

Laporan Praktikum Pemograman Berorientasi Objek

Praktikum 1



Oleh :

Putri Balqis Afradinata

NIM 2411531009

Mata Kuliah Pemograman Berorientasi Objek

Dosen Pengampu: Nurfiah, S.ST, M.Kom

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

PADANG, 2025

## **i. PENDAHULUAN**

Pemrograman Berorientasi Objek (PBO) merupakan paradigma pemrograman yang berfokus pada objek sebagai inti dari pengembangan aplikasi. Objek dibentuk dari class yang berfungsi sebagai blueprint atau rancangan, sehingga dapat merepresentasikan entitas nyata dalam sebuah sistem.

Dalam praktik sehari-hari, PBO sangat membantu untuk membangun aplikasi yang modular, terstruktur, dan mudah dikembangkan. Salah satu contoh penerapan PBO adalah pada aplikasi LaundryApps, di mana terdapat berbagai entitas yang perlu direpresentasikan dalam bentuk class, seperti User, Costumer, Service, dan Order.

Pada praktikum ini, mahasiswa diminta untuk memahami cara membuat class, object, constructor, dan method, serta memanfaatkan konsep encapsulation untuk melindungi data. Selain itu, mahasiswa juga mempraktikkan pembuatan tampilan antarmuka login dengan JFrame dan menghubungkannya ke halaman utama aplikasi.

### **Dasar Teori:**

- Class adalah blueprint atau template dari objek, berisi atribut (data member) dan method (perilaku).
- Object adalah representasi nyata dari class. Setiap objek memiliki state, behavior, dan identity.
- Constructor adalah method khusus untuk menginisialisasi objek, namanya sama dengan nama class.
- Method adalah blok kode yang menjalankan fungsi tertentu.

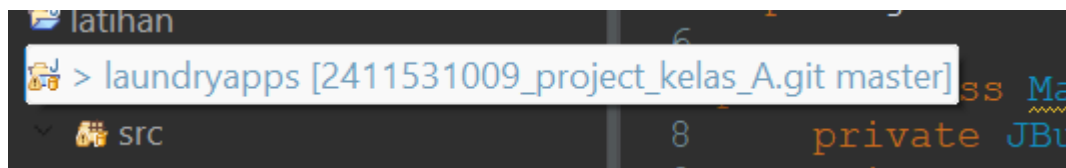
Ada beberapa jenis method di Java:

- Predefined Method → method bawaan Java (misalnya toString(), equals()).
- User-defined Method → dibuat programmer sesuai kebutuhan.
- Static Method → dapat dipanggil tanpa membuat objek.
- Instance Method → membutuhkan objek untuk dipanggil (getter & setter).
- Abstract Method → method tanpa body, dideklarasikan di abstract class.
- Factory Method → mengembalikan objek dari class tertentu.

Encapsulation adalah pembungkusan data agar atribut tidak bisa diakses langsung, melainkan melalui getter dan setter. Dalam praktikum ini konsep tersebut diterapkan pada aplikasi LaundryApps dengan package model untuk class, dan ui untuk tampilan (Login & MainFrame).

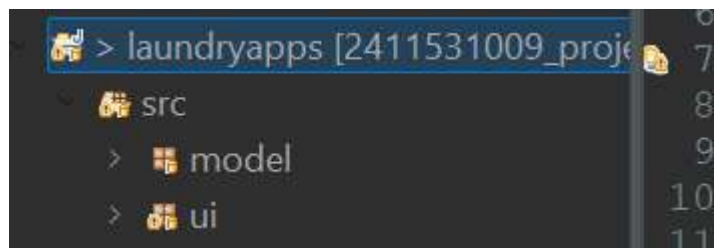
## ii. LANGKAH Pengerjaan

1. Membuat project baru bernama **laundryapps** di Eclipse.

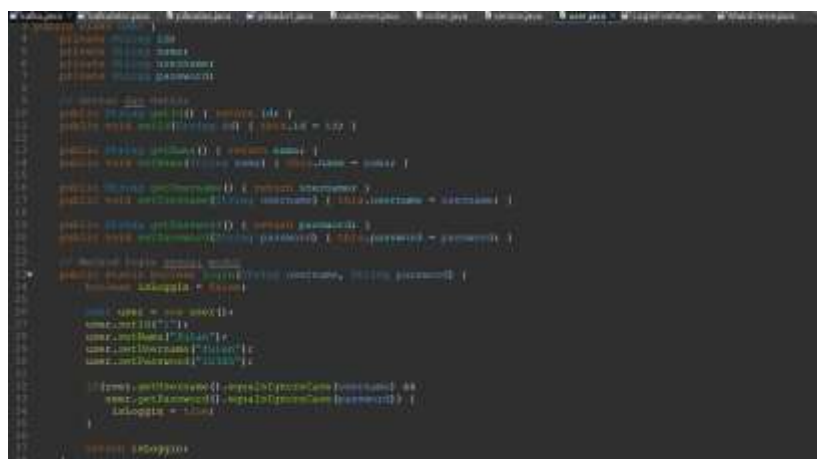


2. Membuat dua package:

- model → menyimpan class (User, Costumer, Service, Order).
- ui → menyimpan tampilan (LoginFrame, MainFrame).

