



 tarent solutions GmbH

Services migrieren - Tücken, Tricks und 3 Fallbeispiele



Services migrieren - Tücken, Tricks und 3 Fallbeispiele



Kurz zu mir ...

- Tim Steffens, **Softwareentwickler** und **Trainer** bei der  **tarent**
- Interessen
 - **Funktionale Programmierung**
 - **Qualität** von Software
 - **Wandern** & Klettern
- die **tarent**
 - **hilft Kunden** gute Software zu bauen
 - in-house oder vor Ort
 - Schulungen
 - baut **eigene Software** (z.B. snabble.io)
 - hat Standorte in **Bonn, Köln, Berlin & Bukarest**



Wer seid Ihr?

A person is sitting on a rocky cliff, looking out over a vast sea of clouds. The sun is setting in the distance, creating a warm, orange glow on the horizon. The clouds are thick and white, filling the lower two-thirds of the image. The person is wearing a dark jacket and shorts, and is looking towards the right side of the frame.

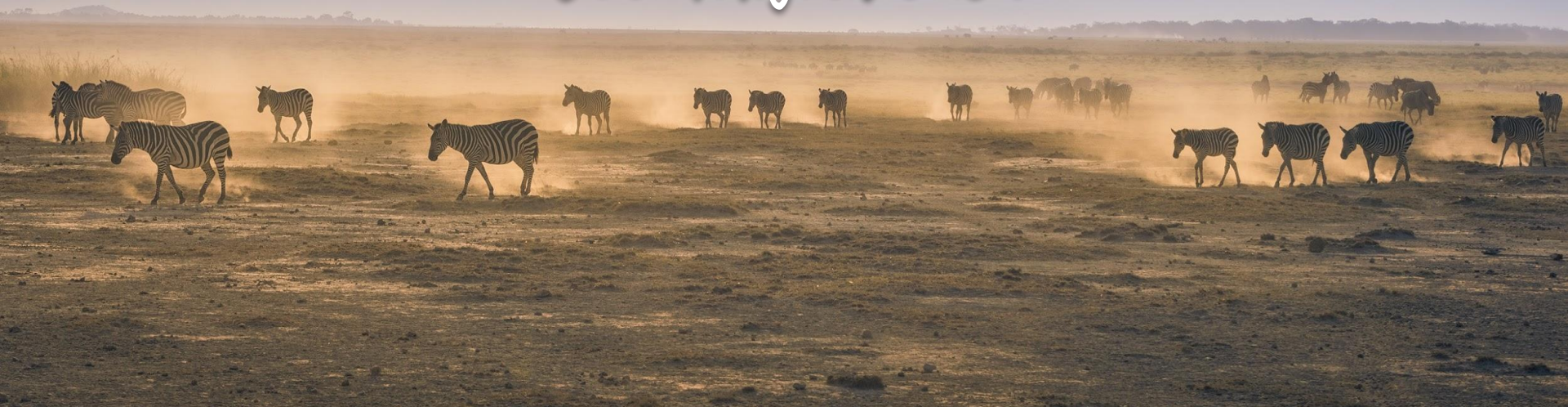
Agenda

- Über Migrationen
- 3 Beispiele
- Vergleich
- Key Learnings

An aerial photograph of a vibrant green lake. Several small, irregularly shaped islands or peninsulas are visible, characterized by a mix of green vegetation and brownish, sandy or rocky patches. The water's surface is textured with small ripples and reflects light, creating a shimmering effect. The overall scene is serene and natural.

Unterbrecht mich!

Über Migrationen





Über Migrationen

Software Migration



*“**Software migration** is the process of **moving** from the use of **one operating environment** to **another** operating environment that is, in most cases, [...] thought to be a better one.”*

- Wikipedia



Software Migration

*“**Software migration** is the process of **moving** from the use of **one operating environment** to **another** operating environment that is, in most cases, [...] thought to be a better one.”*

- Wikipedia

- z.B.
 - Software von **Windows** nach **Linux** portieren
 - Von einer **älteren** zu einer **neueren Framework Version** migrieren
 - Funktionalität von einem **Monolithen** in **Microservices** auslagern



Über Migrationen

Daten Migration



*“Data migration is the process of **selecting**, preparing, extracting, and **transforming data** and **permanently transferring** it from **one computer storage system** to **another**.”*

- Wikipedia



Data Migration

*“Data migration is the process of **selecting**, preparing, extracting, and **transforming data** and **permanently transferring** it from **one computer storage system** to **another**.”*

- Wikipedia

- z.B.
 - **Daten** von **einem Server** auf **einen anderen** verschieben
 - **Daten** von einem DB-Schema **transformieren** und in einem **neuen Schema** ablegen; danach das neue Schema verwenden
 - **Nachrichten** aus einem **IMAP Server** nehmen und sie in einer relationalen **Datenbank** speichern



Beispiele



Der Warenkorb



Beispiel 1 - Der Warenkorb

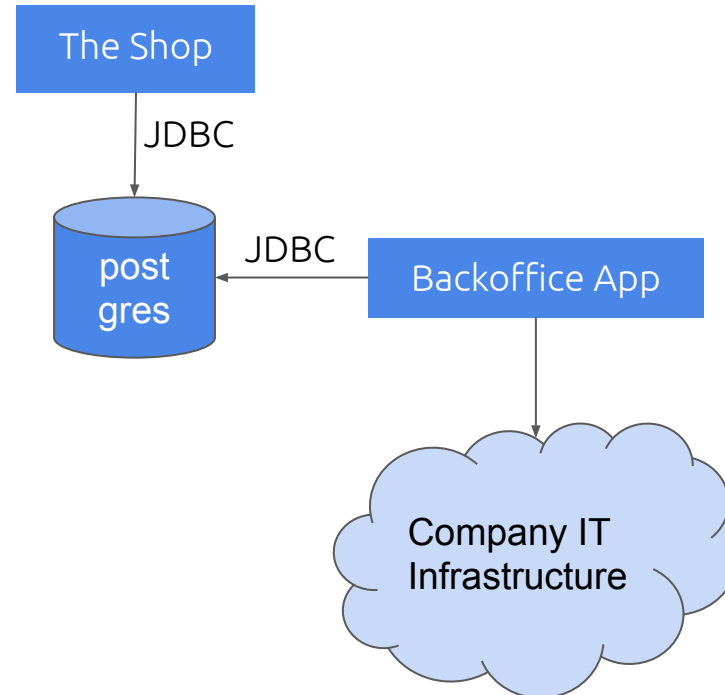
Der Warenkorb

- Großer Onlinehändler
- Produkt: **Onlineshop und Lieferservice**
 - ~ **30.000 Bestellungen / 2 Millionen €** Umsatz pro **Woche** (damals)
 - **Legacy System** von Agentur entwickelt mit **Java & Struts 2**
- Feature: **Der Warenkorb**



Beispiel 1 - Der Warenkorb

Legacy Architektur





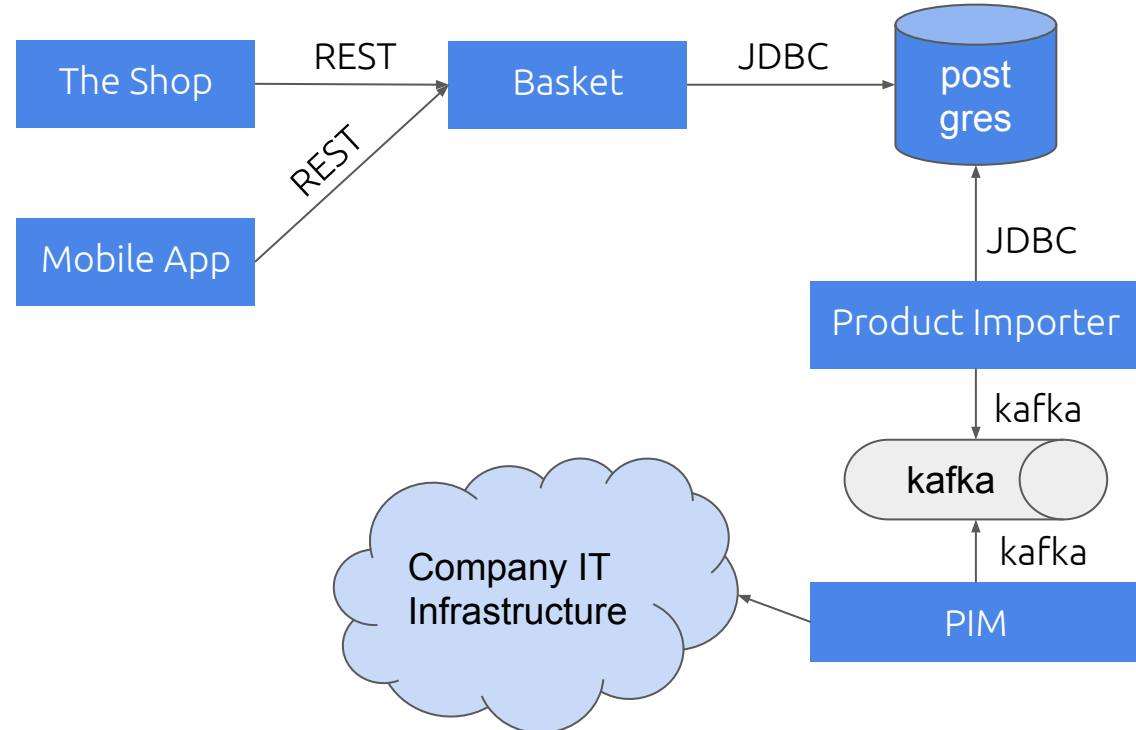
Probleme

- **schlechte Performance**
- **schlechte Wartbarkeit und Erweiterbarkeit**
 - versteckte Abhängigkeiten
 - Das ändern eines Teils zerstört potentiell einen anderen
- **hohe Fehleranfälligkeit**
 - viel zu viel in Session und Cookies gespeichert, Probleme bei Änderungen u.a. wegen Session-Replikation / Persistenz
- **nicht unabhängig entwickelbar** / deploybar / betreibbar
- **schlechte Testbarkeit** (hauptsächlich manuelle Tests, ein paar wenige Unit Tests und noch weniger automatisierte UI Tests)
- langwieriger & **komplexer Release Prozess**
- **API** für **andere Services** benötigt



Beispiel 1 - Der Warenkorb

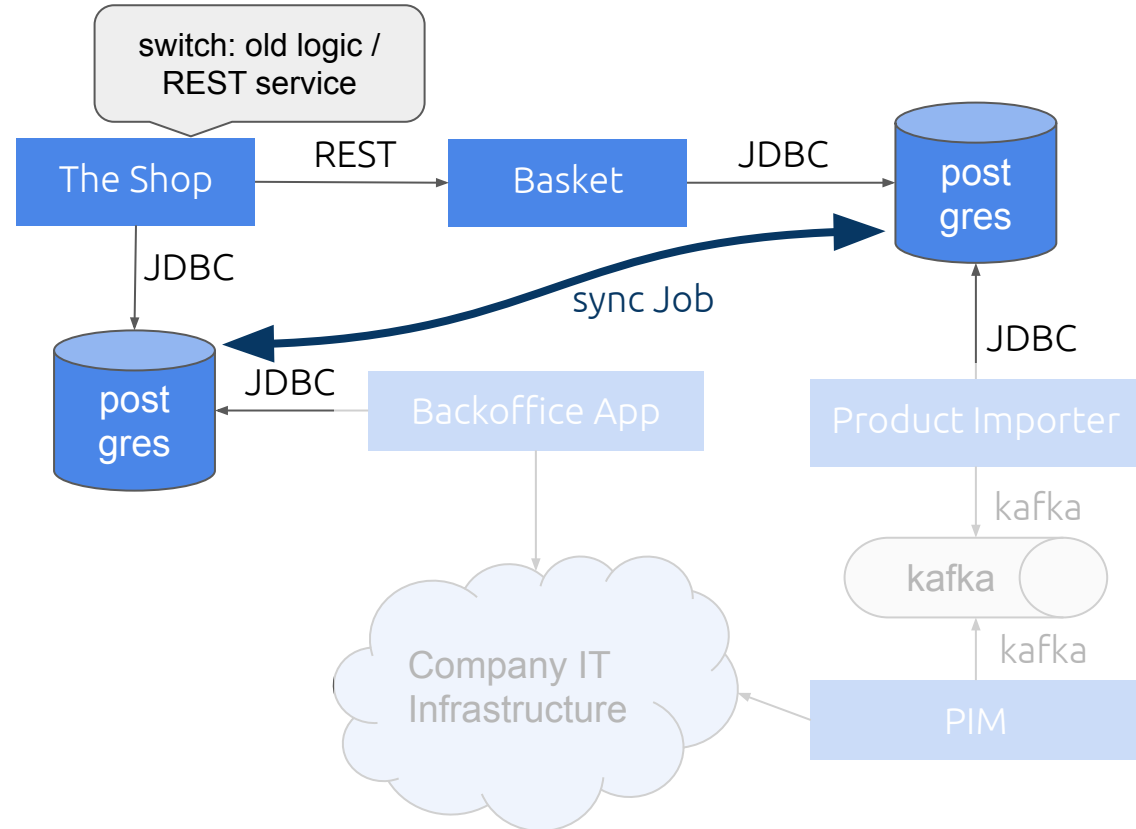
Zielarchitektur





Beispiel 1 - Der Warenkorb

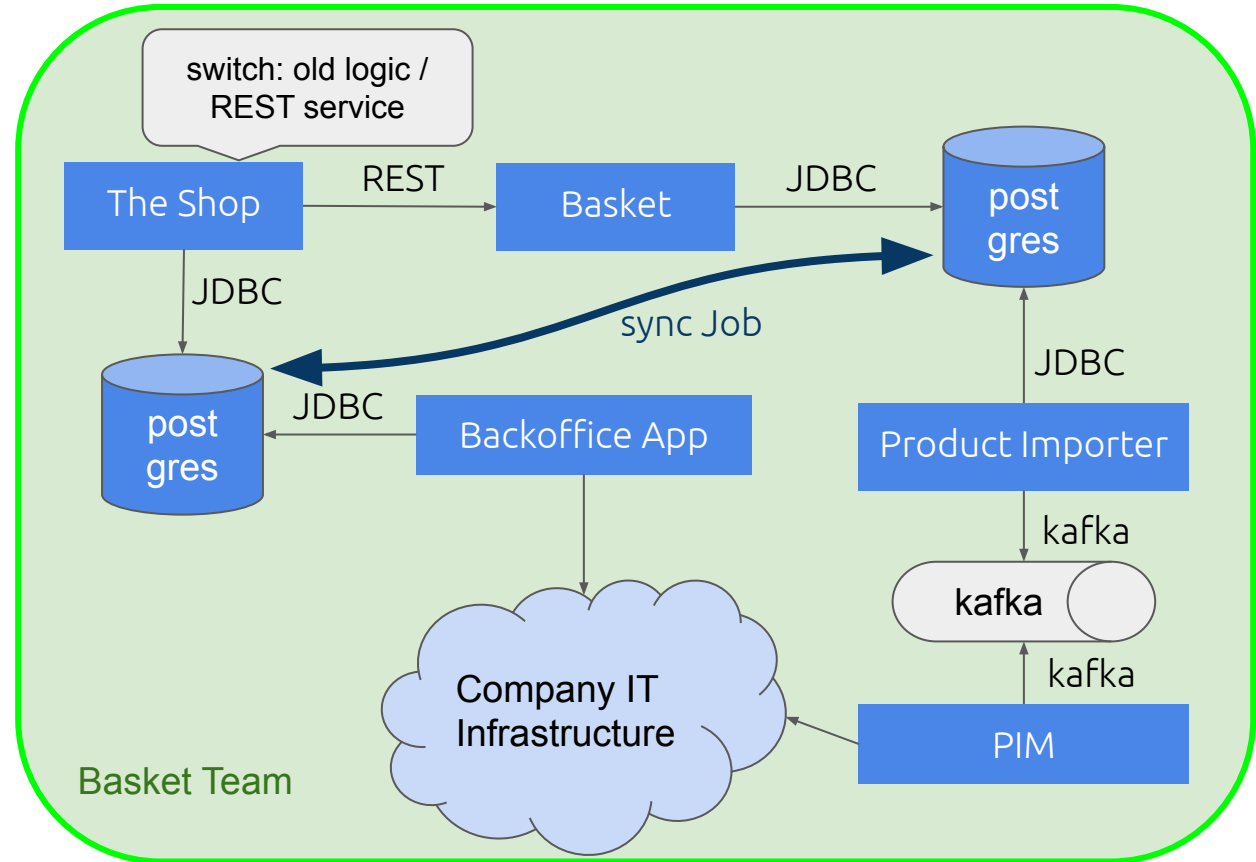
Migration (initialer Plan)





Beispiel 1 - Der Warenkorb

Migration (initialer Plan)



PM

Schneller!



A fox with orange and white fur is shown in profile, facing left, with its mouth wide open as if barking or howling. It is standing in a field of tall green grass and some small white flowers. The background is a soft-focus green field.

