PRIMUS

ABSTRACT

Primus ist ein modernes Mitgliederverwaltungssystem, das im Rahmen eines Schulprojekts entwickelt wurde. Es soll die veraltete Excel-97-Version für den fiktiven Schützenverein "Der glühende Colt" ersetzen. Mit einer benutzerfreundlichen Oberfläche vereinfacht Primus die Mitgliederverwaltungsaufgaben mühelo

Table of Contents

Deutsche Version

Projekt – Allgemein

Gruppe (5 Mitglieder):

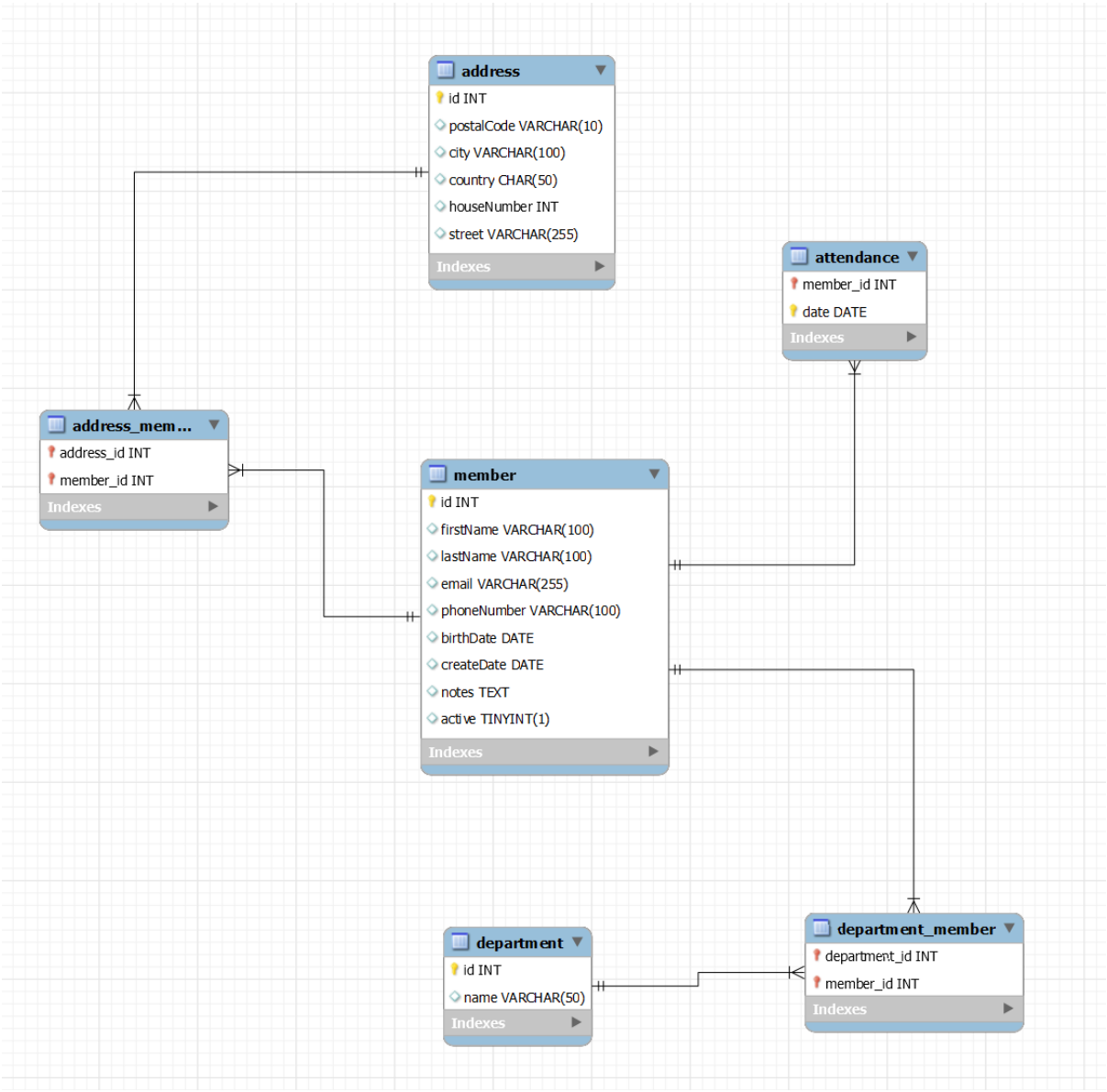
- Sascha
- Mats
- Nathanael
- Sebastian
- Jonas

Vorgehensweise

Da Mats und Sascha die einzigen Auszubildenden mit der Fachrichtung Anwendungsentwicklung sind, haben wir versucht, die Richtung vorzugeben. Wir verfügen bereits über umfangreiche Kenntnisse in C++ aus unserer bisherigen Arbeit. Dennoch wollten wir von Anfang an eine webbasierte Oberfläche erstellen. Obwohl C++ heutzutage nicht unbedingt die erste Wahl für ein Backend ist, verwenden große Unternehmen wie Google immer noch eher solche Low-Level-Programmiersprachen, da sie immer noch schneller und effizienter arbeiten als beispielsweise Python. Aus diesem Grund haben wir uns für ein Webframework für C++ entschieden und eine REST-API erstellt - Oatpp. Oatpp bietet eine einfache Verwaltung von API-Endpunkten und ist daher perfekt für unsere Zwecke geeignet. Außerdem bietet es moderne Elemente wie DTOs, mit denen sich die Webentwicklung im typischen C++-Stil abstrahieren lässt. Die beiden zusätzlichen Module, oatpp-swagger (Integration von Swagger-UI) und oatpp-sqlite (Wrapper um SQLite), machen das Arbeiten mit dem Framework und anderen Komponenten noch angenehmer. Die Entwicklung unseres webbasierten Frameworks wurde maßgeblich durch die Swagger UI beeinflusst, und wir sind froh, uns für diese Lösung entschieden zu haben.

Mats und Sascha übernahmen hierbei die Führung, Plannung, Verwaltung und trugen maßgeblich zur Arbeit bei. Aufgrund ihrer Kompetenz und der Komplexität wurde Sascha im Backend eingesetzt und Mats im Frontend. Mats hat das Frontend geprägt, während Sascha den finalen Schliff insgesamt durchgeführt hat. Ein Github-Repository und ein Projekt-Discord wurden frühzeitig von Sascha erstellt, administriert und moderiert. Mats hat eine Fork vom Repository erstellt und darüber das Frontend parallel entwickelt.

ER-Modell


address
id INT
postalCode VARCHAR(10)
city VARCHAR(100)
country CHAR(50)
houseNumber INT
street VARCHAR(255)
Indexes
attendance
member_id INT
date DATE
Indexes
address_mem...
address_id INT
member_id INT
Indexes
member
id INT
firstName VARCHAR(100)
lastName VARCHAR(100)
email VARCHAR(255)
phoneNumber VARCHAR(100)
birthDate DATE
createDate DATE
notes TEXT
active TINYINT(1)
Indexes
department
id INT
name VARCHAR(50)
Indexes
department_member
department_id INT
member_id INT
Indexes

Komponenten

Frontend

- HTML
- CSS
- Bootstrap - Frontend-Framework
- JavaScript
- jQuery - JavaScript-Bibliothek
- Fetch API - Zum Senden von HTTP-Anfragen
- Popper.js - Zur Erstellung von Popovers und Tooltips

Backend

- C++ - Programmiersprache für das Backend
- oatpp - C++ Web Framework
- oatpp-swagger - Werkzeug zur API-Dokumentation
- oatpp-sqlite - SQLite-Erweiterung für oatpp
- SQLite - Datenbank für das Backend

Software

- [Microsoft Visual Studio 2022](https://visualstudio.microsoft.com/de/downloads/)

- [Obsidian](https://obsidian.md/)

- [Stable Diffusion in Form von AUTOMATIC1111's webui](https://github.com/AUTOMATIC1111/stable-diffusion-webui)

- [CMake](https://cmake.org/)

- [Office 365 (Schulkonto)](https://www.office.com/)

- [Photopea (Photoshop kostenlos)](https://www.photopea.com/)

- MySql Workbench

- Doxygen

- Notepad++

Bildquellen

Shooting-Club-Logo

[[Logo - Der glühende Colt V1.jpg]]

Das Logo wurde mithilfe von [Stable Diffusion](https://github.com/AUTOMATIC1111/stable-diffusion-webui) generiert und mit [Photopea](https://www.photopea.com/) überarbeitet. Im Folgenden sind zu den Arbeitsschritten diesbezüglich nähere Details.

Erstmalige Generierung (2023.11.28)

Erstgenerierung des Bildes: [[Logo - Der glühende Colt RAW.png]]

Parameter

A shooting club logo without text<lora:LCM_SDXL:1.0>

Negative prompt: text

Steps: 20, Sampler: DPM++ 2M Karras, CFG scale: 7, Seed: 1284572753, Size: 512x512, Model hash: 4e089bea6e, Model: realcartoon3d_v8, Denoising strength: 0.6, Lora hashes: "LCM_SDXL: 3d18b05e4f56", Version: v1.6.0-2-g4afaaf8a

postprocessing

Postprocess upscale by: 5, Postprocess upscaler: ESRGAN_4x

extras

Postprocess upscale by: 5, Postprocess upscaler: ESRGAN_4x

Zeiterfassung (monatlich)

Personen	November 2023	Dezember 2023	Januar 2024	Feburar 2024	März 2024	April 2024	Gesamt-Aufwand
Sascha	26	15	9	9	40	48	147
Mats	16	10	6	9	35	45	121
Nathanael	0	4	2	0	0	0	6
Jonas	0	4	2	0	0	0	6
Sebastian	0	4	2	0	0	0	6

Rechnung

Stundenlohn: 45 Euro

Rechnung:

Sascha: 45€/Std * x 147 = 6.615€

Mats: 45€/Std * 121 = 5.445€

Nathanael: 45€/Std * 6 = 270€

Jonas: 45€/Std * 6 = 270€

Sebastian: 45€/Std * 6 = 270€

Gesamt: 12.870 Euro

Installation

Die Installation wurde bisher ausschließlich unter Windows getestet. Im Folgenden finden Sie eine Anleitung für die Windows-Installation. Theoretisch ist der Code jedoch plattformunabhängig und sollte auch unter Linux-Distributionen problemlos funktionieren.

Anforderungen

- Microsoft Visual Studio (getestet mit Visual Studio 2022 Community)

- CMake

- Git

Ort zum Runterladen/Installieren auswählen

PowerShell aufmachen und zu entsprechenden Verzeichnis navigieren.

Folgende Befehle in chronologischer Reihenfolge ausführen:

git clone https://github.com/oatpp/oatpp.git

cd oatpp\

mkdir build

cd build\

cmake ..

cmake --build . --target INSTALL

cd ..

cd ..

git clone https://github.com/oatpp/oatpp-swagger.git

cd oatpp-swagger

mkdir build

```
cd build\

cmake ..
cmake --build . --target INSTALL

cd ..
cd ..

git clone https://github.com/oatpp/oatpp-sqlite.git
cd oatpp-sqlite
mkdir build
cd build\

cmake .. -DOATPP_SQLITE_AMALGAMATION=ON
cmake --build . --target INSTALL

git clone https://github.com/theSaschaM/BSFI22D_Primus.git
cd BSFI22D_Primus
cd PrimusSvr
mkdir build
cd build\

cmake ..
cmake --build .

```

Die ausführbare Datei (Exe-Datei) befindet sich nun im Verzeichnis BSFI22D_Primus\PrimusSvr\bin\Debug. Da die CMake-Konfiguration bisher noch unvollständig ist, müssen die Dateien manuell in das übergeordnete Verzeichnis verschoben werden. Kopieren Sie dazu alle Dateien aus BSFI22D_Primus\PrimusSvr\bin\Debug in BSFI22D_Primus\PrimusSvr\bin\. Anschließend kann der Ordner "Debug" gelöscht werden.

Nachdem dies erledigt ist, kann die PrimusSvr.exe ausgeführt werden. Öffnen Sie dazu einen Webbrowser und geben Sie http://localhost:8000 ein. Sie werden automatisch zur Weboberfläche weitergeleitet. Die Endpunkt-Dokumentation, die mit Swagger erstellt wurde, ist unter http://localhost:8000/swagger/ui/ verfügbar.

Bitte beachten Sie, dass wir keine Authentifizierungssysteme in diese Implementierung einer Mitgliederverwaltung integriert haben. Aus diesem Grund empfehlen wir dringend, diese Version nicht auf einem öffentlichen Server zu hosten.

