

- ♦ Gesamte Ordnerstruktur der Webapplikation
- ♦ Eigene REST-Web-APIs
- ♦ Andere REST-Web-APIs
- ♦ GraphQL
- ♦ Swagger-Dokumentation
- Datenbank
- Architektur und externe Bibliotheken & Tools
- ♦ Live-Demonstration der Web-Applikation
- Allgemeine Reflexion

Agenda

Gesamte Ordnerstruktur der Webapplikation

- ✓ ☐ indexlive E:\!CLOAD\FH\4. Seme
 > ☐ .idea
 > ☐ backend
 - > 🗀 database
 - > 🗀 dokumentation
 - > 🗀 extern
 - > 🗀 frontend
 - > praesentationen
 - docker-compose.yml

- frontend → ☐ dist → □ src ≡ .env.local eslintrc.cjs eslintrc.json ② .gitignore Dockerfile <> index.html {} package.json {} package-lock.json M↓ README.md {} tsconfig.json {} tsconfig.node.json vite.config.ts
- ∨ □ assets ∨ □ components BoxHeader.tsx ChartsBox.tsx FlexBetween.tsx ∨ □ scenes ∨ □ charts mindex.tsx Set1.tsx Set2.tsx Set3.tsx ∨ □ navbar mindex.tsx ∨ □ predictions mdex.tsx ∨ □ state rs api.ts s transactionQueries.ts s types.ts App.tsx s expanded-thems.ts index.css main.tsx s theme.ts

rs vite-env.d.ts

∨ □ data Js data.js graphql Js transactionSchema.js ∨ □ models Js kpiModel.js productModel.js us transactionModel.js ∨ □ routes Js kpiRoute.js productRoute.js Js transactionRoute.js ≡ .env ② .gitignore Dockerfile Js index.js {} package.json {} package-lock.json Js swaggerConfig.js

Eigene REST-Web-APIs



Produkte und Transaktionen:

GET, POST, PUT, DELETE

- http://localhost:10081/product/products
- http://localhost:10081/transaction/transaction/transactions



Key Performance Indicators "KPIs":

GET

• http://localhost:10081/kpi/kpis

Andere REST-Web-APIs

♦ Es werden keine öffentlichen REST-Web-APIs verwendet, auch nicht von anderen Teams

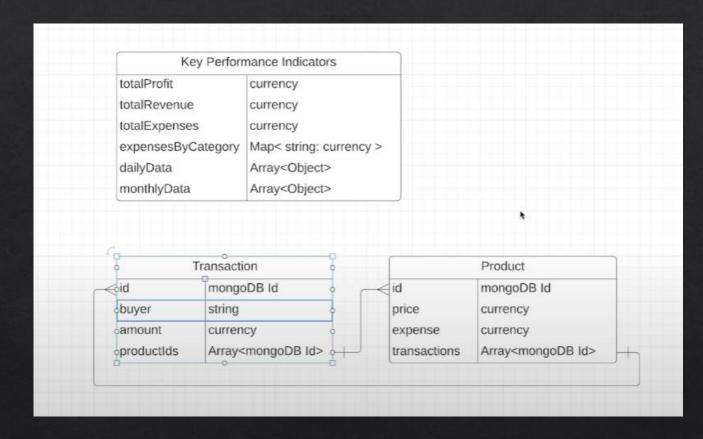
GraphQL

- **⋄** Für Transaktionen:
 - http://localhost:10081/graphql

Swagger-Dokumentation

http://localhost:10081/api-docs/

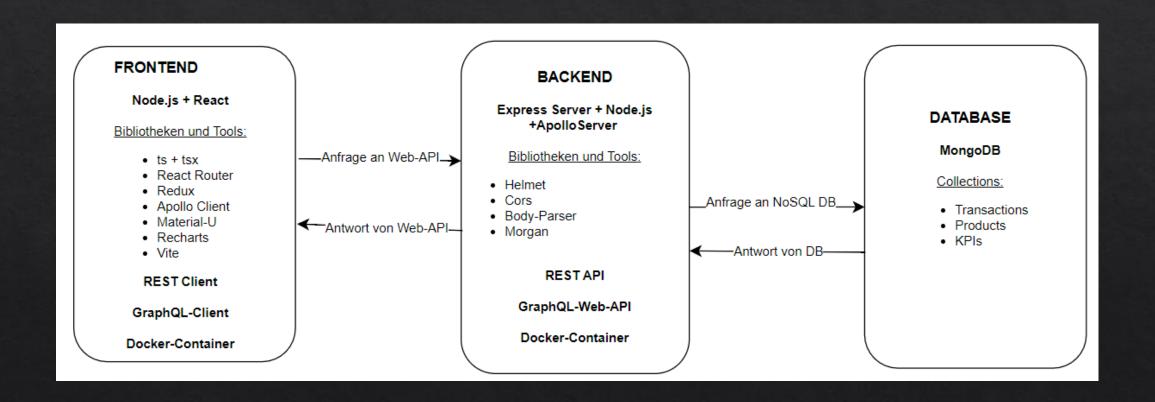
Datenbank



```
const ProductSchema = new Schema(
        price: {
           type: mongoose.Types.Currency,
           currency: "USD",
           type: mongoose.Types.Currency,
       transactions: [
               type: mongoose.Schema.Types.ObjectId,
   { timestamps: true, toJSON: { getters: true } }
```

https://cloud.mongodb.com/v2/66654aebedae674a11aacd3a#/metrics/replicaSet/66654c13b42d544140a745fb/explorer/test

Architektur und externe Bibliotheken & Tools



LiveDemonstration der WebApplikation

♦ http://localhost:10084/

Allgemeine Reflexion

The State S

- Darstellung von Daten in Diagrammen
- ♦ Docker lauffähig in der Projektverwaltung

Fehlende Features:

- ♦ Predictions
- ♦ User-Authentifizierung.

