IDG2001 Oblig 1 – report

# Introduction

This is a project replicating twitter. We have created a restAPI that allows users to create, get, edit and delete accounts and tweets and search for users, tweets and hashtags. The project is hosted in 3 different platforms:

Database is hosted on render.

Back-end API: https://zitter.onrender.com/

Front-end: <https://zitter-six.vercel.app/>

Project Github: https://github.com/Eirikfin/zitter

# Technologies

The project was made using the following technologies:

For the database we used a MySQL relational database,” MariaDB”

For the back-end API we used Python with the FastAPI framework for routing and controllers, we chose to use FastAPI over flask because of the better performance and similarity to “express js” in node.

For the front end we chose to use “React” for creating our components.

## Dependencies

Python/fastAPI:

* SQLAlchemy
* Bcrypt
* JWTToken
* Pydantic
* dotenv

Javascript/React:

* React router dom
* sass-embedded

## Deployment

Back-end:

-install python dependencies

-set .env variables : DB\_URL

-run deploy.py to create the database with tables

Launch server: “fastapi dev main.py”

Front end:

Run: npm install

Set fetch queries to the correct url

Run: npm run dev

# Project structure

## Models and Database structures

For our models we chose to split the data needed for the application into 3 different tables: Users, Tweets and Hashtags. Both Users and Hashtags has a 1 to many relations with tweets.

For the User we store the user’s username, id, email address, password and time the account was created. This will be used for identification, displaying username for related tweets, log, inn and potentially password recovery.

For tweets we store id of tweet, user-id as a foreign key for user posting that tweet, the tweeted message and time the tweet was created.

For hashtags we store the “hashtag” value, and the id of tweets where the hashtag gets used.

## Api endpoints and controllers

In our backend API we offer endpoints that support CRUD operations for Users, Tweets and Hashtags. We also support endpoint for searching tweets and offer paginations for the results of these results.

The endpoint can be found and tested at: https://zitter.onrender.com/docs

## Middleware/validation

Password hashing,

Posting users and tweets validation

## Front-End

For the front end we chose to use react for creating components. We have created several pages for the website. Including a dashboard with an overview of recent tweets, a user page that displays a user’s tweet and when they created their account, a search result page that displays data matching the search and a log in page where users log into their account.

# Summary

For this project we made sure to include crud operators for the back-end API. We also made sure that that users could search for tweets, users and hashtags in both the front end and backend parts of the project. We also made sure to hash passwords for security, added some validators for server requests and added a token for authorization.