



FH Salzburg
MultiMediaTechnology

Studentenfutter 2022/23

Abchlussprojekt im Bachelor-Studium
MultiMediaTechnology

Entwurf



Anhand des Abschluss-Projektes MMP3 zeigen Sie, dass Sie im Web Development professionell arbeiten können.

Im Projektverlauf sammeln Sie in diesem Dokument Beispiel, die Ihre Kompetenzen belegen. In den letzten Wochen des Projektes legen Sie diese Belege in einem **Fachgespräch** und bei einer **Präsentation** vor. Dabei beschreiben Sie die Problemstellung und argumentieren sie Ihre Vorgehensweise. Sie beantworten Fragen und diskutieren Alternativen.

Verwenden Sie für ihre Antworten das Fachvokabular der Informatik bzw. des Web Development, Code Beispiele, Diagramme, sowie Verweise auf das Repository ihres Projekts (bitte möglichst spezifisch auf einzelne Commits, Pull Requests, Files).



Inhaltsverzeichnis

(1)	<i>Ich kann ein Projekt von Idee bis Umsetzung entwickeln</i>	5
(2)	<i>Ich kann meinen Beitrag im User-Centered-Design leisten.....</i>	8
(3)	<i>Ich kann ein Backend auswählen und implementieren.....</i>	12
(4)	<i>Ich kann eine Datenbank entwerfen und implementieren.....</i>	13
(5)	<i>Ich kann ein Frontend aufsetzen und implementieren.....</i>	18
(6)	<i>Ich kann meinen Beitrag zur Barrierefreiheit des Projektes leisten</i>	20
(7)	<i>Ich kann meinen Beitrag zum Datenschutz im Projekt leisten</i>	23
(8)	<i>Ich kann meinen Beitrag zur Security des Projektes leisten</i>	25
(9)	<i>Ich kann meinen Beitrag im Software-Entwicklungs-Prozess eines Web Projekt leisten.....</i>	28
(10)	<i>Ich kann meinen Beitrag zu Web Operations leisten</i>	30
(11)	<i>Tanja kennt die Arbeitsteilung in Web-Projekten</i>	32
(12)	<i>Tanja kann User Stories implementieren und deployen</i>	33
(13)	<i>Tanja kann Fehler im Programm (Bugs) finden und beheben.....</i>	36
(14)	<i>Tanja kann Unit-Tests und End-to-End Tests schreiben.....</i>	37
(15)	<i>Tanja kann ein Refactoring durchführen</i>	39
(16)	<i>Lisa kennt die Arbeitsteilung in Web-Projekten.....</i>	40
(17)	<i>Lisa kann User Stories implementieren und deployen</i>	41

(18)	<i>Lisa kann Fehler im Programm (Bugs) finden und beheben</i>	44
(19)	<i>Lisa kann Unit-Tests und End-to-End Tests schreiben.....</i>	45
(20)	<i>Lisa kann ein Refactoring durchführen</i>	47
(21)	<i>Kerstin kennt die Arbeitsteilung in Web-Projekten</i>	48
(22)	<i>Kerstin kann User Stories implementieren und deployen</i>	49
(23)	<i>Kerstin kann Fehler im Programm (Bugs) finden und beheben</i>	54
(24)	<i>Kerstin kann Unit-Tests und End-to-End Tests schreiben.....</i>	55
(25)	<i>Kerstin kann ein Refactoring durchführen</i>	57

(1) Ich kann ein Projekt von Idee bis Umsetzung entwickeln

Ich kenne, ich kann.

Die Idee

Gemeinsam Essen ist ein sozialer Faktor und eignet sich hervorragend, um neue Leute kennen zu lernen. Außerdem ist es aufwendig, nur für sich alleine zu kochen und oft hat man auch keine Lust dazu. Studentenfutter ist eine Webapp für Studierende, um sich zum Essen zu verabreden. Das Ziel dabei ist, neue Leute kennenzulernen und gemeinsam zu Essen.

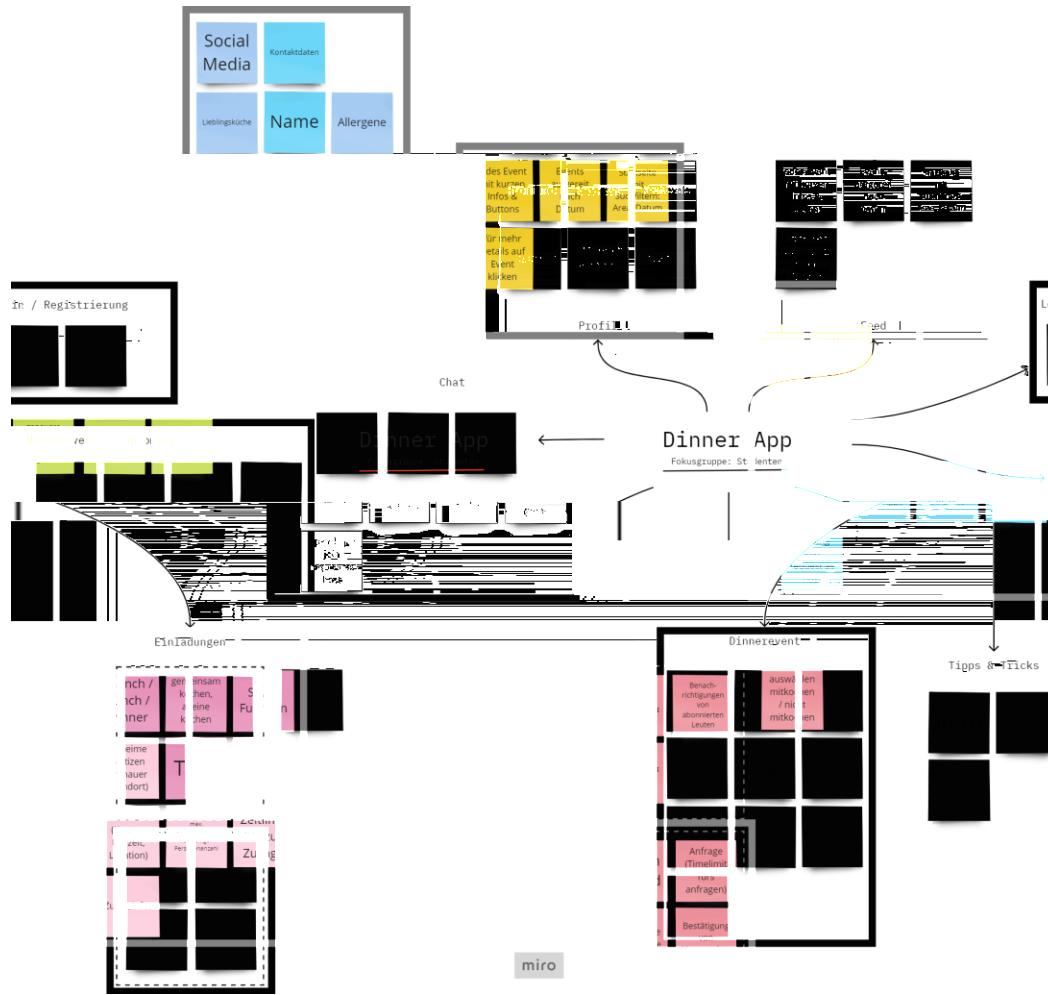


Abbildung 1: Brainstorming

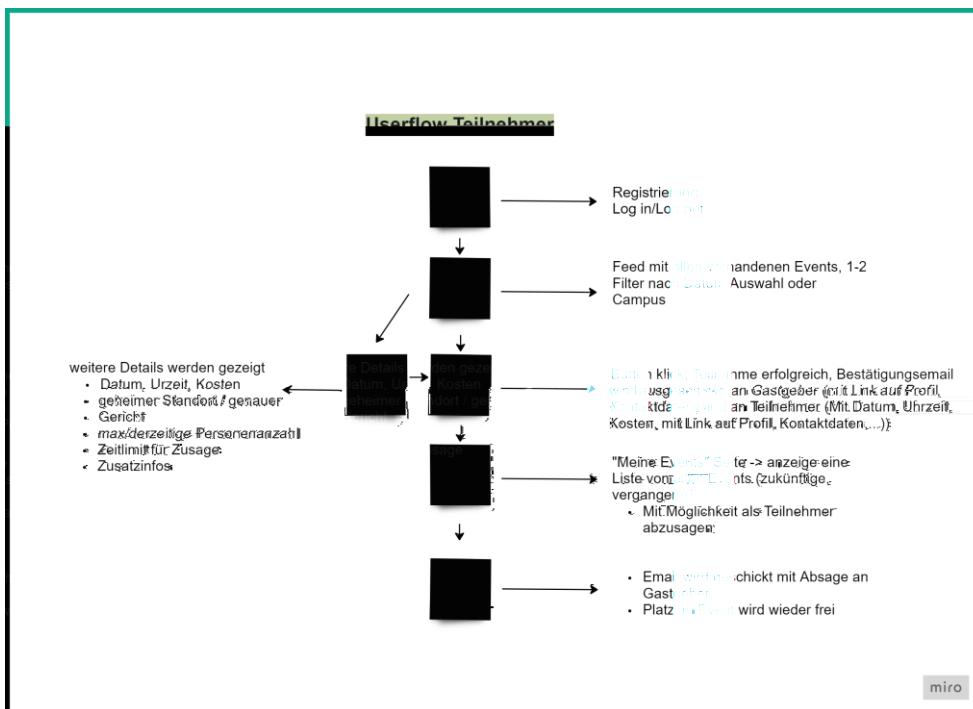


Abbildung 2: Userflow Guest

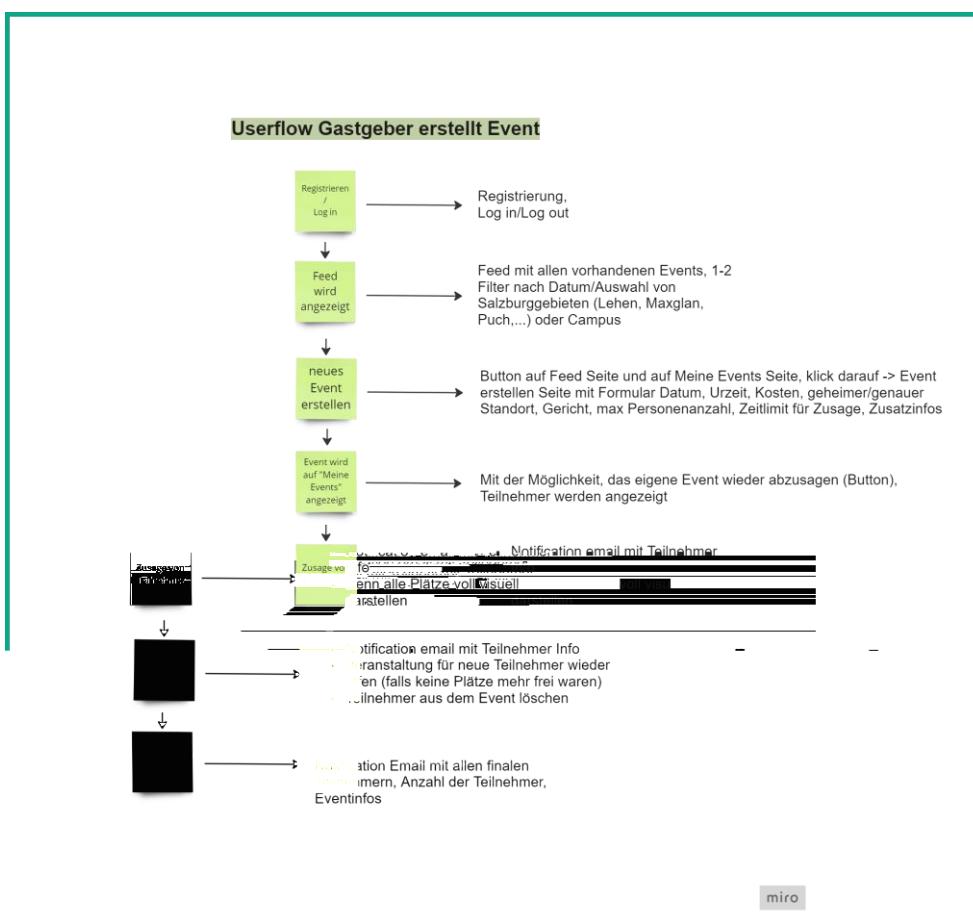


Abbildung 3: Userflow Host

Aufbau der Webapplikation

Zu Beginn muss man sich mit dem FH-Account registrieren und anschließend wird ein Profil erstellt. Hier werden persönliche Informationen wie Standort, Lieblingsküche, Allergene, Präferenzen und Profilbild eingetragen.

Auf der Startseite werden die Events der aktuellen Woche angezeigt. Dabei gibt es Filter-Möglichkeiten nach Campus und Datum. Die Events werden mit den wichtigsten Infos angezeigt, und es gibt die Möglichkeit, direkt an den*die Veranstalter*in eine Anfrage zu senden. Für nähere Informationen kann auf ein Event geklickt werden, um weitere Details zu sehen.

Wenn man sich dazu entscheidet ein Event zu veranstalten, wird ein neues Event erstellt. Dabei werden Informationen wie Gericht, Personenanzahl, Kosten, Datum/Uhrzeit und Standort angegeben. Zusätzlich werden auch Tags für vegan, vegetarisch, glutenfrei usw. hinterlegt. Interessiert man sich für ein Event, wird dem*der Veranstalter*in eine Anfrage geschickt. Der Veranstalter muss diese dann bestätigen.

Es gibt eine Übersicht für alle meine angefragten und zugesagten Events, bei denen ich teilnehme. Zusätzlich werden auch die Events angezeigt, die ich selbst veranstalte. Jedes Event kann im Anschluss bewertet werden. Die Bewertungen der Events sind dann für alle sichtbar, außerdem wird die Gesamtbewertung im Profil angezeigt. Es werden diverse Notifications an die User gesendet, diese werden im "Notifications" Bereich angezeigt und zusätzlich werden Emails versendet (Beispiel: "Dein Event findet morgen statt").

(2) Ich kann meinen Beitrag im User-Centered-Design leisten

*Ich kann Methoden der Analysephase anwenden: Fokusgruppe, Befragung, Contextual Inquiry, Beobachtung, etc. Ich kann Benutzer*innenerhebung konzipieren und durchführen: Testplan erstellen, Benutzer*innenstudie durchführen, Ergebnisse auswerten.*

Beschreiben Sie in welche Analysen und Evaluierungen Sie im Laufe des Projekts durchgeführt haben, und welche Erkenntnisse Sie gewonnen / Maßnahmen Sie ergriffen haben.

