Project Description:

Online Market where Users can make online orders.

The Market has 3 categories:

1- Electrical Appliances
2- Food Products
3- Sports

MVC Web application using .NET core

Languages: C#, HTML5, CSS3, Bootstrap, JS

Project Details:

The project has 3 Controllers & 3 Views & 8 Models:

- **Controllers:-**

1- admin controller

- **ShowAdmins**: view all admins in ShowAdmins view

- **DeleteAdmin** : call function delete admin, return to the view show admins

- **AddAddmin**: show the view add admin

- **Addmin** : show the login form for admin

- **AddminRegister**: show the register form of admin

- **AdminRegister** : call function CheckAdminRegister , if true then call funcion addAmin in Admin Class, return to Addmin function

- **CheckAdminRegister** : return true if the email does not exist in the database, otherwise return false

- **AdminLogin**: check admin email , check admin password , if true show all products in the database , else go to index function in the home controller that contains the login or register form

- **CheckAdminEmail** : return true if the email exists in the database, otherwise return false

- **CheckAdminPassword**: check the password is true or not

- **RenderAdminData**: get admin data from database and put it in temp data

- **Addproduct**: show the view of add product

- **addProductItem**: get the data from the form that admin filled, set the time to the time of inserting the product, set the image product, call the function add product in product class to add the product to database, return to the view of add product again

- **productTable**: call the function get products in product class that gets all products from database, and view it in the product table view

- **getDataByID**: searches for the product in the database with its id and puts its data into temp data

- **SendDataTOform**: gets the product data from temp data and puts it in the model, return this model

- **updateProduct**: get the data of product by its id, send the data to the form and view the form with the product data

- **updateProductItem**: get the id of the item and get the image of the product, call the function updateProduct in product class to save changes, return to productTable function

- **deleteProduct**: get the id of the product, call the function deleteProduct in product class, return to productTable function

- **ShowFeedback**: call the function GetFeedbacks from admin class, show them in ShowFeedback view

- **DeleteFeedback** : call the function deleteFeedback from Feedback class, return to ShowFeedback function

- **ShowComments** : call the function ShowComment in Comments class, show them in ShowComments view

- **DeleteComment** : call the function deleteComment in Comments class, return to function ShowComments

- **ShowUsers** : call the function getUsers from User class, show them in ShowUsers view

- **DeleteUser** : call the function deleteUser from User class, return to function ShowUsers

- **AdminSearch** : call the function searchForProduct in product class, show it in AdminSearch view

2- User Controller:

- **Register**: show the Register view that contains the register form

- **UserRegister** : call the function CheckRegister in User class, if there is a user with this data, then call the function addUser in User class to add user to the database & return to Index function in Home controller that contains the user login form, else return to function Register

- **UserLogin** : call the function CheckUserEmail , if true then call the function CheckUserPassword , if true then call function RenderUserData & return to HomePage in Home Controller, else return to Index function in Home controller

- **CheckUserEmail** : return true if the email is existing in the database, otherwise return false

- **CheckUserPassword** : return true if the password exists in the database, otherwise return false

RenderUserData : get the data from database and render it to temp data

- **Feedback**: show the feedback view

- **AddFeedback** : get the feedback from the form, call the function addFeedback in Feedback class, return to function HomePage in Home controller

- **getDataByID**: get the data of the user from database and render it to temp data

- **Comments**: call the function GetDataByID , return the Comments view

- **AddComment** : get the id of the product that user will comment it, if the id not equal null, then call the function addComment in Comments class to save changes, return to HomePage function in home controller, else return to Comments function

- **ShowProduct** :get the product from the database and render it to temp data, return the view of ShowProduct to show that product

- **UserSearch**: call the function searchForProduct from product class, return to UserSearch view

- **AddItemToCart**: gets the product from database, checks if the product is in the cart, if not in the cart, then adds it, else increase the quantity by 1, return to HomePage function in Home Controller

- **CartItems**: gets list of items I the cart

- **RemoveItem**: calls function GetItem in CartItem Model, remove it from database

- **DecreaseQuantity**: gets item from database, decrease quantity by 1, then save changes to the database

- **CheckZeroQuantity**: checks if the quantity is zero, then removes it from the cart

- **Checkout**: gets the items that user selected

- **CheckoutDetails**: gets the details of items that user selected

- **Confirm**: calls functions GetConfirmData, SaveSalesData, DecreaseProducts, ClearCart, then return to HomePage function in Home Controller

3- Home controller:

- **Index**: user sign in form

- **HomePage**: return all products in the database

- **CartItems**: gets list of items I the cart

- **Models**:

1- MarketDBContext : converts Classes into tables in the database

2- Admin:

- **addAdmin**: adds an admin to the database

- **GetAdmins**: get all admins from the database

- **getAdminByID**: get specific admin from database

- **deleteAdmin**: call the function getAdminByID , remove it from database

- GetFeedbacks : get all feedbacks from database

- **GetComments** : get all comments from database

3- User:

- **getUsers**: get all users from database

- CheckRegister: return true if this email does not exist in the database otherwise return false

- **addUser**: adds user in the database

- **getUserByID:** get specific user from database by ID

- **deleteUser**: deletes user from database

4- product:

- **getProducts**: return all products from database

- **addProduct**: adds product to the database

- **updateProduct**: updates a product in the database

- **getProductByID**: get specific product from database

- **deleteProduct** : deletes product from the database

- **searchForProduct** : searches for a specific product in the database

5- Comments

- **addComment** : adds comment in the database

- **deleteComment** : delete specific comment from the database

- **getComment** : get specific comment from database

- **getProductName**: gets the name of specific product

- **setCommentDetails**: makes an object of ProductJoinComment , set the product and comment, adds it to the list, return this list

- **ShowComment**: call the function setCommentDetails, show the view ShowComment

6- Feedback

- **addFeedback** : adds feedback to the database

- **deleteFeedback** : deletes a specific feedback from database

- **getFeedback** : gets specific feedback from database

7- ProductJoinComment

8- CartItem

- **CheckProductInCart:** returns specific product from database

- **SaveNewProduct**: adds product to the cart in the database

- **SaveChanges**: saves the changes in the database

- **ListOfItems**: gets the products in the cart, adds them in list, return this list

- **GetCartItems**: returns list of all products that specific user order

- **GetItem**: gets the product in the cart

- **GetConfirmData**: returns list of all products that user order

- **ClearCart**: calls function GetCartItems , loop on them, remove them from database

9- SalesReport

- **SaveSalesData**: adds sales report to the database

- **GetSalesData**: gets sales report from database

➢ Views:

1- Admin

- **AddAdmin** : register form for admin
- **Addmin** : login form
- **AddminRegister** : register form for admin
- **AddProduct** : add product form
- **AdminSearch** : search for product page
- **ShowAdmins** : show all admins
- **ShowComments** : show all comments
- **ShowFeedback** : show all feedbacks
- **ShowUsers** : show all users
- **productTable** : show all products
- **updateProduct** : update product form

2- User

- **Comments**: comment form
- **Feedback**: feedback form
- **Register**: user register form
- **ShowProduct** : show details of specific product
- **UserSearch** : search for products
- **cart**: dropdown list to user cart
- **Checkout**: checkout page

- **CheckoutDetails** : checkout details page

3- Home

- **HomePage**

- **Index**: user login form

- **cart**: dropdown list to user cart

➢ Interfaces

1- Admin Interfaces:

- IAdminAuthentication

- AdminRegister
- AdminLogin
- AddminRegister

- IAdminControl

- CheckAdminRegister
- CheckAdminEmail
- CheckAdminPassword
- AdminSearch

- IAdminModification

- DeleteAdmin
- DeleteAdmin
- Addmin

- ICommentData

- ShowComments

- IcommentModification

- DeleteComment

- IFeedbackModification
    - DeleteFeedback
- IproductData
    - Index
    - ProductTable
    - SendDataTOform
- IProductModification
    - Addproduct
    - AddProductItem
    - UpdateProduct
    - UpdateProductItem
    - DeleteProduct
- ISalesReport
    - SalesReport
- IshowAdminData
    - ShowAdmins
    - RenderAdminData
    - GetDataByID
- IShowFeedback
    - ShowFeedback

2- User Interfaces:

- ICart

  - AddItemToCart
  - CartItems
  - RemoveItem
  - DecreaseQuantity

- ICheckout

  - CheckZeroQuantity
  - Checkout
  - CheckoutDetails
  - Confirm

- ICommentModification

  - Comments
  - AddComments

- IFeedbackModification

  - Feedback
  - AddFeedback

- IProductData

  - GetDataByID
  - ShowProduct

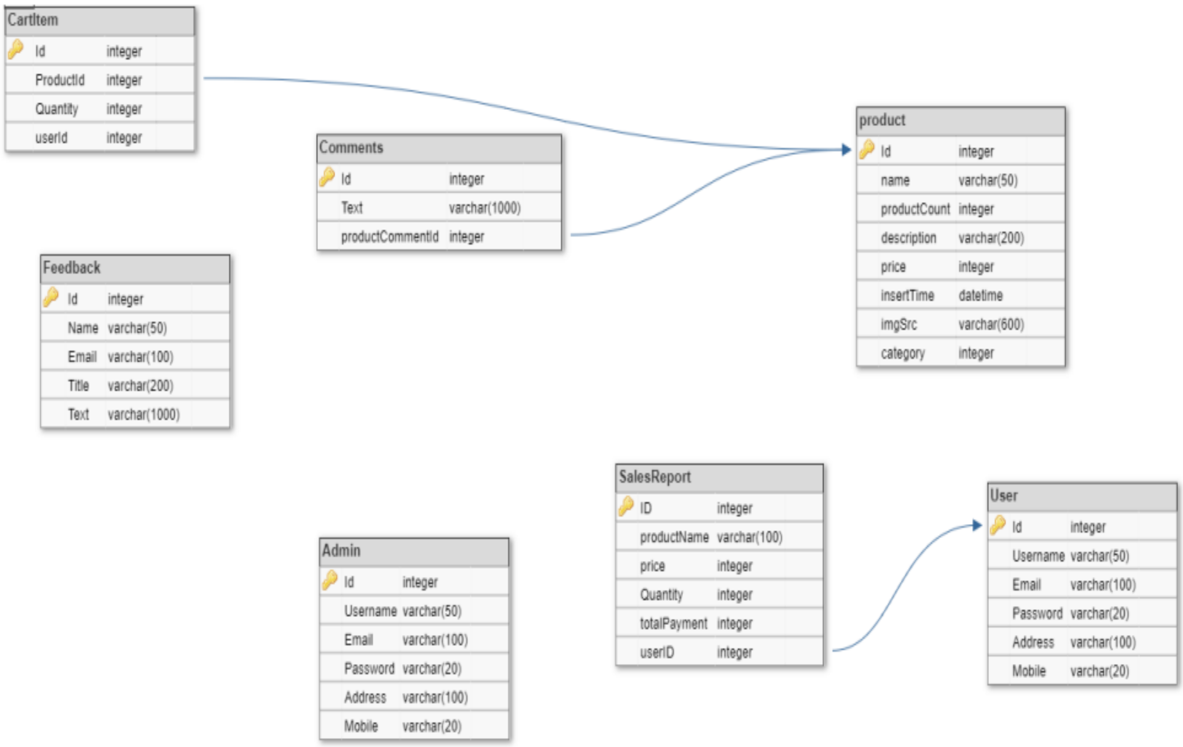
- IUserControl

  - CheckUserEmail
  - CheckUserPassword

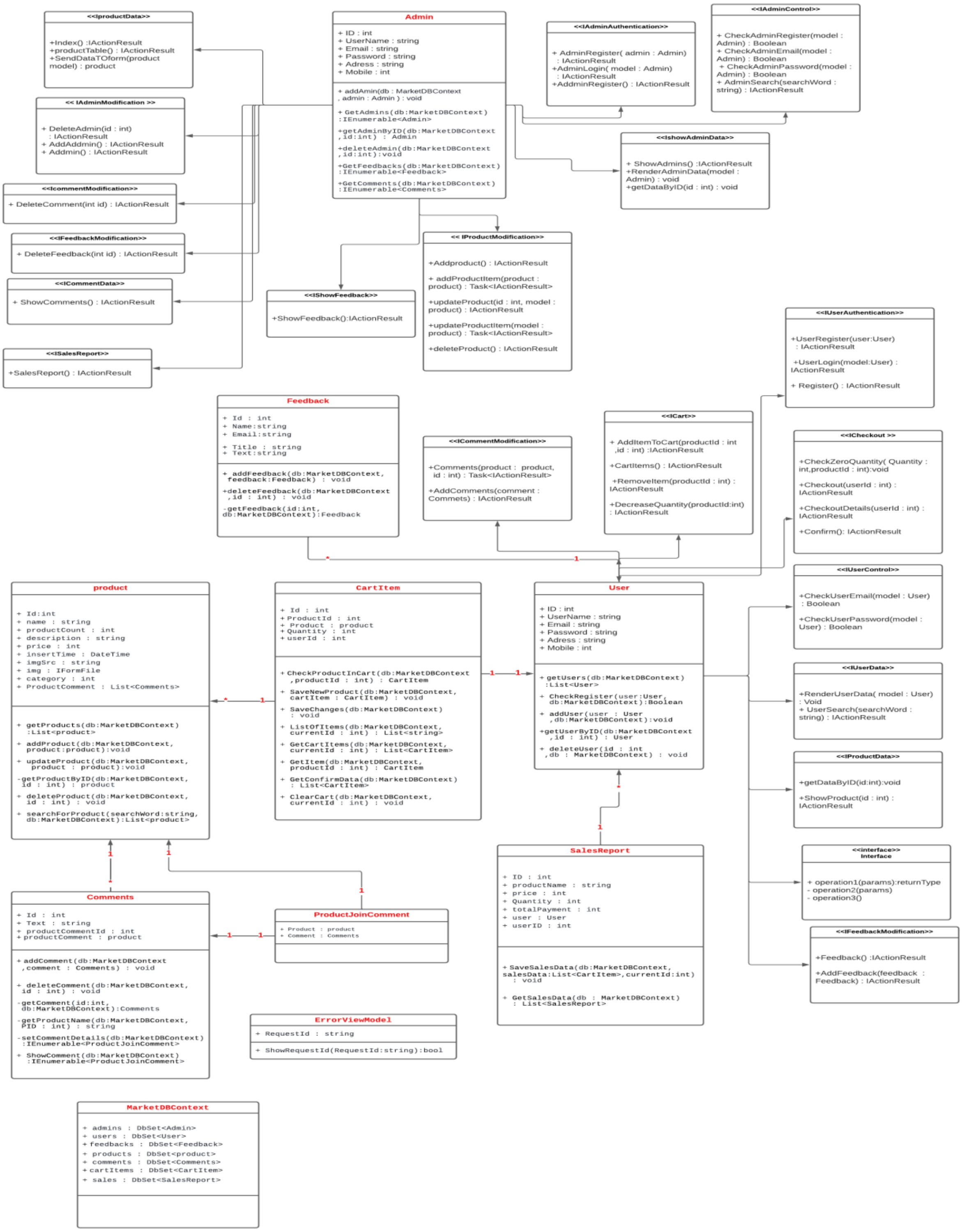
- IUserData

  - RenderUserData
  - UserSearch

Database Schema

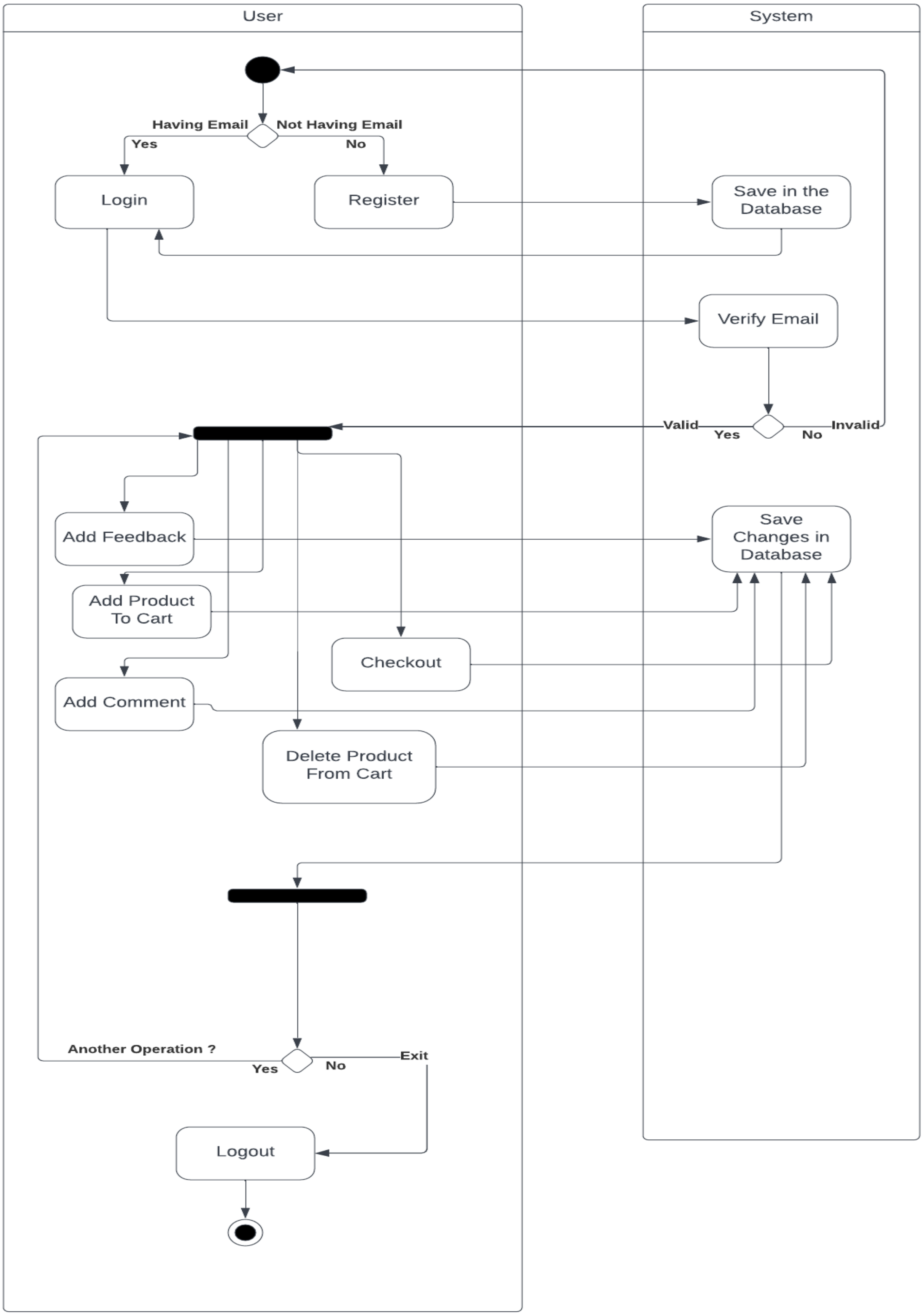


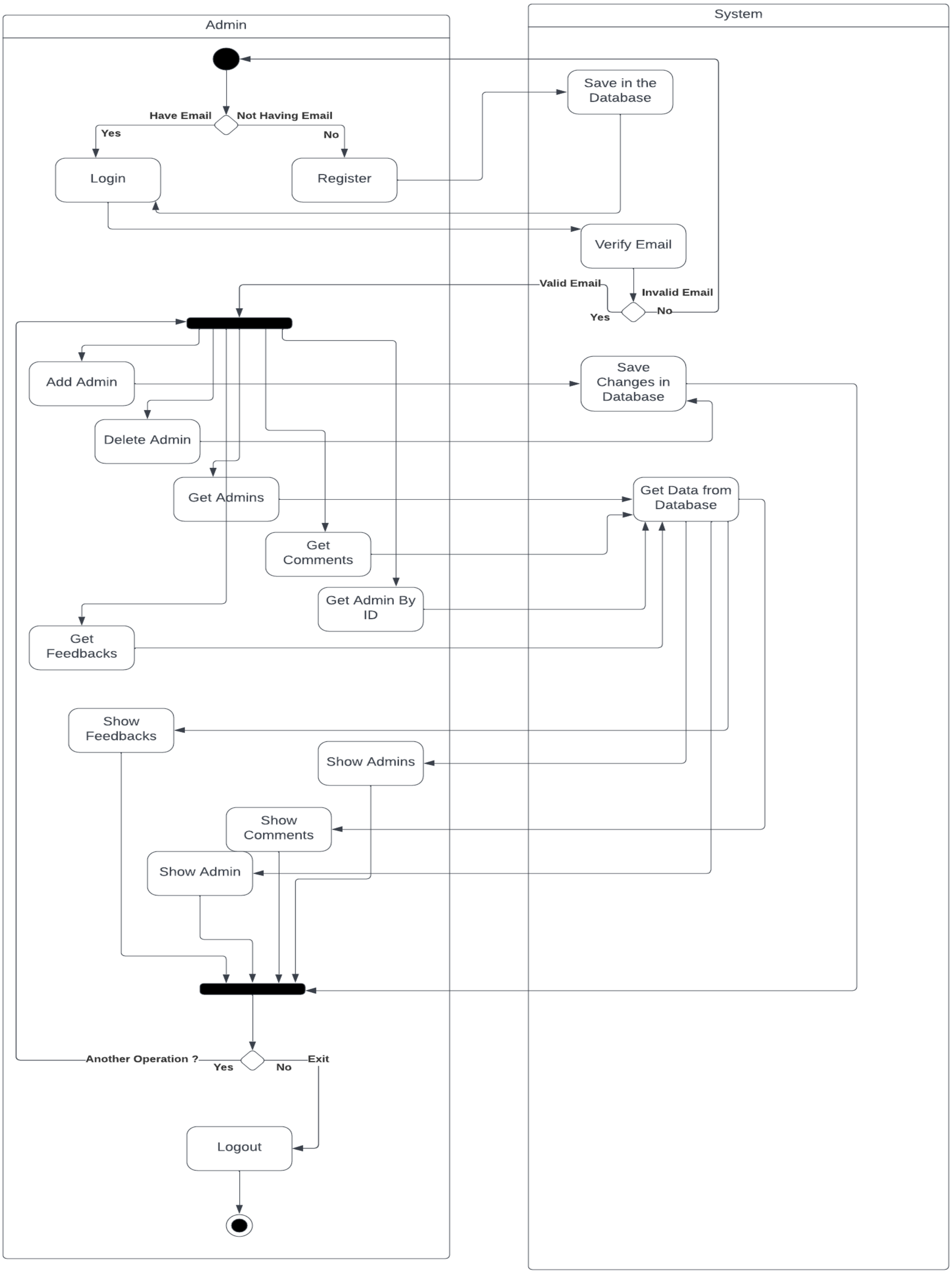
Class Diagram



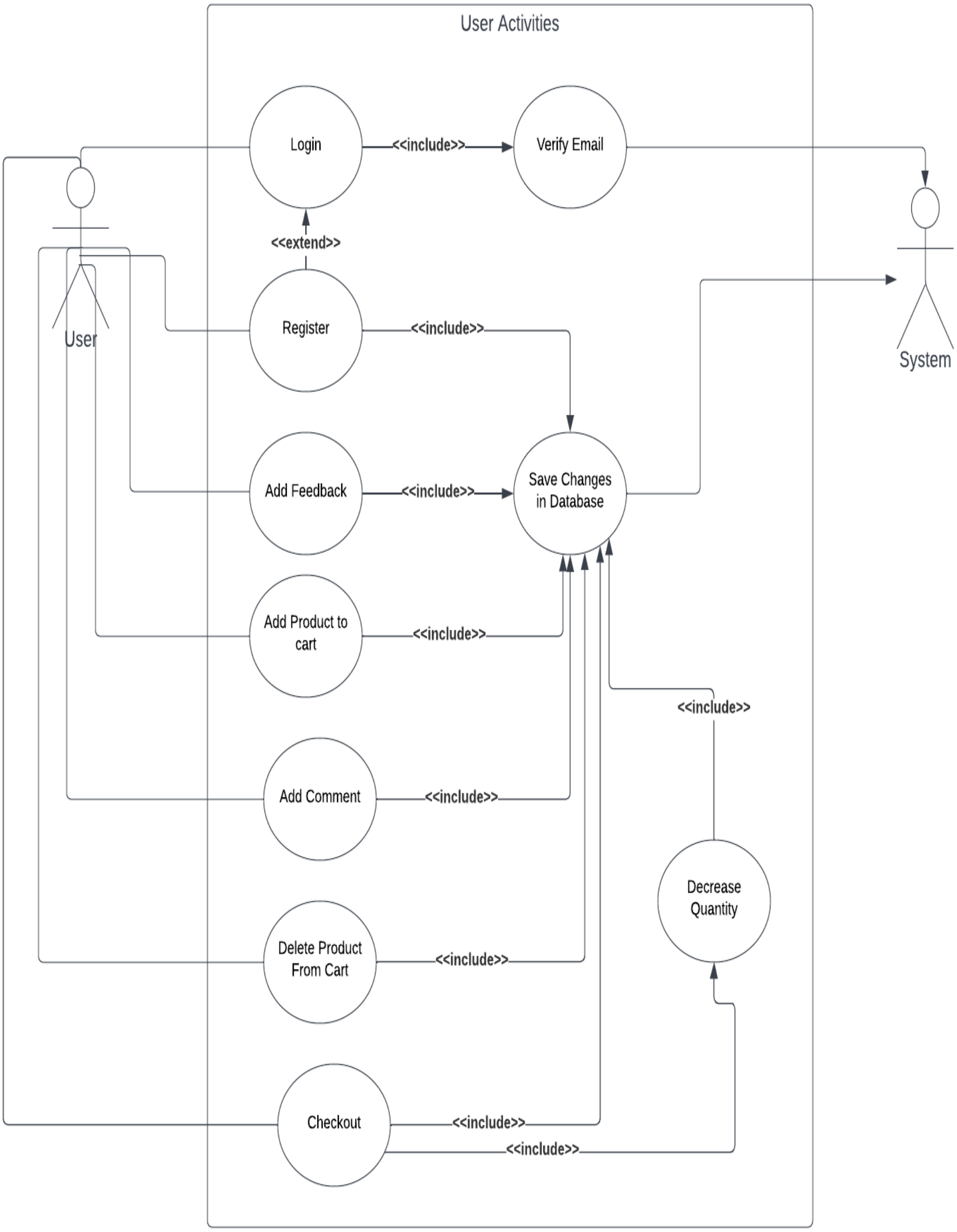
<<IproductData>>
+Index() :IActionResult
+productTable() : IActionResult
+SendDataTOform(product model) : product
Admin
+ ID : int
+ UserName : string
+ Email : string
+ Password : string
+ Adress : string
+ Mobile : int
+ addAmin(db : MarketDBContext , admin : Admin) : void
+ GetAdmins(db:MarketDBContext) :IEnumerable<Admin>
+getAdminByID(db:MarketDBContext ,id:int) : Admin
+deleteAdmin(db:MarketDBContext ,id:int):void
+GetFeedbacks(db:MarketDBContext) :IEnumerable<Feedback>
+GetComments(db:MarketDBContext) :IEnumerable<Comments>
<<IAdminAuthentication>>
+ AdminRegister(admin : Admin) : IActionResult
