DePaul University

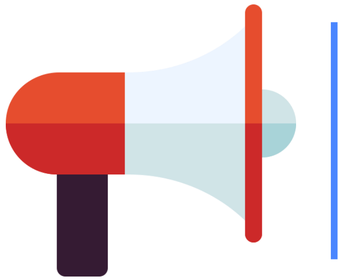
Design Patterns:

Chain of Responsibility, Decorator

Object-oriented Software Development
SE 350– Spring 2021

Vahid Alizadeh

Week 8.2
May 20, 2021



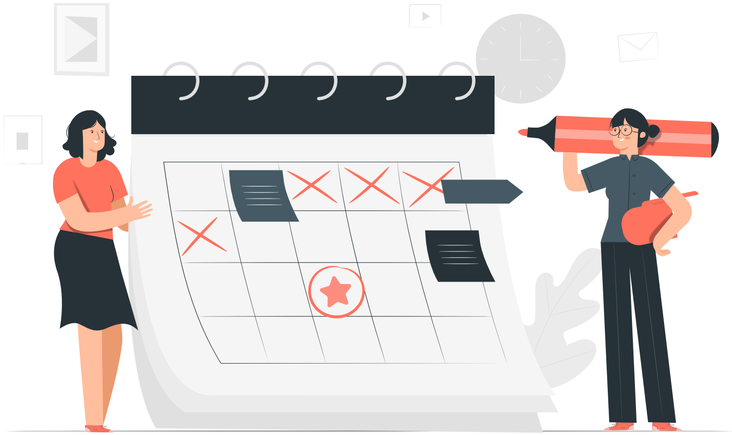
Announcements

Future Schedule

Assignment 4 is released

Due date: May 28, Friday, 11:59PM

- ~~Assignment 1~~
- ~~Assignment 2~~
- ~~Mid Term Exam~~
- **Assignment 3:**
  - Release: Week 7
  - Due: Week 8
- **Assignment 4:**
  - Release: Week 8 (TODAY)
  - Due: Week 9
- **Bonus Research Project:**
  - Presentation Due: Week 10
  - Report Due: Week 11
- **Final Exam:**
  - Week 11



SE 350: OO Software Development

Assignment 4: Design Patterns (2)

