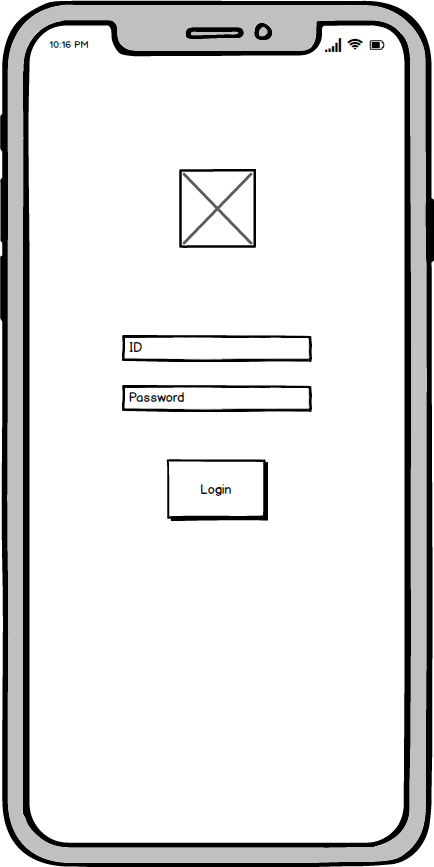
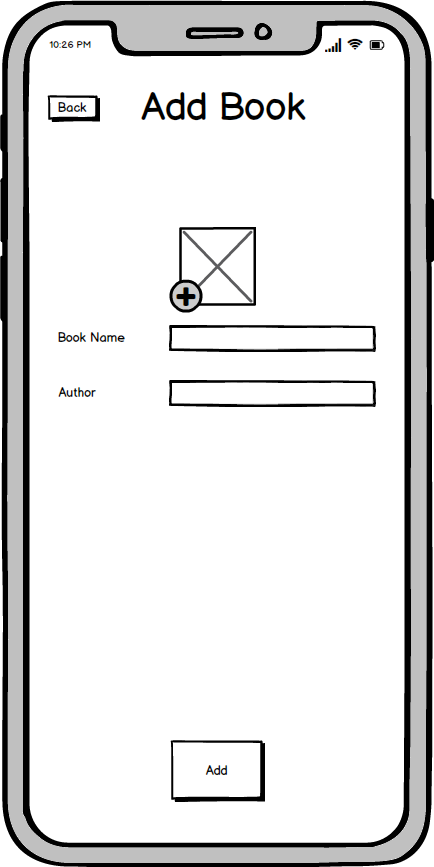
1. SRS:
   1. Chức năng/Thành phần #1: Màn hình giao diện login
      1. Giao diện người dùng



* + 1. Yêu cầu chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Mô tả** | **Hành động** | **Phản hồi** |
| **Giao diện**  **“Đăng nhập”** | Giao diện “Đăng nhập” có:   * Image “Icon” * Textfield “Tên đăng nhập” * Textfield “Mật khẩu” * Button “Đăng nhập” | N/A | Tự động đăng nhập nếu người dùng đã đăng nhập từ trước và chưa đăng xuất |
| **Image**  **“Icon”** | Hình ảnh đại diện hoặc logo của ứng dụng | N/A | N/A |
| **Textfield**  **“Tên đăng nhập”** | Nhập tên đăng nhập của người dùng  Không được để trống | N/A | N/A |
| Finish editing | Focus on TextField “Mật khẩu” |
| **Textfield**  **“Mật khẩu"** | Nhập mật khẩu của tài khoản người dùng  Không được để trống | N/A | N/A |
| Focus on | Nếu “Tên đăng nhập” không hợp lệ, hiển thị thông báo lỗi thiêu "Tên đăng nhập” |
| **Button**  **“Đăng nhập”** | Yêu cầu đăng nhập vào hệ thống | N/A | Vô hiệu hóa nếu người dùng chưa nhập “Tên đăng nhập” hoặc “Mật khẩu” |
| Tap on | Tìm và so sánh “Tên đăng nhập” và “Mật khẩu” trong csdl online  Nếu trùng khớp thì chuyển đến giao diện “Danh sách sách” |

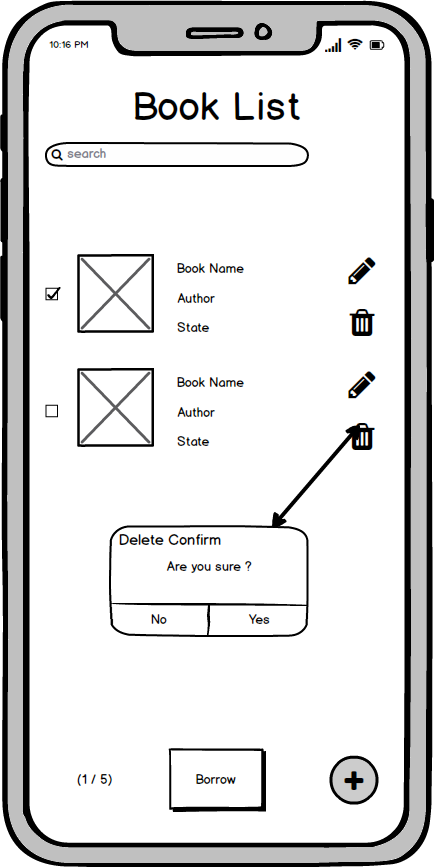
* 1. Chức năng/Thành phần #2: Tạo sách
     1. Giao diện người dùng



