

Software Design Description for Mashbot

George D'Andrea Andrew Gall Josiah Kiehl Cody Ray Vito Salerno

January 31, 2010

Revision History

Name	Date	Reason for Changes	Version
George D'Andrea, Andrew Gall, Josiah Kiehl, Cody Ray, Vito Salerno	17 January 2010	Initial Version	1.0

Contents

1	Introduction	4
1.1	Purpose	4
1.2	Scope	4
1.3	Definitions, Acronyms, and Abbreviations	4
1.4	Context Diagram	4
2	Architecture	5
2.1	Overview	5
2.2	Four-Tier Architecture	5
2.3	Service-Oriented Architecture	5
2.4	Survey of Technologies Used	5
2.4.1	Campaign Manager	5
2.4.2	Publishing and Aggregation Platform	6
2.5	Presentation Layer Components	6
2.5.1	Campaign Views	6
2.5.2	Content Views	9
2.5.3	Scheduling Views	10
2.5.4	Explore View	11
2.6	Business Layer Components	14
2.7	Session and Authentication	14
2.7.1	Log In	14
2.7.2	Log Out	14
2.7.3	OAuth	14
2.7.4	Openid	14
2.7.5	Session Handling	14
2.8	Data Layer Components	14
2.9	External Components	14
2.9.1	Publishing and Aggregation Targets	14
2.9.2	Email/SMTP Service	15
3	Design Features	16
3.1	External Authentication via Openid	16
3.2	Create User Account	16
3.3	Update User Account	16
3.4	Delete User Account	16
3.5	Create Campaign	19
3.6	View Campaign	19
3.7	Update Campaign	19
3.8	Delete Campaign	19
3.9	Schedule Campaign	19

3.10	Create Content	22
3.11	View Content_Unit	22
3.12	Update Content_Unit	22
3.13	Delete Content_Unit	22
3.14	Schedule Content_Unit	22
3.15	View Metrics and Statistics via Explore Panel	26
3.16	Lost User Name	26
3.17	Lost Password	26
4	Publishing and Aggregation Platform	29
4.1	API Design	29
4.1.1	Request	29
4.1.2	Content Types	30
4.1.3	Operation	30
4.2	Server Design	30
4.3	Plugin Design	30
4.3.1	Content-Type Plugins	30
4.3.2	Service Plugins	30
5	Database Design	31
5.1	Summary	31
5.1.1	Advantages of Design	32
5.1.2	Disadvantages of Design	32
5.1.3	Design Rationale	32
5.2	A Requirements Traceability Matrix	32
5.2.1	Traceability by Requirement Numbers	32
5.2.2	Traceability by Design Component	32

Chapter 1

Introduction

1.1 Purpose

This document serves to expand upon the requirement into implementation details and technology choices. This document should be referenced when specific features are being implemented.

1.2 Scope

This covers the general architecture of Mashbot, as well as the design decisions used to apply that architecture via various appropriate technologies, libraries and frameworks.

1.3 Definitions, Acronyms, and Abbreviations

- MVC — Model View Controller
- SASS — Syntactically Awesome Style Sheets
- HAML — XHTML Abstraction Markup Language
- XHTML — Extensible Hypertext Markup Language
- CSS — Cascading Style Sheets
- API — Application Programmimg Interface

1.4 Context Diagram

Chapter 2

Architecture

2.1 Overview

Generally, Mashbot will be implemented using a strict Model-View-Controller architecture. This is augmented by the inclusion of the Presentation and Aggregation Platform, the purpose of which is to abstract the interaction with external service APIs from the application as a whole, thus allowing a pure MVC architecture to be implemented, increasing maintainability, extensibility and accessibility.

2.2 Four-Tier Architecture

- Data Layer / Model
- Presentation Layer / View
- Business Layer / Controller
- Publishing and Aggregation Platform

2.3 Service-Oriented Architecture

Mashbot will be implemented as two distinct yet related services. The Campaign Manager will handle the interaction between the user and the data the Campaign Manager is concerned with, where the Presentation and Aggregation Platform will handle the interaction between external service APIs and the Campaign Manager.

2.4 Survey of Technologies Used

2.4.1 Campaign Manager

- Presentation Layer
 - HAML — HTML replacement markup language, for building web layout structure.
 - SASS — CSS replacement stylesheets, for applying visual styles to the layout built in HAML.
 - jQuery — JavaScript library which provides cross-browser compatibility as well as streamlined Ajax request handling.
 - Google Chart API — Public service provided by Google which generates many different kinds of charts and graphs.

- Business Layer
 - Ruby — Dynamic programming language.
 - Rails — Web application framework written in Ruby which provides a concise Model-View-Controller architecture.
 - Heroku — Rails engine which provides enhanced production deployment via Rails compilation, a fast readonly filesystem, and horizontal scaling.
- Data Layer
 - ActiveRecord — Component of Rails which provides the Active Record pattern of data access, creating data model objects and relationships for interacting with resources in a database.
 - MySQL — Fast and free relational database which plugs into Rails without effort.

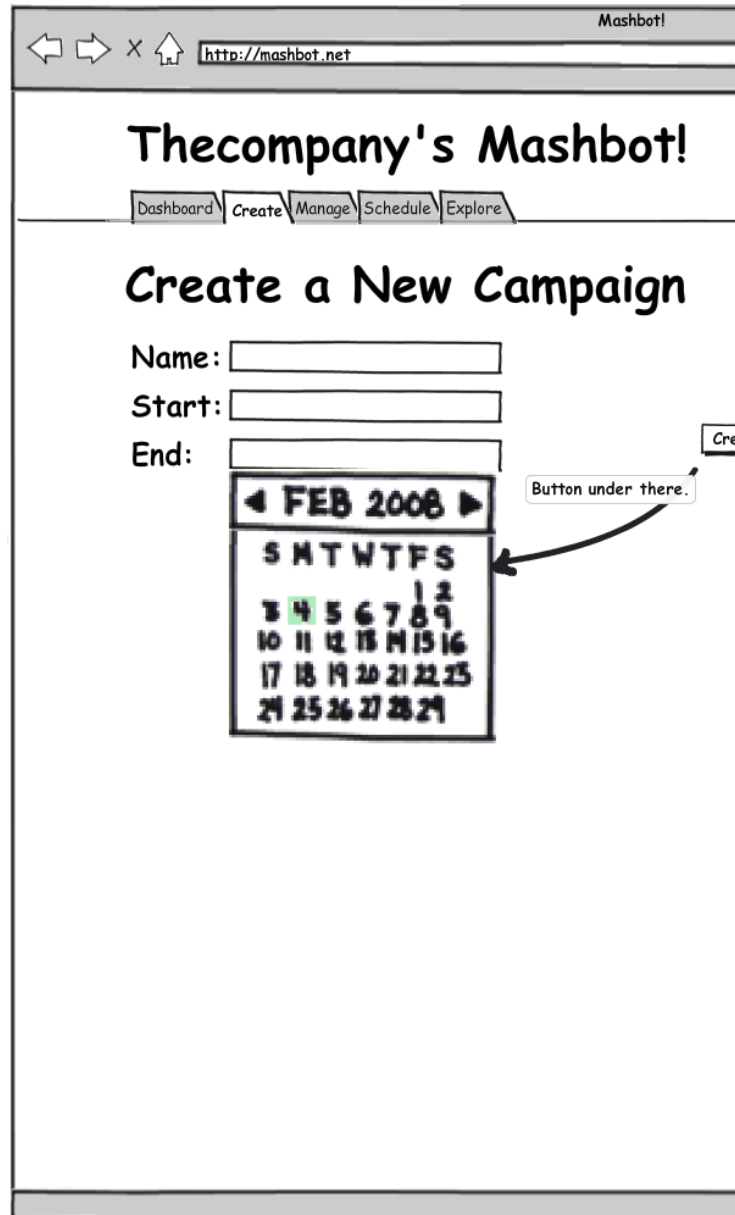
2.4.2 Publishing and Aggregation Platform

- TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TODO:TO
Put tech stuff here.

2.5 Presentation Layer Components

2.5.1 Campaign Views

Campaigns are accessed via the Create and Manage tabs on the primary navigation tabs. Create is for the Create view, Manage is for List, Show and Edit.



- Create — This is where users can create new campaigns.
- List — This is where users can view, update or delete existing campaigns.

Mashbot!
http://mashbot.net

Josiah Kiehl
[Settings](#)
[Sign Out](#)

Thecompany's Mashbot!

Dashboard
Create
Manage
Schedule
Explore

☐
Active

<input type="checkbox"/>	Title	Start Date	End Date	Scheduled By		
<input type="checkbox"/>	Twitteriffic Campaign	00-00-0000	00-00-0000	Josiah Kiehl	edit	delete
<input type="checkbox"/>	Campaign of Doom	00-00-0000	00-00-0000	Andy G.	edit	delete

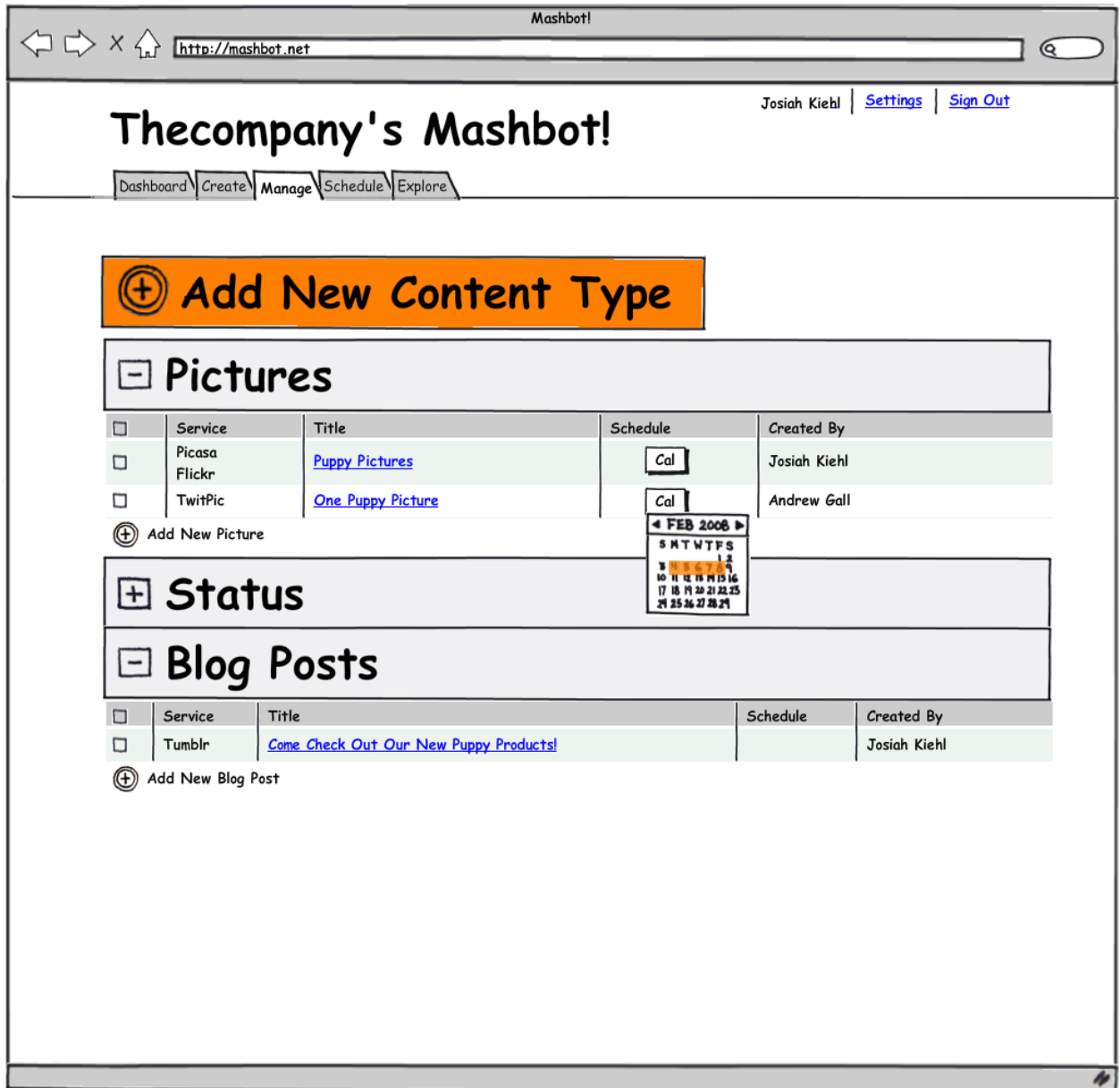
☐
Future

<input type="checkbox"/>	Title	Start Date	End Date	Scheduled By		
<input type="checkbox"/>	Twitteriffic Campaign	00-00-0000	00-00-0000	Josiah Kiehl	edit	delete
<input type="checkbox"/>	Campaign of Doom	00-00-0000	00-00-0000	Andy G.	edit	delete

☐
Past

<input type="checkbox"/>	Title	Start Date	End Date	Scheduled By		
<input type="checkbox"/>	Twitteriffic Campaign	00-00-0000	00-00-0000	Josiah Kiehl	clone	delete
<input type="checkbox"/>	Campaign of Doom	00-00-0000	00-00-0000	Andy G.	clone	delete

- Show — This view is what is shown when the user wants to view an existing campaign via the Show view. This is also where the Content pieces will be listed.



- Edit — This is virtually the same view as Create, however this will be prepopulated with the existing content of the given Campaign.

2.5.2 Content Views

Content pieces are included inside Campaigns. These views are accessible via the Show view of a Campaign for the corresponding Campaign id.

- Create — When on the Show view of a given campaign, the user can enter the Create view for Content.
- Show — This is how the user previews the Content they have created.
- Edit — This is virtually the same view as Create, however this will be prepopulated with the existing content of the given Content.

2.5.3 Scheduling Views

- Primary Scheduling View — consists of a list of Campaigns available to be scheduled (ie: they do not have existing start/stop dates) as well as already scheduled Campaigns placed properly on the calendar.

The screenshot shows the 'Thecompany's Mashbot!' interface. At the top, there's a browser window with the URL 'http://mashbot.net'. Below the browser, the page title is 'Thecompany's Mashbot!'. A navigation bar contains links: 'Dashboard', 'Create', 'Manage', 'Schedule', and 'Explore'. The 'Schedule' link is highlighted. On the left, there's a 'Campaigns' sidebar with a link 'Campaign of Awesome Puppies Galore!'. An arrow labeled 'Drag and Drop' points from this link to a calendar. The calendar is a grid showing days from Sunday to Saturday. A campaign 'Campaign of Awesome' is scheduled from Monday, January 12th to Friday, January 16th. A yellow callout box states: 'Campaigns can be dragged around the page, each drop setting the new date in the database.' At the bottom, there are two small calendar widgets for January 2007 and March 2007, and a 'Notes' section.

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
				1	2 Groundhog Day	3
4	5	6	7	8	9	10
	Campaign of Awesome					
11	12 Lincoln's B-Day	13	14 Valentines Day	15	16	17
18	19 President's Day	20			23	24
25	26	27	28			

January 2007

S	M	T	W	Th	F	Sa
1	2	3	4	5	6	
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

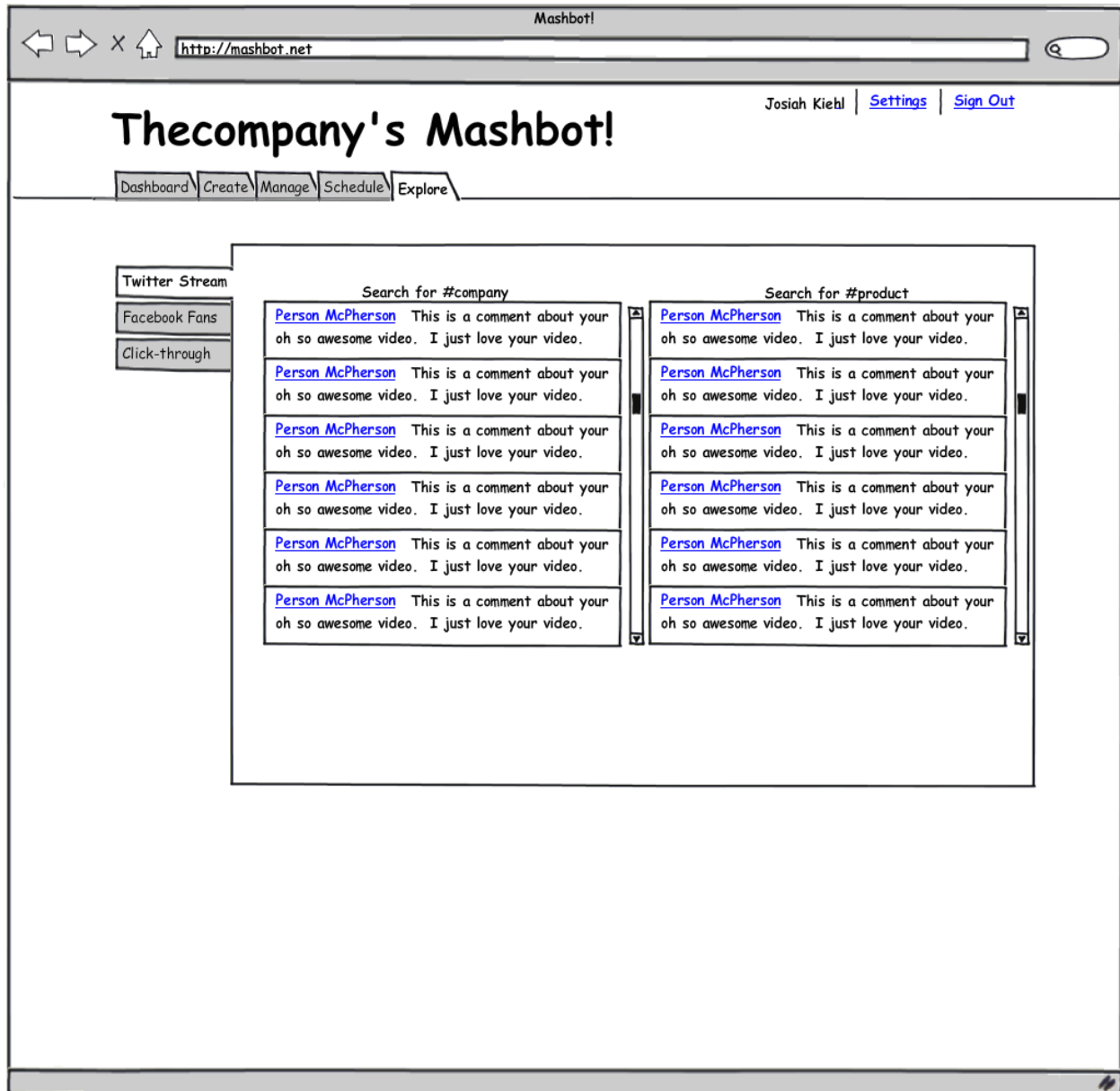
March 2007

S	M	T	W	Th	F	Sa
					1	2
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

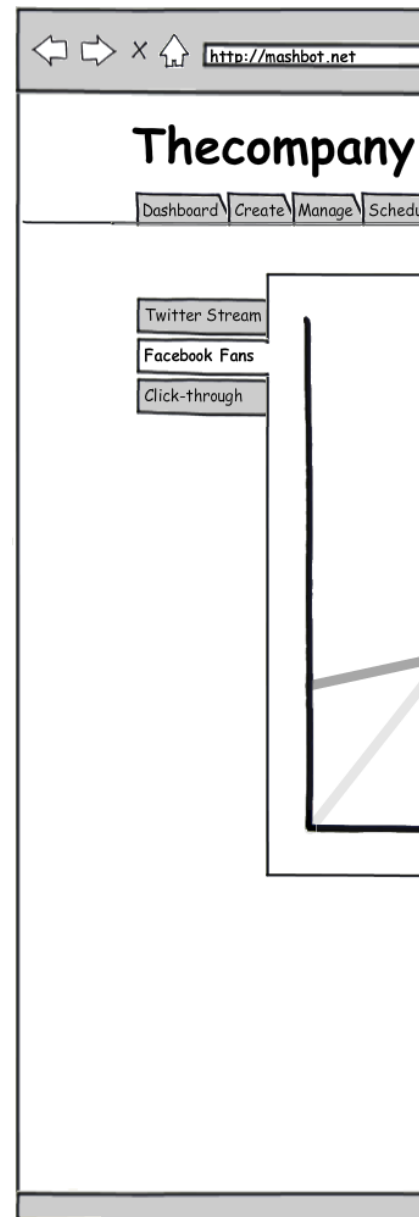
Notes:

www.vertex42.com © 2007 Vertex42 LLC

- Content Scheduling View — similar to the Primary Scheduling View, however the items available to be scheduled here are the individual content pieces of the Campaign. This is accessed via selecting the Campaign from the calendar, or via the List Campaign or Show Campaign views.



- Plugin Independent
 - Clickthrough tracking — Any time a link is generated via Mashbot, it is given a special redirecting URL that will allow Mashbot to track how many times the link has been clicked.
 - Rate of publishing — How often does the user tweet/blog/etc. This will most likely be used to correlate frequency with user engagement.
- Plugin Dependent



- Facebook Fan tracking — A line chart of how many fans the user's fan page has.
- Twitter Follower tracking — A line chart of the number of twitter followers the user's twitter account has.
- Number of times retweeted — A line chart of the number of times a tweet of the user's has been retweeted.

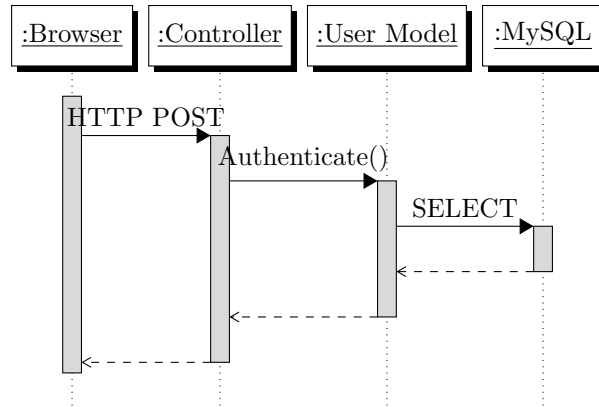


Figure 2.1: Log in to authenticate the user.

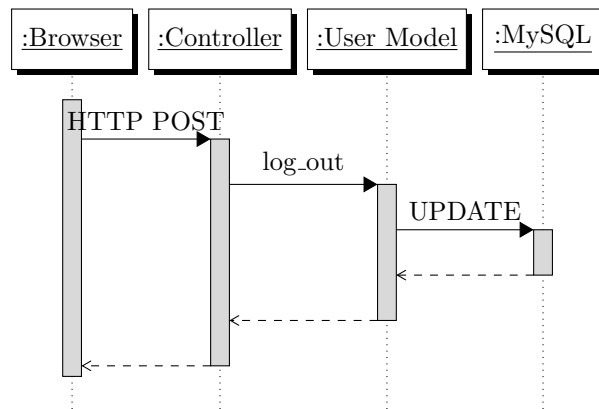


Figure 2.2: End authenticated session by logging out.

2.6 Business Layer Components

2.7 Session and Authentication

2.7.1 Log In

2.7.2 Log Out

2.7.3 OAuth

2.7.4 Openid

2.7.5 Session Handling

2.8 Data Layer Components

2.9 External Components

2.9.1 Publishing and Aggregation Targets

- Twitter

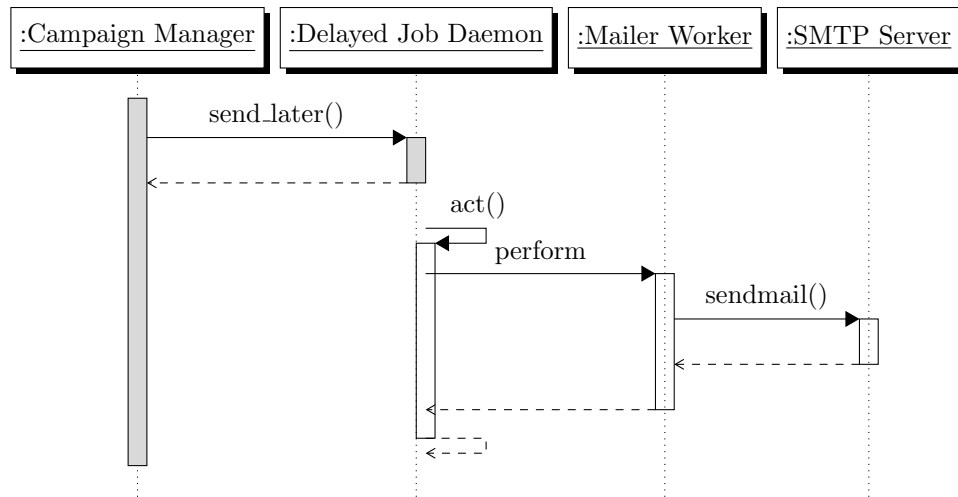


Figure 2.3: Send email (Requirements 3.1.1)

- Tumblr
- Wordpress
- TODO: etc.

2.9.2 Email/SMTP Service

Chapter 3

Design Features

3.1 External Authentication via Openid

3.2 Create User Account

This is a basic CRUD operation: Create User Account.

3.3 Update User Account

This is a basic CRUD operation: Update User Account.

3.4 Delete User Account

This is a basic CRUD operation: Delete User Account.

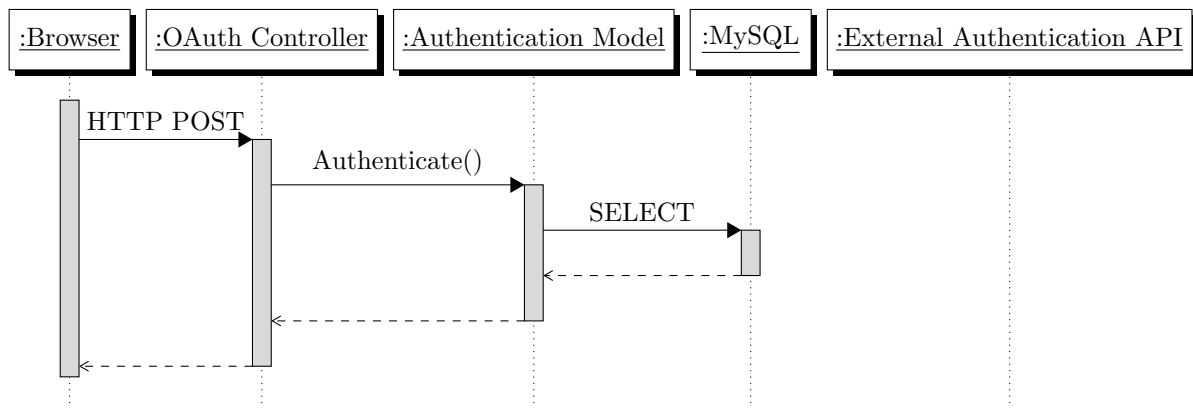


Figure 3.1: Authentication via OAuth with stored token.

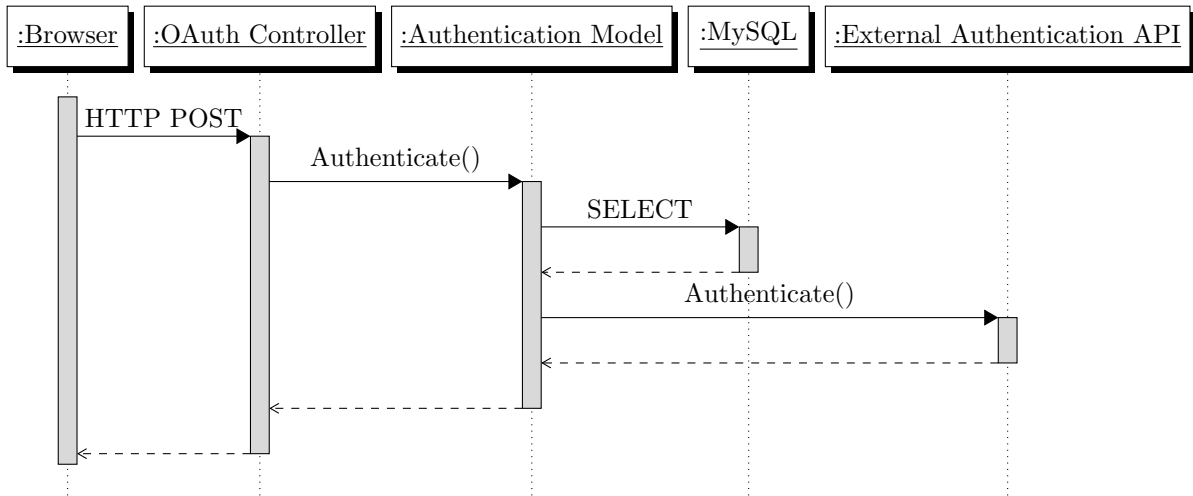


Figure 3.2: Authentication via OAuth without stored token.

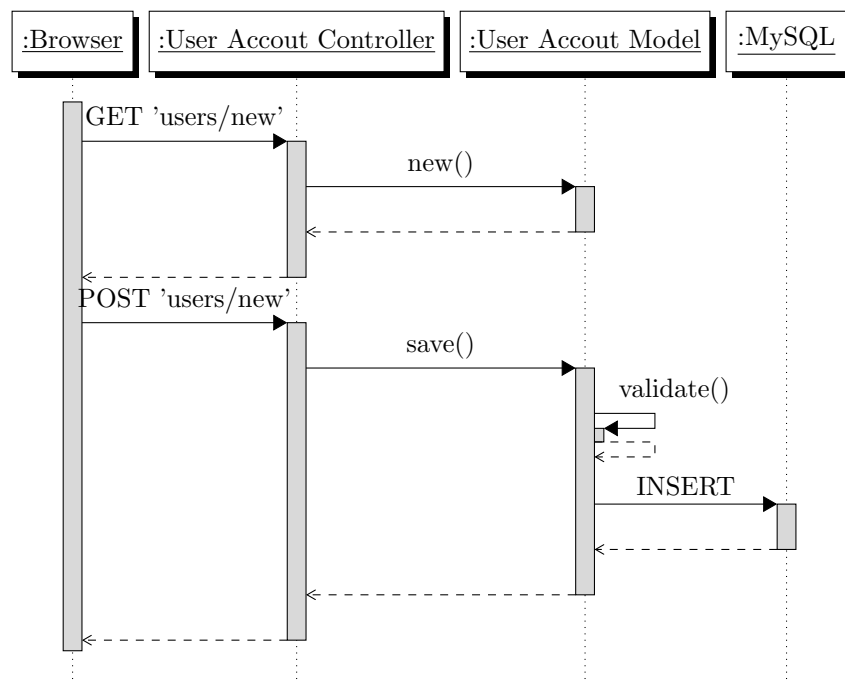


Figure 3.3: Create User Account

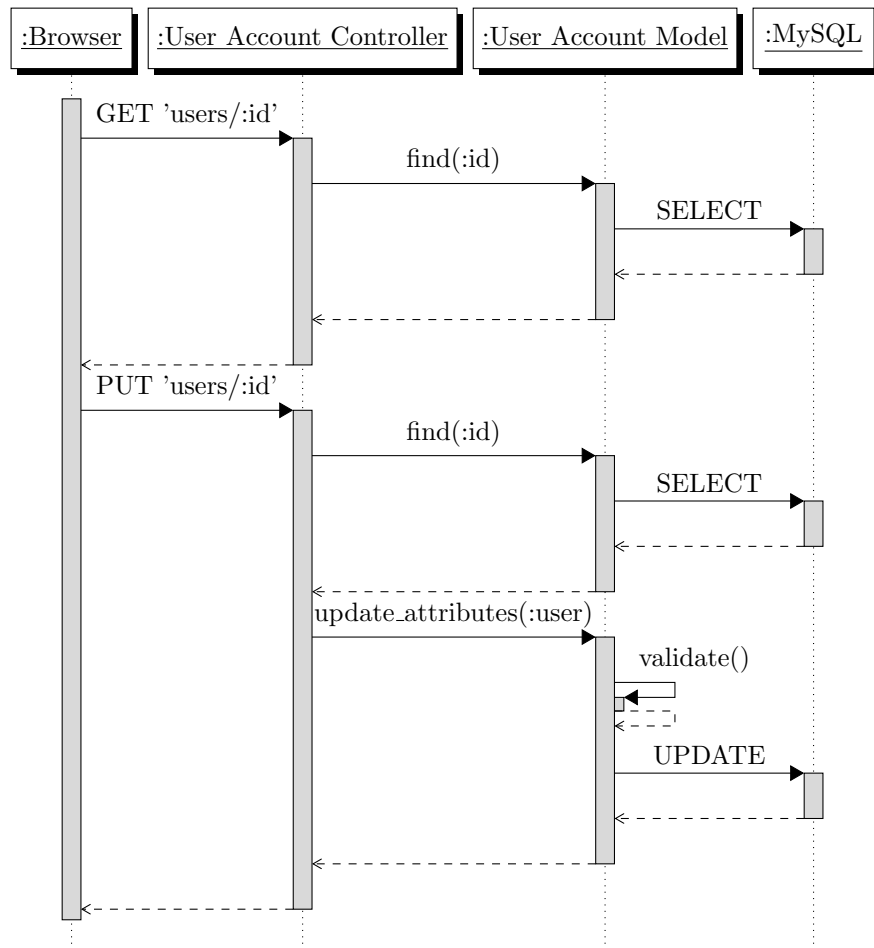


Figure 3.4: Update a User Account

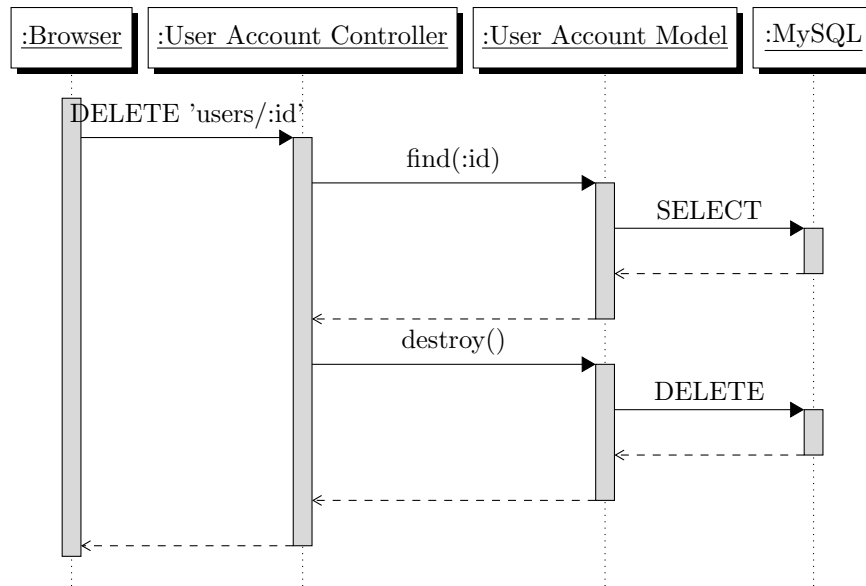


Figure 3.5: Delete User Account

3.5 Create Campaign

This is a basic CRUD operation: Create Campaign.

3.6 View Campaign

This is a basic CRUD operation: View Campaign.

This is a Read operation, similar to the above, but for one Campaign.

3.7 Update Campaign

This is a basic CRUD operation: Update Campaign.

3.8 Delete Campaign

This is a basic CRUD operation: Delete Campaign.

3.9 Schedule Campaign

This is a basic CRUD operation: Schedule Campaign.

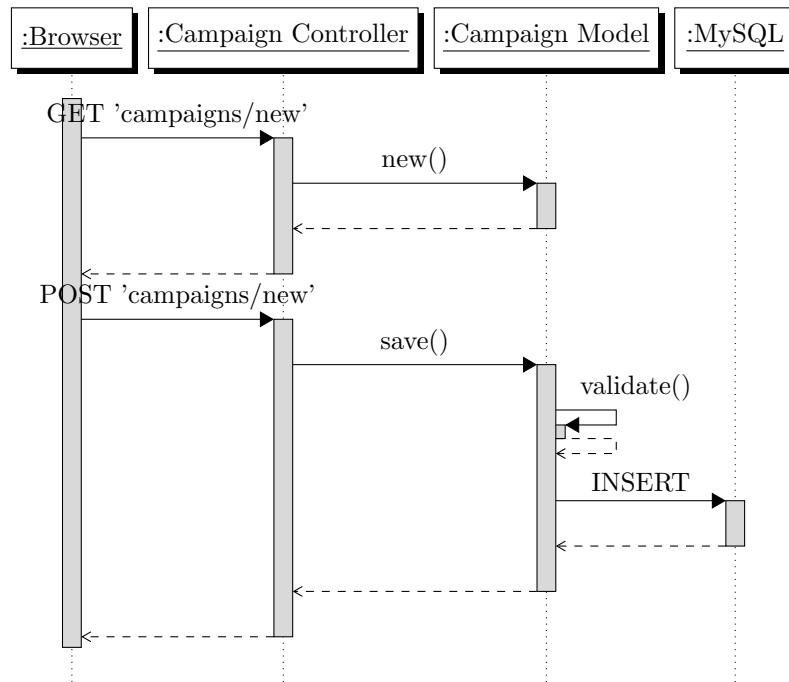


Figure 3.6: Create Campaign

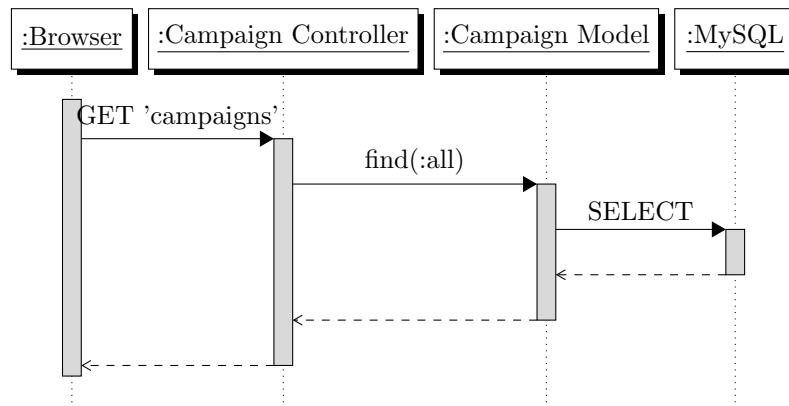


Figure 3.7: List Campaigns

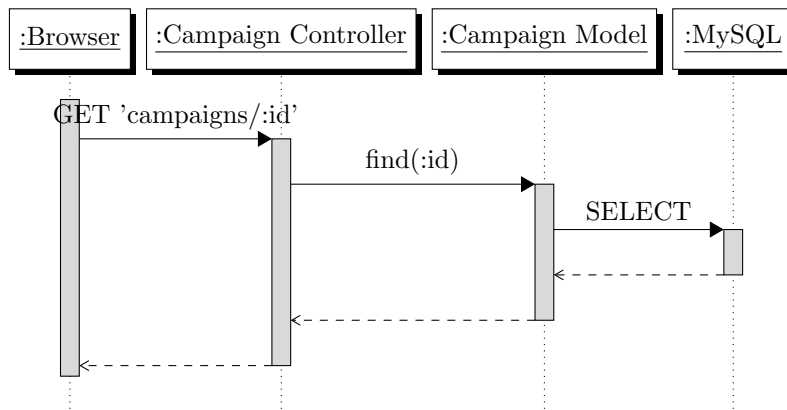


Figure 3.8: Show a Campaign

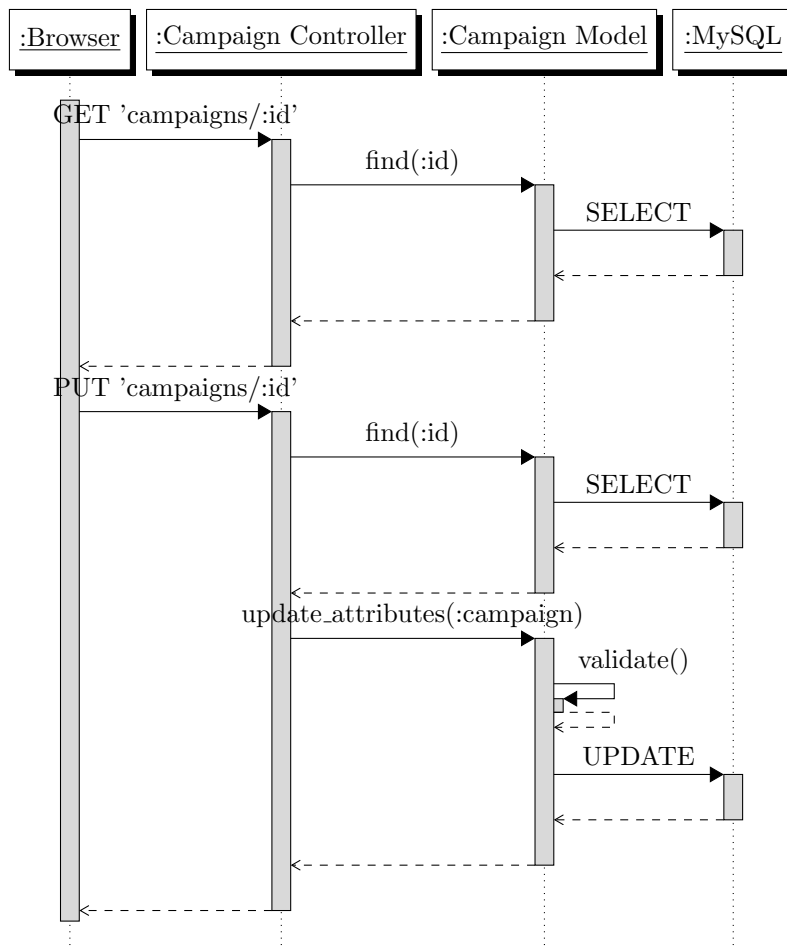


Figure 3.9: Update a Campaign

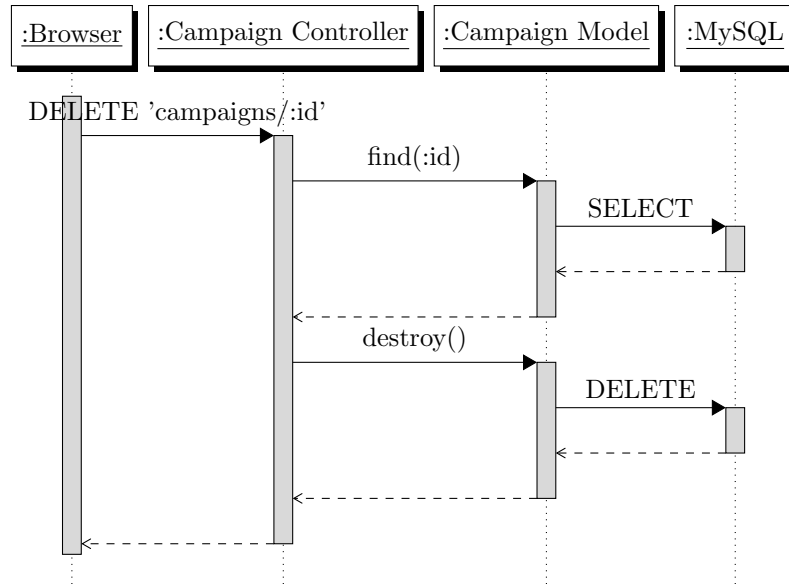


Figure 3.10: Delete Campaign

3.10 Create Content

This is a basic CRUD operation: Create Content.

3.11 View Content_Unit

This is a basic CRUD operation: View Content_Unit.

This is a Read operation, similar to the above, but for one Content Unit.

3.12 Update Content_Unit

This is a basic CRUD operation: Update Content_Unit.

3.13 Delete Content_Unit

This is a basic CRUD operation: Delete Content_Unit.

3.14 Schedule Content_Unit

In order to change the scheduled time of anything that is schedulable, the same process as an update is carried out.

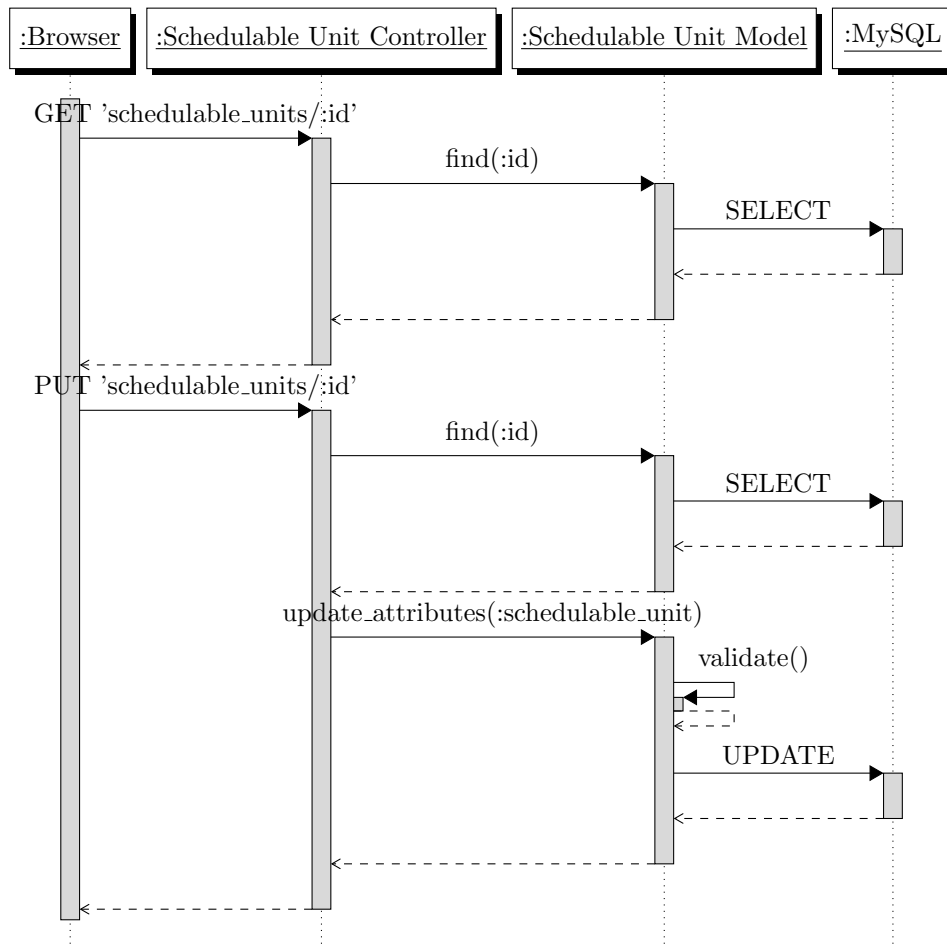


Figure 3.11: Update scheduled time of Schedulable Unit

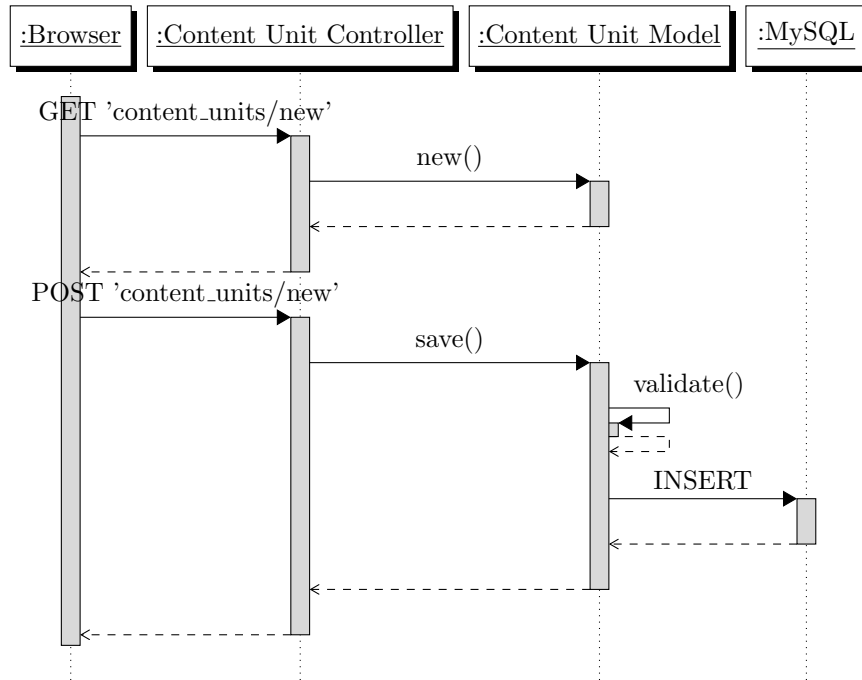


Figure 3.12: Create Content Unit

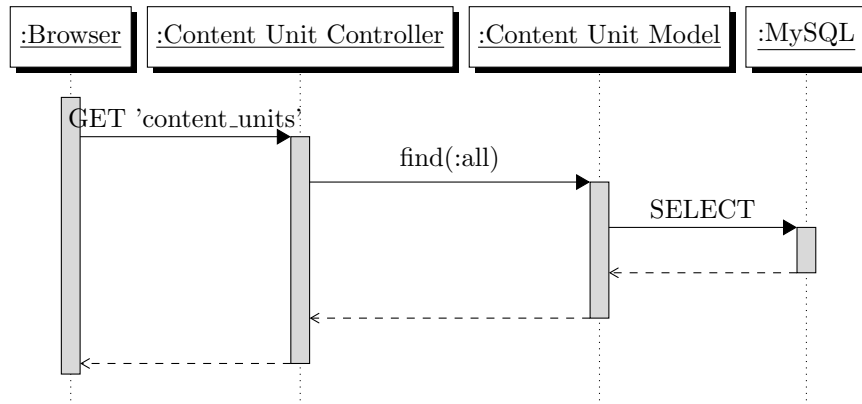


Figure 3.13: List Content Units

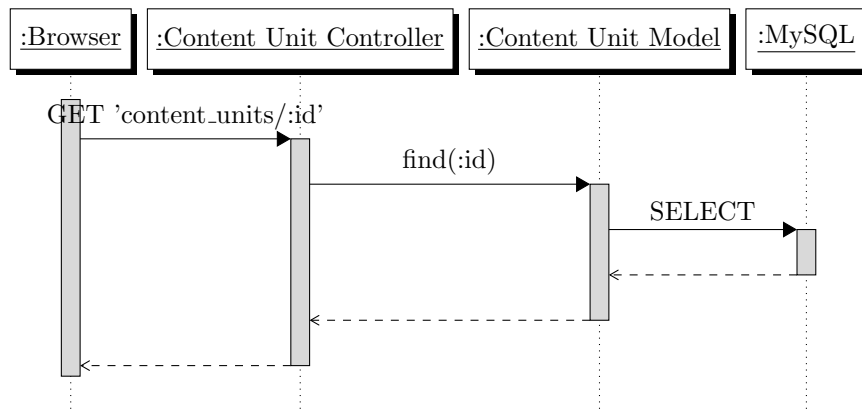


Figure 3.14: Show a Content Unit

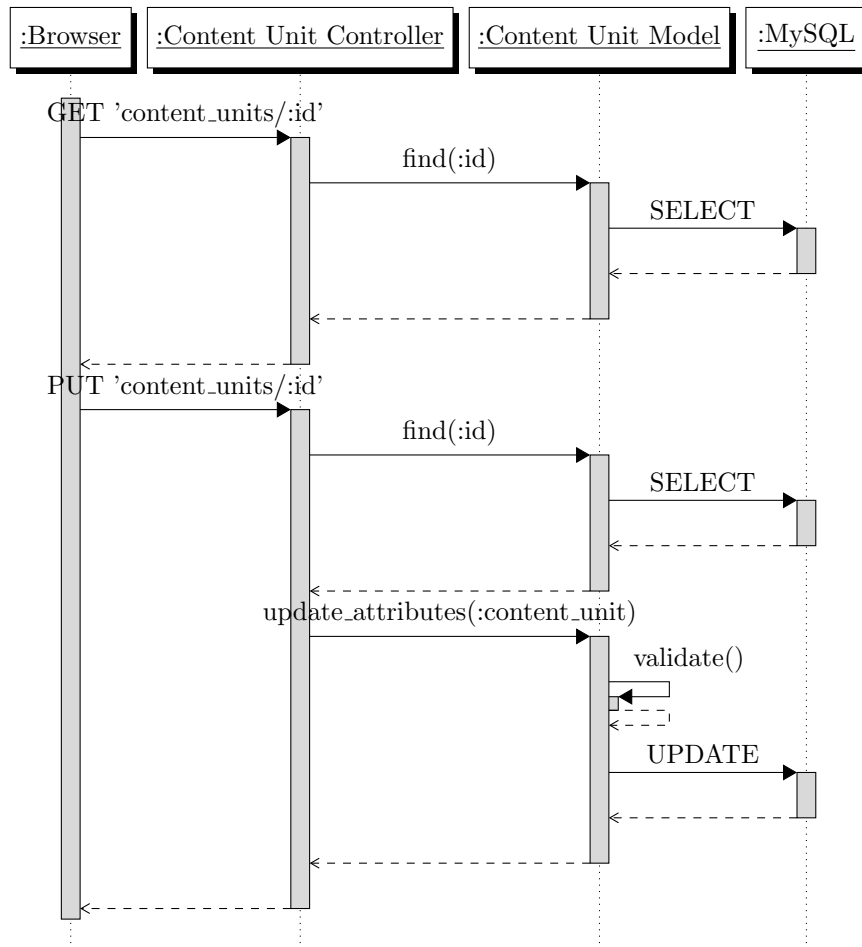


Figure 3.15: Update content_unit

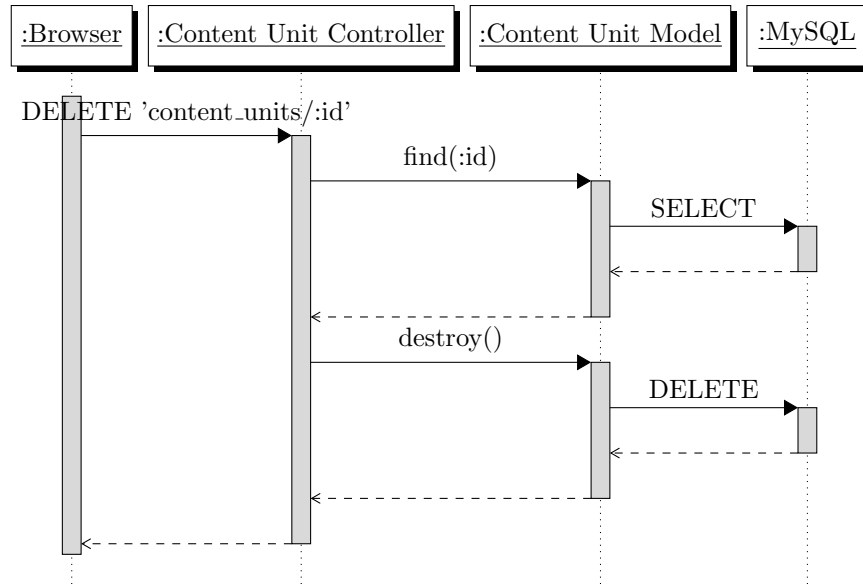


Figure 3.16: Delete Content Unit

After content is scheduled, it can be picked up by the Scheduler, triggered via Cron. Every 30 minutes, Cron will run the Scheduler, which will look in the database for active Campaigns. Within those active campaigns, each piece of content will be checked to see if the go-live time is now or past. If it's now or past, the scheduler calls the Publishing and Aggregation Platform with the content needed for a push to the given External Service.

3.15 View Metrics and Statistics via Explore Panel

3.16 Lost User Name

3.17 Lost Password

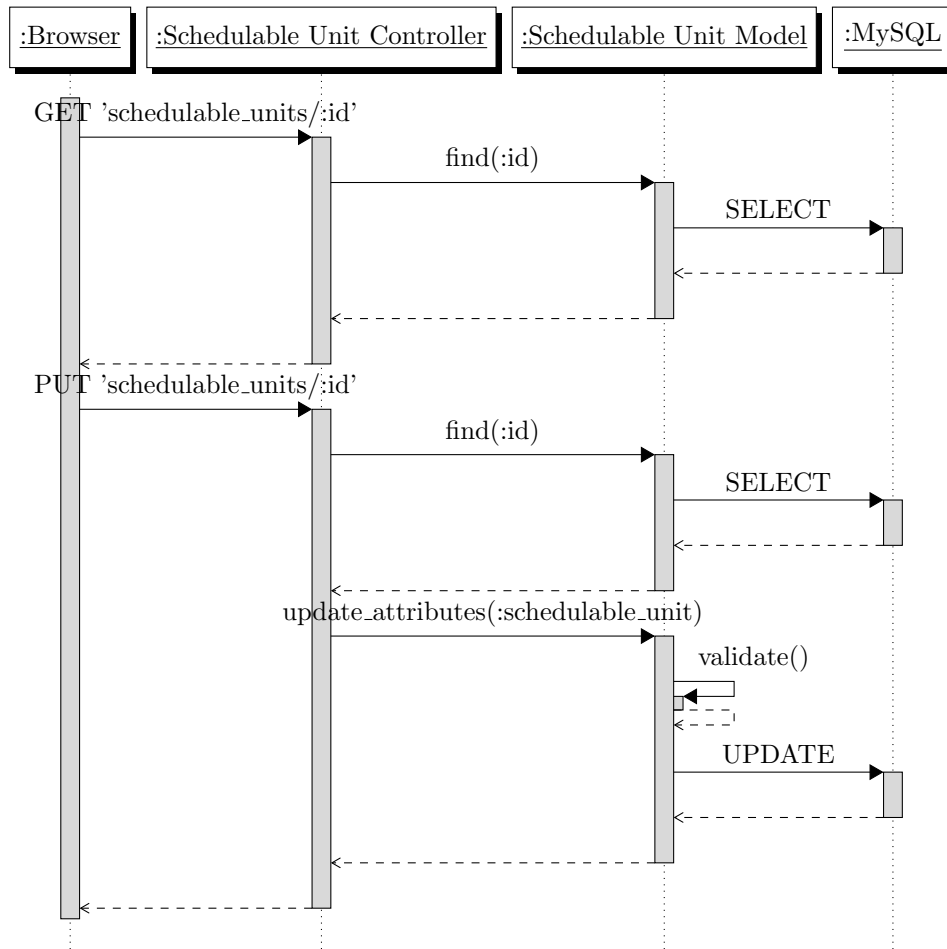


Figure 3.17: Update scheduled time of a Schedulable Unit

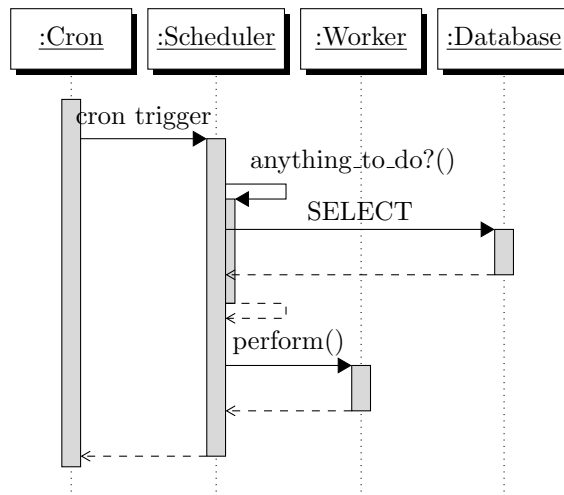


Figure 3.18: Perform a scheduled action

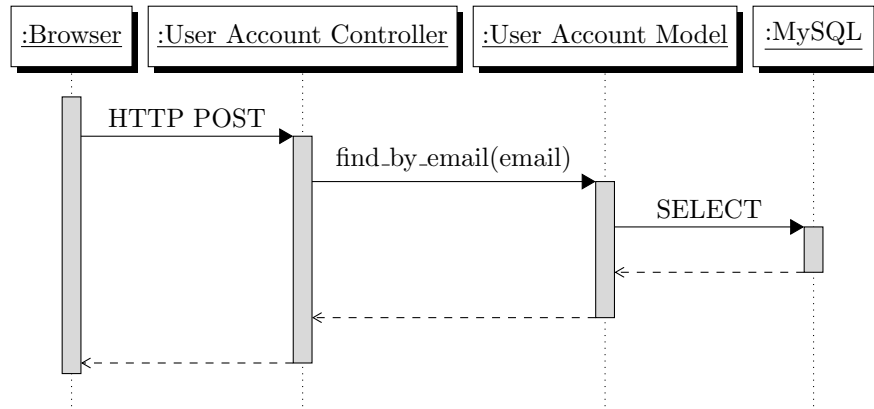


Figure 3.19: Retrieving a lost username by email id.

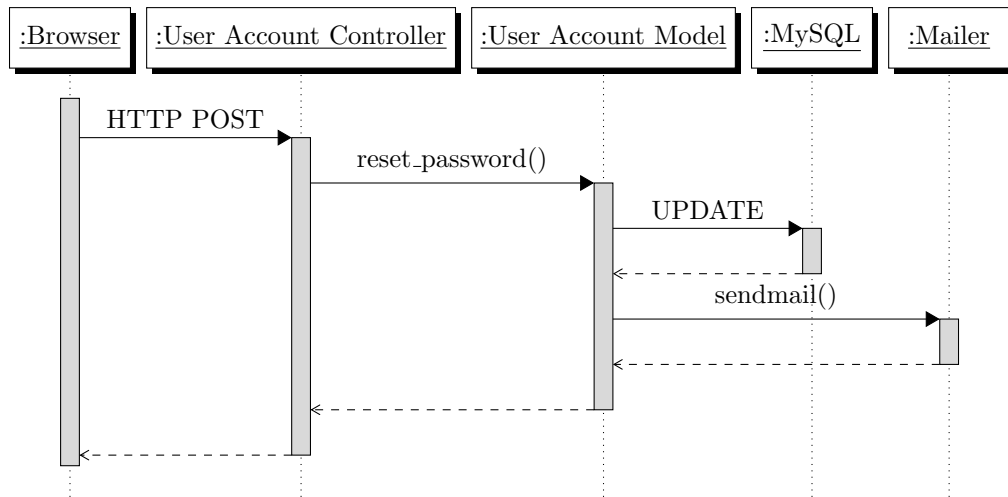


Figure 3.20: Retrieving a lost password. Refer to email figure for mailer actions.

Chapter 4

Publishing and Aggregation Platform

4.1 API Design

4.1.1 Request

A request to our API will have the following fields:

- OAuth Information
 - Realm - A Protection Realm
 - Consumer Key - A key for identifying the customer
 - Access Token - A token generated using the parameters and the Consumer Secret
 - Nonce - A randomly generated string given to all requests sent with the same timestamp.
 - Timestamp - The number of seconds since January 1st, 1970.
 - Signature Method - The signature method that the user used to sign the request
 - Version - Version of OAuth that you are using.
- Content Type - A string containing the content type as detailed below.
- Operation - A string containing the operation as detailed below.
- Content - The content to be passed to the appropriate Content Processors.
- Metadata - A map of key-value pairs used to hold metadata for the content.
- Third Party Authentication Data - The third party authentication data for sending third party requests. It will contain a tree whose hierarchy will contain the following levels.
 1. Service Name
 2. User Name
 3. Authentication - Contains either the authentication token for this user or a password

4.1.2 Content Types

Blog

Picture

Status

Video

4.1.3 Operation

Delete

Edit

Pull

Push

4.2 Server Design

4.3 Plugin Design

4.3.1 Content-Type Plugins

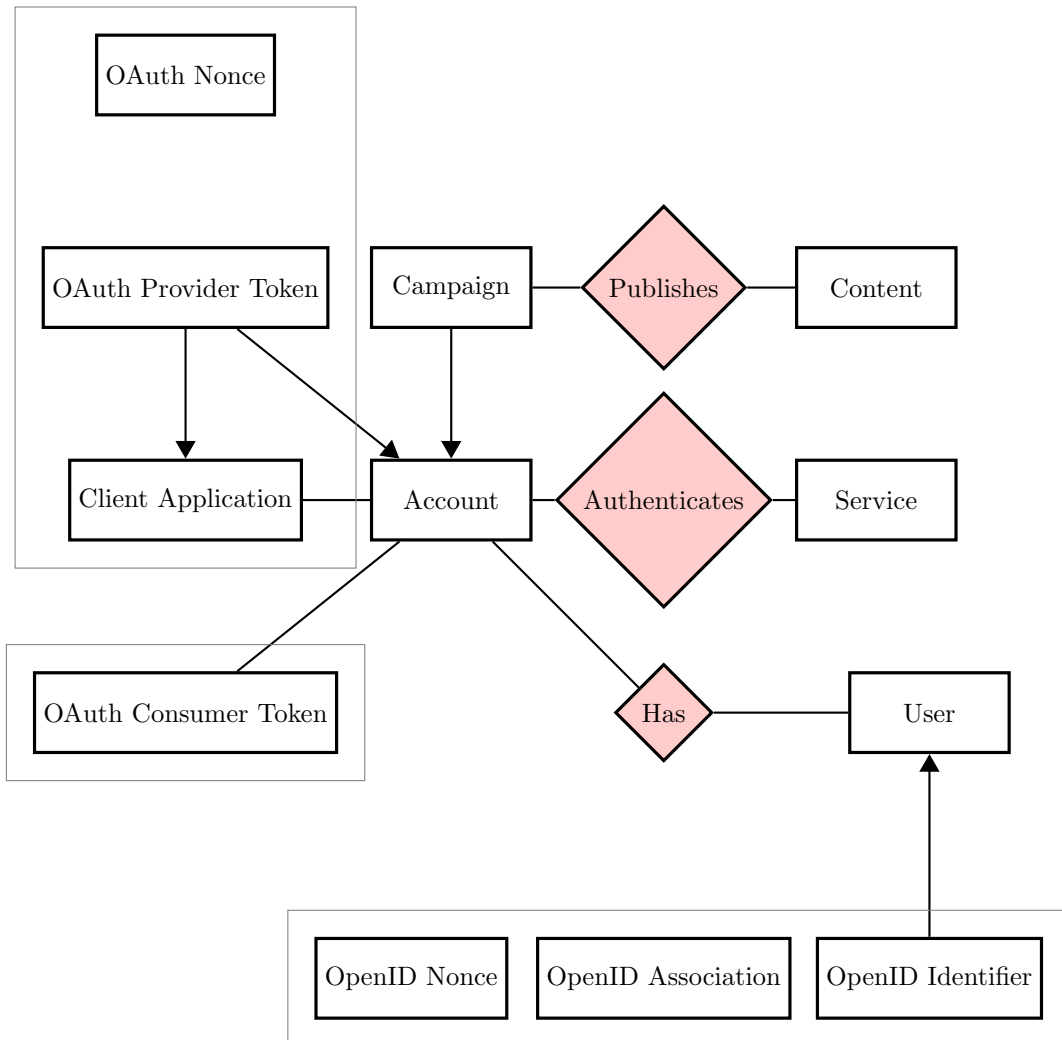
4.3.2 Service Plugins

Chapter 5

Database Design

5.1 Summary

An entity-relationship diagram depicting the data model is shown below.



5.1.1 Advantages of Design

- Decoupled Publishing Platform
 - Allows publishing capabilities to be extended without change to the Campaign Manager.
 - Provides asynchronous publishing so HTTP requests do not time out when pushing a large amount of content.
 - Allows scheduled items to get fired off, even if the Campaign Manager is not fully running.

5.1.2 Disadvantages of Design

5.1.3 Design Rationale

5.2 A Requirements Traceability Matrix

5.2.1 Traceability by Requirement Numbers

5.2.2 Traceability by Design Component