Dokumentation der Änderungen im Projekt

1. Anpassung der Ordnerstruktur

Die Ordnerstruktur wurde angepasst, um eine konsistente und dynamische URL-Struktur zu ermöglichen. Die bisherigen Ordner wie announcements, create-announcement, createEvent, invites, meinEntwurf, myevent, notification, userlist und users wurden in die folgende neue Struktur verschoben:

[instanz]/
 [organisation]/
  [user]/

Beispiel einer Route:
- localhost:3000/hs-fulda/ai/user/announcement

Vorteil der neuen Struktur:
- Jede*r Benutzer*in hat nun eine eigene URL, die auf seine/ihre spezifischen Daten und Ansichten verweist.
- Dies verbessert die Übersichtlichkeit und ermöglicht es, benutzerspezifische Inhalte dynamisch und effizient zu laden.

Auswirkungen:
- Die Pfade zu den StyledComponents mussten aktualisiert werden, da sich die Verzeichnisse geändert haben. Dies führt dazu, dass in Git eine Vielzahl von Änderungen sichtbar ist.

2. Einführung von userControl

Der Ordner userControl wurde hinzugefügt, um zentrale Funktionen und Formalitäten zur Benutzerverwaltung zu bündeln. Diese beinhalten unter anderem:
- Auslesen von Benutzerinformationen wie Benutzername, Instanz, Organisation, Rolle, Rechte.
- Verarbeitung der Benutzerrollen und -rechte: Die Rollen und Rechte werden hier verarbeitet, um zu bestimmen, welche Inhalte einem Benutzer angezeigt werden dürfen.

Funktionalitäten in userControl:
- Dynamische Anpassung der Benutzeransichten.
- Vorbereitung der Benutzerrechte für weitere API-Operationen (z. B. Schreiben, Bearbeiten, Löschen).

3. Erstellung des Ordners aapi/get-organisation

Ein neuer API-Ordner namens aapi/get-organisation wurde erstellt. Dieser dient dazu, Anfragen zur Organisation effizient zu verarbeiten. Hier werden neue Funktionen

hinzugefügt, um Daten wie Organisationen oder Benutzerinformationen zu lesen, zu schreiben, zu aktualisieren und zu löschen.

Hinweis:
Die alte API ist weiterhin im Projekt vorhanden, wird aber derzeit nicht verwendet.

4. Erstellung der Datei users.json

Eine Beispieldatei users.json wurde im Verzeichnis /app/public/data/ hinzugefügt. Diese Datei dient als provisorische Datenbank, um während der Entwicklung bis zur Fertigstellung des Backends funktional arbeiten zu können.

Beispieldaten in users.json:


```
{
 "instanz": "Eventmanager",
 "organisation": [
  {
   "name": "Buchhaltung",
   "users": [
    {
     "id": 1,
     "username": "admin",
     "password": "1234",
     "email": "admin@beispiel.com",
     "name": "Admin Benutzer",
     "role": "organisation-admin",
     "organisation": "Buchhaltung"
    },
    {
     "id": 1733319634984,
     "username": "orgi",
     "password": "1234",
     "email": "asdas",
     "name": "asdas",
     "role": "organisator"
    }
   ]
  }
 ]
}
```


5. Anpassungen in Sidebar.js

In der Sidebar wurde eine Funktion implementiert, um die Benutzerinformationen aus der Datei users.json auszulesen und dynamisch anzuzeigen.

Beispielcode in Sidebar.js:
```javascript
const { userInfo } = useUserContext();
const [role, setRole] = useState("");

useEffect(() => {
  if (userInfo && userInfo.role) {
    setRole(userInfo.role);
  }
}, [userInfo]);

if (!userInfo) {
  return null;
}
```

Dynamischer Basispfad:
Der Basispfad wurde so gestaltet, dass er die Instanz, Organisation und den Benutzernamen berücksichtigt:

```javascript
const basePath = `/${userInfo.instanz}/${userInfo.organisation}/${userInfo.username}`;
```

Vorteil:
- Benutzer*innen können jetzt dynamische Inhalte basierend auf ihrer Rolle und URL-Struktur sehen.

6. Dynamische Inhalte basierend auf Benutzerrollen

Die Sidebar zeigt jetzt Inhalte an, die von der Rolle des/der Benutzer*in abhängen.

Beispiele:
- Rolle: organisation-admin
  - Alle Inhalte werden angezeigt.
- Rolle: teilnehmer
  - Nur folgende Inhalte sind sichtbar:
    - Terminmanagement
    - Meine Veranstaltungen

Vorteil:
- Diese dynamische Anpassung erhöht die Sicherheit und Benutzerfreundlichkeit, da nur relevante Inhalte angezeigt werden.

7. Änderung der Login-Seite

Die Hauptdatei page.js wurde so angepasst, dass sie nicht mehr die LogInOut-Seite aufruft, sondern die neue Seite SimpleLogin.

Hinweis:
- Auth0-Login bleibt unverändert im Code vorhanden, wird jedoch momentan nicht aufgerufen.

Zusammenfassung der wichtigen Änderungen

1. Ordnerstruktur: Neue dynamische Struktur mit Instanz, Organisation und Benutzer.
2. userControl: Verwaltung von Benutzerinformationen, Rollen und Rechten.
3. Neue API-Struktur: aapi/get-organisation für strukturierte Anfragen.
4. Beispieldaten: Temporäre JSON-Datenbank (users.json) zur Überbrückung bis zur Backend-Integration.
5. Sidebar: Dynamische Inhalte basierend auf Benutzerrolle und URL.
6. Login-Seite: Umstellung auf SimpleLogin.

PS: Um die Seite zu Testen Loge dich bitte mit einem Benutzer aus der .json datei ein.
Username: admin Password: 1234

Auch wenn du die Seite aktualisert sollte die Sidebar verschwinden, weil die Regeln hierzu noch nicht festgelegt sind (also beim teste im Sidebar hin und her klicken und NICHT aktualisieren, diese punkte werden noch behoben)