* + 1. Yêu cầu chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Mô tả** | **Hành động** | **Phản hồi** |
| **Giao diện**  **“Tạo sách”** | Giao diện “Tạo sách” có:   * Title * Button “Back” * Image “Ảnh” * TextField “Tên sách” * TextField “Tác giả” * Button “Tạo” | N/A | N/A |
| **Title** | Hiển thị tên của giao diện “Tạo sách” | N/A | N/A |
| **Button “Back”** | Trờ về màn hình chính | Tap on | Chuyển về màn hình “Danh sách sách” |
| **Image**  **“Ảnh”** | Hình ảnh đại diện của sách | Tap on | Yêu cầu người dùng chọn ảnh từ thư viện hoặc chụp ảnh mới |
| **TextField**  **“Tên sách”** | Nhập tên sách  Không được để trống | N/A | N/A |
| **TextField**  **“Tác giả”** | Nhập tên tác giả của sách  Không được để trống | N/A | N/A |
| **Button**  **“Tạo”** | Tạo sách mới | Tap on | Yêu cầu hệ thống tạo sách mới  Nếu thành công, hiển thị thông báo và chuyển về giao diện “Danh sách sách”  Nếu thất bại, hiển thị thông báo lỗi |
| N/A | Vô hiệu hóa nếu bất kì thông tin nào không hợp lệ |

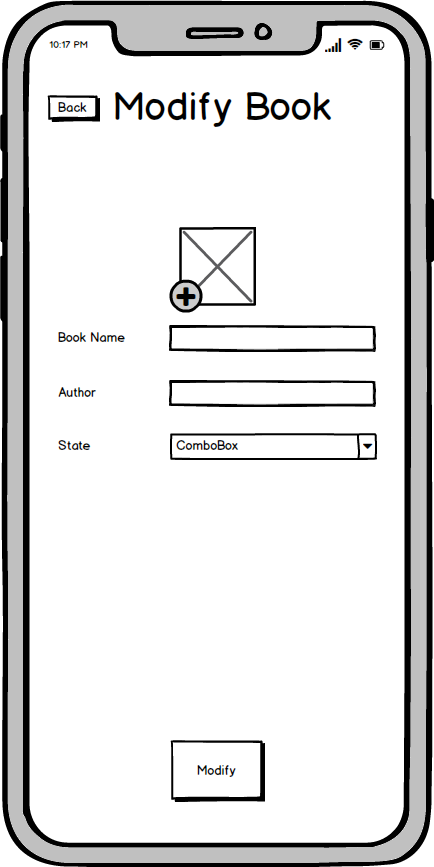
* 1. Chức năng/Thành phần #3: Danh sách sách
     1. Giao diện người dùng



* + 1. Yêu cầu chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Mô tả** | **Hành động** | **Phản hồi** |
| **Giao diện**  **“Danh sách sách”** | Giao diện “Danh sách sách” có:   * Title * SearchView * ListView “Danh sách sách” * Label “Số lượng sách mượn” * Button “Borrow” * Button “+” | N/A | N/A |
| **Title** | Hiển thị tên của giao diện “Danh sách sách” | N/A | N/A |
| **SearchView** | Tìm sách theo tên | Tap on biểu tượng tìm kiếm | Cập nhật danh sách các sách phù hợp với điều kiện tìm kiếm |
| **ListView**  **“Danh sách sách”** | Hiển thị danh sách sách  Mỗi item tương ứng với thông tin 1 quyển sách | N/A | N/A |
| **Item**  **“Sách”** | Hiển thị thông tin nhân viên:   * Checkbox “Selected” * Image “Ảnh” * Label “Tên sách” * Label “Tác giả” * Label “Trạng thái sách” * Button “Xóa” | Tap on | Check checkbox “Selected” |
| **Image**  **“Ảnh”** | Hiển thị hình ảnh đại diện của sách | N/A | N/A |
| **Label**  **“Tên sách”** | Hiển thị tên của quyển sách | N/A | N/A |
| **Label**  **“Tác giả** | Hiển thị tên tác giả của quyển sách | N/A | N/A |
| **Label**  **“Trạng thái sách”** | Hiển thị trạng thái hiện tại của sách: (Đã mượn | Còn) | N/A | N/A |
| **Button**  **“Xóa”** | Xóa sách khỏi danh sách | Tap on | Hiển thị hộp thoại xác nhận xóa sách  Tap “OK” để yêu cầu hệ thống xóa sách khỏi csdl  Nếu thất bại, hiển thị thông báo lỗi |
| **Label**  **“Số lượng sách mượn”** | Hiển thị số lượng sách đã mượn, tối đa là 5 quyển, nếu đã chọn đủ lượng sách, không thể chọn thêm | N/A | N/A |
| **Button**  **“Borrow”** | Cho mượn sách | Tap on | Chuyển đến giao diện “Mượn sách” |
| **Button**  **“+”** | Tạo sách mới | Tap on | Chuyển đến giao diện “Tạo sách” |

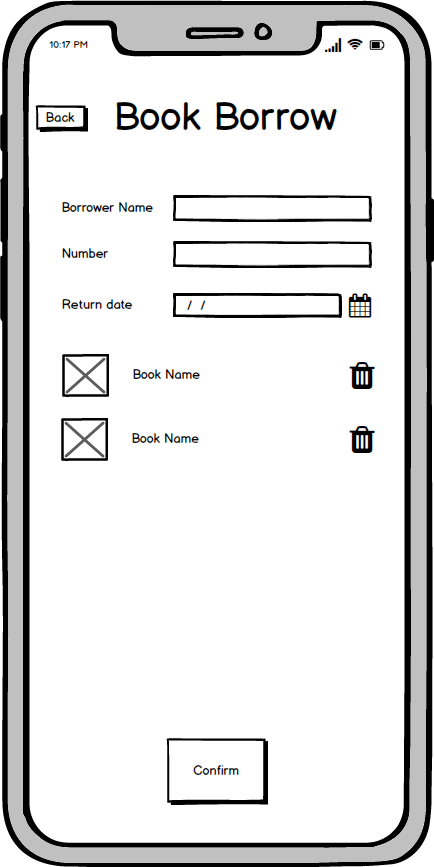
* 1. Chức năng/Thành phần #4: Chỉnh sửa sách
     1. Giao diện người dùng



