BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG THÀNH PHỐ HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

---------------------------



**QUẢN LÝ KHÁCH HÀNG THAM GIA BẢO HIỂM NHÂN THỌ**

**SINH VIÊN THỰC HIỆN**

2001216091- Lê Nhật Quyên- Lớp 12DHTH02

2001215689 - Nguyễn Ngọc Minh Đan – Lớp 12DHTH07  
 2001216245 - Huỳnh Thanh Triệu - Lớp 12DHTH01  
 2001216252 - Trần Thị Trúc-12DHTH11

**GVHD: Th.S Nguyễn Văn Lễ**

**BÁO CÁO TIỂU LUẬN**

Môn học: Cơ sở dữ liệu NoSQL

TP. HỒ CHÍ MINH, tháng 09 năm 2023

**MỤC LỤC**

[Chương 1: GIỚI THIỆU 14](#_Toc179402743)

[1.1. Giới thiệu tổng quan về đề tài 14](#_Toc179402744)

[1.2. Lý do chọn đề tài. 14](#_Toc179402745)

[Chương 2: Xây dựng database 15](#_Toc179402746)

[2.1. Giới thiệu công cụ 15](#_Toc179402747)

[2.1.1. MongoDB 15](#_Toc179402748)

[2.1.2. Thiết kê mô hình cấu trúc dữ liệu cho đề tài 16](#_Toc179402749)

[2.1.2.1. Collection Employee 16](#_Toc179402750)

[2.1.2.2. Collection Customer 18](#_Toc179402751)

[2.1.2.3. Collection Agent 22](#_Toc179402752)

[2.1.3. Diễn giải chi tiết các đối tượng trong mô hình ví dụ minh hoạ 23](#_Toc179402753)

[2.1.3.1. Nhân viên (Employee) 23](#_Toc179402754)

[2.1.3.2. Khách Hàng 24](#_Toc179402755)

[2.1.3.3. Đại Lý 25](#_Toc179402756)

[2.1.4. Tạo cơ sở dữ liệu và các đồi tượng trong CSD 25](#_Toc179402757)

[2.1.4.1. Tạo cơ sở dữ liệu 25](#_Toc179402758)

[2.1.4.2. Tạo đối tượng trong collection và nhập cơ sở dữ liệu 29](#_Toc179402759)

[2.1.4.2.1. Thực hiện bằng công cụ 29](#_Toc179402760)

[2.1.4.2.2. Thực hiện bằng câu truy vấn 30](#_Toc179402761)

[2.1.5. Truy vấn dữ liệu 31](#_Toc179402762)

[2.1.5.1. Truy vấn Yêu Cầu Bồi Thường 31](#_Toc179402763)

[2.1.5.2. Truy vấn Hộp đồng 31](#_Toc179402764)

[2.1.5.3. Truy vấn Khách Hàng 31](#_Toc179402765)

[34](#_Toc179402766)

[2.1.5.4. Truy vấn trường thanh toán 35](#_Toc179402767)

[Chương 3: CÀI ĐẶT CHƯƠNG TRÌNH 43](#_Toc179402768)

[3.1. Giới thiệu chung 43](#_Toc179402769)

[3.1.1. Môi trường phát triển 43](#_Toc179402770)

[3.1.2. Hệ điều hành 43](#_Toc179402771)

[3.1.3. Ngôn ngữ lập trình 44](#_Toc179402772)

[3.1.4. Thư viện và công cụ bổ sung 45](#_Toc179402773)

[3.2. Mô tả giao diện 45](#_Toc179402774)

[3.2.1. Giao diện đăng nhập 45](#_Toc179402775)

[3.2.2. Giao diện form chính 46](#_Toc179402776)

[3.2.3. Giao diện khách hàng 47](#_Toc179402777)

[3.2.4. Giao diện đại lý 49](#_Toc179402778)

[3.2.5. Giao diện hợp đồng 50](#_Toc179402779)

[3.2.6. Giao diện yêu cầu bồi thường 51](#_Toc179402780)

[3.2.7. Giao diện thanh toán 53](#_Toc179402781)

[3.2.8. Giao diện giao dịch 54](#_Toc179402782)

[*Chương 4:* KẾT LUẬN 56](#_Toc179402783)

[4.1. Kết quả đạt được 56](#_Toc179402784)

[4.2. Những hạn chế , thiếu sót ,khó khăn 57](#_Toc179402785)

[4.3. Hướng phát triển của đề tài. 58](#_Toc179402786)

Mục lục hình ảnh :

[Hình 2. 1Cấu trúc collection nhân viên 23](#_Toc179402796)

[Hình 2. 2Cấu trúc collection khách hàng 24](#_Toc179402797)

[Hình 2. 3 Cấu trúc collection đại lý 25](#_Toc179402798)

[Hình 2. 4 Tạo mới database bằng công cụ 26](#_Toc179402799)

[Hình 2. 5 Tạo mới collection 27](#_Toc179402800)

[Hình 2. 6 Nhập tên collection tạo mới 27](#_Toc179402801)

[Hình 2. 7 Collection được tạo thành công 28](#_Toc179402802)

[Hình 2. 8 Sử dụng database bằng lệnh 28](#_Toc179402803)

[Hình 2. 9 Tạo mới collection bằng lệnh 29](#_Toc179402804)

[Hình 2. 10 Hiện thị collection đã tạo 29](#_Toc179402805)

[Hình 2. 11 Chèn dữ liêu bằng công cụ 30](#_Toc179402806)

[Hình 2. 12 Chèn dữ liệu bằng câu lệnh 30](#_Toc179402807)

[Hình 2. 13 Truy vấn thanh toán định kỳ cho một hợp đồng 35](#_Toc179402808)

[Hình 2. 14 Lọc danh sách giao dịch là loại yêu cầu bồi thường 38](#_Toc179402809)

[Hình 2. 15 Thông kế số lương thanh toán 41](#_Toc179402810)

[Hình 3. 1 Tìm kiếm khách hàng nữ trong độ tuổi 20 đến 25 31](#_Toc179402787)

[Hình 3. 2Truy vấn khách hàng có chính sách bảo hiểm 32](#_Toc179402788)

[Hình 3. 33. Truy vấn đại lý bảo hiểm có nhiều khách hàng nhất 34](#_Toc179402789)

[Hình 3. 4Giao diện Đăng Nhập 45](#_Toc179402790)

[Hình 3. 5Giao diện chính 46](#_Toc179402791)

[Hình 3. 6 Giao diện đại lý 49](#_Toc179402792)

[Hình 3. 7 Giao diện Yêu Cầu Bồi Thường 51](#_Toc179402793)

[Hình 3. 8 Giao diện thanh toán 53](#_Toc179402794)

[Hình 3. 9 Giao diện quản lý giao dịch 55](#_Toc179402795)

**Bảng phân công**

|  |  |  |
| --- | --- | --- |
| Tên Thành Viên | Công Việc | Hoàn Thành |
| Nguyễn Ngọc Minh Đan | * Word: * Giới thiệu tổng quan * Tạo cấu trúc cơ sở dữ liệu * Viết cây truy vấn yêu cầu hợp đồng * Mô tả giao diện hợp đồng * Code: * Quản Lý Yêu Cầu Bồi Thường | 100% |
| Trần Thị Trúc | * Word: * Tạo cấu trúc cơ sở dữ liệu * Viết cây truy vấn khách hàng/đại lý * Nhập dữ liệu cơ sở dữ liệu * Mô tả giao diện khách hàng,đại lý * Code: * Quản Lý khách hàng * Quản lý đại lý | 100% |
| Huỳnh Thanh Triệu | * Word: * Tạo cấu trúc cơ sở dữ liệu * Viết cây truy vấn khách hàng/đại lý * Mô tả giao diện hợp đồng * Kết luận * Code: * Quản Lý hợp đồng | 100% |
| Lê Nhật Quyên | * Word: * Tạo cấu trúc cơ sở dữ liệu * Nhập dữ liệu cở sở dữ liệu * Viết cây truy vấn thanh toán/giao dịch * Mô tả giao diện thanh toán / giao dịch * Code: * Thanh toán hợp đồng / yêu cầu bồi thường * Quản Lý giao dịch | 100% |

1. GIỚI THIỆU
   1. Giới thiệu tổng quan về đề tài

Đề tài **“Quản lý khách hàng tham gia bảo hiểm nhân thọ”** tập trung vào việc xây dựng một hệ thống hỗ trợ quản lý thông tin khách hàng, theo dõi quá trình tham gia các hợp đồng bảo hiểm, thanh toán phí, và xử lý các yêu cầu bồi thường. Hệ thống này không chỉ giúp các công ty bảo hiểm dễ dàng quản lý dữ liệu khách hàng một cách toàn diện mà còn cung cấp khả năng tự động hóa các quy trình quan trọng như nhắc nhở thanh toán, tính toán lợi ích và chi phí, cũng như lưu trữ hồ sơ chi tiết về yêu cầu bồi thường.

Cụ thể, hệ thống quản lý khách hàng tham gia bảo hiểm nhân thọ sẽ bao gồm các chức năng chính như:

* Quản lý thông tin cá nhân khách hàng và hợp đồng bảo hiểm.
* Theo dõi trạng thái thanh toán của hợp đồng, các đợt đóng phí định kỳ.
* Xử lý và theo dõi yêu cầu bồi thường từ khách hàng.
* Cập nhật tình trạng hợp đồng và tự động thông báo cho khách hàng khi cần thiết.
* Báo cáo, thống kê về số lượng hợp đồng, yêu cầu bồi thường, và doanh thu.

Việc số hóa và tự động hóa những quy trình này giúp nâng cao hiệu quả quản lý, giảm thiểu rủi ro sai sót và tăng cường sự minh bạch, rõ ràng trong các giao dịch giữa khách hàng và công ty bảo hiểm.

* 1. Lý do chọn đề tài.

Lý do chọn đề tài **"Quản lý khách hàng tham gia bảo hiểm nhân thọ"** xuất phát từ thực tế rằng bảo hiểm nhân thọ đang ngày càng trở thành một phần quan trọng trong cuộc sống của con người. Với sự phát triển của xã hội và nền kinh tế, nhu cầu bảo vệ tài sản và sức khỏe gia tăng, dẫn đến sự bùng nổ của thị trường bảo hiểm nhân thọ. Tuy nhiên, cùng với sự phát triển này là những thách thức trong việc quản lý lượng lớn khách hàng và thông tin phức tạp liên quan đến các hợp đồng bảo hiểm, yêu cầu bồi thường, và các đợt thanh toán định kỳ.

Một số lý do cụ thể bao gồm:

1. **Sự phức tạp trong quản lý thông tin**: Với số lượng khách hàng và hợp đồng bảo hiểm lớn, việc quản lý thủ công sẽ gây ra sai sót, chậm trễ và khó khăn trong việc lưu trữ, tra cứu thông tin.
2. **Nhu cầu số hóa và tự động hóa**: Trong thời đại công nghệ số, các doanh nghiệp bảo hiểm cần ứng dụng công nghệ để nâng cao hiệu quả làm việc, giảm thiểu chi phí vận hành, và tăng cường sự hài lòng của khách hàng.
3. **Tăng cường trải nghiệm khách hàng**: Hệ thống quản lý giúp khách hàng dễ dàng truy cập thông tin, nhận được thông báo nhắc nhở thanh toán hoặc theo dõi yêu cầu bồi thường, từ đó tạo sự tin tưởng và minh bạch trong quá trình tham gia bảo hiểm.
4. **Phân tích dữ liệu và ra quyết định**: Một hệ thống quản lý hiệu quả không chỉ giúp lưu trữ thông tin mà còn cung cấp các công cụ phân tích, báo cáo giúp doanh nghiệp đưa ra các quyết định kinh doanh dựa trên dữ liệu khách quan.

Việc xây dựng một hệ thống quản lý khách hàng tham gia bảo hiểm nhân thọ không chỉ là nhu cầu cấp bách của ngành bảo hiểm, mà còn là một đóng góp thiết thực giúp nâng cao trải nghiệm người dùng, đồng thời tối ưu hóa các quy trình nội bộ của doanh nghiệp.

1. Xây dựng database
   1. Giới thiệu công cụ
      1. MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL phổ biến, được thiết kế để lưu trữ và truy vấn các tập dữ liệu lớn và có cấu trúc linh hoạt. Thay vì lưu trữ dữ liệu theo dạng bảng và hàng như trong các hệ quản trị cơ sở dữ liệu quan hệ, MongoDB lưu trữ dữ liệu dưới dạng tài liệu (document) với định dạng JSON hoặc BSON. Điều này giúp việc xử lý và lưu trữ các loại dữ liệu không đồng nhất dễ dàng và nhanh chóng hơn, thích hợp cho các ứng dụng hiện đại như web, di động, và IoT.

Một số tính năng nổi bật của MongoDB:

* Khả năng mở rộng theo chiều ngang (horizontal scalability).
* Hỗ trợ truy vấn linh hoạt và mạnh mẽ.
* Lưu trữ dữ liệu phi cấu trúc hoặc có cấu trúc linh hoạt.
* Tích hợp tốt với các ứng dụng hiện đại và dễ dàng sử dụng với các ngôn ngữ lập trình khác nhau.
  + 1. Thiết kê mô hình cấu trúc dữ liệu cho đề tài

Hệ thống quản lý khách hàng bảo hiểm nhân thọ bao gồm ba collection chính: **Employee**, **Customer**, và **Agent**. **Collection Employee** lưu trữ thông tin về các nhân viên làm việc trong tổ chức bảo hiểm, với mã nhân viên duy nhất và các thông tin chi tiết như họ tên, ngày sinh, giới tính, chức vụ, và phòng ban. Thông tin liên lạc của nhân viên, bao gồm số điện thoại, email, và địa chỉ, được ghi lại để dễ dàng liên lạc và hỗ trợ khi cần thiết. **Collection Customer** chứa thông tin về khách hàng tham gia các hợp đồng bảo hiểm nhân thọ, với mã khách hàng và thông tin cá nhân như họ tên, ngày sinh, giới tính, và thông tin liên lạc. Hệ thống cũng ghi nhận các hợp đồng bảo hiểm mà khách hàng đã tham gia, các yêu cầu bồi thường đã gửi, và lịch sử thanh toán, từ đó hỗ trợ quản lý và phục vụ khách hàng hiệu quả. Cuối cùng, **Collection Agent** lưu trữ thông tin về các đại lý bảo hiểm, là cầu nối giữa công ty bảo hiểm và khách hàng, với mã định danh riêng, tên, và thông tin liên lạc, tạo điều kiện thuận lợi cho khách hàng tiếp cận và nhận tư vấn về các sản phẩm bảo hiểm.

* + - 1. Collection Employee

**Mô tả**: Collection này lưu trữ thông tin của nhân viên trong tổ chức bảo hiểm nhân thọ. Các nhân viên này thường làm việc trong các phòng ban khác nhau như bán hàng, hỗ trợ khách hàng, và quản lý hợp đồng.

