

경력기술서

프로젝트에 참여하면서 경험한 것들에 대해 소개합니다.

프로젝트 경험에 대한 내용을 요약하여 소개한 뒤, 상세 내용을 서술하는 방식으로 작성하였습니다.

TeamSolution

2019.11. - 2022.11. 3SECONDZ 트랙을 주행하는 차량의 텔레메트리 데이터를 실시간으로 모니터링하는 웹 서비스

RaceSolution & Superrace Broadcast HUD

2020.04. - 2022.10. 3SECONDZ 레이싱 경기 관제를 위한 솔루션 & SUPERRACE 방송 중계 오버레이를 위한 HUD

SolutionMaster

2021.02. - 2022.12. 3SECONDZ 다수의 차량을 그룹별로 제어하는 기관용 종합 주행 관제 솔루션

Session Report : Lap VS Lap

2021.07. - 2022.04. 3SECONDZ 주행 세션을 분석하고, 다른 주행 데이터와 비교하여 시각화하는 웹 솔루션

Portfolio 2023

2023.03. - 2023.07. 개인 프로젝트 프로필 포트폴리오로, 새로 익힌 기술에 대한 활용 능력을 보여주기 위한 프로젝트

TeamSolution

3SECONDZ 2019.11. - 2022.11.

프로젝트 개요

트랙을 주행하는 차량의 텔레메트리 데이터를 실시간으로 모니터링하는 웹 서비스

활용 기술

HTML, CSS, JavaScript, jQuery, EJS, SCSS, D3.js, Google Maps API, AJAX(jQuery), WebSocket(faye), JSON, Git/Github, Zeplin, flatpickr

프로젝트 경험 요약

- 이전 경력에서는 단순 마크업과 사용자 DOM 이벤트 구현에 대한 경험만 해오다가, 이 프로젝트를 통해 처음으로 프론트엔드 작업 경험
- AJAX를 통한 RESTful API 호출
- AJAX를 통해 받은 JSON 데이터의 가공 및 업데이트 처리
- WebSocket을 통한 실시간 양방향 데이터 송수신
- D3.js를 활용한 SVG 그래픽 생성 및 업데이트
- CSS를 활용하여 HUD 그래픽에 실시간 데이터 연동 처리
- 컴포넌트 및 각 기능에 대한 모듈화 작업을 지속적으로 시도하였고, 대략 49% 정도까지 진행까지 하였으나 내부 사정으로 완료하지 못함

프로젝트 참여 상세

텔레메트리 모니터링 및 관제

- 텔레메트리 모니터링 및 관제
- 주행 리플레이
- 개인 & 팀매니저 서비스
- 앱
- 컴포넌트/모듈화 리팩토링

기존에 구현된 서비스의 유지보수 담당

- 2020년 위젯 기능 개발에 프론트엔드 참여
 - 세션 참여 인원의 요약 모니터링 위젯 UI 구현
 - 주행 상태 시각화 HUD 위젯 UI 구현
 - 메시지/콘솔 위젯 UI 구현
 - WebSocket을 통한 텔레메트리 JSON 데이터를 가공하여 활용
 - 송수신 데이터 형태, API 지원 요청 등 백엔드와의 협업

주행 리플레이

- 기존에 구현된 서비스의 유지보수 담당
- 2022년 리뉴얼 개발에 프론트엔드 참여
 - 리플레이 컨트롤 UI를 독립 가능한 컴포넌트로 설계 및 구현
 - 개선된 리플레이 컨트롤 UI의 DOM 조작 이벤트 구현 작업
 - 신규 기능 Marker UI 및 DOM 조작 이벤트 구현 작업

개인 & 팀매니저 서비스

- 기능 개발 이후에 참여하여 마크업 정돈 및 유지보수 담당
- 레이아웃 개선과 모달 추가 위주로 작업
- 이후 홈페이지 리뉴얼에서 개인 회원 서비스 페이지 올체인지 작업 담당