English Version

Projekt – Allgemein

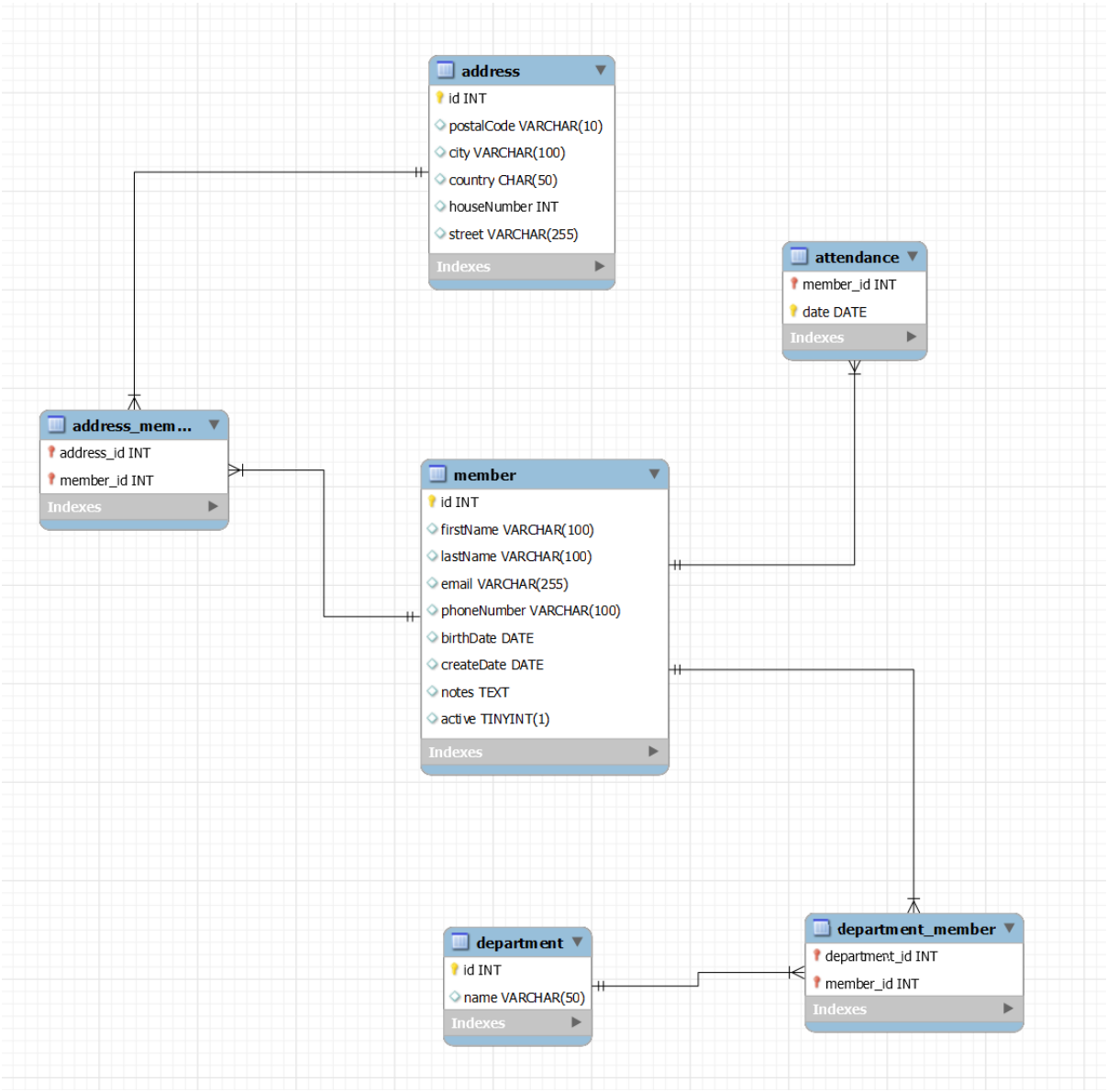
Group (5 members):

- Sascha
- Mats
- Nathanael
- Sebastian
- Jonas

Approach

Since Mats and Sascha are the only trainees specializing in application development, we tried to set the direction. We already have extensive knowledge in C++ from our previous work. However, we wanted to create a web-based interface from the outset. Although C++ is not necessarily the first choice for a backend nowadays, large companies like Google still prefer such low-level programming languages because they still work faster and more efficiently than, for example, Python. For this reason, we decided to use a web framework for C++ and created a REST API - Oatpp. Oatpp provides easy management of API endpoints and is therefore perfect for our purposes. It also offers modern elements like DTOs, which abstract web development in the typical C++ style. The two additional modules, oatpp-swagger (integration of Swagger-UI) and oatpp-sqlite (wrapper around SQLite), make working with the framework and other components even more enjoyable. The development of our web-based framework was significantly influenced by Swagger UI, and we are glad to have chosen this solution. Mats and Sascha took the lead, planning, managing, and contributing significantly to the work. Due to their expertise and the complexity involved, Sascha was assigned to the backend and Mats to the frontend. Mats shaped the frontend, while Sascha provided the final touch overall. A Github repository and a project Discord were created, administered, and moderated early on by Sascha. Mats created a fork from the repository and developed the frontend in parallel through it.

