Title: Build react native offline app with SQLite database

Contents

[Introduction 1](#_Toc102208776)

[Project plan 1](#_Toc102208777)

[Steps to use SQLite database in react native 3](#_Toc102208778)

[Testing the offline app 4](#_Toc102208779)

[Extra plugins installed and their uses: 16](#_Toc102208780)

[guide to use the app: 17](#_Toc102208781)

[Technology used: 17](#_Toc102208782)

[Learned: 17](#_Toc102208783)

# Introduction

The aim of this project is to create react native offline app with create, read, delete, and edit operations. The app includes the screen to display the list of existing data, function to share, search, remove , edit, and post items. This app works offline. It requires internet connection only for sharing the contents.

In this project, I have used SQLite database to perform all the CRUD operations in the react native application. To begin with, we need following tools:

* React native cli or Expo cli
* Node.js (version 16.14.2)

# Project plan

* App flowchart:

Diagram

Description automatically generated

the above flowchart shows the design, and the working structure of the application.

The app will have two screens. One for the main page, and another for viewing individual items.

On main page, the list of the existing item will be displayed. Search feature will also be available to view the desired item. Likewise, it will have form to add new item, and the feature to select multiple items for deleting.

Pressing the item on the main screen will navigate the app to another screen where user will be able to check the details of the pressed item. In the same screen, features such as delete, share, edit will be available.

* Requirements:

Based on the requirements of the app following resource are required:

* To build cross-platform app : react native frame work
* To build offline app with CRUD operations : SQLite
* To navigate between various screen: react native navigation
* To use search bar : react native elements
* Development :
  + User-interface:

This is the initial phase of the development process. This phase involves designing the front end design of the app.

Building the user-interface of the will take 3-4 days.

* + Testing: Testing is the crucial part of the app development process. It helps to build quality product, and meet the expected result. The responsiveness, different operations of the app will be tested.

Testing the app will take 2-3 days.

# Steps to use SQLite database in react native

* Installing SQLite:

expo install expo-sqlite.

Using the above command, we can install SQLite plugin in our react native application.

Import \* as SQLite from ‘expo-sqlite’

* Open database connection:

Const db= SQLite.opendatabase(‘databaseName’);

* Create table

db.transaction(trans => {

    trans.executeSql('CREATE TABLE IF NOT EXISTS SHOP (ID INTEGER PRIMARY KEY AUTOINCREMENT, NAME TEXT, DESCRIPTION TEXT, PRICE NUMBER, IMAGE TEXT, POST\_DATE DATE)', null,

      () => console.log('table created'),

      () => console.log('something went wrong'))

  })

* Fetch data from table

 db.transaction(trans => {

            trans.executeSql('SELECT \* FROM SHOP', null,

                (object, { rows: { \_array } }) => setFoods(\_array),

                () => console.log('something went wrong'))

        })

* Post data

 db.transaction(trans => {

                                trans.executeSql('INSERT INTO SHOP (name,description,price,image,post\_date) values (?,?,?,?,?)', [name, description, price, image, `${current.getDate()}/${current.getMonth()}/${current.getFullYear()}`],

                                    (object, { rows: { \_array } }) => fetch\_all(),

                                    () => console.log('something went wrong'))

                            })

* Delete data

db.transaction(trns => {

    trns.executeSql("DELETE FROM SHOP WHERE ID = (?)", [id])

  })

# Testing the offline app

This test is done in real iOS device. To test the app, run the app using the command

npm start

in the terminal.

A picture containing shape

Description automatically generated

Initially, there will be no data so the initial screen of the app looks like this.

Graphical user interface

Description automatically generated

After posting new data in our database, the initial screen of our app looks like this.

* Post data:

Diagram

Description automatically generated

Type the name , description, and price of the food. Then upload image by pressing upload image. Then press add button to post the data.

On success, the app generates the alert message.

Graphical user interface, application

Description automatically generated

* Deleting single item

A picture containing graphical user interface

Description automatically generatedA picture containing graphical user interface

Description automatically generated

To delete the post, swipe left on the post, the app opens the slide containing delete button.

On success, post gets removed , and the app routes from current screen to the initial screen.

* Pick image from camera and gallery:

Graphical user interface

Description automatically generated

While editing or adding new post, press upload image button. Then the alert box appears that shows two options: camera and gallery. User can select any options to upload the image.

Graphical user interface, application

Description automatically generated

* Edit post:

Graphical user interface, application, chat or text message

Description automatically generatedGraphical user interface

Description automatically generated

Pressing the edit button opens the edit form. Type values in all the input fields and press edit button. On success , the alert message appears on the screen

* Delete multiple items:

Graphical user interface

Description automatically generated

Long press on the images to select multiple item. Delete text on the top right corner shows the number of items selected. To confirm the item is already selected, yellow box appears on the top left corner of the images.

Graphical user interface, application, chat or text message

Description automatically generated

To delete all the selected items, press delete text which is at the top right corner. On success alert message appears. The items gets removed in real time . the yellow circle showing the number of item selected also disappear.

* Search:

Graphical user interface

Description automatically generated with medium confidence

The search function display the item that matches input text.

# Extra plugins installed and their uses:

* react-native-elements: it provides search bar interface
* expo-sqlite : it provides SQLite database
* react-navigation: it helps to route from one screen to another
* react-native-gesture-handler: to make the components Swipeable

# guide to use the app:

the app has two screens: main screen, and individual screen.

features in main screen

* The initial screen display the existing data in the database.
* Search the food item
* Delete multiple items. ( to select the item, long press it. Press the delete text to remove all selected items )
* Add new item (swipe left to open the form to add new item)

Navigate from main screen to individual screen :

* Single press on the item

Features on individual screen:

* Shows the details of the item
* Share and edit the item
* Delete the item (swipe left, the delete button appears. Press the button to delete the item)

Navigate from individual screen to main screen:

* Press back icon on the top of the screen

# Technology used:

* React native
* Expo framework
* Visual studio

# Functional specification:

App size: 55 MB

Works on iOS and android:

* iOS version required: 10.0 or greater
* android version required: android 4.1 or greater

# Learned:

* React native hook: useState , useEffect ( to store data , and update the state in real-time)
* React native navigation : route and pass between various screens
* Way to build swipeable component