

**<<GameStore>>**

**Software Design Document**

– HCM, October 2022 –

**Table of Contents**

[I. Overview 3](#_Toc69802609)

[1. Code Packages 3](#_Toc69802610)

[2. Database Schema 3](#_Toc69802611)

[II. Code Designs 4](#_Toc69802612)

[1. <Feature/Function Name1> 4](#_Toc69802613)

[a. Class Diagram 4](#_Toc69802614)

[b. Class Specifications 4](#_Toc69802615)

[c. Sequence Diagram(s) 4](#_Toc69802616)

[d. Database queries 5](#_Toc69802617)

[2. <Feature/Function Name2> 5](#_Toc69802618)

[III. Database Tables 5](#_Toc69802619)

[1. <Table name 1> 5](#_Toc69802620)

[2. <Table name 2…> 5](#_Toc69802621)

# I. Overview

## 1. Code Packages/Namespaces

Diagram

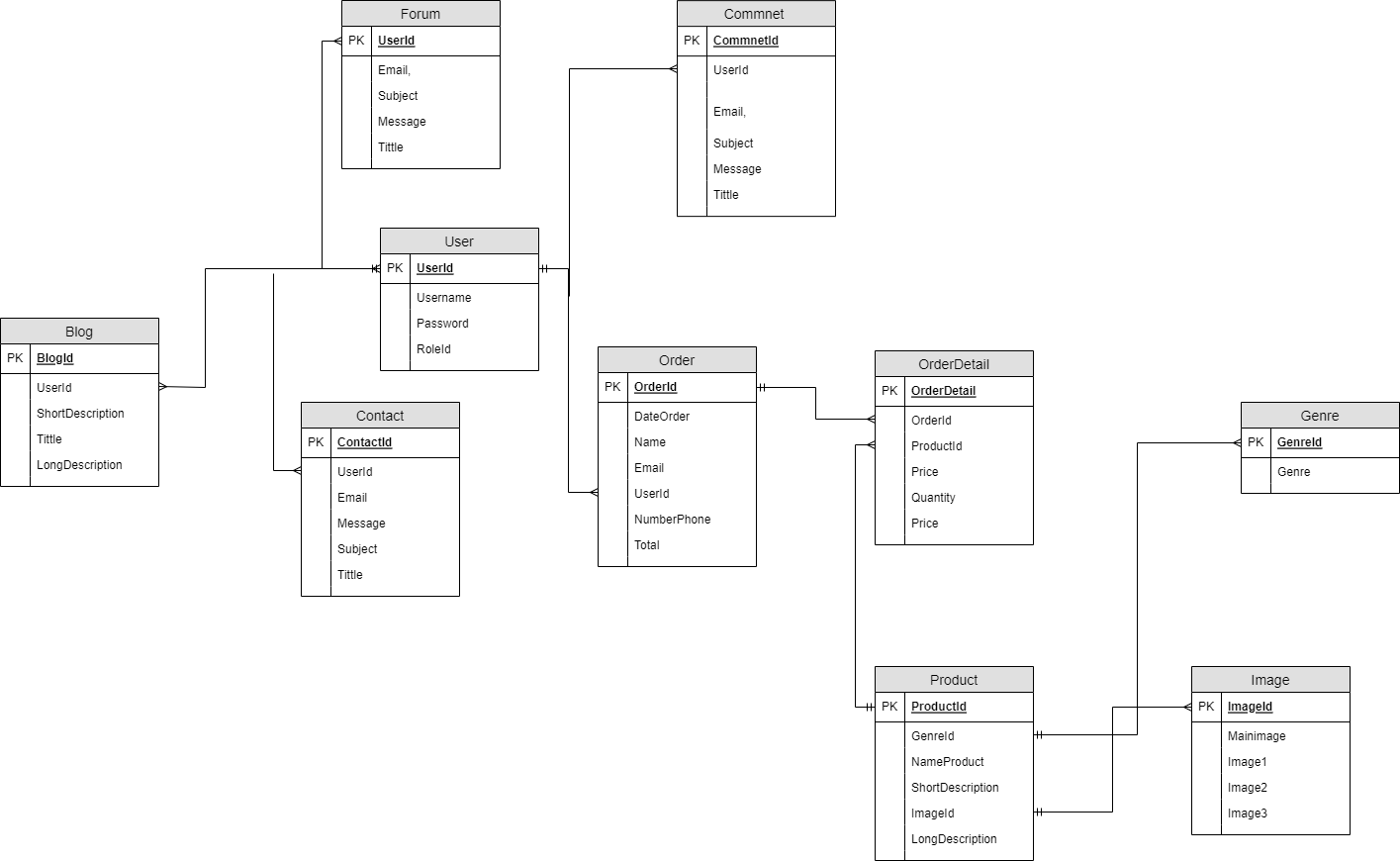
Description automatically generated

***Package descriptions & package class naming conventions***

|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *01* | *View* | *Web view for admin, host, user, guess* |
| *02* | *Controller* | *Controller of admin, host, user, website* |
| *03* | *DAO* | *List shose* |
| *04* | *Filter* | *Sort for admin, host, user* |
| *05* | *Ultils* | *Database sql* |
| *06* | *Model* | *ProductID, ProductName, ProductQuanity, ProductDescription, Rating, ProductPrice, BrandID, Size,....* |

## 2. Database Schema

*[Provide the tables relationship like example below – following MySQL database naming convention]*



***Table descriptions & package class naming conventions are as below***

|  |  |  |
| --- | --- | --- |
| **No** | **Table** | **Description** |
| *01* | *User* | *Save information about users.*  *UserID, Username, Password, Role.*  *- Primary keys: UserID.*  *- Foreign keys:* Non |
| *02* | *Order* | *Save Invoice and Time of user's last payment, Including Total Amount. New Address and Phone Number and Email can be added.*  OrderId, DateOrder, Name, Email, UserId, NumberPhone,Total  *- Primary keys:* OrderId*.*  *- Foreign keys:* UserId*.* |
| *03* | *OrderDetail* | *Report the price corresponding to the quantity of each product that the user buys*  *OrderDetail,* OrderId, ProductId, Price, Quantity, Price*.*  *- Primary keys: OrderDetail.*  *- Foreign keys:* OrderId, ProductId*.* |
| *04* | *Image* | *Show pictures on home page and when buyers want to see details.*  *ImageId,* Mainimage*,* Image1*,* Image2, *Image3 .*  *- Primary keys: ImageId.*  *- Foreign keys:* Non |
| *05* | *InterView* | *In the form of a reporter asking and an interviewee answering*  *NameApplicant,* Gender, MarialStatus, Dob, Address, CMND, Email, Phone, IntroduceMyseft, CMD, Education, University, Major, NameJob, Image, InterviewDate, InterviewTime, Interviewer, CV, Status.  *- Primary keys: NameApplicant.*  *- Foreign keys:* Non |
| *06* | *Genre* | *Show game genres*  *GenreId,* Genre*.*  *- Primary keys: GenreId.*  *- Foreign keys:* Non |
| *07* | *Blog* | *An online diary*  *BlogId,* UserId*,* ShortDescription, Tittle, LongDescription*.*  *- Primary keys: BlogId.*  *- Foreign keys:* UserId*.* |
| *08* | *Product* | *Products are clearly shown*  *ProductId,* GenreId*,* NameProduct*,* ShortDescription*,* ImageId*,* LongDescription*.*  *- Primary keys: ProductId.*  *- Foreign keys:* GenreId*,* ImageId*.* |
| *09* | *Commnet* | A verbal or written remark expressing an opinion or reaction.  *Commnet*Id , UserId, Email, Subject, Message, Tittle  *- Primary keys: Commnet*Id*.*  *- Foreign keys:* UserId |
| *10* | *Forum* | *a place, meeting, or medium where ideas and views on a particular issue can be exchanged.* UserId, Email, Subject, Message, Tittle.  *- Primary keys:* UserId*.*  *- Foreign keys:* Non |
| *11* | *Contact* | *The state or condition of physical touching.*  *ContactId,* UserId, Email, Subject, Message, Tittle.  *- Primary keys: ContactId.*  *- Foreign keys:* UserId |

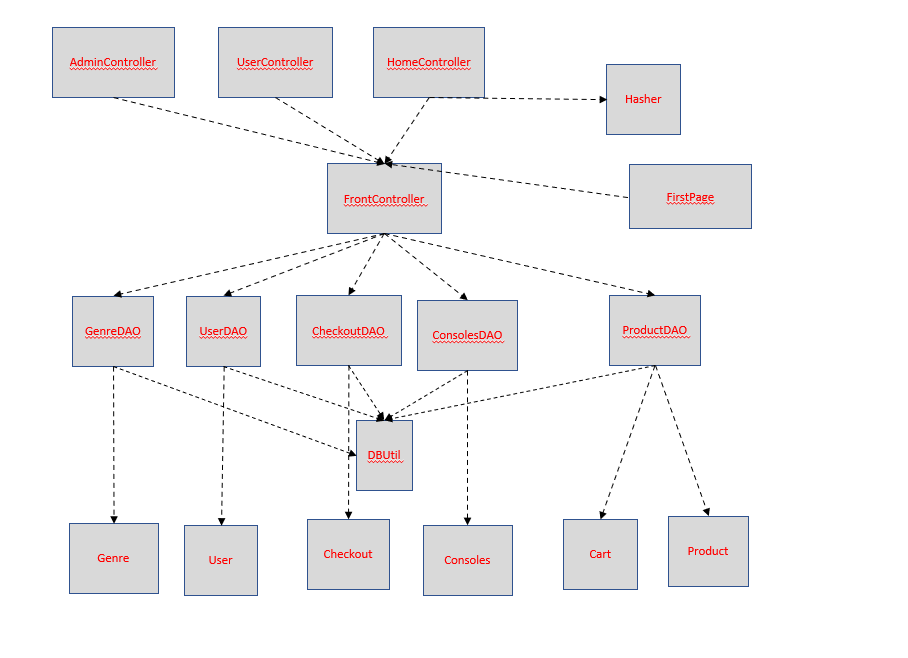
# II. Code Designs

## 1. <Feature/Function Name1>

*[Provide the detailed design for the function <Feature/Function Name1>. It include Class Diagram, Class Specifications, and Sequence Diagram(s)]*

### a. Class Diagram

*[This part presents the class diagram for the relevant feature]*



### b. Class Specifications

*[Provide the description for each class and the methods in each class, following the table format as below]*

#### DBUtil Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *connect to database* |

#### Hasher Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *hash()* | *-Input: Password*  *-Output: Password into format SHA-256*  *Get human password and encrypt SHA-256* |

#### CheckoutDAO Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *addOrder()* | *Add data of phone, email, name, price, address* |

#### ConsolesDAO Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *list()* | *returns a list of console* |

#### ProductDAO Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *listNew()* | *get a list of newly added products* |
|  | *listHome()* | *Show product* |
|  | *listAll()* | *Show all products the store has* |
|  | *minPrice()* | *Minimum price limit* |
|  | *maxPrice()* | *Maximum price limit* |
|  | *filter()* | *product search by money limit* |
|  | *findProductByName*  *()* | *The user enters the product name he wants to find, returns a list of product names he wants to find* |
|  | *getProductbyID()* | *Search products by user-entered product ID, return products by ID* |
|  | *listFull()* | *Return all products in the store* |
|  | *updateProduct()* | *Update product information on store by admin* |
|  | *delete()* | *Delete product on store by admin* |
|  | *createProduct()* | *Add new product by admin* |
|  | *listByID()* | *Search products by user-entered product ID by admin* |
|  | *listByName()* | *The user enters the product name he wants to find by admin* |
|  | *findProductByGenreID()* | *The user enters the product genre he wants to find, returns a list of product names he wants to find* |

#### GenreDAO Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *list()* | *Returns list of genre* |

#### UserDAO Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *check()* | *Check login* |
|  | *check\_web()* | *Get list user* |
|  | *check\_duplication()* | *Check user have already* |
|  | *register()* | *Insert database user register* |
|  | *find()* | *Get list user* |
|  | *update()* | *Update password* |

***FrontController Class***

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *processRequest()* | *Cut urls writing into actions and feed to the right controllers to handle the request* |

#### AdminController Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *paging()* | *used for pagination* |
|  | *countP()* | *used for pagination* |
|  | *listBase()* | *List Genre and List Console* |
|  | *listFull()* | *Show all list produte* |
|  | *update()* | *Update about product* |
|  | *delete()* | *Delete about product* |
|  | *create()* | *Add new product* |
|  | *findByID()* | *Search product by id* |
|  | *findByName()* | *Search product by name* |

#### FirstPage Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *processRequest()* | *first page navigation will open* |

#### HomeController Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *homepage()* | *Get all the information you need to display on the sales page* |
|  | *shop-grid()* | *Cantain function* |
|  | *shop-details()* | *create a new cart where the user is logged in* |
|  | *shoping-cart()* | *Contain function* |
|  | *check\_cookies()* | *Check login user, to show welcome user* |
|  | *listBase()* | *List Genre and List Console* |
|  | *countP()* | *used for pagination* |
|  | *listAll()* | *Page show full product* |
|  | *paging()* | *used for pagination* |
|  | *filter()* | *Get list product by max price min price* |
|  | *search()* | *Get list product by name* |
|  | *listByGenre()* | *Get list product by genre* |
|  | *add()* | *Up quantity product user choose* |
|  | *sub()* | *Down quantity product user choose* |
|  | *delete()* | *Delete product from cart* |
|  | *create\_cartCookie()* | *Save product user have been choose* |
|  | *view()* | *show all products quantity and price that the user has purchased* |
|  | *checkout()* | *Contain function view\_checkout() and placeOrder()* |
|  | *view\_checkout()* | *Redirect* |
|  | *placeOrder()* | *Show cart and buyer's information, otherwise it will not be reported.* |

#### UserController Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *confirmMail()* | *Confirm mail to user when register, insert database* |
|  | *loginHandler()* | *Set cookies* |
|  | *Logout()* | *Logout account* |
|  | *List()* | *Show prooduct* |
|  | *findUser()* | *Check database user have already, send code to usermail* |
|  | *resetPassword()* | *Change pass* |

#### Cart Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Cart()* | *Information about Cart* |

#### Genre Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Genre()* | *Information about Genre* |

#### User Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *User()* | *Information about User* |

#### Checkout Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Checkout()* | *Information about Checkout* |

#### Consoles Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Console()* | *Information about Consoles* |

#### Product Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Product*() | *Information about product* |

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *PayPayPay* | User can pay by paypal |
| 2 | *Contact* | *User contact and admin get it* |
| 3 | *Comment* | *User comment and admin get it* |
| *4* | *Forum* | *user can post anything* |
| *5* | *Paging* | *Paging* |
| *6* | *Comment* | *record user comments on the site and admin can know it* |
| *7* | *Blog* | *contains more delete edits about the blog* |
| *8* | *Account* | *Contains more, delete, edit accounts for employees with the function of adding and editing posts* |
| *9* | *Checkout* | *admin and user can check the whole order* |

### c. Sequence Diagram(s)

*[Provide the sequence diagram(s) for the feature, see the sample below]*



### d. Database queries

Get user by pass and name

select \* from Users where UserName=? and Passwords=?

Count user have name

select count(UserID) as noOfUser from Users where UserName=?

Create database user when register

insert Users values(?, ?, ?, ?)

Get all user have name

select \* from Users where UserName=?

Change pass if user have name

update users set Passwords=? where UserID=?

Get information name product denpend on Genre and Consoles

"SELECT ProductID,ProductPrice,ProductName,\n"

+ " ProductQuantity,ProductDesc,Rating,LinkIMG1,\n"

+ " LinkIMG2,LinkIMG3,LinkIMG4,LinkIMG5,\n"

+ " Product.GenreID,GenreName,Product.ConsolesID,ConsolesName \n"

+ "FROM dbo.Product LEFT JOIN dbo.Genre \n"

+ "ON Genre.GenreID = Product.GenreID \n"

+ "LEFT JOIN dbo.Consoles \n"

+ "ON Consoles.ConsolesID = Product.ConsolesID\n"

+ "Where productName like ?";

Get information name product denpend on Genre

SELECT ProductID, ProductPrice,ProductName,LinkIMG1\n"

+ "FROM dbo.Product \n"

+ "Where genreID = ?");

Get all list Genre

SELECT \* FROM dbo.Genre

Get all list Consoles

SELECT \* FROM dbo.Consoles

Save information about the user's purchases

INSERT INTO dbo.Checkout values (?,?,?,?,?,?,?)

Paging

Select \* from ORDER BY ID OFFSET ? ROWS FETCH NEXT 4 ROWS ONLY

## 2. <Feature/Function Name2>

…

# III. Database Tables

**1. <User>**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | UserId | Varchar | 30 |  | False | PK |  |
| 2 | Username | Nvarchar | 50 |  | True |  |  |
| 3 | Password | Nvarchar | 50 |  | False |  |  |
| 4 | RoleId | Nvarchar | 50 |  | True |  |  |

**2. <Order >**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | OrderId | int |  |  | False | PK |  |
| 2 | DateOrder | Date | 50 |  | False |  |  |
| 3 | Name | Varchar | 50 |  | False |  |  |
| 4 | Email | Varchar | 200 |  | False |  |  |
| 5 | UserId | int | MAX |  | False | FK |  |
| 6 | NumberPhone | Decimal | (1,0) |  | False |  |  |
| 7 | Total | Decimal | 30 |  | False | FK |  |

**3. <Blog>**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | BlogId | Nvarchar | 50 |  | False | PK |  |
| 2 | UserId | Nvarchar | 50 |  | False |  |  |
| 3 | ShortDescription | Nvarchar | 50 |  | False |  |  |
| 4 | Tittle | Nvarchar | 50 |  | False |  |  |
| 5 | LongDescription | Nvarchar | 50 |  | False |  |  |

**4. <Genre>**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | GenreId | int |  |  | False | PK |  |
| 2 | Genre | Varchar |  |  | False |  |  |

**5. <Image>**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | ImageId | int |  |  | False | PK |  |
| 2 | MainImage |  |  |  |  |  |  |
| 3 | Image1 |  |  |  |  |  |  |
| 4 | Image2 |  |  |  |  |  |  |
| 5 | Image3 | Varchar |  |  | False |  |  |

**6. <OrderDatail>**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | OrderDeatail | Int |  |  | False | PK |  |
| 2 | OrderId | Int | 50 |  | False | FK |  |
| 3 | ProductId | Int |  |  | False | FK |  |
| 4 | Price | Int |  |  | False |  |  |
| 5 | Quantity | Int |  |  | False |  |  |
| 6 | Total |  |  |  | False |  |  |

**7. <Product>**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | ProductId | int |  |  | False | PK |  |
| 2 | GenreId | int |  |  | False | FK |  |
| 3 | NameProduct | nvarchar |  |  | False |  |  |
| 4 | ShortDescription | nvarchar |  |  | False |  |  |
| 5 | ImageId | nvarchar |  |  | False |  |  |
| 6 | LongDescription | nvarchar |  |  | False |  |  |

**8. <Contact>**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | UserId | Int |  |  | False | PK |  |
| 2 | Subject | Nvarchar |  |  | False |  |  |
| 3 | Tittle | Nvarchar | 30 |  | False |  |  |
| 4 | Message | Nvarchar | 500 |  | False |  |  |
| 5 | Email | Nvarchar | 50 |  | False |  |  |

**9. <** **Forum >**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | **UserId** | Int |  |  | False | PK |  |
| 2 | **Email** | Nvarchar |  |  | False |  |  |
| 3 | **Subject** | Nvarchar | 30 |  | False |  |  |
| 4 | **Message** | Nvarchar | 500 |  | False |  |  |
| 5 | **Tittle** | Nvarchar | 50 |  | False |  |  |

**10. <** **Comment >**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Allow Null** | **PK/FK** | **Notes** |
| 1 | **UserId** | Int |  |  | False | PK |  |
| 2 | **Email** | Nvarchar |  |  | False |  |  |
| 3 | **Subject** | Nvarchar | 30 |  | False |  |  |
| 4 | **Message** | Nvarchar | 500 |  | False |  |  |
| 5 | **Tittle** | Nvarchar | 50 |  | False |  |  |