Introduction to Software Engineering

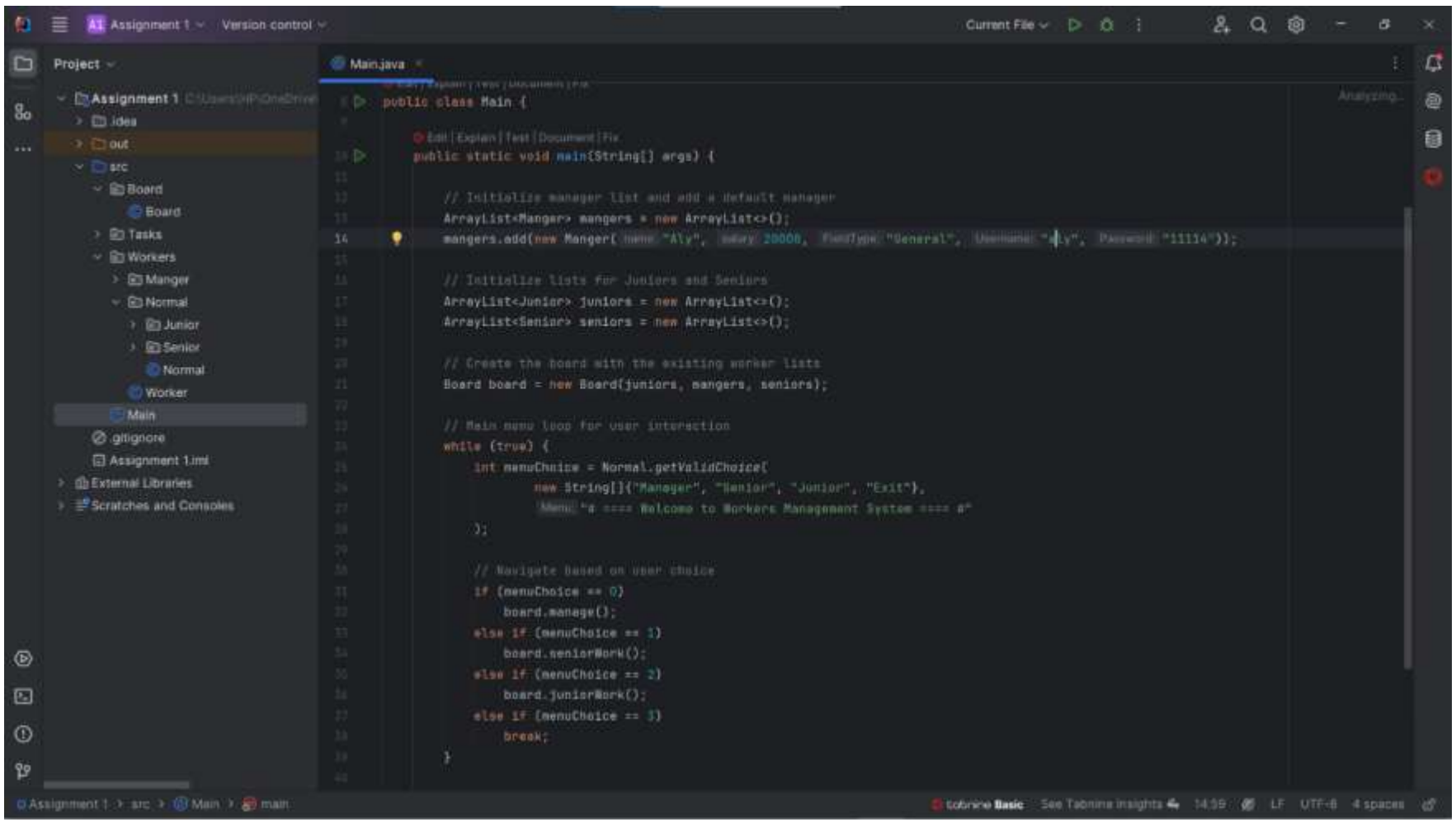
Assignment 1

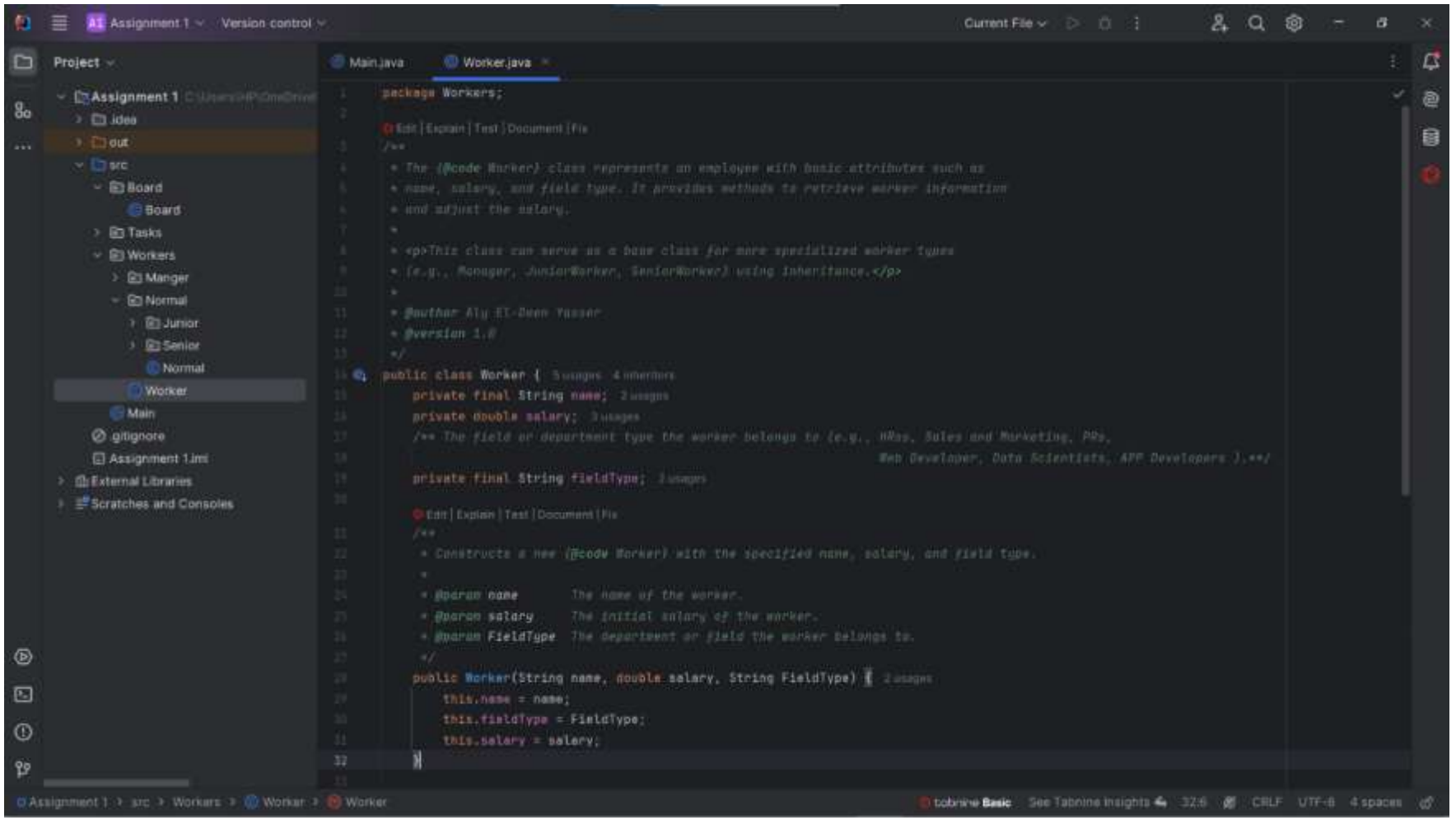
Nagham Wael Mohamed	20231189
Fatema El-Zhraa Ahmed Mohamed El-Fiky	20230280
Aly El-Deen Yasser Ali	20231109

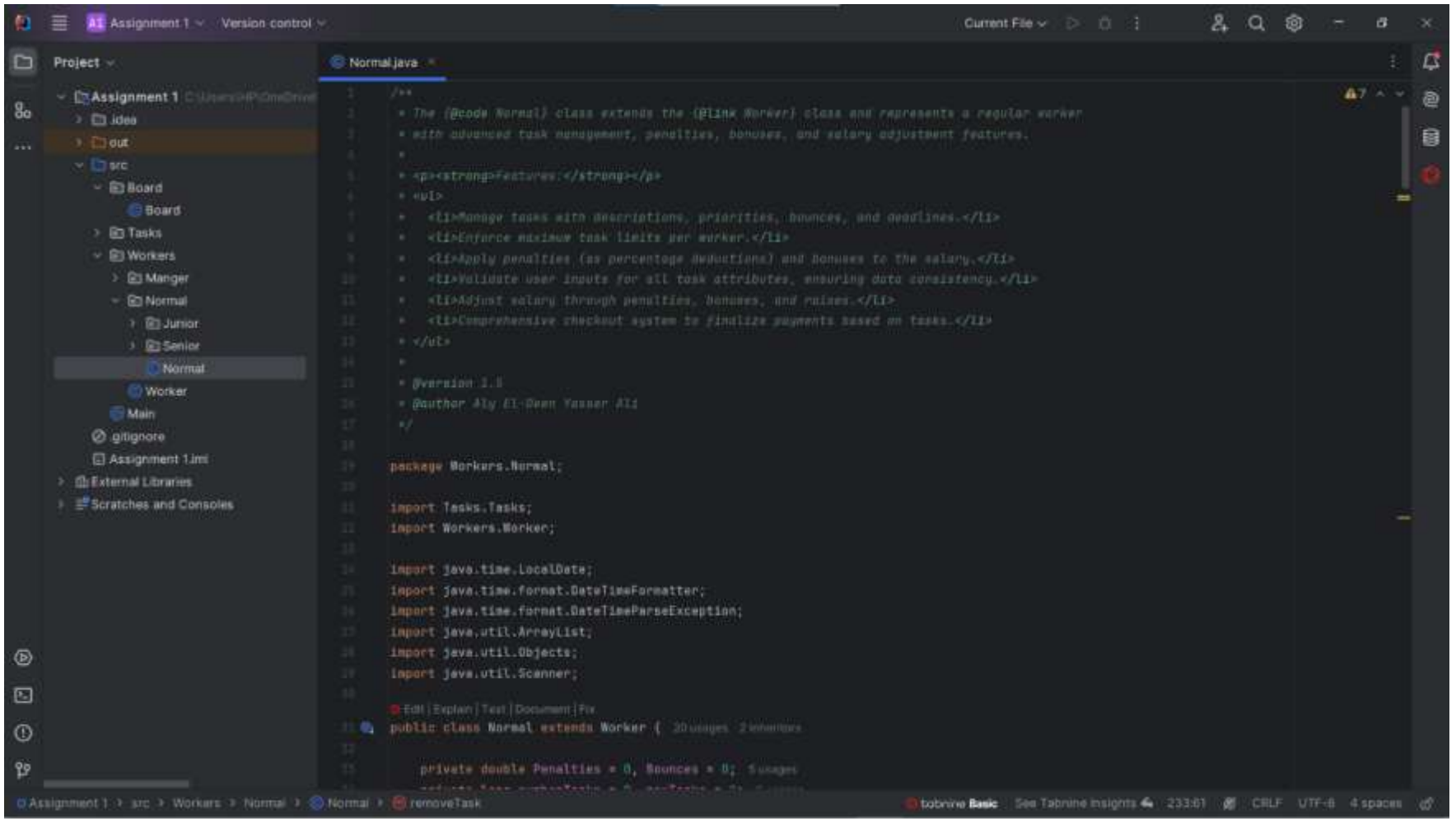
Java Programs

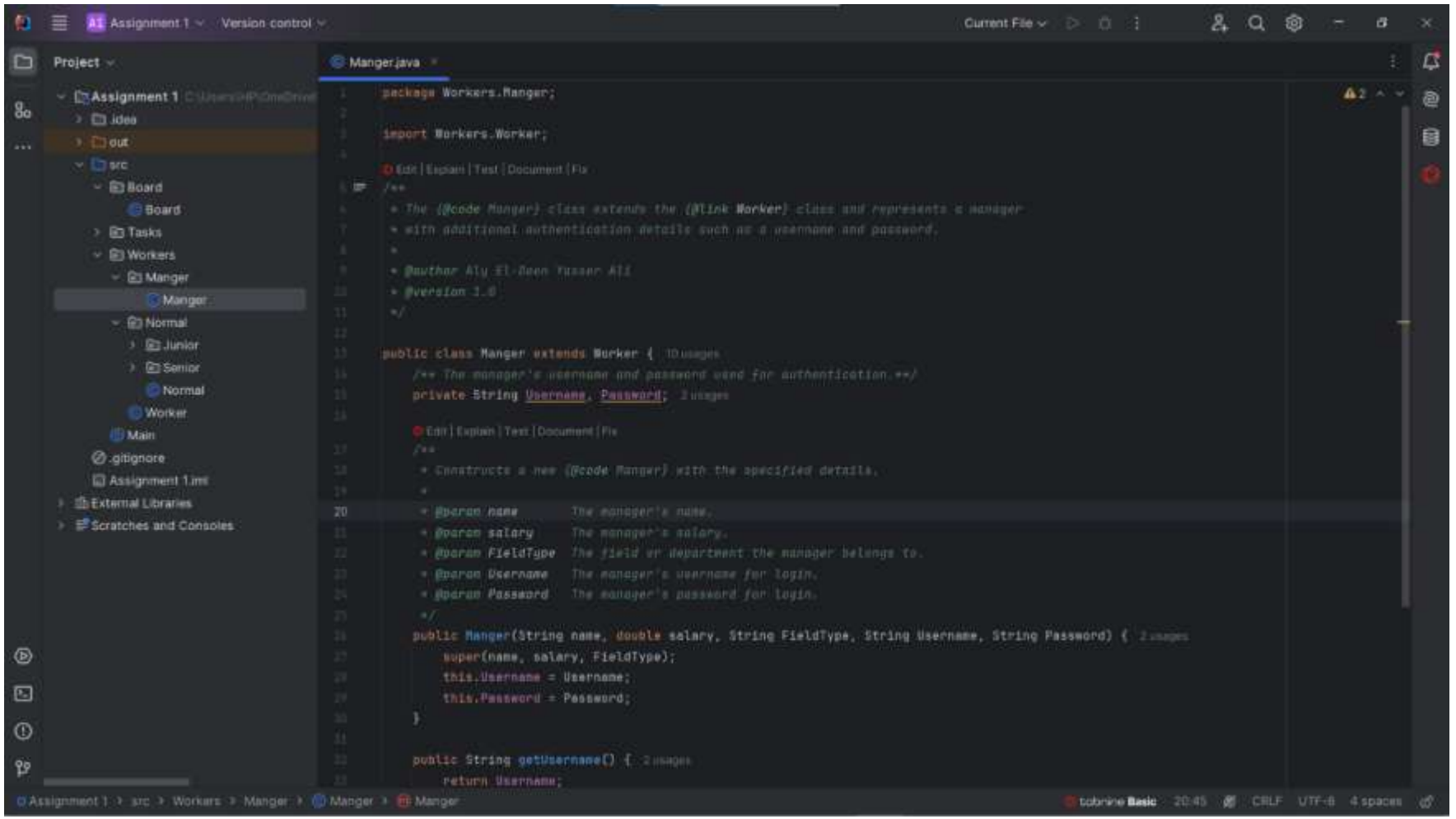
Aly El-Deen Yasser Aly:

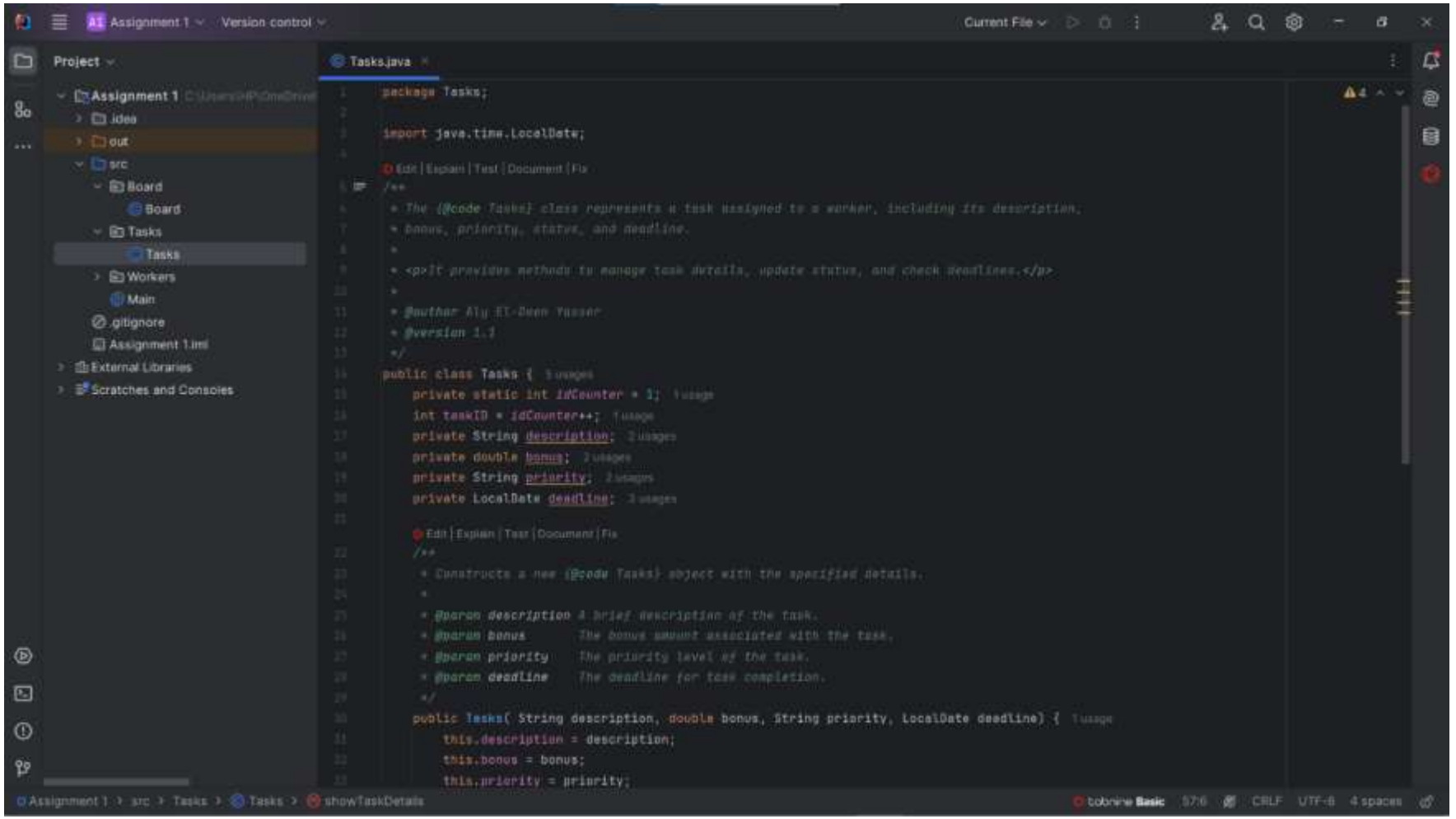
- Hours of Study: 18 hours
- Source of study https://youtube.com/playlist?list=PLCInYL3l2AajYlZGzU_LVrHdoouf8W6ZN&si=tfRlF2iNxOtJehum
- Video Link: https://youtu.be/-brwmS8jSZo

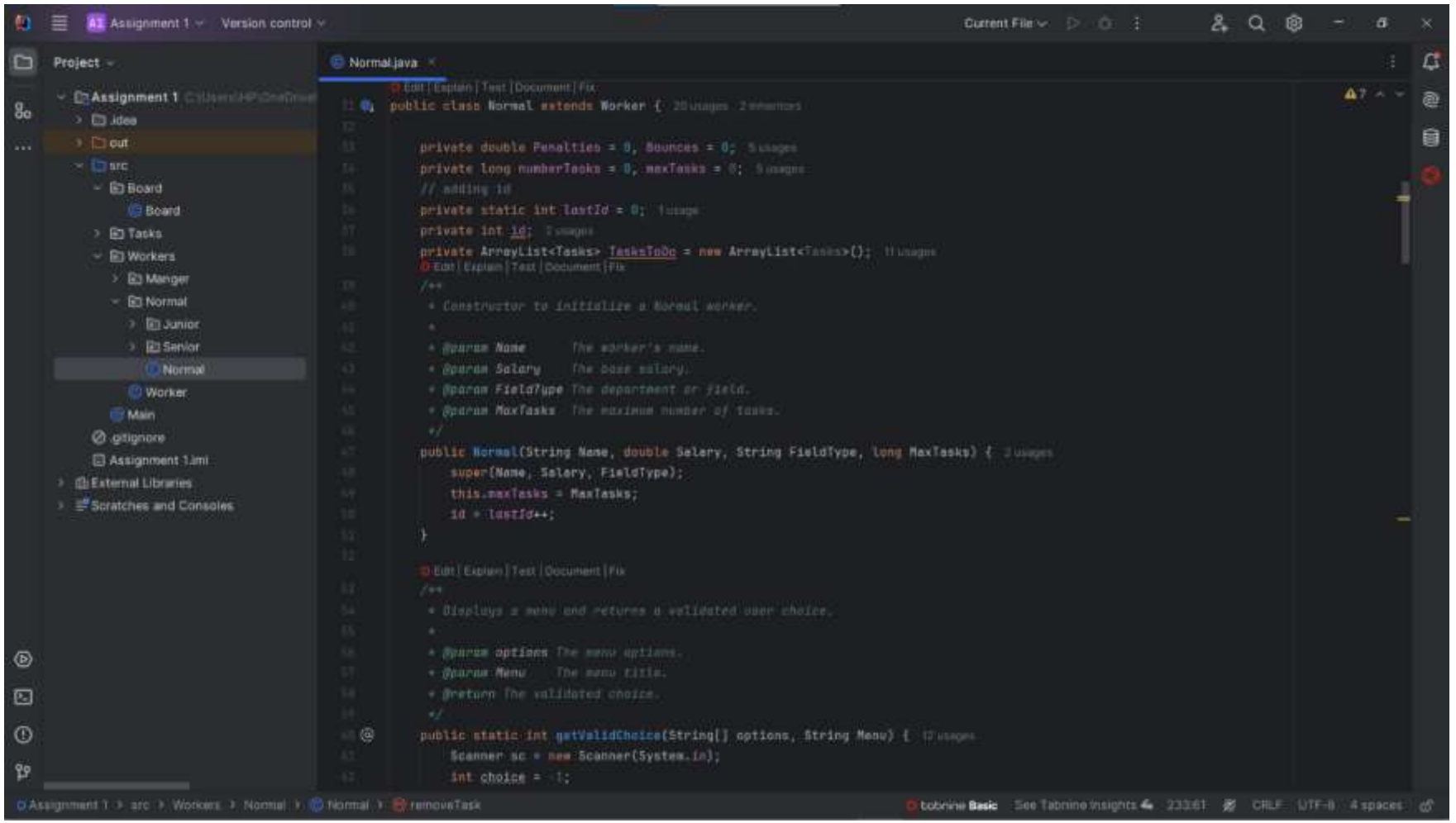

Assignment 1
Version control
Current File
Project
Main.java
Assignment 1
idea
out
src
Board
Board
Tasks
Workers
Manger
Normal
Junior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
public class Main {
public static void main(String[] args) {
// Initialize manager list and add a default manager
ArrayList<Manger> mangers = new ArrayList<>();
mangers.add(new Manger(name: "Aly", salary: 20000, FieldType: "General", Username: "Aly", Password: "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"},
Menu: "\n ==== Welcome to Workers Management System ==== \n"
);
// 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;
}
Assignment 1 > src > Main > main
Tabnine Basic
See Tabnine Insights
14:59
LF
UTF-8
4 spaces


Assignment 1
Version control
Current File
Project
Main.java
Worker.java
Assignment 1
idea
out
src
Board
Board
Tasks
Workers
Manger
Normal
Junior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
package Workers;
Edit | Explain | Test | Document | Fix
/**
* The {@code Worker} class represents an employee with basic attributes such as
* name, salary, and field type. It provides methods to retrieve worker information
* and adjust the salary.
*
* This class can serve as a base class for more specialized worker types
* (e.g., Manager, JuniorWorker, SeniorWorker) using inheritance.
*
* @author Aly El-Deen Yasser
* @version 1.0
*/
public class Worker {
private final String name;
private double salary;
/** The field or department type the worker belongs to (e.g., HRs, Sales and Marketing, PMs,
Web Developer, Data Scientists, APP Developers).**/
private final String fieldType;
Edit | Explain | Test | Document | Fix
/**
* Constructs a new {@code Worker} with the specified name, salary, and field type.
*
* @param name The name of the worker.
* @param salary The initial salary of the worker.
* @param FieldType The department or field the worker belongs to.
*/
public Worker(String name, double salary, String FieldType) {
this.name = name;
this.fieldType = FieldType;
this.salary = salary;
}
Assignment 1 › src › Workers › Worker › Worker
tabnine Basic
See Tabnine Insights
32:6
CRLF
UTF-8
4 spaces


Assignment 1
Version control
Current File
Project
Normal.java
Assignment 1
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:
* <ul>
* <li>Manage tasks with descriptions, priorities, bounces, and deadlines.</li>
* <li>Enforce maximum task limits per worker.</li>
* <li>Apply penalties (as percentage deductions) and bonuses to the salary.</li>
* <li>Validate user inputs for all task attributes, ensuring data consistency.</li>
* <li>Adjust salary through penalties, bonuses, and raises.</li>
* <li>Comprehensive checkout system to finalize payments based on tasks.</li>
* </ul>
*
* @version 1.0
* @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 {
private double Penalties = 0, Bounces = 0;
Assignment 1 › src › Workers › Normal › Normal › removeTask
CRLF
UTF-8
4 spaces

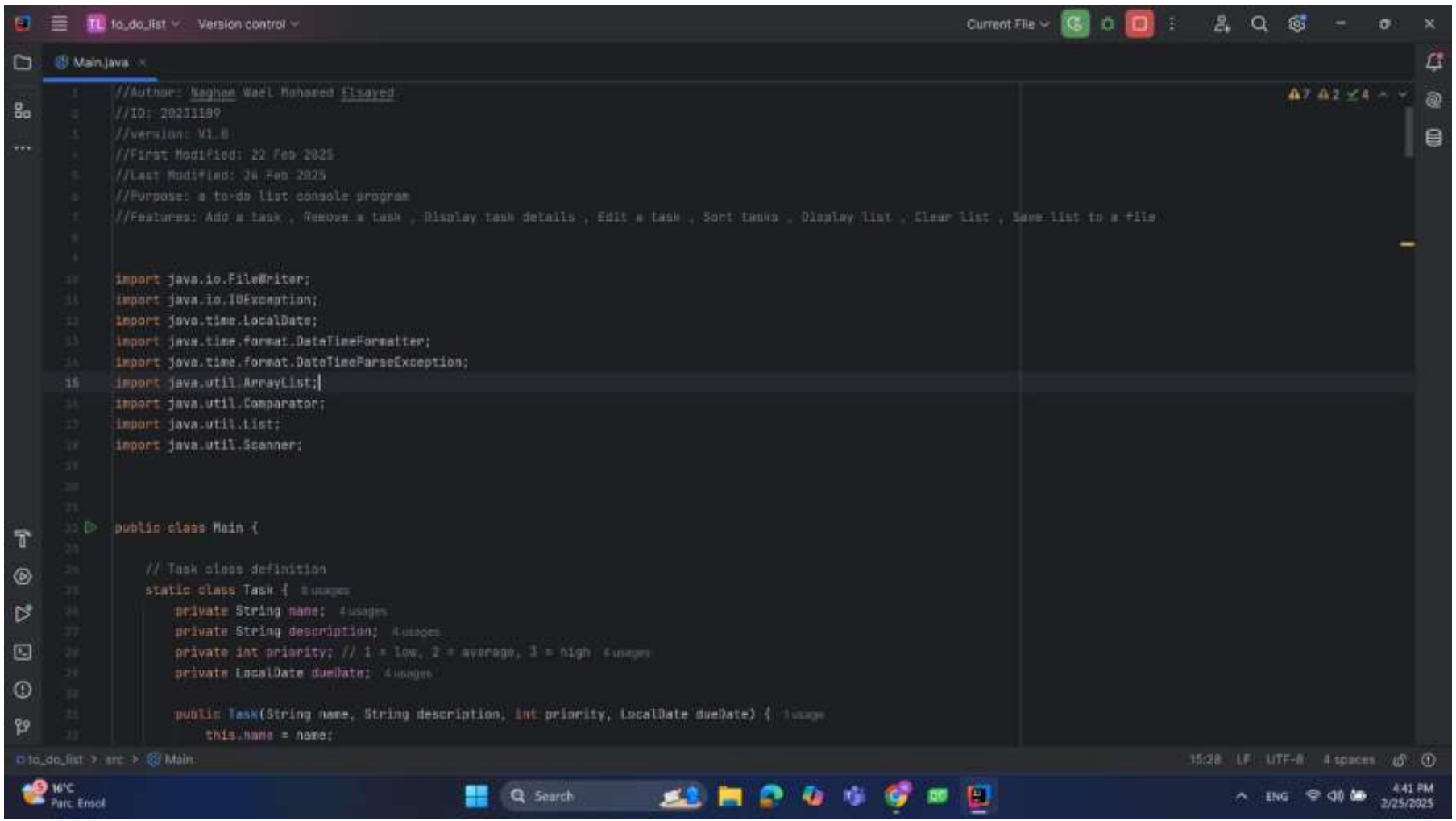

Assignment 1
Version control
Current File
Project
Manger.java
Assignment 1
.idea
out
src
Board
Board
Tasks
Workers
Manger
Manger
Normal
Junior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
package Workers.Manger;
import Workers.Worker;
/**
* The {@code 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 {
/** The manager's username and password used for authentication.*/
private String Username, Password;
/**
* 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) {
super(name, salary, FieldType);
this.Username = Username;
this.Password = Password;
}
public String getUsername() {
return Username;
Assignment 1 › src › Workers › Manger › Manger › Manger
CRLF
UTF-8
4 spaces

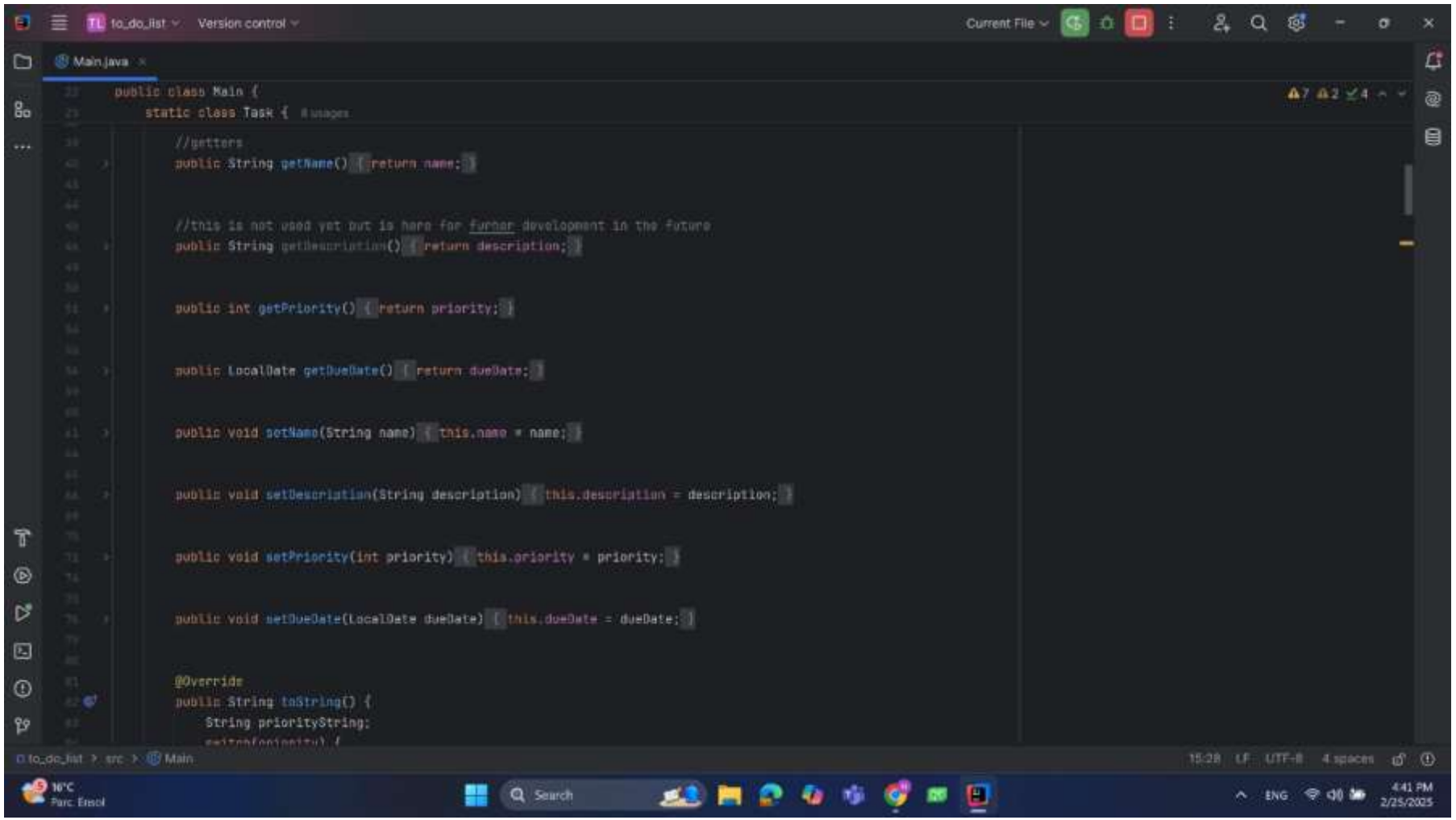

Assignment 1
Version control
Current File
Project
Tasks.java
Assignment 1
idea
out
src
Board
Board
Tasks
Tasks
Workers
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
package Tasks;
import java.time.LocalDate;
Edit | Explain | Test | Document | Fix
/**
* The {@code Tasks} class 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 {
private static int idCounter = 1;
int taskID = idCounter++;
private String description;
private double bonus;
private String priority;
private LocalDate deadline;
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) {
this.description = description;
this.bonus = bonus;
this.priority = priority;
Assignment 1 > src > Tasks > Tasks > showTaskDetails
tabnine Basic
57:6
CRLF
UTF-8
4 spaces

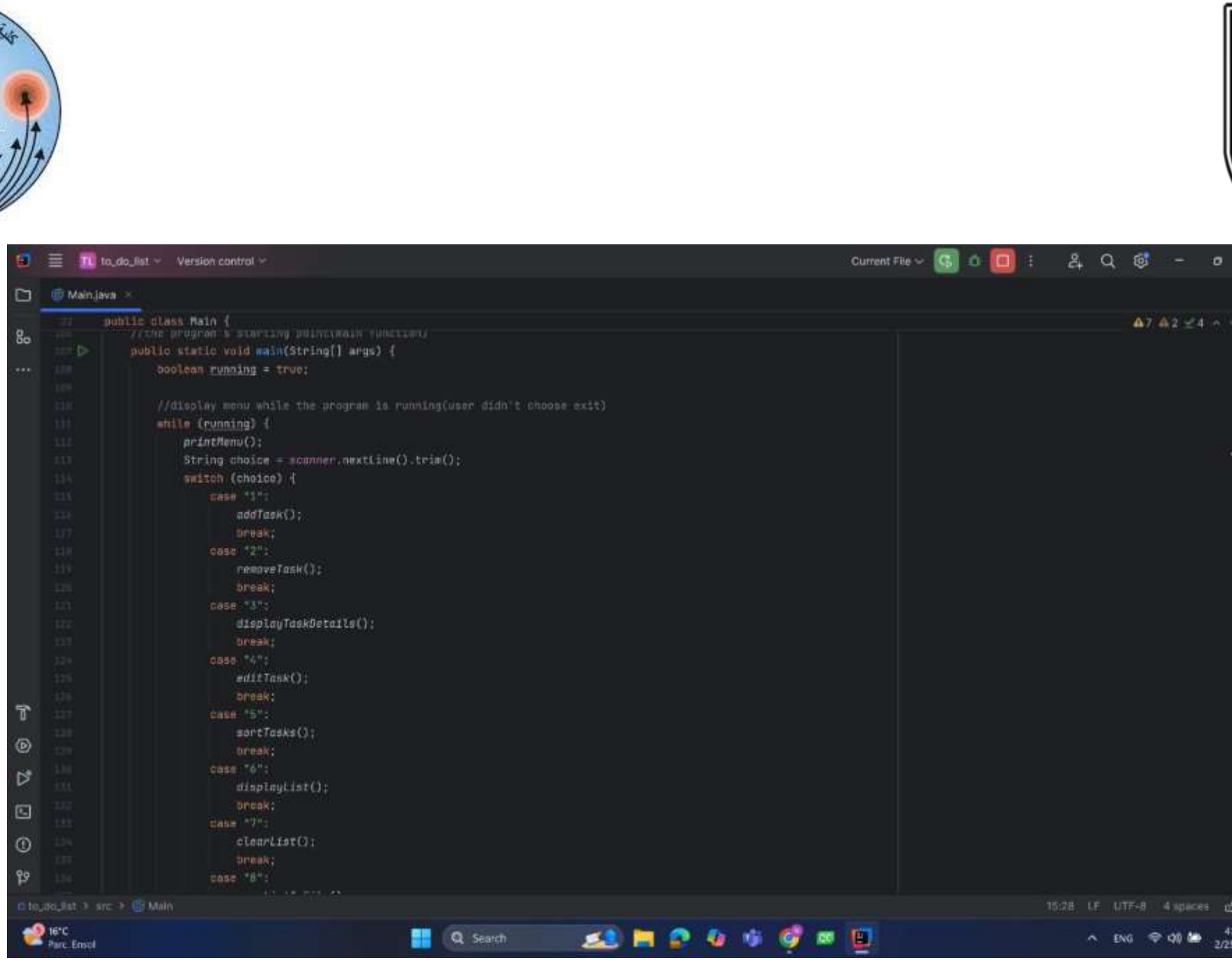

Assignment 1
Version control
Current File
Project
Normal.java
Assignment 1
.idea
out
src
Board
Board
Tasks
Workers
Manger
Normal
Junior
Senior
Normal
Worker
Main
.gitignore
Assignment 1.iml
External Libraries
Scratches and Consoles
public class Normal extends Worker {
private double Penalties = 0, Bounces = 0;
private long numberTasks = 0, maxTasks = 0;
// adding id
private static int lastId = 0;
private int id;
private ArrayList<Tasks> TasksToDo = new ArrayList<Tasks>();
/**
* 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) {
super(Name, Salary, FieldType);
this.maxTasks = MaxTasks;
id = lastId++;
}
/**
* 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) {
Scanner sc = new Scanner(System.in);
int choice = -1;
Assignment 1 > src > Workers > Normal > Normal > removeTask
See Tabnine Insights
CRLF
UTF-8
4 spaces

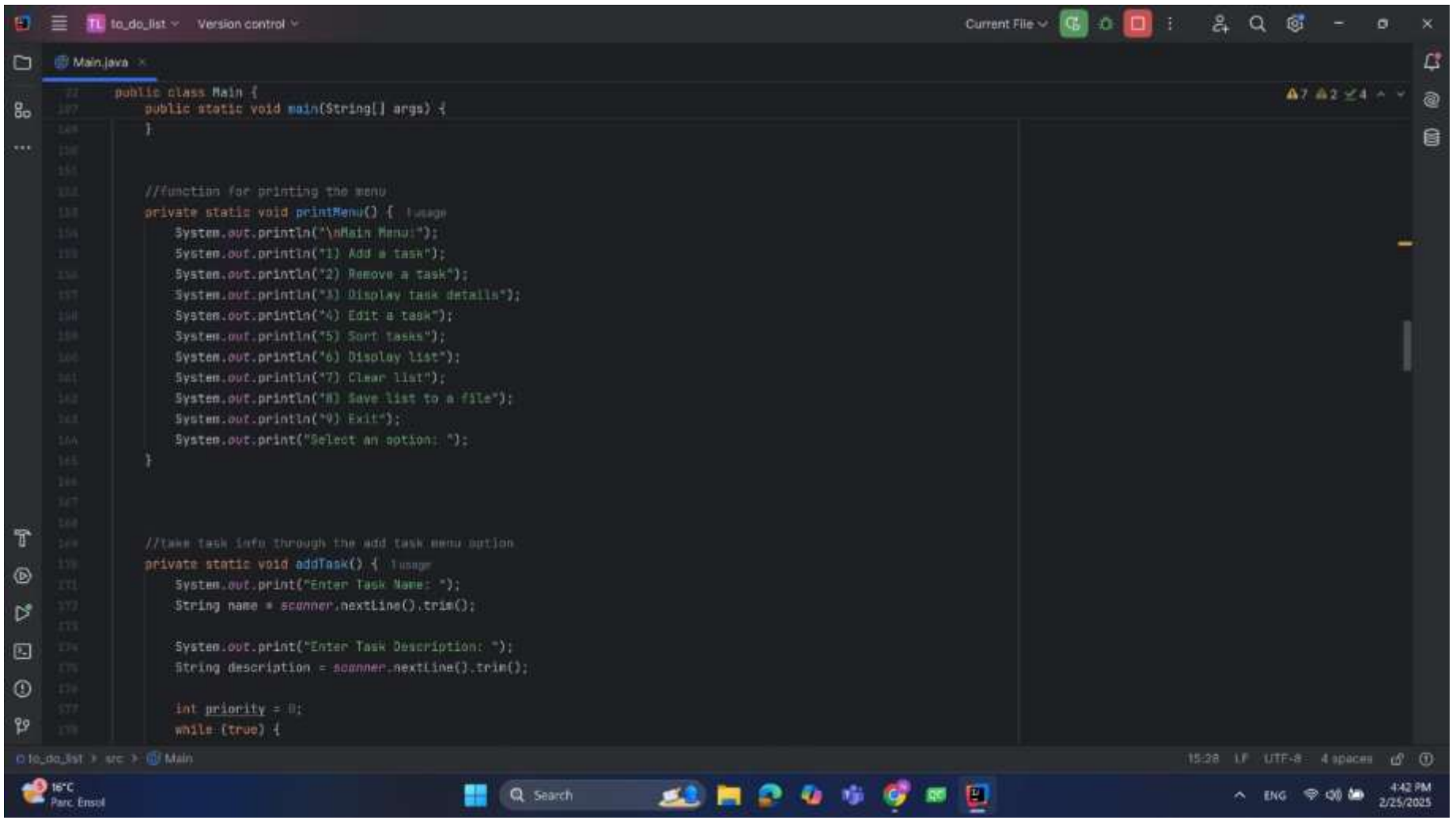
Nagham Wael:

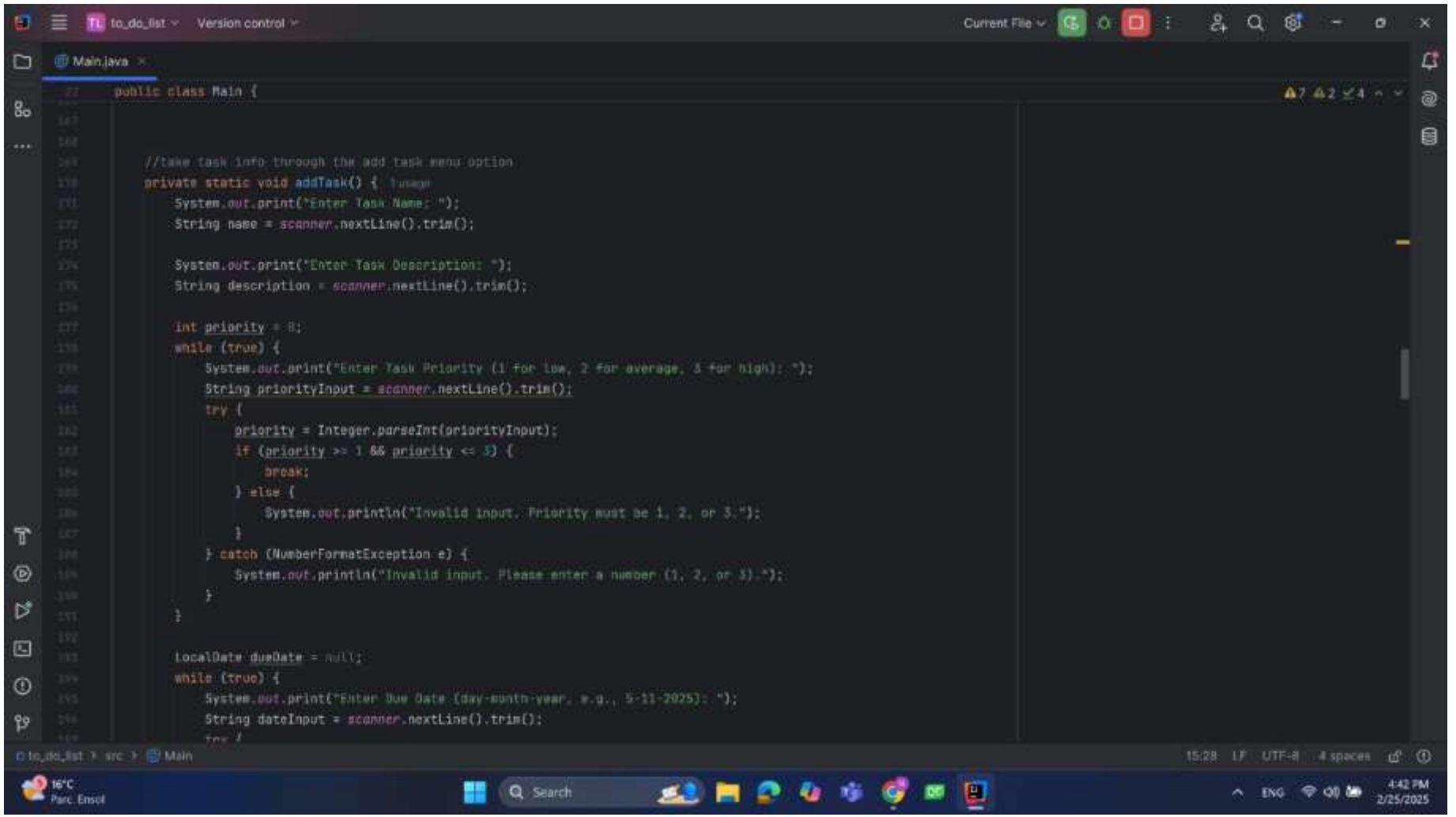
- Hours of Study: 12h
- Source of Study: Geeks For Geeks + https://youtube.com/playlist?list=PLCInYL3l2AajYlZGzU_LVrHdoouf8W6ZN&si=tfRlF2iNxOtJehum
- Video Link: https://youtu.be/VqOBuBvfRiw?si=WCep92XFi12x7LnN