+AdminLogin(model : Admin) : IActionResult
+AddminRegister() : IActionResult
<<IAdminControl>>
+ CheckAdminRegister(model : Admin) : Boolean
+ CheckAdminEmail(model : Admin) : Boolean
+ CheckAdminPassword(model : Admin) : Boolean
+ AdminSearch(searchWord : string) : IActionResult
<< IAdminModification >>
+ DeleteAdmin(id : int) : IActionResult
+ AddAddmin() : IActionResult
+ Addmin() : IActionResult
<<IshowAdminData>>
+ ShowAdmins() :IActionResult
+RenderAdminData(model : Admin) : void
+getDataByID(id : int) : void
<<IcommentModification>>
+ DeleteComment(int id) : IActionResult
<<IFeedbackModification>>
+ DeleteFeedback(int id) : IActionResult
<< IProductModification>>
+Addproduct() : IActionResult
+ addProductItem(product : product) : Task<IActionResult>
+updateProduct(id : int, model : product) : IActionResult
+updateProductItem(model : product) : Task<IActionResult>
+deleteProduct() : IActionResult
<<ICommentData>>
+ ShowComments() : IActionResult
<<IShowFeedback>>
+ShowFeedback():IActionResult
<<IUserAuthentication>>
+UserRegister(user:User) : IActionResult
+UserLogin(model:User) : IActionResult
+ Register() : IActionResult
<<ISalesReport>>
+SalesReport() : IActionResult
Feedback
+ Id : int
+ Name:string
+ Email:string
+ Title : string
+ Text:string
+ addFeedback(db:MarketDBContext, feedback:Feedback) : void
+deleteFeedback(db:MarketDBContext ,id : int) : void
-getFeedback(id:int, db:MarketDBContext):Feedback
<<ICommentModification>>
+Comments(product : product, id : int) : Task<IActionResult>
+AddComments(comment : Commets) : IActionResult
<<ICart>>
+ AddItemToCart(productId : int ,id : int) :IActionResult
+CartItems() : IActionResult
+RemoveItem(productId : int) : IActionResult
+DecreaseQuantity(productId:int) : IActionResult
<<ICheckout >>
+CheckZeroQuantity(Quantity : int,productId : int):void
+Checkout(userId : int) : IActionResult
+CheckoutDetails(userId : int) : IActionResult
+Confirm(): IActionResult
*
1
product
+ Id:int
+ name : string
+ productCount : int
+ description : string
+ price : int
+ insertTime : DateTime
+ imgSrc : string
+ img : IFormFile
+ category : int
+ ProductComment : List<Comments>
+ getProducts(db:MarketDBContext) :List<product>
+ addProduct(db:MarketDBContext, product:product):void
+ updateProduct(db:MarketDBContext, product : product):void
-getProductByID(db:MarketDBContext, id : int) : product
+ deleteProduct(db:MarketDBContext, id : int) : void
+ searchForProduct(searchWord:string, db:MarketDBContext):List<product>
