Cairo University

Faculty of Computers and Artificial Intelligence

Software Engineering

Sent to: Dr. Mohamed El-Ramly

CS251

- **Assignment: A1**
- **Task: T1 &T2**
- **Date: 2025/2/25**
- **Section: S14**
- **Team Programming Language: Java**
- **Team Leader Phone Number: +20 128 696 4627**
- **Name, IDs, and E-mails:**

Name	IDs	E-Mails.
Aly El-Deen Yasser Ali	20231109	ali.el.badry.747@gmail.com
Nagham Wael Mohamed	20231189	naghamw63@gmail.com
Fatema El-Zhraa Ahmed Mohamed El-Fiky	20230280	fatmaelfeky922@gmail.com

---

Table of Content

The Process of Learning :

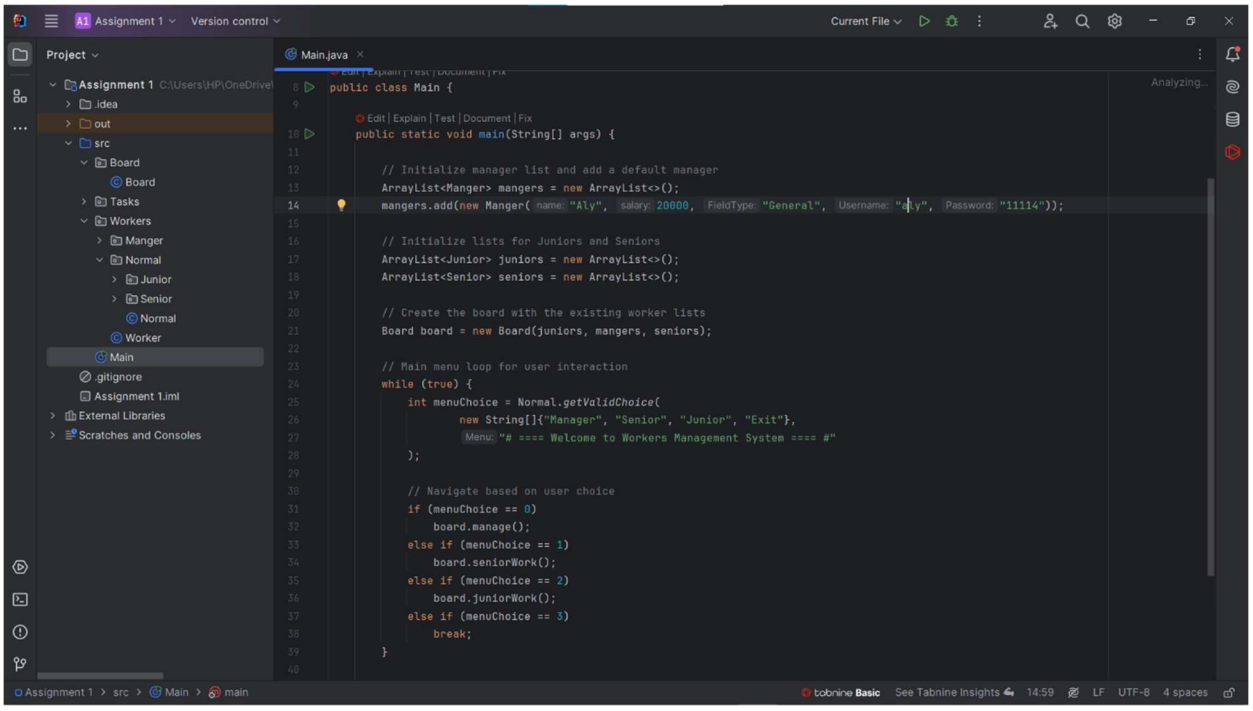
Aly El-Deen Yasser Aly:

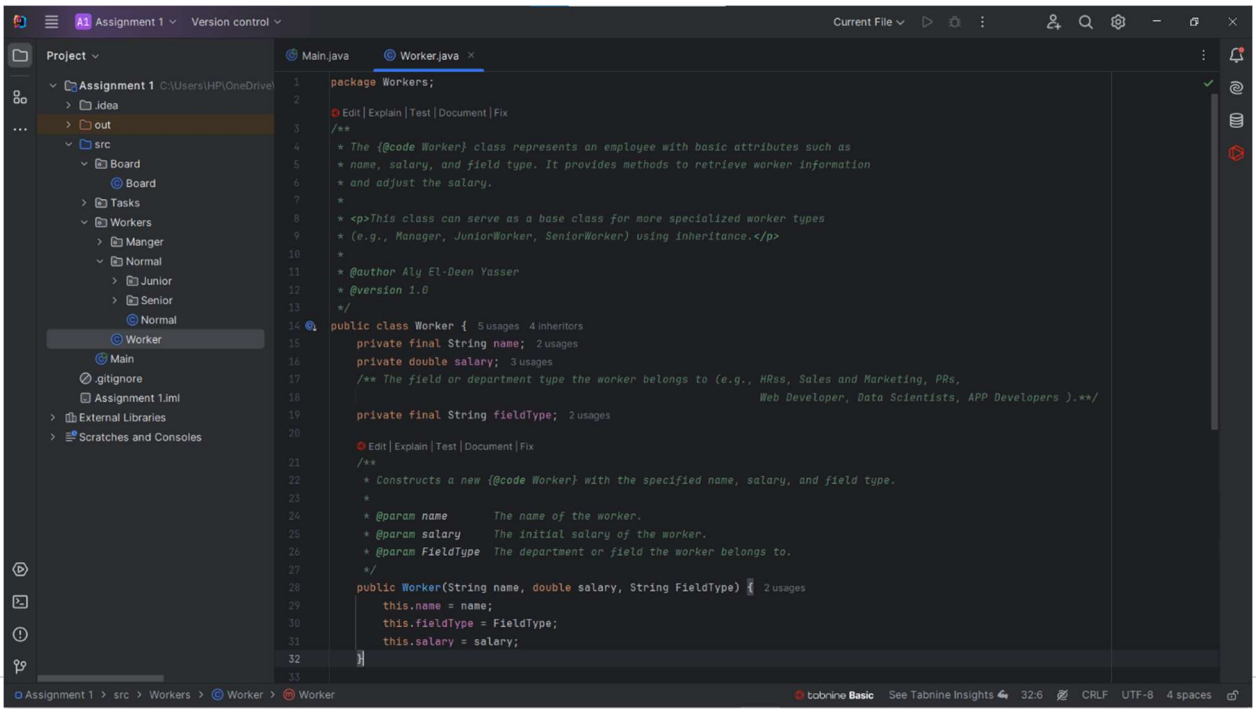
- Hours of Study: 18 hours
- Source of study https://youtube.com/playlist?list=PLCInYL3l2AajYlZGzU_LVrHdoouf8W6ZN&si=tfRlF2iNxOtJehum
- Main Logic of code:

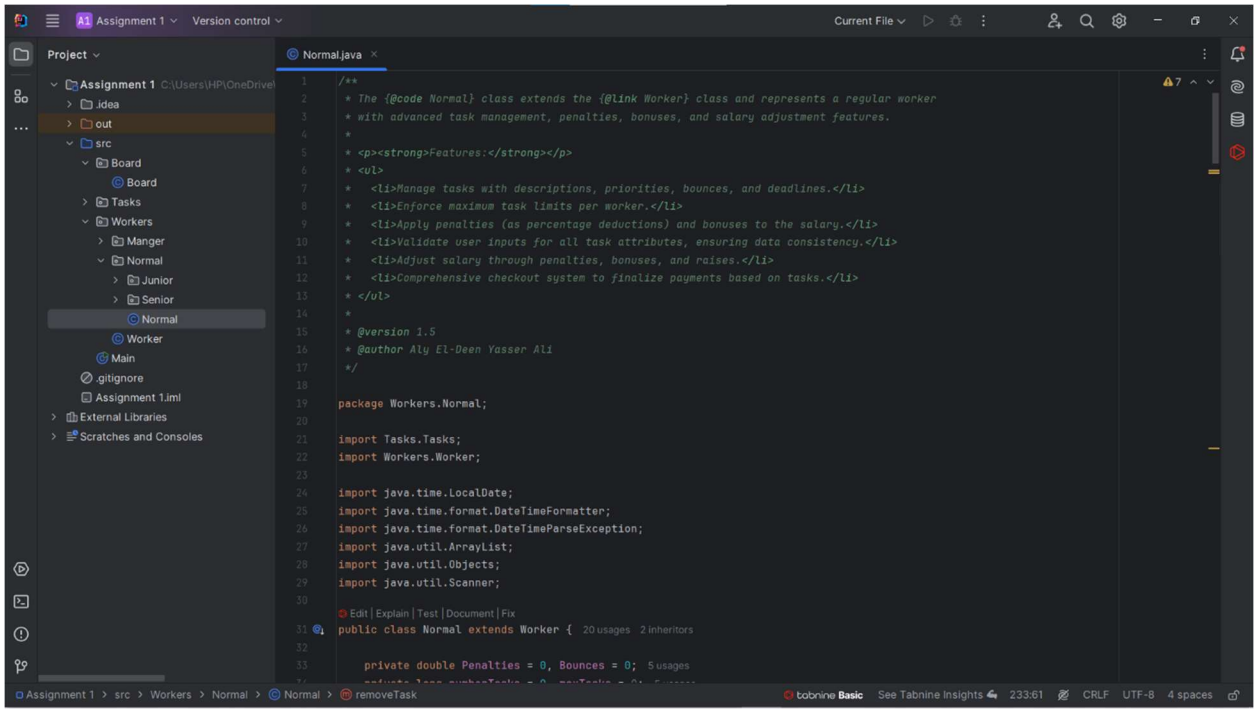
```
import java.util.ArrayList;
import Workers.Manger.Manger;
import Workers.Normal.Junior.Junior;
import Workers.Normal.Normal;
import Workers.Normal.Senior.Senior;
import Board.Board;

public class Main {

    public static void main(String[] args) {

        // Initialize manager list and add a default manager
        ArrayList<Manger> mangers = new ArrayList<>();
        mangers.add(new Manger("Aly", 20000, "General", "aly",
"11114"));

        // Initialize lists for Juniors and Seniors
        ArrayList<Junior> juniors = new ArrayList<>();
        ArrayList<Senior> seniors = new ArrayList<>();

        // Create the board with the existing worker lists
        Board board = new Board(juniors, mangers, seniors);

        // Main menu loop for user interaction
        while (true) {
            int menuChoice = Normal.getValidChoice(
                    new String[]{"Manager", "Senior", "Junior",
"Exit"},
                    "# ==== Welcome to Workers Management System ====
#"
            );

            // Navigate based on user choice
            if (menuChoice == 0)
                board.manage();
            else if (menuChoice == 1)
                board.seniorWork();
            else if (menuChoice == 2)
```

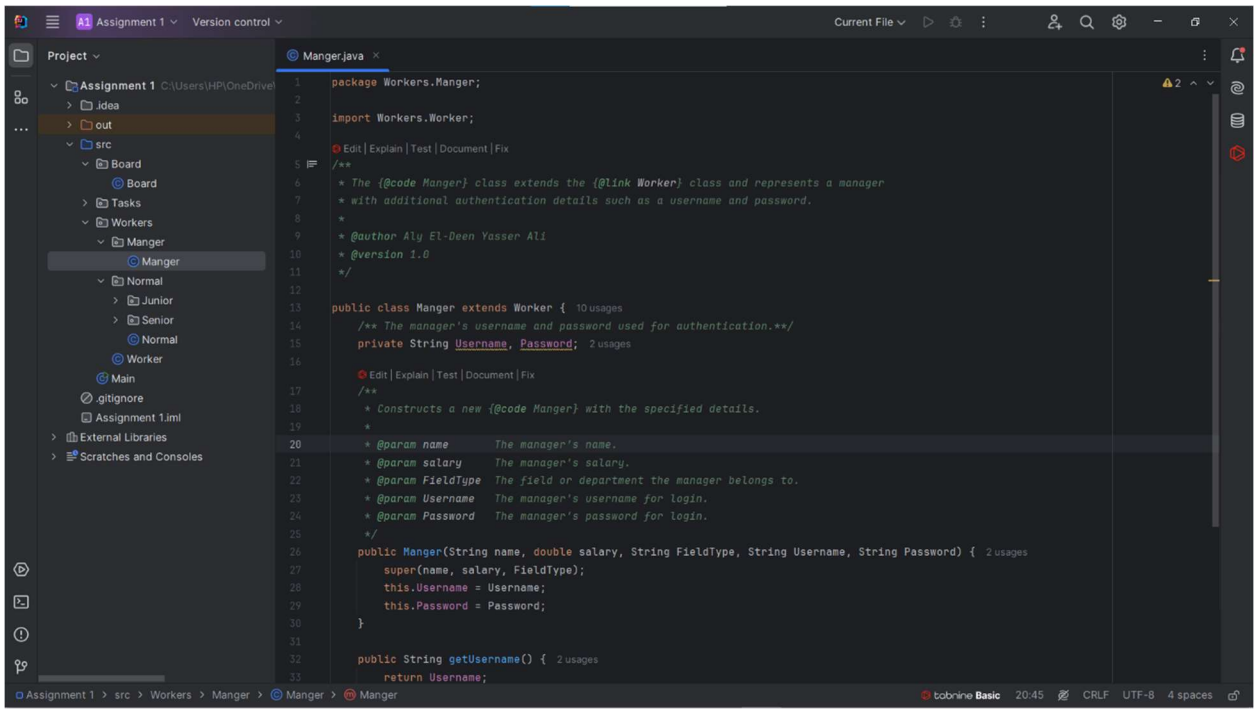
```
                board.juniorWork();
            else if (menuChoice == 3)
                break;
        }

        System.out.println("\nThanks For Using Our Program");
    }
}
```

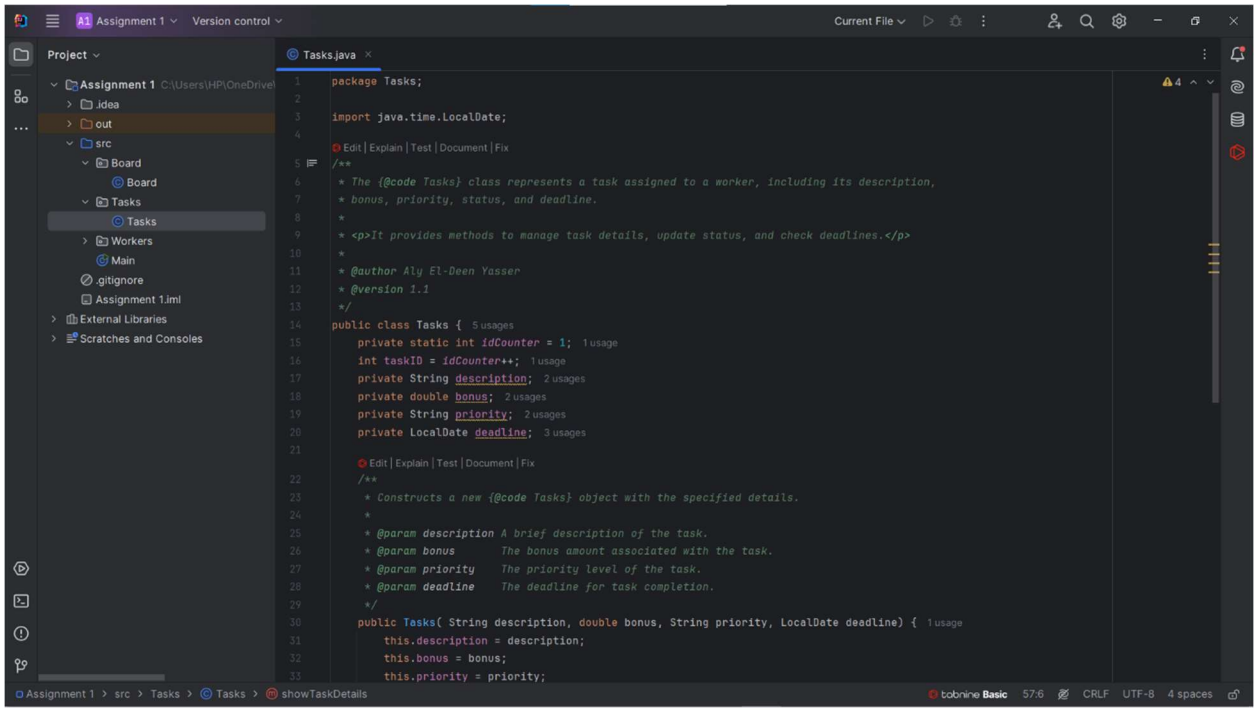
- Images For the code:

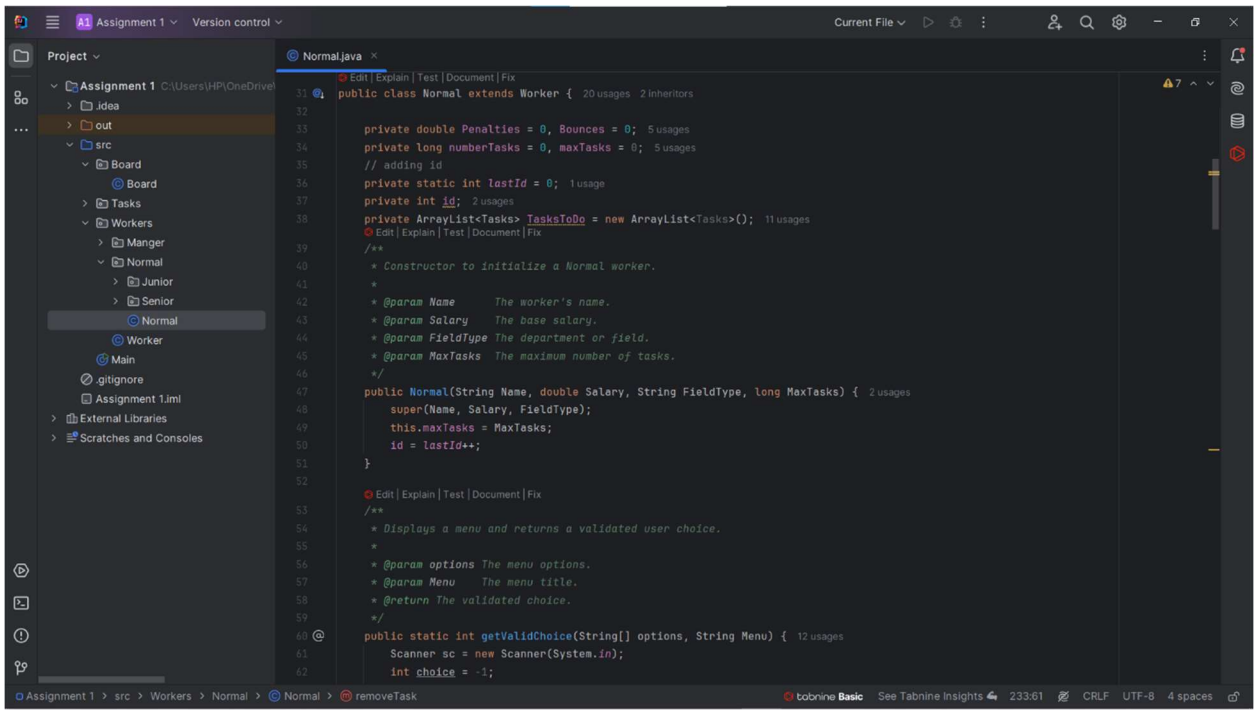


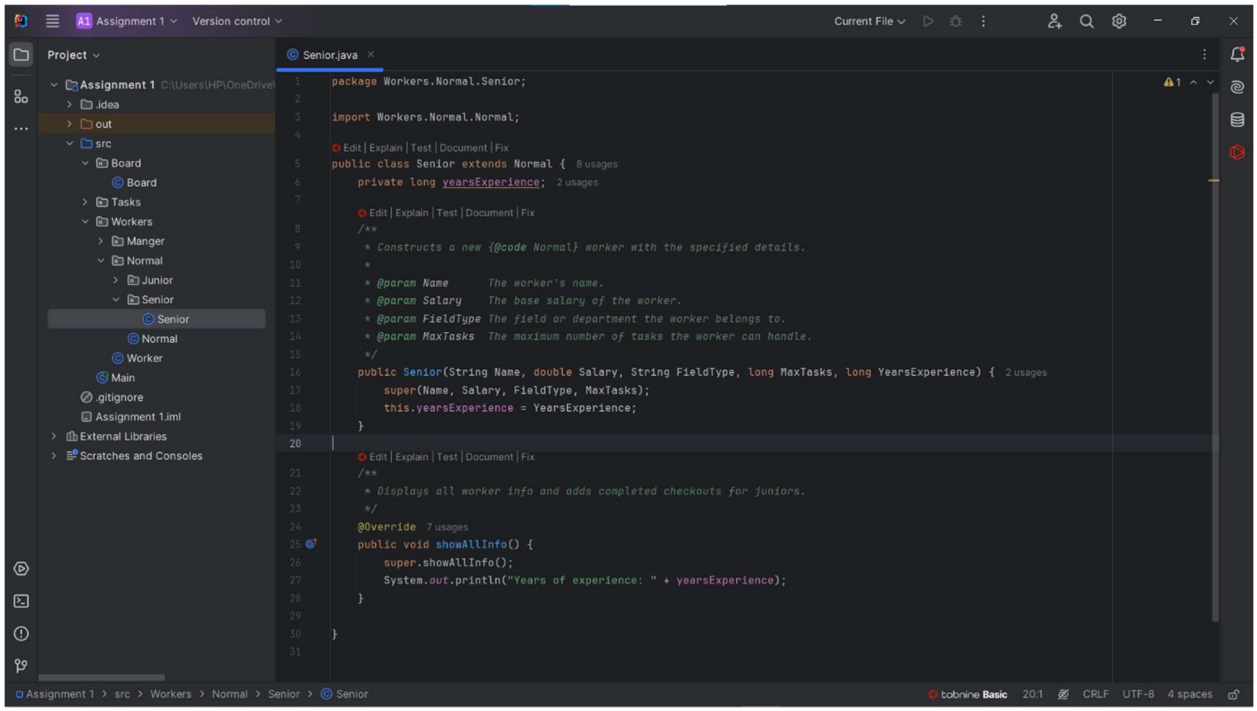


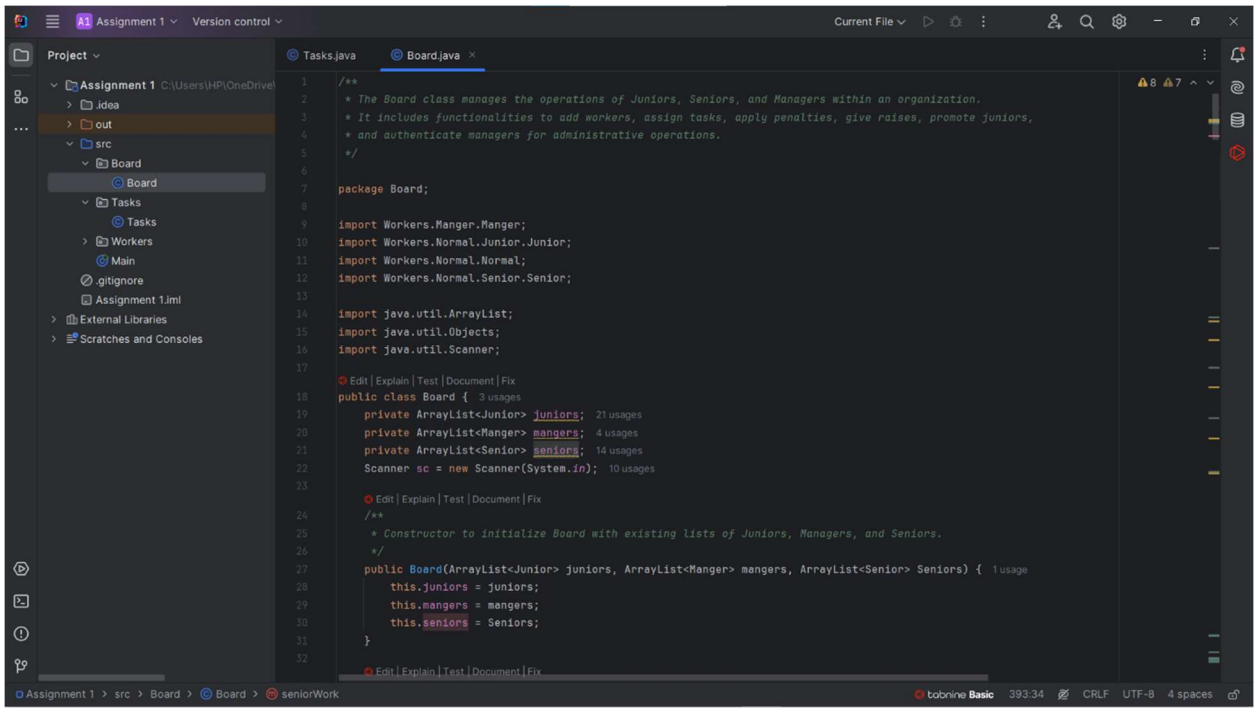

Assignment 1
Version control
Current File
Project
Normal.java
Assignment 1 C:\Users\HP\OneDrive\
.idea
out
src
Board
Board
Tasks
Workers
Manger
Normal
Junior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
/**
* The {@code Normal} class extends the {@link Worker} class and represents a regular worker
* with advanced task management, penalties, bonuses, and salary adjustment features.
*
* Features:
*
* Manage tasks with descriptions, priorities, bounces, and deadlines.
* Enforce maximum task limits per worker.
* Apply penalties (as percentage deductions) and bonuses to the salary.
* Validate user inputs for all task attributes, ensuring data consistency.
* Adjust salary through penalties, bonuses, and raises.
* Comprehensive checkout system to finalize payments based on tasks.
*
*
* @version 1.5
* @author Aly El-Deen Yasser Ali
*/
package Workers.Normal;
import Tasks.Tasks;
import Workers.Worker;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeParseException;
import java.util.ArrayList;
import java.util.Objects;
import java.util.Scanner;
Edit | Explain | Test | Document | Fix
public class Normal extends Worker { 20 usages 2 inheritors
private double Penalties = 0, Bounces = 0; 5 usages
Assignment 1 > src > Workers > Normal > Normal > removeTask
tabnine Basic
See Tabnine Insights
233:61
CRLF
UTF-8
4 spaces

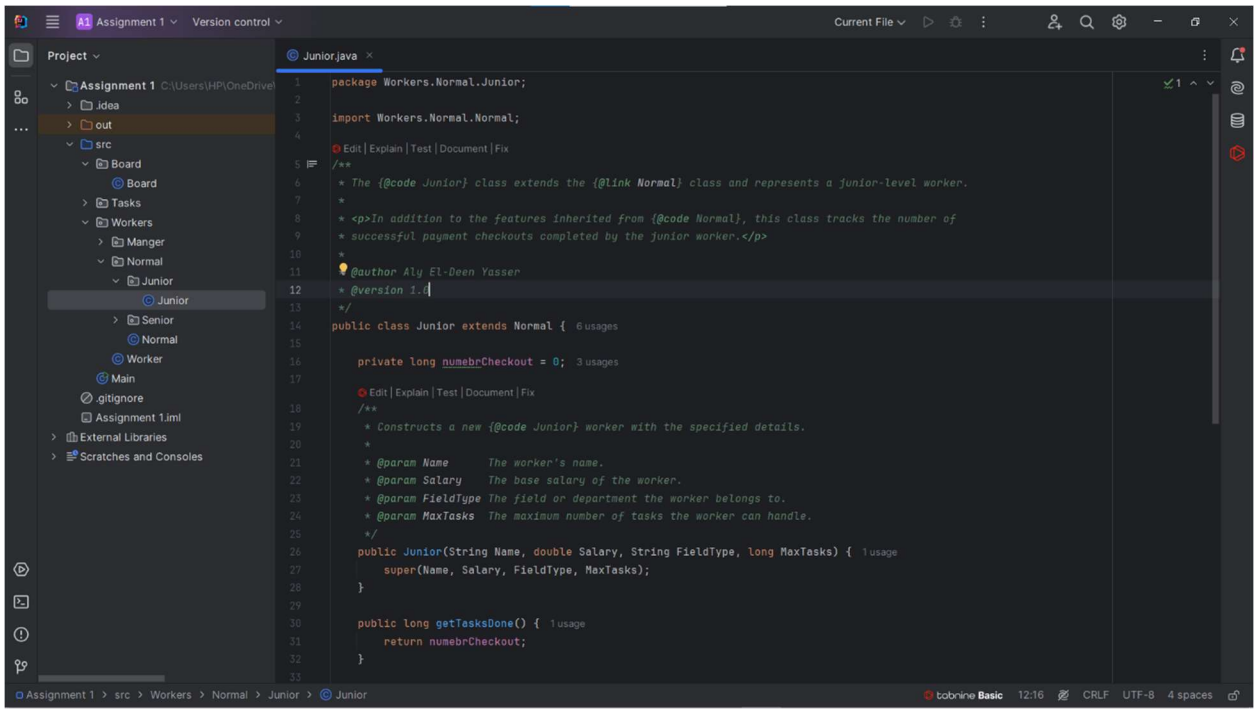

Assignment 1
Version control
Current File
Project
Manger.java
Assignment 1 C:\Users\HP\OneDrive\
.idea
out
src
Board
Board
Tasks
Workers
Manger
Manger
Normal
Junior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
package Workers.Manger;
import Workers.Worker;
Edit | Explain | Test | Document | Fix
/**
* The {@code Manger} class extends the {@link Worker} class and represents a manager
* with additional authentication details such as a username and password.
*
* @author Aly El-Deen Yasser Ali
* @version 1.0
*/
public class Manger extends Worker { 10 usages
/** The manager's username and password used for authentication.**/
private String Username, Password; 2 usages
Edit | Explain | Test | Document | Fix
/**
* Constructs a new {@code Manger} with the specified details.
*
* @param name The manager's name.
* @param salary The manager's salary.
* @param FieldType The field or department the manager belongs to.
* @param Username The manager's username for login.
* @param Password The manager's password for login.
*/
public Manger(String name, double salary, String FieldType, String Username, String Password) { 2 usages
super(name, salary, FieldType);
this.Username = Username;
this.Password = Password;
public String getUsername() { 2 usages
return Username;
Assignment 1 > src > Workers > Manger > Manger > Manger
tabnine Basic
20:45
CRLF
UTF-8
4 spaces


Assignment 1
Version control
Current File
Project
Tasks.java
Assignment 1 C:\Users\HP\OneDrive\
.idea
out
src
Board
Board
Tasks
Tasks
Workers
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
package Tasks;
import java.time.LocalDate;
Edit | Explain | Test | Document | Fix
/**
* The {@code Tasks} class represents a task assigned to a worker, including its description,
* bonus, priority, status, and deadline.
*
* It provides methods to manage task details, update status, and check deadlines.
*
* @author Aly El-Deen Yasser
* @version 1.1
*/
public class Tasks { 5 usages
private static int idCounter = 1; 1 usage
int taskID = idCounter++; 1 usage
private String description; 2 usages
private double bonus; 2 usages
private String priority; 2 usages
private LocalDate deadline; 3 usages
Edit | Explain | Test | Document | Fix
/**
* Constructs a new {@code Tasks} object with the specified details.
*
* @param description A brief description of the task.
* @param bonus The bonus amount associated with the task.
* @param priority The priority level of the task.
* @param deadline The deadline for task completion.
*/
public Tasks(String description, double bonus, String priority, LocalDate deadline) { 1 usage
this.description = description;
this.bonus = bonus;
this.priority = priority;
Assignment 1 › src › Tasks › Tasks › showTaskDetails
tabnine Basic
57:6
CRLF
UTF-8
4 spaces


Assignment 1
Version control
Current File
Project
Normal.java
Assignment 1 C:\Users\HP\OneDrive\
.idea
out
src
Board
Board
Tasks
Workers
Manger
Normal
Junior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
Edit | Explain | Test | Document | Fix
public class Normal extends Worker { 20 usages 2 inheritors
private double Penalties = 0, Bounces = 0; 5 usages
private long numberTasks = 0, maxTasks = 0; 5 usages
// adding id
private static int lastId = 0; 1 usage
private int id; 2 usages
private ArrayList<Tasks> TasksToDo = new ArrayList<Tasks>(); 11 usages
Edit | Explain | Test | Document | Fix
/**
* Constructor to initialize a Normal worker.
*
* @param Name The worker's name.
* @param Salary The base salary.
* @param FieldType The department or field.
* @param MaxTasks The maximum number of tasks.
*/
public Normal(String Name, double Salary, String FieldType, long MaxTasks) { 2 usages
super(Name, Salary, FieldType);
this.maxTasks = MaxTasks;
id = lastId++;
}
Edit | Explain | Test | Document | Fix
/**
* Displays a menu and returns a validated user choice.
*
* @param options The menu options.
* @param Menu The menu title.
* @return The validated choice.
*/
public static int getValidChoice(String[] options, String Menu) { 12 usages
Scanner sc = new Scanner(System.in);
int choice = -1;
Assignment 1 › src › Workers › Normal › Normal › removeTask
tabnine Basic
See Tabnine Insights
233:61
CRLF
UTF-8
4 spaces


Assignment 1
Version control
Current File
Project
Senior.java
Assignment 1 C:\Users\HP\OneDrive\
.idea
out
src
Board
Board
Tasks
Workers
Manger
Normal
Junior
Senior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
package Workers.Normal.Senior;
import Workers.Normal.Normal;
Edit | Explain | Test | Document | Fix
public class Senior extends Normal { 8 usages
private long yearsExperience; 2 usages
Edit | Explain | Test | Document | Fix
/**
* Constructs a new {@code Normal} worker with the specified details.
*
* @param Name The worker's name.
* @param Salary The base salary of the worker.
* @param FieldType The field or department the worker belongs to.
* @param MaxTasks The maximum number of tasks the worker can handle.
*/
public Senior(String Name, double Salary, String FieldType, long MaxTasks, long YearsExperience) { 2 usages
super(Name, Salary, FieldType, MaxTasks);
this.yearsExperience = YearsExperience;
}
Edit | Explain | Test | Document | Fix
/**
* Displays all worker info and adds completed checkouts for juniors.
*/
@Override 7 usages
public void showAllInfo() {
super.showAllInfo();
System.out.println("Years of experience: " + yearsExperience);
}
}
Assignment 1 > src > Workers > Normal > Senior > Senior
tabnine Basic
20:1
CRLF
UTF-8
4 spaces


Assignment 1
Version control
Current File
Project
Tasks.java
Board.java
Assignment 1 C:\Users\HP\OneDrive\
.idea
out
src
Board
Board
Tasks
Tasks
Workers
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
/**
* The Board class manages the operations of Juniors, Seniors, and Managers within an organization.
* It includes functionalities to add workers, assign tasks, apply penalties, give raises, promote juniors,
* and authenticate managers for administrative operations.
*/
package Board;
import Workers.Manger.Manger;
import Workers.Normal.Junior.Junior;
import Workers.Normal.Normal;
import Workers.Normal.Senior.Senior;
import java.util.ArrayList;
import java.util.Objects;
import java.util.Scanner;
Edit | Explain | Test | Document | Fix
public class Board { 3 usages
private ArrayList<Junior> juniors; 21 usages
private ArrayList<Manger> mangers; 4 usages
private ArrayList<Senior> seniors; 14 usages
Scanner sc = new Scanner(System.in); 10 usages
Edit | Explain | Test | Document | Fix
/**
* Constructor to initialize Board with existing lists of Juniors, Managers, and Seniors.
*/
public Board(ArrayList<Junior> juniors, ArrayList<Manger> mangers, ArrayList<Senior> Seniors) { 1 usage
this.juniors = juniors;
this.mangers = mangers;
this.seniors = Seniors;
}
Edit | Explain | Test | Document | Fix
Assignment 1 > src > Board > Board > seniorWork
tabnine Basic
393:34
CRLF
UTF-8
4 spaces



