horze

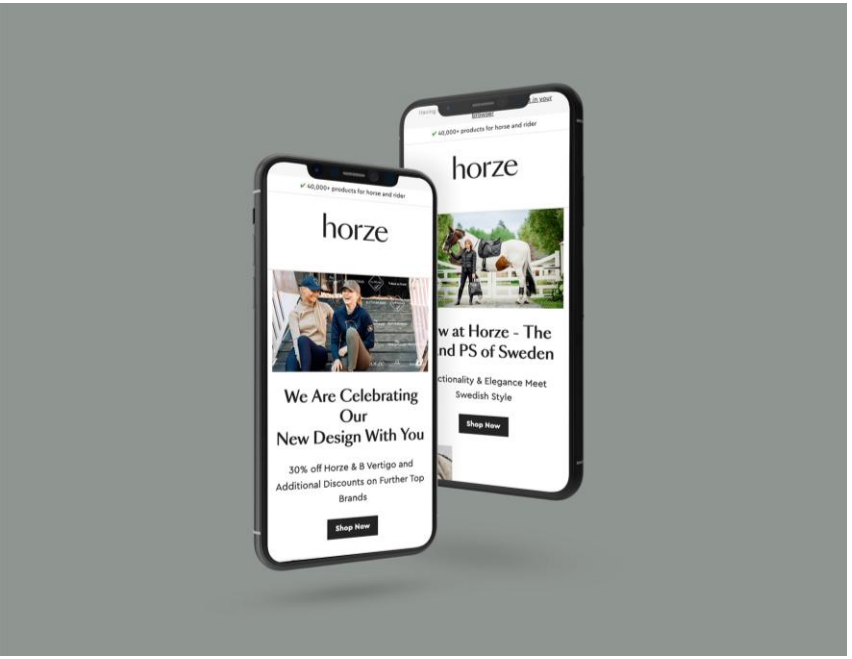
Entwicklung & Bereitstellung eines API-Endpunktes

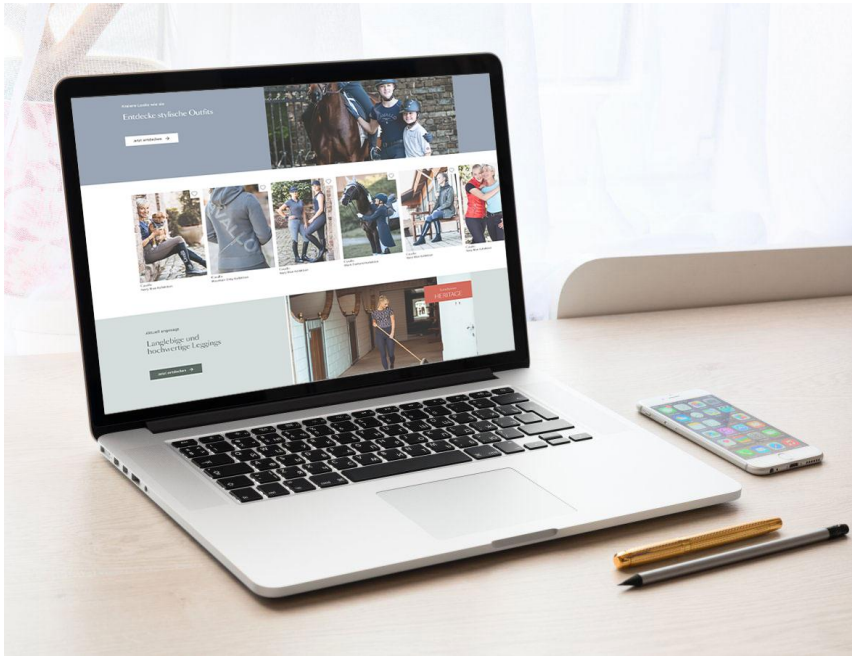
zur optimierten Abwicklung der Retourenanmeldung durch Kunden

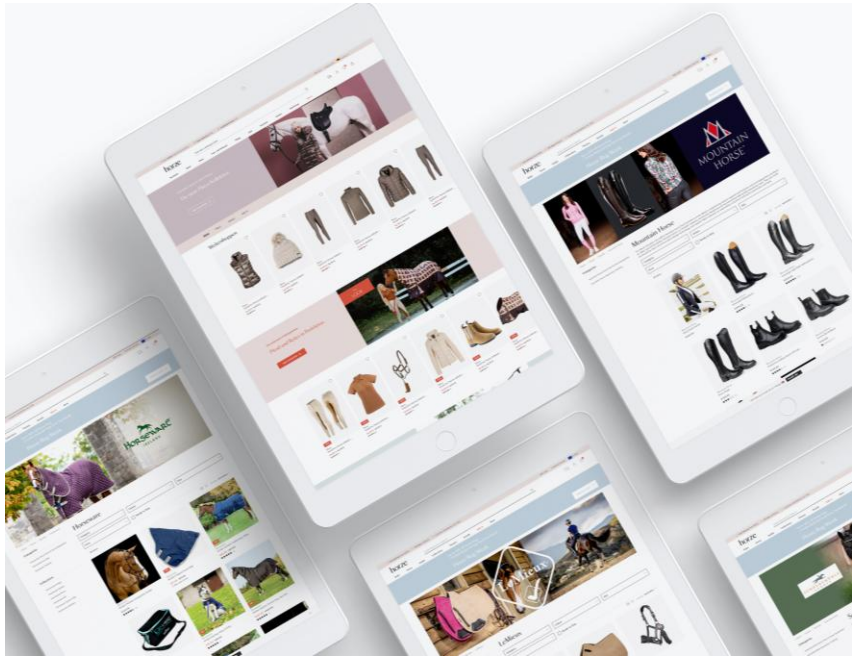
Horze International GmbH

Onlineversandhandel mit 20 Onlineshops in 18 Ländern und weltweitem Versand

Das Sortiment umfasst über 40.000 Artikel für den Reitsport und Pferdebedarf

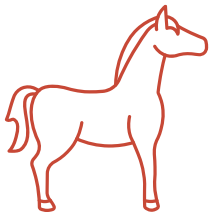




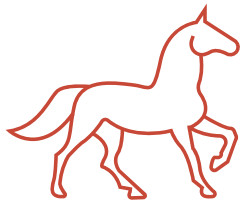


Agenda

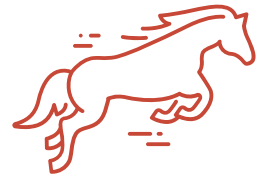
horze



Ausgangsituation
& Projektziel



Projektphasen



Abschluss

Voice of the Customer

horze

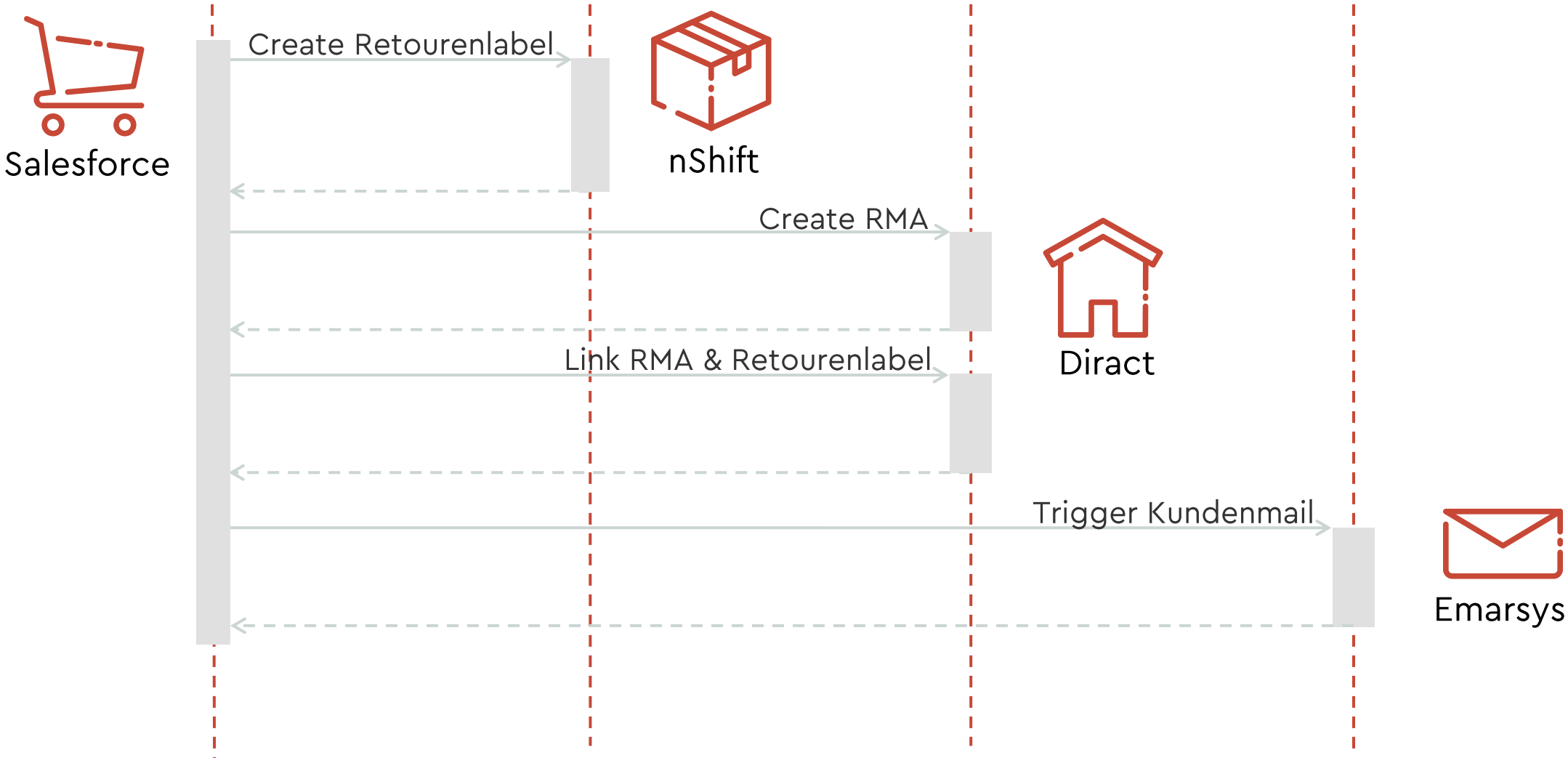
"The return process on the website or app should be improved! IT DID NOT WORK."

"The process is long, and the website is not always up to date to complete the return request."

"The return process is confusing on the website."

