**별첨 5】최종보고서(학습근로자)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **최종 보고서**        **2019년 2학기**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **학 과** | **융합소프트웨어과** | | **학년-반** | **1-1** | | **학 번** | **201953204** | | **이 름** | **이근혁** | | **교과목** | **캡스톤디자인1(PBL)** | | | | | **점 수** |  | **교과목담당교수 : 류중경 교수님**  **이영걸 교수님** | | | |

**【별첨 5】최종보고서 (학습근로자)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **학번** | 201953204 | **성명** | 이근혁 | **기업명** | (주)아임클라우드 |   **1. 주차별 학습일지**   |  |  | | --- | --- | | **주차** | **내용** | | **1주차** | 프로젝트 수행 기간 동안 개발할 프로그램에 대해 기획하고 기획한 프로그램에 대한  개발 배경 및 개요, 개발 목표, 주요 기능에 대해 문서로 작성했습니다.  또한 최종 결과물에 대한 화면(UI)을 디자인하여 전체적인 화면 구성을 진행했으며 프로그램  개발을 위해 기술적인 타당성을 조사하고, 전반적인 아키텍처를 설계하여 문서로 작성했습니다.  최종적으로 기획한 프로젝트는 Noty라는 이름의 서비스이며, 할 일과 메모를 효율적으로  관리할 수 있도록 웹 기반으로 제공하는 서비스를 구상하게 되었습니다. | | **2주차** | NoSQL와 기반의 데이터베이스 설계 방식과 주로 사용하던 RDB와의 차이점, 그리고 설계 시 고려해야하는 부분에 대해서 학습했습니다. 데이터베이스 모델을 효과적으로 정의하고 배포 하도록 도와주는 Prisma라는 프레임워크에 대해 예제를 구현해가며 학습하였고, 서버/클라이언트 간의 인터페이스를 구현하기 위한 GraphQL에 대해서도 새로 학습했습니다.  또한 RESTFul API와 GraphQL의 차이점과 각각의 장단점을 알게 되었고 예제 실습을 진행하며 기본적인 gql 문법을 익혔습니다. NoSQL 기반의 데이터베이스 모델을 설계하고, 이를 기반으로 Prisma 프레임워크의 모델을 정의하는 내용을 문서로 작성했으며 GraphQL을 사용한 인터페이스 설계서 또한 작성했습니다. | | **3주차** | 프로젝트 개발 및 운영을 위한 서버를 고민하다가 AWS의 클라우드 컴퓨팅 서비스를  활용하기로 결정했습니다. AWS의 많은 서비스 중 EC2(Elastic Compute Cloud) 인스턴스를 생성하여 클라우드에 서버를 구축하고, 해당 서버에 필요한 도커와 기타 의존 프로그램을  설치하여 개발할 계획입니다. 클라우드 서비스를 사용하면 비용이 드는 문제가 있기 때문에  프리티어(무료)로 사용할 수 있는 사양을 선택하여 서버를 생성하게 되었고, SSH 접속을 위한 인증키를 발급받아 원격으로 접속할 수 있는 환경을 갖추었습니다. 또한 개발에 사용하게 될 프레임워크와 개발 도구를 정리하여 문서로 작성하였습니다 | | **4주차** | 구축한 개발환경에서 Node.js 기반의 서버 프로그램 구현을 위해 Node.js를 설치하였고, 개발에 필요한 Prisma와 Express등 의존 라이브러리를 설치하였습니다. 