1. 프로젝트 개요

* - 기업이 고객 정보를 관리하는 데 도움을 주는 고객 관리 시스템
* - 고객 정보를 등록하고 관리하며, 필요한 경우 데이터를 검색하고 수정할 수 있는 기능을 제공

2. 기술 스택

* - Frontend: HTML5, CSS3, JavaScript (ES6), JQuery  
  - Backend: Node.js, Express.js, EJS (Embedded JavaScript)  
  - Database: MySQL + docker, Sequelize

3. 기능

-회원 : 로그인, 로그아웃 / 회원가입 / 가입시 이메일 인증 / 마이페이지 / 정보수정페이지  
-게시판 : 게시글 전체 조회 / 게시글 상세 조회 / 게시글 작성, 수정, 삭제 /

4. 구현

1) Frontend

- JavaScript (ES6)를 활용하여 클라이언트 측의 상호작용을 개선, 인터렉티브한 기능 추가

* 2) Backend
* -  Node.js와 Express.js를 사용하여 서버를 구축
* -  EJS를 사용하여 동적인 웹 페이지를 생성
* - Nodemailer를 사용한 사용자 이메일 인증 기능
* 3) Database
* - MySQL을 사용하여 고객 정보를 저장, Sequelize ORM을 통해 데이터베이스와의 상호작용을 처리

5. 결과 및 성과

* - 기업이 고객 정보를 효율적으로 관리할 수 있도록 도와줌으로써 업무 효율성을 높임
* - Sequelize를 사용하여 데이터베이스를 관리함으로써 데이터의 안정성과 일관성을 유지

6. 개발 과정 문제점과 해결

1. 1) 문제점

* - 회원가입 기능을 구현함에 있어서 처음엔 사용자가 입력한 데이터의 유효성 검사를 할 때, 클라이언트 사이드에서 검증을 하고 서버로 데이터를 전송했습니다.
* 하지만 이러한 방식은 사용자 입장에서 충분히 데이터 변조가 가능했고
* 이러한 문제점을 해결하기 위해 fetch통신을 사용하여 비동기 방식으로 사용자가 입력한 데이터를 서버로 전송해 유효성을 검사한 뒤 응답을 전송하는 방법을 사용했습니다.

1. 2) 문제점

* - sequelize orm을 이번 프로젝트에서 처음 사용하다 보니 아직 숙련도가 미숙하여 차질이 많았습니다. 기존에 사용하던 sql문법과는 비슷하지만 살짝 다른 독자적인 문법과 용어들이 존재하여 새로운 지식을 배울 수 있는 기회가 되었습니다.
* 3) 느낀점

Vs code로 작성한 코드들은 Git을 사용해 버전 관리를 했고,

데이터베이스는 프로젝트 초기엔 microsoft의 azure를 사용하여 Ubuntu VM에 docker를 설치하여 mysql을 컨테이너에 올려 사용했습니다.

하지만 이 과정에서 mysql 관련 보안 설정을 잘못한 탓에 랜섬웨어에 걸리고 말았고,

그때까지 작성해온 DB와 데이터들이 모두 사라지는 일이 발생했습니다.

이후 amazon의 ec2를 사용하여 이전보다 더 나은 보안환경을 가진 데이터베이스를 구축했습니다. 이를 통해 프로젝트를 진행함에 있어서도 클라우드나 데이터베이스의 보안에 좀 더 신경을 써야 한다는 점을 배울 수 있었습니다.

7. 향후 계획

* - 보완사항 및 추가내용
* 시간이 촉박하여 아직 구현을 하지 못한 기능들이 많습니다.
* 게시글 좋아요, 댓글, 조회수 기능 / 회원정보 수정, 탈퇴, 차단 기능 /

Java로 구현했던 기능들이지만 이번 프로젝트에서 다시 한번 Node.js 로도 구현을 해볼 예정입니다.