

**Personalized News Aggregation Website Using Crawlers**

A Report

Submitted to

The School of Engineering and Computing

National University

In Partial Fulfillment of the Requirements

for the Degree of Master of Science in Computer Science

By

Mah Kadidia Konate, Pattira Umyai, Yang Zhang and Atitegeb Gebreselassie

July 2016

**Personalized News Aggregation Website Using Crawlers**

A report submitted to the School of Engineering and Computing, National University, in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science by:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mah Kadidia Konate Pattira Umyai

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Yang Zhang Atitegeb Gebreselassie

The report is hereby approved by:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Pradip Peter Dey, Ph.D.

Professor, School of Engineering and Computing

National University

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Bhaskar Raj Sinha, Ph.D.

Professor, School of Engineering and Computing

National University

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date

TABLE OF CONTENTS

ABSTRACT. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . #

ACKNOWLEDGEMENTS . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

CHAPTER 1: INTRODUCTION . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Project Description and Significance of the Project . . . . . . . . . . . . . . . . . . . .

Definition of Terms . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Literature Review . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Description of Remaining Chapters . . . . . . . . . . . . . . . . . . . . . . . . . . . .

CHAPTER 2: REQUIREMENTS ANALYSIS AND PROJECT PLAN . . . . . . . . . . . . .

Functional Requirements and Specifications . . . . . . . . . . . . . . . . . . . . . . .

Nonfunctional Requirements and Specifications . . . . . . . . . . . . . . . . . . . .

Requirement Feasibility Analysis. . . . . . . . . . . . . . . . . . . .

Use Case Diagrams. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Operational Scenarios . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Project Plan . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

CHAPTER 3: DESIGN. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

High Level Design Architecture . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Design Model (Model Driven SW Development, Model Based SW Development)

UML Diagrams (as applicable) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

State Diagrams, Activity Diagrams, Sequence Diagrams, Class Diagrams,

Component Diagrams, Collaboration Diagrams

Data Flow Diagrams

Transition Graphs

Database Design . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Entity Relationship Diagram (ERD) . . . . . . . . . . . . . . . . . . . . . . .

Entity Relationship Schema . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Test Scenario. . . . . . . . . . . . . . . . . . . .

Graphical User Interface . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

CHAPTER 4: IMPLEMENTATION . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Systems Architecture . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Platforms and Technologies Used . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Languages and Tools . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

User Interfaces with Screen Shots . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

CHAPTER 5: RESULTS AND CONCLUSIONS . . . . . . . . . . . . . . . . . . . . . . . .

CHAPTER 6: FUTURE RESEARCH . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

References . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Appendix A - User's Manual (if applicable) . . . . . . . . . . . . . . . . . . . . . . . .

Appendix B - Maintenance Manual (if applicable) . . . . . . . . . . . . . . . . . . . . . . . .

Appendix C – Additional Design Documents. . . . . . . . . . . . . . . . . . . . . . . . .

Appendix D - Source Code or Sample Source code . . . . . . . . . . . . . . . . . . . . . . . .

Appendix E - Test Suite (if applicable) . . . . . . . . . . . . . . . . . . . . . . . .

Appendix F - Any additional documents . . . . . . . . . . . . . . . . . . . . . . . .

Abstract

A comprehensive summary of the project report in one paragraph, not indented. Describe the report’s purpose and content accurately and concisely, include the most important points, and use specific language and key words that would be used by someone searching for articles in your area of research.

KAPY’s news is a web application which gather news for the users using crawlers. It provides currents news from differents sources and divided into various categories. It recommends news based on the user’s interests. and sent the notifications via registered email address . It allows user interaction by enabling them to make comments on each piece of each

ACKNOWLEDGEMENTS

CHAPTER 1

INTRODUCTION

* 1. Project Description and Significance of the Project

Nowadays, online news reading has become a very prevalent way of obtaining latest news since the web provides access to news articles from millions of sources around the world. Compared with traditional model of news delivery via physical newspaper subscription, news websites changes their content frequently and rapidly.

Instead of browsing various news websites, our project develops a news aggregation website which collecting news from various news websites and providing an aggregate view of news from around the world. In our project, we use web crawlers to extract news from two news websites. After filtering data collected by crawler, the web would display news in different categories for users based on the news type. By default, according to the characteristic of news, news would be presented in the order of time.

However, facing massive data, news consumers need to deal with the massive amount of available news and it is hard for them to find news articles that are interesting to read.

Our website is developed to help user to custom their personal interested news by adding a news recommendation feature. Based on the preference of news fields the user set, the website filters the data stored in the database again to only list news in the fields the user selected. Besides, since each piece of news is collected from a source website, the source name is listed on the page for the news. Members who registered the websites have privileges of subscribing the news source. The notification function would notify the user by email or text message about the latest and top news of the fields selected and also the updated news from his or her subscribed sources in certain frequency.

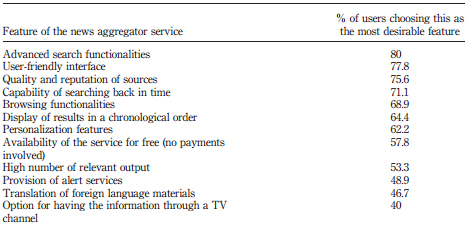
In addition, a new feature is added in each piece of news’s detail page that users could share their personal opinions and comments with each other under each piece of news. Adding like and dislike buttons could collect and simply display the users’ positive or negative attitudes while the comment function provides a chance for every member to read or post their comments under each piece of news.

Besides above features, several other basic websites functions are also implemented in the project. Both guests and members are allowed to browse all the news in the websites. And the search function allows users to search some keywords and display related news for them. The register function is easy for every guest to sign up to become a member. They could set field preferences and notification preferences during the process of registering or access the member profile to reset these options after becoming a member. The login function is same for both members and administrators, but the system would distinguish the administrator by the user names automatically. The map function will allow the user to click one location (such as a state) and get the local news of the particular location.

* 1. Definition of Terms
  2. Literature Review
* Problem statement: Aggregation of news websites using crawlers

We are proposing a novel kind of news aggregator website that uses the web crawler innovation. It will allow a user to get in one place all the news tailored to his/her predefined categories, and features the capacity to provide notification to the user at the frequency he/she desires. The system we are proposing will also enhance user interaction. It will implement capabilities such as liking and commenting on each piece of news.

Chowdhury and Landoni[[1]](#footnote-1), conducted a survey to find out what users expected from news aggregator services and how the available services compare to each other . Their findings reveal that the features that users expect in news aggregator services are as follow, in the order of desirability:



In the system we are building we aim to provide the following services:

* The webpages’ content polling using crawlers
* A user friendly interface to get a consolidated view of the content of different news websites
* The notification system
* The personalization of news to suit user’s preferences
* User user interaction.
* Background: The origin of information retrieval systems

The idea of using computers to search for relevant pieces of information was popularized in the article *As We May Think* by Vannevar Bush in 1945[[2]](#footnote-2). This visionary essay has been very influential for the modern information retrieval systems.

The World wide web has a wealth of information. The growing number of news websites resulted in the sharp rise of news consumption over the past years[[3]](#footnote-3). In fact nearly 4 in 10 americans often get news from digital sources (including websites, apps and social media) [[4]](#footnote-4). The proliferation of news websites incur cost for news consumers such as the time and effort spent browsing different news websites and filtering the news to get the most relevant to their preferences.

* Real time information processing

As digital sources change their content regularly, it is necessary to develop technologies that are able to detect the change automatically and inform the news consumer continuously. Currently Search engines provide the capability to search the internet for the most relevant websites[[5]](#footnote-5) but don’t allow for understanding websites content’s periodical change. Website change detection tools include web crawler, Rich Site Summary (RSS) feed and some websites’s proprietary monitoring and notification services.

* 1. Description of Remaining Chapters

CHAPTER 2

REQUIREMENTS ANALYSIS AND PROJECT PLAN

Kapy Personalized News system is a news website that allows users to view different news articles from different news sources tailored to their preferences. It will be deployed as a web application on the cloud.  Kapy must abide by the copyright laws on accessing and displaying data from news sources.

* 1. Functional Requirements and Specifications

1. Basic functions of websites

Kapy Personalized News system is a website that provides basic functions as websites. The system must be able to give a service for all users, both guests and members with different privileges. The sign up function is implemented for users who would like to register for a membership with more privileges. And the log in and log out functions are used for members to access accounts and sign out accounts. Members of the websites are also able to renew a password if they forgot the current one. The system also has administrators who have authorities of managing all users by deleting or suspending users.

1. News functions

Since the website is used to provide news for users, the collection and display of news are fundamental functions of our system. The system should be able to collect news from two or more news websites using web crawlers in certain frequency and post the news by order of timeline as default. The news is displayed separate in different categories for the user’s convenience of browsing. For example; local news, social news, sport news, and so on. The top news category will be shown at the first page as default and is ranked by the page views of each piece of news. The search function should allow all users to search using keywords and display all news containing the keywords in the order of time. The system also provides a map function which allows the users to click on the interested state and get the local news in the homepage.

1. News personalization function for members

Kapy Personalized News system provides news personalization services for members. All members are able to define the fields of news they are interested in when they sign up an account. They could also update their news preferences any time in their own profile pages when they sign in to the system. News in the fields the member defined would be collected together in the recommender category for the member. Besides news personalization, the system also provides UI personalization for all members. They are able manipulate their personal User Interface (UI) of the application, such as changing background colors of website.

1. Comment functions for members

Kapy Personalized News system also provides services that all members are convenient to share opinions of each piece of news. Members could click like or dislike buttons to simply express their positive or negative attitudes for each piece of news. They could also write down and post their comments about each piece of news. Besides, for each comment, if the member likes or agrees with the opinion, they could click the like button besides. The system would track the number of likes of the comments and displays the top 3 comments under the news.

