**project DOBEE**

**made by**

**DOBISFREE**

이욱재

이혜리

김정균

박성호

최경열

김광민

**목 차**

**팀 소개**

1. DOBEE?
2. 팀원들

**프로젝트 개요**

1. 프로젝트 선정 배경
2. 프로젝트 주제 및 설명
3. 애자일 방법론?

**기획**

1. 일정관리
2. 시나리오
3. 요구사항 정의서
4. 요구사항 명세서
5. USE CASE
6. 코드규칙

**구현**

1. 스토리보드
2. 프로젝트 구조
   1. 프론트 사이트맵
   2. 백엔드 구조도
3. ERD
4. 클래스 다이어그램
5. 기능별 사용 기술
6. 트러블 슈팅

**별도 자료 첨부**

1. 요구사항 정의서 (가-1)
2. 요구사항 명세서 (가-2)
3. 코드규칙 (가-3)

**팀 소개**

1. **DOBEE?**

****

약 1개월 동안 준비한 비트캠프(강남학원) 1조의 파이널 프로젝트 입니다. 6개월 간의 배운 것을 총 활용해서 만든 프로젝트이며, 6명의 팀원이 함께 만들었습니다.

팀이름의 뜻은 Do! Bee 로 꿀벌처럼 열심히 일을 하자! 는 뜻과 집요정 도비가 자유 얻었듯이 DOBEE를 통해 업무로부터 자유를 얻자라는 이중적인 의미를 갖고 있습니다.

1. **팀원소개**

이욱재 이혜리 김정균 최경열 박성호 김광민 총6명으로 구성되었으며, 1명을 제외한 모든 이가 전부 비전공자임에도 불구하고 현실적인 계획과 실행으로 무사히 프로젝트를 마치게되었습니다.

**프로젝트 개요**

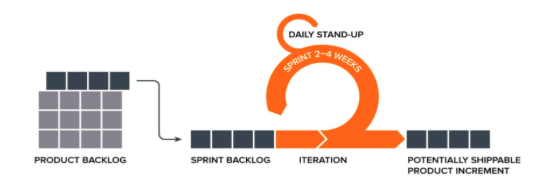
1. **프로젝트 선정배경**

저희는 아직 설립한지 얼마 안되고 체계가 잡히지 않은 4-5인으로 구성된 스타트업을 가정해보았습니다. 소규모 스타트업 입장에서는 기존의 ERP 프로그램은 불필요한 기능이 많고 복잡하다는 문제점이 있었습니다. 또한 소규모 스타트업의 규모에 적합한 프로젝트 관리 툴이 필요하다는 문제의식이 있었습니다. 이를 바탕으로 ERP와 프로젝트 관리 측면에서 필요한 기능만을 갖춘 DOBEE 프로젝트를 만들게 되었습니다.

1. **프로젝트 주제 및 설명**

DOBEE는 소규모 회사에서 프로젝트 관리 기능을 하면서 꼭 필요한 근무관리와 비용관리만을 추가하여 최대한 프로젝트에만 집중 할 수 있는 환경을 제공합니다. 프로젝트 관리 기능에는 애자일 방법론을 적용하여 프로젝트 진행에 있어 자연스럽게 애자일 방법론을 따라갈 수 있도록 직관적으로 만들었고, 프로젝트를 기획하고 업무를 배정하는 팀장과 일을 받는 사원간의 고정되고 수직적인 관계를 최대한 줄이고, 오직 프로젝트를 위한 수평적인 관계가 되도록 하였습니다.

1. **애자일 방법론**

****

애자일이란 어떤 프로젝트를 진행함에 있어 팀원들이 어떤식으로 업무를 진행하고 개발할지에 대한 방법론입니다.

애자일이란 방법론 안에서도 여러가지 변형된 방법론들이 있고, 저희는 이것들을 조사하면서 가장 핵심 키워드를 “공유와 신뢰" 를 정했고, 이 추상적인 개념을 실제로 사용하려면 필요한 것이 결국 수직적인 구조의 탈피와 활발한 커뮤니케이션, 유동적인 업무 진행에 대한 빠른 시각화 전달이라고 생각했습니다.

그리고 이를 구현하기 위한 기능을 담았고 DOBEE을 이용해서 프로젝트를 진행하면 자연스럽게 애자일 방법론을 이용해 프로젝트를 개발 할 수 있습니다.

