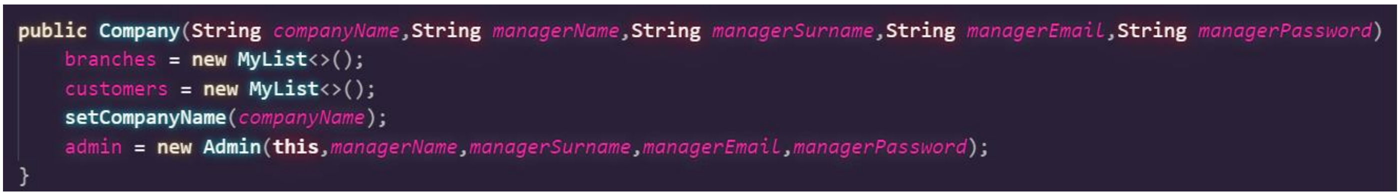
GIT Department of Computer Engineering
CSE 222/505 - Spring 2021
Homework #1 Report

ATAKAN ALTIN
1801042668

1. SYSTEM REQUIREMENTS

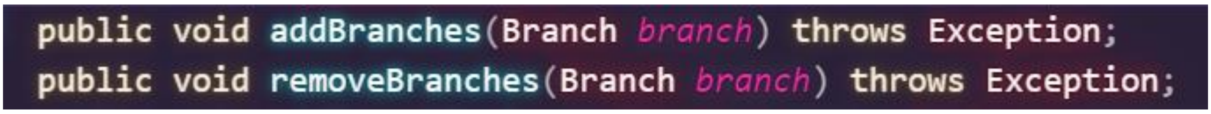
First things first, we should create a **company** object.



```
public Company(String companyName,String managerName,String managerSurname,String managerEmail,String managerPassword)
    branches = new MyList<>();
    customers = new MyList<>();
    setCompanyName(companyName);
    admin = new Admin(this,managerName,managerSurname,managerEmail,managerPassword);
}
```

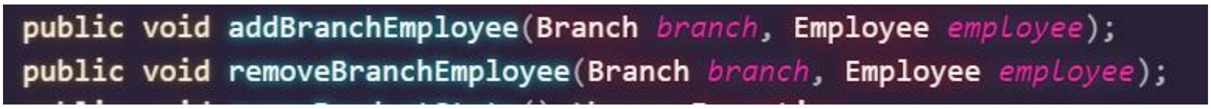
Company constructor initializes company's name and authorizes administrator.

Administrator can add/remove branches and employees.



```
public void addBranches(Branch branch) throws Exception;
public void removeBranches(Branch branch) throws Exception;
```

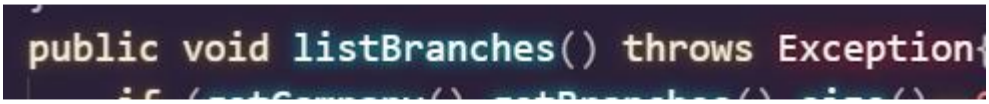
Add/remove branch functions takes branch objects as a parameter.



```
public void addBranchEmployee(Branch branch, Employee employee);
public void removeBranchEmployee(Branch branch, Employee employee);
```

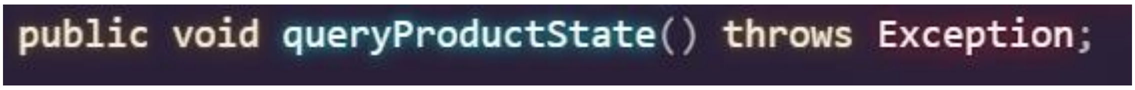
When the admin adding a new employee, he/she must know which store to add it to.

Add/remove employee functions also takes employee objects as a parameter.



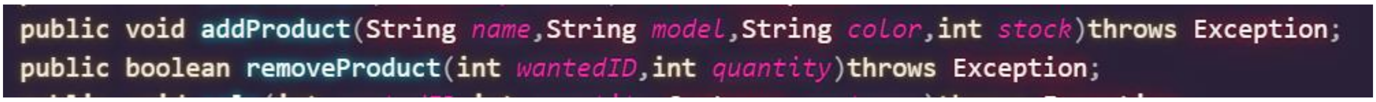
```
public void listBranches() throws Exception{
```

Admin also can list company's all branches and can view if there are products that need to be supplied.



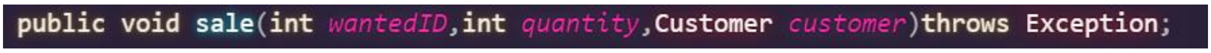
```
public void queryProductState() throws Exception;
```

Employees can add/remove products to his/her company.