* + - Cấu Trúc :

Collection : employee

{

employee\_id : Mã Nhân Viên,

full\_name: Họ Tên Nhân Viên,

dob: Ngày Sinh,

gender: Giới Tính,

username:Tên Đăng Nhập,

password:Mật Khẩu,

contact\_info: thông tin liên lạc,

{

phone : Số điện thoại,

email: Email,

address: Địa chỉ

},

position:Chức vụ,

department:Phòng ban,

hire-date:Ngày bắt đầu làm

salary:mức lương,

status:trạng thái

}

* + - Các trường chính:
* employee\_id: Mã định danh duy nhất của nhân viên, giúp phân biệt các nhân viên trong hệ thống.
* full\_name: Họ và tên của nhân viên.
* dob: Ngày sinh, để quản lý thông tin cá nhân và tuổi tác.
* gender: Giới tính, giúp tổ chức hiểu rõ hơn về đội ngũ nhân viên.
* username và password: Thông tin đăng nhập cho hệ thống.
* contact\_info: Thông tin liên lạc bao gồm số điện thoại, email, và địa chỉ của nhân viên.
* position: Chức vụ của nhân viên trong tổ chức.
* department: Phòng ban mà nhân viên thuộc về (như kinh doanh, hành chính, v.v.).
* hire\_date: Ngày bắt đầu làm việc của nhân viên.
* salary: Mức lương của nhân viên.
* status: Trạng thái hiện tại của nhân viên (đang làm việc, nghỉ việc, v.v.).
  + - 1. Collection Customer

**Mô tả**: Collection này chứa thông tin về khách hàng, người tham gia vào các hợp đồng bảo hiểm nhân thọ. Thông tin này giúp tổ chức quản lý và phục vụ khách hàng một cách hiệu quả.

**Cấu Trúc** :

{

customer\_id: Mã Khách Hàng,

full\_name: Họ Tên Khách Hàng,

dob: Ngày Sinh,

gender: Giới Tính,

contact\_info: Thông Tin Liên Lạc

{

phone:Số Điện Thoại ,

email: email ,

address: Địa Chỉ

},

policies: Hợp đồng bảo hiểm

[

{

policy\_number: Số hợp đồng bảo hiểm,

policy\_type: Loại hợp đồng ,

start\_date: Ngày bắt đầu tạo hợp đồng,

end\_date: Ngày kết thúc hợp đồng,

premium\_amount: Số tiền bảo hiểm cần phải đóng hàng năm (VNĐ),

coverage\_amount: Là số tiền tối đa mà công ty bảo hiểm sẽ chi trả trong trường hợp xảy ra sự kiện bảo hiểm (ví dụ: tử vong, tai nạn, bệnh tật, hỏng hóc tài sản...). sau yêu cầu thụ hưởng (VNĐ),

status: trạng thái ,

beneficiary: Người Hưởng Thụ

{

full\_name: Tên Người Hưởng Thụ,

relationship:Mối quan hệ ,

contact\_info: Thông tin liên lạc

{

phone: số điện thoại,

email: email

},

},

next\_payment\_date: Ngày đóng bảo hiểm định kỳ

}

],

claims:

[

{

claim\_number: Mã số bồi thường,

policy\_number: Số hợp đồng ,

claim\_date: Ngày tạo bồi thường,

claim\_amount: số tiền được bồi thường,

status: trạng thái,

documents: tài liệu liên quan

{

death\_certificate:giấy chứng tử,

insurance\_policy: bản sao hợp đồng,

proof\_of\_relationship: tài liệu chứng minh quan hệ

}

},

],

[…]

},

payments:

[

{

payment\_id: Mã số thanh toán,

payment\_date: ngày tạo hóa đơn thanh toán,

amount: số tiền thanh toán,

method: hình thức thanh toán,

status: trạng thái,

payment\_type: loại thanh toán(thanh toán hợp đông/ yêu cầu bồi thường),

related\_transaction: Thông Tin Giao Dịch

{

policy\_number:Mã số được giao dịch (hợp đồng / bồi thường)

},

employee\_id:mã nhân viên lập hóa đơn thanh toán

],

[…]

}

status: Trạng Thai,

agent\_id: mã đại lý bảo hiểm

}

**Các trường chính:**

* customer\_id: Mã định danh duy nhất của khách hàng, giúp phân biệt giữa các khách hàng trong hệ thống.
* full\_name: Họ và tên của khách hàng.
* dob: Ngày sinh của khách hàng, quan trọng cho việc xác định độ tuổi và mức độ rủi ro.
* gender: Giới tính, thông tin này có thể ảnh hưởng đến các sản phẩm bảo hiểm và mức phí bảo hiểm.
* contact\_info: Thông tin liên lạc bao gồm số điện thoại, email, và địa chỉ.
* policies: Danh sách các hợp đồng bảo hiểm mà khách hàng đã tham gia, bao gồm các thông tin chi tiết như số hợp đồng, loại hợp đồng, ngày bắt đầu và ngày kết thúc.
* claims: Danh sách các yêu cầu bồi thường của khách hàng, giúp theo dõi các tình huống xảy ra trong quá trình sử dụng dịch vụ.
* payments: Lịch sử thanh toán của khách hàng cho các hợp đồng bảo hiểm, bao gồm thông tin về ngày thanh toán, số tiền, và hình thức thanh toán.
* status: Trạng thái hiện tại của khách hàng (đang hoạt động, đã ngừng tham gia, v.v.).
* agent\_id: Mã đại lý bảo hiểm đại diện cho khách hàng, giúp dễ dàng theo dõi mối quan hệ giữa khách hàng và đại lý.
  + - 1. Collection Agent

**Mô tả**: Collection này lưu trữ thông tin về các đại lý bảo hiểm, những người làm trung gian giữa công ty bảo hiểm và khách hàng. Đại lý đóng vai trò quan trọng trong việc tiếp cận khách hàng và tư vấn các sản phẩm bảo hiểm.

**Cấu Trúc**:

Collection : agent

{

agent\_id: mã đại lý,

name: Tên Đại Lý ,

contact\_info:

{

phone: Số điện thoại đại lý,

email:email đại lý,

address: địa chỉ đại lý

}

}

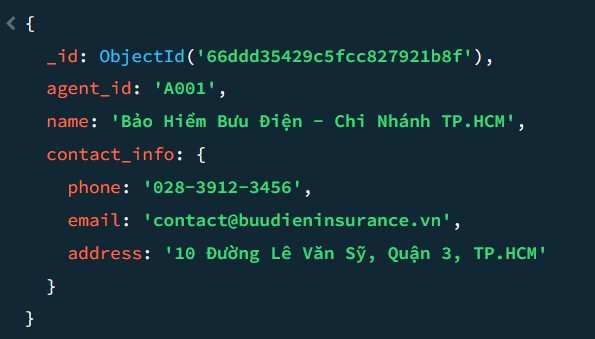
**Các trường chính :**

* **agent\_id**: Mã định danh duy nhất của đại lý, dùng để phân biệt giữa các đại lý.
* **name**: Tên của đại lý.
* **contact\_info**: Thông tin liên lạc của đại lý, bao gồm số điện thoại, email, và địa chỉ. Thông tin này giúp khách hàng dễ dàng liên hệ với đại lý.
  + 1. Diễn giải chi tiết các đối tượng trong mô hình ví dụ minh hoạ
       1. Nhân viên (Employee)
       - Cấu Trúc :

Hình 2. 1Cấu trúc collection nhân viên

* + - * Giải thích : Lê Duy Khương là một nhân viên bảo hiểm làm việc tại Bộ phận bảo hiểm của công ty từ ngày 01/03/2021. Với mức lương 15,000,000 VND hàng tháng, anh phụ trách tư vấn và hỗ trợ khách hàng trong việc chọn sản phẩm bảo hiểm phù hợp. Thông tin liên lạc của anh bao gồm số điện thoại 0909-876-543 và email tran.c@example.com, giúp khách hàng dễ dàng liên hệ khi cần hỗ trợ.
      1. Khách Hàng
      * Cấu Trúc

Hình 2. 2Cấu trúc collection khách hàng

* + - * Giải Thích : Nguyễn Văn An là một khách hàng sử dụng dịch vụ bảo hiểm nhân thọ trọn đời với hai hợp đồng bảo hiểm khác nhau. Ông sinh ngày 01/01/1980 và hiện đang sống tại 123 Đường Lê Lợi, Quận 1, TP.HCM. Với số điện thoại 123-456-7890 và email nguyen.an@example.com, ông dễ dàng liên hệ với đại lý hoặc công ty để giải quyết các vấn đề liên quan đến hợp đồng của mình. Trong thời gian qua, ông đã thực hiện hai yêu cầu bồi thường, một yêu cầu đang trong quá trình xử lý.
      1. Đại Lý
      * Cấu trúc

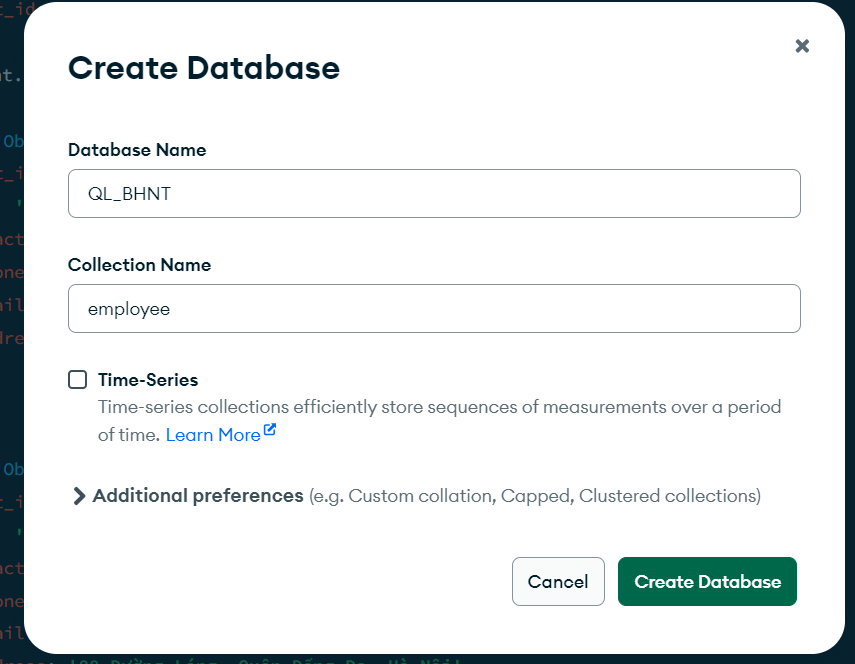
Hình 2. 3 Cấu trúc collection đại lý

* + - * Giải Thích: Bảo Hiểm Bưu Điện - Chi Nhánh TP.HCM là một đại lý bảo hiểm, có nhiệm vụ tư vấn và hỗ trợ khách hàng trong việc chọn lựa các sản phẩm bảo hiểm. Với số điện thoại 028-3912-3456 và email contact@buudieninsurance.vn, đại lý này giúp khách hàng có thể dễ dàng liên hệ để được tư vấn và giải đáp thắc mắc. Địa chỉ của chi nhánh là 10 Đường Lê Văn Sỹ, Quận 3, TP.HCM.
    1. Tạo cơ sở dữ liệu và các đồi tượng trong CSD
       1. Tạo cơ sở dữ liệu
* Tạo database “QL\_BHNT”

Bước 1**: Mở MongoDB Compass**

1. **Khởi động MongoDB Compass** trên máy tính của bạn.
2. **Kết nối** với MongoDB Server bằng cách nhập URI kết nối. Nếu bạn đang chạy MongoDB trên máy cục bộ, URI thường sẽ là:

**Bước 2: Tạo Mới Database**

1. Sau khi kết nối thành công, bạn sẽ thấy giao diện chính của MongoDB Compass.
2. Nhấn vào **"Create Database"** ở góc trên bên trái màn hình.

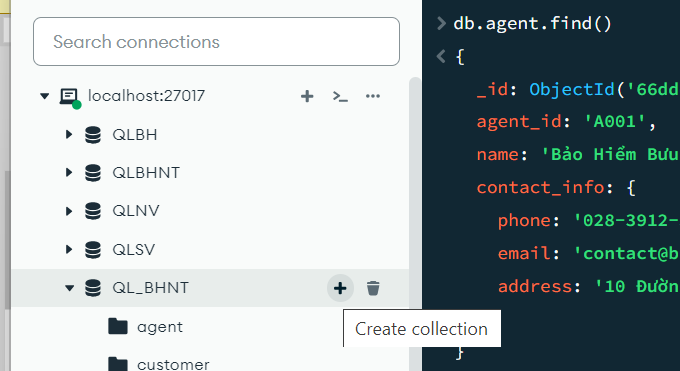
Hình 2. 4 Tạo mới database bằng công cụ

1. Một cửa sổ mới sẽ xuất hiện yêu cầu bạn nhập thông tin cho database mới. Nhập:**Database Name**: Nhập tên database mà bạn muốn tạo ( QL\_BHNT), **Collection Name**: Nhập tên collection đầu tiên (employee).
2. Nhấn **Create Database** để hoàn tất việc tạo database và collection.

**Bước 3:**Thêm 2 collection “customer” và “agent” vào cơ sở dữ liệu.

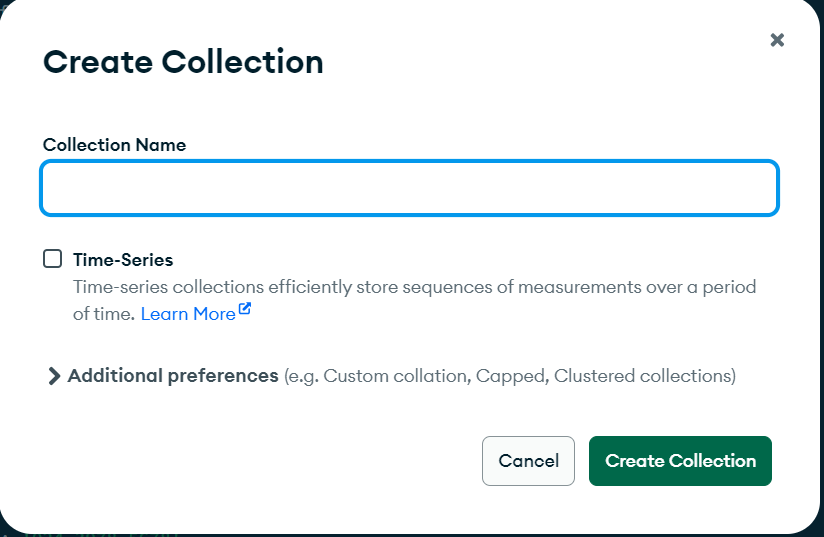
* Thực hiện bằng công cụ :

**Bước 1: Chọn Database QL\_BHNT**

1. Trong bảng điều khiển bên trái, tìm kiếm database **QL\_BHNT** mà bạn đã tạo.
2. Nhấp vào tên database **QL\_BHNT** để mở nó.

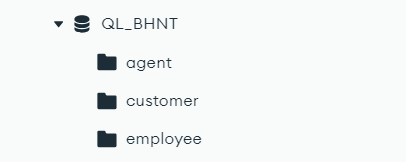
Hình 2. 5 Tạo mới collection

**Bước 2: Tạo Collection “customer”**

1. Trong trang database **QL\_BHNT**, bạn sẽ thấy nút **"Create Collection"**. Nhấp vào đó.
2. Nhập tên collection là **customer** vào ô nhập liệu và nhấn **Create Collection**

Hình 2. 6 Nhập tên collection tạo mới

**Bước 3: Tạo Collection “agent”**

1. Tiếp tục, nhấp vào nút **"Create Collection"** một lần nữa.
2. Nhập tên collection là **agent** và nhấn **Create Collection**.