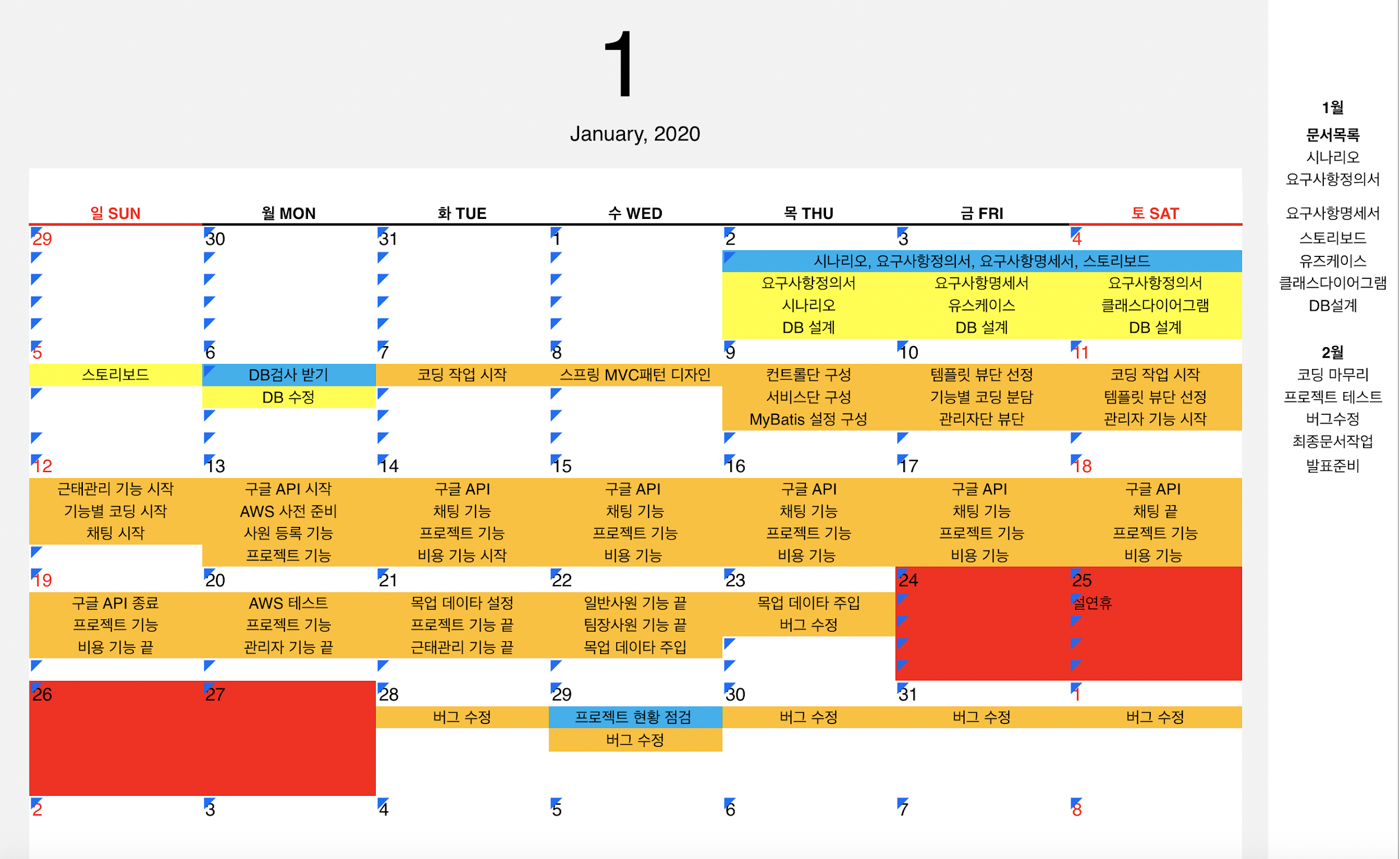
**기획**

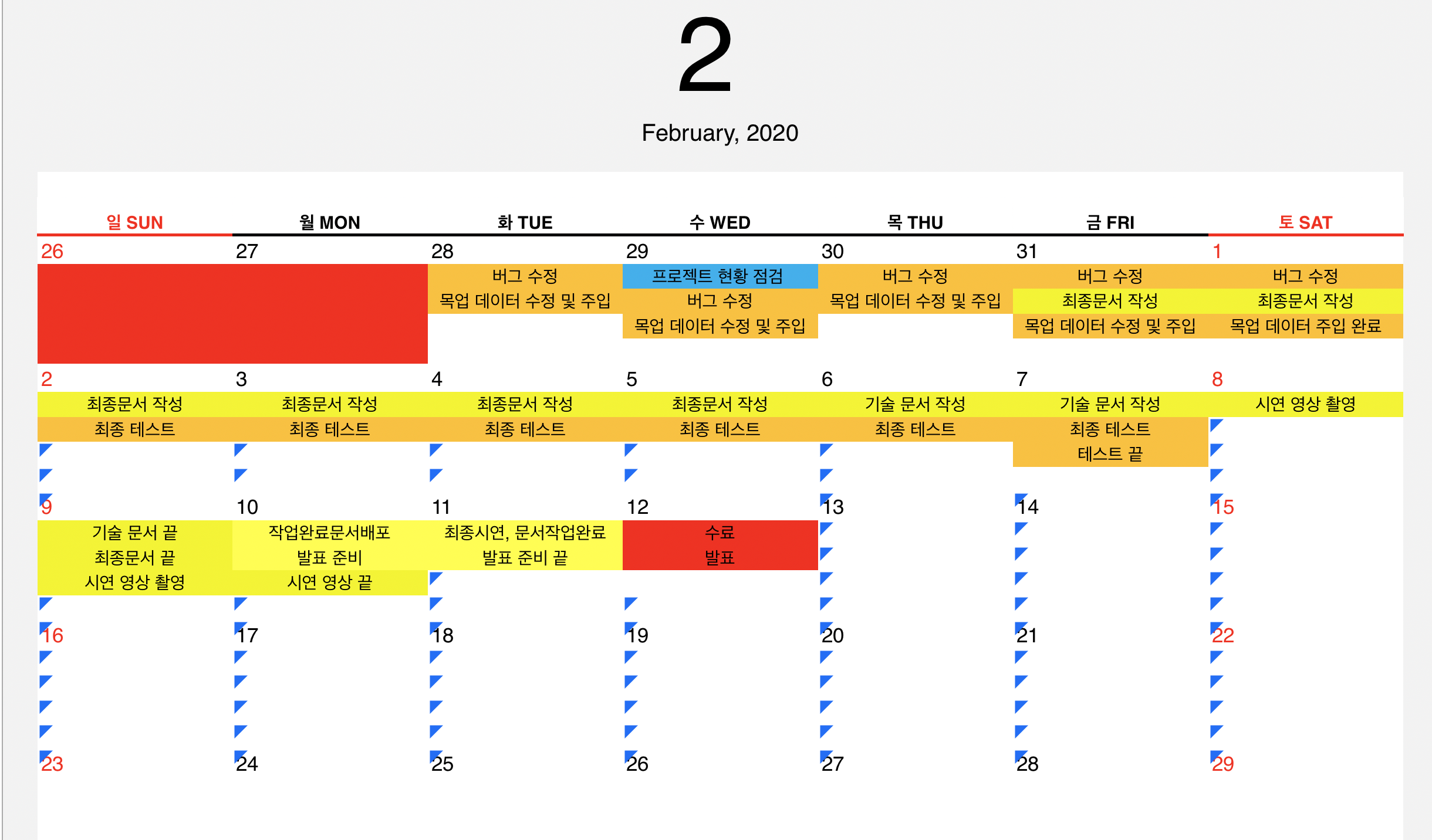
1. **일정관리**

프로젝트 기간 : 20-01-02 ~ 20-02-11

노랑색은 코딩작업을 제외한 문서작업 이며 주황색은 코딩작업입니다.

설날이 중간에 있어 오히려 일정관리하기가 많이 힘들었습니다. 그래도 주말에도 계속 학원에 나와 힘든 일정을 열심히 소화하였습니다.

****

****

1. **시나리오**

**요구사항**

회계팀이나 인사팀이 따로 없는 소규모 회사의 입장에서는 기능이 너무 많은 ERP는 오히려 피곤하게 만들고 진짜로 하고 싶은 일에 집중력을 잃게된다. 따라서 정말로 필요한 기능인 간단한 비용정산기능과 근태관리기능 그리고 실제로 프로젝트 업무를 진행하기 위해 필요한 프로젝트 관리기능만있었으면 좋겠다.

**사이트 이용자**

사이트 이용자는 일반 사원, PM 사원, 관리자로 나뉜다.

1. 일반 사원은 출/퇴근 기능, 공지사항 열람 기능, 근태 관리 기능, 비용 처리 기능, 프로젝트 참여 기능, 채팅 기능을 사용할 수 있다.

2. PM 사원은 출/퇴근 기능, 공지사항 열람 및 작성 기능, 근태 관리 및 승인 기능, 비용 처리 기능, 프로젝트 생성, 관리, 참여 기능, 채팅 기능을 사용할 수 있다.

3. 관리자 사원은 사원 등록 및 관리, 팀 관리, 법인 카드 등록 및 관리 기능을 사용할 수 있다.

**페이지 구성**

팀장(PM) / 일반사원의 프론트 구성은 다음과 같다.

**1. 로그인**

1.1. 회원은 이메일과 비밀번호를 입력하고 로그인할 수 있다.

1.2. 입력한 이메일과 비밀번호에 대해서 유효성 검사를 한다. 아이디 혹은 비밀번호를 입력하지 않은 경우 해당 필드에 ‘이 항목을 필수로 입력해주십시오’라는 문구가 삽입된다. 아이디 혹은 비밀번호가 일치하지 않는 경우 로그인 창 아래에 ‘로그인 실패, 아이디와 비밀번호를 확인하세요’라는 문구가 삽입된다.

