todo

Dokumentation der Anwendung

Full-Stack-Webanwendungen
Sommersemester 2023

Lucas Schiessl123
lucas.schiessl@hs-augsburg.de
Informatik

Hannes Ziereis
hannes.ziereis@hs-augsburg.de
Informatik

Christoph Herb
christoph.herb@hs-augsburg.de
Informatik

Zeichen (mit Leerzeichen)

Contents

1 Allgemein **2**
- 1.1 Beschreibung . . . 2
- 1.2 Organisation . . . 2
- 1.3 Aufbau . . . 2

2 Frontend **3**
- 2.1 Design . . . 3
- 2.2 Funktionen . . . 4

3 Backend **4**
- 3.1 Funktionen . . . 4
- 3.2 Tests . . . 4

4 Datenbank **4**

5 Anhang **5**
- 5.1 GitLab Issues . . . 5
- 5.2 API Dokumentation . . . 7

1 Allgemein

1.1 Beschreibung

* paar infos über die Anwendung * cooles features

1.2 Organisation

* auf gitlab issues zeigen (anhang) * feature based entwicklung erwähnen

1.3 Aufbau

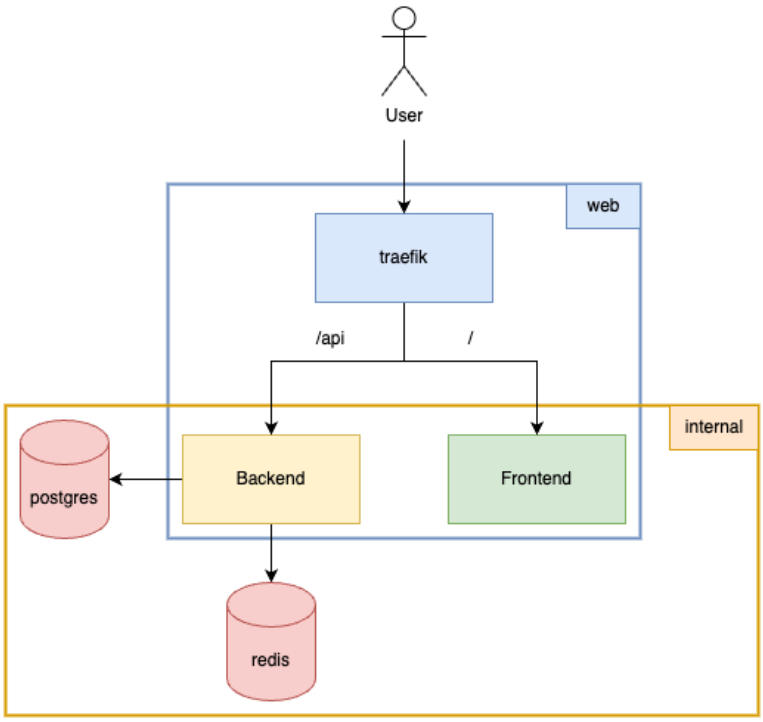

Figure 1: Architektur der deployten Anwendung

2 Frontend

2.1 Design

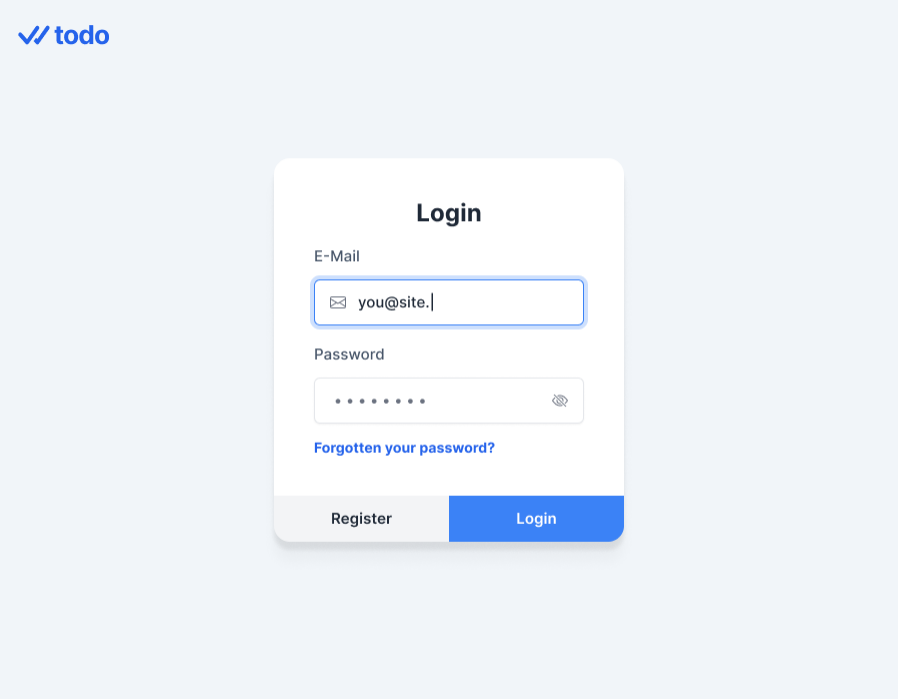

Figure 2: Figma Design des Logins

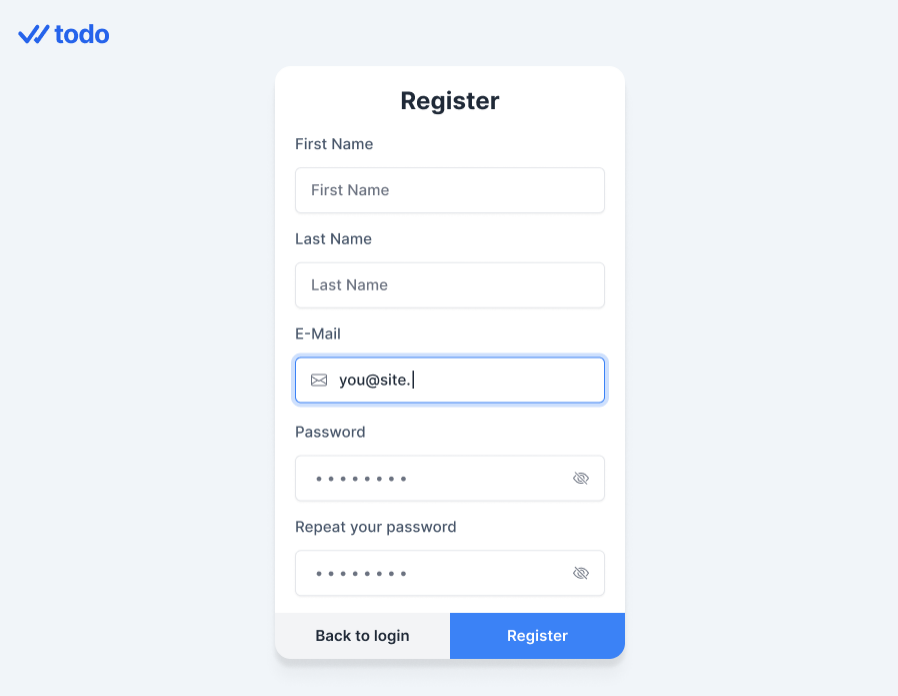

Figure 3: Figma Design des Registrierungsformulars

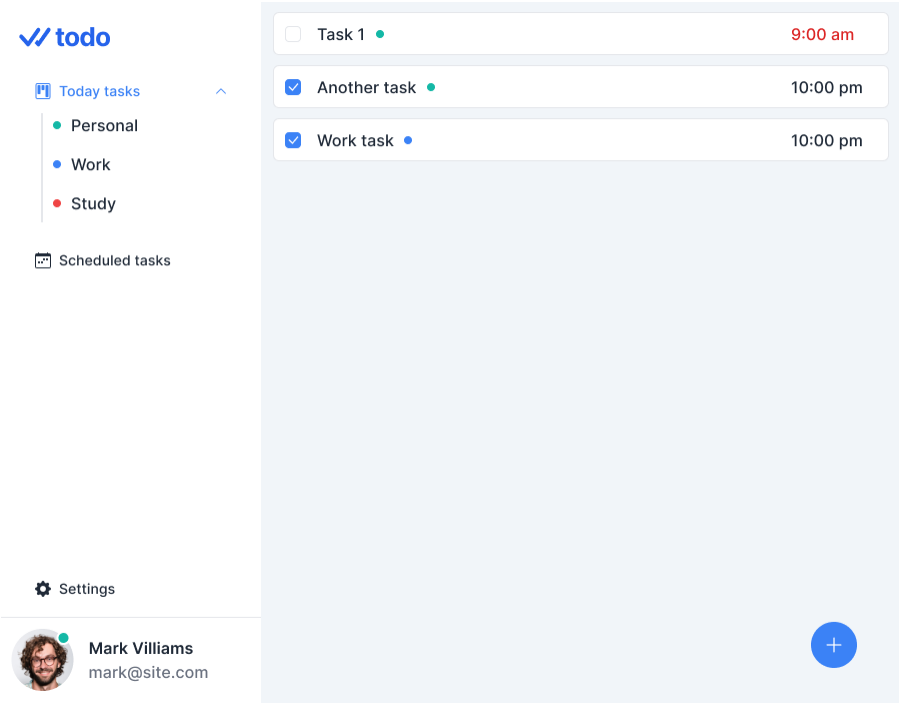

Figure 4: Figma Design der Hauptseite

2.2 Funktionen

* kernfeatures

3 Backend

3.1 Funktionen

* features

3.2 Tests

* cucumber tests erwähnen

4 Datenbank

* ER diagramm

5 Anhang

5.1 GitLab Issues

Title	Assignee
FE Install Tailwindcss	Lucas Schießl
BE Create blueprint for backend	Christoph Herb
BE Add postgresql	Lucas Schießl
BE Add refresh endpoint	Christoph Herb
BE Add getting started documentation	Christoph Herb
DOC Create documentation	NaN
FE Create Figma Design	Christoph Herb
BE Create Database model	Hannes Ziereis
BE Implement swagger docs or create REST-API diagram manually	Lucas Schießl
FE Implement basic design	Lucas Schießl
FE First interaction with BE	Christoph Herb
FE Implement authentication with BE	Lucas Schießl
FE Cleanup frontend	Christoph Herb
FE Implement Nav Bar and design Login Form	Lucas Schießl
BE Endpoints for tasks	Hannes Ziereis
FE Interaction with tasks	Hannes Ziereis
BE Implement better logger	Christoph Herb
BE Hash password before saving into DB	Christoph Herb
FE Implement login in frontend	Lucas Schießl
FE style login form	Lucas Schießl
FE create Registration Form	Lucas Schießl
BE Add logout endpoint	Christoph Herb
FE Add error handling to login and registration form	Lucas Schießl
BE Fix tasks endpoint, swagger docs and cucumber tests	Christoph Herb
BE, FE Add docker support	Christoph Herb
FE Call Logout and refresh endpoints from frontend	Lucas Schießl
BE Fix bugs in backend	Christoph Herb
DOC Add postman collection	Christoph Herb
FE Add more functionality to task bar	Lucas Schießl
FE Add functionality to settings page	Lucas Schießl
FE Refactoring	Lucas Schießl
FE Refresh does not work as expected	Lucas Schießl
BE, FE Fix docker prod environment	Christoph Herb
BE Error messages for frontend	Christoph Herb
BE Sorting of tasks with date	Christoph Herb
BE Add filtering for categories	Christoph Herb
FE Add support for multiple languages	Lucas Schießl
FE Show taskdetails in task list	Hannes Ziereis
FE Refactor AppTask into its components	Hannes Ziereis

Continued on next page

Title	Assignee
BE Move password hashing to postgres	Christoph Herb
FE Cleanup	Christoph Herb
BE Generate secrets instead of hard coding	Christoph Herb
BE Add optional Tags to Tasks	Lucas Schießl
FE Fix token refresh endless loop	Christoph Herb
FE Add calendar for scheduled tasks	Christoph Herb
FE Show Tag in overview and details modal	NaN
FE BUG - Modals opens two times	Christoph Herb
FE Detailed View - Done status doesn't update	Christoph Herb
FE Fix styling of task list	Christoph Herb
FE Edit Task in Detail modal	Hannes Ziereis
FE Improve modal	Christoph Herb

5.2 API Dokumentation

Tasks API

Overview

Backend API for the Tasks service.

MIT

Tags

Auth

Bearer authentication

Me

Me endpoint

Tasks

Task endpoint

User

User endpoint

Security

Type	Name	Scopes
apiKey	BearerAuth	

Paths

POST `/api/v1/auth/login` Login

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links

Code	Description	Links
500	Internal Server error *Content* **application/json**	No Links

GET `/api/v1/auth/verify` Verify logged in user

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

GET `/api/v1/auth/refresh` Refresh token after expiration of your access token

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links

Code	Description	Links
500	Internal Server error *Content* **application/json**	No Links

GET `/api/v1/auth/logout` Logout user

Responses

Code	Description	Links
204	success response	No Links
401	Unauthorized error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

GET `/api/v1/me` Get infos about current logged in user

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

GET `/api/v1/me/tasks` Get tasks of current logged in user

Parameters

Type	Name	Description	Schema
query	**orderBy** *optional*	orderBy	enum (asc,desc)
query	**sortBy** *optional*	sortBy	enum (dueDate,isChecked,createdAt,updatedAt)
query	**tag** *optional*	tag	string
query	**isChecked** *optional*	isChecked	enum (false,true)

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

POST `/api/v1/me/tasks` Create a new Task with current user

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
400	Validation error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

GET `/api/v1/me/tags` Get Tags of tasks of current logged in user

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

GET `/api/v1/tasks` Get all Tasks

Parameters

Type	Name	Description	Schema
query	**orderBy** *optional*	orderBy	enum (asc,desc)
query	**sortBy** *optional*	sortBy	enum (dueDate,isChecked,createdAt,updatedAt)
query	**tag** *optional*	tag	string
query	**isChecked** *optional*	isChecked	enum (false,true)

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
403	Forbidden error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

POST `/api/v1/tasks` Create a new Task

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links

6

Code	Description	Links
400	Validation error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

GET `/api/v1/tasks/{taskId}` Get specific task

Parameters

Type	Name	Description	Schema
path	**taskId** *optional*	Task ID	string

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
403	Forbidden error *Content* **application/json**	No Links
404	NotFound error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

PUT `/api/v1/tasks/{taskId}` Update a Task

Parameters

Type	Name	Description	Schema
path	**taskId** *optional*	Task ID	string

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
400	Bad Request *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
403	Forbidden error *Content* **application/json**	No Links
404	NotFound error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

DELETE `/api/v1/tasks/{taskId}` Delete a Task

Parameters

Type	Name	Description	Schema
path	**taskId** *optional*	Task ID	string

Responses

Code	Description	Links
204	success response	No Links
400	Bad Request *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
403	Forbidden error *Content* **application/json**	No Links
404	NotFound error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

PATCH `/api/v1/tasks/{taskId}/toggle` Toggle the isChecked field

Parameters

Type	Name	Description	Schema
path	**taskId** *optional*	Task ID	string

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
400	Bad Request *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
403	Forbidden error *Content* **application/json**	No Links
404	NotFound error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

GET `/api/v1/users` Get all users

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links

Code	Description	Links
403	Forbidden error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

POST `/api/v1/users` Create a new user

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
400	Validation error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

GET `/api/v1/users/{userId}` Get user by id

Parameters

Type	Name	Description	Schema
path	**userId** *optional*	userId	string

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links
403	Forbidden error *Content* **application/json**	No Links
404	Not Found error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