개발하기에 앞서 OJT 수업에서 배운 내용을 기반으로 전체적인 서버 프로그램의 프로젝트 구조를 구성한 후 개발을 시작 했습니다. 모델 및 인터페이스 설계 내용을 참고하여 Prisma 데이터 모델링을 진행했고, GraphQL 스키마를 작성했으며 회원가입과 기본적인 유저 데이터를 제어하는 GraphQL 리졸버를 서버 프로그램에 구현하였습니다. 회원가입 시 유저 데이터의 보안을 위해 비밀번호는 암호화한 후 저장할 수 있도록 bcrypt를 이용하여 비밀번호를 평문이 아닌 암호화된 해시로 저장하도록 구현했습니다. 또한 JWT 방식의 인증 및 로그인 기능을 구현하여 회원가입 – 로그인에 대한 기본 구성을 마무리 했습니다.  마지막으로 정의한 스키마와 리졸버를 통해 회원가입 및 로그인 기능을 테스트하기 위해  graphql-yoga에 내장된 GraphQL Playground를 활용하여 테스트를 진행했습니다 | | **5주차** | 지난주에 이어 서버 프로그램 구현을 진행했습니다. 지난주에 구현한 간단한 사용자 등록,  로그인에 대한 기능 외에 노트 작성, 수정, 삭제(태그, 할 일도 동일) 기능에 대한 리졸버를  추가적으로 구현 했습니다. 기존에 설계해두었던 GraphQL 스키마를 참고하여 개발하게 되니 큰 문제없이 계획적으로 잘 진행되었습니다. 데이터베이스 트랜잭션 부분은 어느 정도 개발이 마무리 되었으며, 서버 로깅을 위해 기능을 추가로 구현했습니다. | | **6주차** | 지난주에 GraphQL 리졸버 구현을 마무리하였고, 이번 주에는 푸시 알림을 위한 파이어베이스 연동을 진행했습니다. Push 동작 방식을 학습하였고 Pull 방식과의 차이점을 알게 되었습니다. 실제 앱 개발시에도 애플의 APNS, 구글의 FCM(구 GCM)의 푸시 서버 제공 서비스를 활용하여 개발한다는 것을 알게 되었습니다. 웹 기술의 발전으로 이제 네이티브 앱이 아닌 웹 브라우저에서도 푸시 알림을 받을 수 있게 되었으며 애플의 경우 현재 지원되지 않지만 앞으로 발전할 가능성이 있다고 보입니다. 구글의 경우 파이어베이스 프로젝트를 생성하면 서버 키가 주어지는데 이를 통해 FCM 서비스를 사용할 수 있습니다. 웹 푸시에서 VAPID라는 암호화 키를 요구하는데 키를 생성할 경우 비공개키와 공개키가 생성되며 클라이언트에서는 공개키를 통해 인증을 진행하도록 되어있다는 것을 학습하였습니다. 서버키를 통한 파이어베이스 연동과 더불어 서버의 알 수 없는 문제로 인한 프로세스 종료를 방지하기 위해 Node.js 실행 프로세스 전역에서 예외를 처리할 수 있도록 관련 로직도 구현하였습니다. | | **7주차** | 6주차까지 서버프로그램의 설계와 개발을 마무리한 후 본격적으로 클라이언트(웹 앱)개발을 시작했습니다. 먼저, 설계 단계에 정의한 필요 라이브러리를 준비하고 서버에 구현한 GraphQL API를 통해 인터페이스가 정상적으로 동작하는지 간단한 테스트를 진행했습니다. 웹 애플리케이션 개발을 위해 Vue.js 기반의 프로젝트를 새로 생성하였고 vue-cli 도구를 사용하여 모던 웹 개발에 사용하는 Webpack, Babel등의 번들러 & 트랜스파일러를 준비하였습니다. 또한 Vue.js의 에코 라이브러리로 웹 애플리케이션의 상태관리를 위한 Vuex, 라우팅 처리를 위한 Vue-router도 함께 포함하여 환경을 구성했습니다. 클라이언트와 서버의 인터페이스 테스트는 데이터베이스 설계 당시 정의한 GraphQL 스키마를 참고하여 어려움 없이 서버와 상호작용할 수 있었습니다. | | **8주차** | vue-cli를 통해 웹 애플리케이션 개발 환경을 구성하고 필요 라이브러리(Apollo 등)를  준비했습니다. 