# Rose-Hulman Undergraduate Mathematics Conference Web Application Software

## User Guide

### INFO

This guide and corresponding software was created and written by Nick Crawford (SE/MA/CS 2013; crawfonw@gmail.com) by request of Dr. Alan Holder. 

# Table of Contents

# Administrator Portal

## Management System

### *Auth Module*

### *Conference Module*

Attendees Submodule

Conferences Submodule

Contactees Submodule

Days Submodule

Pages Submodule

Rooms Submodule

Schedules Submodule

Sessions Submodule

Time Slots Submodule

### *Sites Module*

# Important Actions

## In the Management System

### *Creating a New Conference*

### *Creating Sessions for a Conference*

### *Manually Creating an Attendee*

### *Creating new Website Pages*

### *Creating new Website Links*

## Other Administrator Portal Actions

### *Emailing all Attendees in a Conference*

### *Generating LaTeX Schedule File*

# Setup Information

## Installing the Application from Scratch

### *Linux*

#### Requirements

Latest Apache server
Python 2.7.x
Django 1.5+
Some sort of database MySQL*, PostgreSQL**, SQLite

*Requires MySQLdb Python library
**Requires psycopg2 Python library

#### Installing Required Packages

If deploying on a Rose CSSE server, go bug the admins if some of the above packages are missing. They'll know what to do. Bring cookies if you want it done faster. ;)

If you are deploying on your own server, I recommend using a *nix machine as they play the nicest with this kind of stuff. These directions should work on OS X machines, but they are minor bits and bobs that you need to change depending on the OS version. I have successfully deployed on 10.6-10.8 machines.

If you don't expect this program to store lots of data (i.e. resetting the system after each conference, I recommend using SQLite as it is the easiest to set up.

Installation links:
http://www.python.org/download/
https://code.google.com/p/modwsgi/

Once Python is downloaded open a fresh terminal and type

```
sudo easy_install pip
```

Once pip is installed, you can now install Django

```
sudo pip install django
```

If you are using MySQL or PostgreSQL then install them accordingly

```
sudo pip install MySQLdb
sudo pip install psycopg2
```

To deploy you will then need to install `mod_wsgi`.

#### Installing the App

Once all software is installed, pick a place where you would like to have the application's code run. The first installation on Rose's AFS was located at

```
/afs/csse/users/projects/mathconf
```

on either Derek or Clive.

To install, either:
Download this file and unzip

https://github.com/crawfonw/RHUMC/archive/master.zip

Or navigate to the folder where you want the code to live and run this command in the terminal

```
git clone https://github.com/crawfonw/RHUMC.git
```

Note: The latest stable versions of the code will be in the master branch. If you want to check out experimental builds you will have to look at the other branches. If you do not know what this means, just ignore it.

That's all there is to it! Well, almost, we still need to add some files manually. I recommend using the `git clone` method for installation, because it handles updates more easily. (Git might also not be on the server. If you are using afs, then it should be at least on Derek or Clive.)

### Set Up mod_wsgi

We now need to add some files to both your newly-cloned RHUMC application folder and to Apache. First we will set up mod_wsgi.

First create a file named wsgi.py in /path/to/installation/RHITUMC/RHITUMC/. The only other files in that folder should be __init__.py, context_processors.py, and urls.py.

Paste the following into the newly created file:

```
"""
WSGI config for RHITUMC project.

This module contains the WSGI application used by Django's development
server
and any production WSGI deployments. It should expose a module-level
variable
named ``application``. Django's ``runserver`` and ``runfcgi`` commands
discover
this application via the ``WSGI_APPLICATION`` setting.

Usually you will have the standard Django WSGI application here, but it
also
might make sense to replace the whole Django WSGI application with a
custom one
that later delegates to the Django one. For example, you could
introduce WSGI
middleware here, or combine a Django application with an application of
another
framework.

"""
import os, sys

path = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
if path not in sys.path:
    sys.path.append(path)

os.environ.setdefault("DJANGO_SETTINGS_MODULE", "RHITUMC.settings")

# This application object is used by any WSGI server configured to use
this
# file. This includes Django's development server, if the
WSGI_APPLICATION
# setting points here.
from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()
```

Next we will create a file for Apache. **If you are having the CSSE admins run the server, they will need to do this part!** If it does not already exist, please create an `httpd.conf` file. This is located in the root apache installation folder, which varies depending on your OS.

Please paste this into `httpd.conf`. If it already exists **do not** write over it! Simply add this to the top of the file.

```
<VirtualHost IP/HOSTNAME:80>
  ServerName mathconf.csse.rose-
hulman.edu<http://mathconf.csse.rose-hulman.edu/>
  DocumentRoot "ABS_PATH_TO_PROJECT"

  WSGIScriptAlias / ABS_PATH_TO_WSGI.py

  ErrorLog logs/mathconf/error_log
  CustomLog logs/mathconf/access_log common

  <Directory "ABS_PATH_TO_PROJECT">
    <Files wsgi.py>
      Order deny,allow
      Allow from all
    </Files>
  </Directory>
</VirtualHost>
```

The bolded item you will have to change. Here is what each of them means:
`IP/HOSTNAME:` The IP or hostname of the machine running the app.
`ABS_PATH_TO_PROJECT:` The exact path to where the app is being stored. I.e. `/afs/csse/users/projects/mathconf`. The second instance might be different depending on how it is installed.
`ABS_PATH_TO_WSGI:` The exact path to where `wsgi.py` is being stored. Might differ based on installation. I.e. `/Projects/mathconf/web/wsgi.py`.

### Configuring Application Settings

Next we need to go back to where we made the `wsgi.py` file and create a new `settings.py` file. Paste the following into it:

```
# Django settings for RHITUMC project.

DEBUG = False
TEMPLATE_DEBUG = DEBUG
#DEBUG_PROPAGATE_EXCEPTIONS = DEBUG

AUTH_PROFILE_MODULE = 'conference.UserProfile'

ADMINS = (
    ('Nick Crawford', 'crawfonw@gmail.com),
)

MANAGERS = ADMINS

DATABASES = {
```

```
    'default': {
        'ENGINE': 'django.db.backends. ', # Add 'postgresql_psycopg2',
'mysql', 'sqlite3' or 'oracle'.
        'NAME': '',                       # Or path to database file if
using sqlite3.
        'USER': '',                       # Not used with sqlite3.
        'PASSWORD': '',                   # Not used with sqlite3.
        'HOST': '',                       # Set to empty string for
localhost. Not used with sqlite3.
        'PORT': '',                       # Set to empty string for
default. Not used with sqlite3.
    }
}

# Local time zone for this installation. Choices can be found here:
# http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
# although not all choices may be available on all operating systems.
# In a Windows environment this must be set to your system time zone.
TIME_ZONE = 'America/Chicago'

# Language code for this installation. All choices can be found here:
# http://www.i18nguy.com/unicode/language-identifiers.html
LANGUAGE_CODE = 'en-us'

SITE_ID = 1

# If you set this to False, Django will make some optimizations so as
not
# to load the internationalization machinery.
USE_I18N = True

# If you set this to False, Django will not format dates, numbers and
# calendars according to the current locale.
USE_L10N = True

# If you set this to False, Django will not use timezone-aware
datetimes.
USE_TZ = True

# Absolute filesystem path to the directory that will hold user-
uploaded files.
# Example: "/home/media/media.lawrence.com/media/"
MEDIA_ROOT = '/path/to/app/RHUMC/RHITUMC/media/'

# URL that handles the media served from MEDIA_ROOT. Make sure to use a
# trailing slash.
# Examples: "http://media.lawrence.com/media/",
"http://example.com/media/"
MEDIA_URL = ''

# Absolute path to the directory static files should be collected to.
# Don't put anything in this directory yourself; store your static
files
# in apps' "static/" subdirectories and in STATICFILES_DIRS.
# Example: "/home/media/media.lawrence.com/static/"
STATIC_ROOT = ''

# URL prefix for static files.
```

```
# Example: "http://media.lawrence.com/static/"
STATIC_URL = '/static/'

# Additional locations of static files
STATICFILES_DIRS = (
    # Put strings here, like "/home/html/static" or
"C:/www/django/static".
    # Always use forward slashes, even on Windows.
    # Don't forget to use absolute paths, not relative paths.
    '/path/to/app/RHUMC/RHITUMC/static',
)

# List of finder classes that know how to find static files in
# various locations.
STATICFILES_FINDERS = (
    'django.contrib.staticfiles.finders.FileSystemFinder',
    'django.contrib.staticfiles.finders.AppDirectoriesFinder',
#    'django.contrib.staticfiles.finders.DefaultStorageFinder',
)

# Make this unique, and don't share it with anybody.
SECRET_KEY = Please email me for this. It is very important.

TEMPLATE_CONTEXT_PROCESSORS = (
    "django.contrib.auth.context_processors.auth",
    "django.core.context_processors.debug",
    "django.core.context_processors.i18n",
    "django.core.context_processors.media",
    "django.core.context_processors.static",
    "django.core.context_processors.tz",
    "django.contrib.messages.context_processors.messages",
    "RHITUMC.context_processors.project_context_processor",
)

# List of callables that know how to import templates from various
sources.
TEMPLATE_LOADERS = (
    'django.template.loaders.filesystem.Loader',
    'django.template.loaders.app_directories.Loader',
#     'django.template.loaders.eggs.Loader',
)

MIDDLEWARE_CLASSES = (
    'django.middleware.common.CommonMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    # Uncomment the next line for simple clickjacking protection:
    # 'django.middleware.clickjacking.XFrameOptionsMiddleware',
)

ROOT_URLCONF = 'RHITUMC.urls'

# Python dotted path to the WSGI application used by Django's
runserver.
WSGI_APPLICATION = 'RHITUMC.wsgi.application'
```

```
TEMPLATE_DIRS = (
    # Put strings here, like "/home/html/django_templates" or
"C:/www/django/templates".
    # Always use forward slashes, even on Windows.
    # Don't forget to use absolute paths, not relative paths.
    '/path/to/app/RHUMC/RHITUMC/templates',
)

INSTALLED_APPS = (
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.sites',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'django.contrib.admin',
    'django.contrib.admindocs',
    'conference',
)

# A sample logging configuration. The only tangible logging
# performed by this configuration is to send an email to
# the site admins on every HTTP 500 error when DEBUG=False.
# See http://docs.djangoproject.com/en/dev/topics/logging for
# more details on how to customize your logging configuration.
LOGGING = {
    'version': 1,
    'disable_existing_loggers': False,
    'filters': {
        'require_debug_false': {
            '()': 'django.utils.log.RequireDebugFalse'
        }
    },
    'handlers': {
        'mail_admins': {
            'level': 'ERROR',
            'filters': ['require_debug_false'],
            'class': 'django.utils.log.AdminEmailHandler'
        }
    },
    'loggers': {
        'django.request': {
            'handlers': ['mail_admins'],
            'level': 'ERROR',
            'propagate': True,
        },
    }
}
```

