FALL 2014 GROUP 1

CSCI 50600

SOFTWARE DESIGN DOCUMENT

Twitter Project

Patrick Burton

Arvind Nair

Lakshmi Swathi Chavvakul

IUPUI

Table of Contents

[2 Introduction 6](#_Toc405912180)

[2.1 Purpose 6](#_Toc405912181)

[2.2 Scope 6](#_Toc405912182)

[2.3 Methodology 6](#_Toc405912183)

[2.4 Abbreviations 6](#_Toc405912184)

[3 Detailed Software Design Description in Iteration 1 7](#_Toc405912185)

[3.1 Use Case Model (General View) 7](#_Toc405912186)

[3.1.1 Diagram 7](#_Toc405912187)

[3.1.2 Description 7](#_Toc405912188)

[3.2 Actors View 8](#_Toc405912189)

[3.2.1 Diagram 8](#_Toc405912190)

[3.2.2 Description 8](#_Toc405912191)

[3.3 Primary Use Case View 9](#_Toc405912192)

[3.3.1 Diagram 9](#_Toc405912193)

[3.3.2 Description 9](#_Toc405912194)

[3.4 Retry View 10](#_Toc405912195)

[3.4.1 Diagram 10](#_Toc405912196)

[3.4.2 Description 10](#_Toc405912197)

[3.5 Login Sequence View (Correct Credentials) 11](#_Toc405912198)

[3.5.1 Diagram 11](#_Toc405912199)

[3.5.2 Description 11](#_Toc405912200)

[3.6 Login Sequence View (Retry Sequence) 12](#_Toc405912201)

[3.6.1 Diagram 12](#_Toc405912202)

[3.6.2 Description 12](#_Toc405912203)

[4 Detailed Software Design Description in Iteration 2 13](#_Toc405912204)

[4.1 Use Case Model (General View) 13](#_Toc405912205)

[4.1.1 Diagram 13](#_Toc405912206)

[4.1.2 Description 13](#_Toc405912207)

[4.2 Primary Use Case View 14](#_Toc405912208)

[4.2.1 Diagram 14](#_Toc405912209)

[4.2.2 Description 14](#_Toc405912210)

[4.3 Auxillary Use Case View 15](#_Toc405912211)

[4.3.1 Diagram 15](#_Toc405912212)

[4.3.2 Description 15](#_Toc405912213)

[4.4 Login Sequence View (Correct Credentials) 16](#_Toc405912214)

[4.4.1 Diagram 16](#_Toc405912215)

[4.4.2 Description 16](#_Toc405912216)

[5 Detailed Software Design Description in Iteration 3 18](#_Toc405912217)

[5.1 Use Case Model (General View) 18](#_Toc405912218)

[5.1.1 Diagram 18](#_Toc405912219)

[5.1.2 Description 19](#_Toc405912220)

[5.2 Actors View 19](#_Toc405912221)

[5.2.1 Diagram 19](#_Toc405912222)

[5.2.2 Description 19](#_Toc405912223)

[5.3 Primary Use Case Sequence (General View) 20](#_Toc405912224)

[5.3.1 Description 20](#_Toc405912225)

[5.4 Primary Use Case View 20](#_Toc405912226)

[5.4.1 Description 20](#_Toc405912227)

[5.4.2 Diagram: 21](#_Toc405912228)

[5.5 Auxillary Use Case View 21](#_Toc405912229)

[5.5.1 Description 21](#_Toc405912230)

[5.5.2 Diagram 22](#_Toc405912231)

[5.6 Login Sequence View 22](#_Toc405912232)

[5.6.1 Description 22](#_Toc405912233)

[5.6.2 Diagram 23](#_Toc405912234)

[5.7 Post Tweet Sequence View 23](#_Toc405912235)

[5.7.1 Description 23](#_Toc405912236)

[5.7.2 Diagram 24](#_Toc405912237)

[5.8 Load More Sequence View 24](#_Toc405912238)

[5.8.1 Description 24](#_Toc405912239)

[5.8.2 Diagram 25](#_Toc405912240)

[5.9 Database 25](#_Toc405912241)

[5.9.1 Description 25](#_Toc405912242)

[5.9.2 Diagram 26](#_Toc405912243)

[6 Detailed Software Design Description in Iteration 4 27](#_Toc405912244)

[6.1 Use Case Model (General View) 27](#_Toc405912245)

[6.1.1 Diagram 27](#_Toc405912246)

[6.1.2 Description 28](#_Toc405912247)

[6.2 Primary Use Case View 28](#_Toc405912248)

[6.2.1 Diagram 28](#_Toc405912249)

[6.2.2 Description 29](#_Toc405912250)

[6.3 Primary Use Case Sequence (General View) 29](#_Toc405912251)

[6.3.1 Diagram 29](#_Toc405912252)

[6.3.2 Description 29](#_Toc405912253)

[6.4 Login Sequence View 30](#_Toc405912254)

[6.4.1 Diagram: 30](#_Toc405912255)

[6.4.2 Description 30](#_Toc405912256)

[6.5 Load More Sequence View 31](#_Toc405912257)

[6.5.1 Diagram 31](#_Toc405912258)

[6.5.2 Description 31](#_Toc405912259)

[6.6 Database 32](#_Toc405912260)

[6.6.1 Diagram 32](#_Toc405912261)

[6.6.2 Description 33](#_Toc405912262)

[7 Software Architecture 33](#_Toc405912263)

[7.1 Client-Server Architecture 33](#_Toc405912264)

[7.1.1 Presentation Layer: 33](#_Toc405912265)

[7.1.2 Business Logic Layer: 33](#_Toc405912266)

[7.1.3 Database Layer: 33](#_Toc405912267)

[7.2 Repository Architecture 33](#_Toc405912268)

[7.3 Parallel Communicating Processes Architecture 33](#_Toc405912269)

[Figure 1 : Use Case Model (General View) - Iteration 1 7](#_Toc405889136)

[Figure 2 : Actors View - Iteration 1 8](#_Toc405889137)

[Figure 3 : Primary Use Case View - Iteration 1 9](#_Toc405889138)

[Figure 4 : Retry View - Iteration 1 10](#_Toc405889139)

[Figure 5 : Login Sequence View (Correct Credentials) - Iteration 1 11](#_Toc405889140)

[Figure 6 : Login Sequence View (Retry Sequence) - Iteration 1 12](#_Toc405889141)

[Figure 7 : Use Case Model (General View) - Iteration 2 13](#_Toc405889142)

[Figure 8 : Primary Use Case View - Iteration 2 14](#_Toc405889143)

[Figure 9 : Auxillary Use Case View - Iteration 2 15](#_Toc405889144)

[Figure 10 : Login Sequence View (Correct Credentials) - Iteration 2 16](#_Toc405889145)

[Figure 11 : Use Case Model (General View) - Iteration 3 18](#_Toc405889146)

[Figure 12 : Actors View - Iteration 3 19](#_Toc405889147)

[Figure 13 : Primary Use Case Sequence (General View) - Iteration 3 20](#_Toc405889148)

[Figure 14 : Primary Use Case View - Iteration 3 21](#_Toc405889149)

[Figure 15 : Auxillary Use Case View - Iteration 3 22](#_Toc405889150)

