Kelompok : LAH
1. 13523023 / Muhammad Aufa Farabi
2. 13523025 / Joel Hotlan Haris Siahaan
3. 13523030 / Julius Arthur
4. 13523051 / Ferdinand Gabe Tua Sinaga
5. 13523111 / Muhammad Iqbal Haidar

Asisten Pembimbing : Jeffrey Chow

1. Deskripsi Umum Aplikasi

Merupakan aplikasi desktop berbasis JavaFX yang dirancang untuk membantu mengelola logistik pengiriman barang secara efisien dan modern. Aplikasi ini menyediakan fitur lengkap seperti manajemen pengiriman (domestik dan internasional), manajemen kurir, serta pengelolaan parsel. Setiap pengiriman dapat dilacak status dan riwayat perubahannya secara real-time, dengan dukungan sistem pencarian dan filter yang fleksibel. Data disimpan secara persisten dalam format XML atau JSON, sesuai pilihan pengguna.

Selain itu, aplikasi ini mendukung fitur plugin yang dapat menambahkan modul tambahan seperti analitik visualisasi data yang telah dikumpulkan selama berjalannya aplikasi. Pengguna dapat berganti akun antara Admin dan Kurir, serta memanfaatkan fitur notifikasi untuk parsel yang baru di-assign. Akun admin dan kurir memiliki tampilan halaman yang disesuaikan dengan kebutuhan dan kewenangannya. Aplikasi juga dilengkapi dengan tampilan GUI dan threading untuk proses latar belakang.

2. Kakas GUI: JavaFX Description

1. Tuliskan deskripsi tentang kakas gui yang kalian pakai.

   JavaFX adalah kakas GUI berbasis Java yang digunakan untuk membangun aplikasi desktop dengan antarmuka interaktif dan modern. Dalam proyek ini, JavaFX digunakan untuk menampilkan halaman-halaman seperti manajemen pengiriman, parsel, kurir, dan plugin, menggunakan komponen-komponen bawaan yang sudah disediakan. JavaFX mendukung styling berbasis CSS dan struktur scene graph, serta memudahkan pengelolaan event dan layout.

2. Sebutkan juga lifecycle/cara menggunakan kakas gui tersebut

   Pada umumnya lifecycle standar yang biasa digunakan adalah sebagai berikut

   1. Inisialisasi: digunakan untuk menginisialisasi data yang akan diproses
   2. Start: Merupakan titik masuk utama GUI. Di sinilah elemen-elemen visual disusun dan ditampilkan menggunakan Scene dan Stage
   3. Dipanggil saat aplikasi akan ditutup. Cocok digunakan untuk menutup koneksi database atau thread.

3. apa saja komponen utamanya agar gui bisa muncul.

   Komponen utama yang diperlukan agar GUI bisa muncul antara lain adalah

   1. Application: Kelas utama yang meng-extend javafx.application.Application. Di sinilah start() didefinisikan.
   2. Stage: Merupakan jendela utama aplikasi (seperti JFrame pada Swing).
   3. Scene: Menyimpan konten yang ditampilkan pada stage. Scene berisi hierarki node melalui Scene Graph.
   4. FXML: File XML yang mendefinisikan struktur UI, terpisah dari kode Java.
   5. Controller: Kelas Java yang menangani logika dan event dari komponen dalam file FXML.

4. Apa saja komponen yang disediakan kakas gui tersebut.

   Beberapa komponen yang disediakan pada oleh JavaFX antara lain adalah:

   - UI Controls: Button, Label, TextField, PasswordField, TableView, ListView, ComboBox, CheckBox, RadioButton, DatePicker, Slider, ProgressBar, Alert, Dialog
   - Layout Containers: HBox, VBox, BorderPane, GridPane, StackPane, AnchorPane, FlowPane, TilePane, Pane
   - Komponen tambahan: ImageView, Canvas, WebView, MediaPlayer

3. Plugin & Class Loader

Class loader di Java adalah bagian dari Java Virtual Machine (JVM) yang bertugas memuat kelas ke dalam memori saat program dijalankan. Proses ini dilakukan secara dinamis, yaitu kelas baru dapat dimuat saat runtime berdasarkan kebutuhan, bukan saat kompilasi. Untuk

mendukung plugin, kita bisa membuat custom class loader yang membaca file .jar eksternal dan memuat kelas-kelas yang ada di dalamnya secara dinamis. Ini memungkinkan aplikasi menjalankan fitur baru tanpa harus di-rebuild atau restart.

