

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN
TỐT NGHIỆP ĐẠI HỌC

Đề tài:

**XÂY DỰNG PHẦN MỀM GIẢI PHÁP VẬN TẢI
DRIPER TRÊN THIẾT BỊ DI ĐỘNG**

Sinh viên thực hiện:	LÊ HỒNG DƯƠNG
Lớp:	ĐH HTTT1 - K9
Giảng viên hướng dẫn:	TH.S AN VĂN MINH

Hà Nội, 5/2018

MỤC LỤC

LỜI NÓI ĐẦU	1
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT	2
1.1 Cơ sở dữ liệu.....	2
1.1.1 Định nghĩa	2
1.1.2 Ưu điểm:.....	2
1.1.3 Nhược điểm:	2
1.1.4 Tính chất:.....	3
1.1.5 Tổ chức cơ sở dữ liệu:	3
1.1.6 Phân loại:	3
1.2 Hệ quản trị cơ sở dữ liệu	3
1.2.1 Định nghĩa	3
1.2.2 Ưu điểm của Hệ quản trị cơ sở dữ liệu:	3
1.2.3 Nhược điểm:	4
1.2.4 Các khả năng của hệ quản trị cơ sở dữ liệu.....	4
1.2.5 Các hệ quản trị cơ sở dữ liệu đều thực hiện các chức năng sau :.....	4
1.2.6 Giới thiệu về hệ quản trị cơ sở dữ liệu SQLite	5
1.3 Phân tích thiết kế hệ thống	5
1.3.1 Các khái niệm.....	5
1.3.2 Vai trò.....	6
1.3.3 Nhiệm vụ	6
1.3.4 Các phương pháp phân tích thiết kế hệ thống.....	6
1.3.5 Các giai đoạn phân tích thiết kế hệ thống	6
1.4 Công nghệ Java.....	7
1.4.1 Lịch sử phát triển.....	7
1.4.2 Phương châm	8
1.4.3 Khả năng của ngôn ngữ Java.....	8
1.4.4 Những đặc điểm của ngôn ngữ Java	8
1.4.5 Máy ảo Java (JVM - Java Virtual Machine)	9
1.4.6 Hai kiểu ứng dụng dưới ngôn ngữ java.....	9

1.4.7 Bộ phát triển ứng dụng Java (JDK- Java Development Kit)	9
1.5. Tìm hiểu về Hệ điều hành Android trên thiết bị di động	10
1.5.1 Giới thiệu.....	10
1.5.2 Cài đặt môi trường JDK và công cụ lập trình Android Studio	11
1.5.3 Vòng đời của một ứng dụng Android	12
1.5.4 Tính năng của Android.....	12
1.5.5 Các ứng dụng Android	13
1.5.6 Kiến trúc của hệ điều hành Android	13
1.5.7 Các thành phần cơ bản của một ứng dụng android	15
1.5.8 Các tệp tin quan trọng trong Android	17
1.5.9 Các giao diện cơ bản trong Android	20
1.5.10 Giới thiệu một số widget cơ bản trong lập trình android	28
1.5.11 Các thành phần được thiết kế trong Material Design	38
1.5.12 Lưu trữ dữ liệu trong Android.....	39
1.6 Google Map API trên Hệ điều hành Android.....	40
1.6.1 Google Maps	40
1.6.2 Các loại bản đồ Google Maps	40
1.6.3 Ứng dụng Google Maps trên nền tảng Android.....	41
1.6.4 Các phương thức của Google Maps	41
1.6.5 Marker trong Google Maps.	42
1.7 Công nghệ Nodejs	45
1.7.1 Tổng quan.....	45
1.7.2 Điểm nổi bật của NodeJS so với các ngôn ngữ lập trình khác	45
1.7.3 Đặc điểm của Nodejs	45
1.7.4 Ưu điểm.....	45
1.7.5 Nhược điểm.....	46
1.7.6 NodeJS core-module	46
1.7.7 Node.js third-party modules.....	46
1.8. Express framework trên NodeJS	47
1.8.1 Giới thiệu.....	47
1.8.2 Cài đặt Express Framework	47

1.8.3 Express route	47
1.9 Socket.io framework Real-time trong Nodejs	48
1.9.1 Giới thiệu.....	48
1.9.2 Cài đặt socket.io	49
1.10. Sử dụng MongoDB với NodeJs.....	51
1.10.1 Giới thiệu:.....	51
1.10.2 Nền tảng và ngôn ngữ hỗ trợ.....	51
CHƯƠNG 2: XÂY DỰNG ỨNG DỤNG	52
2.1 Quy trình nghiệp vụ.....	52
2.1.1 Khách hàng.....	52
2.1.2 Tài xế.....	52
2.2 Yêu cầu hệ thống	52
2.2.1 Khách hàng.....	52
2.2.2 Tài xế.....	53
2.3 Biểu đồ ca sử dụng	53
2.3.1 Xác định các tác nhân của hệ thống	53
2.3.2 Xác định các ca sử dụng.....	54
2.3.3 Các biểu đồ ca sử dụng.....	55
2.4 Đặc tả các ca sử dụng	56
2.4.1 Khách hàng.....	56
2.4.2 Tài xế.....	61
2.5 Giao diện chương trình.....	65
2.5.1 Giao diện của phần mềm dành cho khách hàng.....	65
2.5.2 Giao diện của phần mềm dành cho tài xế	69
KẾT LUẬN CHUNG	73
TÀI LIỆU THAM KHẢO	74

LỜI NÓI ĐẦU

Hiện nay trong cuộc Cách mạng công nghiệp 4.0 chúng ta biết đến định nghĩa kinh tế chia sẻ có thể hiểu là xóa bỏ tâm lý tư hữu, tận dụng tài sản nhàn rỗi để có thể tạo ra doanh thu bằng cách cho thuê hoặc là mượn lại. Vấn đề hiện nay ở khu vực Hà Nội, một thành phố với 7,5 triệu dân và 5 triệu xe máy và xe ga.

Đổi lập lại thì khoảng hơn 500.000 xe hơi trong thành phố Hà Nội, nếu so sánh giữa xe 2 bánh và ô tô thì tỷ lệ này là 10:1. Các nhà hoạch định chính sách của thành phố Hà Nội còn ước tính số lượng xe mô tô sẽ tăng lên 6 triệu chiếc vào năm 2020, còn xe ô tô sẽ vượt ngưỡng 800.000 chiếc.

*Với số lượng lớn những chiếc xe moto hiện có ở Hà Nội như hiện tại và vấn đề giải quyết vấn đề việc làm, an sinh xã hội hiện nay với thời gian học tập ở trường học tôi muốn tổng kết lại một số công nghệ và kiến thức đã học ở trường để xây dựng và phát triển một phần mềm giúp giải quyết các vấn đề đã nêu trên bằng **“Phần mềm giải pháp vận tải Driper”** trên thiết bị di động sử dụng hệ điều hành Android.*

“Phần mềm giải pháp vận tải Driper” giúp kết nối những người có xe máy và người có nhu cầu đi lại trong ngày bằng một chiếc di động thông minh một phần mềm theo dõi những kết nối đó. Giúp cho người có xe máy có thêm một khoản thu nhập bằng cách giúp những người có nhu cầu đi lại và vận chuyển hàng.

Nội dung báo cáo:

Chương 1: Cơ sở lý thuyết

Chương 2: Xây dựng ứng dụng

Tôi xin chân thành cảm ơn đến Ths. An Văn Minh, thầy đã hướng dẫn cũng như giúp đỡ tôi trong việc nghiên cứu và hoàn thành nội dung báo cáo đồ án tốt nghiệp này!

Hà Nội, ngày 20 tháng 4 năm 2018

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1 Cơ sở dữ liệu

1.1.1 Định nghĩa

Cơ sở dữ liệu là một hệ thống các thông tin có cấu trúc, được lưu trữ trên các thiết bị lưu trữ nhằm thỏa mãn yêu cầu khai thác thông tin đồng thời của nhiều người sử dụng hay nhiều chương trình ứng dụng chạy cùng một lúc với những mục đích khác nhau

1.1.2 Ưu điểm:

Giảm sự trùng lặp thông tin xuống mức thấp nhất. Do đó đảm bảo thông tin có tính nhất quán và toàn vẹn dữ liệu.

Đảm bảo dữ liệu có thể được truy xuất theo nhiều cách khác nhau.

Nhiều người có thể sử dụng một cơ sở dữ liệu.

1.1.3 Nhược điểm:

Tính chủ quyền của dữ liệu:

- Thể hiện ở phương diện an toàn dữ liệu.
- Khả năng biểu diễn mối liên hệ ngữ nghĩa của dữ liệu và tính chính xác của dữ liệu.
- Người khai thác cơ sở dữ liệu phải cập nhật cho cơ sở dữ liệu những thông tin mới nhất.
- Tính bảo mật và quyền khai thác thông tin của người sử dụng:
- Do ưu điểm cơ sở dữ liệu có thể cho nhiều người khai thác đồng thời. nên cần phải có một cơ chế bảo mật phân quyền khai thác cơ sở dữ liệu.
- Các hệ điều hành nhiều người sử dụng hay cục bộ đều cung cấp cơ chế này.
- Tranh chấp dữ liệu:
- Khi nhiều người cùng truy nhập cơ sở dữ liệu với các mục đích khác nhau. Rất có thể sẽ xảy ra hiện tượng tranh chấp dữ liệu.
- Cần có cơ chế ưu tiên khi truy cập cơ sở dữ liệu. Ví dụ: Người quản trị luôn có thể truy cập cơ sở dữ liệu.
- Cấp quyền ưu tiên cho những người dùng khác.
- Cần đảm bảo an toàn dữ liệu khi có sự cố:

Khi cơ sở dữ liệu nhiều và được quản lý tập trung. Khả năng rủi ro mất dữ liệu rất cao. Các nguyên nhân chính là mất điện đột ngột hoặc hỏng thiết bị lưu trữ.

Hiện tại có một số hệ điều hành đã có cơ chế tự động sao lưu ổ cứng và sửa lỗi khi có sự cố xảy ra.

Tuy nhiên: cần tắc vô áy náy. Chúng ta nên sao lưu dự phòng cho dữ liệu đề phòng trường hợp xấu xảy ra.

1.1.4 Tính chất:

Phản ánh một phần của thế giới thực được cập nhật phản ánh sự thay đổi của thế giới nó biểu diễn.

Cơ sở dữ liệu là một tập hợp dữ liệu liên kết với nhau một cách logic và mang một ý nghĩa nào đó.

1.1.5 Tổ chức cơ sở dữ liệu:

Các dữ liệu lưu trữ có cấu trúc thành các bản ghi , các trường dữ liệu.

Các dữ liệu lưu trữ có mối quan hệ với nhau.

Khả năng truy xuất thông tin từ cơ sở dữ liệu => Dễ dàng truy cập, quản lí, cập nhật dữ liệu.

1.1.6 Phân loại:

Cơ sở dữ liệu được phân chia ra nhiều loại khác nhau:

- Cơ sở dữ liệu dạng file: dữ liệu được lưu trữ dưới dạng các file có thể là text, ascii, *.dbf. Tiêu biểu cho cơ sở dữ liệu dạng file là *.mdb Foxpro
- Cơ sở dữ liệu quan hệ: dữ liệu được lưu trữ trong các bảng dữ liệu gọi là các thực thể, giữa các thực thể này có mối liên hệ với nhau gọi là các quan hệ, mỗi quan hệ có các thuộc tính, trong đó có một thuộc tính là khóa chính. Các hệ quản trị hỗ trợ cơ sở dữ liệu quan hệ như: MS SQL server, Oracle, MySQL,...
- Cơ sở dữ liệu hướng đối tượng: dữ liệu cũng được lưu trữ trong các bảng dữ liệu nhưng các bảng có bổ sung thêm các tính năng hướng đối tượng như lưu trữ thêm các hành vi, nhằm thể hiện hành vi của đối tượng. Mỗi bảng xem như một lớp dữ liệu, một dòng dữ liệu trong bảng là một đối tượng. Các hệ quản trị có hỗ trợ cơ sở dữ liệu hướng đối tượng như: MS SQL server, Oracle, Postgres,...
- Cơ sở dữ liệu bán cấu trúc: dữ liệu được lưu dưới dạng XML, với định dạng này thông tin mô tả về đối tượng thể hiện trong các thẻ. Đây là cơ sở dữ liệu có nhiều ưu điểm do lưu trữ được hầu hết các loại dữ liệu khác nhau nên cơ sở dữ liệu bán cấu trúc là hướng mới trong nghiên cứu và ứng dụng.

1.2 Hệ quản trị cơ sở dữ liệu

1.2.1 Định nghĩa

Hệ quản trị cơ sở dữ liệu (Database Management System - DBMS): Là một hệ thống phần mềm cho phép tạo lập cơ sở dữ liệu và điều khiển mọi truy nhập đối với cơ sở dữ liệu đó.

Trên thị trường phần mềm ở Việt Nam đã xuất hiện khá nhiều phần mềm hệ quản trị cơ sở dữ liệu như: SQL Server, Oracle, v.v...

Hệ quản trị cơ sở dữ liệu quan hệ (Relation Database Management System - RDBMS) là một hệ quản trị cơ sở dữ liệu theo mô hình quan hệ.

1.2.2 Ưu điểm của Hệ quản trị cơ sở dữ liệu:

Quản lý được dữ liệu dư thừa.

Đảm bảo tính nhất quán cho dữ liệu.
Tạo khả năng chia sẻ dữ liệu nhiều hơn.
Cải tiến tính toàn vẹn cho dữ liệu.

1.2.3 Nhược điểm:

Hệ quản trị cơ sở dữ liệu tốt thì khá phức tạp.
Hệ quản trị cơ sở dữ liệu tốt thường rất lớn chiếm nhiều dung lượng bộ nhớ.
Giá cả khác nhau tùy theo môi trường và chức năng.
Hệ quản trị cơ sở dữ liệu được viết tổng quát cho nhiều người dùng thì thường chậm.

1.2.4 Các khả năng của hệ quản trị cơ sở dữ liệu

Có hai khả năng chính cho phép phân biệt các hệ quản trị cơ sở dữ liệu với các kiểu hệ thống lập trình khác:

Khả năng quản lý dữ liệu tồn tại lâu dài: đặc điểm này chỉ ra rằng có một cơ sở dữ liệu tồn tại trong một thời gian dài, nội dung của cơ sở dữ liệu này là các dữ liệu mà hệ quản trị cơ sở dữ liệu truy nhập và quản lý.

Khả năng truy nhập các khối lượng dữ liệu lớn một cách hiệu quả.

Ngoài hai khả năng cơ bản trên, hệ quản trị cơ sở dữ liệu còn có các khả năng khác mà có thể thấy trong hầu hết các hệ quản trị cơ sở dữ liệu đó là:

Hỗ trợ ít nhất một mô hình dữ liệu hay một sự trừu tượng toán học mà qua đó người sử dụng có thể quan sát dữ liệu.

Đảm bảo tính độc lập dữ liệu hay sự bất biến của chương trình ứng dụng đối với các thay đổi về cấu trúc trong mô hình dữ liệu.

Hỗ trợ các ngôn ngữ cao cấp nhất định cho phép người sử dụng định nghĩa cấu trúc dữ liệu, truy nhập dữ liệu và thao tác dữ liệu.

Quản lý giao dịch, có nghĩa là khả năng cung cấp các truy nhập đồng thời, đúng đắn đối với cơ sở dữ liệu từ nhiều người sử dụng tại cùng một thời điểm.

Điều khiển truy nhập, có nghĩa là khả năng hạn chế truy nhập đến các dữ liệu bởi những người sử dụng không được cấp phép và khả năng kiểm tra tính đúng đắn của cơ sở dữ liệu.

Phục hồi dữ liệu, có nghĩa là có khả năng phục hồi dữ liệu, không làm mất mát dữ liệu với các lỗi hệ thống.

1.2.5 Các hệ quản trị cơ sở dữ liệu đều thực hiện các chức năng sau :

- Lưu trữ dữ liệu.
- Tạo ra và duy trì cơ sở dữ liệu.
- Cho phép nhiều người dùng truy xuất đồng thời.
- Hỗ trợ tính bảo mật và riêng tư.

- Cho phép xem và xử lý dữ liệu lưu trữ.
- Cho phép cập nhật và lưu trữ dữ liệu sau khi cập nhật.
- Cung cấp một cơ chế chỉ mục (index) hiệu quả để truy cập nhanh các dữ liệu lựa chọn.
- Cung cấp tính nhất quán giữa các bản ghi khác nhau.
- Bảo vệ dữ liệu khỏi mất mát bằng các quá trình sao lưu (backup) và phục hồi (recovery).

1.2.6 Giới thiệu về hệ quản trị cơ sở dữ liệu SQLite

SQLite là một cơ sở dữ liệu quan hệ, mã nguồn mở, nó được tích hợp sẵn trên hệ điều hành Android, vì vậy có thể sử dụng nó bất cứ lúc nào, và không cần phải cấu hình gì thêm.

SQLite là hệ thống cơ sở dữ liệu quan hệ nhỏ gọn, hoàn chỉnh, có thể cài đặt bên trong các ứng dụng khác. SQLite được Richard Hipp viết dưới dạng thư viện bằng ngôn ngữ lập trình C.

SQLite là một thư viện thực thi các chức năng của một database engine với đặc điểm giống như một phần mềm portable, không cần cài đặt, không cần cấu hình, không cần server. Những điểm này rất khác so với việc sử dụng SQL Server hoặc Oracle, nhưng nó vẫn có chuyển giao để đảm bảo tính toàn vẹn và an toàn trong quá trình thao tác dữ liệu.

SQLite có các ưu điểm sau:

- Tin cậy: các hoạt động chuyển giao nội trong cơ sở dữ liệu được thực hiện trọn vẹn, không gây lỗi khi xảy ra sự cố phần cứng.
- Tuân theo chuẩn SQL92 (chỉ có một vài đặc điểm không hỗ trợ).
- Không cần cài đặt cấu hình.
- Kích thước chương trình gọn nhẹ, với cấu hình đầy đủ chỉ không đầy 300 kB.
- Thực hiện các thao tác đơn giản nhanh hơn các hệ thống cơ sở dữ liệu khách/chủ khác.
- Không cần phần mềm phụ trợ.
- Phần mềm tự do với mã nguồn mở, được chú thích rõ ràng.
- SQLite được đánh giá là nhanh, tin cậy.

1.3 Phân tích thiết kế hệ thống

1.3.1 Các khái niệm

Hệ thống là tập hợp các phần tử có những mối quan hệ ràng buộc lẫn nhau cùng hoạt động chung cho một số mục tiêu nào đó, thực hiện trao đổi vào/ra với môi trường ngoài.

Hệ thống quản lý là một hệ thống có mục đích mang lại lợi nhuận hoặc lợi ích nào đó. Đặc điểm của hệ thống là có con người tham gia và trao đổi thông tin

Hệ thống thông tin là hệ thống sử dụng công nghệ thông tin để thu thập, truyền, lưu trữ, xử lý và biểu diễn thông tin trong một hay nhiều quá trình.

1.3.2 Vai trò

- Hệ thống thông tin đóng vai trò trung gian giữa hệ quyết định và hệ tác nghiệp trong hệ thống quản lý.
- Xây dựng hệ thống thông tin là một hoạt động tổng hợp của các môn học thuộc lĩnh vực Công nghệ thông tin. Hệ thống thông tin tin học là một ứng dụng đầy đủ và toàn diện nhất các thành tựu của CNTT vào một tổ chức
- Xây dựng hệ thống thông tin là một hoạt động trải qua một loạt các giai đoạn để có sản phẩm cuối cùng là một hệ thống thông tin.
- Phân tích và thiết kế là giai đoạn đầu của quá trình phát triển hệ thống thông tin. Nó quyết định đến chất lượng và giá thành của hệ thống.
- Nhờ thiết kế tốt mà các hệ thống thông tin được duy trì hoạt động tốt và phát triển.
- Quy mô và mức độ phức tạp của các hệ thống ngày càng tăng, nên phân tích và thiết kế trở thành một yêu cầu bắt buộc để có được một hệ thống tốt.

1.3.3 Nhiệm vụ

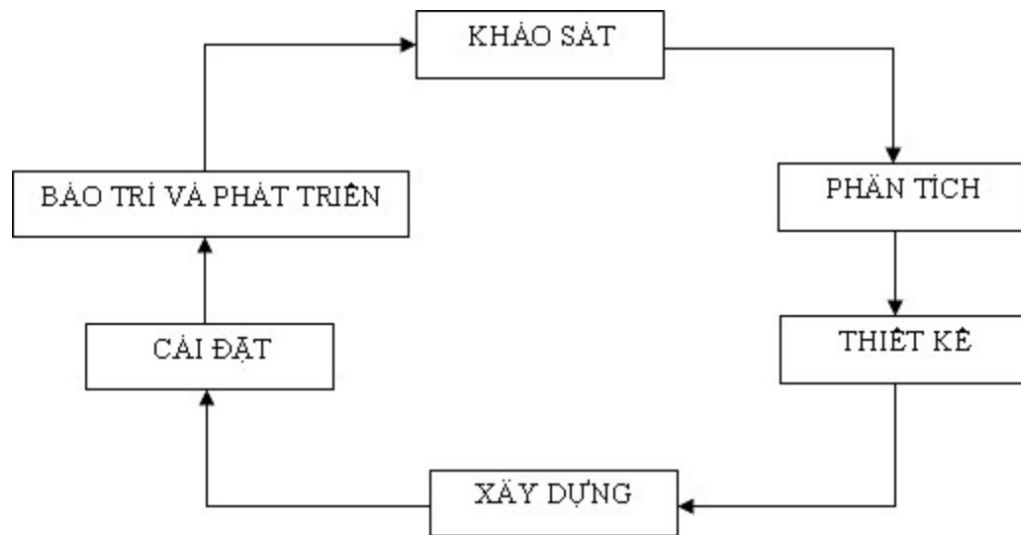
- Trao đổi thông tin với người dùng.
- Thực hiện việc liên lạc giữa các bộ phận và cung cấp thông tin cho các hệ tác nghiệp và hệ quyết định.

