|  |
| --- |
| TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  **VIỆN ĐIỆN TỬ - VIỄN THÔNG**  logo_128  ĐỒ ÁN  **TỐT NGHIỆP ĐẠI HỌC**  **Đề tài:**  **HỆ THỐNG CHO THUÊ – CHIA SẺ**  **XE ĐẠP CÔNG CỘNG**  Sinh viên thực hiện: PHẠM HỒNG ĐẠT  Lớp ĐTVT 09 – K62  Giảng viên hướng dẫn: TS. PHÙNG THỊ KIỀU HÀ  Cán bộ phản biện:  Hà Nội, 7-2022 |

**ĐÁNH GIÁ QUYỂN ĐỒ ÁN TỐT NGHIỆP**

(Dùng cho cán bộ phản biện)

Giảng viên đánh giá:

Họ và tên sinh viên: MSSV:

Tên đồ án:

**Chọn các mức điểm phù hợp cho sinh viên trình bày theo các tiêu chí dưới đây:**

Rất kém (1); Kém (2); Đạt (3); Giỏi (4); Xuất sắc (5)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Có sự kết hợp giữa lý thuyết và thực hành (20)** | | | | | | |
| 1 | Nêu rõ tính cấp thiết và quan trọng của đề tài, các vấn đề và các giả thuyết (bao gồm mục đích và tính phù hợp) cũng như phạm vi ứng dụng của đồ án | 1 | 2 | 3 | 4 | 5 |
| 2 | Cập nhật kết quả nghiên cứu gần đây nhất (trong nước/quốc tế) | 1 | 2 | 3 | 4 | 5 |
| 3 | Nêu rõ và chi tiết phương pháp nghiên cứu/giải quyết vấn đề | 1 | 2 | 3 | 4 | 5 |
| 4 | Có kết quả mô phỏng/thưc nghiệm và trình bày rõ ràng kết quả đạt được | 1 | 2 | 3 | 4 | 5 |
| **Có khả năng phân tích và đánh giá kết quả (15)** | | | | | | |
| 5 | Kế hoạch làm việc rõ ràng bao gồm mục tiêu và phương pháp thực hiện dựa trên kết quả nghiên cứu lý thuyết một cách có hệ thống | 1 | 2 | 3 | 4 | 5 |
| 6 | Kết quả được trình bày một cách logic và dễ hiểu, tất cả kết quả đều được phân tích và đánh giá thỏa đáng. | 1 | 2 | 3 | 4 | 5 |
| 7 | Trong phần kết luận, tác giả chỉ rõ sự khác biệt (nếu có) giữa kết quả đạt được và mục tiêu ban đầu đề ra đồng thời cung cấp lập luận để đề xuất hướng giải quyết có thể thực hiện trong tương lai. | 1 | 2 | 3 | 4 | 5 |
| **Kỹ năng viết quyển đồ án (10)** | | | | | | |
| 8 | Đồ án trình bày đúng mẫu quy định với cấu trúc các chương logic và đẹp mắt (bảng biểu, hình ảnh rõ ràng, có tiêu đề, được đánh số thứ tự và được giải thích hay đề cập đến trong đồ án, có căn lề, dấu cách sau dấu chấm, dấu phẩy v.v), có mở đầu chương và kết luận chương, có liệt kê tài liệu tham khảo và có trích dẫn đúng quy định | 1 | 2 | 3 | 4 | 5 |
| 9 | Kỹ năng viết xuất sắc (cấu trúc câu chuẩn, văn phong khoa học, lập luận logic và có cơ sở, từ vựng sử dụng phù hợp v.v.) | 1 | 2 | 3 | 4 | 5 |
| **Thành tựu nghiên cứu khoa học (5) (chọn 1 trong 3 trường hợp)** | | | | | | |
| 10a | Có bài báo khoa học được đăng hoặc chấp nhận đăng/đạt giải SVNC khoa học giải 3 cấp Viện trở lên/các giải thưởng khoa học (quốc tế/trong nước) từ giải 3 trở lên/ Có đăng ký bằng phát minh sáng chế | 5 | | | | |
| 10b | Được báo cáo tại hội đồng cấp Viện trong hội nghị sinh viên nghiên cứu khoa học nhưng không đạt giải từ giải 3 trở lên/Đạt giải khuyến khích trong các kỳ thi quốc gia và quốc tế khác về chuyên ngành như TI contest. | 2 | | | | |
| 10c | Không có thành tích về nghiên cứu khoa học | 0 | | | | |
| **Điểm tổng** | | **/50** | | | | |
| **Điểm tổng quy đổi về thang 10** | |  | | | | |

***Nhận xét khác của cán bộ phản biện***

|  |  |
| --- | --- |
|  | Ngày: … / … / 20…  **Người nhận xét**  (Ký và ghi rõ họ tên) |

**ĐÁNH GIÁ QUYỂN ĐỒ ÁN TỐT NGHIỆP**

(Dùng cho cán bộ phản biện)

Giảng viên đánh giá:

Họ và tên sinh viên: MSSV:

Tên đồ án:

**Chọn các mức điểm phù hợp cho sinh viên trình bày theo các tiêu chí dưới đây:**

Rất kém (1); Kém (2); Đạt (3); Giỏi (4); Xuất sắc (5)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Có sự kết hợp giữa lý thuyết và thực hành (20)** | | | | | | |
| 1 | Nêu rõ tính cấp thiết và quan trọng của đề tài, các vấn đề và các giả thuyết (bao gồm mục đích và tính phù hợp) cũng như phạm vi ứng dụng của đồ án | 1 | 2 | 3 | 4 | 5 |
| 2 | Cập nhật kết quả nghiên cứu gần đây nhất (trong nước/quốc tế) | 1 | 2 | 3 | 4 | 5 |
| 3 | Nêu rõ và chi tiết phương pháp nghiên cứu/giải quyết vấn đề | 1 | 2 | 3 | 4 | 5 |
| 4 | Có kết quả mô phỏng/thưc nghiệm và trình bày rõ ràng kết quả đạt được | 1 | 2 | 3 | 4 | 5 |
| **Có khả năng phân tích và đánh giá kết quả (15)** | | | | | | |
| 5 | Kế hoạch làm việc rõ ràng bao gồm mục tiêu và phương pháp thực hiện dựa trên kết quả nghiên cứu lý thuyết một cách có hệ thống | 1 | 2 | 3 | 4 | 5 |
| 6 | Kết quả được trình bày một cách logic và dễ hiểu, tất cả kết quả đều được phân tích và đánh giá thỏa đáng. | 1 | 2 | 3 | 4 | 5 |
| 7 | Trong phần kết luận, tác giả chỉ rõ sự khác biệt (nếu có) giữa kết quả đạt được và mục tiêu ban đầu đề ra đồng thời cung cấp lập luận để đề xuất hướng giải quyết có thể thực hiện trong tương lai. | 1 | 2 | 3 | 4 | 5 |
| **Kỹ năng viết quyển đồ án (10)** | | | | | | |
| 8 | Đồ án trình bày đúng mẫu quy định với cấu trúc các chương logic và đẹp mắt (bảng biểu, hình ảnh rõ ràng, có tiêu đề, được đánh số thứ tự và được giải thích hay đề cập đến trong đồ án, có căn lề, dấu cách sau dấu chấm, dấu phẩy v.v), có mở đầu chương và kết luận chương, có liệt kê tài liệu tham khảo và có trích dẫn đúng quy định | 1 | 2 | 3 | 4 | 5 |
| 9 | Kỹ năng viết xuất sắc (cấu trúc câu chuẩn, văn phong khoa học, lập luận logic và có cơ sở, từ vựng sử dụng phù hợp v.v.) | 1 | 2 | 3 | 4 | 5 |
| **Thành tựu nghiên cứu khoa học (5) (chọn 1 trong 3 trường hợp)** | | | | | | |
| 10a | Có bài báo khoa học được đăng hoặc chấp nhận đăng/đạt giải SVNC khoa học giải 3 cấp Viện trở lên/các giải thưởng khoa học (quốc tế/trong nước) từ giải 3 trở lên/ Có đăng ký bằng phát minh sáng chế | 5 | | | | |
| 10b | Được báo cáo tại hội đồng cấp Viện trong hội nghị sinh viên nghiên cứu khoa học nhưng không đạt giải từ giải 3 trở lên/Đạt giải khuyến khích trong các kỳ thi quốc gia và quốc tế khác về chuyên ngành như TI contest. | 2 | | | | |
| 10c | Không có thành tích về nghiên cứu khoa học | 0 | | | | |
| **Điểm tổng** | | **/50** | | | | |
| **Điểm tổng quy đổi về thang 10** | |  | | | | |

