**General Information**

I generated Django models using the "inspectdb" feature and made slight modifications before creating and applying migrations. Subsequently, I made additional alterations to the SQL tables by adding primary key columns ("id") to the "filterwords" and "notifications" tables to ensure compatibility with Django models and the admin panel.

Afterwards, I focused on creating web forms for these models within my website. I integrated CSRF tokens for enhanced security and incorporated the Django Crispy Forms library to improve the forms' styling. Furthermore, I transformed integer fields for "filter words ID" and "sites ID" into dropdown fields, using Django's "ModelChoiceField."

Regarding the relationship between "client" and "filter words," although it wasn't explicitly defined in the database, I devised a workaround within my Django application. Specifically, whenever a new client was created and a filter word was chosen, I added a new entry to the "filterwords" table with a different "clientID." This effectively resulted in a duplicate entry with a distinct client association.

I successfully implemented Create, Read, Update, and Delete (CRUD) functionalities for the "Clients," "Notifications," and "Filter Words" tables. For the "articles" table, I focused on Read and Search operations. Additionally, I leveraged the "sites" table for display and creation operations on my website.

1. To effectively run the application several things need to be modified in the DB, specifically command below should be executed and db must be modified.:

ALTER TABLE filterwords ADD COLUMN id INT AUTO\_INCREMENT PRIMARY KEY;

ALTER TABLE notifications ADD COLUMN id INT AUTO\_INCREMENT PRIMARY KEY;

1. In settings.py in DATABASES user and password must be modified according to your DB settings.
2. All necessary packages need to be installed.

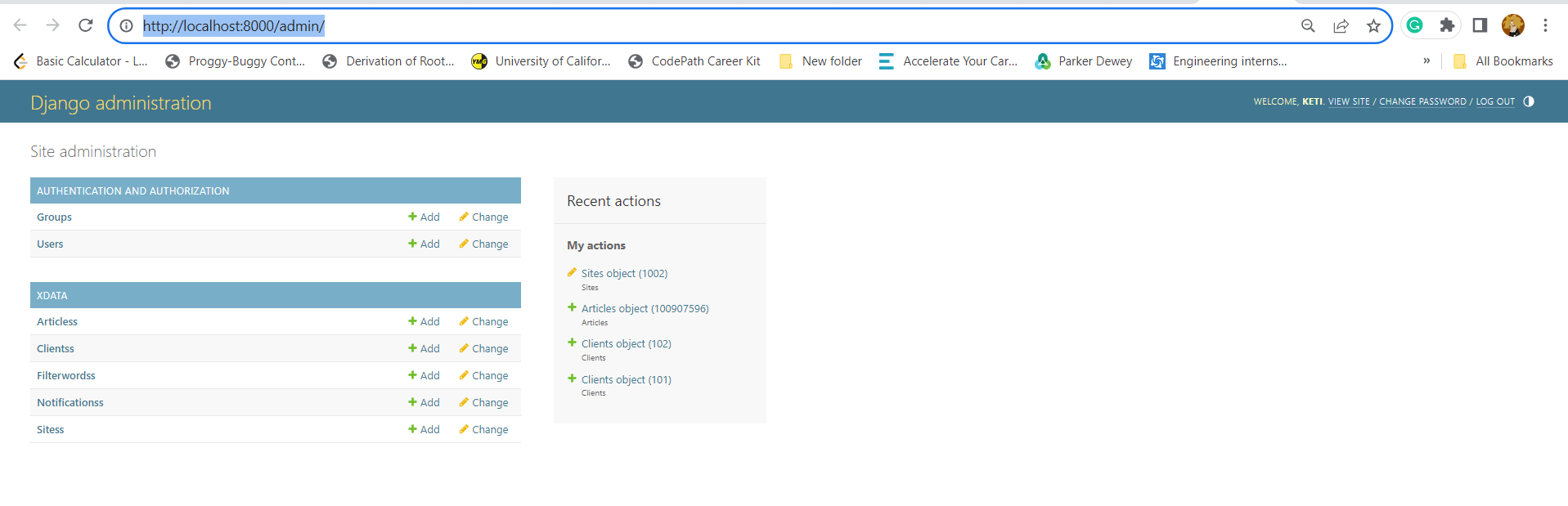
\*\* In order to run superuser you can go to http://localhost:8000/admin/ and use the following credentials:

**Username**: keti

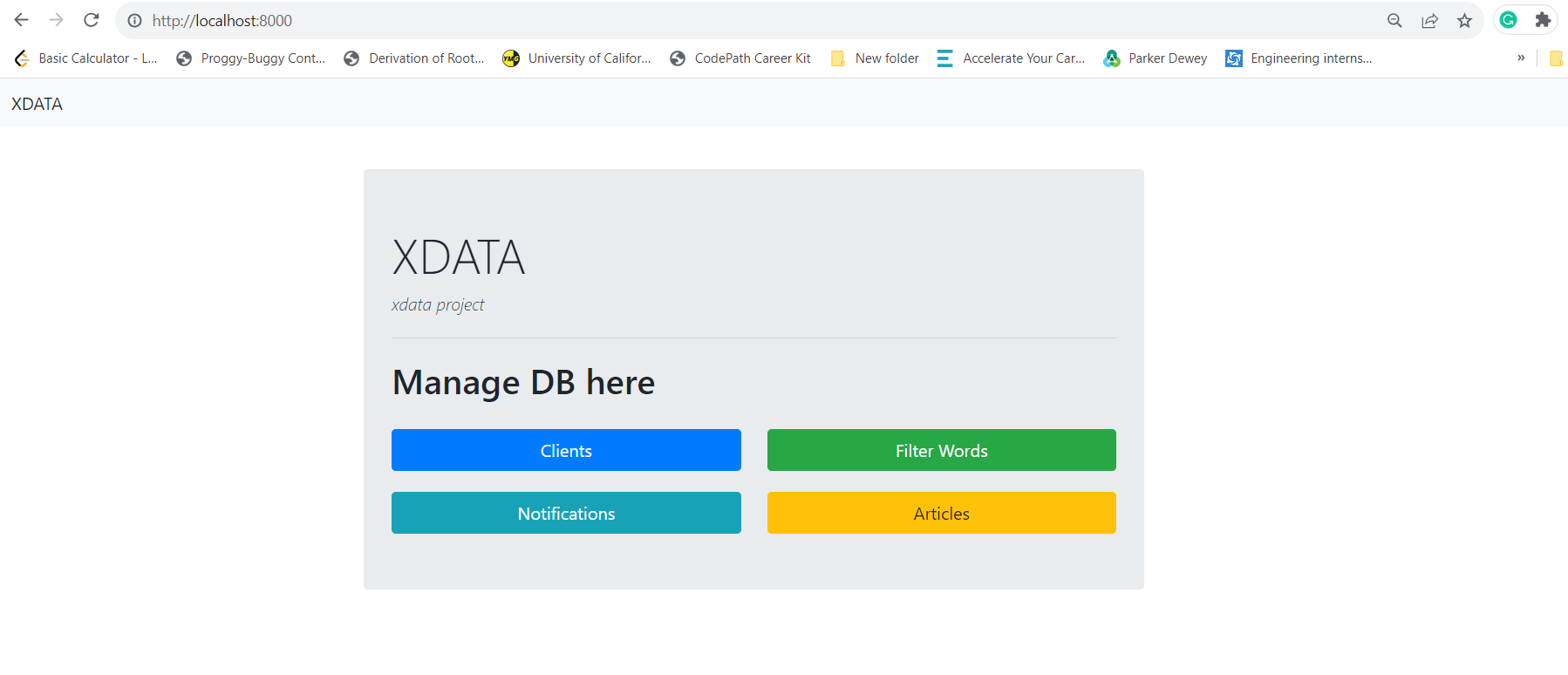
**Password**: 1234

**Visual Overview**

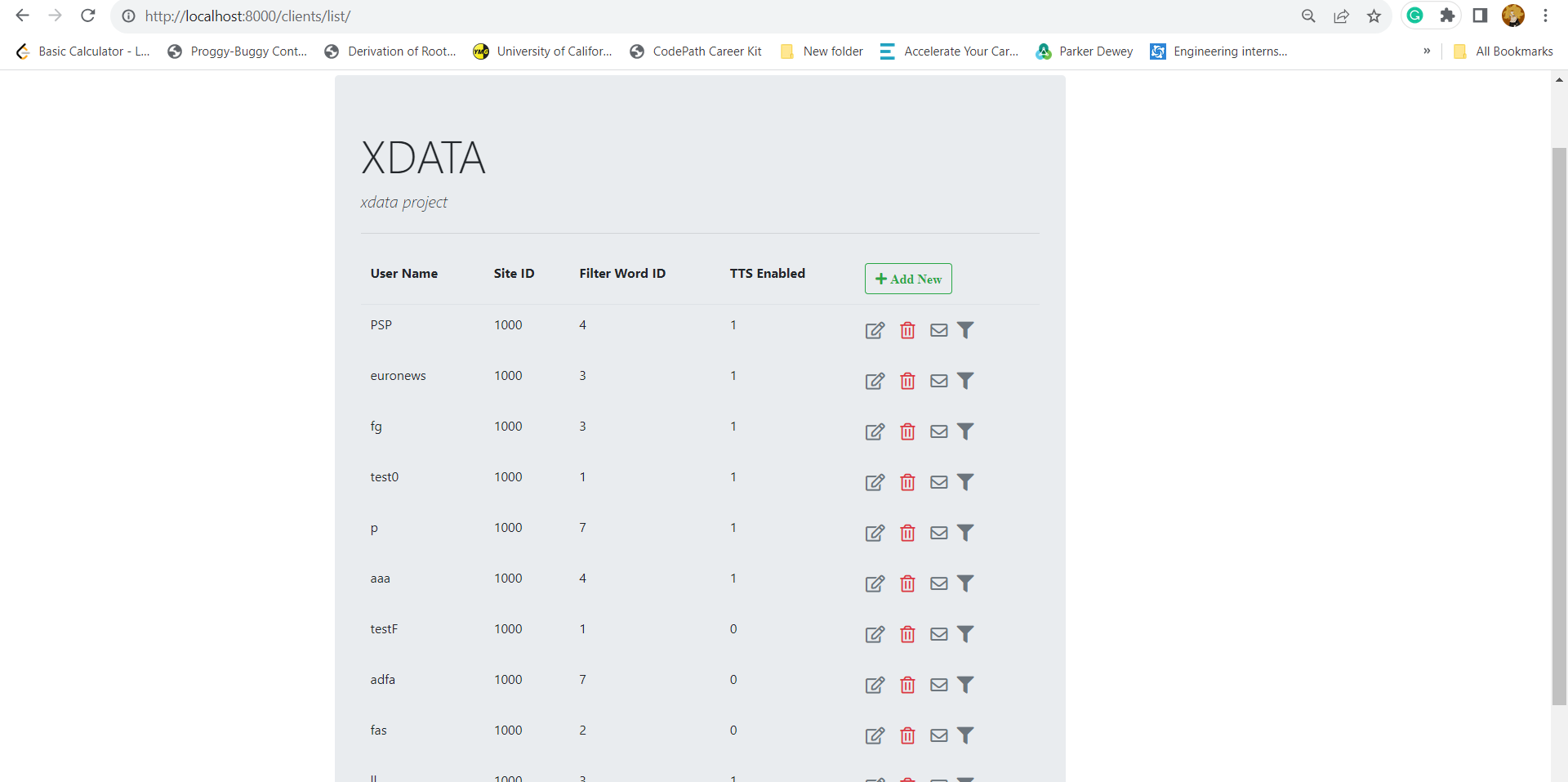
Superuser can control all tables:



Server runs at port 8000 and from here anyone can manage DB, all features there are implemented by me. In the next steps, I will discuss each section.

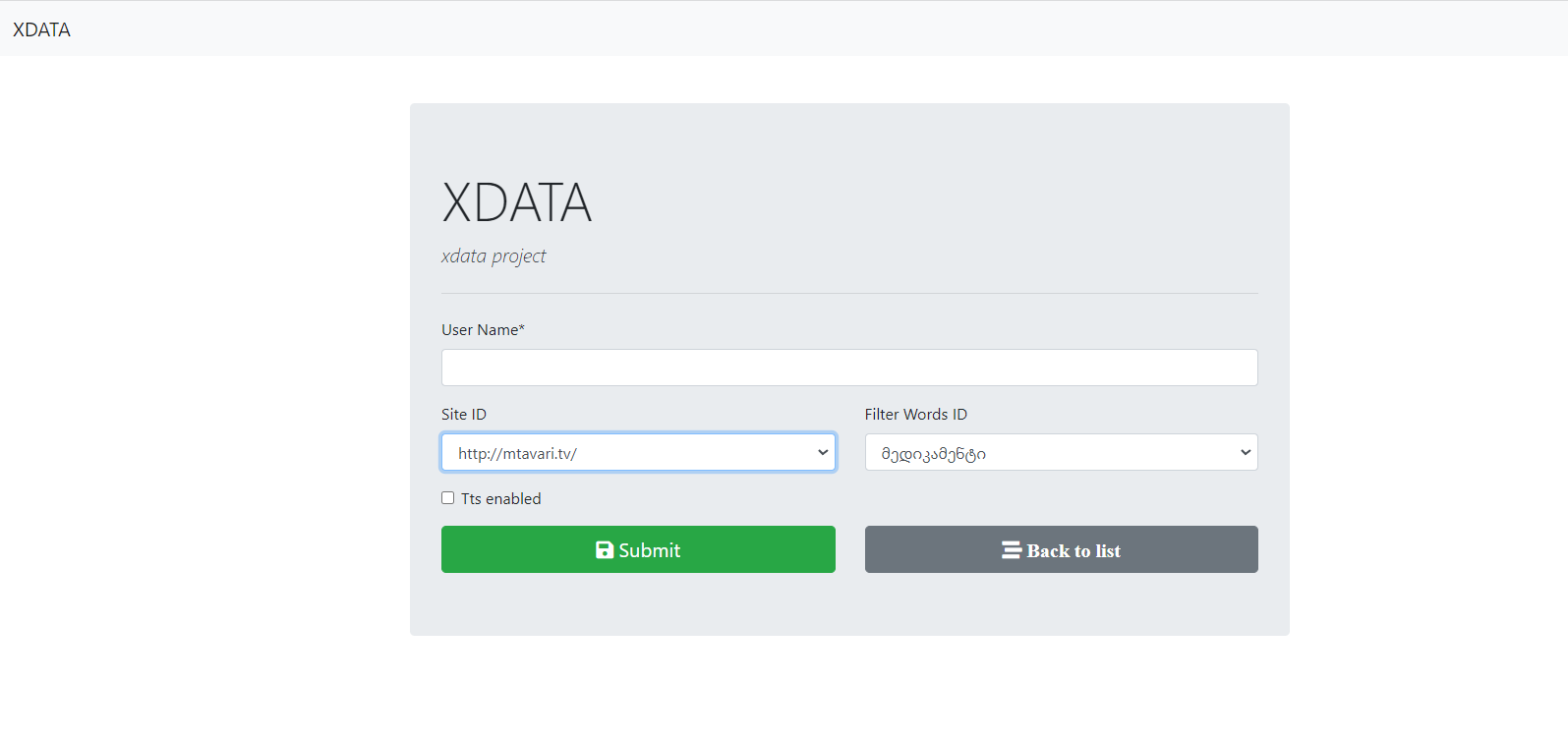


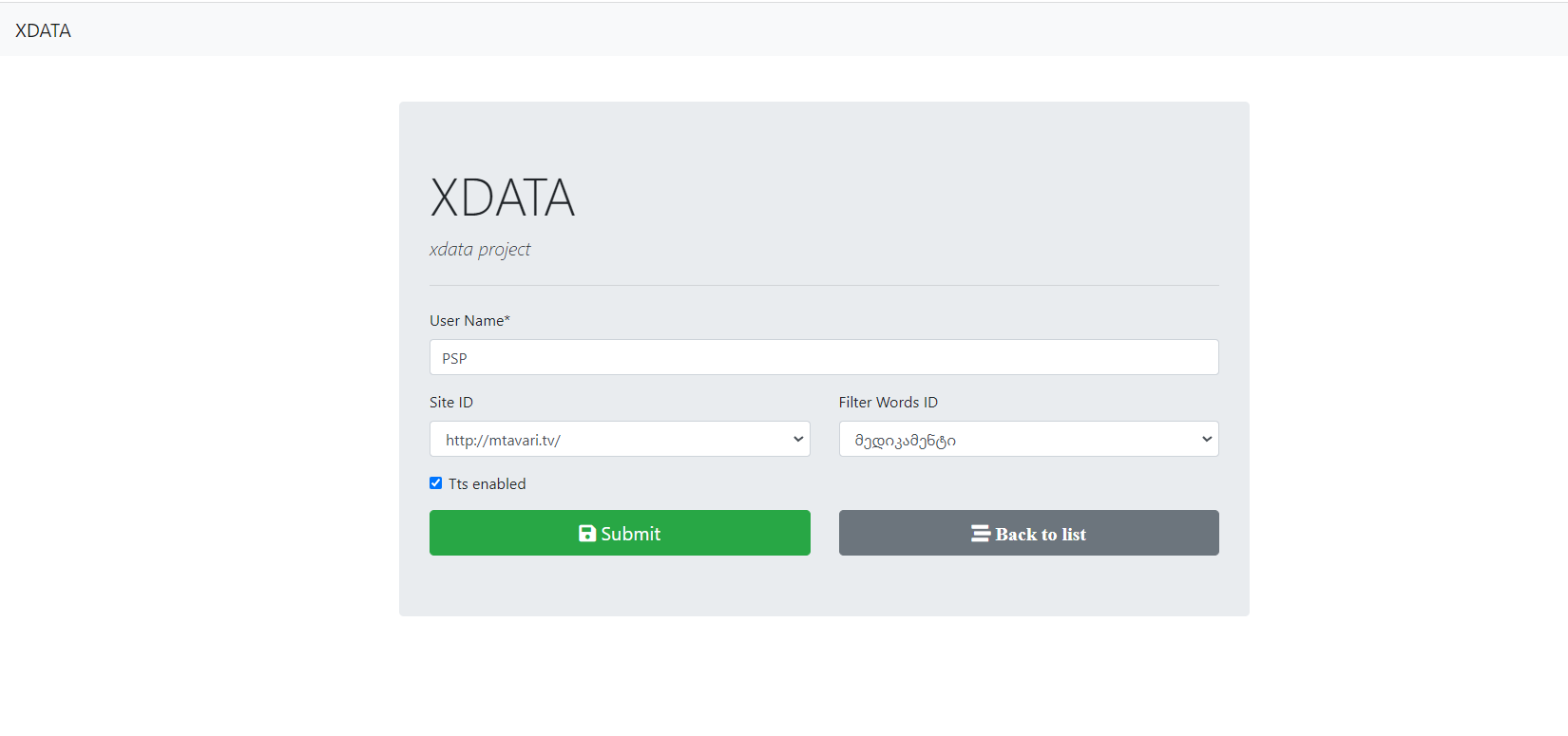
**Clients section:**



From here, one can add, delete, modify and view all clients.

Add and update the following form:



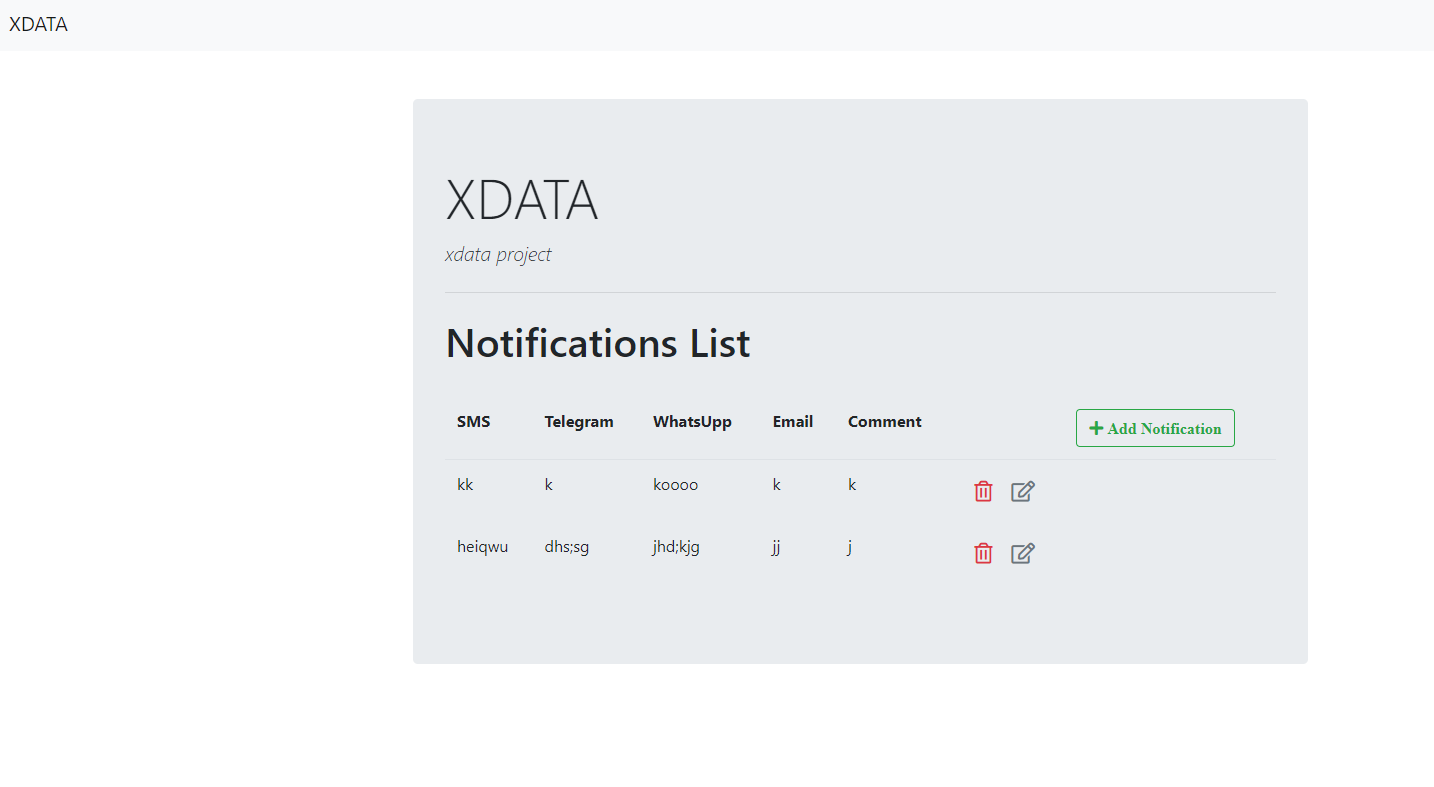


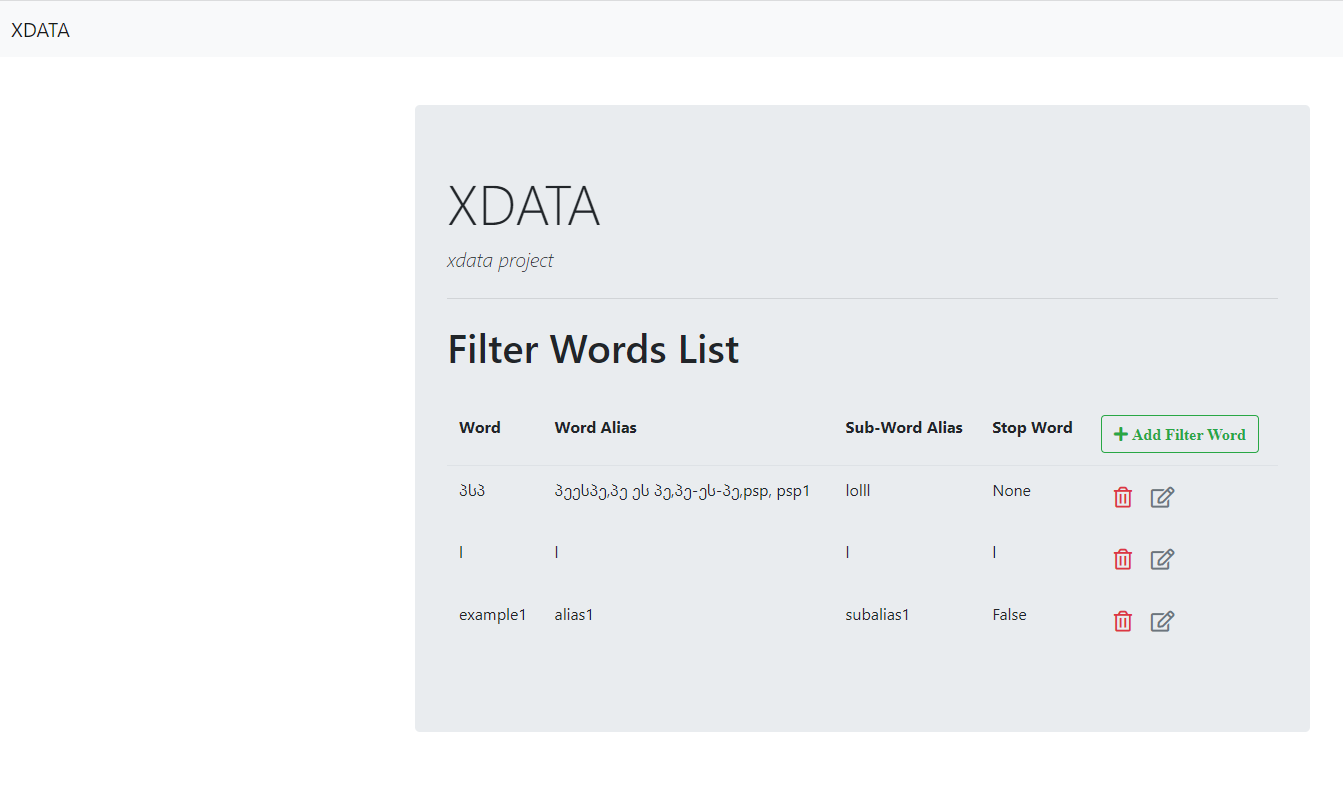
They look almost similar except modified form fields are populated from the DB.

Site ID and Filter Word ID are presented as dropdown, even though one is choosing the name of the filter word or website, behind-the-scene ID is used.

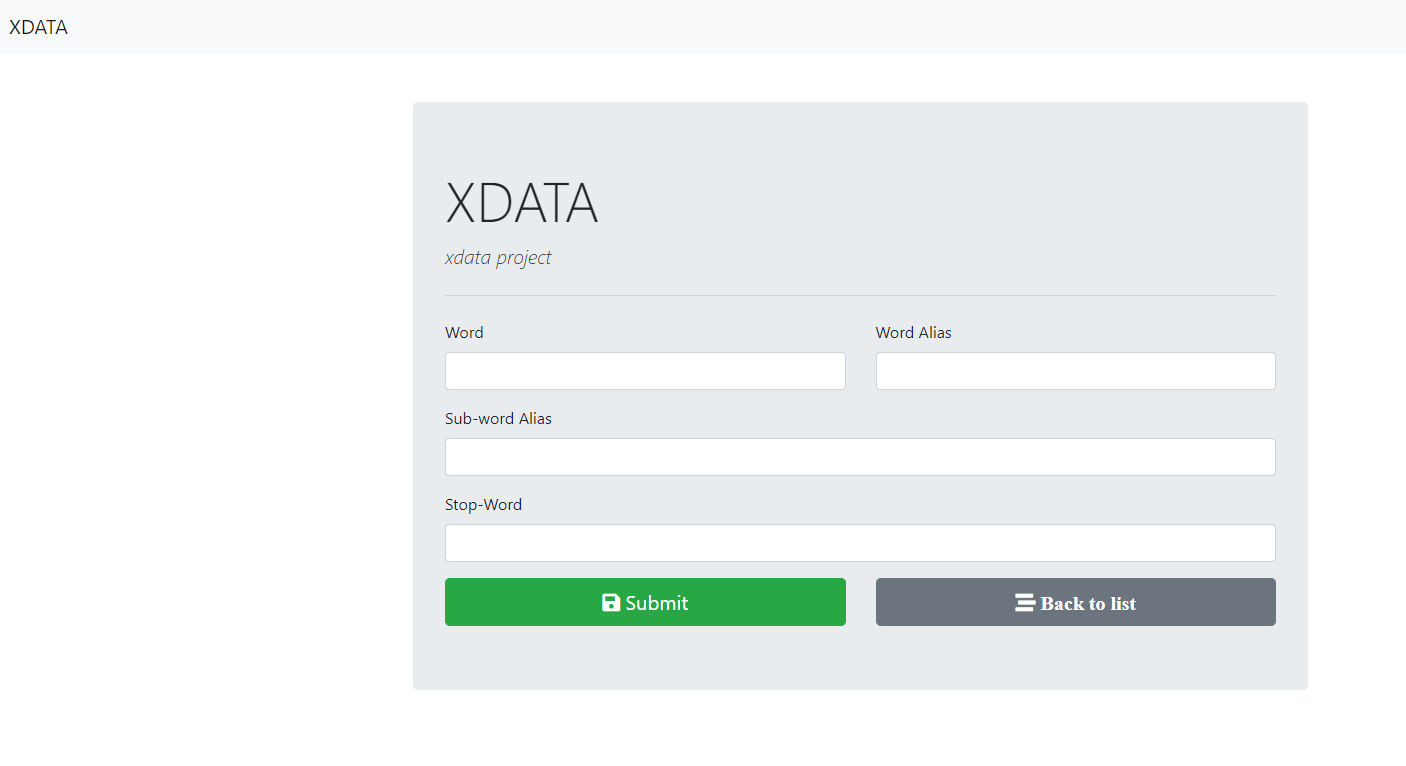
Also, even though Tts enabled is an integer field, here I modified it (using forms) to look as a toggle.

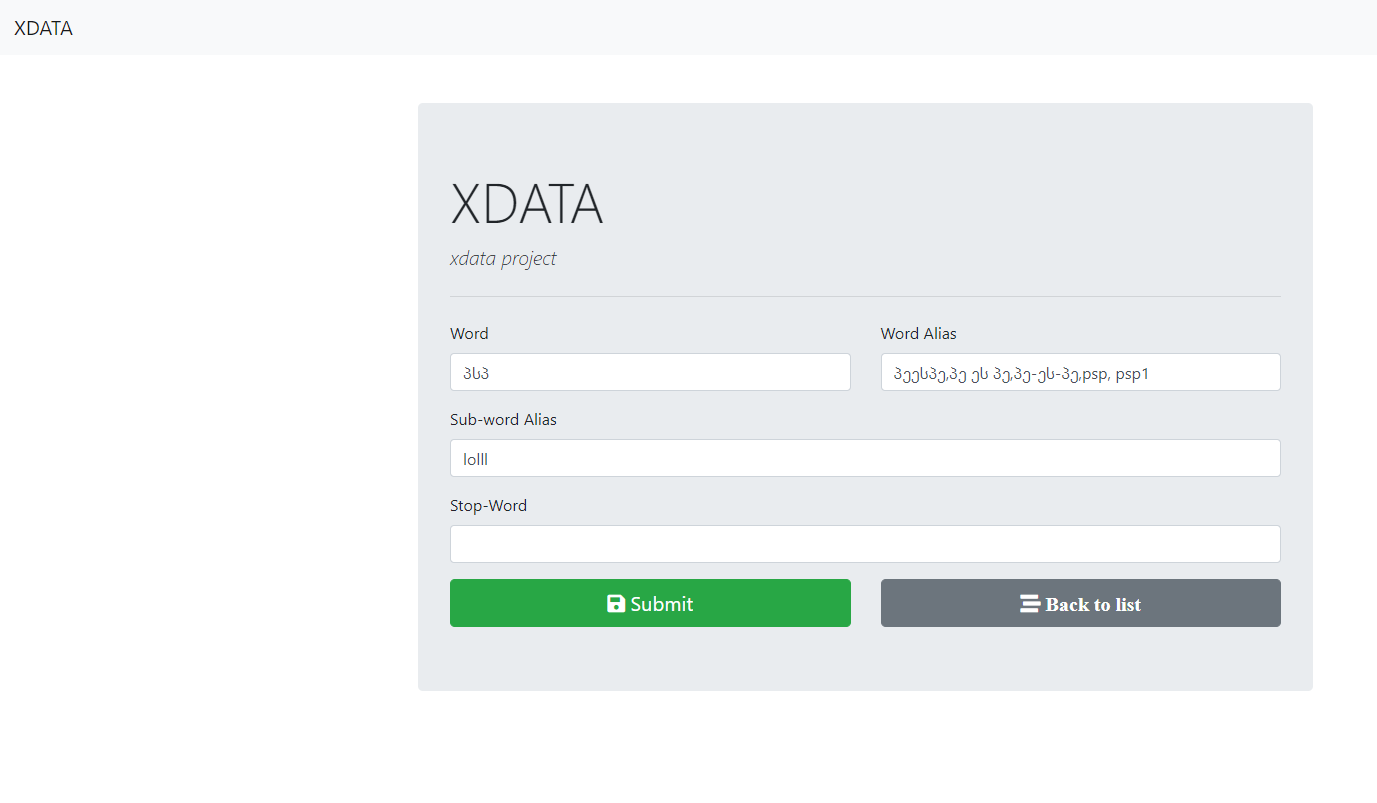
In the clients section, there are also notifications and filter words sections, which display information specific to the user based on their ID.

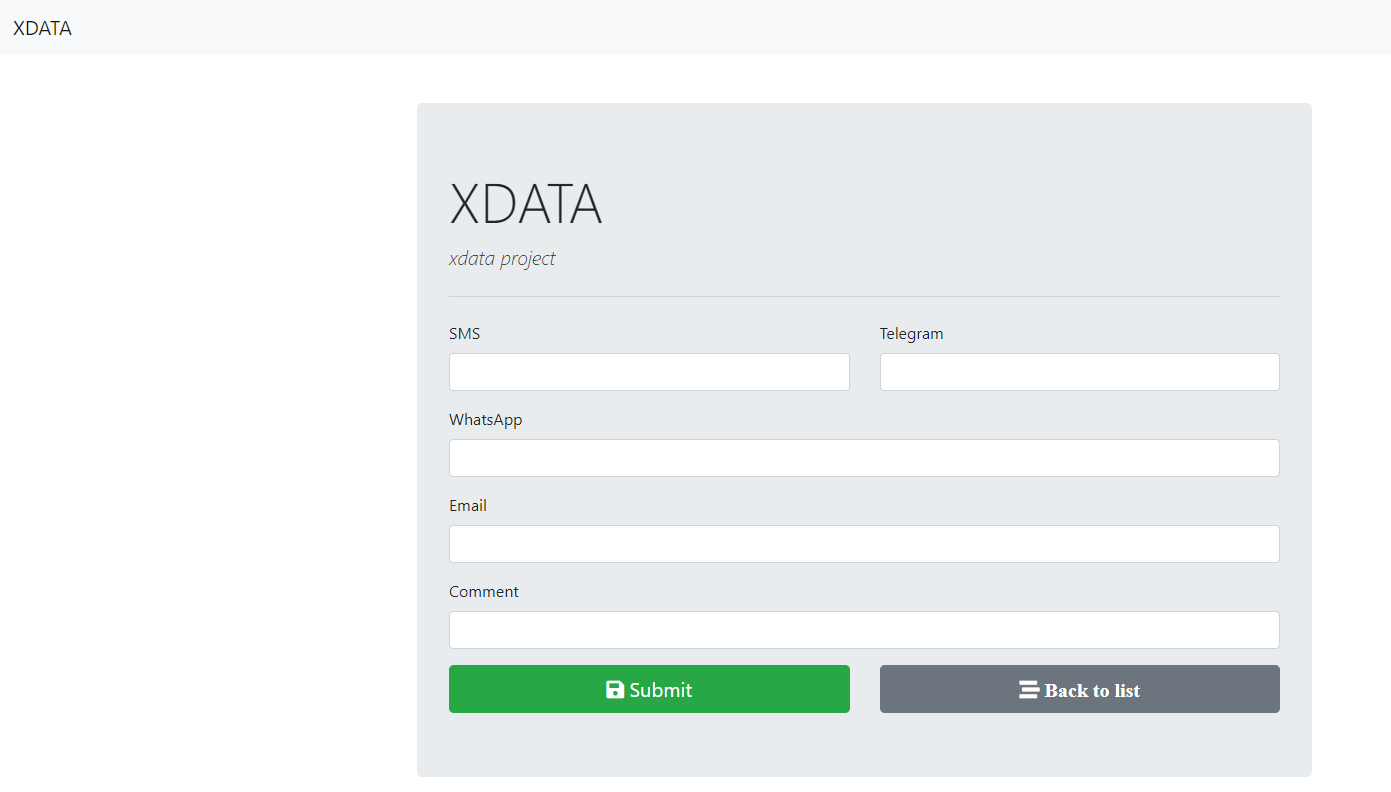


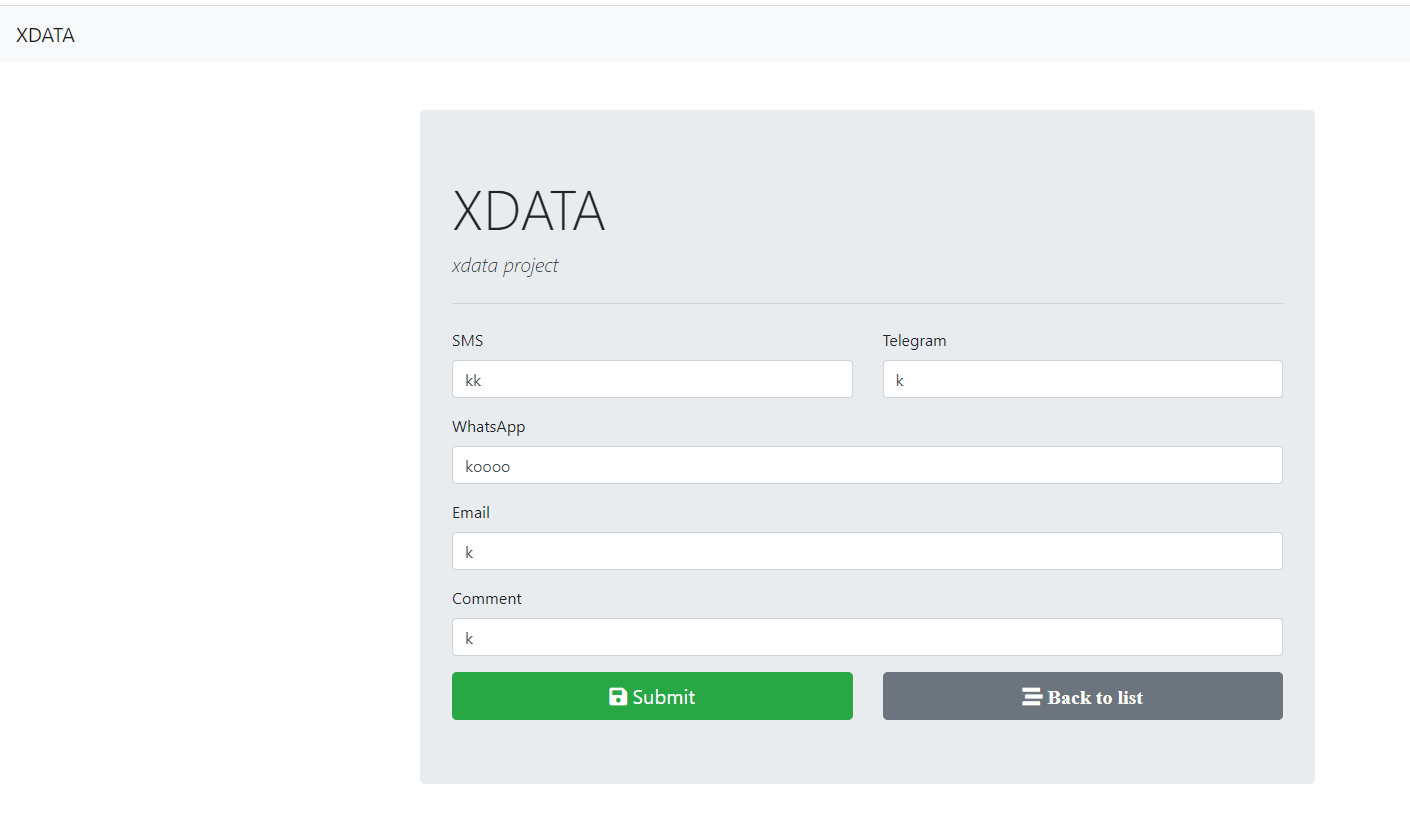


From there one can add, delete, modify and view filter words and notifications specific to the user.



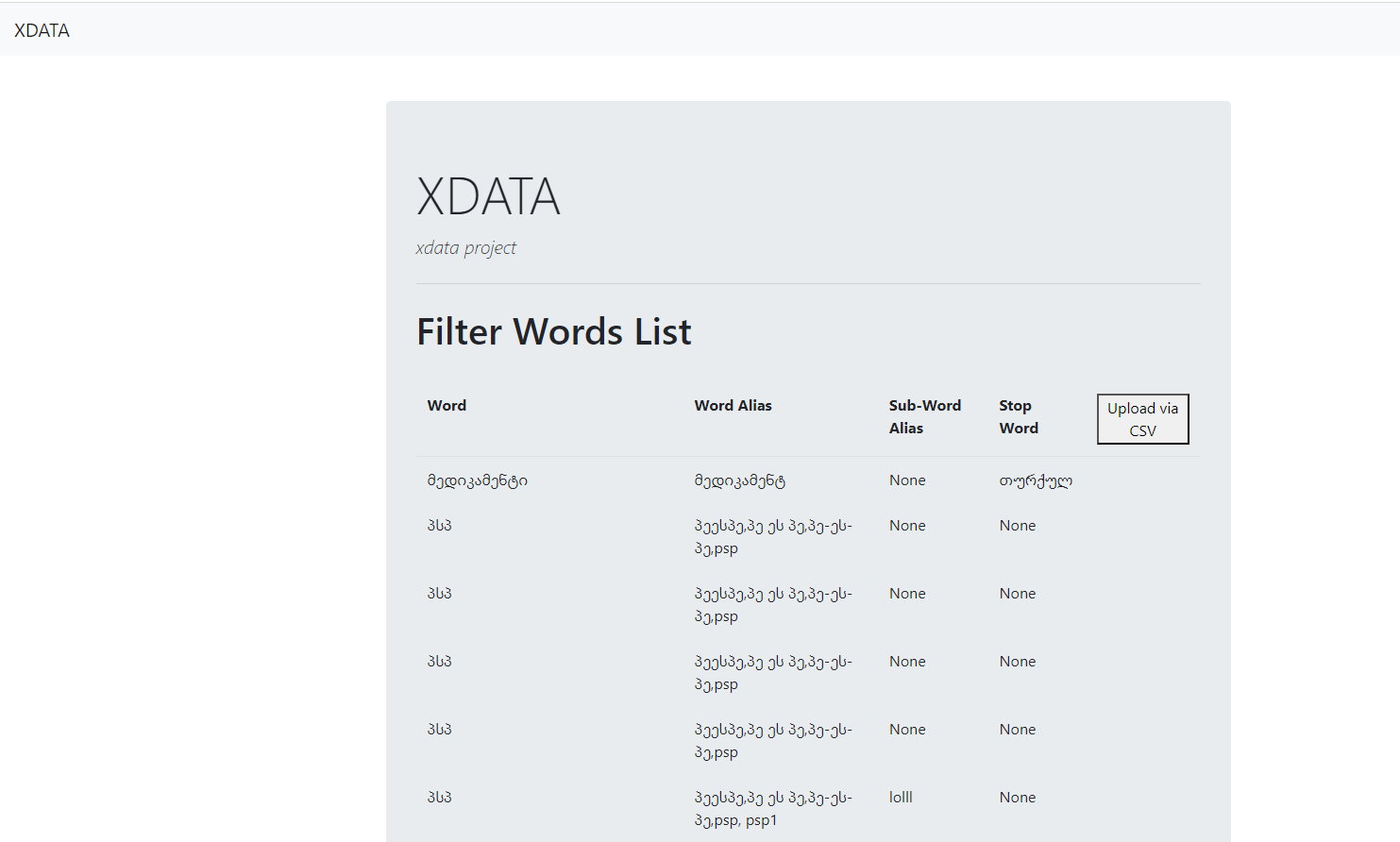




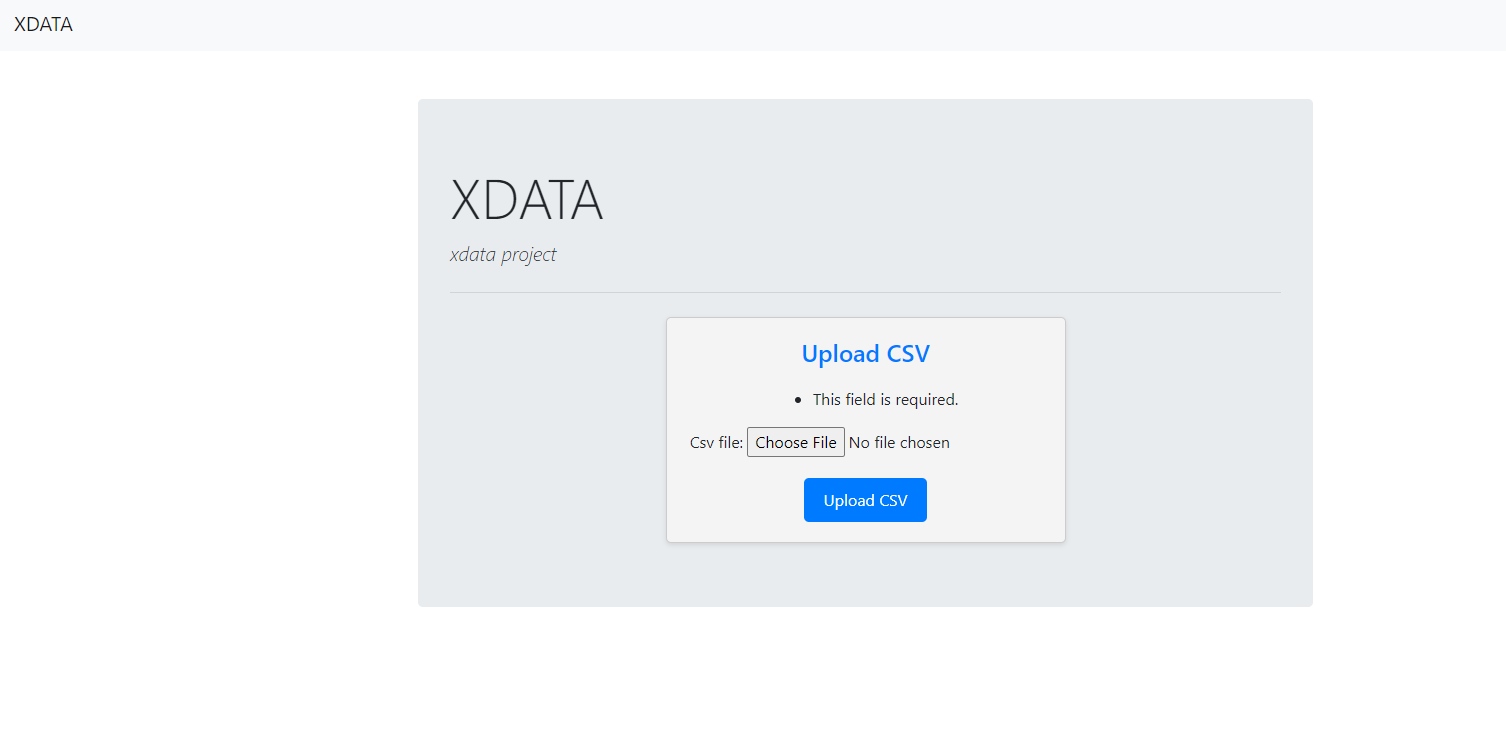


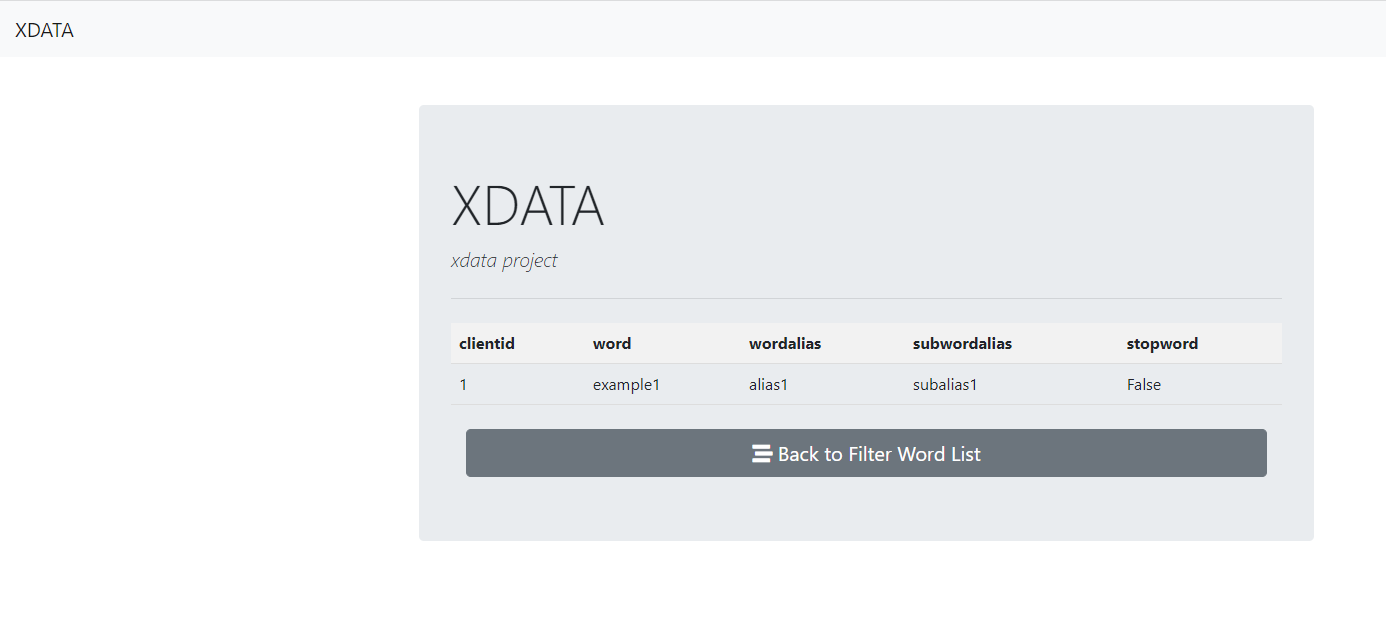
**Filter Words Section:**

This section has two purposes - first display all the filter words for all the users, just as a list and second add new filter words using a CSV file.