IST-Situation

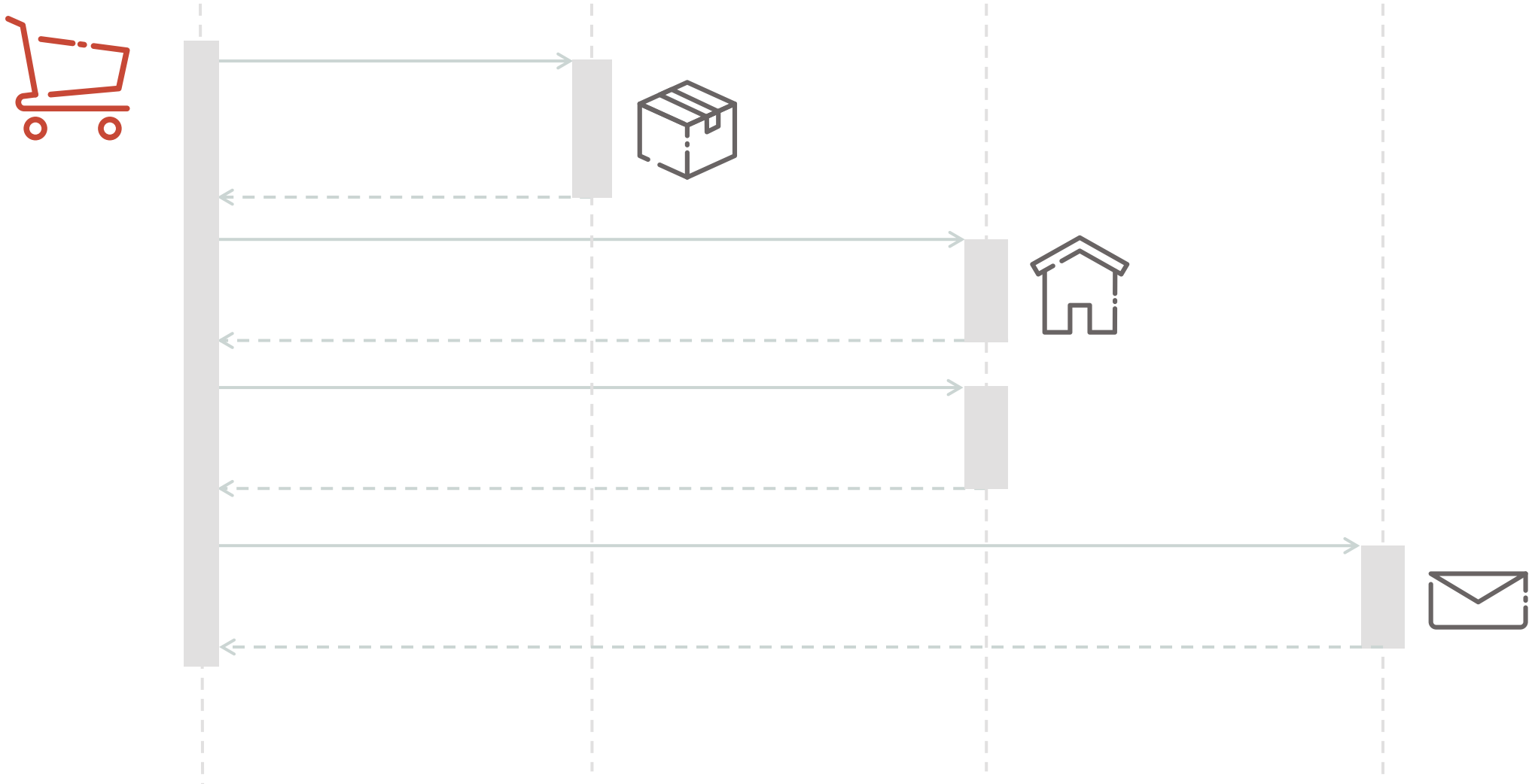
horze



IST-Situation

horze

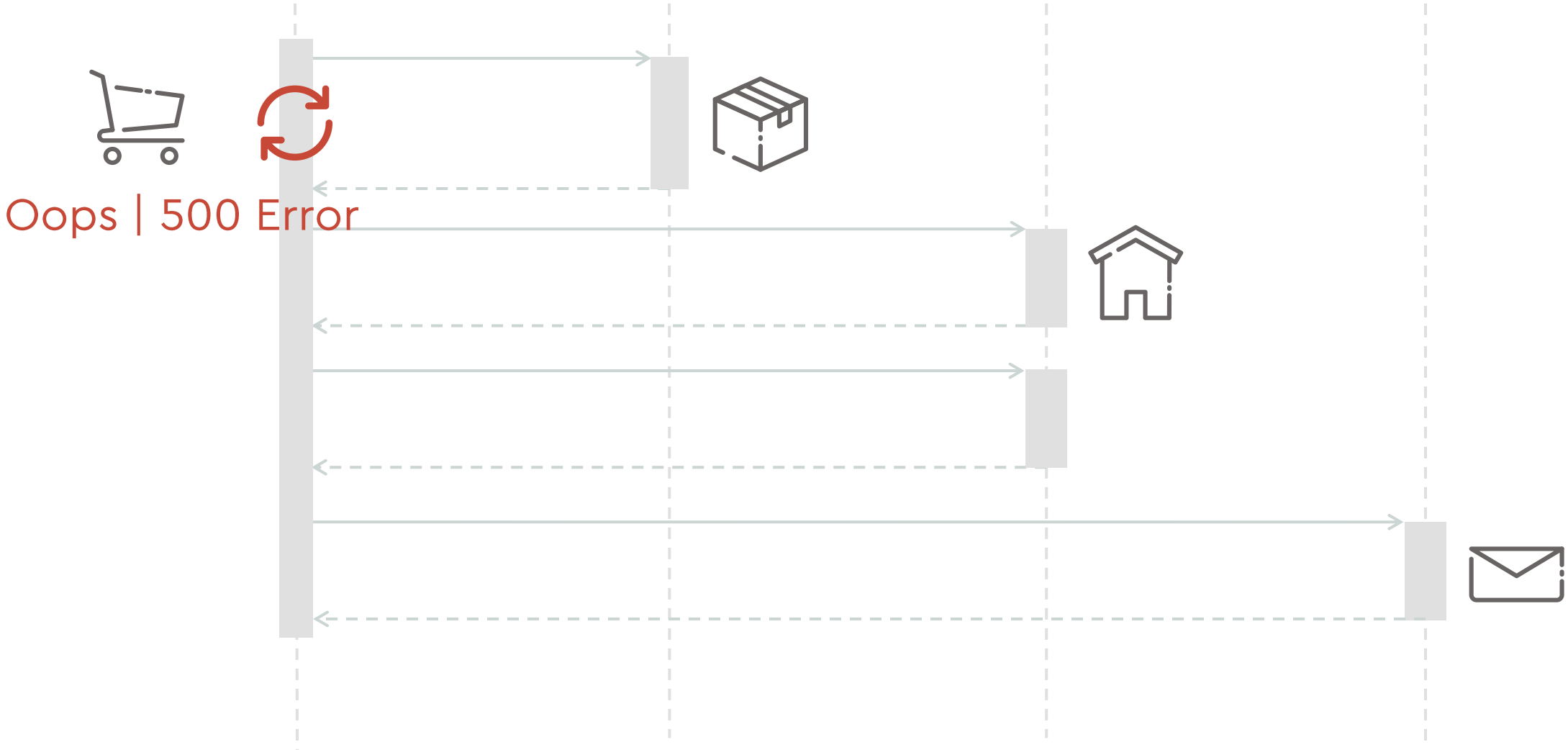
Nur sequenzielle Abarbeitung möglich



IST-Situation

horze

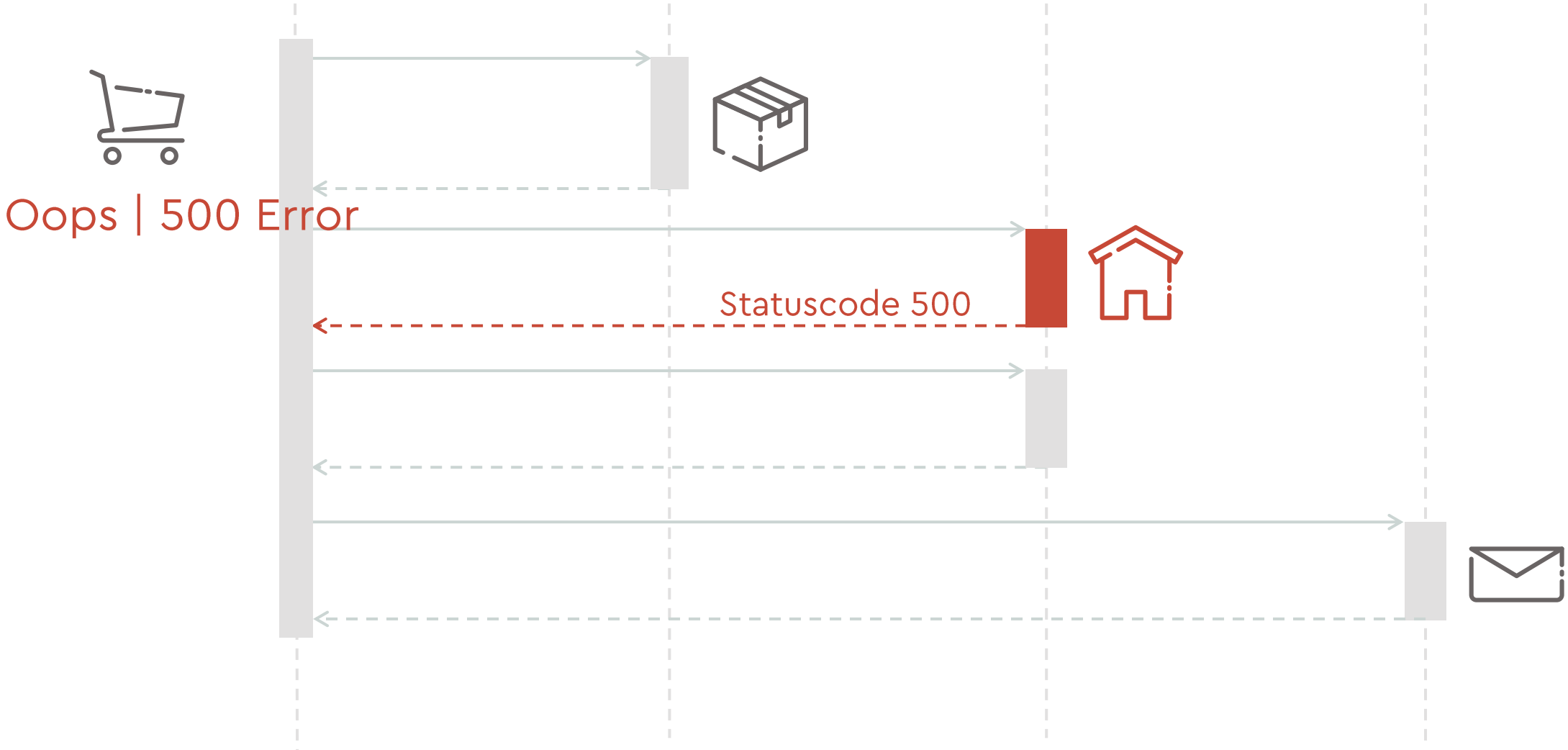
Lange Bearbeitungszeit, Salesforce bricht nach > 5 Sekunden ab