ER-Model


address
id INT
postalCode VARCHAR(10)
city VARCHAR(100)
country CHAR(50)
houseNumber INT
street VARCHAR(255)
Indexes
attendance
member_id INT
date DATE
Indexes
address_mem...
address_id INT
member_id INT
Indexes
member
id INT
firstName VARCHAR(100)
lastName VARCHAR(100)
email VARCHAR(255)
phoneNumber VARCHAR(100)
birthDate DATE
createDate DATE
notes TEXT
active TINYINT(1)
Indexes
department
id INT
name VARCHAR(50)
Indexes
department_member
department_id INT
member_id INT
Indexes

Components

Frontend

HTML

CSS

- Bootstrap - Frontend Framework
- JavaScript
- jQuery - JavaScript Library
- Fetch API - For sending HTTP requests
- Popper.js - For creating popovers and tooltips

Backend

- C++ - Programming language for the backend
- oatpp - C++ Web Framework
- oatpp-swagger - Tool for API documentation
- oatpp-sqlite - SQLite extension for oatpp
- SQLite - Database for the backend

Software

- Microsoft Visual Studio 2022
- Obsidian
- Stable Diffusion in the form of AUTOMATIC1111's webui
- CMake
- Office 365 (school account)
- Photopea (free Photoshop alternative)
- MySql Workbench
- Doxygen
- Notepad++

Picture sources

Shooting-Club-Logo

[[Logo - Der glühende Colt V1.jpg]]

The logo was generated using Stable Diffusion and revised with Photopea. Below are further details regarding the steps involved in this process.

Parameters:

A shooting club logo without text<lora:LCM_SDXL:1.0>

Negative prompt: text

Steps: 20, Sampler: DPM++ 2M Karras, CFG scale: 7, Seed: 1284572753, Size: 512x512, Model hash: 4e089bea6e, Model: realcartoon3d_v8, Denoising strength: 0.6, Lora hashes: "LCM_SDXL: 3d18b05e4f56", Version: v1.6.0-2-g4afaaf8a

postprocessing

Postprocess upscale by: 5, Postprocess upscaler: ESRGAN_4x

extras

Postprocess upscale by: 5, Postprocess upscaler: ESRGAN_4x

Time tracking (monthly)

Persons	november 2023	december 2023	jarnuary 2024	february 2024	march 2024	april 2024	Total effort
Sascha	26	15	9	9	40	48	147
Mats	16	10	6	9	35	45	121
Nathanael	0	4	2	0	0	0	6
Jonas	0	4	2	0	0	0	6
Sebastian	0	4	2	0	0	0	6

Invoice:

Invoice Hourly rate: 45 euros

Sascha: 45€/hr * 147 hrs = 6,615€
Mats: 45€/hr * 121 hrs = 5,445€
Nathanael: 45€/hr * 6 hrs = 270€
Jonas: 45€/hr * 6 hrs = 270€
Sebastian: 45€/hr * 6 hrs = 270€

Total: 12,870 euros

Installation

The installation process has been tested exclusively on Windows. Below is a guide for installing on Windows. However, theoretically, the code is platform-independent and should also work smoothly on Linux distributions.

Requirements:

- Microsoft Visual Studio (tested with Visual Studio 2022 Community)
- CMake
- Git

Steps to Install:

Choose a location to download/install the files.

Open PowerShell and navigate to the corresponding directory.

Execute the following commands in chronological order:

bashCopy code

git clone https://github.com/oatpp/oatpp.git

cd oatpp\

mkdir build

cd build\

cmake ..

cmake --build . --target INSTALL

cd ..

cd ..

git clone https://github.com/oatpp/oatpp-swagger.git

cd oatpp-swagger

mkdir build

cd build\

cmake ..

cmake --build . --target INSTALL

cd ..

```
cd ..

git clone https://github.com/oatpp/oatpp-sqlite.git

cd oatpp-sqlite

mkdir build

cd build\

cmake .. -DOATPP_SQLITE_AMALGAMATION=ON

cmake --build . --target INSTALL

git clone https://github.com/theSaschaM/BSFI22D_Primus.git

cd BSFI22D_Primus

cd PrimusSvr

mkdir build

cd build\

cmake ..

cmake --build .
```

The executable file (.exe) is now located in the directory BSFI22D_Primus\PrimusSvr\bin\Debug. Since the CMake configuration is still incomplete, the files need to be manually moved to the parent directory. To do this, copy all files from BSFI22D_Primus\PrimusSvr\bin\Debug to BSFI22D_Primus\PrimusSvr\bin\. Afterwards, the Debug folder can be deleted.

Once this is done, you can execute PrimusSvr.exe. Open a web browser and enter http://localhost:8000. You will be automatically redirected to the web interface. The endpoint documentation, created with Swagger, is available at http://localhost:8000/swagger/ui/.

Please note that we have not integrated any authentication systems into this implementation of membership management. Therefore, we strongly recommend against hosting this version on a public server.