PUT `/api/v1/users/{userId}` Update user by id

Parameters

Type	Name	Description	Schema
path	**userId** *optional*	userId	string

Responses

Code	Description	Links
200	success response *Content* **application/json**	No Links
401	Unauthorized error *Content* **application/json**	No Links

Code	Description	Links
403	Forbidden error *Content* **application/json**	No Links
404	Not Found error *Content* **application/json**	No Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

DELETE `/api/v1/users/{userId}` Delete user by id

Parameters

Type	Name	Description	Schema
path	**userId** *optional*	userId	string

Responses

Code	Description	Links
204	success response	No Links
401	Unauthorized error *Content* **application/json**	No Links
403	Forbidden error *Content* **application/json**	No Links
404	Not Found error *Content* **application/json**	No Links

Code	Description	Links
500	Internal Server error *Content* **application/json**	No Links

Type	Name	Scopes
apiKey	BearerAuth	

Components

Schemas

BaseError

BaseError

Properties

Name	Description	Schema
errorCode *optional*	ErrorCode	number
errorMessage *optional*	ErrorMessage	string
details *optional*	Details	< ErrorSchema > array

LoginSchema

LoginSchema

Properties

Name	Description	Schema
email *required*	Email	string
password *optional*	Password	string

AuthLoginSchema

AuthLoginSchema

Properties

Name	Description	Schema
tokenType *optional*	Type of token	string
accessToken *optional*	Access token	string

JwtPayloadSchema

JwtPayloadSchema

Properties

Name	Description	Schema
userId *optional*	UserId	string
email *optional*	Email	string
isAdmin *optional*	IsAdmin	boolean

ErrorSchema

ErrorSchema

Properties

Name	Description	Schema
field *optional*	Field	string
value *optional*	Value *nullable*	string
replyCode *optional*	ReplyCode *nullable*	number
replyMessage *optional*	ReplyMessage *nullable*	string

CreateTaskSchema

CreateTaskSchema

Properties

Name	Description	Schema
title *required*	Title	string
userId *required*	UserId	string

Name	Description	Schema
description *optional*	Description	string
dueDate *optional*	Due date	string (date-time)
tag *optional*	Tag	string

CreateTaskMeSchema

CreateTaskMeSchema

Properties

Name	Description	Schema
title *required*	Title	string
description *optional*	Description	string
dueDate *optional*	Due date	string (date-time)
tag *optional*	Tag	string

UpdateTaskSchema

UpdateTaskSchema

Properties

Name	Description	Schema
title *optional*	Title *nullable*	string
description *optional*	Description *nullable*	string
dueDate *optional*	Due date *nullable*	string (date-time)
isChecked *optional*	Is completed? *nullable*	boolean
tag *optional*	Tag *nullable*	string

ReadTaskSchema

ReadTaskSchema

Properties

Name	Description	Schema
id *optional*	ID	string
userId *optional*	UserId	string
title *optional*	Title	string
description *optional*	Description	string
dueDate *optional*	Due date	string (date-time)
isChecked *optional*	Is Checked	boolean
tag *optional*	Tag	string

GetTasksWithSpecifiedTagSchema

GetTasksWithSpecifiedTagSchema

Properties

Name	Description	Schema
tag *optional*	Tag	string

CreateUserSchema

CreateUserSchema

Properties

Name	Description	Schema
email *required*	Email	string
firstName *required*	First Name	string
lastName *required*	Last Name	string
password *required*	Pasword	string

UpdateUserSchema

UpdateUserSchema

Properties

Name	Description	Schema
email *optional*	Email	string
firstName *optional*	First Name	string
lastName *optional*	Last Name	string
password *optional*	Password	string

ReadUserSchema

ReadUserSchema

Properties

Name	Description	Schema
id *optional*	Id	string
email *optional*	Email	string
firstName *optional*	First Name	string
lastName *optional*	Last Name	string
isAdmin *optional*	IsAdmin	boolean