1.3.4 Các phương pháp phân tích thiết kế hệ thống

- Phương pháp thiết kế cổ điển.
- Phương pháp phân tích thiết kế hệ thống bán cấu trúc.
- Phương pháp phân tích thiết kế hệ thống có cấu trúc.

1.3.5 Các giai đoạn phân tích thiết kế hệ thống

- Khảo sát hiện trạng và xác lập dự án.
- Phân tích hệ thống: Phân tích các chức năng và dữ liệu của hệ thống cũ để đưa ra mô tả của hệ thống mới.
- Thiết kế hệ thống.
- Xây dựng hệ thống.



*Sơ đồ thể hiện các giai đoạn triển khai
xây dựng một dự án*

1.4 Công nghệ Java

Java là một ngôn ngữ lập trình hướng đối tượng và dựa trên các lớp. Khác với phần lớn ngôn ngữ lập trình khác như C/C++, thay vì biên dịch mã nguồn thành mã máy hoặc thông dịch mã nguồn khi chạy, Java được thiết kế để biên dịch mã nguồn thành bytecode, bytecode sau đó sẽ được môi trường thực thi chạy.

Trước đây, Java chạy chậm hơn những ngôn ngữ dịch thẳng ra mã máy như C và C++, nhưng sau này nhờ công nghệ biên dịch tại chỗ, khoảng cách này đã được thu hẹp, và trong một số trường hợp đặc biệt Java có thể chạy nhanh hơn. Java chạy nhanh hơn những ngôn ngữ thông dịch như Python, Perl, PHP. Java chạy tương đương so với C#, một ngôn ngữ khá tương đồng về mặt cú pháp và quá trình dịch/chạy.

Cú pháp Java được vay mượn nhiều từ C/ C++ nhưng có cú pháp hướng đối tượng đơn giản hơn và ít tính năng xử lý cấp thấp hơn. Do đó việc viết một chương trình bằng Java dễ hơn, đơn giản hơn, đỡ tốn công sửa lỗi hơn.

Trong Java, hiện tượng rò rỉ bộ nhớ hầu như không xảy ra do bộ nhớ được quản lý bởi Java Virtual Machine (JVM) bằng cách tự động "dọn dẹp rác". Người lập trình không phải quan tâm đến việc cấp phát và xóa bộ nhớ như C, C++. Tuy nhiên khi sử dụng những tài nguyên mạng, file IO, database mà người lập trình không đóng các luồng thì rò rỉ dữ liệu vẫn có thể xảy ra.

1.4.1 Lịch sử phát triển

Năm 1990, Sun Microsystems thực hiện dự án Green nhằm phát triển phần mềm trong các thiết bị dân dụng. James Gosling, chuyên gia lập trình đã tạo ra một ngôn ngữ

lập trình mới có tên là Oak. Ngôn ngữ này có cú pháp gần giống như C++ nhưng bỏ qua các lập trình cấp thấp của C++ như truy cập trực tiếp tài nguyên hệ thống, con trỏ, định nghĩa chồng các tác tử...

Khi ngôn ngữ Oak trưởng thành, WWW cũng đang vào thời kỳ phát triển mạnh mẽ, Sun cho rằng đây là một ngôn ngữ thích hợp cho Internet. Năm 1995, Oak đổi tên thành Java và sau đó đến 1996 Java đã được xem như một chuẩn công nghiệp cho Internet.

1.4.2 Phương châm

Có 5 mục tiêu chính trong việc xây dựng ngôn ngữ Java:

- Đơn giản, hướng đối tượng và quen thuộc.
- Mạnh mẽ và an toàn.
- Kiến trúc trung lập và di động.
- Thực thi với hiệu suất cao.
- Dịch ra bytecode, phân luồng và năng động.

1.4.3 Khả năng của ngôn ngữ Java

Là một ngôn ngữ bậc cao như C++, Perl, SmallTalk,.. Cho nên có thể được dùng để tạo ra các ứng dụng để giải quyết các vấn đề về số, xử lý văn bản, tạo ra trò chơi, và nhiều thứ khác.

Có các môi trường lập trình đồ họa như Visual Java, Symantec Cafe, Jbuilder, Jcreator, ...

Có khả năng truy cập dữ liệu từ xa thông qua cầu nối JDBC (Java DataBase Connectivity)

Hỗ trợ các lớp hữu ích, tiện lợi trong lập trình các ứng dụng mạng (Socket) cũng như truy xuất Web.

Hỗ trợ lập trình phân tán (Remote Method Invocation) cho phép một ứng dụng có thể được xử lý phân tán trên các máy tính khác nhau.

Và luôn được bổ sung các tính năng cao cấp khác trong các phiên bản sau.

1.4.4 Những đặc điểm của ngôn ngữ Java

Ngôn ngữ thuần túy hướng đối tượng.

Ngôn ngữ đa nền cho phép một chương trình có thể thực thi trên các hệ điều hành khác nhau (MS Windows, UNIX, Linux) mà không phải biên dịch lại chương trình. Phương châm của java là "Viết một lần , Chạy trên nhiều nền" (Write Once, Run Anywhere).

Ngôn ngữ đa luồng, cho phép trong một chương trình có thể có nhiều luồng điều khiển được thực thi song song nhau, rất hữu ích cho các xử lý song song.

Ngôn ngữ phân tán, cho phép các đối tượng của một ứng dụng được phân bố và thực thi trên các máy tính khác nhau.

Ngôn ngữ động, cho phép mã lệnh của một chương trình được tải từ một máy tính về máy của người yêu cầu thực thi chương trình.

Ngôn ngữ an toàn, tất cả các thao tác truy xuất vào các thiết bị vào ra đều thực hiện trên máy ảo nhờ đó hạn chế các thao tác nguy hiểm cho máy tính thật.

Ngôn ngữ đơn giản, dễ học, kiến trúc chương trình đơn giản, trong sáng.

1.4.5 Máy ảo Java (JVM - Java Virtual Machine)

Để đảm bảo tính đa nền, Java sử dụng cơ chế **Máy ảo của Java**. ByteCode đó là ngôn ngữ máy của Máy ảo Java tương tự như các lệnh nhị phân của các máy tính thực. Một chương trình sau khi được viết bằng ngôn ngữ Java (có phần mở rộng là .java) phải được biên dịch thành tập tin thực thi được trên máy ảo Java (có phần mở rộng là .class). Tập tin thực thi này chứa các chỉ thị dưới dạng mã Bytecode mà máy ảo Java hiểu được phải làm gì.

Khi thực hiện một chương trình, máy ảo Java lần lượt thông dịch các chỉ thị dưới dạng Bytecode thành các chỉ thị dạng nhị phân của máy tính thực và thực thi thực sự chúng trên máy tính thực.

Máy ảo thực tế đó là một chương trình thông dịch. Vì thế các hệ điều hành khác nhau sẽ có các máy ảo khác nhau. Để thực thi một ứng dụng của Java trên một hệ điều hành cụ thể, cần phải cài đặt máy ảo tương ứng cho hệ điều hành đó.

1.4.6 Hai kiểu ứng dụng dưới ngôn ngữ java

Khi bắt đầu thiết kế một ứng dụng dưới ngôn ngữ Java, phải chọn kiểu cho nó là **Application** hay **Applet**.

Applet: Là một chương trình ứng dụng được nhúng vào các trang web. Mã của chương trình được tải về máy người dùng từ Web server khi người dùng truy xuất đến trang web chứa nó.

Application: Là một chương trình ứng dụng được thực thi trực tiếp trên các máy ảo của Java.

1.4.7 Bộ phát triển ứng dụng Java (JDK- Java Development Kit)

JDK là một bộ công cụ cho phép người lập trình phát triển và triển khai các ứng dụng bằng ngôn ngữ java được cung cấp miễn phí bởi công ty JavaSoft (hoặc Sun). Có các bộ JDK cho các hệ điều hành khác nhau. Các ấn bản của JDK không ngừng được phát hành, có thể tải về từ địa chỉ <http://java.sun.com>.

Bộ công cụ này gồm các chương trình thực thi đáng chú ý sau:

- javac: Chương trình biên dịch các chương trình nguồn viết bằng ngôn ngữ java ra các tập tin thực thi được trên máy ảo Java.
- java: Đây là chương trình làm máy ảo của Java, thông dịch mã Bytecode của các chương trình kiểu application thành mã thực thi của máy thực.
- appletviewer: Bộ thông dịch, thực thi các chương trình kiểu applet.
- javadoc: Tạo tài liệu về chú thích chương trình nguồn một cách tự động.

- jdb: Trình gỡ rối.
- rmic: Tạo các sơ khai cho ứng dụng kiểu RMI (Remote Method Invocation), một kỹ thuật cài đặt các đối tượng phân tán vô cùng hiệu vô cùng hiệu quả và linh động.
- rmiregistry: Phục vụ danh bạ (Name Server) trong hệ thống RMI.
- Các phiên bản Java đã phát hành:
 - JDK 1.0 (23 tháng 01, 1996)
 - JDK 1.1 (19 tháng 2, 1997)
 - J2SE 1.2 (Playground) 08 tháng 12, 1998
 - J2SE 1.3 (Kestrel) 08 tháng 5, 2000
 - J2SE 1.4.0 (Merlin) 06 tháng 02, 2002
 - J2SE 5 (1.5.0) (Tiger) 30 tháng 9, 2004
 - Java SE 6 (còn gọi là Mustang), được công bố 11 tháng 12 năm 2006, thông tin chính tại <http://java.sun.com/javase/6/>. Các bản cập nhật 2 và 3 được đưa ra vào năm 2007, bản cập nhật 4 đưa ra tháng 1 năm 2008.
 - JDK 6.18, 2010
 - Java SE 7 (còn gọi là Dolphin), được bắt đầu từ tháng 8 năm 2006 và công bố ngày 28 tháng 7 năm 2011.
 - JDK 8, 18 tháng 3 năm 2014
 - JDK 9, năm 2016.

1.5. Tìm hiểu về Hệ điều hành Android trên thiết bị di động

1.5.1 Giới thiệu

Android là một hệ điều hành dựa trên nền tảng **Linux** được thiết kế dành cho các thiết bị di động có màn hình cảm ứng như điện thoại thông minh và máy tính bảng. Ban đầu, **Android** được phát triển bởi Tổng công ty **Android**, với sự hỗ trợ tài chính từ Google và sau này được chính **Google** mua lại vào năm 2005. **Android** ra mắt vào năm 2007 cùng với tuyên bố thành lập Liên minh thiết bị cầm tay mở: một hiệp hội gồm các công ty phần cứng, phần mềm, và viễn thông với mục tiêu đẩy mạnh các tiêu chuẩn mở cho các thiết bị di động. Chiếc điện thoại đầu tiên chạy **Android** được bán vào tháng 10 năm 2008.

Các phiên bản đã phát hành của Android:

Tên mã	Phiên bản	Ngày phát hành	Cấp API
Alpha	1.0	September 23, 2008	1
Beta	1.1	February 9, 2009	2

Cupcake	1.5	April 27, 2009	3
Donut	1.6	September 15, 2009	4
Eclair	2.0 – 2.1	October 26, 2009	5–7
Froyo	2.2 – 2.2.3	May 20, 2010	8
Gingerbread	2.3 – 2.3.7	December 6, 2010	9–10
Honeycomb	3.0 – 3.2.6	February 22, 2011	11–13
Ice Cream Sandwich	4.0 – 4.0.4	October 18, 2011	14–15
Jelly Bean	4.1 – 4.3.1	July 9, 2012	16–18
KitKat	4.4 – 4.4.4	October 31, 2013	19
Lollipop	5.0 – 5.1.1	November 12, 2014	21–22
Marshmallow	6.0 – 6.0.1	October 5, 2015	23
Nougat	7.0 – 7.1.1	August 22, 2016	24–25

Bảng 1.1: Các phiên bản hệ điều hành Android phát triển theo thời gian.

1.5.2 Cài đặt môi trường JDK và công cụ lập trình Android Studio

Để có thể xây dựng được phần mềm trên nền tảng Android thì máy tính phải được cài môi trường java và công cụ lập trình, ở đề tài này nhóm em sử dụng công cụ Android Studio để lập trình, ngoài công cụ này thì các lập trình viên còn có thể sử dụng bộ ADT dùng cho Eclipse để xây dựng phần mềm.

- Cài đặt JDK:

Tải bộ cài đặt JDK tại:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Thiết lập biến môi trường theo trang web sau

<http://windybook.com/java-bai-2-thiet-lap-bien-moi-truong-java-va-cai-dat-ide-netbean/>

- Cài đặt Android Studio:

Tải file cài đặt tại: <https://developer.android.com/studio/index.html>

Xem hướng dẫn cài đặt tại link sau

<https://developer.android.com/studio/install.html>

1.5.3 Vòng đời của một ứng dụng Android

Trước khi tìm hiểu về vòng đời của một ứng dụng Android thì ta cần nắm rõ một số các khái niệm sau:

Application: Mỗi một Android Project khi biên dịch thành công thì sẽ được đóng gói thành tập tin *.apk*, tập tin *.apk* được gọi là một Application.

Activities: Thông thường trong một ứng dụng (Application) sẽ có một hoặc nhiều Activity. Mỗi một Activity này sẽ có một vòng đời riêng độc lập hoàn toàn với các Activity khác và bắt buộc nó phải được khai báo trong Manifest.

Activity Stack: Activity Stack hoạt động theo cơ chế ngăn xếp. Mỗi một Activity mới được mở lên nó sẽ ở bên trên Activity cũ, để trở về Activity thì chỉ cần nhấn nút “Back” để trở về hoặc viết lệnh.

Tasks: là khả năng thực hiện một công việc nào đó giữa các ứng dụng với nhau, cụ thể là các Activity.

Vòng đời của một Activity thường có 3 trạng thái sau:

Running (đang kích hoạt): Khi màn hình là Foreground (Activity nằm trên cùng ứng dụng và cho phép người sử dụng tương tác).

Paused (tạm dừng): Activity bị mất focus nhưng mà vẫn nhìn thấy được Activity này. Trường hợp này nó vẫn có khả năng bị hệ thống tự động “Kill” trong tình huống bộ nhớ quá ít.

Stopped (dừng – không phải Destroyed): Activity mất focus và không nhìn thấy được (ví dụ mở một Activity mới lên mà Full màn hình chẳng hạn). Trong trường hợp này nó có thể bị hệ thống “Kill” trong bất kỳ tình huống nào.

Như vậy vòng đời của một ứng dụng Android là bao gồm nhiều vòng đời của một Activity.

1.5.4 Tính năng của Android

Android là một hệ điều hành mạnh mẽ cạnh tranh với Apple, với nhiều tính năng tuyệt vời. Một vài tính năng chính như sau:

Giao diện đẹp: Android cung cấp một giao diện người dùng đẹp và trực quan.

Kết nối: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.

Lưu trữ: SQLite: một cơ sở dữ liệu nhẹ, được sử dụng cho mục đích lưu trữ.

Media: H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, and BMP.

Tin nhắn: SMS, MMS.

Trình duyệt Web: Cơ bản dựa trên mã nguồn mở WebKit, cùng với Chrome's V8 JavaScript, hỗ trợ HTML5 và CSS3.

Cảm ứng đa điểm: Android có hỗ trợ cho cảm ứng đa điểm mà ban đầu đã được tạo sẵn trong các thiết bị cầm tay như HTC Hero.

Đa tác vụ: Cùng thời điểm các ứng dụng khác nhau có thể chạy đồng thời.

Thay đổi kích thước widgets: widgets có thể thay đổi kích thước vì vậy người dùng có thể mở rộng để hiển thị nhiều nội dung, hoặc thu nhỏ để tiết kiệm không gian.

Đa ngôn ngữ.

FCM (Firebase Cloud Messaging) là một dịch vụ cho phép nhà phát triển gửi các thông điệp ngắn đến người dùng của họ trên thiết bị Android, mà không cần giải pháp đồng bộ độc quyền.

Wi-Fi: Công nghệ cho phép phát hiện và kết nối wifi nhanh, trên một băng thông cao thông qua kết nối Peer-to-Peer.

Android Beam: Phổ biến là NFC, một công nghệ cho phép chia sẻ ngay lập tức, chỉ cần chạm hai điện thoại cùng hỗ trợ NFC.

1.5.5 Các ứng dụng Android

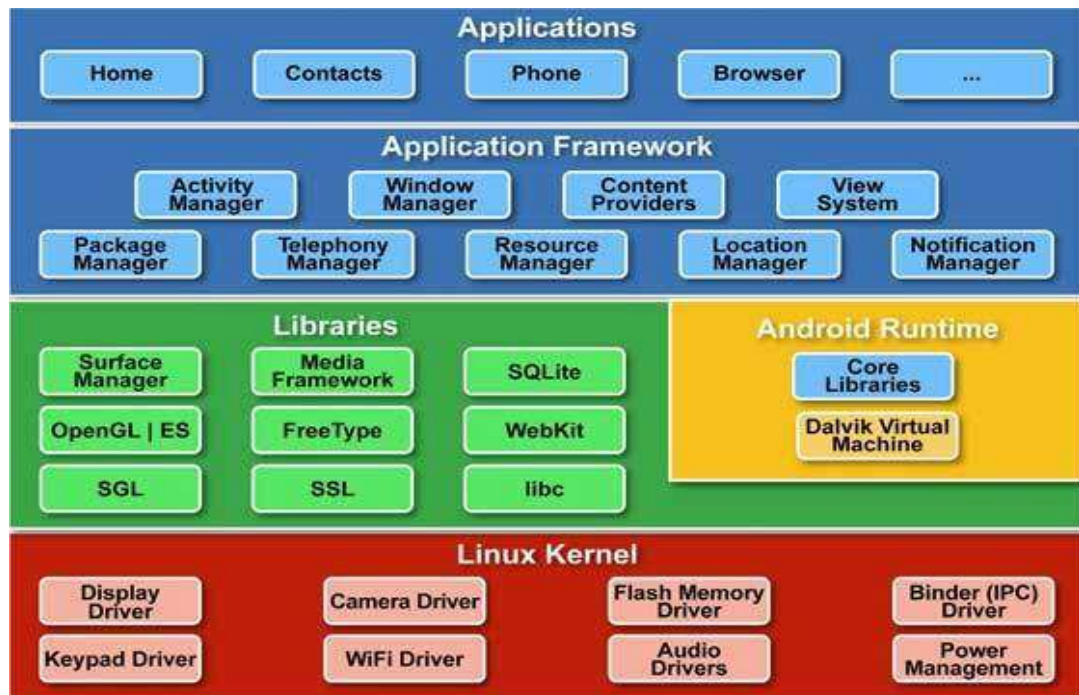
Ứng dụng Android thông thường được phát triển bằng ngôn ngữ Java, sử dụng Android Software Development Kit.

Mỗi lần phát triển, ứng dụng Android có thể đóng gói dễ dàng và bán ra thông qua Google Play hoặc Amazon Appstore.

Android được dùng cho hàng trăm triệu thiết bị di động tại hơn 190 quốc gia trên toàn thế giới. Mỗi ngày có hơn 1 triệu thiết bị Android mới được kích hoạt.

1.5.6 Kiến trúc của hệ điều hành Android

Hệ điều hành Android là một tập hợp của các thành phần phần mềm được chia thành 5 phần và 4 lớp chính:



Hình 1.1: Mô tả kiến trúc của hệ điều hành Android

a. Linux Kernel

Linux Kernel là lớp thấp nhất. Nó cung cấp các chức năng cơ bản như quản lý tiến trình, quản lý bộ nhớ, quản lý thiết bị như: Camera, bàn phím, màn hình,... Ngoài ra, nó còn quản lý mạng, driver của các thiết bị, điều này gỡ bỏ sự khó khăn về giao tiếp với các thiết bị ngoại vi.

b. Libraries

Phía trên Linux Kernel là tập hợp các bộ thư viện mã nguồn mở WebKit, bộ thư viện nổi tiến libc, cơ sở dữ liệu SQLite hữu ích cho việc lưu trữ và chia sẻ dữ liệu, bộ thư viện thẻ phát, ghi âm về âm thanh, hoặc video. Thư viện SSL chịu trách nhiệm cho bảo mật Internet.

c. Android Runtime

Đây là thành phần thứ 3 trong cấu trúc, thuộc về lớp 2 tính từ dưới lên. Phần này cung cấp một thành phần quan trọng gọi là Dalvik Virtual Machine là một máy ảo Java đặt biệt, được thiết kế tối ưu cho Android.

Máy ảo Dalvik sử dụng các tính năng cốt lõi của Linux như quản lý bộ nhớ, đa luồng, mà thực chất là bên trong ngôn ngữ Java. Máy ảo Dalvik cho phép tất cả các ứng dụng Android chạy trong tiến trình riêng của nó.

