COS60010 Technology Enquiry Project
Semester 2, 2021

Deliverable 1: Individual Research Report

Individual Assignment

Course Code: 60004
Convenor: Dr. Charlotte Pierce
Tutor: Dr. Eureka Priyadarshanni
Student ID: 103527269
Student Name: Lucas Qin
Word Count: 2716 (including reference)
Date: 25/08/2021

Content

1. Introduction

Most of the web and software today consist of two parts, client side and server side. Client side is also called front end, which could be seen and accessed by users and interact with users. However, the client side can't work without the server side. The server side is not visible and accessed by the user, but it stores almost all contents of a web and software and enables all the functions of client side. This article will focus on what a server is, how a server works and what technologies are involved in building and maintaining a server.

2. Backend server

2.1 What is a server?

Server is the computer that is behind the front end, server store, transfer and organize all the data for the front end[1]. Server is the brain of a web application and enables different functions. The back end of a web application consists mostly of APIs, databases, and servers. To get started, API stands for application programming interfaces which is the application that connects frontend and servers. API defines what type of requests that it can handle, and what type of response will be sent to the client side[2]. One API is capable of dealing with requests from multiple client sides. When the client side sends a request to retrieve data via URL (uniform resource identifier), the request will be received by API, then API transfers the request to relevant servers, and servers retrieve the requested data then send back to API in a specific format. API interprets data format into the format that either client side or server side can understand and implement[2]. During this process, API not only decides which server or database to connect to, but also manages authorization credentials which add an extra layer of protection to servers and databases.

There are many types of servers, each of them is a server with different purposes. Below is some most common example of server types:

1. Email server includes Incoming Mail (IMAP) Server, Incoming (POP) Server and Outgoing Mail (SMTP) Server[3]. IMAP or POP servers are used to download messages from the user's email client while SMTP is used for sending messages.

2. FTP servers use File Transfer Protocol to transfer one file from one place to another. FTP allows remote access via the server's in-app FTP function or specific FTP program.

3. Identity server is very common in medium and large size companies, an Identify server stores all user's register information.

4. Proxy servers allow users to have an extra protection between client side and its servers, proxy servers can hide the user's real IP and instead send website servers a pseudo-IP to avoid any tracking of the user's IP.

5. DNS servers store a database of public IP addresses. When a user types in a website, the computer needs to connect to its server, to find out which server is requested, DNS server translates the user's type-in website into IP address.

2.2 How do servers work with the client side?

A client-side uses HTTP to communicate with its backend, every time a user clicks a new page or submit a form, the browser will send a HTTP request to servers. There are few common types of requests:

1. GET: for retrieving specific data that is identified by URL. [4]
2. POST: for creating new data.
3. HEAD: for obtaining metainformation of an entity. [4]
4. PUT: for updating an existing resource.
5. DELETE: for deleting an existing resource.

To understand how a URL could be used in requesting specific information, it is important to know how the URL works.

Fig 2.1 URL components[6]

URL is like an address to the specific data and resources, within a URL there are protocol, domain, path, parameter and anchor[5]. The basic idea is that after user send out a URL, the URL is received by Domain Name Server (DNS), DNS store the database of public IP address and then transform URL into IPv6 or Ipv4 form which is the real address of a website and complete comprise of number. When a website has the real address of what the user requests, the website sends out a HTTP request to the server, and all the transmission is via internet protocol TCP/IP. If the server approved the request, HTTP would return a '200 ok' back to client side, if not, there also will be a '404 – not found' or other codes to tell client side what happened. After the server

approves the request, data in pieces retrieved from different parts of the database and reassembled by server in a specific format, then data will be sent to the client side.

2.3 What are the most commonly used programming languages for servers?

While client-side use HTML, CSS and JavaScript as programming languages, server side is using different programming languages, because different languages serve different purposes. Among these languages, JavaScript is the only language that could be used in both frontend and backend. Choosing different languages will result in different software scalability, compatibility, development life cycle and software responsive performance.

Below are some of the most popular programming languages to be used by backend developer:

2.3.1 High-level programming languages

High-level programming languages use plain English words that humans could read and understand, but a compiler and assembler are needed to translate those code into machine code that hardware could understand.

Table 3.1.1 High-level server languages compare

Type	Languages	Pros	Pons

Object Oriented Language	JavaScript	1. JavaScript can be used in both frontend and backend. This could reduce development cost and increase development efficiency[6]. 2. JS supports most of the middleware in backend development. 3. JavaScript is open-source and has great community support[6]. 4. JavaScript is capable of running on client-side hence could reduce server load[12].	1. Event-driven functions are complicated[6]. 2. JavaScript uses MySQL for database service[6]. 3. Node.js's API changes frequently and is backward-incompatible. 4. JavaScript does not support multi-threaded programming.

Object Oriented Language	PHP	1. It's easy for beginners to learn and implement. 2. PHP is an open-source language and supported by many developers globally. 3. PHP supports all operating systems, browsers and servers[6]. 4. PHP can be connected to over 20 databases. 5. Applications coded with PHP load faster than many other languages.	1. PHP is facing a declining popularity[9]. 2. PHP is not suitable for huge applications. 3. PHP is an open-source language, and anyone can access its source code, which means the website using PHP is not secure[11].

Object Oriented Language	Java	1. Java allows the server side to run several instances at the same time[6]. 2. Java's syntax is similar to plain English hence easy to understand[6]. 3. It's open-source languages that are supported by a large community of developers[6]. 4. Java has enormous libraries[6]. 5. Java has security features for backend development.	1. It's expensive in development and implementation[6]. 2. Java does not have a garbage data collection feature[6]. 3. High memory usage compared to other languages[7]. 4. Slow start up for Java application[7]. 5. It's hard to integrate necessary backend features for developing an application.

Object Oriented Language	Python	1. Python is easy for beginners. 2. Large number of libraries for Python efficiency[6]. 3. Python is compatible with low-level languages like C++ and can be embedded in C++. 4. Intuitive syntax to prevent weird errors. 5. Python is a portable language[13]. 6. Python offers database interfaces for all major DBMS systems[8].	1. More testing and debugging processes than other languages[6]. 2. Python depends heavily on third-party libraries and frameworks[6]. 3. Most of the developers in the community focus on AI and machine learning. 4. Python is not memory efficient in mobile and client-side computing.

Object Oriented Language	Ruby	1. Ruby is similar to Python and Java in syntax[6] and easy to learn. 2. Ruby is supported by powerful third-party libraries. 3. Meta-programming feature[6]. 4. Ruby can use libraries for testing. 5. Ruby is reliable and quick in software development.	1. Ruby's runtime is slower than other languages[6]. 2. Ruby does not have many libraries[6]. 3. Hard to debug a program that uses Ruby.
Object Oriented Language	C#	1. C# supports cross-platform development[6]. 2. Programs coded in C# could run compatible on older versions of the system[6]. 3. C# use garbage data collection features to reduce error[6].	1. It can only be installed in the Windows system[6]. 2. C# can only use the .Net framework[6].

2.3.2 Low-level programming languages

Quite opposite to high-level programming languages, low-level languages are machine-oriented and understandable to machine but abstract to human, low-level languages are designed to work with hardware components and logic, hence less execution time and memory space are needed. Many high-level languages like Java and JavaScript can be embedded in low-level languages and work together.

Table 3.2.1 Low-level server languages compare

Type	Language	Pros	Cons
Procedural Oriented Language	C	1. C is a portable language. 2. C is a middle-level language that supports both high-level-languages and low-level-language in many features[14]. 3. As a procedural language, C is able to identify code structure and to fix issues in a specific series of code. 4. C offers dynamic memory allocation[14].	1. C does not check for coding errors after each line, instead, C shows all errors after all writing is done[14]. 2. C does not support the concept of OOP[14]. 3. C does not support the concept of construction and destruction[15].

Partly Object-Oriented Language	C++	1. C++ can work on all different operating systems. 2. C++ can be understandable by hardware without much translation by compiler and assembler. 3. C++ have powerful memory management.	1. No garbage data collection. 2. C++ do not have security features.

2.4 What are the frameworks for server programming?

Framework in backend is a combination of develop tools and libraries that enable developers to quickly establish necessary functions and build server-side structures, hence accelerating the whole project. Framework enables minimal coding and maintenance.

Table 4.1 Server-side framework compare

Framework	Pros	Cons	Language's support

Spring	1. Spring offers many API for different domains. 2. Spring is lightweight[20]. 3. It's a portable framework. 4. Spring has security features. 5. Banner can be customized[17].	1. Spring is difficult to learn[18]. 2. Spring is complex and requires a lot of knowledge of the system. 3. Lack of documentation. 4. Deep learning curve[20]. 5. Require a lot of XML in development[20].	Java

Ruby on Rails	1. Better for prototyping[16]. 2. Quick development. 3. Ruby on Rails provides some ready-made modules and plugins that could substantially reduce development time[17]. 4. It's open-source and free to use. 5. Great scalability means Ruby and Rails are suitable for huge applications[17].	1. Not suitable for huge applications[16]. 2. It does not have appropriate citations[16]. 3. Very slow run time speed. 4. Bad documentation on code and configuration. 5. Ruby on Rails is single threading.	Ruby

Express JS	1. Good flexibility. 2. Offer package for building API[16]. 3. Easy to learn for developers that use JavaScript. 4. Express JS integrated many of the node.js features which enables developers to significantly reduce coding and development time. 5. Express JS offers routing features[17]. 6. Offer easy scalability for medium-size applications.[19]	1. Node.js is a single threaded framework. 2. Express.js is built on the concept of middleware. 3. Do not have much library support. 4. Difficult to maintain code due to its asynchronous programming model[19].	Node.js

ASP.NET Core	1. It's a cross-platform framework. 2. ASP.NET Core offers features that enable minimal coding hence reduce development time and reduce cost[17]. 3. Less code enables less maintenance. 4. Better performance than other frameworks[17].	1. It's expensive, developers need to pay for many licenses like SQL server license, Windows server license, etc. 2. Do not have many securities choices. 3. Documentation is not great compared to other frameworks.	C#

Django	1. Great documentation for users[16]. 2. Better for a database-driven application that has many features. 3. Django provides many features in helping developers in web development. 4. Django has security features. 5. Django can be highly customized[18]. 6. Supports ORM.	1. Not suitable for small applications due to its high volume of plugins[18]. 2. Deep learning curve for developers[18]. 3. Django is monolithic[16]. 4. Developer needs to learn the whole system to use Django.	Python

Flask	1. Developers who use Python could find Flask easy to use. 2. Flask is easy to configure. 3. Flask uses modular code[17] which enables easy deployment and testing. 4. Offer debugger functions and servers[17].	1. Not suitable for huge applications. 2. Third party modules post potential risk to the project. 3. Flask can only process one request at a time. 4. Community support is weaker than Django.	Python
Laravel	1. Supported by a huge developer community on GitHub[16]. 2. Great documentation for users. 3. Offer authentication for using. 4. Laravel offers a simple API that works fluidly with SwiftMailer[17].	1. Laravel is a rather slow and new tool. 2. Some features like reverse routing are complex. 3. Developer community is not huge.	PHP

2.5 What environment to get started in building a server?

There are some local server building environments available for developers to build and test their web applications locally. In fact, one of the biggest advantages of building a local testing server is no frustration in transferring all types of files to the web server and doing the testing. In the local server, developers can make instant changes to see the effect. Besides, most of the environments combine HTTP server, database, developer tools together, making it convenient for beginners to deploy a functional server as soon as possible. Some recommendations are XAMPP, Wampserver,

AMPPS, Desktop Server, ERBuilder, MAMP and so on. Due to the word limitation, relevant recommendations' introduction and comparison have been placed in further information.

3. Summary

Based on the above demonstration and comparison, there are many technologies involved in server-side development, and each of them have different advantages and disadvantages. Choosing different technologies will result in different costs, learning curve, project time, maintenance difficulty and cost, etc. Most importantly, different technologies are suitable for different projects and require expertise in different systems.

4. Further Information

[1] Comparison of web frameworks - Wikipedia. *En.wikipedia.org*, 2021.
https://en.wikipedia.org/wiki/Comparison_of_web_frameworks.
[2] Advantages and Disadvantages of Django | Hacker Noon. *Hackernoon.com*, 2021.
https://hackernoon.com/advantages-and-disadvantages-of-django-499b1e20a2c5.
[3] Disadvantages of Flask. *Medium*, 2021. https://allwin-raju-12.medium.com/disadvantages-of-flask-95f1ff8f83bf.
[4] *Software-developer-india.com*, 2021.
https://www.software-developer-india.com/advantages-and-disadvantages-of-asp-net/.
[5] ASP.NET Core Advantages and Disadvantages | Redwerk. / *Redwerk*, 2021.
https://redwerk.com/blog/asp-net-core-pros-and-cons/.
[6] The Good and the Bad of Node.js Web App Development. *AltexSoft*, 2021.
https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-node-js-web-app-development/.
[7] Express.js Mobile App Development: pros and cons of Node.js framework. *Apiko* / *Learn*, 2021.
https://apiko.com/blog/express-mobile-app-development/.
[8] Client-Server Overview - Learn web development | MDN. *Developer.mozilla.org*, 2021.
https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Client-Server_overview.
[9] Hamilton, E. Advantages And Disadvantages Of PHP You Should know. *iTech Post*, 2021.
https://www.itechpost.com/articles/105069/20210320/advantages-and-disadvantages-of-php-you-should-know.htm.
[10] Server-side web frameworks - Learn web development | MDN. *Developer.mozilla.org*, 2021.
https://developer.mozilla.org/en-US/docs/learn/Server-side/First_steps/Web_frameworks.
[11] What are some alternatives to XMPP? - StackShare. *Stackshare*, 2021. https://stackshare.io/xmpp/alternatives.
[12] Communications, S. XMPP Alternatives and Similar Apps / Services | AlternativeTo. *AlternativeTo*, 2021.
https://alternativeto.net/software/xmpp/.

5. Reference

[1] MDN Web Docs. 2021. Introduction to the server side. Retrieved from

https://developer.mozilla.org/en-US/docs/learn/Server-side/First_steps/Introduction

[2] Andrew Parks. 2021. How do APIs work? Retrieved from https://tray.io/blog/how-do-apis-work

[3] Paessler AG. 2021. What is a server? Retrieved from https://www.paessler.com/it-explained/server

[4] Dan Connolly. 2004. 9 Method Definitions. Retrieved from https://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html

[5]MDN Web Docs. 2021. What is a URL? Retrieved from https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL

[6]Back4App. 2020. The best ten backend languages. Retrieved from https://blog.back4app.com/best-backend-language/

[7]Matthias Graf. 2020. Which Backend Language Should You Choose? Retrieved from https://medium.com/swlh/which-backend-language-should-you-choose-eb924902a9b3

[8]Aman Goel. 2021. Python vs PHP in 2021. Retrieved from https://hackr.io/blog/python-vs-php

[9]Alexander Roznovsky. 2021. WHY USE PHP? MAIN ADVANTAGES AND DISADVANTAGES. Retrieved from https://light-it.net/blog/why-use-php-main-advantages-and-disadvantages/

[10]SIMPLY TECHNOLOGIES. 2018. NODE.JS: PROS AND CONS OF THE SERVER-SIDE JAVASCRIPT. Retrieved from https://www.simplytechnologies.net/blog/2018/6/19/nodejs-pros-and-cons-of-server-side-javascript

[11Jenifer Hoo. 2018. Advantages and Disadvantages of PHP. Retrieved from]https://bigcheaphosting.com/advantages-and-disadvantages-of-php/

[12]Ian Deed. 2021. Pros and Cons of JavaScript Development. Retrieved from https://www.pangea.ai/javascript-resources/javascript-resources-best-practices/

[13]TechVidvan. 2021. Python advantages and disadvantages - Step in the right direction. Retrieved from https://techvidvan.com/tutorials/python-advantages-and-disadvantages/

[14]DataFlair. 2021. Advantages and disadvantages of C programming - Discover the secret of C. Retrieved from https://data-flair.training/blogs/advantages-and-disadvantages-of-c/

[15]techcrashcourse.com. What are the disadvantages and limitations of C programming languages. Retrieved from https://www.techcrashcourse.com/2015/11/Disadvantages-and-limitations-of-C-Programming-languages.html

[16]Monocubed. 2021. Detailed Comparison of Web Frameworks with Pros and Cons. Retrieved from https://www.monocubed.com/web-development-framework-comparison/

[17]Back4App. 2020. The best ten backend frameworks.. Retrieved from https://blog.back4app.com/backend-frameworks/

[18]Roger James. 2019. Best Web Development Frameworks Comparison. Retrieved from https://www.pixelcrayons.com/blog/best-web-development-frameworks-comparison/

[19]Tejas Kaneriya. 2020. Advantages & Disadvantages of Node.js: Why use Node.js? Retrieved from https://www.simform.com/blog/nodejs-advantages-disadvantages/

[20]www.javatpoint.com. 2021. Java Spring Prons and Cons. Retrieved from https://www.javatpoint.com/java-spring-pros-and-cons