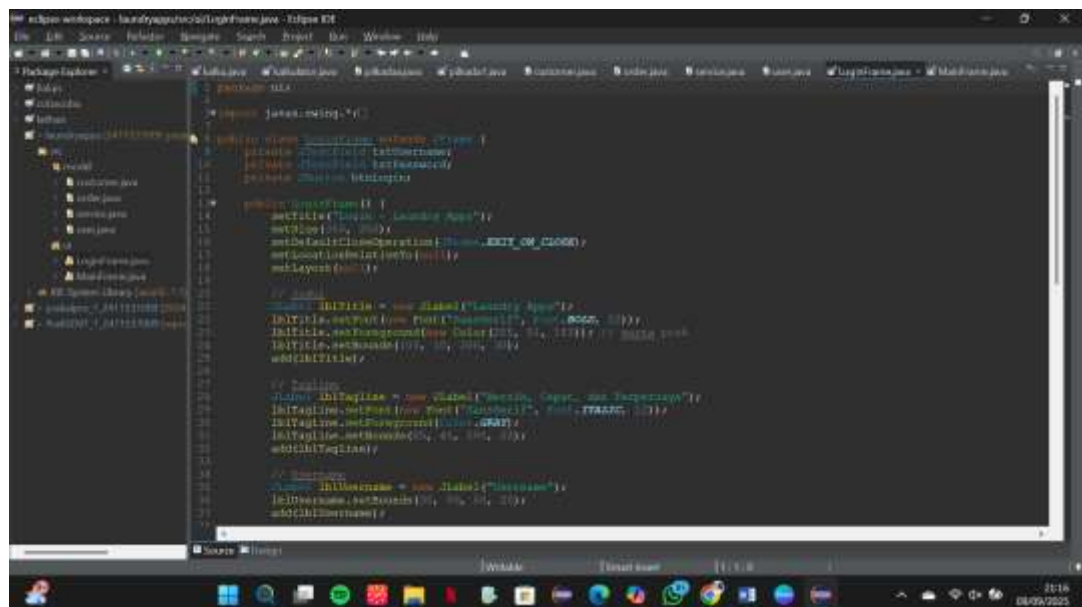


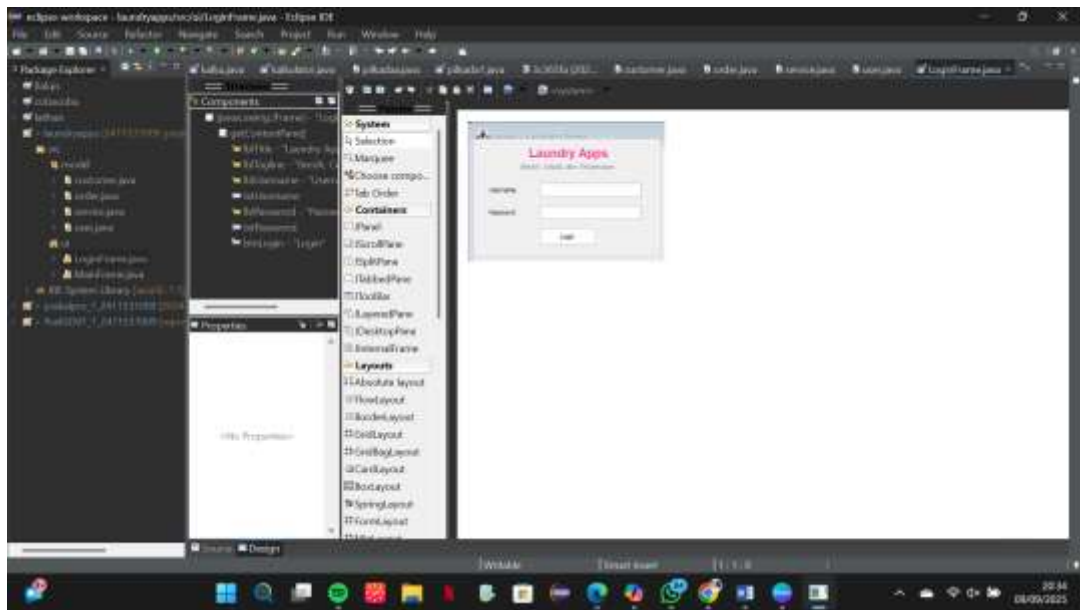
3. Membuat class User dengan atribut id, nama, username, password, lengkap dengan getter & setter, serta method login().