Video Link: https://youtu.be/-brwmS8jSZo

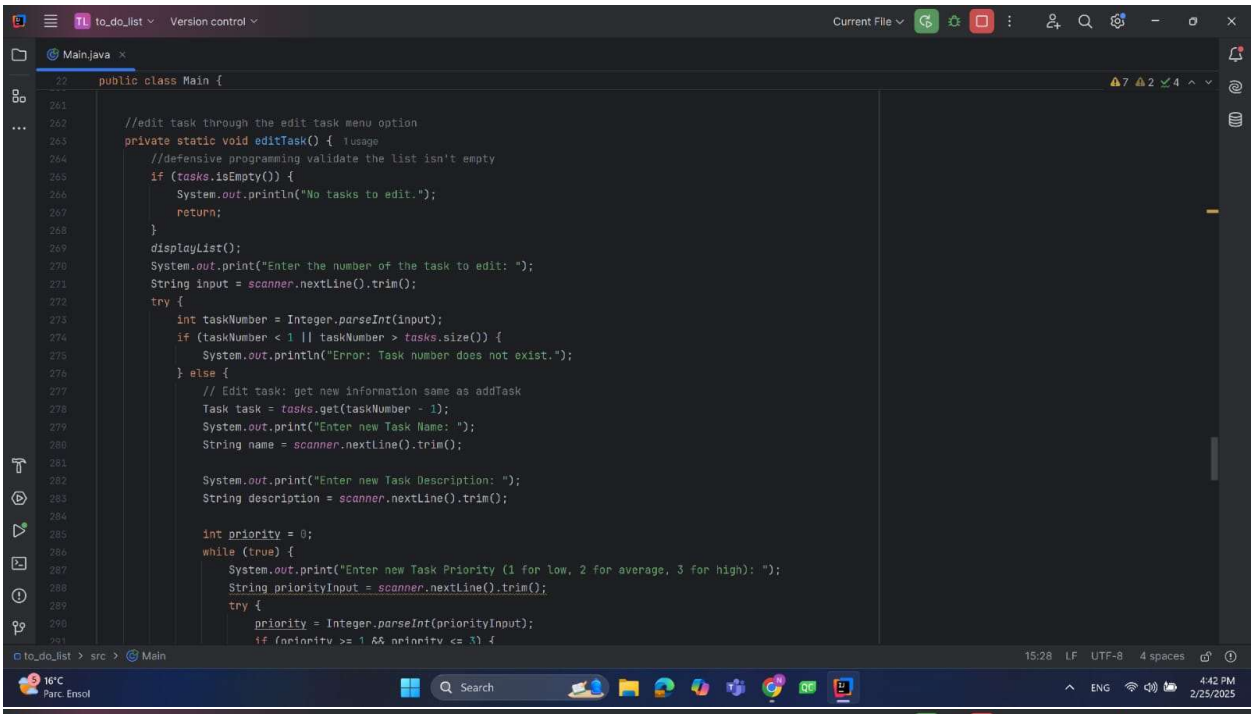
Nagham Wael:

- Hours of Study: 12h
- Source of Study: Geeks For Geeks + https://youtube.com/playlist?list=PLCInYL3l2AajYlZGzU_LVrHdoouf8W6ZN&si=tfRlF2iNxOtJehum
- Main Code Logic:

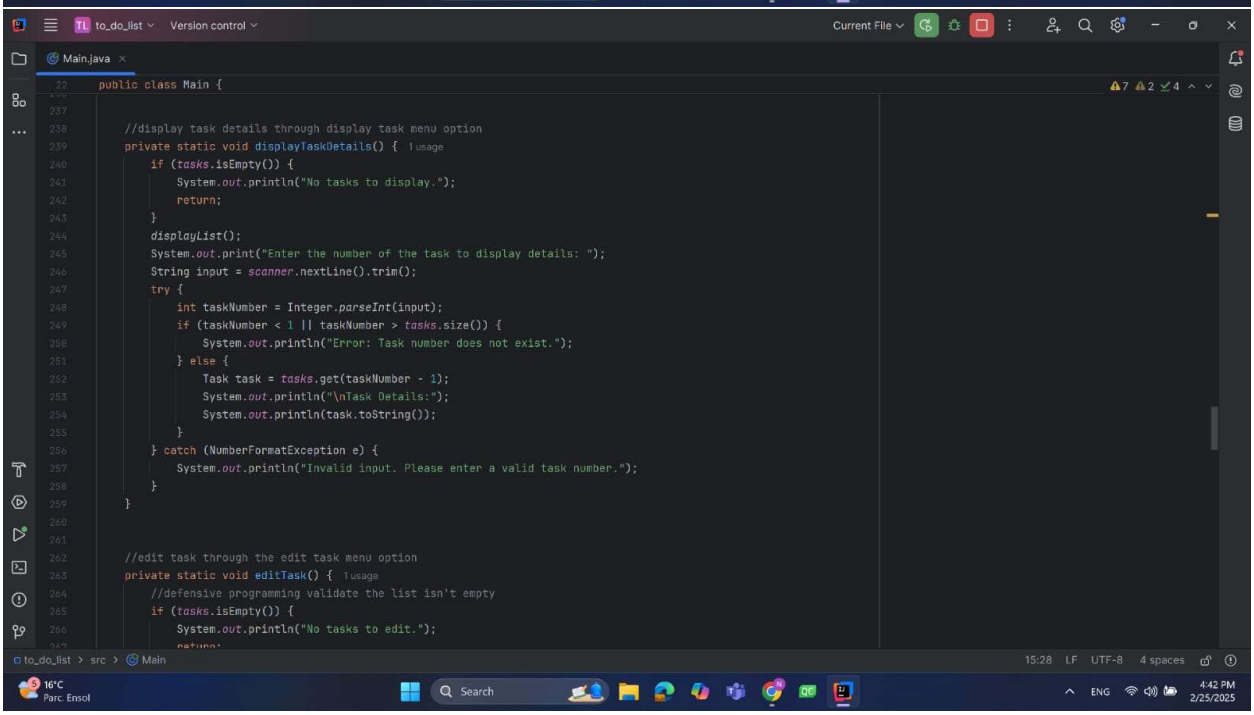
```
•  public static void main(String[] args) {
        boolean running = true;

        //display menu while the program is running(user didn't choose exit)
        while (running) {
            printMenu();
            String choice = scanner.nextLine().trim();
            switch (choice) {
                case "1":
                    addTask();
                    break;
                case "2":
                    removeTask();
                    break;
                case "3":
                    displayTaskDetails();
                    break;
                case "4":
                    editTask();
                    break;
                case "5":
                    sortTasks();
                    break;
                case "6":
                    displayList();
                    break;
                case "7":
                    clearList();
                    break;
                case "8":
                    saveListToFile();
                    break;
                case "9":
                    running = false;
                    System.out.println("Exiting program...");
                    break;
                default:
                    System.out.println("Invalid option. Please choose from 1
    to 9.");
            }
```

```
        }

        scanner.close();
    }
```

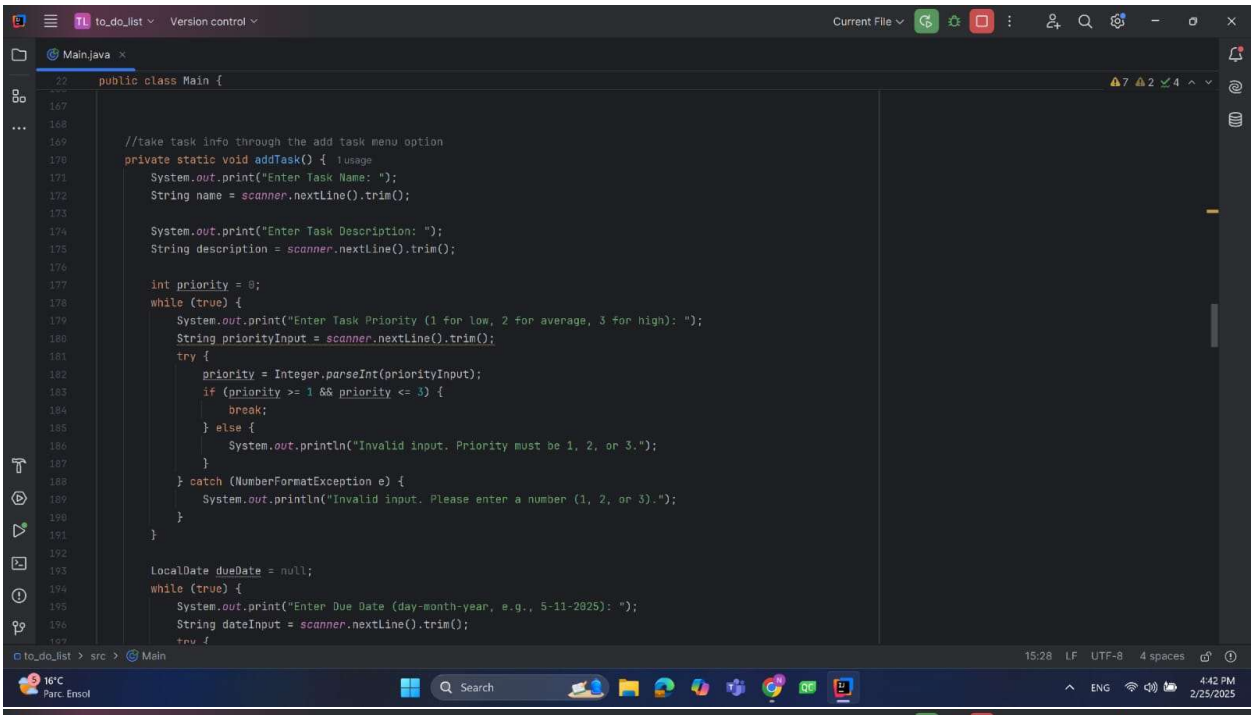
- Images for code :



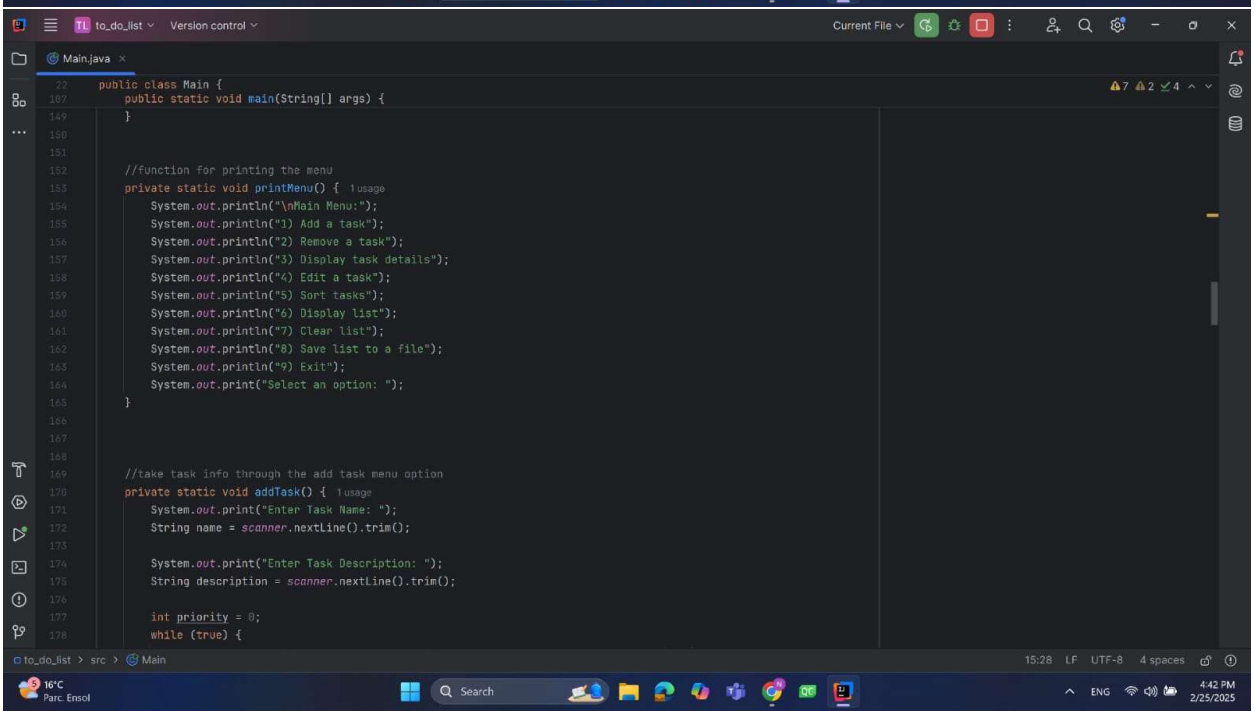
```
    public class Main {


        //edit task through the edit task menu option
        private static void editTask() {  1 usage
            //defensive programming validate the list isn't empty
            if (tasks.isEmpty()) {
                System.out.println("No tasks to edit.");
                return;
            }
            displayList();
            System.out.print("Enter the number of the task to edit: ");
            String input = scanner.nextLine().trim();
            try {
                int taskNumber = Integer.parseInt(input);
                if (taskNumber < 1 || taskNumber > tasks.size()) {
                    System.out.println("Error: Task number does not exist.");
                } else {
                    // Edit task: get new information same as addTask
                    Task task = tasks.get(taskNumber - 1);
                    System.out.print("Enter new Task Name: ");
                    String name = scanner.nextLine().trim();

                    System.out.print("Enter new Task Description: ");
                    String description = scanner.nextLine().trim();

                    int priority = 0;
                    while (true) {
                        System.out.print("Enter new Task Priority (1 for low, 2 for average, 3 for high): ");
                        String priorityInput = scanner.nextLine().trim();
                        try {
                            priority = Integer.parseInt(priorityInput);
                            if (priority >= 1 && priority <= 3) {
```



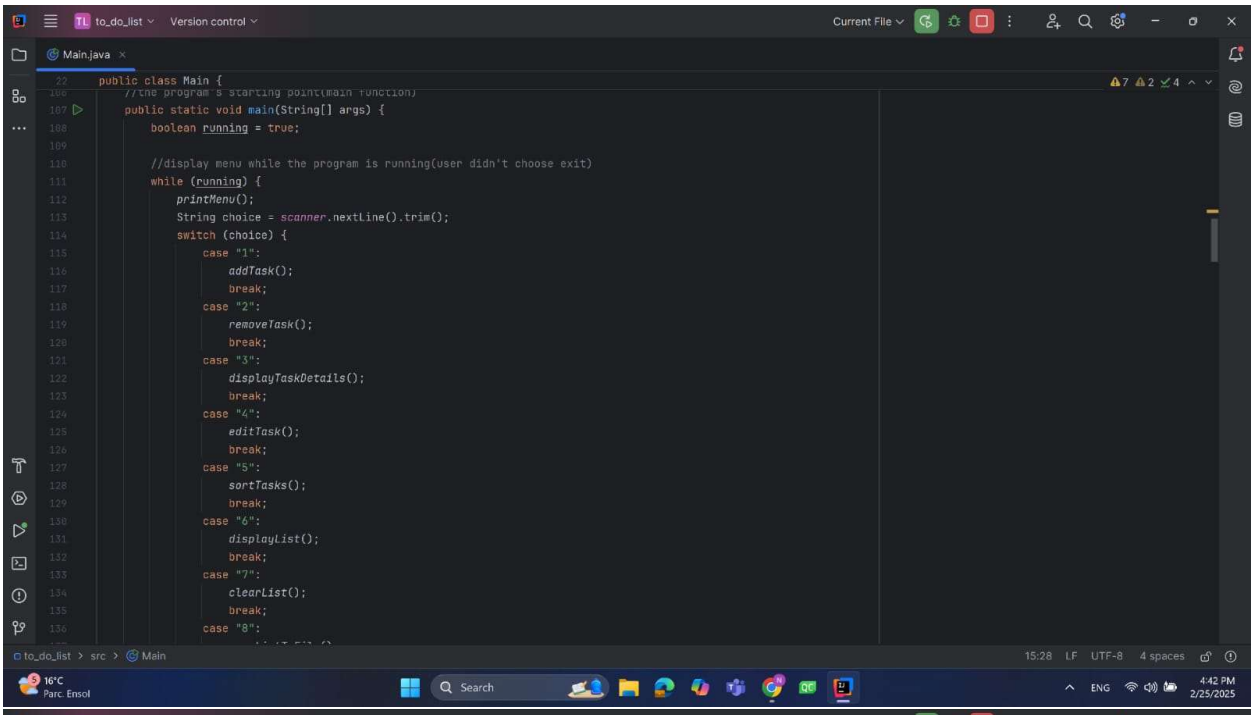
```
    public class Main {


        //display task details through display task menu option
        private static void displayTaskDetails() {  1 usage
            if (tasks.isEmpty()) {
                System.out.println("No tasks to display.");
                return;
            }
            displayList();
            System.out.print("Enter the number of the task to display details: ");
            String input = scanner.nextLine().trim();
            try {
                int taskNumber = Integer.parseInt(input);
                if (taskNumber < 1 || taskNumber > tasks.size()) {
                    System.out.println("Error: Task number does not exist.");
                } else {
                    Task task = tasks.get(taskNumber - 1);
                    System.out.println("\nTask Details:");
                    System.out.println(task.toString());
                }
            } catch (NumberFormatException e) {
                System.out.println("Invalid input. Please enter a valid task number.");
            }
        }


        //edit task through the edit task menu option
        private static void editTask() {  1 usage
            //defensive programming validate the list isn't empty
            if (tasks.isEmpty()) {
                System.out.println("No tasks to edit.");
```