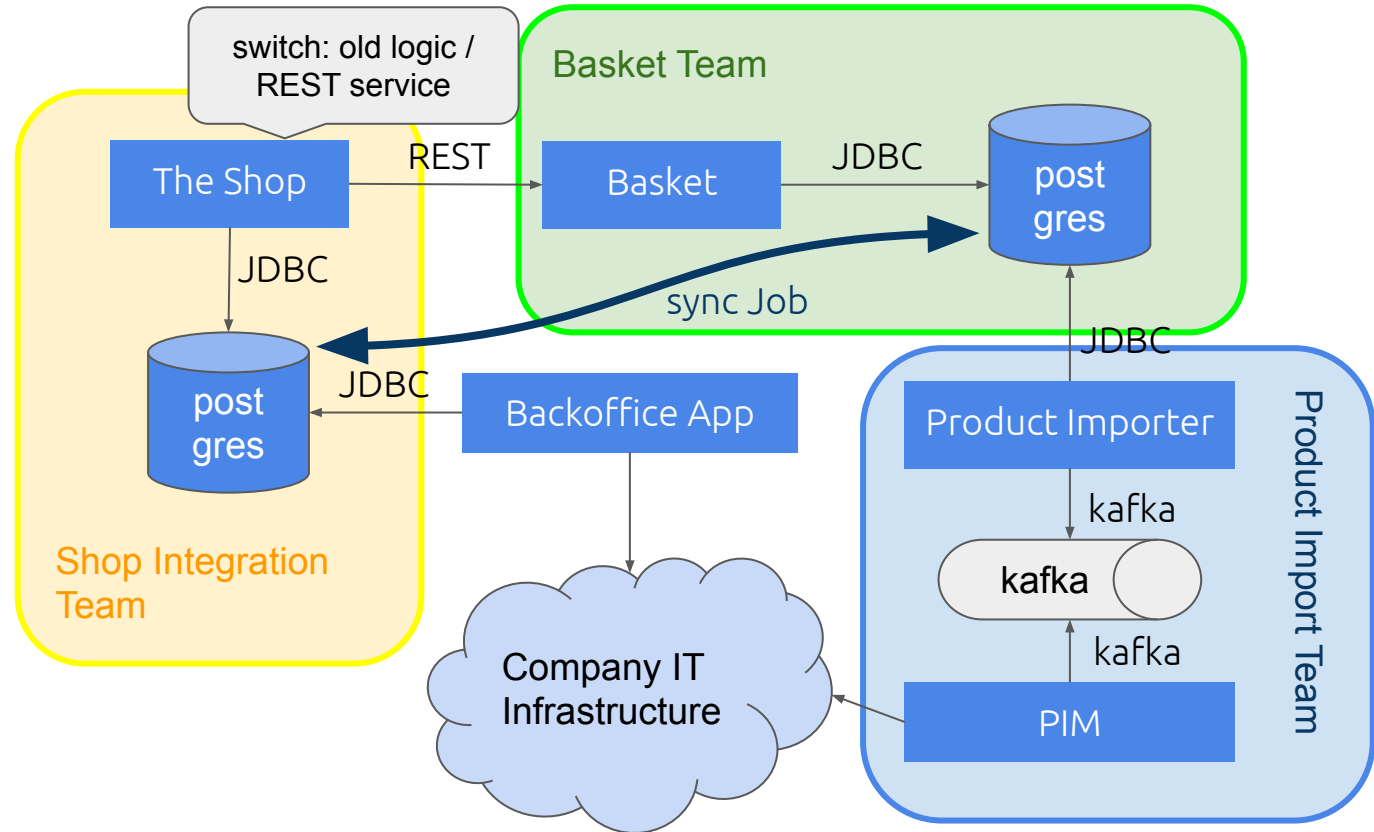
*Lasst uns
mehr Team
drauf werfen!*

schlauer Typ



Beispiel 1 - Der Warenkorb

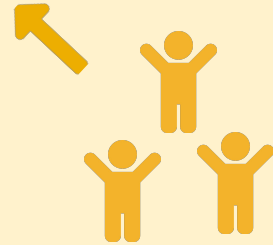
Migration (Neuer Plan ...)





Beispiel 1 - Der Warenkorb

Teams und Ziele

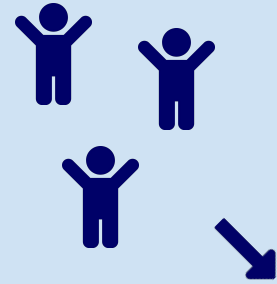


Shop Integration
Team

Basket Team



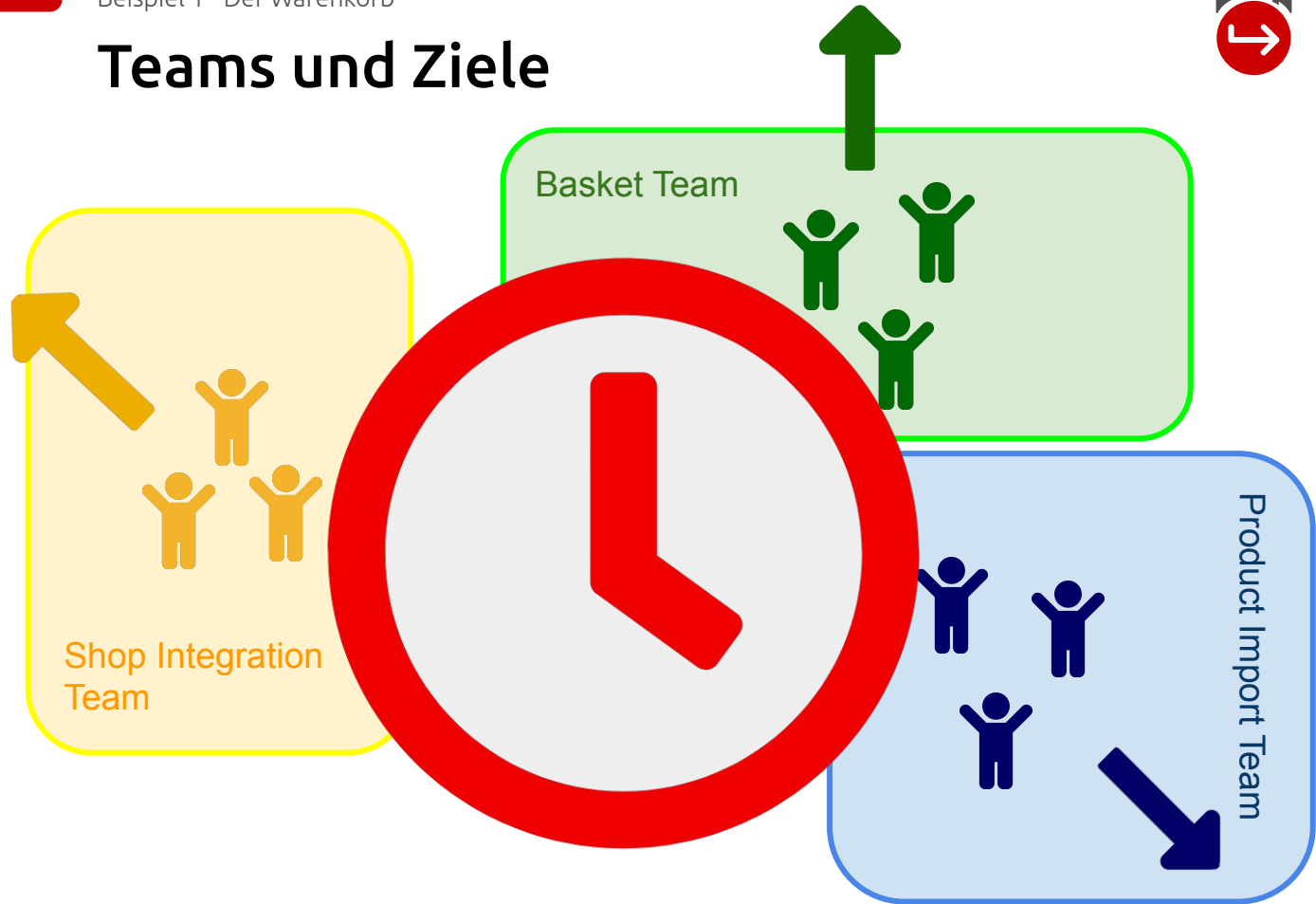
Product Import Team





Beispiel 1 - Der Warenkorb

Teams und Ziele



A black and white photograph of the band Pink Floyd performing on stage. The band members are silhouetted against a large, textured wall. Large, dark shadows of the band members are cast onto the wall behind them, creating a dramatic visual effect. A speech bubble is overlaid on the image, containing the text "We don't need no synchronization!".

*We don't need no
synchronization!*

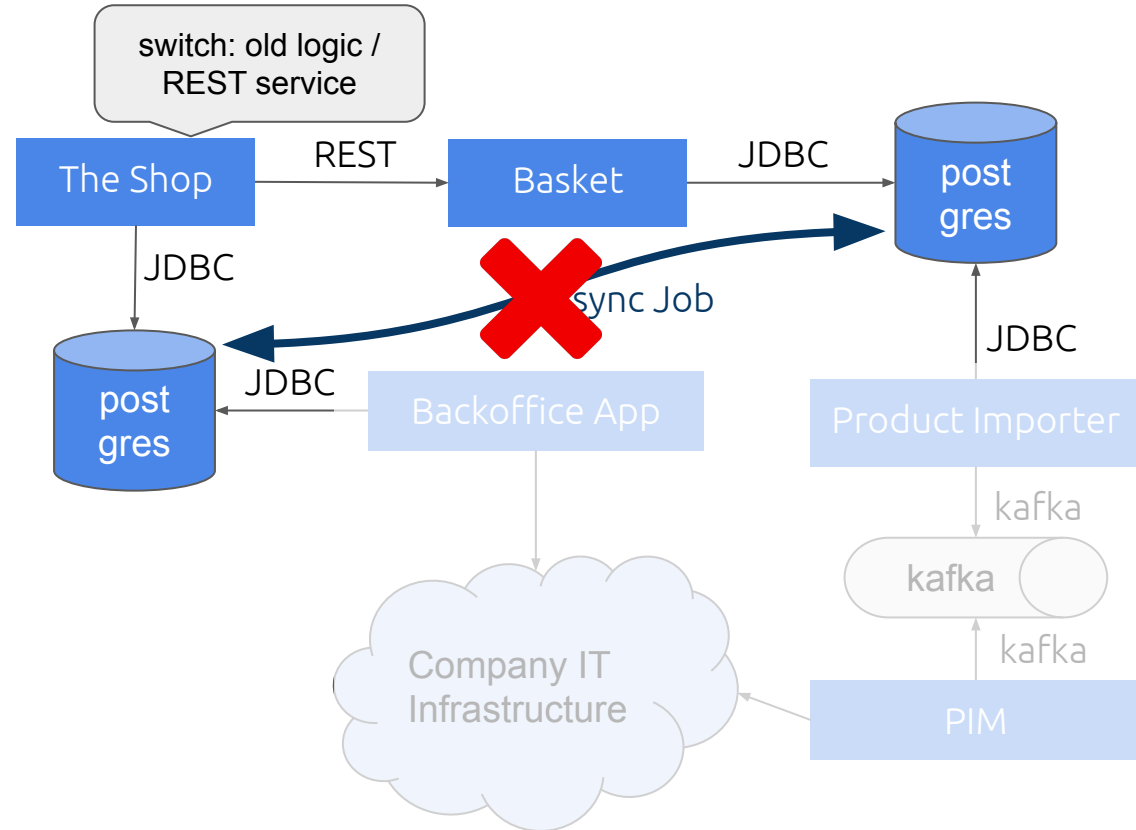
PO

© 2000 Redfern



Beispiel 1 - Der Warenkorb

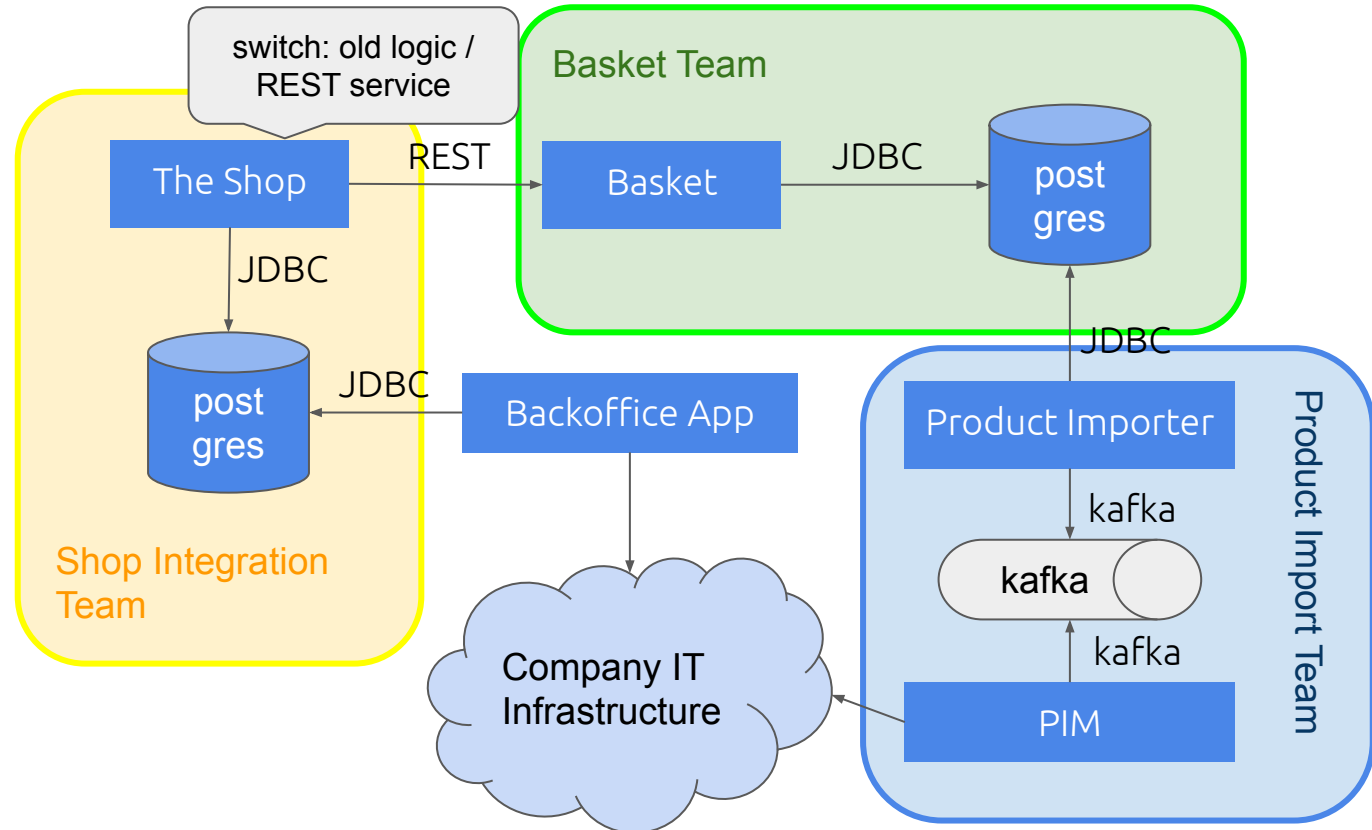
Migration (Neuer Plan 2 ...)





Beispiel 1 - Der Warenkorb

Migration (Neuer Plan 2 ...)





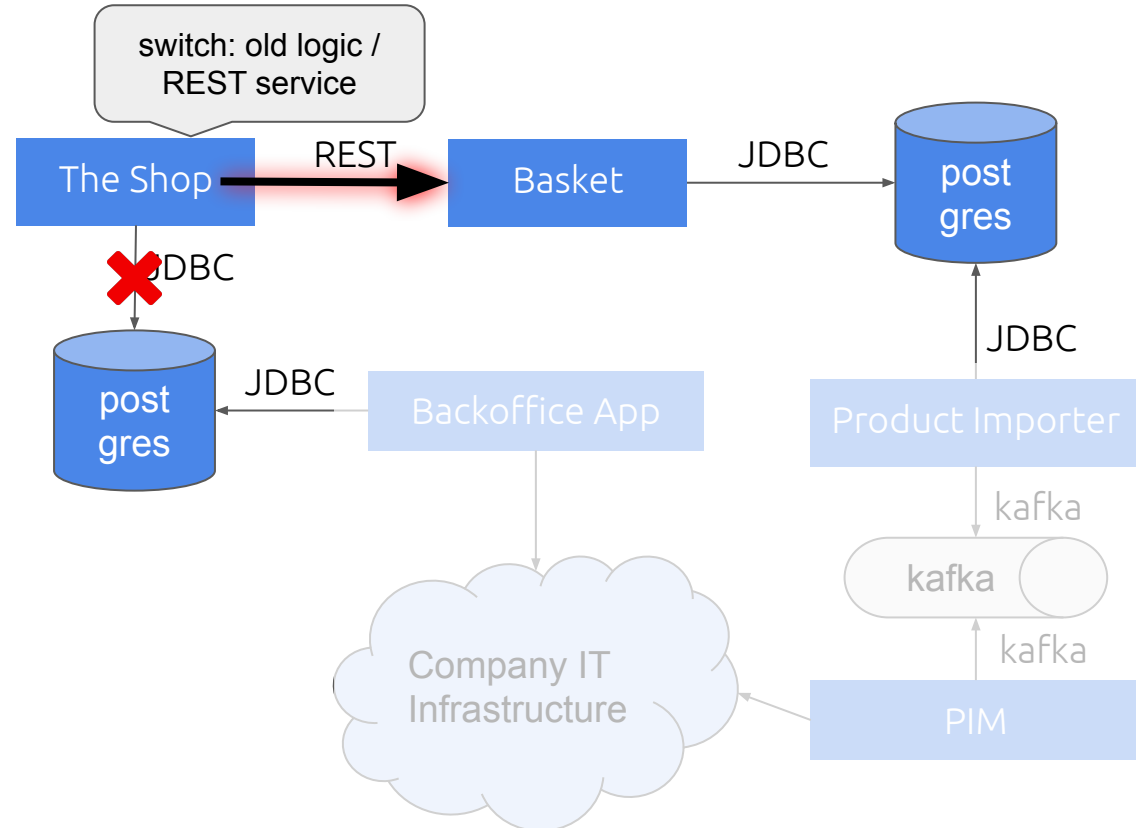
Integration on PRE

- natürlich haben wir das ganze irgendwann auch integriert **ausprobiert**
 - inkl. Load Tests
- Integrationsumgebung: **PRE**
- **PRE:**
 - **< 100.000 Warenkörbe** ✓
 - **~ 10 (dummy) Produkteinträge** ✓
- **PROD:**
 - **> 1.000.000 Warenkörbe** ?
 - **> 4.000.000 Produkteinträge** ?
- unglücklicherweise: **JOIN** über **Warenkörbe** und **Produkteinträge** ...



