3. Design

RE:WEAR

Student No	22311952
Name	석지윤
E-Mail	owo_8@naver.com

[Revision history]

Revision date	Version #	Description	Author
3/27/2025	1.0.0	First Documentation	석지윤
7/5/2025	1.0.1	Second Documentation	석지윤
9/5/2025	1.1.1	Third Documentation	석지윤
5/6/2025	1.2.1	Fourth Documentation	석지윤

= Contents =

1. Introduction

본 문서는「RE:WEAR」시스템 개발을 위한 세 번째 단계인 Design 문서입니다.
이 문서는 앞선 Analysis 문서에서 도출된 기능 요구사항과 도메인 모델을 바탕으로, 실제 구현 단계에서 활용될 수 있도록 시스템의 구조와 흐름을 구체적으로 설계한 자료입니다.

RE:WEAR는 사용자가 불필요한 의류를 간편하게 기부할 수 있도록 돕는 웹 기반 플랫폼입니다. 사용자는 로그인 후 기부 박스를 신청하고, 포장을 완료한 뒤 원하는 기부처를 선택하거나 새로 등록함으로써 기부 과정을 완료할 수 있습니다. 이러한 일련의 과정을 보다 편리하게 처리할 수 있도록, 본 시스템은 사용자 중심의 직관적인 흐름을 제공하는 것을 목표로 합니다.

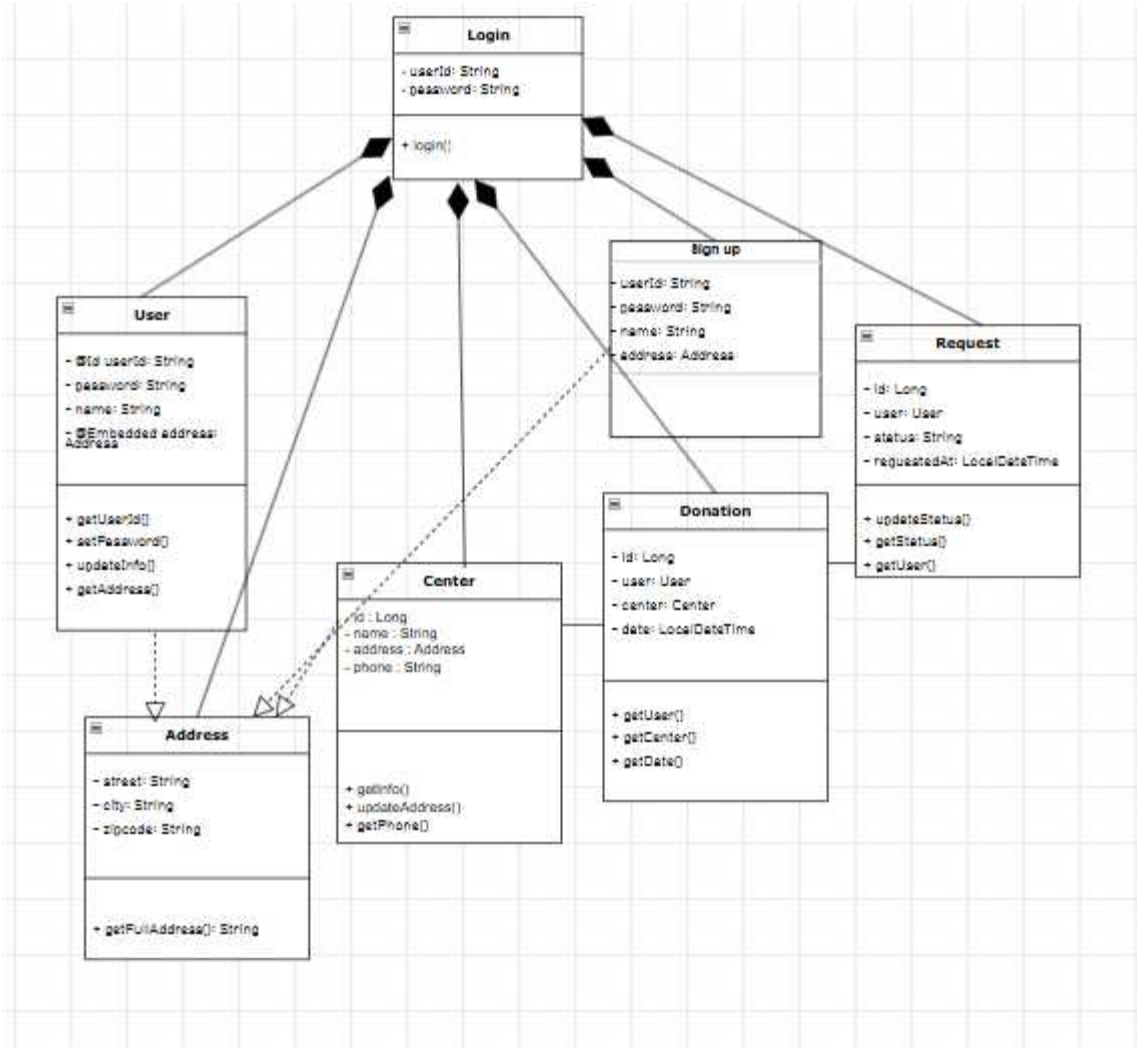
본 문서에서는 전체 시스템을 구성하는 클래스들의 정적 구조를 나타낸 Class Diagram과, 각 기능 수행 시 객체 간의 메시지 흐름을 시간 순으로 보여주는 Sequence Diagram, 그리고 객체의 상태 변화와 전이 조건을 설명하는 State Machine Diagram을 중심으로 설계를 진행하였습니다.

RE:WEAR 설계의 주요 포인트는 다음과 같습니다. 첫째, 사용자 경험을 고려한 기부 프로세스 흐름입니다. 사용자는 별도의 복잡한 절차 없이 웹사이트를 통해 기부를 진행할 수 있도록 구성되어 있습니다.