이번 주에는 개발 계획서에 맞는 화면 구성을 위해 라우팅 처리할 뷰를 생성하고, 라우터에 등록하는 작업을 진행했습니다. 싱글 파일 컴포넌(SFC)트 방식의 .vue 파일을 생성하여 뷰와 컴포넌트 요소들을 생성하여 개발할 수 있도록 프로젝트 구성을 마무리했습니다. 각 뷰에 해당하는 SFC 파일을 생성하였고, 쉽게 구분 및 분류하기 위해 views 디렉토리에 따로 저장하였습니다. 뷰 파일을 생성한 후 각각의 뷰를 vue-router에 등록하여 추후 웹  애플리케이션에서의 라우팅 처리를 쉽게 할 수 있도록 환경을 구성했습니다. | | **9주차** | 원활하고 구조적인 웹 개발을 위해 Noty의 공통 스타일을 파일로 모듈화하여 분리시키는 작업을 진행했습니다. 기존 웹 개발에 사용하던 CSS가 아닌 SCSS(SASS)를 사용하여 효율성이 크게 증가했습니다. 또한 SCSS의 변수, 맵, 믹스인, 스타일 상속 등에 대한 내용을 다시 한 번 조사하여 기초적인 내용을 다졌습니다. 변수에 Noty의 스타일 색상을 저장해두고 변수를 통해 같은 색상을 불러와서 사용하는 등 개발 편의성이 크게 증가했습니다. SCSS도 결국 컴파일 과정을 거쳐 일반 CSS로 변환되는데 어떤 방식으로 변환되는지 새로 학습했습니다. | | **10주차** | 지난주에 모듈화한 SCSS 모듈을 활용하여 로그인 및 회원가입, 공통 헤더 컴포넌트를 개발  하였습니다. 로그인(또는 회원가입) 기능은 사용자가 <input> 태그에 값을 입력하고 제출하는 방식으로 작동하는데 입력한 데이터(아이디, 비밀번호 등)를 Vue.js 데이터에 양방향 바인딩하여, 사용자가 입력하면 입력한 값이 Vue 데이터로, Vue 데이터를 수정하면 사용자에게 보이는 input의 값으로 설정되도록 구현하였습니다. Vue를 사용하여 개발하면서 MVVM (Model – View – View model) 아키텍처에 대해 학습하였습니다. 또한 Vue-router의 라우팅 이름값을 웹 앱 전역에서 감지할 수 있도록 Vue의 watch (값 변경 감지) 프로퍼티를 구현하여 로그인/회원가입 페이지인 경우 헤더를 숨김처리 할 수 있도록 구현하였습니다. 페이지 전환 없이 컴포넌트가 보이고, 숨겨지고, 페이지가 이동하게 되니 기존 웹 페이지가 아닌 앱을 사용하는 듯한 사용자 경험을 제공할 수 있었습니다. | | **11주차** | 기본적인 회원 기능에 대한 화면 UI(로그인, 회원가입)를 구현하여 서버와 연동하는 작업을 진행했습니다. 설계 단계에 진행한 GraphQL 스키마를 참고하여 클라이언트에서 형식에 맞춰 요청만 보내면 되기 때문에 비교적 수월하게 진행되었습니다. 사용자 로그인시 JWT을 웹 브라우저의localStorage에 저장해두고 인증이 필요한 API (GraphQL)를 요청할 때 요청 헤더에 토큰을 넣어 서버에게 전달하는 방식으로 구현하였습니다. 인증이 되지 않은 유저는 해당 API 데이터를 받을 수 없고, 인증된 유저는 정상적으로 데이터를 조회하고 조작할 수 있었습니다. | | **12주차** | 서버와 통신하여 할 일, 태그 정보를 불러오기에 앞서 UI 개발을 진행했습니다. Vue.js의 데이터 바인딩 기능을 적극적으로 활용하여 쉽고 빠르게 개발할 수 있었습니다. 지난번 로그인 및 회원가입 페이지 개발시 활용한 v-model 디렉티브는 양방향 데이터 바인딩을 위해 사용했지만, 이번에는 부모 -> 자식 컴포넌트간 데이터를 전달하고, 받은 데이터를 단방향으로 사용하기 위해 props와 v-bind 디렉티브를 활용했습니다. 