SE III Requirements and Design Document

Name: Daniel Tilley
Student Number: C14337041
Application Idea: Web based forum

Table of Contents

Introduction

Topic Description

The idea for this assignment is to create a web based forum for cyclists. A user will be able to sign up, log in and upload posts or comment on existing posts by other users. The forum will allow cyclists to share things such as tips and tricks for fixing bikes, routes that are good to cycle, bikes parts / manufactures that cyclists would recommend etc.

This will demonstrate the methodology of Software Engineering by using UML and diagrams / models such as Use case diagrams, Class diagrams and Sequence diagrams.

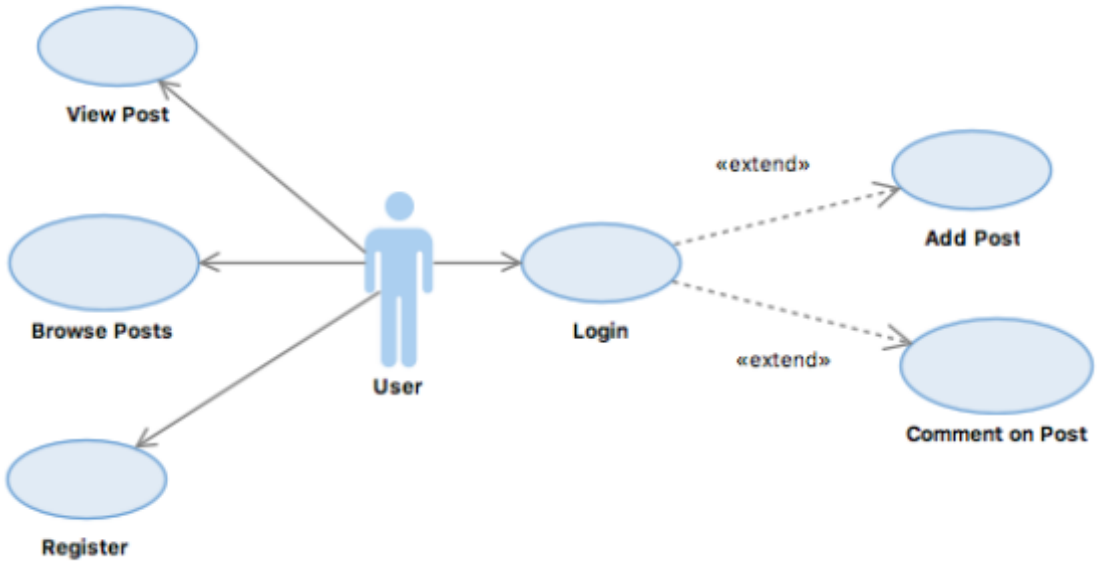
There will also be a link at the end of the document to an online video explaining the project as well as the models / diagrams included in this report.

From doing this project, I hope to improve my understanding of:

1. Java, Servlets and JSP
2. Software Engineering methodologies
3. UML and RSA
4. Use Cases, Class diagrams and Sequence diagrams

Requirements

Use Cases



Register

1. The user clicks on “Register” on the home page. They are then brought to the registration page.
2. The user enters in a username and password and confirms password a second time.
3. The user clicks “Submit”.
4. The system will then check if username is taken and if passwords match
    a. **Normal Flow:** User is brought to login page.
    b. **Alternative Flow:** Error message is displayed on screen and user asked to try again.

Login

1. The user clicks “Login” on home page, or when they try to view a page that requires them to be logged in, they are brought to the login page.
2. The user enters their username and password
3. The user clicks “Login”.
4. The system validates the username and password.
    a. **Normal Flow:** User is brought to Browse Posts screen.
    b. **Alternative Flow:** Error message is displayed and user is asked to try again.

Browse Posts

1. The user clicks “Browse Posts” or has arrived on the page after logging in.
2. The system gets a list of posts from the database and displays them on screen. The user can also search for posts using a search bar at the top of the screen.

3. The system displays the list on screen

View Post

1. The user clicks on a post they want to see from the browse post screen.
2. The system re-directs the user to a new page in which it will display info about the post
3. The system gets all info associated with the post from the database.
4. The system displays all info about the post on screen.

