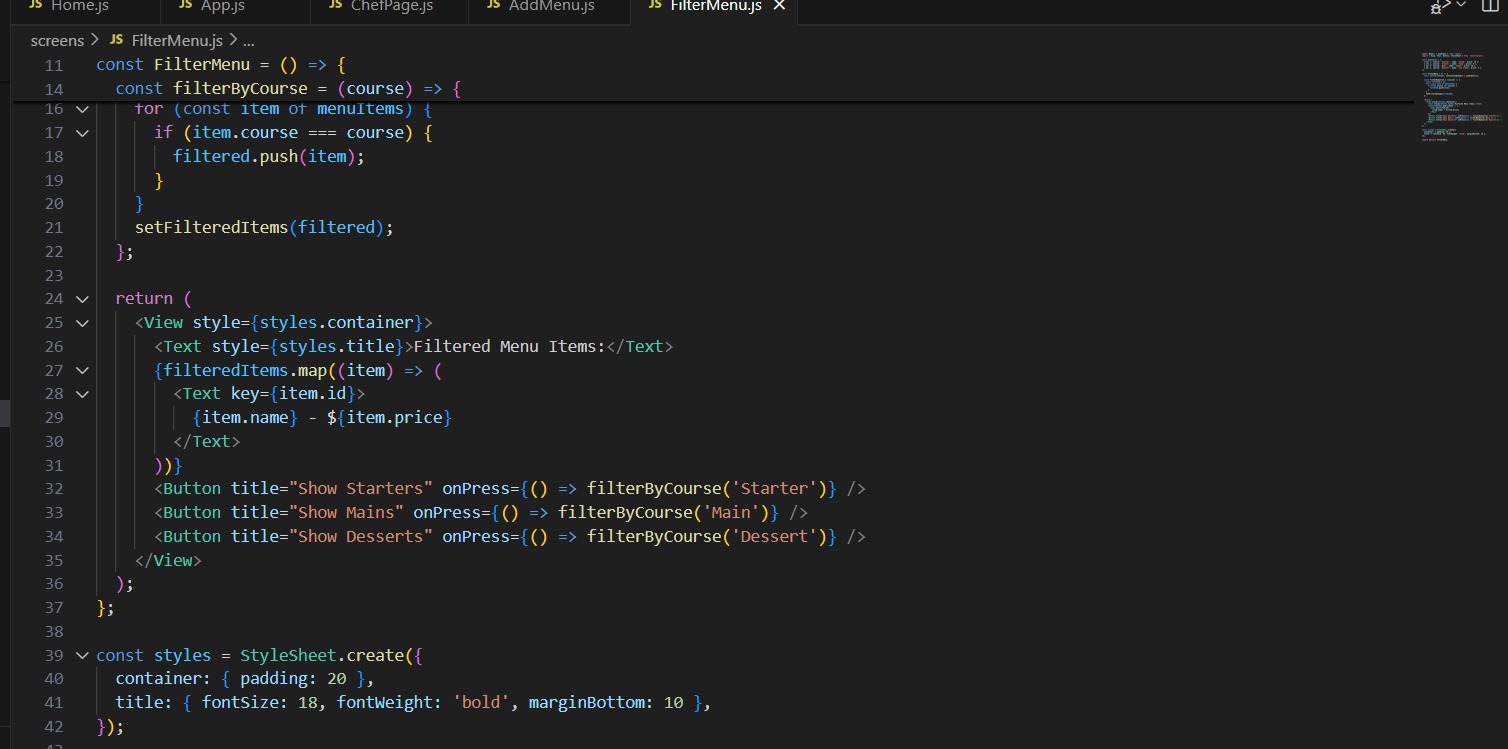
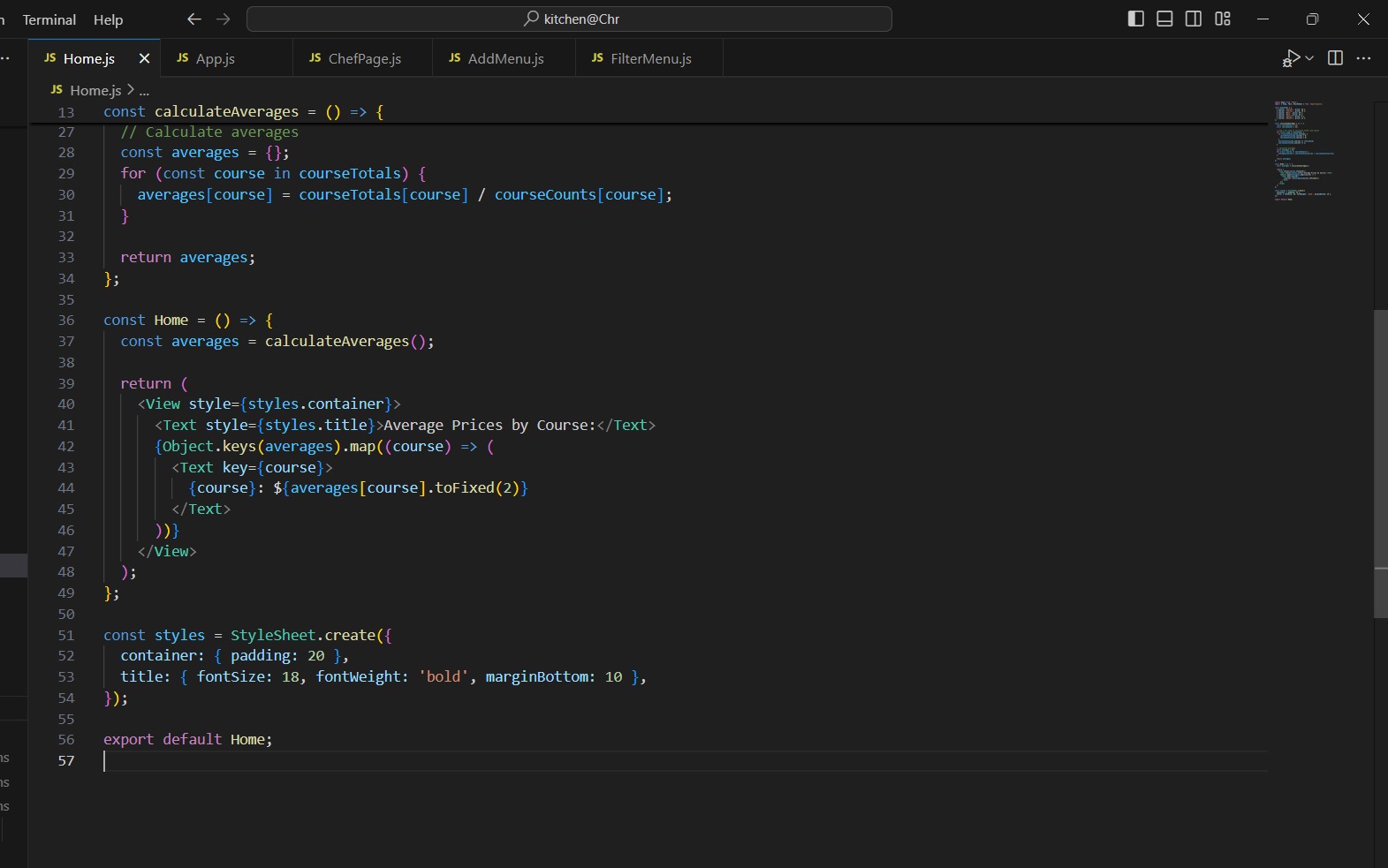
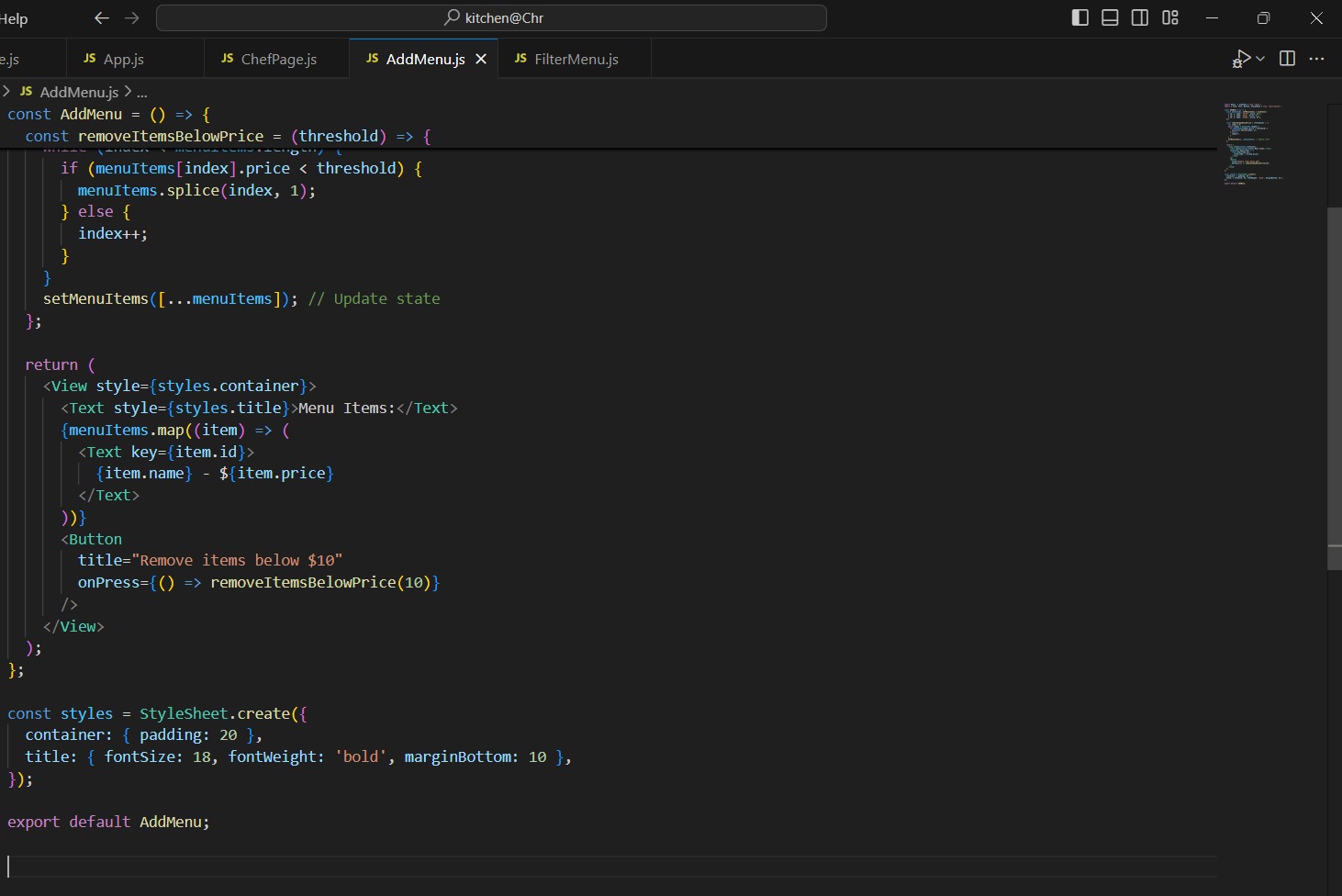
