UNIVERSITI TUNKU ABDUL RAHMAN
Faculty of Engineering & Sciences

UECS3253/ WIRELESS APPLICATION DEVELOPMENT

NAME	STUDENT ID
SIVANRAJ A/L VASU	1507022
ERIC DANIEL	1605611

Course	: SOFTWARE ENGINEERING
Practical Group	: P2
Lecturer	: MR. OOI EAN HUAT

TITLE: **TOURIST ATTRACTION GUIDE**

Description

This application will be useful for the tourist to get to know about attraction places in Malaysia. There are beautiful attraction spot in each states in Malaysia. Hence this application has been created to highlight what are the attraction places situated in Malaysia.

Key Features

This application consist of four screens, which is Loading screen, Home screen, View screen and Detail screen.

Loading screen: - To display a simple welcome message when the user enter the application, and the message will be displayed for six seconds.

Home screen: - In this screen there will be a picker with label consist of all the 13 states in Malaysia with a search button. When user select a state and press the search button, it will navigate to the View screen.

View screen: - In this screen, user will be able to see, location and the name of the attraction spot that situated in the state that user has selected in the Home screen. When user click on particular location, it will navigate to Detail screen.

Detail screen: - In this screen user could be able to see all the additional information about a particular tourist spot. They will be able to know the location, name of the tourist spot, address and additional information about the place such as background and highlights of the place.

Data Persistence

For this application, SQLite has been used for data persistence. SQLite database script has been created using python. In this application, we used it to store all the information that we have been used to create this application such as state, location, name of the place, address, additional information (description) of a particular place. All the needed information has been stored here and when we execute this python script, SQLite database file named place.sqlite has been created. We fetch all the data from this file by using a SQLite SELECT query with query parameters. We fetch the data according to our needs.

Connectivity and the Cloud

We used Web-based API to connect to the cloud. As we all know, React-Native provides the Fetch API for connecting to a web-based API via HTTP/HTTPS. Hence we used a fetch method and pass an URL to fetch data from an URL. For that we have created a SQLite database for the web service and created a web service program using python. In this application we used Web-based API to store all the information such as state, location, name of the place, address and description and retrieved it using fetch() method. Hence, the fetch method will return a Promise that makes it straightforward to write code that works in an asynchronous manner.

SOURCE CODE (JAVASCRIPT):

App.js:

```
import {
  createStackNavigator,
} from 'react-navigation';
import HomeScreen from './Screen/HomeScreen';
import ViewScreen from './Screen/ViewScreen';
import DetailScreen from './Screen/DetailScreen';
import LoadingScreen from'./Screen/LoadingScreen';

export default createStackNavigator({
  LoadingScreen:{
    screen: LoadingScreen,
  },
  HomeScreen: {
    screen: HomeScreen,
  },
  ViewScreen: {
    screen: ViewScreen,
  },
  DetailScreen: {
    screen: DetailScreen,
  },
}, {
  initialRouteName: 'LoadingScreen',
});
```

LoadingScreen.js:

```
import React, { Component } from 'react';
import {
  Platform,
  StyleSheet,
  Text,
  View,
} from 'react-native';
console.disableYellowBox = true; //to avoid the warning
import MainServices from './MainServices';
export default class LoadingScreen extends Component<Props>{
  static navigationOptions = {
    title: 'MY TOURIST APP',
  };
  state = {
    loaded: false
  }
  constructor(){
    super();
    MainServices.load(v => this.setState({loaded:true}));
  }
  render(){
    return(
      <View style={styles.container}>
        {this.state.loaded ? this.props.navigation.navigate('HomeScreen') :
          <Text style={styles.text}>
            {`WELCOME TO TOURIST APP\n
                Loading.....`}
          </Text>}
      </View>
    );
  }
}
const styles = StyleSheet.create({
  container:{
    flex: 1,
    alignItems: 'center',
    justifyContent: 'center',
    backgroundColor: '#00ffff',
  },
  text:{
    color: '#000000',
    fontSize: 30,
    fontWeight: 'bold',
    alignItems: 'center',
    justifyContent: 'center',
    paddingBottom: 4,
  }

});
```