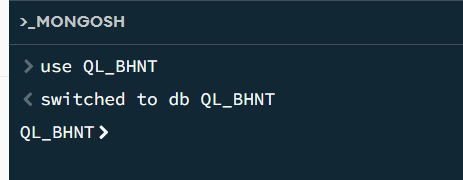
Hình 2. 7 Collection được tạo thành công

* Thực hiện bằng câu truy vấn :

**Bước 1: Mở MongoDB Compass và Kết Nối**

* + Mở MongoDB Compass và nhập thông tin kết nối của bạn (thường là mongodb://localhost:27017 nếu bạn chạy MongoDB trên máy local).

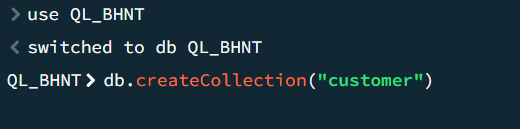
**Bước 2: Tạo Database QL\_BHNT**

* + Sau khi kết nối, bạn sẽ sử dụng lệnh sau trong tab "Shell":

Hình 2. 8 Sử dụng database bằng lệnh

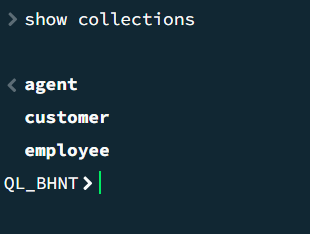
**Bước 3: Tạo Collection “customer” và “agent”**

* + Sau khi đã chọn database QL\_BHNT, sẽ tạo hai collection bằng các lệnh sau:
  + Tạo Collection customer



Hình 2. 9 Tạo mới collection bằng lệnh

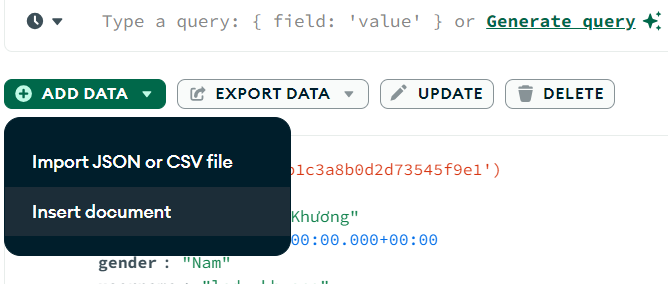
**Bước 4: Kiểm Tra Các Collection Đã Tạo**

* + Sau khi tạo các collection, bạn có thể kiểm tra chúng bằng lệnh:

Hình 2. 10 Hiện thị collection đã tạo

* + - 1. Tạo đối tượng trong collection và nhập cơ sở dữ liệu
         1. Thực hiện bằng công cụ

Bước 1: Mở Collection employee

1. Trở lại danh sách các collection trong database QL\_BHNT.
2. Chọn collection employee từ danh sách các collection bên trái.

Bước 2: Chọn cách thức nhập cơ sở dữ liệu

* + Trường hợp import Json or CSV file : cơ sở dữ liệu sẽ tự động lưu trữ vào collection chọn “add data”.
  + Trường hợp insert document

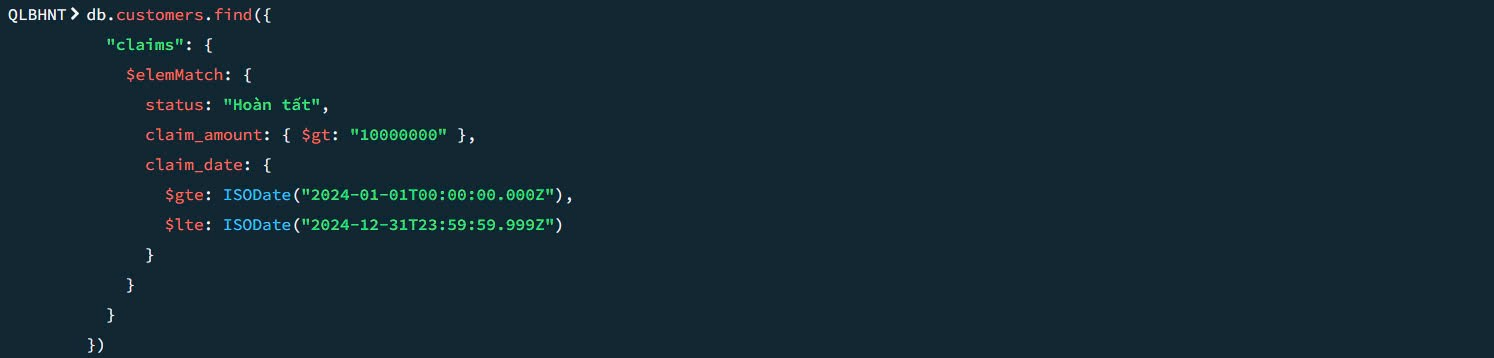
Hình 2. 11 Chèn dữ liêu bằng công cụ

* + - * 1. Thực hiện bằng câu truy vấn

Hình 2. 12 Chèn dữ liệu bằng câu lệnh

* + 1. Truy vấn dữ liệu
       1. Truy vấn Yêu Cầu Bồi Thường

**1.Truy vấn để tìm các yêu cầu bồi thường trong khoảng thời gian cụ thể, với trạng thái "Hoàn tất" và số tiền yêu cầu bồi thường lớn hơn 10,000,000:**



Giải thích từng thành phần:

db.customers.find({ ... }):

Đây là phương thức tìm kiếm của MongoDB, tìm tất cả các khách hàng trong bộ sưu tập customers thỏa mãn điều kiện chỉ định.

"claims": { $elemMatch: { ... } }:

$elemMatch được sử dụng để tìm kiếm các phần tử bên trong mảng. Trong trường hợp này, nó đang tìm kiếm các phần tử bên trong mảng claims của mỗi khách hàng.

claims là một mảng chứa nhiều yêu cầu bồi thường của khách hàng. Mỗi yêu cầu là một phần tử trong mảng này.

status: "Hoàn tất":

Điều kiện này tìm các yêu cầu bồi thường có trạng thái là "Hoàn tất" (đã hoàn thành quá trình xử lý).

claim\_amount: { $gt: "10000000" }:

claim\_amount là số tiền yêu cầu bồi thường. Điều kiện này tìm các yêu cầu bồi thường có số tiền lớn hơn 10,000,000 đồng.

Toán tử $gt (greater than) chỉ định rằng giá trị claim\_amount phải lớn hơn giá trị đã chỉ định (trong trường hợp này là 10,000,000 đồng).

claim\_date: { $gte: ..., $lte: ... }:

Điều kiện này tìm các yêu cầu bồi thường có ngày tạo (claim\_date) nằm trong khoảng thời gian chỉ định.

$gte (greater than or equal) được sử dụng để chỉ định ngày bắt đầu của khoảng thời gian, ở đây là ngày 1 tháng 1 năm 2024.

$lte (less than or equal) được sử dụng để chỉ định ngày kết thúc của khoảng thời gian, ở đây là ngày 31 tháng 12 năm 2024.

ISODate:

ISODate là hàm MongoDB dùng để biểu diễn giá trị ngày giờ theo tiêu chuẩn ISO, giúp lưu trữ và so sánh các giá trị thời gian một cách chính xác.

**2. Truy vấn để tìm tất cả các khách hàng có yêu cầu bồi thường với trạng thái là "Chờ xử lý", kèm theo chi tiết hợp đồng bảo hiểm có số tiền bảo hiểm (coverage\_amount) lớn hơn 400 triệu VND:**

Giải thích từng thành phần:

1. db.customers.find({ ... }):

Đây là hàm find của MongoDB, được sử dụng để tìm kiếm các tài liệu (documents) trong bộ sưu tập customers. Kết quả trả về sẽ là những khách hàng thỏa mãn tất cả các điều kiện trong truy vấn.

2. "claims.status": "Chờ xử lý":

claims là một trường có thể chứa mảng các yêu cầu bồi thường (claims).

Truy vấn này tìm các khách hàng có ít nhất một yêu cầu bồi thường trong mảng claims với trạng thái là "Chờ xử lý". Điều này có nghĩa là yêu cầu bồi thường này vẫn đang trong quá trình chờ xử lý và chưa được hoàn thành.

Ví dụ:

Một khách hàng có một yêu cầu bồi thường trong mảng claims với trạng thái là "Chờ xử lý" sẽ được trả về trong kết quả của truy vấn.

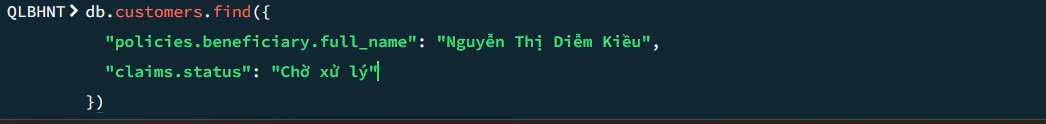
3. "policies.coverage\_amount": { $gt: 40000000 }:

policies là một trường khác, có thể là mảng chứa các hợp đồng bảo hiểm mà khách hàng đã mua.

coverage\_amount là trường trong mỗi hợp đồng, lưu trữ số tiền bảo hiểm. Truy vấn này tìm các hợp đồng bảo hiểm có số tiền bảo hiểm lớn hơn 40,000,000 đồng.

Toán tử $gt (greater than) được sử dụng để lọc các giá trị của coverage\_amount lớn hơn giá trị chỉ định, trong trường hợp này là 40,000,000 đồng.

**3. Truy vấn để tìm tất cả các yêu cầu bồi thường có người thụ hưởng (beneficiary) là "Nguyễn Thị Diễm Kiều" và trạng thái yêu cầu là "Chờ xử lý":**

Giải thích từng phần:

1. db.customers.find({ ... }):

find là phương thức MongoDB được sử dụng để tìm kiếm tài liệu (documents) trong một bộ sưu tập. Trong trường hợp này, bạn đang tìm kiếm trong bộ sưu tập customers, nơi lưu trữ thông tin của tất cả các khách hàng.

2. "policies.beneficiary.full\_name": "Nguyễn Thị Diễm Kiều":

policies là một trường có thể chứa mảng các hợp đồng bảo hiểm mà khách hàng đã mua. Trong mỗi hợp đồng bảo hiểm, có thể có một trường beneficiary chứa thông tin về người thụ hưởng của hợp đồng bảo hiểm.

beneficiary.full\_name là trường lưu trữ tên đầy đủ của người thụ hưởng. Truy vấn này tìm các khách hàng có người thụ hưởng trong bất kỳ hợp đồng bảo hiểm nào có tên đầy đủ là "Nguyễn Thị Diễm Kiều".

Ví dụ:

Một khách hàng có hợp đồng bảo hiểm và người thụ hưởng có tên đầy đủ là Nguyễn Thị Diễm Kiều sẽ được trả về trong kết quả của truy vấn.

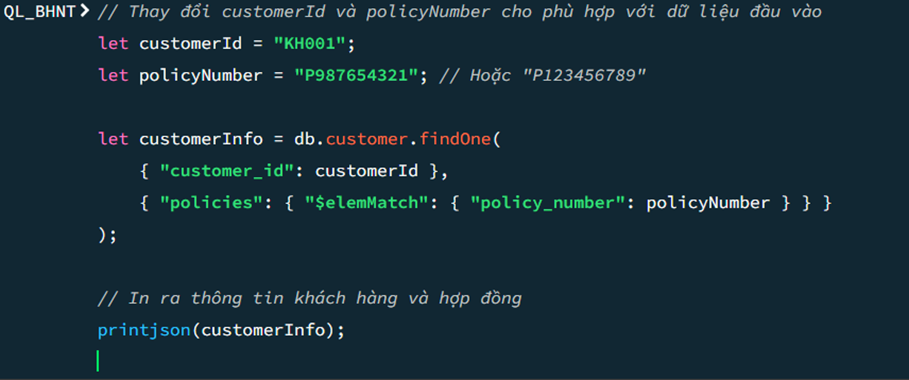
3. "claims.status": "Chờ xử lý":

claims là một trường khác, có thể là mảng chứa các yêu cầu bồi thường (claims) mà khách hàng đã thực hiện.

status là trường lưu trữ trạng thái của mỗi yêu cầu bồi thường. Truy vấn này tìm các khách hàng có ít nhất một yêu cầu bồi thường trong mảng claims với trạng thái là "Chờ xử lý".

* + - 1. Truy vấn Hợp Đồng

**Câu truy vấn 1: Tìm kiếm thông tin khách hàng theo CustomerId và PolicyNumber**



Mô Tả Chi Tiết Các Giai Đoạn Truy Vấn

Giai Đoạn 1: Tìm Kiếm Khách Hàng

·        Mục đích: Lọc thông tin của khách hàng theo customer\_id.

·        Cơ chế hoạt động:

o   Sử dụng findOne() để tìm tài liệu khách hàng có customer\_id là "KH001".

o   Sử dụng $elemMatch để lọc ra chính sách mà có policy\_number là "P987654321" từ mảng policies.



**Câu truy vấn 2: Tính tổng số tiền bảo hiểm cho tất cả các hợp đồng của một khách hàng**



Mô Tả Chi Tiết Các Giai Đoạn Truy Vấn

Giai Đoạn 1: Lọc Khách Hàng

·        Mục đích: Lọc tài liệu khách hàng dựa trên customer\_id.

·        Cơ chế hoạt động:

o   Sử dụng $match để chỉ giữ lại tài liệu của khách hàng có customer\_id là "KH001".

o   Kết quả của giai đoạn này sẽ chỉ bao gồm các thông tin của khách hàng cụ thể đó.

Giai Đoạn 2: Tách Các Chính Sách

·        Mục đích: Tách mảng policies để mỗi chính sách trở thành một tài liệu riêng biệt.

·        Cơ chế hoạt động:

o   Sử dụng $unwind để tạo ra một bản sao của tài liệu cho mỗi phần tử trong mảng policies.

o   Nếu khách hàng có nhiều chính sách, giai đoạn này sẽ tạo ra nhiều tài liệu, mỗi tài liệu chứa thông tin về một chính sách.

Giai Đoạn 3: Nhóm và Tính Tổng Số Tiền Bảo Hiểm

·        Mục đích: Nhóm các tài liệu đã lọc và tính tổng số tiền bảo hiểm cho các chính sách.

·        Cơ chế hoạt động:

o   Sử dụng $group để nhóm tất cả các tài liệu của khách hàng lại với nhau. Giá trị \_id là "$customer\_id" để nhóm theo customer\_id.

o   Sử dụng $sum để tính tổng số tiền bảo hiểm từ trường coverage\_amount của các chính sách.



**Câu truy vấn 3: Tìm kiếm hợp đồng có số tiền phí cao nhất cho một khách hàng cụ thể**



* Mô Tả Chi Tiết Các Giai Đoạn Truy Vấn

Giai Đoạn 1: Lọc Khách Hàng

·        Mục đích: Lọc tài liệu khách hàng dựa trên customer\_id.

·        Cơ chế hoạt động:

o   Sử dụng $match để chỉ giữ lại tài liệu của khách hàng có customer\_id là "KH001".

o   Kết quả của giai đoạn này sẽ chỉ bao gồm thông tin của khách hàng cụ thể đó.

Giai Đoạn 2: Tách Các Chính Sách

·        Mục đích: Tách mảng policies để mỗi chính sách trở thành một tài liệu riêng biệt.

·        Cơ chế hoạt động:

o   Sử dụng $unwind để tạo ra một bản sao của tài liệu cho mỗi phần tử trong mảng policies.

o   Nếu khách hàng có nhiều chính sách, giai đoạn này sẽ tạo ra nhiều tài liệu, mỗi tài liệu chứa thông tin về một chính sách.

