Employee management and payroll system - Technical report

Cojocaru George, Dumitru Daniel-Antoniu, Popa Ștefan-Eduard

Problem presentation

In a company, people have to be organized in different teams, depending on the position they hold. Each team has specific announcements, tasks and attributes, and the employees should receive these announcements in an effective time. At the same time, people in other teams should not be able to see these messages. There can also be a common group for all of the members of the company where general messages and announcements are sent, for example, if it is a team building or if the employees go on furlough for a certain amount of time. Another aspect of the company is the payment of the employees. They should have a daily payment (and based on that, a monthly salary). These aspects can be digitized and implemented in an application to ease the administration of the employees and their payment.

State-of-the-art

So far, multiple applications have been made to digitize these aspects. Some examples are Workday, Oracle HCM and Sage People. Workday provides cloud-based software for financial management. It helps businesses manage payroll, benefits, HR and employee data. It is considered one of the best HR systems globally. Oracle HCM is another application that covers all aspects of HR management. Some of the pros of this application are the friendly user interface, the easy configuration and the flexible custom pricing.

Solution

As a solution, we made an application for the management of the employees. Given an account, the user can create a new business. In this profile, several posts can be made by the employees. The posts can then be seen in the news feed section. There is also a minimal sarcasm detector that tracks the text of the posts.

To build the application, we used several programming languages and technologies. The backend part is made with the Java Spring framework. One of the reasons for choosing this framework is because of its annotations. They have multiple functionalities, from turning a Java class into a database table to making a rest controller for the endpoints of the application. Another utility is given by its prebuild classes and func-

tions. For example, a HTTP request can be processed by a security filter, and based on the needs, the filter can be customized. Here it can be customized the authentication process.

For the frontend React and TypeScript were used. React is a library for user interface written in JavaScript. It is easy to learn, you can develop an application in a short amount of time, you can reuse the components in other buildings and it has a big community, meaning that if you have any problem, you have people that can help you in effective time. TypeScript is a programming language derived from JavaScript. The main advantage is that it is a static typed language, meaning that potential type errors can be observed at compile time. Another advantages are scalability and readability.

For the sarcasm detector, Python was used. This language is the most popular when it comes to machine learning/artificial intelligence. It is easy to learn and understand, and the syntax is simple compared to other programming languages. Several frameworks and libraries were made, such as PyTorch, Pandas and Scikit-learn. Another useful library is Transformers, which gives tokenizers and models in order to process text in a corpus, train it and classify it in categories.

Results

To use the application, the first step is to authenticate based on your Google account. The JWT authentication generates 2 different tokens: one for access and one for refresh. The access token is used to verify the user and it is available for 1 hour. If the access token is expired, the refresh token is used in order to generate another access token. The refresh token is available for 24 hours. After that, the user has to login again in order to generate new tokens. After the user is authenticated, he can create a business group, and set its name, domain and location. If an user creates the business, he automatically becomes an employee. In a business group, posts can be made. An user can make a post only if it is an employee. An user can see a post in a business only if it is an employee at the same business. Only the employee that created a post can update it or delete it. The posts are added to the news feed of the business.

For the posts part, a sarcasm detector was implemented. If a comment is considered sarcastic, the post is not made, and a popup is shown to the user and explains why the post is not created. A corpus was made by grouping a database with article titles, labeled as sarcastic and non-sarcastic (the corpus had around 26.000 examples)

[7] and 10.000 Reddit comments in the SARC corpus (not necessarily in a specific context or topic) [8]. The results are the following: for data extracted from the article titles database, but not used in the training process, and some Reddit comments from SARC that were not used in the training process, the accuracy is 0.8888888888888888, the precision is 0.9545454545454546 and the f1-score is 0.865979381443299.

Comparison

Compared to other solutions, an additional feature of our application is the presence of a sarcasm detector. Many studies were made on this topic and many corpora were created, but none of them were specific for the context of working environment and business posts. One of the closest dataset to such a corpus is SARC, that groupes reddit comments from different topics. The sarcasm detector is not the best when it comes to interpret comments and posts in the context of company work. For an example of 130 comments generated by Chat GPT in the context of work environment, 70 sarcastic and 60 non-sarcastic, the accuracy is 0.46153846153846156, the precision is 0.5 and the f1-score is 0.46153846153846156. Another aspect of this sarcasm detector is that it tends to classify some phrases that contain the ! symbol as being sarcastic, even though the text is not sarcastic.

Future work

In the future, the project can be extended and multiple functionalities can be added. A payroll system can be made such that each employee can receive the daily pay. Another useful functionality could be a calendar with the vacation period of each employee, the days off due to holidays and special days in the company (for example, team buildings). The sarcasm detector can also be improved, by adding other comments from the SARC corpus, looking for business posts and work environment texts and extracting them through web scraping and/or by creating some specific examples.

Bibliography

[1] https://www.geeksforgeeks.org/why-choose-react-for-web-development/.

[2] https://dev.to/laxminarayana31/reasons-to-choose-typescript-over-javascript-16nd.

[3] https://www.newhorizons.com/resources/blog/why-is-python-used-for-machine-learning.

[4] https://www.workday.com/.

[5] https://www.oracle.com/human-capital-management/.

[6] https://www.sage.com/en-gb/sage-business-cloud/people/.

[7] https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection.

[8] Mikhail Khodak, Nikunj Saunshi, Kiran Vodrahalli. A large self-annotated corpus for sarcasm. `https://arxiv.org/pdf/1704.05579`.