

TRƯỜNG CAO ĐẲNG CÔNG NGHỆ THỦ ĐỨC
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO KẾT THÚC MÔN HỌC
Lập trình di động 3

ỨNG DỤNG ĐỌC TIN TỨC

Giảng viên hướng dẫn:
Sinh viên thực hiện:

TRƯỜNG BÁ THÁI
Đỗ Minh Văn
Nguyễn Thị Mỹ Ái
Đặng Phương Linh
Lê Minh Vũ

Ngành: Công nghệ thông tin

Lớp: CD16TT2 Khoá: 2016

Tp. Hồ Chí Minh, ngày 27 tháng 12 năm 2018

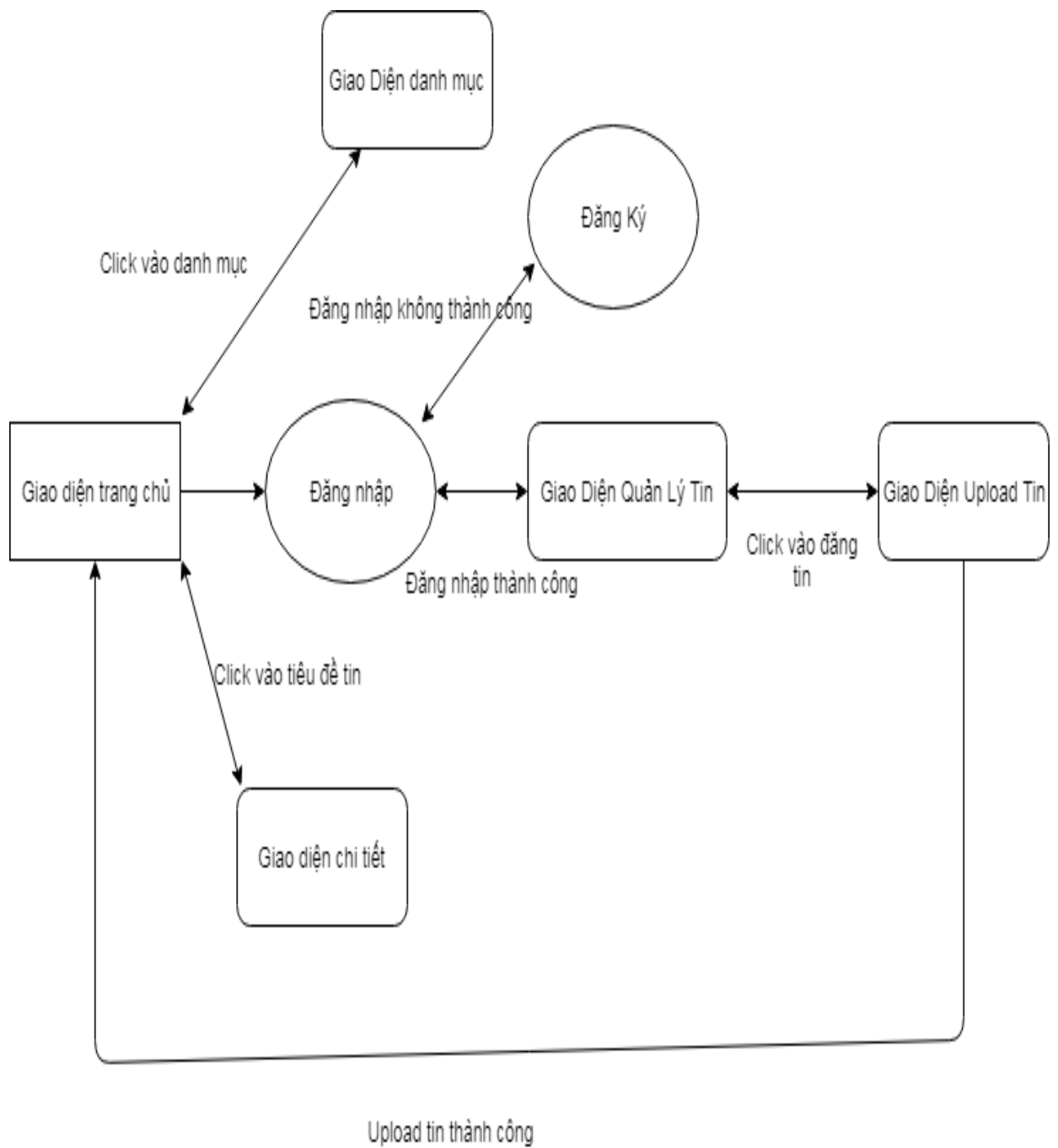
NHẬT KÝ HOẠT ĐỘNG NHÓM

Stt	Họ và tên	Công việc đã thực hiện	Tự đánh giá	Nhóm đánh giá	Chữ ký
1	Đỗ Minh Văn	Thiết kế giao diện trang chủ, danh mục, chi tiết tin.	8	9	
2	Nguyễn Thị Mỹ Ái	Giao diện quản lý tin, thêm xóa sửa tin.	8	9	
3	Đặng Phương Linh	Màn hình đăng nhập và đăng ký	8	9	
4	Lên Minh Vũ	Màn hình đăng tin	8	9	

MỤC LỤC

DANH MỤC BẢNG BIỂU, HÌNH VẼ, SƠ ĐỒ	4
CHƯƠNG 1. TỔNG QUAN VỀ REACT NATIVE	5
1. Giới thiệu tổng quan về React native.....	5
2. Kiến thức cơ bản về ES6 trong React native	5
3. Component trong react native	9
4. API Trong React Native.....	12
CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG	18
2.1 Phân tích hệ thống.....	18
2.1.1. Feature/Component #1: MyMobile Registration screen	18
2.1.1.1 User Interfaces	18
2.1.1.2. Functional Requirements	24
2.2 Thiết kế hệ thống	27
2.2.1. Sudoku Main Screen	27
2.2.1.1. Screen Shot for Sudoku Main Screen	27
2.2.1.2. Objects and actions for Sudoku Main Screen	29
CHƯƠNG 3. CÀI ĐẶT VÀ KIỂM THỬ.....	40
3.1 Cài đặt	Error! Bookmark not defined.
3.2 Kiểm thử	Error! Bookmark not defined.
CHƯƠNG 4. KẾT QUẢ ĐẠT ĐƯỢC	42
4.1 Kết quả đạt được	42
4.2 Các kết luận và kiến nghị.....	49
PHỤ LỤC.....	Error! Bookmark not defined.
TÀI LIỆU THAM KHẢO	51

DANH MỤC BẢNG BIỂU, HÌNH VẼ, SƠ ĐỒ



CHƯƠNG 1. TỔNG QUAN VỀ REACT NATIVE

1. Giới thiệu tổng quan về React native

React Native là một framework do công ty công nghệ nổi tiếng Facebook phát triển nhằm mục đích giải quyết bài toán hiệu năng của Hybrid và bài toán chi phí khi mà phải viết nhiều loại ngôn ngữ native cho từng nền tảng di động.

Chúng ta sẽ build được ứng dụng React Native, và chúng ta cũng có thể build ứng dụng đó một cách đa nền tảng (multi-platform) chứ không phải là một “mobile web app”, không phải là “HTML5 app”, và cũng không phải là một “hybrid app” hay cũng không chỉ build trên iOS hay Android mà chúng ta build và chạy được cả hai hệ sinh thái.

React Native đó chính là chúng ta chỉ cần sử dụng JS để phát triển được một ứng dụng di động hoàn chỉnh.

Ưu Điểm:

- Hiệu quả về mặt thời gian khi mà bạn muốn phát triển một ứng dụng nhanh chóng.
- Hiệu năng tương đối ổn định.
- Cộng đồng phát triển mạnh.
- Tiết kiệm tiền.
- Team phát triển nhỏ.
- Ứng dụng tin cậy và ổn định.
- Xây dựng cho nhiều hệ điều hành khác nhau với ít native code nhất.
- Trải nghiệm người dùng tốt hơn là hybrid app.

Nhược Điểm:

- Vẫn đòi hỏi native code.
- Hiệu năng sẽ thấp hơn với app thuần native code.
- Bảo mật không cao do dựa trên JS.
- Quản lý bộ nhớ.
- Khả năng tùy biến cũng không thực sự tốt đối với một vài module.

2. Kiến thức cơ bản về ES6 trong React native

ES6 là phiên bản mới nhất của bộ tiêu chuẩn **ECMAScript** – một bộ đặc tả tiêu chuẩn dành cho Javascript do Hiệp hội các nhà sản xuất máy tính Châu Âu (*European Computer Manufacturers Association – ECMA*) đề xuất.