Giai Đoạn 3: Sắp Xếp Theo Số Tiền Phí

·        Mục đích: Sắp xếp các chính sách theo số tiền phí từ cao đến thấp.

·        Cơ chế hoạt động:

o   Sử dụng $sort với điều kiện { "policies.premium\_amount": -1 } để sắp xếp các tài liệu đã tách ra theo trường premium\_amount theo thứ tự giảm dần.

Giai Đoạn 4: Giới Hạn Kết Quả

·        Mục đích: Lấy chính sách có số tiền phí cao nhất.

·        Cơ chế hoạt động:

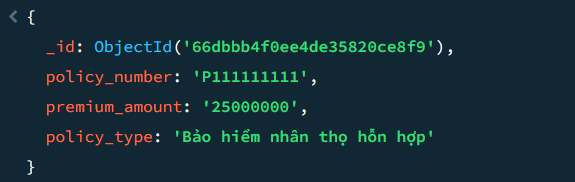
o   Sử dụng $limit: 1 để chỉ lấy một tài liệu, đó chính là chính sách có số tiền phí cao nhất.

Giai Đoạn 5: Lựa Chọn Các Trường Cần Hiển Thị

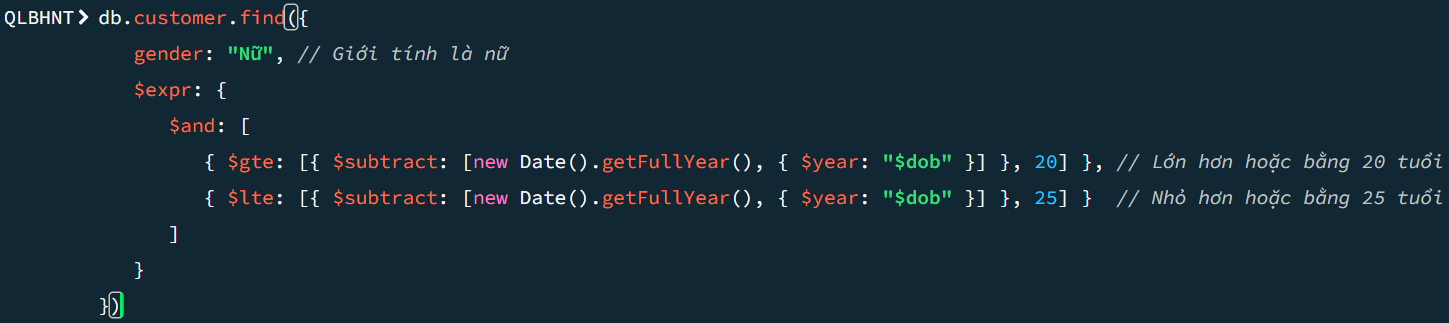
·        Mục đích: Chọn các trường thông tin cần thiết để hiển thị trong kết quả cuối cùng.

·        Cơ chế hoạt động:

o   Sử dụng $project để chỉ định các trường mà chúng ta muốn giữ lại, bao gồm policy\_number, premium\_amount, và policy\_type.



* + - 1. Truy vấn Khách Hàng

Tìm kiếm khách hàng nữ trong độ tuổi từ 20 đến 25.

Hình 3. 1 Tìm kiếm khách hàng nữ trong độ tuổi 20 đến 25

* Kết quả:
* Giải thích :
* **Lọc theo giới tính**:

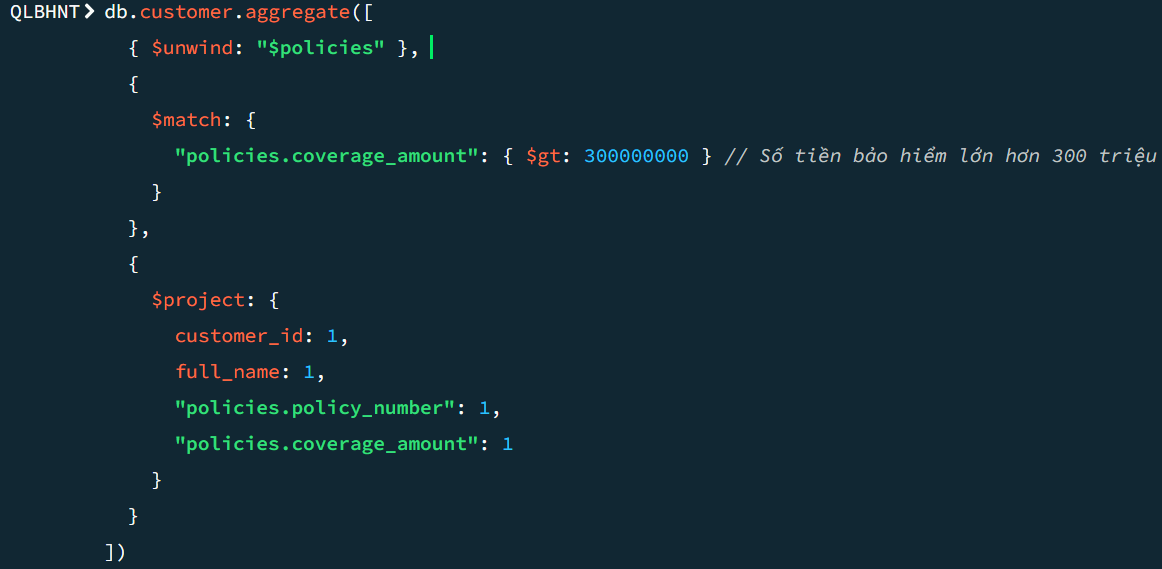
**gender: "Nữ"**: Lọc các khách hàng có giới tính là "Nữ".

* **Tính toán độ tuổi**:

**$expr: { $and: [...] }**: Sử dụng biểu thức để tính toán và so sánh độ tuổi.

**$subtract**: Tính tuổi bằng cách lấy năm hiện tại trừ năm sinh (dob).

**$gte và $lte**: Kiểm tra xem độ tuổi có nằm trong khoảng từ 20 đến 25 hay không, yêu cầu cả hai điều kiện phải thỏa mãn.

Truy vấn khách hàng có chính sách bảo hiểm với giá trị cao hơn một mức nhất định:

Hình 3. 2Truy vấn khách hàng có chính sách bảo hiểm

* Kết quả:
* Giải thích:
* Giai đoạn 1: $unwind:

$unwind: "$policies": Tách từng phần tử trong mảng policies (các hợp đồng bảo hiểm) của mỗi khách hàng thành các tài liệu riêng biệt.

Mục đích: Mỗi hợp đồng bảo hiểm sẽ được xử lý riêng lẻ, giúp dễ dàng thao tác trên từng hợp đồng của khách hàng.

* Giai đoạn 2: $match:

$match: { "policies.coverage\_amount": { $gt: 300000000 } }: Lọc ra những hợp đồng bảo hiểm có số tiền bảo hiểm (coverage\_amount) lớn hơn 300 triệu.

Mục đích: Chỉ giữ lại những khách hàng có hợp đồng bảo hiểm với số tiền lớn hơn 300 triệu để phân tích hoặc báo cáo.

* Giai đoạn 3: $project:

$project: { customer\_id: 1, full\_name: 1, "policies.policy\_number": 1, "policies.coverage\_amount": 1 }: Chọn và hiển thị các trường cần thiết trong kết quả bao gồm:

customer\_id: Mã khách hàng.

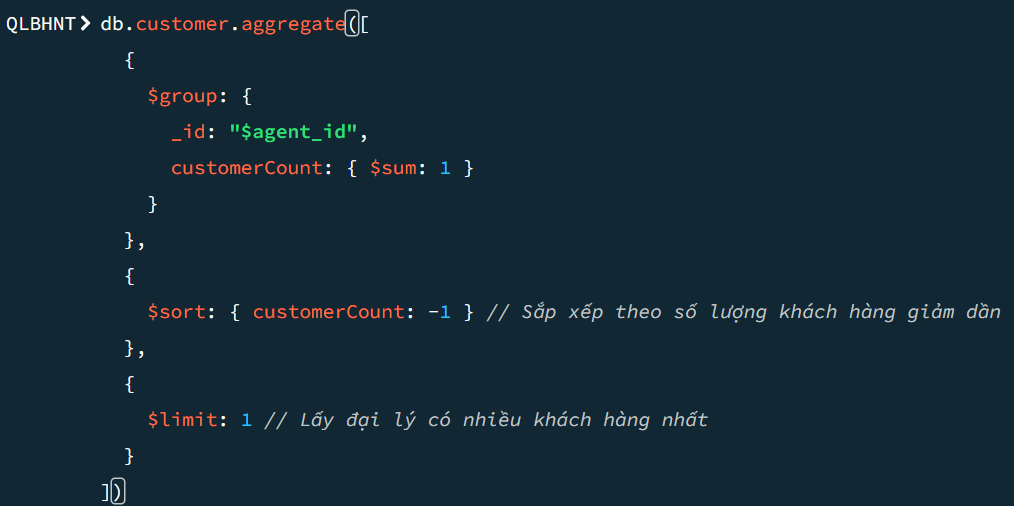
full\_name: Tên đầy đủ của khách hàng.

policies.policy\_number: Mã số hợp đồng bảo hiểm.

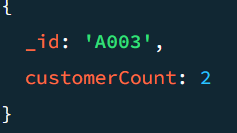
policies.coverage\_amount: Số tiền bảo hiểm.

Mục đích: Chỉ giữ lại các trường thông tin quan trọng về khách hàng và hợp đồng bảo hiểm cho kết quả cuối cùng, giúp dữ liệu trở nên dễ hiểu và ngắn gọn hơn.

Truy vấn đại lý bảo hiểm có nhiều khách hàng nhất



Hình 3. 33. Truy vấn đại lý bảo hiểm có nhiều khách hàng nhất

* Kết quả:
* Giải thích chi tiết
* **Giai đoạn 1: Nhóm khách hàng theo mã đại lý**

**$group**:

**Mục đích**: Gộp các khách hàng lại dựa trên **agent\_id** (mã đại lý). Điều này có nghĩa là tất cả các khách hàng thuộc cùng một đại lý sẽ được gộp lại thành một nhóm.

**Cơ chế**: Đối với mỗi **agent\_id**, chúng ta tạo ra một nhóm. **$sum: 1** được sử dụng để đếm số lượng khách hàng trong mỗi nhóm, tương ứng với số lượng khách hàng của mỗi đại lý.

* **Giai đoạn 2: Sắp xếp theo số lượng khách hàng**

**$sort**:

**Mục đích**: Sắp xếp các đại lý theo số lượng khách hàng giảm dần.

**Cơ chế**: **$sort: { customerCount: -1 }** sắp xếp các nhóm đại lý dựa trên **customerCount** (số lượng khách hàng), từ lớn đến nhỏ.

* **Giai đoạn 3: Lấy đại lý có nhiều khách hàng nhất**

**$limit**:

**Mục đích**: Giới hạn kết quả chỉ lấy đại lý có số lượng khách hàng nhiều nhất.

**Cơ chế**: **$limit: 1** giới hạn kết quả trả về là 1 tài liệu, nghĩa là chỉ lấy ra đại lý đứng đầu trong danh sách sau khi sắp xếp.

* + - 1. Truy vấn trường thanh toán

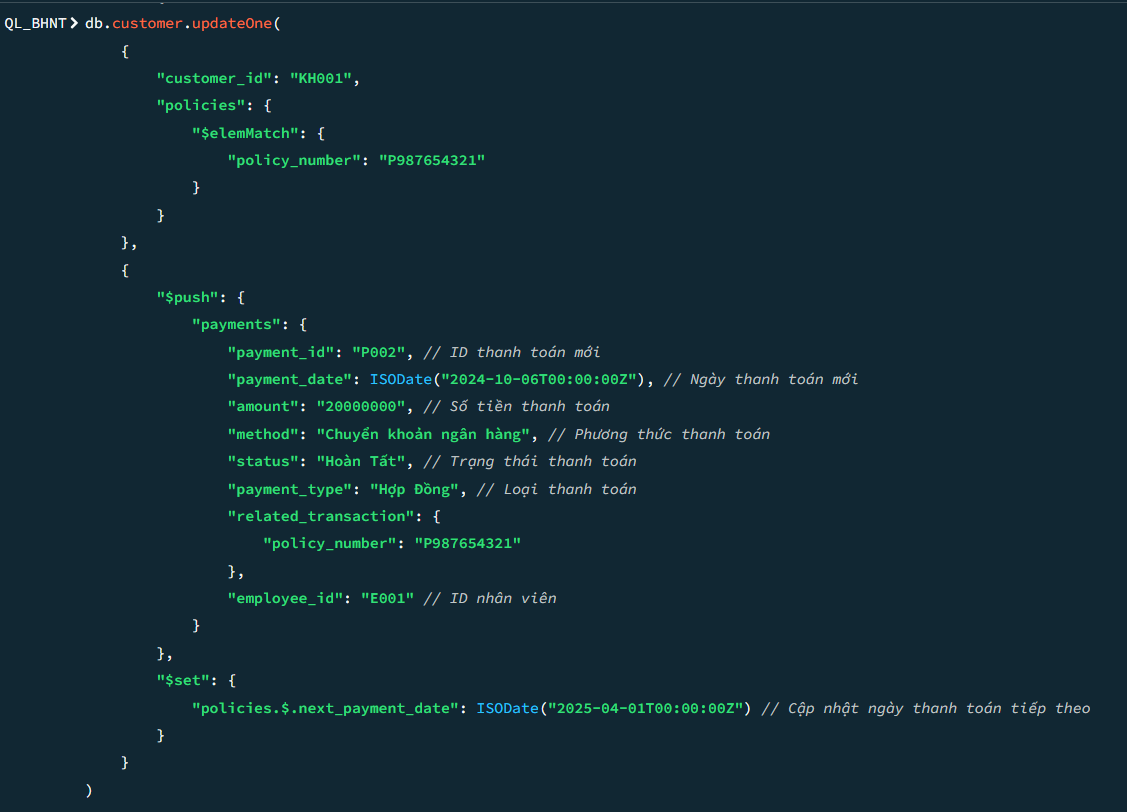
Thực hiện thanh toán tiền đóng bảo hiểm định kỳ của khách hàng [customer\_id] với hợp đồng được thanh toán là [policy\_number].

* Cấu trúc câu lệnh :  
  db.customers.updateOne(

{ // Giai đoạn 1: Tìm kiếm },

{ // Giai đoạn 2: Thêm vào mảng

// Giai đoạn 3: Cập nhật trường thông tin}

)

Hình 2. 13 Truy vấn thanh toán định kỳ cho một hợp đồng

### Giải thích chi tiết:

### Giai đoạn 1: Tìm kiếm

* + - * **Mục đích**: Tìm tài liệu của khách hàng cần thực hiện thanh toán.
      * **Điều kiện tìm kiếm**:
* "customer\_id": "KH001": Xác định khách hàng có customer\_id là "KH001".
* "policies": { "$elemMatch": { "policy\_number": "P987654321" } }: Kiểm tra trong mảng policies có tồn tại hợp đồng với policy\_number là "P987654321" hay không.
  + - * **Cơ chế**:
* **customer\_id: "KH001"**: MongoDB tìm các tài liệu có customer\_id bằng "KH001".
* **$elemMatch**: Kiểm tra xem mảng policies có chứa một phần tử có policy\_number khớp với "P987654321" hay không.