Preproduction: Definition der Personas

Der Austauschstudent:

Name: Ola, Norwegen

Gender: männlich

Alter: 24 Jahre

Studium: MMT - Game

Interessen: Ski fahren, Eishockey spielen, Gitarre spielen, reisen besonders Backpacking, Sprachinteresse, Gaming, feiern,

Kochlevel: 4

Räumlichkeiten: Studentenheim am Campus

Erste Phase (1. Studiwoche und danach bis vor der 2.Studiwoche)

Das Ergebnis vom Usertesting:

USER TESTING FEEDBACK

- Name der App sollte englisch sein, oder sonst name der App erklären -> Erklärung auf Landingpage, Name bleibt gleich (Alex), Kerstin
- Eventdetail:
 - hostname nicht rechtsbündig mit Bild (Bugfix, Lisa) ✓
 - Restplätze und Zeitlimit präsenter (Farbe ändern auf StandardTextFarbe und FETT, Lisa) ✓
 - mehr ersichtlich machen dass es ein Link ist beim Rezept (Wenn es ein link ist, dann Icon anzeigen neben Infoclon + Text soll auch klickbar bleiben, Lisa) ✓
 - host_anklicken und zu Profile kommen. Profilbild ± hv.host.soll.auf.Profil verlinken! (Lisa) ✓ (wenn man auf dem Profil ist muss der go_back_button noch eingeblendet werden)
 - Gästeliste nicht richtig aligned (Bild und Name immer links aligned und X rechts, (space-between)) Lisa ✓
 - Dish/Menü mit infoclon nicht aligned (Bugfix, Lisa) ✓
 - Gästeliste auf jeweilige Profile verlinken (Bild und Name klickbar machen) Lisa ✓
 - Überschriften bei menu, info und teilnehmer, so ist das nicht ersichtlich (In die jeweilige Card selbst das Label: Menu, About the event, Guestlist) Lisa ✓
 - Menu tooltip mobile verschiebt Content wenn es geöffnet ist (soll nach links geöffnet werden mit left/right position und wenn man rausklickt, soll es auch zugehen) Lisa!!
 - Handy und Phone anzeigen wenn erfolgreich angemommen -> Datenschutz (check ob request accepted ist) Lisa ✓
- (wie Gemüse) und das mit opacity regulieren (wird geändert, Vorschläge von Alex abwarten) - Kerstin ✓
- Hauptfarbe -> verbindung mit fresh, gesund, food sharing ist öko (Primärfarbe wird auf grün geändert, Alex) - Kerstin ✓
- entansicht für Fragen in WhatsApp?
- Infocards (Location Icon große ändern, Tanja) ✓
- mehr Beschreibung lesen (erstes dish wird immer vollständig angezeigt, die anderen schrumpfen, Kerstin) ✓
- Bugfix_Tanja) ✓
- a) ✓
- bearbeiten vom User default bild verwenden und Bild optional machen, Tanja) ✓
- uen um die Email von anderen Emails zu unterscheiden (einen Smiley oder icon(löffel und kebab),kerstin) ✓
- requests: kein reload der page wäre gut (Kerstin) ✓
- thentifizierung geschickt werden -> anzeigen dass Lehrende die App nicht nutzen können (Text für Landingpage, Alex, Kerstin)
- ra page "finish profile" / "Profil erstellen" (Tanja, neue Page "finish profile" + next-auth nochmal checken) ✓
- senter machen (icon von Alex, aus der "my-events" page rausnehmen kerstin) ✓
- in (loading-komponente erstellen, kerstin + icon von Alex) ✓
- kerstin ✓
- man das default Styling beeinflussen? (z.B. beim Datum Breite, Farbe, positionierung) (tanja)
- mular oder noch nicht vollständig ausgefüllt (lisa)
- en, wenn man es richtig eingetippt hat (onChange prüfen) (lisa) ✓
- gesetzt werden (tanja) ✓
- en) (bildkomprimierung beim upload?, tanja) ✓
- gezeigt (tanja) ✓
- rstellen (tanja) ✓
- t mehr in die vergangenheit (lisa) ✓
- 1 Stunde später default wert setzen, lisa) ✓
- Date" ist noch möglich (bei onchange die zwei datums vergleichen und fehler setzen wenn es nicht passt) (lisa)
- eldern z.B. in Profil-Formular (lisa) ✓
- egen infotext (Adresse als input feld anzeigen aber nicht bearbeitbar (vl. ausgegraut?), lisa) ✓
- unten (kerstin) ✓
- Info bei timelimit dazuschreiben für was das ist und info-text dazu mit infoicon) (lisa) ✓
- scrollen eigentlich hinfällig wenn onchange bei feldern funktioniert) (lisa) ✓
- nach anderem ist dann keine fehler meldung (lisa)
- zen und nicht im time input (lisa) ✓
- tipants) (lisa) ✓
- at least 1 statt 0 (lisa) ✓
- (logout aus footer raus und als button unter navigation) kerstin ✓
- iante wenn info fehlt (tanja) ✓
- on ID (apostroph hinzufügen, tanja) ✓
- have a look (kerstin) ✓
- uch in eventdetail (kerstin) ✓
- card is also clickable (kerstin) ✓
- nte), z.B.: Notification dass Emails ausgesendet werden und in Requests die requests angeschaut werden können (Bei Join event ein pop up mit info was gerade passiert ist und nächste Schritte, komponente erstellen) (kerstin) ✓
- (Kerstin) ✓
- extendedEventPreview und formular (lisa) ✓
- Navbars wird höher (kerstin) ✓
- Studenten, die am Campus wohnen

Krone zu Chefmütze ändern, Krone ein bisschen anders
 Hintergrund popiger und vielleicht Icons drüber
 Farben wirken nicht nach essen, Grün als Hauptfarbe
 Nice to have: Kommentarfunktion bei den Events
 Nice to have: SMS funktion für Benachrichtigungen
 Location icon nicht inline mit der Info in der Liste
 EventDetailPreview: Anzeige des Menüs ändern
 Small Event Preview Fotos object-fit: cover (oder anders)
 auth teamname auf studentenfutter ändern (Tanja) ✓
 Profil Fotos sollten freiwillig sein (beim erstellen + ändern)
 im Betreff von der email ein eindeutiges Icon einbauen
 Join event action bei Eventdetail und all events unklar
 Login: Lehrende haben andere Infos die bei der Anmeldung voraussetzen
 first login redirect -> after first login route to an event
 Invitation Updates: Naming nicht klar, vl. etwas präziser
 LadeScreen Komponente / Loading noch nicht schick
 Bottom Navigation bei Edit Seiten ausblenden? (Kerstin) ✓
 Formulare generell:
 • Mobile (Android) Styles weichen ab, vl. kann man das ändern
 • Submit Button disaben wenn Fehler im Formular
 • Fehlermeldung bei Formular soll verschwinden

- Create Profile Formular:
 - wenn bild schon da ist soll value beim editieren nicht überdecken
 - Bildgröße ((downscale function oder limit erhöhen)
 - Passwortmanager wird bei Feld "Instagram" aktiviert
 - Email adresse studiengang... als input felder
- Create Event Formular:
 - wenn datum gesetzt ist, dann kommt man nicht mehr zurück
 - timelimit date etwas später auf default setzen
 - händisches Eintragen von Timelimit vor "Event detail"
 - required fields mit sternchen darstellen (lisa) ✓
 - Info für genaue Adresse: info i wie bei allen infofeldern
 - Mobiles Styling Location: ist nicht so schlüssig
 - Rechtschreibfehler: Title statt Titel (lisa) ✓
 - Hintergrund Dishes mobil geht nicht ganz nach oben
 - time und timelimit untereinander in der form
 - hei_fehlerschlaufen_siehbit zum Fehler bei timelimit
 - Validierung datumfelder, wenn timelimit nicht gesetzt
 - Bei dish: delete X icon über den Input sichtbar
 - Placeholder alle links alignen (price, participants)
 - Error.messages überarbeiten -> must be filled
 - Label für Menü (lisa) ✓
 - Limit für Gäste und Kosten (max) (lisa) ✓
- Desktop Styling: Imprint / data-privacy / logo
- Profil:
 - Edit button padding fehlt in desktop version
 - Markus hosted events --> nicht so schön
- Requests: accept request application error --> keine Fehlermeldung
- Request kann geschickt obwohl event voll ist
- Request send links to event detail --> because
- Mehr feedback an den User (Popup Komponente)
- up mit info was gerade passiert ist und nächster Schritt
- 1 hours/ 1 days left (soll 1 hour/1 day left sein)
- price p.p hinzufügen im eventdetail und in der Ladezeit
- fh logo im imprint dazu (tanja) ✓
- Mobile Bug: Textfeld reinklicken, runterscrollen
- Nice-To-Have: sollte für alle sein, nicht nur für iOS

Alle Punkte wurden von den jeweiligen Personen abgearbeitet.

Zweite Phase (2. Studiwoche und danach)

USERTESTING 2 - PUNKTE

- timelimit create event - timelimit now() -> min-value auf eine stunde nach jetzt (lisa)
- timelimit eine minute vorher, ansonsten könnte man sie auswählen und es kommt aber ein validation error (das selbe wie 1.)
- Raumnummer Characters beschränken (10) -> Tanja
- Profilpage: alle Infos in Card (kerstin), Rückseite enthält zum übertragung (alex)
- Eventdetail: alle Infos in cards (alex)
- all events: leave event präsentier (leave und withdraw als secondary button) (kerstin)
- remove event: 500er ?? warum? loading state setzen (lisa)
- Info im navbar bei requests bei anfragen (roter punkt mit count) -> neuer Task im Board, studioworke einplanen
- toggle pass/upcoming margin, ohne slider! (tanja)
- filter: offizielle ist mehr margin-top (kerstin)
- nav desktop positionierung content defer (kerstin)
- vorlinken von events in requests -> link über eventnamen (kerstin)
- profil sieht komisch aus wenn viele events gehostet wurden (paginatio und content links soll nicht nach unten geschoben werden) -> tanja

- tooltip cursor on hover (lisa)
- profile page alignen (margin/padding nicht regelmäßig)
- edit profile button sticky mobil (tanja)
- toggle state in my events speichern für go back (nice-to-have)
- filter Zeit begrenzen und fehlermeldung, bei zu vielen zahlen fehler (kerstin)
- 404 seite stylen (alex illu)
- multiple select bei filter (Standorte) (nice-to-have)
- My events headlines mit "my" davor und wörter klein schreiben (tanja)
- Datumseingabe, wenn darüber gehover wird sollte ersichtlich sein, dass man es auch eingeben kann: Cursor ändern (Lisa)
- Titel zu stark eingeschränkt: 4 characters aktuell, sollte auch 2 character möglich sein fehlermeldung anpassen (lisa)
- im nachrichtenformular kann man nicht mehr in einem inputfeld mehrere linien eingeben, wenn man einen br drückt, dann geht es nicht weiter (alex)
- standard timelimit immer ein tag vorher -> 12:00 zu 11:00 (kerstin)
- cursor bei insta logo nicht darin (over) (tanja)
- Bug: 2/1 seats taken siehe screenshot lisa (kerstin abwickeln)

Alle Punkte werden von der jeweiligen Person abgearbeitet.

Dritte Phase (was ist für die 3. Studiwoche geplant)



Alle Punkte werden von der jeweiligen Person abgearbeitet.

(3) Ich kann ein Backend auswählen und implementieren

Ich kenne mehrere Backend-Technologien: statische Webseiten, PHP, Ruby on Rails, Node.js, Wordpress. Ich kenne Kern-Konzepte für das Backend wie: Routing, Model View Controller (MVC) Pattern, Templates, Object Relational Mapper (ORM), die jeweiligen Packagemanager / Erweiterungsmöglichkeiten: composer, gem, bundler, npm, yarn, themes, plugins.

Beschreiben Sie welche Anforderungen ihres Projekts für die Auswahl des Backends ausschlaggebend waren, und für welches Backend Sie sich entschieden haben.

Studentenfutter wird mit dem Web-Framework Next.js implementiert. Für die backend-seitige Implementierung verwenden wir die API-Routen von Next.js womit wir unser Backend implementieren. Wenn man Next.js in Kombination mit API-Routen verwendet, kann man folgende Vorteile erwarten:

- **Serverseitiges Rendern (SSR):** Next.js bietet die Möglichkeit, Webseiten auf dem Server zu rendern, bevor sie an den Client gesendet werden. Dies verbessert die Geschwindigkeit der Seiten und die Suchmaschinenoptimierung (SEO), da die Inhalte schneller geladen werden und Suchmaschinen-Crawler auf den vollständigen HTML-Inhalt der Seite zugreifen können.
- **Einfache API-Integration:** Next.js bietet eine einfache Möglichkeit, API-Endpunkte zu erstellen und zu integrieren. Mit den API-Routen kann man API-Endpunkte erstellen, die direkt in die Next.js-Anwendung integriert sind, was die Entwicklung und Wartung von Anwendungen erleichtert.
- **Bessere Sicherheit:** Wenn man API-Endpunkte in Next.js integriert, kann man die Sicherheit verbessern, indem man beispielsweise Token-Authentifizierung und Rate-Limiting implementiert.
- **Bessere Skalierbarkeit:** Wenn man Next.js in Kombination mit API-Routen verwendet, kann man eine bessere Skalierbarkeit der Anwendung erreichen. Indem man API-Endpunkte auf separate Server auslagert, kann man die Last auf verschiedene Server verteilen und die Verfügbarkeit der Anwendung erhöhen.
- **Geringerer Ressourcenverbrauch:** Durch die Verwendung von Next.js und API-Routen kann man den Ressourcenverbrauch der Anwendung reduzieren. Durch die Verwendung von SSR und die Reduzierung der Anzahl der Anfragen an die API-Endpunkte kann man die Anforderungen an die Infrastruktur reduzieren und Kosten sparen.
- **Serverless-Deployment:** Next.js kann auf Serverless-Plattformen wie Vercel, AWS Lambda oder Google Cloud Functions bereitgestellt werden. Das bedeutet, dass kein separates Backend benötigt wird, um die Anwendung auszuführen, da der Code direkt auf der Plattform ausgeführt wird.
- **Einfache Datenverarbeitung:** Wenn die Anwendung keine komplexen Datenverarbeitungsanforderungen hat und keine umfangreiche Datenbank benötigt, können Daten in statischen Dateien oder in einem externen Service wie Firebase oder Airtable gespeichert werden.
- **Schnelle Entwicklung:** Wenn Zeit ein wichtiger Faktor ist und eine schnelle Entwicklung bevorzugt wird, kann Next.js als Full-Stack-Framework verwendet werden, das sowohl den Frontend- als auch den Backend-Teil abdeckt. Dadurch können Entwickler schnell Prototypen erstellen und ihre Anwendungen schnell auf den Markt bringen.
- **Weniger Komplexität:** Die Verwendung eines separaten Backend-Frameworks kann die Komplexität des Codes erhöhen und das Debugging erschweren. Wenn alle Funktionen in Next.js geschrieben werden, können Entwickler möglicherweise einfacher und schneller Fehler beheben.

Insgesamt bietet Next.js in Kombination mit API-Routen viele Vorteile, die die Entwicklung von Webanwendungen erleichtert, daher haben wir uns dazu entschieden diese zu verwenden.

(4) Ich kann eine Datenbank entwerfen und implementieren

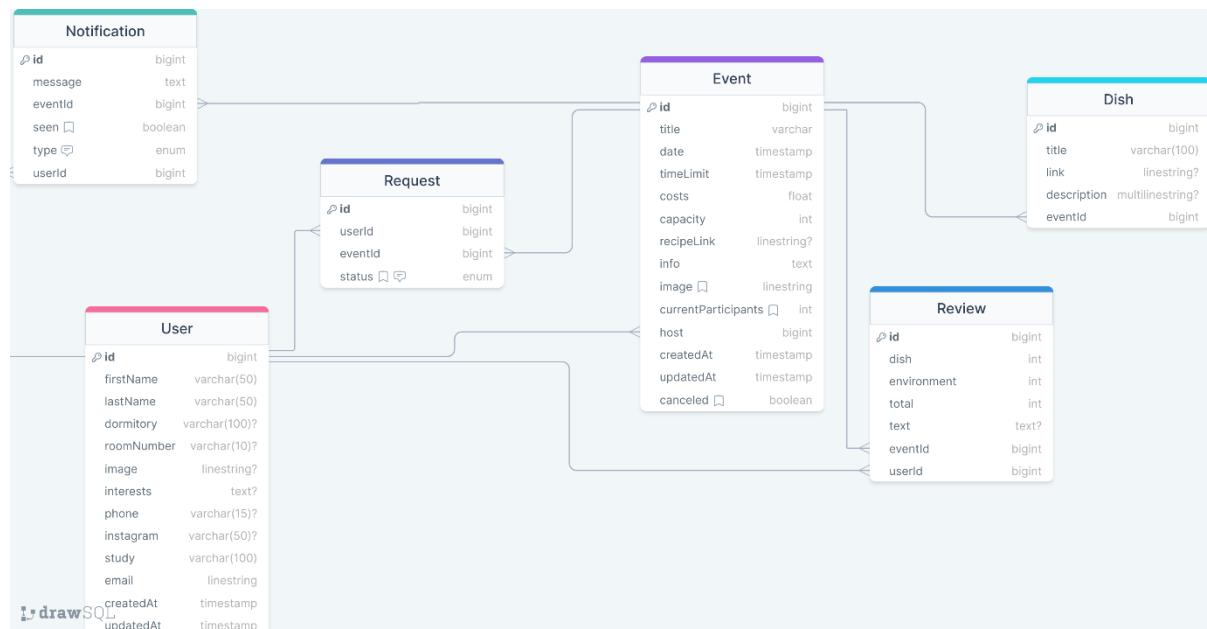
Ich kenne die Prinzipien und die praktische Anwendung von relationalen Datenbanken. Ich kann SQL in der Datenbank PostgreSQL verwenden. Ich kann ein ORM verwenden. Ich kann Veränderungen im Datenbank-Schema mittels Migrations in den Development Prozess einbinden. Ich kann Indexe und Transaktionen einsetzen. Ich kenne die speziellen Anforderungen an Volltextsuche, und kann Solr oder ElasticSearch einsetzen.

