جامعة بيروت العربية
BEIRUT ARAB UNIVERSITY

Beekeeper-Management-System

28.11.2022

Presented By : Hadi Youness Maher Lawand

ID: 202000489 202001472

Course Name: Data Structures

Course Number: CMPS347

Proposed to : Dr. Ali El-Zaart

TA: Hala Sweidan/Nada Baydoun

Faculty/Major: Science/Math & CS

Fall(2022-2023)

Table of Contents:

Abstract

The Beekeeper Management System is designed to help beekeepers manage the amount of data handled each day , allowing them to concentrate on their beekeeping side of their business ,This system can keep track of Bees/QueenBees/Hives/Apiaries/Stock/Sales/Customers , and Most Importantly Beekeepers ! This system has been developed using Java as its language and its data structures. The types used are Nodes , Linked Lists , Arrays , Arraylists ,HashMaps , HashSets.

Introduction

Topic Motivation:

The inspiration behind the Beekeeper Management System was actually Hadi's Cousin who is a beekeeper himself. He expressed how hard it was to keep track of all the Apiaries , Hives , Customers , Sales , Stock and of course the Bees. So we as developers set out to build and design a fully working Java program that is able to handle , manipulate and display all this data in a way that is User-Friendly to non-developers and that is also able to Write and View files after running. Multiple Beekeepers can Login/Register accounts by using an email and password.

General Idea:

Beekeeper Management System is a java software program that eases out the process of keeping track of all the Apiaries , Hives , Customers , Sales , Stock and of course the Bees and saving all the data and dates after runtime ends.

Project Development:

We started out by talking to the client "Hadi's Cousin" which is the most important step because it helped us understand the needs of a Beekeeper and we set out to meet those needs. After that We started designing a UML diagram that would involve all the classes and methods needed to fulfill the clients needs and beyond. Then we discussed the appropriate type of data structure to implement in each class. We worked together in person and through discord to build all the classes and share ideas , concerts and new methods to add.

Related Work and Technologies

Tech Stack Used:

Program Used:

Git

GitHub

VsCode

Eclipse

How it was used:

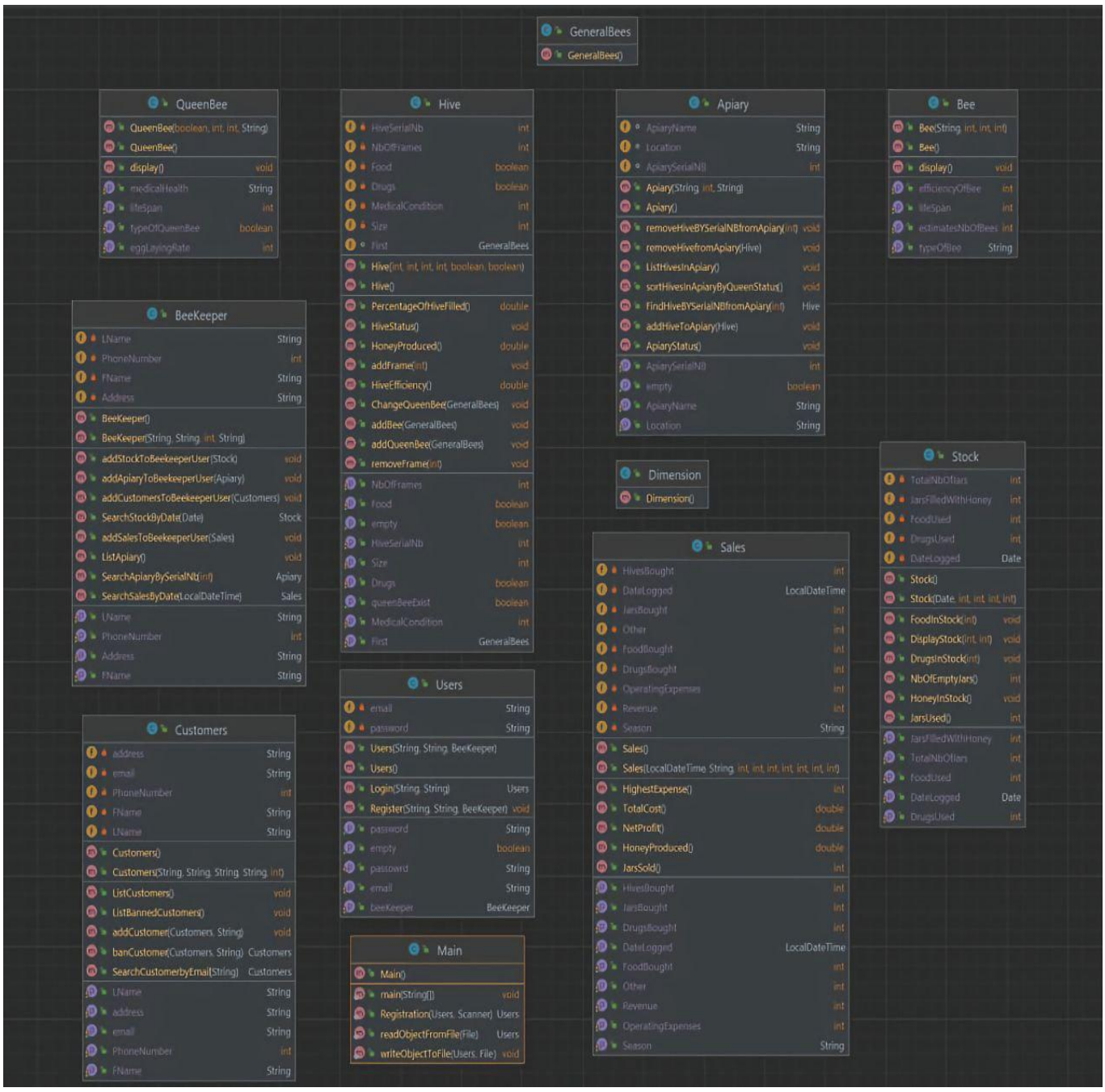
VsCode was our go to Java IDE because it gave us a bug free environment to build our code And helped us in debugging faster due to the availability of many extensions.

We used Eclipse to make sure our code works with multiple Java IDEs..

Git & GitHub was used to store the code in an online repository and for version control, you can find the repository at the following link: https://github.com/Hadious15/BMS.git https://github.com/MaherLawand/BeeKeeper-Management-System.git

Alternatively Scan this QrCode:

UML Diagram


GeneralBees
GeneralBees()
QueenBee
QueenBee(boolean, int, int, String)
QueenBee()
display() void
medicalHealth String
lifeSpan int
typeOfQueenBee boolean
eggLayingRate int
Hive
HiveSerialNb int
NbOfFrames int
Food boolean
Drugs boolean
MedicalCondition int
Size int
First GeneralBees
Hive(int, int, int, int, boolean, boolean)
Hive()
PercentageOfHiveFilled() double
HiveStatus() void
HoneyProduced() double
addFrame(int) void
HiveEfficiency() double
ChangeQueenBee(GeneralBees) void
addBee(GeneralBees) void
addQueenBee(GeneralBees) void
removeFrame(int) void
NbOfFrames int
Food boolean
empty boolean
HiveSerialNb int
Size int
Drugs boolean
queenBeeExist boolean
MedicalCondition int
First GeneralBees
Apiary
ApiaryName String
Location String
ApiarySerialNB int
Apiary(String, int, String)
Apiary()
removeHiveBYSerialNBfromApiary(int) void
removeHivefromApiary(Hive) void
ListHivesInApiary() void
sortHivesInApiaryByQueenStatus() void
findHiveBYSerialNBfromApiary(int) Hive
addHiveToApiary(Hive) void
ApiaryStatus() void
ApiarySerialNB int
empty boolean
ApiaryName String
Location String
Bee
Bee(String, int, int, int)
Bee()
display() void
efficiencyOfBee int
lifeSpan int
estimatesNbOfBees int
typeOfBee String
BeeKeeper
LName String
PhoneNumber int
FName String
Address String
BeeKeeper()
BeeKeeper(String, String, int, String)
addStockToBeekeeperUser(Stock) void
addApiaryToBeekeeperUser(Apiary) void
addCustomersToBeekeeperUser(Customers) void
SearchStockByDate(Date) Stock
addSalesToBeekeeperUser(Sales) void
ListApiary() void
SearchApiaryBySerialNb(int) Apiary
SearchSalesByDate(LocalDateTime) Sales
LName String
PhoneNumber int
Address String
FName String
Dimension
Dimension()
Stock
TotalNbOfJars int
JarsFilledWithHoney int
FoodUsed int
DrugsUsed int
DateLogged Date
Stock()
Stock(Date, int, int, int, int)
FoodInStock(int) void
DisplayStock(int, int) void
DrugsInStock(int) void
NbOfEmptyJars() int
HoneyInStock() void
JarsUsed() int
JarsFilledWithHoney int
TotalNbOfJars int
FoodUsed int
DateLogged Date
DrugsUsed int
Sales
HivesBought int
DateLogged LocalDateTime
JarsBought int
Other int
FoodBought int
DrugsBought int
OperatingExpenses int
Revenue int
Season String
Sales()
Sales(LocalDateTime, String, int, int, int, int, int, int, int)
HighestExpense() int
TotalCost() double
NetProfit() double
HoneyProduced() double
JarsSold() int
HivesBought int
JarsBought int
DrugsBought int
DateLogged LocalDateTime
FoodBought int
Other int
Revenue int
OperatingExpenses int
Season String
Users
email String
password String
Users(String, String, BeeKeeper)
Users()
Login(String, String) Users
Register(String, String, BeeKeeper) void
password String
empty boolean
password String
email String
beeKeeper BeeKeeper
Customers
address String
email String
PhoneNumber int
FName String
LName String
Customers()
Customers(String, String, String, String, int)
ListCustomers() void
ListBannedCustomers() void
addCustomer(Customers, String) void
banCustomer(Customers, String) Customers
SearchCustomerbyEmail(String) Customers
LName String
address String
email String
PhoneNumber int
FName String
Main
Main()
main(String[]) void
Registration(Users, Scanner) Users
readObjectFromFile(File) Users
writeObjectToFile(Users, File) void

The Code

Bee class:

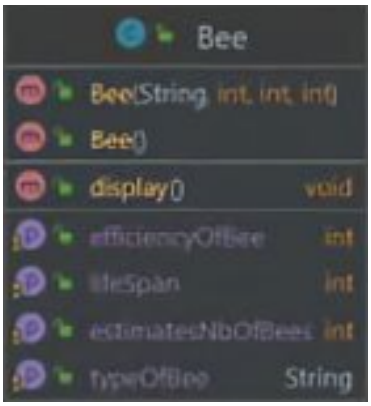
The Bee class is a Node that has a constructor

Which takes three *integers:*

- The (Est)Number of Bees
- The Efficiency of the Bees
- The Lifespan of the Bees

And a *String:*

- The Type of the Bees



The class also has a *display method* and *SetterAndGetters* .