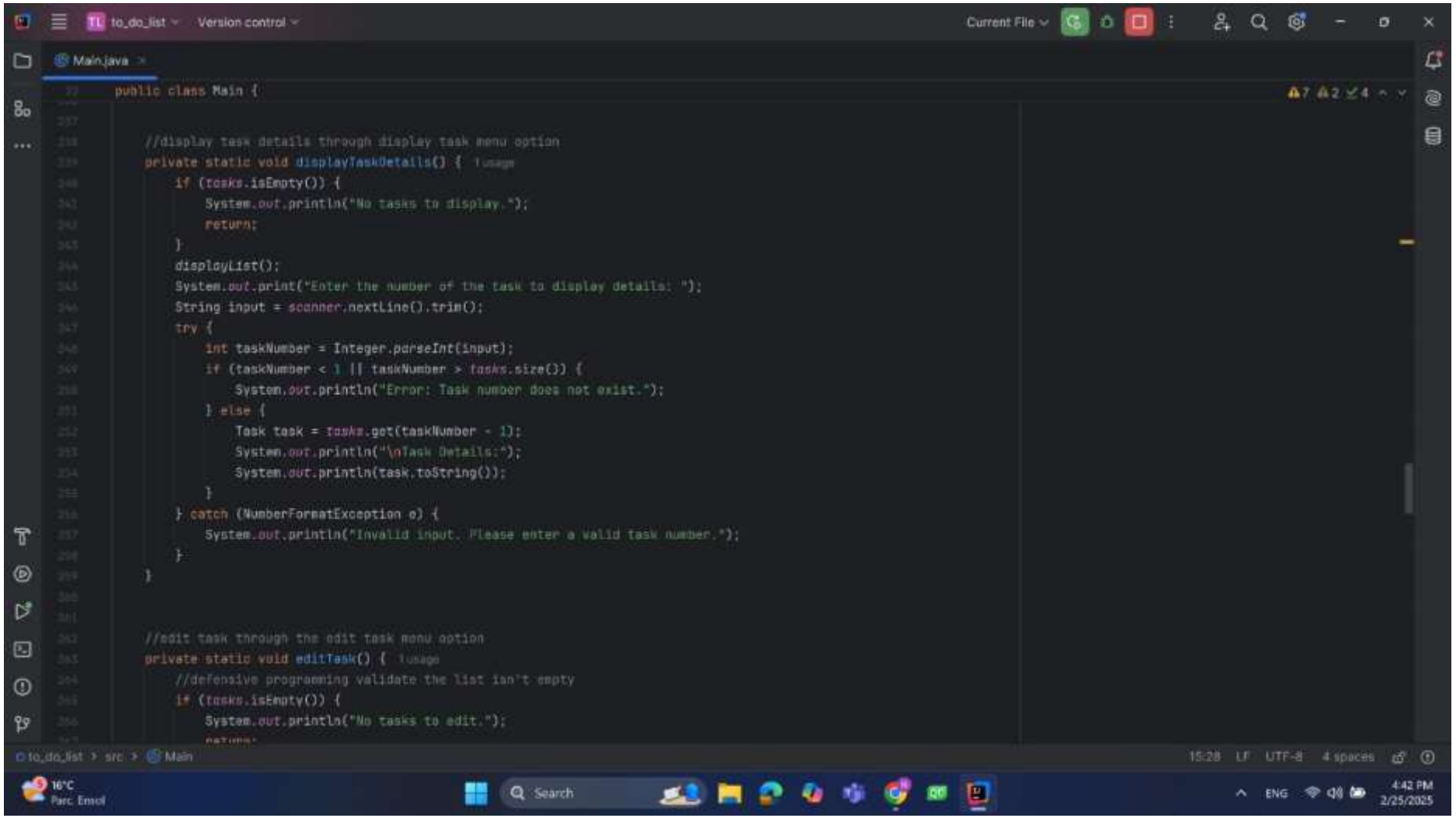

to_do_list
Version control
Current File
Main.java
//Author: Nagham Wael Mohamed Elsayed
//ID: 20231189
//version: V1.0
//First Modified: 22 Feb 2025
//Last Modified: 24 Feb 2025
//Purpose: a to-do list console program
//Features: Add a task , Remove a task , Display task details , Edit a task , Sort tasks , Display list , Clear list , Save list to a file
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeParseException;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
import java.util.Scanner;
public class Main {
// Task class definition
static class Task {
private String name;
private String description;
private int priority; // 1 = low, 2 = average, 3 = high
private LocalDate dueDate;
public Task(String name, String description, int priority, LocalDate dueDate) {
this.name = name;
to_do_list > src > Main
15:28 LF UTF-8 4 spaces
16°C
Parc Ensol
Search
ENG
4:41 PM
2/25/2025

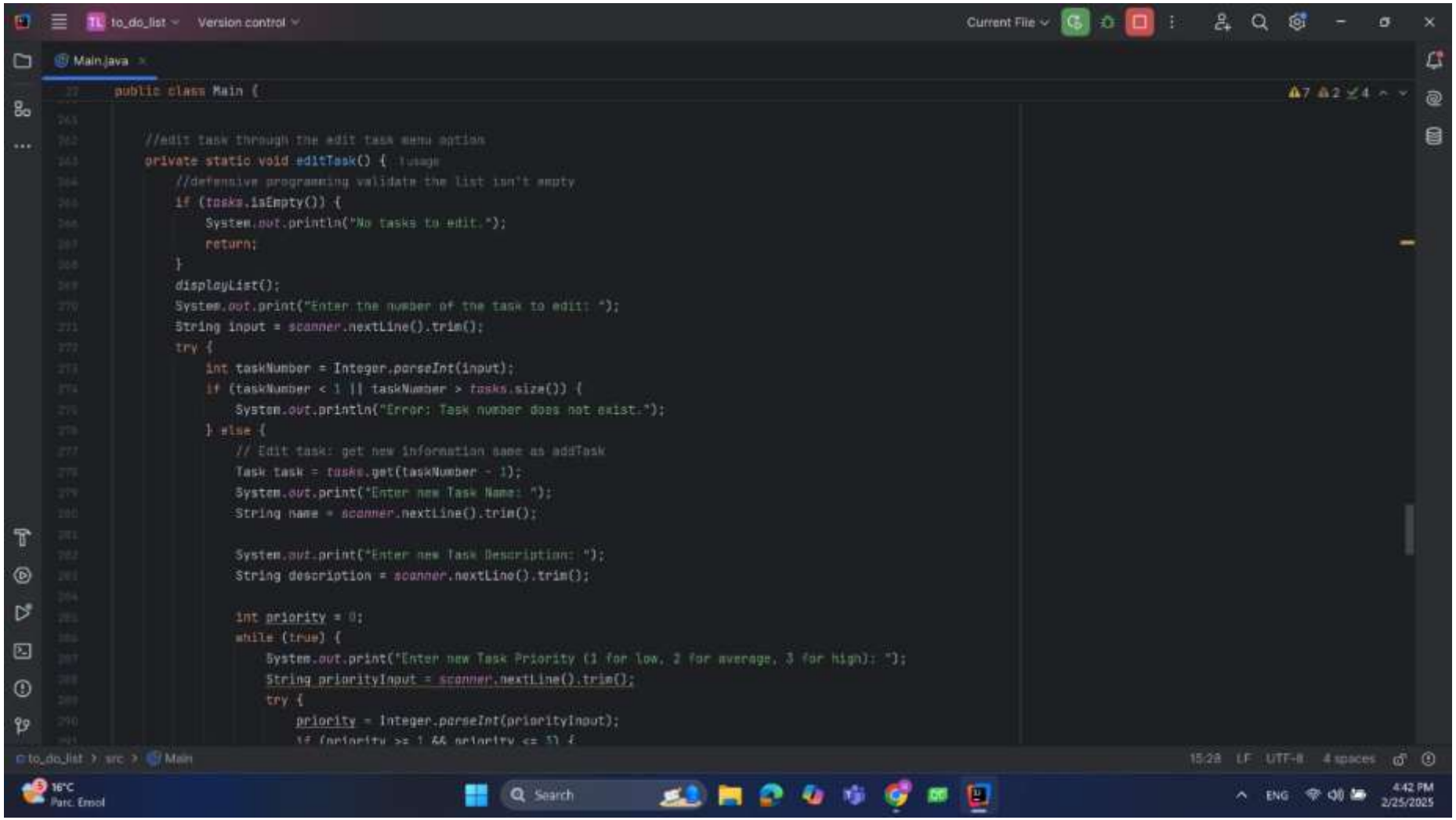

to_do_list
Version control
Current File
Main.java
public class Main {
static class Task {
//getters
public String getName() { return name; }
//this is not used yet but is here for further development in the future
public String getDescription() { return description; }
public int getPriority() { return 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;
to_do_list > src > Main
15:28 LF UTF-8 4 spaces
16°C
Parc. Ensol
Search
ENG
4:41 PM
2/25/2025


to_do_list
Version control
Current File
Main.java
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":
to_do_list > src > Main
15:28 LF UTF-8 4 spaces
16°C
Parc. Ensol
Search
ENG
4:42 PM
2/25/2025


to_do_list
Version control
Current File
Main.java
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) {
to_do_list > src > Main
15:28 LF UTF-8 4 spaces
16°C
Parc Ensol
Search
ENG
4:42 PM
2/25/2025

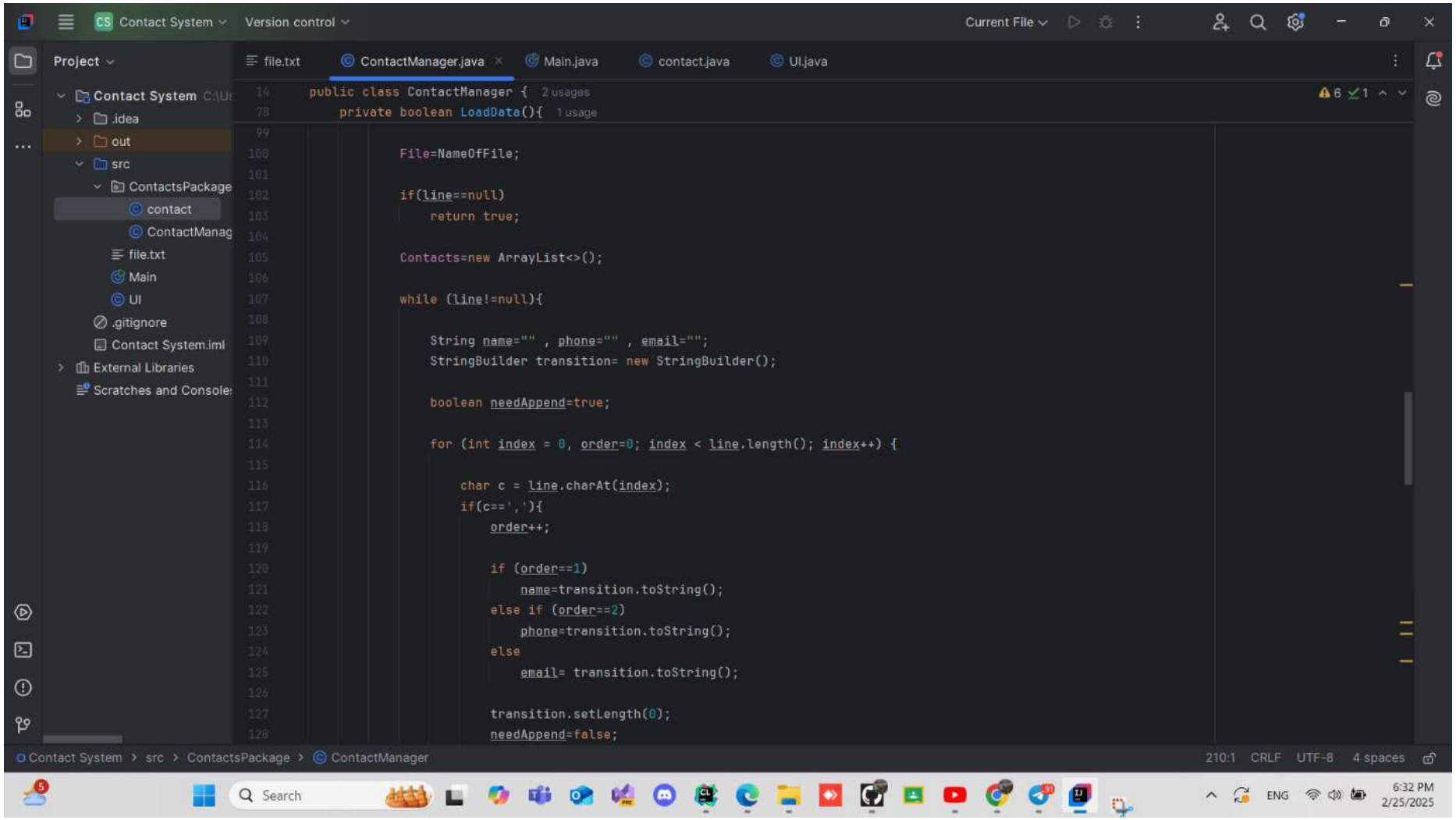

to_do_list
Version control
Current File
Main.java
public class Main {
//take task info through the add task menu option
private static void addTask() { 1 usage
System.out.print("Enter Task Name: ");
String name = scanner.nextLine().trim();
System.out.print("Enter Task Description: ");
String description = 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();
to_do_list > src > Main
15:28 LF UTF-8 4 spaces
16°C
Parc. Ensol
Search
ENG
4:42 PM
2/25/2025