* + 1. Yêu cầu chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Mô tả** | **Hành động** | **Phản hồi** |
| **Giao diện**  **“Chỉnh sửa sách”** | Giao diện “Chỉnh sửa sách” có:   * Title * Button “Back” * Image “Ảnh” * TextField “Tên sách” * TextField “Tác giả” * Spinner “Trạng thái sách” * Button “Chỉnh sửa” | N/A | N/A |
| **Title** | Hiển thị tên của giao diện “Chỉnh sửa sách” | N/A | N/A |
| **Button “Back”** | Trờ về màn hình chính | Tap on | Chuyển về màn hình “Danh sách sách” |
| **Image**  **“Ảnh”** | Hình ảnh đại diện của sách | Tap on | Yêu cầu người dùng chọn ảnh từ thư viện hoặc chụp ảnh mới |
| **TextField**  **“Tên sách”** | Nhập tên sách  Không được để trống | N/A | N/A |
| **TextField**  **“Tác giả”** | Nhập tên tác giả của sách  Không được để trống | N/A | N/A |
| **Button**  **“Chỉnh sửa”** | Chỉnh sửa thông tin sách | Tap on | Yêu cầu hệ thống chỉnh sửa thông tin của sấch  Nếu thành công, hiển thị thông báo và chuyển về giao diện “Danh sách sách”  Nếu thất bại, hiển thị thông báo lỗi |
| N/A | Vô hiệu hóa nếu bất kì thông tin nào không hợp lệ |

* 1. Chức năng/Thành phần #5: Mượn sách
     1. Giao diện người dùng



* + 1. Yêu cầu chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Mô tả** | **Hành động** | **Phản hồi** |
| **Giao diện**  **“Thông tin nhân viên”** | Giao diện “Thông tin nhân viên” có:   * Title * Button “Back” * TextField “Người mượn” * TextField “CMND” * DatePicker “Ngày trả” * ListView “Danh sách sách” * Button “Xác nhận” | N/A | N/A |
| **Title** | Hiển thị tên của giao diện “Mượn sách” | N/A | N/A |
| **Button**  **“Back”** | Trở về màn hình chính | Tap on | Chuyển đến màn hình “Danh sách sách” |
| **TextField**  **“Người mượn”** | Nhập tên người mượn sách | N/A | N/A |
| **TextField**  **“CMND”** | Nhập CMND của người mượn sách | N/A | N/A |
| **DatePicker**  **“Ngày trả”** | Hiển thị ngày trả sách  Mặc định là 7 ngày tính từ thời điểm xác nhận mượn sách | N/A | N/A |
| **ListView**  **“Danh sách sách”** | Hiển thị danh sách sách  Mỗi item tương ứng với thông tin 1 quyển sách | N/A | N/A |
| **Item**  **“Sách”** | Hiển thị thông tin nhân viên:   * Checkbox “Selected” * Image “Ảnh” * Label “Tên sách” * Button “Xóa” | Tap on | Check checkbox “Selected” |
| **Image**  **“Ảnh”** | Hiển thị hình ảnh đại diện của sách | N/A | N/A |
| **Label**  **“Tên sách”** | Hiển thị tên của quyển sách | N/A | N/A |
| **Button**  **“Xóa”** | Xóa sách khỏi danh sách | Tap on | Hiển thị hộp thoại xác nhận xóa sách  Tap “OK” để xóa sách khỏi danh sách  Nếu thất bại, hiển thị thông báo lỗi |
| **Button**  **“Xác nhận”** | Xác nhận cho mượn sách | Tap on | Yêu cầu hệ thống cập nhật trạng thái sách  Thông báo khi cập nhật thành công hoặc thông báo lỗi khi thất bại  Nếu thành công, chuyển về màn hinh “Danh sách sách” |

1. TÌM HIỂU CÁC CHỨC NĂNG CƠ BẢN CỦA ES6:
   1. Nguyễn Phan Huy
      1. Class

class Point {

    constructor (x, y) {

        this.x = x;

        this.y = y;

    }

}

class Square extends Point {

    constructor (id, x, y, length) {

        super(x, y)

        this.id = id

        this.x = x

        this.y = y

        this.length = length

    }

    isInside(point) {

        if (point.x >= this.x

            && point.x <= (this.x + this.length)

            && point.y >= this.y

            && point.y <= (this.y + this.length)) {

                return true;

            }

        return false;

    }

}

let sqr = new Square(0, 1, 3, 2)

console.log(sqr)

let pts = new Point(0, 0)

console.log(sqr.isInside(pts))

pts = new Point(99, 3)

console.log(sqr.isInside(pts))

* + 1. Var – Let – Const

const A = "Const is unchangeable variable";

// A = 2 => Uncaught TypeError: Assignment to constant variable

var x = 'global'

let y = 'global'

console.log(' ', x) // => global

// console.lot(this.y) => undefined

let name = 'Lam Pham';

let greeting = `I'm ${name}`;

console.log(greeting);

function dumb() {

    var x = 1

    if (true) {

        var x = 2

        console.log(x) // => 2

    }

    console.log(x) // => still 2

}

dumb()

function anotherDumb() {

    let x = 1

    if (true) {

        let x = 2

        console.log(x) // => 2

    }

    console.log(x) // => 1

}

anotherDumb()

function dumbVarLoop() {

    for (var i = 0; i < 3; ++i) {

        setTimeout(() => console.log(i, " "), 1000) // Arrow Function

    }

    // output: 3 3 3

}

dumbVarLoop()

function dumbLetLoop() {

    for (let i = 0; i < 3; ++i) {

        setTimeout(() => console.log(i, " "), 1000)

    }

    // output: 0 1 2

}

dumbLetLoop()

1. COMPONENT
   1. Nguyễn Phan Huy (Flatlist, Button, Navigation
      1. Prop và style

import React, { Component } from 'react';

import { Text, View } from 'react-native';

class Blink extends Component {

  componentDidMount(){

    // Toggle the state every second

    setInterval(() => (

      this.setState(previousState => (

        { isShowingText: !previousState.isShowingText }

      ))