Queen Bee class:

The Queen Bee class is a Node that has

a constructor that takes two *integers* :

- The Egg Laying Rate of the Queen Bee
- The Lifespan of the Queen Bee

And a *String:*

- A description of the Medical Condition ofThe
  added to the hive.

A *Boolean* :

- To Confirm that The Bee Is In fact A Queen .

The class also has a *display method* and *SetterAndGetters*.

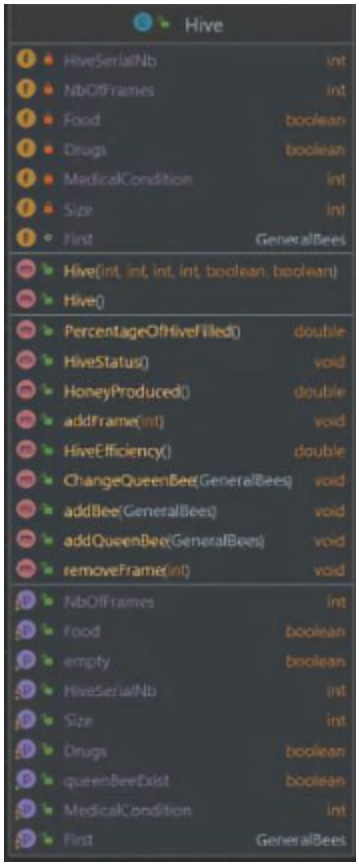
General Bee:

This class is used for Polymorphism inorder to add

An Object from Bee class and An Object from Queen Bee class in the

Same Linked list “Hive” to be talked about later.

Hive:



This class is a Linked List of A Singular Queen Bee

And many Bee Nodes.

It has a Constructor that takes four *integers*:

- Hive Serial Number
- Number of Frames
- Medical condition Rating(0-10)
- Size(Area) of the hive (Square Cm)

And Two *Booleans*:

- Bees Drugged
- Bees Fed

And Many *Methods:*

- PercentageOfHiveFilled( ):int Returns the % filled Of The Hive
- HiveStatus( ):Void displays everything about the Hive
- HoneyProduced( ):double Returns the amount of Honey Produced In Liters
- AddFrames(int):void Adds Frames to A hive
- RemoveFrames(int):void removes Frames to from hive

- HiveEfficiency( ):int Returns the Efficiency of the whole hive
- ChangeQueenBee(GeneralBees):void Changes the queen bee in a hive
- addBee(GeneralBees):void Add a Group of bees to a hive
- AddQueenBee(GeneralBees):void adds the queen bee to a hive
- SettersAndGetters

The Hive Code:

```
import java.io.Serializable;


public class Hive implements Serializable{

    GeneralBees First;

    private int Size;

    private int HiveSerialNb;

    private int MedicalCondition;

    private int NbOfFrames;

    private boolean Food;

    private boolean Drugs;


    public Hive(){

        First=null;

        Size=0;

        HiveSerialNb=0;

        MedicalCondition=0;

        NbOfFrames=0;

        Food=false;

        Drugs=false;

    }

    public int getSize() {
```

```
        return Size;
    }
    public void setSize(int size) {
        Size = size;
    }
    public int getMedicalCondition() {
        return MedicalCondition;
    }
    public void setMedicalCondition(int medicalCondition) {
        MedicalCondition = medicalCondition;
    }
    public int getNbOfFrames() {
        return NbOfFrames;
    }
    public void setNbOfFrames(int nbOfFrames) {
        NbOfFrames = nbOfFrames;
    }
    public boolean isFood() {
        return Food;
    }
    public void setFood(boolean food) {
        Food = food;
    }
    public boolean isDrugs() {
        return Drugs;
    }
    public void setDrugs(boolean drugs) {
        Drugs = drugs;
    }
```

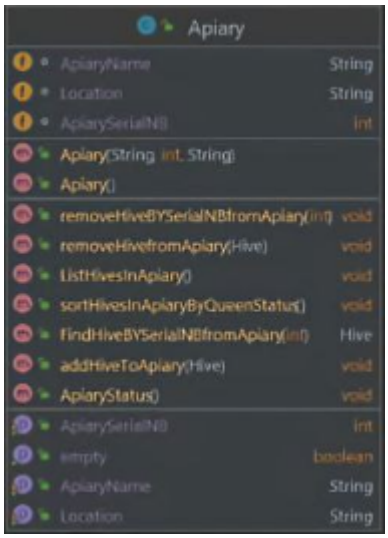
```
    public Hive(int Size,int HiveSerialNb, int MedicalCondition, int
NbOfFrames,boolean Food,boolean Drugs){
        First=null;
        this.Size=Size;
        this.HiveSerialNb=HiveSerialNb;
        this.MedicalCondition=MedicalCondition;
        this.NbOfFrames=NbOfFrames;
        this.Food=Food;
        this.Drugs=Drugs;
    }
    public GeneralBees getFirst() {
        return First;
    }
    public void setFirst(GeneralBees first) {
        First = first;
    }
    public void addBee(GeneralBees bee){
        double Percent=0.0;
        Percent=((bee.EstimatesNbOfBees*0.23)/Size)*100;
        if(PercentageOfHiveFilled()+Percent<=100) {
        GeneralBees current=First;
        if(First==null){
            First=bee;
        }else{
            while(current.next!=null){
                current=current.next;
            }
            current.next=bee;
        }
```

```
        }else {
            System.out.println(ANSI_RED + "Number of Bees Exceeded the
Hive Capacity!" + ANSI_RESET);
            System.out.println(ANSI_YELLOW +
"--------------------------------------------" + ANSI_RESET);
        }
    }
    public boolean isEmpty() {
        return First==null;
    }
    public boolean isQueenBeeExist(){
        return First.TypeOfQueenBee;
    }
    public void addQueenBee(GeneralBees bee){
            First=bee;
    }
    public void ChangeQueenBee(GeneralBees bee) {
         bee.next=First.next;
         First=bee;
    }
    public double PercentageOfHiveFilled() {
        GeneralBees Current=First;
        int sum=0;
        double Percent=0.0;
        while(Current!=null) {
            sum+=Current.EstimatesNbOfBees;
            Current=Current.next;
        }
        Percent=((sum*0.23)/Size)*100;
```

```
        return Percent;
    }
    public double HiveEfficiency() {
        GeneralBees Current=First;
        int sum=0;
        double Efficiency=0.0;
        while(Current!=null) {
            sum+=Current.EstimatesNbOfBees;
            Current=Current.next;
        }
        Efficiency=HoneyProduced()/sum;
        return Efficiency;
    }
    public double HoneyProduced(){
        GeneralBees Current=First;
        double Honey=0.0;
        while(Current!=null) {
            Honey+=(Current.EstimatesNbOfBees*Current.EfficiencyOfBee);
            Current=Current.next;
        }
        return Honey;
    }
    public void HiveStatus(){
        System.out.println(ANSI_CYAN + "Hive: " + ANSI_GREEN +
getHiveSerialNb() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Size: " +ANSI_GREEN + getSize() +
ANSI_RESET);
        System.out.println(ANSI_CYAN + "Medical Condition (1-10) : "
+ANSI_GREEN + getMedicalCondition() + ANSI_RESET);
```

```
        System.out.println(ANSI_CYAN + "Number Of Frames: " +ANSI_GREEN +
getNbOfFrames() + ANSI_RESET);

        System.out.println(ANSI_CYAN + "Fed: " +ANSI_GREEN + isFood() +
ANSI_RESET);

        System.out.println(ANSI_CYAN + "Drugged: " +ANSI_GREEN + isDrugs()
+ ANSI_RESET);

        System.out.println(ANSI_CYAN + "Percentage of Hive Filled % : "
+ANSI_GREEN + PercentageOfHiveFilled() +" %" + ANSI_RESET);

        System.out.println(ANSI_CYAN + "Hive Efficiency : " +ANSI_GREEN +
HiveEfficiency() + ANSI_RESET);

        System.out.println(ANSI_CYAN + "Honey Produced : " +ANSI_GREEN +
HoneyProduced()+" L" + ANSI_RESET);

        GeneralBees BeeCurrent=First;

        while(BeeCurrent!=null){

            if(BeeCurrent.TypeOfQueenBee) {

                System.out.println(ANSI_CYAN+ "Queen Bee: "+ ANSI_GREEN +
BeeCurrent.TypeOfQueenBee +"\n" + ANSI_CYAN + "Egg Laying Rate: " +
ANSI_GREEN + BeeCurrent.EggLayingRate + "\n" + ANSI_CYAN + "LifeSpan: " +
ANSI_GREEN + BeeCurrent.LifeSpan + "\n" + ANSI_CYAN +  "Medical Health: "
+ ANSI_GREEN + BeeCurrent.MedicalHealth + ANSI_RESET);

            }else {

                System.out.println(ANSI_CYAN+ "Type of Bee: "+ ANSI_GREEN
+ BeeCurrent.TypeOfBee +"\n" + ANSI_CYAN + "Efficiency of Bee: " +
ANSI_GREEN + BeeCurrent.EfficiencyOfBee + "\n" + ANSI_CYAN + "LifeSpan: "
+ ANSI_GREEN + BeeCurrent.LifeSpan + "\n" + ANSI_CYAN + "Estimated Number
Of Bees: " + ANSI_GREEN + BeeCurrent.EstimatesNbOfBees + ANSI_RESET);

            }

            BeeCurrent=BeeCurrent.next;

        }


    }

    public void addFrame(int i) {

        NbOfFrames = NbOfFrames +i;
```

```
        System.out.println(" Frames Added: " + i );


    }
    public void removeFrame(int i) {
        NbOfFrames = NbOfFrames -i;
        System.out.println(" Frames Removed: " + i );


    }
    public int getHiveSerialNb() {


        return HiveSerialNb;
    }
    public void setHiveSerialNb(int hiveSerialNb) {
        HiveSerialNb = hiveSerialNb;
    }


    public static final String ANSI_RESET = "\u001B[0m";
    public static final String ANSI_BLACK = "\u001B[30m";
    public static final String ANSI_RED = "\u001B[31m";
    public static final String ANSI_GREEN = "\u001B[32m";
    public static final String ANSI_YELLOW = "\u001B[33m";
    public static final String ANSI_BLUE = "\u001B[34m";
    public static final String ANSI_PURPLE = "\u001B[35m";
    public static final String ANSI_CYAN = "\u001B[36m";
    public static final String ANSI_WHITE = "\u001B[37m";



}
```



Apiary:

An Apiary is a collection of bee hives .