Instructor: Vahid Alizadeh
Email: v.alizadeh@depaul.edu
Quarter: Spring 2021

DEPAUL UNIVERSITY
COLLEGE OF COMPUTING AND DIGITAL MEDIA

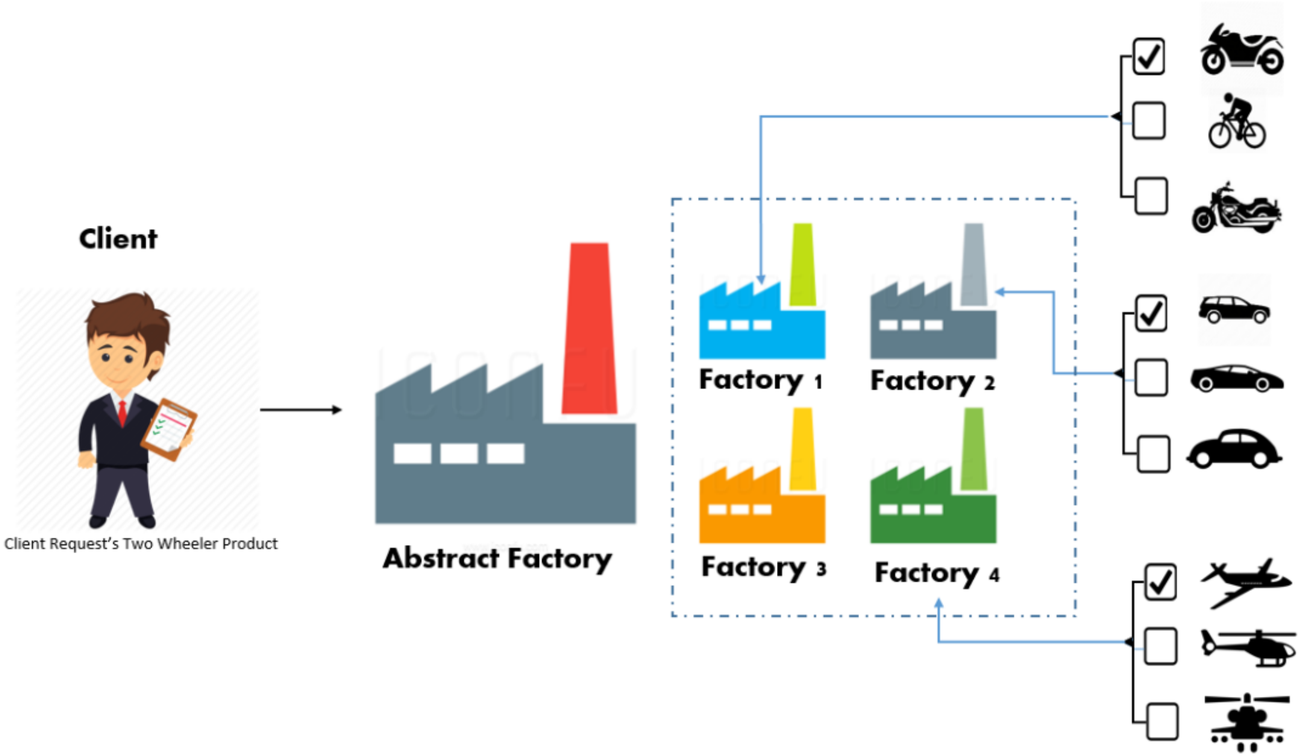
Last update: May 19, 2021

Abstract Factory Pattern

CREATIONAL

Abstract Factory Pattern Introduction

Abstract Factory is a creational design pattern that lets you produce families of related objects.



Abstract Factory Pattern Pros & Cons

Pros

- Making sure to have compatible products.
- Avoiding tight coupling.
- Single Responsibility Principle.
- Open/Closed Principle.

Cons

- The code may become more complicated because of introducing many interfaces, classes, and subclasses.

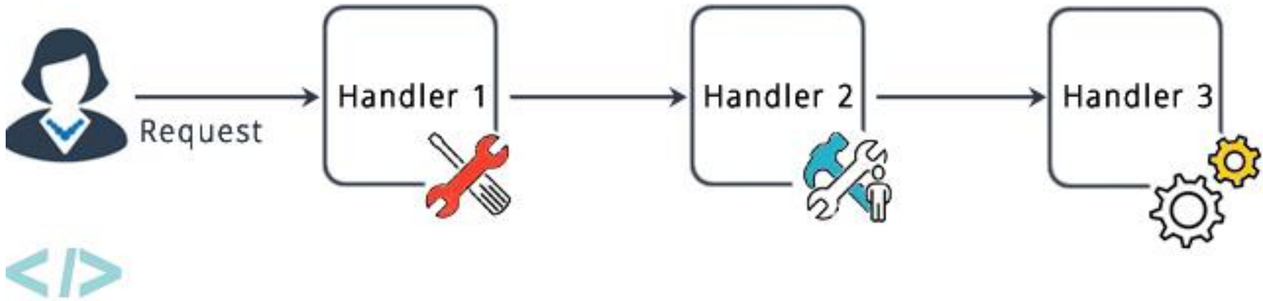
Chain of Responsibility Pattern

CoR / Chain of Commands

BEHAVIORAL

CoR Pattern Introduction

Chain of Responsibility is a behavioral design pattern that can be used when we want to give more than one object a chance to handle a request.