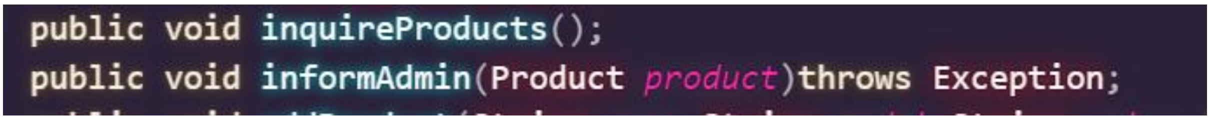
```
public void addProduct(String name,String model,String color,int stock)throws Exception;
public boolean removeProduct(int wantedID,int quantity)throws Exception;
```

Employees can sale products to customers in store. They add products to customer's order history. If customer is not a part of automation system then employee can add customer to the system. It takes product's id, quantity and customer object.



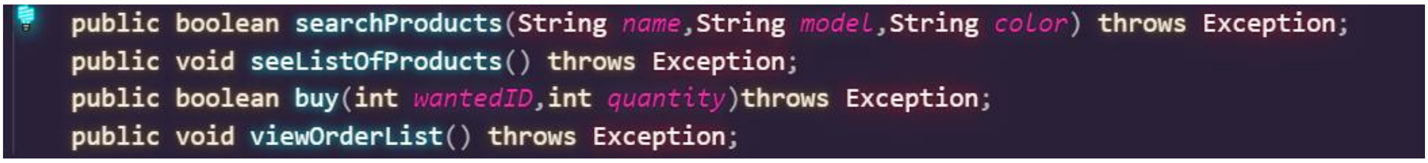
```
public void sale(int wantedID,int quantity,Customer customer)throws Exception;
```

They also inquire all products and they can observe products status. They can inform the admin about product's status.



```
public void inquireProducts();
public void informAdmin(Product product)throws Exception;
```

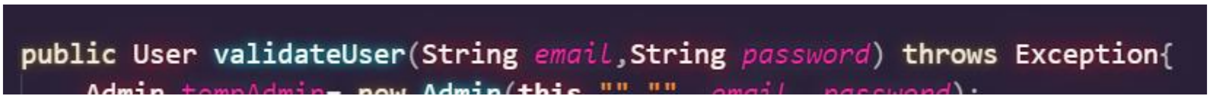
Customer's able to subscribe to the automation system. It can use sign up function after the customer's object is created.



```
public boolean searchProducts(String name,String model,String color) throws Exception;
public void seeListOfProducts() throws Exception;
public boolean buy(int wantedID,int quantity)throws Exception;
public void viewOrderList() throws Exception;
```

