

Elias Amstein, Ben Stadelmann, Fabrizio Poli & Ghiath Sardast

Inhaltsverzeichnis

[Projektbeschreibung 3](#_Toc137655990)

[Projektname: Dream Journal 3](#_Toc137655991)

[Beschreibung: 3](#_Toc137655992)

[Funktionen: 3](#_Toc137655993)

[Technologien: 3](#_Toc137655994)

[Vorgehen 4](#_Toc137655995)

[Hauptkomponente 5](#_Toc137655996)

[Vue-Komponenten: 5](#_Toc137655997)

[Routing: 5](#_Toc137655998)

[LocalStorage: 5](#_Toc137655999)

[Styling: 5](#_Toc137656000)

[Learning how 6](#_Toc137656001)

[StyleGuide 6](#_Toc137656002)

[Screenshots 7](#_Toc137656003)

# Projektbeschreibung

### Projektname: Dream Journal

### Beschreibung:

Das Dream Journal ist eine Webanwendung, die es Benutzern ermöglicht, ihre Träume aufzuzeichnen und zu verwalten. Die Anwendung bietet verschiedene Funktionen zur Erfassung, Anzeige und Bearbeitung von Traumdaten. Benutzer können ihre Träume mit Titel, Datum und einer detaillierten Beschreibung erfassen. Die erfassten Träume werden im Local Storage des Browsers gespeichert, um sie lokal auf dem Gerät des Benutzers zu speichern.

### Funktionen:

**Startseite**: Auf der Startseite werden Benutzer mit einer Begrüssungsnachricht und Links zu verschiedenen Bereichen der Anwendung begrüsst. Benutzer können auf den Link "View Dreams" klicken, um eine Liste ihrer erfassten Träume anzuzeigen, oder auf den Link "Add a Dream", um einen neuen Traum hinzuzufügen.

**Traumliste**: Auf der Traumliste-Seite werden alle erfassten Träume in einer übersichtlichen Liste angezeigt. Jeder Traum wird mit dem Titel und einer Kurzbeschreibung dargestellt. Benutzer haben auch die Möglichkeit, den Traum detaillierter anzuzeigen, zu bearbeiten oder zu löschen.

**Traumdetails**: Auf der Traumdetails-Seite werden alle Informationen eines bestimmten Traums angezeigt, einschliesslich des Titels, des Datums und einer ausführlichen Beschreibung.

**Traum** **hinzufügen**: Der Traum hinzufügen-Seite enthält ein Formular, in dem Benutzer einen neuen Traum mit einem Titel, einem Datum und einer detaillierten Beschreibung erfassen können. Nach dem Absenden des Formulars wird der Traum in der Liste der erfassten Träume angezeigt.

**Traum bearbeiten**: Der Traum bearbeiten-Seite ermöglicht es Benutzern, einen vorhandenen Traum zu bearbeiten. Benutzer können die Informationen aktualisieren und speichern.

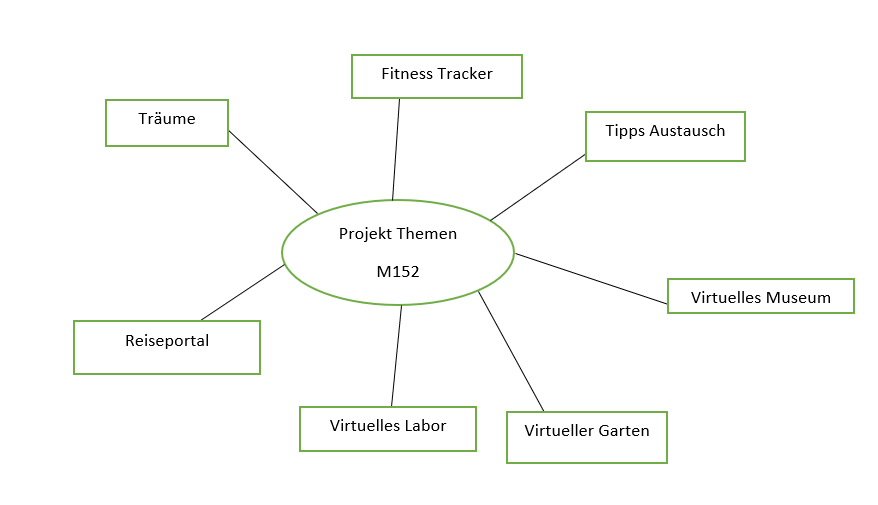
# Technologien:

**Vue.js**: Das Projekt wird mithilfe des Vue.js-Frameworks entwickelt, das eine reaktive und komponentenbasierte Entwicklung ermöglicht.

**Vue Router**: Der Vue Router wird verwendet, um die Routing-Funktionalität in der Anwendung bereitzustellen und die Navigation zwischen den verschiedenen Seiten zu ermöglichen.

**Local Storage**: Der Local Storage des Browsers wird genutzt, um die Traumdaten lokal zu speichern, so dass Benutzer ihre erfassten Träume auch nach dem Schliessen der Anwendung wieder abrufen können.

# Vorgehen



Wir haben damit begonnen, die Anforderungen an unsere Traumtagebuchanwendung zu verstehen. Wir haben definiert, dass wir eine Startseite, eine Traumliste, eine Funktion zum Hinzufügen von Träumen, eine Ansicht für die Details eines Traums und eine Bearbeitungsfunktion benötigen. Die Daten, die wir speichern möchten, umfassen den Titel, das Datum und eine Beschreibung für jeden Traum. Wir haben eine Benutzeroberfläche skizziert, die ein ansprechendes Design aufweist und es den Benutzern ermöglicht, intuitiv zwischen den verschiedenen Seiten zu navigieren.

