Ein Bild, das Text, Schrift, Logo, Screenshot enthält.

Automatisch generierte Beschreibung

Projektdokumentation Melodia

**Auftraggeber:**

DI. Michael Leeb

**Ausgeführt im Schuljahr 2025/26 von:**

Fabian Lampert 5bWI

Christian Schallner 5bWI

Linus Wörndle 5bWI

Dornbirn, am 01.10.2025

Inhaltsverzeichnis

[Vorwort 3](#_Toc191415338)

[Abstract 3](#_Toc191415339)

[Projektteam 4](#_Toc191415340)

[Einleitung 5](#_Toc191415341)

[Technische Anforderungen & Technologien 6](#_Toc191415342)

[Systemanforderungen 6](#_Toc191415343)

[Hardware-Anforderungen 6](#_Toc191415344)

[Software-Anforderungen 6](#_Toc191415345)

[Abhängigkeiten 6](#_Toc191415346)

[Setup und Installation 7](#_Toc191415347)

[Voraussetzungen 7](#_Toc191415348)

[Schritt-für-Schritt-Anleitung 7](#_Toc191415349)

[Features und Funktionalität 9](#_Toc191415350)

[HomeSphere Frontend – Übersicht und Grundlagen 10](#_Toc191415351)

[1. Allgemeiner Überblick 10](#_Toc191415352)

[2. Ordnerstruktur 10](#_Toc191415353)

[3. Grundkonzepte 11](#_Toc191415354)

[3.1 API-Schicht 11](#_Toc191415355)

[3.2 Komponenten 11](#_Toc191415356)

[3.3 Views (Seiten) 11](#_Toc191415357)

[3.4 Routing 11](#_Toc191415358)

[3.5 Styling und Layout 11](#_Toc191415359)

[3.6 Hauptanwendung 11](#_Toc191415360)

[4. Zusammenfassung 11](#_Toc191415361)

[HomeSphere Backend – Übersicht und Grundlagen 11](#_Toc191415362)

[1. Allgemeiner Überblick 11](#_Toc191415363)

[2. Grundlegende Konzepte und Komponenten 12](#_Toc191415364)

[2.1 Models (Modelle) 12](#_Toc191415365)

[2.2 Controller 12](#_Toc191415366)

[2.3 Routes (Routen) 12](#_Toc191415367)

[2.4 Middlewares 12](#_Toc191415368)

[2.5 Utilities 12](#_Toc191415369)

[3. Ordnerstruktur 13](#_Toc191415370)

[4. Zusammenfassung 13](#_Toc191415371)

[Abbildungsverzeichnis 14](#_Toc191415372)

# Vorwort

Willkommen zur Dokumentation von *Melodia* – einem webbasierten Onlinespiel, bei dem Songs erraten werden müssen. *Melodia* wurde für Unterhaltungszwecke entwickelt. Die Spieler sollen Spaß beim Spielen haben.

In dieser Webanwendung kann aus verschiedene Modi ausgewählt werden, um persönliche Präferenzen der Spieler abzudecken. Ziel des Projekts ist es, ein spaßiges Onlinespiel zu erstellen, welches die Spieler motiviert wieder zu spielen.

Diese Dokumentation beschreibt die technische Umsetzung, die Architektur und die Funktionen von *Melodia*.

# Abstract

Many households are managed inefficiently, often due to an unclear distribution of household tasks. HomeSphere was developed to solve this problem by providing a central management platform for the household.  
  
The goal of the web application is to offer families and shared living communities an intuitive and efficient solution that significantly simplifies everyday life.  
  
The implementation was carried out through a structured analysis of requirements, followed by key project management analyses and subsequent development.  
  
Important results include an optimized user experience, high usability, and the successful integration of features such as shared shopping lists and task management.

Viele Haushalte werden ineffizient verwaltet, während die Aufgabenverteilung der Hausarbeiten oft nicht klar ist. *HomeSphere* wurde entwickelt, um dieses Problem zu lösen, indem die Plattform eine zentrale Verwaltung für den Haushalt bietet.

Das Ziel der Webanwendung ist es Familien und Wohngemeinschaften eine intuitive und effiziente Lösung anzubieten die den Alltag maßgeblich erleichtert.

Die Umsetzung erfolgte durch eine strukturierte Analyse der Anforderungen, gefolgt von den wichtigsten Projektmanagement-Analysen und der anschließenden Entwicklung.

Wichtige Ergebnisse sind eine optimierte Nutzererfahrung, eine hohe Usability sowie die erfolgreiche Integration von Funktionen wie gemeinschaftlichen Einkaufslisten und Aufgabenverwaltung.