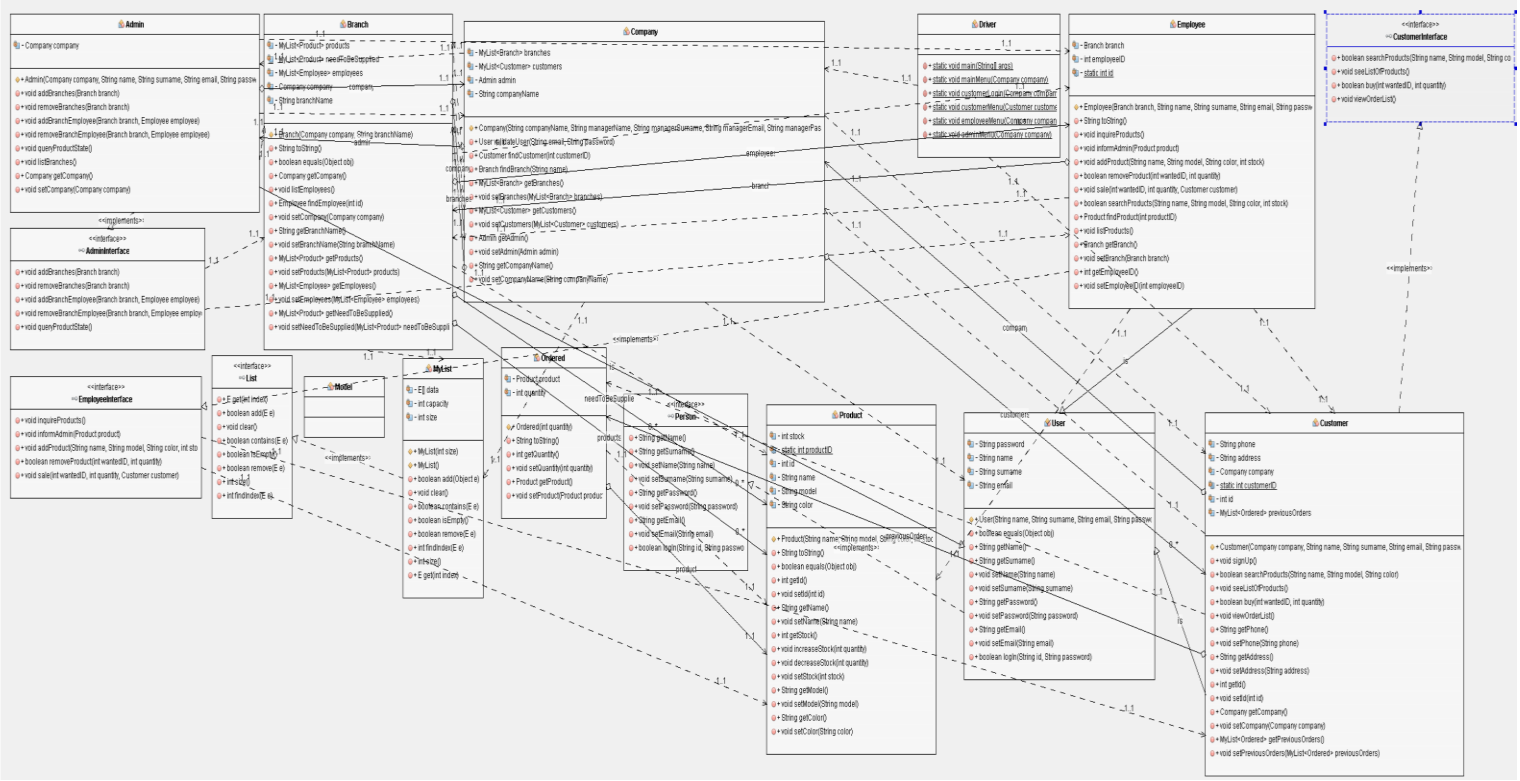
They can search individual products, see all of the products in the system, buy products and view order history.

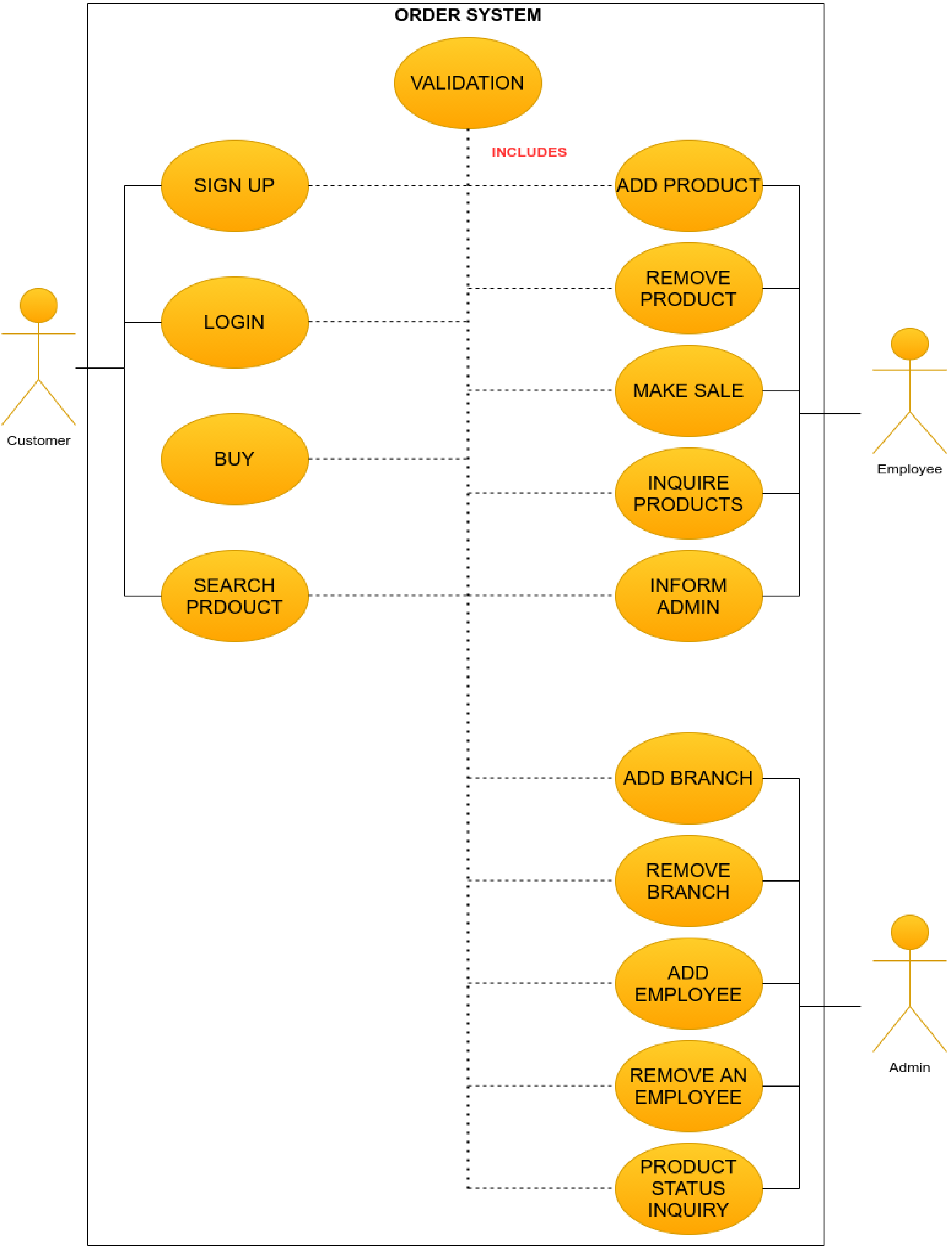
All users can login to the system and it is checked by company's validation function.



```
public User validateUser(String email,String password) throws Exception{
    Admin tempAdmin= new Admin(this,"","", email, password);
```

