# **Skitter Documentation**

Joseph Bartelmo <jfb3657@g.rit.edu>

Christopher Tran <ctt1414@g.rit.edu>

# **Table of Contents**

[**Skitter Documentation**](#_lsrh3drh9djw) **1**

[**Table of Contents**](#_o6r2keouvct2) **1**

[**Development Procedures**](#_qskj59xhjuad) **3**

[Github Location: https://github.com/christophert/skitter/](#_gjbitmmh6n3k) 3

[**Documentation Procedure**](#_p44kx3f1i2dn) **3**

[**Feature List**](#_mahz6mybxoj6) **5**

[Server Structure](#_1nl8xx9d3tu6) 5

[Authentication](#_xr6hh8w68z2j) 5

[Tutorial and Profiling](#_selaan6ch4u6) 5

[Posting/Viewing Skits](#_gligvzrcumot) 5

[Advanced Skits Features](#_xqzngeq2ucq2) 5

[Following Users](#_fwtbx7d6r6ln) 6

[Replying to other User’s Skits](#_hv832vp7q80u) 6

[**UML Full Diagrams**](#_yp2wd5pwihjz) **7**

[Network Topology](#_dlc2u5muycrb) 7

[Database Schema (Entity Relationship Model)](#_33j9lvuff3fj) 7

[Use Cases](#_ybj0ncs94buj) 8

# **Development Procedures**

**As a developer I would like to layout my development path so that I can more successfully and quickly develop my application**

Through the use of this project, we will be using github. This will enable us to enforce agile methodology and develop in the most efficient source controlled manner.

##### **Github Location**: <https://github.com/christophert/skitter/>

We will be following standard gitflow. When a feature is being implemented a new branch will be opened, and all development work for the new feature will be put into the new branch. Once all changes are completed, the branch has implemented tests for the new feature, and the branch has been manually verified by all team members, a pull request will be opened and it will be merged into the master branch. To read more about this flow, visit this url: <https://guides.github.com/introduction/flow/>

Travis CI is setup on the github repository such that with every commit to any branch the automated suite of tests will automatically launch. This can be accessed through the github repository.

**How did you break up your projects and what are the security ramifications?**

Our project is broken up into six parts in following the microservice architecture and we will be utilizing Java, HTML5/CSS, PHP, NodeJS, Python, and Ruby on Rails. The security ramifications of this project markup is that we will be dealing with many different codebases and the different technologies may have their independent security vulnerabilities that have to be compensated for and protected properly.

**How did you choose to break down your milestones into various issues (tasks)?**

For the authentication milestone, we chose to break it down to logging in and out, registration, LDAP login, unit tests, and user storage.

For the Homepage milestone, we broke it down to implementation of our design mockup and Selenium tests for different platforms.

For the settings milestone, we will focus on data retrieval from the database to be returned to our frontend, then modifications of the user’s email, display name, and profile picture. We’ll then implement unit tests for our settings page to ensure that the routes are secure and functional.

For the Skits, we will first focus on implementing ElasticSearch as our datastore. Next, we will implement the routes for getting, deleting, adding skits from the ElasticSearch instance and tests for each route as well in NodeJS. We will also implement replying to skits with Ruby on Rails and its associated tests. Finally, we’ll implement authentication on each of those routes once they have been properly implemented.

For profiles, we will implement following and unfollowing of other users in Python using the Flask framework and interactions with the existing MySQL datastore.

**How do you ensure that after each issue/milestone that security has been verified? How would you identify such issues in an ideal environment?**

To ensure that security has been verified after each issue/milestone, we will run a fuzzer on our application inputs and verify that our application is not vulnerable to input attacks. We will also implement authorization checks on all API routes in our tests to ensure that data cannot be accessed without the proper authentication.

# **Documentation Procedure**

**As a developer I would like to have documentation about how to implement the application**

The primary means for editing this documentation is through google drive, a shared document editing platform. The viewable link will be located [here](https://docs.google.com/document/d/1BL7dvBZ8WKVV2HUQVZ2wPw93BOA4xp43rcUtG96cHPo/edit?usp=sharing). If you wish to edit the documentation then you will have to contact one of the people on the development team, located on the title page of this document.

In addition, all important documentation will be located on the github repository.

**As a developer I would like to use test-driven development in order to write my code, as a result test cases need to be generated.**

Test driven development is the concept of developing the code around tests that are written prior to the main development of the code. The intended test frameworks that will be used are:

|  |  |
| --- | --- |
| **Test Framework** | **Purpose** |
| Selenium | User interface testing |
| JMeter | Load balance testing and stress testing |
| JUnit | Backend testing (both database testing and server-side tests) |
| Mocha | Presentation layer testing (NodeJS) |

The tests themselves will be behavioral driven and will require confirmation from all developers working on the project to guarantee that the test tests the intended feature component.

**As a developer I would like all of the various infrastructures to be in a state where they are ready to host web applications.**

The various infrastructures are implemented on a server that is hosted through a virtual private network located on RIT’s campus. It is hosted on one server with ESXi installed. To access this server you will need the vpn ip address, a certificate, and a username and password to be setup on the ESXi instance.