# Projektteam

**Projektleiter**

**Linus Wörndle**

****Linus Wörndle übernahm die Rolle des Projektleiters und war für das Projektmanagement verantwortlich. Aufgrund seiner Erfahrungen in früheren Projekten brachte er wertvolle Kenntnisse in diesem Bereich ein. Zusätzlich beteiligte er sich an der Front-End-Entwicklung und übernahm spezifische Teilaufgaben, da er bereits Erfahrungen in der Webentwicklung hatte.

**Projektteammitglieder**

**** **Christian Schallner**

Christian Schallner war für die Entwicklung des Backends verantwortlich. Zu seinen Aufgaben gehörten die Implementierung der Serverlogik, die Datenbankverwaltung sowie die Anbindung an das Frontend.

**** **Fabian Lampert**

Fabian Lampert war für die Entwicklung des Frontends zuständig. Er implementierte das Benutzerinterface und stellte sicher, dass die Anwendung eine intuitive und ansprechende Nutzererfahrung bietet. Dabei arbeitete er eng mit dem Backend-Team zusammen, um eine reibungslose Kommunikation zwischen Frontend und Backend zu gewährleisten.

# Einleitung

Unser Projekt trägt den Namen HomeSphere und ist eine Software für die Hausverwaltung, die den Alltag erleichtert und besser organisiert. Die Idee entstand aus einem alltäglichen Problem: Wer kennt es nicht, Dinge zu vergessen, die man eigentlich einkaufen wollte? Oder: Wer hat Dinge vergessen zu erledigen, die man eigentlich machen sollte? Mit HomeSphere gehört das der Vergangenheit an.

Die Kernfunktion der Software ist eine benutzerfreundliche Einkaufsliste und ToDo Liste, die es ermöglicht, Produkte und ToDos einfach einzutragen und jederzeit darauf zuzugreifen. Sollte man etwas vergessen haben, kann es unkompliziert hinzugefügt werden, sodass es beim nächsten Male nicht erneut übersehen wird.

Die Integration von Familienmanagement macht uns einzigartig. Nutzer können Familien erstellen, Mitglieder hinzufügen und Einkaufslisten oder To-Do-Listen gemeinsam verwalten. Diese Funktion fördert die Zusammenarbeit und Transparenz innerhalb eines Haushalts.

Ursprünglich hatten wir geplant, ein Punkte- und Belohnungssystem einzubauen, um den Nutzern spielerisch Anreize für eine bessere Organisation zu geben. Aufgrund begrenzter Zeit war dies in der aktuellen Version jedoch nicht umsetzbar.

Die Hauptfunktionen von HomeSphere umfassen:

* Einkaufslisten: Produkte einfach hinzufügen, aktualisieren und verwalten.
* Familienmanagement: Einkaufs- und To-Do-Listen mit mehreren Nutzern teilen und bearbeiten.
* To-Do-Listen: Aufgaben planen und delegieren, um den Alltag strukturierter zu gestalten.

Mit HomeSphere bieten wir eine einfache, intuitive Lösung für die alltäglichen Herausforderungen im Haushalt und ermöglichen es Familien, effizienter zusammenzuarbeiten.

# Technische Anforderungen & Technologien

## Systemanforderungen

Da HomeSphere als Webanwendung entwickelt wurde, gibt es keine speziellen Hardwareanforderungen. Die Software ist plattformunabhängig und kann auf jedem Gerät mit einem modernen Webbrowser genutzt werden. Nachfolgend sind die technischen Anforderungen und verwendeten Technologien aufgeführt:

### Hardware-Anforderungen

* Client:
  + Ein Gerät mit Internetzugang (PC, Laptop, Tablet oder Smartphone).
* Server:
  + Ein Server mit Node.js-Unterstützung für das Backend (z. B. ein Cloud-Service oder lokaler Entwicklungsserver).

### Software-Anforderungen

Abbildung 1: https://iconduck.com/icons/27728/node-js

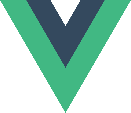
* Frontend:
  + Framework: Vue.js für eine reaktive und benutzerfreundliche Oberfläche.
  + Technologien: HTML, CSS und JavaScript.

Abbildung :  
https://en.m.wikipedia.org/wiki/File:Vue.js\_Logo\_2.svg

* Backend:
  + Framework: Express.js für die API-Entwicklung und Serverlogik.
  + Ein Bild, das Grafiken, Schrift, Grafikdesign, Logo enthält.

    KI-generierte Inhalte können fehlerhaft sein.Datenbank: MongoDB.
* Browser-Kompatibilität:
  + Unterstützt alle modernen Webbrowser (Chrome, Firefox, Edge, Safari).