IST-Situation

horze

Lange Bearbeitungszeit, Salesforce bricht nach > 5 Sekunden ab



Projektziel

horze

Um das Kundenerlebnis beim Anmelden einer Retoure über den Webshop zu verbessern, sollen fehlgeschlagene Retourenanmeldungen **(HTTP 500 Fehler) auf ein Minimum reduziert werden**. Den Kunden soll stets vermittelt werden, dass die Retourenanmeldung erfolgreich war. Um dies zu erreichen, soll die bestehende Prozesslogik aus dem Webshop in die interne Backend-Umgebung verlagert werden. Dies erfolgt über einen neuen API-Endpunkt. Zusätzlich wird die Prozesslogik durch ein Warteschlangen-System (Queue) ergänzt. Falls eine Anfrage aufgrund eines Fehlers nicht bearbeitet werden kann, wird sie von einem Service Worker erneut verarbeitet.

Projektziel

horze

Um das Kundenerlebnis beim Anmelden einer Retoure über den Webshop zu verbessern, sollen fehlgeschlagene Retourenanmeldungen **(HTTP 500 Fehler) auf ein Minimum reduziert werden**. Den Kunden soll stets vermittelt werden, dass die Retourenanmeldung erfolgreich war. Um dies zu erreichen, soll **die bestehende Prozesslogik aus dem Webshop in die interne Backend-Umgebung verlagert werden**. Dies erfolgt über einen neuen API-Endpunkt. Zusätzlich wird die Prozesslogik durch ein Warteschlangen-System (Queue) ergänzt. Falls eine Anfrage aufgrund eines Fehlers nicht bearbeitet werden kann, wird sie von einem Service Worker erneut verarbeitet.

Projektziel

horze

Um das Kundenerlebnis beim Anmelden einer Retoure über den Webshop zu verbessern, sollen fehlgeschlagene Retourenanmeldungen **(HTTP 500 Fehler) auf ein Minimum reduziert werden**. Den Kunden soll stets vermittelt werden, dass die Retourenanmeldung erfolgreich war. Um dies zu erreichen, soll **die bestehende Prozesslogik aus dem Webshop in die interne Backend-Umgebung verlagert werden**. Dies erfolgt über einen neuen **API-Endpunkt**. Zusätzlich wird die Prozesslogik durch ein Warteschlangen-System (Queue) ergänzt. Falls eine Anfrage aufgrund eines Fehlers nicht bearbeitet werden kann, wird sie von einem Service Worker erneut verarbeitet.

Projektziel

horze

Um das Kundenerlebnis beim Anmelden einer Retoure über den Webshop zu verbessern, sollen fehlgeschlagene Retourenanmeldungen **(HTTP 500 Fehler) auf ein Minimum reduziert werden**. Den Kunden soll stets vermittelt werden, dass die Retourenanmeldung erfolgreich war. Um dies zu erreichen, soll **die bestehende Prozesslogik aus dem Webshop in die interne Backend-Umgebung verlagert werden**. Dies erfolgt über einen neuen **API-Endpunkt**. Zusätzlich wird die Prozesslogik durch ein **Warteschlangen-System (Queue)** ergänzt. Falls eine Anfrage aufgrund eines Fehlers nicht bearbeitet werden kann, wird sie von einem **Service Worker** erneut verarbeitet.

Projektphasen

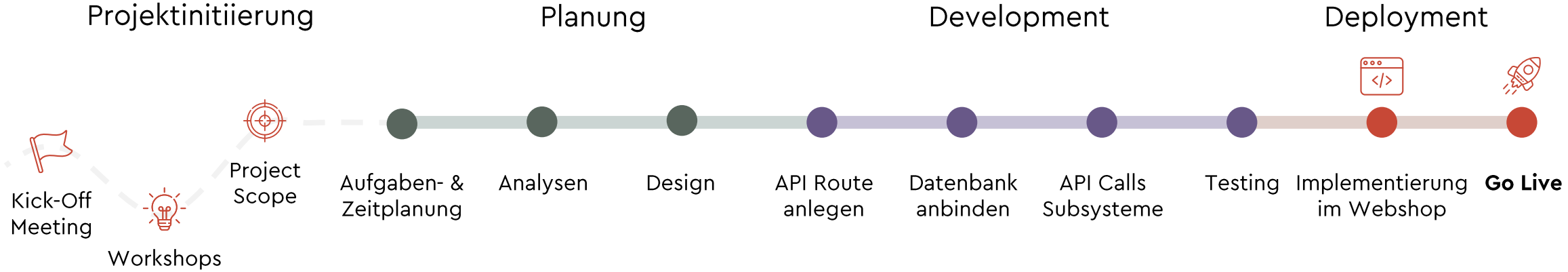
horze

Zur Umsetzung des Projektes standen **80** Zeitstunden zur Verfügung

Projektinitiierung	Planung	Development	Deployment
8 h	16 h	40 h	16 h

Roadmap

horze



Projektinitiierung

horze

Kick-Off Meeting

Workshops

Project Scope

Projektinitiierung

horze

Kick-Off Meeting

Workshops

Project Scope

Web

Supply Chain

Customer Service

Projektinitiierung

horze

Kick-Off Meeting

Workshops

Project Scope

Azubi

Projektleiter

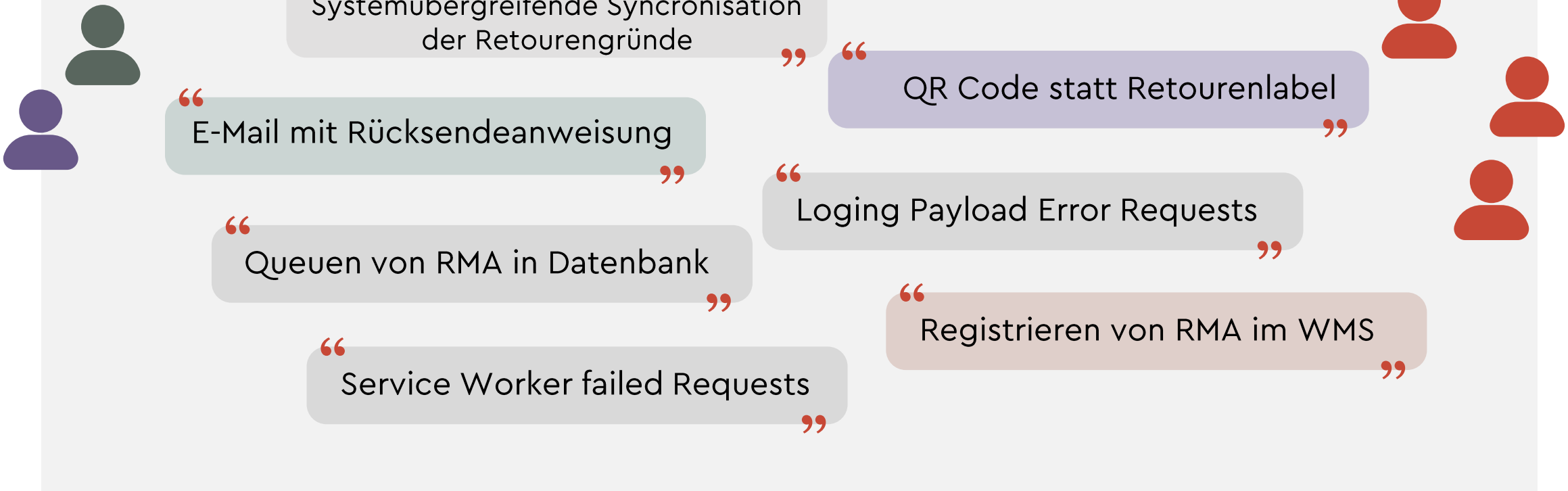
Projektinitiierung

horze

Kick-Off Meeting

Workshops

Project Scope



Projektinitiierung

horze

Kick-Off Meeting

Workshops

Project Scope

M
Must have

- Queuen von RMA in Datenbank
- Generieren der Returnlable
- Registrieren von RMA im WMS
- Verknüpfung RMA & Trackingcode
- E-Mail mit Rücksendeanweisung
- Service Worker failed Requests

S
Should have

- Loging Payload Error Requests

C
Could have

- QR Code statt Retourenlabel
- Auswahl Versanddienstleister

W
Won't have

- Systemübergreifende Syncronisation der Retourengründe

Projektinitiierung

horze

Kick-Off Meeting

Workshops

Project Scope

M
Must have

- Queuen von RMA in Datenbank
- Generieren der Returnlable
- Registrieren von RMA im WMS
- Verknüpfung RMA & Trackingcode
- E-Mail mit Rücksendeanweisung
- Service Worker failed Requests

S
Should have

- Loging Payload Error Requests

C
Could have

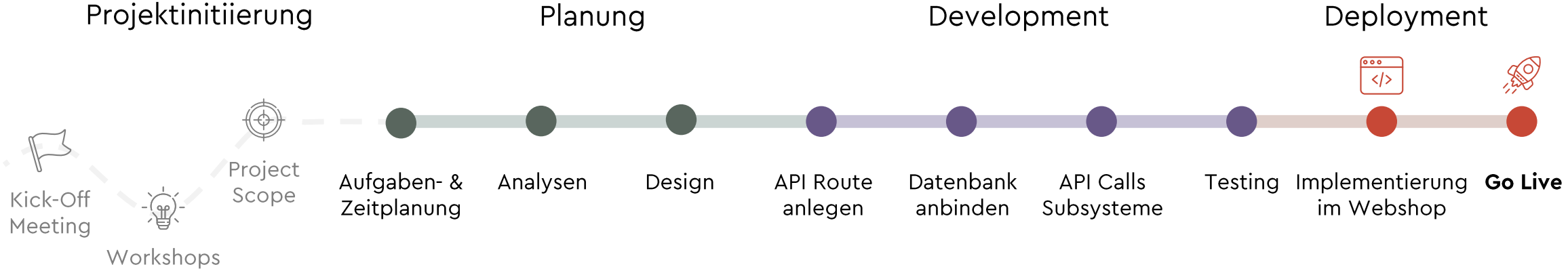
- QR Code statt Retourenlabel
- Auswahl Versanddienstleister

W
Won't have

- Systemübergreifende Syncronisation der Retourengründe

Roadmap

horze



Planung

horze

- Aufgaben- & Zeitplanung
- Analysen
- Design

Planung

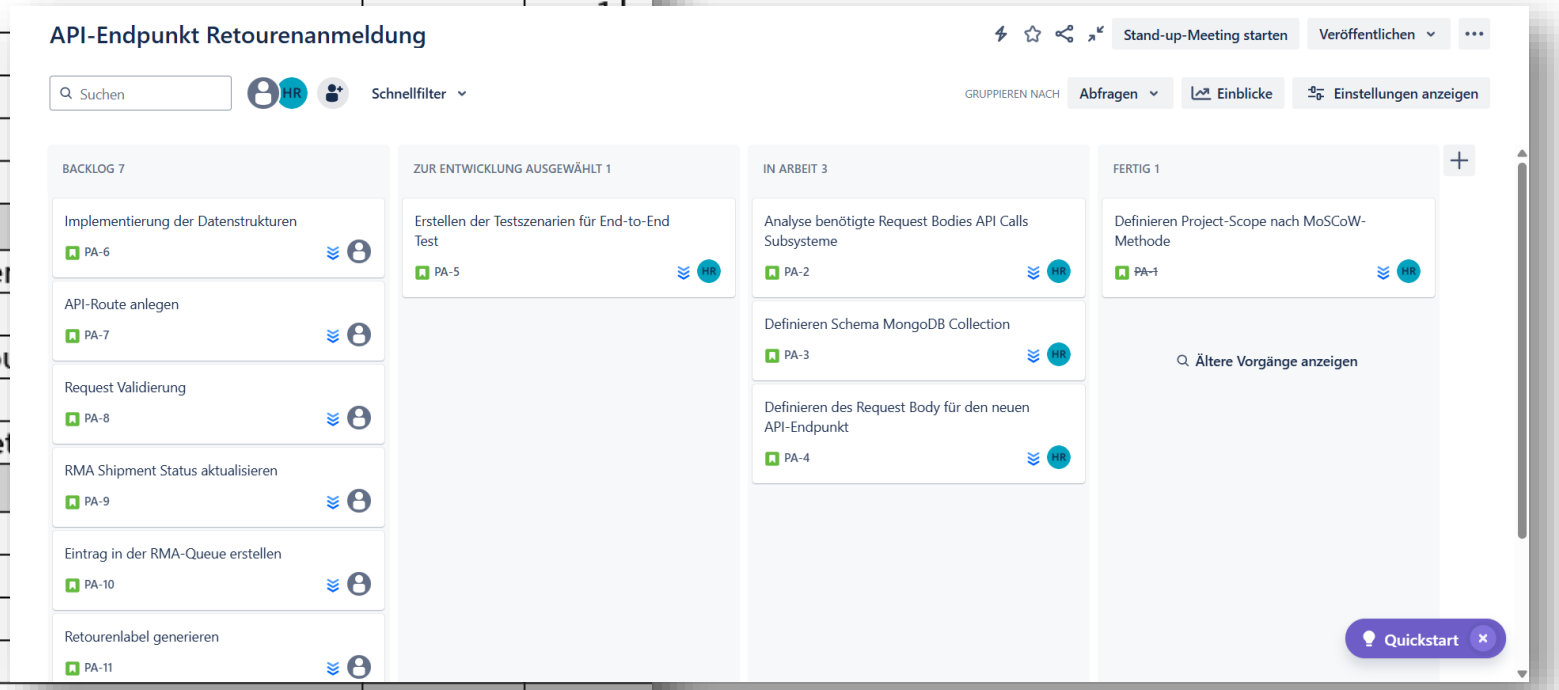
horze

Aufgaben- & Zeitplanung

Analysen

Design

	Ticket	Zeit
Projektinitiierung		
Kick-Off Meeting Fachbereiche Web, Customer Service & Supply Chain	-	1 h
Workshop Fachbereich Web		
Workshop Fachbereich Customer Service		
Workshop Fachbereich Supply Chain		
Definieren Project-Scope nach MoSCoW-Methode		
Präsentation Project-Scope		
Planung		
Analyse benötigte Request Bodies API Calls Subsyste		
Definieren Schema MongoDB Collection		
Definieren des Request Body für den neuen API-Endpu		
Erstellen der Testszenarien für End-to-End Test		
Beschreiben von Arbeitspaketen/ Erstellen von Ticket		
Development		
Einrichten der Entwicklungsumgebung		
Implementierung der Datenstrukturen		
API-Route anlegen		
Request Validierung		
RMA Shipment Status aktualisieren	PA-9	4 h
Eintrag in der RMA-Queue erstellen	PA-10	4 h
Retourenlabel generieren	PA-11	6 h



Planung

horze

 Analysen

 Design

Planung

horze

Analysen

Design

0.000 €

Gesamtkosten

horze

Planung

 Analysen

 Design

Projektinitiierung	480 €
Analyse & Planung	160 €
Implementierung	420 €
Tests	100 €
Deployment	105 €
Nachbereitung	80 €

1.345 €
Gesamtkosten

horze

Planung

Analysen

Design

Projektinitiierung	480 €
Analyse & Planung	160 €
Implementierung	420 €
Tests	100 €
Deployment	105 €
Nachbereitung	80 €
Ressourcenkostenpauschale	200 €

1.545 €

Gesamtkosten

Planung

horze

Analysen

Design

$$600\ Tickets \times 5\ min/Ticket = 50\ h/Monat$$

$$50\ h \times 25\ €/h = 1.250\ €/Monat$$

$$1.250\ €/Monat \times 12 = 15.000\ €/Jahr$$

$$\frac{\mathbf{1.545}€}{15.000€/\ Jahr} \approx 0{,}10\ Jahre \approx 1{,}25\ Monate$$

Planung

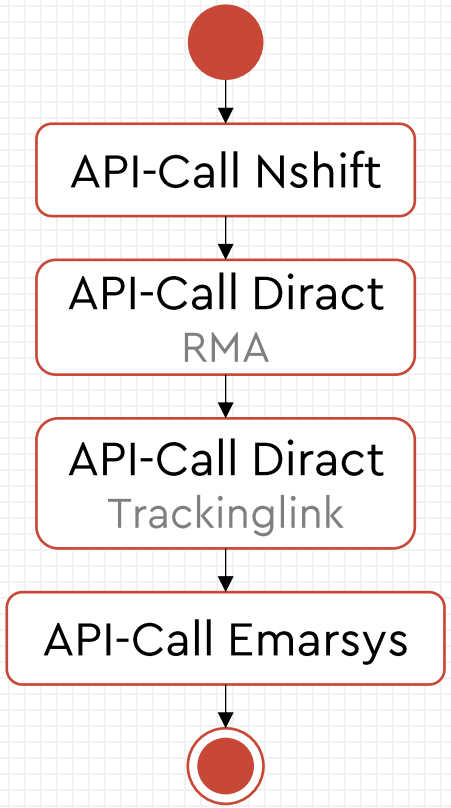
horze

Analysen

Design

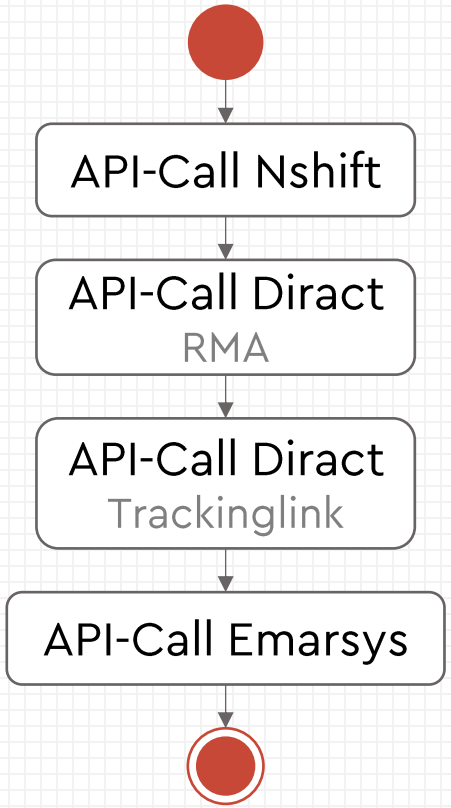
Design

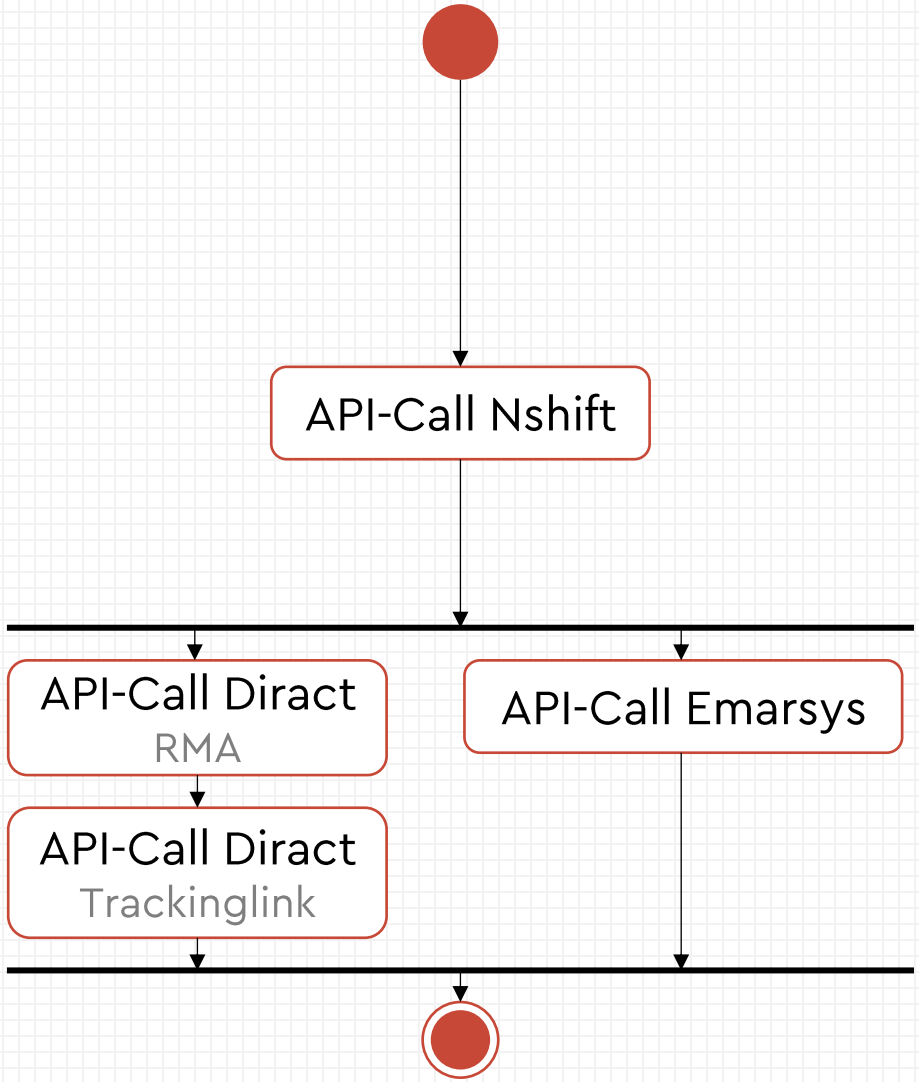
horze



Design

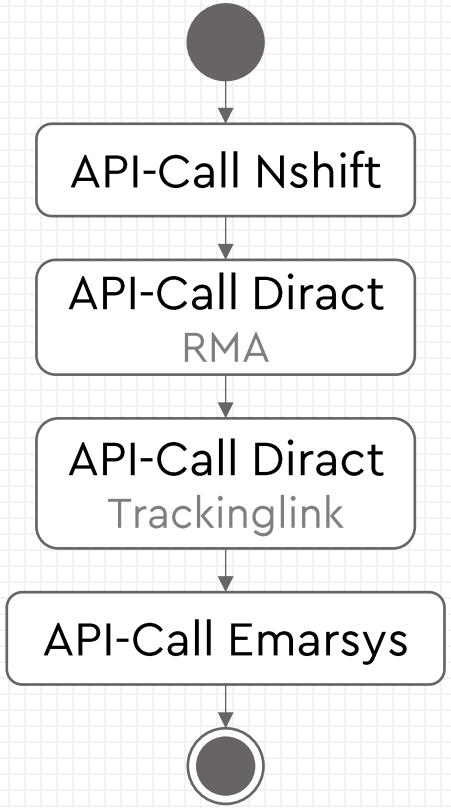
horze

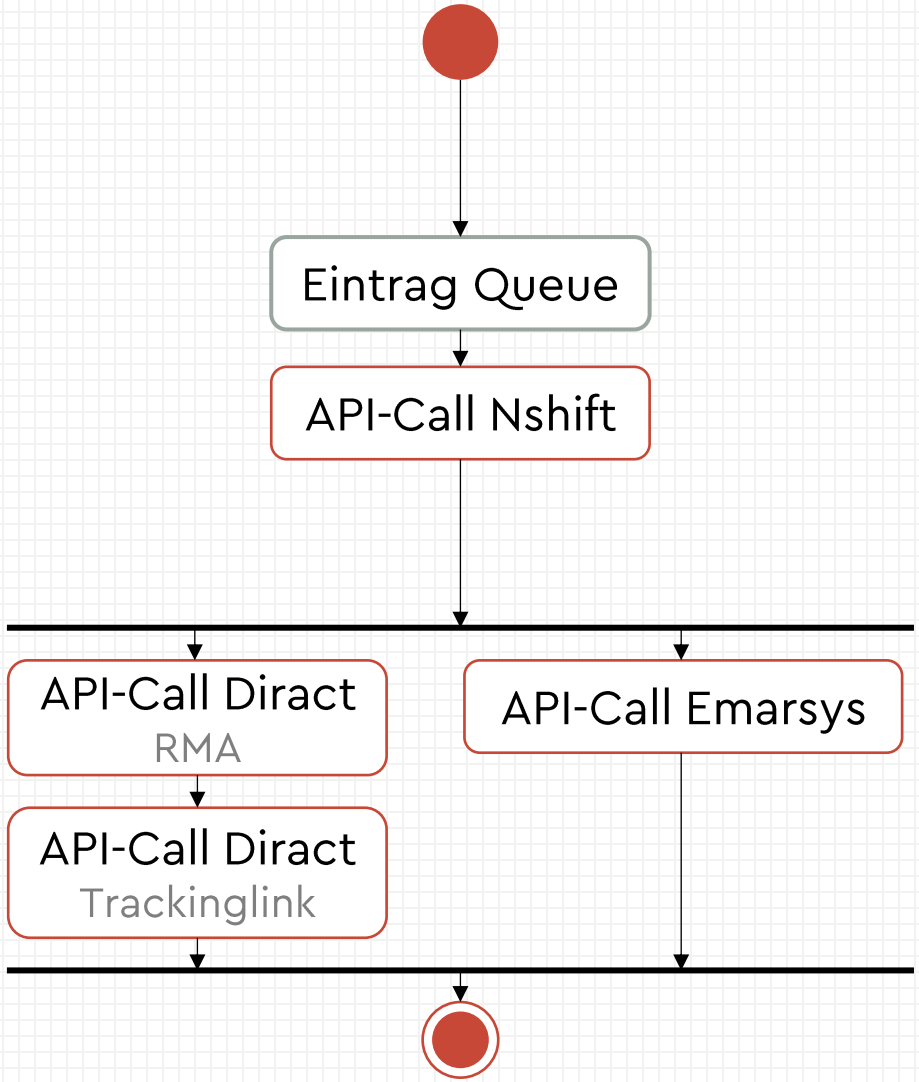




Design

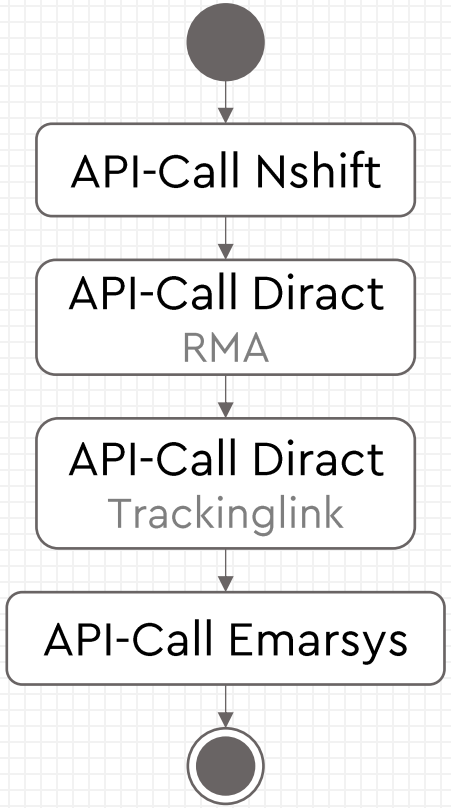
horze

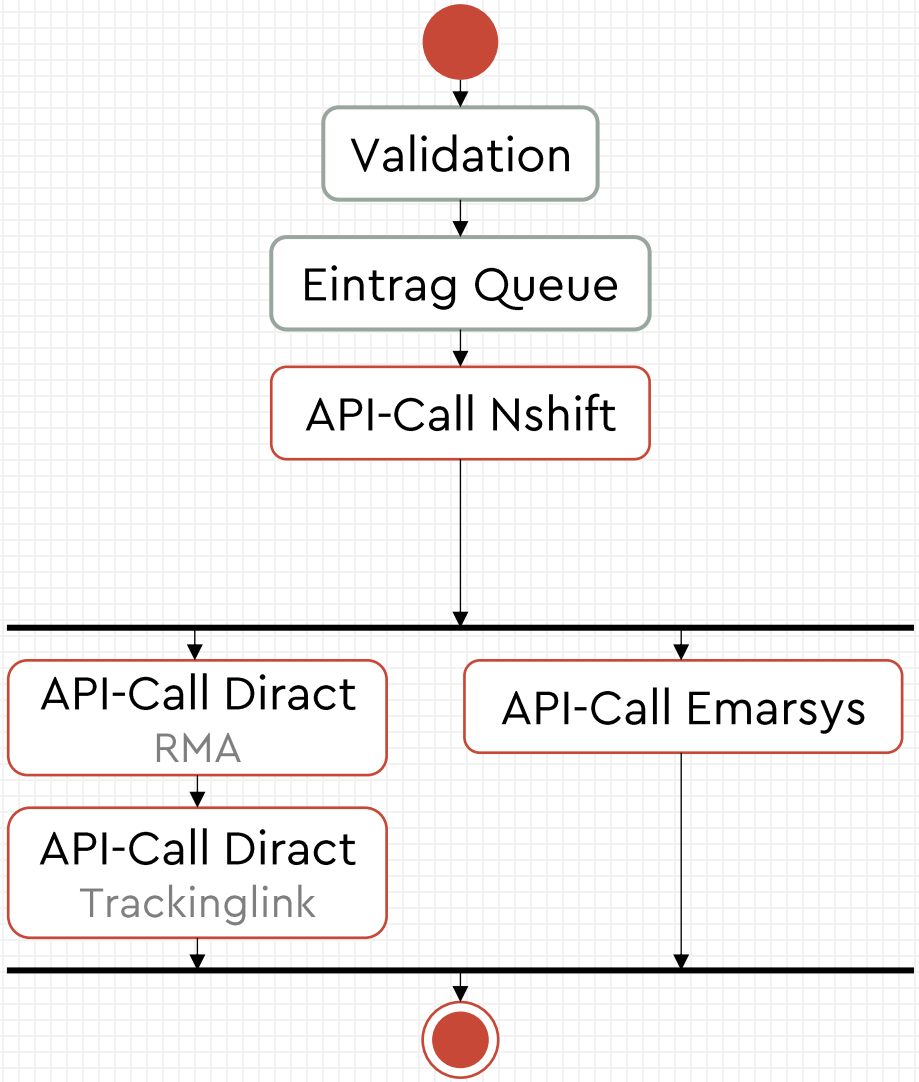




Design

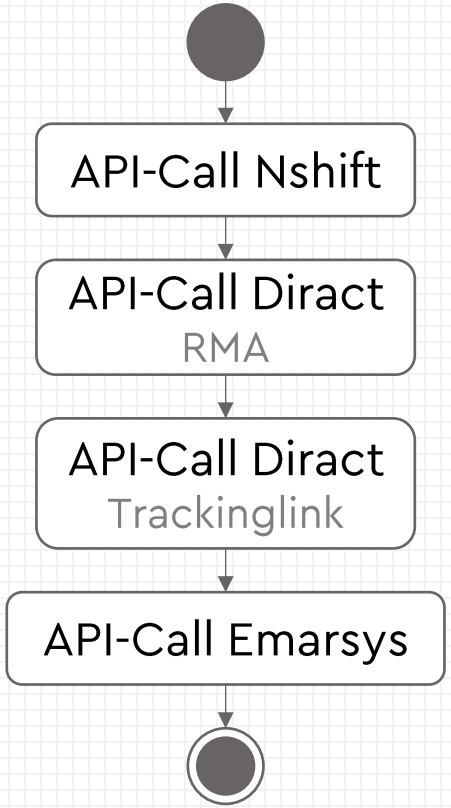
horze

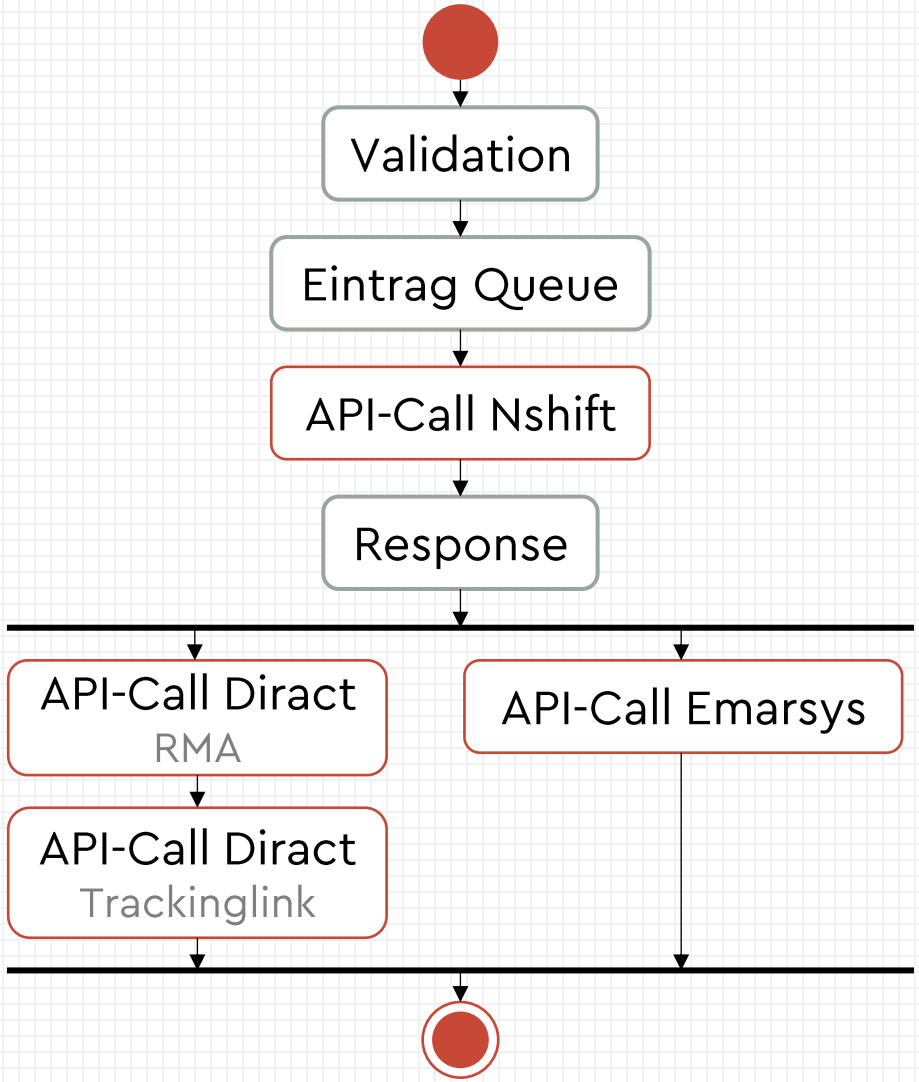




Design

horze

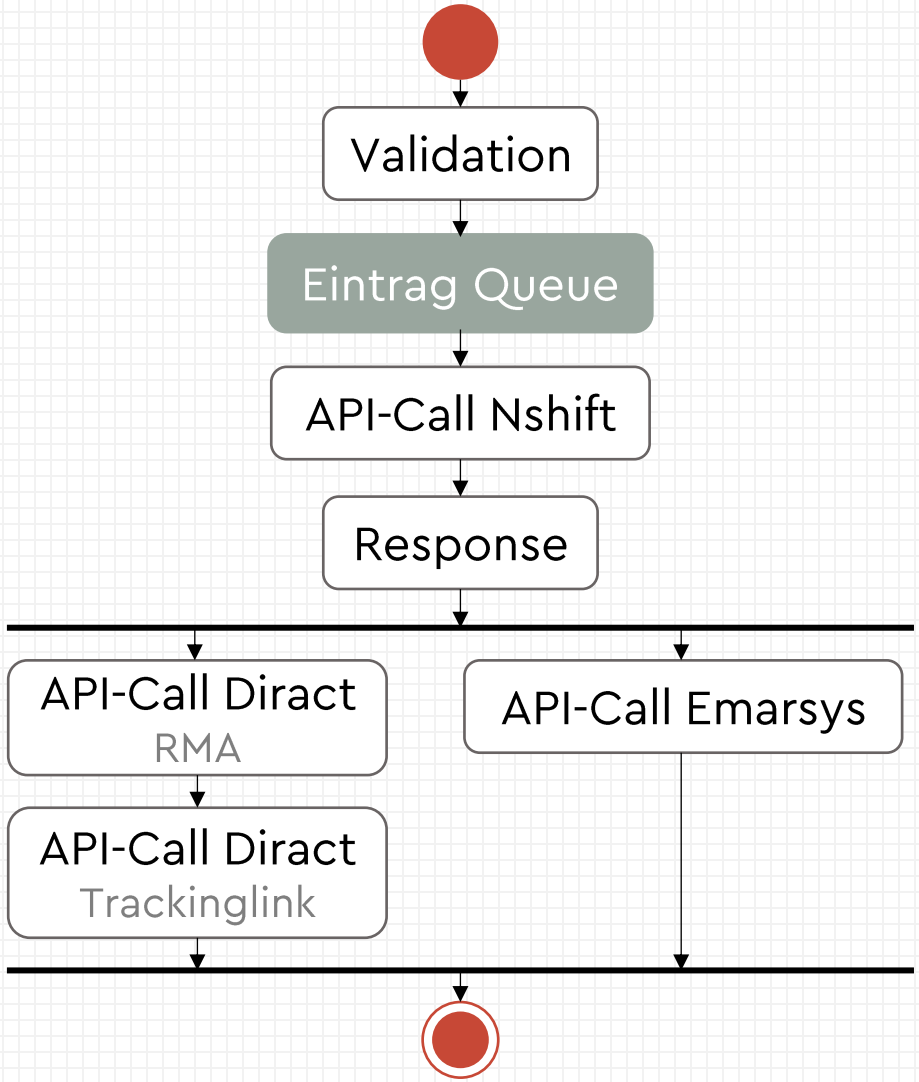




Design

horze

```
6   export const schema = new mongoose.Schema(
7       {
8           _id: { type: mongoose.Schema.Types.ObjectId, required: true, auto: true },
9           rmaNo: { type: String, index: -1 },
10          rmaInProcess: { type: Boolean },
11          isHandled: { type: Boolean },
12          timestampRegistrationCompleted: { type: Date },
13          statusCode: { type: Number, index: -1 },
14          statusMessage: { type: String },
15          orderNo: { type: String },
16          carrierConfigId: { type: String },
17          returnLabel: { type: String },
18          trackingCode: { type: String },
19          trackingLinkUrl: { type: String },
20          sfccPayload: {···
51          },
52          nshiftStatusCode: { type: Number, index: -1 },
53          nshiftStatusMessage: { type: String },
54          nshiftTimestamp: { type: Date },
55          diractRmaStatusCode: { type: Number, index: -1 },
56          diractRmaStatusMessage: { type: String },
57          diractRmaTimestamp: { type: Date },
58          diractRmaPackageStatusCode: { type: Number, index: -1 },
59          diractRmaPackageStatusMessage: { type: String },
60          diractRmaPackageTimestamp: { type: Date },
61          emarsysStatusCode: { type: Number, index: -1 },
62          emarsysStatusMessage: { type: String },
63          emarsysTimestamp: { type: Date },
64          timestampCreated: { type: Date, auto: true, index: -1 },
65          timestampModified: { type: Date },
```



