PANORBIT®

Different Types of Backend Technologies

CONTENTS

Introduction

We visit a lot of websites these days. Some are well-designed but fall behind in intended functionalities. On the other hand, some websites are just basic in design with an excellent functionality. And yes, there are websites that are good in both ways. Don’t even think of ones that are bad in both departments.

The appreciation for well-designed part of a website or a web application goes to frontend developers while the credits for a well-functioning website or application goes to the backend developers. Still confused on frontend vs backend? Let us dive deep into the web development process to give you a broader view.

Web Development Process

To understand what goes on behind developing a website, let us discuss the steps involved in web development life cycle.

The steps can be broadly categorised as

01
Information Gathering

02
Planning

03
Design

04
Build/Development

05
Testing

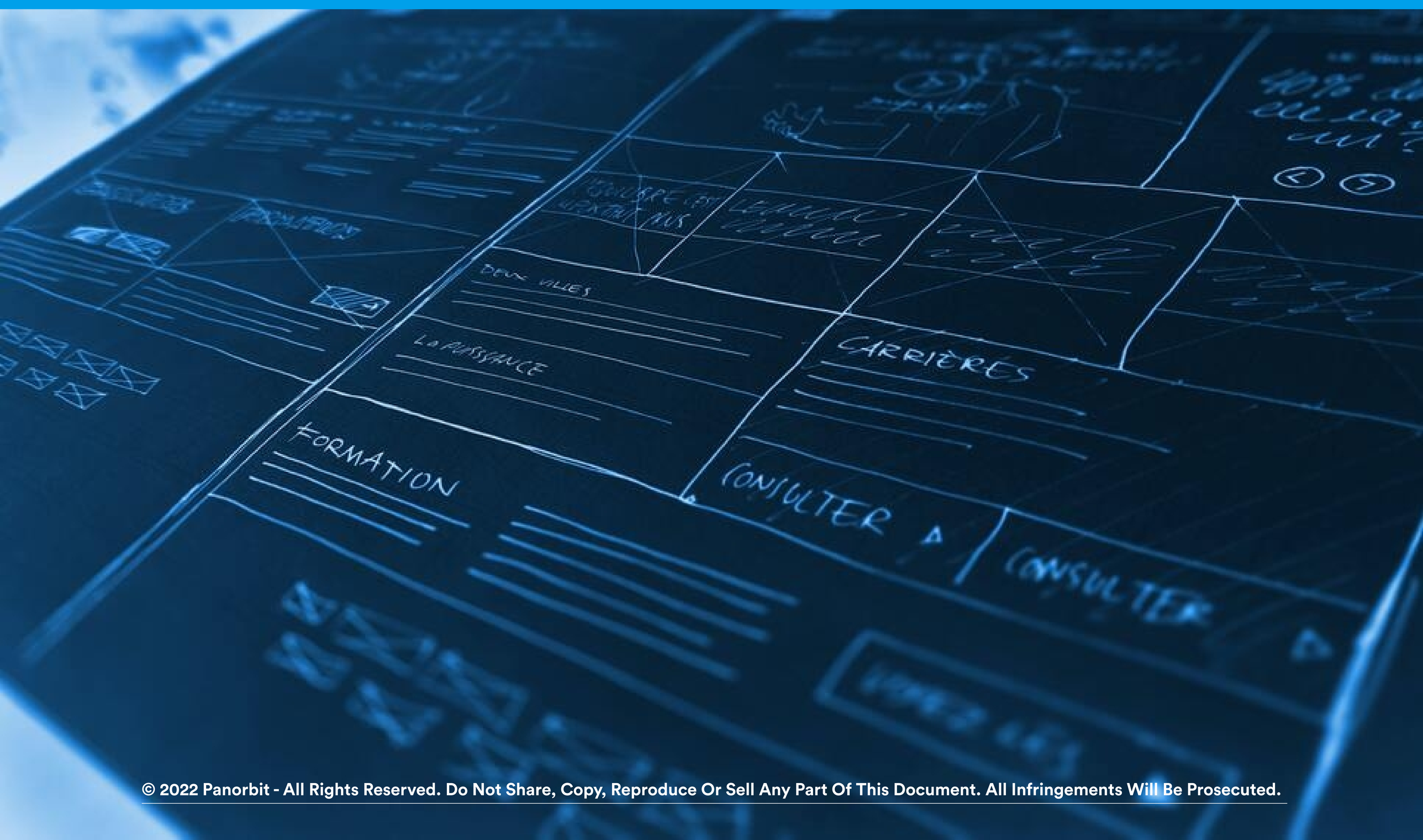
06
Launch

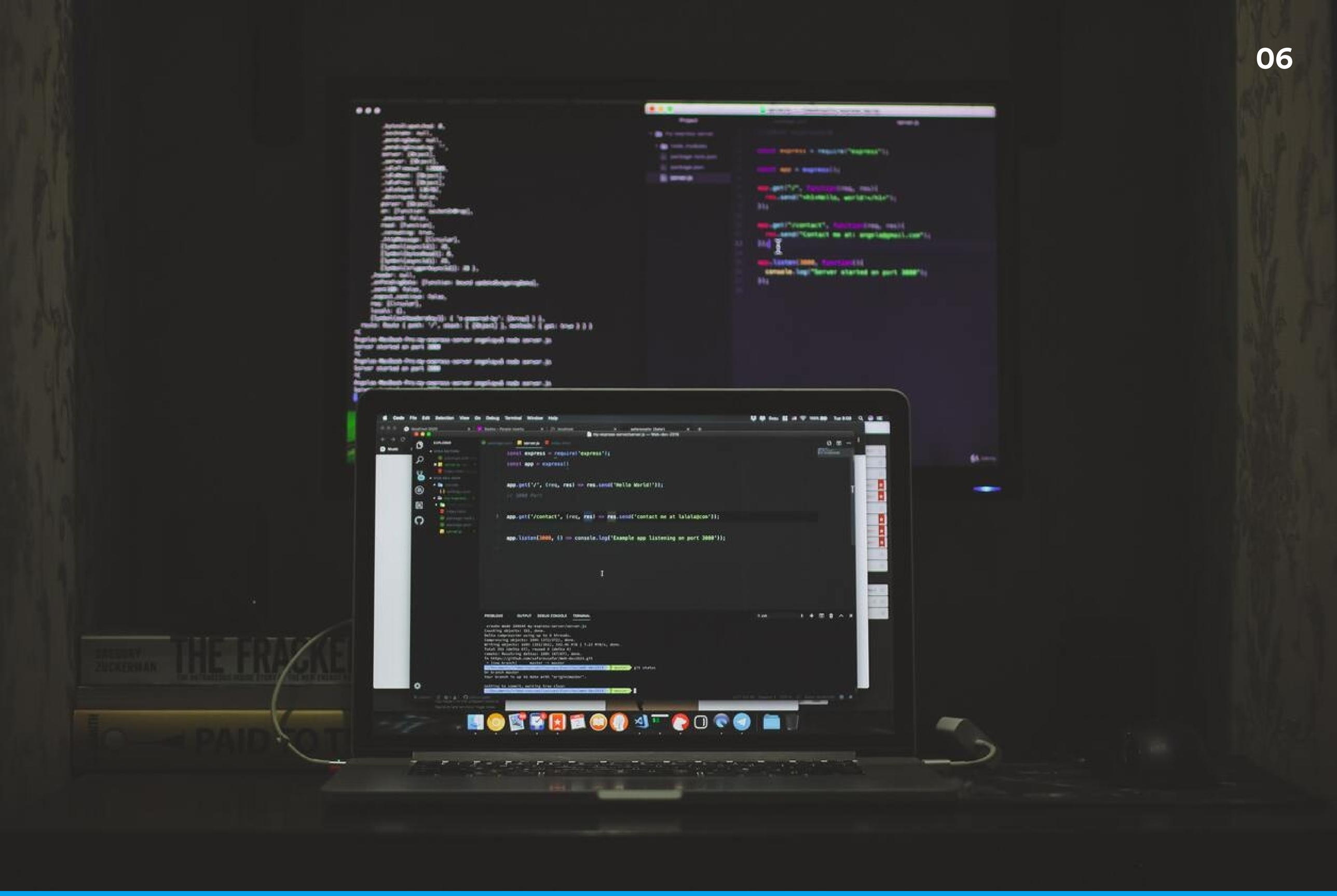
07
Maintenance

The role of frontend and backend developers mainly come into the picture during the development or the build phase. Once the designers finalise the look and feel of the website or application as per initial planning and gathered information, the role of a developer begins.

What is FRONTEND?

In the process of web development, frontend developers deal with parts of the website or web application that users see and interact with. The frontend developers will be working towards making websites accessible, functional and aesthetically pleasing. Frontend developers bring life to concepts developed by web designers. They work closely with the designers to bring out the best possible experience for the customers or visitors.

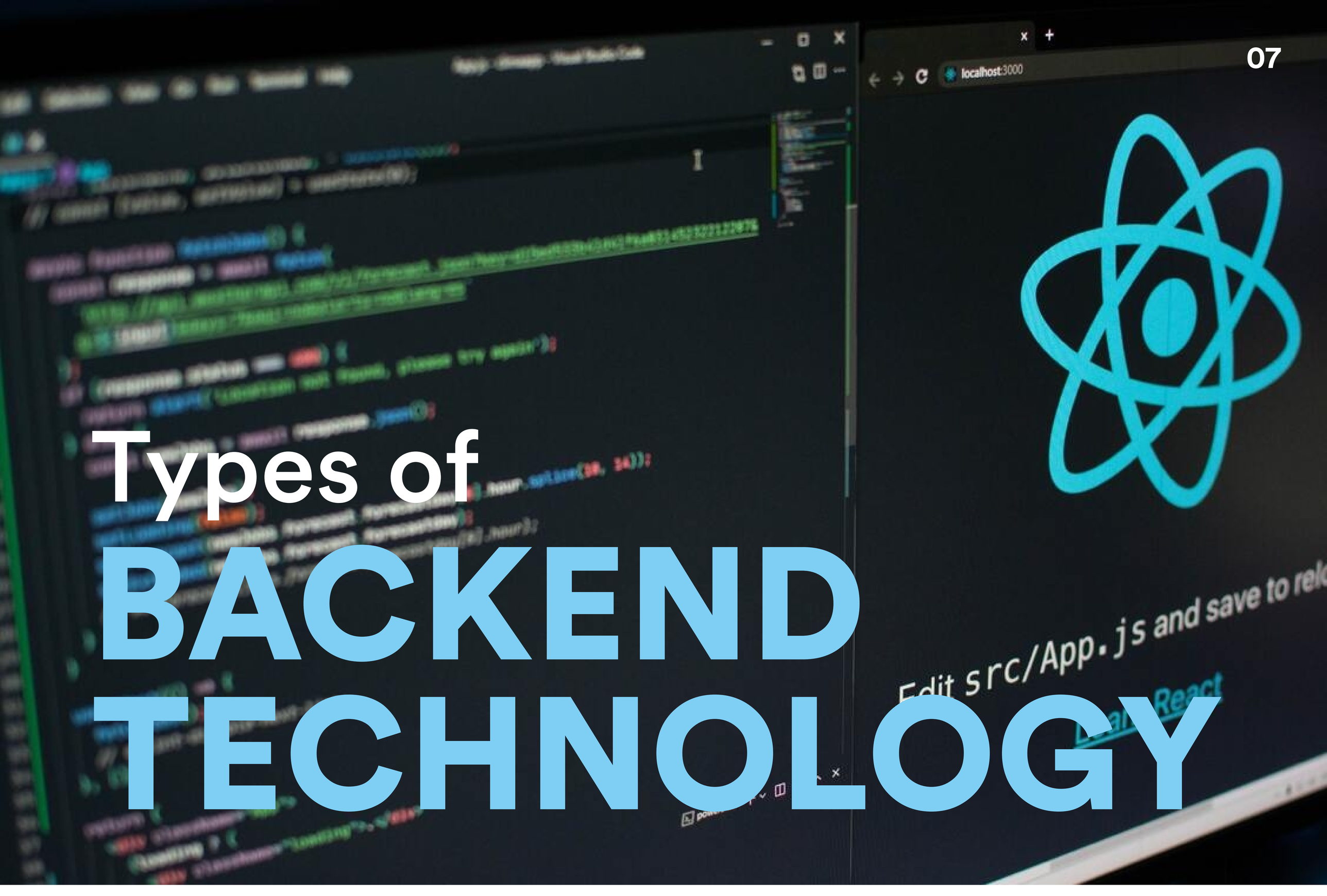




What is BACKEND?

The backend development in the process of web development deals with the server side of a website or a web application. In simple terms, backend developers work on the parts of websites users cannot see. They ensure that the website functions correctly with focus on databases, APIs (Application Programming Interfaces), architecture and servers with backend logic in mind. Backend developers make the process of frontend user requests smooth and seamless.

Without further ado, let us get knowing about the different types of backend technologies preferred by backend developers all around the world.



Types of BACKEND TECHNOLOGY

01 JavaScript

JavaScript, the most popular language for backend development is also the most frequently used programming language in the world. However, when we talk about JavaScript for backend, we imply the usage of JS runtime environment, which is Node.js backend. How can Node.js be beneficial for your business?

With the assistance of Node.js, you can execute server-side operations under JavaScript programming technology. But it is also essential to understand that Node.js works as a platform and mostly uses Express.js as its server-side framework.

In simple words, Express.js is an open-source backend framework for Node.js, using frameworks that streamline application development and improve general user experience.

Express.js in conjunction with Node.js uses JavaScript as both front and backend language. It also develops application programming interfaces (APIs) for web, mobile, hybrid, single and multi-page applications.

Why use JavaScript?
(USPs/Features)

Scalability

Frontend and backend developers benefit from the high level of scalability Node.js offers. This language provides both vertical and horizontal scalability options and allows the addition of extra nodes as well as additional resources.

High Performance

Node.js uses Google’s V8 engine to swiftly compile the JavaScript code into Machine code. The V8 engine is known for allowing an easier and faster code interpretation, which saves time and money for software development process.

Minimalist Backend Technology

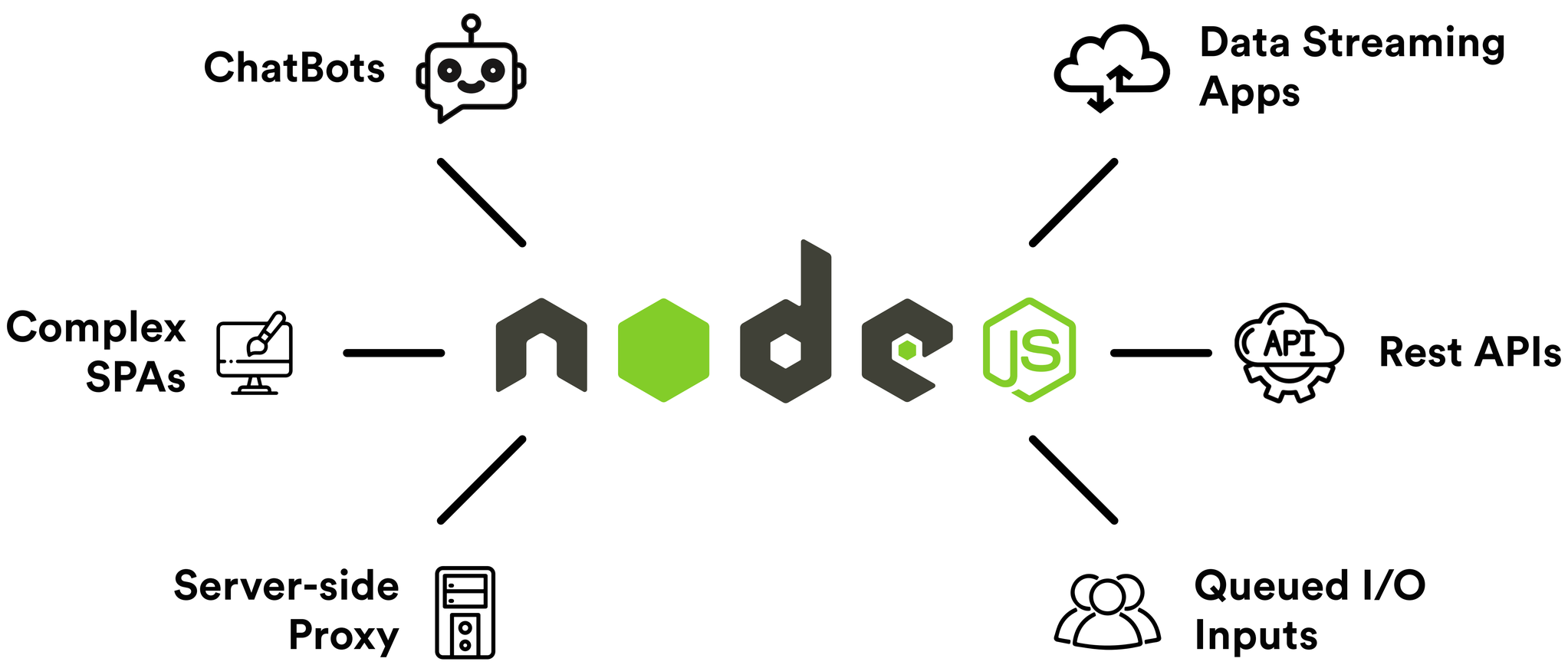
JavaScript’s backend platform like Express.js offers Express middleware modules to figure out different development challenges. These middleware packages consist of HTTP request logger, error handling, security headers and POST data functions.

Open-Source Community

The scope of improvement becomes lesser without community contribution and feedback. Luckily, JavaScript is an open-source community and backend engineers always have the chance to be reviewed. Such reviews lead to an improvement in programmers’ backend coding skills.

When to use Node.js?

Its event-driven architecture and asynchronous data processing makes Node.js a perfect choice for IoT, real-time chats and single page applications. Node.js' Stream API feature allows developers to build streaming apps, while Node.js' best backend frameworks like Express facilitate the creation of apps with microservices architecture.



JavaScript Limitations

- It isn't easy to understand the event-driven nature of JavaScript backends. Especially developers who work with other coding languages misread the call-backs and draft entire code in call-backs. However, this problem is rare in the latest versions.
- Programmers usually misinterpret the concept of middleware while using JavaScript server-side programming.
- It is also hard to host backend frameworks of JavaScript with MySQL databases.
- Few backend engineers also complain against the unopinionated nature of JavaScript backend technologies. According to them, standard designing is much better instead of this freedom.

Backend Technologies

Types of

02 Python

Python is a leading all-purpose programming language created by a Dutch programmer Guido van Rossum in 1991. With core competency of concise and readable codes, Python supports backend programmers to write rational and explicit scripts.

Why use Python?
(USPs/Features)

Short Libraries

Python delivers a variety of standard libraries that simplify the process of development. Libraries for string operations, web service tools and protocols allow coders to reuse the available scripts which shortens the development cycles.

Conciseness

Python’s motto is concise yet expressive coding that reduces the coding time and allows developers to execute the same operations with fewer code lines. Python is also known to be less complicated than other backend technologies, which makes coding easier and less prone to errors and bugs.

Extensive Libraries

The backend programming technology downloads extensive libraries. These libraries include codes for unit-testing, databases, regular expressions, browsing and email, etc. In short, with the support of extensive libraries, backend engineers don’t have to write each code manually.

IoT Opportunities

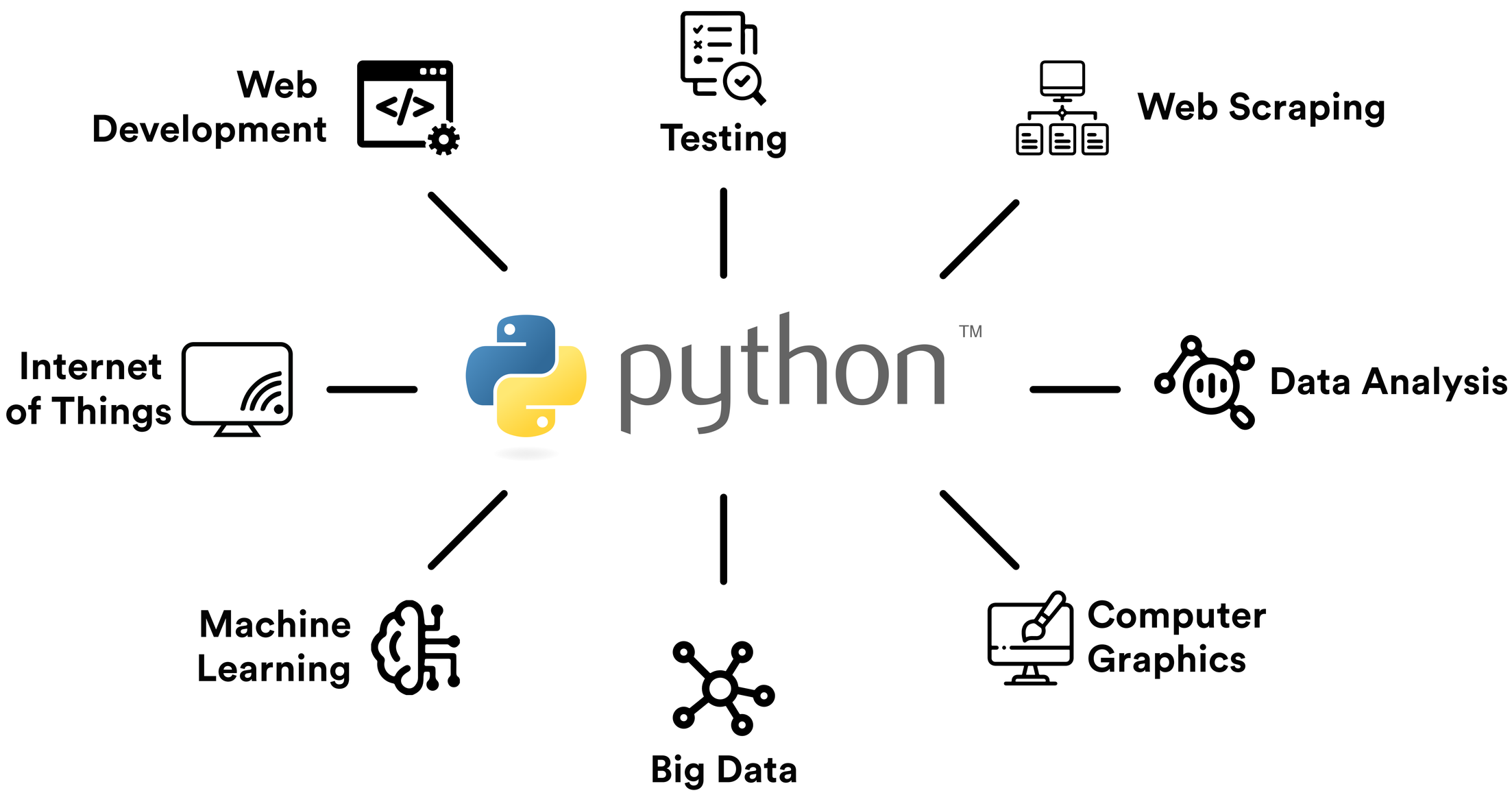
With the help of Python’s modern programming features, you can build physical projects on Raspberry Pi.

Embeddable

It is easy to insert your Python code in the source code of languages like C++.

When to use Python?

Python's wide range of frameworks like Django, Pyramid and Flask made this backend programming language the number one choice for web development. The rich selection of libraries entices developers to utilize Python for game development as well as complex scientific and numeric applications. Finally, Python's simplicity and conciseness allows coders to use it for AI, ML and IoT projects.



Python Limitations

- Any interruption in Python's coding can lead to slower execution. It can badly affect the speed of the project as well.

- As compared to other backend technologies, the database access layers of Python are not fully developed.

03 Ruby

Ruby is another programming language used for the web development backend. Mainly used for prototyping, Ruby is relatively easy to learn and supports various programming options like functional, object-oriented and procedure.

Tech businesses like Etsy, Airbnb and Shopify are using Ruby as backend scripting technology due to its object-oriented, scalability and flexible programming features.

Why use Ruby?
(USPs)

Metaprogramming

Metaprogramming is an exciting feature of Ruby as it allows developers to create a program that will autonomously write the code. Using a variety of sophisticated tools like introspection, class macros and ghost methods, metaprogramming provides flexibility and shorter development cycles.

Ruby on Rails

Probably the best part about Ruby is Rails - a web application framework. Rails simplifies the coding with Ruby and provides a testing framework to write test scenarios that make the application more robust and secure.

RubyGems

Ruby’s libraries for web application features are called Gems and there are already more than 160,000 of them. Gems are of great help when developers face problems that they cannot solve on their own, as gems offer a solution that you can access in the RubyGems repository. And if new coders still cannot find the right solution, they can refer to Ruby’s large community for help and answers.

Productive

Due to short readable code and third party libraries’ availability, Ruby is a very good product backend technology. Developers typically need fewer separate documentations with Ruby. It allows backend technologies to use already available projects.

When to use Ruby?

Ruby's extensive libraries, also known as gems, help developers create web applications, especially eCommerce apps, as many gems are directed at eCommerce issues. Besides that, Ruby is also great for prototyping and creating custom databases.

Image Source: https://invozone.com/blog/why-use-ruby-on-rails/

Ruby Limitations

- The runtime speed of Ruby is slow as compared to other scripting languages.
- Although Ruby is an open-source community, unluckily it doesn't have enough libraries and sources.

04 PHP

PHP is a leading server-side scripting technology that was introduced by Rasmus Lerdorf in 1994. This open-source backend technology is commonly used for websites. Around 79.1% of websites on the internet use PHP as server-side technology, according to the recent survey by W3Techs.

This general-purpose scripting language is easy to use and emend information in the databases. The availability of several modern frameworks, massive community, robust codebase and easy deployment adds tremendous value to this technology.

PHP developers argue it is better to use PHP with MySQL and Linux Apache for a smooth development process.

Why use PHP?
(USPs)

Database Flexibility

PHP has a built-in module that facilitates a seamless connection to databases. PHP allows for smooth working with multiple databases, such as MySQL, Postgres, Oracle, SQLite and many others.

Part of a well-tested stack

LAMP stack is one of the most used tech stacks and together with Apache HTTP, MySQL and Linux, PHP is a part of it. First, this tech stack has been tested and validated by multiple businesses, which makes it an appealing backend technology. Secondly, all the LAMP's elements are free of charge, which saves companies a good portion of their budgets.

Access to cloud services

PHP back-end applications can be easily deployed on a cloud server like Amazon Web Services. The popularity of cloud services derives from the scalability that they provide, the reason why PHP apps are known for their scalability.

Best for Beginners

PHP is suitable to learn for beginner backend engineers. Features like running sequence and low learning curve make PHP beginners' best choice to learn.

Automate the Development Tasks

With PHP scripting technology, it is easy to automate development tasks like session management, URL mapping, authentication, etc.

Security Against Target Attacks

No doubt developers take PHP as an unsecured backend language. But with built-in security functions, you can avoid these threats.

When to use PHP?

The database accessibility and flexibility that PHP provides makes it one of the best backend programming languages for search engines and eCommerce platforms. PHP is also a popular choice for gaming apps, social media platforms and remote access for businesses.

PHP Limitations

- The popularity of PHP is declining day-by-day. Developers rarely consider including PHP in their skillset nowadays.
- PHP is not competing with modern backend technologies like Python and Ruby due to lack of advanced libraries.

05 Java

Java is one of the most powerful backend technologies, which has the second rank, according to TIOBE Index. James Gosling originally developed this programming technology in 1991, but it was published in 1995 by Sun Microsystems.

Developers prefer to make feature-rich and adaptable web applications with Java for years. However, one can use Java for mobile devices and microcontroller software development as well.

Why use Java?
(USPs)

Automatic Garbage Collection

The feature is self-explanatory as it implies that Java automatically gets rid of unused or redundant objects. This is possib le with the Java Virtual Machine that facilitates automatic memory management that tracks the objects in the code and removes unused parts without the help of human involvement.

Efficiency

Java is proven to be a simple and well-designed backend technology that delivers efficient solutions. The powerful set of APIs, advanced features and Java backend frameworks like Grails and Spring play into the popularity of the language and make it a beloved technology for many backend developers.

Simple & Highly Scalable

Java EE is highly scalable because it permits numerous instances to server requests. Instant availability of Java components and unambiguous syntax technology also makes it simple for developers to use this backend language.

Multi-Threading

Java allows to handle all requests in independent threads due to a multi-threaded web server. With this multi-threading feature, Java functions very well for CPU-intensive applications.

Security

Java offers terrific features to overcome security risks. Likewise, Java Virtual Machines examine java bytecodes to detect and reduce the risk of viruses. Similarly, the security model of Java and testing of reusable codes help developers in avoiding security threats.

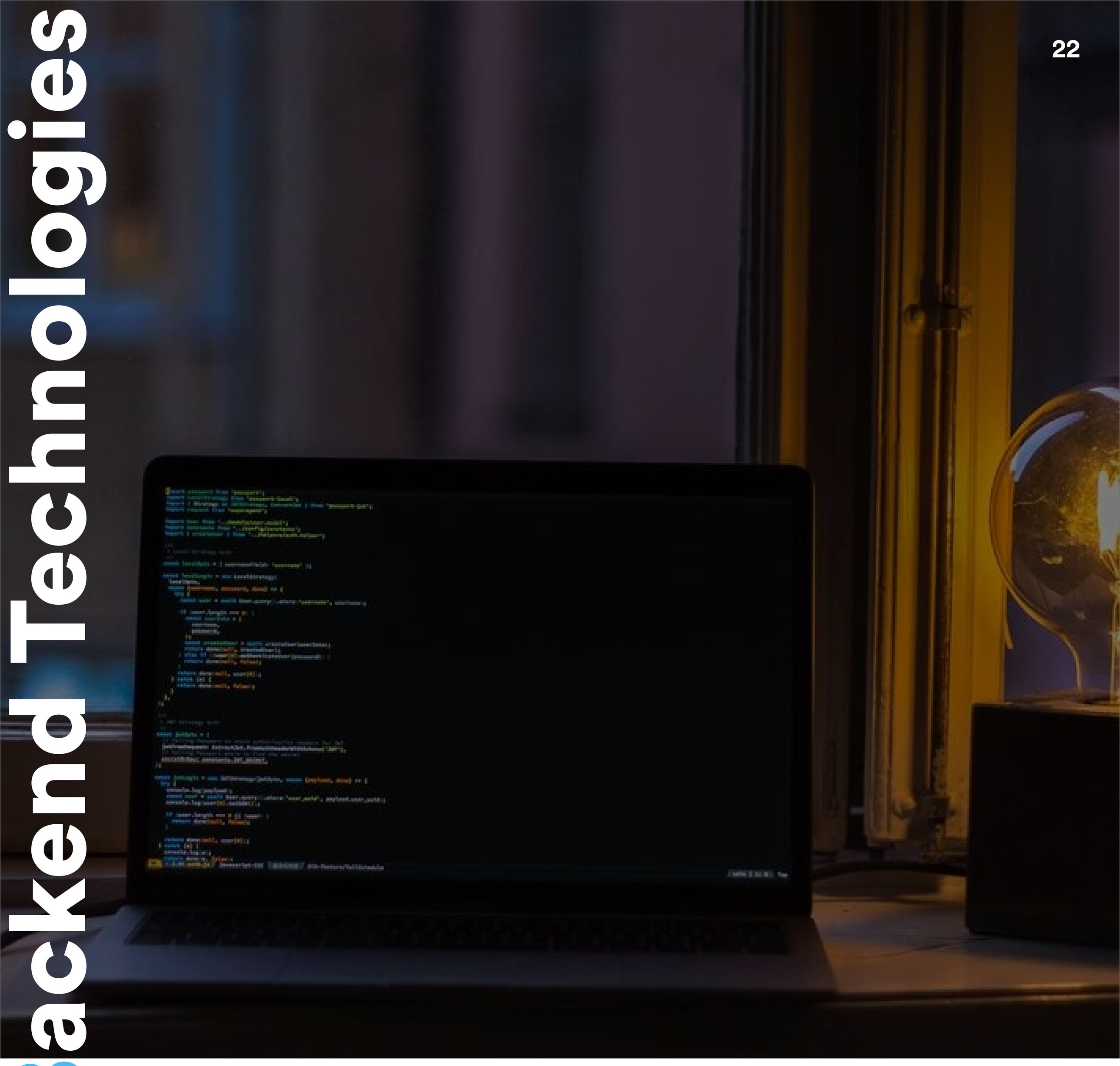
When to use Java?

Java's security and portability make it the best backend language for building scientific applications, enterprise solutions and games. Besides that, Java is a common choice for web applications and Android mobile products.

Java Limitations

- The server-side programming consumes more time and memory.
- Java doesn't offer command over garbage collection and low-level programming support is also missing in Java.
- Due to high hardware cost, it might be expensive to use Java.
- Swing toolkit, which Java is using for GUI applications is different as compared to trendy ones.

Types of Backend Technologies



06 Golang

Golang is one of the newest web development technologies that has already made a splash in the software development community. Created in 2009 by Google developers, Golang has simplified the building of complex architectures and processes.

Why use Golang? (USPs)

Concurrency

Concurrency in Golang is the ability for functions to run independent of each other without disturbing the overall flow of the program. The capability of running concurrently with other functions is attained via a goroutine function.

Backend Development Tools

Golang is open-source and provides a multitude of different development tools for any project. You can use open-source editors, IDEs and plugins that can be accessed through the GitHub repository, as well as online communities where Golang developers will answer any queries.

High Speed

Golang’s simple structure and syntax make it an easy-to-learn language with a high velocity. Without using classes and types, developers can write code based on functions, which makes the code cleaner and simpler.

When to use Golang?

Having been created by Google and mainly for use by Google, Golang is the gateway to native cloud solutions. Golang is also a great backend technology for building highly scalable databases, as well as for web development.

Golang limitations

- Golang, being a young and still developing language, developers might have to write certain libraries themselves.
- Lack of manual memory management may lead to overhead garbage collection.
- The error handling has not been perfected yet for which solutions are still being searched for.
- The runtime safety level of Golang is not that good.

Types of Backend Technologies

07 C#

C# known as C-sharp is one of the most famous backend programming languages preferred for automation in the Windows environment. C# is also used for web development in the ASP.net framework. It is one of the oldest programming languages and an extension to C++.

This backend technology is most used for desktop applications and embedded systems in this era. The execution speed of C# is faster than most of the other programming languages. In the modern world, C# is most widely used in game development with platforms like Unity. Backend engineers also use this technology to develop console applications.

Why use C#?
(Features)

Object-Oriented Language

C# is an object-oriented language that means you can structure your code using classes and relationships. It is helpful to implement the system with easy troubleshooting if something goes wrong.

Cross-Platform

The applications that use the C# backend can operate in different operating systems such as Windows, macOS, etc.

Compatibility

C# applications allow interoperability with older legacy systems. This backend scripting technology is also useful for organisations that are not updated and are using old programming frameworks.

C# Limitations

- C# is a high-level language, so it does not let the programmer interact with the hardware directly.

- This backend technology is less flexible as compared to other scripting languages. It runs only in the .NET framework and can only be hosted on the Windows platform.

Types of Backend Technologies

08 Perl

Perl is another general-purpose backend technology that Larry Wall developed 34 years ago. According to the Stack Overflow Developer Survey, Perl is the topmost paying technology globally. By the way, it is also significant to know that the same survey ranks Perl among the top three dreaded languages.

Whereas Perl is old-fashioned, developers are still using Perl 5 for quick automation and prototyping.

Why use Perl?
(Features)

Multi-Platform

Perl can operate with different platforms including Macintosh, Windows, and most UNIX variants.

Embeddable & Extensible

Perl can be easily embedded in any of C++ and C applications. Through SWIG and XS, Perl also backs external libraries of C and C++.

Text Processing

Perl is a perfect scripting language for text processing. Luckily, Perl's recent versions also maintain POSIX-compliant systems and process socket calls with other advanced features.

Perl Limitations

- Perl doesn't provide high-performance processing as compared to other backend technologies.
- Perl libraries are not advanced enough.
- Perl is not the right option if you want to improve the scalability and speed of your project.
- It is expensive to hire Perl backend engineers because it is slightly outdated, and developers don't prefer to learn it.

09 C++

C++ is an extended version of C language. C++ was introduced with classes. This concept of classes or Object-Oriented Programming (OOPs) was missing in C language.

The idea of OOPs is vital for any programming language in the modern world to write structured code using classes and defining their relationships.

C++ is one of the oldest programming languages and is mostly used in system programming & embedded systems. C++ is a low-level language and it can interact with the hardware resources. It is used for gaming applications, operating systems, database software, etc.

Why use C++?

(Features)

Object-Oriented Programming

C language does not support object-oriented programming, but C++ fulfills this deficiency. This feature makes this language more powerful and easy to code with structured programming.

Memory Management

Programmers get complete control over memory management using C++. It can help them to manage memory effectively to execute the program.

Low-Level Language

C++ is a low-level language and close to the system. Therefore, most of the embedded systems are built in C++ because it can directly interact with the hardware resources.

C++ Limitations

- There are some security issues in C++. With C++ backend technology, users can directly interact with the hardware using C++ as a low language.
- There is no garbage collection in C++ to filter out unnecessary data.

Types of Backend Technologies

10 Kotlin

Kotlin is a backend programming language that is being used for the development of android applications. It is taking over Java as the most preferred programming language for development of android applications. Its demand is also increasing day-by-day with over 60% of android app developers using Kotlin for the backend development. Kotlin interoperates completely with Java & JVM.

JetBrains introduced Kotlin as a backend programming language for Android applications in 2011. Since then, it has become one of the most famous programming languages in the world of computer science.

Why use Kotlin?
(Features)

Concise Code

Programmers can solve more significant problems by writing fewer code lines compared to other languages.

Easy to Maintain

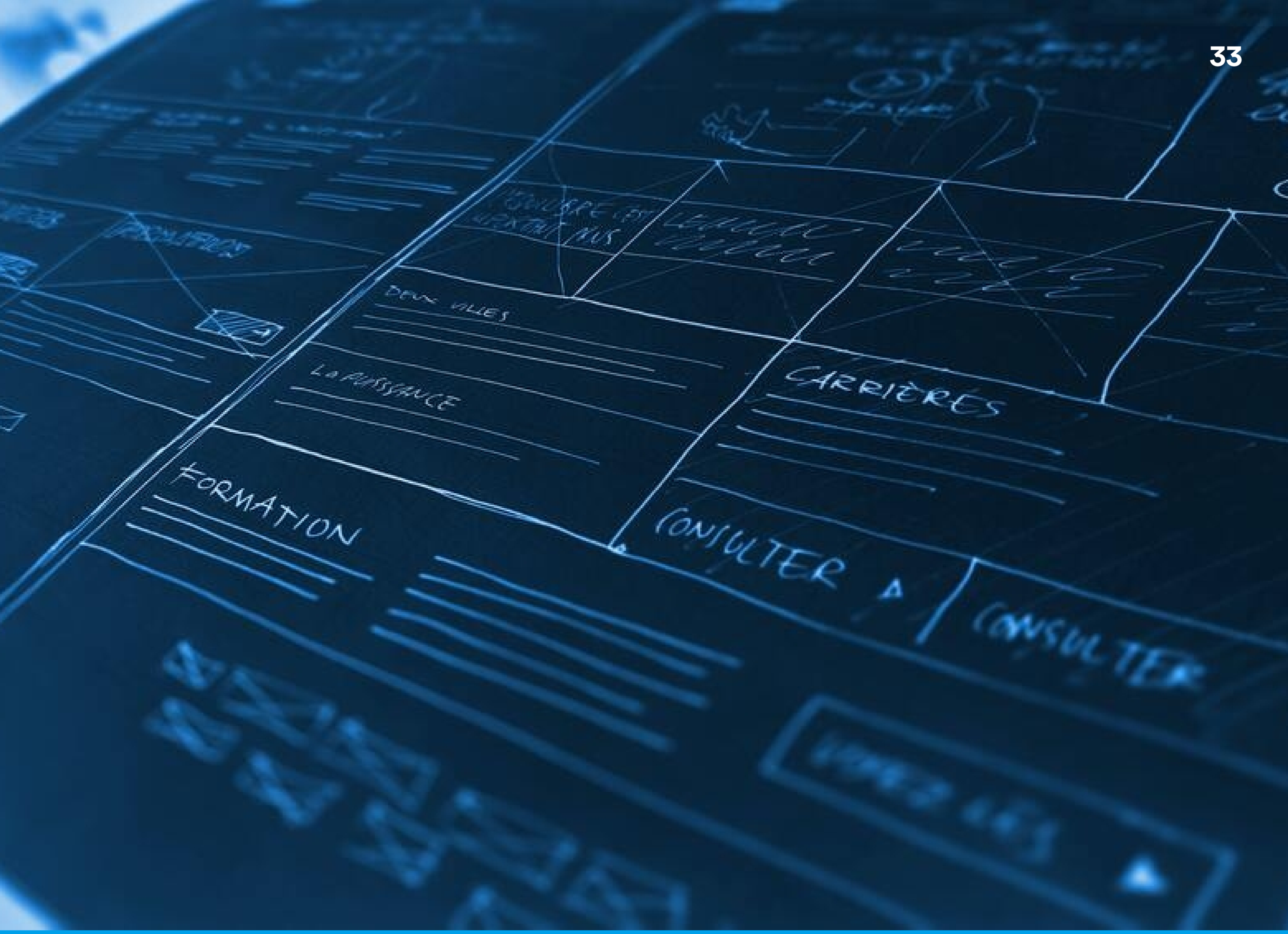
Due to concise code, Kotlin helps the programmers easily read and maintain the code.

Interoperable with Java

This programming technology is completely compatible with Java. Programmers do not need to change the whole project to switch to Kotlin. With all Java tools & technologies, they can add it to their Kotlin project to implement additional functionalities.

Kotlin Limitations

- Java is faster in compilation than Kotlin when it comes to clean builds of android applications



Serverless ARCHITECTURE

Despite being called serverless architecture, it does use servers. What it means for an architecture to be serverless in this context is to abstract the deployment process.

The less abstract method of deploying code is to put the code on a ‘bare metal’ server hosted by you. Then you would have to connect it to the internet with an IP address and connect the IP address to a DNS. This will require you to manage your server hardware, take care of security updates, create backups in case of failure, scale up and down your servers manually. All of this can take up too much of your time in deployment and leave you with less time to focus on development.

There are multiple levels of deployment abstractions, such as,

VPS – Virtual Private Servers

VPS gives you a dedicated resource to host your code. It uses virtualization to host multiple servers for multiple users.

IaaS – Infrastructure as a Service

IaaS gives you compute, storage and networking resources to host and manage your code.

PaaS – Platform as a Service

PaaS will host your code and manage it on their infrastructure.

Deployment Abstractions

	Less Abstracted				More Abstracted
Concept	"Bare metal" Servers	Virtual private servers (VPS)	Infrastructure-as -a-service(IaaS)	Platform-as-a-service (PaaS)	Serverless Compute
Implementation Examples	Dell, Lenovo, Build-your-own	Linode, Digital Ocean, OVH	AWS, Azure, Google Cloud	Heroku, CodeStar, PythonAnywhere	AWS Lambda, Azure Function

The further right means giving more control and trust to a platform. There are tradeoffs along the entiredeployment spectrum so left or right is not inherently better.

Serverless

Serverless Computes like AWS Lambda will abstract away all the deployment processes. It integrates third party services that can do server management, automatic scaling of applications and security updates.

Serverless Architecture uses microservice design in which the app is isolated into small services that can communicate as a whole. A deployment task can also be a service that is executed on an HTTP request call.

Important Characteristics:

Lowered Operation Costs:

Serverless Architecture will have no services running unless an event occurs, which reduces the cost when you are paying per event and not time.

Reduced Development Time:

Using third party libraries wherever possible will reduce codebase size and microservice design will make all services well-defined and concise.

Trade-offs:

Vendor Lock-in:

Complete reliance on using a third-party platform as running your own platform will be quite resource consuming

Complexity:

If design principles are not followed properly, the risk of increasing complexity in codebase is high.

Use-cases:

Trigger-based Tasks:

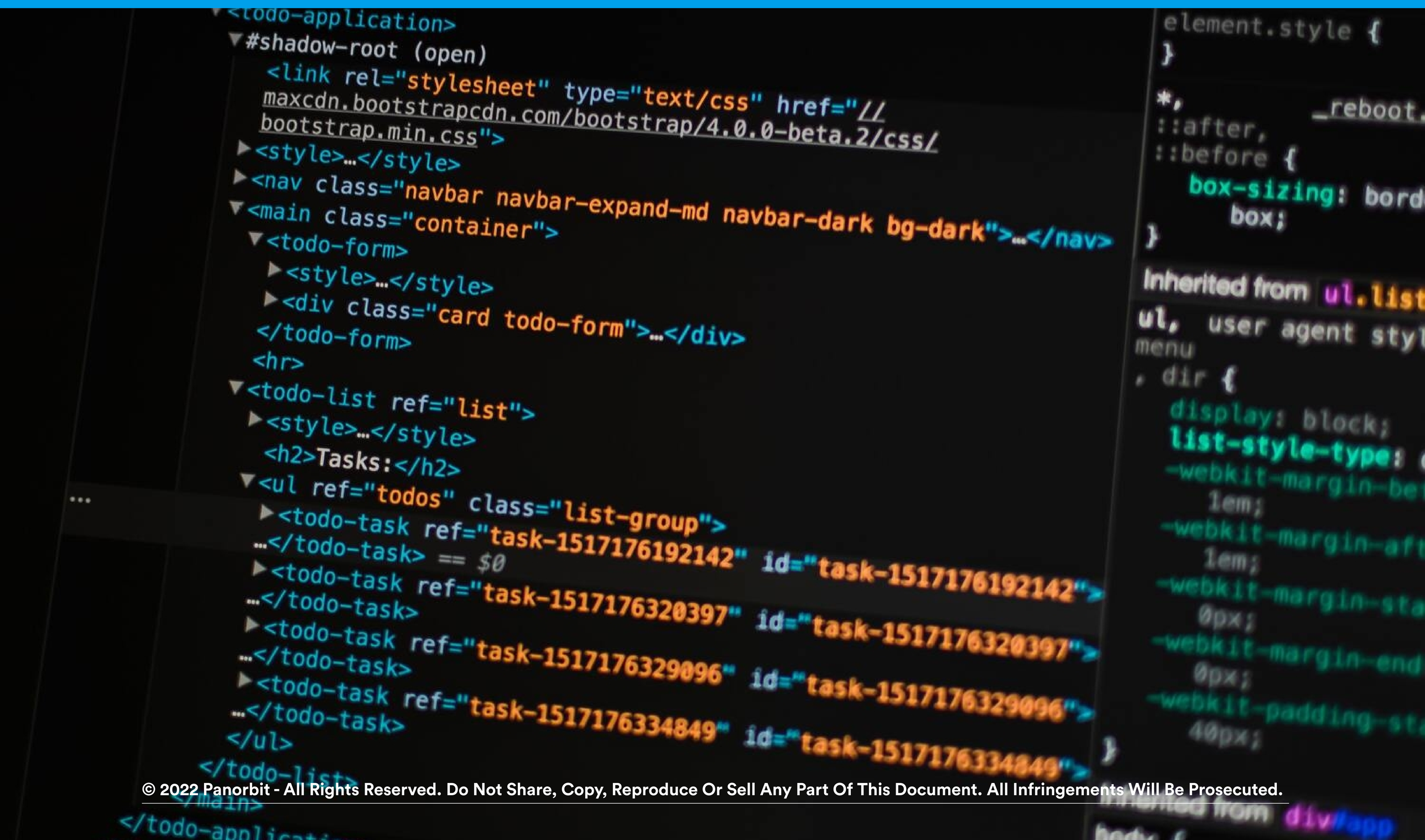
Services where a chain of events is triggered, can be handled through a chain of serverless functions.

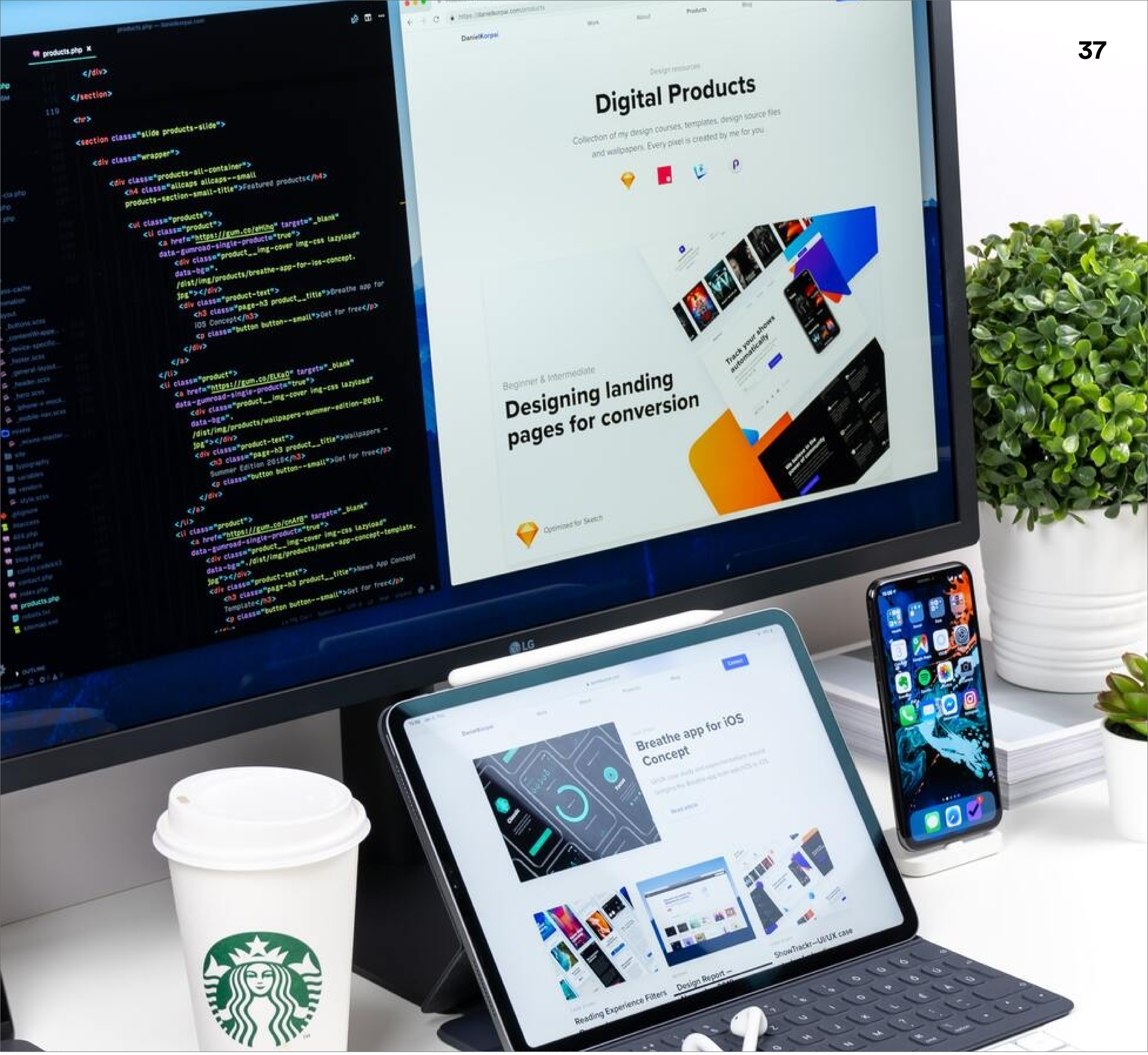
CI/CD:

Continuous Integration and Continuous Development stages can be automated using serverless architecture.

What makes a good BACKEND DEVELOPER?

Being well versed in some or all of the backend technologies listed above doesn't necessarily make someone a good backend developer. A good backend developer is inherently a good problem solver. Proper understanding of frontend technologies to an extent also makes the work of backend developers coherent, helping the process of web development as a whole.





Conclusion

Choosing the right backend technology for websites or web applications depends on a lot of factors such as personal preferences, timelines, budget and flexibility to name a few. The demand for backend developers has been on a constant rise in today's digital world. Hope this eBook helped you in one way or the other.

Aspiring frontend developers, there is no need to feel left out. We are coming back with a list of frontend technologies, soon!