2. USE CASE AND CLASS DIAGRAMS

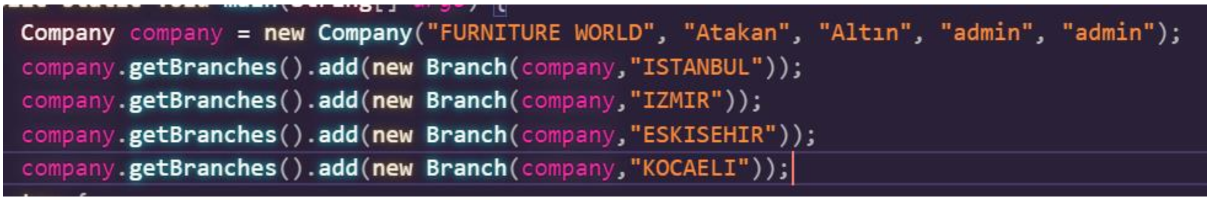

Admin
- Company company
+ Admin(Company company, String name, String surname, String email, String passw
+ void addBranches(Branch branch)
+ void removeBranches(Branch branch)
+ void addBranchEmployee(Branch branch, Employee employee)
+ void removeBranchEmployee(Branch branch, Employee employee)
+ void queryProductState()
+ void listBranches()
+ Company getCompany()
+ void setCompany(Company company)
Branch
- MyList<Product> products
- MyList<Product> needToBeSupplied
- MyList<Employee> employees
- Company company
- String branchName
+ Branch(Company company, String branchName)
+ String toString()
+ boolean equals(Object obj)
+ Company getCompany()
+ void listEmployees()
+ Employee findEmployee(int id)
+ void setCompany(Company company)
+ String getBranchName()
+ void setBranchName(String branchName)
+ MyList<Product> getProducts()
+ void setProducts(MyList<Product> products)
+ MyList<Employee> getEmployees()
+ void setEmployees(MyList<Employee> employees)
+ MyList<Product> getNeedToBeSupplied()
+ void setNeedToBeSupplied(MyList<Product> needToBeSuppli
Company
- MyList<Branch> branches
- MyList<Customer> customers
- Admin admin
- String companyName
+ Company(String companyName, String managerName, String managerSurname, String managerEmail, String managerPas
+ User validateUser(String email, String password)
+ Customer findCustomer(int customerID)
+ Branch findBranch(String name)
+ MyList<Branch> getBranches()
+ void setBranches(MyList<Branch> branches)
+ MyList<Customer> getCustomers()
+ void setCustomers(MyList<Customer> customers)
+ Admin getAdmin()
+ void setAdmin(Admin admin)
+ String getCompanyName()
+ void setCompanyName(String companyName)
Driver
+ static void main(String[] args)
+ static void mainMenu(Company company)
+ static void customerLogin(Company compan
+ static void customerMenu(Customer custome
+ static void employeeMenu(Company compan
+ static void adminMenu(Company company)
Employee
- Branch branch
- int employeeID
- static int id
+ Employee(Branch branch, String name, String surname, String email, String passw
+ String toString()
+ void inquireProducts()
+ void informAdmin(Product product)
+ void addProduct(String name, String model, String color, int stock)
+ boolean removeProduct(int wantedID, int quantity)
+ void sale(int wantedID, int quantity, Customer customer)
+ boolean searchProducts(String name, String model, String color, int stock)
+ Product findProduct(int productID)
+ void listProducts()
+ Branch getBranch()
+ void setBranch(Branch branch)
+ int getEmployeeID()
+ void setEmployeeID(int employeeID)
<<interface>>
CustomerInterface
+ boolean searchProducts(String name, String model, String co
+ void seeListOfProducts()
+ boolean buy(int wantedID, int quantity)
+ void viewOrderList()
<<implements>>
<<interface>>
AdminInterface
+ void addBranches(Branch branch)
+ void removeBranches(Branch branch)
+ void addBranchEmployee(Branch branch, Employee employee)
+ void removeBranchEmployee(Branch branch, Employee employ
+ void queryProductState()
<<interface>>
EmployeeInterface
+ void inquireProducts()
+ void informAdmin(Product product)
+ void addProduct(String name, String model, String color, int sto
