Oauth 2.0 Template

Gigup Solutions

WE DESIGN INFORMATIONS

Visit us: www.gigup.ch
Follow us on: Linkedin

OAuth 2.0 System Specifications
Verfasser: Carlson Patrick
Email: hello@gigup.ch
Version: 1.0 / 13.06.2023

1 Inhalt:

2 Versionsverlauf

Version	Requirements
Full Auth System 1.0, 03.06.2023 Carlson Patrick *Check*: 05.06.2023	Design, - SPA, Cross-Platform, Userexperience, SEO - UI – Template + Dark & Bright Mode - Database (PSQL), Setup - Separate Frontend from Backend (Modularity) Globals, - Auth-System, Middleware & Access Management - Server Request: Apis + Response Handling - Client: Userstore & Response Handling Visitor, - Landingpage simple Navigation - User Auth: Create Account, Verify email, Reset Password, Login User, - Profilemanagement: Avatar, Name, Password, Transfer Account (Change Email), Delete Account (CASCADE) Admin, - Backpanel (Useramount, etc.)

3 Features

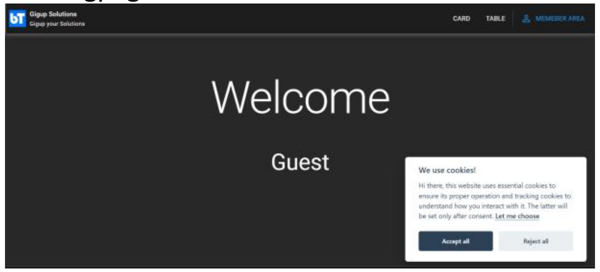
3.1 Globals

Middleware & Authentication
System-Access-Management
	Frontend: Vue 3 (Client)
	☒Userstore Pinia: Client Access & Data Management
	☒Axios: API – Requests to Server
	☒Response Handling: Client <- Server
	☒Cookie-Consent
	
	Backend: Laravel 10 (Server)
	☒Laravel Passport: User Authentication
	☒Middleware: Access-Management / Userauth
	☒API Calls: from Client
	- Received Data (by Client), must be checked for Datatype foreach request
	☒PostgreSQL, Default Migrations
		Userstore {
		  access: {},
		  user: {},
		  actions: () => {
		    setUser(),
		    setToken(),
		    setSession(),
		    removeToken(),
		    removeSession(),
		    removeAdmin(),
		  }
		}

Response Handling	Success: 200,
	Errors
	- **422**, Error occured
	Custom Error Handling
	- **401 && 'email_not_verified'**
	Always remove Session (userstore) & Token, push to '/email-verification'
	- **401 && 'no_admin'**
	Always remove Admin (userstore), push '/dashboard'
	- **401, no access**
	User must login again (Local Token may be set)
	Only removeSession (userstore), push to '/login'
		ResponseHandling: {
		  globalAttributes,
		  errorHandling: (response, router) => {
		    // Custom error Handling
		  },
		  load(),
		  loaded(),
		  sucess(String),
		  error(String | Object) // Includes errorHandling
		}

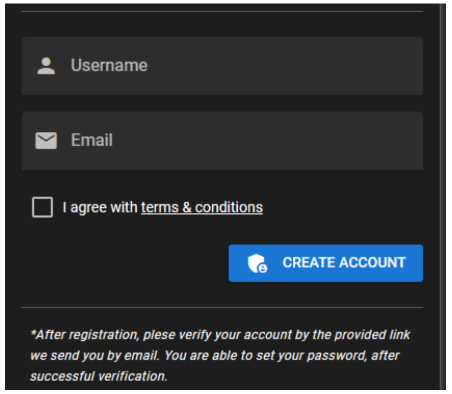
3.2 Visitor

3.2.1 Landingpage

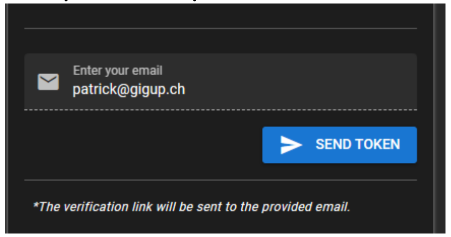
Features	Requirements	API & Data
Landingpage  Gigup Solutions Gigup your Solutions CARD TABLE MEMEBER AREA Welcome Guest Accept all Reject all We use cookies! Hi there, this website uses essential cookies to ensure its proper operation and tracking cookies to understand how you interact with it. The latter will be set only after consent. Let me choose	☒Cookie Consent must be shown and must initialize after user accepted / rejected ☒Userstore - Client, must be initializet, so User / Visitor is able to gain access to certain pages. ☒Visitor can choose between dark and bright mode. ☒Visitor must be able to Navigate by Landingpage ☒SEO Optimization	Only client, yet.

3.2.2 Visitor Accountmanagement

Create Account		

 Username
Email
I agree with terms & conditions
CREATE ACCOUNT*After registration, plese verify your account by the provided link we send you by email. You are able to set your password, after successful verification.
	☒Visitor can create account, after he entered username and email.
	☒Users password will be set, after verifing its email
	☒Set random password (hashed, 255 chars)
	☒ Visitor must agree with terms and conditions.
	☒After successfull registration, he will be redirected to verify it's emai.
		```
		axios.post("/create-account", {
		        'name': String,
		        'email': String,
		        'terms': Boolean,
		    });
		```

Verify Email Request


Enter your email
patrick@gigup.ch
SEND TOKEN
*The verification link will be sent to the provided email.

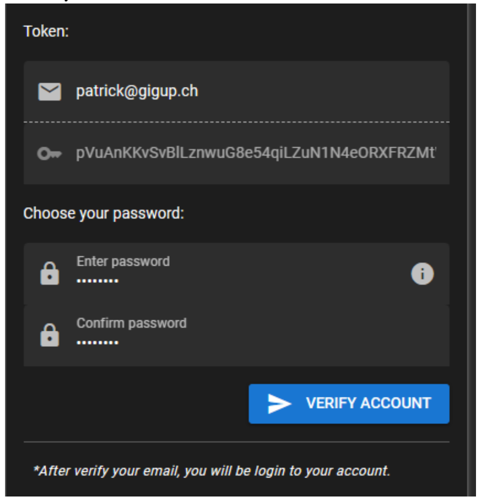
☒ Visitor must klick button send token, to receive the verification token to its email.
☒Token is 255 Character, no hashing required
☒Link must expire
☒Link is signed
☒Email with Token will be sent from our SMTP Email Server

Token: Email Verification

```
Token sent to Useremail:
{signedRoute}/{email}/{token}?{expires}&{signatur}
```

```
axios.post("/email-verification-request", {
        'email': String,
    });
```

Verify Email

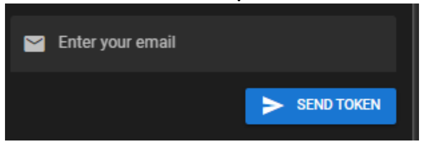

Token:
patrick@gigup.ch
pVuAnKKvSvBlLznwuG8e54qiLZuN1N4eORXFRZMt
Choose your password:
Enter password
••••••••
Confirm password
••••••••
VERIFY ACCOUNT
*After verify your email, you will be login to your account.

See: Token: Email Verification Request

☒User has to verify its account by the token, sent to its mail.
☒ Signedroute, Email, Token, Signatur must be valid! User is not verified yet. He has 5 attemps before baning for 1min
☒Only working, if user is not verified yet.
☒User must enter new password (ac. Requirements), which will be hashed and updated.
☒After successfull verification, token must be set to null
☒After successfull verification email_verified_at must be set to now()
☒Password must succeed Requirements and will be hashed
☒User will be login after successfull verification of email

```
// Redirecting Request from Client-URL
// to Origin-URL (SERVER)
// Server Request to signedRoute
axios.put(signedRoute, {
        'password': String,
        'password_confirmation': String,
    });
```

Password Reset Request



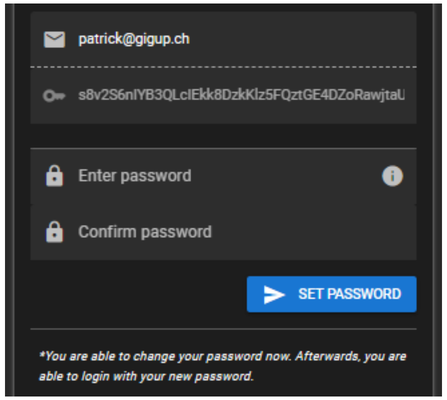
☒Visitor can enter email
☒ Visitor will be redirected to Landingpage
☒We do not inform Visitor, if email is existing
☒Token is 255 Character, no hashing required
☒Email with Token will be sent from our SMTP Email Server
☒We make limit of 5 attemps : 1min Ban

Token: Password reset

```
Token sent to Useremail:
{signedRoute}/{email}/{token}?{expires}&{signatur}
```

```
return axios.post(
"/password-reset-request", {
      'email': String
   });
```

Password Reset


patrick@gigup.ch

s8v2S6nlYB3QLclEkk8DzkKlz5FQztGE4DZoRawjtaU

**You are able to change your password now. Afterwards, you are able to login with your new password.*

See: Token: Password Reset Request

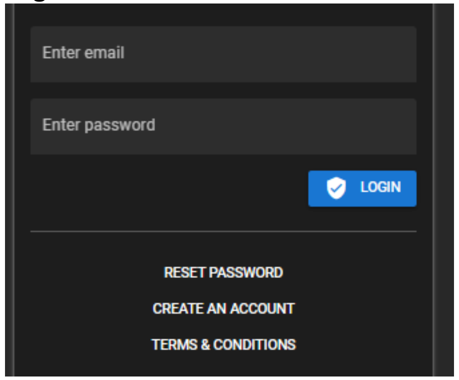
☒ Visitor has to verify its account by the token, sent to its mail.
☒Signedroute, Email, Token, Signatur must be valid! He has 5 attemps before baning for 1min
☒ Visitor must enter new password (ac. Requirements), which will be hashed and updated.
☒After successfull verification, token must be set to null
☒ Visitor will be login after successfull update of ist hashed password

```
return axios.put(
   signedRoute, {
        'password': String,
        'password_confirmation': String
   });
```

3.2.3 Visitor Login

Vistor can create an account, as well as manage his account credentials.

Features	Stories	API & Data

Login  Enter email
Enter password
LOGIN
RESET PASSWORD
CREATE AN ACCOUNT
TERMS & CONDITIONS
	☒Before accessing to Member area, we must check, if Visitor has already a ongoing session in its localstore (Bearer Token set, Laravel Passport).
	☒If Bearer Token exists, the token will be set in HTML Head and Visitor will be redirected to authorization (see «Authorization») as User.
	☒ If Bearer Token is not set, Visitor must manually login by email & password. Attempts: 7 : 1min penalty.
	☒ If login is successfull, Bearer Token will be set in HTML Header and Localstore. Visitor will be redirected to Autorization (see «Authorization»).
	
	☒User email must be verified. If not, redirected to «Verify-Email-Request»
	☒Redirections: *Reset password Create an account,* Terms & Conditions (Popup)
		`axios.post("/login", {`
		`        'email': String,`
		`        'password': String,`
		`    });`
		Response:
		`return response()->json([`
		`     'token' => String,`
		`     'message' => String`
		`], 200);`
		Email not verified:
		`return response()->json([`
		`  'status' => email_not_verified,`
		`  'email' => String,`
		`  'message' => String,`
		`], 401);`

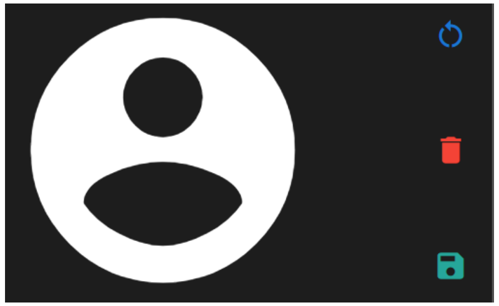
Authorization	☒**For each Userrequest**, check if user is authorized by Bearer Token set in HTML Header (Middleware)
	☒Get default user data, to store in Userstore (user, access)
	☒Check if user is Admin
	☒Every failed Authorization of user will be handled by custom Errorhandling
		`axios.get('/auth')`
		Response:
		`return response()->json([`
		`   'id' => ID,`
		`   'name' => String,`
		`   'avatar' => String | null,`
		`   'email' => String,`
		`   'is_admin' => Boolean`
		`], 200);`

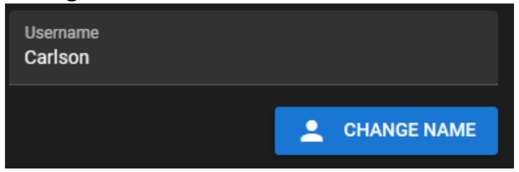
Logout	☒After Logout, we want to delete Token in DB, so Session is removed
	☒Userstore (Client) will be reset to visitor
	☒Client Token in Local store must be removed, see Login: Check Session
		`axios.post('/logout');`
		Response
		`return response()->json([`
		`    'message' => String`
		`], 200);`

3.3 User

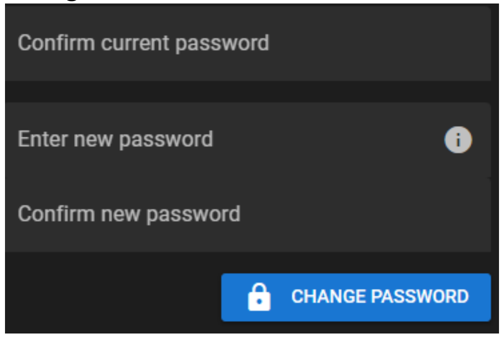
3.3.1 Profilemanagement

Features	Stories	API & Data

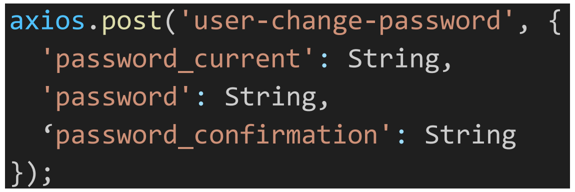
Avatar

	☒User can create, update, delete its avatar
	☒Avatar size < 2MB, only 1 Avatar per User.
	☒Avatar-Image will be saved on Servers «Public Store». The current Avatarname will be saved in users DB. Avatarname must be UNIQUE and can be Nullable!
	☒Only 1 Avatar per User (Update, old avatar must be deleted)
	☐If user is deleting it’s account, avatar must be deleted as well (See «Delete Account»)
		`axios.post('/user-change-avatar',`
		`formData as Object)`
		Formdata:
		- Avatar: Imagedata
		- Delete: ‘1’ ? ‘0’
		
		`return response()->json([`
		`    'message' => String,`
		`], 200);`

Change Name

Username
Carlson
 	
	☒User can change name easily
	☒Name must be updated in Userstore
		`axios.post('/user-change-name', {`
		`        name: String`
		`    });`
		
		`return response()->json([`
		`            'message' => String,`
		`        ], 200);`

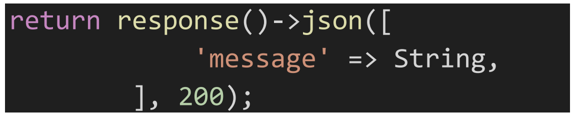
Change Password



☒User can change its password
☒User must confirm by current password
☒New Password must pass requirements
☒New Password will be hashed and saved in DB

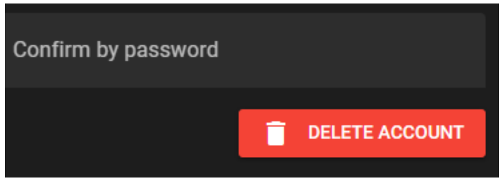


```
axios.post('user-change-password', {
  'password_current': String,
  'password': String,
  ‘password_confirmation': String
});
```

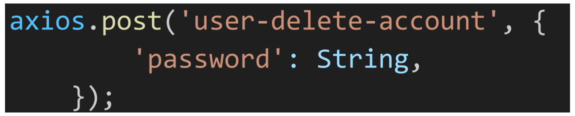


```
return response()->json([
            'message' => String,
        ], 200);
```

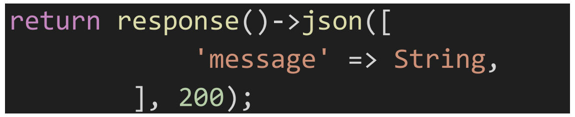
Delete Account



☒User can delete account, by confirming with password
☒All data in DB of current User will be deleted (CASCADE) + Tokens
☒Avatar must be deleted
☒User will be logged out and Session is destroyd



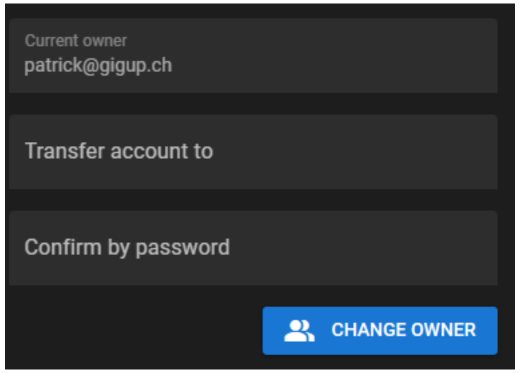
```
axios.post('user-delete-account', {
        'password': String,
    });
```



```
return response()->json([
            'message' => String,
        ], 200);
```

3.3.1.1 Transfer Account

User Transfer Account



Current owner
patrick@gigup.ch

Transfer account to

Confirm by password

CHANGE OWNER

☒User can transfer its mail to new user, with another mail (transfer)
☒User must confirm by password
☒Email with Token will be sent from our SMTP Email Server to the provided email
☒Token must be 255 characters, no hash required
☒Link must expire, Link is signed
☒Email_verified_at must set to NULL
☒Before changing current email, new user has to verify his new email adress. Old user is still able to undone its transfer, by verifying its old email again see «Make Transfer undone»
☒After successfull request, user will be logged out & Bearer Token is removed

Token: New Email Verification

```
{signedRoute}/{email}/{token}/
    {transfer}?{expires}&{signature}
```

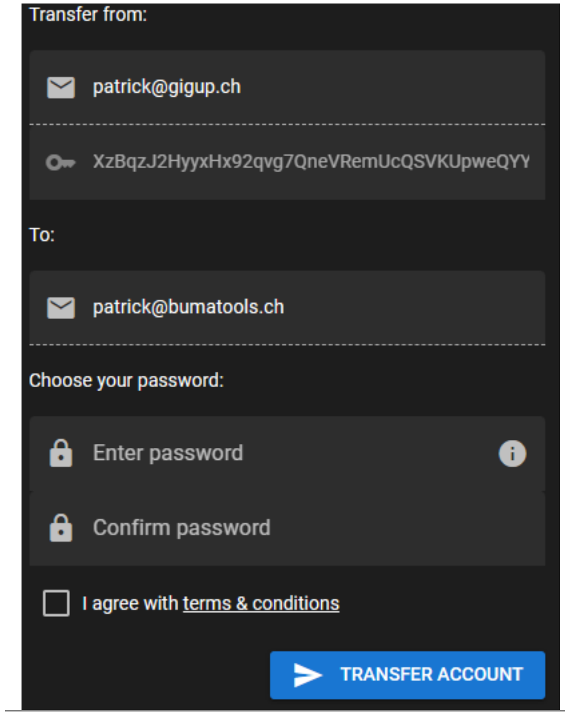
```
axios.post('user-transfer-account',
    {
        'email': String,
        'password': String,
    });
```

```
response()->json([
     'message' => String,
], 200);
```

Make Transfer undone

☒Login: User must login with its old credentials
☒Email_verified_at = null
☒User can verify its account again, see «Email Verification Request»

Transfer Account


Transfer from:

patrick@gigup.ch

XzBqzJ2HyyxHx92qvg7QneVRemUcQSVKUpweQY

To:

patrick@bumatools.ch

Choose your password:

Enter password

Confirm password

☐ I agree with terms & conditions

☒Transfer from (old email, token) to (new email) only read
☒New Password will be set (hashed)
☒User must agree with Terms & Conditions
☒Check if Link is valid and token is correct
☒After successfull Transfer, new user will be logged in automatically.

```
axios.put(signedRoute, {
   'password': String,
   'password_confirmation': String,
   'terms': Bool
 });
```

```
return response()->json([
        'message' => String,
   ], 200);
```

3.4 Admin

3.4.1 Backpanel

Features	Stories	API & Data

Dashboard	☒Only if user is admin, user can See Admin link on left Navbar.
	☒Redirect to Adminbackpanel and get infos. Adapt Left Navbar with Adminfeatures.
	☒Admin can switch to User again by left Navbar. Adapt left Navbar with Userfeatures.
		`axios.get("/admin-backpanel");`
		
		`return response()->json([`
		`      'users' => Int`
		`], 200);`

4 Backlog