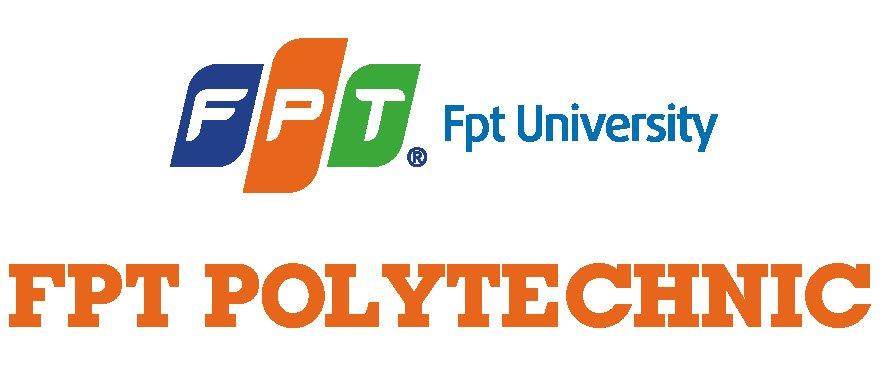
**TRƯỜNG CAO ĐẲNG THỰC HÀNH**

**FPT-POLYTECHNIC TP.HCM**

****

**BÁO CÁO PRO205 – DỰ ÁN 2**

**TÊN ĐỀ TÀI**

**QUẢN LÝ SHIPPER**

**Thực hiện: Nhóm “Quản lý shipper”**

**Thành viên :**

**- Nguyễn Quốc Đạt**

**- Trần Chí Linh**

**- Võ Thành Đạt**

**- Nguyễn Nhật Trường**

**Lớp: LTDD\_6**

**Giảng viên hướng dẫn: Bùi Minh Nhựt**

TP. HỒ CHÍ MINH - 2014

MỤC LỤC

[1. Giới thiệu. 6](#_Toc467152616)

[2. Khảo sát hệ thống 9](#_Toc467152617)

[3. Phân tích hệ thống 11](#_Toc467152618)

[3.1 Chức năng quản lý tài khoản. 11](#_Toc467152619)

[3.2 Chức năng quản lý sản phẩm. 15](#_Toc467152620)

[4. Thiết kế giao diện ứng dụng 15](#_Toc467152621)

[4.1 SplashscreenActivity: 15](#_Toc467152622)

[Xác định yêu cầu 15](#_Toc467152623)

[Thiết kế mẫu 15](#_Toc467152624)

[Mô tả form 15](#_Toc467152625)

[Validation 15](#_Toc467152626)

[Mô tả hoạt động 15](#_Toc467152627)

[4.2 LoginActivity : 16](#_Toc467152628)

[Xác định yêu cầu 16](#_Toc467152629)

[Thiết kế mẫu 16](#_Toc467152630)

[Validation 17](#_Toc467152631)

[4.3 UpdateThongTinChuShopActivity : 19](#_Toc467152632)

[Xác định yêu cầu 19](#_Toc467152633)

[Thiết kế mẫu 19](#_Toc467152634)

[Validation 20](#_Toc467152635)

[4.4 NavDrawerActivity : 21](#_Toc467152636)

[Xác định yêu cầu 21](#_Toc467152637)

[Thiết kế mẫu 21](#_Toc467152638)

[Validation 22](#_Toc467152639)

[4.5 TaoHoaDonActivity : 24](#_Toc467152640)

[Xác định yêu cầu 24](#_Toc467152641)

[Thiết kế mẫu 24](#_Toc467152642)

[Validation 25](#_Toc467152643)

[4.6 QuanLyHoaDonShipperActvity : 26](#_Toc467152644)

[Xác định yêu cầu 26](#_Toc467152645)

[Thiết kế mẫu 26](#_Toc467152646)

[Validation 27](#_Toc467152647)

[4.7 ChiTietHoaDonShipperActivity : 28](#_Toc467152648)

[Xác định yêu cầu 28](#_Toc467152649)

[Thiết kế mẫu 28](#_Toc467152650)

[Validation 30](#_Toc467152651)

[4.8 QuanLyHoaDonChuShopActivity : 30](#_Toc467152652)

[Xác định yêu cầu 30](#_Toc467152653)

[Thiết kế mẫu 30](#_Toc467152654)

[Validation 31](#_Toc467152655)

[4.9 ChiTietHoaDonChuShopActivity : 32](#_Toc467152656)

[Xác định yêu cầu 32](#_Toc467152657)

[Thiết kế mẫu 32](#_Toc467152658)

[Validation 33](#_Toc467152659)

[4.10 MapTrackingActivity : 35](#_Toc467152660)

[Xác định yêu cầu 35](#_Toc467152661)

[Thiết kế mẫu 35](#_Toc467152662)

[Validation 36](#_Toc467152663)

[4.11 SOS dialog : 36](#_Toc467152664)

[Xác định yêu cầu 36](#_Toc467152665)

[Thiết kế mẫu 36](#_Toc467152666)

[Validation 36](#_Toc467152667)

[4.12 PushCoDonHangMoi : 37](#_Toc467152668)

[Xác định yêu cầu 37](#_Toc467152669)

[Thiết kế mẫu 37](#_Toc467152670)

[Validation 37](#_Toc467152671)

[4.13 PushHoanThanhDonHang : 38](#_Toc467152672)

[Xác định yêu cầu 38](#_Toc467152673)

[Thiết kế mẫu 38](#_Toc467152674)

[Validation 38](#_Toc467152675)

[4.14 DanhGia Dialog : 39](#_Toc467152676)

[Xác định yêu cầu 39](#_Toc467152677)

[Thiết kế mẫu 39](#_Toc467152678)

[Validation 40](#_Toc467152679)

[4.15 HoanThanhDonHang Dialog : 40](#_Toc467152680)

[Xác định yêu cầu 40](#_Toc467152681)

[Thiết kế mẫu 40](#_Toc467152682)

[Validation 41](#_Toc467152683)

[5. Thiết kế dữ liệu 41](#_Toc467152684)

[1.1 Mô hình quan hệ 41](#_Toc467152685)

[1.2 Thiết kế chi tiết 45](#_Toc467152686)

[1.2.1 Bảng chủ shop 45](#_Toc467152687)

[1.2.2 Bảng shipper 46](#_Toc467152688)

[1.2.3 Bảng hóa đơn 47](#_Toc467152689)

[1.2.4 Bảng đánh giá 48](#_Toc467152690)

[1.2.5 Bảng SOS 49](#_Toc467152691)

[1.2.6 Bảng loại SOS 50](#_Toc467152692)

[1.2.7 Bảng tình trạng 50](#_Toc467152693)

[6. Thiết kế giao diện ứng dụng 52](#_Toc467152694)

[6.1 Phác thảo giao diện: 52](#_Toc467152695)

[7.1 Thiết kế giao diện quản lý server: 55](#_Toc467152696)

[55](#_Toc467152697)

[7.2 Thiết kế giao diện: 59](#_Toc467152698)

[7.Xây dựng ứng dụng 61](#_Toc467152699)

[7.1 Hệ thống lớp quản lý dữ liệu: 61](#_Toc467152700)

[7.1.1 Lớp Database: 61](#_Toc467152701)

[7.1.2 Lớp quản lý đối tượng : 63](#_Toc467152702)

[7.1.3 Hệ thống Activity 69](#_Toc467152703)

[PlashScreenActivity : 69](#_Toc467152704)

[MainActivity: 71](#_Toc467152705)

[ChiTietHoaDonActivity: 74](#_Toc467152706)

[NavDrawerActivity 80](#_Toc467152707)

[QuanLyHoaDonChuShopActivity: 92](#_Toc467152708)

[TaoHoaDonActivity: 95](#_Toc467152709)

[7.1.4 Hệ thống Respon : 109](#_Toc467152710)

[7.1.5 Hệ Thông Sever: 118](#_Toc467152711)

[10.Kết luận 168](#_Toc467152712)

# Giới thiệu.

Trên thực trạng xã hội hiện nay, mua bán online ngày càng phát triển, các phương thức mua bán và giao hàng cổ điển dần trở nên lạc hậu. Người mua và người bán hàng cần một phương thức bán hàng mới thuận tiện hơn.

Người bán hàng : chỉ cần ở tại shop, quoảng cáo mặt hàng và bán hàng của mình cho những khách hàng dù xa hay gần một cách thuận tiện.

Người mua hàng : chỉ cần ở nhà online trên máy tính hoặc điện thoại, mua hàng và thanh toán trực tuyến.

Chính vì vậy dựa theo thực trạng này chúng tôi phát triển ứng dụng quản lý shipper này để giúp người bán có thể chuyển hàng đến khách hàng một cách nhanh chóng hơn. Người bán tạo đơn hàng bằng ứng dụng này của chúng tôi, úng dụng sẽ xử lý gửi cho shipper của chúng tôi đến nhận hàng và gửi đến người mua một cách nhanh chóng và hoàn toàn theo quy trình tự động.

1. **Chức năng:** 
   * + Chủ shop:

* Tạo tài khoản mới bằng tài khoản google, facebook
* Đăng nhập tài khoản đã tạo
* Tạo đơn hàng

+ Điểm bắt đầu

+ Điểm giao hàng

+ Giá trị món hàng

+ Hình ảnh món hàng

+ Tính phí ship theo quãng đường + giá trị món hàng.

+ Chọn người trả tiền phí ship ( chủ shop hoặc khách hàng )

* Thống kê đơn hàng
* Tracking đơn hàng
* Report shipper
* Xóa đơn hàng.
  + - Shipper:
* Đăng nhập
* Dẫn đường
* Phát tín hiệu SOS
* Thống kê các đơn hàng đã chạy, chưa chạy.
* Thông kê số tiền hàng mà shipper đang giữ + tiền ship trong ngày làm việc. Khi shipper đã nhận đơn hàng thì tiền hàng sẽ được trừ và khi shipper đã giao hàng thì tiền hàng sẽ được thêm vào lại.
* Khách hàng của shop:
* Tracking đơn hàng.
* Các tính năng khác :
* Tự động phân chia đơn hàng.
* Tự động nối đường đi nhiều đơn hàng.
* Lưu lại quoãng đường mà shipper di chuyển trong phiên làm việc.
* Những tính năng có thê phát triển thêm :

-1. Chat với shipper: có thêm send hình ảnh.

-2. Xin phép chuyển đơn hàng cho shipper khác giao.

1. **Điểm mới:**

* Tracking hóa đơn.
* Sử dụng sever xử lý trung gian tự động.
* Hệ thông push notification cập nhật thông tin mới liên tục dù cho app không được mở.

1. **Phạm vi:**

* Trên hệ điều hành android.
* Quản lý shipper, đơn hàng.
* Quản lý khách hàng, đơn hàng.
* Người dùng có nhu cầu ship hàng hóa.
* Người dùng trong địa phận TP.Hồ Chí Minh.

# Khảo sát hệ thống

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | STORY / FEATURE / REQUEST | | | Priority | Business Value | Acceptance Criteria |
| As a/an [User role] | I want to  [Goal] | So that [reason] |
| RQ01 | Người nhận hàng | Có chức năng tracking | Cho người nhận hàng theo dõi đơn hàng | 1 | High | Để theo dõi đừng đi của shipper |
| Để biết khi nào shipper đến giao hàng |
| Để biết ai là người giao hàng cho mình |
| RQ02 | Chủ shop | Có chức năng đăng nhập | Cho người dùng đăng nhập | 1 | High | Để bảo mật tài khoản |
| Để xác thực danh tính |
| Để không lộn với các người dùng khác |
| RQ3 | Có chức năng đăng ký | Có chức năng đăng ký cho người dùng | 1 | High | Để tạo tài khoản mới cho người mới sử dụng. |
| RQ04 | Có chức năng xem thông tin tài khoản của người bán | Cho chủ shop có thể biết thông tin tài khoản | 1 | High | Để quản lý thông tin |
| Để thay đổi thông tin khi cần |
| RQ05 | Có chức năng tìm kiếm địa điểm | Cho chủ shop có thể tìm kiếm địa điểm từ và đến để ship hàng | 1 | High | Để tìm kiếm địa điểm |
| Để xem quoảng đường bao xa và giá tiền gửi |
| Để xem thời gian dự kiến shipper đến nhận hàng |
| RQ06 | Có chức năng tạo hóa đơn | Cho chủ shop nhập vào các thông tin cần thiết về hàng hóa cần gửi | 1 | High | Để nhập thông tin hàng hóa, thông tin người nhận hàng. |
| RQ07 | Có chức năng quản lí hóa đơn | Cho chủ shop có thể quản lý những hóa đơn mà mình đã đặt một cách tổng quan | 1 | High | Để có thể quản lý hóa đơn, tiền bạc và tình trạng của hóa đơn của mình |
| RQ08 | Có chức năng xem chi tiết hóa đơn | Cho chủ shop xem lại thông tin chi tiết mà mình đã đặt | 1 | High | Để có thể kiểm tra lại thông tin của hóa đơn đã đặt có sai xót hay nhầm lẫn gì không |
| RQ09 | Có chức năng xóa hóa đơn | Cho chủ shop hủy yêu cầu của hóa đơn đã đặt hàng | 1 | High | Để hủy yêu cầu gửi hàng khi có sai xót hoặc trường hợp đột xuất nào đó |
| RQ10 |  | Có chức năng đánh giá shipper | Cho chủ shop đánh giá thái độ, nghiệp vu của shipper | 1 | Hight | Để nhận xét về shipper, khiếu nại shipper hoặc tán thưởng shipper |
| RQ11 | Shipper | Có chức năng đăng nhập | Cho người dùng đăng nhập | 1 | High | Để bảo mật tài khoản |
| Để xác thực danh tính |
| Để không lộn với các người dùng khác |
| RQ12 | Có chức năng xem thông tin tài khoản của shipper | Cho shipper có thể biết thông tin tài khoản của mình | 1 | High | Để quản lý thông tin |
| Để có thể kiểm tra xem mình có đăng nhập đúng với tài khoản mình được cấp không. |
| RQ13 | Có chức năng hiển thị hóa đơn và chỉ dẫn đường | Cho shipper biết mình được nhận những hóa đơn nào và đường đi như thế nào | 1 | High | để biết danh sách hóa đơn của mình |
| Để biết điểm giao và nhận hàng |
| Để biết đường đến điểm giao và nhận hàng |
| Để biết có đồng nghiệp nào cầu cứu không |
| RQ14 | Có chức quản lý hóa đơn | Cho shipper quản lý hóa đơn mà mình được nhận một cách tổng quan | 1 | High | Để thống kê đơn hàng trong ngày |
| Để xem số tiền mình làm ra có đúng không |
| Để biết tình trạng đơn hàng mình đang nhận |
| RQ15 | Có chức năng xem chi tiết hóa đơn | Cho shipper biết chi tiết thông tin hóa đơn mà mình được nhận | 1 | High | Để xem chi tiết hàng hóa mình nhận gửi |
| Để biết người gửi và người nhận, cách liên lạc ra sao |
| Để có thể xác nhận khi chắc chắn giao dịch tránh sai xót xảy ra |
| Để biết số tiền mình phải trả cho mặt hàng là bao nhiêu |
| Để biết số tiền ship mình được nhận là bao nhiêu |
| RQ16 | Có chức năng nhận hàng | Cho shipper có thể thông báo đã nhận hàng |  |  | Để xác nhận đã nhận hàng và bắt đầu đi giao hàng |
| RQ17 | Có chức năng hoàn thành hóa đơn | Cho shipper có thể thông báo đã hoàn thành hóa đơn | 1 | High | Để xác nhận hoàn thành hóa đơn khi giao hàng xong |
| RQ18 | Có chức năng SOS | Cho shipper cầu cứu khi gặp nạn | 1 | High | Để shipper gửi tín hiệu cầu cứu khi gặp nạn |

# Phân tích hệ thống

## Chức năng quản lý tài khoản.

1. **User Story “Đăng Nhập”:**

**Mô tả hoạt động:**

* Là form đăng nhập là giao diện giúp người dùng đăng nhập tài khoản của mình trước khi thực hiện các tác vụ liên quan của app, nếu người dùng chưa có tài khoản thì sẽ được tạo tài khoản mới ngay lập tức từ tài khoản google và facebook đăng nhập vào
* Có chứa chức năng tracking hóa đơn của người nhận hàng ngay tại giao diện đăng nhập để người sử dụng có thể dễ dàng sử dụng, tránh rườm ra khi phải tạo tài khoản mới chỉ để dùng 1 vài chức năng.

**Kiểm tra:**

- Không vì mọi việc kiểm tra tính hợp lệ của đăng nhập tài khoản đều thông qua API facebook và google

**Đánh giá:**

Loại tương tác: 1

Quy tắc nghiệp vụ: 1

Thực thể dữ liệu: 2

CRUD: 1

1. **User story “Cập nhật tài khoản”:**

**Mô tả hoạt động:**

* Là form để chủ shop cập nhật thông tin về tài khoản của mình

**Kiểm tra:**

-Kiểm tra bỏ trống tất cả thông tin.

- Kiểm tra không bắt đầu bằng khoảng trắng

- Số điện thoại phải là số từ 10 đến 11 kí tự.

**Đánh giá:**

- Loại tương tác: 1

- Quy tắc nghiệp vụ: 2

- Thực thể dữ liệu: 1

- CRUD: 1

1. **User story “Tìm vị trí”:**

**Mô Tả Hoạt Động:**

* Là giao diện bản đồ giúp chủ shop có thể tìm kiếm địa điểm giao và nhận chính xác trước khi tiến hành tạo một yêu cầu (hóa đơn) đến cho shipper.

**Kiểm tra:**

- Không vì nếu chủ shop chưa chọn được địa điểm thì không thể tạo hóa đơn, tác vụ tạo hóa đơn chỉ bắt đầu khi chủ shop chọn được địa điểm giao và nhận mà mình mong muốn.

**Đánh giá:**

- Loại tương tác: 1

- Quy tắc nghiệp vụ: 1

- Thực thể dữ liệu: 2

- CRUD: 1

1. **User Story “Tạo hóa đơn”:**

**Mô tả hoạt động:**

* Là form để người dùng có thể nhập thông tin chi tiết hàng hóa, người gửi và nhận hàng, các lưu ý trong quá trình giao hàng để shipper biết**.**

**Kiểm tra:**

- Kiểm tra bỏ trống

- Kiểm tra bắt đầu bằng khoảng trắng

- Kiểm tra số điên thoại phải từ 10-11 số

**Đánh giá:**

- Loại tương tác: 1

- Quy tắc nghiệp vụ: 1

- Thực thể dữ liệu: 2

- CRUD: 1

1. **User Story “Đánh giá shipper”:**

**Mô tả hoạt động:**

* Là form để chủ shop có thể đánh gia hoặc tố cáo shipper tới công ty.

**Kiểm tra:**

- Kiểm tra dưới 3 sao phải nêu lý do

**Đánh giá:**

- Loại tương tác: 1

- Quy tắc nghiệp vụ: 1

- Thực thể dữ liệu: 2

- CRUD: 1

1. **User Story “Nhận hàng”:**

Khi shipper nhận hàng và mở ứng dụng nhấn vào nút Nhận Hàng thì tình thái sẽ chuyển đổi

**Kiểm tra:**

- không có

**Đánh giá:**

- Loại tương tác: 1

- Quy tắc nghiệp vụ: 1

- Thực thể dữ liệu: 2

1. **User Story “Hoàn thành đơn hàng”**

Khi shipper đã giao hàng dùng chức năng này để hoàn thành đơn hàng

**Mô tả hoạt động:**

Khi shipper đã giao hàng dùng chức năng này để hoàn thành đơn hàng

**Kiểm tra:**

- không có

**Đánh giá:**

- Loại tương tác: 1

- Quy tắc nghiệp vụ: 1

- Thực thể dữ liệu: 2

- CRUD: 1

## Chức năng quản lý sản phẩm.

# 4. Thiết kế giao diện ứng dụng

## 4.1 SplashscreenActivity:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Xác định yêu cầu | |  |  | | --- | --- | | **Màn hình:** | Intro | | **Người sử dụng:** | * + Người dùng mới | | **Tác vụ:** | * + Giới thiệu sơ lược về sản phẩm | | **Khi nào dùng:** | Chỉ trong 3 lần mở ứng dụng lên đầu tiên. | |
| Thiết kế mẫu | Untitled:Users:intel:Desktop:Screen Shot 2016-11-10 at 8.37.54 PM.png |
| Mô tả form | * Viewpaper   + Name: paper   * Button bỏ qua   + Name: done |
| Validation | * Số lần đăng nhập ứng dụng >= 3. Chuyển thẳng đến trang MainActivity. |
| Mô tả hoạt động | * Viewpaper:   + Dùng tay lướt ngang màn hình, các View đổi tương ứng.   * Nút Bỏ qua: Chuyển đến trang MaiActivity (form đăng nhập). |

## 4.2 LoginActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Login  Người sử dụng :   * shipper * khách hàng   Tác vụ :   * shipper : đăng nhập vào ứng dụng * khách hàng : đăng nhập vào ứng dụng   Dùng khi : Bất kì khi nào muốn đăng nhập vào hệ thống. |
| Thiết kế mẫu | Untitled:Users:intel:Downloads:15050289_1715154238803376_2065886456_n.jpg |
| Mô tả form | * ImageView logo   Name : loginImgLogo   * Button đăng nhập bằng tài khoản google.   Name : loginBtnGoogle   * Button đăng nhập bằng tài khoản facebook.   Name : loginBtnFacebook   * Button theo dõi đơn hàng.   Name : loginBtnTheoDoiDonHang |
| Validation | Kiểm tra tài khoản google hoặc facebook có nằm trong hệ thống không.   * Nếu có : chuyển sang giao diện người dùng * Nếu không : chuyển sáng giao diện nhập thông tin tài khoản mới. |
| Mô tả hoạt động | Nhấn button google hoặc facebook:   * Chuyển sang giao diện đăng nhập google hoặc facebook * Nếu đăng nhập hợp lệ, kiểm tra tài khoản có tồn tại trong hệ thống không. * Chuyển về giao diện người dùng hoặc giao diện nhập thông tin tương ứng (đúng/sai).   Nhấn button theo dõi đơn hàng : chuyển đến giao diện map tranking Activity. |

## 4.3 UpdateThongTinChuShopActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Nhập thông tin  Người sử dụng :   * khách hàng   Tác vụ :   * khách hàng : khi đăng nhập một tài khoản mới. |
| Thiết kế mẫu |  |
| Mô tả form | * Label thông tin người dùng   Name : thongTinLbThongTinNguoiDung   * Edittext Họ và tên   Name : thongTinEdHoTen   * Checkbox nam/nữ   Name : thongTinCbNam  Name : thongTinCbNu   * Edittext ngày sinh   Name : thongTinEdNgaySinh   * Button chọn ngày sinh   Name : thongTinBtnNgaySinh   * Label thông tin shop   Name : thongTinLbShop   * Edittext tên shop   Name : thongTinEdTenShop   * Edittext số điện thoại   Name : thongTinEdSDT   * Edittext địa chỉ   Name : thongTinEdDiaChi   * Button xác nhận   Name : thongTinBtnXacNhan |
| Validation | * Không được bỏ trống tất cả các trường. * Tên không chứa số. * Tuổi lớn hơn 16. * Số điện thoại từ 10-11 số. |
| Mô tả hoạt động | Nhấn button xác nhận : kiểm tra tính hợp lệ của các trường thông tin   * Nếu đúng chuyển sang giao diện Map Activity * Nếu sai thông báo lỗi trên những trường sai. |

## 4.4 NavDrawerActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Map  Người sử dụng :   * khách hàng * Shipper   Tác vụ :   * khách hàng : Xem các shipper đang hoạt động. Tìm kiếm địa điêm. Định vị địa điểm. Có thể sử dụng chức năng tạo đơn hàng. * Shipper : |
| Thiết kế mẫu | Untitled:Users:intel:Desktop:Screen Shot 2016-11-10 at 9.40.54 PM.png  https://scontent-hkg3-1.xx.fbcdn.net/v/t34.0-12/15032457_1715877812064352_320435820_n.png?oh=75a6b5e18de62d41e8ed78d005c0464b&oe=5826824E |
| Mô tả form | MAP :   * Edittext từ   Name : mapEdTu   * Edittext đến   Name : mapEdDen   * Button tạo đơn hàng   Name : mapBtnTaoDonHang   * Button định vị   Name : mapBtnDinhVi  MAP SHIPPER :   * Button cứu hộ   Name : mapShipperSOS   * Button định vị   Name : mapShipperDinhVi |
| Validation | Map :   * Không bỏ trống từ đến |
| Mô tả hoạt động | MAP :   * Khi bấm nút định vị marker chuyển đến vị trí hiện tại. * Khi bấm nút tạo đơn hàng thì kiểm tra đã nhập từ đến. Nếu đã nhập chuyển đến giao diện tạo đơn hàng. Nếu chưa nhập thì báo lỗi.   MAP SHIPPER :   * Khi bấm nút định vị marker chuyển đến vị trí hiện tại. * Khi nhân nút SOS chuyển đến giao diện SOS. |

## 4.5 TaoHoaDonActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Tạo hóa đơn  Người sử dụng :   * Chủ shop   Tác vụ :   * Chủ shop : Tạo đơn hàng vận chuyển hàng hóa |
| Thiết kế mẫu |  |
| Mô tả form | * Button chọn hình ảnh hàng   Name : donHangBtnChonAnh   * ViewPager load ảnh đã chọn   Name : donHangVpLoadAnh   * Edittext tên hàng   Name : donHangEdTenHang   * Edittext từ   Name : donHangEdTu   * Edittext đến   Name : donHangEdDen   * Edittext Khoảng cách   Name : donHangEdKhoangCach   * Edittext Giá tiền hàng   Name : donHangEdTienHang   * TextView Tiền ship   Name : donHangTvTienShip   * TextView Tổng tiền   Name : donHangTvTongTien   * TextView Người trả   Name : không   * Radio button chủ shop và người mua   Name : donHangRdChuShop / donHangRdNguoiMua   * Text Area ghi chú   Name : donHangTaGhiChu   * Button Tạo đơn hàng   Name : donHangBtnTaoDonHang |
| Validation | * Hình : phải chọn ít nhất 1 tấm và nhiều nhất là 3. * Tất cả các edittext không được bỏ trống. * Tiền hàng phải là số, lớn hơn 0. |
| Mô tả hoạt động | * Khi bấm vào nút tạo đơn hàng thì kiểm tra tính hợp lệ của đơn hàng. Nếu thỏa điều kiện thông báo tạo thành công, chuyển sang giao diện Map Activity. Nếu không thỏa điều kiện, báo lỗi trên mỗi trường. |

## 4.6 QuanLyHoaDonShipperActvity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Quản lý hóa đơn shipper  Người sử dụng :   * Shipper   Tác vụ :   * Shipper : Liệt kê và Quản lý các hóa đơn mà shipper được nhận * Khi cần tìm lộ trình các đơn hàng chưa chuyển |
| Thiết kế mẫu |  |
| Mô tả form | * Text View Số Tiền Đang Có   Name: QLHoaDonTvSoTienDangCo   * ListView Hoa Don   Name : QLHoaDonLvHoaDon   * Cell Trong ListView Hóa Đơn:   TextView Tên Món Hàng  Name: QLHoaDonCellTenMonHang  TextView Gía Tiền  Name: QLHoaDonCellGiaTien  TextView Tiền Ship  Name: QLHoaDonCellTienShip   * ImageView Tình Trạng Đơn Hàng   Name: QLHoaDonImgTinhTrang   * Button Chỉ Đường   Name: QLHoaDonBtnChiDuong |
| Validation | Không Có |
| Mô tả hoạt động | * Khi bấm vào cell trên listview sẽ chuyển sang màn hình chi tiết của hóa đơn * Khi Nhấn vào button chỉ đường sẽ chuyển sang màn hình MapShipper |

## 4.7 ChiTietHoaDonShipperActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Chi tiết hóa đơn shipper  Người sử dụng :   * Shipper   Tác vụ :   * Shipper : xem thông tin chi tiết hóa đơn , Hoàn thành hóa đơn. |
| Thiết kế mẫu |  |
| Mô tả form | * Viewpager Hình ảnh hàng hóa   Name: CTHoaDonVpHangHoa   * TextView Tên Hàng   Name: CTHoaDonTvTenHang   * TextView Từ   Name: CTHoaDonTvTu   * TextView Đến   Name: CTHoaDonTvĐến   * TextView Khoảng Cách   Name: CTHoaDonTvKhoangCach   * TextView Tiền hàng   Name: CTHoaDonTvTienHang   * TextView Tiền Ship   Name: CTHoaDonTvTienShip   * TextView Tổng tiền   Name: CTHoaDonTvTongTien   * TextView Người Trả   Name: CTHoaDonTvNguoiTra   * TextView Ghi Chú   Name: CTHoaDonTvGhiChu   * Button Hoàn Thành   Name:CTHoaDonBtnHoanThanh |
| Validation | Không Có |
| Mô tả hoạt động | * Nếu đơn hàng đã chuyển rồi thì nút Button Hoàn Thành được disable * Ngược lại khi bấm vào Button Hoàn Thành thì sẽ hoàn thành đơn hàng |

## 4.8 QuanLyHoaDonChuShopActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Quản lý hóa đơn của Khách Hàng  Người sử dụng :   * Chủ Shop   Tác vụ :   * Chủ Shop : Liệt kê và Quản lý các hóa đơn mà Chủ đã tạo |
| Thiết kế mẫu |  |
| Mô tả form | * Spiner Filter Đơn Hàng   Name: QLHoaDonSpnDonHang   * ListView Hoa Don   Name : QLHoaDonLvHoaDon   * Cell Trong ListView Hóa Đơn:   TextView Tên Món Hàng  Name: QLHoaDonCellTenMonHang  TextView Gía Tiền  Name: QLHoaDonCellGiaTien  TextView Tiền Ship  Name: QLHoaDonCellTienShip   * ImageView Tình Trạng Đơn Hàng   Name: QLHoaDonImgTinhTrang   * Button Chỉ Đường   Name: QLHoaDonBtnChiDuong |
| Validation | Không Có |
| Mô tả hoạt động | * Khi bấm vào cell trên listview sẽ chuyển sang màn hình chi tiết của hóa đơn * Khi chọn filter theo thời gian . listview sẽ sắp xếp hóa đơn theo thời gian * Khi chọn filter theo theo tình trạng vận chuyển thì listview sẽ sắp hóa đơn theo tình trạng là đã vận chuyển và chưa vận chuyển. |

## 4.9 ChiTietHoaDonChuShopActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Chi tiết hóa đơn khách hàng  Người sử dụng :   * Chủ shop   Tác vụ :   * Chủ shop : xem thông tin chi tiết, xóa, cập nhật hóa đơn, đánh giá shipper |
| Thiết kế mẫu |  |
| Mô tả form | * Menu chỉnh sửa hóa đơn   Name: CTHoaDonMnChinhSua   * Menu xóa đơn hàng   Name: CTHoaDonMnXoa   * Viewpager Hình ảnh hàng hóa   Name: CTHoaDonVpHangHoa   * TextView Tên Hàng   Name: CTHoaDonTvTenHang   * TextView Từ   Name: CTHoaDonTvTu   * TextView Đến   Name: CTHoaDonTvĐến   * TextView Khoảng Cách   Name: CTHoaDonTvKhoangCach   * TextView Tiền hàng   Name: CTHoaDonTvTienHang   * TextView Tiền Ship   Name: CTHoaDonTvTienShip   * TextView Tổng Tiền   Name: CTHoaDonTvTongTien   * TextView Người Trả   Name: CTHoaDonTvNguoiTra   * TextView Ghi Chú   Name: CTHoaDonTvGhiChu   * TextView Mã   Name: CTHoaDonTvMa   * TextView Người Ship   Name: CTHoaDonTvNguoiShip   * Button Đánh giá   Name:CTHoaDonBtnDanhGia   * Button Theo dõi đơn hàng   Name: CTHoaDonBtnTracking |
| Validation | Không Có |
| Mô tả hoạt động | * Khi bấm menu cập nhật chuyển đến giao diện Cập nhật Activity. Thông báo đến cho shipper có sự thay đổi hóa đơn và lộ trình. * Khi bấm menu xóa, hiện thông báo xác nhận có muốn xóa không, nếu chọn có thì đơn hàng bị xóa, nếu chọn không thì thông báo được tắt đi, đơn hàng không bị xóa. Thông báo đến cho shipper có sự thay đổi về hóa đơn và lộ trình. * Nút đánh giá chỉ hiện thị khi đơn hàng đã hoàn thành. * Khi bấm vào nút đánh giá thì chuyển qua giao diện feedbackActivity * Nút tracking chỉ hiển thị khi đơn hàng đang vận chuyển. * Khi nhấn nút tracking thì chuyển qua giao diện MapTrackingActivity * Chỉ feedback 1 lần. |

## 4.10 MapTrackingActivity :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Tracking  Người sử dụng :   * Chủ shop * Người mua hàng   Tác vụ :   * Chủ shop : theo dõi đường đi của món hàng. * Người mua hàng: theo dõi đường đi của người mua hàng |
| Thiết kế mẫu | https://scontent-hkg3-1.xx.fbcdn.net/v/t34.0-12/15032457_1715877812064352_320435820_n.png?oh=75a6b5e18de62d41e8ed78d005c0464b&oe=5826824E |
| Mô tả form | * Map   Name: TDDHMapView   * Marker shipper   Name TDDHMarker |
| Validation | Không Có |
| Mô tả hoạt động | * Theo dõi đường đi của đơn hàng |

## 4.11 SOS dialog :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : SOS Dialog  Người sử dụng :   * Shipper   Tác vụ :   * Gửi thông báo SOS cho những shipper khác khi cần hỗ trợ |
| Thiết kế mẫu |  |
| Mô tả form | * TextArea Nội dung SOS   Name: SOSTaNoiDung   * Button Gửi   Name: SOSBtnGui |
| Validation | * Kiểm tra không được bỏ trống nội dung |
| Mô tả hoạt động | * Khi nhấn nút gửi , kiểm tra tính hợp lệ. Nếu hợp lệ thì gửi thông báo SOS cho những shipper khác. Còn không thì báo lỗi chưa nhập. |

## 4.12 PushCoDonHangMoi :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Thông báo có đơn hàng Dialog  Người sử dụng :   * Shipper   Tác vụ :   * Thông báo có đơn hàng mới. Xem chi tiết đơn hàng |
| Thiết kế mẫu | Untitled:Users:intel:Desktop:Screen Shot 2016-11-10 at 9.49.09 PM.png |
| Mô tả form | * TextView có đơn hàng mới   Name : không   * Button Xem chi tiết   Name : DHMBtnXemChiTiet |
| Validation | * Không |
| Mô tả hoạt động | * Khi nhấn nút xem chi tiết, chuyển đến giao diện chi tiết đơn hàng shipper. |

## 4.13 PushHoanThanhDonHang :

|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Thông báo hoàn thanh đơn hàng Dialog  Người sử dụng :   * Chủ shop.   Tác vụ :   * Thông báo đơn hàng đã hoàn thanh. Đánh giá đơn hàng |
| Thiết kế mẫu | Untitled:Users:intel:Desktop:Screen Shot 2016-11-10 at 9.48.13 PM.png |
| Mô tả form | * TextView đơn hàng đã hoàn thành   Name : không   * Button Xem chi tiết   Name : DHHTBtnXemChiTiet   * Button close   Name : DHHTBtnClose |
| Validation | * Không |
| Mô tả hoạt động | * Khi nhấn nút xem chi tiết, chuyển đến giao diện chi tiết đơn hàng. * Khi nhấn nút close, tắt dialog |

## 4.14 DanhGia Dialog :

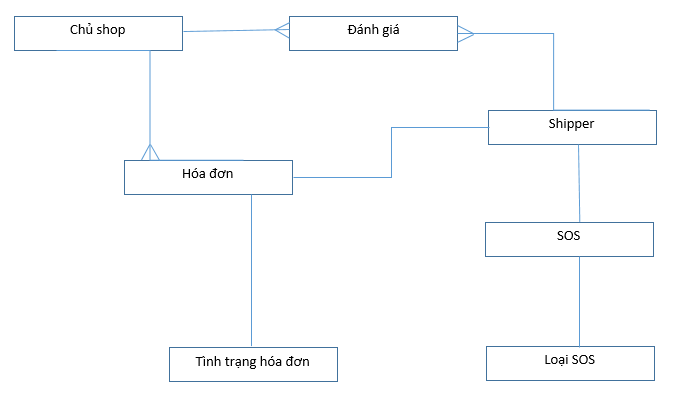
|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Đánh giá hàng Dialog  Người sử dụng :   * Chủ shop.   Tác vụ :   * Đánh giá shipper đã sử lí đơn hàng. |
| Thiết kế mẫu | Untitled:Users:intel:Desktop:Screen Shot 2016-11-10 at 9.47.57 PM.png |
| Mô tả form | * TextView Hãy đánh giá shipper của chúng tôi   Name : không   * ImageView Hình shipper   Name: FbRbHinhShiper   * TextView Người ship   Name : FbRbNguoiShip   * Ratingbar đánh giá   Name : FbRbDanhGia   * Text Area Lý do   Name : FbRbLyDo   * Button Gửi   Name : FbRbGui   * Button close   Name : FbRbClose |
| Validation | * Kiếm tra đã đánh giá trong rattingbar chưa * Không bỏ trống lí do nếu rattingbar dưới 3 sao. |
| Mô tả hoạt động | * Khi bấm nút gửi, kiểm tra điều kiện. Nếu thỏa điều kiên, đánh giá được gửi đi đồng thời thông báo đánh giá thành công. Còn không thì thông báo lỗi. * Khi nhấn nút close, tắt dialog |

## 4.15 HoanThanhDonHang Dialog :

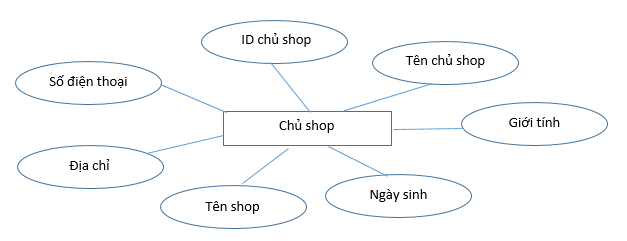
|  |  |
| --- | --- |
| Xác định yêu cầu | Màn hình : Hoàn thành đơn hàng Dialog  Người sử dụng :   * Khách hàng   Tác vụ :   * Kết thúc đơn hàng. |
| Thiết kế mẫu | Untitled:Users:intel:Desktop:Screen Shot 2016-11-10 at 9.45.45 PM.png |
| Mô tả form | * TextView Nhập mã vào đây để hoàn thành   Name : không   * EditText Nhập mã   Name: HTMaHoanThanh   * Button Hoàn Thành   Name : HTBtnHoanThanh |
| Validation | * Không bỏ trống nhập mã hoàn thành |
| Mô tả hoạt động | * Khi bấm nút hoàn thành, kiểm tra điều kiện. Nếu thỏa điều kiên, đánh giá được gửi đi đồng thời thông báo thành công. Còn không thì thông báo lỗi. * Khi nhấn nút close, tắt dialog |

# 5. Thiết kế dữ liệu

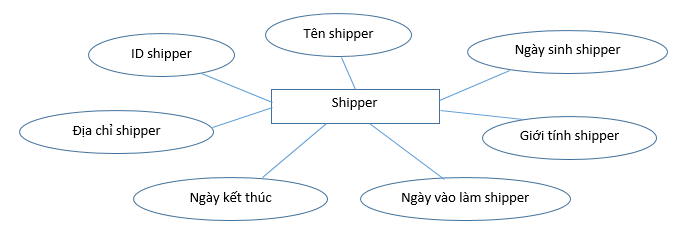
## Mô hình quan hệ



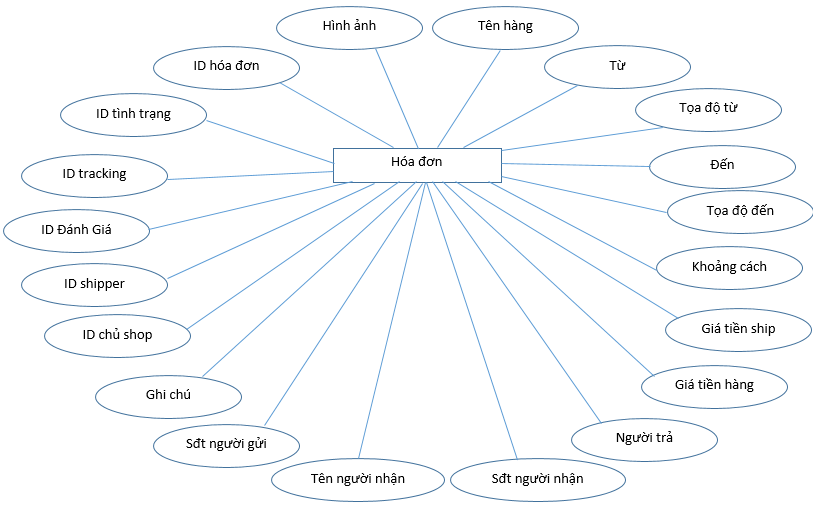
1. **Chi tiết Entities:**
   1. **Chủ shop:**



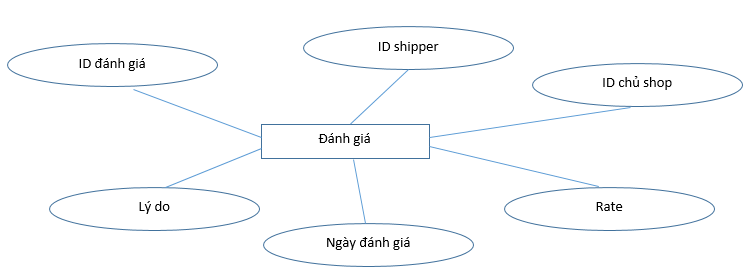
* 1. **Shipper:**



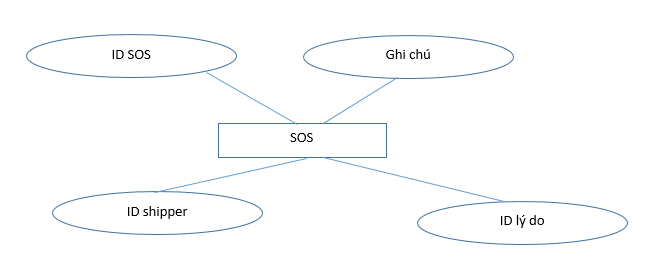
* 1. **Hóa đơn:**



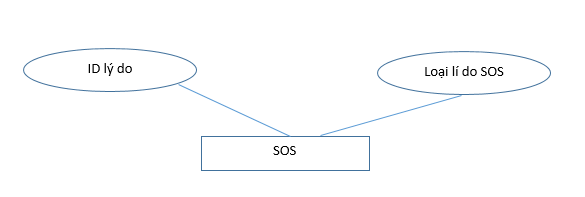
* 1. **Đánh giá :**



* 1. **SOS :**



* 1. **Loại SOS**



## Thiết kế chi tiết

CREATE DATABASE quanlyshipper;

### Bảng chủ shop

#### Bảng chủ shop

Bảng này được sử dụng để lưu trữ thông tin chủ shop

|  |  |  |  |
| --- | --- | --- | --- |
| Cột | Khoá | Kiểu Dữ Liệu | Mô tả |
| **idChuShop** | PK | Text | Mã chủ shop |
| **tenChuShop** |  | Text | Tên chủ shop |
| **gioiTinh** |  | Bool | Giới tính |
| **ngaySinh** |  | Text | Ngày sinh |
| **tenShop** |  | Text | Tên shop |
| **diaChi** |  | Text | Địa chỉ shop |
| **soDienThoai** |  | Text | Số điện thoại shop |

#### Câu lệnh SQL cần thiết

* **Tạo bảng loại sản phẩm**

CREATE TABLE chushop(idChuShop TEXT PRIMARY KEY NOT NULL, tenChuShop TEXT, gioiTinh BOOL, ngaySinh TEXT, tenShop TEXT, diaChi TEXT, soDienThoai TEXT);

* **Thêm loại sản phẩm**

INSERT INTO chushop(tenChuShop, gioiTinh, ngaySinh, tenShop, diaChi, soDienThoai)

VALUES("123", "Nguyễn Quốc Đạt",1, "11/1/1993",”Shop ABC”,”Bùi Hữu Nghĩa Bình Thạnh TP.HCM”,”0989976217”);

### Bảng shipper

#### Bảng shipper

Bảng này được sử dụng để lưu trữ thông tin shipper

|  |  |  |  |
| --- | --- | --- | --- |
| Cột | Khoá | Kiểu Dữ Liệu | Mô tả |
| **idShipper** | PK | Text | Mã shipper |
| **tenShipper** |  | Text | Tên shipper |
| **ngaySinh** |  | Text | Ngày sinh |
| **gioiTinh** |  | Bool | Giới tính |
| **ngayVaoLam** |  | Text | Ngày vào làm việc |
| **ngayKetThuc** |  | Text | Ngày nghỉ việc |
| **diaChi** |  | Text | Địa chỉ của shipper |

#### Câu lệnh SQL cần thiết

* **Tạo bảng sản phẩm**

CREATE TABLE shipper(idShipper TEXT PRIMARY KEY NOT NULL, tenShipper TEXT, ngaySinh TEXT, gioiTinh bool, ngayVaoLam TEXT, ngayKetThuc TEXT, diaChi TEXT);

* **Thêm sản phẩm**

INSERT INTO shipper (tenShipper, ngaySinh, gioiTinh, ngayVaoLam, ngayKetThuc, diaChi)

VALUES("Võ Thành Đạt", "11/2/1995", 1, "11/1/2016", "", "65 Đinh Tiên Hoàng Bình Thạnh TP.HCM");

### Bảng hóa đơn

#### Bảng hóa đơn

Bảng này được sử dụng để lưu trữ thông tin hóa đơn

|  |  |  |  |
| --- | --- | --- | --- |
| Cột | Khoá | Kiểu dữ liệu | Mô tả |
| IdHoaDon | PK | Int | Mã hóa đơn |
| hinhAnh |  | Text | Ảnh hàng hóa |
| tenHang |  | Text | Tên hàng hóa |
| tu |  | Text | Điểm nhận hàng |
| toaDoTu |  | Text | Tọa độ điểm nhận hàng |
| den |  | Text | Điểm giao hàng |
| toaDoDen |  | Text | Tọa độ điểm giao hàng |
| khoangCach |  | Double | Khoảng cách giữa điểm giao và nhận hàng |
| giaTienHang |  | Int | Giá tiền hàng |
| giaTienShip |  | Int | Giá tiền ship |
| nguoiTra |  | Bool | Người trả tiền ship |
| ghiChu |  | Text | Ghi chú |
| tenNguoiGui |  | Text | Tên người gửi |
| sdtNguoiGui |  | Text | Số điện thoại người gủi |
| tenNguoiNhan |  | Text | Tên người nhận |
| sdtNguoiNhan |  | Text | Số điện thoại người nhận |
| idShipper | FK | Text | Mã shipper |
| idChuShop | FK | Text | Mã chủ shop |
| idTinhTrang | FK | Int | Mã tình trạng |

#### Câu lệnh SQL cần thiết

* **Tạo bảng tài khoản**

CREATE TABLE hoadon(IdHoaDon TEXT PRIMARY KEY NOT NULL, hinhAnh TEXT, tenHang TEXT, tu TEXT, toaDoTu TEXT, den TEXT, toaDoDen TEXT, khoangCach Double, giaTienHang INT, giaTienShip INT, nguoiTra BOOL, ghiChu TEXT, tenNguoiGui TEXT, sdtNguoiGui TEXT, tenNguoiNhan TEXT, sdtNguoiNhan TEXT, idShipper TEXT FOREIGN KEY REFERENCES shipper(idShipper), idChuShop TEXT FOREIGN KEY REFERENCES chushop(idChuShop), idTinhTrang INT FOREIGN KEY REFERENCES tinhtrang(idTinhTrang));

* **Thêm tài khoản**

INSERT INTO TaiKhoan(hinhAnh, tenHang, tu, toaDoTu, den, toaDoDen, khoangCach, giaTienHang, giaTienShip, nguoiTra, ghiChu, tenNguoiGui, sdtNguoiGui, tenNguoiNhan, sdtNguoiNhan, idShipper, idTinhTrang)

VALUES(“dasdsadsadsad.png”, “Hàng 1”, “vũ tùng”, “56756756756-5675675676”, “Bùi Hữu nghĩa”, “56756756756-56765765”, 1.2, 20000,10000,true,”hàng dễ vỡ”,”Nguyễn Quốc Đạt”,”0989976217”,”Trần Chí Linh”,”0989976218”,”SP1”,”CS2”,0);

### Bảng đánh giá

#### Bảng đánh giá

Bảng này được sử dụng để chỉ ra mã tài khoản, mã sản phẩm, số lượng, tổng tiền

|  |  |  |  |
| --- | --- | --- | --- |
| Cột | Khoá | Kiểu dữ liệu | Mô tả |
| idDanhGia | PK | Int | Mã đánh giá |
| rate |  | Int | Mức đánh giá |
| lyDo |  | Text | Lý do |
| idChuShop | FK | Text | Mã chủ shop – người đánh giá |
| idShipper | FK | Text | Mã shipper – người nhận đánh giá |

#### Câu lệnh SQL cần thiết

* **Tạo bảng giao dịch**

CREATE TABLE danhgia(idDanhGia TEXT PRIMARY KEY NOT NULL, rate INT, lyDo TEXT, idChuShop TEXT FOREIGN KEY REFERENCES chushop(idChuShop), idShipper TEXT FOREIGN KEY REFERENCES shipper(idShipper));

* **Thêm giao dịch**

INSERT INTO danhgia(rate, lyDo, idChuShop, idShipper)

VALUES(3, "Đến không đúng giờ", "CS2", "SP3");

### Bảng SOS

#### Bảng SOS

Bảng này được sử dụng để lưu trữ lịch sử yêu cầu cứu hộ của shipper

|  |  |  |  |
| --- | --- | --- | --- |
| Cột | Khoá | Kiểu dữ liệu | Mô tả |
| idSOS | PK | Int | Mã cứu hộ |
| ghiChu |  | Text | Ghi chú |
| idLyDoSOS |  | Int | Mã lý do |
| idShipper | FK | Text | Mã shipper |

#### Câu lệnh SQL cần thiết

* **Tạo bảng giao dịch**

CREATE TABLE sos(idSOS TEXT PRIMARY KEY NOT NULL, ghiChu TEXT, idLyDoSOS INT, idShipper TEXT FOREIGN KEY REFERENCES shipper(idShipper));

* **Thêm giao dịch**

INSERT INTO sos(ghiChu, idLyDo, idShipper)

VALUES(“”,1,"SP3");

### Bảng loại SOS

#### Bảng loại SOS

Bảng này được sử dụng để chỉ ra loại cầu cứu mà shipper gặp phải

|  |  |  |  |
| --- | --- | --- | --- |
| Cột | Khoá | Kiểu dữ liệu | Mô tả |
| idLyDoSOS | PK | Int | Mã lý do |
| LoaiLyDoSOS |  | TEXT | Chi tiết lý do |

#### Câu lệnh SQL cần thiết

* **Tạo bảng giao dịch**

CREATE TABLE loaisos(idLyDoSOS INT PRIMARY KEY NOT NULL, LoaiLyDoSOS TEXT);

* **Thêm giao dịch**

INSERT INTO loaisos(LoaiLyDoSOS)

VALUES("Tông xe");

### Bảng tình trạng

#### Bảng tình trạng

Bảng này được sử dụng để lưu

|  |  |  |  |
| --- | --- | --- | --- |
| Cột | Khoá | Kiểu dữ liệu | Mô tả |
| idTinhTrang | PK | Int | Mã tình trạng |
| loaiTinhTrang |  | TEXT | Loại tình trạng |

#### Câu lệnh SQL cần thiết

* **Tạo bảng giao dịch**

CREATE TABLE tinhtrang(idTinhTrang INT PRIMARY KEY NOT NULL, loaiTinhTrang TEXT);

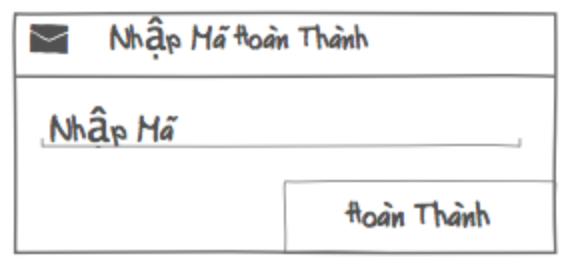
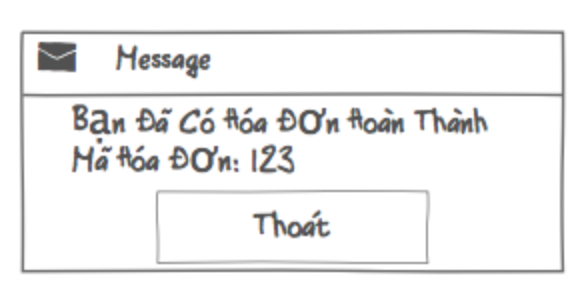
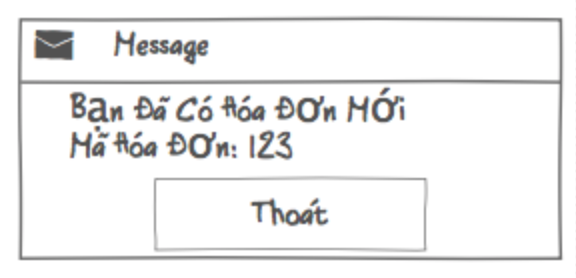
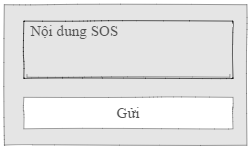
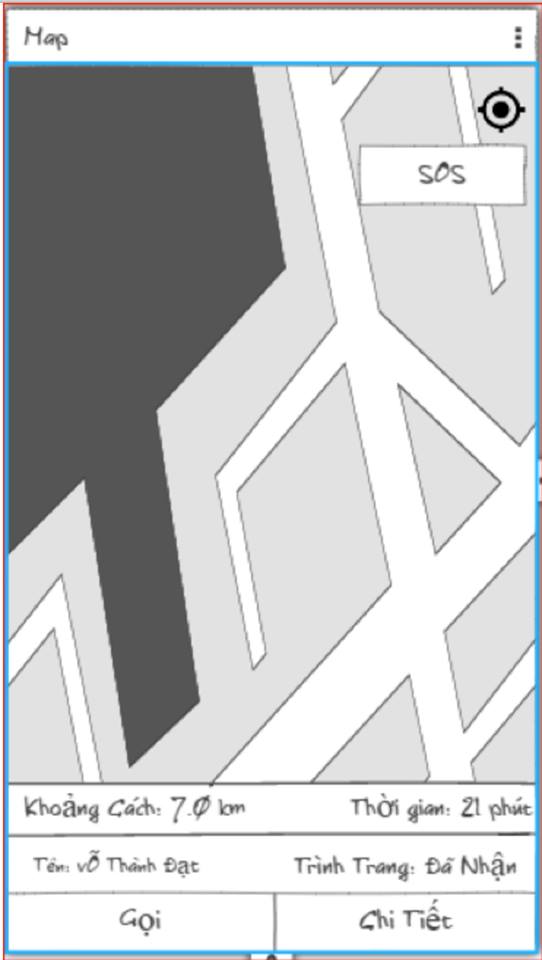
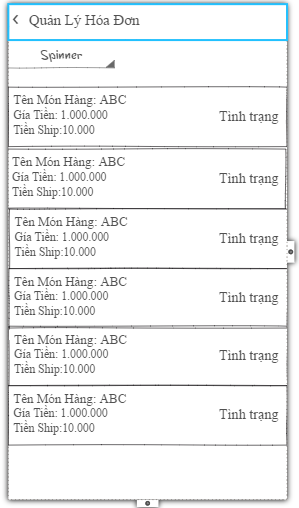
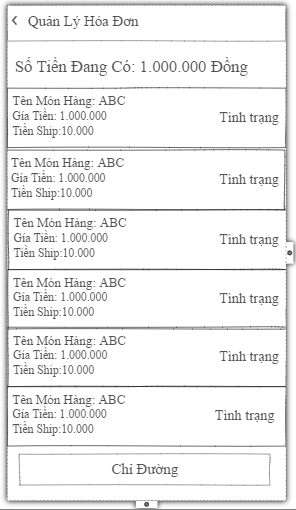
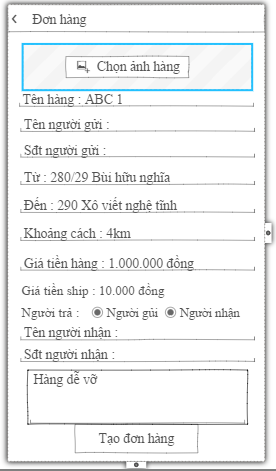
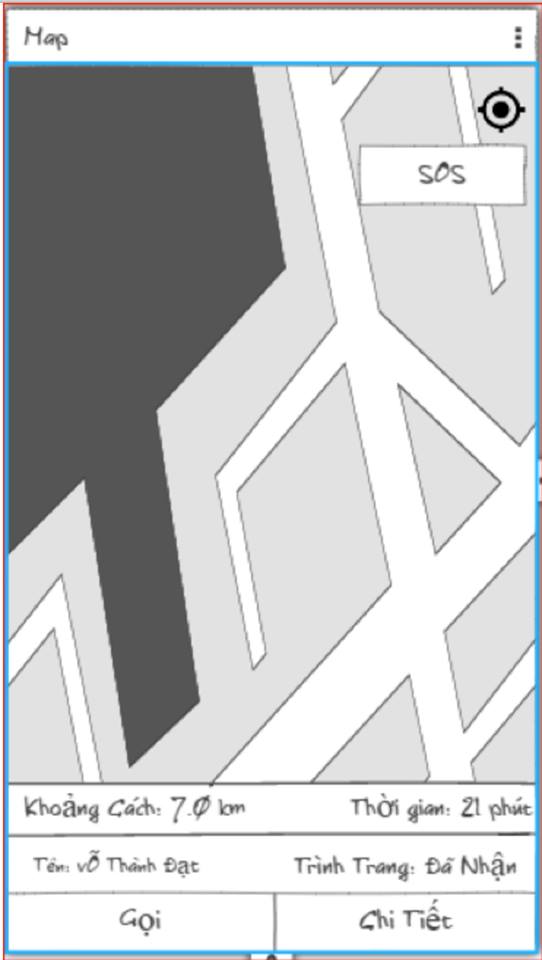
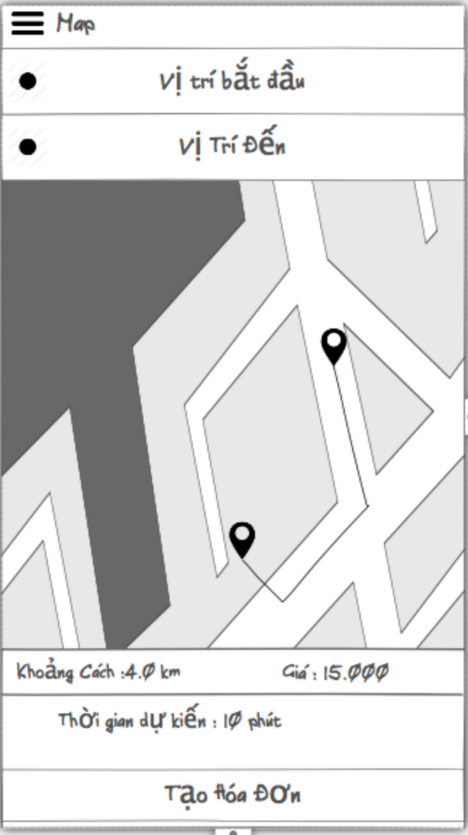
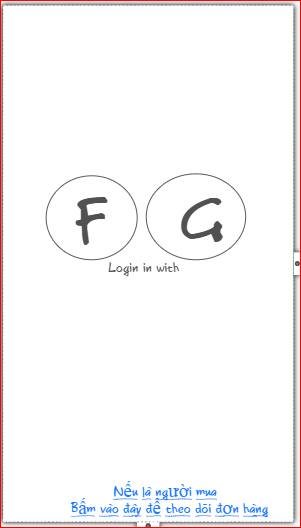
* **Thêm giao dịch**

INSERT INTO danhgia(loaiTinhTrang)

VALUES(“Đã nhận hàng”);

# 6. Thiết kế giao diện ứng dụng

## Phác thảo giao diện:

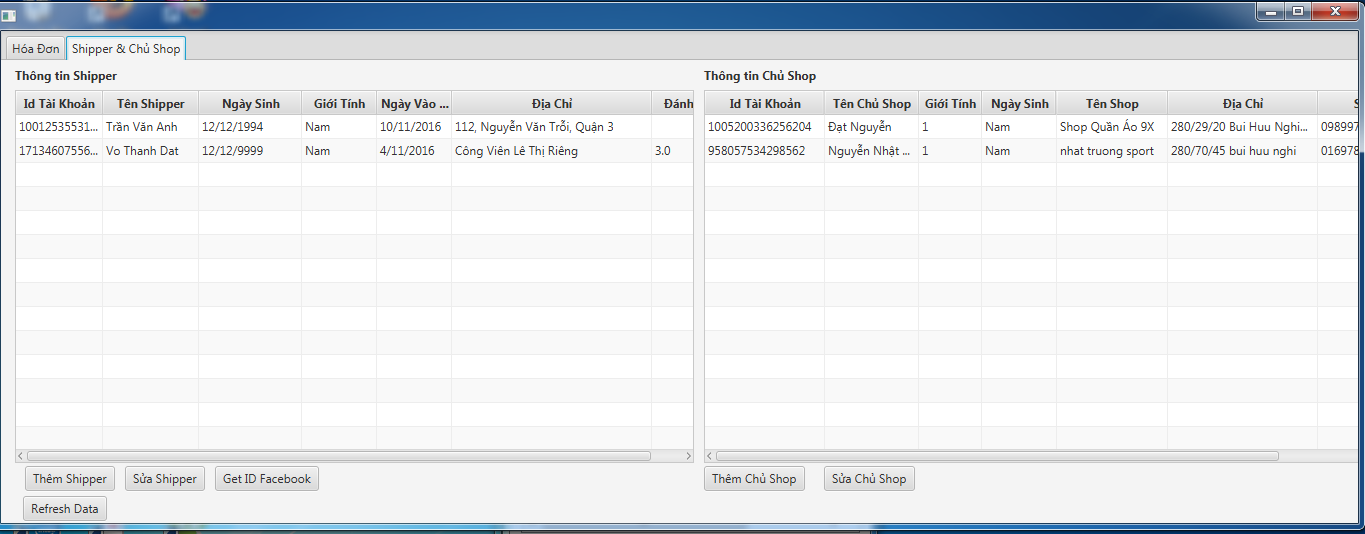
1. 