***Nhận xét khác của cán bộ phản biện***

|  |  |
| --- | --- |
|  | Ngày: … / … / 20…  **Người nhận xét**  (Ký và ghi rõ họ tên) |

# LỜI NÓI ĐẦU

Theo báo cáo năm 2018 của Cơ quan năng lượng quốc tế (IEA), giao thông vận tải đóng góp 24.34% lượng khí thải carbon mỗi năm. Xét riêng lĩnh vực giao thông vận tải, các loại ô tô hạng nhẹ, ô tô tải và ô tô bus lần lượt chiếm 44%, 27% và 6% lượng khí thải carbon mỗi năm. Ở Việt Nam, tại thành phố Hà Nội và thành phố Hồ Chí Minh là nơi bị ô nhiễm không khí nặng nhất của cả nước, có nhiều thời điểm bụi mịn (PM 2.5) bao phủ cả bầu trời làm hạn chế tầm nhìn, ảnh hưởng rất lớn đến sức khỏe của người dân. Giảm thiểu sử dụng các phương tiện cá nhân có xả thải ra mỗi trường đang là một giải pháp tốt không chỉ về vấn đề môi trường mà còn giải quyết được tình trạng tắc đường tại các thành phố lớn.

Đạp xe để đi làm là một sự lựa chọn vừa tiết kiệm hơn so với phương tiện dùng nhiên liệu, vừa rèn luyện được sức khoẻ. Xe đạp cũng là một phương tiện tốt để giảm được tình trạng kẹt xe tại các con đường trung tâm giờ tan tầm. Ngoài ra đạp xe còn để thư giãn, rèn luyện sức khỏe. Nhiều câu lạc bộ, hội nhóm mê xe đạp thể thao được thành lập nhằm kết nối những người có cùng đam mê, sở thích, qua đó tổ chức các buổi gặp gỡ và giao lưu bằng xe đạp thể thao, trải nghiệm những cung đường mới lạ. Đạp xe đúng cách giúp xây dựng cơ bắp và xương chắc khỏe, tốt cho tim mạch, tốt cho trí não, … Sau một ngày dài với đủ thứ công việc bộn bề, 30 phút đạp xe sẽ là khoảng thời gian quý giá cho cơ thể được thư giãn và phục hồi.

Tuy nhiên, không phải ai cũng sẽ sở hữu 1 chiếc xe cho riêng mình vì họ không có thói quen sử dụng hàng ngày, đôi khi chỉ ngẫu hứng muốn đạp xe ngắm phố phường hay còn là do giá để mua một chiếc xe đạp mới không phải là rẻ. Vì vậy, dịch vụ cho thuê xe đạp theo giờ ngày càng trở nên phổ biến và thu hút rất nhiều sự quan tâm. Dù vậy nhưng không phải ai cũng biết được địa chỉ của các quán cho thuê xe đạp và liệu địa chỉ đang muốn đến thuê xe còn có đang còn xe để cho thuê không. Khi thuê xe thì cần nộp tiền trước và nộp chứng mình thư hoặc giấy tờ các nhân có thể làm chứng. Giá tiền thuê xe bị đẩy lên cao do luôn cần người túc trực để bàn giao xe hay là nhận lại xe.

Việc thuê xe sẽ dễ dàng và nhanh chóng hơn trong thời đại phát triển công nghệ số 4.0 như hiện nay khi chỉ cần một chiếc Smartphone. Nhận thấy việc dùng điện thoại thông minh để tìm địa điểm thuê xe cũng như thành toán là một biện pháp thông minh, tiết kiệm được chi phí và thời gian so với việc thuê xe truyền thống nên em đã lựa chọn xây dựng “hệ thống cho thuê – chia sẻ xe đạp công cộng” để nghiên cứu trong Đồ Án Tốt Nghiệp của mình. Hệ thống cần phải đáp ứng được khả năng tiếp giữa thiết bị khoá thông mình trên xe đạp và server, giữa server và mobile app. Việc truyền tải thông tin giữa khoá và server được thực hiện bằng giao thức MQTT (Message Queuing Telemetry Transport). Giao tiếp giữa server và mobile app được thực hiện bằng giao thức HTTP (HyperText Transfer Protocol) kết hợp giao thức WebSocket.. Với kết quả này, em hy vọng sẽ mang lại một giải pháp tốt cho bài toán thuê – chia sẻ xe đạp thông minh ở Việt Nam với chi phí thấp và cách triển khai dễ dàng.

# LỜI CAM ĐOAN

Em là Phạm Hồng Đạt, mã số sinh viên 20172459 là sinh viên lớp ĐTVT.09, khóa 62. Giảng viên hướng dẫn Đồ án Tốt nghiệp của em là TS. Phùng Thị Kiều Hà. Em xin cam đoan rằng toàn bộ mọi nội dung được trình bày trong đồ án *Hệ thống cho thuê- chia sẻ xe đạp công cộng* là kết quả thu được từ quá trình tìm tòi và nghiên cứu em dưới sự hướng dẫn trực tiếp của giảng viên hướng dẫn và TS. Phùng Thị Kiều Hà có toàn quyền sử dụng các kết quả trong đồ án này. Các dữ liệu được nêu trong đồ án là hoàn toàn trung thực và phản ánh đúng các kết quả đã thực hiện được trên thực tế. Mọi trích dẫn trong báo cáo này đều được tuân thủ các quy định về quyền sở hữu trí tuệ, các tài liệu tham khảo đã được liệt kê rõ ràng. Em xin hoàn toàn chịu trách nhiệm với những nội dung được trình bày trong đồ án này.

Hà Nội, ngày 20 tháng 7 năm 2022

**Người cam đoan**

**Phạm Hồng Đat**

**MỤC LỤC**

[LỜI NÓI ĐẦU v](#_Toc110111731)

[LỜI CAM ĐOAN i](#_Toc110111732)

[DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT i](#_Toc110111733)

[DANH MỤC HÌNH VẼ ii](#_Toc110111734)

[DANH MỤC BẢNG BIỂU iii](#_Toc110111735)

[TÓM TẮT ĐỒ ÁN iv](#_Toc110111736)

[PHẦN MỞ ĐẦU vi](#_Toc110111737)

[CHƯƠNG 1. CƠ SỞ HÌNH THÀNH HỆ THỐNG 1](#_Toc110111738)

[1.1 Vấn đề ô nhiễm mỗi trường, tắc đường do xe cơ giới và các biện pháp hiện có 1](#_Toc110111739)

[1.1.1 Vấn đề ô nhiễm mỗi trường, tắc đường do xe cơ giới ở nước ta 1](#_Toc110111740)

[CHƯƠNG 2. LÝ THUYẾT VỀ JAVA 2](#_Toc110111741)

[2.1 Java Core 2](#_Toc110111742)

[2.1.1 Biến nguyên thuỷ (primitive) và tham chiếu (reference) 2](#_Toc110111743)

[2.1.2 Auto boxing và auto unboxing trong lớp Wrapper 3](#_Toc110111744)

[2.1.3 Lớp String 4](#_Toc110111745)

[2.1.4 Biểu thức chính quy (Regular expression) 5](#_Toc110111746)

[2.2 Lập trình hướng đối tượng (oop) trong Java 5](#_Toc110111747)

[2.2.1 Lớp Object 5](#_Toc110111748)

[2.2.2 Các tính chất của lập trình hướng đối tượng trong Java 6](#_Toc110111749)

[2.2.3 Đa kế thừa 7](#_Toc110111750)

[2.2.4 Abstract class và Interface 8](#_Toc110111751)

[2.2.5 Generic và Collection 9](#_Toc110111752)

