Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of Information Technologies

Bachelor Thesis

Design and Implementation of web application by using Python Django Framework

Author: Abdulvokhid Azimov
Supervisor: Ing. Miloš Ulman, Ph.D.

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

BACHELOR THESIS ASSIGNMENT

Abdulvokhid Azimov

Systems Engineering and Informatics
Informatics

Thesis title

Design and implementation of web application by using Python Django Framework

---

Objectives of thesis

The main goal is to demonstrate the performance of the Python Django Framework in comparison with JavaScript frameworks.
The partial goals of the thesis are:
- To make a comprehensive overview of existing web development frameworks.
- To design and implement a web application according to further specification.
- To conduct comparison of Python Django Framework with JavaScript frameworks.

Methodology

The methodology of this thesis will be based on literature review, analysis of technical and scientific sources related to web application development. The main focus will be on using Python Django Framework, to demonstrate Django capability to design and implement a web application. In practical part, the frameworks will be compared. The possible changes and improvements will be proposed based on the evaluation.

The proposed extent of the thesis

30 – 40 pages

Keywords

Python, frameworks, web applications, programming languages, website, web design.

Recommended information sources

DAUZON, Samuel; BENDORAITIS, Aidas; RAVINDRAN, Arun. Django: Web Development with Python. Packt Publishing Ltd, 2016.

DEL PILAR SALAS-ZÁRATE, María, et al. Analyzing best practices on Web development frameworks: The lift approach. Science of Computer Programming, 2015, 102: 1-19.

FORCIER, Jeff; BISSEX, Paul; CHUN, Wesley J. Python web development with Django. Addison-Wesley Professional, 2008.

HAYWARD, Jonathan. Django javascript integration: Ajax and jquery. Packt Publishing Ltd, 2011.

HOLOVATY, Adrian; KAPLAN-MOSS, Jacob. The definitive guide to Django: Web development done right. Apress, 2009.

MITCHELL, Ryan. Web scraping with Python: Collecting more data from the modern web. " O'Reilly Media, Inc.", 2018.

PLEKHANOVA, Julia. Evaluating web development frameworks: Django, Ruby on Rails and CakePHP. Institute for Business and Information Technology, 2009.

VAINIKKA, Joel. Full-stack web development using Django REST framework and React. 2018.

VAN ROSSUM, Guido, et al. Python Programming Language. In: USENIX annual technical conference. 2007. p. 36.

Expected date of thesis defence

2020/21 SS – FEM

The Bachelor Thesis Supervisor

Ing. Miloš Ulman, Ph.D.

Supervising department

Department of Information Technologies

Electronic approval: 20. 7. 2020	Electronic approval: 19. 10. 2020
Ing. Jiří Vaněk, Ph.D.	**Ing. Martin Pelikán, Ph.D.**
Head of department	Dean

Prague on 09. 03. 2021

Declaration

I declare that I have worked on my bachelor thesis titled "Design and Implementation of web application by using Python Django Framework" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the bachelor thesis, I declare that the thesis does not break copyrights of any their person.

In Prague on 15th of March 2021 ___________________________

Abdulvokhid Azimov

Acknowledgement

I would like to thank my supervisor Ing. Miloš Ulman, Ph.D. for Your time, instructions and advice that was very helpful and essential during writing of this thesis.

Design and Implementation of web application by using Python Django Framework

Abstract

The aim of thesis is to show capability of Python Django framework comparing with JavaScript frameworks in web application area by creating web applications. Moreover, subgoal of thesis is to be familiar of Python Django framework and JavaScript frameworks.

In theatrical part history of programming languages and modern frameworks were reviewed and based on literature review two web application were created by using Python Django framework and JavaScript NodeJS frameworks. NodeJS were selected due to the fact that author found that both frameworks usually used for back-end rather than front-end that is why, author decided to compare Django and NodeJS as back-end frameworks.

In practical part in order to compare those two web frameworks author created for NodeJS real time web chat application and for Django e-commerce website. Both websites were compared based on installation, security, database, packages, and request.

Lastly, based on research author found that both frameworks have their own unique strength and weaknesses. However, developers should pick frameworks based on their taste and projects.

Keywords: Python, frameworks, web applications, programming languages, website, web design.

Návrh a implementace webové aplikace pomocí Python Django Rámce

Abstrakt

Cílem diplomové práce je ukázat schopnosti rámce Python Django ve srovnání s rámci JavaScriptu v oblasti webových aplikací vytvořením webových aplikací. Dále je cílem práce seznámit se s frameworkem Python Django a frameworkem JavaScript.

V divadelní části byla zkontrolována historie programovacích jazyků a moderních rámců a na základě přehledu literatury byly vytvořeny dvě webové aplikace pomocí rámců Python Django a JavaScript NodeJS. NodeJS byly vybrány kvůli skutečnosti, že autor zjistil, že oba rámce se obvykle používají spíše pro backend než frontend, proto se autor rozhodl porovnat Django a NodeJS jako backendové rámce.

V praktické části za účelem porovnání těchto dvou webových frameworků, které autor vytvořil pro webovou chatovací aplikaci NodeJS v reálném čase a pro web elektronického obchodu Django. Oba weby byly porovnány na základě instalace, zabezpečení, databáze, balíčků a požadavků.

A konečně, na základě výzkumu autor zjistil, že oba rámce mají své vlastní jedinečné silné a slabé stránky. Vývojáři by si však měli vybrat rámce podle svého vkusu a projektů.

Klíčová slova: Python, rámce, webové aplikace, programovací jazyky, web, webdesign.

Table of content

List of figures

List of tables

1 Introduction

Programming has continuously been rising to parts art and science. It's simple to see the that, science tries to educate computers how to do things, but once that's out of the way, we frequently attempt to grasp the creative side. We spend our to begin with few a long time learning to form code utilitarian and the rest of our careers attempting to make it lovely.

Django begun its life in much the same way, serving the day-to-day needs of a nearby news organisation. Within a long time since its to begin with open release, Django itself has developed more rich and has made a difference its adopters to type in more exquisite code for their own applications.

In any case, choosing the suitable Web framework for Web development, which best fits the developers' prerequisites, is not a simple assignment, since there are numerous systems based on diverse languages. Besides, selecting an inappropriate framework can lead to 1) sitting around idly studying the subtle elements of another language, 2) disappointment to meet the specified time since developers are not utilized to the framework, and 3) spending time taking remedial activities to choose a diverse framework. In order to maintain a strategic distance from these problems, it is exceedingly vital to know and distinguish the finest hones for Web development. A best practice may be a prepare, method, innovative utilize of technology, or a set of assets with a demonstrated record of success in giving noteworthy improvements in cost, plan, quality, execution, security, environment, or other quantifiable factors that affect on an organization. Best practices on Web frameworks infer decreasing improvement time and exertion whereas saving cash, expanding the qual

At last, nowadays websites are complex applications that perform exchanges, display real-time information, and give intuitively client experiences. Web based software is getting to be as capable and as important as desktop software. Creating web applications that give progressed usefulness is presently a complex assignment that includes different designers, advancing toolsets, and numerous choices. Web frameworks give a brilliant cruel between building an application from scratch and utilizing an out-of-the-box substance administration framework.

2 Objectives and Methodology

2.1 Objectives

The main goal is to demonstrate the performance of the Python Django Framework in comparison with JavaScript frameworks.
The partial goals of the thesis are:
-To make a comprehensive overview of existing web development frameworks.
-To design and implement a web application according to further specification.
-To conduct comparison of Python Django Framework with JavaScript frameworks.

2.2 Methodology

The methodology of this thesis will be based on literature review, analysis of technical and scientific sources related to web application development. The main focus will be on using Python Django Framework, to demonstrate Django capability to design and implement a web application. In practical part web application will be compared. The possible changes and improvement will be proposed based on the evaluation.

3 Literature Review

3.1 History of OOP

These days there's no correct definition of the object-oriented programming (OOP) or the object- oriented programming language. In books, different authors grant a diverse clarification of these terms. Able to characterize the object-oriented programming language as a programming language, fundamental components are objects that have their own attributes and methods, and shaping a progressively organized classes of objects.

According to this definition:

- Object can be described as model(abstraction) of a real substance in a programming system.
- Class can be described as an abstract of attributes and methods for a group of similar objects.
- Attribute identifies as a parameter in a class which characterizes the object.
- Method defines as behaviour of class instances.
- Generally, the object-oriented divides in order to develop programs four main mechanisms: abstraction, encapsulation, polymorphism and inheritance.
- Abstraction is the method of distinguishing of the essential characteristics of an protest that recognize it from all other sorts of objects and an in this way giving freshly characterized conceptual boundaries, from the prospective of the observer.
- Encapsulation is process of classifying the elements which is constitute, structure and behaviour.
- Polymorphism is ability to assign a different or usage to something in different contexts and the property of an object respond to a query according to its type.
- Inheritance is a mechanism to declare new data types on the basis of existing where attributes and methods of the base types become the members of the subtype. [1]

John Von Neumann (1945) while working for Advance Study Institute, he came up with two theory that affected developing computer programming languages. The first idea was about shared program technique. The meaning of the idea was computer hardware supposed to be simple and not need to be manually-wired for each program. Instead of manually wired for each program the instruction supposed to control hardware and allow to be re-programmed faster. The second theory was about conditional control transfer.

In 1957 by IBM designed FORTRAN language for scientific computing. The components were exceptionally basic, and given the software engineer with low-level get to to the computers innards. Nowadays, this language would be considered prohibitive because it as it were included If, DO, and GOTO statements.

John McCarthy(1958) was made Lisp programming language, it was planned for Artificial Intelligence (AI) investigate. A LISP list is indicated by a grouping of things encased by enclosures. LISP programs themselves are composed as a set of records, so that LISP has the special ability to adjust itself, and consequently develop on its own. The LISP sentence structure was known as "Cambridge Polish," because it was exceptionally diverse from standard Boolean logic (Wexelblat, 177):

Dennis Ritchie(1972) while working at Bell Labs in New Jersey. He created C language. The transition in utilization from the primary major languages to the major languages of nowadays occurred with the transition between Pascal and C. Its coordinate precursors are B and BCPL, but its similarities close to Pascal. All features of Pascal, counting the unused ones such as the CASE statement are available in C. However, C uses pointers broadly and was built to be quick and capable at the cost of being difficult to read. But since it settled most of the mistakes Pascal had, it won over former-Pascal users very quickly. Ritchie created C for the modern Unix system being made at the same time. Since of this, C and Unix go hand in hand. Unix gives C such progressed features as dynamic factors, multitasking, interrupt handling, forking, and solid, low-level, input-output. Since of this, C utilized to program operating systems such as Unix, Windows, the MacOS, and Linux. [2]

Bjarne Stroustrup (1979) while working on his Ph.D. thesis, he begun working on "C with Classes", which as the title suggests was implied to be a superset of the C language. His goal was to include object-oriented programming into the C language, and it is language well-respected for its compactness without sacrificing speed or low-level functionality. The product which was developed by him were included classes, basic inheritance, inclining, default function arguments, and strong type checking all the features of the C language. In 1983, the name of the language was changed from C with Classes to C++ and C++ provides a set of integrated classes and opportunity to declare new types by developers. Classes inherits one or several classes, by providing single and multiply inheritance, correspondingly. In C ++ the possibility of describing parameterized classes and functions and the ability to describe the special case dealing with in the program were introduced. [3]

In summary, Programming languages have been under improvement for a long time and will stay so for many years to come. They got begin with a list of steps to wire a computer to perform a task. These steps eventually found their way into software and started to obtain more current and way better features. The primary major languages were characterized by the straightforward truth that they were aiming for one reason while the languages of nowadays are separated by the way they are modified in, as they can be utilized for nearly any purpose. And maybe the languages of tomorrow will be more natural with the innovation of quantum and biological computers.

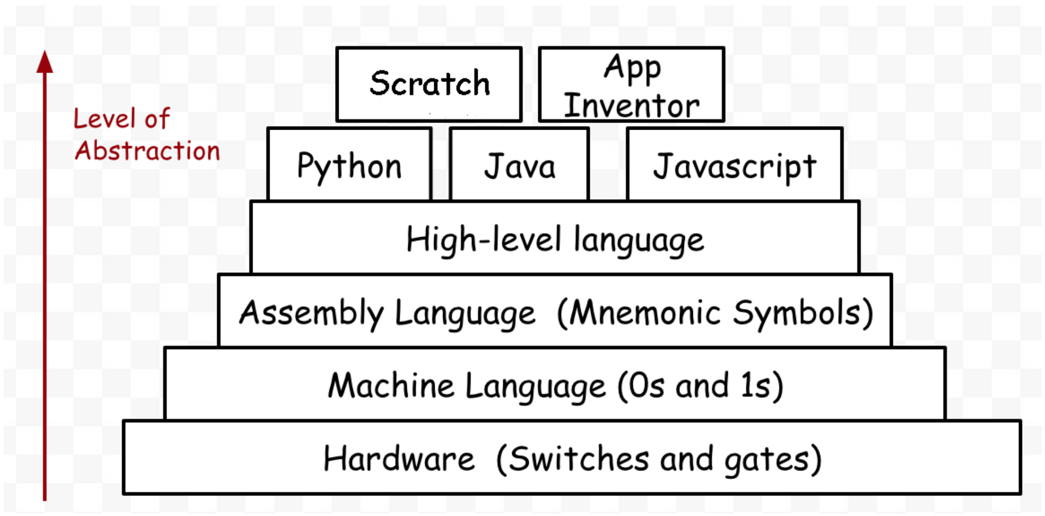

Level of Abstraction
Scratch
App Inventor
Python
Java
Javascript
High-level language
Assembly Language (Mnemonic Symbols)
Machine Language (0s and 1s)
Hardware (Switches and gates)

Figure 1. High Level and Low Level Language. Source: [4]

3.1.1 Python Programming Language

Python is an open-source common purpose programming language, and optimized for quality, execution, transportability and integration. It is utilized by hundreds of thousands of developers around the world in ranges such as creating Web scripts, system programming, designing user interfaces, customization of program items for the client, etc. Among other things, Python supports object-oriented programming (OOP); has a syntax that is very basic, easy read and taken after; integrates with components composed in C language contains a huge collection of pre-programmed interfaces and utilities. In spite of its common purpose, and python is frequently called the scripting dialect, since it is simple to utilize other program components and oversee them. Probably, the greatest advantage of Python is simplicity. However, software development gets to be quicker and more pleasant.[5]

Van der Walt (2011), Numpy is used for data model parameters. Input data is presented as numpy arrays, which integrates seamlessly with other scientific Python libraries. Numpy is view based memory model, which limits copies, even when binding with compiled code.

