

SE0701_2311104003_MARTRYATUS SOFIA

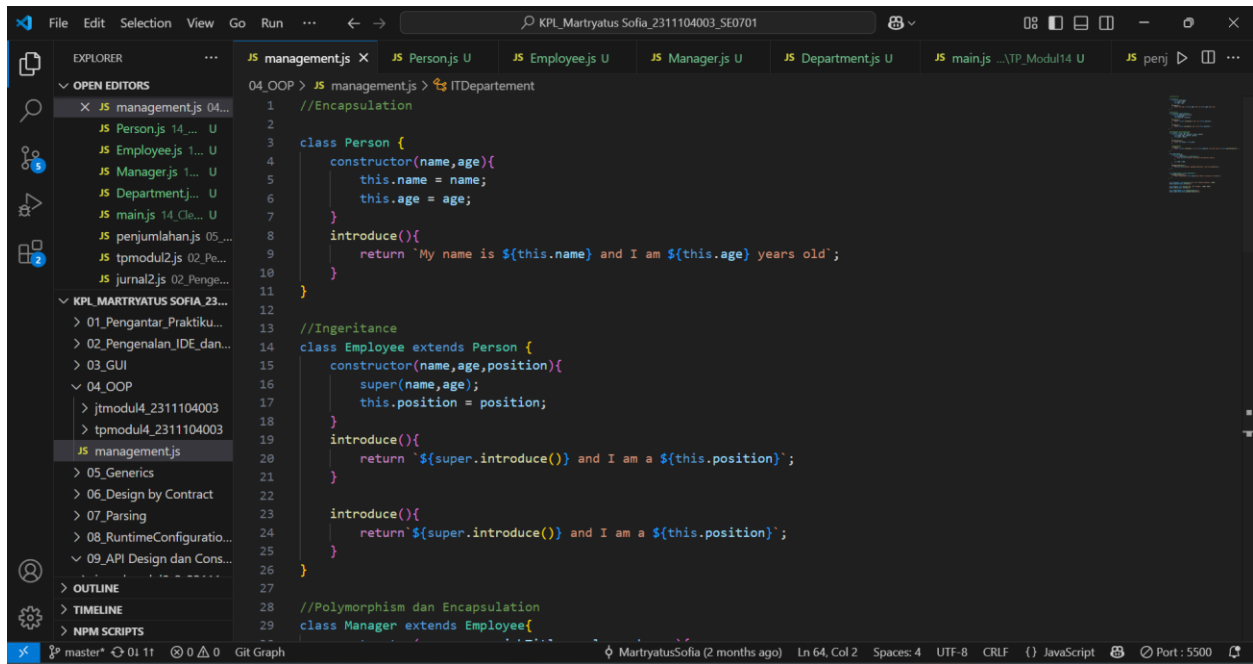
TP MODUL 14

Link Github :

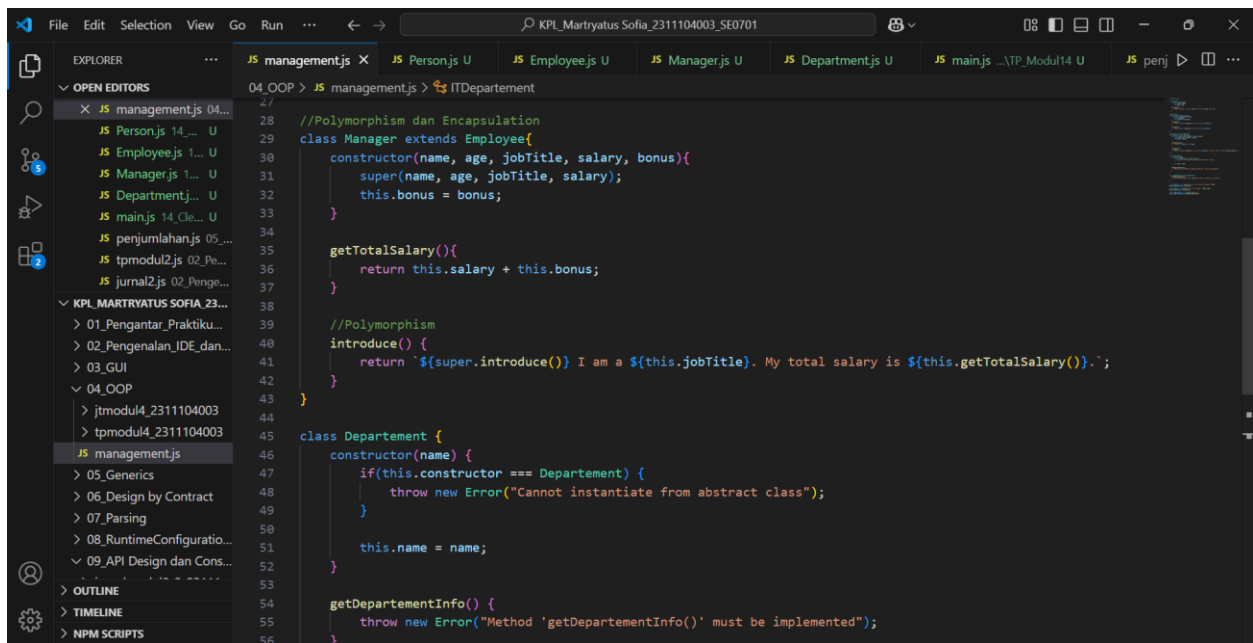
https://github.com/MartryatusSofia/KPL_Martryatus-Sofia_2311104003_SE0701/tree/master/14_Clean_Code/TP_Modul14

