A black text on a white background

Description automatically generated

**Exploring Web Development with Python: A Comprehensive Study with Applications (Blog) Built in Flask and Django**

Julia Gongala

B00402569

School of Computing, Engineering

and Physical Sciences

BSc (Honours) Web and Mobile Development

University of the West of Scotland

Supervisor: Tony Gurney

Moderator: Pablo Salva Garcia

# Declaration

This dissertation is submitted in partial fulfillment of the requirements for the degree of Web and Mobile Development (Honours) in the University of the West of Scotland.

I declare that this dissertation embodies the results of my own work and that it has been composed by myself. Following normal academic conventions, I have made due acknowledgement to the work of others.

Name: Julia Krystyna Gongala

Signature: J. Gongala

Date: 27th March 2024

**COMPUTING HONOURS PROJECT SPECIFICATION FORM**

*(Electronic copy available on the Aula Computing Hons Project Site)*

**Project Title:** "Exploring Web Development with Python: A Comprehensive Study with Applications (Blog) Built in Flask and Django."

**Student:** Julia Gongala  **Banner ID:** B00402569

**Programme of Study:** BSc (Hons) in Web and Mobile Development

**Supervisor:** Tony Gurney

**Moderator:** Pablo Salva Garcia

**Outline of Project:** *(a few brief paragraphs)*

Web development is a cornerstone of the digital age, driving the way we interact with information and services on the internet. In this dynamic landscape, Python, renowned for its versatility and user-friendly syntax, has emerged as a powerhouse for web application development. This dissertation embarks on an exciting journey, offering a comprehensive study enriched with hands-on applications. Our focal point is the creation of a fully functional blog website, a journey we undertake using two prominent Python web frameworks: Flask and Django.

Python's selection as the centrepiece of this exploration is not arbitrary but rooted in its growing popularity among developers. Python's elegant and readable syntax, coupled with a vast ecosystem of libraries and frameworks, has propelled it into the forefront of web development. Its seamless integration with web frameworks makes it an ideal choice for building robust and feature-rich web applications. Flask and Django, two noteworthy representatives of Python's web development prowess, each bring their unique strengths to the table, making them ideal subjects for this comprehensive study.

Our voyage commences with a brief introduction to the expansive realm of web development and Python's pivotal role within it. We set the stage for a deeper dive into Flask and Django, shedding light on their significance in the web development landscape.

As we navigate this study, we combine theoretical insights with hands-on practice. Our primary objective is to construct a functional blog website using both Flask and Django, unravelling their capabilities and dissecting their inner workings. This project-based approach ensures that we not only grasp the theoretical concepts but also gain practical experience in creating web applications. We explore the steps required to set up development environments, configure databases, implement user authentication, and craft user-friendly interfaces.

Furthermore, we undertake a comparative analysis of Flask and Django, offering a nuanced understanding of their strengths and limitations. This analysis empowers developers to make informed decisions when selecting the most suitable framework for their specific projects. It serves as a testament to Python's adaptability, showcasing how it caters to diverse web development needs.

Deployment, testing, and debugging, crucial facets of web development, receive their due attention in this exploration. We explore various deployment options for Flask and Django applications, ensuring that our web projects transcend the realm of local development environments and reach a global online audience.

In addition, we address the critical issue of web application security, examining common vulnerabilities and emphasizing best practices for mitigation. This discussion underscores the significance of secure coding practices within the context of web development with Python.

In conclusion, Exploring Web Development with Python: A Comprehensive Study with Applications (Blog) Built in Flask and Django embarks on a captivating journey through the expansive world of web development with Python. It underscores Python's pivotal role in this domain and showcases the capabilities of Flask and Django through the creation of a fully functional blog website. By the culmination of this exploration, readers will possess not only the knowledge but also the practical skills to embark on their web development ventures confidently or make informed choices when selecting the right framework for their projects.

**A Passable Project will:**

* Introduction to Web Development and Python
* Flask and Django Introduction
* Building basic Blog with Flask and Django where user will be able to add, delete, edit and see posts to compare both frameworks and select better for advanced blog.
* Comparative Analysis for both frameworks showing their features.
* Deployment and Testing of the application

**A First Class Project will:**

* Framework Comparison and Analysis. A detailed and insightful comparison between Flask and Django, highlighting not only their strengths and weaknesses but also their suitability for specific use cases – Blog using literature review and blog development
* Advanced Blog Development. A feature-rich and highly polished blog application built using selected framework, that has been chosen through comprehensive study to justify all arguments and decision, demonstrating not only the core functionality (CRUD) but also advanced features like user profiles, or uploading images. Advanced blog will incorporate modern practices and techniques.
* Demonstration Knowledge. A clear demonstration of knowledge in modern web development, Python, Flask, Django, and related popular technologies.
* Using References and Citations. Accurate and extensive citation and referencing of all sources, demonstrating a deep understanding of the academic context.

**Reading List:**

* *"Django for Beginners"* by William S. Vincen
* *"Django for APIs"* by William S. Vincent
* *"Two Scoops of Django 3.x"* by Daniel Roy Greenfeld and Audrey Roy Greenfeld
* *"Flask Web Development"* by Miguel Grinberg
* *"Flask By Example"* by Gareth Dwyer
* *"Building RESTful Python Web Services"* by Gaston C. Hillar
* *"Python Web Development with Django"* by Jeff Forcier, Paul Bissex, and Wesley Chun
* *"Test-Driven Development with Python"* by Harry J.W. Percival
* Django Official Documentation
* Flask Official Documentation

**Resources Required:** *(hardware/software/other)*

* Personal workstation with Visual Studio Code and Python installed
* Git and GitHub for version control and collaboration.
* Web browser for research and testing.
* Hosting platform (e.g., Heroku, AWS, or a VPS) for deployment.
* Testing tools

**Marking Scheme: Marks**

Introduction 5%

Literature Review 15%

Development 30%

Comparative Analysis 30%

Testing 10%

Conclusion 10%

**AGREED:**

**Student Supervisor Moderator**

**Name:** Julia Gongala  **Name:** Tony Gurney **Name:** Pablo Salva Garcia

**IMPORTANT:**

1. ***By agreeing to this form all parties are confirming that the proposed Hons Project will include the student undertaking practical work of some sort using computing technology / IT, most frequently achieved by the creation of an artefact as the focus for covering all or part of an implementation life-cycle.***
2. ***By agreeing to this form all parties are confirming that any potential ethical issues have been considered and if human participants are involved in the proposed Hons Project then ethical approval will be sought through approved mechanisms of the School of CEPS Ethics Committee.***

Table of Contents

[Declaration 2](#_Toc158817781)

[Table of Figures 7](#_Toc158817782)

[Acknowledgements 8](#_Toc158817783)

[Abstract 9](#_Toc158817784)

[1. Introduction 10](#_Toc158817785)

[2. Background 13](#_Toc158817786)

[2.1 Background of web application development 13](#_Toc158817787)

[2.1.1 Front – End Development 13](#_Toc158817788)

[2.1.2 Back – End Development 13](#_Toc158817789)

[2.1.3 Full – Stack Development 13](#_Toc158817790)

[2.1.4 Responsive Design 13](#_Toc158817791)

[2.1.5 Security 13](#_Toc158817792)

[2.1.6 Front – End Development 13](#_Toc158817793)

[References 14](#_Toc158817794)

# Table of Figures

[Figure 1.1 Growth of major programming languages (StackOverflow blog, 6/09/2017) 11](#_Toc158814363)

# Acknowledgements

I would like to express my heartfelt gratitude to my parents for their unwavering support and encouragement in all aspects of my work. A special thanks to my boyfriend, David, for dedicating his time, providing continuous support, and engaging in long, meaningful discussions about my ideas and challenges. I am also grateful to my supervisor, Tony, whose previous guidance in Python language has been instrumental in shaping my journey in this research topic. Lastly, a big thank you to my moderator, Pablo, for his ongoing support.

# Abstract

In the realm of web development, Python has emerged as a powerhouse, valued for its simplicity, readability, and robust ecosystem. This research undertakes a thorough exploration of Python's pivotal role in web development, focusing on its frameworks, libraries, and practical applications. As scalable and efficient web solutions become increasingly sought after by businesses, a nuanced understanding of Python becomes imperative.

The intricacies of Python's contributions to web development are delved into, shedding light on its flagship frameworks, namely Django and Flask. Through real-world implementations, the aim is to showcase how these frameworks empower developers to create robust and maintainable web applications. The examination extends beyond the server-side, exploring Python's seamless integration with front-end technologies, providing a holistic perspective on its position in the dynamic landscape of contemporary web development.

The proposed technologies and features are meticulously tailored to address the project's goals. Django, with its batteries-included philosophy, offers a comprehensive and structured approach to building web applications, ensuring scalability and maintainability. Flask, on the other hand, provides a lightweight and flexible framework, ideal for smaller projects or when customization is paramount.

Furthermore, the research emphasizes the integration of Python with front-end technologies. This integration ensures a cohesive development experience, allowing for the creation of interactive and responsive user interfaces.

In practical terms, the study will provide developers and businesses with actionable insights into leveraging Python's strengths for optimal web development outcomes. Whether through the powerful abstractions of Django or the lightweight agility of Flask, Python's versatility is harnessed to meet the diverse needs of modern web applications. As businesses navigate the complexities of the digital landscape, understanding the intricacies of Python in web development becomes not only beneficial but essential for achieving success in delivering scalable, efficient, and user-friendly web solutions.

# Introduction

The evolution of programming languages has indeed been marked by a captivating journey, reflecting the dynamic nature of technology and the ever-changing landscape of the computing industry. As time progresses, the narrative of programming languages becomes more intricate, characterized by the emergence of new paradigms, tools, and methodologies.

In the 1960s and 1970s, the advent of high-level programming languages like Fortran, Lisp, and COBOL was witnessed, aiming to enhance productivity and portability. Scientific and engineering computations were the focus of Fortran, while Lisp pioneered the concept of symbolic expressions and recursion, and COBOL targeted business applications. The groundwork laid by these languages set the stage for subsequent innovations and the development of more user-friendly programming languages.

The 1980s and 1990s ushered in the era of personal computing, giving rise to languages like C, C++, and Java. System programming saw C as the language of choice due to its low-level capabilities, while C++ introduced object-oriented programming concepts, adding a layer of abstraction. Java, designed for platform independence, gained popularity for developing applications that could run on any device with a Java Virtual Machine (JVM).

In the late 1990s and early 2000s, dynamic languages such as Python and Ruby emerged, emphasizing simplicity and readability. Widespread acceptance of Python was driven by its clear syntax and versatility. Simultaneously, JavaScript rose to prominence as the scripting language for web browsers, enabling dynamic and interactive web pages.

As the internet continued to evolve, the demand for web development languages and frameworks increased. PHP, alongside Python and Ruby, played a crucial role in server-side scripting, powering dynamic web applications. The shift towards more responsive and interactive web experiences was marked by the introduction of AJAX (Asynchronous JavaScript and XML).

The rise of mobile computing in the 2010s led to the prominence of languages like Swift for iOS and Kotlin for Android development. New avenues for building mobile applications efficiently were provided by the development of cross-platform frameworks such as React Native and Flutter. (Karunanayake, Feb 24, 2023)

A critical aspect of modern programming revolves around the indispensable role of frameworks. These frameworks serve as pivotal tools, furnishing developers with pre-built structures and utilities that significantly streamline the development process. Distinguished by their unique syntax and architecture, these frameworks provide a higher level of abstraction. This, in turn, empowers programmers to direct their focus towards crafting application-specific logic, alleviating the need to grapple with intricate low-level details.

While the adoption of various frameworks may initially present a learning curve, the adaptability of programmers becomes paramount. It is through this adaptability that developers can harness the distinctive advantages that each framework brings to the table. Embracing the diversity of frameworks is not just a necessity; it is a strategic approach that enables developers to optimize their workflow, enhance productivity, and build robust and efficient software solutions tailored to the specific needs of their projects. As the programming landscape continues to evolve, the ability to navigate through different frameworks and judiciously select the most suitable one for a given task emerges as a valuable skill for developers aiming to stay at the forefront of innovation.

Collaboration among programmers, particularly within diverse teams, presents a multifaceted challenge that hinges on clear communication and unified practices. In this intricate landscape, the strategic choice of frameworks becomes pivotal, serving as a cornerstone for harmonizing team expertise and aligning with project requirements. The aim is to establish an environment where collaboration can flourish seamlessly.

The significance of framework selection is underscored by its impact on the team's ability to work cohesively. When frameworks are chosen with consideration for the team's proficiency and project specifications, it lays the groundwork for an efficient collaboration process. A well-aligned framework not only facilitates smoother communication but also enables developers to leverage their expertise, fostering a sense of shared understanding within the team.

However, the integration of disparate frameworks within a project introduces an additional layer of complexity. This complexity necessitates the establishment of seamless communication channels and a shared comprehension of the chosen frameworks among team members. This shared understanding becomes paramount to ensure a cohesive and productive development environment, where team members can collaborate without unnecessary friction or misunderstandings.

In essence, successful collaboration in programming teams is not only contingent on effective communication but also on the thoughtful selection and integration of frameworks. By aligning frameworks with team expertise and project needs, and fostering a shared understanding, development teams can navigate the challenges of diverse collaborations more effectively, resulting in streamlined workflows and successful project outcomes.

The intricate interplay of challenges and solutions within the programming landscape serves as a compelling backdrop for thesis, titled 'Exploring Web Development with Python: A Comprehensive Study with Applications (Blog) Built in Flask and Django.' This research delves into the dynamic realm of web development, placing a particular emphasis on Python, a versatile programming language that has witnessed remarkable growth, as illustrated in Figure 1.1.

A graph of a graph showing different colored lines

Description automatically generated

Figure 1.1 Growth of major programming languages (StackOverflow, 6/09/2017)

As depicted in Figure 1.1, Python has experienced substantial popularity in recent years, emerging as a leading choice among developers for various applications. The thesis aims to unravel the factors contributing to Python's ascendancy in the programming landscape, exploring its versatility and adaptability in the context of web development.

Within the scope of this research, our focus is directed towards two prominent web frameworks Flask and Django. An in-depth study is conducted to elucidate their distinct features, strengths, and applications. Practical examples and the undertaking of blog development serve the purpose of showcasing how these frameworks contribute to the creation of dynamic and robust web applications.

The strategic choice of Flask, renowned for its simplicity and flexibility, alongside Django, celebrated for its batteries-included approach, sets the stage for a comparative analysis. This approach not only brings to light the unique attributes of each framework but also provides insights into the trade-offs involved in selecting a framework based on specific project requirements. Through hands-on exploration and real-world application, the research endeavours to offer a nuanced understanding of how Flask and Django cater to different aspects of web development, enabling developers to make informed choices aligned with their project goals.

Moreover, the challenges and benefits associated with integrating these frameworks into collaborative web development projects are delved into by the study. Emphasis is placed on the importance of clear communication and shared understanding among team members, highlighting the necessity for cohesive collaboration when working with diverse frameworks. Addressing the collaborative aspects of web development, the research aims to contribute practical insights that extend beyond individual proficiency with the frameworks, recognizing the significance of teamwork in achieving successful project outcomes.

In essence, the aspiration of this study is to be a valuable resource for programmers and developers navigating the evolving landscape of web development using Python and its associated frameworks. Through a comprehensive exploration of Flask and Django, coupled with practical examples and a focus on collaborative dynamics, professionals are equipped with the knowledge and perspectives needed to thrive in the dynamic and ever-changing world of web development.

# Background

## Background of web application development

Web application development is a multifaceted process that involves creating and maintaining software applications accessible through web browsers. It encompasses both front-end and back-end components, each playing a crucial role in delivering a seamless and interactive user experience.

### Front – End Development

The front end, often referred to as the client-side, constitutes the user interface (UI) and user experience (UX) layer of a web application. It is the part of the application that users directly interact with, encompassing elements like buttons, forms, and visual components. Figure 2 illustrates the communication flow between the front end and back end, showcasing how user interactions are processed and data is exchanged.

A diagram of a cloud computing system

Description automatically generated

Figure 2.1: How Front-end development works (frontendmasters.com, n/d)

Front-end development primarily relies on three fundamental technologies: HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), and JavaScript. HTML serves as the backbone, structuring the content and defining the page's semantic structure. CSS comes into play for styling, determining the layout, colours, and overall visual presentation of the web pages. JavaScript, on the other hand, introduces interactivity, enabling developers to create dynamic and responsive user interfaces.

The integration of these technologies allows front-end developers to craft visually appealing and user-friendly experiences. HTML provides the skeletal structure, CSS adds the aesthetic layer, and JavaScript injects behaviour and responsiveness, creating a cohesive and engaging user interface. (Robbins, 2018)

The evolution of front-end development has seen the emergence of powerful frameworks such as React, Angular, and Vue.js. These frameworks provide a structured and organized approach to building web applications, facilitating the development process and enhancing application performance. React, developed by Facebook, utilizes a component-based architecture, Angular, maintained by Google, is a comprehensive framework, and Vue.js, known for its simplicity, offers a flexible and adaptable solution. These frameworks help manage the complexity of large-scale applications, promote code reusability, and ensure efficient rendering of UI components.

Moreover, front-end development has been further streamlined by the use of JavaScript libraries and tools. Libraries like jQuery simplify DOM manipulation, making it easier for developers to interact with the document object model. Task runners and bundlers like Webpack assist in optimizing and organizing code, improving performance and maintainability. (Lolugu, 2023)

In conclusion, front-end development is a crucial aspect of web application creation, focusing on delivering seamless and visually appealing user experiences. The utilization of HTML, CSS, and JavaScript, along with modern frameworks and libraries, empowers developers to build interactive and dynamic interfaces, ultimately enhancing the overall usability and success of web applications.

### Back – End Development

### Full – Stack Development

### Responsive Design

### Security

### Front – End Development

* + 1. Integration and Collaboration

# References

Karunanayake, Y., Feb 24, 2023. *Simple History of Programming Languages.* [Online]   
Available at: https://medium.com/@yureshcs/simple-history-of-programming-languages-db299d16b8e4 [Accessed 02 01 2024].

Lolugu, K. V., 2023. *Let’s Code the User Experience.* [Online]   
Available at: https://bootcamp.uxdesign.cc/lets-code-the-user-experience-c47c72f50700  
[Accessed 30 10 2023].

Robbins, J. N., 2018. *Learning Web Design.* 5 ed. s.l.:O'Reilly Media, Inc, USA.