HomeScreen.js:

```
import React, { Component } from 'react';
import {
  StyleSheet,
  TextInput,
  Text,
  ScrollView,
  TouchableOpacity
} from 'react-native';
import {
  PickerWithLabel,
  AppButton,
} from './UI';

let common = require('./CommonData');
let SQLite = require('react-native-sqlite-storage');
let config = require('./Config');

type Props = {};
export default class HomeScreen extends Component<Props> {
  static navigationOptions = {
```

```
  title: 'MY TOURIST APP',
};

constructor(props) {
  super(props)

  this.state = {
    places: [],
    isFetching: false,
    selectedStateId: '',
    noneselectedStateId: true,
    //selectedIndex : '',
  };

  this._load = this._load.bind(this);
  this._query = this._query.bind(this);

  this.db = SQLite.openDatabase({
    name: 'place',
    createFromLocation : '~place.sqlite'
  }, this.openDb, this.errorDb);
}
openDb(){
  console.log('Database opened succesfully');
}
errorDb(err){
  console.log('Error occured', err);
}
_query(theState){
  this.db.transaction((tx) => {
    tx.executeSql('INSERT into places(state) values(?)',[theState])
  })
}
componentDidMount() {
  this._load();
  this._query();
}

_load() {
  let url = config.settings.serverPath + '/api/places/state/';
  let url_url= '';
  if(this.state.noneselectedStateId){
    url_url = url;
  }
  else{
    url_url = url + this.state.selectedStateId;
  }
  this.setState({isFetching: true});
  console.log(url_url);
  fetch(url_url,{
    method: 'GET',
    headers:{
      'Accept':'application/json',
      'Content-Type': 'application/json',
    }})
  .then((response) => {
    if(!response.ok) {
      Alert.alert('Error', response.status.toString());
      throw Error('Error ' + response.status);
    }
    return response.json()
  })
  .then((places) => {
    this.setState({places:places});
    this.setState({isFetching: false});
  })
  .catch((error) => {
    console.log(error)
  });
}

render(){
  return(
    <ScrollView style = {styles.container}>
      <PickerWithLabel style = {styles.picker}
        label = {'STATE:'}
        prompt={'Select State'}
        items = {common.states}
        mode = {'dialog'}
        initValue = {99}
        value = {this.state.selectedStateId}
        onValueChange={(itemValue, itemIndex) => {
          this.setState({
            selectedStateId: itemValue,
            noneselectedStateId: false,
          })
      }}
      />
        <AppButton style={styles.button}
        title={'Search'}
        theme = {'primary'}
        onPress={
          () => {
            this.props.navigation.navigate('ViewScreen',
            {
              selectedStateId : this.state.selectedStateId,
            })
          }
        }
        />
    </ScrollView>
  );
  }
}
  const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20,
    backgroundColor: '#00ffff',
  },
  picker: {
    color: 'maroon',
    marginTop: 10,
    marginBottom: 10,
    fontSize: 50,
  },
});
```