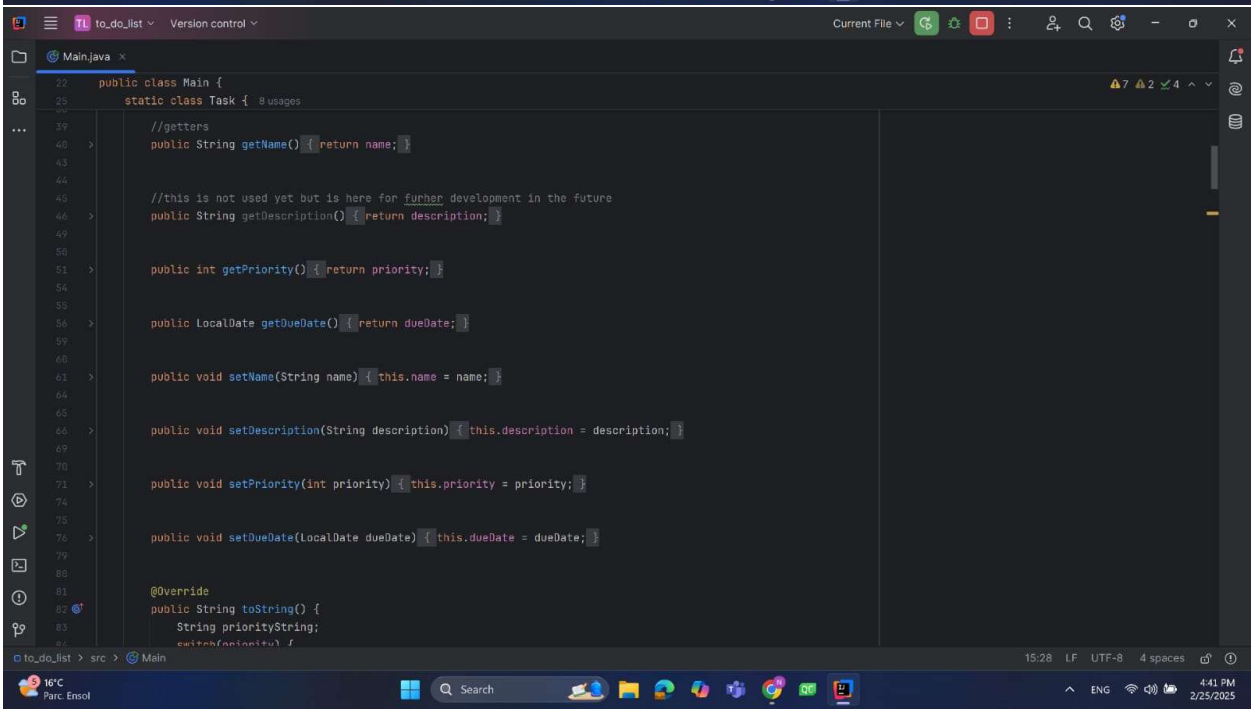
```
public class Main {



    //take task info through the add task menu option
    private static void addTask() {  1 usage
        System.out.print("Enter Task Name: ");
        String name = scanner.nextLine().trim();

        System.out.print("Enter Task Description: ");
        String description = scanner.nextLine().trim();

        int priority = 0;
        while (true) {
            System.out.print("Enter Task Priority (1 for low, 2 for average, 3 for high): ");
            String priorityInput = scanner.nextLine().trim();
            try {
                priority = Integer.parseInt(priorityInput);
                if (priority >= 1 && priority <= 3) {
                    break;
                } else {
                    System.out.println("Invalid input. Priority must be 1, 2, or 3.");
                }
            } catch (NumberFormatException e) {
                System.out.println("Invalid input. Please enter a number (1, 2, or 3).");
            }
        }

        LocalDate dueDate = null;
        while (true) {
            System.out.print("Enter Due Date (day-month-year, e.g., 5-11-2025): ");
            String dateInput = scanner.nextLine().trim();
            try {
```



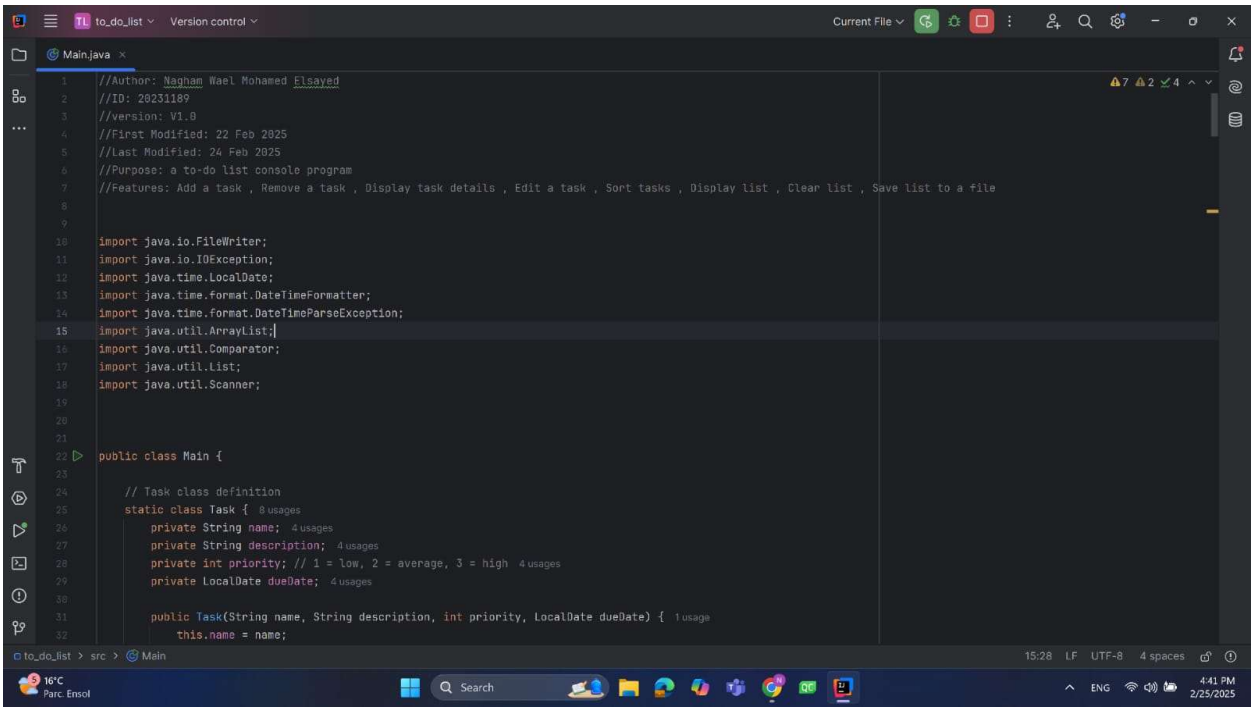
```
public class Main {
    public static void main(String[] args) {
    }



    //function for printing the menu
    private static void printMenu() {  1 usage
        System.out.println("\nMain Menu:");
        System.out.println("1) Add a task");
        System.out.println("2) Remove a task");
        System.out.println("3) Display task details");
        System.out.println("4) Edit a task");
        System.out.println("5) Sort tasks");
        System.out.println("6) Display list");
        System.out.println("7) Clear list");
        System.out.println("8) Save list to a file");
        System.out.println("9) Exit");
        System.out.print("Select an option: ");
    }



    //take task info through the add task menu option
    private static void addTask() {  1 usage
        System.out.print("Enter Task Name: ");
        String name = scanner.nextLine().trim();

        System.out.print("Enter Task Description: ");
        String description = scanner.nextLine().trim();

        int priority = 0;
        while (true) {
```



```
public class Main {
    //the program's starting point(main function)
    public static void main(String[] args) {
        boolean running = true;

        //display menu while the program is running(user didn't choose exit)
        while (running) {
            printMenu();
            String choice = scanner.nextLine().trim();
            switch (choice) {
                case "1":
                    addTask();
                    break;
                case "2":
                    removeTask();
                    break;
                case "3":
                    displayTaskDetails();
                    break;
                case "4":
                    editTask();
                    break;
                case "5":
                    sortTasks();
                    break;
                case "6":
                    displayList();
                    break;
                case "7":
                    clearList();
                    break;
                case "8":
```



```
public class Main {
    static class Task {  8 usages

        //getters
        public String getName() { return name; }


        //this is not used yet but is here for furher development in the future
        public String getDescription() { return description; }


        public int getPriority() { return priority; }


        public LocalDate getDueDate() { return dueDate; }


        public void setName(String name) { this.name = name; }


        public void setDescription(String description) { this.description = description; }


        public void setPriority(int priority) { this.priority = priority; }


        public void setDueDate(LocalDate dueDate) { this.dueDate = dueDate; }


        @Override
        public String toString() {
            String priorityString;
```



Video Link:

https://youtu.be/VqOBuBvfRiw?si=WCep92XFi12x7LnN

Fatema El-Zhraa Ahmed:

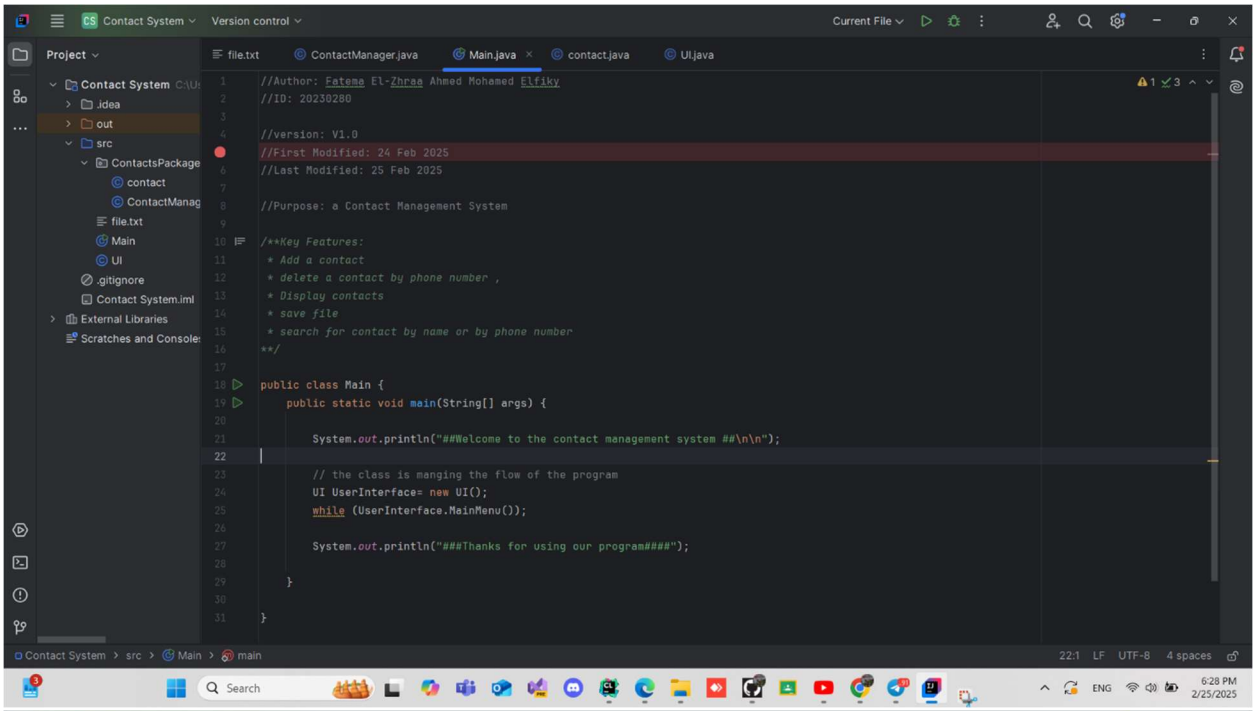
- Hours of Study:12 hours
- Source of Study : https://www.youtube.com/playlist?list=PLCInYL3l2AajYlZGzU_LVrHdoouf8W6ZN
- Main Code Logic:

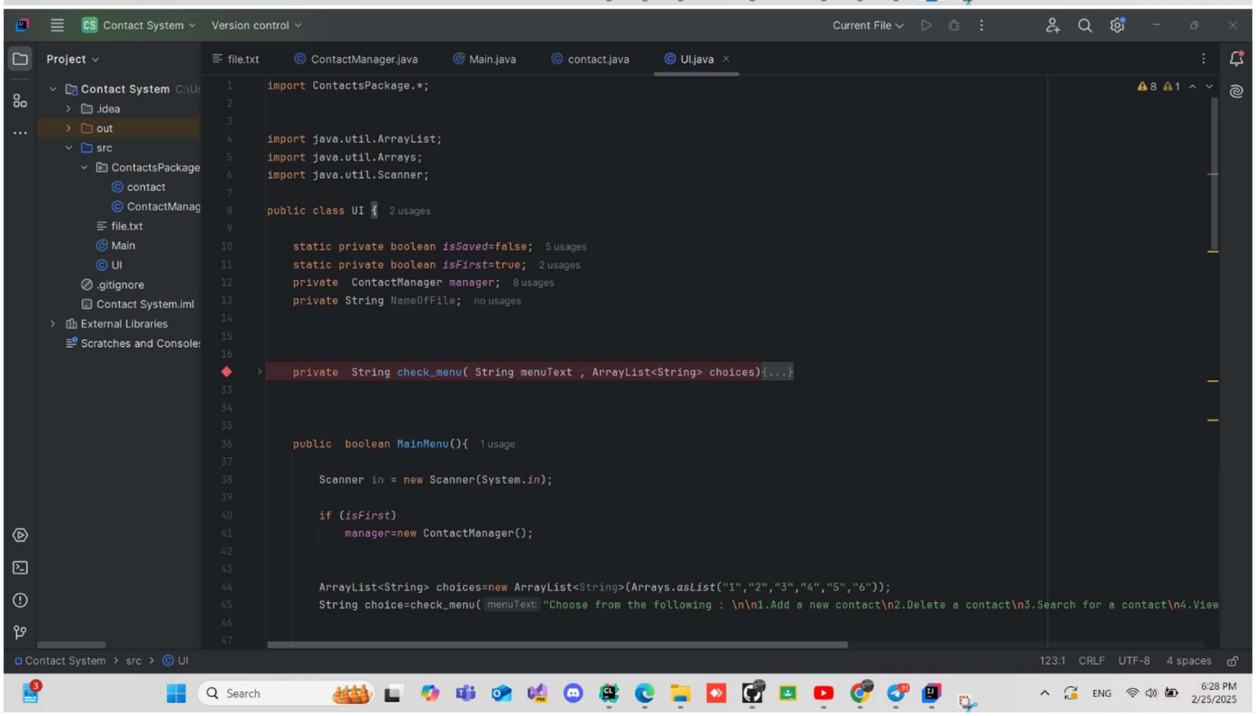
```
• public  boolean MainMenu(){

    Scanner in = new Scanner(System.in);

    if (isFirst)
        manager=new ContactManager();


    ArrayList<String> choices=new
ArrayList<String>(Arrays.asList("1","2","3","4","5","6"));
    String choice=check_menu("Choose from the following :
\n\n1.Add a new contact\n2.Delete a contact\n3.Search for a
contact\n4.View all contacts\n5.Save contacts in a
file\n6.Exit\n\nEnter your answer : ",choices );


    if(choice.equals("1")){//Add

        boolean isRepeated=manager.Add();
        if(!isRepeated)
            System.out.println("Sorry , it will not be added
for repeating phone number or email");
        else
            isSaved=false;

    }
    else if( choice.equals("2")){//Delete

        if(manager.Delete())
            isSaved=false;

    }
    else if( choice.equals("3")){//search for (by name ,
```

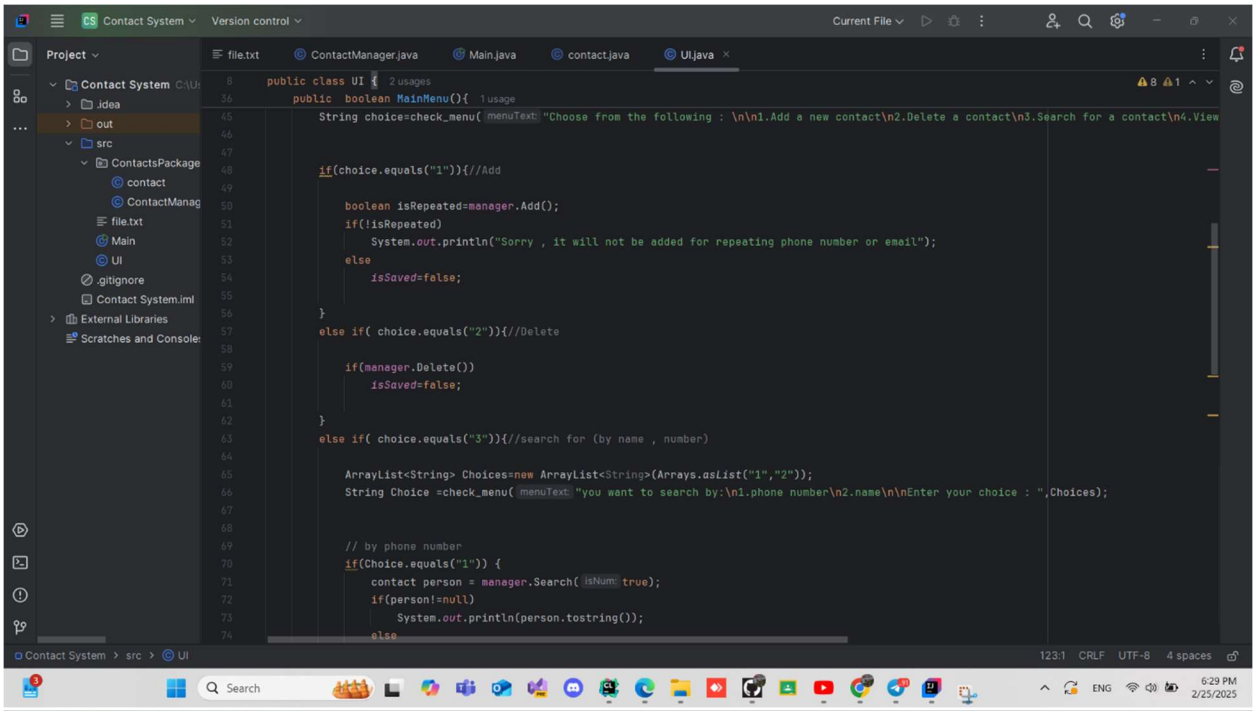
```
number)

        ArrayList<String> Choices=new
ArrayList<String>(Arrays.asList("1","2"));
        String Choice =check_menu("you want to search
by:\n1.phone number\n2.name\n\nEnter your choice :
",Choices);

        if(Choice.equals("1")) {
            contact person = manager.Search(true);
            if(person!=null)
                System.out.println(person.tostring());
            else
                System.out.printf("Sorry the contact is not
found\n\n");

        }
        else {
            contact person = manager.Search(false);
            if(person!=null)
                System.out.println(person.tostring());
            else
                System.out.printf("Sorry the contact is not
found\n\n");

        }

    }
    else if( choice.equals("4")){//view all contacts

       manager.Display();

    }
    else if( choice.equals("5")){//Save contacts in file

        if(!isSaved)
            manager.Save();

        isSaved=true;
```

```
    }else{//Exit

        if(!isSaved){

            ArrayList<String>Choices=new
ArrayList<String>(Arrays.asList("1","2"));
            String Choice=check_menu("Do you want to save
changes before exiting?\n1.Yes\n2.No\n\nEnter your choice :
",Choices);

            if(Choice.equals("1"))// as task 5
                manager.Save();

        }

        return false;
    }

    isFirst=false;
    return true;

}
```