+ boolean removeProduct(int wantedID, int quantity)
+ void sale(int wantedID, int quantity, Customer customer)
<<interface>>
List
+ E get(int index)
+ boolean add(E e)
+ void clear()
+ boolean contains(E e)
+ boolean isEmpty()
+ boolean remove(E e)
+ int size()
+ int findIndex(E e)
Model
MyList
- E[] data
- int capacity
- int size
+ MyList(int size)
+ MyList()
+ boolean add(Object e)
+ void clear()
+ boolean contains(E e)
+ boolean isEmpty()
+ boolean remove(E e)
+ int findIndex(E e)
+ int size()
+ E get(int index)
Ordered
- Product product
- int quantity
+ Ordered(int quantity)
+ String toString()
+ int getQuantity()
+ void setQuantity(int quantity)
+ Product getProduct()
+ void setProduct(Product produc
<<interface>>
Person
+ String getName()
+ String getSurname()
+ void setName(String name)
+ void setSurname(String surname)
+ String getPassword()
+ void setPassword(String password)
+ String getEmail()
+ void setEmail(String email)
+ boolean login(String id, String passwo
Product
- int stock
- static int productID
- int id
- String name
- String model
- String color
+ Product(String name, String model, String color, int stoc
+ String toString()
+ boolean equals(Object obj)
+ int getId()
+ void setId(int id)
+ String getName()
+ void setName(String name)
+ int getStock()
+ void increaseStock(int quantity)
+ void decreaseStock(int quantity)
+ void setStock(int stock)
+ String getModel()
+ void setModel(String model)
+ String getColor()
+ void setColor(String color)
User
- String password
- String name
- String surname
- String email
+ User(String name, String surname, String email, String passw
+ boolean equals(Object obj)
+ String getName()
+ String getSurname()
+ void setName(String name)
+ void setSurname(String surname)
+ String getPassword()
+ void setPassword(String password)
+ String getEmail()
+ void setEmail(String email)
+ boolean login(String id, String password)
Customer
- String phone
- String address
- Company company
- static int customerID
- int id
- MyList<Ordered> previousOrders
+ Customer(Company company, String name, String surname, String email, String passw
+ void signUp()
+ boolean searchProducts(String name, String model, String color)
+ void seeListOfProducts()
+ boolean buy(int wantedID, int quantity)
+ void viewOrderList()
+ String getPhone()
+ void setPhone(String phone)
+ String getAddress()
+ void setAddress(String address)
+ int getId()
+ void setId(int id)
+ Company getCompany()
+ void setCompany(Company company)
+ MyList<Ordered> getPreviousOrders()
+ void setPreviousOrders(MyList<Ordered> previousOrders)
company
admin
employee
branch
branches
needToBeSupplie
products
product
previousOrders
customers
is
1..1
0..*


ORDER SYSTEM
VALIDATION
INCLUDES
SIGN UP
LOGIN
BUY
SEARCH PRDOUCT
ADD PRODUCT
REMOVE PRODUCT
MAKE SALE
INQUIRE PRODUCTS
INFORM ADMIN
ADD BRANCH
REMOVE BRANCH
ADD EMPLOYEE
REMOVE AN EMPLOYEE
PRODUCT STATUS INQUIRY
Customer
Employee
Admin

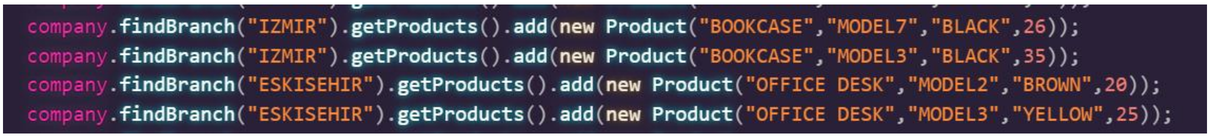
3. PROBLEM SOLUTION APPROACH

Since we are only allowed to use java arrays I create my own list interface and class similar to java ArrayList by using java arrays. I create company class to record all system data easily. These operations allow me to design all class hierarchy much more easily.