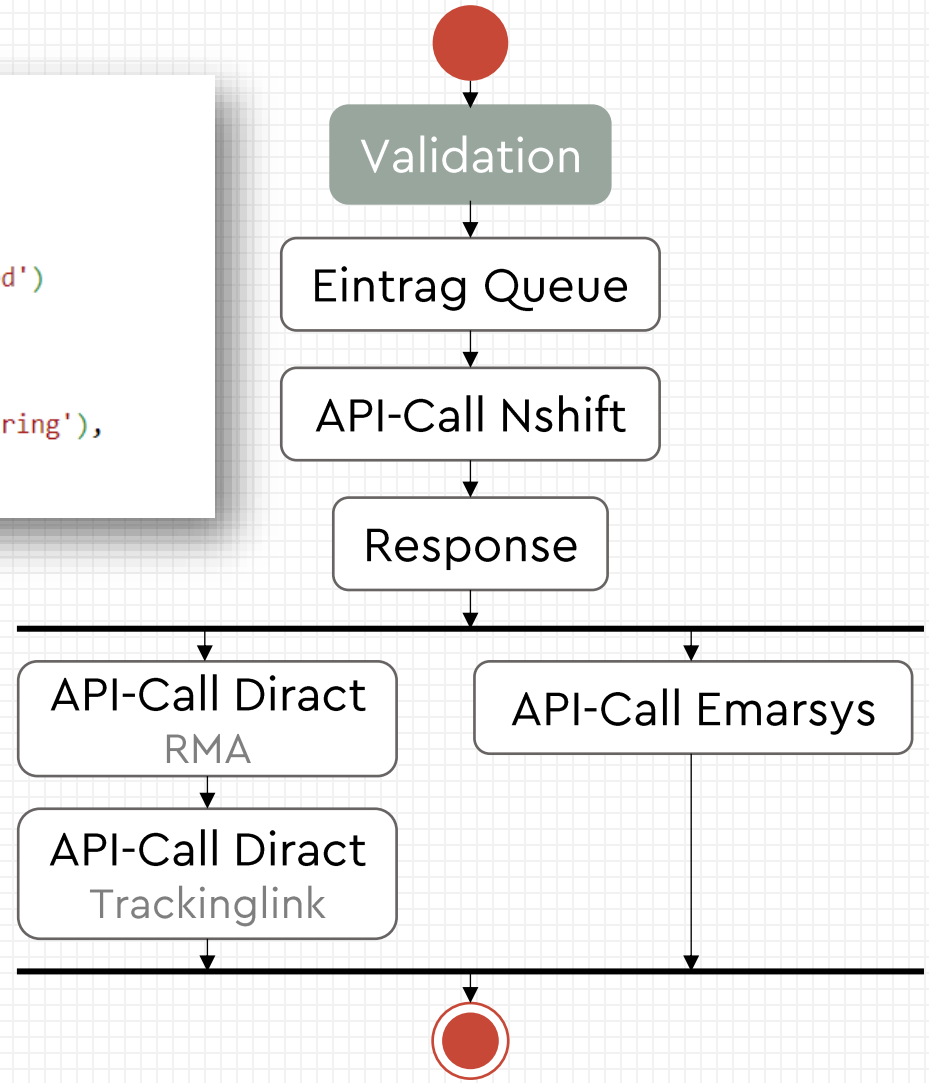
Design

horze

```
export const schema = new mongoose
    {
        _id: { type: mongoose.Sche
        rmaNo: { type: String, inde
        rmaInProcess: { type: Bo
        isHandled: { type: Boolean
        timestampRegistrationCompl
        statusCode: { type: Number
        statusMessage: { type: Str
        orderNo: { type: String },
        carrierConfigId: { type: String },
        returnLabel: { type: String },
        trackingCode: { type: String },
        trackingLinkUrl: { type: String },
        sfccPayload: {···
        },
        nshiftStatusCode: { type: Number, index: -1 },
        nshiftStatusMessage: { type: String },
        nshiftTimestamp: { type: Date },
        diractRmaStatusCode: { type: Number, index: -1 },
        diractRmaStatusMessage: { type: String },
        diractRmaTimestamp: { type: Date },
        diractRmaPackageStatusCode: { type: Number, index: -1 },
        diractRmaPackageStatusMessage: { type: String },
        diractRmaPackageTimestamp: { type: Date },
        emarsysStatusCode: { type: Number, index: -1 },
        emarsysStatusMessage: { type: String },
        emarsysTimestamp: { type: Date },
        timestampCreated: { type: Date, auto: true, index: -1 },
        timestampModified: { type: Date },
```

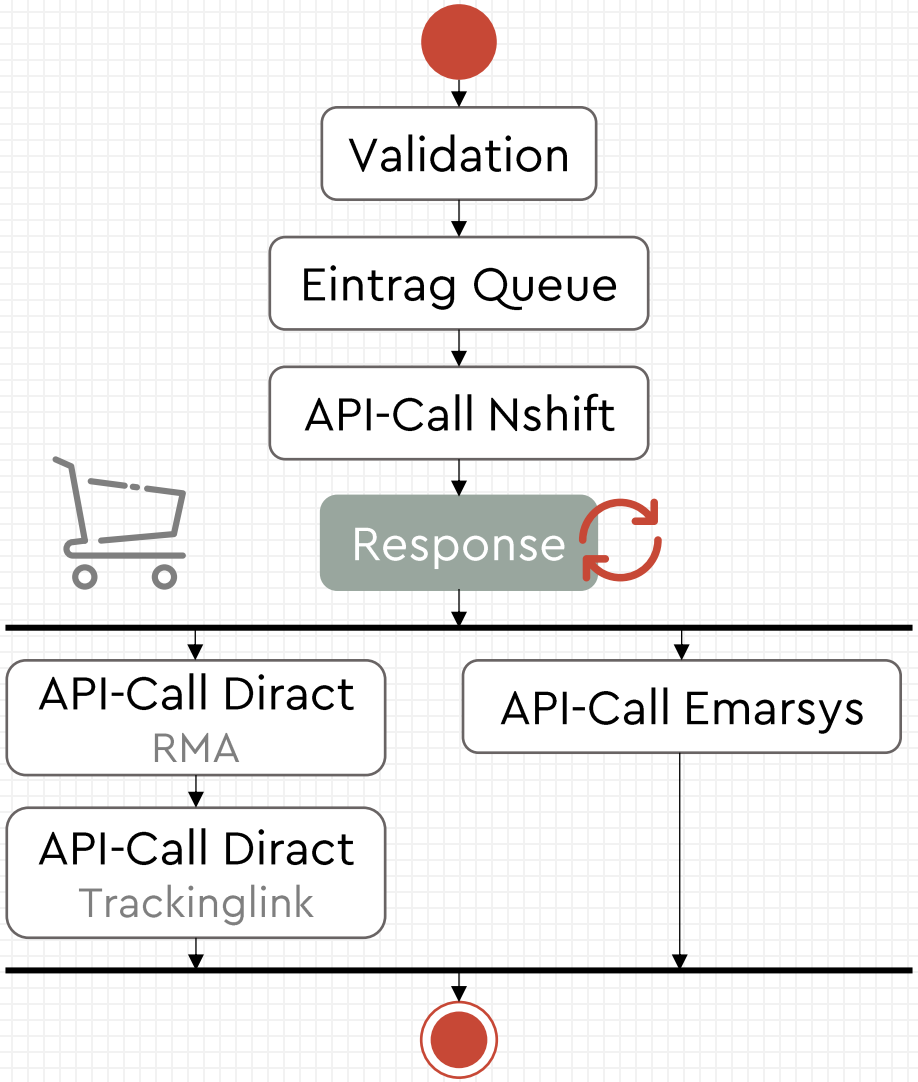
```
import { body } from 'express-validator';

export const queueRmaValidator = [
    body('orderNo')
        .exists()
        .withMessage('Parameter is required')
        .notEmpty()
        .withMessage('Can not be empty')
        .isString()
        .withMessage('orderNo must be a string'),
        ...
];
```



