**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHÊ THÔNG TIN VÀ TRUYỀN THÔNG**

**ĐỒ ÁN TỐT NGHIỆP**

**Triển khai hệ thống server và ứng dụng Android chặn các cuộc gọi làm phiền giựa trên phản hồi của người dùng**

**Nguyễn Văn Nam**

[**nam.nv162840@sis.hust.edu.vn**](mailto:nam.nv162840@sis.hust.edu.vn)

**Ngành Công nghệ thông tin**

**Chuyên nghành Kỹ thuật máy tính**

**Giảng viên hướng dẫn: PSG. TS. Lã Thế Vinh \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**(Chữ ký)**

**Bộ Môn: Kỹ thuật máy tính**

**Viện: Công nghệ thông tin và truyền thông**

**HÀ NỘI 06/2021**

**ĐỀ TÀI TỐT NGHIỆP**

1. **Thông tin sinh viên**

* Họ và tên: **Nguyễn Văn Nam**
* Lớp: **CNTT1.01-K61**
* Email: **nam.nv162840@sis.hust.edu.vn**
* Điện thoại: **0356955552**
* Hệ đào tạo: **Kỹ sư chính quy**
* Đồ án tốt nghiệp được thực hiện tại: **Bộ môn Kỹ thuật máy tính – Viện Công nghệ thông tin và truyền thông – Đại học Bách khoa Hà Nội**
* Thời gian thực hiện:

1. **Mục đính nội dung của ĐATN**

Triển khai hệ thông ứng dụng trên điện thoại Android là một app Điện thoại(Calling App) với chức năng chính là phát hiện và chặn các cuộc gọi làm phiền. Kết hợp hệ thống Backend server lưu trữ phàn hồi từ người dùng, đánh giá và gợi ý cảnh báo tới tất cả người dùng khác.

1. **Các nhiệm vụ cụ thể**

* Tìn hiểu về Implementing Block Phone Numbers trên hệ điều hành Android
* Tìm hiểu về ứng dụng điện thoại (Calling App)
* Tìn hiểu về hệ thống Backend Nodejs kết hợp MYSQL

1. **Lời cam đoan của sinh viên**

Tôi - Nguyễn Văn Nam – cam kết ĐATN này là sản phẩm được thực hiện bởi tôi dưới sự hướng dẫn của PSG. TS. Lã Thế Vinh.

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

Hà Nội ngày tháng năm 2021

Sinh viên

Nguyễn Văn Nam

1. Xác nhận của giảng viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ.

Hà Nội, ngày tháng năm 2021

Giảng viên hướng dẫn

PGS. TS. Lã Thế Vinh

**Lời cảm ơn**

Em xin gửi lời cảm ơn tới tất cả các thầy cô và cán bộ tại trường Đại học Bách Khoa Hà Nội và đặc biệt là các thầy công trong Viện Công nghệ thông tin và truyền thông. Thầy cô đã giảng dạy và cung cấp kiến thức, kĩ năng quan trọng, cần thiết trên con đường học tập, nghiên cứu và làm việc của em sau này. Mang tới khát vọng, sự tự tin và khả năng xây dựng công hiến cho đất nước trong qua trình cách mạng hiện đại hóa công nghiệp 4.0. Em rất tự hào khi được học tập dưới mái trường Bách Khoa thân yêu đó là điểm tự, động lực thúc đẩy em luôn luôn phải cố găng phát triển trên con đường nghiên cứu khoa học.

**Tóm tắt nội dung đồ án**

Với sự phát triển bủng nổ phổ biến của smartphone thì vấn đề giải quyết các cuộc gọi rác, tin nhắn rác làm phiền tới trải nghiệm người dùng cũng được đặc biệt quan tâm. Đã có nhiều giải pháp được các bên đưa ra và áp dụng nhưng vì những khó khăn gặp phải mà vẫn nạn trên vẫn chưa được giiar quyết hợp lý.

Trong đồ án này em đề xuất và triển khai thêm một hế thống để giải quyết bài toán cuộc gọi, tin nhắn rác. Hệ thống là chốt chặn cuối cùng trên chính thiết bị smartphone, tập trung vào sự phản hồi của người dùng nhằm phát hiện vầ cảnh báo sớm cuộc gọi, tin nhắn rác.

Nội dung đồ án gồm những phần:

**Chương 1: Tổng quan đề tài**

Giới thiệu bài toán, tầm quan trọng và tính thực tiễn của bài toán. Định hướng tiếp cận và triển khai của đồ án.

**Chương 2: Cơ sở lý thuyết**

Các lý thuyết cơ bản về công nghệ sẽ triển khai.

**Chương 3: Cách tiếp cận triển khai và công nghệ sử dụng**

Trình bày chi tiết cách thức triển khai hệ thống.

**Chương 4: Thực nghiệm và kết quả**

Đánh giá quá trình và nhận xét kết quả thu được.

**Chương 5: Kết luận và hướng phát triển**

Tổng kết các vấn đề gặp phải trong quá trình triển khai và hướng phát triển trong tương lai.