Beispiel 1 - Der Warenkorb

Neuen Service aktivieren in PROD



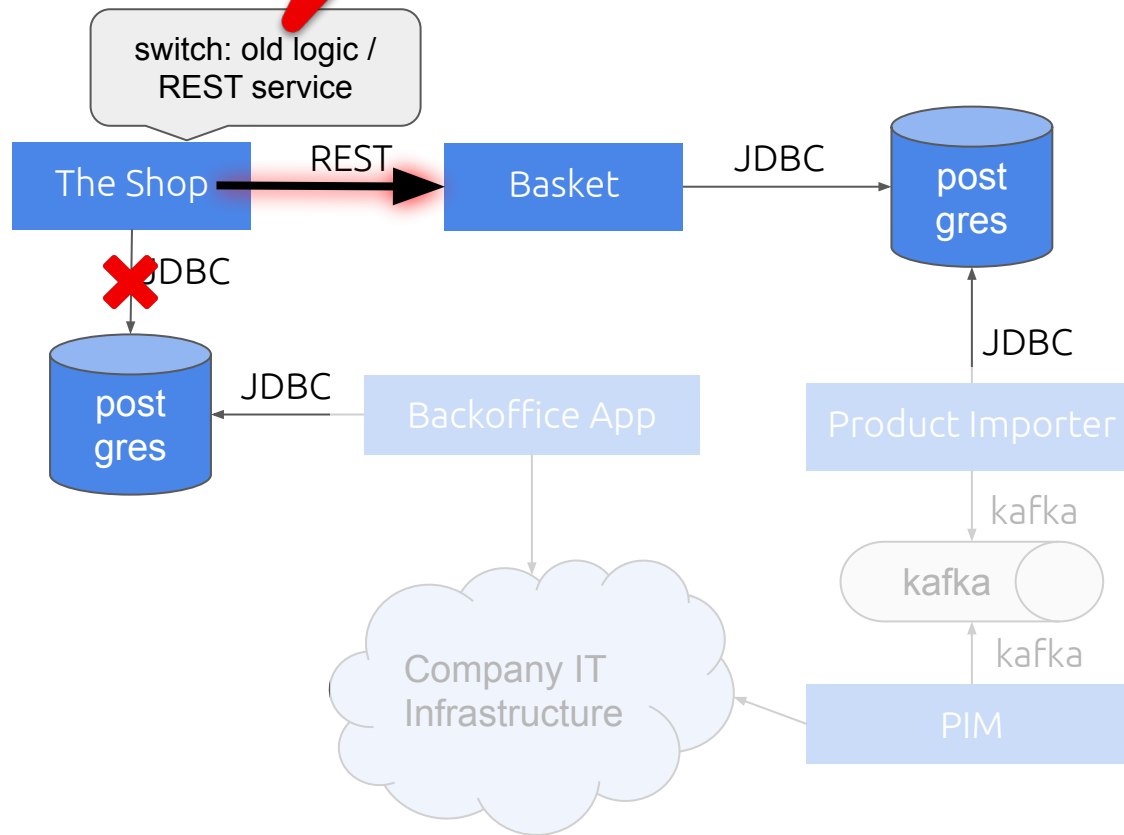


Beispiel 1 - Der Warenkorb

Neuen Service aktivieren in PROD



Alle Warenkörbe leer!





Beispiel 1 - Der Warenkorb

Imaginationsübung

- Du kommst von der **Arbeit** nach Hause, **müde** und **gestresst**
- Du musst noch **einkaufen**
- Großartige Idee: **Alles online bestellen!**
- Du freust Dich Zeit zu sparen und brauchst am Computer etwa 30 Minuten um alles zusammenzusuchen
- Plötzlich ist **Dein Warenkorb leer** und Du musst neu anfangen



Beispiel 1 - Der Warenkorb

Imaginationsübung

- Du kommst von der **Arbeit** nach Hause, **müde** und **gestresst**
- Du musst noch **einkaufen**
- Großartige Idee: **Alles online bestellen!**
- Du freust Dich Zeit zu sparen und brauchst am Computer etwa 30 Minuten um alles zusammenzusuchen
- Plötzlich ist **Dein Warenkorb leer** und Du musst neu anfangen
- **Wie fühlst Du Dich :-)** ?



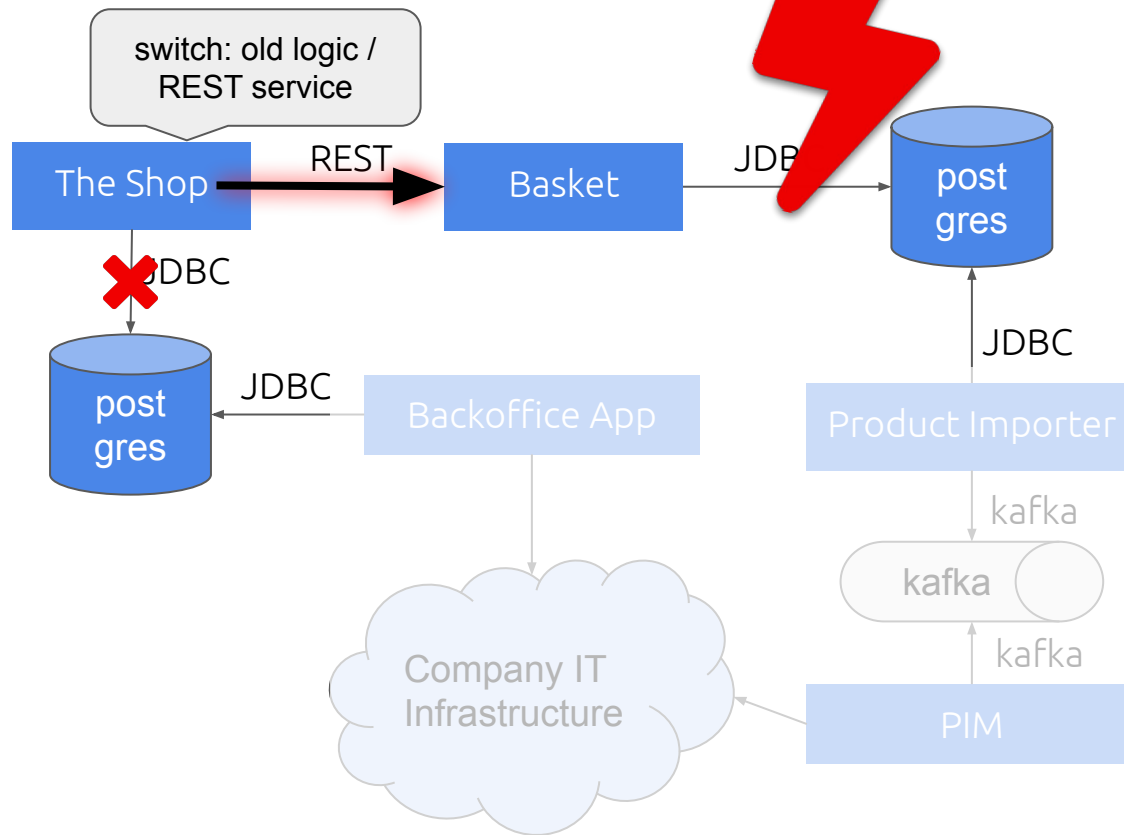
Kunde



Datenbank überlastet, Shop down!



Switch to new Service





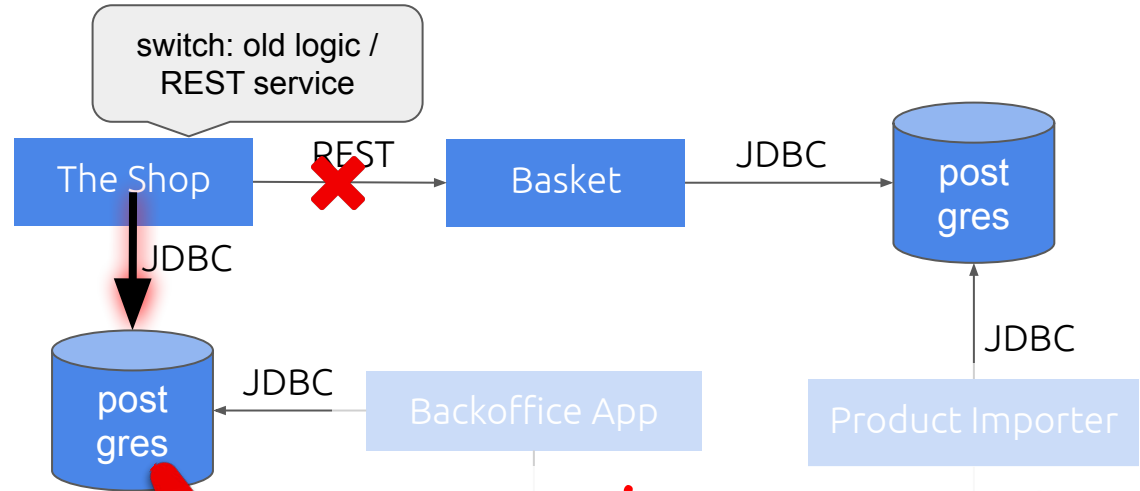
Kunde





Beispiel 1 - Der Warenkorb

Switch back

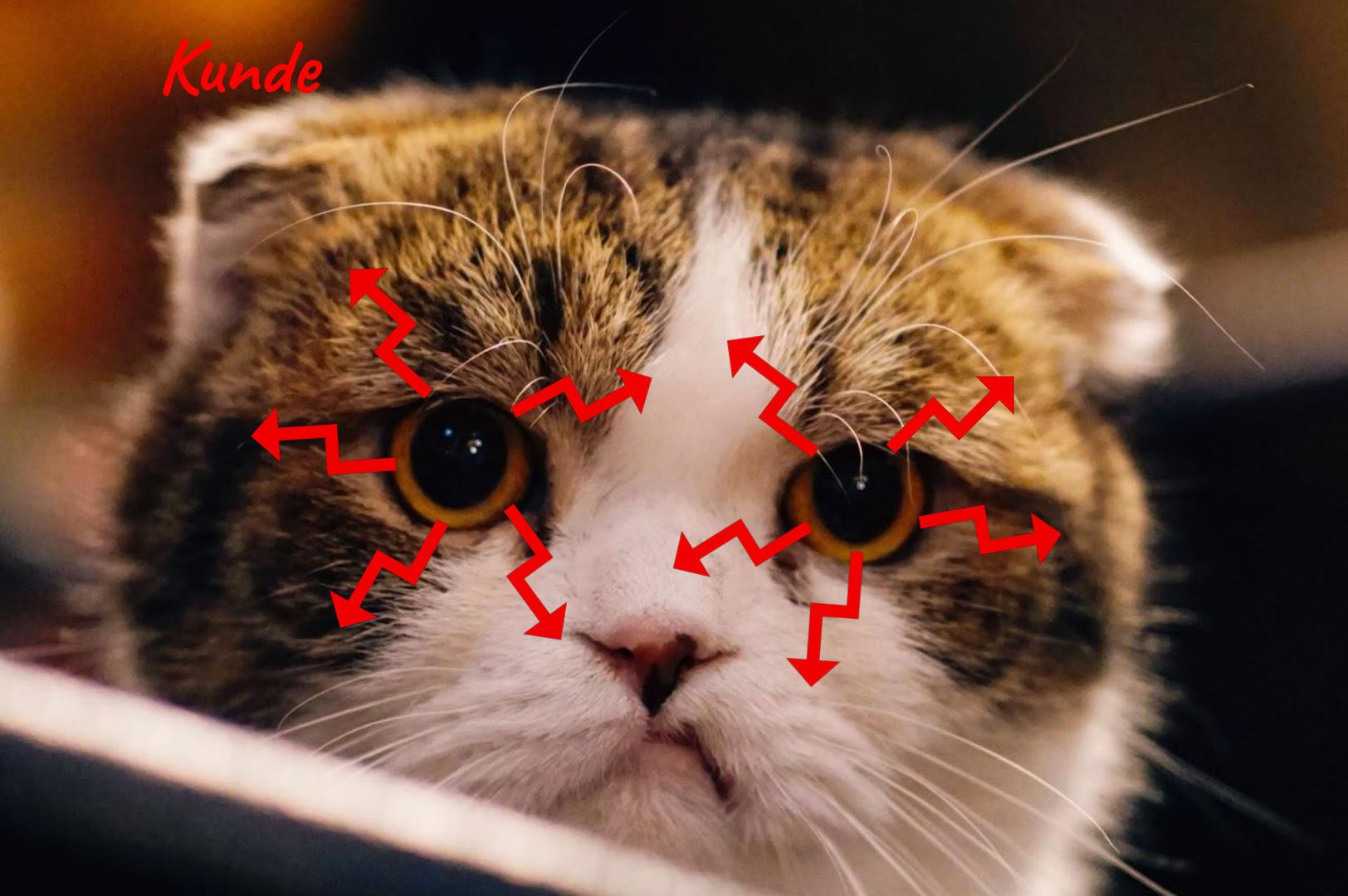


*Neue Warenkörbe
verloren, alte tauchen
wieder auf*

Company
Infrastructure



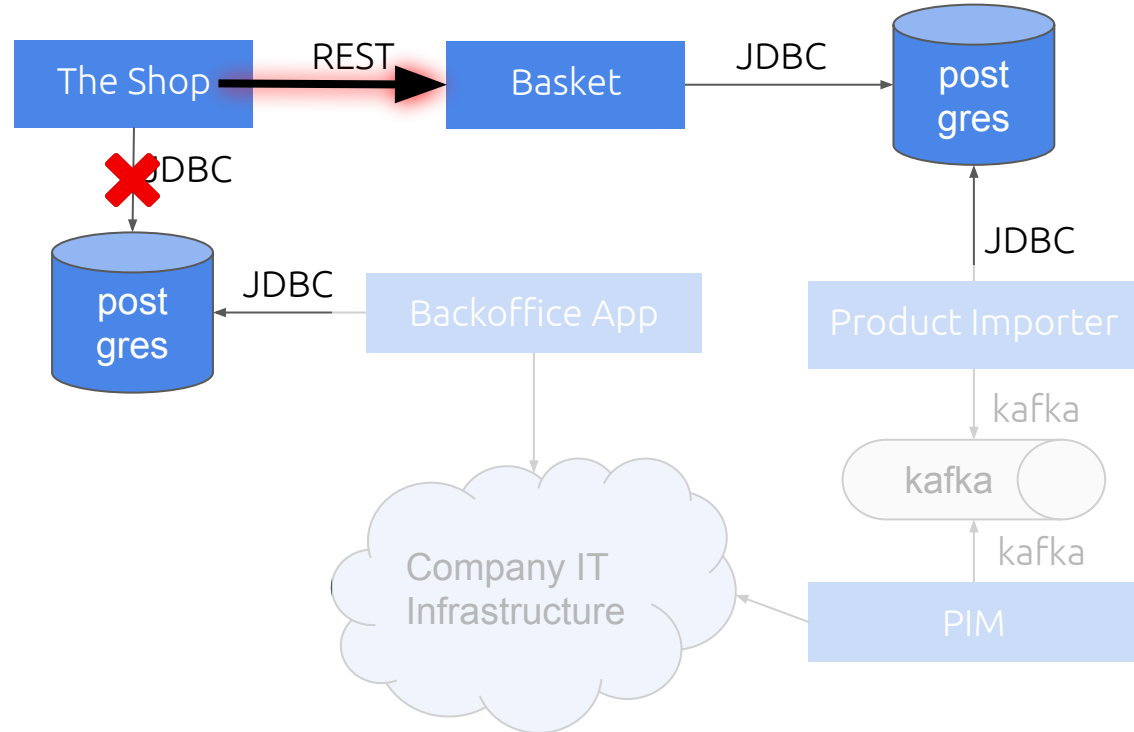
Kunde





Beispiel 1 - Der Warenkorb

Problem fixen & Umschalten (mehrere Iterationen)