Prisma ORM

Für die Kommunikation mit der Datenbank wird Prisma verwendet, und als Datenbank PostgreSQL. Prisma ORM (Object-Relational Mapping) ist ein Tool, das Entwicklern dabei hilft, Datenbankabfragen in ihren Anwendungen zu vereinfachen und zu automatisieren. Da die Dokumentation für Next.js sehr gut ist und folgende Vorteile dadurch bestehen haben wir uns für Prisma entschieden:

1. Typsicherheit: Prisma ORM unterstützt die Typsicherheit von TypeScript und vermeidet damit Laufzeitfehler aufgrund von Typinkonsistenzen.
2. Datenbankunabhängigkeit: Prisma ORM ist datenbankunabhängig, d.h. es unterstützt verschiedene Datenbanken wie PostgreSQL, MySQL, SQLite und MongoDB.
3. Automatisierte Abfragegenerierung: Prisma ORM generiert Abfragen automatisch anhand des Datenbankschemas. Dadurch müssen Entwickler weniger Zeit mit dem Schreiben von Datenbankabfragen verbringen.
4. Leistungsoptimierung: Prisma ORM optimiert Abfragen und reduziert so die Datenbankzugriffe. Dadurch können Anwendungen schneller und effizienter arbeiten.
5. Einfache Integration: Prisma ORM kann einfach in bestehende Anwendungen integriert werden und unterstützt verschiedene Frameworks wie Node.js, Express und Fastify.
6. Skalierbarkeit: Prisma ORM ist skalierbar und kann auch mit großen Anwendungen und komplexen Datenbankstrukturen umgehen.

Datenbankschema



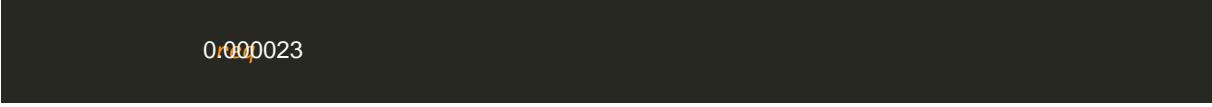
Beispiele

Zeigen Sie drei wichtige Abfragen, ein Beispiel für den Einsatz eines Index, ein Beispiel für den Einsatz einer Transaktion (jeweils kurzes Codebeispiel hier plus Verweis auf das Repository).

Transaktion

REPO: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/api/requests/%5Bid%5D/index.ts>

Hier wird ein bestehender Request von einem Gast zu einem bestimmten Event upgedatet, wenn der Host zusagt oder ablehnt. Wenn der Host zusagt, wird zusätzlich auch das Event upgedatet und die Guest-Anzahl um 1 erhöht. Wenn das Update vom Request nicht funktioniert, dann wird auch die Änderung im Event nicht vorgenommen.



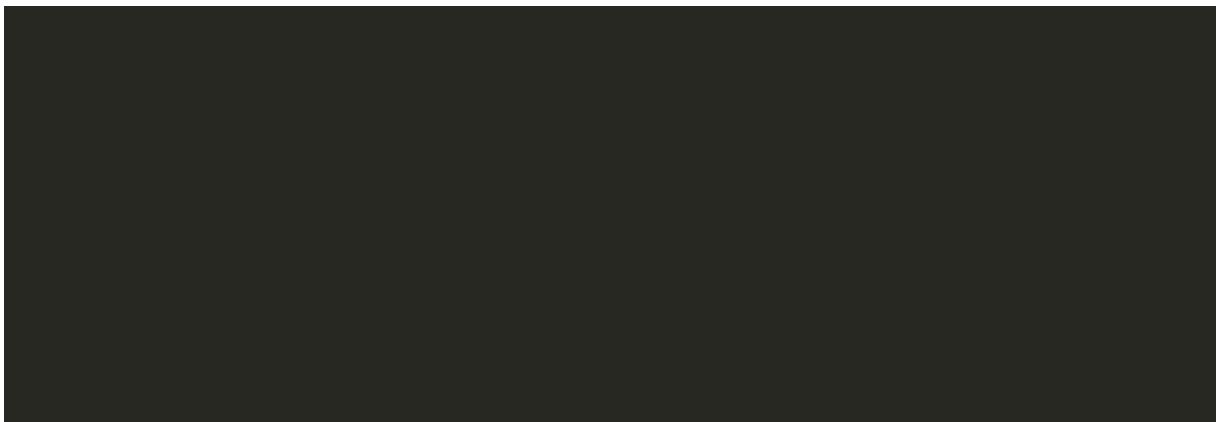
0:00:023

```
 5 prisma/migrations/20230518085745_add_indices/migration.sql □
...
... @@ -0,0 +1,5 @@
1 + -- CreateIndex
2 + CREATE INDEX "events_status_idx" ON "events"("status");
3 +
4 + -- CreateIndex
5 + CREATE INDEX "users_dormitory_idx" ON "users"("dormitory");
```

Abfrage 1: Event-Detail

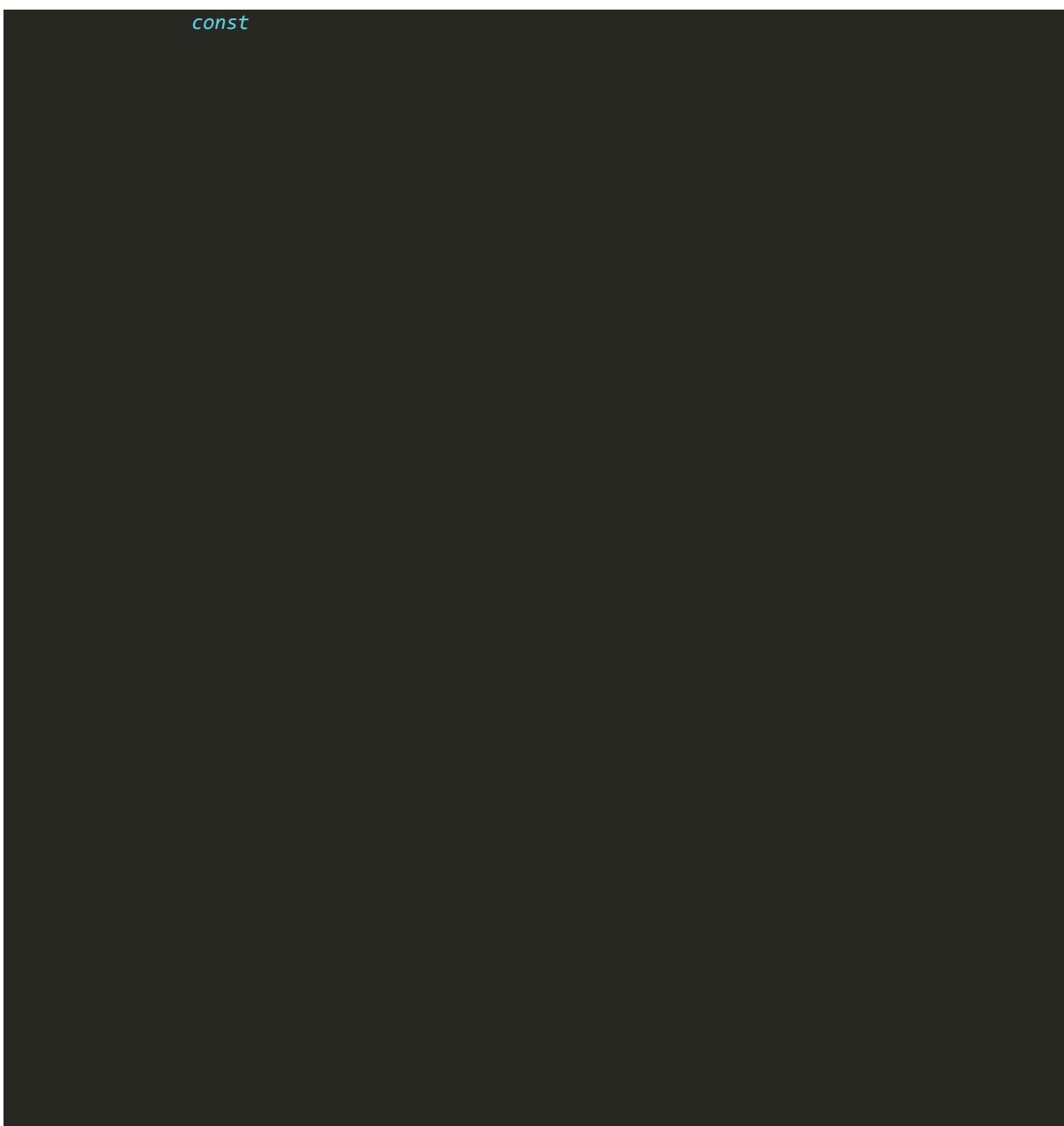
REPO: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/api/events/%5Bid%5D/index.ts>

```
const String req
```



Abfrage 2: Meine upcoming Events

REPO: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/api/my-events/index.ts>





Abfrage 3: Meine Notifications

REPO: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/api/notifications/index.ts>

```
const
```



(5) Ich kann ein Frontend aufsetzen und implementieren

Ich kann Javascript programmieren, kenne moderne Javascript Features wie Klassen, Module, Promises, Async Await, fetch. Ich kann die History API für Routing einsetzen. Ich kenne ein Template System für Javascript. Ich kenne die Grundkonzepte von React und kann Komponenten erstellen. Ich kann eine Architektur für CSS auswählen, kenne BEM. Ich kann eine Build-Pipeline mit npm und webpack für das Frontend aufbauen um babel, css Präprozessoren und Maßnahmen für die Performance zu automatisieren. Ich kenne die Möglichkeiten für Grafik im Browser: Pixelbilder, Canvas, SVG, WebGL.

Next.js

Es gibt mehrere Gründe, warum wir uns für Next.js entschieden haben:

1. Einfache Konfiguration: Next.js erleichtert die Konfiguration von Webanwendungen, indem es automatisch viele grundlegende Aufgaben übernimmt, wie zum Beispiel Routing und Hot Reloading. Wir müssen uns nicht um die Einrichtung von Webpack oder Babel kümmern.
2. Code Splitting: Durch Code Splitting kann Next.js den Code in kleine Module aufteilen und nur die erforderlichen Module laden, wenn sie benötigt werden. Das verbessert die Leistung und die Ladezeit der Webseite.
3. Unterstützung von React: Next.js wurde speziell für React entwickelt und ist damit für uns die ideale Wahl, da wir mit React schon gearbeitet haben
4. Leichtgewichtig und skalierbar: Next.js ist ein leichtgewichtiges Framework, das schnell und skalierbar ist. Es ermöglicht die Erstellung von statischen Webseiten und bietet Optionen für die Verwendung von Serverless-Architekturen.
5. Einfache Integration von Tools: Next.js bietet eine einfache Integration mit verschiedenen Tools und Frameworks wie TypeScript.

TypeScript

Wir verwenden in unserem Projekt TypeScript (v4.5.5). Folgende Punkte waren dafür ausschlaggebend:

1. Typsicherheit: Durch die Typisierung wird eine höhere Sicherheit beim Programmieren erreicht, da Fehler wie das versehentliche Übergeben von falschen Typen an Funktionen oder Variablen bereits während der Entwicklung erkannt werden können.
2. Verbesserte Lesbarkeit: Durch die Typisierung wird der Code lesbarer und verständlicher, da die Typen direkt im Code angezeigt werden.
3. Code-Refactoring: Durch die Typisierung wird das Refactoring von Code erleichtert, da der Compiler bei Änderungen an Typen oder Variablen auf Fehler hinweist und die benötigten Änderungen vorschlägt.
4. Bessere Zusammenarbeit: Durch die bessere Lesbarkeit und Dokumentation wird die Zusammenarbeit zwischen Entwicklern erleichtert.

Styled Components

Styled Components ist eine Bibliothek für React, die es Entwicklern ermöglicht, CSS direkt in ihre React-Komponenten zu integrieren. Statt separate CSS-Dateien zu schreiben oder Inline-Stile zu verwenden, können Entwickler ihre Komponenten mit CSS-Code stylen, indem sie ihn direkt in die JavaScript-Datei der Komponente schreiben.

Aufgrund folgender Vorteile haben wir uns für Styled Components entschieden:

1. Bessere Isolation: Da der CSS-Code direkt in der Komponente definiert wird, ist er automatisch isoliert und hat keinen Einfluss auf andere Komponenten oder Elemente auf der Webseite.

2. Dynamische Styles: Die Definition von Styles als JavaScript-Objekte ermöglicht es, dynamische Styles zu erstellen, die von Props oder Zuständen der Komponente abhängen.
3. Wiederverwendbarkeit: Durch die Definition von Styles als wiederverwendbare Komponenten können Entwickler Code wiederholen und die Wartung von CSS-Styles vereinfachen.
4. Mehr Lesbarkeit: Die Integration von CSS in den JavaScript-Code verbessert die Lesbarkeit, da der gesamte Code an einem Ort geschrieben wird.
5. Bessere Entwicklererfahrung: Durch die Verwendung von modernen JavaScript-Syntax-Features und die Integration mit Tools wie Visual Studio Code wird die Entwicklererfahrung verbessert.

Grafik-Technologien

Für die Pixelgrafiken in unserer Anwendung verwenden wir von Next.js die Image-Komponente. Next.js Image verwendet das HTML-Element **img** zum Laden von Bildern, optimiert jedoch den Ladevorgang durch dynamische Größenanpassung, Skalierung und Caching.

Die verwendeten Icons wurden von unserem MMA-Teammitglied Alex erstellt. Diese liegen im public-Folder im Projekt. Für die Verwendung der Icons im Code wird das npm-Package **svgr** verwendet, um Icons in React-Komponenten zu transformieren.

Responsive-Design

Unsere MMA-Kollegin Alex hat zwei Versionen unserer Anwendung entworfen. Ein mobiles Design und ein Desktop Design, wobei der Breakpoint 768px liegt.

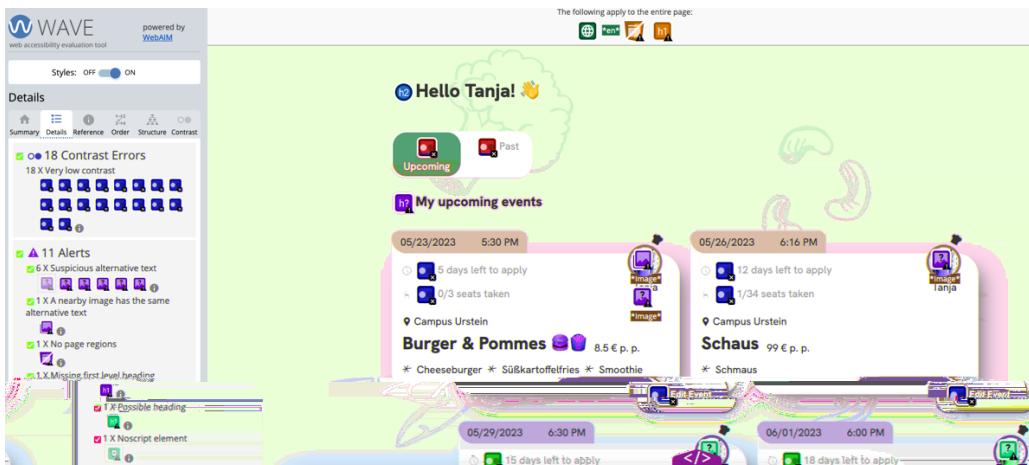
(6) Ich kann meinen Beitrag zur Barrierefreiheit des Projektes leisten

Ich kenne die Konzepte der Barrierefreiheit und des Universellen Design. Ich kann die Web Content Accessibility Guidelines (WCAG) und die Richtlinien für Accessible Rich Internet Applications (ARIA) praktische anwenden.

Beschreiben Sie wie sie die Barrierefreiheit Ihres Projekt überprüfen, und welche Maßnahmen Sie getroffen haben.

Folgende Maßnahmen haben wir für die Barrierefreiheit getroffen:

1. Verwendung von WAVE (Web Accessibility Evaluation Tool), das von der WebAIM-Organisation bereitgestellt wird.
2. Manuelle Überprüfungen: Durch die Webseite navigieren und sicherstellen, dass alle Elemente auf der Seite durch Tastaturbedienung und/oder Screenreader zugänglich sind. Sicherstellen, dass der Kontrast zwischen Text und Hintergrund ausreichend ist, um für Personen mit Sehbehinderungen lesbar zu sein.
3. Barrierefreiheitsrichtlinien: sicherstellen, dass Webseite den Richtlinien für Barrierefreiheit entspricht, wie z.B. den Web Content Accessibility Guidelines (WCAG). Diese Richtlinien definieren, welche Anforderungen für die Zugänglichkeit von Webseiten erfüllt werden müssen, um für Menschen mit Behinderungen zugänglich zu sein.



WAVE hat einige Fehler und Warnungen im Hinblick auf Barrierefreiheit gefunden. Dazu gehören hauptsächlich Kontrastfehler, fehlende strukturelle Elemente wie beispielsweise h1, h2, main, nav und footer, sowie fehlende Labels bei Formularen, doppelte Alt-Texte bei Bildern und fehlender Text bei Links. In ausgewählten Codebeispielen wird gezeigt, wie diese Fehler behoben wurden.