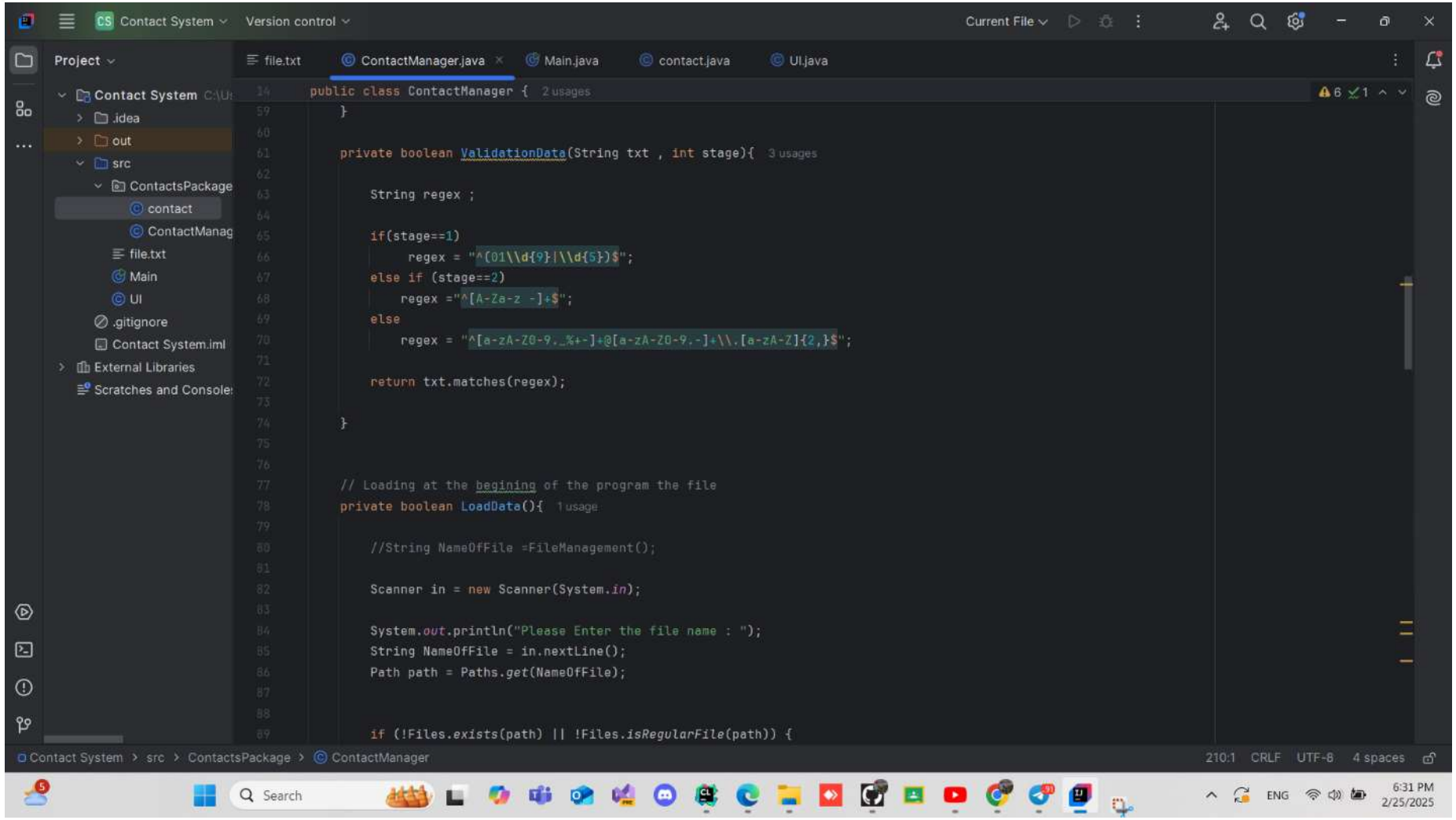

to_do_list
Version control
Current File
Main.java
public class Main {
//display task details through display task menu option
private static void displayTaskDetails() { 1 usage
if (tasks.isEmpty()) {
System.out.println("No tasks to display.");
return;
}
displayList();
System.out.print("Enter the number 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.");
to_do_list > src > Main
15:28 LF UTF-8 4 spaces
16°C
Parc. Ensol
Search
ENG
4:42 PM
2/25/2025

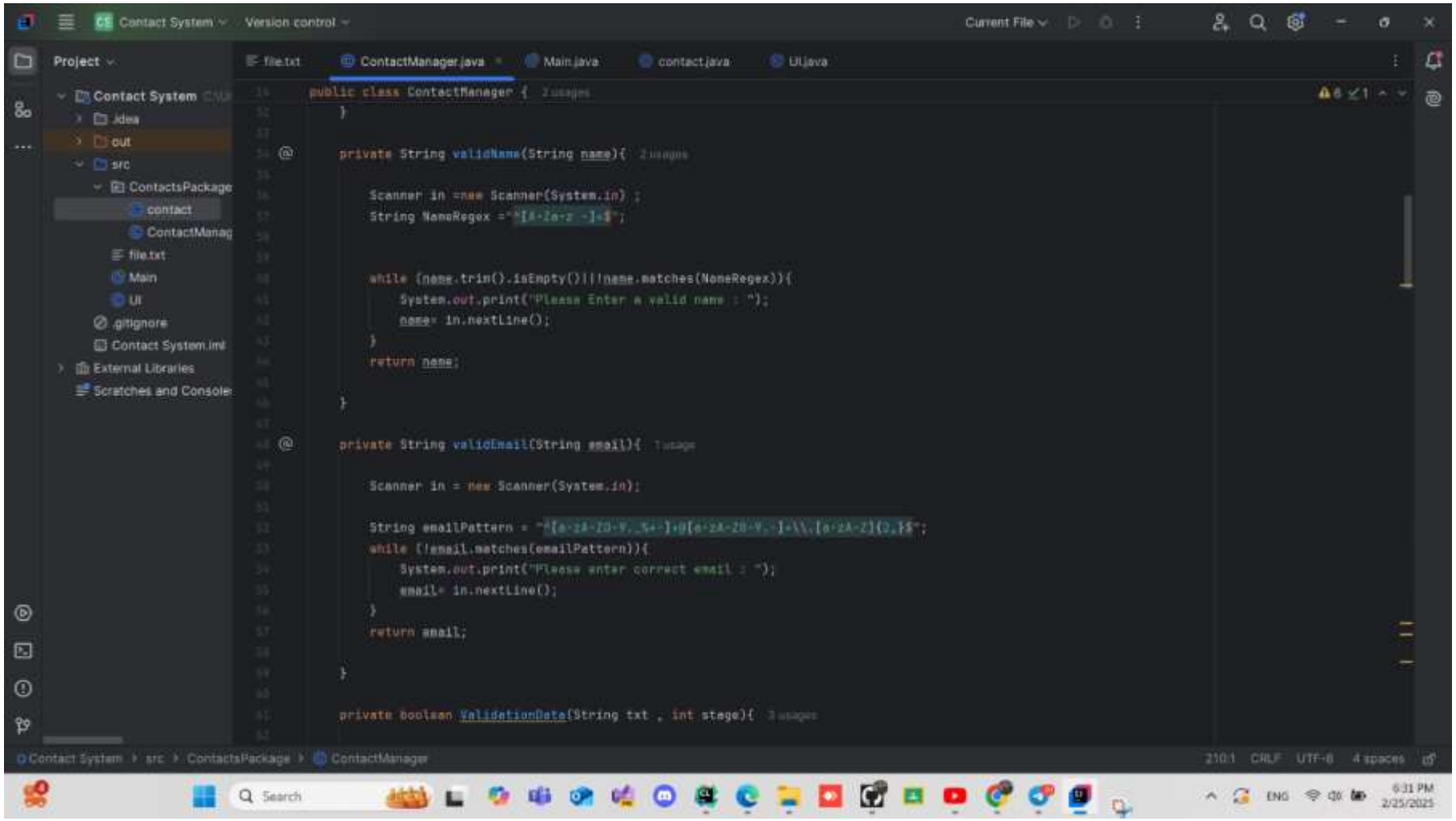

to_do_list
Version control
Current File
Main.java
public class Main {
//edit task through the edit task menu option
private static void editTask() { 1 usage
//defensive programming validate the list isn't empty
if (tasks.isEmpty()) {
System.out.println("No tasks to edit.");
return;
}
displayList();
System.out.print("Enter 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);
to_do_list > src > Main
15:28 LF UTF-8 4 spaces
16°C
Parc. Ensol
Search
ENG
4:42 PM
2/25/2025

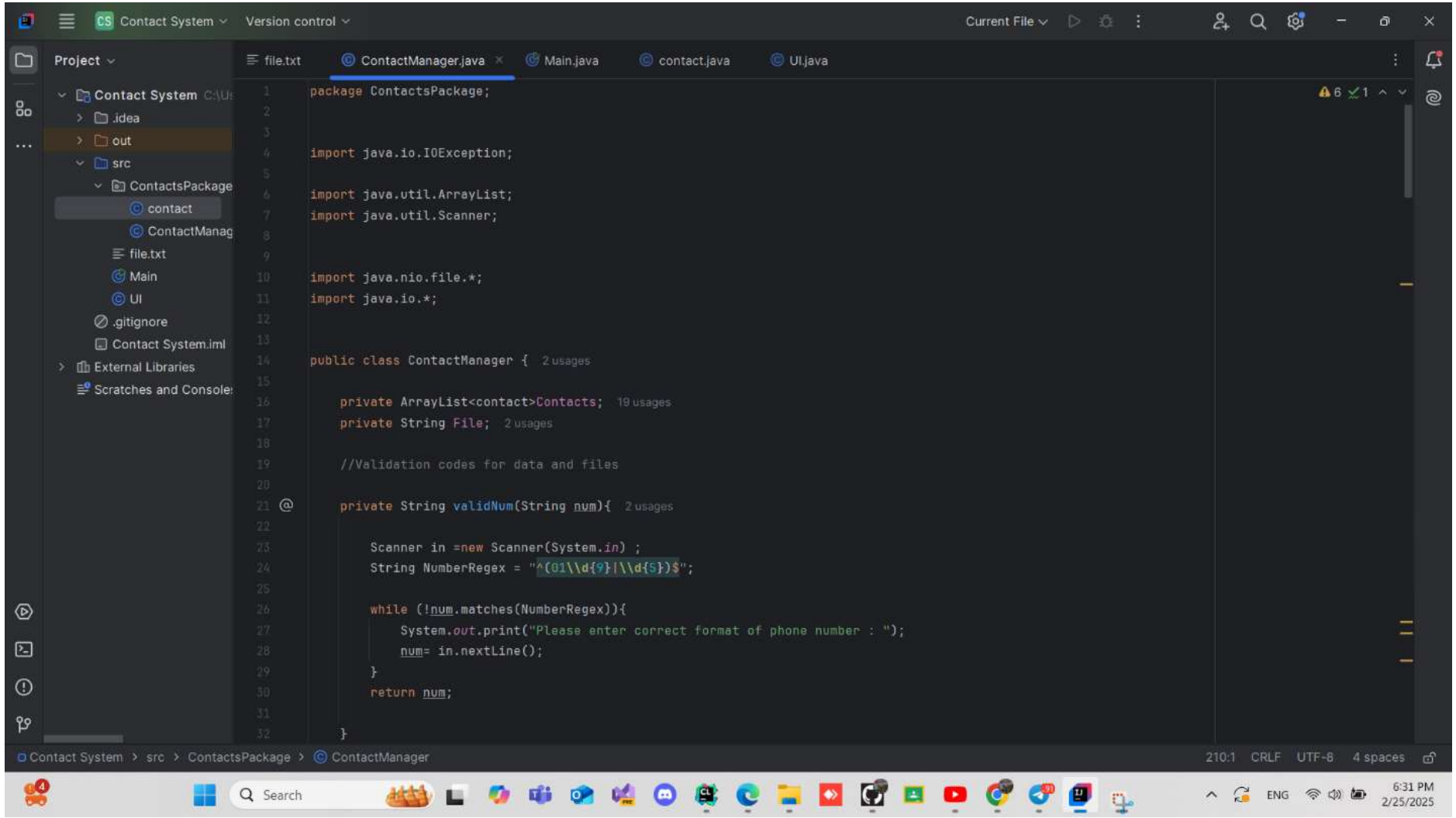
Fatema El-Zhraa Ahmed:

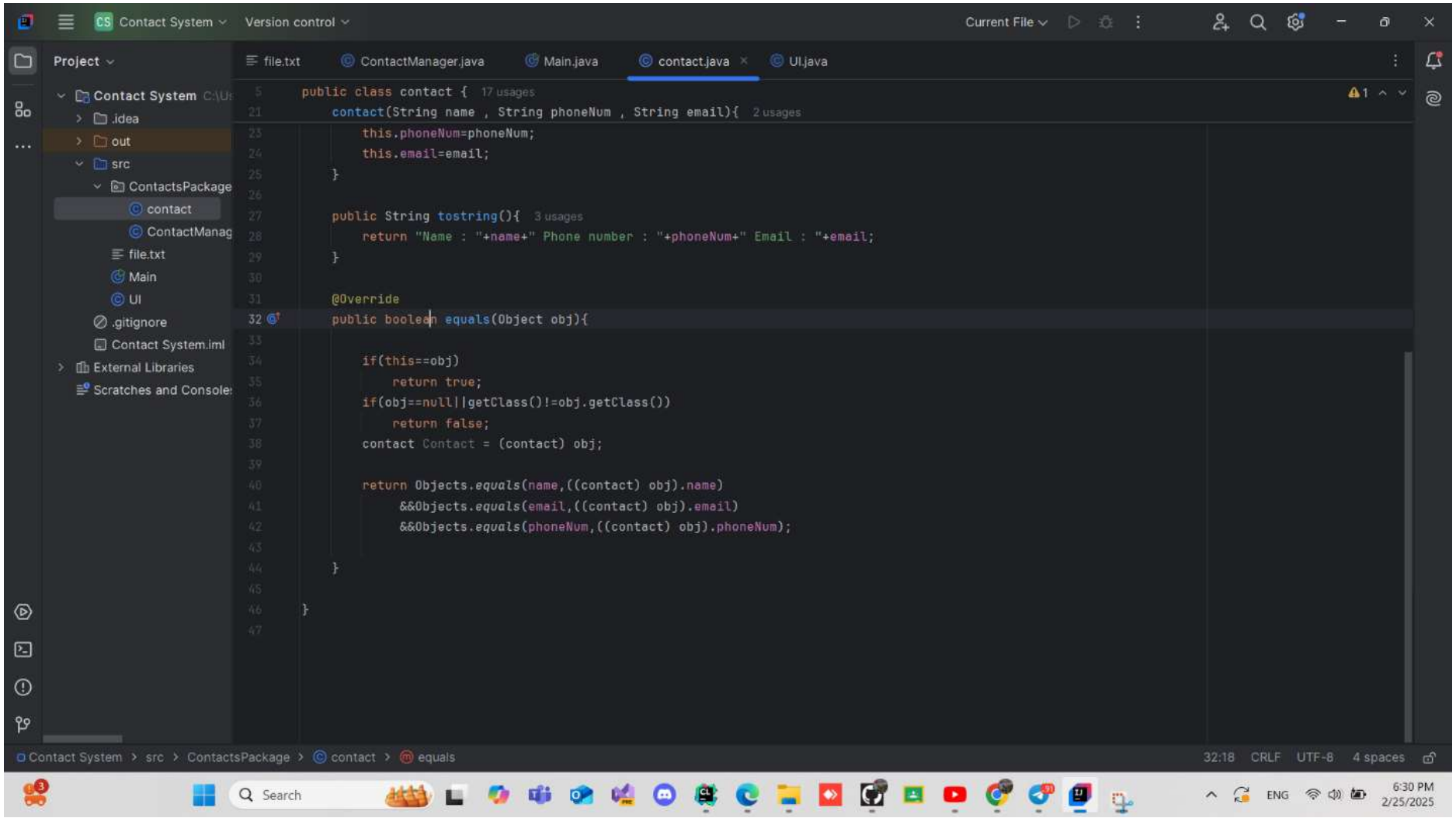
- Hours of Study:12 hours
- Source of Study : https://www.youtube.com/playlist?list=PLCInYL3l2AajYlZGzU_LVrHdoouf8W6ZN
- Video Link: https://www.youtube.com/watch?v=fuV9C_FRDf8