4. Class Diagram

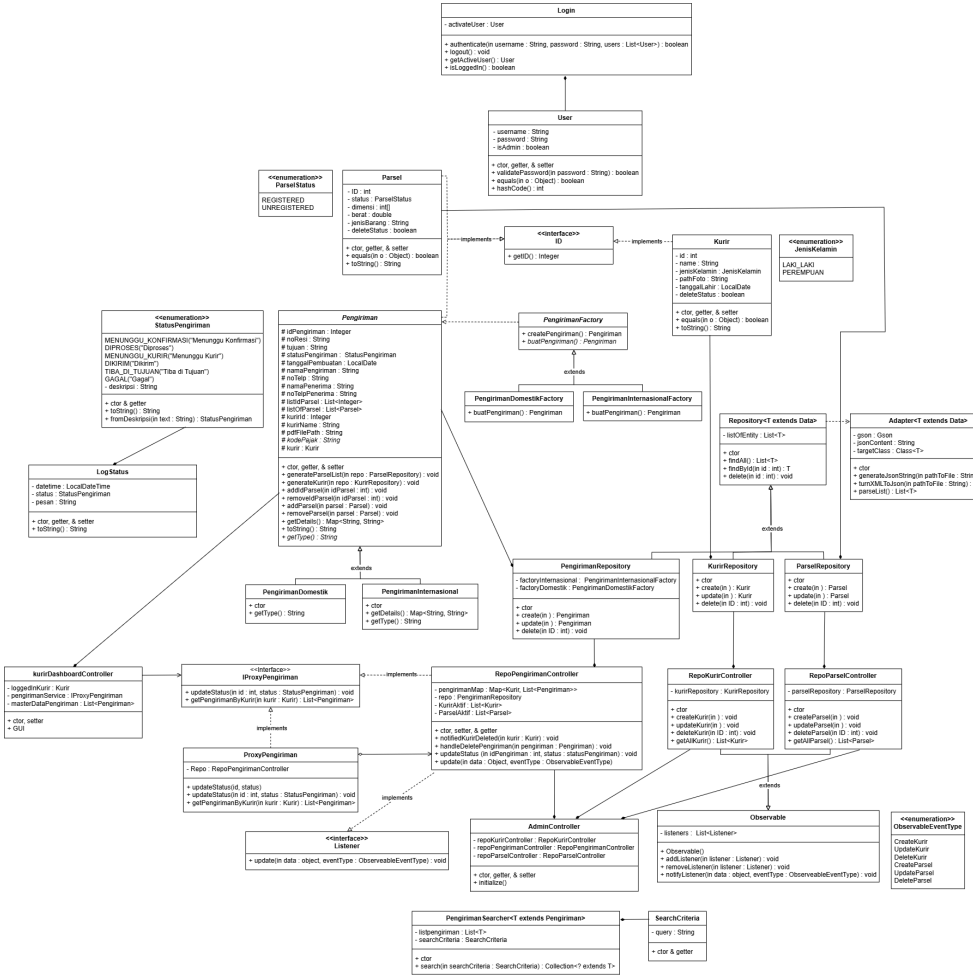


Diagram kelas yang ditampilkan menunjukkan arsitektur sistem manajemen pengiriman berbasis objek, dengan pemisahan yang jelas antara entitas data (seperti User, Parsel, Kurir, dan Pengiriman) dan komponen pengelola seperti repository dan controller. Kelas-kelas entitas seperti Kurir dan Parsel menyimpan informasi data inti, sedangkan Pengiriman memiliki subclass PengirimanDomestik dan PengirimanInternasional untuk menangani logika spesifik tiap jenis pengiriman.