Nach der Anforderungsanalyse haben wir ein neues Vue.js-Projekt erstellt. Wir haben das Vue-CLI verwendet, um die Grundstruktur unseres Projekts zu generieren und die benötigten Abhängigkeiten wie den Vue Router installiert.

Anschliessend haben wir unsere Anwendung in Komponenten aufgeteilt. Wir haben separate Komponenten für die Startseite, die Traumliste, das Hinzufügen von Träumen, die Traumdetails und die Bearbeitung von Träumen erstellt. In diesen Komponenten haben wir die Datenstrukturen für Traumobjekte definiert und die Logik implementiert, um Träume zu speichern, zu aktualisieren und zu löschen.

Um die Navigation zwischen den verschiedenen Seiten zu ermöglichen, haben wir das Vue Router-Modul konfiguriert. Wir haben die Routen für die Startseite, die Traumliste, das Hinzufügen von Träumen, die Traumdetails und die Bearbeitung von Träumen festgelegt.

Damit die Traumdaten persistent gespeichert werden, haben wir die Funktionalität zur Verwendung des Local Storage implementiert. Beim Hinzufügen, Aktualisieren und Löschen von Träumen haben wir die entsprechenden Daten im Local Storage aktualisiert.

Um das Erscheinungsbild unserer Anwendung anzupassen, haben wir CSS verwendet. Wir haben Hintergrundbilder, Schriftarten, Farben und Layouts angepasst, um eine ansprechende Benutzeroberfläche zu gestalten. Zusätzlich haben wir Stile für aktive Links implementiert und visuelle Effekte hinzugefügt, um die Benutzererfahrung zu verbessern.

Um sicherzustellen, dass unsere Anwendung wie erwartet funktioniert, haben wir Tests durchgeführt. Wir haben die Konsolenausgabe überprüft, um Fehler und Warnungen zu identifizieren, und Entwicklertools genutzt, um eventuelle Probleme zu debuggen und zu beheben.

# Hauptkomponente

## Vue-Komponenten:

**App.vue**: Die Hauptkomponente der Anwendung. Hier werden die anderen Komponenten gerendert und die Routen definiert.

**Home.vue**: Eine Komponente, die die Startseite der Anwendung darstellt.

**DreamsPage.vue**: Eine Komponente, die eine Liste von Träumen darstellt. Die Träume werden aus dem Local Storage geladen und angezeigt. Wenn keine Träume vorhanden sind, wird eine entsprechende Nachricht angezeigt.

**DreamDetails.vue**: Eine Komponente, die die Details eines einzelnen Traums anzeigt. Die Traumdaten werden aus dem Local Storage abgerufen und angezeigt.

**AddDream.vue**: Eine Komponente, die ein Formular zur Erfassung eines neuen Traums enthält. Wenn das Formular abgeschickt wird, werden die Daten im Local Storage gespeichert.

**EditDream.vue**: Eine Komponente, die ein Formular zur Bearbeitung eines vorhandenen Traums enthält. Die Daten des Traums werden aus dem Local Storage abgerufen und im Formular angezeigt. Bei der Aktualisierung werden die geänderten Daten im Local Storage gespeichert.

### Routing:

Das Vue Router-Paket wird verwendet, um die Routing-Funktionalität in der Anwendung bereitzustellen. Es werden fünf Routen definiert:

'/': Die Startseite der Anwendung.

'/dreams': Eine Seite, auf der die Liste der Träume angezeigt wird.

'/dreams/:id': Eine Seite, auf der die Details eines bestimmten Traums angezeigt werden.

'/add-dream': Eine Seite, auf der man einen Traum erfassen kann.

'/edit-dream/:id': Eine Seite, auf der man einen Traum bearbeiten kann.

### LocalStorage:

Der Local Storage wird verwendet, um die Traumdaten lokal im Browser zu speichern. Die Daten werden im JSON-Format gespeichert und können mithilfe der localStorage-API abgerufen und aktualisiert werden.

### Styling:

Die Styling-Regeln werden in den einzelnen Komponenten mithilfe von inline-Styles definiert.

# Learning how

* Vue.js-Grundlagen
* Wie man Routing in einer Vue.js-Anwendung implementiert
* Formularverarbeitung in Vue.js
* Wie man den Local Storage in Vue.js nutzt
* CSS-Styling Grundlagen verbessert
* Fehlerbehandlung und Debugging in Vue.js

StyleGuide

Background Color: HEX: #fef6f0; RGB: 254, 246, 240 

Font Color: HEX: #474747; RGB: 71, 71, 71 

Font Family: Cursive

Navigation Background: HEX: #f3dace; RGB: 243, 218, 206 

Links Hover Mode: Unterlined

Title: Font Size: 24px

Button Background: HEX: #22181c; RGB: 34, 24, 28 

Button Font Color: HEX: #ffffff; RGB: 255, 255, 255 

Button Hover Background: HEX: #745260; RGB: 116, 82, 96 

Delete Button Background: HEX: #e21919; RGB: 226, 25, 25 

Delete Button Hover Background:HEX: #ea7171; RGB: 234, 113, 113 

Input Font Family: sans-serif

Input Border: Color = Button Background Color; Width: 1px; Style: Solid

Input Border Hover: Color = Btn Hover Background Color; Width: 1px; Style: Solid

# Screenshots