**MỤC LỤC**

**CHƯƠNG 1. Giới thiệu đề tài**

**CHƯƠNG 2. Cơ sở lý thuyết**

**CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI**

* 1. **Giới thiệu chung**

Ngày nay với sự phát triển công nghệ và kinh tế smartphone đã ngày càng trở nên phổ biến và là thiết bị cần thiết đối với con người trong quá trình làm việc, học tập và giải trí. Một trong nhưng ứng dụng cở bản của điện thoại là cuộc gọi và tin nhắn để giao tiếp một cách nhanh trong. Vì vậy nó bị lợi dụng làm một kênh quảng cáo hoặc lừa đảo tới người dùng, gây ra rất nhiều phiền phức ảnh hửng đến công việc và trải nghiệm của người dùng.

Vấn nạn này đã được quan tâm nhiều hơn trong thời gian gần đây từ nhà nước đến các đơn vị cung cấp dịnh vụ viễn thông. Với nhiều giải pháp được đưa ra như định danh số thuê bao, ra soát quản lý phát hiện từ nguồn của các đơn vị. Nhưng vì nhưng khó khăn gặp phải trong qua trình cung cấp dịch vụ và sự khôn khéo của các đối tượng thì vấn nạn trên vẫn chưa được giải quyết một cách hợp lý.

* 1. **Giới thiệu giải pháp**

Để giải quyết các cuộc gọi, tin nhắn rác ngoài các biện pháp chốt chặn tại nguồn thì tại đồ án này em đề xuất thêm một giải pháp làm chốt chặn ngay tại thiết bị của người dùng đây sẽ là chốt chặn cuối cùng và người dùng có thể can thiệp và quyết định loại bỏ nó. Hệ thống sẽ cung cấp các công cụ để chặn các số thuê bao gây phiền phức và tiếp nhận các thông tin phản hồi từ người dùng. Hệ thống sễ lưu trữ, đánh giá và gửi cánh báo lại cho tất cả các người dùng khác. Giải pháp này sẽ có hiệu quả ngay lập tức cho mội người dùng nhất định và sẽ rất hiệu quả khi có lượng người dùng đủ lớn để phân tính đánh giá.

* 1. **Các bước triển khai giải pháp**

**1.3.1. Xây dựng ứng dụng cuộc gọi cơ bản**

Để đáp ứng được nhu cầu cơ bản của một chiếc điện thoại là thực hiện cuộc gọi, nhận cuộc gọi tin nhắn thì cần xây dưng một ứng dụng cuộc gọi đáp ứng đước các tác vụ trên. Ứng dụng có thể nhận và thực hiện cuộc gọi một cách bình thường ngoài ra cần có một giao diện đẹp và tối ưu cho các thao tác từ người dùng.

**1.3.2. Triển khai tính năng chặn cuộc gọi**

Ngoài các tính năng cơ bản trên, ứng dụng có khả năng chặn cuộc các cuộc gọi được người dùng vào. Các cuộc gọi này sẽ không sẽ được tự động hủy đi và sẽ chỉ hiện trong lịch sử của danh sách cuộc gọi.

**1.3.3. Triển khai hệ thống backend server**

Xây dựng hệ thống server để lưu trữ lại các báo cáo phản hồi từ người dùng gửi từ ứng dụng gửi lên. Hệ thống sẽ phân loại đánh giá và quyết định cho vào danh sách hạn chế. Sau đó danh sách hạn chế sẽ được gửi lại cho tất cả người dùng để cảnh báo trước.

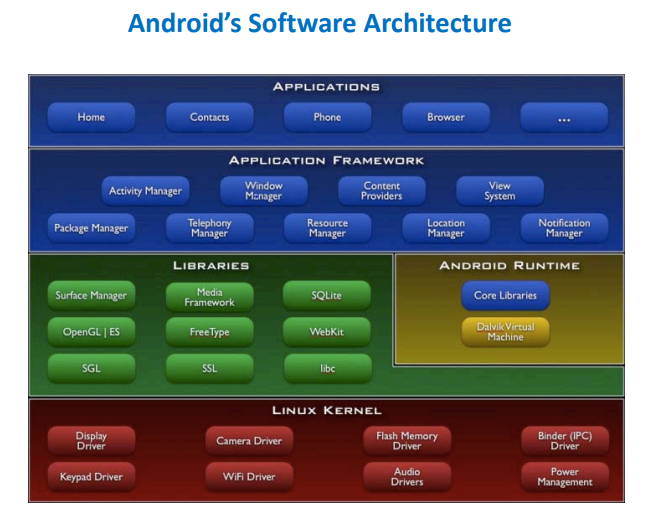
**CHƯƠNG 2. CƠ SỞ LÝ THUYẾT**

**2.1. Ứng dụng trên HĐH Android**

**2.1.1. Giới thiệu về HĐH Android**

Android là một hệ điều hành mã nguồn mở được phát triển bởi google cho các thiết bị thông minh, thiết bị di động có màn hình cảm ứng như điện thoại thông minh, máy tính bảng, smart tivi,... Với yếu tố mã nguồn mở, sử dụng trên ngôn ngữ lập trình Java với cộng động lập trình viên đông đảo đã thu hút các nhà phát triển ứng dụng và nhanh chóng đưa Android trở thành hệ điều hành phổ biến và có kho ứng dụng lớn nhất thế giới. Android là lựa chọn tốt nhất cho mục tiêu phổ cập thiết bị thông minh trên thế giới. Tính đến quý 2 năm 2017, Android đã chiếm 87,7% thị phần điện thoại thông minh trên thế giới với hơn 2 tỷ thiết bị đã được kích hoạt.