웹뷰 앱

- 기능 개발 이후에 참여하여 마크업 정돈 및 유지보수 담당
- 앱 레이아웃 개선 및 반응형 처리를 위한 마크업 리팩토링 진행
 - Rem을 활용한 반응형 작업
 - Portrait / Landscape 전환을 단일 페이지에서 CSS로 처리되도록 적용
- 타이어 모니터링 모드를 WebSocket을 통해 ON/OFF 되도록 처리

컴포넌트/모듈화 리팩토링

- 기능을 확장하며 프로젝트를 담당하다 보니 컴포넌트화에 대한 필요성을 절실히 느낀
- 2020년 말부터 클라이언트 단구조 설계, 팀장님께 지속적으로 피드백 받으며 진행
- 다른 큰 프로젝트에 비해 우선순위가 낮아, 일정이 비교적 여유로운 겨울에 주로 진행
- 2022년 가을부터 본격적으로 다시 시작하며 많이 진전되었으나 내부 사정으로 종료
- EJS단위로 프레임 구성, JS 템플릿 리터럴을 활용하여 컴포넌트 생성 구조로 진행
- 이 때의 경험으로 컴포넌트 중심 개발에 많은 관심을 갖게 됨

RaceSolution & Superrace Broadcast HUD

3SECONDZ 2020.04. - 2022.10.

프로젝트 개요

국내 최대 레이싱 경기 SUPERRACE의 실시간 관제 솔루션 & 방송 중계 HUD 오버레이 서비스

- TeamSolution의 상위 집합 솔루션

활용 기술

HTML, CSS, JavaScript, jQuery, EJS, SCSS, D3.js, Google Maps API, AJAX(jQuery), WebSocket(faye), JSON, Git/Github, Zeplin, Figma

프로젝트 경험 요약

- TeamSolution의 기능 확장 프로젝트
- 작업 간에 기획/백엔드와 활발한 소통으로 진행, 대부분의 레이아웃 구성은 기존 테마를 상속받아 직접 레이아웃 구성
- AJAX를 통한 데이터 조회 및 DB 저장, WebSocket을 통해 데이터 모니터링 및 각 클라이언트에 제어 전파
- 비동기 이슈 해결을 위해, 미들웨어에서 처리된 순위 데이터를 WebSocket을 통해 수신받아 적용
- 경기장 내 네트워크 이슈 해결을 위해 비동기 프로세스를 동기 프로세스로 전환 하는 등의 성능 최적화 리팩토리
- HUD 위젯의 매커니즘을 방송용 오버레이를 위한 HUD 컨트롤 페이지로 작업 (채널A, 유튜브 등 생중계 화면에 활용)
- 자막 중계를 위한 메신저의 디자인 및 UI 구현, 프론트엔드 작업 진행

프로젝트 참여 상세

리더보드

- 리더보드
- 엔트리컨트롤
- 위젯
- 방송 중계용 HUD
- 중계 자막 메신저

성능 개선을 위한 프로세스 재설계

- 새로운 환경에서의 생겨난 네트워크 이슈, 비동기 이슈로 인한 업데이트 오류로 인해 기존 프로세스를 분석하여 알고리즘 보완 작업 실시
- 순위 정렬에 필요한 데이터 계산 처리 개선
 - 클라이언트 단의 성능 저하를 해결하기 위해, 백엔드 파트에서 처리한 미들웨어 계산 값을 WebSocket을 통해 데이터를 수신 받아 적용

엔트리컨트롤

- 기존 팀 매니저의 팀 제어 패널의 기능 확장
 - AJAX를 통해 선수 데이터 배열 호출하여, 컨트롤 테이블 생성 및 업데이트
 - WebSocket을 통해 컨트롤 테이블 내 모니터링 요소에 데이터 업데이트
- 예선/결승의 구분이 있는 세션의 컨트롤 제어 기능 신규 추가
 - AJAX를 통해 세션 개설 DB 업데이트
 - WebSocket을 통해 다른 관제 클라이언트로 전파
- 테이블 정렬 기능
 - DOM 이벤트 재등록 처리, 모니터링 데이터 전역 상태 객체 참조 후 적용