1.3. 메일 찾기 버튼 클릭시 등록된 메일을 찾을 수 있는 페이지로 이동한다.

1.4. 비밀번호 찾기 버튼 클릭시 비밀번호 찾기 페이지로 이동한다.

**2. 메일 찾기**

2.1. 이름과 휴대폰 번호를 입력해 메일을 찾을 수 있다.

2.2. 이름 혹은 휴대폰 번호를 입력하지 않은 경우 입력하라는 경고창이 띄워진다. 입력한 정보에 해당하는 메일이 없는 경우 ‘입력하신 정보에 해당하는 메일을 찾을 수 없습니다’ 라는 경고창을 띄운다.

**3. 비밀번호 찾기**

3.1 메일을 입력후 비밀번호 재설정 버튼을 클릭하면 입력한 메일에 비밀번호를 재설정할 수 있는 메일이 발송된다.

3.2. 메일을 입력하지 않고 비밀번호 재설정 버튼을 클릭하면 ‘메일을 입력하세요’ 라는 경고창을 띄워준다.

3.3. 입력한 메일이 일치하지 않는 경우 ‘입력하신 정보에 해당하는 메일을 찾을 수 없습니다’ 라는 경고창을 띄운다.

3.3. 존재하는 메일을 입력하고 비밀번호 재설정을 누르면 해당 이메일로 비밀번호 재설정이 가능한 메일이 발송된다. 해당 사용자는 메일로 로그인해서 비밀번호를 재설정하면 된다.

**4. 메인**

4.1. 메인 페이지는 헤더, 사이드, 메인 화면으로 구성된다.

4.2. 헤더는 로고, 계정 메뉴로 구성된다.

4.2.1. 계정 메뉴에서 현재 로그인한 회원의 사진, 이름, 메일을 확인할 수 있다. 계정 메뉴를 클릭해 출근/퇴근, 마이페이지, 로그아웃 기능을 사용할 수 있다.

4.2.2. 출근/퇴근 메뉴는 로그인 후 출근 버튼을 누르지 않은 경우 출근으로 나타난다. 출근 버튼 클릭시 ‘출근 하시겠습니까?’라는 문구의 경고창이 뜬다. 확인 버튼 클릭시 출근 처리되며 출근/퇴근 메뉴가 퇴근으로 바뀐다. 퇴근 버튼 클릭시 ‘퇴근 하시겠습니까?’라는 문구의 경고창이 뜬다. 확인 버튼 클릭시 퇴근 처리 되고 당일에는 출근/퇴근 메뉴를 더 이상 이용할 수 없다.

4.2.3. 마이페이지 메뉴 클릭시 사원 정보를 수정할 수 있는 페이지로 이동한다.

4.2.4. 로그아웃 메뉴 클릭시 로그아웃 처리 되고 로그인 페이지로 이동한다.

4.3. 사이드는 메뉴와 현재 로그인한 회원이 참여중인 프로젝트 목록으로 구성된다. 메뉴는 메인, 공지사항, 근무, 비용, 프로젝트, 채팅으로 나뉜다. 참여중인 프로젝트의 제목 클릭시 해당하는 프로젝트 페이지로 이동한다.

4.4. 메인 화면은 나의 일정, 마감이 임박한 업무, 출근한 팀원, 공지사항, 참여중인 프로젝트로 구성된다.

4.4.1. 나의 일정에서 현재 공지사항에 등록된 회사 일정, 참여중인 프로젝트 일정, 나의 업무 일정을 캘린더 형태로 확인할 수 있다.

4.4.2. 마감이 임박한 업무에서 마감일까지 5일이 남은 업무 리스트를 확인할 수 있다.

4.4.3. 출근한 팀원에서 현재 출근 상태인 팀원들의 리스트를 확인할 수 있다.

4.4.4. 공지사항에서 가장 최신 공지사항 글 5개의 글번호, 제목, 작성일, 조회수를 확인할 수 있다. 제목 클릭시 해당 공지사항 글 상세보기 페이지로 이동한다.

4.4.5. 참여중인 프로젝트에서 현재 참여중인 프로젝트의 프로젝트 이름, 진행 상태, 진행도를 확인할 수 있다.

**5. 마이페이지**

5.1. 마이페이지는 회원이 자신의 정보를 확인 및 수정할 수 있는 페이지이다.

5.2. 회원은 비밀번호와 휴대폰 번호를 수정할 수 있다.

5.3. 비밀번호를 입력하지 않거나 비밀번호가 정규표현식에 맞지 않는 경우 ‘비밀번호가 형식에 맞지 않습니다. 다시 입력해주세요.’라는 경고창을 띄운다.

5.4. 휴대폰 번호가 정규표현식에 맞지 않는 경우 ‘휴대폰 번호가 형식에 맞지 않습니다. 다시 입력해주세요.’라는 경고창을 띄운다.

5.5. 수정 버튼 클릭시 회원 정보가 수정된다.

**6 공지사항**

6.1. 공지사항은 회사의 공지사항을 작성하고 일정을 등록할 수 있는 게시판 형태의 페이지이다.

6.2. 일반 회원은 공지사항 글 열람 권한을 가지며 PM 회원은 공지사항 글 열람, 글쓰기, 수정 및 삭제 권한을 가진다.

6.3. 공지사항 메인 화면에서 공지사항 글번호, 제목, 작성일, 조회수를 확인할 수 있다. 기본적으로 한 페이지당 10건의 공지사항 글을 확인할 수 있으며 옵션 선택을 통해 페이지당 10, 25, 50, 100 개씩의 글을 확인할 수 있다. 오른쪽 상단의 검색바를 통해 입력한 값에 해당하는 글을 검색할 수 있다.

6.4. PM 회원의 경우 메인 화면의 하단에 글쓰기 버튼이 활성화 되어 있다.

6.5. 제목 클릭시 글 상세보기 페이지로 이동한다. 글 상세보기 페이지에서 글의 제목, 공지사항 일정, 첨부파일, 내용을 확인할 수 있다. 첨부파일이 있는 경우 해당 첨부 파일의 파일명 클릭시 파일을 다운로드 받을 수 있다. 하단의 목록 버튼 클릭시 메인 화면으로 이동한다. PM 회원의 경우 하단에 수정과 삭제 버튼이 활성화 되어 있다.

6.6. PM 회원은 글쓰기 버튼을 클릭해 글을 작성할 수 있다. 글 작성시 제목, 일정 내용, 일정 시작일과 종료일, 첨부파일, 내용을 작성할 수 있다.

6.7. 제목과 내용을 입력하지 않고 작성 버튼 클릭시 ‘제목 또는 내용을 입력해주세요’라는 문구의 경고창을 띄워준다. 일정 내용을 입력하지 않고 일정 시작일과 종료일을 입력한 경우 ‘일정 내용을 입력해주세요’라는 문구의 경고창을 띄워준다. 일정 시작일과 종료일 중 한 곳을 입력하지 않고 작성 버튼 클릭시 ‘일정을 입력해주세요’라는 문구의 경고창을 띄워준다.

6.8. 글 수정시 제목, 일정 내용, 일정 시작일과 종료일, 첨부파일, 내용을 수정할 수 있다. 글 작성시와 동일하게 유효성 검사를 진행한다.

6.9. 글 삭제 버튼 클릭시 ‘글을 삭제하시겠습니까?’라는 문구의 경고창을 띄워준다. ‘OK’ 버튼 클릭시 글이 삭제된다.

**7. 근무**

7.1. 근무는 회원이 근무와 관련된 일정을 신청하고 관리할 수 있는 페이지이다.

7.2. 근무 메뉴에서 일반 회원이 사용할 수 있는 메뉴는 부재 일정 신청, 연장 근무 신청, 부재 일정 관리, 근무 내역 확인으로 구성된다. PM 회원이 사용할 수 있는 메뉴는 일반 회원이 사용할 수 있는 메뉴에 부재 신청 관리, 연장 근무 신청 관리 메뉴가 추가된다.

7.3. 부재 일정 신청 페이지는 부재 일정을 신청할 수 있는 페이지이며 캘린더와 부재 일정 신청 섹션으로 구성된다.

7.3.1. 캘린더에서 회원은 자신이 신청한 부재 일정을 캘린더 형태로 확인할 수 있다.

7.3.2. 부재 일정 신청 섹션에서 회원은 시작 시간, 종료 시간, 부재 항목, 사유, 결재자를 입력하고 부재 일정을 신청할 수 있다. 부재 항목에서 ‘연차’, ‘반일연차’, ‘출장’, ‘외근’, ‘경조휴가’ 중 하나를 선택할 수 있다. 연차 선택시 하단에 프로그레스바가 활성화되어 연차 사용 일수를 선택할 수 있다. 반일 연차 선택시 연차 사용 일수가 자동으로 0.5일로 설정된다. 결재자 항목에서 신청하는 회원이 속한 팀의 PM 회원을 선택할 수 있다.

7.3.3. 특정 항목을 입력하지 않고 확인 버튼을 클릭한 경우 ‘해당 항목을 입력해주세요’라는 문구의 경고창을 띄워준다.

7.3.4. 초기화 버튼 클릭시 회원이 입력했던 값이 모두 초기화된다.

7.4. 연장 근무 신청 페이지는 연장 근무를 신청할 수 있는 페이지이다. 캘린더와 연장 근무 신청 섹션으로 구성된다.

7.4.1. 캘린더에서 회원은 자신이 신청한 연장 근무 내역을 캘린더 형태로 확인할 수 있다.

7.4.2. 연장 근무 신청 섹션에서 회원은 시작 시간, 종료 시간, 사유, 결재자를 입력하고 연장 근무를 신청할 수 있다. 결재자 항목에서 신청하는 회원이 속한 팀의 PM 회원을 선택할 수 있다.

7.4.3. 특정 항목을 입력하지 않고 확인 버튼을 클릭한 경우 ‘해당 항목을 입력해주세요’라는 문구의 경고창을 띄워준다.

7.5. 부재 일정 관리 페이지는 회원이 자신의 부재 일정을 관리할 수 있는 페이지이며 연차 현황과 부재 신청 현황 섹션으로 구성된다.

7.5.1. 연차 현황에서 남은 연차와 사용 연차를 확인할 수 있다.

7.5.2. 부재 신청 현황에서 부재 항목, 기간, 연차 사용 일수, 신청 일자, 승인 여부를 테이블 형태로 확인할 수 있다. 기본적으로 한 페이지당 10건의 부재 신청 리스트를 확인할 수 있으며 옵션 선택을 통해 페이지당 10, 25, 50, 100 개씩의 리스트를 확인할 수 있다. 오른쪽 상단의 검색바를 통해 입력한 값에 해당하는 리스트를 검색할 수 있다.

7.5.3. 미승인된 부재 신청의 경우 아이콘 클릭시 아이콘 클릭시 해당 부재 신청을 수정 및 삭제할 수 있다.

7.6. 근무 현황 페이지는 회원이 자신의 근무 현황을 확인 및 관리할 수 있는 페이지이다. 근무 현황 차트와 연장 근무 신청 현황 섹션으로 구성된다.

7.6.1. 근무 현황 차트에서 해당 연도와 월에 해당하는 근무 현황을 차트로 확인할 수 있다. 사용자가 선택한 월의 일에 따라 근무 시간과 연장 근무 시간을 차트로 확인할 수 있다. 차트 하단에는 사용자가 선택한 월의 총 근무 시간, 정상 근무 시간, 연장 근무 시간이 자동으로 계산되어 표시된다..

7.6.2. 연장 근무 신청 현황에서 신청 번호, 기간, 신청 일자, 승인 여부를 테이블 형태로 확인할 수 있다. 기본적으로 한 페이지당 10건의 연장 근무 신청 리스트를 확인할 수 있으며 옵션 선택을 통해 페이지당 10, 25, 50, 100 개씩의 리스트를 확인할 수 있다. 오른쪽 상단의 검색바를 통해 입력한 값에 해당하는 리스트를 검색할 수 있다.

7.7. 부재 신청 관리 페이지는 PM 회원이 해당 팀의 일반 회원들이 신청한 부재 신청을 확인 및 승인하는 페이지로, PM 회원만 접근 가능하다. 부재 신청 목록으로 구성된다.

7.7.1. 부재 신청 목록에서 신청 번호, 사원 메일, 사원 이름, 부재 항목, 연차 사용, 기간, 신청 일자, 승인 여부를 테이블 형태로 확인할 수 있다. 기본적으로 한 페이지당 10건의 부재 신청 리스트를 확인할 수 있으며 옵션 선택을 통해 페이지당 10, 25, 50, 100 개씩의 리스트를 확인할 수 있다. 오른쪽 상단의 검색바를 통해 입력한 값에 해당하는 리스트를 검색할 수 있다.

7.7.2. 승인 여부 클릭시 PM 회원이 처리할 수 있는 모달창을 띄워준다. 모달창에서 PM 회원은 부재 신청 사유를 확인하고 승인 여부, 부재 신청 반려 사유를 입력할 수 있다. 승인 여부는 승인, 미승인, 반려로 구성된다.

7.7.3. 승인 여부를 선택하지 않고 확인 버튼 클릭시 미승인으로 처리된다.

7.8. 연장 근무 신청 관리 페이지는 PM 회원이 해당 팀의 일반 회원들이 신청한 연장 근무 신청을 확인 및 승인하는 페이지로, PM 회원만 접근 가능하다.

7.8.1. 연장 근무 신청 목록에서 신청 번호, 사원 메일, 사원 이름, 신청 일자, 기간, 승인 여부를 테이블 형태로 확인할 수 있다. 기본적으로 한 페이지당 10건의 연장 근무 신청 리스트를 확인할 수 있으며 옵션 선택을 통해 페이지당 10, 25, 50, 100 개씩의 리스트를 확인할 수 있다. 오른쪽 상단의 검색바를 통해 입력한 값에 해당하는 리스트를 검색할 수 있다.

