Day 20

Employee ID: 201933938

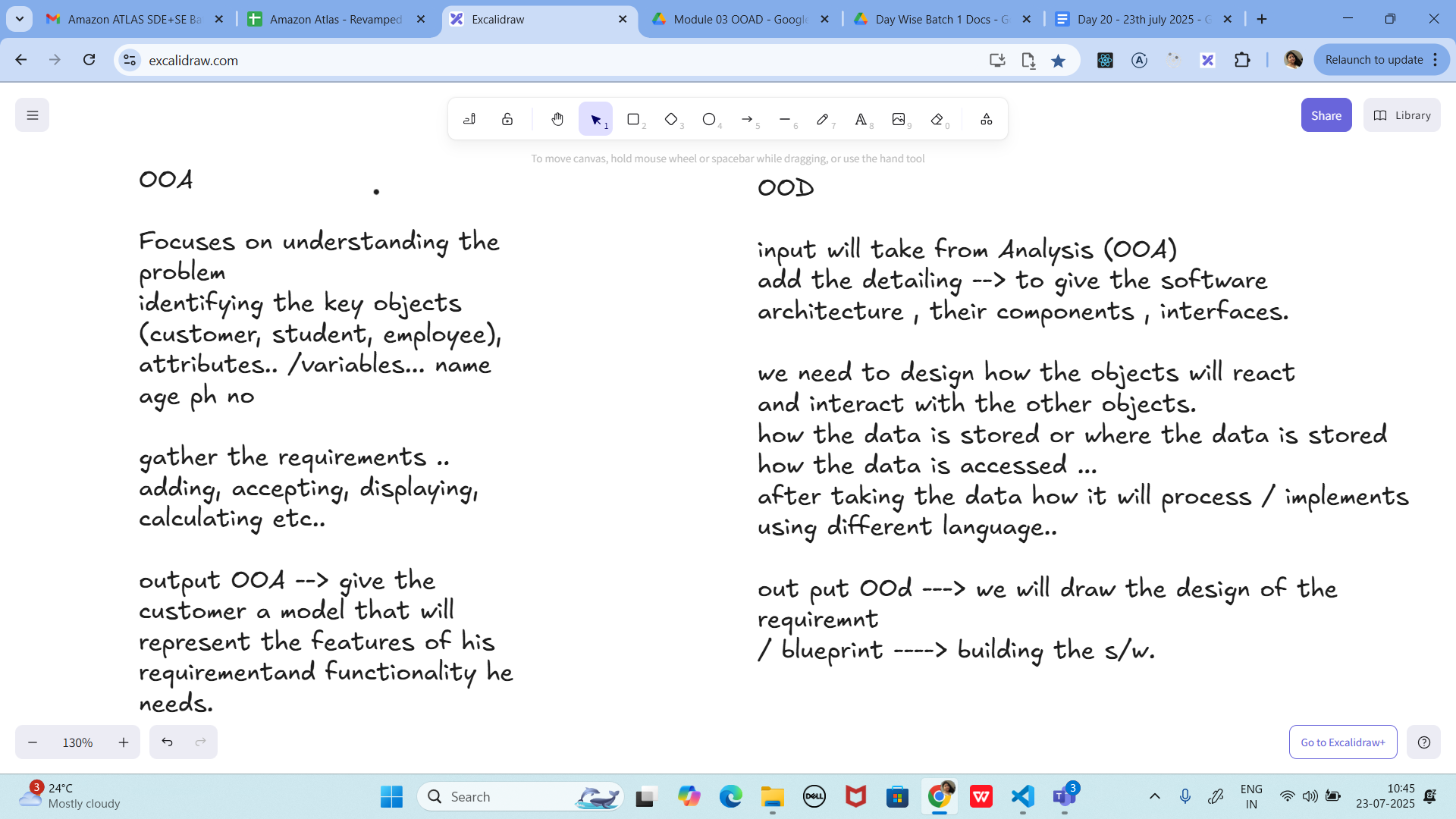
Login ID: iamasif

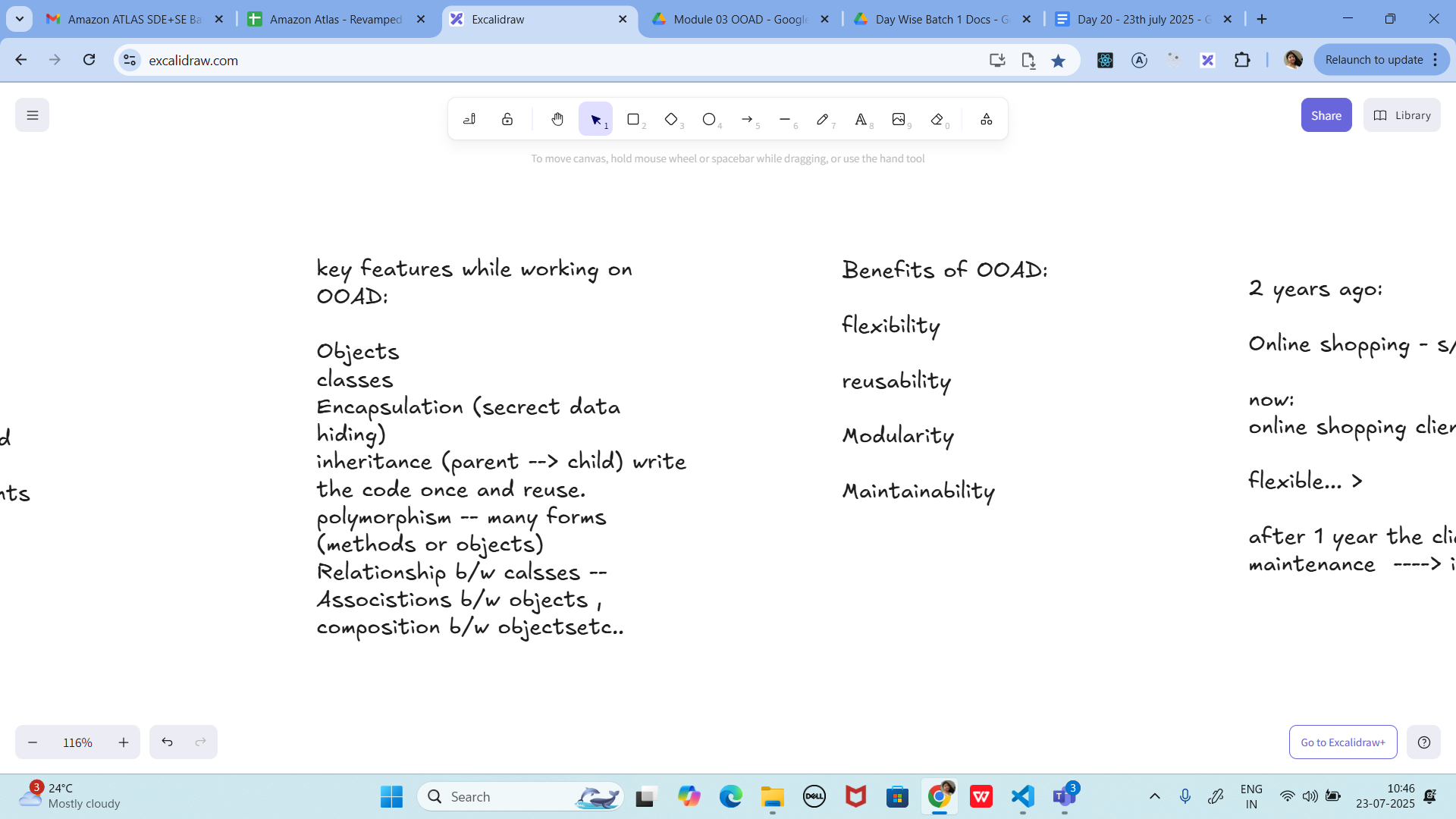
Name: Shaik Asif

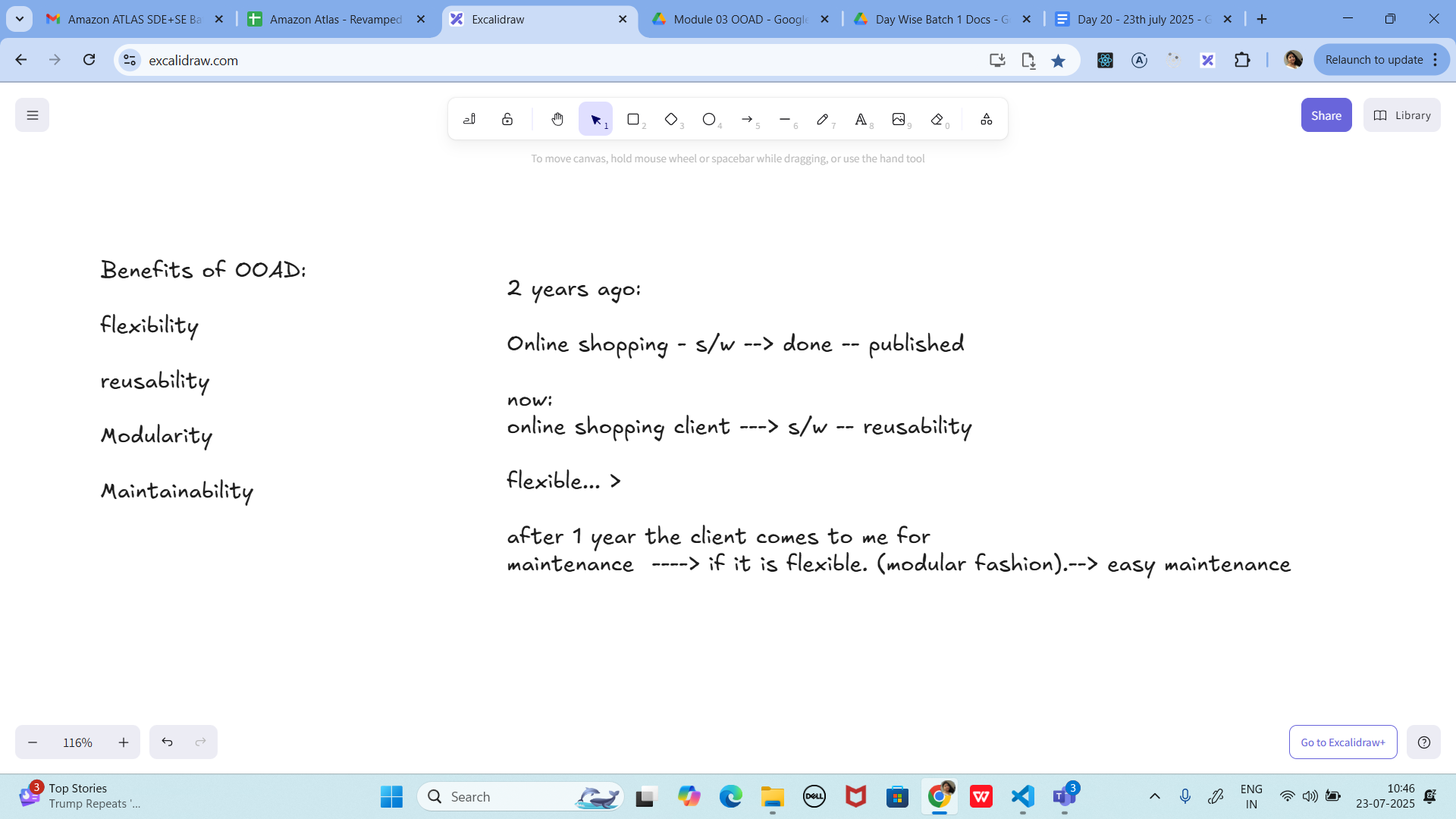
OOAD

Object Oriented paradigm, Design Patterns, Design Workflow, Refactoring