Android Runtime cũng cung cấp bộ thư viện cốt lõi, cho phép các lập trình viên Android sử dụng để viết các ứng dụng Android.

d. Application Framework

Lớp Application Framework cung cấp nhiều dịch vụ cấp cao hơn cho các ứng dụng trong các class Java. Các lập trình viên cũng được phép sử dụng các dịch vụ này trong các ứng dụng của họ.

e. Applications

Ở đây sẽ thấy tất cả các ứng dụng Android ở lớp trên cùng. Ứng dụng được lập trình sẽ được cài đặt vào lớp này.

1.5.7 Các thành phần cơ bản của một ứng dụng android

Thành phần ứng dụng là khối kiến trúc cơ bản của ứng dụng Android. Các thành phần này được liên kết lồng lểo bởi các tập tin kê khai ứng dụng:

AndroidManifest.xml, mô tả mỗi thành phần của ứng dụng và cách chúng tương tác.

Bốn thành phần chính được sử dụng trong ứng dụng Android:

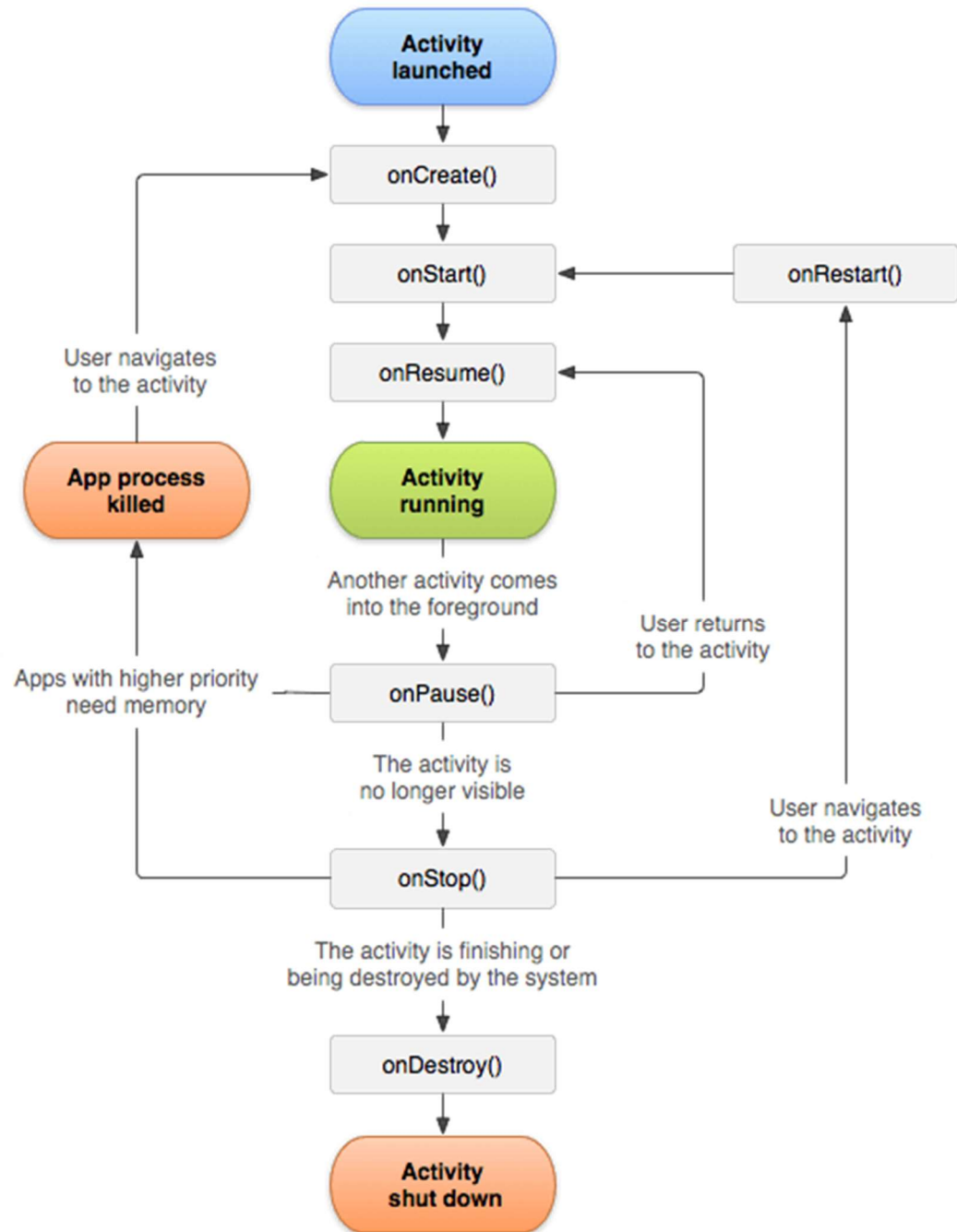
- 1. Activities:** Ra lệnh cho giao diện và xử lý tương tác của người dùng với màn hình smartPhone.
- 2. Services:** Là thành phần chạy phía sau, chạy bên trong của mỗi ứng dụng.
- 3. Broadcast Receivers:** Xử lý việc giao tiếp giữa HĐH Android và ứng dụng.
- 4. Content Providers:** Xử lý dữ liệu và các vấn đề về quản lý cơ sở dữ liệu.

a. Activity

Một Activity đại diện cho một màn hình với một giao diện người dùng.

Ví Dụ: Một ứng dụng email sẽ có một Activity để hiển thị tất cả các email, một Activity khác để soạn email, một Activity khác để đọc email. Nếu một ứng dụng có nhiều hơn 1 Activity thì một trong số chúng sẽ được đánh dấu là Activity hiển thị đầu tiên khi ứng dụng khởi chạy.

Android khởi đầu bởi một activity được đánh dấu là khởi đầu bằng cách gọi phương thức `onCreate()`: . Có một chuỗi các phương thức được gọi để bắt đầu một activity và một chuỗi các phương thức khác để hủy một activity, như hình bên dưới:



Hình 1.2: Vòng đời của một Activity

Class Activity định nghĩa nhiều phương thức cho các sự kiện, không cần phải thực thi tất cả các phương thức này, tuy nhiên việc hiểu rõ và áp dụng chúng vào ứng dụng khá quan trọng, giúp cho ứng dụng có các hành vi như mong đợi của người dùng.

- `onCreate()`: : Phương thức đầu tiên được gọi dùng để tạo một activity vào lần đầu tiên Activity được gọi.
- `onStart()`: : Thực hiện khi nó hiện hữu với người dùng.

- `onResume()`: : Thực hiện khi người dùng tương tác với các ứng dụng.
- `onPause()`: : Tạm dừng một Activity, không nhận dữ liệu do người dùng nhập vào và không thể thực thi lệnh nào. Phương thức này được gọi khi activity hiện tại đang được tạm dừng, và activity trước đó đang được tiếp tục.
- `onStop()`: : Thực hiện một activity đã không được nhìn thấy trong thời gian dài.
- `onDestroy()`: : Thực hiện trước khi hệ thống hủy activity.
- `onRestart()`: : Thực hiện khi activity cần được dùng trở lại sau khi bị gọi `onStop()`: .

b. Services

Một Service là một thành phần được chạy bên trong nền để xử lý các công việc trong thời gian dài. Một ứng dụng nghe nhạc có thể phát nhạc, trong khi đó người dùng đang ở giao diện của ứng dụng khác. Hoặc ứng dụng tải tập tin có thể tải dữ liệu trên mạng về máy mà không ngăn chặn người dùng tương tác với các ứng dụng khác.

c. Broadcast Receivers

Broadcast Receivers chỉ đơn giản là xử lý và phát các thông điệp từ các ứng dụng khác hoặc từ hệ thống. Ví dụ: các ứng dụng có thể bắt đầu phát thông điệp đến các ứng dụng khác để cho biết rằng một số dữ liệu đã được tải thành công xuống thiết bị và sẵn sàng cho việc sử dụng, Broadcast Receivers sẽ đảm nhận việc thông báo vào đưa ra những hành động thích hợp.

Một Broadcast Receiver được thực thi như là một class con của class Broadcast Receiver và mỗi thông điệp là một Intent.

d. Content Providers

Một Content Providers chứa các thành phần cung cấp dữ liệu từ một ứng dụng đến một ứng dụng khác theo yêu cầu. Yêu cầu đó được xử lý bằng các phương thức của class ContentResolver. Dữ liệu được lưu trữ trong file hệ thống, trong cơ sở dữ liệu hoặc ở một nơi nào khác.

Một ContentProviders được thực thi như là class con của class ContentProvider và phải thực thi theo tiêu chuẩn trong bộ các API để cho các ứng dụng khác có thể giao tiếp.

1.5.8 Các tập tin quan trọng trong Android

a. Tập tin MainActivity.java

```
package com.example.helloworld;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
public class MainActivity extends Activity {
```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
}

```

R.layout.activity_main: trỏ đến file activity_main.xml được đặt trong thư mục res/layout.

b. AndroidManifest.xml

Bất cứ thành phần nào được triển khai như một phần trong ứng dụng, phải được khai báo trong tập tin này. File này giống như một interface giữa HĐH Android và ứng dụng. Vì vậy nếu thành phần nào dùng mà không khai báo trong file này, thì HĐH sẽ không xem xét đến chúng.

Đây là code mặc định của AndroidManifest.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloworld">
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.helloworld.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>

```

`<application>...</application>`: Thẻ bao quanh các thành phần có liên quan trong ứng dụng. Thuộc tính `Android:icon`: Sẽ trỏ đến các icon có sẵn trong thư mục `res/drawable-hdpi`. Ứng dụng này sẽ sử dụng hình `ic_launcher` là icon.

`<activity>...</activity>`: Thẻ này được sử dụng để nêu chi tiết về một activity. Thuộc tính `android:name` được sử dụng để mô tả chi tiết về tên class của một activity. Thuộc tính `android:label` xác định một chuỗi ký tự được sử dụng là nhãn cho activity đó. Nếu trong ứng dụng có nhiều Activities thì phải tạo nhiều thẻ `<activity>`.

`<action/>`: với name `android.intent.action.MAIN` để cho Android biết rằng activity này sẽ được load lên đầu tiên khi chạy ứng dụng.

`<category/>`: với name `android.intent.category.LAUNCHER` để cho Android biết ứng dụng có thể được chạy từ một Icon trên thiết bị.

`@string`: sẽ được chỉ định đến file `strings.xml`, `@string/app_name` sẽ được dẫn đến chuỗi `app_name` nó là `HelloWord`.

Sau đây là một số thẻ khác thẻ sử dụng trong file `AndroidManifest.xml`.

`<activity>` là thẻ dành cho các activities.

`<service>` là thẻ dành cho các services.

`<receiver>` là thẻ dành cho các broadcast receivers.

`<provider>` là thẻ dành cho các content providers.

c. R.java

R.java tạo ra mối liên kết giữa các file activity và các file resource (VD: như giữa file `MainActivity.java` và file `activity_main.xml`). File này được tạo ra tự động và không nên thay đổi nội dung trong file này.

d. activity_main.xml

`activity_main.xml` là một file layout nằm trong thư mục `res/layout`. File này sẽ được tham chiếu bởi ứng dụng khi tạo ra giao diện trên màn hình. Sẽ phải thay đổi nội dung file này rất nhiều trong khi tạo giao diện cho ứng dụng.

Đây code mặc định của file `activity_main.xml`.

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```

        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        tools:context=".MainActivity" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/hello_world" />
    </RelativeLayout>

```

RelativeLayout: là một trong số nhiều layout của Android.

1.5.9 Các giao diện cơ bản trong Android

Giới thiệu các layout cơ bản trong android

a. **FrameLayout:**

- Là loại Layout cơ bản nhất, đặc điểm của nó là khi gắn các control lên giao diện thì các control này sẽ luôn được “Neo” ở góc trái trên cùng màn hình, nó không cho phép chúng ta thay đổi vị trí của các control theo một Location nào đó.

- Các control đưa vào sau nó sẽ đè lên trên và che khuất control trước đó (trừ khi ta thiết lập transparent cho control sau):

- Theo dõi đoạn cấu trúc XML dưới này:

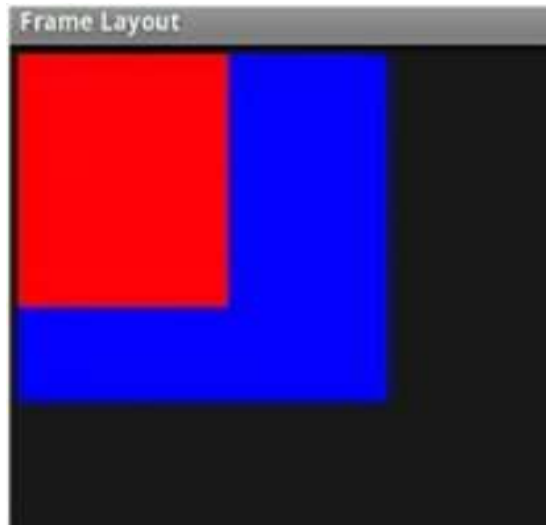
```

<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    android:id="@+id/mainlayout"
    android:layout_height="fill_parent"
    android:layout_width="fill_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android";
    <ImageView
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:padding="5px"
        android:src="@drawable/blue"/>
    <ImageView
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"

```



```
android:padding="5px"  
android:src="@drawable/red"/>  
</FrameLayout>
```



Hình 1.3: Minh họa FrameLayout

Hình màu đỏ và màu xanh luôn được “neo” ở góc trái màn hình. Hình màu đỏ đưa vào sau sẽ đè lên trên hình màu xanh.

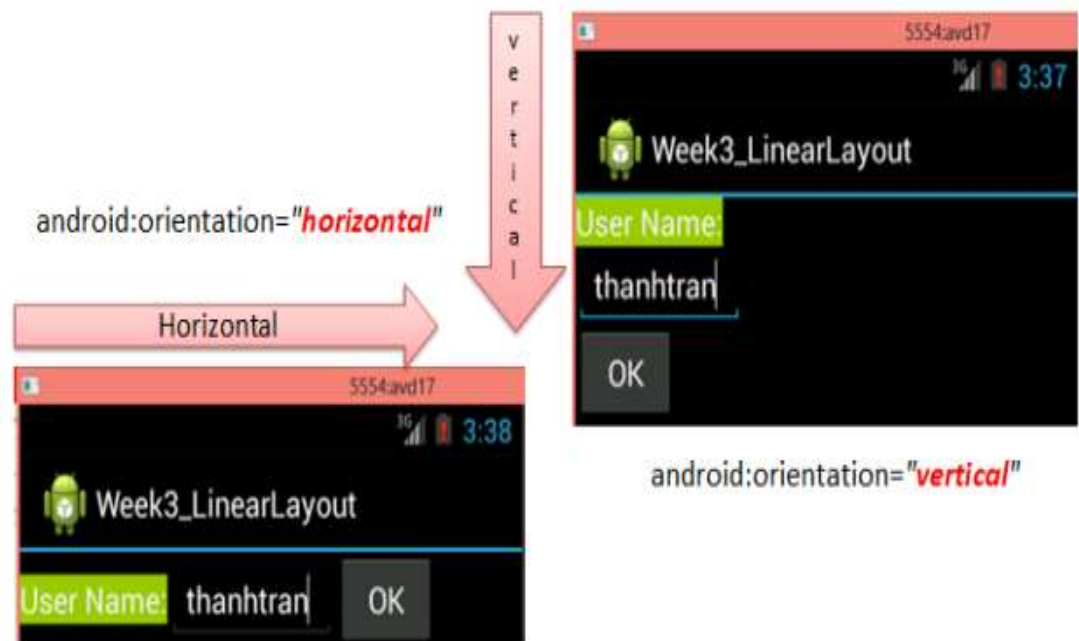
Chú ý 2 dòng lệnh bên dưới:

```
android:src="@drawable/blue"  
android:src="@drawable/red"
```

Là do ta kéo 2 cái hình tên là **blue** và **red** vào thư mục drawable của ứng dụng

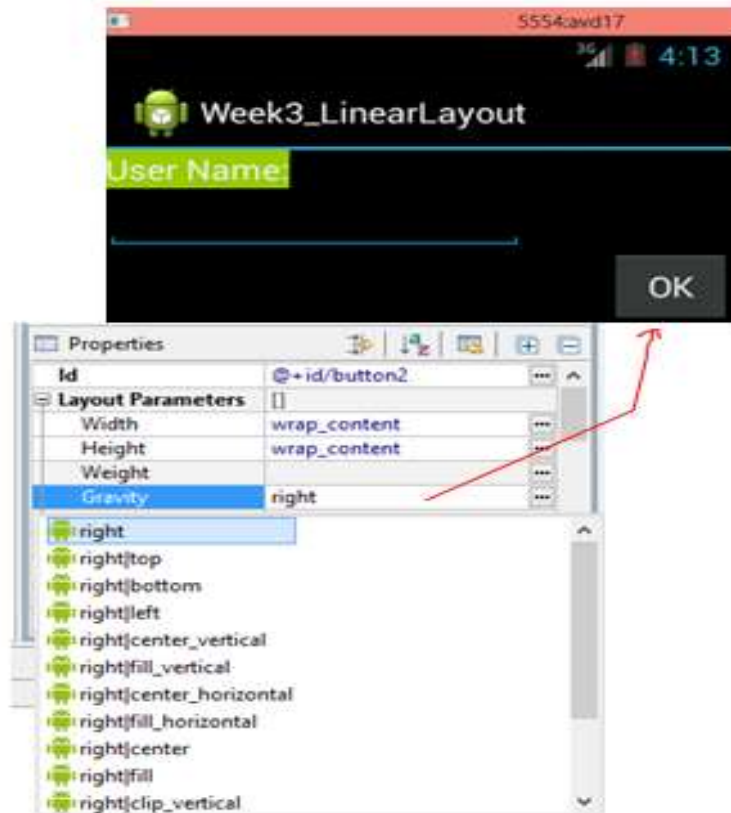
b. LinearLayout

- Layout này cho phép sắp xếp các control theo hai hướng trên giao diện: Hướng từ trái qua phải và hướng từ trên xuống dưới.



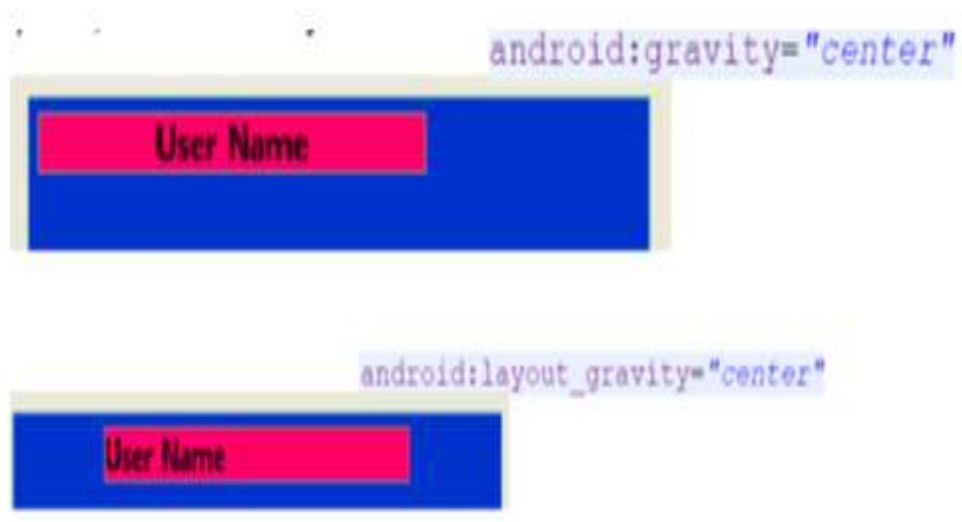
Hình 1.4: Minh họa LinearLayout

- Có thể dùng margin, gravity, weight để hỗ trợ cho việc thiết kế.
- Ta có thể dùng Properties hỗ trợ sẵn trong Eclipse để thiết lập các thuộc tính cho control:
- Ví dụ như để căn lề các control trên giao diện ta dùng layout_gravity:



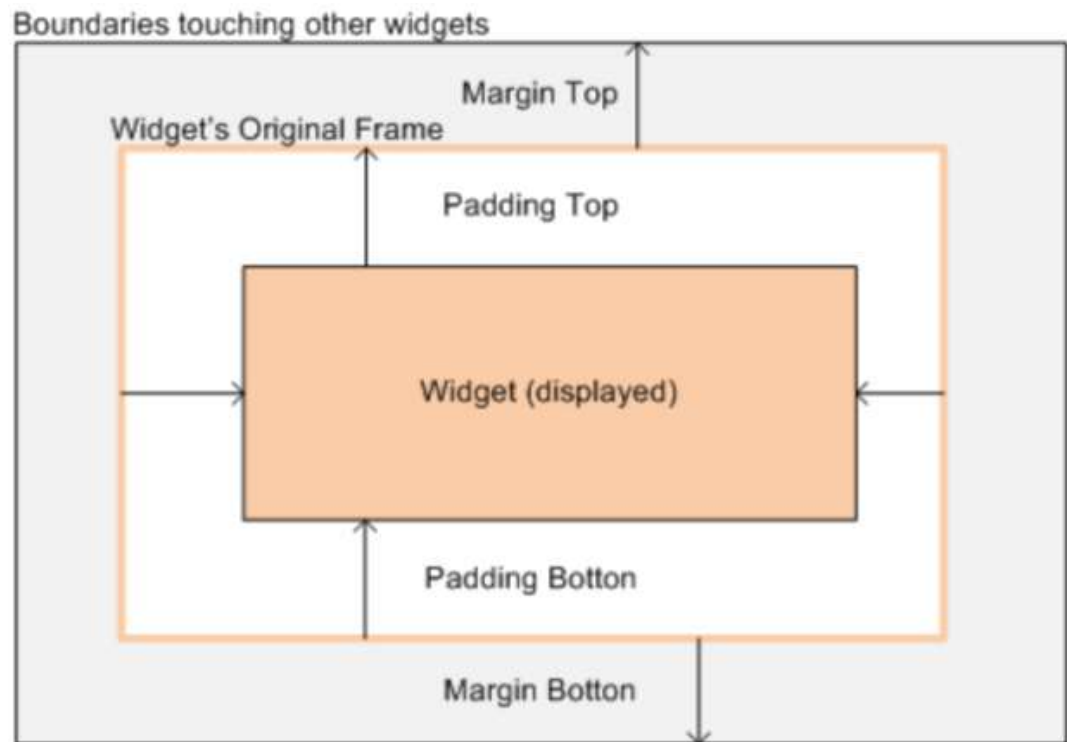
Hình 1.5: Minh họa sử dụng layout_gravity

- Hay để căn lề nội dung bên trong của control dùng **gravity**:



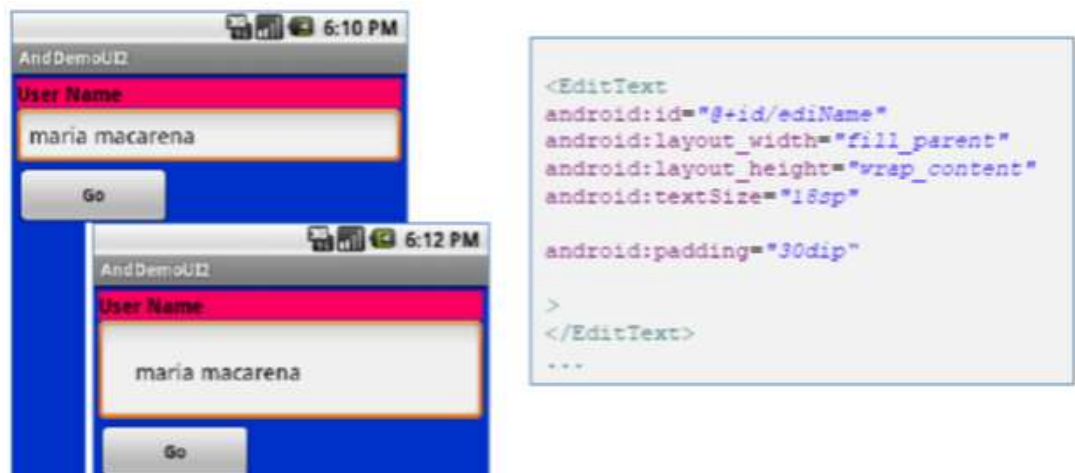
Hình 1.6: Minh họa sử dụng gravity

- So sánh sự khác biệt giữa Padding và Margin:



Hình 1.7: Minh họa chức năng của Padding và Margin

- Ví dụ thay đổi Padding (internal spacing – khoảng cách giữa nội dung bên trong so với đường viền của control):



Hình 1.8: Minh họa sử dụng Padding

- Ví dụ về đổi Margin (external spacing – khoảng cách giữa control này với control khác):



Hình 1.9 Minh họa sử dụng Margin

c. TableLayout

- Cho phép sắp các control theo dạng lưới (dòng và cột)
- TableLayout sẽ xem dòng nào có số lượng control nhiều nhất để xác định rằng nó có bao nhiêu cột (lấy dòng có số lượng control nhiều nhất làm số cột chuẩn).

0		1	
0		1	2
0	1	2	3

Hình 1.10: Minh họa khi sử dụng TableLayout

- Như vậy theo hình trên thì phải nói là TableLayout này có 4 cột, 3 dòng.
- Dùng **layout_span** để trộn các cột:

```

<TableRow>
    <TextView android:text="URL:" />
    <EditText
        android:id="@+id/entry"
        android:layout_span="3" />
</TableRow>

```

- Dùng **layout_column** để di chuyển vị trí của control đến một cột nào đó trên 1 dòng:



Hình 1.10: Minh họa sử dụng layout_column

- Dùng **stretchColumns** để dẫn đều các control, các cell (ta thường dùng dấu "*"):

```
<TableLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:stretchColumns="*"
>
```

d. RelativeLayout:

- RelativeLayout cho phép sắp xếp các control theo vị trí tương đối giữa các control khác trên giao diện (kể cả control chứa nó). Thường nó dựa vào Id của các control khác để sắp xếp theo vị trí tương đối. Do đó khi làm RelativeLayout phải chú ý là đặt Id control cho chuẩn xác, nếu sau khi Layout xong mà lại đổi Id của các control thì giao diện sẽ bị xáo trộn (do đó nếu đổi ID thì phải đổi luôn các tham chiếu khác sao cho khớp với Id mới đổi).

- Dưới đây là ví dụ về cách sử dụng RelativeLayout:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="#ff0000ff"
    android:padding="10px" >

    <TextView android:id="@+id/label"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="#ffff0077"
        android:text="Type here:" />

    <EditText android:id="@+id/entry"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/label" />

    <Button
        android:id="@+id/ok"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/entry"
        android:layout_alignParentRight="true"
        android:layout_marginLeft="10px"
        android:text="OK" />

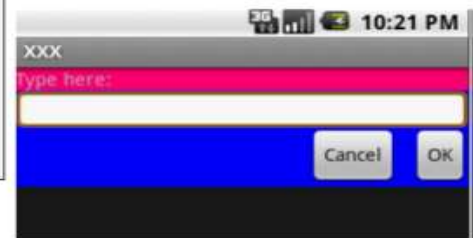
    <Button
        android:text="Cancel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toLeftOf="@+id/ok"
        android:layout_alignTop="@+id/ok" />

</RelativeLayout>
```

```
<Button
    android:id="@+id/ok"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/entry"
    android:layout_alignParentRight="true"
    android:layout_marginLeft="10px"
    android:text="OK" />

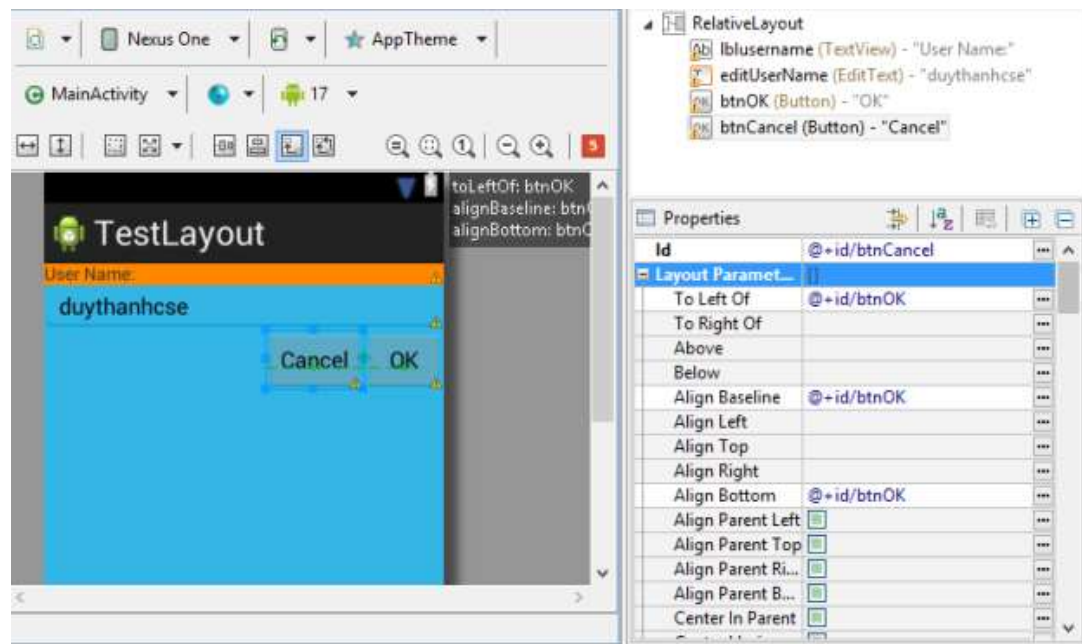
<Button
    android:text="Cancel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toLeftOf="@+id/ok"
    android:layout_alignTop="@+id/ok" />

</RelativeLayout>
```



Hình 1.11: Minh họa sử dụng RelativeLayout

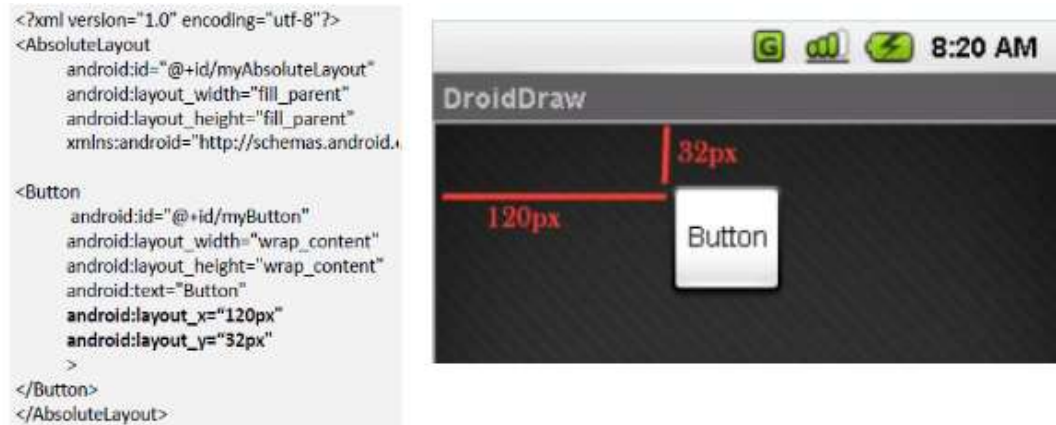
- Ta có thể sử dụng công cụ trong Eclipse để thiết kế :



Hình 1.12: Minh họa sử dụng RelativeLayout

e. **AbsoluteLayout:**

- Cho phép thiết lập các control giao diện theo vị trí tùy thích:



Hình 1.13: Minh họa sử dụng AbsoluteLayout

1.5.10 Giới thiệu một số widget cơ bản trong lập trình android

a. TextView

Nếu chỉ muốn hiển thị thông tin mà không cho phép người dùng chỉnh sửa thì nên sử dụng control này.

- TextView tương tự như JLabel bên Java, và như Label bên C#
- Dưới đây là một số thuộc tính của TextView mà chúng ta thường xuyên sử dụng nhất:

```
<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#ff0000ff"
    android:textStyle="bold"
    android:textSize="25sp"
    android:padding="5dp"
    android:fontFamily="tahoma"
    android:textColor="@android:color/holo_red_dark"
    android:text="Nhập tên:" />
```

- Ta nên thiết lập **id** cho control để tiện bề xử lý.
- layout_width, layout_height nên thiết lập cho điều khiển
- Để thay đổi màu nền dùng background, thay đổi màu chữ dùng textColor...
- Dựa vào Id ta sẽ lấy được control theo đúng Id này, xem code bên dưới để biết cách lấy control theo Id:

```
TextView txt1= (TextView) findViewById(R.id.textView1);
```


- Mọi control đều kế thừa từ View, và hàm **findViewById** cũng trả về 1 View theo đúng Id truyền vào, đó là lý do ta ép kiểu về cho đúng với TextView (cách làm nhanh: ngay dòng lệnh này nhấn tổ hợp phím **Ctrl + 1** tự ép kiểu nhanh.

- Để hiển thị thông tin lên control TextView ta dùng lệnh dưới đây:

```
txt1.setText("Hello world");
```

- Lấy lấy thông tin bên trong control TextView ta dùng lệnh dưới đây:

```
String msg = txt1.getText(): .toString(): ;
```

b. EditText

- Control này kế thừa từ TextView và cho phép chỉnh sửa dữ liệu.

- Để sử dụng EditText rất đơn giản, chỉ việc kéo thả control này vào giao diện và tiến hành thiết lập một số thuộc tính:

```
<EditText
    android:id="@+id/editText2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="text|textAutoCorrect|textCapWords"
    android:hint="Nhập tài khoản"
    android:textSize="18sp"
    android:ems="10" />
```

- Tương tự như TextView cần thiết lập Id, các layout_width, layout_height

- Thuộc tính hint : để hiển thị thông tin gợi ý trong vùng nhập dữ liệu khi chưa nhập bất kỳ dữ liệu nào vào, chỉ cần có dữ liệu là phần hint sẽ tự động mất đi.

- textSize để thiết lập kích cỡ font chữ cho EditText

- textAutoCorrect : Tự động sửa đúng chính tả, giả sử nhập “teh” thì nó sẽ tự động sửa thành “the”

- Tương tự như TextView, ta cũng phải lấy được control thông qua Id, thao tác với dữ liệu bên trong EditText:

- Lấy control theo Id:

```
EditText txtbox = (EditText) findViewById(R.id.editText1);
```

- Thiết lập giá trị cho EditText

```
txtBox.setText("nhập chữ")
```

- Lấy dữ liệu bên trong EditText:

```
String msg = txtBox.getText(): .toString():
```

c. Button

- Dùng để thiết lập sự kiện khi người dùng chọn lựa.

- Cũng kế thừa từ TextView

- Có 2 sự kiện mà người sử dụng thường xuyên thao tác:

```

Button btnok=(Button) findViewById(R.id.btnOk);
btnok.setOnClickListener(new View.OnClickListener() {
    public void onClick(View arg0) {
        //perform click here
    }
});
btnok.setOnLongClickListener(new View.OnLongClickListener() {
    public boolean onLongClick(View arg0) {
        //perform long click here
        return false;
    }
});

```

d. Checkbox, Radio Button

- CheckBox và RadioButton đều sử dụng chung 2 phương thức :

1) Phương thức `setChecked()`: , dùng để thiết lập checked. Nếu ta gọi `setChecked(true)` tức là cho phép control được checked, còn gọi `setChecked(false)` thì control sẽ bị unchecked.

2) Phương thức `isChecked`, kiểm tra xem control có được checked hay không. Nếu có checked thì trả về true ngược lại trả về false

- Checkbox cho phép ta checked nhiều đối tượng, còn RadioButton thì tại một thời điểm nó chỉ cho ta checked 1 đối tượng trong cùng một group mà thôi.

- Tương tự như Checkbox, ta cũng có thể thiết lập checked cho RadioButton bất kỳ trong XML:

```

<RadioGroup
    android:id="@+id/radioGroup1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" >

    <RadioButton
        android:id="@+id/radio0"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Rất thích" />

    <RadioButton
        android:id="@+id/radio1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checked="true" ✓
        android:text="Hơi hơi thích" />
</RadioGroup>

```

- Nhìn vào hình trên thấy là ta phải sử dụng RadioGroup để gom nhóm các RadioButton lại cùng một nhóm nào đó, những RadioButton mà cùng một nhóm thì tại 1 thời điểm chỉ có 1 RadioButton được checked mà thôi. Trong một màn hình ta có thể tạo nhiều nhóm RadioGroup khác nhau.

- Có 2 cách xử lý RadioButton nào được checked như sau:

Cách 1: Dựa vào RadioGroup để biết chính xác Id của RadioButton nào được checked. Dựa vào Id này ta sẽ xử lý đúng nghiệp vụ:

```
RadioGroup group=(RadioGroup) findViewById(R.id.radioGroup1);
int idChecked=group.getCheckedRadioButtonId();
switch(idChecked)
{
case R.id.radrathailong:
    break;
case R.id.radhailong:
    break;
case R.id.radtamchapnhan:
    break;
case R.id.radthayghe:
    break;
}
```

- Như hình trên, hàm getCheckedRadioButtonId(): : hàm này trả về Id của RadioButton nằm trong RadioGroup 1 được checked. Dựa vào Id này so sánh để biết được trên giao diện người sử dụng đang checked lựa chọn nào.

Cách 2: Kiểm tra trực tiếp RadioButton đó có được checked hay không?

```
RadioButton rad=(RadioButton) findViewById(R.id.radrathailong);
if(rad.isChecked())
{
}
}
```

Cả 2 cách trên đều có cùng chung một mục đích chỉ là kỹ thuật xử lý khác nhau, tương tự để xóa bỏ checked trong group, ta dùng lệnh:

`group.clearChecked() ;`

với group là đối tượng RadioGroup.

Vẽ giao diện nâng cao và lập trình sự kiện trong android

Listview, Gridview và lập trình sự kiện

e. Xây dựng Listview

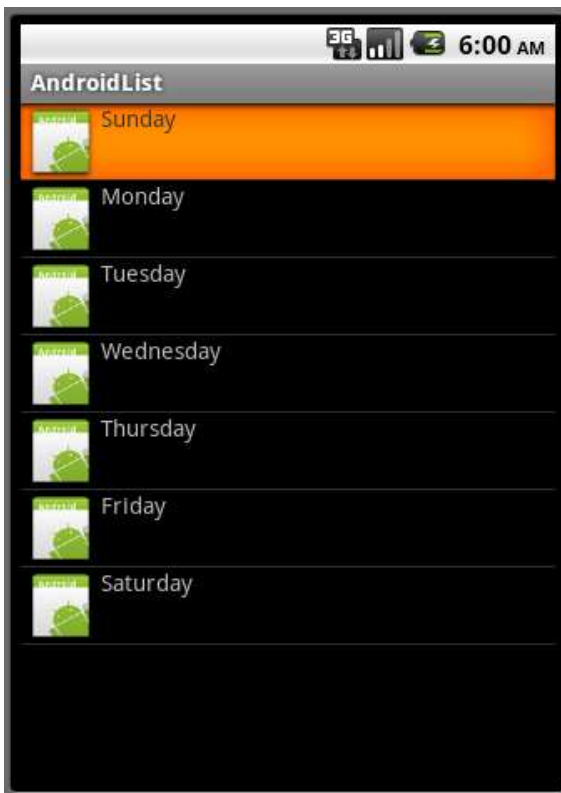
- Dùng hiển thị một danh sách dữ liệu.

- Thường là các view có cùng dạng layout đặt liền nhau.
- Dùng để hiển thị nhiều dữ liệu có cùng kiểu.
- Số lượng các row trong list được cố định bởi dữ liệu được truyền vào và loại thông tin thể hiện.
- Có các thuộc tính cơ bản của một View.
- Một số thuộc tính cơ bản:

Truyền dữ liệu vào listview với các view đã được khai báo: `setAdapter(adapter);`

Tạo adapter: `New ArrayAdapter<Object>;`

Hiện thị hay không hiển thị: `setVisibility(View.VISIBLE);`



Hình 1.14: Minh họa khi sử dụng ListView

f. Xây dựng GridView

Dùng để hiển thị một danh sách dữ liệu theo dạng hình lưới

Thường là các view có cùng dạng layout đặt liền nhau theo chiều ngang và chiều dọc

Dùng để hiển thị nhiều dữ liệu có cùng kiểu (thường được sử dụng để làm phần hiển thị chính của ứng dụng - Main)

Số lượng các row trong Grid được cố định bởi dữ liệu được truyền vào và loại thông tin thể hiện

Có đủ thuộc tính cơ bản của một View

- Một số thuộc tính cơ bản:

Truyền dữ liệu vào gridview với các view đã được khai báo

`setAdapter(adapter);`

Tạo adapter

`new ArrayAdapter<Object>;`

Hiện thị hay không hiển thị

`setVisibility(View.VISIBLE);`

....

g. Custom Adapter cho Listview và Gridview

Customize adapter cho phép chúng ta sắp xếp và tổ chức dữ liệu theo ý muốn.

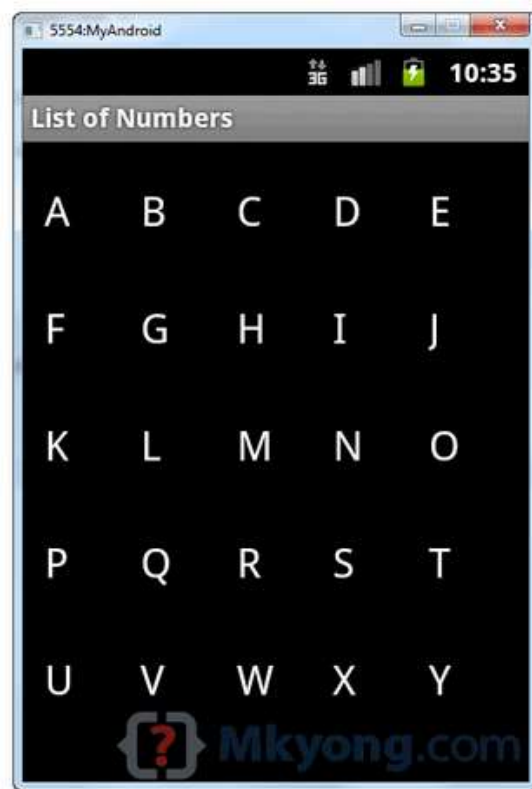
Cách thực hiện.