ViewScreen.js:

```
import React, {Component} from 'react';
import {
    Text, FlatList, View, StyleSheet, TouchableHighlight,Alert,
} from 'react-native';
import { FloatingAction } from 'react-native-floating-action';
console.disableYellowBox = true; //to avoid the warning
let SQLite = require('react-native-sqlite-storage'); // to load the sqlite file
let config = require('./Config');

export default class VieScreen extends Component{

    static navigationOptions = {
        title: 'MY TOURIST APP',
    }

    constructor(props) {
        super(props)

        this.state = {
          places: [],
          isFetching : false,
        };

        this._query = this._query.bind(this);
        this._load = this._load.bind(this);

        this.db = SQLite.openDatabase({
          name: 'place',
          createFromLocation : '~place.sqlite'
        }, this.openDb, this.errorDb);
      }

      componentDidMount() {
```

```
    this._query();
    this._load();
  }

  _query() {
    let id = this.props.navigation.getParam("selectedStateId")
    //alert(id)
    this.db.transaction((tx) => {
      tx.executeSql(`SELECT * FROM places WHERE state = ?` , [
        id
      ], (tx, results) => {
        this.setState({
          places: results.rows.raw(),
        })
      })
    });
  }

  openDb() {
      console.log('Database opened');
  }

  errorDb(err) {
      console.log('SQL Error: ' + err);
  }
  _load(){
    let url = config.settings.serverPath;
    let url_url = url + '/api/places/state/view/' + this.props.navigation.getParam('selectedStateId');

    this.setState({isFetching: true});
    console.log(url_url);
    fetch(url_url,{method: 'GET',
    headers:{
      Accept: 'application/json',
      'Content-Type': 'application/json',
    }})
    .then((response) => {
      if(!response.ok){
        Alert.alert('Error', response.status.toString());
        throw Error('Error'+ response.status);
      }
      return response.json()
    })
    .then((places) => {
      this.setState({places:places});
      this.setState({isFetching:false});
    })
    .catch((error) => {
      console.log(error)
    });
  }
```

```
   render(){
      return(
         <View style={styles.container}>

         <FlatList
            data={ this.state.places }
            extraData={this.state}
            showsVerticalScrollIndicator={ true }
            renderItem={({item}) =>
               <TouchableHighlight
               underlayColor={'#cccccc'}
               onPress={ () => {
                  //Alert.alert('You tapped the button!')
                  this.props.navigation.navigate('DetailScreen', {
                  id: item.id,
                  headerTitle: item.title,
                  refresh: this._query,
                  refresh: this._load,
                  })
               }}
               >
               <View style={styles.item}>
                  <Text style={styles.itemLocation}>{ item.location }</Text>
                  <Text style={styles.itemName}>Tourist Spot : { item.name }</Text>
               </View>

               </TouchableHighlight>
            }
            keyExtractor={(item) => {item.id.toString()}}
         />
         </View>
      );
   }
}

const styles = StyleSheet.create({
   container: {
     flex: 1,
     justifyContent: 'flex-start',
     backgroundColor: '#3366cc',
   },
   item: {
     justifyContent: 'center',
     paddingTop: 10,
     paddingBottom: 10,
     paddingLeft: 25,
     paddingRight: 25,
     borderBottomWidth: 1,
     borderColor: '#660066',
   },
   itemLocation: {
     fontSize: 30,
     fontWeight: '500',
     color: '#000',

   },

   itemName: {
     fontSize: 17,
     color: 'black',
     fontWeight: '500',
   },
 });
```

DetailScreen.js:

```
import React, {Component} from 'react';
import {
    Text, View, ScrollView, StyleSheet,
} from 'react-native';
import {InputWithLabel} from './UI';
let SQLite = require('react-native-sqlite-storage');
let config = require('./Config');

export default class DetailScreen extends Component{

    static navigationOptions = {
      title: 'MY TOURIST APP',
    }

      constructor(props) {
        super(props)

        this.state = {
          placesId: this.props.navigation.getParam('id'),
          places: [],
          isFetching : false,
        };

        this._query = this._query.bind(this);
        this._load = this._load.bind(this);
        this.db = SQLite.openDatabase({name: 'place', createFromLocation : '~place.sqlite'}, this.openDb,
this.errorDb);
      }

      componentDidMount() {
        this._query();
        this._load();
      }



      _query() {
        this.db.transaction((tx) => {
```

```
      tx.executeSql('SELECT * FROM places WHERE id = ?', [this.state.placesId], (tx, results) => {
        if(results.rows.length) {
          this.setState({
            places: results.rows.item(0),
          })
        }
      })
    });
  }

openDb() {
    console.log('Database opened');
}

errorDb(err) {
    console.log('SQL Error: ' + err);
}
_load(){
  let url = config.settings.serverPath;
  let url_url = url + '/api/places/state/detail/' + this.props.navigation.getParam('placesId');

  this.setState({isFetching: true});
  console.log(url_url);
  fetch(url_url,{method: 'GET',
  headers:{
    Accept: 'application/json',
    'Content-Type': 'application/json',
  }})
  .then((response) => {
    if(!response.ok){
      Alert.alert('Error', response.status.toString());
      throw Error('Error'+ response.status);
    }
    return response.json()
  })
  .then((places) => {
    this.setState({places:places});
    this.setState({isFetching:false});
  })
  .catch((error) => {
    console.log(error)
  });
}

render(){
    let places = this.state.places;
    {console.log(places)}
    return(
        <View style={styles.container}>
            <ScrollView>
                <InputWithLabel style={styles.output}
```

```
                        multiline = {true}
                        label={'Location'}
                        value={places ? places.location : ''}
                        orientation={'vertical'}
                        editable={false}
                    />
                    <InputWithLabel style={styles.output}
                        multiline = {true}
                        label={'Tourist Spot'}
                        value={places ? places.name : ''}
                        orientation={'vertical'}
                        editable={false}
                    />
                    <InputWithLabel style={styles.output}
                        multiline = {true}
                        label={'Address'}
                        value={places ? places.address : ''}
                        orientation={'vertical'}
                        editable={false}
                    />

                    <InputWithLabel style={styles.output}
                        multiline = {true}
                        label={'Additional Info'}
                        value={places ? places.description : ''}
                        orientation={'vertical'}
                        editable={false}
                    />

                </ScrollView>
            </View>
        );
    }
}

const styles = StyleSheet.create({
    container: {
      flex: 1,
      padding: 20,
      backgroundColor: '#bfbfbf',
    },
    output: {
      fontSize: 20,
      fontWeight: 'bold',
      color: '#000099',
      marginTop: 10,
      marginBottom: 10,
      borderBottomWidth: 1,
      borderColor: '#660066',
    },
  });
```

