Authentication

Server Side

POST - /register

```
- check if user exist (if exist return error)
- hash password (bcrypt)
- create user in the database
- log user in function
```

POST - /login

```
- get user from database with email
- check if user exist (if not exist return error)
- check if password match the password in the database (bcrypt) - no match return
error
- log user in function
```

POST - /logout

```
- remove cookie with JWT (res.cookie('jwt', '', { expires: new Date(0) }))
- return to client auth false (logout successfull)
```

GET - /isAuth

```
- runs first the AUTH MIDDLEWARE then in the request returns:
    status: 'success',
    message: 'User is authenticated',
    user: {
            email: req.user.email,
            id: req.user._id,
            }
```

LOG USER IN FUNCTION

```
- create JWT token with user data, JWT_SECRET and expiresIn time
- set JWT to cookie - res.cookie(name, {httpOnly and maxAge: 1000 * 60 * 60 * 24 * 7
(7days)})
- return (status: 'success', message: 'Login successfull' and user data)
```

MIDDLEWARE AUTH

```
- get JWT token from the cookie (no token - return error)
- verify token (not valid - return error)
- get user from database with decoded id (no user - return error)
- set user to the req.user = user and next() to request
```

PACKAGES USED

```
- bcrypt - cookie-parser - cors - dotenv - express - jsonwebtoken - mongoose
```

JWT_SECRET

```
- JWT_SECRET=kies veilige string
```

TESTING WITH POSTMAN

Import auth.json in postman

```
- register ( POST )
- login ( POST )
- isAuth ( GET )
- logout ( POST )
```