Faculty Of Computers and Artificial Intelligence

Helwan University

Inventory management system

1-Beshoy Ibrahim Asham

202000215 / 3rd Year / CS Major

2-Thaowpsta Saiid Aziz

202000233/ 3rd Year / CS Major

3-Ziad Mazhar Mahmoud

202000362/ 3rd Year / CS Major

4-Tassneam Mohsen Samy

202000224/ 3rd Year / CS Major

5-Batool Sherif Mohamed

202000195/ 3rd Year / CS Major

6- Ali Mohamed Mohamed

202000573 / 3rd Year / CS Major

Source Code

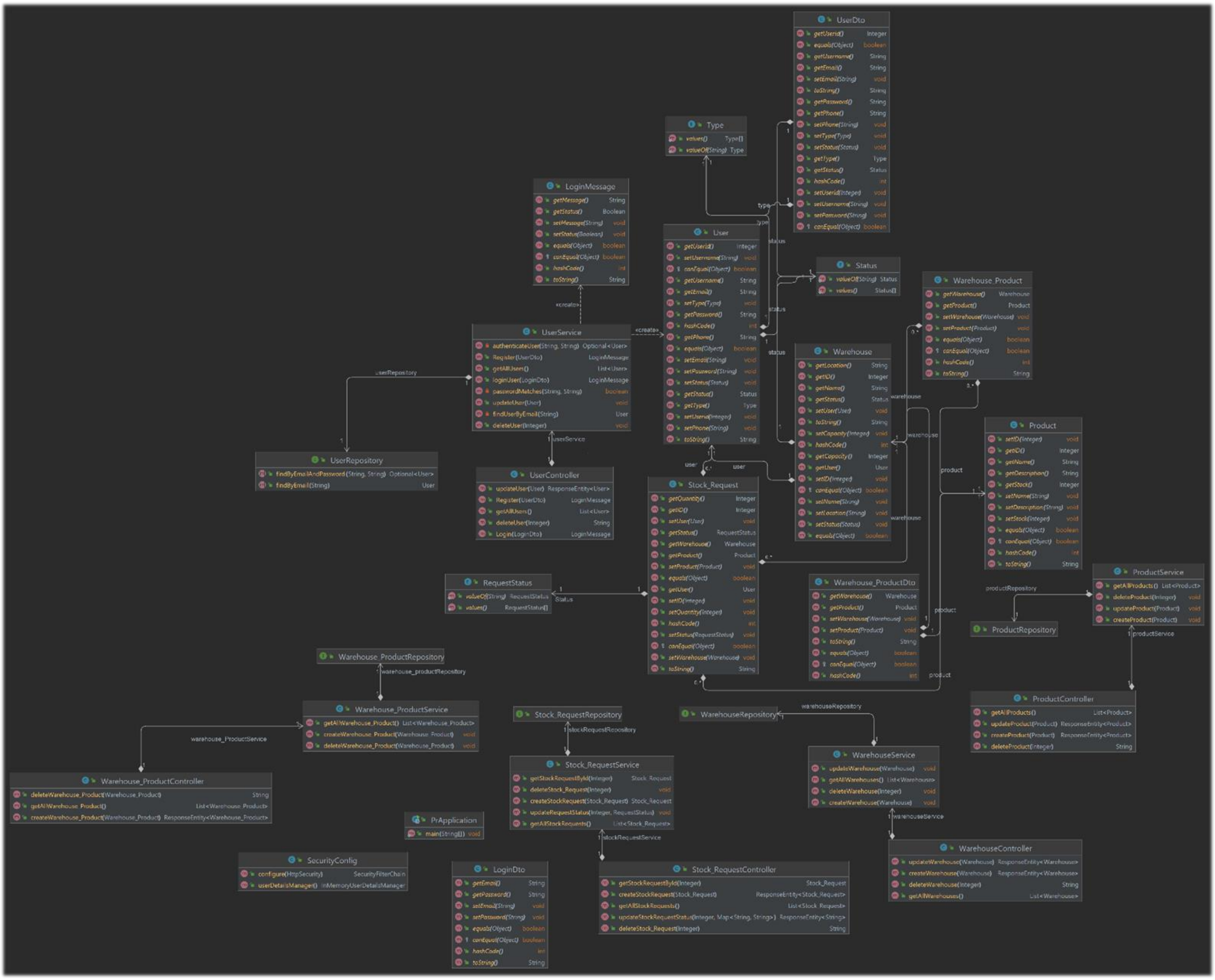
Inventory management system

TABLE OF CONTENTS

Functional Requirements:

FUNCTION NAME	FUNCTION DESCRIPTION
Register	Allows Admins/Supervisors to Create a new account
Login	Allows Admins/Supervisors to Login to the Account they created earlier
getAllUsers	Allows Admins to fetch all users
updateUser	Allows Admins to update user's info
deleteUser	Allows Admins to delete a user
getAllProducts	Allows Admins/Supervisors to fetch all products
createProduct	Allows Admins to Create a new product
updateProduct	Allows Admins to update product's info
deleteProduct	Allows Admins to delete a product
createStockRequest	Allows Supervisors to request stock
getAllStockRequests	Allows Admins to fetch all requests
updateRequestStatus	Allows Admins to update the request status
deleteStock_Request	Allows Admins to delete a request
getAllWarehouse_Product	Allows Admins to fetch all warehouses and their products
createWarehouse_Product	Allows Admins to create a specific product to a specific warehouse
deleteWarehouse_Product	Allows Admin to delete a specific warehouse with its products
getAllWarehouses	Allows Admins to fetch all warehouses
createWarehouse	Allows Admins to create a warehouse
updateWarehouse	Allows Admins to update warehouse's info
deleteWarehouse	Allows Admins to delete a warehouse

Class Diagram:

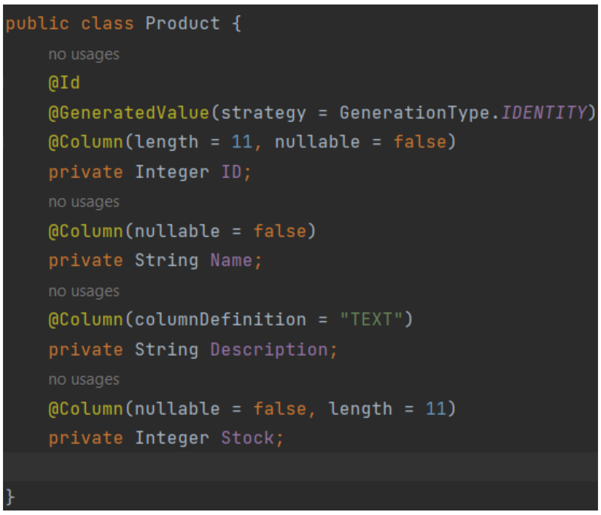


OCL (Object Constraint Language):

Classes:

- **Product**
- **User**
- **Stock_Request**
- **Warehouse**
- **Warehouse_Product**

Product Class:



```java
public class Product {
    no usages
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(length = 11, nullable = false)
    private Integer ID;
    no usages
    @Column(nullable = false)
    private String Name;
    no usages
    @Column(columnDefinition = "TEXT")
    private String Description;
    no usages
    @Column(nullable = false, length = 11)
    private Integer Stock;

}
```

Context: Product

invariant: self.ID.size() <= 11

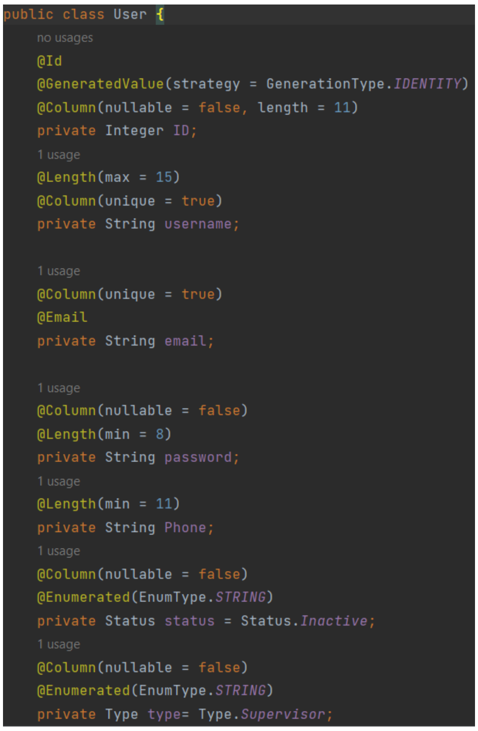
& not self.ID.oclIsUndefined()

invariant: not self.Name.oclIsUndefined()

invariant: not self.Stock.oclIsUndefined()

& self.Stock.size() <=11

User Class:



```java
public class User {
    no usages
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(nullable = false, length = 11)
    private Integer ID;
    1 usage
    @Length(max = 15)
    @Column(unique = true)
    private String username;

    1 usage
    @Column(unique = true)
    @Email
    private String email;

    1 usage
    @Column(nullable = false)
    @Length(min = 8)
    private String password;
    1 usage
    @Length(min = 11)
    private String Phone;
    1 usage
    @Column(nullable = false)
    @Enumerated(EnumType.STRING)
    private Status status = Status.Inactive;
    1 usage
    @Column(nullable = false)
    @Enumerated(EnumType.STRING)
    private Type type= Type.Supervisor;
```

Context: User

Invariant: not self.ID.oclIsUndefined()

& self.ID.size() <= 11

invariant: self.username.size() <= 15

& User.allInstances()->isUnique(username)

Invariant: User.allInstances()->isUnique(email)

&self.email.matches('.+@.+\\..+')

Invariant: not self.password.oclIsUndefined()

& self.password.size() >=8

Invariant: self.Phone.size() <= 11

Invariant: not self.status.oclIsUndefined()

& self.status in {Status::Active, Status::Inactive}

init:

self.status = Status::Inactive

Invariant: not self.type.oclIsUndefined()

& self.type in {Type::Admin, Type::Supervisor}

init

self.type=Type::Supervisor

Stock_Request Class:

Context: Stock_Request

Invariant: not self.ID.oclIsUndefined()

& self.ID.size() <= 11

Invariant: self.Quantity.size() <= 11

& self.Quantity.oclIsUndefined()

Invariant: not self.Status.oclIsUndefined()

& self.Status in {RequestStatus::Approved

, RequestStatus::Pending

, RequestStatus::Rejected}

init

self.Status in RequestStatus::Pending

invariant: not self.warehouse.oclIsUndefined()

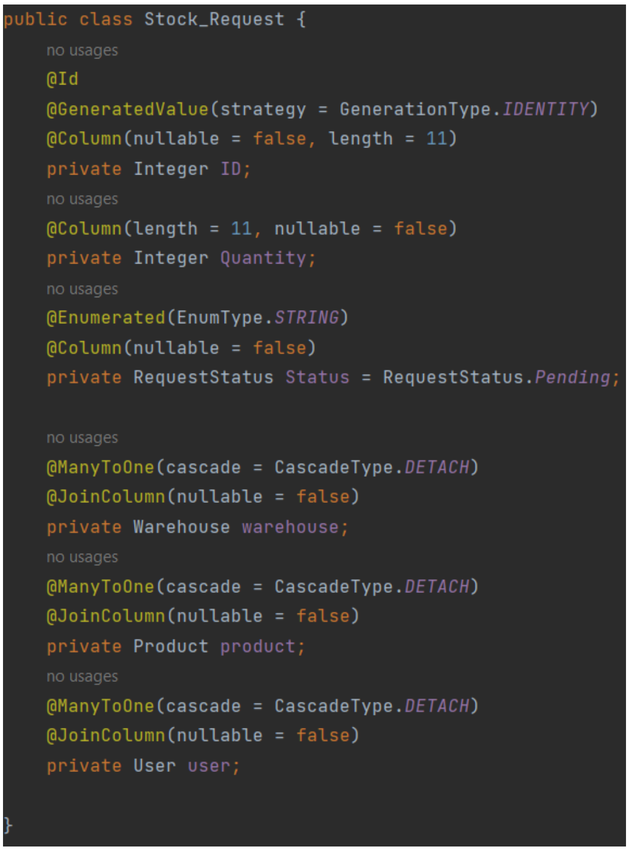
& self.warehouse.requests->includes(self)

Invariant: not self.product.oclIsUndefined()

& self.product.requests->includes(self)

Invariant: not self.user.oclIsUndefined()

& self.user.requests->includes(self)



```
public class Stock_Request {
    no usages
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(nullable = false, length = 11)
    private Integer ID;
    no usages
    @Column(length = 11, nullable = false)
    private Integer Quantity;
    no usages
    @Enumerated(EnumType.STRING)
    @Column(nullable = false)
    private RequestStatus Status = RequestStatus.Pending;

    no usages
    @ManyToOne(cascade = CascadeType.DETACH)
    @JoinColumn(nullable = false)
    private Warehouse warehouse;
    no usages
    @ManyToOne(cascade = CascadeType.DETACH)
    @JoinColumn(nullable = false)
    private Product product;
    no usages
    @ManyToOne(cascade = CascadeType.DETACH)
    @JoinColumn(nullable = false)
    private User user;

}
```

Warehouse Class:

Context: Warehouse

Invariant: not self.ID.oclIsUndefined()

& self.ID.size() <= 11

Invariant: not self.Name.oclIsUndefined()

Invariant: Warehouse.allInstances()->isUnique(Location)

Invariant: not self.status.oclIsUndefined()

& self.status in {Status::Active, Status::Inactive}

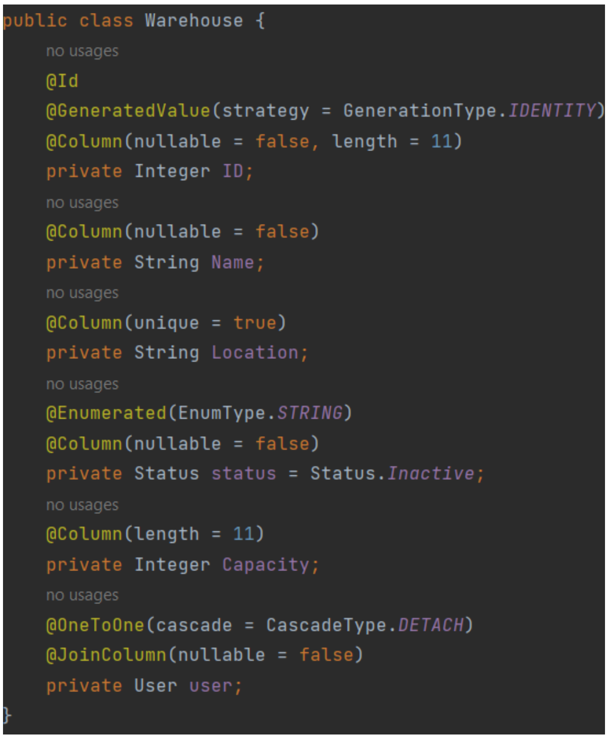
init:

self.status = Status::Inactive

Invariant: self.Capacity.size() <= 11

Invariant: not self.user.oclIsUndefined()

& self.user.warehouse->includes(self)



```
public class Warehouse {
    no usages
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(nullable = false, length = 11)
    private Integer ID;
    no usages
    @Column(nullable = false)
    private String Name;
    no usages
    @Column(unique = true)
    private String Location;
    no usages
    @Enumerated(EnumType.STRING)
    @Column(nullable = false)
    private Status status = Status.Inactive;
    no usages
    @Column(length = 11)
    private Integer Capacity;
    no usages
    @OneToOne(cascade = CascadeType.DETACH)
    @JoinColumn(nullable = false)
    private User user;
}
```

Warehouse_Product Class:

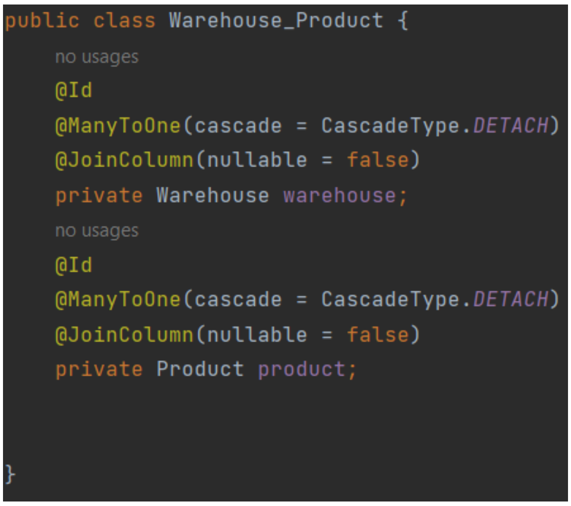
Context: Warehouse_Product

Invariant: not self.warehouse.oclIsUndefined()

& self.warehouse.warehouse_Product->includes(self)

Invariant: not self.product.oclIsUndefined()

