duogradus.

Abschlusspräsentation

19.06.2024

Software Engineering – 4. Semester

Agenda

- Demo
- Projektplanung
  - Was war geplant?
  - Was haben wir umgesetzt?
- Entwicklung
- Dokumentation
- DevOps
- Lessons Learned
- Fazit

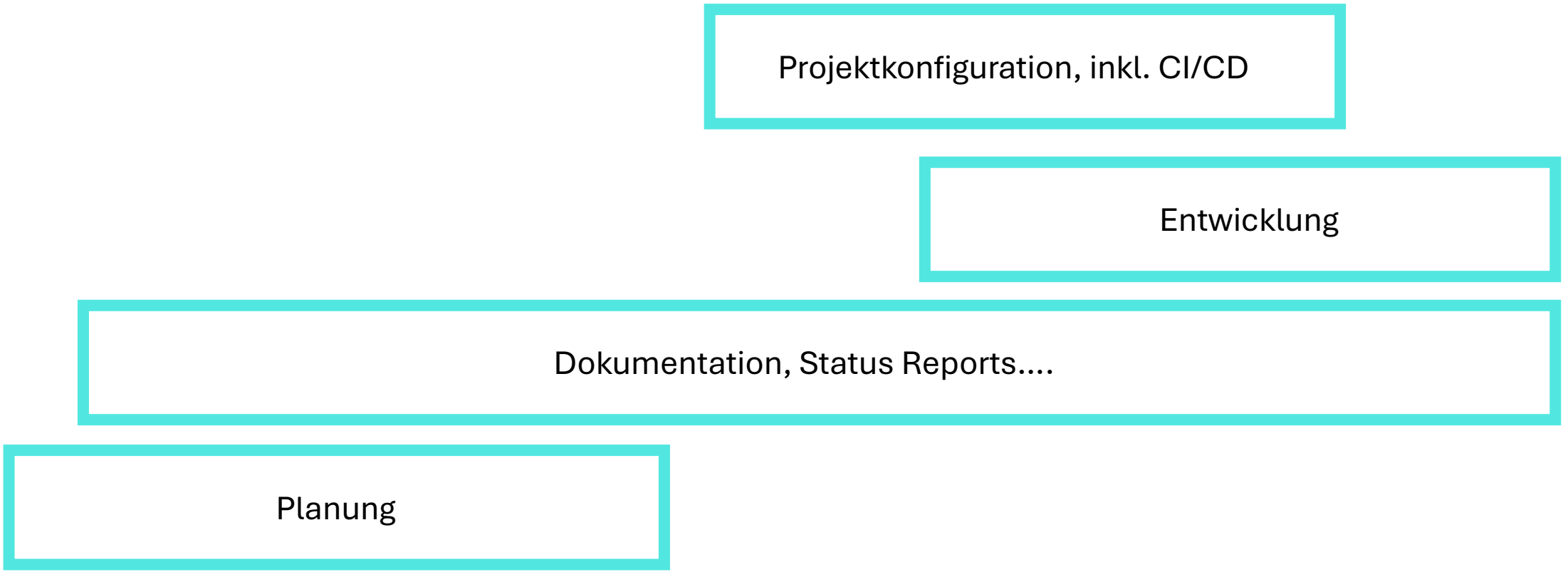
Demo

duogradus.

Projektplanung

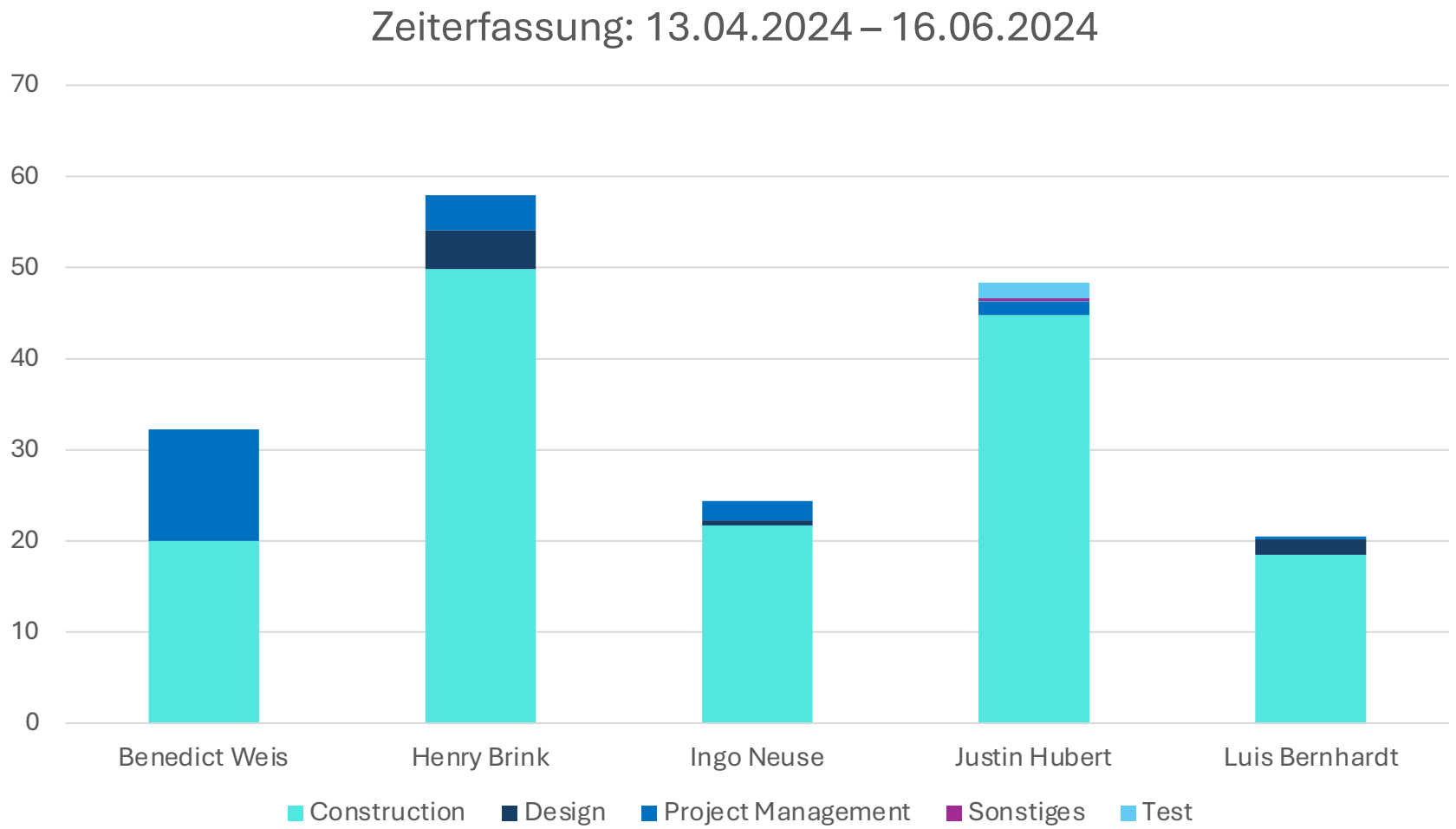
duogradus.