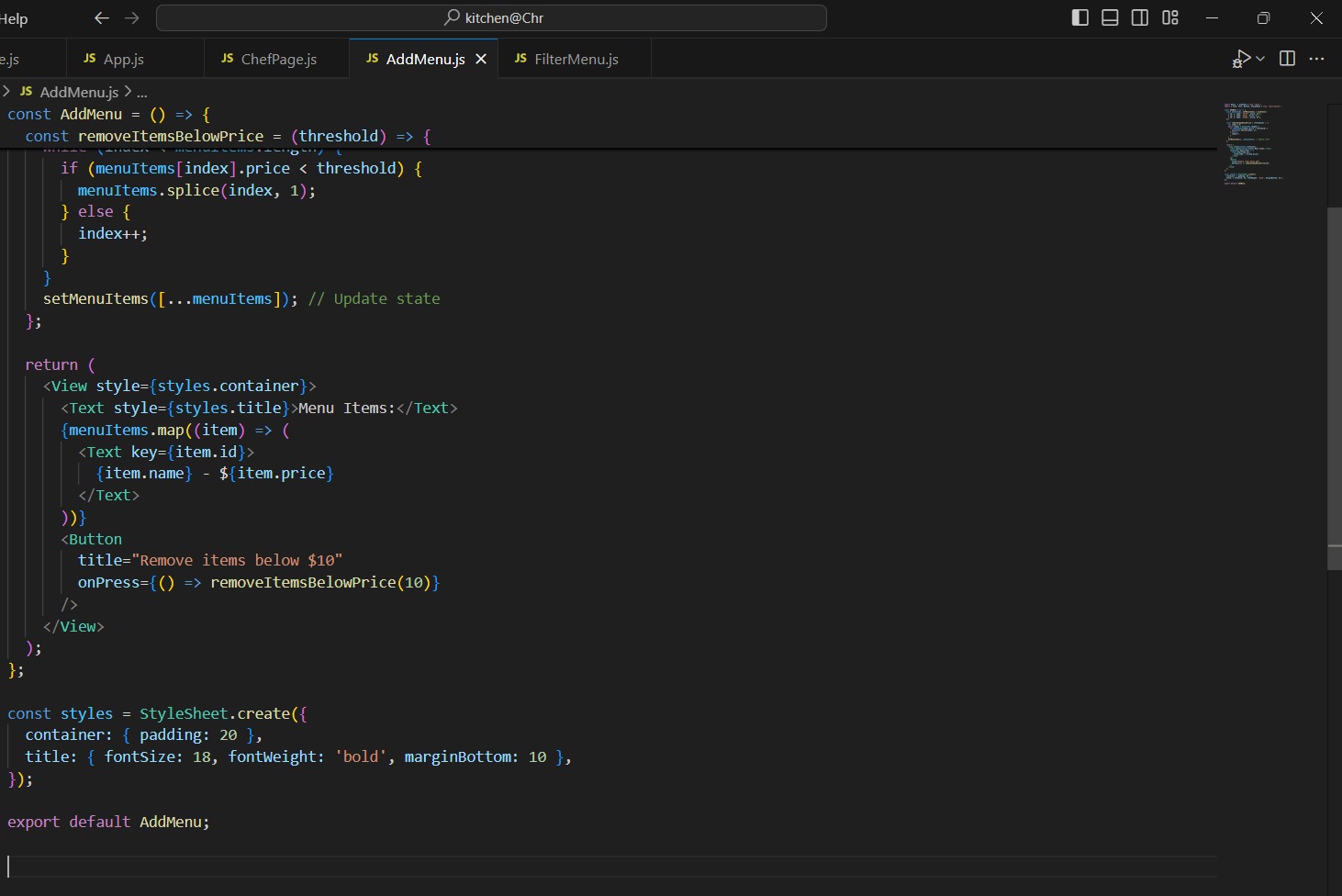
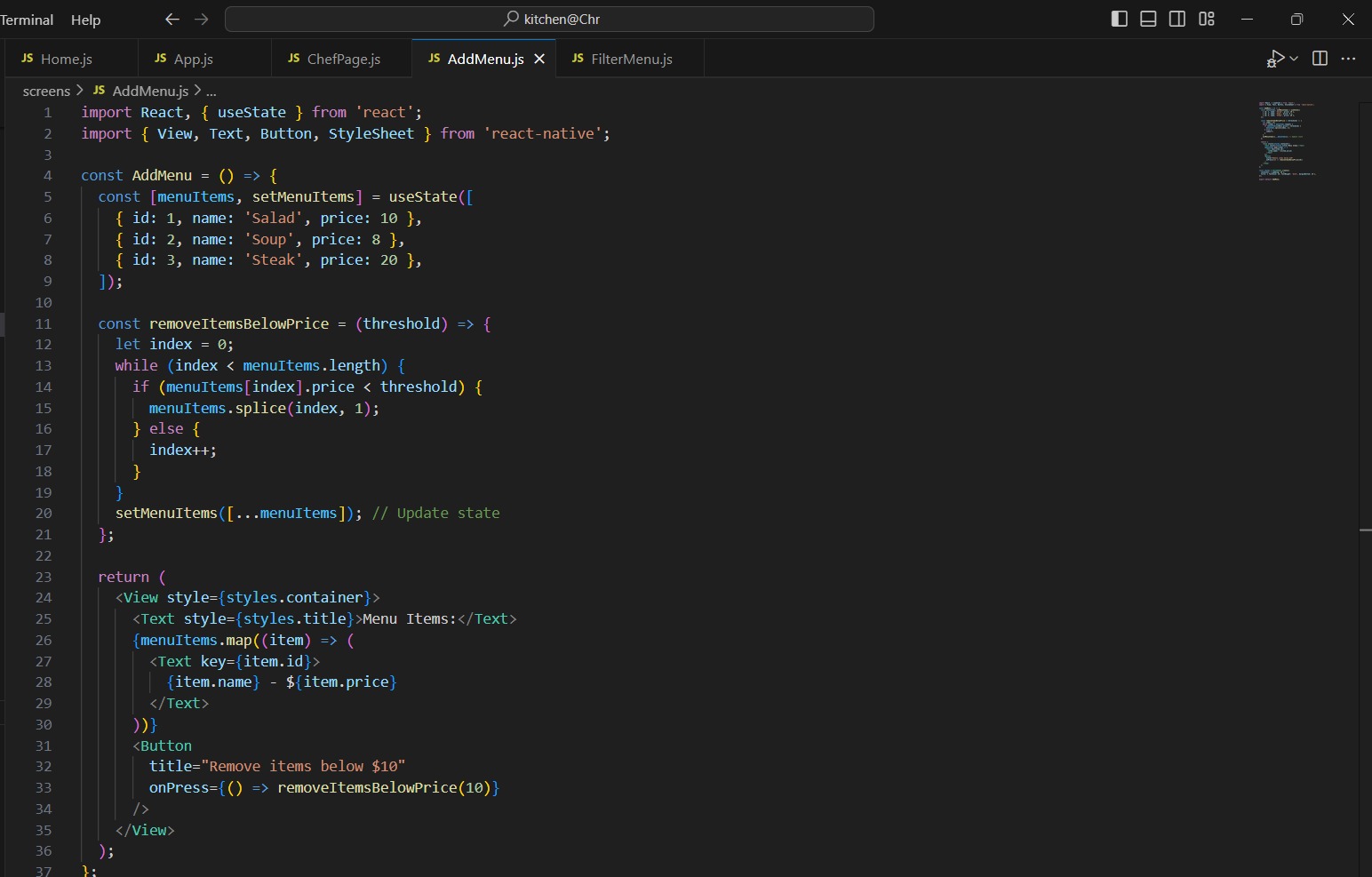
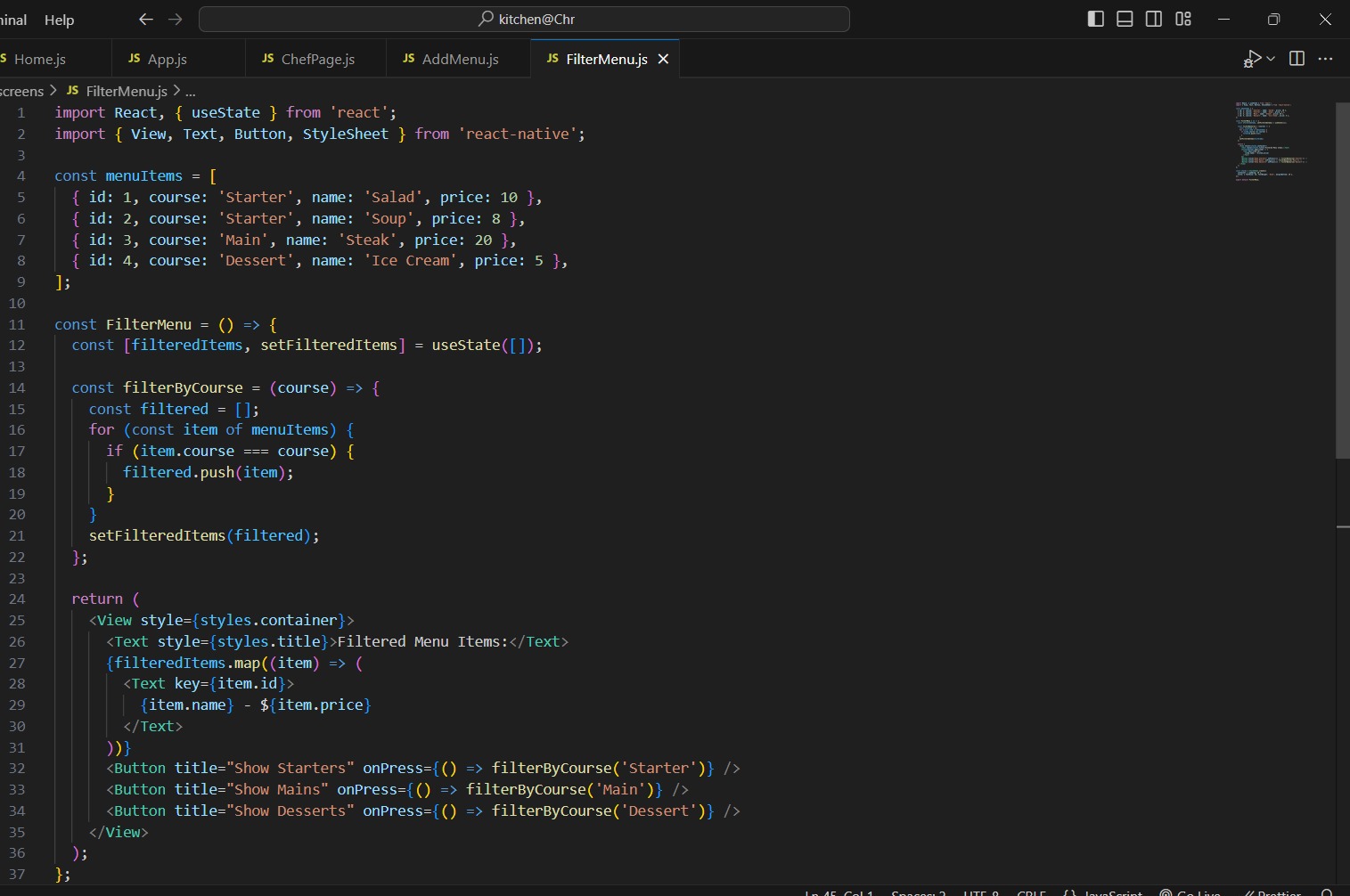
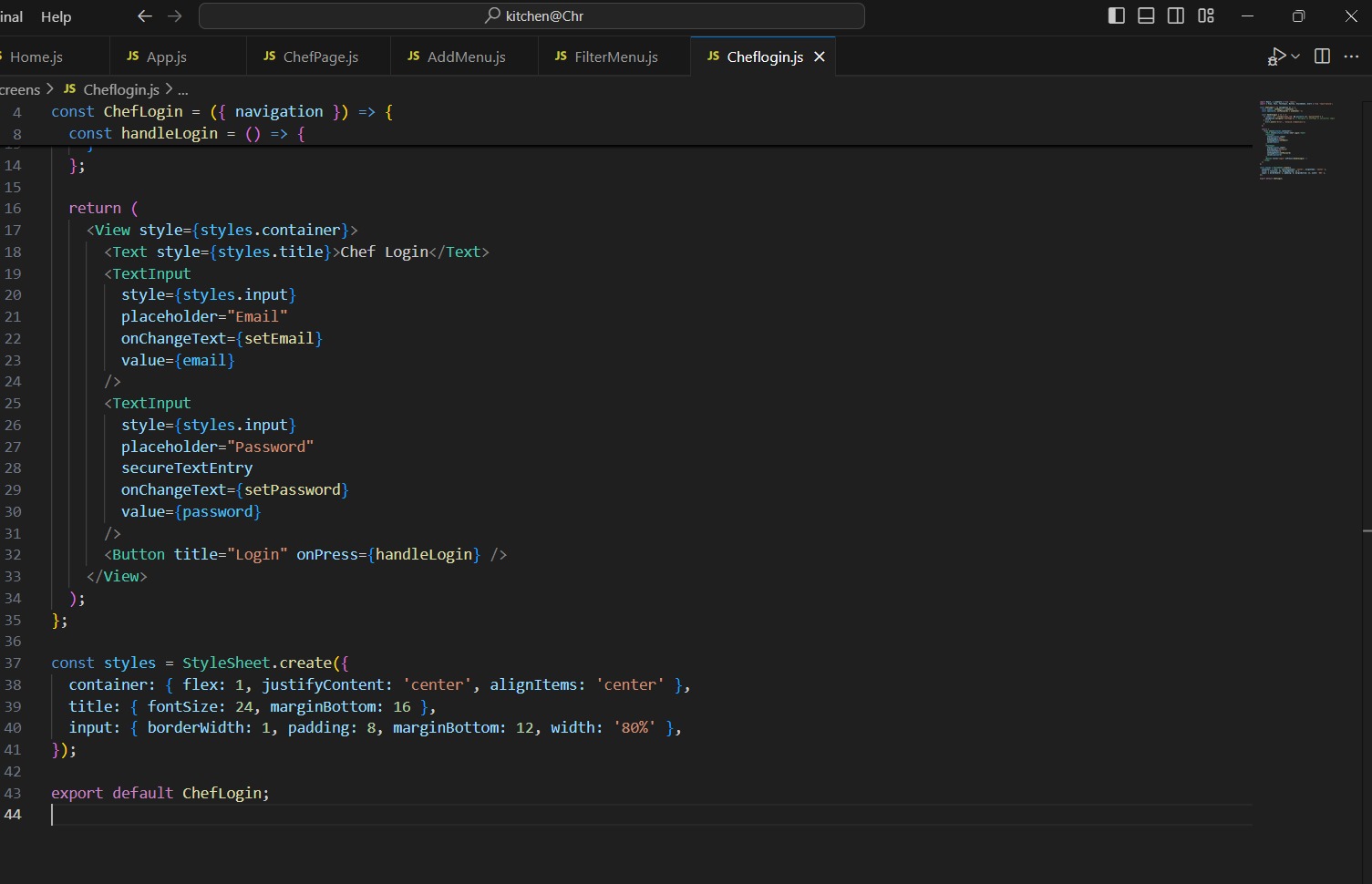
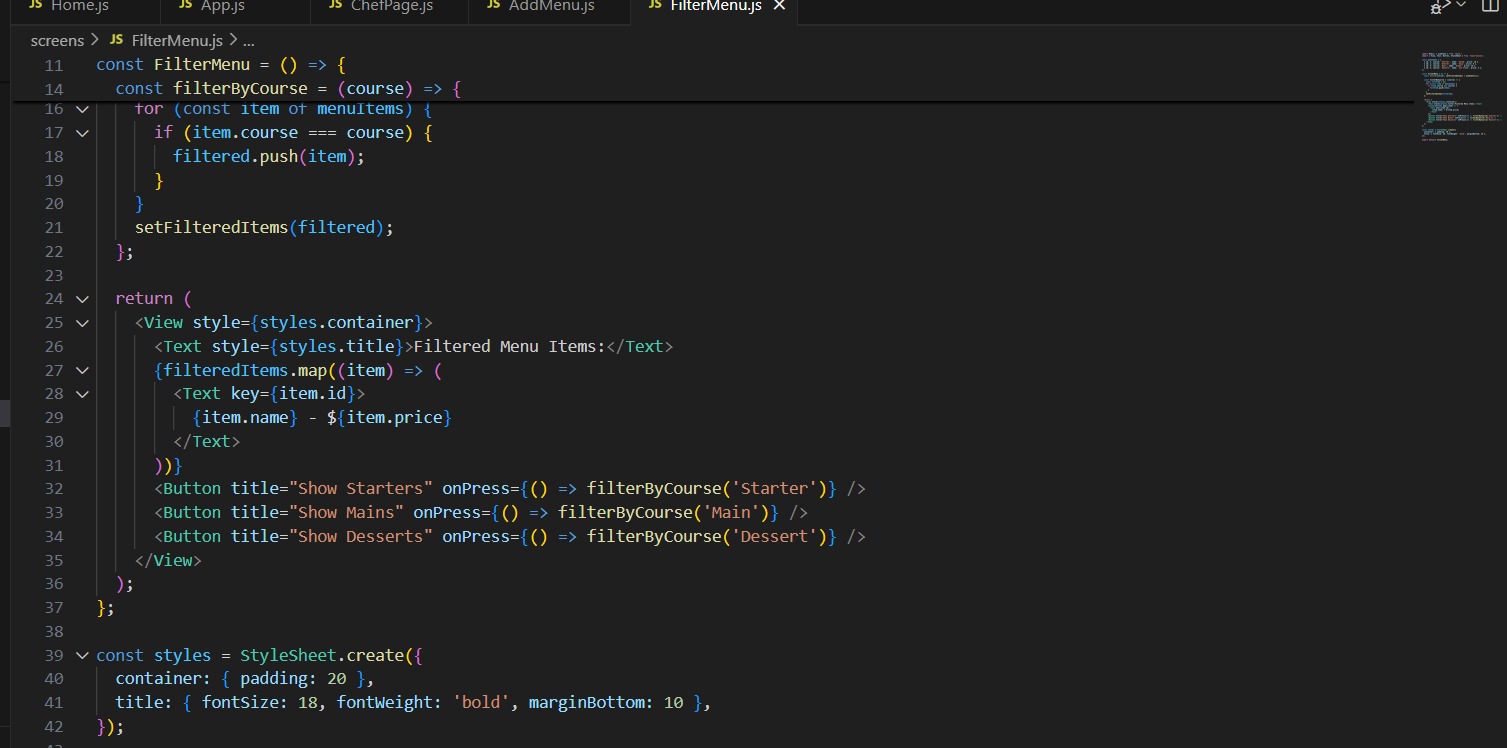