위젯

- 트랙 상의 차량 위치에 대한 그래픽 타입 메뉴 위젯
 - D3.js를 통해 SVG 그래픽 전환 처리 구현
- 트랙 깃발 발령 기능 위젯
 - WebSocket을 통한 즉각 반영과, AJAX를 통해 DB 저장 등을 처리
- 기타 클라이언트 독립적인 기능 ON/OFF 스위치 위젯 구현

방송 중계용 HUD

- 기존에 활용되던 HUD 위젯의 데이터 연동 알고리즘을 새로운 UI로 적용
 - WebSocket을 통한 실시간 텔레메트리 데이터 연동 처리
 - 싱크를 위한 딜레이 기능, 모니터링 선수 변경을 위한 검색 기능

중계 자막 메신저

- 시청자를 위한 중계 안내 자막 송신 기능 강화를 위해 개발
 - 기존의 메시지 콘솔 위젯에서 활용되는 기능을 분리하여 강화
 - 디자인부터 UI 마크업, 프론트엔드 개발 전담

SolutionMaster

3SECONDZ 2021.02. - 2022.12.

프로젝트 개요

다수의 차량을 그룹별로 제어하는 기관용 종합 주행 관제 솔루션

- RaceSolution의 상위 집합 솔루션

활용 기술

HTML, CSS, JavaScript, jQuery, EJS, SCSS, D3.js, Google Maps API, AJAX(jQuery), WebSocket(faye), JSON, Git/Github, Zeplin, Figma, flatplckr

프로젝트 경험 요약

- RaceSolution의 기능 확장 + 신규 디자인과 신규 기능 등이 추가된 매니저 페이지 리뉴얼
- 프로토타입을 공유하며, 기획/디자인/백엔드와 활발한 소통으로 프로젝트 참여
- 팀매니저 페이지와 동일한 URL을 권한에 따라 렌더링 처리
- 차량, 드라이버, 주행 그룹, 하드웨어 등의 각 아이템에 대한 다수의 일괄 관리 콘솔 마크업 및 프론트엔드 참여
- 세션 컨트롤의 트랙 선택, 예약 기능(달력, 시간 등, flatpickr 활용), 세션 개설 상태 등의 마크업 및 프론트엔드 참여
- 주행 스케줄 관리를 위한 스케줄 매니저 기능 마크업 구현 작업, 프론트엔드 설계 참여
- 텔레메트리 모니터링 관제 페이지 내의 “주행 그룹” 운영 시스템 적용을 위한 마크업 및 프론트엔드 참여
- 기존 HOST 단위의 리플레이 서비스에서 세션 단위의 리플레이 서비스 마크업 및 프론트엔드 참여
- 이 프로젝트에서 컴포넌트 별로 생성 및 관리하는 체계 첫 도입 시도

프로젝트 참여 상세

매니저 페이지

- 매니저 페이지
- 세션 컨트롤
- 스케줄 매니저
- 주행 그룹 시스템
- 세션 리플레이

- 리뉴얼 시안으로 마크업 및 프론트엔드 작업

- 기존 팀매니저 페이지 URL 접근 시, 로그인 사용자 권한에 따라 렌더링되도록 처리
- 콘솔 테이블에서 각 항목 또는 체크항목에 대해 모달을 통해 내용이 변경되도록 구현
- AJAX를 통한 편집 데이터, 신규 데이터, 삭제 등 DB 반영 처리
- WebSocket을 통한 차량, 하드웨어 상태 모니터링 구현

세션 컨트롤

- 독립 컴포넌트로 마크업 및 프론트엔드 작업
- 예약 기능에 flatpickr 라이브러리를 활용하여 날짜/시간 선택 기능 구현
- AJAX로 세션 개설/종료, WebSocket을 통한 세션 상태 모니터링 기능 구현

스케줄 매니저

- 세션 일정 관리를 위한 시즌별(연도별) 관리 체계 설계
- 레이아웃 와이어프레임 작업 후 디자인 작업 진행
- 프레임 UI 마크업 작업 후, 컴포넌트 재생성 마크업 작업 진행
- 프론트엔드 프로세스 설계 후, 다른 개발자분께 내용 인계

