Massmine for the Masses

Project Team: Morgan McIntyre, Logan Hornbuckle, Patricia Tanzer, Joshua Moore

Table of Contents

[**1. Project Definition**](#_f6pns9jm1fk3) **4**

[1.1 Why](#_wckxeknchnp2) 4

[1.2 What](#_8pqoao6zguhn) 4

[1.3 How](#_ac67agwzegxq) 4

[**2. Project Requirements**](#_s8tvbcpndfdn) **4**

[2.1 Functional](#_nut9bfa4x5r4) 4

[2.2 Usability](#_p6102w55pc12) 4

[2.2.1 User interface](#_cm0uglhdzq4d) 4

[2.2.2 Performance](#_obcm4tqhdre9) 5

[2.3 System](#_oc4t3wniflsg) 5

[2.3.1 Hardware](#_683g7pjlevav) 5

[2.3.2 Software](#_nc8gzajo58wn) 5

[2.3.3 Database](#_pi0zjke0ru85) 5

[2.4 Security](#_xzvt94xc9fhb) 5

[**3. Project Specification**](#_eksquogfbmju) **6**

[3.1 Focus](#_rg433nvb9yv) 6

[3.2 Domain](#_hmz5zpc4eyke) 6

[3.3 Area](#_y5trxio2ijz9) 6

[3.4 Libraries](#_jked61g3emcr) 6

[3.5 Development Environment](#_13rpc1v4zvqk) 6

[3.6 Framework](#_8uisdeskcnpl) 6

[3.7 Platform](#_mzvwrs9cvj6c) 6

[3.8 Genre](#_5y6p0zahmzml) 7

[**4. System – Design Perspective**](#_a5j4ixnmnpey) **7**

[4.1 Identify subsystems – design point of view](#_3umvy5vsdveo) 7

[4.2 Sub-System Communication (Diagram and Description)](#_ysk0sbffi77) 8

[4.3 Entity Relationship Model (E-R Model)](#_3wlevbyc0hrn) 8

[4.4 Overall operation - System Model](#_3y7zpqrcr1im) 8

[**5. System – Analysis Perspective**](#_lnwvi8acowhv) **8**

[5.1 Identify subsystems – analysis point of view](#_bxo700v8xz0t) 8

[Subsystem 1: Database Management](#_w51e8ezcp18g) 8

[Subsystem 2: GUI for client](#_r9ulzf15cczq) 8

[Subsystem 3: Website-database interaction](#_nyowuu4h95zu) 8

[5.2 System (Tables and Description)](#_bj18u0w7fth6) 8

[5.2.1 Data analysis](#_it8ed9kqjnr) 8

[5.2.2 Process models](#_at9axl9vgcdb) 9

[5.3 Algorithm Analysis](#_933hvher64ls) 9

[5.3.1 Big - O analysis of overall System and Sub-Systems](#_wt06g8nan9n8) 9

[**6. Project Scrum Report**](#_vlc1r9u9hvb9) **9**

[6.1 Product Backlog (Table / Diagram)](#_mcoqhh8a85vm) 9

[6.2 Sprint Backlog (Table / Diagram)](#_jj6jdkr9qw9u) 9

[6.3 Burndown Chart](#_vx7yl22u7qwn) 10

[**7. Subsystems**](#_dbbrecckf7l2) **10**

[7.1 Subsystem 1 – Name 1 - Individual responsibility](#_lwdx0mmrvoty) 10

[7.2 Subsystem 2 – Name 2 - Individual responsibility](#_qwg9ggqud3v0) 11

[7.3 Subsystem 3 – Name 3 - Individual responsibility](#_hqjuk6py8zx) 11

[7.4 Subsystem 4 – Name 4 - Individual responsibility](#_3ixyol1ashjz) 12

[**8. Complete System**](#_hrqicvy4qa2o) **12**

# 

# 

# **1. Project Definition**

## 1.1 Why

Massmine ([www.massmine.org](https://www.massmine.org/)) is a command-line tool useful in the data analysis field in that it can pull massive amounts of data from sources like Twitter and give them to the user for further analysis. However, because it is a command-line tool in the Linux operating system, not every user understands how to work with it. Additionally, it produces that data in a form which is difficult to parse for the non-technically oriented. This makes massmine very difficult to work with, despite its usefulness.

## **1.2 What**

The goal of this project it to simplify this process for the average user by creating a web application that allows a user to create queries for Twitter data in a simple, easily understandable user interface. The data will also be presented in a simple user interface to allow for basic analysis without the user ever having to directly interact with it.

## 1.3 How

The web application will take query information from the user such as search terms and time periods, and send this information to a massmine instance running on the host server. The data retrieved from this massmine instance will be parsed and stored in a database which can then be accessed at a later date through the web application.

# 2. Project Requirements

## 2.1 Functional

2.1.1Web application and associated databases to:

* + 1. Store user information (username, password, 0 to many Twitter authorization keys)
    2. Store collected Twitter data (tweet id, author, content, etc.)
    3. Create and update new users (change password,
    4. Allow users to gather Twitter data. The mass of data collected by a query is a ‘study’.
    5. Allow users to perform basic analysis on this Twitter data.

## 2.2 Usability

### 2.2.1 User interface

* + 1. Login interface: should allow the creation of accounts, as well as signing in. Optional but useful: ‘forgot password’.
    2. User account interface: Allow user to enter or change their Twitter authentication code. Optional but useful: ‘change password’.
    3. Study interface: Lists the studies that a user has access to and allows them to open Analysis interfaces on those studies. Another link allows them to create a new study or remove an old one. Optional but useful: Allow renaming of studies and inviting/removing additional users from viewing a study.
    4. Collection interface: Allow the user to create queries or ‘studies’.
    5. Analysis interface: Allow the user to run basic analyses, such as aggregate by hour, on a study. (Optional: allow user to export analysis for presentation, etc).

### 2.2.2 Performance

* + 1. The user should not notice any significant time delay when running analyses on data already gathered. Depending on the query, data likely will not be available immediately after a query is given, but this should not impact the user from signing out, back in, or running additional queries and analyses in the meanwhile. A collection database will keep past queries to maintain efficient analysis.
    2. Multiple uses should be able to use the web-app at once. Up to 50 accounts expected.

## 2.3 System

### 2.3.1 Hardware

* + 1. A host Linux server with undefined storage resources in the UNCG library system.

### 2.3.2 Software

* + 1. Web application, likely displayed in Javascript and HTML.
    2. Additional modules in Python or similar languages.

### 2.3.3 Database

* + 1. MongoDB

## 2.4 Security

* 1. At-rest encryption of user authentication keys, usernames, and Twitter data. Passwords used to sign in to the account will be salted and hashed. The web application will be accessed through an https:// page.
  2. The framework used, Django, has built-in protections and security features. Django guards against cross site scripting, cross site request forgery, SQL injection, and clickjacking.
  3. Additional measures to be taken include input validation on client and server side

# 3. Project Specification

## 3.1 Focus

* + Develop an open source, web-based application to collect and analyze social network data using MassMine.

## 3.2 Domain

* + The intended user for this application would be academic researchers and universities that may host their own version of this project. This application should appeal to those interested in social media research that lack the necessary command-line skills to fully utilize massmine.

## 3.3 Area

* + Big data/data analysis - This project will involve large sets of metadata that must be parsed through and organized efficiently.
  + Database management - There will be several databases used and maintained within the scope of this project.
  + Web application/web application security

## 3.4 Libraries

* + The following are examples of Python libraries that will be used: shell\_plus, clean\_pyc, json encoding and decoding, create\_template tags, describe\_form, notes

## 3.5 Development Environment

* + Pycharm: A dedicated python and django IDE.
  + Virtual environments for testing
  + Linux host (virtual machine)

## 3.6 Framework

* + Django, a high-level Python web framework. This framework is scalable and secure.
  + Mongodb is our framework for the databases. It is a NoSQL based database system that works well with pymongo for scripting.

## 3.7 Platform

* + Web application meant for desktop access. This platform allows for user usage on any operating system, although the server should be running a Linux distribution to run the Massmine tool.

## 3.8 Genre

* + Web application layered over the command-line tool Massmine.

# **4. System – Design Perspective**

## 4.1 Identify subsystems – design point of view

* + Database management
    - User database:
      * Name: user\_database
      * Collections:
        + user\_collection:

username

encrypted\_password

* + - * + oauth\_collection:

username (references user\_collection username)

oauth\_key

* + - * + study\_collection:

username (references username in user\_collection)

study\_id

* + - * Accessed by: login and user management page
    - Tweet database:
      * Name: twitterDB
      * Collections:
        + tweetInfo:

created\_at

id

id\_str

text

truncated

entities

hashtags

symbols

usermentions

urls

metadata

iso\_language\_code

result\_type

source

in\_reply\_to\_status\_id

in\_reply\_to\_status\_id\_str

in\_reply\_to\_user\_id

in\_reply\_to\_user\_id\_str

user

id

id\_str

name

screen\_name

location

description

url

entities

url

urls{

url

expanded\_url

display\_url

indicies}

description

urls

protected

followers\_count

friends\_count

listed\_count

created\_at

favorites\_count

utc\_offset

time\_zone

geo\_enabled

verified

statuses\_count

lang

contributors\_enabled

is\_translator

is\_translation\_enabled

profile\_background\_color

profile\_background\_image\_url

profile\_background\_image\_url\_https

profile\_background\_tile

profile\_image\_url

profile\_image\_url\_https

profile\_banner\_url

profile\_link\_color

profile\_sidebar\_border\_color

profile\_sidebar\_fill\_color

profile\_text\_color

profile\_user\_background\_image

has\_extended\_profile

default\_profile

default\_profile\_image

following

follow\_request\_sent

notifications

translator\_type

geo

coordinates

place

contributors

is\_quote\_status

retweet\_count

favorite\_count

retweeted

lang

* + - * + tweet\_study\_index:

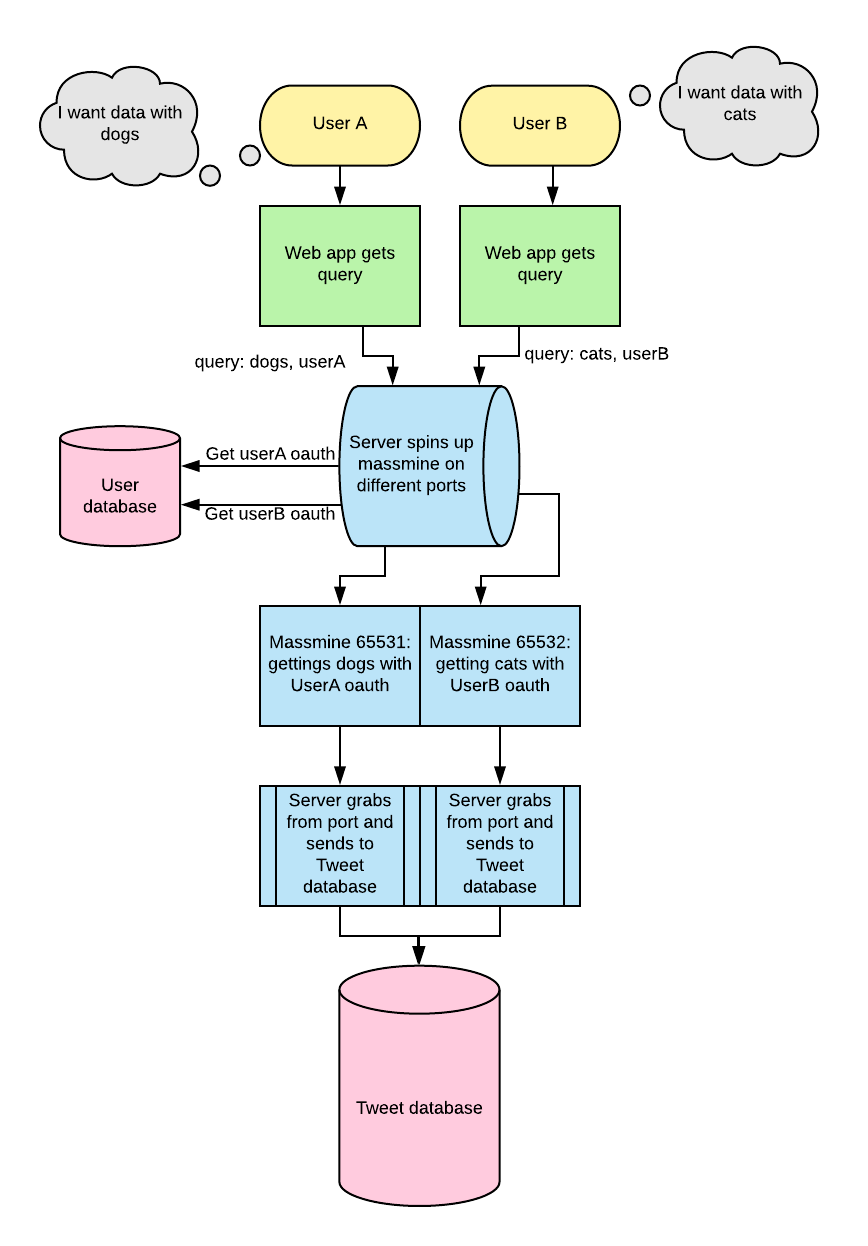
study\_id (references study\_id in study\_collection from user\_database)

id\_str (references id\_str in tweetInfo)

* + - * Accessed by: Query page, analysis page
    - Analysis Database
      * Name: analysis\_database
      * Collections:
        + analysis\_collection

query\_id

query\_str

* + - * Accessed by: Analysis page
  + Analysis
    - Types of analysis commands stored in a database: eg. code for ‘How many tweets from my study contained this phrase between these dates, per day’
    - Website calls this database and gets list of possibly analysis for user to choose from
    - User may choose one of these analyses and feed it desired information (such as a word/phrase, or dates to view), so that the results are graphically displayed
  + Massmine integration:
    - 
  + Web interface - GUI (Screenshots) - Josh

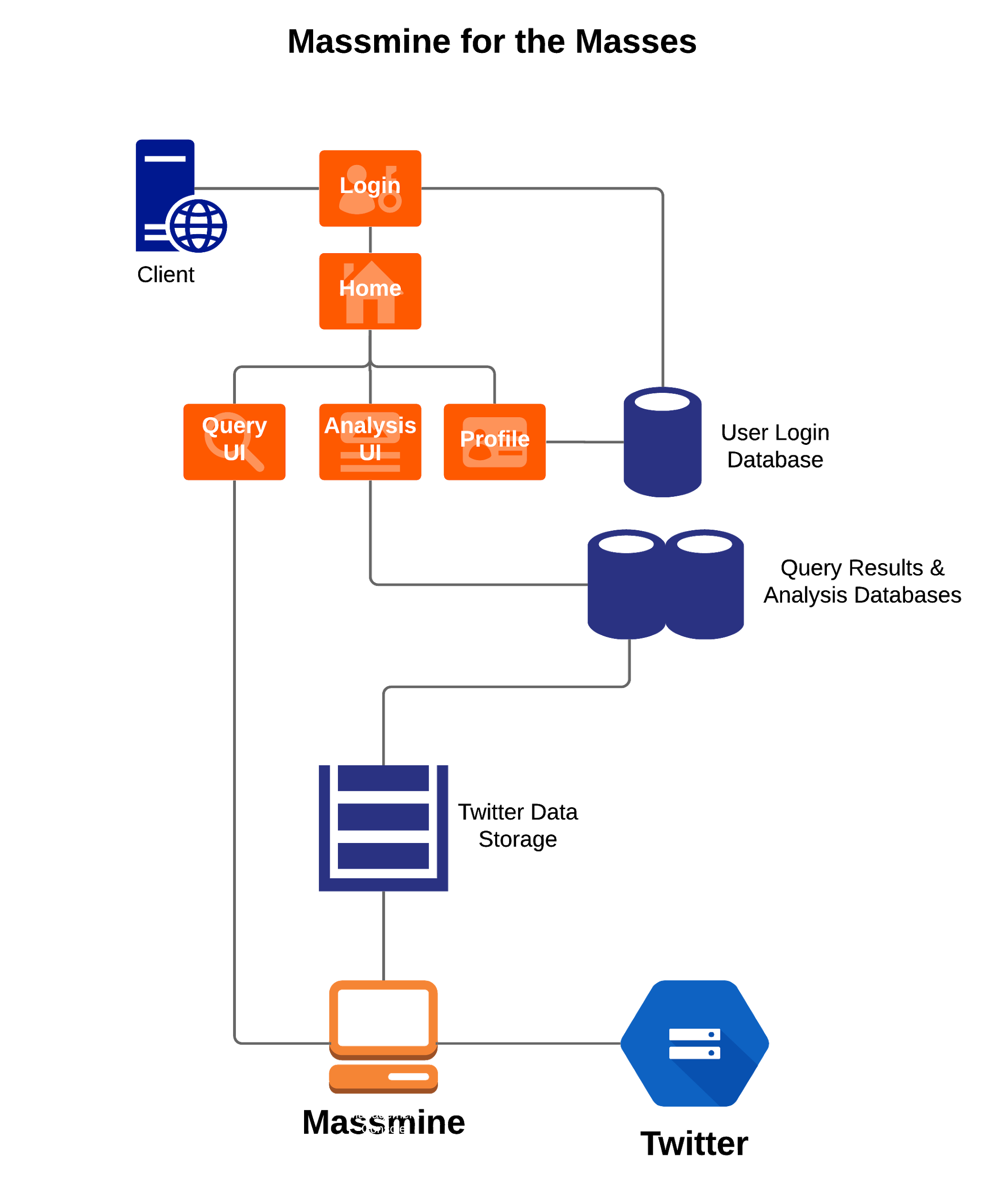
## 4.2 Sub-System Communication (Diagram and Description) - Sequence Diagram -Morgan

* + Controls, Django has built in controls
  + I/O, input from user, output
    - Types of inputs?
      * Password from user
  + DataFlow

## 4.3 Entity Relationship Model - ASK??

* + Example - <https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model>

## 4.4 Overall operation - System Model

* + 

# **5. System – Analysis Perspective**

## 5.1 Identify subsystems – analysis point of view

## Subsystem 1: Database Management

## Subsystem 2: GUI for client

## Subsystem 3: Website-database interaction

* Subsystem 4: Massmine-website-database interaction

## 5.2 System (Tables and Description) - ASK??

### 5.2.1 Data analysis

* + - Data dictionary (Table - Name, Data Type, Description)

### 5.2.2 Process models

## 5.3 Algorithm Analysis - Logan

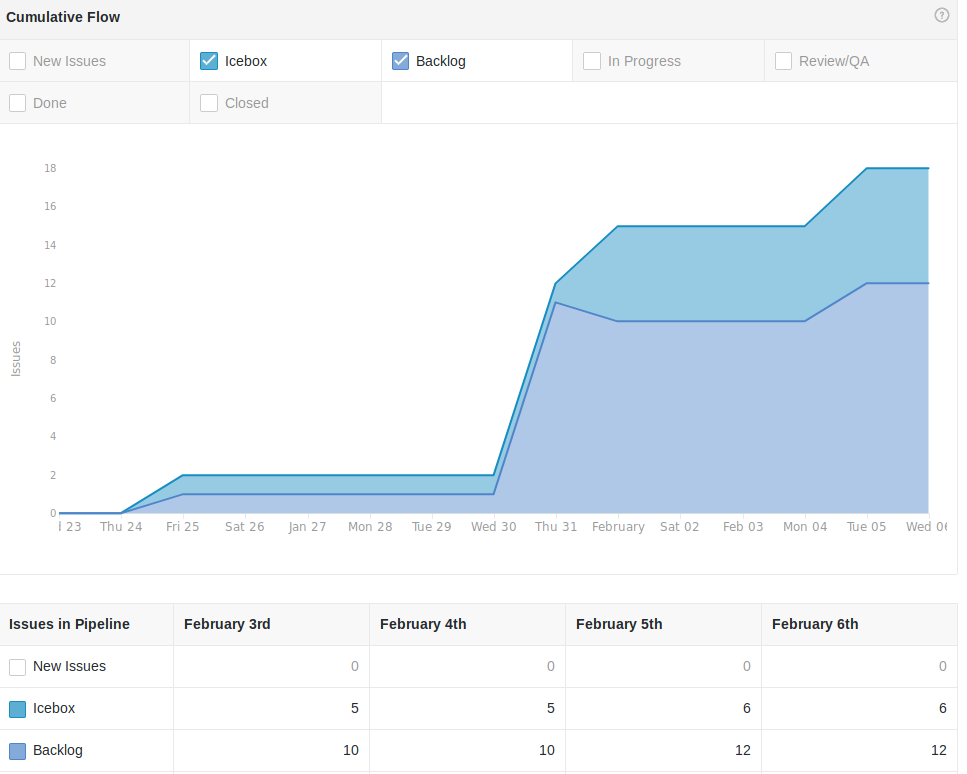
### 5.3.1 Big - O analysis of overall System and Sub-Systems