### Giai đoạn 2: Thêm vào mảng payments

* **Mục đích**: Thêm một giao dịch thanh toán mới vào mảng payments.
* **Cấu trúc $push**:
* Sử dụng $push để thêm đối tượng thanh toán mới vào mảng payments.
* Các trường của đối tượng thanh toán bao gồm:
  + - **payment\_id**: Mã thanh toán.
    - **payment\_date**: Ngày thanh toán.
    - **amount**: Số tiền thanh toán.
    - **method**: Phương thức thanh toán (ví dụ: Chuyển khoản ngân hàng).

**status**: Trạng thái của thanh toán (ví dụ: Đang xử lý).

**payment\_type**: Loại thanh toán (ví dụ: Hợp đồng).

**related\_transaction**: Thông tin về giao dịch liên quan (hợp đồng cần thanh toán).

**employee\_id**: ID nhân viên thực hiện thanh toán.

* **Cơ chế**:
  + **$push** thêm đối tượng thanh toán mới vào mảng payments hiện tại của tài liệu khách hàng.

### Giai đoạn 3: Cập nhật trường next\_payment\_date của hợp đồng

* **Mục đích**: Cập nhật ngày thanh toán tiếp theo của hợp đồng đã thanh toán.
* **Cấu trúc $set**:
* "policies.$.next\_payment\_date": ISODate("2025-04-01T00:00:00Z"): Sử dụng toán tử $set để cập nhật trường next\_payment\_date của phần tử trong mảng policies khớp với điều kiện tìm kiếm ban đầu.
* **Cơ chế**:
* **$** là toán tử chỉ mục để tham chiếu đến phần tử trong mảng policies được tìm thấy.

Lọc danh sách tất cả các giao dịch (payments) mà có payment\_type là "Yêu cầu bồi thường"

* Cấu trúc lệnh :

db.customers.aggregate([

{// Giai đoạn 1: Thao tác tách các phần tử trong mảng},

{ // Giai đoạn 2: Thao tác lọc dữ liệu},

{ // Giai đoạn 3: Thao tác chọn trường để hiển thị }

]);

Hình 2. 14 Lọc danh sách giao dịch là loại yêu cầu bồi thường

* Giải thích chi tiết :
* Giai đoạn 1 : Thao tác tách các phần tử giao dịch trong mảng paymen

$unwind mảng payments

* **Mục đích**:
* $unwind được dùng để tách từng phần tử trong mảng payments thành các tài liệu riêng biệt. Điều này giúp chúng ta dễ dàng thao tác với từng giao dịch thanh toán của khách hàng.
* **Cơ chế**:
* Đối với mỗi phần tử của mảng payments, $unwind sẽ tạo ra một tài liệu mới, mỗi tài liệu chứa một phần tử payment từ mảng đó.
* Điều này hữu ích khi chúng ta cần truy vấn từng thanh toán riêng lẻ của khách hàng thay vì xử lý cả mảng trong tài liệu ban đầu.
* Giai đoạn 2:Thao tác lọc **thanh toán có loại yêu cầu bồi thường**

**$match thanh toán có payment\_type là "Yêu cầu bồi thường"**

* **Mục đích**:
  + $match được dùng để lọc ra các tài liệu phù hợp với điều kiện cụ thể: chỉ lấy những giao dịch có loại thanh toán (payment\_type) là "Yêu cầu bồi thường".
* **Điều kiện lọc**:
  + payments.payment\_type: "Yêu cầu bồi thường": Chỉ những thanh toán có payment\_type là "Yêu cầu bồi thường" mới được giữ lại trong kết quả.
* **Cơ chế**:
  + Sau khi $unwind tách các phần tử của mảng payments, $match sẽ áp dụng bộ lọc để chỉ giữ lại những tài liệu mà payment\_type bằng "Yêu cầu bồi thường".
* Giai đoạn 3: Lựa chọn các trường để hiển thị
* **$project các trường cần hiển thị**
* **Mục đích**:
* $project được dùng để chỉ định các trường mà chúng ta muốn xuất hiện trong kết quả cuối cùng, giúp kiểm soát và làm gọn dữ liệu trả về.
* **Các trường được chọn**:
  + customer\_id: Lấy mã khách hàng (customer\_id).
  + payments.payment\_id: Lấy mã thanh toán (payment\_id).
  + payments.payment\_date: Lấy ngày thanh toán (payment\_date).
  + payments.amount: Lấy số tiền thanh toán (amount).
  + payments.method: Lấy phương thức thanh toán (method).
  + payments.status: Lấy trạng thái thanh toán (status).
  + payments.payment\_type: Lấy loại thanh toán (payment\_type).
  + payments.related\_transaction: Lấy thông tin giao dịch liên quan (related\_transaction).
  + payments.employee\_id: Lấy mã nhân viên (employee\_id).
* **Cơ chế**:
  + "\_id": 0: Không hiển thị trường \_id mặc định.
  + Các trường khác đều được chỉ định với giá trị 1, nghĩa là chúng sẽ được giữ lại trong kết quả cuối cùng.

Thông kê in ra số lượng thanh toán hợp đồng và số lượng thanh toán yêu cầu bồi thường khi truyền vào tháng năm.

- Cấu trúc :  
db.customers.aggregate([

{ Giai đoạn 1: tách các giao dịch },

{ Giai đoạn 2:thêm trường tháng và năm },

{Giai đoạn 3:lọc thanh toán theo tháng ,năm},

{Giai đoạn 4: Nhóm và đếm các thanh toán theo loại

{Giai đoạn 5:Lựa chọn các trường cần hiểu thị}

* Giải thích chi tiết

Hình 2. 15 Thông kế số lương thanh toán

* Giai đoạn 1: Tách các giao dịch

Tách mảng payments ($unwind)

* **Mục đích**: Giai đoạn này sử dụng $unwind để tách từng phần tử trong mảng payments của tài liệu thành các tài liệu riêng biệt.
* **Cơ chế hoạt động**:
* $unwind tạo ra một bản sao của tài liệu ban đầu cho mỗi phần tử của mảng. Ví dụ, nếu khách hàng có 3 giao dịch trong mảng payments, sau khi $unwind, sẽ có 3 tài liệu mới, mỗi tài liệu chứa thông tin về một thanh toán.
* Giai đoạn 2: thêm trường tháng và năm

Thêm các trường payment\_month và payment\_year ($addFields)

* **Mục đích**: Thêm các trường payment\_month và payment\_year vào tài liệu để thuận tiện cho việc lọc dữ liệu sau này.
* **Cơ chế hoạt động**:
* $addFields cho phép thêm hoặc cập nhật các trường mới cho tài liệu hiện tại.
* Các biểu thức $month và $year được sử dụng để lấy giá trị tháng và năm từ trường payment\_date.
* Giai đoạn 3: lọc thanh toán theo tháng ,năm
* Lọc thanh toán theo tháng và năm ($match)
* **Mục đích**: Giai đoạn này dùng để chỉ giữ lại các thanh toán xảy ra trong tháng và năm cụ thể (ví dụ tháng 10 năm 2024).
* **Cơ chế hoạt động**:
* $match được sử dụng để lọc các tài liệu sao cho chúng đáp ứng các điều kiện cụ thể.
* Trong trường hợp này, chỉ giữ lại các tài liệu có payment\_month là 10 và payment\_year là 2024.
* Giai đoạn 4: Nhóm và đếm các thanh toán theo loại

Nhóm và đếm thanh toán theo payment\_type ($group)

* **Mục đích**: Nhóm tất cả các tài liệu lại và đếm số lượng thanh toán theo hai loại: "Hợp Đồng" và "Yêu cầu bồi thường".
* **Cơ chế hoạt động**:
* $group tạo ra một nhóm từ các tài liệu đã lọc trước đó. Giá trị \_id là null có nghĩa là chúng ta sẽ nhóm tất cả lại thành một nhóm duy nhất (không có phân chia nào).
* $sum được dùng để đếm số lượng thanh toán.
* $cond hoạt động như một câu lệnh điều kiện: nếu điều kiện $eq ($payments.payment\_type bằng "Hợp Đồng") là đúng, thì tăng giá trị lên 1, ngược lại tăng lên 0.
* Giai đoạn 5: Lựa chọn các trường cần hiển thị ($project)
* **Mục đích**: Chỉ định các trường cần xuất hiện trong kết quả cuối cùng, giúp kết quả trở nên ngắn gọn và dễ đọc hơn.
* **Cơ chế hoạt động**:
* $project cho phép chỉ định các trường mà chúng ta muốn giữ lại hoặc loại bỏ.
* \_id: 0 có nghĩa là không hiển thị trường \_id trong kết quả.
* contract\_payments: 1 và claim\_payments: 1 cho biết rằng các trường này sẽ được giữ lại trong kết quả.

1. CÀI ĐẶT CHƯƠNG TRÌNH
   1. Giới thiệu chung
      1. Môi trường phát triển

**Visual Studio 2022**: Đây là phiên bản mới nhất của Visual Studio, cung cấp môi trường phát triển tích hợp (IDE) mạnh mẽ cho việc lập trình. Visual Studio hỗ trợ nhiều loại ngôn ngữ và framework, đặc biệt là **C#** và **.NET Framework**, giúp việc phát triển ứng dụng **Windows Forms (WinForms)** dễ dàng.

Các tính năng nổi bật của Visual Studio bao gồm:

* **Code Editor**: Cung cấp nhiều tính năng hỗ trợ lập trình viên như tự động hoàn thành mã (IntelliSense), refactoring, và điều hướng mã dễ dàng.
* **Debugger**: Công cụ gỡ lỗi tích hợp mạnh mẽ giúp lập trình viên phát hiện và sửa lỗi nhanh chóng.
* **Integrated Git Support**: Hỗ trợ tích hợp quản lý phiên bản Git ngay trong môi trường phát triển, giúp theo dõi thay đổi và cộng tác trong nhóm dễ dàng.
* **NuGet Package Manager**: Hỗ trợ tải và cài đặt các thư viện bên ngoài một cách dễ dàng, đặc biệt là các gói cần thiết như MongoDB.Driver để kết nối với cơ sở dữ liệu MongoDB.

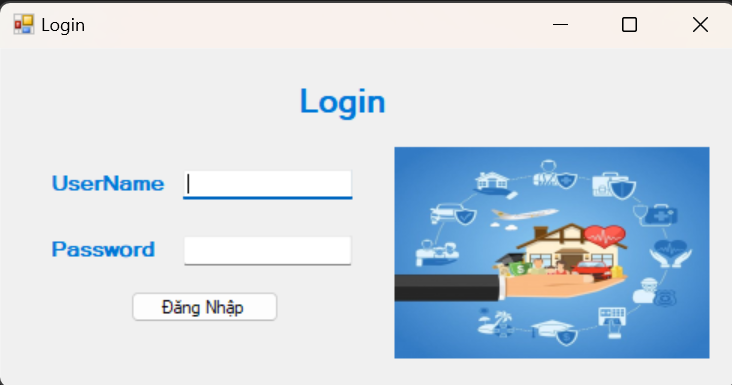
Sử dụng Visual Studio 2022 giúp việc phát triển ứng dụng trở nên nhanh chóng, hiệu quả, và tối ưu hơn nhờ các công cụ hỗ trợ mạnh mẽ này.

* + 1. Hệ điều hành

**Windows**: Ứng dụng được xây dựng trên nền tảng Windows vì nó hỗ trợ tốt nhất cho các ứng dụng được phát triển bằng **C#** và **.NET Framework**. Windows cung cấp các dịch vụ hệ thống, như giao diện đồ họa người dùng (GUI), hệ thống quản lý tập tin, và cơ sở hạ tầng mạng, đảm bảo cho ứng dụng chạy ổn định và tương tác với người dùng một cách trực quan.

Việc phát triển ứng dụng **WinForms** trên hệ điều hành Windows tận dụng tốt sự hỗ trợ này, giúp ứng dụng tương thích với nhiều phiên bản Windows và dễ dàng triển khai.

* + 1. Ngôn ngữ lập trình
* **C# (C-sharp)**: Là ngôn ngữ lập trình hướng đối tượng, được phát triển bởi Microsoft và là lựa chọn lý tưởng cho các ứng dụng **WinForms**. Một số ưu điểm chính của C# bao gồm:
* **Dễ học và dễ sử dụng**: C# có cú pháp thân thiện và dễ hiểu, đặc biệt với các lập trình viên đã quen với các ngôn ngữ hướng đối tượng khác như Java.
* **Tính năng mạnh mẽ**: C# cung cấp đầy đủ các tính năng lập trình hướng đối tượng như kế thừa, đa hình, đóng gói dữ liệu, giúp việc tổ chức mã trở nên dễ dàng và bảo trì hiệu quả.
* **Tích hợp tốt với .NET Framework**: C# được thiết kế để hoạt động tốt với **.NET Framework**, cung cấp hàng ngàn thư viện và công cụ hỗ trợ lập trình viên trong việc phát triển các ứng dụng phức tạp.
* **Windows Forms (WinForms)**: Đây là framework thuộc .NET Framework cho phép phát triển các ứng dụng GUI (giao diện đồ họa người dùng). WinForms cung cấp các thành phần như:
* **TextBox**, **Button**, **Label**, **DataGridView**, v.v. giúp việc tạo giao diện trở nên đơn giản và thân thiện với người dùng.
* WinForms cung cấp khả năng quản lý các sự kiện (event-driven programming), giúp lập trình viên dễ dàng xử lý các thao tác của người dùng, như nhấn nút (button click), nhập liệu (text input), hoặc lựa chọn (combobox selection).
  + 1. Thư viện và công cụ bổ sung
* **MongoDB.Driver**: Đây là thư viện chính để kết nối và thao tác với cơ sở dữ liệu MongoDB trong ứng dụng C#. MongoDB.Driver cung cấp đầy đủ các phương thức cần thiết để thực hiện các thao tác như thêm, sửa, xóa, và truy vấn dữ liệu trong MongoDB.
* **NuGet Package Manager**: Đây là công cụ quản lý các thư viện bên ngoài trong Visual Studio. Thông qua NuGet, lập trình viên có thể dễ dàng cài đặt MongoDB.Driver và các thư viện khác phục vụ cho quá trình phát triển ứng dụng.
  1. Mô tả giao diện

* + 1. Giao diện đăng nhập

Hình 3. 4Giao diện Đăng Nhập

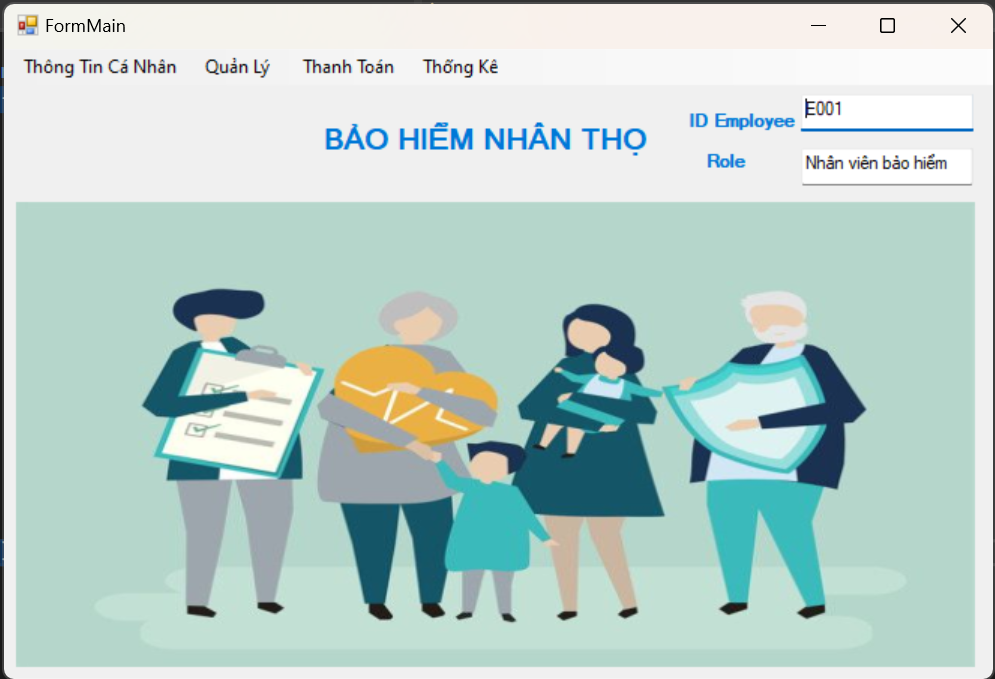
* Các trường nhập liệu:

Username: Trường này dùng để nhập tên đăng nhập của người dùng. Tên đăng nhập thường là một chuỗi ký tự duy nhất dùng để phân biệt từng người dùng với nhau.