Komunikasi antar objek difasilitasi melalui berbagai design pattern. Misalnya, PengirimanFactory menggunakan pola Factory Method untuk membuat objek pengiriman tanpa perlu mengetahui kelas konkret yang digunakan. Repository seperti KurirRepository dan ParselRepository mengikuti pola Repository untuk menyimpan dan mengambil data, dan digunakan oleh controller seperti RepoKurirController agar logika aplikasi tidak terikat langsung pada data storage.

Untuk interaksi dinamis, pola Observer digunakan melalui kelas Observable dan interface Listener agar sistem dapat merespons event secara reaktif. Selain itu, terdapat adapter dan proxy (ProxyPengiriman) yang menjembatani antarmuka dan menyederhanakan interaksi antar modul. Dengan struktur ini, komunikasi antar objek menjadi modular dan mudah dikembangkan, karena antar komponen hanya bergantung pada abstraksi/interface, bukan implementasi langsung.

5. Konsep OOP

5.1. Inheritance

- Kelas PengirimanDomestik (src/main/java/com/labpro/PengirimanDomestik.java) & PengirimanInternasional (src/main/java/com/labpro/PengirimanInternasional.java) merupakan anak dari kelas Pengiriman (src/main/java/com/labpro/Pengiriman.java)
- Kelas PengirimanDomestikFactory (src/main/java/com/labpro/PengirimanDomestikFactory.java) & PengirimanInternasionalFactory (src/main/java/com/labpro/PengirimanInternasionalFactory.java) merupakan anak dari kelas PengirimanFactory (src/main/java/com/labpro/PengirimanFactory.java)
- Kelas PengirimanRepository (src/main/java/com/labpro/PengirimanRepository.java), KurirRepository (src/main/java/com/labpro/KurirRepository.java), & ParselRepository (src/main/java/com/labpro/ParselRepository.java) merupakan anak dari kelas Repository (src/main/java/com/labpro/Repository.java)

5.2. Composition

- Kelas RepoPengirimanController (src/main/java/com/labpro/RepoPengirimanController.java) memiliki tepat satu kelas PengirimanRepository (src/main/java/com/labpro/PengirimanRepository.java)
- Kelas ParselRepository (src/main/java/com/labpro/KurirRepository.java) memiliki banyak kelas Kurir (src/main/java/com/labpro/Parsel.java)
- Kelas kurirDashboardController (src/main/java/com/labpro/KurirDashboardController.java) memiliki banyak kelas Kurir (src/main/java/com/labpro/Kurir.java)

5.3. Interface

- Interface ID (src/main/java/com/labpro/ID.java)
- Interface Listener (src/main/java/com/labpro/Listener.java)
- Interface IProxyPengiriman (src/main/java/com/labpro/IProxyPengiriman.java)

5.4. Method Overriding dan Method Overloading

- Method create pada kelas PengirimanRepository (src/main/java/com/labpro/PengirimanRepository.java)
- Method update pada kelas PengirimanRepository (src/main/java/com/labpro/PengirimanRepository.java)
- Constructor pada kelas Adapter (src/main/java/com/labpro/Adapter.java)
- Method getType() pada kelas Pengiriman (src/main/java/com/labpro/Pengiriman.java)

5.5. Polymorphism

- Kelas pengiriman memiliki List of Pengiriman yang dimana isinya dapat berupa kelas PengirimanDomestik dan kelas PengirimanInternasional
- Method update pada kelas RepoPengirimanController menerima kelas Object dimana kelas tersebut bisa berupa kelas Kurir ataupun parsel