주행 그룹 시스템

- 텔레메트리 모니터링 관제 페이지 체계 개편
- 리더보드에 탭을 추가하여 그룹 별로 리더보드가 필터링되도록 구현
- 엔트리 컨트롤 (주행 관리 제어 패널)
 - 세션에 등록된 주행 차량을 그룹 별로 나누어 테이블 컴포넌트 생성
 - AJAX로 그룹 상태를 DB에 저장, WebSocket을 통해 각 클라이언트에 전파

세션 리플레이

- HOST 내의 여러 세션으로 주행이 이루어지면서 세션 리플레이 체계 도입
- DB 상의 세션 시작/종료 시간에 맞게 주행 데이터를 crop
- 라이브로 진행중인 세션에 대한 리플레이 지원을 위한 프론트단 작업 처리

Session Report : Lap VS Lap

3SECONDZ 2021.07. - 2022.04.

프로젝트 개요

주행 세션을 분석하고, 다른 주행 데이터와 비교하여 시각화하는 웹 솔루션

활용 기술

HTML, CSS, JavaScript, jQuery, EJS, SCSS, D3.js, D3-queue, AJAX(jQuery), JSON, Git/Github, Zeplin, Figma, DOM-to-image, jsPDF

프로젝트 경험 요약

- 입사 전에 진행이 중단되었던 프로젝트에 대한 작업 재개를 위해 기획단계부터 다시 시작
- 기존 디자인 시안에 대해 기획 변경에 따라 재작업이 필요하였으나, 테마에 맞게 직접 레이아웃 디자인 작업 담당
- 마크업과 프론트엔드 기능에 대해 처음부터 다시 시작하면서, 컴포넌트/모듈 단위 개발 본격적으로 시작
- 동기 프로세스를 가진 AJAX 데이터 호출 구조로 구성
- 주행 세션에 대한 출력용 드라이빙 리포트 마크업 및 프론트엔드 참여

프로젝트 참여 상세

- 컴포넌트 체계 도입
- 주행 데이터와 데이터 비교
- 트랙맵 컴포넌트
- 위젯
- 라인 차트
- 출력용 드라이빙 리포트

컴포넌트 체계 도입

- 프로젝트를 다시 구축하면서 컴포넌트 체계 도입
- 이전에 기존 솔루션을 분석하며 나누어둔 UI 구조를 참고하여 구성
- 컴포넌트 별로 내부 요소 생성, 업데이트, 제거 동작이나 DOM 이벤트 작업하여 관리

주행 데이터와 데이터 비교

- AJAX로 주행 데이터 호출하는 과정을 동기적으로 처리하여, 전역 객체 데이터가 완성된 후 UI에 적용되도록 설계하여 적용
- 비교 데이터 변경 시, AJAX 호출 과정에 동기 프로세스를 그대로 적용
- 권한에 따라 비교 데이터 접근 여부 구분 처리
- 재생/정지/탐색 기능을 위치 인덱스와 시간 인덱스를 기준으로 각각 처리되도록 구현

트랙맵 컴포넌트

- 트랙맵은 기존 솔루션에서도 메인으로 활용되어 컴포넌트화 필수
- D3.js 기반으로 그래픽 구현
- 트랙 그래픽 생성, 섹터 구분, 차량 위치 표시, 텔레메트리 데이터 시각화 등 그래픽 요소 전반에 대한 기능 모듈화 진행
- 사용자 인터렉션 탐색 기능에 마우스, 모바일 터치, 트랙패드 동작을 각각 구분하여 부드럽게 동작되도록 성능 개선을 위해 집중

위젯

- 기존 솔루션에도 사용할 수 있도록 위젯 UI 및 동작 구조 전반을 리뉴얼하여 코드 개선
- 랙차트 위젯, 탐색 인덱스 기준 변경 스위치, 재생/정지 스위치 등 신규 기능 구현