Abbildung : https://de.m.wikipedia.org/wiki/Datei:MongoDB\_Logo.svg

### Abhängigkeiten

* Node.js: Mindestens Version 18.x zur Ausführung der Serverlogik.
* npm/Yarn: Für die Verwaltung von Abhängigkeiten im Projekt.
* Module und Bibliotheken:
  + Backend: Express, Body-Parser, CORS (etc.).
  + Frontend: Vue-Router, Axios (etc.).

# Setup und Installation

## Voraussetzungen

Bevor Sie mit der Installation beginnen, stellen Sie sicher, dass die folgenden Softwarekomponenten auf Ihrem Rechner installiert sind:

* **Node.js (Version 18 oder höher)**
* **npm (Node Package Manager)** – Wird automatisch mit Node.js installiert.
  + Alternativ kann auch **Yarn** verwendet werden.

## Schritt-für-Schritt-Anleitung

**1. Repository klonen**

Öffnen Sie ein Terminal und führen Sie den folgenden Befehl aus, um das Projekt-Repository zu klonen:

git clone https://github.com/MasterLini/HomeSphere

**2. Abhängigkeiten installieren**

Führen sie nun im root Ordner (HomeSphere) den angegebenen Command aus und installieren Sie die erforderlichen Abhängigkeiten:

npm install:all

**3. Backend-Umgebungsvariablen festlegen**

Im Verzeichnis backend muss eine Environment-Datei erstellt werden, die folgende Parameter enthält:

MONGO\_URI=mongodb://localhost:27017/homesphere

PORT=3000

JWT\_SECRET=your\_jwt\_secret

SMTP\_USER=test@mail.com

SMTP\_PASS=your\_smtp\_pass

EMAIL\_FROM=HomeSphere <test@mail.com>

**4. Backend starten**

Nach dem Installieren der Abhängigkeiten und dem Festlegen der Umgebungsvariablen, starten Sie das Backend mit dem folgenden Befehl:

npm run dev

**6. Frontend-Umgebungsvariablen festlegen**

Im Verzeichnis frontend muss eine Environment-Datei erstellt werden, die folgende Parameter enthält:

VUE\_APP\_SERVER\_IP=localhost

VUE\_APP\_SERVER\_PORT=3000

**7. Frontend starten**

Nach dem Installieren der Abhängigkeiten und dem Festlegen der Umgebungsvariablen, starten Sie das Frontend mit dem folgenden Befehl:

npm run serve

**8. Anwendung im Webbrowser navigieren**

http://localhost:8080

# Features und Funktionalität

* **Einkaufslisten**
  + Nutzer können Einkaufslisten erstellen, verwalten und Produkte hinzufügen oder entfernen.
  + Listen sind für alle Mitglieder einer Familie sichtbar.
  + Synchronisation in Echtzeit für bessere Zusammenarbeit.
* **Familienmanagement**
  + Möglichkeit, eine Familie zu erstellen und Mitglieder hinzuzufügen.
  + Gemeinsame Verwaltung von Einkaufslisten und To-Do-Listen.
  + Rollenbasierte Berechtigungen (Admin, Mitglied).
* **To-Do-Listen**
  + Erstellung von Aufgaben mit Fälligkeitsdatum.
  + Delegation von Aufgaben an Familienmitglieder.
  + Erinnerungsfunktionen für anstehende Aufgaben.

# HomeSphere Frontend – Übersicht und Grundlagen

## 1. Allgemeiner Überblick

Das Frontend von HomeSphere ist eine Single-Page-Anwendung, die mit Vue.js entwickelt wurde. Mithilfe von Vue Router werden verschiedene Seiten (Views) und Komponenten zu einer dynamischen Benutzeroberfläche zusammengeführt. Die Kommunikation mit dem Backend erfolgt über eine zentrale API-Schicht, die mit Axios realisiert ist.

Das Frontend ist modular aufgebaut und folgt dem Prinzip der Wiederverwendbarkeit. Komponenten (wie Formularfelder, Navigationsleisten oder ToDo-Elemente) werden in eigenen Dateien definiert und in Views eingebunden. Dies ermöglicht eine saubere Trennung von Layout, Logik und Datenkommunikation.

## 2. Ordnerstruktur

Die typische Struktur des Frontend-Projekts sieht folgendermaßen aus:Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung : Ordnerstruktur Frontend

## 3. Grundkonzepte

### 3.1 API-Schicht

Die API-Schicht befindet sich im Ordner `src/api`. Hier werden zentrale API-Funktionen definiert, die mit dem Backend kommunizieren.

### 3.2 Komponenten

Im Ordner `src/components` sind wiederverwendbare UI-Elemente definiert, wie z. B. Login-Formulare, ToDo-Elemente oder Navigationsleisten.

### 3.3 Views (Seiten)

Die Views im Ordner `src/views` repräsentieren einzelne Seiten der Anwendung, wie z. B. HomeView, AuthView oder ShoppingListView.