**Notes on UML:**

UML (Unified Modeling Language) diagrams will be broken out into multiple parts throughout this document. Any applicable UML for the current topic will be included below the topic description to assist with development. At the very end of this document there will be a set of fully completed UML Diagrams that detail the end to end system. The intention is to help developers and readers understand how this system is composed.

# **Feature List**

##### **Server Structure**

As a developer I would like to have a resilient front end to balance the load for my various servers

As a developer I would like to be able to test the load my servers can handle in a reproducible way.

As a developer I would like to use test-driven development in order to write my code, as a result test cases need to be generated.

##### **Authentication**

As a user I’d like to be able to login and be presented with a website that presents all the features (non-functional yet) and information (placeholders) needed so that I can use Skitter on all popular browsers.

##### **Tutorial and Profiling**

As a user I’d like the experience to be as seamless as possible so that I can easily understand and leverage the features of the platform.

A user wants to be able to change their display name, email, and profile image.

##### **Posting/Viewing Skits**

As a user I would like to be able to add Skits (tweets) so that people can see what I am up to

##### **Advanced Skits Features**

As a user I would like to be able to remove Skits in case I make a mistake while posting

As a user I would like a centralist of all my Skits so I can see what I’ve said in the past.

Users would like to be able to search for other users so they can read their Skits

##### **Following Users**

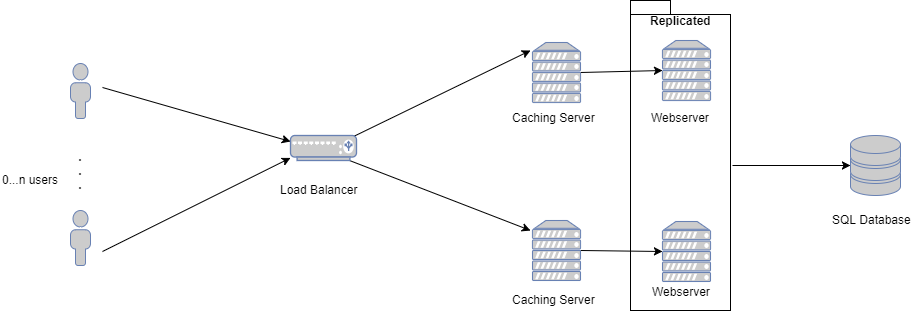
As a user I would like to be able to follow other users, adding the followed user’s Skits to my central list.

##### **Replying to other User’s Skits**

Users would like to be able to reply to other user’s skits and have the replies show up on their central list.

# **UML Full Diagrams**

##### **Network Topology**

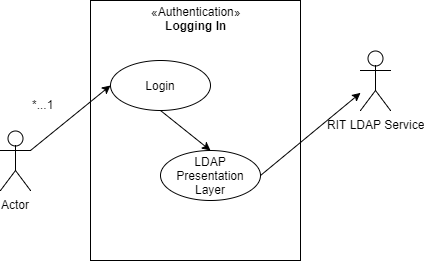


##### **Database Schema (Entity Relationship Model)**

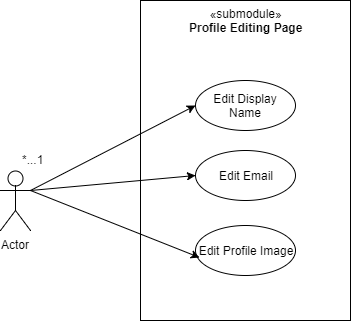
##### 

##### **Use Cases**

1. As a user I’d like to be able to login and be presented with a website that presents all the features (non-functional yet) and information (placeholders) needed so that I can use Skitter on all popular browsers.



2. A user wants to be able to change their display name, email, and profile image



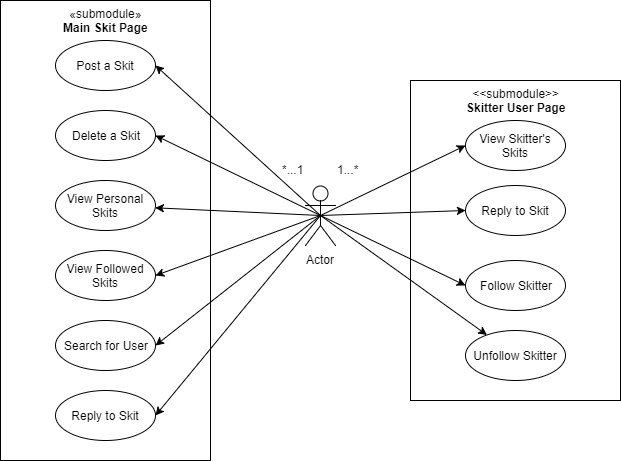
3. As a user I would like to be able to add Skits (tweets) so that people can see what I am up to

4. As a user I would like to be able to remove Skits in case I make a mistake while posting

5. As a user I would like a centralist of all my Skits so I can see what I’ve said in the past.

6. Users would like to be able to search for other users so they can read their SkitsAs a user I would like to be able to follow other users, adding the followed user’s Skits to my central list.

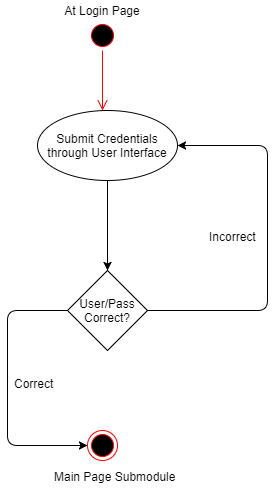
7. Users would like to be able to reply to other user’s skits and have the replies show up on their central list.

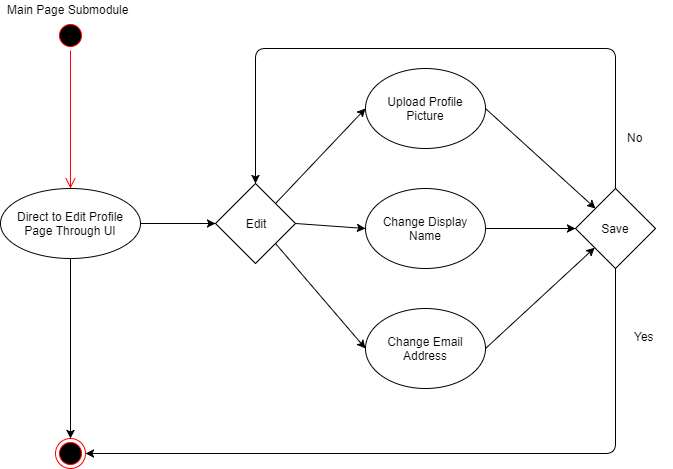


##### **Flow Graphs**

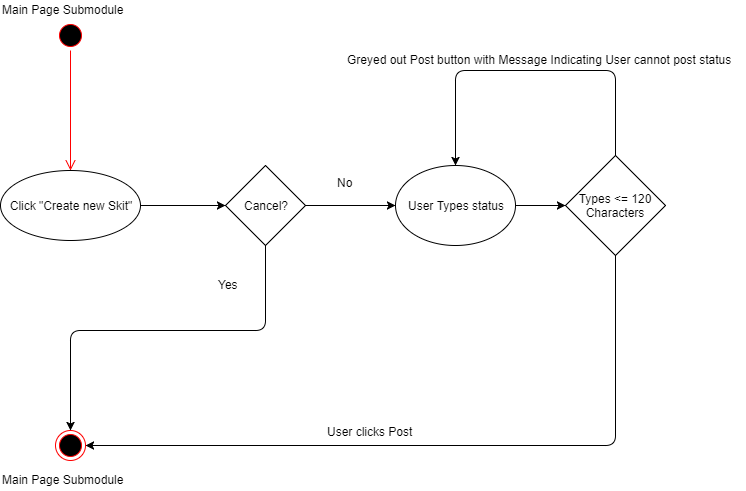
Note that the below flow graphs are directly associated with the use case numbers above.

1. As a user I’d like to be able to login and be presented with a website that presents all the features (non-functional yet) and information (placeholders) needed so that I can use Skitter on all popular browsers.

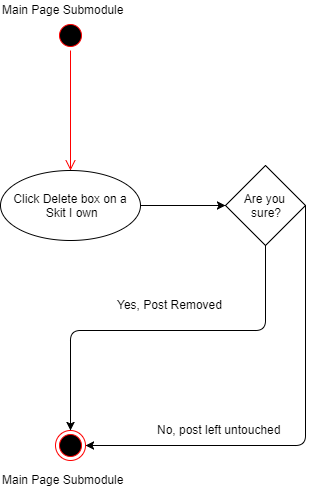


2. A user wants to be able to change their display name, email, and profile image

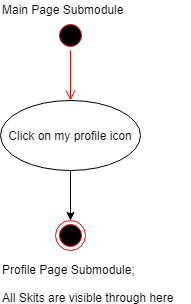
3. As a user I would like to be able to add Skits (tweets) so that people can see what I am up to



4. As a user I would like to be able to remove Skits in case I make a mistake while posting

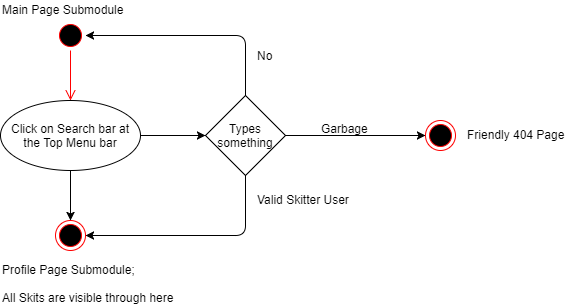


5. As a user I would like a centralist of all my Skits so I can see what I’ve said in the past.



6. Users would like to be able to search for other users so they can read their SkitsAs a user I

would like to be able to follow other users, adding the followed user’s Skits to my central list.



7. Users would like to be able to reply to other user’s skits and have the replies show up on their central list.