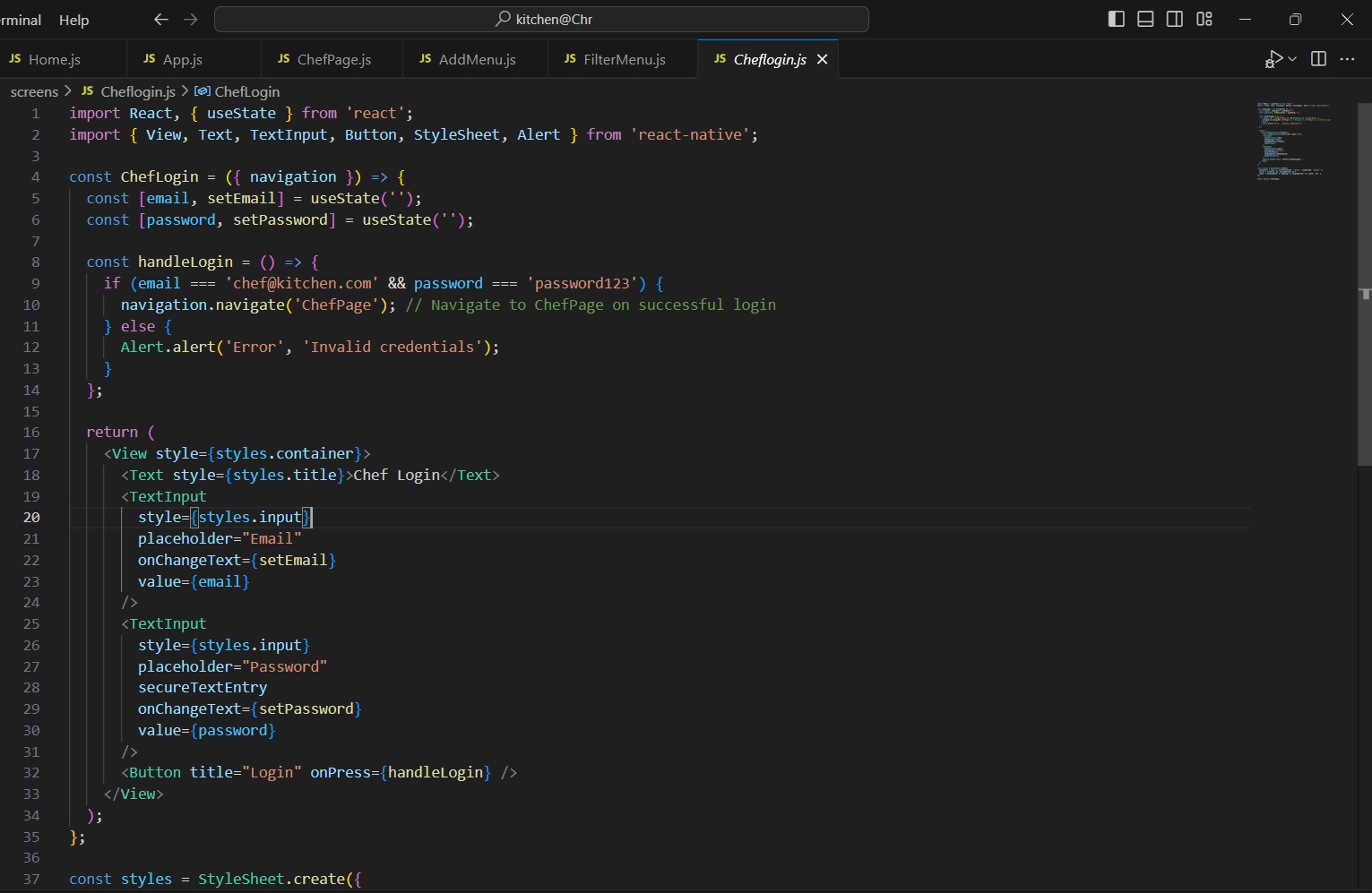
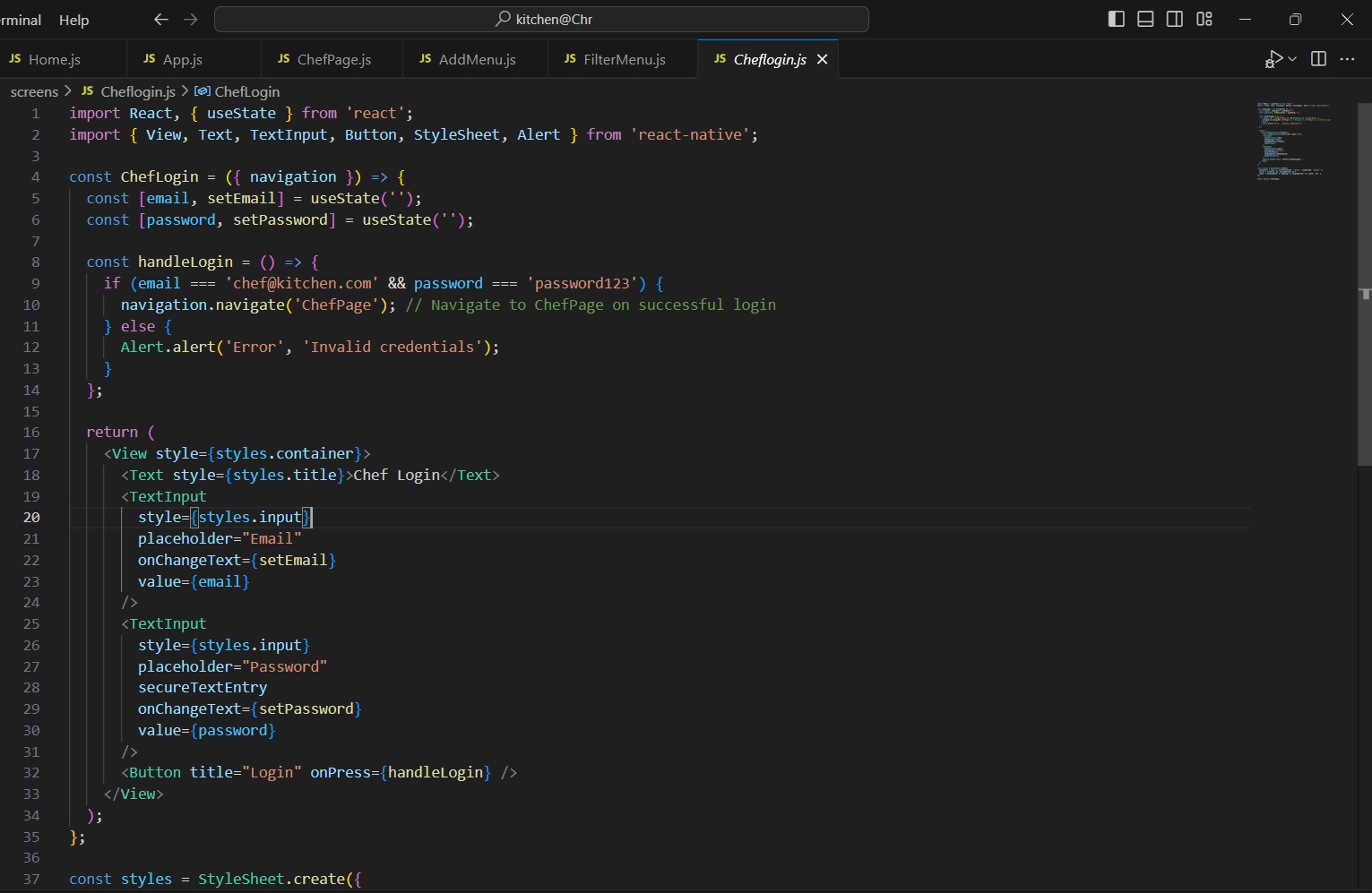
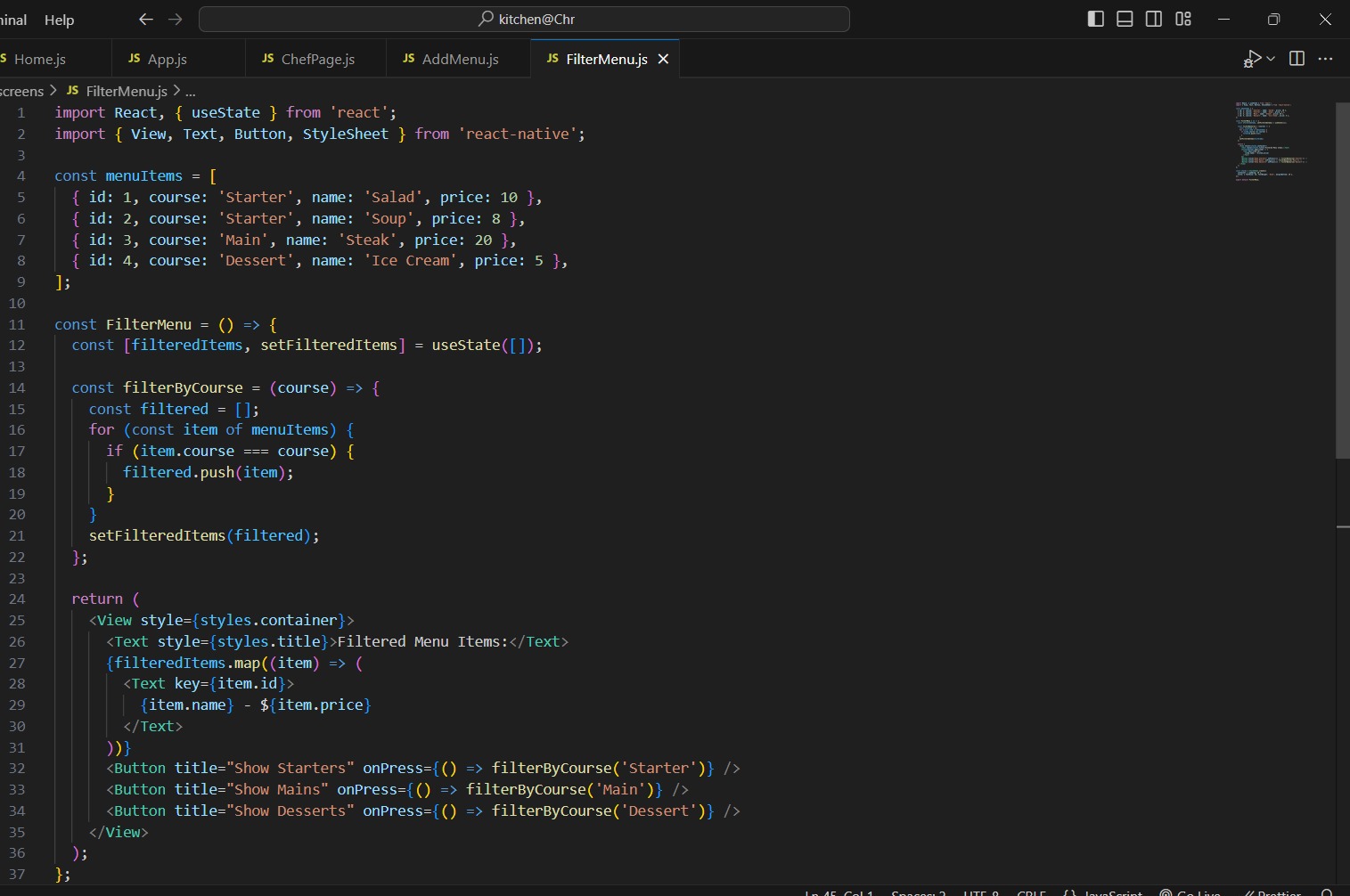
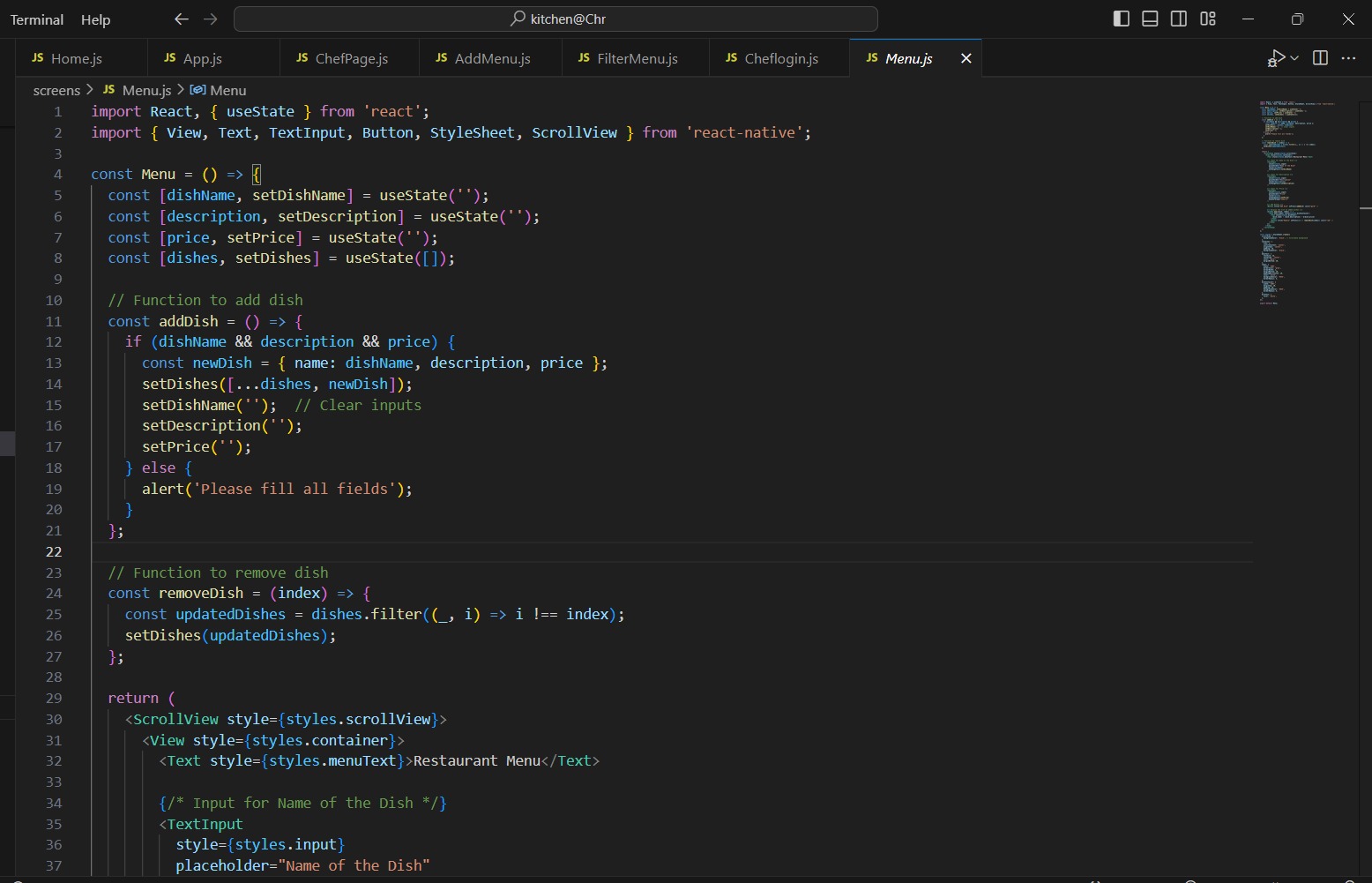
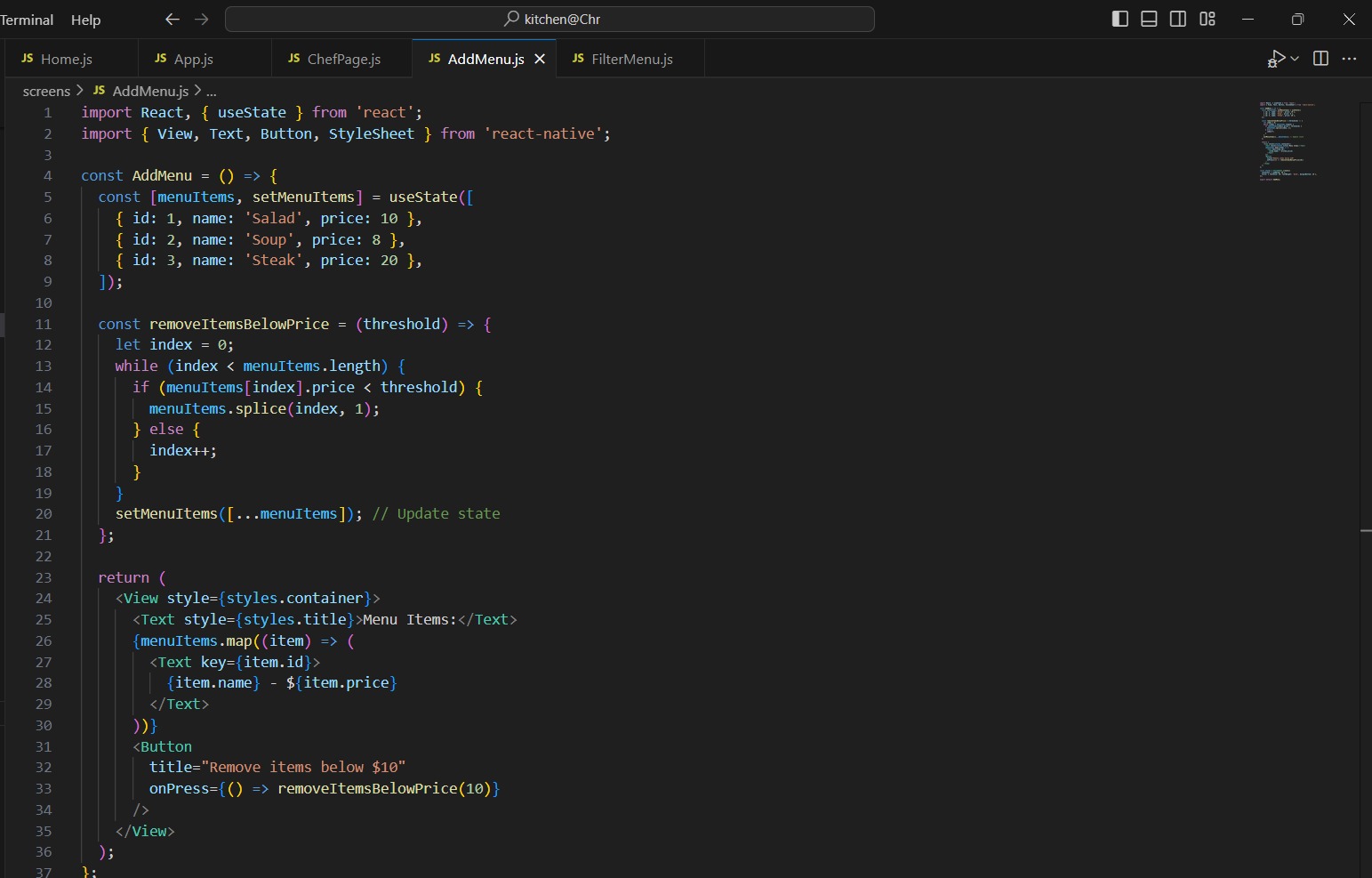
