Sook Sook

Smart Home Platform for Hair Loss Treatment Medical Devices Based on LG Pra.L MediHair

An Hyosung
Dept.of Information Systems
Hanyang Univ.
Seoul, Republic of Korea
anhyosung@hanyang.ac.kr

Zhang Yan
Dept.of Information Systems
Hanyang Univ.
Seoul, Republic of Korea
wkddus@hanyang.ac.kr

Jeon Hyuna
Dept.of Information Systems
Hanyang Univ.
Seoul, Republic of Korea
younghyuna12@hanyang.ac.kr

Park Jinwoo
Dept.of Information Systems
Hanyang Univ.
Seoul, Republic of Korea
jinwoo220@hanyang.ac.kr

***Abstract*—Hair has held a significant position in human society throughout history. However, with time, issues like environmental pollution, social concerns, and improper use of various chemicals have exacerbated the problem of hair loss among people. This issue is now affecting individuals at a younger age, demonstrating a trend towards earlier onset. As the demand for hair loss treatment has grown, various products such as hair loss shampoos and medications have emerged. Additionally, advancements in technology have made it possible for at-home medical devices to become readily available. In 2020, South Korean company LG Electronics introduced its hair care product, LG Pra.L Medihair, which garnered considerable attention. However, it's important to note that treating hair loss is not an overnight process and requires consistent, long-term usage. To prevent individuals from giving up due to a lack of immediate results, we have incorporated features like reminders and visual comparison charts of treatment progress in our device. These enhancements provide users with a more intuitive way to monitor their hair loss treatment. Furthermore, we offer personalized preventive and treatment solutions based on users' age and scalp condition. We also recommend medications and shampoos that can be used in conjunction with the Pra.L Medihair device. In essence, this application aims to assist users in a more effective utilization of the Pra.L Medihair machine for hair loss treatment.**

Keywords—visualization, personalized, effective

Table I:Role Assignments

Roles	***Name***	***Task descriptions and etc.***
Front-end Developer	An Hyosung	Front-end developer is responsible for the coding of the overall UI/UX and the practical development of the front-end. It is responsible for processing data on the screen through api communication with Backend, and is responsible for implementing chatbots and AI solutions using gpt api. Test user interfaces to ensure functionality is is working properly, debugging and fixing potential issues.
Backend Developer	Park Jinwoo	As backend developer responsibilities include building a login and registration system for user authentication. Additionally, implementation of an email-based authentication code sending feature and a password recovery function is undertaken. Design and establishment of a database to store and manage user hair loss data, information collected from User, and recommended treatment methods and product details are carried out. Development of an API that provides personalized information and recommendations based on analysis results from the AI model is also a key task.
Front-end Developer	Zhang Yan	The front-end developer is responsible for the overall UI/UX page design of the software and personally undertakes the front-end development tasks. They ensure that the user interface is not only visually appealing and seamless but also delivers an outstanding interactive experience. The design work is carried out using Figma, a popular design tool, to create and refine the visual elements and user experience.
AI Development	Jeon Hyuna	AI developers crafts algorithms to analyze user behavior and data, with a primary focus on delivering personalized experiences. They harness natural language processing to comprehend text data, elevating user interactions. By seamlessly integrating machine learning and deep learning models into the app's core functionalities, AI developers bring forth advanced features such as predictions, recommendations, image recognition, and voice recognition. Their responsibilities not only contribute to enhancing user experiences but also bolster the app's competitive edge.

I. Introduction

A. Motivation

Hair is becoming a symbol of youth as the desire to maintain healthy youth naturally emerges as life spans increase and social activities increase in line with the rapid changes in modern society. Hair originally has the purpose of maintaining body temperature and protecting the head, but over time, it has changed into one of the means of expressing one's personality. Throughout history, and in most (although not all) civilizations, scalp hair has been associated with positive signals such as beauty and power but Baldness or hair loss on the other hand has a negative attribute [1].

Today, the demand for various hairstyles increases the frequency of physical and chemical beauty, resulting in damage not only to the hair but also to the scalp, causing hair loss. As a result, interest in scalp care for healthy scalp and beautiful hair is increasing, and various hair care products are being created for people who are severely stressed by hair loss. In terms of hair loss, not only genetic factors, which are fundamental reasons, but also acquired factors such as environmental pollution, stress due to social life, and hormonal secretion abnormalities caused by changes in negative eating habits are increasing. Hair loss, which used to be a symbol of aging, has become lower in age, with the average incidence of hair loss for men from 34.1 in 2005 to 31.6 in 2010, and young college students in their 20s are also seriously considering hair loss [2].

In addition to the increase in hair loss among young men in their 20s and 30s, women's hair loss is also increasing due to malnutrition, perm, dyeing, and drying caused by excessive diet, and the hair loss market is expanding rapidly due to this increase in the hair loss population, and the hair loss-related industry has entered a growing period [3]. The domestic hair loss market consists of hair care products, hair care services, hair loss treatments, wigs and hair transplants, and related new products are being developed one after another in the pharmaceutical and household goods industries. South Korean firm LG Electronics has also launched its hair care product, LG Pra.L Medihair.

In this article, we redefine the scope of development for Pra.L Medihair, focusing on the creation of an application that utilizes an AI model for hair and scalp analysis. This a pplication aims to provide users with detailed insights into the condition of their hair and scalp. In addition, we plan to integrate a feature that recommends treatment solutions ba sed on the usage of Pra.L Medihair. These treatment solutio ns are generated by a trained generative AI, which also ena bles interactive chat capabilities, allowing users to receive personalized advice and information. This approach not onl y addresses the current issues faced by users but also enhan ces the overall functionality and user experience of Pra.L M edihair, aligning with our vision of innovative and user-cen tric hair care solutions.

B. Problem Statement (client's needs)

Due to the nature of Medihair's treatment method, it takes a long-term commitment of at least four months to see noticeable results, which can be challenging for first-time users. If significant results are not seen within a short period, it is easy to become discouraged and fail to achieve the expected treatment outcomes. Furthermore, since this product does not require daily use, it is easy to forget to use it.

The list of experiences related to hair loss, the associated costs, and the impacts:

1. Hospital visits and treatment expenses:
   - Hair loss symptoms may require visits to the hospital and consultations with medical professionals.
   - This leads to medical expenses, and often requires regular treatment sessions and monitoring.
   - These costs can be a financial burden and may escalate if long-term treatment is necessary.
2. Purchase of hair loss treatment medications and associated costs:
   - Depending on a doctor's prescription, you may need to purchase hair loss treatment medications.
   - These medications vary in price depending on the type and brand and typically require long-term use.
   - The cost of purchasing and using hair loss treatment medications adds to the overall expenses of managing hair loss.
3. High costs associated with hair transplantation:
   - In cases of severe hair loss, individuals may consider hair transplantation as an option.
   - Hair transplantation is a surgical procedure that can be quite expensive, and it also involves a recovery period.
   - This can result in significant financial expenses and impact daily life during the recovery period.
4. Purchase of wigs:
   - When hair loss leads to significant hair thinning or baldness, individuals may need to purchase wigs.
   - The price of wigs can vary depending on the materials and styles chosen, and there are ongoing maintenance costs to consider as well.
5. Additional stress:
   - Hair loss-related changes in appearance and societal pressures can lead to additional stress.
   - Stress can have negative effects on mental health and quality of life, potentially requiring additional expenses for stress management and psychological support.

Among the various causes of hair loss, severe mental stress can also significantly impact hair growth. Faced with a relatively long waiting period, psychological stress may also increase. Like hair loss shampoos, Medihair only has a therapeutic effect on specific causes of hair loss. Given the diversity of factors causing hair loss, individual analysis based on scalp type and the location of hair loss is necessary before treatment.

C. Research on Any Relative Software

- ATOMOM

This is artificial intelligence (AI) skin health measurement and customized management software for atopic dermatitis patients. It provides a 1:1 customized care program by analyzing the skin condition and lifestyle habits of skin photos taken of the user's face, body, etc., using non-face-to-face atopic dermatitis diagnostic technology obtained by using AI to learn from tens of thousands of dermatological clinical data. It also has an ai health report that analyzes more than 40 types of diagnosis and degree of improvement of atopic dermatitis and other skin diseases, health record management that helps prevent worsening of symptoms, and a peace-of-mind store that provides products that contain only those that have been selected through strict benchmarks.[4]

- LG ThinQ

This is a smart home platform and software from LG Electronics. It is designed to enable LG's appliances and smart devices to connect to the Internet with smart control, automation and connectivity features to provide a more convenient home experience. With LG ThinQ, users can remotely control and monitor LG appliances that support the platform, such as a washing machine that allows you to remotely view its laundry status and formation completion via your phone.[5]

- INOUT

This is a simple and neat recipe, exercise, and weight loss logging software. Users can identify their weight changes at a glance through charts and graphs, as well as get detailed information on calories and nutrients consumed on a daily, weekly, and monthly basis. And you can customize the notifications to receive them at the desired time for healthy habits.[6]

- Google Calendar

This is an online calendar application developed and provided by Google Inc. designed to help users easily manage and organize their time, schedule and appointments. Users can easily create, edit and delete events, appointments and reminders. Google Calendar supports day view, week view, month view and task list view for users to better understand their schedule. And reminders and notifications can be sent to ensure users don't miss important appointments and events. These reminders can be sent via email, mobile notifications, pop-ups, and more.[7]

- Google Analytics

This is a free web analytics tool provided by Google that focuses on measuring and improving online business results by collecting and analyzing data about website visitors. It provides more than 100 standard reports covering a wide range of data about visitors, and by selecting the data to be analyzed, reports can be customized to view data on specific metrics and dimensions. System segments and customized segment functionality for more in-depth data analysis are also available. [8]

II. REQUIREMENT

A. Sign up

1. Login and Sign-up Screen: Upon launching the app, the user is presented with a login and sign-up screen, providing options for both new users and existing users.
2. User Registration: During the sign-up process, the user is required to provide the following information:
    - ID: A unique identifier chosen by the user.
    - Password: A secure password of their choice.
    - Email: A valid email address for communication and verification purposes.
3. Email Verification: After the user completes the registration form, the app sends an authentication code to the provided email address. This code serves as a verification step to ensure the user's email is valid and accessible.
4. Authentication Code Entry: The user is prompted to enter the authentication code sent to their email. The app validates the code, and the sign-up button becomes active only if the correct authentication code is entered.
5. Password Hashing: When storing the user's password in the database, the app employs Flutter's crypto encryption library to perform SHA-256 hashing on the password before storage. Password hashing enhances security by ensuring that plain-text passwords are not stored directly in the database.

When the app starts, it displays the login and sign up screen. The User enters their ID and password. The entered password is hashed using SHA-256. If this hashed password matches the ID stored in the database, the app displays 'Login Successful' message and proceeds to the next screen. If the password is incorrect, messages like 'Incorrect password' or 'The entered ID does not exist' are displayed. Additionally, provide an option for password reset through email verification for users who have forgotten their passwords.

B. Log in

1. Initial Screen: When the app is launched, the login screen is displayed to users as the initial interface, providing access to existing users.
2. User Input: The user is required to enter their credentials, including:
    - ID: A unique identifier associated with the user.
    - Password: The secret passphrase known only to the user.
3. Password Hashing: The entered password is securely hashed using the SHA-256 hashing algorithm before it is compared to the stored hash in the database. Hashing the password enhances security by ensuring that plain-text passwords are not transmitted or stored.

4. Validation and Authentication: The app checks if the entered ID exists in the database.
   - If the ID does not exist, the app displays a message such as 'The entered ID does not exist.'
   - If the ID exists, the app compares the hashed password entered by the user with the stored hash associated with the ID.
   - If the hashes match, the app displays a 'Login Successful' message, indicating a successful login.
   - If the hashes do not match, the app displays an 'Incorrect password' message.
5. Password Reset Option:
   - For users who have forgotten their passwords, the app provides a password reset option.
   - Users can initiate a password reset process by requesting an email verification.
   - An email containing a verification link or code is sent to the user's registered email address.
   - Upon successful verification, the user is guided through the process of resetting their password.

By implementing these requirements, the login screen of the app ensures a secure and user-friendly experience, providing appropriate feedback for login attempts and offering a password reset option for users who encounter password-related issues.

C. *Developing a Hair Loss Analysis AI Model*

1. Data Collection: Gather a substantial dataset consisting of 100,000 scalp images categorized by different types (e.g., dry, oily, hair loss-prone) available on AI-HUB. Ensure that the dataset is diverse and representative of various scalp conditions and hair loss patterns.
2. Machine Learning Model Development: Utilize machine learning techniques to build a robust AI model for analyzing users' hair loss progression. The AI model should be designed to accept input in the form of scalp images and relevant data.
3. User Input and Analysis:
   - Users provide input in the form of scalp images, and the AI model processes this data.
   - The AI model analyzes the images to categorize the user's scalp type (e.g., dry, oily) and identify specific hair loss patterns.
   - The model conducts a comprehensive assessment of the user's hair loss condition.
4. Progression Analysis: Based on the user's scalp type, hair loss patterns, and other relevant factors, the AI model assesses the progression of hair loss. It quantifies the severity and stage of hair loss to provide a clear understanding of the user's current condition.
5. Treatment Recommendations: The AI model generates personalized recommendations for the user, including potential improvements and treatment options. Recommendations may include lifestyle changes, product usage, and treatment duration tailored to the user's specific needs.
6. User Communication: The AI model communicates its findings and recommendations to the user through a user-friendly interface. Users can access information about their hair loss progression, potential improvements, and suggested treatment plans.
7. Data Privacy and Security: Ensure that user data, including scalp images, is handled with the utmost privacy and security. Implement encryption and data protection measures.
8. Model Optimization: Continuously improve and update the AI model using new data and research findings to enhance its accuracy and effectiveness in analyzing hair loss.

By fulfilling these requirements, the AI model for analyzing hair loss progression will provide users with valuable insights and personalized recommendations, enabling them to better understand their hair loss condition and take appropriate steps for improvement and treatment.

D. *Collecting Hair Loss Information from the User*

1. User Input: After successful login, users are prompted to provide information about their hair loss progression. This information is crucial for the AI model to assess the user's condition accurately.
2. Data Collection:

   The following data is collected from the user:
   - Duration of Experiencing Hair Loss: Users provide details on how long they have been experiencing hair loss. This duration helps determine the chronicity of the condition.
   - Medications Currently Being Taken: Users list any medications or treatments they are currently using for hair loss management. This information assists in understanding ongoing treatments.
   - Areas Affected by Hair Loss: Users specify the regions or areas on their scalp or body affected by hair loss. This data aids in identifying the extent of hair loss.
   - Family History of Hair Loss: Users share information about any family history of hair loss, as genetic factors can play a role in hair loss patterns.
3. Database Storage: The collected user data, including hair loss information, is securely stored in a database. Each user's information is associated with their unique user ID for retrieval and analysis.
4. Data Retrieval: When users engage with the AI model for hair loss analysis, their previously

provided information is retrieved from the database for reference.

5. AI Model Assessment: The AI model analyzes the user's hair loss progression based on the provided information, combining it with the data collected from scalp images. The model evaluates the severity, stage, and potential causes of hair loss, considering both the user's input and visual data.

6. Personalized Solutions: Based on the assessment, the AI model generates personalized solutions and recommendations for the user. Solutions may include treatment options, lifestyle changes, or product recommendations tailored to the user's specific needs and condition.

7. User Communication: The AI model communicates its findings and solutions to the user through the app's interface. Users can access information about their hair loss progression and the recommended steps for improvement.

8. Data Privacy and Security: Ensure that user data, including hair loss information, is handled with strict privacy and security measures, following data protection regulations.

By implementing these requirements, the app collects and utilizes valuable user data to assess hair loss progression accurately and provide personalized solutions. Data storage and retrieval from the database ensure that the user's historical information is readily available for analysis and ongoing support.

E. Receive and Analyze Hair and Scalp Information from User

1. Microscope-Equipped Camera Products: Camera products are equipped with specialized microscopes, strategically positioned to capture high-quality images of the user's hair and scalp condition from various angles

   These microscopes are designed to capture detailed data on hair.

2. Data Capture and Analysis:

   - Each time the user engages with Smartphone and the microscopes are used, data is captured and transmitted to the app.

   - The captured data includes images and metrics related to hair and scalp health, such as hair follicle density, hair strand thickness, and any visible scalp issues.

   - An AI model within the app is responsible for analyzing this data to determine the user's hair loss progression.

3. Timestamping: Each set of data received from Medihair is saved in the app's database along with a timestamp indicating the date and time of the data capture. Timestamping allows for tracking and monitoring changes in the user's hair and scalp condition over time.

4. Progression Tracking:

   - A feature is to be developed within the app that enables users to compare their hair and scalp information at different intervals.

   - Users can choose to compare data at 1-month, 3-month, 6-month, and 1-year intervals to observe changes and improvements in their hair and scalp condition over time.

   - This comparison feature provides valuable insights into the effectiveness of any treatments or interventions undertaken.

5. User Access: Users can access their historical hair and scalp information, including images and analysis results, through the app's interface. This information empowers users to make informed decisions about their hair care and treatment options.

6. Data Privacy and Security: Ensure that all data captured and stored from Medihair devices is handled securely and in compliance with data privacy regulations.

By implementing these requirements, the app is equipped to receive, analyze, and store hair and scalp information captured by Medihair products. Users can track the progression of their hair loss, make data-driven decisions, and observe changes in their hair and scalp condition over specific time intervals, fostering an informed and proactive approach to hair care.

F. Recommend Customized Treatment Methods, Solution and Shampoos

1. Treatment Method Recommendation: Based on the user's hair loss status and assessment, the app recommends suitable treatment methods. These recommendations are tailored to address the user's specific needs. For example, a treatment plan may involve different modes for specific days of the week, such as TopCare mode on Monday, FrontCare mode on Wednesday, and TotalCare mode on Friday. The recommended frequency, such as three times a week, is also specified.

2. Treatment Recommendations:

   - The app assesses the type and severity of hair loss the user is experiencing.

   - Depending on the assessment, the app recommends appropriate treatments. These may include:

   - Treatments containing relevant nutrients or active ingredients to address thinning hair or hair density issues.

   - Specific treatments that are suitable for the user's hair loss condition.

   - The app provides detailed information about the recommended treatments, including their benefits and how they should be used.

3. Shampoo Recommendations:
    - The user's scalp condition is taken into consideration for shampoo recommendations.
    - Depending on the assessment, the app suggests suitable shampoos that cater to the user's specific scalp condition. Examples include:
    - Protein-enriched shampoos for strengthening hair.
    - Hair loss prevention shampoos for addressing hair loss issues.
    - Alongside shampoo recommendations, the app provides a link or information on where the recommended shampoo can be purchased.
4. User Access and Information Display: Users can access their customized treatment plan, medication recommendations, and shampoo suggestions through the app's interface. The information is displayed clearly, allowing users to understand and follow their recommended hair care routine.
5. Data Tracking:
    - The app tracks the user's progress and adherence to the recommended treatment methods, medications, and shampoos.
    - Users can review their treatment history and adjust their routine as needed.
6. Medication Availability: For medications, the app displays information about them but does not provide an online purchase option, as medications may require prescriptions or specific regulations.

By implementing these requirements, the app empowers users to receive personalized recommendations for treating their hair loss based on their unique condition. It offers a comprehensive approach by suggesting treatment methods, medications, and shampoos tailored to the user's specific needs, promoting a proactive and informed approach to hair care.

G. Provide Scalp Solutions and Information on Scalp Status

1. Assessment of Hair Loss Progression: The app conducts a thorough assessment of the user's hair loss progression based on data collected from scalp images, user-provided information, and AI analysis. The assessment considers factors such as hair density, thickness, and scalp condition to determine the extent and severity of hair loss.
2. Solution Recommendations: Based on the assessment, the app offers tailored solutions for preventing and treating hair loss. These solutions are designed to address the user's specific condition and needs. The app provides clear and actionable recommendations, such as changes in lifestyle, treatment methods, medications, and shampoos.
3. Expected Improvement Communication: The app communicates the expected improvements that the user can anticipate by implementing the recommended solutions. For example, it may convey that by following the provided solution, the progression rate of hair loss could be slowed down, extending the estimated time it takes for noticeable hair loss from 5 years to 7 years. This information helps users set realistic expectations for their hair loss management.
4. Scalp 'Age' Determination: Using the AI model, the app determines the 'age' of the user's scalp. This metric represents the condition and health of the scalp. Scalp 'age' is calculated based on factors such as scalp condition, hair health, and the severity of hair loss.
5. Potential Scalp Age Improvement: The app informs the user about the potential improvement in their scalp's 'age' if they follow the provided solution. It may communicate that by adhering to the recommendations, the user's scalp condition can be expected to improve, resulting in a 'younger' scalp compared to its initial state.
6. User Access and Tracking: Users can access their personalized solutions, information on expected improvements, and scalp 'age' through the app's interface. The app also tracks the user's progress over time, allowing them to monitor their scalp's response to the recommended treatments.

By fulfilling these requirements, the app provides users with not only tailored solutions for hair loss but also valuable insights into the potential improvement of their scalp condition. This approach promotes informed decision-making and encourages users to follow a comprehensive hair care routine for better results.

H. Personalized Notification

1. User Preference: The AI model offers users the option to choose whether they want to receive notifications based on the recommended treatment plan. Users can set their notification preferences within the app, indicating whether they wish to receive reminders and updates related to their hair care routine.
2. Tailored Notification Content: For users who opt to receive notifications, the app generates personalized messages tailored to the user's ongoing hair care routine. These notifications are designed to remind and guide users on their specific treatment plan.
3. Notification Timing: Notifications are scheduled to align with the user's recommended treatment plan. The timing of notifications is determined by factors such as the day of the week, the specific treatment mode, and the user's preferred time for hair care activities.
4. Example Notification:
    - For instance, if the user's recommended treatment plan includes on Wednesdays, the app generates a notification on Wednesday morning.

- The notification may read: "Today is Wednesday, and it's the day for use Medihair. Please operate Medihair in FrontMode before 9 PM."

5. User Interaction: Users can interact with notifications, acknowledging that they have completed the recommended hair care activities for the day. Acknowledgments or responses from users may be tracked to monitor adherence to the treatment plan.
6. Notification History: The app maintains a notification history, allowing users to review past notifications and their responses. This history serves as a record of the user's engagement with the recommended routine.
7. User-Defined Preferences: Users have the flexibility to customize their notification preferences, including setting specific time windows for receiving reminders or adjusting the frequency of notifications.
8. User Privacy and Consent: Ensure that the app respects user privacy and complies with notification preferences set by the user. Obtain user consent for sending personalized notifications.

By implementing these requirements, the app provides a personalized and user-centric approach to notifications, helping users stay on track with their hair care routine. Users receive timely reminders and guidance tailored to their specific treatment plan, promoting adherence and effectiveness in managing hair loss.

III. Development Environment

A. Choice of Software Development Platform

Our team is planning to develop a cross-platform application that will work on both Android and iOS. To achieve this, we will utilize the Flutter framework, allowing us to build a compatible application for both platforms using a single codebase. The backend of the application will be developed using Node.js, known for its exceptional scalability and data processing capabilities, along with support for various modules/libraries that empower developers to implement a wide range of functionalities. Our AI model, built using Scikit-learn, will leverage machine learning to accurately analyze users' scalp and hair conditions, providing tailored recommendations. Hosting the server will be carried out through AWS's EC2, ensuring stability and security, essential for seamless application operation. Finally, MySQL will serve as the database for efficient management of relational data, guaranteeing secure data storage and retrieval. With this technology stack and configuration, our application is poised to deliver an exceptional user experience and reliability.

Table II: Tools and Language Choice

Tools and Language	***Reason***
Flutter	Flutter is an outstanding mobile app development framework developed by Google, allowing developers to create and maintain apps for both iOS and Android platforms with a single codebase. This cross-platform capability significantly enhances productivity. Flutter utilizes a widget-based approach to building user interfaces, enabling rapid design and modification of screens. It delivers near-native performance and includes excellent tools for optimizing screen performance, enhancing user experience. Additionally, Flutter benefits from a robust community, enabling developers to easily find and implement new features and packages, streamlining the development process.[9]
MySQL	MySQL stands out as a top-tier open-source relational database management system (RDBMS) known for its stability and performance. It excels in processing large volumes of data quickly, making it ideal for large-scale applications. MySQL provides support for various operating systems, allowing developers to use it in diverse environments. With powerful query capabilities, developers can execute complex database operations effortlessly. MySQL ensures data integrity and transaction handling, guaranteeing the safety of the database. Its scalable performance and high reliability make it an ideal choice for reliable data storage and processing in large-scale applications and web services.[10]
Node.js	Node.js is a server-side JavaScript runtime built on Chrome's V8 JavaScript engine, offering asynchronous I/O processing and exceptional scalability. Its event-driven, single-threaded model efficiently handles numerous connections simultaneously, ensuring high performance. Node.js processes data swiftly, significantly reducing response times for web applications. By utilizing Java Script for both frontend and backend, developers maintain

Tools and Language	***Reason***
Node.js	consistent code, enhancing efficiency. Node.js benefits from a vibrant community, providing a wealth of modules and packages, enhancing developer productivity. It is particularly well-suited for developing large-scale web applications and real-time applications, making it an ideal tool for building responsive, high-performance network applications.[11]

B. Cost Estimation

To ensure the successful development and functionality of our application, it is crucial to establish a seamless flow of data between the server and the database while maintaining real-time communication. To achieve this, we have chosen to leverage Amazon's EC2 (Elastic Compute Cloud) as our server hosting solution. While it's true that hosting on EC2 incurs costs, the advantages it offers are instrumental to our application's performance and reliability.

One of the key benefits of utilizing Amazon EC2 is its scalability. EC2 allows us to easily scale our application's resources up or down to accommodate varying workloads and user demands. This scalability ensures that our application remains responsive and efficient, even during peak usage periods. Additionally, EC2 provides a high level of reliability and availability, which are critical for our application's uptime and data accessibility.

In addition to hosting, we are integrating multiple APIs into our system to access database information and live server updates. This approach not only enhances data retrieval but also keeps our application's information up-to-date, contributing to an improved overall user experience. The combination of Amazon EC2's scalability, reliability, and the integration of these APIs allows us to offer a robust and responsive application that meets the expectations of our users.

Table III:Cost Estimation

Tools and Language	***Reason***
AWS EC2	AWS EC2 stands as a Virtual Private Server (VPS) provider, empowering developers with the resources necessary for computing, storage, and networking. It offers the flexibility to deploy and manage websites and web applications in the cloud environment. With AWS EC2, developers have access to a comprehensive suite of tools, including virtual machines, containers, databases, content delivery networks (CDNs), load balancers, and DNS management services. These tools facilitate the swift setup and seamless operation of projects. Moreover, AWS EC2 ensures reliability through its low and predictable monthly rates, providing an affordable yet robust solution for hosting and managing diverse online projects.

C. Software & AI model in Use

1. Git & Github

Git and GitHub are essential components of our project's development process. Git serves as our version control system, enabling us to track changes in our codebase systematically. This is particularly crucial in a project like ours, where code development evolves over time. With Git, we can create branches, make modifications, and merge changes seamlessly, ensuring that our codebase remains well-organized and that we can easily roll back to previous versions if issues arise. GitHub, as a web-based platform, enhances collaboration among team members. It provides a centralized location for our code repository, allowing multiple developers to work concurrently, review each other's code, and manage project issues efficiently. This streamlines our development workflow, ensuring that our project progresses smoothly and that we can effectively address any challenges that may arise during development.

2. Notion

Notion is a versatile tool that plays a pivotal role in our project management and documentation efforts. It enables us to create a cohesive and organized workspace where we can manage project timelines, tasks, and documentation effectively. In the context of our hair loss treatment application, we use Notion to maintain a detailed project plan, track progress, and collaborate on tasks and user stories. Notion's flexibility allows us to adapt our workspace to suit the evolving needs of our project, ensuring that we can easily accommodate changes and updates. By serving as a one-stop platform for project-related information, Notion streamlines communication and collaboration within our team, enhancing our productivity and ensuring that everyone has access to the latest project details and resources.[12]

3. AWS EC2 (Elastic Compute Cloud)

AWS EC2 is a crucial component of our project infrastructure. It provides us with scalable and reliable computing resources in the cloud, which are essential for running our hair loss treatment application efficiently. EC2's flexibility allows us to adjust our server capacity based on demand, ensuring that we can handle varying workloads without over-provisioning or incurring unnecessary costs. Additionally, EC2 gives us full control over our server instances, allowing us to configure security measures, network settings, and storage management according to our project's specific requirements. This level of control and scalability is essential for ensuring the availability and performance of our application, especially as it may experience fluctuations in user traffic and data processing needs.[13]

4. Visual Studio Code

Visual Studio Code serves as our primary integrated development environment (IDE) for writing, testing, and debugging code. Its versatility and robust feature set make it an ideal choice for our project. With Visual Studio Code, our development team can enhance code quality through syntax highlighting, Git integration, and debugging support. Furthermore, we can customize the IDE to match our preferred coding styles, increasing our overall coding efficiency. By fostering collaboration and allowing developers to tailor their workspaces to their specific needs, Visual Studio Code plays a vital role in ensuring that our codebase remains well-maintained, efficient, and easily manageable.[14]

5. AWS RDS (Relational Database Service)

AWS RDS plays a critical role in our project by simplifying the management of relational databases in the cloud. As our hair loss treatment application relies on data storage and retrieval, AWS RDS offers a reliable and scalable solution for handling large datasets efficiently. It automates essential database tasks, such as setup, scaling, and backups, reducing the administrative burden on our team. The cost-effective capacity provisioning ensures that we can manage database resources without incurring unnecessary expenses. Additionally, the automated maintenance and backup features of AWS RDS contribute to data integrity and availability, which are crucial for the success of our application. By leveraging AWS RDS, we can focus on developing our application's core features and functionality while relying on a robust and managed database solution.[15]

6. Google Colab

Google Colab serves as a valuable tool for our data science and machine learning tasks within the project. Its cloud-based, interactive Python environment allows us to perform data analysis, model training, and experimentation efficiently. The integration with Google Drive facilitates data storage and sharing, ensuring that our data science team can collaborate seamlessly. Furthermore, Google Colab provides free access to GPU resources, which is essential for accelerating machine learning tasks, especially those related to image processing and analysis within our hair loss treatment application. With real-time collaboration features and pre-installed libraries, Google Colab simplifies our data science workflow, enabling our team to explore and implement advanced machine learning techniques effectively.

7. Figma

Figma plays a central role in the design and prototyping phases of our project. As a cloud-based design and collaboration tool, it enables our design team to work collaboratively on interface design and user experience (UI/UX) planning. Figma's real-time collaboration features are particularly valuable, as they allow multiple designers to work simultaneously on the same project, fostering creativity and efficiency. Its support for vector graphics editing and interactive prototyping empowers our design team to create visually appealing and user-friendly interfaces. Additionally, Figma's design version control ensures that our design iterations are well-documented and accessible, promoting effective communication between designers and developers. Overall, Figma streamlines our design process, ensuring that our hair loss treatment application's user interface is intuitive and visually appealing.[16]

8. Scikit-learn

Scikit-learn is a fundamental component of our data science and machine learning toolkit. This open-source machine learning library in Python offers a wide range of machine learning algorithms, making it indispensable for our project's data analysis and modeling tasks. With Scikit-learn, our data science team can implement classification, regression, clustering, and dimensionality reduction techniques, among others. Its user-friendly interface and extensive documentation make it accessible to both novice and experienced machine learning practitioners, ensuring that our team can leverage its capabilities effectively. Scikit-learn's tools for model selection, evaluation, and data preprocessing further enhance our data analysis capabilities, enabling us to build robust machine learning models and conduct comprehensive data analysis for our hair loss treatment application.[17]

9. NumPy

NumPy is a foundational library in our data science and numerical computing toolkit. As a powerful numerical computing library in Python, NumPy provides essential support for handling large, multi-dimensional arrays and matrices, along with a comprehensive collection of mathematical functions. These capabilities are crucial for our data preprocessing, mathematical computations, and data analysis tasks within the project. NumPy's efficiency and performance advantages over native Python lists make it the ideal choice for working with large datasets, ensuring that our data science team can process and analyze data efficiently. Additionally, NumPy integrates seamlessly with other scientific computing libraries, such as SciPy and matplotlib, further expanding our capabilities for advanced data analysis and visualization. In summary, NumPy forms the backbone of our data science and numerical computing workflows, enabling us to perform complex calculations and manipulations effectively.[18]

10. EfficientNetB0

EfficientNetB0 is a state-of-the-art deep learning model that serves as a pivotal component in our image recognition and computer vision tasks. Developed as part of the EfficientNet family, this model is renowned for its exceptional balance between accuracy and computational efficiency, making it an ideal choice for processing images in real-time applications. EfficientNetB0 employs a novel scaling method that uniformly scales all dimensions of the network—depth, width, and resolution—using a compound coefficient. This approach ensures that the model scales up efficiently, achieving higher accuracy without a proportional increase in computational cost.[19]

IV. Specification

A. Login Page

The login page consists of the following components: an 'email input field,' a 'password input field,' a 'login button,' a 'forgot password button,' and a 'sign-up button.' The email input field has a maximum length limitation of 20 characters. The password input field must be between 10 and 20 characters in length, and it must contain both English letters and numbers, with the option to include special characters.

When a user enters their email and password and clicks the login button, the system first checks if the email and password meet the specified rules. Subsequently, the login API is invoked, sending the email and the SHA-256 hashed password to the server.

Within the login API, if the email and the SHA-256 hashed password match records in the database, a successful login message is sent along with a token containing the user's information to the page. Upon receiving the successful login message, the user is redirected to the next page using the provided token.

In cases where the email and hashed password do not match any records in the database, the login API sends a login failure message to the page. Upon receiving the login failure message, the system displays the message to the user, indicating that the login attempt was unsuccessful.

Additionally, the login page offers buttons for users who have forgotten their passwords and those who need to sign up for a new account, allowing them to navigate to the 'forgot password' and 'sign-up' pages, respectively.

B. Register page

Email-Based Registration: Users opting for traditional email-based registration are required to enter their email address. Upon submission, a verification code is sent to the provided email address. The 'verify code' button remains active for a duration of three minutes after sending the email.

- Email addresses must be under 20 characters in length to meet system requirements.
- Once the correct verification code is entered, the 'sign-up' button becomes active, allowing users to proceed.
- Users are prompted to enter their chosen password twice for verification. Passwords must adhere to the same criteria mentioned earlier, which include a combination of English letters and numbers, a length between 10 and 15 characters, and the option to include special characters. The double entry of the password helps ensure its accuracy.
- If the verification code is correct and the password meets the specified criteria, the user's email and password details are securely stored in the database, with the password being hashed using the SHA-256 encryption method.

This comprehensive registration process provides users with flexibility and security, accommodating various preferences while maintaining the necessary security measures.

C. User Information Entry Page

This page collects the user's name, age, gender, and nickname. Names must be under 5 characters in Korean. Age is a two-digit integer. Gender selection is configured as either male or female. Nicknames are set to under 10 characters and can consist of Korean, English, or numbers. The information collected is stored in the database. The entered age and gender are used to compare with the average hair thickness and density for that gender and age to assess the user's hair loss condition.

D. Main Page

The main page consists of the following elements: 'Welcome to Sook Sook' text, 'Hair Analysis button,' 'Analysis Records button,' 'Chatbot button,' 'Community button,' and 'MediHair product photos and buttons.' Additionally, at the bottom of the screen, there are 'Settings button,' 'Home button,' 'Notifications button,' and 'My Page button.'

The functionality associated with each button when clicked:

- Hair Analysis: Clicking this button redirects to a page where users can take pictures of their hair for analysis.
- Analysis Records: Clicking this button takes users to a page where they can review their hair analysis records for the past six months.
- Chatbot: This button leads to a page where users can interact with a generative AI model that provides treatment solutions based on user records and engages in a chat-based conversation.
- Community: Clicking this button directs users to a popular hair loss community where they can engage in discussions and seek advice.
- MediHair Product Photos and Buttons: This section links to a page for connecting with MediHair products, featuring product photos and related buttons.

At the bottom of the screen:

- Settings: Clicking this button navigates to a page where users can manage their records, make changes to settings such as record deletion, and customize their preferences.
- Home: This button returns users to the main page of the app.

- Notifications: Clicking this button takes users to a page where they can set up notifications for their MediHair product usage schedule.
- My Page: This button leads to a page where users can view and manage their personal information. On this page, users can perform actions such as changing their email address or updating their password.

E. Hair Analysis Page

On the 'Hair Analysis' page, users have the capability to take photos of their hair and perform an analysis. These photos can be captured using a mobile phone camera, but for enhanced accuracy, users are encouraged to utilize a digital electronic microscope.

While the analysis is in progress, a demonstration video showcasing the operation of the MediHair product is displayed to the user.

Once the user submits the captured photos, these images are transmitted to the hair analysis AI model via an API. The AI model then processes the user-provided images and derives the following information:

- Type of Hair Loss: The specific type of hair loss the user is experiencing
- Hair Density: The user's hair density (e.g., 13 per 1cm²).
- Hair Thickness: The diameter of the user's hair strands (e.g., 45µm).

The server retrieves this data and securely stores it in the database, simultaneously sending the information to be displayed on the page. The types and data formats stored in the database include:

- Analysis Date (in yyyy-mm-dd format)
- Hair Loss Status (varchar(5))
- Severity of Hair Loss (varchar(5))
- Hair Thickness (float type)
- Hair Density (int type)

The AI model presents the analyzed results to the user and recommends an appropriate treatment mode based on the findings (e.g., front mode). Additionally, any further details or recommendations from the analysis can be included as needed.

This comprehensive process empowers users to gain insights into their hair health and receive personalized treatment recommendations for their specific hair loss condition.

F. Hair Record Page

The 'Hair Record Page' offers users the capability to access and review their hair analysis records conducted on the 'Hair Analysis Page' within the past six months. This page features two buttons: 'Today's Record' and 'Previous Records.'

Today's Record: Clicking the 'Today's Record' button queries the database for the user's analysis record corresponding to the current date (yyyy-mm-dd). If a matching record exists, it presents the following information:

- Hair Loss Status
- Severity of Hair Loss
- Hair Thickness
- Hair Density

If there are no records for the day, a message is displayed, indicating 'No analysis was conducted today.'

Previous Records: The 'Previous Records' button retrieves analysis dates from the past six months stored in the database and displays them for the user to select. Upon selecting a specific date, the system fetches the corresponding analysis details, including:

- Hair Loss Status
- Severity of Hair Loss
- Hair Density
- Hair Thickness

These details are then presented on the page for the user's review.

This functionality empowers users to conveniently access and monitor their hair analysis history, facilitating a deeper understanding of their hair health over time.

G. Product Recommendation page

Sook Sook goes beyond hair and scalp analysis by delivering personalized recommendations for hair care products and dietary choices tailored to the unique needs of each user. The platform categorizes hair loss information and its severity across different regions of the head, including the front, center, and back. Utilizing a sophisticated algorithm, it classifies essential nutrients and ingredients based on the type and severity of hair loss, subsequently suggesting corresponding foods and hair care products.

1. The algorithm retrieves users' hair loss data, including the type of hair loss and its severity (degree of severity), from the database.
2. Leveraging the food and hair care product recommendation algorithm, the platform recommends specific dietary choices and hair care products to address the user's individual needs.

Additionally, this page presents a curated selection of various hair care products such as shampoos, conditioners, and scalp massagers. These products are designed to complement the user's hair care routine, aiming to combat hair loss issues and promote healthier, more resilient hair.

The holistic approach of Sook Sook ensures that users receive tailored guidance for both dietary and product choices, empowering them to take proactive steps towards better hair health.

H. Settings

1. Connection:

   This section checks the connection status of Medihair. The device connects to Medihair via Bluetooth on the phone. When navigating to 'Connection Status' within the app, if the connection is complete, it will display 'Connected'.

2. Medihair & App Manual:

   The app provides access to Medihair's product information and the application manual within the app itself. This is available without requiring a separate server connection by incorporating the text of the guide and user manual into the app for the user to view.

3. Dark Mode:

   The app offers a dark mode option, ensuring a more comfortable and visually appealing experience, especially in low-light environments.

I. Alarm

The 'Alarm Page' provides users with a convenient platform for managing their alarms efficiently. Users have the flexibility to set alarms based on specific days of the week and minute intervals. Leveraging the capabilities of the flutter_local_notifications Plugin library, the app sends timely reminders for Medihair usage at the scheduled times on the designated days.

Key features of the 'Alarm Page' include:

- Toggle Alarms On/Off: Users can easily enable or disable alarms with a simple toggle switch, giving them full control over their alarm settings.
- Delete Alarms: Unwanted alarms can be effortlessly removed from the list, ensuring a clutter-free alarm setup.
- Set Specific Dates and Times: Users have the freedom to customize each alarm by specifying particular dates and times for their Medihair usage reminders.

This user-friendly interface streamlines the process of setting and managing alarms, ensuring that users receive timely reminders for their Medi-hair usage. These alarms play a crucial role in helping users monitor and maintain their hair health effectively over time.

J. Profile

On the profile management page, users can log in and log out. Pressing the logout button takes the user back to the initial login screen. Additionally, the option to add a new profile is provided, allowing each profile to have a unique nickname. Clicking the button to add a new user prompts the user to enter their name, age, gender, and nickname. Names must be five characters or less and composed of Korean characters. Age is a two-digit integer. Gender options are set to be selectable between male and female. Nicknames are set to be 10 characters or less and can be composed of Korean, English, and numerals. The information of the new user, such as name, age, gender, and nickname, is added to the User_Info table corresponding to the user's email in the database. The number of users per email is limited to four. If there are more than two users, the app's start page will allow the selection of a user's name to begin using the app. The feature provides flexibility in switching users, allowing users to easily switch between profiles and personalize their experience on the platform.

V. Architecture Design & Implement

A. Overall Architecture

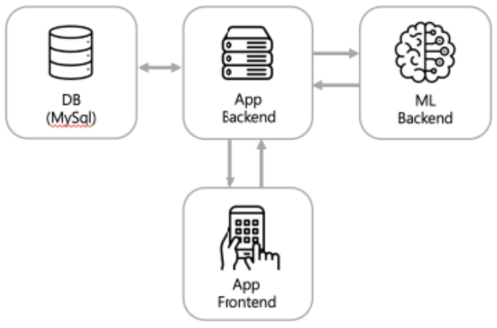


Figure 1. Overall Architecture

Sook Sook application is structured into four main architectural components:

1) App Frontend: The frontend of the application is developed using Flutter. Flutter is chosen for its versatility, allowing developers to write code once and deploy it on both iOS and Android platforms. This cross-platform compatibility significantly reduces development effort and time. Additionally, Flutter offers a rich set of libraries and widgets that enable the implementation of various features and user interface elements, enhancing the user experience.

2) App Backend: The backend of the application is built using Node.js. Node.js is selected for its ease of learning, as it uses JavaScript, a widely-known programming language. Node.js also provides access to a vast ecosystem of modules and libraries, including MySQL for database interactions and Nodemailer for email-related functionalities. The decision to use Node.js is driven by its ability to handle asynchronous tasks efficiently, making it well-suited for building responsive and scalable server-side applications.

   The server is hosted on an Amazon Web Services (AWS) EC2 environment, enabling the setup of a domain and allowing external access to the application. AWS EC2 provides scalable compute capacity, and the use of this cloud environment ensures that the application can handle varying levels of traffic and maintain high availability. It also allows for easy scaling as the user base grows.

3) Database (DB): MySQL is chosen as the relational database management system (RDBMS) for the Sook Sook application. MySQL is well-suited for this application for several reasons:

- Structured Data: MySQL excels at managing structured data, which is crucial for storing user information, hair-related data, and other structured records.
- Data Integrity: MySQL ensures data integrity through features like constraints, foreign keys, and transactions, ensuring that the data remains accurate and consistent.
- Query Capabilities: MySQL offers powerful querying capabilities, enabling efficient retrieval and analysis of data.

The application's MySQL database is hosted on AWS RDS (Relational Database Service), which provides managed database services. AWS RDS handles database administration tasks such as backups, software patching, and scaling, allowing the development team to focus on application development.

4) ML Backend: The machine learning (ML) backend utilized the efficientnet-b0 library for ML model development, leveraging Transfer Learning capabilities. Three AI models were constructed for the application:

- Hair Loss Type and Severity Analysis Model: This model helps classify and assess the severity of hair loss based on input data. It assists users in understanding their hair loss condition.
- Hair Thickness Analysis Model: This model analyzes hair thickness, providing insights into the user's hair characteristics.
- Hair Density Analysis Model: This model evaluates hair density, which is valuable information for understanding the user's hair health.

Google Colab is chosen for its accessibility and ease of use in the machine learning field. It provides a cloud-based environment with pre-installed libraries, making it convenient for ML model development without the need for extensive setup. Additionally, Google Colab offers GPU support, accelerating the training of machine learning models.

In summary, the architectural choices made for the Sook Sook application are driven by the need for versatility, ease of development, scalability, data integrity, and efficient machine learning model development. These components work together to create a robust and user-friendly application that provides valuable insights into hair health and related information.

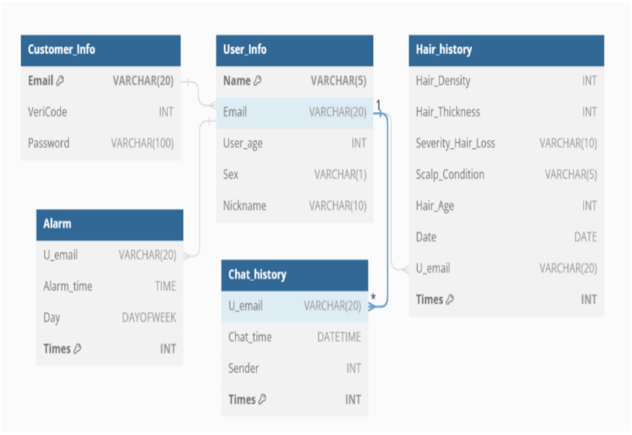


Figure 2.Database

The backend server is implemented using Node.js and is executed through Express. It connects to the MySQL database using the Node.js MySQL library. To achieve this, the necessary database connection information is stored in a .env file and retrieved as follows:

```
var connection = mysql.createConnection({
  host: config.database.host,
  port: config.database.port,
  user: config.database.user,
  password: config.database.password,
  database: config.database.database
});
```

The choice of Node.js for the backend server is made for several reasons. Node.js is known for its non-blocking, event-driven architecture, making it highly scalable and efficient for handling asynchronous tasks, which is crucial in web applications. Express is a popular web application framework for Node.js, offering a robust set of features and middleware for building RESTful APIs and web services. These technologies enable us to build a fast and responsive backend server that can handle a high volume of requests and database interactions.

1. Sook Sook's application uses MySQL as its database system to store data. The decision to use a relational database like MySQL is driven by the need to manage structured data and relationships efficiently. Relational databases provide data integrity, consistency, and support for complex queries, making them suitable for applications that involve data with well-defined structures and associations.

2. User's hair loss information, including hair density, thickness, loss type, scalp condition, hair age, date, user email, and times, is stored in the Hair_history table. The choice of using a separate table for hair loss information allows for organized data storage and efficient querying based on specific attributes. This structured approach makes it easier to analyze and retrieve hair-related data for user profiles and assessments.

3. During the registration process, user information is stored in the Customer_Info table. The Customer_Info table includes columns for email, verification code, and password.

Storing user registration information separately provides security and access control, ensuring that sensitive data such as passwords are properly protected.

4. After registration, the initial user information is stored in the User_Info table. This information includes the user's name, email, age, sex, and nickname. Additionally, this table is used to assess the user's hair loss severity by comparing it to the average hair loss severity for their age group. Storing this information separately enables personalized assessments and recommendations based on the user's profile and hair loss status.

5. All backend components are deployed on the Amazon SWeb Service (AWS) cloud computing platform. The main server and the second server are deployed on separate AWS Elastic Compute Cloud (EC2) instances, while the database is hosted on AWS Relational Database Service (RDS). This cloud-based deployment offers scalability, reliability, and accessibility, ensuring that the application can handle varying workloads and provide high availability to users.

6. The use of AWS services, such as EC2 and RDS, offers various advantages, including automatic scaling, data backup, and security features. AWS's infrastructure-as-a-service (IaaS) and database-as-a-service (DBaaS) offerings simplify the management and maintenance of the backend components, allowing the development team to focus on building and enhancing the application's features and functionality.

Table IV: Database

Table name	***Column name (Datatype)***	***Constraints***

Customer_Info	Email(VARCHAR(20))
	VeriCode(INT)
	Password(VARCHAR(100))
		Email : PK, Not Null
		VeriCode : Not Null
		Password : default/NULL

Hair_history	Hair_Density(INT)
	Hair_Thickness(INT)
	Severity_Hair_Loss (VARCHAR(10))
	Scalp_Condition(VARCHAR(5))
	Hair_Age(INT)
	Date(DATE)
	U_email(VARCHAR(20))
	Times(INT)
		Times : PK, Not Null
		U_email : UQ, Not Null
		Others : default/NULL

User_Info	Name(VARCHAR(5))
	Email(VARCHAR(20))
	User_age(INT)
	Sex(VARCHAR(1))
	Nickname(VARCHAR(10))
		Name : PK, Not Null
		Email : Not Null
		Others : default/NULL

Chat_history	U_email(VARCHAR(20))
	Chat_time(DATETIME)
	SENDER(INT)
	Times(INT)
		Times : PK, Auto Increment, Not Null
		Others : Not Null

Alarm	U_email(VARCHAR(20))
	Alarm_time(TIME)
	Day(DAYOFWEEK)
	Times(INT)
		Times : PK, Auto Increment, Not Null
		Others : Not Null

B. Directory Organization

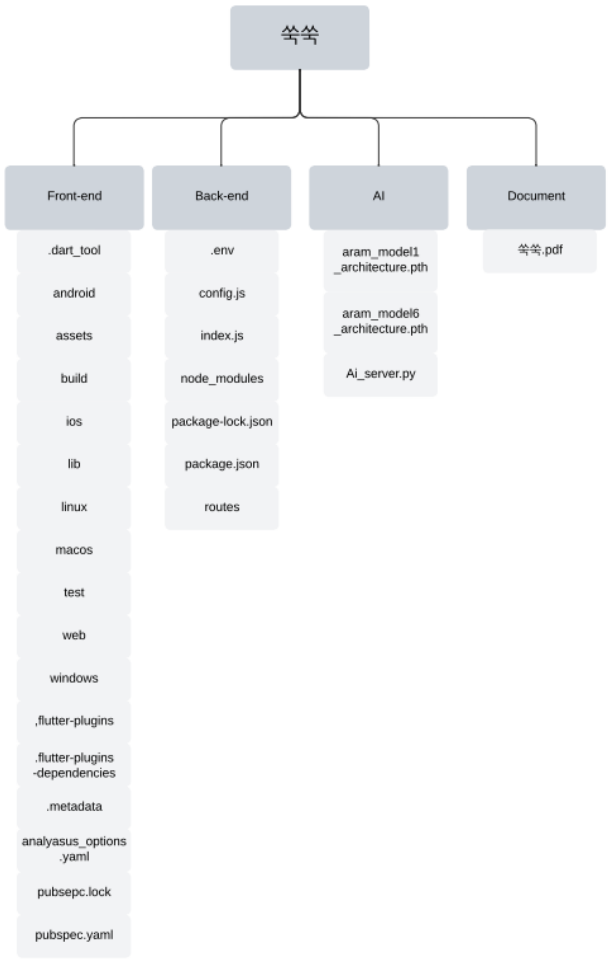


Figure 3. File Directory

a. *Front-end*

Directory	***File name***	***Library***
SE_project/ SE_project/	.dart_tool android assets build ios lib linux macos test web windows .flutter-plugins .flutter-plugins-dependencies .metadata analysis_options.yaml pubspec.lock pubspec.yaml README.md	package:flutter/material.dart dart:async package:http/http.dart dart:convert dart:io package:image_picker/image_picker.dart package:fl_chart/fl_chart.dart package:webview_flutter/webview_flutter.dart
SE_project/assets/images/	analysis.png Analyzing.png background.png backimage.png boldman.png chatbot.png Googleimage.png hairlossman.png history.png Kakaoimage.png loadingnam.png myphoto.png Naverimage.png product_recommand.png product.png	
SE_project/android/	app gradle build.gradle gradle.properties local.properties settings.gradle	io.flutter.plugins androidx.annotation.Keep androidx.annotation.NonNull io.flutter.Log io.flutter.embedding.engine.FlutterEngine
SE_project/ios/	Flutter Runner Runner.xcodeproj Runner.xworkspace	UIKit Flutter
SE_project/ios/Runner	Asset.xcassets Base.lproj AppDelegate.swift GeneratedPluginRegistrant.h GeneratedPluginRegistrant.m Info.plist Runner-Bridging-Header.h Runner.xcodeproj Runner.xcworkspace	GeneratedPluginRegistrant.h image_picker_ios/FLTImagePickerPlugin.h webview_flutter_wkwebview/FLTWebViewFlutterPlugin.h
SE_project/lib/	analysislodinng.dart analysispage.dart analysisresult.dart camera.dart chatpage.dart findpassword.dart graph.dart legistrationscreen.dart lodingscreen.dart loginpage.dart main.dart mainpage.dart noticepage.dart profile.dart recordpage.dart re_password.dart settingpage.dart surveypage.dart	package:flutter/material.dart dart:async package:http/http.dart dart:convert dart:io package:image_picker/image_picker.dart package:fl_chart/fl_chart.dart package:webview_flutter/webview_flutter.dart dart_openai flutter_local_notifications timezone cupertino_icons intl flutter_native_timezone flutter_app_badger permission_handler url_launcher

b. Back-end

Directory	***File name***	***Library***
Back_server/	.env config.js app.js node_modules package-lock.json package.json routes	express body-parser mysql
Back_server/routes	Chat_Save.js Choosen_date.js Choose_date.js Community.js Get_Nickname.js Get_U_Name.js Hair_Analyze.js Load_Chat.js login.js Record_choice.js register.js send_verification.js Solution.js submit_info.js Today_condition.js verify_code.js	express nodemailer body-parser mysql crypto

c. AI

Directory	***File name***	***Library***
AI/	AI.server.py aram_model1_architecture.pth aram_model6_architecture.pth	Pandas Numpy cv2 sklearn.model_selection.train_test_split tensorflow.keras.models.Sequential tensorflow.keras.layers.Conv2D, MaxPooling2D, Flatten, Dense tensorflow.keras.preprocessing.image.img_to_array, load_img tensorflow.keras.utils.to_categorical torch.nn torch.optim torch.optim.lr_scheduler torchvision.datasets torchvision.transforms Flask, request, jsonify Torch, transform Image EfficientNet Image BytesIO

C. Modules

1. Front-end

a. Purpose

To develop Sook Sook, we used Flutter so that a it can be used in a cross-platform for both Android and iOS environment. Frontend serves as a role in providing interface to a user. It connects user and server by providing input fields that a user can enter information. Once this job is done, the values are delivered to backend. Also, Frontend brings data in the database and provides it to the user after changing it to a data that a user can understand.

b. Functionality

Sook Sook application, developed using Flutter, encompasses a wide range of features for comprehensive user engagement. It begins with a robust login page, ensuring secure access with stringent criteria for email and password inputs. Users can also navigate to the 'Forgot Password' and 'Sign-up' pages for account recovery and registration, respectively. The registration process is email-based, featuring email verification and secure password storage. The User Information Entry page collects essential personal data for tailored hair loss assessment. The Main Page offers various functionalities, including hair analysis, record reviews, chatbot interaction, access to a hair loss community, and product integration. The Hair Analysis Page enables users to capture and analyze hair images, receiving personalized recommendations based on the results. Users can review their analysis records on the Hair Record Page. The Settings section offers connection status checks, manuals, and dark mode. The Alarm Page facilitates alarm management for timely Medihair usage reminders. Profile management allows multiple user profiles, enhancing personalization. These features collectively provide a comprehensive and user-friendly platform for hair health management and treatment.

c. Location of source code

https://github.com/SEproject-Medihair/Front_End

d. Class component

- *ios folder*

  This is a folder containing a file that replaces the Dart-written Flutter code with the iOS-only swift code.

- *Android folder*

  This is a folder containing a file that replaces the Dart-written Flutter code with the Android-only java&Kotlin code.

- *lib folder*

  This folder contains Dart code, what was implemented the core logic and user interface of Sook Sook app.

- *assets floder*

  In the assets folder there are stored images, fonts, and other resource files that Sook Sook app uses.

- *lib/analysisloding.dart*

1) Libraries Used and Their Purpose:

   - *import 'package:flutter/material.dart';*

   This library is used to import Flutter's UI components and Material Design for building the user interface. It's essential for creating the visual elements and structure of the app.

   - *import 'dart:async';*

   The Dart async library is imported to handle asynchronous operations. In this code, it's used to implement a timer for delaying the execution of a function.

   - *import 'dart:convert';*

   The Dart convert library is imported for encoding and decoding JSON data. In this code, it's used to encode the data before sending it to a server.

   - *import 'package:http/http.dart' as http;*

   This library is imported to make HTTP requests to a remote server. It's used to send data to the server and handle the server's response.

   - *'analysisresult.dart';*

   This import references another Dart file, likely defining the Analysisresult widget, which is used to display analysis results. It enables code modularity and separation of concerns.

2) Code Functionality:

   - Analysisloding is a StatefulWidget that represents a loading screen during a data analysis or processing task. It receives the user's email, one, and two as parameters.
   - initState(): This method is called when the widget is initialized. It sets a 4-second timer that, when elapsed, calls the sendHairAnalysis function. The purpose of this delay is to simulate a loading screen before sending data to a server.
   - sendHairAnalysis(): This asynchronous function sends a POST request to a server at a specified URL, passing the user's email, one, and two as JSON data. If the server responds with a status code of 200 (success), it navigates the user to the Analysisresult widget to display the analysis results. Otherwise, it handles the error by printing the server's status code.
   - build(): This method is responsible for building the widget's UI. The UI consists of a black background (Colors.black87) with an animated GIF displayed on top. The Image.asset widget loads a GIF image, and it's positioned using a Stack with Positioned widgets to create the loading animation effect.

3) Code Purpose:

   - The primary purpose of this code is to create an "Analysisloding" widget, which serves as a loading screen during data analysis. It visually indicates to the user that the app is actively working on a task. After a 4-second delay, it automatically sends user data to a remote server for analysis. Upon successful analysis, it navigates the user to the Analysisresult widget to display the results. This loading screen enhances the user experience by providing visual feedback during potentially time-consuming operations and separates the analysis logic from the UI logic for maintainability and modularity.

- *lib/analysispage.dart*

1) Libraries Used and Their Purpose:

   - *import 'package:flutter/material.dart';*

   This library is essential for building the user interface using Flutter. It provides widgets and tools for creating the UI components, layouts, and styling.

   - *import 'package:hairapp/camera.dart';*

   This import likely references another Dart file defining the Camera widget, which may be related to capturing images or videos for the analysis process.

   - *'analysisloding.dart';*

   This import references another Dart file, likely defining the Analysisloding widget, which is used to display a loading screen during analysis. It allows for code modularity and separation of concerns.

   - *'package:http/http.dart' as http;*

   This library is used for making HTTP requests to external APIs. In this code, it's used to send an HTTP POST request to a server for fetching analysis data.

   - *'dart:convert';*

   This library is used for encoding and decoding JSON data. It's used here to encode the request body as JSON and decode the response from the server.

2) Code Functionality:

   - The code creates a user interface for the "Analysispage," which appears to be related to hair analysis.
   - There is a graphical representation of the user's previous analysis score and status, along with buttons for starting a new analysis and accessing camera-related features.
   - The "분석 시작" (Start Analysis) button navigates the user to the Camera widget, which may be used for capturing images or videos for analysis.

- Information about the previous analysis score and hair conditions is displayed below, providing users with insights into their hair health.

3) Code Purpose:
   - The purpose of this code is to create the user interface for the "Analysispage," where users can initiate a hair analysis, view their previous analysis results, and potentially capture images or videos for analysis.
   - It provides a visually appealing and informative screen for users interested in monitoring and improving their hair health.
   - The buttons allow users to take actions related to hair analysis and camera functionality, enhancing the overall functionality and user experience of the app.

- *lib/analysisresult.dart*

1) Libraries Used and Their Purpose:
   - *import 'package:flutter/material.dart';*

     This library is essential for building the user interface using Flutter. It provides widgets and tools for creating the UI components, layouts, and styling.
   - *import 'package:hairapp/mainpage.dart';*

     This import likely references another Dart file defining the Mainpage widget, which is used to navigate back to the main page of the app.
   - *import 'package:http/http.dart' as http;*

     This import is for making HTTP requests to a remote server to fetch data. It is used to send an email and receive hair analysis results from a server.
   - *import 'dart:convert';*

     This import allows for encoding and decoding JSON data, which is useful for handling data fetched from the server.

2) Code Functionality:
   - The code creates a user interface for the "Analysisresult" page, where the results of a hair analysis are displayed.
   - It sends an HTTP POST request to a server to retrieve hair analysis data based on the provided email address (widget.email).
   - The retrieved data is displayed on the screen, including the user's name, hair density, hair thickness, hair loss type, scalp condition, and more.
   - The page also suggests recommended hair care based on the analysis results.
   - The time taken for the analysis is determined using a Timer, and this duration is displayed on the screen.
   - There is a button labeled "메인으로" (Back to Main), which allows the user to navigate back to the main page of the app (Mainpage widget) when pressed.

3) Code Purpose:
   - The purpose of this code is to present the results of a hair analysis to the user in a user-friendly manner.
   - It sends an email to a server, receives the analysis results, and updates the UI with the retrieved data.
   - Users can view details about their hair health, such as hair loss type, hair density, hair thickness, and scalp condition.
   - The "메인으로" button provides a convenient way for users to return to the main page of the app after viewing their analysis results.
   - Additionally, the code displays recommended hair care information based on the analysis results, enhancing the user's understanding of how to improve their hair health.

- *lib/camera.dart*

1) Libraries Used and Their Purpose:
   - *dart:io*

     This library is used for working with files and directories. In this code, it's used to handle and display images from the device's camera or gallery.
   - *package:flutter/material.dart*

     This is the core Flutter library for building the user interface (UI). It provides widgets, themes, and tools for creating UI components.
   - *package:flutter/services.dart*

     This Flutter package provides access to the device's platform-specific services and methods. In this code, it's used for setting the preferred screen orientation.
   - *package:image_picker/image_picker.dart*

     This package provides an easy way to access and pick images from the device's camera or gallery. It's used for capturing or selecting images.
   - *package:http/http.dart as http*

     This package is used for making HTTP POST requests to a server. It's used to upload the selected/captured image to a server for analysis.

2) Code Functionality:

- This code creates a Flutter widget called "Camera" that allows users to take a photo using the device's camera or select an image from the device's gallery.
- It uses the ImagePicker class from the image_picker package to interact with the device's camera and ga-llery.
- When the user taps on the camera icon button, it opens the device's camera for capturing a photo.
- When the user taps on the wallpaper icon button, it opens the device's gallery for selecting an image.
- The selected or captured image is displayed in the UI using the showImage method.
- Upon tapping the forward arrow button, the code sends the selected/captured image to a server for analysis. The server's response is processed, and the user is navigated to the "Analysisloding" widget with the analysis results.
- The response text from the server is also displayed to inform the user about the status of the analysis.

3) Code Purpose:

- The purpose of this code is to create a Flutter app screen that allows users to interact with the device's camera and gallery to select or capture images.
- It's typically used as part of a larger app where users need to provide images for analysis or other purposes.
- The code's primary goal is to provide a user-friendly interface for image selection, capture, and analysis, enhancing the user experience.

- *lib/chat_page.dart*

1) Libraries Used and Their Purpose:

- *package:flutter/material.dart*

  This is the core Flutter framework library used for building user interfaces. It provides the necessary tools for creating visual elements and structures in the chat screen.

- *package:http/http.dart as http*

  This library is utilized for making HTTP requests. In this context, it's used to send messages to a server, specifically to the OpenAI API.

- *dart:convert*

  This library is used for encoding and decoding JSON data, which is essential for handling the data exchanged between the Flutter app and the server.

2) Code Functionality:

- The code defines a stateful widget ChatPage that handles the chat interface.
- The _ChatPageState class maintains a list of messages and their types (user or bot).
- The sendMessageToGPT function sends user messages to the OpenAI API and retrieves responses. It uses the HTTP POST method to interact with the API.
- The time taken for the analysis is determined using a Timer, and this duration is displayed on the screen.
- Messages are displayed in a ListView and styled differently based on whether the message is from the user or the bot.
- The chat interface includes a TextField for inputting messages and an IconButton for sending them.
- When a message is sent, it's immediately added to the chat screen, and the bot's response is added once received from the server.

3) Code Purpose:

- This code is primarily intended to create a chat interface in a mobile application using Flutter.
- It facilitates real-time communication with an AI assistant powered by OpenAI's GPT model.
- The widget is designed to handle both sending user messages to the server and displaying responses from the bot, creating an interactive chat experience.
- It also includes error handling for failed requests, ensuring the robustness of the chat feature.

- *lib/legistrationscreen.dart*

1) Libraries Used and Their Purpose:

- *package:flutter/material.dart*

  This is the core Flutter library for building the user interface (UI). It provides widgets, themes, and tools for creating UI components.

- *package:http/http.dart as http*

  This package is used to make HTTP requests to interact with a server. In this code, it's used for sending registration and verification requests to the server.

- *dart:convert:*

  This library is used for encoding and decoding JSON data. In this code, it's used to handle JSON responses from the server.

- *dart:async*

  This library is used to work with asynchronous operations. It's used for implementing timers.

2) Code Functionality:

- This code creates a Flutter widget called "RegistrationScreen" for user registration.
- It includes several text input fields for email, verification code, password, and password confirmation.
- Users can request a verification code via email, verify the code, and then register with their email and password.
- It implements a timer to control the resend verification code functionality.

3) Code Purpose:

- The purpose of this code is to create a user registration screen for an app.
- Users can enter their email, receive a verification code, verify their email, set a password, and register for the service.
- The code handles various validation checks, such as password matching, length, and character requirements.
- It also displays informative dialogs to users in case of errors or successful registration.

In summary, this code is designed to provide a user-friendly registration experience within a Flutter app, including email verification and password setup, and it communicates with a server to facilitate the registration process.

- *lib/lodingscreen.dart*

1) Libraries Used and Their Purpose:

- dart:async: This Dart library is used for working with asynchronous operations. In this code, it's used to create a delay (4 seconds) before navigating to the login page.
- loginpage.dart: This is a user-defined module (assumed to be part of your Flutter project) that likely contains the login page widget. It's imported so that the application can navigate to it when the splash screen is done.

2) Code Functionality:

- This code creates a Flutter widget called "SplashScreen," which serves as the initial screen displayed when the app is launched.
- Inside the initState method, a Timer is set to wait for 4 seconds. After this delay, it navigates to the login page using Navigator.pushReplacement.
- The Navigator.pushReplacement method replaces the current screen (the splash screen) with the login page, ensuring that users cannot navigate back to the splash screen using the back button.

3) Code Purpose:

- The purpose of this code is to create a splash screen for the app, which is typically used to display a loading indicator, logo, or any other introductory content when the app is launched.
- The splash screen provides a smooth transition to the main content of the app, in this case, the login page.
- By delaying the navigation to the login page, it gives the impression of a brief introduction or branding display before users access the app's core functionality.

In summary, this code is designed to create a splash screen that briefly appears when the app is launched, and after a 4-second delay, it automatically navigates the user to the login page. The purpose is to enhance the user experience and provide a polished entry point to the app.

- *lib/loginpage.dart*

1) Libraries Used and Their Purpose:

- *package:flutter/material.dart*

  This is the core Flutter framework library for building user interfaces. It's used to create the visual components and structure of the login screen.

- *package:http/http.dart as http*

  This library is used to make HTTP requests to a remote server for user authentication and login.

- *dart:convert*

  It's used for encoding and decoding JSON data, which is commonly used in communication between the Flutter app and a server.

- *dart:async*

  This Dart library is used for handling asynchronous operations such as setting timers for delayed actions.

- legistrationscreen.dart, findpassword.dart, analysispage.dart, recordpage.dart, chatpage.dart: These are user-defined modules (assumed to be part of your Flutter project) that contain various screens or components used for navigation.

2) Code Functionality:

- This code creates a Flutter widget called "LogIn," which represents the login screen of the application.
- Users can enter their email and password, and upon pressing the "Log in" button, the app sends a POST request to a remote server for authentication.

- If the login is successful, a dialog is shown with a success message, and the user is navigated to the "Analysispage" screen.
- The screen also provides options for password recovery ("비밀번호 찾기") and user registration ("회원가입").

3) Code Purpose:

- The purpose of this code is to create the user interface for the login screen of the mobile application.
- Users can log in by providing their email and password, and the app communicates with a server for authentication.
- If the authentication is successful, the user is directed to the "Analysispage" screen.
- The screen also offers options for password recovery and new user registration.
- The "LogIn" widget serves as the entry point for user authentication and access to the app's features.

In summary, this code is responsible for creating the login screen of the mobile application, handling user authentication, and providing navigation options for password recovery, user registration, and social login. It plays a crucial role in the user onboarding process and app security.

- *lib/mainpage.dart*

1) Libraries Used and Their Purpose:

- *package:flutter/material.dart*

  This is the core Flutter framework library for building user interfaces. It's used to create the visual components and structure of the main page of the application.

2) Code Functionality:

- This code creates a Flutter widget called "Mainpage," representing the main screen of the application.
- The main page displays various sections, including:
- User's profile picture and a welcome message.

Four interactive sections:

"모발분석" (Hair Analysis)

"분석기록" (Analysis Records)

"추천제품" (Recommended Products)

"챗봇" (Chatbot)

- An image banner section.
- A bottom navigation bar with icons and labels for "설정" (Settings), "홈" (Home), "알림" (Notifications), and "마이" (Profile).

3) Code Purpose:

- The purpose of this code is to create the main user interface for the application's main screen.
- Users are greeted with a welcome message and their profile picture.
- The main screen provides quick access to various sections, such as hair analysis, analysis records, recommended products, and a chatbot.
- Users can navigate to different sections by tapping on the respective section buttons.
- The bottom navigation bar allows users to access settings, return to the home screen, check notifications, and access their profile.
- Overall, this "Mainpage" widget serves as the central hub for users to interact with the app's main features and navigate to different sections of the application.

In summary, this code creates the main screen of the mobile application, allowing users to access various sections and features of the app. It provides a visually appealing and interactive user interface for the application's main functionality.

- *lib/noticepage.dart*

1) Libraries Used and Their Purpose:

- package:flutter/material.dart: This library is used for creating the user interface (UI) of the Flutter app. It provides various widgets and tools for building the app's UI components, including buttons, text, and layout.
- package:intl/intl.dart: This package is used for formatting date and time. It allows you to format dates and times in a human-readable way, which is useful for displaying alarms with specific date and time formats.

2) Code Functionality:

- The code defines a Flutter widget called "Noticepage," which represents a page in the app.
- Inside this page, the code manages a list of alarms using the alarms list. Each alarm is represented by the AlarmInfo class, containing a dateTime (the date and time of the alarm) and an isActive flag (indicating whether the alarm is active or not).
- Users can add new alarms by tapping the "add" button. This opens a date picker to select a date and a time picker to select a time. The selected date and time are used to create a new AlarmInfo instance, which is added to the alarms list.

- Existing alarms are displayed in a scrollable list using a ListView.builder. Each alarm is displayed with its day of the week, time, and a switch to activate or deactivate it. Users can also delete alarms by tapping the delete icon.
- The bottom part of the screen contains a FloatingActionButton for adding new alarms.
- A bottom navigation bar with icons is also provided. It allows users to navigate to other pages of the app, such as the settings page, main page, notifications page, and user profile page.

3) Code Purpose:

- The purpose of this code is to create a user interface for managing alarms within the app. Users can add, view, activate, deactivate, and delete alarms based on specific dates and times.
- This feature is often part of apps that require scheduling and reminders, such as alarm clocks, calendar apps, or event reminder apps.
- The code ensures that the user experience is smooth and intuitive by providing a user-friendly interface for managing alarms and easy navigation to other app sections through the bottom navigation bar.

- lib/settingpage.dart

1) Libraries Used and Their Purpose:

- package:flutter/material.dart: This is the core Flutter framework library for building user interfaces. It's used to create the visual components and structure of the settings page.

2) Code Functionality:

- This code creates a Flutter widget called "Settingpage," representing a settings screen within the application.
- The settings screen displays various settings options, each represented as a list item. These options include "MY 메디헤어," "메디헤어 가이드라인," "다크모드," and "고객지원."
- Each setting option is displayed with a title in a card-like container.
- There are navigation buttons at the bottom of the screen, allowing users to switch between different sections of the application: "설정" (Settings), "홈" (Home), "알림" (Notifications), and "마이" (My Page).

3) Code Purpose:

- The purpose of this code is to create a user interface for managing various settings and options within the mobile application.
- Users can navigate to this screen and view different settings options, such as "MY 메디헤얼," "메디헤어 가이드라인," "다크모드," and "고객지원," and potentially configure or access more details related to these settings.
- The navigation buttons at the bottom of the screen provide a convenient way for users to move between different sections of the app, such as the home screen, notifications, and user profile.
- Overall, the "Settingpage" widget serves as a central hub for app settings and navigation, enhancing the user experience by providing easy access to configuration options and different parts of the application.

- *lib/surveypage.dart*

1) Libraries Used and Their Purpose:

- package:flutter/material.dart: This is the core Flutter framework library for building user interfaces. It's used to create the visual components and structure of the survey page.

2) Code Functionality:

- This code creates a Flutter widget called "Surveypage," representing a survey or questionnaire page within the application.
- The survey page displays a series of questions related to the user's ID, age, gender, hair loss symptoms, current treatments, and family history of hair loss.
- Each question is accompanied by a text input field where the user can provide their answers or input.
- There is a "Submit" button at the bottom of the screen to allow users to submit their survey responses.
- The page also includes a back button in the app bar that allows users to navigate back to the previous screen.

3) Code Purpose:

- The purpose of this code is to create a user interface for conducting a survey within the mobile application.
- Users can navigate to this screen to provide information related to their hair loss, such as ID, age, gender, and other relevant details.
- The survey responses can be collected and processed by the application for various purposes, such as personalized recommendations or data analysis.
- The "Submit" button allows users to finalize and submit their survey responses.

Overall, the "Surveypage" widget serves as an essential component for gathering user data and feedback through a user-friendly interface.

- *pubspec.yaml;*

  This file includes app dependencies and configuration information. efine metadata like the app's name, version, and list the Flutter packages your app relies on.

2. Back-end

a. Purpose

The backend is responsible for managing servers and databases. It stores and manages data, as well as handles actions taken by users on the client side of the application. The backend's roles include storing data generated as a result of user behavior in the frontend database (Front Desk) and retrieving data required by users from the frontend database (Front Desk).

Sook Sook chose to implement the backend using Node.js and MySQL. Node.js was chosen due to its extensive selection of modules, offering a wide range of functionalities for implementation. MySQL, a relational database, was selected for its ability to organize data into tables

b. Functionality

Sook Sook's application uses an AI model for analyzing the condition of hair loss and scalp health, and it stores the extracted results in a database. The backend server comes into play during the process of sending photos to the server and storing the extracted results in the database. Additionally, it provides functionality to check the analysis results on a daily basis, which involves retrieving data from the database and sending it to the frontend. Other key features include login, user registration, fetching analyzed dates, record verification, and comparing analysis results on a daily basis.3. Location of source code

c. Location of source code

https://github.com/SEproject-Medihair/Front_End

d. Class component

- *.env*

  This file contains information for Gmail's ID and password for sending authentication codes, as well as the connection address for MySQL RDS.

- *package.json & package-lock.json*

  In a Node.js project, the package.json file serves as a configuration file that contains project details, script commands, and dependency information, allowing developers to define the project's settings, including the required external packages and their versions. On the other hand, the package-lock.json file is generated automatically and provides a detailed record of the project's dependency tree, specifying the exact versions of packages and their interdependencies. It ensures consistency and reproducibility by enabling precise package version control during project replication or deployment in different environments.

- *app.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is a popular web application framework for Node.js. It is used to create the server and handle routing, making it easier to build web APIs and applications.
   - body-parser: The body-parser middleware is used to parse incoming request bodies in JSON format. It's essential for handling JSON data sent in HTTP requests.
   - mysql: The mysql library is used to establish a connection to a MySQL database. It allows the application to interact with and query the database.
   - config: The config module is likely used to store configuration settings for the application, such as database connection details and other environment-specific configurations.
2) Code's Purpose:
   - The code establishes a connection to a MySQL database using the provided configuration.
   - The code defines various routes and uses separate route files (e.g., loginRoute, sendVerificationRoute, etc.) for each route to handle specific API endpoints.

- *routes/Choosen_date.js*

1) Libraries Used and Their Purpose:
   - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries against it.
   - config: The config module stores configuration settings, likely including database connection details, separately from the code.
2) Code's Purpose:
   - This code handles HTTP POST requests at the root route ('/').
   - It expects the user's email and a specific date as inputs in the request body.
   - The code constructs an SQL query to select hair-related data (density, thickness, etc.) for the given user (U_email) and date from the 'Hair_history' table.
   - Additionally, it queries the 'User_Info' table to retrieve the user's name based on their email.
   - The code executes these SQL queries, handles potential errors, and responds with the requested data and the user's name in JSON format.

- *routes/Choose_date.js*

1) Libraries Used and Their Purpose
   - mysql: The mysql library is utilized to establish a connection to a MySQL database. It allows the code to execute SQL queries against the database to fetch specific data.
   - config: The config module is used for storing configuration settings, likely including database connection details, to keep sensitive information separate from the code.
2) Code's Purpose:

- The code defines an Express.js router to handle HTTP POST requests at the root route ('/').
- It retrieves the user's email from the request body and constructs a SQL query to select distinct dates from the 'Hair_history' table associated with the given email.
- Upon executing the query, it handles potential errors and returns the distinct dates as an array in ISO date format ('YYYY-MM-DD') to the client.

- *routes/Get_Nickname.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is used to create a web server and define API routes. It simplifies handling HTTP requests and responses.
   - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries against it.
   - config: The config module stores configuration settings, likely including database connection details, separately from the code.
2) Code's Purpose:
   - This code handles HTTP POST requests at the root route ('/').
   - It expects the user's email as input in the request body.
   - This code constructs an SQL query to select the user's nickname from the 'User_Info' table based on their email.
   - This code executes this SQL query, handles potential errors, and responds with the user's nickname and email in JSON format if a matching user is found.
   - If no user with the provided email is found, it responds with a 404 status and an appropriate message.

- *routes/Get_U_Name.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is utilized to create a web server and define API routes. It simplifies handling HTTP requests and responses.
   - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries against it.
   - config: The config module stores configuration settings, likely including database connection details, separately from the code.
2) Code's Purpose:
   - This code handles HTTP POST requests at the root route ('/').
   - It expects the user's email as input in the request body.
   - This code constructs an SQL query to select the user's name from the 'User_Info' table based on their email.
   - This code executes this SQL query, handles potential errors, and responds with the user's name and email in JSON format if a matching user is found.
   - If no user with the provided email is found, it responds with a 404 status and an appropriate message.

- *routes/login.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is used to create a web server and define API routes for handling HTTP requests and responses.
   - crypto: The crypto library is used to hash the user's password securely using the SHA-256 algorithm.
   - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries for user authentication.
   - config: The config module stores configuration settings, including database connection details, separately from the code.
2) Code's Purpose:
   - This code defines a POST route at '/api/login' that expects email and password in the request body.
   - It securely hashes the provided password using SHA-256.
   - The code then queries the database to check if a user with the provided email and hashed password exists.
   - If a matching user is found, it responds with a success message and the user's email.
   - If there is no matching user or the password is incorrect, it responds with an error message.

- */routers/Hair_Analyze.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is used for building the API server and defining routes to handle HTTP requests and responses.
   - mysql: The mysql library is employed to create a connection to a MySQL database and execute SQL queries to fetch hair-related data.
   - config: The config module contains configuration settings, including database connection details.
2) Code's Purpose:
   - This code defines a POST route that expects certain data (email, Condition, and Hair_skin) in the request body.
   - It calculates random hair-related data (Hair_Density, Hair_Thickness, Scalp_Condition) based on the provided Condition and Hair_skin values. These values simulate hair condition data for the user.
   - The current date is obtained in ISO format to match with the date in the database.
   - The code inserts the calculated hair-related data, along with the user's email and current date, into the "Hair_history" table in the database.
   - If the data insertion and retrieval are successful, it responds with a status code 200 and returns the fetched hair-related data along with the user's name.
   - If no data is found for the given email and date, it responds with a status code 404 and an error message.

This code essentially simulates and records hair-related data for a user based on their input and stores it in a database. It also retrieves the user's name and the most recent hair-related data for that user to provide a response.

- *routes/Record_choice.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is used to create a web server and define API routes for handling HTTP requests and responses.
   - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries to retrieve user-specific data.
   - config: The config module stores configuration settings, including database connection details, separately from the code.
2) Code's Purpose:
   - This code defines a POST route that expects an email and a date in the request body.
   - It queries the database to retrieve hair-related data (Hair_Density, Hair_Thickness) for the provided email and date.
   - Additionally, it fetches the user's name from the database based on their email.
   - If a matching record is found in the database, it responds with the fetched hair data and the user's name.
   - If there's no matching record, it responds with an error message indicating that no data was found for the given email and date.

- *routes/register.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is used to create a web server and define API routes for handling HTTP requests and responses.
   - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries to update user data.
   - config: The config module stores configuration settings, including database connection details, separately from the code.
   - crypto: The crypto module is used to hash the user's password before storing it in the database for security reasons.
2) Code's Purpose:
   - This code defines a POST route that expects an email and a new password in the request body.
   - It hashes the provided password using the SHA-256 algorithm for security.
   - The code then executes an SQL query to update the password in the Customer_Info table of the MySQL database for the user with the specified email.
   - If the update is successful, it responds with a status code 200 and a message indicating that the registration is complete.

- *routes/send_verification.js*

1) Libraries Used and Their Purpose:
   - express: Express.js is used to create a web server and define API routes for handling HTTP requests and responses.
   - nodemailer: The nodemailer library allows sending emails from the application. It's used here to send a verification code to the user's email address.
   - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries to check if the user already exists and to insert user information.
   - crypto: The crypto module is used to generate a random verification code and hash user passwords for security.
   - config: The config module stores configuration settings, including database connection details and email credentials.
2) Code's Purpose:
   - This code defines a POST route that expects an email address in the request body for user registration.

- It generates a random 4-digit verification code.
- The code checks if the provided email already exists in the database. If it does, it returns a status code 400 with a message indicating that the email is already registered.
- If the email is not found in the database, it inserts the email and verification code into the Customer_Info table in the MySQL database.
- It then sends an email to the user's provided email address containing the verification code.
- If the email is sent successfully, it responds with a status code 200 and a message indicating that the email has been sent.

- *routes/submit_info.js*

1) Libraries Used and Their Purpose:
    - express: Express.js is used to create a web server and define API routes for handling HTTP requests and responses.
    - mysql: The mysql library is employed to establish a connection to a MySQL database and execute SQL queries to insert user information.
    - config: The config module stores configuration settings, including database connection details.
2) Code's Purpose:
    - This code defines a POST route that expects user information (name, age, email, sex, and nickname) in the request body for user registration or profile update.
    - It validates that both the name and age fields are provided; otherwise, it returns a status code 400 with an error message.
    - It establishes a connection to a MySQL database using the provided configuration.
    - It inserts the user's information into the User_Info table in the MySQL database.
    - If the insertion is successful, it responds with a status code 200 and a message indicating that the information has been successfully saved.

- *routes/Today_condition.js*

1) Libraries Used and Their Purpose:
    - express: Express.js is used for building the API server and defining routes to handle HTTP requests and responses.
    - mysql: The mysql library is employed to create a connection to a MySQL database and execute SQL queries to fetch hair-related data.
    - config: The config module contains configuration settings, including database connection details.
2) Code's Purpose:
    - This code defines a POST route that expects an email in the request body, indicating the user for whom hair-related data is being requested.
    - It retrieves the current date in ISO format to match with the date in the database.
    - It queries the MySQL database for hair-related data (such as hair density, thickness, loss type, scalp condition, age, and date) associated with the provided email and current date.
    - Additionally, it queries the User_Info table to obtain the user's name based on the provided email.
    - If data is found, it responds with a status code 200 and returns the fetched hair-related data along with the user's name.
    - If no data is found for the given email and date, it responds with a status code 404 and an error message.

- *routes/verify_code.js*

1. Libraries Used and Their Purpose:
    - express: Express.js is used for building the API server and defining routes to handle HTTP requests and responses.
    - mysql: The mysql library is employed to create a connection to a MySQL database and execute SQL queries to verify an email and verification code combination.
    - config: The config module contains configuration settings, including database connection details.
2. Code's Purpose:
    - This code defines a POST route that expects an email and verification code in the request body, which are used to verify the user.
    - It queries the MySQL database to check if the provided email and verification code match any records in the Customer_Info table.
    - If a matching record is found, it responds with a status code 200 and indicates that the verification is successful by setting isVerified to true.
    - If no matching record is found, it responds with a status code 400 and an error message indicating that the verification code is incorrect.

3. *AI*

1. Purpose

Our team has developed an efficient deep learning model based on the EfficientNet architecture to predict the stage of hair loss and scalp condition. EfficientNet is a model designed with a balance between performance and model size, and the B0 version represents the most lightweight variant. This architecture ensures high accuracy while efficiently utilizing computational resources. The model is optimized to learn the features necessary for predicting hair loss.

A convolutional neural network (CNN) is employed to extract features from images, allowing the model to learn useful patterns and features from input images. Specifically, the use of EfficientNetB0 facilitates efficient feature extraction, contributing to high-efficiency learning by identifying crucial information related to hair loss. Adaptive pooling further enables the model to adapt to various input image sizes, enhancing its flexibility for deployment in diverse environments.

2. Functionality

The model is designed to classify hair loss levels and scalp conditions into four classes, labeled 0, 1, 2, and 3. Stage 0 represents a normal state, stage 1 indicates mild conditions, stage 2 corresponds to moderate conditions, and stage 3 signifies severe conditions. After training the model, it was tested using scalp images of individuals who have undergone actual hair loss diagnoses.

The hyperparameters were further fine-tuned to enhance the accuracy of the model. This involved adjusting various parameters to achieve better performance. The testing phase involved evaluating the model's performance on scalp images from individuals who have received real hair loss diagnoses.

3. Location of source code

- https://github.com/SEproject-Medihair/Back_End

4. Class component

- *API_Server.py*

1) Libraries Used and Their Purpose:

- Flask: Flask is a micro web framework for Python used for building web applications. In this code, it's used to create a web API for image classification.
- torch: PyTorch is a deep learning framework. In this code, it's used for loading pre-trained neural network models and performing inference on images.
- transforms (from torchvision): It is used to define a series of image transformations to preprocess input images before feeding them into the neural network.
- PIL (Python Imaging Library): PIL is used for opening and manipulating images.
- EfficientNet (from efficientnet_pytorch): This library contains the EfficientNet neural network architecture, which is used for image classification.

2) Code's Purpose:

- This code creates a Flask web application that exposes an endpoint for image classification.
- It defines a function preprocess_image that takes an image file, performs various transformations (resize, flip, rotate, etc.), and converts it into a tensor suitable for neural network input.
- Two functions, predict_image and predict_image2, are defined to perform image classification using pre-trained neural network models. These functions take the model and the preprocessed input tensor, and return the predicted class and probabilities of the image belonging to different classes.
- Pre-trained EfficientNet models (model and model2) for image classification are loaded from files (aram_model6_architecture.pth and aram_model1_architecture.pth) using PyTorch.
- The /predict route in the Flask app accepts image files as POST requests, preprocesses them using preprocess_image, and performs classification using the loaded models (predict_image and predict_image2). It then returns the predicted classes.

- *aram_model1_architecture.pth*

The aram_model1_architecture.pth file used in this code is a pretrained machine learning model file.

1) Input and Preprocessing
This model takes an image file as input. The image is preprocessed to prepare it for analysis. The preprocessing steps include resizing the image to a specific dimension, applying data augmentation techniques like horizontal and vertical flips, rotation, and affine transformations, converting the image to a tensor, and normalizing it.
2) Analysis
The model is designed to analyze the state of the scalp and hair based on the provided image. It performs a detailed analysis and provides an output prediction. The output is a class label, which can be one of the following: 0, 1, 2, or 3. Each class label corresponds to a different assessment or state of the scalp and hair.
3) Training Data
During the development of this AI model, it was trained on a dataset that included images representing different stages (0, 1, 2, 3) of scalp

and hair conditions. The model learned from these images to classify new, unseen images into one of these stages.

In summary, aram_model1_architecture.pth is a pretrained machine learning model used for analyzing scalp and hair conditions based on input images. It has been trained on a dataset containing images of different scalp and hair conditions, allowing it to make predictions about the condition of new images.

- *aram_model6_architecture.pth*

The aram_model6_architecture.pth file, like the aram_model1_architecture.pth file, is also a pretrained machine learning model. However, while both models are used for analyzing scalp and hair conditions, there are key differences in their specific applications and functionalities:

1. Specific Focus on Hair Loss Stages: The primary distinction of aram_model6_architecture.pth is its specialization in categorizing stages of hair loss. It classifies the condition of the scalp and hair into four distinct stages: 0, 1, 2, and 3. Each stage represents a different level of hair loss severity.

2. Input and Preprocessing: Similar to aram_model1_

   architecture.pth, this model likely takes an image file as input. The preprocessing steps would be similar, involving resizing, data augmentation (like flips, rotation, etc.), converting the image to a tensor, and normalization.

3. Analysis and Output

   The analysis process of aram_model6_ architecture.pth is focused on identifying the stage of hair loss. It provides an output prediction, classifying the image into one of the four stages of hair loss. This is the key area where it differs from aram_model1_architecture.pth, which might be analyzing general scalp and hair conditions without the specific focus on hair loss stages.

4. Training Data

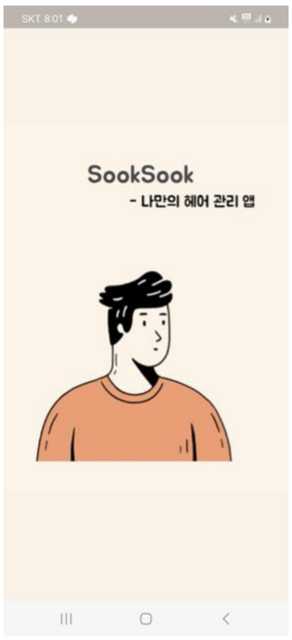
   The training data for aram_model6_architecture.pth would consist of images specifically categorized into the four stages of hair loss. This targeted dataset allows the model to learn and predict the hair loss stage of new, unseen images.

In summary, while both aram_model1_architecture.pth and aram_model6_architecture.pth are machine learning models for scalp and hair analysis, aram_model6_ architecture.pth is specifically tailored to classify the stages of hair loss. This specialized focus is the primary difference between the two models.

VII. Use Cases

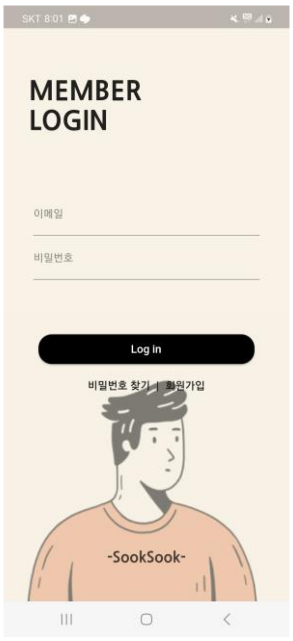
A. *Application Icon*

B. *Loading Page*



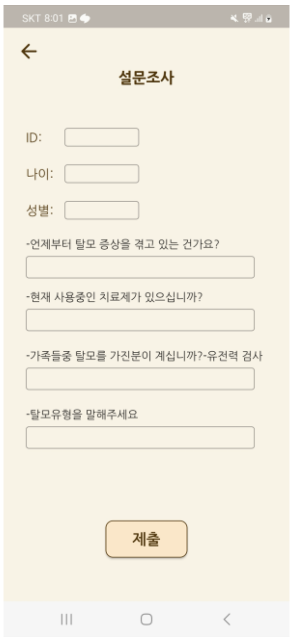
Loading Page is a page that a user will see after turning on the application. This page is turned on only while the application loads the elements needed to operate, and automatically moves on to the next page at the end of loading.

C. *Login Page*



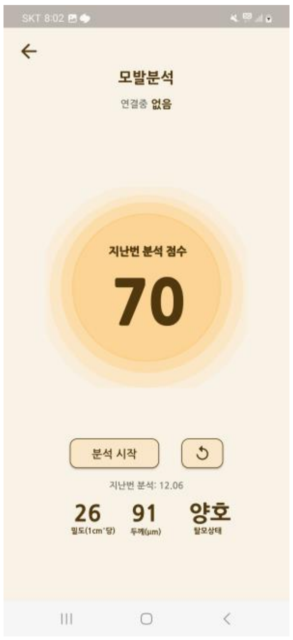
Login Page is a page which allows users to log in by entering their Email and password.

D. *Survey Page*


SKT 8:01
설문조사
ID:
나이:
성별:
-언제부터 탈모 증상을 겪고 있는 건가요?
-현재 사용중인 치료제가 있으십니까?
-가족들중 탈모를 가진분이 계십니까?-유전력 검사
-탈모유형을 말해주세요
제출

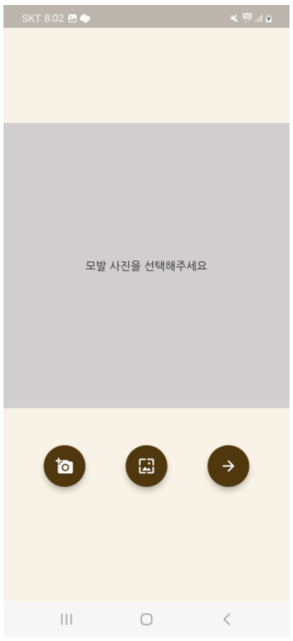
If the user logs in for the first time after signing up, they will be directed to a survey page. This page collects basic information such as the user's ID (nickname), age, gender, etc. This information is stored in the database and is used to provide personalized treatment solutions to the user.

E. *Hair Analysis Page*



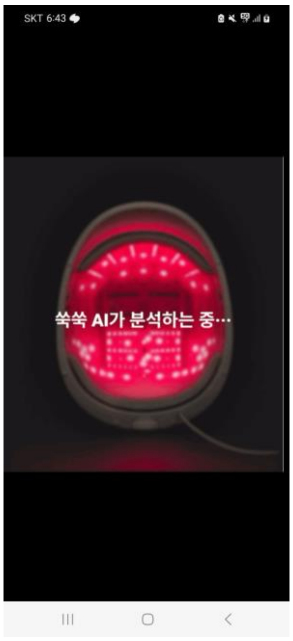
When the hair analysis page starts, it retrieves the most recent analysis information from the database. It displays the user's scalp health score, hair density, thickness, and hair loss condition on the screen. Additionally, you can see the status of the connection to MediHair products on the screen. You can initiate the actual analysis of hair loss and hair condition by clicking the 'Start Analysis' button.

F. *Scalp Photo Upload Page*



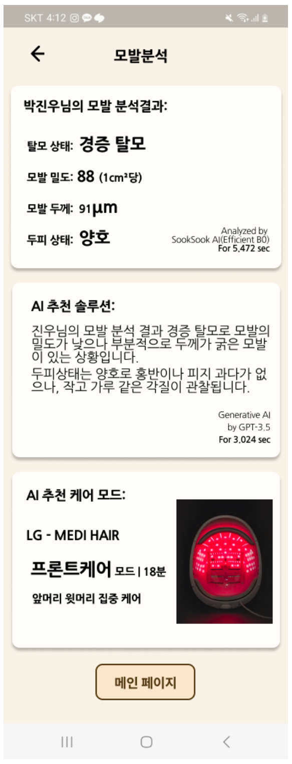
On this page, users can upload photos of their scalp condition. The uploaded photos will be analyzed using an AI model to assess the user's hair loss condition and scalp health. After selecting a photo, users can verify on the screen if the chosen photo is correct. If the selected photo is accurate, they can click the "Submit" button to send the photo to the AI model and proceed to the next page.

