

# thepoet.me

Submitted for partial fulfilment of the Degree  
of  
Bachelor of Technology  
(Information Technology)



**Submitted by:**  
Jagjeet Singh  
1411266  
146034

**Submitted to:**  
Sidharath Jain  
Training Coordinator  
IT Department

---

Information Technology  
**Guru Nanak Dev Engineering College**  
Ludhiana 141006

## Abstract

Thepoet.me (Usually styled as thepoet.me) is a Django based site that I worked upon for my 6-month training. The site offers registered user a simple platform from which to link his/her published book and popular social networking websites such as Facebook, Twitter and Instagram. It is characterized by its one-page user profiles, each with a large, often-artistic background and abbreviated biography. It is a free and Open-source application. The site is fully responsive with all type of devices.

Your thepoet.me acts as a virtual online business card. Put the URL to your site in your e-mail as digital signature, Twitter profile, share it on Facebook, add it to your Instagram as a website.

If you are a poet or writer of some sort who doesn't have a website, you can point colleagues, clients, and prospects to your thepoet.me page so they can find out more about you and connect with you in all the right places.

There are countless platforms out there that you can use to build your own free personal website, but not all of them will deliver the same sense of quality and professionalism. If you're a poet or writer and looking for something fast and simple that you just need to represent a landing page for yourself, thepoet.me could be one of your best alternatives to choose from.

Other website and blog building tools like Blogger and WordPress.com offer a complete platform to build upon, including the ability to host several web pages, write blog posts and feature widgets. thepoet.me gives you just one, single page to display all your links and a summary of yourself, making it an ideal tool to get straight to the point about you and your published books.

## Acknowledgement

I, student of Guru Nanak Dev Engineering College, Ludhiana, have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

The author is highly grateful to Dr. Sehijpal Singh, Guru Nanak Dev Engineering College, Ludhiana for providing him with the opportunity to carry out his Six Weeks Training at Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana.

The author would like to wholeheartedly thank Dr. H.S. Rai Dean, Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana who is a vast sea of knowledge and without whose constant and never ending support and motivation, it would never have been possible to complete the project and other assignments so efficiently and effectively.

Jagjeet Singh

<b>1</b>	<b>INTRODUCTION OF ORGANIZATION</b>	<b>1</b>
1.1	Testing and Consutancy Cell . . . . .	1
<b>2</b>	<b>Introduction To Project</b>	<b>3</b>
2.1	Overview . . . . .	3
2.2	The Existing System . . . . .	3
2.3	User Requirement Analysis . . . . .	4
2.3.1	Functional Requirements . . . . .	4
2.3.2	Non-functional requirements . . . . .	4
2.4	Feasibility Study . . . . .	5
2.5	Objective of Project . . . . .	5
<b>3</b>	<b>PROJECT DESIGN</b>	<b>6</b>
3.1	Product Perspective . . . . .	6
3.2	Product Functions . . . . .	6
3.3	User Characteristics . . . . .	7
3.3.1	The General User . . . . .	7
3.3.2	Developers . . . . .	7
3.4	Flowchart . . . . .	7
3.4.1	Detailed Description . . . . .	8
3.5	Class Diagrams . . . . .	8
3.6	Dependencies . . . . .	8
<b>4</b>	<b>DEVELOPMENT AND IMPLEMENTATION</b>	<b>11</b>
4.1	Python . . . . .	11
4.1.1	Features of Python . . . . .	11
4.1.2	Installation of Python . . . . .	11
4.2	Django . . . . .	12
4.2.1	Features of Django . . . . .	12
4.2.2	Installation of Django . . . . .	13
4.2.3	MTV . . . . .	13
4.2.4	Creating Prject in Django . . . . .	13
4.2.5	Development Server in Django . . . . .	14
4.2.6	Database setup . . . . .	14
4.3	Introduction to $\text{\LaTeX}$ . . . . .	15
4.3.1	Typesetting . . . . .	16
4.3.2	Installing $\text{\LaTeX}$ on System . . . . .	17
4.3.3	Graphical Editors for $\text{\LaTeX}$ . . . . .	18
4.3.4	Pdftscreen $\text{\LaTeX}$ . . . . .	19
4.3.5	Web based graphic generation using $\text{\LaTeX}$ . . . . .	19
4.4	Introduction to Doxygen . . . . .	20
4.4.1	Features of Doxygen . . . . .	20
4.5	Introduction to Github . . . . .	22
4.5.1	What is Git? . . . . .	23
4.5.2	Installation of Git . . . . .	23

4.5.3	Various Git Commands . . . . .	24
4.5.3.1	Create Repositories . . . . .	24
4.5.3.2	Make Changes . . . . .	24
4.5.3.3	Group Changes . . . . .	24
4.5.3.4	Save Fragments . . . . .	25
4.5.3.5	Synchronize Changes . . . . .	25
4.6	Implementation . . . . .	26
4.7	Web pages in thepoet.me . . . . .	26
4.7.1	Website . . . . .	26
4.7.2	User sign up page . . . . .	26
4.7.3	User login page . . . . .	26
4.7.4	User profile page . . . . .	29
4.7.5	Add book page . . . . .	29
4.7.6	User edit profile page . . . . .	29
4.7.7	Forget password page . . . . .	29
4.8	Testing . . . . .	29
<b>5</b>	<b>CONCLUSION AND FUTURE SCOPE</b>	<b>33</b>
5.1	Conclusion . . . . .	33
5.2	Future Scope . . . . .	34
	<b>BIBLIOGRAPHY</b>	<b>34</b>

1.1	Guru Nanak Dev Engineering College . . . . .	1
3.1	Class Diagram for accounts.models.Profile . . . . .	9
3.2	Class Diagram for accounts.forms.FirstBookForm . . . . .	9
3.3	Class Diagram for accounts.forms.ProfileForm . . . . .	10
4.1	Python logo . . . . .	11
4.2	Django logo . . . . .	12
4.3	Output of runserver . . . . .	14
4.4	Donald Knuth, Inventor Of T <sub>E</sub> X typesetting system . . . . .	16
4.5	L <sup>A</sup> T <sub>E</sub> X output of above program. . . . .	17
4.6	Texmaker, A Graphical L <sup>A</sup> T <sub>E</sub> X Editor . . . . .	18
4.7	LEd, A Graphical L <sup>A</sup> T <sub>E</sub> X Editor . . . . .	18
4.8	Web based graphic generation using L <sup>A</sup> T <sub>E</sub> X(input page) . . . . .	20
4.9	Doxygen Logo . . . . .	20
4.10	Documentation using Doxygen (main page) . . . . .	21
4.11	Doxygen documentation of a class hierarchy . . . . .	21
4.12	Github Logo . . . . .	22
4.13	Git Logo . . . . .	23
4.14	Website of thepoet.me . . . . .	27
4.15	Sign up page . . . . .	28
4.16	Login page . . . . .	28
4.17	User's profile page . . . . .	29
4.18	Add book page . . . . .	30
4.19	User's edit profile page . . . . .	30
4.20	Forget password page . . . . .	31
4.21	Performance report by dareboost . . . . .	32
4.22	SEO report . . . . .	32



Figure 1.1: Guru Nanak Dev Engineering College

I had my Six Month Industrial Training at TCC-Testing And Consultancy Cell, GNDEC Ludhiana. Guru Nanak Dev Engineering College was established by the Nankana Sahib Education Trust Ludhiana. The Nankana Sahib Education Trust i.e NSET was founded in memory of the most sacred temple of Sri Nankana Sahib, birth place of Sri Guru Nanak Dev Ji. With the mission of Removal of Economic Backwardness through Technology Shiromani Gurudwara Parbandhak Committee i.e SGPC started a Poly technical was started in 1953 and Guru Nanak Dev Engineering College was established in 1956.

The main goal of this institute is:

- To build and promote teams of experts in the upcoming specialisations.
- To promote quality research and undertake research projects keeping in view their relevance to needs and requirements of technology in local industry.
- To achieve total financial independence.
- To start online transfer of knowledge in appropriate technology by means of establishing multipurpose resource centres.

## 1.1 Testing and Consutancy Cell

I had my Six Month Institutional Training at TCC i.e Testing And Consultancy Cell, GNDEC Ludhiana under the guidance of Dr. H.S.Rai Dean Testing and Consultancy Cell. Testing and Consultancy Cell was established in the year 1979 with a basic aim to produce quality service

for technical problems at reasonable and affordable rates as a service to society in general and Engineering fraternity in particular.

Consultancy Services are being rendered by various Departments of the College to the industry, State Government Departments and Entrepreneurs and are extended in the form of expert advice in design, testing of materials & equipment, technical surveys, technical audit, calibration of instruments, preparation of technical feasibility reports etc. This consultancy cell of the college has given a new dimension to the development programmes of the College. Consultancy projects of over Rs. one crore are completed by the Consultancy cell during financial year 2009-10.

Ours is a pioneer institute providing Consultancy Services in the States of Punjab, Haryana, Himachal, J&K and Rajasthan. Various Major Clients of the Consultancy Cell are as under:

- Northern Railway, Govt. of India
- Indian Oil Corporation Ltd.
- Larson & Turbo.
- Multi National Companies like AFCON & PAULINGS.
- Punjab Water Supply & Sewage Board



## 2.1 Overview

Thepoet.me(Usually styled as thepoet.me) is a Django based site that I worked upon for my 6-month training. The site offers registered user a simple platform from which to link his/her published book and popular social networking websites such as Facebook, Twitter and Instagram. It is characterized by its one-page user profiles, each with a large, often-artistic background and abbreviated biography. It is a free and Open-source application. The site is fully responsive with all type of devices.

Your thepoet.me acts as a virtual online business card. Put the URL to your site in your e-mail as digital signature, Twitter profile, share it on Facebook, add it to your Instagram as a website.

If you are a poet or writer of some sort who doesn't have a website, you can point colleagues, clients, and prospects to your thepoet.me page so they can find out more about you and connect with you in all the right places.

There are countless platforms out there that you can use to build your own free personal website, but not all of them will deliver the same sense of quality and professionalism. If you're a poet or writer and looking for something fast and simple that you just need to represent a landing page for yourself, thepoet.me could be one of your best alternatives to choose from.

Other website and blog building tools like Blogger and WordPress.com offer a complete platform to build upon, including the ability to host several web pages, write blog posts and feature widgets. thepoet.me gives you just one, single page to display all your links and a summary of yourself, making it an ideal tool to get straight to the point about you and your published books.

The main idea of this project is to provide a personal identity page to poets or writers with features to link their published book and social networking sites. User Interface is designed in a way such that layman can easily understand it. The core part of this project is implemented using Django and for GUI part Bootstrap is used. My training being not based on particular language or technology, different types of open-source softwares and technologies are used in this project and many during my training which are not used in this project like Android for Nitnem, Jekyll for blog and shell scripting for plzalert.me

My training being not based on particular language or technology, different type of open-source software's and technologies are used in this project and many during my training which are not used in this project like Django, Facebook's Graph API for *plzalert.me* WebApp.

## 2.2 The Existing System

There are countless platforms out there that you can use to build your own free personal homepage, but not all of them will deliver the same sense of quality and professionalism.

### Limitations of previous system

- Domains are not representing poets
- Cant add published book cover
- No online business card for poets

## 2.3 User Requirement Analysis

User Requirements Analysis for a software system is a complete description of the requirements of the User. It includes functional Requirements and Non-functional Requirements. Non-functional requirements are requirements which impose constraints on the design or implementation.

### Users of the System:

1. **Poet or writer:** A poet is a person who writes poetry. A poet may simply be a writer of poetry.
2. **Author:** A writer of a book, article, or document.

### 2.3.1 Functional Requirements

- **Specific Requirements:** This phase covers the whole requirements for the system. After understanding the system we need the input data to the system then we watch the output and determine whether the output from the system is according to our requirements or not. So what we have to input and then what we'll get as output is given in this phase. This phase also describes the software and non-function requirements of the system.
- **Input Requirements of the System:**
  - **Social Login/Signup:** This means there should be an option so that a user can Login or Register as new user using his/her social accounts. (Facebook, Gmail)
  - **Signup:** This means there should be an option so that a user can register.
  - **Login:** This means there should be an option so that a user can login after successful registration.
  - **Password Reset:** This means there should be an option so that a user can change his/her password.
  - **Lost Password:** This means there should be an option so that a user can set a new password in case he/she forgot the old one.
  - **Edit Profile:** This means there should be an option so that a user can edit his/her profile anytime after successful login.
  - **User profile Link:** This means there should be an option so that a user can copy his profile link.
  - **Social networking links:** This means there should be an option so that a user can add his/her Facebook, Twitter and Instagram links.
  - **Book Cover:** This means there should be an option so that a user can add his/her books cover and details.

### 2.3.2 Non-functional requirements

1. E-mail should be sent using time of registration or password reset.
2. User web page should be shown after clicking on his/her profile link.
3. Site should be fully responsive.

## 2.4 Feasibility Study

Feasibility study aims to uncover the strengths and weaknesses of a project. In its simplest term, the two criteria to judge feasibility are cost required and value to be attained. As such, a well-designed feasibility analysis should provide a historical background of the project, description of the project or service, details of the operations and management and legal requirements. Generally, feasibility analysis precedes technical development and project implementation. These are some feasibility factors by which we can determine that the project is feasible or not:

- **Technical feasibility:** Technological feasibility is carried out to determine whether the project has the capability, in terms of software, hardware, personnel to handle and fulfill the user requirements. This whole project is based on Open Source Environment and is part of an open source software which would be deployed on any OS.
- **Economic feasibility:** In Economic feasibility, we determine whether the benefit is gain according to the cost invested to develop the project or not. If benefits outweigh costs, only then the decision is made to design and implement the system. It is important to identify cost and benefit factors, which can be categorized as follows:
  1. Development costs.
  2. Operating costs.

Thepoet.me is also Economically feasible for a year as It could be developed and maintain with zero cost as It is supported by Open source community and GitHub Students Pack.

## 2.5 Objective of Project

One of the primary benefit of thepoet.me is to create a personal identity page for poets or writers. It's a great place to pull together your online profiles into one place so people know where to find you across different social networks.

1. The goals of this project.
2. Developing a best social platform for poets or writers.
3. Providing a link to poets that present them.
4. Offer them a one page website where they can add their book.
5. Linking their social accounts in one place.

### 3.1 Product Perspective

This site is supposed to be part of an open source, under the GNU general Public License. It offers registered user a simple platform from which to link his/her published book and popular social networking websites such as Facebook, Twitter and Instagram. It is characterized by its one-page user profiles, each with a large, often-artistic background and abbreviated biography.

The following are the main features that are included in thepoet.me

1. **Web Compatibility:** The site easily render on various resolutions, screen sizes, and browsers; and with the increasing popularity of mobile devices, it function properly on the plethora of these types of devices.
2. **Search Engine Optimisation:** The site generally receive many visitors, and one method to attract visitors is search engine optimisation. The site is optimised with Google search Engine.
3. **Simple and Professional Web Design:** There are countless platforms out there that you can use to build your own free personal website, but not all of them will deliver the same sense of quality and professionalism. If youre a poet or writer and looking for something fast and simple that you just need to represent a landing page for yourself, thepoet.me could be one of your best alternatives to choose from.
4. **Clear, User-friendly Navigation:** The site contain a user-friendly navigation scheme that allows visitors to quickly find the information needed. Important links must be easy to find and given logical, simple, and include easy-to-understand labels. Calls to action are placed in conspicuous spots within the navigations scheme.

### 3.2 Product Functions

Functions performed by Customizer are:

- **Social Login/Signup:** This means there should be an option so that a user can Login or Register as new user using his/her social accounts. (Facebook, Gmail)
- **Signup:** This means there should be an option so that a user can register.
- **Login:** This means there should be an option so that a user can login after successful registration.
- **Password Reset:** This means there should be an option so that a user can change his/her password.
- **Lost Password:** This means there should be an option so that a user can set a new password in case he/she forgot the old one.
- **Edit Profile:** This means there should be an option so that a user can edit his/her profile anytime after successful login.

- **User profile Link:** This means there should be an option so that a user can copy his profile link.
- **Social networking links:** This means there should be an option so that a user can add his/her Facebook, Twitter and Instagram links.
- **Book Cover:** This means there should be an option so that a user can add his/her books cover and details.

### 3.3 User Characteristics

We have identified three potential classifications of users of our system:

1. Poet or writer: The people who write poetry. A poet may simply be a writer of poetry
2. Author: The people who are writer of a book, article, or document.
3. Developers: These are people who might want to integrate new features to thepoet.me

#### 3.3.1 The General User

All users can be assumed to have the following characteristics:

1. Ability to read and understand English.
2. Have at least an e-mail or any social account for registration (Facebook, Google).
3. Beyond the above, no further facility with computer technology can be assumed.

#### 3.3.2 Developers

The Developers can be assumed to have the following characteristics:

1. Basic Knowledge of programming.
2. Basic Knowledge of python and Django.
3. Knowledge of web designing (HTML, CSS, javascript and Bootstrap).
4. Knowledge of git.

### 3.4 Flowchart

A flowchart is a type of diagram that represents an algorithm, work flow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows and the flowchart of customizer showing the flow of control and Data in the software.

### 3.4.1 Detailed Description

The basic implementation of this project is almost done in form of prototype. There is need to modify the structure of the project. We have to divide the task into there parts:

1. **Front end** It will deal with how thepoet.me will look to the user.
2. **Back End** On backend, a top layer is HTML pages render on the user screen but in bottom layer Django (Python Web framework) is running. Django attempts to support as many features as possible on all database backends. However, not all database backends are alike, and weve had to make design decisions on which features to support and which assumptions we can make safely.

## 3.5 Class Diagrams

Class Diagrams describe the static structure of the system. Following classes diagram represent the relationship in thepoet.me:

1. Figure 3.1 shows the class diagram of the Django ModelForm class which is the base class of accounts.models.Profile.
2. Figure 3.2 shows the class diagram of the Django ModelForm class which is the base class of accounts.forms.FirstBookForm.
3. Figure 3.3 shows the class diagram of the accounts.forms.ProfileForm.

## 3.6 Dependencies

Dependencies include softwares or framework that need to be installed for proper working of this software.

This webapp could be installed on any given list of operating system.

1. Mac OS X
2. Windows
  - (a) XP or newer on x86 32/64 bit
3. Linux
  - (a) Debian
  - (b) Ubuntu
  - (c) Kubuntu
  - (d) Arch Linux
  - (e) openSUSE
  - (f) Fedora
4. BSD

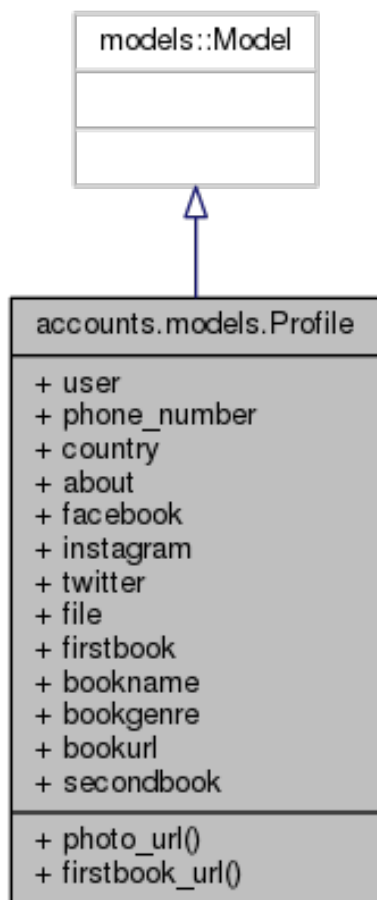


Figure 3.1: Class Diagram for `accounts.models.Profile`

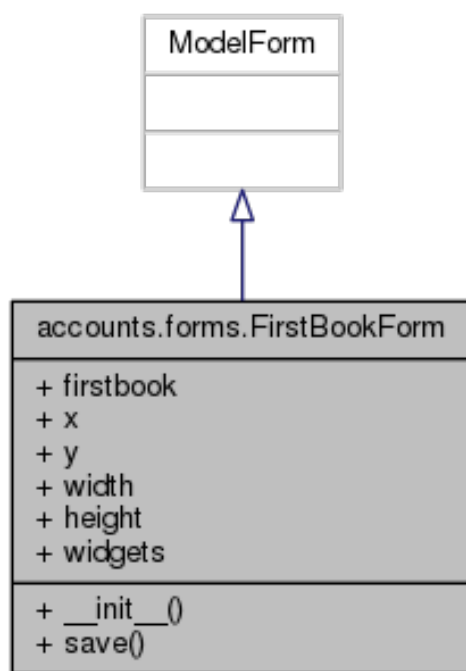


Figure 3.2: Class Diagram for `accounts.forms.FirstBookForm`

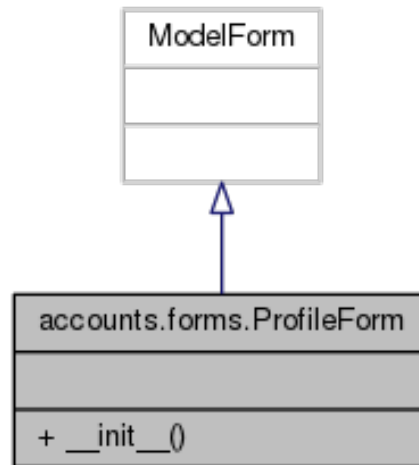


Figure 3.3: Class Diagram for `accounts.forms.ProfileForm`

- (a) NetBSD  $\geq 6.1$
- (b) FreeBSD  $\geq 10$
- (c) OpenBSD

But If you want to run thepoet.me on your system from soucre code on *any OS*, you need some libraries and tools. The version numbers in brackets specify the versions which have been used for development. Other versions may or may not work as well.

If you're using a newer version of Ubuntu, you can install these libraries from aptitude. If you're using Mac, or an older Linux/BSD, there are build scripts that download and compile the libraries from source. Follow the instructions for the platform you're compiling on below. Following are dependencies of thepoet.me.

1. python-decouple
2. django
3. django\_countries
4. django-avatar
5. easy\_thumbnails
6. sorl-thumbnail
7. social-auth-app-django
8. mysqlclient
9. django-image-cropping
10. python-social-auth
11. pyjwtkest
12. libmysqlclient-dev
13. libapache2-mod-wsgi-py3



## 4.1 Python



Figure 4.1: Python logo

Python is a dynamic language, as in python coding is very easy and also it require less coding and about its interpreted nature it is just excellent. Python is a high level programming language and Django which is a web development framework is written in python language.

Python is an easy to learn, powerful programming language. Python runs on Windows, Linux/Unix, Mac OS X. Python is free to use, even for commercial products. Python can also be used as an extension language for existing modules and applications that need a programmable interface. Python is free to use, even for commercial products, because of its OSI-approved open source license.

### 4.1.1 Features of Python

- Very clear, readable syntax.
- Strong introspection capabilities.
- Intuitive object orientation.
- Natural expression of procedural code.
- Full modularity, supporting hierarchical packages.
- Exception-based error handling.
- Very high level dynamic data types.
- Extensive standard libraries and third party modules for virtually every task.
- Extensions and modules easily written in C, C++ (or Java for Jython, or .NET languages for IronPython).
- Embeddable within applications as a scripting interface.

### 4.1.2 Installation of Python

Installation of python is a very easy process. The current python versions are: Python 2.7.1 and Python 3.2. Type the commands in the terminal:

```
$ wget http://www.python.org/ftp/python/2.7/Python-2.7.tgz
```

```
$ tar xzf Python-2.7.tgz
```

This will install the python on your pc/laptop.

## 4.2 Django



Figure 4.2: Django logo

Django is an open source web application framework written in python. It lets you build high-performing, elegant Web applications quickly. Django focuses on automating as much as possible. Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "pluggability" of components, rapid development, and the DRY principal. Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Django takes it's name from the early jazz guitarist Django Reinhardt, a gypsy savant who managed to play dazzling and electrifying runs on his instrument even though two of the fingers on his left hand were paralyzed in an accident when he was young.

Thus its a fitting name for the framework. Django can do some very complex things with less code and a simpler execution than youd expect. It doesn't take a heavy hand to build with Django. The framework does the repetitive work for you, allowing you to get a working website up quickly and easily.

### 4.2.1 Features of Django

- Clean URLs
- Object- Relational Mapping
- Loosely coupled components
- Designer-friendly templates
- Cache framework
- MVC architecture
- Jython support
- DRY ( Don't Repeat Yourself)

## 4.2.2 Installation of Django

Installation of Django is very easy. To install Django version 1.4, type the following commands:

```
$ wget http://www.djangoproject.com/download/1.4/tarball
```

```
$ tar xzvf Django-1.4.tar.gz
```

```
$ cd Django-1.4
```

```
$ sudo python setup.py install
```

This will install the django on your system.

## 4.2.3 MTV

Django adopts the standard MVC (Model-View-Controller) design pattern. But instead, their naming convention is the MTV (Model-Template-View).

- **Model** is an object relational mapping to your database schema. So each model is a class which represents a table in your database. Django models provide easy access to an underlying data storage mechanism, and can also encapsulate any core business logic, which must always remain in effect, regardless of which application is using it. Models exist independent of the rest of the system, and are designed to be used by any application that has access to them. In fact, the database manipulation methods that are available on model instances can be utilized even from the interactive interpreter, without loading a Web server or any application-specific logic.
- **Template** is simply HTML for your views. It also allows you to display different messages depending on whether or not a user is logged in. Templates are Django's provided way of generating text-based output, such as HTML or emails, where the people editing those documents may not have any experience with Python. Therefore, templates are designed to avoid using Python directly, instead favoring an extensible, easy-to-use custom language built just for Django.
- **View** could be a homepage or a page to display a user's information, for instance. A view accepts user input, including simple requests for information; behaves according to the application's interaction logic; and returns a display that is suitable for user's to access the data represented by models.

## 4.2.4 Creating Project in Django

If this is your first time using Django, you'll have to take care of some initial setup. Namely, you'll need to auto-generate some code that establishes a Django project- a collection of settings for an instance of Django, including database configuration, Django-specific options and application-specific settings. From the command line, cd into a directory where you'd like to store your code, then run the command

```
$ django-admin.py startproject mysite
```

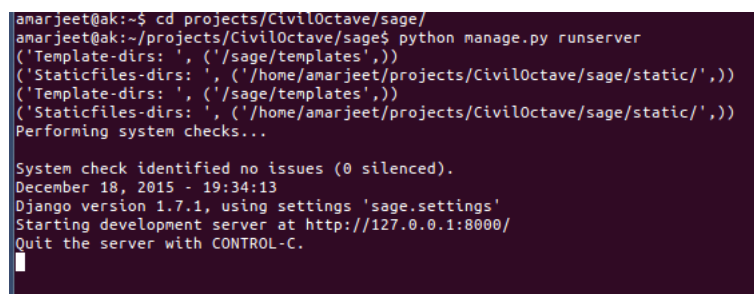
This will create a mysite directory in your current directory.

