Feature Flags (Feature Toggles) - Explained

What are Feature Flags?

Feature Flags (also known as **Feature Toggles** or **Feature Switches**) are a software development technique that allows you to enable or disable specific features in your application without deploying new code. Think of them as on/off switches for features in your software.

How Do They Work?

Instead of hardcoding whether a feature is available, you wrap the feature code with a conditional check:

```
if (isFeatureEnabled('LESSON_MANAGEMENT')) {
  // Show the Lessons tab
  showLessonsTab();
} else {
  // Hide the Lessons tab
  hideLessonsTab();
}
```

The flag's state (enabled/disabled) is typically stored in:
- A database
- A configuration file
- A feature management service
- Environment variables

Why Use Feature Flags?

1. Gradual Rollouts (Canary Releases)

- Deploy a feature to 10% of users first
- Monitor performance and user feedback
- Gradually increase to 50%, then 100%
- Roll back instantly if issues arise

2. Premium Features / Monetization

- Offer different subscription tiers (Basic, Pro, Enterprise)
- Enable premium features only for paying customers
- Example: In your driving school app, "Lesson Management" is a premium feature that requires a license

3. A/B Testing

- Show Feature A to 50% of users
- Show Feature B to the other 50%
- Measure which performs better

4. Risk-Free Deployments

- Deploy code with new features turned OFF
- Test in production with the feature disabled
- Enable the feature when ready (no new deployment needed)
- Quickly disable if bugs are discovered

5. Development & Testing

- Developers can work on incomplete features in the main codebase
- Features stay hidden from users until ready
- QA can test features before public release

6. Regulatory & Compliance

- Disable features in specific regions
- Example: GDPR compliance - disable certain data collection in EU

7. Maintenance & Emergency Response

- Quickly disable problematic features
- No need to redeploy the entire application
- Useful during incidents or outages

Types of Feature Flags

1. Release Flags (Short-term)

- Used during feature development
- Removed after feature is fully released
- Lifespan: Days to weeks

2. Business Flags (Long-term)

- Control premium features
- Used for monetization
- Lifespan: Months to years (or permanent)

3. Ops Flags (Long-term)

- Performance optimization switches
- Circuit breakers for external services
- Lifespan: Permanent

4. Experiment Flags (Medium-term)

- A/B testing and experiments
- Removed after conclusions are drawn
- Lifespan: Weeks to months

Real-World Example: Your Driving School App

In your application, we use feature flags for premium features:

```
// Feature definitions
const FEATURES = {
  STUDENT_ACCESS: 'PREMIUM',        // Premium feature
  VEHICLE_MANAGEMENT: 'PREMIUM',    // Premium feature
  LESSON_MANAGEMENT: 'PREMIUM',     // Premium feature (NEW!)
  SMS_NOTIFICATIONS: 'PREMIUM',     // Premium feature
  // ... more features
};
```

How it works in your app:

1. **Check License Status**: When a user logs in, check which features their license includes
2. **Enable/Disable Features**: Show or hide UI elements based on enabled features
3. **Enforce Access Control**: API endpoints verify feature access before processing requests

Example in Navbar:

```
// Only show "Lessons" tab if feature is enabled
if (isFeatureEnabled('LESSON_MANAGEMENT')) {
  navItems.push({ label: "Lessons", href: "/admin/lessons" });
}
```

Example in API:

```
// Block API access if feature not enabled
const featureCheck = await checkFeatureAccess('VEHICLE_MANAGEMENT');
if (!featureCheck.allowed) {
  return error('Feature not enabled. Please upgrade.');
}
```

Benefits for Your Business

1. **Upselling**: Encourage customers to upgrade to unlock premium features
2. **Flexible Pricing**: Offer different tiers (Starter, Professional, Enterprise)
3. **Trial Period**: Let customers try features before buying
4. **Instant Activation**: Customers get immediate access after purchasing
5. **No Redeployment**: Enable features for customers without touching code

Best Practices

✅ DO:

- Use clear, descriptive flag names (`LESSON_MANAGEMENT` , not `feature_123`)
- Document what each flag controls
- Clean up old flags after features are fully released
- Use a centralized feature flag management system
- Track flag usage and dependencies

❌ DON’T:

- Create too many nested flags (increases complexity)
- Leave temporary flags in production for months

- Forget to test both enabled and disabled states
- Use flags for business logic (only for feature availability)

Tools for Managing Feature Flags

Commercial Services:

- **LaunchDarkly**: Enterprise-grade feature management
- **Split.io**: A/B testing and feature flags
- **Optimizely**: Experimentation platform
- **Unleash**: Open-source feature toggle service

Simple Approaches:

- Database table (like in your app)
- Configuration files (JSON/YAML)
- Environment variables
- Admin dashboard (like your License page)

Your Implementation

Your driving school app uses a **database-backed** feature flag system:

1. **Feature Definitions**: Defined in `lib/config/license-features.ts`
2. **License Management**: Stored in database per organization
3. **Client-Side Checks**: Using `useLicense()` hook
4. **Server-Side Checks**: Using `checkFeatureAccess()` middleware
5. **Admin Control**: Managed via the “License” page

This gives you full control over which customers can access which features!

Summary

Feature flags are like **light switches for your software features**. They give you:
- **Control**: Turn features on/off instantly
- **Flexibility**: Different features for different customers
- **Safety**: Test in production without risk
- **Revenue**: Unlock premium features for paying customers

In your driving school platform, feature flags enable you to offer tiered pricing, control access to premium features like “Lesson Management,” and provide immediate value to customers who upgrade their license.

Questions? Feature flags are a powerful tool that separates what code is deployed from what features are active. This separation gives you incredible flexibility in how you deliver and monetize your software!