

GLOWLAB: AN AI-POWERED SYSTEM FOR PERSONALIZED SKINCARE ANALYSIS AND TRACKING

Laura L. Zimny¹, Andy Liang²

¹Northwood High School, 4515 Portola Parkway, Irvine, CA 92620

²California State University Los Angeles, 5151 State University Dr, Los Angeles, CA 90032

ABSTRACT

This paper addresses the technical challenge of bridging the gap between unstructured product data and personalized skincare guidance. The proposed solution, GlowLab, is an integrated mobile platform that synthesizes diverse data streams to optimize skincare routines efficiently and safely. The system is engineered around three core components: a persistent Data Service for tracking longitudinal skin health, a computer-vision enabled Product Analysis System that parses ingredient lists via the OpenAI API, and a Recommendation Engine that dynamically cross-references these inputs against user sensitivities[1].

Unlike systems relying on invasive biometric surveillance, this project addresses privacy and design challenges by utilizing secure local storage and user-reported data. Technical testing of the image recognition module demonstrated a 100% success rate across 20 test cases, confirming the system's ability to correctly parse complex labels into structured data [2]. The result is a robust, responsive application that transforms static ingredient data into dynamic, actionable health insights, providing immediate value for everyday consumers seeking safe, personalized guidance.

KEYWORDS

Skincare AI, Ingredient Analysis, Personalization, Mobile App

1. INTRODUCTION

The idea for GlowLab was inspired by the widespread challenge many people face in managing acne, breakouts, and overall skin health [3]. Acne is one of the most common skin conditions in the world, affecting millions of people of all ages, with especially high rates among teenagers and young adults. Research shows that the average annual percent change in prevalence was 0.43%, showing a steady upward trend (Zhu et al.). Acne can have a profound impact on confidence, mood, and overall quality of life. While it primarily affects adolescents, young adults, and individuals with sensitive or acne-prone skin, anyone can experience its effects regardless of gender or background [4]. The consequences go beyond appearance, and acne can cause scarring, uneven skin tone, and emotional stress, particularly when skincare products fail or worsen the condition. Managing skin health is often overwhelming, with long ingredient lists, misleading information, and trial-and-error routines that leave people unsure which products are safe and effective. Addressing this problem is important because accessible, personalized guidance can reduce frustration, prevent unnecessary irritation, and empower individuals to care for their skin with confidence. GlowLab aims to provide this support through data-driven tools that help users

track their skin, understand its needs, and make informed decisions about their skincare routine, ultimately promoting healthier skin and greater self-assurance [5].

Section 5 reviewed three different approaches to using AI in beauty and skincare, each with distinct goals and limitations. First, Pambudi and Dwinata used surveys to study customer intention to use AI, analyzing factors like comfort, social influence, and technology readiness. Their goal was to understand attitudes, but the limitation is that it doesn't produce a functional tool for immediate user benefit. Second, Lambert and Grzybowski reviewed AI in dermatology, focusing on diagnosing skin cancer and analyzing cosmetics. While aiming to improve medical accuracy, their systems carry safety risks and require doctor verification, making them unsuitable for casual consumer use. Third, Hash et al. examined AI-driven personalized skincare regimens using computer vision, genetics, and environmental data [6]. This methodology creates highly tailored routines but relies on invasive biometric collection, raising privacy concerns. My project improves on all three by offering a practical, consumer-friendly AI system that provides personalized product recommendations safely, instantly, and without requiring sensitive data or medical oversight.

GlowLab is built around a simple idea that most people want healthier skin, but figuring out what actually works for them can feel overwhelming. The concept behind the app is to give users a personalized place where they can understand their skin, track changes, and make confident decisions about the products they use. I decided to build it this way because skincare often becomes a frustrating cycle of guessing, testing, and hoping a product will finally help. GlowLab solves this problem by combining product analysis, skin check-ins, and breakout mapping into one connected system that learns from the user over time. When someone scans or searches for a product, the app reviews the ingredients, identifies possible irritants, and presents clear and helpful information that is easy to understand. Regular check-ins and breakout tracking allow users to notice patterns and understand what their skin responds to. GlowLab is unique because it focuses on personal history instead of general advice that may not apply to everyone. Many skincare tools only scan products or only track routines, but GlowLab ties both together to create a more complete picture. This approach makes it more effective than other options because it offers guidance that is specific to each individual, helping users make safer and more confident skincare choices.

In section 4, we designed an experiment to test the accuracy of GlowLab's AI in recognizing products and matching them to a user's skin profile. The goal was to determine whether the system could correctly identify products from photos or typed names and provide recommendations suited to individual users. To set up the experiment, we created a set of 20 diverse inputs, including clear product images and typed product names, representing a variety of skin types, routines, and product formats. For each input, we recorded the expected output and compared it to the AI's response. Our most significant finding was that the AI produced correct results for all tested inputs, accurately identifying products and matching them to the user's profile. These results occurred because the AI leverages OpenAI's extensive database of ingredient knowledge and was tested using clear, representative inputs to ensure it interprets the data correctly. Overall, the experiment was successful, confirming that GlowLab reliably provides accurate, personalized product recommendations.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Ensuring Data Integrity in the Data Service System

The Data Service System manages all user information, including saved products, skin check-ins, and profile details, by storing and retrieving it through the database. This system works with every part of the app because each screen depends on accurate and consistent data. Home Page routines, skin history, product libraries, and recommendations all rely on the Data Service to stay organized and accessible. However, because the entire user experience relies on this data, the system must account for potential failures, such as data becoming corrupted, falling out of sync, or failing to save correctly. To mitigate these risks, the system utilizes structured models and consistent read-write methods to maintain stability.

2.2. AI-Driven Product Ingredient Analysis System

The AI Product Analysis System processes product images or descriptions by sending them through the OpenAI service for ingredient analysis and safety evaluation. It supports the Product Details Page and Product Search Library by turning unstructured inputs into readable results, such as irritant warnings or breakout risks. This system strengthens the core idea of GlowLab by giving users immediate, intelligent feedback about products before they add them to their routine. As a current limitation, the system relies on the raw accuracy of the AI and the quality of the image the user takes without an external ingredient model or image-clearance check. However, for future improvement, the system is designed to notify the user if it misidentifies the product, serving as a manual check to ensure the information provided is as accurate as possible.

2.3. Personalized Recommendation Service System

The Recommendation Service System uses the user's profile, product history, and skin check-in data to generate personalized product suggestions. It connects directly with the Product Details Page and Profile Page to match recommendations with the user's needs, concerns, and sensitivities. This system helps the app feel meaningful rather than generic because it adapts as the user's skin changes and the database evolves. Its purpose is to reduce guessing and give users confident guidance grounded in their own data. To ensure these suggestions never feel random or unhelpful, the system bases recommendations entirely on the user's stored preferences and tracked skin history. Furthermore, if a user's skin changes over time, the system seamlessly adjusts using new check-ins and history, while also allowing the user to search for specific products, keywords, and skin goals to maintain total control over their experience.

3. SOLUTION

When you first open GlowLab, the page you see is the Home Page. From this screen, the user can add products to their routine, complete a skin check-in, update their breakout map, and read the tip of the day. As soon as this page loads, the app connects to its database in the background to retrieve any previously saved information, such as the user's skin history, stored products, past check-ins, and profile details. This background retrieval ensures that every option shown on the Home Page is personalized and up to date with the user's current skin concerns. From the Home Page, the user can also navigate to the Profile Page, where they can add or clear their profile details. When the user makes changes here, the app immediately updates the database so that their information stays consistent across the rest of the system. Another page the user can access is the Product Info Page, which displays recommended products based on the user's saved profile data. This page also includes a search tool where the user can look up products using either an image or a text description. Once an input is submitted, the app performs several background tasks: it analyzes the ingredient list, checks for irritants, reviews breakout-related factors, retrieves product descriptions, and compiles all results into an easy-to-read display. From here,

the user can save any recommended product to their Product Library. Navigating to the Product Library Page allows the user to view their personalized shelf of saved products. Finally, the Overview Page organizes and displays the user's history of skin activity based on the data gathered throughout the app's various pages.

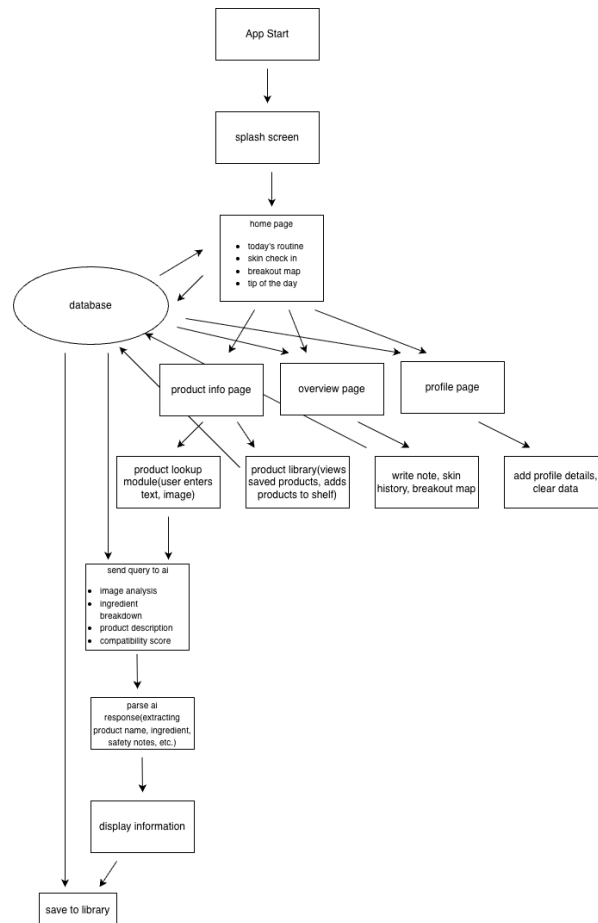


Figure 1. Overview of the solution

The product search system utilizes the OpenAI API to identify products based on images and descriptions. The AI provides detailed information about the product and recommends similar alternatives based on the input. This helps users learn more about their products, understand how to use them, and find safe alternatives for their skin concerns. Crucially, the app retrieves user preferences from the local Hive database and injects them into the AI prompt to personalize results and generate a compatibility score. It relies on sophisticated processing, such as OpenAI's API, to handle image recognition and natural language understanding [7].

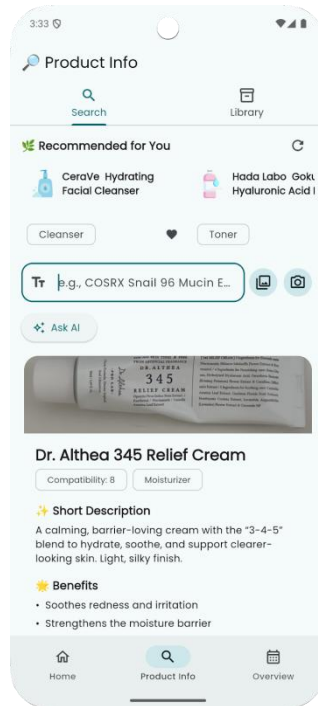


Figure 2. User Interface of the Product Analysis Module

```

90 Product selection rules:
91 - If mode = recommend_one:
92   - Prefer SKUs from the provided 'catalog' (hidden context). Choose ONE best match.
93   - If multiple are good, select the single best and include up to 2 alternatives.
94   - If the catalog has none, choose ONE widely available product (well-known, not obscure).
95   - Never invent products. If uncertain, pick a safe, common, real SKU and set "confidence":"medium".
96 - If mode = lookup:
97   - Identify the product from name/image; if ambiguous, pick the most likely real SKU and set "confidence":"medium". Do NOT be generic.
98 - If mode = generic:
99   - Provide generic guidance succinctly, but ONLY use this mode if selection is truly impossible.
100
101 User profile:
102 - You have a 'user_profile' context (hidden). Use it to tailor advice (fit_for_goals, conflicts_with_user). Do NOT reveal the profile itself.
103
104 Brand/name correction:
105 - You may correct minor brand/name typos only when there is a clear, real match (e.g., "COSRX" + "COSRX"). Do not guess obscure SKUs.
106
107 Images:
108 - If an image is provided, extract visible brand/name/claims/ingredients and assess compatibility (1-10).
109
110 Output:
111 - Return strict JSON with these fields. Keep the same keys.

```

```

109 // ---- 1) Pull user profile (exclude name) and provide as hidden context ----
110 // Assumes HiveService.getProfile() returns a Map<String, dynamic>
111 final Map<String, dynamic> rawProfile = HiveService.getProfile();
112 final Map<String, dynamic> userProfile = {
113   // include everything useful EXCEPT name
114   'ageRange': rawProfile['ageRange'],
115   'gender': rawProfile['gender'],
116   'skinType': rawProfile['skinType'],
117   'goals': rawProfile['goals'],
118   'skinConcerns': rawProfile['skinConcerns'],
119   'allergies': rawProfile['allergies'], // if you store it
120   'sensitivities': rawProfile['sensitivities'], // if you store it
121   'productPrefs': rawProfile['productPrefs'], // e.g., fragrance-free, vegan
122   // add other stable fields you keep in profile, but NOT the name
123 }.removeWhere((_, v) => v == null || (v is String && v.trim().isEmpty));
124
125 final uri = Uri.parse('https://api.openai.com/v1/chat/completions');
126 final headers = {
127   'Authorization': 'Bearer $_apiKey',
128   'Content-Type': 'application/json',
129 };

```

```

287 final resp = await http.post(uri, headers: headers, body: body);
288 if (resp.statusCode < 200 || resp.statusCode >= 300) {
289   return 'OpenAI error: ${resp.statusCode} ${resp.body}';
290 }
291
292 // ---- 8) Return raw JSON string (so your UI can decide how to render) ----
293 try {
294   final data = jsonDecode(resp.body);
295   final content = data['choices'][0]['message']['content'];
296
297   // content is a JSON string because of response_format=json_object
298   final Map<String, dynamic> j = (content is String)
299     ? jsonDecode(content)
300     : Map<String, dynamic>.from(content);
301
302   // If you want to keep returning markdown, convert here.
303   // Otherwise, return canonical JSON (recommended for your new "clean slate" UI).
304   return jsonEncode(j);
305 } catch (e) {
306   // Minimal safe fallback: short friendly text
307   return "Here's a quick take:\n• Hydration + barrier support are safe bets.\n• Look for glycerin/HA/ceramides; avoid heavy oils if you're oily."
308 }

```

Figure 3. Implementation of the OpenAI API Integration Function

The function starts by checking the API key and returns an error string if missing. It creates a system prompt to direct the assistant's behavior to return a JSON-formatted output [8]. It reads the profile using the local database and builds a filtered profile map. It prepares the user's message and checks if an image was provided, and converts it into a format that the OpenAI API can read. It combines the system prompt, hidden profile, example, and user input into a single list. The function sends that list to OpenAI's API endpoint [10]. It inserts a few-shot example to shape responses. The function parses the response into a JSON format that is processed by the application. If there is an error with the response, it will default to a fallback output.

The purpose of the dashboard component is to display the user's skin data in a clear and organized way so they can understand their progress over time. This component uses a local backend database to store information such as skin mood chips, daily routines, and breakout heat map data. It relies on Flutter Material Design widgets for consistent user interface elements and Flutter heat map tools to visually represent breakout patterns. A custom painter is used to allow drawing directly within the app, which supports interactive visual features like the breakout map. This component does not rely on machine learning but instead focuses on data visualization and user interaction, turning stored information into meaningful insights that help users track and manage their skin health effectively over time.

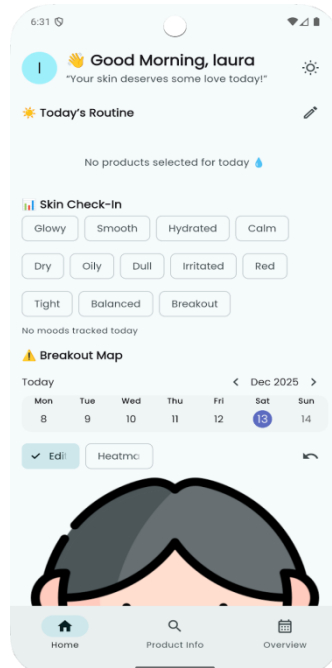


Figure 4. Dashboard Interface for Skin Activity Visualization

```

child: TableCalendar(
  firstDay: DateTime.utc(2024, 1, 1),
  lastDay: DateTime.utc(2030, 12, 31),
  focusedDay: _focused,
  rowHeight: 35,
  startingDayOfWeek: StartingDayOfWeek.monday,
  selectedDayPredicate: (d) =>
    d.year == _selected.year &&
    d.month == _selected.month &&
    d.day == _selected.day,
  onDaySelected: (sel, foc) => setState() {
    _selected = sel;
    _focused = foc;
  },
  calendarStyle: const CalendarStyle(
    outsideDaysVisible: false,
  ), // CalendarStyle
  headerStyle: const HeaderStyle(
    formatButtonVisible: false, titleCentered: true), // HeaderStyle
  calendarBuilders: CalendarBuilders(
    markerBuilder: (context, day, events) => _marker(day),
  ), // CalendarBuilders
), // TableCalendar
), // DecoratedBox
const SizedBox(height: 16),

```

```

DayDetailsCard(
  date: _selected,
  all: all,
  onChanged: () => setState() {},
), // DayDetailsCard

const SizedBox(height: 16),

// Face breakout / heatmap
Text('Breakout Map',
  style: Theme.of(context).textTheme.titleMedium), // Text
const SizedBox(height: 8),
FaceBreakout(
  date: _selected,
  onChanged: () => setState() {},
  imageAsset: 'assets/images/face_outline.png',
), // FaceBreakout
],
), // ListView
), // Scaffold
); // SafeArea
}
}

```

```

AspectRatio(
  aspectRatio: 3 / 3, // portrait face
  child: LayoutBuilder(
    builder: (context, c) {
      final size = Size(c.maxWidth, c.maxHeight);

      // choose dataset
      final List<Offset> points =
        _showHeatmap && widget.aggregateAllDays
          ? _aggregateAllPoints() // across all days
          : (_showHeatmap ? _points : _points);

      return GestureDetector(
        onTapDown: !_showHeatmap ? (d) => _onTapDown(d, size) : null,
        child: Stack(
          fit: StackFit.expand,
          children: [
            ClipRRect(
              borderRadius: BorderRadius.circular(16),
              child: Image.asset(
                widget.imageAsset, // <-- use selected face template
                fit: BoxFit.contain, // keep inside the box
              ), // Image.asset
            ), // ClipRRect
            if (_showHeatmap)
              CustomPaint(painter: _HeatmapPainter(points)),
            if (!_showHeatmap)
              CustomPaint(painter: _DotsPainter(points)),
          ],
        ),
      ),
    ),
  ),

```

Figure 5. Implementation of the Breakout Mapping Component

This dashboard screen is built using a Scaffold wrapped in a SafeArea and organized with a scrollable ListView. At the top, a TableCalendar widget allows users to select the date they want to view. The selected and focused dates are stored in state, and the calendar updates markers and day details based on the chosen date.

Below the calendar, a custom DayDetailsCard displays and updates daily routine and mood data, saving changes to the database when selections are made. The breakout map is implemented in the FaceBreakout widget. It uses an AspectRatio and LayoutBuilder to create a square drawing area that scales correctly across devices. A face outline image is placed in the background using a Stack. User input is captured with a GestureDetector, which records tap locations as Offset points. These points are rendered using CustomPaint, either as individual dots or as a heatmap overlay, depending on the selected mode [9].

The profile screen gives us important information to recommend and avoid products. It is an important part of the app because it is used to give the most accurate, useful, and safe information. It can be overlooked, but it is a very important part of the app. For all of the functions that use AI, we are plugging the store data into our parameters. Without profile data, our functionalities for recommendations, compatibility scores, and warnings would not be accurate.

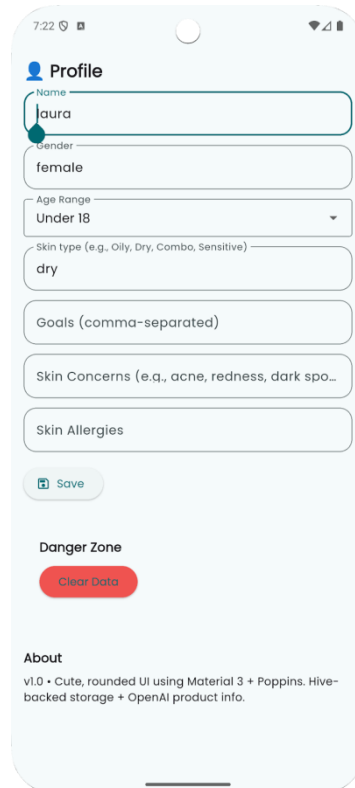


Figure 6. User Profile Interface for Personalized Recommendation Input

```

Future<void> _save() async {
  await HiveService.setProfile(
    name: _name.text.trim(),
    skinType: _skinType.text.trim(),
    goals: _goals.text.trim(),
    ageRange: _selectedAgeRange.trim(),
    gender: _gender.text.trim(),
    skinConcerns: _skinConcerns.text.trim(),
    allergies: _skinAllergies.text.trim(),
  );

  if (!mounted) return;
  ScaffoldMessenger.of(context)
    .showSnackBar(const SnackBar(content: Text('Profile saved')));
  setState(() {});
}

static Future<void> setProfile({
  String? name,
  String? skinType,
  String? goals,
  String? ageRange,
  String? gender,
  String? skinConcerns,
  String? allergies,
}) async {
  final profile = getProfile();
  if (name != null) profile['name'] = name;
  if (skinType != null) profile['skinType'] = skinType;
  if (goals != null) profile['goals'] = goals;
  if (ageRange != null) profile['ageRange'] = ageRange;
  if (gender != null) profile['gender'] = gender;
  if (skinConcerns != null) profile['skinConcerns'] = skinConcerns;
  if (allergies != null) profile['allergies'] = allergies;
  await _prefs.put('profile', profile);
}

```

Figure 7. Profile Data Retrieval and Storage Implementation


In the UI, we create a text field for each profile variable and connect it to a matching `TextEditingController`, which lets the app display saved data and read new user input. When each text editing controller is created, a function called `getProfile` is run. This function retrieves the user's existing profile from the database as a map, where each piece of data is stored with a specific key. We then use each key to get its related value from the map and use those values to initialize the text editing controllers, so the text fields are automatically filled with the user's saved information. All of the profile variables are initialized as nullable strings so that if any value is missing, the code will not crash. When the user clicks the Save button, the `onSave` function runs and collects the current values from the text fields. Inside this function, the database function `setProfile` is called, and all of the profile variables are passed into it as arguments. The `setProfile` function then stores these string values into a database table called `profile`, updating or creating the user's saved profile for future use.

4. EXPERIMENT

4.1. Experiment 1

In this experiment, we tested the accuracy of the AI system to ensure that it accurately recognized products from photos and descriptions and correctly matched them to users' skin profiles. This is important because it prevents errors, avoids recommending unsuitable products, maintains the app's reliability, and builds user confidence in the system's guidance and suggestions.

We designed our experiment by creating a set of 20 inputs for the AI system, including product photos and typed product names. For each input, we defined the expected output, such as correctly identifying the product and matching it to the user’s skin profile. The inputs were intentionally varied to cover different skin types, routines, and product formats. This variety is important because it ensures the AI can handle a wide range of real-world scenarios, remain accurate and reliable, and provide personalized recommendations, ultimately helping users trust and confidently use the system.

Input	Expected Output	Actual Output	Results
	Product info of Dr. Althea 345 relief cream, with short description, benefits, key ingredients, etc.	Product info of Dr. Althea 345 relief cream, with short description, benefits, key ingredients, etc.	Positive
Name of product	Product info for Dr. Althea's 345 line, with short description, benefits, key ingredients, etc.	Product info for Dr. Althea's 345 line, with short description, benefits, key ingredients, etc.	Positive
Profile information for a lady 55+, with very dry skin and her goal is to prevent aging.	Recommendations for products that fits her needs	Product recommendations with very high compatibility scores.	Positive
Profile with dry screen, product type, sunscreen	Product info of a hydrating broad spectrum SPF 30-50 with a short description, benefits, key ingredients, etc	Product info of a hydrating broad spectrum SPF 30-50 with a short description, benefits, key ingredients, etc	Positive


Profile with dry skin, product type, cleanser	Product info of a gentle hydrating cleanser with a short description, benefits, key ingredients, etc	Product info of a gentle hydrating cleanser with a short description, benefits, key ingredients, etc	positive
"A product to exfoliate with"	A product recommended by the ai	A product that helps with exfoliating skin.	Positive
"A sunscreen that can also repel mosquitoes"	Product that possibly exists	A real product called Bullfrog Mosquito Coast Sunscreen + Insect repellent SPF 50	Positive
"A product to smooth out acne scars"	Recommended product fit for the task	Positive result with a high compatibility score	Positive
"An oil cleanser for makeup removal"	Recommended product	An oil cleanser that removes makeup	positive
"Sunscreen without parfum/ fragrance"	Recommended product with detailed information	A fragrance free sunscreen	Positive
"antibacterial moisturizing hand soap"	Relevant product with detailed information	Soap that includes antibacterial cleansing	Positive
	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
Image of product, essence	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive

			
Image of product, toner	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
Image of product, cleansing oil	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
Image of product, spray	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive

			
Image of product, sunscreen	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
			
Image of product, cream	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
			
Image of product, serum	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
			
Image of product, cleanser	Correct product information	Product info with short description, benefits, key ingredients, etc.	Positive
			

Figure 8. Experimental Setup and Input Examples for Product Recognition Evaluation

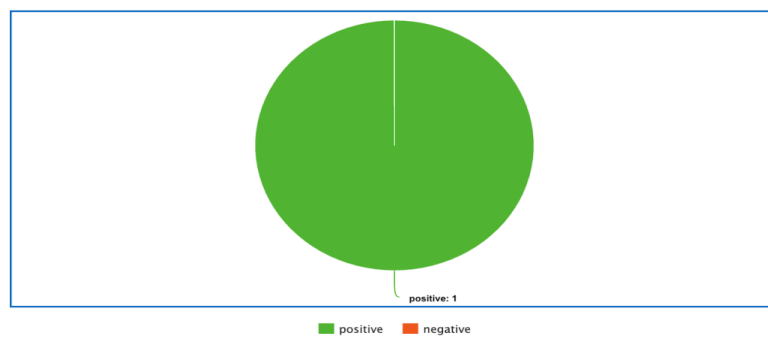


Figure 9. Accuracy Results of AI-Based Product Recognition (n = 20)

The system performed very well overall, correctly identifying all 20 inputs with no errors. Because it got no inputs wrong, the system's overall accuracy was 100%, indicating that it handled every example exactly as expected. Since there were no incorrect responses, there was no clear commonality among wrong inputs, as no mistakes occurred to analyze. However, if the system had gotten certain inputs wrong, the most likely causes would have been unclear wording,

ambiguous patterns, or inputs that were too similar to each other, which can make it harder for a model to distinguish categories accurately. Another possible reason for errors could be limitations in the training data, where the system may not have seen enough examples of a particular format or phrasing. Even though no mistakes happened in this case, these are typical factors that often lead to errors in similar evaluations.

5. RELATED WORK

The paper titled "Customer Intention to Use AI Technology in the Beauty Industry" by Yustikarani Julianti Pambudi and I Putu Wahyu Dwinata J.S. focuses on understanding why customers choose to use AI in beauty and skincare [11]. Instead of building an AI system, the authors used survey data to analyze factors like comfort, social influence, and technology readiness. Their research explains customer attitudes toward AI, while my project applies AI by giving users personalized skincare guidance. I created a functional tool for discovery. My system offers a distinct advantage for everyday users because it provides immediate, accessible utility without the life-critical risks associated with medical diagnosis.

The article "Dermatology and artificial intelligence" by W. Clark Lambert and A. Grzybowski reviews AI's role in diagnosing skin cancer and analyzing cosmetics, warning about "hallucinations" and safety risks [12]. Unlike the high-stakes medical systems discussed in the paper, which require doctor verification to prevent dangerous errors, my project is a consumer-focused AI for product search and recommendations. While the research focuses on theory and regulation, I built a functional tool for discovery. My system is "better" for everyday users because it offers immediate, accessible utility without the life-critical risks of medical diagnosis.

In "Artificial Intelligence in the Evolution of Customized Skincare Regimens," Hash et al. review how AI uses computer vision, genetic data, and environmental tracking to create dynamic, personalized skincare routines [13]. While the systems analyzed in the paper rely on invasive data collection, such as facial scanning and biometric sensors, to formulate custom products, my project focuses on accessible product search and recommendations via user input. My system is "better" for the everyday consumer because it avoids the significant privacy and security risks associated with storing sensitive biometric data, a major concern highlighted in the research.

6. CONCLUSIONS

One limitation of my project is that the AI analysis depends on the quality of product photos and text descriptions. Since the system uses OpenAI's existing model rather than a custom-trained one, the results rely entirely on the clarity of the input provided [14]. If the image is blurry or the label is unclear, the system may not identify the product correctly and could miss certain ingredients. In the future, I would like to expand the program to include a community review system and smarter personalized routines that adapt daily based on patterns in the user's skin history. To make suggestions more medically supported, I would also want to connect the recommendation system to real dermatology resources [15]. Another key feature I would add is a safety check that cross-references ingredients against previous reviews to ensure a product is not harmful or damaging for that specific user. If I were to start the project over, I would plan out the data systems more thoroughly from the beginning, especially the connections between product info, routines, and skin tracking. This would have made the program much easier to expand and more consistent across all its features.

REFERENCES

- [1] Auger, Tom, and Emma Saroyan. "Overview of the OpenAI APIs." *Generative AI for Web Development: Building Web Applications Powered by OpenAI APIs and Next.js*. Berkeley, CA: Apress, 2024. 87-116.
- [2] Cafarella, Michael J., Alon Halevy, and Jayant Madhavan. "Structured data on the web." *Communications of the ACM* 54.2 (2011): 72-79.
- [3] Woodby, Brittany, et al. "Skin health from the inside out." *Annual review of food science and technology* 11.1 (2020): 235-254.
- [4] Hay, Roderick J., et al. "The global burden of skin disease in 2010: an analysis of the prevalence and impact of skin conditions." *Journal of investigative dermatology* 134.6 (2014): 1527-1534.
- [5] Streifer, Philip A. *Tools and techniques for effective data-driven decision making*. Bloomsbury Publishing PLC, 2004.
- [6] Khan, Rida E. Haram, and Karan Singh. "AI-Driven Personalized Skincare: Enhancing Skin Analysis and Product Recommendation Systems." *Journal of Scientific Innovation and Advanced Research (JSIAR)* 1.2 (2025): 178-184.
- [7] Sarikaya, Ruhi, Geoffrey E. Hinton, and Anoop Deoras. "Application of deep belief networks for natural language understanding." *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22.4 (2014): 778-784.
- [8] Spivak, Iryna, et al. "Validation and data processing in JSON format." *IEEE EUROCON 2021-19th International Conference on Smart Technologies*. IEEE, 2021.
- [9] Inoue, Michiko, et al. "Heatmap overlay using neutral body model for visualizing the measured gaze distributions of observers." *Asian Conference on Pattern Recognition*. Cham: Springer Nature Switzerland, 2023.
- [10] Taulli, Tom, and Gaurav Deshmukh. "OpenAI GPTs and the Assistants API." *Building Generative AI Agents: Using LangGraph, AutoGen, and CrewAI*. Berkeley, CA: Apress, 2025. 57-79.
- [11] Adawiyah, Siti Rabiatal, et al. "The influence of AI and AR technology in personalized recommendations on customer usage intention: a case study of cosmetic products on shopee." *Applied Sciences* 14.13 (2024): 5786.
- [12] Hogarty, Daniel T., et al. "Artificial intelligence in dermatology—where we are and the way to the future: a review." *American journal of clinical dermatology* 21.1 (2020): 41-47.
- [13] Hash, Mary Grace, et al. "Artificial intelligence in the evolution of customized skincare regimens." *Cureus* 17.4 (2025).
- [14] Roumeliotis, Konstantinos I., and Nikolaos D. Tselikas. "Chatgpt and open-ai models: A preliminary review." *Future Internet* 15.6 (2023): 192.
- [15] Ko, Hyeyoung, et al. "A survey of recommendation systems: recommendation models, techniques, and application fields." *Electronics* 11.1 (2022): 141.