MainServices.js:

```
export default class MainServices{
  static load(cb){
    setTimeout(cb,3000);
  }
}
```

Config.js:

```
exports.settings = {
  serverPath: 'http://10.0.2.2:5000',
}
```

CommonData.js:

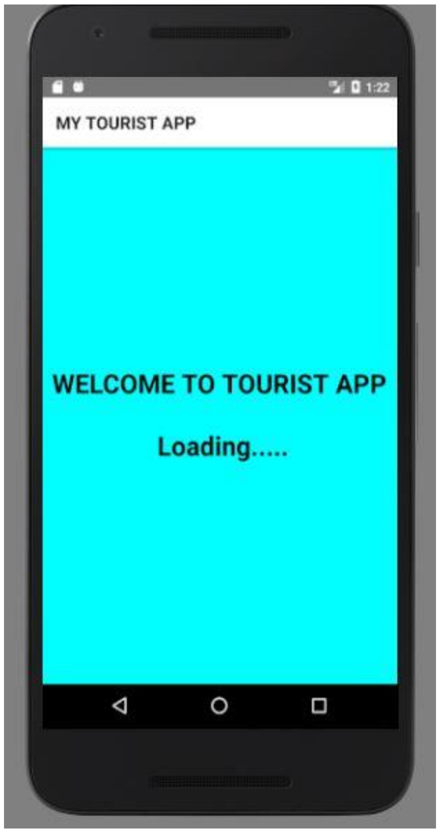
```
exports.getValue = (array, key) => {
  return array.filter((o) => o.key === key)[0].value
}

exports.states = [
  {key: '01', value: 'Penang'},
  {key: '02', value: 'Selangor'},
  {key: '03', value: 'Johor'},
  {key: '04', value: 'Sabah'},
  {key: '05', value: 'Sarawak'},
  {key: '06', value: 'Melaka'},
  {key: '07', value: 'Perak'},
  {key: '08', value: 'Kedah'},
  {key: '09', value: 'Pahang'},
  {key: '10', value: 'Terengganu'},
  {key: '11', value: 'Kelantan '},
  {key: '12', value: 'Negeri Sembilan'},
  {key: '13', value: 'Perlis'},
];
```

UI.js:

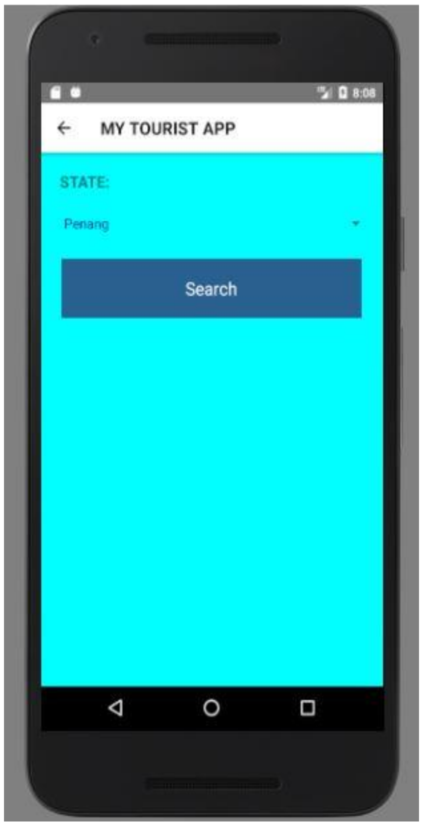
```
import React, { Component } from 'react';
import {
    Platform,
    View,
    Text,
    TextInput,
    Picker,
    TouchableNativeFeedback,
    StyleSheet,
} from 'react-native';

/**
 * InputWithLabel
 */
class InputWithLabel extends Component {
    constructor(props) {
        super(props);

        this.orientation = this.props.orientation ? (this.props.orientation == 'horizontal' ? 'row' : 'column') :
'column';
    }

    render() {
        return (
            <View style={[inputStyles.container, {flexDirection: this.orientation}]}>
                <Text style={inputStyles.label}>
                    {this.props.label ? this.props.label : ''}
                </Text>
                <TextInput style={[inputStyles.input, this.props.style]}
                    placeholder={this.props.placeholder ? this.props.placeholder : ''}
                    value={this.props.value}
                    onChangeText={this.props.onChangeText}
                    multiline={this.props.multiline ? this.props.multiline : false}
                    keyboardType={this.props.keyboardType ? this.props.keyboardType : 'default'}
                    secureTextEntry={this.props.secureTextEntry ? this.props.secureTextEntry : false}
                    selectTextOnFocus={this.props.selectTextOnFocus ? this.props.selectTextOnFocus : false}
                    editable={this.props.editable !== null ? this.props.editable : true}
                />
            </View>
        )
    }
}

/**
 * PickerWithLabel
 */
class PickerWithLabel extends Component {
    constructor(props) {
        super(props);

        this.orientation = this.props.orientation ? (this.props.orientation == 'horizontal' ? 'row' : 'column') :
'column';
    }

    render() {
        return (
            <View style={[inputStyles.container, {flexDirection: this.orientation}]}>
              <Text style={inputStyles.label}>
                {this.props.label ? this.props.label : ''}
              </Text>
              <Picker
                style={this.props.style}
                mode={this.props.mode ? this.props.mode : 'dropdown'}
                prompt={this.props.prompt ? this.props.prompt : ''}
                selectedValue={this.props.value}
                onValueChange={this.props.onValueChange}
                textStyle={this.props.textStyle ? this.props.textStyle : {fontSize: 18}}
        >
          {this.props.items.map((item, index) => {
            return(<Picker.Item label={item.value} value={item.key} key={item.key} />)
          })}
        </Picker>
      </View>
    )
  }
}

const inputStyles = StyleSheet.create({
    container: {
        flex: 1,
    },
    label: {
        flex: 1,
        fontSize: 18,
        fontWeight: 'bold',
        marginLeft: 3,
        textAlignVertical: 'center',
    },
    input: {
        flex: 3,
        fontSize: 20,
    },
});

/**
 * AppButton
 */
class AppButton extends Component {
    constructor(props) {
        super(props);

        if(props.theme) {
            switch(props.theme) {
                case 'success':
                    this.backgroundColor = '#449d44';
                    break;
                case 'info':
                    this.backgroundColor = '#31b0d5';
                    break;
                case 'warning':
                    this.backgroundColor = '#ec971f';
                    break;
                case 'danger':
                    this.backgroundColor = '#c9302c';
                    break;
                case 'primary':
                default:
                    this.backgroundColor = '#286090';
            }
        }
        else {
            this.backgroundColor = '#286090';
        }
    }

    render() {
        return (
            <TouchableNativeFeedback
                onPress={this.props.onPress}
                onLongPress={this.props.onLongPress}
                background={Platform.OS === 'android' ? TouchableNativeFeedback.SelectableBackground() :
''}>
                <View style={[buttonStyles.button, this.props.style, {backgroundColor:
this.backgroundColor}]}>
                    <Text style={buttonStyles.buttonText}>{this.props.title}</Text>
                </View>
            </TouchableNativeFeedback>
        )
    }
}

const buttonStyles = StyleSheet.create({
    button: {
        margin: 5,
        alignItems: 'center',
    },
    buttonText: {
        padding: 20,
        fontSize: 20,
        color: 'white'
    },
});

/**
 * Export modules
 */
module.exports = {
    InputWithLabel: InputWithLabel,
    PickerWithLabel: PickerWithLabel,
    AppButton: AppButton,
}
```