둘째, 기능별 책임을 명확히 나눈 클래스 설계입니다. 사용자, 기부 요청, 기부처, 기부 내역 등을 도메인별로 세분화하여 시스템의 유지보수성과 확장성을 높였습니다.
셋째, 실제와 유사한 사용자 인터페이스 흐름을 제공하여 UI 설계와 시스템 동작 간의 정합성을 확보하였습니다.

이 문서를 통해 RE:WEAR 시스템의 전체적인 설계 구조를 명확히 이해하고, 향후 구현 단계에서의 개발 방향성을 설정할 수 있을 것입니다.

2. Class diagram



[그림 1] class diagram

2.1. User

Attributes
- @Id userId: String
- password: String
- name: String
- @Embedded address: Address
Methods
+ getUserId()
+ setPassword()
+ updateInfo()
+ getAddress()

2.2. Address

Attributes
- street: String
- city: String
- zipcode: String
Methods
+ getFullAddress(): String

2.3. Login

Attributes
- userId: String
- password: String
Methods
+ login()

2.4. Sign Up

Attributes
- userId: String
- password: String
- name: String
- address: Address
Methods

2.5. Request

Attributes

- id: Long
- user: User
- status: String
- requestedAt: LocalDateTime
Methods
+ updateStatus()
+ getStatus()
+ getUser()

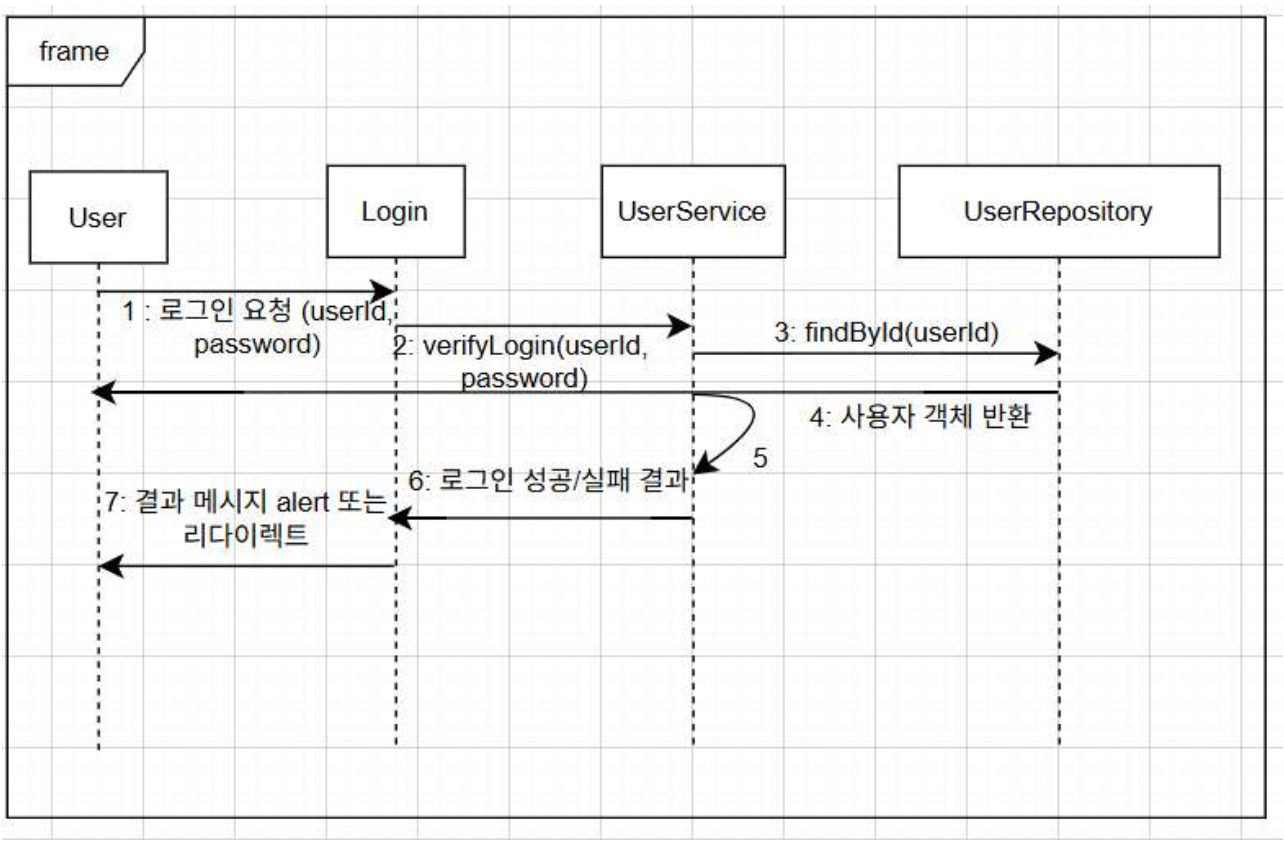
2.6. Center

Attributes
- id: Long
- name: String
- address: Address
- phone: String
Methods
+ getInfo()
+ updateAddress()
+ getPhone()

2.7. Donation

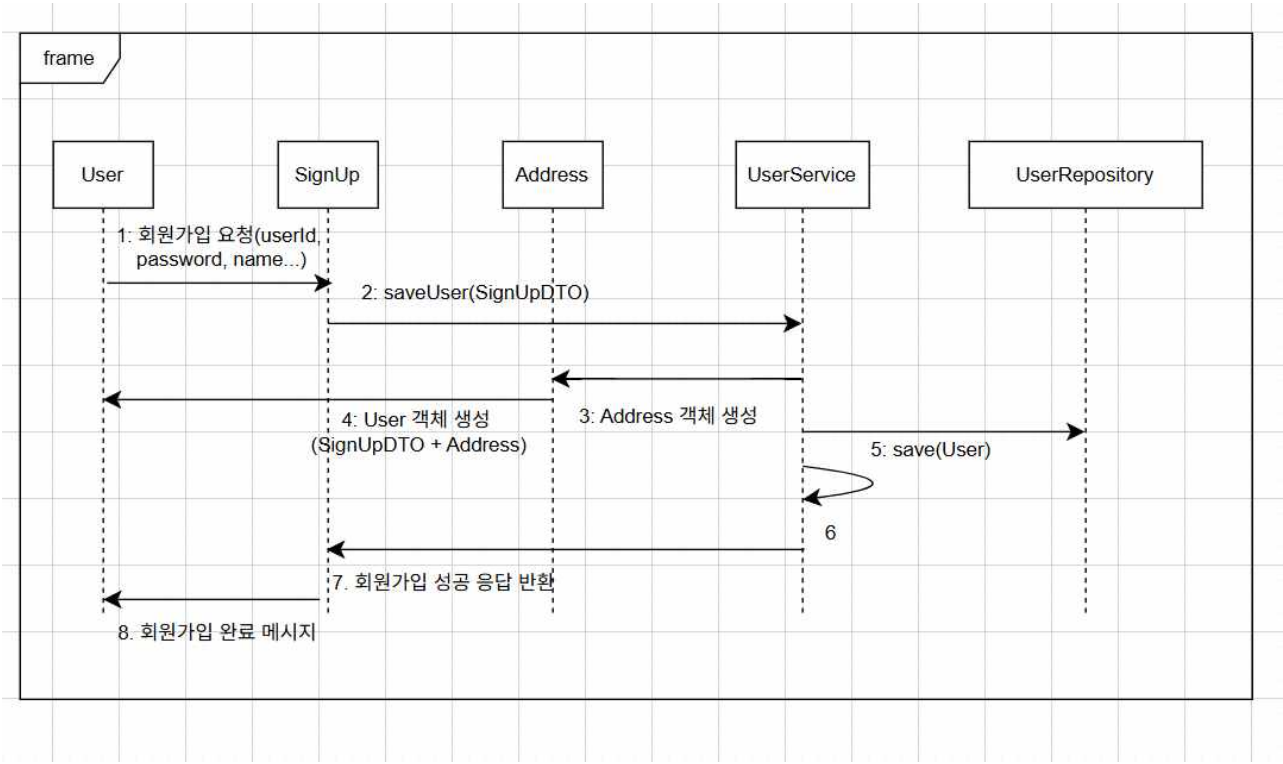
Attributes
- id: Long
- user: User
- center: Center
- date: LocalDateTime
Methods
+ getUser()
+ getCenter()
+ getDate()

3. Sequence diagram



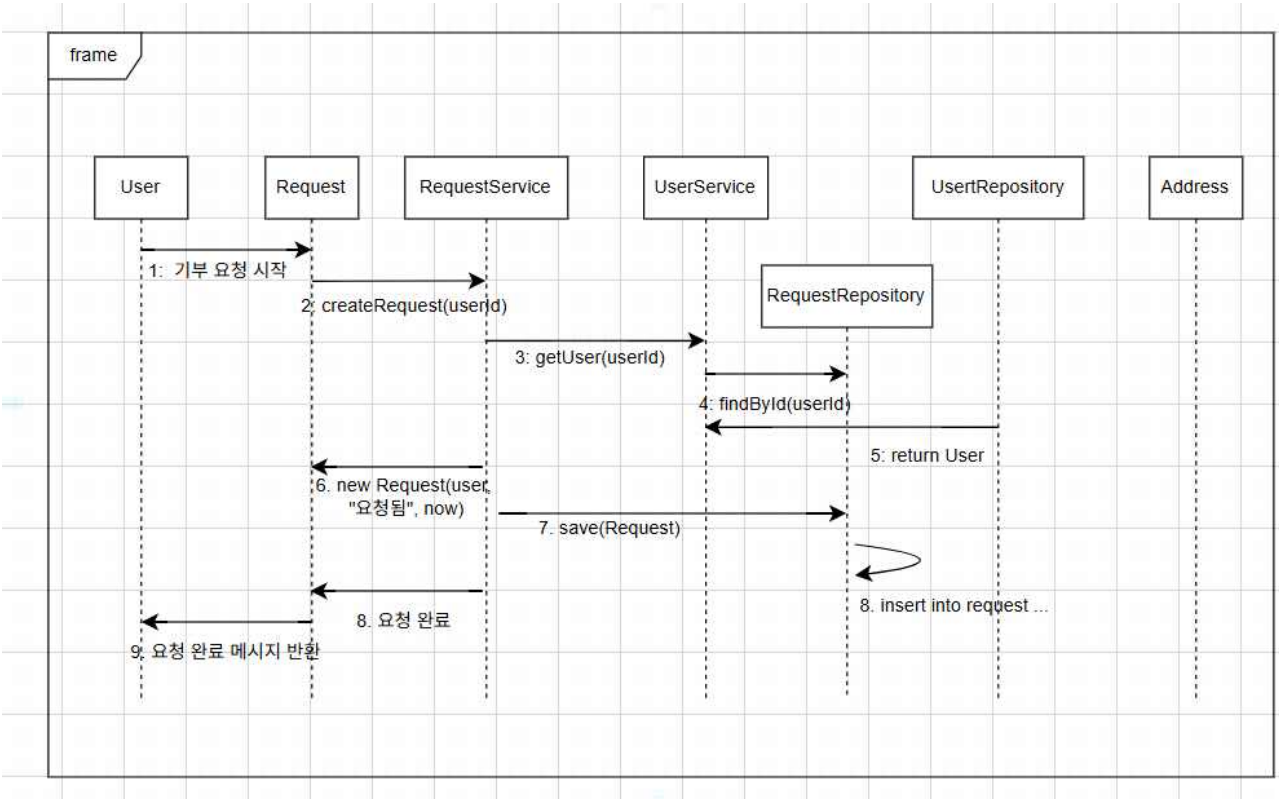
[그림 2-1] Login Sequence Diagram

사용자가 로그인 화면에서 아이디와 비밀번호를 입력하면, 해당 정보가 Login 클래스를 통해 서버로 전달된다.
Login 클래스는 이 정보를 UserService에 전달하여 해당 사용자 ID가 존재하는지를 UserRepository를 통해 조회한다.
조회된 User 객체의 비밀번호가 입력된 값과 일치하는지 확인한 후, 결과를 Login에 반환한다.
비밀번호가 일치할 경우 로그인에 성공하고, 사용자에게 메인 페이지로의 이동 또는 성공 메시지를 전달한다.
일치하지 않으면 오류 메시지를 반환하여 사용자가 다시 로그인할 수 있도록 유도한다.



