* Create new project with the command:  
  npx react-native@latest init RestaurantFinderApp
* Open emulator and create device and start the device.
* Check the project if it is running of not by using the following command:  
  npx react-native run-android
* The project will take time to configure files and it will run smoothly on the emulator device.
* Now stop the project and I have made the respective **folders i.e., components, model, screens**
* I have added the following files in the **component folder.**
* **Card.js:**

Is a React Native component file that renders a card view for displaying restaurant information, including an image, title, ratings, and description. It uses the StarRating component for displaying star ratings.

* **context.js:**Is a file that defines an authentication context using React's createContext API. It can be used to provide authentication-related data and functions to components within the app.
* **StarRating.js:**Is a reusable React Native component that displays star ratings based on the ratings prop passed to it. It uses the Ionicons component from the react-native-vector-icons library and also displays the number of reviews.

Now for **model folder:**

* **data.js:**

This file exports an array named data that contains objects representing food places. Each object has properties such as id, coordinate, title, description, image, rating, reviews, and categories. The Images array contains objects with an image property that stores image paths for the food place banners.

* **mapData.js:**

This file exports an array named markers that represents markers on a map. Each marker object contains properties such as coordinate, title, description, image, rating, and reviews. The Images array in this file is similar to the one in data.js and stores image paths for the markers.Additionally, this file exports two arrays named mapDarkStyle and mapStandardStyle, which represent styles for the map.

* **Notifications.js:**

This file exports an array named Notifications that contains objects representing notifications. Each notification object has properties such as id, title, and details, which provide information about the notification.

* **User.js:**

Each user object represents a user with properties such as id, email, username, password, and userToken.

Now for **screens folder:**

**BookmarkScreen.js:**

This file exports a React component called BookmarkScreen. It displays a screen with a title and a button. When the button is clicked, an alert is shown.

* **CardItemDetails.js:**

This file exports a React component called CardItemDetails. It displays details of a card item, including an image, title, description, categories, and a map view. The component uses various libraries such as react-native-image-header-scroll-view and react-native-maps.

* **CardListScreen.js:**

This file exports a React component called CardListScreen. It displays a list of cards using the FlatList component. Each card item can be pressed to navigate to the CardItemDetails screen.

* **DrawerContent.js:**

This file exports a React component called DrawerContent. It represents the content of a drawer in a navigation menu. It includes user information, navigation items, and a preference section for toggling a dark theme.

* **EditProfileScreen.js:**

This file exports a React component called EditProfileScreen. It displays a screen for editing a user's profile. It includes options for uploading a profile picture using the device's camera or photo library.

The ExploreScreen component is a screen that displays a map with markers and provides a search box and filter options. It uses the react-native-maps library for displaying the map and markers. The screen also includes a scrollable list of cards below the map, which show information about specific locations.

The HomeScreen component is another screen that displays a slider of images and some categories for different types of restaurants. It also includes a section for recently viewed items.

* **ExploreScreen:**

Renders a map using MapView from react-native-maps. Displays markers on the map based on the state.markers data. Provides a search box for searching locations. Includes a horizontal scroll view (ScrollView) for displaying filter options. Renders a scrollable list of cards below the map using Animated.ScrollView. Each card in the list displays information about a specific location and includes an "Order Now" button. The map and the list of cards are synchronized using animations and gestures.

* **HomeScreen:**

Renders a slider of images using Swiper from react-native-swiper. Displays categories as buttons that can navigate to different screens. Includes a section for recently viewed items, represented by cards.

Both screens use various icons from different icon libraries, such as Ionicons, MaterialCommunityIcons, Fontisto, and MaterialIcons, to enhance the UI.

* It defines three stack navigators (HomeStack, NotificationStack, and ProfileStack) and a tab navigator (Tab) using the createStackNavigator and createMaterialBottomTabNavigator functions from React Navigation.
* **The MainTabScreen:**

The entry point of the tab-based navigation. It renders the tab navigator (Tab.Navigator) with four screens: "Home", "Notifications", "Profile", and "Explore". Each screen has a name, component, and options such as tabBarLabel, tabBarColor, and tabBarIcon.

