LinkedDit

Summer 2021

Team 128

Andreas Giannoutsos
sdi1700021

Smaragda Reppa
sdi1600143

Abstract

The following report will describe the summer project in the Internet Technologies course, which is the development of a online social network application, LinkedDit. First we will explain the technologies and the peculiarities of the implementation in the back-end. Then we will explain about the organization and the modification of api. Then we will describe the technologies and the particularities of the implementation of the front-end and finally we will describe the difficulties we encountered in our work and the ways we solved them.

Contents

1. Development Process

For the development process we used the Rapid Prototyping technique. Developing user interfaces with rapid prototyping means making prototypes and simulating future behavior of a website or software system rapidly and efficiently. To explain and evaluate their design ideas, prototypes make it easy for users, investors, developers and designers. Rapid prototyping is a collaborative approach that allows you to work with management, engineers, users, and other designers to create a product.

According to this technique, we initially decided on the pages of the front end. Then we developed them individually. Along with these pages we formed the types of databases that you would need from the back end. Then as we had formed the data and its formulas we made a server to simulate this data. So we worked on our page on a simulation server to see how it handles the data. With this process we had developed the API as well as all the post and get commands that we would need. With that in the end we developed the back end. Finally we connected the 2 implementations for the final application.

2. Back End

2.1. Technologies

The back-end part of the appllication is build with Django. Free and open-source web framework Django is built on Python and follows the MTV architectural pattern. The Django Software Foundation is responsible for its upkeep. Primary goal of Django is to simplify the creation of complex, database-driven websites. Less code, low coupling, rapid development and the don't repeat yourself principle are all emphasized in the framework. As a result of this, Python is used for everything: settings, files, and data models alike. Also available in Django is an optional administrative interface that allows users to create, read, update, and delete data.

The MySQL database is used for storage. MySQL is a relational database management system that is free and open source. One or more data tables in a relational database include data types that may be connected to each other; these relationships assist shape the data. Sql is a programming language used by programmers to create and change data in a relational database. It is also used by programmers for access control to the database. Relational database management systems (RDBMS) like MySQL interact with an operating system to create a relational database in the computer's storage system, manage users and allow for network access. They also permit checking database integrity as well as backups.

2.2. Implementation Details

Django cors headers and Django rest framework libraries were also used extensively for implementation.

Linkedit project contains files generally for the backend eg settings.py. Manage.py is the file that coordinates and activates the database and the server. The server is activated with the command 'python manage.py runserver' in the directory where manage.py is located. Inside the linkedit app are the rest of the backend files. The models from which the database tables are defined can be found in models.py The views.py contain the code that is necessary

for the GET-POST requests by the frontend. In serializers.py are the serializers for each model, necessary for the operation of the application.

2.3. Bonus

In the bonus initially 3 pieces have been implemented, the extraction of the features, the algorithm and the recommendation system. The first is the extraction of information from the packages of posts and posts of the user. Their extraction is done with the use of TfIdf and Count Vectorizers. This data is converted to a size table (total posts X 1000). The next piece is the algorithm implemented in Python and is an iterative algorithm that stops when the error reaches 0.001 and the result is a table of size (total posts X 10). In the last part of the bonus, which is the recommendation system, we take 2 such tables, one from all the posts and one from the user's posts, and we try to find the ones that are most compatible with the user.

3. REST API

The REST API has been implemented and tested with the help of the postman. Postman is a collaboration platform for API development. In this section we will list the get and post methods.

3.1. GET methods

- getAdminAllUsers() get admin all users
- getUser() get user
- getConnectionRequests() get connection requests
- getInteractions() get interactions of users (like, comment...)
- getDiscussions() get discussions
- getConnectedUsers() get all connected users
- getRecommendedPosts() get recommened posts
- getMyPosts() get my posts
- getProposals() get recommended job proposals
- getMyProposals() get my proposals

3.2. POST methods

- postLogin() login
- postLogout() logout
- postForgotPassword() forgot password

- postSignUp() sign up
- postUserSettings() update user settings
- postUserSettingsPassword() change password
- postConnectionRequest() answer to a connection request (friend request)
- postSearchResults() post name of a user and get the search results
- postDiscussion() post the user and start a new discussion or get the already having one
- postMessage() post a messsage on a discussion
- postComment() post a comment on a post
- postPersonalData() change your personal data
- postPost() post a post
- postThumbsUp() post thumbs up or down
- postProposal() post a job proposal
- postApplyUp() post aplly up or down to a job proposal
- postProfileImage() post a profile image

4. Front End

4.1. Technologies

The front-end part of the application is built with Vue.js. JavaScript framework for developing user interfaces and single-page apps, Vue.js is an open-source model-view-viewmodel front end framework. He and the rest of the active core team members built it and manage it. There is a declarative rendering and component composition focus in the Vue architecture. The view layer is the only focus of the core library. Officially maintained supporting libraries and packages provide advanced functionality necessary for complicated applications, such as routing, state management, and build tools.

4.2. Implementation Details

For the development of the front end, the DRY technique was used, which we do not repeat in the code. In addition, some modules have been developed which are reused in the application, thus reducing the lines of the code and making maintenance and development easier.

The modules that are reused are 3. The first one is the Personal Data module. This module is used to view the profile data when one user wants to see the other user's profile. It also has the ability to modify personal data and this is done by the user through the corresponding page. The second module is the post. This module is used to display the

posts by the users themselves and not but also the job ads as with small modifications it can perform the same purpose. The latest model is the User Card. This module is used everywhere in all lists that display the user profile. Appears on most pages such as the network, search, connection requests and notifications, in discussions but also on the admin page. This is displayed with minor modifications for each different case as it has buttons for various actions to the user such as profile view and chat start.

Initially the front end is organized in 3 layouts. One is the home page that contains the link and subscription. The second is the page that only the application administrator has access to. In this there is a list that can be seen by all users.

The last one is the main layout of the application in which the user navigates as soon as he logs in to the application. On this page in the top bar there is an option to navigate all the pages of the application. The first is the Home page which is the "Wall". It shows all the posts of the users but also the user has the option to see his own. In addition, it has the ability to upload a post. The next page is that of the network from which he can see which users are on his network but also search for users by their name. The next page is that of the Job Proposals that is very similar to the "Wall", just the modules have been modified to change their names. On the next page of discussions the user can see old conversations and send a message to them. The next one is the notifications page and it uses the User Card module with a few modifications for both types of notifications. Next we have the page of personal data in which the user can change the details but also their visibility and his profile picture. At the end is the settings page and there you can change the account settings such as passwords and names.

5. Conclusion

During the implementation of the application we encountered several difficulties. The most important was that we had different schedules that we were able to offer at work. For this reason, the rapid prototyping method helped us a lot. In the beginning one member, having enough time available at that time, with help from the other, developed the front end and with the contract that was made in the data forms, the REST API was ready. Later the other member who had in that specific period more time than the other developed the back end having the REST API ready, which helped a lot in its rapid development. And in this way we were able to develop this application.