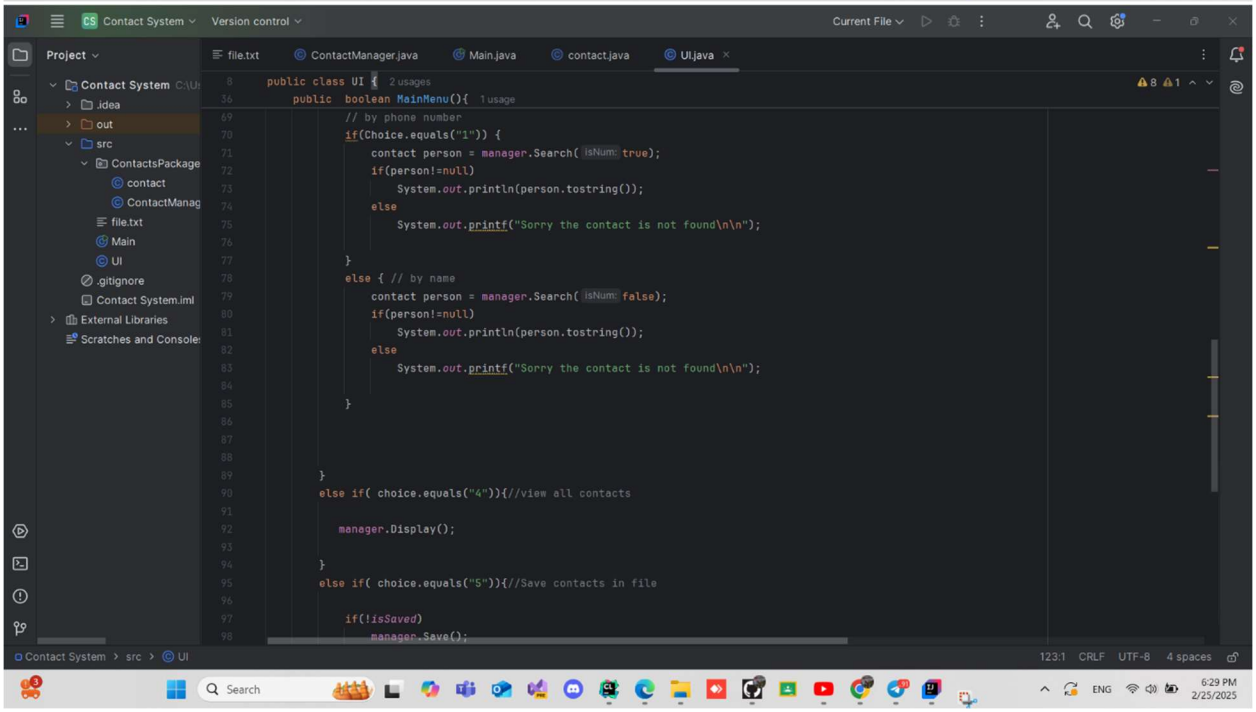
Screen Shots :


Contact System
Version control
Current File
Project
file.txt
ContactManager.java
Main.java
contact.java
UI.java
//Author: Fateme El-Zhraa Ahmed Mohamed Elfiky
//ID: 20230280
//version: V1.0
//First Modified: 24 Feb 2025
//Last Modified: 25 Feb 2025
//Purpose: a Contact Management System
/**Key Features:
* Add a contact
* delete a contact by phone number ,
* Display contacts
* save file
* search for contact by name or by phone number
**/
public class Main {
public static void main(String[] args) {
System.out.println("##Welcome to the contact management system ##\n\n");
// the class is manging the flow of the program
UI UserInterface= new UI();
while (UserInterface.MainMenu());
System.out.println("###Thanks for using our program####");
}
}
Contact System › src › Main › main
22:1 LF UTF-8 4 spaces
Search
ENG
6:28 PM
2/25/2025

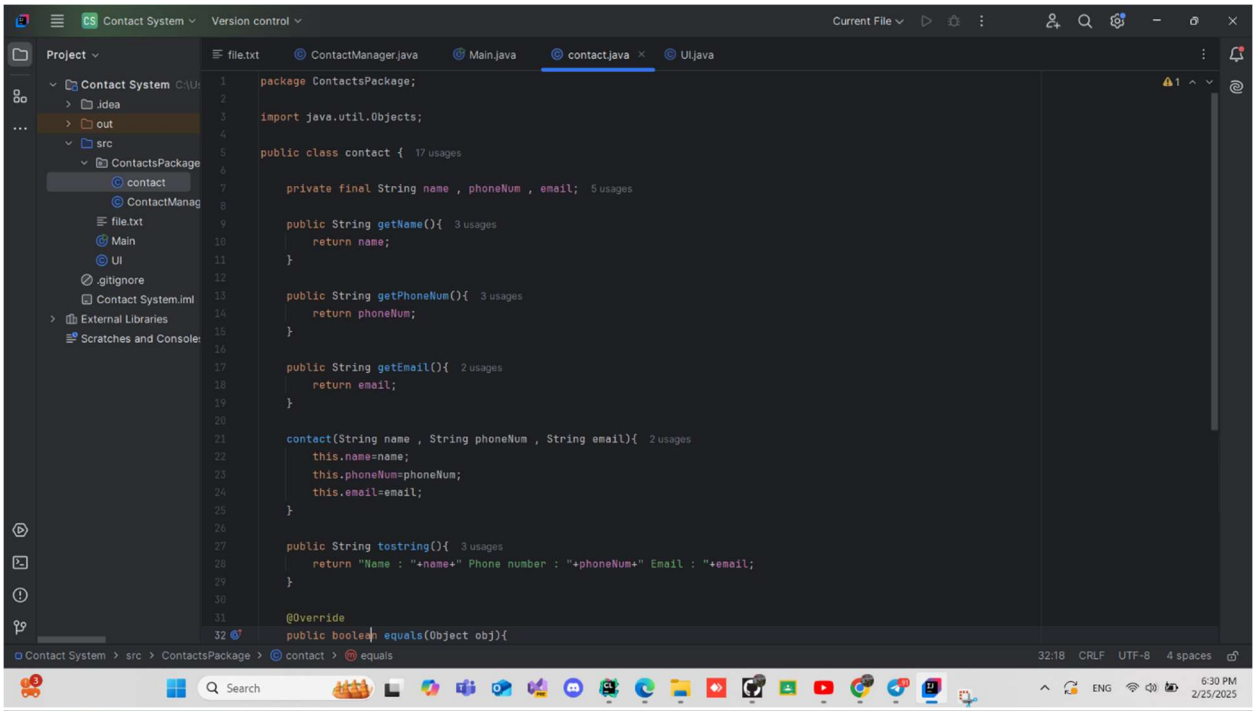

Contact System
Version control
Current File
Project
file.txt
ContactManager.java
Main.java
contact.java
UI.java
import ContactsPackage.*;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;
public class UI {
static private boolean isSaved=false;
static private boolean isFirst=true;
private ContactManager manager;
private String NameOfFile;
private String check_menu(String menuText , ArrayList<String> choices){...}
public boolean MainMenu(){
Scanner in = new Scanner(System.in);
if (isFirst)
manager=new ContactManager();
ArrayList<String> choices=new ArrayList<String>(Arrays.asList("1","2","3","4","5","6"));
String choice=check_menu(menuText: "Choose from the following : \n\n1.Add a new contact\n2.Delete a contact\n3.Search for a contact\n4.View
Contact System › src › UI
123:1 CRLF UTF-8 4 spaces
Search
ENG
6:28 PM
2/25/2025

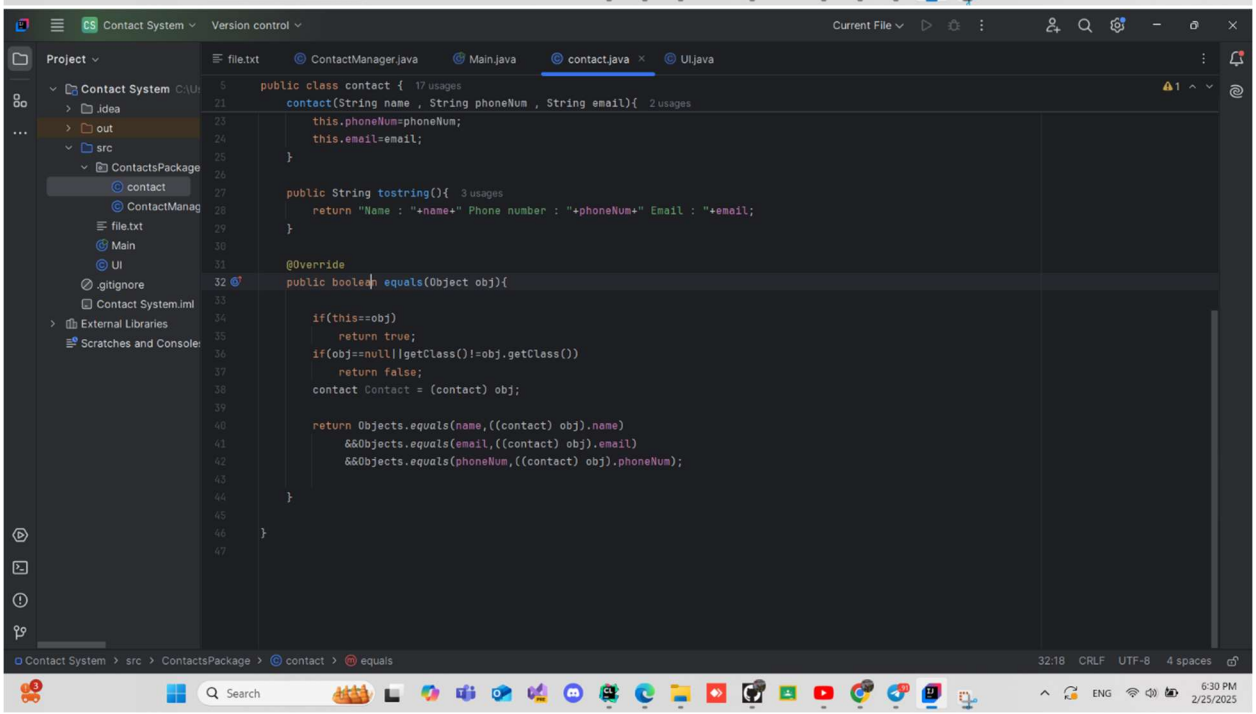


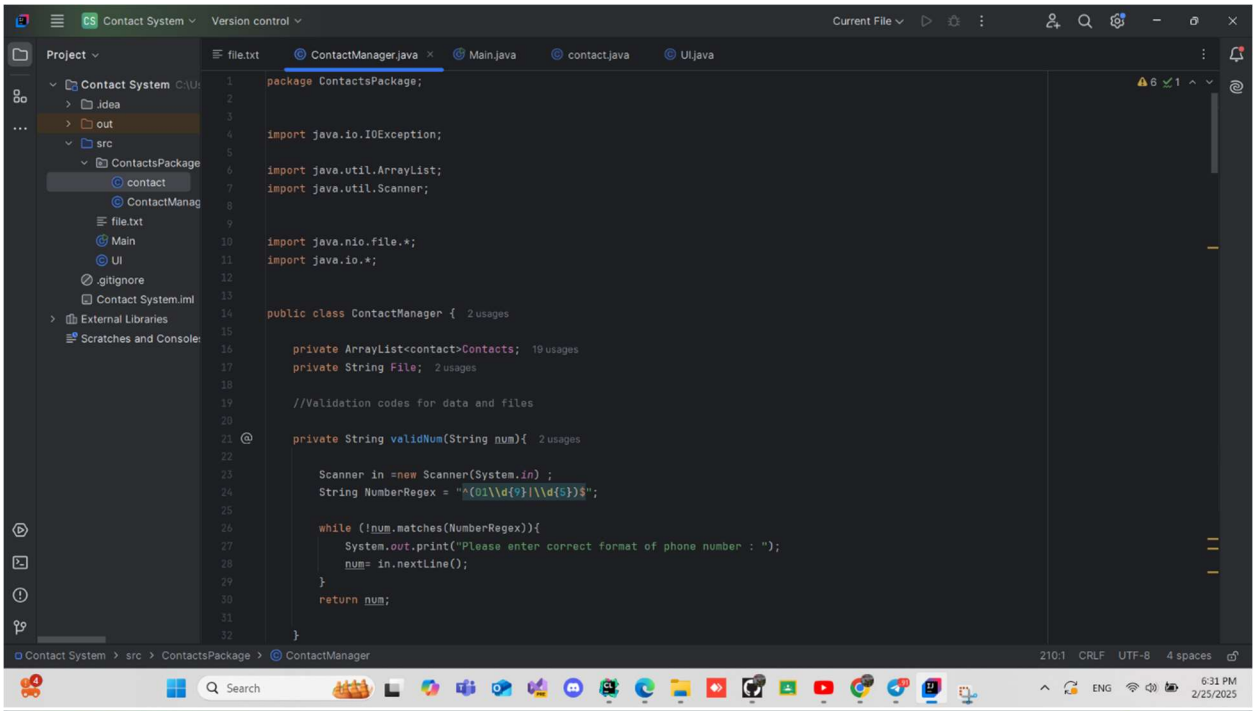
```
public class UI {  2 usages
    public  boolean MainMenu(){  1 usage
        String choice=check_menu( menuText: "Choose from the following : \n\n1.Add a new contact\n2.Delete a contact\n3.Search for a contact\n4.View

        if(choice.equals("1")){//Add

            boolean isRepeated=manager.Add();
            if(!isRepeated)
                System.out.println("Sorry , it will not be added for repeating phone number or email");
            else
                isSaved=false;

        }
        else if( choice.equals("2")){//Delete

            if(manager.Delete())
                isSaved=false;

        }
        else if( choice.equals("3")){//search for (by name , number)

            ArrayList<String> Choices=new ArrayList<String>(Arrays.asList("1","2"));
            String Choice =check_menu( menuText: "you want to search by:\n1.phone number\n2.name\n\nEnter your choice : ",Choices);


            // by phone number
            if(Choice.equals("1")) {
                contact person = manager.Search( isNum: true);
                if(person!=null)
                    System.out.println(person.tostring());
                else
```