Michel (2011), Scikit-learn uncovered a wide assortment of machine learning algorithms, which is supervised and unsupervised, using a consistent and task-oriented interface, in this way empowering simple comparison of strategies for a given application. Since it depends on the logical Python ecosystem, it effectively integrates into applications outside the conventional run of statistical information analysis. Imperatively, the algorithms, executed in a high-level language, can be utilized as building blocks for approaches particular to a utilize case, for illustration, in medical imaging. [6]

K.Jarrod Millman was a researcher at the University of California in Berkeley's Brain Imaging Center, while working there he helped found the Neuroimaging in Python (NIPY) project. He is on the SciPy steering committee and a supporter to both the NumPy and SciPy ventures. K.Jarrod research was about reproducible research, functional brain imaging, informatics, configuration management, and computer security.

Michael Aivazis is foremost researcher at the Caltech Center for Progressed Computational Research, He focuses on design and implementation of Pyre, a comprehensive, Python-based component system for high-performance logical computing. Additionally, he is coprincipal investigator at the Caltech Predictive Science Scholarly Alliance Program Center, in that center he leads the exertion to develop and integrate large-scale enormously parallel multiphysics simulation codes in a large-scale, worldwide optimization framework. MOreover, he leads to create the next generation of solvers for the center, focusing on scalable parallel algorithms for meshing, contact, fracture, and fragmentation. His research interface include software engineering and techniques for object-oriented programming. [7]

De Hoon (2004),supervised statistical learning for Biopython modules, such as Bayesian methods and Markov models, as well as unsu prevised learning, such as clustering.

Pritchard (2006), Module Bio.Motif provides support for sequence motif analysis (searching, comparing and de novo learning). However, biopython is graphical output which capabilities significantly expanded by the incorporation of GenomeDiagram. [8]

3.1.2 JavaScript (Angular JS, Node JS)

JavaScript has been created to let HTML engineers to compose scripts straightforwardly in their documents. In December 1995 Netscape and Sun Microsystems together announced that the scripting language from that point would be known as JavaScript. [9] However, JavaScript is a scripting language planned for improving web pages. JavaScript programs are deployed in HTML documents and interpreted by all major web browsers. They give valuable client-side computation facilities and access to the client system, making web pages more engaging, interactive, and responsive.

AngularJS is client-side framework developed by Google. It is written in JavaScript as well as, with a reduced jQuery library. The theory behind AngularJS is to provide a framework that makes easy to implement well designed and structured webpages and applications, using an MVC framework. AngularJS provides functionality to handle user input in the browser and manipulates data on the client side, and control how elements are displayed in the browser view. Here are some of benefits AngularJS:
Data binding in AngularJS: it has a clean method for binding data to HTML elements, and using its powerful scope mechanism.
Extensibility in AngularJS: it is architecture allows you to easily extend almost every aspect of the language to provide your own customer implementation.
Clean in AngularJS: it forces you to write clean logical code.
Reusable code in AngularJS: The combination of extensibility and clean code makes it very easy to write reusable codes. In fact, the language often forces you to do when creating custom services.
Support in AngularJS: Google is investing his resources to this project, which gives it in advantage over similar initiatives that have failed.
Compatibility in AngularJS: it's based on JavaScript and has close relationship with jQuery. This makes it integrating AngularJS into your environment easier and reuse pieces of your existing code within the structure of the AngularJS framework.[10]

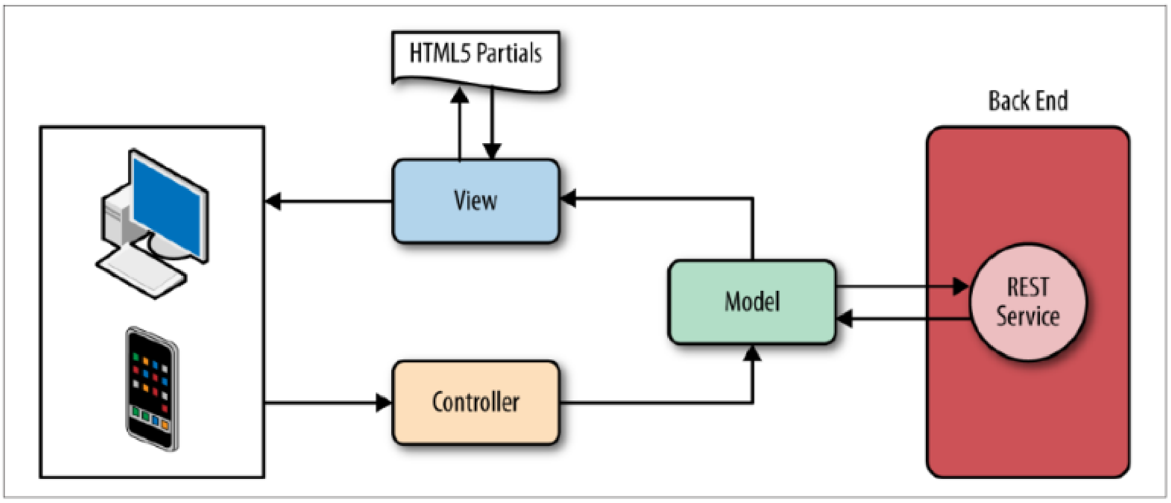


Figure 2. Shows diagram of an AngularJS application and all related MVC components. Soruce: [10]

NodeJS is one later system to implement the event model through the entire stack. Created in 2009 by Ryan Dahl, Node.js (or just Node) is a single-threaded server-side JavaScript environment implemented in C and C++. Nodes architecture makes it simple to utilize as an expressive, functional language for server-side programming and its well known among engineers. Node utilizes the JavaScript V8 engine, developed by Google, a quick and powerful implementation of JavaScript that helps Node accomplish top execution. [12]

3.2 Frameworks

Nowadays websites are complex applications that perform exchanges, display real-time data, and provide interactive user experiences. Web based computer programs is getting to be as effective and as crucial as desktop software. Creating web applications that provide progressed functionality is presently a complex task that involves different developers, advancing toolsets, and numerous options. Web systems give a brilliant cruel between building an application from scratch and utilizing an out-of-the-box content management system (CMS). This report focuses on three driving open-source web development frameworks: Django, Ruby on Rails and CakePHP composed in three different languages – Python, Ruby and PHP individually. All three frameworks have comparative structures and claim to have similar characteristics, such as significantly upgraded efficiency and code re-use.
All three frameworks share the fallowing features and attributes:

- Streamline automates some of the parts;
- Add structure to the code and make code clearer;
- Re-use components, in order to speed up the development process;
- Support concurrent creation, and update content development

The three frameworks share the model-view-controller (MVC) architecture. In spite of the fact that Django calls its architecture model-view-template (MVT), it is exceptionally comparable to MVC. In MVC, an application's information model, user interface, and control logic are isolated into three components. The model manages the information of the application and the trade rules; the view is responsible for showing information to the client through an interface; and the controller translates client inputs and communicates with the model to create the fitting changes.

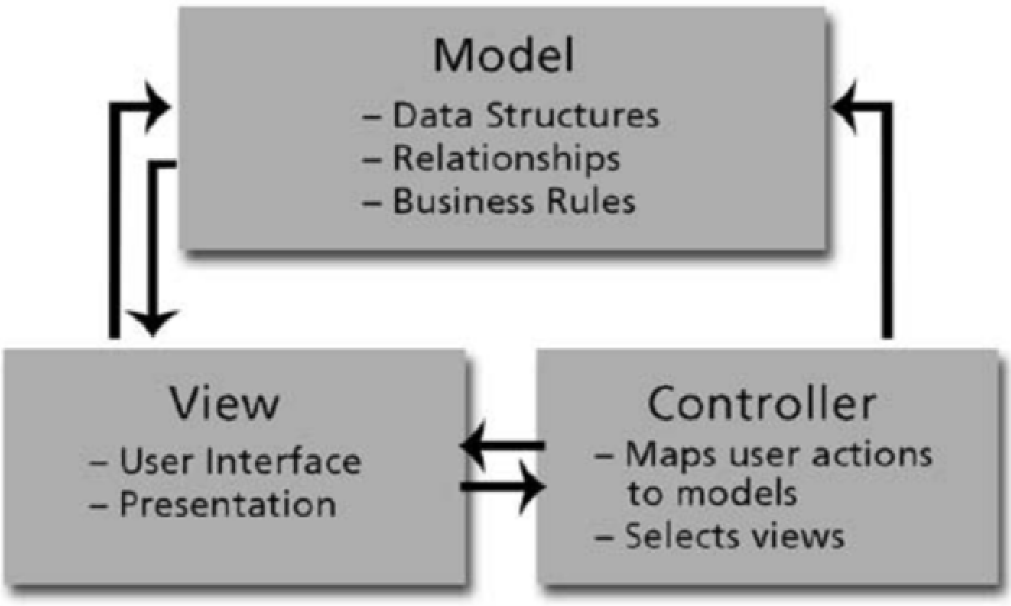


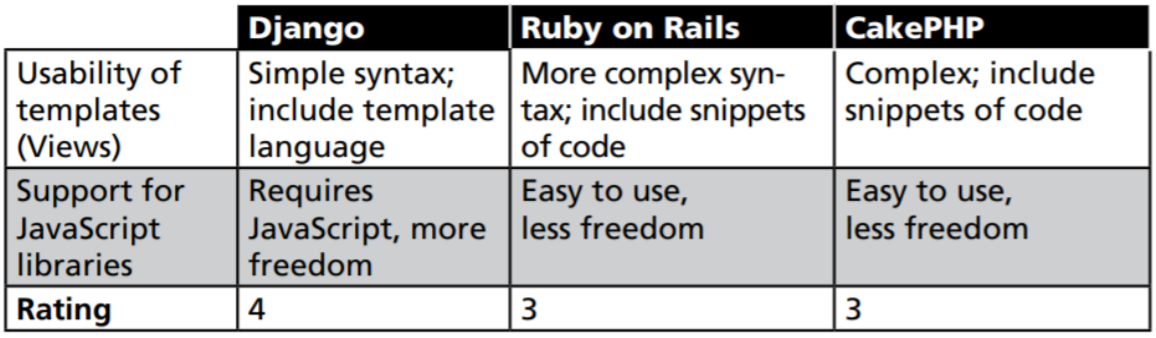
Figure 3. Exhibit 1 illustrates MVC and the interaction between the components. Soruce: [13]

Django. Django is Python library for web development which we can say as “web framework that encourages quick development and clean, pragmatic design”. Created in a fast-paced onlinenews environment in Lawrence, Kansas, it claims to be “the web framework for perfectionists with deadlines.” Django was created in 2003, but wasn’t released until July 2005, official release 1.0 was in September 2008. “Django focuses on automating as much as possible and tries not to repeat (DRY) like don’t repeat yourself principle”. Django is based on Python, which is a very well-known programming language. Regularly compared to Perl, Ruby, and Java, it is a dynamic object-oriented language. Python offers solid support for integration with other language tools and comes with broad standard libraries. Huge firms such as NASA, Google, YouTube, Yahoo!, and Apple utilize Python for their applications.

Ruby on Rails. Rails was created by 37Signals Inc, in Chicago, Illinois for project management application called Basecamp. Rails was officially announced as open source in July 2004, version 1.0 came out in December 2005. Comparable to Django, “Rails also uses as full-stack framework for creating database-backed web applications” that's composed in Ruby and follows the standards of agile web development, increases efficiency and speeds up development. One of the popular Rails websites is yellowpages.com, which views per month approximately 100- 170 million. Ruby is a dynamic, open-source programming language which focus on simplicity and efficiency. It has rich sentence structure that's natural to read and simple to write. Ruby is utilized by NASA Langley Research Center, Motorola, Lucent and other major firms. Ruby is popular because of his Rails framework.

CakePHP. CakePHP is framework for PHP that gives an extensible architecture for creating, keeping up, and deploying applications. CakePHP was created in 2005, when Ruby on Rails was picking up popularity. The objective of CakePHP is to empower clients to quickly develop strong and well-structured web applications. CakePHP is based on PHP, a general-purpose scripting language that's particularly suited for web programming. There's broad support for this language and hundreds of web application examples. The language is simple to memorize and get it, and numerous software engineers type in in PHP. The most standards behind PHP are strength and straightforwardness.

Support of JavaScript libraries. Those three frameworks have diverse approaches towards JavaScript support. In spite of the fact that engineers can utilize nearly any JavaScript library, CakePHP and Rails have standardized on particular default toolkits - Model and Scrip.aculo.us - and include a number of aide capacities. On the other hand, Django clears out to the developer the choice of a JavaScript library. “Django includes a JSON module, taking off JavaScript code, and leaves the choice of a JavaScript library to the developer”. Developers who have a broad knowledge of JavaScript, the Django approach appears to be superior. However, engineers with less JavaScript experience will likely incline toward the Rails and CakePHP approach. [13]



	Django	Ruby on Rails	CakePHP
Usability of templates (Views)	Simple syntax; include template language	More complex syn-tax; include snippets of code	Complex; include snippets of code
Support for JavaScript libraries	Requires JavaScript, more freedom	Easy to use, less freedom	Easy to use, less freedom
Rating	4	3	3

Figure 4. User Interface Development. Soruce: [13]

The main advantages of Django are automatically generated administrative interface and simple templates even can be used by non-programming web designers. Advantages of Ruby on Rails is simple implementation of JavaScript like evolving schema, and large community.