### 4.2.5 Development Server in Django

Change into the outer mysite directory, if you haven't already, and run the command

```
$ python manage.py runserver
```

You'll see the following output on the command line:



```
amarjeet@ak:~$ cd projects/CivilOctave/sage/
amarjeet@ak:~/projects/CivilOctave/sage$ python manage.py runserver
('Template-dirs: ', ('/sage/templates',))
('Staticfiles-dirs: ', ('/home/amarjeet/projects/CivilOctave/sage/static/',))
('Template-dirs: ', ('/sage/templates',))
('Staticfiles-dirs: ', ('/home/amarjeet/projects/CivilOctave/sage/static/',))
Performing system checks...

System check identified no issues (0 silenced).
December 18, 2015 - 19:34:13
Django version 1.7.1, using settings 'sage.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Figure 4.3: Output of runserver

### 4.2.6 Database setup

In this, we need to edit the settings.py file of the Project, that is the configuration file. It's a normal Python module with module-level variables representing Django settings. Change the following keys in the DATABASES 'default' item to match your database connection settings.

- **ENGINE** – Either 'django.db.backends.postgresql\_psycopg2', 'django.db.backends.mysql', 'django.db.backends.sqlite3' or 'django.db.backends.oracle'. Other backends are also available.
- **NAME** – The name of your database. If you're using SQLite, the database will be a file on your computer; in that case, NAME should be the full absolute path, including filename, of that file. If the file doesn't exist, it will automatically be created when you synchronize the database for the first time (see below). When specifying the path, always use forward slashes, even on Windows (e.g. C:/homes/user/mysite/sqlite3.db).
- **USER** – Your database username (not used for SQLite).
- **PASSWORD** – Your database password (not used for SQLite).
- **HOST** – The host your database is on. Leave this as an empty string if your database server is on the same physical machine (not used for SQLite).

If you're new to databases, we recommend simply using SQLite by setting ENGINE to 'django.db.backends.sqlite3' and NAME to the place where you'd like to store the database.

SQLite is included as part of Python 2.5 and later, so you won't need to install anything else to support your database.

While you're editing `settings.py`, set `TIME_ZONE` to your time zone. The default value is the Central time zone in the U.S. (Chicago).

Also, note the `INSTALLED_APPS` setting toward the bottom of the file. That holds the names of all Django applications that are activated in this Django instance. Apps can be used in multiple projects, and you can package and distribute them for use by others in their projects.

By default, `INSTALLED_APPS` contain the following apps, all of which come with Django:

- `django.contrib.auth` – An authentication system.
- `django.contrib.contenttypes` – A framework for content types.
- `django.contrib.sessions` – A session framework.
- `django.contrib.sites` – A framework for managing multiple sites with one Django installation.
- `django.contrib.messages` – A messaging framework.
- `django.contrib.staticfiles` – A framework for managing static files.

These applications are included by default as a convenience for the common case.

Each of these applications makes use of at least one database table, though, so we need to create the tables in the database before we can use them. To do that, run the following command:

```
$ python manage.py syncdb
```

The `syncdb` command looks at the `INSTALLED_APPS` setting and creates any necessary database tables according to the database settings in your `settings.py` file. You'll see a message for each database table it creates, and you'll get a prompt asking you if you'd like to create a superuser account for the authentication system. Go ahead and do that.

## 4.3 Introduction to L<sup>A</sup>T<sub>E</sub>X

L<sup>A</sup>T<sub>E</sub>X, I had never heard about this term before doing this project, but when I came to know about its features, found it excellent. L<sup>A</sup>T<sub>E</sub>X (pronounced /letk/, /letx/, /ltx/, or /ltk/) is a document markup language and document preparation system for the T<sub>E</sub>X typesetting program. Within the typesetting system, its name is styled as L<sup>A</sup>T<sub>E</sub>X.

Within the typesetting system, its name is styled as L<sup>A</sup>T<sub>E</sub>X. The term L<sup>A</sup>T<sub>E</sub>X refers only to the language in which documents are written, not to the editor used to write those documents. In order to create a document in L<sup>A</sup>T<sub>E</sub>X, a `.tex` file must be created using some form of text editor. While most text editors can be used to create a L<sup>A</sup>T<sub>E</sub>X document, a number of editors have been created specifically for working with L<sup>A</sup>T<sub>E</sub>X.

L<sup>A</sup>T<sub>E</sub>X is most widely used by mathematicians, scientists, engineers, philosophers, linguists, economists and other scholars in academia. As a primary or intermediate format, e.g., translating DocBook and other XML-based formats to PDF, L<sup>A</sup>T<sub>E</sub>X is used because of the high quality of typesetting achievable by T<sub>E</sub>X. The typesetting system offers programmable desktop publishing features and extensive facilities for automating most aspects of typesetting and



Figure 4.4: Donald Knuth, Inventor Of TeX typesetting system

desktop publishing, including numbering and cross-referencing, tables and figures, page layout and bibliographies.

L<sup>A</sup>T<sub>E</sub>X is intended to provide a high-level language that accesses the power of TeX. L<sup>A</sup>T<sub>E</sub>X essentially comprises a collection of TeX macros and a program to process L<sup>A</sup>T<sub>E</sub>X documents. Because the TeX formatting commands are very low-level, it is usually much simpler for end-users to use L<sup>A</sup>T<sub>E</sub>X.

### 4.3.1 Typesetting

L<sup>A</sup>T<sub>E</sub>X is based on the idea that authors should be able to focus on the content of what they are writing without being distracted by its visual presentation. In preparing a L<sup>A</sup>T<sub>E</sub>X document, the author specifies the logical structure using familiar concepts such as chapter, section, table, figure, etc., and lets the L<sup>A</sup>T<sub>E</sub>X system worry about the presentation of these structures. It therefore encourages the separation of layout from content while still allowing manual typesetting adjustments where needed.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\title{\LaTeX}
\date{}
\begin{document}
  \maketitle
  \LaTeX{} is a document preparation system
  for the \TeX{} typesetting program.
  \par
   $E=mc^2$ 
\end{document}
```

# L<sup>A</sup>T<sub>E</sub>X

August 10, 2013

L<sup>A</sup>T<sub>E</sub>X is a document preparation system for the T<sub>E</sub>X typesetting program.  
 $E = mc^2$

Figure 4.5: L<sup>A</sup>T<sub>E</sub>X output of above program.

## 4.3.2 Installing L<sup>A</sup>T<sub>E</sub>X on System

Installation of L<sup>A</sup>T<sub>E</sub>X on personal system is quite easy. As i have used L<sup>A</sup>T<sub>E</sub>X on Ubuntu 13.04 so i am discussing the installation steps for Ubuntu 13.04 here:

- Go to terminal and type

*\$ sudo apt-get install texlive-full*

- Your Latex will be installed on your system and you can check for manual page by typing.

*\$ man latex*

in terminal which gives manual for latex command.

- To do very next step now one should stick this to mind that the document which one is going to produce is written in any type of editor whether it may be your most common usable editor Gedit or you can use vim by installing first vim into your system using command.

*\$ sudo apt-get install vim*

- After you have written your document it is to be embedded with some set of commands that Latex uses so as to give a structure to your document. Note that whenever you wish your document to be looked into some other style just change these set of commands.

- When you have done all these things save your piece of code with .tex format say test.tex. Go to terminal and type

*latex path of the file test.tex Or pdflatex path of the file test.tex*

*eg: pdflatex test.tex*

for producing pdf file simultaneously.

After compiling it type command

*\$ evince filename.pdf*

eg: *evince test.pdf*

To see output pdf file.

### 4.3.3 Graphical Editors for L<sup>A</sup>T<sub>E</sub>X

L<sup>A</sup>T<sub>E</sub>X is not restricted to command line only there are so many graphical based editors available to be used. These GUI based editors provide an easy interface to user so as to do typesetting in an efficient manner. Some of them are listed below:

- Texmaker

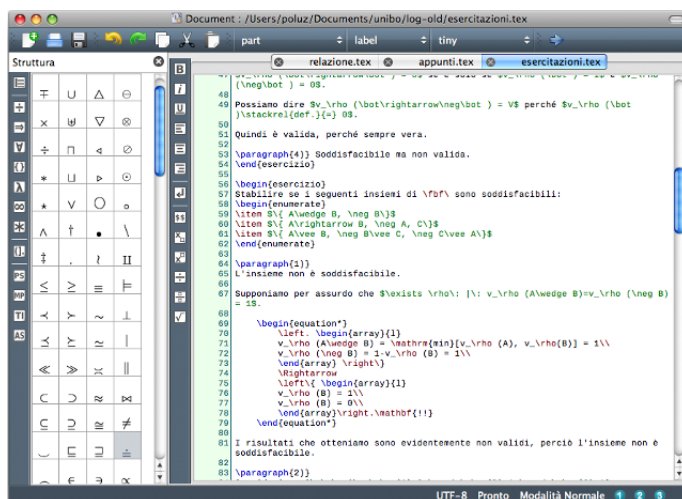


Figure 4.6: Texmaker, A Graphical L<sup>A</sup>T<sub>E</sub>X Editor

- L<sup>E</sup>D

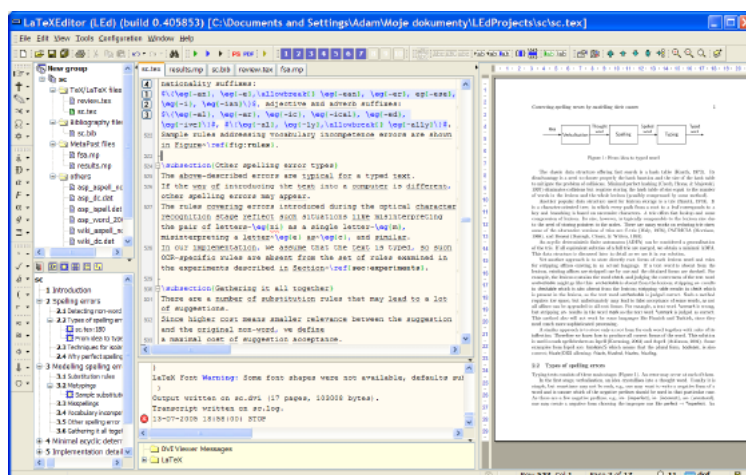


Figure 4.7: L<sup>E</sup>D, A Graphical L<sup>A</sup>T<sub>E</sub>X Editor

And many more but the preferred method to produce L<sup>A</sup>T<sub>E</sub>X document is through console mode only.



### 4.3.4 Pdfscreen L<sup>A</sup>T<sub>E</sub>X

There are some packages that can help to have unified document using L<sup>A</sup>T<sub>E</sub>X. Example of such a package is pdfscreen that let the user view its document in two forms-print and screen. Print for hard copy and screen for viewing your document on screen. Download this package from [www.ctan.org/tex-archive/macros/latex/contrib/pdfscreen/](http://www.ctan.org/tex-archive/macros/latex/contrib/pdfscreen/).

Then install it using above mention method.

Just changing print to screen gives an entirely different view. But for working of pdfscreen another package required are comment and fancybox.

The fancybox package provides several different styles of boxes for framing and rotating content in your document. Fancybox provides commands that produce square-cornered boxes with single or double lines, boxes with shadows, and round-cornered boxes with normal or bold lines. You can box mathematics, floats, center, flushleft, and flushright, lists, and pages.

Whereas comments package selectively include/excludes portions of text. The comment package allows you to declare areas of a document to be included or excluded. One need to make these declarations in the preamble of your file. The package uses a method for exclusion that is pretty robust, and can cope with ill-formed bunches of text.

So these extra packages needed to be installed on system for the proper working of pdfscreen package.

### 4.3.5 Web based graphic generation using L<sup>A</sup>T<sub>E</sub>X

L<sup>A</sup>T<sub>E</sub>X is also useful when there is need of generating the graphics from browser. For example to draw a circle by just entering its radius in html input box. So this kind A of project can be conveniently handled using L<sup>A</sup>T<sub>E</sub>X. Basic idea behind this generation process is that when user clicks on submit button after entering radius a script will run that enter the radius in already made .tex file and recompiles it on server and makes its pdf and postscript file. After that user can view those files by clicking on link provided to view the files. See some screen shots of such a graphic generation project made by Dr. H.S. Rai:

So here in the above input page which is also the index page user can enter input for length of rectangle, breadth of rectangle and for radius of circle after that user can submit the values. After the values get submitted a script get runs by php code at server side. This script first enters the dimensions of rectangle and circle that were selected by user in to an already existing .tex file and replace with the older dimensions there. After that script recompiles the the tex file and make it available for user.

In above figure it gets clear that .tex file has been compiled and pdf and postscript files are available to user and user can download the graphics so produced. Hence graphics can be generated in L<sup>A</sup>T<sub>E</sub>X through web interface.



Figure 4.8: Web based graphic generation using  $\text{\LaTeX}$ (input page)



Figure 4.9: Doxygen Logo

## 4.4 Introduction to Doxygen

Doxygen is a documentation generator, a tool for writing software reference documentation. The documentation is written within code, and is thus relatively easy to keep up to date. Doxygen can cross reference documentation and code, so that the reader of a document can easily refer to the actual code.

Doxygen supports multiple programming languages, especially C++, C, C#, Objective-C, Java, Python, IDL, VHDL, Fortran and PHP.[2] Doxygen is free software, released under the terms of the GNU General Public License.

### 4.4.1 Features of Doxygen

- Requires very little overhead from the writer of the documentation. Plain text will do, Markdown is support, and for more fancy or structured output HTML tags and/or some of doxygen's special commands can be used.
- Cross platform: Works on Windows and many Unix flavors (including Linux and Mac OS X).
- Comes with a GUI frontend (Doxywizard) to ease editing the options and run doxygen. The GUI is available on Windows, Linux, and Mac OS X.
- Automatically generates class and collaboration diagrams in HTML (as clickable image maps) and  $\text{\LaTeX}$  (as Encapsulated PostScript images).
- Allows grouping of entities in modules and creating a hierarchy of modules.

- Doxygen can generate a layout which you can use and edit to change the layout of each page.
- Can cope with large projects easily.

## thepoet.me

Main Page	Namespaces	Classes	Files
-----------	------------	---------	-------

### django-accounts

Custom user profile model including views for sign-up, e-mail activation and profile management.

**Requirements:**

- django-avatar
- django-countries

Generated on Mon Dec 18 2017 21:50:32 for thepoet.me by [doxygen](#) 1.8.11

Figure 4.10: Documentation using Doxygen (main page)

## thepoet.me

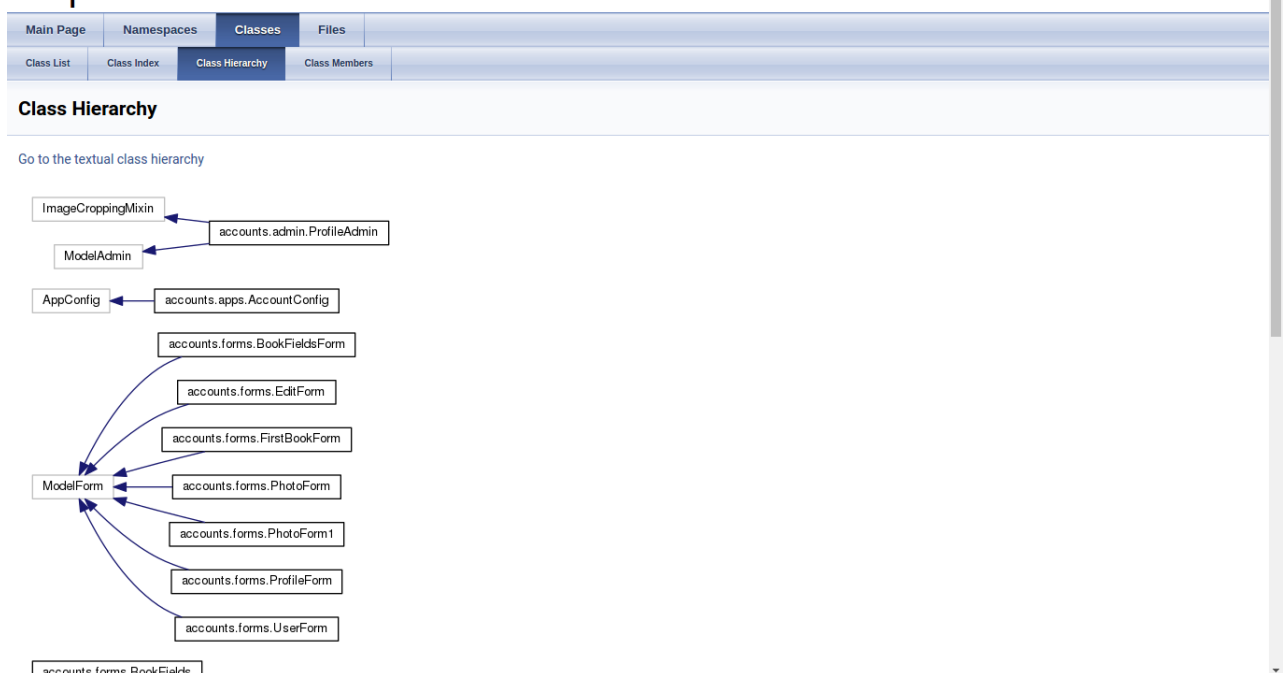


Figure 4.11: Doxygen documentation of a class hierarchy

## 4.5 Introduction to Github



Figure 4.12: Github Logo

GitHub is a Git repository web-based hosting service which offers all of the functionality of Git as well as adding many of its own features. Unlike Git which is strictly a command-line tool, Github provides a web-based graphical interface and desktop as well as mobile integration. It also provides access control and several collaboration features such as wikis, task management, and bug tracking and feature requests for every project.

GitHub offers both paid plans for private reposito handle everything from small to very large projects with speed and efficiency. ositories, and free accounts, which are usually used to host open source software projects. As of 2014, Github reports having over 3.4 million users, making it the largest code host in the world.

GitHub has become such a staple amongst the open-source development community that many developers have begun considering it a replacement for a conventional resume and some employers require applications to provide a link to and have an active contributing GitHub account in order to qualify for a job.

The Git feature that really makes it stand apart from nearly every other Source Code Management (SCM) out there is its branching model.

Git allows and encourages you to have multiple local branches that can be entirely independent of each other. The creation, merging, and deletion of those lines of development takes seconds.

This means that you can do things like:

- **Frictionless Context Switching.**  
Create a branch to try out an idea, commit a few times, switch back to where you branched from, apply a patch, switch back to where you are experimenting, and merge it in.
- **Role-Based Codelines.**  
Have a branch that always contains only what goes to production, another that you merge work into for testing, and several smaller ones for day to day work.
- **Feature Based Workflow.**  
Create new branches for each new feature you're working on so you can seamlessly switch

back and forth between them, then delete each branch when that feature gets merged into your main line.

- Disposable Experimentation.  
Create a branch to experiment in, realize it's not going to work, and just delete it - abandoning the work with nobody else ever seeing it (even if you've pushed other branches in the meantime).

Notably, when you push to a remote repository, you do not have to push all of your branches. You can choose to share just one of your branches, a few of them, or all of them. This tends to free people to try new ideas without worrying about having to plan how and when they are going to merge it in or share it with others.