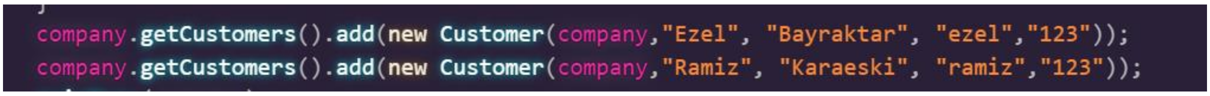
4. TEST CASES



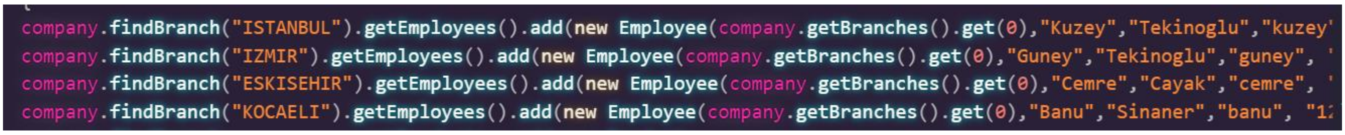
```
Company company = new Company("FURNITURE WORLD", "Atakan", "Altın", "admin", "admin");
company.getBranches().add(new Branch(company,"ISTANBUL"));
company.getBranches().add(new Branch(company,"IZMIR"));
company.getBranches().add(new Branch(company,"ESKISEHIR"));
company.getBranches().add(new Branch(company,"KOCAELI"));
```



```
company.findBranch("IZMIR").getProducts().add(new Product("BOOKCASE","MODEL7","BLACK",26));
company.findBranch("IZMIR").getProducts().add(new Product("BOOKCASE","MODEL3","BLACK",35));
company.findBranch("ESKISEHIR").getProducts().add(new Product("OFFICE DESK","MODEL2","BROWN",20));
company.findBranch("ESKISEHIR").getProducts().add(new Product("OFFICE DESK","MODEL3","YELLOW",25));
```



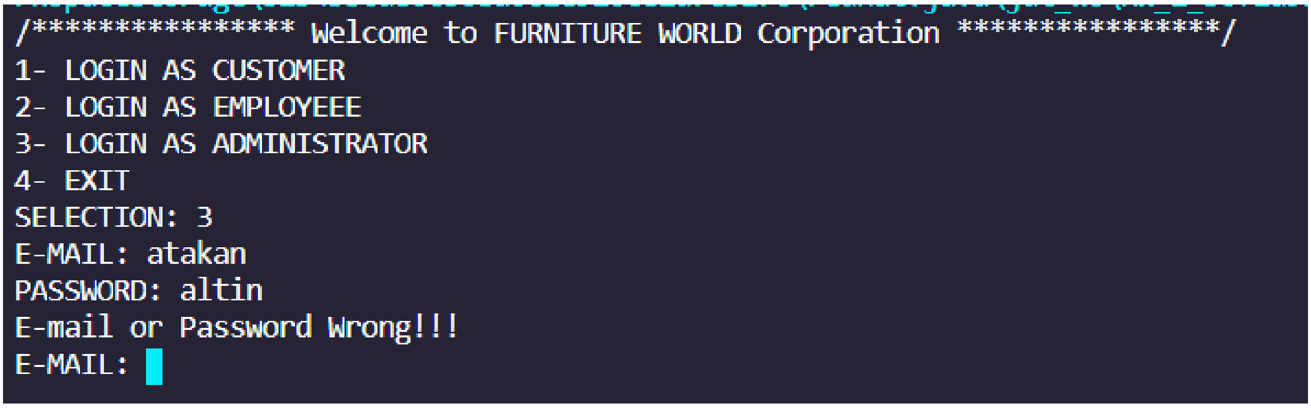
```
company.getCustomers().add(new Customer(company,"Ezel", "Bayraktar", "ezel","123"));
company.getCustomers().add(new Customer(company,"Ramiz", "Karaeski", "ramiz","123"));
```



```
company.findBranch("ISTANBUL").getEmployees().add(new Employee(company.getBranches().get(0),"Kuzey","Tekinoglu","kuzey"
company.findBranch("IZMIR").getEmployees().add(new Employee(company.getBranches().get(0),"Guney","Tekinoglu","guney",
company.findBranch("ESKISEHIR").getEmployees().add(new Employee(company.getBranches().get(0),"Cemre","Cayak","cemre",
company.findBranch("KOCAELI").getEmployees().add(new Employee(company.getBranches().get(0),"Banu","Sinaner","banu", "1
```