Kontrastfehler:

```
@@ -8,9 +8,9 @@ const size = {
  8   8   export const theme = {
  9   9     body: '#FFFFFF',
 10  10    text: '#343434',
 11  -   red: '#EF4B4B',
 11  +   red: '#DB1F1F',
 12  12    hoverRed: '#8a0606',
 13  -   primary: '#3d754c',
 13  +   primary: '#457251',
 14  14    hoverPrimary: '#306145',
 15  15    secondary: '#B3F0AB',
 16  16    green: '#a7dbbc',
```

Strukturelle Elemente:

```
@@ -42,7 +42,7 @@ export const Footer = () => {
 42   42     );
 43   43   };
 44   44
45 - const Wrapper = styled.div`
45 + const Wrapper = styled.footer`
 46   46     padding: 60px 30px;
 47   47     @media ${({props}) => props.theme.breakpoint.tablet} {
 48   48       padding: 60px;
```

```
@@ -50,7 +50,7 @@ const StyledLayout = styled.div<StyledLayoutProps>`
 50   50   }
 51   51   `;
 52   52
53 - const LayoutWrapper = styled.div`
53 + const LayoutWrapper = styled.main`
 54   54     @media ${({ theme }) => theme.breakpoint.tablet} {
 55   55       margin-left: 320px;
 56   56     }
```

Labels bei Formularen:

```
@@ -330,7 +330,7 @@ const EditEvent = () => {
 330   330     <StyledFormComponentsInRow>
 331   331       <StyledInputWithError className="small">
 332   332         <InputText
333 -         id=""
333 +         id="dormitory"
 334   334           placeholder={event.host.dormitory}
 335   335           value={event.host.dormitory}
 336   336           disabled={true}>
 337   337     </StyledInputWithError>
 338   338     <StyledInputWithError className="small">
 339   339       <InputText
340 -         id=""
340 +         id="roomNumber"
 341   341           placeholder={event.host.roomNumber}
 342   342           value={event.host.roomNumber}
 343   343           disabled={true}>
 344   344     </StyledInputWithError>
 345   345   
```

Alt-Texte bei Bildern:

```
@@ -18,13 +18,14 @@ const GuestListItem: React.FC<{
 18   eventStatus: EventStatus;
 19   onClick?: (e: any) => void;
 20 }> = ({ guest, userIsHost, eventStatus, onClick }) => {
 21 +   const altText = `photo of ${guest.firstName}`;
 22   return (
 23     <StyledGuestListItem>
 24       <StyledImageAndName
 25         onClick={() => router.push(`/profile/${guest.id}`)}>
 26         <StyledImage
 27           alt="Image"
 28           alt={altText}
 29           style={{ objectFit: 'cover' }}
 30           width={60}
 31           height={60}
```

Text bei Links:

```
v ⌂ 6 ━━━━ components/organisms/events/MenuItem.tsx □
  ↑
@@ -29,6 +29,8 @@ export const MenuItem: React.FC<MenuItemProps> = ({

29   29           {dishLink && (
30   30             <a href={dishLink}>
31   31               <StyledLinkIcon />
32 + 32             /* text is required for accessibility */
33 + 33             <FakeLinkText>Dish Link</FakeLinkText>
34   34           </a>
35   35         )}
36   36         {dishDescription && (
37   37           ...
38   38           ...
39   39           ...
40   40           ...
41   41           ...
42   42           ...
43   43           ...
44   44           ...
45   45           ...
46   46           ...
47   47           ...
48   48           ...
49   49           ...
50   50           ...
51   51           ...
52   52           ...
53   53           ...
54   54           ...
55   55           ...
56   56           ...
57   57           ...
58   58           ...
59   59           ...
60   60           ...
61   61           ...
62   62           ...
63   63           ...
64   64           ...
65   65           ...
66   66           ...
67   67           ...
68   68           ...
69   69           ...
70   70           ...
71   71           ...
72   72           ...
73   73           ...
74   74           ...
75   75           ...
76   76           ...
77   77           ...
78   78           ...
79   79           ...
80   80           ...
81   81           ...
82   82           ...
83   83           ...
84   84           ...
85   85           ...
86   86           ...
87   87           ...
88   88           ...
89   89           ...
90   90           ...
91   91           ...
92   92           ...
93   93           ...
94   96           width: 100vw;
95   97           height: 100vh;
96   98           `;

99 + 99           +
100 + 100         const FakeLinkText = styled.p`
101 + 101         display: none;
102 + 102         `;
```

Alle Commits zur Barrierefreiheit sind hier zu finden: <https://github.com/search?q=repo%3Abachelor-project-mmmp%3Fmmmp%3+166&type=commits&p=1>

(7) Ich kann meinen Beitrag zum Datenschutz im Projekt leisten

Ich kenne die Anforderungen der DSGVO, die für Web Projekte relevant sind. Ich kenne die Prinzipien Datenminimierung und Privacy by Design und setze sie schon beim Datenbankentwurf und im gesamten Projekt um.

Beschreiben Sie inwieweit ihr Projekt von der DSGVO betroffen ist, und welche Maßnahmen sie getroffen haben.

Für die Anforderungen der DSGVO haben wir in die Datenschutz-Seite <https://mmp3.vercel.app/privacy> umgesetzt. Für die Erstellung des Inhalts haben wir den „Free Privacy Policy Manager“ verwendet (<https://www.freeprivacypolicy.com/free-privacy-policy-generator/>) Folgender Ausschnitt ist dabei relevant:

In this application the OAuth Login of the University of Applied Sciences Salzburg is used. The following data is transferred automatically:

- *Firstname*
- *Lastname*
- *Email*
- *Username*
- *Status*
- *Studies*
- *Department*
- *Collecting your personal data*

While using this website, we are saving the following personal data:

- *Firstname*
- *Lastname*
- *Email*
- *Dormitory*
- *Room Number*
- *Study*
- *Department*
- *Image**
- *Interests**
- *Phone**
- *Instagram (Username)**

** optional*

Beim ersten Login und somit auch dem Vervollständigen vom Profil muss der User die Datenschutzbestimmungen akzeptieren:

* Required

Firstname*
Kerstin 

Lastname*
Reichinger

Course of studies:
MMT

FH email address:
kreichinger.mmt-b2020@fh-salzburg.ac.at

ⓘ The email address will only be shared with guests

Accommodation* **Room number***

Campus Urstein  123

ⓘ The room number will only be shared with guests

About you
I am Kerstin and I love cooking! I would like to meet new people here! :)

Contact Information

 kersoleynsta

 +43 123 45 67 890

ⓘ The phone number will only be shared with guests

I have read and agree to the [privacy policy](#)

Save

(8) Ich kann meinen Beitrag zur Security des Projektes leisten

Ich kenne die OWASP Top 10 Security Probleme von Web-Applicationen und geeignete Maßnahmen dagegen. Ich kenne für die Technologien, die ich verwende (Repository, Passwörter / API Keys, Authentifizierung, Package Manager, Webserver, DB, Backend, Frontend) die Maßnahmen zur Verhinderung der Sicherheitsprobleme.

Beschreiben Sie die Angriffs-Oberfläche Ihres Projekts. Wo sind Sicherheitsmaßnahmen nötig? Welche Werkzeuge setzen Sie im Projekt ein, um Sicherheitsprobleme zu vermeiden? Ist eine Authentifizierung der User*innen nötig? Falls ja: welche Methode setzen Sie ein? Zeigen Sie Beispiel-Konfiguration (z.B. CORS, Let's Encrypt), Beispiel-Code hier und verweisen Sie auf das Original-Repository.

Cron Jobs

In unserem Projekt werden automatisiert Cron-Jobs für verschiedene Anforderungen ausgeführt. Diese wurden mithilfe von Github-Actions eingerichtet. Um die aufgerufenen Endpunkte zu schützen, wurde ein API-secret-key generiert, welcher in den Environment-Variablen gespeichert wird und nicht für die Öffentlichkeit zugänglich ist. Somit wird sichergestellt, dass niemand „fremdes“ diesen Code ausführen kann.

Beispiel: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/.github/workflows/eventOver.yml>

```
name: eventOver
on:
  schedule:
    - cron: '0 1 * * *'
jobs:
  cron:
    runs-on: ubuntu-latest
    steps:
      - name: Event is over Cron Job
        run: |
          curl --request POST --url "https://mmp3.vercel.app/api/cron/eventOver" --header "Authorization: Bearer ${{ secrets.API_SECRET_KEY }}"
```

Abbildung 4: Cron Job Beispiel

Authentifizierung

Um auf unserer Webseite diverse Daten zu sehen wie zum Beispiel den Events Feed oder verschiedene User-Profile, muss der User eingeloggt sein. Für die Authentifizierung wird der FH-Login verwendet. Dieser wurde mithilfe von de next-auth eingebunden.

Code next-auth API: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/api/auth/%5B...nextauth%5D.tsx>

Um die Seiten (Frontend) mit geschütztem Inhalt vor nicht eingeloggten Usern zu verbergen, wurde eine Middleware angewendet, worin definiert werden kann, welche Pfade der Applikation nur eingeloggt sichtbar sind. Wenn der User nicht eingeloggt ist, wird er direkt zum FH-Login weitergeleitet.

```
export { default } from 'next-auth/middleware';

export const config = {
  matcher: [
    '/events',
    '/events/:path*',
    '/profile/:path*',
    '/my-events',
    '/requests',
  ],
};
```

Abbildung 5: Middleware Next Auth

Code middleware: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/middleware.ts>

```

export const authOptions: NextAuthOptions = {
  providers: [
    [
      idToken: true,
      id: 'fhs',
      name: 'fhs',
      type: 'oauth',
      wellKnown:
        'https://auth.projects.multimediatechnology.at/well-known/openid-configuration',
      clientId: process.env.CLIENT_ID,
      clientSecret: process.env.CLIENT_SECRET,
      async profile(profile, token) {
        const response = await fetch(
          'https://auth.projects.multimediatechnology.at/oauth/userinfo',
          {
            headers: {
              Authorization: `Bearer ${token.access_token}`,
            },
          }
        );
        const fetchedUser = await response.json();
        let user = await prisma.user.findUnique({
          where: {
            email: String(fetchedUser?.email),
          },
        });
        if (!user) {
          try {
            const createdUser = await prisma.user.create({
              data: {
                firstName: fetchedUser.given_name,
                lastName: fetchedUser.family_name,
                study: fetchedUser.studies.split(',')[0],
                dormitory: 'Campus Urstein',
                email: fetchedUser.email,
                image: 'https://firebasestorage.googleapis.com/v0/b/studentenfutter-dba6a.appspot.com/o/profile%2Fdefault-profile.jpg?alt=media&token=578a83b8-ef61-474c-8d9a-e3b59af528f',
              },
            });
            user = createdUser;
          } catch (err) {
            console.log(err);
          }
        }
        return {
          id: profile.sub,
          name: `${fetchedUser.given_name} ${fetchedUser.family_name}`,
          email: fetchedUser.email,
        };
      },
    ],
  ],
  callbacks: {
    async session({ session, token }: any) {
      let user = await prisma.user.findUnique({
        where: {
          email: String(session?.user?.email),
        },
      });
      if (token) {
        session.userId = user.id;
        session.user.firstName = user.firstName;
        session.user.roomNumber = user.roomNumber;
        session.user.image = user.image;
      }
      return session;
    },
  },
};

export default NextAuth(authOptions);

```

Abbildung 6: Next Auth API Endpunkt

Backendseitige Prüfung

Damit die API-Requests auch geschützt sind vor nicht-autorisierten Anfragen wird pro API-Endpunkt die Session mithilfe des Sessionproviders von next-auth geprüft. Wenn keine Session vorhanden ist, wird der http-Code 401 zurückgegeben.

Beispiel Events Feed API Endpunkt: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/api/events/index.ts>

```
import { NextApiRequest, NextApiResponse } from 'next';
import prisma from '../../../../../lib/prisma';
import { getSession } from 'next-auth/react';
import {.getServerSession } from 'next-auth/next';
import { authOptions } from '../auth/[...nextauth]';
import { Prisma } from '_prisma/client';

async function handler(
  req,
  res
) {
  const session = await getSession({ req });

  if (!session) {
    return res.status(401).json({ message: 'Not authenticated' });
  }

  try {
    const event = await prisma.event.create({
      data: {
        title: req.body.title,
        description: req.body.description,
        date: req.body.date,
        location: req.body.location,
        creatorId: session.user.id,
      },
    });

    return res.status(201).json(event);
  } catch (error) {
    console.error(error);
    return res.status(500).json({ message: 'An error occurred while creating the event' });
  }
}

export default handler;
```

Abbildung 7: Backend-seitige Session Prüfung

(9) Ich kann meinen Beitrag im Software-Entwicklungs-Prozess eines Web Projekt leisten

Ich kenne Coding Conventions für HTML, CSS, JavaScript, SQL, PHP, Ruby, und Tools, die die Einhaltung überprüfen und/oder unterstützen. Ich kenne das Vorgehensmodell nach Wasserfall und agile Vorgehensmodelle. Ich kann git in der Teamarbeit im Softwareentwicklungsprozess einsetzen, ich kenne die Konventionen zu Git Commits und zur Arbeit mit Branches. Ich kann development, testing, staging und production-environments sinnvoll einsetzen.

Beschreiben Sie welches Vorgehensmodell und welche Konvention in Ihrem Projekt verwendet werden, welche Tools und Maßnahmen zur Überprüfung und Unterstützung eingesetzt werden.

Naming Conventions

Wir haben uns auf folgende Naming-Conventions geeinigt und im Readme vom Projekt dokumentiert:

```
# Naming Conventions

## Git Commits

- write in present time
- Start with Capital letter
- if ticket available, add number at the end e.g. `#123`

## Branches

- begin with `feature/` or `bugfix/`
- between words add underscores

## Styled Components

- begin with `Styled`
- append with html tag like `Button`


## Comments

- Comments should only describe why you a specific code and not what the code does

## Functions

- naming should be self-descriptive
- camelCase
```

Abbildung 8: Naming Conventions

Arbeiten mit Git

Wir haben uns dazu entschieden, einen dev-Branch und einen main-branch zu verwenden. Der dev-branch wird immer dann in den main-branch gemerged, wenn der dev-Branch neue abgeschlossene Features oder Bugfixes enthält und der neue Stand produktiv deployed werden soll.

Alle neuen Features werden auf Feature-Branches entwickelt, welche vom dev-Branch gezogen werden. Bugfixes werden ebenfalls in Branches gemacht, welche ebenfalls vom dev-Branch gezogen werden. Bei Fertigstellung werden die Branches in den dev Branch gemerged.

Kurz zusammengefasst ist der Ablauf:

1. Branch erstellen
2. Implementierung
3. Testen
4. (optional) kurze Besprechung
5. (optional) Weitergabe des Task an andere Person -> zurück zu Schritt 2
6. Pull Request – Merge Feature Branch in Dev Branch

7. Pull Request – Merge Dev Branch in Main Branch

Deployment Zyklus

Immer wenn der main-Branch einen neuen Commit erhält, also ein Branch gemerged wird, wird automatisch die Deploy-Pipeline von Vercel angestoßen. Diese ist mit dem Github-Repository verknüpft und erkennt solche Änderungen automatisch. Sollen Fehler beim Build passieren, wird automatisch eine Mail an uns Entwicklerinnen ausgeschickt.

Vorgehensmodell

In unserem Projekt verwenden wir das agile Modell, welches ein iteratives Modell ist, bei dem der Entwicklungsprozess in kurze Entwicklungszyklen unterteilt wird, die als Sprints bezeichnet werden. Ein Sprint stellt bei uns hauptsächlich die Studiwochen dar. In jedem Sprint werden bestimmte Funktionalitäten entwickelt, getestet und implementiert. Die Anforderungen können sich während des Prozesses ändern, wodurch es einfacher wird, auf Feedback zu reagieren. Im agilen Modell erhält man regelmäßig Feedback, um sicherzustellen, dass das Produkt den Erwartungen des Kunden entspricht. In unserem Fall wären das die User-Testings worauf wir mit Bugfixes und Änderungen der Anforderungen reagieren.

Wir planen vor jeder Studiwoche, welche Features wir umsetzen wollen und teilen diese den entsprechenden Personen zu. Sollte sich während der Studiwoche herausstellen, dass jemand schneller fertig ist als der andere, werden die Features auch untereinander hin und wieder verschoben. Bugfixes und Änderungen, welche aus dem Usertesting hervorgehen, werden zumeist vor der Studiwoche umgesetzt. Dazu wird ein Meeting mit dem ganzen Projektteam gemacht und das Feedback durchbesprochen. Die daraus resultierenden Tasks werden dann den jeweiligen Personen zugewiesen, welche diese dann selbstständig umsetzen.

(10) Ich kann meinen Beitrag zu Web Operations leisten

Ich kenne verschiedene Methoden, um Web-Applikationen zu hosten. Ich kann Apache und Nginx auf UNIX Server konfigurieren, kann einen Load Balancer konfigurieren. Ich kann Rails und Node.js Projekte auf einem PaaS wie Heroku oder Dokku deployen. Ich kenne die 12 Faktoren der [12 factor app](#).

Beschreiben Sie wie Ihr Projekt deployed wird.

Datenbank auf Heroku

Da wir PostgreSQL als Datenbank verwenden, entschieden wir uns beim Hosting für Heroku. Mit der Studenten-Lizenz können wir kostenlos (jedoch limitiert) die benötigten Funktionen nutzen, welche aber ausreichend sind.

Um eine Verbindung mit Heroku herzustellen, muss man sich über die Heroku CLI einloggen und das „git remote“ hinzufügen. Danach kann man jederzeit die Änderungen pushen.

```
### Production migrations

First setup Heroku connection:

- for migrations in production login in terminal to heroku `heroku login`
- add git remote to heroku `heroku git:remote --app teamspaghetti`

After pushing changes to master branch, run `git push heroku HEAD:master`
```

Für die erfolgreiche Ausführung des Builds auf Heroku muss auch ein Procfile im Projekt-Root erstellt werden. Darin wird angegeben, welche Befehle beim Release ausgeführt werden müssen:

```
# Procfile
1 release: npx prisma migrate deploy
2 |
```

Next.js App auf Vercel

Die Next.js Applikation ist auf Vercel gehostet. Dazu wurde im Vercel Projekt das Github Repository angegeben, um einen production-Build automatisch anzustoßen, sobald ein neuer Main-Branch Stand vorhanden ist.

The screenshot shows the Vercel settings interface. At the top, under "Connected Git Repository", it says "Seamlessly create Deployments for any commits pushed to your Git repository." It lists a connected repository "bachelor-project-mmp3/mmp3" from 156d ago, with a "Disconnect" button. Below this, there's a link "Learn more about Vercel for Git". Under "Production Branch", it says "By default, every commit pushed to the `main` branch will trigger a Production Deployment instead of the usual Preview Deployment. You can switch to a different branch here." A "Branch Name" input field contains "master", and a "Save" button is at the bottom right. There's also a link "Learn more about Production Branch".

Zusätzlich werden sogenannte Preview-Deployments automatisch deployed sobald ein neuer Branch gepusht wird.

Deployments

All Branches...		Select Date Range	All Environments	Status 4/5
mmp3-dtb18fna-spagettigirls.vercel.app	Production (Current)	● Ready 33s	↳ master -o 2039301 Merge pull request #148 from bac...	5d ago by kerstin97 🌐
mmp3-e4qj7dtxm-spagettigirls.vercel.app	Preview	● Ready 33s	↳ dev -o 217150c Merge pull request #147 from bac...	5d ago by kerstin97 🌐
mmp3-fzp5hnd9q-spagettigirls.vercel.app	Preview	● Ready 47s	↳ feature/landingpage-content -o edd7cf3 Implement Landingpage content ...	5d ago by kerstin97 🌐
mmp3-4y06wzbuv-spagettigirls.vercel.app	Production	● Ready 1m 28s	⟳ Redeploy of i42lt6fdz	6d ago by kerstin97 🌐
mmp3-iyOp4otdv-spagettigirls.vercel.app	Preview	● Ready 1m 27s	⟳ Redeploy of oty1et21s	6d ago by kerstin97 🌐
mmp3-i42lt6fdz-spagettigirls.vercel.app	Production	● Error 17s	↳ master -o 0120bac Merge pull request #145 from bac...	6d ago by kerstin97 🌐
mmp3-oty1et21s-spagettigirls.vercel.app	Preview	● Error 17s	↳ dev -o 989a49c Fix next build	6d ago by kerstin97 🌐

(11) Tanja kennt die Arbeitsteilung in Web-Projekten

Ich kann ein Web-Projekt allein umsetzen oder in verschiedenen Rollen einem Team mitwirken: Web Operations, Backend, Frontend, Web-Design. Ich kenne meine Stärken.

Beschreiben Sie Ihre Rolle und Ihren Verantwortungsbereich im Projekt. Beschreiben Sie anhand einer konkreten User Story wie der Arbeitsablauf im Projekt ist, an welcher Stelle Sie übernehmen bzw. an andere KollegInnen übergeben.

Im Projekt nehme ich verschiedene Rollen ein, darunter fällt die Konzeption, die Backend-Entwicklung sowie die Frontend-Entwicklung. Im Bereich Konzeption fällt die gesamte Planung des Projekts an. Hierbei bringe ich meine Meinung ein, mache Vorschläge und führe Recherchen durch. Im Bereich Backend-Entwicklung sorge ich für eine reibungslose Funktionalität der Anwendung und Features. Meine Schwerpunkte liegen dabei auf die Implementierung eines Foto-Uploads, Logins sowie die Erstellung eines Profils. Im Frontend-Bereich fällt die Gestaltung der Benutzeroberfläche zu dem jeweiligen zugewiesenen Feature an.

Bevor das Projekt in die Implementierungsphase überging, haben wir als Team gemeinsam die Kernfeatures definiert und User Stories erarbeitet, welche wir in unserem Kanban-Board festgehalten haben. Vor Beginn jeder Studiwoche wurde entschieden, welche Features in dieser Woche umgesetzt werden sollten. Anhand der Implementierung des Features "Notifications" möchte ich nun den Arbeitsablauf näher erläutern:

Das Feature „Notifications“ dient dazu, Benutzer*innen über wichtige Neuigkeiten und Änderungen innerhalb der Anwendung zu informieren. Dafür gibt es einen eigenen Bereich auf der „My Events“ Seite. Beispiele einer Benachrichtigung sind: „Das Event XY wurde abgesagt“, „Reminder: Event XY findet morgen statt“, „Das Event wurde bearbeitet“ etc.

Während Kerstin sich um die Backend-Umsetzung kümmerte, konzentrierte ich mich auf die Frontend-Implementierung. Bevor mit der Implementierung gestartet werden konnte, musste ein neuer Feature Branch in Github erstellt werden. Meine Aufgabe bestand darin, eine Komponente für die Notifications zu erstellen und diese nach dem vorgegebenen Design zu gestalten. Nach Abschluss meiner Arbeit (inkl. lokalem Testen) konnte ich den Task an Kerstin übergeben. Bevor Kerstin mit ihrer Umsetzung begann, hatten wir noch ein kurzes Meeting, in dem wir kurz meine Implementierung der Komponente besprochen haben. Unter anderem sind wir auch nochmal alle möglichen Szenarien der Notifications gemeinsam durchgegangen und haben definiert wann welche Notification ausgesteuert werden soll, da die Logik recht komplex ist. Nach dem Meeting konnte Kerstin mit ihrer Umsetzung starten und die Logik implementieren. Anschließend folgte wieder lokales Testen und danach konnte ein Pull-Request erstellt werden und schlussendlich vom Feature-Branch in den Dev-Branch gemerged werden. Der Task konnte dann im Kanban-Board in die Spalte „Done“ verschoben werden.

Zu einem späteren Zeitpunkt wurde der Dev-Branch in den Main-Branch gemerged und somit das Feature erfolgreich auf der Produktionsumgebung live geschaltet.

Wenn eine User-Story nur für eine Person relevant ist, bleibt der Ablauf gleich, jedoch entfällt die Notwendigkeit, den Task an eine weitere Person weiterzugeben.

Kurz zusammengefasst ist der Ablauf:

8. Branch erstellen
9. Implementierung
10. Testen
11. (optional) kurze Besprechung
12. (optional) Weitergabe des Task an andere Person -> zurück zu Schritt 2
13. Pull Request – Merge Feature Branch in Dev Branch
14. Pull Request – Merge Dev Branch in Main Branch

(12) Tanja kann User Stories implementieren und deployen

Ich kann bei der Ausformulierung von User Stories mitarbeiten, um alle zur Implementierung nötigen Fragen zu klären. Ich kann den nötigen Programmcode in HTML, CSS, Javascript, SQL, PHP, Ruby schreiben, der die User Story umsetzt. Ich kann die nötigen Schritte durchführen um die fertig implementierte User Story in production zu deployen.

- a) Listen Sie 5 User Stories auf, die sie allein umgesetzt haben oder zu denen Sie den größten Beitrag geleistet haben
- b) Wählen Sie eine der User Story aus, Zeigen Sie die User Story, die Implementierung (Codebeispiel hier und Verweis auf des Original-Repository) und das fertige Feature im Projekt (Verweis auf production-Server)

User Story 1 – Profile Page:

As a user, I want to see my name, my hosted events and the one's I attended, my personal information, my profile picture, my location and my interests on my profile page.

User Story 2 – Walkthrough (mobile):

As a user, I want to get a walk through after my first log in, in order to know, how the app works.

User Story 3 – Landingpage:

As a user, I want to access a page that provides me with all relevant information about the app. Additionally, I would like to have the option to log in on that page.

User Story 4 – Login:

As a user, I want to click on a button to get to the FH log in page. I want to sign in with my FH credentials via the Authorization page and get sent back to my personal profile.

If I sign up for the first, I want to enter my personal information, press a button to save them and get redirected to my personal profile.

User Story 5 – Edit profile:

As a user, I want to be able to edit my profile. I want to change my location, my profile picture, my interests and my personal information. I want to click on a button to edit all this information in a form, save it and see all the updated information in my profile page.

Login:

Verwendung OAUTH Login FH-Salzburg – Implementierung mit NextAuth

Login/Logout Button:

1. Status wird überprüft, ob User*in authentifiziert ist
2. Funktionen signIn und signOut werden von NextAuth bereit gestellt

Link: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/components/organisms/LandingpageHeader.tsx>

```
{status != 'authenticated' ? (
  <StyledLog>
    |> onClick={() =>
      signIn('fhs', {
        |> callbackUrl: '/api/auth/signin',
      })
    }
  >>
  Log in
</StyledLog>
) : (
  <StyledLog>
    |> onClick={() => signOut({ callbackUrl: '/' })}
    Log out
</StyledLog>
)}
```

Konfiguration:

1. Zuerst wird der Provider definiert
2. Abfrage, ob User*in bereits in der Datenbank vorhanden ist
 - a. Wenn nicht, User*in wird erstellt
3. Der Session werden noch Daten wie UserId, Vorname, etc mitgegeben

Link: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/api/auth/%5B...nextauth%5D.tsx>

```
import NextAuth, { NextAuthOptions } from 'next-auth';
import prisma from '../../../../../lib/prisma';

// For more information on each option (and a full list of options) go to
// https://next-auth.js.org/configuration/options

export const authOptions: NextAuthOptions = {
    providers: [
        {
            idToken: true,
            id: 'fhs',
            name: 'fhs',
            type: 'oauth',
            wellKnown: 'https://auth.projects.multimediatechnology.at/.well-known/openid-configuration',
            clientId: process.env.CLIENT_ID,
            clientSecret: process.env.CLIENT_SECRET,

            async profile(profile, token) {
                const response = await fetch(`https://auth.projects.multimediatechnology.at/oauth/userinfo`, {
                    headers: {
                        Authorization: `Bearer ${token.access_token}`,
                    },
                });
                const fetchedUser = await response.json();

                let user = await prisma.user.findUnique({
                    where: {
                        email: String(fetchedUser?.email),
                    },
                });

                if (!user) {
                    try {
                        const createdUser = await prisma.user.create({
                            data: [
                                {
                                    firstName: fetchedUser.given_name,
                                    lastName: fetchedUser.family_name,
                                    study: fetchedUser.studies.split('-')[0],
                                    dormitory: 'Campus Urstein',
                                    email: fetchedUser.email,
                                    image: 'https://firebasestorage.googleapis.com/v0/b/studentenfutter-dba6a.appspot.com/o/avatars%2Fplaceholder.png?alt=media&token=1683400000000'
                                },
                            ],
                            user: createdUser,
                        });
                        catch (err) {
                            console.log(err);
                        }
                    }
                    return {
                        id: profile.id,
                        name: `${fetchedUser.given_name} ${fetchedUser.family_name}`,
                        email: fetchedUser.email,
                    };
                }
            },
        },
    ],
    callbacks: {
        async session({ session, token }: any) {
            let user = await prisma.user.findUnique({
                where: {
                    email: String(session.user?.email),
                },
            });

            if (token) {
                session.user.userId = user.id;
                session.user.firstName = user.firstName;
                session.user.roomNumber = user.roomNumber;
                session.user.image = user.image;
            }
            return session;
        },
    },
};

export default NextAuth(authOptions);
```

Profil vervollständigen (first login):

- 1) Schema für die Validierung des Formulars erstellen
 - 2) Diverse Formularfelder erstellen (mit State arbeiten, damit Feld vorausgefüllt ist)
 - 3) Foto-Upload (Firestore)
 - 4) Daten an die Datenbank senden (HTTP Request)

Link (Profileformular): <https://github.com/bachelor-project-mmfp3/mmfp3/blob/dev/components/organisms/forms/ProfileForm.tsx>

Screenshots für das Formular würde den Rahmen sprengen

Link (Fotoupload): <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/components/organisms/forms/ProfileForm.tsx>

```
import { storage } from '../firebaseConfig';
import imageCompression from 'browser-image-compression';
import {
  ref,
  uploadBytes as upload,
  getDownloadURL as getUrl,
} from 'firebase/storage';
import { v4 as randomId } from 'uuid';

export async function uploadImage(image, uploadFolder) {
  if (image.size > 2100000) {
    const options = { maxSizeMB: 2 };
    image = await imageCompression(image, options);
  }

  return new Promise(function (resolve) {
    const fileName = `${image.name}_${randomId()}`;
    const imageRef = ref(storage, `${uploadFolder}/${fileName}`);

    upload(imageRef, image).then((snapshot) => {
      getUrl(snapshot.ref).then((url) => {
        resolve(url);
      });
    });
  });
}
```

Link (API Endpoint): <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/components/organisms/forms/ProfileForm.tsx>

```
try {
  // PATCH /api/profile/${id}
  if (req.method === 'PATCH') {
    try {
      const {
        imageUrl,
        firstName,
        lastName,
        dormitory,
        roomNumber,
        aboutYou,
        instagram,
        phone,
      } = req.body;

      const session = await getSession({ req });
      const userId = session?.user?.userId;

      const result = await prisma.user.update({
        where: {
          id: userId,
        },
        data: {
          image: imageUrl,
          firstName: firstName,
          lastName: lastName,
          dormitory: dormitory ? dormitory : 'Campus Urstein',
          roomNumber: roomNumber,
          interests: aboutYou,
          phone: phone,
          instagram: instagram,
        },
      });

      res.json(result);
    } catch (err) {
      res.status(500).json({ message: err.message });
    }
  }
}
```

Link Production: <https://mmp3.vercel.app/>

(13) Tanja kann Fehler im Programm (Bugs) finden und beheben

Ich kann einen Bug, den ich gefunden habe, so beschreiben, dass eine Suche nach den Ursachen möglichst einfach wird. Ich kann die Ursache eines Bugs in HTML, CSS, Javascript, SQL, PHP, Ruby Code mittels Developer Tools und Log Messages finden und beheben.

Wählen Sie einen Beispiel-Bug aus dem Projekt aus, und beschreiben Sie: wie wurde der Bug entdeckt, wie haben Sie die Ursache gefunden, wie haben Sie den Bug behoben.

Folgefrage: gibt es Regressions-Test die ein Wiederauftreten des Bugs entdecken würden?

Die meisten Bugs werden von Benutzer*innen aus dem User Testing gefunden. Wenn ich selbst einen Bug finde, passiert das aktuell durch manuelles Testen meiner Funktionen, da noch keine Testfälle implementiert sind.

Beispiel Bug:

Fotos konnten nicht hochgeladen werden, weil das Größenlimit zu sehr eingeschränkt war (2MB). Insbesondere bei der Verwendung der Funktion „Foto aufnehmen“ auf mobilen Geräten kam es dadurch zu Schwierigkeiten.

Um den Bug zu beheben, habe ich ein npm-package namens „Browser Image Compression“ verwendet. Dadurch wird das Bild vor dem Hochladen komprimiert. Dank dieser Lösung muss der*die Benutzer*in sich nicht mehr um die Komprimierung kümmern und kann jede beliebige Größe hochladen.

Dafür muss nur die Funktion „imageCompression“ importiert und Optionen (z.B: maxSizeMB: 2) definiert werden. In der Funktion „imageCompression“ übergibt man als Parameter das Foto und die definierten Optionen.

```
import { storage } from '../firebaseConfig';
import imageCompression from 'browser-image-compression';
import {
  ref,
  uploadBytes as upload,
  getDownloadURL as getUrl,
} from 'firebase/storage';
import { v4 as randomId } from 'uuid';

export async function uploadImage(image, uploadFolder) {
  if (image.size > 2100000) {
    const options = { maxSizeMB: 2 };
    image = await imageCompression(image, options);
  }

  return new Promise(function (resolve) {
    const fileName = `${image.name}_${randomId()}`;
    const imageRef = ref(storage, `${uploadFolder}/${fileName}`);

    upload(imageRef, image).then((snapshot) => {
      getUrl(snapshot.ref).then((url) => {
        resolve(url);
      });
    });
  });
}
```

(14) Tanja kann Unit-Tests und End-to-End Tests schreiben

Ich kenne Test-frameworks für die Technologien, die ich einsetze, und kann sowohl Unit Tests als auch End-to-End Tests schreiben. Ich kann Test Driven Development einsetzen und ich kann Tests für bestehenden Code schreiben. Ich verstehe welchen Teil des Projekts mein Test testet.

a) Wählen Sie einen Unit Test aus dem Projekt den Sie entweder alleine geschrieben haben, oder an dessen Umsetzung sie wesentlich beteiligt waren. Was wird in diesem Test getestet, warum, wie? Zeigen Sie Code Beispiele hier und geben Sie Links zum Commit im Original Repository an.

Der UNIT-Test prüft, ob die Komponente „SmallEventPreview“ korrekt gerendert wird und die folgenden Inhalte angezeigt werden: Titel des Events, Foto des Events, Datum, Foto des Hosts, Bewertungssterne

File: https://github.com/bachelor-project-mmp3/mmp3/blob/dev/__tests__/smallEventPreview.jsx

```
57     describe('SmallEventPreview', () => {
58       const event = {
59         title: 'Suppen Event',
60         imageEvent: '/event-image.jpg',
61         imageHost: '/host-image.jpg',
62         date: '2023-12-10',
63         reviews: [{ total: 5 }],
64       };
65
66       it('renders correctly with all information', () => {
67         render(
68           <ThemeProvider theme={theme}>
69             <SmallEventPreview {...event} />
70           </ThemeProvider>
71         );
72
73         const eventTitle = screen.getByText('Suppen Event');
74         const eventImage = screen.getByAltText('photo of Suppen Event');
75         const date = screen.getByText('12/10/2023');
76         const hostImage = screen.getByAltText('photo of host');
77         const reviewStars = screen.getAllByText('*');
78
79         expect(eventTitle).toBeInTheDocument();
80         expect(eventImage).toBeInTheDocument();
81         expect(date).toBeInTheDocument();
82         expect(hostImage).toBeInTheDocument();
83         expect(reviewStars.length).toBe(5);
84       });
85     });

```

b) Wählen Sie einen End-to-End Test aus dem Projekt den Sie entweder alleine geschrieben haben, oder an dessen Umsetzung sie wesentlich beteiligt waren. Was wird in diesem Test getestet, warum, wie? Zeigen Sie Code Beispiele hier und geben Sie Links zum Commit im Original Repository an.

Der E2E-Test prüft, ob sich ein*e User*in einloggen und anschließend die Seite „All Events“ aufrufen kann. Auf der „All Events“ Seite wird überprüft, ob die H1 „Find an event to join“ ersichtlich ist.

File: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/cypress/e2e/login.cy.js>

 santnerin Add e2e test login #157 ✓

Code Blame 15 lines (13 loc) · 640 Bytes

```
describe('Multimedia login', () => {
  it('should successfully log in a user and user can access all events page', () => {
    cy.visit('https://localhost:3001/');
    cy.contains("Let's get started").should('be.visible').click();
    technology.at', () => {
      cy.get('input[name="username"]').type(Cypress.env('username'));
      cy.get('input[name="password"]').type(Cypress.env('password'));
      cy.contains('Log In').click();
    });
    cy.contains('Show all events').click();
    cy.contains('Find an event to join').should('be.visible');
  });
});
```

(15) Tanja kann ein Refactoring durchführen

Ich kenne allgemeine Code Smells und spezifische Code Smells für Ruby und JavaScript. Ich kenne Refactorings, um diese Code Smells zu beheben. Ich kann ein Refactoring durchführen, das nur den Code verbessert, ohne neue Funktionalität zu implementieren. Ich kann die Rolle von Tests beim Refactoring erklären.

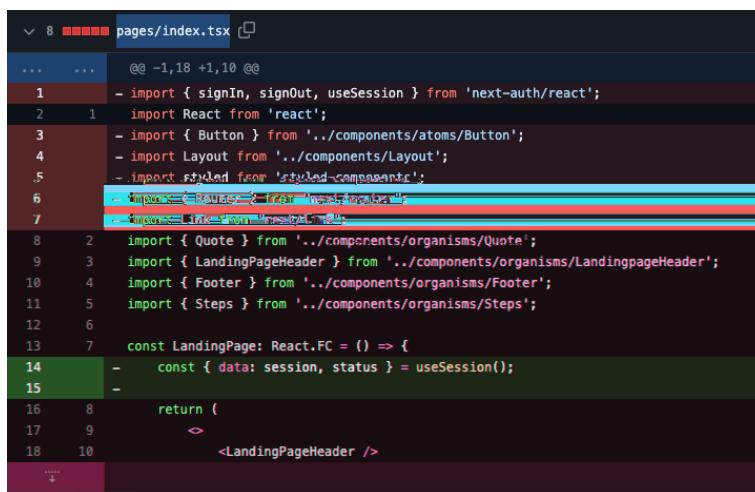
Wählen Sie einen Beispiel-Refactoring aus, das Sie durchgeführt haben. Beschreiben Sie: warum wurde das Refactoring durchgeführt? Wie sind Sie dabei vorgegangen? Hat das Refactoring den gewünschten Erfolg erzielt?

- Bei den Bewertungssternen wurde die Farbe hardcoded hinterlegt. Da die Bewertungssterne an mehreren Stellen im Code und in unterschiedlichen Files vorkommen, ist dieser Ansatz nicht optimal. Wenn eine Farbänderung durchgeführt werden muss, dann ist das mühsam und aufwendig die Farbe auszutauschen. In der ThemeConfig gibt es definierte Variablen für Farben. Der Hex-Code wurde deshalb durch die Variable aus der ThemeConfig ausgetauscht – ein Beispiel ist im folgenden Screenshot ersichtlich.

Commit: <https://github.com/bachelor-project-mmp3/mmp3/commit/44200ca7123a956535544587f4364197ee1c05a7>

- In der index.tsx wurden überflüssige Imports und Konstanten entfernt. Diese sind wahrscheinlich durch Copy-Paste-Fehler dort gelandet oder wurden ausgelagert, obwohl sie für Teile der Seite benötigt wurden.

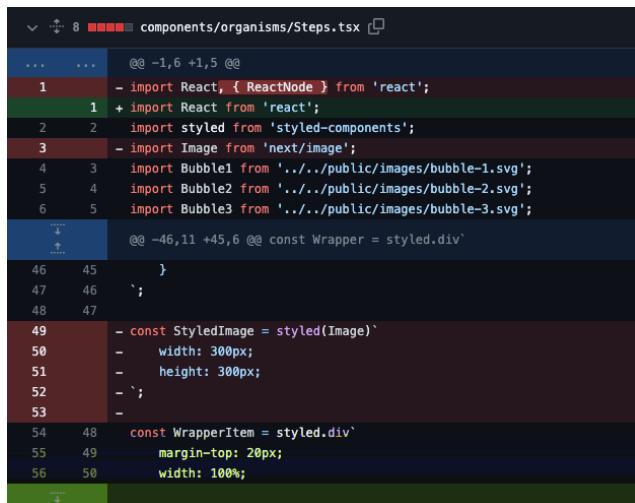
Link: <https://github.com/bachelor-project-mmp3/mmp3/commit/1d9128b3a1dda9208aa360cda829817bc57138a1>



```
@@ -1,18 +1,10 @@
1   - import { signIn, signOut, useSession } from 'next-auth/react';
2   1     import React from 'react';
3   3     - import { Button } from '../components/atoms/Button';
4   4     - import Layout from '../components/Layout';
5   5     - import styled from 'styled-components';
6   6     + import { Button } from 'next-auth/react';
7   7     + import { Layout } from 'next-auth/react';

8   2     import { Quote } from '../components/organisms/Quote';
9   3     import { LandingPageHeader } from '../components/organisms/LandingpageHeader';
10  4     import { Footer } from '../components/organisms/Footer';
11  5     import { Steps } from '../components/organisms/Steps';
12  6
13  7   const LandingPage: React.FC = () => {
14  14     -   const { data: session, status } = useSession();
15  15
16  16     8     return (
17  17       <>
18  18         <LandingPageHeader />
```

Link: <https://github.com/bachelor-project-mmp3/mmp3/commit/956093e567e835672006e73621df385c5215630d>



```
@@ -1,6 +1,5 @@
1   - import React, { ReactNode } from 'react';
2   1     + import React from 'react';
3   2     import styled from 'styled-components';
4   3     - import Image from 'next/image';
5   4     import Bubble1 from '../../../../../public/images/bubble-1.svg';
6   5     import Bubble2 from '../../../../../public/images/bubble-2.svg';
7   6     import Bubble3 from '../../../../../public/images/bubble-3.svg';

@@ -46,11 +45,6 @@ const Wrapper = styled.div`
```

(16) Lisa kennt die Arbeitsteilung in Web-Projekten

Ich kann ein Web-Projekt allein umsetzen oder in verschiedenen Rollen einem Team mitwirken: Web Operations, Backend, Frontend, Web-Design. Ich kenne meine Stärken.

Beschreiben Sie Ihre Rolle und Ihren Verantwortungsbereich im Projekt. Beschreiben Sie anhand einer konkreten User Story wie der Arbeitsablauf im Projekt ist, an welcher Stelle Sie übernehmen bzw. an andere KollegInnen übergebenen.

Unser Projekt wurde von 3 Entwicklerinnen implementiert und unsere Arbeitsteilung war gleichmäßig aufgeteilt, somit hat jede Person im Frontend genauso wie im Backend einen Beitrag zum Code geleistet. Meine Hauptaufgaben waren die Erstellung und die Bearbeitung von Events durch Formulare, welche React Hook Validierung mit einem Yup Schema verwendet.

Bei uns war eigentlich alles so aufgeteilt, dass jeder eine User Story von Anfang bis zum Ende implementiert hat. Wir haben uns gemeinsam ein Schema und eine Projektorganisation erstellt und jeder hat mit diesem gearbeitet.

The screenshot shows a GitHub issue page for a project named 'mmp3'. The issue is titled 'Create a new event #18' and is labeled as 'Open' by 'LisMaRad' on November 21, 2022. The user story in the description reads:

As a user, I want to create an event with the click on a button in the feed or in the "Your events" page. I want to see a form where I can add all of the following information:

- name
- the date and the time of the event
- price for the guests
- the location where the event is happening
- the area where the event is happening
- the dish
- link to the recipe (optional)
- the ingredients
- maximum of attending people
- time limit to join the event and leave the event
- upload an inviting picture
- price
- additional information

Below the user story, there is a note: 'I want to click on a button and see the event in the feed with all events and in my upcoming events as a host.'

The right side of the screenshot shows the issue's configuration fields:

Assignees	LisMaRad
Labels	Add labels...
Milestone	Add milestone...
Status	<input checked="" type="checkbox"/> Done
Linked pull requests	No linked pull requests
Repository	mmp3
Priority	Choose an option...
Size	Choose an option...
Packages	Create new event
Role	Host
Studio Woche	Anfang März

Somit habe ich bei diesem Event das File erstellt und im Frontend mit Hilfe von Styled Components das Formular gestyled und implementiert. Anschließend habe ich eine Validierung hinzugefügt, welche nach dem Abschicken des Formulars den Input nochmal checkt, es wird aber auch während der Eingabe des Users der Input bereits gecheckt, ob er passend ist. Am Ende habe ich den passenden API Call erstellt um einen Eintrag in die Datenbank zu ermöglichen.

(17) Lisa kann User Stories implementieren und deployen

Ich kann bei der Ausformulierung von User Stories mitarbeiten, um alle zur Implementierung nötigen Fragen zu klären. Ich kann den nötigen Programmcode in HTML, CSS, Javascript, SQL, PHP, Ruby schreiben, der die User Story umsetzt. Ich kann die nötigen Schritte durchführen um die fertig implementierte User Story in production zu deployen.

- Listen Sie 5 User Stories auf, die sie allein umgesetzt haben oder zu denen Sie den größten Beitrag geleistet haben
- Wählen Sie eine der User Story aus, Zeigen Sie die User Story, die Implementierung (Codebeispiel hier und Verweis auf des Original-Repository) und das fertige Feature im Projekt (Verweis auf production-Server)

The screenshot shows a list of five user stories from a ticket management system:

- mmp3 #31**: Kick the guest from an event when already accepted. Status: Host, Due: Ende März.
- mmp3 #19**: Cancel my event as host. Status: Host, Due: Ende März.
- mmp3 #25**: Edit event as host. Status: Host, Due: Anfang März.
- mmp3 #18**: Create a new event. Status: Host, Due: Anfang März.
- mmp3 #11**: Infoview of event. Status: Guest, Due: Anfang März.
- mmp3 #26**: Receive edit notification email as guest. Status: Guest, Due: Anfang März.

Ticket # 18 Create a new event:

In diesem Feature geht es darum, dass der User/die Userin ein neues Event erstellen möchte und folgende Informationen in einem Formular eingeben möchte:

- Titel des Events
- Das Datum und die Zeit wann das Event stattfindet
- Ein Datum und eine Uhrzeit bis wann die Gäste Teilnahmeanfragen schicken können
- Die Kosten pro Person
- Die Anzahl der Gäste
- Eine kurze Information zum Event
- Das Menü, welches aus einer unbegrenzten Anzahl an Gerichten bestehen kann
- Für jedes Gericht soll man einen Titel eingeben können
- Man kann einen Link für das Rezept oder ähnliche Infos zum Gericht einfügen können
- Jedes Gericht kann eine kurze Beschreibung bekommen

Die Gerichte sollen vermehrt oder auch verringert werden können.

Anhand von einem YUP Schema und das React Hook Package wird die Eingabe der User validiert und kann somit nicht gespeichert werden wenn Fehler vorkommen und die Fehlermeldungen werden angezeigt. Gleichzeitig wird auch während der Eingabe anhand der OnChange Funktion gecheckt, ob die Einträge denn valide sind.

Während der Input nicht valide ist, wird der Create event button disabled. Somit kann der User/die Userin keine unpassenden Informationen speichern.

```
const schema = yup
  .object({
    title: yup.string().min(3).required(),
    date: yup.string().required(),
    timelimit: yup.string().required(),
    costs: yup.number().positive().min(0).max(99),
    guests: yup.number().positive().integer().min(1).max(99).required(),
  })
  .required();
type FormData = yup.InferType<typeof schema>;
```

Yup Schema:

```
const {
  register,
  handleSubmit,
  setValue,
  setError,
  clearErrors,
  formState: { errors },
} = useForm<FormData>({
  resolver: yupResolver(schema),
});

React.useEffect(() => {
  register('title');
  register('date');
  register('timelimit');
  register('costs');
  register('guests');

  if (session) {
    fetch(`/api/profile/${session?.user?.userId}`, {
      method: 'GET',
    })
    .then((res) => res.json())
    .then((data) => {
      data.profile.roomNumber; setRoomnumber(data.profile.roomNumber);
      setLoading(false)
    });
  }
}, [register, session]);
```

Wir nutzen React hook, wofür jedes Feld zuerst registriert werden muss, um schlussendlich validiert zu werden. Da wir styled Components verwenden, müssen die Felder im useEffect registriert werden um eine erfolgreiche Validierung zu erhalten. Es muss auch der User gefetcht werden um die Daten zu bekommen, wo der User/die Userin die Veranstaltung abhalten wird.

```

const CheckTimelimitInputTime = (e) => {
  const inputTimelimit = new Date(e);
  const eventDate = new Date(date);

  if (inputTimelimit <= currentDate || inputTimelimit >= eventDate) {
    setError('timelimit', { type: 'min' });
  } else {
    if (errors.timelimit) {
      clearErrors('timelimit');
    }
  }
};

```

Hilfsfunktion um zu versichern, das der Zeitpunkt für das Zeitlimit der Anfragen sich zwischen jetzt und dem Zeitpunkt der Veranstaltung befindet.

```

</StyledInputWithError>
<StyledInputWithError>
  <InputDateTime
    id="timelimit"
    value={timeLimit}
    min={dateTimePlusOneHour}
    max={date}
    onChange={({e) => {
      setValue('timelimit', e.target.value);
      setTimeLimit(e?.target?.value);
      CheckTimelimitInputTime(e?.target?.value);
    })}
    isInvalid={errors.timelimit ? 'true' : 'false'}
    required>
    Time limit to receive join requests until*
  </InputDateTime>
  {errors.timelimit && errors.timelimit.type === 'min' && (
    <ErrorMessage>
      Please enter a date and time between today and the
      event
    </ErrorMessage>
  )}
</StyledInputWithError>

```

Inputfeld für den Zeitpunkt bis wann die Gäste eine Anfrage schicken können für die Teilnahme.