CÁC CHỨC NĂNG MỚI CỦA ES6

ARROW

Arrow là một dạng viết tắt của các function sử dụng dấu =>, tương tự như trong C#, Java 8,...

```
// Expression bodies
var odds = evens.map(v => v + 1);
var nums = evens.map((v, i) => v + i);
var pairs = evens.map(v => ({even: v, odd: v + 1}));

// Statement bodies
nums.forEach(v => {
  if (v % 5 === 0)
    fives.push(v);
});

// Lexical this
var bob = {
  _name: "Bob",
  _friends: [],
  printFriends() {
    this._friends.forEach(f =>
      console.log(this._name + " knows " + f));
  }
}
```

CLASS

Đối với Javascript truyền thống, để sử dụng khai báo và kế thừa các class, chúng ta phải thiết kế theo hướng sử dụng Prototype (prototype-based OO). Việc khai báo và kế thừa các class trong ES6 dễ hơn rất nhiều với cú pháp gần giống với Java và C++, ngoài ra, class trong ES6 cũng hỗ trợ kế thừa thông qua prototype, các static method, constructor,...

```
class SkinnedMesh extends THREE.Mesh {
  constructor(geometry, materials) {
    super(geometry, materials);

    this.idMatrix = SkinnedMesh.defaultMatrix();
    this.bones = [];
    this.boneMatrices = [];
    //...
  }
  update(camera) {
    //...
    super.update();
  }
}
```

```

    get boneCount() {
        return this.bones.length;
    }
    set matrixType(matrixType) {
        this.idMatrix = SkinnedMesh[matrixType]();
    }
    static defaultMatrix() {
        return new THREE.Matrix4();
    }
}

```

XỬ LÝ CHUỖI

Xử lý chuỗi trong ES6 đã trở nên dễ dàng và tiện dụng hơn rất nhiều, mang hơi hướng của các ngôn ngữ như Python, Perl,... đặc biệt, hỗ trợ chuỗi nhiều dòng, đây có lẽ là một cải tiến khiến rất nhiều người cảm thấy thích thú.

```

// Basic literal string creation
`In JavaScript 'n' is a line-feed.`

// Multiline strings
`In JavaScript this is
not legal.`

// String interpolation
var name = "Bob", time = "today";
`Hello ${name}, how are you ${time}?`

```

GIÁ TRỊ DEFAULT CHO THAM SỐ

Ở phiên bản này, Javascript đã có thể sử dụng các giá trị mặc định cho tham số truyền vào các hàm như những ngôn ngữ lập trình khác như C++, C#.

```

function f(x, y=12) {
    // y = 12 nếu không truyền giá trị cho nó (hoặc truyền undefined)
    return x + y;
}
f(3) == 15

```

TRUYỀN THAM SỐ KHÔNG XÁC ĐỊNH SỐ LƯỢNG

Việc này trước đây có thể thực hiện thông qua biến arguments có trong từng hàm, nhưng giờ đây chúng ta có thể sử dụng nó một cách linh hoạt hơn rất nhiều.

```

function f(x, ...y) {
    // y là một mảng
    return x * y.length;
}

```

```
f(3, "hello", true) == 6
```

TRUYỀN THAM SỐ THÔNG QUA TỪNG PHẦN TỬ CỦA MẢNG

Với kĩ thuật này, bạn có thể truyền một mảng hoặc một đối tượng vào một hàm, các phần tử của mảng/đối tượng này sẽ được tự động truyền vào thành các tham số của hàm đó

```
function f(x, y, z) {  
    return x + y + z;  
}  
// Pass each elem of array as argument  
f(...[1,2,3]) == 6
```

TỪ KHOÁ LET VÀ CONST

const, đúng như tên gọi của nó, là cách khai báo hằng số, một hằng số thì không thể thay đổi giá trị được.

```
const x = 10;  
x = 5; // Lỗi
```

let cũng là một dạng khai báo biến giống như var, tuy nhiên, biến được định nghĩa bằng từ khoá let có phạm vi truy cập khép kín trong khối lệnh chứa nó.

```
function testLet() {  
    // a *không* truy cập được tại đây  
    for( let a = 0; a < 5; a++ ) {  
        // a chỉ truy cập được trong này  
    };  
    // a *không* truy cập được tại đây  
};
```

MODULES

```
// lib/math.js  
export function sum(x, y) {  
    return x + y;  
}  
export var pi = 3.141593;  
// app.js  
import * as math from "lib/math";  
alert("2 $\pi$  = " + math.sum(math.pi, math.pi));  
// otherApp.js  
import {sum, pi} from "lib/math";  
alert("2 $\pi$  = " + sum(pi, pi));
```

MAP, SET, WEAKMAP, WEAKSET

ES6 còn giới thiệu thêm một số kiểu dữ liệu mới để hỗ trợ chúng ta thực hiện các thuật toán phức tạp hơn.


```
// Sets
var s = new Set();
s.add("hello").add("goodbye").add("hello");
s.size === 2;
s.has("hello") === true;

// Maps
var m = new Map();
m.set("hello", 42);
m.set(s, 34);
m.get(s) === 34;

// Weak Maps
var wm = new WeakMap();
wm.set(s, { extra: 42 });
wm.size === undefined

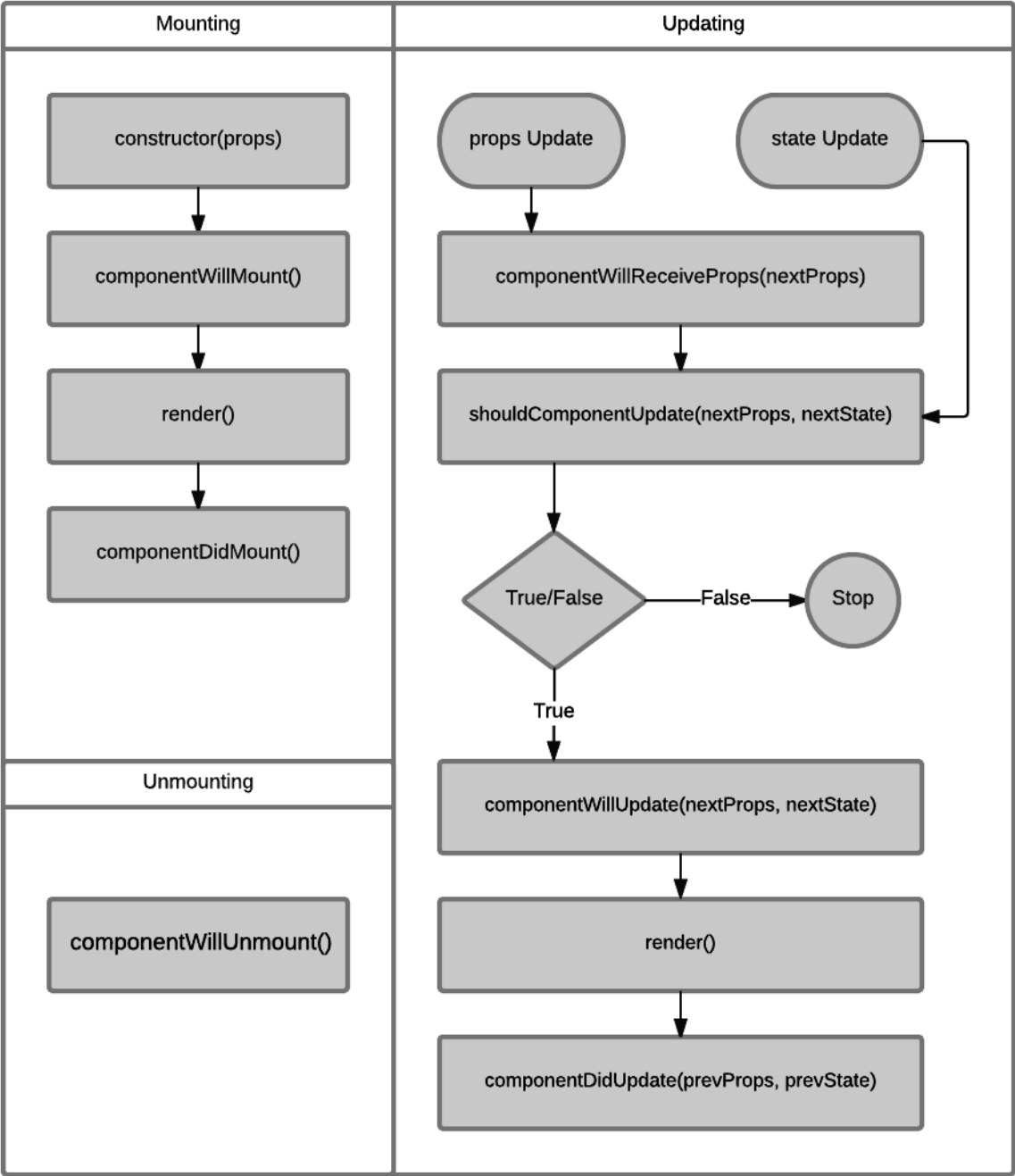
// Weak Sets
var ws = new WeakSet();
ws.add({ data: 42 });
```

3. *Component trong react native*

Trong React, chúng ta xây dựng trang web sử dụng những thành phần (component) nhỏ. Chúng ta có thể tái sử dụng một component ở nhiều nơi, với các trạng thái hoặc các thuộc tính khác nhau, trong một component lại có thể chứa thành phần khác. Mỗi component trong React có một trạng thái riêng, có thể thay đổi, và React sẽ thực hiện cập nhật component dựa trên những thay đổi của trạng thái.

Một số component cơ bản:

Component lifecycle



Mounting:

`constructor(props)`

Hàm khởi tạo này là hàm đầu tiên được gọi khi một Component được tạo ra.

Tại đây thường thì chúng ta sẽ khởi tạo các giá trị ban đầu của state, đọc ra giá trị props được truyền vào hoặc bind các hàm của class.

`componentWillMount()`

Khi hàm này được gọi đến thì component của chúng ta đã được khởi tạo thông qua constructor . Thường thì trong hàm này chúng ta ko nên thay đổi state hãy khởi tạo state trong constructor.

`render()`

Đây là một hàm bắt buộc phải có trong một component class. Hàm này trả về các React element (thường thì tạo bằng JSX).

