Phase 2
CS 4376.0U1
Team 1
Team URL: **https://cs-4376-cyberminer.herokuapp.com/**

Comet Crawler Web Search Engine System Design Specification: Phase 2

Team 1:

Areebah Fatima
Tyler Hargreaves
Darrien Kramer
Ilhaam Syed
Nathan Heindl
Matthew Bedford

TABLE OF CONTENTS

1. INTRODUCTION:

1.1 BRIEF DESCRIPTION OF PROJECT PHASE 1

1.1.1 PURPOSE

The project's end goal is to create a system that will provide users with relevant information using the keywords entered by the user. The resulting data will be a sorted and filtered list of web page URLs to the end user. The major components of the search engine will include a search interface, indexing system, result filtering, query processing, etc. The search interface will allow the end user to interact with the system, enter search queries, and retrieve information. The indexing system will organize, store, and rank information to allow for a reasonably fast search. Result filtering will ensure that our system provides users with relevant, up-to-date data in sorted order. Finally, the query processing component of the project will be responsible for interpreting user inputs, identifying keywords in said input, performing index lookup, and retrieving results.

This document is the final project report. This document aims to explain our project's Unified Process and highlight all significant project changes. It includes Use Case Diagrams, Sequence Diagrams, and Class Diagrams for our Comet Crawler Web Search Engine.

1.1.2 SCOPE

- This project strictly focuses on the Comet Crawler Search engine and its additional functions.
- The system comprises a search engine including a search interface, indexing system, result filtering, query processing, etc.
- The system does not have privileges for specific functionalities. Therefore, search users can perform searches, add URL entries, modify the index, delete URL entries, and manage system settings.
- The system allows multiple users to search at the same time.

1.2 PROJECT PHASE 2 DESCRIPTION:

1.2.1 PURPOSE:

The project's end goal is to create a system that will provide users with relevant information using the keywords entered by the user. The resulting data will be a sorted and filtered list of web page URLs to the end user. The major components of the search engine will include a search interface, indexing system, result filtering, query processing, etc. The search interface will allow the end user to interact with the system, enter search queries, and retrieve information. The indexing system will organize, store, and rank information to allow for a reasonably fast search. Result filtering will ensure that our system provides users with relevant, up-to-date data in sorted order. Finally, the query processing component of the project will be responsible for interpreting user inputs, identifying keywords in said input, performing index lookup, and retrieving results.

This document is the final project report. This document aims to explain our project's Unified Process and highlight all major project changes. It includes Use Case Diagrams, Sequence Diagrams, Class Diagrams, Package Diagrams, State Transition Diagrams, and Activity Diagrams for our Comet Crawler Web Search Engine.

1.2.2 SCOPE:

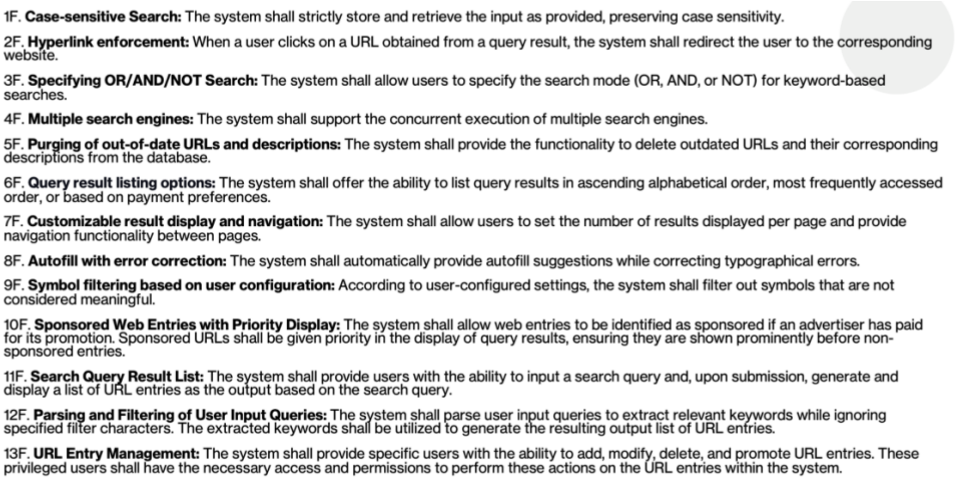
- This project strictly focuses on the Comet Crawler Search engine and its additional functions.
- The system comprises a search engine including a search interface, indexing system, result filtering, query processing, etc.
- All users can interact with the search engine interface, enter search parameters, perform searches, and redirect themselves to their desired result page.
- Only privileged Administrator users can add URLs, Modify the index, delete URL entries, and manage system settings.
- Privileged Administrator users must log in to ensure their credentials; only then will they be allowed to perform privileged operations.
- The system allows multiple users to search at the same time.

1.2.3 ASSUMPTION:

The following assumptions constrain the Comet Crawler Search Engine:

- Because we are implementing a long-term-support version of Ruby on Rails, we will be operating under the presumption that HTTP dispatching, database access, and other core functionality is handled appropriately. This design philosophy allows us to consider engineering the search algorithm and other important application-specific features.
- The application is dependent on Rails and its dependencies. A full list of requirements can be found at: **https://github.com/tyharg/CS-4376/blob/main/Gemfile**.

1.2.4 Functional Requirements:



1F. **Case-sensitive Search:** The system shall strictly store and retrieve the input as provided, preserving case sensitivity.

2F. **Hyperlink enforcement:** When a user clicks on a URL obtained from a query result, the system shall redirect the user to the corresponding website.

3F. **Specifying OR/AND/NOT Search:** The system shall allow users to specify the search mode (OR, AND, or NOT) for keyword-based searches.

4F. **Multiple search engines:** The system shall support the concurrent execution of multiple search engines.

5F. **Purging of out-of-date URLs and descriptions:** The system shall provide the functionality to delete outdated URLs and their corresponding descriptions from the database.

6F. **Query result listing options:** The system shall offer the ability to list query results in ascending alphabetical order, most frequently accessed order, or based on payment preferences.

7F. **Customizable result display and navigation:** The system shall allow users to set the number of results displayed per page and provide navigation functionality between pages.

8F. **Autofill with error correction:** The system shall automatically provide autofill suggestions while correcting typographical errors.

9F. **Symbol filtering based on user configuration:** According to user-configured settings, the system shall filter out symbols that are not considered meaningful.

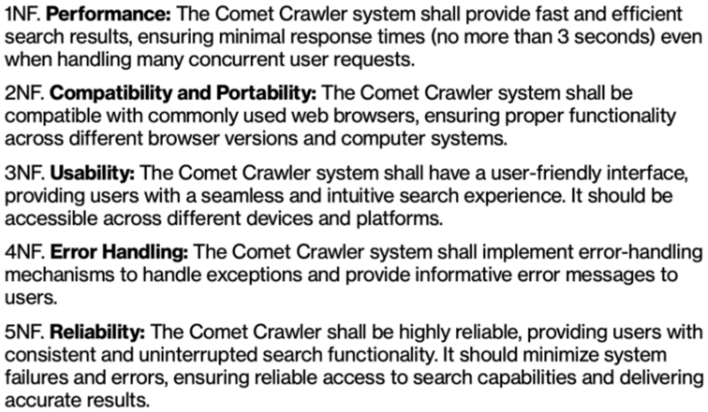
10F. **Sponsored Web Entries with Priority Display:** The system shall allow web entries to be identified as sponsored if an advertiser has paid for its promotion. Sponsored URLs shall be given priority in the display of query results, ensuring they are shown prominently before non-sponsored entries.

11F. **Search Query Result List:** The system shall provide users with the ability to input a search query and, upon submission, generate and display a list of URL entries as the output based on the search query.

12F. **Parsing and Filtering of User Input Queries:** The system shall parse user input queries to extract relevant keywords while ignoring specified filter characters. The extracted keywords shall be utilized to generate the resulting output list of URL entries.

13F. **URL Entry Management:** The system shall provide specific users with the ability to add, modify, delete, and promote URL entries. These privileged users shall have the necessary access and permissions to perform these actions on the URL entries within the system.

1.2.5 Non-Functional Requirements:



1NF. **Performance:** The Comet Crawler system shall provide fast and efficient search results, ensuring minimal response times (no more than 3 seconds) even when handling many concurrent user requests.

2NF. **Compatibility and Portability:** The Comet Crawler system shall be compatible with commonly used web browsers, ensuring proper functionality across different browser versions and computer systems.

3NF. **Usability:** The Comet Crawler system shall have a user-friendly interface, providing users with a seamless and intuitive search experience. It should be accessible across different devices and platforms.

4NF. **Error Handling:** The Comet Crawler system shall implement error-handling mechanisms to handle exceptions and provide informative error messages to users.

5NF. **Reliability:** The Comet Crawler shall be highly reliable, providing users with consistent and uninterrupted search functionality. It should minimize system failures and errors, ensuring reliable access to search capabilities and delivering accurate results.

1.2.6 URL Syntax Context Free Grammar (BNF):

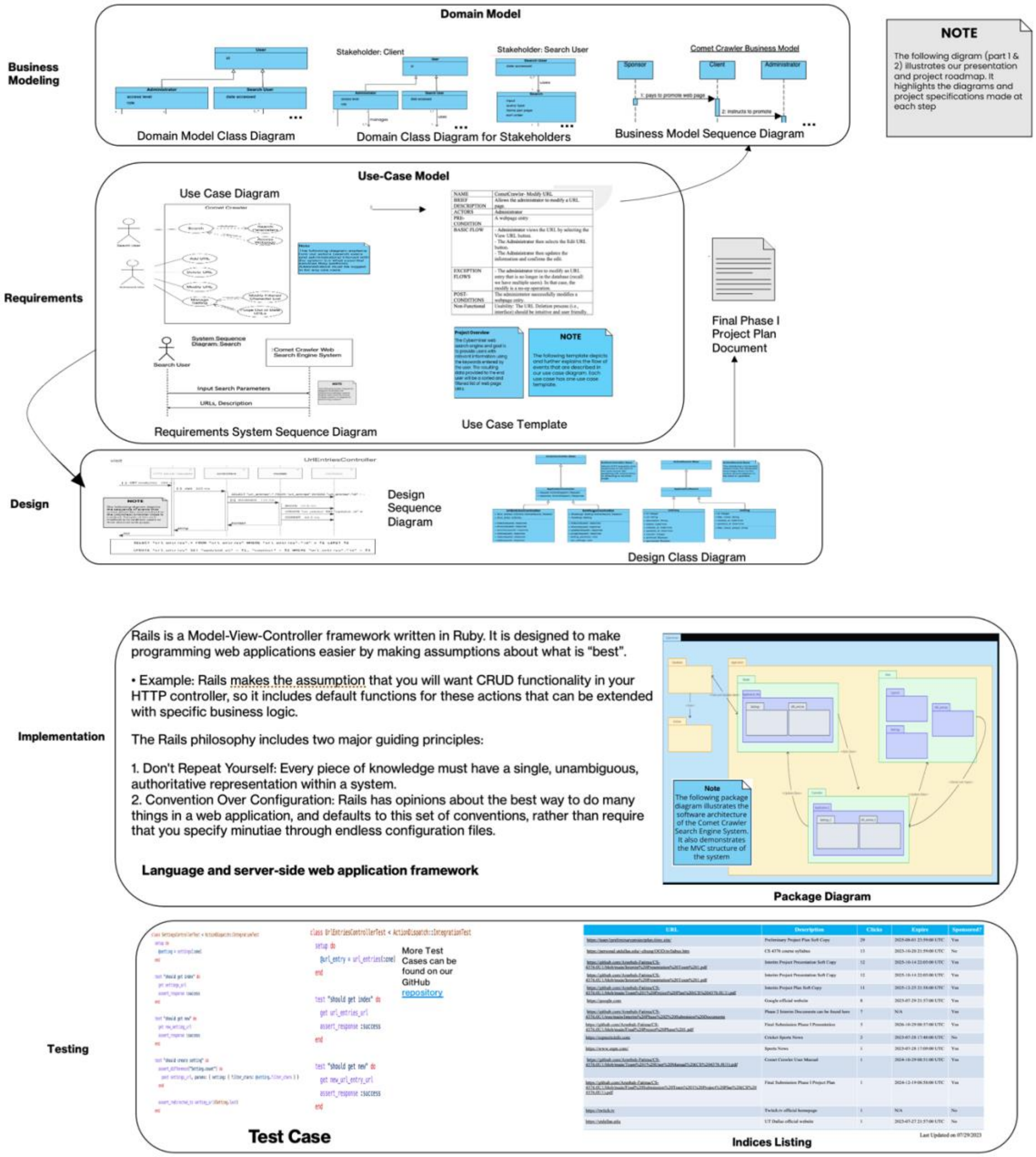
URL ::= {['http://'|'https://']} {www.} Identifier '.' ['edu' | 'com' | 'org' | 'net' | 'gov' | 'mil’]

Identifier ::= [letter | digit]$^+$

letter ::= ['a' | 'b' | 'c' | 'd' |… | 'y' | 'z' | 'A' | 'B' | 'C' | 'D' | ... | 'Y' | 'Z']

digit ::= ['1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' | '0']

1.3 ENHANCED AND UPDATED PROJECT UP / ROAD MAP:



NOTE

The following digram (part 1 & 2) illustrates our presentation and project roadmap. It highlights the diagrams and project specifications made at each step

NAME	CometCrawler- Modify URL
BRIEF DESCRIPTION	Allows the administrator to modify a URL page.
ACTORS	Administrator
PRE-CONDITION	A webpage entry
BASIC FLOW	- Administrator views the URL by selecting the View URL button. - The Administrator then selects the Edit URL button. - The Administrator then updates the information and confirms the edit.
EXCEPTION FLOWS	- The administrator tries to modify an URL entry that is no longer in the database (recall we have multiple users). In that case, the modify is a no-op operation.
POST-CONDITIONS	The administrator successfully modifies a webpage entry.
Non-Functional	Usability: The URL Deletion process (i.e., interface) should be intuitive and user friendly.

Project Overview

The Cybercrawler web search engine and goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

NOTE

The following template depicts and further explains the flow of events that are described in our use case diagram. Each use case has one use case template.

Rails is a Model-View-Controller framework written in Ruby. It is designed to make programming web applications easier by making assumptions about what is "best".

• Example: Rails makes the assumption that you will want CRUD functionality in your HTTP controller, so it includes default functions for these actions that can be extended with specific business logic.

The Rails philosophy includes two major guiding principles:

1. Don't Repeat Yourself: Every piece of knowledge must have a single, unambiguous, authoritative representation within a system.
2. Convention Over Configuration: Rails has opinions about the best way to do many things in a web application, and defaults to this set of conventions, rather than require that you specify minutiae through endless configuration files.

Language and server-side web application framework

Note

The following package diagram illustrates the software architecture of the Comet Crawler Search Engine System. It also demonstrates the MVC structure of the system

```
class SettingsControllerTest < ActionDispatch::IntegrationTest
  setup do
    @setting = settings(:one)
  end

  test "should get index" do
    get settings_url
    assert_response :success
  end

  test "should get new" do
    get new_setting_url
    assert_response :success
  end

  test "should create setting" do
    assert_difference("Setting.count") do
      post settings_url, params: { setting: { filter_chars: @setting.filter_chars } }
    end

    assert_redirected_to setting_url(Setting.last)
  end
```

```
class UrlEntriesControllerTest < ActionDispatch::IntegrationTest
  setup do
    @url_entry = url_entries(:one)
  end

  test "should get index" do
    get url_entries_url
    assert_response :success
  end

  test "should get new" do
    get new_url_entry_url
    assert_response :success
  end
```

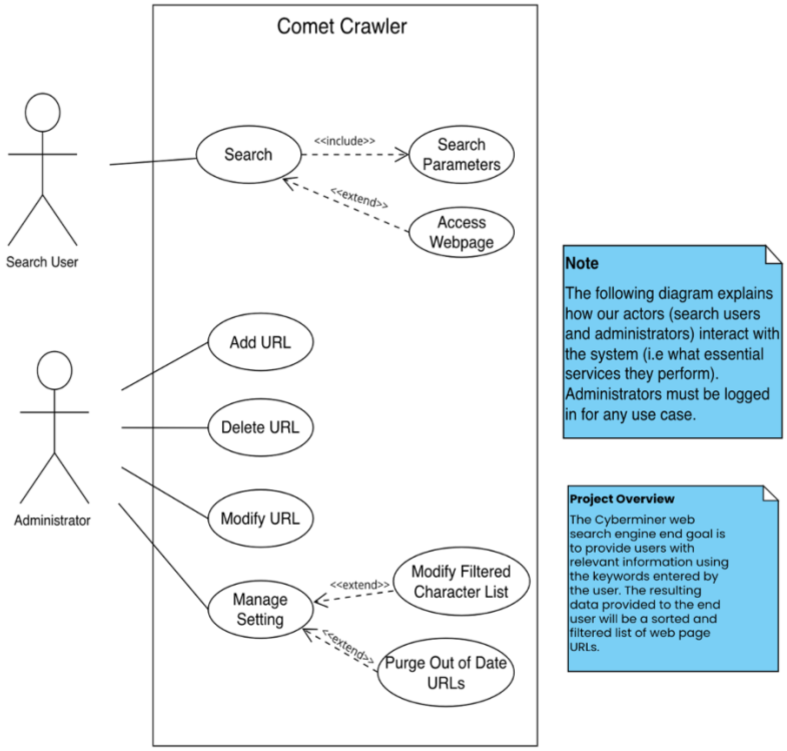
More Test Cases can be found on our GitHub repository

URL	Description	Clicks	Expire	Sponsored?
[illegible]	Preliminary Project Plan Soft Copy	29	2023-08-03 23:59:00 UTC	Yes
https://personal.utdallas.edu/~chung/OOD/syllabus.htm	CS 4376 course syllabus	13	2023-10-20 21:59:00 UTC	No
[illegible]	Interim Project Presentation Soft Copy	12	2023-10-14 22:03:00 UTC	Yes
[illegible]	Interim Project Presentation Soft Copy	12	2023-10-14 22:03:00 UTC	Yes
[illegible]	Interim Project Plan Soft Copy	11	2023-12-25 21:38:00 UTC	Yes
https://google.com	Google official website	8	2023-07-29 21:37:00 UTC	Yes
[illegible]	Phase 2 Interim Documents can be found here	7	N/A	Yes
[illegible]	Final Submission Phase I Presentation	5	2026-10-29 08:37:00 UTC	Yes
[illegible]	Cricket Sports News	2	2023-07-28 17:40:00 UTC	No
https://www.espn.com/	Sports News	1	2023-07-28 17:09:00 UTC	Yes
[illegible]	Comet Crawler User Manual	1	2024-10-29 08:31:00 UTC	Yes
[illegible]	Final Submission Phase I Project Plan	1	2024-12-19 06:38:00 UTC	Yes
https://twitch.tv	Twitch.tv official homepage	1	N/A	No
https://utdallas.edu	UT Dallas official website	1	2023-07-27 21:57:00 UTC	No

Last Updated on 07/29/2023

1.3.1 USE CASE DIAGRAM:

Definition: A use case diagram models the observable behaviors of a system from an actor/user’s perspective. This allows us to communicate essential services and the scope of the system. The main goal of this diagram is to help people understand the system and aid in achieving the client’s primary goals.



Note

The following diagram explains how our actors (search users and administrators) interact with the system (i.e what essential services they perform). Administrators must be logged in for any use case.

Project Overview

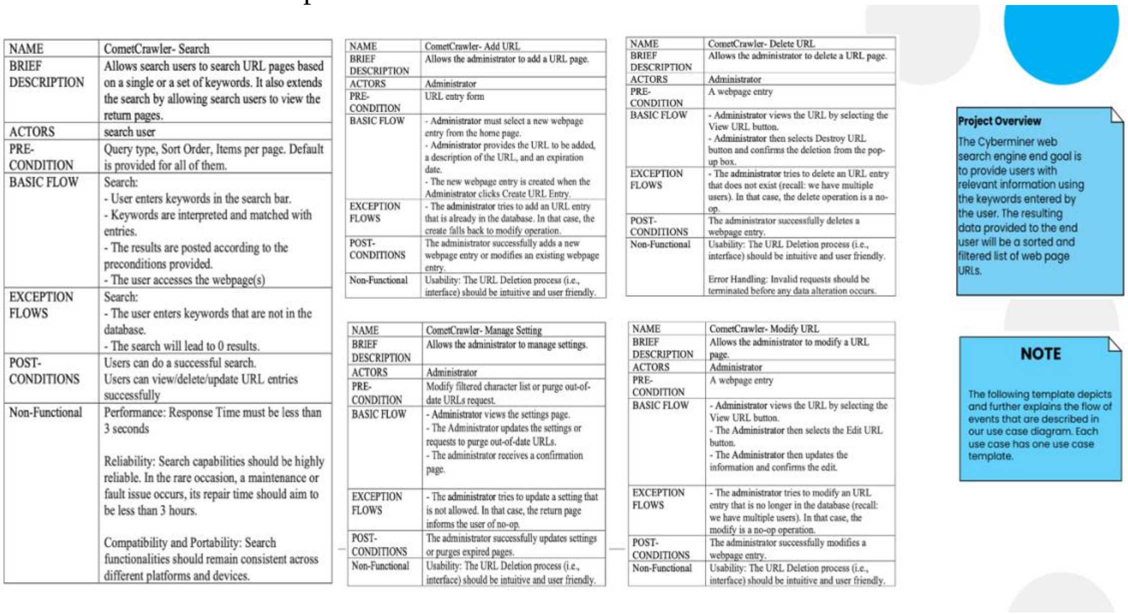
The Cyberminer web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

Figure 1.1 Comet Crawler Web Search Engine System Use case diagram (Part of Requirements)

DESCRIPTION:

The Search User actors will enter search parameters to perform a search and have the option to click a URL from their search results list to redirect themselves to a different webpage. Administrator actors will add URLs, delete URLs, modify URLs, and manage system settings. It is assumed they have logged in before performing any such activities.

Associated Use Case Templates:



NAME	CometCrawler- Search
BRIEF DESCRIPTION	Allows search users to search URL pages based on a single or a set of keywords. It also extends the search by allowing search users to view the return pages.
ACTORS	search user
PRE-CONDITION	Query type, Sort Order, Items per page. Default is provided for all of them.
BASIC FLOW	Search: - User enters keywords in the search bar. - Keywords are interpreted and matched with entries. - The results are posted according to the preconditions provided. - The user accesses the webpage(s)
EXCEPTION FLOWS	Search: - The user enters keywords that are not in the database. - The search will lead to 0 results.
POST-CONDITIONS	Users can do a successful search. Users can view/delete/update URL entries successfully
Non-Functional	Performance: Response Time must be less than 3 seconds Reliability: Search capabilities should be highly reliable. In the rare occasion, a maintenance or fault issue occurs, its repair time should aim to be less than 3 hours. Compatibility and Portability: Search functionalities should remain consistent across different platforms and devices.

NAME	CometCrawler- Add URL
BRIEF DESCRIPTION	Allows the administrator to add a URL page.
ACTORS	Administrator
PRE-CONDITION	URL entry form
BASIC FLOW	- Administrator must select a new webpage entry from the home page. - Administrator provides the URL to be added, a description of the URL, and an expiration date. - The new webpage entry is created when the Administrator clicks Create URL Entry.
EXCEPTION FLOWS	- The administrator tries to add an URL entry that is already in the database. In that case, the create falls back to modify operation.
POST-CONDITIONS	The administrator successfully adds a new webpage entry or modifies an existing webpage entry.
Non-Functional	Usability: The URL Deletion process (i.e., interface) should be intuitive and user friendly.

NAME	CometCrawler- Delete URL
BRIEF DESCRIPTION	Allows the administrator to delete a URL page.
ACTORS	Administrator
PRE-CONDITION	A webpage entry
BASIC FLOW	- Administrator views the URL by selecting the View URL button. - Administrator then selects Destroy URL button and confirms the deletion from the pop-up box.
EXCEPTION FLOWS	- The administrator tries to delete an URL entry that does not exist (recall: we have multiple users). In that case, the delete operation is a no-op.
POST-CONDITIONS	The administrator successfully deletes a webpage entry.
Non-Functional	Usability: The URL Deletion process (i.e., interface) should be intuitive and user friendly. Error Handling: Invalid requests should be terminated before any data alteration occurs.

Project Overview

The Cyberminer web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

NAME	CometCrawler- Manage Setting
BRIEF DESCRIPTION	Allows the administrator to manage settings.
ACTORS	Administrator
PRE-CONDITION	Modify filtered character list or purge out-of-date URLs request.
BASIC FLOW	- Administrator views the settings page. - The Administrator updates the settings or requests to purge out-of-date URLs. - The administrator receives a confirmation page.
EXCEPTION FLOWS	- The administrator tries to update a setting that is not allowed. In that case, the return page informs the user of no-op.
POST-CONDITIONS	The administrator successfully updates settings or purges expired pages.
Non-Functional	Usability: The URL Deletion process (i.e., interface) should be intuitive and user friendly.

NAME	CometCrawler- Modify URL
BRIEF DESCRIPTION	Allows the administrator to modify a URL page.
ACTORS	Administrator
PRE-CONDITION	A webpage entry
BASIC FLOW	- Administrator views the URL by selecting the View URL button. - The Administrator then selects the Edit URL button. - The Administrator then updates the information and confirms the edit.
EXCEPTION FLOWS	- The administrator tries to modify an URL entry that is no longer in the database (recall: we have multiple users). In that case, the modify is a no-op operation.
POST-CONDITIONS	The administrator successfully modifies a webpage entry.
Non-Functional	Usability: The URL Deletion process (i.e., interface) should be intuitive and user friendly.

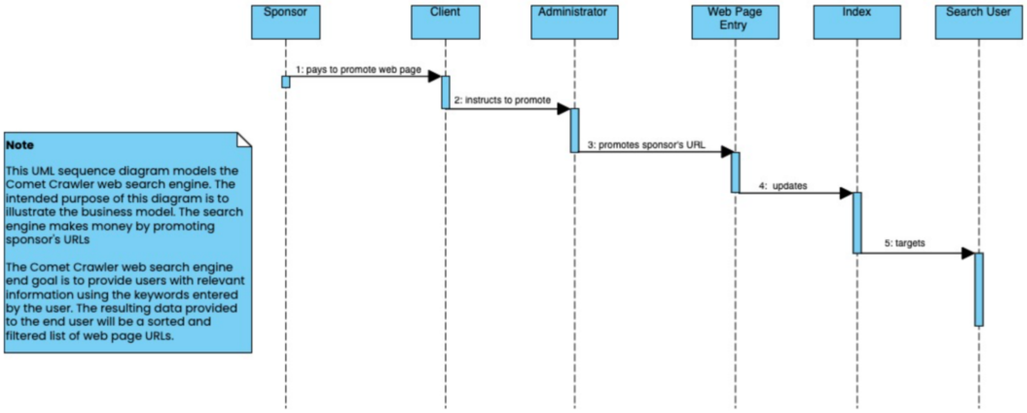
NOTE

The following template depicts and further explains the flow of events that are described in our use case diagram. Each use case has one use case template.

1.3.2 SEQUENCE DIAGRAMS:

Definition: A sequence diagram models the sequence of interactions (as time flows downward) to help understand and communicate how various system components cooperate to bring a use case functionality into fruition.

BUSINESS MODEL SEQUENCE DIAGRAMS:



Note

This UML sequence diagram models the Comet Crawler web search engine. The intended purpose of this diagram is to illustrate the business model. The search engine makes money by promoting sponsor's URLs

The Comet Crawler web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

Sequence Diagram to Model the Comet Crawler Web Search Engine's Business Model

SYSTEM SEQUENCE DIAGRAMS:

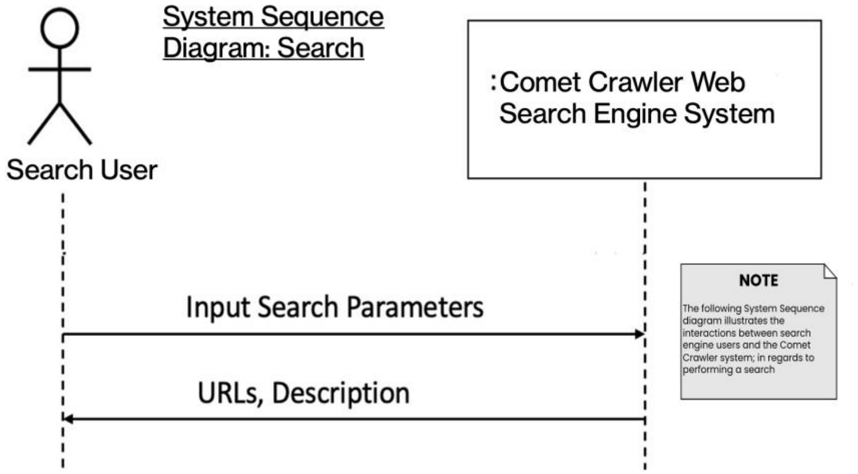

System Sequence Diagram: Search
Search User
:Comet Crawler Web Search Engine System
Input Search Parameters
URLs, Description
NOTE
The following System Sequence diagram illustrates the interactions between search engine users and the Comet Crawler system; in regards to performing a search

Figure 1.2 Comet Crawler Web Search Engine System Sequence Diagram for Search

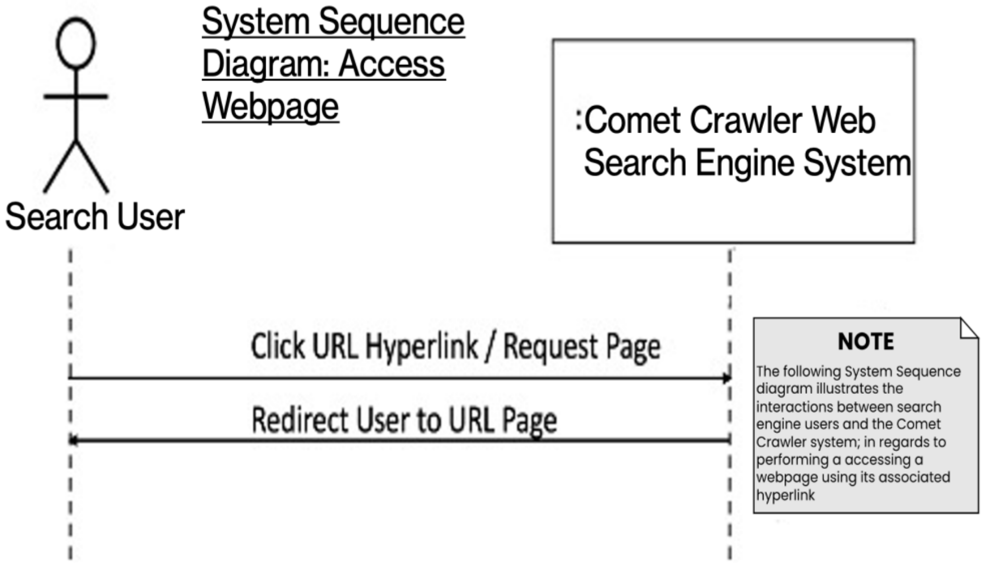

System Sequence Diagram: Access Webpage
Search User
:Comet Crawler Web Search Engine System
Click URL Hyperlink / Request Page
Redirect User to URL Page
NOTE
The following System Sequence diagram illustrates the interactions between search engine users and the Comet Crawler system; in regards to performing a accessing a webpage using its associated hyperlink

Figure 1.3 Comet Crawler Web Search Engine System Sequence Diagram for Accessing Webpages

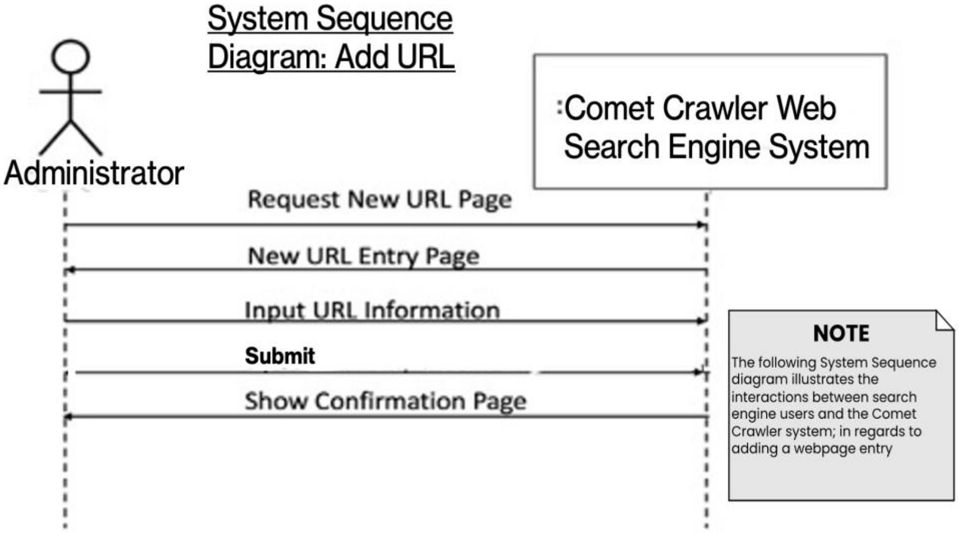

System Sequence Diagram: Add URL
Administrator
:Comet Crawler Web Search Engine System
Request New URL Page
New URL Entry Page
Input URL Information
Submit
Show Confirmation Page
NOTE
The following System Sequence diagram illustrates the interactions between search engine users and the Comet Crawler system; in regards to adding a webpage entry

Figure 1.4 Comet Crawler Web Search Engine System Sequence Diagram for Adding URLs

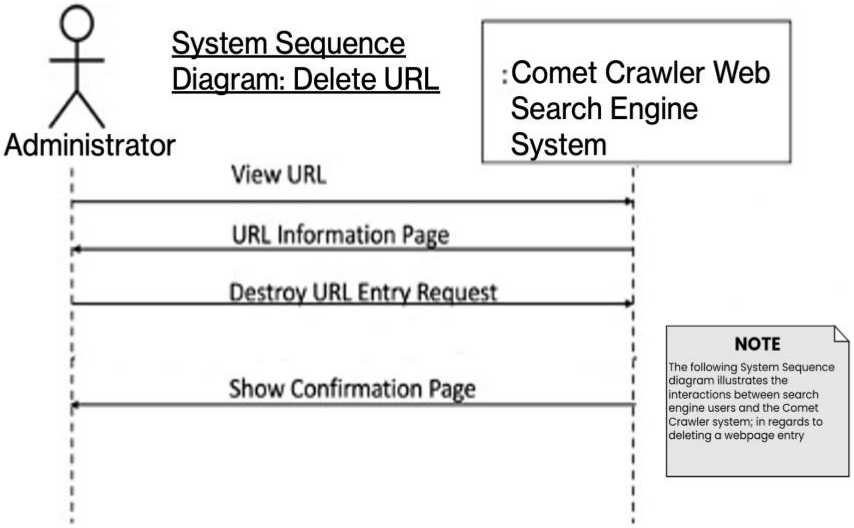
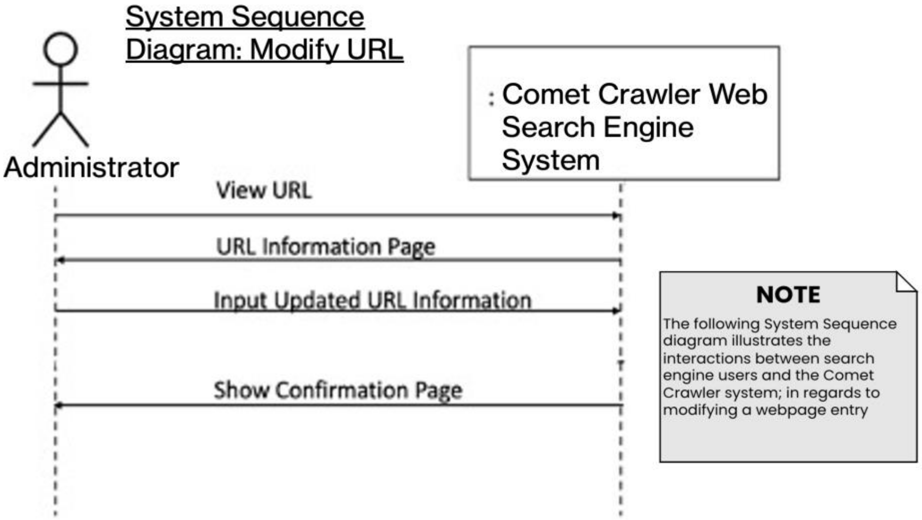
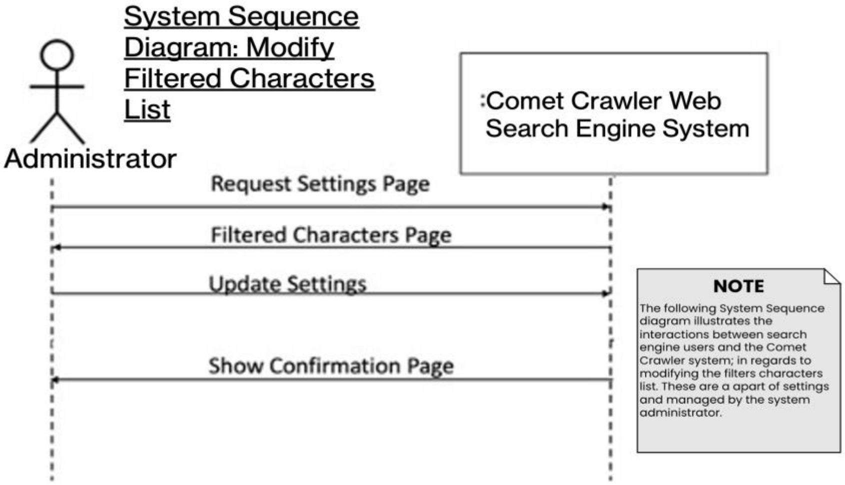

System Sequence Diagram: Delete URL
Administrator
:Comet Crawler Web Search Engine System
View URL
URL Information Page
Destroy URL Entry Request
Show Confirmation Page
NOTE
The following System Sequence diagram illustrates the interactions between search engine users and the Comet Crawler system; in regards to deleting a webpage entry

Figure 1.5 Comet Crawler Web Search Engine System Sequence Diagram for Deleting URLs


System Sequence Diagram: Modify URL
Administrator
: Comet Crawler Web Search Engine System
View URL
URL Information Page
Input Updated URL Information
Show Confirmation Page
NOTE

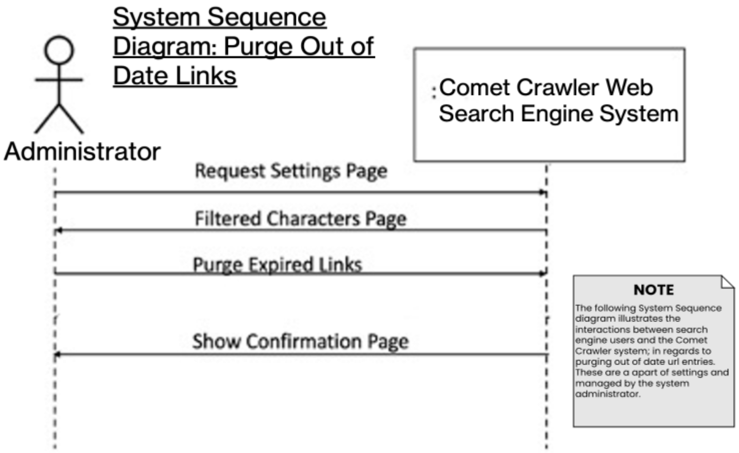
The following System Sequence diagram illustrates the interactions between search engine users and the Comet Crawler system; in regards to modifying a webpage entry

Figure 1.6 Comet Crawler Web Search Engine System Sequence Diagram for Modifying URLs



The following System Sequence diagram illustrates the interactions between search engine users and the Comet Crawler system; in regards to modifying the filters characters list. These are a apart of settings and managed by the system administrator.

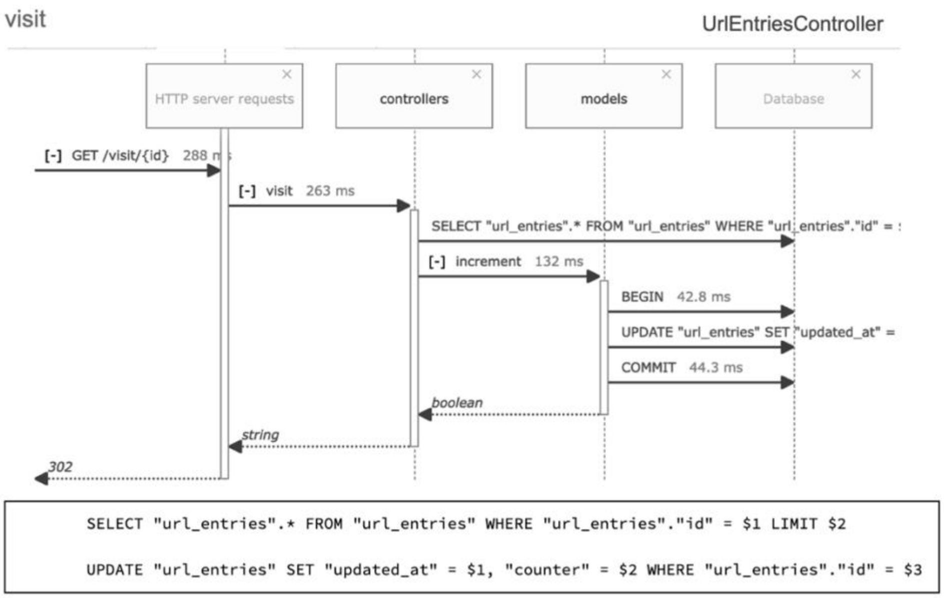
Figure 1.7 Comet Crawler Web Search Engine System Sequence Diagram for Modifying the System Setting’s Filtered Characters List



The following System Sequence diagram illustrates the interactions between search engine users and the Comet Crawler system; in regards to purging out of date url entries. These are a apart of settings and managed by the system administrator.

Figure 1.8 Comet Crawler Web Search Engine System Sequence Diagram for Purging Out-of-Date URLs

DESIGN SEQUENCE DIAGRAM:



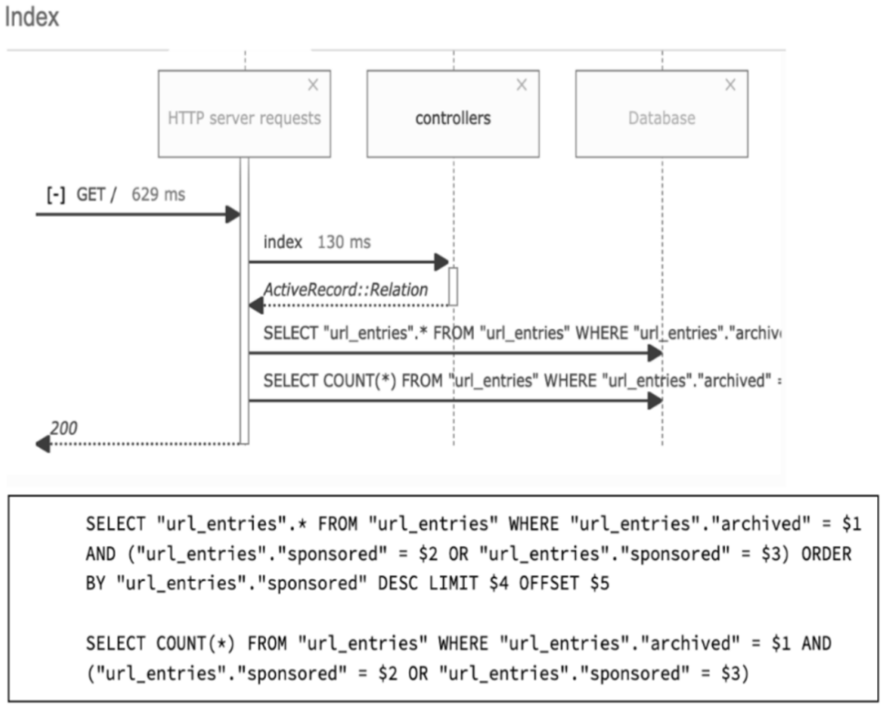
```
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."id" = $1 LIMIT $2

UPDATE "url_entries" SET "updated_at" = $1, "counter" = $2 WHERE "url_entries"."id" = $3
```

NOTE

The following digram depicts the sequence of events that occur when the visit method in the UrlEntriesController class is invoked. The job of the visit method is to redirect users to their desired web page

Figure 1.9 Comet Crawler Web Search Engine Sequence diagram for visit function


Index
HTTP server requests
controllers
Database
[-] GET / 629 ms
index 130 ms
ActiveRecord::Relation
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."archiv
SELECT COUNT(*) FROM "url_entries" WHERE "url_entries"."archived" =
200

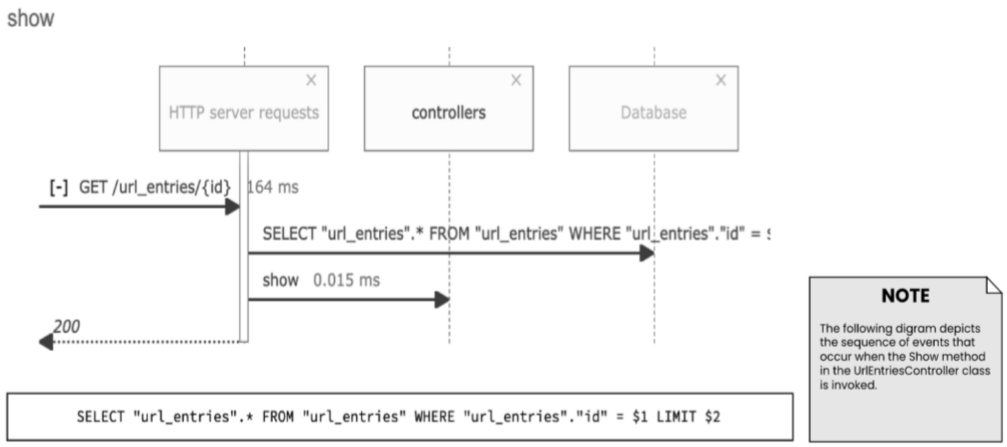
```
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."archived" = $1
AND ("url_entries"."sponsored" = $2 OR "url_entries"."sponsored" = $3) ORDER
BY "url_entries"."sponsored" DESC LIMIT $4 OFFSET $5

SELECT COUNT(*) FROM "url_entries" WHERE "url_entries"."archived" = $1 AND
("url_entries"."sponsored" = $2 OR "url_entries"."sponsored" = $3)
```

NOTE

The following digram depicts the sequence of events that occur when the Index method in the UrlEntriesController class is invoked.

Figure 1.10 Comet Crawler Web Search Engine Sequence diagram for the System's Index (Part of UrlEntriesController)

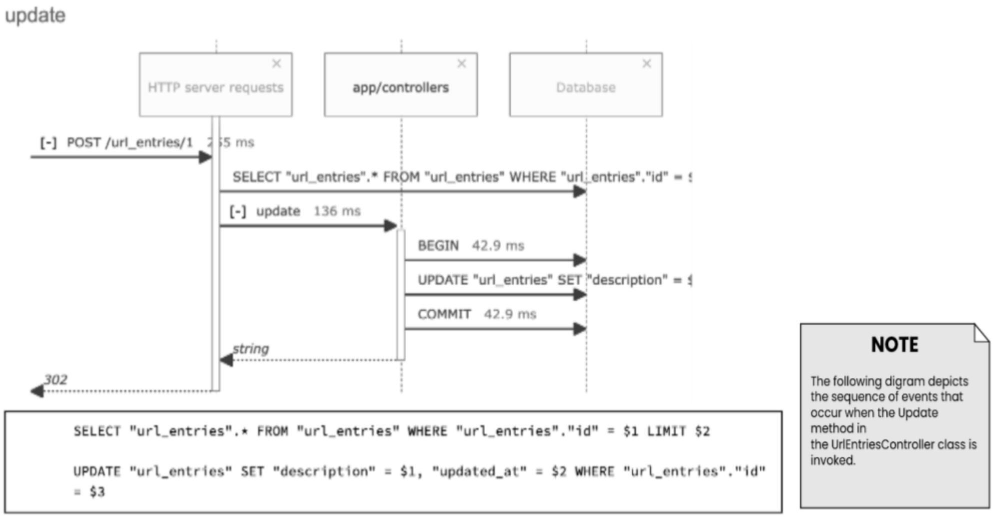

show
HTTP server requests
controllers
Database
[-] GET /url_entries/{id} 164 ms
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."id" = $
show 0.015 ms
200

```
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."id" = $1 LIMIT $2
```

NOTE

The following digram depicts the sequence of events that occur when the Show method in the UrlEntriesController class is invoked.

Figure 1.11 Comet Crawler Web Search Engine Sequence diagram for the show function. (Part of UrlEntriesController)


update
HTTP server requests
app/controllers
Database
[-] POST /url_entries/1 2 5 ms
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."id" = $
[-] update 136 ms
BEGIN 42.9 ms
UPDATE "url_entries" SET "description" = $
COMMIT 42.9 ms
string
302

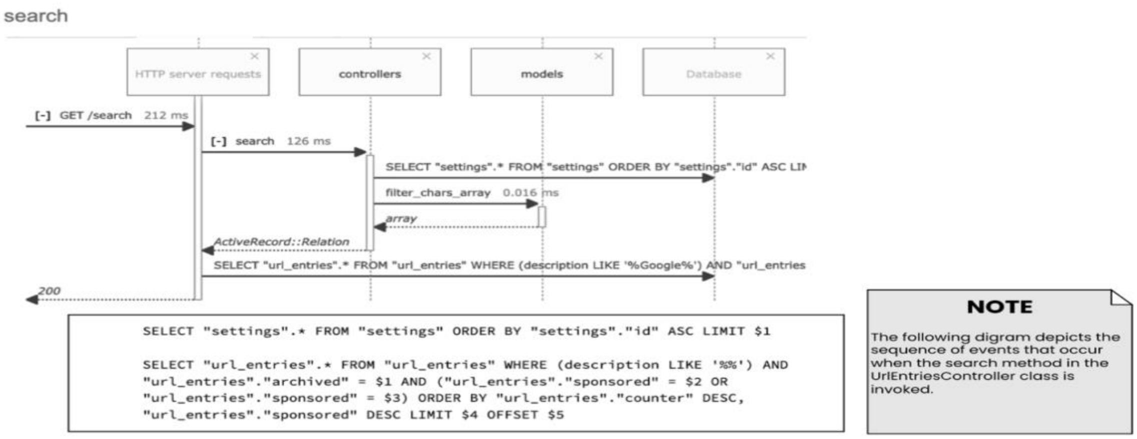
```
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."id" = $1 LIMIT $2

UPDATE "url_entries" SET "description" = $1, "updated_at" = $2 WHERE "url_entries"."id"
= $3
```

NOTE

The following digram depicts the sequence of events that occur when the Update method in the UrlEntriesController class is invoked.

Figure 1.12 Comet Crawler Web Search Engine Sequence diagram for the update function. (Part of UrlEntriesController)


search
HTTP server requests
controllers
models
Database
[-] GET /search 212 ms
[-] search 126 ms
SELECT "settings".* FROM "settings" ORDER BY "settings"."id" ASC LIM
filter_chars_array 0.016 ms
array
ActiveRecord::Relation
SELECT "url_entries".* FROM "url_entries" WHERE (description LIKE '%Google%') AND "url_entries
200

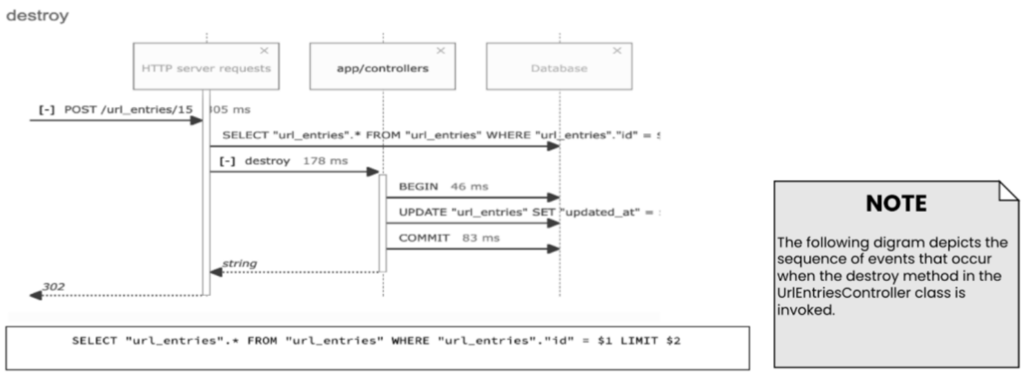
```
SELECT "settings".* FROM "settings" ORDER BY "settings"."id" ASC LIMIT $1

SELECT "url_entries".* FROM "url_entries" WHERE (description LIKE '%%') AND
"url_entries"."archived" = $1 AND ("url_entries"."sponsored" = $2 OR
"url_entries"."sponsored" = $3) ORDER BY "url_entries"."counter" DESC,
"url_entries"."sponsored" DESC LIMIT $4 OFFSET $5
```

NOTE

The following digram depicts the sequence of events that occur when the search method in the UrlEntriesController class is invoked.

Figure 1.13 Comet Crawler Web Search Engine Sequence diagram for the search function. (Part of UrlEntriesController)

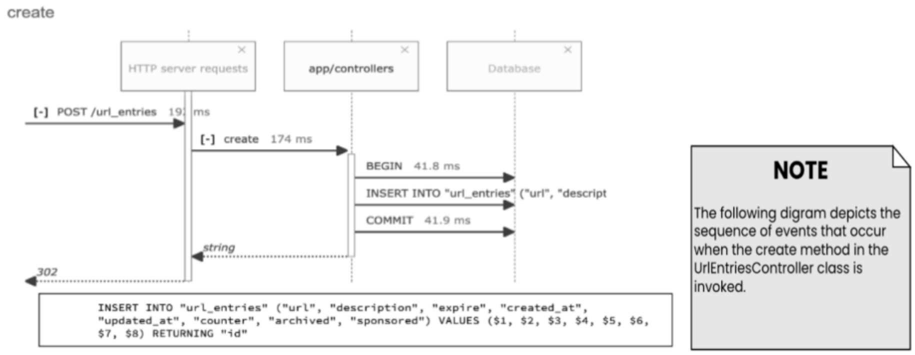

destroy
HTTP server requests
app/controllers
Database
[-] POST /url_entries/15 05 ms
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."id" =
[-] destroy 178 ms
BEGIN 46 ms
UPDATE "url_entries" SET "updated_at" =
COMMIT 83 ms
string
302

```
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."id" = $1 LIMIT $2
```

NOTE

The following digram depicts the sequence of events that occur when the destroy method in the UrlEntriesController class is invoked.

Figure 1.14 Comet Crawler Web Search Engine Sequence diagram for the destroy function. (Part of UrlEntriesController)

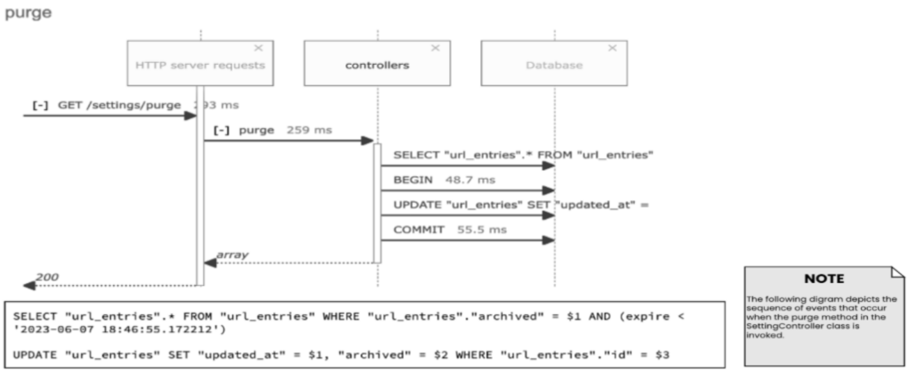


```
INSERT INTO "url_entries" ("url", "description", "expire", "created_at",
"updated_at", "counter", "archived", "sponsored") VALUES ($1, $2, $3, $4, $5, $6,
$7, $8) RETURNING "id"
```

NOTE

The following digram depicts the sequence of events that occur when the create method in the UrlEntriesController class is invoked.

Figure 1.15 Comet Crawler Web Search Engine Sequence diagram for the create function. (Part of UrlEntriesController)


purge
HTTP server requests
controllers
Database
[-] GET /settings/purge 93 ms
[-] purge 259 ms
SELECT "url_entries".* FROM "url_entries"
BEGIN 48.7 ms
UPDATE "url_entries" SET "updated_at" =
COMMIT 55.5 ms
array
200

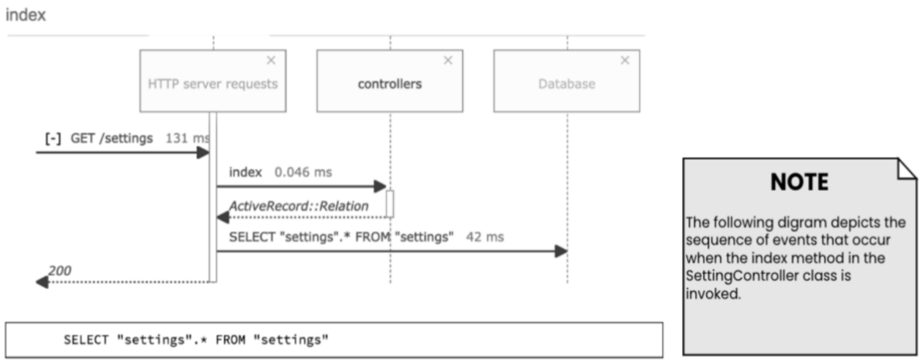
```
SELECT "url_entries".* FROM "url_entries" WHERE "url_entries"."archived" = $1 AND (expire <
'2023-06-07 18:46:55.172212')

UPDATE "url_entries" SET "updated_at" = $1, "archived" = $2 WHERE "url_entries"."id" = $3
```

NOTE

The following digram depicts the sequence of events that occur when the purge method in the SettingController class is invoked.

Figure 1.16 Comet Crawler Web Search Engine Sequence diagram for the purging expired links function. (Part of SettingController)

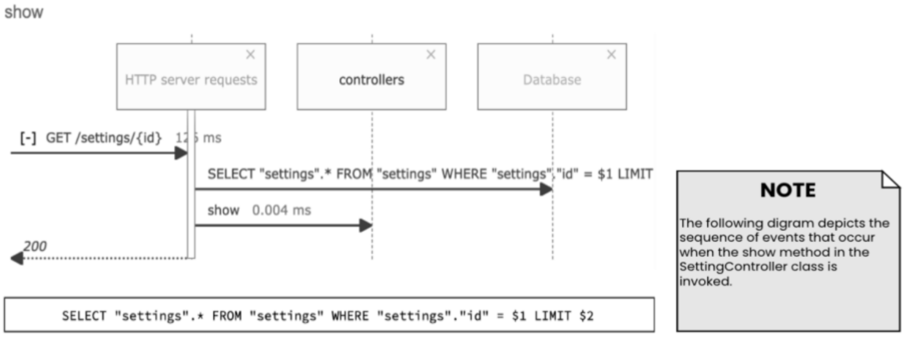


```
SELECT "settings".* FROM "settings"
```

NOTE

The following digram depicts the sequence of events that occur when the index method in the SettingController class is invoked.

Figure 1.17 Comet Crawler Web Search Engine Sequence diagram for the Setting’s Index. (Part of SettingController)

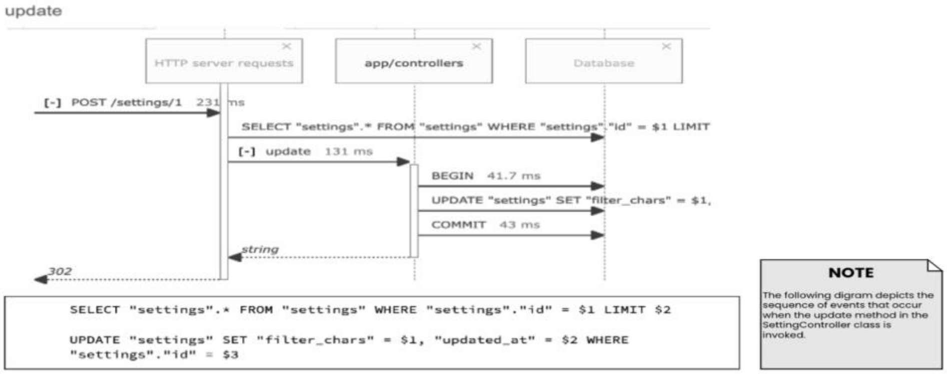

show
HTTP server requests
controllers
Database
[-] GET /settings/{id} 125 ms
SELECT "settings".* FROM "settings" WHERE "settings"."id" = $1 LIMIT
show 0.004 ms
200

```
SELECT "settings".* FROM "settings" WHERE "settings"."id" = $1 LIMIT $2
```

NOTE

The following digram depicts the sequence of events that occur when the show method in the SettingController class is invoked.

Figure 1.18 Comet Crawler Web Search Engine Sequence diagram for the show settings function. (Part of SettingController)


update
HTTP server requests
app/controllers
Database
[-] POST /settings/1 231 ms
SELECT "settings".* FROM "settings" WHERE "settings"."id" = $1 LIMIT
[-] update 131 ms
BEGIN 41.7 ms
UPDATE "settings" SET "filter_chars" = $1,
COMMIT 43 ms
string
302

```
SELECT "settings".* FROM "settings" WHERE "settings"."id" = $1 LIMIT $2

UPDATE "settings" SET "filter_chars" = $1, "updated_at" = $2 WHERE
"settings"."id" = $3
```

NOTE

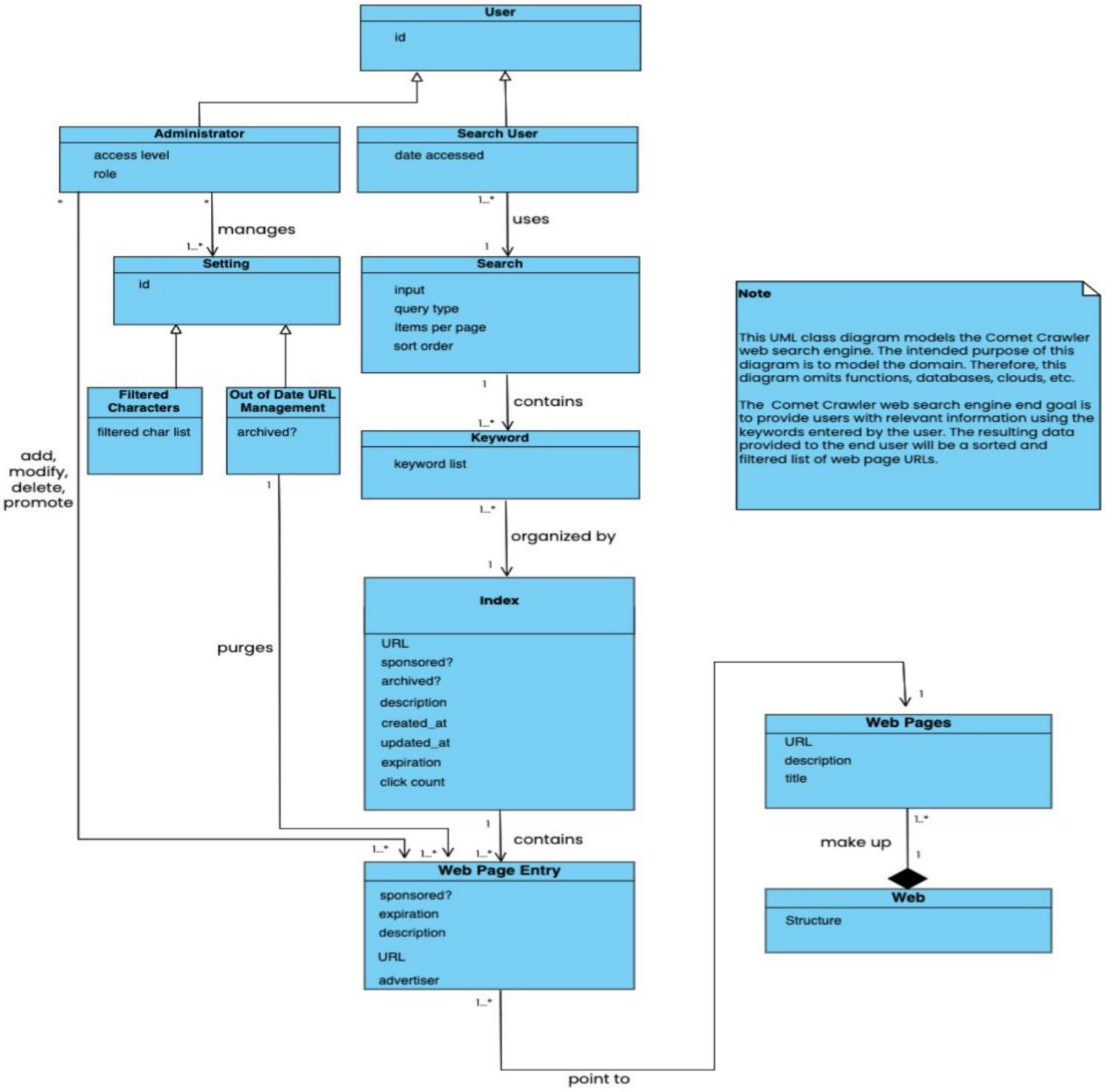
The following digram depicts the sequence of events that occur when the update method in the SettingController class is invoked.

Figure 1.19 Comet Crawler Web Search Engine Sequence diagram for the update settings function. (Part of SettingController)

1.3.3 CLASS DIAGRAM:

Definition: A class diagram provides a static view of the system’s structure. It highlights key concepts from the problem and solution domain to better understand the structure of classes and how they interact. A class diagram shows classes, class attributes, class methods, and relationships between classes.

Class Diagram for Domain Model:



This UML class diagram models the Comet Crawler web search engine. The intended purpose of this diagram is to model the domain. Therefore, this diagram omits functions, databases, clouds, etc.

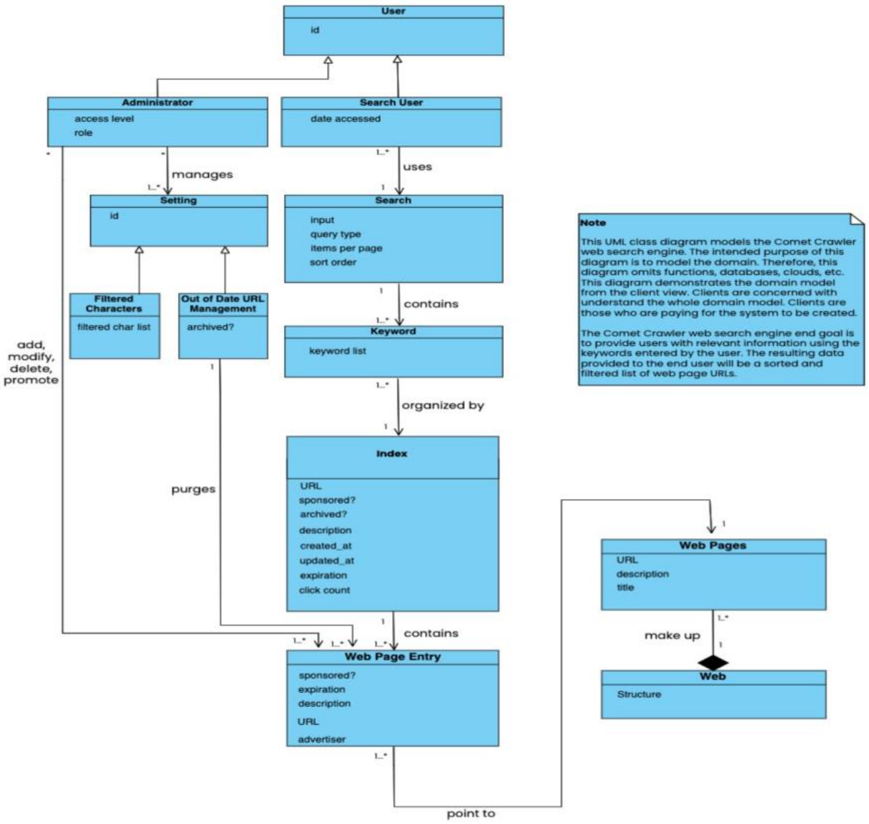
The Comet Crawler web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

Figure 1.20 Comet Crawler Web Search Engine System Class Diagram

Comet Crawler Stakeholders:

- **Administrator:** Are privileged users with the ability to add, delete, and modify URL entries. In addition to this, they also manage the system filtering and purging settings.
- **Client:** Are the ultimate owner of the project. This ownership allows them to specify project requirements, making them the recipients of project deliverables and the resulting product.
- **Sponsor:** Are the individuals that pay money to promote their websites to the top of the search result list. Stakeholders are the central piece of the business model as they bring monetary incentives for clients.
- **Search Users:** Are the individuals who interact with the Comet Crawler System. They will provide the search parameters and view the associated results. Sponsors will target these users.

Class Diagrams Based on Stakeholder Views:

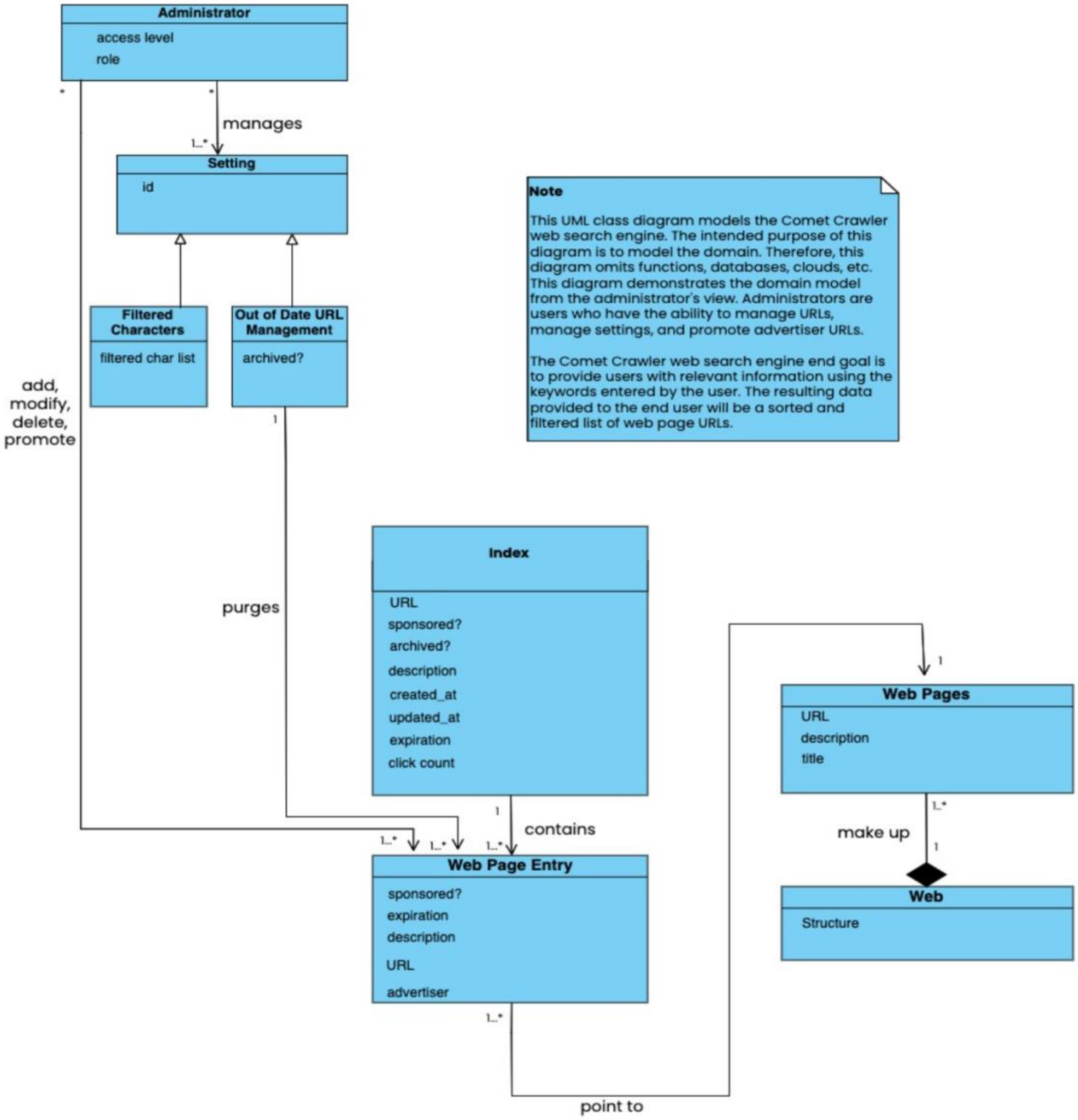


Note

This UML class diagram models the Comet Crawler web search engine. The intended purpose of this diagram is to model the domain. Therefore, this diagram omits functions, databases, clouds, etc. This diagram demonstrates the domain model from the client view. Clients are concerned with understand the whole domain model. Clients are those who are paying for the system to be created.

The Comet Crawler web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

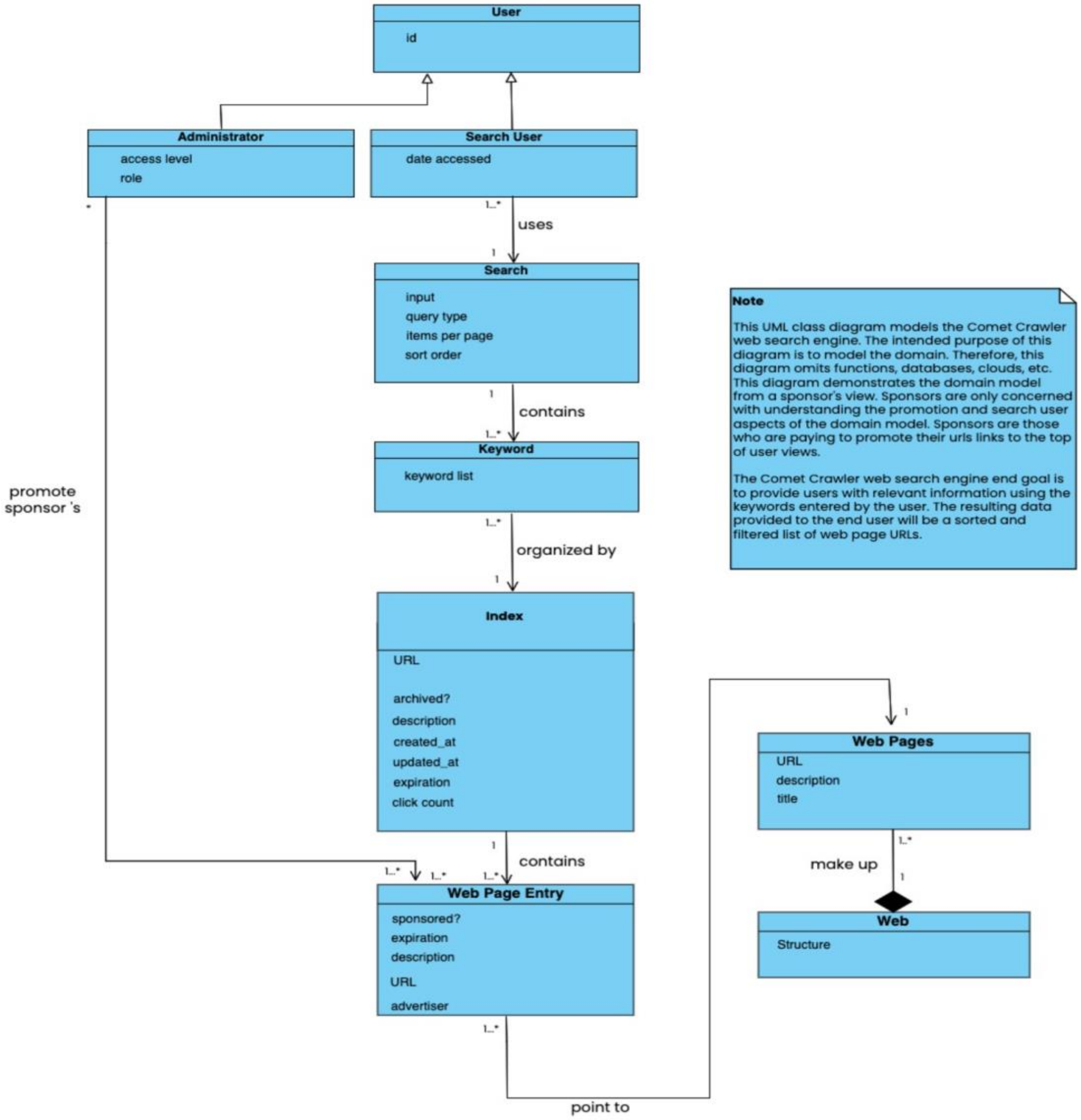
Figure 1.21 Comet Crawler Web Search Engine System Class Diagram (Based on Client's View)



This UML class diagram models the Comet Crawler web search engine. The intended purpose of this diagram is to model the domain. Therefore, this diagram omits functions, databases, clouds, etc. This diagram demonstrates the domain model from the administrator's view. Administrators are users who have the ability to manage URLs, manage settings, and promote advertiser URLs.

The Comet Crawler web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

Figure 1.22 Comet Crawler Web Search Engine System Class Diagram (Based on Administrator’s View)

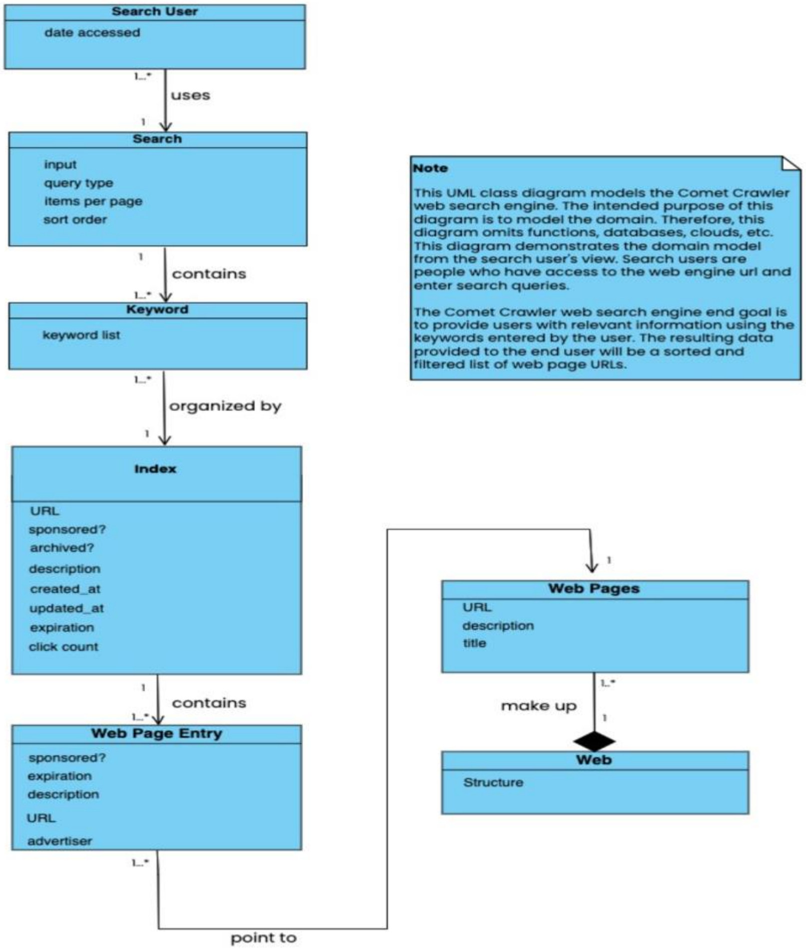


Note

This UML class diagram models the Comet Crawler web search engine. The intended purpose of this diagram is to model the domain. Therefore, this diagram omits functions, databases, clouds, etc. This diagram demonstrates the domain model from a sponsor's view. Sponsors are only concerned with understanding the promotion and search user aspects of the domain model. Sponsors are those who are paying to promote their urls links to the top of user views.

The Comet Crawler web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

Figure 1.23 Comet Crawler Web Search Engine System Class Diagram (Based on Sponsor’s View)



Note

This UML class diagram models the Comet Crawler web search engine. The intended purpose of this diagram is to model the domain. Therefore, this diagram omits functions, databases, clouds, etc. This diagram demonstrates the domain model from the search user's view. Search users are people who have access to the web engine url and enter search queries.

The Comet Crawler web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.

Figure 1.24 Comet Crawler Web Search Engine System Class Diagram (Based on Search User's View)

Design Class Diagrams:

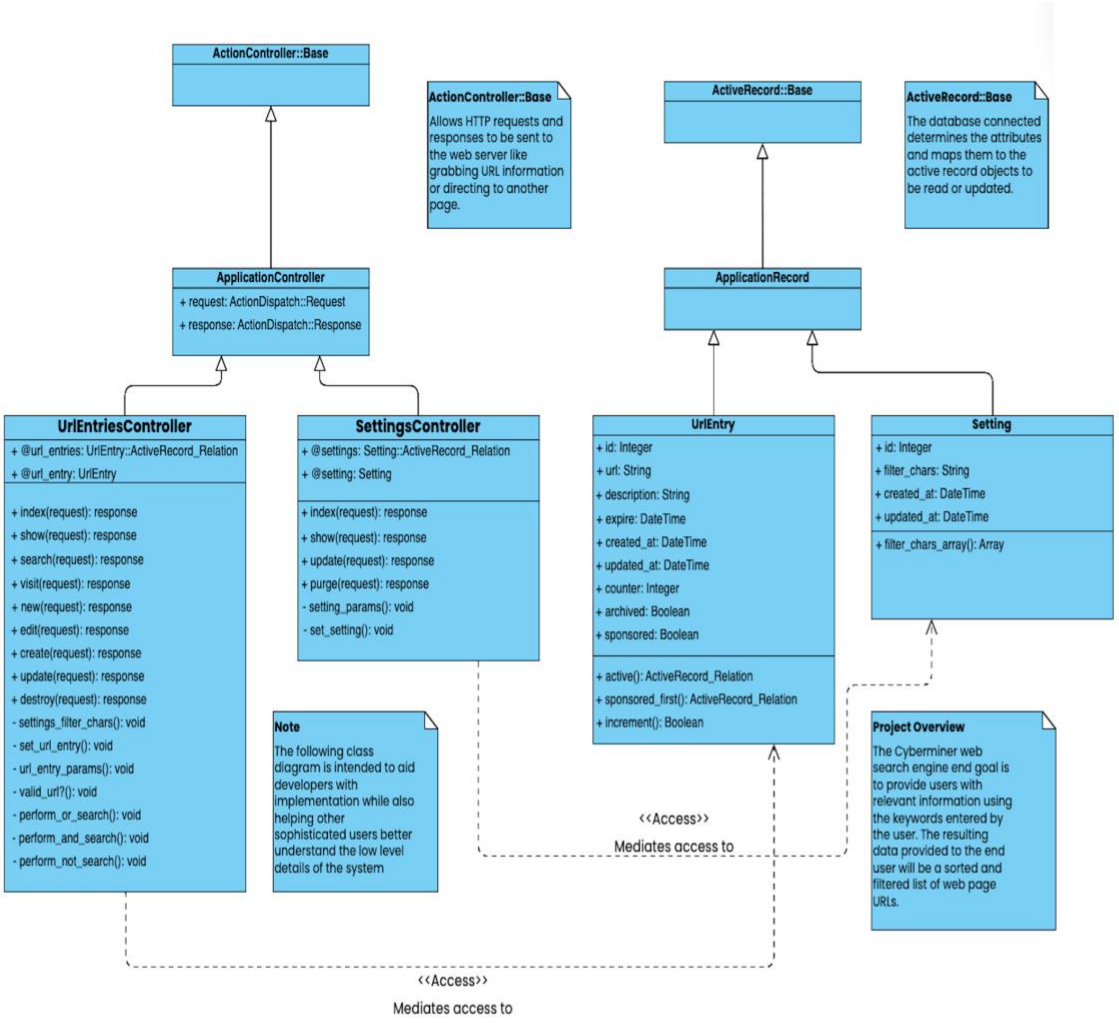

ActionController::Base
ActionController::Base
Allows HTTP requests and responses to be sent to the web server like grabbing URL information or directing to another page.
ActiveRecord::Base
ActiveRecord::Base
The database connected determines the attributes and maps them to the active record objects to be read or updated.
ApplicationController
+ request: ActionDispatch::Request
+ response: ActionDispatch::Response
ApplicationRecord
UrlEntriesController
+ @url_entries: UrlEntry::ActiveRecord_Relation
+ @url_entry: UrlEntry
+ index(request): response
+ show(request): response
+ search(request): response
+ visit(request): response
+ new(request): response
+ edit(request): response
+ create(request): response
+ update(request): response
+ destroy(request): response
- settings_filter_chars(): void
- set_url_entry(): void
- url_entry_params(): void
- valid_url?(): void
- perform_or_search(): void
- perform_and_search(): void
- perform_not_search(): void
SettingsController
+ @settings: Setting::ActiveRecord_Relation
+ @setting: Setting
+ index(request): response
+ show(request): response
+ update(request): response
+ purge(request): response
- setting_params(): void
- set_setting(): void
UrlEntry
+ id: Integer
+ url: String
+ description: String
+ expire: DateTime
+ created_at: DateTime
+ updated_at: DateTime
+ counter: Integer
+ archived: Boolean
+ sponsored: Boolean
+ active(): ActiveRecord_Relation
+ sponsored_first(): ActiveRecord_Relation
+ increment(): Boolean
Setting
+ id: Integer
+ filter_chars: String
+ created_at: DateTime
+ updated_at: DateTime
+ filter_chars_array(): Array
Note
The following class diagram is intended to aid developers with implementation while also helping other sophisticated users better understand the low level details of the system
Project Overview
The Cyberminer web search engine end goal is to provide users with relevant information using the keywords entered by the user. The resulting data provided to the end user will be a sorted and filtered list of web page URLs.
<<Access>>
Mediates access to
<<Access>>
Mediates access to

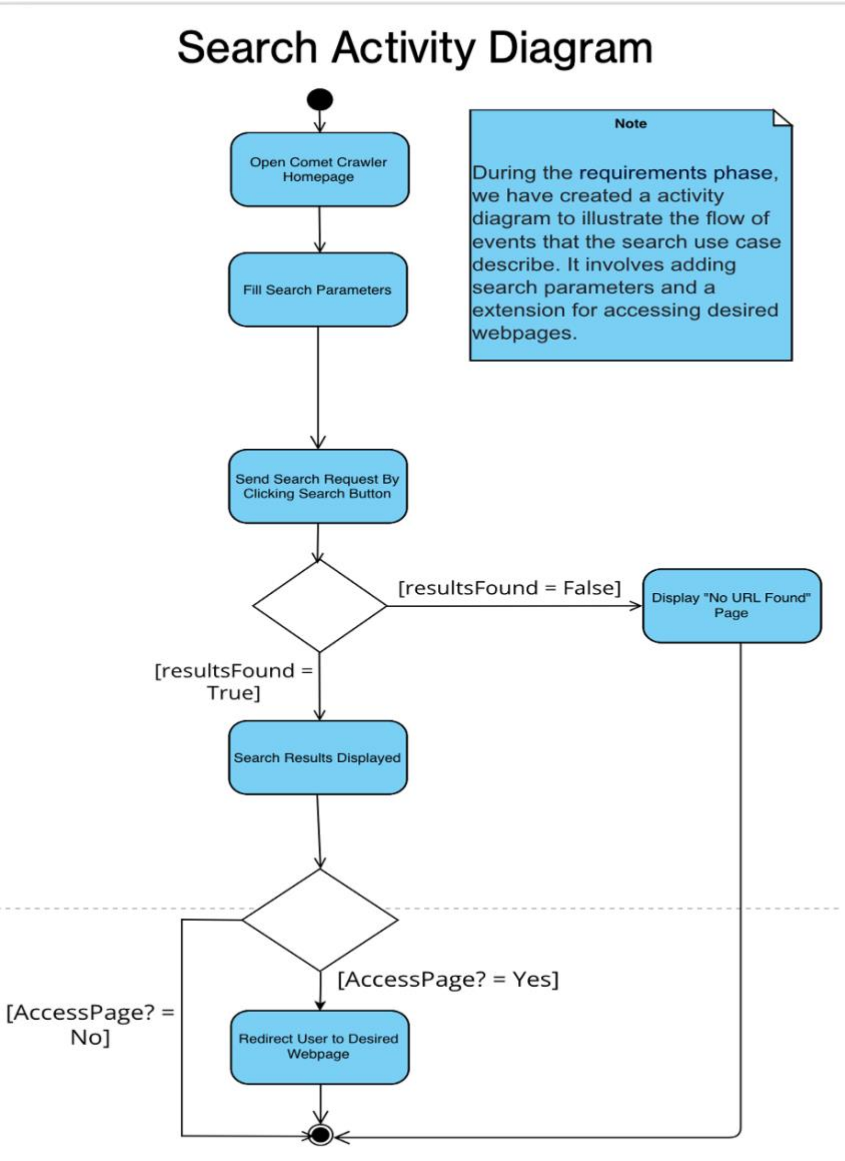
Figure 1.25 Comet Crawler Web Search Engine System’s Design Class Diagram

2. ACTIVITY DIAGRAM:

Definition: Activity diagrams are used to model the flow of activities and actions a system performs.

ACTIVITY DIAGRAM FOR THE COMET CRAWLER WEB SEARCH ENGINE SYSTEM:

2.1



Search Activity Diagram

Note

During the requirements phase, we have created a activity diagram to illustrate the flow of events that the search use case describe. It involves adding search parameters and a extension for accessing desired webpages.

Open Comet Crawler Homepage
Fill Search Parameters
Send Search Request By Clicking Search Button
[resultsFound = False]
Display "No URL Found" Page
[resultsFound = True]
Search Results Displayed
[AccessPage? = Yes]
[AccessPage? = No]
Redirect User to Desired Webpage

Figure 2.1 Comet Crawler Web Search Engine System's Activity diagram demonstrates the actions that are involved in a search.

2.2

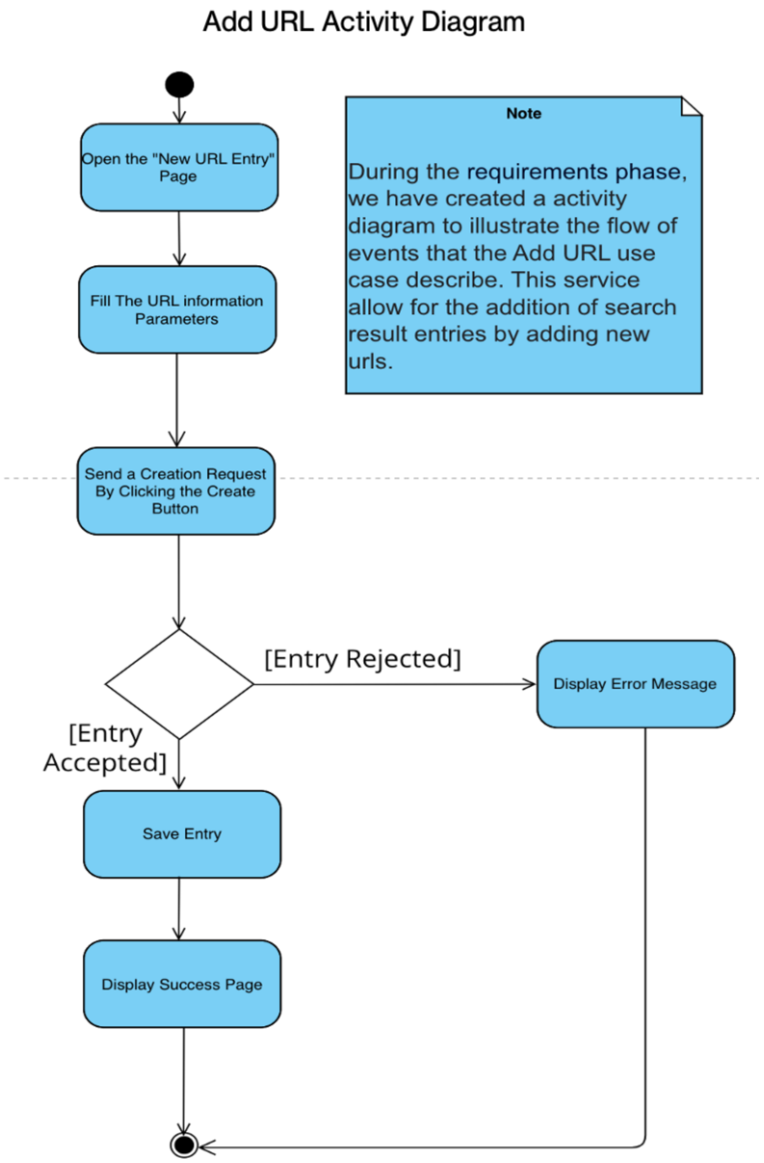

Add URL Activity Diagram
Open the "New URL Entry" Page
Fill The URL information Parameters
Send a Creation Request By Clicking the Create Button
[Entry Rejected]
Display Error Message
[Entry Accepted]
Save Entry
Display Success Page
Note
During the requirements phase, we have created a activity diagram to illustrate the flow of events that the Add URL use case describe. This service allow for the addition of search result entries by adding new urls.

Figure 2.2 Comet Crawler Web Search Engine System's Activity diagram to demonstrate the actions that are involved in adding a URL.

2.3

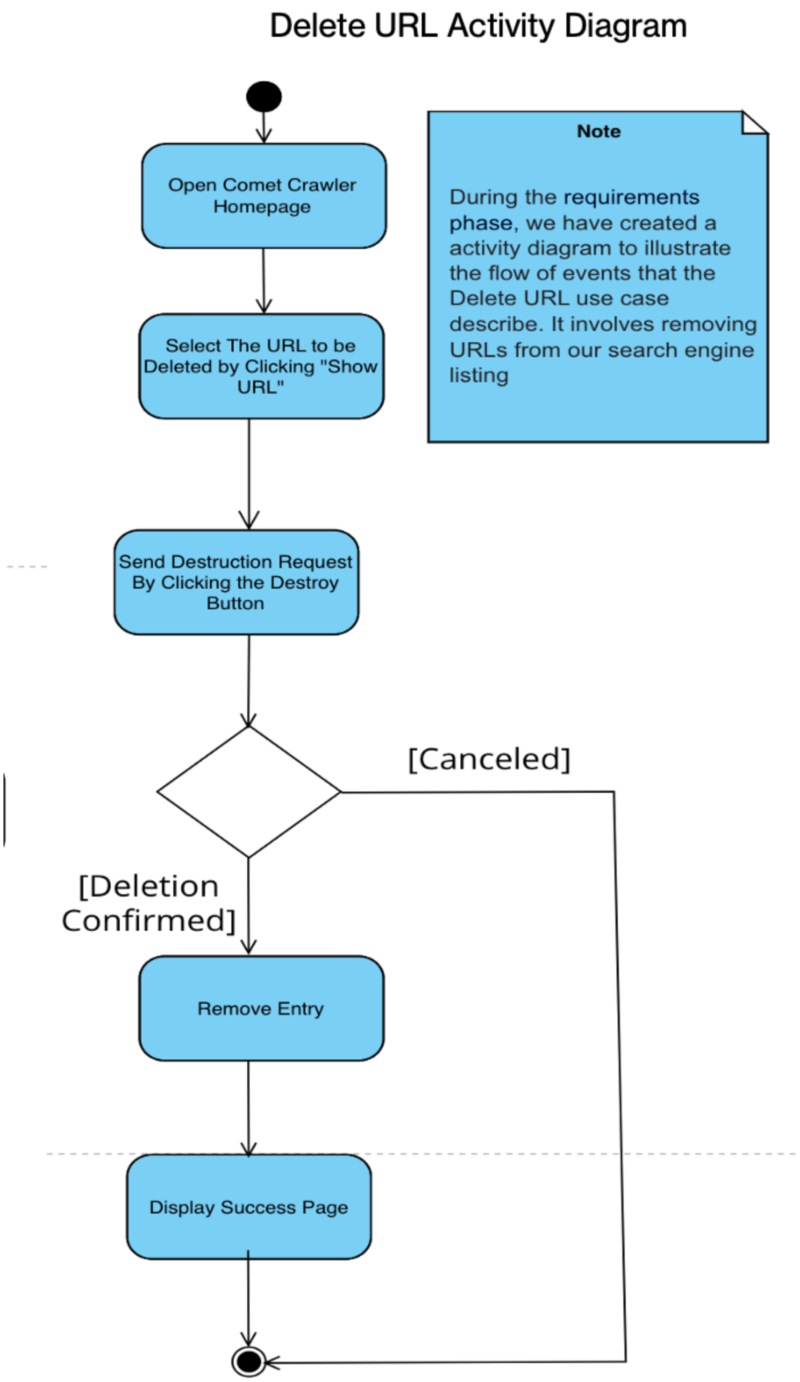
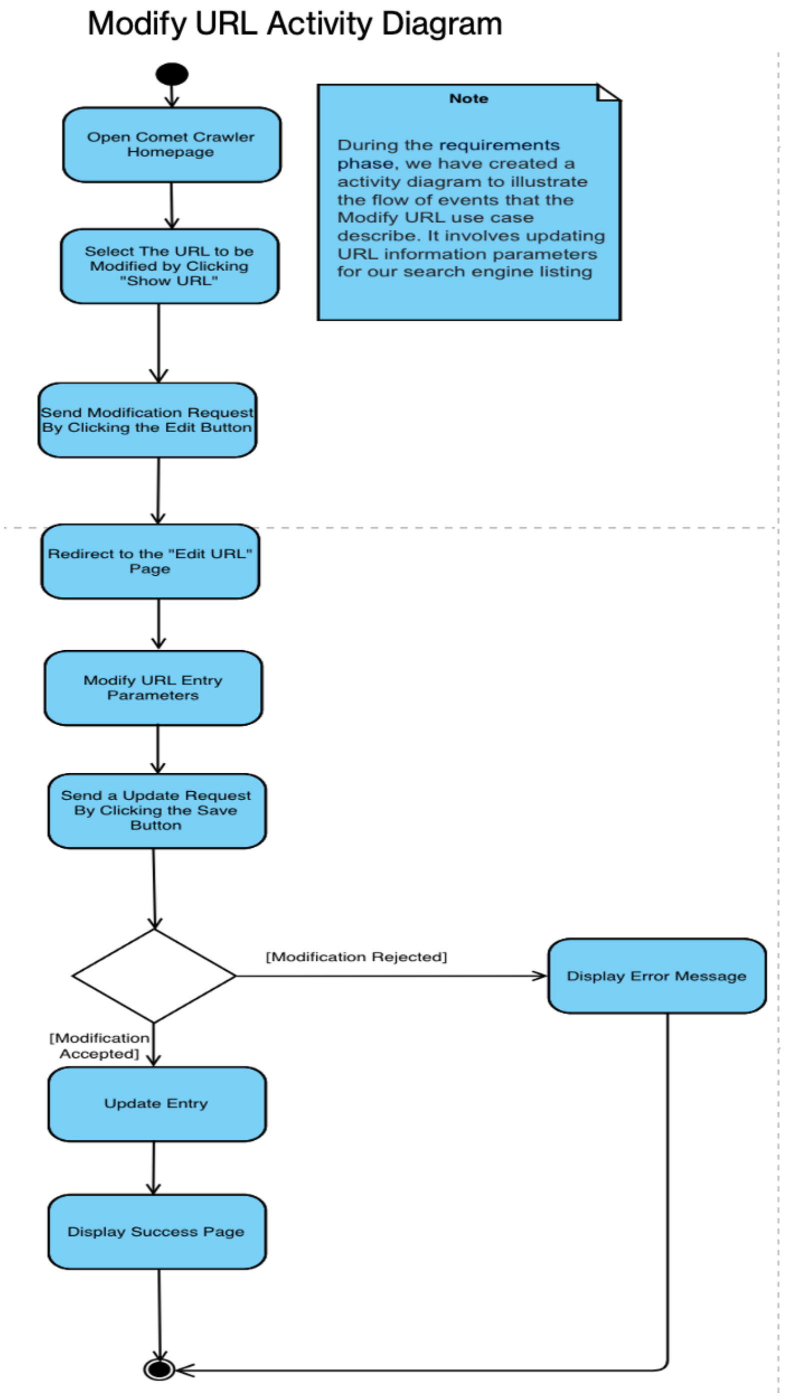

Delete URL Activity Diagram
Note
During the requirements phase, we have created a activity diagram to illustrate the flow of events that the Delete URL use case describe. It involves removing URLs from our search engine listing
Open Comet Crawler Homepage
Select The URL to be Deleted by Clicking "Show URL"
Send Destruction Request By Clicking the Destroy Button
[Canceled]
[Deletion Confirmed]
Remove Entry
Display Success Page

Figure 2.3 Comet Crawler Web Search Engine System's Activity diagram to demonstrate the actions that are involved in deleting a URL.

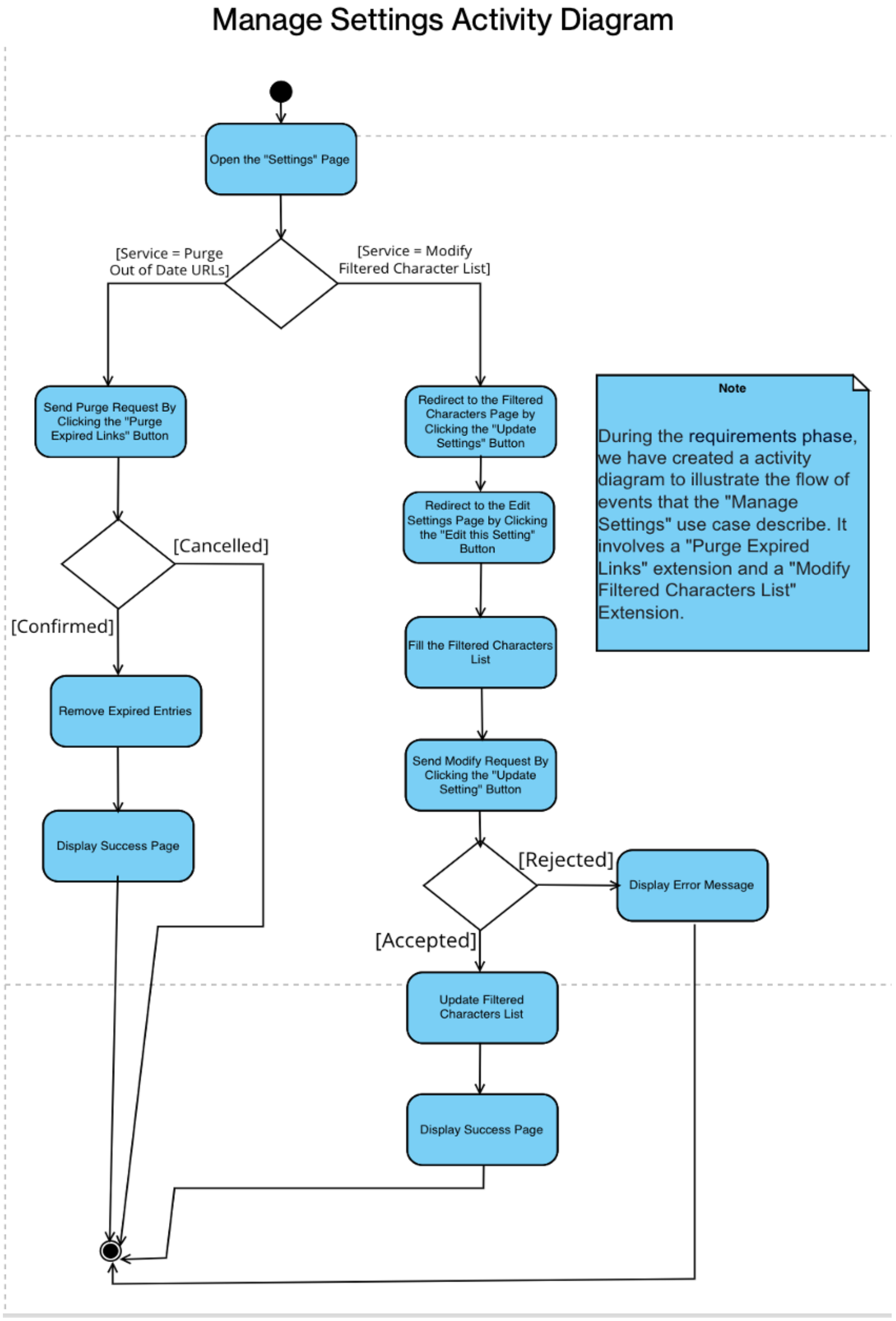
2.4


Modify URL Activity Diagram
Open Comet Crawler Homepage
Select The URL to be Modified by Clicking "Show URL"
Send Modification Request By Clicking the Edit Button
Redirect to the "Edit URL" Page
Modify URL Entry Parameters
Send a Update Request By Clicking the Save Button
[Modification Rejected]
Display Error Message
[Modification Accepted]
Update Entry
Display Success Page
Note

During the requirements phase, we have created a activity diagram to illustrate the flow of events that the Modify URL use case describe. It involves updating URL information parameters for our search engine listing

Figure 2.4 Comet Crawler Web Search Engine System's Activity diagram to demonstrate the actions that are involved in modifying a URL.

2.5


Manage Settings Activity Diagram
Open the "Settings" Page
[Service = Purge Out of Date URLs]
[Service = Modify Filtered Character List]
Send Purge Request By Clicking the "Purge Expired Links" Button
[Cancelled]
[Confirmed]
Remove Expired Entries
Display Success Page
Redirect to the Filtered Characters Page by Clicking the "Update Settings" Button
Redirect to the Edit Settings Page by Clicking the "Edit this Setting" Button
Fill the Filtered Characters List
Send Modify Request By Clicking the "Update Setting" Button
[Rejected]
Display Error Message
[Accepted]
Update Filtered Characters List
Display Success Page
Note

During the requirements phase, we have created a activity diagram to illustrate the flow of events that the "Manage Settings" use case describe. It involves a "Purge Expired Links" extension and a "Modify Filtered Characters List" Extension.

Figure 2.5 Comet Crawler Web Search Engine System’s Activity diagram demonstrates the actions that are involved in managing the system’s settings.

3. STATE DIAGRAM:

3.1 INTRODUCTION:

State diagrams (also called state chart diagrams) model the state transitions a system will experience in its existence. This helps understand the behavioral aspects of the system.

We have two State diagrams: one to show the state transitions the system will go through with search user operations and one to show the state transitions the system will go through with administrator operations.

3.2 STATE DIAGRAMS FOR THE COMET CRAWLER WEB SEARCH ENGINE SYSTEM:

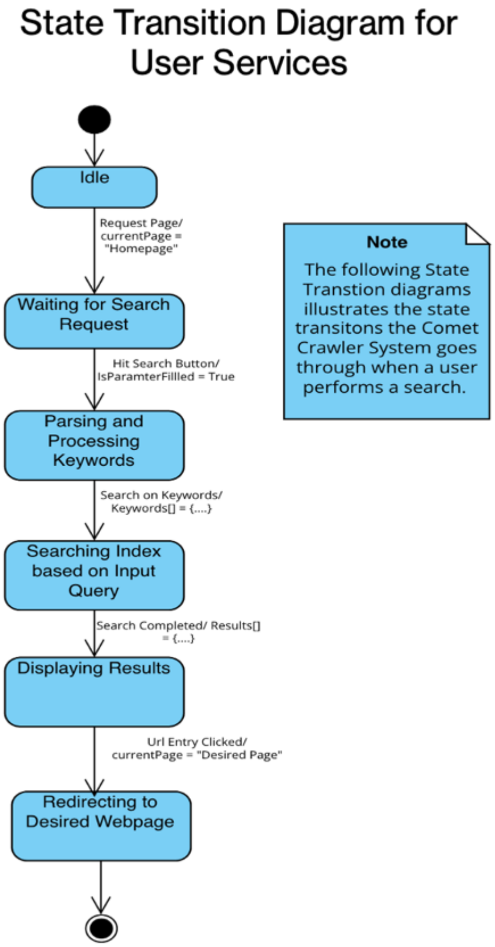
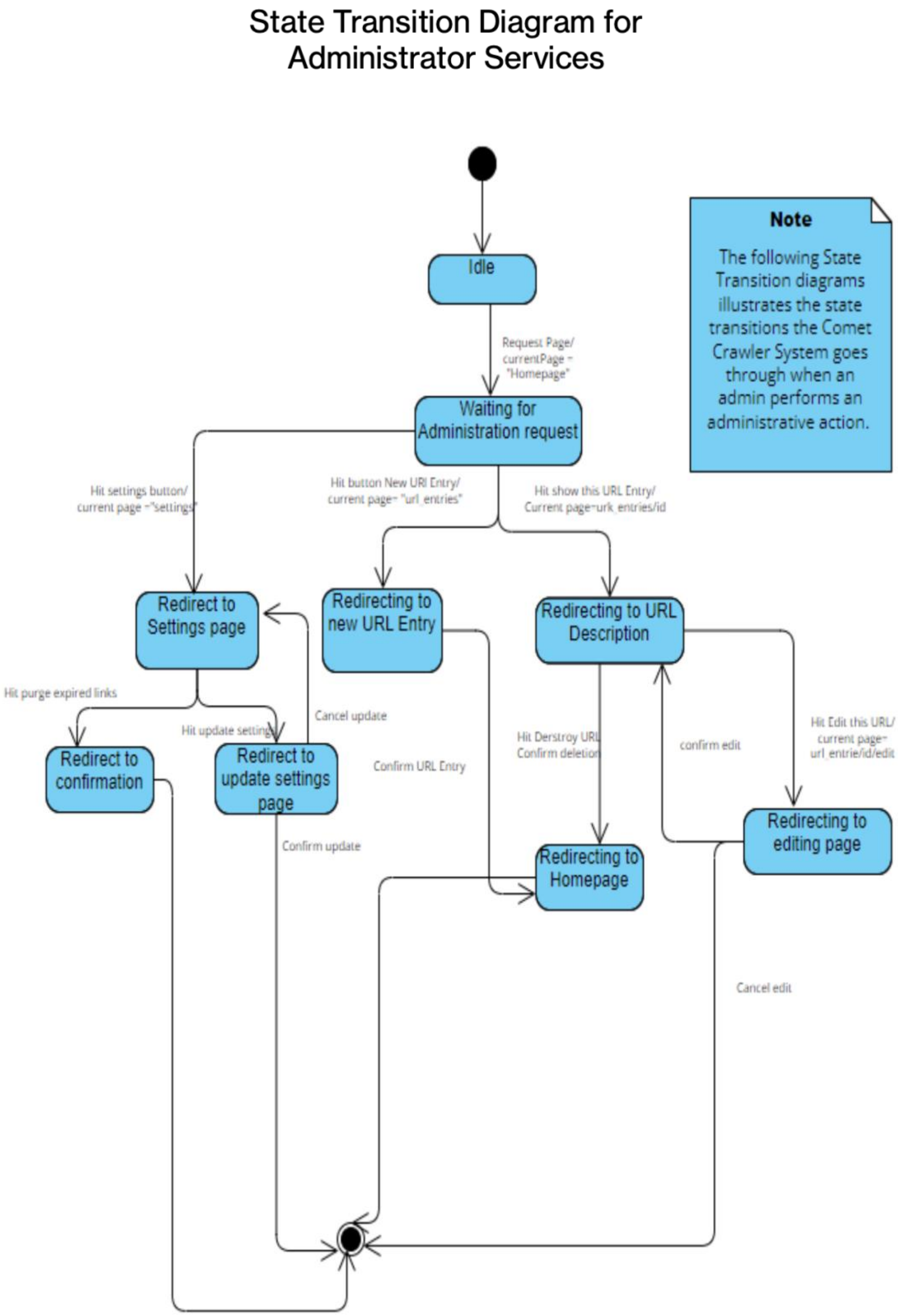

State Transition Diagram for User Services
Idle
Request Page/ currentPage = "Homepage"
Waiting for Search Request
Hit Search Button/ IsParamterFillled = True
Parsing and Processing Keywords
Search on Keywords/ Keywords[] = {....}
Searching Index based on Input Query
Search Completed/ Results[] = {....}
Displaying Results
Url Entry Clicked/ currentPage = "Desired Page"
Redirecting to Desired Webpage
Note
The following State Transtion diagrams illustrates the state transitons the Comet Crawler System goes through when a user performs a search.

Figure 3.1 State Diagram for User Services (Specifically Searching)


State Transition Diagram for Administrator Services
Idle
Request Page/ currentPage = "Homepage"
Waiting for Administration request
Hit settings button/ current page ="settings"
Hit button New URl Entry/ current page= "url_entries"
Hit show this URL Entry/ Current page=urk_entries/id
Redirect to Settings page
Redirecting to new URL Entry
Redirecting to URL Description
Hit purge expired links
Hit update settings
Cancel update
Redirect to confirmation
Redirect to update settings page
Confirm URL Entry
Hit Derstroy URL Confirm deletion
confirm edit
Hit Edit this URL/ current page= url_entrie/id/edit
Confirm update
Redirecting to Homepage
Redirecting to editing page
Cancel edit

Note

The following State Transition diagrams illustrates the state transitions the Comet Crawler System goes through when an admin performs an administrative action.

Figure 3.1 State Diagram for Administrator Services (Add/Delete/Modify/Manage System Settings)

4. PACKAGE DIAGRAM:

Definition: Package diagrams aim to communicate and model a system’s architecture. It visualizes the grouping of components, classes, interfaces, etc., to present a high-level view of the system.

4.1 PACKAGE DIAGRAM FOR THE COMET CRAWLER WEB SEARCH ENGINE SYSTEM:

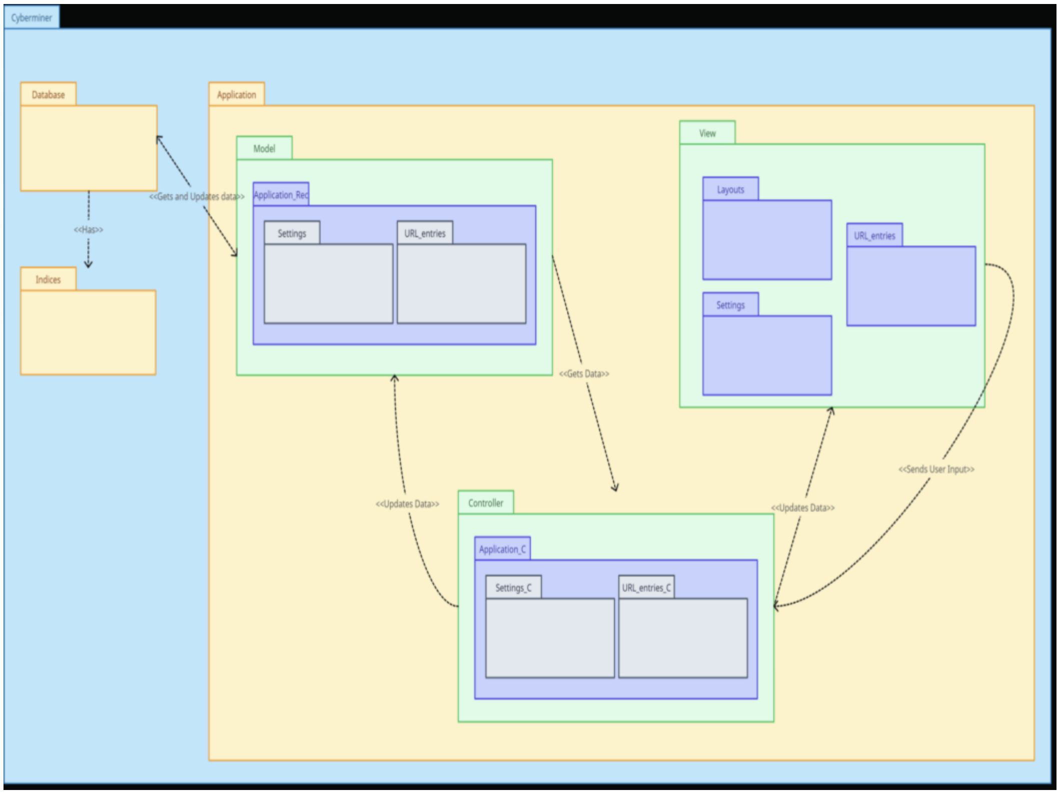


Figure 4.1 Package Diagram for The Comet Crawler System

4.2 DESCRIPTION:

- The System has two major components the search engine (Application Package) and the Index (Indices Package).
- The index and model will interact or relate to the database, as shown in the above diagram.
- The system uses the Model-View-Controller Architectural Pattern (MVC). Therefore, we have 3 packages (Model, View, and Controller)

5. Additional Points:

5.1 List of Additional Points and Links

1. All our test cases can be found here: **https://github.com/tyharg/CS-4376/tree/main/test**
2. The User Manual can be found here: **User Manual**
3. The Updated Preliminary Project Plan can be found here: **Up to Date Preliminary Project Plan**