Apiary Uses An Arraylist of LinkedLists "Hive"

This Class has a Constructor that takes

Two *Strings:*

The Apiary Name

The Location of the Apiary

One *Integer:*

- The Serial Number of the apiary

It Has Many methods:

- removeHivefromApiary(Hive l):void Removes A Specific Hive From the Apiary
- removeHiveBYSerialNBfromApiary(int l):void Removes A Hive Using its Serial Number
- FindHiveBYSerialNBfromApiary(int l):Hive Finds a Hive Using its serial number
- sortHivesInApiaryByQueenStatus( ):void Sorts the Hives according to the presence of the QueenBee
- ListHivesInApiary( ):void Displays All the Hives in an Apiary
- ApiaryStatus( ):void Displays the Name , Serial Number , and Location
- addHiveToApiary(Hive l):void Adds a Hive To An Apiary
- SettersAndGetters

The Apiary Code :

```
import java.io.Serializable;

import java.util.ArrayList;

public class Apiary implements Serializable{
```

```
    ArrayList<Hive> Apiaryone=new ArrayList<Hive>();
    String ApiaryName ;
    int ApiarySerialNB;
    String Location;

    public Apiary() {
        ApiaryName="";
        ApiarySerialNB=0;
        Location="";
    }

    public Apiary(String apiaryName, int apiarySerialNB, String location)
{
        ApiaryName = apiaryName;
        ApiarySerialNB = apiarySerialNB;
        Location = location;

    }

    public String getApiaryName() {
        return ApiaryName;
    }

    public void setApiaryName(String apiaryName) {
        ApiaryName = apiaryName;
    }

    public int getApiarySerialNB() {
        return ApiarySerialNB;
```

```
    }

    public void setApiarySerialNB(int apiarySerialNB) {
        ApiarySerialNB = apiarySerialNB;
    }

    public String getLocation() {
        return Location;
    }

    public void setLocation(String location) {
        Location = location;
    }

    public void addHiveToApiary(Hive l) {
        Apiaryone.add(l);
    }
    public boolean isEmpty(){
        return Apiaryone.size()==0;
    }

   public void removeHivefromApiary(Hive l) {
        Apiaryone.remove(l);
        return;
   }

    public void removeHiveBYSerialNBfromApiary(int l) {
       for (int i = 0; i < Apiaryone.size();i++)
       {
```

```
            if(Apiaryone.get(i).getHiveSerialNb() == l ) {
                Apiaryone.remove(i);
            }
        }
    }

    public Hive FindHiveBYSerialNBfromApiary(int l) {
        for (int i = 0; i < Apiaryone.size();i++)
           {
          if(  Apiaryone.get(i).getHiveSerialNb() == l ) {
              return Apiaryone.get(i)     ;
          }
           }
        return null;
    }


    public void  sortHivesInApiaryByQueenStatus() {
         for (int i = 0; i < Apiaryone.size();i++)
           {
          if( !Apiaryone.get(i).isQueenBeeExist()) {
              for (int j = i; j < Apiaryone.size();j++)
               {
              if( Apiaryone.get(j).isQueenBeeExist()) {
                  Hive temp = Apiaryone.get(i);
                  Hive temp2 = Apiaryone.get(j);
                  Apiaryone.set(i, temp2);
                  Apiaryone.set(j, temp);
              }
```

```
        }
    }
          }
    }
    public void ListHivesInApiary() {
        for (int i = 0; i < Apiaryone.size();i++)
           {
             Apiaryone.get(i).HiveStatus();
           }
    }
    public void ApiaryStatus(){
        System.out.println(ANSI_CYAN + "Apiary: " + ANSI_GREEN  +
getApiarySerialNB() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Name: "  + ANSI_GREEN +
getApiaryName() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Location: "+  ANSI_GREEN +
getLocation() + ANSI_RESET);
    }

    public static final String ANSI_RESET = "\u001B[0m";
    public static final String ANSI_BLACK = "\u001B[30m";
    public static final String ANSI_RED = "\u001B[31m";
    public static final String ANSI_GREEN = "\u001B[32m";
    public static final String ANSI_YELLOW = "\u001B[33m";
    public static final String ANSI_BLUE = "\u001B[34m";
    public static final String ANSI_PURPLE = "\u001B[35m";
    public static final String ANSI_CYAN = "\u001B[36m";
    public static final String ANSI_WHITE = "\u001B[37m";

}
```

Sales:

The Sales Class has a constructor that takes

Seven *Integers* :

- Number Of Hives Bought
- Number of Jars Bought
- Other Expenses
- Operating Expenses
- Food Bought
- Drugs Bought
- Revenue

A *String:*

- The Season

A *LocalDateTime Object*:

- DateLogged

This Class Has Many *Methods:*

- NetProfit():double Calculates the Net Profit
- TotalCost():double Calculates the Total Cost
- HighestExpense():Int Returns And Displays the Highest Expense
- JarsSold( ):int Return The Number of Jars Sold
- HoneyProduced():double Return the Amount of Honey Produced in Liters
- ListSales():void Displays Everything Related to sales
- SettersAndGetters

The Sales Code :

```
import java.io.Serializable;

import java.util.Date;


public class Sales implements Serializable{

    private Date DateLogged;

    private String Season;

    private int Revenue;

    private int HivesBought;

    private int JarsBought;

    private int FoodBought;

    private int DrugsBought;

    private int OperatingExpenses;

    private int Other;

    public Sales() {

        super();

        DateLogged =null;

        Season ="";

        Revenue = 0;

        HivesBought = 0;

        JarsBought = 0;

        FoodBought = 0;

        DrugsBought = 0;

        OperatingExpenses = 0;

        Other = 0;

    }

    public Sales(Date now,String season, int revenue, int hivesBought, int
jarsBought, int foodBought, int drugsBought,
```

```
        int operatingExpenses, int other) {
    super();
    DateLogged = now;
    Season = season;
    Revenue = revenue;
    HivesBought = hivesBought;
    JarsBought = jarsBought;
    FoodBought = foodBought;
    DrugsBought = drugsBought;
    OperatingExpenses = operatingExpenses;
    Other = other;
}
public Date getDateLogged() {
    return DateLogged;
}
public void setDateLogged(Date dateLogged) {
    DateLogged = dateLogged;
}
public String getSeason() {
    return Season;
}
public void setSeason(String season) {
    Season = season;
}
public int getRevenue() {
    return Revenue;
}
public void setRevenue(int revenue) {
    Revenue = revenue;
```

```
    }
    public int getHivesBought() {
        return HivesBought;
    }
    public void setHivesBought(int hivesBought) {
        HivesBought = hivesBought;
    }
    public int getJarsBought() {
        return JarsBought;
    }
    public void setJarsBought(int jarsBought) {
        JarsBought = jarsBought;
    }
    public int getFoodBought() {
        return FoodBought;
    }
    public void setFoodBought(int foodBought) {
        FoodBought = foodBought;
    }
    public int getDrugsBought() {
        return DrugsBought;
    }
    public void setDrugsBought(int drugsBought) {
        DrugsBought = drugsBought;
    }
    public int getOperatingExpenses() {
        return OperatingExpenses;
    }
    public void setOperatingExpenses(int operatingExpenses) {
```

```
        OperatingExpenses = operatingExpenses;
    }
    public int getOther() {
        return Other;
    }
    public void setOther(int other) {
        Other = other;
    }





    public double NetProfit() {
        double profit =getRevenue() - TotalCost();
        return profit;
    }

    public double TotalCost() {
        double sum = 150*getHivesBought()+
1*getJarsBought()+75*getFoodBought()+25*getDrugsBought()+getOperatingExpen
ses()+getOther();
        return sum;
    }

    public int HighestExpense() {
        int[] a = {150*getHivesBought()
,1*getJarsBought(),75*getFoodBought(),25*getDrugsBought(),getOperatingExpe
nses(),getOther()};
        int max = a[0];
        int q=0;
        for(int i = 0;i<a.length;i++) {
```

```
            if(a[i]>max) {
                max=a[i];
                q=i;
            }
        }
        if(q==0) {
            System.out.println( ANSI_CYAN + "The Highest expense is hives
built : " + ANSI_GREEN +max + "$" + ANSI_RESET);
        }
        else if(q==1) {
            System.out.println( ANSI_CYAN + "The Highest expense is jars
bought : " + ANSI_GREEN +max +"$" + ANSI_RESET);
        }
        else if(q==2) {
            System.out.println( ANSI_CYAN + "The Highest expense is food
bought : " + ANSI_GREEN +max + "$" + ANSI_RESET);
        }
        else if(q==3) {
            System.out.println( ANSI_CYAN + "The Highest expense is Drugs
bought : " + ANSI_GREEN +max + "$" + ANSI_RESET);
        }
        else if(q==4) {
            System.out.println( ANSI_CYAN + "The Highest expense is
Operating the business : " + ANSI_GREEN +max + "$" + ANSI_RESET);
        }
        else if(q==5) {
            System.out.println( ANSI_CYAN + "The Highest expense is
Miscellaneous Fees : " + ANSI_GREEN +max +"$" + ANSI_RESET);
        }
        return max;
```

```
    }

    public int JarsSold() {
        return getRevenue()/5;
    }

    public double HoneyProduced(){
        return JarsSold()*0.25;
    }
    public void ListSales(){
        System.out.println(ANSI_CYAN + "Date: " + ANSI_GREEN +
getDateLogged() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Season: " + ANSI_GREEN +
getSeason() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Revenue: " + ANSI_GREEN +
getRevenue()+" $" + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Hives Bought: " + ANSI_GREEN +
getHivesBought() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Jars Bought: " + ANSI_GREEN +
getJarsBought() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Food Bought: " + ANSI_GREEN +
getFoodBought() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Drugs Bought: " + ANSI_GREEN +
getDrugsBought() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Operating Expenses: " + ANSI_GREEN
+ getOperatingExpenses() +" $"+ ANSI_RESET);
        System.out.println(ANSI_CYAN + "Other Expenses: " + ANSI_GREEN +
getOther() +" $"+ ANSI_RESET);
        System.out.println(ANSI_CYAN + "Net Profit: " + ANSI_GREEN +
NetProfit()+" $" + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Total Cost: " + ANSI_GREEN +
TotalCost()+" $" + ANSI_RESET);
```

```
        HighestExpense();
        System.out.println(ANSI_CYAN + "Jars Sold: " + ANSI_GREEN +
JarsSold() + ANSI_RESET);
        System.out.println(ANSI_CYAN + "Honey Produced: " + ANSI_GREEN +
HoneyProduced() +" L"+ ANSI_RESET);
    }
    public static final String ANSI_RESET = "\u001B[0m";
    public static final String ANSI_BLACK = "\u001B[30m";
    public static final String ANSI_RED = "\u001B[31m";
    public static final String ANSI_GREEN = "\u001B[32m";
    public static final String ANSI_YELLOW = "\u001B[33m";
    public static final String ANSI_BLUE = "\u001B[34m";
    public static final String ANSI_PURPLE = "\u001B[35m";
    public static final String ANSI_CYAN = "\u001B[36m";
    public static final String ANSI_WHITE = "\u001B[37m";
}
```