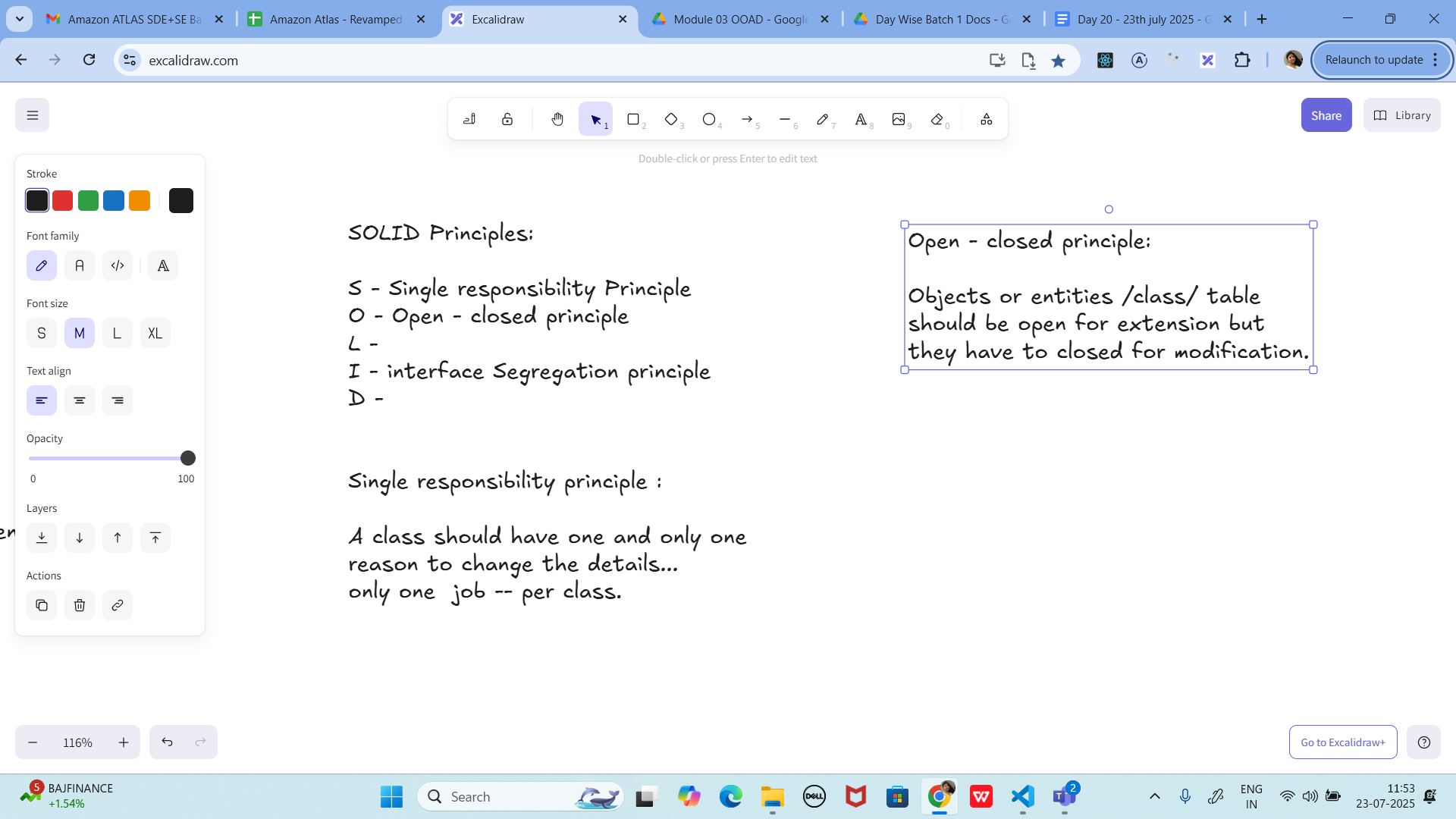
OOAD 👍

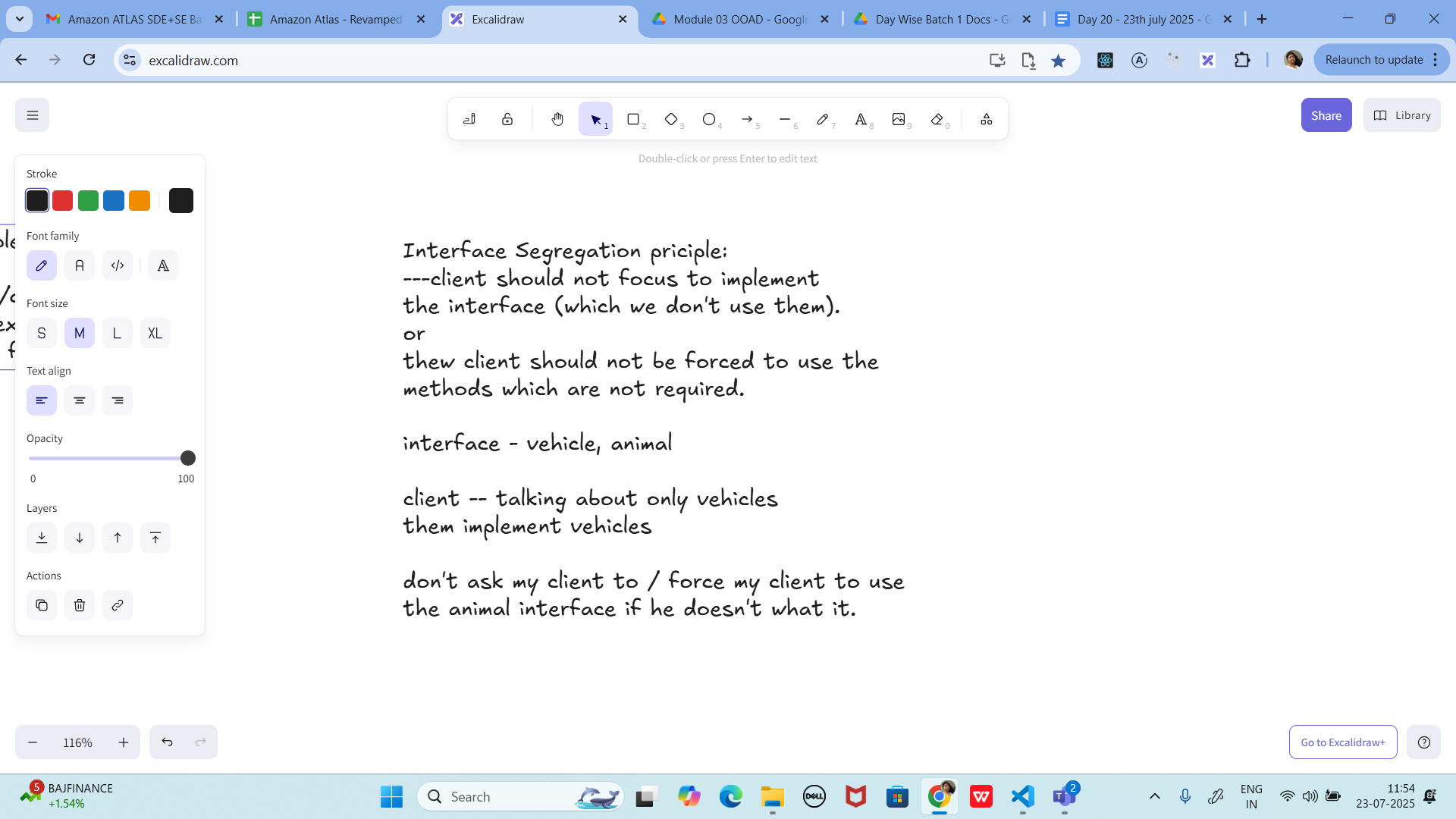






SOLID principles





Task 1:

Wap to create a class Student.. With 3 methods

registrationDetails(), marksCalc(), feesCalc().