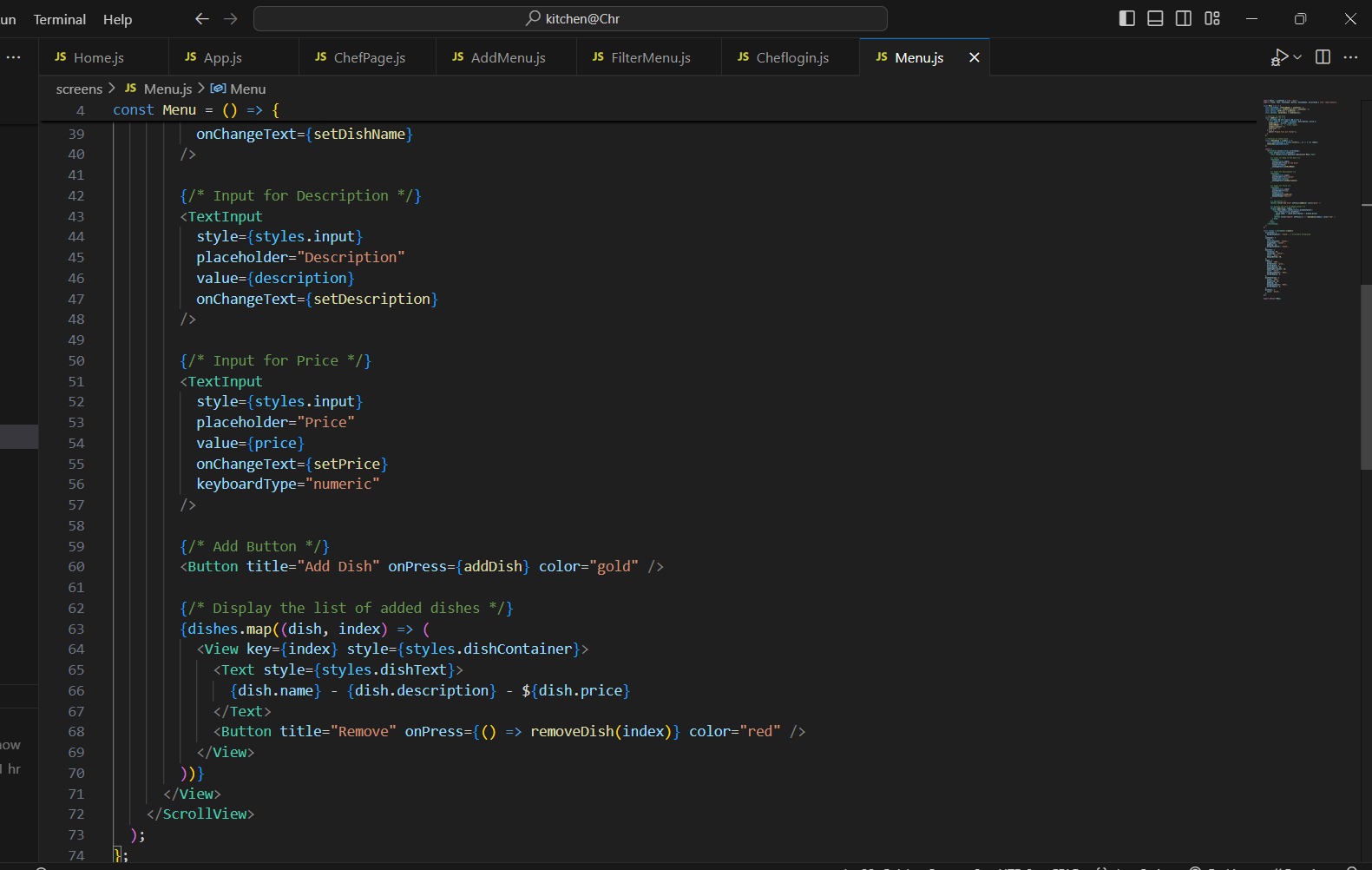
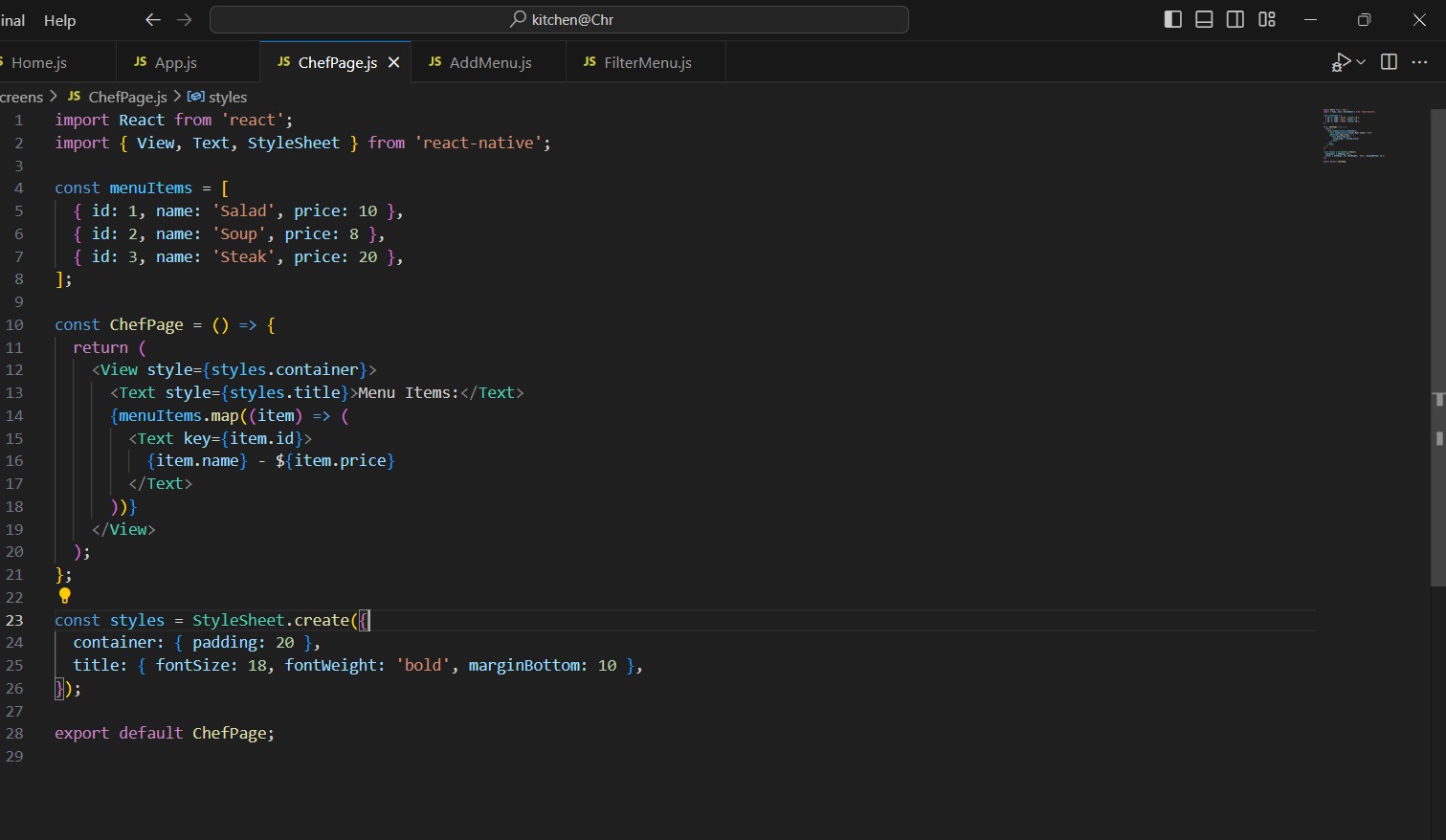
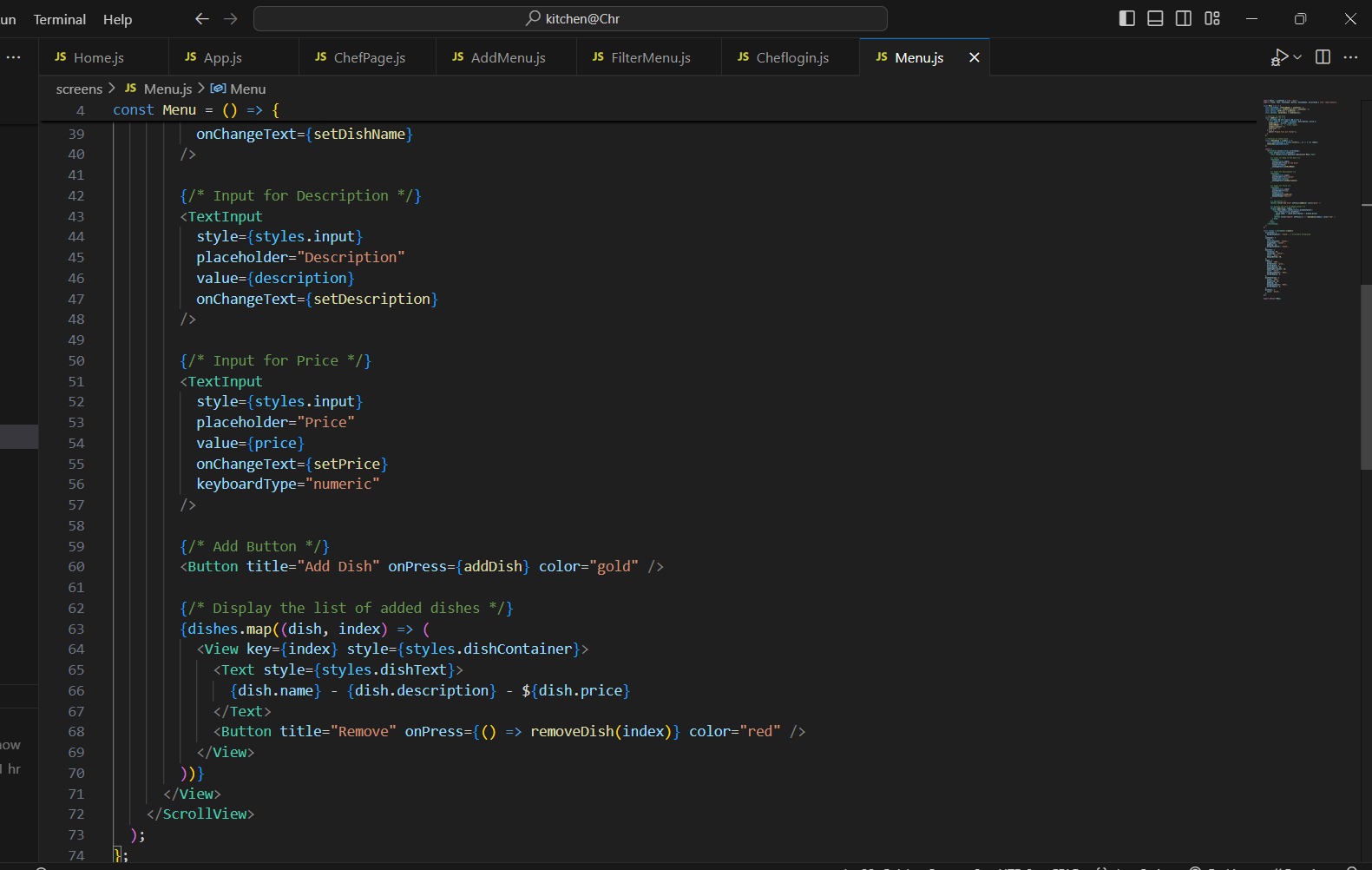
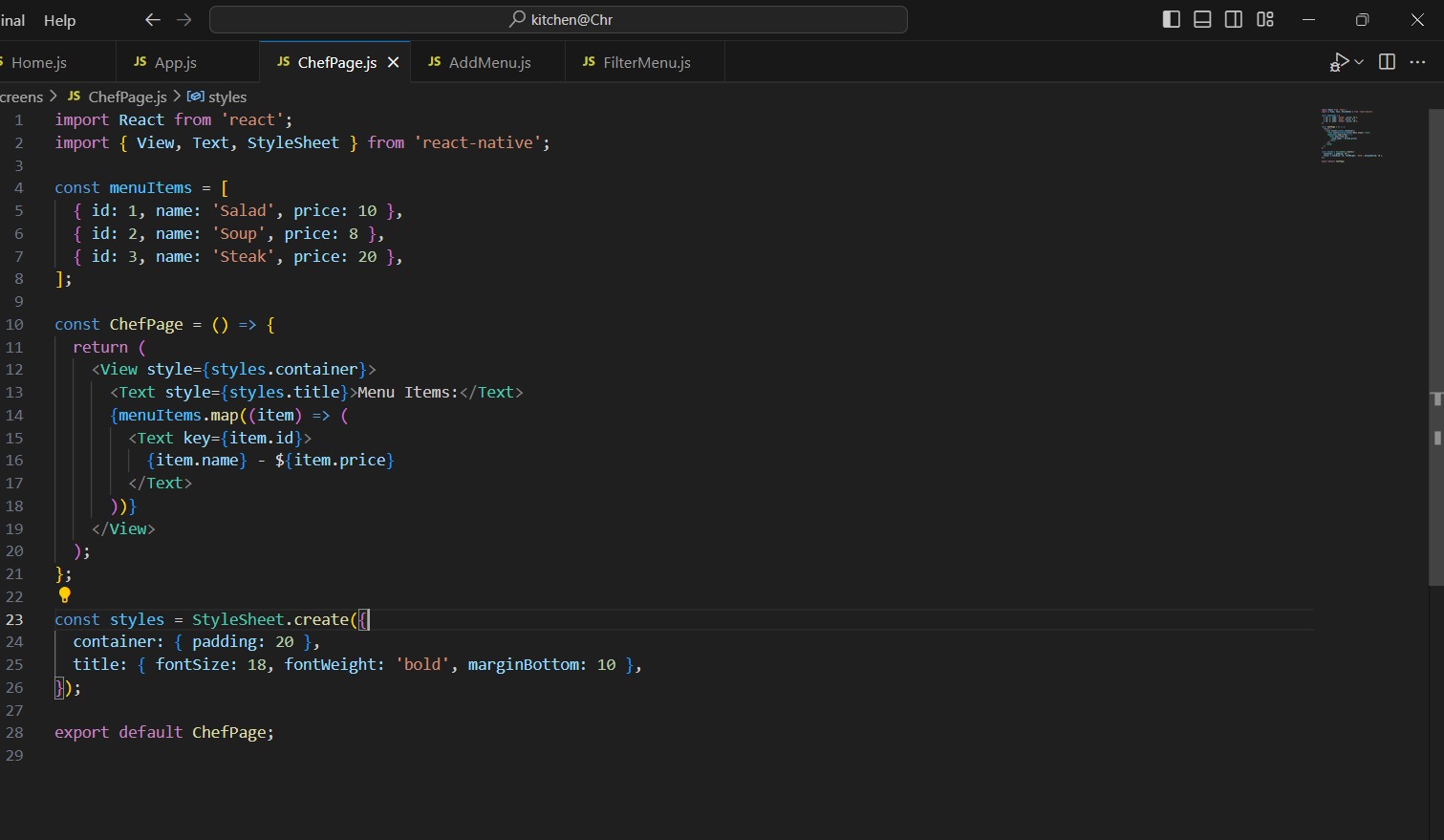
**LINK TO GITHUB:**

https://github.com/ST10461605/MASTPOE

**LINK TO YOUTUBE:**

<https://youtube.com/shorts/Cnn9fQFODxU?si=001gJ3Vbcm1z4Di0>

**SCREENSHOT AND SHORT EXPLANATION OF APP:**



Explanation of Each Screen:

\*\*Kitchen@Chr App Overview\*\*

The Kitchen@Chr app simplifies menu management for chefs and improves the dining experience for guests. Here's what the app does and how it works in simple terms:

---

### \*\*Guest Features\*\*

\*\*1. Home Screen\*\*

- \*\*Purpose\*\*: The first screen guests see, showing an overview of menu prices by course (starters, mains, desserts).

- \*\*Features\*\*:

- Displays the average price for each course.

- Shows links to explore the full menu or filter items.

\*\*2. Filter Menu Screen\*\*

- \*\*Purpose\*\*: Helps guests find specific types of dishes quickly.

- \*\*Features\*\*:

- Filter options for starters, mains, or desserts.

- Displays a list of items based on the selected filter.

\*\*3. Menu Screen\*\*

- \*\*Purpose\*\*: Lets guests browse the full menu.

- \*\*Features\*\*:

- Shows all available dishes with names, descriptions, and prices.

---

### \*\*Chef Features\*\*

\*\*1. Chef Login Screen\*\*

- \*\*Purpose\*\*: Allows chefs to securely log in to manage the menu.

- \*\*Features\*\*:

- Login form with email and password fields.

- Access granted to the ChefPage upon successful login.

\*\*2. Chef Page\*\*

- \*\*Purpose\*\*: A dashboard for chefs to manage the menu.

- \*\*Features\*\*:

- Displays the current menu items.

- Links to add new items, filter options, and other management tools.

\*\*3. AddMenu Screen\*\*

- \*\*Purpose\*\*: Allows chefs to update the menu.

- \*\*Features\*\*:

- Add new dishes by filling out a form (name, price, description, etc.).

- Remove dishes that don’t meet certain criteria (e.g., too cheap).

---

### \*\*Key Functions\*\*

- Dynamic Price Averages\*\*: On the Home screen, the app calculates and displays average prices for starters, mains, and desserts.

- \*\*Filtering\*\*: Guests or chefs can filter menu items to see only what they need (e.g., all starters).

- \*\*Menu Updates: Chefs can add or remove items quickly and see changes immediately.

-How It Works

For Guests:

1. Start on the Home screen to see price averages.

2. Browse all dishes on the Menu screen or use filters to find specific courses.

For Chefs:

1. Log in on the Chef Login screen.

2. Go to the ChefPage to see menu management tools.

3. Add, remove, or filter dishes as needed.

The app keeps things simple while making it easy to manage the menu and explore dining options!

**COPY OF APP’S CODE:**

**Home:**

import React from 'react';

import { View, Text, StyleSheet } from 'react-native';

const menuItems = [

{ course: 'Starter', price: 10 },

{ course: 'Starter', price: 15 },

{ course: 'Main', price: 20 },

{ course: 'Main', price: 25 },

{ course: 'Dessert', price: 8 },

{ course: 'Dessert', price: 12 },

];

const calculateAverages = () => {

const courseTotals = {};

const courseCounts = {};

// Use a for loop to calculate totals and counts

for (const item of menuItems) {

if (!courseTotals[item.course]) {

courseTotals[item.course] = 0;

courseCounts[item.course] = 0;

}

courseTotals[item.course] += item.price;

courseCounts[item.course] += 1;

}

// Calculate averages

const averages = {};

for (const course in courseTotals) {

averages[course] = courseTotals[course] / courseCounts[course];

}

return averages;

};

const Home = () => {

const averages = calculateAverages();

return (

<View style={styles.container}>

<Text style={styles.title}>Average Prices by Course:</Text>

{Object.keys(averages).map((course) => (

<Text key={course}>

{course}: ${averages[course].toFixed(2)}

</Text>

))}

</View>

);

};

const styles = StyleSheet.create({

container: { padding: 20 },

title: { fontSize: 18, fontWeight: 'bold', marginBottom: 10 },

});

export default Home;

**FilterMenu**

import React, { useState } from 'react';

import { View, Text, Button, StyleSheet } from 'react-native';

const menuItems = [

{ id: 1, course: 'Starter', name: 'Salad', price: 10 },

{ id: 2, course: 'Starter', name: 'Soup', price: 8 },

{ id: 3, course: 'Main', name: 'Steak', price: 20 },

{ id: 4, course: 'Dessert', name: 'Ice Cream', price: 5 },

];

const FilterMenu = () => {

const [filteredItems, setFilteredItems] = useState([]);

const filterByCourse = (course) => {

const filtered = [];

for (const item of menuItems) {

if (item.course === course) {

filtered.push(item);

}

}

setFilteredItems(filtered);

};

return (

<View style={styles.container}>

<Text style={styles.title}>Filtered Menu Items:</Text>

{filteredItems.map((item) => (

<Text key={item.id}>

{item.name} - ${item.price}

</Text>

))}

<Button title="Show Starters" onPress={() => filterByCourse('Starter')} />

<Button title="Show Mains" onPress={() => filterByCourse('Main')} />

<Button title="Show Desserts" onPress={() => filterByCourse('Dessert')} />

</View>

);

};

const styles = StyleSheet.create({

container: { padding: 20 },

title: { fontSize: 18, fontWeight: 'bold', marginBottom: 10 },

});

export default FilterMenu;

AddMenu

import React, { useState } from 'react';

import { View, Text, Button, StyleSheet } from 'react-native';

const AddMenu = () => {

const [menuItems, setMenuItems] = useState([

{ id: 1, name: 'Salad', price: 10 },

{ id: 2, name: 'Soup', price: 8 },

{ id: 3, name: 'Steak', price: 20 },

]);

const removeItemsBelowPrice = (threshold) => {

let index = 0;

while (index < menuItems.length) {

if (menuItems[index].price < threshold) {

menuItems.splice(index, 1);

} else {

index++;

}

}

setMenuItems([...menuItems]); // Update state

};

return (

<View style={styles.container}>

<Text style={styles.title}>Menu Items:</Text>

{menuItems.map((item) => (

<Text key={item.id}>

{item.name} - ${item.price}

</Text>

))}

<Button

title="Remove items below $10"

onPress={() => removeItemsBelowPrice(10)}

/>

</View>

);

};

const styles = StyleSheet.create({

container: { padding: 20 },

title: { fontSize: 18, fontWeight: 'bold', marginBottom: 10 },

});

export default AddMenu;

import React from 'react';

import { View, Text, StyleSheet } from 'react-native';

const menuItems = [

{ id: 1, name: 'Salad', price: 10 },

{ id: 2, name: 'Soup', price: 8 },

{ id: 3, name: 'Steak', price: 20 },

];

const ChefPage = () => {

return (

<View style={styles.container}>

<Text style={styles.title}>Menu Items:</Text>

{menuItems.map((item) => (

<Text key={item.id}>

{item.name} - ${item.price}

</Text>

))}

</View>

);

};

const styles = StyleSheet.create({

container: { padding: 20 },

title: { fontSize: 18, fontWeight: 'bold', marginBottom: 10 },

});

export default ChefPage;

import React, { useState } from 'react';

import { View, Text, TextInput, Button, StyleSheet, Alert } from 'react-native';

const ChefLogin = ({ navigation }) => {

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const handleLogin = () => {

if (email === 'chef@kitchen.com' && password === 'password123') {

navigation.navigate('ChefPage'); // Navigate to ChefPage on successful login

} else {

Alert.alert('Error', 'Invalid credentials');

}

};

return (

<View style={styles.container}>

<Text style={styles.title}>Chef Login</Text>

<TextInput

style={styles.input}

placeholder="Email"

onChangeText={setEmail}

value={email}

/>

<TextInput

style={styles.input}

placeholder="Password"

secureTextEntry

onChangeText={setPassword}

value={password}

/>

<Button title="Login" onPress={handleLogin} />

</View>

);

};

const styles = StyleSheet.create({

container: { flex: 1, justifyContent: 'center', alignItems: 'center' },

title: { fontSize: 24, marginBottom: 16 },

input: { borderWidth: 1, padding: 8, marginBottom: 12, width: '80%' },

});

export default ChefLogin;

**REFERENCE:**

*-Britannica*. [online] Available at: <https://www.britannica.com/technology/graphical-user-interface>.

-Canva (2024). *Canva*. [online] Canva.com. Available at: <https://www.canva.com>.

-Levy, S. (2018). Graphical user interface | computing. In: *Encyclopædia*

-mockitt.wondershare.com. (n.d.). *A Free Online Design and Prototyping Tool - Wondershare Mockitt-copy*. [online] Available at: <https://mockitt.wondershare.com/home.html>.

-Pinterest. (n.d.). *Pinterest*. [online] Available at: <https://za.pinterest.com>.

‌-Sketch Elements. (2022). *Sketch Elements | Free UI Kits, Templates, and more Sketch Resources*. [online] Available at: https://sketchelements.com [Accessed 22 Aug. 2024].

reactnative.dev. (n.d.). *React Native*. [online] Available at: <https://reactnative.dev>