라인 차트

- 탭이 존재하고, 각 탭에 그래프를 추가, 업데이트 제거할 수 있는 데이터 구조로 DB에 저장되도록 관리하는 구조 설계 및 적용
- 주행 데이터 채널 선택에 따라 라인 차트의 추가, 생성, 교체, 제거가 되도록 작업
- 라인 차트는 D3.js를 활용하여 구현
- 각 라인 차트 셀을 리사이징하고, 리사이징된 높이가 저장되도록 처리
- 사용자 인터렉션 탐색 기능에 마우스, 모바일 터치, 트랙패드 동작을 각각 구분하여 부드럽게 동작되도록 성능 개선을 위해 집중

출력용 드라이빙 리포트

- A4 포맷으로 마크업을 진행
- D3.js 활용하여 트랙 그래픽, G plot 등을 생성
- DOM-to-image 라이브러리를 활용하여 JPG, PNG 등으로 변환 및 다운로드 처리
- jsPDF 라이브러리를 활용하여 PDF 파일로 변환 및 다운로드 처리

Portfolio 2023

개인 프로젝트 2023.03. - 2022.07.

프로젝트 개요

프로필 포트폴리오로, 새로 익힌 기술에 대한 활용 능력을 보여주기 위한 프로젝트

활용 기술

공통 기술

HTML, CSS, JavaScript, SCSS, React, Recoil, Tailwind CSS, JSON, Git/Github, Vercel, Figma

Vite 프로젝트

Vite.js, React Router, React Helmet, React Intersection Observer

NextJS 프로젝트

Next.js, TypeScript, Styled-components, React Query, GSAP, Firebase Database, ESLint, Prettier

프로젝트 경험 요약

- 퇴사 후 공부한 기술들을 활용하고, 결과물 완성까지를 목표로 프로젝트 시작
- 프로젝트를 위한 시안 디자인 작업 진행
- 최종 결과물은 NextJS 프로젝트로 Vercel을 통해 배포
- 실무에서 Vite 환경에서의 React 작업 경험으로 프로젝트 작업 후 완성
 - Recoil, Tailwind CSS, React Router, React Helmet 등 새로 접하는 라이브러리들을 활용
- NextJS 13.4의 App Router 환경에서의 SSR 작업 경험과, TypeScript 활용을 위해 프로젝트 마이그레이션 진행
 - AJAX 기술 활용, Vite 프로젝트보다 개선된 코드

프로젝트 참여 상세

Vite 프로젝트

- Vite 프로젝트
- Next 프로젝트

- React로 완성한 첫 프로젝트
- 상태관리를 위한 Recoil 도입
- Dark/Light Mode 적용
- SEO를 적용하기 위한 React Helmet 도입
- 여러 페이지에 대한 URL 접근이 가능하도록 React Router 도입
- URL Query Strings를 통해 프로젝트 상세 내용에 접근
- React Intersection Observer로 뷰포트 진입시 스크롤 인터렉션 모션 구현
- 마크업에는 Tailwind CSS 기본 반응형 클래스 활용
- Github Pages로 배포하다가, Vercel로 배포 플랫폼 변경

NextJS 프로젝트

- Next.js 활용을 위해 프로젝트 마이그레이션 진행
- TypeScript 도입
- 상태관리에는 Recoil로 유지하되 상태관리 코드 다시 작성
- SSR 환경에서의 AJAX 기술 활용
 - 프로필, 프로젝트 페이지에는 NextJS API를 활용한 서버 단에서의 fetching 적용
 - 프로젝트 상세 내용에는 React Query를 활용하여 클라이언트 단의 fetching 적용
 - 외부 데이터베이스로는 Firebase의 Realtime Database 활용
- 배포 환경에서는 데이터를 Firebase Database에서 가져오도록 처리
- 스타일링에는 Tailwind CSS로 프로토타이핑 후, Styled-components로 정리
 - Dark/Light Theme, Mixin, Keyframes, Global Style 등 활용
- 스크롤 인터렉션에 GSAP - ScrollTrigger 활용
- Vercel을 통해 배포