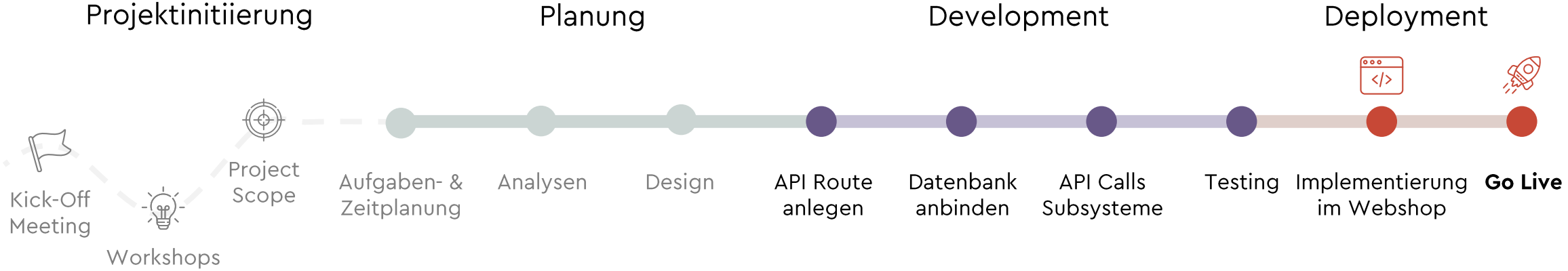
Design

horze



Roadmap

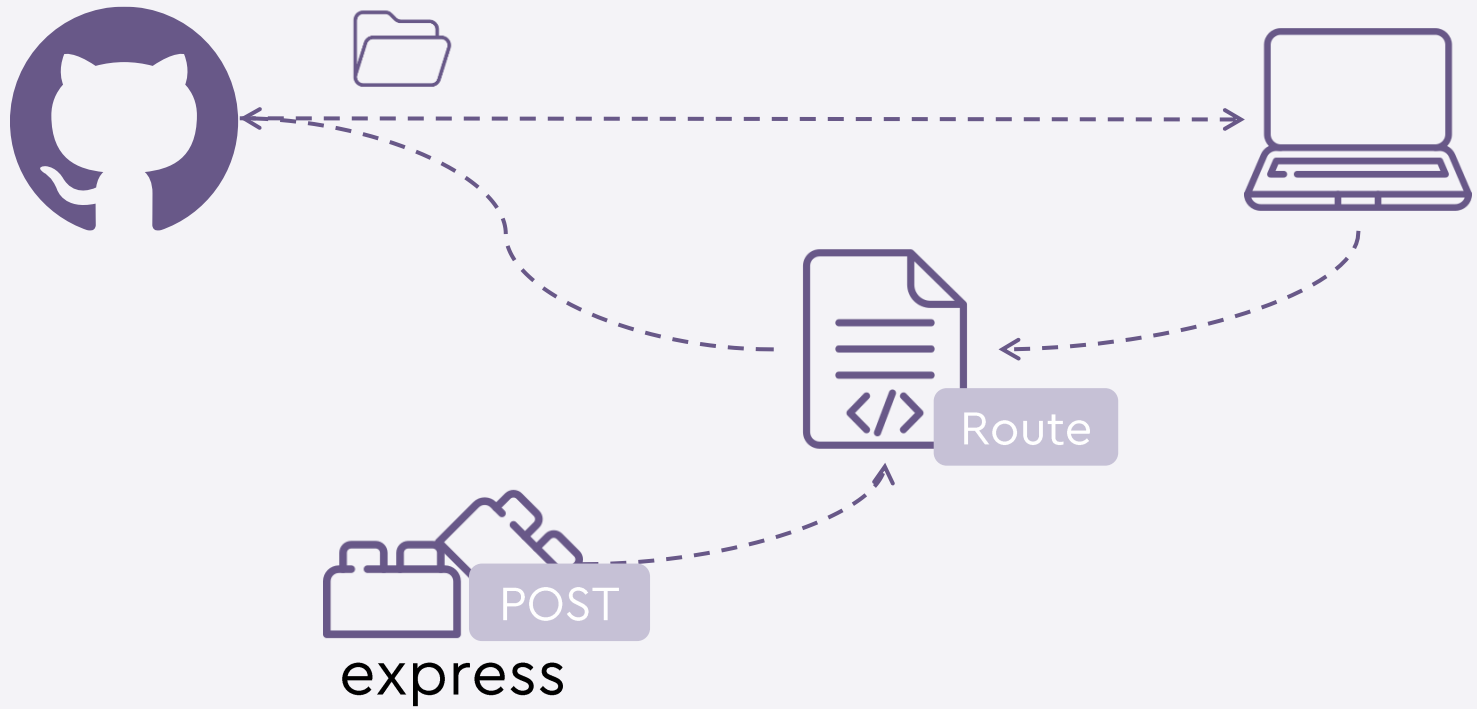
horze



Development

horze

1 API Route anlegen



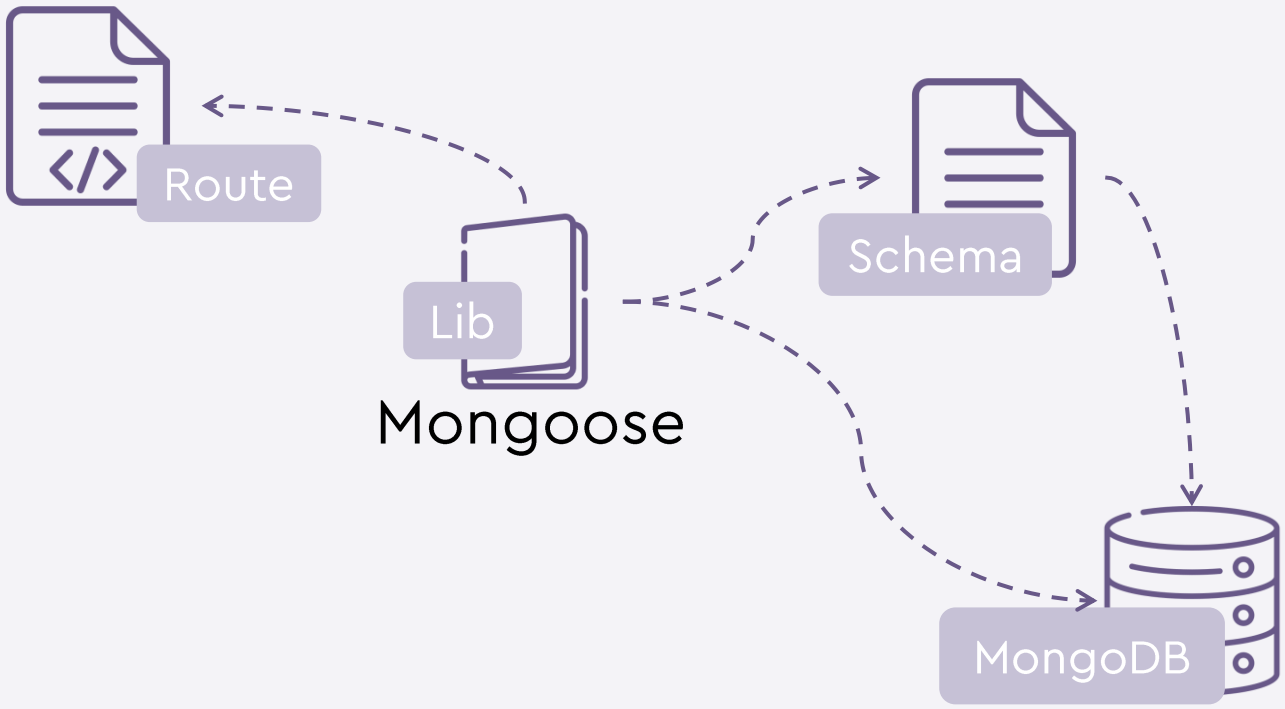
API URL

HTTP Methode

Development

horze

2 Datenbank anbinden

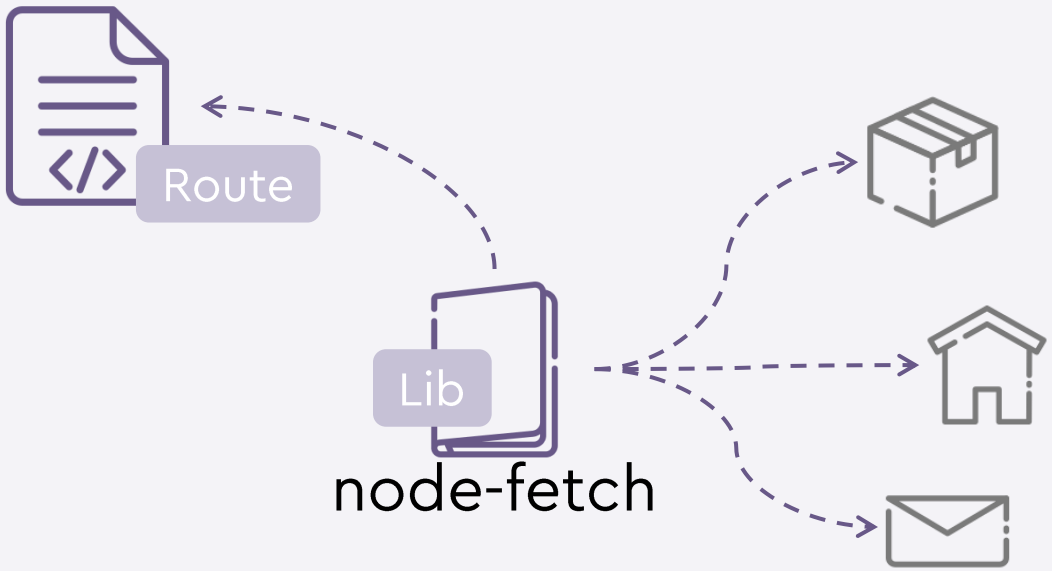


- Connection
- Schema
- Create document
- Update document
- Query Collection

Development

horze

3 API Calls Subsysteme

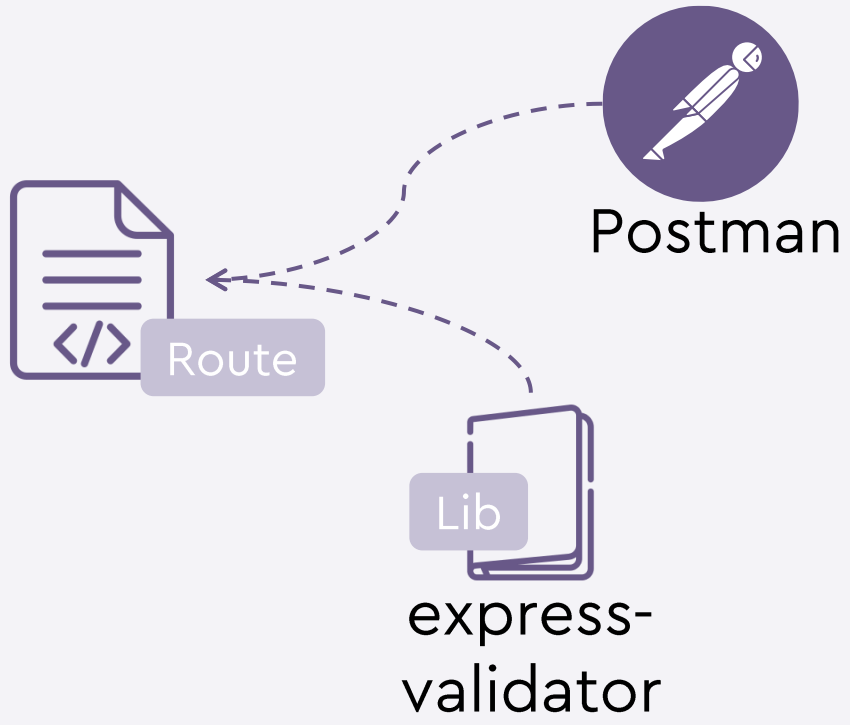


- API-Requests senden
- Rückmeldung verarbeiten
- Rückmeldung an Webshop
- Parallele Verarbeitung

Development

horze

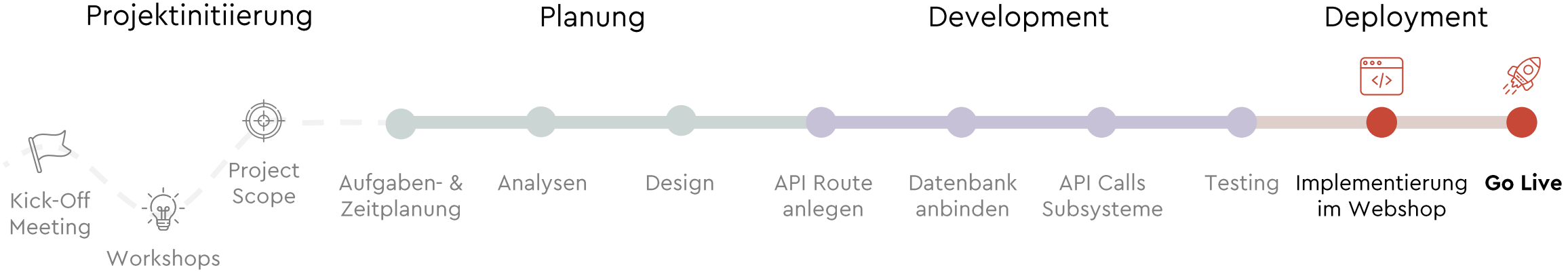
4 Testing



- Definieren Validierungsfelder
- Manuelles Testing
- Bereitstellung auf STG
- E2E-Tests

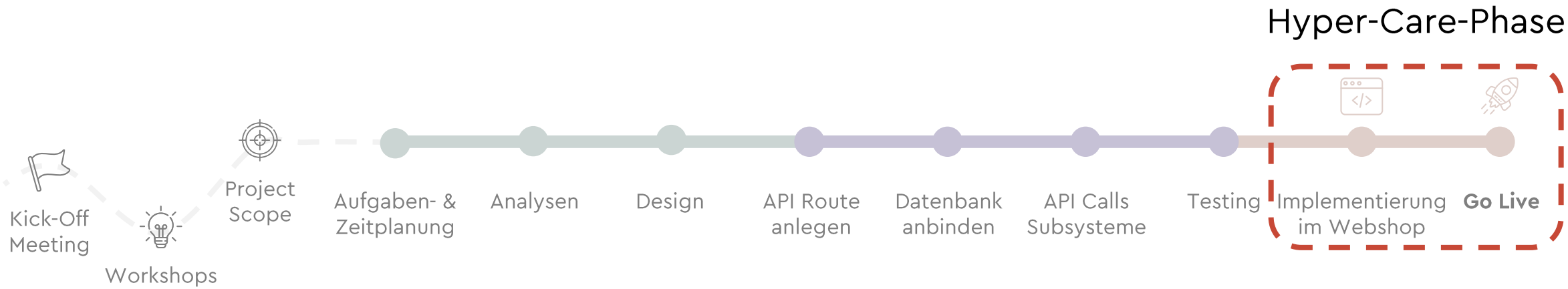
Roadmap

horze



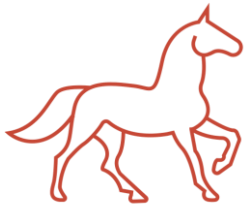
Roadmap

horze



Abschluss

horze



Lessons Learned

Ausblick

horze

Fachgespräch

Abschlussprüfung