Shape, square

Description automatically generated

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**Báo cáo bộ môn**

**HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**ĐỀ TÀI:**

Logo

Description automatically generated**HỆ THỐNG PHÂN PHỐI GAME**

**Giáo viên hướng dẫn:**

GVCC, TS. NGUYỄN THÀNH SƠN

**Sinh viên thực hiện:**

**NHÓM 7**

|  |  |
| --- | --- |
| Huỳnh Nhật Minh | **MSSV:20133066** |
| Đỗ Hoàng Thịnh | **MSSV:20133122** |
| Ngô Hoàng Khánh Duy | **MSSV:20133030** |
| Nguyễn Phạm Ngọc Duy | **MSSV:20133031** |

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

**Ký tên**

MỤC LỤC

[**MỤC LỤC 3**](#_heading=h.1fob9te)

[**1. Tổng quan về phần mềm 4**](#_heading=h.3znysh7)

[**1.1. Mục tiêu** 4](#_heading=h.2et92p0)

[**1.2. Đối tượng sử dụng** 4](#_heading=h.tyjcwt)

[**1.3. Mô tả** 4](#_heading=h.3dy6vkm)

[**1.4. Thông tin phần mềm** 4](#_heading=h.1t3h5sf)

[**1.5. Chức năng của phần mềm** 4](#_heading=h.4d34og8)

[**2. Lược đồ quan hệ 4**](#_heading=h.17dp8vu)

[**2.1. Mô hình ERD** 4](#_heading=h.3rdcrjn)

[***2.1.1.* Lược đồ** 4](#_heading=h.26in1rg)

[***2.1.2.* Quan hệ, giải nghĩa và các ràng buộc có trong cơ sở dữ liệu** 5](#_heading=h.lnxbz9)

[***2.1.3.* Cài đặt cơ sở dữ liệu** 7](#_heading=h.35nkun2)

[**3. Danh sách các Views, Stored procedures, Functions, Triggers 8**](#_heading=h.44sinio)

[**3.1. Views** 8](#_heading=h.2jxsxqh)

[**3.2. Stored procedures** 8](#_heading=h.z337ya)

[**3.3. Functions** 9](#_heading=h.3j2qqm3)

[**3.4. Triggers** 9](#_heading=h.1y810tw)

[**4. Tạo user, phân quyền 12**](#_heading=h.4i7ojhp)

[**5. Hướng dẫn sử dụng 12**](#_heading=h.2xcytpi)

[**1.1. Cài đặt chương trình** 12](#_heading=h.1ci93xb)

[**1.2. Chi tiết** 12](#_heading=h.3whwml4)

[***1.2.1. Khởi động ứng dụng (Guest)*** *12*](#_heading=h.2bn6wsx)

[***1.2.2. Đăng nhập*** *12*](#_heading=h.qsh70q)

[**6. Kết luận 12**](#_heading=h.3as4poj)

[**1.3. Các phần đã đạt được trong đồ án** 12](#_heading=h.1pxezwc)

[**1.4. Các phần đặc sắc, đặc biệt trong đồ án** 12](#_heading=h.49x2ik5)

[**1.5. Một số hạn chế** 12](#_heading=h.2p2csry)

[**1.6. Cam kết** 12](#_heading=h.147n2zr)

1. **Tổng quan về phần mềm**
   1. **Mục tiêu**

Hiện nay việc mua bán trao đổi game trên thị trường hay không gian mạng ngày càng phổ biến, việc lưu trữ thông thường cũng như việc sử dụng các phương thức thủ công để quản lí thông tin người dùng ngày càng khó khăn trong việc quản lí và cập nhật kịp thời khiến thông tin mua bán và quản lí người dùng trở nên khó khăn.

Để hổ trợ cho việc quán lý mua bán game trở nên dễ dàng, nhanh chóng và hiệu quả, đảm bảo thông tin luôn chính xác và tức thời. Để thuận tiện cho việc này việc số hoá và tạo ra một hệ thống quản lí lượng dữ liệu sinh ra là rất cần thiết, bởi hệ thống quản lí không chỉ để lưu trữ dữ liệu mà nó còn hổ trợ cho người dùng dễ dàng sử dụng và thao tác với dữ liệu một cách hiệu quả nhất.

