



Prorg (Agile Project Management Tool)

Ankit Srivastava, Rohit Mehra, Nikhil Sulegaon

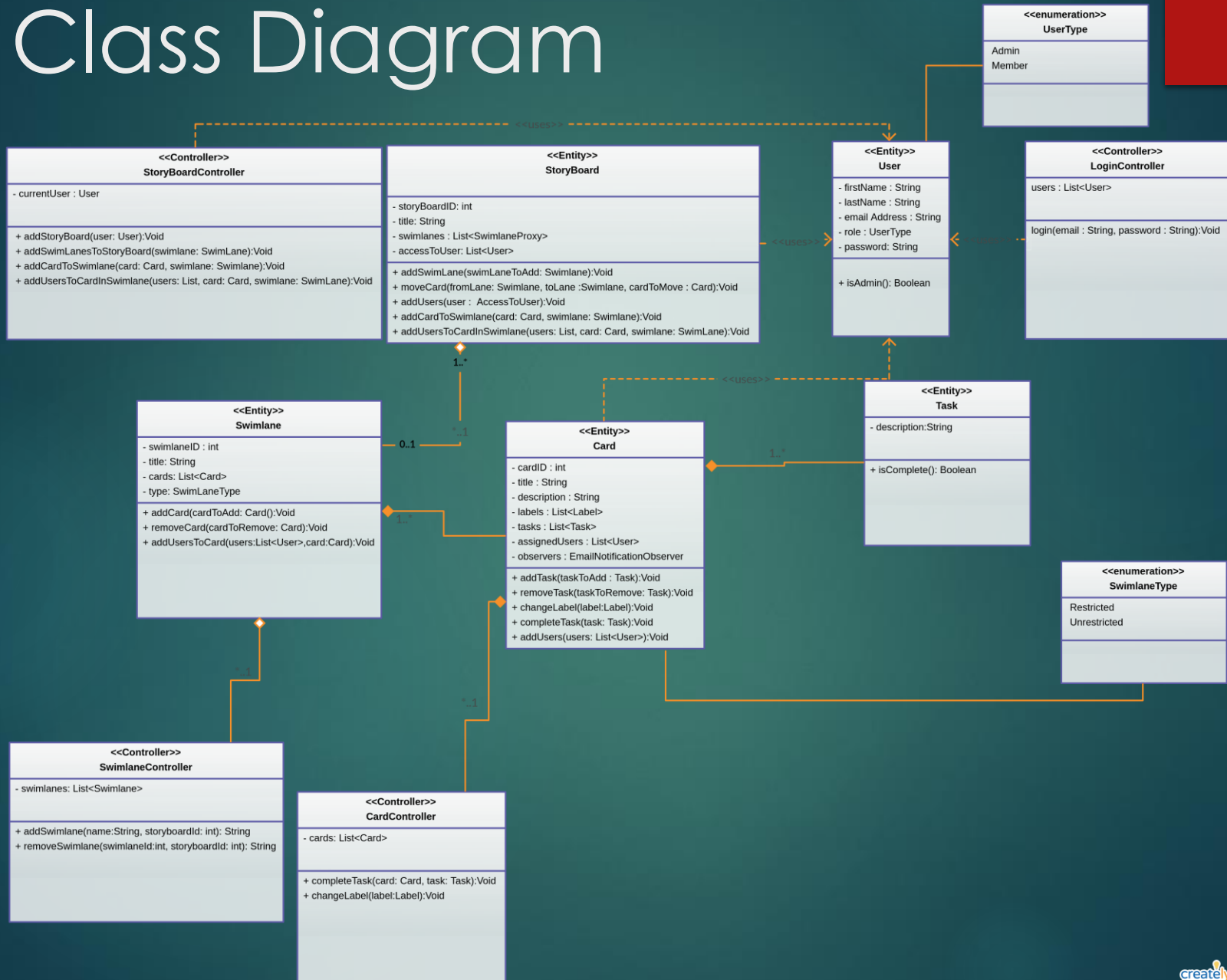
Introduction about Prorg



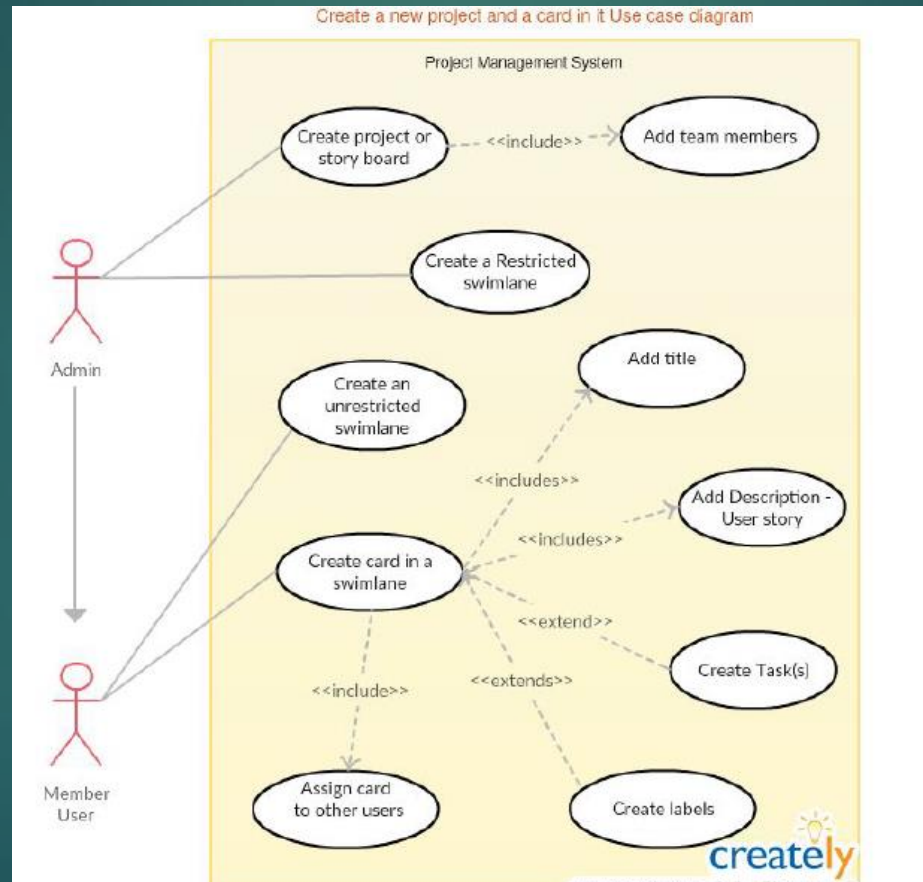
Prorg is a web-based collaboration tool that organizes projects into boards. In one glance, Prorg tells you what's being worked on, who's working on what, and where something is in a process.

Imagine a white board, filled with lists of sticky notes, with each note as a task for you and your team. Now imagine that each of those sticky notes has photo-attachments. Now imagine that you can take that whiteboard anywhere you go, and can access it from any computer through the web. That's Prorg!

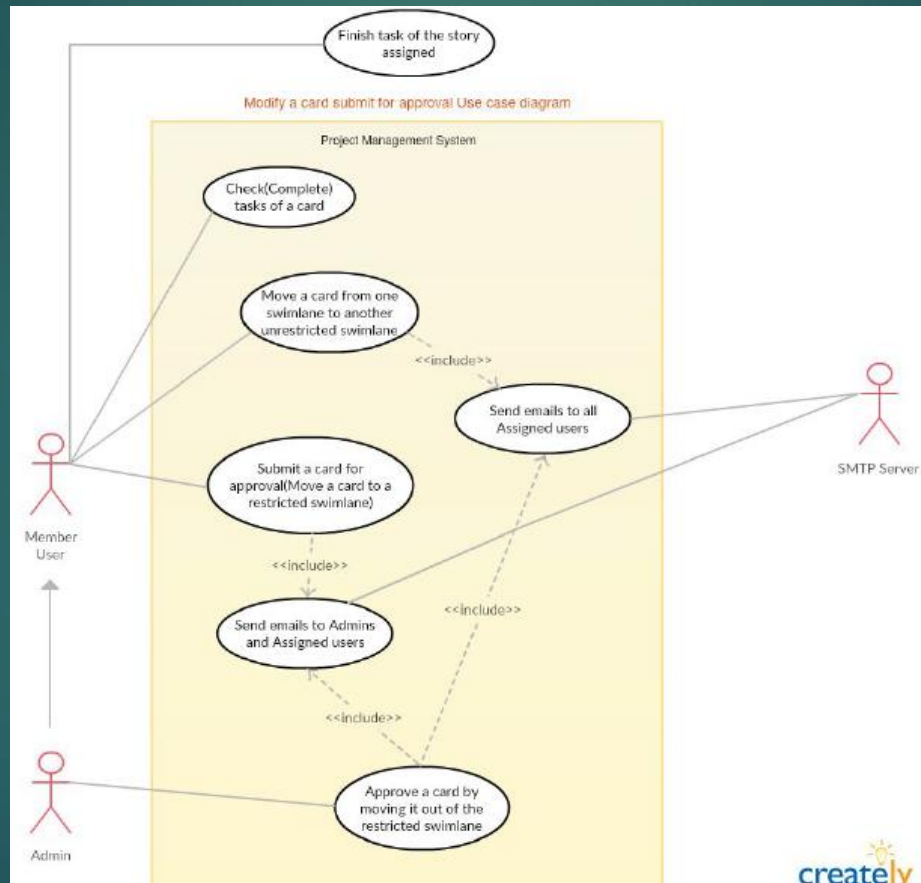
Class Diagram



Use Case Diagram-1



Use Case Diagram-2

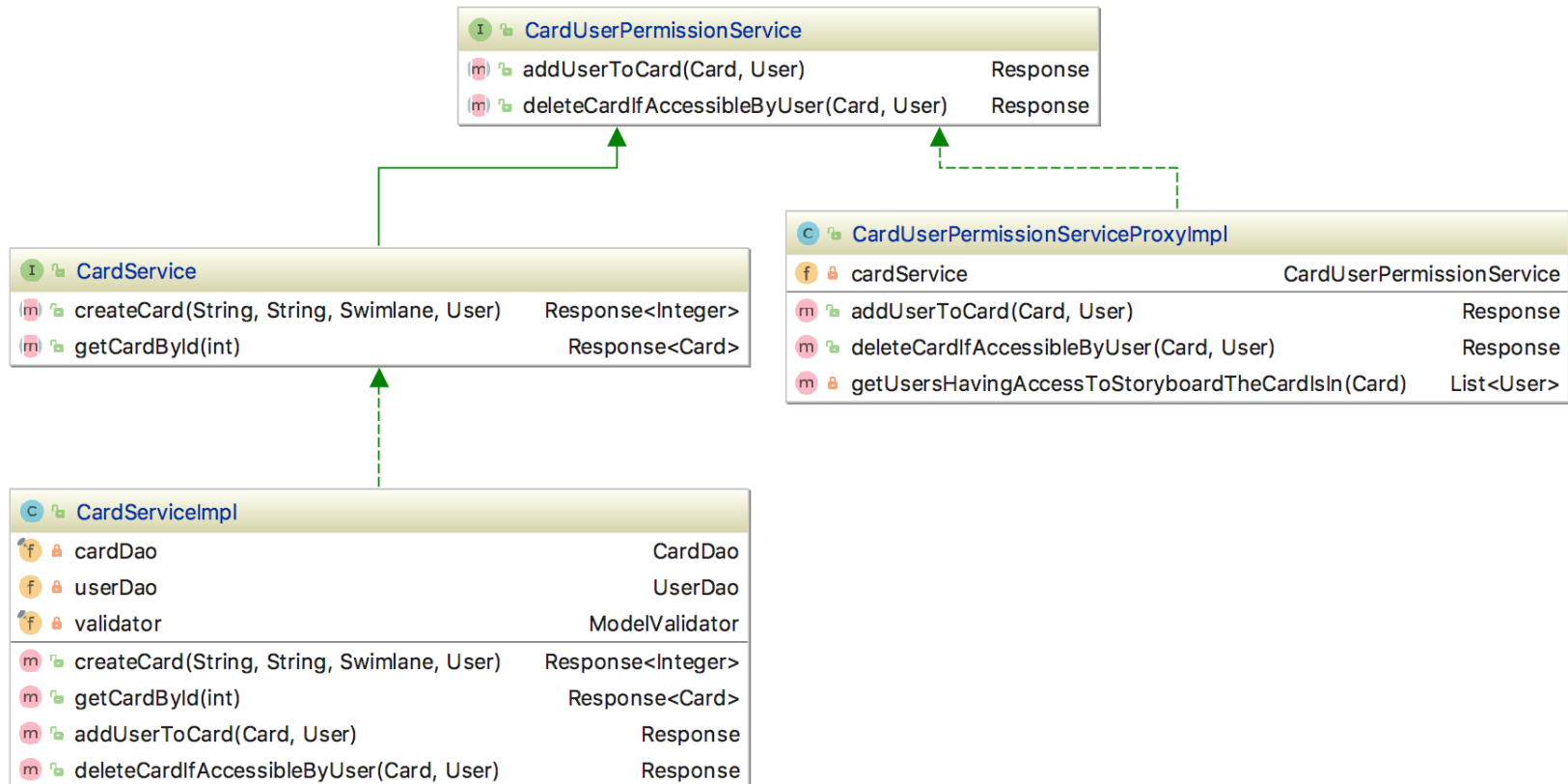


DEMO



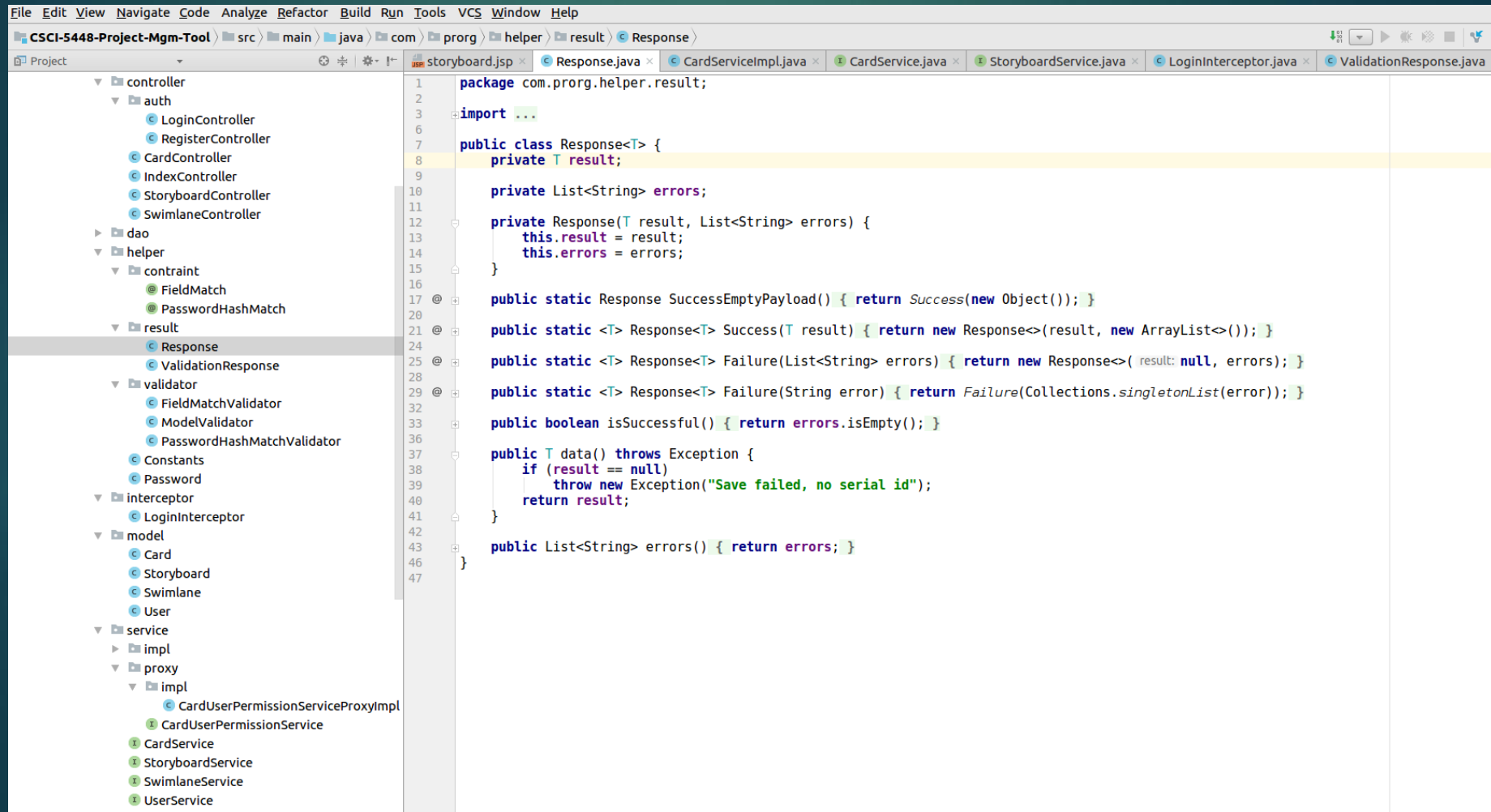
Design Patterns

1). Proxy Design Pattern



Design Patterns

2). Custom Design Pattern



The screenshot displays an IDE with a project named "CSCI-5448-Project-Mgm-Tool". The left sidebar shows a project tree with the following structure:

- controller
 - auth
 - LoginController
 - RegisterController
 - CardController
 - IndexController
 - StoryboardController
 - SwimlaneController
- dao
- helper
 - constraint
 - FieldMatch
 - PasswordHashMatch
 - result
 - Response
 - ValidationResponse
 - validator
 - FieldMatchValidator
 - ModelValidator
 - PasswordHashMatchValidator
 - Constants
 - Password
- interceptor
 - LoginInterceptor
- model
 - Card
 - Storyboard
 - Swimlane
 - User
- service
 - impl
 - CardUserPermissionServiceProxyImpl
 - proxy
 - impl
 - CardUserPermissionService
 - CardService
 - StoryboardService
 - SwimlaneService
 - UserService

The main editor window shows the code for `Response.java` in the package `com.prorg.helper.result`. The code is as follows:

```
1 package com.prorg.helper.result;
2
3 import ...
4
5 public class Response<T> {
6     private T result;
7
8     private List<String> errors;
9
10    private Response(T result, List<String> errors) {
11        this.result = result;
12        this.errors = errors;
13    }
14
15    @ public static Response SuccessEmptyPayload() { return Success(new Object()); }
16
17    @ public static <T> Response<T> Success(T result) { return new Response<>(result, new ArrayList<>()); }
18
19    @ public static <T> Response<T> Failure(List<String> errors) { return new Response<>( result: null, errors); }
20
21    @ public static <T> Response<T> Failure(String error) { return Failure(Collections.singletonList(error)); }
22
23    public boolean isSuccessful() { return errors.isEmpty(); }
24
25    public T data() throws Exception {
26        if (result == null)
27            throw new Exception("Save failed, no serial id");
28        return result;
29    }
30
31    public List<String> errors() { return errors; }
32 }
```


Technologies



- ▶ We have used the following technologies in our project.

- 1). Spring MVC
- 2). Hibernate as ORM.
- 3). Gradle build script to run application.
- 4). Jetty Server to run the project.

Creative Edge



- ❖ Migration Scripts
- ❖ Custom Hibernate Validations
- ❖ Namespacing String Constants
- ❖ Heroku Deployment