## Thiết kế giao diện quản lý server:

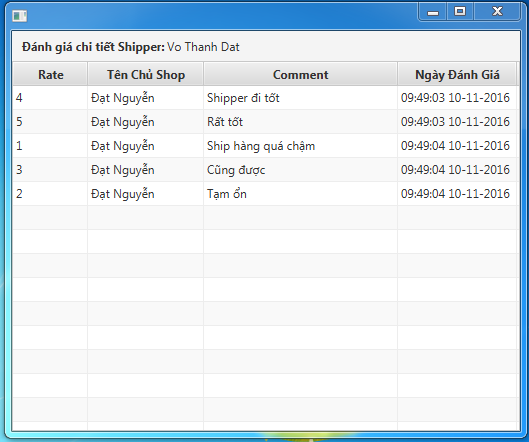
Hóa đơn:

## C:\Users\ADMIN\Desktop\Hoa don.PNG

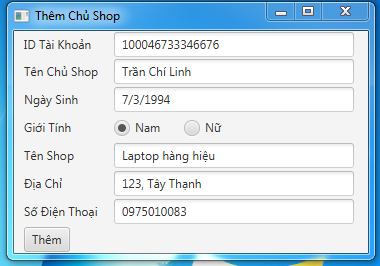
Shipper và chủ shop :



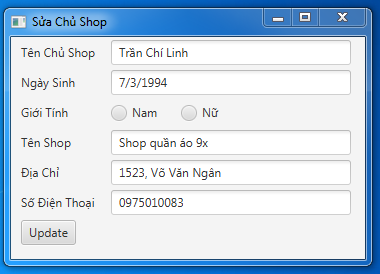
Đánh giá shipper :



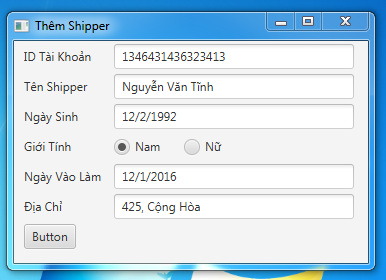
Thêm chủ shop :



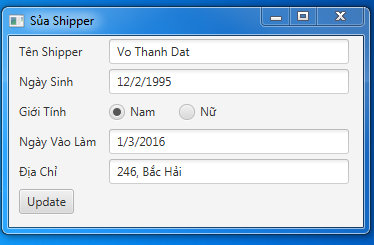
Sửa chủ shop :



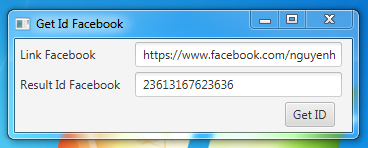
Thêm shipper :



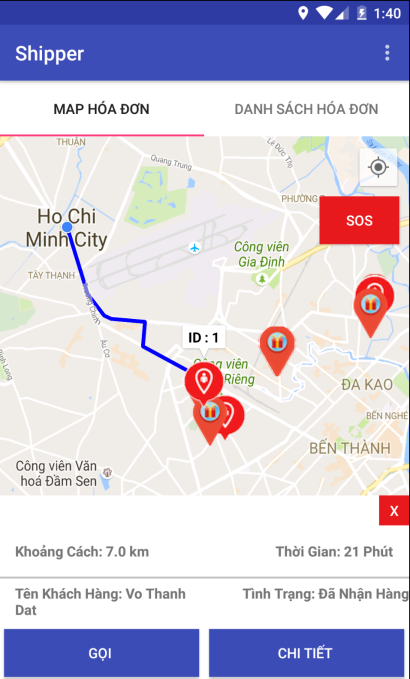
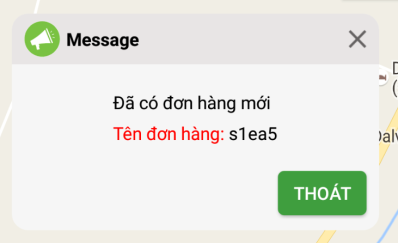
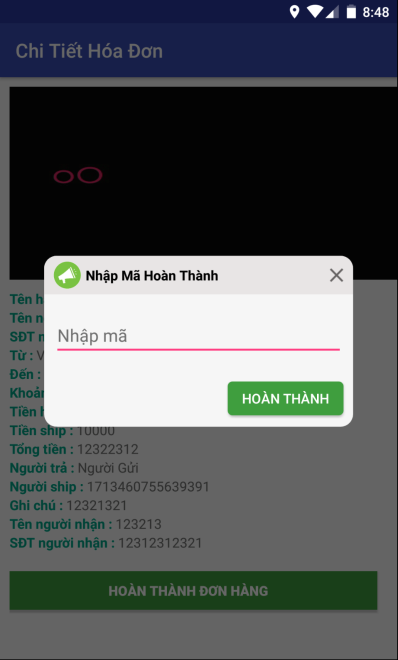
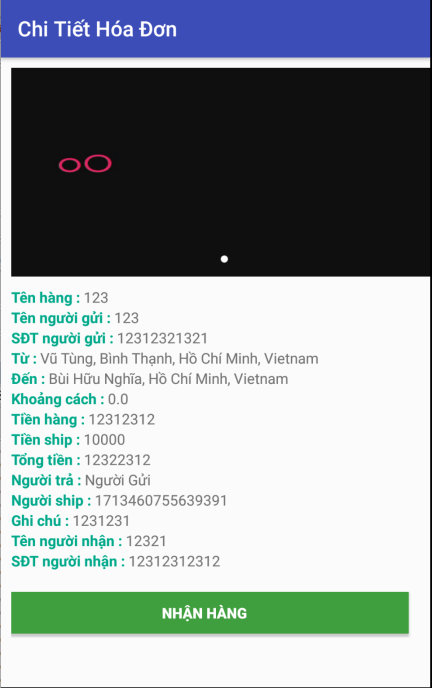
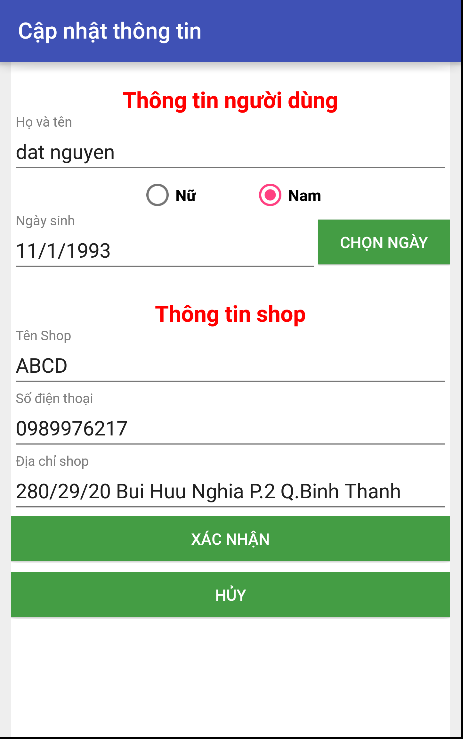
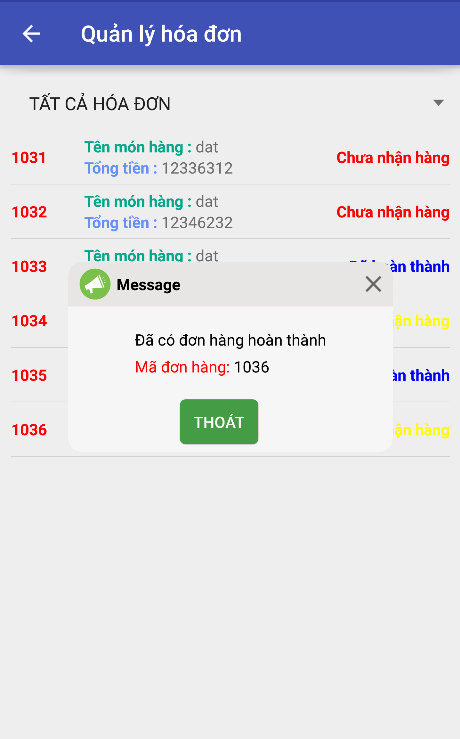
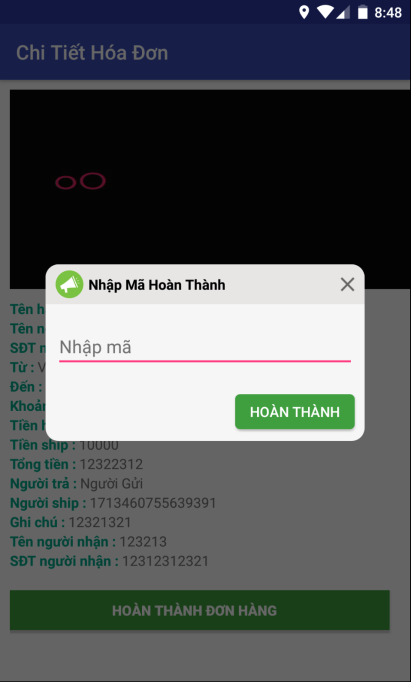
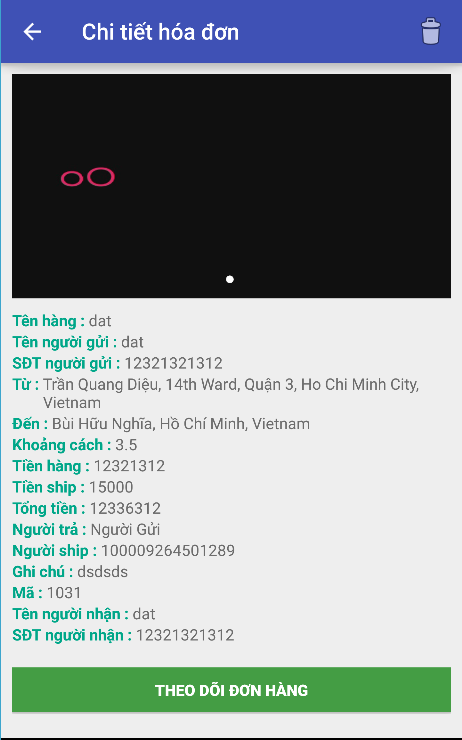
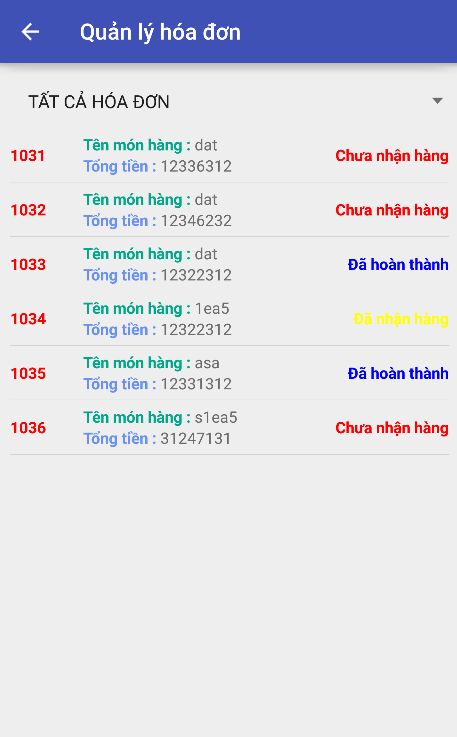
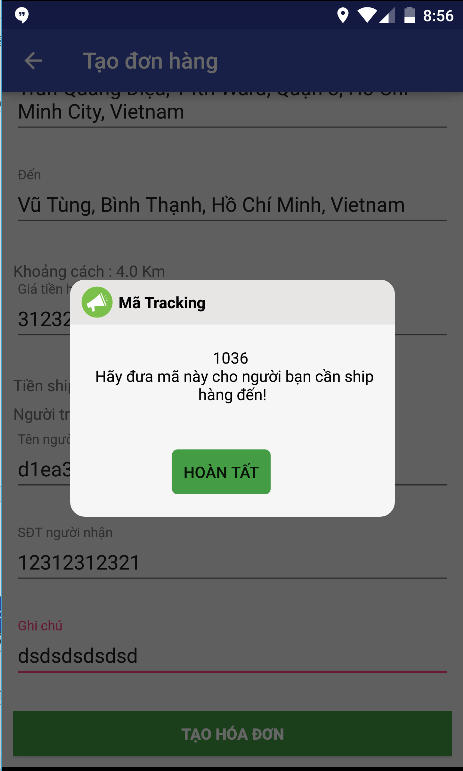
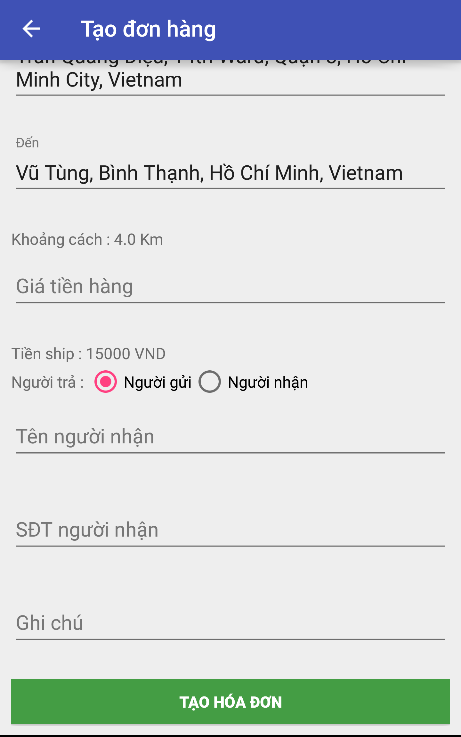
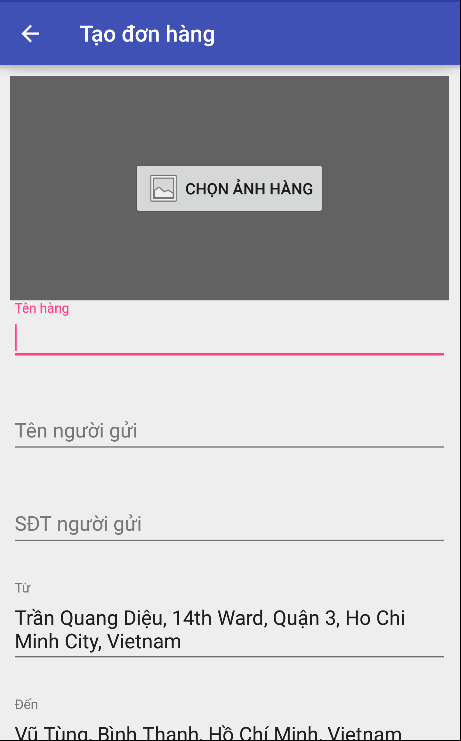
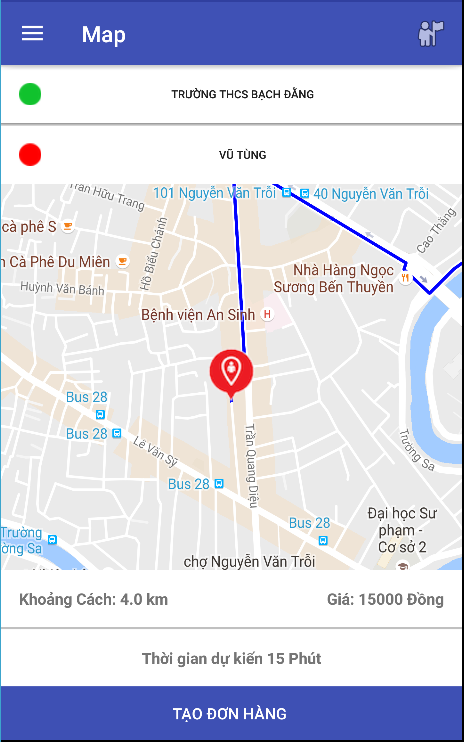
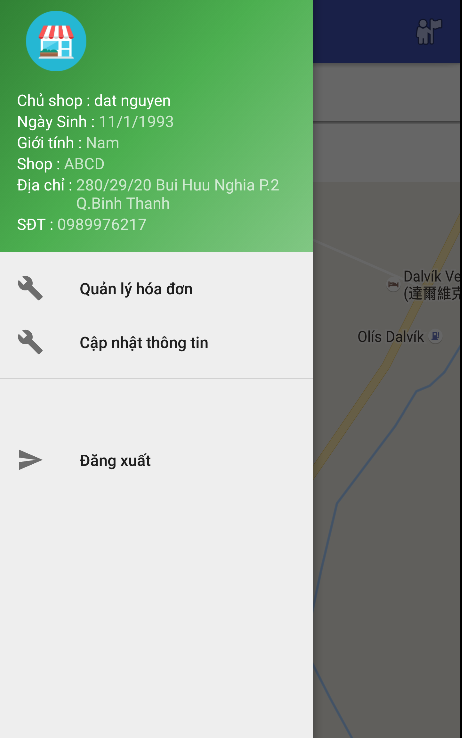
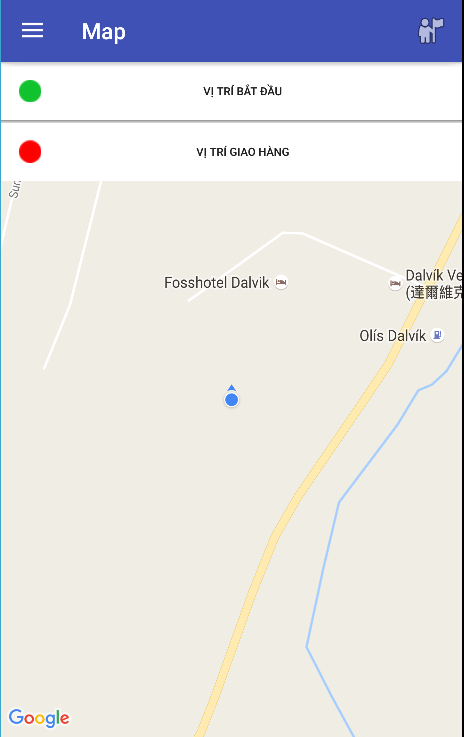
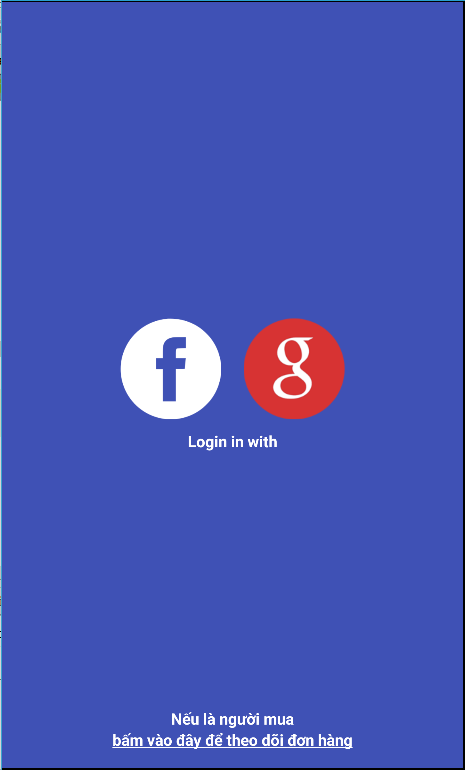
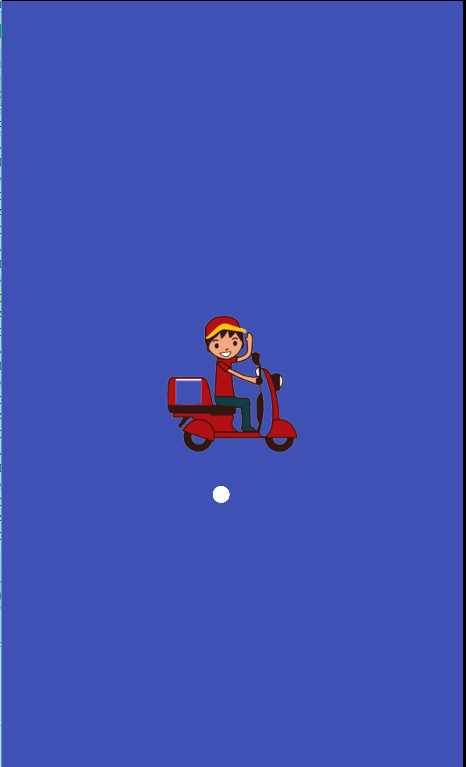
Sửa shipper :



Get ID facebook :



## Thiết kế giao diện:

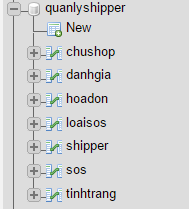


# 7.Xây dựng ứng dụng

## Hệ thống lớp quản lý dữ liệu:

### Lớp Database:

Cấu trúc các bảng như sau :

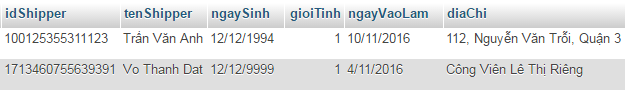


Chi tiết các bảng :

Bảng chủ shop :



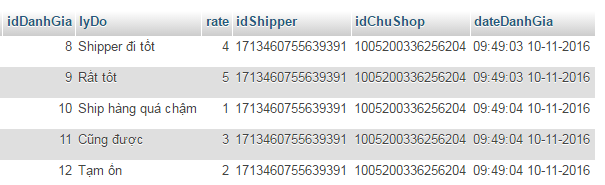
Bảng shipper:



Bảng hóa đơn :



Bảng Đánh giá :



### Lớp quản lý đối tượng :

Tương ứng với 5 table database ta có 5 lớp đối tượng như sau :

DTHoaDon:

**package** com.poly.shipper.model;  
  
/\*\*  
 \* Created by quocdat on 10/30/2016.  
 \*/  
**public class** HoaDon {  
 String hinhAnh;  
 String tenHang, tenNguoiGui, tu, toaDoTu, den,  
 toaDoDen, tenNguoiNhan, ghiChu, idShipper, idchushop, sdtNguoiGui, sdtNguoiNhan;  
 **int** tienHang, tienShip, idTinhTrang, idHoaDon;  
 **double** khoangCach;  
 **boolean** nguoiTra;  
 **public** HoaDon(){  
 **this**.idHoaDon = 0;  
 **this**.hinhAnh = **null**;  
 **this**.tenHang = **null**;  
 **this**.tenNguoiGui = **null**;  
 **this**.sdtNguoiGui = **null**;  
 **this**.tu = **null**;  
 **this**.toaDoTu = **null**;  
 **this**.den = **null**;  
 **this**.toaDoDen = **null**;  
 **this**.khoangCach = 0;  
 **this**.tienHang = 0;  
 **this**.tienShip = 0;  
 **this**.nguoiTra = **true**;  
 **this**.idShipper = **null**;  
 **this**.ghiChu = **null**;  
 **this**.tenNguoiNhan = **null**;  
 **this**.sdtNguoiNhan = **null**;  
 **this**.idchushop = **null**;  
 **this**.idTinhTrang = 0;  
 }  
  
  
 **public** HoaDon(String tenHang, String tenNguoiGui, String sdtNguoiGui, String tu, String toaDoTu, String den, String toaDoDen,  
 **double** khoangCach, **int** tienHang, **int** tienShip, **boolean** nguoiTra, String ghiChu, String tenNguoiNhan, String sdtNguoiNhan, String idchushop){  
 **this**.hinhAnh = **null**;  
 **this**.tenHang = tenHang;  
 **this**.tenNguoiGui = tenNguoiGui;  
 **this**.sdtNguoiGui = sdtNguoiGui;  
 **this**.tu = tu;  
 **this**.toaDoTu = toaDoTu;  
 **this**.den = den;  
 **this**.toaDoDen = toaDoDen;  
 **this**.khoangCach = khoangCach;  
 **this**.tienHang = tienHang;  
 **this**.tienShip = tienShip;  
 **this**.nguoiTra = nguoiTra;  
 **this**.idShipper = **null**;  
 **this**.ghiChu = ghiChu;  
 **this**.idHoaDon = 0;  
 **this**.tenNguoiNhan = tenNguoiNhan;  
 **this**.sdtNguoiNhan = sdtNguoiNhan;  
 **this**.idchushop = idchushop;  
 **this**.idTinhTrang = 0;  
 }  
  
 **public void** setIdHoaDon(**int** idHoaDon) {  
 **this**.idHoaDon = idHoaDon;  
 }  
  
 **public int** getIdHoaDon() {  
  
 **return** idHoaDon;  
 }  
  
 **public void** setIdchushop(String idchushop) {  
 **this**.idchushop = idchushop;  
 }  
  
 **public void** setIdTinhTrang(**int** idTinhTrang) {  
 **this**.idTinhTrang = idTinhTrang;  
 }  
  
 **public int** getIdTinhTrang() {  
 **return** idTinhTrang;  
 }  
  
 **public** String getIdchushop() {  
 **return** idchushop;  
 }  
 **public void** setHinhAnh(String hinhAnh) {  
 **this**.hinhAnh = hinhAnh;  
 }  
  