[Figure 16 : Login Sequence View - Iteration 3 23](#_Toc405889151)

[Figure 17 : Post Tweet Sequence View - Iteration 3 24](#_Toc405889152)

[Figure 18 : Load More Sequence View - Iteration 3 25](#_Toc405889153)

[Figure 19 : Database - Iteration 3 26](#_Toc405889154)

[Figure 20 : Use Case Model (General View) - Iteration 4 27](#_Toc405889155)

[Figure 21 : Primary Use Case View - Iteration 4 28](#_Toc405889156)

[Figure 22 : Primary Use Case Sequence (General View) - Iteration 4 29](#_Toc405889157)

[Figure 23 : Login Sequence View - Iteration 4 30](#_Toc405889158)

[Figure 24 : Load More Sequence View - Iteration 4 31](#_Toc405889159)

[Figure 25 : Database - Iteration 4 33](#_Toc405889160)

# 

# Introduction

## Purpose

This Software Design Document describes in detail the architecture of the software system and also the Use Case and sequence diagrams of each of the four iterations which we have done as we had adopted an agile development process. This document is intended for a clear understanding of our software system and its working for the users and also for future extensions as and when needed.

## Scope

The software which we developed is a simple messaging application which will help users to perform the following basic activities:

1. The user can after logging in post tweets or messages.

2. He can read messages posted by others also.

3. He can perform these functions from Desktop, PC and mobile devices.

## Methodology

For developing the software, we have followed an agile methodology of 4 iterations. For each of the iterations we have implemented the basic requirements but added onto them the extra features depending on the requirements and metric measurements to improve the quality of software. We have gradually added and improved the features and it is shown in the metrics measurements documents.

## Abbreviations

1. CAS : Central Authentication Server

2. IUPUI: Indiana University Purdue University-Indianapolis

3. MAC: Math Assistance Center

4. AWS: Amazon Web Services

# Detailed Software Design Description in Iteration 1

## Use Case Model (General View)

### Diagram

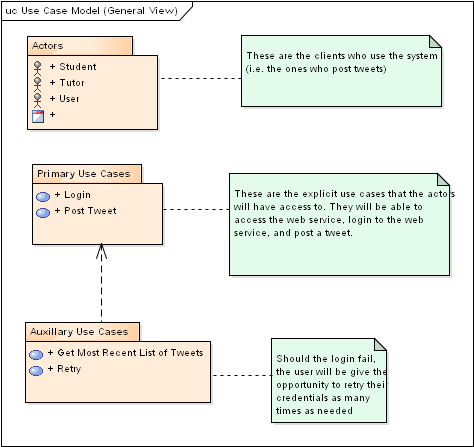


Figure 1 : Use Case Model (General View) - Iteration 1

### Description

Actors: The actors are the users who will use the system. They are student, tutor and user.

Primary Use Cases: It shows the functions of login and posting tweet by any of the actors.

Auxillary Use Cases: It shows additional scenarios or functions like getting most recent tweets and retrying if the login fails.

## Actors View

### Diagram

### Actors View.bmp

Figure 2 : Actors View - Iteration 1

### Description

It shows the in detail the user having types of student and tutor.

## Primary Use Case View

### Diagram

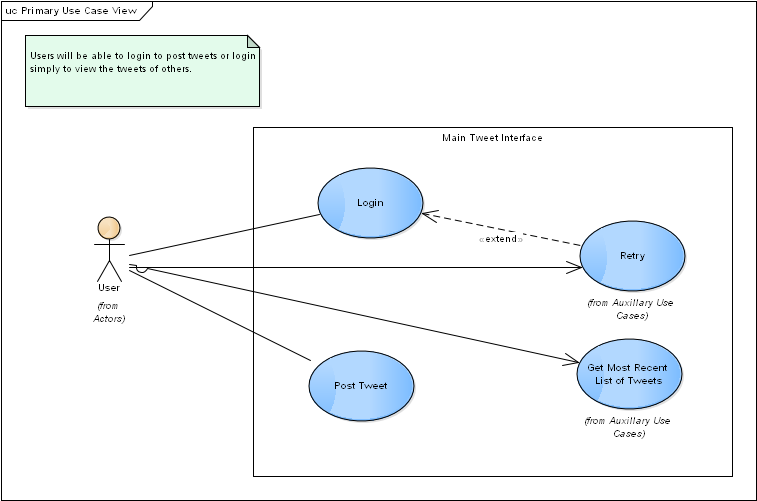


Figure 3 : Primary Use Case View - Iteration 1

### Description

Here, the functions which can be performed by the user like login, posting tweet, retrying and getting most recent tweets are shown.

## Retry View

### Diagram

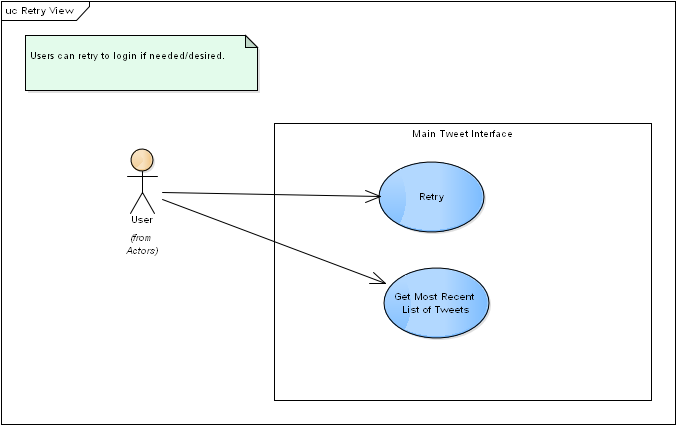


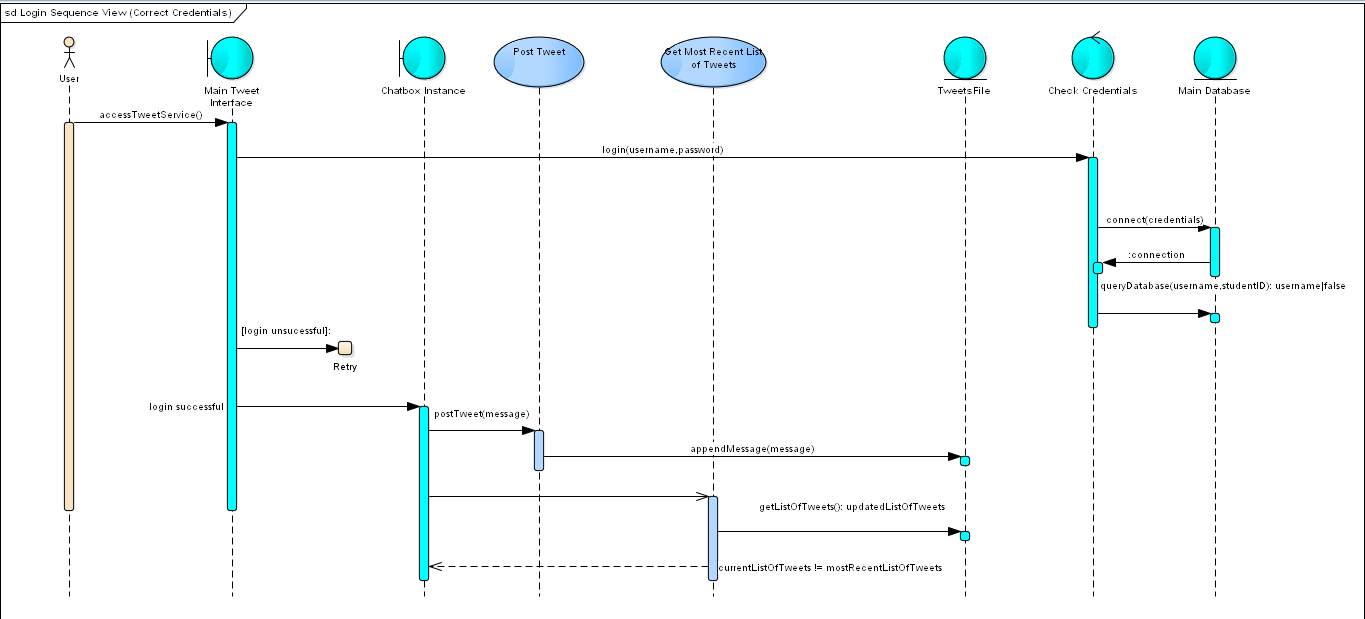
Figure 4 : Retry View - Iteration 1

### Description

Here, the user can retry and once successful, can get the most recent tweets.

## Login Sequence View (Correct Credentials)

### Diagram



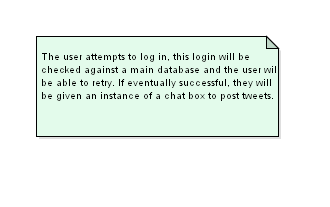


Figure 5 : Login Sequence View (Correct Credentials) - Iteration 1

### Description

The sequence diagram above shows the events in the system. They are described below:

1.The user first accesses the website by visiting the URL.

2. He enters his credentials in the username and password field and clicks the login button.

3. The credentials are checked with the database and then once approved the user is taken to the main chat interface.

4. Here, he can type in the message box and press enter and the message is tweeted out and displayed on the main message wall.

5. The tweets are stored in a tweets file and the user can get the most recent tweets from them.

## Login Sequence View (Retry Sequence)

### Diagram

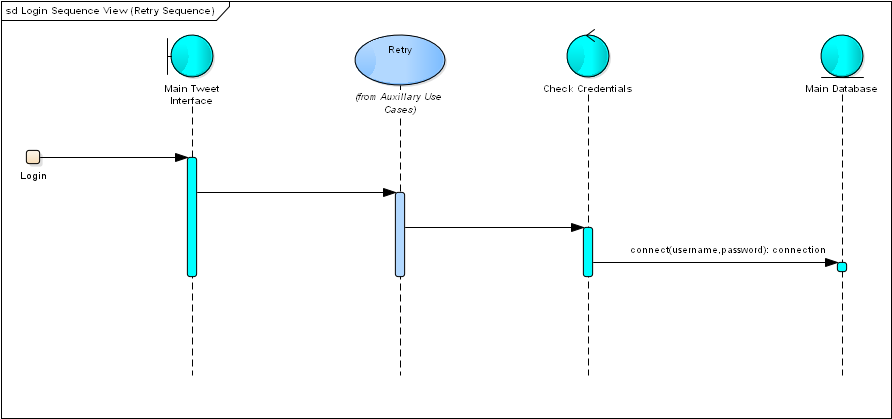


Figure 6 : Login Sequence View (Retry Sequence) - Iteration 1

### Description

This just shows the retry scenario in which the user has to retry and the credentials are checked with the database. If they match, the user can login otherwise he has to retry again.

# Detailed Software Design Description in Iteration 2

## Use Case Model (General View)

### Diagram

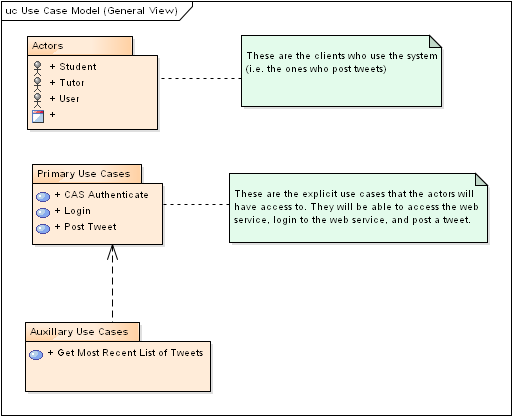


Figure 7 : Use Case Model (General View) - Iteration 2

### Description

In the second iteration, the use case remains almost the same. We have added CAS Authentication to enhance security.

## Primary Use Case View

### Diagram

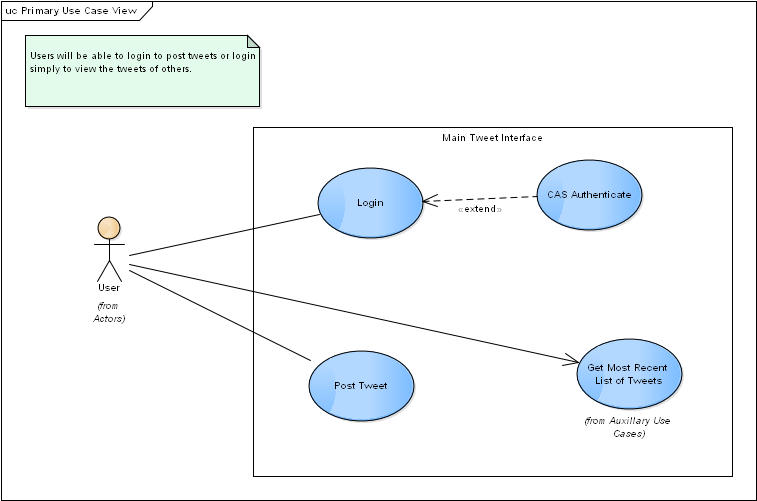


Figure 8 : Primary Use Case View - Iteration 2

### Description

The primary use case also shows the CAS authentication which is used during login.

