**PROJECT TWATTER**

**Project Report Document**

Group 4

Ahror Abdulhamidov (id: 014712854)

CMPE 131-02 | Fall 2020

San Jose State University

November 19th, 2020

**Executive Summary**

This document will provide a detailed report on the Project Twatter, a Twitter clone web application. This document provides detailed description of the project, purpose, scope, objectives, software design approach, data design and management, key software components, and the results. The project is part of the requirements for the Software Engineering major in San Jose State University, and was done entirely by a single student.

**Table of Contents**

**Title Page** 1

**Executive Summary** 2

**Table of Contents**  2

**1. Introduction** 4

1.1 Description 4

1.2 Scope 4

1.3 Development Team 4

1.4 Development Process 4

1.5 Reference Material 4

**2. Objectives and Background**  5

2.1 Objectives 5

2.2 Background 6

**3. Software Design** 6

3.1 Architectural Design Approach 6

3.2 Data Design 8

**4. Data Design** 7

**5. Current Results and Analysis** 10

5.1 Results 10

5.2 Software Analysis 13

**6. Conclusion** 13

**6. Appendix** 14

**1. Introduction**

**1.1 Description**

This project imitates the popular Twitter web application. Essentially, it is minimalistic version of Twitter, and it is a web application, built using PHP with Laravel framework. The purpose is to create a web application that allows users to post tweets, view other people’s tweets, like them, and make profiles.

**1.2 Scope**

Twatter is a blog posting social network web application. In general, it is a Twitter clone, a more simplistic and minimalistic version of it. This project will benefit the public, helping them to spread the word across the world in this online platform, which runs on any modern browser from any place in the world. Currently, due to the limitations of the server machine, the application can is limited to handle only the scope of the University’s people, however, with better hardware, it be expanded, which makes this application scalable.

**1.3 Development Team**

This project was made as part of the required academic project by a single-student team in Professor Hungwen Li’s CMPE 131 class, fall 2020 semester, SJSU.

**1.4 Development Process**

The first development steps of the application started on September 18th, 2020, and the project’s state is currently close to its first release, on December 1st, 2020.

**1.5 Reference Material**

The software, described in this document, uses Laravel version 6, which is a PHP based framework, and therefore, some of the features / components / tools / key words are related to Laravel and PHP. For more information, check this website: <https://laravel.com/docs/8.x/>

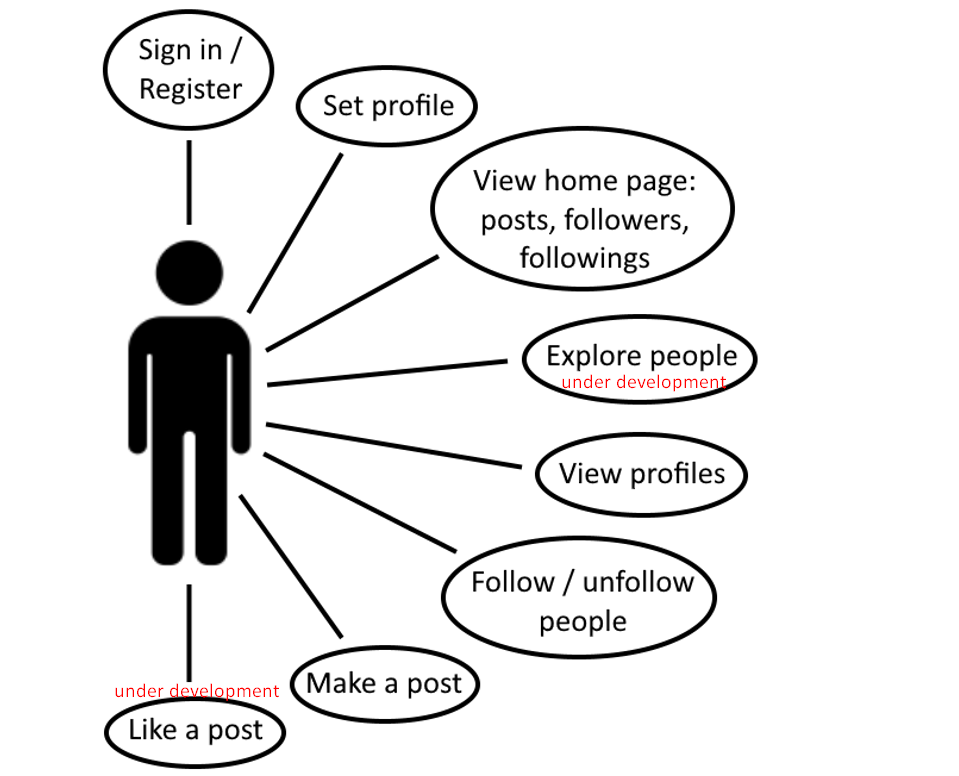
**2. Objectives and Background**

**1.1 Objectives**

Since the project is a web application, it has a user interface (front-end web pages) and a running server (back-end), so it is an MVC-pattern type application, which means, model-view-controller – three essential parts of the software architecture.

The application allows users to register / sign in to the system, post messages or short blogs online that can be viewed by others. In addition, users can follow other users, and therefore, have their posts on their personal feed. In the finished prototype, users will also be able to like other people’s posts and look for other people in the explore page.

Here is a diagram showing what users should eventually be able to do:



**1.2 Background**

The application server side can be run on Windows / Linux / Mac OS computer, while for the users, it will be accessible through any smart device with modern browser and stable internet connection.

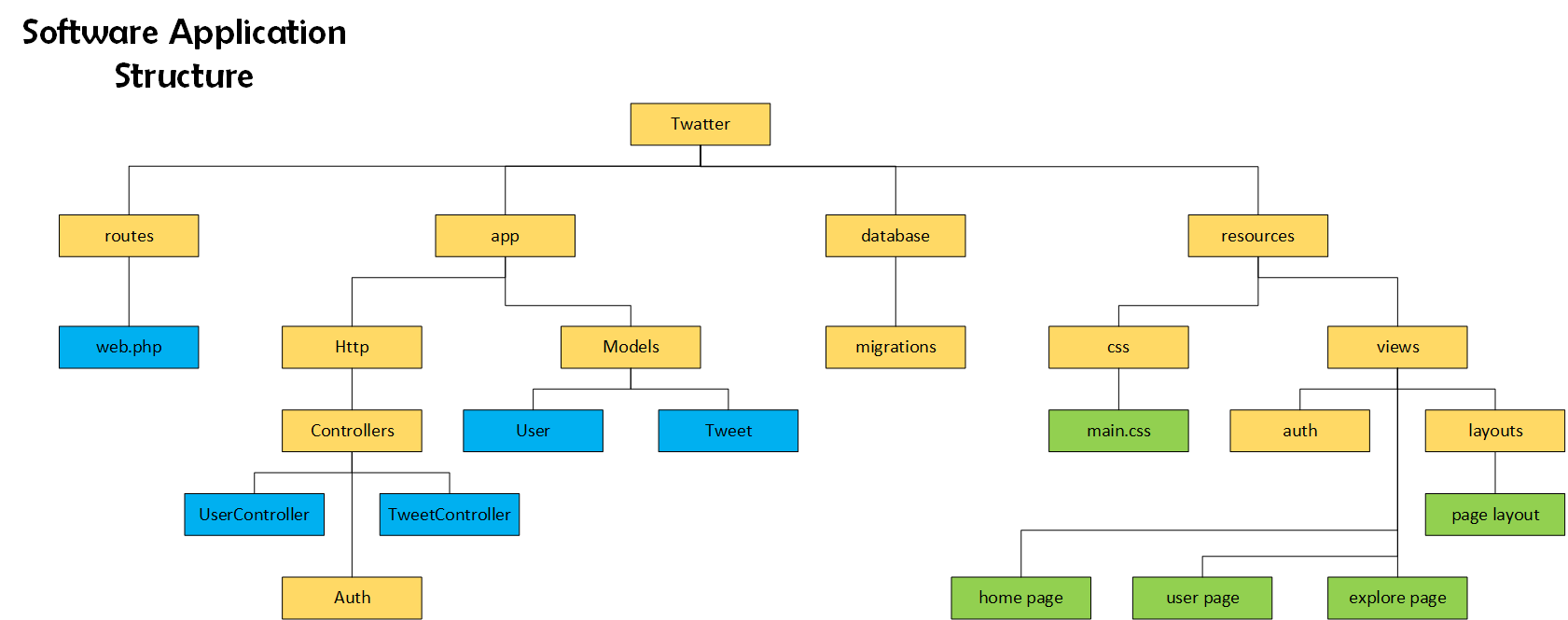
The web application server will run on a Dell Precision workstation laptop connected to a stable internet connection.

**3. Software Design**

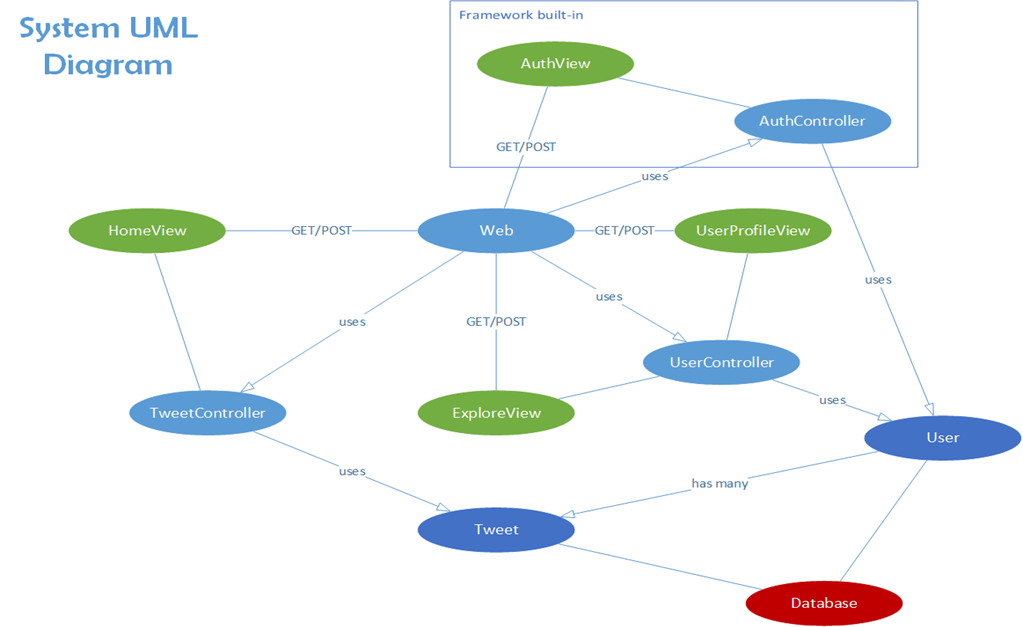
**3.1 Architectural Design Approach**

Architecture of the software is based on Laravel, PHP-based framework for creating MVC web applications. The models in our applications are: User, model that describes users, and Tweet, model that describes tweets (posts). The views (browser web pages) are controlled through controllers that directly access models. Controller access and distribution is done through the routes file (web.php), which handles POST and GET requests. Models are accessed through and saved in database tables, which are created through migrations files using php artisan, a command tool provided by the Laravel framework. In addition, the authentication API will also be provided and integrated through Laravel during the creation of the application.

Here is a brief structure of the software file/package structure:



Here is the top-level system design:



The core object is the Web file in the “routes” package, and it is in the middle of the diagram. The light blue objects describe controller classes, green ones – web pages, the dark blue – model classes, and red one – database.

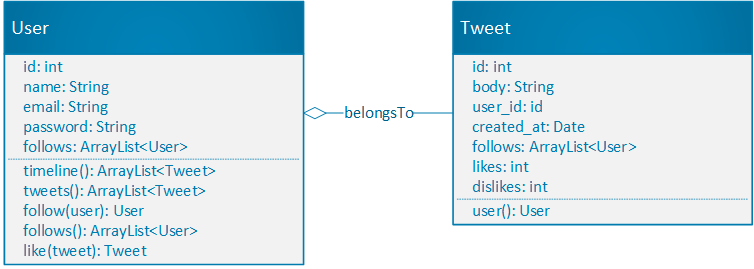
**3.2 Data Design**

As it was said previously, Twatter is an MVC pattern web application, and therefore, the core of the application would be its models, User and Tweet.

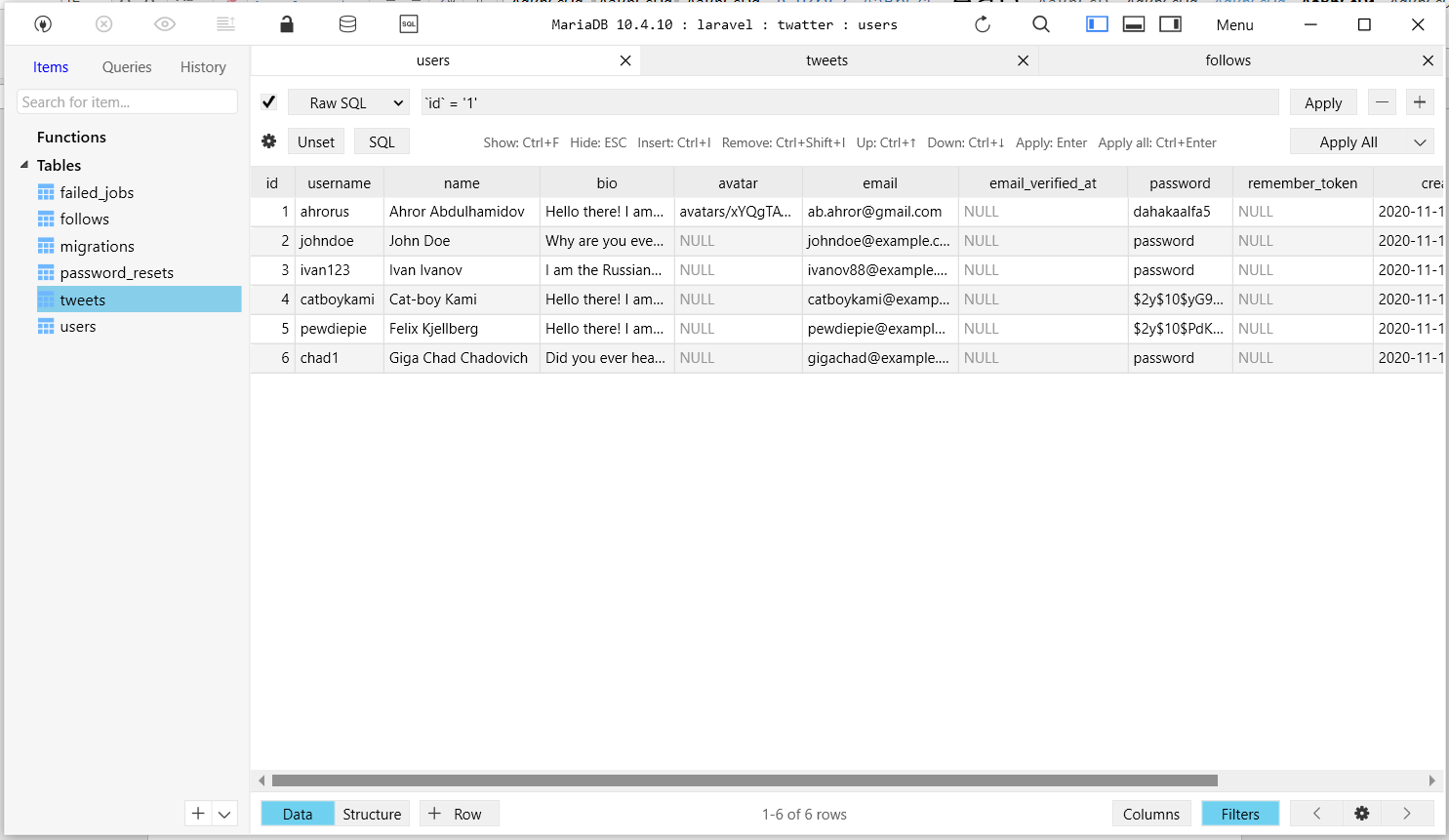
Model User describes a user who created his account through authentication. It contains id of the user, name, email, and password. In addition, user have his own many tweets and followings (other users whom the user follows). User can follow another user, return his tweets, return his timeline (tweets of the user and of his followings), and return his followings (users whom he follows).

Model Tweet describes a tweet (post) that was created by a user. It contains the id of the tweet, its title, text body, date of creation, and id of the user to whom it belongs. User and Tweet are connected with a “has many” relationship. Tweet can return the user it belongs to.

Here is a diagram, describing models:



Models implement Laravel’s Eloquent ORM, which is a simple and beautiful way of implementing ActiveRecord for working with databases. Both models have database tables (users\_table and tweets\_table), which are created and changed through migration files, which is an easier way of initial database management for major changes, provided by the framework. The type of database for this project will be MySQL. The software that is used to manually manage database is TablePlus application (screenshot below):

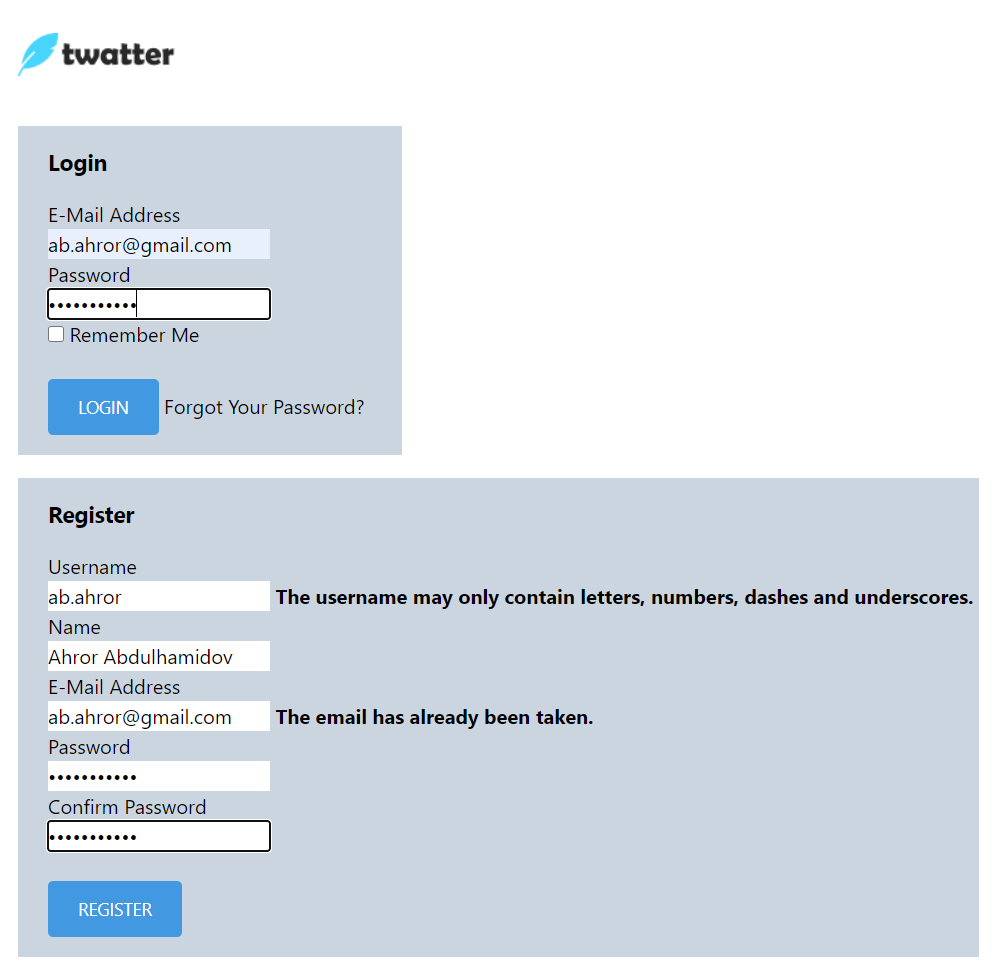


**4. Current Results and Analysis**

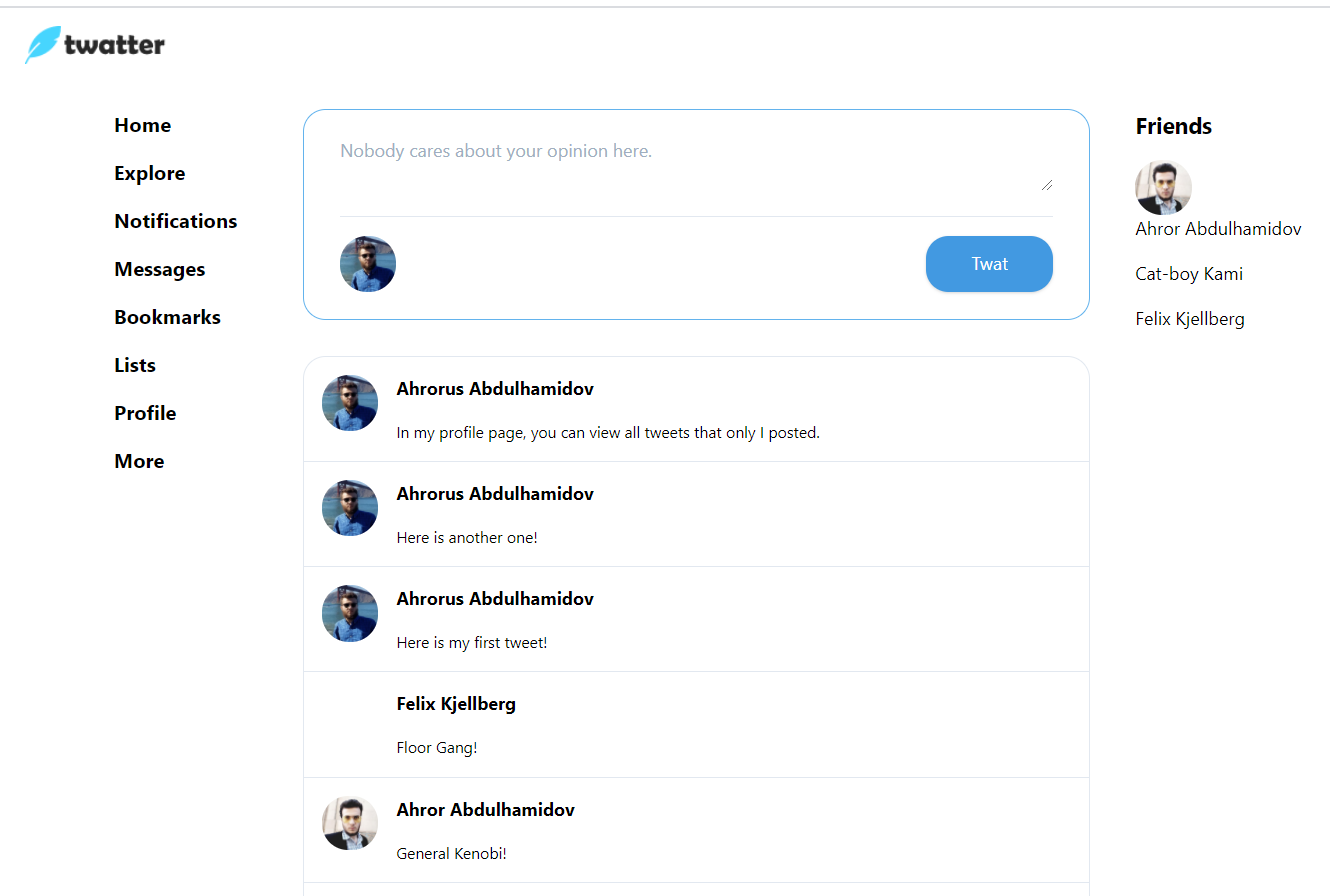
**4.1 Results**

So far, the application is practically complete: users can create/edit profiles and sign in to the system, including setting personal avatar picture, they can follow other users, post and view tweets. By the release date, which is in the beginning of December 2020, users will also be able to like posts and search for other users in the explore page. Here are screenshots of the working web application:

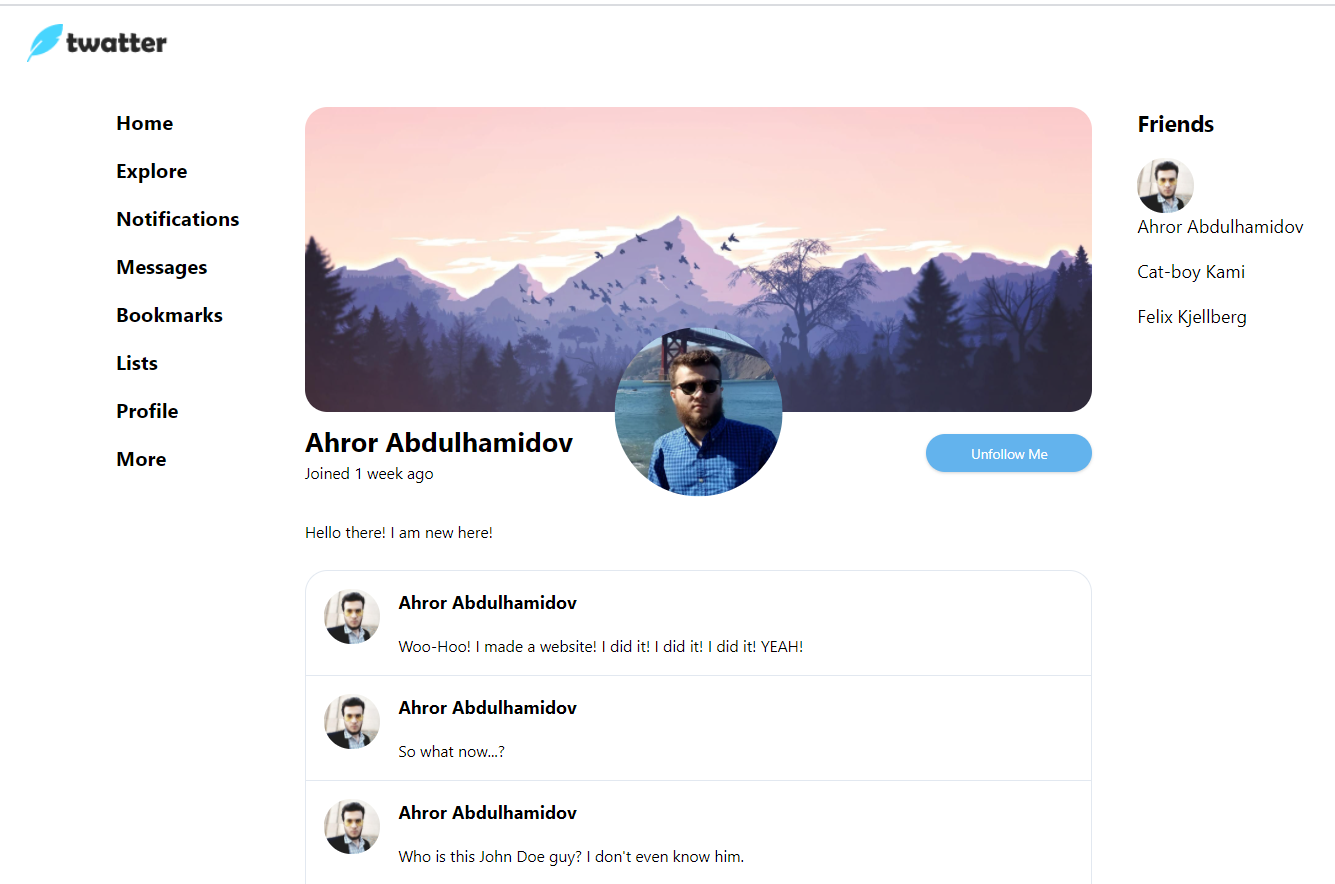
1) Authentication (login and register):



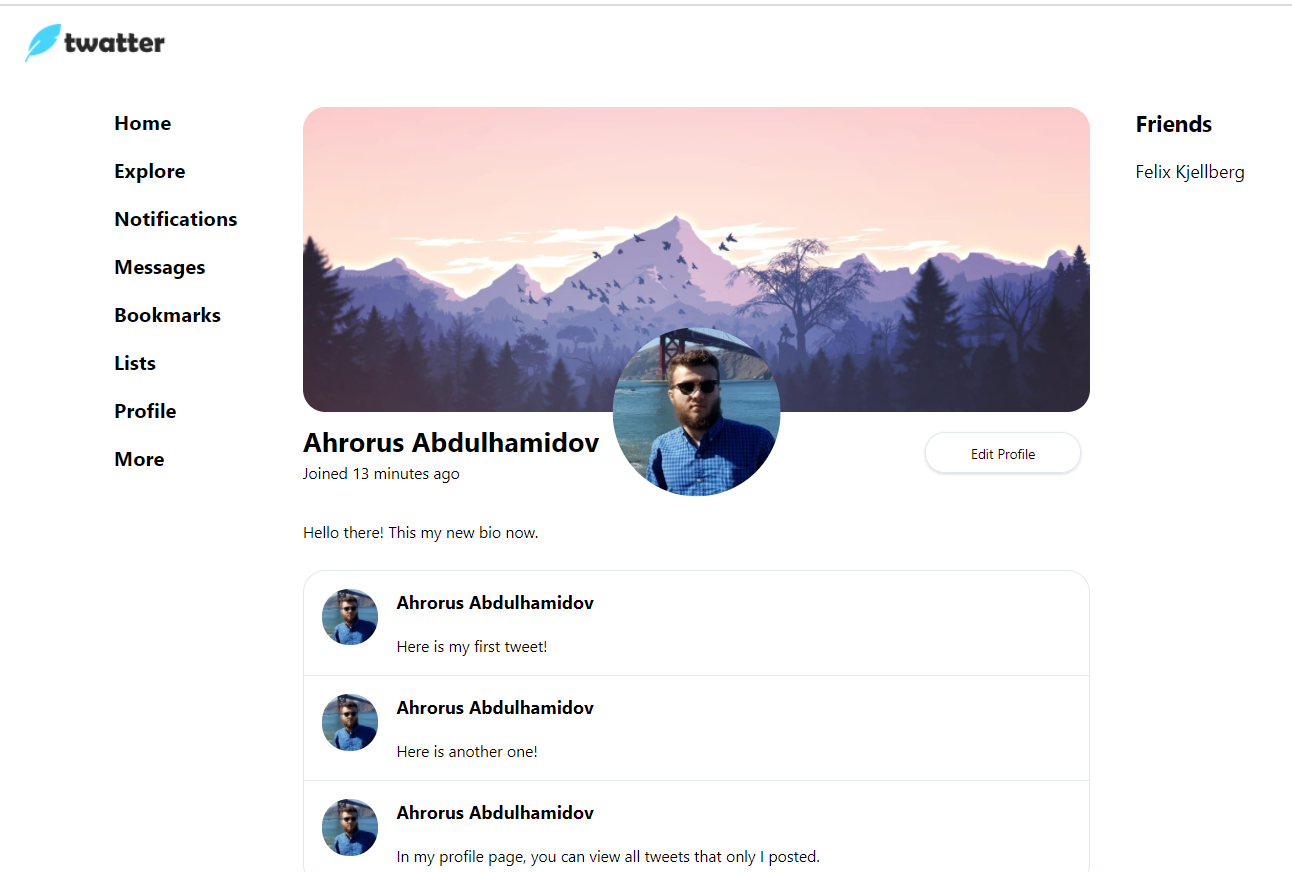
2) View and post tweets:

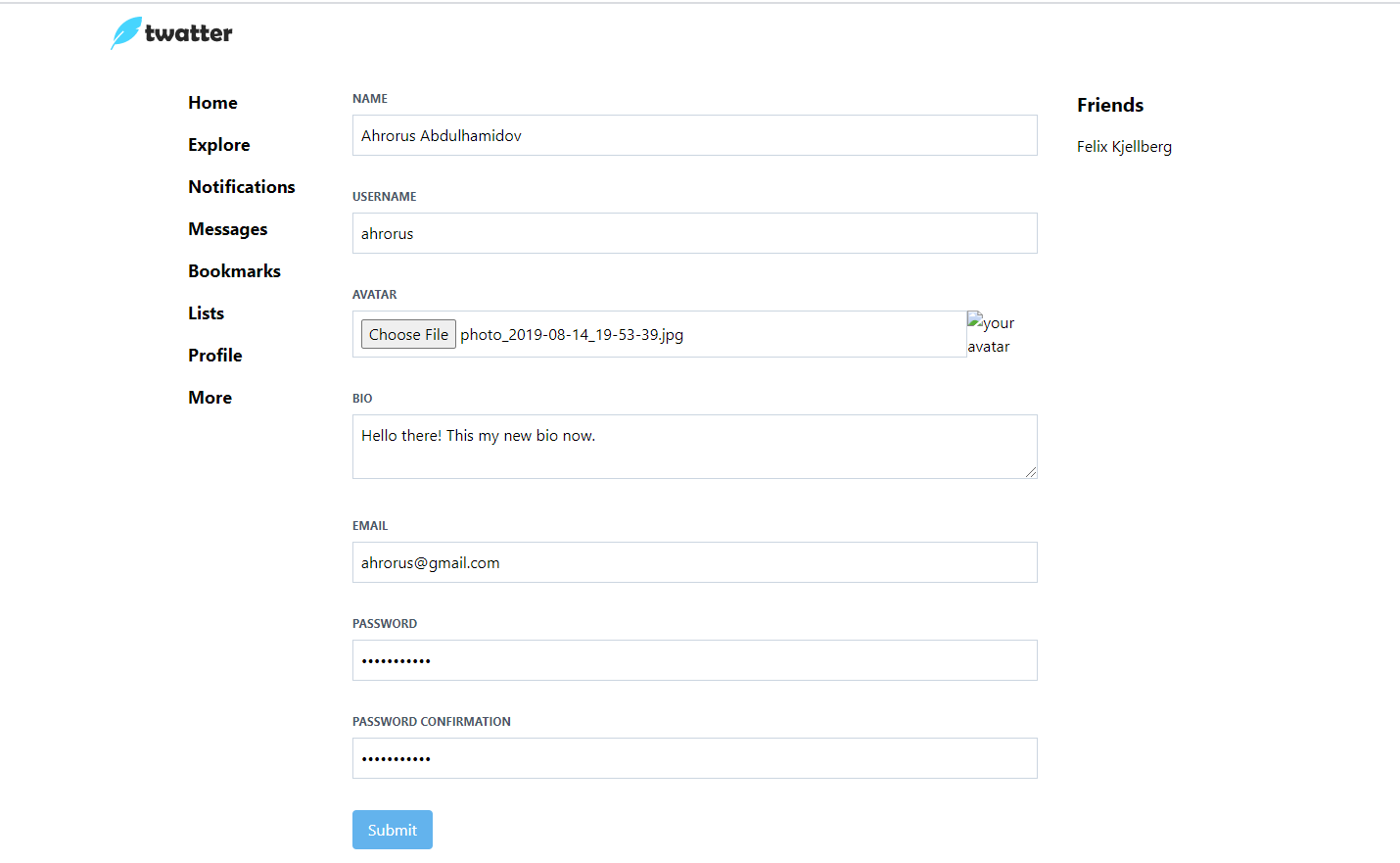


3) View profiles and follow:



4) Edit personal profile:





**4.2 Software Analysis**

For the backend, MVC pattern was chosen as the core of the software design. Model-View-Controller software architecture pattern is the best possible pattern for this type of applications, where user interacts with interface, by viewing, creating, or updating different objects. Essentially, it will be our backend software, specifically, models and controllers that will manage the website.

In addition, the architecture is standard for the projects built using Laravel framework. While Laravel is simpler to use PHP framework compared to others, it is also very suitable for creating PHP MVC web applications that are of medium or smaller scale; large scale social network applications, like Facebook, should not be written in Laravel. Overall, it is not the most efficient or secure, but it is also not worst, and it is not hard to integrate, which makes it the best fit for this project.

The front-end is built using Html/CSS and PHP with Laravel’s Blade Markup integration. While Html and CSS is standard for building web pages, PHP is standard for making pages dynamic. The web pages are also protected from SQL injections and errors by using CSRF token protection and error checking when filling forms. In case of errors, like making unauthorized actions, will through a corresponding error page (in this case, it will be the error 403).

**5. Conclusion**

To wrap up, the project Twatter is a web application that imitates Twitter. It is relatively more complex than other projects since to build it a new framework needs to be learned and implemented, which is Laravel, popular PHP framework. It is essentially an MVC-pattern application with a great practice of back-end concepts: OOP, data modeling, database integration. All of this makes the project an outstanding software engineering project, perfect for the Software Engineering course. The learning outcomes of this back-end applications are imperative for the Software Engineer career, and, hopefully, this project will benefit the developing team in their learning path.

**6. Appendix**

This section is supposed to provide the source code for this project, however, since this is a fullstack web application for a social network, which uses an existing framework and its certain features that auto-generate certain codes, the necessary source code for the entire is too enormous for this document. Therefore, a Github repository link that has the source code is be provided instead:

<https://github.com/Ahrorus/Twatter>