인사 자원 관리 시스템을 위한 챗봇 설계 및 구현

고경규[O], 진호용, 문유빈, 정은성
홍익대학교 소프트웨어융합학과
k5602o@nate.com, hoyong8532@naver.com, moon.pinnamon@gmail.com,
ejung@hongik.ac.kr

Design and Implementation of a chatbot for human resource management systems

Kyoungkyu Ko[O], Hoyong Jin, Yubeen Moon, Eun-Sung Jung
Department of Software and Communications Engineering

요 약

회사자원을 효율적으로 관리하기 위해서는 다양한 기능들을 수행할 ERP 시스템이 필요하다. 하지만 대체적으로 중소 규모 회사는 이러한 ERP 시스템을 관리하는데 비용적인 문제에 직면할 수 밖에 없기 때문에 비용도 적게 들면서 간편하게 구현 가능한 챗봇을 활용하였고 챗봇 시스템 개발에 컨테이너 기술을 채택하였다. 컨테이너 기술은 Host의 OS, 인프라 등에 영향을 받지 않고 어떠한 환경에서도 사용할 수 있고, 배포 시 속도가 빠르며, 새로운 기능들을 추가해야 되는 경우 컨테이너만 추가해서 환경을 구축할 수 있는 장점이 있다. 본 논문에서는 컨테이너 기반 챗봇 소프트웨어 구성이 어떻게 되어 있으며, 해당 기능들이 컨테이너 방식으로 어떻게 동작하는지 기술하였다.

1. 서 론

회사를 효율적으로 유지하기 위해서 사원을 비롯한 회사자원을 관리해야 한다. 이를 관리하기 위해서 다양한 기능들이 필요하게 되었으며 해당 기능들을 수행할 ERP 시스템이 필요하였다. 하지만, 대부분의 중소 규모 회사는 이러한 시스템을 관리하는데 비용적인 문제에 직면하게 된다. 따라서, 이러한 문제점을 고려하여 비용은 적게 들면서 간편하게 시스템을 관리할 수 있는 챗봇을 활용하였다.

기존에 개발된 메신저 기반의 bot에 회사 시스템 관리를 용이하게 하는 데 필요한 여러 가지 기능들을 구현하였다. 해당 기능에는 인사 자원 관리를 위한 사원들의 정보, 출/퇴근 관리, 작업 현황, 설문조사 등이 있으며 이를 웹 인터페이스와 연결하여 웹에서 데이터베이스를 분석할 수 있도록 했다. 또한 회사의 상황에 따라 또 다른 기능이 계속 추가될 수 있음을 감안해 관리와 속도에 대한 측면을 고려하게 됐고 이에 대한 솔루션으로 컨테이너(Container) 기술을 채택했다.

우리는 컨테이너를 이용해서 챗봇 소프트웨어 구조를 설계 및 구현했다. 각 컨테이너에 챗봇 각각의 기능을 수행하는 응용 프로그램(Application)을 올리면 관리자는 웹을 통해서 사원들을 관리할 수 있고, 사원들의 데이터들이 저장되는 저장소를 제공하며 설문조사 시 사원들에게 설문송부 및 모니터링을 할 수 있다.

본 논문은 다음과 같은 구조로 작성되었다. 2절에서 도커와 컨테이너에 대한 배경 설명을 하고, 3절에서는 챗봇의 전반적인 구현 구조 및 챗봇에 사용된 컨테이너 기술들의 환경 및 구현 결과에 대해 자세히 기술한다. 마지막으로 4절에서는 결론으로 논문을 마무리한다.

2. 배경 기술

일반적으로 회사에 시스템을 배포할 때 먼저 다른 서버에서 테스트를 진행한 후 회사에 배포를 한다. 하지만 만약 테스트한 환경과 회사의 환경이 다르다면 또는 동일한 환경이어도 배포 서버에 다른 작업이 진행중인 경우 해당 작업 환경과 충돌이 있을 수도 있다. 컨테이너 기술은 이러한 문제점들을 보완해주는데, 컨테이너 기술을 이용하면 각 컨테이너는 독립적인 공간이기 때문에 서버의 OS 나 인프라에 영향을 받지 않는다. 또한 Host 의 Kernel 을 공유해서 사용하기 때문에 필요한 라이브러리 및 실행 파일만 존재해 가볍고 속도 측면에서도 빠르다. 또한 컨테이너 기술을 사용하면 백엔드 프로그램, 데이터베이스 서버, 메시지 큐 등 다양한 프로그램 및 실행 환경 등 어떠한 환경도 추상화할 수 있고 동일한 인터페이스를 제공하여 프로그램의 배포 및 관리를 용이하게 한다.

3. 챗봇 기능 구조 설계 및 구현

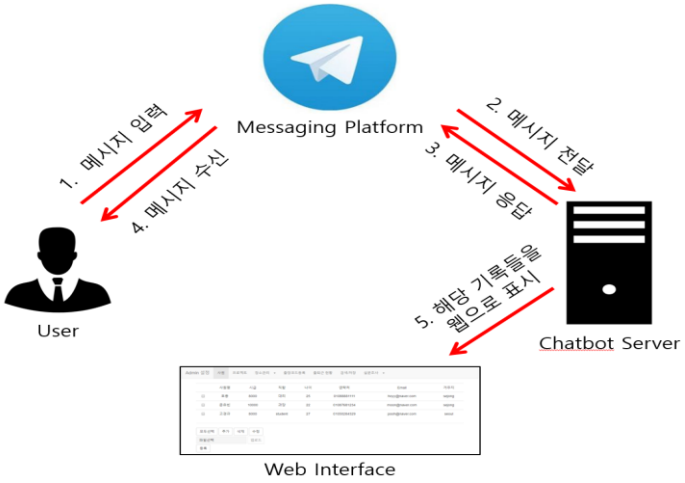

그림 1 챗봇 구성도

이번에 개발한 챗봇은 그림 1 처럼 구성이 되어 있다. 하나의 기능을 예로 들면 해당 사용자가 출/퇴근 기록을 텔레그램이라는 플랫폼에 입력한다. 그럼 텔레그램이 이를 서버로 전송하고 이 서버에서 해당 기록을 DB 에 저장한다. DB 에 저장된 기록을 다시 텔레그램을 통해 사용자에게 보여주고 또한 웹에서 DB 에 저장된 기록들을 불러와 보여줄 수도 있다. 이러한 과정을 서버의 컨테이너들이 처리한다. 메시지 플랫폼을 텔레그램으로 한 이유는 다른 플랫폼들은 알림 메시지 송부 시 과금 정책이 있어 회사 측에서 부담이 될 수 있기 때문에 과금 정책이 없는 플랫폼을 채택했다.

3.1 컨테이너 기반의 챗봇 소프트웨어 구조

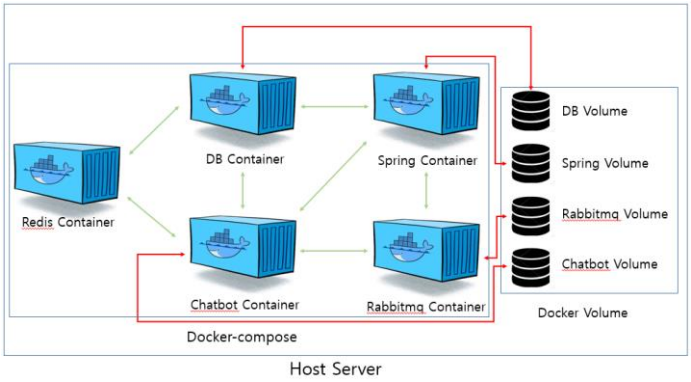

그림 2 챗봇 소프트웨어 구성도

챗봇 소프트웨어 구조(그림 2)를 보게 되면 하나의 컨테이너가 독자적으로 기능을 수행하는 것이 아니라 여러 컨테이너들이 상호 작용을 하며 시스템이 동작하기 때문에 Docker-Compose 를 통해서 각각의 컨테이너를 동시에 관리할 수 있도록 했다. 또한 컨테이너의 데이터가 휘발성인 점을 고려해 해당 컨테이너에서 작업에 필요한 파일들을 호스트(서버)의 도커 볼륨(Docker Volume)과 연동시켰다. 이러면 컨테이너에 문제가 생기거나 재시작되더라도 해당 파일들이 저장된 상태로 도커 볼륨에서 불러와지기 때문에 작업상태는 계속 유지된다.

3-2. Docker-compose

Docker-compose는 Multi 컨테이너의 도커 응용 프로그램을 정의하고 실행하기 위한 도구 yml 파일을 사용하여 응용 프로그램의 서비스를 구성할 수 있다.

이번 챗봇 프로젝트에서 이 yml 파일에 웹 서비스를 하는 Spring 컨테이너와 Rabbitmq 컨테이너의 특정 포트들을 Host의 특정 포트와 연결해서 바로 접근할 수 있도록 했고 각 컨테이너에 필요한 계정들을 설정해줬으며 데이터 유지가 필요한 컨테이너에 도커 볼륨을 연동시켜 시스템 환경을 구축했다. 이 yml을 실행시키면 yml에 정의된 설정값과 컨테이너들이 실행되고 변경사항이 생기면 yml 파일을 수정해서 다시 실행시키면 된다.

Docker-compose에 정의된 컨테이너들은 다음과 같다. Python으로 작성된 메시지 플랫폼의 기능들이 동작하는 챗봇(Chatbot) 컨테이너, 사원들의 데이터가 저장되어 있는 DB 컨테이너, 설문조사 시 사원들에게 설문을 보내주고 설문 현황을 모니터링 해주는 Rabbitmq 컨테이너, 관리자가 웹에서 사원들을 관리할 수 있도록 해주는 Spring(웹) 컨테이너, 인라인 키보드 버튼 구현을 위한 Redis 컨테이너, 이렇게 5개의 컨테이너로 이루어져 있으며 각 컨테이너들이 서로 통신을 하며 여러 기능들을 수행한다.

3.3 설문조사 기능 구조

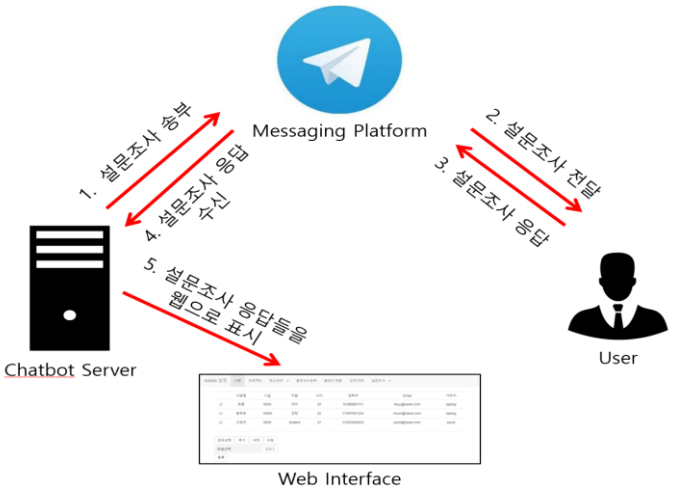

그림 3 설문조사 기능 구성도

이번 기능에서 제일 주목할 만한 기능은 설문조사 기능이다(그림 3). 회사에서 선물 지급, 점심 식사 여부 등의 조사를 해야 하는 경우를 위해 개발했는데 설문조사를 관리자가 웹에서 관리할 수 있도록 구현하였으며 정해진 날짜에 정해진 사원수대로 설문조사 내용을 보내도록 했다. 즉, 웹에서 설문조사를 메신저 봇(챗봇)으로 보내면 메신저 봇이 사원들에게 보내주는 구조이다. 이를 좀 더 확장해서 설명하면, 기존의 챗봇이 사용자(클라이언트)가 챗봇(서버)에게 메시지를 보낸 다음 해당 메시지에 대한 응답을 다시 사용자(클라이언트)에게 보내는 방식이었다면 설문조사 기능의 경우는 반대로 챗봇(서버)이 사용자(클라이언트)에게 설문조사를 보내고 이 설문조사를 사용자가 실시한 다음 해당 결과를 다시 챗봇으로 보내는 구조이다.

하지만, 만약 웹에서 설문조사를 보내는 도중에 챗봇 코드에 문제가 발생하면 결국 사원들에게 설문조사가 전달되지 않는다. 이러한 상황을 방지하기 위해 중간에 메시지 큐를 놓아 웹에서 보낸 설문조사를 메시지 큐에 저장한 후 메시지 큐에 저장된 설문조사를 메신저 봇이 가져가도록 했다. 이러면 메신저 봇이 설문조사를 가져가는 도중에 문제가 발생해도 메시지 큐에 설문조사가 저장되어 있기 때문에 다시 가져가기만 하면 된다. 이러한 메시지 큐를 사용하기 위해 Rabbitmq를 이용했다.

3.4 챗봇 기능 구현

1) 웹에서의 관리자 로그인

관리자의 웹 페이지로 로그인을 하려면 웹에서 입력한 비밀번호와 DB 에 저장되어 있는 비밀번호 값과 비교했을 때 일치해야만 로그인이 된다. 또한 웹에서 사원 추가, 출장, 장소 등록 및 각 사원정보 및 기록을 확인해야 하는 경우 해당 정보들을 웹을 통해 DB 에 저장하고 DB 에 저장된 값을 웹으로 보여준다. 이는 DB 컨테이너와 Spring(웹) 컨테이너가 통신하는 구조이다.

2) 인라인 키보드 구현

챗봇을 버튼 형식으로 만들기 위해 인라인 키보드를 파이썬 코드로 만든 후 키보드 코드를 레디스에 저장한다. 인라인 키보드 입력 이벤트가 발생하면 레디스에서 검사한 후 일치하면 값을 가져옴과 동시에 DB 에 저장한다. 이는 Redis 컨테이너와 DB 컨테이너, 챗봇 컨테이너가 서로 통신하는 구조이다.

3) 챗봇 및 웹을 통한 사원 관리

웹에서 사원을 등록하면 해당 정보들이 DB 에 입력된다. 이 DB 에 등록된 사용자여야만 챗봇에 접근 가능하고 버튼 형식으로 된 챗봇에 출/퇴근 기록을 입력하면 Python 으로 만들어진 챗봇 기능들이 해당 기록들을 DB 에 저장하고 이 DB 에 저장되어 있는 데이터들을 웹에서 불러와 관리할 수 있다. 이는 챗봇 컨테이너, Redis 컨테이너, DB 컨테이너, Spring 컨테이너가 서로 통신하는 구조이다.

4) 설문조사 송부

먼저, 웹(Spring 컨테이너)에서 정해놓은 시간에 설문조사를 등록하면 해당 시간에 설문조사 내용이 DB 에 등록된 사원 수에 맞춰서 Rabbitmq 컨테이너로 전송된다. Rabbitmq 컨테이너는 이를 저장하고 있다가 챗봇 컨테이너가 호출하면 설문조사를 챗봇 컨테이너로 보내는데, 챗봇 컨테이너는 Rabbitmq 컨테이너를 통해 받은 설문조사를 텔레그램으로 각 DB 에 등록된 사원들에게 전송해준다. 각 사원들은 Redis 컨테이너에 의해 버튼 형식으로 된 설문조사에 대한 응답을 할 수 있다. 이는 챗봇 컨테이너와 DB 컨테이너, Spring 컨테이너, Rabbitmq 컨테이너, 즉, 5 개의 컨테이너가 모두 통신하는 구조이다.

4. 결 론

본 논문에서는 인사 자원 관리 시스템을 위한 챗봇 설계 및 구현에 대해 기술하였다. 챗봇 구현을 위해 컨테이너 기술을 사용하여 유연한 소프트웨어 구조 및 신속한 배포가 가능함을 보였으며 챗봇을 이용한 기본적인 인사 관리 및 설문조사와 같은 양방향 메시지 교환 기능을 구현하였다.

참 고 문 헌

[1] Human-resources-management, https://www.ecount.co.kr/ecount/product/payroll_human-resources-management

[2] Docker documentation, https://docs.docker.com/

[3] Why switch to Telegram?, https://telegram.org/

[4] Koh Mitsuda, Ryuichiro Higashinaka, and Yoshihiro Matsuo, “What information should a dialogue system understand?: Collection and analysis of perceived information in chat-oriented dialogue”, pp.1-9

[5] Transforming Chatbot Responses to Mimic Domain-specific Linguistic Styles, “Siddhartha Banerjee, Prakhar Biyani , Kostas Tsioutsiouliklis”, pp.1-9

[6] RabbitMQ Features, https://www.rabbitmq.com/