One can upload a CSV file:



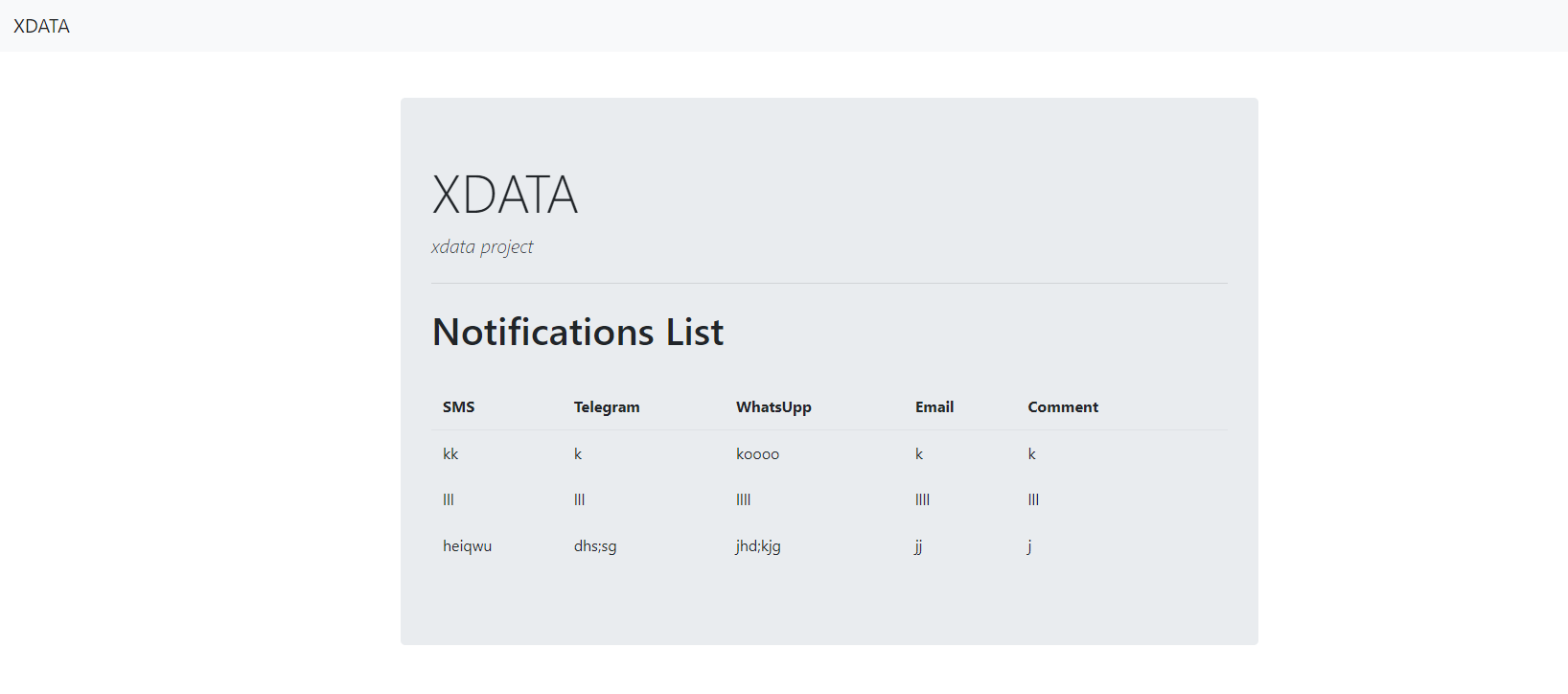
View what is being added:  


And go back to the list to check a new item.

\*\*The project contains a sample.csv file to test this feature.

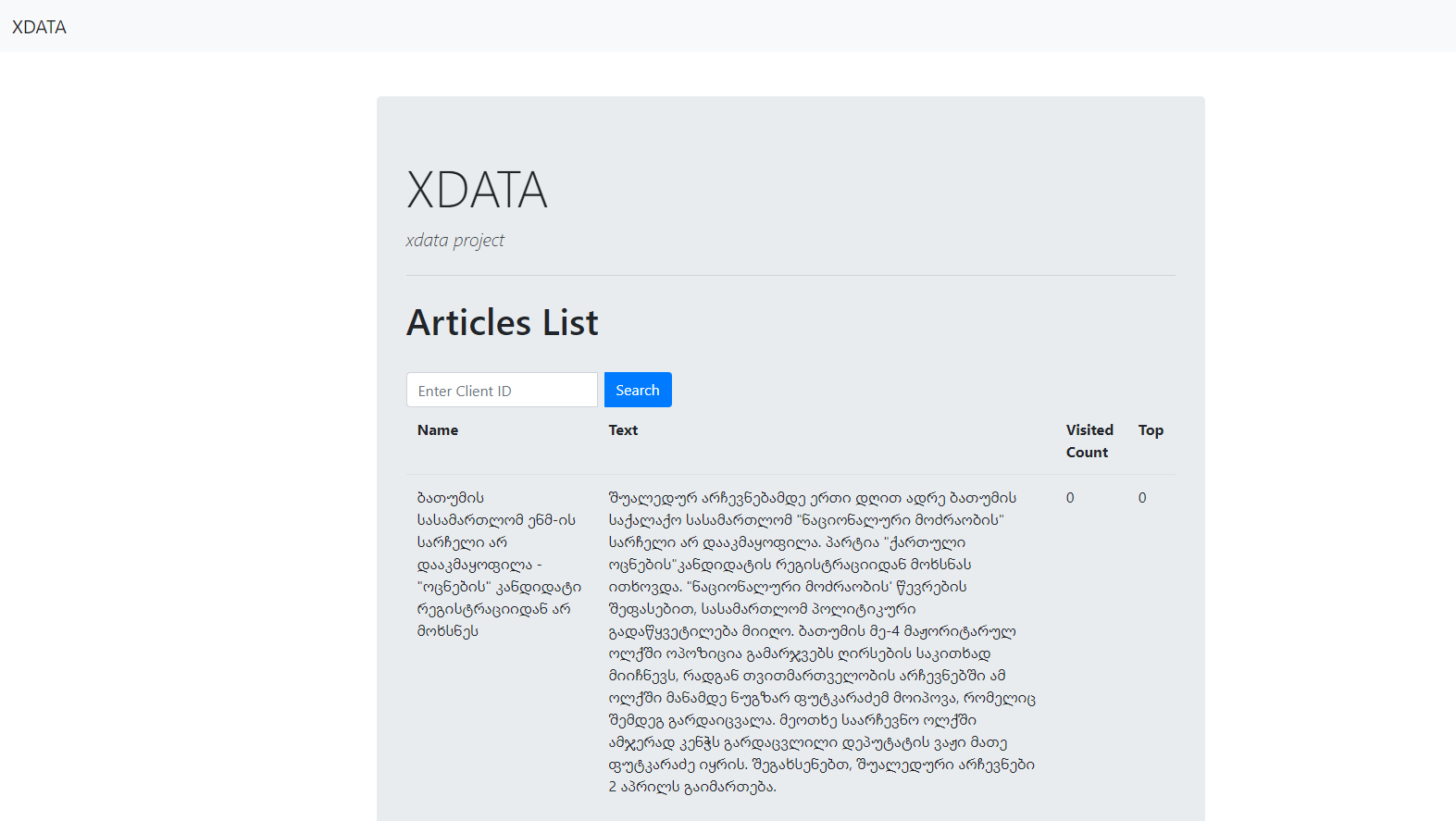
**Notifications Section:**

This section simply lists all the notifications for all the users.



**Articles Section:**

This section also serves two purposes, first see all the articles for all the clients and second, search articles based on clients ID.



One should simply enter the client ID in the search field:

