Available Online at www.ijcsmc.com/journal

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 11, Issue. 4, April 2022, pg.36 – 48

PYTHON FOR WEB DEVELOPMENT

Dr. Uday Patkar (HOD Computer Dept.)
Priyanshu Singh; Harshit Panse; Shubham Bhavsar; Chandramani Pandey
Department of Computer Engineering, Bharati Vidyapeeth's College of Engineering, Lavale, Pune

DOI: https://doi.org/10.47760/ijcsmc.2022.v11i04.006

Abstract— With evolution of web, several competitive languages such as Java, PHP, Python, Ruby are catching the attention of the developers. Recently Python has emerged as a popular and the preferred web programming language, because of its simplicity to code and ease of learning. Being a flexible language, it offers fast development of web-based applications. It offers development using CGI and WSGI. Web development in Python is aided by the powerful frameworks such as Django, web2py, Pyramid, and Flask that Python supports. Thus, Python promises to emerge as one of the preferred choice language for web applications.

Web is a rapidly growing repository of resources. Internet is used as a medium for accessing these resources. Web architecture mainly comprises of two entities, namely client and server. Web client is an application (browser) on host machine that urges these resources, and web server is a machine on web that is responsible for fulfilling the request issued by client. Hypertext Transfer Protocol (HTTP) is the most popular protocol used by client and server for web communication. In a static web, browser issues HTTP request to the HTTP server, which searches for the requisite resource in its database and returns it as an HTTP response. To avoid any compatibility issues, every request issued by browser is in form of a URL (Uniform Resource Locator). The URL protocol defines the rules for communication between client and server. It comprise of host name (IP address) which helps in identifying the server system on the web, port number which determines the service (for example, FTP, email service) on the server that should respond to request, and the access path of the resource (web page) on server. The web where responses are already stored in server database in form of static web pages is termed static web. However, response returned by server to the client may be generated on the fly depending upon the request of the client.

Web applications offer several benefits over traditional applications that are required to be installed at each host computer that wishes to use them. Web applications do not incur publishing and distribution costs as opposed to traditional applications where the software (applications) were published using CD's and distributed. They need not be installed at each client host; rather they are placed at a central server and accessed by large number of clients. Since a web application is managed centrally, application updates and data backups can be performed easily. The web applications are easily accessible irrespective of the boundaries of space and time. Since, they are accessed through browser, the platform accessing them is not an issue, and thus they provide cross-platform compatibility. Inspite of above- mentioned advantages, web applications have a few limitations. Internet connectivity and server availability is required for accessing web application through browser. However, accessing them through Internet my take more time as compared to applications installed on host systems. Also, web applications require compatible web browsers. Since they are deployed on web, they are vulnerable to several Internet attacks.

Web programming using CGI and WSGI requires building web applications from the scratch by using Python standard libraries. Python provides with web frameworks in the form of packages/ modules that simplify the task of writing application programs. These frameworks lighten tedious job of developers. They

support server and client side programming by providing support for several activities such as request interpretation (getting form parameters, handling cookies and sessions), response generation (generating data in HTML or other format such as pdf, excel), and storing data. The web frameworks are further categorized as full- stack and non-full-stack frameworks. Full-stack frameworks provide components for every phase of programming in contrast to non-full-stack frameworks.

All the frameworks include templates and data persistence as key ingredients for constructing web. Templates are used to avoid complex code that results when HTML and Python code is mixed in a single file. Templates are HTML files with placeholder for the data depending upon user input. Data persistence deals with storing and retrieving data and maintaining consistency. The data can be stored and maintained using plain text files, relational database engines such as MYSQL, Oracle, or some object-oriented databases. The web framework providing support for WSGI should be preferred. This makes deploying an application easier.

Keywords— Python, Web server application development, Django frameworks, Model View Controller, MVC, MVT, WSGI, CGI, PHP, Flask, Pyramid, Web2py, PyQt, Artificial Intelligence, Machine Learning, Data Science and Visualization, Web Scraping

---

I. INTRODUCTION

1. Web Development:

The critical point was a few years ago. We all thought that high-speed Internet and advanced browsers would allow us to create unwieldy web pages. However, mobile devices changed this world. Lightweight, portable, and adaptive are the three factors that shape a website Development these days. Most users prefer lightweight devices, even if they have a powerful desktop. Also, people can use different devices to visit a website.

Ionic, the JavaScript versatile system instead of Angular, is a significant marker for what modern web development really resembles. Angular is not necessarily dead. There are a lot of reasons for people to continue use Angular over React.js and other JavaScript apparatuses if the utilization case is right.

On the other hand, a website needs to deal with an offset of complex necessities with execution and effortlessness. That is why web developers do not always use technologies that give a simple and templated approach to manufacture portable and responsive websites.

1.1 Web Components:

A few years ago, web developers used complex HTML and complicated frameworks to render custom UI controls. These tasks could be completed through reusing code; however using code multiple times can turn your page into a mess. Web Components aims to solve such problems.

Web Components are a set of features that provide a standard component model for the Web allowing for encapsulation and interoperability of individual HTML elements with technologies: Custom elements, Shadow DOM, HTML templates.

Custom Elements. Quite simply, these are fully valid HTML elements with custom templates, behaviours and tag names (e.g. <one-dialog>) made with a set of JavaScript APIs.

Shadow DOM. Capable of isolating CSS and JavaScript. HTML templates. User- defined templates in HTML that aren't rendered until called upon.

Web components simplify many complexities in front-end development. Web Components are a collection of building blocks for web pages or web applications in simple terms. With web components, developers have a wider range of opportunity.

Surely, JavaScript frameworks such as React, Vue or Angular follow a similar approach, however these frameworks do so at a cost. Language features need to be compiled, and many JS frameworks rely on a runtime to manage all their abstractions. Web components allow developers to get the benefits without all that heavy weight. web components are generally available in all the major browsers, except for Microsoft Edge and Internet Explorer 11.

1.2 APIs and Microservices:

The microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP

resource API. Nowadays, many companies are using the microservices approach; or rather they are using the micro frontends approach to build their web apps.

The reason microservices are all the rage right now is that they make it so much easier to develop, integrate, and maintain applications. Microservices are especially useful for larger companies since they allow teams to work on separate items without the need for any horribly complicated orchestration between them. This ultimately comes down to the fact that individual functionalities are treated separately, initially allowing you to build applications step-by-step, and later allowing you to work on each element separately (so you can add, improve, or fix without risking breaking the entire application).

API stands for Application Programming Interface, where the keyword is interface. APIs are the doorway, so to speak, that allows developers to interact with an application.
In short, APIs allow internal and external developers to accomplish one of two things: access an application's data or use an application's functionality. Ultimately, this is how the world's electronics, applications, and web pages are linked up to communicate with one another and work together.

Things like using a social account to authenticate on a website, having the weather on your phone, being able to access Google maps from a separate application, or triggering an Internet of Things devices — they all rely on APIs to function. APIs and microservices have changed the web development world profoundly.

2. Python:

Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large- scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0.Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting. Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward compatible and much Python 2 code does not run unmodified on Python 3. Python 2 was discontinued with version 2.7.18 in 2020.

2.1 Python Version Releases:

Version	Release Date	Important Features
Python 0.9.0	February 1991	• Classes with inheritance exception handling • Functions • Modules
Python 1.0	January 1994	• Functional programming tools (lambda, map, filter and reduce). • Support for complex numbers. • Functions with keyword arguments
Python 2.0	October 2000	• List comprehension. • Cycle-detecting garbage collector. • Support for Unicode. Unification of data types and classes
Python 2.7.0	July 2010	
Python 3	December 2008	• Backward incompatible. • print keyword changed to print() function • raw input() function depreciated
Python 3.6	December 2016	

Python 3.6.5	March 2018	• Unified str/Unicode types. • Utilities for automatic conversion of Python 2.x
Python 3.7.0	May 2018	• New C API for thread-local storage • Built-in breakpoint() • Data classes • Context variables
Python 3.8	October 2019	• Assignment Expression • Positional-only parameters • Parallel filesystem cache for compiled bytecode files
Python 3.9 (Current Version)	October 2020	• Dictionary Merge & Update Operators • New removeprefix() and removesuffix() string methods • Built-in Generic Types

II. CONTENT

3.1 Advantages of Python:

According to Stackoverflow, Python is the most preferred language which means that the majority of developers use python.

Python today has multiple implementations including Jython, scripted in Java language for Java Virtual Machine; IronPython written in C# for the Common Language Infrastructure, and PyPy version written in RPython and translated into C. To be noted, Cpython which is written in C and developed by Python Software Foundation is the default and most popular implementation of Python. While these implementations work in the native language they are written in, they are also capable of interacting with other languages through use of modules. Most of these modules work on community development model and are open-source and free.

The diverse application of the Python language is a result of the combination of features which give this language an edge over others. Some of the benefits of programming in Python include:

3.1.1. Presence of Third-Party Modules:

The Python Package Index (PyPI) contains numerous third-party modules that make Python capable of interacting with most of the other languages and platforms.

3.1.2. Extensive Support Libraries:

Python provides a large standard library which includes areas like internet protocols, string operations, web services tools and operating system interfaces. Many high use programming tasks have already been scripted into the standard library which reduces length of code to be written significantly.

3.1.3. Open Source and Community Development:

Python language is developed under an OSI-approved open source license, which makes it free to use and distribute, including for commercial purposes. Further, its development is driven by the community which collaborates for its code through hosting conferences and mailing lists and provides for its numerous modules.

3.1.4. Learning Ease and Support Available:

Python offers excellent readability and uncluttered simple-to-learn syntax which helps beginners to utilize this programming language. The code style guidelines, PEP 8, provide a set of rules to facilitate the formatting of code. Additionally, the wide base of users and active developers has resulted in a rich internet resource bank to encourage development and the continued adoption of the language.

3.1.5. User-friendly Data Structures:

Python has built-in list and dictionary data structures which can be used to construct fast runtime data structures. Further, Python also provides the option of dynamic high-level data typing which reduces the length of support code that is needed.

3.1.6. Productivity and Speed:

Python has clean object-oriented design, provides enhanced process control capabilities, and possesses strong integration and text processing capabilities and its own unit testing framework, all of which contribute to the increase in its speed and productivity. Python is considered a viable option for building complex multi- protocol network applications.

3.2. Real-world Applications of Python:

Python has significantly evolved since its creation in 1991 by Guido Van Rossum. In short, it's an interpreted, dynamic, and high-level programming language that facilitates building a plethora of apps. It's also easy to get into, thanks to its lower learning curve and easy to read syntax. Python is a programming language that does it all, from web applications to videogames, Data Science, Machine Learning, real- time applications to embedded applications, and so much more. In this section,

3.2.1. Web Development:

We hope you're all familiar with what web development is. It's one of the most quintessential applications of Python. What makes Python one of the most popular programming languages for web development is that Python comes with a wide array of frameworks and Content Management Systems(CMS) that exist to simplify a web developer's life. Popular examples of these web development frameworks include Flask, Django, Pyramid, and Bottle, while well-known Content Management Systems include Django CMS, Plone CMS, and Wagtail.

Using Python for web development also offers several other benefits, such as security, easy scalability, and convenience in the development process. More so, Python comes with out-of-the-box support for various web protocols such as HTML, XML, frequently used e-mail protocols, FTP. Python also has one of the largest collections of libraries that not only enhance the functionality of web applications but also make it easier to do so.

3.2.2. Game Development

Just like web development, Python comes equipped with an arsenal of tools and libraries for game development, Python was also used to develop one of the most favourite shooters of the early 2000s, Battlefield 2. Many 2D and 3D game development libraries that make this possible are PyGame, Pycap, Construct, Panda3D, PySoy, PyOpenGL.

Python has also been used to develop several other modern popular titles such as Sims 4, World of Tanks, Civilization IV, and Eve Online, which heavily use Python for a majority of tasks. Mount & Blade, Doki Doki Literature Club, Frets on Fire, and Disney's Toontown Online are among the few other titles that use Python.

3.2.3. Artificial Intelligence and Machine Learning

Python is GitHub's second-most popular language and the most popular language for machine learning.

Artificial Intelligence and Machine Learning are undoubtedly among the hottest topics of this decade. These are the brains behind the smart tech that we so rely on today to help us make optimized decisions. Python, along with a handful of other programming languages, has seen a steep increase in their use for developing AI and ML-powered solutions.

Python's stability and security make it a perfect programming language for handling the intensive computations that keep the Artificial Intelligence and Machine Learning systems running. More so, Python's vast collection of libraries facilitate the development of models and algorithms that run modern AI and ML systems

3.2.4. Desktop GUI

A Graphical User Interface(GUI) is the first thing your user sees and interacts with when he opens your application or website. Having a good GUI goes a long way in increasing your platform's reputation and user count. A user interface usually includes a host of visual elements like icons, buttons, graphics, displayed text, and several other forms of input, like checkbox, text input boxes, and such.

It's the combination of all these elements that makes up a vital part of your application or website's user experience. Input to these visual elements can be from the usual mediums, such as keyboards, mice, and touchscreens. It goes without saying that Python's comprehensible syntax and a modular programming approach are key to creating super-fast and responsive GUI while making the entire development process a breeze. Some

of the many tools available for GUI development using Python are PyQt, Tkinter, Python GTK+, wxWidgets, and Kivy.

3.2.5. Image Processing

Due to the ever-increasing use of Machine Learning, Deep Learning, and Neural Networks, the role of image (pre)processing tools has also skyrocketed. To fulfil this demand, Python offers a host of libraries that simplify much of the initial preparatory tasks of a Data Scientist.

Some of the popular image processing Python libraries include OpenCV, Scikit- Image, and Python Imaging Library(PIL). Other examples of more common image processing applications that use Python are GIMP, Corel PaintShop, Blender.

3.2.6. Text Processing

Text Processing is among the most common uses of Python. For the uninitiated, Text Processing is very closely related to Natural Language Processing. Text Processing allows you to handle enormous volumes of text while giving you the flexibility to structure it as you wish. If you're thinking about sorting lines, extracting text, reformatting paragraphs, and such, you're correct. What else can you do with Text Processing? Well, with Python's text processing capabilities, you can do a lot more than that.

3.2.7. Web Scraping Applications

The internet is home to an enormous amount of information ready to be utilized. What Web Scrapers essentially do is they crawl the websites they're directed towards and store all the collected information from their web pages in one place. From there onwards, this data could be used by researchers, analysts, individuals, organizations for a broad range of tasks.

With Python's simple code, building and using Web Scrapers becomes a lot easier. A few examples of the tools behind the Web Scrapers are PythonRequest, BeautifulSoup, MechanicalSoup, Selenium, and a handful more. Web Scrapers are being commonly used in price trackers, research and analysis, sentiment analysis in social media, Machine Learning projects, and probably every project in the real- world that benefits from a huge repository of data.

3.2.8. Data Science and Data Visualization

Data plays a decisive role in the modern world. Why? It's because it is key to understanding the people and their taste in things around them by gathering and analysing crucial insights about them. This is what the entire domain of Data Science revolves around. Data Science involves identifying the problem, data collection, data processing, data exploration, data analysis, and data visualization. The Python ecosystem offers several libraries that can help you tackle your Data Science problems head-on. We have TensorFlow, PyTorch, Pandas, Scikit-Learn, NumPy, SciPy, and more libraries that specialize in creating and fine-tuning Deep Learning and Machine Learning models, performing intensive data crunching and data manipulation.

Data Visualization comes into play when you need to communicate your findings to the stakeholders and your team. Now, even for visualizations, there is no shortage of libraries in the Python ecosystem. We have Plotly, Matplotlib, Seaborn, Ggplot, Geoplotlib as the most widely used data visualization tools.

3.2.9. Scientific and Numeric Applications

Remember we talked about a few scientific and numerical libraries for Python while covering Artificial Intelligence, Machine Learning, and Data Science? Well, for projects that aren't specifically from the AI, ML, and DS spectrum but still require intensive computations in the form of linear algebra, high-level mathematical functions, and similar, Python is well equipped for them too.

Python's collection of scientific and numerical tools and libraries that include Pandas, IPython, SciPy, Numeric Python, Matplotlib, and many other libraries like them, have helped scientists and researchers conclude countless number-crunching problems and uncover new findings. FreeCAD and Abaqus are some real-world examples of numerical and scientific applications built with Python.

3.2.10. Embedded Applications

By far one of the most fascinating applications of Python is the ability to run on embedded hardware. For those new to this, embedded hardware is a tiny computer that's created to perform limited actions. An embedded application is what drives the hardware, aka the firmware. Popular examples of these applications include MicroPython, PyMite, and EmbeddedPython.

