The Technology of Digital Customer Engagement

Contents

1. Summary……….page 1
2. Introduction………page 1
3. The Technology Stack……………………………………………………………………………………………………. page 3
4. Finding the Best Technology Stack – why is it important?..page 3
5. Technology Stacks, The Key Types………………………………………………………………………………………..page 4
6. Coding Languages……page 7
7. Development Environments, Tools and Frameworks……………………………………………………………………. page 9
8. The AwareX Technology Stack……………………………………………………………………………………….... page 12

1. Summary

At AwareX we use a wide range of the latest technology, code languages and development frameworks to produce the best possible 'Carrier Grade' Digital Engagement Platform. Just as an example these include, React, Bootstrap, Docker, Spring and Node.JS. In this document you will see we have considered a great many different approaches and tools to build our system, but we are not dogmatic or fixed on any one solution, rather we believe as a platform company in the right tool for the right job. We thus provide Apps which are Native and Web which is Reactive, we use the best of Native development and mobile web development tools to deliver the best possible technology stack to enable a preferred customer experience.

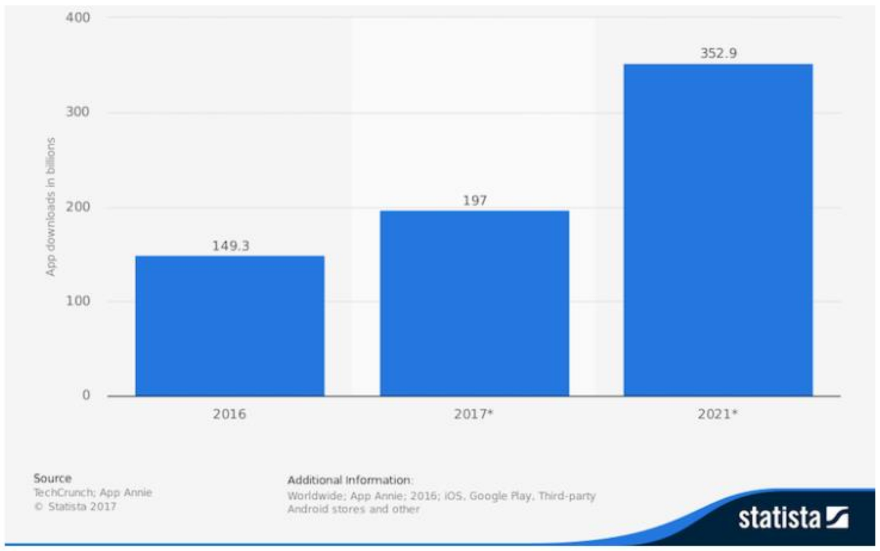
2. Introduction

Digital customer engagement and currently Mobile Apps are the best way to serve your customers, all demographics (except the over 65's) are now spending more time in Apps each day than any other form of communication. This is what makes awareX different, we have a customer-first philosophy which dictates our development is customer centric, our usability is customer centric and our approach to selecting the right technology to build and maintain our system demands customer focused attributes like speed, agility, stability, quality, security and maintainability. We maintain a single code base not a multitude of threads which are unsupportable and difficult to upgrade, awareX provides a steady stream of innovation to our customers whilst ensuring our technology is fit for purpose.

There are of course a large number of possible technology's in fact a huge number of different code languages, frameworks, development environments and methods to use as a product company when developing new platforms, frameworks and Apps. We will provide an overview of many of the technologies and state which ones we at awareX selected to build our product. There is however no generic 'right' technology which is better than all the others and the reason is that with over 352 Billion App downloads expected by 2021 the diversity of usage, type, geography, economy and social positioning means that a Buzz word technology that is perfect for one App can be completely wrong for another.

At AwareX our apps are designed, built, maintained and updated with technology appropriate for the communications service provider market. We have the product goal to deliver superior customer engagement, reduce a CSP's operating costs, increase revenues and improve the Net Promoter Score NPS.

Number of mobile app downloads worldwide, 2016, 2017 and 2021



Source: Statista

- Around 61% of global mobile phone users are reported to access the internet from mobile devices in the year 2018.
- According to comScore reports, individuals spend 50% of their media time on mobile applications.
- The total number of mobile app downloads in 2017 were 197 billion and it is expected to reach 352 billion by 2021.
- Almost 91% of smartphone users are reported to turn to mobile apps for business information and research.
- The year 2018 marks a massive increase in consumers spending on Apple App and Google Play stores by 22%.