7.8.2. 승인 여부 클릭시 PM 회원이 처리할 수 있는 모달창을 띄워준다. 모달창에서 PM 회원은 연장 근무 신청 사유를 확인하고 승인 여부, 연장 근무 반려 사유를 입력할 수 있다. 승인 여부는 승인, 미승인, 반려로 구성된다.

7.8.3. 승인 여부를 선택하지 않고 확인 버튼 클릭시 미승인으로 처리된다.

**8. 비용**

8.1. 비용은 회원이 비용과 관련된 업무를 신청하고 관리할 수 있는 페이지이다.

8.2. 비용 메뉴는 정산 신청, 비용 현황, 비용 차트로 구성된다.

8.3. 정산 신청 페이지는 회원이 업무 관련 지출 내역을 신청할 수 있는 페이지이다.

8.3.1. 회원은 법인 카드, 비용 항목, 사용일, 사용처, 사용 금액, 상세 내용을 입력해 정산을 신청할 수 있다.

8.3.2. 영수증 등록 버튼 클릭시 영수증을 사진 형태로 첨부할 수 있다. 구글 VISION API를 통해 자동으로 사용일, 사용처, 사용 금액이 자동으로 입력된다.

**9. 프로젝트**

9.1. 프로젝트 메인에 가면 현재 자신이 참여하고 있는 프로젝트가 보이고 해당 프로젝트를 선택하여 해당 프로젝트의 상세 업무 화면으로 갈 수 있다.

9.2. 진행 중인 프로젝트와 이미 완료된 프로젝트 현황에 대해 알 수 있다.

9.3. 프로젝트 상세 페이지에 들어가면 칸반보드 / 내업무 / 현황 / 구글 드라이브 메뉴가 있다.

9.4. 칸반보드에는 해당 프로젝트의 전체 업무를 진행 상태에 따라 한 눈에 볼 수 있다.

9.5. 업무상태는 예정 / 진행 / 테스트 / 완료 별로 업무를 확인 할 수 있다.

9.6. 자기에 할당된 업무가 끝나면 업무상태를 수정할 수 있으며, 이에 따라 프로젝트참여자들이 언제들지 어떤 업무가 현재 어떤 상태인지 신속히 확인 할 수 있다.

9.6. 내 업무에서는 나에게 할당된 업무만을 편하게 볼 수 있으며, 마감이 임박한 업무나 계획한 완료 시간이 지난 업무에 대해서 확인 할 수 있다.

9.7. 업무를 클릭하게되면 속성 / 상세업무 / 체크리스트를 확인 할 수 있고, 속성탭에서는 업무에 대한 기한과 중요도 진행상황을 수정 및 확인 할 수 있다.

9.8. 현황 탭을 가게되면 차트 형태로 전체 프로젝트 진행률 / 전체 프로젝트 업무 비중 / 전체 프로젝트 담당자별 업무 진행률 / 나의 업무 진행 현황 / 중요도별 나의 업무 비중을 확인 할 수 있다.

9.9. 새로운 프로젝트 생성은 오직 ‘팀장' 회원에게만 권한이 있으며, 팀장은 새로운 프로젝트를 만들고, 그에 필요한 작업들을 업무단위로 쪼개 칸반보드의 예정칸에 등록 할 수 있다.

9.10. 업무를 할당 받은 사원은 해당 업무가 얼마나 걸릴 지 계획하여 기한시간을 정할 수 있다. 어떤 업무에 대한 기한시간은 그 업무를 할당 받은 사원만이 할 수 있다.

9.11. 팀장은 프로젝트 생성과 업무 분담에 역할이 집중되어 있고, 사원은 자신이 할당 받은 업무에 대한 기한과 변동에 따른 수정 및 업무 상태 변경으로 업무 진행도 모두에게 공유해서 프로젝트에 참여하고있는 인원이라면 프로젝트가 현재 어떤 상태인지 쉽게 파악하고 변동점이 있을때마다 유연하게 대처할 수 있다.

**10. 채팅**

1.1. 언제나 필요 할 때 커뮤니케이션이 가능 하도록 채팅 기능을 구현하였다.

1.2. 채팅은 그룹채팅과 1:1개인 채팅이 있다.

1.3. 그룹채팅방은 프로젝트가 생성될 때 프로젝트에 합류하고 있는 사람들을 초대해 프로젝트 진행중에 변동사항이 있을 경우, 신속히 알려준다.

1.4. 자신이나 자신이 포함된 그룹채팅방에 말이 올라오면 알림을 받을 수 있다.