Kiến trúc của hệ điều hành android - gồm 5 tầng:



*Hình 2.1. Kiến trúc hệ điều hành Android*

* Tầng Applications: Tầng này chứa các ứng dụng cơ bản cho người dùng gọi điện, danh bạ, trình duyệt, camera,…
* Tầng Framework: Tầng chứa các API cần thiết để cho việc xây dựng app trên hệ điều hành. Quản lý các view, activity, lấy các thông tin như danh bạ, tin nhắn
* Tầng Libraries: Tầng gồm các thư viện cốt lõi được viết bằng c/c++ kết nối với phần cứng và các bộ phận ngoại vi như Surface Manager, camera, mic, …
* Tầng Android Runtime: Tầng thực hiện convert các ứng dụng sang định dạng nén Dalvik Executable (.dex), cung cấp các tính năng như thread, low-level memory management.
* Tầng Kernel: Là nhân lõi của hệ điều hành dựa trên hệ điều hành linux, chứa các tập lệnh, driver.

Ngôn ngữ lập trình và môi trường phát triển:

Lập trình android là quá trình lập trình viên sử dụng ngôn ngữ lập trình nào đó để xây dựng lên ứng dụng, tiện ích hoạt động trên hệ điều hành android.

Lập trình android có thể sử dụng nhiều ngôn ngữ lập trình như Java, Kotlin, Dart(Flutter), c++, python, …

Java chính là ngôn ngữ đơn giản và phổ biến nhất cho việc lập trình android, là ngôn ngữ hướng đối tượng với khả năng xứ lý tốt, cấu trúc rõ ràng công với kho thư viện, API khổng lồ được cung cấp bởi cộng đồng lập trình viên đông đào thì việc xử lý các yêu cầu cần độ phức tạp cao sẽ trở lên đơn giản hơn.

Kotlin: có nhiều nét tương đồng với Java, đây là ngôn ngữ có cú pháp đơn giản, ngắn gọn hơn. Kể từ 2017, nó được google ưu tiên hỗ trợ nhiều hơn.

Môi trường phát triển(IDE) dành riêng cho android chính là Android Studio. Đây là IDE mã nguồn mở có thể chạy trên cả Windows, Mac và Linux.

**2.1.2. Xây dựng ứng dụng gọi điện**

Tại trang web dành cho lập trình viên, google đã cung cấp tài liệu rất chi tiết về ứng dụng gọi điện. Đây là ứng dụng gọi điện cho phép người dùng nhận và thực hiện cuộc gọi âm thanh hoặc video. Nó có thể xây dựng giao diện khác với ứng dụng mặc định của hệ thống để có thể đáp ứng nhưng yêu cầu khác nhau.

Android framework sẽ chứa các gói thư viện android.telecom, các lớp cần thiết để lập trình viên có thể xây dựng một ứng dụng hoản chỉnh.

* Ứng dụng sẽ tương tác chính xácvới hệ thống viễn thông của thiết bị.
* Ứng dụng có thể giao tiếp được với các ứng dụng gọi điện khác nếu cũng viết theo tiểu chuẩn này.
* Framework sẽ giúp ứng dụng quán lý định tuyến âm thanh và video.
* Framework sẽ giúp ứng dụng quản lý phân phối độ ưu tiên của cuộc gọi.

Khai báo quyền trong Manifest

Trong file manifest của project cần khai báo các quyền cho hệ thống như:

* MANAGE\_OWN\_CALLS
* READ\_CALL\_LOG
* READ\_PHONE\_STATE
* READ\_PHONE\_NUMBERS



*Hình 2.1. Khai báo quyền cho ứng dụng gọi điện*

Triển khai service InCallService

Ứng dụng sẽ thông qua service này để quản lý các cuộc gọi điện thoại. Để có thể làm điều này thì ứng dụng cần được người dùng cài đặt làm ứng dụng điện thoại mặc định.

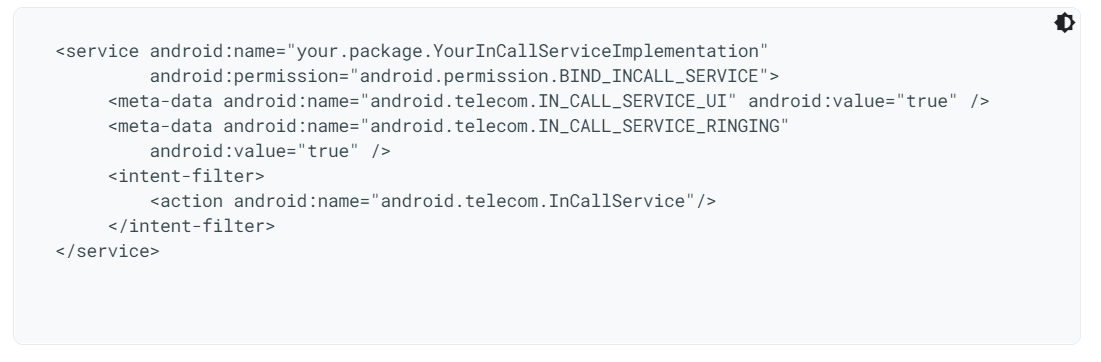
Ứng dụng điện thoại mặc định(default dialer/phone app) sẽ cung cấp giao diện cho thiết bị khi kết nối cuộc gọi và cung cấp các tiên ích như bắt đầu/kết thúc cuộc gọi, xem lịch sử cuộc gọi. Mỗi thiết bị sẽ có một ứng dụng default dialer/phone app mặc định của hệ thống. Để ứng dụng của chúng ta có thể thay thế ứng dụng hệ thống cần sử dụng RoleManager để yêu cầu đảm nhận vai trò RoleManager.ROLE\_DIALER.