Kode tersebut merupakan implementasi class user pada Java. Di dalam class ini terdapat empat atribut utama yang dideklarasikan sebagai private, yaitu id, nama, username, dan password. Karena bersifat private, atribut-atribut tersebut tidak bisa diakses langsung dari luar class, sehingga disediakan setter dan getter untuk masing-masing atribut agar nilainya dapat diubah dan dibaca dengan aman sesuai prinsip encapsulation. Selanjutnya, terdapat sebuah method static bernama login() yang berfungsi untuk memverifikasi data login pengguna. Method ini menerima dua parameter, yaitu username dan password, lalu membuat objek user baru dengan data contoh: id = "1", nama = "fulan", username = "fulan", dan password = "12345". Setelah itu, dilakukan pengecekan menggunakan equalsIgnoreCase untuk membandingkan input username dan password dengan data pada objek. Jika cocok, variabel isLoggin diubah menjadi true, menandakan login berhasil; jika tidak cocok, maka tetap bernilai false. Pada akhir method, nilai isLoggin dikembalikan sebagai hasil proses login. Dengan cara ini, method login() dapat digunakan tanpa membuat objek baru karena bersifat static, dan langsung menentukan apakah kombinasi username dan password valid atau tidak.

4. Membuat LoginFrame dengan komponen txtUsername, txtPassword, btnLogin.
  - Event btnLogin memanggil method User.login().
  - Jika login benar → masuk ke MainFrame.

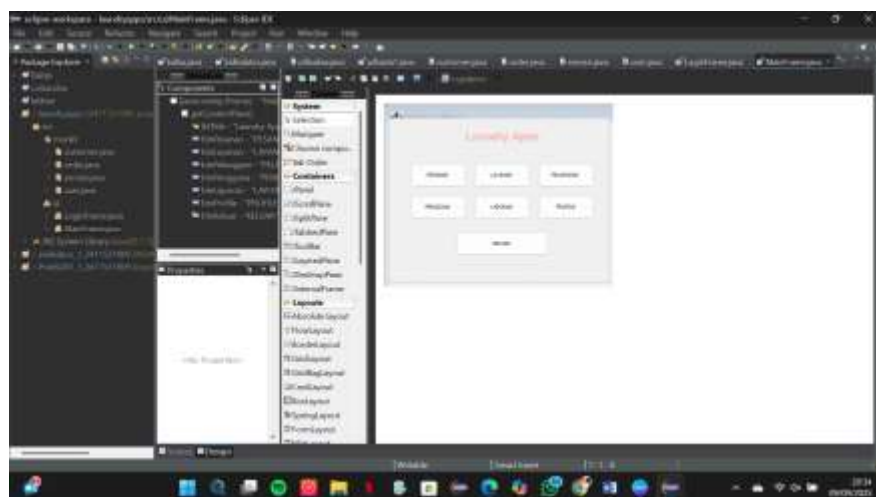
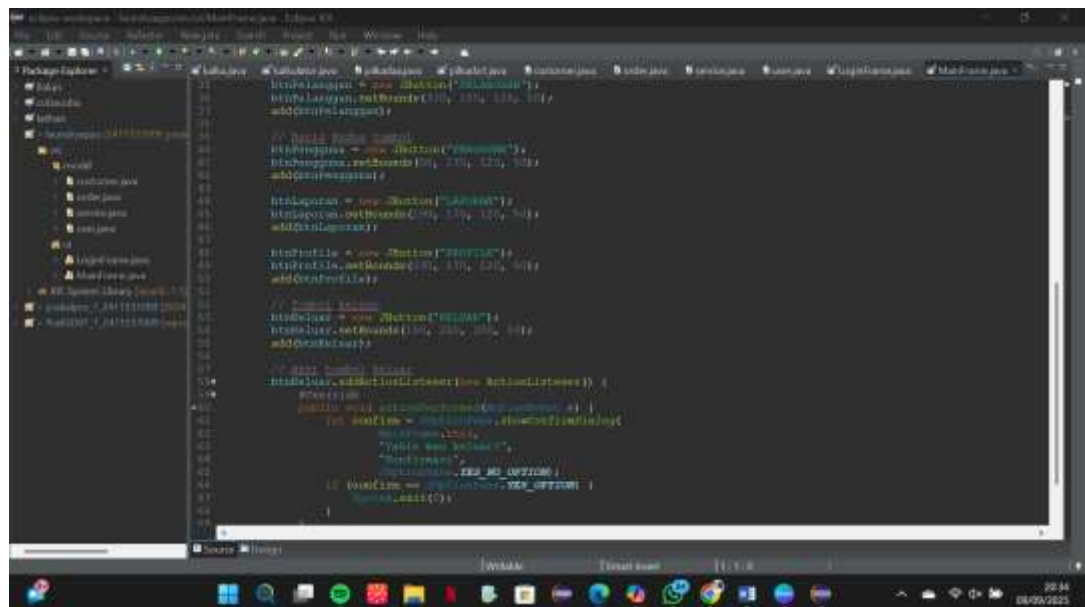
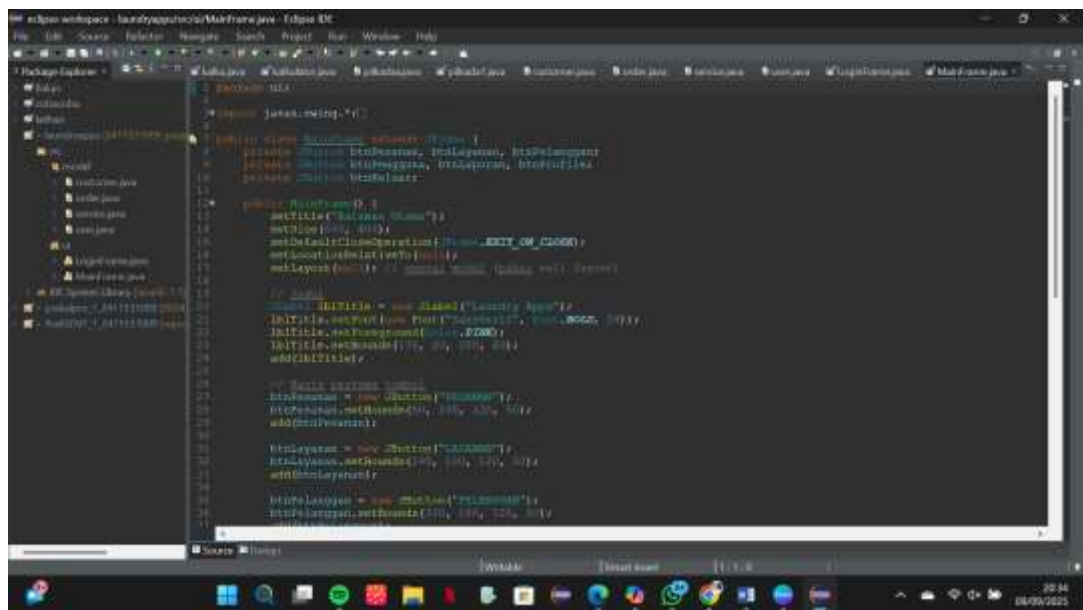




Program di atas adalah implementasi tampilan form login untuk aplikasi LaundryApps menggunakan Java Swing. Class LoginFrame diturunkan dari JFrame sehingga berfungsi sebagai sebuah window aplikasi. Di dalamnya terdapat beberapa komponen: label judul aplikasi “Laundry Apps”, tagline, input teks untuk username (txtUsername), input teks untuk password (txtPassword), serta tombol Login (btnLogin). Komponen-komponen tersebut diatur secara manual menggunakan setBounds() sehingga setiap posisi dan ukurannya ditentukan secara eksplisit.

Ketika tombol login ditekan, event handler ActionListener akan dijalankan. Program mengambil nilai dari txtUsername dan txtPassword, lalu memanggil method login() dari class user untuk memverifikasi kecocokan data. Jika login berhasil, ditampilkan pesan “Login berhasil” menggunakan JOptionPane, kemudian window utama (MainFrame) ditampilkan dan window login ditutup dengan dispose(). Jika username atau password salah, maka program akan menampilkan pesan peringatan “Username / Password salah”. Dengan cara ini, program sudah mampu menghubungkan antarmuka login sederhana dengan logika autentikasi user yang sebelumnya didefinisikan di class user.

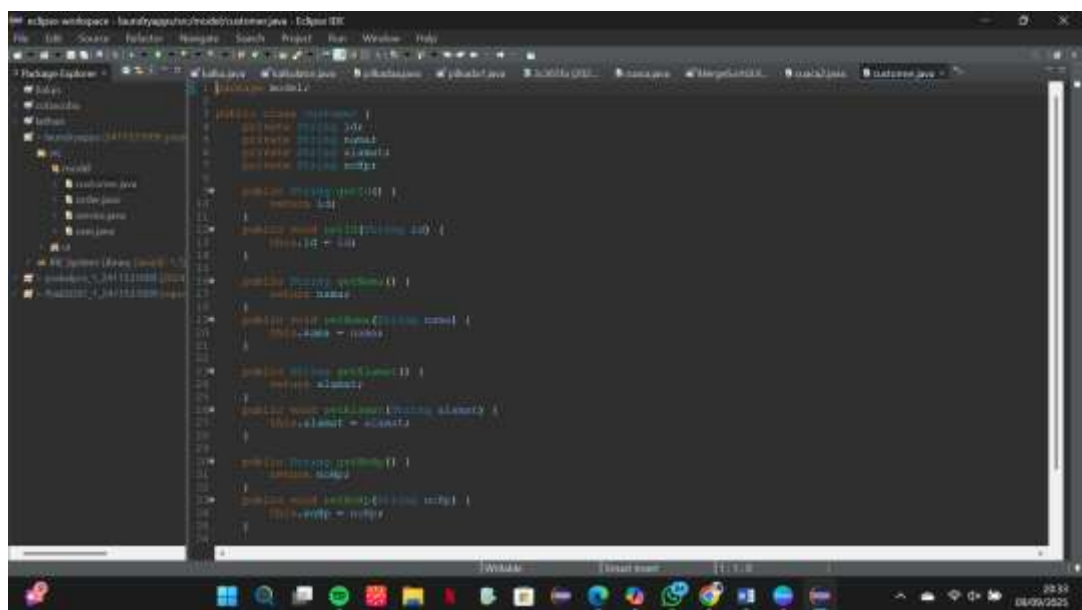
##### 5. Membuat MainFrame sebagai halaman utama.



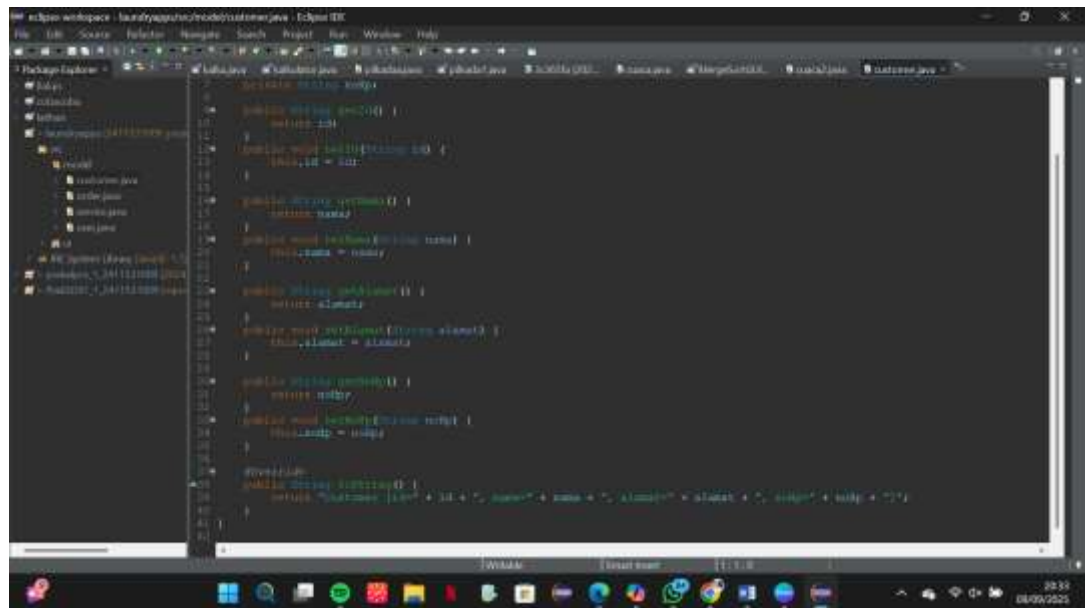
Program di atas adalah class LoginFrame yang berfungsi sebagai tampilan form login pada aplikasi LaundryApps menggunakan Java Swing. Class ini diturunkan dari JFrame, sehingga dapat ditampilkan sebagai sebuah jendela aplikasi. Di dalam konstruktor, jendela diatur dengan ukuran 350x250 piksel, judul “Login - Laundry Apps”, serta posisi yang ditengah layar. Program kemudian menambahkan beberapa komponen grafis, seperti label judul “Laundry Apps” dengan warna pink dan tagline “Bersih, Cepat, dan Terpercaya”, label untuk username dan password, dua JTextField sebagai input username dan password, serta sebuah tombol Login. Semua komponen ditempatkan menggunakan setBounds() dengan tata letak null layout.

Fungsi utama form ini terletak pada event handler tombol login. Ketika tombol ditekan, program mengambil teks dari input username dan password, lalu memanggil method login() pada class user untuk memverifikasi data. Jika login berhasil, program menampilkan pesan “Login berhasil”, membuka tampilan utama (MainFrame), dan menutup jendela login dengan dispose(). Jika data tidak cocok, akan muncul pesan kesalahan “Username / Password salah”. Dengan demikian, program ini menghubungkan antarmuka login dengan proses autentikasi user, sekaligus menjadi gerbang menuju halaman utama aplikasi Laundry.

## 6. Membuat class tambahan (Costumer)





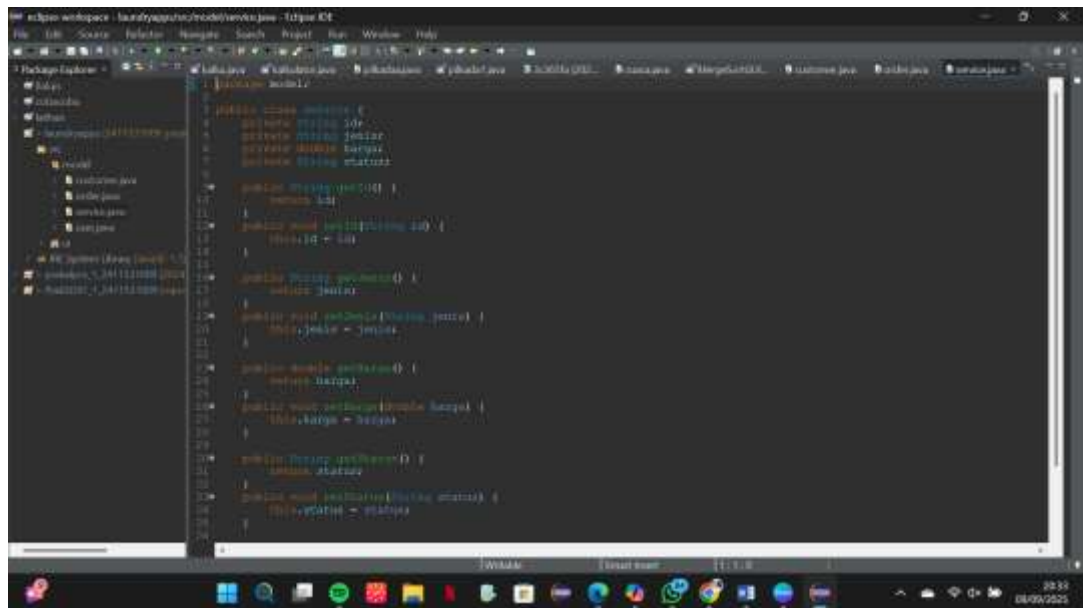


Program di atas adalah class customer yang merepresentasikan data pelanggan dalam aplikasi LaundryApps. Class ini memiliki empat atribut bersifat private, yaitu id, nama, alamat, dan noHp, yang masing-masing digunakan untuk menyimpan identitas pelanggan, nama pelanggan, alamat rumah, serta nomor telepon. Untuk mengakses dan mengubah nilai atribut tersebut, disediakan method getter dan setter sesuai prinsip encapsulation di Java.

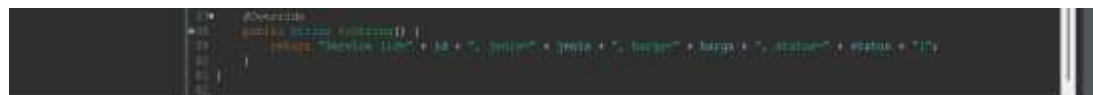
Dengan cara ini, data pelanggan tetap terlindungi dan hanya bisa dimodifikasi melalui method yang sudah ditentukan. Selain itu, class ini juga mengoverride method toString(), yang mengembalikan representasi teks dari objek customer dalam format "Customer [id=..., nama=..., alamat=..., noHp=...]". Method ini sangat berguna ketika objek ingin ditampilkan dalam bentuk string, misalnya saat debugging atau menampilkan daftar pelanggan di aplikasi.

## 7. Membuat class tambahan (Service)





```
1 package model;
2
3 public class service {
4     private int id;
5     private String jenis;
6     private double harga;
7     private String status;
8
9     public service(int id) {
10         this.id = id;
11     }
12     public service(String jenis) {
13         this.jenis = jenis;
14     }
15     public service(double harga) {
16         this.harga = harga;
17     }
18     public service(String status) {
19         this.status = status;
20     }
21
22     public int getId() {
23         return id;
24     }
25     public String getJenis() {
26         return jenis;
27     }
28     public double getHarga() {
29         return harga;
30     }
31     public String getStatus() {
32         return status;
33     }
34 }
```

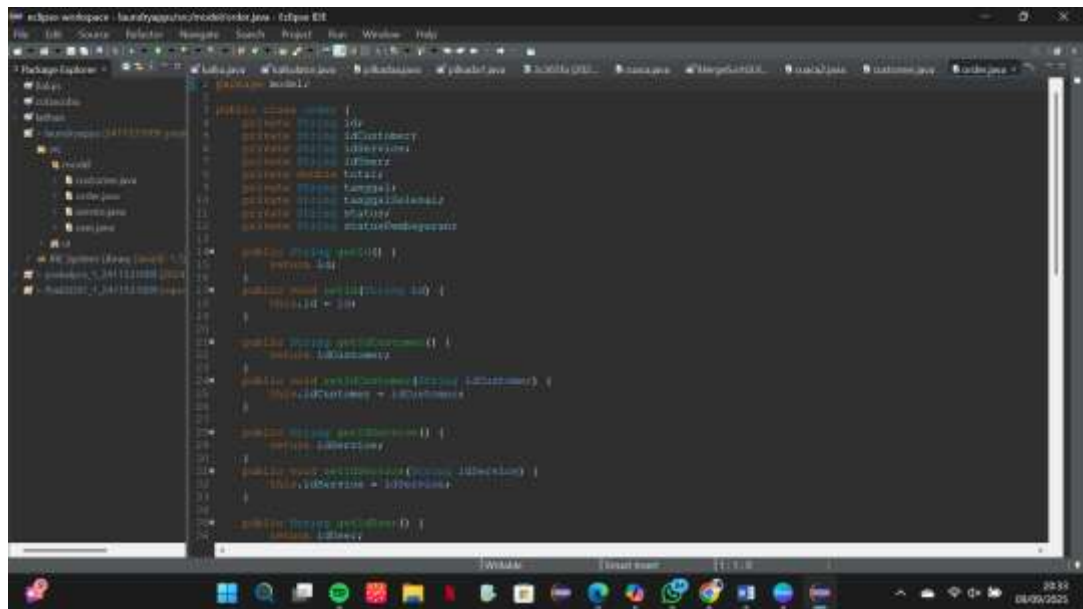


```
1 public String toString() {
2     return "Service [id=" + id + ", jenis=" + jenis + ", harga=" + harga + ", status=" + status + "]";
3 }
```

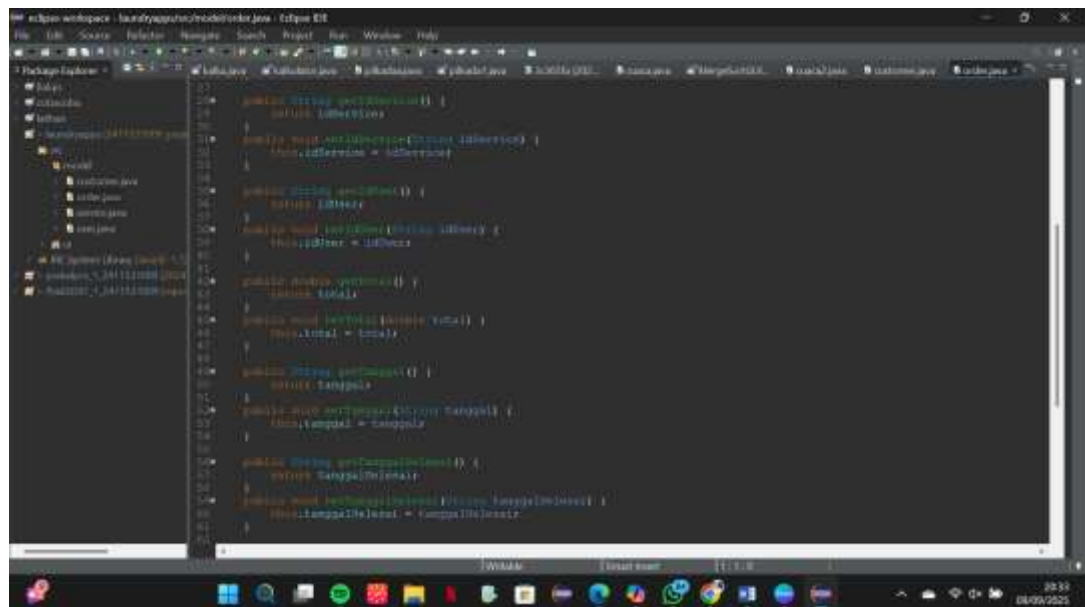
Program di atas adalah class service yang berfungsi untuk merepresentasikan data layanan laundry dalam aplikasi. Class ini memiliki empat atribut utama, yaitu id sebagai identitas layanan, jenis untuk menyimpan jenis layanan (misalnya cuci kering, setrika, dll.), harga untuk biaya layanan, serta status yang menunjukkan ketersediaan layanan. Seluruh atribut dideklarasikan sebagai private, sehingga tidak bisa diakses langsung dari luar class.

