Projektarbeit

For the attention of

MAS SE

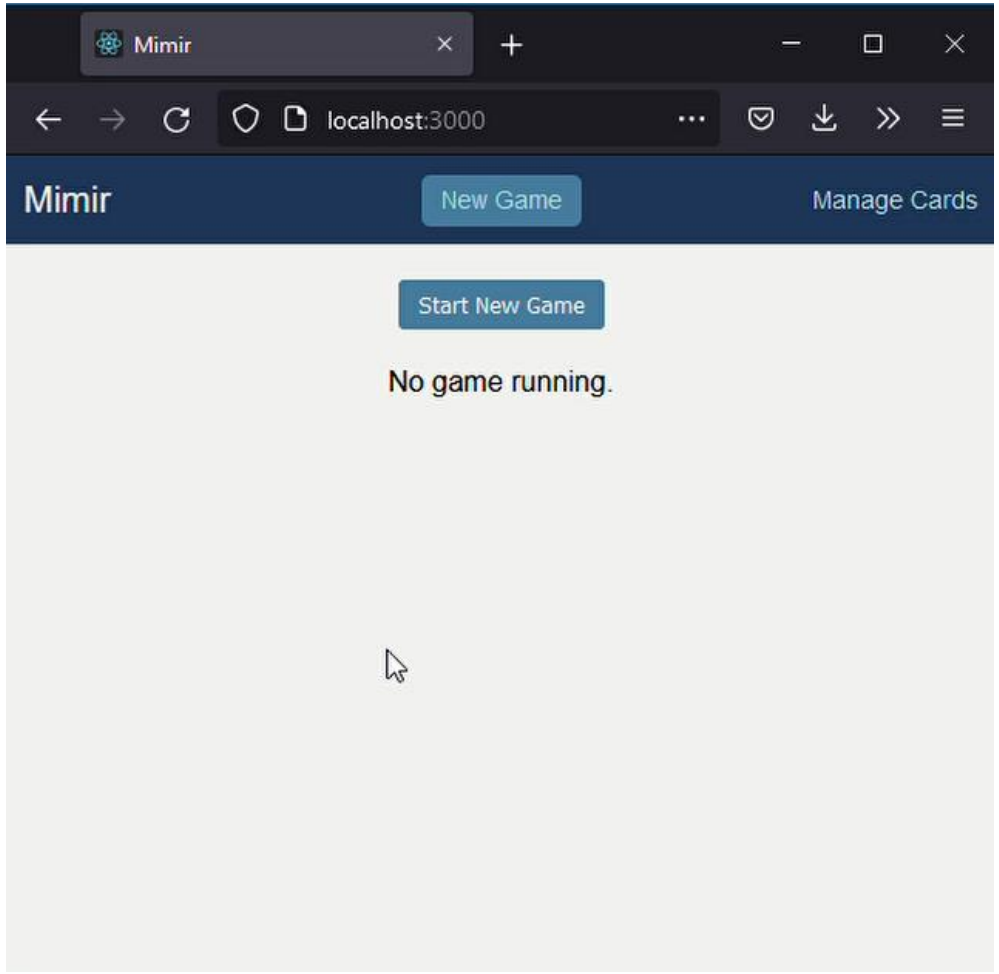
Mimir Learning App

- Learn-Karteikarten-App nach Spezifikation
  - Backend inkl. Swagger vorgegeben
  - Feature-Set vorgegeben - keine optionalen Features
  - Frei in UI-Design

- 2er-Gruppen eintragen: https://moodle.ost.ch/mod/choicegroup/view.php?id=238107
  - 3er-Gruppen nur in Abklärung mit Tobi - Zusatzaufwand

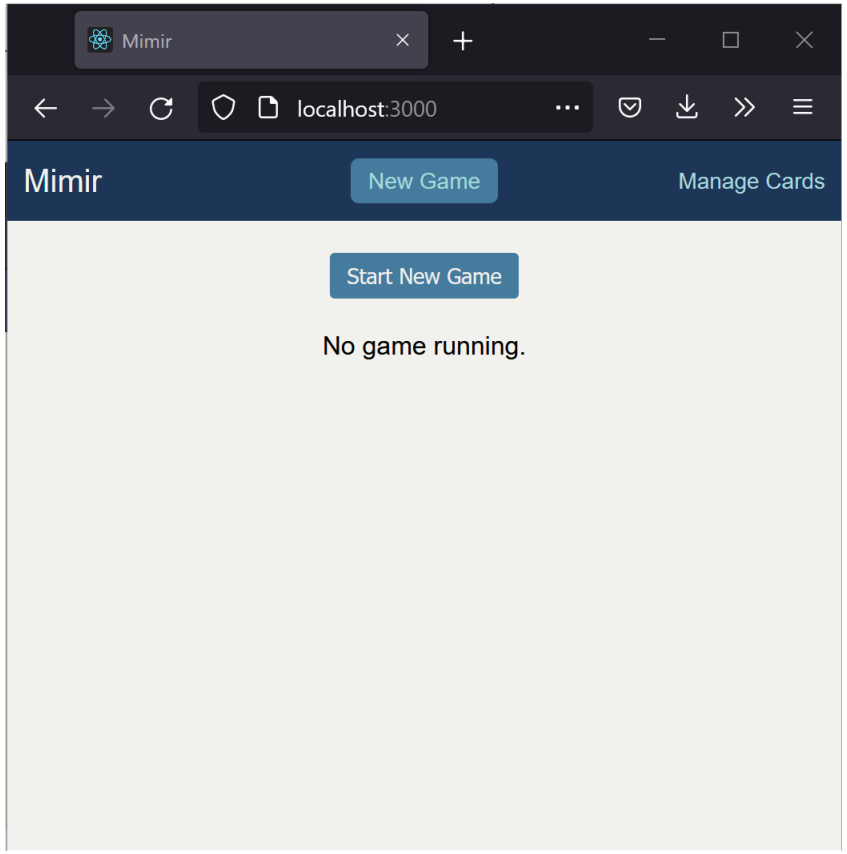
- Abgabe bis 9. Oktober (vor Herbstferien) in Moodle: https://moodle.ost.ch/mod/assign/view.php?id=238110
  - Link wird später freigegeben.

Mimir Learning App



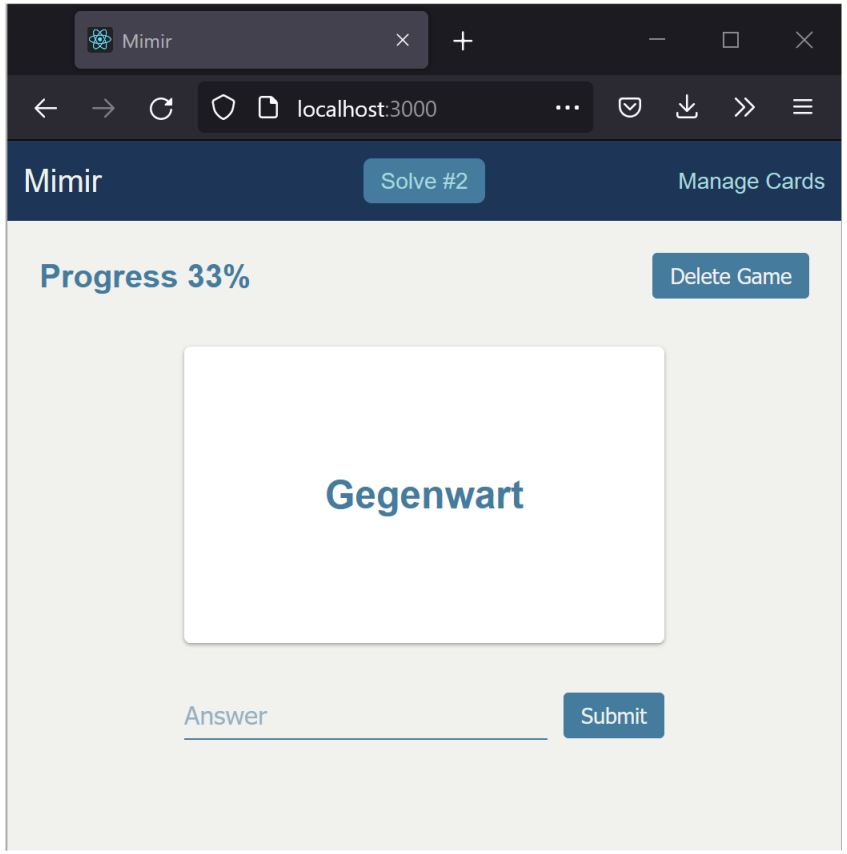
Game Page (new)

- NavBar
  - Link zu Game Page mit Frage-Nummer
  - Link zu Card Übersicht
- Button in NavBar zeigt "New Game" an
- Spielfeld hat Button um neues Spiel zu erstellen
- Meldung, dass gerade kein Spiel aktiv ist



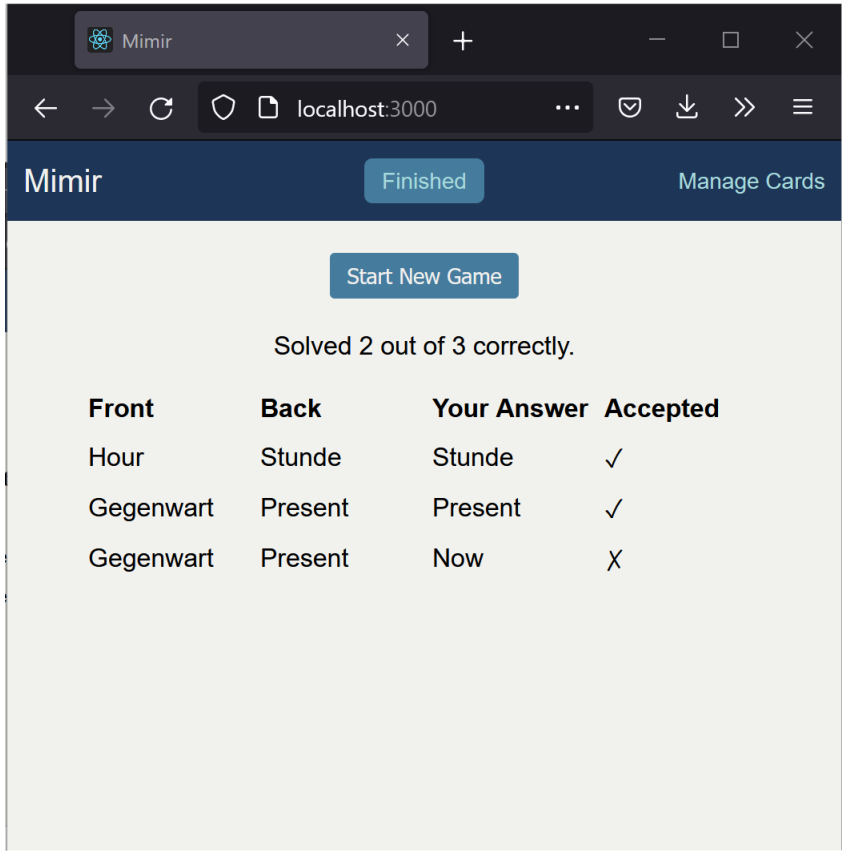
Game Page (ongoing)

- Fortschrittsanzeige in Prozent (Text)
- Button zum Spiel Löschen
  - Löscht Spiel und zeigt initale Seite an
- Vorderseite der Card
- Textbox für Antwort
- Submit-Button schickt Antwort ab und zeigt nächste Card an - oder Summary (end).



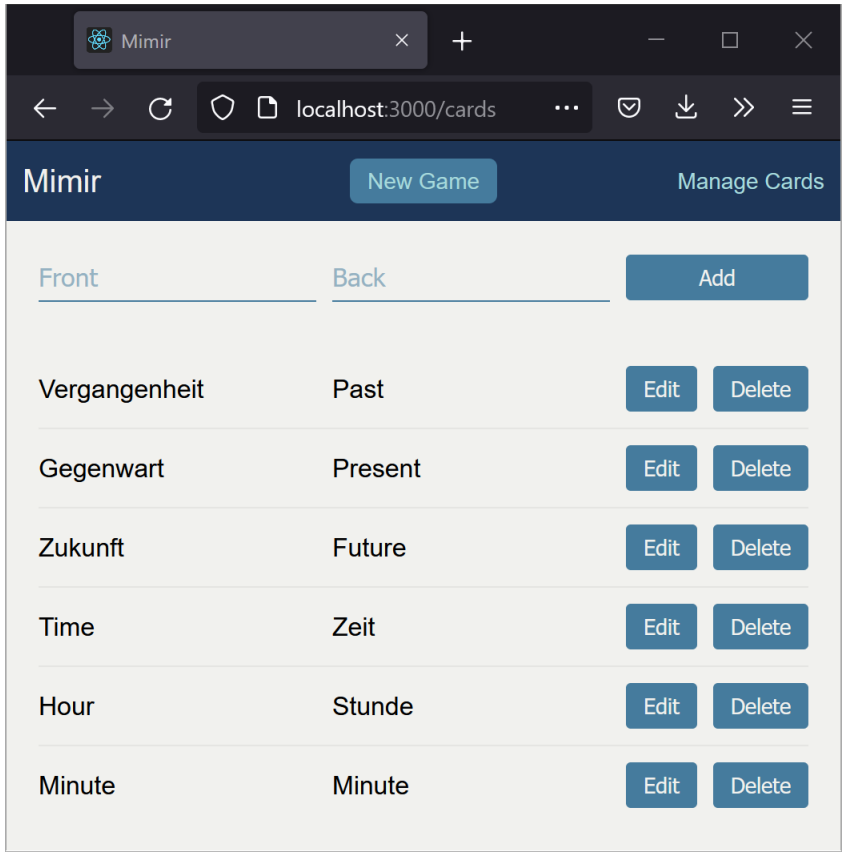
Game Page (end)

- Button, um neus Spiel zu starten.
  - Achtung: Altes Spiel muss zuerst gelöscht werden (2 Requests)
- Message mit Summe der korrekt gelösten und totalen Cards
- Tabelle mit Übersicht der einzelnen Lösungen



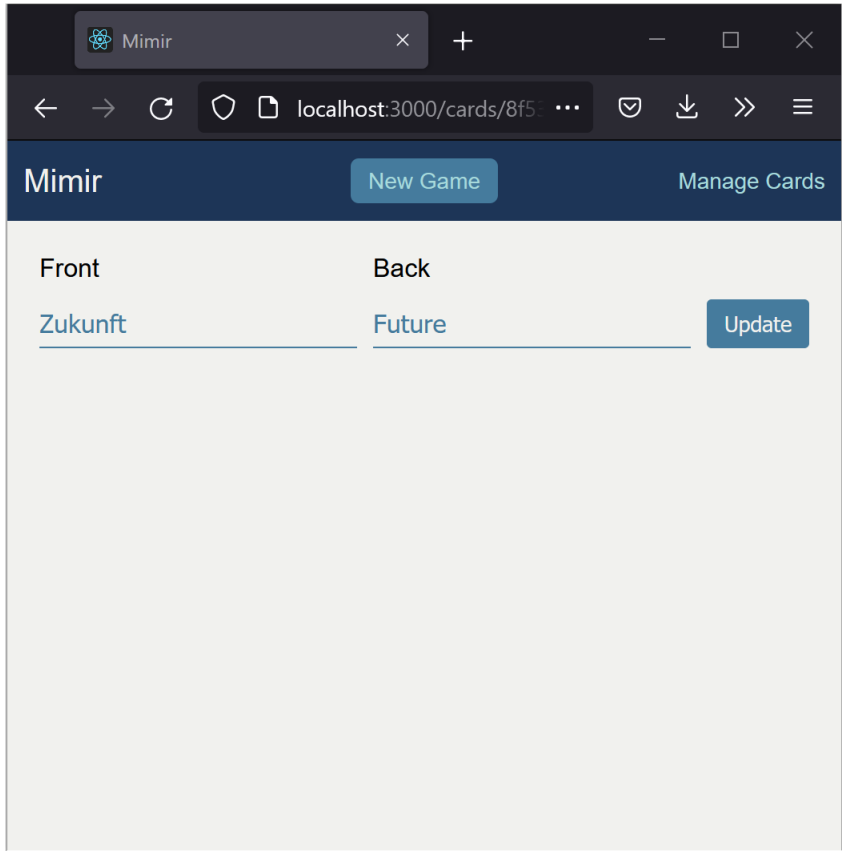
Cards Overview Page

- Erfassung & Übersicht auf gleicher Page
  - Neue Card wird sofort an Tabelle angefügt
  - Erfolgreiches Hinzufügen löscht setzt "Front" und "Back"-Input auf leeren String zurück
- Delete-Button löscht Card aus Tabelle & Backend
- Edit-Button leitet auf Detail-Seite weiter



Card Details

- Seite, um bestehende Cards anzupassen
- Erfolgreiches Update leitet zurück zur Übersichts-Page



Technische Anforderungen

- State-Management mit Context & Reducers
  - useState() für Formulare auf Game-Page & Card-Details
  - API-Daten via Context lesen & via dispatch() schreiben
- Styling mit styled-components - keine CSS-Dateien im Projekt
- Keine unnötigen API-Request - beachte Responses von API (siehe Swagger)
- Nur Frameworks/Libs aus Vorlesungen verwenden
  - Im Zweifelsfall fragen

Zusatzanforderungen 3er-Teams

- Server-Side-Validierung mit Fehleranzeige im Frontend
  - Keine leeren Antworten möglich
  - Erfassung/Editierung von leeren Texten verhindern

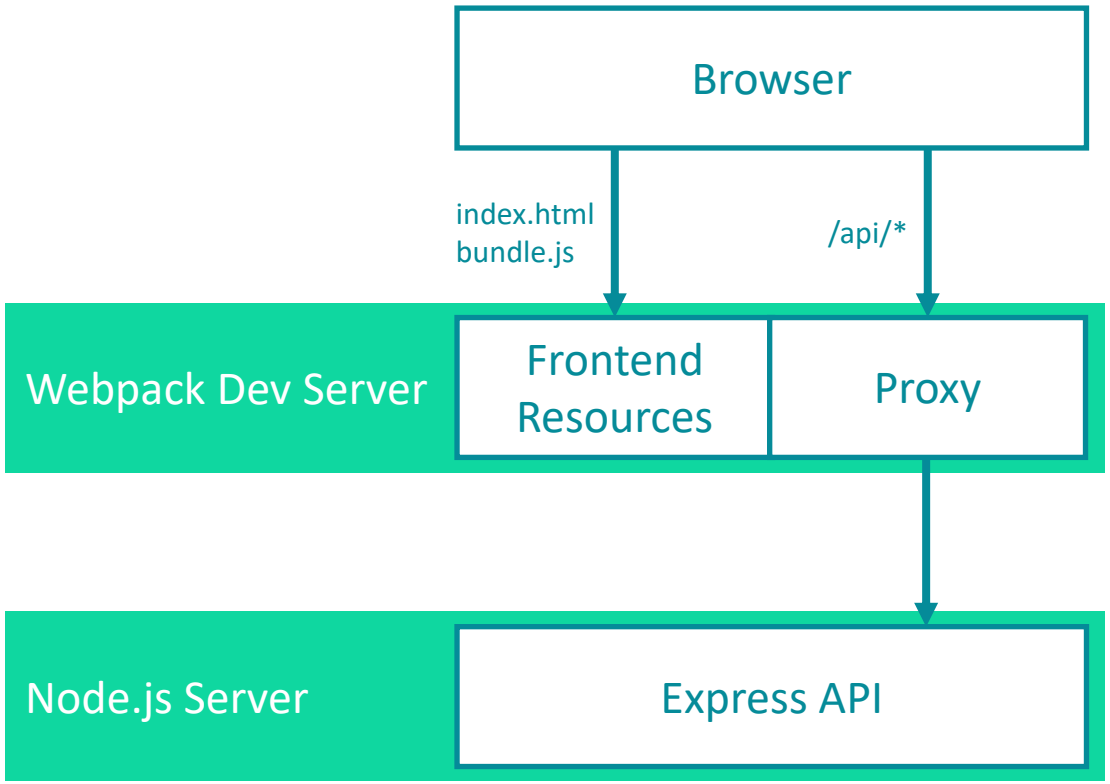
- Löschen von Cards nur mit Bestätigung
  - Möchten Sie die Card wirklich löschen?

- Sprachauswahl (z.B. EN/DE in NavBar)
  - Alle UI-Texte in mind. 2 Sprachen
  - Auswahl persistiert (Client oder Server)
  - ohne Frameworks/Libraries
  - Tipps bei Tobi abholen ;)

Architektur

- Zur Entwicklung API-Requests an Express-Backend weiterleiten
- Requests, welche keiner Frontend-Resource entsprechen, werden an Proxy weitergeleitet
- `package.json` erweitern

```
{
  "name": "simple-webpack",
  "proxy": "http://localhost:3001",
  "scripts": {
    ...
  },
  "dependencies": {
    ...
  }
}
```



Projektsetup

Backend

- git clone https://github.com/mas-se-we3/mimir-api
  - Enthält Swagger-Datei

Frontend

- npm -g i npm
- npx clear-npx-cache
- npx create-react-app *mimir* --template=typescript --use-npm
- tsconfig.json

```
{
  "compilerOptions": {
    "baseUrl": "src",
    …
  },
  "include": ["src"]
}
```

https://prettier.io

Fragen? Fragen!

Stellt viele Fragen, speziell in der Übungsstunde!

Marcel & Tobi helfen grosszügig.

Auch Fragen zu Grundlagen beantworten wir gerne.

Fehlt euch eine Übung oder Ressourcen zu einem Thema? Frühzeitig Fragen, wir machen euch gerne auf gute Tutorials oder Docs aufmerksam.

QUESTIONS AND ANSWERS

SYON-GROUP.COM