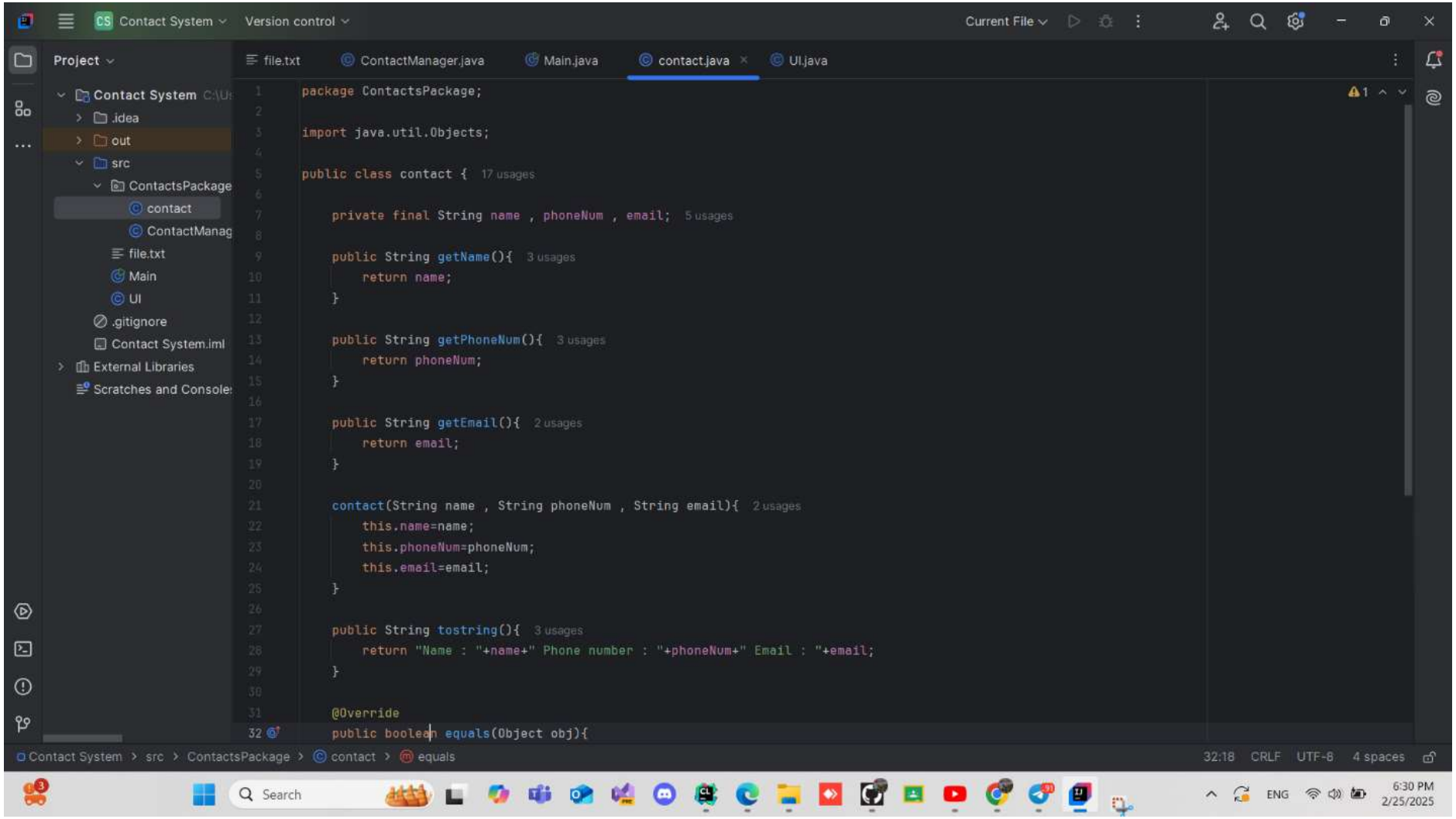

Contact System
Version control
Current File
Project
file.txt
ContactManager.java
Main.java
contact.java
UI.java
Contact System
.idea
out
src
ContactsPackage
contact
ContactManag
file.txt
Main
UI
.gitignore
Contact System.iml
External Libraries
Scratches and Console
public class ContactManager { 2 usages
private boolean LoadData(){ 1 usage
File=NameOfFile;
if(line==null)
return true;
Contacts=new ArrayList<>();
while (line!=null){
String name="" , phone="" , email="";
StringBuilder transition= new StringBuilder();
boolean needAppend=true;
for (int index = 0, order=0; index < line.length(); index++) {
char c = line.charAt(index);
if(c==','){
order++;
if (order==1)
name=transition.toString();
else if (order==2)
phone=transition.toString();
else
email= transition.toString();
transition.setLength(0);
needAppend=false;
Contact System › src › ContactsPackage › ContactManager
210:1 CRLF UTF-8 4 spaces
Search
ENG
6:32 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 Console
public class ContactManager { 2 usages
}
private boolean ValidationData(String txt , int stage){ 3 usages
String regex ;
if(stage==1)
regex = "^(01\\d{9}|\\d{5})$";
else if (stage==2)
regex ="^[A-Za-z -]+$";
else
regex = "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$";
return txt.matches(regex);
}
// Loading at the begining of the program the file
private boolean LoadData(){ 1 usage
//String NameOfFile =FileManagement();
Scanner in = new Scanner(System.in);
System.out.println("Please Enter the file name : ");
String NameOfFile = in.nextLine();
Path path = Paths.get(NameOfFile);
if (!Files.exists(path) || !Files.isRegularFile(path)) {
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 {
private String validName(String name){
Scanner in =new Scanner(System.in) ;
String NameRegex ="[A-Za-z -]+$";
while (name.trim().isEmpty()||!name.matches(NameRegex)){
System.out.print("Please Enter a valid name : ");
name= in.nextLine();
}
return name;
}
private String validEmail(String email){
Scanner in = new Scanner(System.in);
String emailPattern = "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$";
while (!email.matches(emailPattern)){
System.out.print("Please enter correct email : ");
email= in.nextLine();
}
return email;
}
private boolean ValidationData(String txt , int stage){
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 Console
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 C:\Us
.idea
out
src
ContactsPackage
contact
ContactManag
file.txt
Main
UI
.gitignore
Contact System.iml
External Libraries
Scratches and Console
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 C:\Us
.idea
out
src
ContactsPackage
contact
ContactManag
file.txt
Main
UI
.gitignore
Contact System.iml
External Libraries
Scratches and Console
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

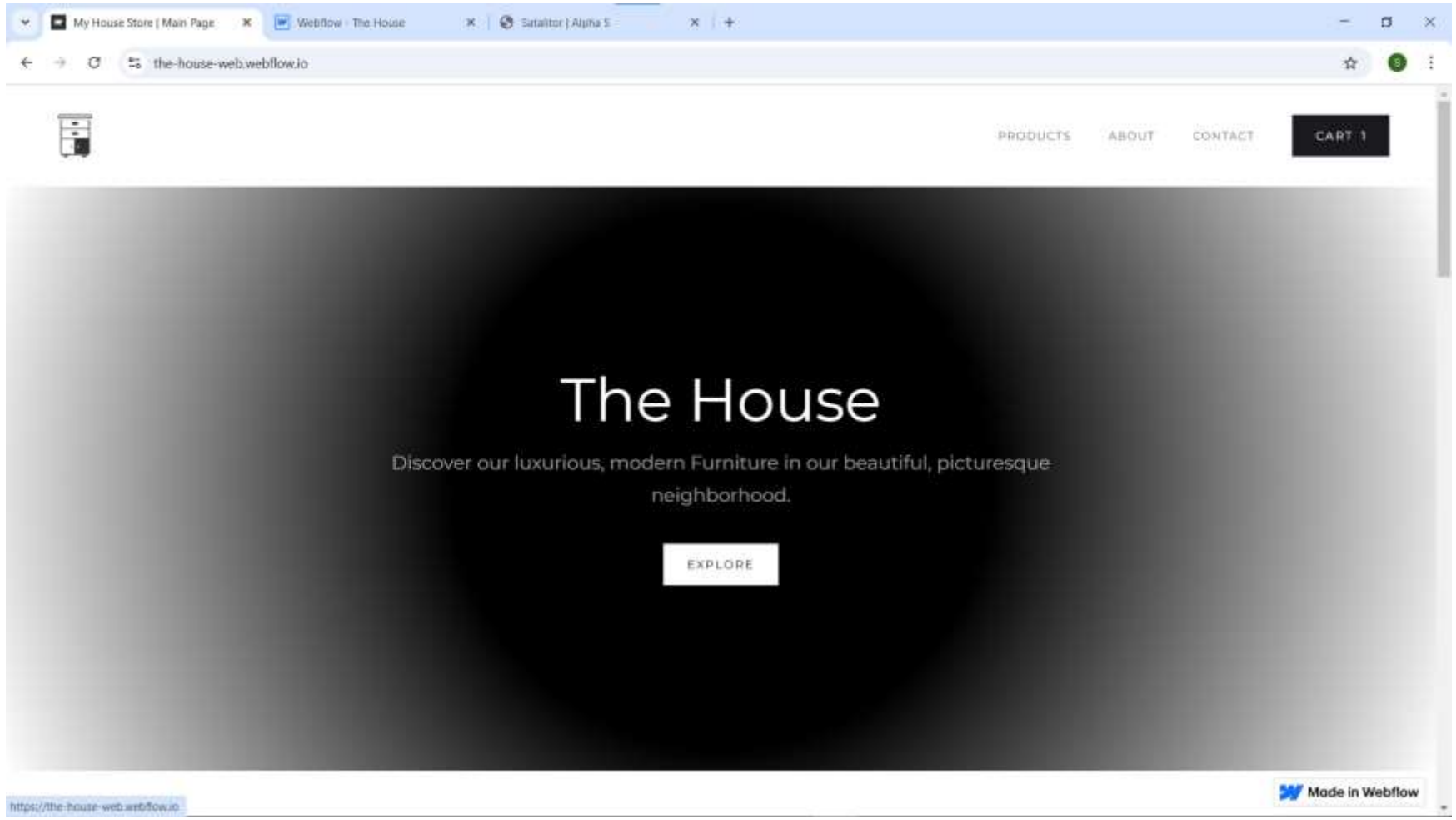
LCNC

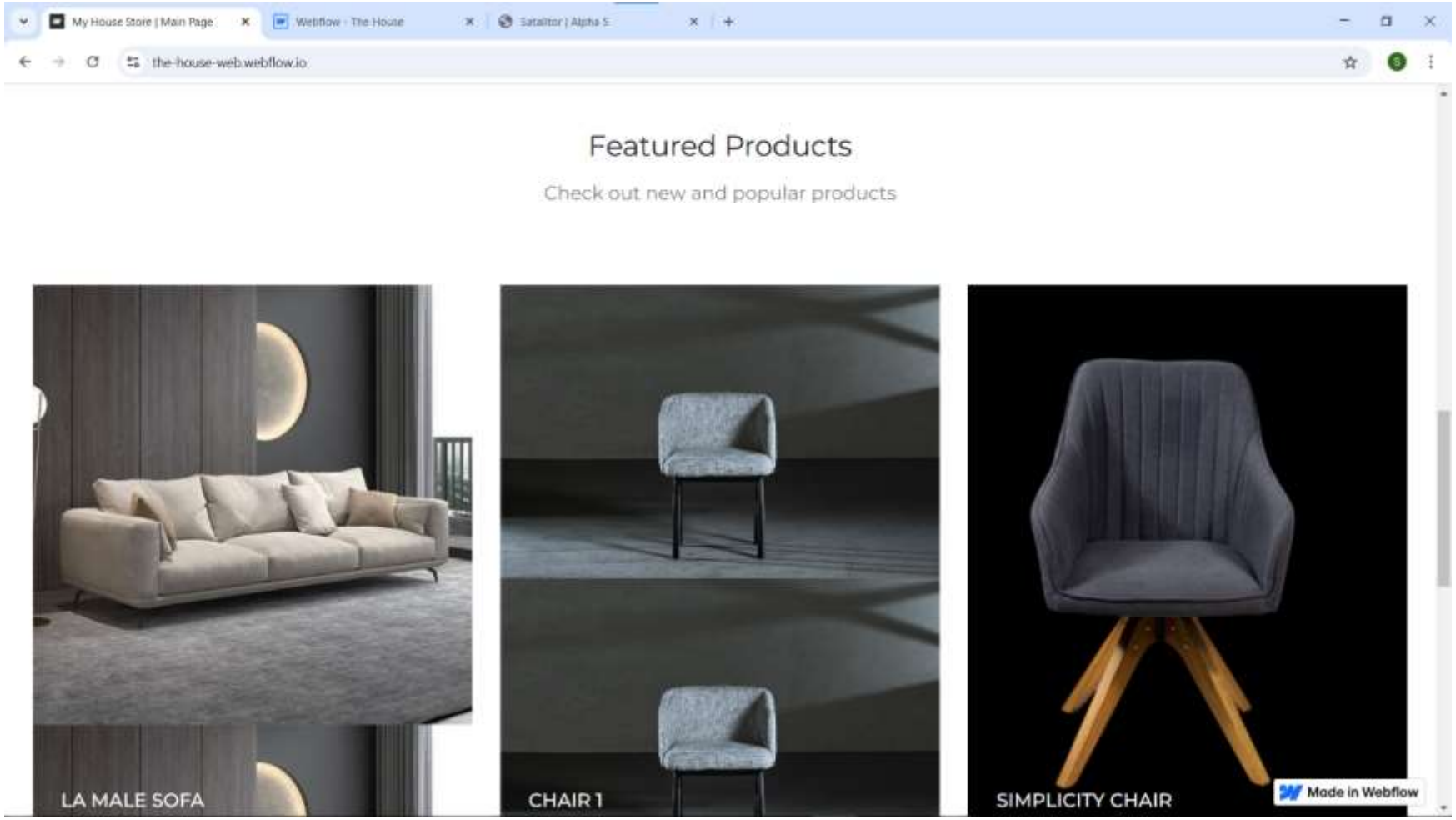
Compare Between Two LCNC Tools:

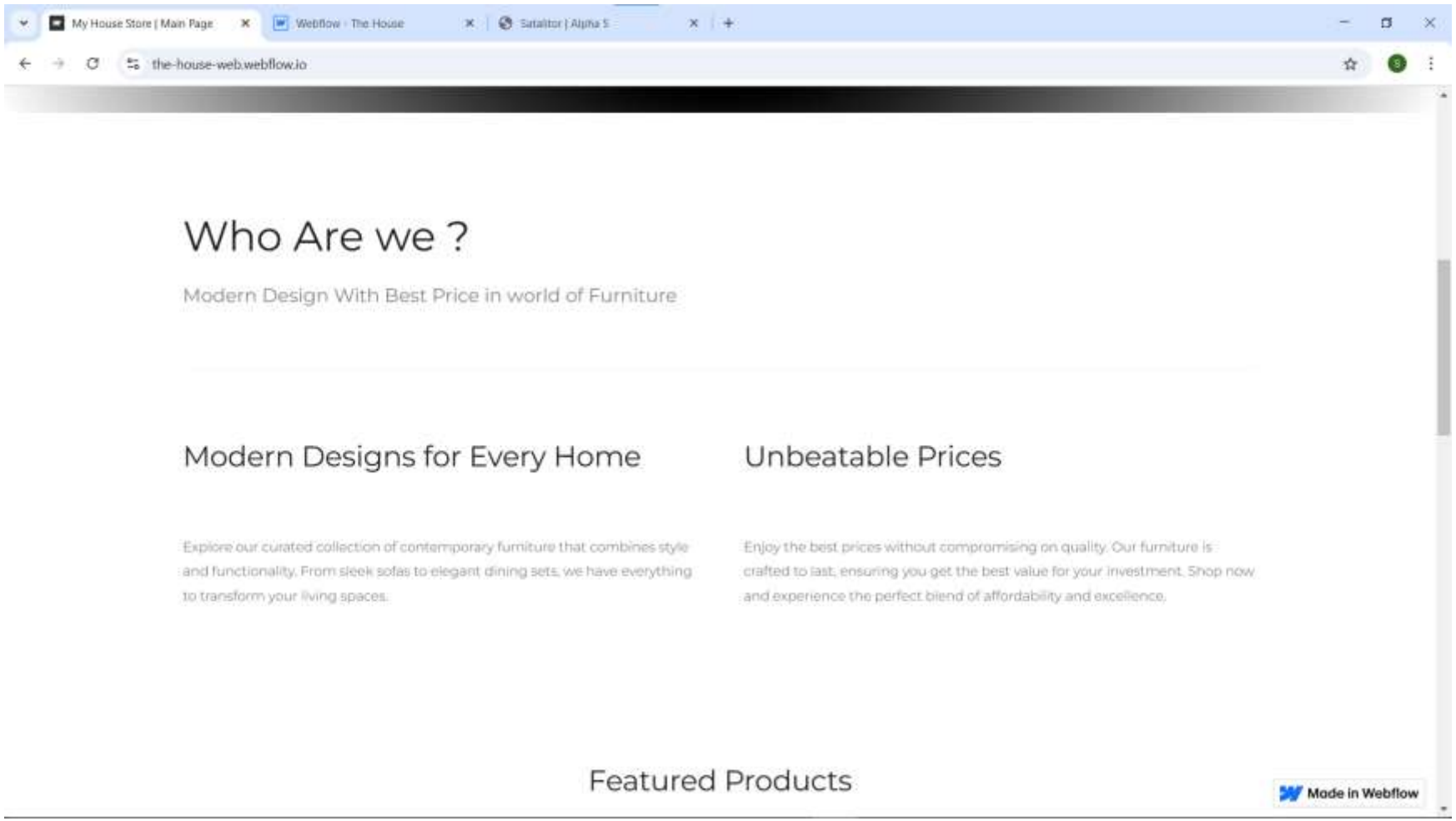
Key of Compare	WebFlow	Thunkable
Purpose	Website design and development	Mobile app development (iOS & Android)
Target Users	Designers, developers, and businesses looking to create responsive websites	Individuals and businesses looking to create mobile apps without coding
Drag-and-Drop Interface	Yes, 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

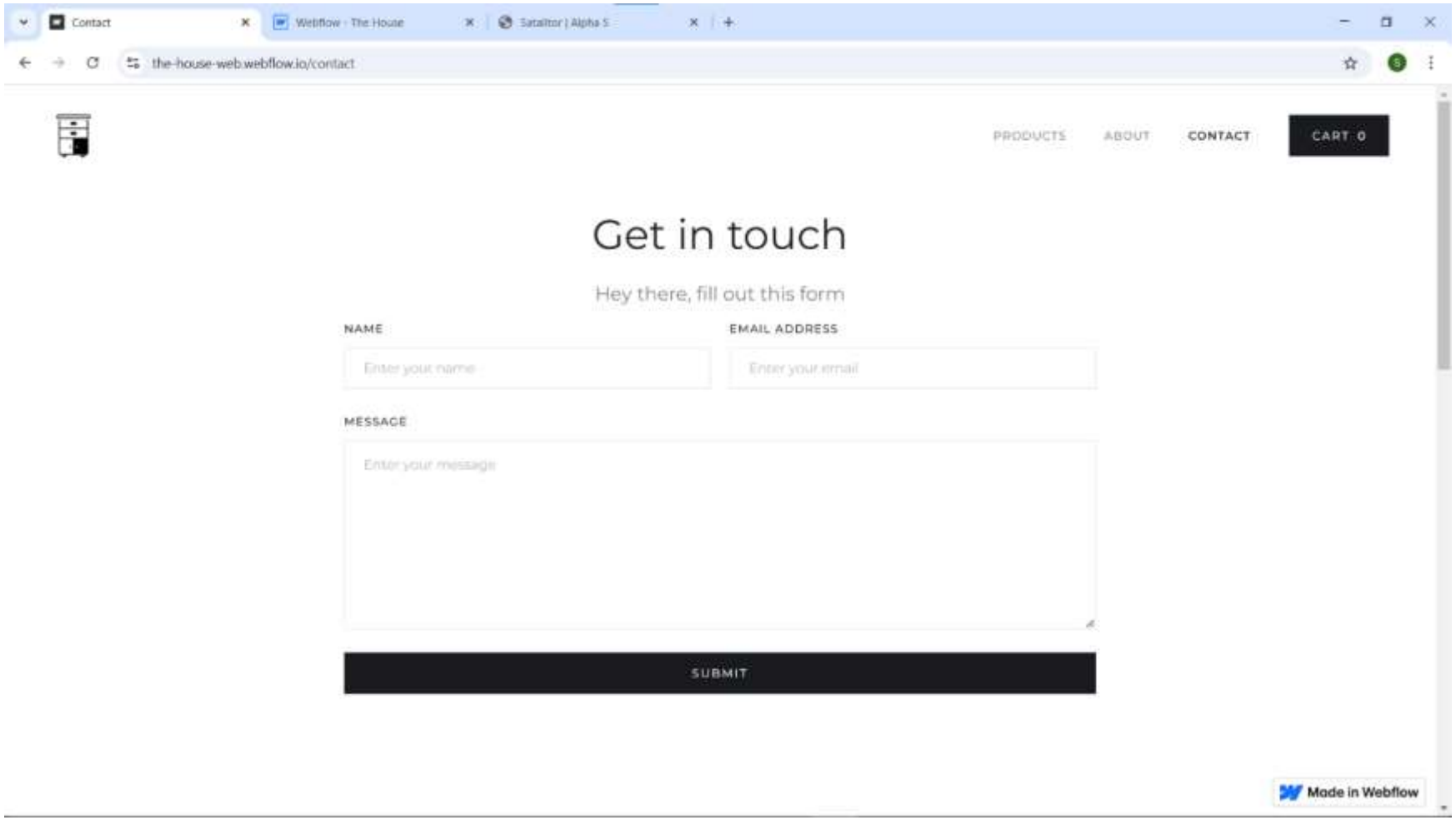
Website link : https://the-house-web.webflow.io/products

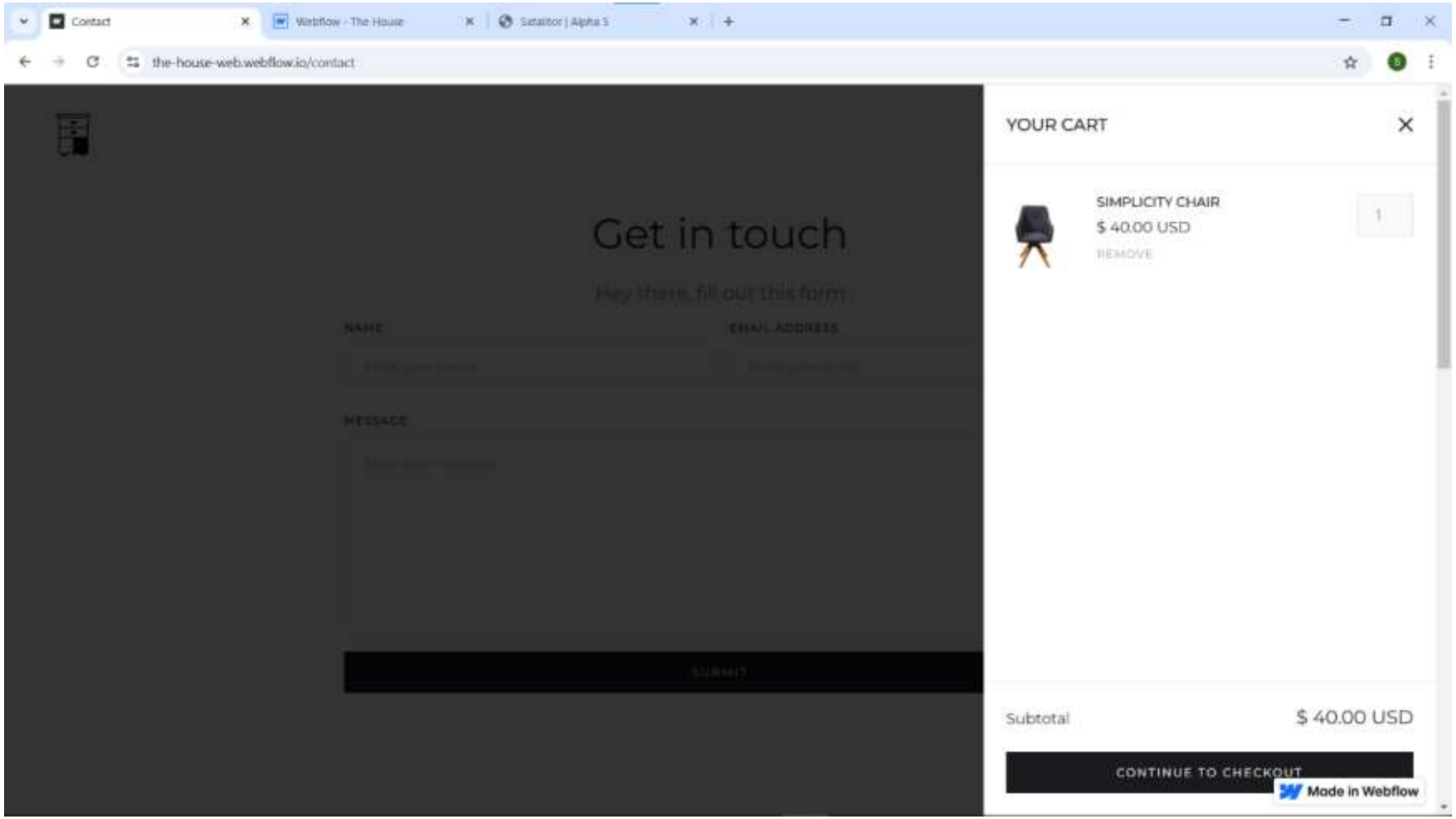
Video link: https://youtu.be/c071WQAifNM?si=sWq_lknsMG3GbJbN

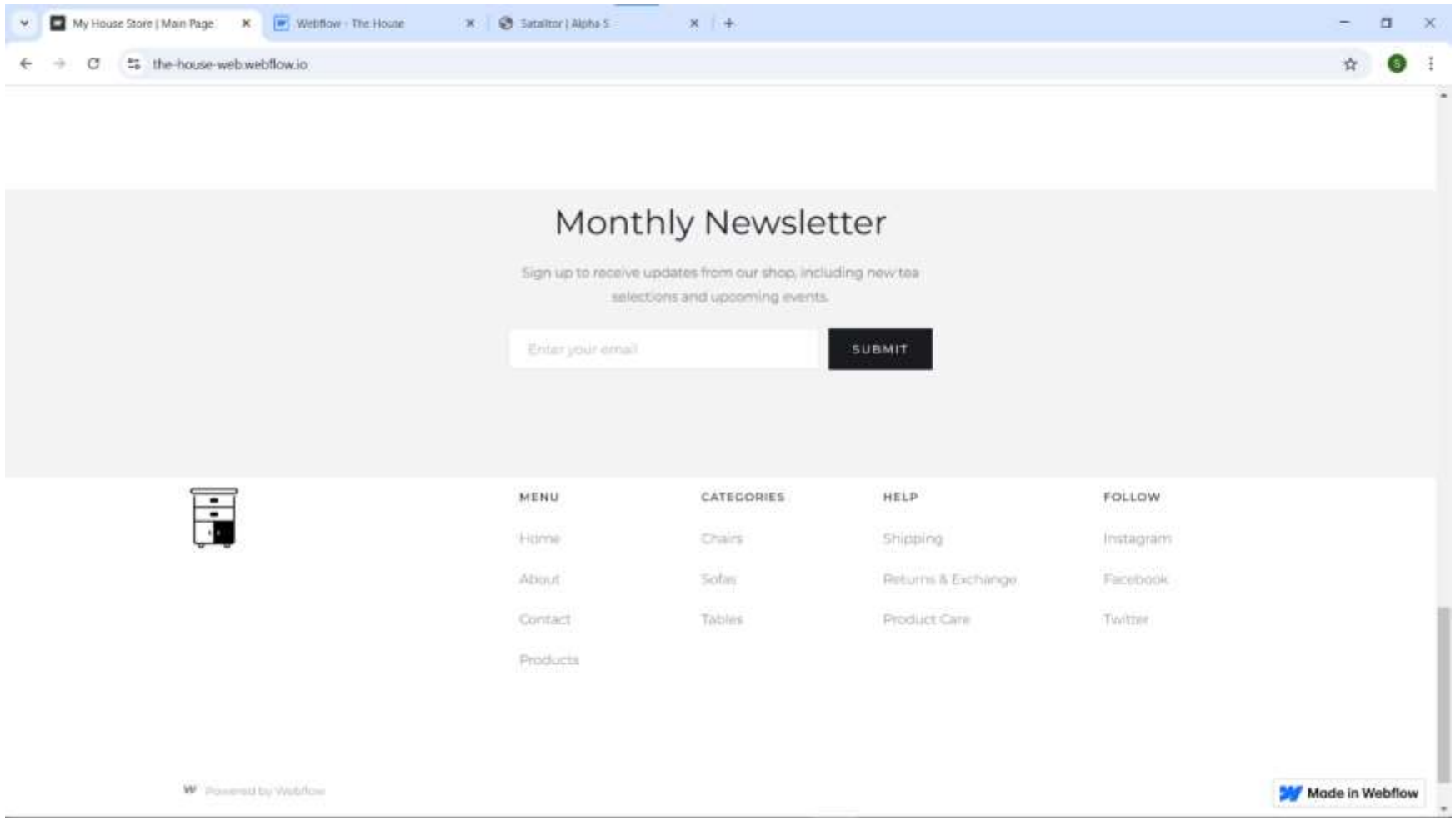

My House Store | Main Page
Webflow - The House
Satalitor | Alpha 5
the-house-web.webflow.io
PRODUCTS
ABOUT
CONTACT
CART 1
The House
Discover our luxurious, modern Furniture in our beautiful, picturesque neighborhood.
EXPLORE
Made in Webflow
https://the-house-web.webflow.io