Wir nutzen React hook, wofür jedes Feld zuerst registriert werden muss, um schlussendlich validiert zu werden. Da wir styled Components verwenden, müssen die Felder im useEffect registriert werden um eine erfolgreiche Validierung zu erhalten. Es muss auch der User gefetcht werden um die Daten zu bekommen, wo der User/die Userin die Veranstaltung abhalten wird.

Feature im Repo: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/pages/events/create.tsx>

Feature in Production: <https://mmp3.vercel.app/events/create>

(18) Lisa kann Fehler im Programm (Bugs) finden und beheben

Ich kann einen Bug, den ich gefunden habe, so beschreiben, dass eine Suche nach den Ursachen möglichst einfach wird. Ich kann die Ursache eines Bugs in HTML, CSS, Javascript, SQL, PHP, Ruby Code mittels Developer Tools und Log Messages finden und beheben.

Wählen Sie einen Beispiel-Bug aus dem Projekt aus, und beschreiben Sie: wie wurde der Bug entdeckt, wie haben Sie die Ursache gefunden, wie haben Sie den Bug behoben.

Folgefrage: gibt es Regressions-Test die ein Wiederauftreten des Bugs entdecken würden?

Bug: im Header wird auf einigen Seiten ein „Go back“ Button eingefügt. Dieser ist in einer eigenen Component und wird nur importiert in die anderen Components. Dieser „Go back“ Button nutzt den React Router und somit wird auf den Router zugegriffen und er soll einfach auf die letzte Seite zurückbringen. Das Problem jedoch war, dass der Router immer einen Fehler geworfen hat, dass er nicht zurück routen kann. Wenn die Seite jedoch neu geladen wurde, hat es funktioniert. Bevor das Ganze in die Compononente ausgelagert wurde hat es noch funktioniert.

Vorgehen: Ich habe einen ganzen Vormittag damit verbracht diesen Bug zu lösen und unendlich viel gegoogelt. Ich habe sogar zwischendurch an etwas anderem gearbeitet um etwas Distanz zu dem Problem zu bekommen um vielleicht eine andere Lösung zu finden. Schlussendlich, nachdem wir sogar zu zweit nach dem Fehler und einer möglichen Lösung gesucht haben, haben wir beschlossen, den Seiteninhalt zu löschen und dann soweit die Components einzufügen, bis der Fehler auftritt. Schon nach kurzer Zeit habe ich gemerkt, dass es an der Header Component liegt und habe den Fehler auch nach vielen Stunden der Verzweiflung gefunden. Es handelte sich um einen einfachen falschen Import der Routers, welcher nicht der React Router sondern der Client Router war. Somit hat die Componente nicht auf den richtigen Router zugegriffen und deswegen auch keine vorigen Seitenbesuche gespeichert gehabt. Behoben habe ich ihn dann durch das Einfügen des richtigen Routers ☺

(19) Lisa kann Unit-Tests und End-to-End Tests schreiben

Ich kenne Test-frameworks für die Technologien, die ich einsetze, und kann sowohl Unit Tests als auch End-to-End Tests schreiben. Ich kann Test Driven Development einsetzen und ich kann Tests für bestehenden Code schreiben. Ich verstehe welchen Teil des Projekts mein Test testet.

a) Wählen Sie einen Unit Test aus dem Projekt den Sie entweder alleine geschrieben haben, oder an dessen Umsetzung sie wesentlich beteiligt waren. Was wird in diesem Test getestet, warum, wie? Zeigen Sie Code Beispiele hier und geben Sie Links zum Commit im Original Repository an.

Mein Unittest testet das Tooltip, ob es richtig im geöffneten und im geschlossenen Zustand angezeigt wird. Die Tooltip Komponente ist eine Styled Componenten, welche einen Parameter von außen für den Zustand bekommen hat. In der Komponente wo das Tooltip aufgerufen wird, gibt es einen Bool State und je nachdem ob der User es öffnet oder schließt ist dieser State true oder false.

Um eben zu prüfen, ob der Tooltip content angezeigt wird, gebe ich einem das Property open als true und einmal als false mit. Danach test ich, ob der Text im Document ist oder nicht.

```
± fhs45899
describe( title: 'ToolTip', fn: () => {
  it( title: 'renders ToolTip open correctly with information', config: () => {
    render(
      <ThemeProvider theme={theme}>
        <ToolTip open>Zucchini, Tomaten, Melanzani</ToolTip>
      </ThemeProvider>
    );
    const children = screen.getByText('Zucchini, Tomaten, Melanzani');
    expect(children).toBeInTheDocument();
  });
});

± fhs45899
describe( title: 'ToolTip', fn: () => {
  it( title: 'renders ToolTip closed correctly with information', config: () => {
    render(
      <ThemeProvider theme={theme}>
        <ToolTip open={false}>Zucchini, Tomaten, Melanzani</ToolTip>
      </ThemeProvider>
    );
    const children = screen.getByText('Zucchini, Tomaten, Melanzani');
    expect(children).not.toBeVisible();
  });
});
```

<https://github.com/bachelor-project-mmp3/mmp3/commit/4993a4efea0cbaf12a4bf1a7a1547ee198db284e>

b) Wählen Sie einen **End-to-End Test** aus dem Projekt den Sie entweder alleine geschrieben haben, oder an dessen Umsetzung sie wesentlich beteiligt waren. Was wird in diesem Test getestet, warum, wie? Zeigen Sie Code Beispiele hier und geben Sie Links zum Commit im Original Repository an.

```

describe('title: "Edit profile and cancel", fn: () => {
  beforeEach(() => {
    cy.visit('http://localhost:3000/');
    cy.contains("Let's get started").should(chainer: 'be.visible').click();

    cy.origin('https://auth.projects.multimediatechnology.at', () => {
      cy.get('input[name="username"]').type(Cypress.env('username'));
      cy.get('input[name="password"]').type(Cypress.env('password'));
      cy.contains('Log In').click();
    });
    cy.visit(`http://localhost:3000/profile/${Cypress.env('id')}`);
  });

  it('title: "should try to edit the profile, but cancels", config: () => {
    cy.contains(`${Cypress.env('firstname')}'s hosted events`).should(
      chainer: 'be.visible'
    );
    cy.contains('Edit profile').click();
    cy.get('input[name="firstName"]').type(text: 'Max');
    cy.get('input[name="lastName"]').type(text: 'Mustermann');
    cy.contains('Cancel').click();

    cy.contains(`${Cypress.env('firstname')}`).should(chainer: 'be.visible');
  });
});
```

Mein E2e test besteht darin zu prüfen, ob beim Edit Profile Form der Input nicht verändert wird, wenn der User auf Cancel drückt.

Zuerst muss der/die User/in sich anmelden, was im beforeEach durch die end Variablen im cypress config passiert. Nach der erfolgreichen Anmeldung wird auf das Profile geklickt und dort ein anderer Name in das FirstName und das LastName Inputfeld eingefügt. Dann soll auf Cancel geklickt werden und durch das anschließende Überprüfen des Contents des Namens wird sichergestellt, dass er immer noch gleich ist. Da beim Cancel ja nicht die onSubmit Funktion aufgerufen wird, darf sich der Inhalt der Datenbank auch nicht verändern.

<https://github.com/bachelor-project-mmp3/mmp3/commit/37f4fa2cc72cb8a9fbc551660519e5e7370a004f>

(20) Lisa kann ein Refactoring durchführen

Ich kenne allgemeine Code Smells und spezifische Code Smells für Ruby und JavaScript. Ich kenne Refactorings, um diese Code Smells zu beheben. Ich kann ein Refactoring durchführen, das nur den Code verbessert, ohne neue Funktionalität zu implementieren. Ich kann die Rolle von Tests beim Refactoring erklären.

Wählen Sie einen Beispiel-Refactoring aus, das Sie durchgeführt haben. Beschreiben Sie: warum wurde das Refactoring durchgeführt? Wie sind Sie dabei vorgegangen? Hat das Refactoring den gewünschten Erfolg erzielt?

Eines meiner Refactorings war es, die Pagination in eine eigene Komponente zu geben, da sie in jeder anderen Komponente einfach kopiert worden war. Das vermeidet Code Duplikationen auf 3 verschiedenen Seiten und sogar auf eine Seite zwei mal, das die Pagination bei My-Events einmal bei den Upcoming und einmal bei den Past Events, dann im Profile bei der Past events und bei den All Events vorkommt. Ursprünglich war die Pagination nur bei All events eingebaut wodurch noch keine eigene Komponente zwingend gebraucht wurde. Durch eine verbesserte Usability und ein besseres Layout haben wir beschlossen es auf weiteren Seiten einzubauen, wohin es nur gecopied wurde. Jetzt gibt es eine einzige Komponente mit 4 Properties die von außen mitgegeben werden können.

<https://github.com/bachelor-project-mmp3/mmp3/pull/281/commits/9be674424d49d6b7daa931ae428df121cf27445b>

```
5+ usages  ± fhs45899
const Pagination: React.FC<{
  eventsPageIndex: number;
  eventsPageCount: number;
  setEventsPageIndex: (number) => void;
}> = ({ eventsPageIndex: number, eventsPageCount: number, setEventsPageIndex }) => {
  return (
    <StyledPagination>
      <PaginationEvents>
        <PaginationAction
          onClick={
            eventsPageIndex !== 1
              ? () => {
                  setEventsPageIndex( number: eventsPageIndex - 1 );
                }
              : null
          }
          disabled={eventsPageIndex === 1}
        <StyledFilterIcon option="prev" />
        Prev
      </PaginationAction>
      <PaginationPageCount>` ${eventsPageIndex}/${eventsPageCount}` </PaginationPageCount>
      <PaginationAction
        onClick={
          eventsPageIndex !== eventsPageCount
            ? () => {
                setEventsPageIndex( number: eventsPageIndex + 1 );
              }
            : null
        }
        disabled={eventsPageIndex === eventsPageCount}
      >
        Next
        <StyledFilterIcon option="next" />
      </PaginationAction>
    </PaginationEvents>
  </StyledPagination>
);
};
```

(21) Kerstin kennt die Arbeitsteilung in Web-Projekten

Ich kann ein Web-Projekt allein umsetzen oder in verschiedenen Rollen einem Team mitwirken: Web Operations, Backend, Frontend, Web-Design. Ich kenne meine Stärken.

Beschreiben Sie Ihre Rolle und Ihren Verantwortungsbereich im Projekt. Beschreiben Sie anhand einer konkreten User Story wie der Arbeitsablauf im Projekt ist, an welcher Stelle Sie übernehmen bzw. an andere KollegInnen übergeben.

Unser Projektteam ist von Beginn an sehr agil, daher nehme auch ich verschiedene Rollen und Verantwortungsbereiche ein. Meine Verantwortungsbereiche fallen unter Konzeption, Setup/Deployment, Fullstack-Entwicklung (Frontend und Backend), Projektmanagement und Testing.

In der Konzeptionsphase überlegten wir uns intensiv die Kernfeatures, wobei die technische Machbarkeit, also wie wir Features umsetzen können auch diskutiert wurde. Anschließend erfassten wir gemeinsam im Kanban-Board alle Userstories. Diese User-Stories priorisierten wir und stimmten uns gemeinsam ab, wer welche Features umsetzen wird.

Mein besonderer Fokus zu Beginn des Projektes war das gesamte Projekt-Setup und Deployment. Zuerst habe ich die Next.js App lokal aufgesetzt und die Datenbank eingerichtet. Anschließend habe ich die Infrastruktur für Vercel konfiguriert und Heroku mit Unterstützung von Lisa und Tanja eingerichtet. Das erste erfolgreiche Deployment schloss diese Aufgabe ab.

Bei der Fullstack-Entwicklung lag mein Fokus auf dem Events-Feed, den Requests und den Cron-Jobs. Ich möchte anhand von den Requests näher auf den Arbeitsablauf eingehen:

Die Userstory wurde für die erste Studiwoche eingeplant und mir zugeteilt:

Requests: Host accepts/declines join request #24

Open LisMaRad opened on Dec 10, 2022

LisMaRad now (edited) Edit

As a user, I want to see my join requests in the request tab .

I want to be able to just click on a button and accept or deny. -> quick way
I want to be able to click on the profile card and get to the user's profile. After I look through the profile, I want to press on a go back button and click on a button and accept or deny on my.

When I accept or decline, I want the guest to receive a notification email and a notification in their profile.

TODO

when the time is over for sending a request, an automated deny should be sent.

Bevor ich startete, gab ich meinen Kolleginnen Bescheid, dass ich nun damit beginne und zog mir vom aktuellsten dev-Branch einen Feature-Branch. Ich setzte sowohl Frontend als auch Backend um, somit musste ich währenddessen keine anderen Commits berücksichtigen. Während dem backendseitigen Umsetzen, klärte ich immer wieder mit Tanja und Lisa die Logik und den Ablauf eines Request Zyklus ab, um diesen auch wirklich richtig umzusetzen. Nach der Fertigstellung der Implementierung testete ich lokal alle möglichen Szenarien durch und fakte dafür auch Requests von anderen Personen, um alles abzudecken. Nach dem Testen erstellte ich einen Merge Request auf den dev-Branch. Dieser wurde dann gemerged und gegen Ende der Woche wurde dann der dev-Branch in den main-Branch gemerged. Anschließend wurden die Requests intensiv von uns allen auf der Produktiv-Umgebung getestet.

(22) Kerstin kann User Stories implementieren und deployen

Ich kann bei der Ausformulierung von User Stories mitarbeiten, um alle zur Implementierung nötigen Fragen zu klären. Ich kann den nötigen Programmcode in HTML, CSS, Javascript, SQL, PHP, Ruby schreiben, der die User Story umsetzt. Ich kann die nötigen Schritte durchführen um die fertig implementierte User Story in production zu deployen.

- c) Listen Sie 5 User Stories auf, die sie allein umgesetzt haben oder zu denen Sie den größten Beitrag geleistet haben
- d) Wählen Sie eine der User Story aus, Zeigen Sie die User Story, die Implementierung (Codebeispiel hier und Verweis auf des Original-Repository) und das fertige Feature im Projekt (Verweis auf production-Server)

User Story 1 – Events Feed:

As a user, I want to see all the upcoming events in my feed.

User Story 2 – Quick join an event

As a user, I want to be able to join the event via a click on a button in the events feed without clicking on the event detail to get extra information.

After the button click, I want to see a pop up with the message "Join request sent.

User Story 3 – Requests: Host accepts/declines join request:

As a user, I want to see my join requests in the request tab.

I want to be able to just click on a button and accept or deny. -> quick way

User Story 4 – Get join request information per email as host:

As a user, I want to receive an email if a guest wants to join.

User Story 5 – Filter Feed Campus:

As a user, I want to filter via Campus. I want to select one campus and only see the events happening in those area.

Filter Feed Campus:

Feature Pull Request: <https://github.com/bachelor-project-mmp3/mmp3/pull/99>

Userstory:

The screenshot shows a GitHub issue page with the following details:

- Title:** Filter Feed Campus #10
- Status:** Opened by LisMaRad on Nov 21, 2022
- Description:**

As a user,

 - I want to filter the events feed via campus.
 - I want to select one campus and only see the events happening in this area.
 - If there is no campus selected, the filter shows "Any Campus", an all events are shown.
- Editor:** A rich text editor with standard formatting tools (bold, italic, etc.) and a preview section.
- Comments:** A text input field for leaving a comment.
- Markdown support:** A note indicating "Markdown is supported".
- File upload:** A placeholder for "Paste, drop, or click to add files".
- Buttons:** Close issue, Comment, and a large empty area for the comment body.

Production URL: <https://mmp3.vercel.app/events>

Studentenfutter

- My Events
- All Events
- Create Event
- Requests
- Profile

[Logout](#)

Imprint Data Privacy

Find an event to join

Any campus ▾ Any date ▾

- Campus Urstein
- Campus Kuchl
- Campus Schwarzach

[Reset](#) [Save](#)

Kerstin

test 1 € p. p.
* sdfsdssfdf

09/03/2023 00:30 AM

100 days left to apply
4/99 seats taken

Daniel-Markus

Campus Urstein

Sepps Suppen... 5 € p. p.

