ISSN: 2454 - 4248
62 – 68

A Survey: Django Framework

Darshana Anil Chopade
Computer Science and Engineering
Mauli Group of Institution's College of Engineering & Technology
Shegaon, India
e-mail: darshana24c@gmail.com

Shweta Panjabrao Deshmukh
Computer Science and Engineering
Mauli Group of Institution's College of Engineering & Technology
Shegaon, India
e-mail: deshmukhshweta588@gmail.com

Mayuri Sanjay Khune
Computer Science and Engineering
Mauli Group of Institution's College of Engineering & Technology
Shegaon, India
e-mail: mayurikhune11@gmail.com

Chaitanyakumar Mahadeo Manwar
Computer Science and Engineering
Mauli Group of Institution's College of Engineering & Technology
Shegaon, India
e-mail: chaitanyakumarmanwar@gmail.com

Prof. Niraj.N.Kasliwal
Computer Science and Engineering
Mauli Group of Institution's College of Engineering & Technology
Shegaon, India
e-mail: kasliwaln@gmail.com

***Abstract*—**Web application framework is use by web developers for building applications that interface with Web and database servers. Some of the more popular frameworks and their associated languages include Rails Ruby, Spring MVC (Java) and Struts (Java). Django is a framework which provides much more support for Web applications than older CGI (Common Gateway Interface) libraries. Django is also open source and free and it follows the principle of "Don't Repeat Yourself". Similarly, it includes an API for generating HTML forms and retrieving data from them. It is a major web framework for Python developers these days and it's not too hard to see why, as it excels in hiding a lot of the configuration logic and letting you focus on being able to build application, quickly.

Keywords-Django, MVT,

*****

I. INTRODUCTION

A. Django

Written in pure Python, Django has a clean pythonic structure. It started as Model–View–Template (MVT) framework, and this concept still exists in the current version. People with little understanding of Django think it's a mere content management system. In reality, it's a software tool designed to build and run web applications. The origin of the framework's name is key to understanding its multifaceted nature. The Django framework owes its name to jazz guitarist Django Reinhardt, who was able to play dazzling runs on his guitar even though two of his fingers were paralyzed after an accident. In the same way, the Django framework can take on numerous tasks.[1]

B. Framework

A framework or software framework is a platform for developing software applications. For example, a framework may include predefined classes and functions that can be used to process input, manage hardware devices, and interact with system software. A framework may also include code libraries, a compiler and other programs used in the software development process. Framework automate the implementation of common solutions, cutting development time and allowing developers to focus more on application logic instead of routine elements. Several different types of software frameworks exist Django is one of them.

C. Django Framework

In the fall of 2003 Django was created when the web programmers at the Lawrence Journal World newspaper, Adrian Holovaty and Simon Willison, began using Python to build applications. Django was released publicly under a BSD license in July 2005. The framework was named after the jazz guitarist Django Reinhardt. Django is a free and open source web application framework, written in Python. A web framework is a set of components that helps you to develop websites faster and easier.[1]

When you're building a website, you always need a similar set of components for handling user authentication (signing up, signing in, signing out), a management panel for your website, forms, a way to upload files, etc. Django is a widely used Python web application framework with a "batteries-included" philosophy. The principle behind batteries-included is that the common functionality for building web applications should come with the framework instead of as separate libraries.

For example, authentication, URL routing, an engine, an object-relational mapper (ORM), and database schema migrations are all included with the Django framework. Compare that included functionality to the Flask framework which requires a separate library such as Flask-Login to perform user authentication. The batteries-included and extensibility philosophies are simply two different ways to tackle framework building. [2]

II. LITERATURE REVIEW

The world of Python web frameworks is full of choices. Django, Zope, CubicWeb, Aquarium and many more are

ISSN: 2454 - 4248
62 – 68

competing for developer mindshare. As a developer we want to cut the legions of options down to the one that will help to finish the project.