3 min 1

**PROGRAM:**

**class Student {**

**// Attributes (fields)**

**String name;**

**int rollNo;**

**int[] marks;**

**double feesPerSubject;**

**// Constructor**

**Student(String name, int rollNo, int[] marks, double feesPerSubject) {**

**this.name = name;**

**this.rollNo = rollNo;**

**this.marks = marks;**

**this.feesPerSubject = feesPerSubject;**

**}**

**// 1. Show registration details**

**void registrationDetails() {**

**System.*out*.println("---- Registration Details ----");**

**System.*out*.println("Name: " + name);**

**System.*out*.println("Roll No: " + rollNo);**

**}**

**// 2. Calculate marks (total + average)**

**void marksCalc() {**

**int total = 0;**

**for (int m : marks) {**

**total += m;**

**}**

**double average = (double) total / marks.length;**

**System.*out*.println("Total Marks: " + total);**

**System.*out*.println("Average Marks: " + average);**

**}**

**// 3. Calculate fees**

**void feesCalc() {**

**double totalFees = marks.length \* feesPerSubject;**

**System.*out*.println("Total Fees: ₹" + totalFees);**

**}**

**}**

**public class Task1 {**

**public static void main(String[] args) {**

**// Sample data**

**int[] marks = {85, 90, 78, 92, 88};**

**// Create Student object**

**Student s1 = new Student("Asif", 101, marks, 5000);**

**// Call methods**

**s1.registrationDetails();**

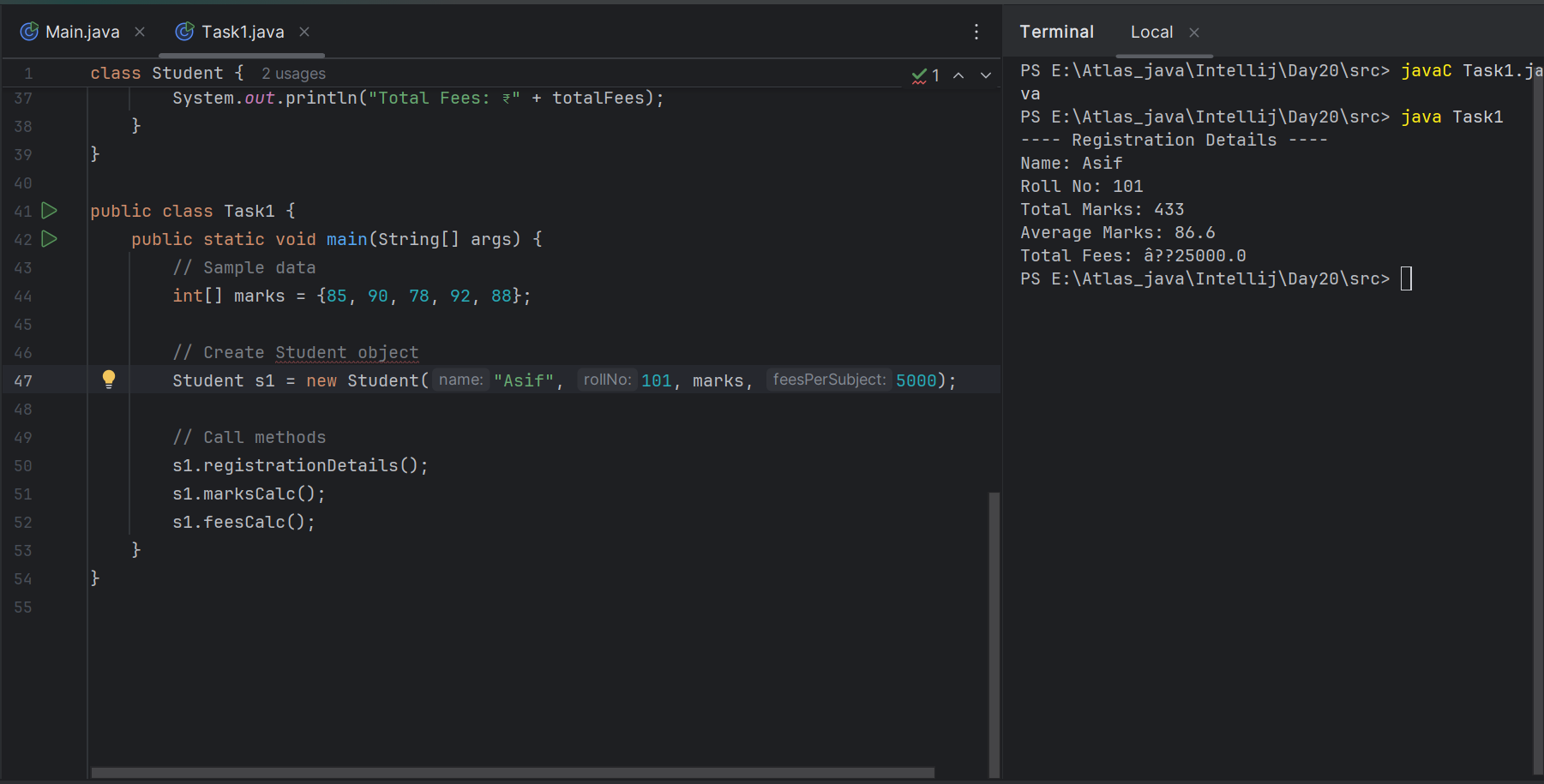
**s1.marksCalc();**

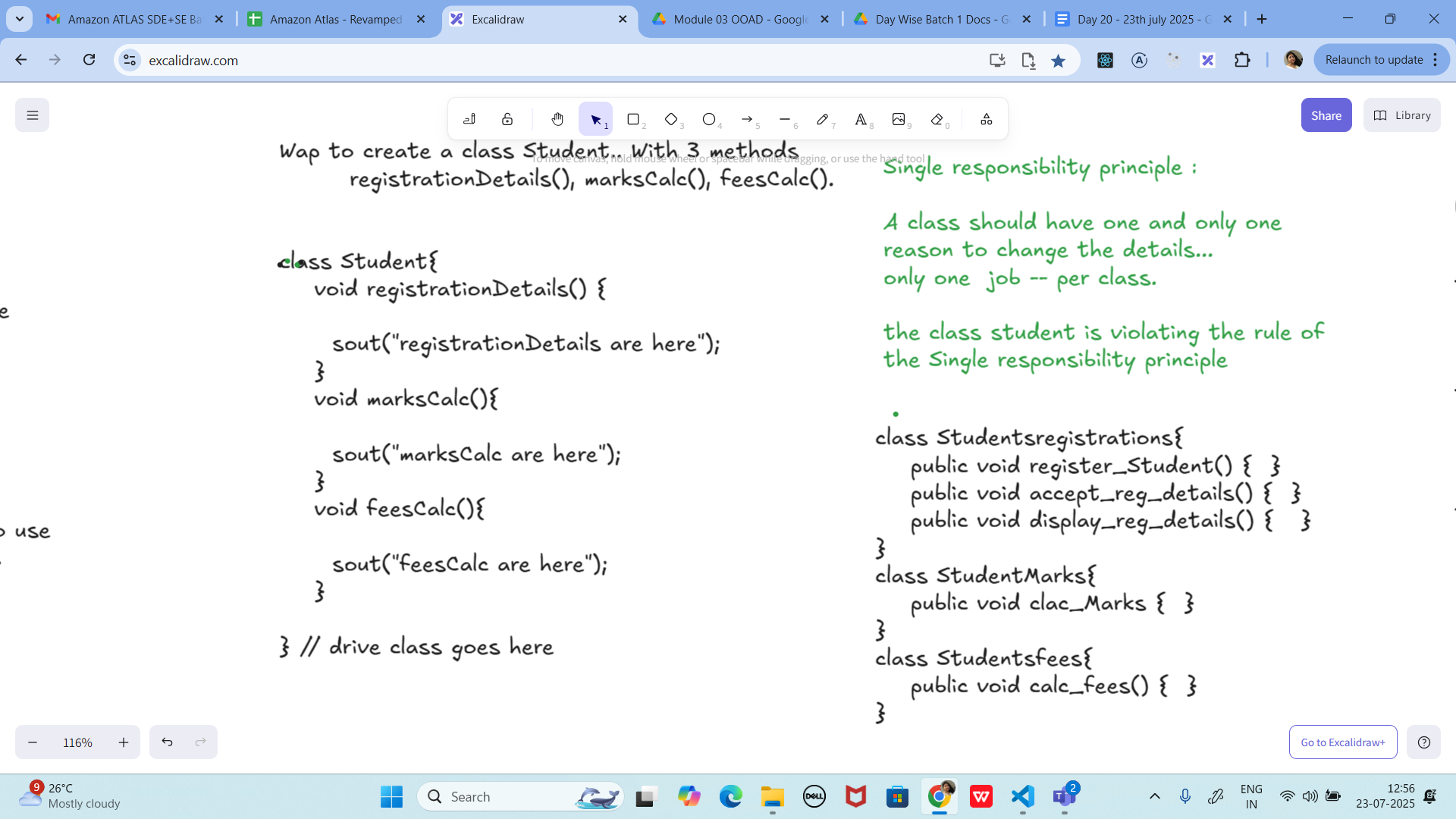
**s1.feesCalc();**

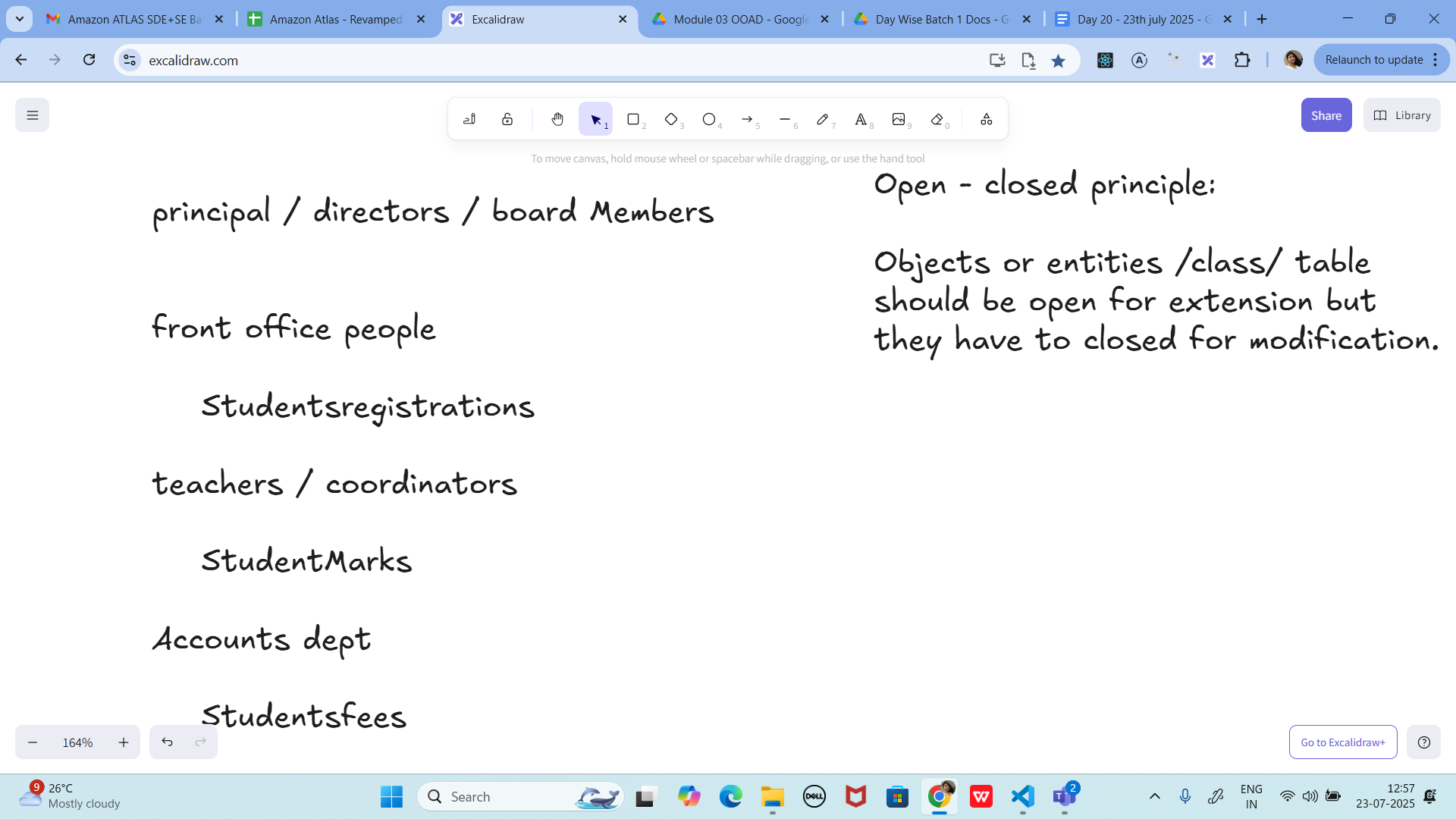
**}**

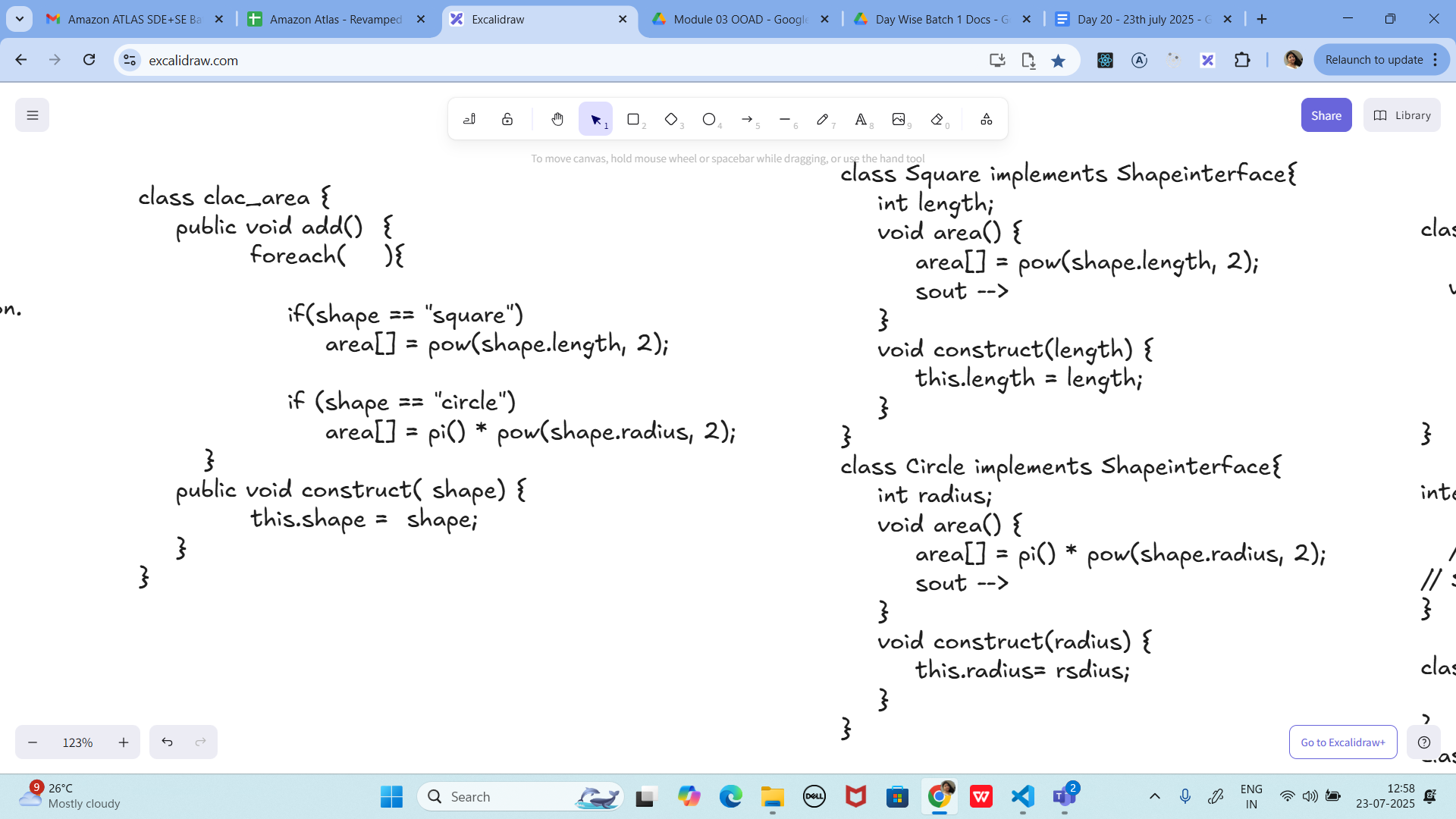
**}**

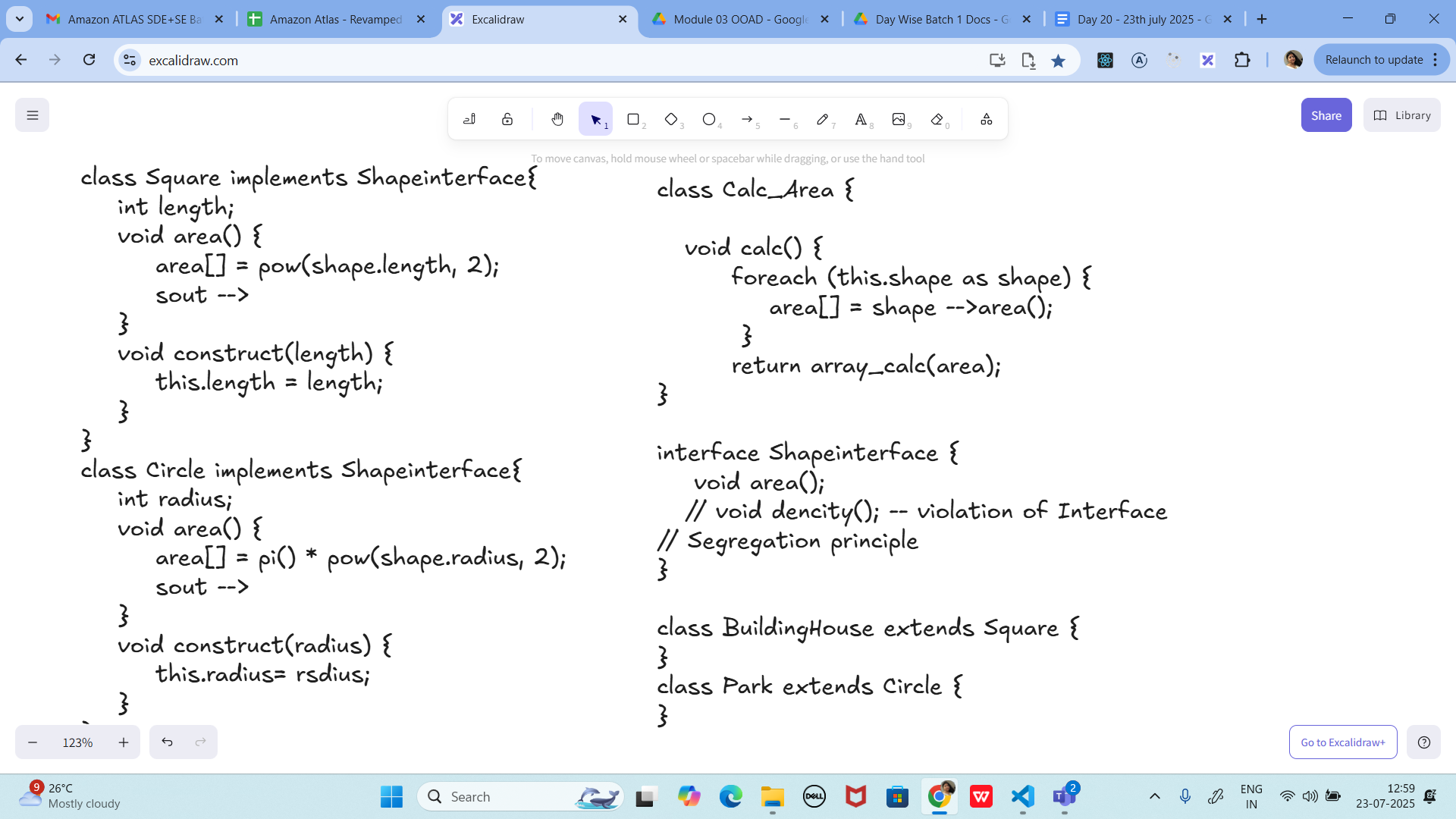
**OUTPUT:**











Task 02:

class Employee {

private String name;

private String email;

private double salary;