Penjelasan Code :

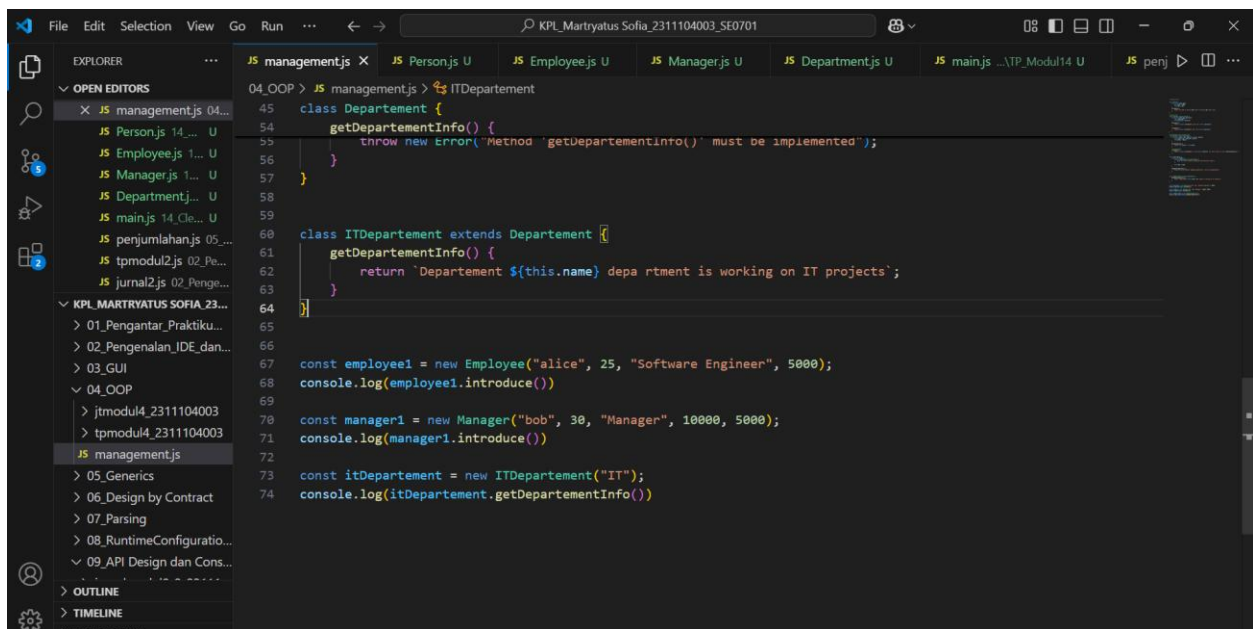
Modul 2 – Code yang belum rapih



```
04_OOP > JS management.js > ITDepartement
1 //Encapsulation
2
3 class Person {
4   constructor(name,age){
5     this.name = name;
6     this.age = age;
7   }
8   introduce(){
9     return `My name is ${this.name} and I am ${this.age} years old`;
10  }
11 }
12
13 //Ingeritance
14 class Employee extends Person {
15   constructor(name,age,position){
16     super(name,age);
17     this.position = position;
18   }
19   introduce(){
20     return `${super.introduce()} and I am a ${this.position}`;
21   }
22   introduce(){
23     return `${super.introduce()} and I am a ${this.position}`;
24   }
25 }
26
27
28 //Polymorphism dan Encapsulation
29 class Manager extends Employee{
30 }
```



```
27 //Polymorphism dan Encapsulation
28
29 class Manager extends Employee{
30     constructor(name, age, jobTitle, salary, bonus){
31         super(name, age, jobTitle, salary);
32         this.bonus = bonus;
33     }
34
35     getTotalSalary(){
36         return this.salary + this.bonus;
37     }
38
39     //Polymorphism
40     introduce() {
41         return `${super.introduce()} I am a ${this.jobTitle}. My total salary is ${this.getTotalSalary()}`;
42     }
43 }
44
45 class Departement {
46     constructor(name) {
47         if(this.constructor === Departement) {
48             throw new Error("Cannot instantiate from abstract class");
49         }
50
51         this.name = name;
52     }
53
54     getDepartementInfo() {
55         throw new Error("Method 'getDepartementInfo()' must be implemented");
56     }
57 }
```



```
45 class Departement {
46     getDepartementInfo() {
47         throw new Error("Method 'getDepartementInfo()' must be implemented");
48     }
49 }
50
51 class ITDepartement extends Departement {
52     getDepartementInfo() {
53         return `Departement ${this.name} depa rtment is working on IT projects`;
54     }
55 }
56
57 const employee1 = new Employee("alice", 25, "Software Engineer", 5000);
58 console.log(employee1.introduce())
59
60 const manager1 = new Manager("bob", 30, "Manager", 10000, 5000);
61 console.log(manager1.introduce())
62
63 const itDepartement = new ITDepartement("IT");
64 console.log(itDepartement.getDepartementInfo())
65
66
67
68
69
70
71
72
73
74
```