There are ways to accomplish some of this with other systems, but the work involved is much more difficult and error-prone. Git makes this process incredibly easy and it changes the way most developers work when they learn it.

### 4.5.1 What is Git?



Figure 4.13: Git Logo

Git is a distributed revision control and source code management (SCM) system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows. Git was initially designed and developed by Linus Torvalds for Linux kernel development in 2005, and has since become the most widely adopted version control system for software development.

As with most other distributed revision control systems, and unlike most clientserver systems, every Git working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server. Like the Linux kernel, Git is free and open source software distributed under the terms of the GNU General Public License version 2 to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

### 4.5.2 Installation of Git

Installation of git is a very easy process. The current git version is: 2.0.4. Type the commands in the terminal:

```
$ sudo apt-get update
```

```
$ sudo apt-get install git
```

This will install the git on your pc or laptop.

### 4.5.3 Various Git Commands

Git is the open source distributed version control system that facilitates GitHub activities on your laptop or desktop. The commonly used Git command line instructions are:-

#### 4.5.3.1 Create Repositories

Start a new repository or obtain from an exiting URL

```
$ git init [ project-name ]
```

Creates a new local repository with the specified name

```
$ git clone [url ]
```

Downloads a project and its entire version history

#### 4.5.3.2 Make Changes

Review edits and craft a commit transaction

```
$ git status
```

Lists all new or modified files to be committed

```
$ git diff
```

Shows file differences not yet staged

```
$ git add [file ]
```

Snapshots the file in preparation for versioning

```
$ git reset [file ]
```

Unstages the file, but preserve its contents

```
$ git commit -m "[descriptive message ]"
```

Records file snapshots permanently in version history

#### 4.5.3.3 Group Changes

Name a series of commits and combine completed efforts

```
$ git branch
```

Lists all local branches in the current repository

**\$ git branch [branch-name ]**

Creates a new branch

**\$ git checkout [branch-name ]**

Switches to the specified branch and updates the working directory

**\$ git merge [branch ]**

Combines the specified branches history into the current branch

**\$ git branch -d [branch-name ]**

Deletes the specified branch

#### 4.5.3.4 Save Fragments

Shelve and restore incomplete changes

**\$ git stash**

Temporarily stores all modified tracked files

**\$ git stash pop**

Restores the most recently stashed files

**\$ git stash list**

Lists all stashed changesets

**\$ git stash drop**

Discards the most recently stashed changeset

#### 4.5.3.5 Synchronize Changes

Register a repository bookmark and exchange version history

**\$ git fetch [bookmark ]**

Downloads all history from the repository bookmark

**\$ git merge [bookmark /[branch]]**

Combines bookmarks branch into current local branch

**\$ git push [alias [branch]]**

Uploads all local branch commits to GitHub

**\$ git pull**

Downloads bookmark history and incorporates changes

## 4.6 Implementation

Development of thepoet.me started with development in phases which focus on particular need of project. Various phases and their detail are given below:

- **Phase I (The model layer):**  
Django provides an abstraction layer (the "models") for structuring and manipulating the data of your Web application. In this phase, I worked on data structures of thepoet.me.
- **Phase II (The view layer):**  
Django has the concept of views to encapsulate the logic responsible for processing a users request and for returning the response. In this phase I worked on the following things:
  - **Writing views:** A view function, or view for short, is simply a Python function that takes a Web request and returns a Web response. This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image or anything, really. The view itself contains whatever arbitrary logic is necessary to return that response. This code can live anywhere you want, as long as its on your Python path. Theres no other requirementno magic, so to speak. For the sake of putting the code somewhere, the convention is to put views in a file called views.py, placed in your project or application directory.
  - **Mapping URLs to views:** So, to recap, this view function returns an HTML page that includes the current date and time. To display this view at a particular URL, youll need to create a URLconf; see URL dispatcher for instructions.
  - **Returning errors:** Returning HTTP error codes in Django is easy. There are subclasses of HttpResponse for a number of common HTTP status codes other than 200 (which means OK). You can find the full list of available subclasses in the request/response documentation. Just return an instance of one of those subclasses instead of a normal HttpResponse in order to signify an error.
- **Last phase (The template layer):** In this last phase, I worked on look & feel of the thepoet.me. The template layer provides a designer-friendly syntax for rendering the information to be presented to the user.

## 4.7 Web pages in thepoet.me

### 4.7.1 Website

Figure 4.14 is the main landing page of thepoet.me.

### 4.7.2 User sign up page

In this page Figure 4.15, user register his/her account in thepoet.me.