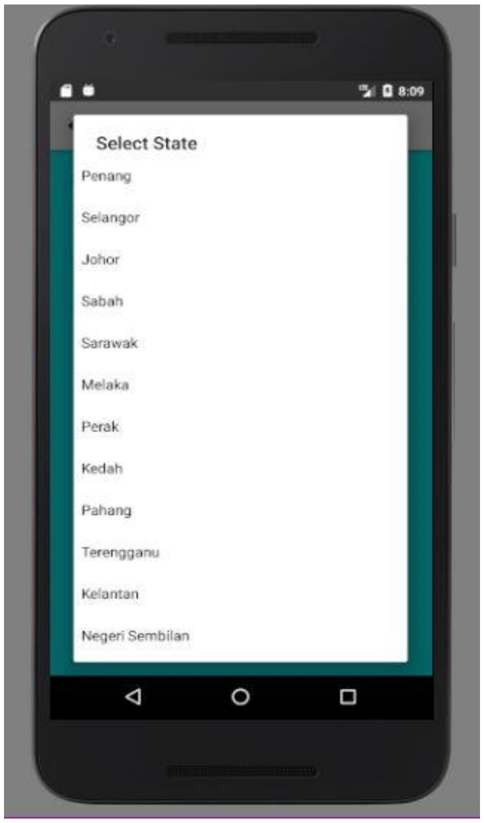
Sample Screen Output

LoadingScreen:

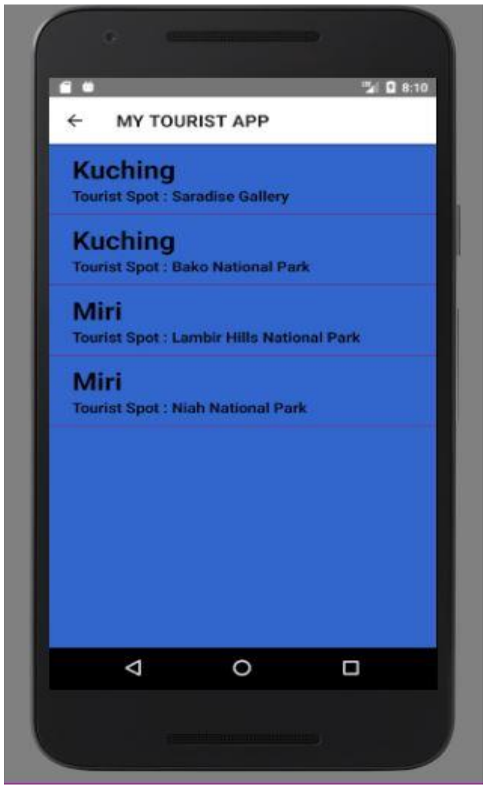


HomeScreen:





ViewScreen:



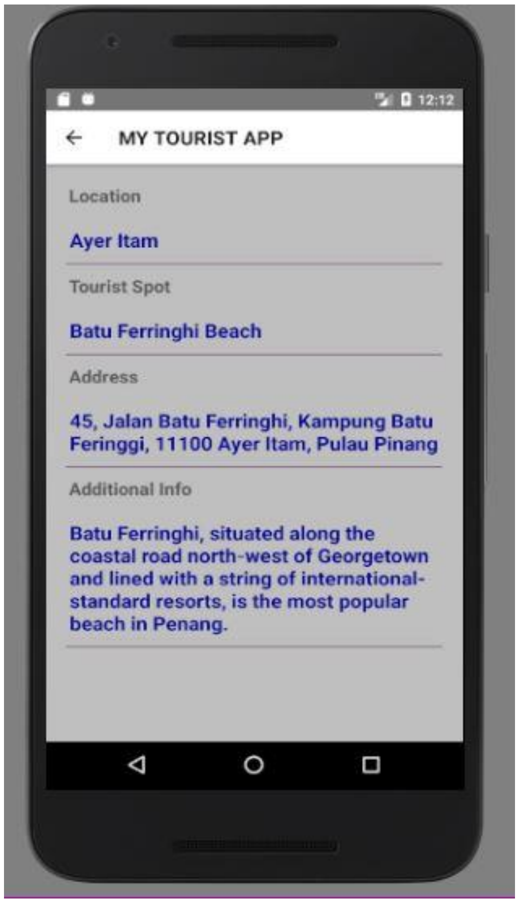
Kuching
Tourist Spot : Saradise Gallery

Kuching
Tourist Spot : Bako National Park

Miri
Tourist Spot : Lambir Hills National Park

Miri
Tourist Spot : Niah National Park

DetailScreen:



Location
Ayer Itam

Tourist Spot
Batu Ferringhi Beach

Address
45, Jalan Batu Ferringhi, Kampung Batu Feringgi, 11100 Ayer Itam, Pulau Pinang

Additional Info
Batu Ferringhi, situated along the coastal road north-west of Georgetown and lined with a string of international-standard resorts, is the most popular beach in Penang.