INTELL: The Craft of Intelligence
Installation Guide

Dylan Wright
Christina Campbell
Braden Fullington
Casey O'Kane

April 28, 2016

1 Introduction

This document serves as the installation guide for the Spring 2016 CS 499 senior design project INTELL: The Craft of Intelligence. This document specifies requirements, and instructions for installation of the product.

2 Acquisition

This project can be acquired as a tarball [1]

3 Requirements

3.1 Environment

This project should be installed on a Linux machine. While Django can be installed on other operating systems, there is no guarantee that all features will work if not installed on a Linux system. This project was originally developed using Arch Linux on the development server but other distributions can be used if other software dependencies are met. This software's hardware requirements are as follows:

RAM 1 GB minimum. A swap partition/file is recommended but not required.

Storage Dependent on use. For development purposes, the database required about .5 MB maximum. If this software is deployed to many users this would grow.

CPU Dependent on use.

3.2 Dependencies

The following software packages are required by this software. Instructions for their acquisition can be found in the appendix.

Python 3.X

Django 1.9 MVC framework used as backend of package. The following additional Django applications are required and can be acquired from the locations specified in the appendix.

[1] `https://github.com/dylan-wright/cs499-intell/releases/tag/v1.0`

- `django-crontab`
- `django-jenkins`
- `django-casper`

SQLite3 Relational database. MySQL/MariaDB and PostgreSQL can be used with some modifications to the project settings.

Jenkins Continuous integrations server. Dependency can be removed by removing django-jenkins application and JENKINS_TASKS from settings file.

pep8/pyflakes optional Python linters which can be used in conjunction with Jenkins build server.

Bootstrap Already installed. HTML pages which require it use CDN's.

vis.js Visualization javascript library. Already installed.

js-cookie Javascript cookie library. Already installed.

4 Instructions

These instructions are for installation of the product in a development environment. To install the product in a production setting refer to the Django documentation on the subject [2].

4.1 Installation

These instructions assume a tarball with the source files *and all dependencies have been acquired*. These instructions assume the user is familiar with a command line environment and UNIX tools.

1. Extract the tarball
   `$ tar -xvf vx.y.tar.gz`
2. Enter the `cs499-intell-x.y/` directory
3. Prepare the database
   `$ python3 manage.py migrate`
4. Start the server
   `$ python3 manage.py runserver 0.0.0.0:PORT`

4.2 Configuration

At this point the Django server should be running on the specified port and accessible from any IP address. The next steps are optional but recommended.

- Run the tests (should execute in about 30 seconds)
  `./manage test`
- Create an admin
  `$ python3 manage.py createsuperuser`
- Start the crontab job
  `$ python3 manage.py crontab add`
- Set up a Jenkins CI server (useful but not required)

[2] `https://docs.djangoproject.com/en/1.9/howto/deployment/`

4.3 Usage

While running the unittests from the installation steps should reveal most issues with the installation. To see if you can connect, try to access the root URI of the machine/port your have installed on from a browser.

A Acquiring Dependencies

Several dependencies can by obtained using `pip`. `pip` is a package manager for installing python packages. Installation syntax is:
`# pip install packagename`
Full documentation for `pip` is available at `https://pip.pypa.io/en/stable/`

A.1 Python 3

On most Linux distributions can be obtained through package manager. Otherwise available at `https://www.python.org/downloads/`

A.2 Django 1.9+

Can be obtained with some Linux package managers (`pacman` yes, not sure about `apt`). Can also be obtained using `pip`. Can also be obtained from `https://www.djangoproject.com/download/`

A.3 `django-crontab`

Installation instructions at: `https://github.com/kraiz/django-crontab` (uses `pip`).

A.4 `django-jenkins`

Installation instructions at: `https://github.com/kmmbvnr/django-jenkins` (uses `pip`)

A.5 `django-casper`

Requires phantomjs and casperjs. Casper can be obtained with package manager on Linux or directly at `http://casperjs.org/`. Installation instructions at: `https://github.com/dobarkod/django-casper` (uses `pip`). Must modify `/usr/lib/python3.5/site-packages/casper/tests.py` to have CasperTestCase subclass StaticLiveServerTestCase instead of LiveServerTestCase (see below):

```
from django.contrib.staticfiles.testing import StaticLiveServerTestCase
#from django.test import LiveServerTestCase
from subprocess import Popen, PIPE
import os
import sys

from django.contrib.staticfiles.handlers import StaticFilesHandler
from django.contrib.staticfiles.views import serve
from django.utils.http import http_date
from django.conf import settings

__all__ = ['CasperTestCase']


def staticfiles_handler_serve(self, request):
    import time
    resp = serve(request, self.file_path(request.path), insecure=True)
    if resp.status_code == 200:
        resp["Expires"] = http_date(time.time() + 24 * 3600)
    return resp
```

```
#class CasperTestCase(LiveServerTestCase):
class CasperTestCase(StaticLiveServerTestCase):
...
```

A.6 SQLite3

Can be obtained with package manager or at `https://www.sqlite.org/`

A.7 Jenkins (optional)

Can be obtained with package manager or at `https://jenkins.io/`. See online documentation for CI server configuration.