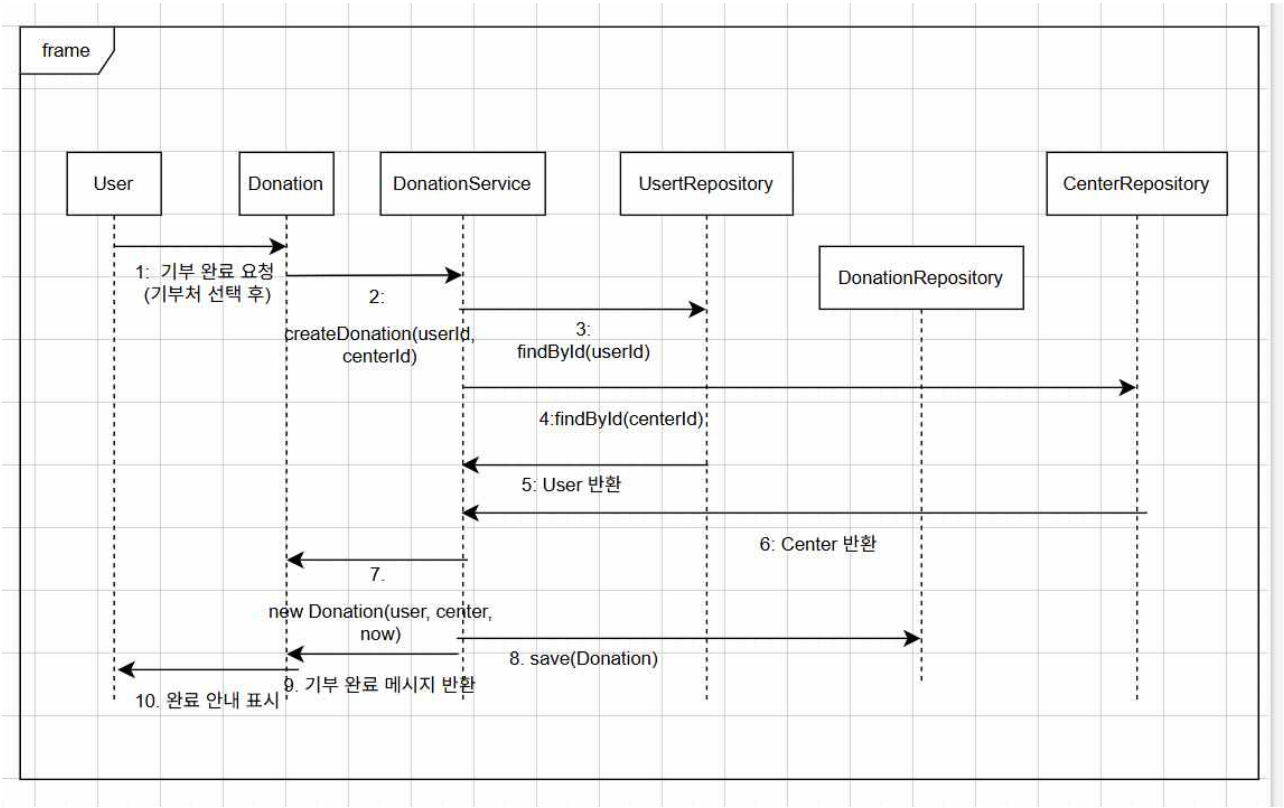
[그림 2-2] SignUp Sequence Diagram

회원가입 화면에서 사용자가 아이디, 비밀번호, 이름, 주소 정보를 입력하고 제출하면 SignUp 클래스가 이를 수신한다.
SignUp 클래스는 해당 정보를 UserService로 전달하고, UserService는 이를 바탕으로 User와 Address 객체를 생성한다.
생성된 User 객체는 UserRepository를 통해 DB에 저장된다.
저장이 완료되면 회원가입이 성공했다는 메시지를 반환하고, 사용자는 로그인 페이지로 리다이렉트된다.
이 과정을 통해 사용자의 계정 정보가 시스템에 정식으로 등록된다.



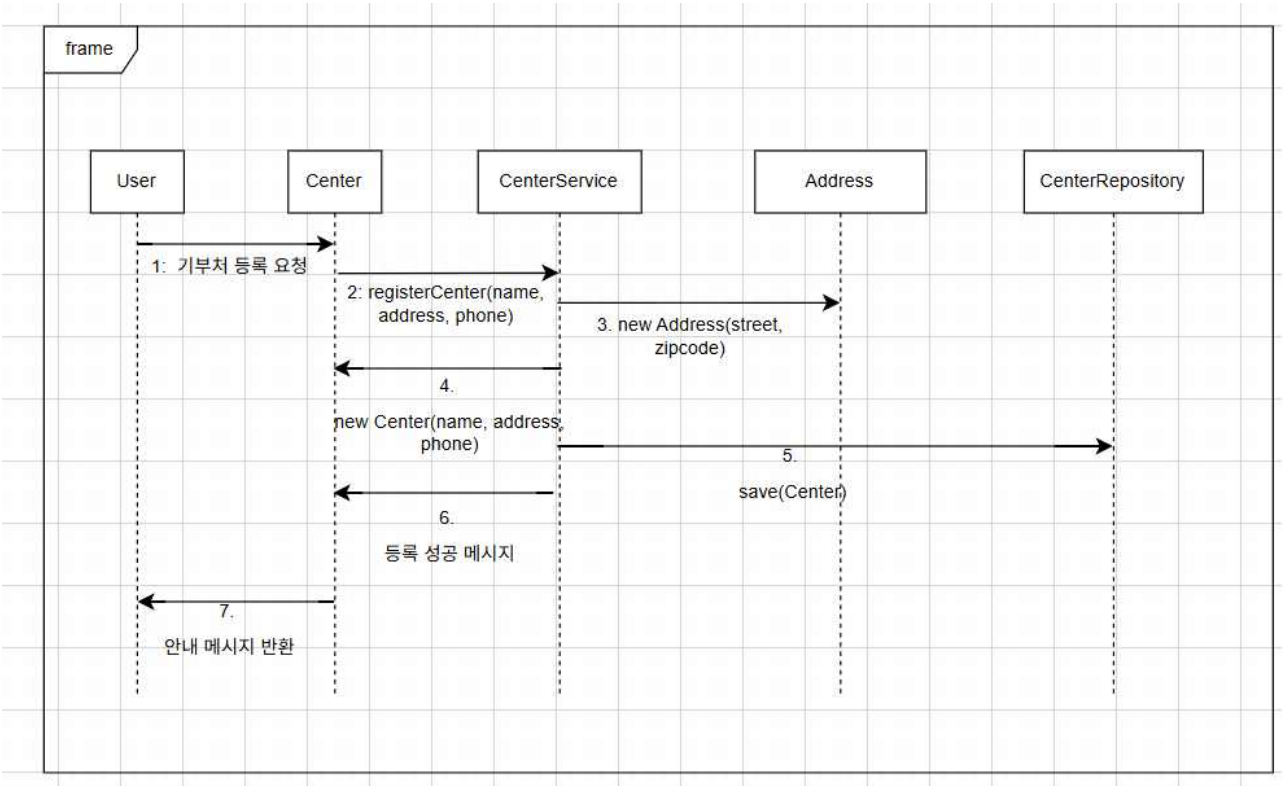
[그림 2-3] Request Sequence Diagram

사용자가 기부 요청 페이지에서 각 단계를 완료하면, Request 클래스는 요청 정보를 RequestService로 전달한다.
RequestService는 먼저 해당 사용자의 정보를 확인하기 위해 UserService를 호출하고, UserRepository에서 사용자 정보를 조회한다.
그 후 사용자 정보를 기반으로 Request 객체를 생성하고, 기부 상태를 '요청됨'으로 설정하여 DB에 저장한다.
저장이 완료되면 사용자는 다음 단계(주소 입력 등)로 진행할 수 있으며, 단계 완료 여부는 화면에 체크 표시로 시각화된다.
이 시퀀스를 통해 사용자의 기부 흐름이 서버에 기록되고 추적 가능하게 된다.



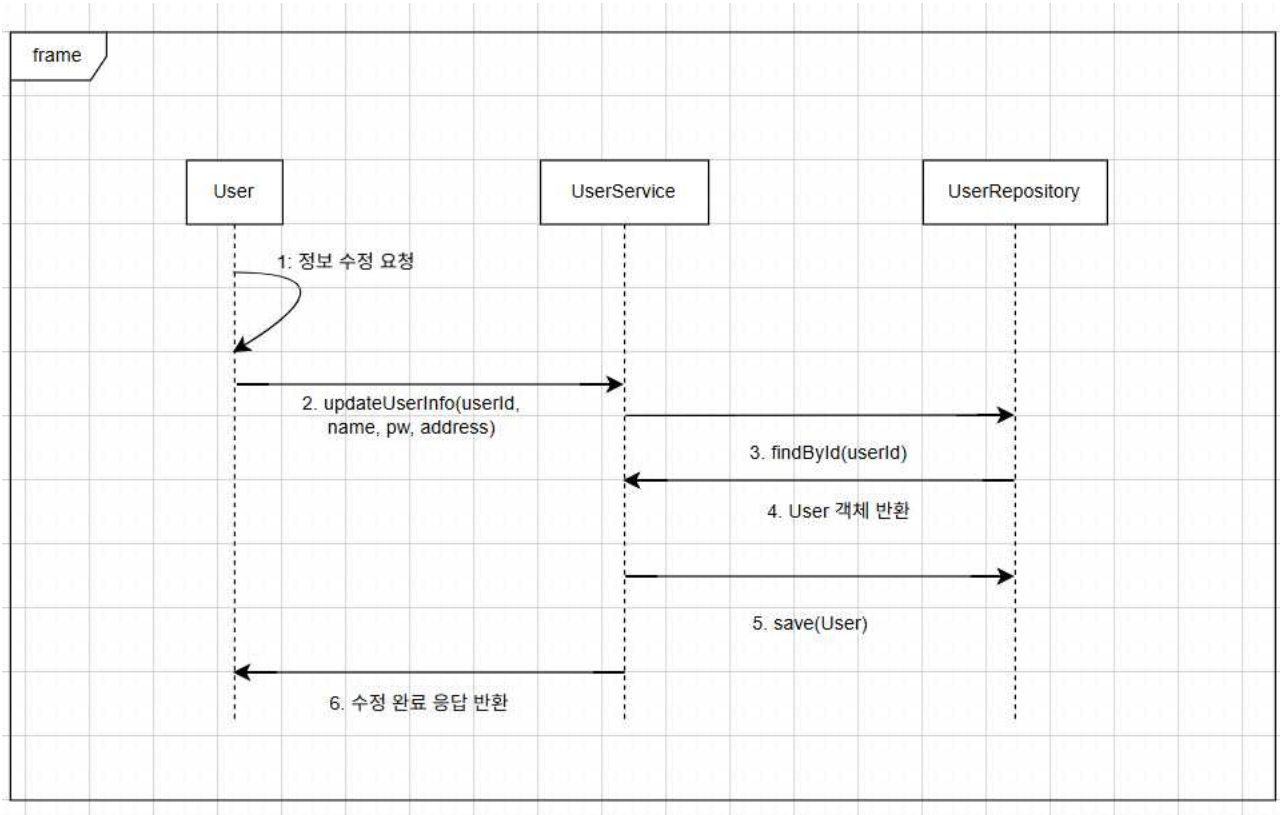
[그림 2-4] Donation Sequence Diagram

기부 요청이 완료된 사용자가 기부처를 선택하고 '기부 완료'를 클릭하면 Donation 클래스가 해당 정보를 DonationService로 전달한다.
DonationService는 먼저 userId와 centerId를 기준으로 각각 UserRepository와 CenterRepository에서 사용자와 기부처 정보를 조회한다.
조회된 객체들을 기반으로 Donation 객체가 생성되며, 기부 일자는 현재 시각으로 설정된다.
생성된 Donation 객체는 DonationRepository를 통해 DB에 저장되며, 성공 메시지가 사용자에게 전달된다.
이 과정을 통해 사용자의 기부 내역이 기록되며 마이페이지에서 조회할 수 있게 된다.



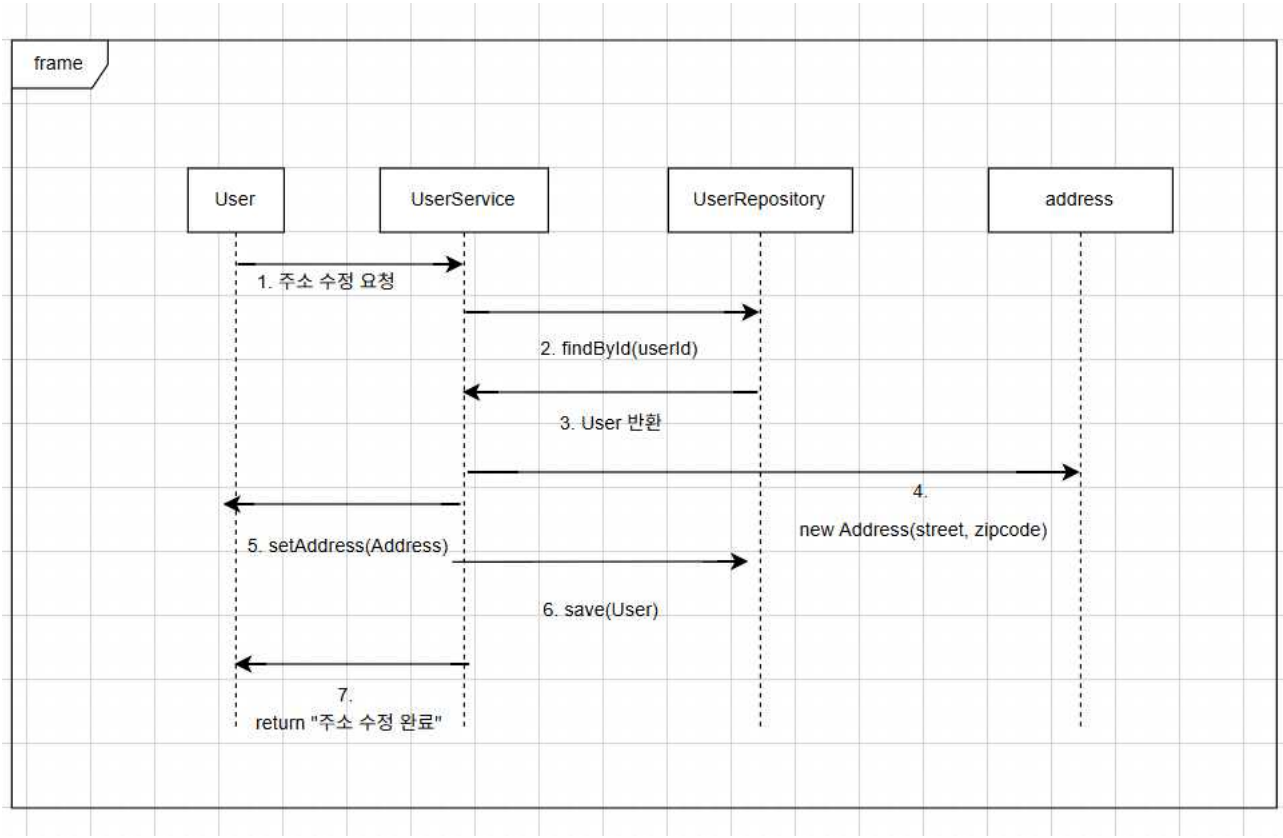
[그림 2-5] Center Sequence diagram

사용자가 센터 등록 페이지에서 센터 이름, 주소, 전화번호를 입력하고 제출하면 Center 클래스가 CenterService를 호출한다.
CenterService는 Address 객체를 먼저 생성하고, 이를 포함한 Center 객체를 생성한다.
완성된 Center 객체는 CenterRepository를 통해 DB에 저장된다.
저장 성공 후, 등록 완료 메시지가 사용자에게 전달되며, 목록에 새로운 기부처가 표시된다.
이 시퀀스는 사용자 또는 관리자가 새로운 기부처를 추가하는 흐름을 정의한다.



[그림 2-6] User Sequence Diagram

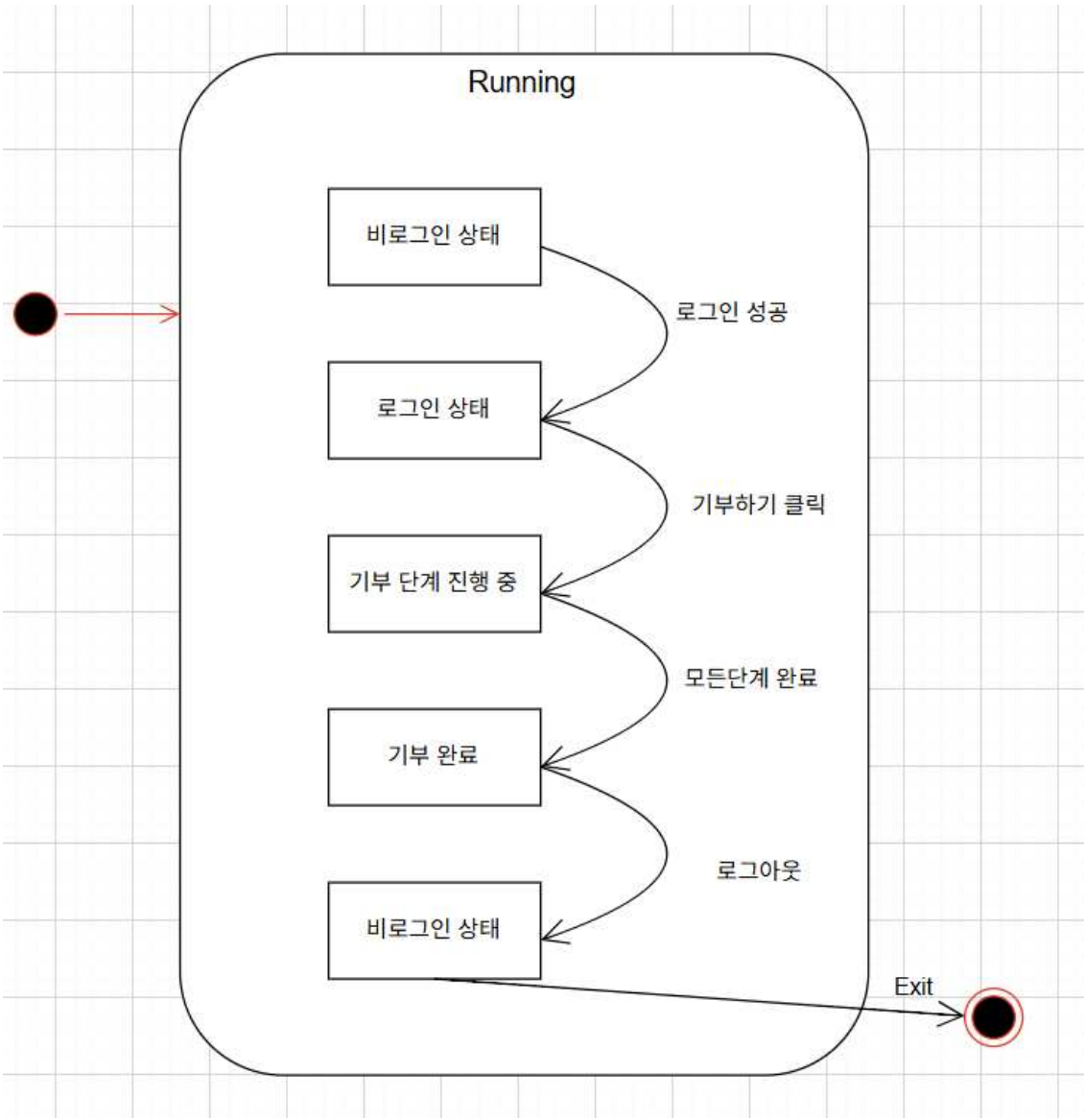
로그인한 사용자가 마이페이지에서 이름, 비밀번호, 주소를 수정하고 '수정 완료'를 클릭하면 UserService로 정보가 전달된다.
UserService는 먼저 사용자 ID를 기반으로 UserRepository에서 기존 정보를 불러온다.
이후 수정된 name, password, address 값을 set 메서드를 통해 User 객체에 반영하고, 다시 UserRepository를 통해 저장한다.
업데이트가 완료되면 사용자에게 수정 완료 메시지가 표시되며 변경 사항이 반영된다.
이 시퀀스는 사용자의 개인정보를 수정하는 기능의 내부 동작 흐름을 나타낸다.



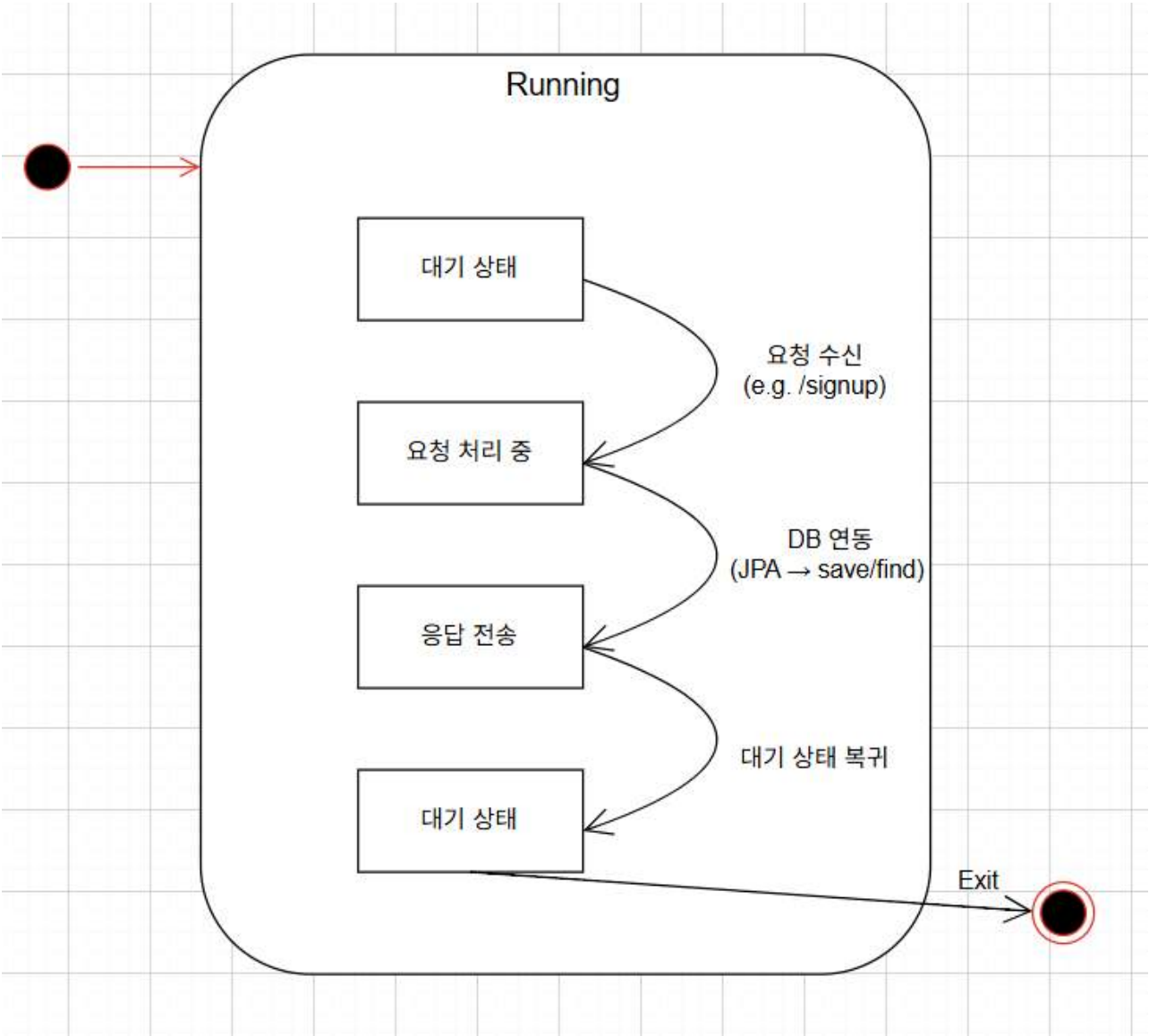
[그림 2-7] Address Sequence Diagram

Address 클래스는 User 또는 Center 클래스에 내장되어 있으며, 주소 관련 데이터의 생성 및 수정에 사용된다.
주소 수정 요청이 발생하면 UserService 또는 CenterService에서 new Address 객체를 생성한다.
새로 생성된 Address 객체는 setAddress() 메서드를 통해 User 또는 Center 객체에 주입된다.
이후 해당 엔티티는 Repository를 통해 저장되고, 변경된 주소 정보가 DB에 반영된다.
이 시퀀스는 주소 데이터가 독립적으로 처리되지 않고, 포함된 엔티티와 함께 변경되는 구조를 보여준다.

4. State machine diagram



RE:WEAR의 클라이언트 상태 다이어그램은 사용자가 웹 플랫폼을 통해 서비스를 이용할 때의 주요 상태 흐름을 표현한다.
사용자는 초기에는 비로그인 상태이며, 로그인 성공 시 로그인 상태로 전이된다.
로그인한 사용자는 '기부하기'를 통해 기부 단계를 순차적으로 진행하며, 각 단계가 완료되면 다음 상태로 이동한다.
모든 기부 단계가 완료되면 사용자는 '기부 완료' 상태로 전환되며, 이후 로그아웃을 수행할 경우 다시 비로그인 상태로 돌아간다.
이 상태 흐름은 로그인, 기부 신청, 완료, 로그아웃까지 사용자 중심의 인터랙션을 시각화하며, 직관적인 UX 흐름을 설계하는 데 도움을 준다.



RE:WEAR 서버 측 상태 다이어그램은 Spring Boot 기반 시스템이 사용자 요청을 처리하는 과정의 상태 흐름을 표현한다.
서버는 기본적으로 대기 상태에 있으며, 사용자의 HTTP 요청(/login, /signup, /donation 등)을 수신하면 요청 처리 상태로 전이된다.
요청 처리 중에는 필요한 비즈니스 로직을 수행하며, User, Request, Donation 등과 관련된 DB 연산이 포함된다.
처리가 완료되면 응답 상태로 전환되어 결과 데이터를 사용자에게 반환하고, 이후 다시 대기 상태로 복귀한다.
이 흐름은 RESTful 서비스 구조에서 서버의 핵심 역할과 요청-응답 사이클을 명확히 보여주며, 백엔드 시스템의 안정적인 처리를 시각화한다.

5. Implementation requirements

5.1. H/W Platform Requirements

CPU : Intel i3 이상
RAM : 4GB 이상
HDD / SSD : 10GB 이상
Network : 인터넷 연결 필수

5.2. S/W Platform Requirements

OS : Microsoft Windows 7 이상
Implementation Language : JAVA

구분	세부 항목	내용
시스템 구조	플랫폼	Web 기반 시스템 (PC/모바일 반응형)
프론트엔드	언어	HTML5, CSS3, JavaScript
백엔드	프레임워크	Spring Boot, Java
데이터베이스	RDBMS	MySQL 8.0 이상

6. Glossary

Terms	Description
RE:WEAR	본 프로젝트의 명칭으로, "Re-wear(다시 입다)"의 의미를 결합하여, 의류 재사용과 기부를 통해 지속 가능한 사회를 만들고자 하는 의지를 담고 있음.
기부 요청	사용자가 시스템을 통해 의류 기부를 신청하는 과정
포장 승인	사용자가 기부할 의류를 포장 완료한 후, 시스템이 이를 확인하고 승인하는 과정
승인 상태 전달	사용자의 요청이 처리되었는지 여부를 사용자에게 알려주는 기능 (예: 승인 완료, 대기 중, 거부됨)
배송 추적	기부된 의류가 기부처로 이동하는 과정에서 배송 상태를 확인하는 기능
기부처	사용자가 기부한 의류를 전달받아 필요한 사람들에게 제공하는 기관 (예: 아동센터, 복지시설, 학교 등)
Attribute	객체의 상태를 나타내며, 데이터 또는 속성이라고 함
Class Diagram	"Class"라고 하는 객체지향 설계단위를 이용하여 시스템의 정적인 structure를 표현한 다이어그램
Method	객체가 메시지를 받아 실행해야 할 객체의 구체적인 연산을 정의한 것, 전통적 시스템의 함수나 프로시저에 해당하는 연산 기능
Operation	객체의 행동(behavior) 및 다른 객체에 대하여 제공하는 것
Sequence Diagram	객체간의 동적 상호작용을 시간적 개념을 중심으로 모델링 하는 과정
Sate Machine Diagram	객체 lifetime동안의 변환될 수 있는 모든 상태를 정의해둔 다이어그램

7. References

1. https://www.drawio.com/ - class diagram 제작,
Sequence diagram 제작,
State machine diagram 제작