Để đáp ứng vai trò RoleManager.ROLE\_DIALER ứng dụng cần đáp ứng yêu cầu:

* Xử lý định hướng Intent#ACTION\_DIAL. Có bàn phím số để người dùng có thể thực hiện quay số.
* Triển khai đây đủ API InCallService và có giao diện cho cuộc gói đến, cuộc gọi đi.

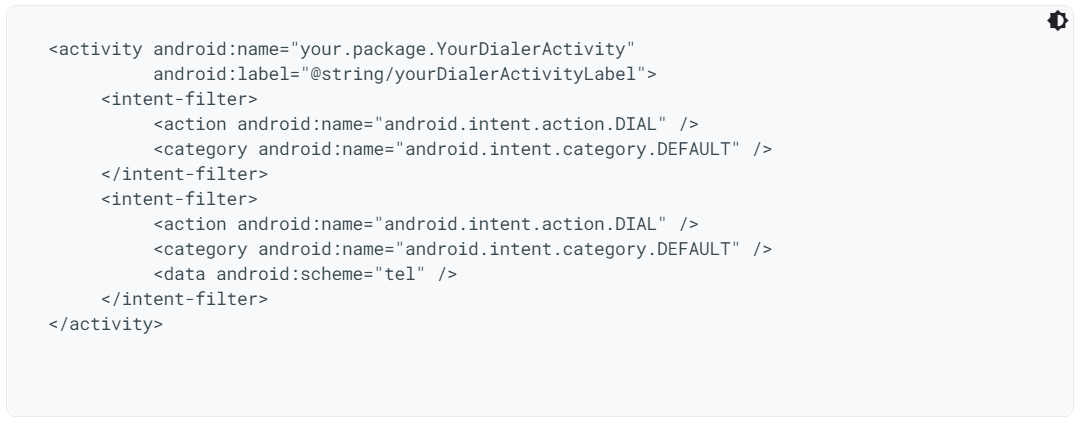
Nếu trong quá trình hoạt động ứng dụng gặp lỗi khi triển khai API InCallService thì hệ thống sẽ tự động chuyển định hướng cho ứng dụng mặc định ban đầu của hệ thống thực thi. Và thông báo tới người dùng đã gặp sự cố nhưng cuộc gọi của họ vẫn được tiếp túc trên ứng dụng hệ thống.

Khai báo service trong manifest là điều bắt buộc, với meta-data TelecomManager#METADATA\_IN\_CALL\_SERVICE\_UI để thông báo sẽ thay đổi giao diện mặc định và meta-data TelecomManager#METADATA\_IN\_CALL\_SERVICE\_RINGING để thông báo sẽ có nhạc chuông khi có cuộc gọi đến.



*Hình 2.2. Khai báo service InCallService trong manifest*

Ngoài ra còn cần khai báo định hướng Intent#ACTION\_DIAL cho màn hình sẽ làm giao diện gọi điện của ứng dụng.



*Hình 2.3. Khai báo màn hình giao diện gọi điện*

Để ứng dụng của chúng ta làm ứng dụng gọi điện mặc định cần sự đồng ý cho phép đến từ người dùng. Vì vậy khi người dùng truy cập vào ứng dụng cần hiện lên thống báo để người dùng có thể chọn cài đặt ứng dụng gọi điện mặc định.



*Hình 2.4. Hiện thông báo chọn ứng dụng điện thoại mặc định*

Tích hợp với thiết bị đeo tay cần:

* Khai báo quyền MANAGE\_ONGOING\_CALLS trong manifest
* Thiết bị đeo tay cần tích hợp CompanionDeviceManager API
* Triển khai InCallService này với quyền BIND\_INCALL\_SERVICE

Hiện thị thông báo cuộc gọi đến

InCallService#onCallAdded(Call) khi ứng dụng nhận cuộc gọi đến, cần hiện thị giao diện tương ứng cho người dùng. Sử dụng các API NotificationManager để tạo thông báo cuộc gọi đến mới.

TelecomManager#METADATA\_IN\_CALL\_SERVICE\_RINGING nếu được khia báo ứng dụng cần phát nhạc chuông thông báo cuộc gọi đến.