// Methods related to employee data

// Method to generate PDF report

public void generatePdfReport() {

// Code to generate PDF report

}

// Method to send email

public void sendEmail() {

// Code to send email

}

}

**PROGRAM:**

**// Employee class: only holds employee data**

**class Employee {**

**private String name;**

**private String email;**

**private double salary;**

**public Employee(String name, String email, double salary) {**

**this.name = name;**

**this.email = email;**

**this.salary = salary;**

**}**

**public String getName() { return name; }**

**public String getEmail() { return email; }**

**public double getSalary() { return salary; }**

**}**

**// Class responsible for generating PDF reports**

**class PdfReportGenerator {**

**public void generate(Employee emp) {**

**System.*out*.println("Generating PDF report...");**

**System.*out*.println("Employee Name: " + emp.getName());**

**System.*out*.println("Employee Email: " + emp.getEmail());**

**System.*out*.println("Employee Salary: " + emp.getSalary());**

**System.*out*.println("PDF report created successfully!\n");**

**}**

**}**

**// Class responsible for sending emails**

**class EmailService {**

**public void sendEmail(Employee emp) {**

**System.*out*.println("Sending email to " + emp.getEmail());**

**System.*out*.println("Hello " + emp.getName() + ", your salary is " + emp.getSalary());**

**System.*out*.println("Email sent successfully!\n");**

**}**

**}**

**// Main class**

**public class Task2 {**

**public static void main(String[] args) {**

**// Create an employee**

**Employee emp = new Employee("Shaik Asif", "asifshannu.sk@gmail.com", 25000);**

**// Generate PDF report**

**PdfReportGenerator pdfGen = new PdfReportGenerator();**

**pdfGen.generate(emp);**

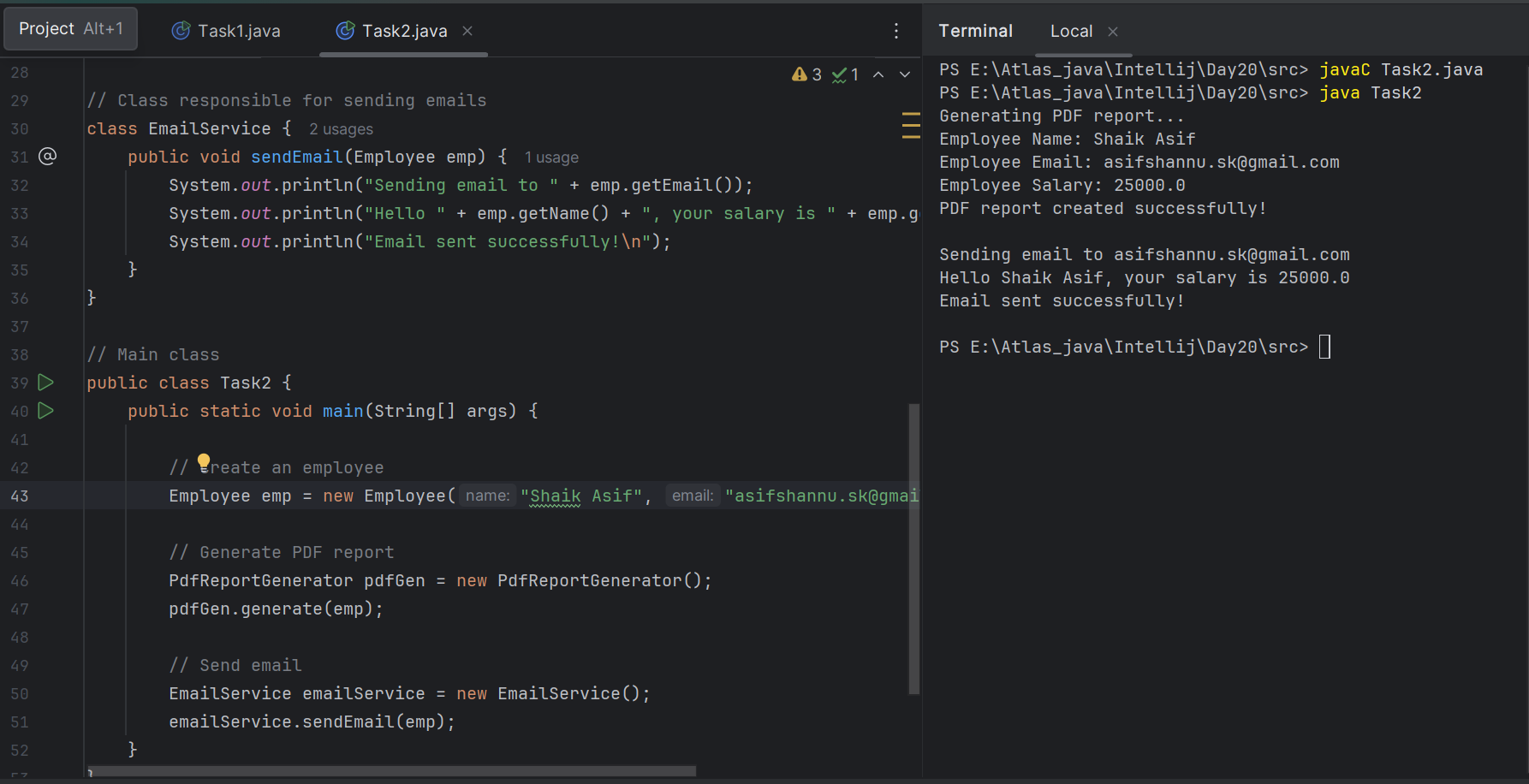
**// Send email**

**EmailService emailService = new EmailService();**

**emailService.sendEmail(emp);**

**}**

**}**

**OUTPUT:**

In the above example code, the Employee class violates the SRP because it has multiple responsibilities: managing employee data, generating PDF reports, and sending emails. These responsibilities are not cohesive and may change for different reasons.

Task 03:

class PaymentProcessor {

public void processPayment(String paymentType, double amount) {

if (paymentType.equals("credit card")) {

// Process credit card payment

} else if (paymentType.equals("paypal")) {

// Process PayPal payment

} else if (paymentType.equals("bitcoin")) {

// Process Bitcoin payment

}

// ... more payment types

}

}

**PROGRAM:**

// 1. PaymentMethod interface

interface PaymentMethod {

void pay(double amount);

}

// 2. Different payment implementations

class CreditCardPayment implements PaymentMethod {

public void pay(double amount) {

System.*out*.println("Processing credit card payment of rs " + amount);

}

}

class PayPalPayment implements PaymentMethod {

public void pay(double amount) {

System.*out*.println("Processing PayPal payment of rs " + amount);

}

}

class BitcoinPayment implements PaymentMethod {

public void pay(double amount) {

System.*out*.println("Processing Bitcoin payment of rs " + amount);

}

}

// 3. PaymentProcessor is now OPEN for extension (no if-else)

class PaymentProcessor {

public void processPayment(PaymentMethod paymentMethod, double amount) {

paymentMethod.pay(amount); // Polymorphism in action

}

}

// 4. Main class

public class Task3 {

public static void main(String[] args) {

PaymentProcessor processor = new PaymentProcessor();

// Process different payments without modifying PaymentProcessor

processor.processPayment(new CreditCardPayment(), 100.0);

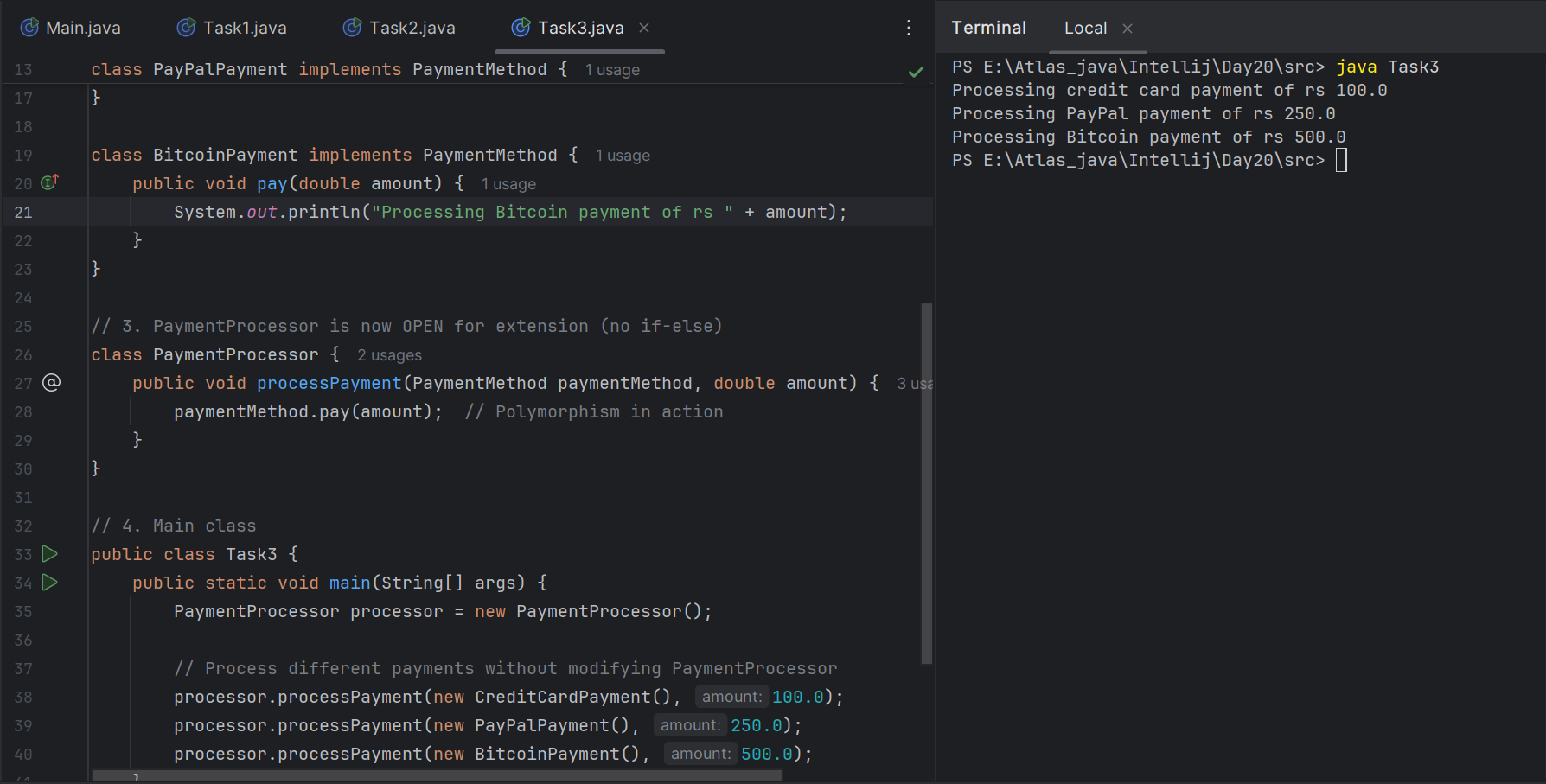
processor.processPayment(new PayPalPayment(), 250.0);

processor.processPayment(new BitcoinPayment(), 500.0);