    ), 1000);

  }

  //state object

  state = { isShowingText: true };

  render() {

    if (!this.state.isShowingText) {

      return null;

    }

    return (

      <Text>{this.props.text}</Text>

    );

  }

}

export default class BlinkApp extends Component {

  render() {

    return (

      <View>

        <Blink text='I love to blink' />

        <Blink text='Yes blinking is so great' />

        <Blink text='Why did they ever take this out of HTML' />

        <Blink text='Look at me look at me look at me' />

      </View>

    );

  }

}

* + 1. TouchableOpacity

import React, { Component } from 'react'

import { StyleSheet, Text, View, TouchableOpacity } from 'react-native'

import Book from './Book'

export default class DSSanPham extends Component {

    render() {

        return (

            <View style={{ alignItems: 'center', top: 56 }}>

                <Book

                    productId="SP01"

                    productName="Áo"

                    img="https://facebook.github.io/react-native/img/tiny\_logo.png"

                ></Book>

                <Book

                    productId='SP01'

                    productName='Áo'

                    img='https://facebook.github.io/react-native/img/tiny\_logo.png'

                ></Book>

                <View style={{ alignItems: 'flex-end' }}>

                    <TouchableOpacity

                        style={styles.buttonContainer}

                        onPress={() => this.props.navigation.navigate('Insert')} >

                        <Text style={styles.buttonText}>+</Text>

                    </TouchableOpacity>

                </View>

            </View>

        );

    }

}

const styles = StyleSheet.create({

    buttonContainer: {

        backgroundColor: '#6193ab',

        padding: 15,

        borderRadius: 50,

        height: 50,

        width: 50,

        marginBottom: 20,

        alignSelf: 'flex-end'

    },

    buttonText: {

        textAlign: 'center',

        color: '#FFFFFF',

        fontWeight: 'bold',

        fontSize: 15

    },

})

* + 1. Flex

import React, { Component } from 'react';

import { View } from 'react-native';

export default class FixedDimensionsBasics extends Component {

  render() {

    return (

      <View>

        <View style={{width: 50, height: 50, backgroundColor: 'powderblue'}} />

        <View style={{width: 100, height: 100, backgroundColor: 'skyblue'}} />

        <View style={{width: 150, height: 150, backgroundColor: 'steelblue'}} />

      </View>

    );

  }

}

* + 1. FetchExample.js

import React from 'react';

import {

  StyleSheet,

  TouchableOpacity,

  FlatList,

  ActivityIndicator,

  Text,

  View,

  Alert,

} from 'react-native';

import Book from './Book';

export default class FetchExample extends React.Component {

  constructor(props) {

    super(props);

    this.state = {isLoading: true};

  }

  state = {

    search: '',

    typing: false,

    typingTimeout: 0,

  };

  componentDidMount() {

    //Here is the Trick

    const {navigation} = this.props;

    //Adding an event listner om focus

    //So whenever the screen will have focus it will set the state to zero

    this.focusListener = navigation.addListener('didFocus', () => {

      fetch('http://10.0.2.2:8080/webservice/Select\_SanPham.php')

        .then(response => response.json())

        .then(responseJson => {

          this.setState(

            {

              isLoading: false,

              dataSource: responseJson,

            },

            function() {},

          );

        })

        .catch(error => {

          console.error(error);

        });

    });

  }

  componentWillUnmount() {

    // Remove the event listener before removing the screen from the stack

    this.focusListener.remove();

  }

  render() {

    if (this.state.isLoading) {

      return (

        <View style={{flex: 1, padding: 20}}>

          <ActivityIndicator />

        </View>

      );

    }

    return (

      <View style={{flex: 1, paddingTop: 20}}>

        <FlatList

          data={this.state.dataSource}

          renderItem={({item}) => (

            <TouchableOpacity

              onPress={() =>

                this.props.navigation.navigate('Modify', {

                  bookId: item.bookId,

                  bookName: item.bookName,

                  bookAmount: item.bookAmount,

                })

              }>

              <Book

                bookId={item.bookId}

                bookName={item.bookName}

                bookImage={item.bookImage}

              />

            </TouchableOpacity>

          )}

        />

        <View style={{alignItems: 'flex-end'}}>

          <TouchableOpacity

            style={styles.buttonContainer}

            onPress={() => this.props.navigation.navigate('Insert')}>

            <Text style={styles.buttonText}>+</Text>

          </TouchableOpacity>

        </View>

      </View>

    );

  }

}

const styles = StyleSheet.create({

  buttonContainer: {

    backgroundColor: '#6193ab',

    padding: 15,

    borderRadius: 50,

    height: 50,

    width: 50,

    marginBottom: 20,

    marginEnd: 20,

    alignSelf: 'flex-end',

  },

  buttonText: {

    textAlign: 'center',

    color: '#FFFFFF',

    fontWeight: 'bold',

    fontSize: 15,

  },

});

1. API
   1. Nguyễn Phan Huy
      1. WebService
         1. DBConfig.php

<?php

//Define your host here.

$HostName ="localhost";

//Define your database name here.

$DatabaseName = "qlsanpham";

//Define your database username here.

$HostUser = "root";

//Define your database password here.

