Feature flags and licensed products

Introduction

This document outlines the distinct purposes and use cases of feature flags and licensed products within our software. It is designed to provide clarity on how these two tools should be utilised by different teams in our organisation, ensuring that each team can leverage them effectively to meet their specific goals.

Feature flags: for engineering and product teams

Purpose of feature flags

Feature flags are a tool primarily used by engineering and product teams to manage the deployment and availability of features in our software. They allow for dynamic control over which features are enabled in production environments without requiring code changes or redeployment.

Use Cases for feature flags

- **Controlled rollouts**: Gradually release new features to a small subset of users, allowing for monitoring and testing before a full release. For example, the attendee webapp currently sends the `organizationPaymentPlan` attribute to LaunchDarkly.
- **A/B testing**: Experiment with different versions of a feature to see which performs better with users, based on real-world usage data.
- **Instant rollback (aka kill switch)**: Quickly disable a feature if it causes issues, without the need for a hot-fix or emergency deployment.
- **Continuous deployment**: Deploy features incrementally and enable them when they are ready, allowing for faster, more reliable software releases.

How engineering and product teams should use feature flags

- **Feature development**: When developing a new feature, wrap it in a feature flag to control its visibility. This allows you to merge code into the main branch without making the feature live immediately.
- **Testing in production**: Use feature flags to enable features for internal users or beta testers, allowing real-world testing without exposing the feature to all users.
- **Gradual rollouts**: Start by enabling the feature for a small percentage of users, gradually increasing this number as confidence in the feature grows.
- **Monitoring and analytics**: Utilize LaunchDarkly's monitoring tools to track the performance and usage of feature-flagged features. This helps in making data-driven decisions about the feature's future.
- **Clean-Up**: Regularly review and remove unused feature flags to reduce technical debt and maintain a clean codebase.

Licensed products: for sales and customer success teams

Purpose of licensed products

Licensed products (formerly known as "payment plan features") are designed to manage the specific features and capabilities available to individual customers based on their subscription, contract, or licensing agreement. This tool is primarily used by sales and customer success teams to tailor the software experience to the needs and agreements of each customer.

Use cases for licensed products

- **Custom offerings**: Provide tailored software packages to different customers based on their specific needs, industry requirements, or budget.
- **Tiered pricing models**: Implement tiered pricing models where different levels of access and features are provided based on the customer's subscription plan (e.g., Basic, Pro, Enterprise).
- **Contract compliance**: Ensure that customers only have access to the features they have paid for, in accordance with their licensing agreements.
- **Feature add-ons**: Enable or disable additional features based on customer purchases or upgrades, without needing engineering intervention.
- **Renewals and upsells**: Use data on feature usage to inform renewal discussions or identify opportunities for upselling additional features or higher-tier packages.

How sales and customer success teams should use licensed products

- **Onboarding new customers**: Assign the correct licensed product to each new customer based on their contract. This determines the specific set of features they will have access to.
- **Managing upgrades**: When a customer decides to upgrade their plan or purchase additional features, update their licensed product configuration to reflect these changes.
- **Customer support**: Use licensed products to quickly identify what features a customer should have access to when resolving support queries or issues.
- **Contract renewal**: Review the customer's licensed product configuration and feature usage before renewal discussions. This allows you to propose relevant upgrades or changes to their plan.

- **Compliance monitoring**: Ensure that customers are compliant with their licensing terms by regularly checking that their licensed product configuration matches their contract.

Keeping feature flags and licensed products separate

To maintain clarity and efficiency:
- **Engineering and product teams** should focus on using feature flags for managing the development, testing, and gradual release of features.
- **Sales and customer success teams** should focus on using licensed products to configure and manage what features are available to each customer, based on their specific licensing agreement.

By keeping these tools distinct, we ensure that each team can operate effectively within their domain while contributing to the overall success of our product and customer satisfaction.

This means that, if a feature is not (or might not always be) available to all customers. the code might sometimes need to check for a feature flag **and** a licensed product. In those cases, once the feature has been fully released, the feature flag can disappear (unless we want to keep it as a kill switch for extra safety), but the licensed product will always remain.

As a general rule, we should not use feature flags to target specific customers, except in these particular scenarios:

- Internal testing in production, e.g. for specific organisations
- Gradual roll-out, e.g. enable for free plans only before GA

For deciding between feature flag and licensed product, the question should be "who is going to modify it?", and sometimes we will need both.

Conclusion

Feature flags and licensed products are powerful tools that, when used correctly, can greatly enhance the flexibility and responsiveness of our software. By clearly defining their roles and usage within our organisation, we can ensure that each team can maximise their impact while maintaining a seamless and efficient workflow.