Thừa kế lại lớp BaseAdapter

public class AdapterCustom extends BaseAdapter

Override `getView` (hàm thực hiện customize)

public View getView(int arg0, View arg1, ViewGroup arg2)



Hình 1.15: Minh họa khi sử dụng Listview và Gridview

- Xử lý sự kiện trong ListView và GridView

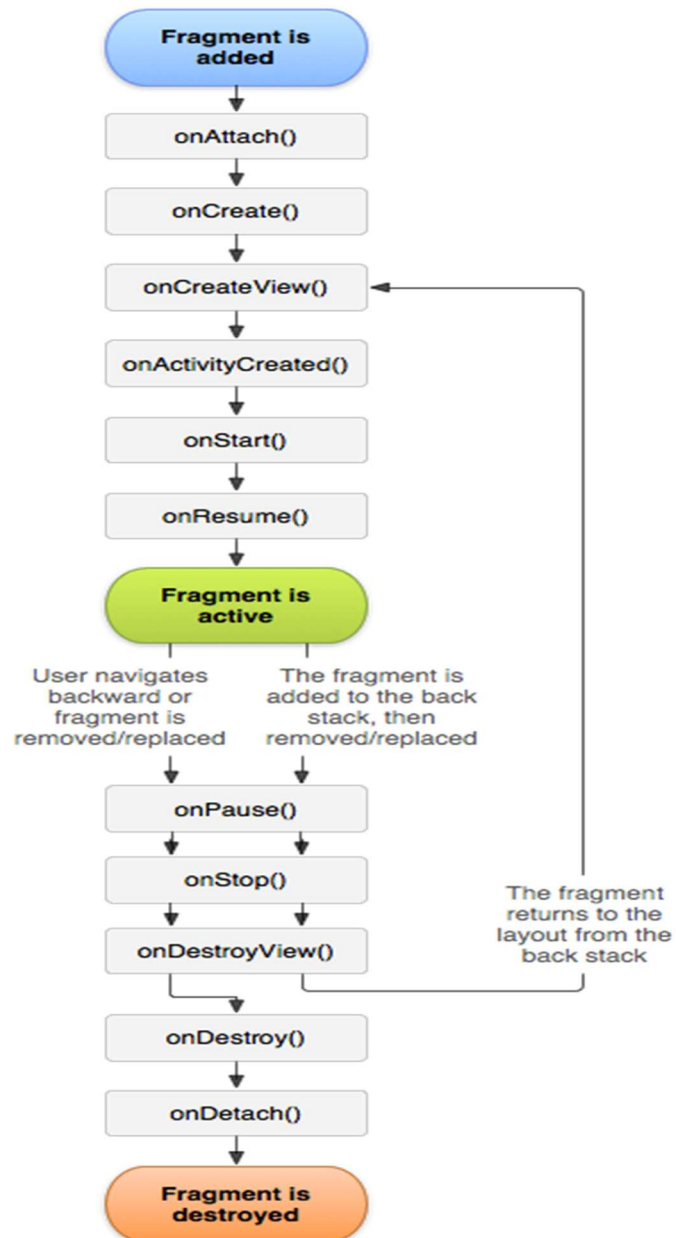
Có 2 sự kiện cơ bản mà ta thường dùng:

onItemClickListener

onListItemLongClick

h. Fragments

- Fragment là một control được nhúng trong một layout bất kỳ
- Fragment cũng có vòng đời sống như Activity
- Fragment dùng trong TH muốn thiết kế layout động
- Fragment Life cycle



Hình 1.16: Vòng đời của một Fragment

Sử dụng Fragment như thế nào

XML:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <fragment android:name="com.example.news.ArticleListFragment"
        android:id="@+id/list"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="match_parent" />
    <fragment android:name="com.example.news.ArticleReaderFragment"
        android:id="@+id/viewer"
        android:layout_weight="2"
        android:layout_width="0dp"
        android:layout_height="match_parent" />
</LinearLayout>
```

Code file .java

```
public static class ExampleFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
        Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false);
    }
}
```

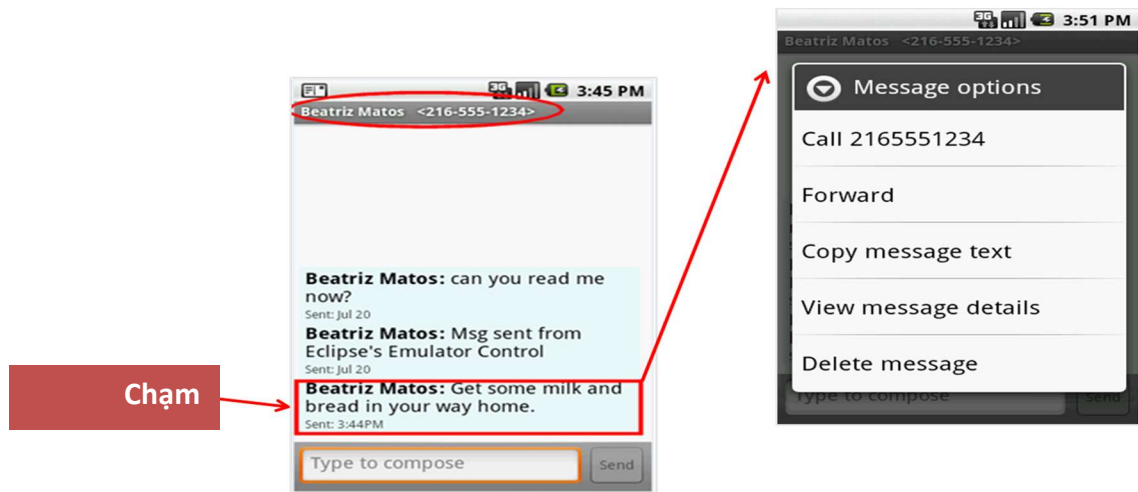
Thay giao diện mới cho Fragment

```
// Create new fragment and transaction
Fragment newFragment = new ExampleFragment();
FragmentManager transaction = getFragmentManager().beginTransaction();

// Replace whatever is in the fragment_container view with this fragment,
// and add the transaction to the back stack
transaction.replace(R.id.fragment_container, newFragment);
transaction.addToBackStack(null);

// Commit the transaction
transaction.commit();
```

i. Option menu và Context menu



Hình 1.17: Minh họa khi sử dụng menu

Option và Context Menu có thể bao gồm :

- Text
- Icons
- Radio Buttons
- Check Boxes
- Sub-Menus
- Short-cut keys

Tạo option menu và context menu:

- Đăng ký activity có các Menu sử dụng phương thức:
`registerForContextMenu(theWidget)`.
- Hiện thực phương thức:
`onCreateContextMenu(...)`.
- Xử lý sự kiện khi click vào các Item trên Option và Context menu:

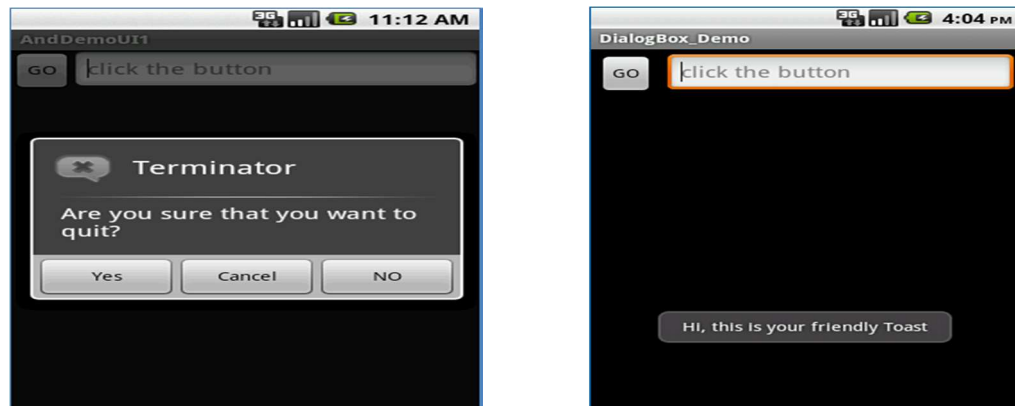
Option menu :

`onOptionsItemSelected()`:

Context menu :

`onContextItemSelected()`:

j. Dialog và Toast



Hình 1.18: Minh họa khi sử dụng Dialog và Toast

AlertDialog là màn hình:

- Hiện thị các thông tin tóm tắt trên 1 cửa sổ nhỏ.
- Khẳng định 1 tác vụ.
- Toast là một UI trong suốt, chỉ chứa thông điệp hiển thị cho người dùng biết.
- Không bao giờ focus đến UI này được.
- Giới thiệu và thiết kế giao diện theo phong cách Material Design

- Giới thiệu.

Material Design là một phong cách thiết kế mới được Google giới thiệu cùng lúc với phiên bản Android 5.0 Lollipop trở lên.

Phong cách thiết kế Material Design nhắm đến những đường nét đơn giản, sử dụng nhiều mảng màu đậm nổi bật, các đối tượng đồ họa trong giao diện dường như: “trôi nổi” lên. Ngoài ra, nó còn bao gồm cả những hiệu ứng chuyển động tự nhiên khi các nút, menu hiện diện trên màn hình. Tất cả đều nhằm mang lại cho người dùng trải nghiệm mới mẻ hơn, thú vị hơn và gần giống đời thực hơn.

- Đặc điểm:

- Sử dụng các màu nổi bật, thường có một mảng màu chủ đạo nằm ở cạnh trên ứng dụng
- Các biểu tượng phẳng, đơn giản nhưng dễ hiểu
- Một số ứng dụng sẽ có một nút tròn to nằm ở góc dưới bên phải, thường có chức năng tạo mới
- Giao diện phẳng, ít hoặc không có hiệu ứng chuyển màu, có hoặc không có hiệu ứng đổ bóng đen
- Menu, nút nhấn, chữ viết... có nhiều khoảng cách trắng nên trông thoáng đãng
- Có các hiệu ứng chuyển động tự nhiên, dễ hiểu, có thể gợi ý cho một tính năng nào đó

1.5.11 Các thành phần được thiết kế trong Material Design

a. ActionBar/ToolBar



Hình 1.20: Minh họa khi sử dụng Dialog và Toast

Activity phải kế thừa ActionBarActivity.

Sử dụng các theme của Theme.AppCompat như là:

Màu sáng:

```
<activity android:theme="@style/Theme.AppCompat.Light" ... >
```

Màu tối:

```
Theme.AppCompat.Light.DarkActionBar
```

b. ViewPager

ViewPager là một đối tượng khá giống như Slide trình diễn của MS PowerPoint.

ViewPager có thể trượt chuyển đổi giữa các giao diện một cách nhẹ nhàng và mượt, thay vì chuyển đổi màn hình qua một sự kiện chớp đen như trên tivi. Màn hình hiển thị trước nó hoặc sau nó sẽ được hiển thị ra ngay tức thì liền với nó. ViewPager hỗ trợ từ Android API 13 trở lên.

ViewPager không phải là một View chuẩn của Android, mà là một thành phần nằm trong gói android.support.design:23.4.0

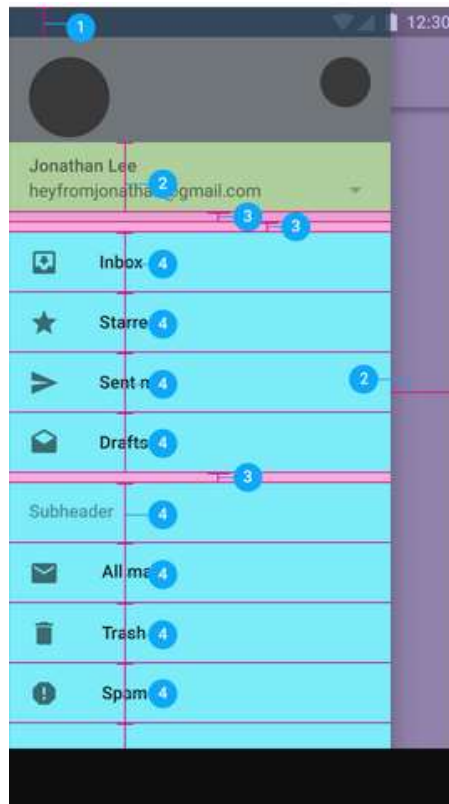
c. Buttons: Floating Action Button

Floating Action Button (FAB) là một thành phần mới được giới thiệu trong Material Design guideline, nó thiết kế cho việc nhấn mạnh cho hành động chính của một màn hình. Đây là một thiết kế phá cách làm nổi bật lên để thu hút người dùng chú ý vào hành động đó.

Có rất nhiều ý kiến xung quanh thiết kế của thành phần này. Điều quan trọng là sử dụng nó trong trường hợp thật sự cần thiết, chẳng hạn như thể hiện các hành động xảy ra thường xuyên (thường là hiển thị Activity, hoặc những điều có ảnh hưởng đến trải nghiệm người dùng). Tuy nhiên, cũng có ý kiến trái chiều cho rằng Floating Action Button là một ý tưởng tồi về mặt trải nghiệm người dùng.

d. Navigation Drawer (Slider menu)

Navigation Drawer là một bản điều hướng xuất hiện khi vuốt trên màn hình từ cạnh trái sang cạnh phải hay từ cạnh phải sang trái. Ngoài ra Navigation Drawer còn có thể xuất hiện khi chạm vào một biểu tượng trên thanh ActionBar. Navigation Drawer thường thấy xuất hiện trong rất nhiều app hay xử dụng như Gmail, Facebook...



Hình 1.21: Minh họa khi sử dụng Navigation Drawer (Slider menu)

Tham khảo thêm chi tiết tại:

<https://www.google.com/design/spec/patterns/navigation-drawer.html>

1.5.12 Lưu trữ dữ liệu trong Android

a. Share Preferences

Android cung cấp sẵn một cơ chế đơn giản giúp chúng ta lưu trữ nhanh các dữ liệu ngắn như cấu hình ứng dụng, tên đăng nhập, email... và lấy lại dữ liệu đã ghi này trong các lần chạy ứng dụng tiếp theo.

Cách thức thuận tiện nhất là mô tả tập trung các cấu hình ta cần lưu lại trong một Activity (thường là màn hình “*Settings*” của ứng dụng).

- References – cặp giá trị <key – value>.

Key: String

Value: primitive data type

- Một số hàm của References:

`getPreferences()`:

`getSharedPreferences()`:

`getDefaultSharedPreferences()`:

- Các hàm được sử dụng để lưu trữ, lấy dữ liệu: `putXxx...`, `getXxx...`

Hàm `put`.

Hàm `get`.

Xxx: là các kiểu { *Long, Int, Double, Boolean, String* }.

Xxx = {*Long, Int, Double, Boolean, String* }

Key: String.

1.6 Google Map API trên Hệ điều hành Android

1.6.1 Google Maps

Google Maps được xem là dịch vụ bản đồ số được google phát triển với mục đích thay thế cho các loại bản đồ giấy thông thường. Giờ đây chỉ bằng chiếc điện thoại thông minh nhỏ gọn có thể tự do lựa chọn những địa điểm mà mình muốn đến và nhanh chóng tiếp cận được những dịch vụ xung quanh các địa điểm đó.

Để sử dụng Google map một cách chính xác nhất cần GPS- hệ thống định vị toàn cầu giúp có thể biết rõ vị trí hiện tại của bản thân và thông qua GPS con người có thể dễ dàng xác định được phương hướng và đường đi một cách nhanh chóng nhất có thể.

1.6.2 Các loại bản đồ Google Maps

Google Maps cung cấp 5 loại bản đồ như sau:

`int MAP_TYPE_HYBRID` Bản đồ vệ tinh với một lớp trong suốt của các đường phố lớn.

`int MAP_TYPE_NONE` Không có bản đồ nền.

`int MAP_TYPE_NORMAL` Bản đồ cơ bản.

`int MAP_TYPE_SATELLITE` Bản đồ vệ tinh không có nhãn.

`int MAP_TYPE_TERRAIN` Bản đồ địa hình.

1.6.3 Ứng dụng Google Maps trên nền tảng Android.

Không thể tạo ra một đối tượng GoogleMap trực tiếp mà phải bằng cách lấy một từ phương thức `getMapAsync ()` trên `MapFragment` hoặc `MapView` mà bạn đã thêm vào ứng dụng của mình.

Lưu ý: Tương tự đối tượng `Xem`, `GoogleMap` chỉ có thể đọc và sửa đổi từ chủ đề Giao diện người dùng Android. Gọi các phương thức `GoogleMap` từ một chủ đề khác sẽ dẫn đến một ngoại lệ.

Bạn có thể điều chỉnh quan điểm của bản đồ bằng cách thay đổi vị trí xem bản đồ (trái ngược với di chuyển bản đồ). Bạn có thể sử dụng chế độ xem của bản đồ để đặt các thông số như vị trí, mức độ phóng, góc độ nghiêng, và mang.

1.6.4 Các phương thức của Google Maps

`final Circle addCircle(CircleOptions options):` Thêm một vòng tròn vào bản đồ.

`final Marker addMarker(MarkerOptions options):` Thêm điểm đánh dấu vào bản đồ.

`final Polygon addPolygon(PolygonOptions options):` Thêm đa giác vào bản đồ.

`Final Polyline addPolyline(PolylineOptions options):` Thêm một polyline vào bản đồ..

`final void animateCamera(CameraUpdate update):` Hoạt ảnh chuyển động của máy ảnh từ vị trí hiện tại đến vị trí được định nghĩa trong bản cập nhật.

`final void clear():` Xóa tất cả các điểm đánh dấu, đa giác, đa giác, lớp phủ, v.v ... khỏi bản đồ.

`final int getMapType():` Lấy loại bản đồ hiện đang được hiển thị.

`float final getMaxZoomLevel ():` Trả lại mức thu phóng tối đa cho vị trí hiện tại của máy ảnh.

`float final getMinZoomLevel ():` Trả lại mức thu phóng tối thiểu.

`final Location getLocation():` Lấy vị trí của bạn trên bản đồ.

`final boolean isMyLocationEnabled()` Lấy trạng thái của lớp vị trí của bạn.

`final void moveCamera(CameraUpdate update):` Thay đổi vị trí máy ảnh theo hướng dẫn được định nghĩa trong bản cập nhật.

`boolean setMapStyle(MapStyleOptions style):` Thiết lập kiểu dáng của bản đồ cơ sở.

`final void setMapType(int type):` Đặt kiểu bản đồ cần được hiển thị.

final void stopAnimation(): Ngừng hiệu ứng của ảnh nếu có một trong tiến trình.

.....

1.6.5 Marker trong Google Maps.

a. Chức Năng

- Vẽ các đối tượng trên bản đồ và gắn chúng vào một vĩ độ và kinh độ.
- Google Maps cung cấp các marker (dấu) để đánh dấu một địa điểm duy nhất trên bản đồ.
- Sử dụng các biểu tượng mặc định hoặc hoặc tùy chỉnh thành các icon theo ý muốn.

b. Các thuộc tính của Marker

- Độ mờ

Đặt độ mờ đục của điểm đánh dấu. Mặc định là 1.0.

- Vị trí của icon hiển thị

Điểm trên hình ảnh sẽ được đặt ở vị trí LatLng của điểm đánh dấu. Điều này mặc định là 50% từ bên trái của hình ảnh và ở dưới cùng của hình ảnh.

- Vị trí tọa độ trên Google Map

Giá trị LatLng cho vị trí của điểm đánh dấu trên bản đồ. Bạn có thể thay đổi giá trị này bất kỳ lúc nào nếu bạn muốn di chuyển điểm đánh dấu.

- Tên tiêu đề

Một chuỗi văn bản được hiển thị trong cửa sổ thông tin khi người dùng chạm vào điểm đánh dấu. Bạn có thể thay đổi giá trị này bất kỳ lúc nào.

- Snippet

Văn bản bổ sung được hiển thị dưới tiêu đề. Bạn có thể thay đổi giá trị này bất kỳ lúc nào.

- Biểu tượng

Một bitmap được hiển thị cho điểm đánh dấu. Nếu biểu tượng không được đặt, biểu tượng mặc định sẽ được hiển thị. Bạn có thể chỉ định một màu thay thế của biểu tượng mặc định bằng cách sử dụng defaultMarker (float).

- Thay đổi trạng thái

Nếu bạn muốn cho phép người dùng kéo marker, thiết lập thuộc tính này là true. Bạn có thể thay đổi giá trị này bất kỳ lúc nào. Mặc định này sai.

- Khả năng hiển thị

Theo mặc định, điểm đánh dấu là hiển thị. Để làm cho các điểm đánh dấu vô hình, thiết lập tài sản này để sai. Bạn có thể thay đổi giá trị này bất kỳ lúc nào.

- Flat hoặc Billboard

Nếu điểm đánh dấu bằng phẳng so với bản đồ, nó sẽ vẫn bị kẹt vào bản đồ khi máy quay xoay và nghiêng nhưng vẫn giữ được kích thước giống như máy ảnh phóng to, không giống như GroundOverlay. Nếu điểm đánh dấu là bảng quảng cáo, nó sẽ luôn luôn được vẽ phải đối mặt với máy ảnh và sẽ xoay và nghiêng với máy ảnh. Mặc định là bảng quảng cáo (sai)

- Quay

Vòng quay của điểm đánh dấu theo chiều kim đồng hồ về điểm neo của điểm đánh dấu. Trục quay xoay vuông góc với điểm đánh dấu. Vòng quay của 0 tương ứng với vị trí mặc định của điểm đánh dấu. Khi điểm đánh dấu bằng phẳng trên bản đồ, vị trí mặc định là Bắc được căn lề và vòng quay sao cho điểm đánh dấu vẫn giữ nguyên trên bản đồ. Khi điểm đánh dấu là bảng hiệu, vị trí mặc định chỉ lên và xoay vòng sao cho điểm đánh dấu luôn đối mặt với máy ảnh. Giá trị mặc định là 0.

- zIndex

Lệnh vẽ cho điểm đánh dấu. Các dấu được vẽ theo thứ tự của zIndex, với điểm đánh dấu zIndex cao nhất được vẽ trên đầu trang. Bằng cách thiết lập thuộc tính zIndex cho mỗi điểm đánh dấu, bạn có thể kiểm soát xem mục tiêu nào mà người dùng của bạn có nhiều khả năng đạt được. Giá trị mặc định là 0.

- Nhãn

Một đối tượng kết hợp với điểm đánh dấu. Ví dụ, đối tượng có thể chứa dữ liệu về những gì các điểm đánh dấu đại diện. Điều này dễ dàng hơn việc lưu trữ một Map riêng biệt <Marker, Object>. Ví dụ khác, bạn có thể liên kết một ID Chuỗi tương ứng với ID từ một bộ dữ liệu. Google Maps Android API không đọc và cũng không viết thuộc tính này.

Các phương pháp trong lớp này phải được gọi trên chủ đề Giao diện người dùng Android. Nếu không, một IllegalStateException sẽ bị ném ra khi chạy.

Ví dụ:

```
GoogleMap map = ... // get a map.

// Add a marker at San Francisco.
Marker marker = map.addMarker(new MarkerOptions()
    .position(new LatLng(37.7750, 122.4183))
    .title("San Francisco")
    .snippet("Population: 776733"));
```

c. Các phương thức của Marker

- float getAlpha () : Lấy alpha của marker.
- String getId () : Lấy mã này của điểm đánh dấu.LatLng
- getPosition () : Trả về vị trí của marker.
- float getRotation () : Nhận được vòng quay của marker.
- String getSnippet () : Lấy đoạn mã của marker.

- `Object getTag ()` : Lấy thẻ cho marker.
- `String getTitle ()` : Lấy tiêu đề của marker.
- `float getZIndex ()` : Trả về zIndex của marker.
- `void hideInfoWindow ()` : Ẩn cửa sổ thông tin nếu nó được hiển thị từ điểm đánh dấu này.
- `boolean isDraggable ()` : Nhận được khả năng kéo của điểm đánh dấu.
- `boolean isFlat ()` : Lấy vị trí phẳng của Marker.
- `boolean isInfoWindowShown ()` : Trả lại cho dù cửa sổ thông tin hiện đang hiển thị ở trên điểm đánh dấu này.
- `boolean isVisible ()` : Lấy cài đặt hiển thị của điểm đánh dấu này.
- `void remove ()` : Loại bỏ điểm đánh dấu này khỏi bản đồ.
- `void setAlpha (float alpha)` Đặt alpha (opacity) của marker.
- `void setAnchor (float anchorU, float anchorV)` Thiết lập điểm neo cho marker.
- `void setDraggable (boolean draggable)` Thiết lập khả năng kéo của điểm đánh dấu.
- `void setFlat (boolean flat)` Đặt cho dù điểm đánh dấu này phải phẳng so với bản đồ đúng hay bảng hiệu billboard đối mặt với máy ảnh giả.
- `void setIcon (biểu tượng BitmapDescriptorDescriptor)` Thiết lập biểu tượng cho điểm đánh dấu.
- `void setInfoWindowAnchor (float anchorU, float anchorV)` Xác định điểm trong ảnh điểm đánh dấu để neo cửa sổ thông tin khi nó được hiển thị.
- `void setPosition (LatLng latlng)` Đặt vị trí của điểm đánh dấu.
- `void setRotation (float rotation)` Thiết lập quay của điểm đánh dấu theo chiều kim đồng hồ về điểm neo của điểm đánh dấu.
- `void setSnippet (String snippet)` Thiết lập đoạn mã của marker.
- `void setTag (Object tag)` Thiết lập tag cho marker.
- `void setTitle (String title)` Thiết lập tiêu đề của marker.
- `void setVisible (boolean visible)` Thiết lập khả năng hiển thị của điểm đánh dấu này.
- `void setZIndex (float zIndex)` Thiết lập zIndex của marker.
- Hiển thị cửa sổ thông tin của điểm đánh dấu này trên bản đồ, nếu điểm đánh dấu này là `Visible ()`;
-

1.7 Công nghệ Nodejs

1.7.1 Tổng quan

NodeJS là ngôn ngữ lập trình mã nguồn mở, ngôn ngữ lập trình phía server, bắt đầu được phát triển từ năm 2009, là nền tảng cho việc xây dựng các ứng dụng web. Mặc dù NodeJS không phải là một JavaScript framework nhưng hầu hết các module của nó được viết bằng JavaScript.

Không những có thể sử dụng nó cho các ứng dụng phía client mà Nodejs có thể sử dụng nó để lập trình phía server.

1.7.2 Điểm nổi bật của NodeJS so với các ngôn ngữ lập trình khác

Hoạt động với một luồng duy nhất và có khả năng asynchronous (bất đồng bộ). Không giống như server được viết bằng PHP thì mỗi ông request đến server thì server sẽ tạo ra một thread để xử lý trong khi đó server Node xử lý mọi hành động trong một thread duy nhất.

1.7.3 Đặc điểm của Nodejs

- Không đồng bộ và Phát sinh sự kiện (Event Driven): Tất cả các APIs của thư viện Node.js đều không đồng bộ, nghĩa là không blocking (khóa). Nó rất cần thiết vì Node.js không bao giờ đợi một API trả về dữ liệu. Server chuyển sang một API sau khi gọi nó và có cơ chế thông báo về Sự kiện của Node.js giúp Server nhận đưa phản hồi từ các API gọi trước đó.
- Chạy rất nhanh: Dựa trên V8 Javascript Engine của Google Chrome, thư viện Node.js rất nhanh trong các quá trình thực hiện code.
- Các tiến trình đơn giản nhưng hiệu năng cao: Node.js sử dụng một mô hình luồng đơn (single thread) với các sự kiện lập. Các cơ chế sự kiện giúp Server trả lại các phản hồi với một cách không khóa và tạo cho Server hiệu quả cao ngược lại với các cách truyền thống tạo ra một số lượng luồng hữu hạn để quản lý request. Nodejs sử dụng các chương trình đơn luồng và các chương trình này cung cấp các dịch vụ cho số lượng request nhiều hơn so với các Server truyền thống như Apache HTTP Server.
- Không đệm: Ứng dụng Node.js không lưu trữ các dữ liệu buffer.
- Có giấy phép: Node.js được phát hành dựa vào MIT License.

1.7.4 Ưu điểm

- Tối ưu hóa thời gian thực hiện tiến trình
- Có khả năng mở rộng trong các ứng dụng web với nhiều hoạt động I/O liên tục.
- Phù hợp để xây dựng các ứng dụng web stream hay các game chơi trên nền web đảm bảo việc độ trễ thời gian xử lý hành động là nhỏ nhất.
- Dễ dàng để xây dựng các ứng dụng real-time.
- Cách viết ứng dụng với Node đó là các ứng dụng được cấu tạo từ các module nhỏ sau đó được kết hợp lại với nhau điều này đảm bảo cho việc sửa đổi bảo trì một cách nhanh chóng.

- Hiệu năng cao

1.7.5 Nhược điểm

- Bad concurrency
- Single - Threaded:
- Đơn luồng cũng có thể là một điểm kém của NodeJs. Lợi thế của single-thread là việc không gặp phải vấn đề synchronize giữa các tiến trình, không cần chia sẻ trạng thái hiện tại... Có thể dễ dàng viết một đoạn code mà sẽ ngốn của server khá nhiều thời gian để xử lý xong một vài trường hợp nào đó và có thể dẫn tới lock toàn bộ server.
- Lack of inherent code organization
- Sự thiếu tổ chức code đối với một ứng dụng NodeJs lớn là một vấn đề thực sự. Khi mà nguyên tắc asynchronise (bất đồng bộ) dẫn tới có nhiều cách để thực hiện code khác nhau giữa các thành viên hay có thể sử dụng nhiều design pattern khác nhau.
- Leak message

1.7.6 NodeJS core-module

Là những modules được biên dịch vào trong Node binary, đó là những modules khá quan trọng và được biên dịch kèm với Node khi cài đặt

- Cung cấp những functionalities cơ bản nhất của Node
- Gồm có: filesystem access, HTTP, HTTPS interfaces, và một số modules khác.
- Để sử dụng core module trong trường trình cần sử dụng phương thức require trong JavaScript file.
- Ví dụ muốn sử dụng module file system trong chương trình:

```
var fs = require('fs');
```

Khi module được require, Node sẽ tìm module đó trong thư mục core_modules. Sau đó ta có thể sử dụng được các phương thức mà module đó cung cấp.

1.7.7 Node.js third-party modules

Node.js third-party modules là bên thứ ba cung cấp các sản phẩm, dịch vụ sẵn có cho mình ở đây đó là module. Các module này không được cài đặt sẵn sàng cùng với node như các core module mình đã giới thiệu phía trên.

Để sử dụng third-party modules trước tiên cần cài đặt những modules muốn sử dụng thông qua 'npm' (node manage package).

Ví dụ muốn sử dụng 'express' module cần thực hiện lệnh sau trên terminal:

```
npm install express
```

Sau đó Node sẽ tải express module vào thư mục 'node_modules' dưới đường dẫn thư mục đang làm việc

Để sử dụng thì ta sẽ load module vào chương trình sử dụng require như bình thường.

Express module là một third-party rất quan trọng được sử dụng rất nhiều và nó cũng đem đến rất nhiều phương thức hữu ích giúp dễ dàng trong việc thực hiện các ứng dụng web.

1.8. Express framework trên NodeJS

1.8.1 Giới thiệu

Express là một web application framework for nodejs, nó cung cấp cho chúng ta những rất nhiều tính năng mạnh mẽ trên nền tảng web. Express rất dễ dàng để phát triển các ứng dụng nhanh dựa trên Node.js cho các ứng dụng Web. Express hỗ trợ các phương thức HTTP và middleware tạo ra 1 API rất mạnh mẽ và sử dụng dễ dàng hơn. Khi mới tiếp cận với Express mình thực sự bị cuốn hút bởi các API của nó, từ cách sử dụng route, template, đều khá dễ tùy chọn và làm việc. Các tính năng của Express framework phải kể đến như:

- Cho phép thiết lập các lớp trung gian để trả về các HTTP request.
- Định nghĩa routing có thể được sử dụng với các hành động khác nhau dựa trên phương thức HTTP và URL.
- Cho phép trả về các trang HTML dựa vào các tham số truyền vào đến template.

1.8.2 Cài đặt Express Framework

Để cài đặt Express framework sử dụng npm như sau:

```
npm install express --save
```

Một số module quan trọng đi cùng với express:

• **body-parser** - Đây là một lớp trung gian node.js để xử lý JSON, dữ liệu thô, text và mã hóa URL.

• **cookie-parser** - Chuyển đổi header của Cookie và phân bổ đến các req.cookies

• **multer** - Đây là một thành phần trung gian trong node.js để xử lý phần multipart/form-data.

```
npm install body-parser --save
```

```
npm install cookie-parser --save
```

```
npm install multer --save
```

1.8.3 Express route

Route là một thành phần cực kỳ quan trọng của một website, nó giúp website biết được người dùng truy cập đến nơi nào của trang web, từ đó phản hồi lại một cách thích hợp. Trong ExpressJs, route được tích hợp sẵn và dễ dàng sử dụng. Bài viết này hãy cùng mình đi tìm hiểu về route trong Express nhé.

a. Route methods

Để sử dụng các phương thức GET, POST, PUT, DELETE trong route:

- GET


```
app.get('/', function (req, res) {
    res.send('Hello World!');
  });
```
- POST


```
app.post('/', function (req, res) {
    res.send('Got a POST request');
  });
```
- PUT


```
app.put('/user', function (req, res) {
    res.send('Got a PUT request at /user');
  });
```
- DELETE


```
app.delete('/user', function (req, res) {
    res.send('Got a DELETE request at /user');
  });
```

Với khai báo như trên, khi truy cập vào địa chỉ localhost:3000/user bằng các phương thức GET, POST, PUT, DELETE server sẽ gửi lại một đoạn text tương ứng.

b. Route parameters

```
app.get('/user/:id', function(req, res) {
    var id = req.params['id'];
    res.send('The id is ' + id);
  });
```

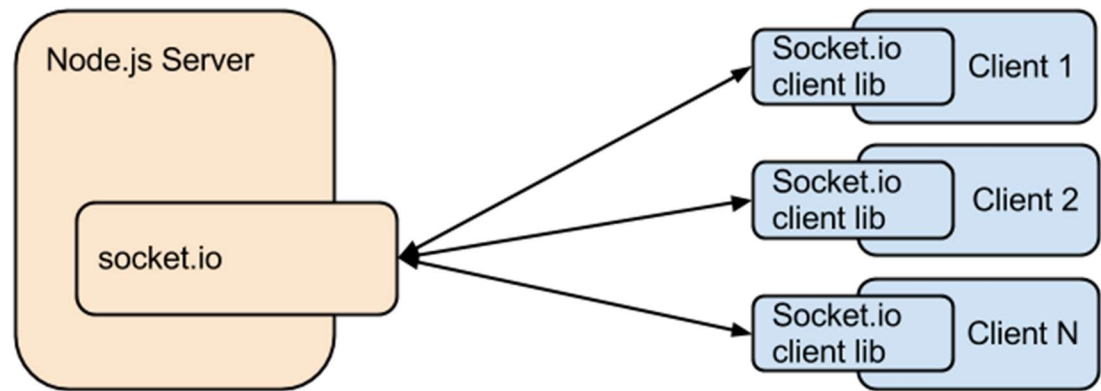
Với khai báo bên trên khi truy cập vào địa chỉ localhost:3000/user/6969 thì server sẽ trả lại đoạn text: The id is 6969.

1.9 Socket.io framework Real-time trong Nodejs

1.9.1 Giới thiệu

- Là một module của NodeJs
- Được xây dựng nhằm mục đích tạo ra real time NodeJS application. Socket.io cung cấp cho lập trình viên các đặc trưng như event, room và tự động phục hồi lại kết nối.
- Khi chúng ta include Socket.io module vào trong ứng dụng của mình nó sẽ cung cấp cho chúng ta hai object đó là: socket server quản lý functionality phía server và socket client điều khiển functionality phía client.
- Khi client muốn kết nối tới Socket.io server, nó sẽ gửi cho server một “handshake HTTP request”. Server sẽ phân tích request đó với những thông tin cần thiết trong

suốt quá trình kết nối. Nó sẽ tìm cấu hình của middleware mà đã được đăng ký với server và thực thi chúng trước khi đưa ra sự kiện kết nối. Khi kết nối thành công thì connection event listener được thực thi, tạo ra một instance mới của socket có thể coi như định danh của client mà mỗi một client kết nối tới sẽ có 1 định danh. Các có thể thấy rõ khi xem hình dưới đây



Hình 1.22: Minh họa về mô hình SocketIO

- Một module khác của Node.js là LightStreamer-adapter cũng có tạo các kết nối từ client tới server nhưng không trực tiếp mà thông qua LightStreamer Server, đó là các máy chủ theo thời gian thực và nằm ngoài tiến trình của Node.js Server

1.9.2 Cài đặt socket.io

Cài đặt trên giao diện dòng lệnh

```
$ npm install --save socket.io
```

Nếu quá trình cài đặt gặp vấn đề gì với quyền truy cập ta sẽ sử dụng thêm với từ khóa sudo

Sau khi cài đặt xong ta có một thư mục awesomechat với cấu trúc như sau

```
.
├── app.js
├── bin
│   └── www
├── package.json
├── public
│   ├── images
│   ├── javascripts
│   └── stylesheets
│       └── style.css
├── routes
└── index.js
```

```

|   └─ users.js
└─ views
    ├── error.pug
    ├── index.pug
    └─ layout.pug
      7 directories, 9 files

```

File package.json của chúng ta lúc này sẽ có dạng như sau:

```

{
  "name": "driper",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "cookie-parser": "~1.4.3",
    "debug": "~2.6.9",
    "ejs": "~2.5.7",
    "express": "~4.16.0",
    "http-errors": "~1.6.2",
    "morgan": "~1.9.0"
  }
}

```

Theo các required dependencies chúng ta sẽ khai báo như trên, ở đây express-generator đã làm việc đó, công việc của mình lúc này chỉ là chạy lệnh `npm install` để được cài đặt các module là được. Khi đó các module sẽ được tìm kiếm như đã được khai báo trong dependencies và được tải về vào thư mục node module

Bắt đầu start server với lệnh sau:

```
$ npm start
```

Có thể thấy trong package.json có đoạn sau:

```
"scripts": { "start": "node ./bin/www" }
```

Khi chạy lệnh `npm start` thì nó sẽ tìm trong file package.json tới câu lệnh đã được gán cho “start” và thực thi nó. Trong file `./bin/www` thì `app.js` đã được required trong đó, trong `app.js` là toàn bộ code phía server hiện tại mà ta có, trong `www` sẽ bổ xung thêm đó là tạo ra một server http và lắng nghe các request tới ứng dụng.

Vào trình duyệt với đường dẫn `localhost:3000` nếu nhận được kết quả như bên dưới là đã cài đặt thành công một ứng dụng nodejs cơ bản với Nodejs sử dụng Express framework.

1.10. Sử dụng MongoDB với NodeJs

1.10.1 Giới thiệu:

MongoDB là một NoSQL, hiệu suất cao rất nổi tiếng, nó được xây dựng với ý tưởng dựa trên cấu trúc document.

Trong MongoDB, dữ liệu được lưu trữ như một document, một tập của các cặp key-value. Có thể định nghĩa nhiều database trong MongoDB và mỗi database có nhiều collections, những collections này đơn giản là tập của các documents được lưu trữ dạng cặp key-value.

Cấu trúc dữ liệu định nghĩa document được gọi là BSON(Binary JSON). BSON là một dạng nhị phân của JSON và cũng hỗ trợ những kiểu dữ liệu như Date, những kiểu mà không được hỗ trợ trong định dạng JSON. MongoDB sẽ chuyển đổi JSON tới BSON và đổi lại lợi ích hiệu suất cao, mặc dù người dùng có thể save, query và nhận dữ liệu như JSON.

1.10.2 Nền tảng và ngôn ngữ hỗ trợ

MongoDB là một cơ sở dữ liệu NoSQL hỗ trợ đa nền tảng, nó có thể chạy trên Windows, Linux và Mac...Nó hỗ trợ hầu hết các ngôn ngữ lập trình phổ biến như C#, Java, PHP, Javascript...và các môi trường phát triển khác nhau.

CHƯƠNG 2: XÂY DỰNG ỨNG DỤNG

2.1 Quy trình nghiệp vụ

Ứng dụng Driper được chia thành hai phiên bản một phiên bản là dành cho khách hàng và hai là dành cho tài xế.

2.1.1 Khách hàng

Khách hàng đăng ký thông tin cần thiết để có thể tạo một cuộc xe theo mong muốn của mình bằng thông tin về địa điểm đón và địa điểm đến.

Khách hàng có thể thấy trên bản đồ những tài xế gần vị trí mình lựa chọn là điểm xuất phát và giá tiền cho cuộc xe do khách hàng yêu cầu.

Sau khi cuộc xe thành công khách hàng có thể cho điểm và đánh giá về chất lượng phục vụ của tài xế.

2.1.2 Tài xế

Tài xế cần đăng ký thông tin cần thiết để trở thành đối tác của Driper và những thông tin có thể cho khách hàng của bạn cần thiết khi thực hiện giao dịch.

Tài xế có thể nắm bắt được mật độ khách hàng đang online và chưa có yêu cầu cuộc xe trên bản đồ để biết được những nơi có nhu cầu đi lại nhiều trong phạm vi trên bản đồ.

Khi nhận được yêu cầu cuộc xe tài xế có quyền nhận hoặc để trôi cuộc xe đã được yêu cầu từ khách hàng. Sau khi cuộc xe bị trôi yêu cầu cuộc xe đó sẽ được gửi tới một tài xế khác.

2.2 Yêu cầu hệ thống

Driper là một ứng dụng trung gian thứ ba nhằm kết nối giữa khách hàng và tài xế bằng cách tìm và chuyển tới tài xế yêu cầu di chuyển của mình với một quãng đường đã chọn.

2.2.1 Khách hàng

Khách hàng cần có một thiết bị di động chạy hệ điều hành Android, kết nối với internet, biết vị trí của mình và cung cấp các thông tin cá nhân cơ bản như tên và số điện thoại liên hệ để có thể tạo và gửi một yêu cầu. Hệ thống sẽ thực hiện các công việc sau:

- Hiện thị các tài xế đang ở gần khách hàng trong đường kính 1km.

- Cho biết khoảng cách, giá tiền giữa hai địa điểm xuất phát và kết thúc của quãng đường yêu cầu của khách hàng.
- Tìm kiếm các tài xế ở gần nhất với khách hàng và có trạng thái sẵn sàng nhận cuộc xe.
- Sau khi đã có tài xế nhận cuộc xe của khách hàng hiển thị các thông tin của tài xế và một số tính năng có thể kết nối với tài xế.
- Trước quá trình chấp nhận giao dịch với tài xế khách hàng có thể hủy cuộc xe đã gửi yêu cầu.
- Sau khi cuộc xe thành công hiển thị các thông tin khách hàng có thể đánh giá được chất lượng phục vụ của tài xế.

2.2.2 Tài xế

Tài xế cần có một thiết bị di động chạy hệ điều hành Android, kết nối với internet, bật vị trí của mình và cung cấp các thông tin cần thiết để đăng ký trở thành đối tác tài xế của Driper. Sau khi trở thành đối tác tài xế của Driper tài xế chỉ cần bật trạng thái sẵn sàng nhận cuộc và bắt đầu nhận các yêu cầu cuộc xe từ phía khách hàng do hệ thống phân phối:

- Hiển thị mật độ khách hàng đang online và chưa yêu cầu cuộc xe trên bản đồ.
- Sau khi nhận được yêu cầu từ phía khách hàng hệ thống sẽ phân phối yêu cầu cho tài xế gần nhất với khách hàng trong phạm vi 1km so với khách hàng.
- Tài xế sẽ nhận được thông tin về cuộc xe và giá tiền từ phía khách hàng. Tài xế có thể chấp nhận hoặc bỏ qua cuộc xe đó.
- Sau khi đã nhận cuộc xe trạng thái của tài xế sẽ thay đổi sang đã nhận cuộc xe mà cuộc xe chưa được thực hiện tài xế có thể hủy cuộc xe đó.
- Sau khi cuộc xe thành công hóa đơn sẽ được gửi tới tài xế và tài xế tiến hành thanh toán với khách hàng của mình.

2.3 Biểu đồ ca sử dụng

2.3.1 Xác định các tác nhân của hệ thống

Dựa vào văn bản mô tả bài toán, ta xác định được các tác nhân của hệ thống :

- Tác nhân Khách hàng
- Tác nhân Tài xế

2.3.2 Xác định các ca sử dụng

Dựa trên văn bản mô tả bài toán và việc phân tích để tìm ra các tác nhân, ta xác định được các ca sử dụng như sau:

a. Khách hàng

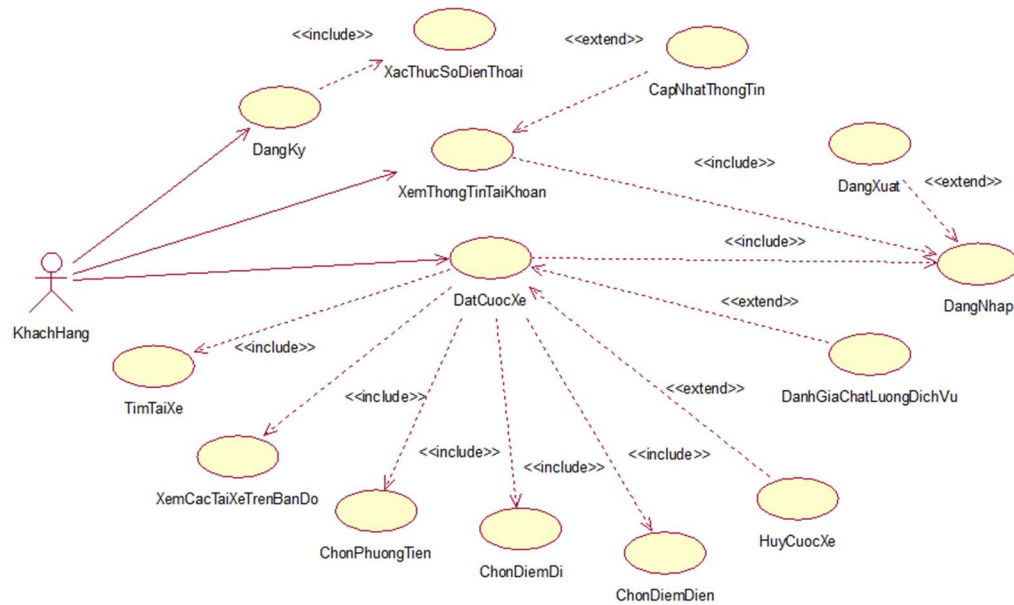
- Đăng ký
- Xác thực số điện thoại
- Đăng nhập
- Đăng xuất
- Xem thông tin tài khoản
- Cập nhật thông tin
- Xem các tài xế trên bản đồ
- Chọn loại hình phương tiện
- Chọn điểm đi
- Chọn điểm đến
- Đặt cước xe
- Hủy cước xe
- Đánh giá chất lượng dịch vụ

b. Tài xế

- Đăng ký
- Xác thực số điện thoại
- Đăng nhập
- Đăng xuất
- Xem thông tin tài khoản
- Xem mật độ khách hàng
- Nhận cước xe
- Hủy cước xe
- Thanh toán

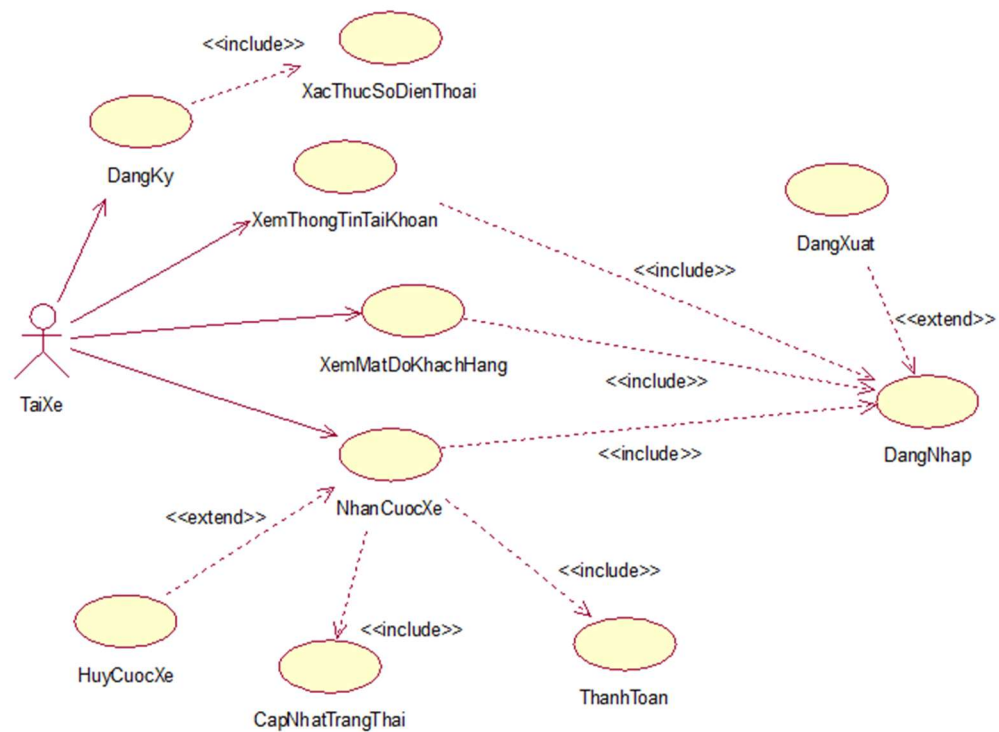
2.3.3 Các biểu đồ ca sử dụng

a. Biểu đồ ca sử dụng của khách hàng



Hình 2.1: Biểu đồ ca sử dụng của Khách hàng

b. Biểu đồ ca sử dụng của Tài xế



Hình 2.1: Biểu đồ usecase Tài xế

2.4 Đặc tả các ca sử dụng

2.4.1 Khách hàng

a. Đặc tả ca sử dụng “Đăng ký”

Tiêu đề	Nội dung
Tên ca sử dụng	Đăng ký
Mô tả	Khi người dùng muốn đặt một cuộc xe người dùng cần phải đăng ký một số thông tin cơ bản để tài xế lái xe có thể liên hệ và giao dịch
Điều kiện kích hoạt	Khi bắt đầu vào chương trình
Tác nhân	Khách hàng
Ca sử dụng liên quan	Xác thực số điện thoại
Tiền điều kiện	Cần phải xác thực số điện thoại của khách hàng là số điện thoại vẫn đang hoạt động và là của khách hàng sử dụng.
Hậu điều kiện	Tự động đăng nhập vào chương trình khi đã đăng ký thành công
Biến thể	Nếu không xác thực đúng số điện thoại khách hàng không thể đăng nhập được vào chương trình.
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng.

b. Đặc tả ca sử dụng “Xác thực số điện thoại”

Tiêu đề	Nội dung
Tên ca sử dụng	Xác thực số điện thoại
Mô tả	Xác thực số điện thoại của khách hàng là số điện thoại vẫn đang hoạt động và là của khách hàng sử dụng.
Điều kiện kích hoạt	Sau khi đã nhập các thông tin đăng ký.
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đăng ký
Tiền điều kiện	Đã nhập đầy đủ các thông tin đăng ký là tên và số điện thoại liên hệ.
Hậu điều kiện	Tự động đăng nhập vào chương trình khi đã đăng ký thành công
Biến thể	Nếu không xác thực đúng số điện thoại khách hàng không thể đăng nhập được vào chương trình.

Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng
----------	--

c. Đặc tả ca sử dụng “Đăng nhập”

Tiêu đề	Nội dung
Tên ca sử dụng	Đăng nhập
Mô tả	Tự động đăng nhập vào chương trình khi đã đăng ký thành công
Điều kiện kích hoạt	Đăng ký và xác thực thông tin của khách hàng.
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đăng xuất, Xem thông tin tài khoản, Xem các tài xế trên bản đồ, Đặt cuộc xe
Tiền điều kiện	Đăng ký và xác thực thông tin của khách hàng.
Hậu điều kiện	Vào chương trình chính.
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

d. Đặc tả ca sử dụng “Đăng xuất”

Tiêu đề	Nội dung
Tên ca sử dụng	Đăng xuất
Mô tả	Đăng xuất ra khỏi chương trình
Điều kiện kích hoạt	Khi người dùng chọn chức năng đăng xuất từ chương trình
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đăng nhập
Tiền điều kiện	Đã đăng nhập vào chương trình
Hậu điều kiện	Người dùng đã đăng xuất thành công
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

e. Đặc tả ca sử dụng “Xem thông tin tài khoản”

Tiêu đề	Nội dung
Tên ca sử dụng	Xem thông tin tài khoản
Mô tả	Xem thông tin đã đăng ký với chương trình
Điều kiện kích hoạt	Khi người dùng chọn chức năng xem thông tin tài khoản từ chương trình
Tác nhân	Khách hàng

Ca sử dụng liên quan	Cập nhật thông tin tài khoản
Tiền điều kiện	Đã đăng nhập vào chương trình
Hậu điều kiện	Xem thông tin đã đăng ký với chương trình
Ngoại lệ	Khi tải dữ liệu bị lỗi thông báo lỗi lên màn hình Người dùng không bật thông tin GPS và kết nối mạng

f. Đặc tả ca sử dụng “Cập nhật thông tin tài khoản”

Tiêu đề	Nội dung
Tên ca sử dụng	Cập nhật thông tin tài khoản
Mô tả	Cập nhật thông tin tài khoản của khách hàng đã đăng ký
Điều kiện kích hoạt	Sau khi khách hàng chọn chức năng cập nhật thông tin tài khoản
Tác nhân	Khách hàng
Ca sử dụng liên quan	Xem thông tin tài khoản
Tiền điều kiện	Khi đã đăng ký và đã đăng nhập vào chương trình
Hậu điều kiện	Hiện thị trạng thái đã cập nhật thành công thông tin tài khoản

g. Đặc tả ca sử dụng “Xem các tài xế trên bản đồ”

Tiêu đề	Nội dung
Tên ca sử dụng	Xem các tài xế trên bản đồ
Mô tả	Hiện thị các tài xế lái xe theo phương tiện đã chọn xung quang vị trí đón của khách hàng với bán kính là 1km
Điều kiện kích hoạt	Đã chọn loại hình phương tiện.
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đặt cước xe
Tiền điều kiện	Đã chọn loại hình phương tiện.
Hậu điều kiện	Hiện thị các tài xế lái xe theo phương tiện đã chọn xung quang vị trí đón của khách hàng với bán kính là 1km.

h. Đặc tả ca sử dụng “Đặt cước xe”

Tiêu đề	Nội dung
---------	----------

Tên ca sử dụng	Đặt cước xe
Mô tả	Khách hàng lựa chọn phương tiện di chuyển , điểm đi và điểm đến và tiến hành đặt cước xe theo yêu cầu của mình
Điều kiện kích hoạt	Đã đăng nhập vào chương trình.
Tác nhân	Khách hàng
Ca sử dụng liên quan	Tìm tài xế, Xem các tài xế trên bản đồ, Chọn phương tiện, chọn điểm đi, Chọn điểm đến, Hủy cước xe.
Tiền điều kiện	Sau khi đã chọn điểm đi, điểm đến và chọn phương tiện vận chuyển và tiến hành chọn đặt cước xe.
Hậu điều kiện	Hiện thị đang tìm tài xế cho cước xe đã đặt
Biến thể	Không tìm thấy tài xế nào thì thông báo lên màn hình là không tìm thấy tài xế
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

i. Đặc tả ca sử dụng “Chọn điểm đi”

Tiêu đề	Nội dung
Tên ca sử dụng	Chọn điểm đi
Mô tả	Hành khách lựa chọn điểm xuất phát của mình trên bản đồ
Điều kiện kích hoạt	Khách hàng lựa chọn chức năng tìm điểm đi của chương trình
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đặt cước xe
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Chọn điểm xuất phát của khách hàng
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

j. Đặc tả ca sử dụng “Chọn điểm đến”

Tiêu đề	Nội dung
Tên ca sử dụng	Chọn điểm đến
Mô tả	Hành khách lựa chọn điểm đến của mình trên bản đồ
Điều kiện kích hoạt	Khách hàng lựa chọn chức năng tìm điểm đến của chương trình
Tác nhân	Khách hàng

Ca sử dụng liên quan	Đặt cước xe
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Chọn điểm đến của khách hàng
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

k. Đặc tả ca sử dụng “Chọn phương tiện”

Tiêu đề	Nội dung
Tên ca sử dụng	Chọn phương tiện
Mô tả	Giúp hành khách lựa chọn phương tiện di chuyển của mình trên chương trình
Điều kiện kích hoạt	Khách hàng lựa chọn chức năng tìm điểm đến của chương trình
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đặt cước xe
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống
Hậu điều kiện	Hiển thị thông tin giá tiền về phương tiện di chuyển và hiển thị các tài xế xung quang vị trí xuất phát của khách hàng với phương tiện đã chọn
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

l. Đặc tả ca sử dụng “Hủy cước xe”

Tiêu đề	Nội dung
Tên ca sử dụng	Hủy cước xe
Mô tả	Sau khi đã đặt cước xe hành khách có thể hủy cước xe của mình
Điều kiện kích hoạt	Đã đặt cước xe
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đặt cước xe
Tiền điều kiện	Khách hàng lựa chọn chức năng hủy cước xe trên hệ thống
Hậu điều kiện	Thông báo hủy cước xe thành công

m. Đặc tả ca sử dụng “Đánh giá chất lượng dịch vụ”

Tiêu đề	Nội dung
---------	----------

Tên ca sử dụng	Đánh giá chất lượng dịch vụ
Mô tả	Sau khi cuộc xe của khách hàng thành công, khách hàng có thể đánh giá, phản hồi lại chất lượng dịch vụ của tài xế lái xe.
Điều kiện kích hoạt	Sau khi cuộc xe đã hoàn thành
Tác nhân	Khách hàng
Ca sử dụng liên quan	Đặt cuộc xe
Tiền điều kiện	Sau khi cuộc xe đã hoàn thành và hành khách xuống xe
Hậu điều kiện	Hiển thị lời cảm ơn hành khách đã sử dụng dịch vụ

2.4.2 Tài xế

a. Đặc tả ca sử dụng “Đăng ký”

Tiêu đề	Nội dung
Tên ca sử dụng	Đăng ký
Mô tả	Khi người dùng muốn trở thành đối tác tài xế của Driper
Điều kiện kích hoạt	Khi bắt đầu vào chương trình