*
1
CartItem
+ Id : int
+ ProductId : int
+ Product : product
+ Quantity : int
+ userId : int
+ CheckProductInCart(db:MarketDBContext ,productId : int) : CartItem
+ SaveNewProduct(db:MarketDBContext, cartItem : CartItem) : void
+ SaveChanges(db:MarketDBContext) : void
+ ListOfItems(db:MarketDBContext, currentId : int) : List<string>
+ GetCartItems(db:MarketDBContext, currentId : int) : List<CartItem>
+ GetItem(db:MarketDBContext, productId : int) : CartItem
+ GetConfirmData(db:MarketDBContext) : List<CartItem>
+ ClearCart(db:MarketDBContext, currentId : int) : void
1
1
User
+ ID : int
+ UserName : string
+ Email : string
+ Password : string
+ Adress : string
+ Mobile : int
+ getUsers(db:MarketDBContext) :List<User>
+ CheckRegister(user:User, db:MarketDBContext):Boolean
+ addUser(user : User ,db:MarketDBContext):void
+getUserByID(db:MarketDBContext ,id : int) : User
+ deleteUser(id : int ,db : MarketDBContext) : void
<<IUserControl>>
+CheckUserEmail(model : User) : Boolean
+CheckUserPassword(model : User) : Boolean
<<IUserData>>
+RenderUserData(model : User) : Void
+ UserSearch(searchWord : string) : IActionResult
<<IProductData>>
+getDataByID(id:int):void
+ShowProduct(id : int) : IActionResult
1
1
*
1
*
1
