

# 신성우

Frontend Engineer

[Github](#) · [TechBlog](#)

24siefil@gmail.com · 010-9147-1041



저는 \_\_\_\_\_ 엔지니어 입니다.

- 제품의 변경을 우선적으로 고민하는
- 소신있게 반대하지만 언제든 틀릴 수 있음을 견지하는
- 도전적인 미션과 밀도 높은 성장을 갈망하는

## 경험 - 뱅키즈 창업

참여 기간	8개월 (22.04 ~ 진행중)
참여 인원	9명
담당 직무	프론트엔드 엔지니어
기술 스택	React, TypeScript, React-Query, Redux, storybook
깃허브	<a href="https://github.com/bankidz/bankidz-client">github.com/bankidz/bankidz-client</a>
서비스 개발기	<a href="https://24siefil.oopy.io/bankidz">24siefil.oopy.io/bankidz</a>

### 서비스 개요

- 어린이를 위한 핀테크 서비스. 주요 기능은 저축 기반의 실전 금융 경험 제공.
- 기획자, 디자이너, 백엔드 엔지니어와 긴밀히 협업하여 초기 기획부터 출시 및 운영까지 A ~ Z를 경험
- 푸시알림을 제외한 모든 비즈니스 로직을 웹 기술로 구현 (모바일 앱 내 웹앱)

### 기술 기여

#### 상태관리 체계 이관/개선

- [Redux](#) (Thunk) 기반 비동기 통신에 대한 상태관리 시 서버 데이터를 위한 규격화 되지 않은 로직의 비대화, 장황한 Boilerplate, Client/Server 데이터에 대한 RTK Slice 혼재로 인한 응집도 악화 발생  
→ [React-Query](#)를 도입하여 Client/Server 각각에 대한 상태관리 분리
- [\[기술 블로그\] Redux는 본연의 역할에 충실하고 있는가?](#)

## 기술 기여

### 디자인 패턴 도입

- [Headless](#) 패턴과 [Compound Components](#)를 통해 [재사용성](#)이 높고 [변경에 유연한](#) 컴포넌트 설계/적용
- 팀내 불규칙한 컴포넌트 분할/분류 기준으로 인해 컴포넌트 재사용성 악화  
→ [Atomic Design](#) 패턴 도입. 해당 패턴의 [한계를 절충](#)하여 3단계로 컴포넌트 분할/분류 기준 정립.
- 페이지 일부 영역의 Fetch 실패에 대한 ‘재시도 UI’를 절차형으로 구현 시  
안티패턴(Presentational-Container) 및 Props Drilling 발생  
→ [선언형 컴포넌트](#)를 도입하여 관심사 분리
- [\[기술 블로그\] 지속 가능한 컴포넌트](#)

### App-like UX 구현, 사용성 개선

- 모든 UI에 대해 [반응형](#) 디자인 지원, react-transition-group 기반 [Parallax Routing Animation](#) 적용
- Skeleton UI, React-Query 기반 API Caching, localStorage Caching 적용

### OAuth 전략 수립, JWT 운용

- 초기에 [httpOnly & secure cookie](#)를 통한 JWT 관리로 높은 보안성 달성. 추후 웹뷰 호환성 문제로 인해 JWT 관리를 localStorage로 이관(다운그레이드)하며 [Over Engineering](#) 학습.

### 디자인 시스템 정립

- [storybook](#) 기반 컴포넌트 및 디자인 시스템의 테스트/관리 체계화
- [Figma](#)의 디자인 시스템과 [ThemeProvider](#) 동기화를 통해 UI 개발 효율 제고, Single Source of Truth 달성

### 에러처리 체계화

- try/catch문 기반 분산된 API 에러처리 시 유지보수성 악화, 에러처리 누락 등 휴먼에러 발생 → ‘에러 핸들러의 정의 위치, HTTP Status, 서비스 표준 에러 Code’ 기준 우선순위에 따른 [전역 API 에러처리](#) 체계 설계/적용

### 전역상태로 모달관리

- 모달 제어를 위한 Props Drilling 발생, 모달 액션 후 닫기 등 부수적인 코드 반복 발생, 모달 관리의 파편화로 인한 유지보수성 악화 → 모달 관리에 [전역상태](#)를 도입하여 컴포넌트간 커플링 해소, 부수적인 반복 로직 추상화, 중앙화된 모달 관리를 통해 유지보수성 개선. [Code Splitting](#)을 통해 로딩 성능 개선.

## 협업 기여

### 애자일 방법론 도입 주도

- 칸반 보드 기반 이슈 관리, 스프린트 단위 개발, 프리징 규칙, 스크럼 회의 정착 주도

### 업무 프로세스 개선

- 기존 디자인, 기획, API 수정이 발생하는 경우 일련의 대응작업이 순서에 따라 처리돼야 하지만 일부 작업이 누락된 채 파트간 이슈 이전 반복 → 칸반 보드에 이슈 템플릿(체크리스트, 이슈 이전 규칙) 도입

### 개발 프로세스 개선

- Git-flow, Github Automated Kanban 기반 CI 개발 프로세스 정착 주도. PR 및 코드리뷰 규칙, 코딩 컨벤션 정립.

## 교육

---

### 컴퓨터과학

#### Ecole 42 (20개월, 21.03 ~ 22.10)

- 프로젝트 기반 동료학습을 통해 학부 수준의 컴퓨터과학 학습
- [\[깃허브\] 수행한 프로젝트 모음](#)
- [\[기술 블로그\] 학습한 컴퓨터과학 이론 정리](#)

#### 홍익대학교 (졸업 유예, 16.03 ~ 23.02)

- 학부 과정의 주요 컴퓨터과학 과목 수료
- GPA 3.84 / 4.50
- [\[깃허브\] 수행한 프로젝트 모음](#)

### 웹 개발

#### 신촌 연합 IT 창업 학회 CEOS 수료 (6개월, 22.03 ~ 22.08)

- 웹 프론트엔드 개발 기초 학습
- बैंकि즈 MVP 개발