*Hình 2.5. Hiện thông báo khi có cuộc gọi đến*

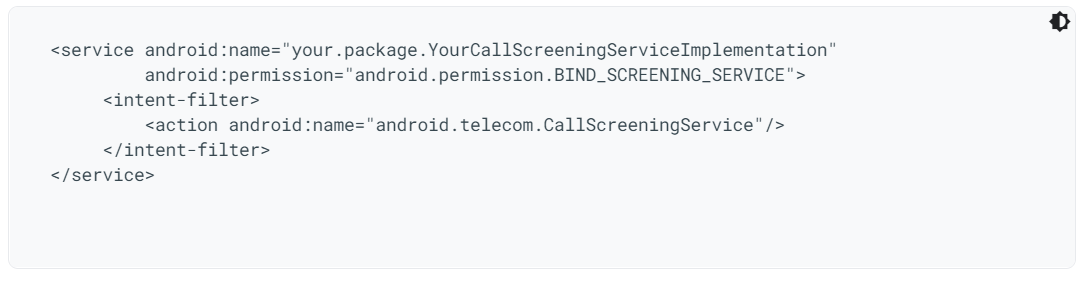
Khi ứng dụng nhận được cuộc gọi đến mới, ứng dụng sẽ tạo thông báo cho cuộc gọi đến và liên kết nó với kênh thông báo cuộc gọi đến của ứng dụng. Chúng ta có thể chỉ định PendingIntent trên thông báo để thiết lập khi chạm vào thông báo sẽ ngay lập tức hiện lên giao diện cuộc gọi toàn màn hình đã được thiết kế của ứng dụng. Nếu người dùng đang sử dụng thì cuộc gọi đến sẽ được hiện thị dưới dạng thông báo, còn người dùng không thao tác trên thiết bị thì cuộc gọi đến sẽ được hiện thị theo cách toàn màn hình.

Một số hàm InCallService cung cấp:

* canAddCall()-boolean: thiết bị có hỗ trợ cuộc gọi?
* getCallAudioState()-CallAudioState: lấy trạng thái hiện tại của âm thanh cuộc gọi
* onBringToForeground(boolean showDialpad): hiện thị bàn phím số lên
* onCallAdded(Call call): hoạt đông khí có thêm cuộc gọi vào phiên quản lý
* onCallAudioStateChanged(CallAudioState audioState): thay đổi trạng thái âm thanh
* onCallRemoved(Call call): hoạt động khi có hành động xóa cuộc gọi khỏi phiên quản lý
* onSilenceRinger(): tắt tiếng chuông điện thoại nếu cuộc gọi đổ chuông
* onUnbind(Intent intent): hoạt động khi đã kết thuốc hoàn toàn cuộc gói đến
* requestBluetoothAudio(BluetoothDevice bluetoothDevice): yêu cầu được định tuyến âm thanh đến thiết bị bluetooth
* Sets the audio route (speaker, bluetooth, etc...): Thiết lập tuyến âm thanh
* setMuted(boolean state): thiết lập trạng thái tắt Mic

**2.1.3. Chặn cuộc gọi – CallScreeningService**

Service này được triển khai bởi ứng dụng gọi điện mặc định(Default call phone app) hoặc ứng dụng bên thứ ba đã được cấp quyền để thực hiện các hạnh động trước khi cuộc gọi đến hiện thị cho người dùng. Nó cũng có thể xem các cuộc gọi đi với mục đích cung cấp dịch caller ID services cho cuộc gọi đó.



*Hình 2.6. Đăng ký CallScreeningService trong Manifest*

Hai chức năng chính của một CallScreeningService

* Chặn/sàng lọc cuộc gọi, service có thể lựa chọn cuộc gọi nào sẽ đổ chuông trên thiết bị của người dùng và cuộc gọi nào sẽ được gửi ẩm thâm đến thư thoại.
* Định danh cuộc gọi, service cung cấp chức năng định danh cuộc gọi và có thể hiển thị giao diện tùy ý chưa thông tin định danh cho cuộc gọi.

Triển khai CallScreeningService

Telecom sẽ liên kết với một ứng dụng duy nhất do người dùng chọn, ứng dụng này triển khai API CallScreeningService khi có các cuộc gọi đến và đi mới. Để ứng dụng có khả năng trên thì cần phải yêu cầu quyền thực hiện vai trò sàng lọc cuộc gọi.



*Hình 2.7. Code xin quyền ROLE\_CALL\_SCREENING*

Vòng đời CallScreeningService

Ứng dụng được người dùng chọn làm RoleManager.ROLE\_CALL\_SCREENING sẽ nhận được cuộc gọi trước khi đổ chuông và khi cuộc gọi đi được thực hiện. Nó sẽ gọi phương thức onScreenCall(android,telecom.Call.Details) để cung cấp cho dịch vụ của chúng ta thông tin chi tiết về cuộc gọi.

Đối với các cuộc gọi đến, CallScreeningService phải gọi responseToCall(android.telecom.CallScreeningService.CallResponse) trong vòng 5 giây kể từ khi bị ràng buộc để cho nền tảng biết cuộc gọi có nên bị chặn hay không. Cần phải thực hiện các thao tác sàng lọc kịp thời vì thiết bị của vì thiết bị của người dùng sẽ không bắt đầu đổ chuông cho đến khi nhận được phản hồi(hoặc hết thời gian chờ). Một CallScreeningService có thể chọn thực hiện tra cứu cơ sở dữ liệu cục bộ để giúp xác định xem một cuộc gọi có nên được sàng lọc hay không. Cần chú ý để đảm bảo thời gian chờ không bị nhấn liên tục, gây ra sự chậm trễ trong luồng cuộc gọi đến.

Một số hàm hỗ trợ

* Ibinder onBind(Intent intent): trả về kênh liên lạc của service
* abstract void onScreenCall(Call.Details callDetails): Được gọi khi có cuộc gọi đến hoặc cuộc gọi đi mới
* boolean onUnbind(Intent intent): được gọi khi tất cả các máy khách đã ngắt kết nối

Chi tiết về hàm onScreenCall

Được gọi khi có cuộc gọi đến hoặc cuộc gọi đi mới. Một CallScreeningService phải cho biết cuộc gọi đến có được phép hay không bằng cách gọi CallScreeningService#responseToCall(Call.Details, CallScreeningService.CallResponse). Ứng dụng của chúng ta có thể biết một cuộc gọi có phải là cuộc gọi đến hay không bằng cách kiểm tra xem Call.Details#getCallDirection() có phải là Call hay không.

Lưu ý: Một CallScreeningService phải trả lời cuộc gọi trong vòng 5 giây. Sau thời gian này, khung sẽ hủy liên kết khỏi CallScreeningService và bỏ qua phản hồi của nó.

**2.2. Hệ thống Backend Server NodeJs**

**2.2.1. Giới thiếu về NodeJS**

NodeJS là một nền tảng (Platform) được xây dựng trên nền tảng Javascript V8 Engine của Chorme. Được phát triển vào năm 2009 bởi google trong đó phần core sử dụng JavaScript và C++ cho phép xử lý với hiệu năng cao phù hợp với các ứng dụng đòi hỏi real time, xử lý nhanh với cơ chế lập trình bất đồng bộ. Với mục đích xây dựng các ứng dụng mạng nhanh chóng và có thể mở rộng được một cách dễ dàng. NodeJS gọn nhẹ và hiểu quả cho các công việc chuyển sâu về dữ liệu theo thời gian thực, nhờ sử dụng mô hình I/O lập trình theo sự kiện, non-blocking.

Chế độ đồng bộ các dòng lệnh sẽ được thực thi tuần tự, dòng trên thực thi xong thì dòng dưới sẽ bắt đầu được thược thi.

Chế độ bất đồng bộ tất cả dòng lệnh sẽ được thục thi cùng một lúc.

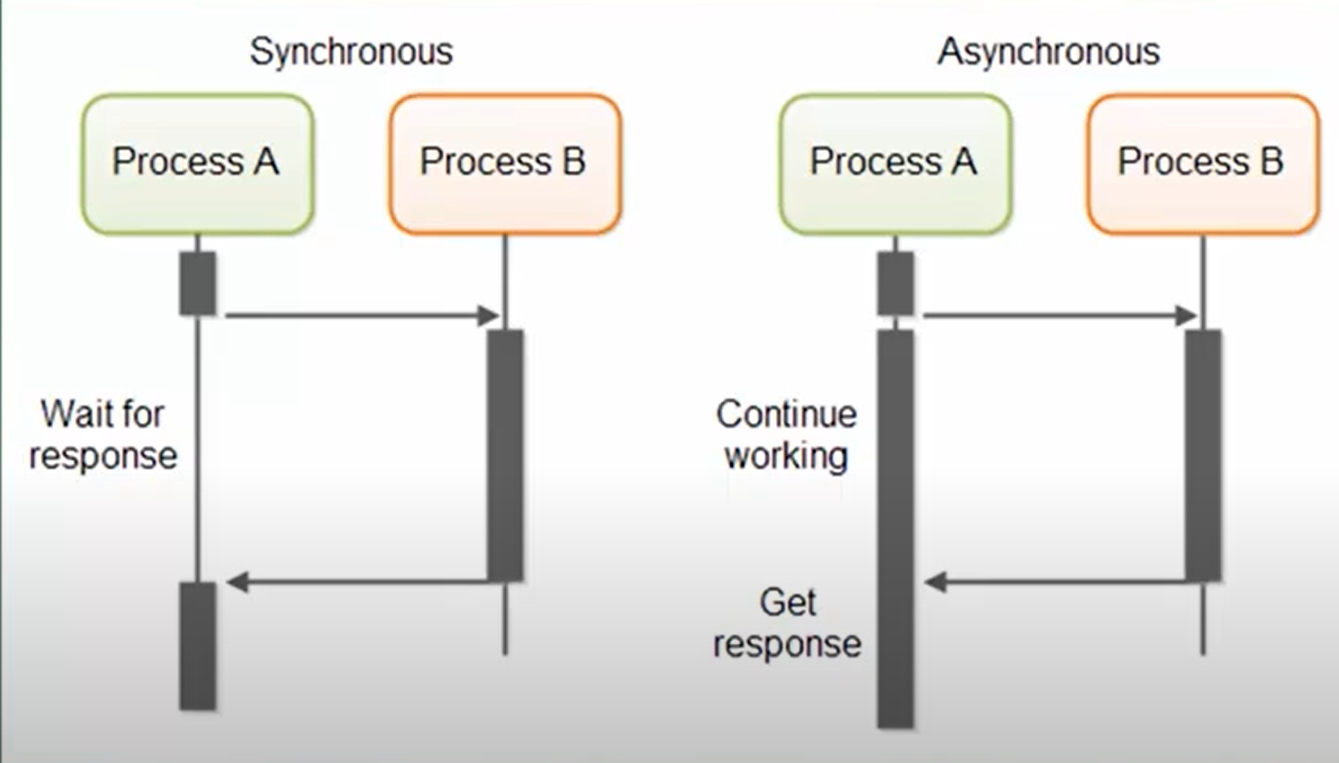
NodeJS là môi trường runtime đa nền tảng mã nguồn mở, thích hợp phát triển các ứng dụng mạng và ứng dụng server-side.

Một số đặc điểm cơ bản NodeJS

Lập trình hướng sự kiện, không đồng bộ: API trong thư viện NodeJS đều không đồng bộ, điều này cũng có nghĩa việc thực thi của một server NodeJS sẽ không chờ một API nào trả về data. Server sẽ chuyển sang API kế tiếp sau khi gọi API trowcs đó và sử dụng cơ chế thông báo cúa Events trong NodeJS để nhận được phản hồi từ các API đã gọi trước đó.

Cực kỳ nhanh chóng: Được xây dưng trên Công cụ JavaScript V8 của google chrome vì vậy thư viện NodeJS có khả năng xử lý mẫ rất nhanh.

Đơn luồng/Single thread: NodeJS sử dụng mô hình đơn luồng với cơ chế vòng lặp sự kiện Events. Cơ chế này cho phép máy chủ phản hồi non-blocking với khả năng cung cấp dịch vụ cho một số lượng yêu cầu lớn hơn so với các máy chủ truyền thống.



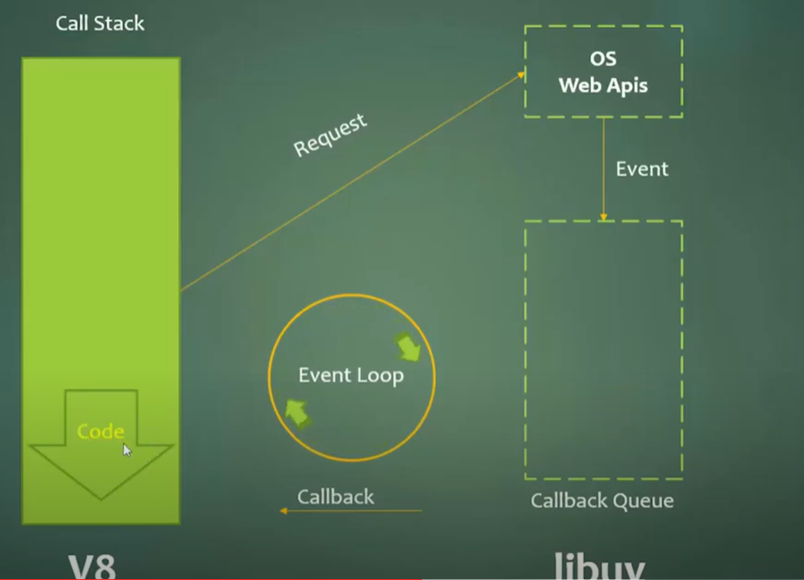
*Hình 2.8. Synchronous và Asynchronous*

Không có buffer: Ứng dụng NodeJS sẽ không có vùng nhớ tam thời(buffer). Cá ứng dụng chỉ đơn giản sẽ xuất dữ liệu theo khối.

License: NodeJS được phát hành theo giấy phép MIT.

Một số thành phần cơ bản

Non Blocking IO-Event Loop: là cơ chế được NodeJS dùng để xử lý đồng thời nhiều request trên single thread mà không cần đợi hoàn thành xử lý của resquest trước đó.



*Hình 2.9. Cơ chế Event Loop*

Package manager: là một kho lưu trữ các package/module để public cho công đồng. Nó dễ dàng cài cặt, quản lý version, quản lý các gói phụ thuộc. Hai package manager phổ biến nhất là npm, yarn

Socket.io: là một công cụ thời gian thực rất nhanh và đáng tin cậy. Nó cho phép giao tiếp 2 chiều theo thời gian thực thông qua các sự kiện. Gồm 2 phần chính là một máy chủ NodeJS và một Javascript client library cho client. Javascript socket.io đã được thêm vào một số ngôn ngữ khác như Java, C++, Dart, Swift.

**2.2.2. MySQL trong NodeJS**

Ngôn ngữ SQL

SQL là ngôn ngữ tiêu chuẩn cho các hệ thống quản lý cơ sở dữ liệu quan hệ. Một cơ sở dữ liệu quan hệ sẽ quản lý dữ liệu dưới dạng các bảng. Các câu lệnh SQL sẽ thức thi các tác vụ như cập nhật dữ liệu hoặc truy xuất dữ liệu từ các bảng. Một số hệ quản lý cơ sở dữ liệu sử dụng SQL là MySQL, Oracle, Microsoft SQL Server,..

Hệ quản trị cơ sở dữ liệu MySQL

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở(RDBMS) dựa trên ngôn ngữ truy vấn có cấu trúc(SQL) được phát triển, phân phối và hỗ trợ bởi tập đoàn Oracle. MySQL chạy trên hầu hết tất cả các nền tàng, bao gồm cả Linux, UNIX và Windows.

* MySQL được phát hành theo giấy phép nguồn mở
* MySQL sử dụng dạng chuẩn của ngôn ngữ dữ liệu SQL
* MySQL được tích hợp cho nhiều ngôn ngữ như JAVA, C, C++, NodeJS, PHP
* MySQL hoạt động nhanh và tốt trên các tập dữ liệu lớn. Hỗ trợ lên tới 50 triệu hàng trong 1 bảng. Giới hạn kích thước tệp mặc định cho 1 bẳng là 4GB và có thể nâng thêm theo lý thuyết có thể lên tới 8TB.

Một số ưu điểm của MySQL như dễ dàng sử dụng, tính bảo mật cao, khả năng mở rộng và khôi phục.

So sánh giữa SQL và MySQL

|  |  |  |
| --- | --- | --- |
| Tiêu chí | SQL | MySQL |
| Kiểu | Ngôn ngữ truy vấn cấu trúc | Hệ cơ sở dữ liệu sử dụng SQL |
| Hỗ trợ kết nối | SQL không cung cấp trình kết nối | Cung cấp công cụ MySQL workbench |
| Múc đích | Để truy vấn và vận hành hệ thống cơ sở dữ liệu | Cho phép xử lý dữ liệu, lưu trữ, truy vấn theo bảng |
| Sử dụng | Mã và lệnh SQL được sử dụng ở nhiều hệ cơ sở dữ liệu khác nhau | Được sử dụng làm cơ sở dữ liệu RDBMS |
| Cập nhật | Ngôn ngữ là cố định, mã lệnh vẫn được giữa nguyên | Có các bản cập nhật thường xuyên |

Một số câu truy vấn quan trọng

INSERT: chèn dữ liệu vào trong bảng

SELECT: lấy dữ liệu trong bảng

UPDATE: cập nhật dữ liệu đã tồn tại

DELETE: xóa dữ liệu từ một bảng

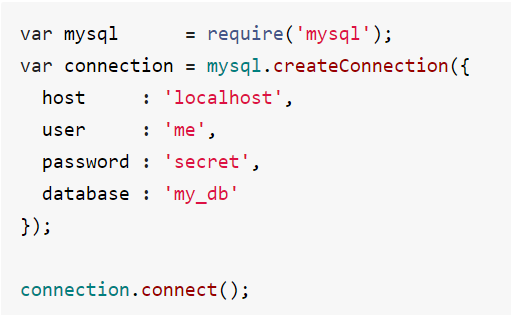
CREATE: tạo ra một bảng dữ liệu

DROP: xóa bảng dữ liệu

Sử dụng MySQL với NodeJS

Tiến hành cài đặt MySQL database và phần mềm quản lý MySQL workbench hoặc Xampp

Cài đặt module mysql qua npm: npm install mysql



*Hình 2.10. Cấu hình connect mysql*

Trong đó:

* host: hostname database của bạn
* user: username để đăng nhập vào mysql
* password: mật khẩu của username để đăng nhập vào mysql
* database: database các bạn cần kết nối

**2.2.3. RESTful API**

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng API service. Tập trung vào quản lý trao đổi tài nguyên hệ thống(văn bản, file, âm thanh, video,..), tài nguyên được định danh và truyền tải qua HTTP.

Trong đó:

API(Application Programming Interface) là một tập các quy tắc và cơ chế để các ứng dụng tương tác qua lại. API sẽ trả về kết quả data dưới dạng cơ bản là JSON, XML,..

REST(Representational State Transfer) là một dạng chuyển đối cấu trúc dữ liệu, một kiểu kiến trúc cho viết API. Gồm các phương thwucs HTTP đợn giản để giao tiếp giữa các loại thiết bị khác nhau như GET, POST, DELETE.

Chức năng quan trọng nhất của REST là quy định cách sử dụng các HTTP method(GET, POST, PUT, DELETE). RESTful không quy định logic code và không giới hạn bởi ngôn ngữ lập trình, với bất kỳ ngôn ngữ hay framework nào cũng có thể tiếp cận với RESTful API.

REST hoạt động chủ yếu dựa vào giao thức HTTP:

* GET (SELECT): trả về dữ liệu cần thiết cho device user
* POST (CREATE): gửi yêu cầu tạo data mới lên hệ thống
* PUT (UPDATE): cập nhật cho data đã tồn tại trên hệ thống
* DELETE (DELETE): xóa data

Status code trong API khi thực hiện resquest API:

* 200 OK – Trả về thành công cho những phương thức
* 201 Created – Trả về khi một Resouce vừa được tạo thành công.
* 204 No Content – Trả về khi Resource xoá thành công.
* 304 Not Modified – Client có thể sử dụng dữ liệu cache.
* 400 Bad Request – Request không hợp lệ
* 401 Unauthorized – Request cần có auth.
* 403 Forbidden – bị từ chối không cho phép.
* 404 Not Found – Không tìm thấy resource từ URI
* 405 Method Not Allowed – Phương thức không cho phép với user hiện tại.
* 410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.
* 415 Unsupported Media Type – Không hỗ trợ kiểu Resource này.
* 422 Unprocessable Entity – Dữ liệu không được xác thực
* 429 Too Many Requests – Request bị từ chối do bị giới hạn