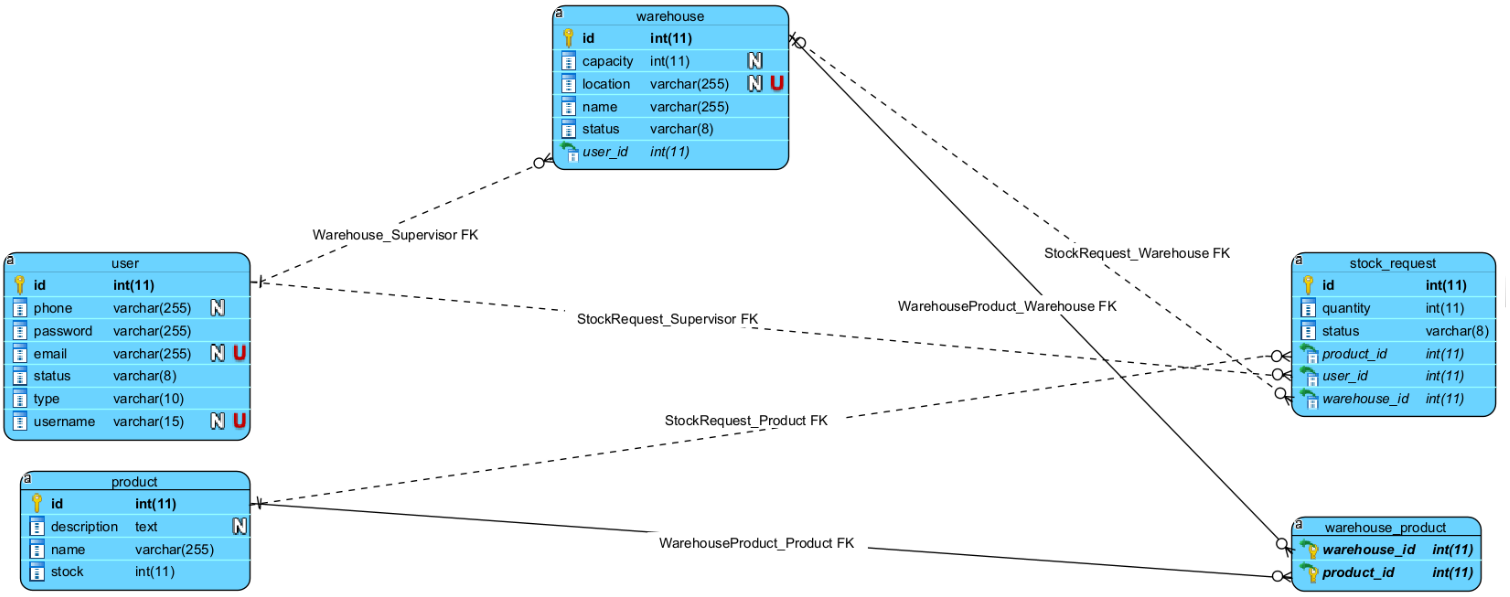
& self.product.warehouse_Product->includes(self)



```
public class Warehouse_Product {
    no usages
    @Id
    @ManyToOne(cascade = CascadeType.DETACH)
    @JoinColumn(nullable = false)
    private Warehouse warehouse;
    no usages
    @Id
    @ManyToOne(cascade = CascadeType.DETACH)
    @JoinColumn(nullable = false)
    private Product product;


}
```

Database Specification: ERD Tables:


Warehouse_Supervisor FK
StockRequest_Supervisor FK
StockRequest_Warehouse FK
WarehouseProduct_Warehouse FK
StockRequest_Product FK
WarehouseProduct_Product FK

warehouse		
id	**int(11)**	
capacity	int(11)	N
location	varchar(255)	N U
name	varchar(255)	
status	varchar(8)	
user_id	*int(11)*	

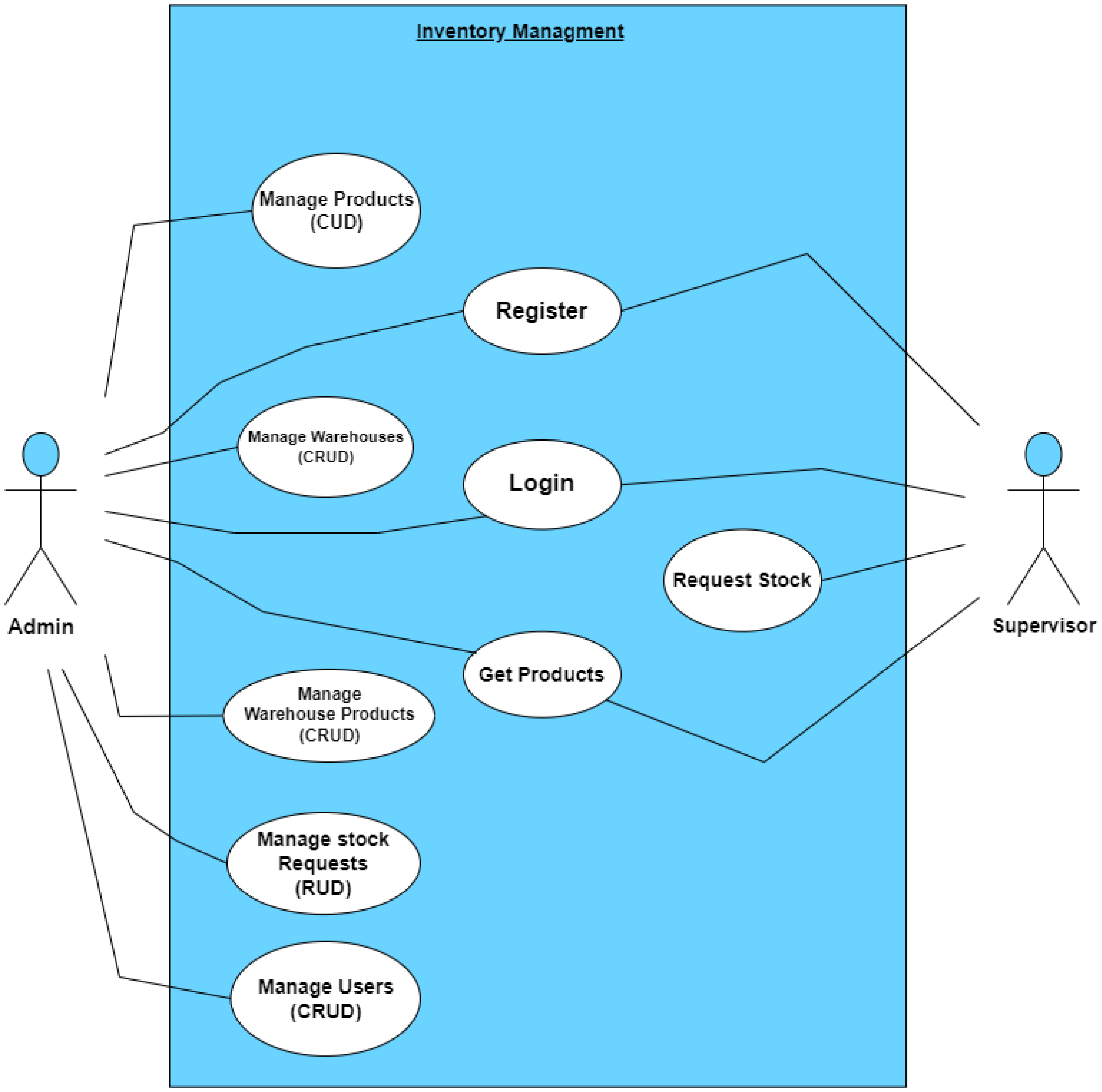
user		
id	**int(11)**	
phone	varchar(255)	N
password	varchar(255)	
email	varchar(255)	N U
status	varchar(8)	
type	varchar(10)	
username	varchar(15)	N U

stock_request	
id	**int(11)**
quantity	int(11)
status	varchar(8)
product_id	*int(11)*
user_id	*int(11)*
warehouse_id	*int(11)*

product		
id	**int(11)**	
description	text	N
name	varchar(255)	
stock	int(11)	

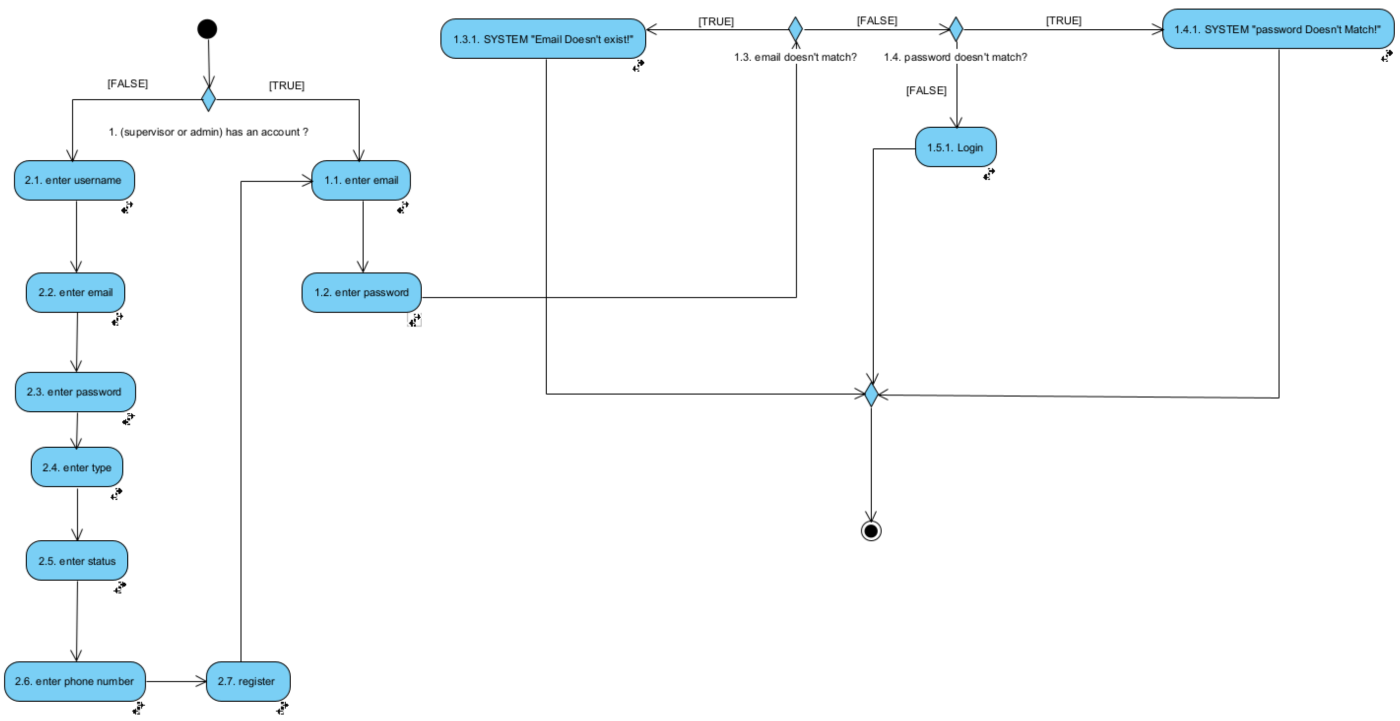
warehouse_product	
warehouse_id	***int(11)***
product_id	***int(11)***

Use Case Diagram:

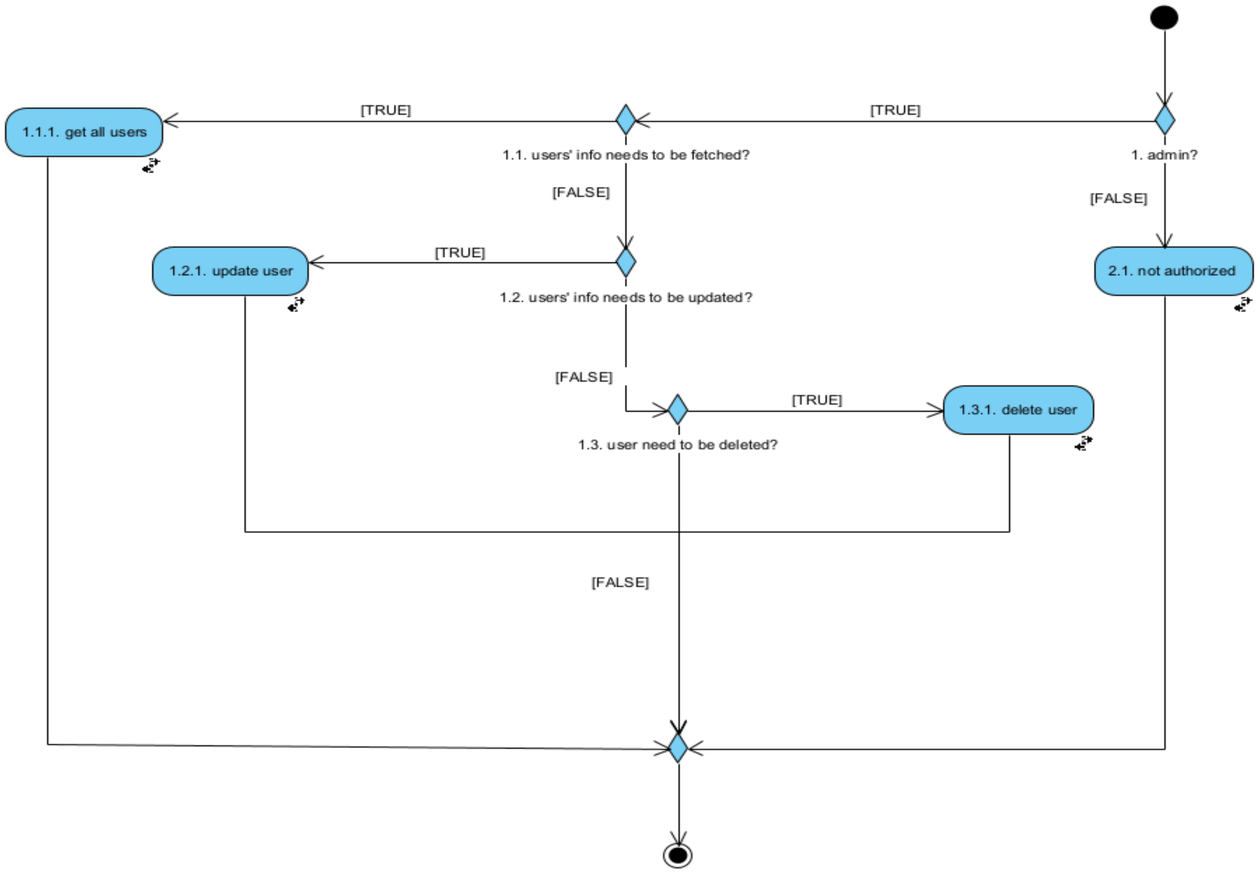


Activity Diagrams:

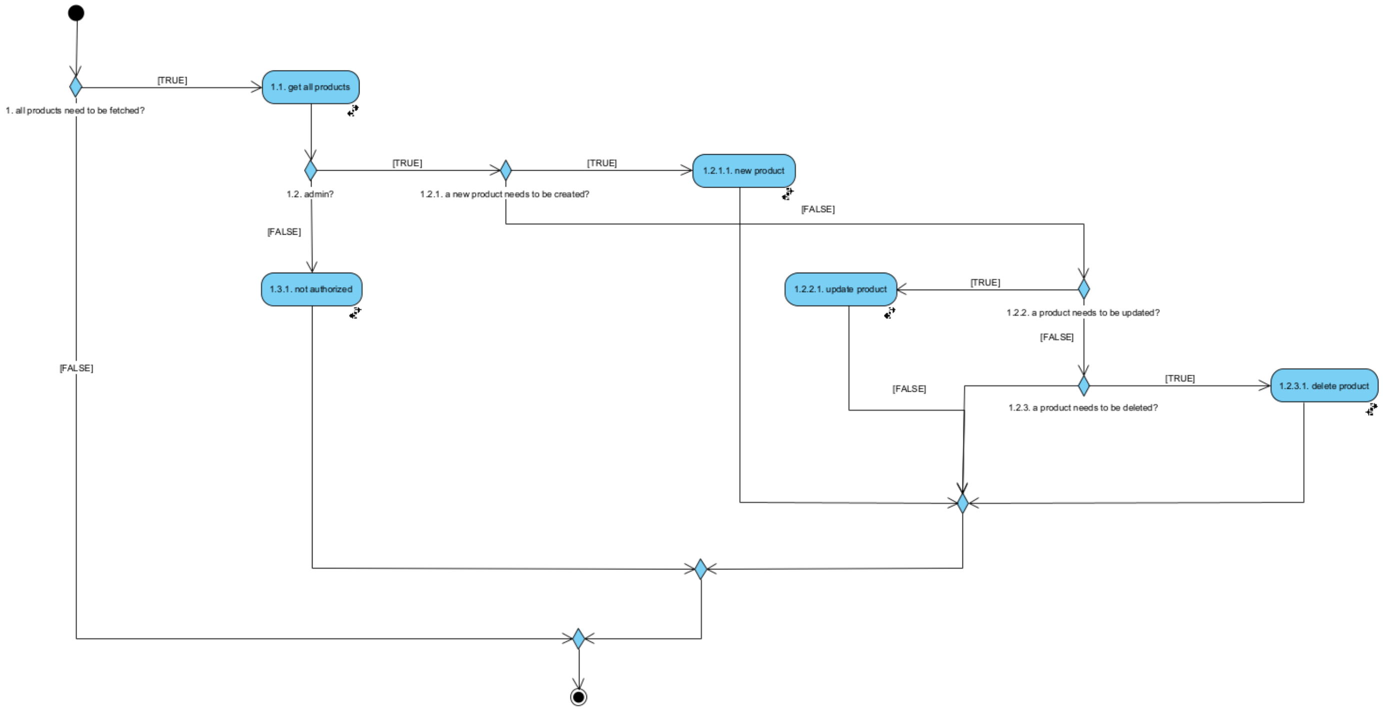
Login&Registration:



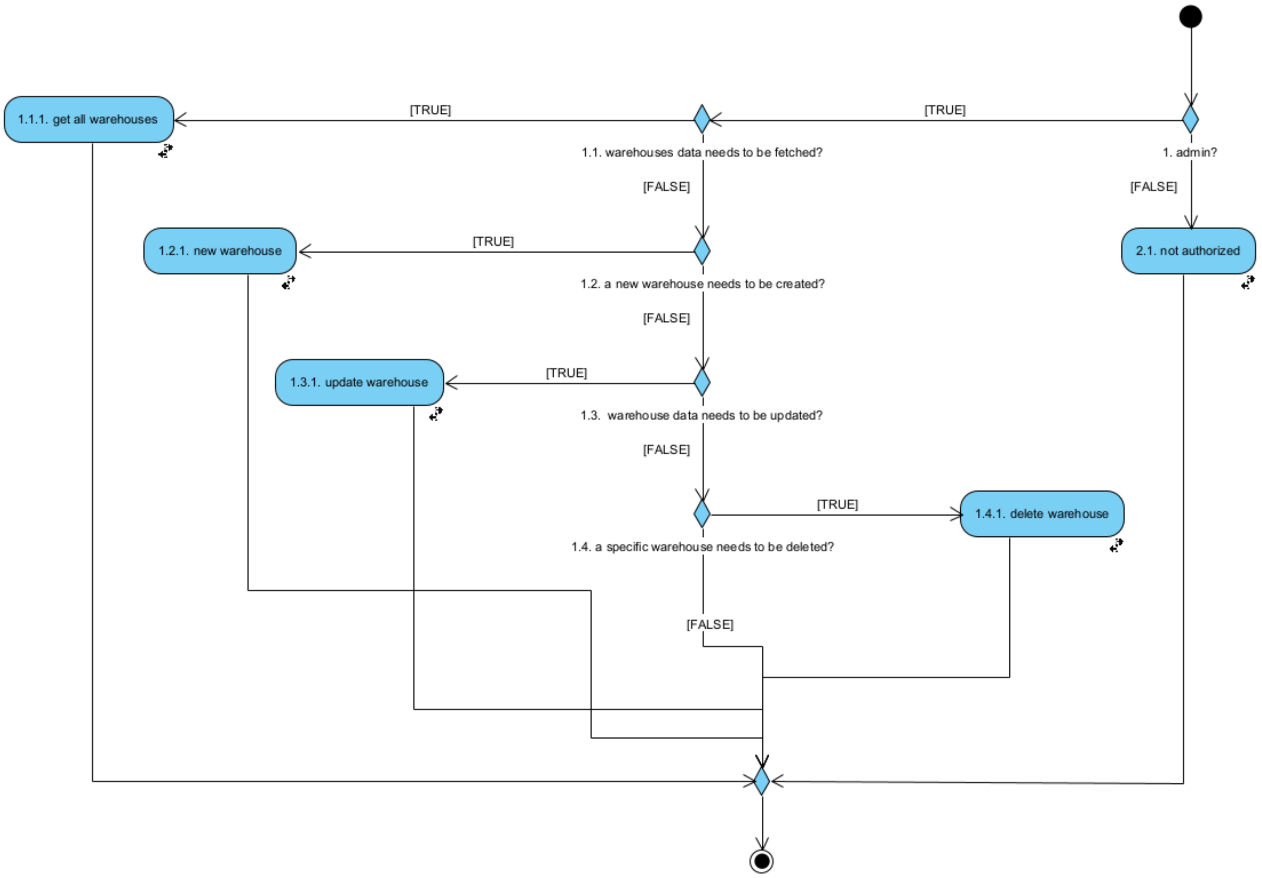
Manage users:



Manage products:

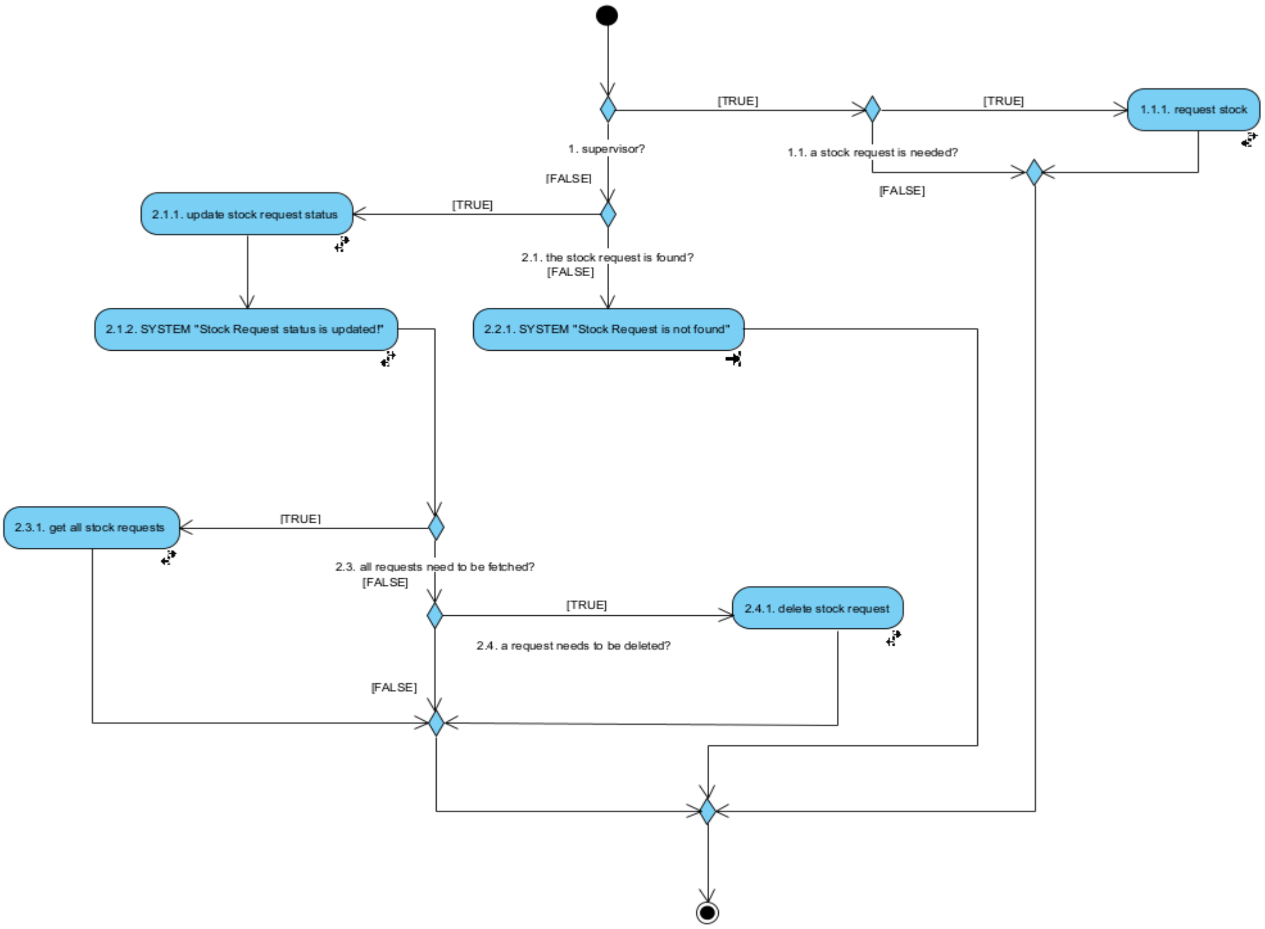


Manage warehouses:

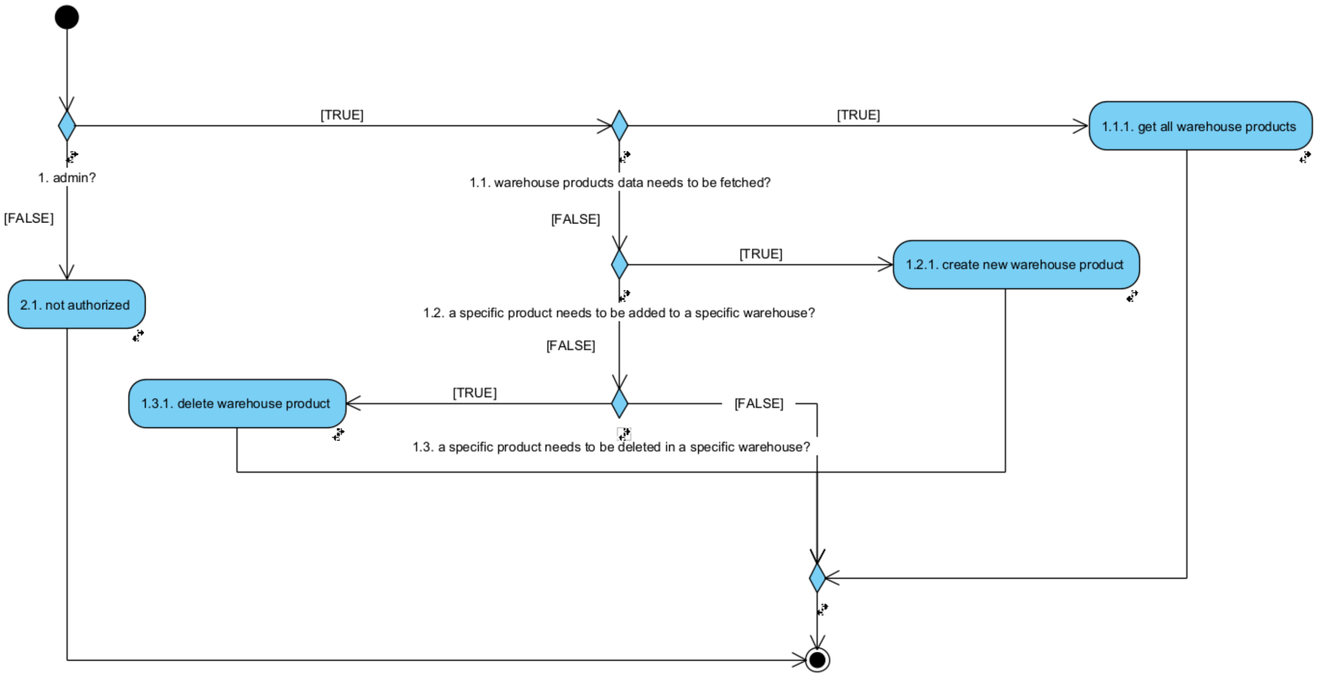


1.1.1. get all warehouses
[TRUE]
[TRUE]
1.1. warehouses data needs to be fetched?
1. admin?
[FALSE]
[FALSE]
1.2.1. new warehouse
[TRUE]
2.1. not authorized
1.2. a new warehouse needs to be created?
[FALSE]
1.3.1. update warehouse
[TRUE]
1.3. warehouse data needs to be updated?
[FALSE]
[TRUE]
1.4.1. delete warehouse
1.4. a specific warehouse needs to be deleted?
[FALSE]

Manage stock requests:

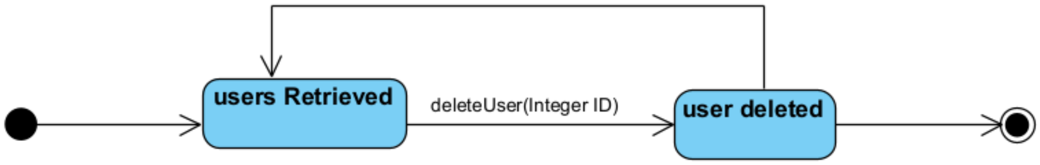

[TRUE]
[TRUE]
1.1.1. request stock
1. supervisor?
1.1. a stock request is needed?
[FALSE]
[FALSE]
[TRUE]
2.1.1. update stock request status
2.1. the stock request is found?
[FALSE]
2.1.2. SYSTEM "Stock Request status is updated!"
2.2.1. SYSTEM "Stock Request is not found"
2.3.1. get all stock requests
[TRUE]
2.3. all requests need to be fetched?
[FALSE]
[TRUE]
2.4.1. delete stock request
2.4. a request needs to be deleted?
[FALSE]

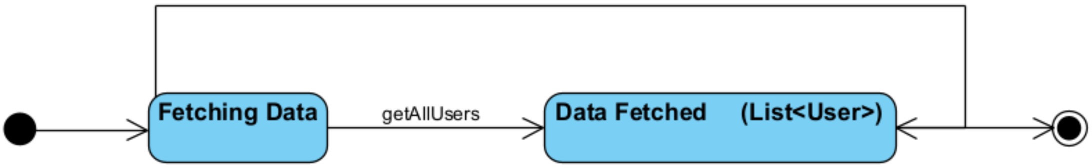
Manage warehouse products:

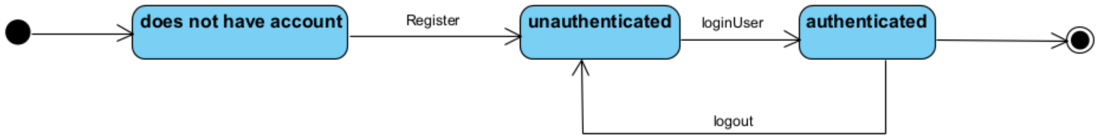

[TRUE]
[TRUE]
1.1.1. get all warehouse products
1. admin?
1.1. warehouse products data needs to be fetched?
[FALSE]
[FALSE]
[TRUE]
1.2.1. create new warehouse product
2.1. not authorized
1.2. a specific product needs to be added to a specific warehouse?
[FALSE]
[TRUE]
1.3.1. delete warehouse product
[FALSE]
1.3. a specific product needs to be deleted in a specific warehouse?

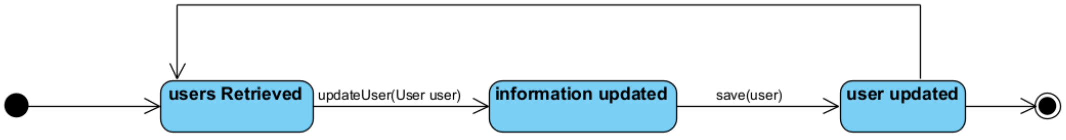
State Diagrams:

Manage users:

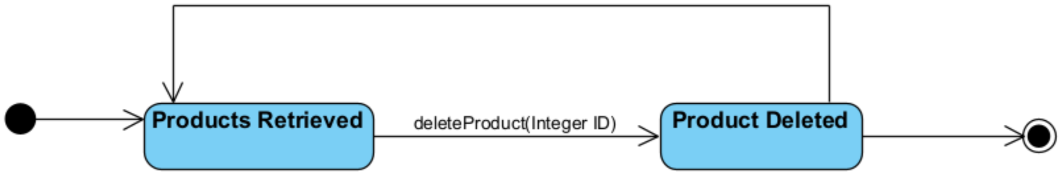


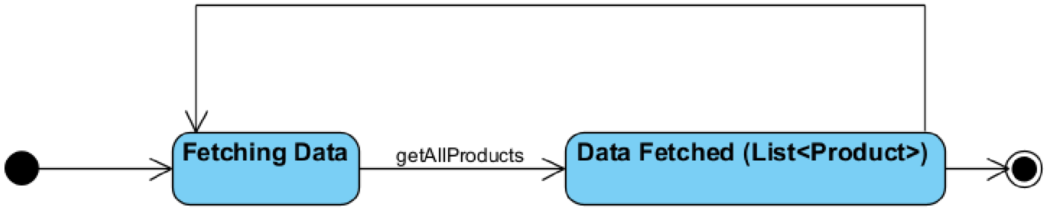


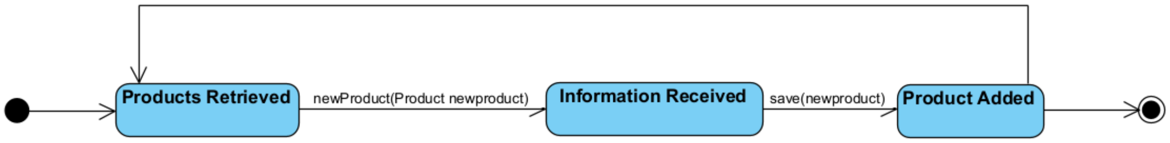


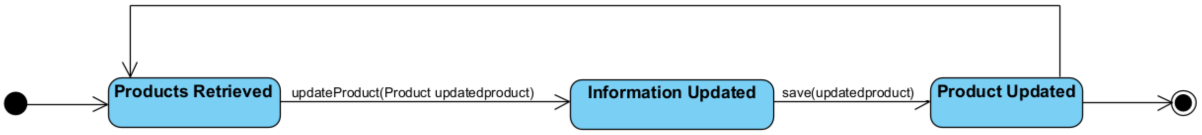


Manage products:

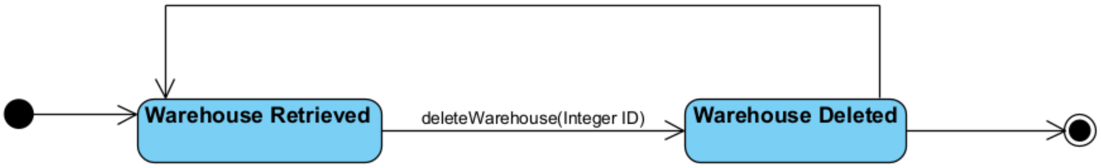


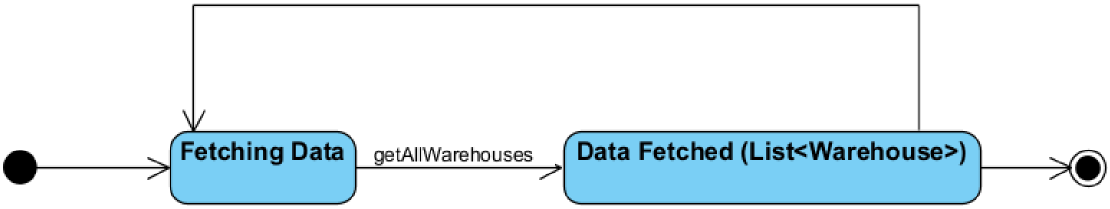


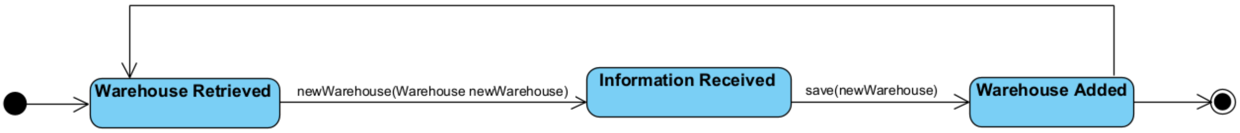


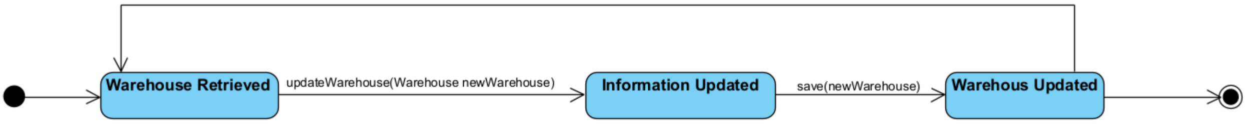


Manage warehouses:

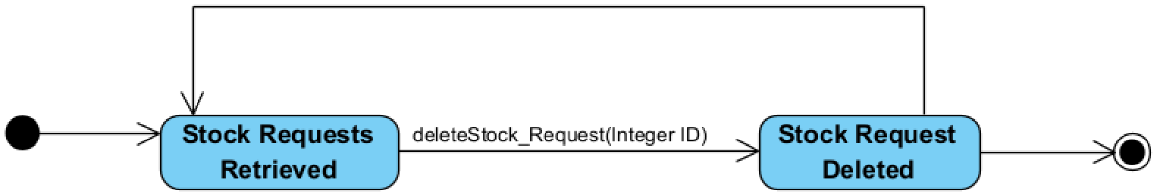


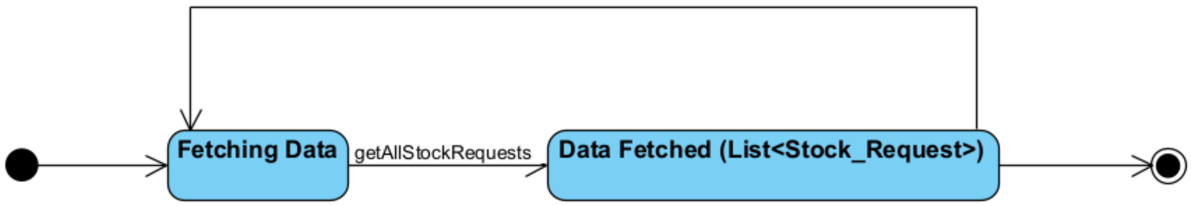


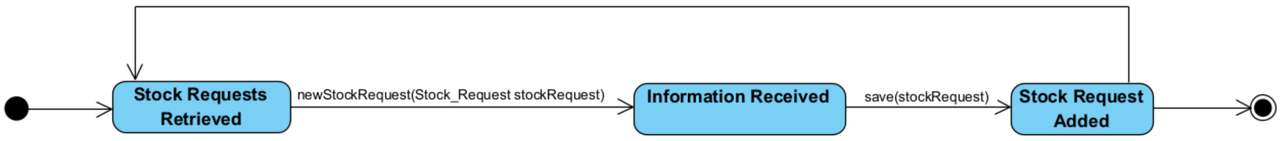


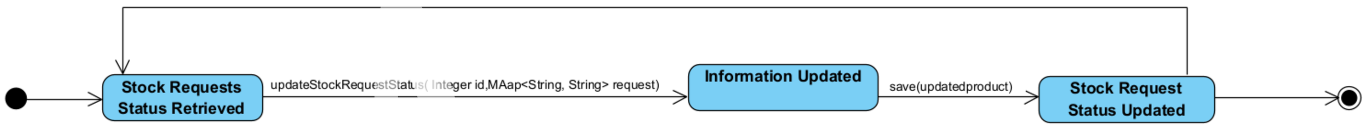


Manage stock requests:

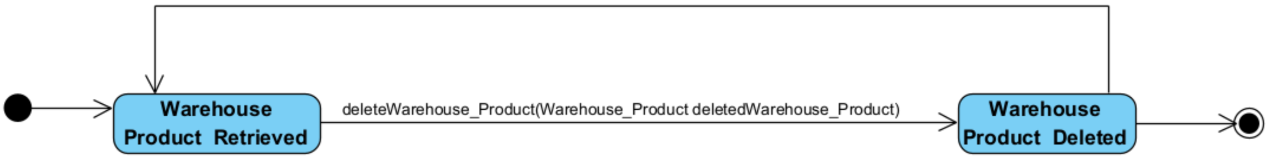


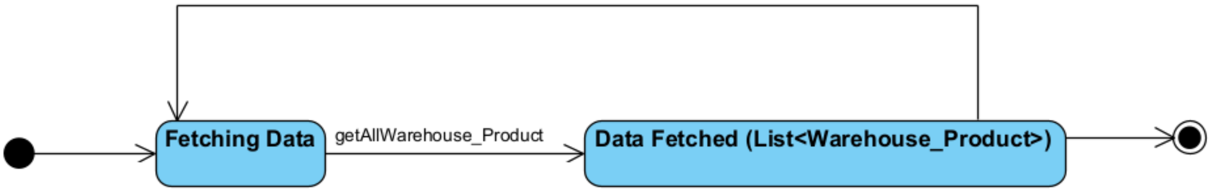


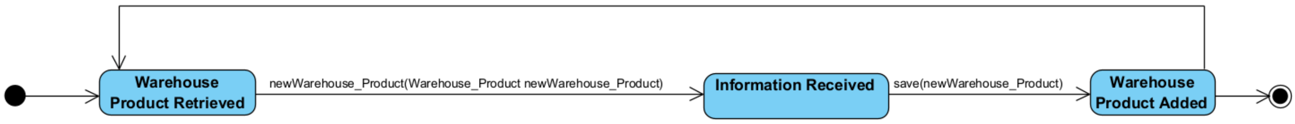



Stock Requests Status Retrieved
updateStockRequestStatus(Integer id,MAap<String, String> request)
Information Updated
save(updatedproduct)
Stock Request Status Updated

Manage warehouse products:

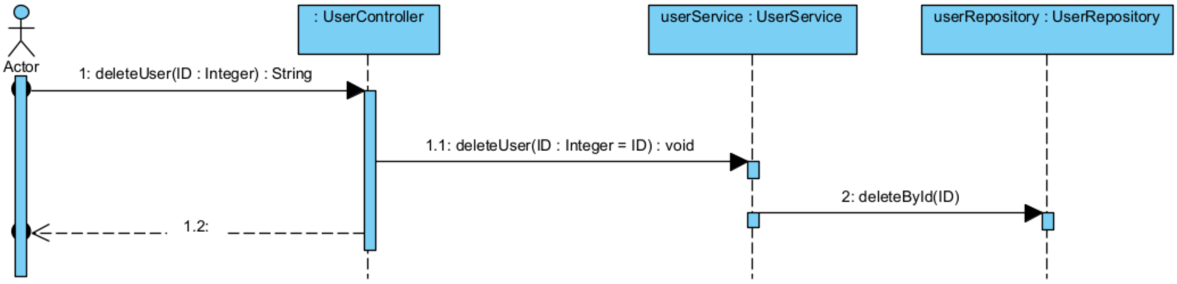

Warehouse Product Retrieved
deleteWarehouse_Product(Warehouse_Product deletedWarehouse_Product)
Warehouse Product Deleted

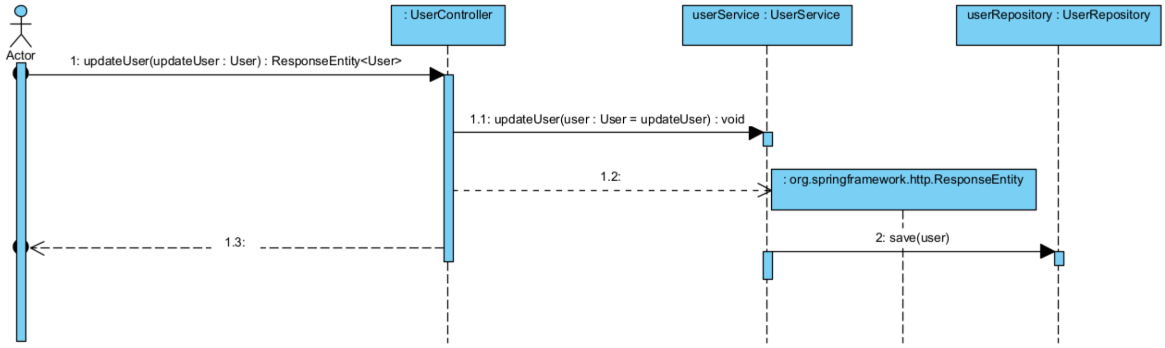


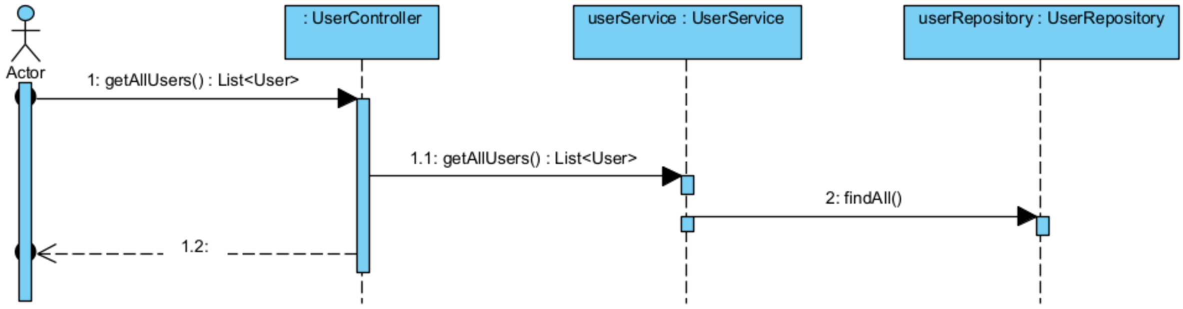

Warehouse Product Retrieved
newWarehouse_Product(Warehouse_Product newWarehouse_Product)
Information Received
save(newWarehouse_Product)
Warehouse Product Added

Sequence Diagrams:

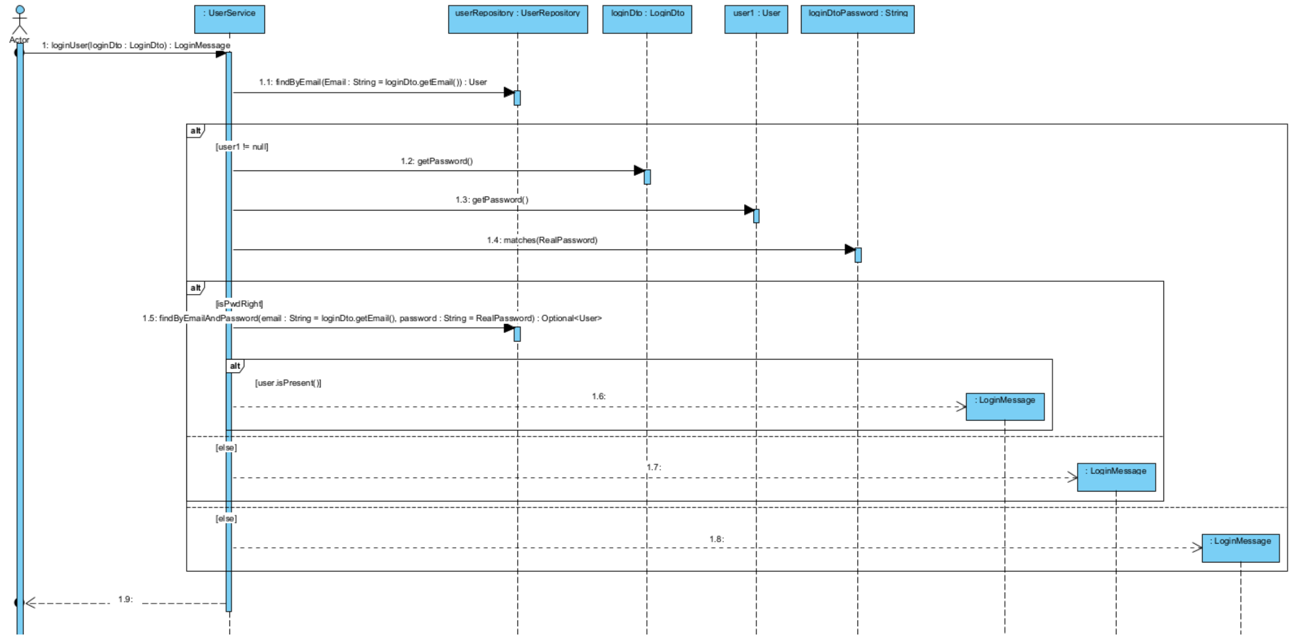
Manage users:


Actor
: UserController
userService : UserService
userRepository : UserRepository
1: deleteUser(ID : Integer) : String
1.1: deleteUser(ID : Integer = ID) : void
2: deleteById(ID)
1.2:

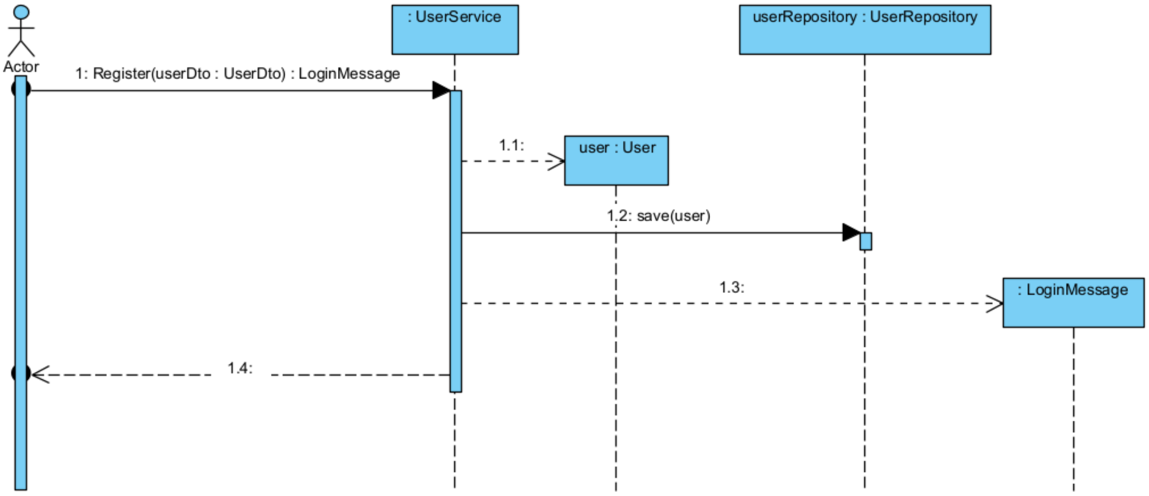

Actor
: UserController
userService : UserService
userRepository : UserRepository
1: updateUser(updateUser : User) : ResponseEntity<User>
1.1: updateUser(user : User = updateUser) : void
1.2:
: org.springframework.http.ResponseEntity
2: save(user)
1.3:


Actor
: UserController
userService : UserService
userRepository : UserRepository
1: getAllUsers() : List<User>
1.1: getAllUsers() : List<User>
2: findAll()
1.2:

Login&Registration:

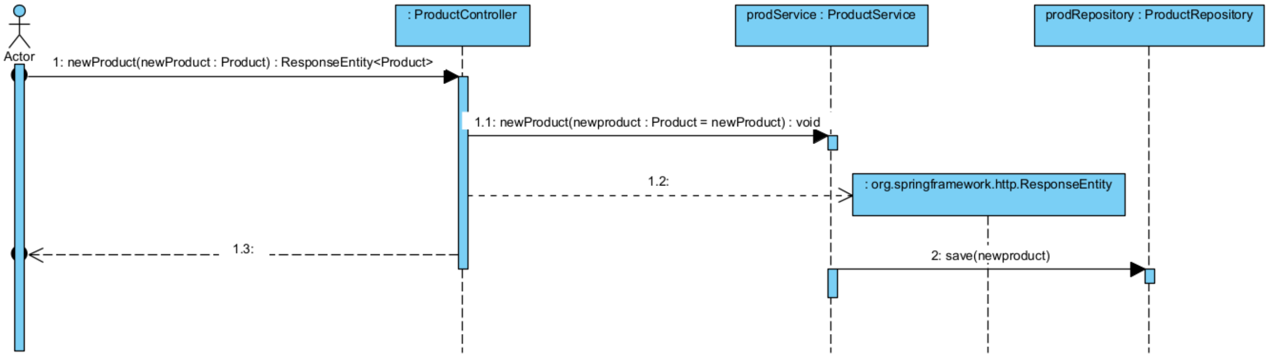


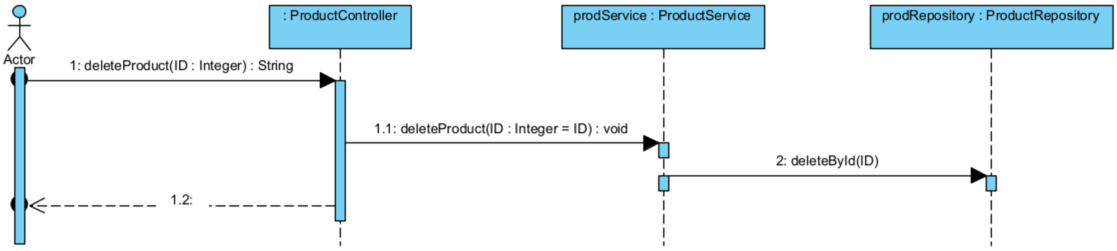
Actor
: UserService
userRepository : UserRepository
loginDto : LoginDto
user1 : User
loginDtoPassword : String
1: loginUser(loginDto : LoginDto) : LoginMessage
1.1: findByEmail(Email : String = loginDto.getEmail()) : User
alt
[user1 != null]
1.2: getPassword()
1.3: getPassword()
1.4: matches(RealPassword)
alt
[isPwdRight]
1.5: findByEmailAndPassword(email : String = loginDto.getEmail(), password : String = RealPassword) : Optional<User>
alt
[user.isPresent()]
1.6:
: LoginMessage
[else]
1.7:
: LoginMessage
[else]
1.8:
: LoginMessage
1.9:

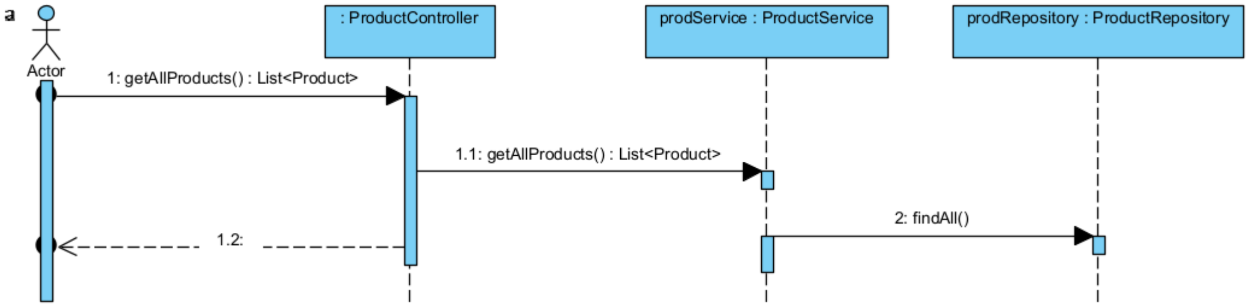


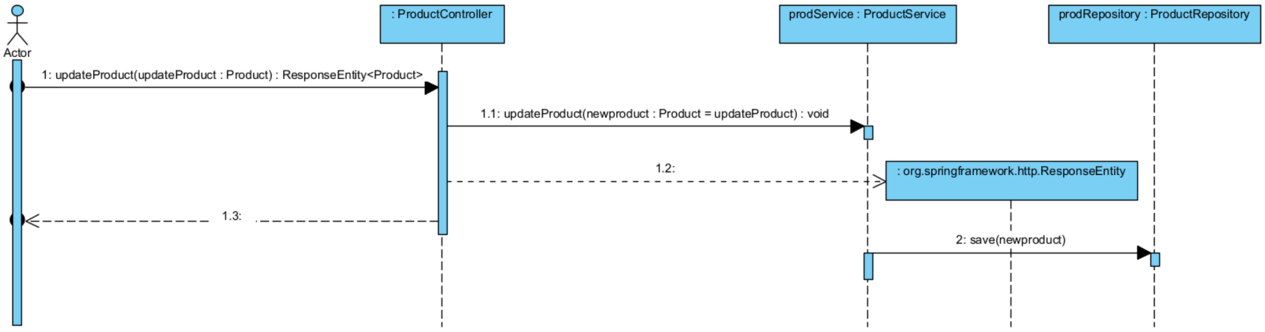
Actor
: UserService
userRepository : UserRepository
1: Register(userDto : UserDto) : LoginMessage
1.1:
user : User
1.2: save(user)
1.3:
: LoginMessage
1.4:

Manage products:

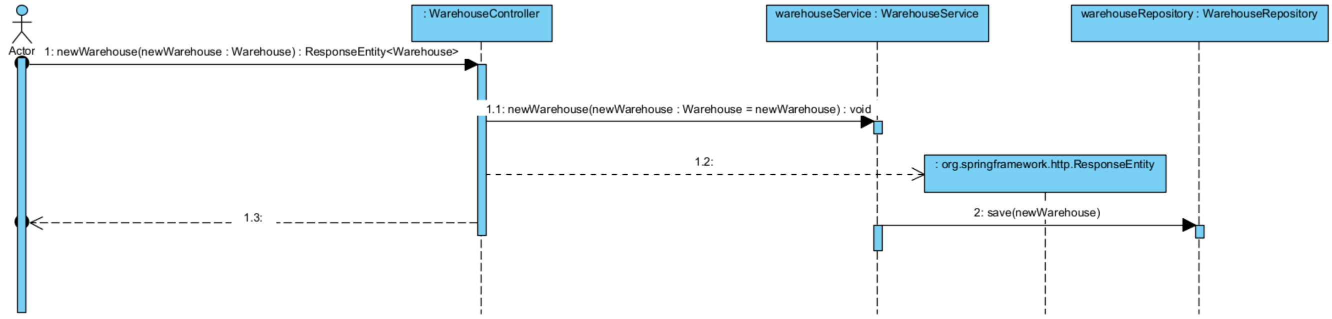

Actor
: ProductController
prodService : ProductService
prodRepository : ProductRepository
1: newProduct(newProduct : Product) : ResponseEntity<Product>
1.1: newProduct(newproduct : Product = newProduct) : void
1.2:
: org.springframework.http.ResponseEntity
1.3:
2: save(newproduct)

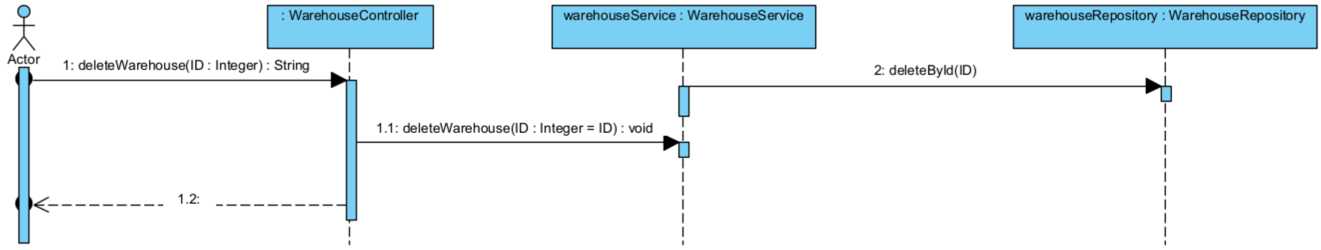

Actor
: ProductController
prodService : ProductService
prodRepository : ProductRepository
1: deleteProduct(ID : Integer) : String
1.1: deleteProduct(ID : Integer = ID) : void
2: deleteById(ID)
1.2:

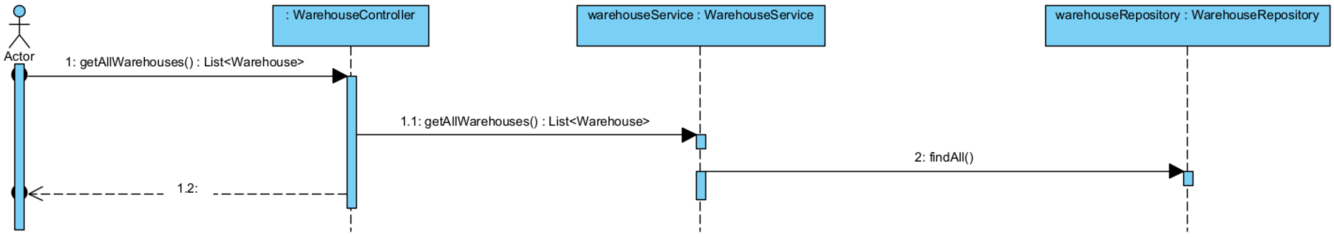

a
Actor
: ProductController
prodService : ProductService
prodRepository : ProductRepository
1: getAllProducts() : List<Product>
1.1: getAllProducts() : List<Product>
2: findAll()
1.2:

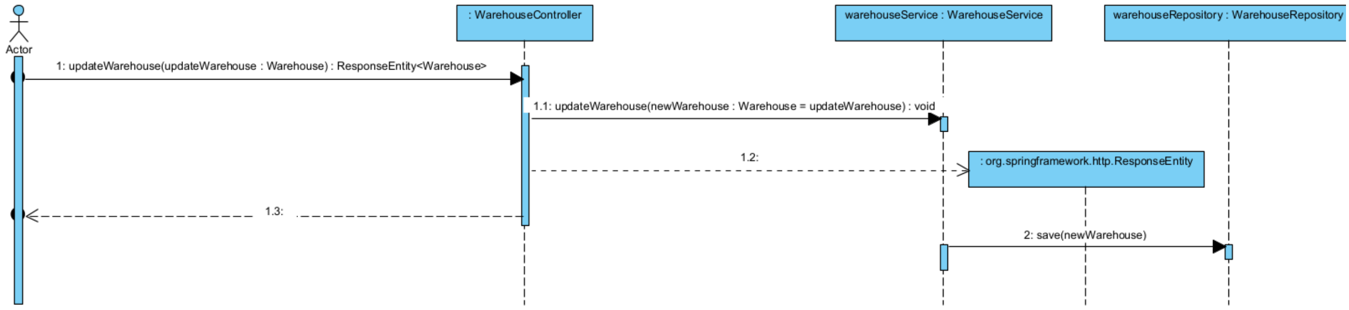

Actor
: ProductController
prodService : ProductService
prodRepository : ProductRepository
1: updateProduct(updateProduct : Product) : ResponseEntity<Product>
1.1: updateProduct(newproduct : Product = updateProduct) : void
1.2:
: org.springframework.http.ResponseEntity
1.3:
2: save(newproduct)

Manage warehouses:

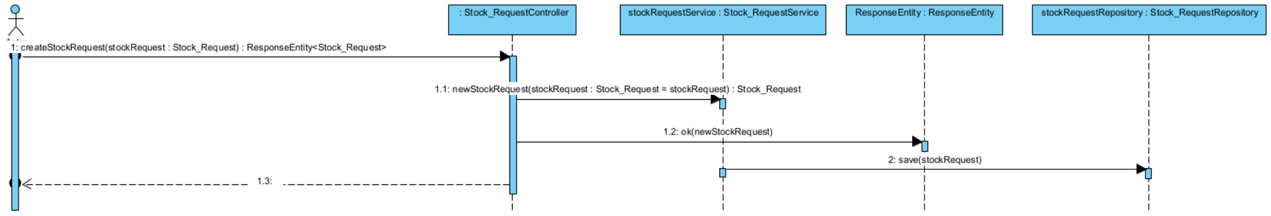

Actor
: WarehouseController
warehouseService : WarehouseService
warehouseRepository : WarehouseRepository
1: newWarehouse(newWarehouse : Warehouse) : ResponseEntity<Warehouse>
1.1: newWarehouse(newWarehouse : Warehouse = newWarehouse) : void
1.2:
: org.springframework.http.ResponseEntity
1.3:
2: save(newWarehouse)

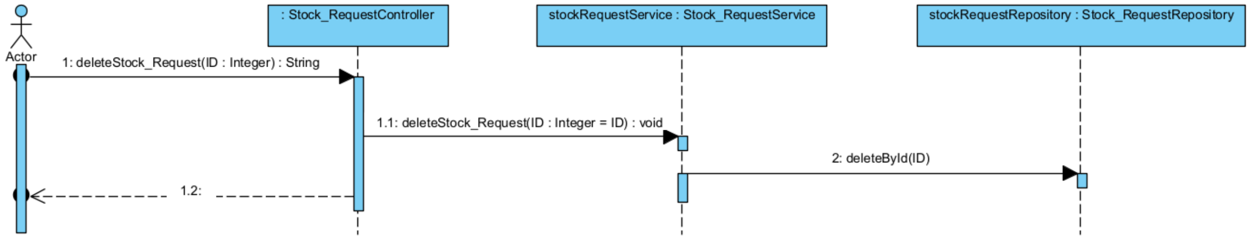

Actor
: WarehouseController
warehouseService : WarehouseService
warehouseRepository : WarehouseRepository
1: deleteWarehouse(ID : Integer) : String
2: deleteById(ID)
1.1: deleteWarehouse(ID : Integer = ID) : void
1.2:

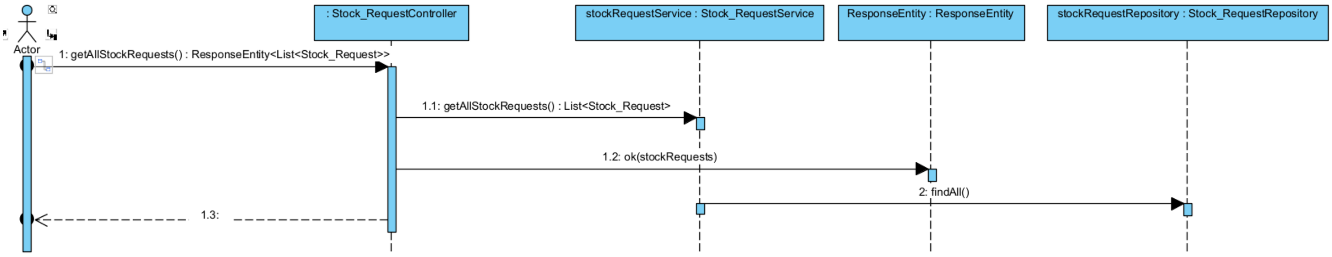

Actor
: WarehouseController
warehouseService : WarehouseService
warehouseRepository : WarehouseRepository
1: getAllWarehouses() : List<Warehouse>
1.1: getAllWarehouses() : List<Warehouse>
2: findAll()
1.2:

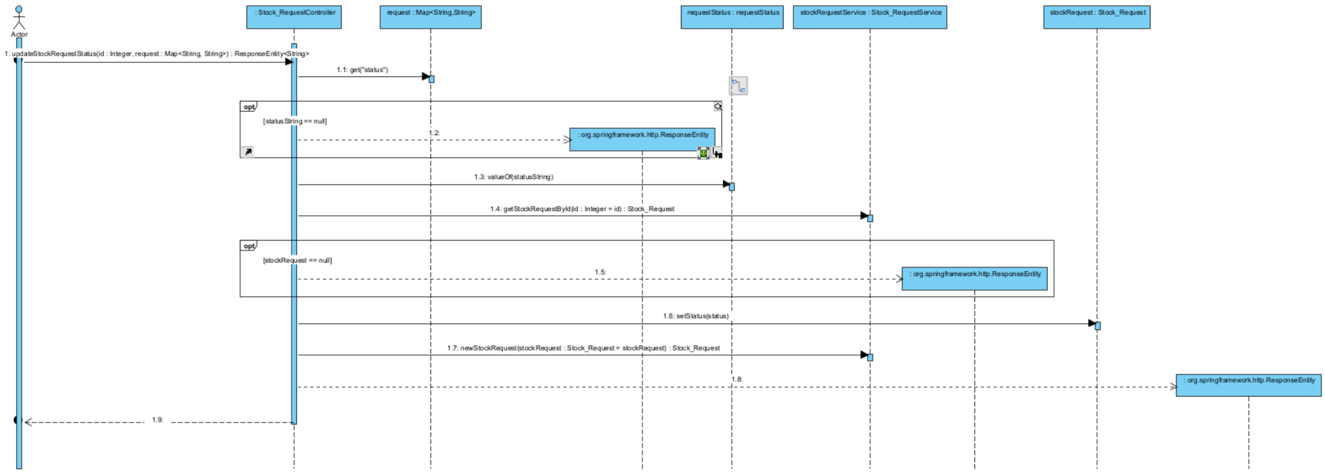

Actor
: WarehouseController
warehouseService : WarehouseService
warehouseRepository : WarehouseRepository
1: updateWarehouse(updateWarehouse : Warehouse) : ResponseEntity<Warehouse>
1.1: updateWarehouse(newWarehouse : Warehouse = updateWarehouse) : void
1.2:
: org.springframework.http.ResponseEntity
1.3:
2: save(newWarehouse)

Manage stock requests:

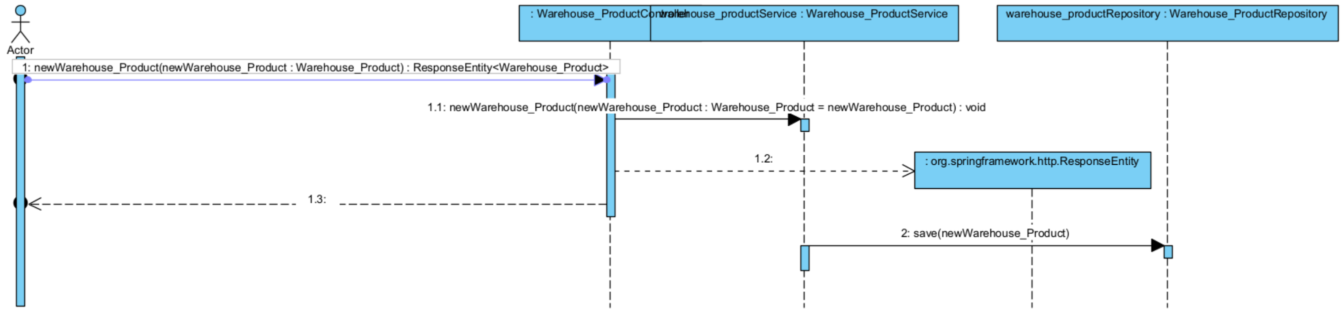

: Stock_RequestController
stockRequestService : Stock_RequestService
ResponseEntity : ResponseEntity
stockRequestRepository : Stock_RequestRepository
1: createStockRequest(stockRequest : Stock_Request) : ResponseEntity<Stock_Request>
1.1: newStockRequest(stockRequest : Stock_Request = stockRequest) : Stock_Request
1.2: ok(newStockRequest)
2: save(stockRequest)
1.3:

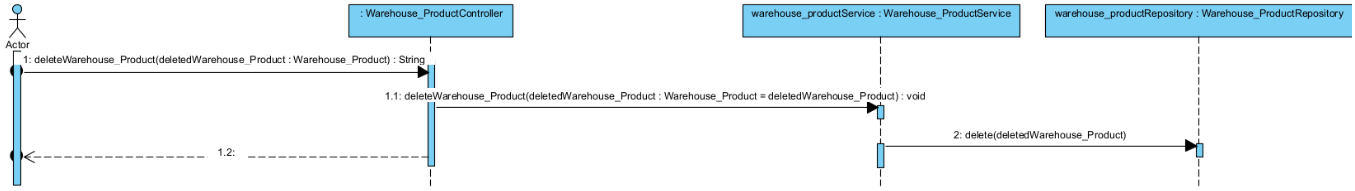

Actor
: Stock_RequestController
stockRequestService : Stock_RequestService
stockRequestRepository : Stock_RequestRepository
1: deleteStock_Request(ID : Integer) : String
1.1: deleteStock_Request(ID : Integer = ID) : void
2: deleteById(ID)
1.2:

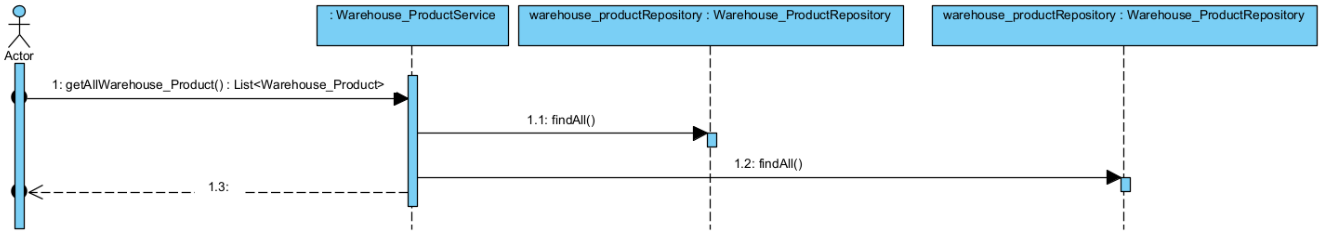

Actor
: Stock_RequestController
stockRequestService : Stock_RequestService
ResponseEntity : ResponseEntity
stockRequestRepository : Stock_RequestRepository
1: getAllStockRequests() : ResponseEntity<List<Stock_Request>>
1.1: getAllStockRequests() : List<Stock_Request>
1.2: ok(stockRequests)
2: findAll()
1.3:


Actor
: Stock_RequestController
request : Map<String,String>
requestStatus : requestStatus
stockRequestService : Stock_RequestService
stockRequest : Stock_Request
1: updateStockRequestStatus(id : Integer, request : Map<String, String>) : ResponseEntity<String>
1.1: get("status")
opt
[statusString == null]
1.2:
: org.springframework.http.ResponseEntity
1.3: valueOf(statusString)
1.4: getStockRequestById(id : Integer = id) : Stock_Request
opt
[stockRequest == null]
1.5:
: org.springframework.http.ResponseEntity
1.6: setStatus(status)
1.7: newStockRequest(stockRequest : Stock_Request = stockRequest) : Stock_Request
1.8:
: org.springframework.http.ResponseEntity
1.9:

Manage warehouse products:

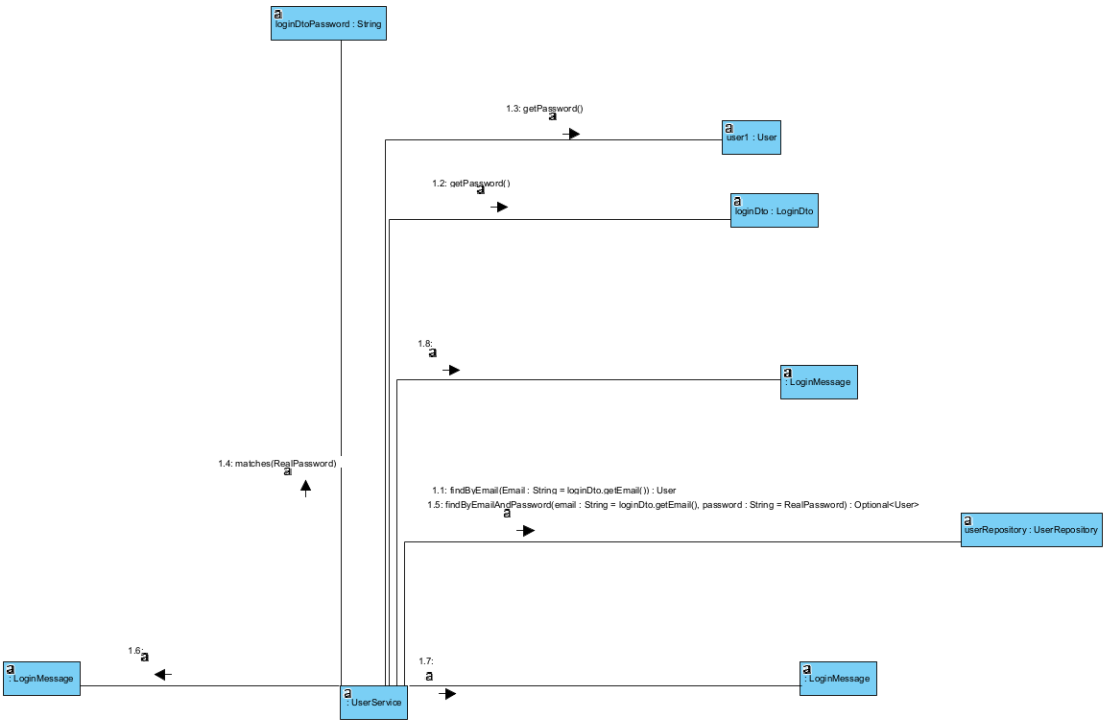

Actor
: Warehouse_ProductController
warehouse_productService : Warehouse_ProductService
warehouse_productRepository : Warehouse_ProductRepository
1: newWarehouse_Product(newWarehouse_Product : Warehouse_Product) : ResponseEntity<Warehouse_Product>
1.1: newWarehouse_Product(newWarehouse_Product : Warehouse_Product = newWarehouse_Product) : void
1.2:
: org.springframework.http.ResponseEntity
1.3:
2: save(newWarehouse_Product)

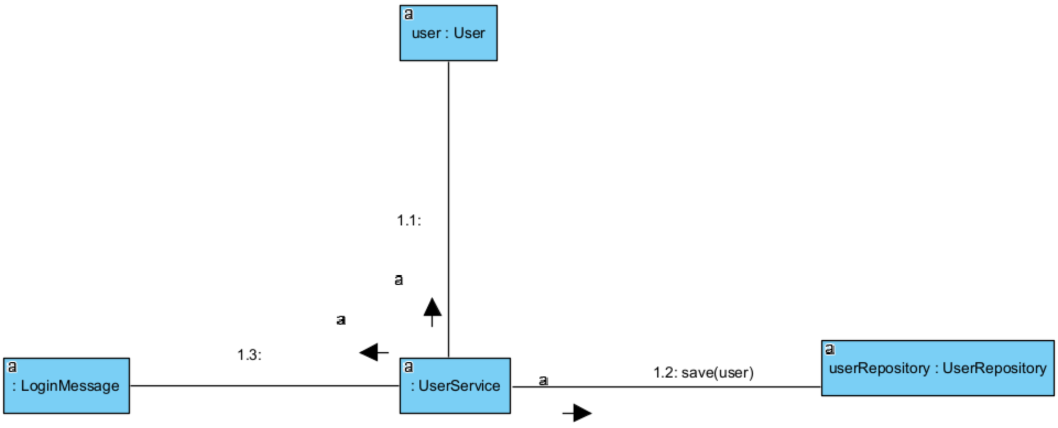

Actor
: Warehouse_ProductController
warehouse_productService : Warehouse_ProductService
warehouse_productRepository : Warehouse_ProductRepository
1: deleteWarehouse_Product(deletedWarehouse_Product : Warehouse_Product) : String
1.1: deleteWarehouse_Product(deletedWarehouse_Product : Warehouse_Product = deletedWarehouse_Product) : void
2: delete(deletedWarehouse_Product)
1.2:


Actor
: Warehouse_ProductService
warehouse_productRepository : Warehouse_ProductRepository
warehouse_productRepository : Warehouse_ProductRepository
1: getAllWarehouse_Product() : List<Warehouse_Product>
1.1: findAll()
1.2: findAll()
1.3:

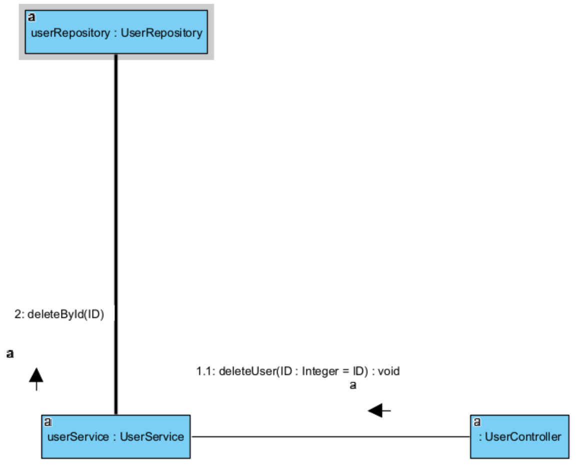
Communication Diagrams:

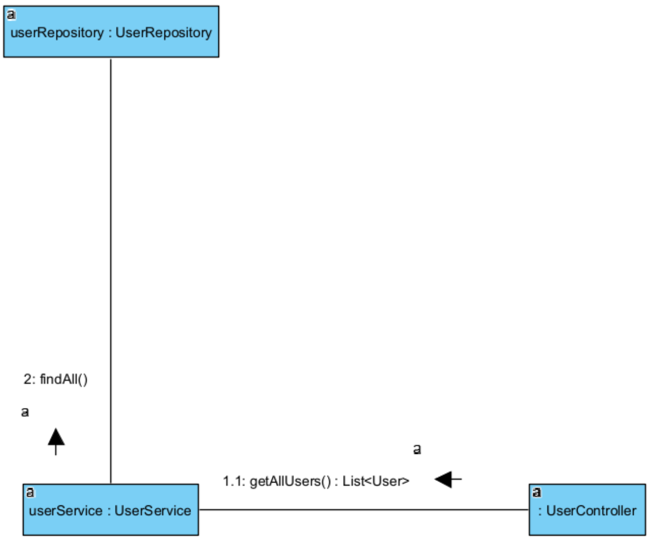
Login&Registration:

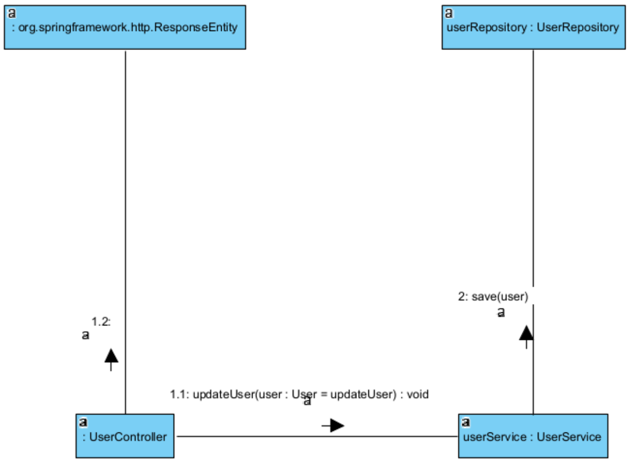

loginDtoPassword : String
1.3: getPassword()
user1 : User
1.2: getPassword( )
loginDto : LoginDto
1.8:
: LoginMessage
1.4: matches(RealPassword)
1.1: findByEmail(Email : String = loginDto.getEmail( )) : User
1.5: findByEmailAndPassword(email : String = loginDto.getEmail(), password : String = RealPassword) : Optional<User>
userRepository : UserRepository
1.6:
: LoginMessage
: UserService
1.7:
: LoginMessage



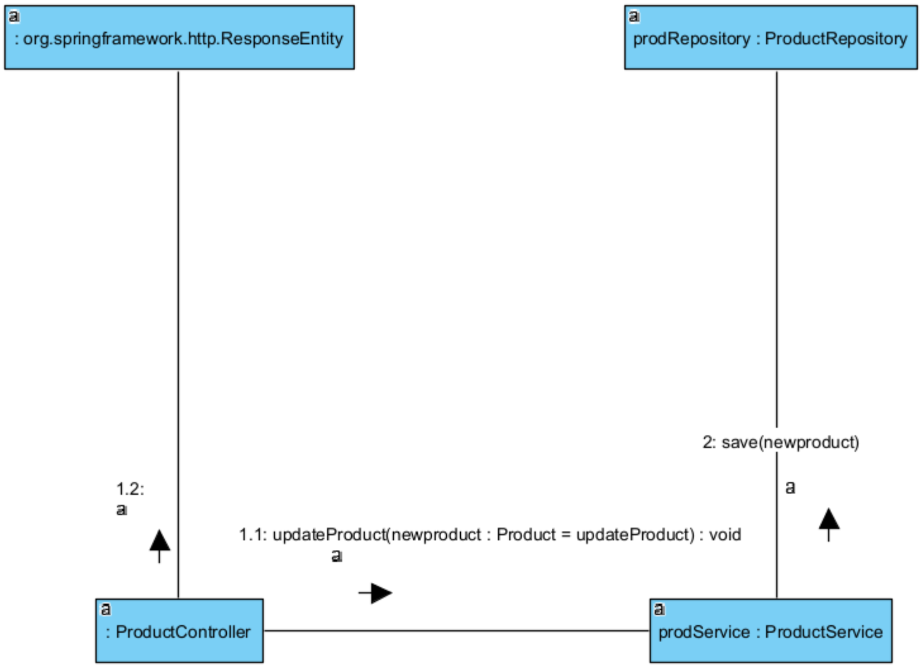
Manage users:

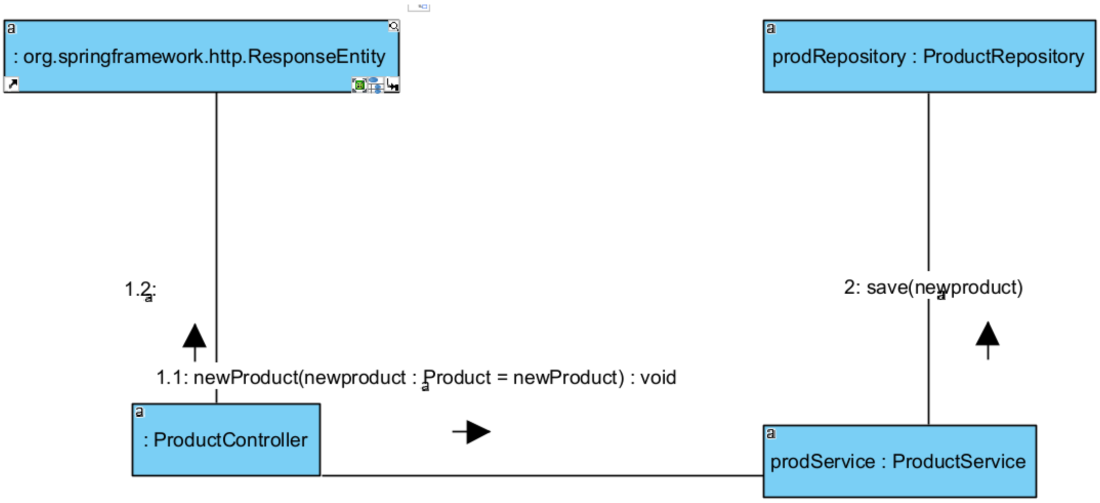

userRepository : UserRepository
2: deleteById(ID)
1.1: deleteUser(ID : Integer = ID) : void
userService : UserService
: UserController

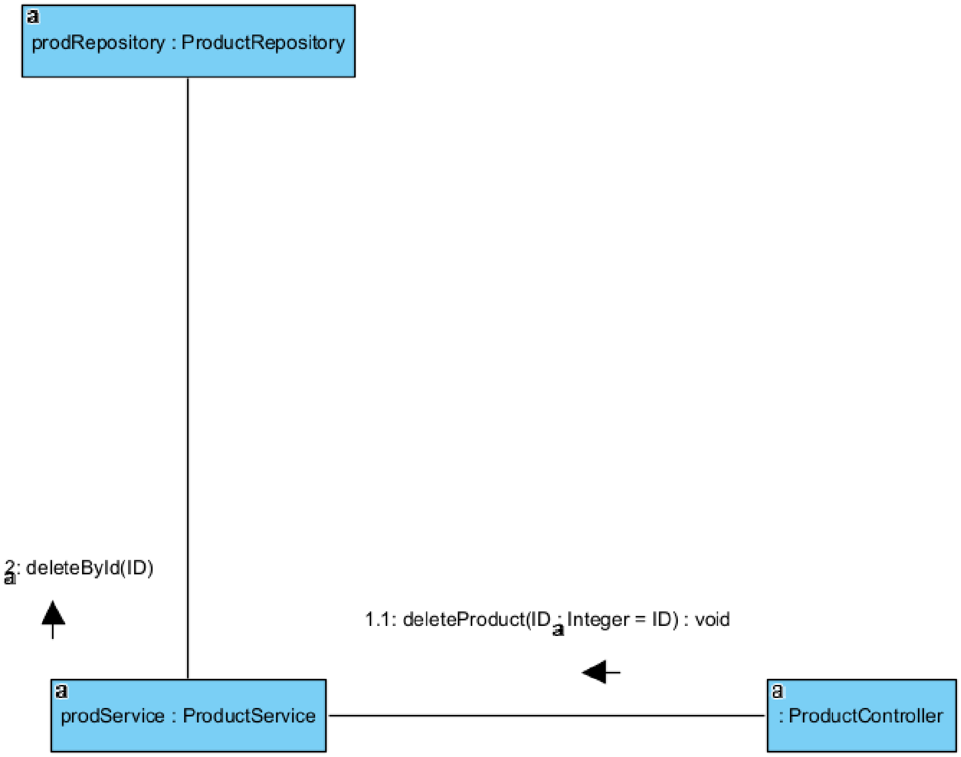

userRepository : UserRepository
2: findAll()
1.1: getAllUsers() : List<User>
userService : UserService
: UserController

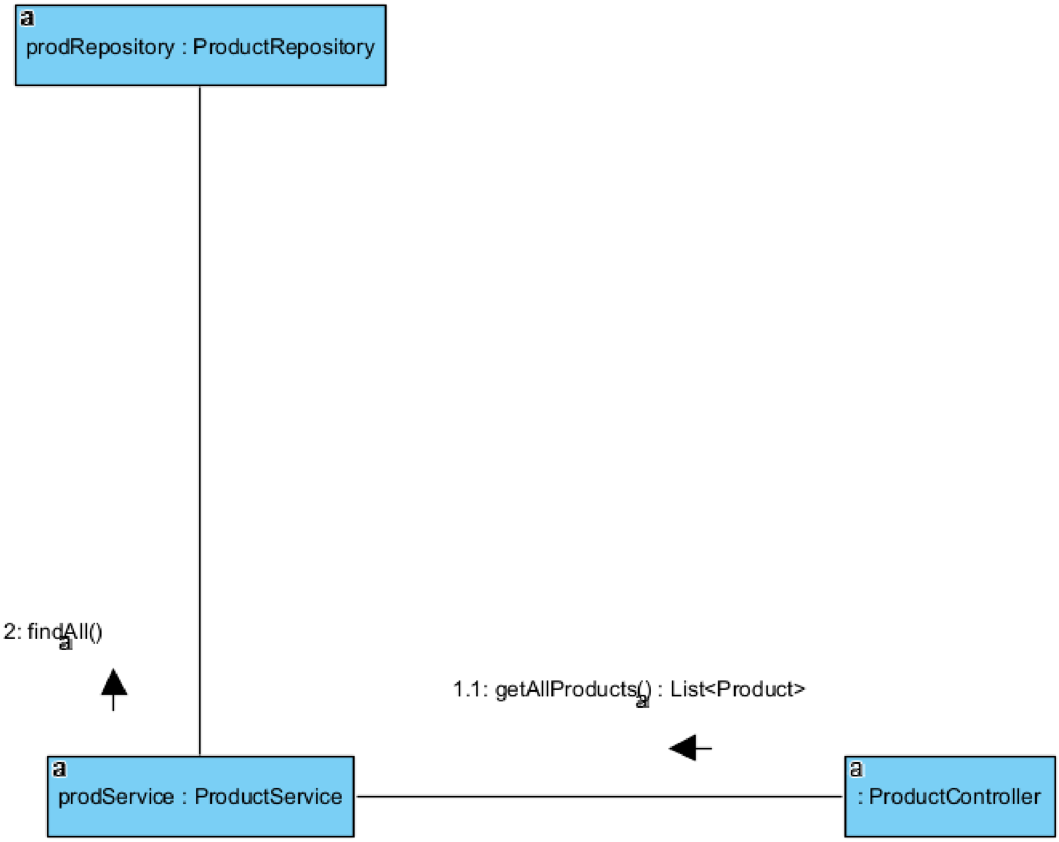

: org.springframework.http.ResponseEntity
userRepository : UserRepository
2: save(user)
1.2:
1.1: updateUser(user : User = updateUser) : void
: UserController
userService : UserService

Manage products:

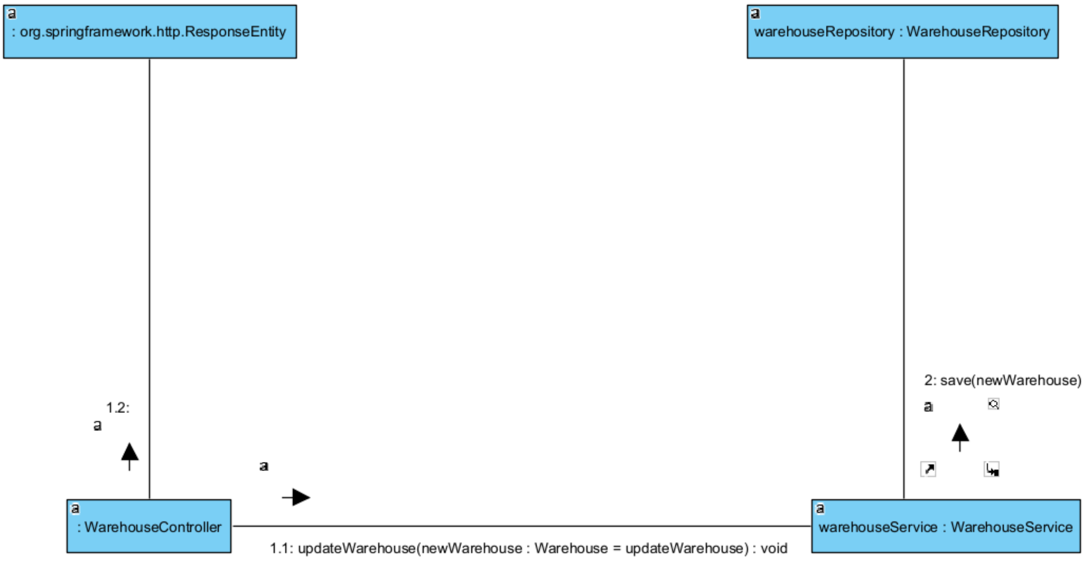

: org.springframework.http.ResponseEntity
prodRepository : ProductRepository
2: save(newproduct)
1.2:
1.1: updateProduct(newproduct : Product = updateProduct) : void
: ProductController
prodService : ProductService

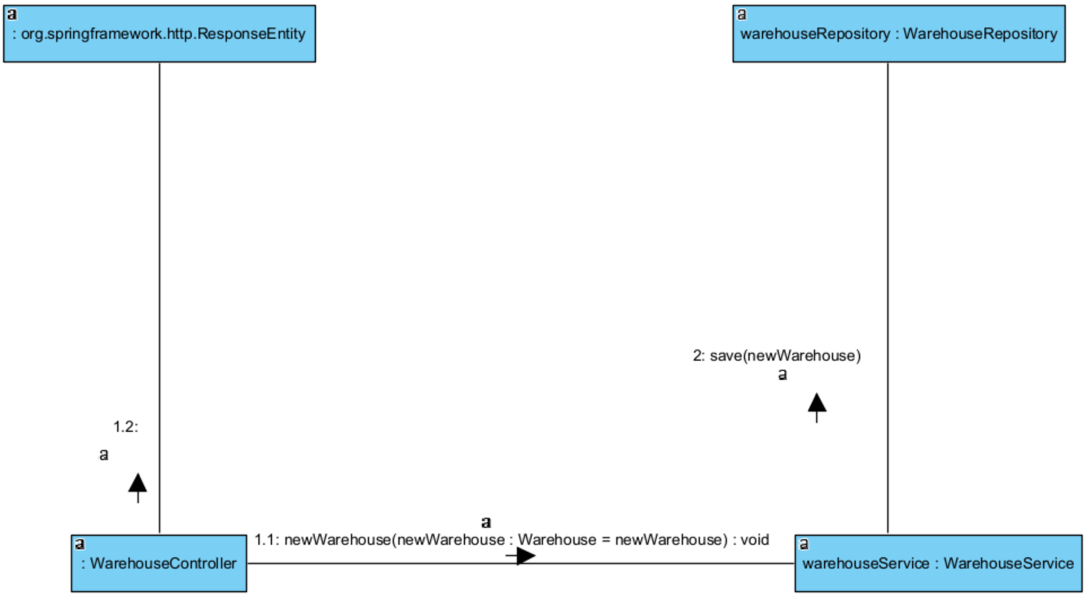

: org.springframework.http.ResponseEntity
prodRepository : ProductRepository
2: save(newproduct)
1.2:
1.1: newProduct(newproduct : Product = newProduct) : void
: ProductController
prodService : ProductService

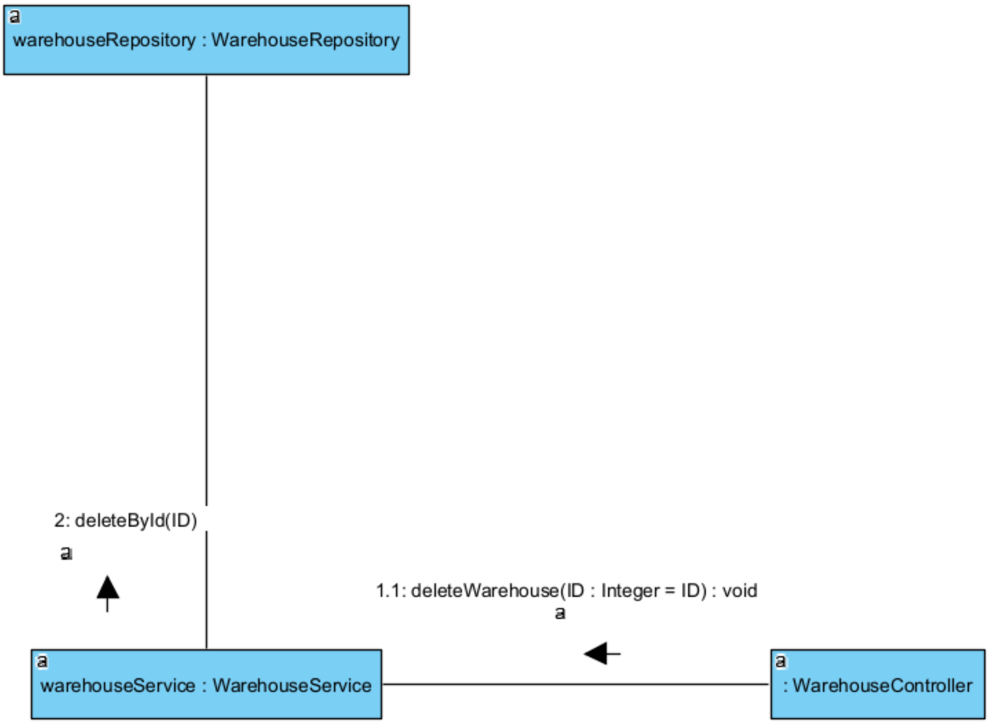

prodRepository : ProductRepository
2: deleteById(ID)
1.1: deleteProduct(ID : Integer = ID) : void
prodService : ProductService
: ProductController

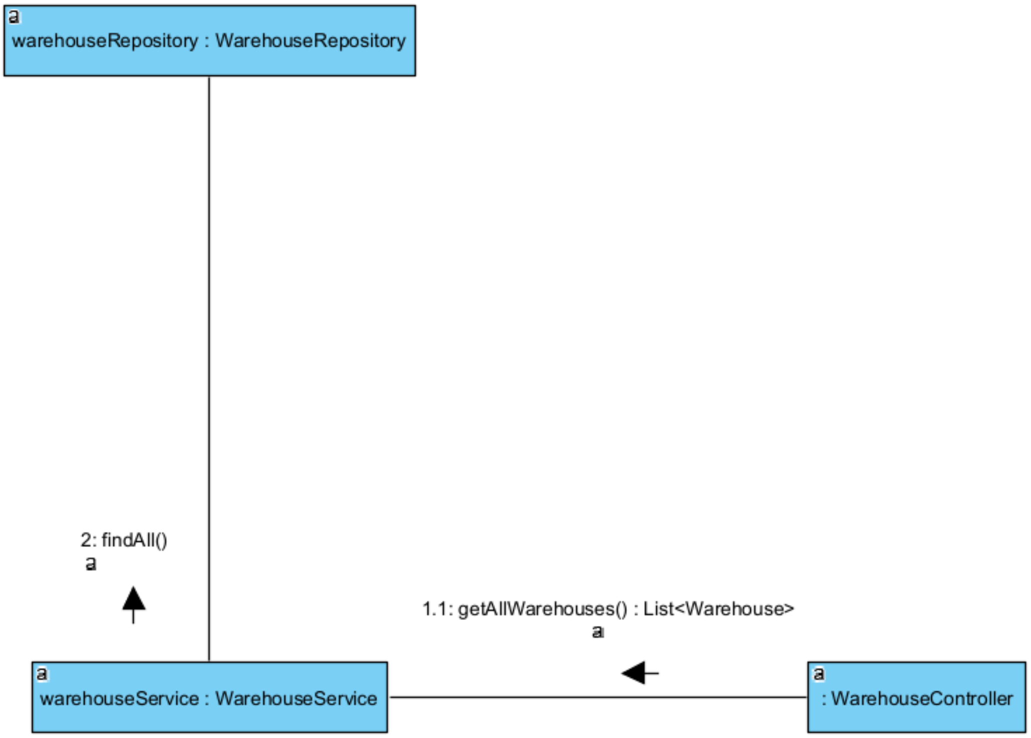

prodRepository : ProductRepository
2: findAll()
1.1: getAllProducts() : List<Product>
prodService : ProductService
: ProductController

Manage warhouses:

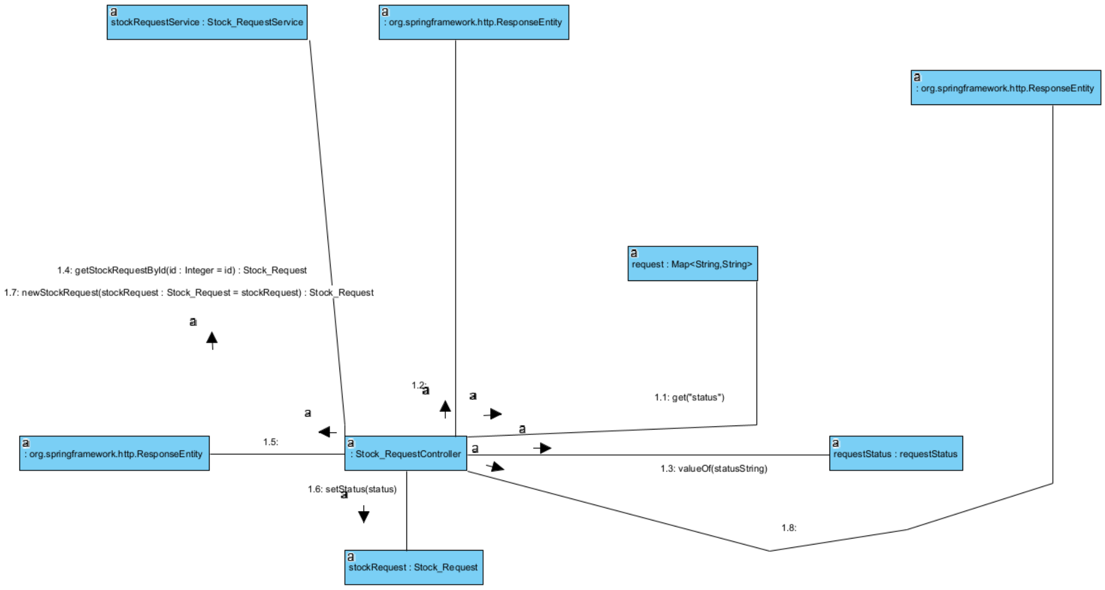

: org.springframework.http.ResponseEntity
warehouseRepository : WarehouseRepository
2: save(newWarehouse)
1.2:
: WarehouseController
warehouseService : WarehouseService
1.1: updateWarehouse(newWarehouse : Warehouse = updateWarehouse) : void

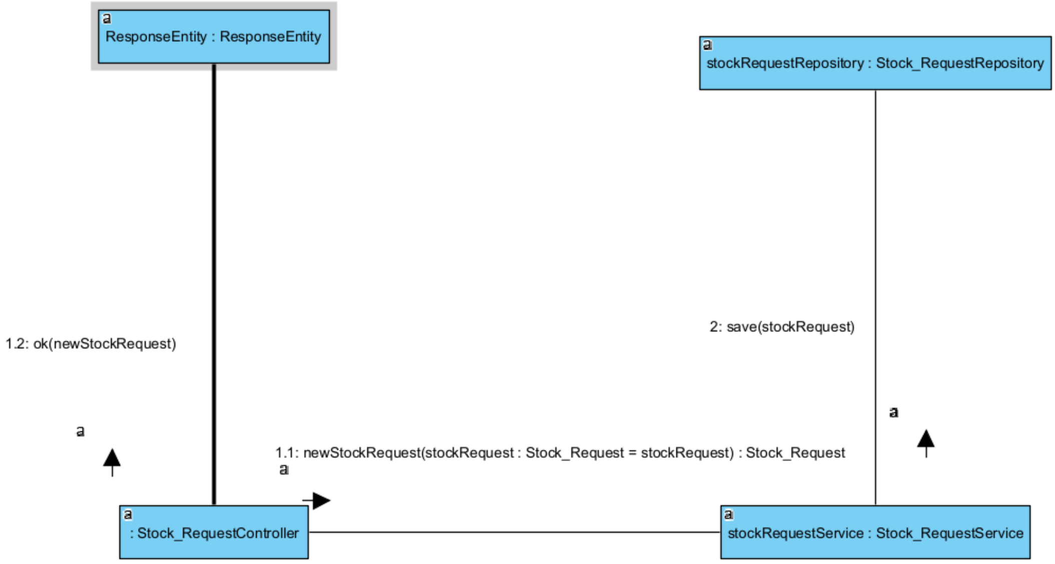

: org.springframework.http.ResponseEntity
warehouseRepository : WarehouseRepository
2: save(newWarehouse)
1.2:
1.1: newWarehouse(newWarehouse : Warehouse = newWarehouse) : void
: WarehouseController
warehouseService : WarehouseService

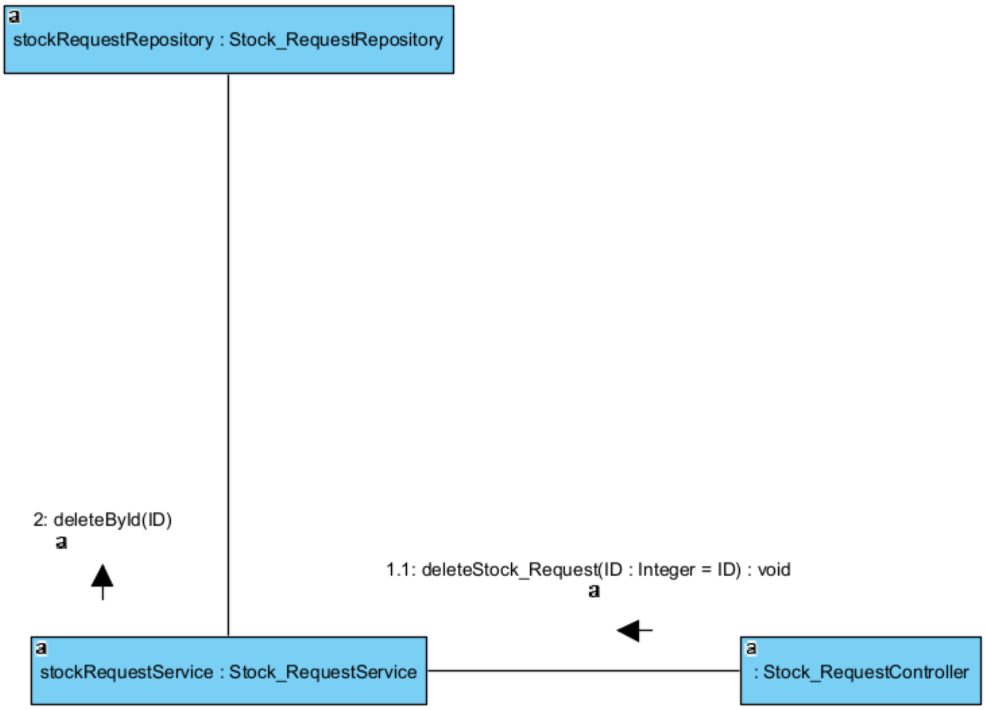

a
warehouseRepository : WarehouseRepository
2: deleteById(ID)
a
1.1: deleteWarehouse(ID : Integer = ID) : void
a
a
warehouseService : WarehouseService
a
: WarehouseController

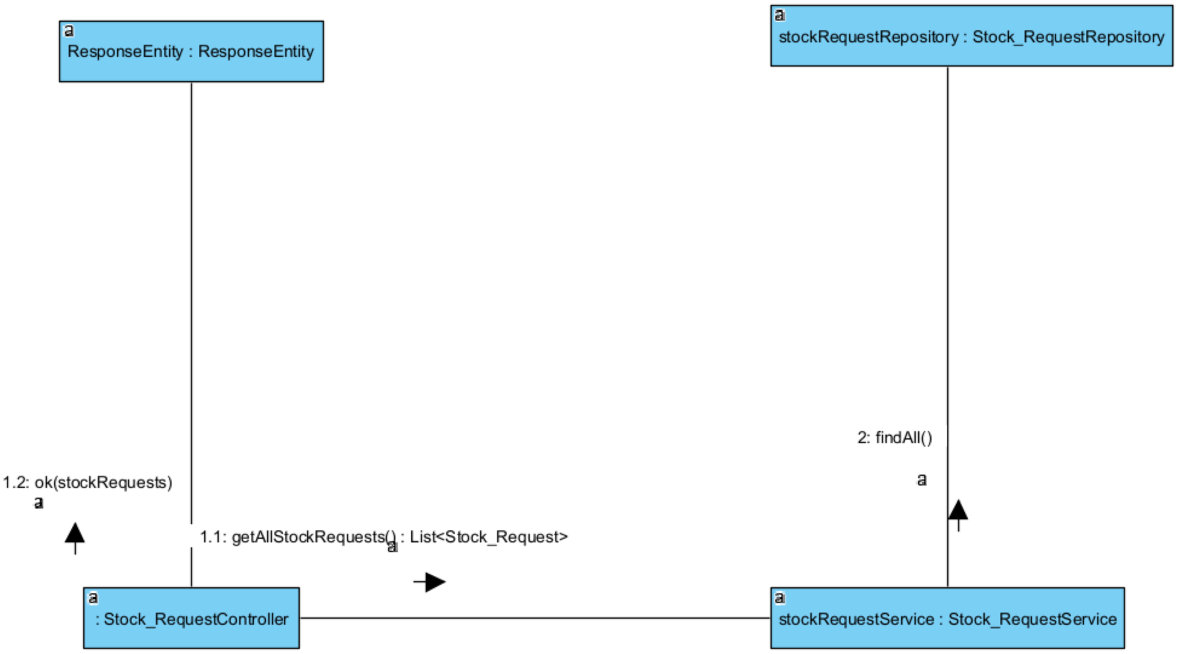

a
warehouseRepository : WarehouseRepository
2: findAll()
a
1.1: getAllWarehouses() : List<Warehouse>
a
a
warehouseService : WarehouseService
a
: WarehouseController

Manage stock requests:

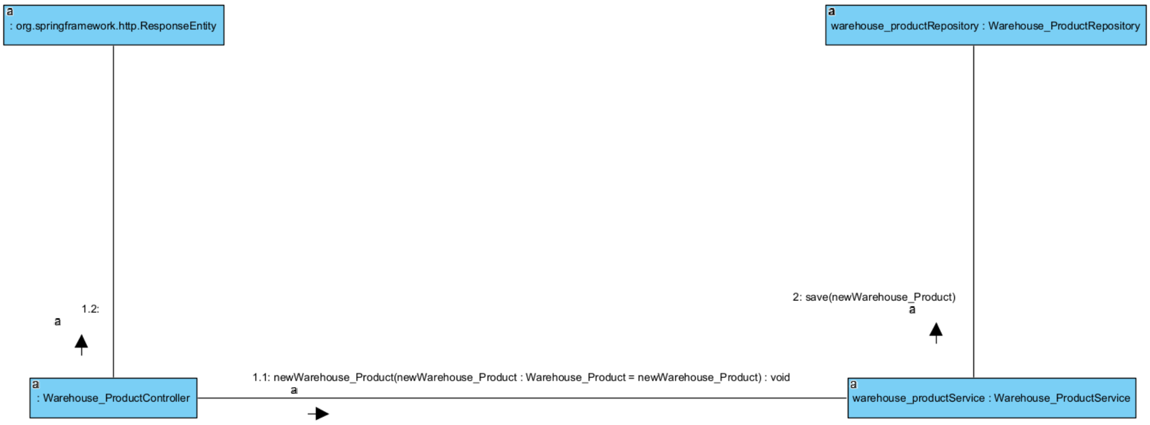

stockRequestService : Stock_RequestService
: org.springframework.http.ResponseEntity
: org.springframework.http.ResponseEntity
request : Map<String,String>
1.4: getStockRequestById(id : Integer = id) : Stock_Request
1.7: newStockRequest(stockRequest : Stock_Request = stockRequest) : Stock_Request
1.2:
1.1: get("status")
: org.springframework.http.ResponseEntity
1.5:
: Stock_RequestController
requestStatus : requestStatus
1.3: valueOf(statusString)
1.6: setStatus(status)
1.8:
stockRequest : Stock_Request

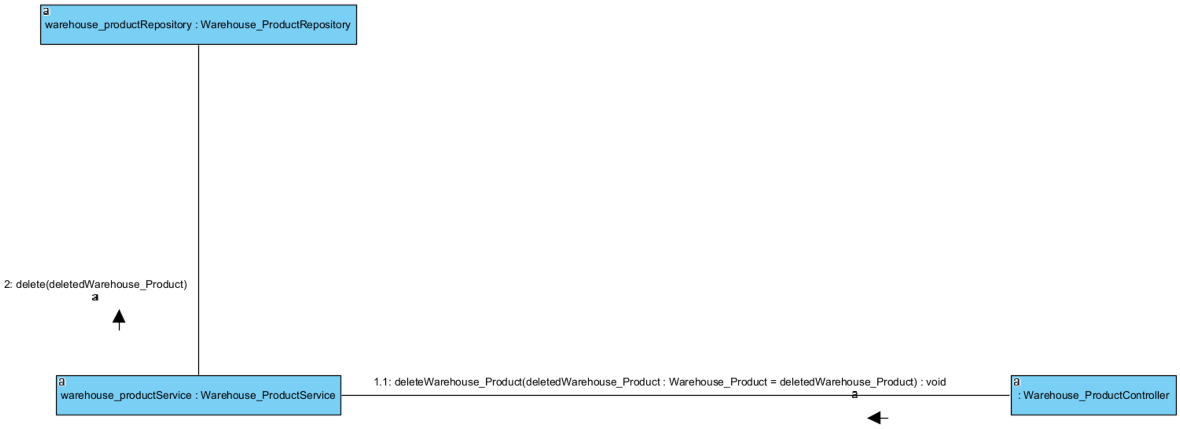

ResponseEntity : ResponseEntity
stockRequestRepository : Stock_RequestRepository
1.2: ok(newStockRequest)
2: save(stockRequest)
1.1: newStockRequest(stockRequest : Stock_Request = stockRequest) : Stock_Request
: Stock_RequestController
stockRequestService : Stock_RequestService

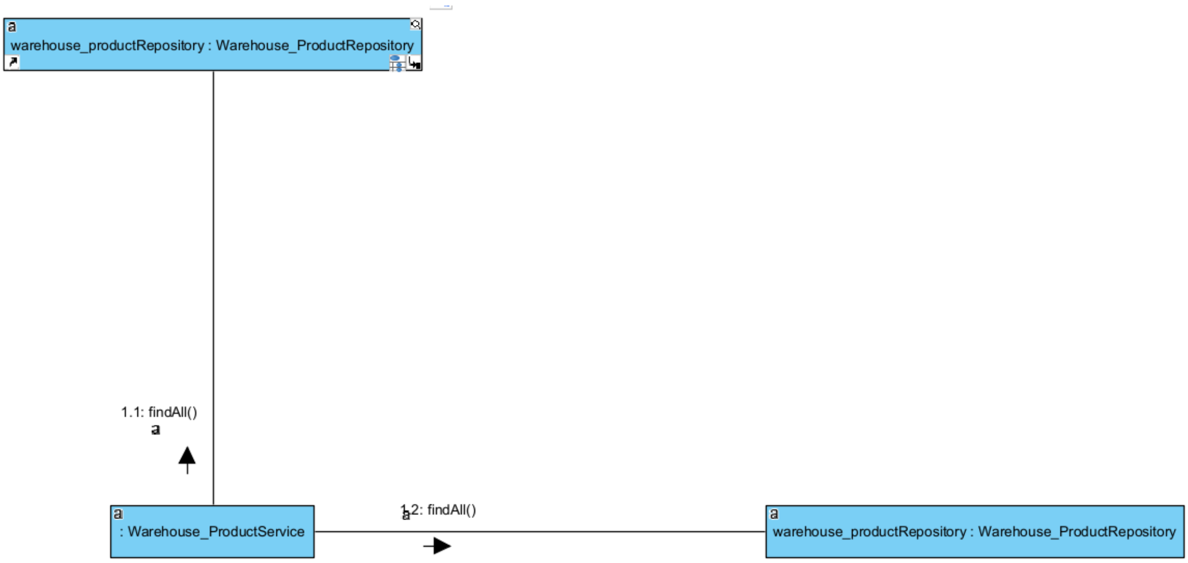

stockRequestRepository : Stock_RequestRepository
2: deleteById(ID)
1.1: deleteStock_Request(ID : Integer = ID) : void
stockRequestService : Stock_RequestService
: Stock_RequestController


ResponseEntity : ResponseEntity
stockRequestRepository : Stock_RequestRepository
2: findAll()
1.2: ok(stockRequests)
1.1: getAllStockRequests() : List<Stock_Request>
: Stock_RequestController
stockRequestService : Stock_RequestService

Manage warehouse products:


a
: org.springframework.http.ResponseEntity
a
warehouse_productRepository : Warehouse_ProductRepository
1.2:
a
2: save(newWarehouse_Product)
a
a
: Warehouse_ProductController
1.1: newWarehouse_Product(newWarehouse_Product : Warehouse_Product = newWarehouse_Product) : void
a
a
warehouse_productService : Warehouse_ProductService


a
warehouse_productRepository : Warehouse_ProductRepository
2: delete(deletedWarehouse_Product)
a
a
warehouse_productService : Warehouse_ProductService
1.1: deleteWarehouse_Product(deletedWarehouse_Product : Warehouse_Product = deletedWarehouse_Product) : void
a
a
: Warehouse_ProductController


a
warehouse_productRepository : Warehouse_ProductRepository
1.1: findAll()
a
a
: Warehouse_ProductService
1.2: findAll()
a
a
warehouse_productRepository : Warehouse_ProductRepository