ECE 458

Development Guide

Lucas Donaldson, Johnny Kumpf, Arthur Schweitzer, Niklas Sjoquist

Project Overview: The purpose of our project is to create a web inventory tracking system for the Duke ECE department. We built our web app using Django with a Postgresql database. The application is running on an Apache webserver on an Ubuntu virtual machine provided by Duke OIT.

Link: https://colab-sbx-379.oit.duke.edu/

High-Level Design: Our project uses the standard Django project architecture, with three applications: home, manager, and administration. Each app has a few main files:

- models.py: Specifies the persistent models which Django translates into Postgresql tables
- views.py: A set of Python functions that take Web requests and return Web responses
- urls.py: The URL declarations for the project; a “table of contents” of the site

These files, in addition to html templates for the different pages, control the behavior of the application.

Front End: We use Django’s built-in html templates to serve html to the client. Within our front-end, we use bootstrap components for many of our tables, forms, buttons, and lists. We also use some in-line javascript code for interactive UI components such as autocomplete fields.

ReST API: We used the Django Rest Framework to implement a ReST API to achieve the same functionality as available in the web application.

Database Schema: The database schema is as follow:

Item(id, item_name, count, model_number, description)
Tag(id, tag)
    Tags have a many-to-many relationship with Items
Request(id, owner, item_id, reason, admin_comment, quantity, status, parent_cart)
    Requests have a many-to-one relationship with Items
    Requests have a many-to-one relationship with Cart_Requests
Cart_Request(id, cart_owner, cart_reason, cart_admin_comment, cart_status)
CustomFieldEntry(id, field_name, is_private, value_type)
    CustomFieldEntries have a one-to-many relationship with CustomShortTextFields, CustomLongTextFields, CustomIntFields, and CustomFloatField
CustomShortTextField(id, parent_item, field_name, field_value)
CustomLongTextField(id, parent_item, field_name, field_value)
CustomIntField(id, parent_item, field_name, field_value)

CustomFloatField(id, parent_item, field_name, field_value)
Log(id, initiating_user, involved_item, nature, timestamp, related_request, affected_user)
    Logs have implicit relationships with Users, Items, and Cart_Requests
User(id, username, email, password)

Development Environment: The technology versions used for development are specified below:

Django 1.10.5
Python 3.4.3
Postgresql 9.3
Ubuntu 14.04

A development environment can be configured on an Ubuntu machine by installing the above programs and cloning our project:

```
### Install required python, postgresql, apache packages

sudo apt-get update
sudo apt-get install python3-pip
sudo apt-get install libpq-dev
sudo apt-get install postgresql postgresql-contrib
sudo apt-get install git
sudo pip3 install virtualenv

### Now clone the project
git clone https://github.com/lad-47/inventory-project.git

cd inventory-project
cd mysite
virtualenv mysiteenv  ###virtualenv -p python3 mysiteenv if default is python 2 (check python
                      ###--version first)
source mysiteenv/bin/activate

pip install -r requirements.txt

### Set up database
sudo -i -u postgres
createdb inventory
### Switch back to main user (bitnami for Duke VMs)
su - bitnami

sudo service postgresql restart
```

```
cd inventory-project
cd mysite
source mysiteenv/bin/activate
python manage.py makemigrations
python manage.py migrate

python manage.py createsuperuser
python manage.py collectstatic
```

For full details on how to expand your development environment to deploy to a secure Apache server, please refer to our Deployment Guide.