A. Zope

Zope was released in 1999. Zope works on Windows, Mac OS and Linux. Two different persistence models are used in the packages reviewed such as Object databases and Relational databases. In Zope, persistence is provided through object classes that inherit from a Persistent class, and a single object database (ZODB) is used to store the data. A transaction model with the possibility of conflicts and retries is provided. ZODB frameworks is having low performance, and that implementers have to build separate indexes (which are a great source of database conflict errors and retries) to speed up searching for data. [3]

Zope puts object instances in container objects, and the container hierarchy has a root. This leads to path-based URL mapping where http://site/a/b/c/d maps to the "d" method on the object tied to 'c' in the object tied to 'b' in the object tied to 'a' in the root container object that is it works like a file-system. Zope comes with 2 template packages, DTML and TAL. DTML is an SGML template language, while TAL is an XML based one. TAL has one advantage above all the other notations here in that it can be loaded and saved cleanly in nearly all HTML editors. The disadvantage of both the Zope template notations is the difficulty in editing them, and the implied context in which they operate, which makes getting the right data into the template renderer sometimes difficult.

B. CubicWeb

CubicWeb is a sematic web application framework that Logilab started developing in 2001. It is supported on platforms Loke UNIX, Mac, Windows, Linux, and Solaris. It has Yams as its inbuilt server. CubicWeb is written in Python and includes a data server and a web engine. A cube is a software component made of three parts: its data model (schema), its logic (entities) and its user interface (views). The data repository encapsulates and groups an access to one or more data sources (including SQL databases, LDAP repository, other CubicWeb instance repositories, file systems, Google App Engine's Data Store, etc.) All interactions with the repository are done using the Relation Query Language.

CubicWeb is a semantic web application framework that Logilab started developing in 2001. It is supported on platforms like UNIX, Mac, Windows, Linux and Solaris. It has Yams as its inbuilt server. CubicWeb is written in Python and includes a data server and a web engine. A cube is a software component made of three parts: its data model (schema), its logic (entities) and its user interface (views). The data repository encapsulates and groups an access to one or more data sources (including SQL databases, LDAP repositories, other CubicWeb instance repositories, file systems, Google App Engine's Data Store, etc.) All interactions with the repository are done using the Relation Query Language. [4]

C. CherryPi

CherryPy is an open-source, minimalist web framework. An HTTP/1.1-compliant WSGI thread pooled webserver. It provides Simplicity of running multiple HTTP servers at once. CherryPy has a powerful configuration system. It comes with flexible plugin system. It has Out-of-the-box tools for caching, encoding, sessions, authentication, static content, and so on. CherryPy has its own server. CherryPy is supported on platforms like Windows, Mac OS and Linux.[5]

Fig 1. CherryPy

D. Aquarium

Aquarium release in 30 july 2004. It is a framework for creating highly dynamic, custom Web applications in Python. It offers convenient libraries, tight integration with Cheetah, adaptors for various Web environments, and a convenient approach to Web development. Web applications can be thought of as a combination of a bunch of different types of modules. Aquarium serves as the framework for dynamically tying all of these modules together. A view's superclass should take care of layout details automatically. In fact, a view should not need to do anything other than declare who it is sub classing. It has an inbuilt Glass server. Aquarium works on UNIX and Windows. [6]

III. OJECTIVE AND SCOPE

A. Objective

The main objective of Django is to build an application based on Python. Its main goals are simplicity, flexibility, reliability, and scalability.

a. *Simplicity*

One of Django's main goals is to simplify work for developers. To do that, the Django framework uses:

1. The principles of rapid development, which means developers can do more than one iteration at a time without starting the whole schedule from scratch.
2. DRY philosophy Don't Repeat Yourself which means developers can reuse existing code and focus on the unique one.

b. *Flexibility*

No restrictions mean that the developer can implement everything exactly as they want it, using a huge range of external libraries and add-ons, making it flexible and extensible.

c. *Reliability*

Django is quite reliable as it consistently making effort for good quality or performance and it is able to be trusted due to its security features.

d. *Scalability*

One of the nicest advantages of Django is that it can handle traffic and mobile app API usage of more than 400 million+ users helping maximize scalability. And when talking about hosting, we need to mention that

ISSN: 2454 - 4248
62 – 68

the number of hosts is high and hosting price is relatively cheap and even free.

.

B. Future Scope for Work

Due to the advanced features and low pricing nature of Software as a Service, we were able to build a fully self-sustaining application with limited intervention and cost-overheads. Due to Python Django app development combo, we have been able to execute complex program logic in simple syntaxes without compromising on robustness. In future many companies will switch to Django as it is growing rapidly. Day by Day changes has been made in Django to improve its quality and give the users more functionality inbuilt in its own framework. Due to the ease of programing provided by DRY principle attracts many clients. A combination of Python and Django has proven to be extremely powerful for application development. There is a future scope in Django for default support to web sockets also its admin panel behavior need improvement as its behavior is hard to tune sometimes.

IV. METHODOLOGY AND DESIGN PROCESS

Django prescribe a certain methodology for building applications, which is helpful in deciphering complex apps.

A. Workflow

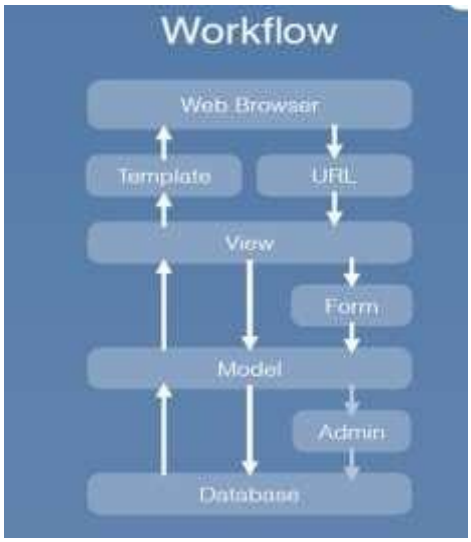

Figure 2. Workflow

Web Browser is users actually see and responds to what users do when they click, type and enter. URL is often referred as the „web addresses. It provides mapping to View. View renders content to Template and get information from Model before rendering content. It put information into Model and Database through Form.HTML Form consists of input element, checkbox, submit button, radio button and much more. Django Form aims to fetch data from html form helps to connect to Model. Model describes the structure of an object you want to store in Database. It goes to Database and create, edit and request information as you wish. Admin helps to register your object in your Model so you can manage data in Database. The registration has to be done in the first place. Database is a collection of data provided with a wonderful back-end platform for easy management template. We can systematically store all of your Html files. You will have a „static" folder to store other CSS files, JavaScript files, or Images.

B. Installation

a. Installing Python

Django is written in 100% pure Python code, so you'll need to install Python on your system. Django requires Python 2.0 or higher version. To download Python visit: http://www.python.org/download/ to get started. The installation is fast and easy.

b. Installing Django

1. pip install virtualenvwrapper-win
2. mkvirtualenv myproject
3. workon myproject
4. pip install Django

C. Running the development Server

Django comes with a lightweight web server to run your code quickly, without needing to spend time configuring a production server. When you run the Django development server, it keeps checking for changes in your code. It reloads automatically, freeing you from manually reloading it after code changes. However, it might not notice some actions, such as adding new files to your project, so you will have to restart the server manually in these cases. Start the development server by typing the following command from your project's root folder:

- *python manage.py runserver*

We should see something like this:

- *Performing system checks...*
- **System check identified no issues (0 silenced).**
- **May 06, 2016 - 17:17:31**
- **Django version 2.0.5, using settings 'mysite.settings'**
- **Starting development server at http://127.0.0.1:8000/**
- **Quit the server with CONTROL-C**.

Now, open http://127.0.0.1:8000/ in your browser. You should see a page stating that the project is successfully running, as shown in following screenshot:

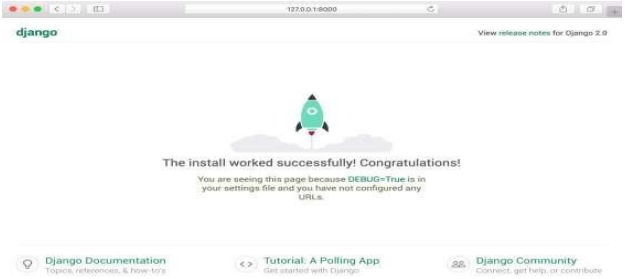

Figure 3. Setting up Django server

ISSN: 2454 - 4248
62 – 68

The preceding screenshot indicates that Django is running. If you take a look at your console, you will see the GET request performed by your browser:

[06/May/2016 17:20:30] "GET / HTTP/1.1" 200 16348

Each HTTP request is logged in the console by the development server. Any error that occurs while running the development server will also appear in the console. [7]

D. Directory Architecture

Start the project by running following command.