Stock

The Stock Class Has A Constructor that takes

Four *integers* :

- Total Number Of Jars
- Jars Filled With Honey
- Food Used
- Drugs Used

One *Date Object*:

- DateLogged

This Class has many *methods*:

- FoodInStock(int):void Prints the Amount of food in stock
- DrugsInStock(int):void Print the Amount of drugs in stock
- JarsUsed( ):int Returns theAmount of Jars Used
- DisplayStock( ):void Displays Everything in Stock
- SettersAndGetters

The Stock Code :

```
import java.io.Serializable;

import java.util.Date;


public class Stock implements Serializable{

    private Date DateLogged;

    private int TotalNbOfJars;

    private int JarsFilledWithHoney;

    private int FoodUsed;
```

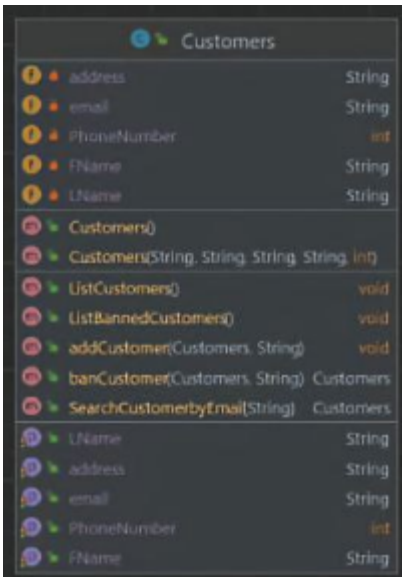
```
    private int DrugsUsed;
    public Date getDateLogged() {
        return DateLogged;
    }
    public Stock() {
        super();
        DateLogged =null;
        TotalNbOfJars = 0;
        JarsFilledWithHoney = 0;
        FoodUsed = 0;
        DrugsUsed = 0;
    }
    public Stock(Date dateLogged, int totalNbOfJars, int
jarsFilledWithHoney, int foodInStock, int drugsInStock) {
        super();
        DateLogged = dateLogged;
        TotalNbOfJars = totalNbOfJars;
        JarsFilledWithHoney = jarsFilledWithHoney;
        FoodUsed = foodInStock;
        DrugsUsed = drugsInStock;
    }
    public void setDateLogged(Date dateLogged) {
        DateLogged = dateLogged;
    }
    public int getTotalNbOfJars() {
        return TotalNbOfJars;
    }
    public void setTotalNbOfJars(int totalNbOfJars) {
        TotalNbOfJars = totalNbOfJars;
```

```
    }
    public int getJarsFilledWithHoney() {
        return JarsFilledWithHoney;
    }
    public void setJarsFilledWithHoney(int jarsFilledWithHoney) {
        JarsFilledWithHoney = jarsFilledWithHoney;
    }
    public int getFoodUsed() {
        return FoodUsed;
    }
    public void setFoodUsed(int foodInStock) {
        FoodUsed = foodInStock;
    }
    public int getDrugsUsed() {
        return DrugsUsed;
    }
    public void setDrugsUsed(int drugsInStock) {
        DrugsUsed = drugsInStock;
    }
    public void HoneyInStock() {
         System.out.println(ANSI_GREEN + getJarsFilledWithHoney()*0.25 + "
L" + ANSI_CYAN + "Honey In Stock" + ANSI_RESET);
    }
    public int NbOfEmptyJars() {
        return getTotalNbOfJars()-getJarsFilledWithHoney();
    }
    public void FoodInStock(int n) {
        // in main n = sales.getFoodBought
        System.out.print(ANSI_GREEN);
```

```
        System.out.print(n-getFoodUsed());
    }
    public void DrugsInStock(int n) {
        System.out.print(ANSI_GREEN);
        System.out.print(n-getDrugsUsed());
    }
    public int JarsUsed() {
        return JarsFilledWithHoney;


    }
    public void DisplayStock(int n , int m) {
        System.out.println(ANSI_CYAN + "On " + ANSI_GREEN +
getDateLogged() + ANSI_CYAN + ": \nIn Stock: " + ANSI_GREEN +
getTotalNbOfJars() + ANSI_CYAN + " jars \n" + ANSI_GREEN +
getJarsFilledWithHoney() + ANSI_CYAN + " jars filled with honey \n" +
ANSI_GREEN + getFoodUsed() + ANSI_CYAN +  " Food used \n" + ANSI_GREEN +
getDrugsUsed() + ANSI_CYAN + " Drugs Used \n" + ANSI_GREEN +  JarsUsed() +
ANSI_CYAN + " Jars Used \n" + ANSI_GREEN + NbOfEmptyJars() + ANSI_CYAN + "
empty jars" + ANSI_RESET);
        FoodInStock(n);
        System.out.println(ANSI_CYAN + " Food in Stock" + ANSI_RESET);
        DrugsInStock(m);
        System.out.println(ANSI_CYAN + " Drugs in Stock" + ANSI_RESET);
        HoneyInStock();
    }
    public static final String ANSI_RESET = "\u001B[0m";
    public static final String ANSI_BLACK = "\u001B[30m";
    public static final String ANSI_RED = "\u001B[31m";
    public static final String ANSI_GREEN = "\u001B[32m";
    public static final String ANSI_YELLOW = "\u001B[33m";
    public static final String ANSI_BLUE = "\u001B[34m";
```

```
    public static final String ANSI_PURPLE = "\u001B[35m";

    public static final String ANSI_CYAN = "\u001B[36m";

    public static final String ANSI_WHITE = "\u001B[37m";

}
```

Customers



The Customer class Uses Two HashMaps “Active”

And “Banned” Customers

To Avoid duplicates

has a constructor that takes

One *integer:*

- Phone number

And Four Strings:

- First Name
- Last Name
- Email
- Address

It Also Has Many *Methods*:

- addCustomer(Customers,String):void Adds a Customer to the Active Customers HashMap and An Email As a key
- banCustomer(Customers,String):customer Searches for the customer in the Active Hashmap and move it to the Banned HashMap
- Customers SearchCustomerbyEmail(String):customer Searches for a Customer with a pacific email and returns it
- ListCustomers():void lists all the attributes of all the Active customers
- SettersAndGetters

- ListBannedCustomers():void lists all the attributes of all the Banned customers

The Customers Class Code:

```
import java.io.Serializable;

import java.util.Arrays;

import java.util.Comparator;

import java.util.HashMap;


public class Customers implements Serializable{

    HashMap <String,Customers> ActiveCustomers = new HashMap<>();

    HashMap <String,Customers> BannedCustomers = new HashMap<>();

    private String FName;

    private String LName;

    private String email;

    private String address;

    private int PhoneNumber;

    public Customers() {

        super();

        FName ="";

        LName ="";

        email ="";

        address ="";

        PhoneNumber=0;

    }

    public String getFName() {

        return FName;

    }

    public void setFName(String fName) {
```

```
        FName = fName;
    }
    public String getLName() {
        return LName;
    }
    public void setLName(String lName) {
        LName = lName;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
    public int getPhoneNumber() {
        return PhoneNumber;
    }
    public void setPhoneNumber(int phoneNumber) {
        PhoneNumber = phoneNumber;
    }
    public Customers(String fName, String lName, String email, String
address, int phoneNumber) {
        super();
```

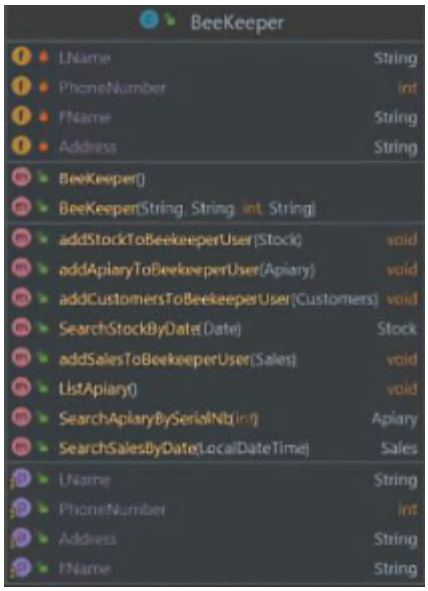
```
        FName = fName;
        LName = lName;
        this.email = email;
        this.address = address;
        PhoneNumber = phoneNumber;
    }
    public void addCustomer(Customers s,String email) {
        if(ActiveCustomers.keySet().contains(email)){
            System.out.println(ANSI_RED+"Already An Active
Customer!!"+ANSI_RESET);
        }else if(BannedCustomers.keySet().contains(email)){
            System.out.println(ANSI_RED+"Customer "+ANSI_GREEN  +
s.getFName() + " " + s.getLName()+ANSI_RED+ " Banned!"+ANSI_RESET);
        }else{
            ActiveCustomers.put(email, s);
            System.out.println(ANSI_YELLOW+"Successfully Added Customer
"+ANSI_GREEN + s.getFName() + " " + s.getLName()+ANSI_RESET);
        }
    }
    public Customers banCustomer(Customers s,String email) {
        Customers banned=null;
        if(!BannedCustomers.keySet().contains(email) &&
ActiveCustomers.keySet().contains(email)){
            BannedCustomers.put(email, s);
            banned=s;
            ActiveCustomers.remove(email, s);
            System.out.println(ANSI_YELLOW+"Successfully Banned Customer
"+ANSI_GREEN + s.getFName() + " " + s.getLName()+ANSI_RESET);
        }else if(!ActiveCustomers.keySet().contains(email)){
```

```
            System.out.println(ANSI_RED+"Customer Doesnt
Exist!"+ANSI_RESET);
        }else{
            System.out.println(ANSI_RED+"Customer Already
Banned!"+ANSI_RESET);
        }
        return banned;
    }
    public Customers SearchCustomerbyEmail(String email) {
        Customers[] customers = ActiveCustomers.values().toArray(new
Customers[ActiveCustomers.size()]);
        // for(int i=0;i<customers.length;i++){
        //  System.out.println(customers[i].email);
        // }
        Arrays.sort(customers, new Comparator<Customers>() {
            @Override
            public int compare(Customers o1, Customers o2) {
                return o1.getEmail().compareTo(o2.getEmail());
            }
        });
        int l = 0, r = customers.length - 1;
        while (l <= r) {
            int m = (l + r) / 2;
            if (customers[m].getEmail().compareTo(email) < 0) {
                l = m + 1;
            } else if (customers[m].getEmail().compareTo(email) > 0) {
                r = m - 1;
            }
            else{
                return customers[m];
```

```
            }
    }
    return null;
}


    public void ListCustomers() {
        System.out.println(ANSI_YELLOW + "Active Customers:" +
ANSI_RESET);
        for (Customers s : ActiveCustomers.values()) {
            System.out.println(ANSI_CYAN + "First Name:" + ANSI_GREEN + "
" + s.getFName() + ANSI_RESET);
            System.out.println(ANSI_CYAN + "Last Name:" + ANSI_GREEN + " "
+ s.getLName() + ANSI_RESET);
            System.out.println(ANSI_CYAN + "Email:" + ANSI_GREEN + " " +
s.getEmail() + ANSI_RESET);
            System.out.println(ANSI_CYAN + "Address:" + ANSI_GREEN + " " +
s.getAddress() + ANSI_RESET);
            System.out.println(ANSI_CYAN + "Phone Number:" + ANSI_GREEN +
" " + s.getPhoneNumber() + ANSI_RESET);
            System.out.println(ANSI_YELLOW +
"-------------------------------------" + ANSI_RESET);


        }
    }
    public void ListBannedCustomers() {
        System.out.println(ANSI_YELLOW + "Banned Customers: " +
ANSI_RESET);
        for (Customers s : BannedCustomers.values()) {
            System.out.println(ANSI_CYAN + "First Name: " + ANSI_GREEN + "
" +  s.getFName() + ANSI_RESET);
            System.out.println(ANSI_CYAN + "Last Name: " + ANSI_GREEN + "
" +  s.getLName() + ANSI_RESET);
```

```
            System.out.println(ANSI_CYAN + "Email: " + ANSI_GREEN + " " +
s.getEmail() + ANSI_RESET);

            System.out.println(ANSI_CYAN + "Address: " + ANSI_GREEN + " "
+  s.getAddress() + ANSI_RESET);

            System.out.println(ANSI_CYAN + "Phone Number: " + ANSI_GREEN +
" " + s.getPhoneNumber() + ANSI_RESET);

            System.out.println(ANSI_YELLOW +
"-------------------------------------" + ANSI_RESET);


        }

    }


    public static final String ANSI_RESET = "\u001B[0m";

    public static final String ANSI_BLACK = "\u001B[30m";

    public static final String ANSI_RED = "\u001B[31m";

    public static final String ANSI_GREEN = "\u001B[32m";

    public static final String ANSI_YELLOW = "\u001B[33m";

    public static final String ANSI_BLUE = "\u001B[34m";

    public static final String ANSI_PURPLE = "\u001B[35m";

    public static final String ANSI_CYAN = "\u001B[36m";

    public static final String ANSI_WHITE = "\u001B[37m";



}
```