$HostPass = "";

?>

* + - 1. Insert\_SanPham.php

<?php

include 'DBConfig.php';

$conn = mysqli\_connect($HostName, $HostUser, $HostPass, $DatabaseName);

if (!$conn) {

    die("Connection failed: " . mysqli\_connect\_error());

}

$json = file\_get\_contents('php://input');

$obj = json\_decode($json, true);

$bookId = $obj["bookId"];

$bookName = $obj["bookName"];

$bookAmount = $obj["bookAmount"];

$sql = "INSERT INTO tbsanpham (`bookId`, `bookName`, `bookAmount`)

 VALUES ('$bookId','$bookName','$bookAmount')";

if (mysqli\_query($conn, $sql)) {

    $MSG = 'Successfully';

    $json = json\_encode($MSG);

    echo $json;

} else {

    echo 'Wrong';

}

mysqli\_close($conn);

* + - 1. Modify\_SanPham.php

<?php

include 'DBConfig.php';

$conn = mysqli\_connect($HostName, $HostUser, $HostPass, $DatabaseName);

if (!$conn) {

    die("Connection failed: " . mysqli\_connect\_error());

}

$json = file\_get\_contents('php://input');

$obj = json\_decode($json, true);

$bookId = $obj["bookId"];

$bookName = $obj["bookName"];

$bookAmount = $obj["bookAmount"];

$sql = "UPDATE tbsanpham SET `bookName`='$bookName', `bookAmount`='$bookAmount' WHERE `bookId`='$bookId'";

if (mysqli\_query($conn, $sql)) {

    $MSG = 'Successfully';

    $json = json\_encode($MSG);

    echo $json;

} else {

    echo 'Wrong';

}

mysqli\_close($conn);

* + - 1. login.php

<?php

    include 'DBConfig.php';

    $conn = mysqli\_connect($HostName,$HostUser,$HostPass,$DatabaseName);

    $json = file\_get\_contents('php://input');

    $obj = json\_decode($json, true);

    // check connection

    if (mysqli\_connect\_errno()) {

        // echo json\_encode("Failed to connect to MySQL: " . mysqli\_connect\_error());

        echo json\_encode('try again');

    }

    $username = $obj['username'];

    $password = $obj['password'];

    $sql = "SELECT \* FROM account WHERE username='$username' AND password ='$password'";

    $result = mysqli\_query($conn,$sql);

    $row = mysqli\_fetch\_array($result);

    if ($row['username'] != "") {

        if ($row['username'] == $username && $row['password'] == $password){

            echo json\_encode("true");

        }

        else {

            echo json\_encode("false");

        }

    }

    else {

        echo json\_encode('try again');

    }

* + - 1. Select\_SanPham.php

<?php

include 'DBConfig.php';

$conn = new mysqli($HostName, $HostUser, $HostPass, $DatabaseName);

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$sql = "SELECT \* FROM tbsanpham";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

    while ($row[] = $result->fetch\_assoc()) {

        $tempt = $row;

        $json = json\_encode($tempt);

    }

} else {

    echo "No Results Found.";

}

echo $json;

$conn->close();

* + - 1. Delete\_SanPham.php

<?php

include 'DBConfig.php';

$conn = new mysqli($HostName, $HostUser, $HostPass, $DatabaseName);

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$json = file\_get\_contents('php://input');

$obj = json\_decode($json, true);

$bookId = $obj["bookId"];

$sql = "DELETE FROM `tbsanpham` WHERE `bookId`='$bookId'";

$result = $conn->query($sql);

if (mysqli\_query($conn, $sql)) {

    $MSG = 'Successfully';

    $json = json\_encode($MSG);

    echo $json;

} else {

    echo 'Wrong';

}

$conn->close();

* + 1. FetchExample:
* Lấy dữ liệu từ WebService
* Đẩy dữ liệu vào FlatList
* Chuyển dữ liệu vào từng view tương ứng sau đó hiển thị lên list

import React from 'react';

import {

  StyleSheet,

  TouchableOpacity,

  FlatList,

  ActivityIndicator,

  Text,

  View,

  Alert,

} from 'react-native';

import Book from './Book';

export default class FetchExample extends React.Component {

  constructor(props) {

    super(props);

    this.state = {isLoading: true};

  }

  state = {

    search: '',

    typing: false,

    typingTimeout: 0,

  };

  componentDidMount() {

    //Here is the Trick

    const {navigation} = this.props;

    //Adding an event listner om focus

    //So whenever the screen will have focus it will set the state to zero

    this.focusListener = navigation.addListener('didFocus', () => {

      fetch('http://10.0.2.2:8080/webservice/Select\_SanPham.php')

        .then(response => response.json())

        .then(responseJson => {

          this.setState(

            {

              isLoading: false,

              dataSource: responseJson,

            },

            function() {},

          );

        })

        .catch(error => {

          console.error(error);

        });

    });

  }

  componentWillUnmount() {

    // Remove the event listener before removing the screen from the stack

    this.focusListener.remove();

  }

  render() {

    if (this.state.isLoading) {

      return (

        <View style={{flex: 1, padding: 20}}>

          <ActivityIndicator />

        </View>

      );

    }

    return (

      <View style={{flex: 1, paddingTop: 20}}>

        <FlatList

          data={this.state.dataSource}

          renderItem={({item}) => (

            <TouchableOpacity

              onPress={() =>

                this.props.navigation.navigate('Modify', {

                  bookId: item.bookId,

                  bookName: item.bookName,

                  bookAmount: item.bookAmount,

                })

              }>

              <Book

                bookId={item.bookId}

                bookName={item.bookName}

                bookImage={item.bookImage}

              />

            </TouchableOpacity>

          )}

        />

        <View style={{alignItems: 'flex-end'}}>

          <TouchableOpacity

            style={styles.buttonContainer}

            onPress={() => this.props.navigation.navigate('Insert')}>

            <Text style={styles.buttonText}>+</Text>

          </TouchableOpacity>

        </View>

      </View>

    );

  }

}

const styles = StyleSheet.create({

  buttonContainer: {

    backgroundColor: '#6193ab',

    padding: 15,

    borderRadius: 50,

    height: 50,

    width: 50,

    marginBottom: 20,

    marginEnd: 20,

    alignSelf: 'flex-end',

  },

  buttonText: {

    textAlign: 'center',

    color: '#FFFFFF',

    fontWeight: 'bold',

    fontSize: 15,

  },

});

1. PROJECT: Ứng dụng quản lý thư viện:

Ứng dụng quản lý những đầu sách có trong thư viện  
Để việc quản lý sách trở nên dễ dàng và nhanh chóng hơn:  
- Hiển thị danh sách  
- Thêm / Xem / Sửa sách  
- Đăng nhập, đăng kí  
- Mượn / Trả