* 1. **Đối tượng sử dụng**

Đề tài được áp dụng trong Hệ thống phân phối game đến người dùng. Các chức năng mà phần mềm tạo ra dùng cho 2 đối tượng là người quản lí hệ thống và người mua game.

* 1. **Mô tả**

Phần mềm quản lý hệ thống bán game là phần mềm dùng cho việc quản lý thông tin mua bán game trên hệ thống với 2 nhóm người dùng là người mua và người quản lý với 2 loại tài khoản phân biệt.

Với tài khoản người dùng, phần mềm cung cấp cho người mua công cụ hổ trợ cho việc tìm kiếm và mua game trên hệ thống cùng với đó là cung cấp công cụ để người mua quản lý game mà họ đã sở hữu trên hệ thống.

Với tài khoản của người quản lí, phần mềm cung cấp cho người quản lí công cụ để thêm xoá sửa game cùng với những thông tin bên lề như studio game, tag game,... cùng với đó là công cụ hiển thị thông tin doanh số của hệ thống.

* 1. **Thông tin phần mềm**

*Tên phần mềm:*Dormitory Management **–** Phần mềm Quản lí Ký túc xá.

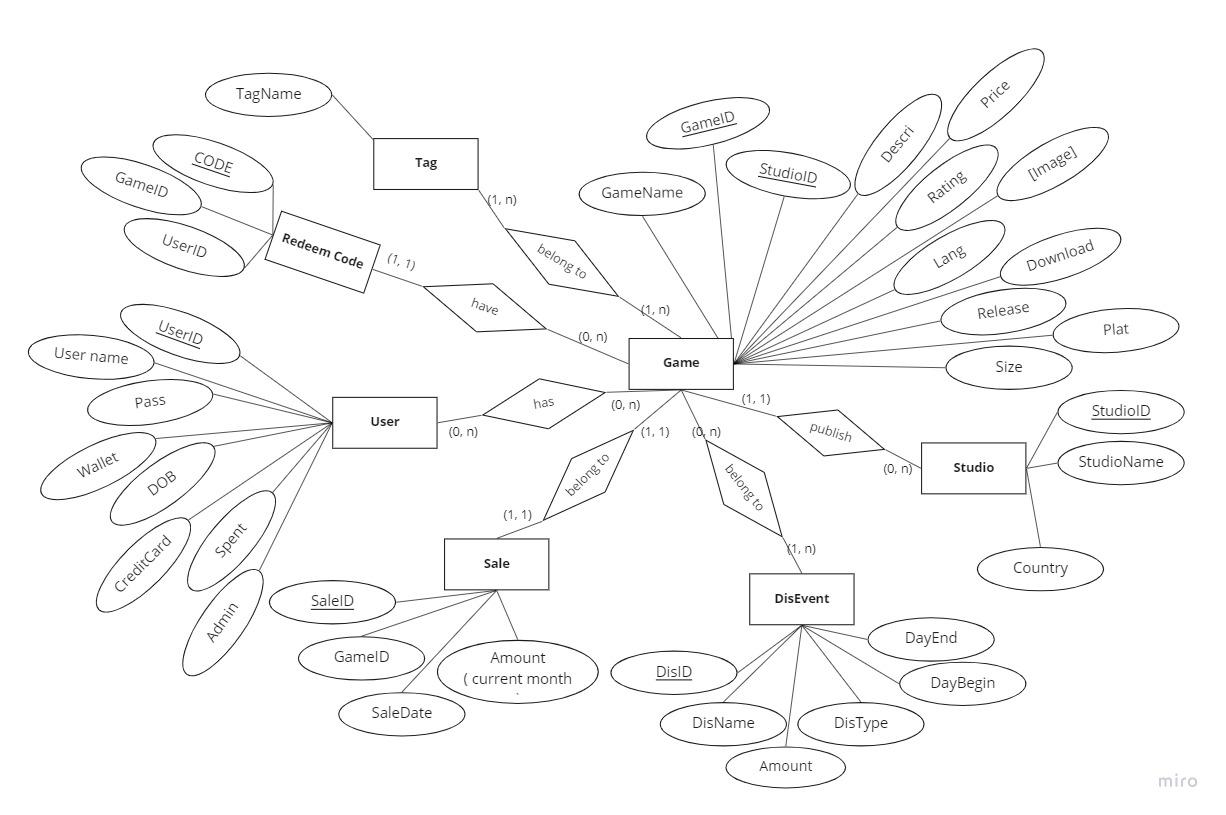
*Ngôn ngữ lập trình:* C# (Window Form).

*Công nghệ sử dụng:* ADO.NET

*Phần mềm hỗ trợ:* SQL Server.

* 1. **Chức năng của phần mềm**
* Thêm, sửa, xóa các dịch vụ, thông tin hóa đơn và phiếu đăng ký phòng.
* Xem thông tin về các dịch vụ, loại phòng của kí túc xá.
* Thống kê mức sử dụng dịch vụ theo tháng, theo năm.
* Phân quyền:
  + Quản lí: có đầy đủ tất cả các quyền đã nêu ở trên.
  + Nhân viên: Có đầy đủ các quyền trên, ngoại trừ việc thêm, xóa, sửa nhân viên.
  + Người dùng: xem thông tin cá nhân, thông tin hóa đơn.
  + Khách truy cập: xem thông tin về dịch vụ ký túc xá.

1. **Lược đồ quan hệ**
   1. **Mô hình ERD**
      1. **Lược đồ**



* + 1. **Quan hệ, giải nghĩa và các ràng buộc có trong cơ sở dữ liệu**

Chương trình sẽ được thiết kế sử dụng 7 quan hệ trong cơ sở dữ liệu để lưu trữ và quản lý, truy cập các thông tin cần thiết cho một hệ thống phân phối game:

* **User**(UserID, Username, Pass, Wallet, DOB, CreditCard, Spent,[Admin]): Quan hệ này lưu trữ thông tin về mỗi người dùng với ID là khoá chính, một User có thể có nhiều Game lưu trữ trong bảng Library.
  + Username [nvarchar]: Tên tài khoản.
  + Password [nvarchar]: Mật khẩu.
  + Wallet[int]: Wallet là ví tiền của người dùng có thể nạp vào để thanh toán dịch vụ game.
  + DOB[date]: Ngày tháng năm sinh của người dùng.
  + CreditCard [char]: mã thẻ của User
  + Spent [int]: Spent hiển thị số tiền mà người dùng đã bỏ ra.
  + [Admin]: Dùng để định danh phân quyền, mặc đinh bằng 0 là user còn 1 là Admin.
* **GAME** (GameID, StudioID, GameName,Descri, Price, Rating, [Image], Lang, Download, Release, Plat, Size): Quan hệ này lưu trữ những thuộc tính về game với ID là khóa chính, thuộc tính StudioID là khóa ngoại được tham chiếu tới ID trong bảng Studio
  + GameName[nvarchar]: Tên game.
  + Desc[nvarchar]: Lưu trữ thông tin giới thiệu của game.
  + Price[int]: Giá tiền của game.
  + Rating[float]: Đánh giá của người dùng dành cho game.
  + Lang,Platform[nvarchar]: Ngôn ngữ, nền tảng hổ trợ game.
  + Picture[byte]: Các hình ảnh giới thiệu game
  + Download,Size[int]: Số lượt tải, dung lượng của game
  + StudioID [int]: Là khóa ngoại tham chiếu tới thuộc tính ID trong bảng Studio
* **Lib** (UID, GID): Quan hệ này để lưu trữ những game mà người dùng đã mua. Quan hệ này có các thuộc tính:
  + GID [int]: Khóa ngoại tham chiếu tới ID trong bảng GAME
  + UID [int]: Khóa ngoại tham chiếu tới ID trong bảng User
* **Redeem\_Code**(Code, GameID, UserID): Quan hệ này lưu trữ thông tin về những code để kích hoạt game với Code là khoá chính, thuộc tính GameID này được tham chiếu tới thuộc tính GameID trong bảng GAME.
  + Code[NVARCHAR]: mã game
  + GameID[int]: khoá ngoại tham chiếu tới khoá chính của bảng Game.
  + UserID[int]: ID tài khoản người sở hữu
* **Sale** (SaleID, GameID, SaleDate, Amount): Quan hệ này lưu trữ thông tin về doanh của game với ID là khoá chính, thuộc tính GameID này được tham chiếu tới thuộc tính ID trong bảng GAME.
  + SaleDate[date]: Mốc thời gian cập nhật doanh thu gần nhất theo tháng.
  + Amount[int]: Doanh thu theo mốc thời gian của biến Date.
  + GameID[int]: khoá ngoại tham chiếu tới khoá chính của bảng Game.
* **Dis\_Event**( DisID, DisName, Amount, DisType, DayBegin, DayEnd): Quan hệ này lưu trữ thông tin về những game đang trong sự kiện giảm giá định kì với ID là khoá chính.
  + DisName[int]: Tên sự kiện.
  + Amount[int]: tỉ lệ giảm giá tính theo %
  + DisType[NVARCHAR]: Theo thể loại game nào.
  + DayBegin[date]: Ngày bắt đầu.
  + DayEnd[date]: Ngày kết thúc
* **DisGame** (GameID, DisID): Quan hệ này để lưu trữ những Game đang trong event giảm giá. Quan hệ này có các thuộc tính:
  + GameID [nvarchar]: Khóa ngoại tham chiếu tới ID trong bảng GAME
  + DisID [int]: Khóa ngoại tham chiếu tới ID trong bảng Discount\_Event
* **Studio** (StudioID, StudioName, Country ): Quan hệ này lưu trữ thông tin về những thông tin của nhà phát hành game với StudioID là khoá chính, thuộc tính StudioID này được tham chiếu tới thuộc tính StudioID trong bảng GAME.
  + StudioName, Country [NVARCHAR]: Tên nhà phát hành game, quốc gia.
* **Tag** (TagName): Quan hệ này lưu trữ thông tin về thể loại của game với TagName là khoá chính.
  + Name[NVARCHAR]: thể loại game.
* **Category** (TagName, GameID): Quan hệ này để lưu trữ những thể loại tương ứng với game. Quan hệ này có các thuộc tính:
  + TagName [nvarchar]: Khóa ngoại tham chiếu tới TagName trong bảng Tag
  + GameID [nvarchar]: Khóa ngoại tham chiếu tới GameID trong bảng GAME
    1. **Cài đặt cơ sở dữ liệu**

|  |
| --- |
| -- Create DB  Create database StemDB  go  Use StemDB  go  -- Create TABLES  Create table dbo.Users(  UserID int IDENTITY (1,1) primary key,  Username nvarchar(15) not null,  Pass nvarchar(16) not null,  Wallet int check(Wallet >= 0),  DOB date,  CreditCard char(12),  Spent int check(Spent >= 0),  [Admin] int default 0,  )  go  Create table dbo.Studio(  StudioID int IDENTITY (1,1) primary key,  StudioName nvarchar(max) not null,  Country nvarchar(max),  )  Go  Create table dbo.Game(  GameID nvarchar(5) primary key,  StudioID int,  GameName nvarchar(max) not null,  Descri nvarchar(max),  Price int check (Price >=0 ),  Rating float Check(Rating >=0 and Rating <=5),  [Image] varbinary(max),  Lang nvarchar(100),  Download int check(Download >=0),  Release date not null,  Plat nvarchar(100),  Size float check(Size >=0 and Size <=100),  CONSTRAINT FK\_StudioGame FOREIGN KEY (StudioID) REFERENCES Studio(StudioID)  )  go  Create table dbo.RedeemCode(  Code nvarchar(10) primary key,  GameID nvarchar(5),  UserID int null,  CONSTRAINT FK\_RedeemCode FOREIGN KEY (GameID) REFERENCES Game(GameID),  CONSTRAINT FK\_RedeemUser FOREIGN KEY (UserID) REFERENCES Users(UserID),  )  go  Create table dbo.Lib(  GameID nvarchar(5),  UserID int,  BuyDate date,  CONSTRAINT FK\_GameLib FOREIGN KEY (GameID) REFERENCES Game(GameID),  CONSTRAINT FK\_UserLib FOREIGN KEY (UserID) REFERENCES Users(UserID),  )  go  Create table dbo.DisEvent(  DisID int Primary key,  DisName nvarchar(max) not null,  Amount int check(Amount >= 0),  DisType nvarchar(20) not null,  DayBegin date,  DayEnd date  )  go  Create table dbo.DisGame(  DisID int,  GameID nvarchar(5),  CONSTRAINT FK\_DisGame\_DisEvent FOREIGN KEY (DisID) REFERENCES DisEvent(DisID),  CONSTRAINT FK\_DisGame\_Game FOREIGN KEY (GameID) REFERENCES Game(GameID)  )  go  Create table dbo.Tag(  TagName nvarchar(40) Primary key  )  go  Create table dbo.Category(  TagName nvarchar(40),  GameID nvarchar(5),  CONSTRAINT FK\_Category\_Game FOREIGN KEY (GameID) REFERENCES Game(GameID),  CONSTRAINT FK\_Category\_Tag FOREIGN KEY (TagName) REFERENCES Tag(TagName)  )  go  Create table dbo.Sale(  SaleID int IDENTITY (1,1) primary key,  GameID nvarchar(5),  SaleDate date not null,  Amount int Check(Amount >=0),  CONSTRAINT FK\_Sale FOREIGN KEY (GameID) REFERENCES Game(GameID)  )  Go |

1. **Danh sách các Views, Stored procedures, Functions, Triggers**
   1. **Views**
   2. **Stored procedures**
   3. **Functions**
   4. **Triggers**

|  |
| --- |
| -- Create TRIGGERS  create trigger Delete\_Tags  on dbo.Tag  instead of delete  as  declare @TagName nvarchar(40) select @TagName = TagName from deleted  Alter table Category drop constraint FK\_Category\_Tag  Alter table Category drop constraint FK\_Category\_Game  delete Category from Category  where Category.TagName = @TagName  delete Tag from Tag  where Tag.TagName = @TagName  Alter table Category add CONSTRAINT FK\_Category\_Game FOREIGN KEY (GameID) REFERENCES Game(GameID)  Alter table Category add CONSTRAINT FK\_Category\_Tag FOREIGN KEY (TagName) REFERENCES Tag(TagName)  go  Create trigger Insert\_Category  on dbo.Category  instead of insert  as  declare @TagName nvarchar(40) select @TagName = TagName from inserted  declare @GameID nvarchar(5) select @GameID = GameID from inserted  if not exists(select \* from Category where @TagName = Category.TagName and @GameID = Category.GameID)  insert into Category values(@TagName,@GameID)  go  Create trigger Insert\_DiscountEvent  on dbo.DisGame  instead of insert  as  declare @DisID int select @DisID = DisID from inserted  declare @GameID nvarchar(5) select @GameID = GameID from inserted  if not exists(select \* from DisGame where @DisID = DisGame.DisID and @GameID = DisGame.GameID)  insert into DisGame values(@DisID,@GameID)  go  Create trigger Insert\_Studio  on dbo.Studio  instead of insert  as  declare @StudioName nvarchar(max) select @StudioName = StudioName from inserted  declare @Country nvarchar(max) select @Country = Country from inserted  if not exists(select \* from Studio where @StudioName = Studio.StudioName)  insert into Studio values(@StudioName,@Country)  go  create trigger Update\_Studio  on dbo.Studio  instead of update  as  declare @StudioNameOld nvarchar(max) select @StudioNameOld = StudioName from deleted  declare @StudioName nvarchar(max) select @StudioName = StudioName from inserted  declare @Country nvarchar(max) select @Country = Country from inserted  if not exists(select Studio.StudioName from Studio where @StudioName = Studio.StudioName)  begin  update Studio set StudioName = @StudioName , Country=@Country where StudioName = @StudioNameOld  end  else  begin  update Studio set Country=@Country where StudioName = @StudioNameOld  end  go  Create trigger Delete\_Studio  on dbo.Studio  instead of delete  as  declare @StudioID int select @StudioID = StudioID from deleted  update Game  set Game.StudioID = NULL  from Game  where Game.StudioID = @StudioID  delete from Studio where Studio.StudioID = @StudioID  go  Create trigger Delete\_Users  on dbo.Users  instead of delete  as  declare @UserID int select @UserID = UserID from deleted  Alter table Lib drop constraint FK\_GameLib  Alter table Lib drop constraint FK\_UserLib  delete Lib from Lib  where Lib.UserID = @UserID  delete Users from Users  where Users.UserID = @UserID  Alter table Lib add constraint FK\_GameLib foreign key (GameID) references Games(GameID)  Alter table Lib add constraint FK\_UserLib foreign key (UserID) references Users(UserID)  go  create trigger Delete\_DisEvent  on dbo.DisEvent  instead of delete  as  declare @DisID int select @DisID = DisID from deleted  Alter table DisGame drop constraint FK\_DisGame\_DisEvent  Alter table DisGame drop constraint FK\_Disgame\_Game  delete DisGame from DisGame  where DisGame.DisID = @DisID  delete DisEvent from DisEvent  where DisEvent.DisID = @DisID  Alter table DisGame add constraint FK\_DisGame\_DisEvent foreign key (DisID) references DisEvent(DisID)  Alter table DisGame add constraint FK\_Disgame\_Game foreign key (GameID) references Game(GameID)  go  create TRIGGER DeleteGame ON dbo.Game  instead of delete  as  declare @GameID nvarchar(5) select @GameID = GameID from deleted    Alter table Lib drop CONSTRAINT FK\_GameLib  Alter table Lib drop CONSTRAINT FK\_UserLib  Alter table DisGame drop CONSTRAINT FK\_DisGame\_DisEvent  Alter table DisGame drop CONSTRAINT FK\_DisGame\_Game  Alter table Category drop CONSTRAINT FK\_Category\_Game  Alter table Category drop CONSTRAINT FK\_Category\_Tag  Alter table RedeemCode drop CONSTRAINT FK\_RedeemCode  Alter table Sale drop CONSTRAINT FK\_Sale  delete Game from Game  where Game.GameID = @GameID  delete Lib from Lib  where Lib.GameID = @GameID  delete RedeemCode from RedeemCode  where RedeemCode.GameID = @GameID  delete DisGame from DisGame  where DisGame.GameID = @GameID  delete Category from Category  where Category.GameID = @GameID  delete Sale from Sale  where Sale.GameID = @GameID  Alter table Lib add CONSTRAINT FK\_GameLib FOREIGN KEY (GameID) REFERENCES Game(GameID)  Alter table Lib add CONSTRAINT FK\_UserLib FOREIGN KEY (UserID) REFERENCES Users(UserID)  Alter table DisGame add CONSTRAINT FK\_DisGame\_DisEvent FOREIGN KEY (DisID) REFERENCES DisEvent(DisID)  Alter table DisGame add CONSTRAINT FK\_DisGame\_Game FOREIGN KEY (GameID) REFERENCES Game(GameID)  Alter table Category add CONSTRAINT FK\_Category\_Game FOREIGN KEY (GameID) REFERENCES Game(GameID)  Alter table Category add CONSTRAINT FK\_Category\_Tag FOREIGN KEY (TagName) REFERENCES Tag(TagName)  Alter table RedeemCode add CONSTRAINT FK\_RedeemCode FOREIGN KEY (GameID) REFERENCES Game(GameID)  Alter table Sale add CONSTRAINT FK\_Sale FOREIGN KEY (GameID) REFERENCES Game(GameID)  go |

1. **Tạo user, phân quyền**
2. **Hướng dẫn sử dụng**
   1. **Cài đặt chương trình**

* Bước 1: Restore Database
  + Chạy các file \*.sql theo thứ tự sau
    - 00\_DB\_Blank.sql
    - 01\_View.sql
    - 02\_Function.sql
    - 03\_Procedure.sql
    - 04\_ProcedureWithTrans.sql
    - 05\_Trigger.sql
    - 06\_RecentralizedRights.sql
    - 07\_InsertDataSample.sql
* Bước 2: Khởi chạy project
  + Mở file DormitoryManagement.sln để khởi động project bằng Visual Studio 2019
  + Chạy phần mềm: các tài khoản đăng nhập:
  1. **Chi tiết** 
     1. ***Khởi động ứng dụng (Guest)***
     2. ***Đăng nhập***

1. **Kết luận**
   1. **Các phần đã đạt được trong đồ án**
   2. **Các phần đặc sắc, đặc biệt trong đồ án**
   3. **Một số hạn chế**
   4. **Cam kết**

Chúng em xin cam đoan đồ án “Quản lí Kí Túc Xá” là thành quả của quá trình học tập và tìm hiểu dưới sự hướng dẫn của thầy Nguyễn Thành Sơn. Với lòng biết ơn sâu sắc và chân thành nhất, chúng em xin gửi đến thầy đã cùng với tri thức và tâm huyết của mình, truyền đạt vốn kiến thức quý báu cho chúng em trong suốt học kỳ này. Với điều kiện thời gian cũng như kinh nghiệm còn hạn chế, bài báo cáo này không thể tránh được những thiếu sót. Chúng em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của thầy để chúng em có điều kiện bổ sung, nâng cao ý thức của mình, phục vụ tốt hơn công tác thực tế sau này. Mọi sự tham khảo đều có trích nguồn. Nếu có sự vi phạm nào chúng em xin hoàn toàn chịu trách nhiệm.