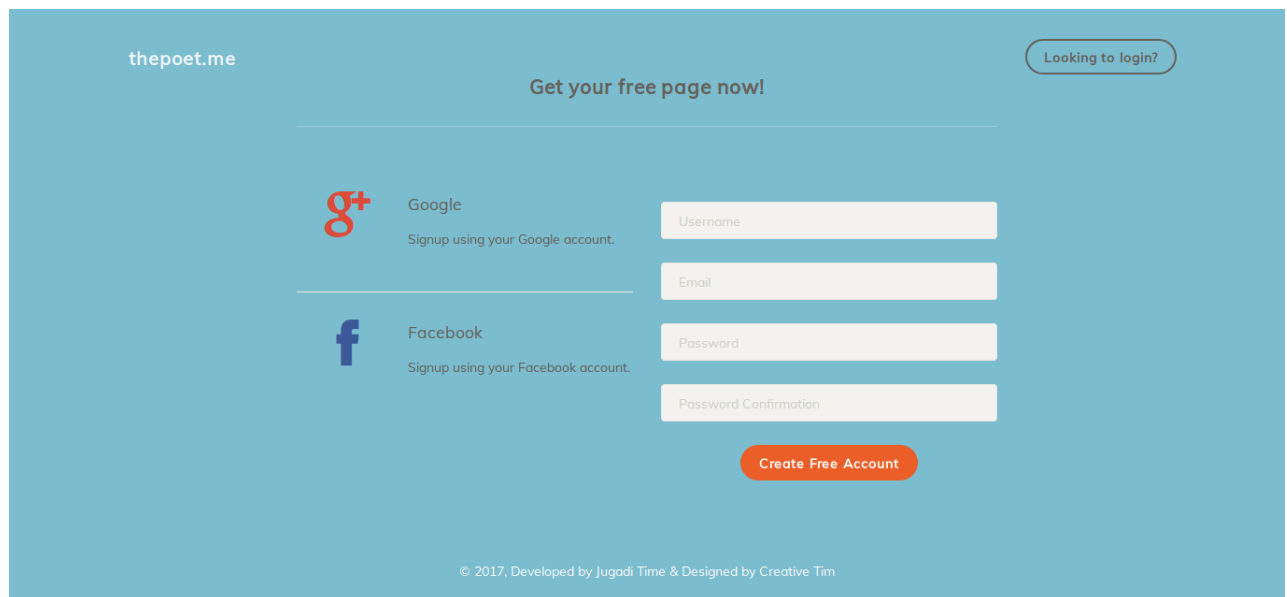
### 4.7.3 User login page

Registered user can open his/her dashboard of thepoet.me. Figure 4.16



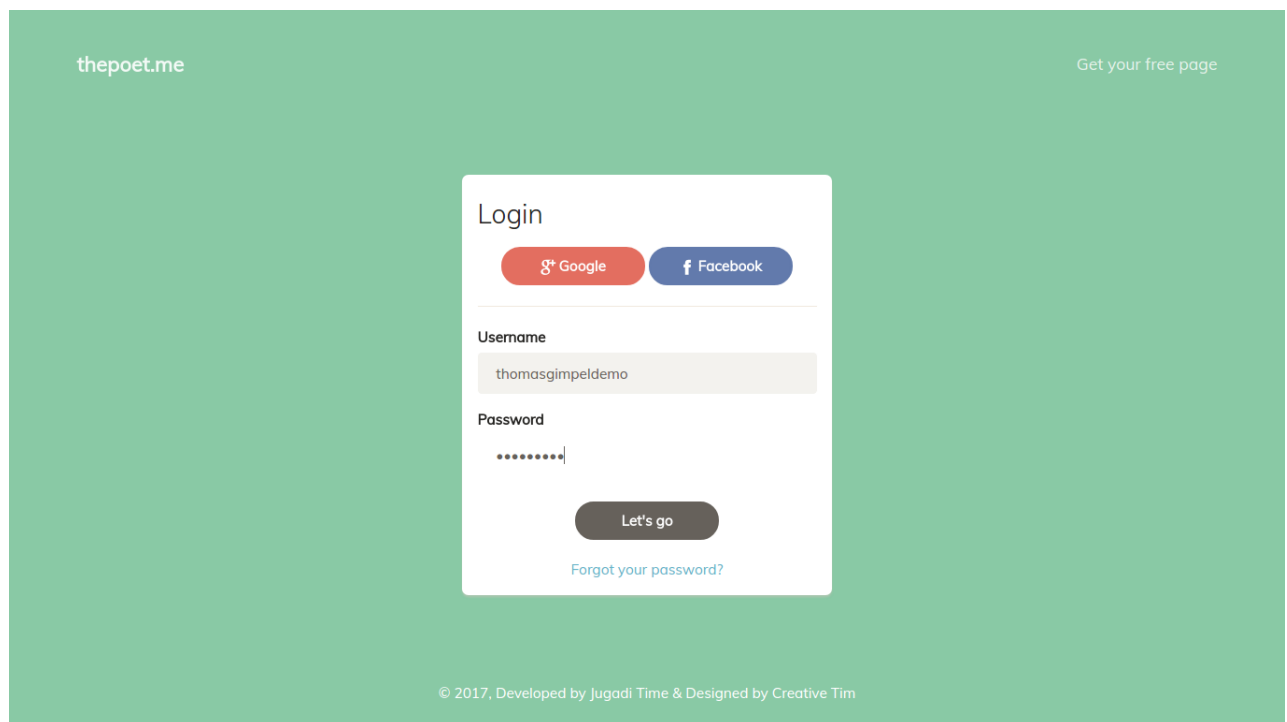


27



The sign-up page for thepoet.me has a light blue background. In the top left corner is the logo 'thepoet.me'. In the top right corner is a button labeled 'Looking to login?'. Centered at the top is the text 'Get your free page now!'. Below this, there are two main sections for social media sign-up. The first section is for Google, featuring the 'g+' logo, the text 'Google', and 'Signup using your Google account.' To the right of this text are three input fields: 'Username', 'Email', and 'Password'. The second section is for Facebook, featuring the 'f' logo, the text 'Facebook', and 'Signup using your Facebook account.' To the right of this text are two input fields: 'Password' and 'Password Confirmation'. Below these sections is an orange button labeled 'Create Free Account'. At the bottom center, there is a small copyright notice: '© 2017, Developed by Jugadi Time & Designed by Creative Tim'.

Figure 4.15: Sign up page



The login page for thepoet.me has a green background. In the top left corner is the logo 'thepoet.me'. In the top right corner is the text 'Get your free page'. Centered on the page is a white login form. The form has a title 'Login' at the top. Below the title are two buttons: a red one with the 'g+' logo and 'Google' text, and a blue one with the 'f' logo and 'Facebook' text. Below these buttons are two input fields. The first is labeled 'Username' and contains the text 'thomasgimpeldemo'. The second is labeled 'Password' and contains a series of dots. Below the password field is a dark grey button labeled 'Let's go'. At the bottom of the form is a link that says 'Forgot your password?'. At the bottom center of the page, there is a small copyright notice: '© 2017, Developed by Jugadi Time & Designed by Creative Tim'.

Figure 4.16: Login page

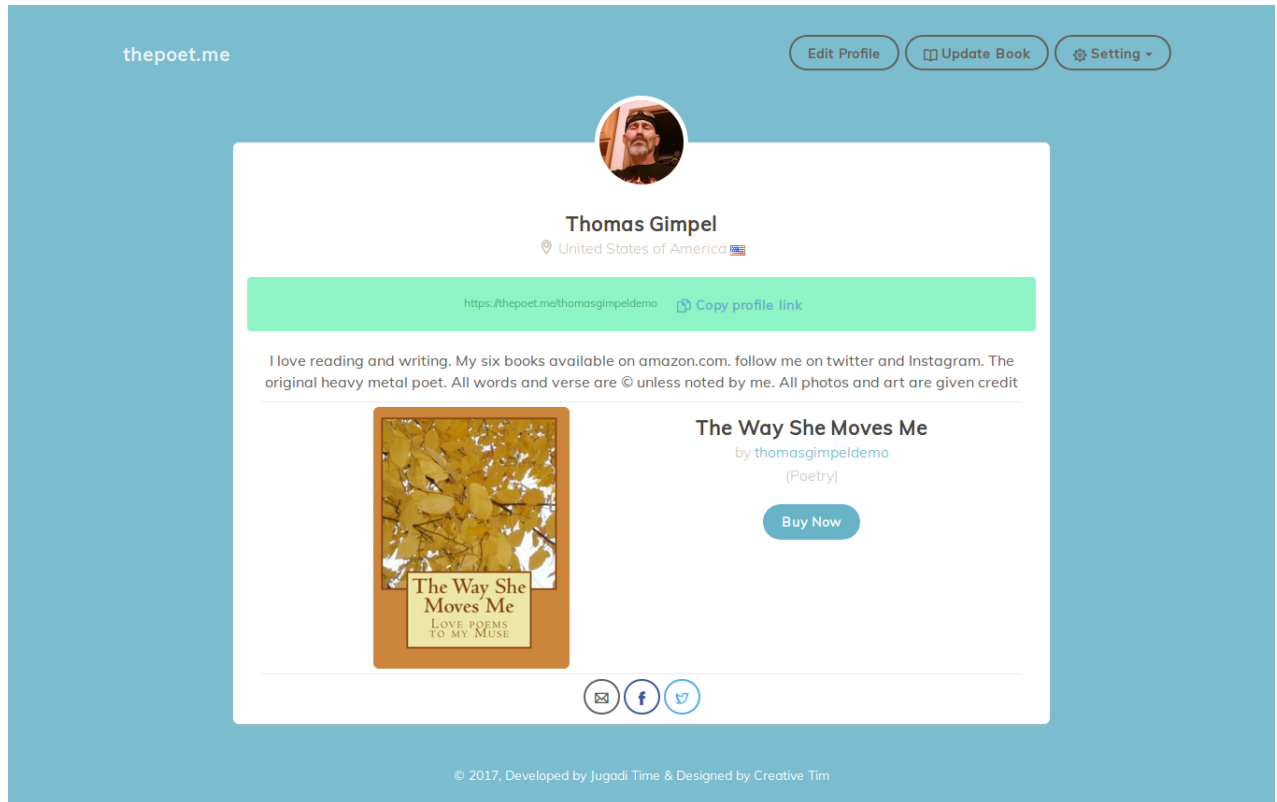


Figure 4.17: User's profile page

#### 4.7.4 User profile page

After login, thepoet.me redirect user to his/her profile page. Figure 4.17

#### 4.7.5 Add book page

Here user can add his/her book to its profile. Figure 4.18

#### 4.7.6 User edit profile page

User can also edit his/her profile. Figure 4.19

#### 4.7.7 Forget password page

User can also change his/her password. Figure 4.20

### 4.8 Testing

Software testing is a process of executing a program or application with the intent of finding the software bugs. It can also be stated as the process of validating and verifying that a software program or application or product: Meets the business and technical requirements that guided it's design and development.

Test Suit for thepoet.me is developed using following two software:

thepoet.me

Back to profile

**Book Name**

The Way She Moves Me

**Book Genre**

Poetry

**Link to Buy**

https://www.amazon.com/Way-She-Moves-Me-poems/dp/1546799516

Update Details

Change Book Cover

© 2017, Developed by Jugadi Time & Designed by Creative Tim

Figure 4.18: Add book page

thepoet.me

View Profile Update Book Setting

**Edit Profile**

Change profile photo

**Username**

thomasgimpeldemo

**Email address**

parmodramniwas@gmail.com

**First Name**

Thomas

**Last Name**

Gimpel

**About Me**

I love reading and writing. My six books available on amazon.com. follow me on twitter and Instagram. The original heavy metal poet. All words and verse are © unless noted by me. All photos and art are given credit

f / thomas.gimpel.12

ig / username

twitter / thomas\_gimpel

**Country**

United States of America

Update Profile

© 2017, Developed by Jugadi Time & Designed by Creative Tim

Figure 4.19: User's edit profile page

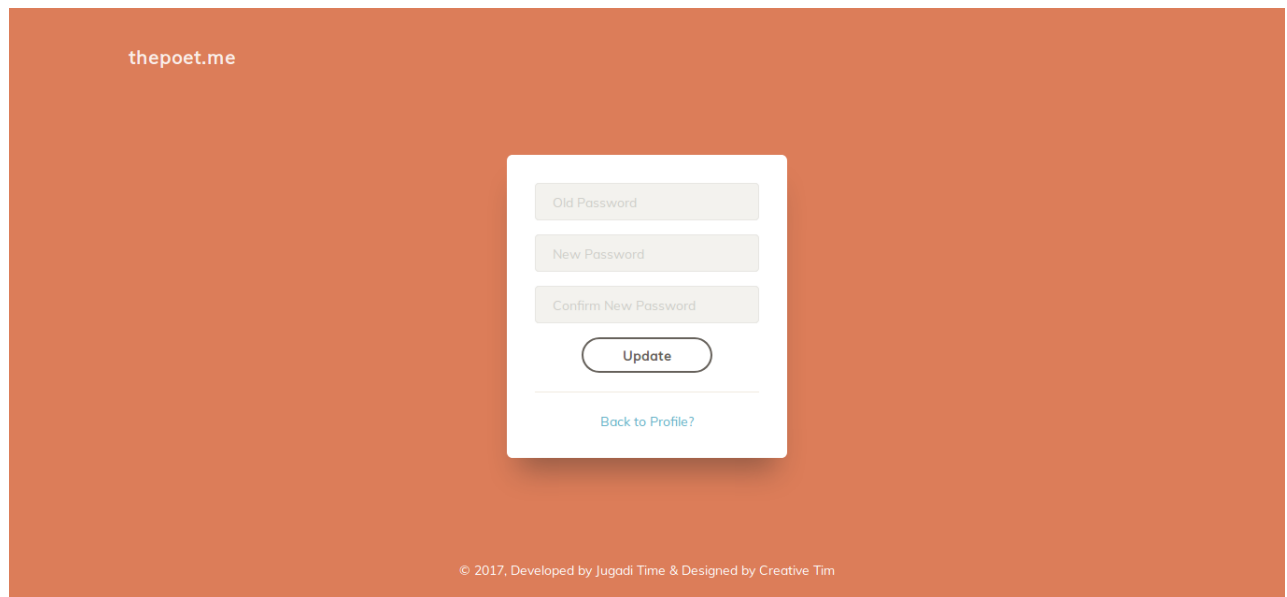


Figure 4.20: Forget password page

1. **dareboost:** One tool to test, analyze and monitor speed and user experience of web pages. Figure 4.21
2. **SEO Tool:** A good search engine optimization strategy is based on careful planning, a lot of work and good choices made on key issues. Everything will get analyzed at the end of each month, when the time comes to send out monthly reports to your customers. Figure 4.22