3.3 Python programming language

Python can be explained as high-level and general-purpose programming language. Philosophy of Python is emphasize code readability with its outstanding utilize of significant whitespace. Language constructs and object-oriented approach tries to help software engineers write clear, logical code for little and large-scale projects. Additionally, python is powerfully written and garbage-collected, it supports different programming paradigms, counting structured (particularly, procedural), object-oriented, and functional programming. Lastly, python can be described as a "batteries included" language because of comprehensive standard library.

Python was conceived within the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) within the Netherland and implementation started in December 1989.

First python release was on 16th of October 2000, the release version was Python 2.0 with numerous major modern features, counting a cycle-detecting garbage collector and support for Unicode. Python 3.0 was released on 3 December 2008. It had a major modification of the language that's not totally backward-compatible. Numerous of features were backported to Python 2.6.x and 2.7.x adaptation series. Releases of Python 3 incorporate the 2to3 utility, which automates (at least somewhat) the interpretation of Python 2 code to Python 3. Python 2.7's end-of-life date was at first set at 2015 at that point delayed till 2020 out of concern that a huge body of existing code could not easily be forward-ported to Python 3. No more security patches or other enhancements will be released for it. With Python 2's end-of-life, as it were Python 3.6.x and later are supported.

Most of the developers admirers of Python, those who considered knowledgeable of language are referred to as Pythonistas.

Syntax and Semantics: Python is implied to be an effectively clear language. Its designing is visually uncluttered, and it regularly uses English keywords where other languages utilize punctuation. Unlike numerous other languages, it does not utilize curly brackets to delimit blocks, and semicolons after explanations are optional. It has less syntactic exceptions and uncommon cases than C or Pascal.

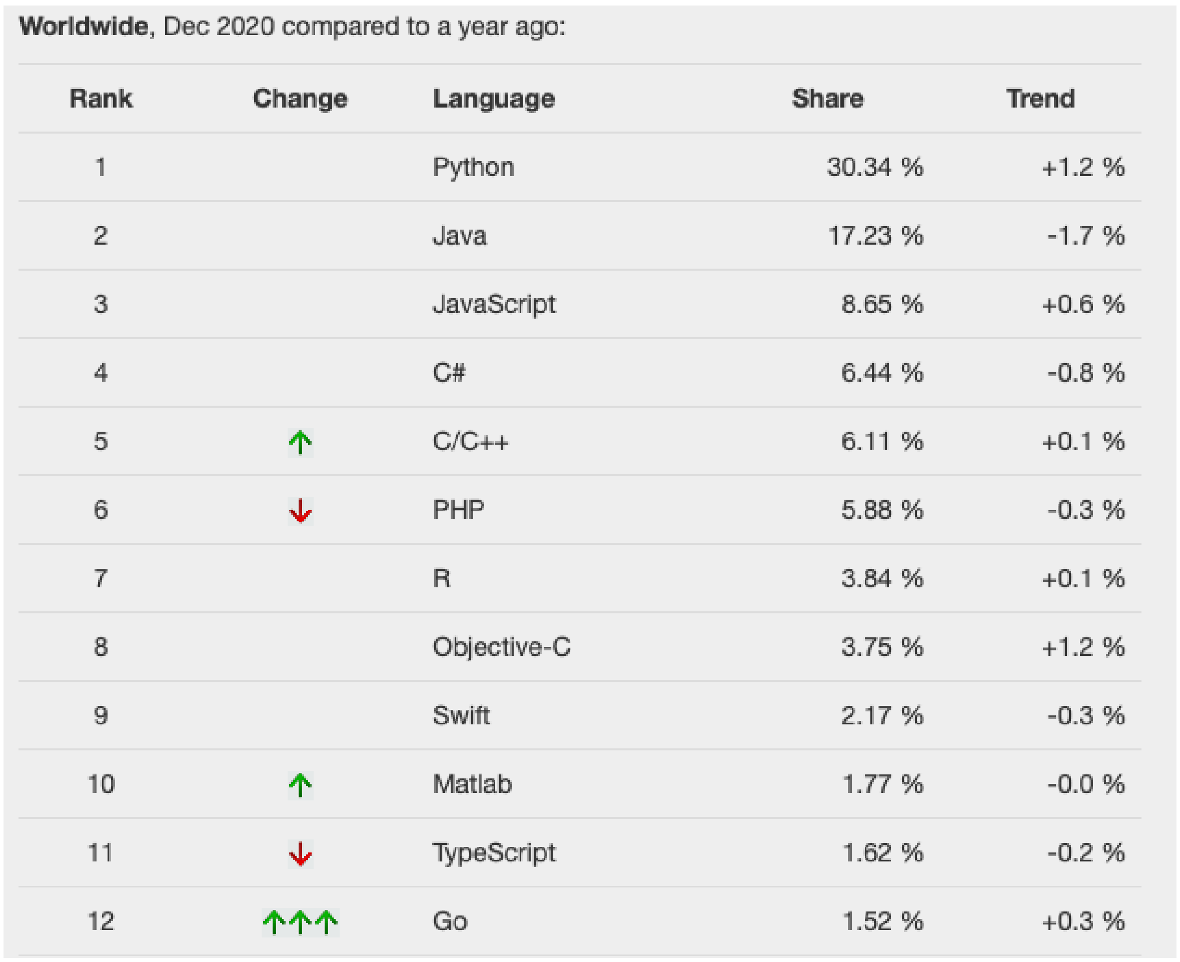
Indentation: Python uses whitespace indentation, instead of curly brackets or keywords to delimit blocks. An increment in indentation comes after certain statements as an example, a decrease in indentation implies the conclusion of the current block. In this way, program's visual structure accurately speaks to the program's semantic structure. This feature called as off-side rule, which some of the other languages share, but most the languages, indentation does not have any semantic meaning.

Statements: The task statement (token '=', the equals sign). This operation in an unexpected way than in traditional imperative programming languages, and this crucial mechanism (counting the nature of Python's version of variables) illuminates numerous other highlights of the language, as an example, x = 2, translates to "typed variable name x gets a copy of numeric value 2".

Expressions: Some of the python expressions are similar to other languages like C and Java, while some of them not. Python uses human words like “and or, not” for his boolean operators rather than the symbolic &&, ||, ! used in Java and C. [14]

Based on PYPL popularity index programming languages analysed, in a way how regularly language tutorials are reviewed via Google. The more a language tutorial is reviewed, the more well known the language is assumed to be. It is a driving indicator. The crude information comes from Google Patterns.[15]

According PYPL community index Python takes 1st place in worldwide 2020, However its 30.34% of total market which means python one of the tops wanted programming language according Google search, and over the 1-year python increased 1.2 times. It shows that Python going to stay on the top over the upcoming years.



Worldwide, Dec 2020 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	30.34 %	+1.2 %
2		Java	17.23 %	-1.7 %
3		JavaScript	8.65 %	+0.6 %
4		C#	6.44 %	-0.8 %
5	↑	C/C++	6.11 %	+0.1 %
6	↓	PHP	5.88 %	-0.3 %
7		R	3.84 %	+0.1 %
8		Objective-C	3.75 %	+1.2 %
9		Swift	2.17 %	-0.3 %
10	↑	Matlab	1.77 %	-0.0 %
11	↓	TypeScript	1.62 %	-0.2 %
12	↑↑↑	Go	1.52 %	+0.3 %

Figure 5. PYPL Popularity of programming languages. Source: [15]

However, it’s clear that each developer has their own taste because of it in the market there is lots of option which developer can pick any programming language and work on in different products with different companies.

3.3.1 Python Django Framework

Django provides a high-level framework that empowers developers to construct Web application with less line of code comparing other frameworks. It is basic, robust, and adaptable, permitting user to plan solutions without much overhead. Django was built utilizing Python, an object-oriented applications development language which combines the power of frameworks languages, such as C/C++ and Java, with the ease and fast development of scripting languages, such as Ruby and Visual Basic. This gives users the capacity to make applications that solve numerous different types of problems. [16]

Django uses MVC as the Model-Template-View (MTV) architecture. There's separation of concerns between the database interfacing classes (Model), request-processing classes (Views), and a templating language for the ultimate introduction (Template). Comparing Django architecture with classic MVC-"Model", it is absolutely comparable with Django's Models, "View" identifies as Django's formats, and "Controller" is the system which processes an incoming HTTP request and routes it to the proper view function.

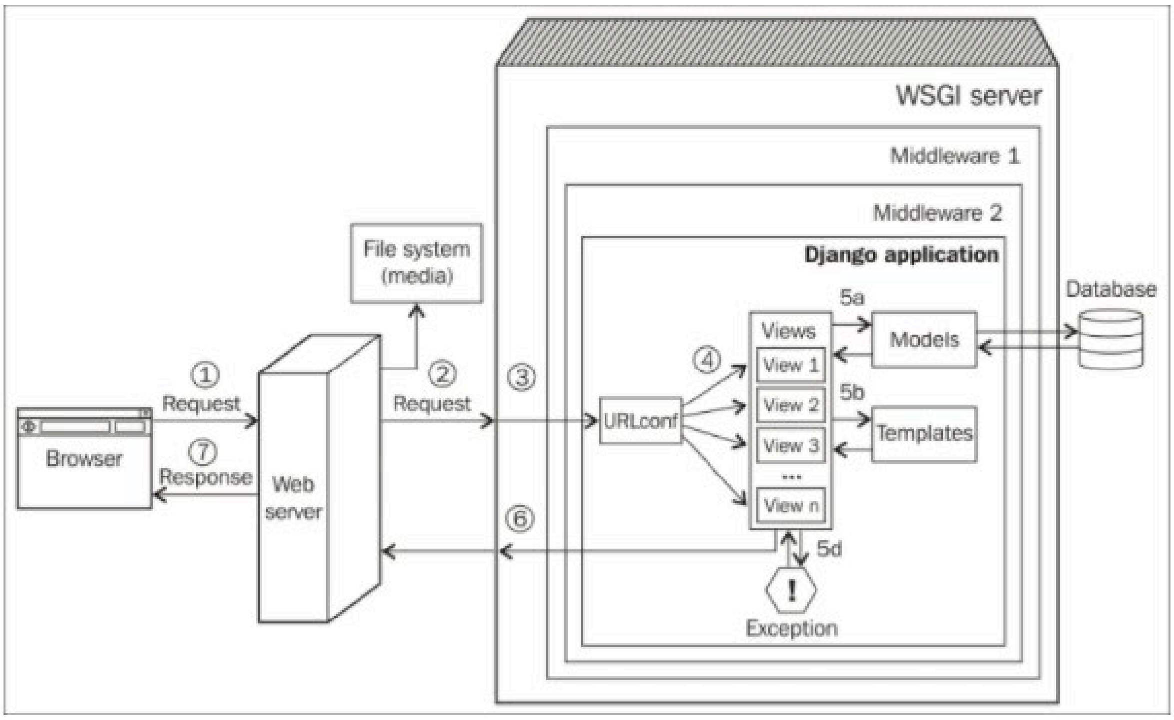


Figure 6. How does Django work. Source: [17]

The figure-6 shows the simplified journey of a web request from a visitor's browser to our Django application and responding back to user. This are number steps:

1. The browser sends the request (basically, a string of bytes) to our web server.
2. Our web server (say, Nginx) hands over the request to a WSGI server (say, uWSGI) or straightforwardly servers file (say, a CSS file) from the filesystem.
3. WSGI server runs Python applications where the request appears a Python dictionary called environ and alternatively, passes through a few layers of middleware, eventually coming to our Django application.
4. URLconf contained within the urls.py of our application chooses a view to handle the request based on the asked URL. The request has turned into HttpRequest in Python object.
5. The chosen view ordinarily does one or more of the fallowing things:
   - 5a. Talks to a database through the models
   - 5b. Renders HTML or any other organized response utilizing templates
   - 5c. Returns a plan content response(it's not shown in figure)
   - 5d. Raises an exception

6. The HttpResponse object gets rendered into a string, because it takes off Django application.
7. Rendered web page is seen in our user's browser.

Through certain details are overlooked, this representation should help us appreciate Django's high-level architecture. Moreover, it shows the roles played key components, such as models, views, and layouts. [17]

Why we have to use Django? Let's try to answer given question by looking following list of advantages of using Django:

Django distributed under the BSD licence, which guarantees that web applications can be utilized and modified freely without any issues as well as it's free.

Django is completely customized. Engineers can adjust to it effectively by creating modules or superseded framework methods.

This modularity includes other advantages. There is a part of Django modules simply can coordinate into Django. Developers can get a few helps from other people's work since you may frequently find high-quality modules you might need.

Utilizing Python in this framework permits us to have benefits from all Python libraries and guarantees a really great readability.

Django is a framework whose primary objective is perfection. It was uncommonly made for people who need clear code and a great architecture for their applications. It completely respects the Don't Repeat Yourself (DRY) philosophy, which implies keeping the code basic without having to copy/paste the same parts in different places.

With respects to quality, Django coordinating lots of effective ways to perform unit tests.

Django is supported by a great community. This a really important resource since it permits you to resolve issues and settle bugs exceptionally fast. Thanks to the community, we able to find code example that shows the best practises.

Django has got a few disadvantages as well, when an engineer begins utilize a framework, he/she starts with a learning stage. The duration of this stage depends on the framework and developer. The learning, stage of Django is moderately short if the developer knows Python and object-oriented programming. Moreover, it can happen that a modern version of the framework is distributed that adjusts a few syntaxes. For example, the syntax of the URLs within the formats was changed with version 1.5 of Django. [18]

3.4 JavaScript programming language

JavaScript is an object-oriented programming language based on the prototype-based object model. The language is best known for its utilize as a scripting language on the internet. JavaScript is a key building block of Dynamic HTML (DHTML) a collection of technologies that are included in about all web browsers to support the creation of animated and interactive web applications. [19]

JavaScript, HTML, and CSS have ended up so predominant, that numerous working frameworks have received the open web guidelines as the presentation layer for local apps, counting Windows 8, Firefox OS, Elf and Google's Chrome OS. Moreover, the iPhone and Android mobile gadgets support web views that permit them to join JavaScript and HTML5 functionality into local application. In any case, JavaScript is additionally moving into the hardware world. Projects like Arduino, Tessel, Espruino, and NodeBots foreshadow a time within the close future where JavaScript could be a common language for implanted frameworks and robotics.

3.4.1 Advantages of JavaScript.

JavaScript did not fair luckiness into position as the domain client-side language on the web sphere. It is really very well-suited language that took over the world. It is one of the foremost progressed and expressive programming languages created to date. The taking after areas outlines a few of the futures you may or may not be recognizable with.

Performance. JavaScript compiled is highly optimized and executed like local code> Runtime execution is near to C or C++ programming languages.

Objects. JavaScript has exceptionally wealthy object-oriented features, and JSON (JavaScript Protest Country) library utilized in about all advanced web applications for communication and data persistence is a subject of JavaScript's amazing object-literal notation.

Syntax. With JavaScript syntax ought to be familiar anyone who has experience with C-family languages, like C++, Java, C#, and PHP. Part of the JavaScript popularity is due to its recognition, through its important to get it that JavaScript carries on very differently from all of those under the hood. JavaScript's object-literal syntax is so basic, flexible, and brief, it was adjusted to become the dominant standard for client/server communication in front of JSON.

Events. Inside the browser, everything runs in an occasion circle. JavaScript coders quickly learn to think in terms of even handlers, and as a result, code from experienced JavaScript engineers tends to be well organized and proficient. Operations that can block processing in JavaScript happens concurrently.

Reusability. JavaScript code is more portable, reusable code around. What other language lets you compose the same code that runs natively on both the client and other server. JavaScript can be modular and encapsulated, and it is common to see scripts written by six distinctive groups of engineers who have never communicated working on the same page. [20]

3.4.2 JS Technologies (NodeJS, React, jQuery)

NodeJS is a program stage that built on Chrome's V8 JavaScript runtime for building adaptable network applications easily. NodeJS uses an event-driven, non-blocking I/O model that creates it lightweight and effective, idealize for data-intensive real-time applications that runs over distributed devices. Why JavaScript included in NodeJs?

Asynchronous. JavaScript is normally asynchronous with occasion model well suited for building profoundly versatile web applications through callbacks.

Less Learning curve. Lots of programmers already got familiard with both JavaScript and asynchronous programming.

Lighting Fast Script engine. Colossal advances in execution speed has made it practical to type in server side computer program completely in JavaScript

Code Sharing. Developers can type in web applications in one language, which makes a difference by reducing the "context", between client and server development, permitting code sharing between client and server.

Code Transformation. JavaScript is a compilation target as well as there are a number of languages that have compiled already to it.

Support for NoSQL. JavaScript language utilized in different NoSQL databases such as CouchDB / MongoDB interfacing with them is a natural way.

JSON. is one of the popular data interchange format that uses today and it is native JavaScript.

NodeJS architecture platform comprises of three layers. The base layer contains all the core components, middle layer acts as a middle-ware by setting up communication from lower to best layer. The ultimate top layer comprises of all JavaScript API. [21]

Respond is UI library created at Facebook to encourage the creation of interactive, stateful and reusable UI components. It utilized at Facebook productions. ReactJS is best for rendering complex client interfaces with tall execution. The fundamental principal behind React is the concept of virtual DOM. ReactJS effectively utilize virtual DOM, which can be rendered either at client side or server side and communicate back and forward. The Virtual DOM render subtrees of nodes based upon state changes and its slightest amount of DOM manipulation in order to keep your components up to date. React is lighter than Angular and it filled with the least conditions and eliminates the need to utilize additional components like plugins. React is against two-way binding, it intentionally remains away from it and make utilize of explicit updates instead. [22]

ReactJs features:

Declarative. React makes an intelligent and dynamic UI for sites and versatile applications. Make essential viewpoints for each state in your application. React successfully revive and deliver the perfect parts when your data changes. Authoritative perspectives make your code more recognizable and less complex to debug.

Short and Easy Learning Curve. The clear and non-complex nature of ReactJS empowers one to rapidly get settled with the structure. React is included in various strong highlights. Comprehensibility is likely the leading quality of React. It is successfully comprehensible indeed to the people who are new to it. Whereas different frameworks require learning various ideas about the structure itself, neglecting the language basics, React does verifiably the inverse. So, beginning levels of ability within the structure can without a doubt be refined without any obstacles or complexities.

One-way data binding. ReactJS is one-way unidirectional information stream between the state and layers in a web application, it means data can stream in a single direction between the application state and layers. On the other hand, in two-way data binding like Angular if model is changed, the view will change and vice versa. The advantages of single course data restricting provide you superior control all through the application.

JSX. JSX can be considered as an increased language structure that enthusiastically takes after HTML. It is essentially comparative to a blend of JavaScript and XML. JSX makes making React segments, the structure squares of React UIs. It lets you determine the DOM components before the components specifically within JavaScript records. This suggests the rationale behind the parts and the visuals are across the board place. This can be such a great thought when different structures are expecting lines to position them. JSX is without a doubt uncommon compared to other ReactJS highlights. [23]

jQuery's popularity is capacity to help in wide range of tasks. The jQuery library gives a general-purpose abstraction layer for common web scripting, and therefore, valuable in nearly every scripting situation. Its extensible nature means that we might never cover all possible uses and capacities. As plugins are always being created to add unused abilities. To preserve the wide, extend of features outlined over whereas remaining relatively compact, jQuery utilizes several techniques such as:

Leverage knowledge of CSS. By utilizing the mechanism for finding page components on CSS selectors, jQuery acquires a concise, however clear way of communicating a document's structure. The jQuery library gets to be an entry point for designers who need to include behaviors to their pages because a prerequisite for doing proficient web development.

Support extensions. In order to dodge feature crawl, jQuery relegates special-case uses to plugins and the strategy for creating unused plugins is basic and well-documented, which has impelled the development of a wide variety of inventive and valuable modules. Most of the fundamental jQuery download are inside realized through the plugin architecture.

Abstract away browser quirks: A sad reality of web development is that each browser has its own set of deviations from distributed measures. jQuery includes an abstract layer that normalizes the common assignments, reducing the measure of code.

Always work with sets. When we taught jQuery, "Find all components with the course collapsible and cover up them," there's no need to circle through each returned component, instead methods like hide() are outlined to automatically work on sets of objects rather than individuals ones. This strategy, called verifiable emphasis, means that numerous looping gets to be pointless.

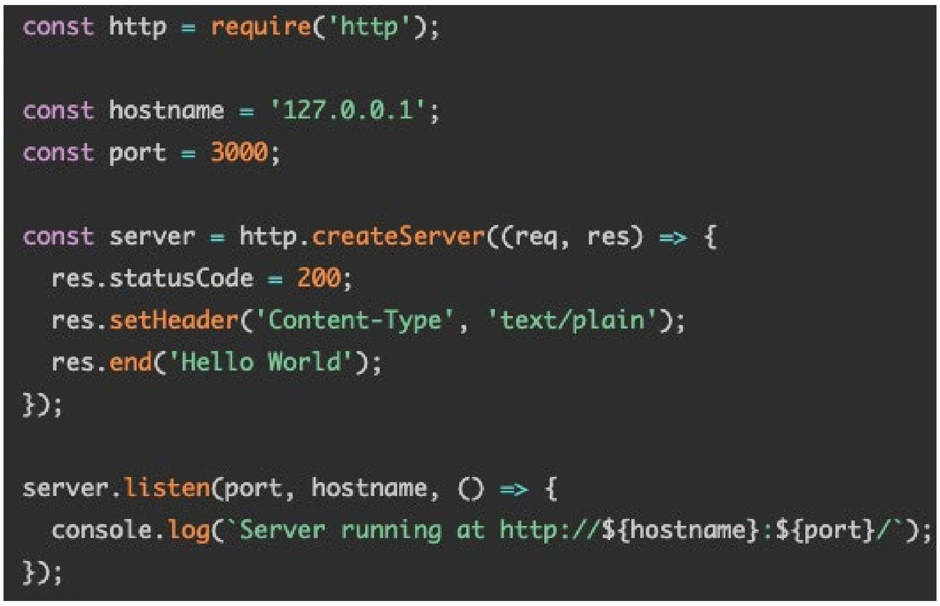
Allow multiple actions in one line. To maintain a strategic distance from abuse of transitory factors or inefficient repetition, jQuery utilizes a programming pattern called chaining for the larger part of its methods. This means, result of operations on an object is the object itself. [24]

What jQuery is good at? The core is excellent at navigating the DOM tree and selecting components. It's exceptionally lightweight, with the generation version standing at 29kb, which is minified and compressed. That's a part of kick for a small file, when debugging and testing, you're superior off utilizing the uncompressed development version, which is 212kb. Moreover, jQuery is nice at taking care of fairly complex JavaScript code with relatively small code. This was one of numerous initial attractions to the framework. For a newbie attempting to learn the "ropes" with Ajax, jQuery brings down the learning curve. Understanding what's going on under the hood is critical, but first step, utilizing $.ajax() isn't an awful to start. [25]

3.5 Comparison Python and JavaScript frameworks

In this section, we will take a look server-side, security, template-documentation and database comparison of Django and NodeJS. In any case, within the nearness of so numerous scripting languages or technologies accessible for server-side handling, it is usually a difficult choice for a beginner which technology to learn or utilize for development.

NodeJS, a straightforward installer may be utilized to get the technology locally in order to start developing. The installation regularly includes the node package manager as well any command line devices to get started with nearby server development. In any case, the installation package does not incorporate any database administration tools, but left to the engineer to select openly which database system would be necessary to make their wanted product. The code for making a "Hello, World!" for Node.JS can be seen in fallowing table.



```javascript
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Figure 7. Node.js, ''Hello World!'' server code. Source: [26]

Node.js depends intensely on the utilize of packages from the Node Package Manager (npm) to complete assignments. The primary line of code requires that an http package be utilized to create a server. Another line specifies the hostname and port number, which is localhost address and port 3000. To create a server, which is createServer function is utilized where the response status code, the header, and the data are set. At last, the server listens on the port and indicated host.

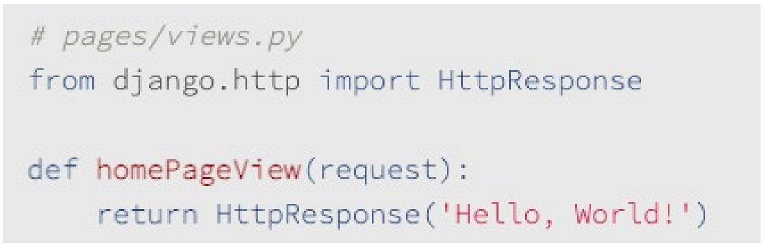
Security: Node.js does not provide security settings, developers have to pass several steps in order to add security to make secure their app.

Template and Documentation: Node.js is an adaptable to design and customize templates and its favoured tool for experienced software engineers. On the other hand, to construct an app may take more time than Django as you need to construct app from the scratch.

Scalability: Node.js is non-blocking I/O and event driven architecture, supports microservices. Moreover, Node.js can build real time streaming apps, networking and data intensive apps.

Django, is very simple. The first step is to install python if it is not already installed, and after that utilize the python package manager type there, pip install Django. The development form of Django can be installed by from GitHub. Django does not provide any sort of database management system and its left to the developer who will create a project.

Creating a Django "Hello World!" is simple, but it is important to modify the file controlling server settings, in order to be able to properly develop it from local host. [27] The project code shown in the following table.



```python
# pages/views.py
from django.http import HttpResponse

def homePageView(request):
    return HttpResponse('Hello, World!')
```

Figure 8. Primary code responsible for creating "Hello, World!" in Django. Source:[28]

The function called homePageView that takes a request parameter and returns an HttpResponse with the “Hello, World!” text to our screen as the parameter.

Database: Django supports many variants of the database. Majority of the database backend is already included in the framework. According to the official documentation website, curPostgresSQL, MariaDB, MYSQL, Oracle and SQLite are supported out of the box.

Security: Since Django is an open-sourced project, all the security tools of the framework are thought out. It covers the following security issues:

- Cross-site request forgery protection (CSRF)
- Cross-site scripting (XSS)
- SQL injection
- Clickjacking
- Host header validation
- SSL/HTTPS

Django’s templating framework ensures the app from the majority of XSS attack. Within the case of CSRF, it contains a Csrf Middleware which checks whether the requests referring header is coming from the same origin. Django comes with inbuilt ORM which helps one to inquiry the database without hardcoding the SQL code. These queries sets constructed utilizing query parameterization. By utilizing these standard query sets most of the SQL injections issues are resolved already. In terms of Clickjacking issue, the system comes with the middleware called X-Frame-Options and Django comes with the plausibility to configure the SSL/HTTPS, so that end to end requests and response are encrypted.[29]

Both Node.js and Django have extraordinary tools/platforms for building web-based applications. Whereas Node.js could be a server-side runtime environment, it is perfect for apps which require strongly information handling such as social organizing locales, live-streaming apps, amusement apps, and computer program with critical calculations. On the other hand, Django is high-level python system that might be utilized for backend and frontend improvement such as full-stack apps, apps with fast advancement as prime require, and developers at beginner-level. Considering Node.js vs Django, Django is almost your choice whereas Node.js is for proficient web designers who need to construct real-world apps. Django is built for highly scalable apps.

4 Practical Part

4.1 Research questions

Based on the conducted literature review, the following research questions were formulated. These questions were
RQ1: What are the differences between Django and JavaScript frameworks in implementation of an e-commerce and real-time chat websites?
RQ2: What parameters are important to consider during framework selection for an e-commerce website development?

4.2 Research design

In the practical part, two web applications were built, both applications implemented based on theoretical part of the thesis. However, mostly author tried to focus on Django rather than Node.js as it is main goal to demonstrate capability of Django by comparing JavaScript frameworks.

In the first project, a simple beginner friendly functional website was built with user and guest checkout capabilities by using Django. On the other hand, in the second project was used JavaScript library Node.js framework where was built real time chat room app. For second application basic JavaScript was used and node.js to handle backend side of the work. However, let's take a look first project features, and what type of technologies author used in order to implement it.

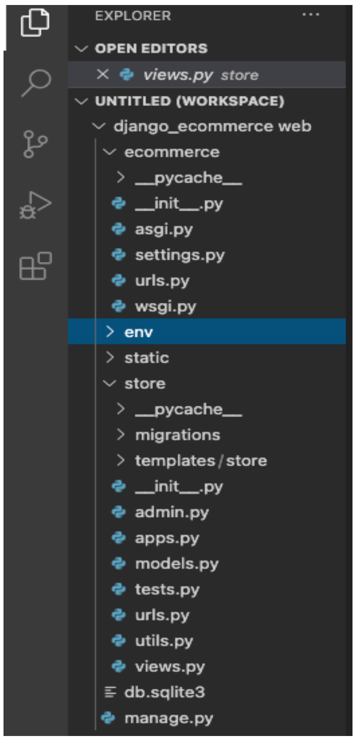
Author tried to keep it very beginner friendly project as author was also beginner on this area, during the development did not used any frameworks for the front-end, and any RestAPI, but used very basic JavaScript to handle front-end portion of things, to take care of add to cart functionality and side cookies. Moreover, most of the portion of things to handle author tried to use Django and most of the comparison was based on back-end side of the website as Node.js also used for back-end.

Type of features that website has:

- Authenticated user vs Guest checkout have almost the same process with some differences.
- Authenticated, add item to cart- edit order- checkout and view pending and previous order.
- Guest user, add item to cart- edit order- checkout.

As operating system used macOS and usually for the mac python installed already, it was checked by typing command "python - -version" in the terminal it showed an older version of python 2.7.14 which is not recommended to use as the official supporting was ended in 2020. Then author visited official website of python and downloaded the latest version from there. Usually, it comes with pip package but just in case if you don't get it, we just need to type command "install pip", pip allows to the user to instal python libraries and as Django also python library author typed command "pip install django". Once Django installed in our operating system author used VS code(IDE) as text editor for our code. However, there is a lot of IDE's where developer can choose according his/her taste, and VS code is one of the popular IDEs for writing code. Lastly, author created virtual environment for website by typing command “django-admin.py startproject ecommerce”, after this command it created all python files you can see in the fallowing figure.

As shown in the Figure-9 below, an organization of different sections of apps has been packaged in a separate directory. Within those directories, we can see “admin.py”, “models.py” and “views.py”, “urls.py” those are the standard filename from where Django tries to find the resources. Most of the database related implementation resides on the models.py and views.py consisting of the endpoint serving functions and urls.py containing the routings.



```
EXPLORER ···
∨ OPEN EDITORS
  × views.py store
∨ UNTITLED (WORKSPACE)
  ∨ django_ecommerce web
    ∨ ecommerce
      > __pycache__
        __init__.py
        asgi.py
        settings.py
        urls.py
        wsgi.py
    > env
    > static
    ∨ store
      > __pycache__
      > migrations
      > templates / store
        __init__.py
        admin.py
        apps.py
        models.py
        tests.py
        urls.py
        utils.py
        views.py
      db.sqlite3
      manage.py
```

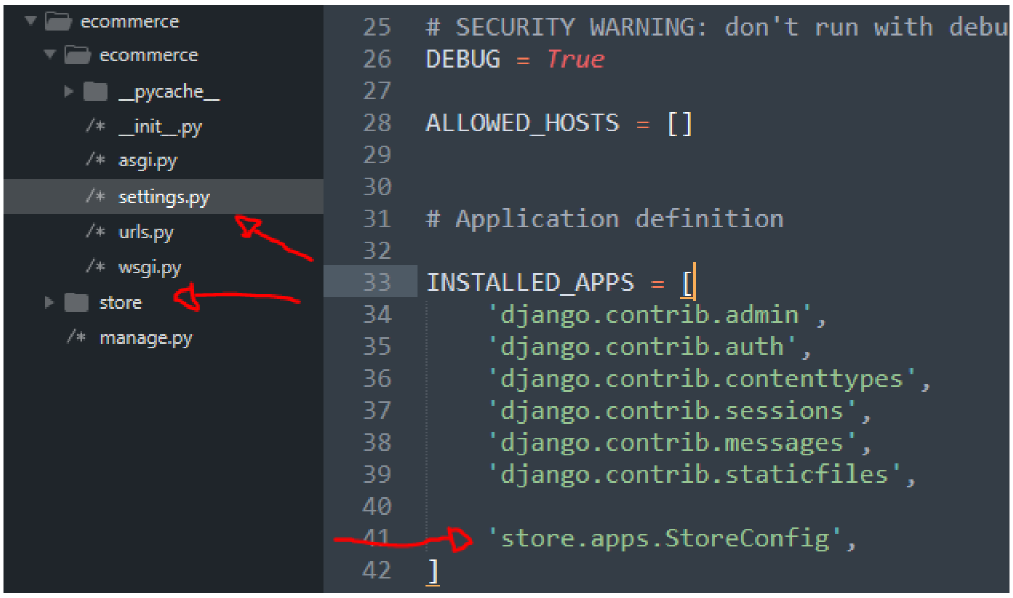
Figure 9. Design pattern. Source: author

4.3 Prototype of Django web application

This section explains how author implemented ecommerce web application by using Django and little bit of JavaScript for front-end portion of work. Project view is created in mind as well as while prototyping pen and paper was used for taking some notes and design.

While creating the website core features of Django was used such as requests and routing, blueprints, and database. Firstly, as it was mentioned earlier, author checked python in macOS than downloaded latest version of it as well as pip package was checked and version of it, but pip version was not latest and it asked to update it, author updated it by typing following command in the terminal. **“pip install –upgrade pip”** this command gave the latest version of the pip 20.3.3 and once it was done, ecommerce project was created with Django, and it was already mentioned command in the previous section. Lastly virtual environment was created to our project and VS code was used as text editor as it’s common text editor for author. All created files were mentioned in research design section, in figure-9 so you can see it a list of files that were created for the project.

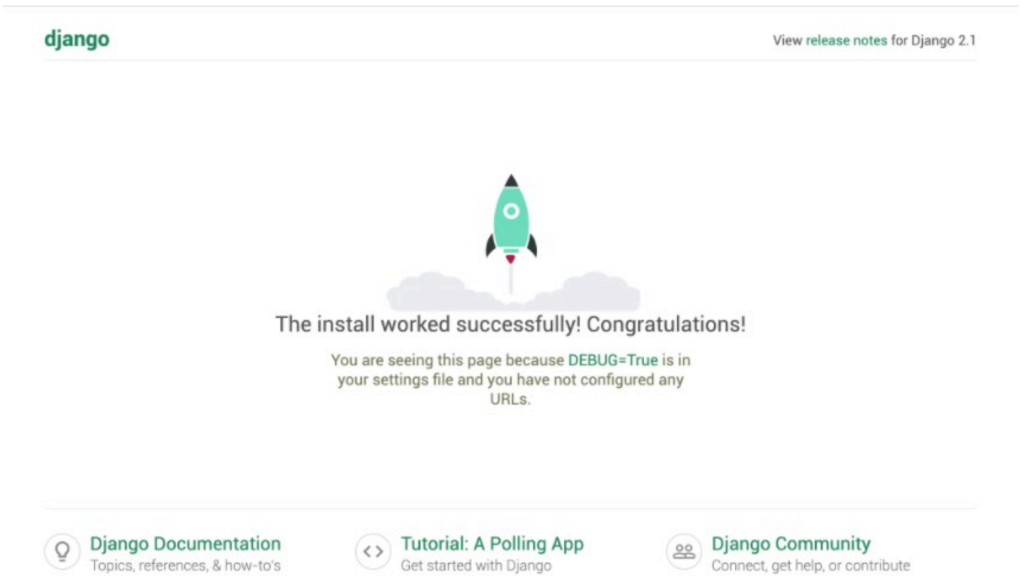
In the first step, store app was created and inside of store app author created template file where html files were stored. Once it's done, author opened settings.py and author added their store app in order to run code smoothly, and you can see it in following figure-10.



```
# SECURITY WARNING: don't run with debu
DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'store.apps.StoreConfig',
]
```

Figure 10. Add app to settings.py. Source: author

Once it was finished, the program was checked for correctness by typing an inside terminal command "python manage.py runserver" that should open the following window and a port 127.0.0.1:8000 (figure-11).


django
View release notes for Django 2.1
Django Documentation
Topics, references, & how-to's
Tutorial: A Polling App
Get started with Django
Django Community
Connect, get help, or contribute

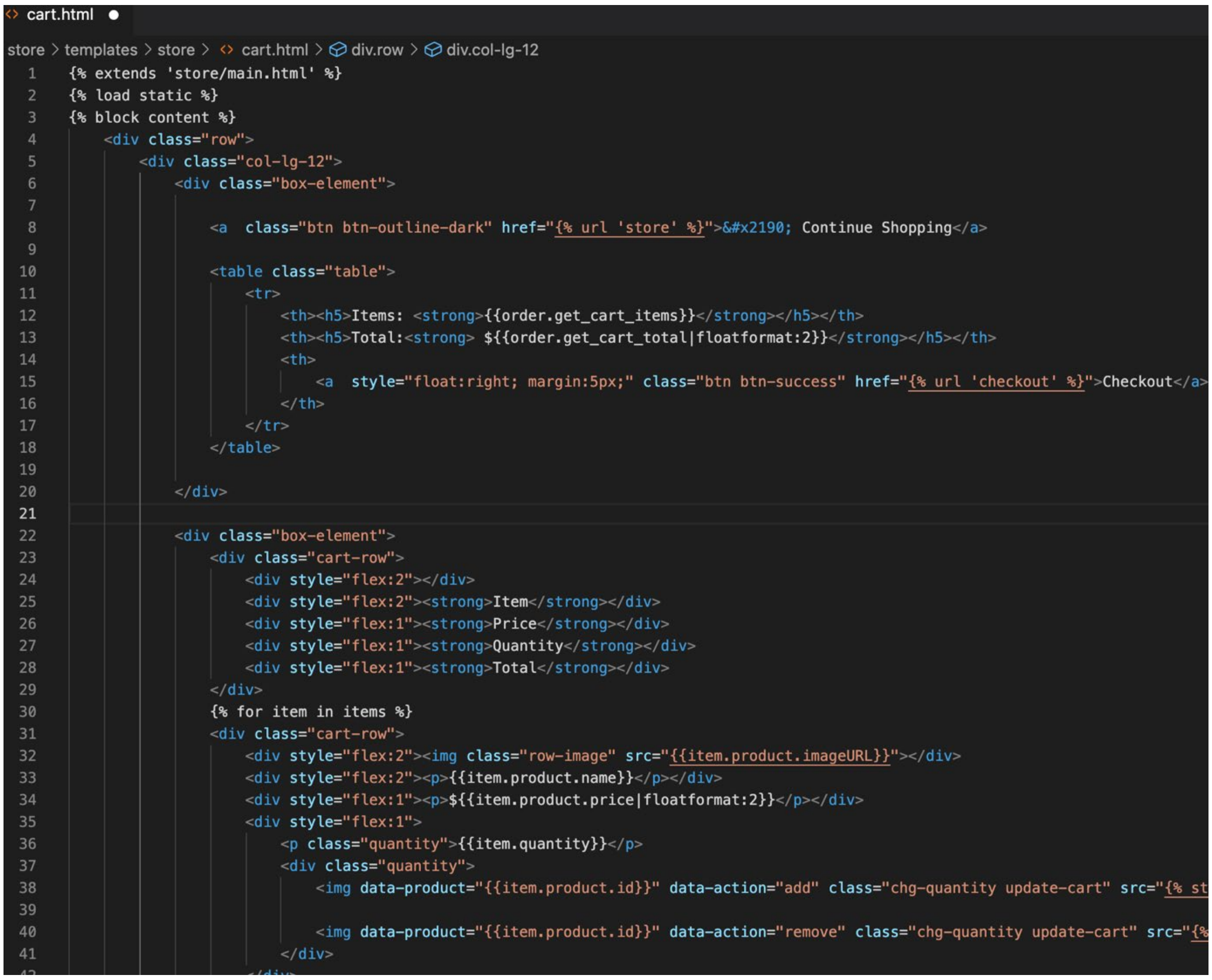
The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

Figure 11. Django site. Source: author

If the following window opens, it means that all commands were typed correctly and we can move on to next steps. In the second step author worked on template folder and inside of this folder, html files created as example main.html, store.html, cart.html, and lastly checkout.html

- In main.html there is template which all will inherit from
- In store.html it's our home page/store font with all products.
- In cart.html it's our users shopping cart and checkout is checkout.



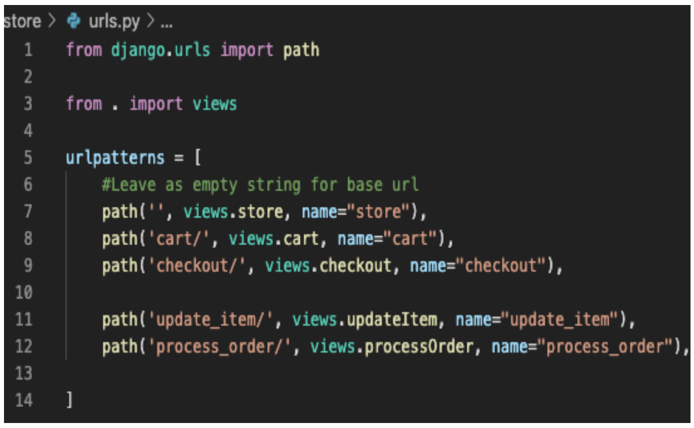
cart.html

store > templates > store > cart.html > div.row > div.col-lg-12

```
{% extends 'store/main.html' %}
{% load static %}
{% block content %}
    <div class="row">
        <div class="col-lg-12">
            <div class="box-element">

                <a  class="btn btn-outline-dark" href="{% url 'store' %}">&#x2190; Continue Shopping</a>

                <table class="table">
                    <tr>
                        <th><h5>Items: <strong>{{order.get_cart_items}}</strong></h5></th>
                        <th><h5>Total:<strong> ${{order.get_cart_total|floatformat:2}}</strong></h5></th>
                        <th>
                            <a  style="float:right; margin:5px;" class="btn btn-success" href="{% url 'checkout' %}">Checkout</a>
                        </th>
                    </tr>
                </table>

            </div>

            <div class="box-element">
                <div class="cart-row">
                    <div style="flex:2"></div>
                    <div style="flex:2"><strong>Item</strong></div>
                    <div style="flex:1"><strong>Price</strong></div>
                    <div style="flex:1"><strong>Quantity</strong></div>
                    <div style="flex:1"><strong>Total</strong></div>
                </div>
                {% for item in items %}
                <div class="cart-row">
                    <div style="flex:2"><img class="row-image" src="{{item.product.imageURL}}"></div>
                    <div style="flex:2"><p>{{item.product.name}}</p></div>
                    <div style="flex:1"><p>${{item.product.price|floatformat:2}}</p></div>
                    <div style="flex:1">
                        <p class="quantity">{{item.quantity}}</p>
                        <div class="quantity">
                            <img data-product="{{item.product.id}}" data-action="add" class="chg-quantity update-cart" src="{% st

                            <img data-product="{{item.product.id}}" data-action="remove" class="chg-quantity update-cart" src="{%
                        </div>
```

Figure 12. Cart.html file structure. Source: author

In the third step views and URLs was created to actually render this out and in Django routing is handled through URL dispatcher. The view function/class is written somewhere and the URL with the required pattern. Default URL dispatcher is already configurable in the settings. Once the root URL is configured, other URL routes can be added. These routes can either be global or application specific. The routes can be configured in the "urls.py" as shown below.



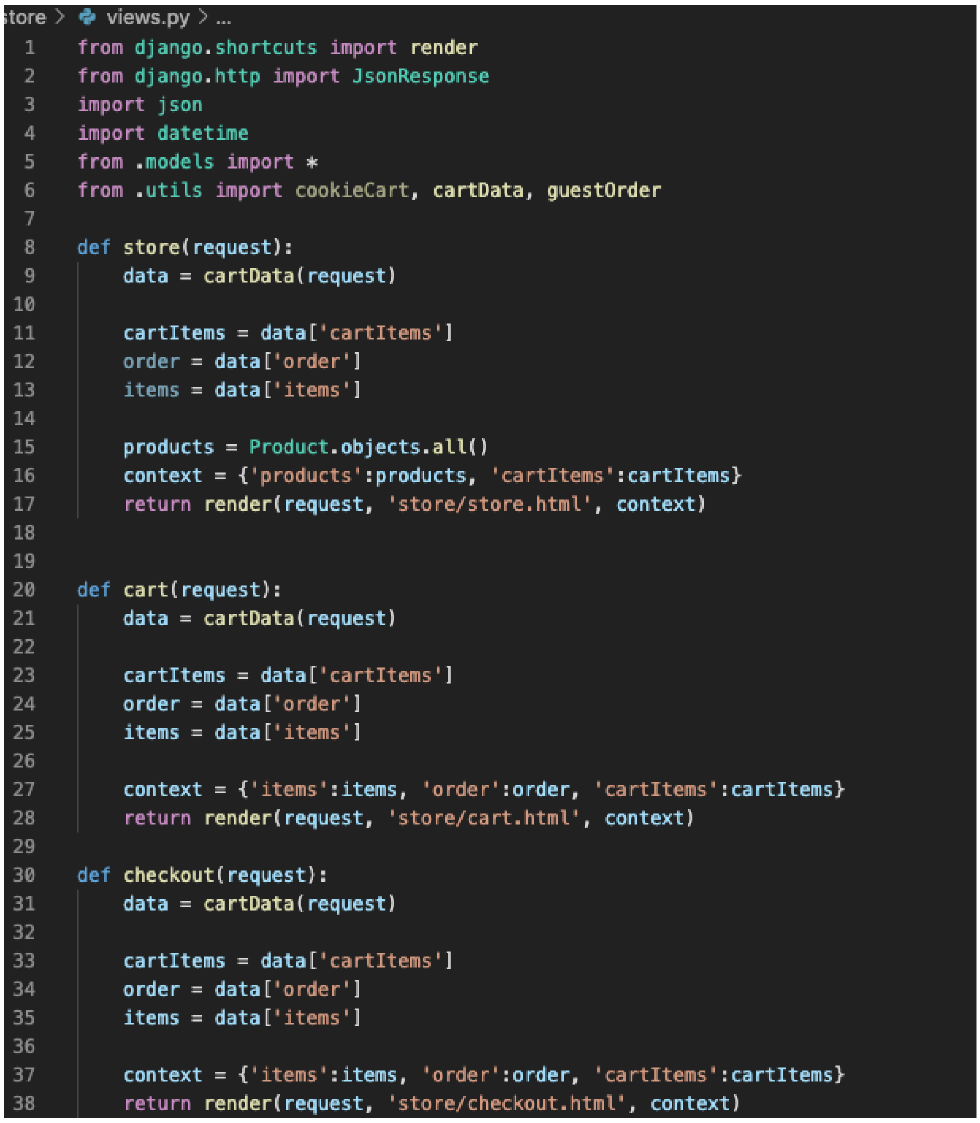
store > urls.py > ...

```
from django.urls import path

from . import views

urlpatterns = [
    #Leave as empty string for base url
    path('', views.store, name="store"),
    path('cart/', views.cart, name="cart"),
    path('checkout/', views.checkout, name="checkout"),

    path('update_item/', views.updateItem, name="update_item"),
    path('process_order/', views.processOrder, name="process_order"),

]
```

Figure 13. Django URL routing. Source: author

In the fourth step author worked on request. In Django, requests object needs to be explicitly provided to the view function or the class. The request object then contains all the required information from the current application state to the current session. When a client requests a resource from an application, the HttpRequest object is created, and the corresponding view function is called. If the request object needs to be modified or accessed, it needs to be provided explicitly to the implemented function.

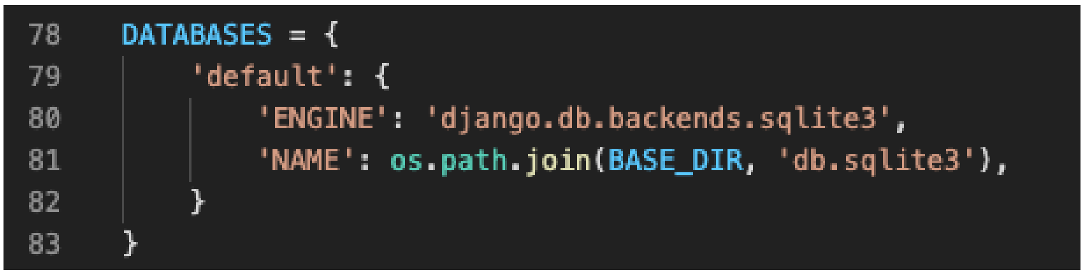
As shown in the figure below the request context is explicitly passed through the "store, cart and checkout" functions of a class-based view or access it through the class scope.



```
store > views.py > ...
from django.shortcuts import render
from django.http import JsonResponse
import json
import datetime
from .models import *
from .utils import cookieCart, cartData, guestOrder

def store(request):
    data = cartData(request)

    cartItems = data['cartItems']
    order = data['order']
    items = data['items']

    products = Product.objects.all()
    context = {'products':products, 'cartItems':cartItems}
    return render(request, 'store/store.html', context)


def cart(request):
    data = cartData(request)

    cartItems = data['cartItems']
    order = data['order']
    items = data['items']

    context = {'items':items, 'order':order, 'cartItems':cartItems}
    return render(request, 'store/cart.html', context)

def checkout(request):
    data = cartData(request)

    cartItems = data['cartItems']
    order = data['order']
    items = data['items']

    context = {'items':items, 'order':order, 'cartItems':cartItems}
    return render(request, 'store/checkout.html', context)
```

Figure 14. Django request. Source: author

As it was mentioned in previous section Django has many options to use database as it supports curPostgresSQL, MariaDB, MYSQL, Oracle and SQLite. In this project, however, the SQLite database was used with a very minimal relational database model is used. Database host, its backend and the port can be added in the settings file of use. For brevity the project does not use the separate database server, instead it uses the same host as a database server as shown below.



```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

Figure 15. Database backend. Source: author

Finally, author decided to deploy created project in the server but in the future. However, if author decide to deploy, author have to make sure that all settings configured correctly, in this fallowing list shows us mandatory settings for Django project deployment.

1. SECRET_KEY,
2. ALLOWED_HOSTS,
3. DEBUG,
4. CACHES,
5. DATABASES,
6. STATIC_ROOT and STATIC URL,
7. MEDIA_ROOT and MEDIA_URL

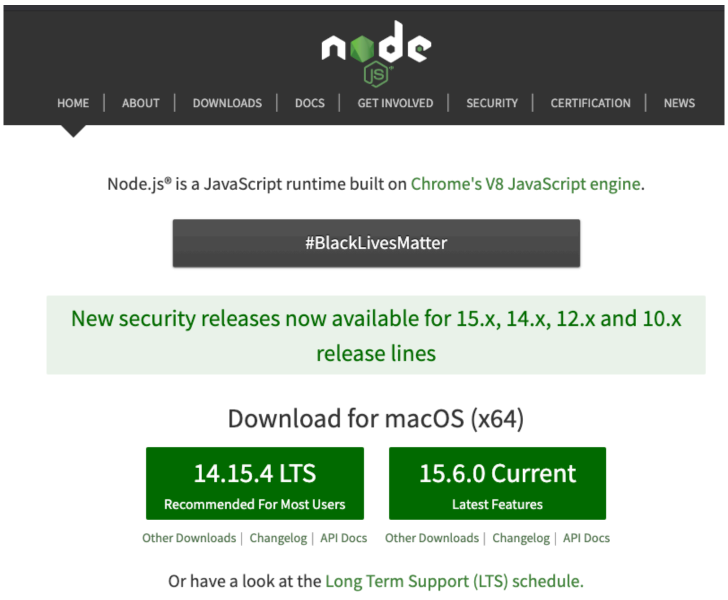
4.4 Prototype of Node.js web application

As a second project, real time chat web application was built, this project was chosen because node.js can handle large number of the users, based on review author found out that node.js is good for building real time chats and its simple to build as author was absolute beginner in node.js author decided to do this project, that's why this simple beginner friendly real time chat application was built, where 2 user can chat and see when user disconnected from the server, like previous project author not deployed project to the server it works from local host.

Fallowing technologiest was used in web application.

- Basic JavaScript
- HTML and CSS to give some style to our app
- Node.js and express for handling server side of our project
- Socket.io which allows us to use web socket which allow real time communication from the browser to the server as well as everybody else's browser that are connected to the server.

Just like in Django the first thing to do is to install the pre-requisites. In case with node there is a bit more to it though. First of all the node.js itself. This one is pretty simple. At the official website of node.js (https://nodejs.org/) there is a download link (see a screenshot below). All that is need to be done is pressing the green button and follow the instructions there:


HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

#BlackLivesMatter

New security releases now available for 15.x, 14.x, 12.x and 10.x release lines

Download for macOS (x64)

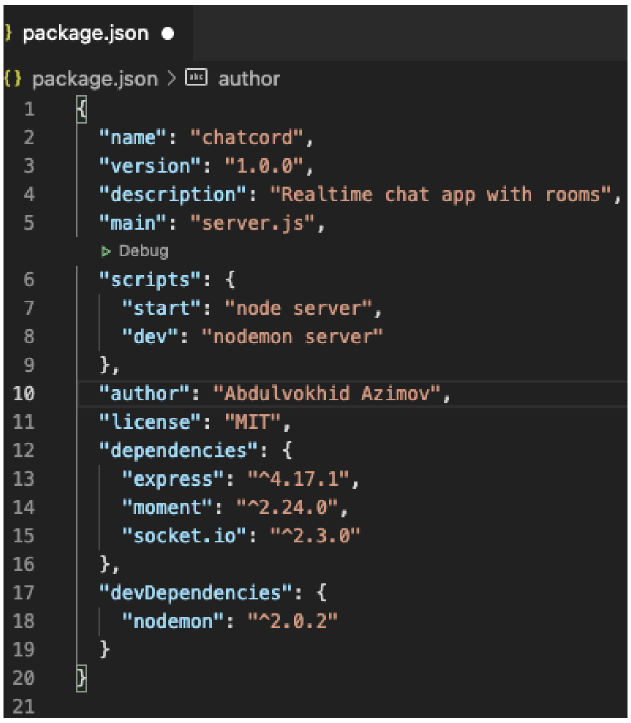
14.15.4 LTS
Recommended For Most Users
Other Downloads | Changelog | API Docs

15.6.0 Current
Latest Features
Other Downloads | Changelog | API Docs

Or have a look at the Long Term Support (LTS) schedule.

Figure 16.Node.js installation Source: [30]

Once node.js installed to macOS, we need to choose text editor to develop our code. As author used in the first project VS code in this second project also VS code was used as text editor. In VS code terminal we need to type fallowing command to initialize "package.json" file by doing "**npm init**'' once done it. You can see the file which we created fallowing figure.


package.json > author

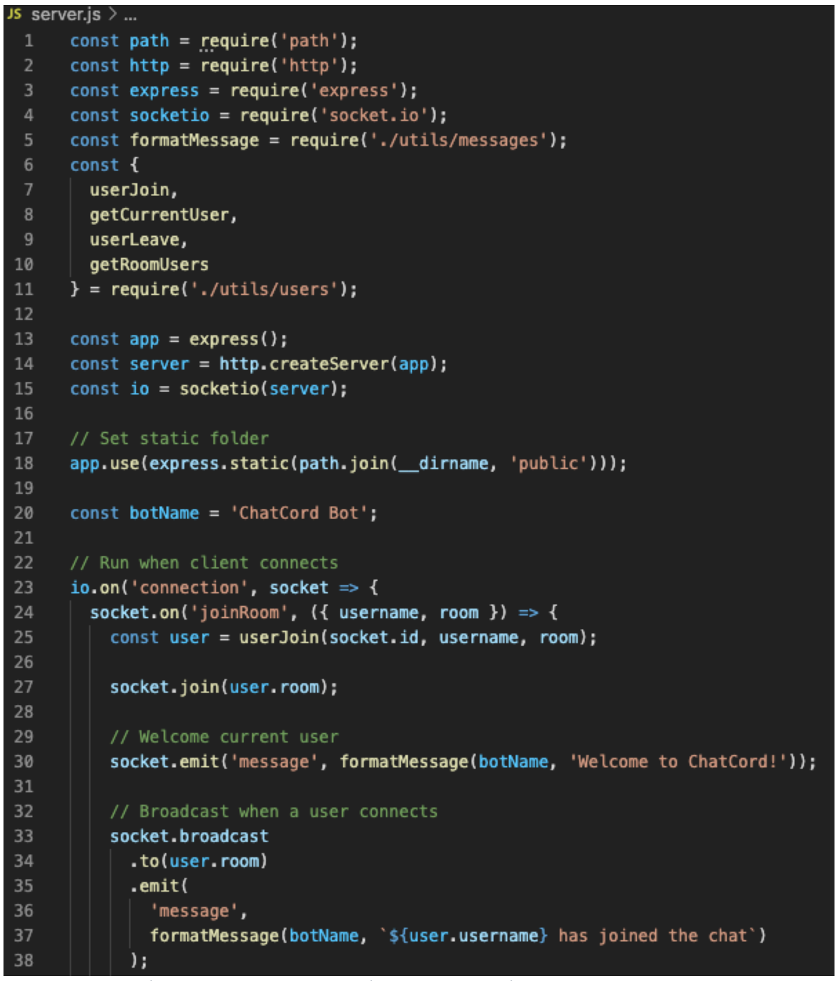
```
{
  "name": "chatcord",
  "version": "1.0.0",
  "description": "Realtime chat app with rooms",
  "main": "server.js",
  "scripts": {
    "start": "node server",
    "dev": "nodemon server"
  },
  "author": "Abdulvokhid Azimov",
  "license": "MIT",
  "dependencies": {
    "express": "^4.17.1",
    "moment": "^2.24.0",
    "socket.io": "^2.3.0"
  },
  "devDependencies": {
    "nodemon": "^2.0.2"
  }
}
```

Figure 17. Initializing package.json. Source: author

Npm node package manager. NPM is a built-in tool that is included by default with every installation of Node. NPM helps in easily managing modules in Node projects by downloading packages, resolving dependencies, running tests, and installing command line utilities.

In the second step express socket.io was created by typing command "**npm install express socket.io**" which deals web framework websocket it should create for us package-local.json file and author installed nodemon as dev dependency by writing fallowing command in VS code terminal "**npm install -D nodemon**" (-D means dev dependency).

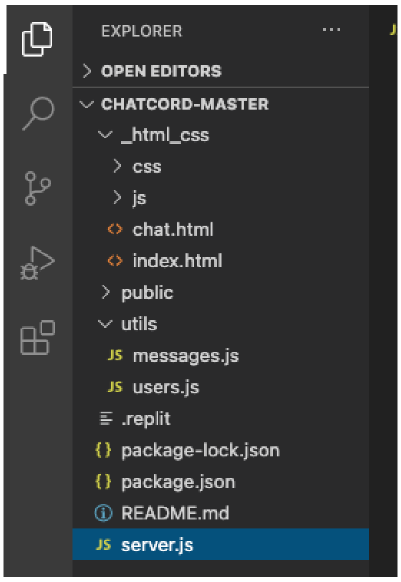
In the third step server.js file was created to deal all backend portion of work. How node.js works? The main distinctive features of the Node architecture are the usage of non-blocking, event-driven, asynchronous I/O calls that operate in a single thread. Conventional web servers handle concurrency by spawning new threads for each new request, which can max out the available memory. [31]


JS server.js > ...

```
const path = require('path');
const http = require('http');
const express = require('express');
const socketio = require('socket.io');
const formatMessage = require('./utils/messages');
const {
  userJoin,
  getCurrentUser,
  userLeave,
  getRoomUsers
} = require('./utils/users');

const app = express();
const server = http.createServer(app);
const io = socketio(server);

// Set static folder
app.use(express.static(path.join(__dirname, 'public')));

const botName = 'ChatCord Bot';

// Run when client connects
io.on('connection', socket => {
  socket.on('joinRoom', ({ username, room }) => {
    const user = userJoin(socket.id, username, room);

    socket.join(user.room);

    // Welcome current user
    socket.emit('message', formatMessage(botName, 'Welcome to ChatCord!'));

    // Broadcast when a user connects
    socket.broadcast
      .to(user.room)
      .emit(
        'message',
        formatMessage(botName, `${user.username} has joined the chat`)
      );
```

Figure 18. Node.js request source code. Source: author

Inside of server.js file author created all backend portion of works and this figure-18 shows some of them.


EXPLORER ···

OPEN EDITORS

CHATCORD-MASTER

```
_html_css
  css
  js
  chat.html
  index.html
public
utils
  messages.js
  users.js
.replit
package-lock.json
package.json
README.md
server.js
```

Figure 19. Design Pattern. Source: author

As you can see in figure-19 illustrates the final version of the file list for our second JavaScript project.

By finalizing the project, we did not use database and security settings. As this project did not require database and we did not deploy it, that's why we did not use any security tools. Comparing Django to Node.JS, Node.JS does not come with his own database and security tools. In order to apply database, we can install MongoDB to handle portion of tasks where data is required and there is JavaScript security tool named SSJS we can use it as well for make secure our application, as it was mentioned we did not use both tools as it did not need.

5 Results and Discussion

5.1 Django and Node.js comparison

This section tries to answer research questions that has been formulated in the section 4.1 while prototyping projects and this part focuses on RQ1: What are the differences between Django and JavaScript frameworks in implementation of an e-commerce and real-time chat websites?

1. Node.js is effective to use to build efficient and fast web applications. On the other hand, Django is efficient to use eliminating repetitive tasks.
2. Node.js and Django are both used to develop web applications and both of them is unique in their own ways.
3. Node.js architecture is run-time environment, works on an event-driven model. On the other hand, Django architecture is model-template-view, works with handling data.
4. Node.js complexity is better performance due to flexibility provided to developers. On the other hand, Django complexity is little more complex because developers have to follow fixed path to solve problems.
5. Node.js performance is less complicated system, more flexibility to developers. On the other hand, Django is high-performance framework due to built-in template system.
6. Node.js is runtime environment and on the other hand, Django is web framework.
7. Node.js security is comparatively comes, as less secure built-in tool, but manual operations covers the security deficiency, on the other hand, Django comes as more secure tool compared to Node.js, built-in system prevents from security threats.
8. Node.js is more flexibility due to rich ecosystem, giant JavaScript library, frameworks and tools, and on the other hand, Django flexibility is limited availability of tools and features, comparatively follows a strict approach.
9. Node.js does not come with his own database tools comparing Django, on the other hand, Django framework included majority database according to the official documentation website.

Finally, it could be said that learning both framework features, it turned out that there is a lot of frameworks to develop web application. Its only up to the developer and language what he knows he can simply pick which he likes and what he would like to build. Both, framework has its advantages and disadvantages and both are unique in their own way.

RQ2*:* What parameters are important to consider during framework selection for an e-commerce website development?

The important parameters for framework selection are summarized in Table 1 below.

	JavaScript(Node.js)	Python(Django)
Installation	Node.js,Socket.io (everything needs to be installed separately)	A single installation file that installs all required tools to the system
Package	npm (package manager for node.js)	pip (package manager for pyhton)

Request	Event Loop (single-threaded, non-blocking I/O operation)	In Django, requests object needs to be explicitly provided to the view function or the class
Database	Needs to take extra steps (does not come with his own tools)	Majority of the database is already included in the framework
Security	Needs to take extra steps to make secure(does not come his own tools)	-Django comes with inbuild ORM which helps the database without hardcoding the SQL code. -It comes with possibility to configure SSL/HTTPS end to end requests and response are encrypted.

Table 1. Results of comparison. Source: author.

5.2 Final view of websites

The implementation of both projects e-commerce and ChatCord apps development was carried out successfully. Both projects developed in order to understand key features of both programming language frameworks and to demonstrate capability of Python Django framework in web application area. For creating web applications, Django and Node.js was used.

For the first ecommerce application was used Django framework and somehow JavaScript to deal frontend side of work. As it was mentioned before Django comes with his own administrative files where user does not need to deal with it, he/she can just modify them based on his project needs. Moreover, Django comes with already included databases. In this project SQLlite was used for dealing database portion of work and was already shared in Figure-15.

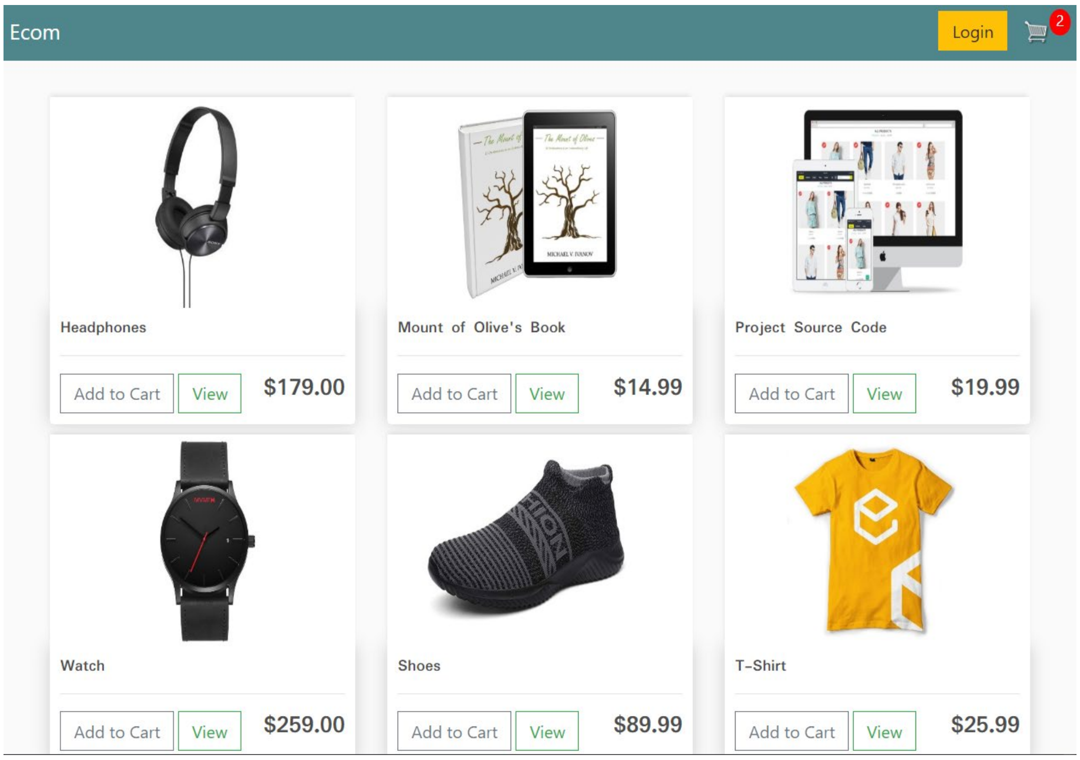
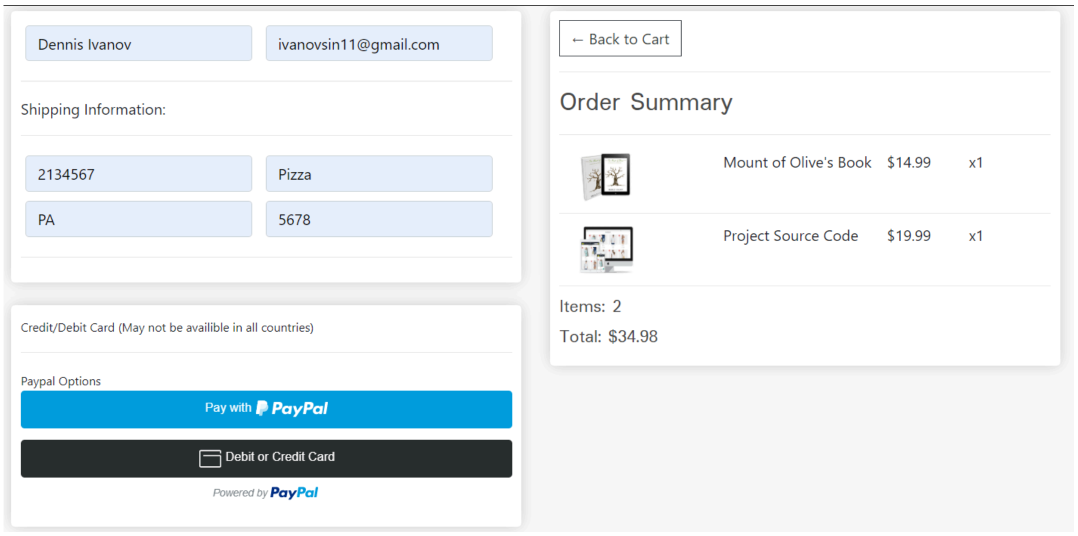


Figure 20. Main page. Source: author

Main page illustrates the ecommerce of the application where app has a navigation menu. On the top navigation bar, implemented login and cart options where user can check his/her order list. In main body, user able to see type of products price list and add to cart options.



Dennis Ivanov
ivanovsin11@gmail.com

Shipping Information:

2134567
Pizza
PA
5678

Credit/Debit Card (May not be availible in all countries)

Paypal Options

← Back to Cart

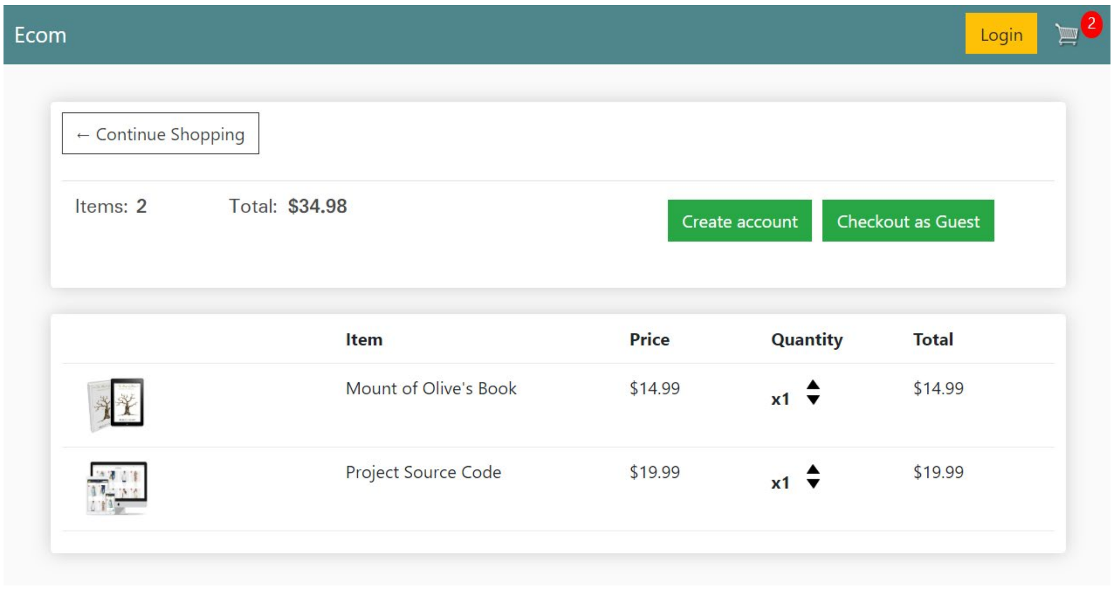
Order Summary

	Mount of Olive's Book	$14.99	x1
	Project Source Code	$19.99	x1

Items: 2
Total: $34.98

Figure 21. Payment integration. Source: author

In this page user need to fill his/her shipping address. Moreover, user able to see order summary as well as payment method. Shipping address will be available only when user creates an account otherwise as a guest user, he/she can just review products and cost. This figure 11 was created in our checkout.html file.


Ecom

Login

← Continue Shopping

Items: 2 Total: $34.98

Create account | Checkout as Guest

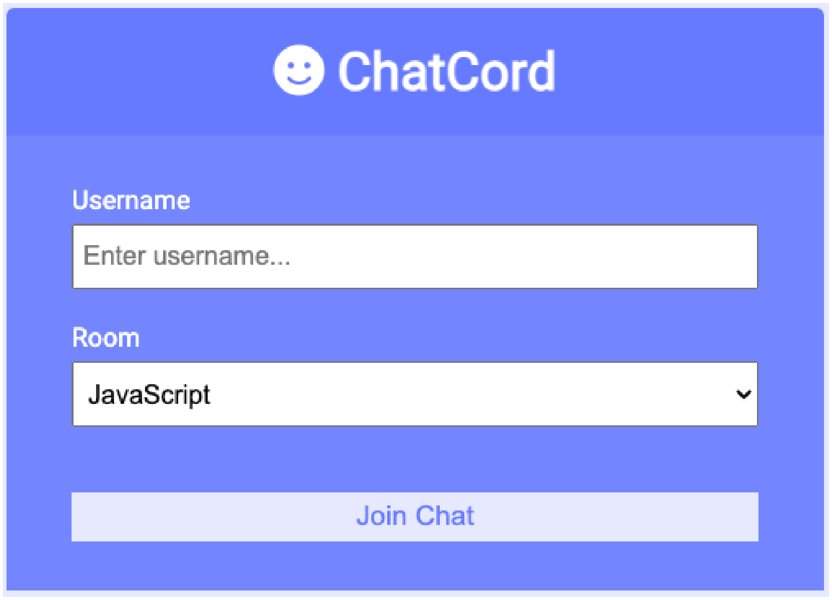
	Item	Price	Quantity	Total
	Mount of Olive's Book	$14.99	x1	$14.99
	Project Source Code	$19.99	x1	$19.99

Figure 22. Order list. Source: author

In this section user can review his/her order list and he can increase quantity of his/her orders, once user increase product quantity total price list will increase automatically.

As a second project real chat web application was chosen and implemented by using JavaScript frameworks such as node.js and library called socket.io to able to create real time chat room. In this project we focused on how node.js can handle with requests and as our application does not required any database and security tools as application did not deploy it into the server that's why we did not use such a tool. But comparing Django, Node.js does not come up with his database or security tools in order to implement such a tool to the project we have to take several extra steps to achieve what we want.

In this section we will take a look what we achieved as a result, source codes were already provided in previous sections now let's check final view of second web application.


ChatCord

Username

Enter username...

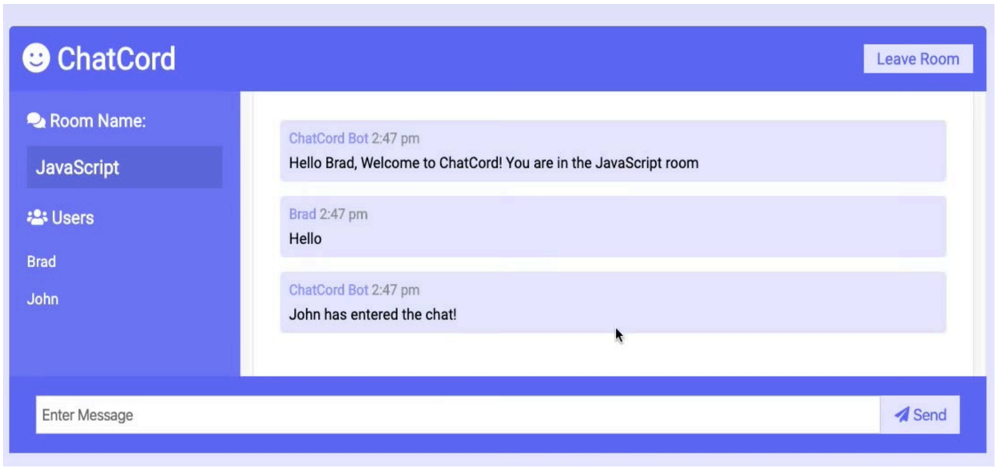
Room

JavaScript

Join Chat

Figure 23. Main page of Node.js application. Source: author

In this page user has to type his or her name and choose the room which user would like to connect, there is 6 chat rooms (JavaScript, Python, PHP, C#, Ruby, Java). Let's say user connected to the JavaScript room and another user connected to Python room so both users cannot see messages of this chats, user able to see only his chat room messages.



ChatCord Bot 2:47 pm
Hello Brad, Welcome to ChatCord! You are in the JavaScript room

Brad 2:47 pm
Hello

ChatCord Bot 2:47 pm
John has entered the chat!

Figure 24. Chat room. Source: author

In this page you can see that two users connected to the same server and the user named John has just connected it 1st user Brad can see that John is connected to the chat room. So, they basically can chat with each other unless one of them will not leave from that chat room.

Finally, as you saw in figure-23 and figure-24 it was our final view of project. To develop this application, we used very basic Node.js tools as author was not familiar with JavaScript and Node.js before it challenged me a lot while developing application and it was my first project in this language. Once this project was finished, author learned how to create real time chat application by using Node.js and JavaScript.

During the implementation of the e-commerce project, the foremost important aspect of the Django was that most of the necessary details are covered in the documentation. Developers do not need to go out of the documentation for the standard implementation. However, it is not always the case when there is a need for a different approach when solving the same problem. For example, if one has to modify the view to work differently than it might require much work. This requires much investigation and reading the source code. The framework itself comes with wide ranges of features and supports. Hence it is inevitable that documentation will also be vast. Nevertheless, developers who have some experience with Python programming language will likely learn the framework faster than developers coming from other languages. On the other hand, it has a broader community, and one can get help from the fellow developers very quickly. At the time of writing, there are 227,839 Django tagged questions posted on stackoverflow.com, which is one of the popular platforms for tech-related questions and answers.

6 Conclusion

The objective of the thesis was to compare the Python and JavaScript web development frameworks. The comparison was made while developing a basic web application from each of the frameworks. Each framework has its advantages and disadvantages. Both, the app tries to satisfy the same goal with a different approach because of the framework's features and limitations.

Every web application is made to solve problems with a specific business rationale. These commerce motives might come from a little start-up to a huge organization. Ordinarily, products are the aspects of the commerce motive. At times the product must be built and transported and evaluated rapidly and vice versa. Depending upon the business choice, and the available assets, one can select the appropriate framework. Both frameworks are developed and production ready, as well as each has their unique and standard features.

One of the most accomplishments of this project was that the comparative study helped to understand both frameworks. When tested through different approaches to solve the problems, it was found that both frameworks have their advantages and disadvantages. Both frameworks appear to be reasonable options if they take into utilize when solving business issues. There are also limitations to think about as not all aspects of the framework have been covered. Based on the study, it is obvious that Django can be the most excellent fit for large-scale projects with the taken a toll of the learning curve. On the other hand, Node.js is most fitting for applications that have a broad number of clients. There could be two circumstances to utilize this framework. Firstly, the processing time for each client is more. Secondly, the strategy utilizes outstandingly few CPU cycles. When each plan takes a sensible sum of time, you wouldn't need to misuse the string by keeping it on hold. Instep, you'll utilize that string to handle other requests until the past request's response is ready. One such case can be an informing application. Expect you're building an application where clients send messages to each other. You can't be beyond any doubt when a client sends a message, and unnecessary to say that a client wouldn't send messages 24/7. In such cases, within the occasion that you allot a string to each client, they are misused when the client isn't sending a message.

Lastly, considering Node.js vs Django, Django is approximately your choice and involvement, whereas Node.js is for proficient web designers who ought to construct real-world apps. Django is built for highly scalable apps, but one would utilize it when python is their uncommon ability.

7 References

1. BERDONOSOV, Victor; ZHIVOTOVA, Alena; SYCHEVA, Tatiana. TRIZ evolution of the object-oriented programming languages. *Procedia engineering*, 2015, 131: 333-342.
2. History of programming languages [online]. [cit. 2020-5-23]. Available from: https://cs.brown.edu/~adf/programming_languages.html
3. *History of C++ [*online]. [cit. 2020-5-23]. Available from: http://www.cplusplus.com/info/history/
4. Aillende, T. 2020 *High and low level programming* [online] [2020-11-29] Available from: https://trustonailende.com/programming-languages/
5. LUTZ, Mark. Programming python. " O'Reilly Media, Inc.", 2001.
6. PEDREGOSA, Fabian, et al. Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 2011, 12: 2825-2830.
7. MILLMAN, K. Jarrod; AIVAZIS, Michael. Python for scientists and engineers. Computing in Science & Engineering, 2011, 13.2: 9-12.
8. COCK, Peter JA, et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics, 2009, 25.11: 1422-1423.
9. GOODMAN, Danny. *JavaScript bible*. John Wiley & Sons, 2007.
10. DAYLEY, Brad. *Node. js, MongoDB, and AngularJS web development*. Addison-Wesley Professional, 2014.
11. WILLIAMSON, Ken. *Learning AngularJS: a guide to AngularJS development*. " O'Reilly Media, Inc.", 2015.
12. MCCUNE, Robert Ryan. Node. js paradigms and benchmarks. *Striegel, Grad Os F*, 2011, 11: 86.
13. PLEKHANOVA, Julia. Evaluating web development frameworks: Django, Ruby on Rails and CakePHP. *Institute for Business and Information Technology*, 2009.
14. VAN ROSSUM, Guido, et al. Python programming language. In: *USENIX annual technical conference*. 2007. p. 36.
15. *What is the PyPl index*? [online] [cit. 2020-12-2]. Available from: https://pypl.github.io/PYPL.html
16. FORCIER, Jeff; BISSEX, Paul; CHUN, Wesley J. *Python web development with Django*. Addison-Wesley Professional, 2008.
17. RAVINDRAN, Arun. *Django Design Patterns and Best Practices*. Packt Publishing Ltd, 2015.
18. DAUZON, Samuel; BENDORAITIS, Aidas; RAVINDRAN, Arun. *Django: Web Development with Python*. Packt Publishing Ltd, 2016.
19. MIKKONEN, Tommi; TAIVALSAARI, Antero. Using JavaScript as a real programming language. 2007.
20. ELLIOTT, Eric. *Programming JavaScript applications: Robust web architecture with node, HTML5, and modern JS libraries*. " O'Reilly Media, Inc.", 2014.
21. JS, Node; JS, NODE. Node. js. *Tradução de: SILVA, AG Disponível em*, 2020.
22. KUMAR, Anurag; SINGH, Ravi Kumar. Comparative Analysis of AngularJS and ReactJS. *International Journal of Latest Trends in Engineering and Technology*, 2016, 7.4: 225-227.
23. RAWAT, Prateek; MAHAJAN, Archana N. ReactJS: A Modern Web Development Framework.
24. CHAFFER, Jonathan; SWEDBERG, Karl. *Learning jQuery*. Packt Publishing Ltd, 2011.
25. OTERO, Cesar; LARSEN, Rob. *Professional JQuery*. John Wiley & Sons, 2012.
26. Node.js "Hello World!" source code [online] [cit. 2021-1-15]. Available from: https://nodejs.org/en/docs/guides/getting-started-guide/
27. CRAWFORD, Tyler; HUSSAIN, Tauqeer. A comparison of server side scripting technologies. In: *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2017. p. 69-76.
28. Primary code responsible for creating "Hello, World!" in Django [online] [cit. 2021-1-15]. Available from: https://djangoforbeginners.com/hello-world/
29. Django security configuration [online] [cit. 2021-1-18] Available from: https://docs.djangoproject.com/en/3.1/topics/security/
30. Node.js installation [online] [cit. 2021-1-18] Available from: https://nodejs.org/en/
31. CHHETRI, Nimesh. A Comparative Analysis of Node. js (Server-Side JavaScript). 2016.