Beekeeper:



This class Utilizes an Arraylist of Stock , an Arraylist of Apiary , And a HashMap of Sales And Date as A key

Beekeeper has a constructor that takes one *integer:*

- Phone Number

And three *Strings:*

- First Name
- Last Name
- Address

This Class Has Many *methods :*

- addApiaryToBeekeeperUser(Apiary):void Adds a Apiary to a Specific Beekeeper
- addStockToBeekeeperUser(Stock):void Adds a Stock to a Specific Beekeeper
- addSalesToBeekeeperUser(Date ,Sales):void Adds a Sales to a specific Beekeeper
- addCustomersToBeekeeperUser(Customers):void Adds a Customer to a Specific Beekeeper
- SearchApiaryBySerialNb(int):apiary Finds and returns an Apiary using its specific Serial Number
- ListApiary():void Displays all the Apiaries a Beekeeper has
- ListAllCustomers():void Displays all the Customer a Beekeeper has

- SearchStockByDate(Date):stock Returns Stock with this specific Date
- ListAllStock():void Displays all the Stock a Beekeeper has
- SearchSalesByDate(Date):sales Returns Sales with this specific Date
- ListAllSales():void Displays all the Sales a Beekeeper has
- FindApiaryBYSerialNBfromHive(int):apiary finds and returns an apiary using the serial number of the hive
- SettersAndGetters

The Beekeeper Code:

```
import java.io.Serializable;

import java.util.Date;

import java.util.ArrayList;

import java.util.HashMap;


public class BeeKeeper implements Serializable {

    private String FName;

    private String LName;

    private int PhoneNumber;

    private String Address;

    public BeeKeeper(){

        FName="";

        LName="";

        PhoneNumber=0;

        Address="";

    }

    public BeeKeeper(String FName,String LName,int PhoneNumber,String
Address){

        this.FName=FName;
```

```
        this.LName=LName;
        this.PhoneNumber=PhoneNumber;
        this.Address=Address;
    }
    public String getFName() {
        return FName;
    }
    public void setFName(String fName) {
        FName = fName;
    }
    public String getLName() {
        return LName;
    }
    public void setLName(String lName) {
        LName = lName;
    }
    public int getPhoneNumber() {
        return PhoneNumber;
    }
    public void setPhoneNumber(int phoneNumber) {
        PhoneNumber = phoneNumber;
    }
    public String getAddress() {
        return Address;
    }
    public void setAddress(String address) {
        Address = address;
    }
    ArrayList<Apiary> apiary = new ArrayList<Apiary>();
```

```
    ArrayList<Stock> stock = new ArrayList<Stock>();

    HashMap<Date,Sales> sales = new HashMap<>();

    Customers s;

    //Customers

     public void addCustomersToBeekeeperUserNoPrint(Customers c){

            s=c;


        }

    public void addApiaryToBeekeeperUser(Apiary A){

        apiary.add(A);

        System.out.println(ANSI_YELLOW +"Successfully Added Apiary " +
ANSI_GREEN + A.getApiarySerialNB() + ANSI_RESET);

    }

    public void addStockToBeekeeperUser(Stock st){

        stock.add(st);

        System.out.println(ANSI_YELLOW +"Successfully Added Stock! " +
ANSI_RESET);

        System.out.println(ANSI_YELLOW
+"----------------------------------------" + ANSI_RESET);


    }

    public void addSalesToBeekeeperUser(Date sdf,Sales s){

        sales.put(sdf,s);

        System.out.println(ANSI_YELLOW +"Successfully Added Sales! " +
ANSI_RESET);

    }

    public void addCustomersToBeekeeperUser(Customers c){

        s=c;

        System.out.println(ANSI_YELLOW +"Successfully Added Customers!" +
ANSI_RESET);
```

```
    }
    public Apiary SearchApiaryBySerialNb(int SerialNb){
        for(int i=0;i<apiary.size();i++){
            if(apiary.get(i).getApiarySerialNB()==SerialNb){
                return apiary.get(i);
            }
        }
        return null;
    }
    public void ListApiary(){
        for(int i=0;i<apiary.size();i++){
            apiary.get(i).ApiaryStatus();
            apiary.get(i).ListHivesInApiary();
            System.out.println(ANSI_YELLOW
+"----------------------------------------" + ANSI_RESET);
        }
    }
    public void ListAllCustomers(){
        s.ListCustomers();
        s.ListBannedCustomers();
    }

    public Stock SearchStockByDate(Date d){
        for(int i=0;i<stock.size();i++){
            if(stock.get(i).getDateLogged()==d){
                return stock.get(i);
            }
        }
        return null;
```

```
    }
    public void ListAllStock(){
        for(int i=0;i<stock.size();i++){
            Date d=stock.get(i).getDateLogged();
            stock.get(i).DisplayStock(sales.get(d).getFoodBought(),
sales.get(d).getDrugsBought());
            System.out.println(ANSI_YELLOW
+"----------------------------------------" + ANSI_RESET);
        }
    }
    public Sales SearchSalesByDate(Date d){
            return sales.get(d);
    }
    public void ListAllSales(){
        for (Sales s : sales.values()) {
            s.ListSales();
            System.out.println(ANSI_YELLOW +
"----------------------------------------" + ANSI_RESET);
        }
    }
    public Apiary FindApiaryBYSerialNBfromHive(int l) {
        for (int i = 0; i < apiary.size();i++)
           {
            if(!apiary.get(i).isEmpty()){
                if(apiary.get(i).FindHiveBYSerialNBfromApiary(l)!=null){
                    if(
apiary.get(i).FindHiveBYSerialNBfromApiary(l).getHiveSerialNb() == l ) {
                        return apiary.get(i);
                    }
                }
```

```
            }


        }
        return null;
    }
    public static final String ANSI_RESET = "\u001B[0m";
    public static final String ANSI_BLACK = "\u001B[30m";
    public static final String ANSI_RED = "\u001B[31m";
    public static final String ANSI_GREEN = "\u001B[32m";
    public static final String ANSI_YELLOW = "\u001B[33m";
    public static final String ANSI_BLUE = "\u001B[34m";
    public static final String ANSI_PURPLE = "\u001B[35m";
    public static final String ANSI_CYAN = "\u001B[36m";
    public static final String ANSI_WHITE = "\u001B[37m";

}
```

Users

This class is used in order to allow multiple

Beekeepers to Register and Login.

It utilizes a HashSet

It's Constructor Takes in two *Strings:*

- Email
- Password

And A beekeeper object.

This class has many *methods:*

- Register(String,String ,BeeKeeper):void Registers a User "Binds a beekeeper with user"
- Login(String ,String):users Allows the user to login "Searches for a specific user and Returns it"
- SettersAndGetters

The Users Code:

```
import java.io.Serializable;

import java.util.HashSet;


public class Users implements Serializable{

    HashSet<Users> users = new HashSet<Users>();

    private String email;

    private String password;

    private BeeKeeper BeeK;

    public Users(){

        email="";
```

```
        password="";
        BeeK=null;
    }
    public Users(String email,String password,BeeKeeper BeeK){
        this.email=email;
        this.password=password;
        this.BeeK=BeeK;
    }
    public String getEmail() {
        return email;
    }
    public void setPassowrd(String password) {
        this.password = password;
    }
    public String getPassword() {
        return password;
    }
    public void setBeeKeeper(BeeKeeper BeeK) {
        this.BeeK = BeeK;
    }
    public BeeKeeper getBeeKeeper() {
        return BeeK;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public boolean isEmpty(){
        return users.size()==0;
    }
```

```java
    public void Register(String email,String pass,BeeKeeper B){
        Users[] allUsers = users.toArray(new Users[users.size()]);
        for(int i=0; i< users.size();i++){
            if(allUsers[i].getEmail().equals(email)){
                System.out.println("Email Already Exists!");
                return;
            }
        }
            Users user = new Users(email,pass,B);
            users.add(user);


    }
    public Users Login(String email,String password){
        Users[] allUsers = users.toArray(new Users[users.size()]);
        int PasswordCount=0;
        int EmailCount=0;
        for(int i=0;i<users.size();i++){
            if(allUsers[i].getEmail().equals(email)){
                EmailCount++;
            }
            if(allUsers[i].getPassword().equals(password)){
                PasswordCount++;
            }
            if(allUsers[i].getEmail().equals(email)  &&
allUsers[i].getPassword().equals(password)){
                return allUsers[i];
            }
        }
        if(EmailCount==0 && PasswordCount==0){
```

```
            System.out.println(ANSI_RED + "Wrong Password and Email!" +
ANSI_RESET);
        }else if(PasswordCount==0){
            System.out.println(ANSI_RED + "Wrong Password!" + ANSI_RESET);
        }else{
            System.out.println(ANSI_RED + "Wrong Email!" + ANSI_RESET);
        }
        return null;
    }
    public static final String ANSI_RESET = "\u001B[0m";
    public static final String ANSI_BLACK = "\u001B[30m";
    public static final String ANSI_RED = "\u001B[31m";
    public static final String ANSI_GREEN = "\u001B[32m";
    public static final String ANSI_YELLOW = "\u001B[33m";
    public static final String ANSI_BLUE = "\u001B[34m";
    public static final String ANSI_PURPLE = "\u001B[35m";
    public static final String ANSI_CYAN = "\u001B[36m";
    public static final String ANSI_WHITE = "\u001B[37m";
}
```

