CRUD Application

[ ] [ ]

[ ] [ ]

(Create) (Update) (Delete) (Reset)

			Edit Delete
			Edit Delete
			Edit Delete

CrudController

@/crud/index
@/crud/edit/{id}
@/crud/create
@/crud/update
@/crud/delete/{id}

CrudDAO

Table

CRUD Application Implementation

4. Lập trình ứng dụng

<<Interface>>
JpaRepository

3. Truy xuất dữ liệu

@Controller
UserController

<<Interface>>
UserDAO

Users

@Entity
User

2. Mô tả cấu trúc dữ liệu

1. Dữ liệu
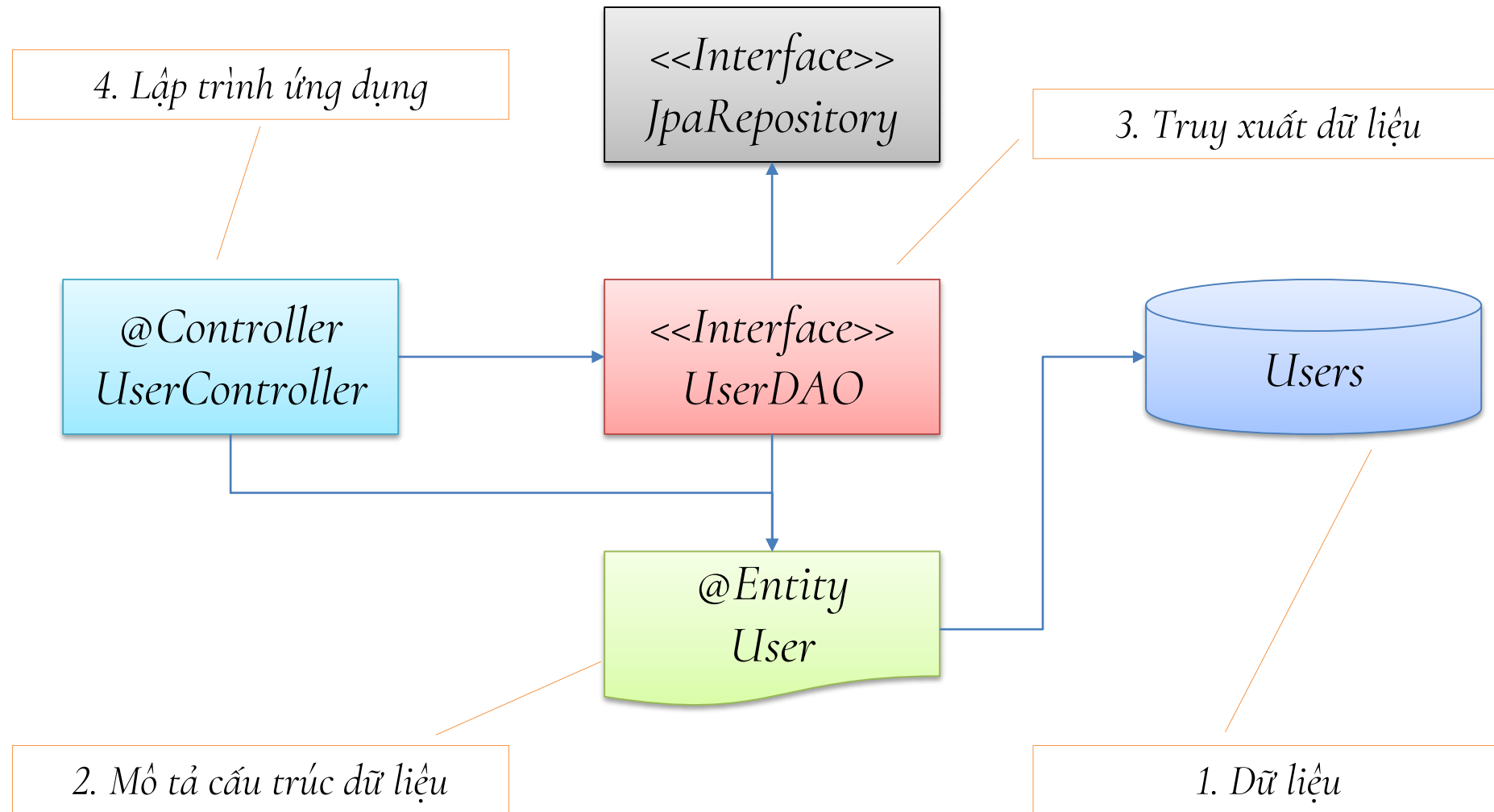
```java
@Controller
public class UserController {
    @Autowired
    UserDAO dao;
    @RequestMapping("/url")
    public String  method(){
        List<User> list = dao.findAll();
        User item = dao.getById(1000);
        User item = dao.save(item);
        dao.deleteById(1000);
        ...
    }
}
```

(4)

```java
@Data
@Entity @Table(name="Users")
public class User{
    @Id
    @Column(name="Username")
    String username;
    @Column(name="Password")
    String password;
    @Column(name="Activated")
    Boolean activated;
}
```

(2)

```java
public interface UserDAO extends
JpaRepository<User, String>{}
```

(3)

(1)

Column	Type
Username	VARCHAR(50)
Password	VARCHAR(50)
Activated	BIT

```xml
<!-- Spring Data JPA -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<!-- SQL Server Driver -->
<dependency>
    <groupId>com.microsoft.sqlserver</groupId>
    <artifactId>mssql-jdbc</artifactId>
    <scope>runtime</scope>
</dependency>
```

```
#SQL Server Driver
spring.datasource.driver-class-name=com.microsoft.sqlserver.jdbc.SQLServerDriver
spring.datasource.url=jdbc:sqlserver://localhost;database=DB
spring.datasource.username=sa
spring.datasource.password=******

#Spring Data JPA/Hibernate
spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=none
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.SQLServer2012Dialect
```
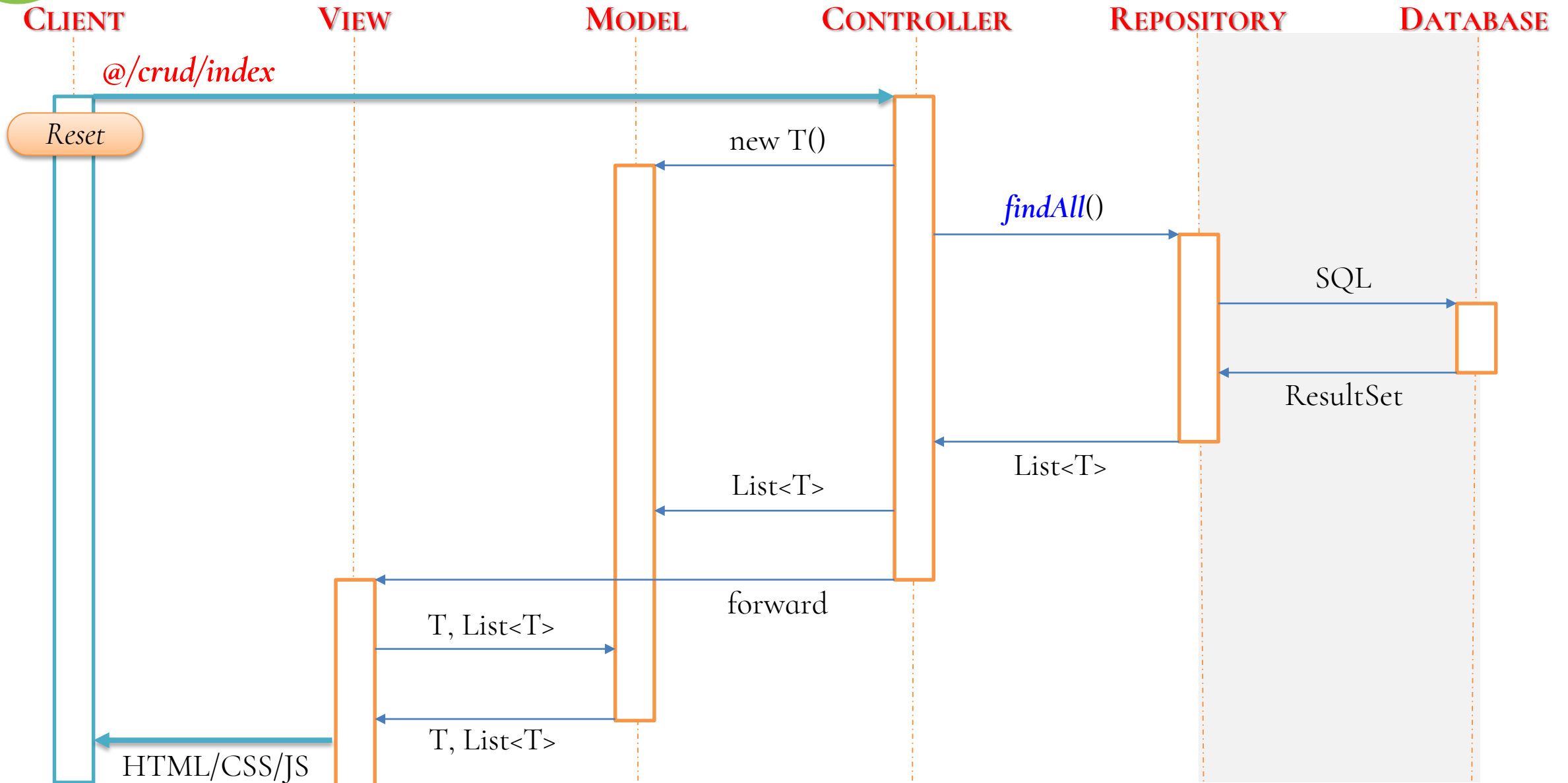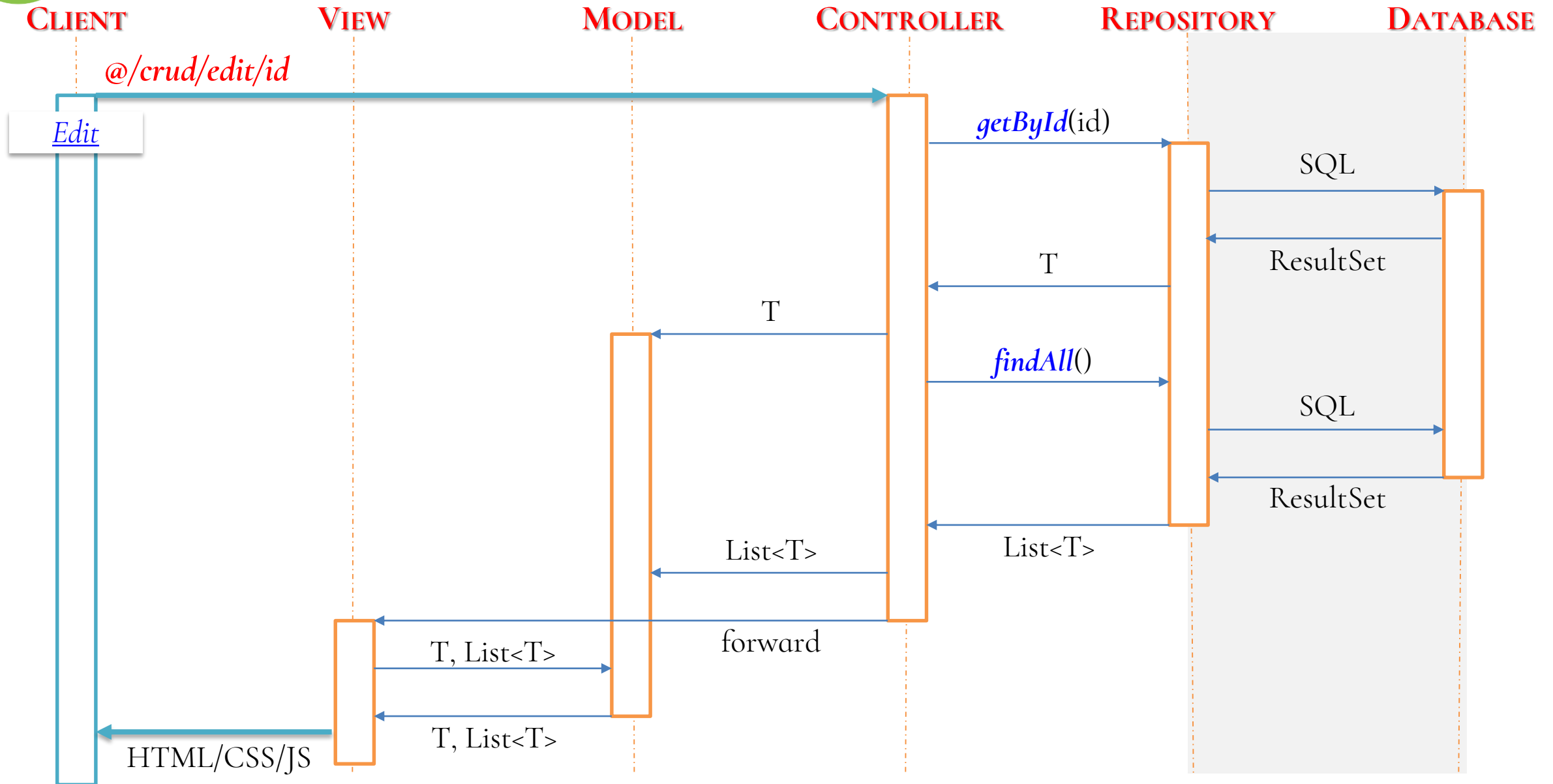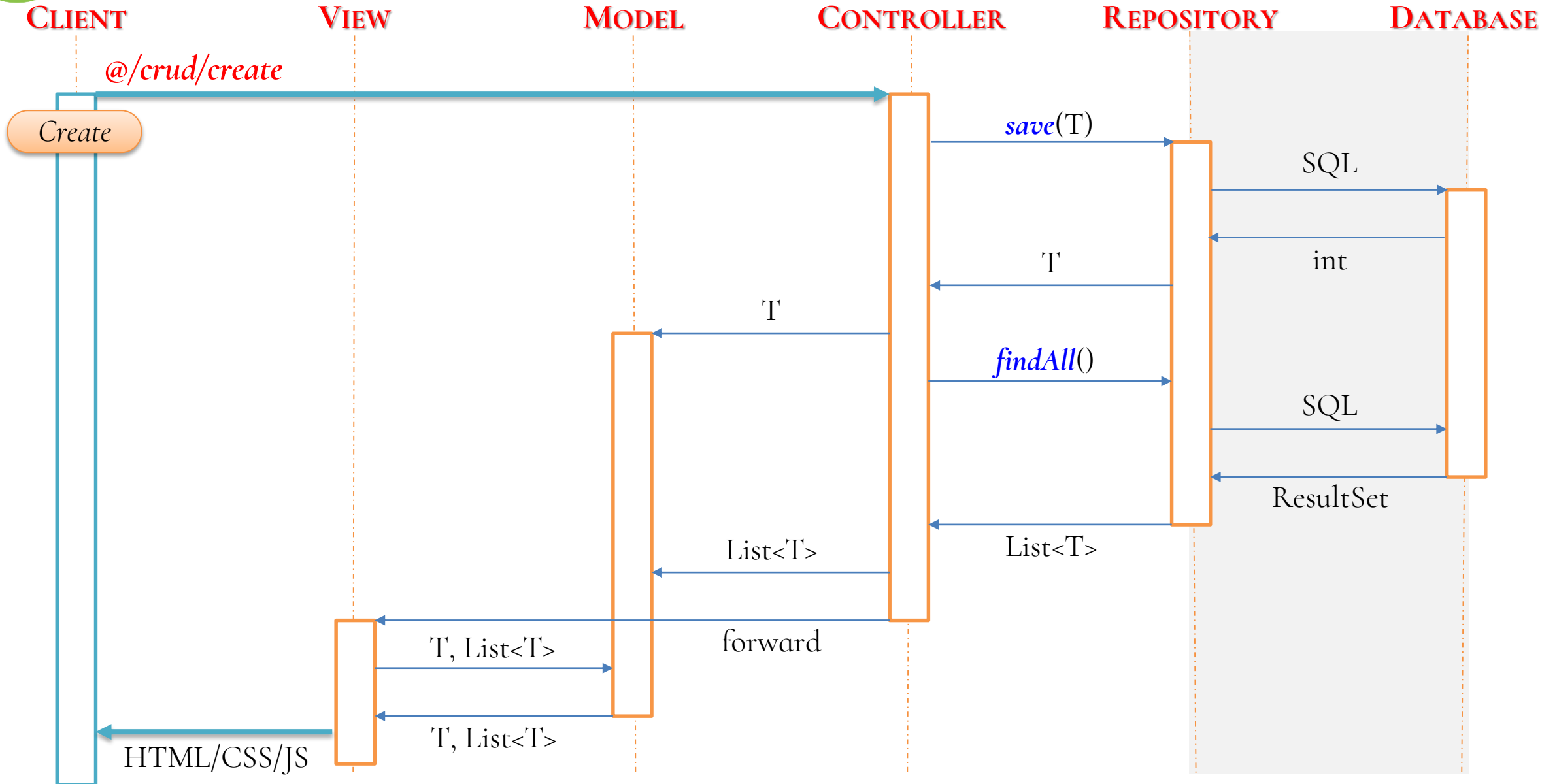
Demo

SEQUENCE DIAGRAM - @/CRUD/INDEX

CLIENT | VIEW | MODEL | CONTROLLER | REPOSITORY | DATABASE

@/crud/index

Reset

new T()

findAll()

SQL

ResultSet

List<T>

List<T>

forward

T, List<T>

T, List<T>

HTML/CSS/JS

CLIENT **VIEW** **MODEL** **CONTROLLER** **REPOSITORY** **DATABASE**

@/crud/edit/id

Edit

getById(id)

SQL

ResultSet

T

T

findAll()

SQL

ResultSet

List<T>

List<T>

forward

T, List<T>

T, List<T>

HTML/CSS/JS

SEQUENCE DIAGRAM - @/CRUD/UPDATE

CLIENT   **VIEW**   **MODEL**   **CONTROLLER**   **REPOSITORY**   **DATABASE**

@/crud/update

Update

save(T)

SQL

int

T

T

findAll()

SQL

ResultSet

List<T>

List<T>

forward

T, List<T>

T, List<T>

HTML/CSS/JS
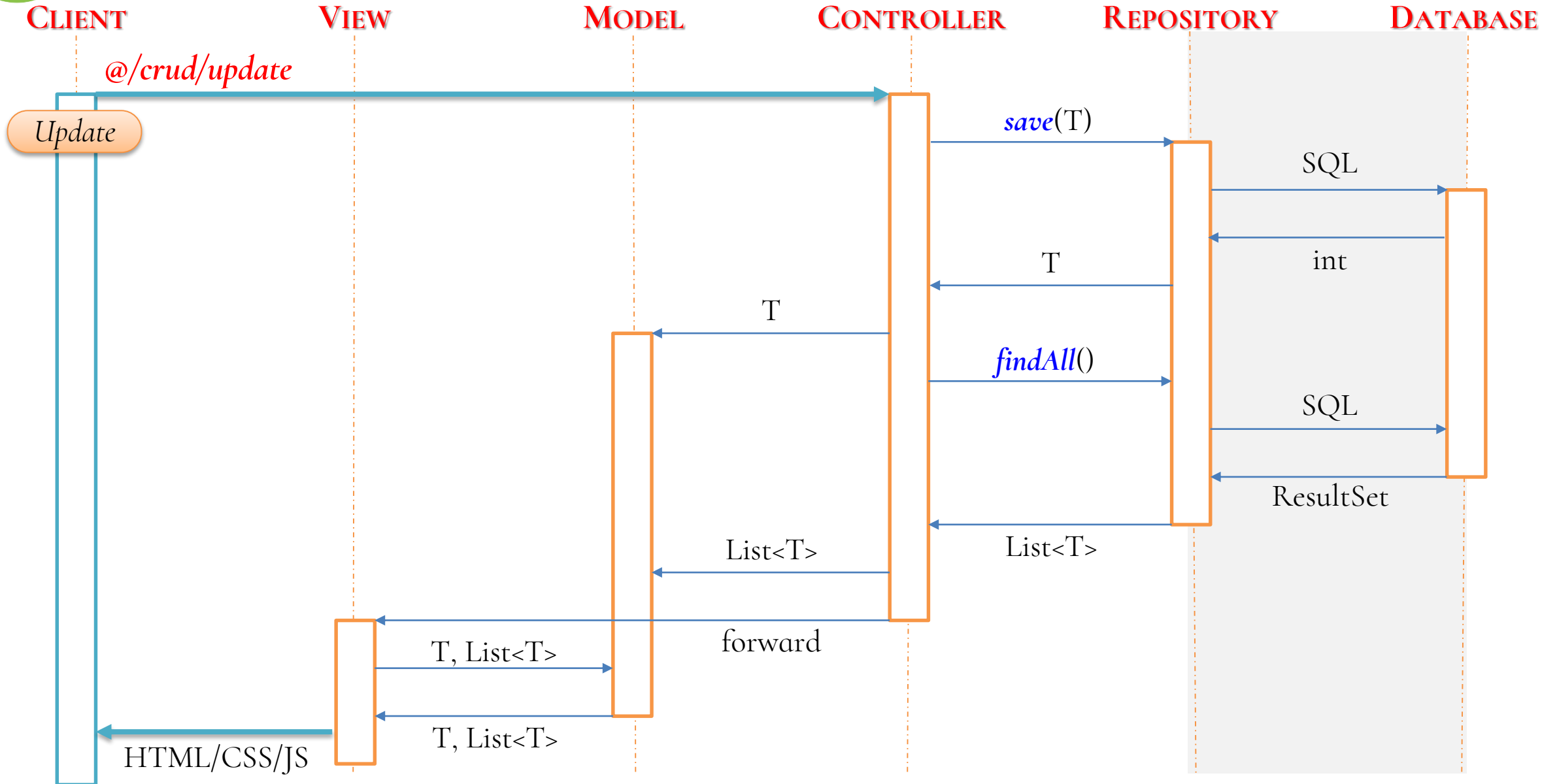
SEQUENCE DIAGRAM - @/CRUD/DELETE/{ID}

CLIENT   VIEW   MODEL   CONTROLLER   REPOSITORY   DATABASE

@/crud/delete/id

Delete

Delete

deleteById(id)

SQL

int

new T()

findAll()

SQL

ResultSet

List<T>

List<T>

forward

T, List<T>

T, List<T>

HTML/CSS/JS
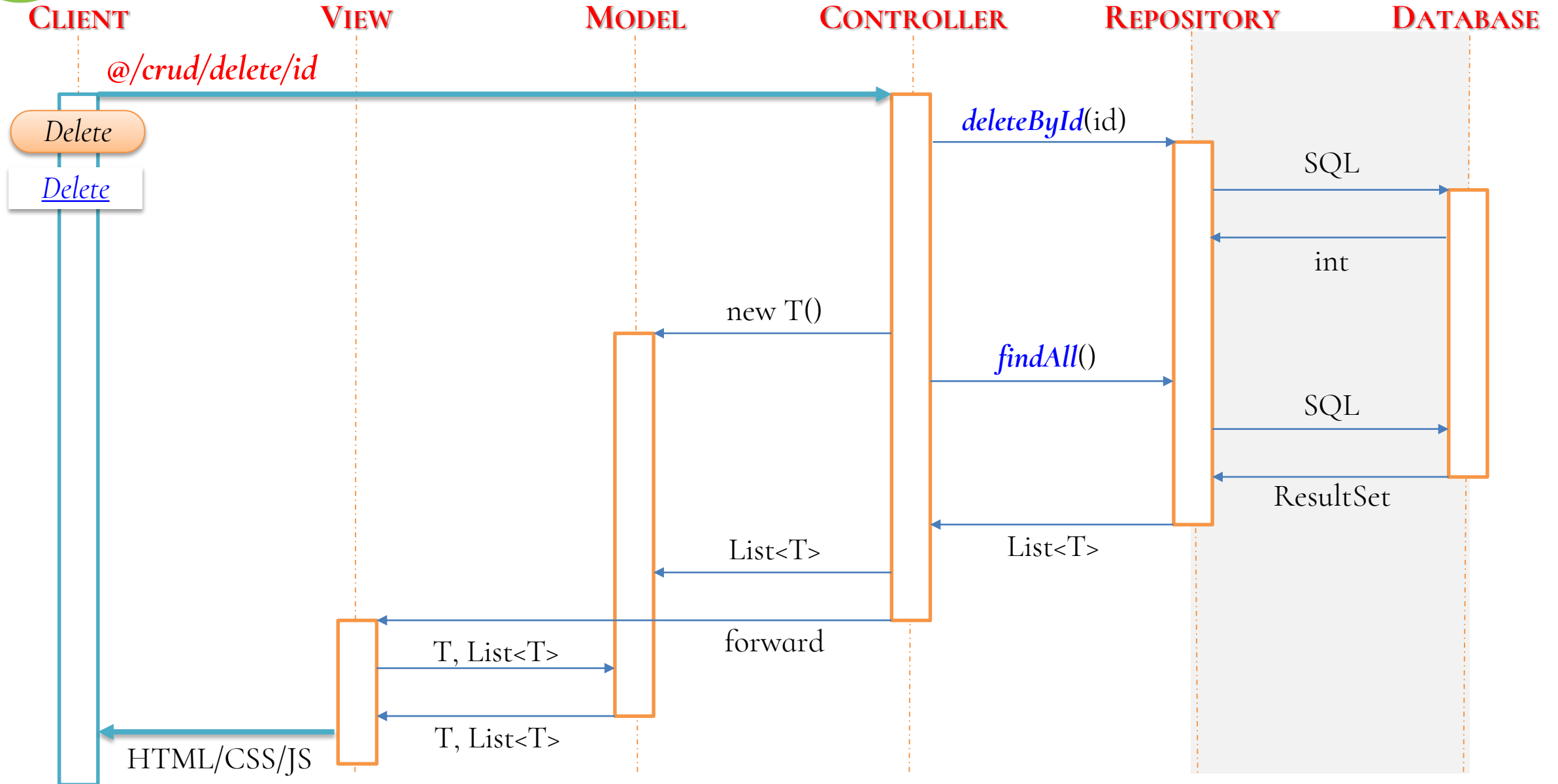
Truy vấn

- ✓ **findAll**(): List<T>
- ✓ *findAll*(Sort): List<T>
- ✓ *findAll*(Pageable): Page<T>
- ✓ **getById**(ID): T
- ✓ *findById*(ID): Optional<T>
- ✓ *existsById*(ID): Boolean
- ✓ *count*(): long

Thao tác

- ✓ **save**(T): T
- ✓ *saveAll*(Iterable<T>): Iterable<T>
- ✓ **deleteById**(ID)
- ✓ *delete*(T)
- ✓ *deleteInBatch*(Iterable<T>)

- ▪ *T là kiểu của @Entity*
- ▪ *ID là kiểu của @Id*
- ▪ ***Sort & Pageable** là sắp xếp và phân trang (nghiên cứu sau)*

☑ **TRUY VẤN**

- ✓ *findAll(): List<T>*
- ✓ *getById(ID): T*
- ✓ *findById(ID): Optional<T>*
- ✓ *existsById(ID): Boolean*
- ✓ *count(): long*

☑ **THAO TÁC**

- ✓ *save(T): T*
- ✓ *saveAll(Iterable<T>): Iterable<T>*
- ✓ *delete(T)*
- ✓ *deleteInBatch(Iterable<T>)*
- ✓ *deleteById(ID)*