Mobile apps have become the most convenient source of contact between businesses and customers. When choosing the best technology to create ultimate mobile apps, there are a multitude of options which we describe below. At awareX we will use the right technology to deliver superior digital engagement, end to end technology stacks and Apps.

AwareX delivers a product that works, is continually evolving and is low risk, utilizing the right technology as described in this document.

3. The Technology Stack

Firstly, our technology at awareX is not just about App development but rather the full technology stack. We use a technology which works ‘end to end’ to meet CSP’s business objectives, It is a strong and stable technology stack, designed to deliver the results you are aiming for. The future success of any mobile app is heavily dependent on the technology stack it uses.

The technology stack for mobile app development may be categorized into 4 aspects:

- Frontend Development

This is essentially the user interface or UI of the mobile app which enables end users to interact with the app.

- Backend Development

This is the part that is responsible for taking the user input, processing this input, integrating to multiple 3rd party systems of record, creating a total environment not just a user interface and then converting it into an output.

This is crucial for the effective operation of the end to end system. The AwareX architecture allows the Microservices Integration Platform to make functional and logic changes for live customer systems without affecting the clients (apps, web).

- Development Platform

This is the place where interfaces and libraries come together to design and develop the mobile application.

- Additional Requirements

These are mostly the non-functional technology elements that affect the performance of the app, scalability, security, high availability and robustness. In the Telecom world this is what we call ‘Carrier Grade’.

4. Finding the Best Technology Stack – Why is it important?

The Apps that you build will be defined by the selection of the technology stack. It governs what decides if the app is robust, will the app be able to scale to various versions of different operating systems? and many other such traits. It will not be Carrier grade if you get this wrong.

Choosing anything but the right technology stack is sure to affect the immediate performance of the app and undermine its future growth prospects. The time to market of the app and the cost of app development are only the most obvious and the initial issues you would face but choosing the wrong tech stack can actually lead to more deep-rooted problems in your app altogether! However, as we have already stated there is no generic ‘best’ technology stack, only the right stack for your particular business need, market and customer engagement situation. The awareX stack is intended to be the best for Communications Service Providers who want the best Digital Customer Engagement. As an example the awareX stack has been engineered to allow on-line, real-time changes in the microservices integration platform which provide new, additional and better information to end users without the need for an App or client new version release which is a critical feature for achieving ‘Carrier Grade’ as required by Communication Service Providers.

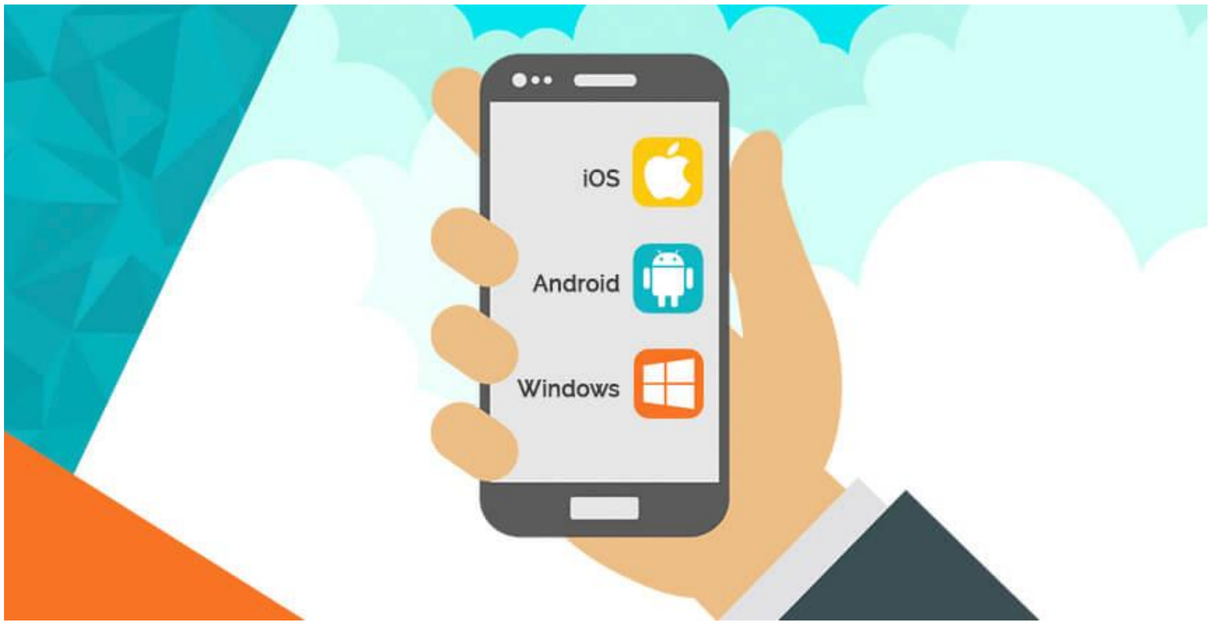
5. Key Technology Stack Types

This section lists the key types of technology stack you can choose from.

- **Native App Technology Stack**

Preferred by most companies for their robustness and high performance, Native Apps, present themselves as a near perfect solution when it comes to mobile app development for business. A native app lets the developers integrate the device's in-built functionalities in their mobile app without having to rely on any external third-party API.

When you think about a native app it is natural that iOS and Android apps would pop up in your head. Now, both the Operating systems are different so being native in each of them does require different constructs to deliver the same user experience in each native environment.



  - **Technology Stack for Android Mobile Apps**

The official programming language of android App development is called Kotlin, but the Android app developer community still prefer to use the Java language. When it comes to IDE, the most commonly used software for development purposes are Eclipse and Android Studio.

  - **Technology Stack for iOS Mobile Apps**

Swift & Objective-C are the most preferred languages for iOS app development. IntelliJ AppCode and Apple XCode are the most commonly used toolkits among iPad and iPhone app developers.

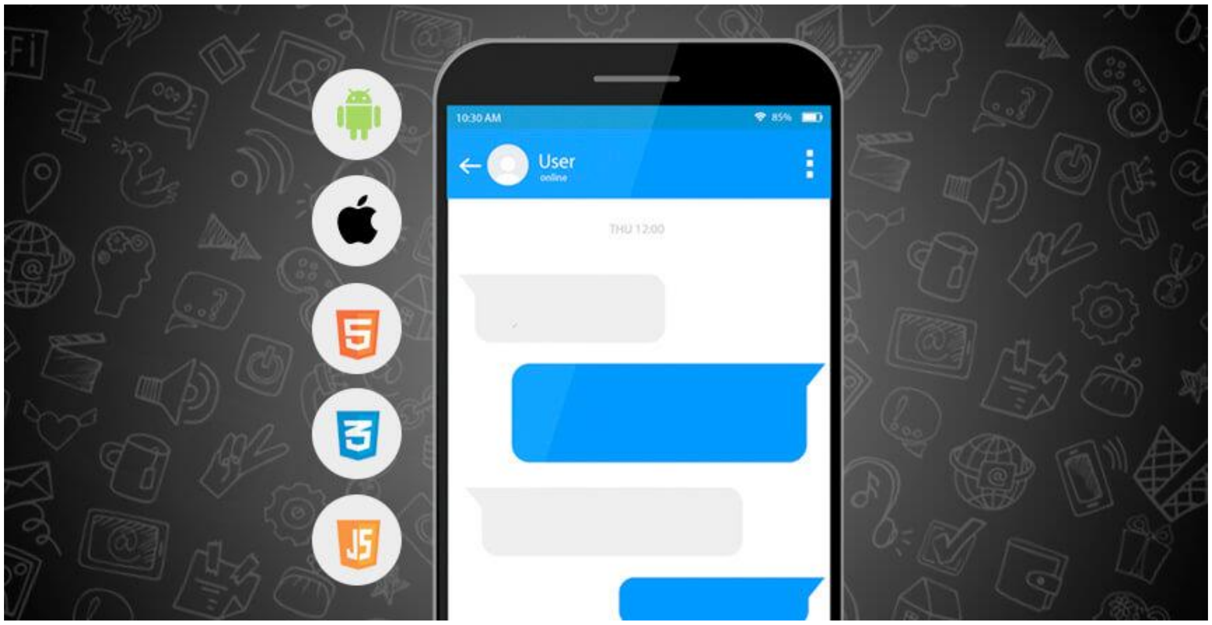
- **Hybrid Apps Technology Stack**

The technologies typically used for hybrid app development include CSS, HTML 5, and JavaScript. These apps are typically sectioned into two portions – backend and native shell.

The three most commonly used frameworks here are – Cordova, Sencha Touch 2, and Ionic.

- **Cross-Platform Apps Technology Stack (also called Mobile Web apps).**

Cross platform app development refers to the process of using the same code for app development that can be run on several platforms including Android, iOS, and the web.



There are three major technologies used in develop of cross platform technology stacks – Flutter, React Native, and Xamarin.

- **Best Technology Stack – How to choose it?**

It is important to be aware of the factors that influence the choice of technology stack.

- **App Considerations**

Each app is different from the others in many different ways. The devices on which the app would have to function, the kind of network conditions it would have for operation, the intended user experience, the go to market time, and the platform it would run on, and many other such factors differentiate one app development from the other.

All these elements and more, come together to help the developer decide which framework, library, language, and software they should use to attain the best results.

- **App Objective**

The objective or the end goal of the app is critical in the technology stack a developer picks for developing the app. For example, the technology stack would wildly differ for developing an app with high latency as compare to a quick response mobile app with low latency.

If the mobile app is dependent on heavy load processing, then there is a need for a more robust technology stack as compared to an app that operates around precisely streamlined interactions.

- **Developer Skill Set**

There are quite a few languages and frameworks that bear similar results, but have certain typical differentiating factors distinguishing one from the other. If you were to decide between two technology stacks which are close to each other, then the differentiator becomes the developer's skill set rather than the actual technology.

- **Parent Company (of the technology)**

The parent company of your chosen technology stack is also important in the development of your mobile app. Some of the more established brands are less 'trendy' but provide better documentation and community support as compared to the 'latest and greatest' which itself may offer a more innovative environment such as Facebook React Native for example.

- **Multi-Platform Operability**

If we compare an app that runs on a single platform to an app that would run on all the platforms, there would be a world of difference between the technology stacks chosen to build the two. Similarly, the tool set required for the kind of scalability you would require incorporating into your mobile app in order for it to be ported on to other platforms, like in Hybrid or Cross Platform app development would be entirely different from the tool set required for Native app development.

- **Security Considerations**

Each technology to a greater or lesser extent provides an inbuilt ability to ward off any weaknesses in the security that an app might encounter after it has been released in the market. However, it becomes tricky when the tech stack you chose is insufficiently robust or requires writing long codes thus increasing the security risks during the entire process of development. The use of static security code scanners to identify weaknesses from Veracode and NCC is highly recommended.

- **Compatibility**

It is of absolute importance to choose a technology stack which is compatible with the technology that you have as a back end or data source of record if your App requires real time data access to 3rd party systems. Communication Service Providers have a unique requirement to integrate with BSS a class of high transaction volume real time systems covering multiple service types, it is not a standard APP development requirement and understanding of the Telecom BSS environment confers significant advantage.

6. Coding Languages - Technologies to Develop A Mobile App

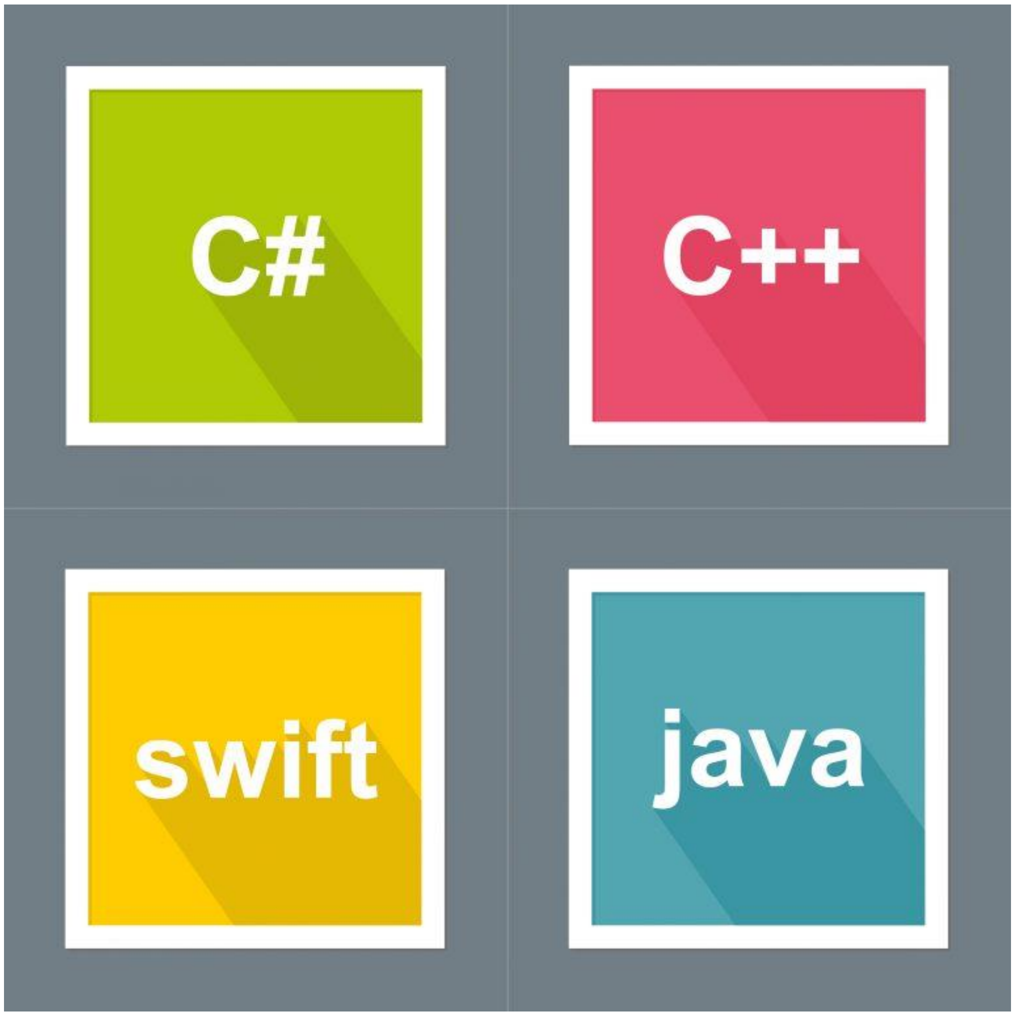
Once you have selected a Technology Stack approach you then have a further choice of code language Technologies to develop your Apps in. There are multiple mobile app code technologies that are widely used for a specific platform or for cross platform app development. The five major programming languages that are largely used for mobile app development are:

- **Swift**

If you are building something specific for Apple products only then Swift is a by far the strongest candidate. It enables you to build native apps with a full range of advanced features with minimal coding that can be easily maintained.

- **C++**

C++ is the object orientated version of the C programming language associated with Unix. It forms the base for most programming languages and possesses the power to create dynamic apps. The simple and effective compiler-based approach makes it a versatile tool that can be used for multiple platforms. Its sister language, Objective-C, is used for app development in Apple systems. Variants include C# (C Sharp).



- **Java**

This object-oriented programming language is the de-facto language of choice for Android development (official language is Kotlin). This language is easy to handle and benefits from many open source libraries that are made available for users to choose from. This should not be confused with Java Script which is as the name suggests a scripting language used for mobile web and not a programming language. Progressive Web Apps (PWA) require the use of Java Script its library's and frameworks to deliver enterprise apps.

- **HTML5**

A development of HTML 4 specifically to allow a build once, deploy many approach across multiple web platforms. This is mainly used for developing web-front-end applications for mobile devices and mobile web apps which run on multiple platforms.

- **PHP**

Considered to be a relatively easy language for a developer to learn, PHP is object-oriented and uses a three-layered model to help create dynamic mobile apps and web applications. It is useful for apps that require database integration.

7. Development Environments, Tools and Frameworks

Once you have chosen a Full Stack Environment approach and a Development Language there are a host of advanced technology tools available in which to create an App (app builders, app creators, app makers, app building platforms, app builder software, mobile development software). These developer tools are used to create Apps and can offer multi-platform capabilities for reaching larger audiences. It is critical to understand that these tools cover the whole range of those 352Bn downloaded Apps and only a few of them are suitable as an enterprise development platform and even less for carrier grade development. None uniquely provide an environment suitable for the development and maintenance of a carrier grade product, which is why at awareX we use combinations of best in class tools to develop our product.

- **Here is a (partial) list of APP development environment tools; -**
    - PhoneGap

This Adobe tool is available for multiple platforms such as Android, Windows, and iOS. It comes with a framework that wraps HTML5 apps in native containers for applications that require web applications for mobile platforms.

- Appcelerator

Allows the development of native apps for mobiles, tablets, and desktops using HTML, PHP, and JavaScript.

- RhoMobile

An open-source framework that is based on Ruby. It allows users to develop native apps for multiple platforms.

- WidgetPad

An open-source development environment for mobile apps. It provides a collaborative approach and uses web technologies such as JavaScript and HTML5. It can be used to create apps for Android, iOS, and web.

- MoSync

A multi-platform software development tool kit that is based on web programming. It supports Eclipse-based IDE for C/ C++ programming, JavaScript, PHP, Ruby, and Python.

- Flutter

An open source cross-platform SDK by Google that supports both Apple iOS and Android. Flutter uses DART as a programming language instead of JavaScript. A Flutter Framework provides reactive style views. It also offers a wide range of plugins backed by Google.

- React Native

Is a JavaScript frameworks that enable the development of native apps for iOS and Android platforms. This open-source framework offers ample support to IDEs and other mobile app development tools. It has become one of the most preferred choices for mobile app developers.

- Ionic

Widely adopted mobile app development frameworks for developments requiring the HTML5 programming language. Ionic combines HTML, CSS3, and JavaScript to build native based apps.

- Xamarin

A cross-platform app development framework for coding with C#. Using single code across iOS, Android and Windows. It attempts to provide native code across multiple platforms.

- Kalipso Studio

A mobile application generator that was originally focused Windows Mobile, Windows CE, Windows 10 and latterly Android, and iOS.

- Appery

A visual editor approach to the production of native apps (Android, iOS, Windows Phone), responsive web apps and hybrid apps. Additional functions for advanced development include push notifications and REST APIs. Support for jQuery, Angular.js, Bootstrap.

- Bizness Apps

More advanced platform compared to other app builders: visual editor, multiple integrations (including 3rd party services), m-commerce features, music and video players, GPS-based directions and notifications, shopping cart.

- Appy Pie

Focused on providing industry specific templates (dating apps, church apps, restaurant apps, SMB apps) allows the build of mobile-friendly versions of website.

- GoodBarber

An App development software used for progressive web apps (PWA) also supporting native (iOS/Android) apps via HTML5. Focus is on mid-size and smaller companies. Features include–social networking, chat, geofencing, push notifications, CMS, audio/podcast features, event listings, user profiling, sharing/comments section.

- Appsmakerstore

An HTML5 platform for Android and iOS. Industry focus on healthcare/medicine, food and restaurants, education, nightclubs, hotels, charities, government agencies.

- Mobincube

Creates Apps comprised of ads and banners, lacking user-friendliness in terms of design and sequence of actions.

- Shoutem

A React Native development environment with drag and drop user design templates.

- MobileRoadie

An enterprise focused development environment. Offices in the USA, UK and Norway. Have been used to develop Apps for Disney, TED, Universal. Along with useful design features and templates, there are functions for chats, music player, geo-targeting, digital commerce, analytics and app submission.

- AppInstitute

Enterprise environment for Progressive Web Apps, fully featured focused on the Food and Drink industry. Includes e-commerce, loyalty, booking/appointment, videos, CMS, image gallery, event scheduling, maps and geolocation, listings and HTML.

- GameSalad

As the name suggests a development environment for complex Game development in iOS and Android.

- Swiftic

Basic development environment for Apps which utilize a back-end database.

- AppSheet

Targeted at basic business Apps it works in an innovate way using only Google Docs and Google Forms to generate App content, which is then configured in a visual editor.

- Snappii

Focused on business apps for tablets, involving lots of data. (timesheet managers, construction/building log apps, travel and expense reports, inspection apps, fleet management/logistics apps, work order assignments, warehouse management) includes 200 templates, image galleries, audio/video, blog to events, catalogs, task management.:

- Yapp

Apps for iOS and Android for social events, business meetings, conventions, trade shows, good for organizing events, as it has features like one-track/multi-track scheduling, lists, invitations.

- Appsmoment

Cross-platform app development. Supports both mobile apps and games. Supports development of both PWA and native apps.

- AppMachine

Environment for turning an existing web site into a mobile app with a degree of automation.

- Verivo

Verivo AppStudio, purchased by Appery (see above), solid development environment. It has code-free visual studio, incorporating JavaScript and HTML5 to build any kind of apps. Automation for user authentication, data synchronization, integrations and deployment.

- Nevercode

A platform for advanced developers working with numerous projects and the need to optimize development cycles. Automates app configuration and setup, deployment, testing, code analysis, publishing to multiple app stores. Supports iOS, Android, Cordova, Ionic and React Native projects. Development tools are Python, Angular, Celery, Bootstrap, Android SDK and iOS SDK.

- AppYourself

Environment to build native and pure HTML5 apps. Originally intended to enable reverse engineering from an app to create a responsive website. (This feature never actually worked unfortunately).

8. The AwareX Technology Stack

You will have understood that the main considerations are native as opposed to one-time developments which run on any platform, native delivers the best experience but mobile web is lower cost to develop and maintain. Add to this the new consideration of PWA which are an enhancement of mobile web apps which try and deliver a native like experience. You can choose to use a development environment and you have a choice of code language which needs to align to the type of Apps you are building.

- **Why did AwareX select the Technology we use?**

At AwareX we use a wide range of Technologies. It is not a question of picking just one and forcing it into a use case, rather we use the right technology for the right digital engagement system. This leads us to prefer native apps for smartphones but reactive mobile web for our web portal systems which also run on tablets and smartphones. Our Bots use further appropriate technologies and they all access customer data through an enterprise microservice integration platform built with its own compatible and related tool set.

- **JavaScript vs. Java/Swift/Objective-C**

When building a long term stable and capable product you need to consider the skillset of the development team, it's clear that JavaScript is the primary technology used for mobile web apps, rather than programming languages such as Java, Swift, and Objective-C. Most JavaScript programmers are not expert Java and Swift/Objective-C programmers, and most Java and Swift/Objective-C programmers are not expert JavaScript programmers. The differences in skillsets is fairly wide and needs to be considered.

Progressive web apps of any substance will need to integrate JavaScript libraries and frameworks to provide a complete offering close to what a native app can offer. Progressive web app technologies alone do not integrate native SDK's directly, and would require the use of JavaScript, proprietary scripting languages (React Native), or HTML/CSS (Cordova) to access native API's and services.

- **Browser Considerations**

Since browsers can vary across devices and their operating system versions, they can behave differently with respect to caching web page assets, caching HTTPS responses, accessing persistent stores, etc.… There are various mechanisms and fall backs to accommodate some of these limitations, but they do exist and need to be taken into consideration when retrieving, caching, and persistent data. Native apps would perform consistently without these types of inconsistencies.

JavaScript itself varies across browsers (iOS devices vs. Android devices and iOS versions/device vs. Android OS/device), whereas native apps tend to work properly on all their supported devices, without the need to fork code to handle browser inconsistencies.

- **Native Capabilities**

Some native capabilities of an operating system can be utilized via frameworks such as React Native and Cordova, but not all.

Due to the open source nature of these frameworks, products are at the mercy of contributions, with respect to missing/limited capabilities, defect resolutions, updates for new OS versions, and inclusion of new capabilities introduced in new OS versions.

Of the native capabilities supported via React Native or Cordoba, it’s fair to say that not every available capability can be utilized consistently across devices/OSs.

There will always be a host of native capabilities that will not be able to be utilized by a Progressive Web App, whether those capabilities are provided by the native SDK itself or a third-party library (e.g. Analytics, Crashlytics, etc…). That being said, the benefits of a universal app need to be measured against the limitation in offering.

- **Progressive Web Applications**

Progressive Web Apps (PWA) are intended to give a native like experience across multiple platforms with one code set. They are Responsive Web Applications and should be able to adapt to different screen sizes and orientations. Traditionally, native applications have been much more engaging than web applications. Having an icon on the home screen makes it easy to get into the app and push notifications can help alert the user to important information that requires their attention. Progressive Web Applications build on the base of Web Applications to try and give a similar user experience to native Apps.

The accepted consensus is that PWA are relevant for development of internal enterprise applications where optimum usability can be compromised for the cost of developing, testing, and maintaining applications for several platforms. It is worth noting that all Android devices are PWA compatible, however Apple has only implemented support from Safari 11.1 and iOS 11.3.

- **AwareX choice and use of Technology.**

After the discussion of the wide range of potential Technologies listed above this section lists the actual technology that awareX uses to develop our Digital Customer Engagement System.

We believe in a customer first approach, that means we must deliver the best possible end user experience; hence we deliver Native Apps for Smartphones. We also deliver response apps for Web, since the customer experience requires the ability to use the web apps on any device.

- **Apple iOS**

AwareX uses Apple specific tools to ensure the best possible native app experience for iOS users.

Development: iOS SDK

Languages: Swift, Objective-C

- **Google Android**

AwareX uses Android community tools to ensure the best possible native app experience for android users.

Development: Android SDK (but we do not use React Native)

Languages: Java

- **Microservices Integration Platform**

As described in this document AwareX is a full stack product company, the AwareX Microservices integration platform uses the following:

Development: Node Java Script, REST JSON, for the connection between the platform and the client devices, facilitating enhanced architectural flexibility.

Languages: Java

- **Content Management System CMS**

WordPress

- **Monitoring Platform**

Elastic Search, Search and analytics engine

Kibana, Operational data visualization

Logstach, Server-side data processing pipeline for simultaneous multi source data ingestion

- **Bot Clients**

AwareX's extended digital omni clients which leverage the Microservices integration platform and include:

Facebook Bot, SMS Bot, Google Bot, In App Assistant for Android

Development: Spring Framework, Dialog flow for NLP (Natural Language Processing)

Language: Java enterprise application

- **Web**

AwareX omni client system includes a full Web portal which also works on Laptops, Tablets and Smartphones. (A responsive mobile web system).

Development: Node Java Script, Java Script Framework, React and Redux, Java Script, ES6, CSS Framework, Bootstrap

Language: HTML5, CSS3(SASS)

- **Build Creation:**

Webpack

- **Build Deployment:**

Jenkins (enables deployment to a production environment including rapid changes in the integration platform).

- **Cloud Environment:**

AWS allows us to deploy secure virtual servers quickly and if needed, scale the systems quickly within minutes.

Certificate Manager: AWS Certificate Manager lets you easily provision, manage, and deploy Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with AWS services.

CloudFormation: AWS CloudFormation lets you create and update a collection of related AWS resources in a predictable fashion.

CloudSearch: AWS CloudSearch is a fully managed search service for websites and apps.

DynamoDB: Amazon DynamoDB is a scalable NoSQL data store that manages distributed replicas of your data for high availability.

EC2: Amazon Elastic Compute Cloud (EC2) provides resizable compute capacity in the cloud.

EC2 Container Service: Amazon ECS allows you to easily run and manage Docker containers across a cluster of Amazon EC2 instances.

Elastic Beanstalk: AWS Elastic Beanstalk is an application container for deploying and managing applications.

ElastiCache: Amazon ElastiCache improves application performance by allowing you to retrieve information from an in-memory caching system.

Elastic File System: Amazon Elastic File System (Amazon EFS) is a file storage service for Amazon Elastic Compute Cloud (Amazon EC2) instances.

Glacier: Amazon Glacier is a low-cost storage service that provides secure and durable storage for data archiving and backup.

IAM: AWS Identity and Access Management (IAM) lets you securely control access to AWS services and resources.

Inspector: Amazon Inspector enables you to analyze the behavior of the applications you run in AWS and helps you to identify potential security issues.

Lambda: AWS Lambda is a compute service that runs your code in response to events and automatically manages the compute resources for you.

Route 53: Amazon Route 53 is a scalable and highly available Domain Name System (DNS) and Domain Name Registration service.

Storage Gateway: AWS Storage Gateway securely integrates on-premises IT environments with cloud storage for backup and disaster recovery.

SNS – Alerting Service

S3: Amazon Simple Storage Service (S3) can be used to store and retrieve any amount of data.