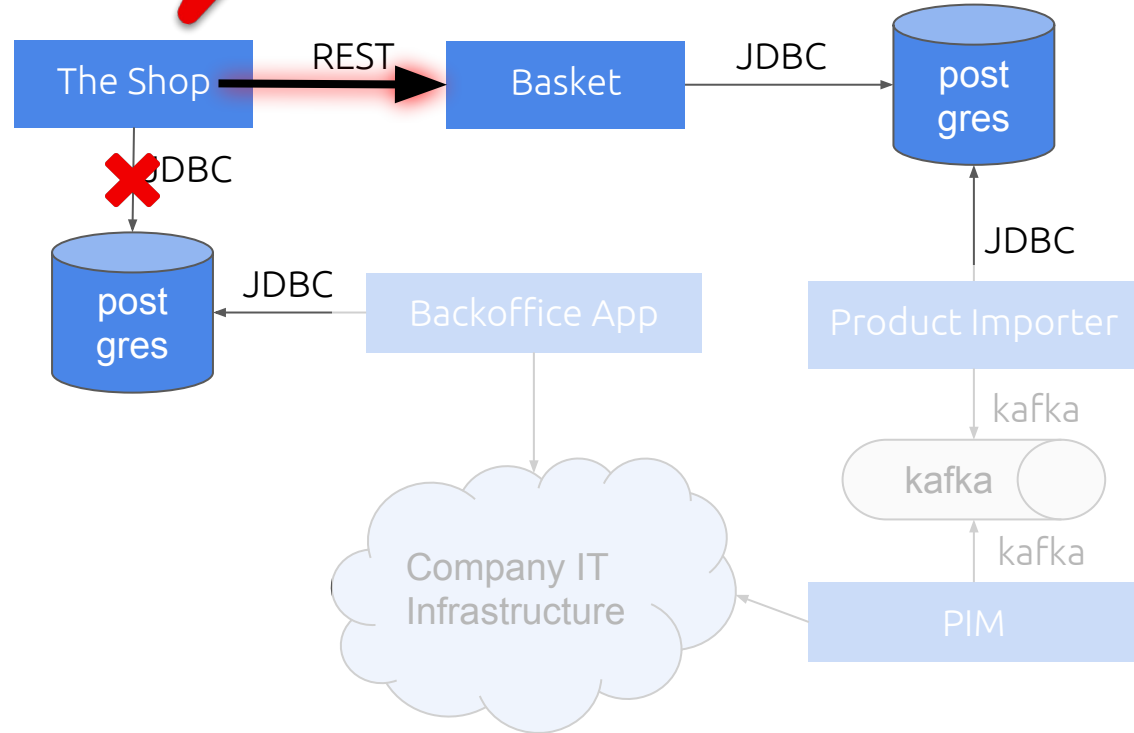




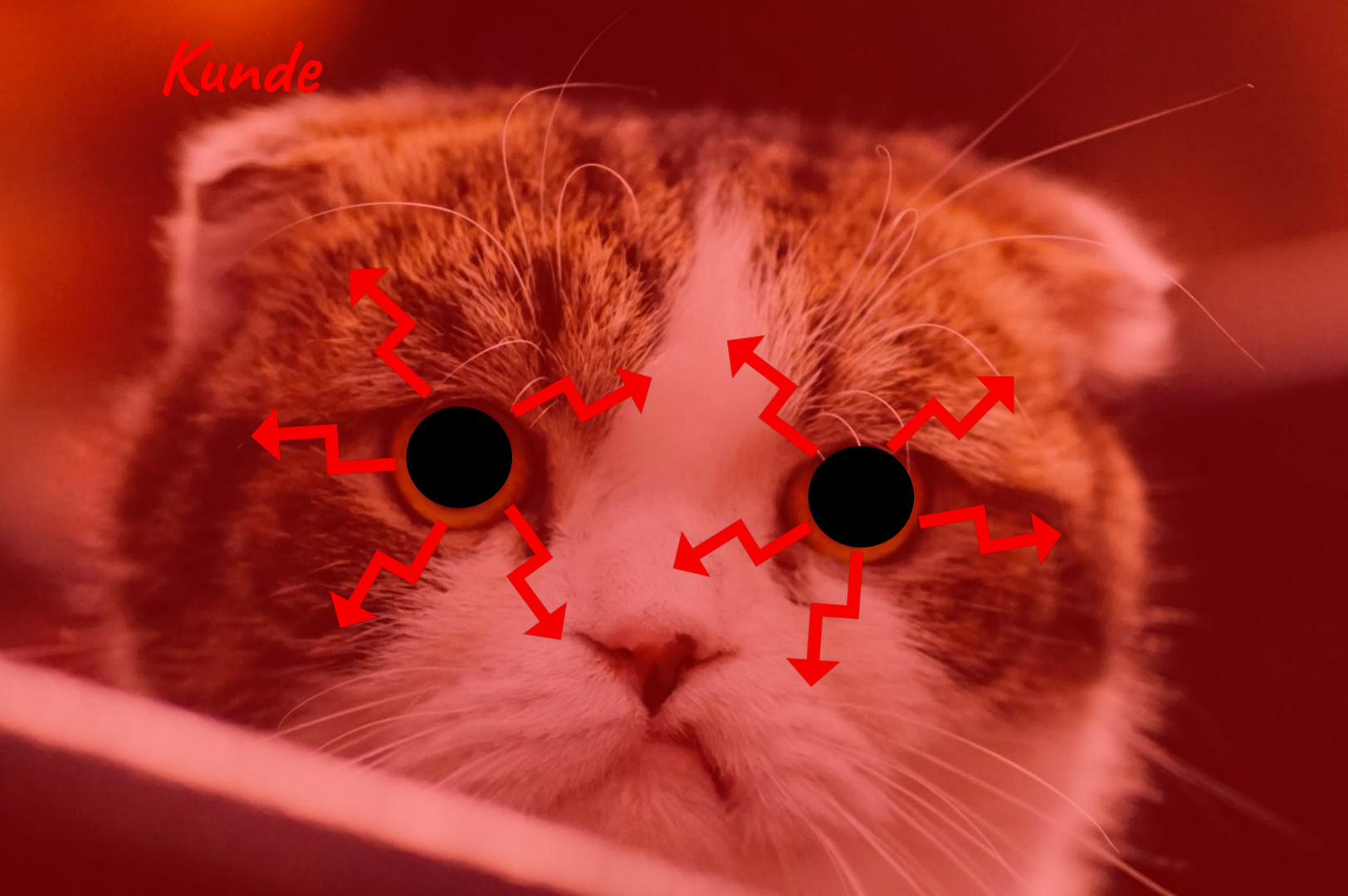
Beispiel 1 - Der Warenkorb

Totales Warenkorb-Chaos!

Problem fixen & Umschalten (mehrere Iterationen)



Kunde



Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

Die Einkaufsliste

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Beispiel 2 - Die Einkaufsliste

Die Einkaufsliste

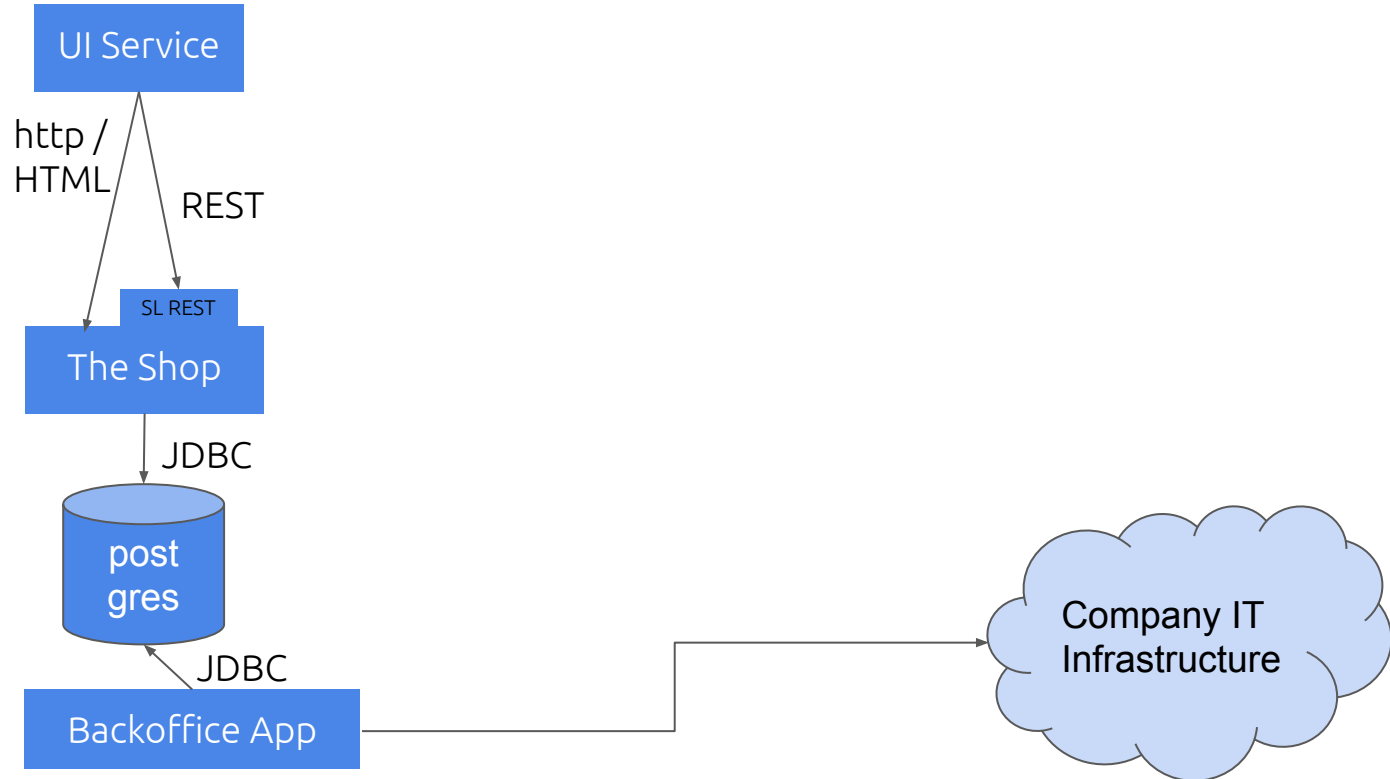
- Großer **Onlinehändler**
- Produkt: **Onlineshop und Lieferservice**
 - ~ **30.000 Bestellungen / 2 million €** Umsatz pro **Woche**
(damals)
 - **Legacy System** von Agentur entwickelt mit **Java & Struts 2**
- Feature: **Die Einkaufsliste**

Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Legacy Architektur

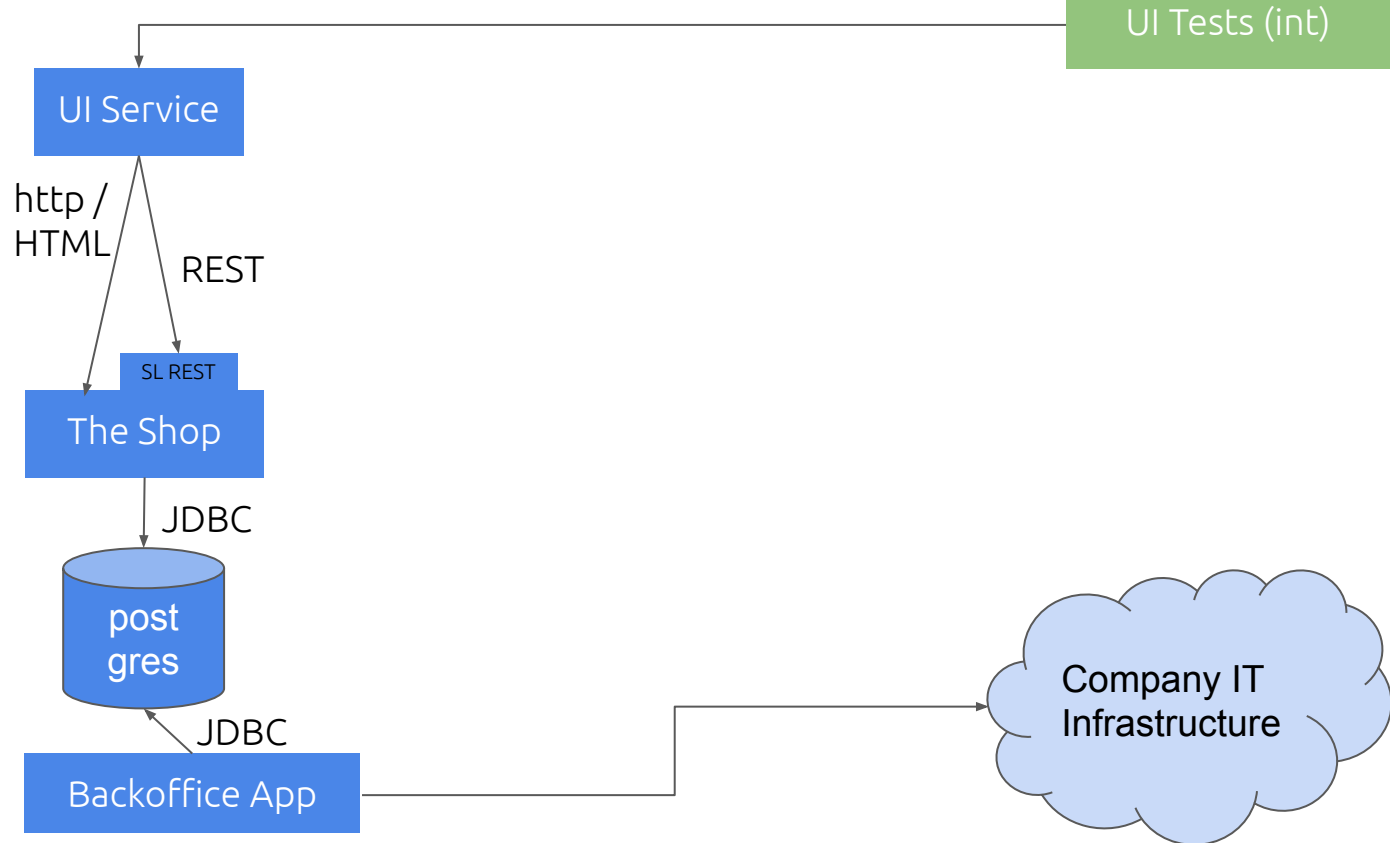


Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Legacy Architektur



Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Probleme

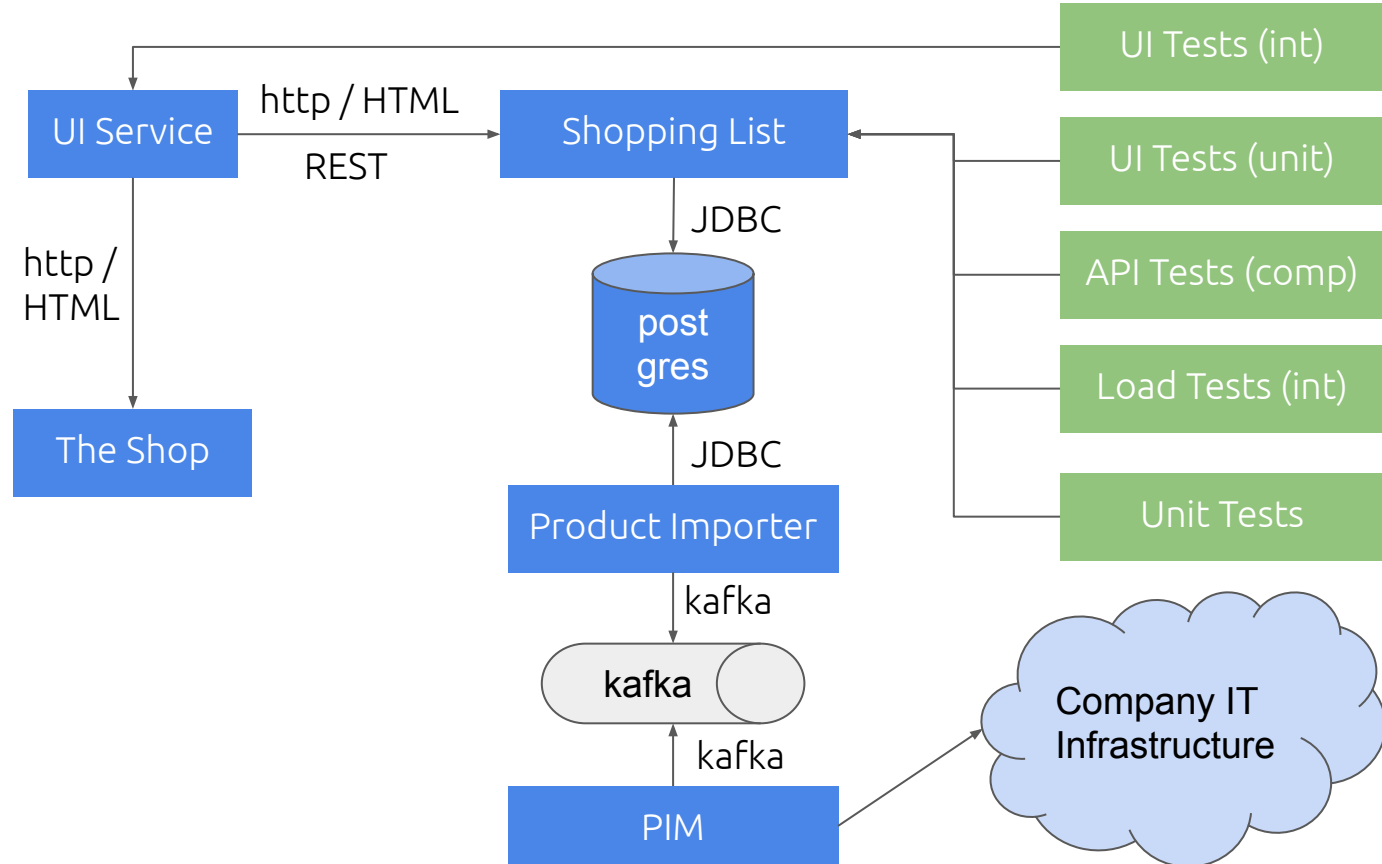
Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