As of today, we have an exhaustive list of embedded devices because they're almost everywhere. For example, Digital Cameras, Smartphones, Raspberry-Pi, and Industrial Robots are just some of the many devices that can be controlled with Python. There are various aspects of Python that makes it perfect for software development of any kind. Python offers a host of features, such as quick execution, high compatibility, strong

community support, and an enormous collection of libraries. Python can also be used as an abstraction layer in a device firmware while C/C++ handles the system level side of things.

4. Web Server Application Development

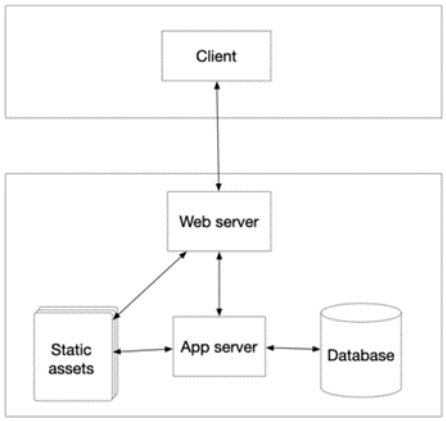

Figure 1 / Web Application Architecture

Web Development involves programming both client and server side. Programming at server side is associated with writing web applications, providing web site, web pages, etc.

First tier is the client which is a web browser which renders the static/dynamic content returned by server, middle tier is the server which uses dynamic web languages and tools such as Python, PHP, Ruby, ASP.NET for web application development which handles the user specific request, and returns the data. Third tier is the database which provides storage means.

Programming at client side is associated with developing tools or interfaces for accessing these web applications, sites, or pages. Server-side programming languages include Python, PHP, Ruby, ASP.NET, Java, CGI (C, Perl). Client-side programming languages include HTML, CSS, and JavaScript.

CGI and WSGI - Web Server Application Development using Python Standard Library

4.1 CGI

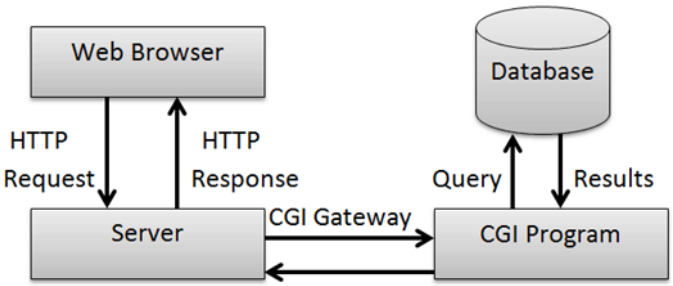

Figure 2 / Common Gateway Interface (CGI)

Dynamic web involves generating the reply on the fly based upon the user input (such as input in a form). Since, server can only take the user request and to return the response; it cannot handle user-specific data and generate the response. So, it uses web applications for dynamically generating response.

Creation of a web server needs a base server and a handler. Base server is responsible for carrying out HTTP communication between client and server, whereas handler is responsible for processing the request and returning the result in the form of a web page or document. The module BaseHTTPServer has the common base server class HTTPServer available in it. Within the same module, basic handler termed BASEHTTPRequestHandler is available, but it offers only the functionality to take the client input. SimpleHTTPRequestHandler found in SimpleHTTPServer module offers additional GET functionality. GET method is used for fetching the requisite document from the database. Parameters of GET method are part of URL. Similar to GET method is the POST method that is used for updating data. POST parameters are included in the body of the document. CGIHTTPRequestHandler found in CGIHTTPServer module takes SimpleHTTPRequestHandler and provides additional functionality of POST method.

When a web server receives a client request via GET or POST method, it invokes the web application, and returns the dynamically generated HTML page to the client. This process takes place through CGI. CGI is the Common Gateway Interface, which act as an interface between server and the application program. Figure 2 presents overview of working of CGI. CGI application depicted in Figure may use database for data storage and retrieval purpose.

CGI provides support for use of cookies using which server saves data on client side. It also supports processing of multiple inputs provided for a field, for example using checkboxes. Web programming using CGI is not preferred since for each client request, server forks a process of CGI python program. This will lead to wastage of time since python interpreter will be initiated for each request. For a large number of requests, this may bring the server to halt. The problem associated with CGI can be overcome by using two modes– embedded or daemon.

4.2 WSGI

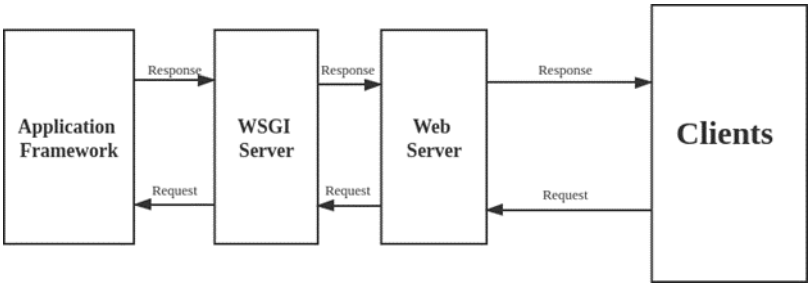

Figure 3 / Web Server Gateway Interface

Web Server Gateway Interface (WSGI) is a standard that facilitate interaction between the server and the application by acting as an interface. It is currently the most preferred interface for python web programming. Figure 3 presents overview of working of WSGI.

A WSGI application is an invokable application that takes two parameters. First parameter is a dictionary type that comprises of environment variables such as HTTP_HOST, SERVER_PROTOCOL and their values. Second parameter is a callable function that must be executed for initiating response to the client. The response prepared contains response code indicating status of response which comprises of status code and reason phrase (200 OK, 302 Found, 403 Forbidden, 500 Server Error). HTTP response may also contain HTTP headers such as Date (date and time of message initiation), Server (server software and its version), Content-Type (type of content in response body) application is sent back to the client. It may be noticed that WSGI acts as an interface between the application and the server. The program at server side that invokes WSGI application can either be written by the developer or he may use the python provided reference server available in library wsgiref.simple_server.WSGIServer. Thus, using WSGI one can implement both sides of the interface, server as well as application.

III. PYTHON WEB DEVELOPMENT FRAMEWORKS

Figure 4 / Python web frameworks

5.1 Django:

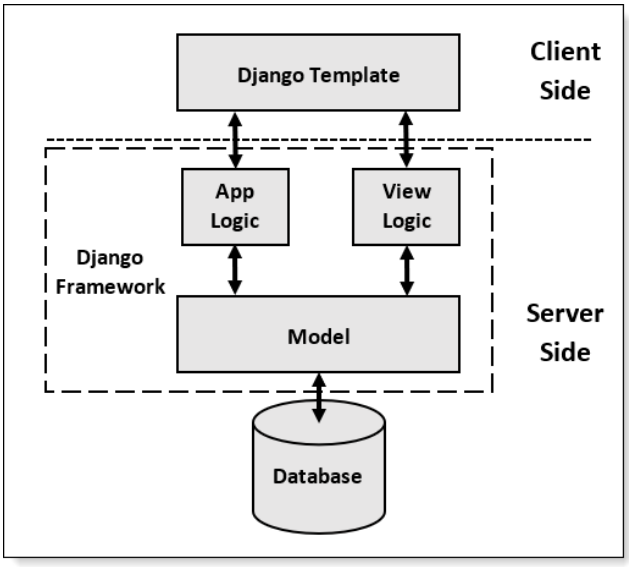

Figure 5 / Django's MVT architecture

Being a full stack framework, Django supports quick web application development requiring lesser core to be done. It is popularly known as "the web framework for perfectionists with deadlines" . Django was first released back in 2005 when front- ends were relatively straightforward. There were no dedicated JavaScript front-end frameworks like React, Angular, Vue, Ember, and the rest. It provides ease in creating web applications with fewer lines of code and is scalable. It includes a built-in server that can be used for developing and testing the applications. The framework comes with comprehensive, well-written documentation. Features of this framework include templates, support for relational database model (Databases include MySQL, SQLite, Oracle, and PostgreSQL), comprehensive security, etc. It is well-suited for database driven applications. The framework is based on the principle of reusability of code and no redundancy of information. Examples of applications built using this framework include Pinterest, Instagram, Mozilla, Spotify, The Onion, Diqus.

Why Django?

• Django is a rapid web development framework that can be used to develop fully fleshed web applications in a short period of time.

• It's very easy to switch database in Django framework.

• It has built-in admin interface which makes easy to work with it.

• Django is fully functional framework that requires nothing else.

• It has thousands of additional packages available.

• It is very scalable.

MVT Structure

MVT Structure has the following three parts (Figure 5) –

Model: Model is going to act as the interface of your data. It is responsible for maintaining data. It is the logical data structure behind the entire application and is represented by a database (generally relational databases such as MySQL, Postgres).

View: The View is the user interface — what you see in your browser when you render a website. It is represented by HTML/CSS/JavaScript and Jinja files.

Template: Django's template language is designed to strike a balance between power and ease. It's designed to feel comfortable and easy-to-learn to those used to working with HTML, like designers and front-end developers. But it is also flexible and highly extensible, allowing developers to augment the template language as needed.

Django Project Structure –
The fundamental unit of a Django web application is a Django project. A Django project comprises one or more Django apps (Figure 6). A Django Project when initialised contains basic files by default such as manage.py, view.py, etc. A simple project structure is enough to create a single page application.

Here are the major files and their explanation -

\myApp_project\myApp_root\
\myApp
  init.py
asgi.py # Django 3 only settings.py
urls.py wsgi.py

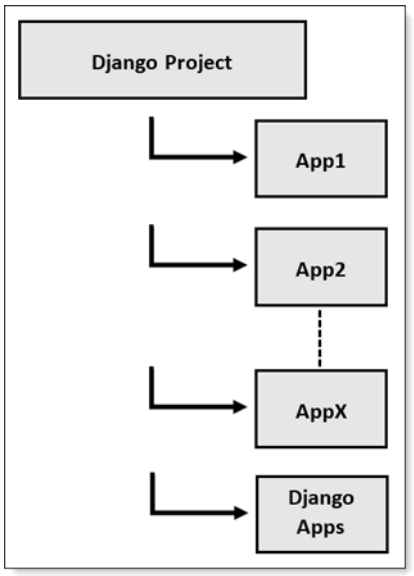

Figure 6 / Django's project structure

manage.py-This file is used to interact with your project via the command line(start the server, sync the database… etc.
$ python manage.py help

folder (myApp) – This folder contains all the packages of your project. Initially it contains four files –
init.py – It is python package.

settings.py – As the name indicates it contains all the website settings. In this file we register any applications we create, the location of our static files, database configuration details, etc.

urls.py – In this file we store all links of the project and functions to call.

wsgi.py – This file is used in deploying the project in WSGI. It is used to help your Django application communicate with the web server

Now let's look inside the \myApp_root folder to see what Django has created -

• The **migrations** folder is where Django stores migrations, or changes to your database. There's nothing in here you need to worry about right now.

• **init .py** - tells Python that your events app is a package.

• **admin.py** is where you register your app's models with the Django admin application.

• **apps.py** - is a configuration file common to all Django apps

• **models.py** - is the module containing the models for the app.

• **tests.py** - contains test procedures that run when testing of the app starts.

• **views.py** - is the module containing the views for the app.

URLs and views

A clean, elegant URL scheme is an important detail in a high-quality Web application. Django encourages beautiful URL design and doesn't put any cruft in URLs, like .php or.asp.

To design URLs for an application, you create a Python module called a URLconf. Like a table of contents for your app, it contains a simple mapping between URL patterns and your views.

```
from django.urls import path

from . import viewsurlpatterns = [
    path('bands/', views.band_listing, name='band-list'), path('bands/<int:band_id>/',
    views.band_detail, name='band-detail'),path('bands/search/', views.band_search,
    name='band-search'),
]
```

```
  """A view of all bands."""

from django.shortcuts import renderdef

band_listing(request):
    bands = models.Band.objects.all()
    return render(request, 'bands/band_listing.html', {'bands': bands})
```

Authentication

Django comes with a full-featured and secure authentication system. It handles user accounts, groups, permissions, and cookie-based user sessions. This lets you easily build sites that allow users to create accounts and safely log in/out.

```
  from django.contrib.auth.decorators import login_requiredfrom
  django.shortcuts import render

  @login_required
  def my_protected_view(request):
      """A view that can only be accessed by logged-in users"""
      return render(request, 'protected.html', {'current_user': request.user})
```

Security

Django provides multiple protections against:

• Clickjacking
• Cross-site scripting
• Cross Site Request Forgery (CSRF)
• SQL injection
• Remote code execution

5.2. TurboGears

TurboGears framework combines SQLAlchemy, Pylons, Genshi, Repoze, and Tosca Widgets. This framework includes 1.x and 2.x series where the 2.x series comprises of all the above specified components. However, 1.x series consists of SQLAlchemy, Genshi, CherryPy, and MonchKit. It supports both client and server-side web programming. It offers designer friendly template system. It has found its use mainly for offering solutions for industrial problems with high complexity. The framework supports several databases among

which SQLAlchemy is considered to be most powerful database management system. This framework is used by SourceForge, Fedora Community, TavolaClandestina, Glossom, ShowMeDo, etc.

5.3. web2py

The framework supports fast development of scalable, secure, and portable web applications. It does not require any installation and can be run via USB drive. This framework has no dependencies. The web2py application itself acts as an Integrated Development Environment offering all the capabilities such as creating application, debugging, and testing. It comes with a ticketing system that helps in tracking the source of error occurred by assigning a ticket to the user. It supports several databases such as MySQL, PostgreSQL, SQLite, Oracle.
Examples of applications built using this framework include Movuca, Instant Press, and LinkFindr.

5.4. Flask

Flask framework is based on Werkzeug, and Jinja 2, and good intensions [28]. This framework has no dependencies apart from Python Standard Library. Flask does not include components that need third party support such as validating form or providing a means of communication between the application and the database. However, such features may be added using extensions. Services offered by this framework include built-in HTTP server, support for unit testing, and RESTful web service [29]. Applications built using this framework are minitwit, flaskr, flask.pocoo.org etc.

5.5. Pyramid

Pyramid is a fast, reliable, simple, and mini web development framework [30]. This framework is compatible with Python 3. Being a fully documented framework, developers find an easy start for developing web applications. It supports several databases such as SQLalchemy, Zope Object Database, and CouchDB. Websites built using this framework include newcars.com, SurveyMonkey, PwnedList.

Specifications of popular web frameworks are provided in Table -

Framework	Latest Stable Version	MVC	ORM	Python Version
Django	1.6.5 (14 May 2014)	Yes	Yes	2.7 & above
TurboGears	2.3.2 2014)	Yes	Yes	2.4 & above
web2py	2.9.5 (16 March 2014)	Yes	Yes	2.4 & above
Flask	0.10.1 (14 June 2013)	Yes	No	2.5 & above
Pyramid	1.4.6 (31 May 2014)	Yes	Yes	2.6 & above

Almost all python web frameworks specified above support features such as built-in server, internationalization (helps in adding languages and their translation), authentication and authorization of end-user, caching, debugging, error logging (help in keeping track of errors occurred along with their description, and time), routing the request to another URL, and managing sessions.

IV. CONCLUSIONS

Inspite of availability of so many web development languages, Python is evolving as the popular language among developers. Web development in Python can be done using CGI or WSGI that uses python standard libraries. However, such development requires building web applications from scratch. So, Python provides web development frameworks such as Django, Pyramid, web2py that helps in web application development. This helps the Python as a Tool for Web Server Application Development developer in doing the minimal requisite core needed for building the applications.

REFERENCES

[1]. Author name(s), paper name, Journal/ Magazine/ Periodical name, issue no., page nos.
[2]. Grove, Ralph F. "Web Based Application Development." Jones & Bartlett Publishers, 2009
[3]. Forcier, jeff, Paul Bissex, and Wesley “Python web development with Django”, Addison-Wesley Professional, 2008
[4]. https://f5-studio.com/articles/introduction-to-modern-web- development/
[5]. https://en.wikipedia.org/wiki/Python_(programming_language)
[6]. https://www.python.org/doc/versions/
[7]. https://towardsdatascience.com/top-16-python-applications-in-real-world- a0404111ac23
[8]. "Django" https://www.djangoproject.com/
[9]. "TurboGears" http://turbogears.org/
[10]."web2py" http://web2py.com/
[11]."Flask" http://flask.pocoo.org/