* 1. App.js: Navigator

import React from 'react';

import { createStackNavigator } from 'react-navigation-stack'; // Version can be specified in package.json

import { createAppContainer } from 'react-navigation'; // Version can be specified in package.json

import LoginScreen from "./LoginScreen";

import List from "./FetchExample";

import Insert from "./InsertBook";

import Modify from "./ModifyBook";

const RootStack = createStackNavigator(

  {

    Login: LoginScreen,

    List: List,

    Insert: Insert,

    Modify: Modify

  },

  {

    initialRouteName: 'Login',

  }

);

const AppContainer = createAppContainer(RootStack);

export default class App extends React.Component {

  render() {

    return <AppContainer />;

  }

}

* 1. Book.js: Hiển thị thông tin của từng quyển sách trong danh sách

/\*\*

 \* Sample React Native App

 \* https://github.com/facebook/react-native

 \* @flow

 \*/

import React, {Component} from 'react';

import {TouchableOpacity, StyleSheet, Text, View, Image, Alert} from 'react-native';

import {Colors} from 'react-native/Libraries/NewAppScreen';

var icon;

export default class Book extends Component {

  constructor(props) {

    super(props);

  }

  render() {

    return (

      <View

        style={{

          flex: 1,

          flexDirection: 'column',

        }}>

        <View style={styles.book}>

          <Image

            style={styles.bookImage}

            source={{uri: this.props.bookImage}}

          />

          <View style={styles.containerContent}>

            <Text> Mã sách : {this.props.bookId} </Text>

            <Text> Tên sách : {this.props.bookName} </Text>

          </View>

        </View>

      </View>

    );

  }

}

const styles = StyleSheet.create({

  book: {

    flex: 1,

    flexDirection: 'row',

  },

  bookDescription: {

    marginBottom: 5,

    backgroundColor: 'mediumseagreen',

  },

  bookImage: {

    width: 100,

    height: 100,

    margin: 5,

  },

  buttonText: {

    textAlign: 'center',

    color: '#FFFFFF',

    fontWeight: 'bold',

    marginHorizontal: 15,

    fontSize: 15,

  },

  buttonContainer: {

    backgroundColor: 'red',

    margin: 5,

    justifyContent: 'center',

  },

  containerContent: {

    flex: 1,

    flexDirection: 'column',

    justifyContent: 'center',

  },

});

* 1. InsertBook.js

import React, {Component} from 'react';

import {

  StyleSheet,

  View,

  TextInput,

  Text,

  ActivityIndicator,

  TouchableOpacity,

} from 'react-native';

export default class InsertBook extends React.Component {

  constructor() {

    super();

    this.state = {

      bookId: '',

      bookName: '',

      bookAmount: 1,

      ActivityIndicator\_Loading: false,

    };

  }

  Insert\_Data\_Into\_MySQL = () => {

    const {navigation} = this.props;

    this.setState({ActivityIndicator\_Loading: true}, () => {

      fetch('http://10.0.2.2:8080/webservice/Insert\_SanPham.php', {

        method: 'POST',

        headers: {

          Accept: 'application/json',

          'Content-Type': 'application/json',

        },

        body: JSON.stringify({

          bookId: this.state.bookId,

          bookName: this.state.bookName,

          bookAmount: this.state.bookAmount,

        }),

      })

        .then(response => response.json())

        .then(responseJsonFromServer => {

          alert(responseJsonFromServer);

          this.setState({ActivityIndicator\_Loading: false});

          this.props.navigation.goBack();

        })

        .catch(error => {

          console.error(error);

          this.setState({ActivityIndicator\_Loading: false});

        });

    });

  };

  render() {

    const {navigation} = this.props;

    return (

      <View style={styles.MainContainer}>

        <TextInput

          placeholder="Nhập mã sách"

          style={styles.TextInputStyleClass}

          underlineColorAndroid="transparent"

          onChangeText={TextInputText => this.setState({bookId: TextInputText})}

        />

        <TextInput

          placeholder="Nhập tên sách"

          style={styles.TextInputStyleClass}

          underlineColorAndroid="transparent"

          onChangeText={TextInputText =>

            this.setState({bookName: TextInputText})

          }

        />

        <TextInput

          keyboardType="number-pad"

          placeholder="Nhập số lượng"

          style={styles.TextInputStyleClass}

          underlineColorAndroid="transparent"

          onChangeText={TextInputText =>

            this.setState({bookAmount: TextInputText})

          }

        />

        <TouchableOpacity

          activeOpacity={0.5}

          style={styles.TouchableOpacityStyle}

          onPress={this.Insert\_Data\_Into\_MySQL}>

          <Text style={styles.TextStyle}>Thêm sách</Text>

        </TouchableOpacity>

        {this.state.ActivityIndicator\_Loading ? (

          <ActivityIndicator

            color="#009688"

            size="large"

            style={styles.ActivityIndicatorStyle}

          />

        ) : null}

      </View>

    );

  }

}

const styles = StyleSheet.create({

  MainContainer: {

    flex: 1,

    justifyContent: 'center',

    alignItems: 'center',

    margin: 20,

  },

  TextInputStyleClass: {

    textAlign: 'center',

    height: 40,

    backgroundColor: '#fff',

    borderWidth: 1,

    borderColor: '#009688',

    borderRadius: 7,

    marginBottom: 10,

    width: '95%',

  },

  TouchableOpacityStyle: {

    paddingTop: 10,

    paddingBottom: 10,

    backgroundColor: '#009688',

    marginBottom: 20,

    width: '90%',

  },

  TextStyle: {

    color: '#fff',

    textAlign: 'center',

    fontSize: 18,

  },

  ActivityIndicatorStyle: {

    position: 'absolute',

    left: 0,

    right: 0,

    top: 0,

    bottom: 0,

    alignItems: 'center',

    justifyContent: 'center',

  },

});