Untuk mengatur dan mengambil nilai dari setiap atribut, disediakan method getter dan setter sesuai prinsip encapsulation. Sebagai tambahan, class ini juga mengoverride method toString(), yang menghasilkan representasi teks dari objek service dalam format "Service [id=..., jenis=..., harga=..., status=...]". Dengan cara ini, data layanan dapat dikelola dengan rapi sekaligus mudah ditampilkan ketika dibutuhkan, misalnya dalam daftar layanan laundry di aplikasi.

#### 8. Membuat class tambahan (Order)



```
1 public class Order {
2     private String id;
3     private String idCustomer;
4     private String idService;
5     private String idUser;
6     private String tanggal;
7     private String tanggalSelesai;
8     private String status;
9     private String statusPembayaran;
10
11     public String getId() {
12         return id;
13     }
14
15     public void setId(String id) {
16         this.id = id;
17     }
18
19     public String getIdCustomer() {
20         return idCustomer;
21     }
22
23     public void setIdCustomer(String idCustomer) {
24         this.idCustomer = idCustomer;
25     }
26
27     public String getIdService() {
28         return idService;
29     }
30
31     public void setIdService(String idService) {
32         this.idService = idService;
33     }
34
35     public String getIdUser() {
36         return idUser;
37     }
38 }
```



```
39     public String getIdUser() {
40         return idUser;
41     }
42
43     public void setIdUser(String idUser) {
44         this.idUser = idUser;
45     }
46
47     public String getTanggal() {
48         return tanggal;
49     }
50
51     public void setTanggal(String tanggal) {
52         this.tanggal = tanggal;
53     }
54
55     public String getTanggalSelesai() {
56         return tanggalSelesai;
57     }
58
59     public void setTanggalSelesai(String tanggalSelesai) {
60         this.tanggalSelesai = tanggalSelesai;
61     }
62 }
```



```
63
64
65 @Override
66 public String toString() {
67     return "Order [id=" + id + ", idCustomer=" + idCustomer +
68         ", idService=" + idService + ", idUser=" + idUser +
69         ", total=" + total + ", tanggal=" + tanggal +
70         ", tanggalSelesai=" + tanggalSelesai +
71         ", status=" + status + ", statusPembayaran=" + statusPembayaran + "]";
72 }
```

Program di atas merupakan class order yang digunakan untuk merepresentasikan data transaksi pemesanan laundry dalam aplikasi. Class ini memiliki beberapa atribut utama, yaitu id, idCustomer, idService, idUser, total, tanggal, tanggalSelesai, status, dan statusPembayaran. Semua atribut dibuat private agar

tidak bisa diakses langsung dari luar class, sesuai prinsip encapsulation. Untuk setiap atribut disediakan method getter dan setter, sehingga data dapat diambil dan diubah dengan aman. Misalnya, getTotal() digunakan untuk membaca nilai total biaya, sedangkan setTotal(double total) untuk menentukannya. Selain itu, class ini juga mengoverride method toString() yang berfungsi mengembalikan representasi objek order dalam bentuk string dengan menampilkan seluruh informasi pesanan, mulai dari identitas order, customer, service, user, hingga detail pembayaran. Dengan struktur ini, class order dapat digunakan untuk menyimpan, mengolah, dan menampilkan informasi transaksi laundry secara lengkap dan terorganisir.

### **iii. KESIMPULAN**

Dari praktikum ini dapat disimpulkan bahwa:

- Class berfungsi sebagai blueprint untuk membuat objek.
- Object adalah instance dari class yang memiliki atribut dan method.
- Constructor digunakan untuk menginisialisasi objek.
- Method berperan dalam mengatur perilaku class.
- Dengan menggunakan konsep OOP (PBO), pembuatan aplikasi Laundry menjadi lebih terstruktur, modular, dan mudah dikembangkan.

