Dongwoo Kwon | Frontend Developer

Email : hyeonggyugwon3@gmail.com
Phone : +82-10-8786-9260
GitHub : @kwondongwoo0424

A Developer Who Embraces Failure

Hello, I am **Dongwoo Kwon**, **a junior developer** who believes that failure is essential for growth.
I actively work with **frontend technologies** and enjoy developing with Agile methodologies and the MVP approach.
I have experience across the full development lifecycle: **service planning, design, development, deployment, and operation.**

I **continuously challenge** myself to become a better developer, learning and growing from failures.
When working on projects, I maintain **product ownership** while consistently learning and improving UX.
Like Nietzsche's quote, **"What does not kill me makes me stronger"**, I use failure as a **stepping stone** to become a better developer.

Skills

React | TypeScript | JavaScript | Next.js | HTML
styled-components | Tailwind CSS | SCSS | CSS
Storybook | TanStack Query | Zustand
GitHub | Figma | Vercel | GitHub Pages
Java | Spring Boot | Express.js | Node.js | pandas

Education

Daegu Software Meister High School

Mar 2024 - Feb 2027
(Expected Graduation)

Relevant Coursework

Python Programming / Web Programming Fundamentals / Database Programming / Data Structures / Artificial Intelligence Fundamentals / Software Engineering / Java Programming / Web Programming / Networking / Advanced Web Programming / Advanced Java Programming

Projects

Featured Project

Sync Desktop

AI-powered project management SaaS for collaboration between non-developers and developers

Team Project
7 members

React | TypeScript | Electron | styled-components | TanStack Query | Zustand

Desktop app development / UI design / Design system implementation / Landing page development / SSE-based real-time term interpretation feature development

Side Projects

DIA

GPA calculation service for Daegu Software Meister High School

Team Project
4 members

React | TypeScript | styled-components | React Context API

Grade calculation page development / Grade calculation algorithm modularization

ColorVerse

Web service for color palette recommendations

Personal Project
Solo

React | styled-components | Express.js | OpenAI API

Express.js server implementation / OpenAI API integration and prompt engineering

Flick

Digital currency-based real-time transaction and settlement platform for school festival booth operations

Team Project
5 members

Next.js | Tailwind CSS | TypeScript | pnpm

Admin page real-time booth ranking system development

External Project Contributions

Legacy web

"Learning Korean Culture and History through Games" web/app-based educational service

React | TypeScript | styled-components

OAuth button component refactoring (PR #75 merged) /
Identified and reported security vulnerability in hardcoded Kakao API keys

Starthub web

AI-powered personalized startup support information service

React | TypeScript | styled-components

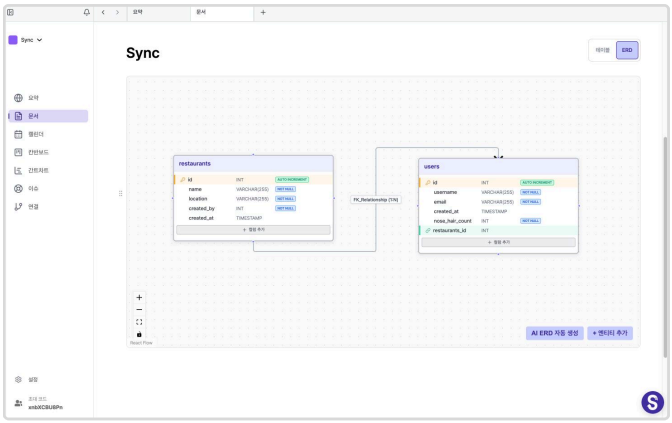
UX improvement with optimistic updates (PR #199 merged)

