This is the Title of my Thesis

Your Name

December 2012

PROJECT THESIS
Department of Production and Quality Engineering
Norwegian University of Science and Technology

Supervisor 1: Professor Ask Burlefot
Supervisor 2: Professor Fingal Olsson

Foreword

Originally inspired by the Nordic Collegiate Programming Contest (NCPC), it has been held at NTNU every spring since 2007. The format is a five-hour contest with competing teams consisting of one, two or three contestants. A team of volunteer judges write the problems and answer clarification requests during the contest, while another team hands out balloons for each solved problem. Usually a rather hectic affair, it is extremely important that everything is well prepared. The number of teams is often more than 100, with the record being 162 teams in 2011

The contest system that verifies solutions is at the heart of the contest when it is in progress, and needs to be working perfectly at all times. The system must handle several submissions per second, while verifying that each one is correct and runs within the set resource limits. Submissions must show up on the high score list, and when problems are solved the team handing out balloons must be notified. In addition to this there were a lot of other functional requirements having to do with the bureaucracy of organizing the contest

A requirement was that new features could be easily added in the future, and the code was written with this in mind. The project will now become open source, and all programming contest enthusiasts will soon be able to request and implement their desired features

All aspects of this project have been pleasing and delightful for us. The team has exceeded all our expectations and their system will be used for years to come.

Preface

Before there were computers, there were algorithms. But now that there are computers, there are even more algorithms, and algorithms lie at the heart of computing. Designing a system for eager students to hone their skill in the heart of computing has been a true joy

Our group never wanted to settle for adequacy and mere requisiteness. For the past few months, weve taught ourselves a new programming language and framework and used advanced development frameworks - while tackling many social and technical conflicts.

We have ve proven how Ambition is a dream with a V8 engine, as Elvis Presley once said.

The group would like to thank our eager customers, Finn Inderhaug Holme, Christian Chavez and Christian Neverdal Jonassen for their time to meet us and provide constructive feedback. We also owe a big thanks to our supervisor, Hong Guo, for constructive criticism and reflections; without which, we would not ascertain the peak of our own potential

Contents

Chapter 1

Installation guide

This is the complete install guide for GentleIDI. The guide will assume that the reader is has got some basic linux skills. You should be capable of installing packages by means of a package-manager like apt, yum etc.

Though GentleIDI is not tightly linked with any specific linux distro, this guide assumes that you're using Ubuntu Server 14.04. This is the only distro on which the system has been tested thoroughly at the time of writing.

GentleIDI is in many ways a straightforward django-based website, and hence there are a lot of possible setups to choose from. This guide is inspired by a guide written by Michal Karzynski, and will guide you through the steps of setting up the system using a combination of gunicorn and nginx.

1.1 Creating your users

Running a website as a user with root privileges or anything of the sort is far from recommended. Therefore you are recommended to create a new user and a new usergroup. The names of both the group and the user can be chosen as you please, but the rest of the guide will stick to using a user called gentleidi and a group named webapps.

```
sudo mkdir -p /webapps/gentleidi
sudo groupadd -{}-system webapps
sudo useradd -{}-system -{}-gid webapps -{}-home /webapps/gentleidi gentleidi
sudo chown gentleidi:webapps /webapps/gentleidi/
```

Now you have a user named gentleidi which is a member of the usergroup webapps, and whose home directory is /webapps/gentleidi.

In addition to the user we just created, we need another user, specifically used to run the untrusted software submitted by the contestants. GentleIDI assumes that this user is named gentlemember. However, changing this value in the source is no complicated matter.

```
sudo useradd -{}-system gentlemember
```

The system needs to be able to execute commands both as gentleidi and gentlemember. As the web server runs as gentleidi we need to make sure that gentleidi can execute commands as gentlemember. Add the following line to your sudoers file.

gentleidi ALL=(gentlemember) NOPASSWD:ALL

If you don't know how to edit your sudoers, to open the sudoers file in a text editor simply type the following command:

```
Now we've got two users, one capable of executing
commands as the other. What we want to do now is to ensure that
gentlemember is unable to communicate via network. This is done by
applying two rather straightforward iptable rules.

\begin{lstlisting}sudo iptables -A OUTPUT -m owner -{}-uid-owner gentlemember -j LOG
sudo iptables -A OUTPUT -m owner -{}-uid-owner gentlemember -j REJECT
```

Though this will restrict the user's network access, be aware of software installed on your system which is capable of switching to another user.