[2.2.6 Stream Api và Lamda Expressions 10](#_Toc110111753)

[2.3 Bảo mật trong Java 10](#_Toc110111754)

[2.3.1 Phòng chống được lỗi XSS 11](#_Toc110111755)

[2.3.2 Phòng chống được lỗi SQL Injection 11](#_Toc110111756)

[2.3.3 Mã hoá dữ liệu nhậy cảm 12](#_Toc110111757)

[CHƯƠNG 3. THIẾT KẾ HỆ THỐNG 13](#_Toc110111758)

[3.1 Thiết kế hệ thống 13](#_Toc110111759)

[3.1.1 Khối IoT Messaging server và khối IoT Messaging client 13](#_Toc110111760)

[3.1.2 Khối backend 14](#_Toc110111761)

[3.1.3 User app 18](#_Toc110111762)

[3.1.4 Admin web 18](#_Toc110111763)

[3.1.5 Khối Google Map Platform 19](#_Toc110111764)

[3.1.6 Khối Device 19](#_Toc110111765)

[3.1.7 Khối Database 19](#_Toc110111766)

# DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

Xem Mục 1.3.6

# DANH MỤC HÌNH VẼ

[Hình 1.1 Sơ đồ khối của hệ thống 3](#_Toc9951164)

(Xem thêm Mục 1.3.7)

# DANH MỤC BẢNG BIỂU

[Bảng 1.1 Kết quả thí nghiệm 3](#_Toc9951165)

(Xem thêm Mục 1.3.8)

TÓM TẮT ĐỒ ÁN

1. **Tiếng Việt**

Hiện nay, hệ thống giao thông công cộng như xe buýt, xe buýt nhanh, tàu đường sắt trên cao đang thử nghiệm, vẫn chưa đáp ứng được nhu cầu tăng trưởng của thành phố. Bên cạnh đó, với xu hướng năng lượng xanh, giảm phát thải CO2, xe đạp là phương tiện giao thông đang được chú ý tại nhiều thành phố. Thực tế, với các độ thi lớn (megacity), việc sử dụng phương tiện cá nhân (ô tô con) đi từ các vùng ngoại ô vào vùng rìa và sau đó dùng phương tiện công cộng (bus, metro, tram…) hoặc các phương tiện thân thiện môi trường như xe đạp sẽ giúp giảm lượng xe cá nhân trong vùng trung tâm, vùng lõi đô thị. Trong hoàn cảnh đó bằng việc lựa chọn đề tài “*Hệ thống cho thuê – chia sẻ xe đạp công cộng*”, em hy vọng có thể đưa ra một giải pháp đơn giản về mặt tính toán cũng như tiết kiệm chi phí để giải quyết bài toán đã nêu trên.

Sau quá trình tìm hiểu và nghiên cứu, em đã hoàn thành việc thiết kế một hệ thống có tác dụng cho người dùng tìm kiếm các bãi đỗ xe đạp, thuê xe và người quản lý kiểm soát được thông tin trong hệ thống. Kỹ thuật quan trọng nhất trong hệ thống chính là việc truyền tải thông tin giữa khoá và server được thực hiện bằng giao thức MQTT (Message Queuing Telemetry Transport), giao tiếp giữa server và mobile app được thực hiện bằng giao thức HTTP (HyperText Transfer Protocol) kết hợp giao thức WebSocket.

1. **Tiếng Anh (English)**

Currently, public transport systems such as buses, express buses, and elevated railways are being tested, which have not yet met the growth needs of the city. Besides, with the trend of green energy, reducing CO2 emissions, bicycles are a transport that are being noticed in many cities. In fact, with large cities (megacity), the use of private vehicles (cars) to go from the suburbs to the downtown and then use public transport (bus, metro, tram...) or environmentally friendly vehicles such as bicycles will help reduce the number of private vehicles in the central and urban core areas. In that situation, by choosing the topic "Public bicycle rental - sharing system", I hope to be able to come up with a simple solution in terms of calculation as well as cost savings to solve the problem. calculation mentioned above.

After a process of understanding and research, i have completed the design of a system that works for users to search for bicycle parking, rent a car and the manager to control the information in the system. The most important technique in the system is that the transmission of information between the smart key and the server is done using the MQTT protocol (Message Queuing Telemetry Transport), the communication between the server and the mobile app is done using the HTTP protocol (HyperText Transfer Protocol) combined with the WebSocket protocol.

# PHẦN MỞ ĐẦU

1. **Vấn đề đồ án cần giải quyết**

Như đã nói trong lời nói đầu, tình trạng ô nhiễm không khí và tắc nghẽn giao thông ngày càng trở nên nghiêm trọng nên các phương tiện công cộng được sử dụng với tần suất cao. Đạp xe góp phần giảm ô nhiếm không khí và mang đến môi trường xanh, sạch đẹp cho cộng đồng. Phí di chuyển tiết kiệm, tốt cho sức khỏe và hỗ trợ di chuyển ở cự li ngắn. Với việc lựa chọn đề tài “Hệ thống cho thuê - chia sẻ xe đạp công cộng”, vấn đề chính mà em giải quyết là thuê xe nhanh gọn bằng việc quét mã QR và thánh toán online, lưu lại quãng đường di chuyển của xe và tìm đến các bãi đỗ xe gần nhất để trả xe. Cùng với đó hệ thống cho phép người quản lý theo dõi được các thông tin của hệ thống.

1. **Phương pháp hiện có**

Dịch vụ cho thuê xe đạp theo giờ ngày càng trở nên phổ biến và thu hút rất nhiều sự quan tâm. Khi thuê xe thì cần nộp tiền trước và nộp chứng mình thư hoặc giấy tờ các nhân có thể làm chứng. Luôn cần người túc trực để cho thuê xe khi khách đến và tiếp nhận lại xe khi khách kết thúc lần thuê xe.

1. **Mục đích và phạm vi đồ án**

Khi lựa chọn thực hiện Đồ Án Tốt Nghiệp bằng một đề tài về thuê xe đạp công cộng nhằm kết nối giao thông và lan tỏa hình thức đi xe đạp trên các tuyến phố văn minh, hiện đại. Em đã xác định mục đích chính là cho thuê xe một cách đơn giản và nhanh chóng chỉ cần một chiếc điện thoại di động thông minh.

1. **Tóm tắt cấu trúc đồ án**

Đồ án được thực hiện bằng việc sử dụng công cụ lập trình là Android Studio, ngôn ngữ lập trình Java – ngôn ngữ phổ biến thường được sử dụng lập trình ứng dụng.Sử dụng MySQL để quản lí dữ liệu và Google Map Api để xác định vị trí các bãi đỗ xe và đường đi của xe. Việc truyền tải thông tin giữa khoá và server được thực hiện bằng giao thức MQTT (Message Queuing Telemetry Transport), giao tiếp giữa server và mobile app được thực hiện bằng giao thức HTTP (HyperText Transfer Protocol) kết hợp giao thức WebSocket. Nội dung chính của đồ án sẽ được chúng em trình bày theo 3 phần chính tương ứng với 3 chương và một kết luận cuối cùng.

Chương 1 trình bày về hiện trạng ô nhiêm không khí và tắt nghẽn giao thông tại các thành phố lơn hiện này. Tiếp đó là giới thiệu các phương pháp hiện có để giải quyết bài bài toàn cho thuê xe đạp hiện nay. Sau đó là trình bày kỹ hơn về sơ đồ khối các bước thực hiện, cơ sở thực lý thuyết về các công nghệ cần sử dụng trong đề tài.

Chương 2 sẽ trình về lý thuyết về ngôn ngữ lập trình Java – ngôn ngữ được sử dụng chính và xuyên suốt trong cả đề tài. Sau đó lẽ trình bày về các bảo mật cơ bản trong Java web.

Chương 3 đưa ra trình bày về phân tích, thiết kế hệ thống. Sau đó đi vào chi tiếp chức năng, nhiệm vụ của từng khối trong hệ thống. Cuối cùng là kết quả thu được sau khi hoàn thiện hệ thống.

Phần kết luận sẽ tổng kết lại các kết quả đã đạt được và chưa giải quyết được trong đồ án, từ đó đưa ra đánh giá về mức độ hoàn thành công việc. Bên cạnh đó em cũng sẽ trình bày một chút về hạn chế của đồ án này và các khả năng phát triển của đề tài này trong tương lai.

# CƠ SỞ HÌNH THÀNH HỆ THỐNG

Tóm tắt nội dung Chương 1 được trình bày tại đây, dài khoảng từ 5 đến 10 dòng.

## Vấn đề ô nhiễm mỗi trường, tắc đường do xe cơ giới và các biện pháp hiện có

### Vấn đề ô nhiễm mỗi trường, tắc đường do xe cơ giới ở nước ta

Theo thống kê của Tổng Cục Thống Kê năm 2019, tổng dân số ở hai thành phố lớn của nước ta là Hà Nội và thành phố Hồ Chí Minh chiếm tới gần 18% dân số cả nước (17 triệu người trên tổng số 96 triệu người), dân số thành thị chiếm 34,4% tổng dân số cả nước. Do vậy có thể nói rằng dân cư nước ta đang ngày càng tập trung về các đô thị lớn – những nơi có điều kiện kinh tế phát triển nhất cả nước. Do vậy mà số lượng xe cơ giới trong các đô thị đã và đang gia tăng nhanh chóng trong khi cơ sở hạ tầng cụ thể thì cần nhiều thời gian để quy hoạch và xây dựng. Vấn

# LÝ THUYẾT VỀ JAVA

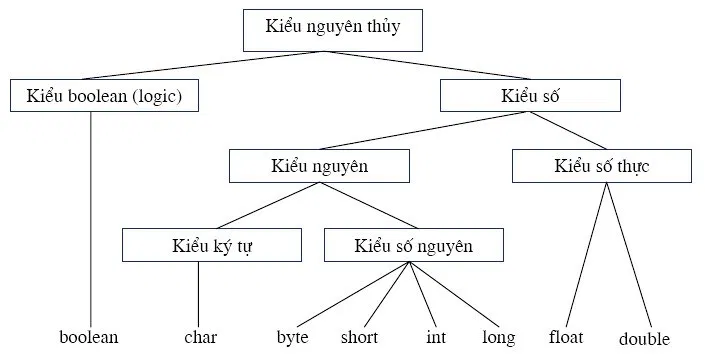
Java là một một ngôn ngữ lập trình hiện đại, bậc cao, hướng đối tượng, bảo mật và mạnh mẽ. Java được biết đến với tính năng viết 1 lần dùng ở mọi nơi vì nó có thể hoạc động ở bất cứ môi trường phần cứng hoặc phần mềm nhờ môi trường runtime riêng cho mình (JRE). Phiên bản mới nhất của java hiện đang là phiên bản java 17 nhưng phiên bản phổ biến nhất, mang lại đột phá nhất đang là java 8.

## Java Core

Java đã phát triển rất nhiều, có thể nói để học, hiểu hết được tất cả mọi kiến thức về Java gần như là không thể, ta chỉ nên học sâu về 1 số kiến thức phục vụ cho 1 công việc nhất định. Tuy nhiên để làm được điều đó thì đều cần phải kĩ các kiến thức cơ bản nhất của Java (Java core).

### Biến nguyên thuỷ (primitive) và tham chiếu (reference)

Biến nguyên thuỷ trong java bao gồm 8 kiểu dữ liệu:



Hình 2.1 Các kiểu dữ liệu nguyên thuỷ

Tất cả các biến còn lại đều là biến tham chiếu. Để hiểu rõ về sự khác nhau của 2 loại biến thì ta cùng tìm hiểu ví dụ sau:

Text

Description automatically generated

Hình 2.2 Ví dụ về biến nguyên thuỷ và biến tham chiếu

Và đây là kết quả:

primitive :1 3

reference: 3 3

Giải thích cho sự khác biệt này, vì kiểu dữ liệu nguyên thủy chứa dữ liệu ở trong chính ô nhớ của nó nên khi thay đổi b thì chỉ giá trị trong ô nhớ của b thay đổi mà giá trị trong ô nhớ của a không thay đổi. Còn đối với kiểu dữ liệu tham chiếu thì sau phép gán myNumber2 = myNumber1, thì cả myNumber1 và myNumber2 sẽ cùng tham chiếu tới 1 ô nhớ, điều này làm cho khi thay đổi giá trị myNumber1.value chính là làm thay đổi giá trị value ở tại cái ô nhớ mà cả myNumber1 và myNumber2 cùng tham chiếu tới.

Cần lưu ý là khi so sánh giá trị 2 biến trong java, nếu là 2 giá trị nguyên thuỷ thì sẽ so sánh về giá trị, nếu là 2 giá trị tham chiếu thì sẽ so sánh về địa chỉ ô nhớ.

### Auto boxing và auto unboxing trong lớp Wrapper

Lớp Wrapper trong java cung cấp cơ chế để chuyển đổi kiểu dữ liệu nguyên thủy thành kiểu đối tượng và từ đối tượng thành kiểu dữ liệu nguyên thủy. Từ java 5.0, tính năng và chuyển đổi kiểu dữ liệu nguyên thủy thành kiểu đối tượng (autoboxing) và từ đối tượng thành kiểu dữ liệu nguyên thủy (unboxing) một cách tự động đã được thêm vào. Trong java, có 8 lớp Wrapper tương ứng với 8 kiểu dữ liệu nguyên thuỷ đã đề cập ở trên. Wrapper mang lại những lợi ích sau:

* Wrapper Class sẽ chuyển đổi các kiểu dữ liệu nguyên thủy thành các đối tượng . Các đối tượng là cần thiết nếu chúng ta muốn sửa đổi các đối số được truyền vào phương thức (vì các kiểu nguyên thủy được truyền theo giá trị ).
* Các lớp trong gói java.util chỉ xử lý các đối tượng và do đó các lớp trình bao bọc  cũng giúp ích trong trường hợp này.
* Các cấu trúc dữ liệu trong khung công tác Collection như ArrayList và Vector  chỉ lưu trữ các đối tượng (kiểu tham chiếu) chứ không phải các  kiểu nguyên thủy.
* Đối tượng là cần thiết để hỗ trợ đồng bộ hóa trong đa luồng

### Lớp String

Lớp không chỉ dùng để lưu chữ 1 chuỗi mà còn cung cấp các phương thức giúp cho thao tác với 1 chuỗi trở nên rất dễ dàng. Cần lưu ý, String là một đối tượng không thể modify sau khi đã khởi tạo. Ta cùng đến với ví dụ sau:

Text

Description automatically generated

Hình 2.3 Ví dụ tính không thể modify của String

Kết quả thu được:

true

hongdatchy

hongdat

Giải thích kết quả thu được như sau: chúng ta đang khai báo biến s và s1 trực tiếp, mà không dùng từ khoá String, Java sẽ tự động cho 2 biến s và s1 trỏ đến cùng 1 ô nhớ do ta khai báo 2 chuỗi là như nhau, chính vì vậy nến khi so sánh (bản chất là so sánh 2 biến tham chiếu nên sẽ so sánh địa chỉ ô nhớ mà nó trỏ đến) thì sẽ được kết quả là true. Tuy nhiên khi ta thay đổi biến s thành 1 chuỗi khác thì biến s1 lại không hệ thay đổi theo. Từ đây ta có thể thấy được tính chất không thể modify của String, bản chất khi ta thay đổi 1 String thì java sẽ tạo 1 biến mới, và ô nhớ chứa chuỗi cũ sẽ vẫn ở đấy cho đến khi chương trình kết thúc thì mới được Java dọn dẹp.

Chính vì vậy nếu khi cần thay đổi 1 chuỗi và tránh lãng phí bộ nhớ thì nên cần nhắc sử dụng các lớp khác như StringBuilder, StringBuffer…Tuy nhiên các lớp này sẽ có tốc độ xử lý các phương thức với chuỗi khác không nhanh như lớp String. Ngoài ra Java còn cung cấp lớp StringTokenizer giúp việc phân tách 1 chuỗi trở nên dễ dàng hơn.

### Biểu thức chính quy (Regular expression)

Biểu thức chính quy là một chuỗi các ký tự tạo thành một mẫu tìm kiếm. Biểu thức chính quy có thể là một ký tự đơn hoặc một mẫu phức tạp hơn. Biểu thức chính qua là không thể thiếu đối với các hệ thống vần kiểm tra dữ liệu người dùng trước khi đưa vào hệ thống.

## Lập trình hướng đối tượng (oop) trong Java

Đối tượng (Object) là một thực thể có trạng thái và hành vi. Nó có thể mang tính vật lý hoặc logic. Lớp (Class) có thể xem lớp như một khuôn mẫu (template) của đối tượng. Trong đó bao gồm dữ liệu của đối tượng (fields hay properties) và các phương thức(methods) tác động lên thành phần dữ liệu đó gọi là các phương thức của lớp.

### Lớp Object

Mặc định lớp Object là lớp cha của tất cả các lớp trong java. Ta không cần khai báo nhưng Java tự hiểu tất cả các class đều kế thừa class Object.Lớp Object có sẵn 1 số các phương thức và tất cả các lớp trong Java đều được thừa hưởn các phương thức đó. Việc hiểu và ghi đề (Override) các phương thức này khi cần thiết là vô cùng quan trọng. Có thể kể đến 1 vài phương thức như

|  |  |
| --- | --- |
| HashCode | Trả về số hardcode cho đối tượng hiện tại. Mỗi đối tượng sẽ có 1 số hardcode riêng, nó có ứng dụng rất lớn trong cấu trúc dữ liêu. |
| ToString | Phương thức định nghĩa kiểu hiên thị dữ liệu khi in ra màn hình của lớp |
| Equal | Phương thức định nghĩa các so sánh 2 Object như thế nào thì được cao là băng nhau |
| CompareTo | Phương thức so sánh 2 Object, thường được dùng trong Comparator hoặc Comparator |

Bảng 2.1 Các phương thức quan trong của lớp Object

### Các tính chất của lập trình hướng đối tượng trong Java

* Tính đóng gói và bảo mật bên trong. Tức là các thuộc tính của các đối tượng được bảo vệ, không thể thay đổi từ bên ngoài 1 cách trực tiếp, việc này đảm bảo sự toàn vẹn, bảo mật của đối tượng trong Java. Tính đóng gói được thể hiện thông qua phạm vi truy cập (access modifier). Có 4 loại phạm vi truy cập trong Java. Bảng dưới đây mô tả khả năng truy cập của các Access Modifier trong java:

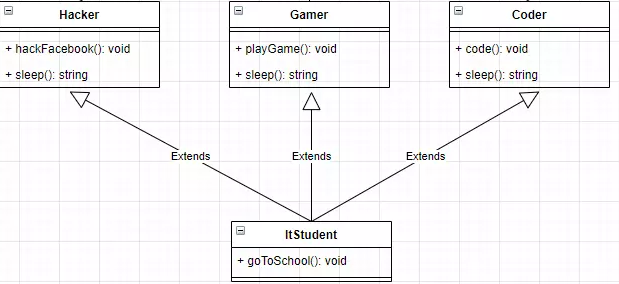
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Access Modifier** | **Trong lớp** | **Trong package** | **Ngoài package bởi lớp con** | **Ngoài package** |
| Private | Y | N | N | N |
| Default | Y | Y | N | N |
| Protected | Y | Y | Y | N |
| Public | Y | Y | Y | Y |

Bảng 2.2 Khả năng truy cập của các Access Modifier

* Tính kế thừa trong Java là khả năng cho phép ta xây dựng một lớp mới dựa trên các định nghĩa của một lớp đã có. Lớp đã có gọi là lớp Cha, lớp mới phát sinh gọi là lớp Con và đương nhiên kế thừa tất cả các thành phần của lớp Cha, có thể chia sẻ hay mở rộng các đặc tính sẵn có mà không phải tiến hành định nghĩa lại.
* Tính đa hình:
* Nạp chồng (Overloading): Đây là khả năng cho phép một lớp có nhiều thuộc tính, phương thức cùng tên nhưng với các tham số khác nhau về loại cũng như về số lượng. Khi được gọi, dựa vào tham số truyền vào, phương thức tương ứng sẽ được thực hiện
* Ghi đè (Overriding): là hai phương thức cùng tên, cùng tham số, cùng kiểu trả về nhưng lớp con viết lại và dùng theo cách của nó, và xuất hiện ở lớp cha và tiếp tục xuất hiện ở lớp con. Khi dùng override, lúc thực thi, nếu lớp Con không có phương thức riêng, phương thức của lớp Cha sẽ được gọi, ngược lại nếu có, phương thức của lớp Con được gọi
* Tính trừu tượng là một tiến trình ẩn các chi tiết trình triển khai và chỉ hiển thị tính năng tới người dùng. Tính trừu tượng cho phép bạn loại bỏ tính chất phức tạp của đối tượng bằng cách chỉ đưa ra các thuộc tính và phương thức cần thiết của đối tượng trong lập trình. Trong Java, chúng là sử dụng abstract class và abstract interface để có tính trừu tượng

### Đa kế thừa

Trước hết ta cùng tìm hiểu đa kế thừa là gì. Đơn giản là một class sẽ kế thừa từ nhiều hơn một class khác. Điều này là rất hưu ích trong các trường hợp lớp con cần kết thừa nhiều phương thức của lớp cha. Tuy nhiên lại có thể dẫn tới 1 vấn đề như trong ví dụ dưới đây.



Hình 2.4 Vấn đề khi dùng đa kế thừa

Class ItStudent kế thừa các lớp cha là các lớp: Hacker, Gamer, Coder. Các lớp cha có đều có phương thức sleep thì class con sẽ không biết kế thừa phương thức nào trong các lớp cha.

Vì lý do trên mà Java không hỗ trợ đa kết thừa đối với các class. Vậy làm thế nào để có thể đạt được đa kế thừa trong Java, chúng ta sẽ sử dụng interface. Một class không thể kế thừa từ nhiều class khác nhưng có thể implements nhiều interface khác nhau một lúc. Ở đây nếu Hacker, Gamer, Coder chỉ là các Interface, thì dù chings có chung làm sleep nhưng khi ItStudent implements chúng thì chỉ có một hàm được triển khai.

### Abstract class và Interface

Abstract class và interface đều được sử dụng để có được sự trừu tượng mà ở đó chúng ta có thể khai báo các phương thức trừu tượng. Nhưng có một vài sự khác nhau giữa abstract class và interface được liệt kê trong bảng sau:

|  |  |
| --- | --- |
| **Abstract class** | **Interface** |
| Abstract class có phương thức abstract (không có thân hàm) và phương thức non-abstract (có thân hàm) | Interface chỉ có phương thức abstract. Từ java 8, nó có thêm các phương thức default và static. |
| Abstract class không hỗ trợ đa kế thừa. | Interface có hỗ trợ đa kế thừa |
| Abstract class có các biến final, non-final, static and non-static | Interface chỉ có các biến static và final |
| Abstract class có thể cung cấp nội dung cài đặt cho phương thức của interface | Interface không thể cung cấp nội dung cài đặt cho phương thức của abstract class |
| Abstract class thường được dùng khi class chả và class con có quan hệ “is a” | Interface thường được dùng khi class chả và class con có quan hệ “has a” |
| Không thể thay đổi hành vi khi Run Time | Có thể thay đổi hành vi khi Runtime |
| Có Contructor và Decontructor | Không có Contructor và Decontructor |

Bảng 2.3 Sự khác nhau giữa Abstract class và Interface

### Generic và Collection

Thuật ngữ “Generics” được hiểu là tham số hóa kiểu dữ liệu. Việc tham số hóa kiểu dữ liệu giúp cho lập trình viên có thể dễ bắt lỗi các kiểu dữ liệu không hợp lệ, đồng thời giúp dễ dàng hơn cho việc tạo và sử dụng các class, interface, method với nhiều kiểu dữ liệu khác nhau. Có 2 kiểu Generics Class và Generics Static method. Generic hoàn toàn có thể được thay thế bằng cách dùng Object để khai báo kiểu dữ liệu đầu vào, tuy nhiên hiệu suất sẽ không bằng dùng Generic. Generic có ứng dụng khá nhiều trong cấu trúc dữ liệu, tiêu biểu là trong Collection.