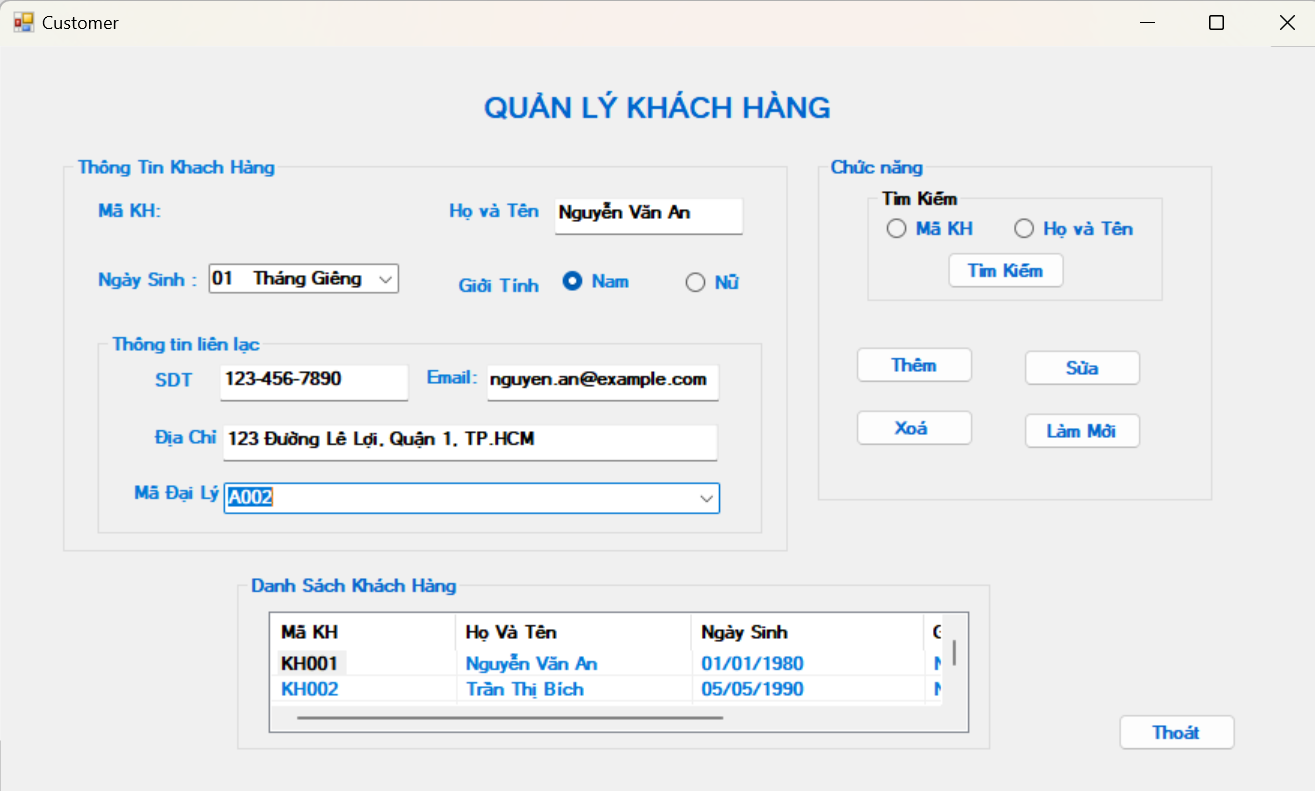
Password: Trường này dùng để nhập mật khẩu. Mật khẩu là một chuỗi ký tự bí mật, dùng để xác thực danh tính của người dùng.

* Nút:

Đăng nhập: Khi người dùng nhập đầy đủ thông tin vào hai trường trên và nhấn vào nút này, hệ thống sẽ tiến hành kiểm tra thông tin đăng nhập. Nếu thông tin chính xác, người dùng sẽ được đăng nhập vào hệ thống.

* + 1. Giao diện form chính
* **FormMain:** Đây là tên của cửa sổ giao diện chính, cho biết đây là form chính của ứng dụng.
* **Thanh Menu:**
* **Thông Tin Cá Nhân:** Có khả năng đây là nơi người dùng có thể xem và cập nhật thông tin cá nhân của mình.
* **Quản Lý:** Có thể là nơi để quản lý các dữ liệu liên quan đến bảo hiểm, chẳng hạn như khách hàng, hợp đồng, sản phẩm.
* **Thanh Toán:** Có thể dùng để quản lý các giao dịch thanh toán liên quan đến bảo hiểm.
* **Thống Kê:** Chức năng này có thể cung cấp các báo cáo, biểu đồ thống kê về hoạt động kinh doanh bảo hiểm.
* **Thông tin người dùng:**
* **ID Employee:** Mã số nhân viên, cho biết người đang đăng nhập vào hệ thống là ai.
* **Role:** Vai trò của người dùng.

Hình 3. 5Giao diện chính

* + 1. Giao diện khách hàng

- Chức năng :

* Load dữ liệu khách hàng lên listview

Yêu cầu : chỉ load 4 field (Mã KH , Họ và Tên , Ngày Sinh , Giới Tính) với điều kiện nhân viên đó status là 'active'.

* Hiện thị thông tin từng khách hàng

Yêu cầu : click chuột vào một row thì hiển thị thông tin khách hàng bao gồm (Mã KH ,Họ và Tên ,Ngày Sinh ,Giới Tính ,SDT , Email ,Địa Chỉ)

Thực hiện :

1. Viết hàm hiển thị thông tin khách hàng bao gồm (Họ và Tên ,Ngày Sinh ,Giới Tính ,SDT , Email ,Địa Chỉ) với tham số truyền vào là Mã KH ở tầng DAO

2. Lấy thông Mã KH ở colums ở cột thứ nhất trong danh sách listview làm tham số truyền vào hàm hiện thị danh sách khách hàng theo mã

* Tìm kiếm theo Mã KH , Họ và Tên

Yêu cầu :

1. Nhấn vào radio Mã KH :( các tất cả các field trừ field Mã KH visible :false) và ngược lại với radio Họ và Tên

2. Khi nhấn button tìm kiếm (radio Mã KH /radio Họ và Tên): hiện thị tìm kiếm lên listview

• radio Mã KH: listview hiện thị theo mã KH đã tìm kiếm

• radio Họ và Tên :listview hiển thị theo Họ và Tên gần đúng đã tìm kiếm

4.Thêm ,Xoá ,Sửa Khách Hàng

Yêu cầu :

- Thêm : với mã khách hàng tự động tăng (2 kí tự đầu KH 3 kí tự sau là số được tăng dần lên người dùng không nhập mã KH)

Ràng buộc : các field không được bỏ trống

Field Họ và Tên :không nhập số

Field Ngày Sinh : phải trên 18 tuổi

Field Email : đúng format email

Field SDT : không nhập chữ ,kí tự đặc biệt ,nhập số (gồm 10 số,…)

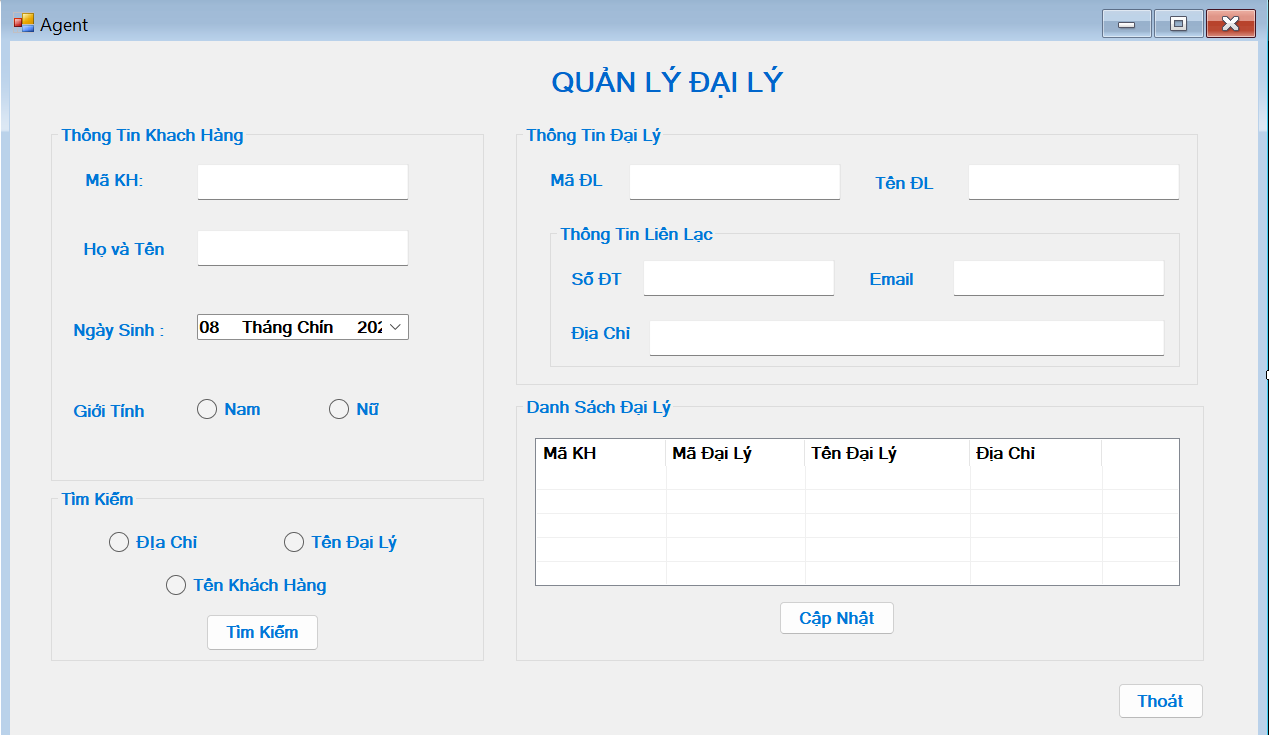
Khi thêm thành công listview được update lại.

Xoá : nghĩa là update field status : 'active' thành “inactive” -> khi xoá listview được update lại

Sửa : không cho phép sửa field Mã KH -> khi sửa listview sẽ được update lại

Làm Mới : xoá dữ liệu ở all field hiện trên UI

Thoát : thoát khỏi giao diện “Quản Lý Khách Hàng”

* + 1. Giao diện đại lý

Hình 3. 6 Giao diện đại lý

* Chức năng
  + - Load dữ liệu đại lý lên listview theo mã khách hàng

Yêu cầu : chỉ load 4 field (Mã KH ,Mã Đại Lý , Tên Đại Lý , Địa Chỉ ) với điều kiện khách hàng đó status là 'active'.

* + - Hiện thị thông đại lý theo từng khách hàng

Yêu cầu : click chuột vào một row thì hiển thị thông tin khách hàng và đại lý tướng ứng

Thực hiện :

1. Viết hàm hiển thị thông tin đại lý (….),hiện thị thêm thông tin khách hàng (Ma KH , Họ và Tên ,Ngày Sinh ,Giới Tính ) theo mã khách hàng

2. Lấy thông Mã KH ở colums ở cột thứ nhất trong danh sách listview làm tham số truyền vào hàm hiện thị danh sách khách hàng theo mã

* + - Cập nhập lại thông tin đại lý của khách hàng cập nhập các field thông tin đại lý trừ cập nhật mã đại lý
    - Tìm kiếm đại lý theo địa chỉ ,tên đại lý , tên khách hàng

Yêu cầu :

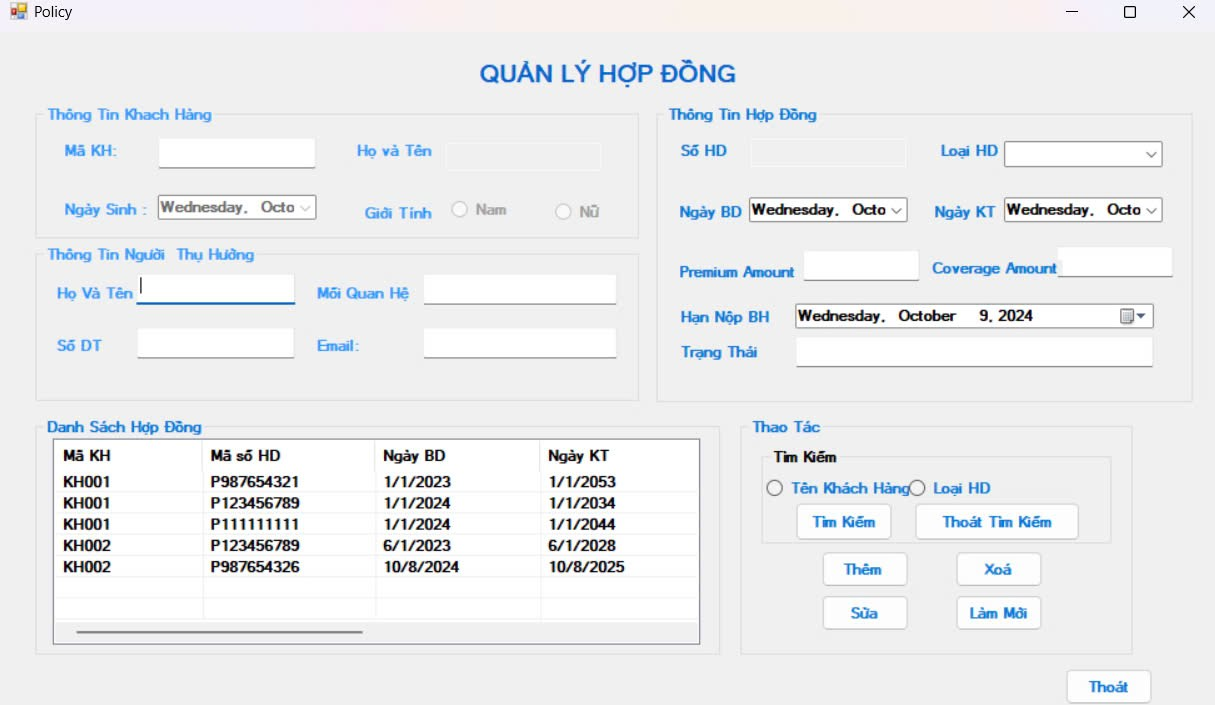
Nhấn vào radio địa chỉ :( các tất cả các field trừ field địa chỉ có visible :false) và ngược lại với radio tên đại lý , tên khách hàng)

Khi nhấn button tìm kiếm (địa chỉ ,tên đại lý , tên khách hàng): hiện thị tìm kiếm lên listview

o Radio địa chỉ : listview hiện thị theo địa chỉ đã tìm kiếm

o radio tên đại lý :listview hiển thị theo tên đại lý tìm kiếm

o radio tên khách hàng : listview hiển thị theo tên khách hàng tìm kiếm.

