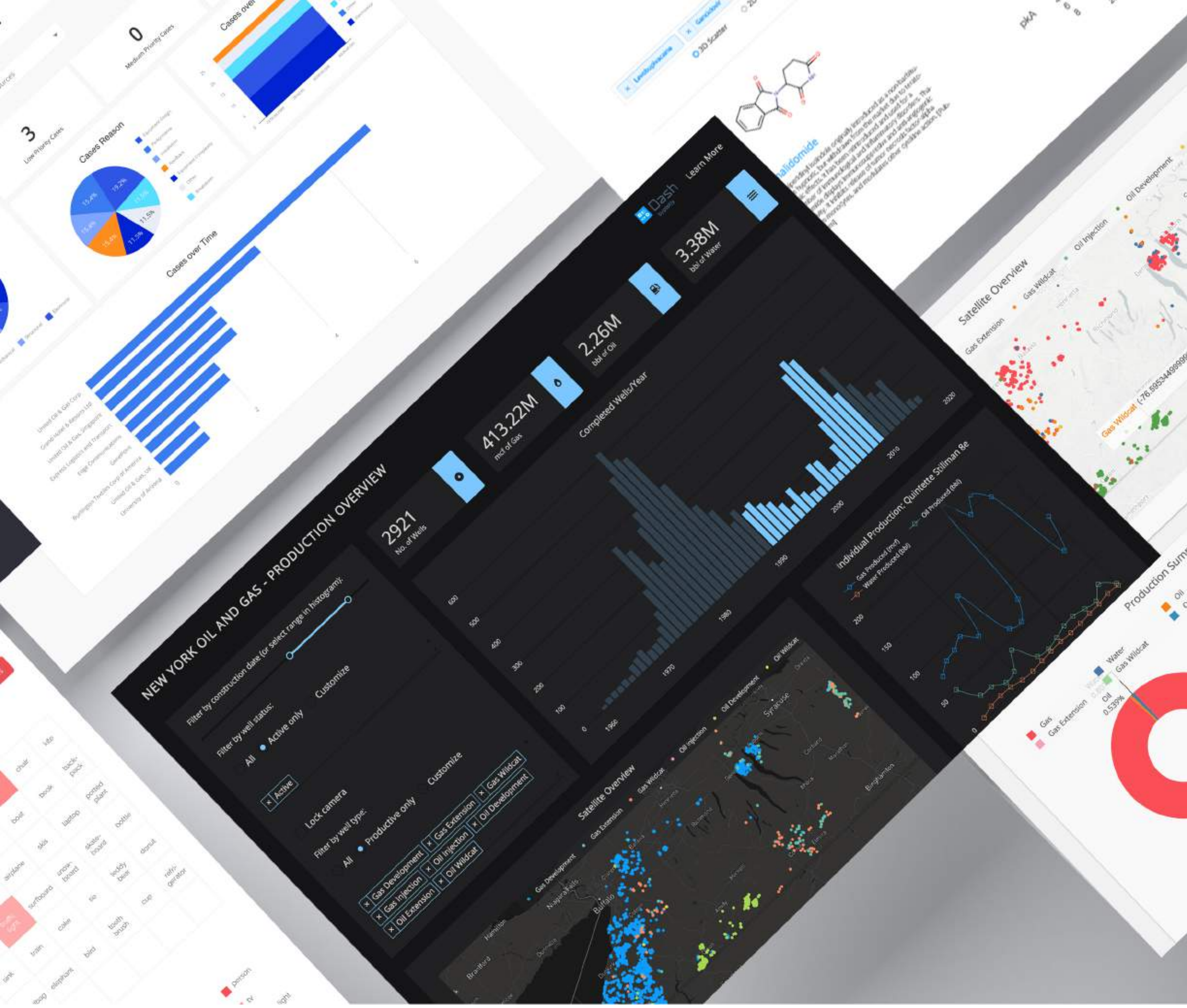


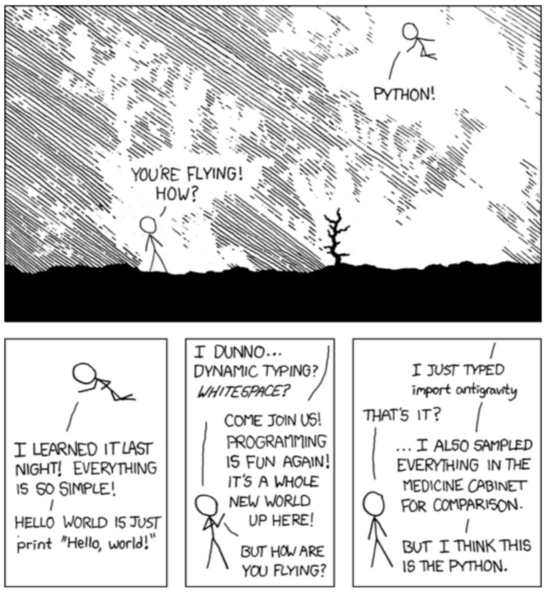
plotly | Dash Enterprise

Dash Enterprise's Incredible 21x Cost Savings

Dash delivers the power & magic of Python analytics while saving 95% of the cost of building full-stack analytics software in-house.

Summary

Part I: Python as the New World Order



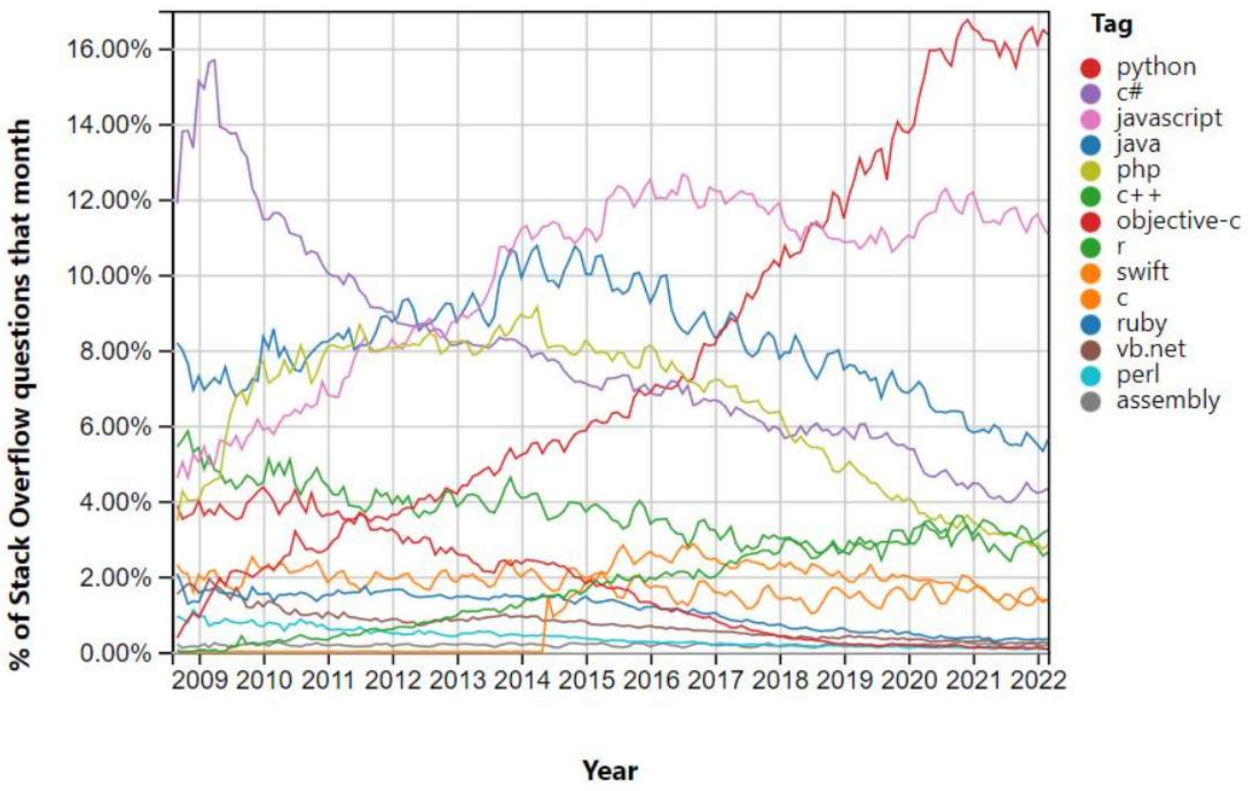
PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

It is now common for forward-thinking companies to build their own analytic stacks for problems that do not yet have packaged, off-the-shelf analytics software. For example, Tesla built their own Python analytics software for monitoring their supercharger network. Microsoft Quantum Research built their own Python analytics software for their quantum computers. Amgen and Moderna Therapeutics built their own Python analytics stack for drug development and clinical trial analysis. (All of these companies or initiatives use Plotly's Python-integrated software.)

These companies build their own analytics stacks in-house because their technologies have outpaced the traditional analytics industry. There is no off-the-shelf analytics software that you can buy to advance these technologies because they are at the bleeding edge. In recent years, the key enabler of this in-house advanced analytics trend is the Python programming language. Python has unseated its commercial rivals to become the lingua franca of scientific computing. If MATLAB ruled the pre-aughts engineering world of signals processing, image analysis, wireless communications, and numerical differential equations, Python is light years ahead in Machine Learning, "Big Data" crunching, online code collaboration (e.g., Github and Stack Overflow), quantum computing, climate science, options for web servers, and bioinformatics.

The Python trend is exponential. Because Python is open-source, researchers and engineers share their recipes and tools freely online, universities have switched to Python in their engineering curriculums, and the big clouds and VCs have invested billions in machine learning stacks with Python code bases at their core.

Given this super-trend, it's unsurprising that virtually all F500 companies already invest millions per annum in Python analytic pipelines aimed at getting Python analytics into the hands of business users. For BI and analytics of the past, executives at blue chip industries went to Looker, Tableau, or Power BI. For the AI analytics that future industries need, business executives at companies like Tesla, Apple, Moderna, and Colgate have turned to Python.

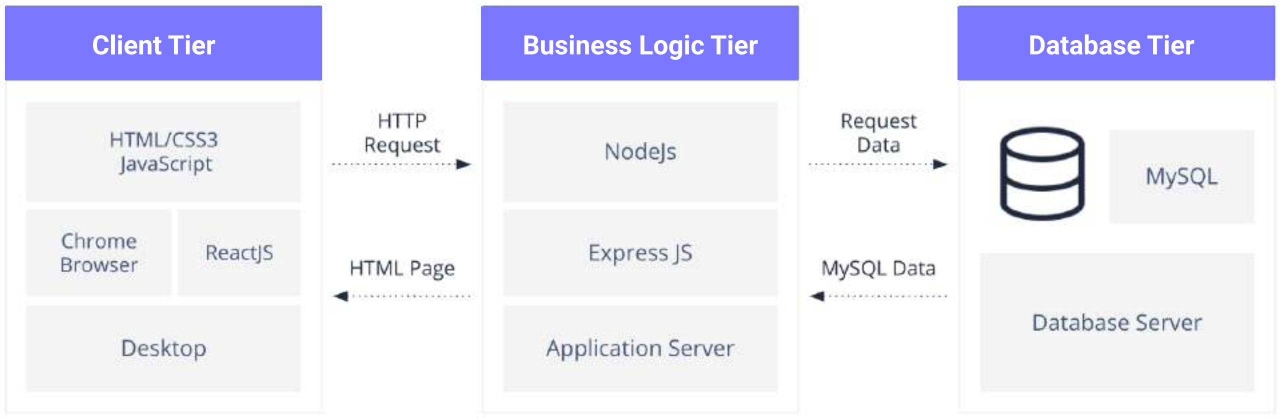


Source: Stack Overflow

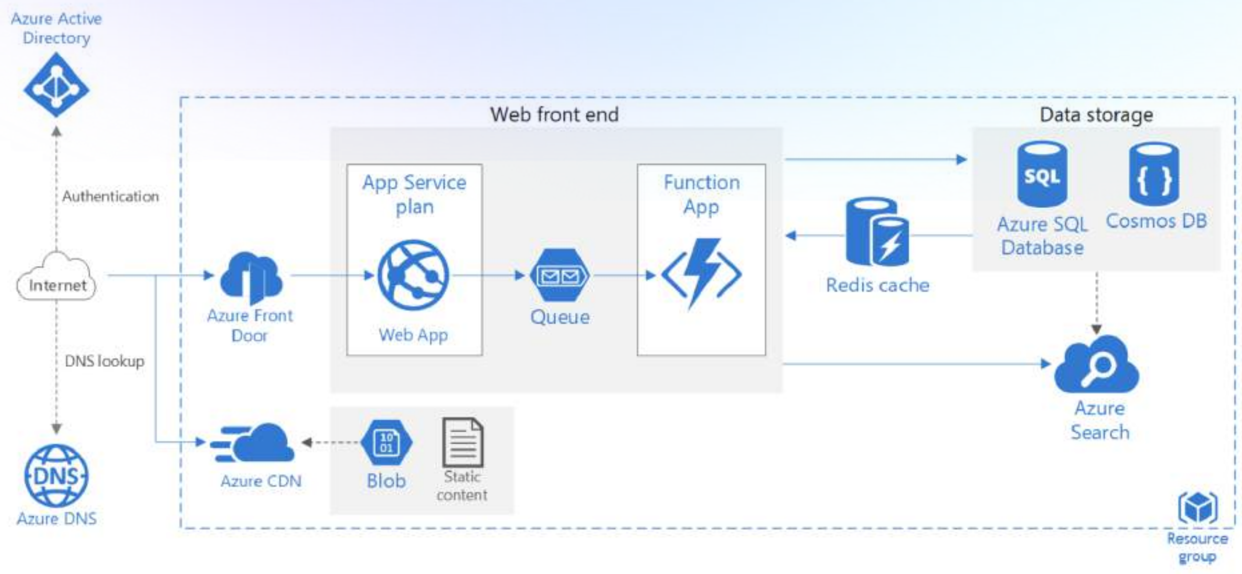
Part II: Full-Stack Application Primer

Today, “getting Python analytics into the hands of business users” typically means building a “full-stack” application. What is a “full-stack” application, you ask? Full-stack development is the software paradigm by which web applications are made. Facebook is a full-stack app, so is Twitter, or even a rental car agency website. In a full-stack app, you have the front end written in HTML, CSS, and JavaScript. The front end is the part of the full-stack app with which you interact. It runs in your web browser.

You also have the “back end,” which is written in languages like Python, ASP.NET, Scala, or Go. The back end runs on a server and processes requests from the front end. The back-end code is also often the gatekeeper to some kind of database. For example, say you request a car on a rental car agency website. The front-end code (JavaScript) responds to the request, sends it to the back end (Python, ASP.NET, Scala, etc). The back end checks in a database whether this car is available, then the front-end dispatches the result. The front end and back end together make up a ‘full-stack” web application.



*An example of a **simplified full-stack architecture diagram**, where NodeJS is the back-end language, ExpressJS is the back-end web server, ReactJS is the front-end (JavaScript) framework, and MySQL is the back-end database. Part IV describes how Dash Enterprise collapses this stack so that a data scientist can easily own all 3 tiers.*



An example of a more ***realistic full-stack architecture diagram****. This diagram takes into account additional moving parts such as an in-memory cache (Redis), a jobs queue, DNS configuration, and corporate authentication (Active Directory). This full-stack architecture diagram is particular to building on Microsoft Azure. If your company is using Azure, please note that Dash Enterprise is available on* ***Azure Marketplace****.*

"Full-stack developers" are a relatively rare breed of engineer that can work in both the front end (JavaScript) and back end (Python, ASP, .NET, etc) languages. In addition, they must understand how the back-end web server works (such as Flask or Django in Python) and how to deploy these applications to a cloud server when the app is ready for prime-time, mass consumption. It's hard to be good at everything, so most developers tend to specialize. By and large, you're either a "front-end engineer" or a "back-end engineer". Finally, an IT or "DevOps engineer" helps deploy the finished full-stack app on a cloud server for mass consumption.

Part III: The Rise of the Analytics Application

To deliver the rich and powerful Python analytic stack to business users, standard practice among today's F500 companies and all companies with over 500 people is to use data to make business decisions. To do this, they need to use an analytics application. The back end of this application ties into a data science or machine learning model. Through the front end, business executives or operators run the model through a custom user interface. Suddenly, the Python analytics goodness that every data scientist has been talking about is in the hands of every business executive. Facial recognition, advanced forecasting, natural language processing, cohort clustering...these Python-driven AI capabilities can be very powerful when effectively made available to a company's business team. Analytics applications are the most common way to get them into a business user's hands.

The problem with building an analytics application using full-stack engineering practices is that it's slow and expensive to build and maintain. At a minimum, you need 5 senior engineers:

- **One data scientist** to make the models (if you're a large company you likely have a team) that deliver value to the business.
- **One back-end engineer** who writes the web server code and connects the models to the web server.
- **One front-end engineer** who writes the web user interface for the business operator to interact with the model.
- **One DevOps or IT engineer** who deploys the app to an internal server, investigates when it goes down, and helps deploy updates as feature requests and bug reports roll in.
- **One program manager and/or designer**. The PM keeps everyone above on track and on schedule. The designer designs the user interface for the front-end developer. If you're on a budget, you can get away without one of these roles.

At a minimum, you're looking at 5 senior engineers to bring your analytics into the hands of your users. At a F500, call the average fully loaded salary of each of these engineers $150k/year — that's a **$750,000/year** minimum buy-in to be able to build analytical applications at all.

In reality, there are going to be several data science and ML models that your leadership team wishes to operationalize for the business. Depending on the app complexity, you might be able to spread 1 engineer across 3 apps, but most likely you'll need to hire more engineers. The full-stack software paradigm thrives for companies building 1 external-facing, revenue-generating product (e.g., Facebook). It wasn't meant for companies deploying dozens of internal-facing, revenue-consuming analytic apps.

Due to the ephemeral nature of analytics, traditional full-stack development is not an ideal approach for internal, analytics apps. When you have a problem, you solve it through Python analytics operationalized as an app, and then you move onto the next problem. New business problem = new analytics app = new $750,000/year team.

Through this lens, it's easy to see how advanced analytics budgets for F500s can run in the tens of millions of dollars per annum, or, more commonly, how great opportunities just don't get followed up on for lack of capacity.

Part IV: The Life Changing Magic of Dash & Dash Enterprise

Plotly encountered the problem of applying full-stack techniques to analytical applications in the late 2000's. On one hand, they found analytical apps to be a very effective medium for delivering Python analytics to the business team and lab technicians. On the other hand, they worked 80-hour weeks keeping the web servers running, writing the data science Python code, and writing the front-end (JavaScript) interface for the business users and technicians. As non-software engineers, they were good at writing data science code, but terrible at writing JavaScript and web server code. Their applications performed and delivered massive value to the business, but the apps were a gigantic task to write and maintain.

Plotly was motivated by the transformative effect of putting Python analytics in the hands of business users, but they knew from experience that full-stack development was an overly burdensome and inefficient way to do it. With this in mind, the company created Dash, Dash Enterprise, and the world's most downloaded, interactive Python data visualization libraries to make data science, AI, and ML more universally actionable.

Dash and Dash Enterprise reduce the 5-person, full-stack development team needed to build and deploy analytical apps to only 1 person (the data scientist, analyst or engineer). Moreover, Dash apps are better in every way than the artisan, hand-crafted apps built by a 5-person, full-stack team. Compared to most analytical apps, Dash apps load faster, look better, crash less, scale horizontally, don't have security vulnerabilities, and update in minutes rather than days. This is because Dash is built upon Plotly's decades of experience in building these kinds of apps. We've taken the best front-end technology (React), the best back-end technologies (Flask, Kubernetes, stateless design), and the best data visualization libraries (Plotly Express, Datashader), and packaged them so that a beginner data scientist can build and deploy their first analytical app in a couple of hours. No CSS required. No JavaScript required. And no DevOps required. All you need is a data scientist. Dash and Dash Enterprise supercharge the data scientists by giving them a PaaS to rapidly deploy world-class web applications without needing full-stack development skills.

Dash is the open-source software that lets you build analytical apps on your desktop, or from your browser. Dash Enterprise is the PaaS ("Platform as a Service") that provides you with advanced development tools for faster prototyping, lets you deploy these Dash apps to a server where your entire company can log in and run Python models in real time through an easy to use user interface, provides support of Single Sign On (SSO), security, scalability, customer support, and a host of other features. Let's look at the minimum analytical app team again and see why you only need a single data scientist with Dash Enterprise (vs. 5 full-time engineers).

- **One data scientist.** The data scientist writes the data science, ML or AI models that will transform the company's business operation. Because they know Python, Dash gives them the full front-end and back-end analytical app creation capabilities. You no longer need front-end and back-end engineers if your data scientists are equipped with Dash.
- **One back-end engineer.** Under the hood, Dash gives the data scientist a best-in-class web server, but abstracts all of the messy details away, so that the data scientist can focus on their core modeling work. Trust us — we've been doing this for decades and through Dash, we're giving data scientists the best of what's out there for apps.
- **One front-end engineer.** Under the hood, Dash spins up a single-page React application on the front end. As a Python programmer, the data scientist doesn't need to know what React is. All they need to know is that it is the best modern front-end framework and that Dash gives it to their app for free.
- **One IT/DevOps engineer.** Dash Enterprise allows the data scientist to deploy their Dash apps to a server for mass consumption with a simple, 1-line command. There is no need to throw the code over the fence to IT every time you want to update the app. Dash Enterprise takes IT out of the loop, so data scientists can deploy and update their apps autonomously.

  Dash Enterprise automatically "hot swaps" containers upon deployment so that there is never downtime when you deploy or update an app.

  In the full-stack paradigm, to deploy a single app to a single server with downtime, you may only need 1 DevOps/IT engineer. If you also want authentication/authorization with single-sign-on, zero downtime deployments, horizontal scalability, security patches and other "must-haves" to build enterprise-grade analytical apps, **you'll need at least 4 DevOps engineers**. In our experience, to deliver a successful AI or data science initiative, you don't want to skimp here. Business users don't like downtime or crashing apps. This balloons the minimum **$750,000** analytical app engineering team cost to **$1,350,000/year**. F500s will sometimes couch this DevOps team cost in external Azure or AWS consultants.

- **One designer.** Dash Enterprise ships with Dash Design Kit, an Adobe-like GUI for styling Dash apps with preset themes and templates. With Design Kit, the data scientist will create apps that look better than if you had a full time designer, because our team of designers have already put thousands of hours into the Design Kit templates and have worked in collaboration with F500 brand teams.
- **One project manager.** With Dash and Dash Enterprise, your analytical app team is now a single data scientist, so you don't need a project manager.

Part V: Why Full-Stack Teams Often Struggle

Most AI, ML, or data science models that could transform large companies' bottom lines are never integrated into the business operation. The models never make it from the lab to the business, despite millions of dollars of investment in the model's development. This is because the de facto way of operationalizing these models, the full-stack approach, is risky and costly.

"Even in locked-down environments, we're able to build an app in 3 just days with Dash, something that is impossible with full-stack platforms."

-F100 Energy Company

Here are the most common ways that full-stack analytical app initiatives fail to get off the ground, or if they do, fail to soar:

- **Staffing Cost:** In Part III, we baselined **$750,000/year** for a F500 full-stack team with the skills to operationalize analytical apps, plus an additional **$600,000/year** for IT niceties such as zero downtime, deployments, continuous integration, a password/API key vault, and horizontal scalability. Above $1M, the AI initiative must show business value, but often the first idea is not the right one, or business needs change, rendering the first model obsolete. Many operationalized data science initiatives deliver multiples on >$1M/year investments, but not all succeed on the first try. Compare this with the staffing costs of building Dash applications: **a single $150,000/year data scientist**.

- **Iteration Speed:** Even with an infinite budget for a 5-20 person full-stack development team, all the moving parts and coordination in a full-stack initiative make building and updating the app slow: **simple updates can involve multiple meetings and take days**. Building the app in the first place will take months. Compare this development cycle to that of Dash Enterprise: developing a Dash app for a finished model takes days, deploying it takes minutes, and scaling it horizontally across multiple servers takes seconds. After a Dash app is deployed and operationalized for the business, updating it takes minutes instead of days. This can all be done by a single data scientist, so there is no need to have an all-hands SCRUM meeting when the CTO demands an update to the app. The executive can just email the data scientist and minutes later, the updates are live. Dash Enterprise integrates with all continuous integration platforms (like CircleCI, so automated testing and quality assurance gates can be easily implemented for deploying updates to mission-critical Dash apps. Moreover, Dash Enterprise uses git to version control Dash apps, so rolling back to previous Dash app versions is simple. The bottom line is that **teams using Dash Enterprise are able to ship initial apps and subsequent iterations 5-10 times faster**, and this means that the same small number of data scientists (often just one!) can work on many more apps at once than a bigger team can.

- **Outcomes:** In a traditional, full-stack paradigm, even though a data scientist is nominally “part of the team”, **in practice siloization sets in** and the communication between data scientists and developers takes on more of a “throw it over the wall” dynamic. This results in analytical apps largely built by developers who don’t understand the nuances of the data and models, and this can lead to applications which don’t enable end users to get access to the insights they need. The net result is applications that fail to get traction, and fail to result in impacts on the business. Contrast this to the Dash Enterprise approach, where the data scientist who did the modeling and intimately knows the details of the data and the users is the one building the app. Nothing is lost in translation, and apps correspond more closely to their users’ needs. This is what is needed to **avoid being one of the 3 of out 5 businesses who fail to report any business gains from AI**, as reported by MIT Sloan and BCG.

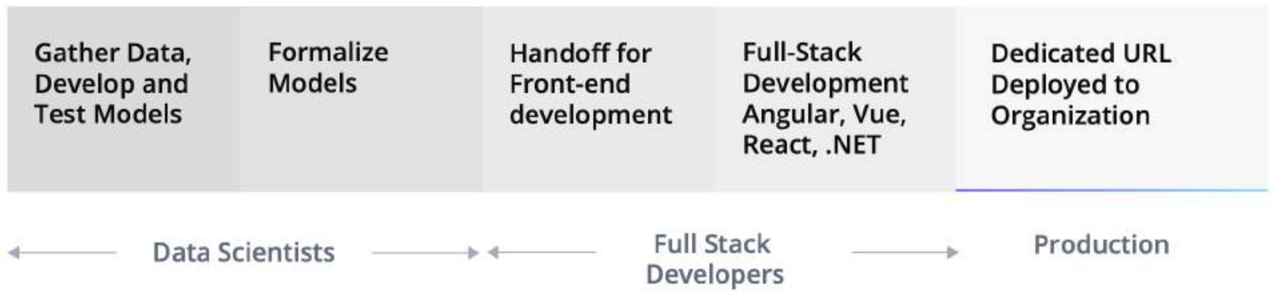
- **Scope:** Analytical apps have many needs that traditional FSAs do not, especially if ML, high performance computing (HPC), or big data pipelines are involved. A full-stack app that runs a car rental website is complicated, but substantially less so than a full-stack app that asynchronously runs an ML model on real-time data and sends a report when the model is done. So, when you hire an ace full-stack crew with decades of experience, you may find that their skills don’t cut it. A full-stack app for running Python-based ML or data science models will need ML, HPC, big data pipelines and much more. Each of the following features will need at least 1 full-time intermediate developer building and maintaining them:

- A centralized "Portal" for the business team to access the analytics applications
- Integration with you company's authentication system
- Job queues
- High Performance Computer (HPC)
- Big Data processing and pipelines
- Embedding middleware to embed analytical apps in legacy reporting software or Salesforce
- Automated report scheduling, reporting emailing, and PDF generation
- Database cache. Dash Enterprise ships with onboard Redis and Postgres databases that you can use as caches for your deployed Dash applications.

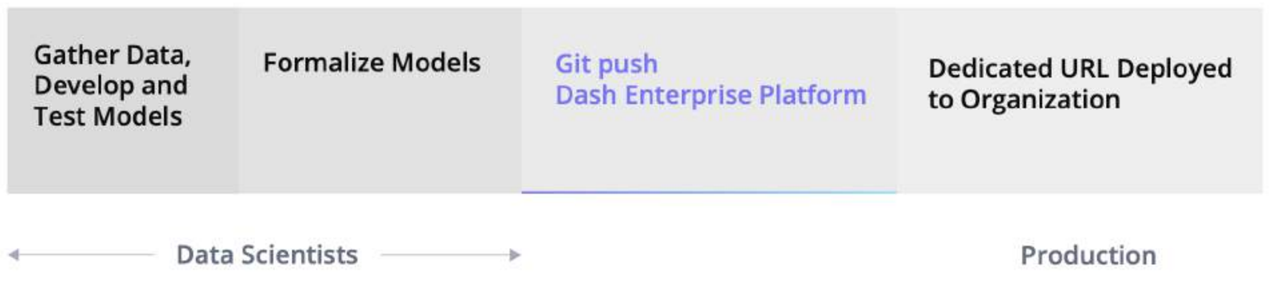
All of these capabilities will take 1 engineer a year to build, and then 50% of their time to maintain. For all of these capabilities, all the build and maintenance cost **$750,000/year** in engineering salaries for a platform team.

All of these are features of Dash Enterprise. We've spent years building and perfecting them with a 30 person engineering team. You will likely find out the hard way that you need them if you build a full-stack AI application at your company. Even if you don't strictly need these capabilities, you'll be shorting yourself the full potential of the Python analytics ecosystem without them. Instead of spending $750,000/year and losing a year building them, you can get them immediately with Dash Enterprise.

Full-Stack Development Process


Gather Data, Develop and Test Models
Formalize Models
Handoff for Front-end development
Full-Stack Development Angular, Vue, React, .NET
Dedicated URL Deployed to Organization
Data Scientists
Full Stack Developers
Production

Dash Enterprise Development Process


Gather Data, Develop and Test Models
Formalize Models
Git push
Dash Enterprise Platform
Dedicated URL Deployed to Organization
Data Scientists
Production

Part VI: Cost Table

The starting price point for Dash Enterprise is $50,000/year. This includes authoring and deployment access for up to 5 data scientists and *UNLIMITED* end users. This means that a single data scientist at Ford Motor Company can build and deploy *UNLIMITED* Dash apps to 10,000 employees, and the price is still $50,000/year. The incremental cost of deploying Dash applications is $0.

Compare this to the traditional full-stack approach for operationalizing Python analytics:

	Traditional Full-Stack Approach		**Dash Enterprise Approach**		**Dash Enterprise Cost Advantage**
# of apps	Team composition to deliver and maintain apps	Staffing costs per annum (whole team)	Staffing costs per annum (1 data scientist)	Dash Enterprise yearly license costs	
1-3 apps	Full-stack team	$750k (5 FTEs)	$150k (1 FTE)	$50k	3.75x (27%)
1-3 apps	Full-stack team + DevOps team	$1.35M (9 FTEs)	$150k (1 FTE)	$50k	6.75x (15%)
4-10 apps	Full-stack team x2	$1.5M (10 FTEs)	$150k (1 FTE)	$50k	7.5x (13%)
4-10 apps	Full-stack team x2 + DevOps team x2	$2.7M (18 FTEs)	$150k (1 FTE)	$50k	13.5x (7%)
10-15 apps	Full-stack team x3 + DevOps team x2 +platform team	$4.2M (28 FTEs)	$150k (1 FTE)	$50k	**21x (<5%)**

If you'd like to discuss how to increase your likelihood of success while saving costs for your AI initiative with Dash Enterprise, ***please get in touch*** *with our sales or executive team.*

Part VII: Accelerating a Pre-Existing Full-Stack Team with Dash Enterprise

We speak with many team leads at F500 companies who have already hired a full-stack development team to build analytic apps with Python back ends. In this case, the Dash architecture and "way of doing things" will still reduce your dev cycle time by an order of magnitude and de-risk your app delivery timeline, performance, value to the business and security vulnerability surface area. With Dash Enterprise, everyone on your full-stack development team has an important role to play, but the efficiency in overall analytic app output and quality will improve 10x.

Front-end Engineers: In the Dash framework, front-end engineers can create Dash components for the data scientists to consume in their Dash analytic apps. Rather than write an entire React front end complete with Redux, React hooks, and AJAX handling, the front-end engineers focus on user interface ("UI") elements for the data scientists to build their Dash apps. This is because Dash takes care of all the other front-end boilerplate for you. Any React component can be turned into a Dash component through Dash's React-to-Dash component pipeline. Many companies already have a set of React components as part of their UI and brand guidelines (see Palantir's Blueprint, for example). Dash makes it easy to "export" these components as Python libraries for easy import into Dash apps. Your front-end engineers are the ones who can create and mainline this customized Dash components suite, rather than maintain an ocean of React boilerplate code.

IT/DevOps Engineers: Dash Enterprise can also be installed on a single-node, on-premises Linux server. Your IT or DevOps engineers can accelerate and optimize the Dash Enterprise installation and configuration. For example, autoscaling rules, corporate authentication, mail server integration (for auto-emailing Dash reports), DNS configuration, continuous integration, and Dash Enterprise's free, quarterly update patches are a few ways that IT/DevOps accelerate and optimize delivering AI, ML, and data science applications through Dash Enterprise.

Designer: Dash Enterprise ships with an onboard styling GUI for Dash apps, Dash Design Kit. Design Kit enables pixel-perfect, point-&-click styling of Dash applications created by the data scientists. Design Kit also exports and saves themes so the designer can create one theme that matches corporate brand guidelines and apply it to all of the data scientists' Dash apps. Finally, if the front-end engineers create their own suite of customized Dash components, the designers can help perfect their styling, UX, and on-brand appearance.

Back-end Engineers: Just like Dash takes care of the front-end boilerplate, Dash also takes care of the back-end web server boilerplate. This frees up your back-end engineer to implement and fine-tune Dash Enterprise's MLOps features, rather than writing a boilerplate that has already been written hundreds of times and adds no intrinsic value to your AI and advanced analytics initiatives.

Conclusion

Spend resources delivering AI, ML, and data science models to the business; rather than writing & maintaining onerous full-stack web application code.

In the Dash Enterprise paradigm, your full-stack team is closer to the core AI, ML, or data science software that you are trying to operationalize, because all of the full-stack boilerplate has already been written for you. Companies get no value from their full-stack data science teams writing and maintaining an ocean of boilerplate web software in order to productionize an AI model. Let Dash Enterprise do that boilerplate for you in an optimized way, so that you can productionize more AI applications faster and with less complexity, less IT security risk, and less financial risk.

If you lead the productionizing of an AI model at your business, **go light and fast with Dash Enterprise**. Within one month of installing Dash Enterprise, you'll have deployed your first Dash app (or several). Quick results at a low cost will give you more managerial trust funding to transform even more of the business with productionized Dash apps for AI, ML, and data science models.

About Plotly

Plotly is a software company whose mission is to enable every company, around the world, to build data applications. Our product, Dash Enterprise, is a platform of best-in-class development tools to quickly and easily visualize data in Python from virtually any data warehouse source. With over twenty Fortune 500 and over 150 enterprise customers, Plotly is a category-defining leader in enabling data-driven decisions from advanced analytics, machine learning, and artificial intelligence. For more information, visit https://plotly.com/.