CoR Design Pattern

INTENT

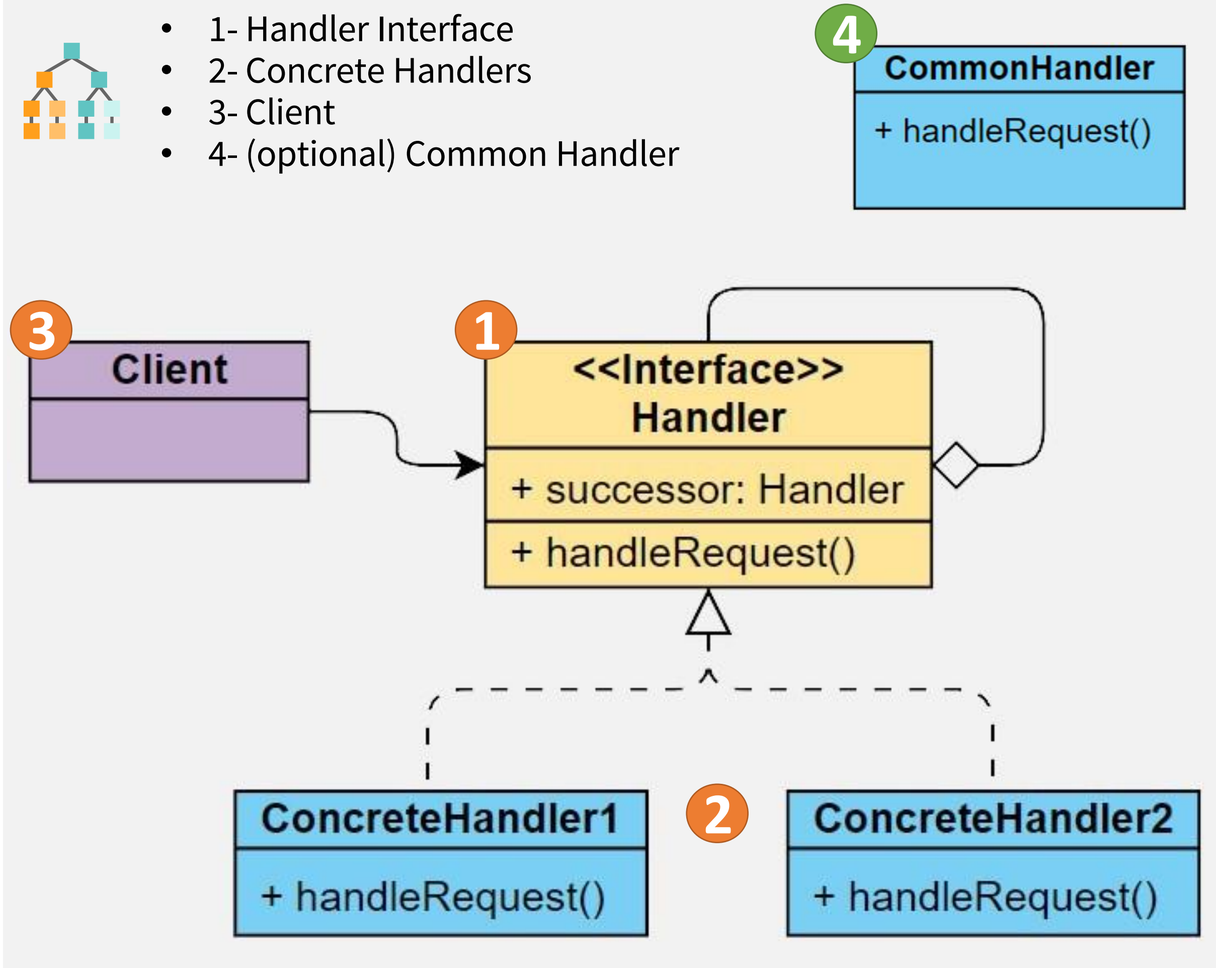
- The chain-of-responsibility pattern chains the handlers in such a way that they will be able to process the request or pass it on if they are not able to do it.

PROBLEM

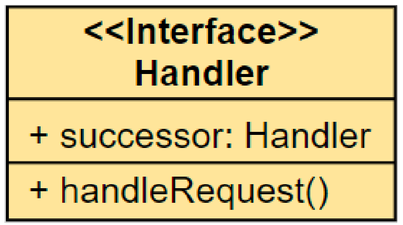
- There is a potentially variable number of "handler" objects, and a stream of requests that must be handled.

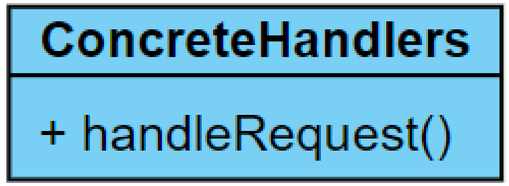
STRUCTURE

- 1- Handler Interface
- 2- Concrete Handlers
- 3- Client
- 4- (optional) Common Handler