G. *Analysis Loading page*



When the user selects a photo and clicks the "Submit" button, this page will be executed. A GIF that simulates a machine scanning process will repeatedly play on the screen until the photo arrives at the server and the analysis is completed. Once the analysis is finished, the user will be redirected to a page where they can review the analysis results.

H. Analysis Result Page


SKT 4:12

모발분석

박진우님의 모발 분석결과:

탈모 상태: 경증 탈모

모발 밀도: 88 (1cm²당)

모발 두께: 91µm

두피 상태: 양호

Analyzed by SookSook AI(Efficient B0) For 5.472 sec

AI 추천 솔루션:

진우님의 모발 분석 결과 경증 탈모로 모발의 밀도가 낮으나 부분적으로 두께가 굵은 모발이 있는 상황입니다.
두피상태는 양호로 홍반이나 피지 과다가 없으나, 작고 가루 같은 각질이 관찰됩니다.

Generative AI by GPT-3.5 For 3.024 sec

AI 추천 케어 모드:

LG - MEDI HAIR

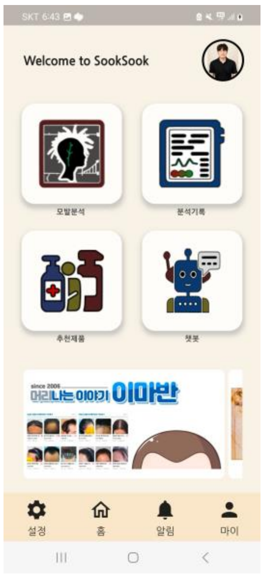
프론트케어 모드 | 18분

앞머리 윗머리 집중 케어

메인 페이지

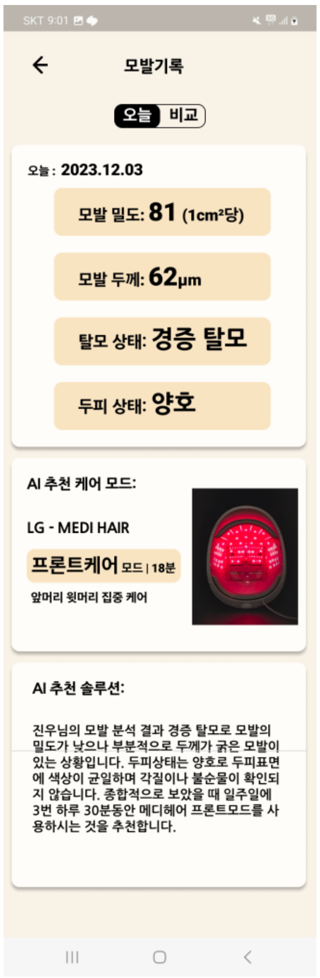
Once the analysis is completed, the user will be redirected to a page where they can review the analysis results. On this page, they can check the 'Hair Loss Condition,' 'Hair Density,' 'Hair Thickness,' and 'Scalp Condition.' Additionally, fine-tuning will be performed, where the trained ChatGPT analyzes the user's analysis information and provides recommended care and treatment solutions. Displays the time taken for analysis on the screen. These recommendations will suggest utilizing MediHair products for the recommended care and solutions.

I. Main Page



The main page's top section displays the user's photo and offers four key functions: 'Hair Analysis,' 'Analysis Records,' 'Recommended Products,' and 'Chatbot,' which users can navigate to. At the bottom, there are banners that can be scrolled to access well-known hair loss community websites. Clicking on these banners will take users to the respective sites. Finally, at the very bottom, there are buttons for 'Settings,' 'Home,' 'Notifications,' and 'My Page' for navigation.

J. Analysis Records Page – 1


SKT 9:01

모발기록

오늘 비교

오늘 : 2023.12.03

모발 밀도: 81 (1cm²당)

모발 두께: 62µm

탈모 상태: 경증 탈모

두피 상태: 양호

AI 추천 케어 모드:

LG - MEDI HAIR

프론트케어 모드 | 18분

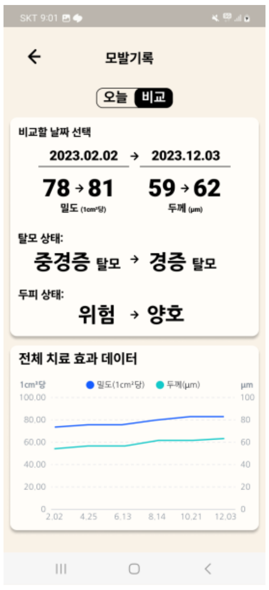
앞머리 윗머리 집중 케어

AI 추천 솔루션:

진우님의 모발 분석 결과 경증 탈모로 모발의 밀도가 낮으나 부분적으로 두께가 굵은 모발이 있는 상황입니다. 두피상태는 양호로 두피표면에 색상이 균일하며 각질이나 불순물이 확인되지 않습니다. 종합적으로 보았을 때 일주일에 3번 하루 30분동안 메디헤어 프론트모드를 사용하시는 것을 추천합니다.

On this page, User can check the most recent analysis results. And user can check 'hair loss condition,' 'hair density,' 'hair thickness,' and 'scalp condition.' Additionally, through fine-tuning, the trained ChatGPT analyzes user's analysis information and provides recommended care and treatment solutions. The recommended care and solutions will recommend a method using Medihair. Press the 'Compare' button to move to the hair analysis page 2.

K. Analysis Records Page – 2


SKT 9:01

모발기록

오늘 비교

비교할 날짜 선택

2023.02.02 → 2023.12.03

78 → 81 밀도 (1cm²당)

59 → 62 두께 (μm)

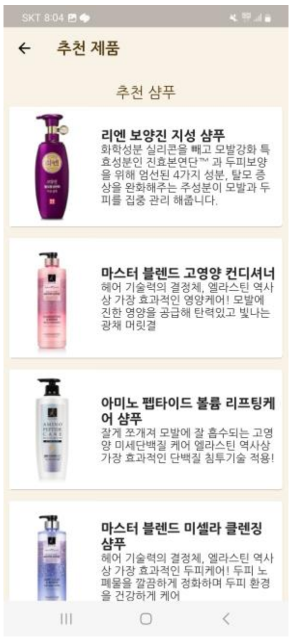
탈모 상태:
중경증 탈모 → 경증 탈모

두피 상태:
위험 → 양호

전체 치료 효과 데이터

On this page, users can select and compare the records of their analyses conducted using the app. User can choose a total of 2 dates, and it is activated to allow selection only of the dates when the user conducted the analysis. Once you select a date, it retrieves the hair density and thickness records from the database for that date and displays them on the screen. Additionally, at the bottom of the screen, it presents a graph of hair density and thickness for the analyses conducted by the user over the past 3 years. This allows users to check the progress of their hair loss condition.

L. Recommended Product Page


SKT 8:04

추천 제품

추천 샴푸

리엔 보양진 지성 샴푸
화학성분 실리콘을 빼고 모발강화 특효성분인 진효본연단™ 과 두피보양을 위해 엄선된 4가지 성분, 탈모 증상을 완화해주는 주성분이 모발과 두피를 집중 관리 해줍니다.

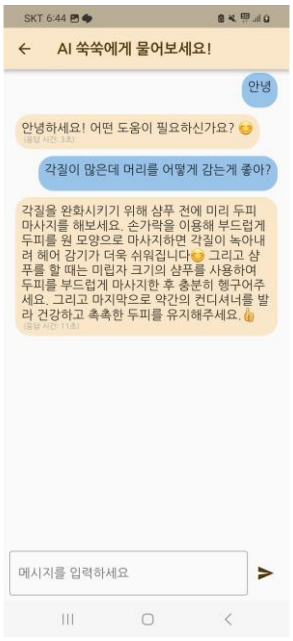
마스터 블렌드 고영양 컨디셔너
헤어 기술력의 결정체, 엘라스틴 역사상 가장 효과적인 영양케어! 모발에 진한 영양을 공급해 탄력있고 빛나는 광채 머릿결

아미노 펩타이드 볼륨 리프팅케어 샴푸
잘게 쪼개져 모발에 잘 흡수되는 고영양 미세단백질 케어 엘라스틴 역사상 가장 효과적인 단백질 침투기술 적용!

마스터 블렌드 미셀라 클렌징 샴푸
헤어 기술력의 결정체, 엘라스틴 역사상 가장 효과적인 두피케어! 두피 노폐물을 깔끔하게 정화하며 두피 환경을 건강하게 케어

This page recommends products to user. ChatGPT trained through fine-tuning based on the user's analysis information, recommends products. Out of a total of 40 products, ChatGPT recommends 5 shampoos and 5 treatments based on the knowledge it has learned. These recommendations are displayed on the screen. Additionally, by clicking on a product, you can navigate to a page where you can make a purchase.

M. ChatBot Page


SKT 6:44

AI 쑥쑥에게 물어보세요!

안녕

안녕하세요! 어떤 도움이 필요하신가요? 😊
(응답 시간: 3초)

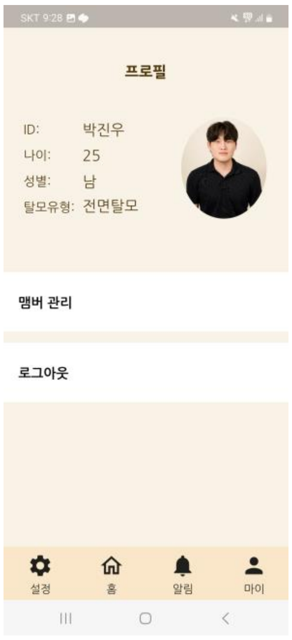
각질이 많은데 머리를 어떻게 감는게 좋아?

각질을 완화시키기 위해 샴푸 전에 미리 두피 마사지를 해보세요. 손가락을 이용해 부드럽게 두피를 원 모양으로 마사지하면 각질이 녹아내려 헤어 감기가 더욱 쉬워집니다😊 그리고 샴푸를 할 때는 미립자 크기의 샴푸를 사용하여 두피를 부드럽게 마사지한 후 충분히 헹구어주세요. 그리고 마지막으로 약간의 컨디셔너를 발라 건강하고 촉촉한 두피를 유지해주세요.👍
(응답 시간: 11초)

메시지를 입력하세요

ChatGPT, trained through fine-tuning, helps with the user's hair loss treatment consultation. Displays the time taken for analysis on the screen. Below is an example conversation.

N. Profile page


SKT 9:28

프로필

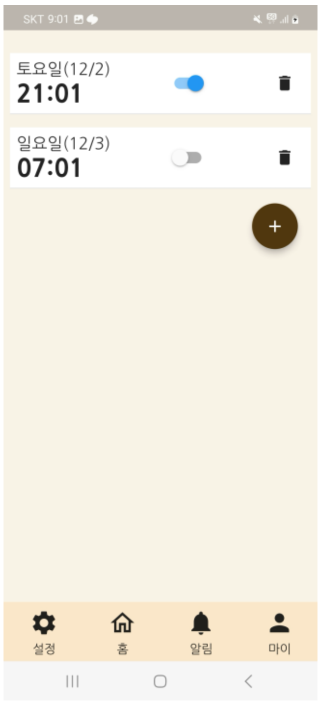
ID:	박진우
나이:	25
성별:	남
탈모유형:	전면탈모

맴버 관리

로그아웃

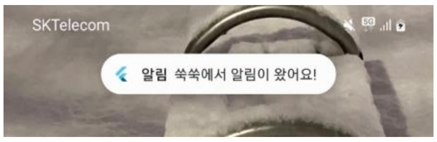
On the profile page, you can view the user's photo and information. It displays the user's ID (nickname), age, gender, and hair loss type. You can also log out of the app on this page.

O. *Alarm Page*



On this page, users can set up alarms. It provides the functionality to receive treatment reminders at specific times based on the solutions provided by the app. By clicking the '+' button, you can select the day and time. After selecting, it offers the option to turn on/off for the chosen day and time.

Example of the notification screen.



REFERENCES

[1] Kingsley, D.H. The development and validation of a quality of life measure for the impact of androgen-dependent alopecia. PhD Thesis, University of Portsmouth, Portsmouth (1999).

[2] Medi-Consumer News. The age of hair loss is getting lower... http://www.medisobizanews.com

[3] Sanghee Park, trends in cosmetic patents related to hair loss prevention and hair growth, report Gun Industry Technology Trend, p.24-32, 2003

[4] ATOMAM Korean. (n.d.). from http://atomom.co.kr/m/index.jsp

[5] LG ThinQ. (n.d.). from https://www.lge.co.kr/lg-thinq

[6] InOut Team. (n.d.). from https://www.inout.team

[7] Google Calendar. from https://calendar.google.com/calendar/

[8] Google Analytics.from https://marketingplatform.google.com/about/analytics/

[9] Flutter. (n.d.). from https://flutter.dev

[10] MySQL. (n.d.). from https://www.mysql.com

[11] Node.js. (n.d.). from https://nodejs.org/en

[12] GitHub. (n.d.). from https://github.com/

[13] Amazon EC2. (n.d.). from https://aws.amazon.com/pm/ec2/?nc1=h_ls

[14] Visual Studio Code. (n.d.). from https://code.visualstudio.com/

[15] Amazon RDS Free Tier. (n.d.). from https://aws.amazon.com/free/database/

[16] Figma. (n.d.). from https://www.figma.com/

[17] scikit-learn. (n.d.). from https://scikit-learn.org/stable/

[18] NumPy. (n.d.). form https://numpy.org

[19] TensorFlow. (n.d.). EfficientNetB0 - TensorFlow. from https://www.tensorflow.org/api_docs/python/tf/keras/applications/efficientnet/EfficientNetB0

Contents.