### 3.4 Routing

Das Routing wird mit Vue Router umgesetzt, um die Navigation zwischen den Seiten zu ermöglichen. Dabei werden auch Route Guards genutzt, um den Zugriff auf bestimmte Seiten einzuschränken.

### 3.5 Styling und Layout

Das Styling erfolgt über CSS (oder SCSS) und sorgt für ein responsives Design, das auf verschiedenen Bildschirmgrößen gut funktioniert.

### 3.6 Hauptanwendung

Die Hauptanwendung besteht aus der `App.vue`-Datei (Root-Komponente) und `main.js`, die die Vue-Instanz erstellt und die App startet.

## 4. Zusammenfassung

Das Frontend von HomeSphere ist modular aufgebaut und leicht erweiterbar. Eine klare Trennung von API-Schicht, Komponenten und Views ermöglicht eine saubere Struktur. Das Routing sorgt für eine benutzerfreundliche Navigation, und das responsive Design stellt sicher, dass die Anwendung auf verschiedenen Geräten optimal funktioniert.

# HomeSphere Backend – Übersicht und Grundlagen

## 1. Allgemeiner Überblick

Das HomeSphere Backend ist eine RESTful API, die mit Node.js und Express entwickelt wurde. Als Datenbank dient MongoDB, auf das über Mongoose zugegriffen wird. Die API stellt Endpunkte für verschiedene Bereiche bereit:

**Benutzerverwaltung und Authentifizierung:**

Registrierung, Login, E-Mail-Verifizierung, Profilbearbeitung, Passwort-Reset und Logout (mit JSON Web Tokens, kurz JWT).

**Familienverwaltung:**

Erstellen und Verwalten von Familien, Beitritt zu einer Familie mittels Join-Codes, Versenden von Einladungen, Rollenänderungen (Promote/Demote) und Entfernen von Mitgliedern.

**ToDo-Verwaltung:**

Erstellen, Abrufen, Aktualisieren und Löschen von ToDo-Elementen, die einer Familie zugeordnet sind.

**Einkaufsliste:**

Verwaltung von Einkaufsartikeln (Erstellen, Abrufen, Aktualisieren, Löschen).

Zudem wird Nodemailer verwendet, um E-Mails (z. B. zur Verifizierung oder zum Passwort-Reset) zu versenden, und Winston übernimmt das Logging.

## 2. Grundlegende Konzepte und Komponenten

### 2.1 Models (Modelle)

Modelle definieren die Datenstruktur in der Datenbank. Mit Hilfe von Mongoose werden Schemata erstellt, die bestimmen, welche Felder ein Dokument besitzt und wie diese validiert werden.

**Beispiele in HomeSphere:**

User (Benutzer), Family (Familie), Todo, ShoppingItem (Einkaufsartikel)

### 2.2 Controller

Controller enthalten die Geschäftslogik der Anwendung. Sie verarbeiten die Anfragen der Clients, interagieren mit den Modellen und senden die entsprechenden Antworten zurück.

### 2.3 Routes (Routen)

Routen ordnen HTTP-Anfragen (wie GET, POST, PUT, DELETE) den entsprechenden Controller-Funktionen zu.

### 2.4 Middlewares

Middlewares sind Funktionen, die während des Anfrage-Antwort-Zyklus ausgeführt werden (z.B. Authentifizierung, d.h. token Abfrage).

### 2.5 Utilities

Utilities sind Hilfsfunktionen, die häufig benötigte Aufgaben erledigen (z.B. Mails versenden).

## 3. Ordnerstruktur

Ein Bild, das Text, Screenshot, Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung : Ordnerstruktur Backend

## 4. Zusammenfassung

Das Backend von HomeSphere ist modular aufgebaut und folgt einem klaren MVC-Prinzip (Model-View-Controller). Modelle definieren die Datenstrukturen, Controller enthalten die Logik, Routen verbinden HTTP-Anfragen mit den Controller-Funktionen, Middlewares übernehmen Aufgaben wie Authentifizierung und Fehlerbehandlung, und Utilities unterstützen häufig benötigte Funktionen.

# Abbildungsverzeichnis

[Abbildung 1: https://iconduck.com/icons/27728/node-js 6](#_Toc191415323)

[Abbildung 2: https://en.m.wikipedia.org/wiki/File:Vue.js\_Logo\_2.svg 6](#_Toc191415324)

[Abbildung 3: https://de.m.wikipedia.org/wiki/Datei:MongoDB\_Logo.svg 6](#_Toc191415325)

[Abbildung 4: Ordnerstruktur Frontend 10](#_Toc191415326)

[Abbildung 5: Ordnerstruktur Backend 13](#_Toc191415327)