5.6. Java API Collection

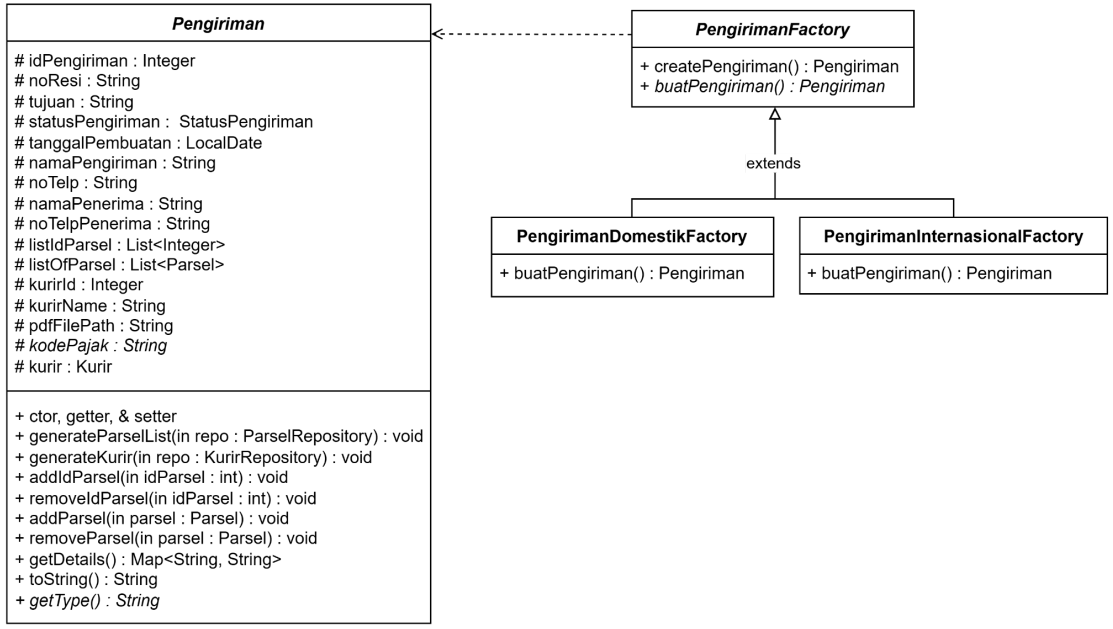
- Kelas Pengiriman (src/main/java/com/labpro/Pengiriman.java) memiliki List of Parsel
- Kelas Repository (src/main/java/com/labpro/Repository.java) memiliki List of T dimana T bertipe Parsel/Kurir/Pengiriman
- Kelas RepoPengirimanController (src/main/java/com/labpro/RepoPengirimanController.java) memiliki Map<Kurir, List of Pengiriman>