Django-admin.py start project projectname

This is the first step towards creating a project. A project is a collection of settings for an instance of Django including database configuration, Django specific options and application-specific settings. These files are described below:

Figure 4. Directory Architecture

- django-admin.py startproject mysite

That"ll create mysite directory in your current directory.

1. **manage.py:** This is a command-line utility used to interact with your project. It is a thin wrapper around the django-admin.py tool. You don't need to edit this file.
2. mysite/**:** This is your project directory.
3. **__init .**py: An empty file that tells Python to treat the mysite directory as a Python module.
4. settings**.py**: This indicates settings and configuration for your project and contains initial default settings.
5. **url.**py**:** This is the place where your URL patterns live. Each URL defined here is mapped to a view.
6. **migrations:** This directory will contain database migrations of your application. Migrations allow Django to track your model changes and synchronize the database accordingly.
7. **/static:** This file stores CSS, JS and images.
8. **/templates:** This file store all your HTML pages
9. **admin.py:** This is where you register models to include them in the Django administration site using the Django admin site is optional.
10. **models.py:** Data models of your application all Django applications need to have a models.py file, but this file can be left empty.
11. **views.py:** The function needed to perform and respond user request. [8]

E. Django Commands:

1. python manage.py startproject project_name. This command is use to start an project.
2. python manage.py start app_name This command is use to start an application.
3. python manage.py createsuperuser. This command is use to create a superuser.
4. python manage.py run server 127.0.0.1:8000 This command is use to start up your server
5. python manage.py makemigrations
   This command is use to create new migration based on the changes you made in your models.
6. python manage.py migrate
   This command is use to apply migration into your database. Django comes with a migration system that tracks the changes done to models and allows propagating them into the database. The migrate command applies migrations for all applications listed in INSTALLED_APPS, it synchronizes the database with the current models and existing migrations.
7. python manage.py –h.
   This command is use to ask for what command can be used.
8. python manage.py runserver –h
   This command is use to ask for what kind of command can be used after runserver.

F. Design your model

Django can be use without a database, it comes with an object-relational-mapper which describe database layout in Python code. The rich ways of representing models offers many rich ways of representing models. Here"s a quick example:

mysite/news/models.py

```
from django.db import models

class Reporter(models.Model):
    full_name = models.CharField(max_length=70)

    def __str__(self):
        return self.full_name

class Article(models.Model):
    pub_date = models.DateField()
    headline = models.CharField(max_length=200)
    content = models.TextField()
    reporter = models.ForeignKey(Reporter, on_delete=models.CASCADE)

    def __str__(self):
        return self.headline
```

Figure 5. Model Example

For running the Django command-line utilities create the database tables automatically:

a. python manage.py make migrations
b. python manage.py migrate

The make migrations command looks at available models and creates migrations for whichever tables don't already exist. The migrate command runs the migrations and creates tables in database, as well as optionally providing much richer schema

ISSN: 2454 - 4248
62 – 68

control. Once the models are defined, Django can automatically create a professional, production ready administrative interface. Through this model the user can users add, change and delete objects. It is very easy as registering model in admin site.

mysite/news/models.py

```
from django.db import models

class Article(models.Model):
    pub_date = models.DateField()
    headline = models.CharField(max_length=200)
    content = models.TextField()
    reporter = models.ForeignKey(Reporter, on_delete=models.CASCADE)
```

Figure 6. Models

mysite/news/admin.py

```
from django.contrib import admin

from . import models

admin.site.register(models.Article)
```

Figure 7. Registering model in admin

The philosophy here is that your site is edited by a staff, or a client, or maybe just you and you don‟t want to have to deal with creating backend interfaces just to manage content. One typical workflow in creating Django apps is to create models and get the admin sites up and running as fast as possible, so your staff (or clients) can start populating data. Then, develop the way data is presented to the public. [8]

G. Adding URL patterns for your Views

An elegant URL scheme is an important detail in a high-quality Web application. Django encourages beautiful URL design and doesn‟t put any complication in URLs, like **.php** or **.asp**. To design URLs for any app, you can create a Python module called as an URLconf. A table of contents contain a simple mapping between URL patterns and Python callback functions. URLconfs also serve to seperate URLs from Python code.

mysite/news/urls.py

```
from django.urls import path

from . import views

urlpatterns = [
    path('articles/<int:year>/', views.year_archive),
    path('articles/<int:year>/<int:month>/', views.month_archive),
    path('articles/<int:year>/<int:month>/<int:pk>/', views.article_detail),
]
```

Figure 8. Adding URL for view

The code above maps URL paths to Python callback functions (“views”). The path strings use parameter tags to “capture” values from the URLs. After requesting for a page, Django runs through each path, in order and stops at the first one that matches the requested URL. It is fast, because the paths are compiled into regular expressions at load time.

The Python function is called when one the URL pattern matches. Each view gets passed a request object which contains request metadata and the values captured in the pattern. For example, if a user requested URL “/articles/2005/05/39323/”, Django would call the function news. views. article detail (request, year=2005, month=5, pk=39323). URL patterns allow you to map URLs to views. A URL pattern is composed of a string pattern, a view and, optionally, a name that allows you to name the URL project-wide. Django runs through each URL pattern and stops at the first one that matches the requested URL. Then, Django imports the view of the matching URL pattern and executes it, passing an instance of the HTTP Request class and keyword or positional arguments. [8]

H. Write your view

Each view is responsible for doing one of two things: Returning an HttpResponse object containing the content for the requested page, or raising an exception such as Http404. The rest is up to you. Generally, a view retrieves data according to the parameters, loads a template and renders the template with the retrieved data. Here‟s an example view for year_archive. [8]

mysite/news/views.py

```
from django.shortcuts import render

from .models import Article

def year_archive(request, year):
    a_list = Article.objects.filter(pub_date__year=year)
    context = {'year': year, 'article_list': a_list}
    return render(request, 'news/year_archive.html', context)
```

Figure 9. Views

I. Creating template for your view

Django has a powerful template language that allows you to specify how data is displayed. It is based on template tags, template variables, and template filters.Template tags control the rendering of the template and look like {% tag %}.Template variables get replaced with values when the template is rendered and look like {{ variable }}.Template filters allow you to modify variables for display and look like {{ variable|filter }}. [8]

Django provides template search path, which allows you to minimize redundancy among the templates. In Django settings, specify a list of directories to check for templates with DIRS. If a template doesn‟t exist in the first directory, it checks the second, and so on. Let‟s say the news/year_archive.html template was found.

ISSN: 2454 - 4248
62 – 68

mysite/news/templates/news/year_archive.html

```
{% extends "base.html" %}

{% block title %}Articles for {{ year }}{% endblock %}

{% block content %}
<h1>Articles for {{ year }}</h1>

{% for article in article_list %}
    <p>{{ article.headline }}</p>
    <p>By {{ article.reporter.full_name }}</p>
    <p>Published {{ article.pub_date|date:"F j, Y" }}</p>
{% endfor %}
{% endblock %}
```

Figure 10. Template

Variables are surrounded by double-curly braces. {{ article.headline }} means "Output the value of the article's headline attribute." But dots aren't used only for attribute lookup. They also can do dictionary-key lookup, index lookup and function calls. Finally, Django uses the concept of "template inheritance". That's what the {% extends "base.html" %} does. It means First load the template called base, which has defined a set of blocks, and fill the blocks with the following blocks." In short, that lets you dramatically cut down on redundancy in templates: each template has to define only what's unique to that template. [8] The "base.html" template, includes the use of static files, which might look like:

mysite/templates/base.html

```
{% load static %}
<html>
<head>
    <title>{% block title %}{% endblock %}</title>
</head>
<body>
    <img src="{% static "images/sitelogo.png" %}" alt="Logo">
    {% block content %}{% endblock %}
</body>
</html>
```

Figure 11. Base File

This describes the overview of Django Framework.

V. OBSERVATION

A. *Django Community*

Django an open source and available for free online, Django is supported by active volunteers who constantly provide updates and resources on djangoproject.com and on Github (in the last, there are 11,300 Django stars). As well, people in the community are cool and they support each other via Mailing list, IRC channel, Blog posts and Stackoverflow.