Tại đây thì nó sẽ phân tích state và props của Component này để vẽ lên DOM. Chúng ta ko nên thay đổi state trong hàm này mà hãy làm ở những chỗ khác.

`componentDidMount()`

Hàm này được gọi sau khi component đã được khởi tạo và vẽ lên DOM. Nếu bạn cần lấy dữ liệu từ server thì đây là nơi thích hợp để request API.

Updating

`componentWillReceiveProps(nextProps)`

Chú ý: Hàm này sẽ bị deprecated trong tương lai.

Hàm này sẽ được gọi đến khi một component (đã qua giai đoạn mounting) nhận được props mới. Nếu bạn cần update state tương ứng với thay đổi của props thì bạn có thể làm tại đây, bằng việc so sánh `this.props` (props hiện tại) và `nextProps` (props mới). Chú ý là nếu bạn gọi `setState` thì hàm này sẽ không được gọi đâu, nhìn tên hàm là biết.

`shouldComponentUpdate()`

Hàm này khá đặc biệt bởi nó quyết định việc các hàm khác của vòng đời có được chạy hay không. Mặc định nó sẽ trả về true, và nếu trả về false thì những hàm như `componentWillUpdate`, `render`, `componentDidUpdate` sẽ không được gọi. Về cơ bản, chúng

ta không nên thực hiện những tiến trình so sánh quá phức tạp trong này, mà thay vào đó hãy sử dụng `PureComponent` (tự động perform shallow equal với `state` và `props`). Trong tương lai, React sẽ chỉ coi hàm này là một gợi ý cho việc có nên update lại component hay không, nghĩa là kể cả bạn có trả về `false` nhưng sẽ không đảm bảo là component không được update.

`componentWillUpdate(nextProps, nextState)`

Hàm này được gọi ngay trước khi render khi nhận được `props` hoặc `state` mới. Tuy chúng ta có cả `state` và `props` mới tại đây, chúng ta không thể gọi `setState` trong hàm này, hoặc làm bất cứ cái gì có thể gây ra update cho component. Trong hầu hết trường hợp thì chúng ta có thể thay hàm này bằng `componentDidUpdate()`.

`render()`

Chúng ta chỉ nhìn thấy thay đổi trong DOM khi hàm này được gọi trong một component. Như đã nói ở trên, một trong những cách để hàm này được gọi đó là sử dụng `setState`. Render trong giai đoạn này chỉ được gọi khi `shouldComponentUpdate()` trả về `true`, và nếu `state` hoặc `props` thay đổi.

`componentDidUpdate(prevProps, prevState)`

Hàm này được gọi ngay khi DOM được update. Bạn có thể truy cập `state` và `props` cũ tại đây, và có thể thực hiện so sánh với `state` và `props` hiện tại để tiến hành request API hay làm gì đó khi có thay đổi.

Unmounting

`componentWillUnmount()`

Hàm này sẽ được gọi mỗi khi một component bị xóa khỏi DOM. Đây là lúc để bạn có thể thực hiện những công việc như clear dữ liệu, tắt các connection. Ví dụ nếu bạn có một component chat real time, bạn có thể khởi tạo connection trong `componentWillMount` và close connection trong `componentWillUnmount`. Nếu bạn không đóng các connection thì nó sẽ vẫn tồn tại ở đây và làm chậm ứng dụng.

4. API Trong React Native

React Native cung cấp module [Fetch API](#) để sử dụng cho việc kết nối network. `Fetch` sẽ rất thân thuộc nếu như bạn đã từng sử dụng `XMLHttpRequest` hoặc các networking APIs trước đây. Bạn có thể sẽ cần phải tham khảo hướng dẫn sử dụng `Fetch` [tại đây](#) để có được thêm các thông tin.

Tạo một HTTP Request

Khi bạn muốn lấy nội dung bằng cách gọi đơn giản nhất từ một URL, rất đơn giản bạn chỉ cần đặt URL đó trong fetch:

```
fetch('https://mywebsite.com/mydata.json')
```

Fetch sẽ có một số tùy chọn tham số để bạn có thể tùy chỉnh HTTP request. Ví như bạn muốn thêm cụ thể một header nào đó và muốn gọi với phương thức POST. Ví dụ dưới đây sẽ cho các bạn thấy một cách đơn giản các tùy chọn:

```
fetch('https://mywebsite.com/endpoint/', {  
  method: 'POST',  
  headers: {  
    'Accept': 'application/json',  
    'Content-Type': 'application/json',  
  },  
  body: JSON.stringify({  
    firstParam: 'yourValue',  
    secondParam: 'yourOtherValue',  
  })  
})
```

Để có được các cái nhìn đầy đủ hơn các bạn có thể đọc thêm ở tài liệu [Fetch Request docs](#).

Xử lý response

Ví dụ ở bên trên đã cho các bạn thấy cách để tạo ra một request. Và trong phần này, khi các bạn muốn xử lý dữ liệu lấy được về thì phần này sẽ nói rõ hơn về cách thức xử lý dữ liệu sau khi được lấy về.

Networking bản chất là một hình thức bất đồng bộ (Lan man một chút, vì sự bất đồng bộ này mà ở Android từ API 11 trở lên hệ điều hành đã ngăn cản việc chạy Network trên main thread để ngăn cản độ trễ của chương trình trong thời gian chờ dữ liệu mạng được trả về dưới client). Phương thức Fetch sẽ trả về một [Promise](#), điều này sẽ dễ dàng để các bạn có thể viết các đoạn code xử lý cho các thao tác bất đồng bộ:

```
function getMoviesFromApiAsync() {
  return fetch('https://facebook.github.io/react-native/movies.json')
    .then((response) => response.json())
    .then((responseJson) => {
      return responseJson.movies;
    })
    .catch((error) => {
      console.error(error);
    });
}
```

Bạn cũng có thể sử dụng mẫu cấu trúc ES2017 về `async/await` trong ứng dụng React Native:

```
async function getMoviesFromApi() {
  try {
    let response = await fetch('https://facebook.github.io/react-native/movies.json');
    let responseJson = await response.json();
    return responseJson.movies;
  } catch(error) {
    console.error(error);
  }
}
```

Đừng quên câu lệnh `catch` để bắt bất kỳ một lỗi nào xảy ra khi mà bạn thực hiện `fetch`. Bên cạnh đó chúng ta cũng không nên âm thầm bỏ qua các lỗi.

Mặc định, iOS sẽ khóa tất cả những request nào không được mã hóa khi sử dụng SSL. Nếu như bạn muốn `fetch` từ một cleartext URL (là URL được bắt đầu với `http`), bạn sẽ cần phải thêm một ngoại lệ App Transport Security. Nếu như bạn biết thời gian trước để có thể truy cập vào một

domain sẽ cần rất nhiều các lớp bảo mật nên sẽ có rất nhiều các ngoại lệ được thêm vào để bắt ngoại lệ ở mỗi lớp bảo mật. Nếu như tên miền không được biết trong quá trình ứng dụng chạy thì bạn có thể vô hiệu hóa hoàn toàn AST. Chú ý rằng dù thế nào thì từ tháng 1 năm 2017, Apple Reviewing team của App Store sẽ yêu cầu một lý do hợp lý nếu như bạn muốn vô hiệu hóa ATS. Bạn có thể xem Apple's documentation để biết thêm thông tin.

Sử dụng các thư viện Networking khác

XMLHttpRequest API là một thành phần được tích hợp sẵn vào trong React Native. Điều này đồng nghĩa với việc bạn có thể sử dụng các thư viện của bên thứ ba như frisbee hoặc axios và thao tác trên chúng. hoặc bạn cũng có thể sử dụng XMLHttpRequest API nếu như bạn muốn. Bạn có thể xem ví dụ ngay dưới đây:

```
var request = new XMLHttpRequest();

request.onreadystatechange = (e) => {

  if (request.readyState !== 4) {

    return;

  }

  if (request.status === 200) {

    console.log('success', request.responseText);

  } else {

    console.warn('error');

  }

};

request.open('GET', 'https://mywebsite.com/endpoint/');

request.send();
```

Mô hình bảo mật của XMLHttpRequest là khác biệt so với web. nó dường như là không có khái niệm trong một ứng dụng native, bạn có thể đọc trong CORS.

Hỗ trợ WebSocket

React Native đồng thời hỗ trợ [WebSocket](#), một phương thức tạo kênh kết nối liên tục bằng cách sử dụng kết nối TCP.

```
var ws = new WebSocket('ws://host.com/path');

ws.onopen = () => {
  // connection opened

  ws.send('something'); // send a message
};

ws.onmessage = (e) => {
  // a message was received

  console.log(e.data);
};

ws.onerror = (e) => {
  // an error occurred

  console.log(e.message);
};

ws.onclose = (e) => {
  // connection closed

  console.log(e
```

Như vậy với bài hướng dẫn này bạn đã có thể tích hợp các liên kết internet vào trong ứng dụng của mình, đã có rất nhiều thứ có thể làm được sau bài hướng dẫn này rồi. Và trong bài hướng

dẫn sau tôi sẽ nói nhiều hơn về cách sử dụng, trao đổi dữ liệu giữa các màn hình trong ứng dụng React Native. Các bạn chờ tiếp nhé.

CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

2.1 Phân tích hệ thống

2.1.1. Màn hình trang chủ

2.1.1.1 User Interfaces

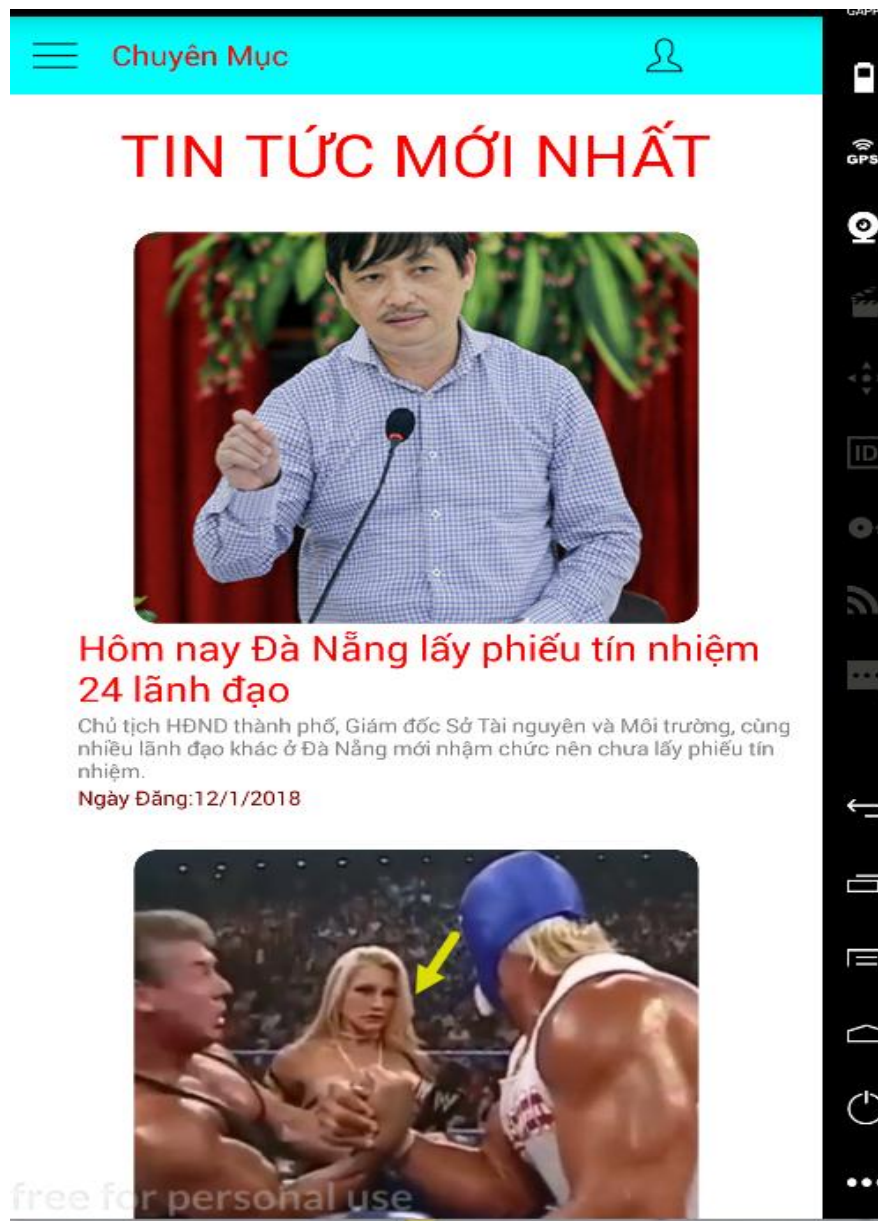
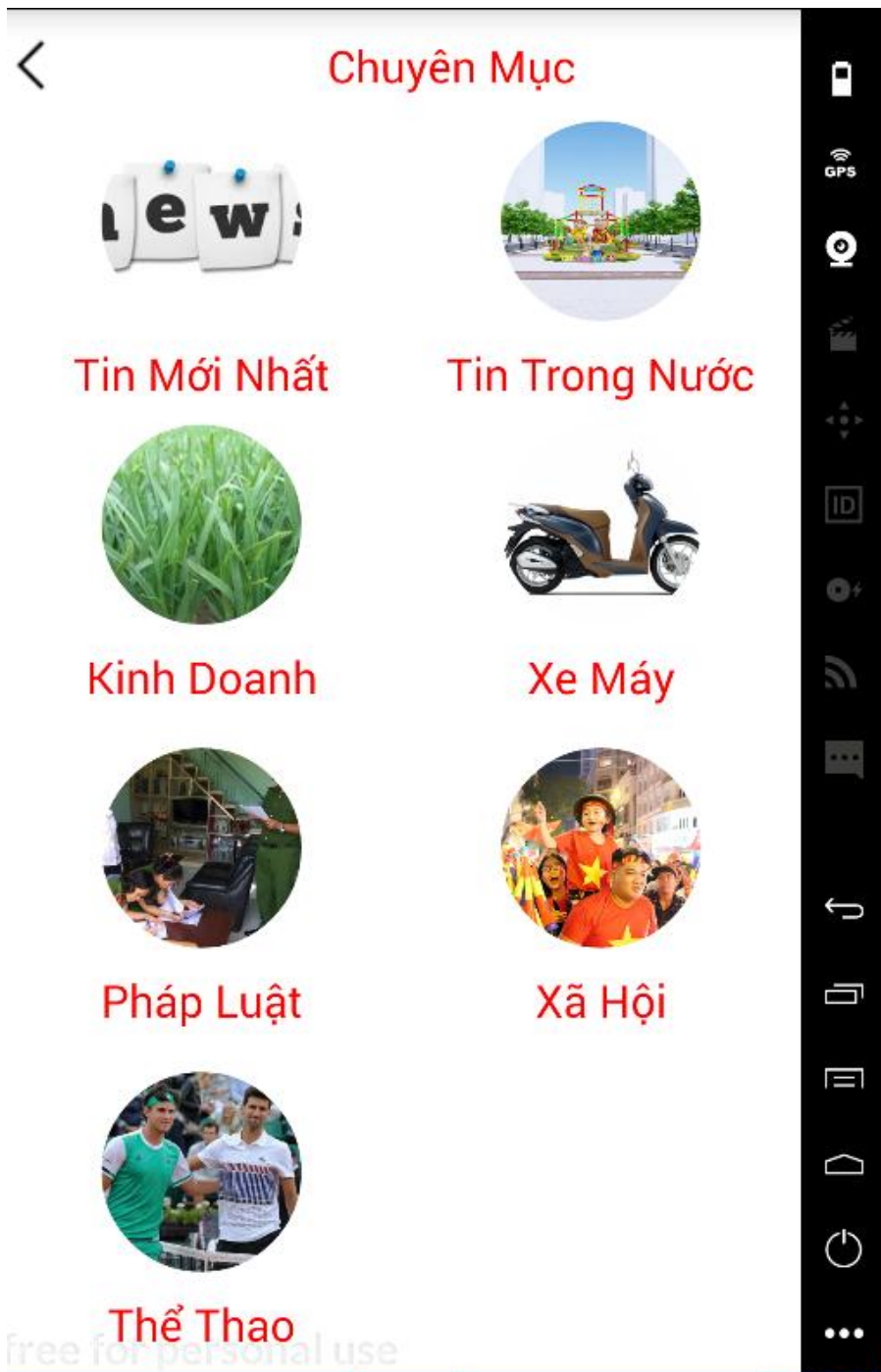


Image1: Trang chủ



Màn hình :danh mục



Quay Lại

Hôm nay Đà Nẵng lấy phiếu tín nhiệm 24 lãnh đạo

Ngày Đăng: 12/1/2018

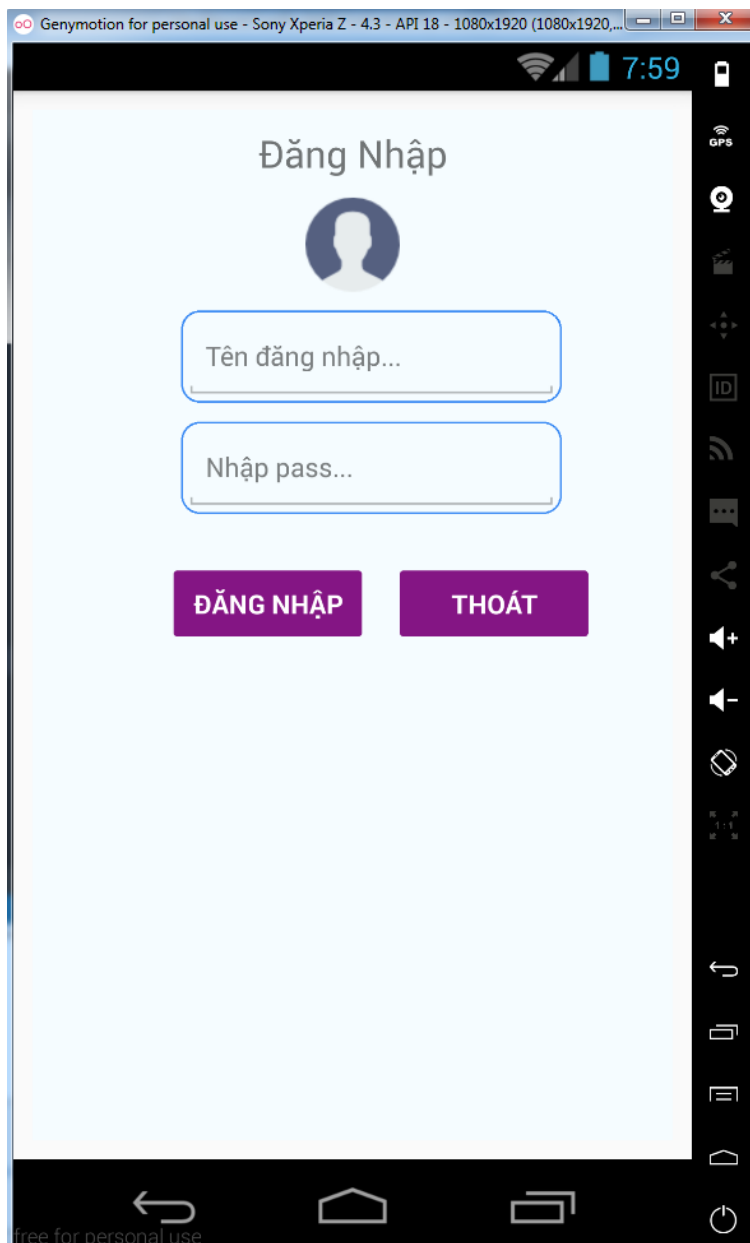
Chủ tịch HĐND thành phố, Giám đốc Sở Tài nguyên và Môi trường, cùng nhiều lãnh đạo khác ở Đà Nẵng mới nhậm chức nên chưa lấy phiếu tín nhiệm.



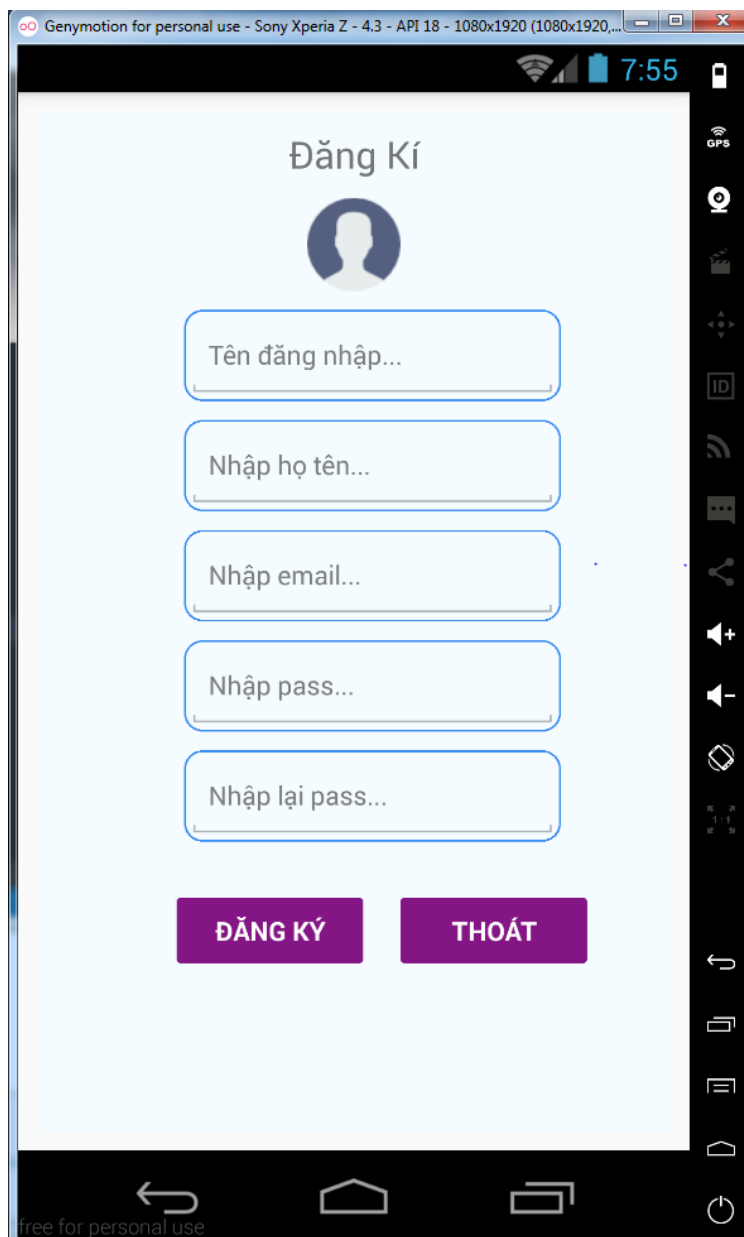
Kỳ họp thứ 9 HĐND TP Đà Nẵng sẽ diễn ra vào ngày 17 đến 19/12. Trong đó HĐND thành phố sẽ lấy phiếu tín nhiệm với 24 vị trí lãnh đạo chủ chốt của UBND và HĐND.

Theo đó, khối chính quyền sẽ gồm ông Huỳnh Đức Thơ - Chủ tịch UBND, ba trong số bốn chức danh Phó chủ tịch UBND, cùng 17 ủy viên UBND thành phố. Phía HĐND thành phố sẽ có ba người là chánh văn phòng

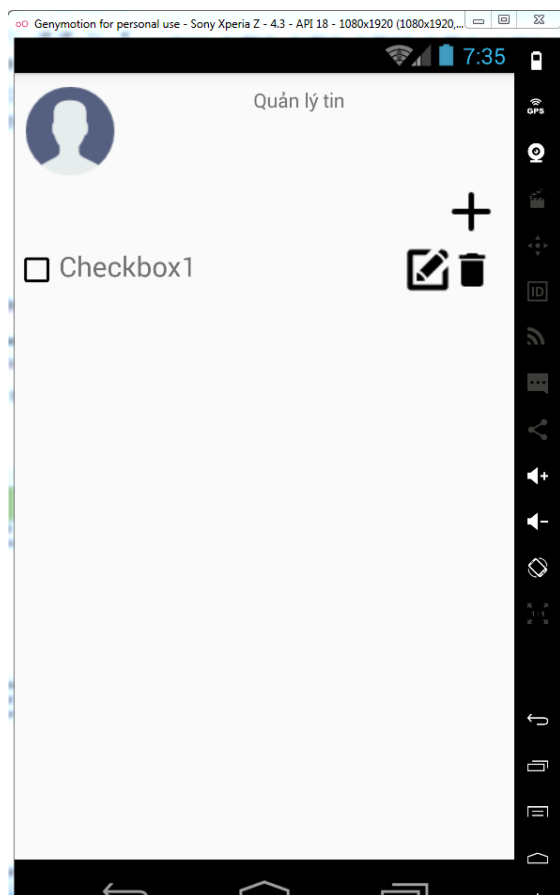
Màn hình đăng nhập



Màn hình đăng ký




Màn hình quản lý tin



Màn hình đăng tin

UPLOAD NEWS



Wellcome ABC!

Nhập tiêu đề bảng tin

Nhập mô tả bảng tin

Loại tin

Thể thao

Nhập nội dung 1

UPLOAD

IMG

CANCEL

Màn hình: chi tiết

2.1.1.2. Functional Requirements

Item	Description	Action	Response
Màn hình trang	Màn hình trang chủ sử dụng:	N/A	N/A

chủ	Image Text Flatlist		
	Image		Hiển thị hình ảnh của tin tức
	Text	Click vào text	Đi tới trang chi tiết của tin đó
	Flatlist		Hiển thị các tin có trong cơ sở dữ liệu(5 tin 1 trang)
	Icon danh mục	Click vào icon	Hiển thị ra trang danh mục tất cả loại tin
	Icon admin	Click vào icon	Hiển thị ra giao diện đăng nhập hoặc đăng ký vào hệ thống
Màn hình Danh mục	Màn hình danh mục sử dụng: Image Text		
	Image, text	Click vào image hoặc text	Đi tới trang chủ và hiển thị ra tất cả các tin thuộc thể loại đó
Màn hình Chi tiết	Màn hình chi tiết sử dụng : Text, Scollview		
	Text, Scollview		Hiển thị ra tất cả nội dung của tin đã được chọn
Màn hình đăng ký	Màn hình đăng ký có 5 textinput 2 button Icon	N/A	N/A
	Text đăng ký	Click vào text đăng ký	Thêm dữ liệu vào webservice
	Text thoát	Click vào button đăng thoát	Quay về màn hình trang chủ
	TextInput tên đăng nhập	Điền thông tin tương ứng	Thêm dữ liệu vào webservice
	TextInput nhập họ tên	Điền thông tin tương ứng	Thêm dữ liệu vào webservice
	TextInput nhập email	Điền thông tin tương ứng	Thêm dữ liệu vào webservice
	TextInput nhập pass	Điền thông tin tương ứng	
	TextInput nhập lại pass	Điền thông tin tương ứng	Thêm dữ liệu vào webservice

Màn hình đăng nhập	Màn hình đăng nhập gồm có:		
	2 text		
	2 textInput		
	1 icon	N/A	N/A
	Text đăng nhập	Click vào button đăng nhập	Đăng nhập vào hệ thống
	Text đăng thoát	Click vào button đăng thoát	Quay về màn hình trang chủ
	TextInput	Điền thông tin tương ứng	Thông tin được kiểm tra trên webservice
Màn hình quản lý tin tức	Màn hình quản lý tin bao gồm: <ul style="list-style-type: none"> - Text - 1 Image - Các Checkbox. - 3 button. 	N/A	N/A
Text		N/A	N/A
Nút Checkbox	<ul style="list-style-type: none"> - Dùng để hiển thị ra các tin tức đã được đăng - Sử dụng checkbox khi muốn chỉnh sửa các tin tức đã đăng 	Click vào ô checkbox	
Nút Thêm	<ul style="list-style-type: none"> - Nhấp chọn vào nút thêm khi người dung (đã đăng ký tài khoản) muốn đăng các tin tức mới lên trang báo. 	Click vào nút thêm	Thêm dữ liệu
Nút Xóa	<ul style="list-style-type: none"> - Dùng để xóa các tin tức. - Nhấp chọn vào checkbox tin tức và chọn vào nút Xóa để xóa các tin tức mà người dung cần. 	Click vào nút xóa	Xóa dữ liệu
Nút Sửa	<ul style="list-style-type: none"> - Dùng để chỉnh sửa những tin tức. - Nhấp chọn vào checkbox tin tức để chỉnh sửa các thông tin trong tin tức mà người dùng muốn. 	Click vào nút sửa	Sửa dữ liệu
My UploadNews Screen	MyMobile Màn Hình UploadNews có 3 buttons: <ul style="list-style-type: none"> - Upload the new News - IMG(upload the image of news) - Cancel - Picker(choose the type news) 	N/A	N/A

	"Thể thao" là Picker(chọn thể loại tin) "Avatar" có thể Touch(Di chuyển tới màn hình Edit User Screen) "Nhập tiêu đề bản tin" là TextInput "Nhập mô tả bản tin" là TextInput "Nhập nội dung 1" là TextInput		
Loại tin	Chạm và chọn loại tin phù hợp	Tap on the Picker	Picker sẽ xuất hiện danh sách loại tin để chọn ra loại tin phù hợp
	-		
Button UPLOAD	Sau khi nhập đầy đủ thông tin nội dung thì sẽ gửi những thông tin đó vào cơ sở dữ liệu	Tap on UPLOAD button	Gửi những thông tin vào cơ sở dữ liệu
Button IMG	Chọn hình ảnh cần thiết liên quan đến nội dung của bản tin	Tap on the IMG button	Gửi hình ảnh đến cơ sở dữ liệu
Cancel	Hủy màn hình UploadNews và quay về màn hình trước	Tap on Cancel button	Quay về màn hình trước đó là màn hình chính
Avartar User	Sử dụng để chuyển đến màn hình chỉnh sửa User	Tap on the Avatar of User	Chuyển đến màn hình chỉnh sửa user
Input Tiêu đề	Nhập tiêu đề bản tin		
Input mô tả	Nhập mô tả bản tin		
Input nội dung	Nhập nội dung bản tin		

2.2 Thiết kế hệ thống

2.2.1. Màn hình Trang chủ

2.2.1.1. Giao diện trang chủ



Chuyên Mục



TIN TỨC MỚI NHẤT



Hôm nay Đà Nẵng lấy phiếu tín nhiệm 24 lãnh đạo

Chủ tịch HĐND thành phố, Giám đốc Sở Tài nguyên và Môi trường, cùng nhiều lãnh đạo khác ở Đà Nẵng mới nhậm chức nên chưa lấy phiếu tín nhiệm.

Ngày Đăng: 12/1/2018



2.2.1.2. Đối tượng và chức năng trên trang chủ

Đối tượng :

- Icon danh mục:
- Icon admin:
- Flatlist

Actions:

Nhấn vào icon danh mục:

```
<View style={styles.header}>
  <View style={styles.img1}>
    <TouchableOpacity onPress={() => this.props.navigation.navigate('DanhMucs')}>
      <Image
        style={styles.imgMenu}
        source={require('./giaodien/menu.png')}
      />
    </TouchableOpacity>
  </View>
  <Text style={styles.txtMenu}>Chuyên Mục</Text>
</View>
```

Thực hiện hàm onPress di chuyển tới trang danh mục thông qua phương thức navigation

Nhấn vào icon Admin:

```
<View style={styles.img2}>
  <TouchableOpacity onPress={() => this.props.navigation.navigate('Login')}>
    <Image
      style={styles.imgMenu}
      source={require('./giaodien/login.png')}
    />
  </TouchableOpacity>
</View>

</View>
```

Thực hiện hàm onPress di chuyển tới trang đăng nhập thông qua phương thức navigation

Flastlist:

Hiển thị tất cả tin khi load chương trình lên

Sử dụng componentDidMount để khởi chạy đầu tiên và load dữ liệu về thông qua phương thức fetch

```
componentDidMount() {  
  fetch("http://192.168.143.2/webtintuc/Select_TinTuc.php?id=" + id)  
    .then((response) => response.json())  
    .then((responseJson) => {  
      this.setState({  
        isLoading: false,  
        dataSource: responseJson,  
      });  
    })  
    .catch((error) => {  
      console.error(error);  
    });  
}
```

2.2.2. Màn hình Danh Mục

2.2.2.1. Giao diện Danh Mục



2.2.2.2. Đối tượng và chức năng trên Danh Mục

Đối tượng :

Text và Image

Chức năng:

Khi nhấn vào text hoặc image:

Sẽ thực hiện navigation chuyển qua trang chủ và truyền theo tham số id và title để lấy dữ liệu về

```
<View style={styles.cot}>
<TouchableOpacity onPress={() => this.props.navigation.navigate('Home',{id:1,title:"TIN TRONG NƯỚC"})}>
  <Image
    style={styles.image}
    id="1"
    source={require('./giaodien/tintrongnuoc.jpg')}
  />
</TouchableOpacity>
<Text style={styles.tl}>Tin Trong Nước</Text>
</View>
</View>
<View style={styles.hang}>
```

Bên trang chủ sẽ lấy về id và tiếp tục thực hiện fetch giá trị từ webservice trả về

```
const { navigation } = this.props;
const id = navigation.getParam('id');
const title = navigation.getParam('title');

fetch("http://192.168.143.2/webtintuc/Select_TinTuc.php?id=" + id)
  .then((response) => response.json())
  .then((responseJson) => {
    this.setState({
      isLoading: false,
      dataSource: responseJson,
    });
  })
  .catch((error) => {
    console.error(error);
  });
```

2.2.2. Màn hình Chi Tiết

2.2.2.1. Giao diện Chi Tiết



Quay Lại

Hôm nay Đà Nẵng lấy phiếu tín nhiệm 24 lãnh đạo

Ngày Đăng: 12/1/2018

Chủ tịch HĐND thành phố, Giám đốc Sở Tài nguyên và Môi trường, cùng nhiều lãnh đạo khác ở Đà Nẵng mới nhậm chức nên chưa lấy phiếu tín nhiệm.



Kỳ họp thứ 9 HĐND TP Đà Nẵng sẽ diễn ra vào ngày 17 đến 19/12. Trong đó HĐND thành phố sẽ lấy phiếu tín nhiệm với 24 vị trí lãnh đạo chủ chốt của UBND và HĐND.

Theo đó, khối chính quyền sẽ gồm ông Huỳnh Đức Thợ - Chủ tịch UBND, ba trong số bốn chức danh Phó chủ tịch UBND, cùng 17 ủy viên UBND thành phố. Phía HĐND thành phố sẽ có ba người là chánh văn phòng

2.2.2.2. Đối tượng và chức năng trên Chi Tiết

Đối tượng Scollview

Action:

Click vào tiêu đề trên trang chủ để truyền tham số qua trang chủ

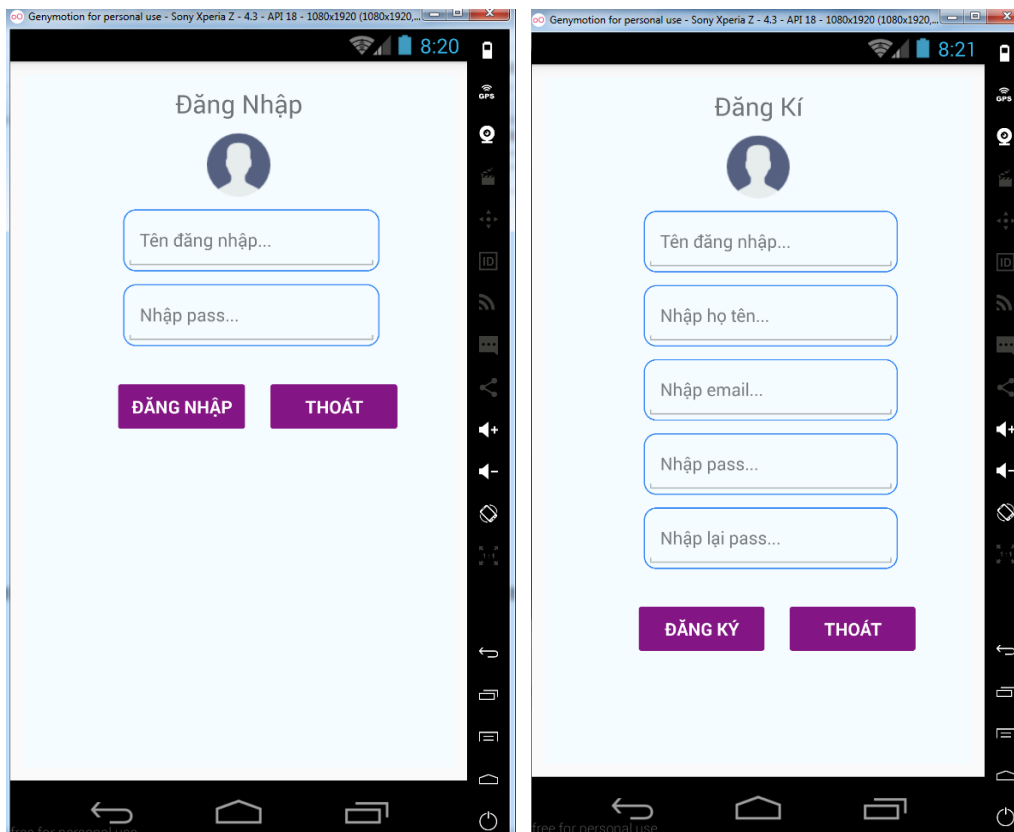
```
</View>
<View style={styles.text}>
  <TouchableOpacity onPress={() => this.props.navigation.navigate('Details',
    {
      tieude: item.tieu_de, ngaydang: item.ngay_dang, nguoidang: item.nguoi_dang, mota: item.mo_ta,
      noidung: item.noi_dung, hinhanh: item.hinh_anh
    })}>
    <Text style={styles.tieude}>{item.tieu_de}</Text>
  </TouchableOpacity>
</View>
```

Hiển thị chi tiết 1 tin bằng cách nhận tham số truyền qua từ trang chủ

```
</View>
<Text style={styles.title1}>{this.props.navigation.state.params.tieude}</Text>
</View>
```

Sử dụng câu lệnh trên để nhận giá trị gửi qua từ trang chủ

Màn hình đăng nhập và đăng ký



2.2.1.2. Màn hình đăng nhập và đăng ký

Objects: Đăng nhập

Text Đăng nhập

Text Đăng thoát

TextInput tên đăng nhập

TextInput mật khẩu

Actions:

```
dangnhap=()=> {  
  const {user,password} = this.state;  
  fetch('http://192.168.143.2/webtintuc/login.php',  
    {  
      method: 'POST',  
      headers:  
      {  
        'Accept': 'application/json',  
        'Content-Type': 'application/json',  
      },  
      body: JSON.stringify(  
        {  
          userName : user,  
          passWord : password ,  
        })  
    })  
  ).then((response) => response.json()).then(  
    (responseJson)=>{  
      if(responseJson=='ok'){  
        this.props.navigation.navigate('quanlytin')  
      }else{  
        alert('Dang Nhap That Bai !');  
      }  
    }  
  )  
};
```

Khi nhấp vào tên đăng nhập:

```
<Text style={{ textAlign: 'center', marginTop: 20, color: 'red', fontSize: 30, marginBottom: 50 }}>ĐĂNG NHẬP</Text>  
<TextInput  
  placeholder="Nhập Tên Đăng Nhập"  
  style={styles.text}  
  underlineColorAndroid="transparent"  
  onChangeText = {(TextInputText) => this.setState({ user: TextInputText })}/>  
</TextInput>
```

Thông tin được kiểm tra trên webservice

Khi nhập nhấp vào nhập pass

```

    <TextInput
      placeholder="Nhập Mật Khẩu"
      style={styles.text}
      underlineColorAndroid="transparent"
      onChangeText = {(TextInputText) => this.setState({ password: TextInputText })} />
    <TouchableOpacity
      activeOpacity={0.5}

```

Thông tin được kiểm tra trên webservice

Objects Đăng ký

Text đăng ký

Text thoát

TextInput

Actions:

Khi nhấn vào đăng ký

```

<View style={styles.container}>
  <View style={styles.img1}>
    <TouchableOpacity onPress={() => this.props.navigation.navigate('Home')}>
      <Image
        style={styles.imgMenu}
        source={require('./giaodien/back.png')}
      />
    </TouchableOpacity>
    <Text style={styles.txt}>Quay Lại</Text>
  </View>
  <View style={styles.dangnhap}>
    <Text style={{ textAlign: 'center', marginTop: 20, color: 'red', fontSize: 30, marginBot
  <TextInput
    placeholder="Nhập Họ Tên"
    style={styles.text}
    underlineColorAndroid="transparent"
    onChangeText={({text) => this.setState({ hoten: text })} />
  <TextInput
    placeholder="Nhập Tên Đăng Nhập"
    style={styles.text}
    underlineColorAndroid="transparent"
    onChangeText={({text) => this.setState({ user: text })} />
  <TextInput
    placeholder="Nhập Email"
    style={styles.text}
    underlineColorAndroid="transparent"
    onChangeText={({text) => this.setState({ email: text })} />
  <TextInput
    placeholder="Nhập Mật Khẩu"
    style={styles.text}
    underlineColorAndroid="transparent"
    onChangeText={({text) => this.setState({ password: text })} />
  <TextInput
    placeholder="Nhập Lại Mật Khẩu"
    style={styles.text}
    underlineColorAndroid="transparent"
    onChangeText={({text) => this.setState({ password2: text })} />
  <TouchableOpacity
    activeOpacity={0.5}
    style={styles.txtButton}
    onPress={() => { this.dangKy() }}>
    <Text style={styles.txt1}>Đăng Ký</Text>
  </TouchableOpacity>

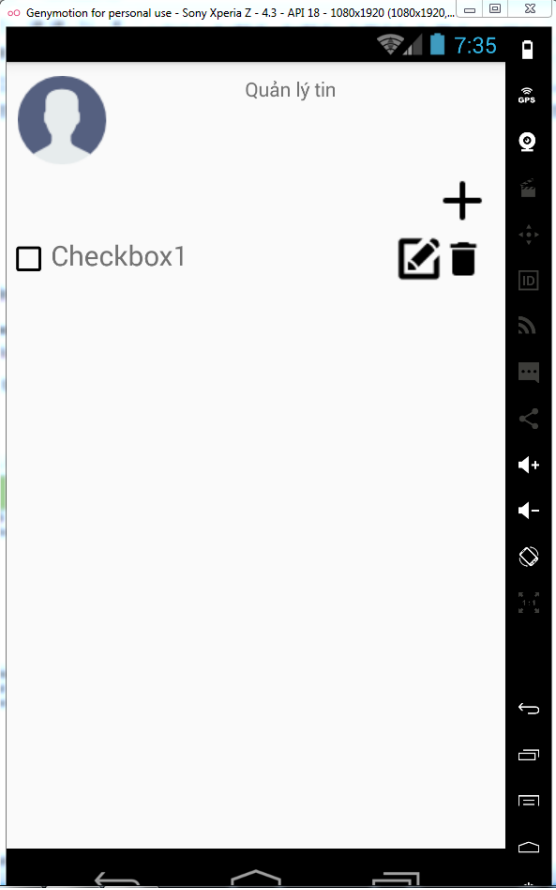
```

```

    <Text style={styles.txt1}>Đăng Ký</Text>
  </TouchableOpacity>
  <TouchableOpacity
    activeOpacity={0.5}
    style={styles.txtButton}
    onPress={() => this.props.navigation.navigate('Home')}>
    <Text style={styles.txt1}>THOÁT</Text>
  </TouchableOpacity>

```

Thông tin khi điền vào được lưu trên webservice
Màn hình quản lý tin



2.2.1.2. Objects and actions

Objects:

- Checkbox
- Thêm
- Xóa
- Sửa

Actions:

- Load dữ liệu trong cơ sở dữ liệu vào Checkbox tin tức.

```
export default class Quanlytin extends Component {
  constructor(props) {
    super(props);
    this.state = { isLoading: true }
  }

  componentDidMount() {
    return fetch('http://192.168.143.2/webtintuc/Select_TinTuc.php?id=0')
      .then((response) => response.json())
      .then((responseJson) => {

        this.setState({
          isLoading: false,
          dataSource: responseJson,
        }, function () {

        });

      })
      .catch((error) => {
        console.error(error);
      });
  }
}
```

- Thêm Xóa Sửa các tin tức


```
<View style={styles.iconadd}>
  <TouchableOpacity >
    <Image style={styles.imgedt} source={require('./img/add.png')}></Image>
  </TouchableOpacity>
</View>
<FlatList
  data={this.state.dataSource}
  renderItem={({ item }) => <View style={styles.hang}>
    <View style={styles.title}>
      <Text>{item.tieu_de}</Text>
    </View>
    <View style={styles.hinh}>
      <Image
        style={styles.imgMenu}
        source={require('./img/edit.png')}
      />
    </View>
    <View style={styles.hinh}>
      <Image
        style={styles.imgMenu}
        source={require('./img/delete.png')}
      />
    </View>
  </View>
</FlatList>
```

2.1 Phân tích hệ thống

2.1.1. Feature/Component #1: MyMobile UploadNews screen

2.1.1.1 User Interfaces

UPLOAD NEWS

 Wellcome ABC!

Nhập tiêu đề bảng tin

Nhập mô tả bảng tin

Loại tin


Thể thao

Nhập nội dung 1

UPLOAD IMG CANCEL

2.1.2. Feature/Component #1: MyMobile EditUser screen

2.1.2.1 User Interfaces



Chỉnh sửa thông tin cá nhân

Họ tên

Email

SĐT

Password

Nhập lại Password

ENTER

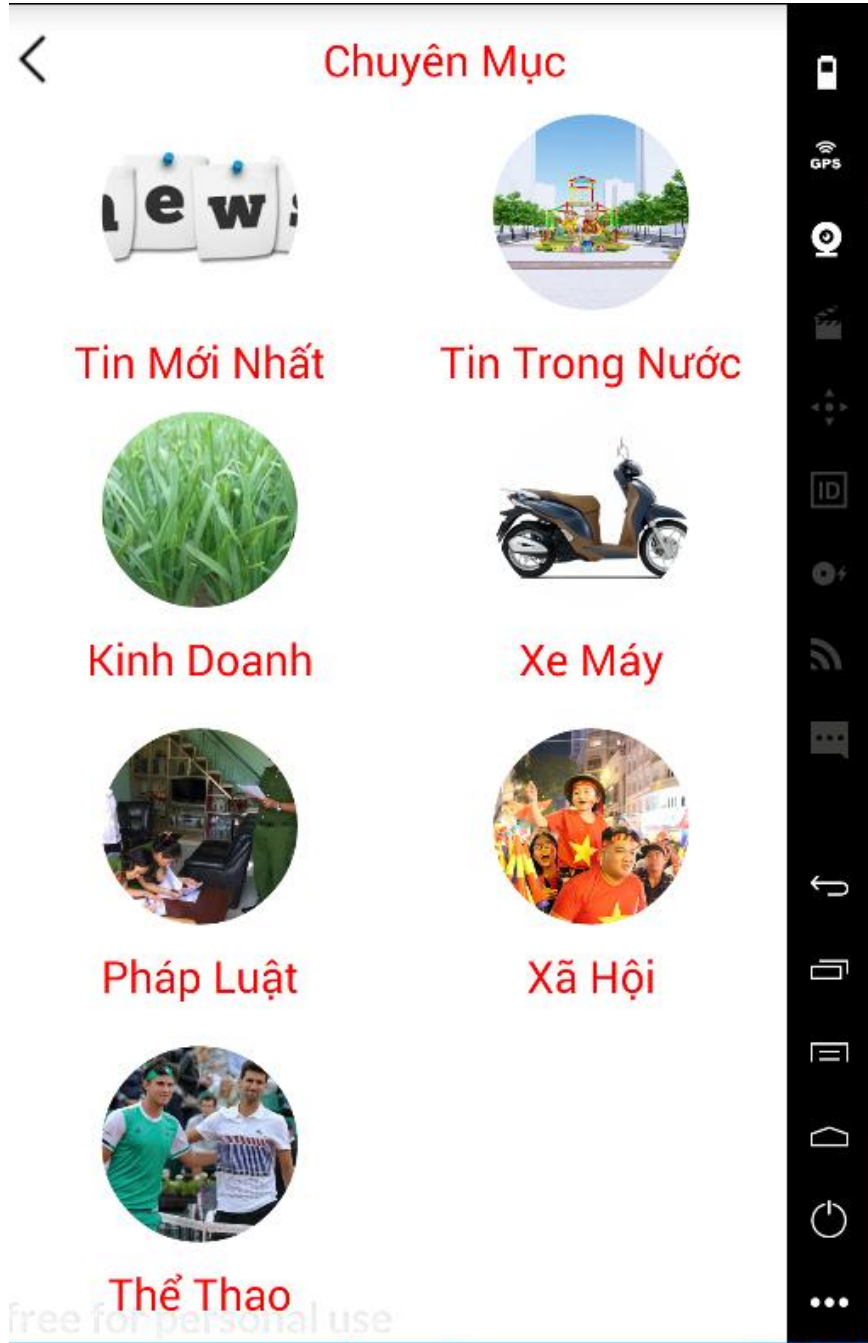
CHƯƠNG 3. KẾT QUẢ ĐẠT ĐƯỢC

4.1 Kết quả đạt được

Hiện thị được danh sách bài viết



Click vào image hiển thị trang danh mục



Click vào tiêu đề hiển thị chi tiết bài viết

[← Quay Lại](#)

Hôm nay Đà Nẵng lấy phiếu tín nhiệm 24 lãnh đạo

Ngày Đăng: 12/1/2018

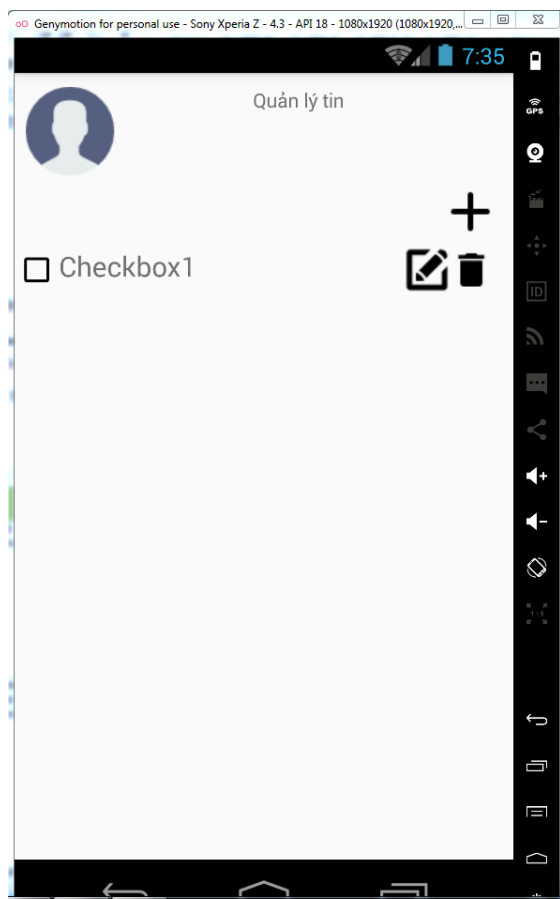
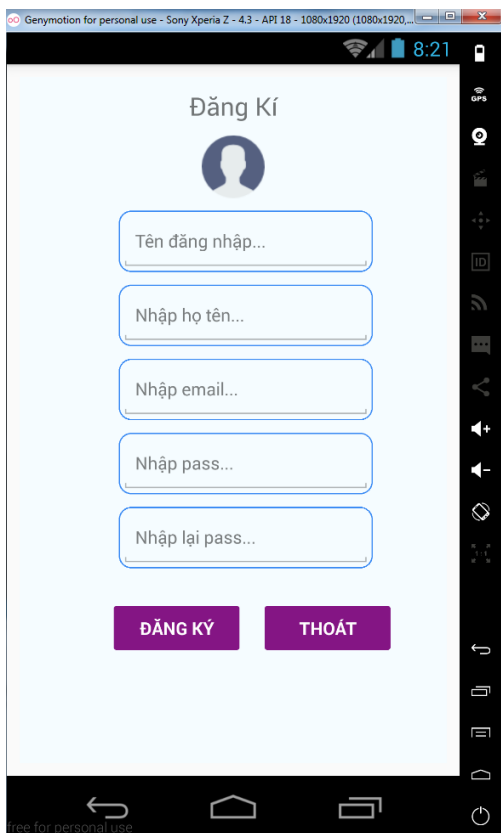
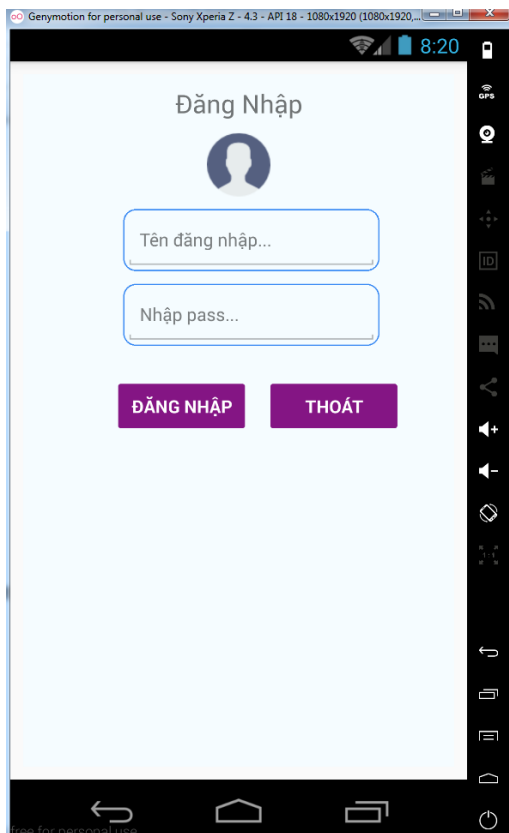
Chủ tịch HĐND thành phố, Giám đốc Sở Tài nguyên và Môi trường, cùng nhiều lãnh đạo khác ở Đà Nẵng mới nhậm chức nên chưa lấy phiếu tín nhiệm.



Kỳ họp thứ 9 HĐND TP Đà Nẵng sẽ diễn ra vào ngày 17 đến 19/12. Trong đó HĐND thành phố sẽ lấy phiếu tín nhiệm với 24 vị trí lãnh đạo chủ chốt của UBND và HĐND.

Theo đó, khối chính quyền sẽ gồm ông Huỳnh Đức Thợ - Chủ tịch UBND, ba trong số bốn chức danh Phó chủ tịch UBND, cùng 17 ủy viên UBND thành phố. Phía HĐND thành phố sẽ có ba người là chánh văn phòng

Click vào image admin hiển thị trang đăng nhập vào hệ thống





Chỉnh sửa thông tin cá nhân

Họ tên

Email

SDT

Password

Nhập lại Password

ENTER

UPLOAD NEWS



Wellcome ABC!

Nhập tiêu đề bảng tin

Nhập mô tả bảng tin

Loại tin

Thể thao

Nhập nội dung 1

UPLOAD IMG CANCEL

4.2 Các kết luận và kiến nghị

- Những điểm đã làm được

- Sử dụng được flatlist fetch dữ liệu từ webservice đổ vào
- Hiểu được một số tính năng, các component của react native

- Những điểm chưa làm được

- Kết nối các chức năng chưa được logic
- Còn giao diện chưa xử lý kịp

- Các chức năng bổ sung nếu có thêm thời gian...

Thiết kế lại giao diện và thêm đầy đủ tính năng của 1 App tin tức

TÀI LIỆU THAM KHẢO