Mobile App Scripting

ST10269866

Cameron Naidu

Contents

[Part 1 2](#_Toc183115894)

[1. Design and Interface Outline 2](#_Toc183115895)

[2. Navigation Diagram 2](#_Toc183115896)

[3. Screen-by-Screen Implementation Description 3](#_Toc183115897)

[Home Screen 3](#_Toc183115898)

[Starters Screen 4](#_Toc183115899)

[Mains and Desserts Screens 5](#_Toc183115900)

[4. Considerations for Designing in React Native: 5](#_Toc183115901)

# Part 1

1. Design and Interface Outline  
     
   - Home Screen: Displays options for "Starters," "Mains," and "Desserts." As well as the Logo for restaurant.  
   - Starters Screen: Lists three starter dishes with descriptions of eat starter.  
   - Mains Screen: Lists three main dishes with descriptions of eat starter.  
   - Desserts Screen: Lists three dessert options with descriptions of eat starter.

# Navigation Diagram

Home Screen

# Screen-by-Screen Implementation Description

## Home Screen

Purpose:  
This screen allows users to select between "Starters," "Mains," and "Desserts." This allows users to enter each individual page by clicking a meal option.

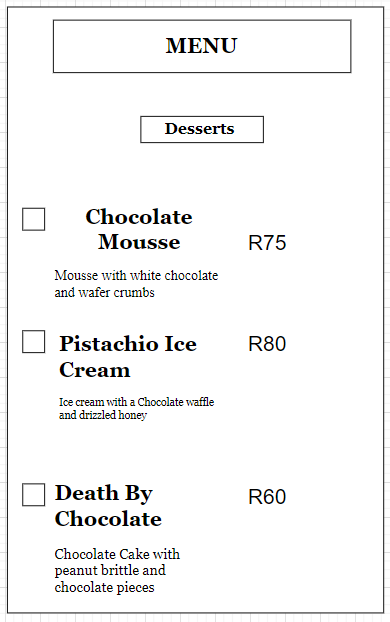
Basic React Native Implementation:\*\*  
- Imports:  
  javascript  
  import React from 'react';  
  import { View, Text, Button } from 'react-native';  
  import { useNavigation } from '@react-navigation/native';  
    
- Components:  
  - `View` for the container.  
  - `Text` for titles or labels.  
  - `Button` for navigation to other screens.  
  
Implementation Example:  
javascript  
function HomeScreen() {  
  const navigation = useNavigation();  
  
  return (  
    <View>  
      <Text>Welcome to the Menu App</Text>  
      <Button title="Starters" onPress={() => navigation.navigate('Starters')} />  
      <Button title="Mains" onPress={() => navigation.navigate('Mains')} />  
      <Button title="Desserts" onPress={() => navigation.navigate('Desserts')} />  
    </View>  
  );  
}

## A menu with text and images Description automatically generated with medium confidenceStarters Screen

Purpose:  
Display a list of available starters. This also allows user to be able to choose what dish they would like by clicking the check box on the left hand side of a dish that they would like to order.  
  
Basic React Native Implementation:  
- Imports:  
  javascript  
  import React from 'react';  
  import { View, Text, FlatList } from 'react-native';  
  import { useNavigation } from '@react-navigation/native';  
    
- Components:  
  - `FlatList` to render a list of items.  
  - `TouchableOpacity` or `Button` to allow interaction with list items.  
  
Implementation Example:  
javascript  
const starters = [  
  { id: '1', name: 'Starter 1' },  
  { id: '2', name: 'Starter 2' },  
  { id: '3', name: 'Starter 3' },  
];  
  
function StartersScreen() {  
  const navigation = useNavigation();  
  
  return (  
    <View>  
      <FlatList  
        data={starters}  
        renderItem={({ item }) => (  
          <TouchableOpacity onPress={() => navigation.navigate('DishDetail', { dish: item })}>  
            <Text>{[Dish](http://item.name/) Name}</Text>  
          </TouchableOpacity>  
        )}  
        keyExtractor={(item) => [dish.id](http://item.id/)}  
      />  
    </View>  
  );

## Mains and Desserts Screens

Purpose:  
Similar to the Starters Screen, these screens list the available main courses and desserts, respectively.  
  
Basic React Native Implementation:  
- Similar to the Starters Screen with appropriate changes to the list data (e.g., `mains` or `desserts`).

A menu with text and numbers

Description automatically generated with medium confidence

1. Considerations for Designing in React Native:  
     
   - Component Reusability: To cut down on duplication, think about making reusable components (such a `DishItem} component for every dish).   
   - Styling: To arrange your styles and create a visually pleasing user interface, use the `StyleSheet`.  
   - Navigation: To manage screen transitions, use React Navigation.   
   - State Management: To handle complicated state, if your project expands, think about utilizing a state management framework like Redux or Context API.   
   - Performance: Make your program as efficient as possible by minimizing pointless re-renders and taking into account lazy loading for images or displays.

# Part 2 Changes made for Final POE

**Initial State of the Project**

Initially, your project comprised of basic React components for displaying a menu, managing categories, and a simple form. The CSS styles were minimal and some elements were unstyled, leading to a basic look and feel. The structure was as follows:

* App.js contained the main application logic.
* Menu.js and Categories.js were used for displaying menu items and filtering them.
* CSS styles were basic, with minimal design elements.

**Improvements and Enhancements**

**1. Enhanced Styles**

The global styles were improved to provide a more polished and cohesive design. Key enhancements include:

* Improved typography for better readability.
* Enhanced color scheme for better contrast and aesthetics.
* Added smooth transitions and hover effects for a more engaging user experience.
* Improved layout and spacing for better organization.

