## Hello 👋🟫

Welcome to the Freshworks developer community, (If you haven't already signed up to join our community, now is a great time to do that!)

We are super excited to have you aboard and look forward to working with you on an equally exciting journey on the Freshworks Developer Platform!

This document is designed to help you along on your first few milestones of this journey. We will start with what Freshworks offers to developers like yourself on its modern, SaaS platform and how organisations use the apps built on this platform. We will also explore app development on the platform from a bird's eye view and point you to useful resources to build your first couple of apps. We will then leave you with the foundational resources that help you become a pro  Freshworks App developer with strong fundamentals of our platform. Fasten your seatbelts!

**Pre-requisites**

Before we dive in, let's take a minute to ensure we are on the same page with respect to what this document assumes regarding you, our new developer, and the experience you already possess.

- You have a broad idea of how teams use CRM, ITSM, HRMS, and Customer Support software or, even better, have used one yourself. Examples of such software include JIRA, Salesforce, Intercom, Service Now, Zendesk, or our very own Freshdesk, Freshservice, and Freshworks CRM.
- You have a working knowledge of web development concepts and apply them using HTML, CSS, and Javascript regularly while consuming SDKs and APIs.
- You have built a full stack web application and therefore are aware of the components involved.
- You have been an active participant in software development lifecycles and are aware of what it takes to build an application locally and make it live in production.

---

## A bird’s eye view

Business software is a serious world. In today’s automated and heavily integrated business landscape, software and how it all comes together to run an efficient shop can often make or break a business. Every 'integration' a business builds and leverages comes with a cost and every customisation done to a piece of software the business acquires requires effort. As an experienced developer in this arena, you probably know this all too well having chased and delivered to deadlines around complex software requirements.

At Freshworks, we have set out to build a platform for developers of business software with the objective of not only helping them solve problems for businesses using software, but also delighting in the experience of leveraging the platform to get their job done.

Let’s begin with a quick survey of the concepts you will want to familiarize with in order to succeed on the platform.

Let’s do this by imagining you want to build a web application to manage a TODO list. Although there are many ways to go about building this, a possible simple approach would include,

1. Use HTML and CSS to construct a form that takes in a title & description of the task.
2. Stand up a backend service exposing APIs to manipulate a *Task* resource.
3. Create a browser-based client to use these APIs and manipulate *Tasks* based on user interactions and inputs.
4. Choose a data persistence service to store/retrieve the *Task* records and connect this with your backend service.
5. Setup a virtual machine to host your backend service and securely expose the APIs to the client.
6. If someday your web app becomes popular (and it will!), add more servers, and make the backend highly available and reliable.

As we can see here, the task (mind the pun!) of creating a task manager involves more than just writing the code to handle the business logic.

Specifically, this also requires,

1. A development environment to write, organise, test, and build code.
2. A cloud infrastructure provider to host and scale your services.
3. A provider to host, manage, and scale your database.
4. If you plan on updating your application often, you also need an automated way to build and deploy your updated source code to your cloud instances and get them into the hands of your users.

When you work with the  Freshworks Developer Platform however, our endeavor is to help you build this very same web application without having to learn how to host, manage or scale your own servers, databases, CDNs, and so on. As a developer, we really just want you to do two things:

1. Understand the business requirements from your customers and translate them to business logic encoded in source code.
2. Test your code using our development toolkit and upload a ZIP file carrying your Freshworks app package.

Freshworks will take care of the rest!

Now that you have seen what this experience might look like, let's try and understand what encompasses the Freshworks Developer Platform. Yes, as you might imagine, this includes the tools, the services, APIs, and SDKs that we offer.

---

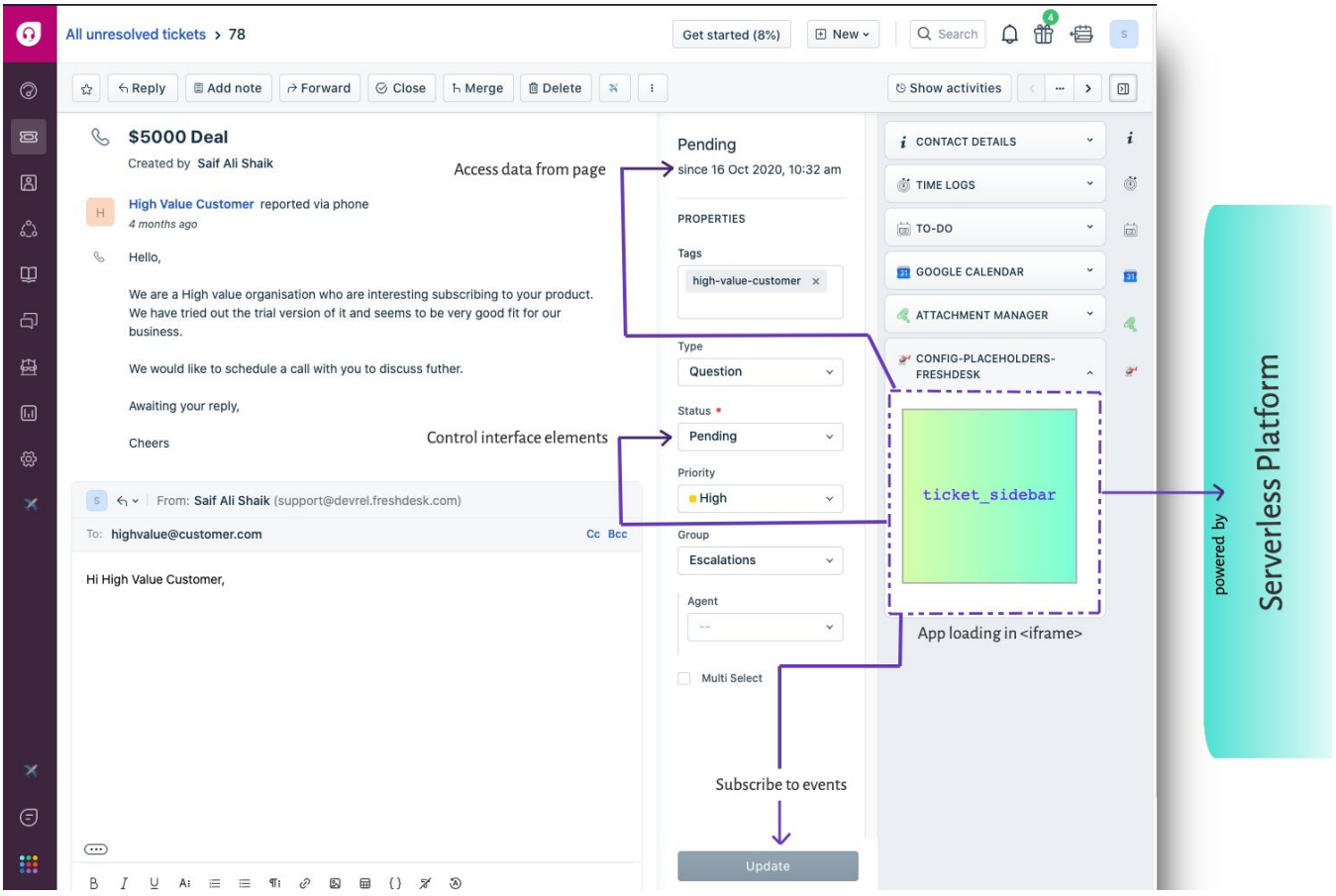
## A sneak peek

Let us begin by trying to picture what happens after an app (let's say, your Task Manager app) that you build is deployed to our platform.

Once the app is up and being served, a user with the right privileges will be able to discover this app and complete the installation process through the Freshworks App Gallery. Typically, the app is installed within the context of a Freshworks product *account*. An *account* is always associated with a unique domain URL (For example, our support engineers use *support.freshdesk.com* to provide support) and a Freshworks product (For example, Freshdesk, Freshservice, Freshworks CRM, and so on).

Tip: You can get your own account right away, by signing up for a free trial.

Next, a logged-in user (For example,a support agent / sales person having access to the *account*) will be able to access the app within the Freshworks product user interface that they use to get their work done. A front-end app typically loads within a secure inline Frame as shown in the following image and has access to the browser runtime like any other web application. However, as it is sandboxed within an iFrame, it has limited access to the UI elements and interactions of the Freshworks product UI it loads within. We will see later how your app can still access those securely - hold tight for now! (Sneak peek - it involves the use of Data Methods, Interface Method,s and Event Methods). In addition, the app also has access to a powerful back-end platform that Freshworks offers to each app runtime - this is primarily a NodeJS serverless runtime, which includes data persistence, among other capabilities.




*Sample App placeholders for Freshdesk*

**Time to get hands-on**

Let's get a first taste of the app development experience on the Freshworks platform.

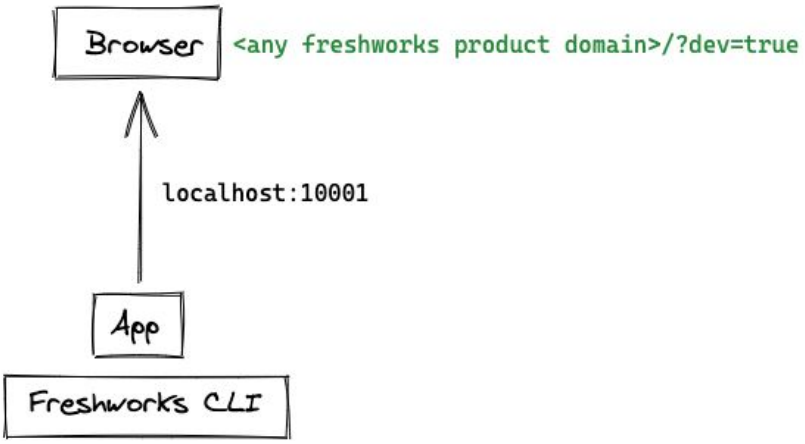
The Freshworks platform offers a CLI (lovingly named *fdk*) built specifically for developers like you to write, organise, and build Freshworks apps. It runs on most popular operating systems including MacOS, Windows, and Linux. Go ahead and install it. We will wait!

With your FDK setup, we can now start playing around with it to build your first app. For this exercise let's create a Freshdesk account and run through this tutorial.

**Congratulations! You have now built your first Freshworks app and even tested it.**

A quick segue into some interesting and relevant theory about the FDK you just used to build your first app. When you run the FDK using the *fdk run* command, it creates a local server and serves your app over it on port `:10001`. Each Freshworks product is capable

of rendering this app being served on your localhost on any page the app is configured to be displayed by just appending `?dev=true` to the URL.




While the first app you built did not use them, it is worth knowing that apps can also consume REST APIs. Most integrations you build will have a need for this. For example, you can programmatically create contacts on Freshdesk or view an appointment on CRM. Each product has its own set of REST APIs that any app can consume as a client.

For now, we will leave you with that tidbit and come back to this a little later!

---

**Common problems solved by apps**

To get a better perspective of what kind of problems apps solve for businesses that face them everyday, let's look at a few enlightening use-cases. After you've seen a couple of them, we will pick a Freshdesk account and build an app for ourselves following a tutorial.

## Customer Experience

1. Freshdesk displays tickets in a list. Some teams, especially team leaders, prefer to work with Kanban style. Explore and check out how the Kanban Board app can help with this.
2. Customer facing teams often need to converse with customers in their native tongue. Yes, there is an app that helps agents translate text within the product before sending it the customer's way.
3. Most customers use some tool outside of the Freshworks family - sometimes this is an in-house tool - and wish that the information in those tools is in sync with the Freshworks product(s) they use. You can check out our most popular integration here, or if you have a favorite tool to integrate to, you can build one!

## IT Service Management

1. IT teams manage assets. While at it, asset managers have to update all the information into Freshservice upon receiving a new delivery of assets. An app can eliminate time-consuming tasks and reduce human errors.
2. At times, emails are exchanged on a ticket for months and the conversation is simply just too long. Finding relevant attachments in this long thread could be a nightmare, except when there is an app to show you all attachments at one place!
3. Maybe the IT service team has decided to ship a couple of items to employees who work from home. We could help the IT team with an app that allows picking from an approved list of critical products and map them to the employees eligible for a WFH kit.

## Customer Relationship Engagement

1. When replying to Leads within CRM, you might want to first check if this email is already captured in the Ticketing tool that the support team uses and pull additional details from there.
2. Because your marketing teams use a number of other tools such as Mailchimp, Google Analytics, and your account teams use QuickBooks, while your Sales team tracks their day-to-day work in Trello and Asana, your CRM might want to get integrated with all of them!

**Dive In**

Now that we have discussed use-cases and a few problem statements, why not get into one and try to solve it? Follow this tutorial and build your first functional Freshworks App that can create tickets! [Bonus: You will learn to make your first Freshdesk REST API call]

Wasn't that straightforward? If you are still up for more, let's try another hands on experience that will get you another badge - your first Freshworks serverless app!Before we let you explore more of our tutorials, let's take a few minutes to introduce you to somerelevant concepts.

There are two types of apps you are likely to build on our platform.

- Apps that are intended to be published on our public marketplace, referred to as Marketplace Apps.
- Apps that are built to be used specifically for an individual customer/business/team depending on their unique requirements and are therefore only available to one product *account*, referred to as **Custom Apps.**

If you have stuck along this far and built your first three apps, give yourself a pat on the back. There is much more you can accomplish with the platform as you learn and explore the features we offer you. As always, the platform requires you to simply focus on the use case and the code that you write, and leave the rest to us.

What more is there to learn, you ask? Well, there are so many other things an app can do (and we have tutorials for most of them here) by leveraging the various capabilities offered by our platform.

For example, you can,

1. Listen to an event inside a Freshworks' product UI and execute your app logic.
2. Run cron jobs to trigger your app.
3. Have a webhook generated and registered for your app to listen to.
4. Render UI with Crayons.
5. Securely manage sensitive information required by your app at runtime.
6. Have access to a complete serverless node environment to run your back end.
7. Persist key value pairs as stateful data for your app.
8. Perform OAuth 2.0 handshakes and access resources securely.

Wow! That might look like a lot to digest, but think about all the things you can build if you became comfortable with it all.

Before we leave you to yourself and the platform, we would love to share a few handy resources that will help you keep going and growing on the platform.

- **Documentation** - Each Freshworks product attracts its own set of problems to be solved and APIs and methods available to developers might vary across products. Learn more by diving into any of our product documentation.
- **Sample apps** - For every feature and product, we try to provide a simple application that could help picture a solution. Start exploring Sample Apps today.
- **Tutorials -** With the basic concepts of the platform in your backpocket, the next obvious step would be to gain more hands-on experience building apps. With this exact intent, we have prepared some step by step guided tutorials for developers. Explore the tutorials now.
- **Forum -** Ever ran into a problem and wondered if someone else has already faced it? This is one reason we have built a community around the Freshworks platform! You can also connect with other fellow developers and learn from each other in our community. Explore the forum for Freshworks Developers**.**
- **Office hours -** If you did not find a solution from the community and have broken your head enough, you probably need some 1-1 help with the use case you are solving., This is precisely why our Developer Relations team offers an Office Hours program where you can book a 30-min session with a clear agenda on what you wish to discuss with us!
- **Crayons -** If you wondered how you can build an app UI that meets the Freshworks Style Guide and looks and feels exactly like our products, Crayons is the web component library for you! Check it out. It's open source and we welcome issues and contributions.

- **Blogs -** We frequently share updates and make announcements that ease the app development process and equip your team with newer opportunities. Make sure you keep an eye out on the Developer Platform Blog.

- **FAQs** - We have tried to actively maintain a set of common problems our community has run into. Don't forget to check out Frequently Asked Questions while you swing by our forum.

- **Newsletter -** Every month we learn and release updates for developers like you. Consider subscribing to the newsletter and be first to grab a seat at developer events, join beta programs and lot more!

You have built your first few apps and are armed with a quiver full of resources to continue building. You have a good, basic understanding of what Freshworks has to offer and how this might be different from anything else you have been building on. **Congratulations!** You are now well on your way to becoming a successful app developer on the Freshworks Developer Platform. We will certainly run into you more often henceforth, either on one of our exciting developer events, or our developer community, or as you are taking up one of our certification programs!

From the Developer Relations team here at Freshworks, we look forward to meeting you soon!