1. Notification functions for members

The News website offers all members privileges to receive notifications for news. Members could set the email address and also the frequencies that would like to receive notifications during the process of sign up. And they can also modify the settings any time when they login. The member should be able to subscribe news sources for accepting notifications of updated news from those sources. The system would record all the settings and notify the member via email about the updated news from fields he or she selected or from subscribed sources.

* 1. Nonfunctional Requirements and Specifications

1. A user who knows how to use any web application should be able use the Kapy News. No user manual is important.
2. The Google News user interface standards used in Kapy News should be familiar for the user.
3. Kapy should be able to open up and display the home page within 3 seconds after the user enters the url kapyNews.azurewebsites.net
4. Kapy should gather and store information every three hours.
5. The modified profile should be updated for the user within 2 seconds
6. Guests should register to use the personalize feature of Kapy for security purposes.
7. Kapy’s strong password should help ensure that guests don’t masquerade as another user.
8. Kapy should send reset password to the user with in 1 minute.
9. The system should interact with external websites every three hours.
10. The administrator should manage the running system.
11. The system should differentiate between a member and an administrator.
    1. Requirement Feasibility

We use MoSCoW method for the analysis of feasibility of features. According to Clegg D., MoSCoW is an acronym derived from the first letter of each prioritization categories.

Must have- are critical to the current three month project.

Should have- are important but not necessary for the delivery within the three month project.

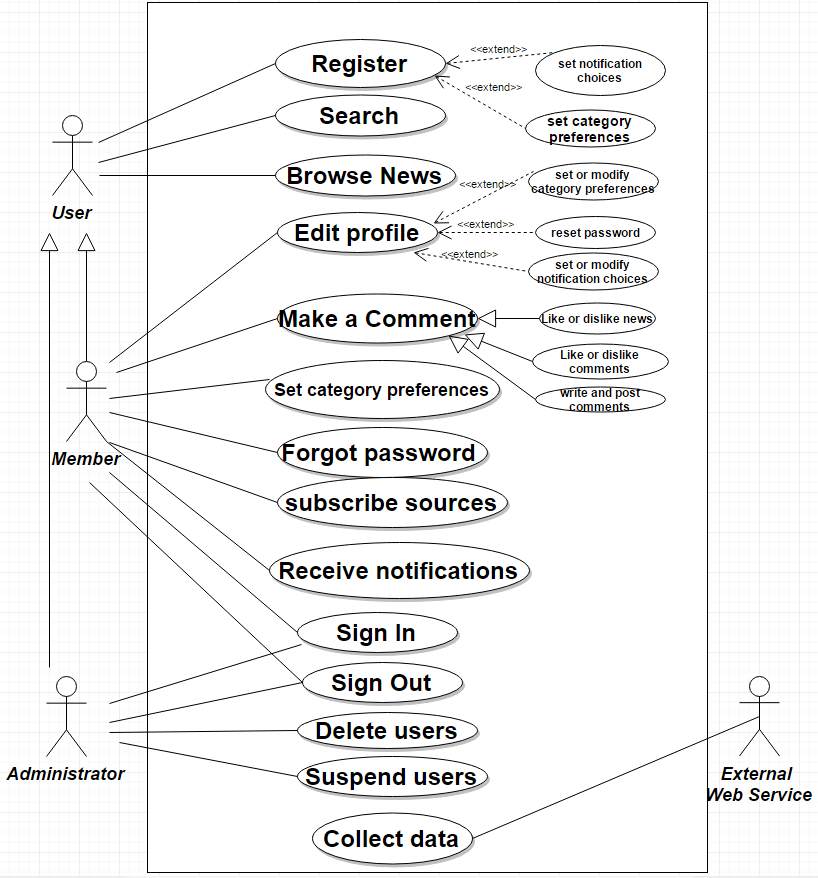
Could have- are desirable but not necessary and could improve user experience or customer satisfaction.

Won’t have-are least critical lowest payback items or not appropriate at the end of three month project.

Following is the table of feature feasibility analysis.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Features | Description | Actors | Tools /techniques | Constraints | Feasibility |
| Crawling news | Collecting news from two or more news websites using crawlers and post the news in order of time as default. | External web services | Scrapy Crawler, Python | Copyright,  Some websites block crawlers | MUST HAVE |
| Set Category preferences | Define different categories, like local, social, sports and so on for personalized references page for the user | Member | SQL, .NET MVC |  | MUST HAVE |
| Top News | Define the top news category which depends on the number of likes. | Member | SQL, .NET MVC |  | SHOULD HAVE |
| The search function | The search function which allows the user to search some keyword. | Guest/member | SQL, .NET MVC |  | SHOULD HAVE |
| Register function | Guests could sign up an account to have member privileges | Guest | SQL, .NET MVC |  | MUST HAVE |
| Login function | Members could login to access more functions than guests;  Administrator needs login manage users’ accounts | Member, Administrator | SQL, .NET MVC |  | MUST HAVE |
| Make comments | Members are allowed to like a piece of news.  Members are allowed to write and post a comment in the news details page and also like or dislike comments of others. | Member, | SQL, .NET MVC |  | SHOULD HAVE |
| Subscribe news sources | Members are allowed to subscribe a news source in the websites. Notifications are sent to the member about the updated news from sources he/she subscribed. | Member | SQL, .NET MVC |  | SHOULD HAVE |
| Notification function | The system send updated news to the members who subscribed the news source or had preference on the news field via email | Member | SQL, .NET MVC, Crawler |  | SHOULD HAVE |
| The map function | The system could provide a map function which allows the users to click on the interested state and get the local news. | Users | SQL, .NET MVC, Map API |  | COULD HAVE |

* 1. Use Case Diagrams



* 1. Use Case Specifications

*UC-001: Sign Up Use Case*

For this use case, the user registers to be a member of the website. A new account is created with a unique member ID.

|  |  |
| --- | --- |
| ID | UC-001 |
| Related UCs | UC-002 : Select interested news categories |
| Description | A new user can register for a new account in the system |
| Actors | User, Database |
| Preconditions | The user is not a member of the website |
| Post conditions | A new account is created and the user sign in |
| Basic Flow | * This flow starts when the user wants to register to be a member * The user enter a username, password, email, and cellphone number * The system checks if the username is valid and not exists in the database * The system checks if the password is valid * The user is asked to select interested news topics * A new account is created and assigned to a unique user ID number * The user is automatically signed in after a successful signup * Here ends the flow |
| Alternative Flow | * Request information does not pass a verification check * A message is shown and the user can try again or cancel * The use case repeats |

*UC-002: Select Interested News Categories Use Case*

The user could set the categories preferences in three different ways. Firstly, they could set preferences when they register accounts. Secondly, they could access their profiles and reset category preferences any time after login the system. Thirdly, after the users login the system and never set any category preferences, the website pops up a set window for him or her. For this use case, the user is asked to select interested news fields that he/she wants to subscribe during the register process.

|  |  |
| --- | --- |
| ID | UC-002 |
| Related UCs | UC-001 : Sign Up Use Case |
| Description | A new user can select interested news categories during Sign Up process |
| Actors | User |
| Preconditions | The user is in the process of register for a membership |
| Post conditions | A new account is created and memorized personal interesting fields of news for the user |
| Basic Flow | * This flow starts when the user register to be a member * The user’s personal info has been validated * The user is asked to select interested news topics * A new account is created and the interested news topics are memorized by the system * The user is automatically signed in after a successful signup * Here ends the flow |
| Alternative Flow | * The user can make changes on the news category preferences in his or her profile anytime he/she signs in |

*UC-003: Log in Use Case*

Both members and administrators could log in the websites in the same login page. For this use case, a member can log in to the system and has privileges to access all of the provided functions. Besides, an administrator can login the system too.

|  |  |
| --- | --- |
| ID | UC-003 |
| Related UCs | UC-004: Forgot Password |
| Description | A member or administrator log in to the system and the system determine the identity by the username. |
| Actors | Member, Administrator |
| Preconditions | The usernames for administrator are already stored in the system |
| Post conditions | A member can access into a member’s privileges.  An administrator can access into the administrator’s privileges. |
| Basic Flow | * This flow starts when the user clicks into the log in page. * The user enters a username and password * The system checks if the username and password are match with database * The system determines the identity of the user according the username entered. * The system checks the database of the member about the interested category preferences * The system filters the news in the specific selected fields. * The system updates and lists the news after filtered in the recommend category of the website for the member. * The system directs the user to the homepage. * Here ends the flow |
| Alternative Flow | * If the username and password are not matched, the system will show a message * The user has choice to try again, or cancel, or choose “Forgot Password” |
| Alternative Flow | * If the username entered is the name set for administrator, the identity determined is administrator. * The system directs the administrator to the user management page. |
| Alternative Flow | * The system determines the database has no information about the interested fields of the member logged in. * A fields preferences selection window popped up to asks the member to select fields that they’re interested from the default candidates * The member clicks fields they’re interested and the submit button. * The system stores the information about the interested fields in the member’s profile. |
| Alternative Flow | * The member ignores the field’s preferences window. * The system lists the news from the default fields (local, top, latest news) in the recommend category. |

*UC-004: Forgot a Password Use Case*

For this use case, the member is able to reset a new password to login to the system

|  |  |
| --- | --- |
| ID | UC-004 |
| Related UCs | UC-003 : Log In |
| Description | the member is able to reset a new password to login to the system |
| Actors | Member |
| Preconditions | The database has stored information about the member |
| Post conditions | The member has a new password to log in to the website.  The database stored the updated password for the member. |
| Basic Flow | * This flow starts when the member enters the log in page of the system but he/she forgot the password * The member clicks “Forgot a Password” * The member enters either email address or username used for register. * The system checks the database of the member’s records of email address or username. * The information the member enters matches. * The system generates a link for the user to reset a new password and emails to him or her. * The system reminds user to check the email about the link for resetting password. * The user clicks the reset link from email. * The user enters new password and password confirmation. * The system verifies the link and updated the new password for the member. * The system directs the user to the log in page. * Here ends the flow |
| Alternative Flow | * The system has no records about the username or email address the member enters. * The system reminds the member “no records. Try again” |

*UC-005: Search use case*

For this use case, the user could search some keywords in the search box and the system returns a page lists all news containing the given keyword.

|  |  |
| --- | --- |
| ID | UC-005 |
| Related UCs |  |
| Description | The user searches for news containing a given keyword |
| Actors | User |
| Preconditions |  |
| Post conditions | The list of news containing the keyword is displayed in a relevant order for the user |
| Basic Flow | * This flow starts when the user wants to search for a given keyword * The user types the keyword in the search Text box * The user clicks on the Search button * The system search the keywords in database, and returns news containing the keywords in the news * The list of relevant news title which contains the keyword is displayed in the page. * The user selects one piece of news that he/she wants to view * The program asks the user his/her preferred display mode: either in the current webpage or in a new tab * The user choses to open the selected webpage in a new tab * Here ends the flow |
| Alternative Flow | * The user choses to open the selected webpage in the current open page |

*UC-006: Subscribe Use Case*

For this use case, the user could subscribe a website source of news and could get notifications from the sources he or she subscribed.

|  |  |
| --- | --- |
| ID | UC-006 |
| Related UCs | UC-003 : Log In; UC-009: receive notification |
| Description | The member subscribes the source they want to receive notifications from. |
| Actors | Members |
| Preconditions | The user logged in the system |
| Post conditions | The database store the sources the user subscribed and the “subscribe” button for the same resource should be unclicked. |
| Basic Flow | * The flow starts when a user clicks one piece of news and goes to the page for the news. * The member clicks “subscribe” button under the source name listed beside the content of news. * The system gets the source name of the user subscribed and stores in the database. * The system updates the “subscribe” buttons for the same source is not clickable. * The system updates the sources for notification of the member. |
| Alternative Flow | * If the “subscribe” button under the source name in the news page is not clickable, the member could not subscribe again. |

*UC-007: Make Comment ---like news use case*

For this use case the user option to like the news. The number of likes will be updated every time the member clicks on this event.

|  |  |
| --- | --- |
| ID | UC-007 |
| Related UCs | None |
| Description | The member likes one piece of news |
| Actors | Member |
| Preconditions | The member signed in and is active. |
| Post conditions | The total number of likes is increased by one and is updated in the database |
| Basic Flow | * This flow starts when the member enters the page of one piece of news  and wants to make a comment on the news * A “like” thumb up button is displayed on the news that the member is viewing * The member clicks on the button * The button is flagged as “liked” * The number of likes for that news is increased by one in the database * The updated total number of likes of the news is displayed for all users * Here ends the flow |
| Alternative Flow | None |

*UC-008: Write and post comments use case*

For this use case the user write and post a comment for one piece of news.

|  |  |
| --- | --- |
| ID | UC-008 |
| Related UCs | None |
| Description | The member write and post his/her comment on one piece of news |
| Actors | Member |
| Preconditions | The member signed in and is active |
| Post conditions | The content of comment is stored in the database |
| Basic Flow | * This flow starts when the member enters the page of one piece of news  and wants to make a comment on the news * The member writes down his/her comments in the textbox under the news content. * The member clicks submit button. * The system checks the content of comment. * The stores the comments and displays under the news. * Here ends the flow. |
| Alternative Flow | * The content of the comment does not pass the criteria set for comments. * A reminder for the member that the comment is not qualified. |

* 1. Operational Scenarios

This is an operational scenario for the user searching for a given keyword.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario # 1 | User’s Objective: to search for a keyword and the program should display the list of news that contain this keyword with a ranking\* | | |
| Scenario Objective: | | | |
| Source | Step | Action | Comments |
| user | 1 | Opens the program |  |
| program | 2 | Displays the home page |  |
| user | 3 | Types a keyword ex: “Key” in the search box on the webpage |  |
| program | 4 | displays the list of 10 news websites that contain the keyword “Key” that are the most relevant\* to the user and the remaining will be hidden by a button “Show more” |  |
| user | 5 | the program displays the remaining results of the search query | user clicks on the “Show more” button |
| user | 6 | selects a web page among the results displayed |  |
| program | 7 | asks the user if he/she wants to open the webpage selected in a new a new tab or in place |  |
| program | 8 | the program opens the web page selected |  |

This is an operational scenario for the user making a comment on specific news.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario # 2 | User’s Objective: to make a comment on a news | | |
| Scenario Objective: Evaluate the user and the program for proper comment handling | | | |
| Source | Step | Action | Comments |
| user | 1 | Open the program |  |
| program | 2 | Display the home page |  |
| user | 3 | Click on a specific news headline |  |
| program | 4 | Display the detail news |  |
| user | 5 | Click on the like button | User is a guest |
| program | 6 | Display Message “Please sign up before making a comment” |  |
| user | 7 | Request to sign up |  |
| program | 8 | Store user name and password on the database |  |
| User | 9 | Sign in with the new user name and password | User is Member |
| program | 10 | Allow the user to make comment |  |
| User | 11 | Click on the like button |  |
| program | 12 | Update the number of likes in the database |  |

This is an operational scenario to implement the function for users to set category they like as recommendation category.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario # 3 | User’s Objective: to set the interested fields preferences in the system and list news in the recommend category from the fields they already set. | | |
| Scenario Objective: to identify program response to different user actions | | | |
| Source | Step | Action | Comments |
| user | 1 | Open the program |  |
| program | 2 | Display the home page | In the recommend category just list default fields news |
| user | 3 | Login the system | User is a member |
| program | 4 | access the user’s profile in database |  |
| program | 5 | Determine the user’s interested fields are empty or not | If not empty, goes to step 12, else goes to next step |
| program | 6 | No record in the user’s interested fields in the database |  |
| program | 7 | Pop up a fields selection window that ask user to select interested fields from candidate fields | The candidate fields are set already. |
| user | 8 | Select some fields from the window |  |
| user | 9 | Click “select” button |  |
| program | 11 | Store the preferred categories  the user set in the user’s profile in database |  |
| program | 12 | Filter the news in the specific fields according to the user’s interested fields stored in database |  |
| program | 13 | Show the home page. Display the news after filtering in the recommend category |  |

In step 3, it relates the login user case in details (log in verification and so on). The user can also visit the program as a guest, but the recommend function won’t work for guests (just list news in default fields).

In step 5, if there is record of the user’s interested fields, go directly to step 11.

In step 9, if the user does not select any fields and clicks “select” button, an error reminder to the user. If the user clicks “cancel” button, jump following steps. Next time, he or she login, the selection window pops up again.

Else, if the user wants to modify the interested fields, he or she could also access his or her profile and edit the interested fields.

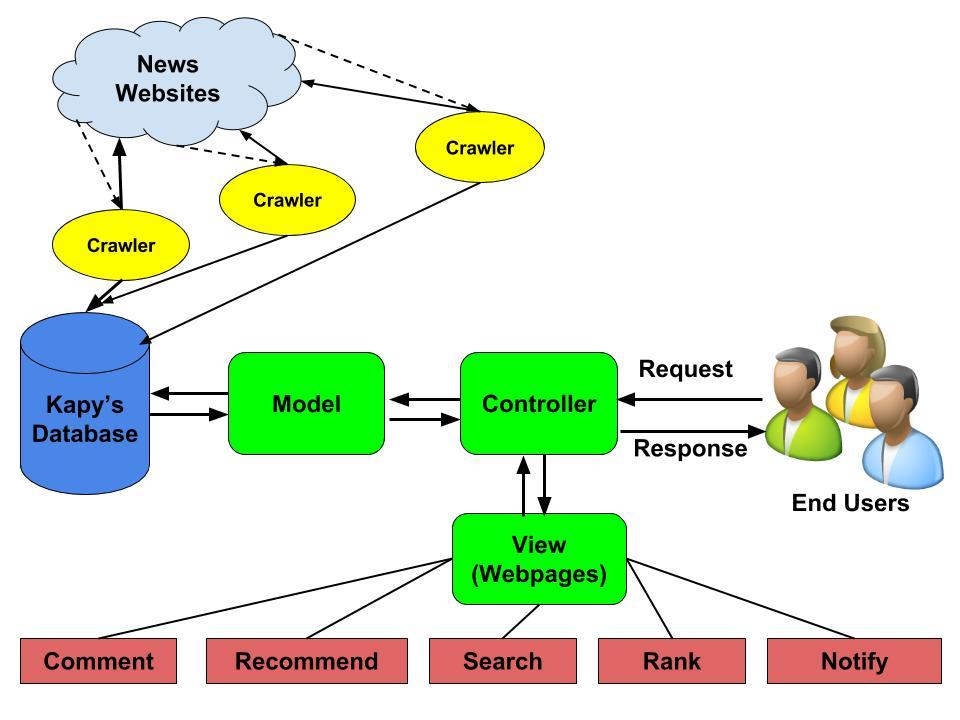
* 1. Project Plan

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| KAPY personalized news | | | |  |  |  |  |  |
|  | Project Start Date: | | | 7/5/2016 (Tuesday) | |  |  |  |
|  | Display Week: | | | 1 |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| **WBS** | **Task** | **Lead** | Prede cessor | **Start** | **End** | Cal. Days | **% Done** | **Work Days** |
| **1** | **Project description** |  |  |  |  |  |  |  |
| 1.1 | Project Title | Ati |  | Tue 7/05/16 | Fri 7/08/16 | 4 | 100% | 4 |
| 1.2 | Project description | Yang |  | Tue 7/05/16 | Fri 7/08/16 | 4 | 100% | 4 |
| 1.3 | References |  |  | Tue 7/05/16 | Fri 7/08/16 | 4 | 100% | 4 |
|  |  |  |  |  |  |  |  |  |
| **2** | **Project Proposal** |  |  |  |  |  |  |  |
| 2.1 | Project presentation | All members K-A-P-Y |  | Tue 7/12/16 | Tue 7/12/16 | 1 | 100% | 1 |
| 2.2 | Oral presentation | All members K-A-P-Y |  | Tue 7/12/16 | Tue 7/12/16 | 1 | 100% | 1 |
| 2.3 | Report |  |  | Tue 7/12/16 | Fri 7/15/16 | 4 | 100% | 4 |
| 2.4 | Use case diagram | Pattira |  | Tue 7/12/16 | Fri 7/15/16 | 4 | 100% | 4 |
| 2.5 | Activity diagram | Pattira |  | Tue 7/12/16 | Fri 7/15/16 | 4 | 100% | 4 |
| 2.6 | Requirements analysis | All members |  | Tue 7/12/16 | Tue 7/12/16 |  | 100% | 1 |
| 2.7 | Set up Azure | All members |  | Thu 7/14/16 | Tue 7/19/16 | 6 | 100% | 4 |
|  | UI design on Visual Studio KAPY.V0 | Ati |  | Tue 7/12/16 | Thu 7/14/16 | 3 | 100% | 3 |
|  | Send assignment to professor | Yang |  | Fri 7/15/16 | Fri 7/15/16 | 1 | 100% | 1 |
|  |  |  |  |  |  |  |  |  |
| **3** | **Database design** |  |  |  |  |  |  |  |
| 3.1 | Define database model - ERD diagram | Kadidia |  | Tue 7/19/16 | Fri 7/22/16 | 4 | 0% | 4 |
| 3.2 | Test Specification/Test Plan | All members |  | Tue 7/19/16 | Thu 7/21/16 | 3 | 0% | 3 |
| 3.3 |  |  |  | Wed 7/20/16 | Wed 7/20/16 | 1 | 0% | 1 |
| 3.4 | Send assignment to professor | Yang |  | Thu 7/21/16 | Fri 7/22/16 | 2 | 0% | 2 |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| **4** | **Prototype - Key elements** |  |  |  |  |  |  |  |
| 4.1 | Website implementation - V1 |  |  | Tue 7/26/16 | Mon 8/01/16 | 7 | 0% | 5 |
| 4.2 | Unit Test1 | Ati |  | Sat 7/23/16 | Thu 7/28/16 | 6 | 0% | 4 |
| 4.3 | Unit Test2 | Yang |  | Sat 7/23/16 | Thu 7/28/16 | 6 | 0% | 4 |
| 4.4 | Unit Test3 | Pattira |  | Sat 7/23/16 | Thu 7/28/16 | 6 |  |  |
| 4.5 | Unit Test4 | Yang |  | Sat 7/23/16 | Thu 7/28/16 | 6 |  |  |
| 4.6 | System Integration testing design | All members |  | Tue 7/26/16 | Thu 7/28/16 | 3 |  |  |
| 4.7 | Project Report: All above and references | All members |  | Tue 7/26/16 | Thu 7/28/16 | 3 |  |  |
| 4.8 | Project Defense: Presentation and Review | All members |  | Thu 7/28/16 | Thu 7/28/16 | 1 |  |  |
| 4.9 | Send assignment to professor | Yang |  | Fri 7/29/16 | Fri 7/29/16 | 1 | 0% | 1 |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| **5** | **Design, Design Review (Discussion), Prototype** |  |  |  |  |  |  |  |
| 5.1 | Powerpoint |  |  | Tue 8/09/16 | Thu 8/11/16 | 3 | 0% | 3 |
| 5.2 | Report |  |  | Tue 8/09/16 | Thu 8/11/16 | 3 | 0% | 3 |
| 5.3 | Prototype |  |  | Tue 8/09/16 | Thu 8/11/16 | 3 | 0% |  |
| 5.4 | Unit Test1 | Ati |  | Sat 8/06/16 | Thu 8/11/16 | 6 | 0% |  |
| 5.5 | Unit Test2 | Yang |  | Sat 8/06/16 | Thu 8/11/16 | 6 | 0% |  |
| 5.6 | Unit Test3 | Pattira |  | Sat 8/06/16 | Thu 8/11/16 | 6 | 0% |  |
| 5.7 | Unit Test4 | Yang |  | Sat 8/06/16 | Thu 8/11/16 | 6 | 0% |  |
| 5.8 | System Integration testing Test1 | All members |  | Tue 8/09/16 | Thu 8/11/16 | 3 | 0% |  |
| 5.9 | Project Defense: Presentation and Review | All members |  | Thu 8/11/16 | Thu 8/11/16 | 1 | 0% |  |
| 5.10 | Send assignment to professor | Yang |  | Fri 8/12/16 | Fri 8/12/16 | 1 | 0% | 1 |
| 5.11 | Project Report: All above and references | All members |  | Fri 8/12/16 | Fri 8/12/16 | 1 | 0% |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| **6** | **Design, Design Review (Discussion), Prototype** |  |  |  |  |  |  |  |
| 6.1 | Powerpoint |  |  | Tue 8/16/16 | Thu 8/18/16 | 3 | 0% | 3 |
| 6.2 | Report |  |  | Tue 8/16/16 | Thu 8/18/16 | 3 | 0% | 3 |
| 6.3 | Prototype |  |  | Tue 8/16/16 | Thu 8/18/16 | 3 | 0% |  |
| 6.4 | Unit Test1 | Ati |  | Sat 8/13/16 | Thu 8/18/16 | 6 | 0% |  |
| 6.5 | Unit Test2 | Yang |  | Sat 8/13/16 | Thu 8/18/16 | 6 | 0% |  |
| 6.6 | Unit Test3 | Pattira |  | Sat 8/13/16 | Thu 8/18/16 | 6 | 0% |  |
| 6.7 | Unit Test4 | Yang |  | Sat 8/13/16 | Thu 8/18/16 | 6 | 0% |  |
| 6.8 | Oral Presentation | All members |  | Thu 8/18/16 | Thu 8/18/16 | 1 | 0% | 1 |
| 6.9 | System Integration testing Test1 | All members |  | Tue 8/16/16 | Thu 8/18/16 | 3 | 0% |  |
| 6.10 | Project Report: All above and references | All members |  | Tue 8/16/16 | Thu 8/18/16 | 3 | 0% |  |
| 6.11 | Send assignment to professor | Yang |  | Fri 8/19/16 | Fri 8/19/16 | 1 | 0% | 1 |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| **7** | **Prototype Demonstrations/Reviews** |  |  |  |  |  |  |  |
| 7.1 | Powerpoint |  |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% | 3 |
| 7.2 | Report |  |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% | 3 |
| 7.3 | Prototype |  |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% |  |
| 7.4 | Unit Test1 | Ati |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% |  |
| 7.5 | Unit Test2 | Yang |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% |  |
| 7.6 | Unit Test3 | Pattira |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% |  |
| 7.7 | Unit Test4 | Yang |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% |  |
| 7.8 | Oral Presentation | All members |  | Thu 8/25/16 | Thu 8/25/16 | 1 | 0% | 1 |
| 7.9 | System Integration testing Test1 | All members |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% |  |
| 7.10 | Project Report: All above and references | All members |  | Tue 8/23/16 | Thu 8/25/16 | 3 | 0% |  |
| 7.11 | Send assignment to professor | Yang |  | Fri 8/26/16 | Fri 8/26/16 | 1 | 0% | 1 |
|  |  |  |  |  |  |  |  |  |
| **8** | **Prototype Demonstrations/Reviews** |  |  |  |  |  |  |  |
| 8.1 | Powerpoint |  |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% | 3 |
| 8.2 | Report |  |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% | 3 |
| 8.3 | Prototype |  |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% |  |
| 8.4 | Unit Test1 | Ati |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% |  |
| 8.5 | Unit Test2 | Yang |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% |  |
| 8.6 | Unit Test3 | Pattira |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% |  |
| 8.7 | Unit Test4 | Yang |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% |  |
| 8.8 | Oral Presentation | All members |  | Thu 9/01/16 | Thu 9/01/16 | 1 | 0% | 1 |
| 8.9 | System Integration testing Test1 | All members |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% |  |
| 8.10 | Project Report: All above and references | All members |  | Tue 8/30/16 | Thu 9/01/16 | 3 | 0% |  |
| 8.11 | Send assignment to professor | Yang |  | Fri 9/02/16 | Fri 9/02/16 | 1 | 0% | 1 |
|  |  |  |  |  |  |  |  |  |
| **9** | **Project Report** |  |  |  |  |  |  |  |
| 9.1 | Powerpoint |  |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% | 3 |
| 9.2 | Report |  |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% | 3 |
| 9.3 | Prototype |  |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% |  |
| 9.4 | Unit Test1 | Ati |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% |  |
| 9.5 | Unit Test2 | Yang |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% |  |
| 9.6 | Unit Test3 | Pattira |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% |  |
| 9.7 | Unit Test4 | Yang |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% |  |
| 9.8 | Oral Presentation | All members |  | Thu 9/08/16 | Thu 9/08/16 | 1 | 0% | 1 |
| 9.9 | System Integration testing Test1 | All members |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% |  |
| 9.10 | Project Report: All above and references | All members |  | Tue 9/06/16 | Thu 9/08/16 | 3 | 0% |  |
| 9.11 | Send assignment to professor | Yang |  | Fri 9/09/16 | Fri 9/09/16 | 1 | 0% | 1 |
|  |  |  |  |  |  |  |  |  |
| **10** | **Formal Project Presentation** |  |  |  |  |  |  |  |
| 10.1 | Powerpoint |  |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% | 3 |
| 10.2 | Report |  |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% | 3 |
| 10.3 | Prototype |  |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% |  |
| 10.4 | Unit Test1 | Ati |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% |  |
| 10.5 | Unit Test2 | Yang |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% |  |
| 10.6 | Unit Test3 | Pattira |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% |  |
| 10.7 | Unit Test4 | Yang |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% |  |
| 10.9 | System Integration testing Test1 | All members |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% |  |
| 10.10 | Project Report: All above and references | All members |  | Tue 9/13/16 | Thu 9/15/16 | 3 | 0% |  |
| 10.11 | Send assignment to professor | Yang |  | Fri 9/16/16 | Fri 9/16/16 | 1 | 0% | 1 |

CHAPTER 3

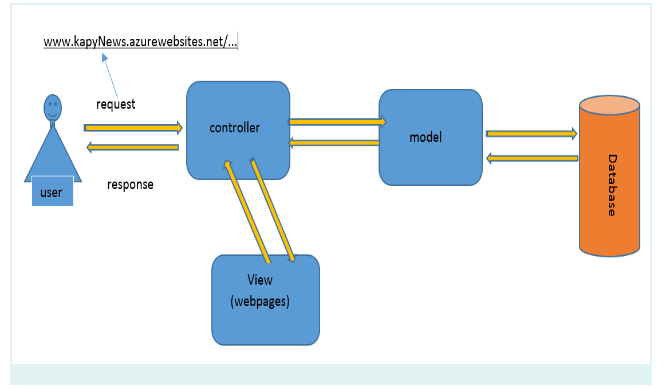
DESIGN

* 1. High Level Design Architecture



* 1. Design Model (Model Driven SW Development, Model Based SW Development)

We will be using the design pattern Model-View-Control (MVC). This pattern provides the ability to separate the representation of data in the database,  (the model),  from how it is displayed on the user interface, ( the view), and how it is accepted by the user,  (the controller).



* 1. UML Diagrams

A Sequence Diagrams for UC-001: Sign Up Use Case

D:\CS\CSC686\usercases\SignUpSequenceDiagram.png

An Activity Diagram for UC-002: Log in Use Case

D:\CS\CSC686\usercases\loginactivitydiagram.png

* 1. Database Design

All the system data stores are going to be realized using a modern relational database. Firstly we designed a conceptual model of the database and then the conceptual model is mapped to a relational database model. The following two subsections describe the two processes of the system database design.

1. Entity Relationship Diagram (ERD)

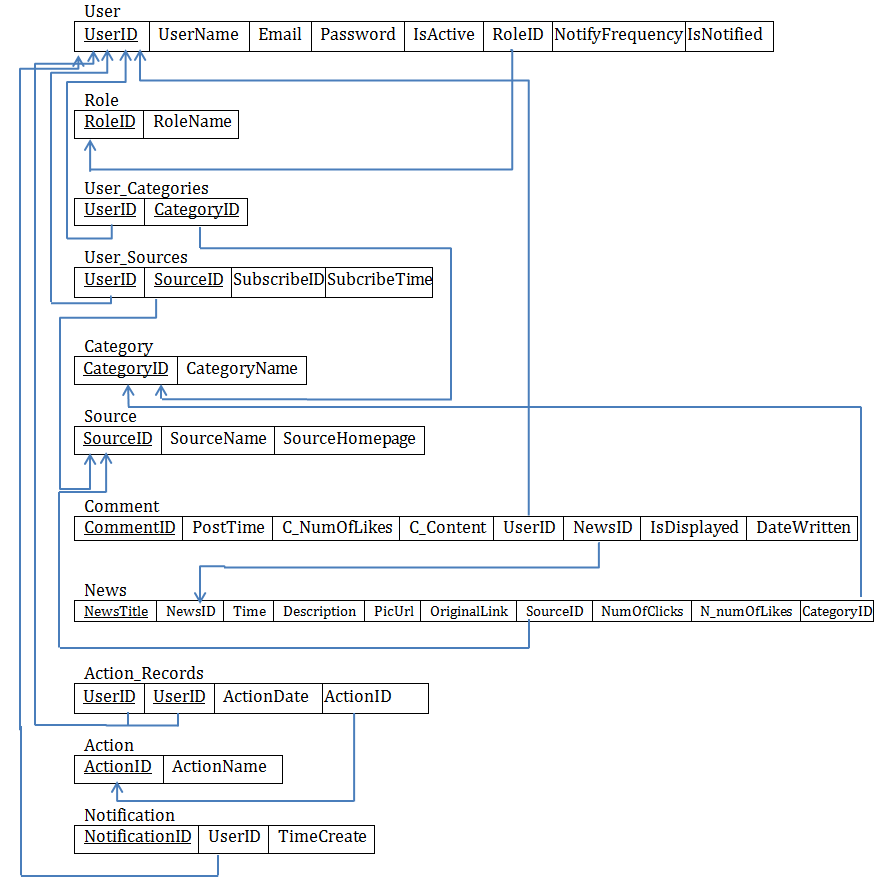
The entity-relationship diagram is the output of entity-relationship conceptual modeling. In our database, User entity is the entity for both administrators and members. Because of the different type of users, the role entity is used to distinguish the user type. The system creates a unique id for each user, so we use user ID as the primary key. In the news entity, to avoid duplicate news, we use newsTitle as the primary key and give each piece of news a newsID. Each piece of news may belong to one or more different categories but only come from one source. For users, they could subscribe to different sources to receive notification of news from. And users could also set categories they preferred. Each piece of comment comes from one user and can only be written for one piece of news while each user could make different comments for different news. The followeing ER diagram displays all entities and relationships in our database.

C:\Users\ChenAmber\Desktop\ERD.png

1. Entity Relationship Schema

In our system, we use SQL server, a cloud database management system in Azure, as our database management system. It is a relational database management system.

Based on the previous entity relationship diagram, we designed a relational database model, relational schema. The relational schema shows the relations, their attributes, primary keys, and foreign keys. The relational schema was normalized up to the fifth normal form ensuring it is free of any anomalies.



* 1. Graphical User Interface

1. UI of news

Kapy news website has three different pages for news display. The home page of website lists the top news as default for all users, including non-members, members without login (figure 1). Figure2 is a user interface design for the category page. In each category, the website lists all news in the specific category. Figure 3 is a detailed page for each piece of news.



Figure1. Homepage

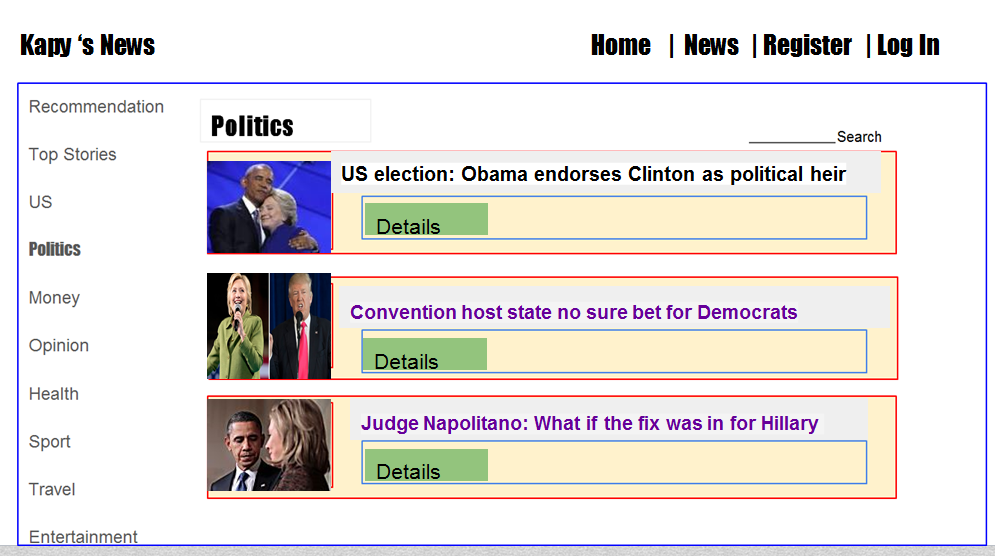


Figure2. List of news in politics category for all users



Figure3. News details page for all users

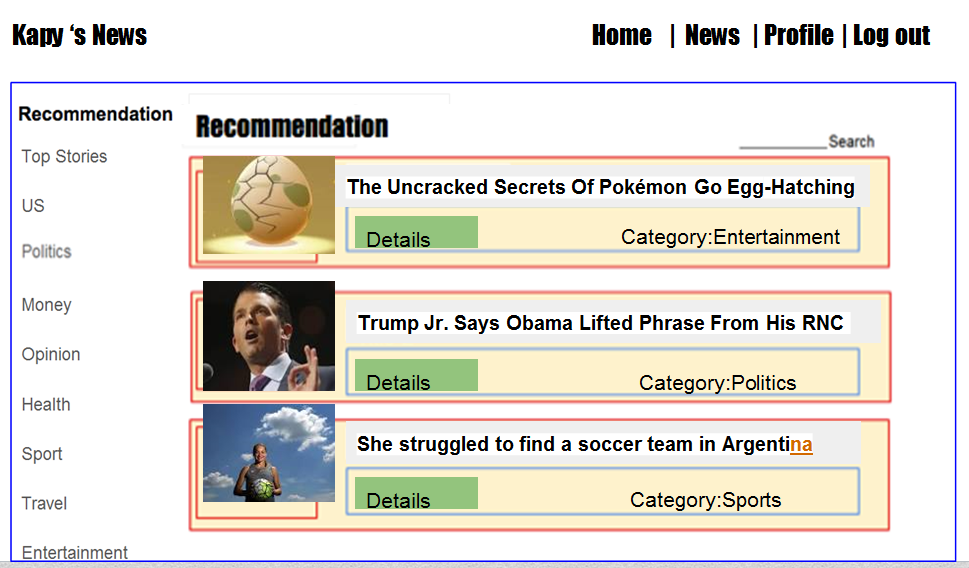


Figure4. News details page for members after login

1. UI of users

For both members and administrators, there is a same log on page (Figure 4). The system determines the role of users by information in the user database and directs the users to different pages after they log in the system. The administrator will be directed to the member management page as Figure 5. And members will be directed to the homepage of website (Figure 1). Figure 6 and Figure 7 are pages of registration. The first step of registration asks the user to enter basic information and the second step asks the user to select preferred categories from the candidates.

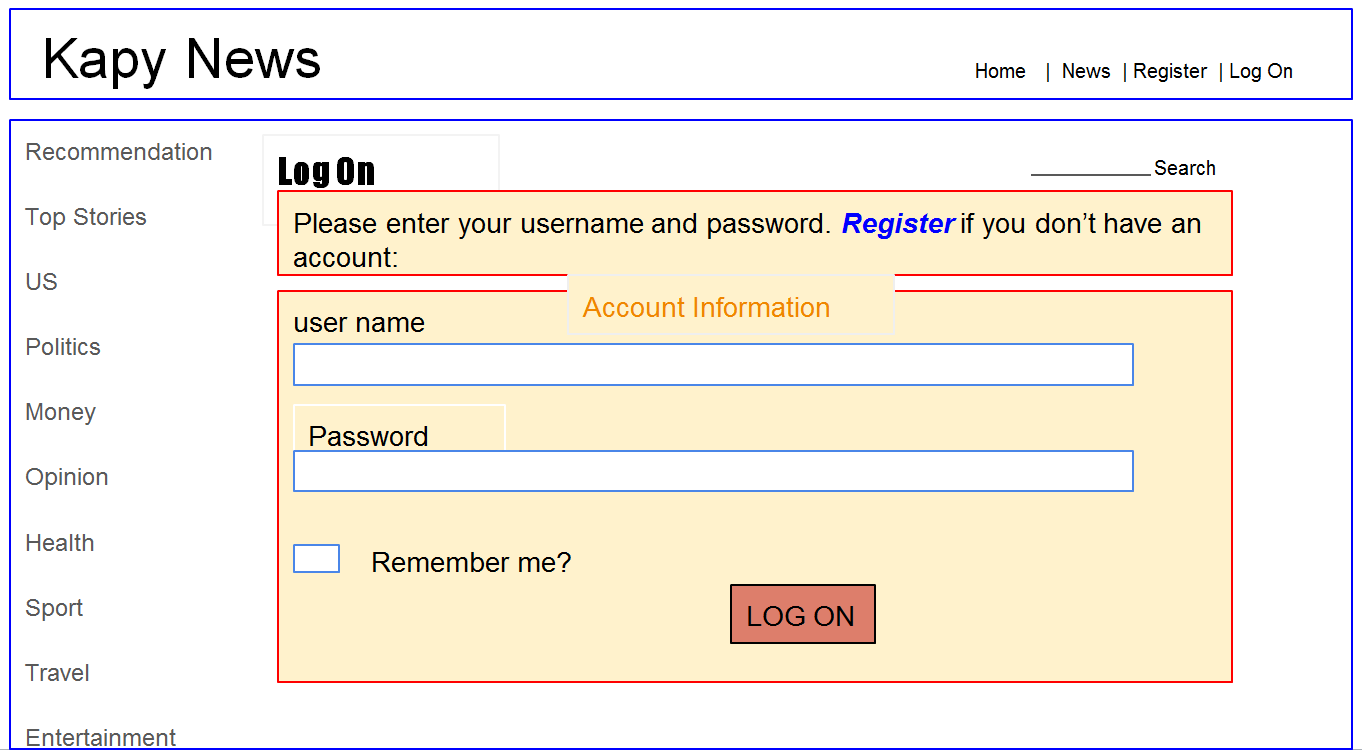


Figure5. Log on page

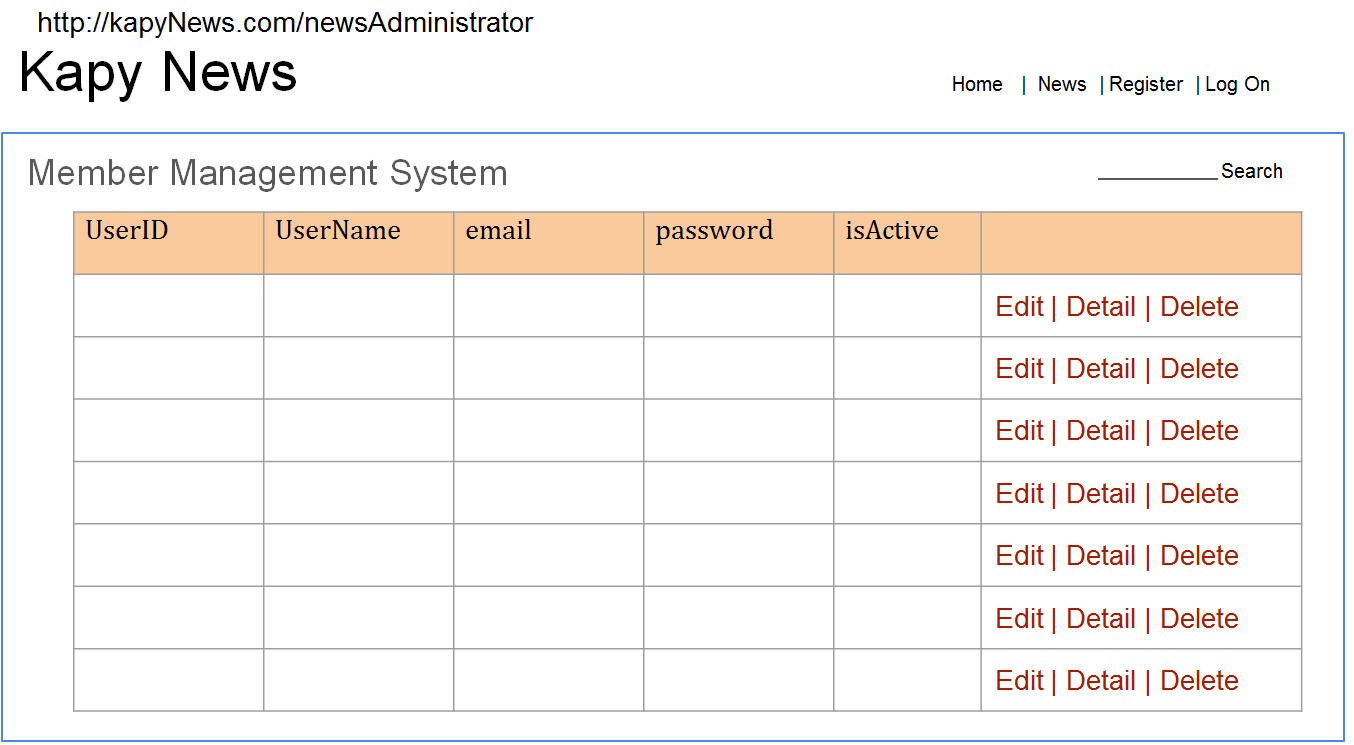


Figure6. Member management system page for administrators

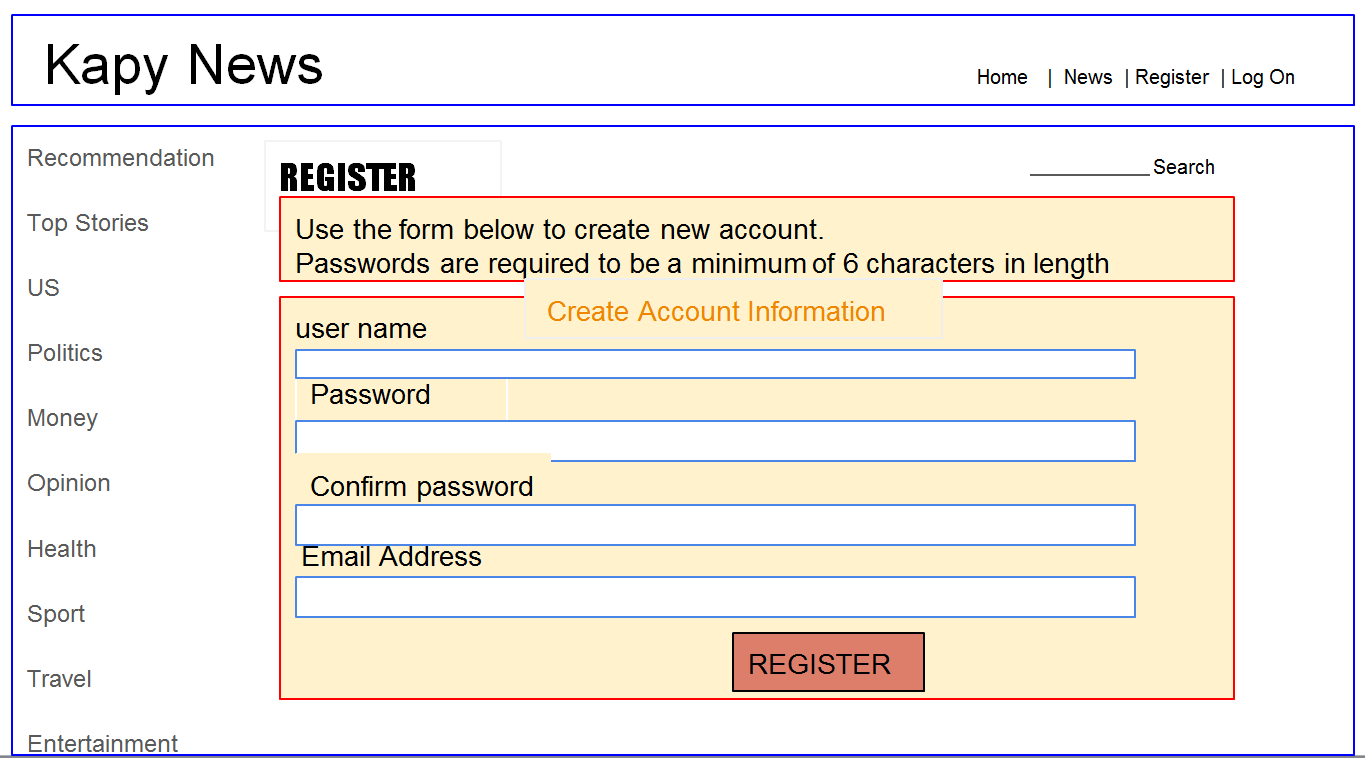


Figure7. Registration page (1)

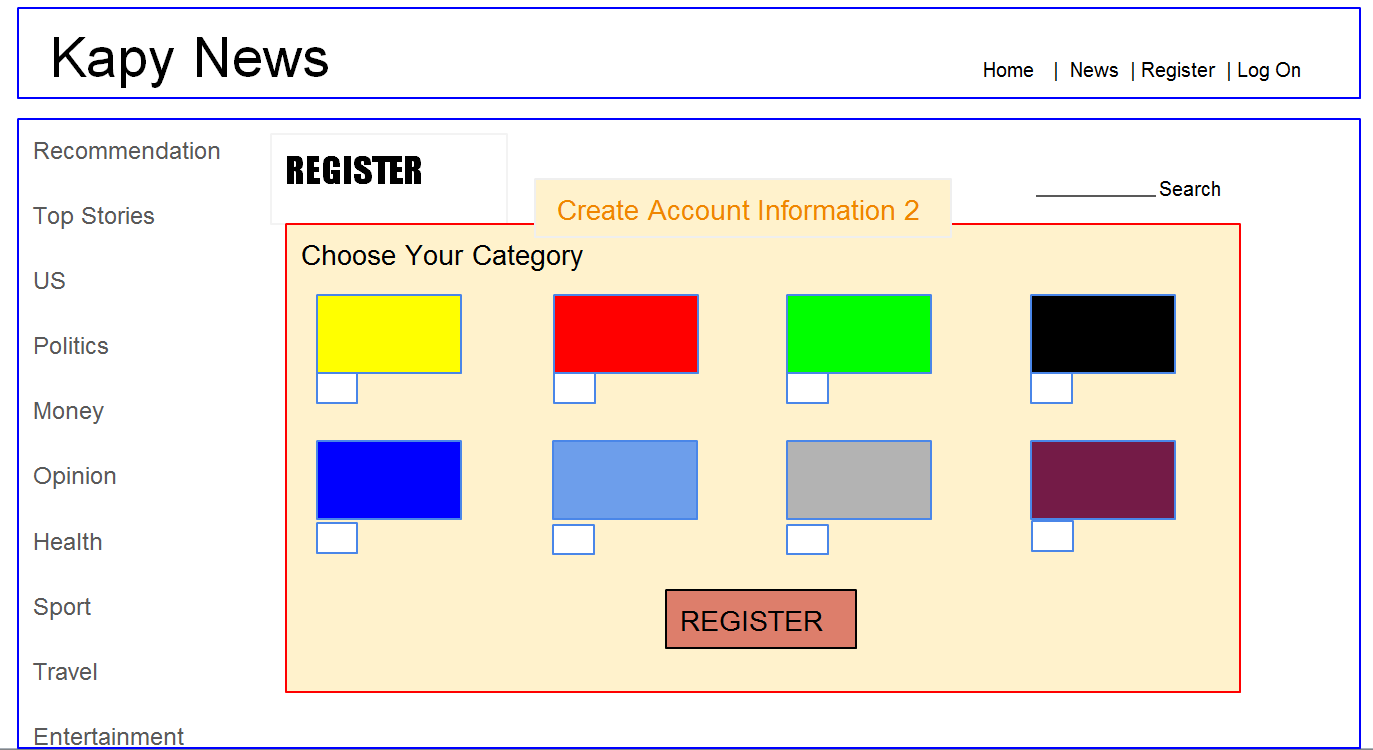


Figure8. Registration page (2)

* 1. Test Scenario

Scenario testing is a software testing activity that uses scenario tests. These tests are based on hypothetical story to help a person think through a complex problem or system and provide the outcome to be easy to evaluate.

|  |  |  |  |
| --- | --- | --- | --- |
| Use case ID | Test case ID | Test Case | Expected Result |
| Test Scenario 1: Validate the Signup Page | | | |
| UC-001 | TC-001 | The user registers with an existing username | The application shows an error message and allows the user to try again with a new username |
|  | TC-002 | The user enters invalid credentials for password | The application shows an error message and allows the user to try again with a new password |
|  | TC-003 | The user enters invalid credentials for email address | The application shows an error message and allows the user to try again with a new email |
|  | TC-004 | Enter existing email address | The application shows an error message and provide a link for “Forgot Password” |
| Test Scenario 2: Validate the Select Category Page | | | |
| UC-002 | TC-005 | Member selects preference news from news categories | The selected news categories are stored in the database |
| Test Scenario 3: Validate the Sign In Page | | | |
| UC-003 | TC-006 | Enter new email address and correct password | The user is signed in successfully and automatically linked to the personal homepage |
|  | TC-007 | Enter invalid email address and/or invalid password | The application should display an error message and allow the user to try again |
|  | TC-008 | Validate that new user created has new userID | The application must uniquely create a new userID for the new user |
|  | TC-009 | The user access to the his/her personal homepage | The user is linked to the personal homepage if and only if the username and password are matched with the database |
| Test Scenario 4: Validate the Reset Password Page | | | |
| UC-004 | TC-010 | The user clicks on “Forgot Password” | The “Forgot Password” links directly to the “Reset Password” page |
|  | TC-011 | The user enters the new password | The password should be validated |
|  | TC-012 | The user re-enter the new password | The password from should be matched with the first type |
|  |  | The password typed twice are matched | The password is reset successfully |
| Test Scenario 5: Validate Search Function | | | |
| UC-005 | TC-013 | The user types a keyword | The textbox is enabled |
|  | TC-014 | The user clicks on “Search” button after typed a keyword | The search function should be able to search for the content for that keyword |
|  |  | The user clicks on “Search” button with no keyword | The application should not reply any response |
| Test Scenario 6: Validate Subscribe  Function | | | |
| UC-006 | TC-015 | The user clicks “Subscribe” button on the webpage | The “Subscribe” button changes to be clickable and the new source should be added in the user subscribe list |
|  | TC-016 | The user tries to subscribe the subscribed source | The “Subscribe” button is already unclickable to prevent the user’s confusion |
| Test Scenario 7: Like News Function | | | |
| UC-007 | TC-017 | The user clicks on the “Like” button once | The “Like” button changes to a different color and the number of likes should be increased by one |
|  | TC-018 | The user clicks on the “Like button twice | The “Like” button changes to the original (default) form and the number of likes should be decreased by one |
| Test Scenario 8: Make Comment Function | | | |
| UC-008 | TC-019 | The user types a comment in the comment box | The text box should be enabled and the user should be able to type a comment |
|  | TC-020 | The user clicks on “Post” button | The message should be display within a minute |
| Test Scenario 9:Validate default page | | | |
|  | TC-021 | The user as a guest is online on the default page | The news are displayed in order of timeline |
|  | TC-022 | The user clicks on each categories | The user sees the news of that category displayed in order of timeline |
|  | TC-023 | The user clicks on each piece of news | The url should link directly to the news website |
|  | TC-024 | The news doesn't not contain any picture | The user should see a default image [we can define] |
|  | TC-025 | Display news on the webpage | The contents should be defined their size, font, and in Unicode [no error for accent] |
|  | TC-026 | The user selects a news by clicking the title link | Once the link is selected, the news link should change font color |

CHAPTER 4

IMPLEMENTATION

* 1. Systems Architecture
  2. Platforms and Technologies Used

For real time news aggregation, we design and create web crawlers using Python’s BeautifulSoup4 library. For the web development, the main technology we used is ASP.NET MVC framework together with entity framework. We also build Microsoft SQL Database Server and design online databases for dynamic data storage in Azure and make use of Microsoft Azure’s web service for application deployment.

1. Crawler implementation

KAPY’s news is a news aggregation system which collecting news from various news websites. After research, we use of Python, a widely used high-level, general-purpose, interpreted, dynamic programming language. Another obvious reason that we use Python is we need to make use of the Beautiful Soup 4 for creating news crawlers. The Beautiful Soup 4 a Python library for pulling data out of HTML and XML files. It works with the parser to provide idiomatic ways of navigating, searching, and modifying the parse tree and commonly saves programmers hours or days of work.[https://www.crummy.com/software/BeautifulSoup/bs4/doc/]

a. Crawlers Architecture:

The crawlers we designed have two main tasks; one is to find all URL links from a given web page and another one is to find the news object. When we start a crawler program, the project creates two text files for recording. The queue file records all links that crawlers found and are waiting to be crawled. The crawled file records all links which have been crawled by crawlers.

For example, given a base URL, the crawler finds all links in the page firstly, writes all links into the queue file and writes the base URL into the crawled file. And then the crawler reads the next link (link A) from queue file, crawl the link (A) to get more links and remove the link (A) from queue file into the crawled file. If the link (B) crawled from the page of link (A) is not in the crawled file or the queue file, add the link (B) into the end of queue file. If there’s a news object in the page of link (A), crawl the news object and insert the news content into database.

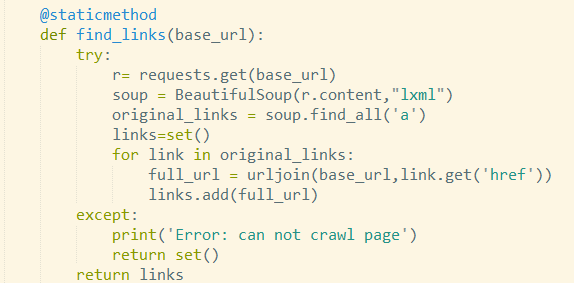


Figure x. method to crawl links from one base link

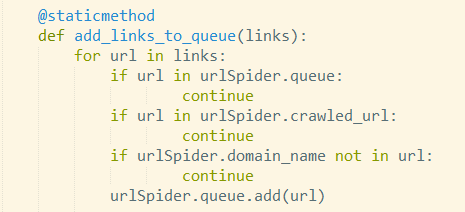


Figure x. method to examine and add links crawled into queue file

b. Crawling News:

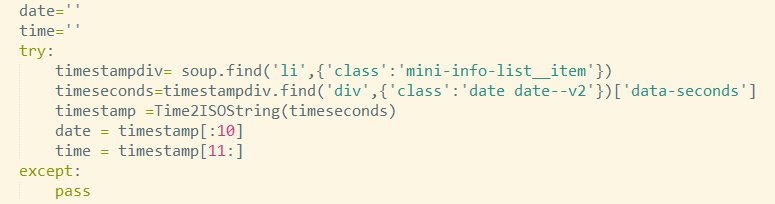
Before crawling news content, we define a News class in our project. The attributes defined for the news are corresponding with the fields designed in the table in our database. For each news object, we define:

news = newsItem(title,complete\_title,time,date,sourceId,description,origin\_url,categoryId,author,pic\_url)

The attribute “title” is the first 20 characters of the complete\_title, which is used to prevent recording duplicate news in the database because one common news may has different URL links.

The information to be crawled depends on the xml format of the web page content. For example, the following code is to get the post time of the piece of news. The value of “data-seconds” attribute inside the div with class “date date--v2” represents the post time in the second format. So we need to convert the time in second format into time in ISO format. After that we could get the date and time from the page.

After getting all elements of news content, we create a news object and insert the news object into news table in the database.



c. Inserting News content into Database:

To implement queries of database in MS SQL Server in python program, we need to make use of the pyodbc library of Python.

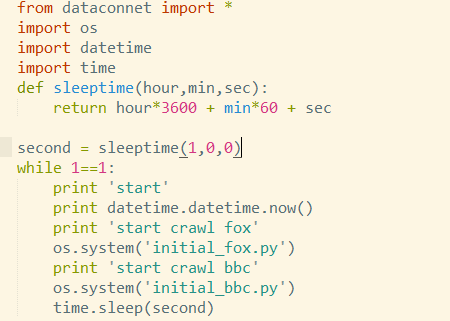


After connecting with the database in Azure, the following code allows inserting news content into the News table. “newsItem” is the object created and stored news content crawled before. “crawlTime” is actually the time that the news content inserted into the News table.



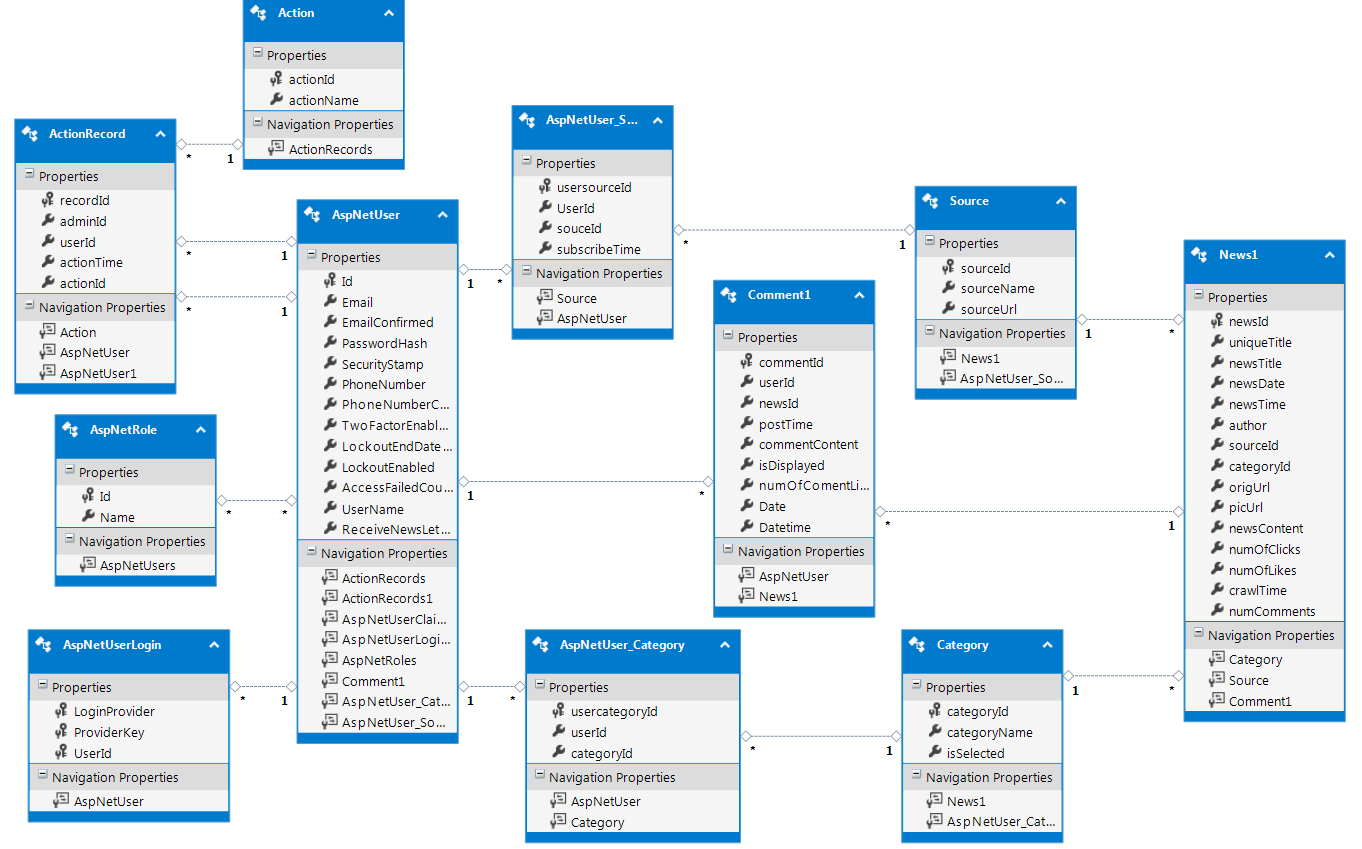
d. Top caller of crawlers:

News is always updated frequently. To get the most recent news, we need to schedule our crawlers to start crawl tasks in certain frequency. The information to be crawled depends on the xml format of the web page content. Since different news websites have different format of webpage, we need to create different crawlers for different news websites. In our project, we have two different kinds of crawlers, one for BBC news and another for Fox news. So we define a newsCrawler class and build two different kinds of newsCrawler objects. For each news websites, the top caller calls the specific crawler to start crawling. After getting 100 of most recent news, the top caller calls another crawler to work. And we define the callers to call crawlers every one hour.



1. Database implementation

The database we created is an online database in Azure. Since we use the database first MVC pattern, our database is implemented firstly before the web implementation. Most tables of our database are corresponding with the database design part except the tables for users. To make use of the Identity2.0



1. ASP.NET MVC framework
   1. Languages and Tools
   2. User Interfaces with Screen Shots

CHAPTER 5

RESULTS AND CONCLUSIONS

* 1. a

CHAPTER 6

FUTURE RESEARCH

* 1. a

References

Oliver Oechslein, Mario Haim, Andreas Graefe. (2015). The Digitization of News Aggregation: Experimental Evidence on Intention to Use and Willingness to Pay for Personalized News Aggregator: *Proceedings of 48th Hawaii International Conference on System Sciences*, Hawaii, U.S.A.

Kyo-Joong Oh, Won-Jo Lee, Chae-Gyun Lim, Ho-Jin Choi. (2014). Personalized News Recommendation using Classified Keywords to Capture User Preference. *IEEE, 16th International Conference on Advanced Communication Technology*, 1283-1287.

Ruchika Patel, Pooja Bhatt. (2015). Semantic Focused Web Crawler for Service Discovery Using Data Mining Technique. *COMPUSOFT*, Vol. 4(7), 253-259.

Vassilis Papavassiliou, Prokopis Prokopidis. (2013). A modular open-source focused crawler for mining monolingual and bilingual corpora from the web. *The 6th Workshop on Building and Using Comparable Corpora*, Sofia, Bulgaria, August 8, 2013.

Clegg, Dai; Barker, Richard. (2004-11-09). *Case Method Fast-Track: A RAD Approach.*  Addison-Wesley .

Hunt, John(2013). "*Gang of four design patterns." Scala Design Patterns,* 135-136.

Bush, Vannevar, and As We May Think. "The atlantic monthly." 176.1 (1945): 101-108.

"Digital News Audience: Fact Sheet | Pew ... - Journalism & Media." 2016. 29 Jul. 2016

Chowdhury, Sudatta, and Monica Landoni. "News aggregator services: user expectations and

experience." *Online Information Review* 30.2 (2006): 100-115.

Appendices

Label each appendix separately, and start each on a new page.

1. Chowdhury, Sudatta, and Monica Landoni. "News aggregator services: user expectations and experience." *Online Information Review* 30.2 (2006): 100-115. [↑](#footnote-ref-1)
2. Bush, Vannevar, and As We May Think. "The atlantic monthly." *As we may think* 176.1 (1945): 101-108. [↑](#footnote-ref-2)
3. "State of the news media 2013: Pew Research Center's Project for ..." 2013. 29 Jul. 2016 <<http://journalistsresource.org/studies/society/news-media/news-media-2013-pew-research-center>> [↑](#footnote-ref-3)
4. "Digital News Audience: Fact Sheet | Pew ... - Journalism & Media." 2016. 29 Jul. 2016 <<http://www.journalism.org/2016/06/15/digital-news-audience-fact-sheet/>> [↑](#footnote-ref-4)
5. [↑](#footnote-ref-5)