TITLE:

INFORMATION OF ORDERS

ASSIGNMENT NO 02

MOBILE APPLICATION DEVELOPMENT

PROGRAM: BSSE-6

SUBMITTED TO:

SIR MUHAMMAD KAMRAN

SUBMITTED BY:

ZARNAB IMRAN (sp22-bse-058)

DATE: 25TH, OCT 2024

INTRODUCTION:

The objective of this assignment is to develop a React Native mobile screen that closely replicates the layout of a user profile page with recent order details. The assignment focuses on implementing key UI components and structures such as profile information, user statistics, and a list of recent orders.

Key operations implemented include:

1. **Profile Section**: Displays a profile image, user name, and statistics such as product sales and earnings.
2. **Recent Orders Section**: Dynamically renders a list of recent orders, including images, descriptions, dates, and prices.
3. **User Interaction**: Includes touchable components like a settings button and "See All" for orders, enhancing user experience.

Through this assignment, essential React Native components like View, Text, Image, and ScrollView are utilized, alongside basic styling techniques using StyleSheet. The focus is to create a visually appealing and functional UI.

CODE EXPLANATION:

The following is an explanation of the logic behind the key components and functions implemented in the code for the profile screen:

1. ProfileScreen Component:

This is the main functional component that renders the entire screen layout. It is structured into different sections such as the **profile card** and **recent orders list**. The component uses React Native elements such as `View`, `Text`, `Image`, `ScrollView`, and `TouchableOpacity`.

```
const ProfileScreen = () => {

  return (

    <View style={styles.container}>

      {/* UI Components here */}

    </View>

  );

};
```

- **`<View>`**: Acts as a container for different UI elements.
- **`<Text>`**: Displays textual information like headings, labels, etc.
- **`<Image>`**: Used to display images (profile picture, icons, etc.).
- **`<ScrollView>`**: Allows vertical scrolling, particularly for listing the recent orders.

- **`<TouchableOpacity>`**: Used to create touchable areas like buttons (e.g., settings button, "See All").

2. Profile Section:

```
<View style={styles.profileCard}>

  <View style={styles.profileInfo}>

    <Image source={require('./assets/images/profile.jpg')} style={styles.profileImage} />

    <View style={styles.profileDetails}>

      <Text style={styles.profileName}>Zarnab Imran</Text>

      <View style={styles.stats}>

        {/* Stats Information */}

      </View>

    </View>

  </View>

</View>
```

This section displays user profile details, including:

- **Profile Picture** (`Image`): The user's profile picture is shown using the `Image` component with the local image path.
- **User Name** (`Text`): Displays the name of the user in a bold style.
- **Stats**: Shows the user's statistics (e.g., product sales and earnings), which are presented as a row of icons and values.

3. Statistics Section:

```
            <View style={styles.stats}>
              <View style={styles.statItem}>
                <Image source={require('./assets/images/stat1.png')} style={styles.statIcon} />
                <View style={styles.statText}>
                  <Text style={styles.statValue}>14.850</Text>
                  <Text style={styles.statLabel}>Product Sells</Text>
                </View>
              </View>
</View>
```

This part renders each statistic (product sales and earnings) with an icon (`Image`), a value (`Text`), and a label (`Text`). It uses a combination of horizontal and vertical layouts with Flexbox to align these items in a row.

- **`statItem`**: A flex container for individual statistics.
- **`statIcon`**: Icon related to the stat (e.g., sales or earnings).
- **`statValue`**: Displays the numerical value of the stat.
- **`statLabel`**: Provides a label for the stat (e.g., "Product Sells").

4. **Settings Button**:

```
        <TouchableOpacity style={styles.settingsButton}>
          <Image source={require('./assets/images/settings-icon.png')}
        style={styles.settingsIcon} />
</TouchableOpacity>
```

- This button renders a settings icon (`Image`) inside a `TouchableOpacity`, making it clickable.
- The `TouchableOpacity` component is used to handle user interaction (such as clicking the settings button).
- The settings button is positioned at the top-right of the profile card for quick access.

5. **Recent Orders Section**:

```
        <View style={styles.recentOrdersHeader}>
          <Text style={styles.sectionTitle}>Recent Orders</Text>
          <TouchableOpacity>
            <Text style={styles.seeAll}>See All</Text>
          </TouchableOpacity>
</View>
```

- This part renders the heading of the "Recent Orders" section, followed by a clickable "See All" button.
- **Section Title**: Displays the "Recent Orders" label.
- **"See All" Button**: Allows users to view all orders (though not implemented in this version).

6. **Order List Rendering**:

```
        <ScrollView style={styles.ordersList}>
          {orderData.map((order, index) => (
            <View key={index} style={styles.orderCard}>
              <Image source={require('./assets/images/order-item.png')}
        style={styles.orderImage} />
              <View style={styles.orderDetails}>
                <Text style={styles.orderDescription}>{order.description}</Text>
                <Text style={styles.orderDate}>{order.date}</Text>
              </View>
              <View style={styles.orderPrice}>
                <Text style={styles.priceText}>${order.price}</Text>
              </View>
            </View>
          ))}
</ScrollView>
```

- **Order Data**: An array of objects (`orderData`) holds the recent order details (e.g., description, date, price). This data is used to dynamically render each order in the list.
- **`map` Function**: The array is iterated using `map()` to create a list of order cards (`orderCard`) for each item in the array.

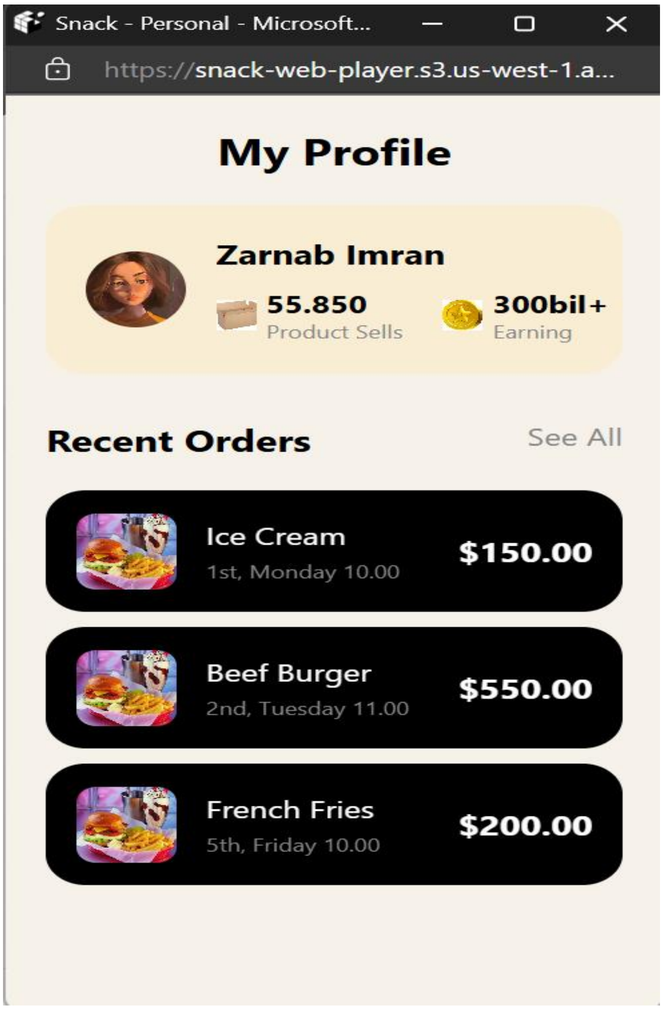
- **Order Card**:
  - **Image**: Shows the image of the order.
  - **Description and Date**: Displays the details of the order, including the items and the order date.
  - **Price**: Shows the price of the order.

7. Styling with StyleSheet:

All the styling in the app is handled using React Native's `StyleSheet.create()` method. Some key styling elements include:

- **`container`**: Defines the layout and background color of the screen.
- **`profileCard`**: Styles the profile card with padding, background color, and rounded corners.
- **`profileImage`**: Styles the profile picture, setting its size and rounding the borders.
- **`orderCard`**: Styles each order card, setting its background color, padding, and border radius.
- **Text Styles:** Various text components such as profile name, stats, order description, and order price have customized font sizes and weights to emphasize importance.

OUTPUT:


Snack - Personal - Microsoft...
https://snack-web-player.s3.us-west-1.a...

My Profile

Zarnab Imran

55.850
Product Sells

300bil+
Earning

Recent Orders

See All

Ice Cream
1st, Monday 10.00
$150.00

Beef Burger
2nd, Tuesday 11.00
$550.00

French Fries
5th, Friday 10.00
$200.00

CONCLUSION:

Through this assignment, I gained hands-on experience in building user interfaces using React Native. I learned how to structure and style complex UI components like profile cards and dynamic lists of recent orders using `View`, `Text`, `Image`, and `ScrollView`. This assignment also helped me to deepen my understanding of Flexbox layout principles and how to achieve responsive designs across different screen sizes.I particularly learned:

- How to handle dynamic data rendering using arrays and the `map()` function to create reusable UI components.
- How to integrate and display images both from remote URLs and local assets.
- How to use `TouchableOpacity` to add user interactions like clickable buttons.

Challenges:

- **Image Handling**: One of the challenges was learning how to integrate images saved locally on my machine into the app, as it required the proper use of the `require()` function in React Native, which differs from handling remote image URLs.
- **Layout Alignment**: Ensuring that all the UI elements were aligned properly, especially the profile section and order list, required careful use of Flexbox for spacing and layout. Achieving consistency across devices was a bit challenging but rewarding.
- **Dynamic List Rendering**: Managing and rendering the order list dynamically posed challenges in maintaining layout consistency and styling, especially when the data changes.

Overall, this assignment helped me to build a stronger foundation in mobile app development using React Native, and it was a valuable experience in designing modern and responsive user interfaces.