Main

We tried to make the main class as dynamic as possible and implemented many error handling measures . We were trying to substitute the need for a UI while still allowing non-developers to use our code as our main class is very User-Friendly

The Main Code:

```
import java.util.Scanner;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;


public class Main {


public static void main(String[] args) throws IOException,
ClassNotFoundException, ParseException{

      Scanner console=new Scanner(System.in);


      File file = new File("C:\\Users\\user\\Desktop\\Users.txt");

      if(file.length()==0){

            Users AllUsers = new Users();
```

```
            Registration(AllUsers,console);
            writeObjectToFile(AllUsers, file);
        }
            Users AllUsers = readObjectFromFile(file);
            Users SignedIn = new Users();
            System.out.println(ANSI_YELLOW +"1) Login: \n2) Register: " +
ANSI_RESET);
            System.out.println(ANSI_CYAN + "Choose One Of The Options: " +
ANSI_RESET);
            int LoginOrSignup=console.nextInt();
            if(LoginOrSignup==1){
                System.out.print(ANSI_CYAN + "Enter an email: " +
ANSI_RESET);
                String email = console.next();
                System.out.print(ANSI_CYAN + "\nEnter Password: " +
ANSI_RESET);
                String password = console.next();
                SignedIn = AllUsers.Login(email, password);
            }else{
                SignedIn=Registration(AllUsers,console);
                writeObjectToFile(AllUsers, file);
            }
            System.out.println(ANSI_YELLOW + "Welcome Back " + ANSI_GREEN
+ SignedIn.getBeeKeeper().getFName() + " " +
SignedIn.getBeeKeeper().getLName() + ANSI_RESET);
            boolean bool=true;
            while(bool){
                System.out.println(ANSI_YELLOW + "1) Bee Management\n2)
Customer Management\n3) Sales Management\n4) Stock Management\n5) Exit" +
ANSI_RESET);
                int Management = console.nextInt();
```

```
                switch(Management){
                    //Bee Management
                    case 1:
                        System.out.println(ANSI_YELLOW +"1) Add\n2)
Edit\n3) View" + ANSI_RESET);
                        int Option = console.nextInt();
                        switch(Option){
                            //Add Everything into apiary
                            case 1:
                                System.out.println(ANSI_YELLOW + "APIARY:"
+ ANSI_RESET);
                                System.out.println(ANSI_YELLOW +
"--------"  + ANSI_RESET);
                                System.out.println(ANSI_CYAN + "Apiary
Name:");
                                String ApiaryName=console.next();
                                System.out.println(ANSI_CYAN + "Apiary
SerialNb:");
                                int ApiarySerialNB=console.nextInt();
                                Apiary CheckApiary =
SignedIn.getBeeKeeper().SearchApiaryBySerialNb(ApiarySerialNB);
                                Apiary newApiary;
                                String ApiaryLocation;
                                if(CheckApiary==null){
                                    System.out.println(ANSI_CYAN + "Apiary
Location:");
                                    ApiaryLocation=console.next();
                                    newApiary = new Apiary(ApiaryName,
ApiarySerialNB, ApiaryLocation);
                                }else{
```

```
                                    System.out.println(ANSI_RED + "Apiary
" + ANSI_GREEN + ApiarySerialNB + ANSI_RED + " Already Exists!" +
ANSI_RESET);
                                    break;
                                }
                                System.out.println(ANSI_YELLOW +
"-----------------------------------------"  + ANSI_RESET);
                                System.out.println(ANSI_YELLOW + "HIVE:"
+ ANSI_RESET);
                                System.out.println(ANSI_YELLOW +
"--------"  + ANSI_RESET);
                                System.out.println(ANSI_CYAN + "Hive
Size:" + ANSI_RESET);
                                int HiveSize= console.nextInt();
                                System.out.println(ANSI_CYAN + "Hive
Serial Number:"  + ANSI_RESET);
                                int HiveSerialNb=console.nextInt();
                                int HiveMedicalCondition;
                                int HiveNbOfFrames;
                                char HiveFeeding;
                                boolean HiveFed;
                                char HiveDrugged;
                                boolean HiveDrug;
                                Hive newHive;
                                Apiary CheckApiaryThatContainsHiveSerialNb
= SignedIn.getBeeKeeper().FindApiaryBYSerialNBfromHive(HiveSerialNb);

if(CheckApiaryThatContainsHiveSerialNb==null){
                                        System.out.println(ANSI_CYAN +
"Medical Condition(1-10):"  + ANSI_RESET);

HiveMedicalCondition=console.nextInt();
```

```
                                        System.out.println(ANSI_CYAN +
"Number Of Frames:"  + ANSI_RESET);
                                        HiveNbOfFrames=console.nextInt();
                                        System.out.println(ANSI_CYAN +
"Are They Fed?(Y,N)"  + ANSI_RESET);

HiveFeeding=console.next().charAt(0);
                                        HiveFed=true;
                                        if(HiveFeeding=='Y'){
                                            HiveFed=true;
                                        }else{
                                            HiveFed=false;
                                        }
                                        System.out.println(ANSI_CYAN +
"Are They Drugged?(Y,N)"  + ANSI_RESET);

HiveDrugged=console.next().charAt(0);
                                        HiveDrug=true;
                                        if(HiveDrugged=='Y'){
                                            HiveDrug=true;
                                        }else{
                                            HiveDrug=false;
                                        }
                                        newHive = new Hive(HiveSize,
HiveSerialNb, HiveMedicalCondition, HiveNbOfFrames, HiveFed, HiveDrug);
                                        System.out.println(ANSI_YELLOW +
"-----------------------------------------"  + ANSI_RESET);
                                    }else{
                                        System.out.println(ANSI_RED +
"Hive " + ANSI_GREEN + HiveSerialNb + ANSI_RED + " Already Exists!" +
ANSI_RESET);
```

```
                                        break;
                                }
                                System.out.println(ANSI_YELLOW + "BEES:"
+ ANSI_RESET);
                                System.out.println(ANSI_YELLOW +
"--------"  + ANSI_RESET);
                                System.out.println(ANSI_CYAN + "Type Of
Bee 'Species' :"  + ANSI_RESET);
                                String BeeType=console.next();
                                System.out.println(ANSI_CYAN + "Estimated
Number Of Bees:"  + ANSI_RESET);
                                int EstimatesNbOfBees=console.nextInt();
                                System.out.println(ANSI_CYAN + "Life
Span(days) :" + ANSI_RESET);
                                int LifeSpan=console.nextInt();
                                System.out.println(ANSI_CYAN + "Efficiency
Of Bees:"  + ANSI_RESET);
                                int EfficiencyOfBee=console.nextInt();
                                GeneralBees newBee = new Bee(BeeType,
EfficiencyOfBee, LifeSpan, EstimatesNbOfBees);
                                System.out.println(ANSI_YELLOW +
"----------------------------------------"  + ANSI_RESET);
                                System.out.println(ANSI_YELLOW +
"QUEENBEE:"  + ANSI_RESET);
                                System.out.println(ANSI_YELLOW +
"--------"  + ANSI_RESET);
                                boolean QueenBeeType=true;
                                System.out.println(ANSI_CYAN + "Egg Laying
Rate:"  + ANSI_RESET);
                                int
QueenBeeEggLayingRate=console.nextInt();
```

```
                                System.out.println(ANSI_CYAN + "Life
Span(years) :"  + ANSI_RESET);
                                int QueenBeeLifeSpan=console.nextInt();
                                System.out.println(ANSI_CYAN + "Medical
Health 'Description' "  + ANSI_RESET);
                                String
QueenBeeMedicalHealth=console.next();
                                GeneralBees newQueenBee = new
QueenBee(QueenBeeType, QueenBeeEggLayingRate, QueenBeeLifeSpan,
QueenBeeMedicalHealth);
                                newHive.addBee(newBee);
                                newHive.addQueenBee(newQueenBee);
                                newApiary.addHiveToApiary(newHive);

SignedIn.getBeeKeeper().addApiaryToBeekeeperUser(newApiary);
                                break;
                            //Edit or Remove(Apiary/Hive/QueenBee)
                            case 2:
                                System.out.println(ANSI_YELLOW + "1)
Edit\n2) Remove"  + ANSI_RESET);
                                int EditOrRemove = console.nextInt();
                                switch(EditOrRemove){
                                    //Edit
                                    case 1:
                                        System.out.println(ANSI_YELLOW +
"1) Edit Apiary\n2) Edit Hive\n3) Edit QueenBee"  + ANSI_RESET);
                                        int EditOption =
console.nextInt();
                                        switch(EditOption){
                                            //Edit Apiary
                                            case 1:
```

```
System.out.println(ANSI_CYAN + "Enter The Apiary Serial Nb:"  +
ANSI_RESET);
                                                int
EditApiarySerialNb=console.nextInt();
                                                CheckApiary =
SignedIn.getBeeKeeper().SearchApiaryBySerialNb(EditApiarySerialNb);
                                                if(CheckApiary!=null){

newApiary=SignedIn.getBeeKeeper().SearchApiaryBySerialNb(EditApiarySerialN
b);

System.out.println(ANSI_YELLOW + "APIARY:" + ANSI_GREEN + " " +
newApiary.getApiarySerialNB() + " " + ANSI_RESET);

System.out.println(ANSI_YELLOW + "--------" + ANSI_RESET);

System.out.println(ANSI_CYAN + "Apiary Name:" + ANSI_RESET);

ApiaryName=console.next();

System.out.println(ANSI_CYAN + "Apiary SerialNb:" + ANSI_RESET);

ApiarySerialNB=console.nextInt();

System.out.println(ANSI_CYAN + "Apiary Location:" + ANSI_RESET);

ApiaryLocation=console.next();

newApiary.setApiaryName(ApiaryName);

newApiary.setApiarySerialNB(ApiarySerialNB);

newApiary.setLocation(ApiaryLocation);
```

```
                                                    break;
                                                }else{

System.out.println(ANSI_RED + "Apiary " + ANSI_GREEN + EditApiarySerialNb
+ ANSI_RED + " Doesn't Exist!" + ANSI_RESET);
                                                    break;
                                                }
                                            //Edit Hive
                                            case 2:

System.out.println(ANSI_CYAN + "Enter The Hive Serial Nb:" + ANSI_RESET);
                                                int
EditHiveSerialNb=console.nextInt();

CheckApiaryThatContainsHiveSerialNb =
SignedIn.getBeeKeeper().FindApiaryBYSerialNBfromHive(EditHiveSerialNb);
                                                Hive EditHive;

if(CheckApiaryThatContainsHiveSerialNb!=null){

EditHive=CheckApiaryThatContainsHiveSerialNb.FindHiveBYSerialNBfromApiary(
EditHiveSerialNb);

System.out.println(ANSI_YELLOW + "HIVE:" + ANSI_GREEN + " " +
EditHive.getHiveSerialNb() + " " + ANSI_RESET );

System.out.println(ANSI_YELLOW + "--------" + ANSI_RESET);

System.out.println(ANSI_CYAN + "Hive Size:" + ANSI_RESET);