* Tomatensuppe

[Withdraw](#)

```

const FilterCampus: React.FC<FilterProps> = ({  
    children,  
    onSubmit,  
    currentFilter,  
}: FilterProps) => {  
    const [showCampusFilterList, setShowCampusFilterList] = useState(false);  
    const [preSettedFilterCampus, setPreSettedFilterCampus] = useState<  
        string | undefined  
>(currentFilter);  
  
    return (  
        <Wrapper>  
            {showCampusFilterList && (  
                <>  
                    <FakeBlur onClick={() => setShowCampusFilterList(false)} />  
                    <CampusList>  
                        <Headline>Filter by campus</Headline>  
                        <FilterListWrapper>  
                            {dormitories.map((dormitory) => (  
                                <FilterItemWrapper  
                                    key={`${dormitory}-filter-entry'}`>  
                                    <FilterItem  
                                        selected={  
                                            dormitory === preSettedFilterCampus  
                                        }  
                                        onClick={() =>  
                                            setPreSettedFilterCampus(dormitory)  
                                        }>  
                                            {dormitory}  
                                    </FilterItem>  
                                    {dormitory === preSettedFilterCampus && (  
                                        <StyledCheck />  
                                    )}  
                                </FilterItemWrapper>  
                            ))}  
                        </FilterListWrapper>  
                        <ButtonWrapper>  
                            <Button  
                                onClick={() => {  
                                    setPreSettedFilterCampus(undefined);  
                                    onSubmit(undefined);  
                                }}  
                                variant="secondary">  
                                Reset  
                            </Button>  
                            <Button  
                                onClick={() => onSubmit(preSettedFilterCampus)}  
                                variant="primary">  
                                Save  
                            </Button>  
                        </ButtonWrapper>  
                    </CampusList>  
                </>  
            )}  
            <FilterButton  
                onClick={() => setShowCampusFilterList(true)}  
                isOpen={showCampusFilterList}>  
                {children}  
            </FilterButton>  
        </Wrapper>  
    );  
};  
  
export default FilterCampus;

```

Abbildung 9: Zuerst wurde die FilterButton-Komponente implementiert, welche als Properties die children, onSubmit und den currentFilter mitbekommt

```
pages/api/events/index.ts
```

```
55         // GET events /api/events
56     else if (req.method === 'GET') {
57         const today = new Date();
58         const events = await prisma.event.findMany({
59             orderBy: [
60                 {
61                     date: 'asc',
62                 }
63             ],
64             where: {
65                 AND: [
66                     {
67                         timeLimit: { gte: today },
68                     },
69                     {
70                         host: {
71                             dormitory: dormitoryFilter as string,
72                         }
73                     }
74                 ],
75             }
76         });
77         res.json(events);
78     } else {
79         const event = await prisma.event.create({
80             data: {
81                 title: req.body.title,
82                 description: req.body.description,
83                 date: req.body.date,
84                 timeLimit: req.body.timeLimit,
85                 host: {
86                     connect: {
87                         id: req.body.hostId
88                     }
89                 }
90             }
91         });
92         res.json(event);
93     }
94 }
```

Abbildung 10: Im Backend musste dann beim GET-Request vom Events-Feed der Campus-Filter vom req-Objekt ausgelesen werden. Wenn ein Filter gesetzt ist, dann muss die Datenbank-Abfrage um den Filter erweitert werden, ansonsten bleibt die Abfrage so wie vorher.

Im Frontend im Events-Feed wurde ein State für den Campus-Filter hinzugefügt und eine onFilterEvents-Methode, welche einen GET-Request mit dem dormitoryFilter an das Backend macht und den Events-State update mit der gefilterten Eventsliste vom Response. Über der Eventsliste wurde eine Filterbar gestyled, welche dann die FilterCampus-Komponente rendernt:

```
const [filterCampus, setFilterCampus] = useState<string | undefined>();  
  
const onFilterEvents = async (filter: string) => {  
    setLoading(true);  
    setFilterCampus(filter);  
    fetch(`api/events?dormitoryFilter=${filter}`, {  
        method: 'GET',  
    })  
        .then((res) => res.json())  
        .then((data) => {  
            setEvents(data.events);  
            setLoading(false);  
        });  
};  
  
// render / find an event to join / add to  
<FilterBar>  
  <FilterCampus  
    onSubmit={onFilterEvents}  
    currentFilter={filterCampus}>  
    {filterCampus ?? 'Any campus'}  
  </FilterCampus>  
</FilterBar>
```

(23) Kerstin kann Fehler im Programm (Bugs) finden und beheben

Ich kann einen Bug, den ich gefunden habe, so beschreiben, dass eine Suche nach den Ursachen möglichst einfach wird. Ich kann die Ursache eines Bugs in HTML, CSS, Javascript, SQL, PHP, Ruby Code mittels Developer Tools und Log Messages finden und beheben.

Wählen Sie einen Beispiel-Bug aus dem Projekt aus, und beschreiben Sie: wie wurde der Bug entdeckt, wie haben Sie die Ursache gefunden, wie haben Sie den Bug behoben.

Folgefrage: gibt es Regressions-Test die ein Wiederauftreten des Bugs entdecken würden?

Der Merge Request: <https://github.com/bachelor-project-mmp3/mmp3/commit/aab03aebed4a7a40b8ca9de066f34fa0322debf1>

Die Hilfsfunktionen `getFormattedTime` und `formatDateForForm` wurden unter anderem angepasst:

```
10 10
11 11      export const getFormattedTime = (date: string) => {
12 -        const time = date.split('T')[1];
12 +        const convertedDate = new Date(date);
13 +        const time = convertedDate.toLocaleTimeString();
14 +
15         const timeArray = time.split(':');
16         const amPm = Number(timeArray[0]) <= 12 ? ' AM' : ' PM';
17         const hours =
18             Number(timeArray[0]) <= 12 ? timeArray[0] : Number(timeArray[0]) - 12;
19
20         return hours + ':' + timeArray[1] + amPm;
21     };
22

+
+ export const formatDateForForm = (date: Date) => {
+     return (
+         date.getFullYear() +
+         '-' +
+         formatDateForDateInput(date.getMonth() + 1) +
+         '-' +
+         formatDateForDateInput(date.getDate()) +
+         'T' +
+         date.toLocaleTimeString().substring(0, 5)
+     );
};
```

(24) Kerstin kann Unit-Tests und End-to-End Tests schreiben

Ich kenne Test-frameworks für die Technologien, die ich einsetze, und kann sowohl Unit Tests als auch End-to-End Tests schreiben. Ich kann Test Driven Development einsetzen und ich kann Tests für bestehenden Code schreiben. Ich verstehe welchen Teil des Projekts mein Test testet.

a) Wählen Sie einen Unit Test aus dem Projekt den Sie entweder alleine geschrieben haben, oder an dessen Umsetzung sie wesentlich beteiligt waren. Was wird in diesem Test getestet, warum, wie? Zeigen Sie Code Beispiele hier und geben Sie Links zum Commit im Original Repository an.

```
import { render, screen, fireEvent } from '@testing-library/react';
import { Button } from '../components/atoms/Button';
import { ThemeProvider } from 'styled-components';
import '@testing-library/jest-dom/extend-expect';
import { theme } from '../ThemeConfig';

describe('Button', () => {
  it('renders with correct primary styles', () => {
    const onClickMock = jest.fn();

    render(
      <ThemeProvider theme={theme}>
        <Button variant="primary" onClick={onClickMock}>
          Click me
        </Button>
      </ThemeProvider>
    );

    const button = screen.getByText('Click me');
    expect(button).toBeInTheDocument();
    expect(button).toHaveStyle(`
      background-color: #22853c;
      border: 2px solid #22853c;
      color: white;
      font-size: 16px;
    `);
  });
  it('calls onClick callback when clicked', () => {
    const onClickMock = jest.fn();

    render(
      <ThemeProvider theme={theme}>
        <Button variant="secondary" onClick={onClickMock}>
          Click me
        </Button>
      </ThemeProvider>
    );

    const button = screen.getByText('Click me');
    fireEvent.click(button);
    expect(onClickMock).toHaveBeenCalledTimes(1);
  });

  it('renders with correct styles for "red" variant', () => {
    render(
      <ThemeProvider theme={theme}>
        <Button variant="red" disabled={false}>
          Red Button
        </Button>
      </ThemeProvider>
    );

    const button = screen.getByText('Red Button');
    expect(button).toHaveStyle(`
      background-color: white;
      color: #DB1F1F;
      border: 2px solid #DB1F1F;
      font-size: 16px;
    `);
  });
  it('renders with correct styles for "secondary" variant', () => {
    render(
      <ThemeProvider theme={theme}>
        <Button variant="secondary" disabled={false}>
          Secondary Button
        </Button>
      </ThemeProvider>
    );

    const button = screen.getByText('Secondary Button');
    expect(button).toHaveStyle(`
      background-color: white;
      color: #22853c;
      border: 2px solid #22853c;
      font-size: 16px;
    `);
  });
  it('renders with correct width', () => {
    render(
      <ThemeProvider theme={theme}>
        <Button variant="primary" width={50}>
          Wide Button
        </Button>
      </ThemeProvider>
    );

    const button = screen.getByText('Wide Button');
    expect(button).toHaveStyle('width: 50%');
  });
  it('renders a disabled button', () => {
    render(
      <ThemeProvider theme={theme}>
        <Button variant="primary" disabled>
          Disabled Button
        </Button>
      </ThemeProvider>
    );

    const button = screen.getByText('Disabled Button');
    expect(button).toHaveAttribute('disabled');
    expect(button).toHaveStyle(`
      background-color: #22853c;
      border: 2px solid #22853c;
      color: white;
      opacity: 0.5;
      cursor: not-allowed;
      font-size: 16px;
    `);
  });
});
```

b) Wählen Sie einen **End-to-End Test** aus dem Projekt den Sie entweder alleine geschrieben haben, oder an dessen Umsetzung sie wesentlich beteiligt waren. Was wird in diesem Test getestet, warum, wie? Zeigen Sie Code Beispiele hier und geben Sie Links zum Commit im Original Repository an.

Der End-to-End Test, den ich alleine geschrieben habe, testet, ob man als eingeloggter User den Eventsfeed und das Eventdetail aufrufen kann und der richtige Content angezeigt wird. Außerdem testet er, ob bei einem nicht eingeloggten User auf die Startseite geleitet wird.

Link zum Code: <https://github.com/bachelor-project-mmp3/mmp3/blob/dev/cypress/e2e/eventsFeedAndDetail.cy.js>

```
describe('Events Feed and Event Detail navigation', () => {
  it('should click the first event in events feed and open event detail when user is logged in', () => {
    cy.visit('http://localhost:3000/');
    cy.contains("Let's get started").should('be.visible').click();

    cy.origin('https://auth.projects.multimediatechnology.at', () => {
      cy.get('input[name="username"]').type(Cypress.env('username'));
      cy.get('input[name="password"]').type(Cypress.env('password'));
      cy.contains('Log In').click();
    });
    cy.contains('Show all events').click();

    // Find all elements with class starting with "ExtendedEventPreview"
    cy.get('[class^="ExtendedEventPreview"]')
      .first().click();

    cy.contains('Menu').should('be.visible');

    cy.get('body')
      .invoke('text') // Retrieve the text content of the entire page
      .then((text) => {
        const wildcardRegex = /Costs:*/; // wildcard pattern
        expect(text).to.match(wildcardRegex); // Assert that the text matches the wildcard pattern
      });

    cy.get('body')
      .invoke('text')
      .then((text) => {
        const wildcardRegex = /:left to apply/;
        expect(text).to.match(wildcardRegex);
      });
  });

  it('should redirect to home if user is not logged in', () => {
    cy.visit('http://localhost:3000/');

    cy.visit('http://localhost:3000/events');

    // Get the current path
    cy.location('pathname').should('eq', '/'); // Replace '' with the expected root path

    cy.visit('http://localhost:3000/events/anyID');
    cy.location('pathname').should('eq', '/'); // Replace '' with the expected root path
  });
});
```

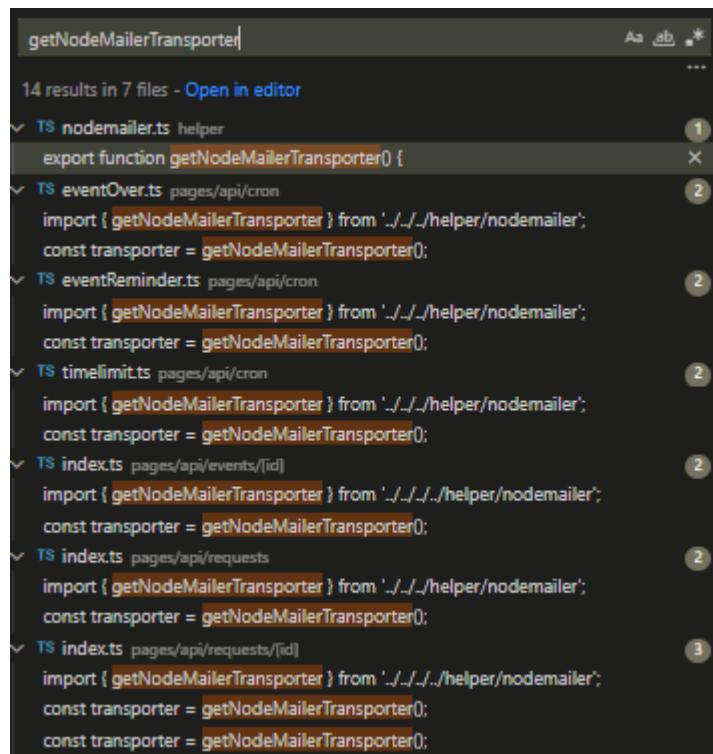
(25) Kerstin kann ein Refactoring durchführen

Ich kenne allgemeine Code Smells und spezifische Code Smells für Ruby und JavaScript. Ich kenne Refactorings, um diese Code Smells zu beheben. Ich kann ein Refactoring durchführen, das nur den Code verbessert, ohne neue Funktionalität zu implementieren. Ich kann die Rolle von Tests beim Refactoring erklären.

Wählen Sie einen Beispiel-Refactoring aus, das Sie durchgeführt haben. Beschreiben Sie: warum wurde das Refactoring durchgeführt? Wie sind Sie dabei vorgegangen? Hat das Refactoring den gewünschten Erfolg erzielt?

1. Ein Beispiel-Refactoring von mir ist die Auslagerung vom Nodemailer transporter. Ich habe dafür eine helper-Funktion geschrieben, welche das transporter-Objekt zurückgibt. Das hat den Grund, weil mehrmals an verschiedenen Stellen im Code dieser transporter zum Mail-versenden benötigt wird und somit nur an einer Stelle statt an diesen vielen Stellen der Code implementiert wird und daher Code-Duplizierung vermieden wird. Somit ist der Code statt siebenmal nur mehr einmal implementiert und wird an den benötigten Stellen importiert.

```
export function getNodeMailerTransporter() {
  let nodemailer = require('nodemailer');
  const transporter = nodemailer.createTransport({
    port: 465,
    host: 'smtp.gmail.com',
    auth: {
      user: 'studentenfuttermmp3@gmail.com',
      pass: `${process.env.PASSWORD}`,
    },
    secure: true,
  });
  return transporter;
}
```



The screenshot shows a code search interface with the query 'getNodeMailerTransporter'. It displays 14 results found in 7 files. The results are listed in a tree view:

- 1 TS nodemailer.ts helper: export function getNodeMailerTransporter0 { ... }
- 2 TS eventOver.ts pages/api/cron: import { getNodeMailerTransporter } from '././helper/nodemailer'; const transporter = getNodeMailerTransporter();
- 2 TS eventReminder.ts pages/api/cron: import { getNodeMailerTransporter } from '././helper/nodemailer'; const transporter = getNodeMailerTransporter();
- 2 TS timelimit.ts pages/api/cron: import { getNodeMailerTransporter } from '././helper/nodemailer'; const transporter = getNodeMailerTransporter();
- 2 TS index.ts pages/api/events/[id]: import { getNodeMailerTransporter } from './././helper/nodemailer'; const transporter = getNodeMailerTransporter();
- 2 TS index.ts pages/api/requests: import { getNodeMailerTransporter } from './././helper/nodemailer'; const transporter = getNodeMailerTransporter();
- 3 TS index.ts pages/api/requests/[id]: import { getNodeMailerTransporter } from './././helper/nodemailer'; const transporter = getNodeMailerTransporter(); const transporter = getNodeMailerTransporter();

2. Weiters habe ich für die Anzeige der Notifications einen React-Context implementiert, sodass der State global verwendet wird und nicht bei jedem Re-Render flackert, weil der State initial undefined ist. -> Siehe Merge Request <https://github.com/bachelor-project-mmp3/mmp3/pull/244>