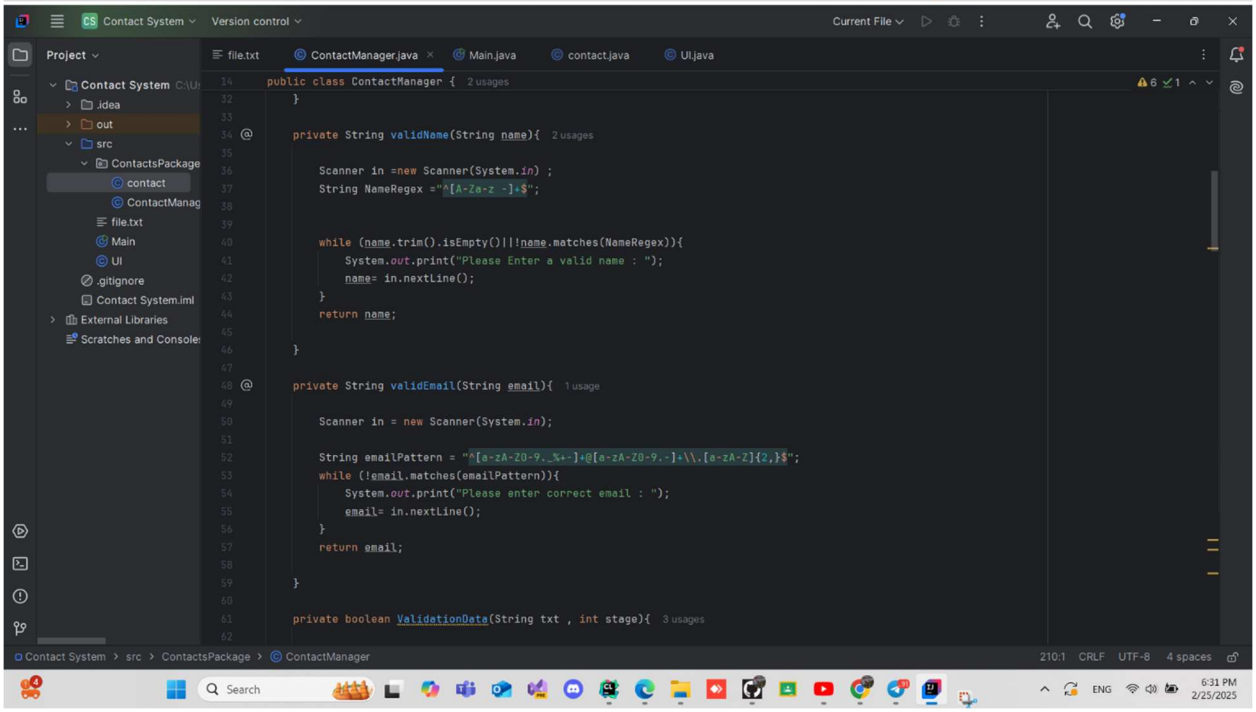


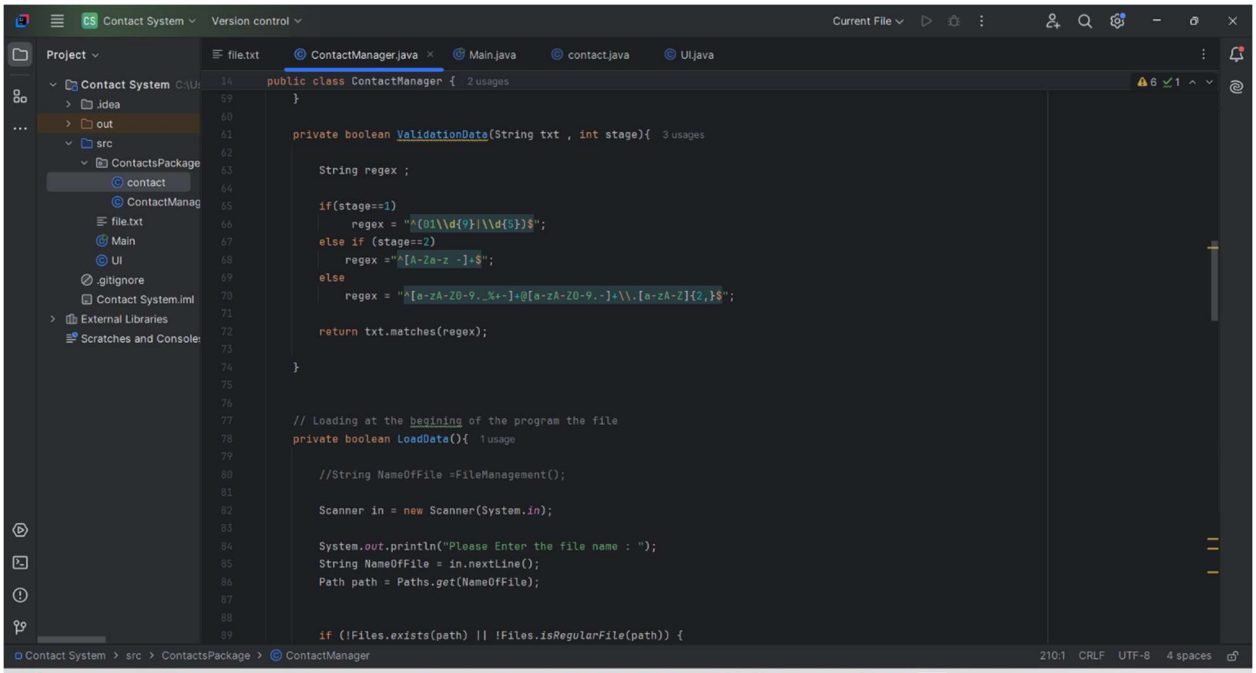
```
public class UI {  2 usages
    public  boolean MainMenu(){  1 usage
            // by phone number
            if(Choice.equals("1")) {
                contact person = manager.Search( isNum: true);
                if(person!=null)
                    System.out.println(person.tostring());
                else
                    System.out.printf("Sorry the contact is not found\n\n");

            }
            else { // by name
                contact person = manager.Search( isNum: false);
                if(person!=null)
                    System.out.println(person.tostring());
                else
                    System.out.printf("Sorry the contact is not found\n\n");


            }



        }
        else if( choice.equals("4")){//view all contacts

            manager.Display();

        }
        else if( choice.equals("5")){//Save contacts in file

            if(!isSaved)
                manager.Save();
```

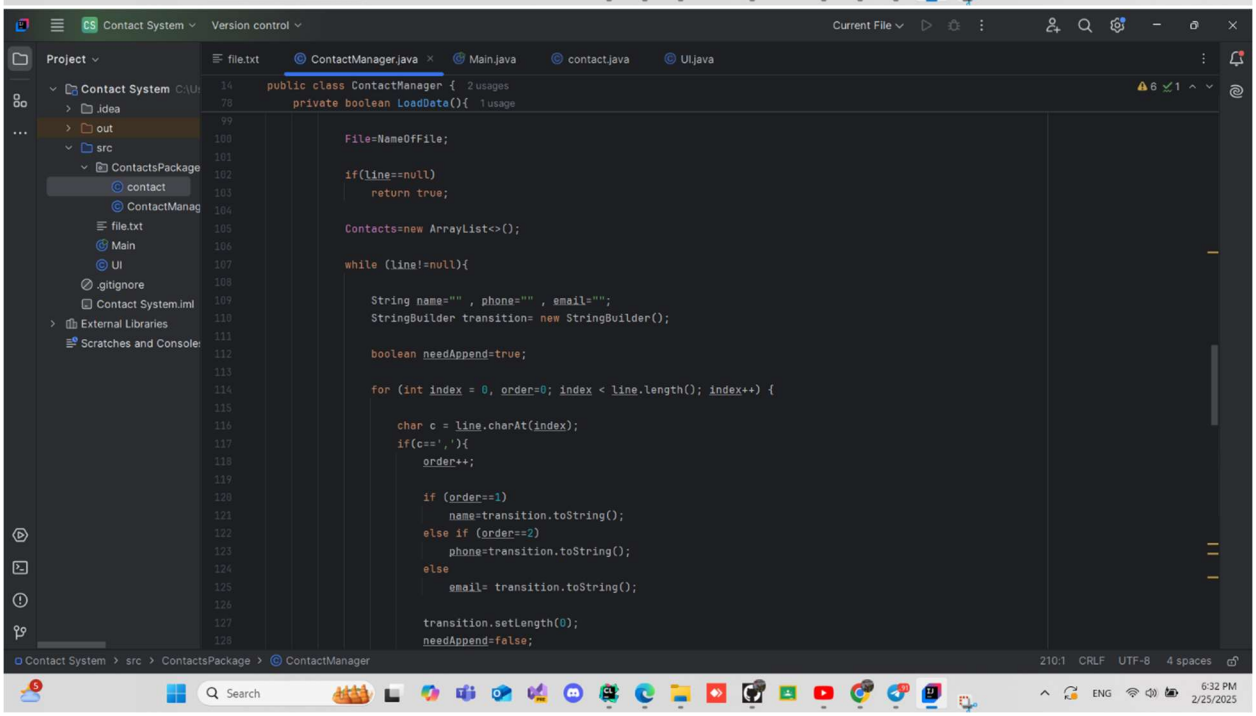

Contact System
Version control
Current File
Project
file.txt
ContactManager.java
Main.java
contact.java
UI.java
Contact System
.idea
out
src
ContactsPackage
contact
ContactManag
file.txt
Main
UI
.gitignore
Contact System.iml
External Libraries
Scratches and Consoles
package ContactsPackage;
import java.util.Objects;
public class contact { 17 usages
private final String name , phoneNum , email; 5 usages
public String getName(){ 3 usages
return name;
}
public String getPhoneNum(){ 3 usages
return phoneNum;
}
public String getEmail(){ 2 usages
return email;
}
contact(String name , String phoneNum , String email){ 2 usages
this.name=name;
this.phoneNum=phoneNum;
this.email=email;
}
public String tostring(){ 3 usages
return "Name : "+name+" Phone number : "+phoneNum+" Email : "+email;
}
@Override
public boolean equals(Object obj){
Contact System › src › ContactsPackage › contact › equals
32:18 CRLF UTF-8 4 spaces
Search
ENG
6:30 PM
2/25/2025


Contact System
Version control
Current File
Project
file.txt
ContactManager.java
Main.java
contact.java
UI.java
public class contact { 17 usages
contact(String name , String phoneNum , String email){ 2 usages
this.phoneNum=phoneNum;
this.email=email;
}
public String tostring(){ 3 usages
return "Name : "+name+" Phone number : "+phoneNum+" Email : "+email;
}
@Override
public boolean equals(Object obj){
if(this==obj)
return true;
if(obj==null||getClass()!=obj.getClass())
return false;
contact Contact = (contact) obj;
return Objects.equals(name,((contact) obj).name)
&&Objects.equals(email,((contact) obj).email)
&&Objects.equals(phoneNum,((contact) obj).phoneNum);
}
}
Contact System › src › ContactsPackage › contact › equals
32:18 CRLF UTF-8 4 spaces
Search
ENG
6:30 PM
2/25/2025


Contact System
Version control
Current File
Project
file.txt
ContactManager.java
Main.java
contact.java
UI.java
Contact System
.idea
out
src
ContactsPackage
contact
ContactManag
file.txt
Main
UI
.gitignore
Contact System.iml
External Libraries
Scratches and Consoles
package ContactsPackage;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Scanner;
import java.nio.file.*;
import java.io.*;
public class ContactManager { 2 usages
private ArrayList<contact>Contacts; 19 usages
private String File; 2 usages
//Validation codes for data and files
private String validNum(String num){ 2 usages
Scanner in =new Scanner(System.in) ;
String NumberRegex = "^(01\\d{9}|\\d{5})$";
while (!num.matches(NumberRegex)){
System.out.print("Please enter correct format of phone number : ");
num= in.nextLine();
}
return num;
}
Contact System › src › ContactsPackage › ContactManager
210:1 CRLF UTF-8 4 spaces
Search
ENG
6:31 PM
2/25/2025


Contact System
Version control
Current File
Project
file.txt
ContactManager.java
Main.java
contact.java
UI.java
Contact System
.idea
out
src
ContactsPackage
contact
ContactManag
file.txt
Main
UI
.gitignore
Contact System.iml
External Libraries
Scratches and Consoles
public class ContactManager { 2 usages
}
private String validName(String name){ 2 usages
Scanner in =new Scanner(System.in) ;
String NameRegex ="^[A-Za-z -]+$";
while (name.trim().isEmpty()||!name.matches(NameRegex)){
System.out.print("Please Enter a valid name : ");
name= in.nextLine();
}
return name;
}
private String validEmail(String email){ 1 usage
Scanner in = new Scanner(System.in);
String emailPattern = "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$";
while (!email.matches(emailPattern)){
System.out.print("Please enter correct email : ");
email= in.nextLine();
}
return email;
}
private boolean ValidationData(String txt , int stage){ 3 usages
Contact System › src › ContactsPackage › ContactManager
210:1 CRLF UTF-8 4 spaces
Search
ENG
6:31 PM
2/25/2025





Video Link:

https://www.youtube.com/watch?v=fuV9C_FRDf8

Low-Code/No-Code (LCNC) Tools: Evaluation, Potential, and Impact

Evaluation of LCNC Tools:

As for Thunkable:

Thunkable is a popular LCNC platform for building mobile applications with a visual drag-and-drop interface.

It allows users to create cross-platform iOS and Android apps without deep programming knowledge. While it excels in simplicity and rapid development, it has limitations in customization and performance for highly complex applications.

As for Webflow:

Webflow is a powerful LCNC tool for building responsive websites without writing code. It provides extensive design customization, CMS integration, and e-commerce capabilities.

However, for more advanced backend functionality, developers may still need to use custom code or external integrations.

The potential of LCNC Tools:

As for Thunkable:

Thunkable has great potential for facilitating mobile app development, particularly for non-developers and small businesses.

It enables rapid prototyping and deployment of functional mobile applications, making app development more accessible.

As for Webflow:

Webflow empowers designers and entrepreneurs to create sophisticated websites without relying on developers.

It is particularly useful for startups and businesses that require dynamic, responsive websites with CMS capabilities but lack coding expertise.

What LCNC Tools Can Do:

As for Thunkable:

- Create cross-platform mobile applications.
- Integrate APIs and external services.
- Implement UI/UX elements with a drag-and-drop editor.
- Directly publish apps to the Apple App Store and Google Play Store.

As for Webflow:

- Build highly customizable, responsive websites.
- Integrate CMS and e-commerce features.
- Use pre-built animations and interactions.
- Export clean HTML, CSS, and JavaScript for further customization.

Benefits of LCNC Tools:

As for Thunkable:

- Simplifies mobile app development.
- Reducing costs and time-to-market.
- Provides a user-friendly interface for non-technical users.
- Allows for quick prototyping and iteration.

As for Webflow:

- Enables high-quality web design without coding.
- Offers powerful customization and design flexibility.
- Integrates seamlessly with CMS and third-party tools.
- Generates clean and exportable code for further modifications.

The Quality of the Systems They Produce:

As for Thunkable:

Thunkable produces functional and user-friendly mobile applications suitable for MVPs, small business apps, and educational purposes.

However, apps developed with Thunkable may face performance and scalability challenges compared to those built with native development.

As for Webflow:

Webflow websites are visually stunning and optimized for performance. The platform ensures high-quality front-end development, but for complex backend functionalities, additional integrations or custom coding may be required.

Will LCNC Tools Take the Jobs of Developers?

As for Thunkable:

While Thunkable simplifies app development, it will not replace developers. Advanced mobile applications still require custom coding, optimization, and backend functionalities that LCNC tools cannot fully provide.

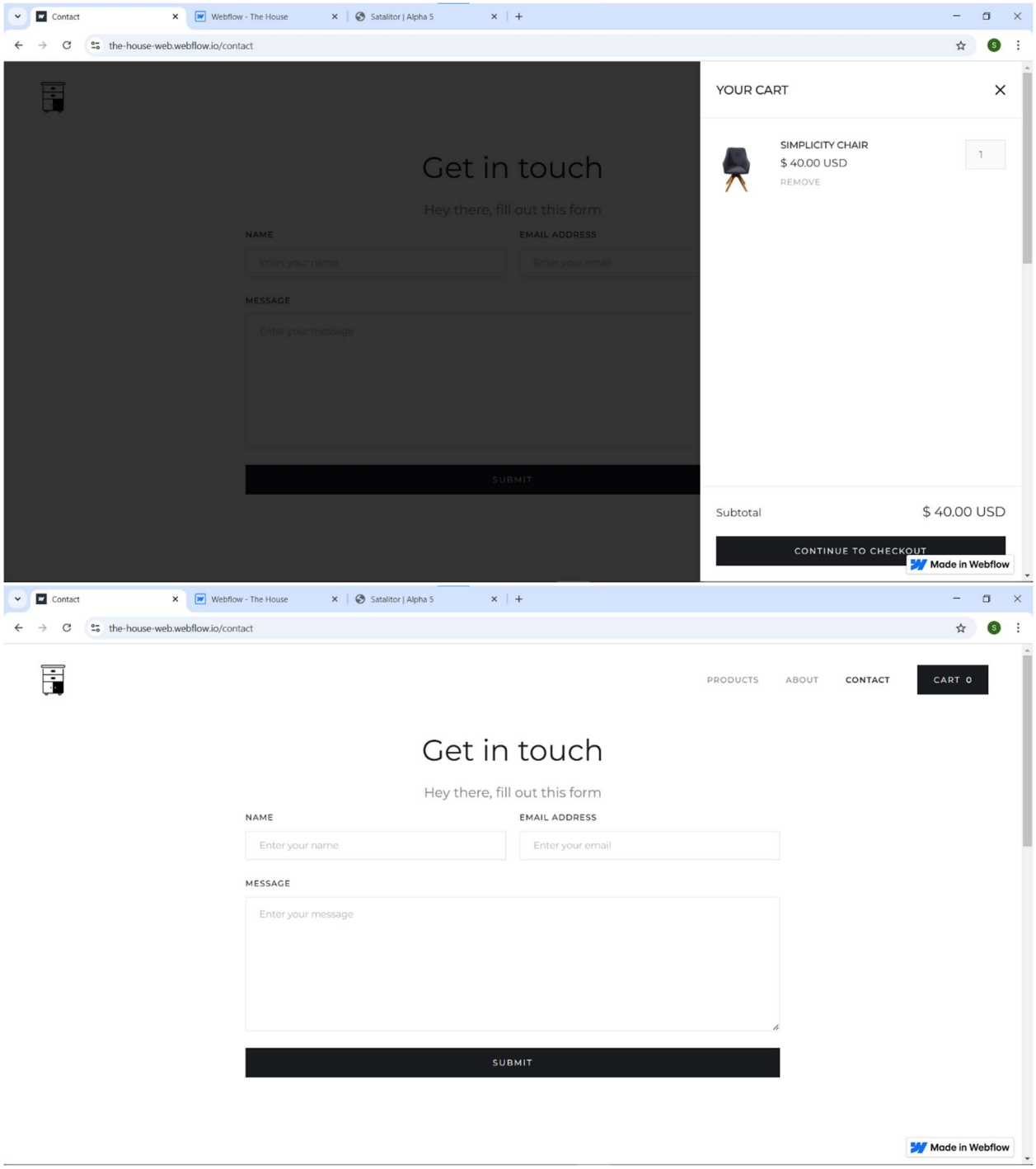
As for Webflow:

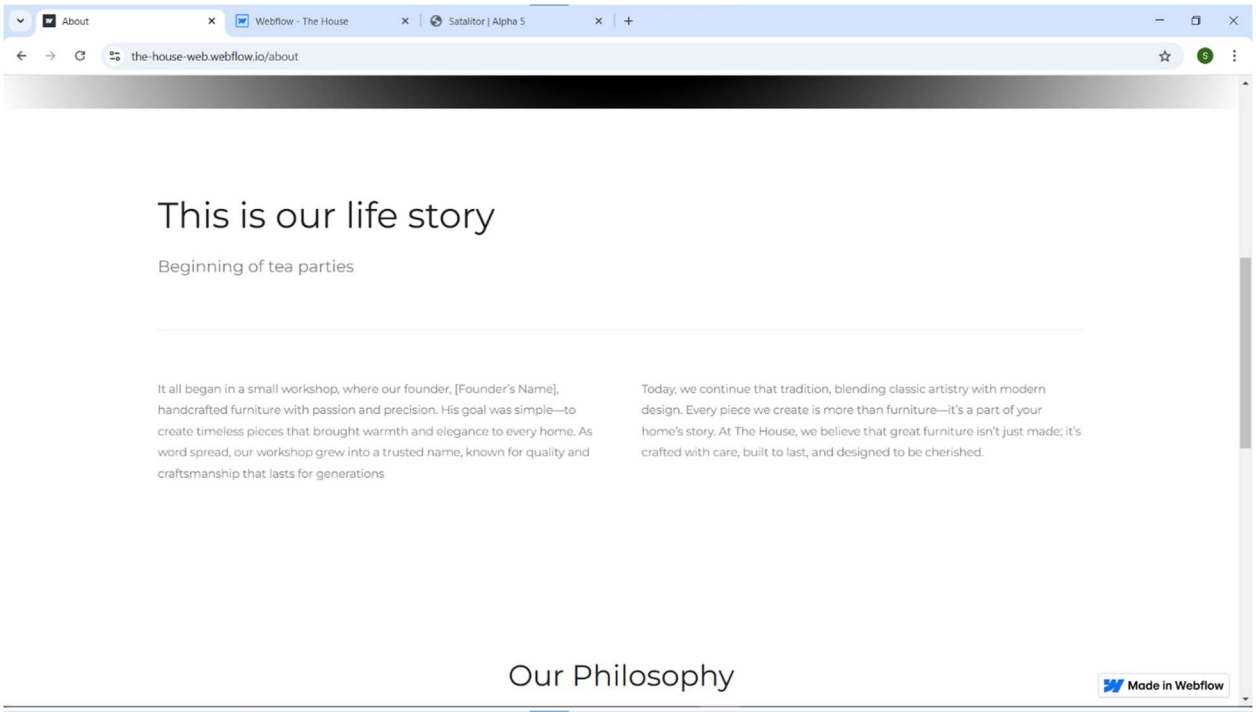
Webflow empowers designers and marketers to build and maintain websites without developers, but for highly dynamic and database-driven applications, developers are still essential. Webflow acts as a tool that enhances productivity rather than replacing professional developers.