Tác nhân	Tài xế
Ca sử dụng liên quan	Xác thực số điện thoại
Tiền điều kiện	Cần phải xác thực số điện thoại của tài xế là số điện thoại vẫn đang hoạt động và là của tài xế sử dụng.
Hậu điều kiện	Tự động đăng nhập vào chương trình khi đã đăng ký thành công
Biến thể	Nếu không xác thực đúng số điện thoại tài xế không thể đăng nhập được vào chương trình.
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng.

b. Đặc tả ca sử dụng “Xác thực số điện thoại”

Tiêu đề	Nội dung
Tên ca sử dụng	Xác thực số điện thoại
Mô tả	Xác thực số điện thoại của tài xế là số điện thoại vẫn đang hoạt động và là của tài xế sử dụng.
Điều kiện kích hoạt	Sau khi tài xế đã nhập các thông tin đăng ký.
Tác nhân	Tài xế

Ca sử dụng liên quan	Đăng ký
Tiền điều kiện	Đã nhập đầy đủ các thông tin đăng ký là tên và số điện thoại liên hệ.
Hậu điều kiện	Tự động đăng nhập vào chương trình khi đã đăng ký thành công
Biến thể	Nếu không xác thực đúng số điện thoại tài xế không thể đăng nhập được vào chương trình.
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

c. Đặc tả ca sử dụng “Đăng nhập”

Tiêu đề	Nội dung
Tên ca sử dụng	Đăng nhập
Mô tả	Tự động đăng nhập vào chương trình khi đã đăng ký thành công
Điều kiện kích hoạt	Đăng ký và xác thực thông tin của tài xế
Tác nhân	Tài xế
Ca sử dụng liên quan	Đăng xuất, Xem thông tin tài khoản, Xem các tài xế trên bản đồ, Đặt cuộc xe
Tiền điều kiện	Đăng ký và xác thực thông tin của tài xế
Hậu điều kiện	Vào chương trình chính.
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

d. Đặc tả ca sử dụng “Đăng xuất”

Tiêu đề	Nội dung
Tên ca sử dụng	Đăng xuất
Mô tả	Đăng xuất ra khỏi chương trình
Điều kiện kích hoạt	Khi tài xế chọn chức năng đăng xuất từ chương trình
Tác nhân	Tài xế
Ca sử dụng liên quan	Đăng nhập
Tiền điều kiện	Đã đăng nhập vào chương trình
Hậu điều kiện	Tài xế đã đăng xuất thành công
Ngoại lệ	Người dùng không bật thông tin GPS và kết nối mạng

e. Đặc tả ca sử dụng “Xem thông tin tài khoản”

Tiêu đề	Nội dung
Tên ca sử dụng	Xem thông tin tài khoản
Mô tả	Xem thông tin đã đăng ký với chương trình
Điều kiện kích hoạt	Khi người dùng chọn chức năng xem thông tin tài khoản từ chương trình
Tác nhân	Tài xế
Ca sử dụng liên quan	Cập nhật thông tin tài khoản
Tiền điều kiện	Đã đăng nhập vào chương trình
Hậu điều kiện	Xem thông tin đã đăng ký với chương trình
Ngoại lệ	Khi tài dữ liệu bị lỗi thông báo lỗi lên màn hình Người dùng không bật thông tin GPS và kết nối mạng

f. Đặc tả ca sử dụng “Xem mật độ khách hàng”

Tiêu đề	Nội dung
Tên ca sử dụng	Xem mật độ khách hàng
Mô tả	Hiện bị biểu đồ Heatmaps trên bản đồ của tài xế lái xe về khách hàng đang trực tuyến trên hệ thống
Điều kiện kích hoạt	Sau khi tài xế đã đăng nhập thành công
Tác nhân	Tài xế
Ca sử dụng liên quan	Đăng nhập
Tiền điều kiện	Đã đăng nhập vào chương trình
Hậu điều kiện	Hiện bị biểu đồ Heatmaps trên bản đồ của tài xế lái xe về khách hàng đang trực tuyến trên hệ thống

g. Đặc tả ca sử dụng “Nhận cuộc xe”

Tiêu đề	Nội dung
Tên ca sử dụng	Nhận cuộc xe
Mô tả	Sau khi tài xế đã thay đổi trạng thái sẵn sàng nhận cuộc xe, khi có yêu cầu cuộc xe của khách hàng được phân phối tới tài xế, tài xế có thể nhận hoặc bỏ qua cuộc xe đó
Điều kiện kích hoạt	Khi tài xế đã bật trạng thái sẵn sàng nhận cuộc xe
Tác nhân	Tài xế

Ca sử dụng liên quan	Đăng nhập, Hủy cuộc xe, Cập nhật trạng thái, Thanh toán
Tiền điều kiện	Khi tài xế đã bật trạng thái sẵn sàng nhận cuộc xe, tài xế gần nhất trong bán kính 1km so với khách hàng
Hậu điều kiện	Hiện thị thông tin cuộc xe và giá tiền cuộc xe yêu cầu từ khách hàng

h. Đặc tả ca sử dụng “Hủy cuộc xe”

Tiêu đề	Nội dung
Tên ca sử dụng	Hủy cuộc xe
Mô tả	Vì một lý do nào đó tài xế có thể hủy cuộc xe do khách hàng yêu cầu.
Điều kiện kích hoạt	Tài xế đã nhận cuộc xe
Tác nhân	Tài xế
Ca sử dụng liên quan	Nhận cuộc xe, Đăng xuất
Tiền điều kiện	Tài xế chọn chức năng hủy cuộc xe sau khi đã nhận cuộc xe
Hậu điều kiện	Thông báo hủy thành công và gửi thông báo đến khách hàng
Biến thể	
Ngoại lệ	

i. Đặc tả ca sử dụng “Cập nhật trạng thái”

Tiêu đề	Nội dung
Tên ca sử dụng	Cập nhật trạng thái
Mô tả	Tài xế có thể thay đổi trạng thái sẵn sàng nhận cuộc xe hoặc không sẵn sàng nhận cuộc xe của mình trên hệ thống.
Điều kiện kích hoạt	Tài xế đã đăng nhập vào hệ thống
Tác nhân	Tài xế
Ca sử dụng liên quan	Đăng nhập
Tiền điều kiện	Tài xế bật tắt trạng thái sẵn sàng nhận cuộc xe
Hậu điều kiện	Thông báo lên màn hình trạng thái của tài xế

j. Đặc tả ca sử dụng “Thanh toán”

Tiêu đề	Nội dung
Tên ca sử dụng	
Mô tả	Sau khi đã tới địa điểm đến theo yêu cầu của hành khách. Tài xế sẽ nhận được thông thanh toán từ hệ thống
Điều kiện kích hoạt	Sau khi cuộc xe đã hoàn thành
Tác nhân	Tài xế
Ca sử dụng liên quan	Nhận cuộc xe
Tiền điều kiện	Chọn chức năng khách xuống xe trên chương trình
Hậu điều kiện	Hiện thị thông tin thanh toán
Biến thể	
Ngoại lệ	

2.5 Giao diện chương trình

2.5.1 Giao diện của phần mềm dành cho khách hàng



Giao diện đầu tiên khi khách hàng chạy ứng dụng yêu cầu đăng nhập sử dụng một số thông tin của người dùng:

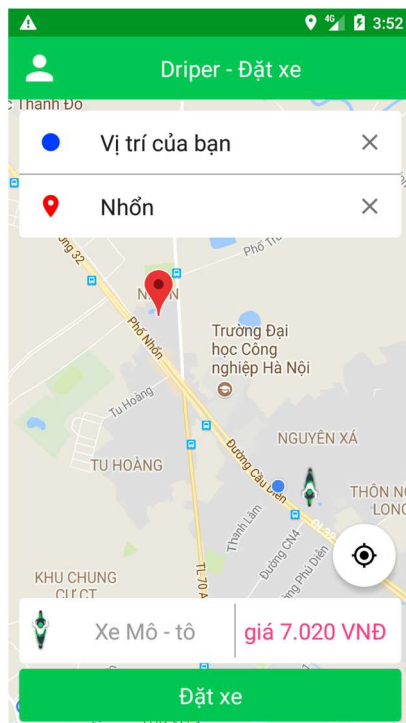
- Số điện thoại của khách hàng khi đặt cuộc xe để tài xế có thể liên hệ và trao đổi
- Tên khách hàng sử dụng khi giao dịch với tài xế

Hình 2.1 Giao diện đăng nhập ứng dụng dành cho khách hàng








Hình 2.2 Giao diện xác thực số điện thoại

Sau khi khách hàng nhập xong các thông tin cần thiết, khách hàng cần xác thực lại số điện thoại của mình là số điện thoại đang hoạt động và tài xế có thể liên hệ với khách hàng bằng mã xác nhận được gửi tới sau khi đã được kiểm tra trên hệ thống.

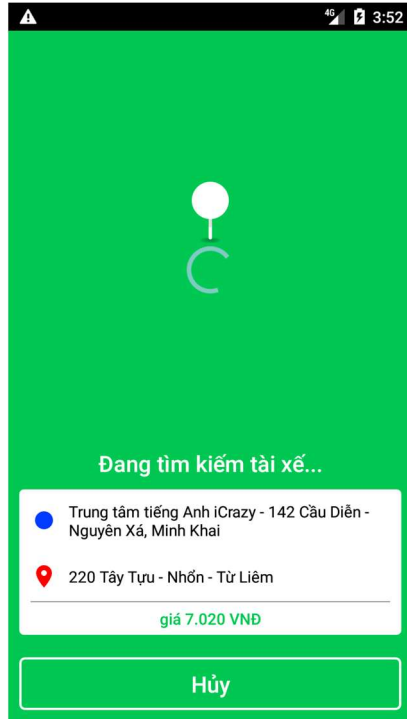


Hình 2.3 Giao diện chính của ứng dụng

Sau khi đăng nhập thành công khách hàng sẽ chuyển sang màn hình chính sửa dụng của ứng dụng:

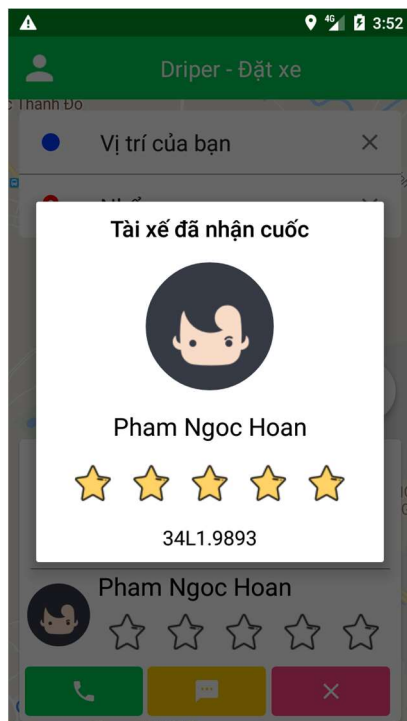
- Khách hàng chọn  để xem và cập nhật lại thông tin đã đăng kí với hệ thống.
- Khách hàng chọn  để cập nhật và di chuyển tới vị trí của mình được hiển thị  trên bản đồ Google Maps.
- Sau khi khách hàng lựa chọn được địa điểm đi  và địa điểm đến  hệ thống tự hiển thị giá tiền tương ứng với quãng đường đi của khách hàng.

Khách hàng chọn nút đặt xe để đặt cuộc xe của mình và bắt đầu tìm tài xế.



- Sau khi đặt cuộc xe và chuyển sang màn hình tìm tài xế hiển thị thông tin của điểm đi, điểm đến và giá tiền cho cuộc xe.
- Hệ thống sẽ tìm tài xế gần nhất ở lân cận vị trí điểm đi của khách hàng trong bán kính 2km.
- Khách hàng có thể hủy cuộc xe trong lúc đang tìm tài xế

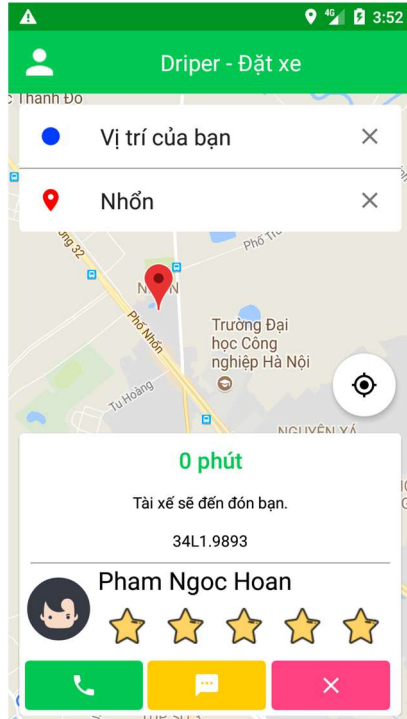
Hình 2.4 Giao diện tìm tài xế



Sau khi tìm thấy được tài xế hệ thống sẽ thông báo thông tin tài xế nhận cuộc xe:

- Hình chân dung của tài xế
- Tên tài xế
- Số điểm sao đánh giá tài xế
- Biển số xe

Hình 2.5 Thông báo tài xế nhận cuộc



Hình 2.6 Thông tin cuộc xe đã đặt

Hiện thị thông tin của tài xế và khách hàng có thể:

- Gọi điện cho tài xế
- Nhắn tin cho tài xế
- Hủy cuộc xe



Hình 2.7 Màn hình phản hồi về cuộc xe sau khi cuộc xe

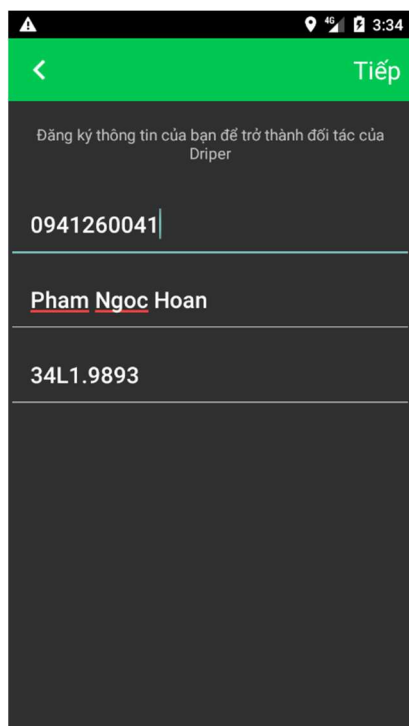
Sau khi cuộc xe hoàn thành khách hàng phải hỏi về cuộc xe đã hoàn thành.

2.5.2 Giao diện của phần mềm dành cho tài xế



Giao diện sẽ xuất hiện khi lần đầu tiên tài xế mở ứng dụng và lựa chọn đăng nhập hay đăng ký để sử dụng.

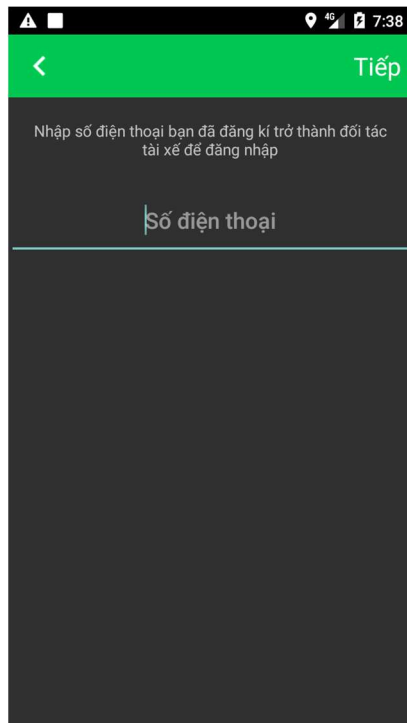
Hình 2.8 Màn hình chào mừng



Sau khi lựa chọn đăng kí từ Màn hình chào mừng tài xế, ở màn hình này tài xế cần đăng kí các thông tin sau và gửi đăng ký lên hệ thống:

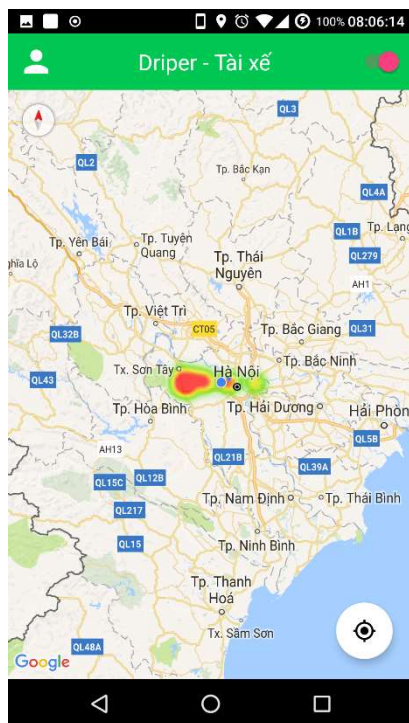
- Số điện thoại.
- Tên tài xế.
- Thông tin biển số xe.
- Ảnh trước và sau chứng minh thư.
- Ảnh trước và sau bằng lái xe.
- Ảnh trước và sau giấy tờ xe.
- Ảnh trước và sau bảo hiểm xe.

Hình 2.9 Màn hình đăng ký







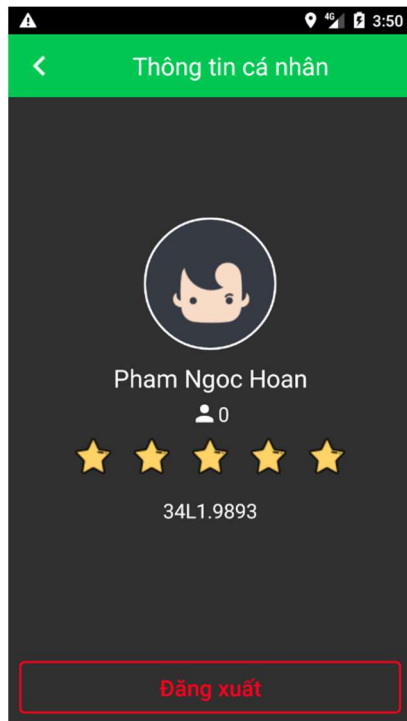
Hình 2.10 Màn hình đăng nhập

- Sau khi lựa chọn đăng nhập từ Màn hình chào mừng tài xế, ở màn hình này tài xế cần nhập số điện thoại để đăng nhập.
- Sau khi khách hàng nhập xong các thông tin cần thiết, khách hàng cần xác thực lại số điện thoại của mình là số điện thoại đang hoạt động và tài xế có thể liên hệ với khách hàng bằng mã xác nhận được gửi tới sau khi đã được kiểm tra trên hệ thống.



Hình 2.11 Màn hình chính của tài xế

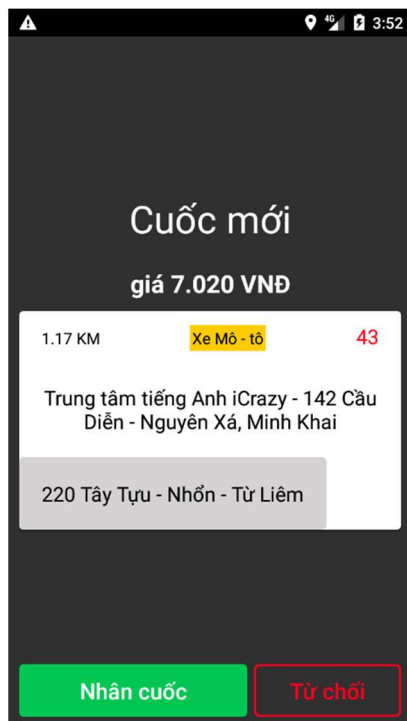
- Hiện thị mật độ khách hàng đang sử dụng ứng dụng và có nhu cầu đặt xe.
- Tài xế chọn  để xem thông tin đã đăng kí với hệ thống.
- Tài xế chọn  để sẵn sàng/ không sẵn sàng nhận cuộc xe.
- Tài xế chọn  để cập nhật và di chuyển tới vị trí của mình được hiển thị  trên bản đồ Google Maps.



Hình 2.12 Giao diện thông tin tài xế

Hiện thị thông tin tài xế:

- Hình ảnh chân dung
- Tên tài xế
- Số người đánh giá
- Số điểm sao
- Thông tin biển số xe



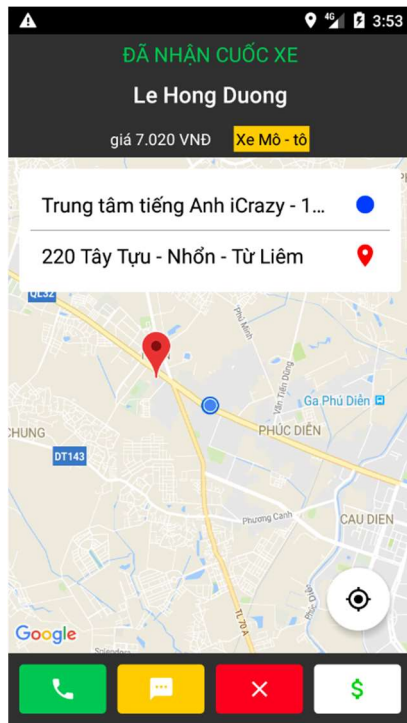
Hình 2.13 Màn hình thông báo cuộc mới

Màn hình thông báo có cuộc xe mới từ khách hàng đặt cuộc xe hiển thị các thông tin:

- Giá tiền của cuộc xe
- Khoảng cách giữa tài xế và khách hàng.
- Điểm đón
- Điểm dừng

Tài xế có 45s để lựa chọn quá 45s cuộc xe sẽ được chuyển đến cho tài xế khác.

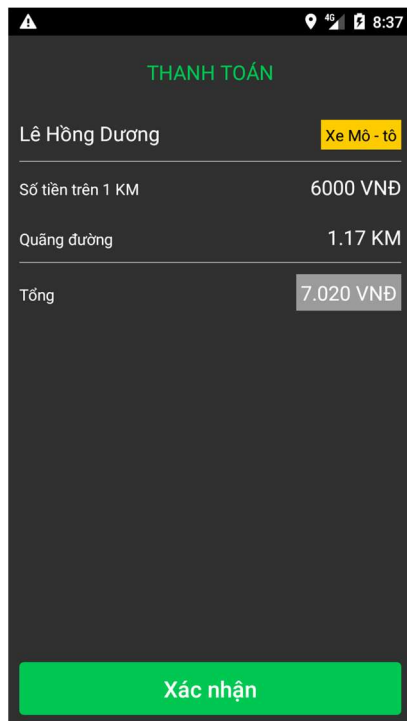
Tài xế có thể chọn “Nhận cuộc” hoặc “Từ chối” cuộc xe của khách hàng.



Hình 2.14 Màn hình sau khi nhận cuộc

Sau khi nhận cuộc xe tài xế sẽ có màn hình thông tin của cuộc xe và một số tùy chọn để liên hệ với khách hàng:

- Chọn  để bật chỉ đường tới điểm đón khách hàng.
- Chọn  để bật chỉ đường tới điểm trả khách hàng.
- Chọn  để gọi cho khách hàng.
- Chọn  để nhắn tin cho khách hàng.
- Chọn  để hủy cuộc xe.
- Chọn  để tiến hành thanh toán cuộc xe.



Hình 2.15 Màn hình thanh toán

Sau khi tài xế chọn thanh toán màn hình thanh toán sẽ xuất hiện với các thông tin

- Giá tiền trên 1km đi xe
- Quãng đường của cuộc xe
- Tổng giá tiền

KẾT LUẬN CHUNG

Sau một thời gian tìm hiểu và nghiên cứu đề tài “Giải pháp vận tải Driper” em đã hoàn thành và phát triển về cơ bản theo đúng những yêu cầu về nội dung và thời gian đã định. Trong quá trình nghiên cứu và thực hiện đề tài với quyết tâm cao nhưng do hạn chế về kinh nghiệm và kiến thức nên đề tài của em chắc chắn sẽ không thể tránh khỏi những thiếu sót. Em rất mong nhận được ý kiến đóng góp từ quý thầy cô và các bạn để đề tài được hoàn thiện hơn.

Kết quả đạt được

Về mặt công nghệ, em đã tìm hiểu và nắm bắt quy trình để xây dựng một chương trình phần mềm chạy trên thiết bị di động chạy hệ điều hành Android bằng ngôn ngữ lập trình Java, áp dụng được công nghệ Nodejs, Google Maps API và MongoDB vào chương trình của mình. Sau đây là các công việc đã đạt được:

- Sử dụng các công cụ lập trình và ngôn ngữ Java để xây dựng phần mềm ứng dụng Driper chạy trên các thiết bị thông minh chạy hệ điều hành Android.
- Sử dụng thư viện Socket.IO chạy trên công nghệ NodeJS tạo nên hiệu quả phần mềm sử dụng thời gian thực giao tiếp giữa client – server.
- Sử dụng MongoDB lưu trữ cơ sở dữ liệu dưới kiểu dữ liệu dạng JSON về dữ liệu của khách hàng, tài xế và chuyến đi.
- Sử dụng Google Maps API để hiển thị những điểm đánh dấu các địa điểm đón trả khách, hiển thị bản đồ mật độ các khách hàng có nhu cầu đi lại, công cụ tìm kiếm bản đồ giúp khách hàng lựa chọn được địa điểm đến và đi.

Hướng phát triển

Trong tương lai, với nhu cầu đi lại, mật độ dân cư tập trung ngày càng tăng đồng nghĩa với lượng phương tiện di chuyển ngày càng nhiều và đa dạng hóa một số ý tưởng có thể phát triển tiếp với “Phần mềm giải pháp vận tải Driper”:

- Thêm tính năng cho những tài xế có xe ô-tô.
- Thêm tính năng vận chuyển hàng hóa.
- Thêm tính năng chia sẻ xe.
- Thêm tính năng sử dụng cho những chiếc xe taxi.

TÀI LIỆU THAM KHẢO

- [1]. <https://nodejs.org/en/docs/>, truy cập cuối cùng 05/04/2018.
- [2]. <https://www.w3schools.com/nodejs/>, truy cập cuối cùng 05/04/2018.
- [3]. <https://viblo.asia/p/nodejs-tutorial-phan-1-gioi-thieu-va-cai-dat-ung-dung-dau-tien-gVQvlwdykZJ>, truy cập cuối cùng 05/04/2018.
- [4]. https://www.w3schools.com/nodejs/nodejs_mongodb.asp, truy cập cuối cùng 05/04/2018.
- [5]. <https://docs.mongodb.com/>, truy cập cuối cùng 05/04/2018
- [6]. <https://kipalog.com/posts/Hoc-MongoDB--Bai-1--MongoDB-la-gi>, truy cập cuối cùng 05/04/2018.
- [7]. <https://socket.io/docs/>, truy cập cuối cùng 05/04/2018.
- [8]. <https://github.com/nkzawa/socket.io-android-chat>, truy cập cuối cùng 05/04/2018
- [9]. <https://viblo.asia/p/buoc-dau-lam-quen-voi-nodejs-va-socketio-MJyGjQrWvPB>, truy cập cuối cùng 05/04/2018.
- [10]. <https://viblo.asia/p/nodejs-va-socketio-can-ban-jlA7GKxdvKZQ>, truy cập cuối cùng 05/04/2018- EpressJS:
- [11]. <https://expressjs.com/en/4x/api.html>, truy cập cuối cùng 05/04/2018.
- [12]. <https://viblo.asia/p/nodejs-voi-express-framework-rQOvPKVgkYj>, truy cập cuối cùng 05/04/2018.
- [13]. <https://developers.google.com/maps/documentation/android-api/>, truy cập cuối cùng 05/04/2018.
- [14]. <https://developers.google.com/maps/documentation/android-sdk/reference>, truy cập cuối cùng 05/04/2018.