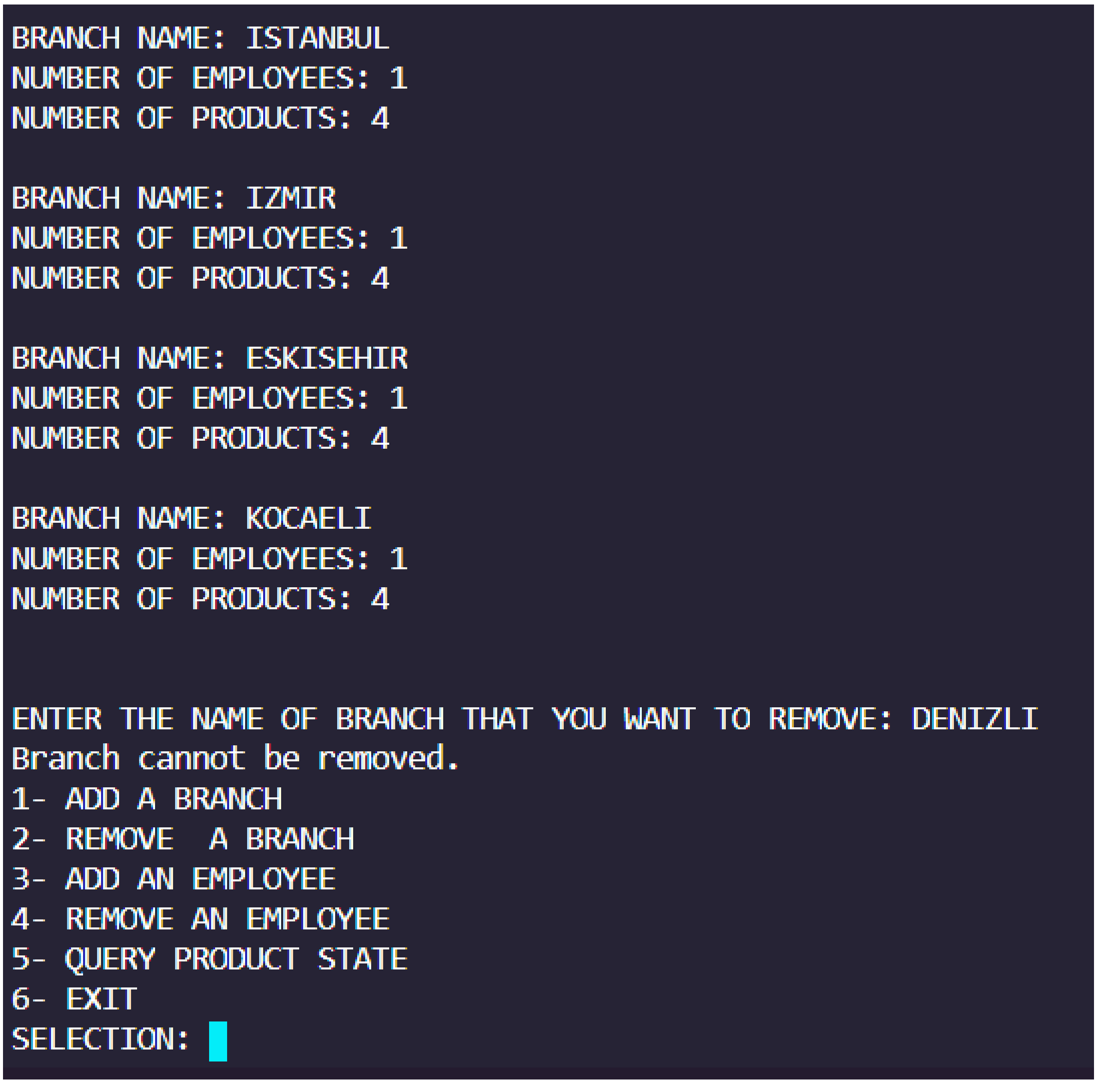
I create one company, 2 customers,4 branches, 4 employees for each branch and also some products to test the entire system.