 **public void** setTenHang(String tenHang) {  
 **this**.tenHang = tenHang;  
 }  
  
 **public void** setTenNguoiGui(String tenNguoiGui) {  
 **this**.tenNguoiGui = tenNguoiGui;  
 }  
  
 **public void** setTu(String tu) {  
 **this**.tu = tu;  
 }  
  
 **public void** setToaDoTu(String toaDoTu) {  
 **this**.toaDoTu = toaDoTu;  
 }  
  
 **public void** setDen(String den) {  
 **this**.den = den;  
 }  
  
 **public void** setToaDoDen(String toaDoDen) {  
 **this**.toaDoDen = toaDoDen;  
 }  
  
 **public void** setTenNguoiNhan(String tenNguoiNhan) {  
 **this**.tenNguoiNhan = tenNguoiNhan;  
 }  
  
 **public void** setGhiChu(String ghiChu) {  
 **this**.ghiChu = ghiChu;  
 }  
  
 **public void** setIdShipper(String nguoiShip) {  
 **this**.idShipper = nguoiShip;  
 }  
  
 **public void** setSdtNguoiGui(String sdtNguoiGui) {  
 **this**.sdtNguoiGui = sdtNguoiGui;  
 }  
  
 **public void** setKhoangCach(**double** khoangCach) {  
 **this**.khoangCach = khoangCach;  
 }  
  
 **public void** setTienHang(**int** tienHang) {  
 **this**.tienHang = tienHang;  
 }  
  
 **public void** setTienShip(**int** tienShip) {  
 **this**.tienShip = tienShip;  
 }  
  
 **public void** setSdtNguoiNhan(String sdtNguoiNhan) {  
 **this**.sdtNguoiNhan = sdtNguoiNhan;  
 }  
  
 **public void** setNguoiTra(**boolean** nguoiTra) {  
 **this**.nguoiTra = nguoiTra;  
 }  
  
 **public** String getTenHang() {  
  
 **return** tenHang;  
 }  
  
 **public** String getHinhAnh() {  
 **return** hinhAnh;  
 }  
  
 **public** String getTenNguoiGui() {  
 **return** tenNguoiGui;  
 }  
  
 **public** String getTu() {  
 **return** tu;  
 }  
  
 **public** String getToaDoTu() {  
 **return** toaDoTu;  
 }  
  
 **public** String getDen() {  
 **return** den;  
 }  
  
 **public** String getToaDoDen() {  
 **return** toaDoDen;  
 }  
  
 **public** String getTenNguoiNhan() {  
 **return** tenNguoiNhan;  
 }  
  
 **public** String getGhiChu() {  
 **return** ghiChu;  
 }  
  
 **public** String getIdShipper() {  
 **return** idShipper;  
 }  
  
 **public double** getKhoangCach() {  
 **return** khoangCach;  
 }  
  
 **public** String getSdtNguoiGui() {  
 **return** sdtNguoiGui;  
 }  
  
 **public int** getTienHang() {  
 **return** tienHang;  
 }  
  
 **public int** getTienShip() {  
 **return** tienShip;  
 }  
  
 **public** String getSdtNguoiNhan() {  
 **return** sdtNguoiNhan;  
 }  
  
 **public boolean** isNguoiTra() {  
 **return** nguoiTra;  
 }  
}

DTChuShop:

**package** com.poly.shipper.model;  
  
/\*\*  
 \* Created by quocdat on 11/7/2016.  
 \*/  
  
**public class** ChuShop {  
 String idChuShop, tenChuShop, ngaySinh, tenShop, diaChi, soDienThoai;  
 **boolean** gioiTinh;  
 **public** ChuShop(){  
 **this**.idChuShop = **null**;  
 **this**.tenChuShop = **null**;  
 **this**.gioiTinh = **false**;  
 **this**.ngaySinh = **null**;  
 **this**.tenShop = **null**;  
 **this**.diaChi = **null**;  
 **this**.soDienThoai = **null**;  
 }  
 **public** ChuShop(String idChuShop, String tenChuShop, **boolean** gioiTinh, String ngaySinh, String tenShop, String diaChi, String soDienThoai){  
 **this**.idChuShop = idChuShop;  
 **this**.tenChuShop = tenChuShop;  
 **this**.gioiTinh = gioiTinh;  
 **this**.ngaySinh = ngaySinh;  
 **this**.tenShop = tenShop;  
 **this**.diaChi = diaChi;  
 **this**.soDienThoai = soDienThoai;  
 }  
  
 **public void** setDiaChi(String diaChi) {  
 **this**.diaChi = diaChi;  
 }  
  
 **public void** setGioiTinh(**boolean** gioiTinh) {  
 **this**.gioiTinh = gioiTinh;  
 }  
  
 **public void** setIdChuShop(String idChuShop) {  
 **this**.idChuShop = idChuShop;  
 }  
  
 **public void** setNgaySinh(String ngaySinh) {  
 **this**.ngaySinh = ngaySinh;  
 }  
  
 **public void** setSoDienThoai(String soDienThoai) {  
 **this**.soDienThoai = soDienThoai;  
 }  
  
 **public void** setTenChuShop(String tenChuShop) {  
 **this**.tenChuShop = tenChuShop;  
 }  
  
 **public void** setTenShop(String tenShop) {  
 **this**.tenShop = tenShop;  
 }  
  
 **public** String getDiaChi() {  
 **return** diaChi;  
 }  
  
 **public boolean** isGioiTinh() {  
 **return** gioiTinh;  
 }  
  
 **public** String getIdChuShop() {  
 **return** idChuShop;  
 }  
  
 **public** String getNgaySinh() {  
 **return** ngaySinh;  
 }  
  
 **public** String getSoDienThoai() {  
 **return** soDienThoai;  
 }  
  
 **public** String getTenChuShop() {  
 **return** tenChuShop;  
 }  
  
 **public** String getTenShop() {  
 **return** tenShop;  
 }  
}

## Hệ thống Activity

### PlashScreenActivity :

**package** com.poly.shipper.activity;  
  
**import** android.app.Activity;  
**import** android.app.AlarmManager;  
**import** android.app.PendingIntent;  
**import** android.content.Context;  
**import** android.content.pm.PackageInfo;  
**import** android.content.pm.PackageManager;  
**import** android.os.SystemClock;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.os.Handler;  
**import** android.content.Intent;  
**import** android.util.Base64;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.view.Window;  
**import** android.view.WindowManager;  
**import** android.view.animation.Animation;  
**import** android.view.animation.AnimationUtils;  
**import** android.widget.LinearLayout;  
  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.broadcast.KeepConnect;  
**import** com.poly.shipper.service.MyService;  
**import** com.poly.shipper.utils.utils;  
  
**import** java.security.MessageDigest;  
**import** java.security.NoSuchAlgorithmException;  
  
**public class** PlashScreenActivity **extends** AppCompatActivity {  
 **private static int** *splashInterval* = 3000;  
 LinearLayout l2;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*,  
 WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);  
 getSupportActionBar().hide();  
  
 setContentView(R.layout.*activity\_plash\_screen*);  
  
 utils.*context* = **this**;  
 showHashKey(**this**);  
 Intent launchIntent = **new** Intent(**this**, KeepConnect.**class**);  
 PendingIntent mAlarmIntent = PendingIntent.*getBroadcast*(**this**, 0, launchIntent, 0);  
 AlarmManager manager = (AlarmManager) getSystemService(Context.*ALARM\_SERVICE*);  
 **long** interval = 30 \* 1000; // 5 giây.  
 manager.setRepeating(AlarmManager.*ELAPSED\_REALTIME*,  
 SystemClock.*elapsedRealtime*() + interval, interval,  
 mAlarmIntent);  
 System.*out*.println("aaaaa");  
 System.*out*.println("zo2");  
  
 LinearLayout l1 = (LinearLayout)findViewById(R.id.*linearLayout1*);  
 setTranslate(l1);  
 l2 = (LinearLayout)findViewById(R.id.*linearLayout2*);  
 l2.setVisibility(View.*GONE*);  
  
  
  
  
 }  
  
 **public void** setTranslate(View view)  
 {  
 Animation animFadein = AnimationUtils.*loadAnimation*(**this**,  
 R.anim.*translate*);  
 animFadein.setAnimationListener(**new** Animation.AnimationListener() {  
  
 @Override  
 **public void** onAnimationStart(Animation animation) {  
 // **TODO Auto-generated method stub** }  
  
 @Override  
 **public void** onAnimationRepeat(Animation animation) {  
 // **TODO Auto-generated method stub** }  
  
 @Override  
 **public void** onAnimationEnd(Animation animation) {  
 l2.setVisibility(View.*VISIBLE*);  
 View circle1 = (View)findViewById(R.id.*circle1*);  
 View circle2 = (View)findViewById(R.id.*circle2*);  
 setAnimationScale(circle1,R.anim.*scale*);  
 setAnimationScale(circle2,R.anim.*scale2*);  
  
 **if** (utils.*alreadyLogin* == **true**){  
 startActivity(**new** Intent(PlashScreenActivity.**this**, NavDrawerActivity.**class**));  
 **return**;  
 }  
 **if** (MyService.*running* == **false**) {  
 startService(**new** Intent(PlashScreenActivity.**this**, MyService.**class**));  
 }  
 }  
 });  
  
  
 view.setAnimation(animFadein);  
  
 }  
  
 **void** setAnimationScale(View v,**int** id\_animation)  
 {  
 Animation animFadein = AnimationUtils.*loadAnimation*(**this**,  
 id\_animation);  
 v.startAnimation(animFadein);  
 }  
  
 **public void** showHashKey(Context context) {  
 **try** {  
 PackageInfo info = context.getPackageManager().getPackageInfo("com.poly.shipper",  
 PackageManager.*GET\_SIGNATURES*);  
 **for** (android.content.pm.Signature signature : info.signatures) {  
 MessageDigest md = MessageDigest.*getInstance*("SHA");  
 md.update(signature.toByteArray());  
  
 String sign = Base64.*encodeToString*(md.digest(), Base64.*DEFAULT*);  
 Log.*e*("KeyHash:", sign);  
 // Toast.makeText(getApplicationContext(),sign, Toast.LENGTH\_LONG).show();  
 }  
 Log.*d*("KeyHash:", "\*\*\*\*------------\*\*\*");  
 } **catch** (PackageManager.NameNotFoundException e) {  
 e.printStackTrace();  
 } **catch** (NoSuchAlgorithmException e) {  
 e.printStackTrace();  
 }  
 }  
}

### MainActivity:

**package** com.poly.shipper.activity;  
  
**import** android.app.AlarmManager;  
**import** android.app.PendingIntent;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.pm.PackageInfo;  
**import** android.content.pm.PackageManager;  
**import** android.content.pm.Signature;  
**import** android.os.SystemClock;  
**import** android.support.annotation.NonNull;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.util.Base64;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.view.WindowManager;  
**import** android.widget.Button;  
**import** android.widget.ImageView;  
**import** android.widget.TextView;  
  
**import** com.google.android.gms.auth.api.Auth;  
**import** com.google.android.gms.auth.api.signin.GoogleSignInAccount;  
**import** com.google.android.gms.auth.api.signin.GoogleSignInOptions;  
**import** com.google.android.gms.auth.api.signin.GoogleSignInResult;  
**import** com.google.android.gms.common.ConnectionResult;  
**import** com.google.android.gms.common.api.GoogleApiClient;  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.broadcast.KeepConnect;  
**import** com.poly.shipper.main.BuildMessage;  
**import** com.poly.shipper.service.MyService;  
  
**import** org.json.JSONObject;  
  
**import** com.facebook.CallbackManager;  
**import** com.facebook.FacebookCallback;  
**import** com.facebook.FacebookException;  
**import** com.facebook.FacebookSdk;  
**import** com.facebook.GraphRequest;  
**import** com.facebook.GraphResponse;  
**import** com.facebook.login.LoginResult;  
**import** com.facebook.login.widget.LoginButton;  
**import** com.poly.shipper.utils.utils;  
  
**import** java.security.MessageDigest;  
**import** java.security.NoSuchAlgorithmException;  
**import** java.util.Arrays;  
  
**public class** MainActivity **extends** AppCompatActivity {  
 **private** CallbackManager callbackManager;  
 **private** LoginButton loginButton;  
 GoogleApiClient mGoogleApiClient;  
 **private** TextView tracking;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*,  
 WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);  
 getSupportActionBar().hide();  
 FacebookSdk.*sdkInitialize*(getApplicationContext());  
 callbackManager = CallbackManager.Factory.*create*();  
 setContentView(R.layout.*activity\_main*);  
 tracking = (TextView)findViewById(R.id.*tracking*);  
 tracking.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(MainActivity.**this**,TrackingActivity.**class**);  
 startActivity(i);  
 }  
 });  
 utils.*context* = MainActivity.**this**;  
  
 loginButton = (LoginButton)findViewById(R.id.*loginBtnFacebook*);  
 loginButton.setReadPermissions(Arrays.*asList*("public\_profile", "email"));  
 loginButton.registerCallback(callbackManager, **new** FacebookCallback<LoginResult>() {  
 @Override  
 **public void** onSuccess(LoginResult loginResult) {  
 String facebookId = loginResult.getAccessToken().getUserId();  
 **try** {  
 BuildMessage xMessage = **new** BuildMessage();  
 JSONObject infoMessage = **new** JSONObject();  
 infoMessage.put("idTaiKhoan", facebookId);  
 infoMessage.put("type", "ChuShop");  
 String message = xMessage.buildMessage("DoLogin", infoMessage);  
 MyService.*client*.send(message);  
 }  
 **catch** (Exception ex){  
 ex.printStackTrace();  
 }  
 }  
  
 @Override  
 **public void** onCancel() {  
  
 }  
  
 @Override  
 **public void** onError(FacebookException e) {  
  
 }  
 });  
  
 ImageView btnSignInFacebook = (ImageView)findViewById(R.id.*btnSignInFacebook*);  
 btnSignInFacebook.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 loginButton.performClick();  
 }  
 });  
 GoogleSignInOptions gso = **new** GoogleSignInOptions.Builder(GoogleSignInOptions.*DEFAULT\_SIGN\_IN*)  
 .requestEmail()  
 .build();  
 mGoogleApiClient = **new** GoogleApiClient.Builder(**this**)  
 .enableAutoManage(**this** /\* FragmentActivity \*/, **new** GoogleApiClient.OnConnectionFailedListener() {  
 @Override  
 **public void** onConnectionFailed(@NonNull ConnectionResult connectionResult) {  
  
 }  
 } /\* OnConnectionFailedListener \*/)  
 .addApi(Auth.*GOOGLE\_SIGN\_IN\_API*, gso)  
 .build();  
 ImageView signUpButton = (ImageView) findViewById(R.id.*btnSignInGoogle*);  
 signUpButton.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 signInGoogle();  
 }  
 });  
  
 }  
  
 **private void** signInGoogle() {  
 Intent signInIntent = Auth.*GoogleSignInApi*.getSignInIntent(mGoogleApiClient);  
 startActivityForResult(signInIntent, 0);  
 }  
  
 **private void** handleSignInResult(GoogleSignInResult result) {  
  
  
 **if** (result.isSuccess()) {  
  
 **try** {  
 GoogleSignInAccount acct = result.getSignInAccount();  
 String id = acct.getId();  
 **try** {  
 BuildMessage xMessage = **new** BuildMessage();  
 JSONObject infoMessage = **new** JSONObject();  
 infoMessage.put("idTaiKhoan", id);  
 infoMessage.put("type", "ChuShop");  
 String message = xMessage.buildMessage("DoLogin", infoMessage);  
 MyService.*client*.send(message);  
 }  
 **catch** (Exception ex){  
 ex.printStackTrace();  
 }  
 }**catch**(Exception e){  
 e.printStackTrace();  
 }  
 }  
 }  
  
  
 @Override  
 **public void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
 **super**.onActivityResult(requestCode, resultCode, data);  
 callbackManager.onActivityResult(requestCode, resultCode, data);  
 **if** (requestCode == 0) {  
 GoogleSignInResult result = Auth.*GoogleSignInApi*.getSignInResultFromIntent(data);  
 handleSignInResult(result);  
 }  
 }  
}

### ChiTietHoaDonActivity:

1. **package** com.poly.shipper.activity;  
     
   **import** android.app.Dialog;  
   **import** android.content.Context;  
   **import** android.content.Intent;  
   **import** android.graphics.Bitmap;  
   **import** android.graphics.BitmapFactory;  
   **import** android.graphics.Color;  
   **import** android.graphics.drawable.ColorDrawable;  
   **import** android.net.Uri;  
   **import** android.os.AsyncTask;  
   **import** android.os.Bundle;  
   **import** android.os.Environment;  
   **import** android.support.v4.view.ViewPager;  
   **import** android.support.v7.app.AppCompatActivity;  
   **import** android.view.Menu;  
   **import** android.view.MenuInflater;  
   **import** android.view.MenuItem;  
   **import** android.view.View;  
   **import** android.view.Window;  
   **import** android.widget.Button;  
   **import** android.widget.EditText;  
   **import** android.widget.RatingBar;  
   **import** android.widget.TextView;  
     
   **import** com.google.android.gms.appindexing.Action;  
   **import** com.google.android.gms.appindexing.AppIndex;  
   **import** com.google.android.gms.appindexing.Thing;  
   **import** com.google.android.gms.common.api.GoogleApiClient;  
   **import** com.poly.shipper.R;  
   **import** com.poly.shipper.adapter.CustomPagerAdapter;  
   **import** com.poly.shipper.main.BuildMessage;  
   **import** com.poly.shipper.model.CirclePageIndicator;  
   **import** com.poly.shipper.model.HoaDon;  
   **import** com.poly.shipper.model.PageIndicator;  
   **import** com.poly.shipper.service.MyService;  
     
   **import** org.json.JSONObject;  
     
   **import** java.io.BufferedInputStream;  
   **import** java.io.ByteArrayOutputStream;  
   **import** java.io.File;  
   **import** java.io.FileOutputStream;  
   **import** java.io.IOException;  
   **import** java.io.InputStream;  
   **import** java.io.OutputStream;  
   **import** java.net.HttpURLConnection;  
   **import** java.net.MalformedURLException;  
   **import** java.net.URL;  
   **import** java.net.URLConnection;  
   **import** java.util.ArrayList;  
     
   **import static** com.poly.shipper.activity.QuanLyHoaDonChuShopActivity.*customListViewAdapter*;  
     
   **public class** ChiTietHoaDonActivity **extends** AppCompatActivity {  
    **private int** rate;  
    **private** ArrayList<String> image\_intro = **new** ArrayList<String>();  
    **private** CustomPagerAdapter mAdapter;  
    **private** ViewPager mPager;  
    **private** PageIndicator mIndicator;  
     
    **private** TextView ctHoaDonTvTenHang, ctHoaDonTvTenNguoiGui, ctHoaDonTvSdtNguoiGui, ctHoaDonTvTu, ctHoaDonTvDen, ctHoaDonTvKhoangCach,  
    ctHoaDonTvTienHang, ctHoaDonTvTienShip, ctHoaDonTvTong, ctHoaDonTvNguoiTra, ctHoaDonTvNguoiShip, ctHoaDonTvGhiChu, ctHoaDonTvMa,  
    ctHoaDonTvNguoiNhan, ctHoaDonTvSdtNguoiNhan;  
    **private** Button ctHoaDonBtnTheoDoi;  
    **static public** Button *ctHoaDonBtnDanhGia*;  
    **private** HoaDon hoaDon;  
    **private** String[] a;  
    **private** Dialog dialog;  
    **private int** index;  
    @Override  
    **protected void** onCreate(Bundle savedInstanceState) {  
    **super**.onCreate(savedInstanceState);  
    setContentView(R.layout.*activity\_chi\_tiet\_hoa\_don*);  
    getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
    khaiBao();  
    invalidateOptionsMenu();  
    viewPager();  
    }  
     
    **public void** khaiBao() {  
    //lấy dữ liệu đầu vào  
     
    Intent i = getIntent();  
    index = i.getIntExtra("index", 0);  
    hoaDon = *customListViewAdapter*.getHoaDonFromAdap(index);  
     
    a = hoaDon.getHinhAnh().split(";");  
    **for** (**int** j = 0; j < a.length; j++) {  
    **new** DownloadImageTask().execute("http://163.44.150.34/"+a[j]);  
    }  
     
     
    // các trường  
    mPager = (ViewPager) findViewById(R.id.*ctHoaDonVpLoadAnh*);  
    mIndicator = (PageIndicator) findViewById(R.id.*ctHoaDonIndicator*);  
    ctHoaDonTvTenHang = (TextView) findViewById(R.id.*ctHoaDonTvTenHang*);  
    ctHoaDonTvTenHang.setText(hoaDon.getTenHang());  
    ctHoaDonTvTenNguoiGui = (TextView) findViewById(R.id.*ctHoaDonTvTenNguoiGui*);  
    ctHoaDonTvTenNguoiGui.setText(hoaDon.getTenNguoiGui());  
    ctHoaDonTvSdtNguoiGui = (TextView) findViewById(R.id.*ctHoaDonTvSdtNguoiGui*);  
    ctHoaDonTvSdtNguoiGui.setText(hoaDon.getSdtNguoiGui());  
    ctHoaDonTvTu = (TextView) findViewById(R.id.*ctHoaDonTvTu*);  
    ctHoaDonTvTu.setText(hoaDon.getTu());  
    ctHoaDonTvDen = (TextView) findViewById(R.id.*ctHoaDonTvDen*);  
    ctHoaDonTvDen.setText(hoaDon.getDen());  
    ctHoaDonTvKhoangCach = (TextView) findViewById(R.id.*ctHoaDonTvKhoangCach*);  
    ctHoaDonTvKhoangCach.setText(String.*valueOf*(hoaDon.getKhoangCach()));  
    ctHoaDonTvTienHang = (TextView) findViewById(R.id.*ctHoaDonTvTienHang*);  
    ctHoaDonTvTienHang.setText(String.*valueOf*(hoaDon.getTienHang()));  
    ctHoaDonTvTienShip = (TextView) findViewById(R.id.*ctHoaDonTvTienShip*);  
    ctHoaDonTvTienShip.setText(String.*valueOf*(hoaDon.getTienShip()));  
    ctHoaDonTvTong = (TextView) findViewById(R.id.*ctHoaDonTvTong*);  
    ctHoaDonTvTong.setText(String.*valueOf*(hoaDon.getTienHang() + hoaDon.getTienShip()));  
    ctHoaDonTvNguoiTra = (TextView) findViewById(R.id.*ctHoaDonTvNguoiTra*);  
    **if** (hoaDon.isNguoiTra()) {  
    ctHoaDonTvNguoiTra.setText("Người Gửi");  
    } **else** {  
    ctHoaDonTvNguoiTra.setText("Người nhận");  
    }  
    ctHoaDonTvNguoiShip = (TextView) findViewById(R.id.*ctHoaDonTvNguoiShip*);  
    ctHoaDonTvNguoiShip.setText(hoaDon.getIdShipper());  
    ctHoaDonTvGhiChu = (TextView) findViewById(R.id.*ctHoaDonTvGhiChu*);  
    ctHoaDonTvGhiChu.setText(hoaDon.getGhiChu());  
    ctHoaDonTvMa = (TextView) findViewById(R.id.*ctHoaDonTvMa*);  
    ctHoaDonTvMa.setText(String.*valueOf*(hoaDon.getIdHoaDon()));  
    ctHoaDonTvNguoiNhan = (TextView) findViewById(R.id.*ctHoaDonTvNguoiNhan*);  
    ctHoaDonTvNguoiNhan.setText(hoaDon.getTenNguoiNhan());  
    ctHoaDonTvSdtNguoiNhan = (TextView) findViewById(R.id.*ctHoaDonTvSdtNguoiNhan*);  
    ctHoaDonTvSdtNguoiNhan.setText(hoaDon.getSdtNguoiNhan());  
     
    ctHoaDonBtnTheoDoi = (Button) findViewById(R.id.*ctHoaDonBtnTheoDoi*);  
    ctHoaDonBtnTheoDoi.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View v) {  
    Intent i = **new** Intent(ChiTietHoaDonActivity.**this**,TrackingActivity.**class**);  
    i.putExtra("idHoaDon",String.*valueOf*(hoaDon.getIdHoaDon() ));  
    startActivity(i);  
    }  
    });  
    *ctHoaDonBtnDanhGia* = (Button) findViewById(R.id.*ctHoaDonBtnDanhGia*);  
    **if**(hoaDon.getIdTinhTrang()==2){  
    *ctHoaDonBtnDanhGia*.setVisibility(View.*VISIBLE*);  
    ctHoaDonBtnTheoDoi.setVisibility(View.*GONE*);  
    }  
    *ctHoaDonBtnDanhGia*.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View v) {  
    **final** Dialog dialog = **new** Dialog(ChiTietHoaDonActivity.**this**);  
    dialog.getWindow().setBackgroundDrawable(**new** ColorDrawable(Color.*TRANSPARENT*));  
    dialog.requestWindowFeature(Window.*FEATURE\_NO\_TITLE*);  
    dialog.setContentView(R.layout.*popupdanhgiashipper*);  
    **final** TextView FbRbNguoiShip = (TextView)dialog.findViewById(R.id.*FbRbNguoiShip*) ;  
    **final** RatingBar FbRbDanhGia = (RatingBar)dialog.findViewById(R.id.*FbRbDanhGia*);  
    **final** EditText FbRbLyDo = (EditText)dialog.findViewById(R.id.*FbRbLyDo*);  
    **final** Button FbRbGui = (Button)dialog.findViewById(R.id.*FbRbGui*);  
    FbRbDanhGia.setOnRatingBarChangeListener(**new** RatingBar.OnRatingBarChangeListener() {  
    @Override  
    **public void** onRatingChanged(RatingBar ratingBar, **float** rating, **boolean** fromUser) {  
    String a = rating + "";  
    rate =Integer.*parseInt*(a.charAt(0)+"") ;  
    }  
    });  
     
    FbRbGui.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View v) {  
    **try** {  
    BuildMessage xMessage = **new** BuildMessage();  
    JSONObject infoMessage = **new** JSONObject();  
    infoMessage.put("lyDo", FbRbLyDo.getText().toString());  
    infoMessage.put("rate", rate);  
     
    infoMessage.put("idChuShop", hoaDon.getIdchushop());  
    infoMessage.put("idShipper", hoaDon.getIdShipper());  
    String message = xMessage.buildMessage("DanhGiaShipper", infoMessage);  
    System.*out*.println(message);  
    MyService.*client*.send(message);  
     
    }  
    **catch** (Exception ex){  
    ex.printStackTrace();  
     
    }  
    dialog.dismiss();  
     
    }  
    });  
     
    dialog.show();  
    }  
    });  
     
    }  
    **public static void** danhGiaHoanTat(){  
    *ctHoaDonBtnDanhGia*.setVisibility(View.*GONE*);  
    }  
    **public void** viewPager(){  
    image\_intro = **new** ArrayList<String>();  
     
    mAdapter = **new** CustomPagerAdapter(**this**,image\_intro);  
     
    mPager = (ViewPager)findViewById(R.id.*ctHoaDonVpLoadAnh*);  
     
    mIndicator = (CirclePageIndicator)findViewById(R.id.*ctHoaDonIndicator*);  
     
     
    mIndicator.setOnPageChangeListener(**new** ViewPager.OnPageChangeListener() {  
    @Override  
    **public void** onPageSelected(**int** position) {  
     
    }  
     
    @Override  
    **public void** onPageScrolled(**int** position, **float** positionOffset, **int** positionOffsetPixels) {  
    }  
     
    @Override  
    **public void** onPageScrollStateChanged(**int** state) {  
    }  
    });  
    }  
     
     
    @Override  
    **public boolean** onCreateOptionsMenu(Menu menu) {  
    MenuInflater inflater = getMenuInflater();  
    inflater.inflate(R.menu.*menu1*, menu);  
    **if**(hoaDon.getIdTinhTrang()!=0){  
    menu.getItem(0).setVisible(**false**);  
    }  
    **return true**;  
    }  
     
    @Override  
    **public boolean** onOptionsItemSelected(MenuItem item) {  
    **switch** (item.getItemId()) {  
    **case** R.id.*menu\_xoa*:  
    dialog = **new** Dialog(ChiTietHoaDonActivity.**this**);  
    dialog.getWindow().setBackgroundDrawable(**new** ColorDrawable(android.graphics.Color.*TRANSPARENT*));  
    dialog.requestWindowFeature(Window.*FEATURE\_NO\_TITLE*);  
    dialog.setContentView(R.layout.*popupmessagexoahoadon*);  
    dialog.show();  
    **final** Button ok = (Button)dialog.findViewById(R.id.*ok*);  
    ok.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View v) {  
    dialog.dismiss();  
    yeuCauXoaHoaDon(hoaDon.getIdHoaDon());  
    Intent i = **new** Intent(ChiTietHoaDonActivity.**this**,QuanLyHoaDonChuShopActivity.**class**);  
    startActivity(i);  
    }  
    });  
    **final** Button cancel = (Button)dialog.findViewById(R.id.*cancel*);  
    cancel.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View v) {  
    dialog.dismiss();  
    }  
    });  
    **return true**;  
    **case** android.R.id.*home*:  
    finish();  
    **return true**;  
    **default**:  
    // If we got here, the user's action was not recognized.  
    // Invoke the superclass to handle it.  
    **return super**.onOptionsItemSelected(item);  
     
    }  
    }  
    **private void** yeuCauXoaHoaDon(**int** idHoaDon){  
    **try** {  
    BuildMessage xMessage = **new** BuildMessage();  
    JSONObject infoMessage = **new** JSONObject();  
    infoMessage.put("idHoaDon", idHoaDon);  
    String message = xMessage.buildMessage("XoaHoaDon", infoMessage);  
    System.*out*.println(message);  
    MyService.*client*.send(message);  
    }  
    **catch** (Exception ex){  
    ex.printStackTrace();  
    }  
    }  
    **private class** DownloadImageTask **extends** AsyncTask<String, Void, String> {  
     
