Setup

The following instructions will walk you through the process necessary to setup FailSafe on a fresh server. If you have any questions, feel free to email David Chou at david.p.chou@gmail.com or any other member of the FailSafe Development team.

Flask

The entire FailSafe project runs on the web using the Python Flask framework. Because of this, downloading Flask is a prerequisite to setting up FailSafe. Instructions for downloading Flask can be found here. On Ubuntu 14, this is simple. Just run the following:

```
sudo easy_install pip                         // installs pip

sudo apt-get install python-virtualenv        // installs virtualenv

pip install Flask                             // installs Flask
```

FailSafe

To download FailSafe, you need only to clone the FailSafe Github repository into the desired directory. To do so, naturally, you'll need to install Git on your server.

```
sudo apt-get update

sudo apt-get install git
```

From there, just clone FailSafe into any directory:

```
sudo git clone git@github.com:duke-compsci408-fall2014/FailSafe.git [Directory
Name]
```

Server

The initial FailSafe production server was built on Vanilla Ubuntu 14. Because we used Python's Flask framework to develop our website, we added Apache/Mod WSGI for compatibility. The following below is very helpful in teaching developers how to get Flask/WSGI up and running:

Deploying Flask Apps with Apache and Mod WSGI

A basic, but incomplete, summary is below. The startup is to first download the necessary packages using the following commands:

```
sudo apt-get update

sudo apt-get install libapache2-mod-wsgi
```

Then you want to make sure you setup your directory properly. You want to then download the flask deployment starter kit:

```
cd ~ wget
https://beagle.whoi.edu/redmine/attachments/download/579/flask_deployment_starte
r.tar.gz

tar zxvf flask_deployment_starter.tar.gz
```

Afterwards, you ought to develop the wsgi files in the proper directory. This is a good example.

```
import sys sys.path.insert(0,'/home/jsmith/public_html/apps/flasktest') from
flasktest1 import app as application
```

You want to make sure that you create an appropriate site configuration file that will enable your website to appear. This will be located in /etc/apache2/sites-available. Again, you should check the tutorial link listed above as that is much more in depth.

SSL

In order to setup proper HTTPS configuration on your website, you will need to make sure that you have ordered an SSL certificate for your website so that it is confirmed to be secure. There should be resources at your institution to do so, but there are also a variety of resources otherwise available.

Shib

In order to enable Shibboleth protection on your webpage, you will need to route particular directories to protect various links. The following links to Duke's Shibboleth registration page (to validate your website as legitimate as well as setting up the information you will need from each user). Note that this will not work if you do not have access to Duke's networks.

Duke Shib Page

Another thing you will have to do is run whatever script is available at your institution to setup the proper xml files that enable shibboleth wrapping around your particular file. To see an example of Duke's Shibboleth setup process, you can use this as an example – Shibboleth Tools.

Databases

The last thing that you'll have to do is setup up the databases that are required by FailSafe. We'll need to install MySQL to start:

```
sudo apt-get install mysql-server
```

When this runs, you'll have to setup a password for *root.* To make it work flawlessly with the code in GitHub, set the password to *efasliaf* (Failsafe backwards). Otherwise, set your password of choice, and edit the config.py file in the FailSafe directory to replace lines 6 and 14 with:

```
dir_app.config['MYSQL_DATABASE_PASSWORD'] = [YOUR PASSWORD]
```

Once MySql is installed, you'll have to boot MySql and set up the FailSafe user and the primary Databases using the following sequences of commands:

```
mysql –u root –p

CREATE USER 'failsafe'@'colab-sbx-245.oit.duke.edu' IDENTIFIED BY '[YOUR
PASSWORD]'                             // creates the FailSafe user

CREATE database calendar;              // creates the database for the calendar

USE calendar;

CREATE TABLE schedule ( Day DATE NOT NULL, Faculty VARCHAR(50) NOT NULL, Fellow
VARCHAR(50) NOT NULL, RN1 VARCHAR(50) NOT NULL, RN2 VARCHAR(50) NOT NULL, Tech1
VARCHAR(50) NOT NULL, Tech2 VARCHAR(50) NOT NULL, PRIMARY KEY(Day));

CREATE TABLE substitutions ( SubID int NOT NULL AUTO_INCREMENT, StartTime
DATETIME NOT NULL, EndTime DATETIME NOT NULL, Role VARCHAR(50), SubName
VARCHAR(50), PRIMARY KEY(SubID));

CREATE database directory;             // creates the database for the directory

USE directory;

CREATE TABLE tblUser ( UserID INT NOT NULL AUTO_INCREMENT, Role VARCHAR(50) NOT
NULL, IsAdministrator BOOLEAN, FirstName VARCHAR(50) NOT NULL, LastName
VARCHAR(50) NOT NULL, CellPhone VARCHAR(14) NOT NULL, HomePhone VARCHAR(14) NOT
NULL, PagerNumber VARCHAR(14) NOT NULL, NetID VARCHAR(10), PRIMARY KEY
(UserID));
```

Now all your databases should be set up – feel free to test these at your leisure. The schemas above are also detailed in the sql_files directory in the root of the FailSafe project.

Test!

Now everything is set up – you should be able to just hit the failsafe.colab.duke.edu URL in order to access the FailSafe page! If this fails, cd into the FailSafe directory and type

```
python runserver.py
```

And go to the designated port in order to look at a non-production version of FailSafe while you try to work out the bugs.