Taehyun Kim

Frontend Engineer

taehyun_fe@naver.com
github.com/qlemql

SUMMARY

Frontend engineer building booking, payment, and store-management products with React and TypeScript. I trace problems across connected services through code and production logs, then work with the responsible teams to resolve them. At T-order, I have developed order-tablet and advertising integrations, shared AI tools, and a bot for searching team work history. I also implement the APIs and database changes needed to deliver features.

EXPERIENCE

T-order

Mar 2026 - Present

Frontend Engineer

Ad Platform team → Frontend Chapter

> At a restaurant table-ordering platform, I worked on the ad admin, an owner-facing React webview, and the integration between the Vue ordering app and React advertising module. My work also covers production issues and team tooling. Since November 2026, I have also been developing a new internal ordering product.

01 Order and advertising integration

A React advertising module inside the Vue ordering app could block menu interaction before it was ready. State handling between ads could also cause the next ad to be skipped.

- Separated iframe creation from display, showing ads only after a ready acknowledgement (ACK). Handled timeouts and late ACKs, and separated the previous ad's closing state from the next pending response, with regression tests.
- Fixed postMessage failures by converting Vue reactive objects to plain objects in the shared sender. Agreed on prepaid and postpaid ad-display rules with the PM, preventing reloads during payment and ads over the cart.

> **Outcome / status**
> Owned the ordering-app implementation and QA through production release. Based on a 95-second p99 first POS response, adjusted the acceptance wait from 60 seconds to at most five minutes.

02 Owner-facing React webview

Owned frontend development for extending store search and sales-report requests from a KakaoTalk chatbot into a webview.

- Investigated authentication across the backends and agreed on a shared design with their developers. Kept tokens on the server and passed a session ID through the webview's shared API client.
- Built search, report requests, and campaign-specific store selection with React, TypeScript, and TanStack Query. Used MSW to check screens without live APIs, prevented duplicate submissions during Korean IME input, and validated external entry URLs.

> **Outcome / status**
> Shipped store search and report requests, then added store selection for owners entering through campaign links.

03 Tracing repeated order queries

Investigated repeated spikes in order-query requests from store tablets.

- Used AI to analyze OpenSearch logs, identified a roughly 41-second interval, and checked both codebases and store settings. Traced the loop to a 30-second idle timer and a 10-second default image.
- Reviewed the operating policy with the advertising team and proposed continuously playing existing news content instead of the default image. After the team applied the change to two stores, compared logs for matching weekdays.

> **Outcome / status**
> Matching-Sunday query totals across the two stores fell from 40,148 to 3,724, a **90.7%** reduction.

04 Slack bot for team work history

Planned, built, and operated a Python Slack bot that answers questions using work records in Confluence, Jira, and Git. It is used by five frontend team members.

- Implemented the bot with AI coding tools and designed a retrieval flow that narrows topic-based candidates with BM25, then lets an LLM select relevant documents and answer from those sources.
- Fixed answers that mixed in deployment history from another product, based on user feedback. Adjusted retrieval rules so prepaid-payment questions search the necessary ordering, store-management, and payment documents together.

> **Outcome / status**
> Expanded registered documents from 143 to 261, adjusted candidate counts, and checked for retrieval misses. Added connection recovery and document-read failure alerts, validating changes with real questions.

05 Shared AI tools and release checks

Turned personal review, release, and documentation-check tools into tools the team could share.

- Distributed Claude Code skills as a team plugin, separating repository-specific commands and branches into configuration. Added CI checks for configuration and version errors and sensitive content.
- Updated development guidelines in two repositories against the code. Improved deployment checks by cross-checking logs, tags, code, and documentation. Also directly fixed a potential duplicate-transaction path found while reviewing a teammate's code.

> **Outcome / status**
> Used the release skill for two regular releases and one hotfix, and added lint, type, and build checks to the ad admin's release process.

06 UI, API, and database work on a new ordering product

Since November 2026, I have been developing an internal ordering-product prototype handed over to me and one other developer.

- Implemented payment details and installment-term persistence across the React UI, Express API, and PostgreSQL migrations, and merged the changes into main.
- Opened a PR carrying tablet/mobile order-source information through responses, webhooks, and kitchen tickets, with a migration for sales totals by order source.

> **Outcome / status**
> The product remains under development. Validated E2E scenarios on older Chromium 81 and Firefox 79/104 browsers.

Device validation and other work

- Built on-device E2E tooling with ADB and GeckoView remote debugging. Reproduced and fixed a screen loop in stores without a logo and a cart-dialog bug, then merged the fixes into the release branch after device checks and regression tests.
- Investigated why rejoining a message room after socket reconnection could take up to 30 minutes, documented the effect on order/payment updates, and shared server and client response options.
- Confirmed with backend developers that discontinued themes had no active stores, then removed about 17,500 lines across 142 unused files. Also configured Sentry filters, per-project intake limits, and daily reports.
- Built and deployed a beverage-recommendation pilot, then added store-specific product configuration in the admin. Checked that ordering could continue when the recommendation screen was delayed or failed.

Riad Corporation · Ria

Mar 2024 - Sep 2025

Frontend / Product Engineer

Tech team · Sole frontend engineer

> Owned the frontend for Ria, a hotel quotation and booking service used by travel agencies, travelers, and hotel staff. Worked with product, design, and backend colleagues to expand into mobile booking and payment and improve the experience using behavioral metrics.

01 Guided input for Ria hotel quote requests

Built screens where AI turns travelers' chat input into quote requests and hotel staff respond with offers. Some users struggled to know what to enter in free-form chat.

- Used Fetch streams to display answers and request data incrementally, adding field validation, correction guidance, and retries for invalid or missing model output.
- Worked with product, design, and backend colleagues on direct-quote and hotel-recommendation paths. Grouped 12 fields into hotel, traveler, and breakfast information, with per-field editing after generation.

> **Outcome / status**
>
> Product event logs showed average request-completion time falling from **over ten minutes to three minutes or less**, measured through request submission.

02 Social login to simplify signup

While investigating falling signup conversion, identified email-only registration as a likely obstacle for users accustomed to social login.

- Researched provider policies and APIs, defined signup rules, and implemented Google, Kakao, and Naver login across React screens, Go callback/account APIs, and database/Ent schemas.
- Requested only required information unavailable from providers. Later preserved the pre-login route and quote information and added guidance for Google login in in-app browsers.

> **Outcome / status**
>
> In a weekly measurement after launch, **75.83%** of new signups used social login. Marketing and business direction also changed, so I did not attribute overall signup growth solely to this feature.

03 Mobile hotel search, booking, and payment

Expanded the initial travel-agency product so individual travelers could also book and pay on mobile.

- Built search, hotel details, quotes, payment, and booking management for widths from 375px. Linked Google Maps with listings and shared hotel information between desktop popups and mobile bottom sheets.
- Integrated Stripe initially with the CTO and owned subsequent UI and SDK changes. Separated payment-method display policy from gateway selection, preserving input and consent through mobile redirects before payment verification and booking confirmation.

> **Outcome / status**
> Shipped mobile booking and payment, and sent GA4 and Meta Pixel events after quote-request and booking-creation APIs succeeded.

Product metrics and development workflow

- Queried behavioral metrics from the database and iterated daily with product and design. Weekly measurements around separate improvements showed quote views rising from 58.16% to 90.63% and guest quote-request conversion from 53.8% to 62.35%. Marketing and acquisition changes also affected these figures.
- Found CI cache restoration slower than dependency installation, removed the cache, and reduced build time from four minutes to one.

Lighthouse

Dec 2021 - Jan 2024

Frontend Engineer

Engineering team

> Built and operated web services for care facilities and families. On FamilyCare, handled requirements, task allocation, estimates, and PR reviews.

Key work

- On FamilyCare, developed with KidsNote, separated business logic from UI and implemented role-based menus, canvas e-signatures, and an external CMS integration. Consolidated repeated CI installation steps, reducing runtime from seven minutes to five.
- Moved Dailybook's API data handling from Redux Saga to React Query hooks for requests, caching, and updates. Removed unused libraries, reducing its bundle from 1.78MB to 1.47MB.
- Independently built and launched Jangji-Helpcall, a burial-site matching service using Naver Maps. Migrated TodayCare to an SPA and introduced TypeScript and Sentry.
- Returned to Hanghae99 as a mentor after graduating, coaching React students alongside my full-time role (Nov-Dec 2022).

SKILLS

Frontend	React · TypeScript · Vue 3 · TanStack Query · React Hook Form · React Native / Expo
APIs and databases	Feature development with Go/Ent and Express/PostgreSQL · OAuth · postMessage · Fetch streams
Quality and operations	Vitest · Playwright · MSW · ADB · GeckoView · OpenSearch · Sentry · GitHub Actions
AI tooling	Claude Code skills/plugins · MCP · BM25 retrieval · Python bot development and operation with AI coding tools

SIDE PROJECTS

Ssak · Habit Tracker

Built and continue to update a habit-tracking app for iOS and Android, including local storage, notifications, widgets, and backup/restore. I also promote it on Instagram.

App Store Google Play GitHub

Fox Walk

Launched an iOS and Android walking app where a fox grows as the user walks. Connected step counts, weather, and widgets, then fixed repeated popups and notification/storage failures after launch.

App Store Google Play

EDUCATION

Catholic Kwandong University

BA in Chinese Studies · Mar 2013 - Aug 2019

Hanghae99 Bootcamp

Sparta Coding Club · Jun-Sep 2021