Zeitlicher Projektablauf



duogradus.

Zeiterfassung



duogradus.

Zeiterfassung - Technologie

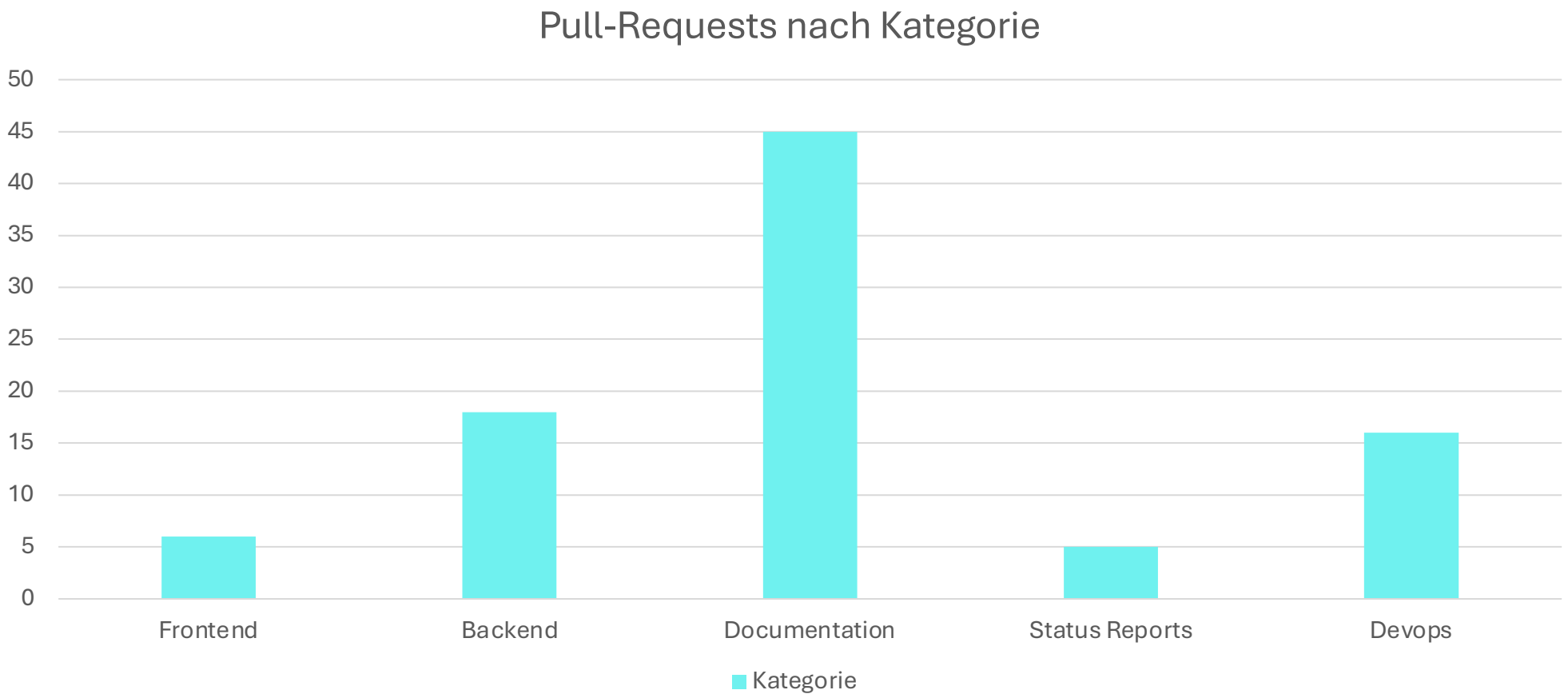
Kimai

Kimai (Time tracking tool)
- Einfaches Timetracking
- Keine komplexe Funktionalität
- Keine Integrationen mit Drittanbietern

CSV

GitHub Action

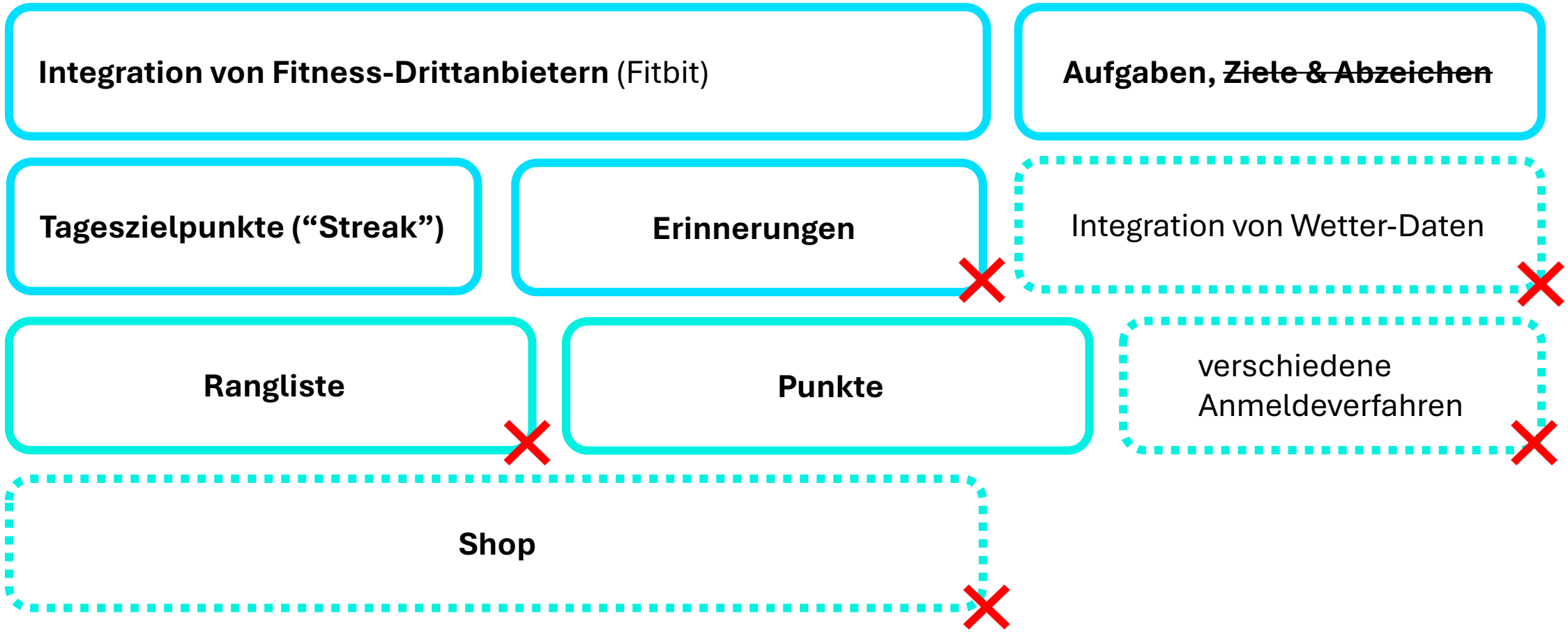
Pull-Requests



duogradus.

Entwicklung

Anforderungen - Übersicht



duogradus.

Technologie - Backend

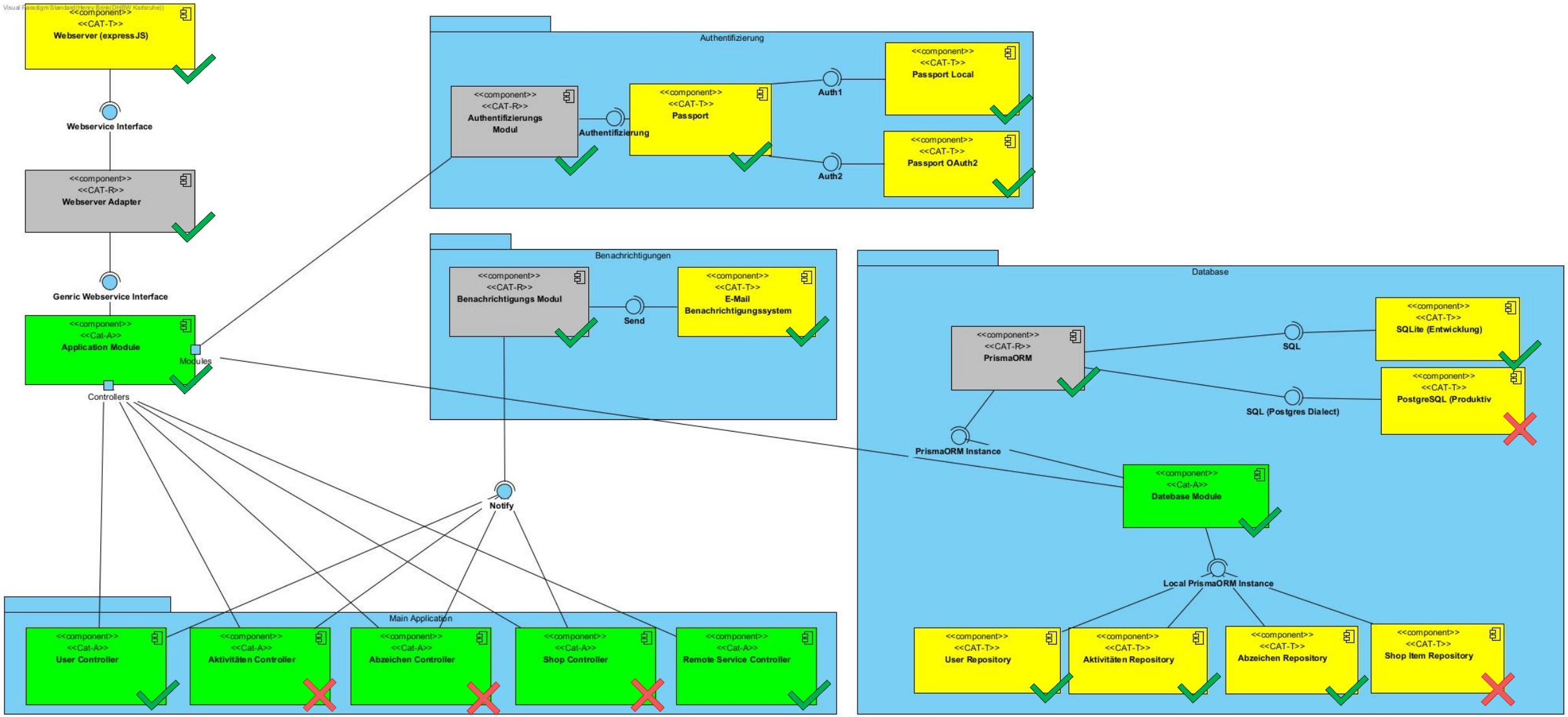
NestJS (Backend Framework)
- MVC-Architektur möglich
- TypeScript
- REST & GraphQL Unterstützung

Prisma ORM

TypeScript

Jest

NodeJS


Webserver (expressJS)
Webservice Interface
Webserver Adapter
Genric Webservice Interface
Application Module
Modules
Controllers
Authentifizierung
Authentifizierungs Modul
Authentifizierung
Passport
Auth1
Auth2
Passport Local
Passport OAuth2
Benachrichtigungen
Benachrichtigungs Modul
Send
E-Mail Benachrichtigungssystem
Notify
Database
PrismaORM
SQL
SQL (Postgres Dialect)
SQLite (Entwicklung)
PostgreSQL (Produktiv
PrismaORM Instance
Datebase Module
Local PrismaORM Instance
User Repository
Aktivitäten Repository
Abzeichen Repository
Shop Item Repository
Main Application
User Controller
Aktivitäten Controller
Abzeichen Controller
Shop Controller
Remote Service Controller

duogradus.

Struktur - Backend

- prisma
- src
  - api
  - app
    - <module-x>
  - auth
  - config
  - db
  - integration
- test

duogradus.

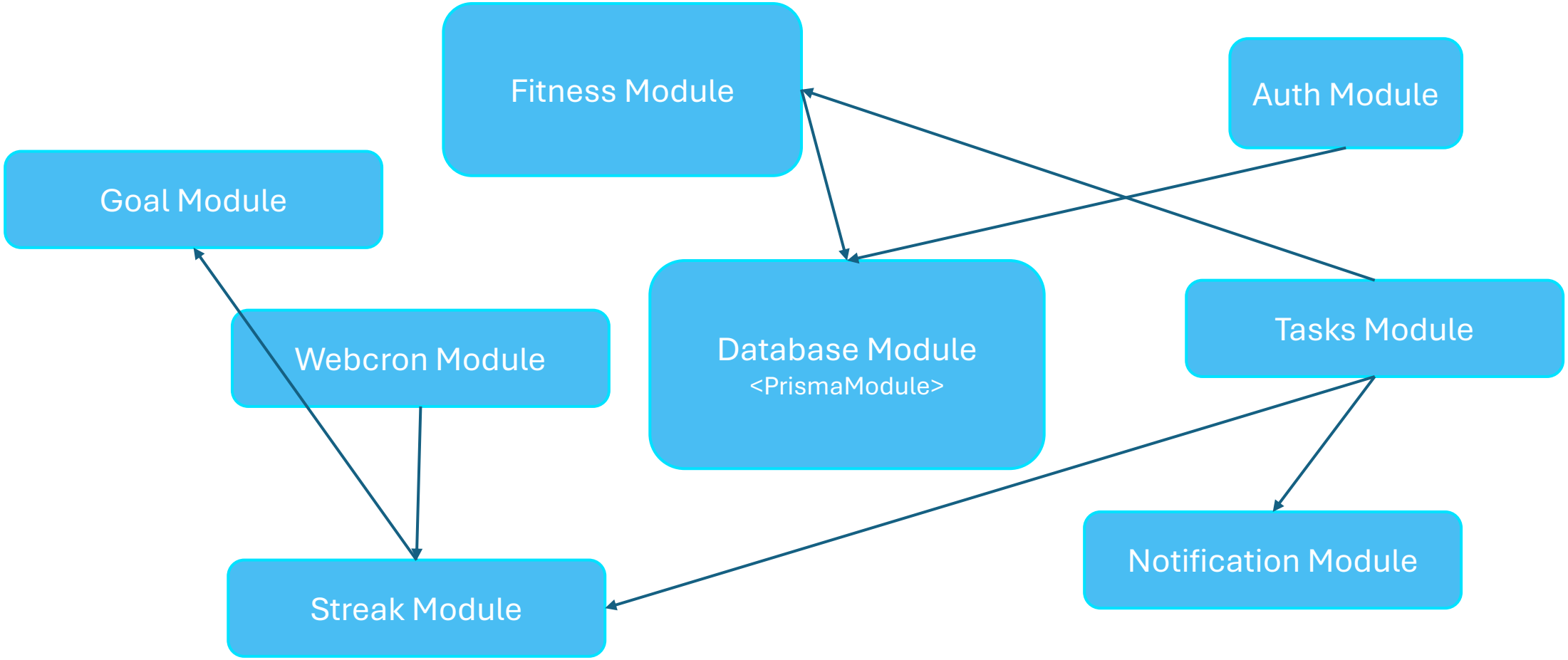
Struktur - Backend

- Neue Module werden unter “app/<modul>” abgelegt
  - Jedes Modul hat mindestens
    - Eine Modul-Klasse
    - Einen Service / äquivalenten Provider
  - und optional
    - Einen Controller
    - Tests

Funktionen - Backend

- Module
  - **Auth-Modul**: Basic Auth
  - **Nutzer-Modul**: Nutzerdaten abfragen / ändern
  - **Integrations-Modul**: Abfrage von Fitness-Daten (z.B. über Fitbit)
  - **Streak-Modul**: Punkteverwaltung (pro Tag)
  - **Goal-Modul**: Fitness-Ziele laden & cachen
  - **Datenbank-Modul**: Repositories für Datenzugriff
  - **Benachrichtigungs-Modul**: E-Mails versenden

Modul – Zusammenhang



duogradus.

Software Engineering – Ansätze & Ideen

- Architektur aus Komponenten
  - Idee: Jede isolierte Funktionalität wird über ein eigenes Modul bereitgestellt
  - Im Modul werden die Verantwortungen idealerweise gebündelt:
    - Controller sind immer für die Kommunikation mit dem Nutzer verantwortlich
    - Ein Service stellt die eigentliche Funktionalität bereit
    - Abhängigkeiten zu anderen Modulen erfolgen nur so, wie sie das Framework vorsieht
- Änderungen im Code erfolgen nach dem bekannten Schema (“Req” > “Des” > “Con” > “Test”)

Anforderungen - Übersicht

Integration von Fitness-Drittanbietern (Fitbit)

Aufgaben, ~~Ziele &~~ ~~Abzeichen~~

Authentifizierung

Tageszielpunkte (“Streak”)

Erinnerungen

Integration von Wetter-Daten

Nutzerverwaltung

Rangliste

Punkte

verschiedene Anmeldeverfahren

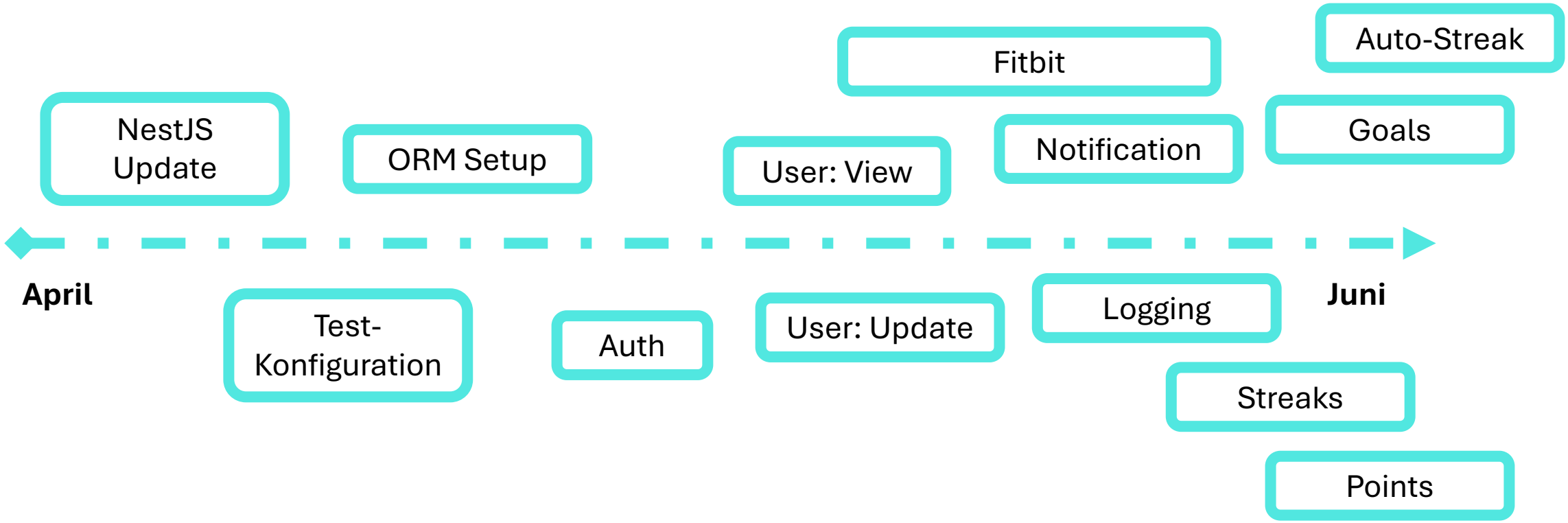
Shop

Tests

Datenbankzugriff, Migrationen etc.

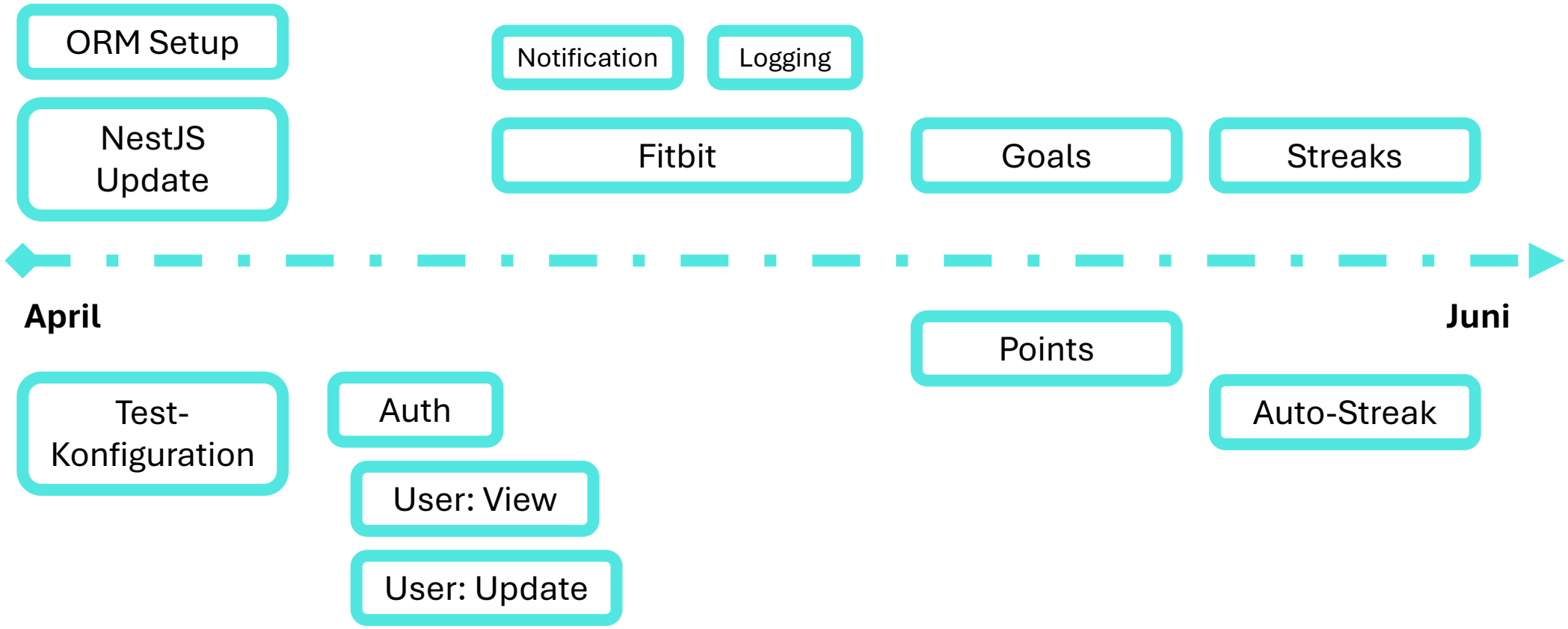
duogradus.

Ablauf – Backend

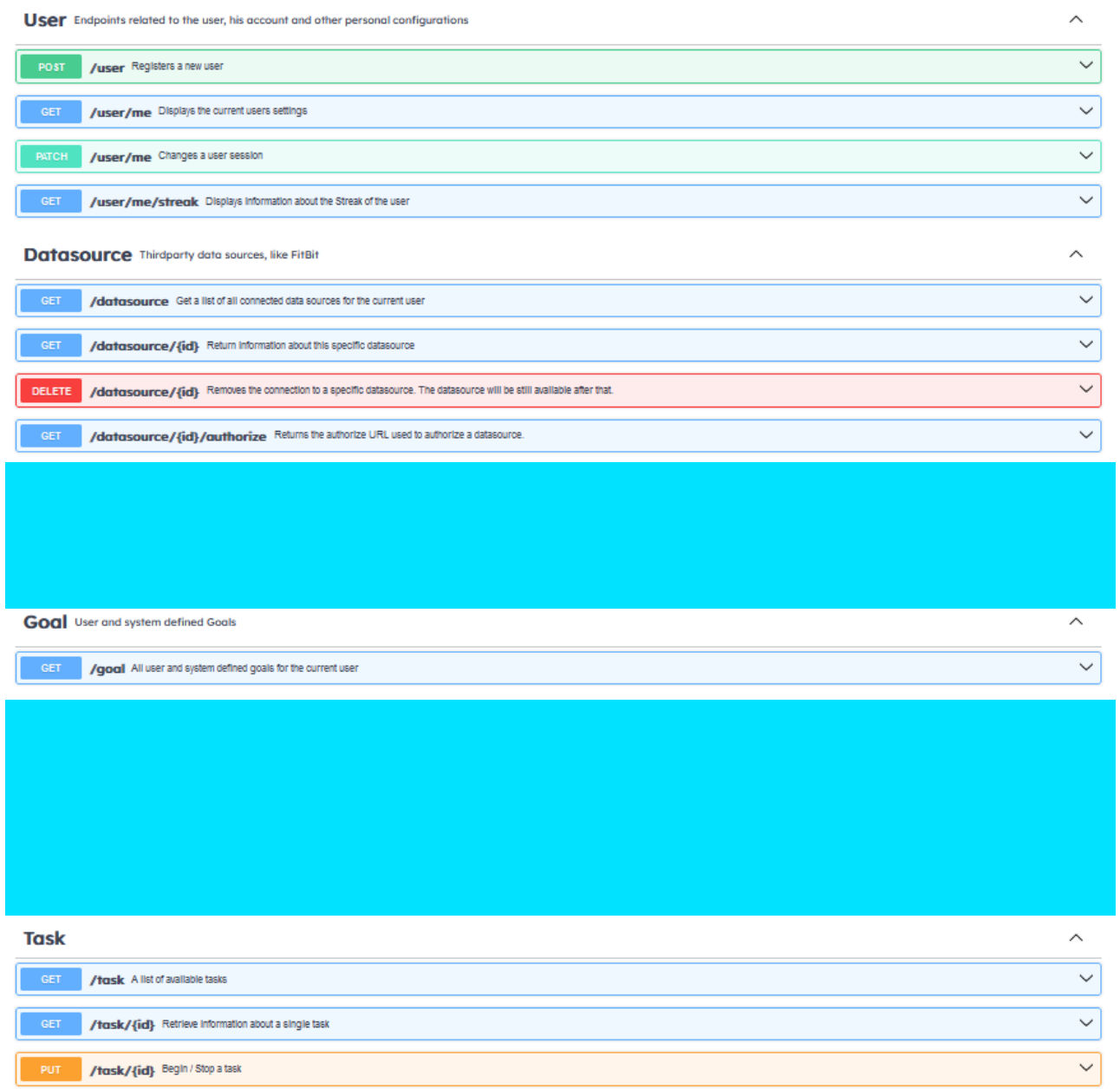


duogradus.

Idealer Ablauf – Backend



duogradus.


User
Endpoints related to the user, his account and other personal configurations
POST /user Registers a new user
GET /user/me Displays the current users settings
PATCH /user/me Changes a user session
GET /user/me/streak Displays information about the Streak of the user
Datasource
Thirdparty data sources, like FitBit
GET /datasource Get a list of all connected data sources for the current user
GET /datasource/{id} Return information about this specific datasource
DELETE /datasource/{id} Removes the connection to a specific datasource. The datasource will be still available after that.
GET /datasource/{id}/authorize Returns the authorize URL used to authorize a datasource.
Goal
User and system defined Goals
GET /goal All user and system defined goals for the current user
Task
GET /task A list of available tasks
GET /task/{id} Retrieve information about a single task
PUT /task/{id} Begin / Stop a task

duogradus.

Technologie - Frontend

Angular(Frontend Framework)

- Komponentenbasiertes UI
- TypeScript
- Große Entwickler-Community

Jasmin

TypeScript

Struktur - Frontend

- src
  - app
    - components
      - atoms
      - organisms
      - pages
    - pipes
    - services
  - assets

Arbeitsweise - Frontend

- Für jedes neue Feature ein Branch erstellen
- Gewünschte Änderungen vornehmen
- Änderungen committen und pushen
- Pull Request von zusätzlicher Person überprüfen lassen

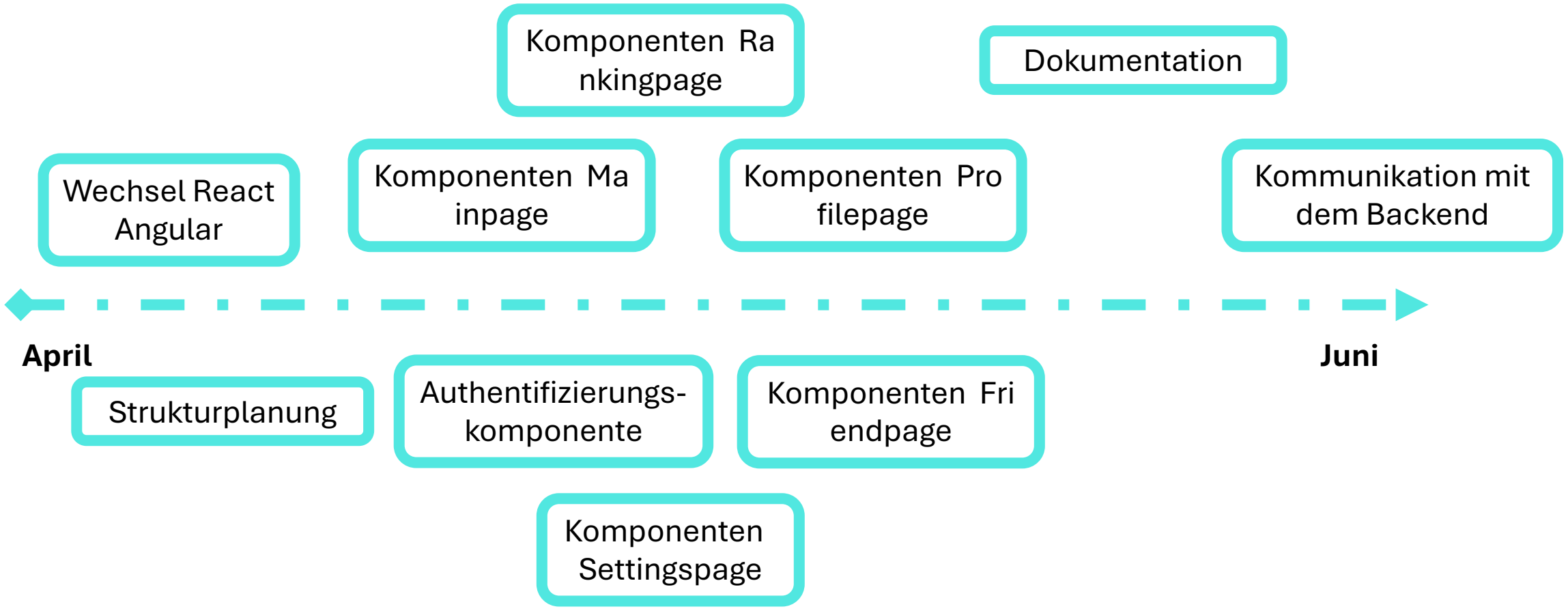
Funktionen - Frontend

- Services
  - **Auth-Service**: User-Authentifizierung und Bereitstellung des Tokens
  - **Event-Service**: Datenaustausch zwischen Komponenten
  - **User-Service**: Bereitstellung der User-Informationen
  - **Mainpage-Service**: Bereitstellung aller Tasks und der Streak des Users
  - **Loader-Service:** Ein- und Ausblenden des Loaders

- Pipes
  - **Timer-Pipe (optional)**: Mapping von Millisekunden in Uhrzeitform
  - **Userfilter-Pipe**: Suche der User in der Freundesliste

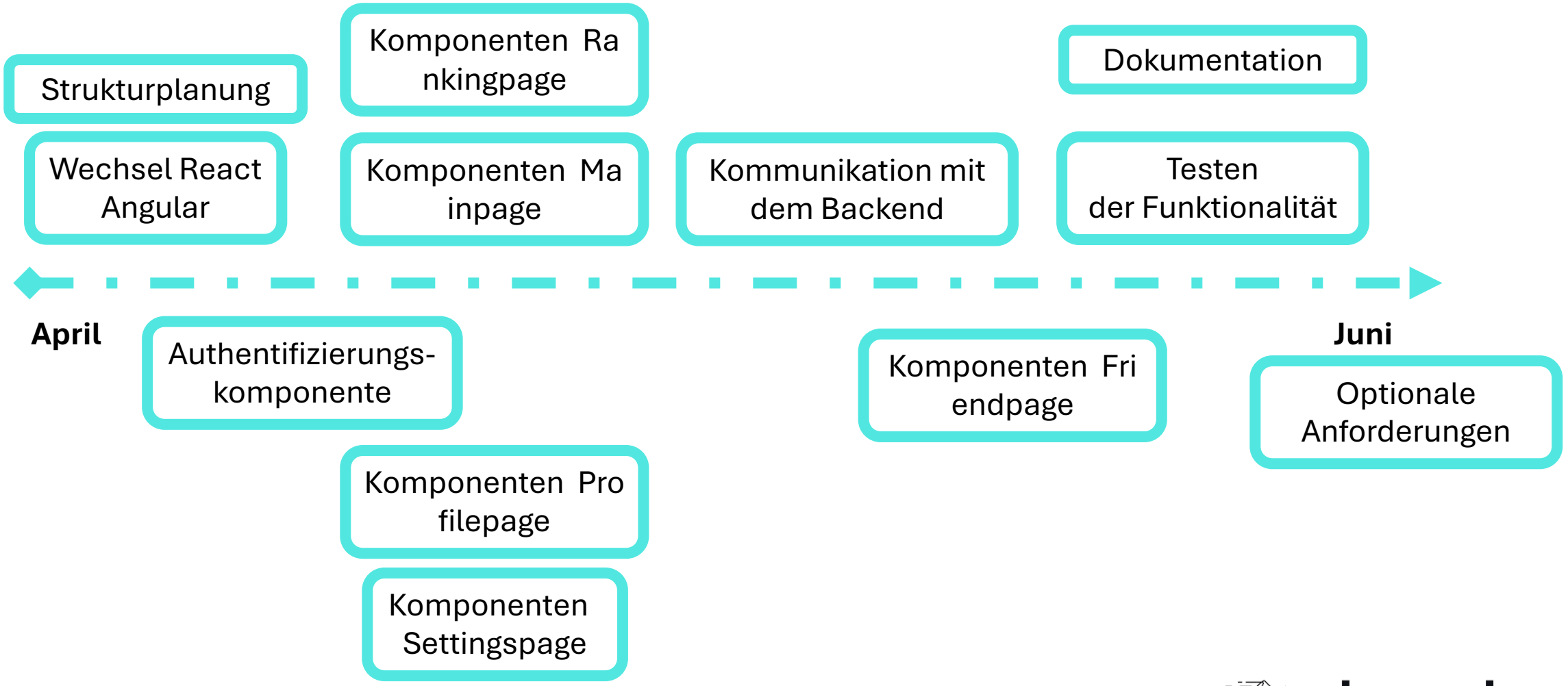
- Animationen
  - **Transition-Animationen:** Enthält den Code für die Animationen

Ablauf – Frontend



duogradus.

Idealer Ablauf – Frontend



duogradus.

DevOps

duogradus.