- **nicht unabhängig entwickelbar** / deploybar / betreibbar
- langwieriger & **komplexer Release Prozess**
- **schlechte Testbarkeit**
- **langsame und unzuverlässige UI-Tests**



Beispiel 2 - Die Einkaufsliste

Zielarchitektur



Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

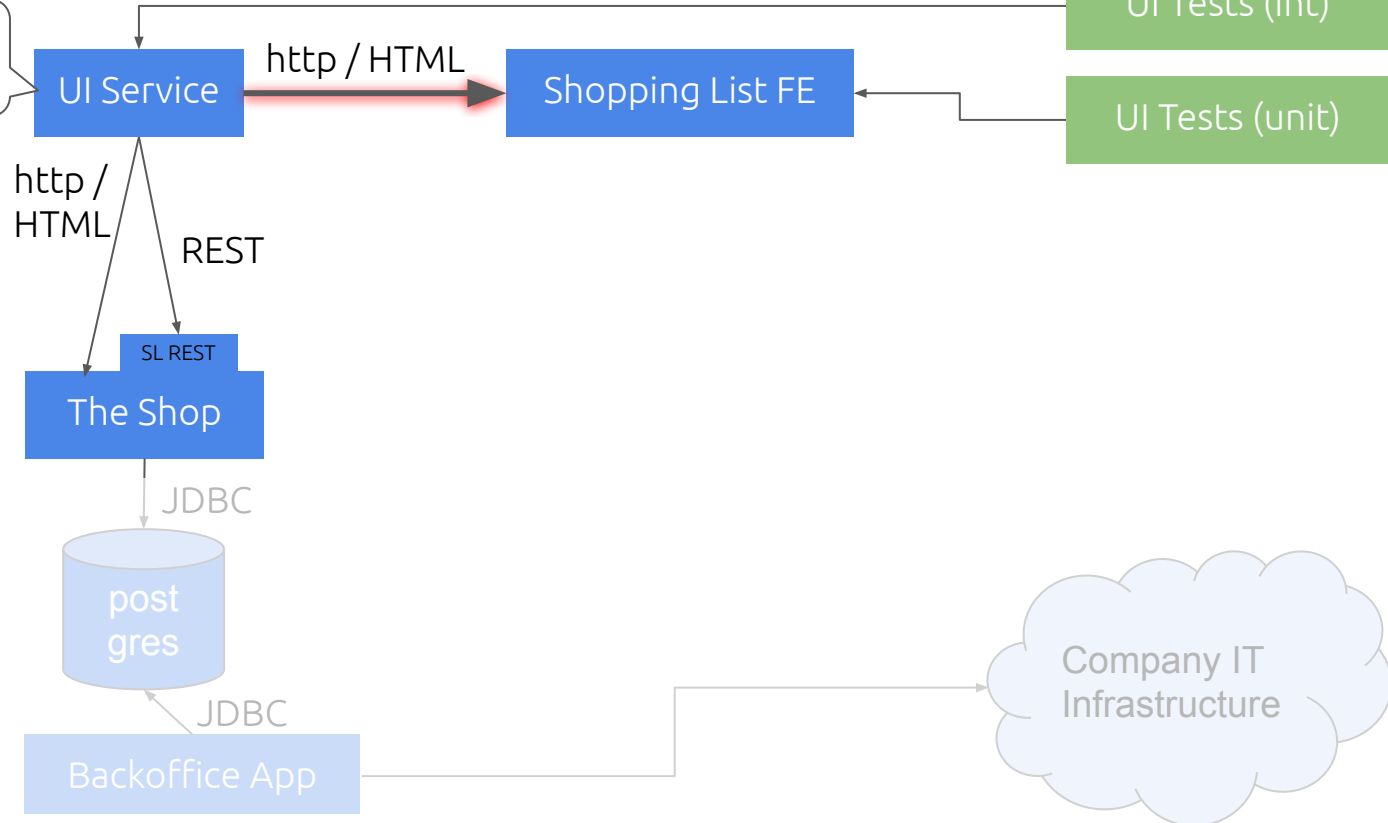
214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Beispiel 2 - Die Einkaufsliste

Migrationsschritt 1: Frontend extr.

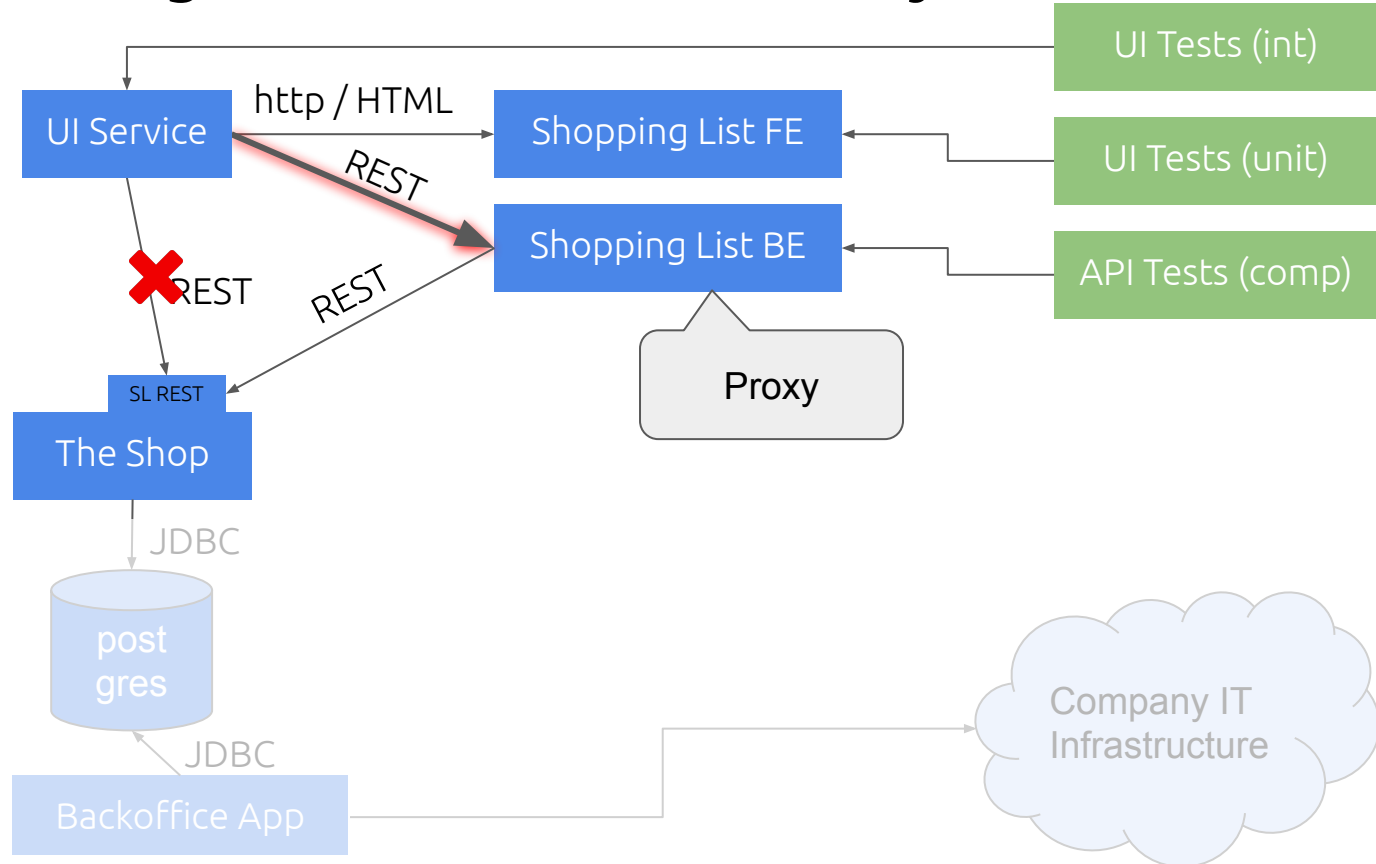
switch: old / new
frontend





Beispiel 2 - Die Einkaufsliste

Migrationsschritt 2: SL Proxy



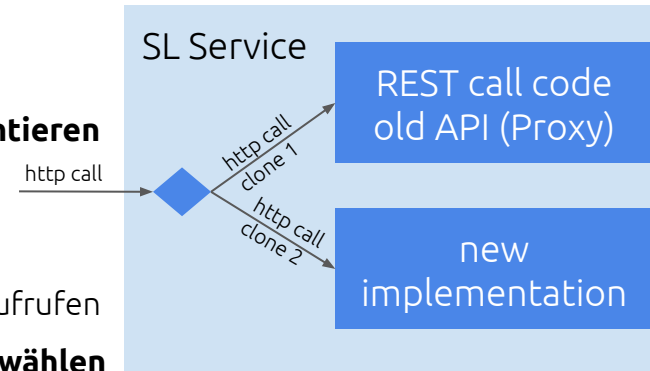
Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Der Proxy-Plan

- den **Service** so bald wie möglich in **PROD** laufen lassen für **schnelles Feedback**
- mit **REST proxy** starten
- Features **Schritt für Schritt implementieren**
- **Jeden HTTP-Call klonen**
 - alte **REST API** async. aufrufen
 - **Neue Implementation** async. aufrufen
 - Feature Toggle um **Antwort zu wählen**
- Vorteil: **beide Systeme bleiben synchron**



Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



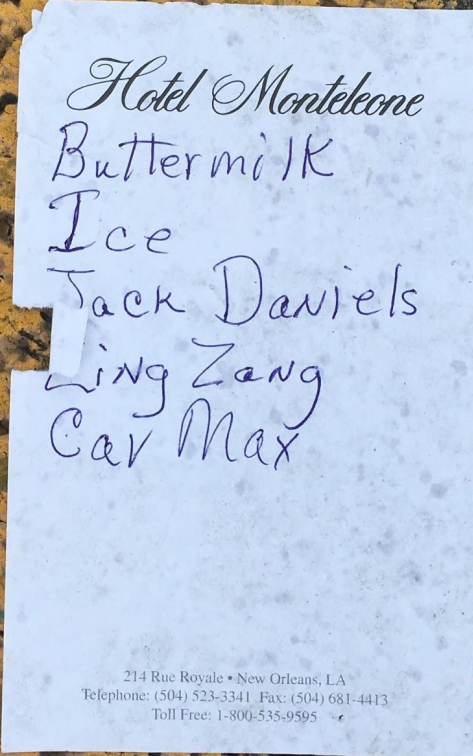
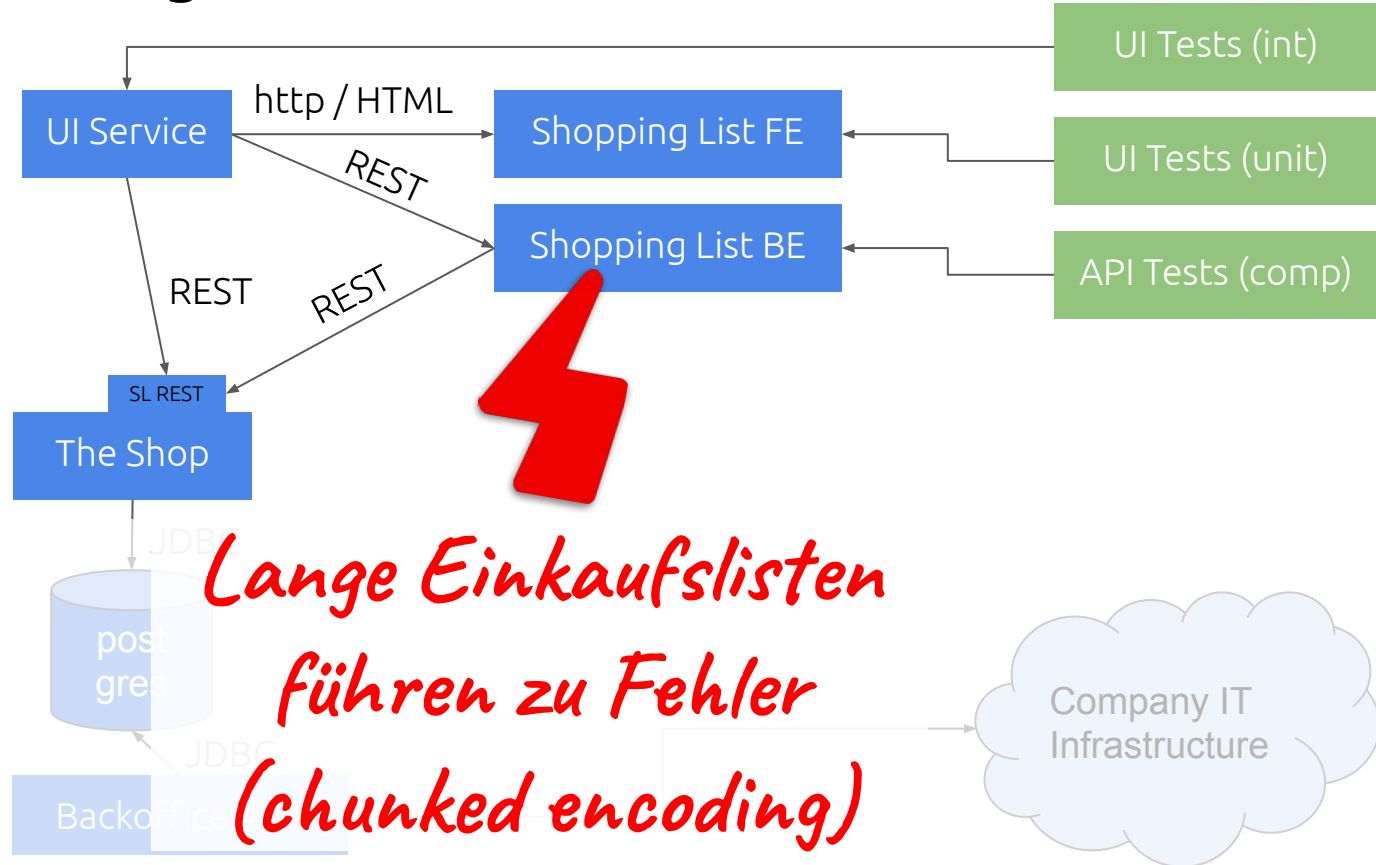
Feature Toggles

- AKA **Feature Flags**
- Erlauben es (ggf. zur Laufzeit) festzulegen ob ein **Feature aktiv** sein soll
 - bzw. **welche Implementierung** eines Features verwendet wird
 - ggf. in Abhängigkeit von **User, Standort, IP-Adresse ...**
- **Feature in Produktion** ausrollen, es aber nur **bestimmten Benutzern** zeigen



Beispiel 2 - Die Einkaufsliste

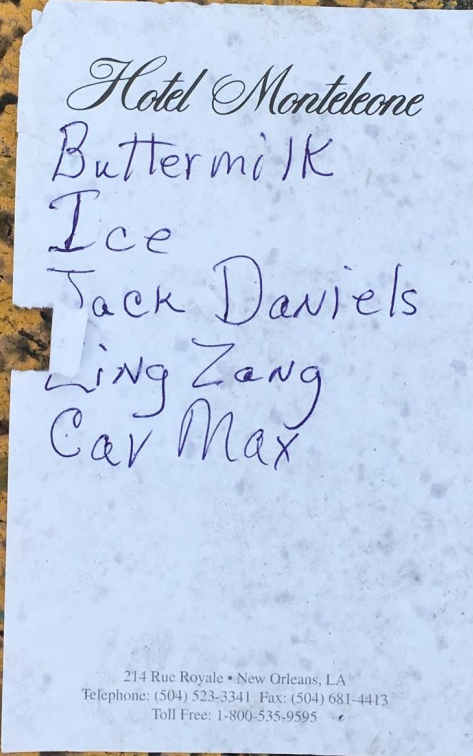
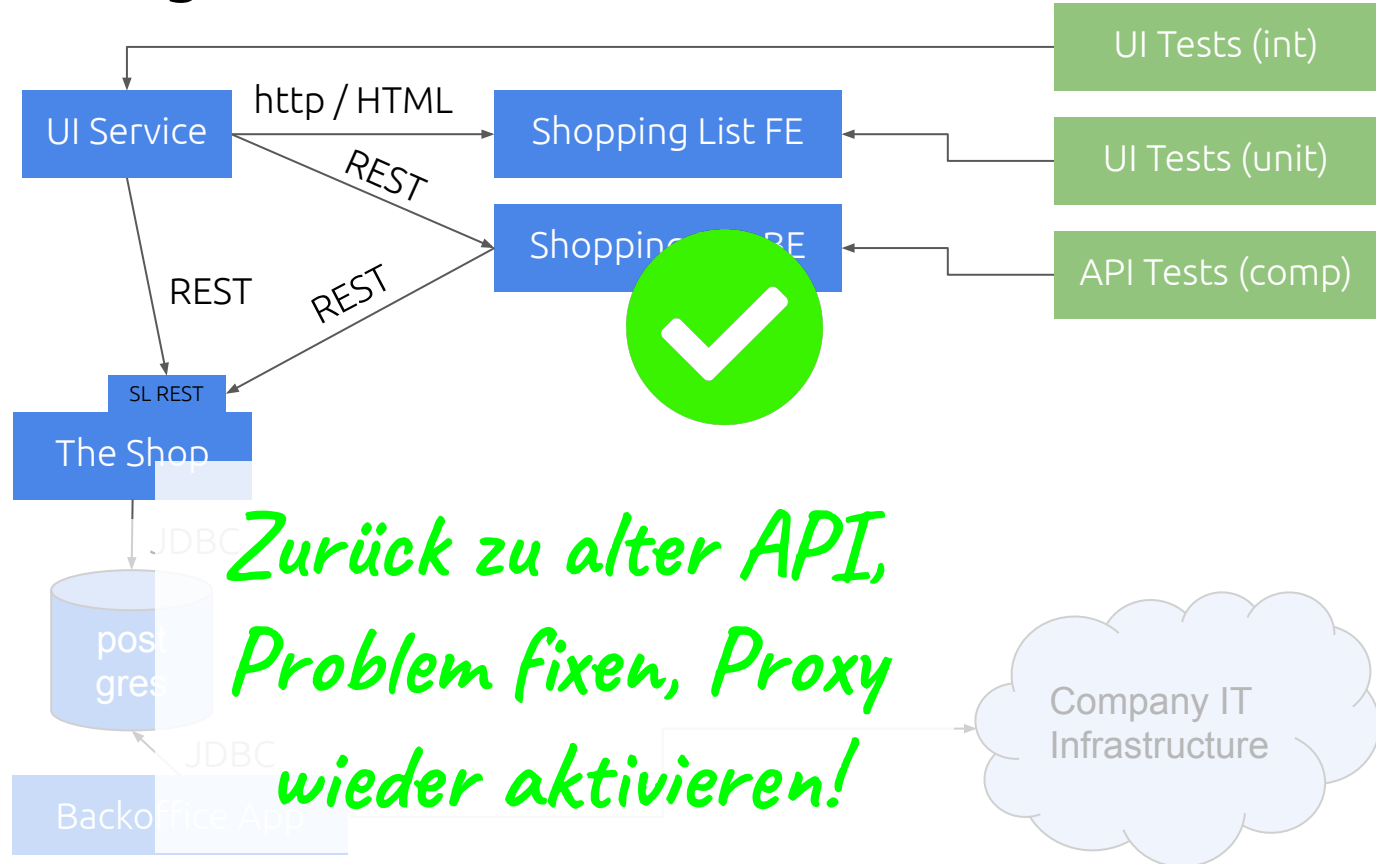
Migrationsschritt 2: Problem





Beispiel 2 - Die Einkaufsliste

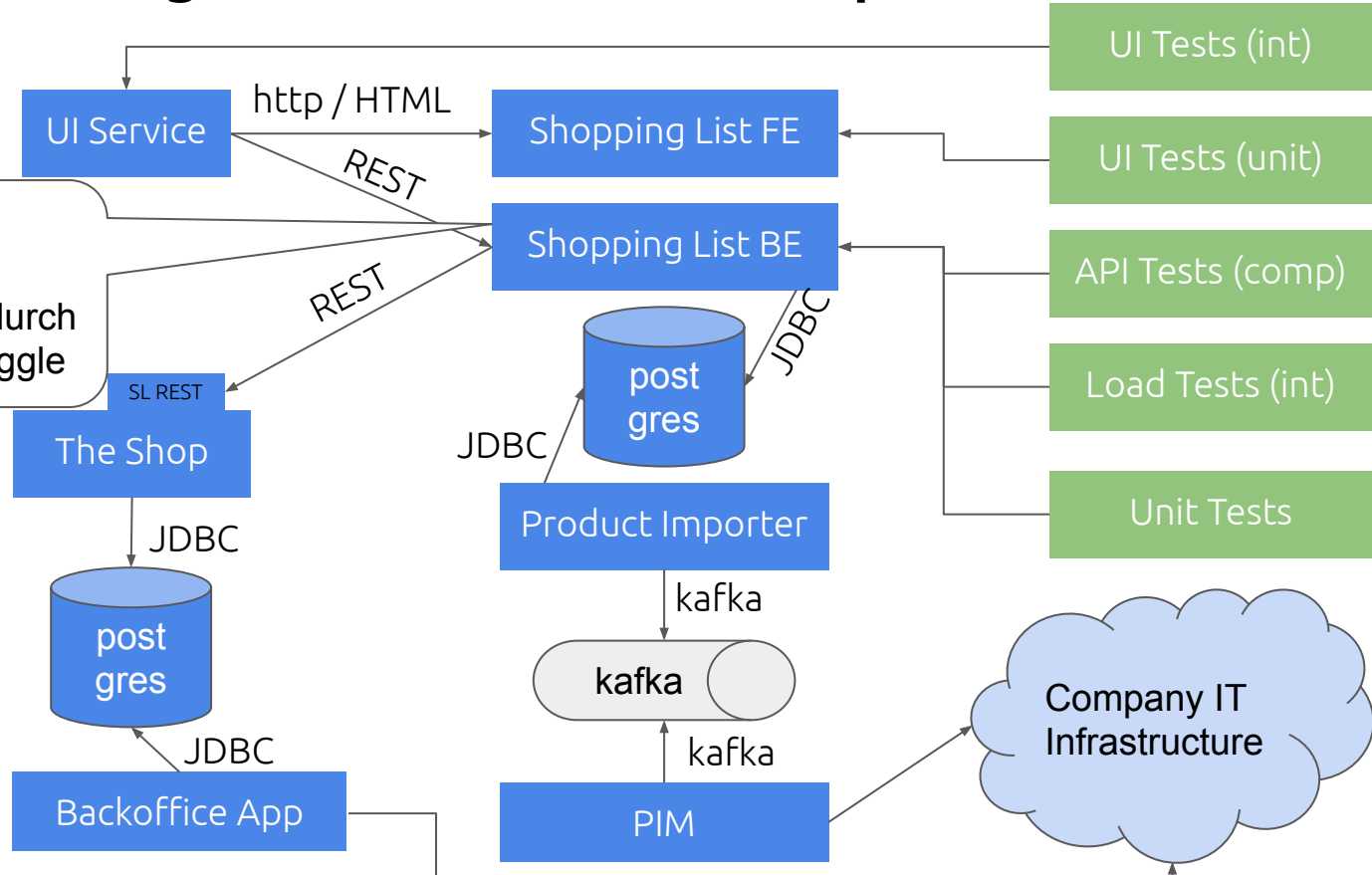
Migrationsschritt 2: Problem





Migrationsschritt 3: BE implementieren

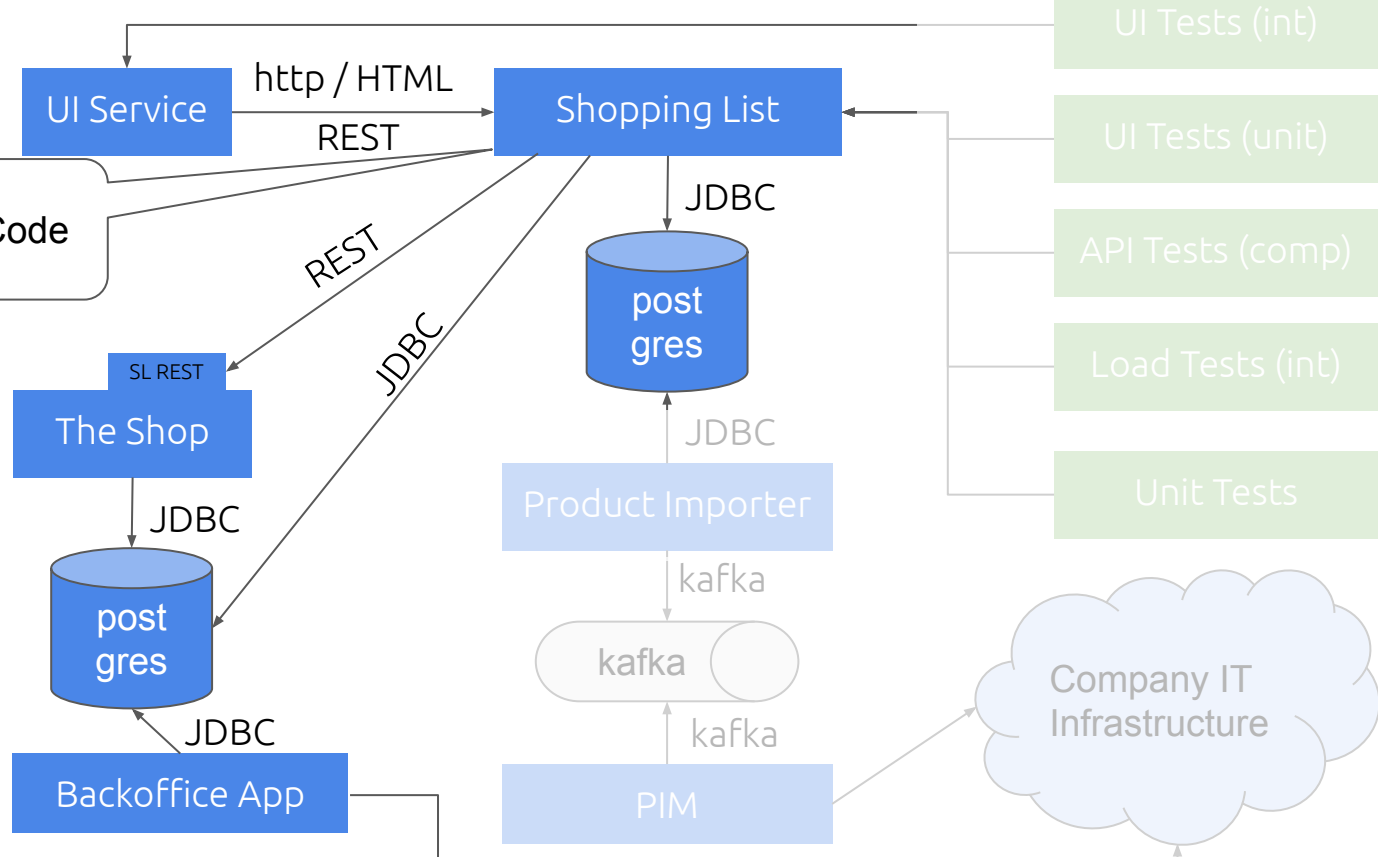
- Schritt für Schritt!
- gesteuert durch Feature Toggle





Beispiel 2 - Die Einkaufsliste

Migrationsschritt 4: Mergen & Migrieren!



Hotel Monteleone
Buttermilk

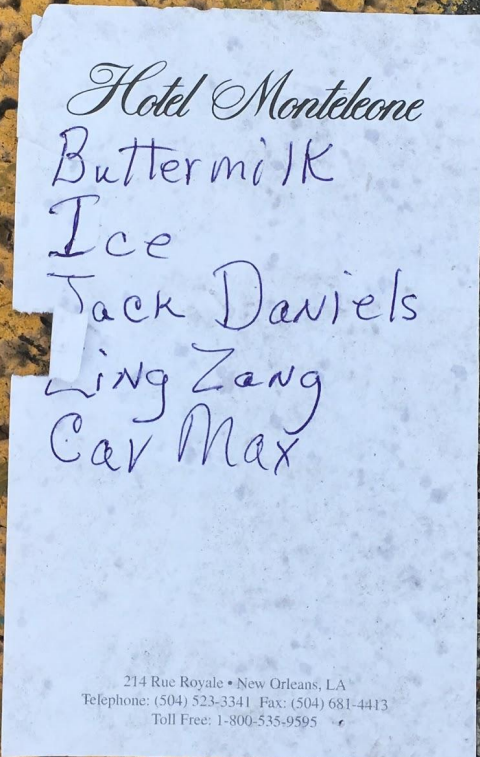
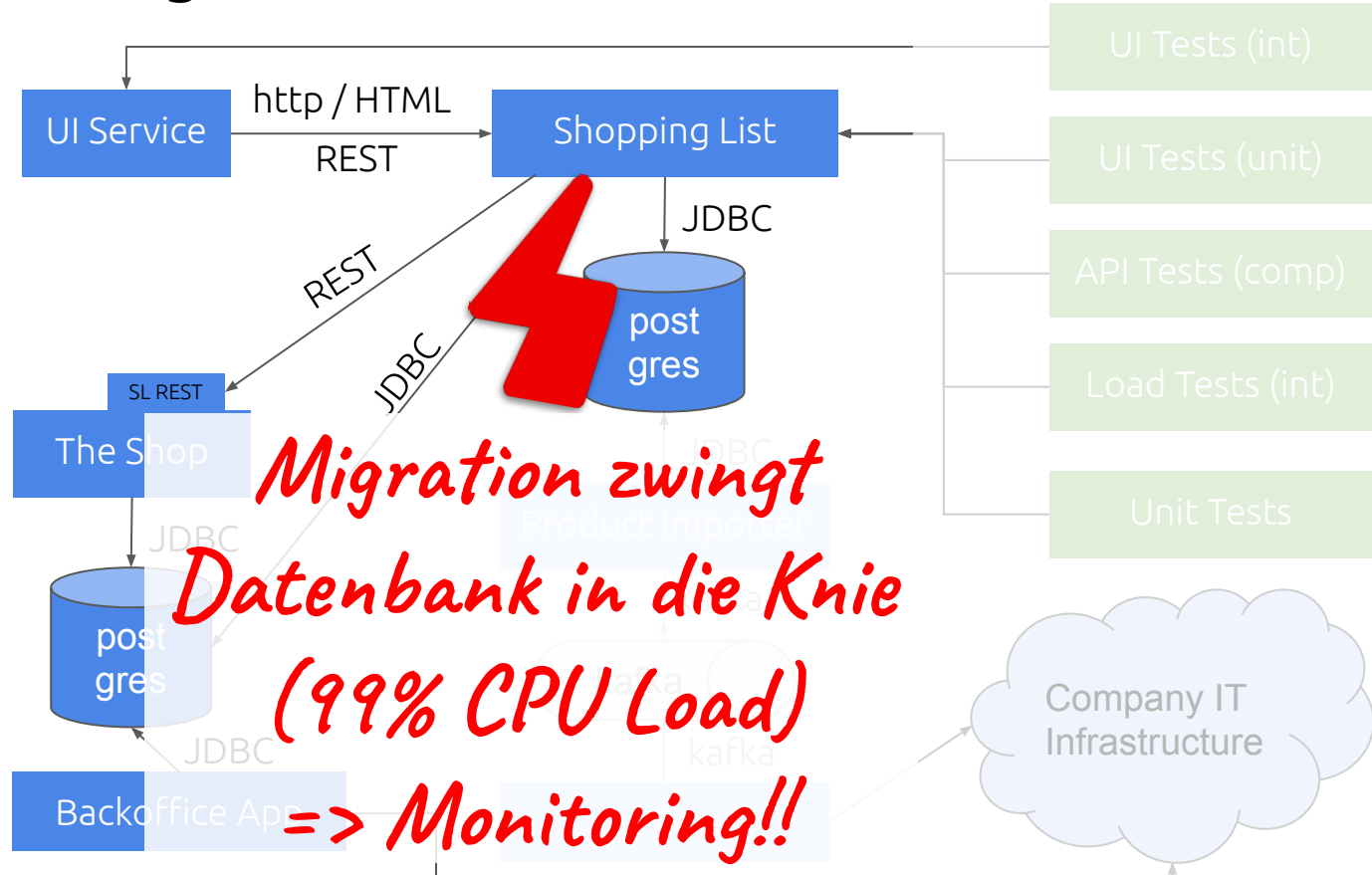
Ice
Taco
Wing Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Beispiel 2 - Die Einkaufsliste

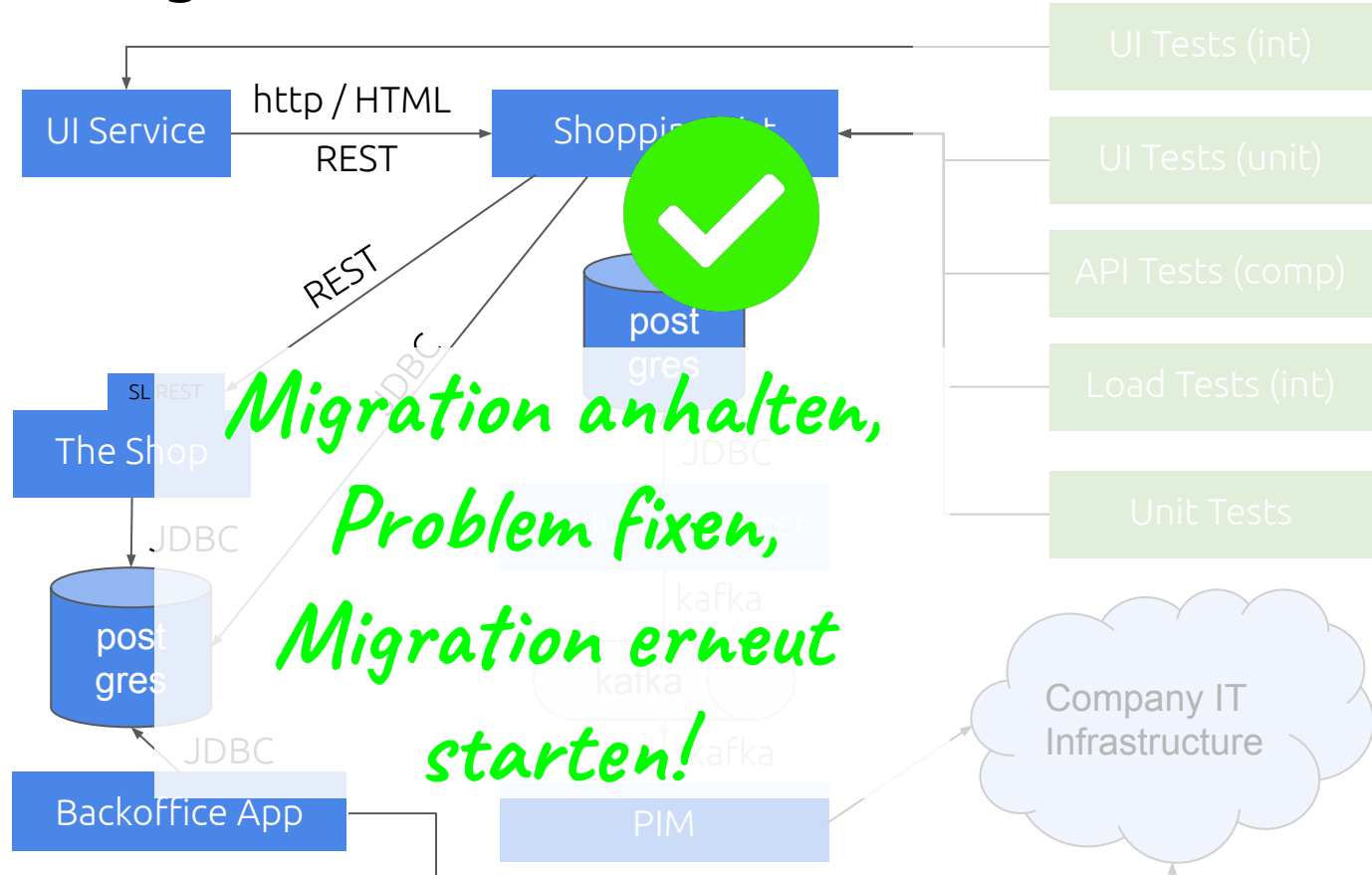
Migrationsschritt 4: Problem 1





Beispiel 2 - Die Einkaufsliste

Migrationsschritt 4: Problem 1



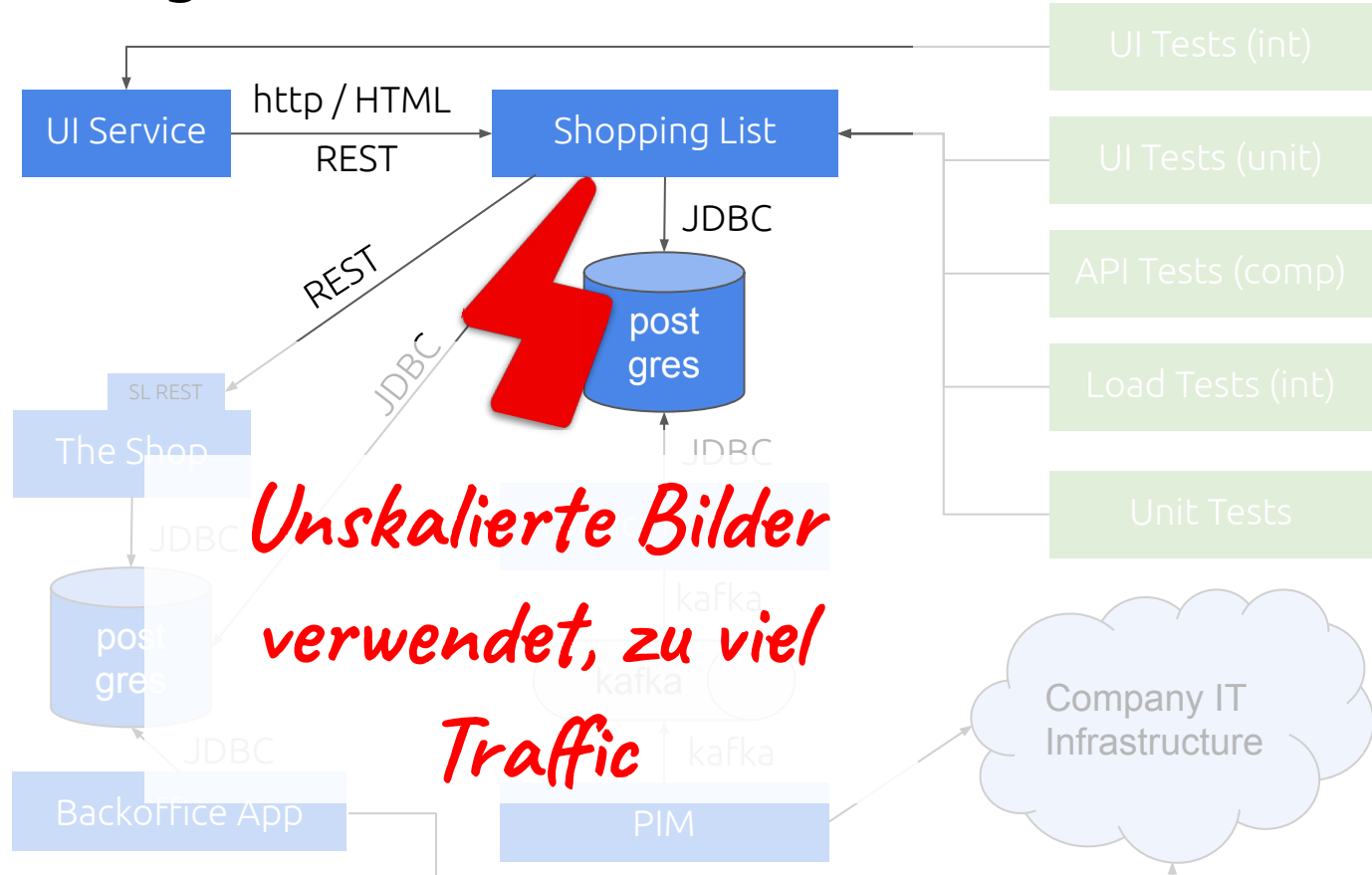
Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Beispiel 2 - Die Einkaufsliste

Migrationsschritt 4: Problem 2



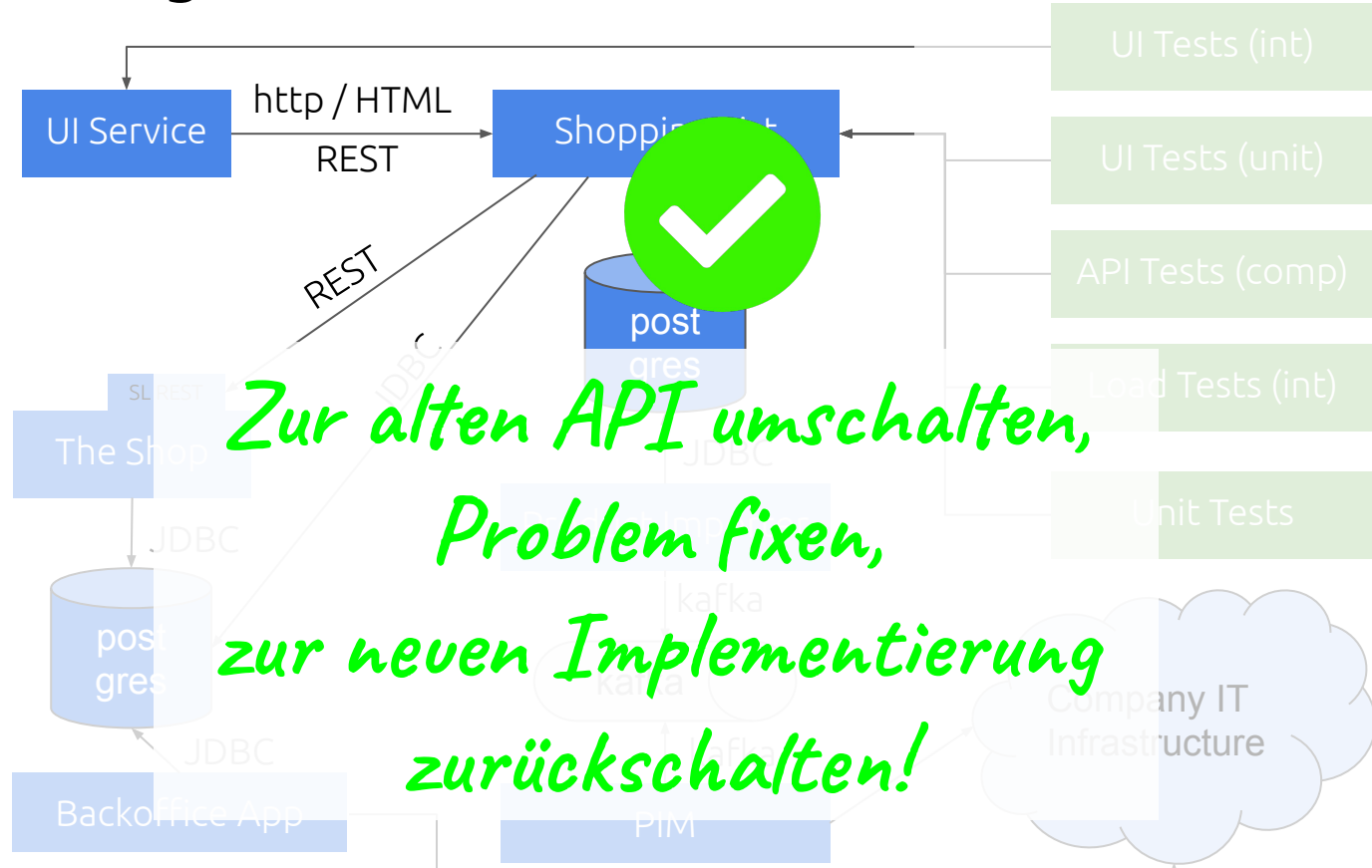
Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595



Beispiel 2 - Die Einkaufsliste

Migrationsschritt 4: Problem 2



Hotel Monteleone
Buttermilk
Ice
Jack Daniels
Ling Zang
Car Max

214 Rue Royale • New Orleans, LA
Telephone: (504) 523-3341 Fax: (504) 681-4413
Toll Free: 1-800-535-9595

Kunde



Der Safe





Beispiel 3 - Der Safe

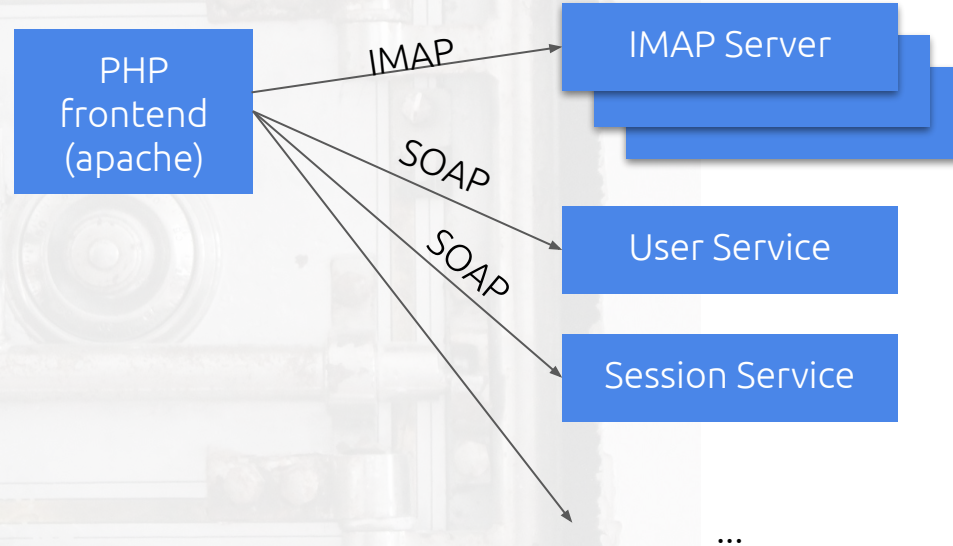
Der Safe

- Großer deutscher Konzern in der **Logistikbranche**
- Produkt: **Sichere E-Mails**
 - **> 1 Million registrierte Kunden** (aktiv?)
 - großes, komplexes **PHP System** (sollte ursprünglich nur ein Prototyp sein ...)
- Zusätzliches Feature: **Der Safe** (sicherer Google Drive clone)
 - **Safe Dokumente** wurden als MIME-Messages in **IMAP-Server** gespeichert



Beispiel 3 - Der Safe

Legacy Architektur





Beispiel 3 - Der Safe

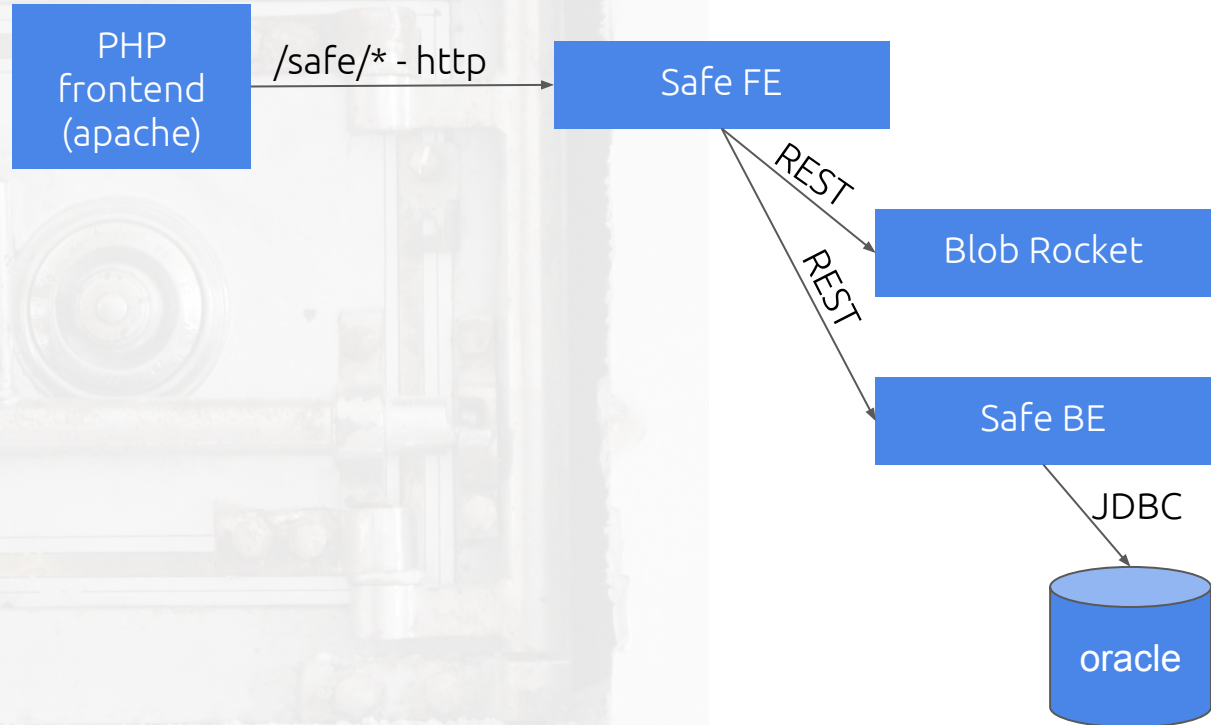
Probleme

- **zu langsam**
- **unnötige Größenbeschränkungen** (so etwas wie 50 MB)
- **nicht unabhängig entwickelbar** / deploybar / betreibbar
- **schlechte Testbarkeit** (nur manuelle UI-Tets)
- **hohe Fehleranfälligkeit**
- **API für andere Services** benötigt



Beispiel 3 - Der Safe

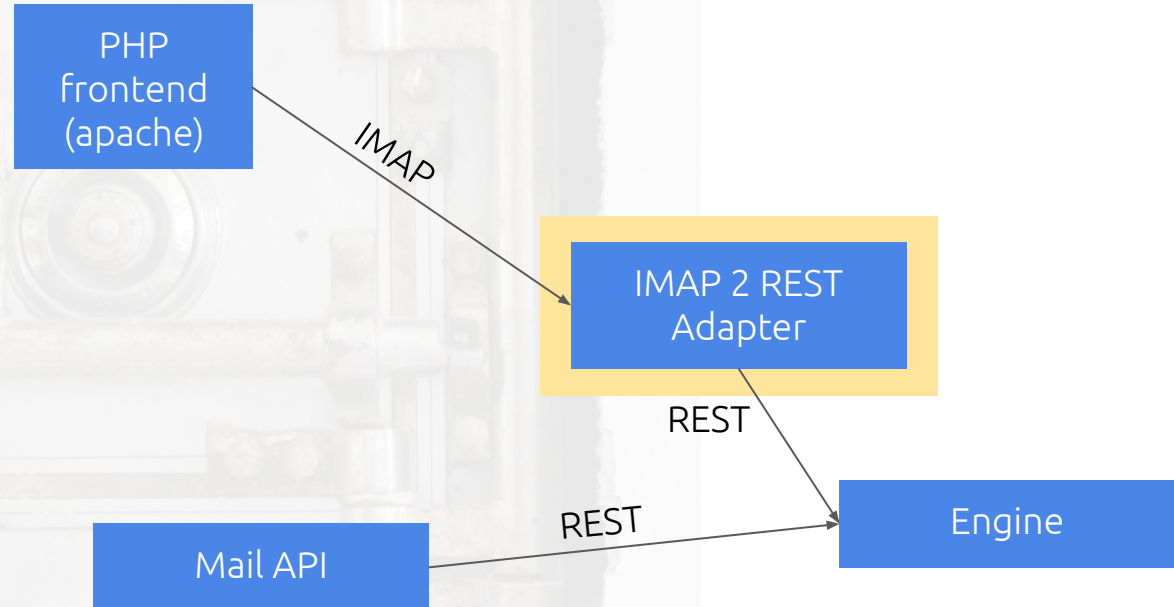
Zielarchitektur





Beispiel 3 - Der Safe

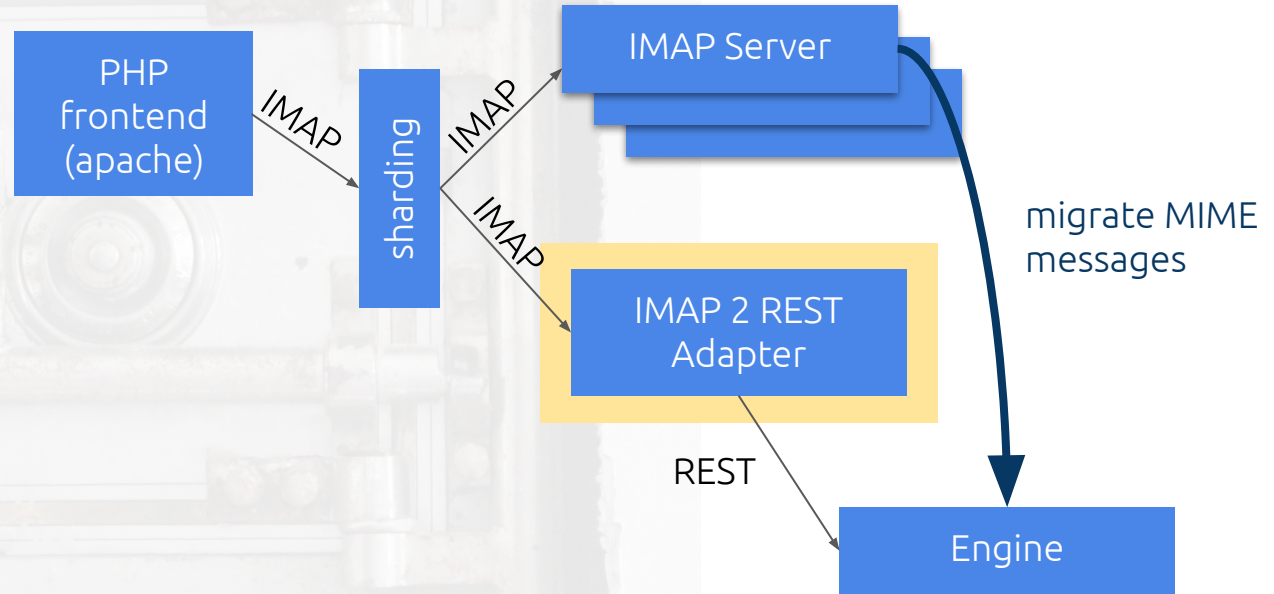
Migrationsschritt 1: Ziel





Beispiel 3 - Der Safe

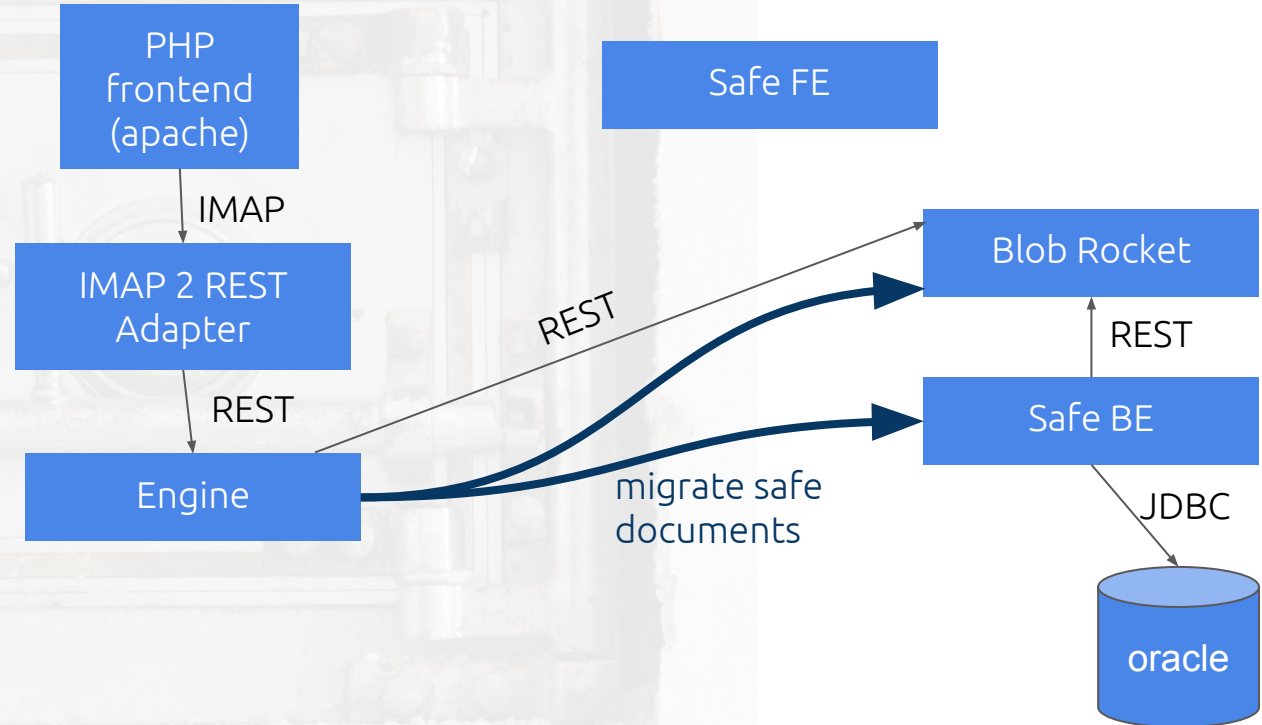
Migrationsschritt 1





Beispiel 3 - Der Safe

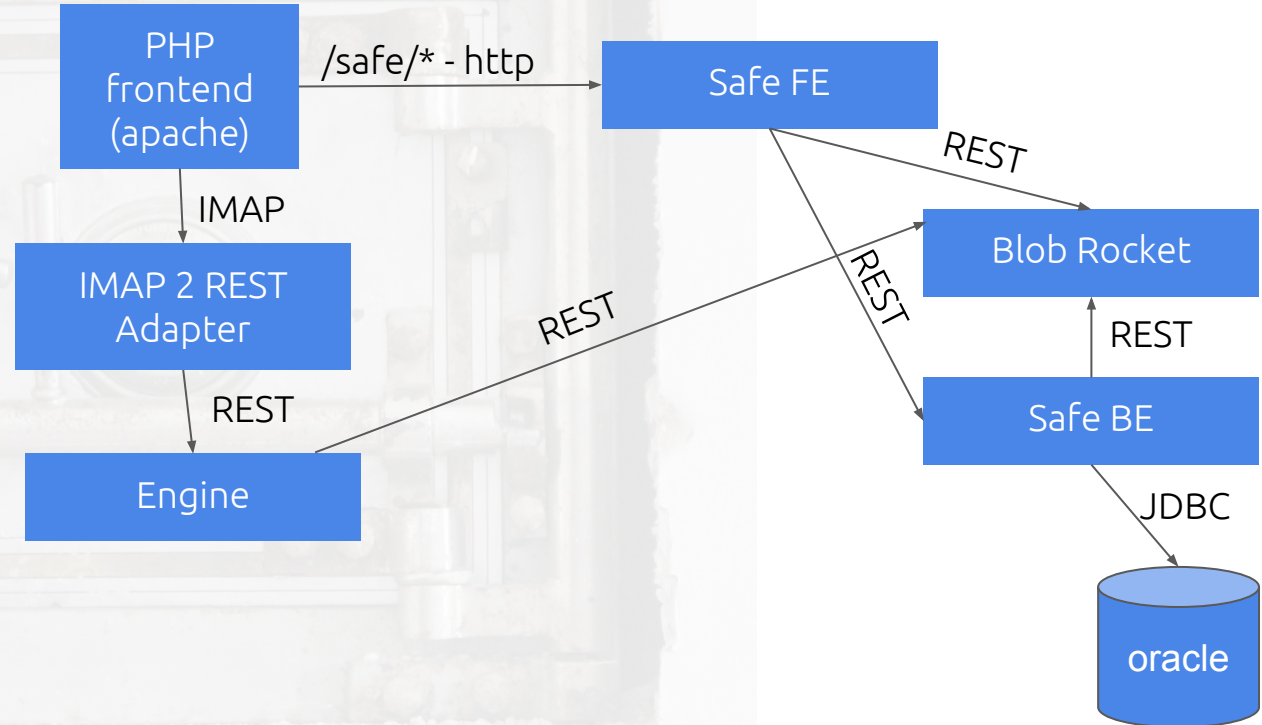
Migrationsschritt 2





Beispiel 3 - Der Safe

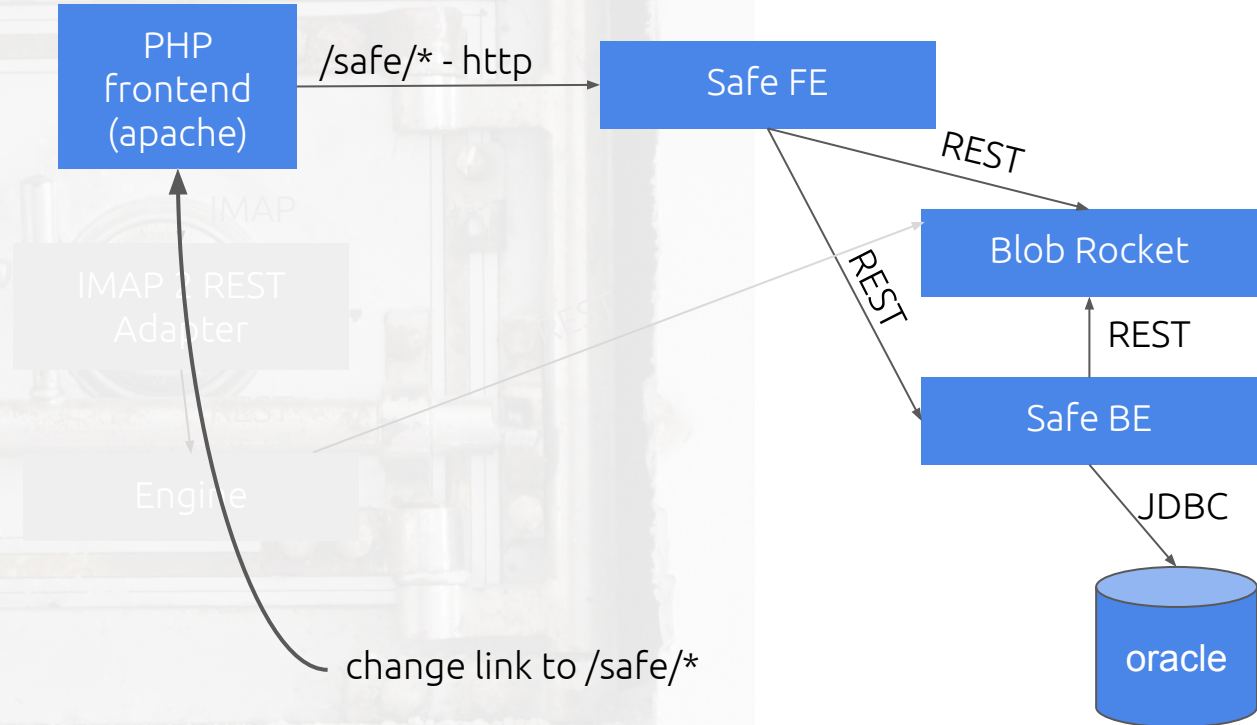
Migrationsschritt 2





Beispiel 3 - Der Safe

Migrationsschritt 2



Kunde



Vergleich



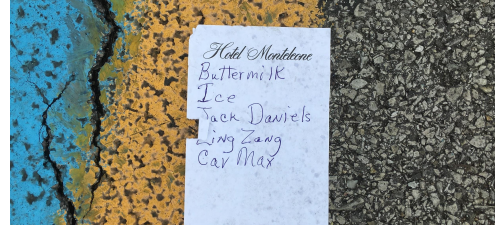


Der Warenkorb



- ⊖ Kunden unglücklich 😡
- ⊖ Rollback mit Datenverlust
- ⊖ keine Datenmigration
- ⊖ keine repräsentativen Testdaten
- ⊖ geteilte Verantwortlichkeit
- ⊖ nicht genügend Tests
- ⊖ nicht genügend Monitoring

Die Einkaufsliste



- ⊕ Kunden zufrieden
- ⊕ Rollback ohne Datenverlust
- ⊕ Datenmigration (einzelne User)
- ⊕ Testen in PROD
- ⊕ einzelne Verantwortlichkeit
- ⊕ aut. Tests für jeden Schritt
- ⊕ ordentliches Monitoring

Der Safe



- ⊕ keine unzufriedenen Kunden 😊
- ⊕ Rollback ohne Datenverlust
- ⊕ Datenmigration (einzelne User)
- ⊕ Testen in PROD
- ⊕ einzelne Verantwortlichkeit
- ❓ automat. Tests für jeden Schritt
- ⊖ nicht genügend Monitoring

Key Learnings





Migrationen werden fehlschlagen

... wegen inkonsistenten & ungültigen Daten, zu wenig Plattenplatz oder RAM, Netzwerkfehler, Nebenläufigkeit, Encoding ...

- **Rollbackfähigkeit herstellen** - falls Dinge schief gehen (z.B. Feature Toggles und Canary Releases).
- **Datenverlust** bei Rollbacks **verhindern** (z.B. mit Adaptern).
- **Gutes Fehlerlogging** - so dass man sehen kann, was schief gegangen ist und genug Informationen hat das zu fixen.
- **Fehlertolerant sein.** Migration nicht abbrechen, wenn ein einzelner Eintrag fehlschlägt
- **Retry Mechanismus** für fehlgeschlagene Einträge bereitstellen



Wahrscheinlichkeit und Auswirkungen von Fehlern reduzieren

- **User einzeln migrieren können.** Ermöglicht Vorab-Test mit ausgewählten Usern. Ein Teil der User kann bereits auf dem Neusystem arbeiten, während problematische User noch auf dem Altsystem sind.
- **Migration stoppen und neu starten können.** Wenn etwas schief geht möchte man eingreifen können.
- **Außerhalb der Nutzungszeit migrieren** - zumindest wenn sich Probleme bei der Migration evtl. direkt auf Benutzer auswirken.
- **Überlast vermeiden.** Ggf. Migrationsjob drosseln.
- **Datenimporte während der Migration anhalten.** Eine Problemquelle weniger.



Gut Testen

- **Produktivdaten verwenden um die Migration zu testen**, wenn möglich. Migration in PROD ausprobieren und das Resultat z.B. über Feature Toggles verstecken, bis es verifiziert ist.
- **Automatisierte Integrationstests erstellen**, die gegen das neue und das alte System laufen können. Das Altsystem dient als [Testorakel](#).
- **Migration wie ein Refactoring behandeln**. Nur die Struktur ändern, nicht die Funktionalität. Mit Tests beweisen, dass der jeweilige Schritt erfolgreich war.



Transparenz herstellen

- **Eine einzelne Kontrollinstanz** für den gesamten Migrationsprozess haben, z.B. das Entwicklungsteam, das für die jeweilige Funktionalität verantwortlich ist.
- **Eine saubere Migration ist teuer und dauert lange.**
Erwartungsmanagement betreiben 😊
- **Sicherstellen, dass es eine gute, leicht verständliche und allgemein zugängliche Dokumentation gibt** von der gesamten Migration inklusive aller geplanten Schritte - und regelmäßig aktualisieren. Das sollte Transparenz schaffen, Missverständnisse vermeiden und es einfacher machen über die Migration zu reden - gerade mit neuen Mitarbeitern.



Migration Monitoren

- **Technisches Monitoring und Alerting einsetzen** um schnell festzustellen wenn etwas schief geht und zeitnah reagieren zu können
- **Fachliche Metriken verwenden.** Das zeigt einem, ob ein System nach der Migration genauso genutzt werden kann wie vorher.
- **Bereitschaft für Mitarbeiter während Migration.** Es ist nicht unwahrscheinlich, dass Probleme auftreten und während der Migration behoben werden müssen.



Key Learnings

Jede Migration ist unterschiedlich



- ... es gibt **keinen einzelnen Ansatz**, der auf **jedes Szenario** passt ...
- Man sollte **Kosten** und **Nutzen** abwägen!



tarent



Wenn Ihr Hilfe braucht

- mit einer **Migration**,
- einem **Software Projekt** im allgemeinen
(Webdevelopment, Microservices, AI, Machine Learning, AR / VR)
- oder beim Aufbauen von **Wissen und Skills** in diesen Gebieten
(**Training**)

Sprecht uns gerne an!

Vielen Dank!

Tim Steffens

Softwareentwickler &
Trainer

t.steffens@tarent.de
[@tmstff](https://github.com/tmstff)

Rochusstraße 2-4
53123 Bonn

Telefon: 0228 54881-0
Telefax: 0228 54881-235

academy@tarent.de
www.tarent.de/academy