The HomeStackScreen, NotificationStackScreen, and ProfileStackScreen components define the stack navigation for the respective tabs. They include nested screens, options for styling the header, and handling navigation actions.

* **The MapTestScreen:**

Renders a map using the react-native-maps library. It includes a MapView component and a Marker with a Callout that displays additional information.

* **The NotificationScreen:**

Displays a list of notifications using the SwipeListView component from the react-native-swipe-list-view library. It allows swipe actions to delete notifications.The listData state variable holds the notification data, and functions like closeRow, deleteRow, and event handlers are defined to manage the swipe actions.

* **RootStackScreen.js:**

This file defines the root stack navigator for my application. It sets up the navigation stack and includes three screens: SplashScreen, SignInScreen, and SignUpScreen.

* **SplashScreen.js:**

This component represents the splash screen of my application.

* **SignInScreen.js:**

This component represents the screen where users can sign in. It includes form fields for username and password, as well as validation logic for input fields.

* **SignUpScreen.js:**

This component represents the screen where users can sign up for a new account. It includes form fields for username, password, and password confirmation.

* **SettingsScreen.js:**

This component represents a settings screen in my application. It includes a simple layout with a text and a button.

App is running in App.js File which has the following code working:

1. Importing necessary dependencies:

* React and useEffect from 'react': React is the library used for building the UI components, and useEffect is a hook used to perform side effects in functional components.
* View and ActivityIndicator from 'react-native': These components are used to define the structure of the UI and display an activity indicator, respectively.
* NavigationContainer, DefaultTheme, and DarkTheme from '@react-navigation/native': These are part of the React Navigation library, which provides navigation functionality in the app.
* createDrawerNavigator from '@react-navigation/drawer': This function is used to create a drawer navigation component.
* Provider, DefaultTheme, and DarkTheme from 'react-native-paper': These components are part of the React Native Paper library, which provides UI components following Material Design guidelines.
* DrawerContent from './screens/DrawerContent': This is a custom component that represents the content of the drawer navigation.
* MainTabScreen, SupportScreen, SettingsScreen, and BookmarkScreen from './screens/': These are custom screen components used in the app.
* AuthContext from './components/context': This is a custom context component that manages the authentication state.

1. Creating a Drawer navigator component using createDrawerNavigator.
2. Defining the App component, which is the root component of the app.

* It uses the useState hook to manage the dark theme state with the initial value of false.
* It defines an initial login state using useState hook with the initial values of true for isLoading, and null for userName and userToken.
* It defines custom theme objects for the light and dark themes using the NavigationDefaultTheme and NavigationDarkTheme from React Navigation, and PaperDefaultTheme and PaperDarkTheme from React Native Paper.
* The current theme is determined based on the isDarkTheme state.
* It defines a reducer function loginReducer using useReducer hook, which handles the state transitions for login-related actions.
* It creates the loginState and dispatch variables using the loginReducer and initial login state.
* It defines the authContext using useMemo hook, which provides functions for signing in, signing out, signing up, and toggling the theme.
* It uses the useEffect hook to retrieve the user token from AsyncStorage (a storage system provided by React Native) and update the login state accordingly.
* If the app is still loading, it displays an activity indicator.
* If the user is authenticated (userToken is not null), it renders the drawer navigator component with the custom drawer content component and the main, support, settings, and bookmark screens.
* If the user is not authenticated, it renders the RootStackScreen, which is likely responsible for handling the authentication flow.
* The app is wrapped with the PaperProvider and AuthContext.Provider components to provide the theme and authentication context to the child components.

1. The App component is exported as the default export of the module.

In summary, this code sets up the basic structure and navigation for a React Native app with support for a drawer navigation component and authentication flow. It uses React Navigation and React Native Paper libraries for navigation and UI components, respectively.

* **Firebase.js:**  
  firebase is connected with application in which users’ data is stored