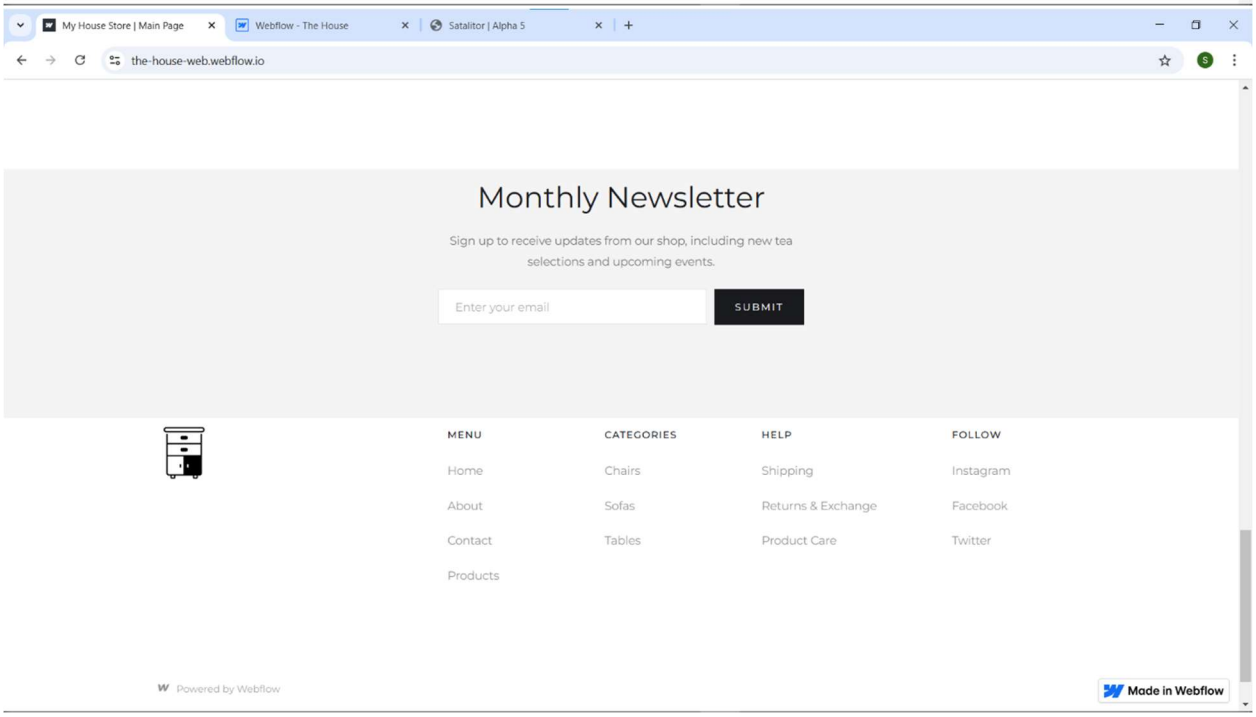
Comparison Between Two LCNC Tools:

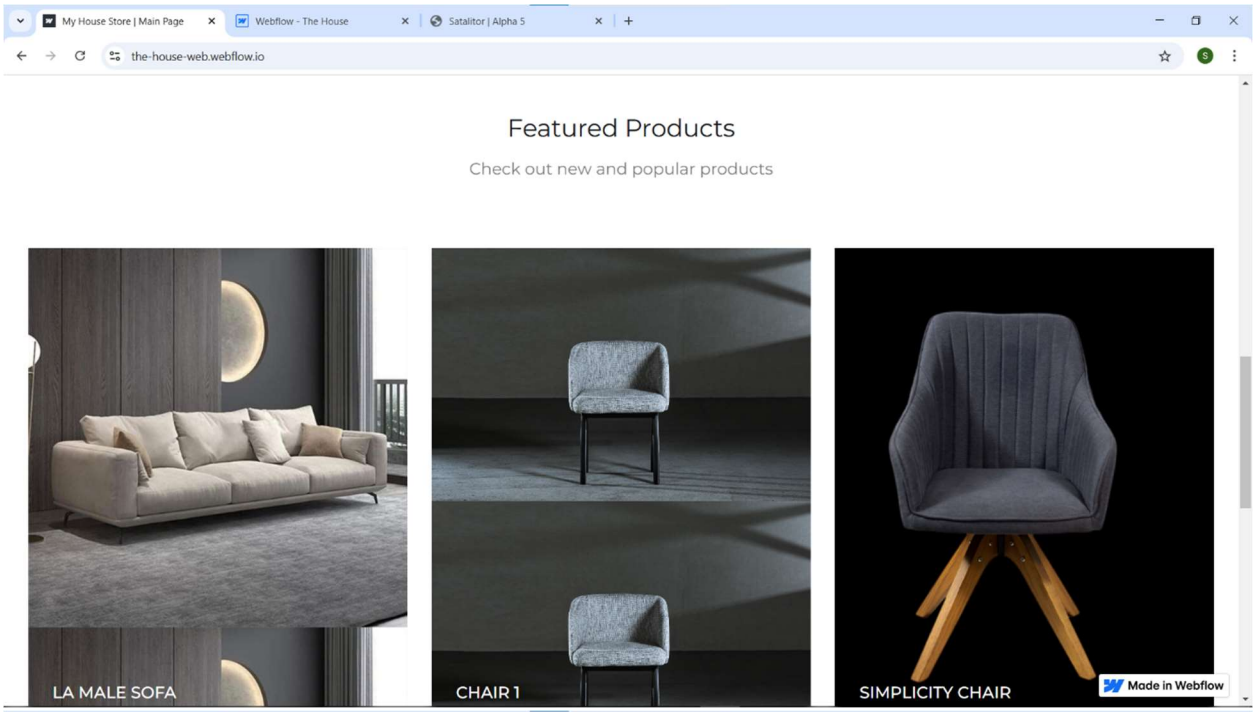
Key of Compare	WebFlow	Thunkable
Purpose	Website design and development	Mobile app development (iOS & Android)
Target Users	Designers, developers, and businesses looking to create responsive websites.	Individuals and businesses looking to create mobile apps without coding.
Drag-and-Drop Interface	Yes, for building websites visually	Yes, for building mobile apps using a block-based interface
Export Code	Can export HTML, CSS, and JavaScript	No direct code export; apps are built within Thunkable's environment
Use Cases	Landing pages, business websites, e-commerce stores, blogs, and portfolios	Prototyping, MVPs, mobile app development for business, education, and personal projects

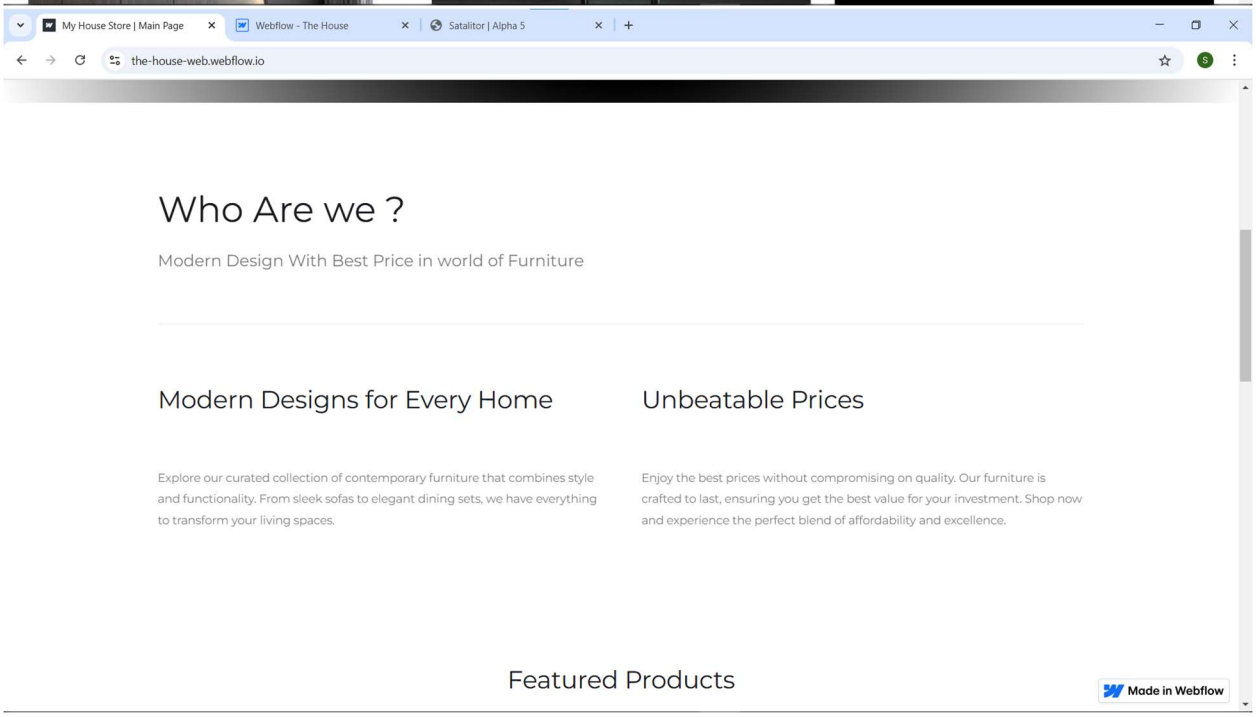
Screenshot of LCNC Code :

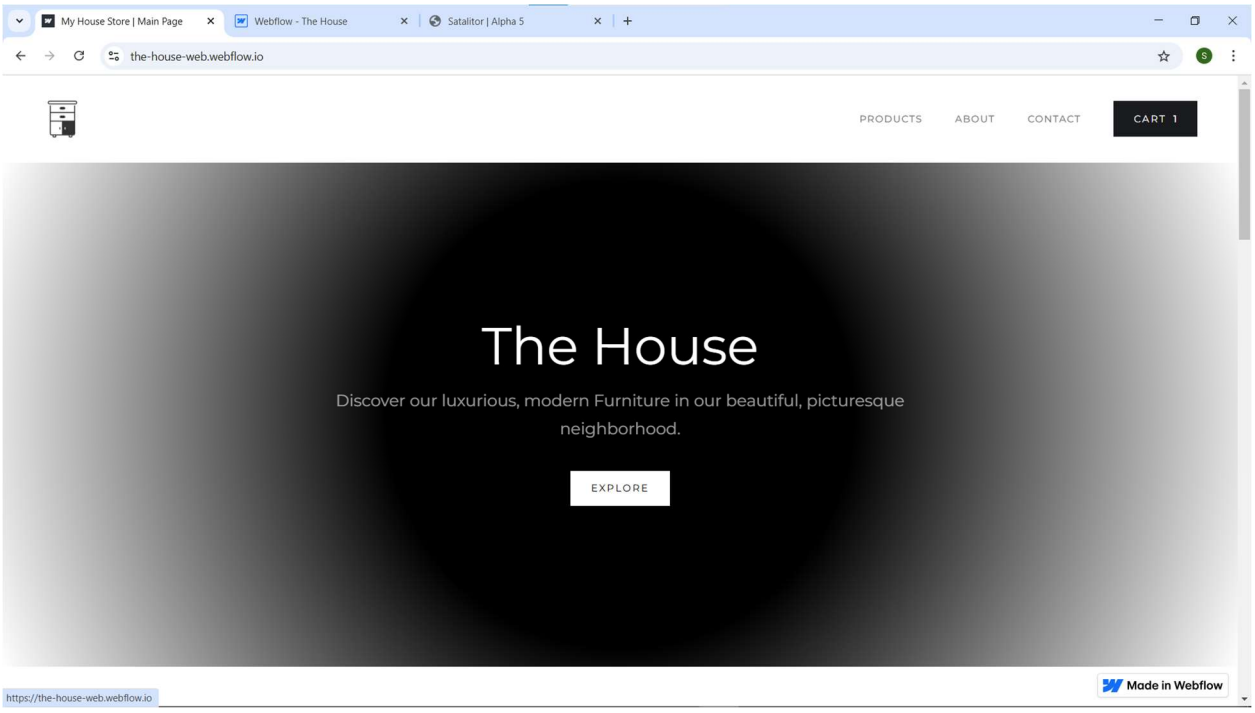



About
Webflow - The House
Satalitor | Alpha 5
the-house-web.webflow.io/about
This is our life story
Beginning of tea parties
It all began in a small workshop, where our founder, [Founder's Name], handcrafted furniture with passion and precision. His goal was simple—to create timeless pieces that brought warmth and elegance to every home. As word spread, our workshop grew into a trusted name, known for quality and craftsmanship that lasts for generations
Today, we continue that tradition, blending classic artistry with modern design. Every piece we create is more than furniture—it's a part of your home's story. At The House, we believe that great furniture isn't just made; it's crafted with care, built to last, and designed to be cherished.
Our Philosophy
Made in Webflow


My House Store | Main Page
Webflow - The House
Satalitor | Alpha 5
the-house-web.webflow.io
Monthly Newsletter
Sign up to receive updates from our shop, including new tea selections and upcoming events.
Enter your email
SUBMIT
MENU
Home
About
Contact
Products
CATEGORIES
Chairs
Sofas
Tables
HELP
Shipping
Returns & Exchange
Product Care
FOLLOW
Instagram
Facebook
Twitter
Powered by Webflow
Made in Webflow


My House Store | Main Page
Webflow - The House
Satalitor | Alpha 5
the-house-web.webflow.io
Featured Products
Check out new and popular products
LA MALE SOFA
CHAIR 1
SIMPLICITY CHAIR
Made in Webflow


My House Store | Main Page
Webflow - The House
Satalitor | Alpha 5
the-house-web.webflow.io
Who Are we ?
Modern Design With Best Price in world of Furniture
Modern Designs for Every Home
Explore our curated collection of contemporary furniture that combines style and functionality. From sleek sofas to elegant dining sets, we have everything to transform your living spaces.
Unbeatable Prices
Enjoy the best prices without compromising on quality. Our furniture is crafted to last, ensuring you get the best value for your investment. Shop now and experience the perfect blend of affordability and excellence.
Featured Products
Made in Webflow



Website link: https://the-house-web.webflow.io/products

Video link:

https://youtu.be/c071WQAifNM?si=sWq_lknsMG3GbJbN

- Pre-Project Activities:

Project Type: Personal Investment Management Software.

1. Market and Gap Analysis:

In Global Practices:

Personal Investment management software is used worldwide, but it is most used in the United States and the United Kingdom. It has a lot of benefits that can help in your evolution, like:

Portfolio tracking consolidates financial data across various assets like stocks, real estate, and savings. This provides users with a clear overview of their investments.

Income tracking monitors revenue streams from different sources, such as salaries, dividends, and rental income. It helps users manage cash flow effectively.

Net worth calculation and ROI analysis compute overall financial standing by assessing assets and liabilities. This helps users track their wealth growth.

Visualization and graphs present financial data through interactive charts. These tools simplify performance analysis and trend identification.

Target setting allows users to establish financial goals, such as savings milestones or investment targets. It helps maintain a structured financial plan.

Risk and asset allocation analyzes portfolio risks and suggests diversification strategies. This ensures a balanced investment approach.

Bank and brokerage integration syncs financial institutions with the app. This enables real-time tracking of transactions and stock prices.

Not only that, but some of the international software are providing comprehensive financial solutions that can help the user and give the customer a better experience that help him in his problems, the solutions is not all the same each app has his own feature that categorize it from other apps

Providing detailed tracking of investments and retirement accounts. also by giving users insights into their financial health by analyzing assets, liabilities, and net worth, and one of the apps that is the best in this is Personal Capital.

Integrating budgeting with investment tracking, at the same time allowing users to set financial goals and track their progress effectively like what Mint does.

Focusing on automated investment management. helping in optimizing taxes, providing goal-based investing, and managing retirement planning, as one of the programs that do it is Wealthfront.

Specializing in commercial real estate investments, giving investors an easy way to track and manage their real estate portfolio like RealtyMogual.

Supporting a diverse range of assets, including stocks, bank accounts, cryptocurrencies, and real estate, makes it a versatile solution for investors like Kubera.

And a lot of other programs that have a lot of good features.

In Egypt's Market:

Personal investment management software is gradually expanding in Egypt, though it is primarily focused on budgeting and stock trading rather than comprehensive wealth management. While global solutions dominate the market, Egyptian apps are emerging to meet local financial needs. These apps help users track their finances, but they often lack the full range of features found in international alternatives.

Stock trading platforms like Thndr provide users with access to the Egyptian stock market, allowing them to invest in and monitor their portfolios. However, they do not integrate real estate, crypto, or net worth tracking.

Budgeting tools such as Wafeer focus on expense management and savings tracking. These apps help users manage their day-to-day spending but do not offer full-fledged investment insights.

Financial services apps like FawryPay and ValU are primarily used for digital payments and consumer credit. While they offer convenience, they lack features for tracking diversified investment portfolios.

Brokerage platforms like Beltone Invest provide stock trading and investment services but do not consolidate different asset classes into a single dashboard.