5.7. SOLID

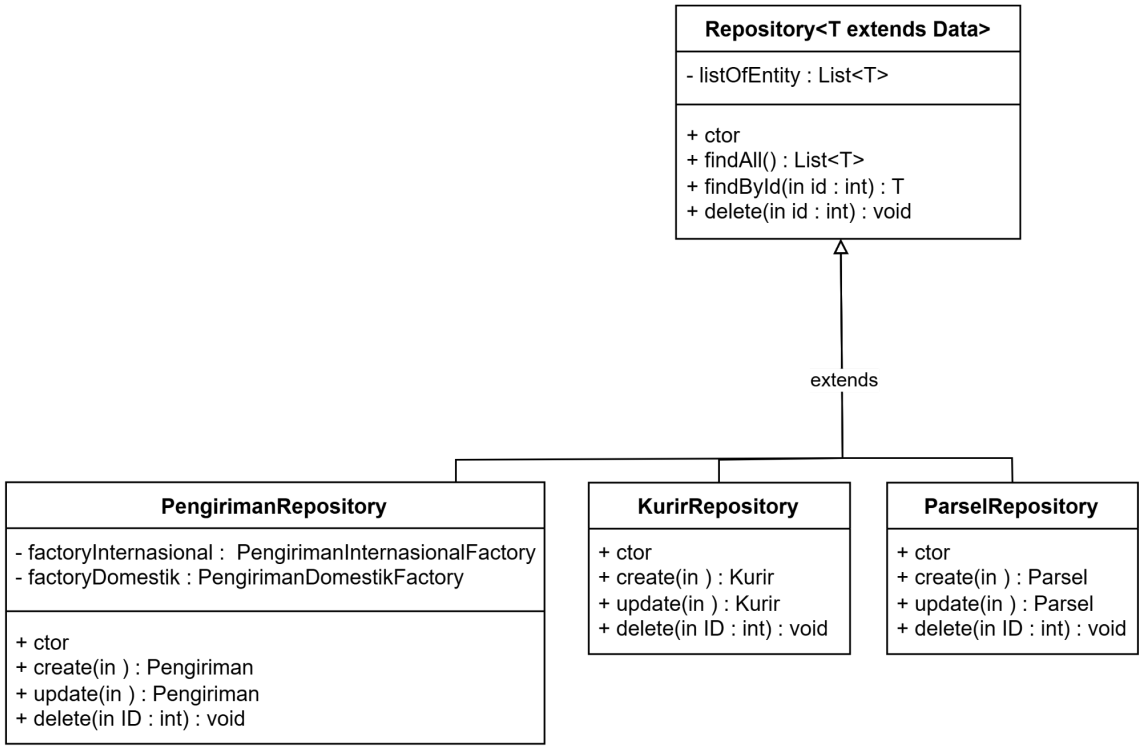
- S: Kelas Repository (src/main/java/com/labpro/Repository.java) memiliki turunan kelas yang khusus mengatur repository khusus berisi sebuah kelas seperti PengirimanRepository hanya bertugas untuk mengatur Pengiriman, begitu juga untuk KurirRepository dan ParselRepository.
- O: Kelas Repository (src/main/java/com/labpro/Repository.java) memanfaatkan konsep generik sehingga membuat kelas ini bisa diperluas untuk menyimpan berbagai jenis data tanpa mengubah kode dasarnya
- L: Kelas turunan Pengiriman (src/main/java/com/labpro/Pengiriman.java) yakni PengirimanDomestik dan PengirimanInternasional dapat menggantikan kelas orangtuanya tanpa mengurangi fungsionalitasnya
- I: Interface ID (src/main/java/com/labpro/ID.java) sebagai interface yang menyediakan method getID() yang hanya perlu diimplementasikan untuk kelas yang membutuhkan saja seperti Kurir dan Parsel tetapi tidak untuk kelas User
- D: Kelas KurirDashboardController (src/main/java/com/labpro/KurirDashboardController.java) memiliki atribut IProxyPengiriman (src/main/java/com/labpro/IProxyPengiriman.java) sehingga tidak bergantung pada implementasi konkret ProxyPengiriman tetapi hanya interfacenya saja

5.8. Design Pattern

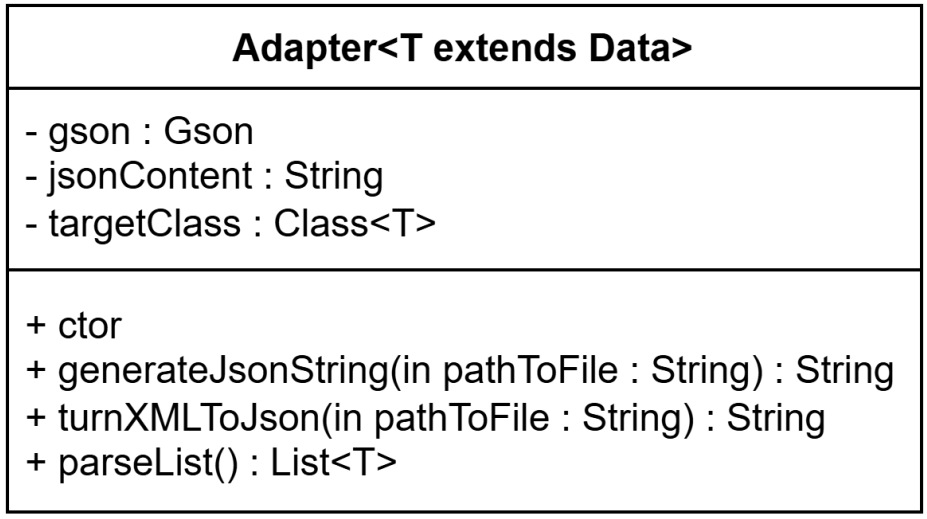
- Factory Method: PengirimanFactory (src/main/java/com/labpro/.java) sebagai kelas factory dan menghasilkan kelas PengirimanDomestikFactory (src/main/java/com/labpro/PengirimanDomestikFactory.java) dan PengirimanInternasionalFactory (src/main/java/com/labpro/PengirimanInternasionalFactory.java) tujuannya agar instansiasi kelas Pengiriman dan turunannya bisa hanya menggunakan satu method saja. Kelebihannya, ketika pengguna kelas ingin menginstansiasi kelas Pengiriman dan turunannya cukup memanggil method createPengiriman(). Apabila tidak diterapkan maka pengguna kelas dapat kesulitan karena harus mengatur supaya constructor yang digunakan harus sesuai dengan kelas yang ingin dinstansiasikan.