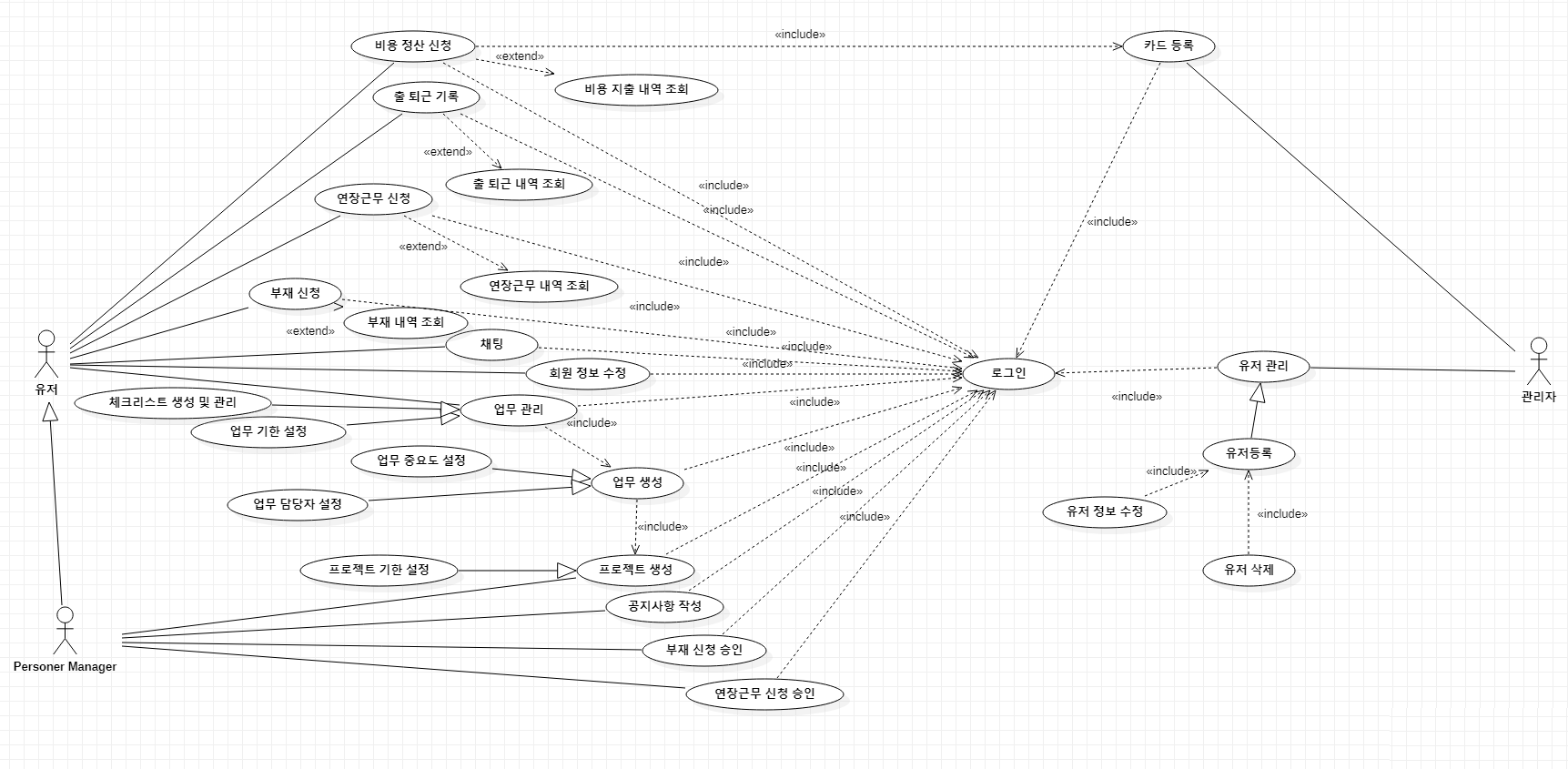
1. **요구사항 정의서**

별도 자료 첨부 : 가-1

1. **요구사항 명세서**

별도 자료 첨부 : 가-2

1. **USE CASE**

****

1. **코드규칙**

별도 자료 첨부 : 가-3

**구현**

1. **스토리보드**

별도 자료 첨부 : 나-1

1. **ERD**

****

1. **프로젝트 구조**

DOBEE 프로젝트는 Spring framework 와 Node.js 로 구성되어 있습니다. Spring의 경우 MVC 패턴을 기반으로 구성되어 있습니다. 사용자 인증 처리는 Spring Security를 통해 진행합니다. 컨트롤러는 동기와 비동기로 나누어 비동기의 경우 Rest Controller Annotation을 통해 json 객체를 주고 받을 수 있도록 했습니다. Mybatis를 통해 자바 객체를 맵핑해 db와 연결했습니다.

데이터 베이스는 오픈 소스의 관계형 데이터 베이스인 MySQL을 사용했습니다. 관계형 데이터베이스는 데이터 중복을 막고 데이터의 정확성을 확보할 수 있다는 장점을 갖고 있습니다.

채팅 서비스의 경우Node.js를 활용했습니다. Node.js는 이벤트 처리 방식의 비동기 프로그래밍을 기반으로 하기 때문에 클라이언트의 요청을 빠르게 처리할 수 있다는 장점을 갖고 있습니다. 이러한 장점을 살려 채팅 서비스의 Back-end 부분을 Node.js로 구현하였습니다.

1. **기능별 사용 기술**

1. **환경설정**

|  |
| --- |
| pom.xml 설정 |
|  |
| - properties setting version  ➀ java-version은 1.8 ➁ spring-framework-version 4.3.14  - ‘spring-context’는 context 정보를 제공하는 설정 파일이다. spring-context에는 JNDI, EJB, e-mail ,internalization, validation, scheduling, functionality 서비스가 포함되어 있다.  - ‘spring-webmvc’는 Web 어플리케이션을 구축하기 위한 완전한 기능을 갖춘 MVC 구조를 구현하고 있다. MVC Framework는 전략 인터페이스를 통해 고급 설정이 가능하며 JSP, Velocity, Tiles, iText 및 POI 등 다양한 view 기술을 지원하고 있다.  - 'spring-jdbc'는 JDBC의 장점과 단순성을 그대로 유지하면서도 기존 JDBC의 단점을 극복할 수 있게 해주고, 간결한 형태의 API 사용법을 제공하며, JDBC API에서 지원되지 않는 편리한 기능을 제공한다.  - 'servlet'은 자바를 사용하여 웹페이지를 동적으로 생성하는 서버측 프로그램이다.  - 'jstl'은 jsp에서 자바코드 대신에 사용할 수 있는 일반적 기능을 모아둔 태그를 제공한다. |

|  |
| --- |
| web.xml |
|  |
| - ContextLoaderListener는 Controller가 공유하는 Bean을 포함하는 Spring Container를 생산한다.  - applicationContext.xml은 DataSource, myBatis 등에 관련한 bean객체의 설정을 생성하고, security-context.xml은 Security에 관한 bean객체의 설정을 하고 있다. 여기서 설정된 bean객체는 DispathcerServlet에서 참조할 수 있다.(https://gmlwjd9405.github.io/2018/10/29/web-application-structure.html)  - DispatcherServlet은 Spring에서 제공하는 클래스로 클라이언트의 요청을 가장 먼저 받는다. mapping을 통하여 \*.do 패턴의 url을 dispathcer-servlet으로 보내도록 설정하였다.  - DelegatingFilterProxy는 스프링 시큐리티가 모든 애플리케이션 요청을 감싸게 해서 모든 요청에 보안이 적용되게 하는 서블릿필터이다. 스프링 프레임워크 기반의 웹 애플리케이션에서 서블릿 필터 라이프 사이클과 연계해 스프링 빈 의존성을 서블릿 필터에 바인딩하는데 사용한다.  - encodingFilter를 사용하여 모든 요청에 대하여 UTF-8 방식으로 인코딩 하도록 하였다. UTF-8 방식은 모든 유니코드 문자를 표현할 수 있기때문에 한글 처리를 위하여 UTF-8 방식을 선택하였다.  - error-page 404, 500 에러시 설정한 에러페이지가 보여지도록 설정하였다. |

|  |
| --- |
| dispatcher-servlet.xml |
|  |
| - dispatcher-servlet.xml은 Http 프로토콜을 통해 들어오는 모든 요청을 처리하는 Front Controller이다. Front Controllerㅣ는 주로 서블릿 컨테이너의 제일 앞에서 서버로 들어오는 클라이언트의 모든 요청을 받아서 처리해주는 Controller로 MVC 구조에서 함께 사용되는 패턴이다.    https://mangkyu.tistory.com/18  - mvc annotation dirven은 MVC 패턴을 위한 기본적인 bean 객체 설정을 해준다.  - context:annotation-config는 이미 등록된 bean객체에 대해 Annotaion을 활성화한다. scan은 config와 달리 xml에 bean 설정을 하여야 한다.  - context:annotation-scan은 bean 등록 여부와 관계없이 Spring이 bean 스캔을 활성화 한다.  xml에서 bean설정을 할 필요없이 base-package를 설정하여 해당 package안을 스캔하여 자동으로 bean 설정을 한다.  - viewResolver  - jsonView는 MappingJackson2JsonView를 사용하여 ModelAndView를 json 형식으로 반환해준다.  -multipartResolver는 Spring의 multipart 지원 기능을 이용하기 위한 설정이다. multipartResolver는 multipart객체를 컨트롤러에 전달하는 역할을 한다. 최대 파일크기를 50mb로 설정하였다. 설정을 하지 않을시 기본 값은 -1로 설정되며 이는 제한없음을 의미한다.  - transactionManager는 Transaction처리를 위한 설정이다. Transaction은 데이터베이스의 상태를 변환시키는 하나의 논리적 기능을 수행하기 위한 작업의 단위 또는 한꺼번에 모두 수행되어야 할 일련의 연산들을 의미한다. Transaction을 적용하기 원하는 부분에 @Transactional 어노테이션을 추가하여 데이터들이 일관성 있게 입력, 수정, 삭제 되도록 한다.  - websocketHandler는 Spring Websocket 구현을 위한 설정이다. Spring Websocket은 Spring 4.0부터 지원하는 기술이다. /alram.do 패턴의 요청이 있을 경우 설정한 Handler객체로 요청을 전송하여 처리하도록 한다.  - errorpage 는 404,500 에러가 발생할 시 해당 페이지로 이동하도록 설정하였다. |