<<interface>>
Interface
+ operation1(params):returnType
- operation2(params)
- operation3()
Comments
+ Id : int
+ Text : string
+ productCommentId : int
+ productComment : product
+ addComment(db:MarketDBContext ,comment : Comments) : void
+ deleteComment(db:MarketDBContext, id : int) : void
-getComment(id:int, db:MarketDBContext):Comments
-getProductName(db:MarketDBContext, PID : int) : string
-setCommentDetails(db:MarketDBContext) :IEnumerable<ProductJoinComment>
+ ShowComment(db:MarketDBContext) :IEnumerable<ProductJoinComment>
1
1
ProductJoinComment
+ Product : product
+ Comment : Comments
SalesReport
+ ID : int
+ productName : string
+ price : int
+ Quantity : int
+ totalPayment : int
+ user : User
+ userID : int
+ SaveSalesData(db:MarketDBContext, salesData:List<CartItem>,currentId:int) : void
+ GetSalesData(db : MarketDBContext) : List<SalesReport>
<<IFeedbackModification>>
+Feedback() :IActionResult
+AddFeedback(feedback : Feedback) : IActionResult
ErrorViewModel
+ RequestId : string
+ ShowRequestId(RequestId:string):bool
MarketDBContext
+ admins : DbSet<Admin>
+ users : DbSet<User>
+ feedbacks : DbSet<Feedback>
+ products : DbSet<product>
+ comments : DbSet<Comments>
+ cartItems : DbSet<CartItem>
+ sales : DbSet<SalesReport>

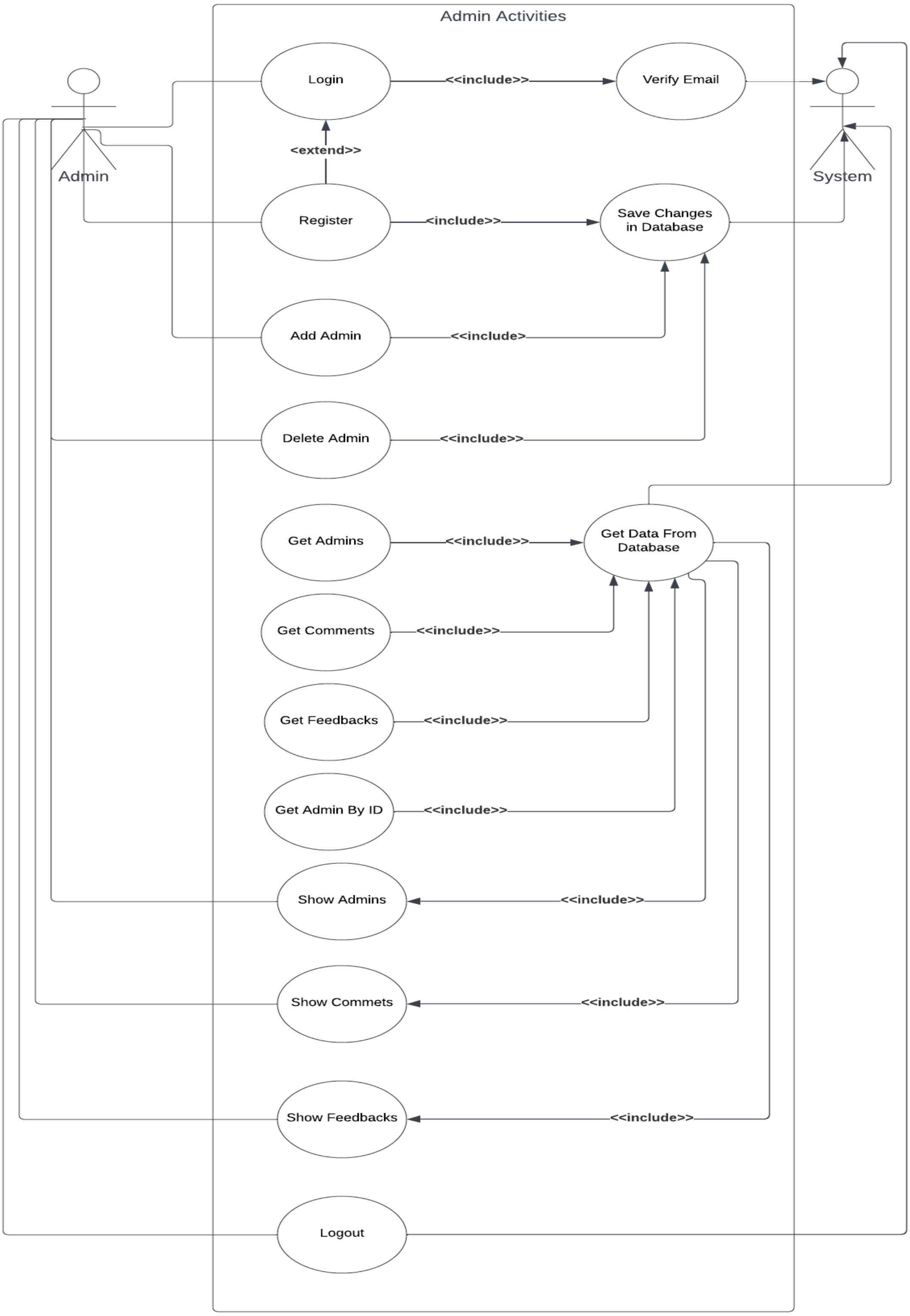
Activity Diagram




Admin
System
Have Email
Not Having Email
Yes
No
Login
Register
Save in the Database
Verify Email
Valid Email
Invalid Email
Yes
No
Add Admin
Delete Admin
Get Admins
Get Comments
Get Admin By ID
Get Feedbacks
Save Changes in Database
Get Data from Database
Show Feedbacks
Show Admins
Show Comments
Show Admin
Another Operation ?
Yes
No
Exit
Logout

Use case Diagram




Admin Activities
Admin
System
Login
<<include>>
Verify Email
<extend>>
Register
<include>>
Save Changes in Database
Add Admin
<<include>
Delete Admin
<<include>>
Get Admins
<<include>>
Get Data From Database
Get Comments
<<include>>
Get Feedbacks
<<include>>
Get Admin By ID
<<include>>
Show Admins
<<include>>
Show Commets
<<include>>
Show Feedbacks
<<include>>
Logout