Collection có thể coi là 1 framework của Java, nó bao gồm rất nhiều kiểu cấu trúc lưu trữ 1 tập dữ liệu. Với mỗi kiểu cấu trúc lưu trữ lại có 1 đặc tính riêng, nên cần phải tìm hiểu rõ yêu cầu bài toán để áp dụng cho chính xác. các yếu tố cần quan tâm:

* Tốc độ truy cập các phần tử
* Tốc độ thêm, sửa, xoá các phần tử
* Đặc tính khi lưu collection (thứ tự, độ ưu tiên…)

Graphical user interface

Description automatically generated

Hình 2.5 Collection Framework

### Stream Api và Lamda Expressions

Lamda Expression giúp cho việc viết code khi implement trực tiếp 1 interface trở nên nhanh gọn hơn. Ta cùng đến với 2 cách viết cho cùng 1 kết quả, 1 cách thông thường, 1 cách dùng Lamda

Text

Description automatically generated

Text

Description automatically generated

Hình 2.5 Ví dụ về Lamda

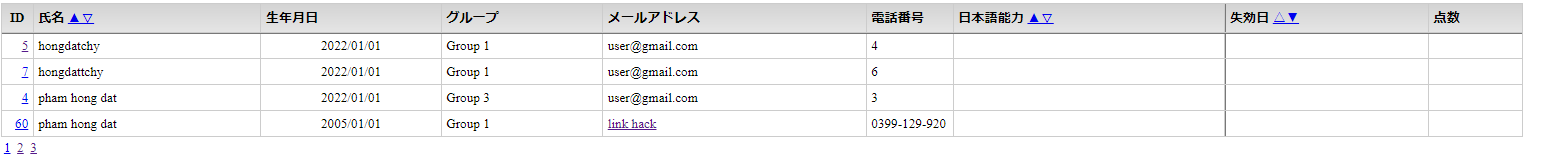
Stream API là một trong những feature chính của java 8 khi nó được giới thiệu, toàn bộ nằm trong package java.util.stream , gồm những API xử lý tuần tự các element cho collection. Stream API cung cấp các thao tác trên chuỗi như: filter, mapping…

## Bảo mật trong Java

Bảo mật là 1 việc không thể thiếu trong quá trình làm phần mềm thực tế, em đã được các anh chị dạy về các lỗi bảo mật và lập trình viên hay mắc phải có thể dẫn tới việc hệ thống bị hacker tấn công:

### Phòng chống được lỗi XSS

Đây là lỗi tấn công dựa vào giao diện, thường được dùng trong các màn hình có hiển thị dữ liệu nhập vào từ người dùng. Để hiểu rõ ta tìm hiểu qua ví dụ sau



Hình 2.6 Ví dụ về Lỗi Xss

Đây là giao diện 1 trang hiển thị danh sách người dùng, ở cột ngoài cùng bên phải là hiển thị email mà người dùng đăng kí, nhưng nếu người dùng nhập 1 chuỗi kí tự đặc biệt thì hoàn toàn có thể hiển thị 1 link như hàng thứ 4 trong hình trên. Nếu click đó độc hại và 1 người dùng khác click phải thì rất nguy hiểm.

Graphical user interface, application

Description automatically generated

Hình 2.7 Ví dụ sau khi phòng chống lỗi XSS

Để phòng chống thì ta cần mã hoá các kí tự đặc biệt của dữ liệu trước khi trả về cho trình duyệt. Sau khi xử lý thì ta sẽ đươc kết quả như trên, dữ liệu hiển thị đúng với dữ liệu nhập vào dưới dạng text.

### Phòng chống được lỗi SQL Injection

Đây là lỗi rất nguy hiểm, nếu không phòng chống tốt hacker hoàn toàn có thể lấy giữ liệu người dùng hay xoá bỏ toàn bộ database. Lỗi thường bị gây ra khi trang web cần có các mệnh đề truy vấn Where Like hoặc mệnh đề Order By, bản chất là các câu lệnh này đều lấy dữ liệu người dùng nhập vào và hoàn toàn có thể gây ra lỗi.

Graphical user interface

Description automatically generated

Hình 2.8 Ví dụ về lỗi SQL Injection

Trong ví dụ trên tại ô tìm kiếm nếu ta nhập 1 chuỗi như sau : **';drop table tbl\_user;** thì hoàn toàn bảng tbl\_user sẽ bị xoá vì chức năng tìm kiếm có sử dụng câu SQL có mệnh đề Where Like. Để phòng chống thì cần xử lý các kí tự đặc biệt trước khi thực hiện câu truy vấn. Mệnh đề Order By thường được dùng trong các trường hợp sắp xếp dữ liệu khi lấy ra theo tên 1 cột nào đó, để bảo mật thì trước khi thực hiện câu truy vấn ta phải kiếm tra dữ liệu nhập vào có phải là tên 1 cột nào đó trong Database không.

### Mã hoá dữ liệu nhậy cảm

Các dữ liệu nhậy cảm cần được mã hoá, ví dự như thông tin mật khẩu người dùng. Đây là hình ảnh mật khẩu sau khi đã mã hoá, tránh được việc lộ mật khẩu khi những người quản lý server có hành vi muốn chiếm đoạt tài khoản.

Text, letter

Description automatically generated

Hình 2.9 Mã hoá dữ liệu nhậy cảm

# THIẾT KẾ HỆ THỐNG

Sau khi đã giới thiệu khái quát về phương pháp được sử dụng trong đồ án, các cơ sở lý thuyết, các kỹ thuật đã áp dụng và cơ sở thực tiễn thì chúng ta sẽ đến với CHƯƠNG 3. Chương này sẽ trình bày chi tiết về cách nghiên thiết kế cho hệ thống và cụ thể cách thức hoạt động từng phần của hệ thống.

## Thiết kế sơ đồ khối

Sơ đồ khối hệ thống của em thể hiện như hình vẽ.

Diagram

Description automatically generated

Hình 3.1 Sơ đồ khối của hệ thống

### Khối IoT Messaging server và khối IoT Messaging client

MQTT(Message Queue Telemetry Transport) là một giao thức nhắn tin tiêu chuẩn OASIS cho Internet of Things (IoT). Nó được thiết kế như một phương tiện truyền tải tin nhắn publish / subscriber cực kỳ nhẹ, lý tưởng để kết nối các thiết bị từ xa và cần tiêu thụ ít năng lượng. Vậy nên MQTT là 1 giao thức rất thích hợp cho hệ thống để liên lạc giữa backend với device.

MQTT gồm 2 thành phần chính:

* **MQTT Broker** hay máy chủ mô giới được coi như trung tâm, nó là điểm giao của tất cả các kết nối đến từ Client (Publisher/Subscriber). Nhiệm vụ chính của Broker là nhận thông điệp (message) từ Publisher, xếp vào hàng đợi rồi chuyển đến một địa điểm cụ thể. Nhiệm vụ phụ của Broker là nó có thể đảm nhận thêm một vài tính năng liên quan tới quá trình truyền thông như: bảo mật message, lưu trữ message, log…MQTT Broker được cung cấp dưới dạng mã nguồn mở hoặc các phiên bản thương mại giúp người dùng có thể tự cài đặt và tạo broker riêng. Ngoài ra các bạn cũng có thể sử dụng Broker trên điện toán đám mây với các nền tảng IOT như hive broker, amazone…
* **MQTT client** là các thiết bị/ứng dụng Client kết nối đến Broker để thực hiện truyền nhận dữ liệu .Client thì được chia thành hai nhóm là Publisher và Subscriber. Một Client có thể có 1 trong 2 nhiệm vụ hoặc cả 2. Publisher là thiết bị gửi bản tin lên broker. Subscriber là người nhận bản tin mỗi khi có bản tin mới gửi lên Broker.

Trong MQTT có các khái niệm như sau:

* **Message** còn được gọi là “message payload”, có định dạng mặc định là plain-text (chữ viết người đọc được), tuy nhiên người sử dụng có thể cấu hình thành các định dạng khác.
* **Topic** có thể coi như một “đường truyền” logic giữa 2 điểm là publisher và subscriber. Về cơ bản, khi message được publish vào một topic thì tất cả những subscriber của topic đó sẽ nhận được message này. MQTT còn hỗ trợ toán tử “#” giúp cho 1 client nhận nhiều hơn 1 topic với chỉ 1 lần subscribe.

### Khối Backend

Backend có thể coi như trung tâm điều khiển cho cả hệt thống, và cần đạp ứng được các yêu cầu sau:

* Cung cấp các Rest api cho user app và admin web hoạt động
* Cung cấp khả năng thông báo tin nhắn cho ser app qua websocket
* Tích hợp được MQTT client vào backend

Spring boot là là một dự án phát triển bởi JAV (ngôn ngữ java) trong hệ sinh thái Spring framework. Nó giúp cho các lập trình viên chúng ta đơn giản hóa quá trình lập trình một ứng dụng với Spring, chỉ tập trung vào việc phát triển business cho ứng dụng. Spring boot hoàn toàn là sự lựa chọn tốt để đáp ứng các yêu cầu của backend ở phía trên.

Có thể nói rằng Spring Framework ra đời là nhờ sự thành công của EJB (Enterprise JavaBeans). EJB là một thành phần quan trọng của nền tảng Java EE. Tuy nhiên EJB là một framework lớn với độ phức tạp cao, các nhà phát triển với EJB đã nhận ra nó không hoạt động như những gì nó đã đề ra.

A picture containing text, tableware, clipart, plate

Description automatically generated

Hình 3.2: Enterprise JavaBeans

Năm 2002, Spring Framework phát hành phiên bản đầu tiên bới Rod Johnson giúp cho việc xây dựng các ứng dụng doanh nghiệp trở nên dễ dàng hơn bao giờ hết. Sau một khoảng thời gian phát triển, Spring nhanh chóng chiếm được cảm tình của các lập trình viên và trở thành sự lựa chọn hoàn hảo cho các ứng dụng Java web hiện nay.

Để hiểu được ý tưởng chính của Spring Boot ta cần tìm hiểu về Singleton design pattern. Singleton Pattern nói rằng chỉ cho phép khai báo 1 class mà chỉ với 1 trường hợp cụ thể (instance) trong hệ thống khi runtime. Có 2 cách thiết kế Class để đáp ứng được điều này:

* Early Instantiation: Tạo Instance ở thời điểm load time
* Lazy Instantiation: Tạo Instance ở thời điểm yêu cầu

Ta cùng đến với ví dụ về Singleton Pattern theo cách Early Instantiation như sau:

Text

Description automatically generated

Hình 3.3 Ví dụ về singleton Pattern

Kết quả thu được như sau:

com.example.demo.A@7343922c

com.example.demo.A@7343922c

Có thể thấy trong ví dụ trên, mỗi khi khai báo class A thì chỉ có 1 instance duy nhất được tạo. Trong Spring boot ta hoàn toàn có thể làm được điều này. Interface ApplicationContext cung cấp khả năng giữ, quản lý, phân phối các intance của các Object cần thiết kế theo kiểu Singlegon pattern. Khi này các Class này được gọi là các Beans. Có 4 chú thích (Anotation) báo cho Spring Boot biết 1 class được thiết kế theo kiểu Singleton pattern là dùng: @Component, @Service, @Reposiory, @Controller.

Text

Description automatically generated

Hình 3.4 Ví dụ về Anotiton Component

Sơ đồ khối của Backend sẽ như sau:

Diagram

Description automatically generated

Hình 3.5 Sơ đồ khối của Backend

* Lớp repository: thao tác trực tiếp với cơ sở dữ liệu, thực hiện các thao tác thêm, sửa, xoá, tìm kiếm
* Lớp service: thực hiện các xử lý logic, là kết nối giữa tầng controller và repository
* Lớp controller: sử dụng tầng service để viết các api cho frontend sử dụng
* Lớp sercurity: phân quyền sử dụng api, từ đó phân quyền sử dụng hệ thống

### User App

Android là hệ điều hành có mã nguồn mở dựa trên nền tảng Linux do chính Google phát hành được thiết kế dành cho các nhà phát triển thiết bị, các nhà mạng, lập trình viên có thể tiếp cận nhanh chóng, điều chỉnh tự do trong mã nguồn mở đó. Vậy nên em quyết định làm User app cho người dùng trên hệ điều hành android.

User App cần có các tính năng như sau:

* Đăng kí tài khoản mới
* Kích hoạt tài khoản bằng mã Code được gửi trên Email
* Đăng nhập, đăng xuất tài khoản
* Khi mở ứng dụng thì tự động vào trang chủ nếu đã đăng nhập từ trước đó
* Cho phép tìm kiếm và chỉ đường đến các bài đỗ xe
* Quét mã QR để hiển thị thông tin xe sau đó thực hiện thuê xe
* Xem lại lịch sử các chuyến đi
* Nếu đang trong quá trình thuê xe mà đóng ứng dụng vào lại thì phải quay về trang thuê xe và giữ nguyên được trạng thái (đang tạm khoá xe hay mở khoá)
* Nếu quét mã QR và thuê xe đang được thuê thì sẽ báo không thành công

### Admin web

Admin web sẽ cung cấp khả năng quản lý các thông tin trong hệ thống như thông tin user, lịch sự các contract, thông tin các station… Em quyết định chọn react js để làm frontend cho web này vì framework này có những ưu điểm sau:

* React cho phép các nhà phát triển sử dụng các phần riêng lẻ của ứng dụng của họ ở cả phía máy khách và phía máy chủ, điều này cuối cùng giúp tăng tốc độ của quá trình phát triển.
* So với các frontend framework khác, mã React dễ bảo trì hơn và linh hoạt hơn do cấu trúc mô-đun của nó. Do đó, sự linh hoạt này giúp tiết kiệm rất nhiều thời gian và chi phí cho doanh nghiệp
* React JS được thiết kế để cung cấp hiệu suất cao. Cốt lõi của khung cung cấp chương trình DOM ảo và kết xuất phía máy chủ, giúp các ứng dụng phức tạp chạy cực nhanh
* Ý tưởng chính của React js là cho phép chia UI thành các phần độc lập, có thể tái sử dụng và bạn có thể sử dụng chúng 1 cách độc lập, mỗi phần đó được gọi là 1 Component

Admin web cần có các tính năng như sau:

* Đăng nhập, đăng nhập
* Quản lý user
* Quản lý các station
* Quản lý lịch sử các chuyến đi

### Khối Google Map Platform

Ứng dụng Google Map là một dịch vụ bản đồ trực tuyến trên web và app miễn phí do Google phát hành và quản lý, ứng dụng hỗ trợ nhiều dịch vụ, tính năng cho người dùng nhưng phổ biến nhất là dò đường, chỉ đường, tìm vị trí; hiển thị những tuyến đường tối ưu cho từng phương tiện tham gia giao thông, ngoài ra còn hướng dẫn cách bắt xe và chuyển tuyến xe dành cho người tham gia các phương tiện lưu thông công cộng (xe bus), hiển thị những địa điểm xung quanh vị trí người dùng hoặc vị trí chỉ định như ATM, trạm xăng, bệnh viện… Theo một số thông báo từ Google, Việt Nam nằm trong danh sách những vùng lãnh thổ bị cấm sử dụng Map API từ họ. Chính vì vậy để sử dụng được những dịch vụ Maps API cung cấp, người dùng cần sử dụng Billing Account mới, bank account này cần nằm ngoài danh sách các quốc gia bị chặn thì mới có thể sử dụng. Chính vì vậy User App cần kết nốt với Google Map Platform để làm được điều đó.

### Khối Device

Khối Device sẽ có trách nhiệm mô phỏng các thiết bị khoá gắn trên xe thật. Khối Device gồm 2 phần chính:

* Website có chức năng mô phỏng các thiết bị thật
* MQTT client cho phép trao đổi thông tin với Server qua giao thức MQTT, giao tiếp với với Website qua giao Rest Api và Websocket

### Khối Database

MySQL là ứng dụng cơ sở dữ liệu mã nguồn mở phổ biến nhất hiện nay (theo www. mysql. com) và được sử dụng phối hợp với Java. Trước khi làm việc với MySQL cần xác định các nhu cầu cho ứng dụng.