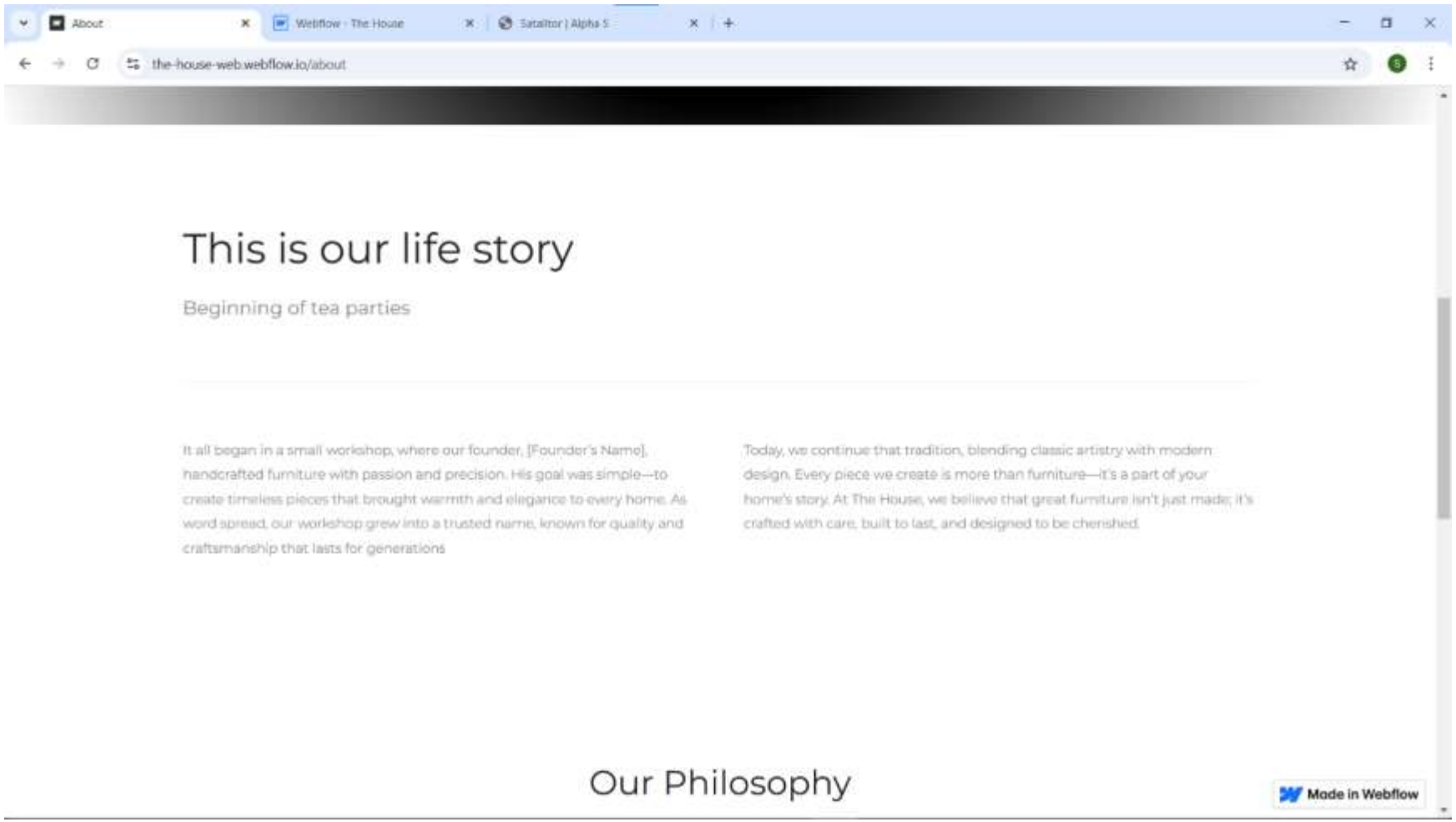

My House Store | Main Page
Webflow - The House
Satalitor | Alpha S
the-house-web.webflow.io
Featured Products
Check out new and popular products
LA MALE SOFA
CHAIR 1
SIMPLICITY CHAIR
Made in Webflow


My House Store | Main Page
Webflow - The House
Satalitor | Alpha S
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


Contact
Webflow - The House
Satellitor | Alpha 5
the-house-web.webflow.io/contact
PRODUCTS
ABOUT
CONTACT
CART 0
Get in touch
Hey there, fill out this form
NAME
Enter your name
EMAIL ADDRESS
Enter your email
MESSAGE
Enter your message
SUBMIT
Made in Webflow


Contact
Webflow - The House
Sabattor | Alpha 5
the-house-web.webflow.io/contact
Get in touch
Hey there, fill out this form
NAME
EMAIL ADDRESS
MESSAGE
SUBMIT
YOUR CART
SIMPLICITY CHAIR
$ 40.00 USD
REMOVE
1
Subtotal
$ 40.00 USD
CONTINUE TO CHECKOUT
Made in Webflow


My House Store | Main Page
Webflow - The House
Satellitor | Alpha S
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


About
Webflow - The House
Satalitor | Alpha S
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

SRS Report

Personal Investment Management Software

Phrase	Definition
ROI (Return on Investment)	A financial metric that calculates the profitability of an investment.
Portfolio Tracking	Monitoring different assets and investments in real-time.
Asset Allocation	The strategy of diversifying investments across different asset classes.
Net Worth Computation	A feature that calculates a user's total financial worth (assets - liabilities).
API (Application Programming Interface)	A set of protocols for integrating with banking systems, stock exchanges, and payment gateways.
Debt Management	Features that automate debt repayment using a predefined percentage of monthly income.
Unbanked & Underbanked	Individuals without full access to traditional banking services, relying on cash or mobile wallets.
Cloud-Based Deployment	Hosting the application on cloud services such as AWS, Google Cloud, or Azure for scalability.

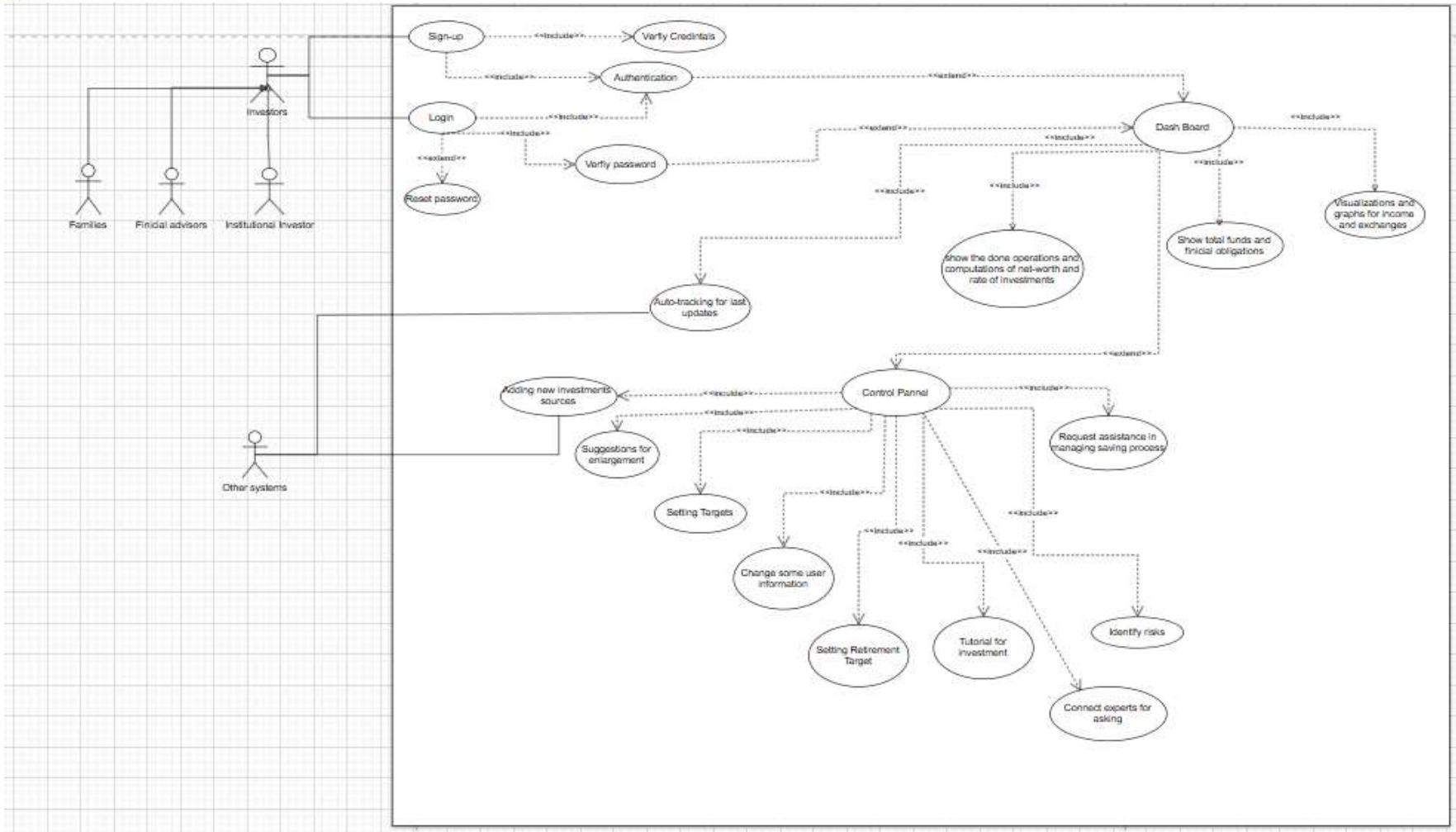
Functional Requirements

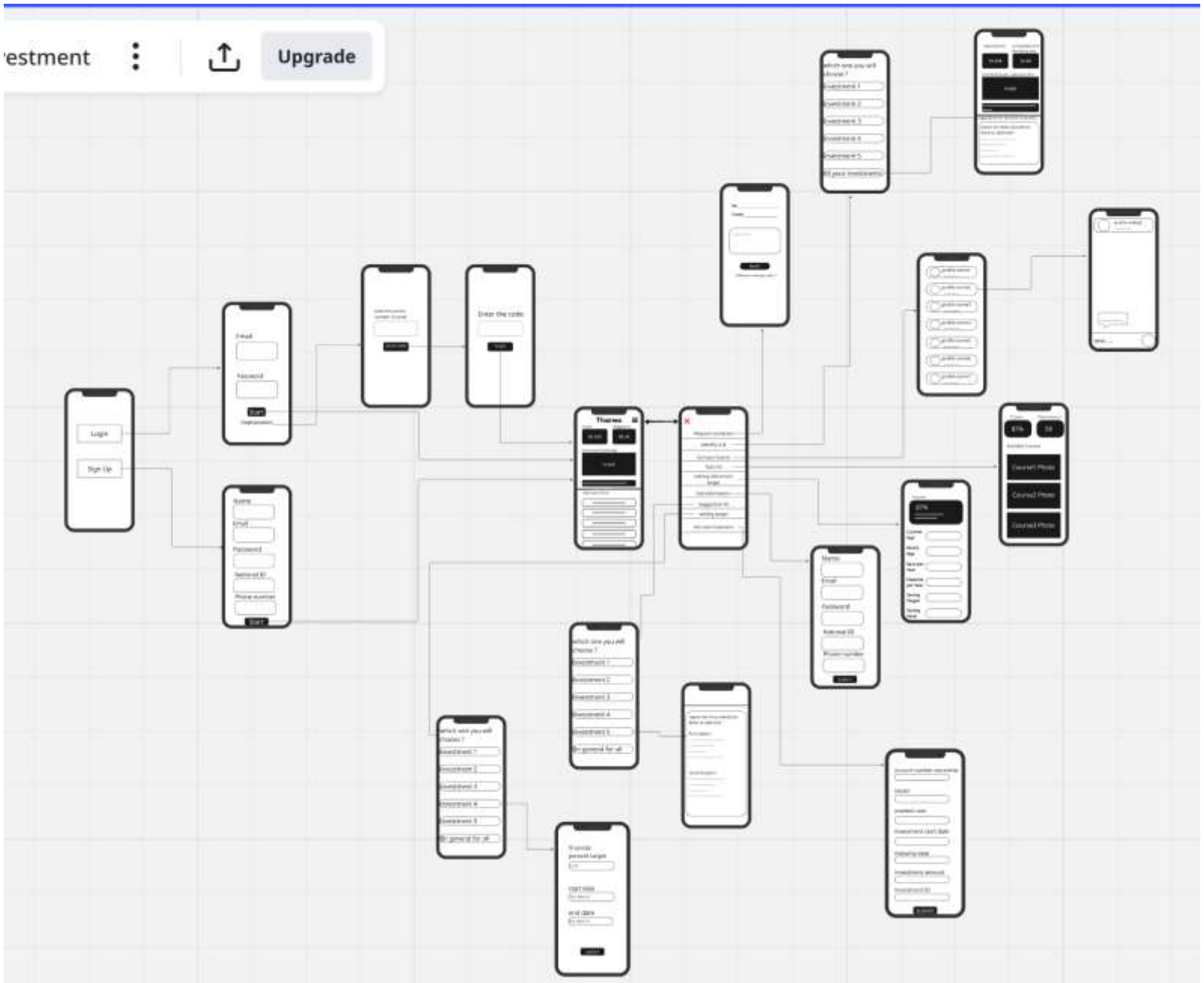
Requirement ID	Requirement Statement
FR01	The First screen appears to the user is the login/sign-up screen
FR02	The application will redirect to the screen showing to two input boxes for the username and password in case the user chooses to log in
FR03	The application checks the registered credentials for security and to enable the user to log in. Also, he could reset his/her password through sending link to his/her email
FR04	The application will show the signup screen if the user chooses to sign up showing some required fields to fill in such as name, username, password, email, phone number, country, city, National ID, Bank account number and Different investments.
FR05	The application will redirect you to your dashboard which consists of: 1. **Some computations of net-worth and rate of investment** 2. **Some graphs and visualizations** displaying how the investment flows and how much more do I need to reach the specified financial target. 3. **The financial obligations** that must be settled

Functional Requirements

Requirement ID	Requirement Statement
FR01	The First screen appears to the user is the login/sign-up screen
FR02	The application will redirect to the screen showing to two input boxes for the username and password in case the user chooses to log in
FR03	The application checks the registered credentials for security and to enable the user to log in. Also, he could reset his/her password through sending link to his/her email
FR04	The application will show the signup screen if the user chooses to sign up showing some required fields to fill in such as name, username, password, email, phone number, country, city, National ID, Bank account number and Different investments.
FR05	The application will redirect you to your dashboard which consists of: 1. **Some computations of net-worth and rate of investment** 2. **Some graphs and visualizations** displaying how the investment flows and how much more do I need to reach the specified financial target. 3. **The financial obligations** that must be settled

Usability	• The platform should be easy to use with a clear dashboard and real-time graphs. • The system must work seamlessly across mobile, tablets, and desktops, adapting to screen sizes dynamically.
Portability and compatibility	• The application should work on Windows, macOS, Linux, iOS, and Android without performance issues. • App should support for multiple payment methods • App should be multilingual, as it should be available in Arabic, English, and other regional languages.


Sign-up
Verfiy Credintals
Authentication
Login
Verfiy password
Reset password
Dash Board
Show the done operations and computations of net-worth and rate of investments
Show total funds and finicial obligations
Visualizations and graphs for income and exchanges
Auto-tracking for last updates
Control Pannel
Adding new investments sources
Suggestions for enlargement
Setting Targets
Change some user information
Setting Retirement Target
Tutorial for investment
Identify risks
Request assistance in managing saving process
Connect experts for asking
Investors
Families
Finicial advisors
Institutional Investor
Other systems
<<include>>
<<extend>>


estment
Upgrade
Login
Sign Up
Email
Password
Start
Enter the code
Theme
Upload Photo
Upload Photo
Upload Photo