    **public** DownloadImageTask() {  
    System.*out*.print("ok");  
    }  
     
    **public** String doInBackground(String... urls) {  
    String urldisplay = urls[0];  
    String filePath = "";  
    System.*out*.print("download");  
    **try** {  
    System.*out*.print("download ok");  
    URL url = **new** URL(urldisplay);  
    InputStream in = **new** BufferedInputStream(url.openStream());  
    ByteArrayOutputStream out = **new** ByteArrayOutputStream();  
    **byte**[] buf = **new byte**[1024];  
    **int** n = 0;  
    **while** (-1 != (n = in.read(buf))) {  
    out.write(buf, 0, n);  
    }  
    out.close();  
    in.close();  
    **byte**[] response = out.toByteArray();  
    String fileName = **new** File(urldisplay).getName();  
    File file = **new** File(Environment.*getExternalStorageDirectory*().getPath() + "/" + fileName);  
    **if** (file.exists() ) file.delete();  
    FileOutputStream fos = **new** FileOutputStream(file);  
    fos.write(response);  
    fos.close();  
    filePath = file.getAbsolutePath();  
    } **catch** (Exception e) {  
    e.printStackTrace();  
    }  
    System.*out*.print("download ok1");  
    **return** filePath;  
    }  
     
    **protected void** onPostExecute(String result) {  
    image\_intro.add(result);  
    mPager.setAdapter(mAdapter);  
    mIndicator.setViewPager(mPager);  
     
    }  
    }  
   }

### NavDrawerActivity

1. **package** com.poly.shipper.activity;  
     
   **import** android.annotation.SuppressLint;  
   **import** android.app.ProgressDialog;  
   **import** android.content.Context;  
   **import** android.content.DialogInterface;  
   **import** android.content.Intent;  
   **import** android.content.pm.PackageManager;  
   **import** android.graphics.Color;  
   **import** android.location.Criteria;  
   **import** android.location.Geocoder;  
   **import** android.location.Location;  
   **import** android.location.LocationManager;  
   **import** android.net.Uri;  
   **import** android.os.Bundle;  
   **import** android.support.design.widget.FloatingActionButton;  
   **import** android.support.design.widget.Snackbar;  
   **import** android.support.v4.app.ActivityCompat;  
   **import** android.support.v7.app.AlertDialog;  
   **import** android.util.Log;  
   **import** android.view.LayoutInflater;  
   **import** android.view.View;  
   **import** android.support.design.widget.NavigationView;  
   **import** android.support.v4.view.GravityCompat;  
   **import** android.support.v4.widget.DrawerLayout;  
   **import** android.support.v7.app.ActionBarDrawerToggle;  
   **import** android.support.v7.app.AppCompatActivity;  
   **import** android.support.v7.widget.Toolbar;  
   **import** android.view.Menu;  
   **import** android.view.MenuItem;  
   **import** android.widget.Button;  
   **import** android.widget.LinearLayout;  
   **import** android.widget.TextView;  
   **import** android.widget.Toast;  
     
   **import** com.facebook.FacebookSdk;  
   **import** com.facebook.login.LoginManager;  
   **import** com.google.android.gms.appindexing.Action;  
   **import** com.google.android.gms.appindexing.AppIndex;  
   **import** com.google.android.gms.appindexing.Thing;  
   **import** com.google.android.gms.common.GooglePlayServicesNotAvailableException;  
   **import** com.google.android.gms.common.GooglePlayServicesRepairableException;  
   **import** com.google.android.gms.common.api.GoogleApiClient;  
   **import** com.google.android.gms.common.api.Status;  
   **import** com.google.android.gms.location.LocationListener;  
   **import** com.google.android.gms.location.places.Place;  
   **import** com.google.android.gms.location.places.ui.PlaceAutocomplete;  
   **import** com.google.android.gms.location.places.ui.PlaceSelectionListener;  
   **import** com.google.android.gms.location.places.ui.SupportPlaceAutocompleteFragment;  
   **import** com.google.android.gms.maps.CameraUpdateFactory;  
   **import** com.google.android.gms.maps.GoogleMap;  
   **import** com.google.android.gms.maps.OnMapReadyCallback;  
   **import** com.google.android.gms.maps.SupportMapFragment;  
   **import** com.google.android.gms.maps.model.BitmapDescriptorFactory;  
   **import** com.google.android.gms.maps.model.LatLng;  
   **import** com.google.android.gms.maps.model.Marker;  
   **import** com.google.android.gms.maps.model.MarkerOptions;  
   **import** com.google.android.gms.maps.model.Polyline;  
   **import** com.google.android.gms.maps.model.PolylineOptions;  
   **import** com.poly.shipper.Manifest;  
   **import** com.poly.shipper.R;  
   **import** com.poly.shipper.main.BuildMessage;  
   **import** com.poly.shipper.model.ChuShop;  
   **import** com.poly.shipper.model.TinyDB;  
   **import** com.poly.shipper.modules.DirectionFinder;  
   **import** com.poly.shipper.modules.DirectionFinderListener;  
   **import** com.poly.shipper.modules.route;  
   **import** com.poly.shipper.service.MyService;  
   **import** com.poly.shipper.utils.utils;  
     
   **import** org.json.JSONObject;  
     
   **import** java.io.UnsupportedEncodingException;  
   **import** java.util.ArrayList;  
   **import** java.util.List;  
   **import** java.util.Locale;  
     
   **public class** NavDrawerActivity **extends** AppCompatActivity  
    **implements** NavigationView.OnNavigationItemSelectedListener, OnMapReadyCallback, DirectionFinderListener,android.location.LocationListener {  
    **private static final int** *TAG* = 1;  
    **private** GoogleMap mMap;  
    **public static** String *diemtu* = "";  
    **public static** String *diemden* = "";  
    **public** LatLng locationTu;  
    **public** LatLng locationDen;  
    **public** LinearLayout mainLayout;  
     
    **public** TextView txtKhoangCach;  
    **public** TextView txtGia;  
    **public** TextView txtThoiGian;  
    List<LatLng> routePoints = **null**;  
    **private** List<Marker> originMarkers = **new** ArrayList<>();  
    **private** List<Marker> destinationMarkers = **new** ArrayList<>();  
    SupportPlaceAutocompleteFragment autocompleteFragment;  
    SupportPlaceAutocompleteFragment autocompleteFragment1;  
    **private** List<Polyline> polylinePaths = **new** ArrayList<>();  
    **private** ProgressDialog progressDialog;  
    **public** Button btnTaoDonHang;  
    String diachiTu, diaChiDen ,LatLngTu,LatLngDen;  
    Double KhoangCach;  
    **int** giatienShip;  
    **int** PLACE\_AUTOCOMPLETE\_REQUEST\_CODE\_TU = 1;  
    **int** PLACE\_AUTOCOMPLETE\_REQUEST\_CODE\_DEN = 2;  
    Toolbar myToolbar;  
    Button bntViTriTu;  
    Button bntViTriDen;  
    **protected** LocationManager locationManager;  
    String provider;  
    //nav  
    **public static** TextView *nav\_header\_tenChuShop*, *nav\_header\_ngaySinh*, *nav\_header\_gioiTinh*, *nav\_header\_tenShop*, *nav\_header\_soDienThoai*, *nav\_header\_diaChi*;  
    **public static** View *headerView*;  
    NavigationView navigationView;  
    /\*\*  
    \* ATTENTION: This was auto-generated to implement the App Indexing API.  
    \* See https://g.co/AppIndexing/AndroidStudio for more information.  
    \*/  
    **private** GoogleApiClient client;  
    @Override  
    **protected void** onCreate(Bundle savedInstanceState) {  
    FacebookSdk.*sdkInitialize*(getApplicationContext());  
    **super**.onCreate(savedInstanceState);  
    setContentView(R.layout.*activity\_nav\_drawer*);  
    Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar*);  
    setSupportActionBar(toolbar);  
     
    yeuCauInfo();  
     
    /\*FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);  
    fab.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
    Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH\_LONG)  
    .setAction("Action", null).show();  
    }  
    });\*/  
     
    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.*drawer\_layout*);  
    ActionBarDrawerToggle toggle = **new** ActionBarDrawerToggle(  
    **this**, drawer, toolbar, R.string.*navigation\_drawer\_open*, R.string.*navigation\_drawer\_close*);  
    drawer.setDrawerListener(toggle);  
    toggle.syncState();  
     
    navigationView = (NavigationView) findViewById(R.id.*nav\_view*);  
    navigationView.setNavigationItemSelectedListener(**this**);  
    *headerView* = LayoutInflater.*from*(**this**).inflate(R.layout.*nav\_header\_nav\_drawer*, navigationView, **false**);  
    navigationView.addHeaderView(*headerView*);  
    *nav\_header\_tenChuShop* = (TextView) *headerView*.findViewById(R.id.*nav\_header\_tenChuShop*);  
    *nav\_header\_ngaySinh* = (TextView)*headerView*.findViewById(R.id.*nav\_header\_ngaySinh*);  
    *nav\_header\_gioiTinh* = (TextView)*headerView*.findViewById(R.id.*nav\_header\_gioiTinh*);  
    *nav\_header\_tenShop* = (TextView)*headerView*.findViewById(R.id.*nav\_header\_tenShop*);  
    *nav\_header\_diaChi* = (TextView)*headerView*.findViewById(R.id.*nav\_header\_diaChi*);  
    *nav\_header\_soDienThoai* = (TextView)*headerView*.findViewById(R.id.*nav\_header\_soDienThoai*);  
    //kkkkk  
    mainLayout =(LinearLayout)**this**.findViewById(R.id.*mapLayOutChiTiet*);  
    txtKhoangCach = (TextView)findViewById(R.id.*mapEdKhoangCach*);  
    txtGia = (TextView)findViewById(R.id.*mapEdGiaTien*);  
    txtThoiGian = (TextView)findViewById(R.id.*mapEdThoiGian*);  
    routePoints = **new** ArrayList<LatLng>();  
    btnTaoDonHang = (Button) findViewById(R.id.*mapBtnTaoDonHang*);  
    btnTaoDonHang.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View v) {  
    Intent i = **new** Intent(NavDrawerActivity.**this**,TaoHoaDonActivity.**class**);  
    i.putExtra("tu",diachiTu);  
    i.putExtra("den",diaChiDen);  
    i.putExtra("khoangCach",KhoangCach);  
    i.putExtra("tienShip",giatienShip);  
    i.putExtra("toaDoTu",LatLngTu);  
    i.putExtra("toaDoDen",LatLngDen);  
    startActivity(i);  
    }  
    });  
     
    bntViTriTu = (Button)findViewById(R.id.*mapBntViTriTu*);  
    bntViTriDen = (Button)findViewById(R.id.*mapBntViTriDen*);  
    bntViTriDen.setOnLongClickListener(**new** View.OnLongClickListener() {  
    @Override  
    **public boolean** onLongClick(View view) {  
    bntViTriDen.setText("Vị Trí Giao Hàng");  
    *diemden* = "";  
    mainLayout.setVisibility(View.*GONE*);  
    **if**(*diemden* == "" && *diemtu* == ""){  
    **for** (Polyline polyline:polylinePaths ) {  
    polyline.remove();  
    }  
    mMap.clear();  
    }  
    **return true**;  
    }  
    });  
    bntViTriTu.setOnLongClickListener(**new** View.OnLongClickListener() {  
    @Override  
    **public boolean** onLongClick(View view) {  
    bntViTriTu.setText("Vị Trí Bắt Đầu");  
    *diemtu* = "";  
    mainLayout.setVisibility(View.*GONE*);  
    **if**(*diemden* == "" && *diemtu* == ""){  
    **for** (Polyline polyline:polylinePaths ) {  
    polyline.remove();  
    }  
    mMap.clear();  
     
    }  
    **return true**;  
    }  
    });  
    bntViTriTu.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View view) {  
    **try** {  
    Intent intent =  
    **new** PlaceAutocomplete.IntentBuilder(PlaceAutocomplete.*MODE\_FULLSCREEN*)  
    .build(NavDrawerActivity.**this**);  
    startActivityForResult(intent, PLACE\_AUTOCOMPLETE\_REQUEST\_CODE\_TU);  
    } **catch** (GooglePlayServicesRepairableException e) {  
    // **TODO: Handle the error.** } **catch** (GooglePlayServicesNotAvailableException e) {  
    // **TODO: Handle the error.** }  
    }  
    });  
    bntViTriDen.setOnClickListener(**new** View.OnClickListener() {  
    @Override  
    **public void** onClick(View view) {  
    **try** {  
    Intent intent =  
    **new** PlaceAutocomplete.IntentBuilder(PlaceAutocomplete.*MODE\_FULLSCREEN*)  
    .build(NavDrawerActivity.**this**);  
    startActivityForResult(intent, PLACE\_AUTOCOMPLETE\_REQUEST\_CODE\_DEN);  
    } **catch** (GooglePlayServicesRepairableException e) {  
    // **TODO: Handle the error.** } **catch** (GooglePlayServicesNotAvailableException e) {  
    // **TODO: Handle the error.** }  
    }  
    });  
    // Obtain the SupportMapFragment and get notified when the map is ready to be used.  
    SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()  
    .findFragmentById(R.id.*xmap*);  
    mapFragment.getMapAsync(**this**);  
     
     
     
     
    // ATTENTION: This was auto-generated to implement the App Indexing API.  
    // See https://g.co/AppIndexing/AndroidStudio for more information.  
    client = **new** GoogleApiClient.Builder(**this**).addApi(AppIndex.*API*).build();  
    }  
    **public static void** addInfoToNav(ChuShop chuShop){  
    *nav\_header\_tenChuShop*.setText(chuShop.getTenChuShop());  
    *nav\_header\_ngaySinh*.setText(chuShop.getNgaySinh());  
    String gt ="";  
    **if**(chuShop.isGioiTinh()){gt = "Nam";}**else** {gt = "Nữ";}  
    *nav\_header\_gioiTinh*.setText(gt);  
    *nav\_header\_tenShop*.setText(chuShop.getTenShop());  
    *nav\_header\_diaChi*.setText(chuShop.getDiaChi());  
    *nav\_header\_soDienThoai*.setText(chuShop.getSoDienThoai());  
    }  
    @Override  
    **protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
    **if** (requestCode == PLACE\_AUTOCOMPLETE\_REQUEST\_CODE\_TU) {  
    **if** (resultCode == *RESULT\_OK*) {  
    Place place = PlaceAutocomplete.*getPlace*(**this**, data);  
    Log.*i*("asd", "Place: " + place.getName());  
    bntViTriTu.setText(place.getName().toString());  
    *diemtu* = place.getLatLng().latitude + "," + place.getLatLng().longitude ;  
    mMap.clear();  
    LatLng sydney = **new** LatLng(place.getLatLng().latitude, place.getLatLng().longitude);  
    mMap.addMarker(**new** MarkerOptions().icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*icon\_user*)).position(sydney).title("Shop Của Bạn "));  
    mMap.moveCamera(CameraUpdateFactory.*newLatLng*(sydney));  
    locationTu = place.getLatLng();  
    diachiTu = place.getAddress().toString();  
    Log.*i*("MapsActitivy",place.getAddress().toString());  
    **if** (*diemtu*.length() > 0 && *diemden*.length() > 0) {  
     
    sendRequest(*diemtu*,*diemden*);  
     
     
     
    }  
    } **else if** (resultCode == PlaceAutocomplete.*RESULT\_ERROR*) {  
    Status status = PlaceAutocomplete.*getStatus*(**this**, data);  
    // **TODO: Handle the error.** Log.*i*("asd", status.getStatusMessage());  
     
    } **else if** (resultCode == *RESULT\_CANCELED*) {  
    // The user canceled the operation.  
    }  
    }**else if** (requestCode == PLACE\_AUTOCOMPLETE\_REQUEST\_CODE\_DEN) {  
    **if** (resultCode == *RESULT\_OK*) {  
    Place place = PlaceAutocomplete.*getPlace*(**this**, data);  
    Log.*i*("asd", "Place: " + place.getName());  
    bntViTriDen.setText(place.getName().toString());  
    *diemden* = place.getLatLng().latitude + "," + place.getLatLng().longitude ;  
    mMap.clear();  
    LatLng sydney = **new** LatLng(place.getLatLng().latitude, place.getLatLng().longitude);  
    mMap.addMarker(**new** MarkerOptions().icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*marker\_gift*)).position(sydney).title("Shop Của Bạn "));  
    mMap.moveCamera(CameraUpdateFactory.*newLatLng*(sydney));  
    locationDen = place.getLatLng();  
    diaChiDen = place.getAddress().toString();  
    Log.*i*("MapsActitivy",place.getAddress().toString());  
    **if** (*diemtu*.length() > 0 && *diemden*.length() > 0) {  
    sendRequest(*diemtu*,*diemden*);  
     
     
     
    }  
    }  
    } **else if** (resultCode == PlaceAutocomplete.*RESULT\_ERROR*) {  
    Status status = PlaceAutocomplete.*getStatus*(**this**, data);  
    // **TODO: Handle the error.** Log.*i*("asd", status.getStatusMessage());  
     
    } **else if** (resultCode == *RESULT\_CANCELED*) {  
    // The user canceled the operation.  
    }  
    }  
     
    @SuppressLint("LongLogTag")  
    **private** String getCompleteAddressString(**double** LATITUDE, **double** LONGITUDE) {  
    String strAdd = "";  
    Geocoder geocoder = **new** Geocoder(**this**, Locale.*getDefault*());  
    **try** {  
    List<android.location.Address> addresses = geocoder.getFromLocation(LATITUDE, LONGITUDE, 1);  
    **if** (addresses != **null**) {  
    android.location.Address returnedAddress = addresses.get(0);  
    StringBuilder strReturnedAddress = **new** StringBuilder("");  
     
    **for** (**int** i = 0; i < returnedAddress.getMaxAddressLineIndex(); i++) {  
    strReturnedAddress.append(returnedAddress.getAddressLine(i)).append(",");  
    }  
    strAdd = strReturnedAddress.toString();  
    Log.*w*("My Current loction address", "" + strReturnedAddress.toString());  
    } **else** {  
    Log.*w*("My Current loction address", "No Address returned!");  
    }  
    } **catch** (Exception e) {  
    e.printStackTrace();  
    Log.*w*("My Current loction address", "Canont get Address!");  
    }  
    **return** strAdd;  
    }  
    **public void** connectServer(){  
    startService(**new** Intent(NavDrawerActivity.**this**, MyService.**class**));  
    **try** {  
    JSONObject obj = **new** JSONObject();  
    JSONObject infoMessage = **new** JSONObject();  
    infoMessage.put("tk", "linhtcps01817@fpt.edu.vn");  
    infoMessage.put("mk", "12345678");  
    obj.put("action", "DoLogin");  
    obj.put("infoMessage", infoMessage);  
    MyService.*client*.send(obj.toString());  
    }  
    **catch** (Exception ex){  
    ex.printStackTrace();  
    }  
     
    }  
    **private void** sendRequest(String OriginTu, String OriginDen) {  
    String origin = OriginTu;  
    String destination = OriginDen;  
    LatLngTu = OriginTu ;  
    LatLngDen = OriginDen;  
    **if** (origin.isEmpty()) {  
    Toast.*makeText*(**this**, "Please enter origin address!", Toast.*LENGTH\_SHORT*).show();  
    **return**;  
    }  
    **if** (destination.isEmpty()) {  
    Toast.*makeText*(**this**, "Please enter destination address!", Toast.*LENGTH\_SHORT*).show();  
    **return**;  
    }  
     
    **try** {  
    **new** DirectionFinder(**this**, origin, destination).execute();  
    } **catch** (UnsupportedEncodingException e) {  
    e.printStackTrace();  
    }  
    }  
     
    @Override  
    **public boolean** onMarkerClick(Marker marker) {  
    **return false**;  
    }  
     
    @Override  
    **public void** onDirectionFinderStart() {  
    progressDialog = ProgressDialog.*show*(**this**, "Please wait.",  
    "Finding direction..!", **true**);  
     
    **if** (originMarkers != **null**) {  
    **for** (Marker marker : originMarkers) {  
    marker.remove();  
    }  
    }  
     
    **if** (destinationMarkers != **null**) {  
    **for** (Marker marker : destinationMarkers) {  
    marker.remove();  
    }  
    }  
     
    **if** (polylinePaths != **null**) {  
    **for** (Polyline polyline:polylinePaths ) {  
    polyline.remove();  
    }  
    }  
    }  
     
    @Override  
    **public void** onDirectionFinderSuccess(List<route> routes) {  
     
    progressDialog.dismiss();  
     
    mainLayout.setVisibility(View.*VISIBLE*);  
    polylinePaths = **new** ArrayList<>();  
    originMarkers = **new** ArrayList<>();  
    destinationMarkers = **new** ArrayList<>();  
    mMap.clear();  
     
     
    **for** (route route : routes) {  
     
    txtThoiGian.setText("Thời gian dự kiến " + route.duration.text.replace("mins","Phút") );  
    txtKhoangCach.setText("Khoảng Cách: " + route.distance.text);  
    String a = route.distance.text.trim();  
    **boolean** km = **false**;  
    **if** (a.lastIndexOf("km") > -1) {  
    KhoangCach =Double.*parseDouble*( a.replace("km",""));  
    }**else**{  
    KhoangCach = Double.*parseDouble*( a.replace("m",""));  
    }  
     
     
     
     
    Log.*i*(**this**.toString(),KhoangCach + "");  
    giatienShip = tinhTien(KhoangCach);  
    txtGia.setText("Giá: " + giatienShip + " Đồng" );  
    mMap.moveCamera(CameraUpdateFactory.*newLatLngZoom*(route.startLocation, 16));  
    originMarkers.add(mMap.addMarker(**new** MarkerOptions()  
    .icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*icon\_user*))  
    .title(route.startAddress)  
    .position(route.startLocation)));  
    destinationMarkers.add(mMap.addMarker(**new** MarkerOptions()  
    .icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*marker\_gift*))  
    .title(route.endAddress)  
    .position(route.endLocation)));  
     
    PolylineOptions polylineOptions = **new** PolylineOptions().  
    geodesic(**true**).  
    color(Color.*BLUE*).  
    width(10);  
     
    **for** (**int** i = 0; i < route.*points*.size(); i++)  
    polylineOptions.add(route.*points*.get(i));  
     
    polylinePaths.add(mMap.addPolyline(polylineOptions));  
    }  
    }  
    **public int** tinhTien(Double KhoangCach){  
    **int** Tien ;  
    **if** (KhoangCach >0 && KhoangCach <= 2){  
    Tien = 10000;  
    }**else if** (KhoangCach >2 && KhoangCach <= 5){  
    Tien = 15000;  
    }**else if** (KhoangCach >5 && KhoangCach <= 10){  
    Tien = 30000;  
    }**else** {  
    Tien = 50000;  
    }  
    **return** Tien;  
    }  
    @Override  
    **public void** onMapReady(GoogleMap googleMap) {  
    mMap = googleMap;  
    mMap.setMyLocationEnabled(**true**);  
    locationManager = (LocationManager) getSystemService(Context.*LOCATION\_SERVICE*);  
     
    //Cài đặt điều kiện lấy Location Provider  
    Criteria criteria = **new** Criteria();  
    provider = locationManager.getBestProvider(criteria, **false**);  
     
    locationManager.requestLocationUpdates(LocationManager.*GPS\_PROVIDER*, 1000, 20, **this**);  
    Location location = locationManager.getLastKnownLocation(provider);  
    **if** (location != **null**) {  
    LatLng loc = **new** LatLng(location.getLatitude(), location.getLongitude());  
    Log.*i*(**this**.toString(), loc + "");  
     
    **if** (mMap != **null**) {  
    mMap.animateCamera(CameraUpdateFactory.*newLatLngZoom*(loc, 16.0f));  
    }  
    }  
     
    mMap.setOnMapClickListener(**new** GoogleMap.OnMapClickListener() {  
     
    @Override  
    **public void** onMapClick(LatLng point) {  
    // **TODO Auto-generated method stub** Log.*i*("MapsActitivy", "asd" + bntViTriDen.getText().toString());  
    **if** (bntViTriTu.getText().toString().equals("Vị Trí Bắt Đầu")){  
    *diemtu* = point.latitude + "," + point.longitude;  
    mMap.clear();  
    LatLng sydney = point;  
    mMap.addMarker(**new** MarkerOptions().icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*icon\_user*)).position(sydney).title("Shop Của Bạn "));  
    mMap.moveCamera(CameraUpdateFactory.*newLatLng*(sydney));  
    locationTu = point;  
    diachiTu = getCompleteAddressString(point.latitude, point.longitude);  
    bntViTriTu.setText(diachiTu);  
    Log.*i*("MapsActitivy", diachiTu);  
    **if** (*diemtu*.length() > 0 && *diemden*.length() > 0) {  
     
    sendRequest(*diemtu*, *diemden*);  
     
     
    }  
    }  
    **else if** (bntViTriDen.getText().toString().equals("Vị Trí Giao Hàng")){  
    *diemden* = point.latitude + "," + point.longitude;  
    mMap.clear();  
    LatLng sydney = point;  
    mMap.addMarker(**new** MarkerOptions().icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*marker\_gift*)).position(sydney).title("Shop Của Bạn "));  
    mMap.moveCamera(CameraUpdateFactory.*newLatLng*(sydney));  
    locationDen = point;  
    diaChiDen = getCompleteAddressString(point.latitude, point.longitude);  
    bntViTriDen.setText(diaChiDen);  
    Log.*i*("MapsActitivy", diaChiDen);  
    **if** (*diemtu*.length() > 0 && *diemden*.length() > 0) {  
     
    sendRequest(*diemtu*, *diemden*);  
     
     
    }  
    }  
     
    }  
    });  
    }  
     
    /\*\*  
    \* ATTENTION: This was auto-generated to implement the App Indexing API.  
    \* See https://g.co/AppIndexing/AndroidStudio for more information.  
    \*/  
    **public** Action getIndexApiAction() {  
    Thing object = **new** Thing.Builder()  
    .setName("Maps Page") // **TODO: Define a title for the content shown.** // **TODO: Make sure this auto-generated URL is correct.** .setUrl(Uri.*parse*("http://[ENTER-YOUR-URL-HERE]"))  
    .build();  
    **return new** Action.Builder(Action.*TYPE\_VIEW*)  
    .setObject(object)  
    .setActionStatus(Action.*STATUS\_TYPE\_COMPLETED*)  
    .build();  
    }  
     
    @Override  
    **public void** onStart() {  
    **super**.onStart();  
     
    // ATTENTION: This was auto-generated to implement the App Indexing API.  
    // See https://g.co/AppIndexing/AndroidStudio for more information.  
    client.connect();  
    AppIndex.*AppIndexApi*.start(client, getIndexApiAction());  
    }  
     
    @Override  
    **public void** onStop() {  
    **super**.onStop();  
     
    // ATTENTION: This was auto-generated to implement the App Indexing API.  
    // See https://g.co/AppIndexing/AndroidStudio for more information.  
    AppIndex.*AppIndexApi*.end(client, getIndexApiAction());  
    client.disconnect();  
    }  
     
     
    @Override  
    **public void** onLocationChanged(Location location) {  
    LatLng loc = **new** LatLng(location.getLatitude(), location.getLongitude());  
     
    **if** (mMap != **null**) {  
    mMap.animateCamera(CameraUpdateFactory.*newLatLngZoom*(loc, 16.0f));  
    }  
    }  
     
    @Override  
    **public void** onStatusChanged(String s, **int** i, Bundle bundle) {  
     
    }  
     
    @Override  
    **public void** onProviderEnabled(String s) {  
     
    }  
     
    @Override  
    **public void** onProviderDisabled(String s) {  
     
    }  
     
    @Override  
    **public boolean** onCreateOptionsMenu(Menu menu) {  
    // Inflate the menu; this adds items to the action bar if it is present.  
    getMenuInflater().inflate(R.menu.*nav\_drawer*, menu);  
    **return true**;  
    }  
     
    @Override  
    **public boolean** onOptionsItemSelected(MenuItem item) {  
    // Handle action bar item clicks here. The action bar will  
    // automatically handle clicks on the Home/Up button, so long  
    // as you specify a parent activity in AndroidManifest.xml.  
    **int** id = item.getItemId();  
     
    //noinspection SimplifiableIfStatement  
    **if** (id == R.id.*dinhvi*) {  
    locationManager.requestLocationUpdates(LocationManager.*GPS\_PROVIDER*, 1000, 20, **this**);  
    Location location = locationManager.getLastKnownLocation(provider);  
    **return true**;  
    }  
     
    **return super**.onOptionsItemSelected(item);  
    }  
     
     
    @Override  
    **public boolean** onNavigationItemSelected(MenuItem item) {  
    // Handle navigation view item clicks here.  
    **int** id = item.getItemId();  
     
    **if** (id == R.id.*nav\_qlHoaDon*) {  
    Intent i = **new** Intent(NavDrawerActivity.**this**,QuanLyHoaDonChuShopActivity.**class**);  
    startActivity(i);  
    } **else if** (id == R.id.*nav\_dangxuat*) {  
    AlertDialog.Builder bui=**new** AlertDialog.Builder(NavDrawerActivity.**this**);  
    LayoutInflater inf=getLayoutInflater();  
    **final** View view=inf.inflate(R.layout.*dialogaddlop*, **null**);  
    bui.setView(view);  
    bui.setCancelable(**false**);  
    bui.setNegativeButton("OK", **new** DialogInterface.OnClickListener() {  
     
