Backstage as a central platform for an ML Capability

Backstage, open-sourced by Spotify, is a single repositary for organising digital assets.

colum.mccoole@analect.com
colum-mccoole-746b946a
analect/technical-docs-hierarchy

Docs: Level 2 AI/ML Value-chain

Topic Navigation

How does this topic relate to other relevant content in this space.

What Problem Are We Solving For?

Any journey into the world of machine learning necessarily means scaling - handling lots of code, developing lots of models, experimenting alot and a requirement to keep track of all this, while documenting it too, for others to follow and understand.

Backstage isn't a purpose-built platform for machine learning, as such. In fact, it characterises itself as an open platform for building developer portals. In essence, it's a powerful set of primitives or framework that allows an organisation to structure it to their needs and can help those developing software (including pipelines and models) to manage, document and monitor the disparate digital assets, often cloud-based, that go with such implementations.

Backstage, which can act as a binding layer for the architecture, dev, ops and AI pillars that we articulate on these pages.

Key out-of-the-box Functionality

- a **software catalogue** - the objective is to map all software assets (entities) in your organization, including websites, APIs, libraries, and resources, in a centralized directory.
- **software templates** - a *scaffolder* provides your developers with the ability to execute software templates that initialize repositories and might also handle ML pipelines and models. Scaffolder assets are automatically added to the Catalogue.
- a **documentation generator** - *TechDocs* is the framework's documentation-as-code solution; it takes markdown files and transforms them into static pages.
- a Kubernetes **cluster visualizer** - Backstage ships with a plugin that helps developers visualize the state of the clusters for each service.
- cross-ecosystem **search capabilities** - allows developers to find information across their ecosystem.

Open-source `Backstage` as a Platform

The proliferation of systems and their underlying resources are becoming harder to coordinate, which is why I think frameworks like Backstage, with their plugin mechanism, are becoming indispensible. Based on my own usage, so far, here are some of my takeaways:

1. **Flexible Front-end** - iterate quickly with prototypes (rather than a polished UI) to expose information drawn from connected systems - only surfacing what's critical and giving users within Backstage a possibility to hop to that system's UI directly, while at the same time managing to remain with the context of their task. Front-end plug-ins (using React and MaterialUI) can be customised, for extending in ways beyond the publically available plugins.
2. **No Context Switching** - related to the above, instead of switching between all these different tools and dashboards and systems and UIs, there's just one frontend for all of it — a single pane of glass.
3. **Experimentation / Documentation in one place** - per the Software Catalog (for building golden pathes or templating pipelines) and Tech Docs, model experimentation and documentation are contained in one place.
4. **Links back to assets** - code pertaining to models as well as infrastructure-as-code required to run-up the infrastructure on which a model is trained or served, is managed alongside eachother, with visibility into the running deployments, whether they be on a kubernetes cluster or on some cloud-based container platform. This could also extend to using Backstage as an external window into systems like Glue, for data-management and relating how data is utilised back to other components like APIs documented via Backstage.
5. **Governance Framework** - related to both points above, Backstage has the potential to bring transparency to an IT system, not only for internal purposes, but also potentially for addressing regulatory issues around AI and model explainability, where auditors are presented with a self-documenting platform.
6. **Self-serve** - the idea behind Backstage, is for users to self-serve using certain golden pathes or tried-and-trusted templates that are used to deploy workloads or pipelines. A platform engineering team has pre-defined how new models are structured and the system self-documents itself out-of-the-box. Have a look at this AWS Proton plugin (https://github.com/awslabs/aws-proton-plugins-for-backstage#demo) developed by AWS themselves, for a good study of how this self-serve approach can work in a cloud context.
7. **Plugin Ecosystem** - There's a rich ecosystem of third-party contributed plugins (https://backstage.io/plugins/) that is expanding all the time. I'll highlight some of the more interesting use-cases I have come across that involve handling serverless workloads in the next sections.

Zalando Uses Backstage as an Internal Dev Platform

Zalando (https://zalando.com) are an online fashion retailer that have a long history of using novel technologies, including machine learning, in their business. They were early adopters of Backstage as a mechanism around which to build much of their developer tooling, including that pertaining to their ML platform.

Their instance of Backstage, which they brand as Sunrise, is proprietary, but they have blogged and presented the ideas behind their solution, which can be found here:

- Sunrise: Zalando's developer platform based on Backstage (Aug. 2023)- https://engineering.zalando.com/posts/2023/08/sunrise-zalandos-developer-platform-based-on-backstage.html
- Sunrise - Zalando's Internal Developer Platform - https://platformengineering.org/talks-library/sunrise-zalandos-internal-developer-platform

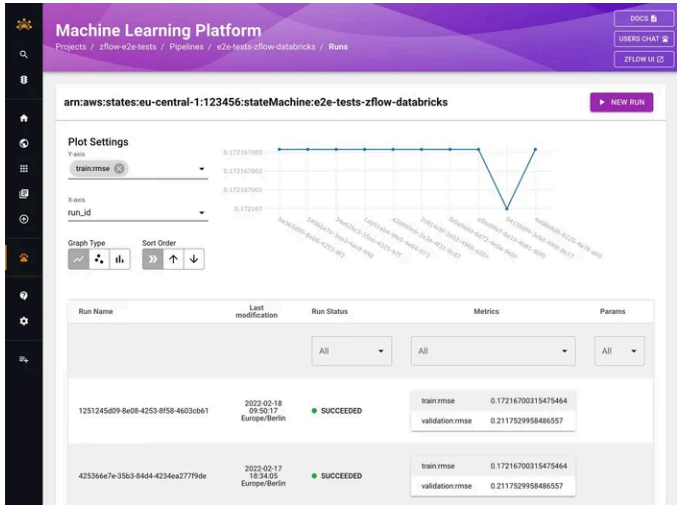

Figure 1. Zalando - a screenshot of their Machine Learning Platform, part of their proprietary Sunrise solution built on Backstage