* Getting a password: O(1)?
* Getting a query
* Doing analysis type X

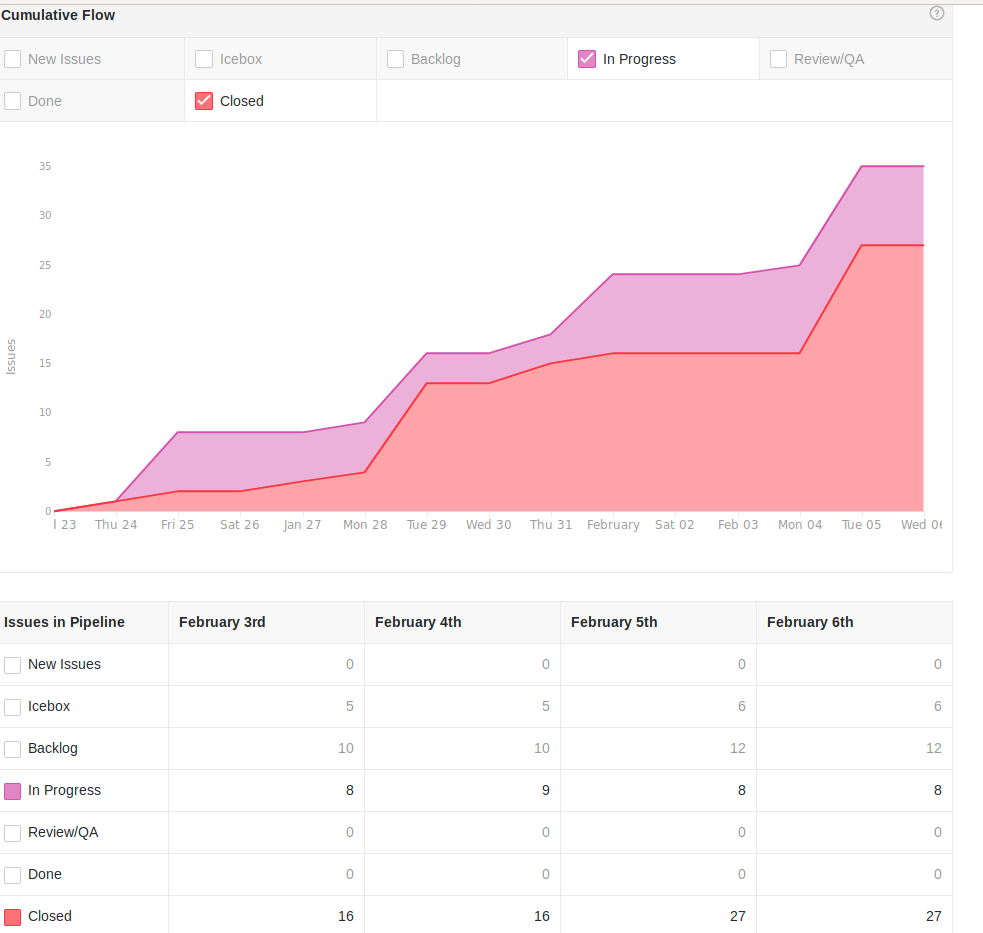
# **6. Project Scrum Report**

## 6.1 Product Backlog (Table / Diagram)

* Jan. 24 - Feb. 06

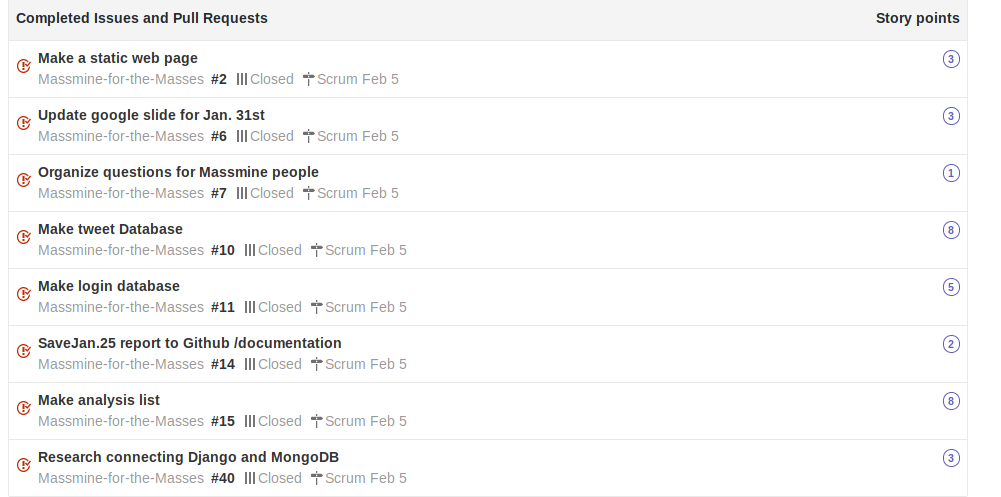
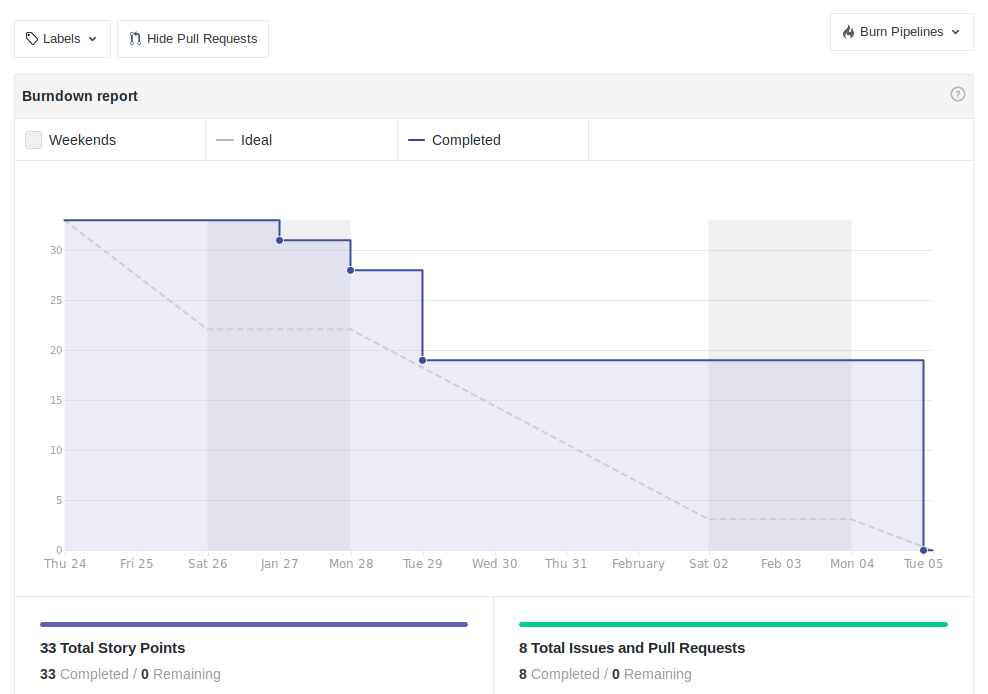


## 6.2 Sprint Backlog (Table / Diagram)

* Jan. 24-Feb.06 

## 6.3 Burndown Chart

* For Scrum Feb 5:



# **7. Subsystems**

## 

## **7.1 Subsystem 1** – Name 1 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams (eg if a UI, do a use-case)
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

## **7.2 Subsystem 2** – Name 2 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

## **7.3 Subsystem 3** – Name 3 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

## **7.4 Subsystem 4** – Name 4 - *Individual responsibility*

* Initial design and model
  + Illustrate with class, use-case, UML, sequence ..... diagrams
  + Design choices
* Data dictionary
* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* Scrum Backlog (Product and Sprint - Link to Section 6)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

# **8. Complete System**

* Final software/hardware product
* Source code and user manual – screenshots as needed - Technical report
  + Github Link
* Evaluation by client and instructor
* Team Member Descriptions

## 