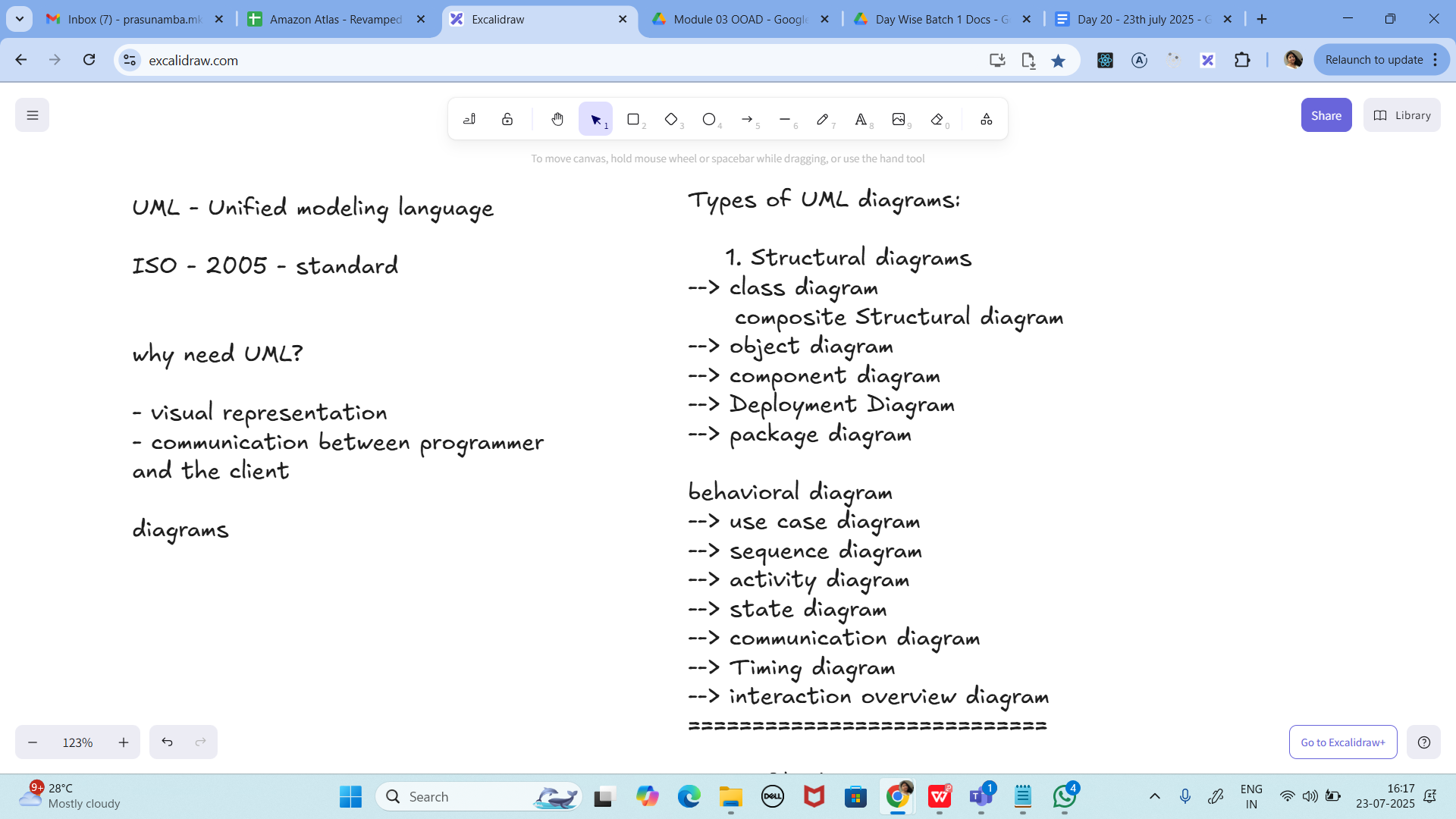
}

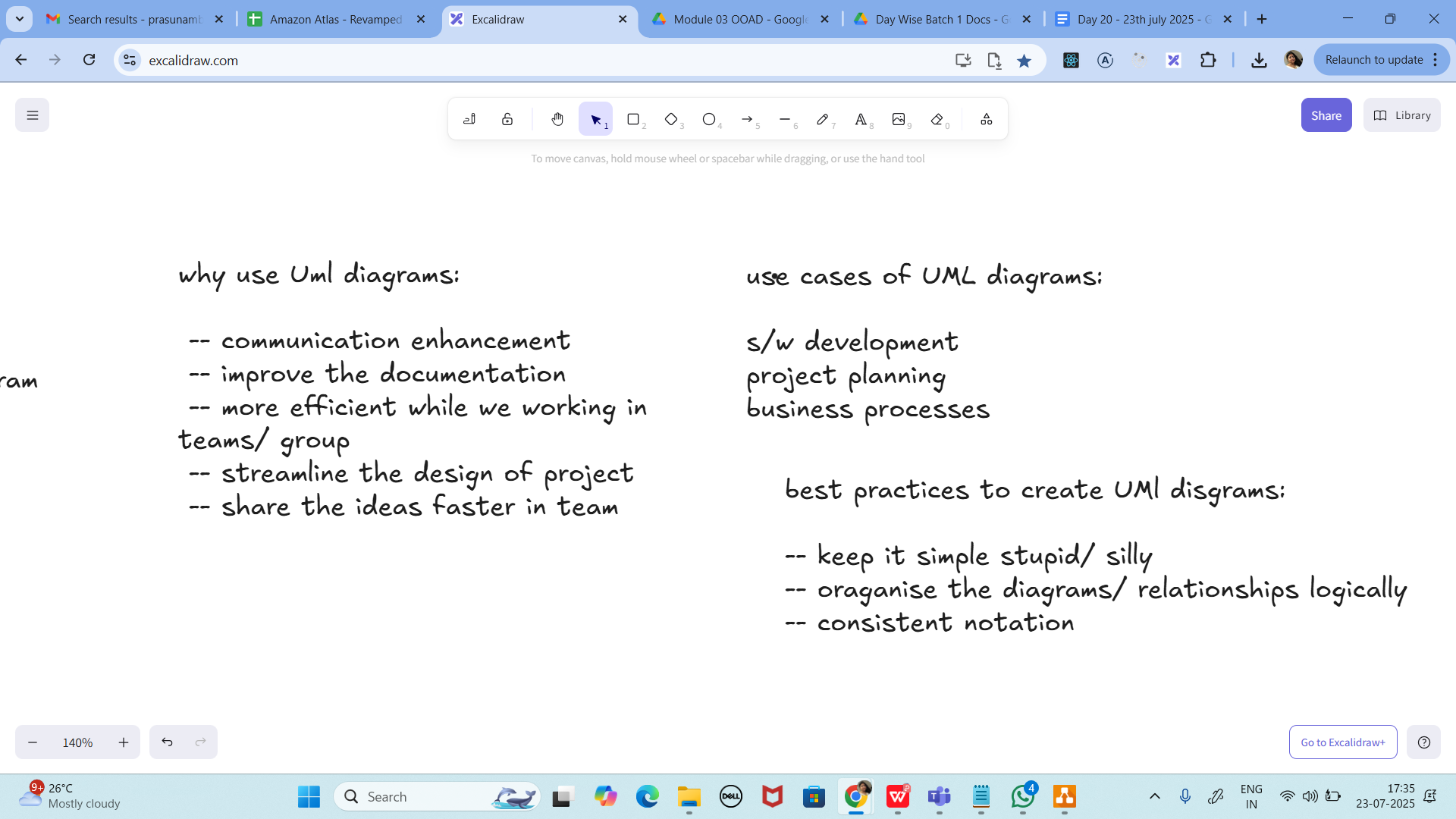
}

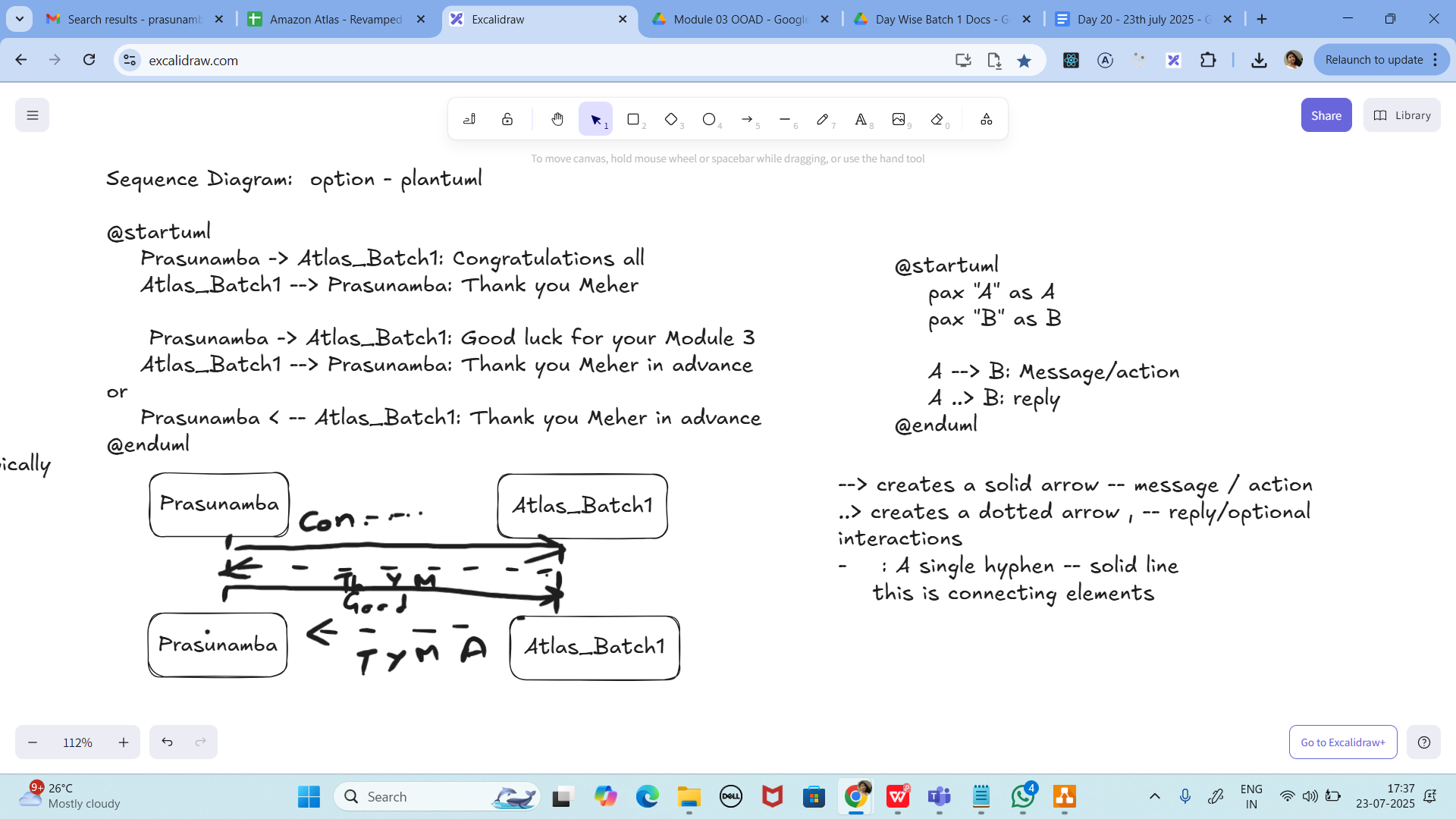
**OUTPUT:**



UML - unified Modeling Language::







Task04:

Sequence diagram:

@startuml

Prasunamba -> Atlas\_Batch1: Congratulations all

Atlas\_Batch1 --> Prasunamba: Thank you Meher

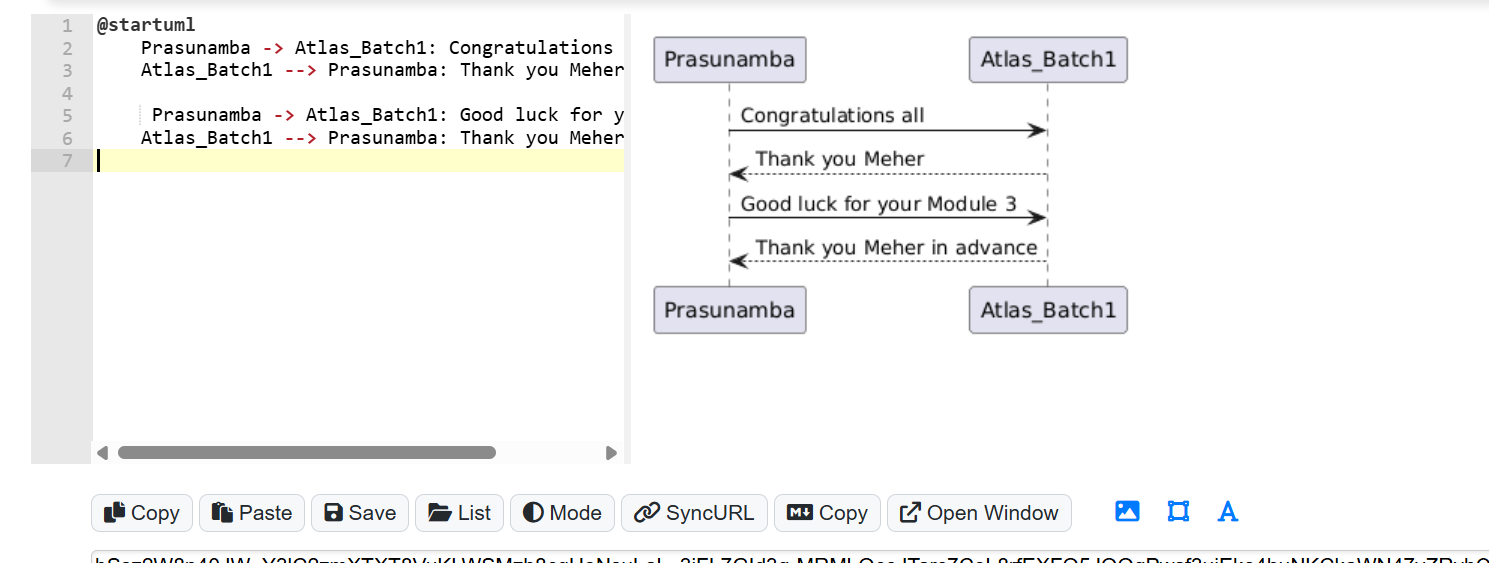
Prasunamba -> Atlas\_Batch1: Good luck for your Module 3