## Auxillary Use Case View

### Diagram

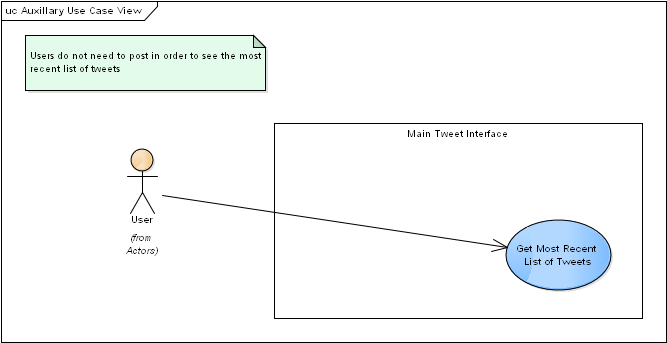


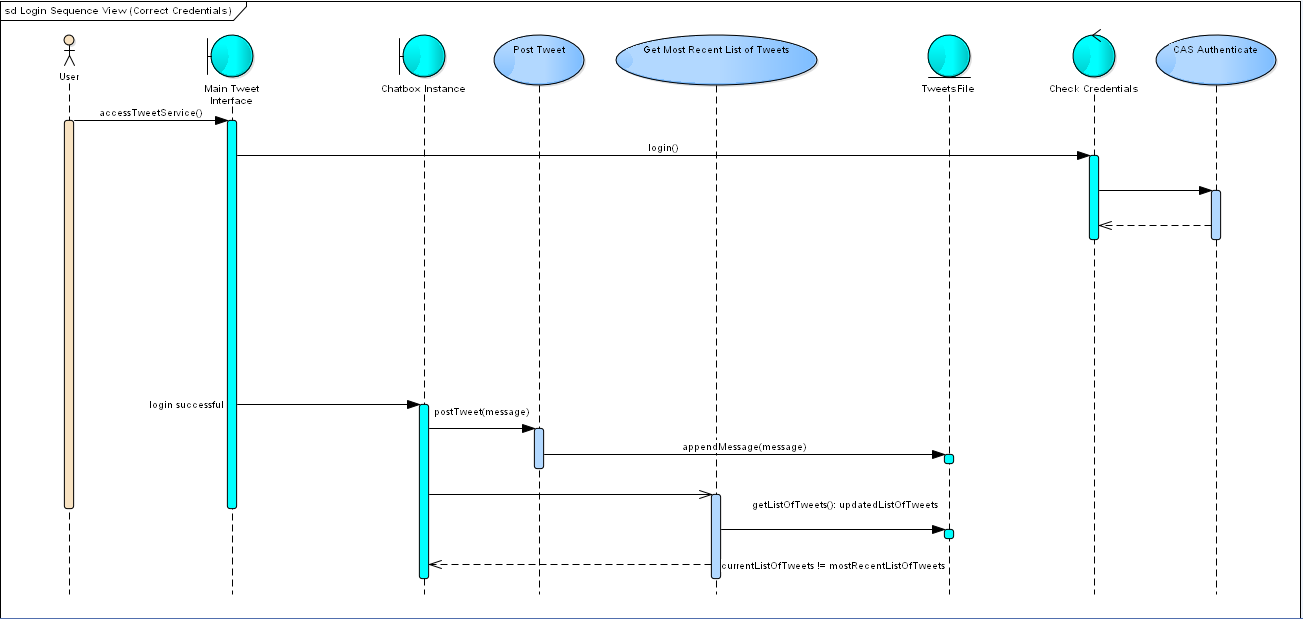
Figure 9 : Auxillary Use Case View - Iteration 2

### Description

Here, the users do not need to post to see the most recent tweets.

## Login Sequence View (Correct Credentials)

### Diagram



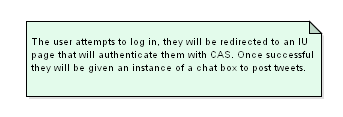


Figure 10 : Login Sequence View (Correct Credentials) - Iteration 2

### Description

1. The difference from the first iteration is that the user when he logs in, he has to go through the CAS authentication which is done by IUPUI and hence is very secure. When the CAS is successful the user can log in to tweet or use the service. The retry count is maintained by the CAS authentication.

2. The user will get a list of most updated tweets when he logs into the system.

# Detailed Software Design Description in Iteration 3

## Use Case Model (General View)

### Diagram

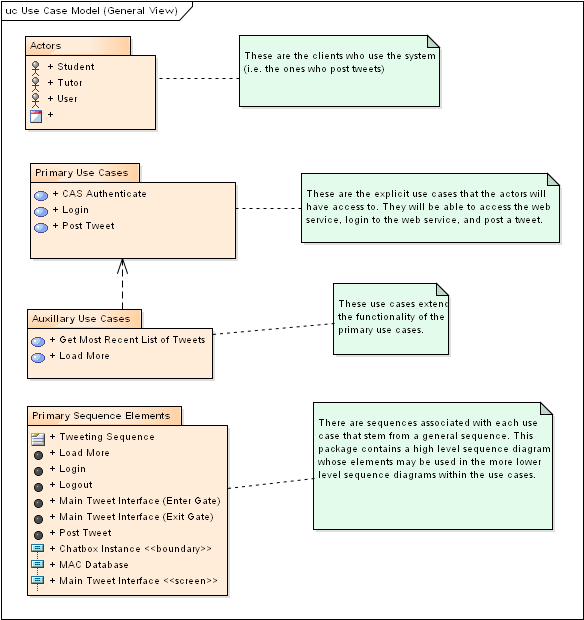


Figure 11 : Use Case Model (General View) - Iteration 3

### Description

1. Here, in Iteration 3 we have added the load more option which will load some previous tweets.

2. We have used MySQL for storing the tweet information.

3. The chatbox has a limit for the amount of characters to be sent.

4. The send button is also introduced to send out a tweet rather than pressing Enter.

## Actors View

### Diagram

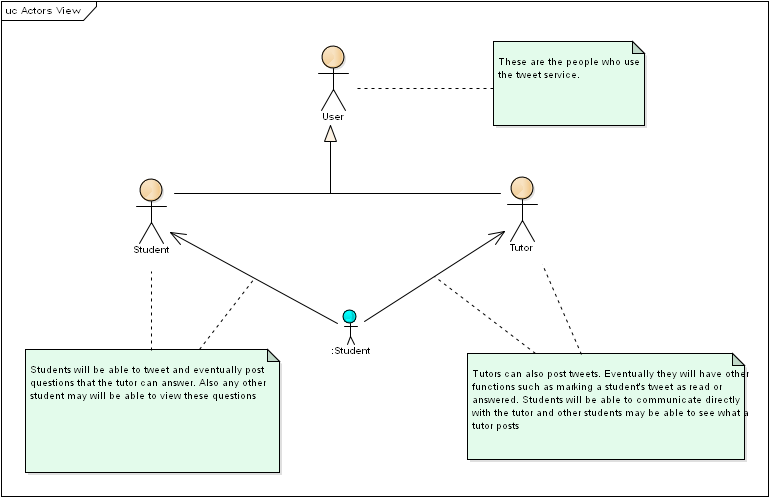


Figure 12 : Actors View - Iteration 3

### Description

This remains the same as Iteration 1.

## Primary Use Case Sequence (General View)

### Description

1. The user after login can post as many tweets as he wants and also can use the load more option to get the previous tweets on the screen.

2. Now, the tweets will be retrieved from MySQL Database.

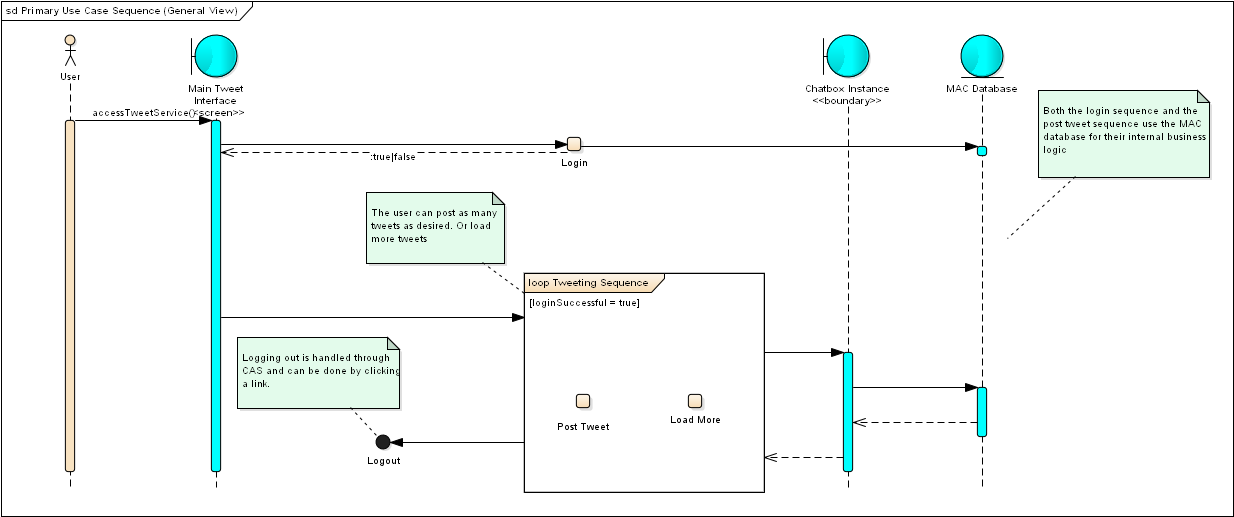


Figure 13 : Primary Use Case Sequence (General View) - Iteration 3

## Primary Use Case View

### Description

This just shows the load more option which the user can use to load more previous tweets.

### Diagram:

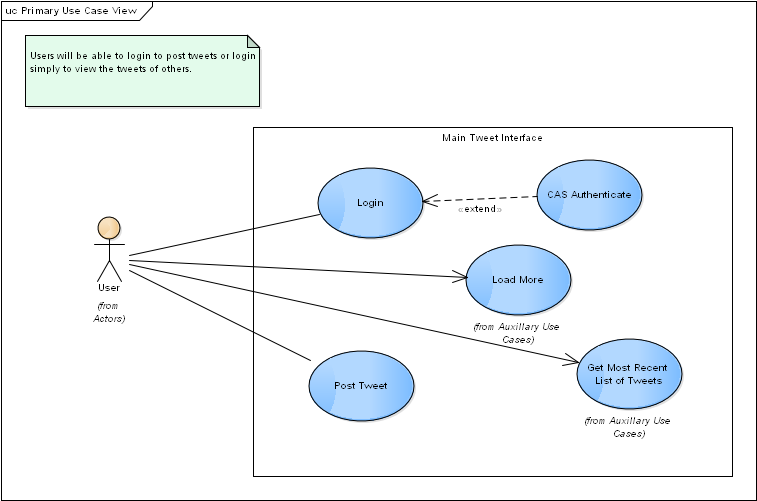


Figure 14 : Primary Use Case View - Iteration 3

## Auxillary Use Case View

### Description

It also shows the load more option which is used to retrieve more previous tweets.

### Diagram

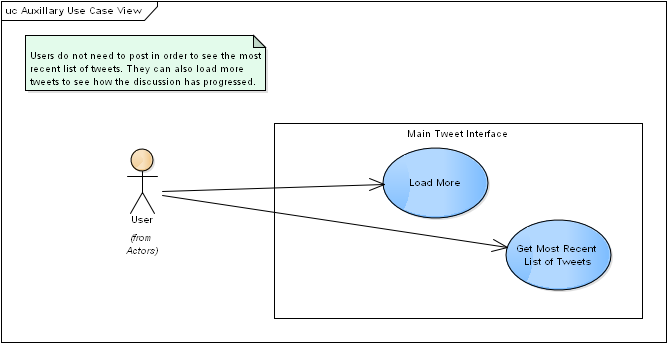


Figure 15 : Auxillary Use Case View - Iteration 3

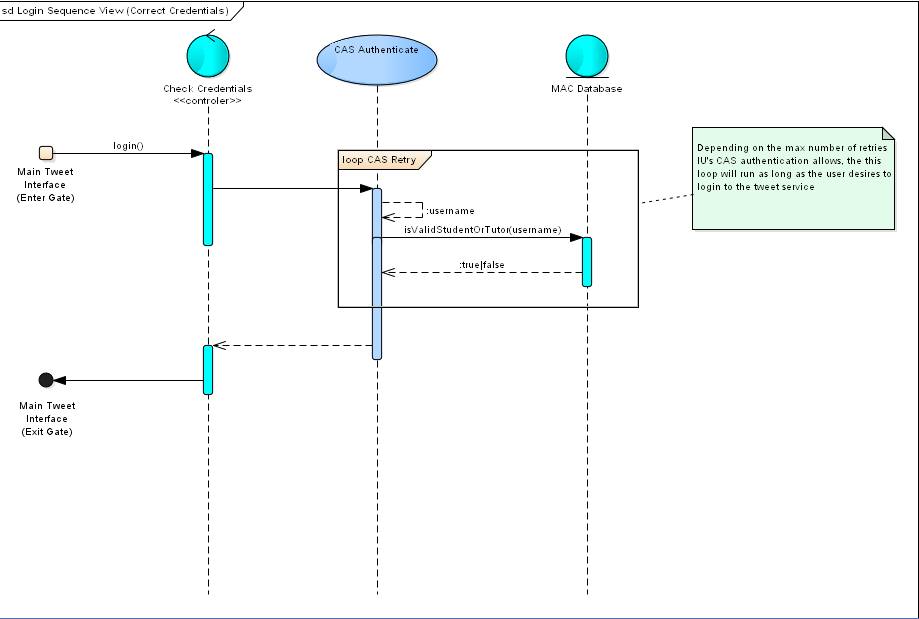
## Login Sequence View

### Description

1. This shows the login sequence controlled by the CAS authentication in which the retry is controlled by CAS and it is checked with the database.

2. The retry is allowed as long as CAS allows it. So, it has a very high level of security.

### Diagram



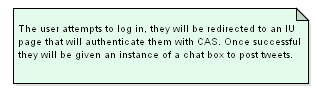


Figure 16 : Login Sequence View - Iteration 3

## Post Tweet Sequence View

### Description

1. Here, the user gets the list of most recent tweets.

2. Instead of being retrieved from the file, the tweets are retrieved from the MySQL server.

### Diagram

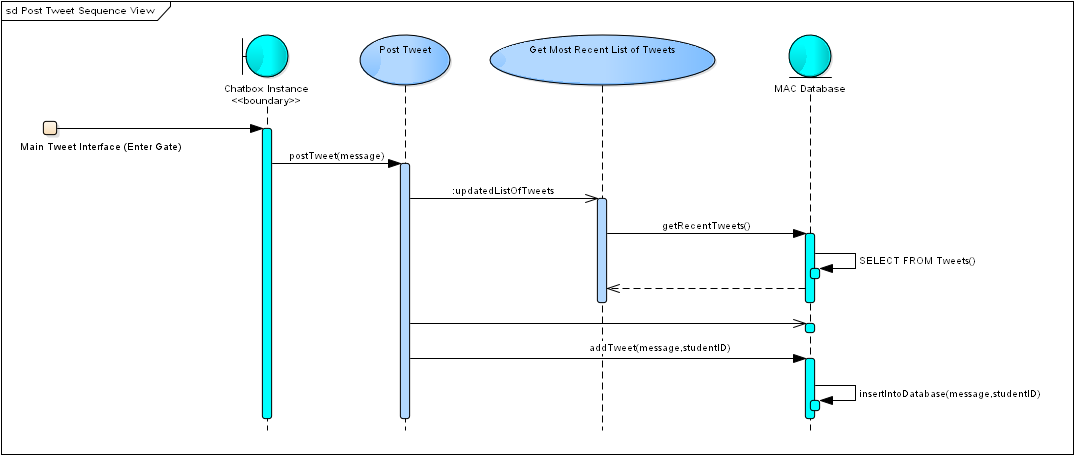


Figure 17 : Post Tweet Sequence View - Iteration 3

## Load More Sequence View

### Description

1. In Iteration 3, the user can after tweeting use the load more option to retrieve the previous tweets for his convenience.

2. When the load more button is clicked then the list of previous tweets from the database is retrieved in backwards fashion and displayed on the screen.

3. The user can continue tweeting.

4. The chatbox has also a limit of sending messages.

### Diagram

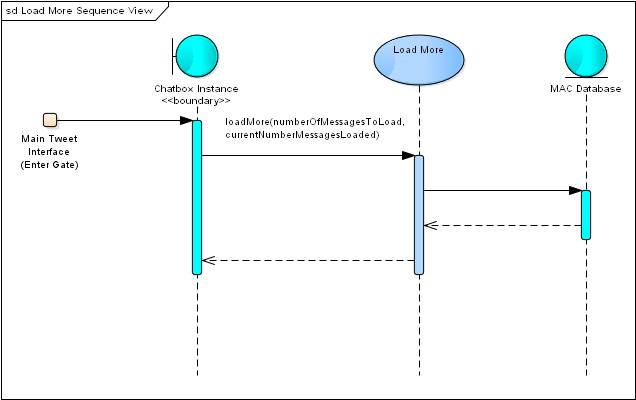


Figure 18 : Load More Sequence View - Iteration 3

## Database

### Description

1. This shows the description of the database how the tables are maintained and the basic structure.

2. The students have ID from their IUPUI ID.

3. The Tweets are stored in message field and they also have date and time tweeted with the student ID who tweeted it.

4. The course part is handled by CAS and the login, username and password details are stored and retrieved by CAS Authentication.

5. Hence, the sensitive data of usernames and passwords are handled by CAS so security has increased.

### Diagram

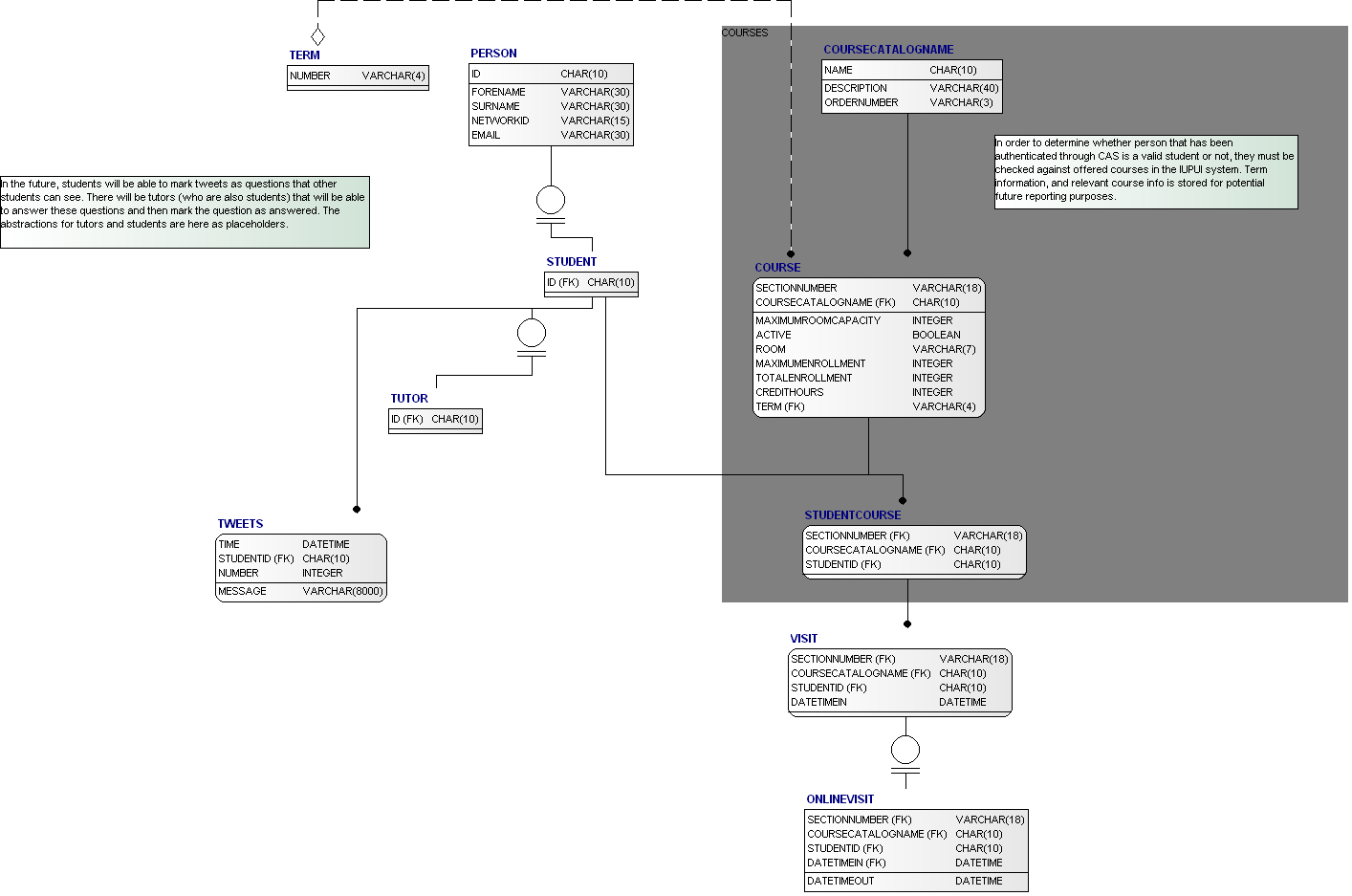


Figure 19 : Database - Iteration 3

# Detailed Software Design Description in Iteration 4

## Use Case Model (General View)

### Diagram

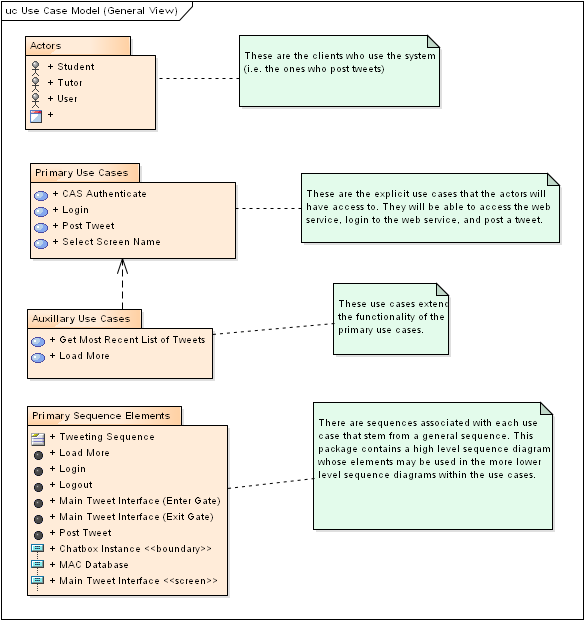


Figure 20 : Use Case Model (General View) - Iteration 4

### Description

1. In Iteration 4, the changes made are that the website is deployed onto Amazon Web Services (AWS) as per the software requirements and the users are now using the website hosted on AWS.

2. The users also have the ability to change screen names which will be reflected in their tweets.

3. But in the main database the tweets are stored as their real user names and the screen names are also tracked to ensure protection against misbehavior.

4. Timestamps are also added as now we track screen names.

## Primary Use Case View

### Diagram

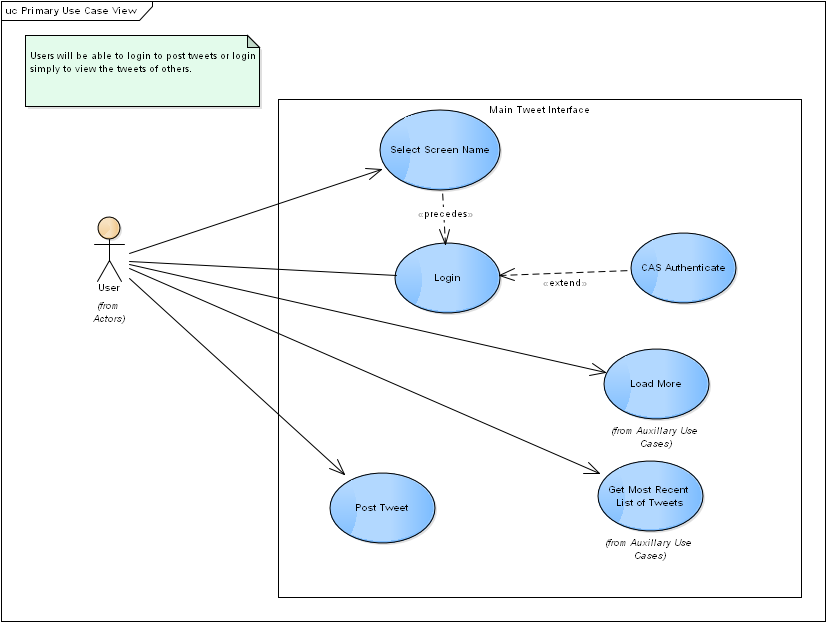


Figure 21 : Primary Use Case View - Iteration 4

### Description

1. The only change in the primary use case is that the select screen name element is introduced and the user can pick his screen name and tweet if he does not want to use the IUPUI user name.

2. The rest of the functionalities remain the same.

## Primary Use Case Sequence (General View)

### Diagram

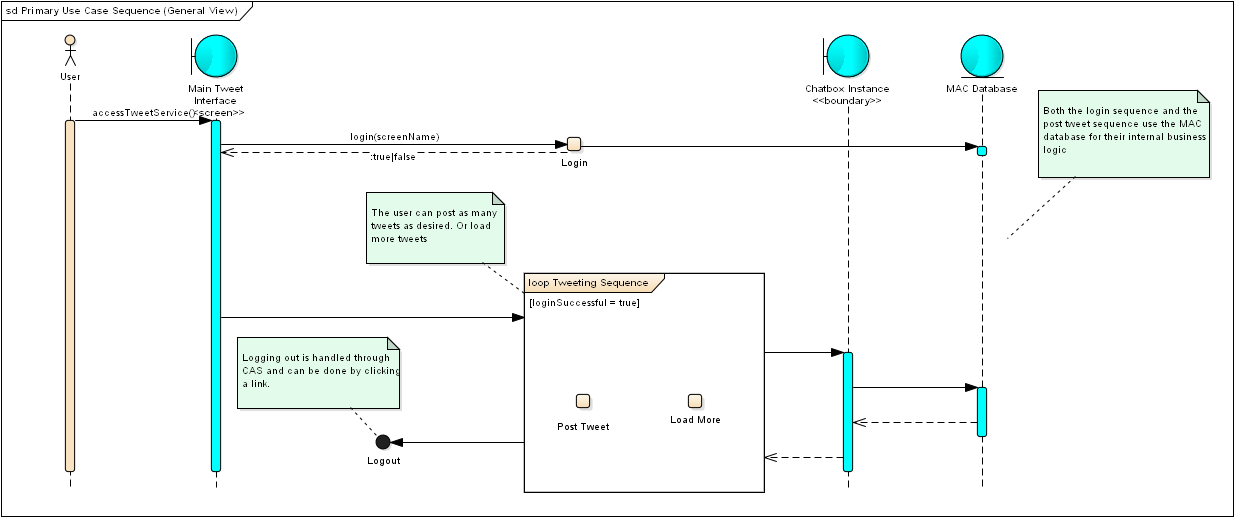


Figure 22 : Primary Use Case Sequence (General View) - Iteration 4

### Description

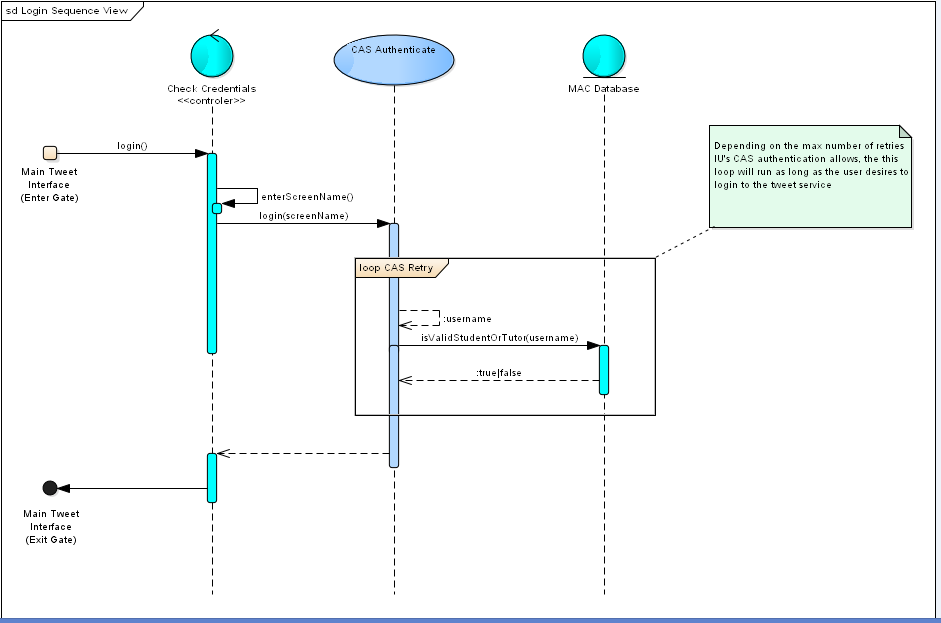
1. In this the user can choose a screen name of his choice and then login and post messages.

