Learn How to Build a Video Conference App (and not die trying)!

feel free to call us (+1) 434 205 3731 team@webrtc.ventures

Hector Zelaya

February 1, 2018 (https://webrtc.ventures/2 how-to-build-a-video-conference-app-and-not-die-trying/)Technical, Thoughts, chat conference call, free video conferencing, realtime communication, video call, video chat, video conference app, video conferencing, webrtc

0

Real-time Communication is quickly becoming a "must-have" feature in many types of applications. From customer service to telehealth, video conferencing is rapidly integrating into different industries' workflows. If you are reading this, chances are that this phenomenon caught your attention and you are now asking yourself: How do I build a video conference app?

However, building such an application can be a complex task without the proper guidance. You may end up with an app that "kills" your business as it cost twice its value or you may have an app that barely satisfy your requirements because it doesn't have the features you need.

In this article, we will provide you some guidelines that will aid you in building a successful video conference application with the help of our core building block: WebRTC.

So you have your million-dollar-idea for an app and you're ready to begin development?

Ok, hold your horses a minute and let's discuss some of the things you might want to look at before getting your hands dirty.

Step 1: Select a Platform (Or not)

The first thing to consider when building a video conference application is the target platform, as this will determine both the tools needed to build the app and the budget you'll need.

So before starting to build your app, ask yourself: Where do you want your application to run? Smartphones? Tablets? Laptops or desktop computers? ALL OF THE ABOVE??? Yes, this is also possible.

(https://webrtc.ventures/wp-content/uploads/2018/02/blogimge2.png)

Using WebRTC you don't need to select a specific platform as you can offer to support all of them.However, you should select the approach that best matches both your requirements and the resources you have at hand.

Available options are: web-based and native.

Initially, you may want to develop a **web-based application**. WebRTC APIs are already included in all major browsers so any device that has one of them installed, can be used to access your app.

Another benefit of web-based applications is that you don't need any third-party tool to build it, you can use plain ol' HTML, CSS and Javascript without problems. Though, using something like ReactJS, Angular or Vue might give you more power to build your app.

Many popular video conference applications offers its web-based application as the default type of application for the desktop platform, that is, laptops and desktop computers in general.

Although this generally works for this platform and SHOULD work well on the mobile spectrum, sometimes you may want to use something more optimized for that specific platform.

Hence, give your app more power.

In this case, a native may make more sense. Also if you want to support legacy devices which don't support WebRTC APIs, native is the road to take.

For **native applications**, you should take into account which mobile platform you wish to support.

According to WebRTC.org, Android and iOS are officially supported through their respective programming languages: Java and Swift/Objective-C. However, that means that in order to support both operating systems, you may need to build a different application for each. Or you can use frameworks like React Native which allows to build native apps that runs on both Android and iOS devices.

A technology that is recently gaining popularity is the concept of Progressive Web App (PWA), as it combines the good things of both worlds: web and native. This is something that you might want to check out when considering a strategy to build your application.

Many popular video conferencing applications like Hangouts or appear.in offers a web based application for desktop and a native application for mobile devices. You should consider what suits best to your needs and adopt a strategy that allows you progressively support your desired target platform(s).

Step 2: Define Your Features.

After you have selected the platform for your application, you need to define the features your app will offer. Depending on your business requirements you might want to prioritize and focus your efforts and budget toward these.

For example, being able to use filters or funny icons during a call is a nice touch, but it might not be suitable for a technical support app or a customer service one, instead the capability for recording the calls can be an useful option, especially when you want to implement quality analysis or be compliant with some regulation.

Below is a short list of the most common realtime features offered by some popular video conference apps out there.

Pre-call Video preview

Popular apps like Hangouts allows the user to check the camera before joining a call and then provides option to disable it if the users wish to. I personally found this really useful, especially when joining business calls at 5AM and didn’t want to scare anyone with my ‘just-woke-up’ face.

Chat

Written communication is a “must” in most video conference applications as it provides an additional channel for the users to communicate during a call.

File Sharing

Allowing call participants to exchange files is a convenient feature. For example, in a telehealth app, it will allow a patient to send test results or any previous medical record to the physician. Just note that you must take in count the measures needed to securely

For our 56th episode of WebRTC Live, Arin Sime was joined by Wonder Co-Founder Leonard Witteler. Leo...

(https://webrtc.ventures/2021/06/webrtclive-magic-webrtc-wonder/)

Types of Software Tests
June 18, 2021

It may not sound comforting to know that there are people who spend 100% of their time finding mista...

(https://webrtc.ventures/2021/06/types-of-software-tests/)

Background Removal with the Vonage Video API
June 10, 2021

Taking WebRTC Video calling to the next step involves providing more functionality and custom behavi...

(https://webrtc.ventures/2021/06/background-removal-with-the-vonage-video-api/)

(/contact/)

store and transport these files.

Multi-party Conference

Allowing more than two users to join a call is one of the most complex tasks to accomplish, as it has to do not only with the application, but with the underlying infrastructure that hosts it. Having a clear idea of the number of users you want your application to be able to support in a call is the key to choose an appropriate strategy later.

Desktop Sharing

The ability to share the content of the screen during a call is helpful in areas like remote technical support, where a specialist can guide users to achieve an specific task on which they're having trouble, by giving instructions based on what the user sees.

Recording

Many apps allow recording calls for several different purposes. If you want your application to be able to make calls, among the things you should consider are the type of storage, the format of the recordings and the security measures to prevent unauthorized users to access them.

Filters/Icons

A lot of popular social media applications are allowing users to add funny filters or icons to its media streams during a call. This is something that can be fun and can serve different purposes but it will depends on your own business requirements to add this capabilities.

Whiteboard

This is a valuable feature for educational applications, in which a user needs to teach something to other users. It gives the "teacher" user a tool to express an idea graphically to his "student" users.

(/contact/)

Live Streaming

Live streaming is another feature that has been popularized thanks to social media. It allows a user to stream video and audio to other users in real-time. Outside of the social media context it can be really helpful in Disaster Control Applications, where a rescue worker with access to disaster zones can give feedback about a situation to government or rescue organizations in real-time, so that they can perform the appropriate necessaryactions.

Step 3: Know the Stack.

Now that you have a well defined idea of the platform where your app is going to run and the features that it will offer, it's time to know how you are actually going to build it.

From the technical point of view, WebRTC is nothing more than a group of standards and features comprised in APIs that can be used to gain access to media devices and establish a peer-to-peer connection with other clients. These APIs used in conjunction with a signaling process and a bunch of other elements, are used to initiate a video/audio call between two or more users.

The signaling process is not defined as part of the technology and the developer is free to use any of the well known signaling protocols such as SIP and XMPP, or implement its own solution using a full-duplex communication technology like Websockets.

There are two ways to develop and run a video conference application using WebRTC: On-premise and using a CPaaS (Communication-Platform-as-a-Service) provider.

On-premise means that you are responsible for both developing the application and managing the required server infrastructure.

On the other hand, using a CPaaS means that you only take care of developing the app while using the infrastructure of a provider, usually paying a monthly fee for it.

This leaves us with three strategies to develop a video conference application:

- On-premise – Peer-to-peer approach
- On-premise – Media Server approach
- Using a Communication-Platform-as-A-Service (CPaaS) Provider

Let's briefly discuss these options below.

On-Premise – Peer to Peer Approach.

WebRTC is peer-to-peer by nature. This means that most of the time, there will be no intermediaries in a WebRTC call. The communication will be direct, browser to browser or device to device. This added to the fact that it encrypts the media transport by default, makes it a secure solution for real time communication.

(/contact/)

However, client devices typically reside behind NAT configurations and/or firewalls restrictions, which makes difficult to establish a direct connection between them, and sometimes they completely prevent it. In order to surpass this issue, servers known as STUN/TURN are used to help with the establishment of a peer-to-peer connection or to rely media to the other user in case such connection is not possible.

When building a web conference application on-premise using this approach you're on charge of building the actual application and also setting up both the signaling layer, whether developing an in-house solution or using something like SIP or XMPP, and the STUN/TURN servers.

The main advantage of this approach is that you own full control over the performance of your application. The downside is that you need to provide and maintain your infrastructure.

One important thing to consider is that due to the peer-to-peer nature, some features, such as enabling recording a call, manipulating the streams or adding multi-party capabilities, may not be implemented that easy, at least not without having to add additional burden to the application. This can lead to the call failing under certain circumstances. In these cases, you may want to add a media server to do the hard work.

On Premise – Media Server Approach.

A media server sits in the middle of the call participants and send and receive streams to and from them. This approach provides you a central point for manipulating media streams, allowing you to add some advanced features, such as recording, simulcast and multi-party calls.

When building an on-premise web conference application using the media server approach, besides building the application and adding the signaling layer and STUN/TURN servers, you need to add the actual media server and configure it accordingly.

Some popular open-source options for media servers are:

- Kurento – http://www.kurento.org/ (http://www.kurento.org/)
- Jitsi – https://jitsi.org/jitsi-videobridge/ (https://jitsi.org/jitsi-videobridge/)
- Janus – https://janus.conf.meetecho.com/ (https://janus.conf.meetecho.com/)

Using a Communication-Platform-as-A-Service (CPaaS) Provider.

This is the simplest way to build a video conference app, as it frees you from having to provision and maintaining your own infrastructure and instead allows you to focus on writing your application.

(/contact/)

Note that when you use a CPaaS you have little to no control on the infrastructure and you should take into count the monthly fee that you will be charged for its use.

Some popular CPaaS providers are:

- Tokbox – https://tokbox.com/ (https://tokbox.com/)
- Vidyo – https://www.vidyo.com/ (https://www.vidyo.com/)
- Twilio – https://www.twilio.com/ (https://www.twilio.com/)

If you want to know more about it you can checkout our comparison of CPaaS providers post. (https://webrtc.ventures/2017/09/which-cpaas-provider-alternative-is-the-best-choice-for-your-application-tokbox-vidyo-temasys-twilio-more/)

Conclusion

Developing a video conference application efficiently is possible and hopefully this article will provide you the insights you need to accomplish it. Focusing on the platform and features you really need and adopting the right strategy according to your needs will give you an application that surely will increase your business value without dying while trying.

Bonus: Learn to Build with Us.

As bonus content, we offer a complete course of video conference application development (https://webrtc.ventures/webrtc-training-tutorials-ecourse/). In the course we go through the definition of each one of the required components of a WebRTC call and provide working examples of the three strategies discussed above. Click on the above link to know more.

Should you want to leave the building to us, we'd be happy to help.

If you are ready to chat about how you can incorporate a video conference app into your business? We have an experienced team ready & happy to help you out. Contact us today (https://webrtc.ventures/contact/).

SHARE THIS

25 likes

(http://www.facebook.com/sharer.php?u=https://webrtc.ventures/2018/02/learn-how-to-build-a-video-conference-app-and-not-die-trying/)

(/contact/)

(https://twitter.com/share?url=https://webrtc.ventures/2018/02/learn-how-to-build-a-video-conference-app-and-not-die-trying/)
(https://plus.google.com/share?url=https://webrtc.ventures/2018/02/learn-how-to-build-a-video-conference-app-and-not-die-trying/)
(http://pinterest.com/pin/create/button/?url=https://webrtc.ventures/2018/02/learn-how-to-build-a-video-conference-app-and-not-die-trying/&media=https://webrtc.ventures/wp-content/uploads/2018/02/blogimage1.png&description=Learn How to Build a Video Conference App (and not die trying)!)
(mailto:?subject=Learn How to Build a Video Conference App (and not die trying)!&body=https://webrtc.ventures/2018/02/learn-how-to-build-a-video-conference-app-and-not-die-trying/)

RELATED ARTICLES

(https://webrtc.ventures/2019/08/connecting-the-phone-with-a-webrtc-call/)

Connecting the Phone with a WebRTC Call (https://webrtc.ventures/2019/08/connecting-the-phone-with-a-webrtc-call/)

WebRTC has growing applications by the day, but some may argue that it's limited by its

(https://webrtc.ventures/2018/11/nexmo-tokbox-and-changes-in-the-cpaas-what-does-it-all-mean-for-you/)

Nexmo, Tokbox, and Changes in the CPaaS – What Does it All Mean for You? (https://webrtc.ventures/2018/11/nexmo-tokbox-and-changes-in-the-cpaas-what-does-it-all-mean-for-you/)

Developing a Secure Messaging App for Telehealth with React Using Virgil Security, Higher-Order Components and React Context API
React
VIRGIL SECURITY
webRTC.ventures

(https://webrtc.ventures/2019/06/developing-a-secure-messaging-app-for-telehealth-with-react-using-virgil-security-higher-order-components-and-react-context-api/)

Developing a Secure Messaging App for Telehealth with React Using Virgil Security, Higher-Order Components and React Context API (https://webrtc.ventures/2019/06/develo

(https://webrtc.ventures/2018/01/webrtc-on-android-tutorial-how-to-build-a-chat-roulette-clone-using-kotlin-and-typescript/)

WebRTC Ventures

Home (https://webrtc.ventures/)

WebRTC Services

Development Services (https://webrtc.ventures/webrtc-development-services/)

Recent Blog Posts

Four Ways to Fix Your WebRTC Application
July 9, 2021

(/contact/)

Our Team (https://webrtc.ventures/our-team/)

WebRTC Resources (https://webrtc.ventures/webrtc-resources/)

Blog (https://webrtc.ventures/blog/)

Telehealth Solutions (https://webrtc.ventures/telehealth-solutions-by-webrtc-ventures/)

Testing Services (https://webrtc.ventures/webrtc-testing-services/)

(https://webrtc.ventures/2021/07/four-ways-to-fix-your-webrtc-application/)

7 Principles of Software Testing
July 2, 2021
(https://webrtc.ventures/2021/07/7-principles-of-software-testing/)

Contact WebRTC.Ventures Today

Real-time technology is the new face of customer communication. Let webRTC.ventures deliver easy-to-use, encrypted, high-quality apps for your business that feature voice, video, and data customization.

Let's build your app! (/contact/)

(https://twitter.com/WebRTCVentures)
(https://www.facebook.com/pages/WebRTC-Ventures/869158246494506?fref=ts)
(https://www.linkedin.com/company/webrtc-ventures/)
(https://www.youtube.com/channel/UC7ty8nH5cBP0mqiZFl9bb9g)
privacy policy (/privacy-policy/)

(/contact/)