Figure 12. Django Community

B. *Django's Scalability*

One of the key features of Django is that it can handle traffic and mobile app API usage of more than 400 million+ users helping maximize scalability and minimize web hosting costs. Today, the Django community unites over 11,000 developers across 166 countries.

Figure 13. Scalability

C. *Django Over globe*

Django is a gaining attraction from many developers and it is spreading raidly. Django started off with great documentation, the best of any other open-source framework. And it"s still maintained on a high level, updated along with the new functions and fixes, so you can easily adapt to changes. Hence Django is preferred by many of the developers.

Figure 14. Websites based on Django

VI. ADVANTAGES

1. **Rich** ecosystem**:** There are many third-party applications that come with Django. These applications can be integrated depending on project requirements. Django consists of many applications such as for

ISSN: 2454 - 4248
62 – 68

authorization and sending emails that can easily be plugged into a system.

2. **Maturity**: Most importantly, when we're trying to figure out how something should work in Django, we can usually find the answer. Thousands of people must have already solved any issue you're dealing with, and you can find a solution provided by the passionate Django community.
3. **Admin panel by default**. A Django admin panel is generated automatically from Python code. There's a lot of room for customization in the Django admin panel due to third-party applications. Additionally, Django allows you to modify the interface with third-party wrappers and add dashboards unique to your needs.
4. **Good for SEO.** Python is famous for having human-readable code, and that's an advantage if you want your site to rank high in search results. With Django, you can generate readable website URLs and links using the most relevant keywords and Search Engine Optimization (SEO) best practices.
5. **Pluggable.** Django is pluggable by nature and can be extended with plugins. In future, if we didn't **want** the plugin we can unplug and replace them with other components.
6. **Libraries:** Django allows developers to use libraries when building any project. Some popular libraries include the Django REST framework, which is responsible for building application programming interfaces (APIs); Django CMS, which is designed to manage website content and Django-allauth, which is an integrated set of Django applications for authentication, registration, account management, and third-party (social) account authentication.

VII. DISADVANTAGES

1. **Not suited for small-scale projects:** Django sometimes can be excessive, but Python allows you to use other frameworks to develop simple solutions. For example, if you need to design a simple chat, Django can be too big of a framework and you can instead go with Flask, a microservice framework.
2. **No default support for Web Sockets:** Web Sockets allow you to update information or events in real time. Django doesn't support real-time web applications yet. Therefore, you need to use other frameworks like aiohttp.
3. **Hard to replace:** Some internal Django modules, such as ORM and forms, are hard to replace; it will **require** a lot of effort from your developers to change the internal structure.
4. **Django's behavior is sometimes hard to tune:** Some internal Django modules such as the admin panel **are** hard to tune because of Django's philosophy. For example, if you want to add a link, dynamic statistics, or something unique that isn't included in the Django ecosystem, this can literally take hours. So don't be surprised when you get the bill.

VIII. CONCLUSION

Django's advantages outweigh the disadvantages. As Django is written in Python language which is truly simple to learn and seems as it was created for newbies. Also Django aims to follow Python's "batteries included" philosophy. It means Django provides a wide range of features and functionalities. The administration interface provided by Django is one of the coolest things. It's truly simple to create and it"s really one of the key advantages when using the framework. Django is supported by active volunteers who constantly provide updates and resources on djangoproject.com and on Github. This makes Django more popular among people.

IX. REFERENCES

[1] *https://steelkiwi.com/blog/why-django-best-web-framework-your-project/*

[2] Prof. B *Nithya* Ramesh, Aashay R Amballi, and Vivekananda Mahanta,"*Django the Python web framework" International Journal of Computer Science and Information Technology Research ISSN 2348- 120X (online) Vol. 6,*

[3] *https://zope.readthedocs.io/en/latest/*

[4] CubicWeb Documentation

[5] CherryPy Documentation

[6] *https://wiki.python.org/moin/Aquarium*

[7] Dhanya Baburaj and Safis Editing,"*Django 2 by Example*" Published by *Packt Publishing Ltd. Livery Place,35 Livery Street,Birmingham,B3 2PB, UK.,ISBN 978-1-78847-248-7,Production reference: 1250518 May 2018*

[8] *https://docs.djangoproject.com/en/2.2/*