* + 1. Giao diện hợp đồng
    2. Chức năng
       1. Load dữ liệu hợp đồng lên listview

Yêu cầu : chỉ load 8 field (Mã KH , Mã số HD , NgayBD,Ngay KT ,Premium Amount , Coverage Amount,Trạng Thái) với điều kiện nhân viên đó status là 'active'.

* + - 1. Load loại hoạt đông vào combobox loại hợp đồng

Load vào combobox 3 loại bảo hiểm mặc định : Bảo hiểm nhân thọ trọn đời ,Bảo hiểm nhân thọ có thời hạn ,bảo hiểm nhân thọ hỗn hợp

Ý nghĩa loại bảo hiểm :

1. **Bảo hiểm nhân thọ trọn đời (Whole Life Insurance)**

* Đặc điểm: Cung cấp bảo vệ suốt đời cho người tham gia, với số tiền bảo hiểm sẽ được chi trả khi người được bảo hiểm qua đời, bất kể thời điểm nào.
* Lợi ích: Ngoài bảo vệ, còn có giá trị tích lũy, nghĩa là một phần của phí bảo hiểm được đưa vào quỹ tích lũy và có thể được rút ra hoặc sử dụng khi cần.
* Ưu điểm: Đảm bảo sự bảo vệ suốt đời và giá trị tích lũy có thể tăng dần theo thời gian.
* Nhược điểm: Phí bảo hiểm thường cao hơn so với các loại khác.

1. **Bảo hiểm nhân thọ có thời hạn (Term Life Insurance)**

* Đặc điểm: Cung cấp bảo vệ trong một khoảng thời gian nhất định (ví dụ: 5, 10, 20 năm). Nếu người được bảo hiểm qua đời trong thời gian này, người thụ hưởng sẽ nhận được tiền bảo hiểm.
* Lợi ích: Phí bảo hiểm thấp hơn so với bảo hiểm trọn đời.
* Ưu điểm: Thích hợp cho những người muốn bảo vệ trong một giai đoạn nhất định như khi có trẻ nhỏ hoặc trả nợ.
* Nhược điểm: Không có giá trị tích lũy và khi hết thời hạn, người tham gia phải mua lại hợp đồng mới nếu muốn tiếp tục bảo vệ.

1. **Bảo hiểm nhân thọ hỗn hợp (Endowment Life Insurance)**

* Đặc điểm: Loại bảo hiểm này vừa cung cấp bảo vệ rủi ro, vừa là một hình thức tích lũy tài chính. Người thụ hưởng sẽ nhận được tiền bảo hiểm khi có sự cố hoặc khi hợp đồng đến hạn.
* Lợi ích: Kết hợp bảo vệ và đầu tư. Số tiền sẽ được trả khi người được bảo hiểm qua đời hoặc khi hợp đồng hết hạn, tùy theo điều kiện nào đến trước.
* Ưu điểm: Người tham gia sẽ nhận lại toàn bộ số tiền đã đóng sau một thời gian nhất định.
* Nhược điểm: Phí bảo hiểm cao hơn so với bảo hiểm có thời hạn.
  + - 1. Hiện thị thông tin từng hợp đồng
* Yêu cầu : click chuột vào một row thì hiển thị các thông tin tương ứng trong groupbox ( thông tin khách hàng , thông tin người hưởng thụ , thông tin hợp đồng)
* Thực hiện :

1. Viết hàm hiển thị thông tin khách hàng bao (với các field trong thông tin khách hàng {mã kh , họ tên ,ngày sinh ,giới tính }, thông tin hợp đồng {Số HD , Loại HD ,Ngày BD ,Ngày KT , Premium Amount , Coverage Amount, next\_payment\_date,Trạng Thái} thông tin người hưởng thụ {họ và tên , mối quan hệ ,phone , email} ) với tham số truyền vào là Mã KH ở tầng DAO
2. Lấy thông Mã KH ở colums ở cột thứ nhất trong danh sách listview làm tham số truyền vào hàm hiện thị danh sách khách hàng theo Mã KH
   * + 1. Tìm kiếm theo tên khách hàng, Loại HD

Yêu cầu :

1. Nhấn vào radio tên KH :( các tất cả các field trừ field tênKH visible :false) và ngược lại với radio Loại HD
2. Khi nhấn button tìm kiếm (radio tên KH /Loại HD): hiện thị tìm kiếm lên listview

* radio tên KH: listview hiện thị theo tên KH đã tìm kiếm
* radio loai HD :listview hiển thị theo loai HD tìm kiếm
  + - 1. Thêm ,Xoá ,Sửa Hợp Đồng

Yêu cầu :

* **Thêm** : Với mã HD tự động tăng (2 kí tự đầu HD 3 kí tự sau là số được tăng dần lên người dùng không nhập mã HD : vd:HD001)

Ràng buộc : các field không được bỏ trống

* Field Ngày BD :khi load hiện thị ngày hiện tại
* Field Ngày KT: phải sau ngày BD ( có thể sau khoảng 3 tháng)
* Field "premium\_amount":không được nhập chữ , không âm
* Field "coverage\_amount": không được nhập chữ , không âm và phải lớn hơn premium amount
* Field Trạng Thái : không cần nhập dữ liệu khi thêm mặc định "status": "Đang hoạt động"
* Field Họ Tên người hưởng thụ: chỉ cho nhập chữ
* Field mối quan hệ của người hưởng thụ : chỉ cho nhập chữ
* Field email người hưởng thụ: đúng format email thông thường
* Field SDT người hưởng thụ : không nhập chữ ,kí tự đặc biệt ,nhập số (gồm 10 số,…)
* Field next\_payment\_date: lớn hơn ngày bắt đầu làm hợp đồng và nhỏ hơn ngày hết hạn hợp đồng
* Thực hiện:

B1: Người dùng nhập MaKH ở textbox nhấn enter thì các thông tin liên quan đến khách hàng sẽ hiện thị tương ứng lên các trường liên quan ( Họ và Tên ,Ngày Sinh ,Giới Tính)

B2: Nhập các thông tin liên quan về thông tin hợp đồng , thông tin người hưởng thụ

B3 : Nhấn nút thêm listview sẽ update hiển thị lại danh sach

* Khi thêm thành công listview được update lại.
* Xoá : xóa hợp đồng đã được click chọn trên listview nghĩa là xóa khỏi document của đã chọn theo mã hợp đồng ra khỏi field Hợp Đồng
* Sửa : không cho phép sửa field Mã HD -> khi sửa listview sẽ được update lại
* Làm Mới : xoá dữ liệu ở all field hiện trên UI
* Thoát : thoát khỏi giao diện “Quản Lý Hợp Đồng”

* + 1. Giao diện yêu cầu bồi thường

Hình 3. 7 Giao diện Yêu Cầu Bồi Thường

* Chức năng
* Load dữ liệu yêu cầu bồi thường lên listview

Yêu cầu : chỉ load 8 field (Mã KH , Mã số HD,Mã số BT , Ngày Bồi thường , Loại Hợp Đồng , Clamin Amount{số tiền yêu cầu bồi thường của hợp đồng}) với điều kiện nhân viên đó status là 'active'.

* Hiện dữ liệu lên các field tương ứng

Yêu cầu : click chuột vào một row thì hiển thị các thông tin tương ứng trong groupbox ( thông tin khách hàng , thông tin hợp đồng , thông tin yêu cầu bồi thường )

Thực hiện :

1. Viết hàm hiển thị thông tin khách hàng bao (với các field trong thông tin khách hàng {mã kh , họ tên }, thông tin hợp đồng {Mã HD , Loại HD }, thông tin yêu cầu bồi thường {Mã số BT ,Ngày BD , Clamin Amount (số tiền bồi thường ,trạng thái , và các giấy tờ liên quan }) với tham số truyền vào là Mã KH ở tầng DAO

2. Lấy thông Mã KH ở colums ở cột thứ nhất trong danh sách listview làm tham số truyền vào hàm hiện thị danh sách khách hàng theo Mã KH

* Tìm kiếm theo Mã BTHD , Loại HD ,Status: "Chờ Thanh Toán"/”Hoàn tất/Đang Xử Lý”

Yêu cầu :

1. Nhấn vào radio Mã BTHD :( các tất cả các field trừ field Mã BTHD có visible :false) và ngược lại với radio Loại HD , Status

2. Khi nhấn button tìm kiếm (radio Mã BTHD /Loại HD/Status ): hiện thị tìm kiếm lên listview

o radio Mã BTHD: listview hiện thị theo mã BTHD đã tìm kiếm

o radio Loai HD :listview hiển thị theo loai HD tìm kiếm

o radio Status : listview hiển thị theo trạng thái đã chọn (“Chờ xử lý “/”Hoàn tất”)

* Thêm ,Xoá ,Sửa yêu cầu bồi thường

Yêu cầu :

- Thêm : Với mã yêu cầu bồi thường tự động tăng (2 kí tự đầu BT 3 kí tự sau là số được tăng dần lên người dùng không nhập mã HD : vd:BT001)

Ràng buộc : các field không được bỏ trống

• Field Ngày BT :khi load hiện thị ngày hiện tại

• Field "clamin\_amount":không được nhập chữ , không âm và phải nhỏ hơn hoặc bằng coverage\_amount đã quy định hợp động trước đó .( viết một hàm yêu cầu so sánh clamin\_amount trong file clamin với lại coverage\_amount trong field policies.

• Field Trạng Thái : không cần nhập dữ liệu khi thêm mặc định "status": "Chờ Thanh Toán"

• Field giấy chứng tử : load ảnh

• Field bản sao hợp đồng : load ảnh

• Field giấy chứng minh quan hệ :load ảnh

- Thực hiện:

B1: Người dùng nhập MaHD cần yêu cầu bồi thường ở textbox MaHD nhấn enter thì các thông tin liên quan đến khách hàng sẽ hiện thị tương ứng lên các trường liên quan ( MaKH ,Họ và tên , loại hợp đồng )

B2: Nhập các thông tin liên quan về thông tin yêu cầu bồi thường

B3 : Nhấn nút thêm listview sẽ update hiển thị lại danh sách

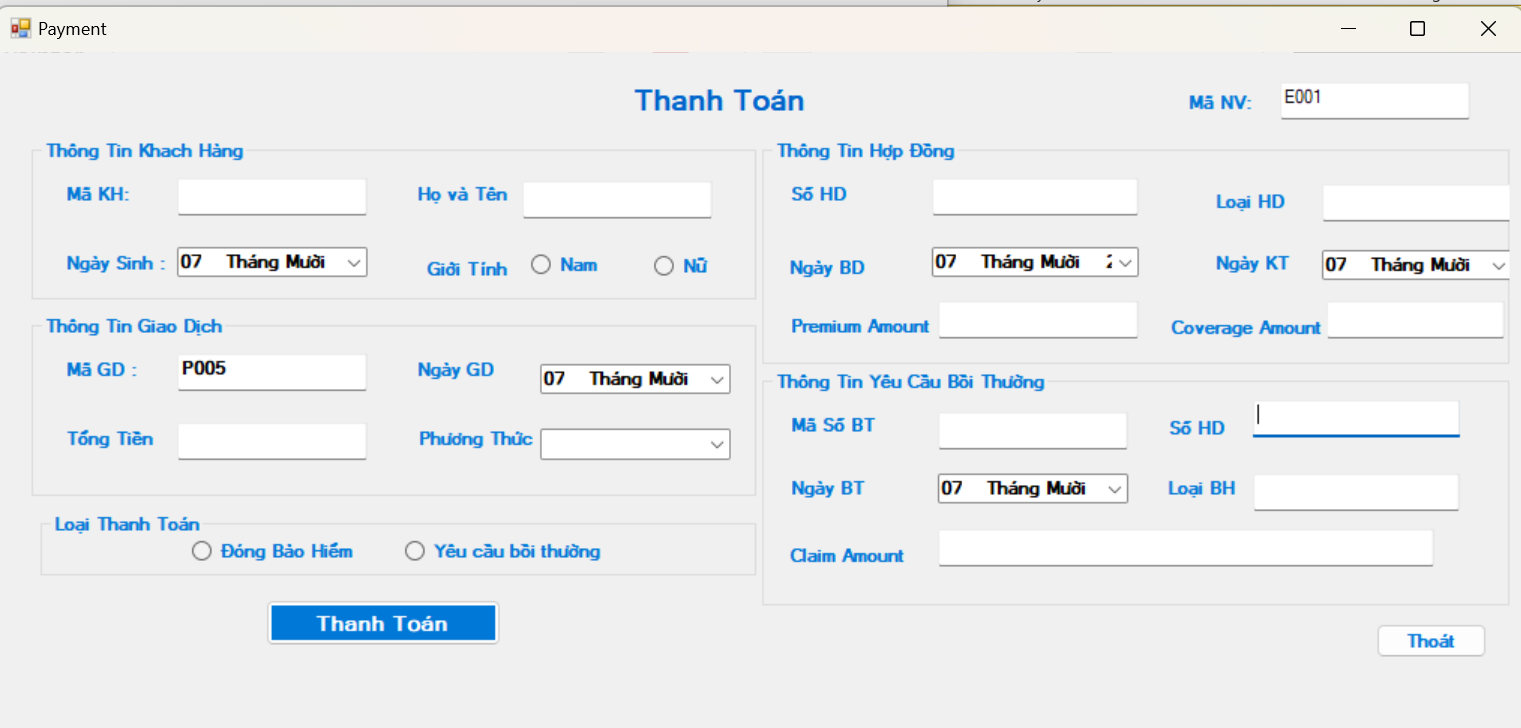
- Khi thêm thành công listview được update lại.

- Xoá : xóa yêu cầu hợp đồng đã được click chọn trên listview nghĩa là xóa khỏi document của đã chọn theo mã yêu cầu hợp đồng ra khỏi field Clamin

- Sửa : không cho phép sửa field Mã BT-> khi sửa listview sẽ được update lại

- Làm Mới : xoá dữ liệu ở all field hiện trên UI

- Thoát : thoát khỏi giao diện “Quản Lý yêu cầu bồi thường”

* + 1. Giao diện thanh toán

Hình 3. 8 Giao diện thanh toán

* Chức năng :Giao diện thanh toán với 2 mục đích chính:
* **Thanh toán hợp đồng bảo hiểm:** Người dùng có thể thực hiện việc thanh toán phí bảo hiểm hàng tháng, hàng năm hoặc các khoản thanh toán khác liên quan đến hợp đồng bảo hiểm của mình.
* **Yêu cầu bồi thường:** Trong trường hợp xảy ra sự kiện bảo hiểm, người dùng có thể sử dụng giao diện này để gửi yêu cầu bồi thường đến công ty bảo hiểm.
* Cách thực hiện

Bước 1. Chọn loại giao dịch:

Người dùng chọn một trong hai tùy chọn:

* Đóng bảo hiểm: Để thực hiện việc thanh toán phí bảo hiểm định kỳ.
* Yêu cầu bồi thường: Để thanh toán các khoản phí liên quan đến việc yêu cầu bồi thường.

Bước 2. Nhập thông tin:

Mã hợp đồng/Mã yêu cầu bồi thường:

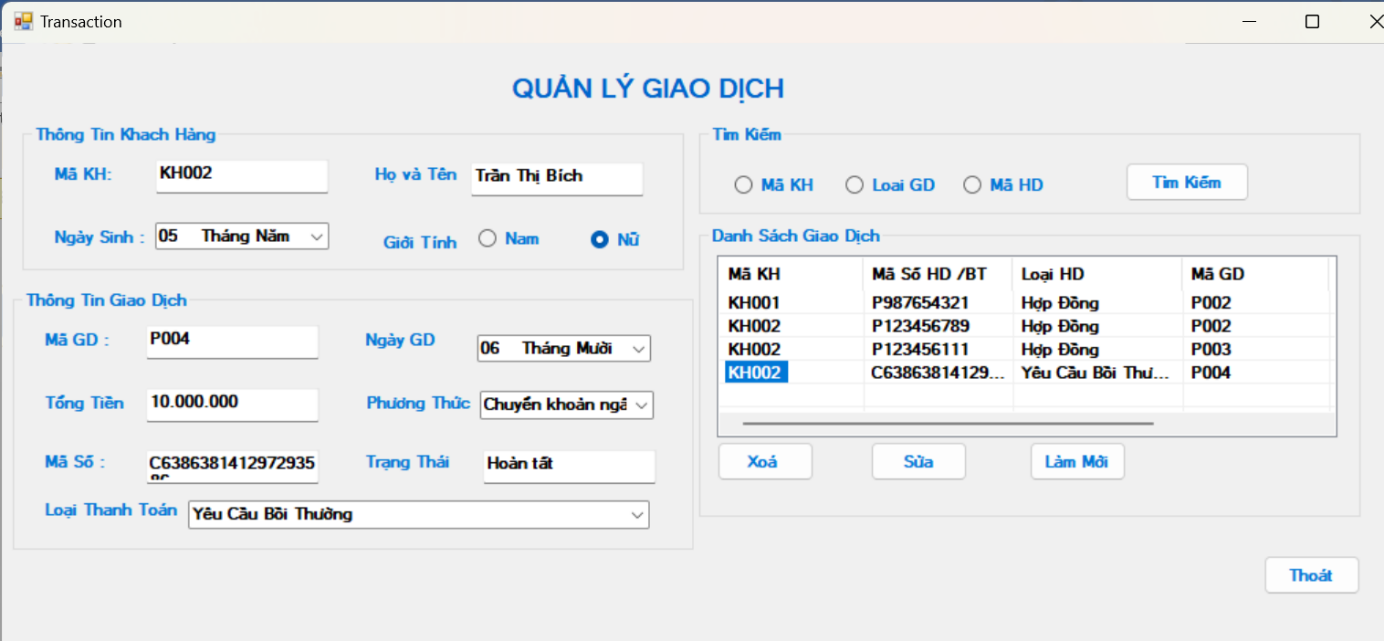
* Người dùng nhập mã hợp đồng hoặc mã yêu cầu bồi thường tương ứng với lựa chọn ở bước 1.
* Hệ thống sẽ tự động kiểm tra tính hợp lệ của mã này trong cơ sở dữ liệu.
* Các trường thông tin khác:
* Nếu chọn "Đóng bảo hiểm": Hệ thống sẽ tự động điền các thông tin liên quan đến hợp đồng như: họ tên khách hàng, loại hợp đồng, ngày bắt đầu, ngày kết thúc, số tiền đóng, số tiền được hưởng khi bồi thường.
* Nếu chọn "Yêu cầu bồi thường": Hệ thống sẽ tự động điền các thông tin liên quan đến yêu cầu bồi thường như: số hợp đồng, loại hợp đồng, ngày lập yêu cầu, số tiền yêu cầu bồi thường.

Bước 3. Chọn phương thức thanh toán:

* Người dùng chọn một trong các phương thức thanh toán có sẵn (ví dụ: tiền mặt, chuyển khoản).

Bước 4. Xác nhận và hoàn tất:

* Người dùng nhấn nút "Thanh toán".
* Hệ thống:Kiểm tra lại tất cả thông tin đã nhập.
* Cập nhật dữ liệu:
* Nếu là thanh toán bảo hiểm: Cập nhật ngày đóng phí tiếp theo trong bảng hợp đồng (policies).
* Nếu là thanh toán yêu cầu bồi thường: Cập nhật trạng thái yêu cầu bồi thường từ "Chờ xử lý" thành "Hoàn tất" trong bảng yêu cầu bồi thường (claims).
* Thông báo: Hiển thị thông báo xác nhận giao dịch thành công.
  + 1. Giao diện giao dịch



Hình 3. 9 Giao diện quản lý giao dịch

- Chức năng : giao diện quản lý giao dịch bao gồm các chức năng

* + Xem chi tiết một thông tin giao diện: Chức năng này cho phép người dùng xem chi tiết một giao dịch cụ thể khi chọn một dòng trong danh sách giao dịch.
  + Xóa một giao dịch : Cho phép người dùng xóa một giao dịch khỏi hệ thống.
  + Chỉnh sửa một giao dịch : Cho phép người dùng chỉnh sửa thông tin của một giao dịch hiện có.
  + Tìm kiếm danh sách giao dịch: Cho phép người dùng tìm kiếm và lọc danh sách giao dịch theo các tiêu chí như mã khách hàng, loại giao dịch, hoặc mã hợp đồng.
  + Làm mới : xóa sạch dữ liệu trên hệ thống
  + Thoát : đóng của sổ quản lý giao dịch

- Cách thực hiện :

* + Xem chi tiết một thông tin giao diện :
* Hiện thông tin lên listview mã KH , mã số hợp đồng / mã số yêu cầu yêu thường ,loại hợp đồng , mã giao dịch
* Khi clik vào một hàng trong list view sẽ lấy dữ liệu ở cột thứ nhất là mã khách hàng và cột thứ 4 là mã giao dịch làm tham số để thực hiện hiện thị các thông tin liên quan đến khách hàng ( mã khách hàng , họ tên , ngày sinh , giới tính) và thông tin giao dịch ( mã giao dịch , ngày giao dịch , tổng tiền , phương thức thanh toán , mã số hợp đồng/ yêu cầu bồi thường, trạng thái , loai thanh toán hợp đồng / yêu cầu bồi thường .
  + Xóa một giao dịch
* Clik vào giao dịch cần xóa trên list view
* Sau khi click list view ác thông tin sẽ hiển thị lên các trường tương ứng khi đó lấy thông tin mã khách hàng , mã giao dịch làm tham sô điều kiện để thực hiện xóa
* Khi thực hiện xóa thì giao dịch có trạng thái chuyển sang là “Đã Hủy”
* Dữ liệu trên list view sẽ được load lại khi xóa
  + Sửa một giao dịch
* Click chọn một giao dịch Clik vào giao dịch cần xóa trên list view
* Sau khi click list view ác thông tin sẽ hiển thị lên các trường tương ứng khi đó lấy thông tin mã khách hàng , mã giao dịch làm tham sô điều kiện để thực hiện sửa không sửa mã giao dịch ,mã số hợp đồng
* Khi thực hiện sửa thì các trường được sửa sẽ được load lại trên list view
  + Tìm kiếm
* Lựa chọn các radio tìm kiếm (bao gồm mã KH , loại KH , mã HD )
* Trường hợp chọn mã KH : thì có trường khác ngoại trừ trường mã khách hàng đều vô hiệu hóa
* Trường hợp chọn loại HD: thì có trường khác ngoại trừ trường loại khách hàn đều vô hiệu hóa
* Trường hợp chọn mã HD : thì có trường khác ngoại trừ trường loại hợp đồng đều vô hiệu hóa
* Nhấn button tìm kiếm : khi đó hệ thống sẽ lọc lại danh sach giao dịch theo trường muốn tìm kiếm

1. KẾT LUẬN
   1. Kết quả đạt được

Sau khi hoàn thành đề tài **Quản lý khách hàng tham gia bảo hiểm nhân thọ**, tôi đã triển khai được các chức năng chính, bao gồm:

* **Chức năng quản lý khách hàng**: Hệ thống cho phép quản lý đầy đủ thông tin của khách hàng tham gia bảo hiểm như họ tên, địa chỉ, số điện thoại, và các thông tin liên quan khác. Mỗi khách hàng được gắn mã định danh duy nhất, giúp việc tìm kiếm và theo dõi trở nên dễ dàng hơn.
* **Chức năng quản lý hợp đồng**: Hệ thống quản lý chi tiết các hợp đồng bảo hiểm của khách hàng, bao gồm loại hợp đồng, giá trị bảo hiểm, thời hạn hợp đồng, và phí bảo hiểm. Các hợp đồng được liên kết trực tiếp với khách hàng, đảm bảo tính chính xác và dễ quản lý.
* **Chức năng quản lý yêu cầu bồi thường**: Hệ thống cung cấp khả năng tiếp nhận, theo dõi và xử lý các yêu cầu bồi thường từ khách hàng. Các yêu cầu bồi thường được phân loại theo trạng thái (chờ xử lý, đã thanh toán, từ chối), giúp doanh nghiệp dễ dàng quản lý quy trình bồi thường.
* **Chức năng thanh toán và quản lý giao dịch**: Hệ thống hỗ trợ các hình thức thanh toán cho các hợp đồng bảo hiểm và yêu cầu bồi thường. Người dùng có thể theo dõi trạng thái của từng giao dịch, bao gồm chi tiết về số tiền, phương thức thanh toán và ngày giao dịch, đảm bảo quá trình thanh toán được minh bạch và an toàn.
* **Chức năng quản lý đại lý**: Hệ thống cung cấp chức năng quản lý đại lý bảo hiểm, cho phép theo dõi thông tin các đại lý và các hợp đồng được ký kết thông qua đại lý. Điều này giúp tăng cường hiệu quả quản lý mạng lưới đại lý và tạo sự minh bạch trong hoạt động kinh doanh.

Những chức năng này đã giúp hệ thống trở nên toàn diện, dễ sử dụng và hiệu quả trong việc quản lý khách hàng, hợp đồng, và các giao dịch bảo hiểm.

* 1. Những hạn chế , thiếu sót ,khó khăn

Trong quá trình tìm hiểu và thực hiện đề tài, một số hạn chế và khó khăn đã gặp phải như sau:

* **Chưa tối ưu hóa hoàn toàn hiệu suất hệ thống**: Do hệ thống ban đầu được thiết kế trên nền tảng **Windows Forms**, với dữ liệu được lưu trữ trên **MongoDB**, có thể gặp vấn đề về hiệu suất khi dữ liệu lớn hoặc số lượng truy vấn tăng cao. Việc xử lý một lượng lớn dữ liệu yêu cầu tối ưu hóa hơn về mặt truy vấn và kiến trúc lưu trữ.
* **Khả năng mở rộng**: Hệ thống hiện tại được thiết kế theo mô hình cơ bản với chức năng chính về quản lý khách hàng, hợp đồng và yêu cầu bồi thường. Tuy nhiên, hệ thống chưa tích hợp được các tính năng nâng cao như phân tích dữ liệu, báo cáo chi tiết hoặc tích hợp với các hệ thống bảo hiểm khác, khiến khả năng mở rộng của hệ thống còn hạn chế.
* **Hỗ trợ cho nhiều nền tảng**: Hệ thống hiện tại chỉ chạy trên nền tảng **Windows**, và không hỗ trợ cho các nền tảng khác như web hoặc di động. Điều này làm hạn chế tính linh hoạt và khả năng truy cập của hệ thống.
* **Khó khăn về kỹ thuật và công nghệ**: Trong quá trình thực hiện, việc tìm hiểu về cách kết nối và thao tác với cơ sở dữ liệu **MongoDB** từ **C#** gặp nhiều khó khăn do sự khác biệt về cách tiếp cận giữa NoSQL và cơ sở dữ liệu quan hệ (SQL). Ngoài ra, việc cài đặt và quản lý các thư viện cần thiết đôi khi gặp khó khăn khi không tương thích hoàn toàn với môi trường phát triển hoặc công cụ IDE.
  1. Hướng phát triển của đề tài.

Dựa trên những kết quả đạt được và hạn chế hiện tại, một số hướng phát triển cho đề tài trong tương lai bao gồm:

* **Tối ưu hóa hiệu suất hệ thống**: Cải thiện và tối ưu hóa các truy vấn tới **MongoDB** để đảm bảo hệ thống có thể xử lý dữ liệu nhanh chóng và hiệu quả ngay cả khi số lượng khách hàng và hợp đồng tăng cao. Bên cạnh đó, có thể áp dụng các phương pháp lưu trữ và truy xuất dữ liệu tiên tiến hơn như caching để cải thiện thời gian phản hồi của hệ thống.
* **Phát triển phiên bản web và di động**: Xây dựng thêm các phiên bản hệ thống trên nền tảng web hoặc di động để người dùng có thể dễ dàng truy cập và quản lý thông tin từ xa, không bị phụ thuộc vào hệ điều hành Windows. Điều này giúp hệ thống linh hoạt hơn và phục vụ nhiều đối tượng người dùng hơn.
* **Tích hợp báo cáo và phân tích**: Xây dựng thêm các chức năng báo cáo tự động và phân tích dữ liệu giúp doanh nghiệp có cái nhìn tổng quan về hiệu suất hoạt động, đánh giá tình hình kinh doanh, dự báo xu hướng và hỗ trợ ra quyết định.
* **Mở rộng tính năng bảo mật**: Tăng cường bảo mật dữ liệu khách hàng và các giao dịch tài chính, đặc biệt là trong quá trình thanh toán. Điều này có thể bao gồm mã hóa dữ liệu, tích hợp xác thực hai yếu tố (2FA), và các phương pháp bảo mật khác nhằm đảm bảo an toàn cho hệ thống.
* **Tích hợp AI và Machine Learning**: Áp dụng công nghệ trí tuệ nhân tạo (AI) và học máy (Machine Learning) để tự động hóa quy trình xử lý yêu cầu bồi thường, dự đoán xu hướng nhu cầu bảo hiểm của khách hàng và tối ưu hóa dịch vụ khách hàng. Điều này sẽ giúp tăng tính cạnh tranh và hiệu quả của hệ thống quản lý bảo hiểm.

Kết quả của đề tài đã xây dựng một hệ thống cơ bản với các chức năng quản lý bảo hiểm hiệu quả, tuy nhiên, để đáp ứng nhu cầu thực tế của doanh nghiệp và người dùng, việc tiếp tục phát triển và mở rộng hệ thống là cần thiết.

# DANH MỤC TÀI LIỆU THAM KHẢO

* 1. Prudential Vietnam. "Những điều nhất định phải biết trước khi mua bảo hiểm nhân thọ." Prudential Vietnam, <https://www.prudential.com.vn/vi/blog-nhip-song-khoe/nhung-dieu-nhat-dinh-phai-biet-truoc-khi-mua-bao-hiem-nhan-tho/> Ngày truy cập 01/10/2024.
  2. Bảo hiểm nhân thọ, cách thức hoạt động và lợi ích của nó đối với người tham gia, <https://www.baovietnhantho.com.vn/goc-chuyen-gia/bao-hiem-nhan-tho-la-gi> , Ngày truy cập : 03/10/2024