Implementation Guide

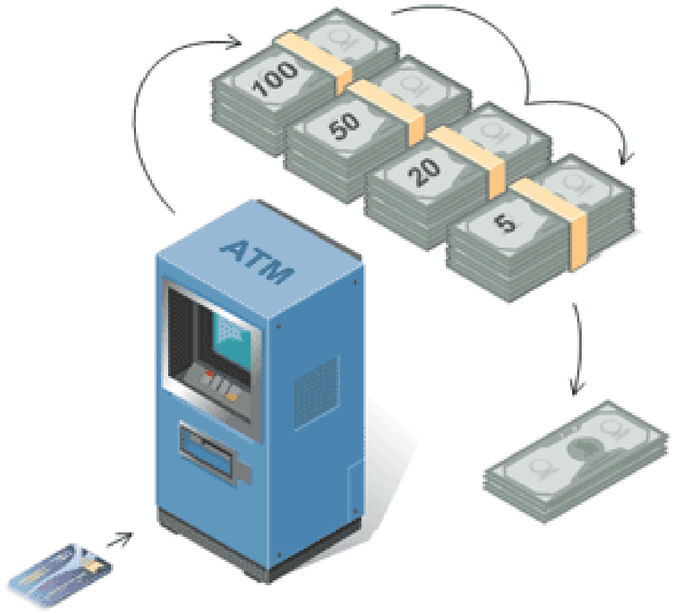





```protected Handler successor;
public void setSuccessor(Handler successor)
{
    this.successor = successor;
}
```

```public void handleRequest(Request request)
{
  if (canHandle(request)) {
    //code to handle the request
  }
  else {
    successor.handleRequest();
  }
}
```

CoR Pattern: Real-world Example

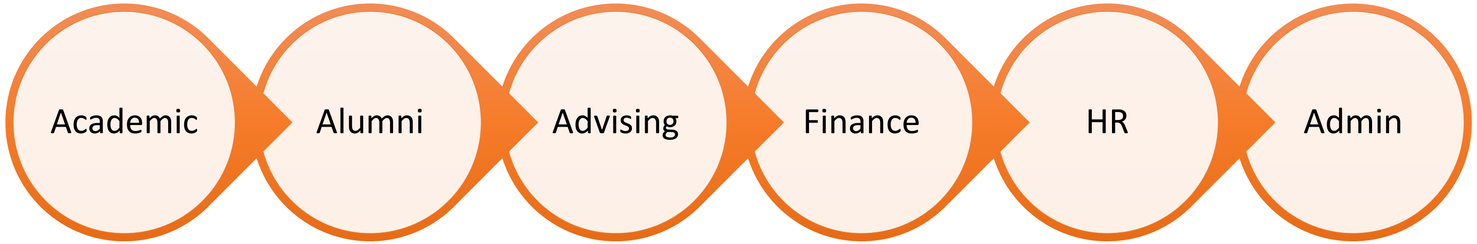


```
try{
      // code that may throw exception
}
catch(e: ExceptionType){
      // code to execute  if exception occurs
}
```

CoR Use Case Example: University Email

Keywords	University Team	Java Handler
academic	Academic Team	`AcademicEmailHandler()`
alumni, transcript	Alumni Affairs	`AlumniEmailHandler()`
advising, schedule, course	Advising Staff	`AdvisingEmailHandler()`
financial, student aid, tuition	Finance Staff	`FinanceEmailHandler()`
career, job, faculty	Human Resources	`HREmailHandler()`
other	Admin Team	`AdminEmailHandler()`