    @Override  
    **public void** onClick(DialogInterface dialog, **int** which) {  
    **try**{  
    LoginManager.*getInstance*().logOut();  
    }  
    **catch** (Exception e){  
    e.printStackTrace();  
    }**finally** {  
    utils.*alreadyLogin* = **false**;  
    TinyDB tinyDB = **new** TinyDB(NavDrawerActivity.**this**);  
    tinyDB.putString("idTaiKhoan","");  
    startActivity(**new** Intent(NavDrawerActivity.**this**, MainActivity.**class**));  
    }  
    }  
    });  
    bui.setPositiveButton("Cancel", **new** DialogInterface.OnClickListener() {  
     
    @Override  
    **public void** onClick(DialogInterface dialog, **int** which) {  
    // **TODO Auto-generated method stub** dialog.cancel();  
    }  
    });  
     
    bui.show();  
    } **else if**(id==R.id.*nav\_cnThongTin*){  
    Intent i = **new** Intent(NavDrawerActivity.**this**,UpdateThongTinChuShop.**class**);  
    i.putExtra("tenChuShop",*nav\_header\_tenChuShop*.getText().toString());  
    i.putExtra("ngaySinh",*nav\_header\_ngaySinh*.getText().toString());  
    i.putExtra("gioiTinh",*nav\_header\_gioiTinh*.getText().toString());  
    i.putExtra("tenShop",*nav\_header\_tenShop*.getText().toString());  
    i.putExtra("diaChi",*nav\_header\_diaChi*.getText().toString());  
    i.putExtra("soDienThoai",*nav\_header\_soDienThoai*.getText().toString());  
    startActivity(i);  
    }  
     
    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.*drawer\_layout*);  
    drawer.closeDrawer(GravityCompat.*START*);  
    **return true**;  
    }  
    **private void** yeuCauInfo(){  
    **try** {  
    BuildMessage xMessage = **new** BuildMessage();  
    JSONObject infoMessage = **new** JSONObject();  
    infoMessage.put("loadInfo", **true**);  
    String message = xMessage.buildMessage("LoadInformation", infoMessage);  
    System.*out*.println(message);  
    MyService.*client*.send(message);  
    }  
    **catch** (Exception ex){  
    ex.printStackTrace();  
    }  
    }  
   }

### QuanLyHoaDonChuShopActivity:

**package** com.poly.shipper.activity;  
  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.MenuItem;  
**import** android.view.View;  
**import** android.widget.AdapterView;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.ListView;  
**import** android.widget.Spinner;  
  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.adapter.CustomListViewAdapter;  
**import** com.poly.shipper.main.BuildMessage;  
**import** com.poly.shipper.model.HoaDon;  
**import** com.poly.shipper.service.MyService;  
  
**import** org.json.JSONObject;  
  
**import** java.util.ArrayList;  
  
**public class** QuanLyHoaDonChuShopActivity **extends** AppCompatActivity {  
  
 **private** Spinner qlHoaDonSpnLoc;  
 **private** String[] listLoaiLoc = {"TẤT CẢ HÓA ĐƠN","HÓA ĐƠN ĐÃ HOÀN THÀNH","HÓA ĐƠN ĐÃ NHẬN HÀNG","HÓA ĐƠN CHƯA NHẬN HÀNG"};  
 **public static** ListView *qlHoaDonLvHoaDon*;  
 **public static** CustomListViewAdapter *customListViewAdapter*;  
 **public static** ArrayList<HoaDon> *hoaDons*;  
 **public** ArrayList<HoaDon> filterHoaDon = **null**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_quan\_ly\_hoa\_don\_chu\_shop*);  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 *hoaDons* = **new** ArrayList<>();  
 *qlHoaDonLvHoaDon* = (ListView)findViewById(R.id.*qlHoaDonLvHoaDon*);  
 qlHoaDonSpnLoc = (Spinner)findViewById(R.id.*qlHoaDonSpnLoc*);  
 ArrayAdapter<String> adapterSpinner = **new** ArrayAdapter<String>(QuanLyHoaDonChuShopActivity.**this**, android.R.layout.*simple\_list\_item\_1*,listLoaiLoc);  
 qlHoaDonSpnLoc.setAdapter(adapterSpinner);  
 yeuCauListHoaDon();  
 filterHoaDon = **new** ArrayList<>();  
 qlHoaDonSpnLoc.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
 @Override  
 **public void** onItemSelected(AdapterView<?> parent, View view, **int** position, **long** id) {  
  
 **if** (*customListViewAdapter* != **null**) {  
 ArrayList<HoaDon> list = *hoaDons*;  
 filterHoaDon.clear();  
  
 **if** (position == 0) {  
 filterHoaDon.addAll(*hoaDons*);  
 }  
 **else if** (position == 1) {  
  
 **for** (HoaDon hoaDon : list) {  
 **if** (hoaDon.getIdTinhTrang() == 2) {  
 filterHoaDon.add(hoaDon);  
 }  
 }  
 }**else if** (position == 2) {  
  
 **for** (HoaDon hoaDon : list) {  
 **if** (hoaDon.getIdTinhTrang() == 1) {  
 filterHoaDon.add(hoaDon);  
 }  
 }  
 }**else if** (position == 3) {  
  
 **for** (HoaDon hoaDon : list) {  
 **if** (hoaDon.getIdTinhTrang() == 0) {  
 filterHoaDon.add(hoaDon);  
 }  
 }  
 }  
  
 *customListViewAdapter*.setList(filterHoaDon);  
  
 }  
 }  
  
 @Override  
 **public void** onNothingSelected(AdapterView<?> parent) {  
  
 }  
 });  
  
  
 *qlHoaDonLvHoaDon*.setOnItemClickListener(**new** AdapterView.OnItemClickListener() {  
 @Override  
 **public void** onItemClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 Intent i = **new** Intent(QuanLyHoaDonChuShopActivity.**this**,ChiTietHoaDonActivity.**class**);  
 i.putExtra("index",position);  
 startActivity(i);  
 }  
 });  
  
 }  
 **public static void** setAdapterToActivity(Context context,ArrayList<HoaDon> list){  
 *hoaDons*.addAll(list);  
 *customListViewAdapter* = **new** CustomListViewAdapter(context,list);  
 *qlHoaDonLvHoaDon*.setAdapter(*customListViewAdapter*);  
 }  
 **private void** yeuCauListHoaDon(){  
 **try** {  
 BuildMessage xMessage = **new** BuildMessage();  
 JSONObject infoMessage = **new** JSONObject();  
 infoMessage.put("loadList", **true**);  
 String message = xMessage.buildMessage("QuanLyHoaDonChuShop", infoMessage);  
 System.*out*.println(message);  
 MyService.*client*.send(message);  
 }  
 **catch** (Exception ex){  
 ex.printStackTrace();  
 }  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **if**(item.getItemId()==android.R.id.*home*){  
 finish();  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
}

### TaoHoaDonActivity:

**package** com.poly.shipper.activity;  
  
**import** android.app.Dialog;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.database.Cursor;  
**import** android.net.Uri;  
**import** android.os.Bundle;  
**import** android.provider.MediaStore;  
**import** android.provider.Settings;  
**import** android.support.v4.view.ViewPager;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.MenuItem;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.RadioButton;  
**import** android.widget.RadioGroup;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.adapter.\*;  
**import** com.poly.shipper.asynctask.upload;  
**import** com.poly.shipper.main.BuildMessage;  
**import** com.poly.shipper.model.CirclePageIndicator;  
**import** com.poly.shipper.model.HoaDon;  
**import** com.poly.shipper.model.PageIndicator;  
**import** com.poly.shipper.service.MyService;  
**import** com.poly.shipper.utils.utils;  
  
**import** org.json.JSONObject;  
  
**import** java.util.ArrayList;  
**import** java.util.StringTokenizer;  
  
**public class** TaoHoaDonActivity **extends** AppCompatActivity {  
 //viewpager  
 **private** CustomPagerAdapter mAdapter;  
 **private** ViewPager mPager;  
 **private** PageIndicator mIndicator;  
 **public static** ArrayList<String> *image\_intro* = **new** ArrayList<String>();  
 **private** String imgDecodableString;  
 **private static int** *RESULT\_LOAD\_IMAGE* = 1;  
 //form  
 **private** Button donHangBtnChonAnh,donHangBtnTaoDonHang;  
 **private** EditText donHangEdTenHang,donHangEdTenNguoiGui,donHangEdSdtNguoiGui,donHangEdTu,donHangEdDen,donHangEdTienHang,  
 donHangEdTenNguoiNhan,donHangEdSdtNguoiNhan,donHangEdGhiChu;  
 **private** TextView donHangTvKhoangCach,donHangTvTienShip,donHangTvBaoLoiAnh;  
 **private** RadioGroup donHangRgNguoiTra;  
 **private** RadioButton donHangRbNguoiGui,donHangRbNguoiNhan;  
 **private** String tu, den,toaDoTu,toaDoDen;  
 **private double** khoangCach;  
 **private int** tienShip;  
 //Hóa đơn  
 **private** HoaDon hoaDon;  
 //dialog mã hoàn thành hóa đơn lúc tạo thành công  
 **public static** Dialog *dialog*;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_tao\_hoa\_don*);  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 utils.*context* = **this**;  
 *image\_intro*.clear();  
 khaiBao();  
 viewPager();  
 chonAnh();  
 taoHoaDon();  
 }  
  
 **public static void** UpLoadImage(Context context,String idHoaDon) {  
 upload xupload = **new** upload(context, idHoaDon, *image\_intro*);  
 xupload.execute();  
 }  
 **public void** khaiBao(){  
 //dialog  
  
 //lấy dữ liệu từ map  
 Intent i = getIntent();  
 tu = i.getStringExtra("tu");  
 den = i.getStringExtra("den");  
 khoangCach = i.getDoubleExtra("khoangCach",1);  
 toaDoTu = i.getStringExtra("toaDoTu");  
 toaDoDen = i.getStringExtra("toaDoDen");  
 tienShip = i.getIntExtra("tienShip",0);  
 //Các trường  
 donHangBtnChonAnh = (Button)findViewById(R.id.*hoaDonBtnChonAnh*);  
 donHangTvBaoLoiAnh = (TextView)findViewById(R.id.*hoaDonTvBaoLoiAnh*);  
 donHangEdTenHang = (EditText)findViewById(R.id.*hoaDonEdTenHang*);  
 donHangEdTenNguoiGui = (EditText)findViewById(R.id.*hoaDonEdTenNguoiGui*);  
 donHangEdSdtNguoiGui = (EditText)findViewById(R.id.*hoaDonEdSdtNguoiGui*);  
 donHangEdTu = (EditText)findViewById(R.id.*hoaDonEdTu*);  
 donHangEdTu.setText(tu);  
 donHangEdTu.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 finish();  
 }  
 });  
 donHangEdDen = (EditText)findViewById(R.id.*hoaDonEdDen*);  
 donHangEdDen.setText(den);  
 donHangEdDen.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 finish();  
 }  
 });  
 donHangTvKhoangCach = (TextView)findViewById(R.id.*hoaDonTvKhoangCach*);  
 donHangTvKhoangCach.setText("Khoảng cách : "+String.*valueOf*(khoangCach)+" Km");  
 donHangEdTienHang = (EditText)findViewById(R.id.*hoaDonEdTienHang*);  
 donHangTvTienShip = (TextView)findViewById(R.id.*hoaDonTvTienShip*);  
 donHangTvTienShip.setText("Tiền ship : " +String.*valueOf*(tienShip)+" VND");  
 donHangRgNguoiTra = (RadioGroup)findViewById(R.id.*hoaDonRgNguoiTra*);  
 donHangRbNguoiGui = (RadioButton)findViewById(R.id.*hoaDonRbNguoiGui*);  
 donHangRbNguoiNhan = (RadioButton)findViewById(R.id.*hoaDonRbNguoiNhan*);  
 donHangEdTenNguoiNhan = (EditText)findViewById(R.id.*hoaDonEdTenNguoiNhan*);  
 donHangEdSdtNguoiNhan = (EditText)findViewById(R.id.*hoaDonEdSdtNguoiNhan*);  
 donHangEdGhiChu = (EditText)findViewById(R.id.*hoaDonEdGhiChu*);  
 donHangBtnTaoDonHang = (Button)findViewById(R.id.*hoaDonBtnTaoDonHang*);  
 }  
 **public void** taoHoaDon(){  
 donHangBtnTaoDonHang.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if**(!batLoi()){  
 hoaDon = **new** HoaDon(donHangEdTenHang.getText().toString(),  
 donHangEdTenNguoiGui.getText().toString(),  
 donHangEdSdtNguoiGui.getText().toString(),  
 donHangEdTu.getText().toString(),  
 toaDoTu,  
 donHangEdDen.getText().toString(),  
 toaDoDen,  
 khoangCach,  
 Integer.*parseInt*(donHangEdTienHang.getText().toString()),  
 tienShip,  
 donHangRgNguoiTra.getCheckedRadioButtonId() == R.id.*hoaDonRbNguoiGui* ? **true** : **false**,  
 donHangEdGhiChu.getText().toString(),  
 donHangEdTenNguoiNhan.getText().toString(),  
 donHangEdSdtNguoiNhan.getText().toString(),  
 "idchushop"  
 );  
 guiHoaDon(hoaDon);  
 }  
 }  
 });  
 }  
 **private void** guiHoaDon(HoaDon hoaDon){  
 **try** {  
 BuildMessage xMessage = **new** BuildMessage();  
 JSONObject infoMessage = **new** JSONObject();  
 infoMessage.put("tenHang", hoaDon.getTenHang());  
 infoMessage.put("tenNguoiGui", hoaDon.getTenNguoiGui());  
 infoMessage.put("sdtNguoiGui", hoaDon.getSdtNguoiGui());  
 infoMessage.put("tu", hoaDon.getTu());  
 infoMessage.put("toaDoTu", hoaDon.getToaDoTu());  
 infoMessage.put("den", hoaDon.getDen());  
 infoMessage.put("toaDoDen", hoaDon.getToaDoDen());  
 infoMessage.put("khoangCach", hoaDon.getKhoangCach());  
 infoMessage.put("tienHang", hoaDon.getTienHang());  
 infoMessage.put("tienShip", hoaDon.getTienShip());  
 infoMessage.put("nguoiTra", hoaDon.isNguoiTra());  
 infoMessage.put("tenNguoiNhan", hoaDon.getTenNguoiNhan());  
 infoMessage.put("sdtNguoiNhan", hoaDon.getSdtNguoiNhan());  
 infoMessage.put("ghiChu", hoaDon.getGhiChu());  
 infoMessage.put("idChuShop", hoaDon.getIdchushop());  
 //infoMessage.put("gioitinh", thongTinCbGioiTinh.getCheckedRadioButtonId() == R.id.thongTinCbNu ? 0 : 1);  
 String message = xMessage.buildMessage("TaoHoaDon", infoMessage);  
 System.*out*.println(message);  
 MyService.*client*.send(message);  
 }  
 **catch** (Exception ex){  
 ex.printStackTrace();  
 }  
 }  
 **private void** chonAnh(){  
 donHangBtnChonAnh.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent galleryIntent = **new** Intent(Intent.*ACTION\_PICK*, MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*);  
 startActivityForResult(galleryIntent, *RESULT\_LOAD\_IMAGE*);  
 }  
 });  
 }  
 **public void** viewPager(){  
 *image\_intro* = **new** ArrayList<String>();  
  
 mAdapter = **new** CustomPagerAdapter(**this**,*image\_intro*);  
  
 mPager = (ViewPager)findViewById(R.id.*hoaDonVpLoadAnh*);  
  
 mIndicator = (CirclePageIndicator)findViewById(R.id.*indicator*);  
  
  
 mIndicator.setOnPageChangeListener(**new** ViewPager.OnPageChangeListener() {  
 @Override  
 **public void** onPageSelected(**int** position) {  
  
 }  
  
 @Override  
 **public void** onPageScrolled(**int** position, **float** positionOffset, **int** positionOffsetPixels) {  
 }  
  
 @Override  
 **public void** onPageScrollStateChanged(**int** state) {  
 }  
 });  
 }  
 **protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
 **super**.onActivityResult(requestCode, resultCode, data);  
 **try** {  
 // When an Image is picked  
 **if** (requestCode == *RESULT\_LOAD\_IMAGE* && resultCode == *RESULT\_OK* && **null** != data) {  
 // Get the Image from data  
  
 Uri selectedImage = data.getData();  
 String[] filePathColumn = { MediaStore.Images.Media.*DATA* };  
  
 // Get the cursor  
 Cursor cursor = getContentResolver().query(selectedImage,  
 filePathColumn, **null**, **null**, **null**);  
 // Move to first row  
 cursor.moveToFirst();  
  
 **int** columnIndex = cursor.getColumnIndex(filePathColumn[0]);  
 imgDecodableString = cursor.getString(columnIndex);  
 System.*out*.println(imgDecodableString);  
 cursor.close();  
 // Set the Image in ImageView after decoding the String  
  
 *image\_intro*.add(imgDecodableString);  
 mPager.setAdapter(mAdapter);  
 mIndicator.setViewPager(mPager);  
 **if**(*image\_intro*.size()>2){  
 donHangBtnChonAnh.setVisibility(View.*GONE*);  
 }**else** {  
 donHangTvBaoLoiAnh.setVisibility(View.*GONE*);  
 donHangBtnChonAnh.setText("CHỌN THÊM ẢNH");  
 donHangBtnChonAnh.setAlpha((**float**) 0.2);  
 }  
 } **else** {  
 Toast.*makeText*(**this**, "Bạn vẫn chưa chọn hình!",  
 Toast.*LENGTH\_LONG*).show();  
 }  
 } **catch** (Exception e) {  
 Toast.*makeText*(**this**, "Something went wrong", Toast.*LENGTH\_LONG*)  
 .show();  
 }  
  
 }  
 **private boolean** batLoi(){  
 **boolean** err = **true**;  
 **if**(*image\_intro*.size()==0||donHangEdSdtNguoiNhan.getText().toString().isEmpty()||donHangEdTenNguoiNhan.getText().toString().isEmpty()||  
 donHangEdTienHang.getText().toString().isEmpty()||donHangEdDen.getText().toString().isEmpty()||donHangEdTu.getText().toString().isEmpty()||  
 donHangEdSdtNguoiGui.getText().toString().isEmpty()||donHangEdTenNguoiGui.getText().toString().isEmpty()||donHangEdTenHang.getText().toString().isEmpty()){  
 **if**(*image\_intro*.size()==0){  
 donHangTvBaoLoiAnh.setVisibility(View.*VISIBLE*);  
 donHangTvBaoLoiAnh.setText("Bạn chưa chọn ảnh nào!");  
 }  
 **if**(donHangEdSdtNguoiNhan.getText().toString().isEmpty()){  
 donHangEdSdtNguoiNhan.setError("Không được bỏ trống!");  
 donHangEdSdtNguoiNhan.requestFocus();  
 }  
 **if**(donHangEdTenNguoiNhan.getText().toString().isEmpty()){  
 donHangEdTenNguoiNhan.setError("Không được bỏ trống!");  
 donHangEdTenNguoiNhan.requestFocus();  
 }  
 **if**(donHangEdTienHang.getText().toString().isEmpty()){  
 donHangEdTienHang.setError("Không được bỏ trống!");  
 donHangEdTienHang.requestFocus();  
 }  
 **if**(donHangEdDen.getText().toString().isEmpty()){  
 donHangEdDen.setError("Không được bỏ trống!");  
 donHangEdDen.requestFocus();  
 }  
 **if**(donHangEdTu.getText().toString().isEmpty()){  
 donHangEdTu.setError("Không được bỏ trống!");  
 donHangEdTu.requestFocus();  
 }  
 **if**(donHangEdSdtNguoiGui.getText().toString().isEmpty()){  
 donHangEdSdtNguoiGui.setError("Không được bỏ trống!");  
 donHangEdSdtNguoiGui.requestFocus();  
 }  
 **if**(donHangEdTenNguoiGui.getText().toString().isEmpty()){  
 donHangEdTenNguoiGui.setError("Không được bỏ trống!");  
 donHangEdTenNguoiGui.requestFocus();  
 }  
 **if**(donHangEdTenHang.getText().toString().isEmpty()){  
 donHangEdTenHang.setError("Không được bỏ trống!");  
 donHangEdTenHang.requestFocus();  
 }  
 }**else** {  
 **if**(donHangEdSdtNguoiNhan.getText().toString().length()<10||donHangEdSdtNguoiNhan.getText().toString().length()>11||  
 donHangEdSdtNguoiGui.getText().toString().length()<10||donHangEdSdtNguoiGui.getText().toString().length()>11||  
 Integer.*parseInt*(donHangEdTienHang.getText().toString())<1000){  
 **if**(donHangEdSdtNguoiNhan.getText().toString().length()<10||donHangEdSdtNguoiNhan.getText().toString().length()>11){  
 donHangEdSdtNguoiNhan.setError("Số điện thoại phải từ 10 đến 11 số!");  
 donHangEdSdtNguoiNhan.requestFocus();  
 }  
 **if**(donHangEdSdtNguoiGui.getText().toString().length()<10||donHangEdSdtNguoiGui.getText().toString().length()>11){  
 donHangEdSdtNguoiGui.setError("Số điện thoại phải từ 10 đến 11 số!");  
 donHangEdSdtNguoiGui.requestFocus();  
 }  
 **if**(Integer.*parseInt*(donHangEdTienHang.getText().toString())<1000){  
 donHangEdTienHang.setError("Tiền hàng phải lớn hơn 1.000 đồng");  
 donHangEdTienHang.requestFocus();  
 }  
 }**else** {  
 **if**(donHangEdTenNguoiNhan.getText().toString().indexOf(" ")==0||  
 donHangEdDen.getText().toString().indexOf(" ")==0||donHangEdTu.getText().toString().indexOf(" ")==0||  
 donHangEdTenNguoiGui.getText().toString().indexOf(" ")==0||donHangEdTenHang.getText().toString().indexOf(" ")==0){  
 **if**(donHangEdTenNguoiNhan.getText().toString().indexOf(" ")==0){  
 donHangEdTenNguoiNhan.setError("Kí tự đầu tiên không chứa khoảng trắng!");  
 donHangEdTenNguoiNhan.requestFocus();  
 }  
 **if**(donHangEdDen.getText().toString().indexOf(" ")==0){  
 donHangEdDen.setError("Kí tự đầu tiên không chứa khoảng trắng!");  
 donHangEdDen.requestFocus();  
 }  
 **if**(donHangEdTu.getText().toString().indexOf(" ")==0){  
 donHangEdTu.setError("Kí tự đầu tiên không chứa khoảng trắng!");  
 donHangEdTu.requestFocus();  
 }  
 **if**(donHangEdTenNguoiGui.getText().toString().indexOf(" ")==0){  
 donHangEdTenNguoiGui.setError("Kí tự đầu tiên không chứa khoảng trắng!");  
 donHangEdTenNguoiGui.requestFocus();  
 }  
 **if**(donHangEdTenHang.getText().toString().indexOf(" ")==0){  
 donHangEdTenHang.setError("Kí tự đầu tiên không chứa khoảng trắng!");  
 donHangEdTenHang.requestFocus();  
 }  
 }**else** {  
 err = **false**;  
 }  
 }  
 }  
 **return** err;  
 }  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **if**(item.getItemId()==android.R.id.*home*){  
 finish();  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
}

**TrackingActivity**

**package** com.poly.shipper.activity;  
  
**import** android.Manifest;  
**import** android.app.Activity;  
**import** android.app.Dialog;  
**import** android.app.ProgressDialog;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.pm.PackageManager;  
**import** android.graphics.Color;  
**import** android.graphics.drawable.ColorDrawable;  
**import** android.location.Criteria;  
**import** android.location.Location;  
**import** android.location.LocationListener;  
**import** android.location.LocationManager;  
**import** android.os.Bundle;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.view.Window;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.LinearLayout;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
**import** java.util.Timer;  
**import** com.google.android.gms.maps.CameraUpdateFactory;  
**import** com.google.android.gms.maps.GoogleMap;  
**import** com.google.android.gms.maps.OnMapReadyCallback;  
**import** com.google.android.gms.maps.SupportMapFragment;  
**import** com.google.android.gms.maps.model.BitmapDescriptorFactory;  
**import** com.google.android.gms.maps.model.LatLng;  
**import** com.google.android.gms.maps.model.Marker;  
**import** com.google.android.gms.maps.model.MarkerOptions;  
**import** com.google.android.gms.maps.model.Polyline;  
**import** com.google.android.gms.maps.model.PolylineOptions;  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.main.BuildMessage;  
**import** com.poly.shipper.modules.DirectionFinder;  
**import** com.poly.shipper.modules.DirectionFinderListener;  
**import** com.poly.shipper.modules.route;  
**import** com.poly.shipper.service.MyService;  
**import** com.poly.shipper.utils.utils;  
  
**import** org.json.JSONObject;  
  
**import** java.io.UnsupportedEncodingException;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**import** java.util.TimerTask;  
  
**public class** TrackingActivity **extends** AppCompatActivity **implements** OnMapReadyCallback ,LocationListener,DirectionFinderListener {  
 **public static** GoogleMap *mMap*;  
 GoogleMap Map;  
 **public static** Marker *mMarker*;  
 **public** Location Mylocation;  
 **protected** LocationManager locationManager;  
 String provider;  
 Button bntTrack;  
 **public static** String *toadoden* = "";  
 **public static** String *trangthai* = "";  
 LinearLayout ln;  
 List<LatLng> routePoints = **null**;  
 **private** List<Marker> originMarkers = **new** ArrayList<>();  
 **private** List<Marker> destinationMarkers = **new** ArrayList<>();  
  
 **private** List<Polyline> polylinePaths = **new** ArrayList<>();  
 **private** ProgressDialog progressDialog;  
 **public** Button btnTaoDonHang;  
 **public** TextView txtKhoangCach;  
 **public** TextView txtGia;  
 **public** TextView txtThoiGian,txtTinhTrang;  
 **public static** String *toadoShiper* = "";  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_tracking*);  
 utils.*context* = **this**;  
 search(**this**);  
  
 ln = (LinearLayout) findViewById(R.id.*mapLayOutChiTietTracking*);  
 txtKhoangCach = (TextView)findViewById(R.id.*mapTrackingEdKhoangCach*);  
 txtThoiGian = (TextView)findViewById(R.id.*mapTrackingEdThoiGian*);  
 txtTinhTrang = (TextView)findViewById(R.id.*mapTrackingTinhTrang*);  
 SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()  
 .findFragmentById(R.id.*mapTracking*);  
 mapFragment.getMapAsync(**this**);  
 **if** (ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.*ACCESS\_FINE\_LOCATION*) != PackageManager.*PERMISSION\_GRANTED*) {  
 **if** (ActivityCompat.*shouldShowRequestPermissionRationale*(**this**,  
 Manifest.permission.*ACCESS\_COARSE\_LOCATION*)) {  
  
 } **else** {  
 // do request the permission  
 ActivityCompat.*requestPermissions*(**this**, **new** String[]{Manifest.permission.*ACCESS\_FINE\_LOCATION*}, 8);  
 }  
 }  
  
  
 }  
 **public void** search(Activity activity){  
  
 **final** Dialog dialog = **new** Dialog(activity);  
 dialog.requestWindowFeature(Window.*FEATURE\_NO\_TITLE*);  
 dialog.setContentView(R.layout.*dialog\_search*);  
 bntTrack = (Button) findViewById(R.id.*showdialog*);  
 bntTrack.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 dialog.show();  
 }  
 });  
 **final** EditText txt1 = (EditText)dialog.findViewById(R.id.*edtTim*);  
 Intent i = getIntent();  
 **if**(i.getStringExtra("idHoaDon")!=**null**){  
 txt1.setText(i.getStringExtra("idHoaDon"));  
 }  
 Button bnt = (Button)dialog.findViewById(R.id.*dialogTim*);  
 bnt.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **try** {  
 timer();  
 BuildMessage xMessage = **new** BuildMessage();  
 JSONObject infoMessage = **new** JSONObject();  
 infoMessage.put("idHoaDon", Integer.*parseInt*(txt1.getText().toString().trim()));  
 String message = xMessage.buildMessage("SetMaTracking", infoMessage);  
 System.*out*.println(message);  
 MyService.*client*.send(message);  
  
 dialog.dismiss();  
 }  
 **catch** (Exception ex){  
 ex.printStackTrace();  
 }  
 }  
 });  
 dialog.show();  
 }  
  
 @Override  
 **public void** onMapReady(GoogleMap googleMap) {  
 *mMap* = googleMap;  
 *mMap*.setMyLocationEnabled(**true**);  
 locationManager = (LocationManager) getSystemService(Context.*LOCATION\_SERVICE*);  
  
 //Cài đặt điều kiện lấy Location Provider  
 Criteria criteria = **new** Criteria();  
 provider = locationManager.getBestProvider(criteria, **false**);  
 **if** (ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.*ACCESS\_FINE\_LOCATION*) != PackageManager.*PERMISSION\_GRANTED* && ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.*ACCESS\_COARSE\_LOCATION*) != PackageManager.*PERMISSION\_GRANTED*) {  
 **return**;  
 }  
 **if** (ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.*ACCESS\_FINE\_LOCATION*) != PackageManager.*PERMISSION\_GRANTED* && ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.*ACCESS\_COARSE\_LOCATION*) != PackageManager.*PERMISSION\_GRANTED*) {  
  
 **return**;  
 }  
 locationManager.requestLocationUpdates(LocationManager.*GPS\_PROVIDER*, 1000, 10, **this**);  
 Location location = locationManager.getLastKnownLocation(provider);  
 **if** (location != **null**) {  
 LatLng loc = **new** LatLng(location.getLatitude(), location.getLongitude());  
 Log.*i*(**this**.toString(), loc + "");  
 *mMarker* = *mMap*.addMarker(**new** MarkerOptions().icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*shipper\_marker*)).position(loc));  
 **if** (*mMap* != **null**) {  
 *mMap*.animateCamera(CameraUpdateFactory.*newLatLngZoom*(loc, 16.0f));  
 }  
 }  
 }  
 **public void** sendRequest(String OriginTu, String OriginDen) {  
 String origin = OriginTu;  
 String destination = OriginDen;  
 **if** (origin.isEmpty()) {  
 //Toast.makeText(this, "Please enter origin address!", Toast.LENGTH\_SHORT).show();  
 **return**;  
 }  
 **if** (destination.isEmpty()) {  
 // Toast.makeText(this, "Please enter destination address!", Toast.LENGTH\_SHORT).show();  
 **return**;  
 }  
  
 **try** {  
 **new** DirectionFinder(TrackingActivity.**this**, origin, destination).execute();  
 } **catch** (UnsupportedEncodingException e) {  
 e.printStackTrace();  
 }  
 }  
 @Override  
 **public void** onLocationChanged(Location location) {  
 LatLng loc = **new** LatLng(location.getLatitude(), location.getLongitude());  
 String diemtu = loc.latitude + "," + loc.longitude;  
 String diemden = "" ;  
 **if** (*mMarker* != **null**){  
 *mMarker*.remove();  
 }  
  
 **if** (*mMap* != **null**) {  
 *mMap*.animateCamera(CameraUpdateFactory.*newLatLngZoom*(loc, 16.0f));  
 }  
  
  
  
  
 }  
  
  
 @Override  
 **public void** onProviderDisabled(String provider) {  
 Log.*d*("Latitude", "disable");  
 }  
  
 @Override  
 **public void** onProviderEnabled(String provider) {  
 Log.*d*("Latitude", "enable");  
 }  
  
 @Override  
 **public void** onStatusChanged(String provider, **int** status, Bundle extras) {  
 Log.*d*("Latitude", "status");  
  
 }  
 **public void** dialogTracking(){  
 Dialog dia = **new** Dialog(TrackingActivity.**this**);  
 dia.getWindow().setBackgroundDrawable(**new** ColorDrawable(android.graphics.Color.*TRANSPARENT*));  
 dia.requestWindowFeature(Window.*FEATURE\_NO\_TITLE*);  
 dia.setContentView(R.layout.*dialog\_tracking*);  
 dia.show();  
 }  
  
 **public static void** nhanduLieu(Double lat, Double lng) {  
 *mMap*.clear();  
 LatLng loc = **new** LatLng(lat,lng);  
 *mMarker* = *mMap*.addMarker(**new** MarkerOptions().icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*shipper\_marker*)).position(loc));  
 **if** (*mMap* != **null**) {  
 *mMap*.animateCamera(CameraUpdateFactory.*newLatLngZoom*(loc, 16.0f));  
 }  
 *toadoShiper* = lat + "," + lng;  
  
 }  
 **public static void** getToaDoDen(LatLng toado) {  
  
 *mMarker* = *mMap*.addMarker(**new** MarkerOptions().title("Điểm Giao Hàng").position(toado));  
  
 }  
 **public void** timer(){  
 **final** Timer timer = **new** Timer();  
 timer.schedule(**new** TimerTask() {  
 @Override  
 **public void** run() {  
 **if**(*trangthai* != "2") {  
 **if** (*toadoden*.length() > 3 && *toadoShiper*.length() > 3) {  
 sendRequest(*toadoden*, *toadoShiper*);  
 }  
 }**else**{  
  
 TrackingActivity.**this**.runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 dialogTracking();  
 timer.cancel();  
 }  
 });  
 }  
 }  
 }, 3000,3000);  
  
 }  
 @Override  
 **public boolean** onMarkerClick(Marker marker) {  
 **return false**;  
 }  
  
 @Override  
 **public void** onDirectionFinderStart() {  
  
 **if** (originMarkers != **null**) {  
 **for** (Marker marker : originMarkers) {  
 marker.remove();  
 }  
 }  
  
 **if** (destinationMarkers != **null**) {  
 **for** (Marker marker : destinationMarkers) {  
 marker.remove();  
 }  
 }  
  
 **if** (polylinePaths != **null**) {  
 **for** (Polyline polyline:polylinePaths ) {  
 polyline.remove();  
 }  
 }  
 }  
  
 @Override  
 **public void** onDirectionFinderSuccess(List<route> routes) {  
 String ab[] = *toadoden*.split(",");  
 LatLng loc = **new** LatLng(Double.*parseDouble*(ab[0]),Double.*parseDouble*(ab[1]));  
 *mMarker* = *mMap*.addMarker(**new** MarkerOptions().icon(BitmapDescriptorFactory.*fromResource*(R.drawable.*marker\_gift*)).position(loc));  
 ln.setVisibility(View.*VISIBLE*);  
 destinationMarkers = **new** ArrayList<>();  
  
  
  
 **for** (route route : routes) {  
  
 txtThoiGian.setText("Thời gian dự kiến " + route.duration.text.replace("mins","Phút") );  
 txtKhoangCach.setText("Khoảng Cách: " + route.distance.text);  
 String a = route.distance.text.trim();  
 **int** tt = Integer.*parseInt*(*trangthai*);  
 **if**(tt == 0){  
 txtTinhTrang.setText("Tình Trạng: Chưa Nhận Hàng");  
 }**else if**(tt == 1){  
 txtTinhTrang.setText("Tình Trạng: Đã Nhận Hàng");  
 }**else if**(tt == 2){  
 txtTinhTrang.setText("Tình Trạng: Đã Hoàn Thành");  
 }  
 }  
  
 }  
}

**UpdateThongTinChuShop**

**package** com.poly.shipper.activity;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.view.WindowManager;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.RadioButton;  
**import** android.widget.RadioGroup;  
  
  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.main.BuildMessage;  
**import** com.poly.shipper.service.MyService;  
**import** com.wdullaer.materialdatetimepicker.date.DatePickerDialog;  
  
**import** org.json.JSONObject;  
  
**import** java.util.Calendar;  
  
**public class** UpdateThongTinChuShop **extends** AppCompatActivity **implements** DatePickerDialog.OnDateSetListener {  
  
 **private boolean** mAutoHighlight;  
 EditText thongTinEdNgaySinh;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_update\_thong\_tin\_chu\_shop*);  
 **this**.getWindow().setSoftInputMode(WindowManager.LayoutParams.*SOFT\_INPUT\_STATE\_ALWAYS\_HIDDEN*);  
 **final** EditText thongTinEdHoTen = (EditText)findViewById(R.id.*thongTinEdHoTen*);  
 thongTinEdNgaySinh = (EditText)findViewById(R.id.*thongTinEdNgaySinh*);  
 **final** EditText thongTinEdTenShop = (EditText)findViewById(R.id.*thongTinEdTenShop*);  
 **final** EditText thongTinEdSDT = (EditText)findViewById(R.id.*thongTinEdSDT*);  
 **final** EditText thongTinEdDiaChi = (EditText)findViewById(R.id.*thongTinEdDiaChi*);  
 **final** Button thongTinBtnNgaySinh = (Button)findViewById(R.id.*thongTinBtnNgaySinh*);  
 **final** Button thongTinBtnXacNhan = (Button)findViewById(R.id.*thongTinBtnXacNhan*);  
 **final** Button thongTinBtnHuy = (Button)findViewById(R.id.*thongTinBtnHuy*);  
 **final** RadioGroup thongTinCbGioiTinh = (RadioGroup)findViewById(R.id.*thongTinCbGioiTinh*);  
 **final** RadioButton thongTinCbNam = (RadioButton)findViewById(R.id.*thongTinCbNam*);  
 **final** RadioButton thongTinCbNu = (RadioButton)findViewById(R.id.*thongTinCbNu*);  
 thongTinBtnHuy.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent a = **new** Intent(UpdateThongTinChuShop.**this**,NavDrawerActivity.**class**);  
 startActivity(a);  
 }  
 });  
 thongTinBtnNgaySinh.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Calendar now = Calendar.*getInstance*();  
 DatePickerDialog dpd = DatePickerDialog.*newInstance*(  
 UpdateThongTinChuShop.**this**,  
 now.get(Calendar.*YEAR*),  
 now.get(Calendar.*MONTH*),  
 now.get(Calendar.*DAY\_OF\_MONTH*)  
 );  
 dpd.show(getFragmentManager(), "Datepickerdialog");  
 }  
 });  
  
 thongTinBtnXacNhan.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **try** {  
 BuildMessage xMessage = **new** BuildMessage();  
 JSONObject infoMessage = **new** JSONObject();  
 infoMessage.put("hoten", thongTinEdHoTen.getText().toString());  
 infoMessage.put("ngaysinh", thongTinEdNgaySinh.getText().toString());  
 infoMessage.put("tenshop", thongTinEdTenShop.getText().toString());  
 infoMessage.put("sdt", thongTinEdSDT.getText().toString());  
 infoMessage.put("diachi", thongTinEdDiaChi.getText().toString());  
 infoMessage.put("gioitinh", thongTinCbGioiTinh.getCheckedRadioButtonId() == R.id.*thongTinCbNu* ? 0 : 1);  
 String message = xMessage.buildMessage("UpdateThongTinChuShop", infoMessage);  
 System.*out*.println(message);  
 MyService.*client*.send(message);  
 }  
 **catch** (Exception ex){  
 ex.printStackTrace();  
 }  
 }  
 });  
 Intent i = getIntent();  
 **if**(i.getStringExtra("tenChuShop")==**null**){  
 thongTinBtnHuy.setVisibility(View.*GONE*);  
 }**else** {  
 thongTinBtnHuy.setVisibility(View.*VISIBLE*);  
 String tenChuShop = i.getStringExtra("tenChuShop");  
 thongTinEdHoTen.setText(tenChuShop);  
 String ngaySinh = i.getStringExtra("ngaySinh");  
 thongTinEdNgaySinh.setText(ngaySinh);  
 String gioiTinh = i.getStringExtra("gioiTinh");  
 **if**(gioiTinh.equals("Nam")){  
 thongTinCbNam.setChecked(**true**);  
 thongTinCbNu.setChecked(**false**);  
 }**else** {  
 thongTinCbNam.setChecked(**false**);  
 thongTinCbNu.setChecked(**true**);  
 }  
 String tenShop = i.getStringExtra("tenShop");  
 thongTinEdTenShop.setText(tenShop);  
 String diaChi = i.getStringExtra("diaChi");  
 thongTinEdDiaChi.setText(diaChi);  
 String soDienThoai = i.getStringExtra("soDienThoai");  
 thongTinEdSDT.setText(soDienThoai);  
 }  
 }  
  
  
 @Override  
 **public void** onDateSet(DatePickerDialog view, **int** year, **int** monthOfYear, **int** dayOfMonth) {  
 thongTinEdNgaySinh.setText(Integer.*toString*(dayOfMonth) + "/" + Integer.*toString*(monthOfYear + 1) + "/" + Integer.*toString*(year));  
 }  
}

### Hệ thống Respon :

**ResponDanhGiaShipper**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.widget.Toast;  
  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.ChiTietHoaDonActivity;  
**import** com.poly.shipper.activity.QuanLyHoaDonChuShopActivity;  
**import** com.poly.shipper.activity.UpdateThongTinChuShop;  
**import** com.poly.shipper.model.TinyDB;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
**import** com.poly.shipper.utils.utils;  
  
**import** org.json.JSONObject;  
  
/\*\*  
 \* Created by ADMIN on 9/21/2016.  
 \*/  
**public class** ResponDanhGiaShipper **extends** WebSocketServlet{  
  
 @Override  
 **public void** doAction(Context context) {  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
 **try** {  
 **boolean** result = obj.getBoolean("InsertDanhGiaShipperResult");  
 **if** (result) {  
 **if** (context != **null** && context **instanceof** Activity) { // chi context la activity moi co the startActivity  
 ChiTietHoaDonActivity.*danhGiaHoanTat*();  
 }  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**ResponHoanThanhDonHang**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.content.Intent;  
  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
**import** com.poly.shipper.service.HeadService;  
  
**import** org.json.JSONObject;  
  
/\*\*  
 \* Created by ADMIN on 9/21/2016.  
 \*/  
**public class** ResponHoanThanhDonHang **extends** WebSocketServlet{  
  
 @Override  
 **public void** doAction(Context context) {  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
 **try** {  
 **boolean** result = obj.getBoolean("HoanThanhDonHangResult");  
 String idHoaDon = Integer.*toString*(obj.getInt("idHoaDon"));  
 Intent i = **new** Intent(context, HeadService.**class**);  
 i.putExtra("idHoaDon",idHoaDon);  
 context.startService(i);  
 **if** (result) {  
 **if** (context != **null** && context **instanceof** Activity) { // chi context la activity moi co the startActivity  
  
 }  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
  
  
 }  
}

**ResponLoadInformation**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.NavDrawerActivity;  
**import** com.poly.shipper.activity.QuanLyHoaDonChuShopActivity;  
**import** com.poly.shipper.model.ChuShop;  
**import** com.poly.shipper.model.HoaDon;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
  
**import** org.json.JSONArray;  
**import** org.json.JSONObject;  
  
**import** java.util.ArrayList;  
  
/\*\*  
 \* Created by quocdat on 11/7/2016.  
 \*/  
  
**public class** ResponLoadInformation **extends** WebSocketServlet {  
 ChuShop chuShop;  
 JSONObject xobj;  
 **public void** doAction(**final** Context context) {  
 chuShop = **new** ChuShop();  
 xobj = **new** JSONObject();  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
  
 **try** {  
 xobj = obj.getJSONObject("info");  
 chuShop.setIdChuShop(xobj.getString("idChuShop"));  
 chuShop.setTenChuShop(xobj.getString("tenChuShop"));  
 chuShop.setGioiTinh(xobj.getBoolean("gioiTinh"));  
 chuShop.setNgaySinh(xobj.getString("ngaySinh"));  
 chuShop.setTenShop(xobj.getString("tenShop"));  
 chuShop.setDiaChi(xobj.getString("diaChi"));  
 chuShop.setSoDienThoai(xobj.getString("soDienThoai"));  
 **if** (context != **null** && context **instanceof** Activity){  
 ((Activity) context).runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 System.*out*.println("nav ok");  
 NavDrawerActivity.*addInfoToNav*(chuShop);  
 }  
 });  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**ResponLocation**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.app.Dialog;  
**import** android.content.Context;  
**import** android.graphics.drawable.ColorDrawable;  
**import** android.util.Log;  
**import** android.view.Window;  
**import** android.widget.TextView;  
  
**import** com.google.android.gms.maps.model.LatLng;  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.TrackingActivity;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
  
**import** org.json.JSONObject;  
  
/\*\*  
 \* Created by intel on 11/4/16.  
 \*/  
  
**public class** ResponLocation **extends** WebSocketServlet {  
 LocationListener listener;  
  
  
 @Override  
 **public void** doAction(Context context) {  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
 **try** {  
 **final** Double lat = Double.*parseDouble*(obj.getString("lat"));  
 **final** Double lng = Double.*parseDouble*(obj.getString("lon"));  
 Log.*i*("áddd",obj.toString());  
 //listener.nhanduLieu(lat,lng);  
 **if** (context != **null** && context **instanceof** Activity) { // chi context la activity moi co the startActivity  
  
 ((Activity) context).runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 TrackingActivity.*nhanduLieu*(lat, lng);  
 }  
 });  
  
 }  
  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**ResponLogin**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.content.Intent;  
  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.MainActivity;  
**import** com.poly.shipper.activity.NavDrawerActivity;  
**import** com.poly.shipper.activity.QuanLyHoaDonChuShopActivity;  
**import** com.poly.shipper.activity.UpdateThongTinChuShop;  
**import** com.poly.shipper.model.TinyDB;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
**import** com.poly.shipper.utils.utils;  
  
**import** org.json.JSONObject;  
  
/\*\*  
 \* Created by ADMIN on 9/21/2016.  
 \*/  
**public class** ResponLogin **extends** WebSocketServlet{  
  
 @Override  
 **public void** doAction(Context context) {  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
 **try** {  
 utils.*alreadyLogin* = **true**;  
 **boolean** loginOK = obj.getBoolean("loginOK");  
 System.*out*.print(loginOK);  
 String idTaiKhoan = obj.getString("idTaiKhoan");  
 TinyDB tinyDB = **new** TinyDB(context);  
 tinyDB.putString("idTaiKhoan",idTaiKhoan);  
 **if** (context != **null** && context **instanceof** Activity) { // chi context la activity moi co the startActivity  
 **if** (loginOK) {  
 context.startActivity(**new** Intent(context, NavDrawerActivity.**class**));  
 } **else** {  
 context.startActivity(**new** Intent(context, UpdateThongTinChuShop.**class**));  
 }  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**ResponQuanLyHoaDonChuShop**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.QuanLyHoaDonChuShopActivity;  
**import** com.poly.shipper.model.HoaDon;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
  
**import** org.json.JSONArray;  
**import** org.json.JSONObject;  
  
**import** java.util.ArrayList;  
  
  
/\*\*  
 \* Created by quocdat on 10/31/2016.  
 \*/  
  
**public class** ResponQuanLyHoaDonChuShop **extends** WebSocketServlet {  
  
 JSONArray quanLyHoaDonChuShopResult;  
 HoaDon hoaDon;  
 ArrayList<HoaDon> listHd;  
  
 @Override  
 **public void** doAction(**final** Context context) {  
 listHd = **new** ArrayList<>();  
 quanLyHoaDonChuShopResult = **new** JSONArray();  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
 **try** {  
 quanLyHoaDonChuShopResult = obj.getJSONArray("listHoaDon");  
 **for** (**int** i = 0;i<quanLyHoaDonChuShopResult.length();i++){  
 JSONObject xobj = quanLyHoaDonChuShopResult.getJSONObject(i);  
 hoaDon = **new** HoaDon();  
 hoaDon.setIdHoaDon(xobj.getInt("idHoaDon"));  
 hoaDon.setHinhAnh(xobj.getString("hinhAnh"));  
 hoaDon.setTenHang(xobj.getString("tenHang"));  
 hoaDon.setTu(xobj.getString("tu"));  
 hoaDon.setToaDoTu(xobj.getString("toaDoTu"));  
 hoaDon.setDen(xobj.getString("den"));  
 hoaDon.setToaDoDen(xobj.getString("toaDoDen"));  
 hoaDon.setKhoangCach(xobj.getDouble("khoangCach"));  
 hoaDon.setTienHang(xobj.getInt("giaTienHang"));  
 hoaDon.setTienShip(xobj.getInt("giaTienShip"));  
 hoaDon.setNguoiTra(xobj.getBoolean("nguoiTra"));  
 hoaDon.setGhiChu(xobj.getString("ghiChu"));  
 hoaDon.setIdTinhTrang(xobj.getInt("idTinhTrang"));  
 hoaDon.setIdShipper(xobj.getString("idShipper"));  
 hoaDon.setIdchushop(xobj.getString("idChuShop"));  
 hoaDon.setTenNguoiGui(xobj.getString("tenNguoiGui"));  
 hoaDon.setSdtNguoiGui(xobj.getString("sdtNguoiGui"));  
 hoaDon.setTenNguoiNhan(xobj.getString("tenNguoiNhan"));  
 hoaDon.setSdtNguoiNhan(xobj.getString("sdtNguoiNhan"));  
 listHd.add(hoaDon);  
 }  
 **if** (context != **null** && context **instanceof** Activity){  
 ((Activity) context).runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 QuanLyHoaDonChuShopActivity.*setAdapterToActivity*(context,listHd);  
 }  
 });  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
}

**ResponTaoHoaDon**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.app.Dialog;  
**import** android.content.Context;  
**import** android.graphics.drawable.ColorDrawable;  
**import** android.view.View;  
**import** android.view.Window;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
  
**import** com.poly.shipper.R;  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.TaoHoaDonActivity;  
**import** com.poly.shipper.asynctask.upload;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
  
**import** org.json.JSONObject;  
  
  
  
/\*\*  
 \* Created by quocdat on 10/31/2016.  
 \*/  
  
**public class** ResponTaoHoaDon **extends** WebSocketServlet {  
 TextView a,b;  
 Button c;  
 Dialog dialog;  
 @Override  
 **public void** doAction(**final** Context context) {  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
 **try** {  
 **final** String ma = obj.getString("maTracking");  
  
 **if** (ma!=**null**) {  
 **if** (context != **null** && context **instanceof** Activity) { // chi context la activity moi co the startActivity  
 TaoHoaDonActivity.*UpLoadImage*(context,ma);  
 ((Activity) context).runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 dialog = **new** Dialog(context);  
 dialog.getWindow().setBackgroundDrawable(**new** ColorDrawable(android.graphics.Color.*TRANSPARENT*));  
 dialog.requestWindowFeature(Window.*FEATURE\_NO\_TITLE*);  
 dialog.setContentView(R.layout.*popupmessagelockscreen*);  
 a = (TextView)dialog.findViewById(R.id.*dialogMess*);  
 b = (TextView)dialog.findViewById(R.id.*dialogAlert*);  
 c = (Button)dialog.findViewById(R.id.*ok*);  
 a.setText("Mã Tracking");  
 b.setText( ma+" \n Hãy đưa mã này cho người bạn cần ship hàng đến!");  
 dialog.show();  
 c.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 dialog.dismiss();  
 ((Activity) context).onBackPressed();  
 }  
 });  
 }  
 });  
 }  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**ResponUpdateInfoChuShop**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.content.Intent;  
  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.NavDrawerActivity;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
  
**import** org.json.JSONObject;  
  
/\*\*  
 \* Created by ADMIN on 9/21/2016.  
 \*/  
**public class** ResponUpdateInfoChuShop **extends** WebSocketServlet{  
  
 @Override  
 **public void** doAction(Context context) {  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
 **try** {  
 **boolean** updateThongTinChuShopResult = obj.getBoolean("UpdateThongTinChuShopResult");  
 **if** (updateThongTinChuShopResult) {  
 **if** (context != **null** && context **instanceof** Activity) { // chi context la activity moi co the startActivity  
 context.startActivity(**new** Intent(context, NavDrawerActivity.**class**));  
 }  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**SuccessSetTracking**

**package** com.poly.shipper.responservlet;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.util.Log;  
  
**import** com.google.android.gms.maps.model.LatLng;  
**import** com.poly.shipper.WebSocketClient.WebSocketClient;  
**import** com.poly.shipper.activity.TrackingActivity;  
**import** com.poly.shipper.model.WebSocketServlet;  
**import** com.poly.shipper.model.WrapWebSocket;  
  
**import** org.json.JSONObject;  
  
/\*\*  
 \* Created by intel on 11/4/16.  
 \*/  
  
**public class** SuccessSetTracking **extends** WebSocketServlet {  
  
 @Override  
 **public void** doAction(**final** Context context) {  
 WrapWebSocket wrapWebSocket = getWrapWebSocket();  
 WebSocketClient client = wrapWebSocket.getWebSocketClient();  
 JSONObject obj = wrapWebSocket.getInfoMessage();  
  
 **try** {  
 JSONObject hoadon = obj.getJSONObject("hoadon");  
 **final** String toadoden = hoadon.getString("toaDoDen");  
 **final** String tinhtrang = hoadon.getString("idTinhTrang");  
 String p[] = toadoden.split(",");  
 TrackingActivity.*toadoden* = p[0] + "," + p[1];  
 //final int statusTracking = hoadon.getInt("statusTracking"); // 0: gui thong bao ko co hoadon, 1: ko tim thay shipper, 2: gui thong bao hoa don da hoan thanh, ko the tracking nua, 3: da set tracking  
 **if** (context != **null** && context **instanceof** Activity) { // chi context la activity moi co the startActivity  
  
 ((Activity) context).runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 // Nho truong viet cai dialog hien thi trang thai statusTracking khi statusTracking = 0, 1, 2;  
 ((Activity) context).runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 TrackingActivity.*toadoden* = toadoden;  
 TrackingActivity.*trangthai* = tinhtrang;  
 String a[] = toadoden.split(",");  
 LatLng loc = **new** LatLng(Double.*parseDouble*(a[0]),Double.*parseDouble*(a[1]));  
  
 }  
 });  
 }  
 });  
  
 }  
  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**Servler**

<AllServlet>  
 <Servlet>  
 <Name>Login</Name>  
 <ClassName>com.poly.shipper.responservlet.ResponLogin</ClassName>  
 <Action>Login</Action>  
 <Description>Xử lý login</Description>  
 </Servlet>  
 <Servlet>  
 <Name>Update Thong Tin Chu Shop</Name>  
 <ClassName>com.poly.shipper.responservlet.ResponUpdateInfoChuShop</ClassName>  
 <Action>UpdateThongTinChuShop</Action>  
 <Description>Xử lý update thong tin chu shop</Description>  
 </Servlet>  
 <Servlet>  
 <Name>Tao hoa don</Name>  
 <ClassName>com.poly.shipper.responservlet.ResponTaoHoaDon</ClassName>  
 <Action>TaoHoaDon</Action>  
 <Description>Tao hóa đơn</Description>  
 </Servlet>  
 <Servlet>  
 <Name>Load list Hoa Don Chu Shop</Name>  
 <ClassName>com.poly.shipper.responservlet.ResponQuanLyHoaDonChuShop</ClassName>  
 <Action>QuanLyHoaDonChuShop</Action>  
 <Description>Tải về danh sách hóa đơn chủ shop</Description>  
 </Servlet>  
 <Servlet>  
 <Name>nhan du lieu tra ve</Name>  
 <ClassName>com.poly.shipper.responservlet.ResponLocation</ClassName>  
 <Action>LayLocation</Action>  
 <Description>Tải về danh sách hóa đơn</Description>  
 </Servlet>  
 <Servlet>  
 <Name>SuccessSetTracking</Name>  
 <ClassName>com.poly.shipper.responservlet.SuccessSetTracking</ClassName>  
 <Action>SuccessSetTracking</Action>  
 <Description>Khi set ma tracking xong thi server se tra ve toadoden cua hoadon va statusTracking</Description>  
 </Servlet>  
 <Servlet>  
 <Name>Load information</Name>  
 <ClassName>com.poly.shipper.responservlet.ResponLoadInformation</ClassName>  
 <Action>LoadInformation</Action>  
 <Description>Trả về thông tin tài khoản khi đăng nhập thành công</Description>  
 </Servlet>  
 <Servlet>  
 <Name>Hoan Thanh Don Hang</Name>  
 <ClassName>com.poly.shipper.responservlet.ResponHoanThanhDonHang</ClassName>  
 <Action>HoanThanhDonHang</Action>  
 <Description>Hoan Thanh Don Hang</Description>  
 </Servlet>  
</AllServlet>

### Hệ Thông Sever:

**DatabaseManager**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package com.poly.shipper.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import org.json.JSONArray;

import org.json.JSONObject;

/\*\*

\*

\* @author PCgicungdchttp://163.44.150.34/

\*/

public class DatabaseManager extends MySQLHelper{

static final String DB\_URL = "jdbc:mysql://163.44.150.34/quanlyshipper?useUnicode=true&characterEncoding=UTF-8";

// Database credentials

static final String USER = "root";

static final String PASS = "123456";

public DatabaseManager(String DB\_URL1, String USER1, String PASS1) {

super(DB\_URL1, USER1, PASS1);

}

public DatabaseManager()

{

super(DB\_URL, USER, PASS);

}

public boolean checkLoginChuShop(String idTaiKhoan){

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "SELECT \* FROM chushop WHERE idChuShop = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, idTaiKhoan); // idChuShop

ResultSet rs = preparedStatement.executeQuery( );

while (rs.next()) {

result = true;

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public boolean checkLoginShipper(String idTaiKhoan){

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "SELECT \* FROM shipper WHERE idShipper = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, idTaiKhoan); // idShipper

ResultSet rs = preparedStatement.executeQuery( );

while (rs.next()) {

result = true;

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public boolean deleteHoaDon(int idHoaDon){

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "DELETE FROM hoadon WHERE idHoaDon = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setInt(1, idHoaDon); // id hóa đơn

preparedStatement.executeUpdate();

result = true;

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public JSONObject getInfoShipper(String idTaiKhoan) {

JSONObject obj = new JSONObject();

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "SELECT \* FROM shipper WHERE idShipper = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, idTaiKhoan); // id from userOnline

ResultSet rs = preparedStatement.executeQuery( );

while (rs.next()) {

obj = new JSONObject();

obj.put("idShipper", rs.getString("idShipper"));

obj.put("tenShipper", rs.getString("tenShipper"));

obj.put("ngaySinh", rs.getString("ngaySinh"));

obj.put("gioiTinh", rs.getBoolean("gioiTinh"));

obj.put("ngayVaoLam", rs.getString("ngayVaoLam"));

obj.put("diaChi", rs.getString("diaChi"));

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return obj;

}

public JSONObject getInfoChuShop(String idTaiKhoan) {

JSONObject obj = new JSONObject();

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "SELECT \* FROM chushop WHERE idChuShop = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, idTaiKhoan); // id from userOnline