5. RUNNING AND RESULTS



```
/**************** Welcome to FURNITURE WORLD Corporation ****************/
1- LOGIN AS CUSTOMER
2- LOGIN AS EMPLOYEEE
3- LOGIN AS ADMINISTRATOR
4- EXIT
SELECTION: 3
E-MAIL: atakan
PASSWORD: altin
E-mail or Password Wrong!!!
E-MAIL: 
```

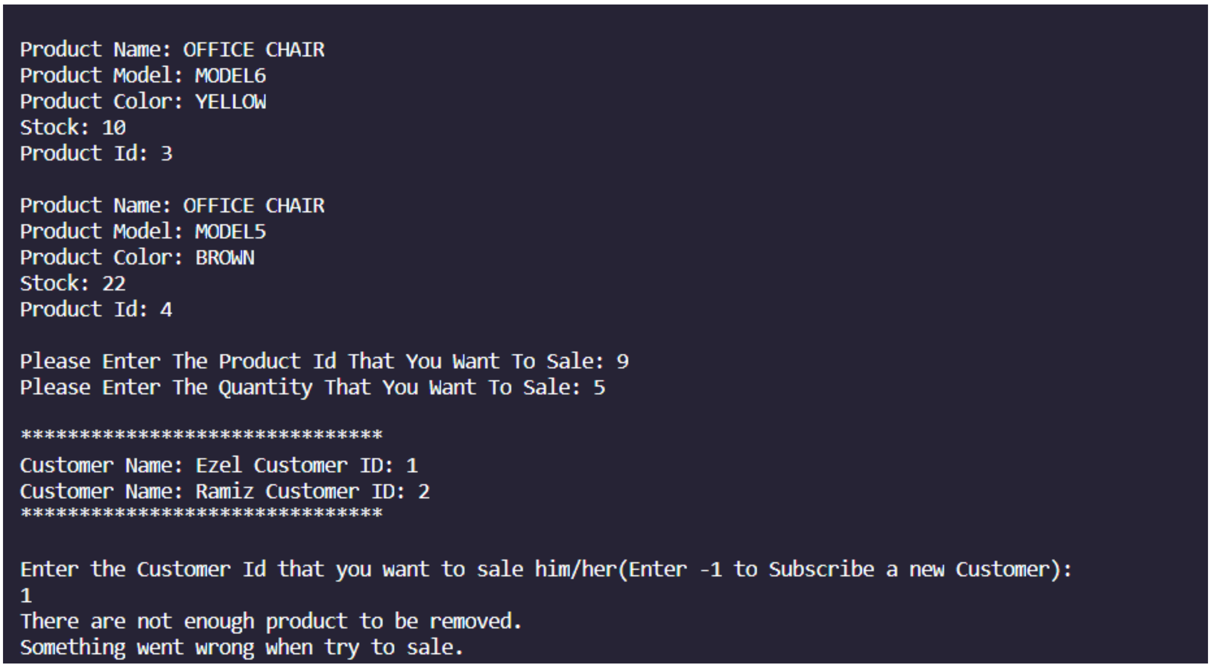
Login validation works for all user types.



```
BRANCH NAME: ISTANBUL
NUMBER OF EMPLOYEES: 1
NUMBER OF PRODUCTS: 4

BRANCH NAME: IZMIR
NUMBER OF EMPLOYEES: 1
NUMBER OF PRODUCTS: 4

BRANCH NAME: ESKISEHIR
NUMBER OF EMPLOYEES: 1
NUMBER OF PRODUCTS: 4

BRANCH NAME: KOCAELI
NUMBER OF EMPLOYEES: 1
NUMBER OF PRODUCTS: 4


ENTER THE NAME OF BRANCH THAT YOU WANT TO REMOVE: DENIZLI
Branch cannot be removed.
1- ADD A BRANCH
2- REMOVE  A BRANCH
3- ADD AN EMPLOYEE
4- REMOVE AN EMPLOYEE
5- QUERY PRODUCT STATE
6- EXIT
SELECTION:
```

Adding and removing branches, employees and products works fine.

Try to buy product that is not exist handled perfectly.



```
Product Name: OFFICE CHAIR
Product Model: MODEL6
Product Color: YELLOW
Stock: 10
Product Id: 3

Product Name: OFFICE CHAIR
Product Model: MODEL5
Product Color: BROWN
Stock: 22
Product Id: 4

Please Enter The Product Id That You Want To Sale: 9
Please Enter The Quantity That You Want To Sale: 5

*******************************
Customer Name: Ezel Customer ID: 1
Customer Name: Ramiz Customer ID: 2
*******************************

Enter the Customer Id that you want to sale him/her(Enter -1 to Subscribe a new Customer):
1
There are not enough product to be removed.
Something went wrong when try to sale.
```