단순한 데이터 바인딩 뿐만 아니라 조건에 따라 클래스 및 스타일을 동적으로 바인딩할 수 있기 때문에 데이터만 있으면 UI 제어도 쉽게 가능했습니다. | | **13주차** | 노트 작성 기능을 구현하였습니다. 마크다운 형식으로 쉽고 빠르게 작성할 수 있도록 tiptap  이라는 라이브러리를 활용하였습니다. #, `` 등 마크다운 문법을 사용하여 다양한 포맷으로 노트 내용을 작성할 수 있습니다. 또한 작성한 노트는 HTML 코드로 변환하여 저장할 수 있도록 기능을 구현하였기 때문에 노트 작성 후 저장 버튼을 누를 경우 노트 내용을 HTML로 변환하고, 서버로 전송하여 DB에 최종적으로 저장될 수 있도록 구성하였습니다. vue 개발자 도구를 적극 활용하여 자식 -> 부모 컴포넌트로 전달하는 이벤트를 디버깅할 수 있었습니다. | | **14주차** | 태그 추가 및 할 일 추가 기능을 서버와 연동했습니다. 관련 기능은 기존에GraphQL 리졸버로 구현해 두어서 쉽게 연동할 수 있었습니다. 개발 마무리 후 최종적으로 웹 브라우저에 서비스워커를 등록할 수 있도록 서비스워커의 생명주기, 바인딩할 수 있는 이벤트에 대해 사전 조사를 진행했습니다. 이후 오프라인 작동을 위한 캐싱과 푸시 알림을 위해 다양한 자료를 확인했습니다. | | **15주차** | 지난 주차에 웹 페이지 기능 개발을 마무리하여 오프라인 동작을 위한 캐싱과 푸시 알림을 수신하기 위해 서비스워커를 개발했다. 서비스워커는 설치 -> 활성화 순서로 작동하게 되며 HTTPS 혹은 로컬호스트에서만 설치되고 동작되는 스크립트이다. 브라우저에서 동작하긴 하지만 UI 스레드(메인)가 아닌 별도의 스레드에서 동작하므로 직접적인 DOM 접근이 불가능하다. 하지만 UI 스레드와 메시지를 통해 데이터를 주고받을 수 있으므로 어느정도 보완할 수 있다. 최종적으로 오프라인 캐싱과 푸시알림 이벤트를 받기 위한 리스너를 구현하여 마무리했다. |   **2. OJT 관련 사진 2장 첨부(활동 또는 결과물)**   |  |  | | --- | --- | | 스크린샷이(가) 표시된 사진  자동 생성된 설명 | 스크린샷, 모니터이(가) 표시된 사진  자동 생성된 설명 |   **3. 소감 및 건의사항(강의 내용에 대하여 구체적으로 작성)**   |  | | --- | | 실무와 비교해보면 매우 간단하게 프로그램 개발 과정의 초기 부분을 직접 진행했지만, 새로운 경험을 하게 되었습니다. 프로젝트 기간 동안 개발할 내용에 대한 기획과 전반적인 밑그림이 그려졌기 때문에 앞으로 세부적인 설계를 진행하여 자세하게 기술하고, 설계한 내용을 기준으로 기획한 서비스를 구현할 수 있던 것 같습니다. 또한 지금까지 학습해온 내용과 평소에 관심 있었던 웹 분야의 신기술을 적용한 서비스를 기획하고 설계해보니 다양한 기술이 있다는 것을 다시 한 번 알게 되었고 이번 PBL 프로젝트를 수행하며 스스로에게 더 성장할 수 있는 기회를 제공한 것 같습니다.  최종적으로 GraphQL, SPA 기반의 PWA (오프라인 캐싱, 푸시 알림)등의 기술을 접하게 되었으며 평소  관심만 가지고 있던 부분을 직접 개발해보며 많은것을 배우게 되었습니다. 비록 이번 PBL은 마무리되었지만, 앞으로도 다양한 기술에 관심을 가지고 꾸준히 배워나갈 수 있도록 습관을 길러 웹 분야에서의 선두주자가 될 수 있도록 노력하겠습니다. | |