**Updated Styles (CSS)**

css

/\* Global Styles \*/

:root {

--font-primary: 'Helvetica Neue', Arial, sans-serif;

--color-bg: hsl(210, 36%, 96%);

--color-text: hsl(209, 61%, 16%);

--color-accent: hsl(360, 71%, 66%);

--color-accent-dark: hsl(360, 71%, 55%);

--color-border: hsl(210, 22%, 49%);

--color-light: #ccc;

--color-dark-bg: hsl(210, 22%, 49%);

}

body {

margin: 0;

font-family: var(--font-primary);

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

background: var(--color-bg);

color: var(--color-text);

line-height: 1.6;

padding: 0;

box-sizing: border-box;

}

code {

font-family: 'Courier New', monospace;

background: var(--color-light);

padding: 0.2rem 0.4rem;

border-radius: 0.25rem;

}

**2. Navbar Enhancements**

The navbar was redesigned to include:

* A sticky position for better user accessibility.
* Improved color contrast and hover effects for better usability.
* Enhanced logo styling.

**Updated Navbar Styles (CSS)**

css

/\* Top Navbar \*/

.navbar {

position: sticky;

top: 0;

background-color: var(--color-dark-bg);

color: white;

padding: 1rem 2rem;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);

z-index: 100;

}

.navbar ul {

list-style-type: none;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

}

.navbar li {

margin: 0 1.5rem;

}

.navbar a {

text-decoration: none;

color: white;

font-size: 1.1rem;

text-transform: capitalize;

transition: color 0.3s ease;

}

.navbar a:hover {

color: var(--color-accent);

}

.navbar .logo {

font-size: 1.8rem;

font-weight: bold;

color: var(--color-accent);

text-transform: uppercase;

letter-spacing: 2px;

margin-right: auto;

}

.navbar .logo:hover {

color: white;

}

**3. Hero Section Improvements**

The hero section was redesigned for better visual impact:

* Increased padding for a more spacious layout.
* Improved typography for headlines and paragraphs.
* Enhanced button styling with hover effects.

**Updated Hero Section Styles (CSS)**

css

/\* Hero Section \*/

.hero {

background-color: var(--color-dark-bg);

color: white;

padding: 4rem 2rem;

text-align: center;

}

.hero h1 {

font-size: 3rem;

margin-bottom: 1.5rem;

text-transform: uppercase;

letter-spacing: 2px;

}

.hero p {

font-size: 1.5rem;

color: var(--color-light);

margin-bottom: 3rem;

}

.hero button {

background-color: var(--color-accent);

color: white;

border: none;

padding: 1rem 2.5rem;

font-size: 1.2rem;

border-radius: 0.25rem;

cursor: pointer;

transition: background-color 0.3s ease, transform 0.2s ease;

}

.hero button:hover {

background-color: var(--color-accent-dark);

transform: scale(1.05);

}

**4. Menu and Item Styles**

The menu items were styled for better presentation:

* Added box-shadow for a subtle 3D effect.
* Improved spacing and layout for better readability.
* Enhanced hover effects for interactive feedback.

**Updated Menu Item Styles (CSS)**

css

/\* Menu Styling \*/

.menu {

padding: 3rem 0;

background: #fff;

border-radius: 0.5rem;

box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);

}

.menu-item {

display: grid;

gap: 1rem 2rem;

max-width: 35rem;

background: #fff;

border-radius: 0.5rem;

padding: 1.5rem;

box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

transition: transform 0.3s ease;

}

.menu-item:hover {

transform: translateY(-5px);

}

.photo {

object-fit: cover;

height: 250px;

width: 100%;

border: 0.25rem solid var(--color-accent);

border-radius: 0.5rem;

}

.item-info header {

display: flex;

justify-content: space-between;

border-bottom: 0.5px dotted var(--color-border);

padding-bottom: 1rem;

}

.item-info h4 {

margin-bottom: 0.5rem;

font-size: 1.2rem;

}

.price {

color: #c59d5f;

font-weight: bold;

}

.item-text {

margin-bottom: 0;

padding-top: 1rem;

font-size: 1rem;

}

**5. Form and Button Styling**

Forms and buttons were enhanced for better user experience:

* Added padding and border-radius for a more modern look.
* Improved focus states for better accessibility.
* Enhanced button hover effects.

**Updated Form and Button Styles (CSS)**

css

/\* Form Styling \*/

.form {

background: #fff;

max-width: 500px;

margin: 0 auto;

margin-bottom: 4rem;

padding: 3rem;

border-radius: 0.5rem;

box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);

}

.form input,

.form textarea,

.form select {

background: var(--color-bg);

border: 1px solid var(--color-border);

border-radius: 0.25rem;

padding: 0.75rem;

width: 100%;

margin-bottom: 1rem;

font-size: 1rem;

}

.form input:focus,

.form textarea:focus,

.form select:focus {

outline: none;

border-color: var(--color-accent);

box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);

}

textarea {

resize: vertical;

}

/\* Button Styling \*/

button {

background-color: var(--color-accent);

border: none;

color: white;

padding: 0.75rem 1.5rem;

font-size: 1rem;

border-radius: 0.25rem;

cursor: pointer;

transition: background-color 0.3s ease, transform 0.2s ease;

}

button:hover {

background-color: var(--color-accent-dark);

transform: scale(1.05);

}

**6. Manage Menu Section Enhancements**

The ManageMenu component was added.