ResultSet rs = preparedStatement.executeQuery( );

while (rs.next()) {

obj = new JSONObject();

obj.put("idChuShop", rs.getString("idChuShop"));

obj.put("tenChuShop", rs.getString("tenChuShop"));

obj.put("gioiTinh", rs.getBoolean("gioiTinh"));

obj.put("ngaySinh", rs.getString("ngaySinh"));

obj.put("tenShop", rs.getString("tenShop"));

obj.put("diaChi", rs.getString("diaChi"));

obj.put("soDienThoai", rs.getString("soDienThoai"));

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return obj;

}

public boolean insertChuShop(String idTaiKhoan) {

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "INSERT INTO chushop (idChuShop,tenChuShop,gioiTinh,ngaySinh,tenShop,diaChi,soDienThoai) VALUES (?,?,?,?,?,?,?)";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, idTaiKhoan); // idChuShop

preparedStatement.setString(2, ""); // tenChuShop

preparedStatement.setBoolean(3, false); // gioiTinh

preparedStatement.setString(4, "2013-09-04"); // ngaySinh

preparedStatement.setString(5, ""); // tenShop

preparedStatement.setString(6, ""); // diaChi

preparedStatement.setString(7, ""); // soDienThoai

preparedStatement .executeUpdate();

result = true;

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public boolean updateInfoChuShop(String idTaiKhoan,String hoTen, String ngaySinh, String tenShop, String sdt, String diaChi, int gioiTinh) {

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "update chushop set tenChuShop = ?, gioiTinh = ?, ngaySinh = ?, tenShop = ?, diaChi = ?, soDienThoai = ? where idChuShop = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, hoTen);

preparedStatement.setInt(2, gioiTinh);

preparedStatement.setString(3, ngaySinh);

preparedStatement.setString(4, tenShop);

preparedStatement.setString(5, diaChi);

preparedStatement.setString(6, sdt);

preparedStatement.setString(7, idTaiKhoan);

preparedStatement.executeUpdate();

result = true;

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public int insertHoaDon(String hinhAnh, String tenHang, String tenNguoiGui,String sdtNguoiGui,

String tu, String toaDoTu, String den, String toaDoDen,

Double khoangCach, int giaTienHang, int giaTienShip, boolean nguoiTra,

String ghiChu, String tenNguoiNhan, String sdtNguoiNhan, int tinhTrang, String idChuShop,

String idShipper) {

int result = -1;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "INSERT INTO hoadon (hinhAnh, tenHang, tenNguoiGui, sdtNguoiGui, tu, toaDoTu, den, toaDoDen, khoangCach, giaTienHang, giaTienShip, nguoiTra, ghiChu, tenNguoiNhan, sdtNguoiNhan, idTinhTrang, idChuShop, idShipper) VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, hinhAnh);

preparedStatement.setString(2, tenHang);

preparedStatement.setString(3, tenNguoiGui);

preparedStatement.setString(4, sdtNguoiGui);

preparedStatement.setString(5, tu);

preparedStatement.setString(6, toaDoTu);

preparedStatement.setString(7, den);

preparedStatement.setString(8, toaDoDen);

preparedStatement.setDouble(9, khoangCach);

preparedStatement.setInt(10, giaTienHang);

preparedStatement.setInt(11, giaTienShip);

preparedStatement.setBoolean(12, nguoiTra);

preparedStatement.setString(13, ghiChu);

preparedStatement.setString(14, tenNguoiNhan);

preparedStatement.setString(15, sdtNguoiNhan);

preparedStatement.setInt(16, tinhTrang);

preparedStatement.setString(17, idChuShop);

preparedStatement.setString(18, idShipper);

int affectedRows = preparedStatement.executeUpdate();

if (affectedRows == 0) {

result = -1;

}

else {

ResultSet generatedKeys = preparedStatement.getGeneratedKeys();

if (generatedKeys.next()) {

result = generatedKeys.getInt(1);

}

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public JSONArray getListHoaDonShipper(String userOnline) {

JSONArray arrObj = new JSONArray();

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "SELECT \* FROM hoadon WHERE idShipper = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, userOnline); // idChuShipper

ResultSet rs = preparedStatement.executeQuery( );

while (rs.next()) {

JSONObject obj = new JSONObject();

obj.put("idHoaDon", rs.getInt("idHoaDon"));

obj.put("hinhAnh", rs.getString("hinhAnh"));

obj.put("tenHang", rs.getString("tenHang"));

obj.put("tu", rs.getString("tu"));

obj.put("toaDoTu", rs.getString("toaDoTu"));

obj.put("den", rs.getString("den"));

obj.put("toaDoDen", rs.getString("toaDoDen"));

obj.put("khoangCach", rs.getDouble("khoangCach"));

obj.put("giaTienHang", rs.getInt("giaTienHang"));

obj.put("giaTienShip", rs.getInt("giaTienShip"));

obj.put("nguoiTra", rs.getBoolean("nguoiTra"));

obj.put("ghiChu", rs.getString("ghiChu"));

obj.put("idTinhTrang", rs.getInt("idTinhTrang"));

obj.put("idShipper", rs.getString("idShipper"));

obj.put("idChuShop", rs.getString("idChuShop"));

obj.put("tenNguoiGui", rs.getString("tenNguoiGui"));

obj.put("sdtNguoiGui", rs.getString("sdtNguoiGui"));

obj.put("tenNguoiNhan", rs.getString("tenNguoiNhan"));

obj.put("sdtNguoiNhan", rs.getString("sdtNguoiNhan"));

arrObj.put(obj);

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return arrObj;

}

public JSONArray getListHoaDon(String userOnline) {

JSONArray arrObj = new JSONArray();

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "SELECT \* FROM hoadon WHERE idChuShop = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, userOnline); // idChuShop

ResultSet rs = preparedStatement.executeQuery( );

while (rs.next()) {

JSONObject obj = new JSONObject();

obj.put("idHoaDon", rs.getInt("idHoaDon"));

obj.put("hinhAnh", rs.getString("hinhAnh"));

obj.put("tenHang", rs.getString("tenHang"));

obj.put("tu", rs.getString("tu"));

obj.put("toaDoTu", rs.getString("toaDoTu"));

obj.put("den", rs.getString("den"));

obj.put("toaDoDen", rs.getString("toaDoDen"));

obj.put("khoangCach", rs.getDouble("khoangCach"));

obj.put("giaTienHang", rs.getInt("giaTienHang"));

obj.put("giaTienShip", rs.getInt("giaTienShip"));

obj.put("nguoiTra", rs.getBoolean("nguoiTra"));

obj.put("ghiChu", rs.getString("ghiChu"));

obj.put("idTinhTrang", rs.getInt("idTinhTrang"));

obj.put("idShipper", rs.getString("idShipper"));

obj.put("idChuShop", rs.getString("idChuShop"));

obj.put("tenNguoiGui", rs.getString("tenNguoiGui"));

obj.put("sdtNguoiGui", rs.getString("sdtNguoiGui"));

obj.put("tenNguoiNhan", rs.getString("tenNguoiNhan"));

obj.put("sdtNguoiNhan", rs.getString("sdtNguoiNhan"));

arrObj.put(obj);

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return arrObj;

}

public boolean insertDanhGia(String lyDo,int rate,int idHoaDon) {

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "INSERT INTO danhgia (lyDo,rate,idHoaDon) VALUES (?,?,?)";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1, lyDo);

preparedStatement.setInt(2, rate);

preparedStatement.setInt(3, idHoaDon);

//System.out.println("cai long");

preparedStatement.executeUpdate();

result = true;

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public JSONObject getHoaDon(int idHoaDon) {

Connection dbConnection = null;

JSONObject obj = null;

PreparedStatement preparedStatement = null;

String sql = "SELECT \* FROM hoadon WHERE idHoaDon = ?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setInt(1, idHoaDon); // idHoaDon

ResultSet rs = preparedStatement.executeQuery( );

while (rs.next()) {

obj = new JSONObject();

obj.put("idHoaDon", rs.getInt("idHoaDon"));

obj.put("hinhAnh", rs.getString("hinhAnh"));

obj.put("tenHang", rs.getString("tenHang"));

obj.put("tu", rs.getString("tu"));

obj.put("toaDoTu", rs.getString("toaDoTu"));

obj.put("den", rs.getString("den"));

obj.put("toaDoDen", rs.getString("toaDoDen"));

obj.put("khoangCach", rs.getDouble("khoangCach"));

obj.put("giaTienHang", rs.getInt("giaTienHang"));

obj.put("giaTienShip", rs.getInt("giaTienShip"));

obj.put("nguoiTra", rs.getBoolean("nguoiTra"));

obj.put("ghiChu", rs.getString("ghiChu"));

obj.put("idTinhTrang", rs.getInt("idTinhTrang"));

obj.put("idShipper", rs.getString("idShipper"));

obj.put("idChuShop", rs.getString("idChuShop"));

obj.put("tenNguoiGui", rs.getString("tenNguoiGui"));

obj.put("sdtNguoiGui", rs.getString("sdtNguoiGui"));

obj.put("tenNguoiNhan", rs.getString("tenNguoiNhan"));

obj.put("sdtNguoiNhan", rs.getString("sdtNguoiNhan"));

}

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return obj;

}

public boolean hoanthanhdonhang(int maHoaDon) {

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "UPDATE hoadon set idTinhTrang = ? WHERE idHoaDon =?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setInt(1, 2);

preparedStatement.setInt(2, maHoaDon);

preparedStatement.executeUpdate();

result = true;

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public boolean nhanhang(int nhanhang) {

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "UPDATE hoadon set idTinhTrang = ? WHERE idHoaDon =?";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setInt(1, 1);

preparedStatement.setInt(2, nhanhang);

preparedStatement.executeUpdate();

result = true;

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

public boolean insertSOS(String ghiChu,int idLydoSOS,String IdShipper) {

boolean result = false;

Connection dbConnection = null;

PreparedStatement preparedStatement = null;

String sql = "INSERT INTO sos (ghiChu,idLydoSOS,IdShipper) VALUES (?,?,?)";

try

{

dbConnection = getConnection();

preparedStatement = dbConnection.prepareStatement(sql);

preparedStatement.setString(1,ghiChu);

preparedStatement.setInt(2,idLydoSOS);

preparedStatement.setString(3,IdShipper);

preparedStatement.executeUpdate();

result = true;

}

catch(Exception e){

e.printStackTrace();

}finally {

try {

if (preparedStatement != null) {

preparedStatement.close();

}

if (dbConnection != null) {

dbConnection.close();

}

} catch (Exception e) {

e.printStackTrace();

}

}

return result;

}

}

**Main**

package com.poly.shipper.main;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import org.eclipse.jetty.server.Server;

import org.eclipse.jetty.servlet.ServletContextHandler;

import org.eclipse.jetty.websocket.server.WebSocketHandler;

import org.eclipse.jetty.websocket.servlet.WebSocketServletFactory;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.utils.utils;

public class Main {

public static void main(String[] args) throws Exception{

utils.userManager = new UserManager();

DatabaseManager db = new DatabaseManager();

Server server = new Server(6702);

WebSocketHandler wsHandler = new WebSocketHandler() {

@Override

public void configure(WebSocketServletFactory factory) {

//factory.getPolicy().setIdleTimeout(1800000);

factory.getPolicy().setIdleTimeout(600000);

factory.register(WebSocketClientHandler.class);

}

};

server.setHandler(wsHandler);

server.start();

server.join();

}

}

**DanhGiaShipper**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class DanhGiaShipper extends WebSocketServlet {

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

boolean InsertDanhGiaShipperResult = true;

String lyDo = obj.getString("lyDo");

int rate = obj.getInt("rate");

String idChuShop = obj.getString("idChuShop");

String idShipper = obj.getString("idShipper");

//DatabaseManager database = new DatabaseManager();

//if (database.insertDanhGia(lyDo, rate, idChuShop)) {

// InsertDanhGiaShipperResult = true;

//}

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("InsertDanhGiaShipperResult", InsertDanhGiaShipperResult);

infoMessage.put("danhGia", rate);

infoMessage.put("idChuShop", idChuShop);

infoMessage.put("idShipper", idShipper);

String result = xMessage.buildMessage("DanhGiaShipper", infoMessage);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**DoLogin**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.dao.MySQLHelper;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

// Xử lý login

public class DoLogin extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

UserManager userManager = getUserManager();

String idTaiKhoan = obj.getString("idTaiKhoan");

String Type = obj.getString("type"); // type login: ChuShop, Shipper

DatabaseManager database = new DatabaseManager();

boolean resultLogin = false;

if ("ChuShop".equals(Type)) {

resultLogin = database.checkLoginChuShop(idTaiKhoan);

if (!resultLogin) {

database.insertChuShop(idTaiKhoan);

}

}

else if ("Shipper".equals(Type)) {

resultLogin = database.checkLoginShipper(idTaiKhoan);

if (!resultLogin) {

//database.insertShipper(idTaiKhoan);

}

}

UserOnline userOnline = null;

if (resultLogin && "Shipper".equals(Type)){ // chỉ khi nào shipper login thành công thì mới lưu vào userManager

userOnline = new UserOnline(session, idTaiKhoan);

System.out.println("zo Shipper");

}

else { // còn trường hợp chushop login thành công hay không thành công thì cũng đều phải lưu vào userManager vì login thành công hay ko thành công thì client cũng sẽ vào giao diện mới (Update tài khoản hoặc map )

userOnline = new UserOnline(session, idTaiKhoan);

}

if (userOnline != null) {

userOnline.setType(Type);

userManager.AddUser(userOnline);

}

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("loginOK", resultLogin);

infoMessage.put("idTaiKhoan", idTaiKhoan);

String result = xMessage.buildMessage("Login", infoMessage);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**HoanThanhDonHang**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class HoanThanhDonHang extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

//UserOnline userOnline = userManager.findUserOnline(session);

boolean HoanThanhDonHangResult = false;

int maTracking = obj.getInt("maTracking");

String idTaiKhoan = obj.getString("idChuShop");

UserOnline userOnlineChuShop = userManager.findChuShop(idTaiKhoan);

Session sessionChuShop = null;

if (userOnlineChuShop != null) {

sessionChuShop = userOnlineChuShop.session;

}

// System.out.println(maTracking);

DatabaseManager database = new DatabaseManager();

if (database.hoanthanhdonhang(maTracking)) {

HoanThanhDonHangResult = true;

}

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("HoanThanhDonHangResult", HoanThanhDonHangResult);

infoMessage.put("idHoaDon", maTracking);

String result = xMessage.buildMessage("HoanThanhDonHang", infoMessage);

if (sessionChuShop != null) {

try {

sessionChuShop.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

try {

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**LoadInformation**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONArray;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class LoadInformation extends WebSocketServlet {

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

boolean LoadInformationResult = false;

boolean load = obj.getBoolean("loadInfo");

DatabaseManager database = new DatabaseManager();

JSONObject xobj = new JSONObject();

if("ChuShop".equals(userOnline.type)){

xobj = database.getInfoChuShop(userOnline.idTaiKhoan);

}else {

xobj = database.getInfoShipper(userOnline.idTaiKhoan);

}

JSONObject sobj = new JSONObject();

sobj.put("info", xobj);

try {

BuildMessage xMessage = new BuildMessage();

String result = xMessage.buildMessage("LoadInformation", sobj);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**NhanHang**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class NhanHang extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

//UserManager userManager = getUserManager();

//UserOnline userOnline = userManager.findUserOnline(session);

boolean HoanThanhDonHangResult = false;

int nhanhang = obj.getInt("idhoadon");

// System.out.println(maTracking);

DatabaseManager database = new DatabaseManager();

if (database.nhanhang(nhanhang)) {

HoanThanhDonHangResult = true;

}

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("NhanHangResult", HoanThanhDonHangResult);

String result = xMessage.buildMessage("NhanHang", infoMessage);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**QuanLyHoaDonChuShop**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONArray;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class QuanLyHoaDonChuShop extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

boolean QuanLyHoaDonChuShopResult = false;

boolean load = obj.getBoolean("loadList");

DatabaseManager database = new DatabaseManager();

JSONArray arrObj = database.getListHoaDon(userOnline.idTaiKhoan);

JSONObject xobj = new JSONObject();

xobj.put("listHoaDon", arrObj);

try {

BuildMessage xMessage = new BuildMessage();

String result = xMessage.buildMessage("QuanLyHoaDonChuShop", xobj);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**QuanLyHoaDonShipper**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONArray;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class QuanLyHoaDonShipper extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

boolean load = obj.getBoolean("loadList");

DatabaseManager database = new DatabaseManager();

JSONArray arrObj = database.getListHoaDonShipper(userOnline.idTaiKhoan);

JSONObject xobj = new JSONObject();

xobj.put("listHoaDon", arrObj);

try {

BuildMessage xMessage = new BuildMessage();

String result = xMessage.buildMessage("QuanLyHoaDonShipper", xobj);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**SetMaTracking**

package com.poly.shipper.servlet;

import java.io.IOException;

import java.util.ArrayList;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.dao.MySQLHelper;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserLocation;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

// Xá»­ lÃ½ update location cho shipper

public class SetMaTracking extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

int idHoaDon = obj.getInt("idHoaDon");

DatabaseManager databaseManager = new DatabaseManager();

JSONObject json = databaseManager.getHoaDon(idHoaDon);

int statusTracking = 0; // 0: gui thong bao ko co hoadon, 1: ko tim thay shipper, 2: gui thong bao hoa don da hoan thanh, ko the tracking nua, 3: da set tracking

if (json != null) {

int tinhTrang = json.getInt("idTinhTrang");

if (tinhTrang != 2) {// chua hoan thanh don hang ( tuc la dang chay toi lay hang hoac dang giao hang

String idShipper = json.getString("idShipper");

System.out.println("[SetMaTracking]" + idShipper);

UserOnline userOnlineShipper = userManager.findShipper(idShipper);

if (userOnlineShipper != null) {

System.out.println("[SetMaTracking] setSessionCustomer");

userOnlineShipper.setSessionCustomer(session);

statusTracking = 3;

}else {

System.out.println("[SetMaTracking] ko tim thay shipper");

statusTracking = 1;

}

}

else {

System.out.println("gui thong bao hoa don da hoan thanh, ko the tracking nua");

statusTracking = 2;

}

}

else {

System.out.println("gui thong bao ko co hoa don nay");

statusTracking = 0;

}

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("hoadon", json);

infoMessage.put("statusTracking", statusTracking);

String result = xMessage.buildMessage("SuccessSetTracking", infoMessage);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**TaoHoaDon**

package com.poly.shipper.servlet;

import java.io.IOException;

import java.util.ArrayList;

import java.util.concurrent.ExecutionException;

import java.util.concurrent.Future;

import java.util.concurrent.TimeUnit;

import java.util.concurrent.TimeoutException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.feature.SelectShipper;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class TaoHoaDon extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

int maTracking = -1; // matracking cÅ©ng lÃ  mÃ£ hÃ³a Ä‘Æ¡n.

boolean noShipperOnline = false;

String tenHang = obj.getString("tenHang");

String tenNguoiGui = obj.getString("tenNguoiGui");

String sdtNguoiGui = obj.getString("sdtNguoiGui");

String tu = obj.getString("tu");

String toaDoTu = obj.getString("toaDoTu");

String den = obj.getString("den");

String toaDoDen = obj.getString("toaDoDen");

double khoangCach = obj.getDouble("khoangCach");

int giaTienHang = obj.getInt("tienHang");

int giaTienShip = obj.getInt("tienShip");

boolean nguoiTra = obj.getBoolean("nguoiTra");

String tenNguoiNhan = obj.getString("tenNguoiNhan");

String sdtNguoiNhan = obj.getString("sdtNguoiNhan");

String ghiChu = obj.getString("ghiChu");

String idChuShop = userOnline.idTaiKhoan;

String[] locationHoaDon = toaDoTu.split(",");

double lonHoaDon = Double.parseDouble(locationHoaDon[0]);

double latHoaDon = Double.parseDouble(locationHoaDon[1]);

ArrayList<UserOnline> allShipper = userManager.findAllShipper();

if (allShipper.size() > 0) { // neu nhu co shipper online

SelectShipper selectShipper = new SelectShipper(allShipper, lonHoaDon, latHoaDon);

UserOnline userOnlineShipper = selectShipper.getShipper();

if (userOnlineShipper != null) { // phai luon luon tim duoc shipper

userOnlineShipper.themDonHang();

String idTaiKhoanShipper = userOnlineShipper.idTaiKhoan;

Session sessionShipper = userOnlineShipper.session;

DatabaseManager database = new DatabaseManager();

maTracking = database.insertHoaDon("", tenHang, tenNguoiGui, sdtNguoiGui, tu, toaDoTu, den, toaDoDen, khoangCach, giaTienHang, giaTienShip, nguoiTra, ghiChu, tenNguoiNhan, sdtNguoiNhan, 0, idChuShop, idTaiKhoanShipper);

if (sessionShipper != null ){ // gui don hang cho shipper

try {

BuildMessage xMessage = new BuildMessage();

obj.put("idHoaDon", maTracking);

String result = xMessage.buildMessage("ThongBaoShipperNhanHoaDon", obj);

sessionShipper.getRemote().sendString(result);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

}

else {

noShipperOnline = true;

}

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("noShipperOnline", noShipperOnline);

infoMessage.put("maTracking", Integer.toString(maTracking));

String result = xMessage.buildMessage("TaoHoaDon", infoMessage);

session.getRemote().sendString(result);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

/\*

public void sendDonHangForShipper(Session session) {

Future<Void> fut = null;

try

{

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("maTracking", Integer.toString(maTracking));

String result = xMessage.buildMessage("NhanDonHang", infoMessage);

fut = session.getRemote().sendStringByFuture(result);

// wait for completion (forever)

fut.get(10,TimeUnit.SECONDS);

}

catch (ExecutionException | InterruptedException e)

{

// Send failed

e.printStackTrace();

}

catch (TimeoutException e)

{

// timeout

e.printStackTrace();

if (fut != null)

{

// cancel the message

fut.cancel(true);

}

}

}

\*/

}

**UpdateLocationShipper**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.dao.MySQLHelper;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserLocation;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

// Xá»­ lÃ½ update location cho shipper

public class UpdateLocationShipper extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

UserManager userManager = getUserManager();

double lon = obj.getDouble("lon");

double lat = obj.getDouble("lat");

UserLocation userLocation = new UserLocation(lon, lat);

UserOnline userOnline = userManager.findUserOnline(session);

userOnline.setUserLocation(userLocation);

Session sessionCustomer = userOnline.getSessionCustomer();

Session sessionChuShop = userOnline.getSessionChuShop();

if (sessionCustomer != null) {

System.out.println("[UpdateLocationShipper] da tim thay customer, send vi tri cua shipper cho customer");

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("lon", lon);

infoMessage.put("lat", lat);

String result = xMessage.buildMessage("LayLocation", infoMessage);

sessionCustomer.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

else {

System.out.println("[UpdateLocationShipper] ko tim thay customer");

}

if (sessionChuShop != null) {

}

}

}

**UpdateThongTinChuShop**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class UpdateThongTinChuShop extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

boolean UpdateThongTinChuShopResult = false;

if (userOnline != null) {

String idTaiKhoan = userOnline.getIdTaiKhoan();

String hoTen = obj.getString("hoten");

String ngaySinh = obj.getString("ngaysinh");

String tenShop = obj.getString("tenshop");

String sdt = obj.getString("sdt");

String diaChi = obj.getString("diachi");

int gioiTinh = obj.getInt("gioitinh");

DatabaseManager database = new DatabaseManager();

if (database.updateInfoChuShop(idTaiKhoan, hoTen, ngaySinh, tenShop, sdt, diaChi, gioiTinh)) {

UpdateThongTinChuShopResult = true;

}

}

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("UpdateThongTinChuShopResult", UpdateThongTinChuShopResult);

String result = xMessage.buildMessage("UpdateThongTinChuShop", infoMessage);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**XoaDonHang**

package com.poly.shipper.servlet;

import java.io.IOException;

import org.eclipse.jetty.websocket.api.Session;

import org.json.JSONArray;

import org.json.JSONObject;

import com.poly.shipper.dao.DatabaseManager;

import com.poly.shipper.main.BuildMessage;

import com.poly.shipper.main.UserManager;

import com.poly.shipper.model.UserOnline;

import com.poly.shipper.model.WebSocketServlet;

import com.poly.shipper.model.WrapWebSocket;

public class XoaHoaDon extends WebSocketServlet{

public void doAction() {

WrapWebSocket wrapWebSocket = getWrapWebSocket();

Session session = wrapWebSocket.getSession();

JSONObject obj = wrapWebSocket.getInfoMessage();

System.out.println(obj.toString());

UserManager userManager = getUserManager();

UserOnline userOnline = userManager.findUserOnline(session);

boolean DeleteHoaDonResult = false;

int idHoaDon = obj.getInt("idHoaDon");

DatabaseManager database = new DatabaseManager();

DeleteHoaDonResult = database.deleteHoaDon(idHoaDon);

try {

BuildMessage xMessage = new BuildMessage();

JSONObject infoMessage = new JSONObject();

infoMessage.put("DeleteHoaDonResult", DeleteHoaDonResult);

String result = xMessage.buildMessage("QuanLyHoaDonChuShop", infoMessage);

session.getRemote().sendString(result);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

**Servlet**

<AllServlet>

<Servlet>

<Name>Login</Name>

<ClassName>com.poly.shipper.servlet.DoLogin</ClassName>

<Action>DoLogin</Action>

<Description>Xử lý login</Description>

</Servlet>

<Servlet>

<Name>TaoDonHang</Name>

<ClassName>com.poly.shipper.servlet.TaoDonHang</ClassName>

<Action>TaoDonHang</Action>

<Description>Tạo Đơn Hàng</Description>

</Servlet>

<Servlet>

<Name>Update Thong Tin Chu Shop</Name>

<ClassName>com.poly.shipper.servlet.UpdateThongTinChuShop</ClassName>

<Action>UpdateThongTinChuShop</Action>

<Description>Xử lý update thong tin chu shop</Description>

</Servlet>

<Servlet>

<Name>Tao Hoa Don</Name>

<ClassName>com.poly.shipper.servlet.TaoHoaDon</ClassName>

<Action>TaoHoaDon</Action>

<Description>Tạo hóa đơn</Description>

</Servlet>

<Servlet>

<Name>Quan Ly Hoa Don Chu Shop</Name>

<ClassName>com.poly.shipper.servlet.QuanLyHoaDonChuShop</ClassName>

<Action>QuanLyHoaDonChuShop</Action>

<Description>Quản lý hóa đơn chủ shop</Description>

</Servlet>

<Servlet>

<Name>Quan Ly Hoa Don Shipper</Name>

<ClassName>com.poly.shipper.servlet.QuanLyHoaDonShipper</ClassName>

<Action>QuanLyHoaDonShipper</Action>

<Description>Quản lý hóa đơn shipper</Description>

</Servlet>

<Servlet>

<Name>Update Location Shipper</Name>

<ClassName>com.poly.shipper.servlet.UpdateLocationShipper</ClassName>

<Action>UpdateLocationShipper</Action>

<Description>Update location cua shipper</Description>

</Servlet>

<Servlet>

<Name>Update Location Shipper</Name>

<ClassName>com.poly.shipper.servlet.SetMaTracking</ClassName>

<Action>SetMaTracking</Action>

<Description>Khach hang set ma tracking</Description>

</Servlet>

<Servlet>

<Name>Hoan Thanh Don Hang</Name>

<ClassName>com.poly.shipper.servlet.HoanThanhDonHang</ClassName>

<Action>HoanThanhDonHang</Action>

<Description>Xu lý hoàn thành don hàng</Description>

</Servlet>

<Servlet>

<Name>SOS Shipper</Name>

<ClassName>com.poly.shipper.servlet.SOSShipper</ClassName>

<Action>SOSShipper</Action>

<Description>Xu lý SOS Shipper</Description>

</Servlet>

<Servlet>

<Name>Load Info</Name>

<ClassName>com.poly.shipper.servlet.LoadInformation</ClassName>

<Action>LoadInformation</Action>

<Description>Lấy thông tin tài khoản khi đăng nhập thành công</Description>

</Servlet>

<Servlet>

<Name>Nhan Hang</Name>

<ClassName>com.poly.shipper.servlet.NhanHang</ClassName>

<Action>NhanHang</Action>

<Description>Xu lý nhan hàng</Description>

</Servlet>

<Servlet>

<Name>Xoa hoa don</Name>

<ClassName>com.poly.shipper.servlet.XoaHoaDon</ClassName>

<Action>XoaHoaDon</Action>

<Description>Xóa hóa đơn</Description>

</Servlet>

</AllServlet>

# 10.Kết luận

**Tóm tắt :**

Sau một thời gian giài phát triển, cuối cùng nhóm em đã hoàn thành sản phẩm của mình.

Sản phẩm mà nhóm phát triển là một ứng dụng cho các chủ shop có một phương thức vận chuyển hàng hóa hoàn toàn mới, an toàn tiện lợi và nhanh chóng.

**Khó khăn :**

Chưa quen sử dụng sever

Khó khăn khi ráp bài của các thành viên

**Hướng khắc phục :**

Cùng nhau thảo luận cách thức hoạt động và làm việc với sever

Sử dụng công cụ hỗ trợ SVN

**Các nguồn tham khảo :**

[http://www.**vn-zoom**.com](http://www.vn-zoom.com)

[**http://stackoverflow**.com](http://stackoverflow.com)

[https://**tinhte**.vn/**forum**s](https://tinhte.vn/forums)

http://[www.**android**.vn/**forum**s](http://www.android.vn/forums)

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

**TP.HCM, ngày 14 /11 /2016**

**GIẢNG VIÊN HD**