MySQL là cơ sở dữ có trình giao diện trên Windows hay Linux, cho phép người sử dụng có thể thao tác các hành động liên quan đến nó. Việc tìm hiểu từng công nghệ trước khi bắt tay vào việc viết mã kịch bản Java, việc tích hợp hai công nghệ java và MySQL là một công việc cần thiết và rất quan trọng

Mục đích sử dụng cơ sở dữ liệu bao gồm các chức năng như: lưu trữ (storage), truy cập (accessibility), tổ chức (organization) và xử lí (manipulation)

* Lưu trữ: Lưu trữ trên đĩa và có thể chuyển đổi dữ liệu từ cơ sở dữ liệu này sang cơ sở dữ liệu khác, nếu sử dụng cho quy mô nhỏ, có thể chọn cơ sở dữ liệu nhỏ như:Microsoft Exel, Microsoft Access, MySQL, Microsoft Visual FoxPro,... Nếu ứng dụng có quy mô lớn, bạn có thể chọn cơ sở dữ liệu có quy mô lớn như :Oracle, SQL Server, MySQL...
* Truy cập:Truy cập dữ liệu phụ thuộc vào mục đích và yêu cầu của người sử dụng, ở mức độ mang tính cục bộ, truy cập cơ sỏ dữ liệu ngay trong cơ sở dữ liệu với nhau, nhằm trao đổi hay xử lí dữ liệu ngay bên trong chính nó, nhưng do mục đích và yêu cầu người dùng vượt ra ngoài cơ sở dữ liệu, nên bạn cần có các phương thức truy cập dữ liệu giữa các cơ sở dữ liệu với nhau như: Microsoft Access với SQL Server, hay SQL Server và cơ sở dữ liệu Oracle....
* Tổ chức:Tổ chức cơ sở dữ liệu phụ thuộc vào mo hình cơ sở dữ liệu, phân tích và thiết kế cơ sở dữ liệu tức là tổ chức cơ sở dữ liệu phụ thuộc vào đặc điểm riêng của từng ứng dụng. Tuy nhiên khi tổ chức cơ sở dữ liệu cần phải tuân theo một số tiêu chuẩn của hệ thống cơ sở dữ liệu nnhằm tăng tính tối ưu khi truy cập và xử lí
* Xử lí:Tùy vào nhu cầu tính toán và truy vấn cơ sở dữ liệu với các mục đích khác nhau, cần phải sử dụng các phát biểu truy vấn cùng các phép toán, phát biểu của cơ sở dữ liệu để xuất ra kết quả như yêu cầu. Để thao tác hay xử lí dữ liệu bên trong chính cơ sở dữ liệu ta sử dụng các ngôn ngữ lập trình như:PHP, C++, Java, Visual Basic,...

**Database của hệ thống**

Diagram

Description automatically generated

**Bảng user**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| email | VARCHAR(100) | ⚫ |  | * email người dùng * dùng để đăng nhập * không trùng lặp |
| id\_number | VARCHAR(20) |  |  | * số chứng minh thư hoặc căn cước công dân |
| password | VARCHAR(64) | ⚫ |  | * mật khẩu người dùng được mã hoá bằng hàm băm SHA-256 |
| phone | VARCHAR(20) | ⚫ |  | * số điện thoại |
| firstname | VARCHAR(45) | ⚫ |  | * tên người dùng |
| lastname | VARCHAR(45) | ⚫ |  | * họ và tên đệm |
| credit\_card | VARCHAR(45) |  |  | * mã tài khoản ngân hàng |
| birthday | DATE | ⚫ |  | * sinh nhật |
| city\_id | INT | ⚫ |  | * mã tỉnh/thành phố * khoá ngoại tham chiếu đến bảng city |
| district\_id | INT | ⚫ |  | * mã quận/huyện * khoá ngoại tham chiếu đến bảng district |
| ward\_id | INT | ⚫ |  | * mã phường/xã * khoá ngoại tham chiếu đến bảng ward |
| last\_time\_access | DATETIME |  |  | * thời gian cuối cùng truy cập vào tài khoản |
| image | VARCHAR(255) |  |  | * link đường dẫn chưa file ảnh chân dung của người dùng |
| gender | VARCHAR(4) | ⚫ |  | * giới tính người dùng |
| reset\_code | VARCHAR(20) |  |  | * mã code cần dùng khi người dùng reset lại mật khẩu |
| active\_code | VARCHAR(20) |  |  | * mã code dùng khi active tài khoản |
| is\_active | BIT(1) | ⚫ |  | * trạng thái người dùng * nếu là 0 thì là chưa active, 1 là đã active |

**Bảng City**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ |  | * khoá chính |
| name | VARCHAR(45) | ⚫ |  | * tên thành tỉnh/thành phố |

**Bảng District**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ |  | * khoá chính |
| name | VARCHAR(45) | ⚫ |  | * tên quận/huyện |
| city\_id | INT | ⚫ |  | * mã city * khoá ngoại tham chiếu đến bảng city |

**Bảng Ward**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ |  | * khoá chính |
| name | VARCHAR(45) | ⚫ |  | * mã district * khoá ngoại tham chiếu đến bảng district |

**Bảng Admin**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| name | VARCHAR(100) | ⚫ |  | * email admin * dùng tài khoản đăng nhập * không trùng lặp |
| password | VARCHAR(64) | ⚫ |  | * mật khẩu người dùng được mã hoá bằng hàm băm SHA-256 |

**Bảng Black list**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| token | VARCHAR(64) | ⚫ |  | * mã token sau khi đã logout |

**Bảng Station**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| name | VARCHAR(45) | ⚫ |  | * tên station |
| slot\_quantity | INT | ⚫ |  | * số slot để xe tối đa |
| current\_number\_car | INT | ⚫ |  | * số xe hiện tại trong station |
| location | VARCHAR(45) | ⚫ |  | * đại chỉ của station |
| longitude | DOUBLE | ⚫ |  | * kinh độ của station |
| latitude | DOUBLE | ⚫ |  | * vĩ độ của station |

**Bảng Bike**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| station\_id | INT | ⚫ |  | * mã station * khoá ngoại tham chiếu đến bảng station |
| frame\_number | VARCHAR(45) | ⚫ |  | * số khung của xe |
| product\_year | VARCHAR(45) |  |  | * năm sản xuất của xe |

**Bảng Device**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| bike\_id | INT | ⚫ |  | * mã bike * khoá ngoại tham chiếu đến bảng bike |
| status\_lock | BIT(1) | ⚫ |  | * trạng thái khoá * 0 là đang khoá, 1 là đang mở khoá |
| longitude | DOUBLE |  |  | * kinh độ của bike |
| latitude | DOUBLE |  |  | * vĩ độ của bike |
| battery | INT |  |  | * phần trăm pin còn lại của device |

**Bảng Contract Bike**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| bike\_id | INT | ⚫ |  | * mã bike * khoá ngoại tham chiếu đến bảng bike |
| user\_id | INT | ⚫ |  | * mã user * khoá ngoại tham chiếu đến bảng bike |
| payment\_method | VARCHAR(45) | ⚫ |  | * tên phương thức thanh toán |
| start\_time | DATETIME | ⚫ |  | * thời gian bắt đầu của chuyến đi |
| end\_time | DATETIME |  |  | * thời gian kết thúc của chuyến đi |

**Bảng Path**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên column** | **Kiểu dữ liệu** | **Not null** | **Tự tăng** | **Mô tả** |
| id | INT | ⚫ | ⚫ | * khoá chính |
| contract\_id | INT | ⚫ |  | * mã contract * khoá ngoại tham chiếu đến bảng contract |
| distance | DOUBLE |  |  | * khoảng các mà xe đã đi được trong 1 lần thuê |
| routes | LONGTEXT | ⚫ |  | * các toạ độ mà xe đã đi trong 1 lần thuê |

## Sơ đồ sequence

### Chức năng thuê xe

Diagram

Description automatically generated

### Chức năng kết thúc chuyến đi

Diagram

Description automatically generated