                                                    HiveSize=
console.nextInt();

System.out.println(ANSI_CYAN + "Hive Serial Number:" + ANSI_RESET);
```

```
HiveSerialNb=console.nextInt();

System.out.println(ANSI_CYAN + "Medical Condition(1-10):" + ANSI_RESET);

HiveMedicalCondition=console.nextInt();

System.out.println(ANSI_CYAN + "Number Of Frames:" + ANSI_RESET);

HiveNbOfFrames=console.nextInt();

System.out.println(ANSI_CYAN + "Are They Fed?(Y,N)" + ANSI_RESET);

HiveFeeding=console.next().charAt(0);
                                                    HiveFed=true;
                                                    if(HiveFeeding=='Y'){
                                                        HiveFed=true;
                                                    }else{
                                                        HiveFed=false;
                                                    }

System.out.println(ANSI_CYAN + "Are They Drugged?(Y,N)" + ANSI_RESET);

HiveDrugged=console.next().charAt(0);
                                                    HiveDrug=true;
                                                    if(HiveDrugged=='Y'){
                                                        HiveDrug=true;
                                                    }else{
                                                        HiveDrug=false;
                                                    }

EditHive.setSize(HiveSize);
```

```
EditHive.setHiveSerialNb(HiveSerialNb);

EditHive.setMedicalCondition(HiveMedicalCondition);

EditHive.setNbOfFrames(HiveNbOfFrames);

EditHive.setFood(HiveFed);

EditHive.setDrugs(HiveDrug);
                                                        break;
                                            }else{

System.out.println(ANSI_RED + "Hive " + ANSI_GREEN + EditHiveSerialNb +
ANSI_RED + " Doesn't Exist!" + ANSI_RESET);
                                                    break;
                                            }
                                            //Edit QueenBee
                                            case 3:
                                            System.out.println(ANSI_CYAN +
"Enter The Hive Serial Nb:" + ANSI_RESET);

EditHiveSerialNb=console.nextInt();

CheckApiaryThatContainsHiveSerialNb =
SignedIn.getBeeKeeper().FindApiaryBYSerialNBfromHive(EditHiveSerialNb);

if(CheckApiaryThatContainsHiveSerialNb!=null){

EditHive=CheckApiaryThatContainsHiveSerialNb.FindHiveBYSerialNBfromApiary(
EditHiveSerialNb);

System.out.println(ANSI_YELLOW + "QUEENBEE:" + ANSI_RESET);
```

```
System.out.println(ANSI_YELLOW + "--------" + ANSI_RESET);
                                                QueenBeeType=true;

System.out.println(ANSI_CYAN + "Egg Laying Rate:" + ANSI_RESET);

QueenBeeEggLayingRate=console.nextInt();

System.out.println(ANSI_CYAN + "Life Span:(years)" + ANSI_RESET);

QueenBeeLifeSpan=console.nextInt();

System.out.println(ANSI_CYAN + "Medical Health" + ANSI_RESET);

QueenBeeMedicalHealth=console.next();
                                                newQueenBee = new
QueenBee(QueenBeeType, QueenBeeEggLayingRate, QueenBeeLifeSpan,
QueenBeeMedicalHealth);

EditHive.ChangeQueenBee(newQueenBee);
                                                break;
                                            }else{

System.out.println(ANSI_RED + "Hive " + ANSI_GREEN + EditHiveSerialNb +
ANSI_RED + " Doesn't Exist!" + ANSI_RESET);
                                                break;
                                            }
                                            default:

System.out.println(ANSI_RED + "Wrong Edit Option!" + ANSI_RESET);
                                                break;
                                        }
```

```
                                        break;
                                    //Remove
                                    case 2:
                                    System.out.println(ANSI_YELLOW + "1)
Remove Apiary\n2) Remove Hive" + ANSI_RESET);
                                    EditOption = console.nextInt();
                                    switch(EditOption){
                                        //Remove Apiary
                                        case 1:
                                            System.out.println(ANSI_CYAN +
"Enter The Apiary Serial Nb:" + ANSI_RESET);
                                            int
EditApiarySerialNb=console.nextInt();

newApiary=SignedIn.getBeeKeeper().SearchApiaryBySerialNb(EditApiarySerialN
b);
                                            if(newApiary!=null){

SignedIn.getBeeKeeper().apiary.remove(newApiary);

System.out.println(ANSI_YELLOW + "Successfully Removed Apiary!" +
ANSI_RESET);
                                            }else{

System.out.println(ANSI_RED + "Apiary Not Found!" + ANSI_RESET);
                                            }
                                            break;
                                        //Remove Hive
                                        case 2:
                                            System.out.println(ANSI_CYAN +
"Enter The Apiary Serial Nb:" + ANSI_RESET);
```

```
EditApiarySerialNb=console.nextInt();

newApiary=SignedIn.getBeeKeeper().SearchApiaryBySerialNb(EditApiarySerialN
b);
                                                if(newApiary!=null){

System.out.println(ANSI_CYAN + "Enter The Hive Serial Nb:" + ANSI_RESET);
                                                    int
EditHiveSerialNb=console.nextInt();
                                                    Hive
EditHive=newApiary.FindHiveBYSerialNBfromApiary(EditHiveSerialNb);
                                                    if(EditHive!=null){

newApiary.removeHivefromApiary(EditHive);

System.out.println(ANSI_YELLOW + "Successfully Removed Hive!" +
ANSI_RESET);
                                                    }else{

System.out.println(ANSI_RED + "Hive Not Found!" + ANSI_RESET);
                                                    }
                                                }else{

System.out.println(ANSI_RED + "Apiary Not Found!" + ANSI_RESET);
                                                }
                                                break;
                                            default:
                                                System.out.println(ANSI_RED +
"Wrong Edit Option!" + ANSI_RESET);
                                                break;
                                        }
```

```
                                        break;


                                    default:
                                        System.out.println(ANSI_RED +
"Wrong Edit Or Remove Option!" + ANSI_RESET);
                                        break;
                                }
                                break;
                            //View Apiary
                            case 3:
                                SignedIn.getBeeKeeper().ListApiary();
                                break;
                            default:
                                System.out.println(ANSI_RED + "Invalid
Add/Edit/Remove Option!" + ANSI_RESET);
                                break;
                        }
                        break;
                    //Customer Management
                    case 2:
                        System.out.println(ANSI_YELLOW + "1) Add\n2)
Ban\n3) View" + ANSI_RESET);
                        Option = console.nextInt();
                        switch(Option){
                            //Add Customers
                            case 1:
                                System.out.println(ANSI_CYAN + "How many
Customers Do You Want To Add?" + ANSI_RESET);
                                int NumberOfCustomers= console.nextInt();
                                int counter=0;
```

```
                                while(counter!=NumberOfCustomers){
                                    System.out.println(ANSI_CYAN + "Enter
Customer " + counter + " First Name: " + ANSI_RESET);
                                    String CustomerFName=console.next();
                                    System.out.println(ANSI_CYAN + "Enter
Customer " + counter + "  Last Name: " + ANSI_RESET);
                                    String CustomerLName=console.next();
                                    System.out.println(ANSI_CYAN + "Enter
Customer " + counter + " email: " + ANSI_RESET);
                                    String CustomerEmail=console.next();
                                    System.out.println(ANSI_CYAN + "Enter
Customer " + counter + " Address: " + ANSI_RESET);
                                    String CustomerAddress=console.next();
                                    System.out.println(ANSI_CYAN + "Enter
Customer " + counter + " PhoneNumber: " + ANSI_RESET);
                                    int CustomerPhoneNb=console.nextInt();
                                    Customers newCustomer = new
Customers(CustomerFName, CustomerLName, CustomerEmail, CustomerAddress,
CustomerPhoneNb);

SignedIn.getBeeKeeper().s.addCustomer(newCustomer, CustomerEmail);
                                    counter++;
                                }
                                break;
                            //Ban Customer
                            case 2:
                                System.out.println(ANSI_CYAN + "Enter
Customer's Email You Wish To Ban:" + ANSI_RESET);
                                String BanningEmail=console.next();
                                Customers
BannedCustomer=SignedIn.getBeeKeeper().s.SearchCustomerbyEmail(BanningEmai
l);
```

```
                                if(BannedCustomer!=null){

SignedIn.getBeeKeeper().s.banCustomer(BannedCustomer, BanningEmail);
                                }else{
                                    System.out.println(ANSI_RED +
"Customer With This Email Doesn't Exist!" + ANSI_RESET);
                                }
                                break;
                            //View Customers
                            case 3:

SignedIn.getBeeKeeper().ListAllCustomers();
                                break;
                            default:
                                System.out.println(ANSI_RED + "Invalid
Add/Ban/View Option!" + ANSI_RESET);
                                break;
                        }
                            break;
                    //Sales Management
                    case 3:
                        System.out.println(ANSI_YELLOW + "1) Add(Sales
then Stock)\n2) Edit\n3) View" + ANSI_RESET);
                        Option = console.nextInt();
                        switch(Option){
                            //Add Sales And Stock
                            case 1:
                                System.out.println(ANSI_YELLOW + "SALES: "
+ ANSI_RESET);
                                System.out.println(ANSI_CYAN + "Enter
Date(dd/MM/yyyy): " + ANSI_RESET);
```

```
                                String Date = console.next();
                                Date date1=new
SimpleDateFormat("dd/MM/yyyy").parse(Date);
                                Sales CheckSalesByDate =
SignedIn.getBeeKeeper().SearchSalesByDate(date1);
                                String Season;
                                int Revenue;
                                int HivesBought;
                                int JarsBought;
                                int FoodBought;
                                int DrugsBought;
                                int OperatingExpenses;
                                int Other;
                                int TotalNbOfJars;
                                int JarsFilledWithHoney;
                                int FoodUsed;
                                int DrugsUsed;
                                if(CheckSalesByDate==null){
                                    System.out.println(ANSI_CYAN + "Enter
Season (Spring,Summer) :" + ANSI_RESET);
                                    Season=console.next();
