

# 송상수

☎ +821036018508   @dvlprsong@gmail.com

실시간 금융 데이터와 시계열 시각화를 주력으로 다뤄온 프론트엔드 개발자 송상수입니다.

TypeScript, React를 기반으로 GraphQL(AWS AppSync), SSE 스트리밍, 실시간 무한스크롤 피드 등을 모듈화·최적화한 경험이 있습니다. Primitive 컴포넌트와 Compound Component 패턴 기반의 디자인 시스템을 구축하고 지속적으로 개선했습니다.

GitHub Actions + AWS(Amplify·S3·CloudFront)로 제품 배포 프로세스에 CI/CD를 적용하였고, 데이터 시각화 기능에 있어 Highcharts, D3 등 관련 라이브러리를 재사용하기 편하게 모듈화함으로써 성능과 유지보수성을 높여왔습니다.

서비스의 가장 앞단에서 사용자와 마주하는 웹 프론트엔드 분야에 매력을 느껴 커리어를 시작한 이래, 현재는 기능 개발을 포함한 제품 생애주기의 전반에 관심을 갖고 개발을 하고 있습니다.

좋은 동료이자 신뢰할 수 있는 전문가로서, 사용자가 쓰기 편하면서도 기술적으로도 안정적이고 효율적인 제품을 만들고 싶습니다.

- GitHub

<https://github.com/dev-song>

경력 6년 11개월



주식회사 한경에이셀

2020.12 - 2025.12 (5년 1개월) | 정규직 | Web Frontend | Software Engineer

금융 리서치 어시스턴트 웹서비스 epic AI 개발

2024.01 - 2025.12

# 역할

- 피드 목록 형태의 무한 스크롤 + 실시간 데이터 (GraphQL) 화면 개발
- 채팅 형태의 LLM 코파일럿 서비스 화면 개발 (SSE Streams)
- 증권사 애널리스트 보고서를 모아서 조회할 수 있는 테이블 및 필터 페이지 개발
- 주식시장 상장회사 정보 대시보드 페이지 개발
- 자체 디자인 시스템 구축 및 고도화 (shadcn/ui, Radix UI 구조 차용)

# 사용 기술

- TypeScript
- React (Vite)
- React Query
- Tailwind CSS
- AWS Amplify

## # 주요 성과

- 디자인 시스템 구축 시 디자인 팀과 적극적으로 논의하며 컴포넌트 설계 기준 마련
- 디자인 시스템 구축 전후로 신규 화면 개발 소요 기간 40% 이상 단축

## # 맞닥뜨린 문제와 해결 과정

- 기획 미비와 의사소통 혼선에 따른 개발 프로세스 비효율
  - 문제 상황  
신규 서비스이고 PM이 없는 환경에서 기획 변경이 수차례 이뤄져 개발 일정이 지연되는 상황이 자주 발생
- 해결 과정  
프로젝트 회의 시 UX 관련 기준을 중점적으로 제시하면서 프로젝트의 완수를 기준으로 주도적으로 팀의 의사결정을 조율

## - 디자인 시스템 구축 및 고도화

- 문제 상황  
디자이너와 개발자 사이의 용어, 컴포넌트 단위, 컴포넌트 설계 개념 불일치 (디자이너는 '외관' 기준, 개발자는 '기능' 기준)  
개발 시 디자인의 구조 방식 차이로 커뮤니케이션 비용이 커지는 상황이 빈번하게 발생
- 해결 과정  
디자이너와 개발자가 모든 개념을 완벽하게 일치시키는 것은 현실적으로 어려워, 가장 기본 개념만 통일하고 그 외는 지속적 조율  
디자인적 외관 속성을 모아놓은 컴포넌트를 Primitive로 분리하여, 디자인에서의 다양한 상황에 유기적으로 대응  
개발에서 사용되는 공통기능은 Compound Component 패턴 및 React Context를 적극적으로 활용하였음  
(Radix UI의 headless 방식 구조 적극 차용)

## 금융 데이터 웹서비스 BigFinance 차트 코드 리팩토링

2023.10 - 2023.12

## # 역할

- 신규 차트 추가 시 boilerplate 중복 및 파편화 해결
- 차트 관련 기능 유지보수 난이도 완화

## # 사용 기술

- TypeScript
- React
- SCSS
- Highcharts (JavaScript 라이브러리)

## # 주요 성과

- 신규 차트 구현 기간 50% 이상 단축 (약 2일 -> 약 1일)
- 차트 유지보수 기간 60% 단축 (8시간 -> 3시간)
- 차트 관련 버그 발생률 75% 감소 (월 평균 4회 -> 1회)

## # 맞닥뜨린 문제와 해결 과정

- 두 가지 차트 Renderer의 혼용
  - 문제 상황  
데이터의 유형 (일반/시계열) 에 따라 라이브러리 렌더링 클래스를 구분해서 사용해야 하는 이슈  
두 클래스의 사용법이 국소적으로 다른데 문서와 TypeScript 정의가 명확하지 않은 경우가 있어 혼동을 유발하기 쉬움
- 해결 과정  
불명확한 동작과 TypeScript 정의를 개발사에 문의하거나 (이메일, GitHub 이슈 생성) 코드를 직접 분석해 명확한 의미와 Type

Script 정의를 갖는 Wrapper 컴포넌트 작성

내부적으로 쉽게 사용할 수 있도록 차트 렌더링, 차트의 부속 기능 (스크린샷, 다운로드, ...), 종류별 차트 설정 보관으로 나눠 모듈화

## 한국 주식시장 실시간 거래 정보 페이지 개발

2023.01 - 2023.09

### # 역할

- KOSPI/KOSDAQ 지수 및 개별주 가격 실시간 표시 (GraphQL)
- 사용자별 권한 확인 (AWS Lambda + AWS Secrets Manager)
- FE CI/CD 파이프라인 구축 (AWS S3 + AWS CloudFront + GitHub Actions)

### # 사용 기술

- AWS Amplify (AppSync, Lambda, Secret Manager)
- TypeScript
- React
- SCSS
- Recoil
- GitHub Actions

### # 주요 성과

- CI/CD 파이프라인 구축 전후 배포 시간 60% 이상 감소 (약 30분 -> 10분)
- 오픈 소스 라이브러리를 직접 분석하며, 거대한 코드베이스를 읽는 노하우 습득

### # 맞닥뜨린 문제와 해결 과정

- AWS AppSync의 구조적 한계

#### - 문제 상황

기업의 실시간 가격을 수신해야 되는데, 동시에 일정 갯수 이상을 구독하려 하면 에러가 발생

AWS의 관련 문서와 예시가 부족해 문제의 원인이 코드를 잘못 작성해서인지, AppSync 자체의 문제인지 판단이 어려운 상황

#### - 해결 과정

수회의 엡지 케이스 테스트, AppSync와 연계된 AWS 서비스 문서 리서치 결과 AWS DynamoDB의 스펙 관련 이슈를 파악

근본적으로 수정하기 어려운 문제이므로, 화면에 동시 표시되는 기업 수가 스펙 상 갯수 이내가 되면서도 UX를 해치지 않도록 개선

안을 도출하여 프로젝트 회의에서 적극적으로 건의 및 해당 방향으로 의사결정 도출

- GraphQL Subscription 라이브러리 타입 정의 오류

#### - 문제 상황

AWS Amplify 라이브러리의 GraphQL subscription 관련 메소드 사용 시 내부 타입을 외부로 노출하지 않아 any 사용이 강제되어 견고한 코드 작성이 어려움

#### - 해결 과정

라이브러리를 직접 분석, 해당 메소드가 내부적으로 사용하는 라이브러리 (rxjs) 파악하여 그 라이브러리에 대한 타입을 별도로 devDependency에 추가

## 한국 주식시장 기업별 주가/시가총액 시각화 페이지 개발

2022.11 - 2023.01

### # 역할

- 트리맵\*과 히트맵\*이 혼합된 형태의 국내주식 주가/시가총액 데이터 시각화

- \* 트리맵: 전체에서 특정 개체의 비중을 사각형의 크기 차이로 표시하는 데이터 시각화 기법
- \* 히트맵: 특정 개체의 값의 크기를 색상의 명도/채도로 표시하는 데이터 시각화 기법
- 줌인/줌아웃 및 마우스 호버 시 정보 토틱 표시

#### # 사용 기술

- TypeScript
- React
- SCSS
- d3.js
- HTML5 canvas

#### # 주요 성과

- 시각화 관련 DOM Node 수 99% 이상 감소 (15,000개 이상 -> 50개 이내)
- 줌/호버 시 프레임 저하 이슈 해결

#### # 맞닥뜨린 문제와 해결 과정

- SVG 기반 렌더링의 성능 제약
  - 문제 상황

D3.js의 예제 코드 기반의 SVG 렌더링은 DOM 요소가 너무 많아져 (15000개 +) 성능이 저하되는 이슈  
특히나 마우스오버 시 스타일링, 줌인/줌아웃 등 리렌더링이 일어나는 시점의 성능 저하 심화
  - 해결 과정

SVG가 아닌 HTML5 Canvas를 기반으로 D3.js를 활용하도록 변경  
불필요한 렌더링이 최소화되도록 알고리즘 최적화  
벡터 그래픽에서 래스터 그래픽으로의 변경에 따른 디스플레이별 해상도 차이 보정

#### # 참고자료

삼프로tv (YouTube)

한국경제TV (YouTube)

회사 기술 블로그 (Medium) (<https://medium.com/aicel-technologies/%EB%B9%85%ED%8C%8C%EC%9D%B4%EB%82%B8%EC%8A%A4-%ED%9E%88%ED%8A%B8%EB%A7%B5-%EA%B0%95%EB%A0%A5%ED%95%98%EB%A9%B4%EC%84%9C%EB%8F%84-%EB%B6%80%EB%93%9C%EB%9F%BD%EA%B2%8C-%EB%A7%8C%EB%93%A4%EA%B8%B0-ac1169f24711>)

### 회사 홈페이지 리뉴얼 및 레거시 코드 리팩토링

2022.10 - 2022.11

#### # 역할

- 홈페이지 리뉴얼에 따른 홈페이지 구조 및 디자인 변경
- 적응형 웹 디자인 반영 (모바일, 태블릿, 데스크탑)
- 홈페이지 언어에 따라 폰트 전환

#### # 사용 기술

- TypeScript
- React
- SCSS + CSS modules

#### # 주요 성과

- 이미지 포맷 최적화로 이미지별 평균 네트워크 전송량 70% 이상 감소 (1MB -> 300KB)
- SVG 애니메이션 영상으로 포맷 변경하여 네트워크 전송량 99% 이상 감소 (20MB -> 100KB)

#### # 맞닥뜨린 문제와 해결 과정

- 팀 내 스타일링 관리의 어려움
  - 문제 상황  
기존 스타일 관리는 SCSS와 BEM 방법론을 함께 적용했었음  
팀원이 추가되며 네이밍 방식의 차이에서 오는 의사소통 오류와 다중으로 중첩된 CSS 코드가 유지보수를 어렵게 함
- 해결 과정  
스타일 관리방법 개편 논의를 통해 유지보수 부담 최소화를 가장 중요한 기준으로 삼아 주도적으로 대안 리서치  
CSS Modules 방식을 SCSS와 결합해서 스타일 관리 scope를 컴포넌트로 국한하도록 변경

#### - 웹페이지 성능 저하

- 문제 상황  
추가된 이미지와 SVG 애니메이션이 각각 네트워크 부하 (이미지 사이즈) 와 렌더링 부하 (DOM 요소 수) 를 유발함
- 해결 과정  
이미지는 WebP 포맷 변환 등 이미지 용량 압축 및 사용자 기기 해상도에 따라 최적화하여 전송  
명시적 크기 지정 및 Skeleton 적용으로 Layout Shift를 줄이고 이미지 로딩 지연에 따른 UX 저하 최소화  
SVG 애니메이션은 해상도별 영상으로 변환하여 렌더링 부하 최소화

#### # 참고자료

홈페이지 주소 <https://aiceltech.com/>

\* 2024년 7월 기준 사내 마케팅 정책 변화로 정적 페이지들은 Webflow 기반 페이지로 전환 완료

\* API 활용한 동적 페이지들 및 사용자 페이지에 한해 리뉴얼 내용 잔존

### 금융 데이터 웹서비스 BigFinance용 관리자 페이지 개발

2022.05 - 2022.09

#### # 역할

- 주요 관리자 페이지 기능 개발
- FE CI/CD 파이프라인 구축 (AWS S3 + AWS CloudFront + GitHub Actions)
- 프로젝트 배포 (AWS S3 + AWS CloudFront + AWS Route53)

#### # 사용 기술

- TypeScript
- React
- SCSS
- AWS
- GitHub Actions

### 금융 데이터 웹서비스 BigFinance 내 대시보드 및 데이터 시각화 페이지 개발

2021.03 - 2022.09

#### # 역할

- 재무지표 등 금융 시계열 데이터 유형별 시각화 (Highcharts 라이브러리)
- 금융 데이터 Excel 파일 다운로드 기능 (ExcelJS 라이브러리)
- 반응형 웹 디자인 반영 (모바일, 태블릿, 데스크탑)

- TypeScript 도입 및 적용

# 사용 기술

- TypeScript

- React

- SCSS

# 참고자료

홈페이지 주소 <https://bigfinance.co.kr/>

## 회사 홈페이지 관리자 페이지 기능 추가 및 레거시 코드 리팩토링

2020.12 - 2021.02

# 역할

- 신규 관리자 기능 추가

- 기존 기능 유지보수 및 리팩토링

# 사용 기술

- JavaScript

- React

- SCSS

# 맞닥뜨린 문제와 해결 과정

- 파편화 및 문서 부재로 인한 레거시 코드 분석 어려움

- 문제 상황

기존에 구현되어 있던 관리자 페이지는 백엔드 개발자 2인과 디자이너(퍼블리셔)가 나눠서 작업했던 상황

공통의 코드 컨벤션이 없어 코드의 목적을 일견에 판단하기 어렵고 중복 코드가 산재하며 라이브러리도 파편화되어 있는 상태

- 해결 과정

각 기능별 기획안을 확인하고 담당자에게 확인해 기획안에 없는 디테일을 파악

최소한의 프론트엔드 코드 컨벤션을 논의해 결정하고 (AirBnB 방식) 기존 담당자별 작성 방식을 파악하고 리팩토링

컴포넌트화를 통해 중복 코드 최소화

동일한 목적의 라이브러리는 가장 보편적으로 사용되는 하나만 남기고, 바닐라 JavaScript로 대체할 수 있는 라이브러리는 제거



에스엘엘중앙 주식회사

2017.09 - 2019.06 (1년 10개월) | 정규직 | 인사담당자

## 디지털 콘텐츠 제작인원 초과근무 보상체계 수립/운영

2018.08 - 2019.05

디지털 영상 콘텐츠 제작인원의 과도한 초과근무 실태 개선을 위한 보상체계 신설

제작 현장의 특성에 따른 시간대별 초과근무 측정 시스템 도입, 해당 시스템 운영 및 수당 지급 담당

## 파견 임직원 급여/재계약 기준 체계화

2018.06 - 2018.12

개인마다 모두 다른 파견 임직원의 급여 지급 및 재계약 기준 부서별/직무별 체계화  
합리적 기준(경력, 재직기간 등) 적용하여 불필요한 직원 간 갈등 및 퇴사를 저하

### 임직원 급여 관리 및 지급

2018.07 - 2019.05

기본급, 성과급, 초과근로수당 등 정기적/비정기적 임직원 급여항목 관리 및 지급  
4대보험, 식비, 연말정산 추가 납부 등 급여공제항목 관리

### 임직원 복지 및 교육 운영

2017.10 - 2019.05

임직원 경조사 및 자녀 학자금 지원, 복지 포인트 지급 등 복지제도 운영  
임직원 대상 정기적/비정기적 교육, 법정 의무교육 등 사내 교육 계획 및 운영

### 공개 채용 및 수시 채용 담당

2017.10 - 2019.05

채용 계획 작성, 채용 프로세스 진행 (공고 게시, 이력서 및 지원자 관리, 필기시험 및 면접 진행 등)

## 학력

---



### 한국방송통신대학교

2022.03 - 2024.02 | 졸업 | 컴퓨터과학과



### 한양대학교

2010.03 - 2017.08 | 졸업 | 경제금융학과

## 스킬

---

React

JavaScript

TypeScript

Git

AWS

## 수상/자격증/기타

---

AWS Certified Developer - Associate



2025.01 | 자격증



**AWS Certified Solutions Architect - Associate**

2024.09 | 자격증



**정보처리기사**

2024.09 | 자격증

## 언어

---



**영어 | 비즈니스 레벨**

TOEIC 990 | 2020.09.27

TOEIC Speaking Level 7 | 2016.11.20

## 링크

---

[🔗 https://github.com/dev-song](https://github.com/dev-song)