Keyword-Handler

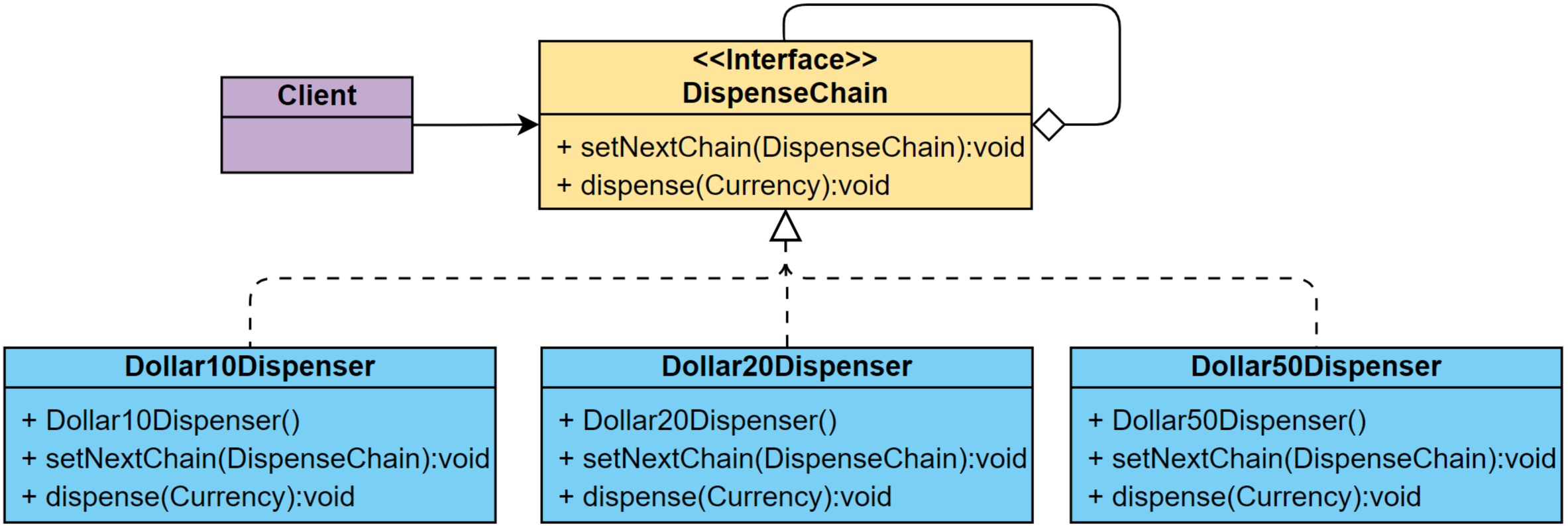


Handler Flow

Class Diagram

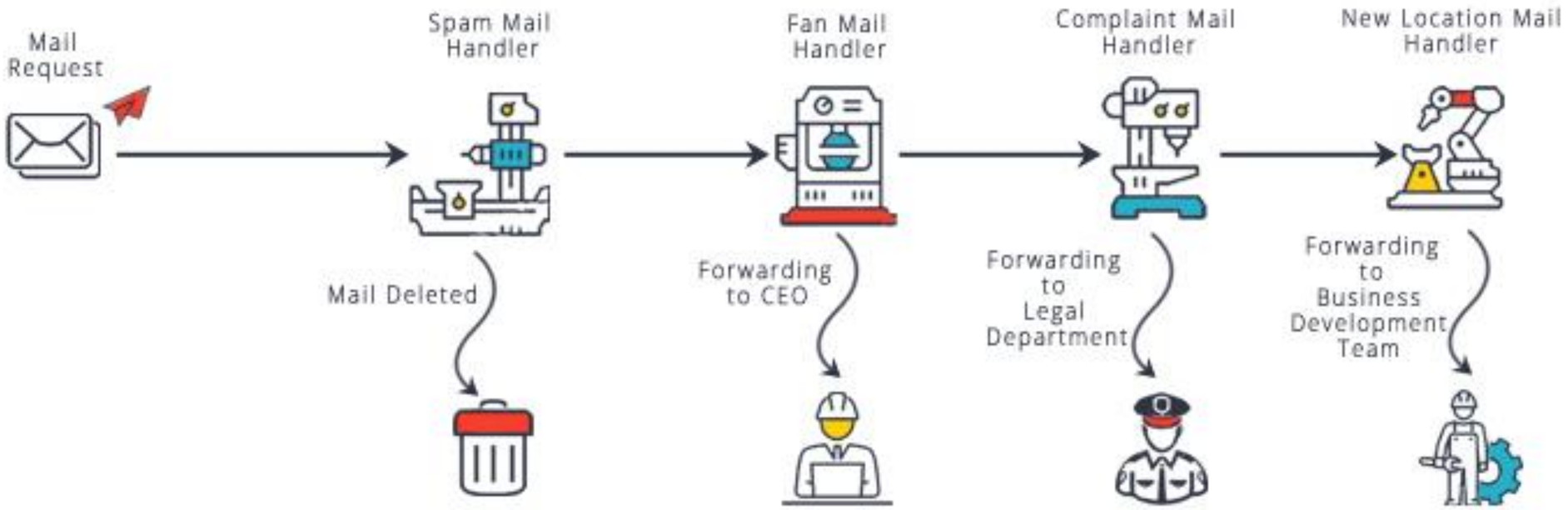
Find the source codes in the course GitHub repository.

CoR Use Case Example: ATM Dispenser App



Find the source codes in the course GitHub repository.

CoR Use Case Example: Mail Forwarding



Assignment 4

Due date: May 28, Friday, 11:59PM

SE 350: OO Software Development

Assignment 4: Design Patterns (2)

Instructor: Vahid Alizadeh

Email: v.alizadeh@depaul.edu

Quarter: Spring 2021

DEPAUL UNIVERSITY
COLLEGE OF COMPUTING AND DIGITAL MEDIA

Last update: May 19, 2021

Any Question

??????????????

How do you feel about the course?

Please Send Your Question or Feedback...

Top **New**