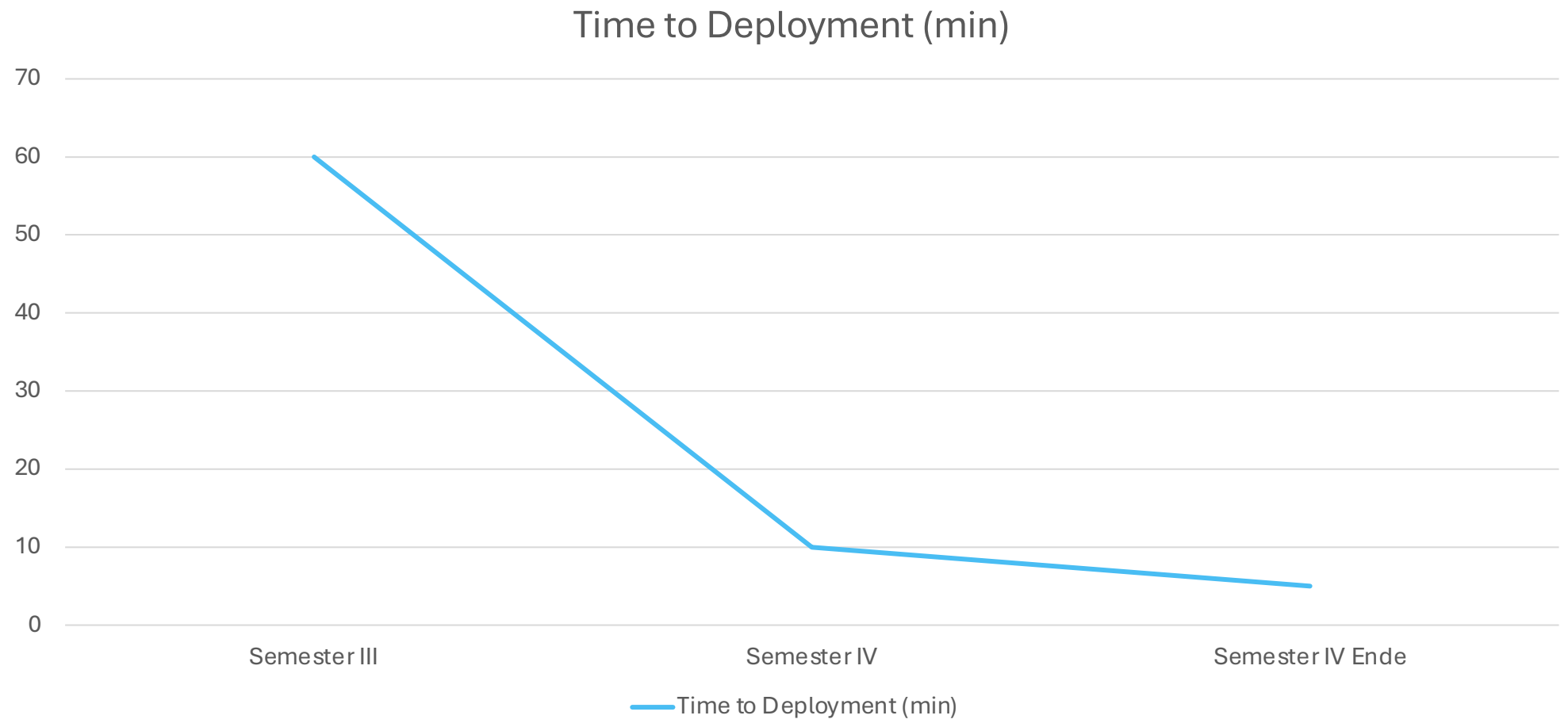
Technologie - DevOps

GitHub Actions
- Self Hosted Runner
- Eigene Skripte / Actions

Bash

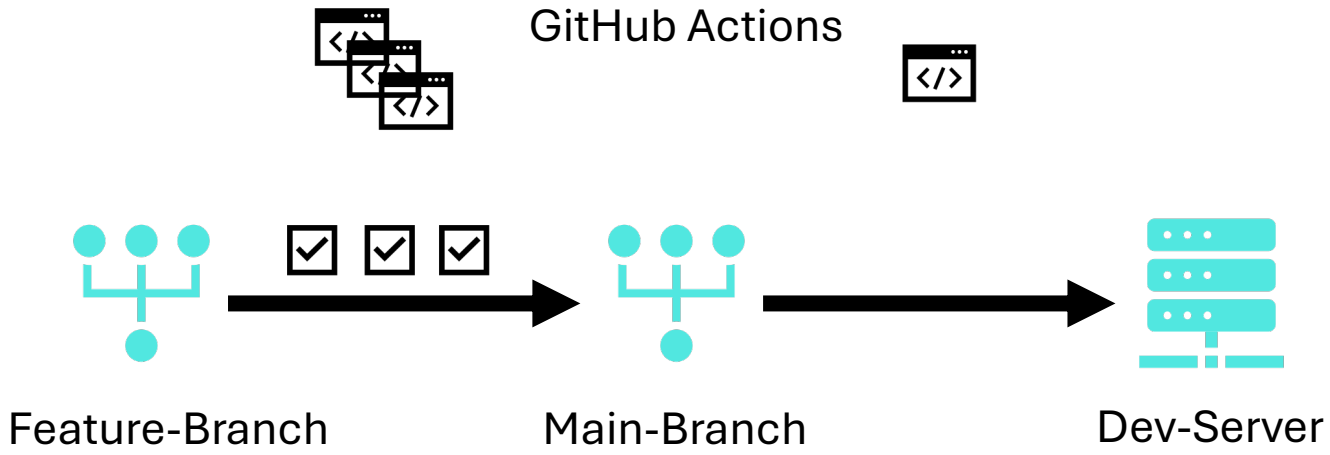
Python

Time to Deployment – Warum DevOps?



duogradus.

DevOps – Implementierung



DevOps – Implementierung

- **Main-Branch bleibt immer stabil**
- Feature-Branches warden nur nach Checks gemerged
- Feature-Branches warden nur mit mindestens einem Approved Review gemerged
- Main-Branch wird nach jedem Merge auf den Dev-Server deployed

- GitHub Actions realisieren in unserem Fall diesen Prozess

Dokumentation

duogradus.

Technologie - Dokumentation

VitePress

VitePress (Static Site Generator)
- Einfache Dateiorganisation
- Zentrale Konfigurationsdatei
- Vue.js Unterstützung

Markdown

TypeScript

Vue.js

NodeJS

Struktur - Dokumentation

- .vitepress
  - index.mts
- development
  - backend
  - frontend
- project
  - timetracking
- public
- reports
- index.md

duogradus.

Struktur - Dokumentation

Neuen Eintrag anlegen

1. Neue Markdown Datei anlegen

2. Datei der Vitepress Konfiguration hinzufügen

Vorteile - Dokumentation

- Alternative zu unserem Ansatz ist das **GitHub Wiki**
- Durch das Ablegen der Dokumentation direkt im Repository ergeben sich aber viele Vorteile
  1. Dokumentation profitiert von CI/CD-Workflows wie Pull Request Checks
  2. Dokumentationseinträge müssen automatisch reviewed werden
  3. Dokumentation wird mit simplem DevOps Workflow deployed
  4. Hürde zum Anlegen von Dokumentation ist gesenkt, da man die Dokumentation automatisch offen hat wenn man das Repo offen hat

Lessons Learned

duogradus.

Lessons Learned - Übersicht

Projektplanung

Teamstruktur

Entwicklung

Projektarbeit

Projektplanung

- High-Level Ansatz: Anforderungen => Arbeitspakete ableiten hätte gut funktioniert, haben wir aber teils nicht umgesetzt
- Feste anfängliche Planung hat gefehlt
- Genauere Meilensteine mit Terminen und Zielen definieren
  - Teams können Prioritäten setzen
  - Erhöht die Motivation
- Selbst die Dinge, die geplant waren, waren nicht allen bekannt, dies geht mit allgemein fehlender Kommunikation einher

Teamstruktur

- Zur Erinnerung: 1 PM, 2 Backend, 2 Frontend
- Keine so strikte Trennung zwischen Frontend und Backend nötig
- In einem so kleinen Team sollte jeder sich im Repository auskennen, egal ob Frontend oder Backend

Projektarbeit

- Motivation zum Arbeiten kam nur in Schüben, mit besserer Planung hätte dies Vermieden werden können
- Zeiterfassungstool mit GitHub Integration hätte massiv geholfen

Entwicklung - Frontend

- Hoher Zeitaufwand, wenn jede Komponente von Grund auf selbst erstellt werden muss

Lösung:

  - Zeitersparnis durch einbinden von bspw. MUI

- Granularität des Atomic-Design ist nicht für jede Projektgröße geeignet

Lösung:

  - Die vorgegebenen 5 Ebenen auf 3 Ebenen vereinfachen

Entwicklung - Frontend

- Extra Entwicklungs-Branches: Zusätzliche Ebene machte es noch schwieriger Fortschritt zu tracken und war unnötig.

Lösung:

  - Weglassen der zusätzlichen Ebene

Entwicklung - Backend

- Detailliertere Planung in der Planungsphase, besonders mit strikten Deadlines hätte die Entwicklung vermutlich beschleunigt
- Test-Coverage und ähnliches sind sinnvolle KPIs, gerade zu Beginn eines Projekts verlangsamen sie die Entwicklungsgeschwindigkeit allerdings massiv.
- Neue Frameworks / Bibliotheken führen oft zu vielen versteckten Problemen und hohem Lernaufwand

Fazit

Fazit - Was lief gut?

- Dokumentation
- DevOps / CI/CD Setup
- Pull Request Ansatz
- Repository Organisation
- Zeiterfassung am Ende

Fazit - Was lief nicht gut?

- Projektplanung war teils nicht so tief, wie notwendig
  - Anforderungen wurden geplant und spezifiziert
  - Es wurden zu spät / zu wenig Issues aus den Anforderungen abgeleitet
  - Eine detaillierte Zeitplanung fehlte
- Entwicklung ging zunächst schleppend voran
- Das Teamgefühl war noch Ausbaufähig
  - Aufteilung 2x2 Entwicklungsteams war nicht fördernd
  - Es gab zu wenig Team-Meetings
  - Keine Regelmäßigkeit

Fazit - Zusammenfassung

DevOps & Dokumentation

Reviews & QA

Entwicklung

Projektplanung

Fazit – Wie würden wir es heute angehen?

- Wenn man sich in der Gruppe nicht gut kennt: Gleich in der ersten Woche **gemeinsames Event** um Teamgefühl zu stärken
- Ein **fester wöchentlicher Termin** (ca. **1h**), wo das Projekt besprochen wird und gemeinsam Entwickelt wird
- **Wechselnde Projektleiter Rolle** um Personen zu entlasten und jeden zu integrieren
- **Wechselnde Teamaufgaben** für jedes Mitglied (am Anfang Fokus auf Backend, dann Fokus auf das Frontend)

Fazit – Wie würden wir es heute angehen?

- Am Anfang einmal **festen High-Level Zeitplan gemeinsam erstellen**
- Einfach mal **einander zuhören / mehr Diskussionen**
- **Am Anfang auf Standards festlegen** (Zeiterfassungstool, Merging Strategie, Pull Request und Issue Templates, etc.)
- **Beibehalten:**
  - DevOps / CI/CD Setup
  - Dokumentationsansatz
  - Strikten Pull Request Review Ansatz

Danke fürs Zuhören!

duogradus.