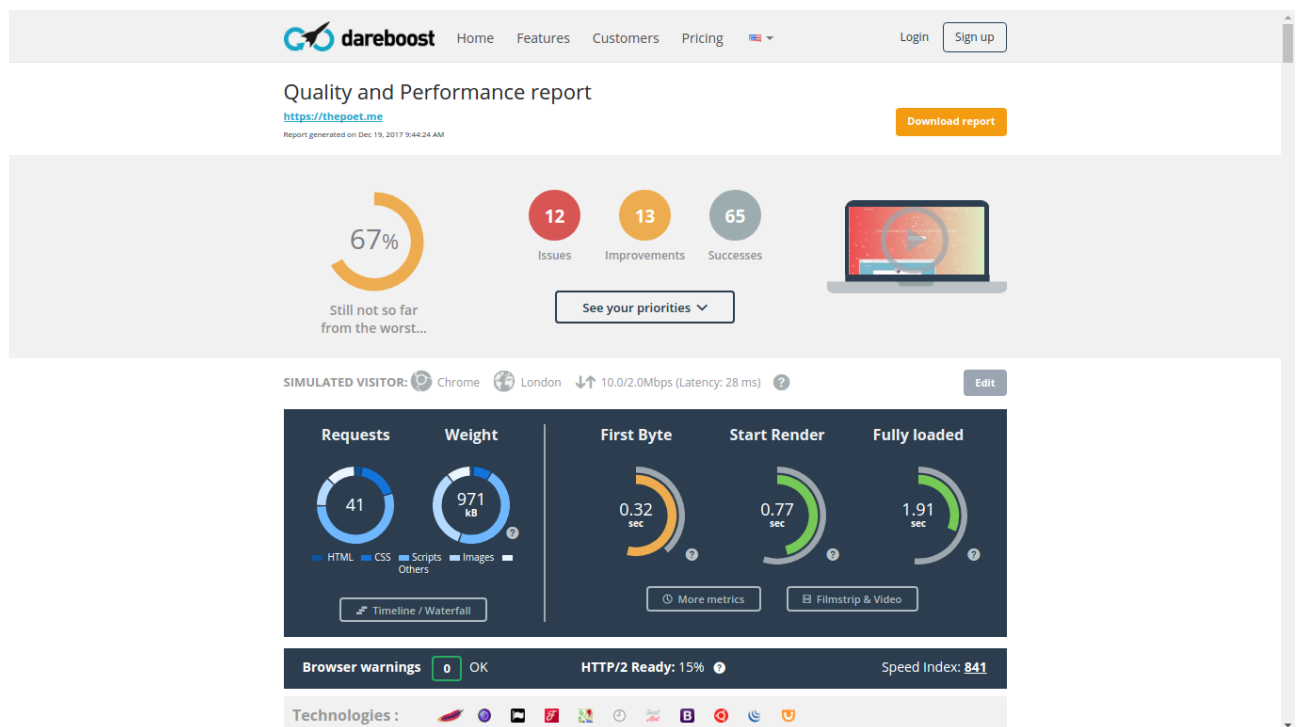


Figure 4.21: Performance report by dareboost

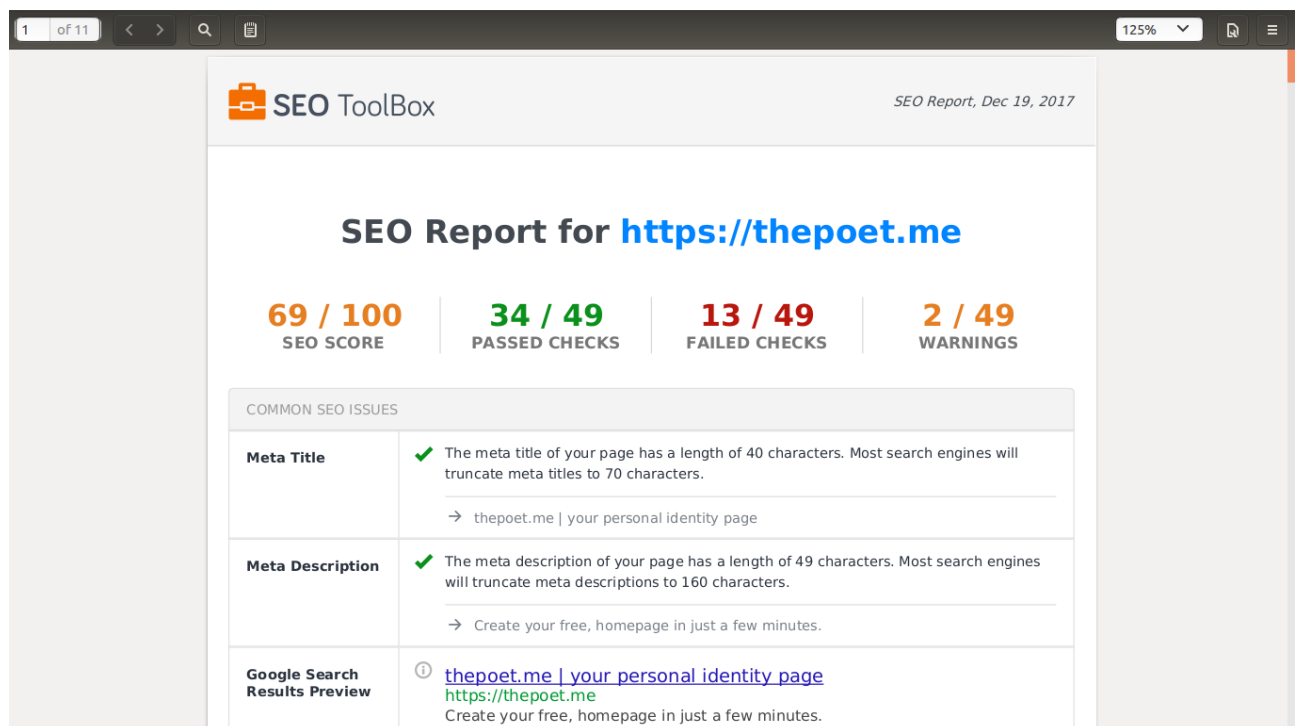


Figure 4.22: SEO report

## 5.1 Conclusion

This project was something which required in-depth view in django. Working on this project has taught me how to develop a social application. This project is aimed at providing user a simple platform from which he/she can link his/her published book and popular social networking websites such as Facebook, Twitter and Instagram. For better user experience, there is change in colour of notification which will indicate any error. Working with the Open Source Community and a variety of people of different age group, one is always challenged by the fundamental difference between classroom coaching and real World experience. But such a challenge is exactly the purpose of six months training. The whole experience of working on this project and contributing to a few others has been very rewarding as it has given great opportunities to learn new things and get a firmer grasp on already known technologies. Here is a reiteration of some of the technologies I have encountered, browsed and learned:

1. Operating System: Linux
2. Language: Python, L<sup>A</sup>T<sub>E</sub>X
3. FrameWork: Django
4. API: Faebook Graph API, Social Auth
5. Softwares: Git, Doxygen
6. Communication tools: IRC, Gitter

So during this project, I learned all the above things. Above all, I got to know how software is developed and how much work and attention to details is required in building even the most basic of components of any project. Apart from above I also learned things like:

1. Planning
2. Designing
3. Developing code
4. Working in a team
5. Testing
6. Licensing Constraints
7. How Open source community work?
8. Writing Readable code
9. Coding Standards

And these are all very precious lessons in themselves.

## 5.2 Future Scope

This project is an open source project and supported by a large open Source community have a lot of scope for future improvements and additions as other individuals can also contribute in it and add additional functionality. Being an Open Source project there is a constant flow of suggestion and demands by people for additional functionality and improvement in existing features. At present a user can upload only one book, this will changed as development of the project continues. Here is a small list of the Features that would be added in near future.

1. User will be able to upload infinite number of books.
2. Also make an cross platform application for mobiles.



- [1] "Jagjeet Singh", Blog, 2017. [Online]. Available: <https://jagjeet.me>. [Accessed: 27- Nov- 2017].
- [2] "Doxygen: Main Page", Doxygen.org, 2017. [Online]. Available: <http://www.doxygen.org>. [Accessed: 27- Nov- 2016].
- [3] "Django" docs, 2017. [Online]. Available: <https://www.djangoproject.com>. [Accessed: 27- Nov- 2016].
- [4] "Git", En.wikipedia.org, 2017. [Online]. Available: <https://en.wikipedia.org/wiki/Git>. [Accessed: 27- Nov- 2017].
- [5] "Python", En.wikipedia.org, 2017. [Online]. Available: <https://en.wikipedia.org/wiki/Python>. [Accessed: 27- Nov- 2017].