1.2 Setting up the environment

Due to a lot of strict changes made in python versions, a lot of libraries do not work across different versions of python. This leaves python in a situation where program A might need python to be version X and program B might need python to be version Y. So, what do you do? You setup a virtual environment.

Virtual environments is a way of setting up separate python setups for different sets of programs.

What we want to do is to turn the home directory of the gentleidi user into a virtual environment.

```
sudo su gentleidi
virtualenv ~{}/env
```

Now that you've got a virtual environment you can start filling it with something useful, like the content of the project's git repo.

```
cp -r /path/to/repo/IDIOpen/ /webapps/gentleidi/
```

Please note that you only need the wsgi folder from the repo, however, updating is a lot easier when all you've got to do is pull the latest version directly using git. The downside is that you could possibly end up committing your production system configuration files etc to the repo. However, we're going to assume that you will not be developing directly in your production system, and thereby avoid the hazard.

Before leaving this step, ensure that the files in /webapps/gentleidi has got the correct file permissions.

```
\section{Installing required packages}
Now it's time to start making sure that
you've got the packages you need to run GentleIDI.

\begin{lstlisting}sudo apt-get install git nginx libmysqlclient-dev python-dev
```

You might already have most of these packages, however, better safe than sorry.

The next thing you need to do before continuing is to log in as gentleidi and activate your newly created virtual environment.

```
sudo su gentleidi
source \~{}/env/bin/activate
```

Installing the required python packages via pip is easily done. In the project root directory there's a file named requirements.txt. This file is simply a list of required packages, to install them simply execute the following:

1.3 Database

GentleIDI needs a database to store its data. This guide will show you how to setup GentleIDI with a mysql database server, however, if you feel like using mariaDB, postgresql, or even SQLite, then please do. Any database server supported by Django is supported by GentleIDI.

Naturally you don't need to install the database server on the same host as the web server, that's what we'll do for now.

Now what we need to do is to create a database and a mysql user that GentleIDI can use. During the install process you were required to set a root password for the mysql-server. Login as root and perform the following commands:

```
CREATE USER
gentledb'@'localhost'
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON * \@. * TO
'newuser'@'localhost';

FLUSH PRIVILEGES;

CREATE DATABASE gentleidi CHARACTER SET uft8 COLLATEutf8\_general\_ci;
```

Remember to replace "gentledb" and "password" with a suitable username and password. Now you need to ensure that GentleIDI uses your newly created database. Edit the DATABASES entry in IDIOpen/wsgi/openshift/settings.py

```
if MYSQL:
    DATABASES = {
        'default': {
            'ENGINE'   : 'django.db.backends.mysql',
            'NAME'     : 'gentleidi',
            'USER'     : 'gentledb',
            'PASSWORD' : 'password',
            'HOST'     : 'localhost',
            'PORT'     : '3306',
```

In order to make sure that the database is working properly, login as gentleidi, activate your environment and synchronize GentleIDI's database.

```
source \~{}/env/bin/activate
python \~{}/IDIOpen/wsgi/manage.py syncdb
python \~{}/IDIOpen/wsgi/manage.py migrate
```

If this command terminates properly, then your database should be good to go. In fact you should be able to run GentleIDI on a development server at this point. But first, you need to create a admin account.To do so, simply execute the following:

```
python \~{}/IDIOpen/wsgi/openshift/manage.py createsuperuser
```

To start the development server run: