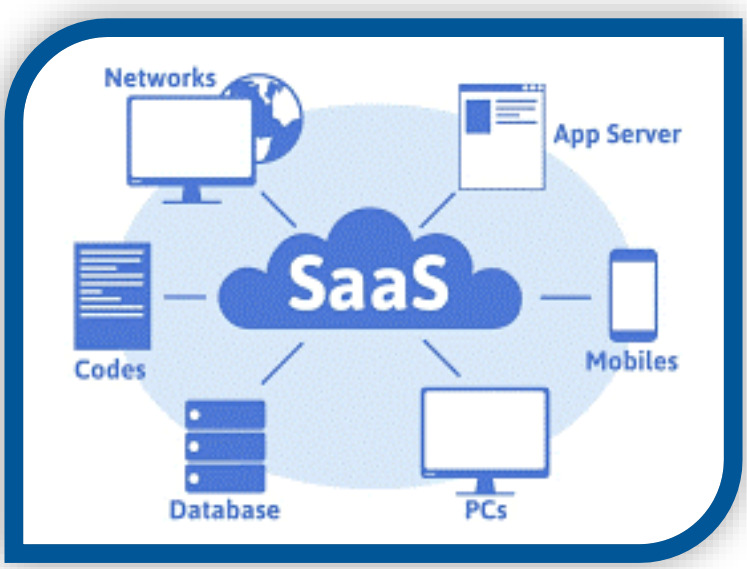
SaaS - Software as a service

Cognitive Convergence
http://cognitiveconvergence.com

CONTENTS

OBJECTIVES

Our objective is to develop solution utilizing the SaaS architecture offerings for two Cognitive Convergence products EdConvergence and PsycheConvergence.



COMPANY PROFILE

Company Location: Lahore, Pakistan

Company Website: http://www.cognitiveconvergence.com/

COGNITIVE CONVERGENCE has an established process, technology stack, and team in terms of roles, culture, and understanding to give the best possible solutions and services to CLIENTS in the USA (more specifically in California and Washington state) and Europe. Cognitive Convergence offers its partners an advantage of reducing their development costs, lessen their work time to market, and restate their solutions as business PARTNERS to remain swift in the face of change and will be able to thrive over time. We follow the principals of developing and growing to help our INVESTORS to reach high levels of excellence, such that you will not be disappointed with our strong and progressive results.

SAAS – SOFTWARE AS A SERVICE

Software as a Service is based entirely on the Internet, and it is an approach to software distribution by which software providers host a combination of servers, databases, and code to create applications that can be accessed by

users from connected devices. Software as a Service (SaaS) brings the power of a firm's workflow to any user anywhere in the world at any time.

SAAS DEVELOPMENT

SaaS software development is a reasonable choice for businesses. It will be much easier to deploy your product. The software runs on the server, so your project requirements won't depend on numerous configurations of every platform. Maintenance and updates are faster to be implemented as well.

SAAS DEVELOPMENT LIFE CYCLE

Every company that is interested in building and marketing the application successfully has to be familiar with the SaaS development life cycle. It ensures the continuous delivery of the product and efficient implementation. Consider the following phases of SaaS development life cycle.

Envisioning

The business strategy is an essential part of any development. Always start with finding business opportunities and ways to attract new customers. Taking into account the distinctive abilities of cloud services, identify the value you are going to deliver. In addition to the development strategy, you should think about the marketing strategy way ahead.

Platform evaluation

Making a choice on the SaaS application development platform, remember that it should acquire certain capabilities. Since the final architecture has to acquire great performance, reliability, compliance, scalability, the chosen platform has to provide the necessary tools.

Planning

As long as you have defined the necessary requirements, it is time to plan step-by-step implementation and focus on the development solutions. Your project plan should cover details regarding not only its functionality but timeline, cost, and required specialists. However, it should be divided into stages and suggest the long-time vision options.

Subscription model

It is an important step in the SaaS development life cycle to acquire a production quality subscription. In order to make the right decision on the cloud platform and the central hosting model, you should trial the chosen services and prepare all the required documentation. Reach the necessary level agreements and make sure that your subscription covers the backup and recovery strategies.

Developing

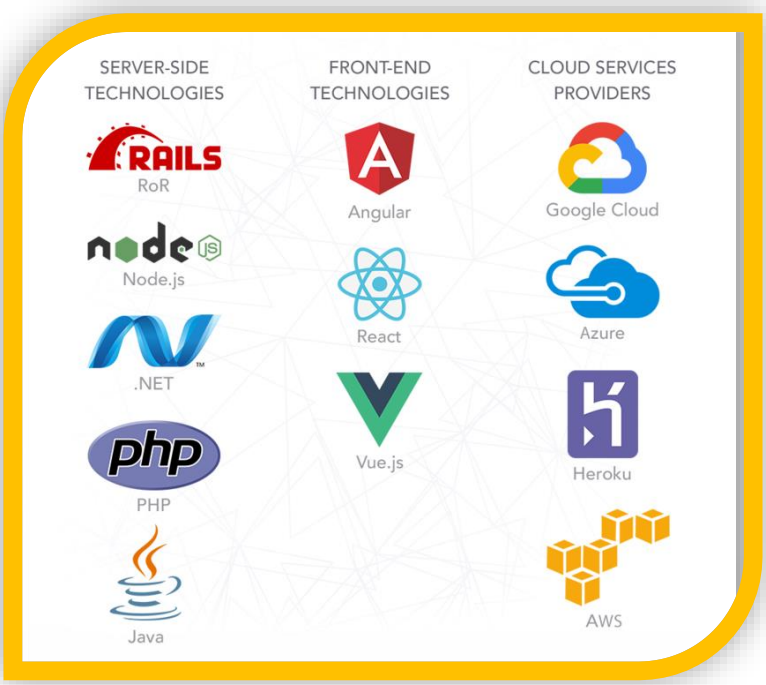
Here comes the development – building the core features and creating scalable software architecture. The custom SaaS development process always goes along with testing. One of the often-used approaches is A/B tests to evaluate the performance and iteration. You couldn't exclude testing as it enables the overall functionality.

Operations

This phase is exceptionally important for deploying all the operations. They should correspond to the level agreements, security, compliance and shared infrastructure of the cloud services. It has to cover the specifications of the previous stages that have already outlined the ways of service deployment in production.

SAAS DEVELOPMENT TOOLS

Today's technology offers a great variety of SaaS development tools to build a valuable product. This software obtains the complex architecture that is delivered by means of different frameworks, libraries and SaaS developer tools. Selecting the tech stack, every cloud services provider works better with a certain programming language, it is needed to find the perfect combination for the SaaS product.



Server-side technologies

Those technologies do not differ from the common server-side web development ones. You can choose among RoR, NodeJS, .NET, Java, PHP, etc. All these programming languages have their own benefits to be applied. The main requirements are easiness to use, customization, available documentation, and strong community support. You can check for more details on server-side frameworks here.

Front-end technologies

It is quite easy to make your choice here, as JavaScript frameworks have successfully gained their leading positions at the market share. Angular, React, Vue.js are known for the distinctive functionality to

deliver excellent UI/UX designs. Learn more about popular front-end technologies, pros and cons of each of them in our article dedicated to this topic.

Cloud platforms for hosting and maintenance

The majority of SaaS companies use third party providers for the hosting of their products. On-premise hosting would be extremely expensive in their case not to mention a tremendous cost of developing anything close to Amazon's platform for cloud management. Cloud platforms like Azure, Google Cloud, AWS, Heroku and others offer a wide choice in terms of hardware, supporting software, and DevOps tools.

To give you some examples of their distinctions, AWS was among the first market representatives. This platform offers flexible, fast and reliable solutions for any type of application in the cloud. Azure is known for its consistency and flexibility. Its scalability allows managing resources according to business needs. As for easy implementation and deployment, a lot of companies go with Heroku.

SAAS ARCHITECTURE

With this model, a single version of the application, with a single configuration is used for all customers. The application is installed on multiple machines to support scalability (called horizontal scaling). In some cases, a second version of the application is set up to offer a select group of customers with access to pre-release versions of the applications for testing purposes. In this traditional model, each version of the application is based on a unique code. Although an exception , some SaaS solutions do not use multitenancy, to cost-effectively manage a large number of customers in place. Whether multitenancy is a necessary component for software-as-a-service is a topic of controversy.

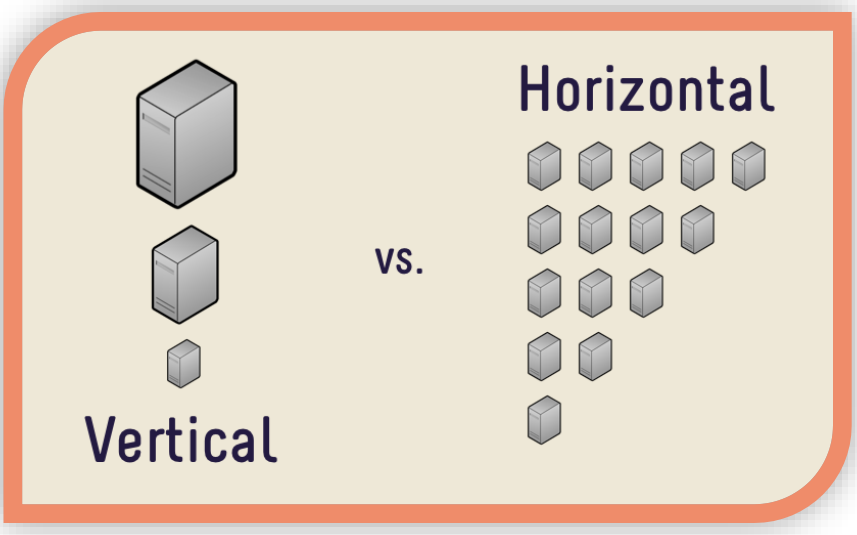
There are two main varieties of SaaS:

Vertical SaaS

A Software which answers the needs of a specific industry (e.g., software for the healthcare, agriculture, real estate, finance industries)

Horizontal SaaS

The products which focus on a software category (marketing, sales, developer tools, HR) but are industry agnostic.



DESIGN CONSIDERATIONS FOR A SAAS PLATFORM

Scalability

Will your SaaS architecture be able to scale and accommodate hundreds, if not thousands of users simultaneously accessing it over the web? Well-designed SaaS applications need to be able to do this.

You can achieve this by using hardware like Network Load Balancers to evenly distribute incoming traffic across multiple web servers. From a software architecture perspective, you can introduce a separation of concerns by having individual layers to handle data access, business logic, and the presentation layers of your application which will help your application scale more easily.

Zero downtime and Service Level Agreements

With internal or on-premise software applications, users are more forgiving on the occasion that software must be taken offline for a given period of time whilst internal IT install a new kit or release a version update.

With a product delivered over SaaS architecture, you don't have such a luxury. Users typically expect the product to be available almost all the time and with no interruption to service. When was the last time you heard that Facebook or Twitter was down?

Take time to consider how you'll factor upgrades, patches, or debugging and troubleshooting production issues into your SaaS applications architecture.

Multi-tenancy

For your software to be delivered as a SaaS product, it must support multi-tenancy. Your product needs to be able to accommodate multiple users whilst at the same time, ensuring that user data, privacy and are all still being observed. Take the time to factor this into the design of your SaaS architecture and ensure that whatever you implement, has a scalable model.

CAPABILITIES OF SAAS SOLUTIONS

SaaS platforms have a wide array of capabilities. Especially when coupled with other cloud offerings such as IaaS (Infrastructure as a Service) and PaaS (Platform as a Service).

Cloud technology such as Microsoft Azure lets you provision servers that can host websites, databases, and much more. Infrastructure that would have historically been physically installed on business premises and ran by internal IT teams, can now be provisioned from an online dashboard with just a few mouse clicks.

SaaS solutions can be deployed to these environments and, in theory, offer any type of service that can be developed as a software application which can include, but is not limited to:

- Office applications
- Email and instant messaging
- Social media

- Exposing 3rd Party APIs
- Security and authentication
- Machine Learning
- Artificial intelligence
- Location Services
- Data streaming and lookup services

KEY FEATURES AND BENEFITS OF A SAAS PLATFORM

SaaS solutions have different features to that of more traditional applications that are installed on your desktop for example, here are some of the other benefits that deploying a SaaS-based product can bring.

Simplicity

Software applications architected as SaaS solutions are typically accessed over the web through various types of devices. Advancements in client-side programming languages such as JavaScript have resulted in more intuitive web interfaces and as such, make using applications delivered over the internet as easy to use as their desktop counterparts.

Economical

The monthly or annual subscript fee payment model makes it easier for businesses to budget, couple this with zero infrastructure setup costs, it's easy to see how opting to use SaaS solutions can save the business money.

Security

Security is an important aspect of software development solutions and SaaS platforms are no different. As a consumer of an application architected using SaaS, you don't need to concern yourself with how your data is being locked down. It's held securely in the cloud!

Compatibility

With traditional software installations, updates and patches can occasionally require enormous amounts of time and money. This is especially true in the enterprise. Additionally, versioning discrepancies between team members of your workforce can lead to compatibility issues and even more wasted time. With SaaS, however, subscribers can simply log-on to already upgraded services.

Scalability

The main advantage of these apps is adding capacity. Handle both vertical (limited by the size of the server) and horizontal scaling (building the logical unit that works on different hardware and software).

DISADVANTAGES OF A SAAS PLATFORM

There are some disadvantages rolling out a SaaS platform or using one. Let's look at some of those now.

Lack of control

As SaaS applications are hosted in the vendor's SaaS environment, you have little to no control over the software you're using. An in-house or on-premise application will give your business more control over its behavior, for example, a Windows-based application might have more configuration options than a regular web application being delivered as a SaaS application.

Limited ecosystem

It can't be denied that SaaS is a growing trend as a software distribution channel. That said, there are still many applications that don't offer a hosted version.

Performance

An in-house, thick client or on-premise application will always run quicker than a product being delivered over the internet.

Data Concerns

When selecting a SaaS product, and for example, with the advent of the GDPR, businesses must pay special attention in terms of where any SaaS implementation stores data in the cloud. Each jurisdiction has its own legislative policies and acts when sensitive data is being processed or stored.

CHALLENGES OF SAAS DEVELOPMENT

With every development come challenges. Having a business is always a risk. Although it is impossible to foresee what problems you may encounter on your way, there are a few most frequent problems in SaaS application development.

Lack of trust from customers

without trust you will not have customers, especially in the SaaS model. Large SaaS companies have loyal users that will follow them everywhere, but when you are entering the market it may become a problem. To gain people's trust show them security, care about your user experience and user interface as well as your logo and branding.

Small target audience

although having a one-of-a-kind product is daunting it is equally dangerous. With a too small of an audience, your solution will not bring enough profit. To avoid that, try a horizontal development - by expanding functionalities. Do not forget to conduct business analysis to further understand your target customers' needs, competition and ongoing trends.

Poor idea

There are in fact no bad ideas, just poor execution of good ideas. However, there is no sense in developing an application that will not meet current trends nor customer demand. For example, a CD recording application will not be of any use in 2021, but a CD recording that allows user to save and share the data would be much better.

DEVELOP A SAAS PRODUCT

Discovering the important details on how to develop a SaaS application, we have prepared a simple step-by-step guide that will help to implement your idea into the valuable market product:

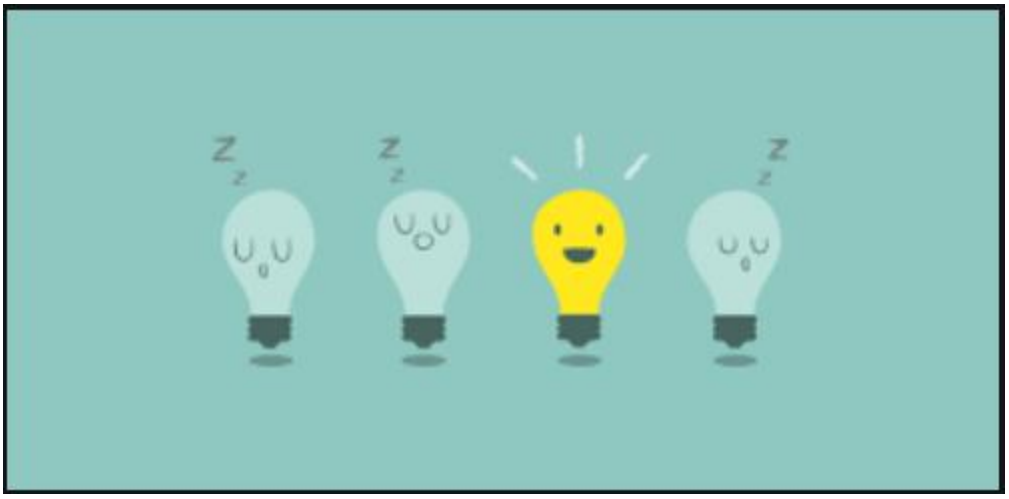
DEFINE A UNIQUE VALUE PROPOSITION THAT WILL HELP YOUR SAAS STAND OUT

There are many SaaS solutions to choose from. Before you start developing your own one, you need to completely understand the idea behind your product, how it will look like, what will its functions be. You also have to be aware who your target customers are and why they will want to use the solution, therefore, what will be necessary to help them reach their goals.

You should always keep in mind that you want to create something different, something that the future users will enjoy and want to continuously use and recommend to others. Start with thinking about what you are passionate about that has not been created yet or what can annoy your target audience in those existing solutions or even, what annoys you personally. This may bring you ideas for your value proposition which will differentiate you from other solutions.

At this point, it is worth to also think about the UX/UI, customer support and third-party integrations. Maybe those are the things that your competitors are lacking?

You should research similar products on the market and understand how you can solve the users' problems better. Market research helps to understand whether the SaaS project you are about to delve into has a potential audience as well as to find your customers aches and pains.



SaaS vendors should focus on showing the value to individual users first, instead of selling to managers or decision-makers who will not be using the software on a daily basis. Retaining customers is about continually showing them the values of your product.

To create an amazing SaaS product you should know its significant feature that distinguishes it from other solutions and deliver it as soon as possible. This will give you the opportunity to test your SaaS idea with real users and after launch, add new functionalities.

Ask yourself which features are the most important and fulfil app's main purpose. Get rid of the 'nice to haves' before implementing the 'musts'. Start with an MVP and share it with your audience.

DESIGN THE CUSTOMER ACQUISITION PROCESS

While developing your SaaS product you need to take into account the average cost that it takes to acquire a customer, and then how much profit the average customer brings in. In other words, you have to consider the customer acquisition cost (CAC) and compare it to the customer lifetime value (LTV). Your aim should be to recover the CAC in less than 12 months. The CAC and LTV will guide you through finding your most profitable and target clients.

You have to put yourself on your customers' place and write User Stories consisting of 1-2 sentences describing how your system will respond to their actions. Do not forget to include the negative cases. Those *'As a < type of user >, I want < some goal > so that < some reason >.'* sentences will help you communicate your ideas to developers and web designers.

SaaS solutions have a longer sales cycle in comparison to other businesses. The average length of the sales cycle for Software as a Service is close to 3 months. In terms of business, it is a long time to maintain a relationship and convince somebody to buy your product. On the other hand, taking into consideration what goes into the buying process it is no surprise that the customers take their time.

You should aim to create a close relationship with your customer in order to start a successful collaboration without looking too desperate. First of all, your potential clients should get acquainted with your business, find out what are you about. In many cases, they will be at the very beginning of their road and may not even be aware that you are what they are missing.

The second step would be to engage them with some of your content. Subscribing to a newsletter, getting downloadable docs is what attracts attention. In the next two steps, your consumers should explore your offering and hopefully convert into a paying subscriber. You can make use of organic traffic by creating a SEO strategy, building your brand on events and showcasing your product on paid advertisements. Those activities will be based strictly on your target audience, but it is the best option to implement them at some point.

Your job is to retain customers and keep them coming back for more with exclusive newsletters and discounts for your loyal clients. Creating a supporting atmosphere is also an important part of customer retention. Boost your customer experience with welcoming emails and propose onboarding services to the new ones. Nurture your relationship through regular interactions.

Development of a SaaS architecture is connected to storing data in the cloud and many new customers may hesitate to do so. With the GDPR laws security of personal data is an even bigger problem, so make sure your product is GDPR compliant.

DESIGN MULTI-TENANCY WITH PERFORMANCE AND SECURITY IN MIND

What is special about SaaS websites is the fact that they are self-service. This means that anyone who is interested in a given product can register and start using the service straight away. Users should be able to customize the solution to their requirements and needs.

Multi-tenancy means that a single instance of a software application is meant to serve multiple customers and it the key to the success of your system. This is achieved either through separate databases or one database that displays adequate information to particular users. Thanks to that the development will be faster since developers can make use of a previously written code base to expand the service and apply changes. Multi-tenancy also means that the application should be very secure since the infrastructure is shared among all users.

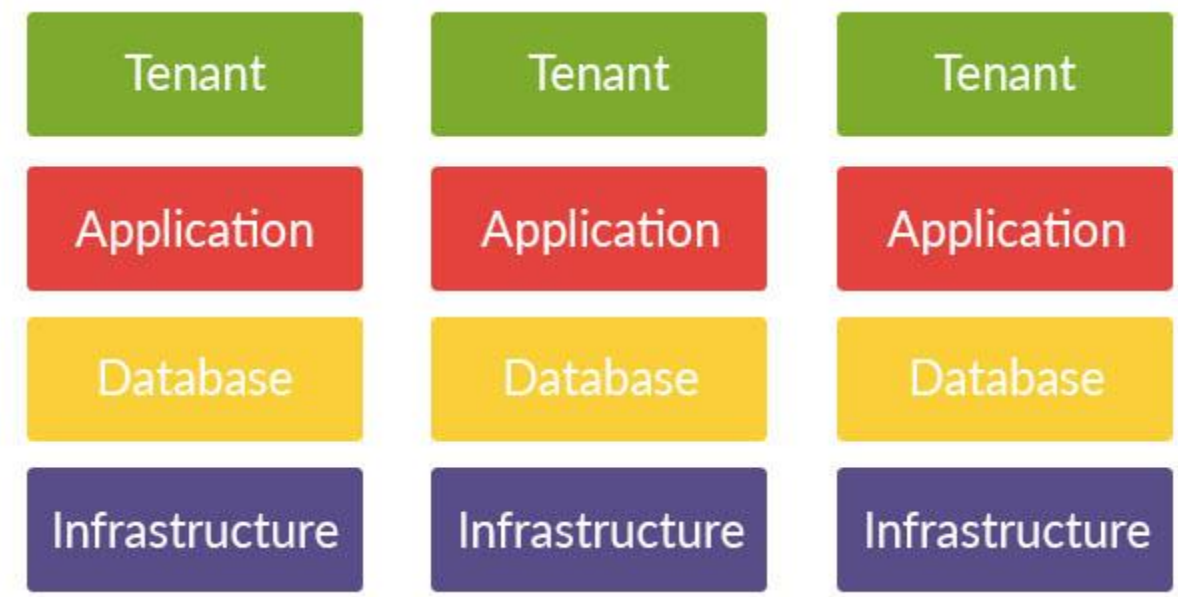
If you want to find out more about multi-tenancy and why you should choose it, read our article Why choose multi-tenant architecture for SaaS application.

Application's multi-tenancy can happen at different layers of the system:

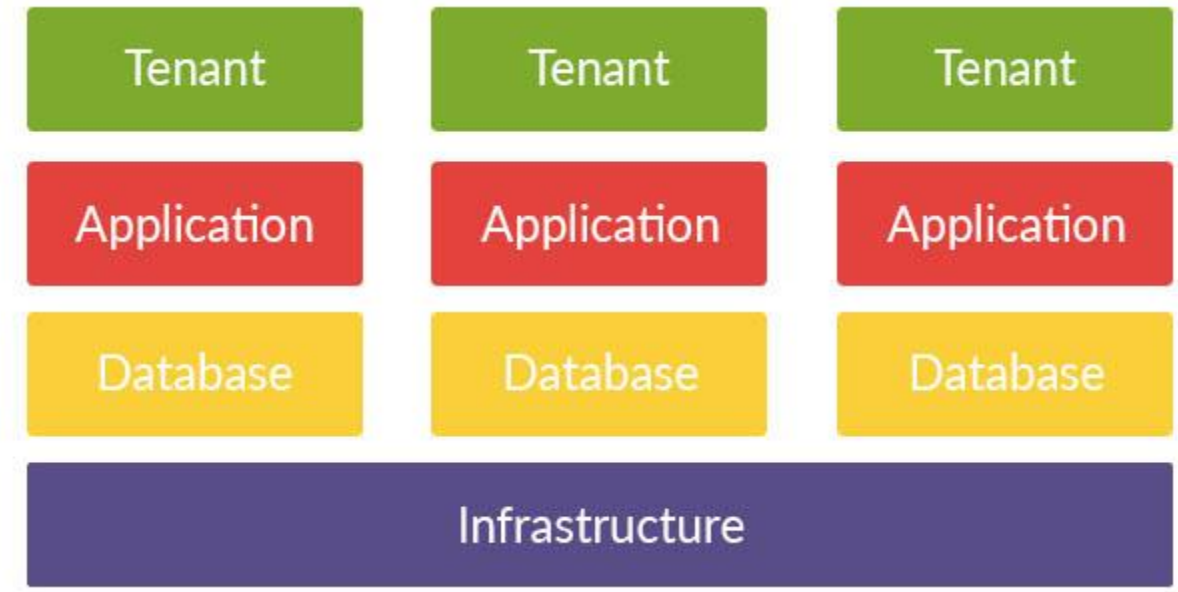
1. Infrastructure Layer
2. Database Layer
3. Application Layer

Based on those layers, we distinguish four types of multi-tenancy models that you can use to architect your SaaS application.

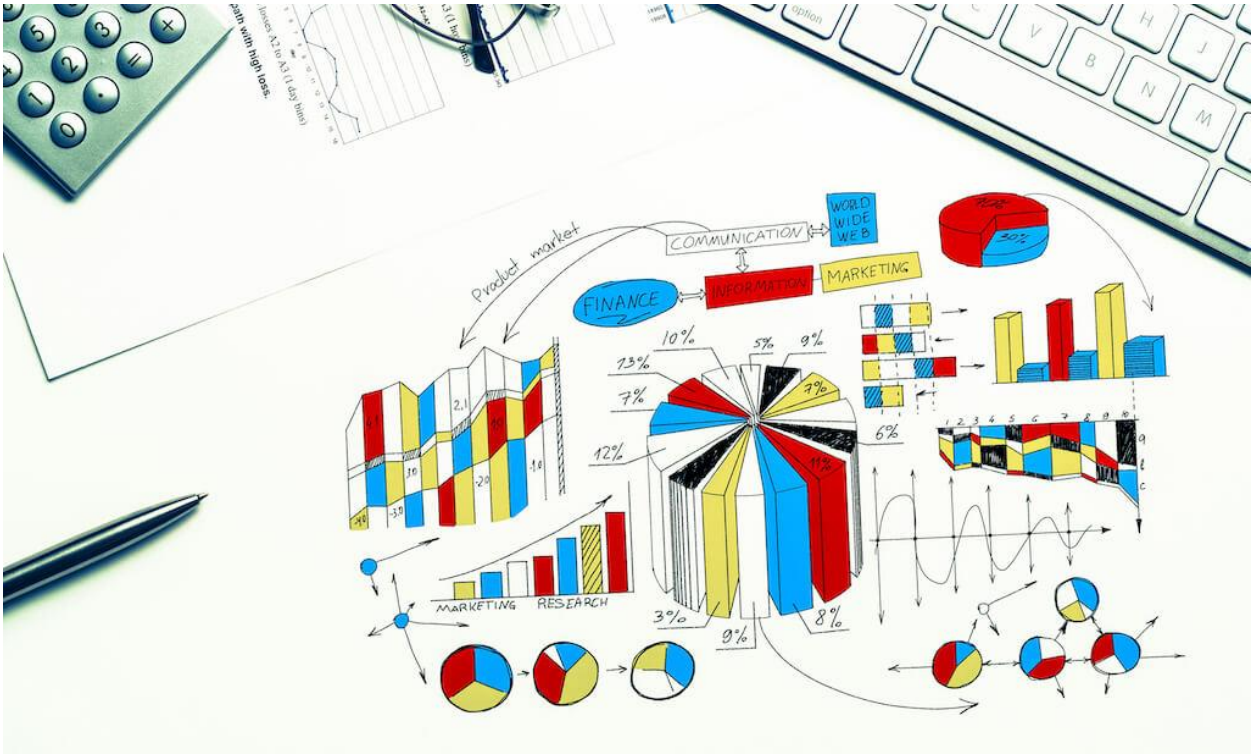
- Isolated Tenancy: the most basic level of tenancy where none of the layers are shared among the tenants. Every tenant has its own infrastructure, application and database. The infrastructure is isolated.



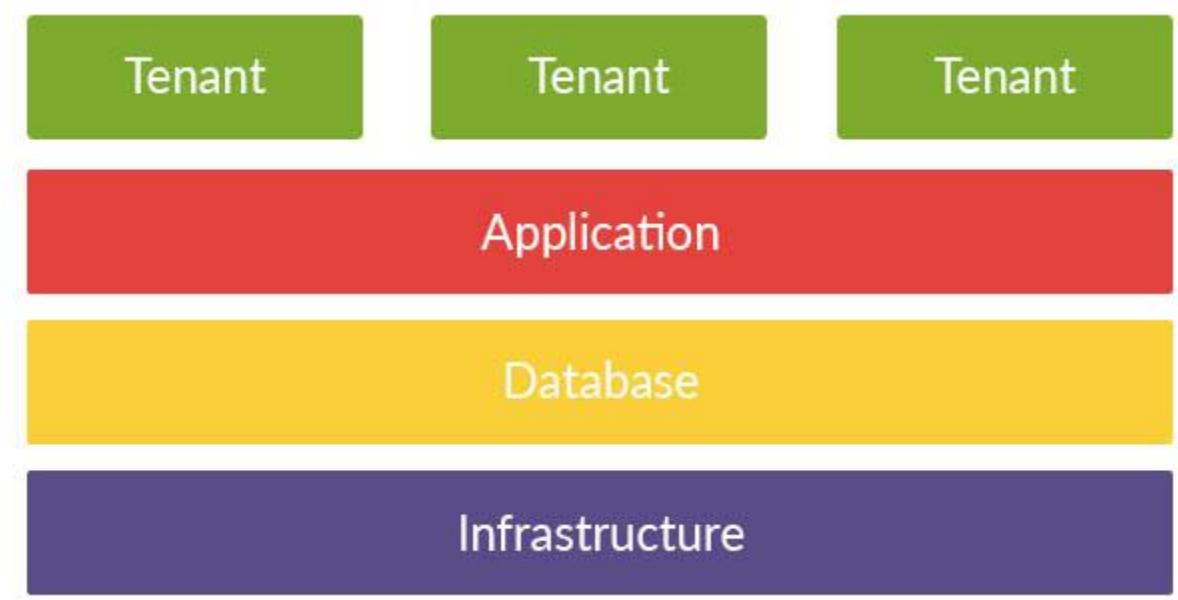
- Infrastructure Tenancy: here the infrastructure of the application is shared across tenants while the application and database remain separate.



- Application Tenancy: the application code and infrastructure are shared among tenants while the database remains separate.



- Shared Tenancy: in the last tenancy model, infrastructure, database and application are shared among the tenants but each tenant in the database is considered to be separate.



Your SaaS application should also be able to integrate with other services, for example, online payment solutions. Since your system will be available online it should be highly performant. An application with good performance should always be available and able to serve tenants of any size.

CHOOSE TECHNOLOGY STACK

Selecting a technology stack and finding SaaS developers is as important as market research. You need to have several tools for developing both client-facing components like JavaScript frameworks, server-side components like Ruby or Node.js and data storage with MySQL or PostgreSQL.

Estimate platform's scalability and potential profits to chose one that aligns with your use cases the best.

Then it is time to find developers that will help you bring your idea to life. Do not hesitate to ask for their opinion about the selected technologies. Bear in mind that the new technologies will have a limited talent pool with a rather fresh community around it.

CHOOSE YOUR PRICING STRATEGY

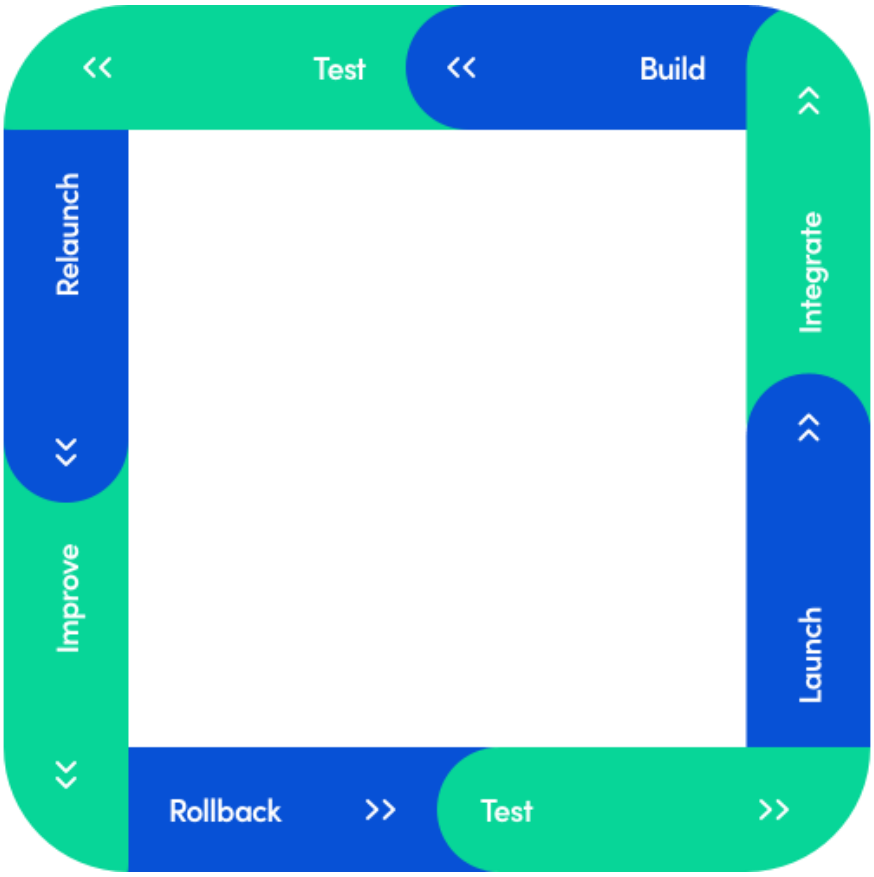
The pricing strategy can be the make or break it for your cloud-based application. There are quite a few options available like:

- Freemium
    - your app provides standard functionalities for free and can be expanded to more premium paid features. This model allows users to get to know the application, but remember to provide enough functions to attract customers while not being too generous
- Subscription
    - similar to freemium, this model is based on a recurring fee (eg. on a monthly basis) for extra content. It is best suited for content-focused applications like video hosting app
- Paid applications
    - your application is available for a one-time purchase. The advantage is that even if the user stops using your product, you still get the money. On the other hand, customers will be less willing to pay the upfront costs if they are not sure of its quality and usefulness

LAUNCH, TEST AND IMPROVE

When your application is ready to be released it is time to test it in-house before releasing. If there are no bugs and your product is working as required you can launch it, therefore, converting the first users into testers. Apply feedback and move on to adding new features. This process is called the iterative and repetitive process.

Launching a new product is connected to a lot of marketing through social media, online communities and advertising. You can also start your own blog.



You should track metrics, even if it sounds boring and time-consuming. Find out which ones are relevant to your business and keep an eye on them. This will give you the opportunity to improve the performance, resolve problems as quickly as possible and help you make decisions with more confidence.

ASP.NET VS NODE.JS

The larger the project the higher the price business owners pay for poor technology decisions. To avoid mistakes, CTOs and tech consultants need to consider dozens of factors and compare multiple options. For backend, enterprises frequently consider Node.js to be a safe choice for their server side needs.

ASP.NET

ASP is referred to as active-server-pages and .Net framework is mainly a software framework built by Microsoft which operates on windows.

ASP.NET is an open-source web application framework built by Microsoft and based on C#.

ASP.NET offers a wide range of web forms MVC, HTML5, JavaScript, CSS, Templates & many more to create highly-functional and dynamic websites and web apps.

Developers can create highly-intuitive, engaging and reliable websites using ASP.NET against different frameworks in the industry like Java.

Pros

- It offers customization in the data storage where you can easily distinguish identity information storage and security information.
- NET is assisted up by Microsoft, and being one of the leading companies in the globe, it ensures better safety and security.
- It is a lot simpler to maintain & update, which results in an affordable and highly-competent framework.

Cons

- The initial compilation will consume ample of time.
- One of the most tedious and daunting work is to upgrade older apps of the ASP.NET framework.
- Third-party integration is somewhat tricky.

NODE.JS

It is basically a JavaScript runtime environment that allows developers to execute JS code on the server-side. Being an open-source platform, it helps in the formation of real-time web applications.

Due to its asynchronous architecture and event-driven I/O API calling, developers can recognize JS code using the Chrome V8 engine.

Pros

- Although it's single-threaded, it consists of power to cater to the demands of multiple core servers instantly.
- It is backed up by an active and vibrant community of developers across the globe.
- Packaging modules are extensive and various new modules are released every month.

Cons

- Due to its asynchronous nature, the massive number of developers find it challenging to grasp and implement.
- It doesn't consist of a robust library, so developers are forced to integrate third-party libraries into their applications.
- For one who doesn't clear basic concepts in Java faces a lot of issues in Node.js development.

ECOSYSTEM: JAVASCRIPT RUNTIME VS .NET FRAMEWORK

Created in 2009 by 29-year-old Ryan Dahl, a Californian software engineer, Node.js is an open-source runtime environment that uses Google's v8 engine for executing Javascript (JS) code at the server side. In other words, it enables developers to write the back-end code in a primarily front-end language. Over the years, JS engineers have created a lot of impressive Node.js frameworks to streamline building apps.

ASP.NET is also an open-source technology, but in contrast with Node.js, it is a server-side framework that contains libraries, tools, and everything developers may need to build an app. The technology, introduced by Microsoft in 2002 allows the creation of dynamic websites and web applications with .NET languages, such as C#, F#, and Visual Basic. Though used in different types of projects, C# is considered to be particularly beneficial for building Windows desktop apps and games.

SUPPORT: CROSS-PLATFORM VS WINDOWS-CENTRIC

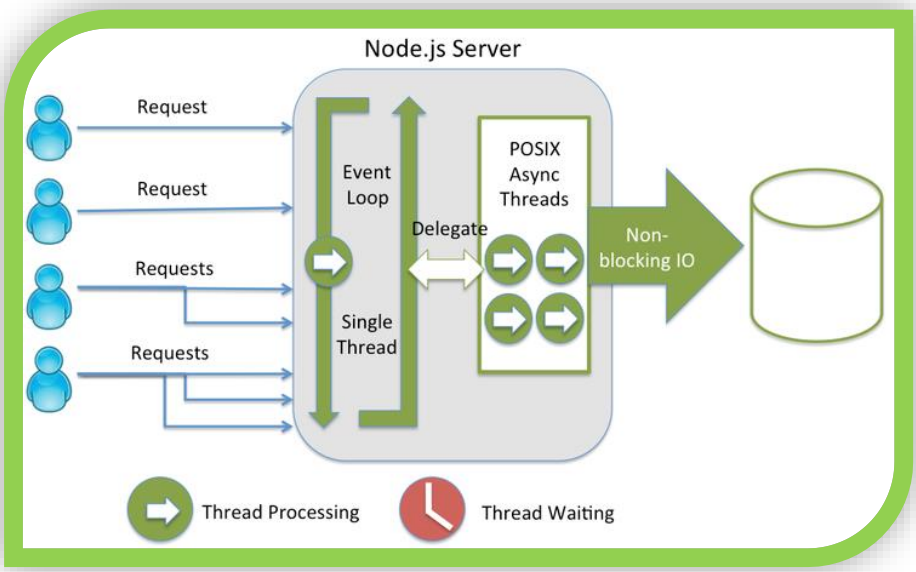
The ability to run on multiple platforms is one of the strong points of Node.js. It officially works on Linux, macOS, Microsoft Windows, SmartOS, FreeBSD, and IBM AIX.

Contrarily, ASP.NET was for a long time limited to the Windows ecosystem. This started changing in 2016 with ASP.NET Core, a redesigned version of the framework running on Linux and macOS as well. The first stable release of the cross-platform technology appeared in December, 2018.

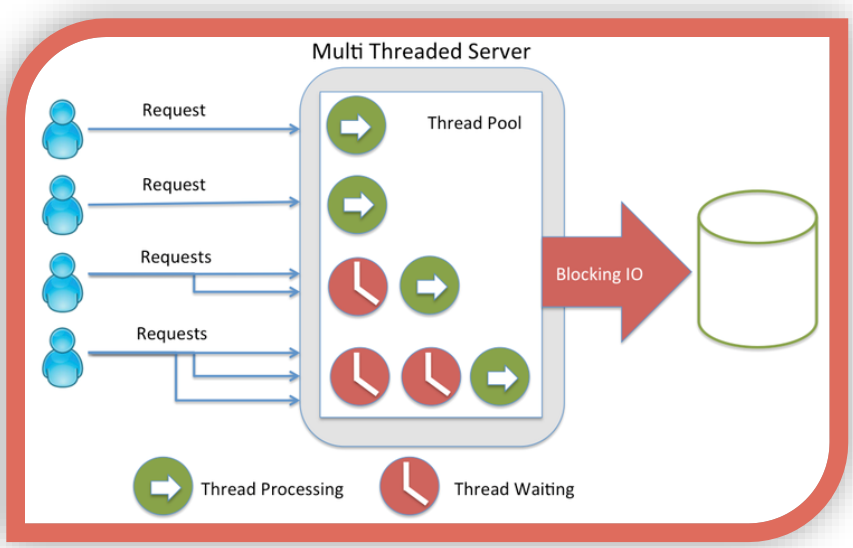
PROCESSING MODEL: ASYNCHRONOUS VS SYNCHRONOUS

To compare ASP.NET vs Node.js, we need to take into account their processing models which differ widely from each other. While JavaScript and, thus, Node.js are asynchronous by nature, ASP.NET offers developers a choice between two options.

An asynchronous model permits the server to handle multiple requests at a time without blocking any of them. Node.js executes async code in a single main thread, however, it spawns multiple background threads to delegate some tasks. This approach allows developers to create lightweight and efficient solutions when building data-intensive real-time apps.



ASP.NET is synchronous and multi-threaded by default, which means that it sequentially processes each request within its own thread inside the thread pool. If the number of tasks exceeds the available threads, they wait until a thread comes free. Theoretically, you can create more threads, but in practice such a solution is not always advantageous, as it requires additional system resources (CPU, memory, etc).



In a nutshell, an async process is more complex and slower when compared to an equivalent sync process, but it allows the system to avoid deadlocks and utilize resources more efficiently. The asynchronous model pays off if your server processes long-running tasks and a high volume of traffic while you'll probably see no difference with an app handling a small amount of requests.

ASP.NET VS NODE.JS BENCHMARK

Which of the two technologies would be a better fit for enterprise application software (EAS)? Now let's check Node.js vs ASP.NET against these major attributes.

Performance

Both back-end technologies are known for their excellent productivity and speed. As you can see from the above graph, 48% of companies point out the increased app performance when switching to Node.js. The JS back-end solution takes advantage of the fast Chrome V8 engine and due to the one-threaded asynchronous model, it can handle many requests at a time without weighing down the server.

However, the results of TechEmpower tests, dated 30th of October, 2018, show that ASP.NET Core wins the Node.js vs ASP.NET performance battle.

Of course, this victory is not final and we should take it with a pinch of salt, as performance vastly depends on your specific use case. According to SSA Backend, Node.js usually handles requests faster when no heavy computation is required, otherwise, it loses to ASP.NET.

The platform and environment also impact performance. For example, Sébastien Ros, one of the ASP.NET developers at Microsoft, states on Twitter that for ASP.NET Core 'Windows tests are faster than their Linux counterparts on the same hardware by 20%, sometimes more'. Anyway, the redesigned cross-platform version is much faster than the old Windows-based framework.

Scalability

An increasing number of enterprises prefer to build their software on top of microservices. In contrast with the traditional monolithic architecture, a microservices-based software allows each component to scale independently, and as a result, the app can grow faster without falling apart under its own weight. Node.js is a perfect fit for such

distributed systems, and that's why it was adopted by actively growing companies like Netflix, eBay, Uber, Twitter, and more.

ASP.NET is also highly scalable and Stack Overflow is great proof of this fact. The question and answer website for programmers with over four million registered users is written in ASP.NET and based on monolithic architecture.

Reliability and robustness

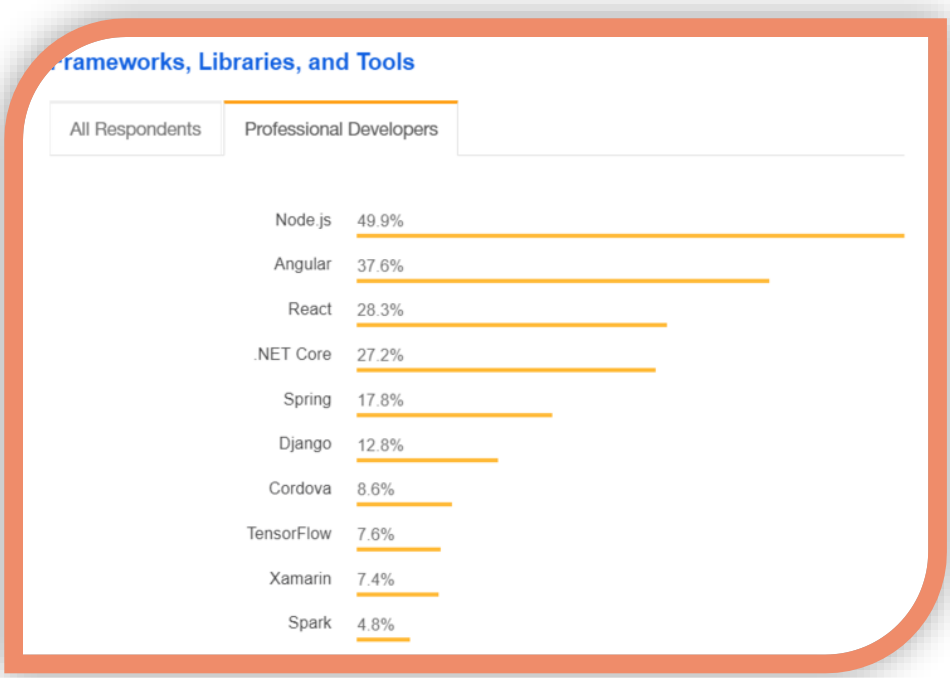
ASP.NET is highly valued by the corporate sector due to its proven reliability and security. The platform provides developers with whatever is required to build solid software and supports the statically-typed C# language known for its robustness.

Quite the opposite, JavaScript is often criticized for lacking static types and strictness, which are essential for complex enterprise-grade applications with several million lines of code. One of the popular ways to solve this problem is to add optional static typing to Node.js by using TypeScript, a strict superset of JS.

Currently, ASP.NET remains well ahead of Node.js in terms of market share, and this gap is especially large when it comes to websites with over a million users.

Availability of developers

The above-mentioned Stack Overflow survey reveals that JavaScript tops the list of most popular programming languages among professional developers, with C# coming in eighth place. At the same time, .Net Core is the fourth most loved technology in the 'Frameworks, Libraries and Tools' category while Node.js is in the lead.



Both Node.js and .NET Core enjoy support from large communities of engineers working for small, midsize and enterprise-level businesses. They are active on GitHub and Stack Overflow which means that you can solve practically any issue with the help of the community.

Criterion	Node.js	ASP.Net	Recommended
Suitable	Node.js is suitable for building asynchronous apps as it functions on one thread via non-blocking I/O calls.	ASP.NET operates simultaneously on multi-threads.	ASP.NET
Benefits	Node.js share one language on the server and the client-side. You can also expand apps in both horizontal and vertical direction. Lastly, it is appropriate for developing an MVP (Minimal Viable Product).	It offers highly-potent because of its added advantage of early-binding, just-in-time compilation, native optimization, and caching services.	Node.js
Real-Time Usage	It can maintain & control over a massive amount of customer data.	Numerous websites that we are accessing on the web are actually made via ASP.NET.	ASP.NET
Language	It is based on JavaScript. To integrate JavaScript into the project, you have to become an expert on asynchronous programming, which requires some time.	It is based on C#. ASP.NET provides a strict type system and compile-type error checks, which is dependent on Microsoft's TypeScript.	ASP.NET
Speed	Being asynchronous in nature, it can handle several callbacks consequently. Developers can access native & third-party libraries in their web development projects.	Manual configurations are challenging and consumes some more time.	Node.js

Support	Node has more than 70K+ stars & 170+ repositories on the GitHub. However, it has low support on StackOverflow.	It has a broader community of developers on the StackOverflow as compared to the GitHub.	Both
Availability	It consists of a wide number of small and reusable libraries, which helps you to achieve more by writing less code.	The most recent ASP.NET core is considered as a highly scalable framework with some unique solutions that increases the development speed.	Node.js
Tools	Node.js supports the majority of text editors in the market. However, a standard option is to use WebStorm (IDE)	Developers prefer to use tools such as Visual Studio, Web Essentials, and Resharper.	Node.js
Hosting	Majority of developers choose to host by forming their own Linux Web Server.	Options like AWS, Heroku, Microsoft Azure, and Google Cloud Platform.	ASP.NET
Performance	Node.js is capable enough to handle multitasking with ease as it works on high-performing JavaScript engine V8. Also, it can tackle more traffic without any requirement of the external server.	The most recent version of the multi-platform .NET core is 15% faster than its earlier versions and highly-potent than ASP.NET.	Node.js
Ecosystem	Developers are more drawn to Node.js as it utilizes a V8 engine to write back-end code using the front-end language.	It is a server-side framework that offers a comprehensive bundle of libraries, tools, etc. which you can use to create apps efficiently.	ASP.NET
Processing Model	It is basically a server-side framework that offers a comprehensive bundle of libraries, tools, etc. which you can use to create apps efficiently.	It works on the synchronous model and is multi-threaded, i.e., it successively handles all the requests present in the thread pool.	Node.js
Traction	Over the last few years, many tech giants across the globe have started adopting Node.js for their development projects.	It has obtained a respectable position in the market; however, we haven't observed big companies using ASP.NET for their project.	ASP.NET
Available Package	npm is the package manager for Node.js	NuGet is a package manager for ASP.NET	Node.js
Platform Autonomy	cross-platform development support	bound to use the Windows system	Node.js
Reliability	Developers find it challenging to handle errors in Node.js due to its asynchronous nature.	It remains more powerful during any errors due to its strict type system.	ASP.NET
Learnability	any beginner can start with Node.js quickly. However, with time, to develop a high-quality code, one must understand different things like promises, generators, and co-routines.	consists of a built-on project/file structure, boilerplate, and a wide number of examples on how to utilize MVC patterns for their project.	ASP.NET
Portability	Neglecting some of the issues, Node.js is portable on most of the platforms.	It is partially portable. For instance, MVC 4 on Mono doesn't have support for async features.	Node.js

Memory Management	ode.js runs on the V8 engine, which has a memory limit of 512MB on 32bit systems and 1GB on 64bit systems. You can extend this limit by using one command to 1GB for 32bit versions and 1.7GB for 64bit versions.	memory management becomes hard when you cross a 2GB limit on the 64bit machine; however, sources are limited.	ASP.NET
Concurrency	It uses a single CPU-thread and is not able to take benefit of multiple-threads automatically. To utilize multiple threads, you have to depend on clustering.	It, by default, can utilize multi-core systems with the help of the Thread pool.	ASP.NET
Development Time	In Node.js, all the things are being developed as well as maintained by different developers. All these are built, keeping in mind different ideas, standards, and requirements. Hence,.you might have to refer to the documentation now & then.	In ASP.NET, the majority of required plugins, modules, NuGet packages are developed by Microsoft. Due to this, there are very rare chances of any deletion of modules or change of names.	ASP.NET
Stability	If you have used some modules in your project and you decided to update it, but all of a sudden, you get to know that they are being outdated or renamed. This has a significant impact on the project's stability.	In ASP.NET, the NuGet packages are released as well as maintained by Microsoft. Hence, you have assured that you're in safe hands.	ASP.NET

SAAS DATABASES

Designing the database for SaaS web application, following factors are considered, not necessarily ranked according to importance:

- Scalability of the proposed model
- The cost of implementing the database design
- The complexity of managing and customizing the database
- Data isolation, or the impact of a tenant's workload on the application's overall performance

Database models that can be used for SaaS application in terms of these factors.

- ✓ **Model 1:** Separate Application and Database Instances for Each Tenant
- ✓ **Model 2:** Multi-Tenant App with One Database per Tenant
- ✓ **Model 3:** Multi-Tenant App with a Single Multi-Tenant Database
- ✓ **Model 4:** Multi-Tenant App with Sharded Multi-Tenant Databases

Table below summarizes models and their corresponding ratings based on the four factors mentioned earlier in the article.

Factor/Model	Model 1: Separate Application and Database Instances Per Tenant	Model 2: Multi-Tenant App with One Database per Tenant	Model 3: Multi-Tenant App with a Single Multi-Tenant Database	Model 4: Multi-Tenant App with Sharded Multi-Tenant Databases
Scalability	Medium	High	Medium	Very High
Cost	High	High	Low	Medium
Complexity of development and management	Low development, medium to high management	Low development, low to medium management	Low development, low to medium management	Medium
Data Isolation	Very High	Very High	Low	Low

An appropriate design model for the application database that is able to handle the initial expected workload is thus required. This model should also account for ease of migration in the face of increasing workloads as the application becomes popular and attracts more users. Following are some data base options offered by Microsoft Azure:

AZURE SQL DATABASE

Azure SQL Database is a relational database-as-a service using the Microsoft SQL Server Engine. SQL Database is a high-performance, reliable, and secure database you can use to build data-driven applications and websites in the programming language of your choice, without needing to manage infrastructure.

Azure Services	1-month pricing	6-month pricing	1-year pricing
Azure App Service	$54.75	$328.50	$657.00
Azure Cognitive Services	$0.00		
Azure SQL Database	$1,476.43	$8,840.17	$17,676.66
Total	$1531.18	$9,168.67	$18,333.66

MYSQL DATABASE

Azure Database for MySQL is a fully-managed database service for app developers and it is capable of handling mission-critical workload with predictable performance and dynamic scalability.

Azure Services	1-month pricing	6-month pricing	1-year pricing
Azure App Service	$54.75	$328.50	$657.00
Azure Cognitive Services	$0.00		
MySQL Database	$138.47	$777.95	$1545.33
Total	$193.22	$1,106.45	$2,202.33

POSTGRESQL

Azure Database for PostgreSQL is a managed database service for app developers that integrates the PostgreSQL community edition with Azure for scalability, high-availability, and your choice of languages and frameworks.

Azure Services	1-month pricing	6-month pricing	1-year pricing
Azure App Service	$54.75	$328.50	$657.00

Azure Cognitive Services	$0.00		
PostgreSQL	$138.47	$777.95	$1545.33
Total	$193.22	$1,106.45	$2,202.33

MARIADB

MySQL application compatible open-source RDBMS, enhanced with high availability, security, interoperability and performance capabilities. With MariaDB ColumnStore a column-oriented storage engine is available too.

Azure Services	1-month pricing	6-month pricing	1-year pricing
Azure App Service	$54.75	$328.50	$657.00
Azure Cognitive Services	$0.00		
MariaDB	$138.47	$777.95	$1545.33
Total	$193.22	$1,106.45	$2,202.33

ARTIFICIAL INTELLIGENCE/MACHINE LEARNING CLOUD MODEL

ARTIFICIAL INTELLIGENCE AS A SERVICE (AIAAS)

Artificial Intelligence as a Service - AIaaS allows individuals and companies to experiment with AI for various business purposes and sample multiple platforms before making a commitment. Because hardware, software and staffing costs for AI can be expensive, many vendors are including AI components in their standard offerings or providing access to artificial intelligence as a service (AIaaS) platform.

The top cloud computing platforms are all betting big on democratizing artificial intelligence. Over the past three years, Amazon, Google, and Microsoft have made significant investments in artificial intelligence (AI) and machine learning. The cloud makes it easy for enterprises to experiment with machine learning capabilities and scale up as projects go into production and demand increases. AWS, Microsoft Azure, and Google Cloud Platform offer many machine learning options that do not require deep knowledge of AI, machine learning theory, or a team of data scientists.

MACHINE LEARNING AS A SERVICE (MLAAS)

It is an umbrella definition of various cloud-based platforms that cover most infrastructure issues such as data pre-processing, model training, and model evaluation, with further prediction. Prediction results can be bridged with your internal IT infrastructure through REST APIs.

Amazon Machine Learning services, Azure Machine Learning, Google Cloud AI, and IBM Watson are four leading cloud MLaaS services that allow for fast model training and deployment.

AI/ML TOOLS FOR CLOUD

The main offerings in this category involves the following:

	Amazon	Microsoft Azure	Google

Image Recognition	Rekognition Image	Computer Vision API Custom Vision Service Face API Emotion API Content Moderator	Vision API AutoML Vision
Video Analysis	Rekognition Video	Computer Vision API Video Indexer Content Moderator	Video Intelligence
Speech to Text	Transcribe	Bing Speech API Custom Speech Service Speaker Recognition API	Speech API
Text to Speech	Polly	Bing Speech API	Text-to-Speech API
Translation	Translate	Translator Text API	Translation API
Language Analysis	Comprehend	Text Analytics API Content Moderator Language Understanding Web Language Model API Linguistic Analysis API	Natural Language API
Chatbot	Lex	Azure Bot Service	Dialogflow

SAAS API ARCHITECTURE

The smartest organizations have discovered a set of best practices to design powerful APIs that leverage existing services, to effectively manage those APIs throughout their lifecycle and to scale their deployment across consumers and devices.

WHAT DRIVES API ADOPTION

Organizations are beginning to understand the importance of APIs and the value they deliver to the business. But what exactly is causing all the fuss?

Business agility – Much more effective than creating business logic and exposing it as a website is creating business logic and exposing it as an API. With APIs, it doesn't matter how the logic is used; clients and consumers can consume the API and expose it to their liking, delivering

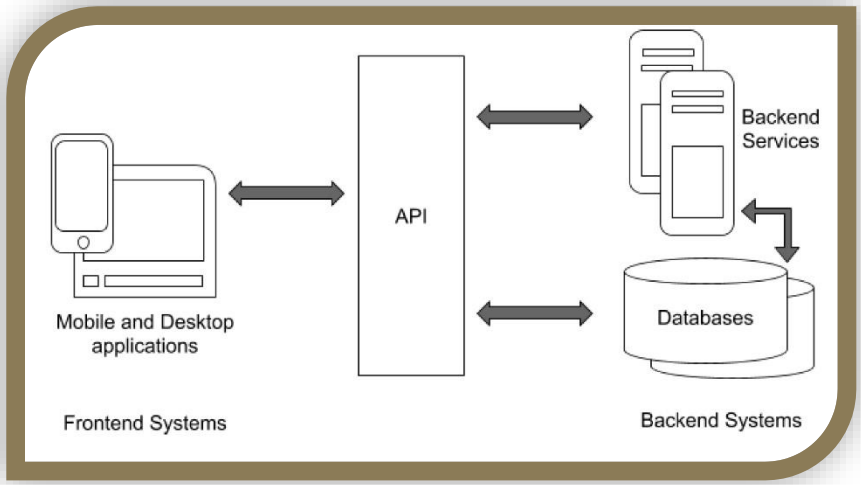
API economy – Businesses are developing "API products" as new sources of revenue. Expedia generates over $2 billion annually through the data made available through their API. Salesforce generates 75% of their revenue through API activity. The API economy is here and growing fast.

Internet of things – The number of things that can be connected is growing fast; everything from your coffee pot to your thermostat to your car can now opportunities to create connectivity in new places.

API-FIRST DESIGN APPROACH

This method allows organizations to isolate concerns and to focus on clear, API-first development provides a simple framework that enforces better REST API design practices, helping businesses focus on API resources and messaging before undergoing the heavy lifting and back-end implementation.

This approach drastically reduces project delays and cost overruns due to miscommunication between frontend and backend teams leading to changes in APIs and backend systems.



CHOOSE AN API RUNTIME

Once an API is designed, created and its back-end implemented, it's time to successful API strategy will be in terms of service, liability, scale and ability to meet future needs. So, what key capabilities should we look for in an API runtime?

Hybrid support: This allows businesses to burst into the cloud when extra resources are needed or even shift from on-premise to the cloud. The ability to develop applications once and deploy them in the cloud or on-premise provides a host of possibilities without complexity.

Scalability, reliability, availability: These "-ilities", along with performance, are crucial when searching for a solid API runtime. Choosing the right enterprise-grade technology for API runtime is crucial to the success of API.

Strong Orchestration: A successful runtime should provide strong orchestration and orchestration capabilities. The ability to carry out complex back-end orchestration plays a key role in translation between the API layer and back-end implementation.

CENTRAL SERVICE REPOSITORY IS A MUST

After designing, developing, and running the API on a solid platform, exposing the API in a central repository is paramount. Surfacing the API facilitates API discoverability and accessibility by people who need it. A central service repository also makes it easy to categorize and search through services while providing a consolidated view of APIs; it is the place for all design and runtime governance.

Forums and documentation make it easy for developers to assess an API's visibility into key metrics of APIs, business owners and architects know how an API is performing so they can take corrective action and continue to improve the API in future versions.

VERSIONS, POLICIES, AND CONTRACTS

Tracking service versions and consumers gives businesses insight into who is using APIs, which versions they're using and how they're using them. This helps with API lifecycle management and allows API publishers to assess the impact

of retiring a version. In addition, policies and contracts are crucial to enforcing security and managing SLAs with API consumers. An API solution should offer a way to easily create associate them with the right APIs and consumers.

API USAGE MONITORING

Understanding which parts of service are being used is important, but it's only part of the picture. With metrics for both overall usage and per consumer usage, businesses can closely monitor API activity and engagement. The ability to monitor API both technical and business perspectives is valuable as it helps business owners and technical teams better understand their users and ultimately create a better service.

WIDENING API AUDIENCE

Creating a developer portal to establish a community around API is important to its success. By making it easy for users to follow the API, download documentation and ask questions, API publishers can engage with API consumers on an ongoing basis. Providing and promoting content helping the API be successful.

ALWAYS IMPROVE

The ability to refactor APIs by iterating through habits 1-6 multiple times allows to optimize API over time to improve consumer experience and productivity.

SAAS FRAMEOWRKS

ABP

ABP Framework is a complete infrastructure based on the ASP.NET Core to create modern web applications and APIs by following the software development best practices and the latest technologies.

Best For: IT decision-makers, software developers, and all enterprises use Microsoft .NET technologies, building ASP.NET Core based applications in particular.

Framework: https://abp.io/

Docs: https://docs.abp.io/en/abp/latest/Getting-Started

GitHub: https://github.com/abpframework/abp

SAASKIT

SaasKit is a .NET toolkit for building SaaS (Software as A Service) applications.

The goal of the project is to help developers build SaaS products without getting in the way. It aims to be platform agnostic and as simple to use as possible.

GitHub: https://github.com/saaskit/saaskit

FINBUCKLE.MULTITENANT

Finbuckle.MultiTenant is an open-source multitenancy library for .NET. It provides for tenant resolution, per-tenant app behaviour, and per-tenant data isolation.

Supported .Net Frameworks: .NET 5.0, .NET Core 3.1, and .NET Core 2.1

Framework: https://www.finbuckle.com/MultiTenant

GitHub: https://github.com/Finbuckle/Finbuckle.MultiTenant

CLOUDSCRIBE

cloudscribe is a related set of projects and components for building cross platform web applications on ASP.NET Core.

The foundational set of projects in this repository, known as cloudscibe Core, provides support for single tenant or multi-tenant management of sites, users, and roles.

Framework: https://www.cloudscribe.com/introduction

GitHub: https://github.com/cloudscribe/cloudscribe

REFERENCE CC SAAS: EDCONVERGERENCE

EdTech Strategy for Education Industry Practice is a platform for students and parents where they can communicate with each other for various purposes and use that platform in their regular lifestyles. Parent-Teacher Communication Platform is one of its concepts by which the educational sector can build a bridge between teachers and parents to improve their communication.

Our solution is a package of all those techniques and tools that accelerates the development of students not only at school but at home as well. Our Parent-Teacher Communication Platform excels in achieving the maximum result by having the positive support of both parents and teachers. It connects teachers with students and parents to build progressive classroom communities. Our Educational product develops a positive sign of cooperation between parents and teachers which are linked with just a single click

REFERENCE CC SAAS: PSYCHECONVERGERENCE

Psychologist Solution Practice provides a personalized Dashboard for Psychologists, Counselors, Therapists, Psychiatrists, and Social Workers, where you can see your today's schedule, who's coming in, what they are coming in for, calendars, a balance due, etc. You can easily set up new appointments, send an automated text and email as appointment reminders, view future appointments against a client, create a profile for new clients, add prospects to a waiting list, create therapy notes, or even take payments, in one single solution.

Our solutions are made simple yet smart, scalable, and secure with easy-to-use features such as Appointment Reminders, Client Portal, Scheduling, Therapy notes, Telehealth Video Conferencing, Credit Card Processing, Billing, Income & Expense Tracking, etc. and many other business management tools in a cloud-hosted environment. We provide the perfect visualization for every task at hand. At Cognitive Convergence, we work hard to provide such solutions to our clients that help them to maintain and grow their practices as well as meet the needs of their clients at the same time.

CONCLUSION

Our Microsoft Office 365 Consulting, add-in Development, Customization, Integration services, and solutions, can help companies maximize business performance, overcoming market challenges, achieving profitability, and providing the best customer care service.

CONTACT US

Shahzad Sarwar
Cognitive Convergence
http://www.cognitiveconvergence.com