Please make sure to change all the bolded items to fit your installation needs. For the folders, if they do not exist then please create them. This should only apply to “media” at most.

Please email me for the secret key. It is an important part of the application for encrypting any important data used by the application. If it is a new installation then you can create your own here http://www.miniwebtool.com/django-secret-key-generator/. Just be aware

than if you ever make a new installation and migrate data then you will need the same secret key for anything encrypted.
For the database, follow what the comments say to fill it out. If you are using SQLite, then make sure the path to the data file is an absolute (full) path (the same way we have been denoting paths in this guide).

### Deployment

Now that you have everything setup correctly, you are almost ready to go. If this is a new installation then you will have to set up the database. If you are using an old database and no update to the code changed how the data is stored, then you should be good to go. Just restart apache (or get the sysadmins to do so), and you should be golden. If you need to set up the database, then we have one or two more steps to do.

If the database is running on the same server as the app (i.e. SQLite) and is a **fresh install**, then you can simply run `python manage.py syncdb` and follow the instructions. If it is running on a different server then you can run `python manage.py sqlall` and copy the output to a file or clipboard and run it in the management tool (phpMyAdmin, etc.).

If the database is not a fresh install and the code has been updated so the database needs an update, then grab the .sql files from the github repository wiki https://github.com/crawfonw/RHUMC/wiki and run them in your management tool.

## Modifying Current Installation

### *Switching Databases*

### *Modifying Email Server*

### *Moving Host Directory*

### *Updating Current Installation*