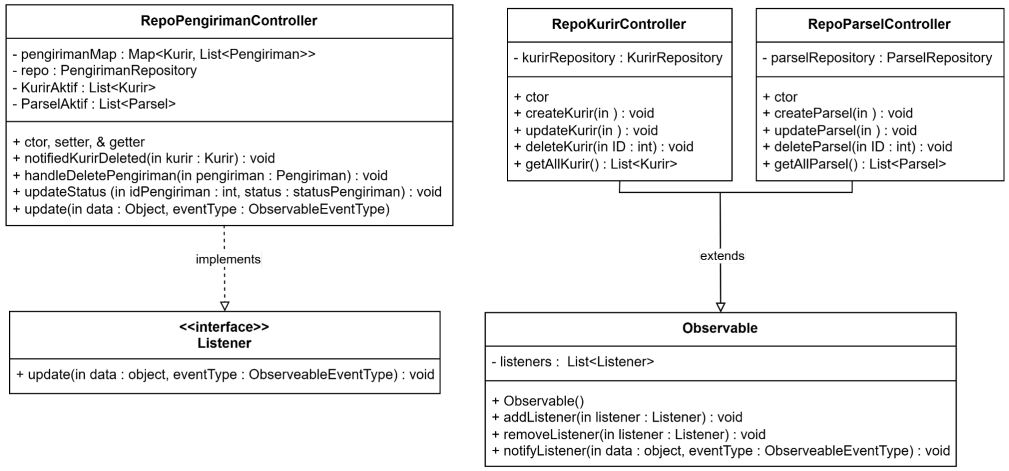
- Repository: Repository (src/main/java/com/labpro/Repository.java) sebagai kelas repository dan menghasilkan kelas PengirimanRepository (src/main/java/com/labpro/PengirimanRepository.java), KurirRepository (src/main/java/com/labpro/KurirRepository.java), dan ParselRepository (src/main/java/com/labpro/ParselRepository.java) tujuannya agar menyederhanakan akses data dengan format/struktur yang berbeda-beda. Kelebihannya, pengguna kelas tidak perlu pusing menyesuaikan kode dengan format/struktur penyimpanan data sebab metode pengaksesan sudah terstandarisasi. Apabila tidak diterapkan maka pengguna kelas dapat kesulitan karena harus memberi perlakuan khusus untuk setiap struktur penyimpanan data.



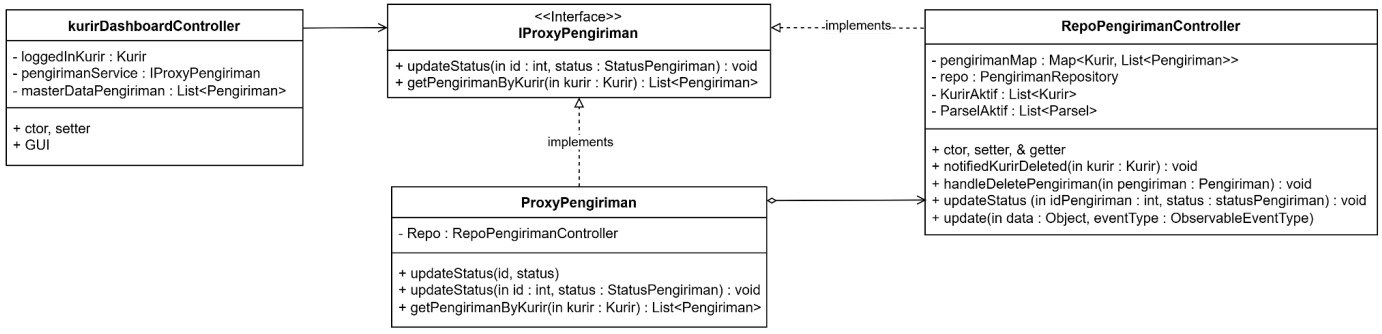
- Adapter: Adapter (src/main/java/com/labpro/Adapter.java) sebagai kelas adapter tujuannya agar program dapat menerima dan membaca file data dengan ekstensi beragam seperti .json dan .xml. Kelebihannya, dengan menggunakan design pattern ini maka cukup membuat satu parser khusus untuk file .json lalu apabila dikemudian hari memerlukan dukungan tambahan untuk ekstensi file lain maka bisa tetap menggunakan parser tersebut. Apabila tidak menerapkan hal ini, maka pembuat kelas harus membuat parser berbeda untuk setiap ekstensi file.