Microfinancing solutions such as Halan cater to financial inclusion by offering small loans and credit services. However, they do not support wealth accumulation or investment tracking.

And it comes better day after day and not stop of there evolution but there is still a lack gap between the international programs and the Egyptian ones.

Gap analysis:

Egypt's investment management sector is growing, but it still lags behind international standards. While global apps offer comprehensive wealth management, Egyptian platforms mostly focus on budgeting and stock trading as:

Limited portfolio tracking prevents users from consolidating all their assets, such as stocks, real estate, and crypto, into a single dashboard. Weak investment analysis and ROI calculation mean most local apps prioritize transaction tracking over wealth growth insights.

Moreover, no Egyptian platform integrates multiple asset classes, such as brokerage accounts, real estate, and banking transactions, into one seamless experience. Minimal goal-setting and planning tools leave users without structured financial forecasting or retirement strategies. Underdeveloped visualization tools make it difficult to analyze financial trends effectively. Additionally, cryptocurrency investment tracking is almost nonexistent in local apps.

To close these gaps, an advanced Egyptian investment management app should offer full-scale asset tracking, AI-powered insights, financial goal-setting, and seamless integration with banks and brokerage services. By adopting these features, the market can move toward global standards, empowering users with smarter financial decisions.

2. Global Market Analysis :

Customer Segments

There are four major target audiences for personal investment management software around the globe:

1. **Millennials (20-30 years old)**
    - Familiar with mobile applications and other fintech solutions.
    - Most probably early adopters of AI powered financial planning tools.
    - They seek investment information and saving towards a financial goal functionalities.
2. **Self-Directed Investors & Retail Traders**
    - Business owners who also self-manage their stocks, cryptocurrency, or real estate investments.
    - They require high level forecasting, risk evaluation, advanced analytics, and automation.
3. **Working & Middle Class Income Families**
    - Households looking for budgeting, saving, and investment solutions and probably have kids they spend money on.
    - They need family spending management, possible debt support, and comprehensive financial management.
4. **Financially Excluded & Underbanked Populations**
    - Large unbanked communities worldwide need digital solutions to access financial services.
    - Software investment platforms can provide financial inclusion.

Demographics

- **Age:** 20-45 years old.
- **Income Level:** Middle and upper-middle income; people looking to grow wealth or manage expenses.
- **Education:** Varies, Educated or financially naive but wanting to learn.
- **Tech Usage Patterns:**
    - Use smartphones a lot.
    - Prefer digital banking & fintech over traditional banks.

- Comfortable with and trust AI-driven solutions.

Market Trends & Opportunities

- **Financial Inclusion:** a trend in fintech apps that target underbanked populations.
- **Digital-Only Finance Growth:** digital banks and mobile wallets witness rapid growth.
- **AI & Automation:** Personalized service and AI-powered solutions and strategies.
- **Regulatory Support:** Governments promoting digital payment and financial inclusion for those who are financially excluded.

Egypt Market Analysis

Market Landscape & Challenges

- **67% of the population is unbanked, operating only with cash transactions**.
- **The mobile penetration rate is high (110%) but the traditional banking sector is not implementing digitalization fast enough.**
- **12 banks have introduced mobile wallets**, which is evidence of a movement towards digital finance..
- **Government regulations encourage financial inclusion**, but legacy banks are slow to adapt.
- **Existing finance apps are not able to provide actionable features, like budgeting or expense tracking**.

Customer Segments in Egypt

1. **Unbanked & Underbanked Individuals**
    - Large segment unbanked and have no access to traditional banking services.
    - They need mobile-first solutions for saving, budgeting, and digital transactions.
2. **Young Professionals & Students (18-35 years old)**
    - **65% of Egypt's population is under 35**, which makes them the most potential category to benefit from FinTechs.
    - Interested in financial planning, investment, and digital banking.
3. **Small Business Owners & Freelancers**
    - Lack of access to traditional corporate banking services is the main obstacle for emerging businesses when it comes to their financial management needs.

- Prefer mobile apps for invoicing, expense tracking, and digital payments instead of traditional programs.

4. **Middle-Class Households**
    - They are seeking broader management tools that can help them in monitoring expenses, setting up saving plans, and tracking debts.
    - Need to be educated about personal finance support.

Demographics in Egypt

- **Age:** 18-40 years old.
- **Income Level:** Low to middle-income, with a growing sense of the importance of good money management.
- **Education Level:** Varies; but most of them require financial education.
- **Tech Usage Patterns:**
    - Intense mobile users; many of them use a variety of apps.
    - Growing confidince in fintech and digital wallets.

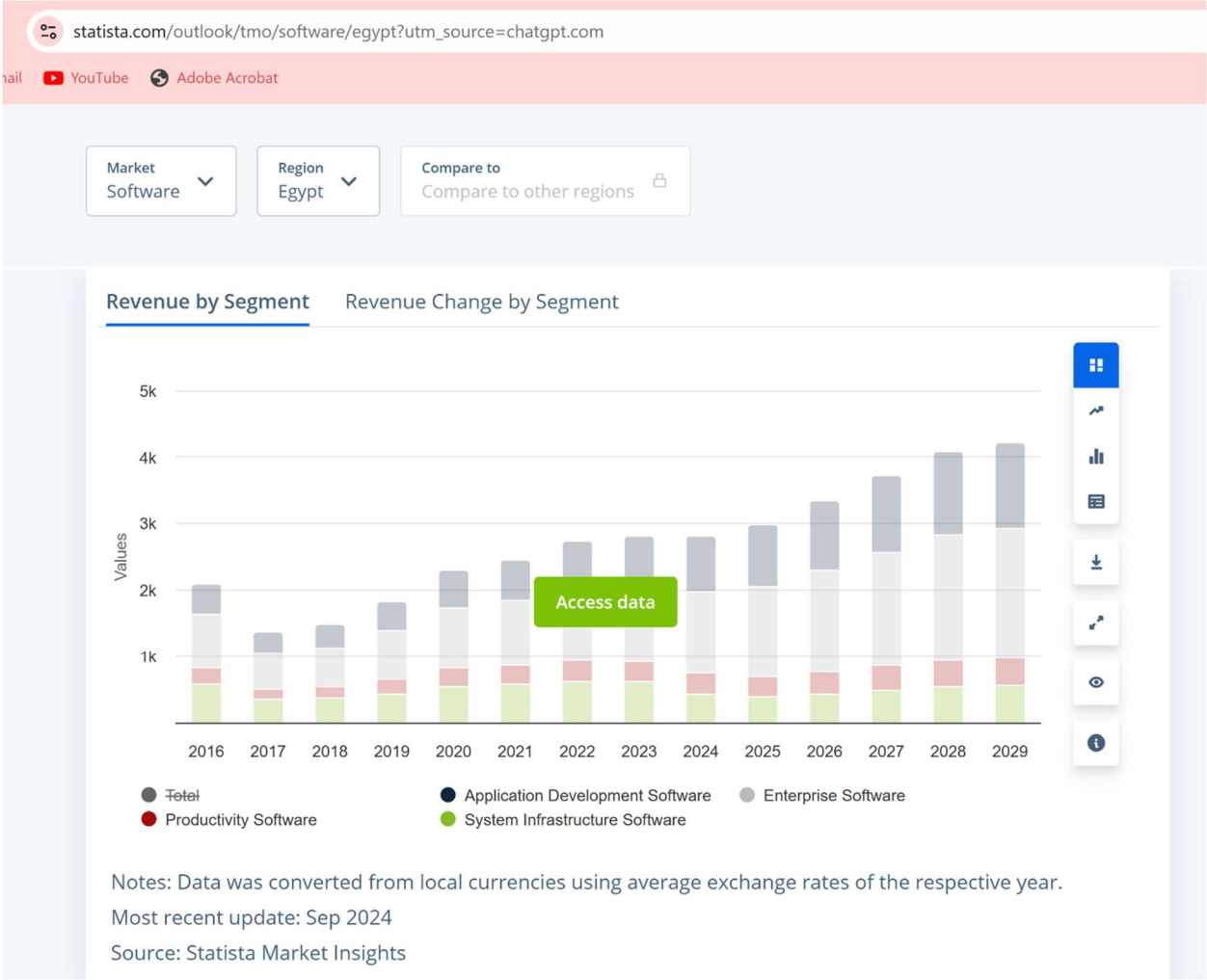
Market Performance & Competition

- **Top Personal Finance Apps in Egypt (Q2 2023 Data from Sensor Tower):**
    - *Cash Book – Daily Expenses* (Steady user growth, peak downloads at 2K/week).
    - *CashKateb (Cassbana B.V.)* (Largest active user base ~27K).
    - *المصاريف (Al-Masareef App)* (Strong revenue performance, peak $71/week).
    - *Income Expense* (Steady downloads, growing user base).
    - *SajalAlhisabat* (Fluctuating downloads, peak at 4.4K/week).

Opportunities for a New Solution in Egypt

1. **Mobile-First & AI-Driven Personal Finance Management**
    - Digital-only finance apps seem more profitable than unmoving traditional banks.
2. **Financial Literacy Features**
    - World Bank research (2014) has revealed that only 27% of adult people in Egypt are financially literate.
    - An app that gives users information on investing and budgeting through educational content (investment guides, budgeting tips) could be the factor that attracts users.

3. **Smart Budgeting & Investment Tools**
    - Most existing finance apps in Egypt lack actionable insights.
    - Opportunity to integrate AI-driven expense tracking, investment suggestions, and saving automation is a possible way to bring fintech tools to the next level.
4. **Government Support & Market Growth**
    - Egypt's **software market to reach $498.36M by 2025**, with **CAGR of 8.93% (2025-2029)**.
    - Fintech startups can grow with the support for digital financial services.



Conclusion: Why Would These Groups Use This Solution?

- **Young professionals and students** → Essentially, they need technology-first, user-friendly financial platforms.
- **Unbanked & underbanked users** → They do not have access to traditional banking; however, mobile wallets may be an entry point.

- **Freelancers & small businesses** → They require financial management tools without the banks involved.
- **Middle-class households** → Require budgeting and expense tracking to manage financial goals.

Final Insights

- **Egypt has young, tech-interested population that is ready for fintech solutions.**
- **Traditional banks are slow to digitize, creating space for mobile-first apps.**
- **Existing apps lack deep insights and educational features, leaving a market gap.**
- **Government support for digital payments accelerates fintech adoption.**

3. Domain Analysis :

1. Introduction :

This document gives a brief about the personal investment and how it goes . This information will help in developing the application and how it will handles the problems that exists in real life .

2. Glossary :

Stocks : These are shares or ownership in a company . When you buy stocks , you are purchasing a small piece of the company.

Real Estate : Investing in real estate typically involves buying property with the hope that its value will increase over time, or it can generate rental income.

Crypto (Cryptocurrency): This refers to digital currencies like Bitcoin,

Ethereum, or other forms of virtual money.

Risk assessments : refers to the process of evaluating and analyzing the potential risks associated with an investment or portfolio.

3. General Knowledge about domain :

1. Tracking income weather comes from crypto , lands , real estate or stocks all of them are being tracked through papers and some calculations are done manually .
2. Saving a certain percent from the revenue of the cryptocurrency , real estate or whatever the source of the money.
3. Spending and investing in the correct path with the correct people who are expert in the certain domain which you invest in .
4. Targeting and putting a certain goal to achieve a certain financial target .
5. Defining your potential risks that you could face in that investment .4
6. Monitoring the market and trends and the market goes in which direction which decreases the potential risks .
7. Discussing with experts and investors which made a great success in that field for taking some important and significant tips .

4. **Customers and users :**

It will be used by the people who are interested in that field as :

1. **Managers** of companies especially who invest in a diversity of source of income and revenue .
2. **Financial Advisors and Planners ,** some financial professionals use these apps to track their clients' portfolios and help them with asset management and risk assessment.
3. **Families** also for multi-purposes as financial education , plan for major life events or manage risks .
4. **Unbanked & underbanked individuals** , Those who rely on cash transactions and mobile wallets but need financial literacy support.
5. **People approaching retirement (35-60) –**people saving money to cover their needs after retirement.

5. **Tasks and procedures :**

- **Portfolio Tracking:** Monitors different assets like stocks, real estate, and savings.
- **Income Tracking:** Track the income streams generated from different investments.
- **Computation of Net-Worth and Rate of Investment (ROI).**
- **Visualization & Graphs:** Displays investment performance through charts and graphs for easy analysis and comparison with historical data.
- **Target Setting:** Allows users to set income goals, wealth targets, and retirement objectives**.**
- **Risk &Asset Allocation:** Analyzes risk and balances investments across asset types to diversify.
- **Integration:** Syncs with bank accounts, stock prices, brokerage accounts, etc.
- **Suggesting ways** to increase the scale of the investment .
- **You can have connections with experts and investors** which could help you to take tips or enlarge your investment .
- **Retirement planning ,** helps users to plan for their retirement savings .
- **Real-Time Stock & Crypto Price Tracking**: Fetches live financial data for better decision-making.
- **Managing the saving process** from the various incomes that are generated from different investments through specifying the desired percentage of the income , this functionality helps the paying debts in case there are debts should be paid .

- **Educational activities and coaching ,** to learn how to invest professionally and enlarge your investments with minimizing risks.

6. Environment :

The software will be available as a mobile app to ensure easy access for users. The system will integrate with third-party financial APIs. The system should be capable of handling large datasets, multiple users, and real-time data updates efficiently.

7. Competing Software :

There are several software products manage the personal investment . but that software app offers integration of functionalities that other software products also offers but not integrated as (Fawry , mint , personal capital ,etc). Also , the added values that the app could suggests for you ways to invest and connect you with investors and experts to take tips from them . Moreover, the educational activities that guide you to invest professionally.

8. Similarities to other domains :

Before technology, managing investments required meticulous record-keeping, personal oversight, strong relationships, and extensive knowledge of local and global markets. It was a highly manual , but of course for that manual processes issues had arisen . then that software apps appeared as (Fawry , mint , Wealthfront , etc.) which provide users with important various points as tracking diverse assets including bank accounts, stocks, and real estate , tracking investments and retirement accounts with performance insights and automating retirement planning. That functionalities are spread in different software systems . Our software product combines that functionalities , furthermore provides management of saving and payment of debts , educational activities and communication with experts and investors

4. Purpose and Goals

Our Personal Investment Management Software is designed to equip individuals to keep a fully-featured and user-friendly application for the control, management, and performance of their investments. It seeks to simplify wealth management and add financial literacy-which will, in turn, allow investment theories for beginners and experienced investors to.

Key Objectives:
- Get users to easily control and manage their investments in the best way for them, including stocks, real estate, bank accounts, and many others.

- Improve financial literacy by interactive educational modules, which have been users in making wise choices.

- Show the users ways of meeting and realizing financial objectives successfully, like wealth accumulation, retirement planning and debt management.

- Provide further insights including visualization tools, risk analysis, and AI-driven financial coaching.

- Integrate finance seamlessly through synchronization with banks, stock exchanges, and brokerage accounts.

---

2. Key Features and Functionality

Financial literacy & insights for users with little to no knowledge about finances

- Interactive Learning Modules: Budgeting, saving, investing, and wealth-building strategies
- Gamified Quizzes & Challenges: Fun activities for cementing financial literacy.
- AI-Powered Financial Coaching: It generates tailored money management recommendations based on how the user behaves.
- Investment Growth Strategies: Recommendations on how to scale investments effectively

Target Setting & Smart Saving Systems

- Income & Wealth Goals: Allow users to set financial targets for income, savings, and retirement.
- Automated Saving Plans: Saves a user-specified percentage of income monthly.
- Retirement Planning: Helps users strategically plan their retirement savings.

Debt Management & Automated Payments

- Auto Debt Repayment: Automatically allocates a percentage of income to pay off debts.

Portfolio & Investment Tracking

- Multi-Asset Portfolio Tracking: Monitors stocks, real estate, crypto, and savings in one dashboard.
- Income Tracking: Tracks revenue from various investment streams.
- Net Worth & ROI Calculation: Computes the user's total wealth and return on investment (ROI).
- Risk & Asset Allocation Analysis: Assesses risk levels and suggests diversified asset allocation.
- Advanced Visualization & Graphs: Provides easy-to-understand charts for performance comparison.

Seamless Financial Integration

- Bank & Brokerage Sync: Connects with bank accounts, stock markets, and trading platforms.
- Real-Time Stock & Crypto Price Tracking: Fetches live financial data for better decision-making.

---

3. Target Users

Individuals & Households

- Young professionals & students (18-35 years old) – These are the first people using the internet to search for financial literacy and investment tools..
- Middle-class families – People managing household expenses and long-term savings paying their bills and saving for the future.
- People approaching retirement (35-60) –people saving money to cover their needs after retirement.

Investors & Entrepreneurs

- Small business owners & freelancers – Need efficient income tracking and investment strategies.
- Financial advisors & planners – Professionals who monitor clients' portfolios and provide data-driven insights.

Financially Underserved Users

- Unbanked & underbanked individuals – Those who rely on cash transactions and mobile wallets but need financial literacy support.

---

4. Technologies

- Java for mobile development (Android), with a possible React/Flutter expansion for cross-platform support.
- MySQL or Microsoft SQL Server for structured investment and financial data storage.
- Java Spring Boot for API handling.

Resources:

Fatema El-Zhraa Resources:

https://www.investopedia.com/terms/p/personalfinance.asp

https://www.youtube.com/watch?v=N5Lgv59Slj8

https://www.youtube.com/watch?v=R4gSwTtBano

https://youtu.be/OEx0ACeemHo?si=zqm5oxeU6rIyHb7c

Aly El-Deen Resources:

https://www.kubera.com/blog/personal-capital-vs-mint-vs-kubera

+ the Software Programs website.

Nagham Resources:

https://www.businessresearchinsights.com/market-reports/investment-portfolio-management-software-market-101260

https://www.grandviewresearch.com/industry-analysis/personal-finance-software-market-report

https://thedocs.worldbank.org/en/doc/b7a35868206ace761909f9bd2daa1f91-0200022021/original/Digital-Economy-Country-Assessment-May-26-Final.pdf

https://www.day1tech.com/wp-content/uploads/2022/08/Monie-Case-Study.pdf

https://www.statista.com/outlook/tmo/software/egypt