Web Service, cơ chế vận hành và cách tương tác giữa web service và mobile app

## Web service là gì?

* **Web service** (theo **wikipedia** định nghĩa) "là một dịch vụ mà nó kết hợp các máy tính cá nhân với các thiết bị khác, các cơ sở dữ liệu và các mạng máy tính để tạo thành một cơ cấu tính toán ảo mà người sử dụng có thể làm việc thông qua các trình duyệt mạng. Bản thân các dịch vụ này sẽ chạy trên các máy chủ trên nền Internet. Các máy chủ của một nhà cung cấp dịch vụ web này cũng cần trở thành nguồn cung cấp cho người sử dụng cả về độ an toàn, độ riêng tư và khả nǎng truy nhập."
* Các **Web Service** thường trả về dữ liệu dưới dạng máy tính có thể đọc được là **XML** hoặc **JSON** rồi trả về trình duyệt ở phía client.
* Các công nghệ thường được sử dụng để xây dựng web service:
  + **SOAP (Simple Object Access Protocol)**
  + **WSDL (Web Services Description Language)**
  + **REST** (sẽ tìm hiểu trong bài này)

## RESTful web service:

* RESTful web service là các dịch vụ web được xây dựng dựa trên cấu trúc **REST (REpresentational State Transfer)**. Tức là nó giống như một kiến trúc, nguyên tắc cần tuân theo để thiết kế, xây dựng một web service.
* Trong kiến trúc REST mọi thứ đều được coi là tài nguyên, chúng có thể là: tệp văn bản, ảnh, trang html, video, hoặc dữ liệu động… REST server cung cấp quyền truy cập vào các tài nguyên, REST client truy cập và thay đổi các tài nguyên đó. Ở đây các tài nguyên được định danh dựa vào URI, REST sử dụng một vài đại diện để biểu diễn các tài nguyên như văn bản, JSON, XML.
* Và nội dung của kiến trúc REST bao gồm bốn nguyên tắc cơ bản sau:

## *Cách sử dụng phương thức HTTP*

* Như chúng ta đã biết, HTTP cung cấp các phương thức dùng để lấy dữ liệu, trèn dữ liệu, cập nhập dữ liệu hoặc xóa dữ liệu. Và khi sử dụng những phương thức này, chúng ta cần xác định rõ ràng mục đích sử dụng mỗi khi gọi tới một phương thức. Và gợi ý cụ thể cho các phương thức như sau:
  + **GET**: dùng để truy xuất một tài nguyên (phương thức này gần như là phổ biến nhất)
  + **POST**: dùng để tạo một tài nguyên trên máy chủ (VD như đăng kí tài khoản, sau khi điền form thông tin, dùng phương thức POST để gửi dữ liệu lên máy chủ)
  + **PUT**: dùng để thay đổi trạng thái một tài nguyên hoặc để cập nhật nó.
  + **DELETE**: dùng để huỷ bỏ hoặc xoá một tài nguyên.

## *Phi trạng thái (stateless)*

* Phi trạng thái có nghĩa là máy chủ sẽ không lưu giữ thông tin của client mà nó giao tiếp, thông tin hoặc được giữ trên client hoặc được chuyển thành trạng thái của tài nguyên. Mỗi request lên server thì client phải đóng gói thông tin đầy đủ để thằng server hiểu được.
* Điểu này đem lại hai lợi ích:
  + Giúp tách biệt client ra khỏi sự thay đổi của server.
  + Giúp hệ thống của bạn dễ phát triển,bảo trì, mở rộng vì không cần tốn công CRUD trạng thái của client.
* VD:

Bạn vừa gửi yêu cầu để xem trang thứ 2 của một tài liệu.

Bây giờ bạn muốn xem trang tiếp theo (sẽ là trang 3).

REST không lưu trữ lại thông tin rằng trước đó nó đã phục vụ bạn trang số 2. Điều đó có nghĩa là REST không quản lý phiên làm việc (Session)  
--> Máy chủ chuyển hầu hết vai trò duy trì trạng thái sang ứng dụng ở máy khách, từ đó giúp cho nó hoạt động tốt hơn.

## *Hiển thị cấu trúc thư mục như URI*

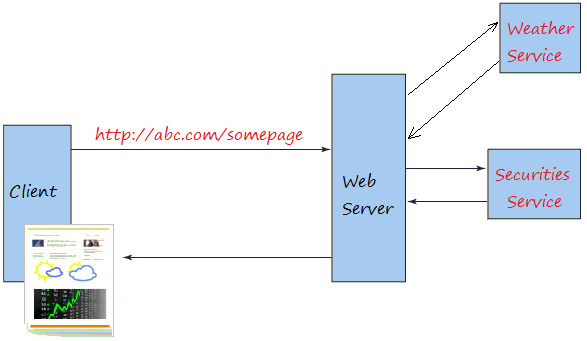
* REST đưa ra một cấu trúc để người dùng có thể truy cập vào tài nguyên của nó thông qua các URL
* Các địa chỉ REST service cần phải thật trực quan đến mức đơn giản, có thể dự đoán, và dễ hiểu. Ví dụ: chỉ cần nhìn vào thanh địa chỉ URL ta có thể đoán rằng nó đang trỏ tới cái gì và cung cấp tài nguyên gì.
* Và để tạo ra đáp ứng yêu cầu trên thì ta nên định nghĩa URI có câu trúc giống thư mục. Loại URI này có phân cấp, có gốc là một đường dẫn đơn, các nhánh từ gốc là các đường dẫn phụ dẫn đến các các vùng service chính.

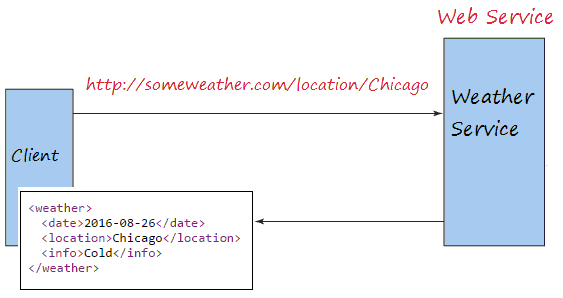
--> cấu trúc này giúp cho nhà phát triển dễ dàng trong việc cài đặt service của mình hướng vào một loại tài nguyên cụ thể nào đó.

## *Định dạng dữ liệu (html, json, text, xml…)*

* Khi Client gửi một yêu cầu tới web service, nó thường được truyền tải dưới dạng dữ liệu mà máy tính hiểu được (XML hoặc JSON) và thông thường nhận về với hình thức tương tự từ máy chủ.
* Tuy nhiên Client cũng có thể chỉ định kiểu dữ liệu nhận về mà nó mong muốn (JSON, hoặc XML,..), các chỉ định này được gọi là các kiểu MIME, nó được gửi kèm trên phần HEADER của request.
* Một số kiểu MIME phổ biến:

| **MIME-TYPE** | **Content-Type** |
| --- | --- |
| JSON | application/json |
| XML | application/xml |
| XHTML | application/xhtml+xml |





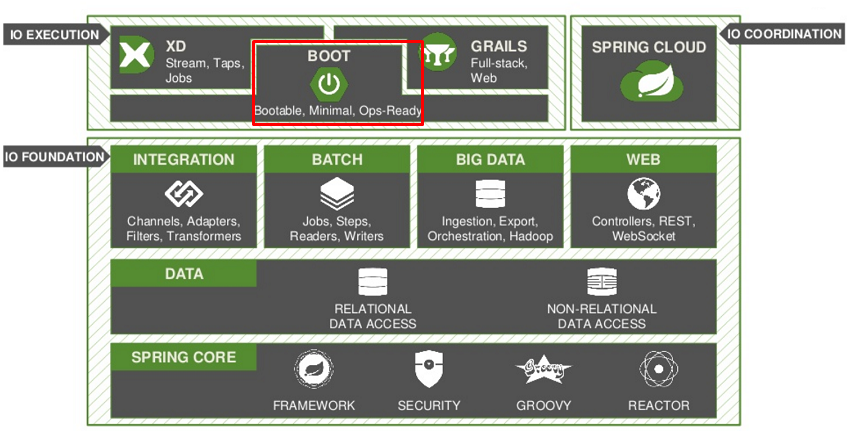
## Phía server

# Tạo RESTful Web Service với Spring boot

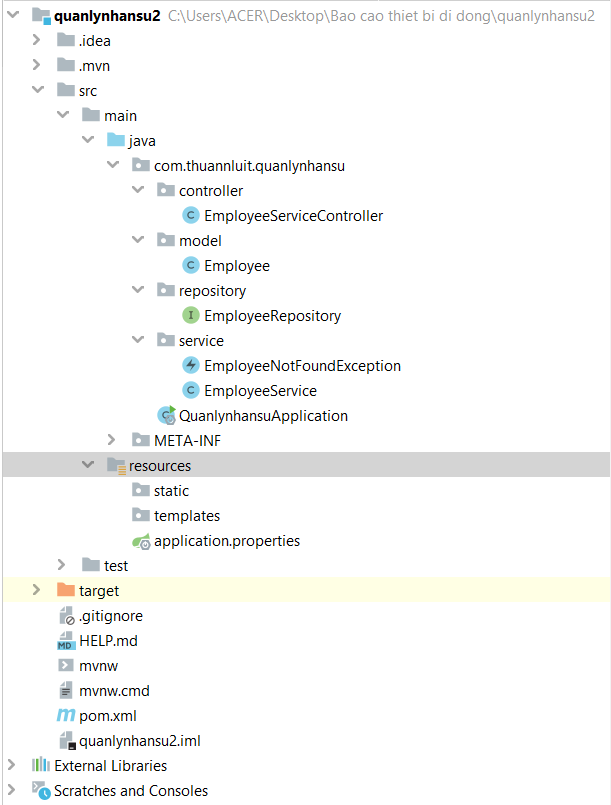
Dùng intellj + spring boot + mysql + java + jpa

## Spring Boot là gì?

* Spring Boot là một module của Spring Framework, cung cấp tính năng RAD (Rapid Application Development) – Phát triển ứng dụng nhanh.
* Spring Boot được dùng để tạo các ứng dụng độc lập dựa trên Spring.
* Spring Boot không yêu cầu cấu hình XML
* Nó là một chuẩn cho cấu hình thiết kế phần mềm, tăng cao năng suất cho developer.



Cấu trúc thư mục



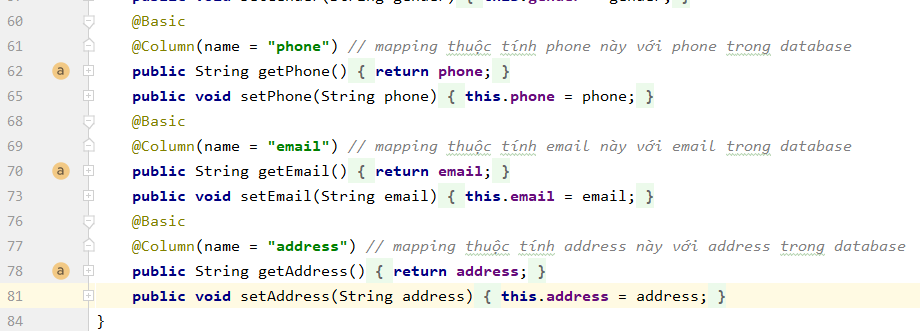
Thư viện sử dụng



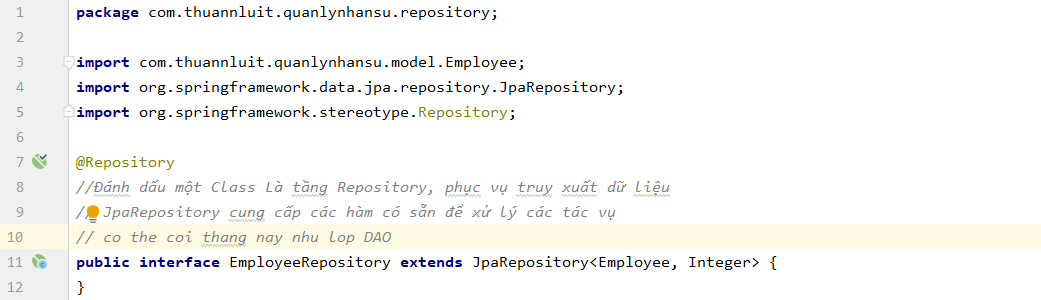


File Entity



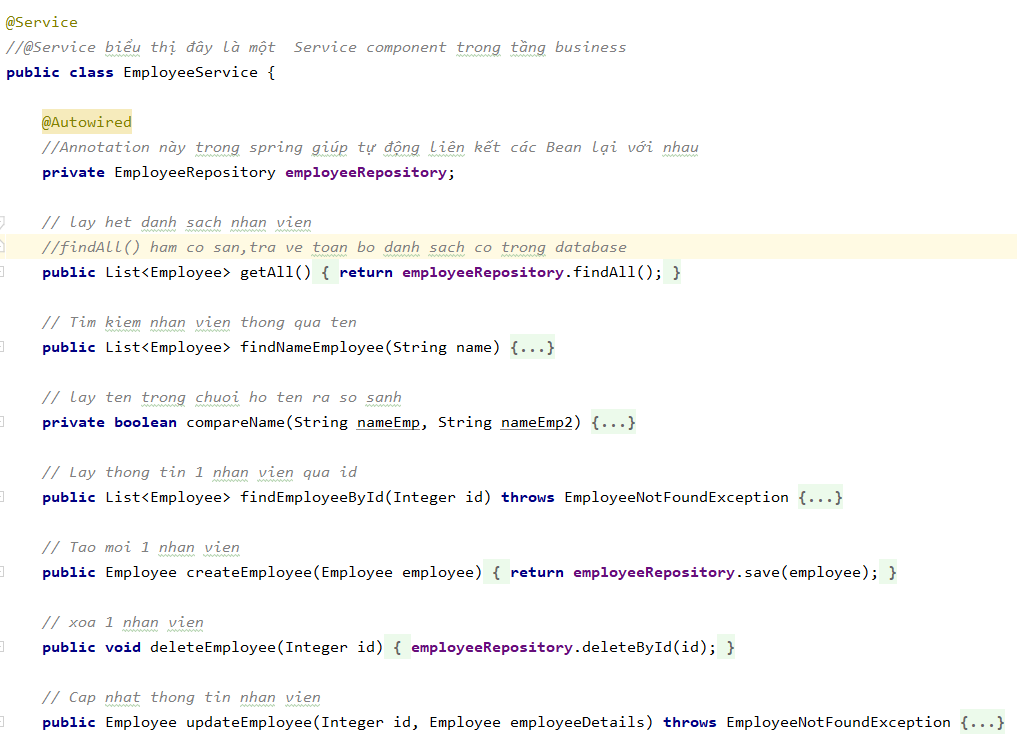


File @Repository

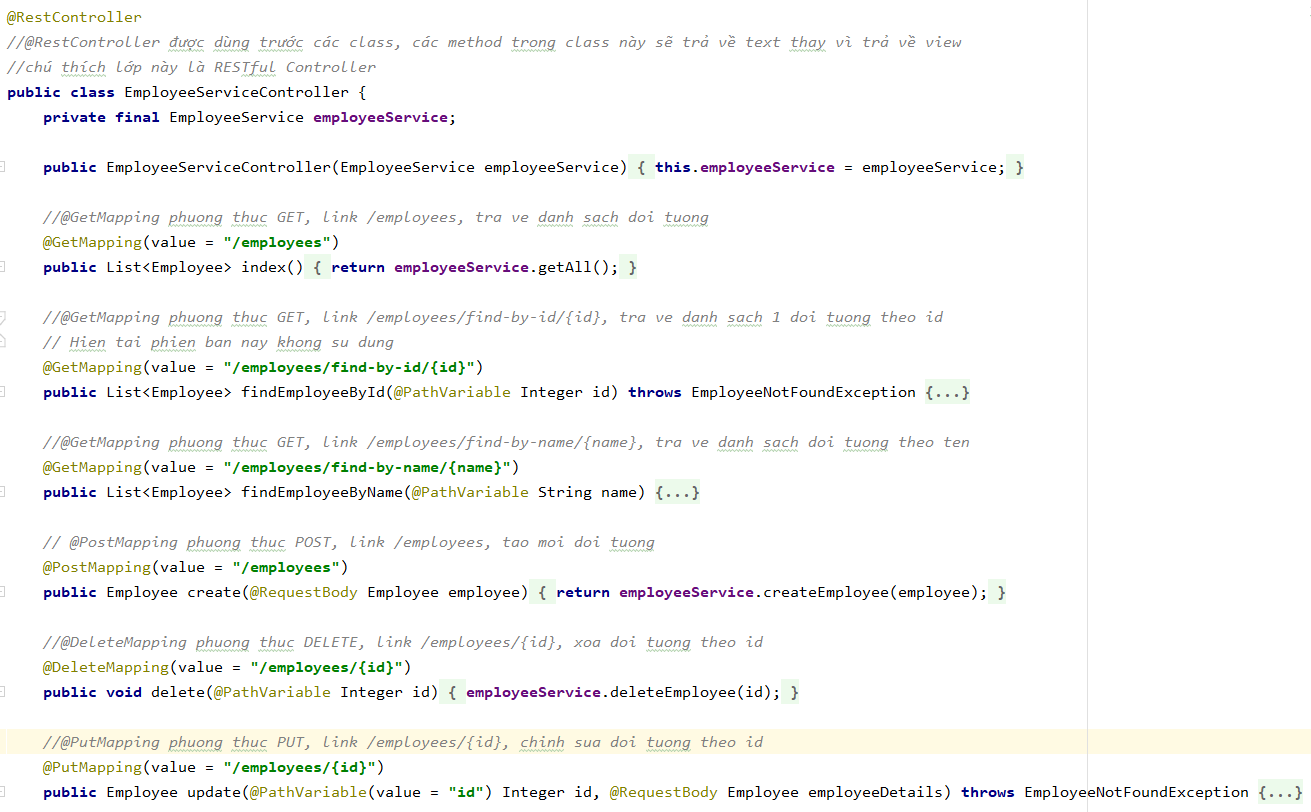


Lớp service :

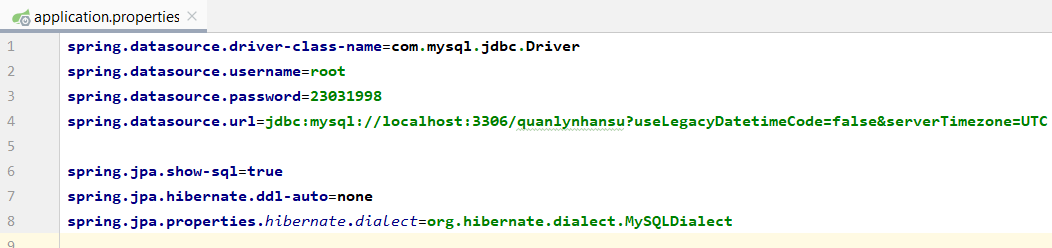
Lớp này dung để tạo ra các hàm xử lý



File Controller



File config database :



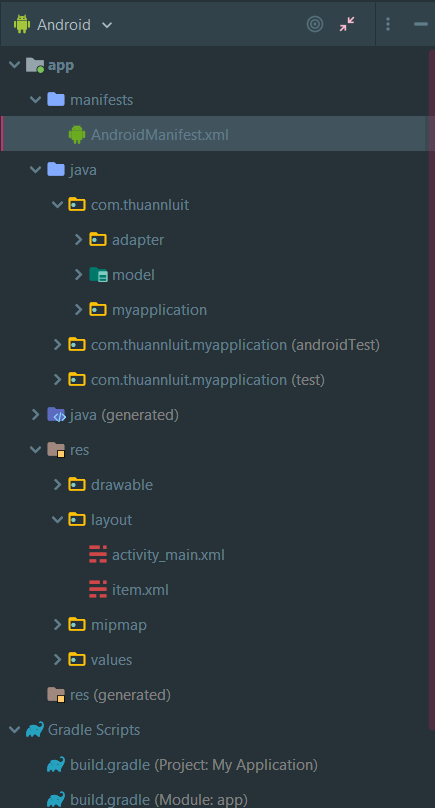
Xử lý thông tin

EmployeeServiceController -> EmployeeService -> EmployeeRepository (để nó gọi hàm có sẵn ra xử lý)

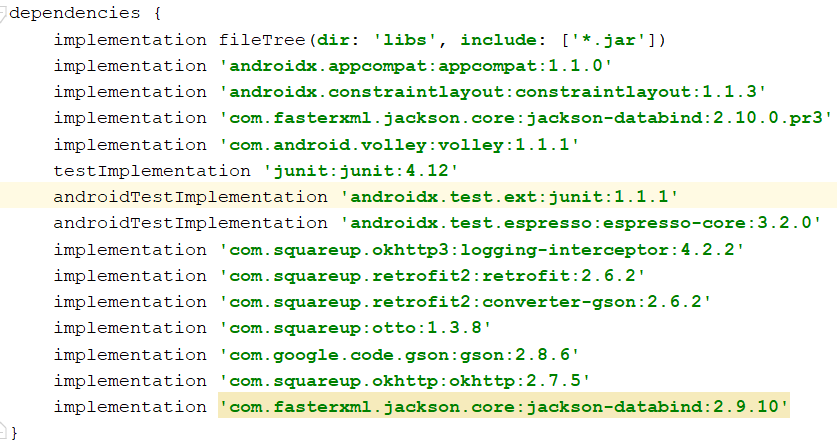
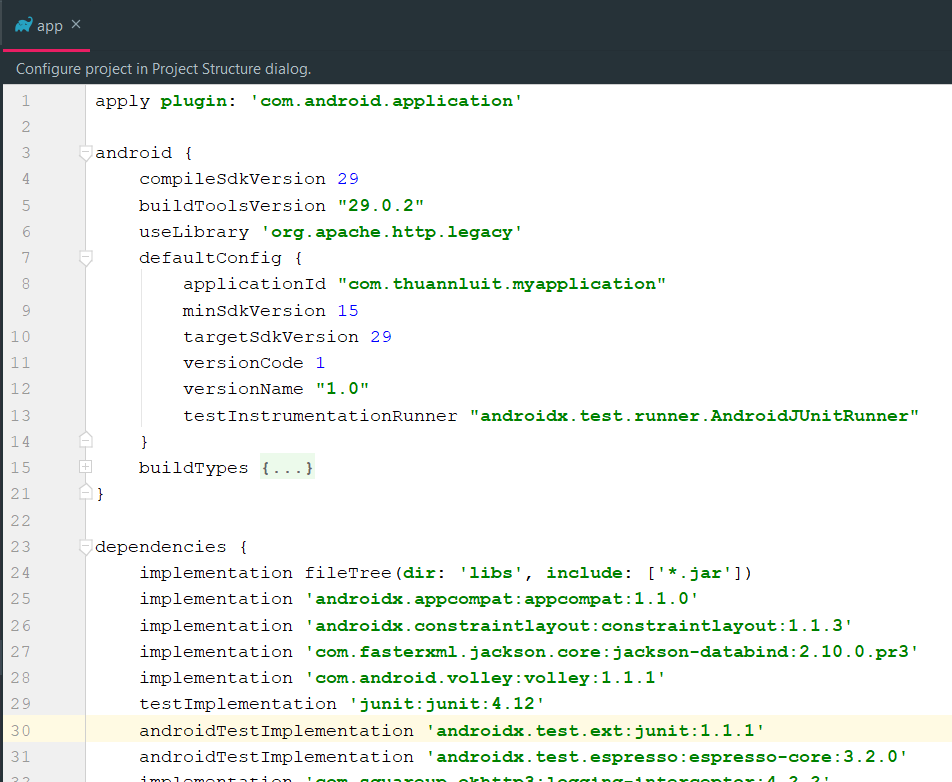
## Client : App quản lý nhân sự cơ bản

Công cụ android studio

Cấu trúc thư mục



File cau hinh



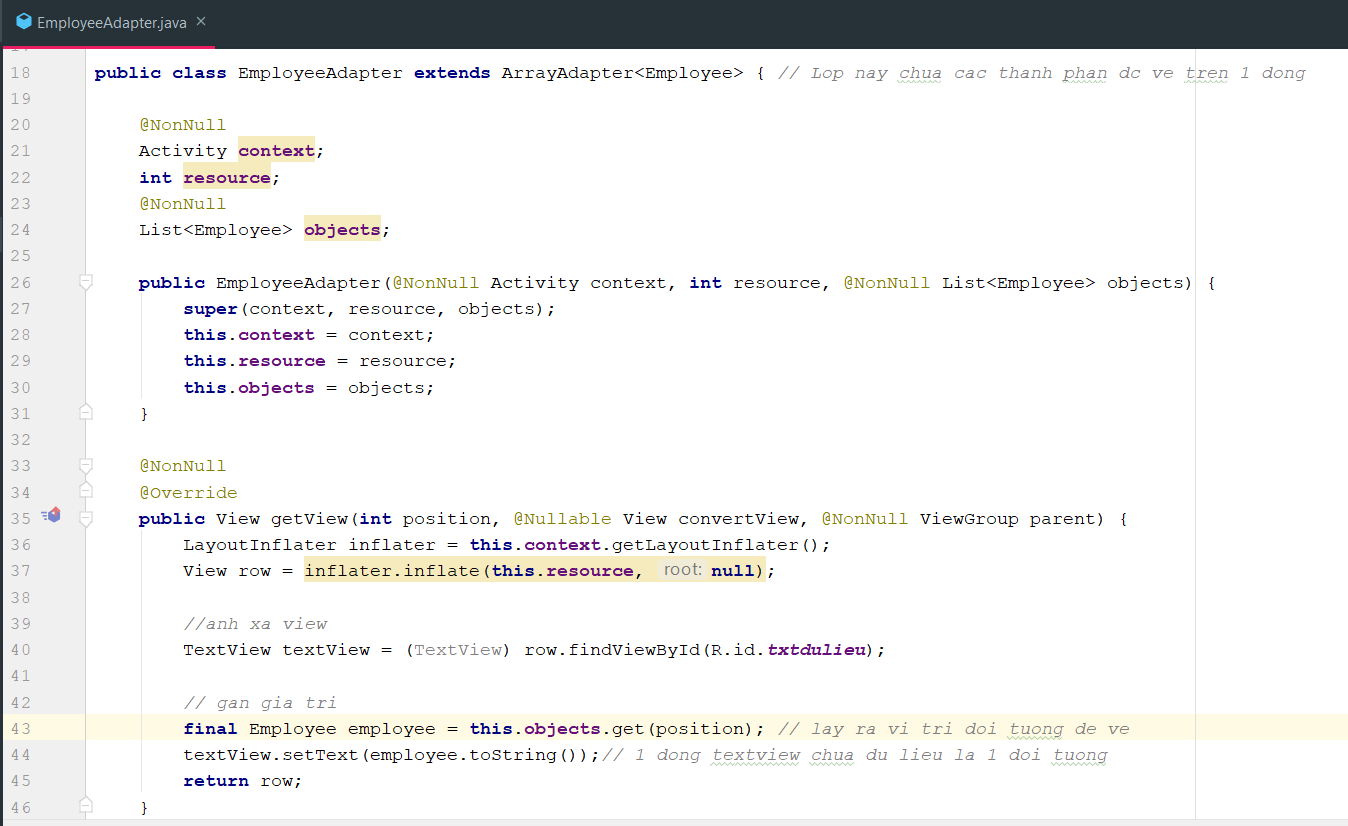
Thư viện OKHTTP là một thư viện cho phép kết nối internet sử dụng giao thức Http một cách dễ dàng và nhanh chóng hơn.

* Kiểm soát kết nối tới server
* Kiểm soát các kết nối không tốt và thử kết nối lại khi có thể
* Nó sẽ thử thay thế server IP address nếu kết nối tới một IP nào đó bị thất bại vào IP thay thế được chuẩn bị sẵn
* Giảm độ trễ của request, giảm size của file cần download
* Tránh lặp lại các request đã được hoàn thành

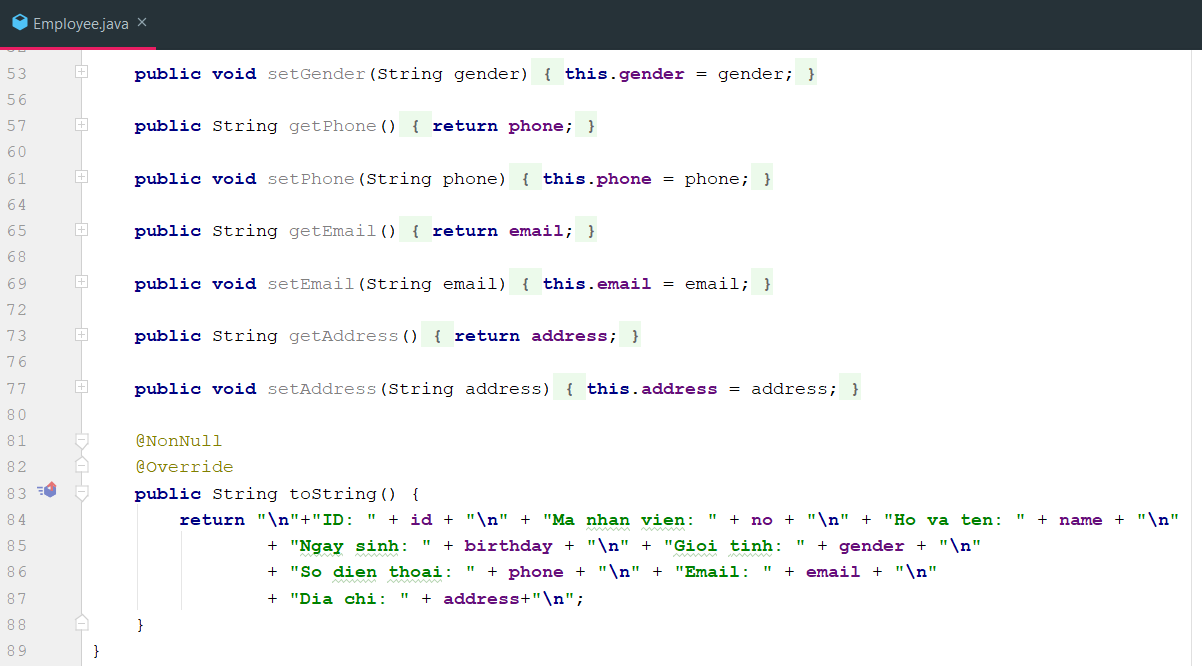
File Manifest



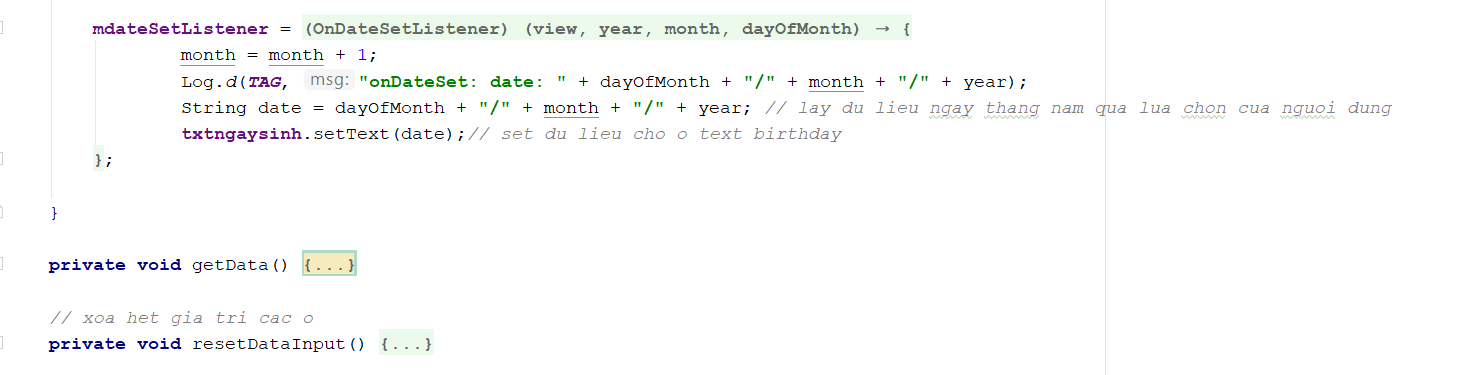
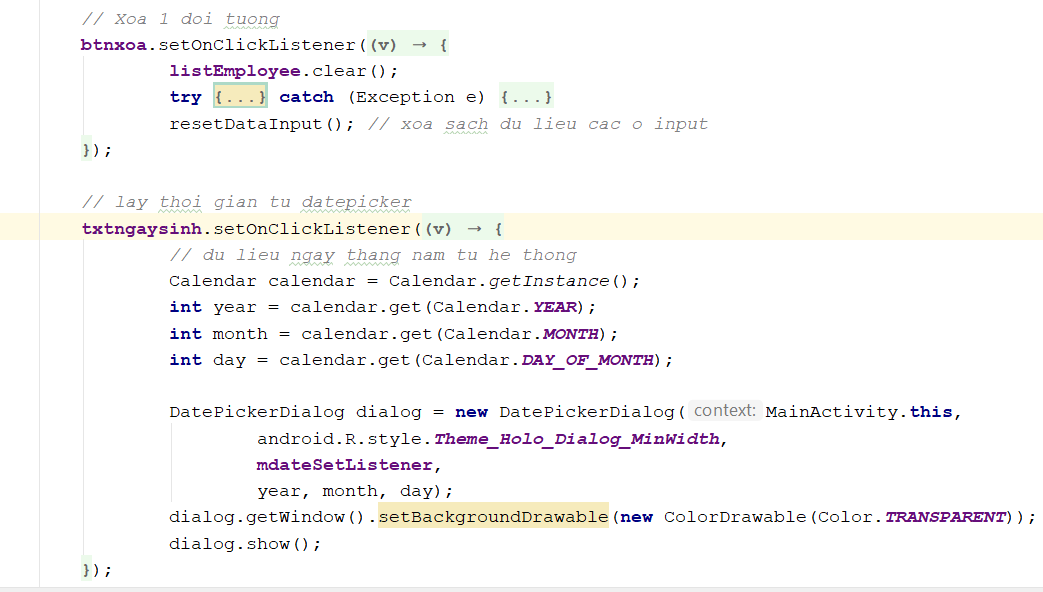
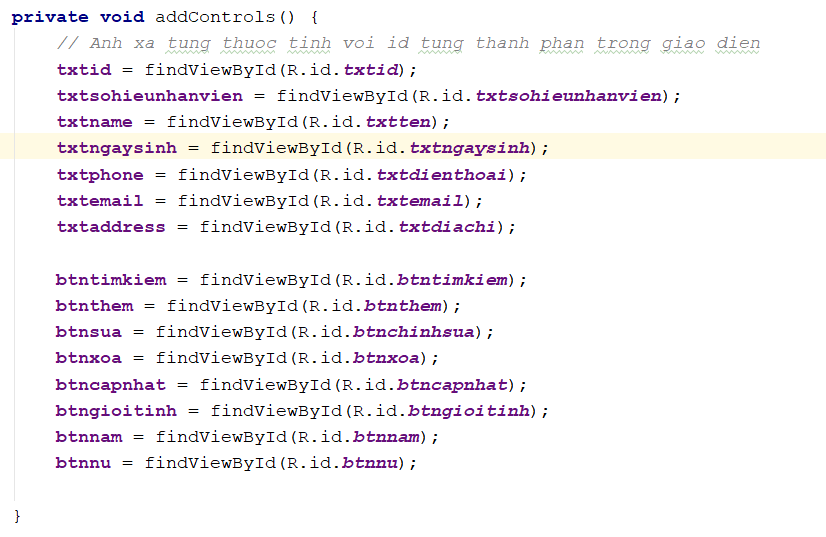
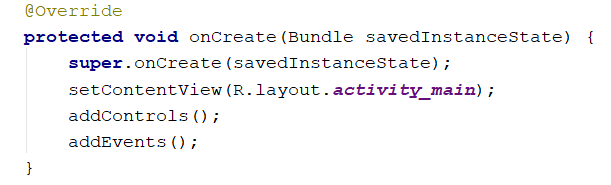
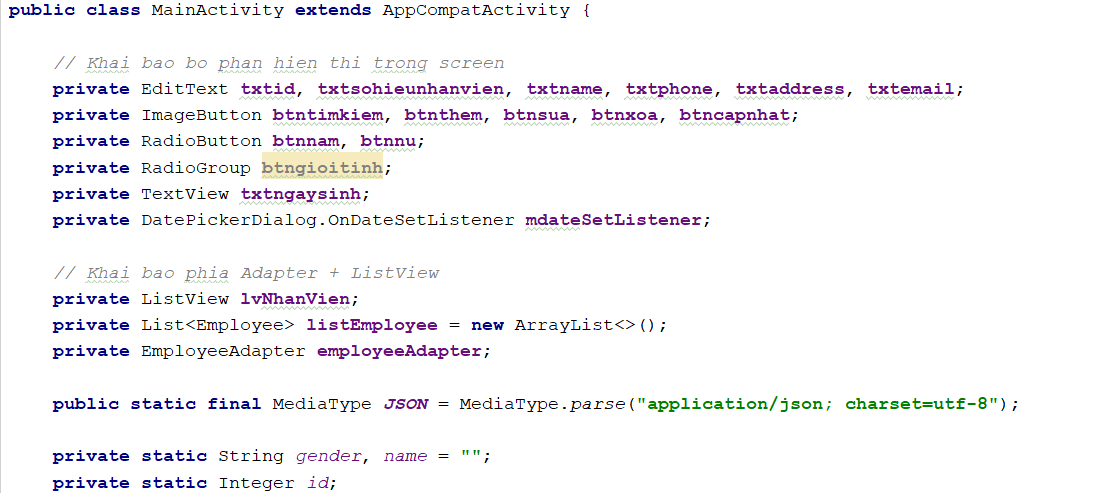
File Adapter



File Entity:

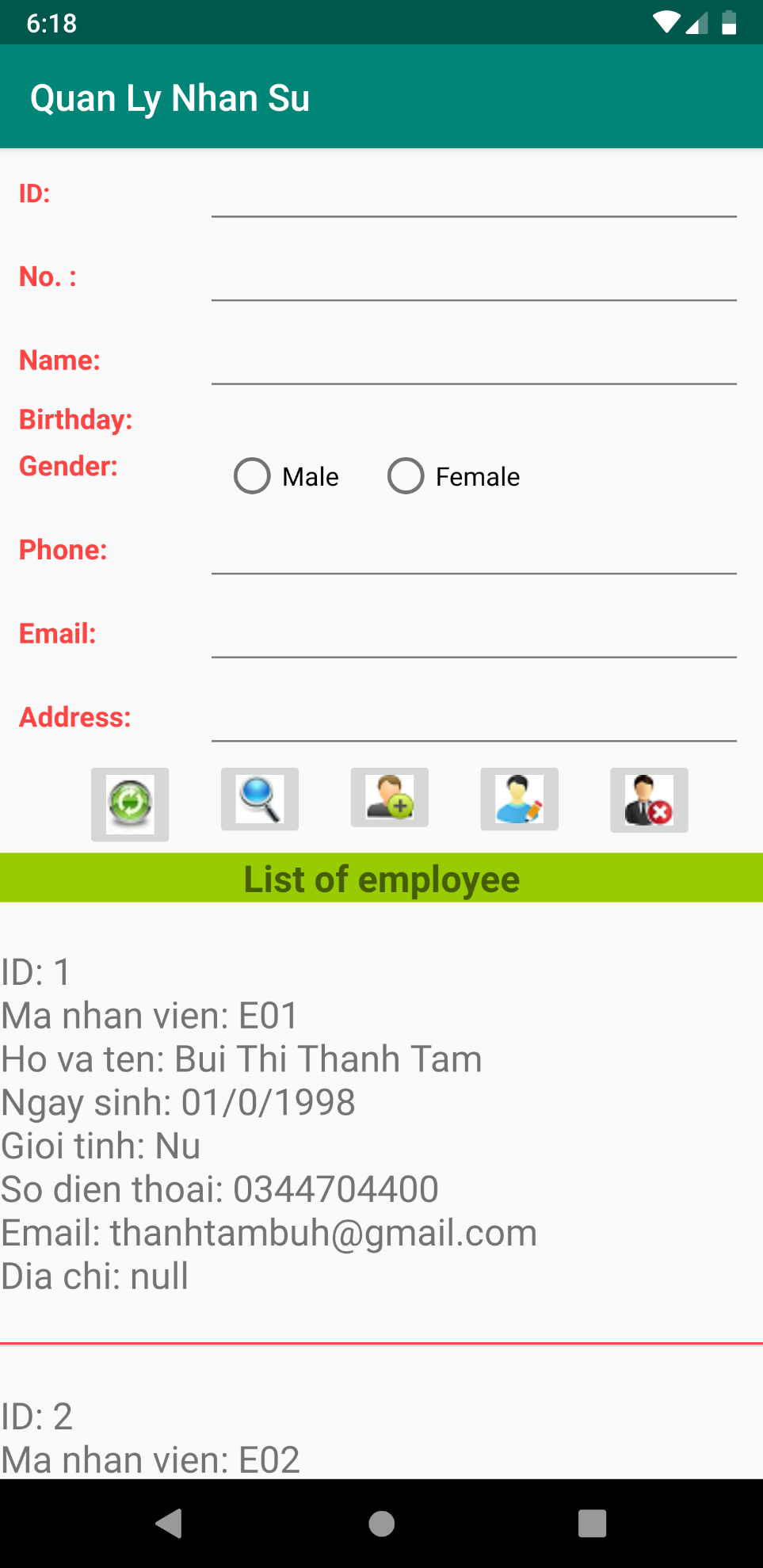


File MainActivity

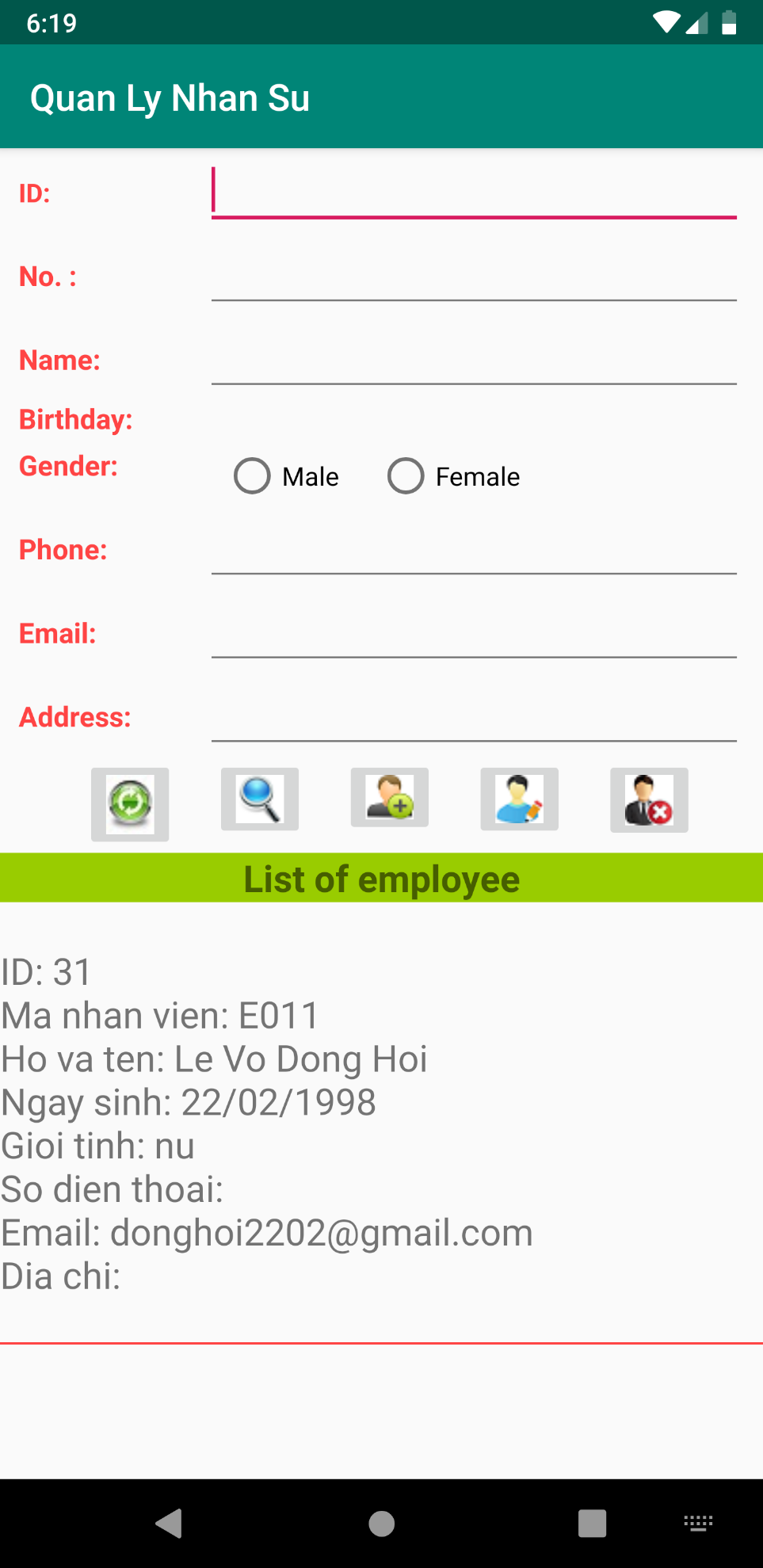


Giao diện

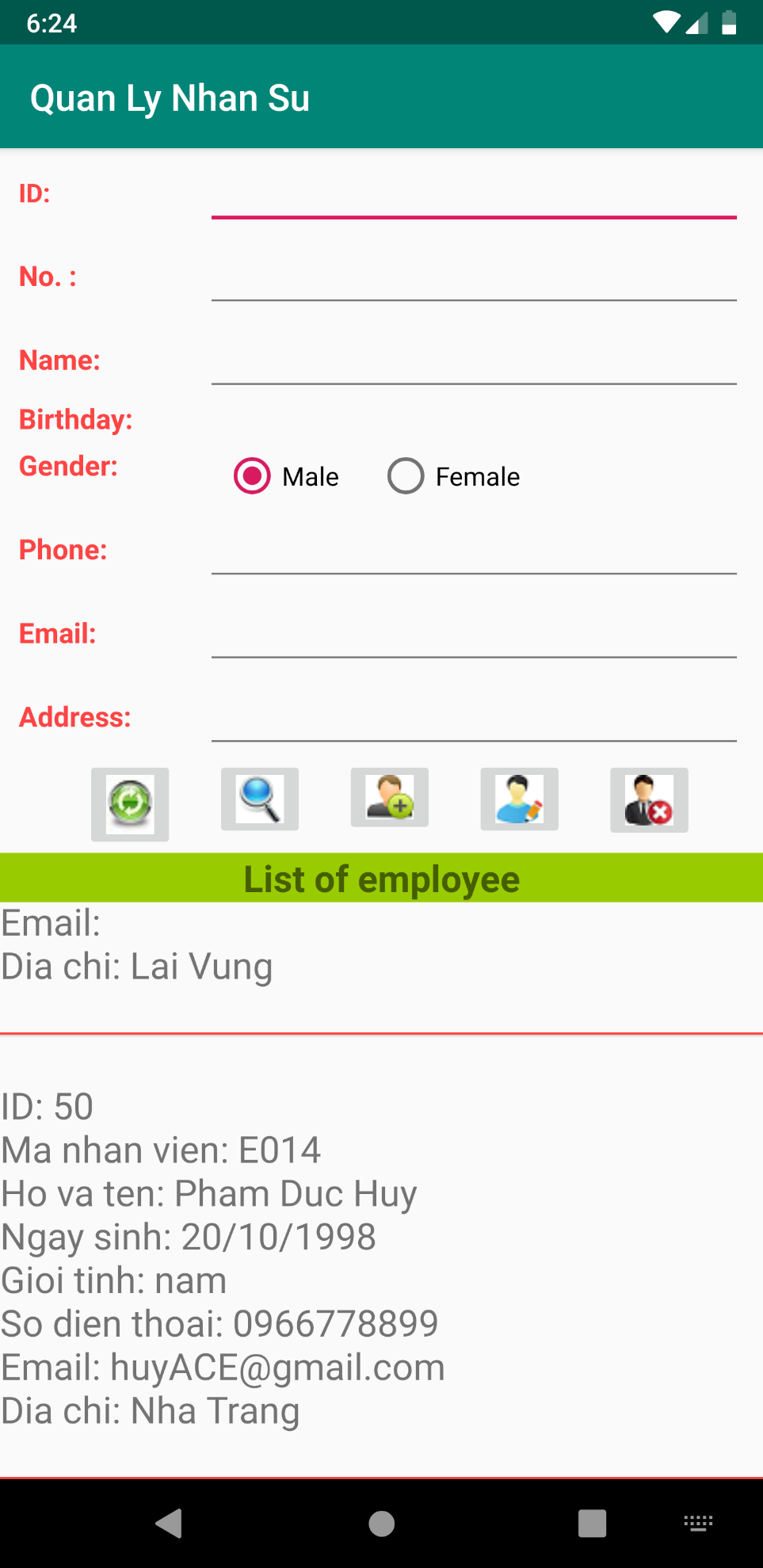
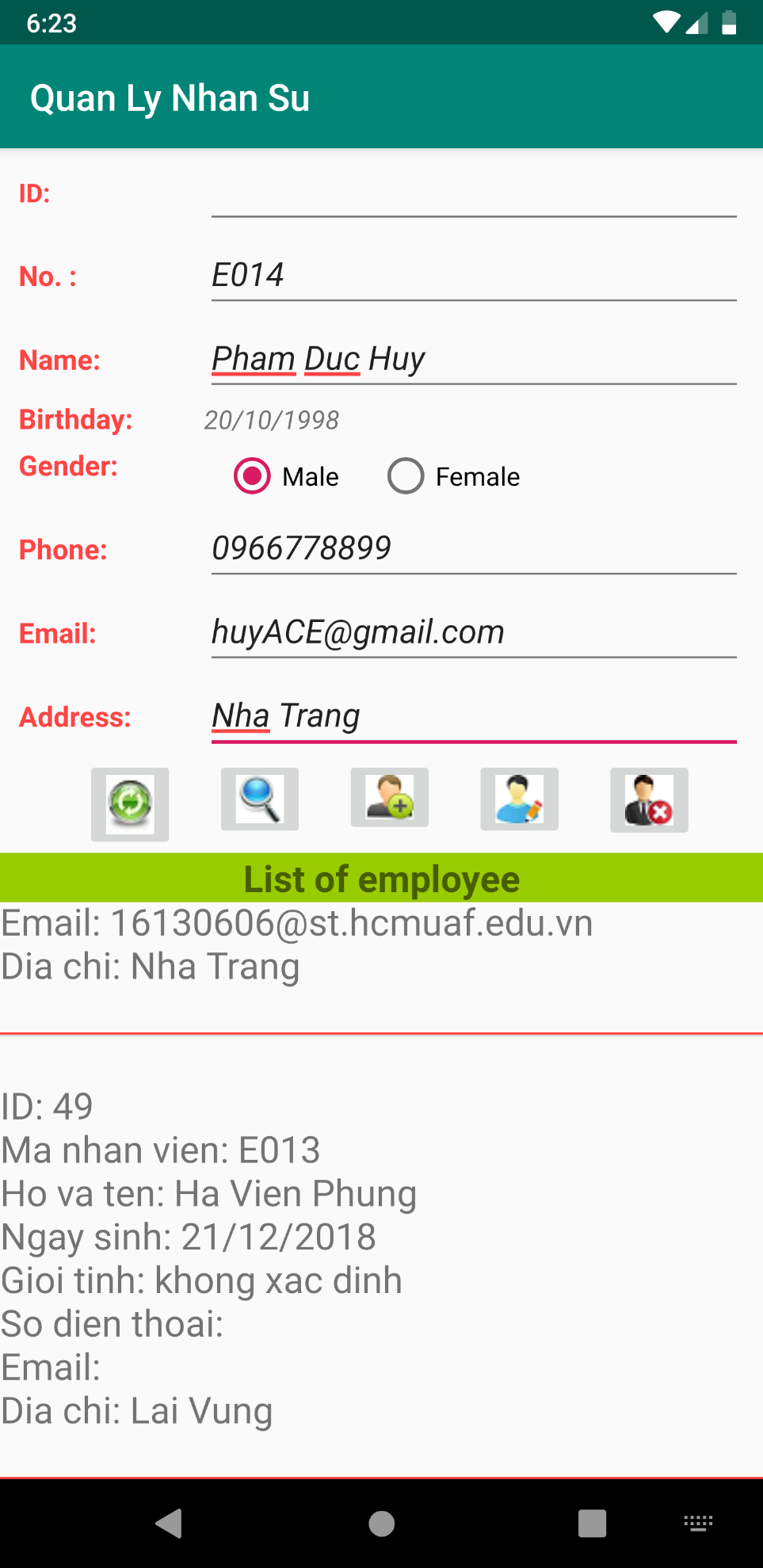
Cập nhật dữ liệu



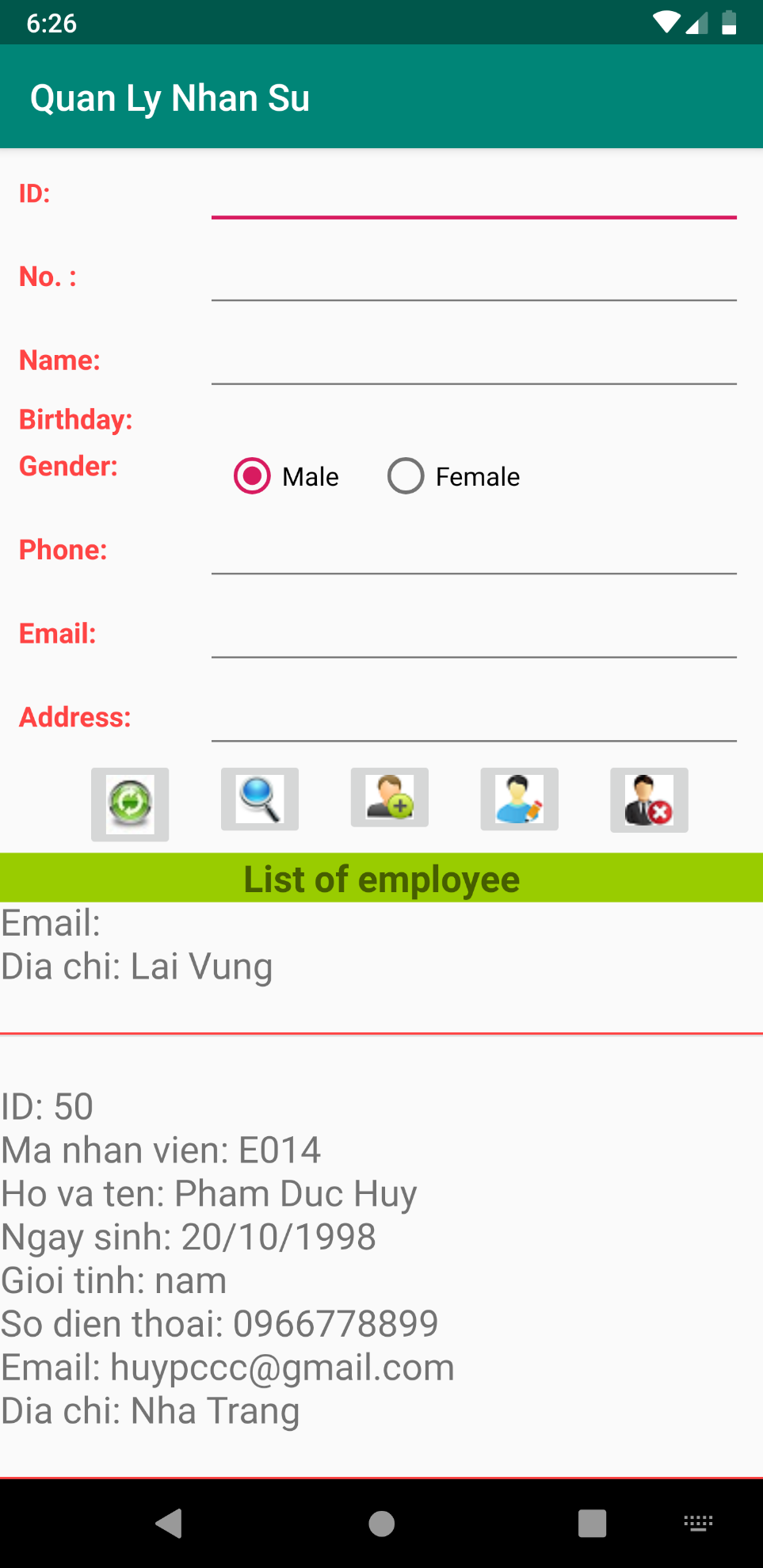
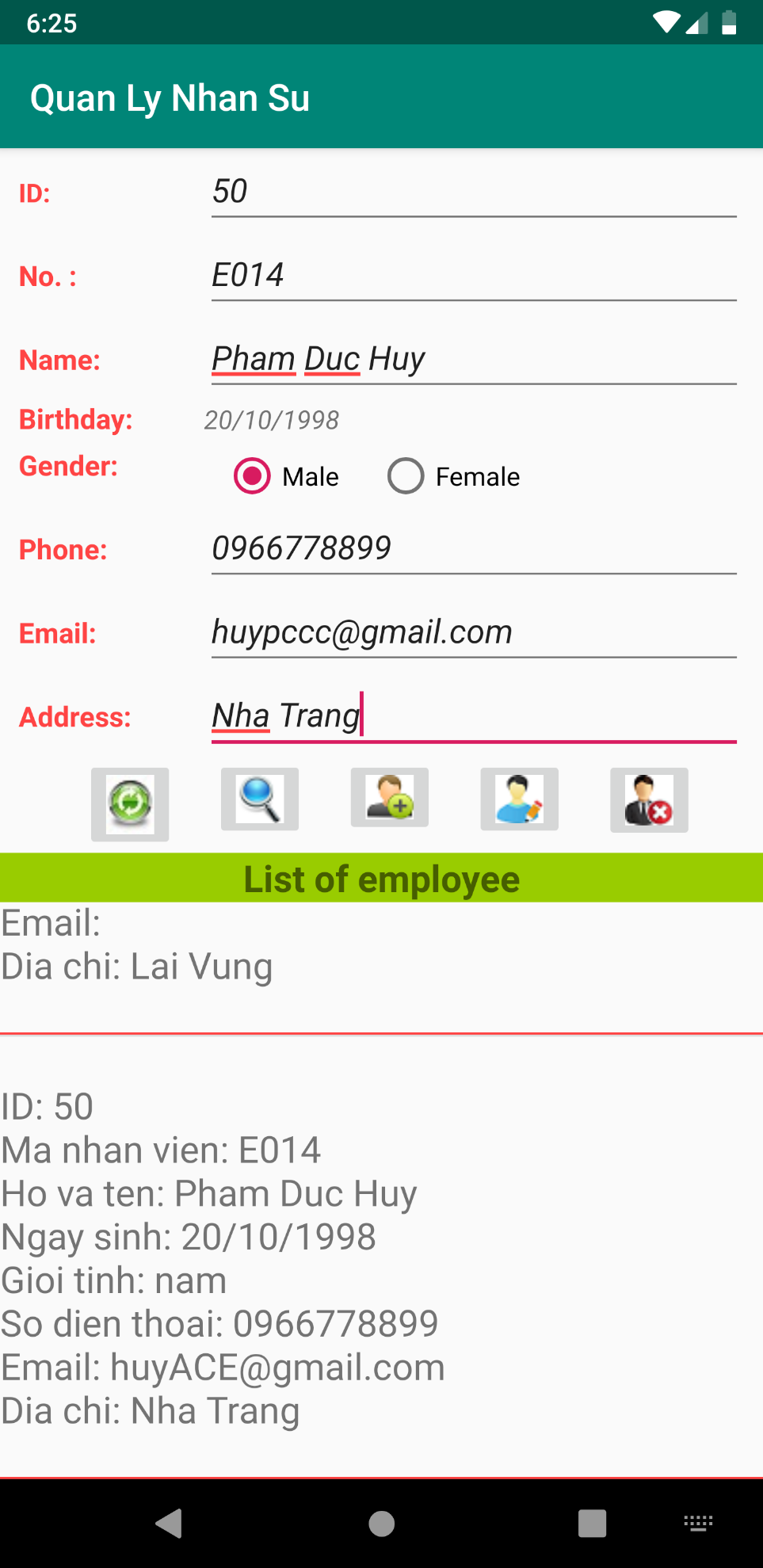
Tim kiếm theo tên



Thêm Nhân Viên



# Chỉnh sửa thông tin



# 

# Tài liệu tham khảo:

<https://en.wikipedia.org/wiki/Web_service>

<https://www.ibm.com/developerworks/webservices/library/ws-restful/>

<https://viblo.asia/p/tim-hieu-ve-restful-web-service-OEqGj5JNM9bL>

<https://o7planning.org/vi/10773/restful-web-service-la-gi>