Featured Project

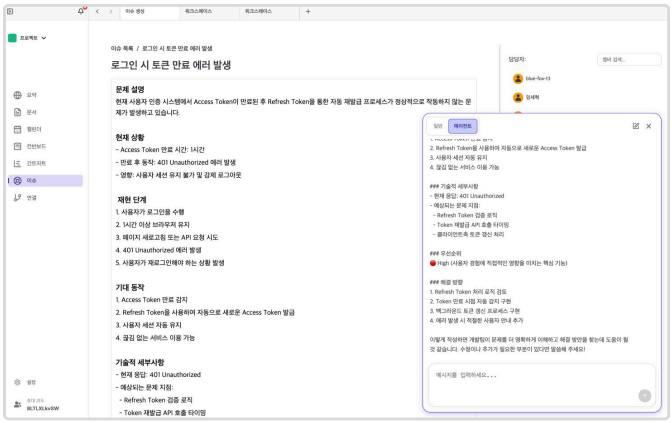
Sync Desktop

Mar 2025 - Oct 2025

AI-powered project management SaaS for collaboration between non-developers and developers



Document Page - Auto ERD generation based on functional specifications



DeepSync - Automatic issue content correction

Overview

An AI-powered project management desktop application designed to address communication issues and collaboration inefficiencies caused by the technical knowledge gap between non-developers (such as designers and product managers) and developers.
The core objective is to bridge the knowledge gap through AI assistance and provide a real-time collaborative environment for team members.

Key Challenges

- Non-developers struggle to articulate requirements due to limited technical knowledge
- Developers miss business context, leading to communication overhead
- Frequent misunderstandings, time waste, and repetitive work in collaboration

Solution

Enhance non-developers' technical understanding through AI while minimizing communication costs with real-time collaboration features.

Key Features

- Drag-based technical term interpretation for non-developers
- AI chatbot with project context awareness for personalized assistance
- Real-time collaborative
- Markdown documentation and AI-powered ERD generation from functional specifications

Team

7 members total
Frontend: 4 (including myself) | Backend: 3 | **Design**: 2 (including myself)
My Role: Frontend Developer & UI/UX Designer

Tech Stack

React

Chosen for rapid development of dynamic, stateful user interfaces

TypeScript

Ensures type safety essential for team collaboration and maintainability

Electron

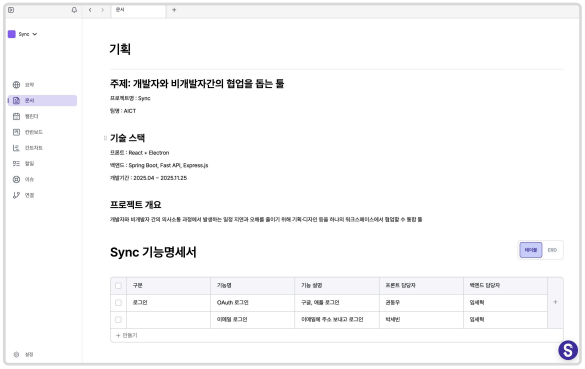
- Leverages familiar web technologies for desktop app development
- Provides cross-platform support for consistent UX across macOS and Windows
- Maintains browser-like development workflow for enhanced productivity

styled-components

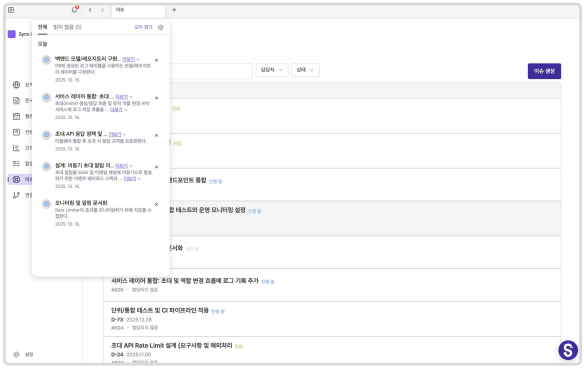
Manages component-level styling with consistency

Key Activities

UI/UX Design



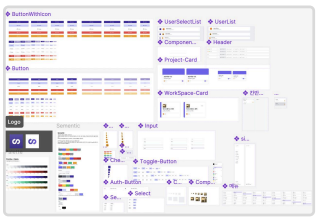
Document Page



Issue Page

Designed user-centric interfaces leveraging **Figma's Auto Layout** feature

Design System Development (sync-design-system)



Design System Implementation (Figma-based)

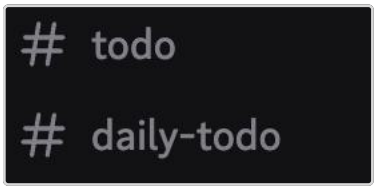
sds published on npmjs

Inconsistent UI components hindered maintainability and reusability.

- Extracted 15 reusable common components
- **Published** as **npm package** (@sync-design-system) for team-wide sharing
- Provided **component documentation** and **usage guidelines** via **Storybook**

→ **Improved development efficiency** and **UI consistency** through systematic component design

Agile Methodology Implementation



Daily todo tracking (via Discord)

Weekly sprint retrospectives

During the initial 2 months of our 7-month project, lack of clear milestones led to feature delays and difficulty tracking team progress.

- **Implemented weekly sprints**: Thursday demos and retrospectives, Friday sprint planning
- Managed backlog, sprint boards, and retrospectives via Notion
- Shared daily reminders and blocking issues through Discord

→ Gained experience improving schedule predictability and team productivity in long-term projects

Key Activities

UX Improvements

Reduced Issue Page Loading Time with React Query Caching

API re-fetching occurred on every navigation between issue list and detail pages
→ Average 2-second loading time with repeated requests increasing server load

- Implemented **React Query** for automatic issue **data caching**
- Set 2-minute staleTime: fresh data displays instantly without re-fetching (short staleTime used due to frequently changing issue data)

→ **Improved performance** and **user experience** simultaneously through caching strategy

Improved Task Check Responsiveness with Optimistic Updates

0.5-1s delay occurred while waiting for server response
when checking tasks as complete → degraded user experience

Implemented React Query **optimistic updates** for instant UI response

- **Reduced perceived response time** from **800ms to 0ms** on average
- Automatic rollback on network failure with error toast notifications

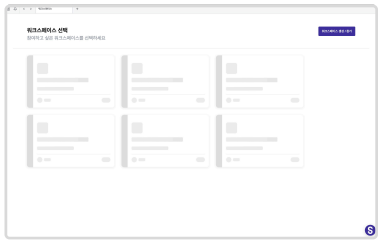
Task list with
optimistic updates applied

Removed Unnecessary Skeleton UI

Skeleton UI on workspace cards caused brief flash as it appeared and disappeared during loading

- Chrome DevTools showed average API response time of 180ms
- Skeleton UI vanished immediately after appearing,
  creating visual noise

→ Removed skeleton UI and rendered content immediately upon data arrival for a **clean, flicker-free loading experience**

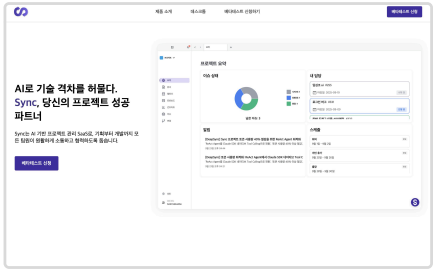


Workspace page
with skeleton UI applied

Beta Test Landing Page (sync introduce) Development

- Built beta signup page using GitHub Pages with custom domain
- Provided Windows/macOS installers via GitHub
  Releases-based deployment

→ Gained experience establishing and executing an **efficient beta distribution strategy** for product validation



Sync-introduce

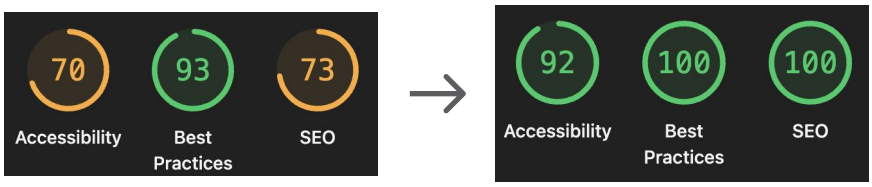
Key Activities

Improved Organic Search Traffic through SEO Optimization

Optimized landing page (Sync-introduce) SEO to improve search visibility and organic traffic for Sync Desktop

- **Enhanced search engine crawling** with sitemap.xml, robots.txt, and meta tag optimization
- Applied custom domain and achieved Lighthouse **SEO score of 100**
- Secured organic user acquisition channels through technical SEO implementation

→ Gained understanding of search engine crawling mechanisms and technical implementation methods



Lighthouse measurement results

Troubleshooting

Optimizing Excessive Re-rendering in SSE Streaming Responses

Problem

While implementing real-time AI word explanations via SSE when users drag text, the UI stuttered and lagged during response display.

Root Cause

Excessive re-rendering from setState calls on every chunk received

- n chunks received → n re-renders triggered
- Virtual DOM generation and diffing process executed per render

→ Performance degradation

```
onChunk: (chunk) => {
  // 현재 처리 중인 단어가 맞는지 확인
  if (currentWordRef.current !== word) {
    return; // 다른 단어의 응답이면 무시
  }

  // 매 청크마다 즉시 상태 업데이트
  setDefinition(prev => prev + chunk);
},
```

Original code

Solution Process

Considered two approaches: Throttling vs. Debouncing

1. Throttling
   - Pros: Consistent, smooth typing animation with regular render intervals
   - Cons: Unnecessary renders occur even when no new chunks arrive
2. Debouncing
   - Delays rendering while chunks arrive continuously
   - Executes single render only after no new chunks for set duration

While Throttling provided smoother animation, it had lingering performance issues from unnecessary renders. Debouncing, though slightly less smooth in typing effect, drastically reduced render count and most effectively resolved UI stuttering.

Final approach: Debouncing selected

Implementation

Debouncing + useRef Buffering Pattern

1. useRef as intermediate buffer → accumulates data without re-rendering
2. 50ms Debouncing → delays rendering during continuous chunk arrival
3. Cancel previous timer → setState called only once based on last chunk

→ Chunks accumulate in buffer in real-time, but screen updates only when no additional chunks arrive for 50ms

```
onChunk: (chunk) => {
  // 현재 처리 중인 단어가 맞는지 확인
  if (currentWordRef.current !== word) {
    return; // 다른 단어의 응답이면 무시
  }

  definitionRef.current += chunk;

  if (updateTimerRef.current) {
    clearTimeout(updateTimerRef.current);
  }

  updateTimerRef.current = setTimeout(() => {
    // 다시 한번 현재 단어 확인
    if (currentWordRef.current === word) {
      setDefinition(definitionRef.current);
    }
  }, 50);
},
```

Code with Debouncing applied

Results & Learnings

- Resolved UI stuttering by reducing setState call frequency
- Gained deep understanding of React rendering mechanisms
- Learned to clearly differentiate Throttling vs. Debouncing and apply appropriately based on context

Reflection & References

Reflection

Through this 7-month team project, I learned that teamwork doesn't happen automatically. We faced conflicts from different experiences and struggled when intentions weren't fully understood. However, open communication quickly resolved misunderstandings, and I learned that the willingness to understand each other matters more than problem-solving itself. This project taught me the value of growing together as a team, not just building features.

Additional Activities

FIX 2025 Korea ICT Convergence Expo Booth Operation	Oct 22-25, 2025
3rd Place, 2025 ITCE Project Competition	Oct 24, 2025

References

Sync-Desktop Source Code:	https://github.com/AICT-SYNC/sync-desktop
Sync Design System Source Code:	https://github.com/AICT-SYNC/sync-design-system
Sync Design System NPM:	https://npmjs.com/package/sync-design-system
Sync Introduce Source Code:	https://github.com/AICT-SYNC/sync-introduce
sync-saas Website (Service Ended):	https://sync-saas.com

Side Project

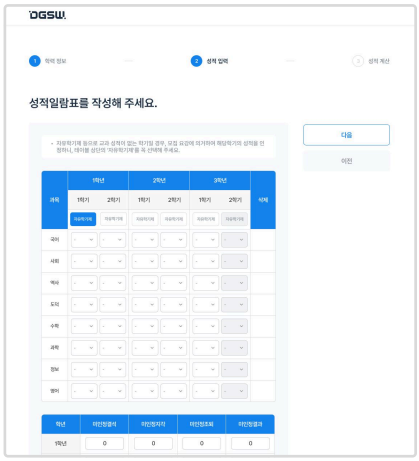
DIA

Aug 9 - Sep 9, 2025

GPA Calculation Service for Daegu Software Meister High School

Key Responsibilities

- GPA calculation page development
- Grade calculation algorithm modularization



DIA Grade Entry Interface

Tech Stack

React | TypeScript | styled-components

References

Frontend Source Code: https://github.com/EntryCNS/DIA
Live Website: https://trydgsw.com

Overview

A simple, intuitive web service that calculates total scores when students enter their academic grades, based on the grading criteria for Daegu Software Meister High School's first-round admissions.

Team

4 members total
Frontend: 4 (including myself)

Key Learnings

Global State Management with React Context API

Developed 3-step form flow: student type selection → grade input → result confirmation

- Props passed through 5-6 components (5-level depth)
- Required modifying all 6 files when adding new data fields

→ **Centralized global state management** with **React Context API**

- Real-time sync: shared state across multiple components
- **Eliminated Props Drilling**: 5-level passing → direct access

→ Built systematic state management architecture for improved maintainability

```
Before (Props Drilling)

[App] - grades, setGrades, freeSem, attendance... (12개 props)
  ↓
[Router] - (12개 props 그대로 전달)
  ↓
[StudentWritePage] - (12개 props 그대로 전달)
  ↓
[Body] props만 전달, 사용 안함 - (12개 props 그대로 전달)
  ↓
[WriteGrade] - grades, setGrades 사용
  ↓
[WriteGradeListItem] - grades 사용
```

→

```
After (Context API)

[App]
  └ [ScoreProvider] 전역 상태
        ├ [StudentWritePage] → useScore()
        ├ [WriteGrade] → useScore()
        └ [WriteGradeListItem] → useScore()
```

Why React Context API over Redux/Zustand

- Project requirements were simple ("share form data across pages"), not requiring Redux's action/reducer structure or Zustand's selector-based architecture
- Low state change frequency meant Redux/Zustand's granular state separation wasn't needed; Context API provided sufficient performance

Side Project

ColorVerse

Color Palette Recommendation Service

Jun 21-25, 2025

Key Responsibilities

- Express.js server implementation

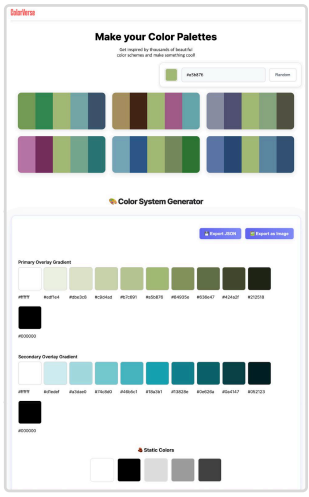
Tech Stack

React | styled-components | Express.js | OpenAI API

References

Source Code: https://github.com/kwondongwoo0424/ColorVerse

Notion Docs: https://buly.kr/GvoBKj0



Main interface

Overview

A web service that helps designers and developers quickly establish consistent color systems and discover harmonious color combinations with AI assistance.

Team

Solo project

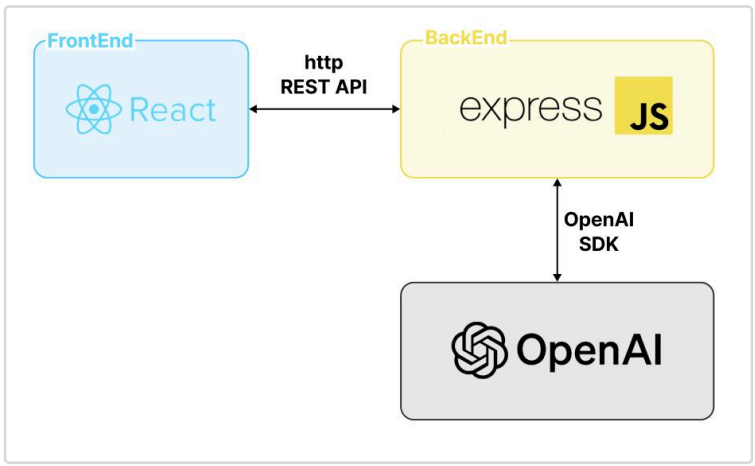
Key Learnings

RESTful API Server with Express.js

- Integrated OpenAI API via /api/chat POST endpoint
- Implemented frontend-backend communication with CORS configuration

OpenAI API Integration & Prompt Engineering

- Integrated GPT-3.5-turbo model using official OpenAI Node.js SDK (openai)
- Implemented complete API flow: authentication → request composition → response handling



Side Project

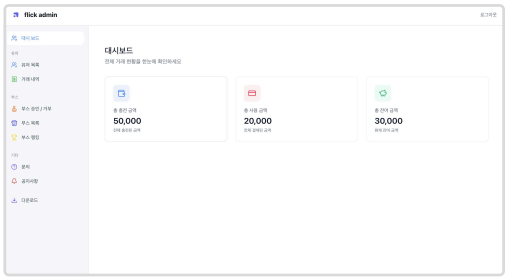
Flick

May 5-20, 2025

Digital Currency-Based Real-Time Transaction and Settlement Platform for School Festival Booth Operations

Key Responsibilities

- Admin page development



Flick Admin Dashboard Page

Tech Stack

Next.js | tailwind CSS | TypeScript

References

Web Source Code: https://github.com/EntryCNS/flick-web

Overview

Flick is a point-based transaction service designed for payments at on-campus festival booths.
The previously developed v1, created by senior students, had several usability issues related to its QR code payment flow.
To address these limitations and enhance the overall user experience, we newly developed Flick v2.

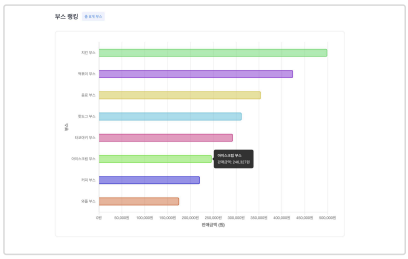
Team

5 members total
Frontend: 2 (including myself) | Backend: 2 | Full-Stack: 1
My Role: Frontend Developer

Web Development

Developed a real-time booth ranking system

- Built a real-time revenue ranking visualization using Chart.js and WebSocket
- Implemented WebSocket reconnection logic to ensure stable and reliable real-time data communication



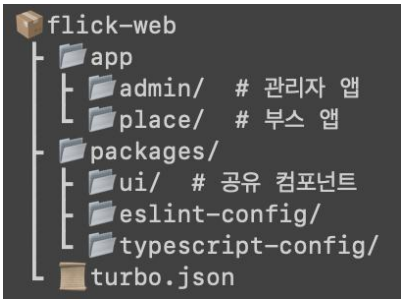
Admin Ranking Page

Key Learnings

Codebase Integration and Management Efficiency

Adopted a Turborepo- and pnpm Workspace–based **monorepo**

- @repo/ui: Centralized shared components to eliminate code duplication
- pnpm hard links: Reduced node_modules storage usage
- workspace:* protocol: Automated dependency management across packages



Flick-web Project Structure

→ Gained hands-on experience achieving both productivity and consistency in a multi-project environment

External Project Contributions

Legacy (legacy-web)

Overview & Project Introduction

Legacy-web is a web and mobile–based learning service developed with the goal of teaching Korean culture and history through games.
The project involved a total of seven members, with the web team consisting of two developers. As an external contributor, I identified areas for improvement in code quality and security and contributed accordingly.

Contribution

Refactoring of the OAuth button component

Issues Identified

- Google and Apple login buttons were componentized, while the Kakao login button was implemented directly within a page
- This caused inconsistency in the OAuth button structure and reduced maintainability and reusability

Improvements Implemented

- Extracted the Kakao login button into a dedicated component
- Unified the component structure to match existing OAuth login buttons

Results

- Achieved structural consistency across OAuth login buttons
- Improved UI component reusability
- Enhanced maintainability and overall developer experience (DX)

References

Pull Request: https://github.com/TeamDetail/legacy-web/pull/75
Repository: https://github.com/TeamDetail/legacy-web

External Project Contributions

Starthub (starthub-web)

Overview & Project Introduction

StartHub is an AI-driven service that provides personalized startup support information.
The web application is built with React and TypeScript and is maintained as
a public GitHub repository.
While using the service, I identified a user experience issue and contributed as an external contributor to address it.

Contribution

Applied optimistic updates to the “Like” feature on the announcement detail page

Issues Identified

- The UI did not respond until the server returned a response when liking or unliking an announcement
- Users could not easily confirm whether their click was successfully processed, leading to poor UX
- The perceived delay became more severe in unstable network conditions

Improvements Implemented

- Utilized React Query’s useMutation
- Applied optimistic updates to immediately reflect UI state changes before server responses
- Implemented rollback logic on request failure to ensure reliability

Results

- Immediate UI feedback on click, significantly improving user experience
- Smooth and natural interactions even under unstable network conditions
- Delivered a robust like feature that handles both success and failure scenarios

Impact

This improvement meaningfully enhanced the user experience on the announcement detail page by providing instant feedback after user interactions, greatly increasing perceived responsiveness.
It was a valuable experience in identifying and solving real user-centric problems, contributing directly to overall service quality.

References

Pull Request: https://github.com/JinInSaDaeCheonMyeong/starthub-web/pull/199
Repository: https://github.com/JinInSaDaeCheonMyeong/starthub-web

Certifications & Others

23rd TOPCIT (Test of Practical Competency in ICT) - 527 points (Level 3)	May 24, 2025
TOEIC Bridge - 76	Dec 23, 2024

Awards

3rd Place, 2025 ITCE Project Competition	Oct 24, 2025
Academic Excellence Award in Physical Education	Jul 22, 2025

Activities

SW Meister High School Joint Hackathon Participant	Nov 5-7, 2025
FIX 2025 Korea ICT Convergence Expo - Booth Operator	Oct 22-25, 2025
School Software Vibe Coding Hackathon Participant	Jul 15-16, 2025
Narshar Project Participant - Sync Desktop	Mar 17 - Oct 25, 2025
1:1 Native English Online Program	May 13 - Dec 23, 2025 (Ongoing)
Library Committee Head	Dec 2024 - Present
Admissions Interview Volunteer	Oct 25, 2024
Narshar Project Participant	Aug 20 - Dec 26, 2024
School Software Hackathon Participant	Jul 17-18, 2024
1:1 Native English Online Program	May 8, 2024 - Jan 8, 2025
Coding Test Competition Participant	Apr 17, 2024
Smarteen App Challenge Participant	Apr 2, 2024
CNS Programming Club Member	Mar 14, 2024 - Present

Thank you for taking the time to review my portfolio.
I'm still learning and growing, but I'm committed to becoming a better developer every single day.