2. But it will show his screen name and not the IUPUI user name.

3. The credentials are appropriately checked with the database using CAS authentication.

## Login Sequence View

### Diagram:



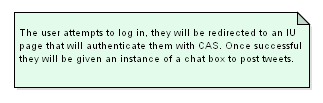


Figure 23 : Login Sequence View - Iteration 4

### Description

1. The user goes to the main tweet interface and then chooses a user name and tries to log in using IUPUI credentials at the CAS authentication.

2.If successful, he will be able to tweet and his screen name will appear.

3. If unsuccessful, he will be allowed to retry as many times as CAS authentication allows.

## Load More Sequence View

### Diagram

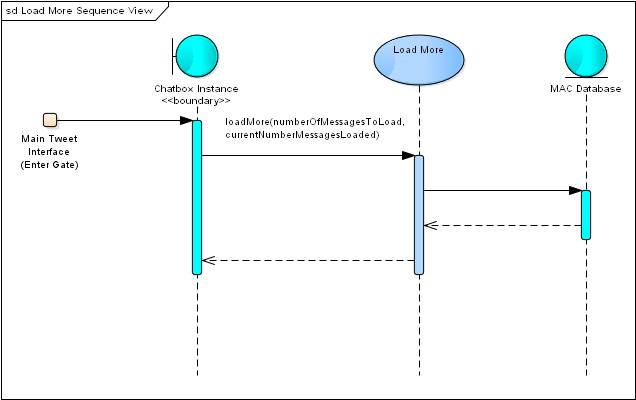


Figure 24 : Load More Sequence View - Iteration 4

### Description

In this, when the user selects the load more option it will load more previous tweets and their associated screen names.

## Database

### Diagram

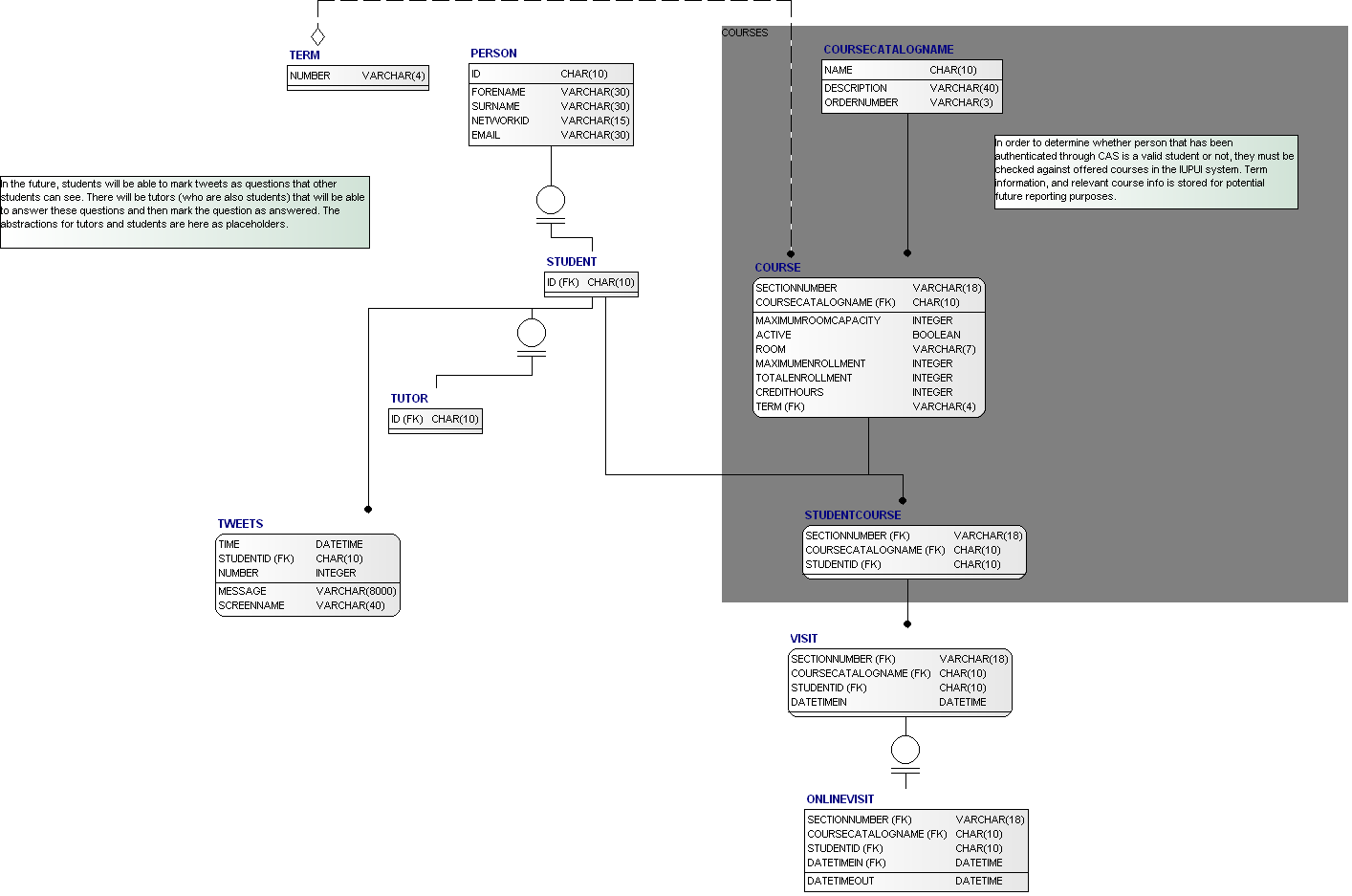


Figure 25 : Database - Iteration 4

### Description

1. The database structure remains the same and it is deployed on AWS.

2. The screen name is added to the tweets table and hence when asked is retrieved from the table or on being created is stored.

3. The timestamps associated with the tweets are also stored as per the screen names.

# Software Architecture

Three styles of software have been used in our software. They are as follows:

## Client-Server Architecture

The software has a client server style with 3 layers i.e., the presentation layer, business logic layer and database layer.

### Presentation Layer:

It is the GUI of the software system. The client or the user interacts with it to perform his tweeting activities with the server.

### Business Logic Layer:

It is the middle layer where the processing engine works for querying to the database layer and retrieving the results and putting it on the application GUI.

### Database Layer:

This is where the data resides and the business logic layer queries it depending upon the instructions provided by the GUI or presentation layer.

## Repository Architecture

The software is centered around data that is tweeting information which it does by sending tweets, loading tweets and storing the tweets. Hence it follows a repository architecture.

## Parallel Communicating Processes Architecture

When a user logs in, a separate session is created for that user. Then as per the environment of the software usage in parallel the users get updates when others add a tweet and the user can do his own operations. Hence, it is a parallel communicating processes architecture.

# References

1. *Software Engineering Modern Approaches Second Edition* Eric J. Braude and Michael E. Bernstein, Wiley Publications.

2. B. W. Boehm, J. R. Brown, M. Lipow, Quantitative Evaluation of Software Quality, TRW Systems and Energy Group, (1976).