Feed-Me-Back: An interview scheduler and feedback app

Professor: Rakesh Ranjan

Shraddha Nayak, Kanika Khanna, Arushi Bhatla, Shivani Suryawanshi
San Jose State University, One Washington Sq., San Jose 95192

shraddha.nayak@sjsu.edu

kanika.khanna@sjsu.edu

arushi.bhatla@sjsu.edu

shivanishivanand.suryawanshi@sjsu.edu

Abstract— **There are thousands of people who appear for interviews everyday. There are several job search portals that help candidates in searching for the right job as well as the company to search for the right candidate. However, there is no existing platform that schedules interview and generates a report automatically for an interview. This report focuses on three goals: 1. To help a company HR to keep track of the list of candidates for interview, read the feedback report and decide whether to hire a candidate or not. 2. To help candidates who appear for interviews to read a feedback report that can help them to work on their areas of improvement so that they can be ready for their upcoming interviews. 3. To help the interviewer to save their time in writing a detailed feedback of a candidate they have recently interviewed, instead the app will do it for them using SimpleNLG.**

Keywords— **Interview Schedule, Feedback Report, Natural Language Generation(NLG), SimpleNLG, React JS, MongoDB, Python Flask**

I. INTRODUCTION

When a candidate goes through a job search, it becomes an exhaustive process since it may lead lead them to face several rejections. This makes a candidate desperate for feedback on what went wrong and areas of improvement on interview skills. The problem is that employers have limited time in consoling an upset candidate and talk about the real reason for their rejection. This report focuses on providing a feedback report automatically using SimpleNLG, a Java based NLG Library for such candidates who wants to know an interviewer's opinion on the interview.

On the other hand, giving the feedback to a candidate can create a positive impact for their company. There is a possibility that they liked a candidate and might want to hire them later in a less competitive environment. So, in such a case giving a positive feedback can help them to maintain a good relation with candidates. It is important for startup companies to maintain a reputation. If they share their valuable feedback, candidates may come back to the company and interview again and share their experiences with their friends and social media which can highly benefit companies. This can reflect a company's transparency in its hiring practices.

It is a common practice for an interviewer to submit a report on a candidate's performance in interview to hiring manager but this process takes a lot of them and it may be inconvenient for most of them to devote time in this. This app renders a set of skills from the candidate's resume and asks interviewer to rate them from 1 to 5. Instead of generating and writing long reports, interviewer can save time and simply fill this form. The skills and their corresponding rating are given as input to SimpleNLG java API that generates a detailed report for a candidate mentioning about what went well and areas of improvement for a candidate.

Thus, it helps a company to maintain a good relation with candidates so that they continue to apply to their company without spending time in creating a report manually or contacting them directly.

II. DESIGN

In this paper we propose an architecture that generates a report of the candidate's performance in an interview.

A. DESIGN METHODOLOGY

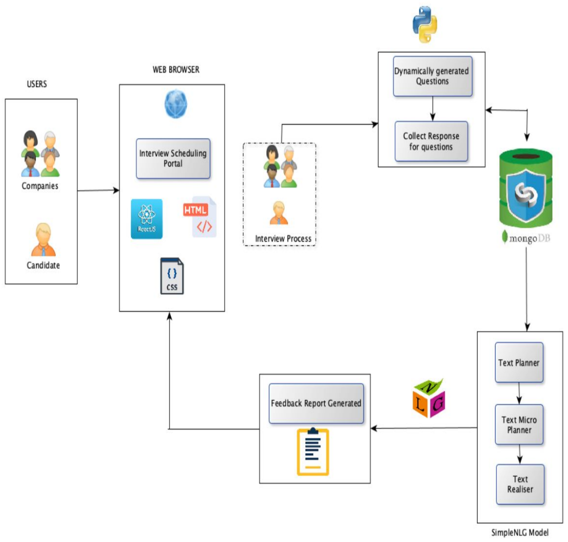
We present an interview scheduler and feedback system as a Web application. The system design contains three modules.

Web browser module connects system users with the NLG and dynamic question generation module. This module allows users to register themselves and manage their account-related activities. Besides this, this module forwards the user's data and displays feedback reports, interview schedules.

The Dynamic question generation module extracts skills from the resume of a candidate. These skills are used to generate a feedback form. Feedback form allows an interviewer to rate the skills of a candidate in-between range of 1 to 5 based on his performance in the interview.

NLG module communicates with other modules and generates report. This module is the basis of our architecture. It generates detailed report for a candidate as well as hiring manager based on the feedback form submitted by the interviewer.

B. ARCHITECTURE DIAGRAM



III. IMPLEMENTATION

A. USER INTERFACE - ReactJS

It is a JavaScript Library used for building User Interfaces. It builds encapsulated components that manage their own state, then compose them to make complex UIs.

The first page of the UI consists talks about product features and its possible integration in the future. It gives our personas a good idea about the product and how it can be used.

When an admin (HR of the company) wants to login into the system, she can login with Gmail and avoid all the hassle of filling in personal information. Instead, Gmail can allow a smooth login into the system. The product user only have to choose login as Admin. Similarly, interviewers have to choose, login as Interviewer and candidates who have to give an interview has to select Login as Candidate.

When an admin logs into the system, he/she can see the list of open positions in their company. The open positions will contain details such as Name of the position, Job ID and job description etc. Admin can click a button called Get Candidates that will show a list of all candidates who have scheduled interview for that position. In case the interview is done for that candidate, the admin can generate a report for that candidate. In the same page, admin can add shortlisted candidates whose interview needs to be scheduled. The candidates will receive a notification to schedule an interview. Admin can also assign the interviewers for that candidate who will receive a notification to open the portal to submit the feedback,

Candidates can login in to the portal as a candidate and can get information about the company and schedule his/her interview.

Once the interview is done, the interviewer can login into the system and see the list of candidates he has to interview. For the candidates whose interview is over, he can fill a feedback form abd rate a candidate on the basis of skills from 1-5. 1 being poor and 5 being excellent. Once this this done, candidates and HR will receive a notification that a report has been generated to read the feedback. The candidate can login again and download the report to read the feedback and HR can feed the feedback report to make a hiring decision.

B. MONGO DB COLLECTIONS

MongoDB is an unstructured database which allows us to create our collections without having the need of defining the table structure. Any new field can be inserted irrespective of the schema, since it has a schemaless structure. We used MongoDB Atlas as our database deployed across AWS.

C. SimpleNLG

A simpleNLG is a realization engine for English. It helps in creating syntactic structures and linearizing them. [4]

In this system, the interviewer has to feed in answers by rating a candidate's skills from 1 to 5. If an interviewer has rated a candidate in a skill for Java is 4, then it is fed to the simpleNLG. The rate 4 is mapped as good. So the output written by SimpleNLG is "**This skill in Java can be considered as good**". In a similar way the SimpleNLG generates a report for various texts.

D. API - using Python Flask

Many Application programming interfaces are created in order to interact with the frontend and backend. We are using python as our language and flask as our framework. Flask applications are usually written on a blank canvas which is very well suited for the APIs we built. We are using JSON formatting for returning our data through API's.

E. Authentication - using Gmail

We have used a google-sign in authentication method for our application. This will remove the burden of users to remember their usernames and passwords for their accounts. All requests to the Gmail API has to be authorized by an authenticated user in our web application. The user will be prompted to grant access to data in his google account. After the user gives his consent, the API requests and gets credentials to access the Gmail API for signing into our application.

IV. CONCLUSION

Our application aimed at two personas, the candidate and the recruiter, served both of them in a rightful manner. The candidate will now be able to schedule interviews seamlessly and also view his performance report at the end of any interview. He will now be able to assess his strong and weak points well. He will have a have a clear understanding of what went well and what areas that needs polishing.

From the recruiters point of view, the process of giving feedback to the candidates became quiet easy. They need to just fill in a questionnaire and our tool(using Natural Language Generation) will do the work for him. The tedious job of writing individual feedback reports can now be done effortlessly in seconds.

The feedback report can be read by the candidate as well as the admin (Hiring manager) who makes a decision to hire him or not. This saves him the follow up time that he would have used for having a discussion with the recruiter who conducted the candidate's interview.

Therefore, Feed-Me-Back provides a win-win situation for each of our personas for a seamless interview experience.

VI. FUTURE IMPROVEMENTS

The application can be used to integrate with various job portals such as LinkedIn, Glassdoor etc so that it becomes convenient for employers to take emailIDs and Candidate's resume from these job portals.

We could enable notifications such that candidates, admins and interviewers can be notified wherever the candidate has to schedule an interview, download a report or fill a form.

References

[1] https://www.thebalancecareers.com/must-employers-tell-applicants-why-they-weren-t-hired-1919151

[2] https://www.aclweb.org/anthology/W09-0613.pdf

[3] https://github.com/simplenlg/simplenlg/wiki/Section-0-%E2%80%93-SimpleNLG-Tutorial

[4] https://www.aclweb.org/anthology/W09-0613.pdf