- Observer: Listener (src/main/java/com/labpro/Listener.java) sebagai interface subscriber dan Observable (src/main/java/com/labpro/Observable.java) sebagai kelas publisher pada design pattern ini tujuannya adalah supaya setiap kali Repositori Kurir ataupun Repositori Parsel mengalami perubahan maka Repository Pengiriman akan diberitahu sehingga data dalam Repository Pengiriman selalu tersinkron dengan repositori lainnya. Kelebihan dari design pattern ini adalah dapat membantu menyelaraskan berbagai data objek dengan objek-objek lainnya yang saling berhubungan. Selain itu apabila diperlukan penambahan repositori lain yang juga berhubungan maka cukup implementasikan interface listener pada kelas repositori tersebut dan masukkan kedalam listeners milik publisher. Apabila tidak mengimplementasikan design pattern pattern ini maka pemrogram akan pusing untuk mengatur notifikasi/pemberitahuan setiap kali ada pertambahan objek baru.



- Proxy: Design pattern ini menghasilkan sebuah interface yakni IProxyPengiriman (src/main/java/com/labpro/IProxyPengiriman.java) dan sebuah kelas ProxyPengiriman (src/main/java/com/labpro/ProxyPengiriman.java) tujuannya adalah untuk memastikan bahwa Kurir tidak bisa mengakses langsung RepoPengirimanController agar menjamin keamanan data Repository Pengiriman. Kelebihan dari design pattern ini adalah setiap kelas dapat diberikan batasan apa saja yang bisa dilakukan sehingga tidak semua method dari kelas lain dapat diakses oleh kelas tersebut. Apabila tidak mengimplementasikan design pattern ini maka pembuat kelas perlu untuk membuatkan mekanisme verifikasi terhadap setiap objek yang memanggil method dari kelas tersebut tentu ini akan menambah kompleksitas.



5.9. Reflection

Tuliskan daftar penggunaan konsep ini di aplikasi, misalnya:

- Method x pada kelas A (src/main/A.java) menggunakan reflection … untuk …
- Method y pada kelas B (src/main/B.java) menggunakan reflection … untuk …

5.10. Threading

- Threading pada kelas TimeThread (src/main/java/com/labpro/TimeThread.java) mengextends kelas Thread dan meng-override method run(). Hal ini supaya objek TimeThread bisa berjalan multithreading sehingga jam bisa terus berjalan tanpa membuat “hang” program utama.

6. Pembagian Tugas

- 13523023 / Muhammad Aufa Farabi
    - Admin UI, Kurir UI, Parsel Repository, Kurir Repository, Kurir Controller, Admin Controller
- 13523025 / Joel Hotlan Haris Siahaan
    - Parsel UI, Pengiriman Repository, Login UI, Parsel, User, Login
- 13523030 / Julius Arthur
    - Pengiriman controller, manajemen pengiriman, time thread, jacoco, observer.
- 13523051 / Ferdinand Gabe Tua Sinaga
    - Kurir UI, Kurir Controller, Factory Method Pengiriman, Pengiriman, Adapter
- 13523111 / Muhammad Iqbal Haidar
    - Laporan, Proxy

7. Foto Kelompok