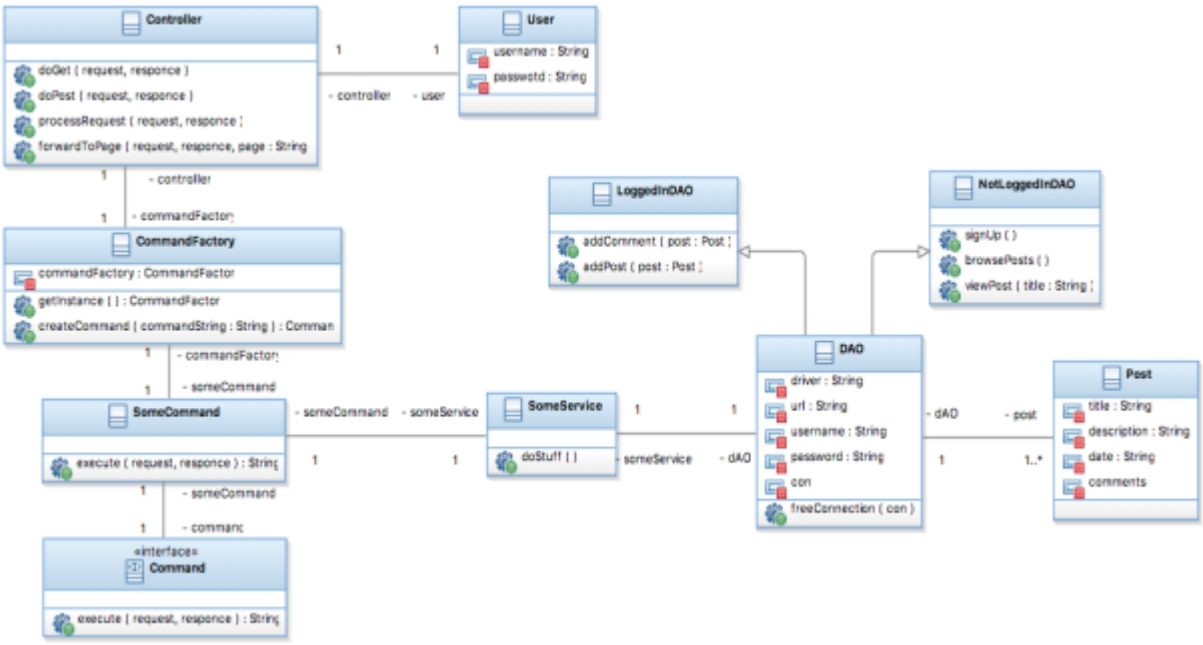
Comment on Post

1. Assumption: The user must be logged in.
2. The user clicks “Add comment” below a post they wish to comment on.
3. The user types in the comment they want to make in a text box and the system captures the username and date.
4. The user clicks “Submit”.
5. The system validates comment.
    a. **Normal Flow:** The comment is added to the post and the page is refreshed
    b. **Alternative Flow:** An error message is displayed and the comment is not added to the post.

Add Post

1. Assumption: The user must be logged in.
2. The user clicks “Add a post” at the top of browse posts screen.
3. The user enters a title and a description. The system captures the user name and date also.
4. User clicks “Submit”.
5. The system validates the post.
    a. **Normal Flow:** The post is added and the user is brought back to the brose posts screen.
    b. **Alternative Flow:** An error is displayed and the user is given a chance to update the post.

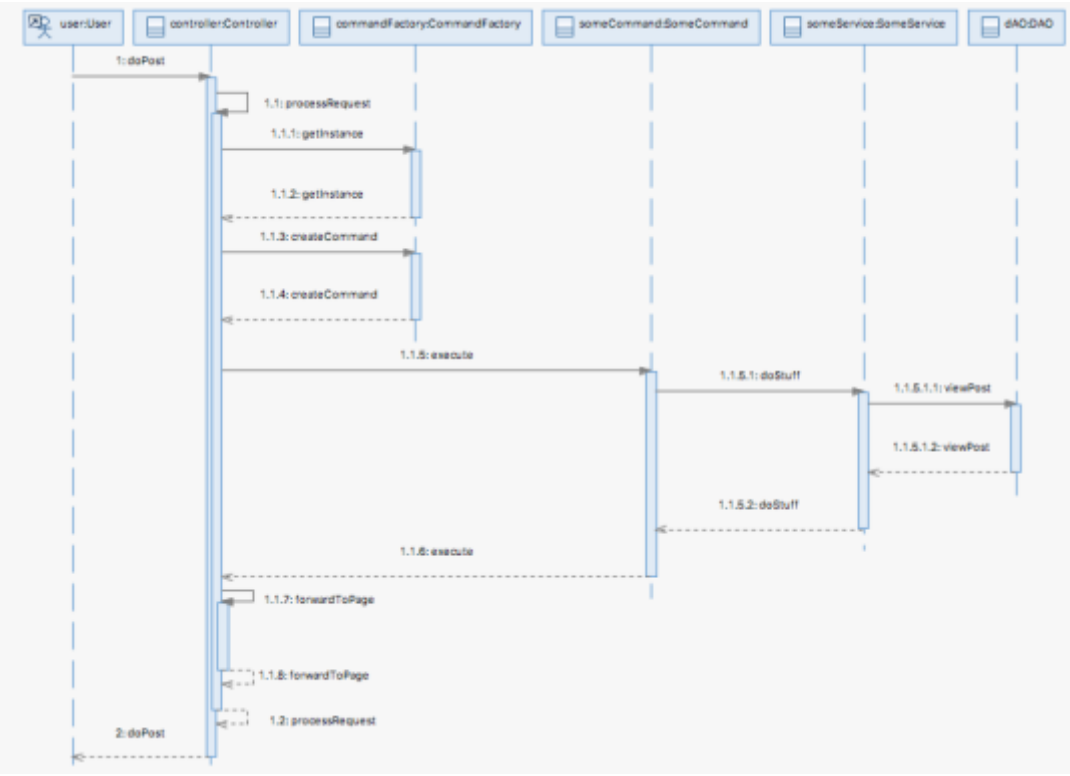
Class Diagram



The relationships in this class diagram are bi-directional. The front controller acts as the functions of the web application and has a 1 to 1 relationship with the user as well as the Command Factory class. This class will be in charge of storing its own instance (singleton) and also creating a command object. Some command class will be used to create an object and method specific code will be executed by the controller once the object is created. Within the execute method, there will be a call to a service class, which will in turn call the DAO and retrieve the data from the database. The post class will be used to store data received and sent from / to the database.

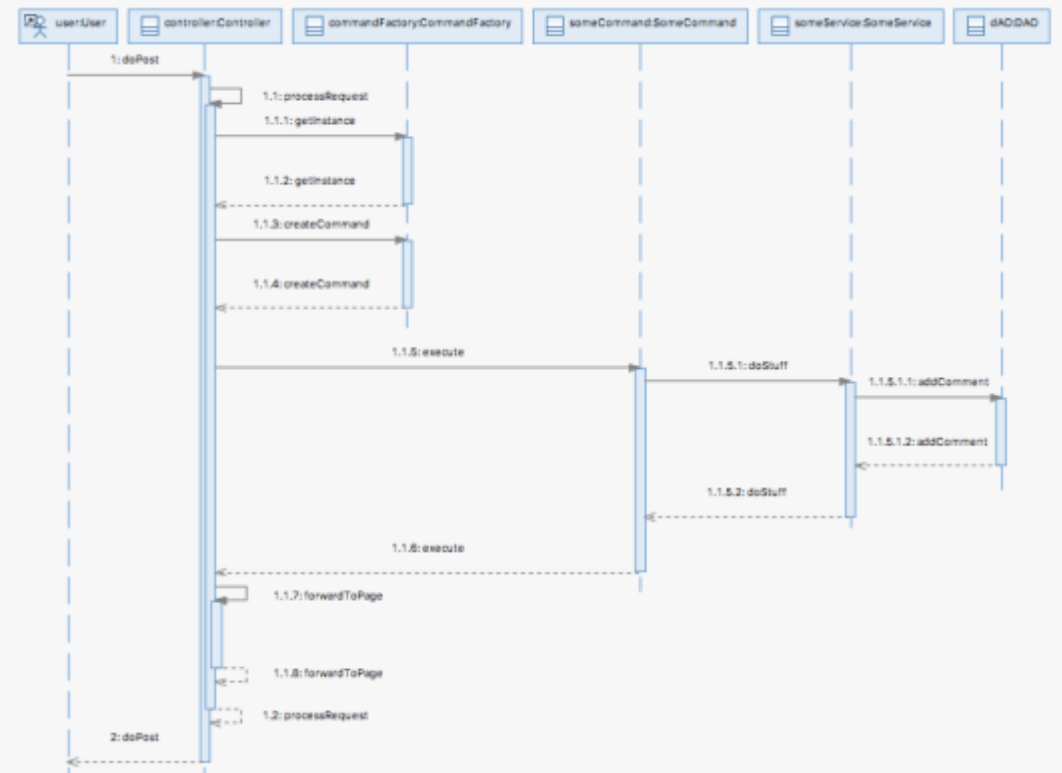
Sequence Diagrams

View Post



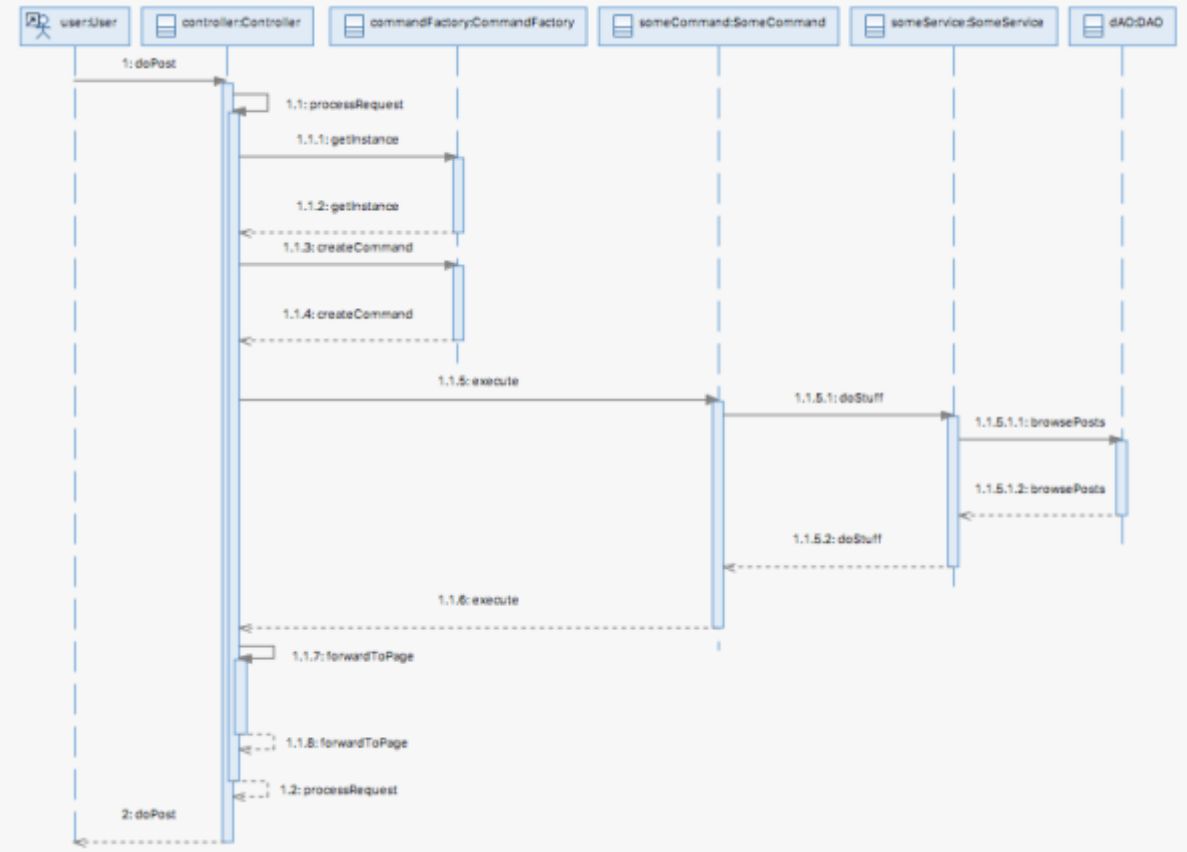
User clicks on a post they want to look at which calls a “doPost” method. The controller processes the request. The controller gets the command factory instance, creates a command object, it is returned to the controller and is then executed. The method “doStuff” is performed and the DAO is accessed to return the data.

Add Comment



User clicks on a post they want to look at which calls a “doPost” method. The controller processes the request. The controller gets the command factory instance, creates a command object, it is returned to the controller and is then executed. The method “doStuff” is performed and the DAO is accessed to return the data.

Browse Posts



User clicks on a post they want to look at which calls a “doPost” method. The controller processes the request. The controller gets the command factory instance, creates a command object, it is returned to the controller and is then executed. The method “doStuff” is performed and the DAO is accessed to return the data.

Recorded Presentation

Here is a link to my recorded presentation:

http://screencast-o-matic.com/watch/cDXiYmjN00