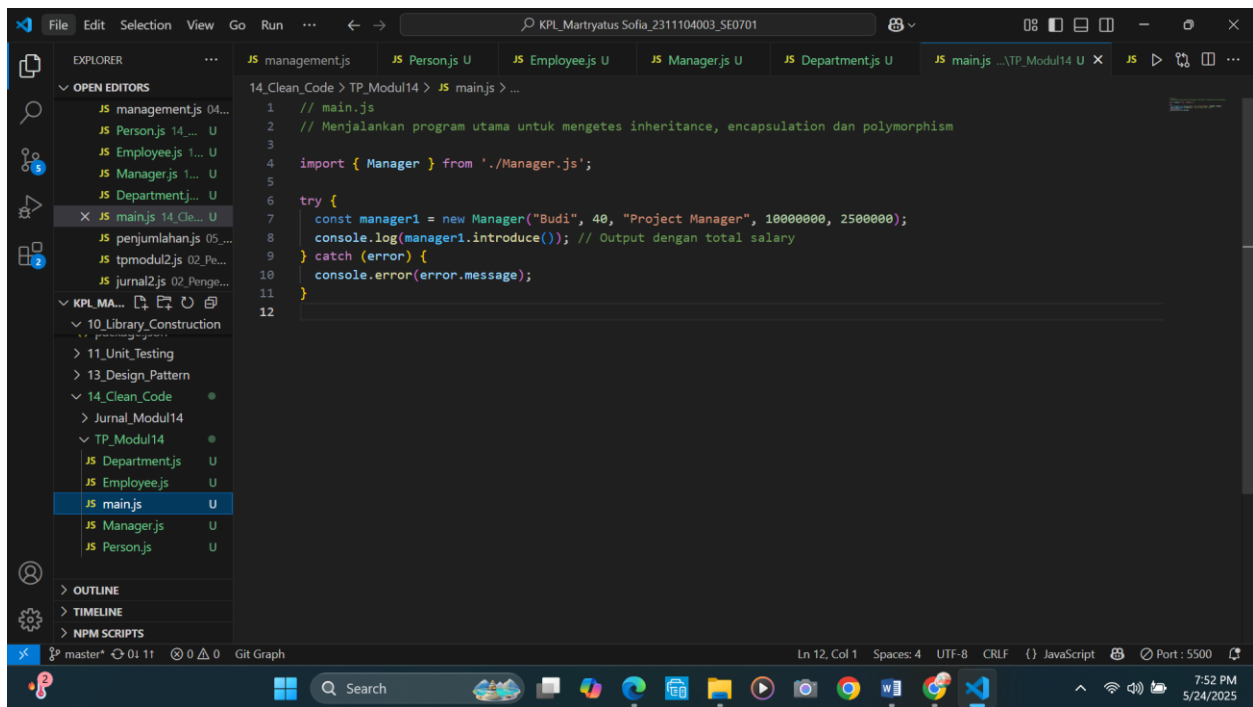
Code yang sudah di rapihkan

```
1 // Department.js
2 // Contoh penggunaan abstract class dengan metode abstrak
3
4 class Department {
5   constructor(name) {
6     if (this.constructor === Department) {
7       throw new Error("Cannot instantiate from abstract class");
8     }
9     this.name = name;
10  }
11
12  // Harus dioverride di subclass
13  getDepartmentInfo() {
14    throw new Error("Method 'getDepartmentInfo()' must be implemented.");
15  }
16
17  export { Department };
18
19
```

```
1 import { Person } from './Person.js';
2
3 export class Employee extends Person {
4   constructor(name, age, position) {
5     super(name, age);
6     this.position = position;
7   }
8
9   introduce() {
10     return `${super.introduce()} and I am a ${this.position}.`;
11   }
12
13
```

```
14_Clean_Code > TP_Modul14 > JS Managerjs > Manager > constructor
1 // Manager.js
2 // Class turunan dari Employee dengan tambahan atribut bonus dan salary (Polymorphism + Encapsulation)
3
4 import { Employee } from './Employee.js';
5
6 class Manager extends Employee {
7   constructor(name, age, jobTitle, salary, bonus) {
8     super(name, age, jobTitle);
9     this.salary = salary; //salary harus didefinisikan di sini
10    this.bonus = bonus;
11    this.jobTitle = jobTitle; // Untuk dipakai di introduce
12  }
13
14  // Mengembalikan total gaji
15  getTotalSalary() {
16    return this.salary + this.bonus;
17  }
18
19  // Override method introduce
20  introduce() {
21    return `${super.introduce()} I am a ${this.jobTitle}. My total salary is ${this.getTotalSalary()}`;
22  }
23 }
24
25 export { Manager };
26
```

```
14_Clean_Code > TP_Modul14 > JS Personjs > ...
1 // Person.js
2 // Class ini merepresentasikan orang dengan nama dan usia (Encapsulation)
3
4 class Person {
5   constructor(name, age) {
6     this.name = name;
7     this.age = age;
8   }
9
10  // Method untuk memperkenalkan diri
11  introduce() {
12    return `My name is ${this.name} and I am ${this.age} years old.`;
13  }
14
15  export { Person };
16
17
```



1. Naming convention

1) Variable / Property / Attribute

```
this.name = name;
this.age = age;
```

```
this.position = position;
```

```
this.salary = salary; //
this.bonus = bonus;
```

Sudah menggunakan **camelCase**.

Nama-nama variabel sudah deskriptif dan sesuai dengan fungsinya.

2) Method / Function / Procedure

```
// Mengembalikan total gaji
getTotalSalary() {
    return this.salary + this.bonus;
}

// Override method introduce
introduce() {
    return `${super.introduce()} I am a ${this.jobTitle}. My total salary is ${this.getTotalSalary()}.`;
}
```

Nama method sudah sesuai konvensi camelCase dan deskriptif.

2. White space dan indentation

```
introduce() {  
    return `${super.introduce()} and I am a ${this.position}.`;  
}
```

secara umum indentasi (tab/spasi) sudah rapi

3. Variable / attribute declarations

```
class Manager extends Employee {  
    constructor(name, age, jobTitle, salary, bonus) {  
        super(name, age, jobTitle);  
        this.salary = salary; //salary harus didefinisikan di sini  
        this.bonus = bonus;  
        this.jobTitle = jobTitle; // Untuk dipakai di introduce  
    }  
}
```

4. Comments

```
// Manager.js  
// Class turunan dari Employee dengan tambahan atribut bonus dan salary (Polymorphism + Encapsulation)
```

Hasil Running

```
PS D:\Kuliah\SEMESTER 4\PRAKTIKUM KPL\KPL_Martryatus Sofia_2311104003_SE0701\14_Clean_Code\TP_Modul14> node main.js  
My name is Sofia and I am 25 years old. and I am a Project Manager. My total salary is 10000000.  
PS D:\Kuliah\SEMESTER 4\PRAKTIKUM KPL\KPL_Martryatus Sofia_2311104003_SE0701\14_Clean_Code\TP_Modul14> 
```