|  |
| --- |
| application-context.xml |
|  |
| - application-context에서 설정한 bean객체는 하위  - DB 연결 자바에서 DB를 사용하기 위해서는 JDBC를 사용하였다. DB는 MySql을 사용하였다.  url은 AWS에 설치한 MySql 경로를 설정하였다.  - bCryptPasswordEncoder는 security 암호화를 위하여 설정하였다.  - Spring에서는 SplSessionFactoryBean를 이용하여 Mybatis를 사용한다.  - SplSessionFactoryBean는 JDBC DataSource가 필요하고 우리가 위에서 만든 JDBC를 적어준다. mapperLocations 프로퍼티는 Mybatis의 XML Mapper 파일들의 위치를 지정하기 위해서 사용한다.  - SqlSessionTemplate은 SqlSession을 구현하고 코드에서 SqlSession을 대체하는 역할을 한다. SqlSessionTemplate은 쓰레드에 안전하고 여러 개의 DAO나 Mapper에서 공유할 수 있다.  - velocity는 java기반의 템플릿 엔진으로 웹 페이지로부터 자바 코드를 분리할 수 있고, 웹사이트를 계속 오랫동안 유지할 수 있으며, 자바 서버 페이지(JSP)의 실용적인 대안을 제공한다. resourceLoadPath를 통하여 vm파일이 저장된 경로를 이용하여 자원을 활용할 수 있도록 설정하였으며, input,output filter를 UTF-8로 하여 한글문자의 입출력이 가능하도록 하였다.  -java mailsender 는 spring이 지원하는 메일 발송방법으로 아이디와 비밀번호 인증을 통하여 메일을 전송할 수 있도록 설정하였다. |

|  |
| --- |
| package.json 설정 |
|  |
| -Node.js는 Express Framework를 기반으로 위와 같은 모듈을 설치했다. |

▶node.js 활용한 채팅 기능

|  |
| --- |
| app.js 설정 |
|  |
| - Node.js는 기본적으로 Express Framework를 사용해 구성되어 있다.  - Express 서버와 웹 소켓을 연결한다. |

|  |
| --- |
| database.js 설정 |
|  |
| - DB 기본 셋팅과 DB 연결 함수를 exports하는 모듈이다.  - MySQL 모듈을 사용하고 Connection Pool을 활용한 방식으로 데이터 베이스에 연결한다.  - dbConfig 변수에 데이터 베이스에 연결하는 기본 설정과 최대 커넥션 개수(connectionLimit)가 설정되어 있다.  - 데이터 베이스 연결 함수를 exports 해서 다른 라우터에서 데이터 베이스에 쉽게 연결할 수 있도록 한다. |

|  |
| --- |
| socket.js 설정 |
|  |
| - 웹소켓 통신을 담당하는 socket.io 라이브러리를 모듈로 사용한다. socket.io는 Node.js를 기반으로 웹소켓 통신이 가능하도록 하는 양방향 통신 라이브러리이다.  - database 모듈을 사용해 데이터 베이스와 연결하고 채팅 대화 내용을 저장한다. |

|  |
| --- |
| 네임스페이스 활용 |
| <socket.js>    <view단> |
| - socket.io의 네임스페이스를 활용해 나에게 메시지 보내기, 그룹 채팅, 1:1 채팅으로 각 채팅방을 나누고 각각의 네임스페이스에 소켓이 연결될 수 있도록 설정한다.  - 네임스페이스는 네임, 즉 이름이 같은 곳에 속해 있는 소켓끼리만 통신하는 개념으로 채팅방 종류에 따라 연결되는 네임스페이스가 다르도록 url path를 다르게 지정한다. |

|  |
| --- |
| Promise 패턴을 활용한 함수 호출 |
| <view - chatMain.jsp>  -view 페이지에서 이벤트로 서버에 요청    <socket.js>  -이벤트 발생시 클라이언트에 응답    -기존 채팅방의 유무 확인하는 함수(Promise 패턴)    <구현 화면>  -1:1 채팅 화면 |
| - self 네임스페이스에 소켓이 연결된후 프론트 페이지에서 ‘send message to self’ 이벤트를 서버에 요청하면 기존에 채팅방이 있는지 확인하는 함수가 실행된다. Node.js는 비동기 프로그래밍을 기반으로 하기 때문에 동기성을 보장하기 위해서는 call back 함수를 활용해야한다. 이 때 Promise 패턴을 적용해 코드의 가독성을 높이고자 했다.  - 기존에 채팅방이 존재하는 지 확인하는 함수는 Promise 패턴으로 구성되어 있다. Promise 함수의 결과에 따라 결과가 “no”이면 새롭게 채팅방을 만든다. 이 때 채팅방을 만드는 함수 Promise 패턴으로 구성되어 함수 호출 결과가 “yes”면 (채팅방이 성공적으로 만들어진 경우) 채팅 내용을 저장하는 함수를 호출한다. |

▶web socket 이용한 알람 구현

|  |
| --- |
| pom.xml |
|  |
| - spring websocket은 spring4.0부터 지원하는 서비스로 자주, 많은 양의, 지연이 짧아야 하는 통신을 해야할 때 적합하다. websocket의 사용을 위해서 의존성을 설정해준다. |

|  |
| --- |
| dispatcher-servlet.xml |
|  |
| - Front Controller에서 알람요청에 대해서는 다른 Controller로 보내지 않고 AlarmHandler로 보내도록 처리하였다. |

|  |
| --- |
| AlarmHandler.java |
|  |
| - AlarmHandler는 알람 요청을 담당하는 Controller의 역할을 하며 WebSocket의 접속, 데이터전송, 해제 요청을 처리한다. 데이터 전송부분에서는 요청정보를 JSON String으로 받아 여러 요청에 대하여 구분하여 처리할 수 있도록 하였다. |

|  |
| --- |
| view단 |
|  |
| - page가 load될 때 마다 websocket에 접속이 이루어지도록하였다. 모든 페이지에 공통으로 사용되는 top부분에 코드를 작성하여 로그인이 되어있을 시 알람을 송신, 수신할 수 있다. |