Atlas\_Batch1 --> Prasunamba: Thank you Meher in advance

or

Prasunamba < -- Atlas\_Batch1: Thank you Meher in advance

@enduml



Task 5:

@startuml

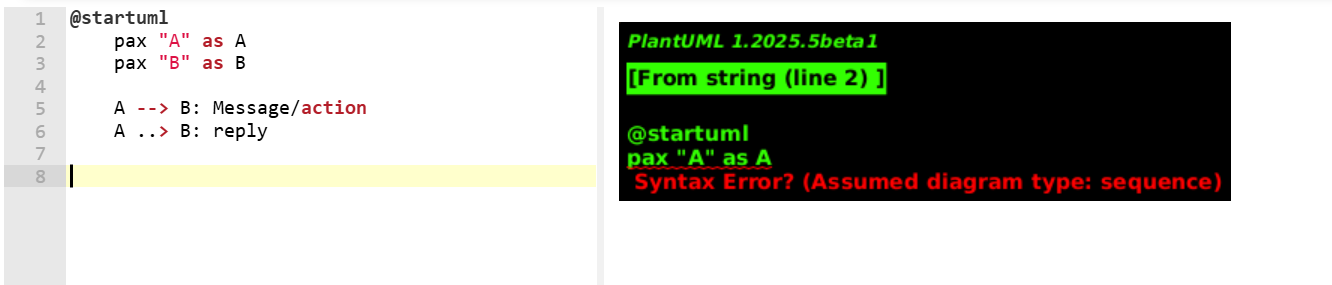
pax "A" as A

pax "B" as B

A --> B: Message/action

A ..> B: reply

@enduml



Class Diagram:

@startuml

class Car

Driver - Car : drives >

Car \*- Wheels : have 4 >

Car -- Person : < owns

@enduml