* 1. ModifyBook.js

import React, {Component} from 'react';

import {

  StyleSheet,

  View,

  TextInput,

  Text,

  ActivityIndicator,

  TouchableOpacity,

  Alert,

} from 'react-native';

export default class ModifyBook extends React.Component {

  constructor(props) {

    super(props);

    this.state = {

      bookId: props.navigation.getParam('bookId', 'NO-ID'),

      bookName: props.navigation.getParam('bookName', 'NO-NAME'),

      bookAmount: props.navigation.getParam('bookAmount', 1),

      ActivityIndicator\_Loading: false,

    };

  }

  Update\_Data\_In\_MySQL = () => {

    this.setState({ActivityIndicator\_Loading: true}, () => {

      fetch('http://10.0.2.2:8080/webservice/Modify\_SanPham.php', {

        method: 'POST',

        headers: {

          Accept: 'application/json',

          'Content-Type': 'application/json',

        },

        body: JSON.stringify({

          bookId: this.state.bookId,

          bookName: this.state.bookName,

          bookAmount: this.state.bookAmount,

        }),

      })

        .then(response => response.json())

        .then(responseJsonFromServer => {

          alert(responseJsonFromServer);

          this.setState({ActivityIndicator\_Loading: false});

        })

        .catch(error => {

          console.error(error);

          this.setState({ActivityIndicator\_Loading: false});

        });

    });

  };

  DELETE\_Data\_In\_MySQL = () => {

    Alert.alert(

      'Alert',

      'Are you sure you want to delete?',

      [

        {

          text: 'No',

          onPress: () => console.log('Cancel pressed'),

          style: 'cancel',

        },

        {

          text: 'Yes',

          onPress: () => {

            fetch('http://10.0.2.2:8080/webservice/Delete\_SanPham.php', {

              method: 'POST',

              headers: {

                Accept: 'application/json',

                'Content-Type': 'application/json',

              },

              body: JSON.stringify({

                bookId: this.state.bookId,

              }),

            })

              .then(response => response.json())

              .then(responseJsonFromServer => {

                alert(responseJsonFromServer);

                this.setState({ActivityIndicator\_Loading: false});

                this.props.navigation.goBack();

              })

              .catch(error => {

                console.error(error);

              });

          },

        },

      ],

      {cancelable: true},

    );

  };

  render() {

    const {navigation} = this.props;

    return (

      <View style={styles.MainContainer}>

        <Text style={{marginBottom: 10}}>Mã sách : {this.state.bookId}</Text>

        <TextInput

          placeholder={this.state.bookName}

          style={styles.TextInputStyleClass}

          underlineColorAndroid="transparent"

          onChangeText={TextInputText => {

            this.setState({bookName: TextInputText});

          }}

        />

        <TextInput

          keyboardType="number-pad"

          placeholder={this.state.bookAmount}

          style={styles.TextInputStyleClass}

          underlineColorAndroid="transparent"

          onChangeText={TextInputText =>

            this.setState({bookAmount: TextInputText})

          }

        />

        <TouchableOpacity

          activeOpacity={0.5}

          style={styles.UpdateButtonStyle}

          onPress={this.Update\_Data\_In\_MySQL}>

          <Text style={styles.TextStyle}>Cập nhật</Text>

        </TouchableOpacity>

        <TouchableOpacity

          activeOpacity={0.5}

          style={styles.DeleteButtonStyle}

          onPress={this.DELETE\_Data\_In\_MySQL}>

          <Text style={styles.TextStyle}>Delete</Text>

        </TouchableOpacity>

        {this.state.ActivityIndicator\_Loading ? (

          <ActivityIndicator

            color="#009688"

            size="large"

            style={styles.ActivityIndicatorStyle}

          />

        ) : null}

      </View>

    );

  }

}

const styles = StyleSheet.create({

  MainContainer: {

    flex: 1,

    justifyContent: 'center',

    alignItems: 'center',

    margin: 20,

  },

  TextInputStyleClass: {

    textAlign: 'center',

    height: 40,

    backgroundColor: '#fff',

    borderWidth: 1,

    borderColor: '#009688',

    borderRadius: 7,

    marginBottom: 10,

    width: '95%',

  },

  UpdateButtonStyle: {

    paddingTop: 10,

    paddingBottom: 10,

    backgroundColor: '#009688',

    marginBottom: 20,

    width: '90%',

  },

  DeleteButtonStyle: {

    paddingTop: 10,

    paddingBottom: 10,

    backgroundColor: 'red',

    marginBottom: 20,

    width: '90%',

  },

  TextStyle: {

    color: '#fff',

    textAlign: 'center',

    fontSize: 18,

  },

  ActivityIndicatorStyle: {

    position: 'absolute',

    left: 0,

    right: 0,

    top: 0,

    bottom: 0,

    alignItems: 'center',

    justifyContent: 'center',

  },

});

* 1. FetchExample.js

import React from 'react';

import {

  StyleSheet,

  TouchableOpacity,

  FlatList,

  ActivityIndicator,

  Text,

  View,

  Alert,

} from 'react-native';

import Book from './Book';

export default class FetchExample extends React.Component {

  constructor(props) {

    super(props);

    this.state = {isLoading: true};

  }

  state = {

    search: '',

    typing: false,

    typingTimeout: 0,

  };

  componentDidMount() {

    //Here is the Trick

    const {navigation} = this.props;

    //Adding an event listner om focus

    //So whenever the screen will have focus it will set the state to zero

    this.focusListener = navigation.addListener('didFocus', () => {

      fetch('http://10.0.2.2:8080/webservice/Select\_SanPham.php')

        .then(response => response.json())

        .then(responseJson => {

          this.setState(

            {

              isLoading: false,

              dataSource: responseJson,

            },

            function() {},

          );

        })

        .catch(error => {

          console.error(error);

        });

    });

  }

  componentWillUnmount() {

    // Remove the event listener before removing the screen from the stack

    this.focusListener.remove();

  }

  render() {

    if (this.state.isLoading) {

      return (

        <View style={{flex: 1, padding: 20}}>

          <ActivityIndicator />

        </View>

      );

    }

    return (

      <View style={{flex: 1, paddingTop: 20}}>

        <FlatList

          data={this.state.dataSource}

          renderItem={({item}) => (

            <TouchableOpacity

              onPress={() =>

                this.props.navigation.navigate('Modify', {

                  bookId: item.bookId,

                  bookName: item.bookName,

                  bookAmount: item.bookAmount,

                })

              }>

              <Book

                bookId={item.bookId}

                bookName={item.bookName}

                bookImage={item.bookImage}

              />

            </TouchableOpacity>

          )}

        />

        <View style={{alignItems: 'flex-end'}}>

          <TouchableOpacity

            style={styles.buttonContainer}

            onPress={() => this.props.navigation.navigate('Insert')}>

            <Text style={styles.buttonText}>+</Text>

          </TouchableOpacity>

        </View>

      </View>

    );

  }

}

const styles = StyleSheet.create({

  buttonContainer: {

    backgroundColor: '#6193ab',

    padding: 15,

    borderRadius: 50,

    height: 50,

    width: 50,

    marginBottom: 20,

    marginEnd: 20,

    alignSelf: 'flex-end',

  },

  buttonText: {

    textAlign: 'center',

    color: '#FFFFFF',

    fontWeight: 'bold',

    fontSize: 15,

  },

});

* 1. LoginScreen.js

import React, {Component} from 'react';

import {

  StyleSheet,

  Text,

  View,

  Image,

  TouchableWithoutFeedback,

  StatusBar,

  TextInput,

  SafeAreaView,

  Keyboard,

  TouchableOpacity,

  KeyboardAvoidingView,

  Button,

  ToolbarAndroid,

} from 'react-native';

export default class LoginScreen extends React.Component {

  static navigationOptions = {

    header: null,

  };

  constructor(props) {

    super(props);

    this.state = {

      password: '',

      username: '',

    };

  }

  render() {

    return (

      <SafeAreaView style={styles.container}>

        <StatusBar barStyle="light-content" backgroundColor="#1c313a" />

        <KeyboardAvoidingView style={styles.container} enabled>

          <TouchableWithoutFeedback

            style={styles.container}

            onPress={Keyboard.dismiss}>

            <View style={styles.logoContainer}>

              <View style={styles.logoContainer}>

                <Image

                  style={styles.logo}

                  source={{

                    uri:

                      'https://cdn0.iconfinder.com/data/icons/BrushedMetalIcons\_meBaze/512/Apple-03.png',

                  }}></Image>

                <Text style={styles.title}>Login</Text>

              </View>

              <View style={styles.infoContainer}>

                <TextInput

                  style={styles.input}

                  value={this.state.username}

                  onChangeText={username => this.setState({username})}

                  placeholder="Enter username/email"

                  placeholderTextColor="#455a64"

                  keyboardType="email-address"

                  returnKeyType="next"

                  autoCorrect={false}

                  onSubmitEditing={() => this.refs.txtPassword.focus()}

                />

                <TextInput

                  style={styles.input}

                  value={this.state.password}

                  onChangeText={password => this.setState({password})}

                  placeholder="Enter password"

                  placeholderTextColor="#455a64"

                  returnKeyType="go"

                  secureTextEntry

                  autoCorrect={false}

                  ref={'txtPassword'}

                />

                <TouchableOpacity

                  style={styles.buttonContainer}

                  onPress={() => this.login()}>

                  <Text style={styles.buttonText}>SIGN IN</Text>

                </TouchableOpacity>

              </View>

            </View>

          </TouchableWithoutFeedback>

        </KeyboardAvoidingView>

      </SafeAreaView>

    );

  }

  login() {

    fetch('http://10.0.2.2:8080/webservice/login.php', {

      method: 'POST',

      headers: {

        Accept: 'application/json',

        'Content-Type': 'application/json',

      },

      body: JSON.stringify({

        username: this.state.username,

        password: this.state.password,

      }),

    })

      .then(response => response.json())

      .then(responseJson => {

        if (responseJson == 'true') {

          alert('Successfully Login');

          // navigate

          this.props.navigation.navigate('List');

        } else {

          alert('Login Failed');

        }

      })

      .catch(error => {

        console.log(error);

      });

  }

}

const styles = StyleSheet.create({

  container: {

    flex: 2,

    // backgroundColor: 'rgb(32, 53, 70)',

    backgroundColor: '#455a64',

    flexDirection: 'column',

  },

  logoContainer: {

    alignItems: 'center',

    justifyContent: 'center',

    flex: 1,

    backgroundColor: '#455a64',

  },

  logo: {

    width: 128,

    height: 120,

  },

  title: {

    color: '#FFFFFF',

    fontSize: 20,

    textAlign: 'center',

    marginTop: 5,

    opacity: 0.9,

    marginBottom: 150,

  },

  infoContainer: {

    marginBottom: 70,

    position: 'absolute',

    left: 0,

    right: 0,

    bottom: 0,

    height: 200,

    padding: 20,

    // backgroundColor: 'red'

  },

  input: {

    marginBottom: 20,

    height: 40,

    color: '#775da9',

    borderColor: '#FFFFFF',

    backgroundColor: '#FFFFFF',

    borderWidth: 2,

    borderRadius: 20,

    fontSize: 15,

  },

  buttonContainer: {

    backgroundColor: '#6193ab',

    paddingVertical: 15,

    borderRadius: 30,

    height: 50,

    marginBottom: 20,

  },

  buttonText: {

    textAlign: 'center',

    color: '#FFFFFF',

    fontWeight: 'bold',

    fontSize: 15,

  },

  hello: {

    marginTop: 450,

  },

});