▶velocity 메일 발송 기능-> 욱재

|  |
| --- |
| pom.xml 설정 |
|  |
| - 메일 전송을 위하여 추가한 의존성은 메일을 보내기 위한 java mailsender와 메일 전송시 템플릿으로 사용한 velocity이다.  - velocity는 spring에서 제공하는 템플릿 엔진으로 템플릿 파일을 비교적 쉽게 작성 가능하다.  - 자바 객체를 템플릿 파일에서 조작할 수 있기 때문에 뷰를 생성하는데 JSP 만큼 널리 사용된다. |

▶ MySQL Trigger

|  |
| --- |
| MySQL Trigger 설정 |
| USER Table    APPLY Table    BREAK Table |
| .- 특정 테이블에 데이터가 입력되거나 수정, 삭제가 이뤄질 경우 추가적인 작업을 Database 내에서 Trigger를 통해 실행한다.  .- 코드 작업이 DB에서도 추가되기 때문에 유지보수가 다소 복잡해지지만 위와 같은 기능을 구현하기 위한 Back단의 코드가 줄어든다.  .- DB 테이블에 입력되는 데이터에 따라 다른 트리거가 작동하도록 구현하였다. |

▶ MySQL Event Scheduler

|  |
| --- |
| MySQL Event Scheduler설정 |
|  |
| .- 사용자가 입력한 쿼리나 프로시저, 함수 등을 사용자가 지정한 시간에 작동하며  SET GLOBAL event\_scheduler = ON; 구문을 입력하여 이벤트 스케줄러가 동작하도록 설정한다.  .- JAVA의 Quartz로 구현 가능하지만 보다 편하고 웹 서버를 거칠 필요 없이 DB서버에서 처리하여 효율적인 리소스 관리가 가능하다.  .- 코드 작업이 DB에서도 추가되기 때문에 유지보수가 다소 복잡해지지만 Back단에서 위와 같은 처리를 위한 코드가 줄어든다.  - 일반적으로 사용자나 트래픽이 적은 시간대를 선택하여 스케줄러가 동작하도록 설정한다. |

▶ AWS

|  |
| --- |
| AWS EC2 Tomcat – tomcat-users.xml |
|  |
| - 사전작업으로 AWS EC2 Linux 인스턴스 생성과 Java JDK, Tomcat이 설치 되어야 한다. DOBEE 는 Node 를 사용하였기때문에 NodeJS 또한 설치한다.  - Tomcat 서버 운영 시 웹 서버의 관리자 계정을 만들어진 EC2 Linux Tomcat에서 설정한다.  - 여기서 설정한 계정은 Maven으로 배포 시 사용된다. |

|  |
| --- |
| AWS EC2 Tomcat – server.xml |
|  |
| - 파일 업로드 시 파일이 저장될 위치를 설정한다.  - Tomcat 으로 배포된 프로젝트 경로의 upload 폴더에 업로드 된 파일이 저장된다. |

|  |
| --- |
| Application-context.xml |
|  |
| - AWS RDS에서 설정 한 MySQL 데이터베이스를 연동한다. Public IP 와 username, password 를 입력한다.  - 기본적으로 AWS 에서는 외부 접근을 제한하기 때문에 AWS 관리 콘솔에서 3306 포트 개방과 Public Access 를 가능하도록 변경한다. |

|  |
| --- |
| pom.xml |
|  |
| - 프로젝트를 Maven build를 통해 특정 주소에 배포하기 위한 설정이다.  - url에는 AWS Public IP:포트번호, username과 password는 위의 tomcat-users.xml 에서 설정한 계정을 입력한다.  - 배포 시 Maven build 에서 배포할 프로젝트를 선택하고 goals에 tomcat:deploy를 입력하고 Run 을 클릭하면 배포가 시작된다. |

▶구글 vision api

|  |
| --- |
| pom.xml 설정 |
|  |
| -google vision api를 사용하기 위한 라이브러리이다.  -spring 에서 google API 쓰기 위해 먼저 gcp 라이브러리 넣어주고 다음 비전을 추가한다. |

|  |
| --- |
| 기술 구현 |
| <컨트롤단>    <서비스단> |
| 1. 기술 사용 이유   영수증만으로 간단하고 빠르게 비용신청을 하고자 하는게 기획의도였기 때문에, 구글 비전으로 이용하여 신속하게 비용신청을 하고싶었다.   1. 기술 사용 설명   ① 컨트롤단에서 요청을 받고 구글 서비스단에 이미지가 올라간 경로를 보내줍니다.  ② 서비스단에서 이미지가 담겨있는 실경로를 타고 들어가서 이미지를 구글 비전 서비스로 보냅니다.  ③ 구글 스토리지를 통해 이미지가 전송되고 구글 비전이 스토리지 내에서 이미지에 접근하여 기능을 수행합니다.  ④ 추출된 TEXT를 다시 요청 브라우저로 리다이렉트 해주고 웹에서는 그 데이터를 받아 개발자가 자기 웹 서비스에 알맞게 가공하여 사용합니다. |
|  |

▶구글 drive api

|  |
| --- |
| pom.xml 설정 |
|  |
| -google Drive 역시 필요한 라이브러리를 설정해줘야한다.  -Vision 과는 다르게 OAuth 2.0 인증과 관련한 라이브러리도 필요하다. |

|  |
| --- |
| 기술 구현 |
| <화면단>    <자바스크립트 코드> |
| 1. 기술 사용 이유   같이 프로젝트를 진행하다 보면 필연적으로 공유해야 할 파일이 필요하고, 이것을 효율적으로 관리하기 위해 프로젝트 마다, 구글드라이브를 연동한 자료실이 있다.  구글 드라이브로 연동하는게 그 무엇보다 파일의 안정성과 편의성을 준다고 판단되어 구글드라이브를 선택했다.  구글 아이디로 구글드라이브 인증을 하면 구글드라이브를 이용해 원하는 자료를 올릴 수 있다.  파일을 업로드하면 타임라인 형태로 보여주고, 업로드파일이 많아지면 검색기능을 이용해 적절하게 필요한 파일을 바로바로 찾을 수 있다.  2) 기술 사용 설명  ① 구글 드라이브를 쉽게 사용하기 위해선 구글에서 지원해주는 Picker 플로그인을 사용하는게 좋다.  ② 위에 있는 코드는 Picker 코드중 핵심 일부분이고, Picker객체를 생성하여 옵션을 사용가자 자신의 웹 서비스에 알맞게 어느정도 수정을 할 수 있다. DOBEE는 큰 수정없이 사이즈 조절이랑 타이틀정도만 수정하고 나머지는 티폴트값을 사용하였다.  ③ 구글 드라이브는 사용자 인증 방식으로 반드시 AOuth 2.0 방식을 사용해야 인증이 가능하다.  ④ 구글 드라이브 서비스 특성상 제3자의 파일에 접근해야 하므로, AOuth 2.0 방식만 구글 측에서 인정을 해준다. |
|  |

1. **트러블 슈팅**

별도 자료 첨부 : 나-2