                                    System.out.println(ANSI_CYAN + "Enter
Revenue: " + ANSI_RESET);
                                    Revenue = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Hives Bought: " + ANSI_RESET);
                                    HivesBought =console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Jars Bought: " + ANSI_RESET);
                                    JarsBought =console.nextInt();
```

```
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Food Bought: " + ANSI_RESET);
                                    FoodBought = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Drugs Bought: " + ANSI_RESET);
                                    DrugsBought = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Operating Expenses: " + ANSI_RESET);
                                    OperatingExpenses = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Other Expenses: " + ANSI_RESET);
                                    Other= console.nextInt();
                                    System.out.println(ANSI_YELLOW +
"----------------------" + ANSI_RESET);
                                    System.out.println(ANSI_YELLOW +
"STOCK: " + ANSI_RESET);
                                    System.out.println(ANSI_CYAN + "Enter
Total Number Of Jars: " + ANSI_RESET);
                                    TotalNbOfJars = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Jars
Filled With Honey: " + ANSI_RESET);
                                    JarsFilledWithHoney=
console.nextInt();
                                    if(JarsFilledWithHoney>TotalNbOfJars){
                                        System.out.println(ANSI_RED +
"Jars Filled With Honey Cant Be More Than The Total Number Of Jars!!" +
ANSI_RESET);
                                    }else{
                                        System.out.println(ANSI_CYAN +
"Food Used: " + ANSI_RESET);
                                        FoodUsed = console.nextInt();
```

```
                                        System.out.println(ANSI_CYAN +
"Drugs Used: " + ANSI_RESET);
                                        DrugsUsed = console.nextInt();
                                        Sales newSales = new Sales(date1,
Season, Revenue, HivesBought, JarsBought, FoodBought, DrugsBought,
OperatingExpenses, Other);
                                        Stock newStock = new Stock(date1,
TotalNbOfJars, JarsFilledWithHoney, FoodUsed, DrugsUsed);

SignedIn.getBeeKeeper().addSalesToBeekeeperUser(date1, newSales);

SignedIn.getBeeKeeper().addStockToBeekeeperUser(newStock);
                                    }
                                }else{
                                    System.out.println(ANSI_RED + "Sales
And Stock With This Date " + ANSI_GREEN + date1 + ANSI_RED +" Already
Exist!" + ANSI_RESET);
                                }
                                break;
                            //Edit Sales
                            case 2:
                                System.out.println(ANSI_CYAN + "Enter
Date(dd/MM/yyyy): " + ANSI_RESET);
                                Date = console.next();
                                date1=new
SimpleDateFormat("dd/MM/yyyy").parse(Date);
                                Sales
EditSales=SignedIn.getBeeKeeper().SearchSalesByDate(date1);
                                if(EditSales!=null){
                                    System.out.println(ANSI_CYAN + "Enter
Season (Spring,Summer) :" + ANSI_RESET);
                                    Season=console.next();
```

```
                                    System.out.println(ANSI_CYAN + "Enter
Revenue: " + ANSI_RESET);
                                    Revenue = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Hives Bought: " + ANSI_RESET);
                                    HivesBought =console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Jars Bought: " + ANSI_RESET);
                                    JarsBought =console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Food Bought: " + ANSI_RESET);
                                    FoodBought = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Amount Of Drugs Bought: " + ANSI_RESET);
                                    DrugsBought = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Operating Expenses: " + ANSI_RESET);
                                    OperatingExpenses = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Enter
Other Expenses: " + ANSI_RESET);
                                    Other= console.nextInt();
                                    EditSales.setSeason(Season);
                                    EditSales.setRevenue(Revenue);
                                    EditSales.setHivesBought(HivesBought);
                                    EditSales.setJarsBought(JarsBought);
                                    EditSales.setFoodBought(FoodBought);
                                    EditSales.setDrugsBought(DrugsBought);

EditSales.setOperatingExpenses(OperatingExpenses);
                                    EditSales.setOther(Other);
```

```
                                    System.out.println(ANSI_YELLOW +
"Successfuly Updated!" + ANSI_RESET);
                                }else{
                                    System.out.println(ANSI_RED + "Sales
With This Date " + ANSI_GREEN + date1 + ANSI_RED +" Doesnt't Exist!" +
ANSI_RESET);
                                }
                                break;
                            //View Sales
                            case 3:
                                SignedIn.getBeeKeeper().ListAllSales();
                                break;
                            default:
                                System.out.println(ANSI_RED + "Invalid
Add/Edit/View Option!" + ANSI_RESET);
                                break;
                        }
                            break;
                    //Stock Management
                    case 4:
                    System.out.println(ANSI_YELLOW + "1) Edit\n2) View" +
ANSI_RESET);
                    Option = console.nextInt();
                    switch(Option){
                        //Edit Stock
                        case 1:
                            System.out.println(ANSI_CYAN + "Enter
Date(dd/MM/yyyy): " + ANSI_RESET);
                            String Date = console.next();
                            Date date1=new
SimpleDateFormat("dd/MM/yyyy").parse(Date);
```

```
                            Stock
EditStock=SignedIn.getBeeKeeper().SearchStockByDate(date1);
                            if(EditStock!=null){
                                System.out.println(ANSI_CYAN + "Enter
Total Number Of Jars: " + ANSI_RESET);
                                int TotalNbOfJars = console.nextInt();
                                System.out.println(ANSI_CYAN + "Jars
Filled With Honey: " + ANSI_RESET);
                                int JarsFilledWithHoney=
console.nextInt();
                                if(JarsFilledWithHoney>TotalNbOfJars){
                                    System.out.println("Jars Filled With
Honey Cant Be More Than The Total Number Of Jars!!" + ANSI_RESET);
                                }else{
                                    System.out.println(ANSI_CYAN + "Food
Used: " + ANSI_RESET);
                                    int FoodUsed = console.nextInt();
                                    System.out.println(ANSI_CYAN + "Drugs
Used: " + ANSI_RESET);
                                    int DrugsUsed = console.nextInt();

EditStock.setTotalNbOfJars(TotalNbOfJars);

EditStock.setJarsFilledWithHoney(JarsFilledWithHoney);
                                    EditStock.setFoodUsed(FoodUsed);
                                    EditStock.setDrugsUsed(DrugsUsed);
                                    System.out.println(ANSI_YELLOW +
"Successfully Updated!" + ANSI_RESET);
                                }
                            }else{
```

```
                                System.out.println(ANSI_RED + "Stock With
This Date " + ANSI_GREEN + date1 + ANSI_RED +" Doesn't Exist!" +
ANSI_RESET);
                            }
                            break;
                        //View Stock
                        case 2:
                            SignedIn.getBeeKeeper().ListAllStock();
                            break;
                        default:
                            System.out.println(ANSI_RED + "Invalid
Add/Edit/View Option!" + ANSI_RESET);
                            break;
                        }
                        break;
                    //Exit
                    case 5:
                        bool=false;
                        break;
                    default:
                        System.out.println(ANSI_RED + "Please Input a
Valid Number!" + ANSI_RESET);
                        break;
                    }


            }
            //Save data into the file
            writeObjectToFile(AllUsers, file);
  }
```

```
  // Serialization
  // Save object into a file.
  public static void writeObjectToFile(Users obj, File file) throws
IOException {
      try (FileOutputStream fos = new FileOutputStream(file);
           ObjectOutputStream oos = new ObjectOutputStream(fos)) {
          oos.writeObject(obj);
          oos.flush();
      }
  }

  // Deserialization
  // Get object from a file.
  public static Users readObjectFromFile(File file) throws IOException,
ClassNotFoundException {
      Users result = null;
      try (FileInputStream fis = new FileInputStream(file);
           ObjectInputStream ois = new ObjectInputStream(fis)) {
          result = (Users) ois.readObject();
      }
      return result;
  }

  //Registration
  public static Users Registration(Users U,Scanner console){
        System.out.println(ANSI_YELLOW + "REGISTRATION:" + ANSI_RESET);
        System.out.println(ANSI_YELLOW + "--------------------" +
ANSI_RESET);
```

```java
        System.out.println(ANSI_CYAN + "First Name: " + ANSI_RESET);
        String FName = console.next();
        System.out.println(ANSI_CYAN + "Last Name: " + ANSI_RESET);
        String LName = console.next();
        System.out.println(ANSI_CYAN + "Phone Number: " + ANSI_RESET);
        int PhoneNumber = console.nextInt();
        System.out.println(ANSI_CYAN + "Address: " + ANSI_RESET);
        String Address = console.next();
        System.out.print(ANSI_CYAN + "Enter an email: " + ANSI_RESET);
        String email = console.next();
        System.out.print(ANSI_CYAN + "\nEnter Password: " + ANSI_RESET);
        String password = console.next();
        BeeKeeper B = new BeeKeeper(FName,LName,PhoneNumber,Address);
        Customers allCustomers=new Customers();
        B.addCustomersToBeekeeperUserNoPrint(allCustomers);
        U.setBeeKeeper(B);
        U.Register(email, password, B);
        Users SignedIn = U.Login(email, password);
        return SignedIn;
    }

    //Colors
    public static final String ANSI_RESET = "\u001B[0m";
    public static final String ANSI_BLACK = "\u001B[30m";
    public static final String ANSI_RED = "\u001B[31m";
    public static final String ANSI_GREEN = "\u001B[32m";
    public static final String ANSI_YELLOW = "\u001B[33m";
    public static final String ANSI_BLUE = "\u001B[34m";
    public static final String ANSI_PURPLE = "\u001B[35m";
```

```
  public static final String ANSI_CYAN = "\u001B[36m";

  public static final String ANSI_WHITE = "\u001B[37m";



}
```

The Result can Vary from User to User as there are a lot of options and methods the user can choose from, that would be really too long to dive in for this Proposal as you can judge from the size of the main class . In our presentation we will be showing you all these options as we run our code live “yes that's how confident we are” .

Thank you for reading and listening to us.

And again for anyone interested to have a look at the code and run it.

Press this link: https://github.com/Hadious15/BMS.git

Alternatively Please scan the QrCode: