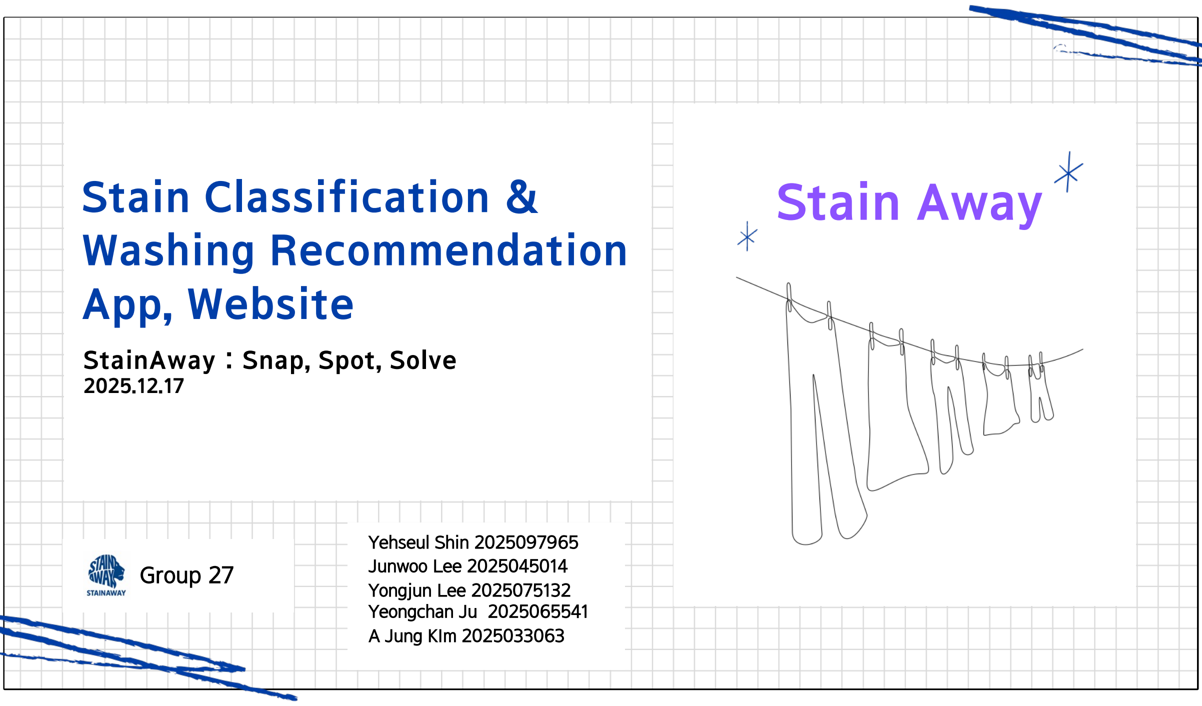


Stain Classification & Washing Recommendation App, Website

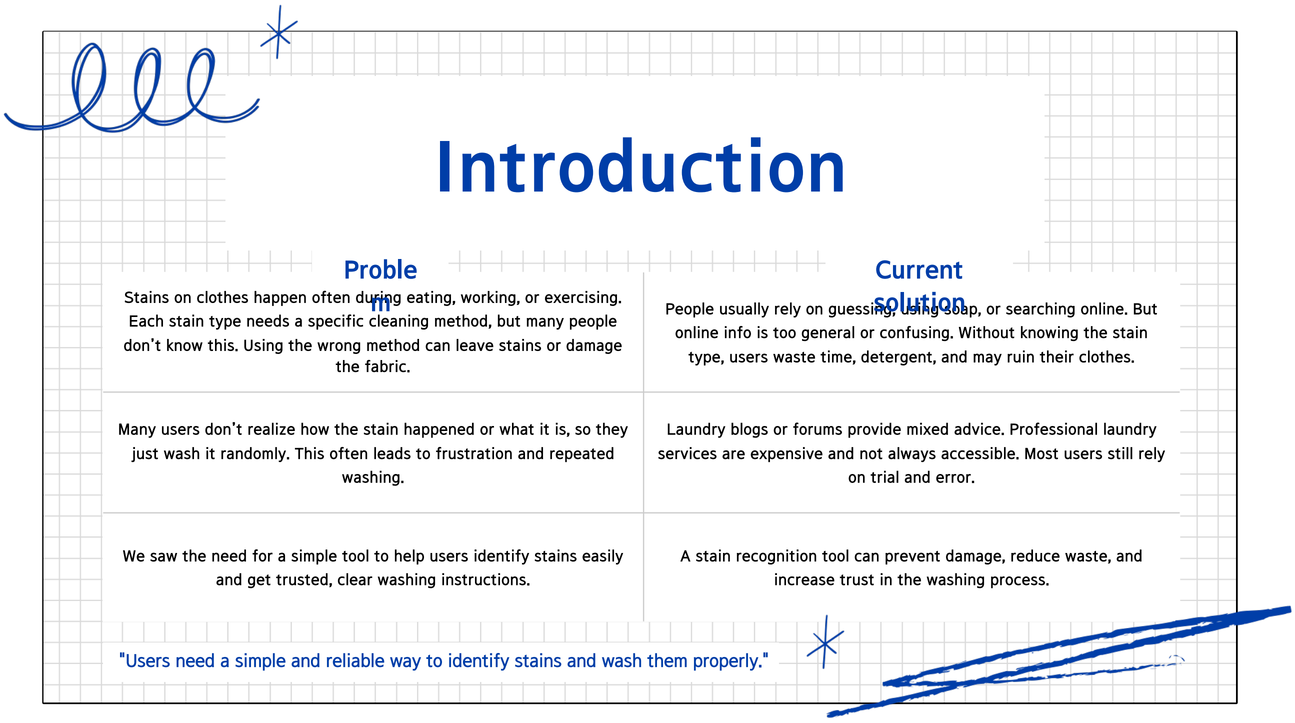
StainAway : Snap, Spot, Solve
2025.12.17

Stain Away

Group 27

Yehseul Shin 2025097965
Junwoo Lee 2025045014
Yongjun Lee 2025075132
Yeongchan Ju 2025065541
A Jung KIm 2025033063

We are Group 27, and our project is called *StainAway*. It's a smart laundry assistant that helps you spot stains and solve them easily.

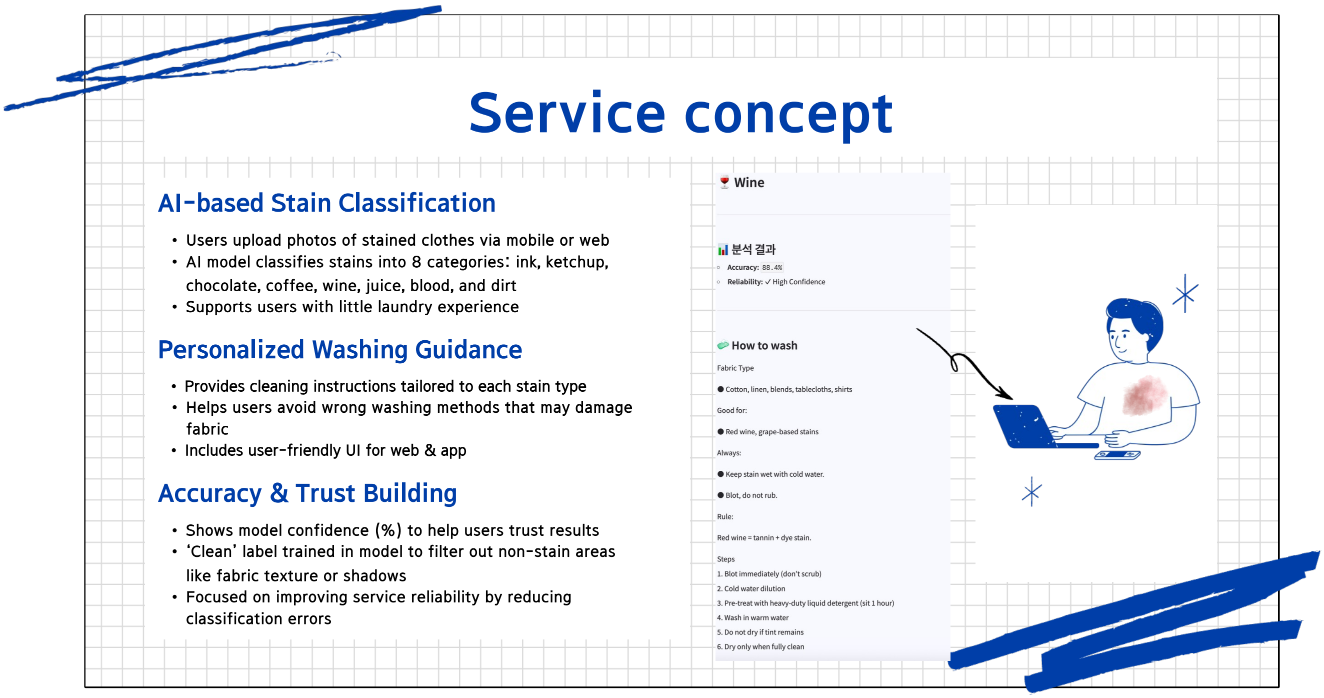


Introduction

Proble m	Current solution
Stains on clothes happen often during eating, working, or exercising. Each stain type needs a specific cleaning method, but many people don't know this. Using the wrong method can leave stains or damage the fabric.	People usually rely on guessing, using soap, or searching online. But online info is too general or confusing. Without knowing the stain type, users waste time, detergent, and may ruin their clothes.
Many users don't realize how the stain happened or what it is, so they just wash it randomly. This often leads to frustration and repeated washing.	Laundry blogs or forums provide mixed advice. Professional laundry services are expensive and not always accessible. Most users still rely on trial and error.
We saw the need for a simple tool to help users identify stains easily and get trusted, clear washing instructions.	A stain recognition tool can prevent damage, reduce waste, and increase trust in the washing process.

"Users need a simple and reliable way to identify stains and wash them properly."

People often wash clothes the wrong way because they don't know the stain type. Our goal is to offer simple, trusted guidance with the help of AI.



Service concept

AI-based Stain Classification

- Users upload photos of stained clothes via mobile or web
- AI model classifies stains into 8 categories: ink, ketchup, chocolate, coffee, wine, juice, blood, and dirt
- Supports users with little laundry experience

Personalized Washing Guidance

- Provides cleaning instructions tailored to each stain type
- Helps users avoid wrong washing methods that may damage fabric
- Includes user-friendly UI for web & app

Accuracy & Trust Building

- Shows model confidence (%) to help users trust results
- 'Clean' label trained in model to filter out non-stain areas like fabric texture or shadows
- Focused on improving service reliability by reducing classification errors

🍷 Wine

📊 분석 결과

- Accuracy: `88.4%`
- Reliability: ✓ High Confidence

How to wash

Fabric Type

● Cotton, linen, blends, tablecloths, shirts

Good for:

● Red wine, grape-based stains

Always:

● Keep stain wet with cold water.

● Blot, do not rub.

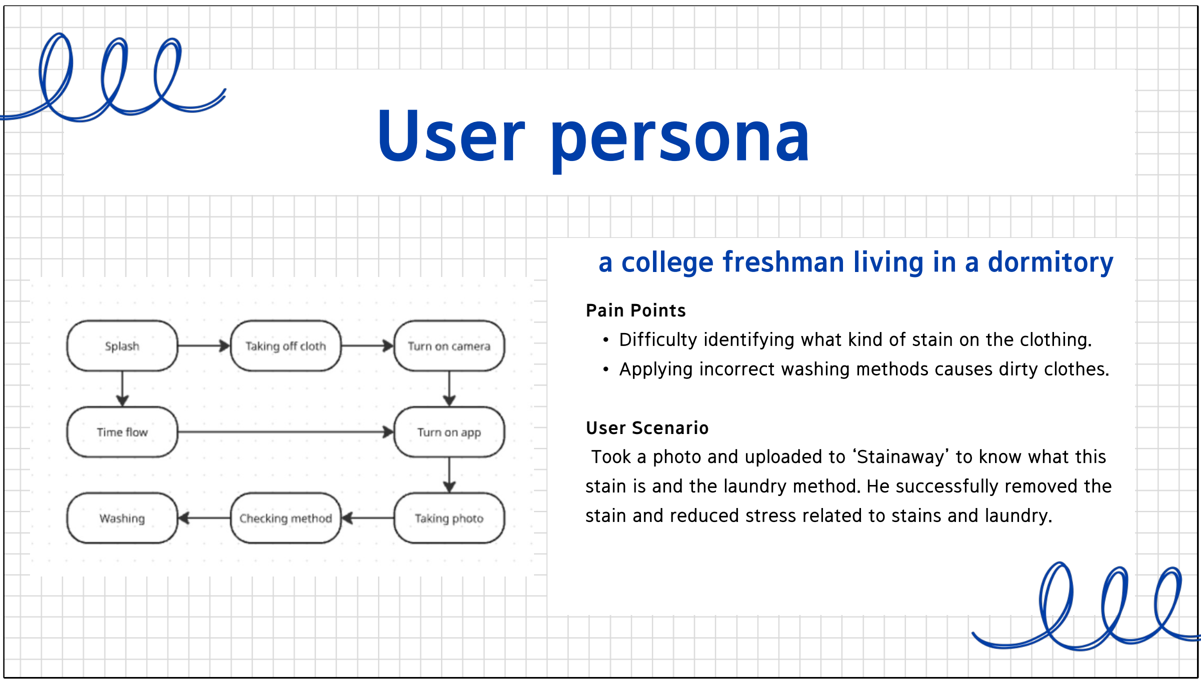
Rule:

Red wine = tannin + dye stain.

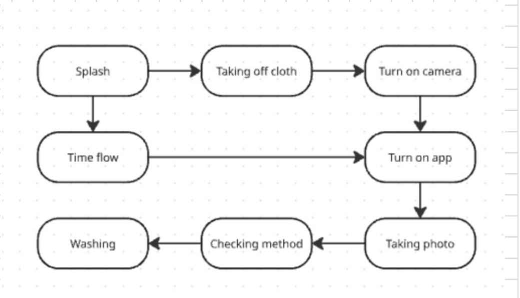
Steps

1. Blot immediately (don't scrub)
2. Cold water dilution
3. Pre-treat with heavy-duty liquid detergent (sit 1 hour)
4. Wash in warm water
5. Do not dry if tint remains
6. Dry only when fully clean

Our service classifies stains using AI into 8 types, like ink or ketchup. Based on the result, it gives step-by-step washing instructions. We also show the model's confidence score to help users trust the result.



User persona



a college freshman living in a dormitory

Pain Points

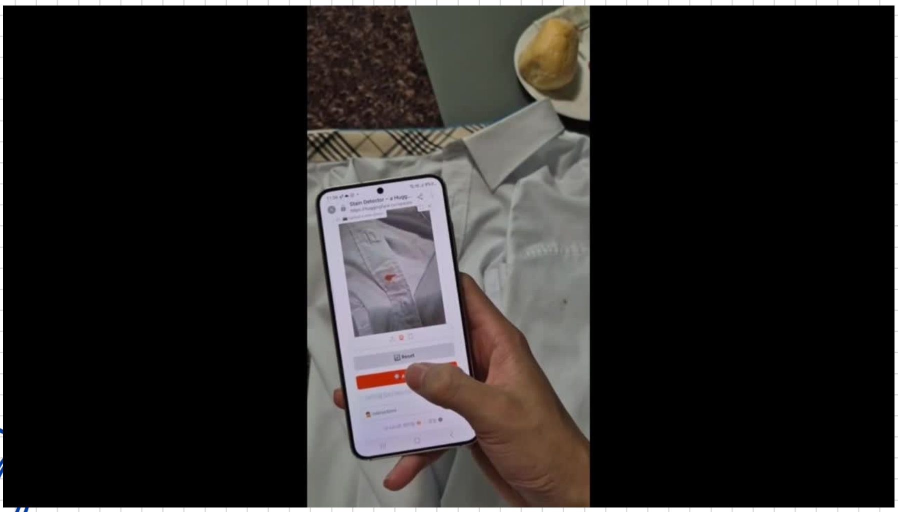
- Difficulty identifying what kind of stain on the clothing.
- Applying incorrect washing methods causes dirty clothes.

User Scenario

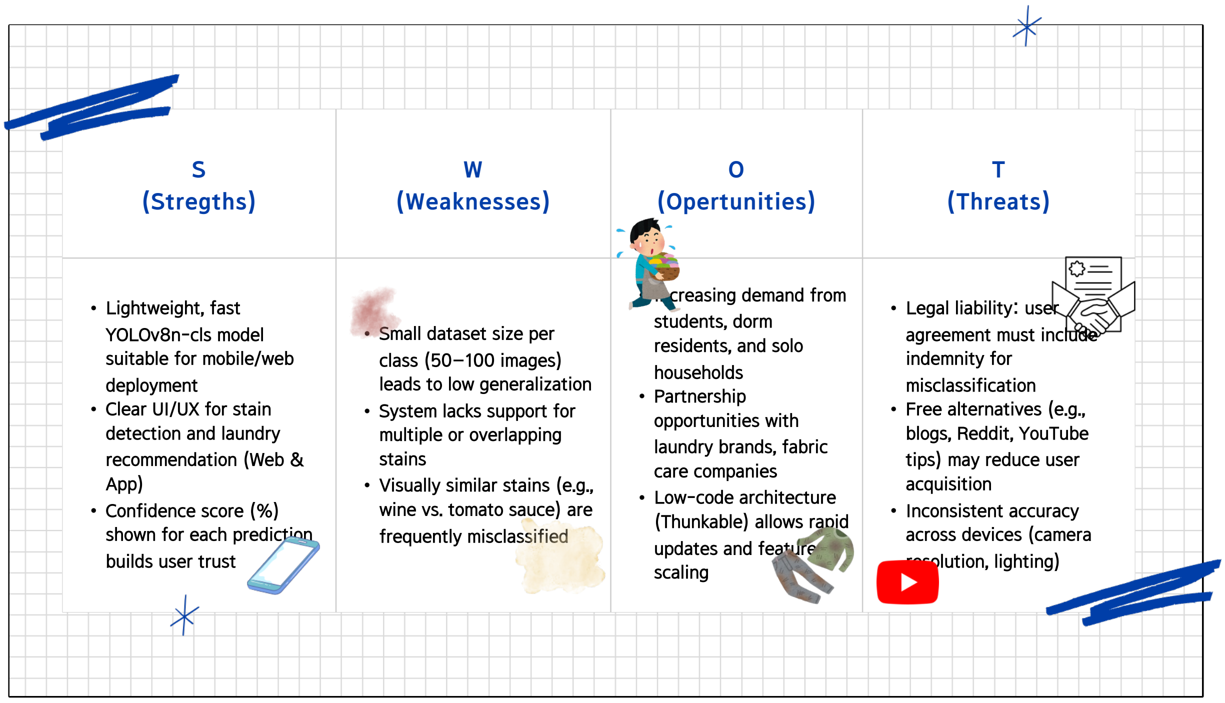
Took a photo and uploaded to 'Stainaway' to know what this stain is and the laundry method. He successfully removed the stain and reduced stress related to stains and laundry.

Our main user is a college freshman who has no laundry experience. They struggle to identify stains and often damage clothes. With StainAway, they can just take a photo, get the answer, and wash correctly.

Useage video https://youtu.be/Chjz9SnvZjs

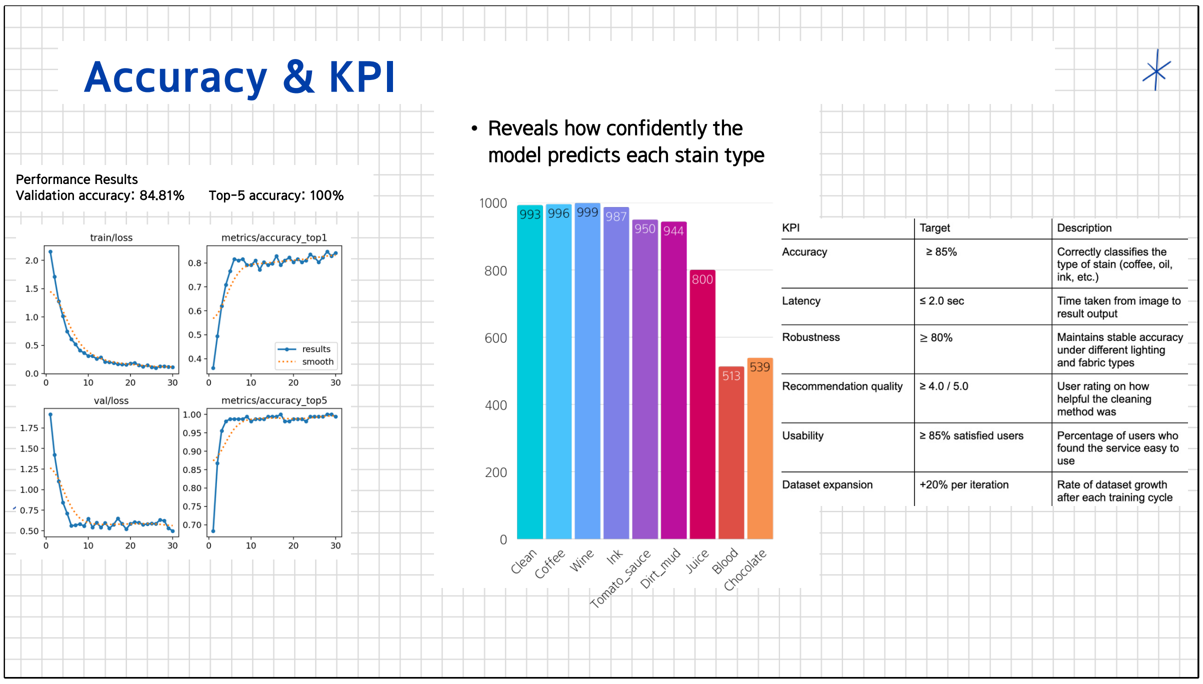


Here's how it works. Users take a photo of the stained area, upload it to the app, and get instant classification and cleaning tips. It's simple, fast, and effective.



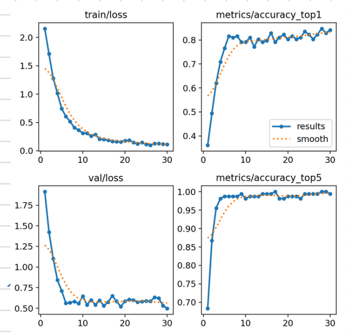
S (Stregths)	W (Weaknesses)	O (Opertunities)	T (Threats)
• Lightweight, fast YOLOv8n-cls model suitable for mobile/web deployment • Clear UI/UX for stain detection and laundry recommendation (Web & App) • Confidence score (%) shown for each prediction builds user trust	• Small dataset size per class (50–100 images) leads to low generalization • System lacks support for multiple or overlapping stains • Visually similar stains (e.g., wine vs. tomato sauce) are frequently misclassified	creasing demand from students, dorm residents, and solo households • Partnership opportunities with laundry brands, fabric care companies • Low-code architecture (Thunkable) allows rapid updates and feature scaling	• Legal liability: user agreement must include indemnity for misclassification • Free alternatives (e.g., blogs, Reddit, YouTube tips) may reduce user acquisition • Inconsistent accuracy across devices (camera resolution, lighting)

Our strengths include a fast model and clean UI. Weaknesses are mainly due to small datasets. Opportunities come from solo households, and threats include legal issues and free alternatives like YouTube or blogs.

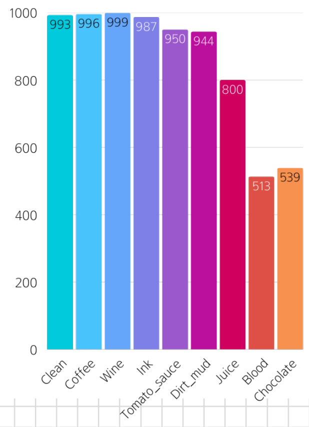


Accuracy & KPI

Performance Results
Validation accuracy: 84.81%   Top-5 accuracy: 100%

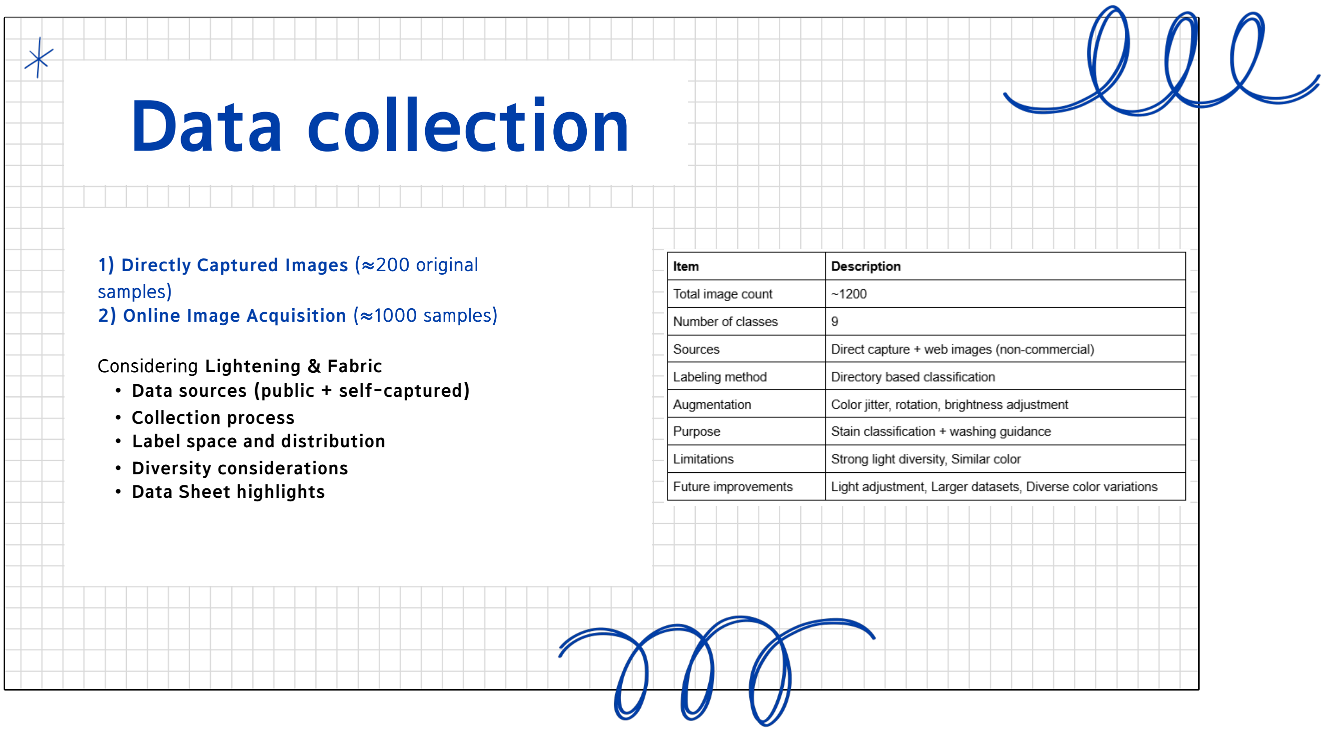


- Reveals how confidently the model predicts each stain type



KPI	Target	Description
Accuracy	≥ 85%	Correctly classifies the type of stain (coffee, oil, ink, etc.)
Latency	≤ 2.0 sec	Time taken from image to result output
Robustness	≥ 80%	Maintains stable accuracy under different lighting and fabric types
Recommendation quality	≥ 4.0 / 5.0	User rating on how helpful the cleaning method was
Usability	≥ 85% satisfied users	Percentage of users who found the service easy to use
Dataset expansion	+20% per iteration	Rate of dataset growth after each training cycle

Our model reaches 84.8% accuracy and 100% top-5 accuracy. KPIs include robustness, latency, and user satisfaction—all aiming to improve reliability and guide quality.



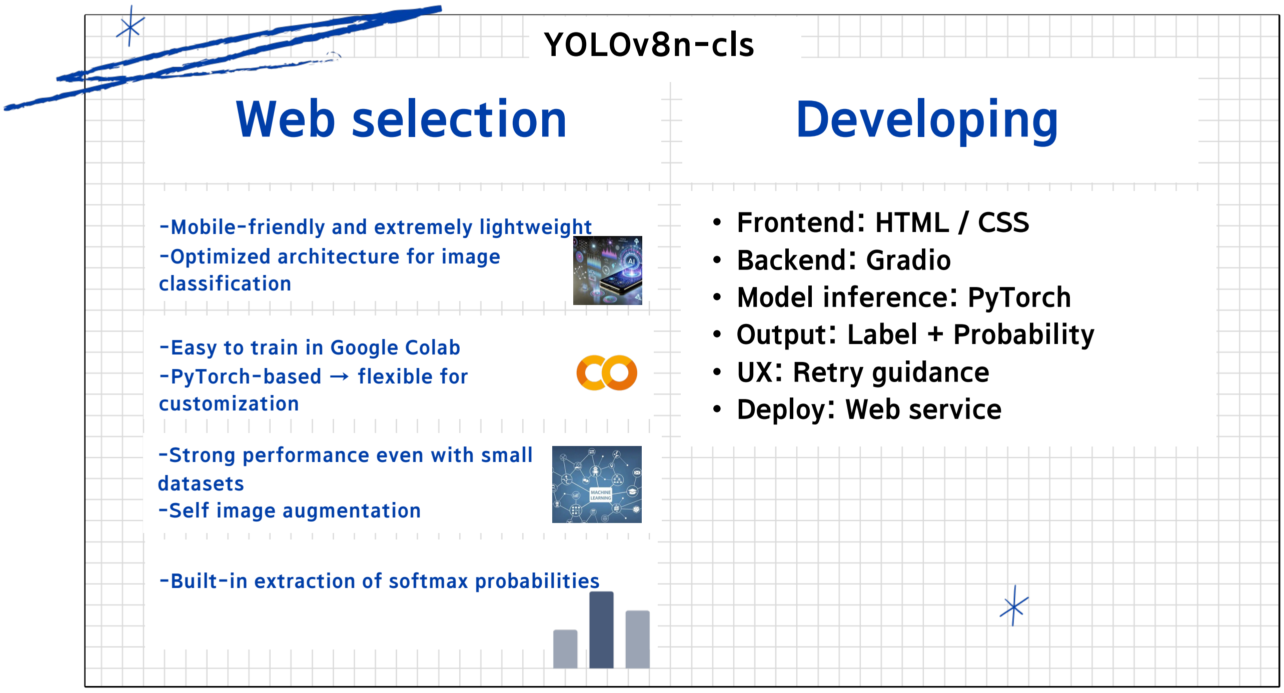
Data collection

1) **Directly Captured Images** (≈200 original samples)
2) **Online Image Acquisition** (≈1000 samples)

Considering **Lightening & Fabric**
- **Data sources (public + self-captured)**
- **Collection process**
- **Label space and distribution**
- **Diversity considerations**
- **Data Sheet highlights**

Item	**Description**
Total image count	~1200
Number of classes	9
Sources	Direct capture + web images (non-commercial)
Labeling method	Directory based classification
Augmentation	Color jitter, rotation, brightness adjustment
Purpose	Stain classification + washing guidance
Limitations	Strong light diversity, Similar color
Future improvements	Light adjustment, Larger datasets, Diverse color variations

We collected about 1,200 images (200 self-captured, 1,000 online), considering light and fabric diversity, using directory-based labeling and augmentation to improve stain classification. We plan to expand the dataset with more diverse and balanced samples.



YOLOv8n-cls

Web selection

-Mobile-friendly and extremely lightweight
-Optimized architecture for image classification

-Easy to train in Google Colab
-PyTorch-based → flexible for customization

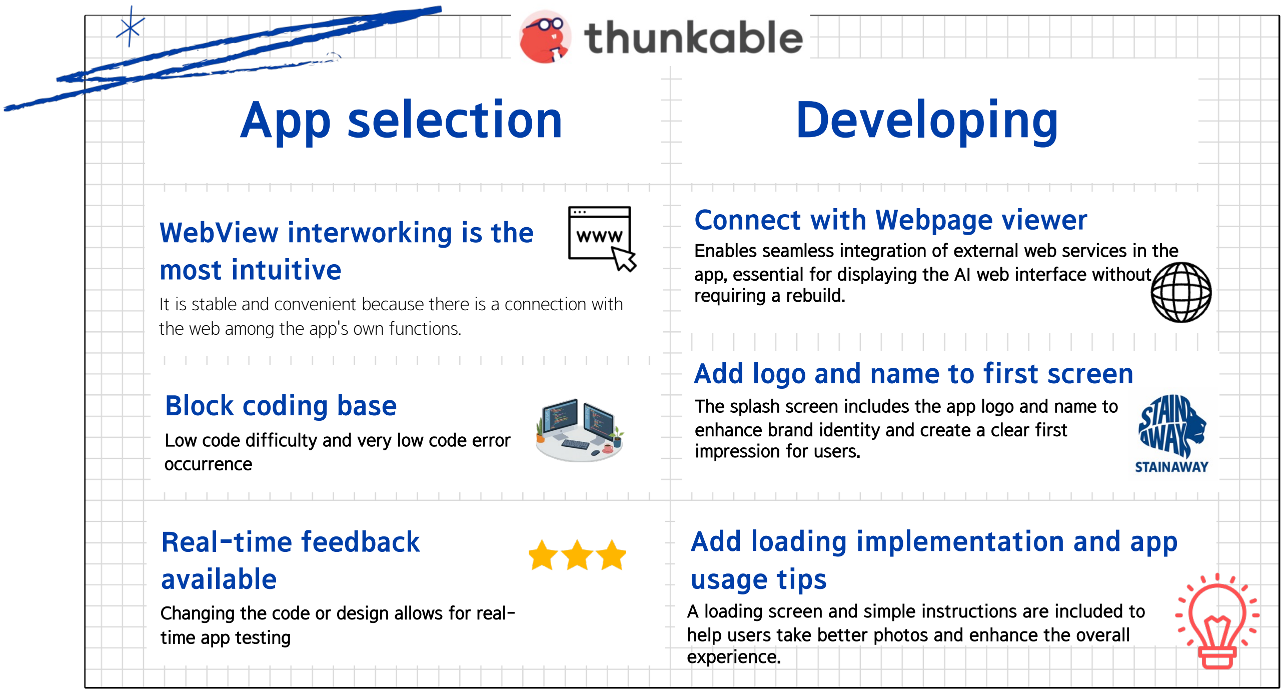
-Strong performance even with small datasets
-Self image augmentation

-Built-in extraction of softmax probabilities

Developing

- Frontend: HTML / CSS
- Backend: Gradio
- Model inference: PyTorch
- Output: Label + Probability
- UX: Retry guidance
- Deploy: Web service

We used YOLOv8n-cls for fast, lightweight image classification, trained in Google Colab with PyTorch, and served via Gradio with a simple HTML/CSS interface for labels and probabilities.



thunkable

App selection

WebView interworking is the most intuitive

It is stable and convenient because there is a connection with the web among the app's own functions.

Block coding base

Low code difficulty and very low code error occurrence

Real-time feedback available

Changing the code or design allows for real-time app testing

Developing

Connect with Webpage viewer

Enables seamless integration of external web services in the app, essential for displaying the AI web interface without requiring a rebuild.

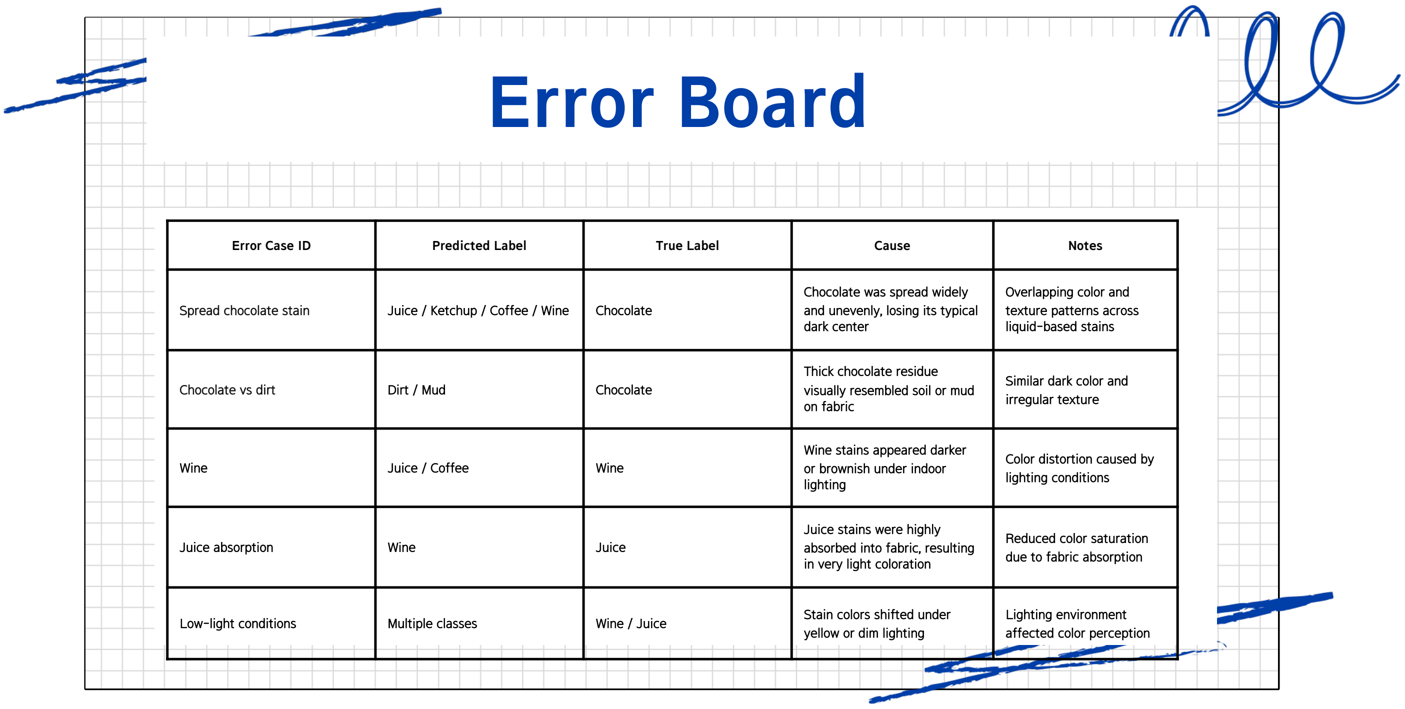
Add logo and name to first screen

The splash screen includes the app logo and name to enhance brand identity and create a clear first impression for users.

Add loading implementation and app usage tips

A loading screen and simple instructions are included to help users take better photos and enhance the overall experience.

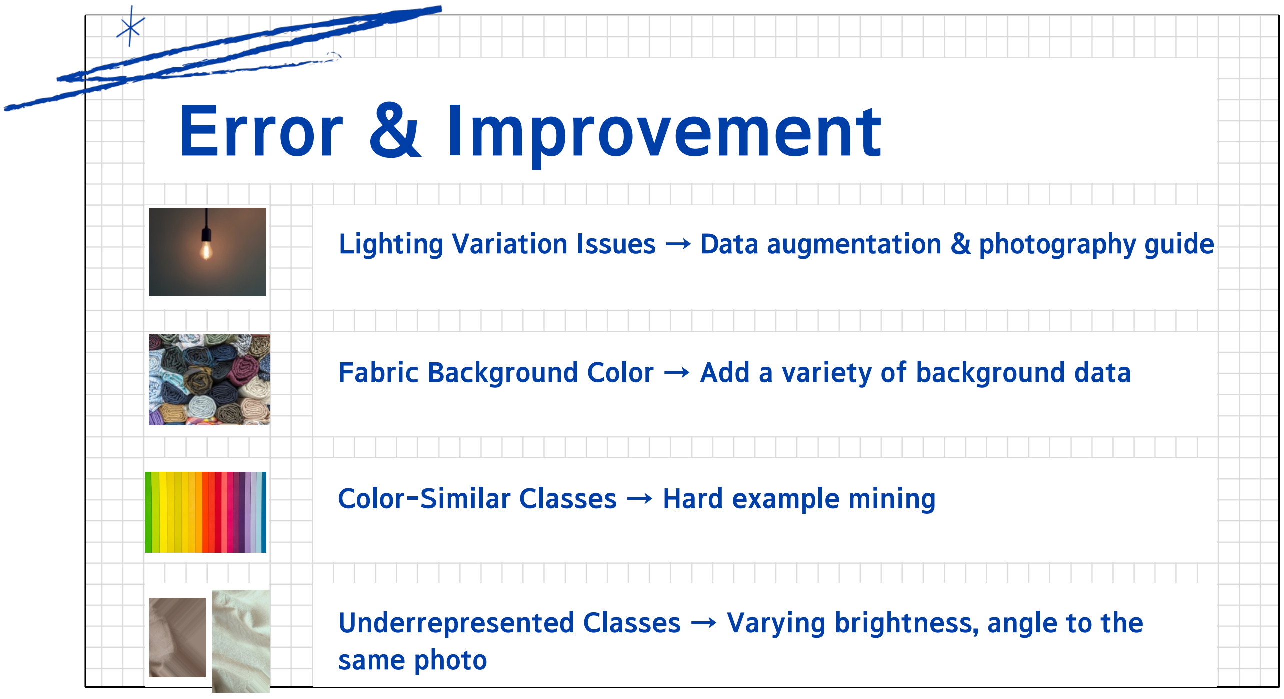
We used Thunkable for its intuitive WebView and block coding, enabling real-time feedback and seamless connection to our AI webpage, plus a logo, loading screen, and tips for a smoother user experience.



Error Board

Error Case ID	Predicted Label	True Label	Cause	Notes
Spread chocolate stain	Juice / Ketchup / Coffee / Wine	Chocolate	Chocolate was spread widely and unevenly, losing its typical dark center	Overlapping color and texture patterns across liquid-based stains
Chocolate vs dirt	Dirt / Mud	Chocolate	Thick chocolate residue visually resembled soil or mud on fabric	Similar dark color and irregular texture
Wine	Juice / Coffee	Wine	Wine stains appeared darker or brownish under indoor lighting	Color distortion caused by lighting conditions
Juice absorption	Wine	Juice	Juice stains were highly absorbed into fabric, resulting in very light coloration	Reduced color saturation due to fabric absorption
Low-light conditions	Multiple classes	Wine / Juice	Stain colors shifted under yellow or dim lighting	Lighting environment affected color perception

Main errors came from similar color stains like chocolate vs mud, or wine vs juice.
Lighting and fabric texture also caused misclassifications in low-light and absorbed stains



Error & Improvement

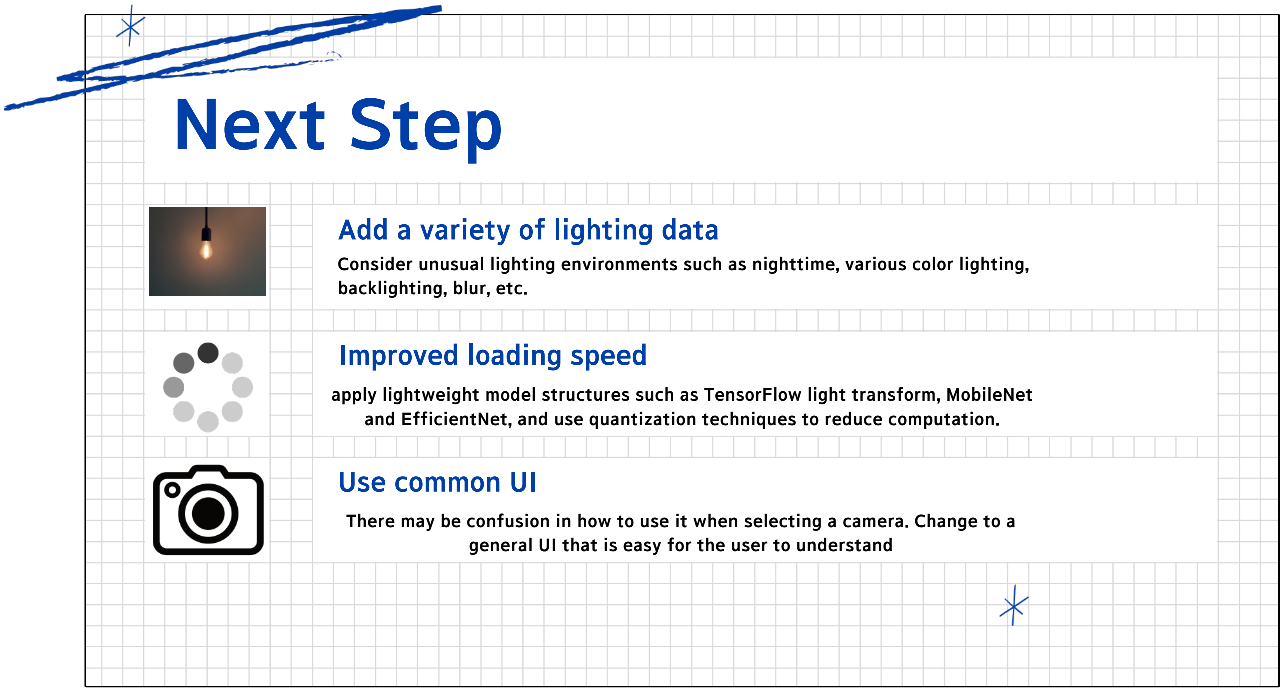
Lighting Variation Issues → Data augmentation & photography guide

Fabric Background Color → Add a variety of background data

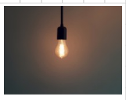
Color-Similar Classes → Hard example mining

Underrepresented Classes → Varying brightness, angle to the same photo

We faced issues with lighting, fabric backgrounds, and similar stain colors. To fix this, we'll improve data variety, use hard examples, and balance underrepresented classes.



Next Step



Add a variety of lighting data

Consider unusual lighting environments such as nighttime, various color lighting, backlighting, blur, etc.

Improved loading speed

apply lightweight model structures such as TensorFlow light transform, MobileNet and EfficientNet, and use quantization techniques to reduce computation.

Use common UI

There may be confusion in how to use it when selecting a camera. Change to a general UI that is easy for the user to understand

Next, we'll collect more lighting data, speed up the model using lighter architectures, and improve the camera UI for easier use.