Overview:

The project involved developing a skincare consulting platform with video and audio functionalities for a highly respected and well-established dermatologist. The client used the application to consult with patients for their skincare-related issues. This project enabled them to seamlessly improve the process and manage the day's schedules and events accordingly.

Client details:

Name: Confidential | **Industry:** Healthcare | **Location:** US

Technologies:

Python / Django REST framework, ReactJS, Twilio Video API, Twilio Authy, Braintree Payment Gateway

Project Description:

The client was a startup run by a renowned dermatologist. As the client's user base grew, it became difficult to manage the process manually for various users. After analyzing the client's need, team@Mindfire offered to develop an application using Django REST Framework & ReactJS technology. This project enabled the client to interact with their customers, manage notes and appointments, give consultation via audio/video, etc. It was critical to have an intuitive and user-friendly UI/UX and eliminate lag during the time of consultation. Some salient features of the application:

- The user can sign up by sharing a few basic details and login into the application. After login, the user can select the mode for consulting services, i.e. chat, audio or video. During the consultation, the dermatologist can recommend products that could be recurring or one-time purchases.

- The estheticians can connect with their patients via chat/audio or video functionalities. This was achieved by using Twilio's APIs.

- In the recurring order tab, the admin can charge customers each month via credit card. The team ensured this by integrating a 3rd party application, Braintree payment gateway into the application.

- The customer can place and manage orders, update payment information, change addresses. Additionally, they can also explore products on the home page based on their preferences.

- As the client wanted to avoid cost overruns; the team implemented this via a serverless stack from Amazon Web Services (AWS). It allowed the client to improve finances by not paying for idle server time.

- The primary intention of the client was an application that could be scaled up easily. AWS Lambda and AWS API Gateway were used to deploy various micro-services of the application.

Architecture:
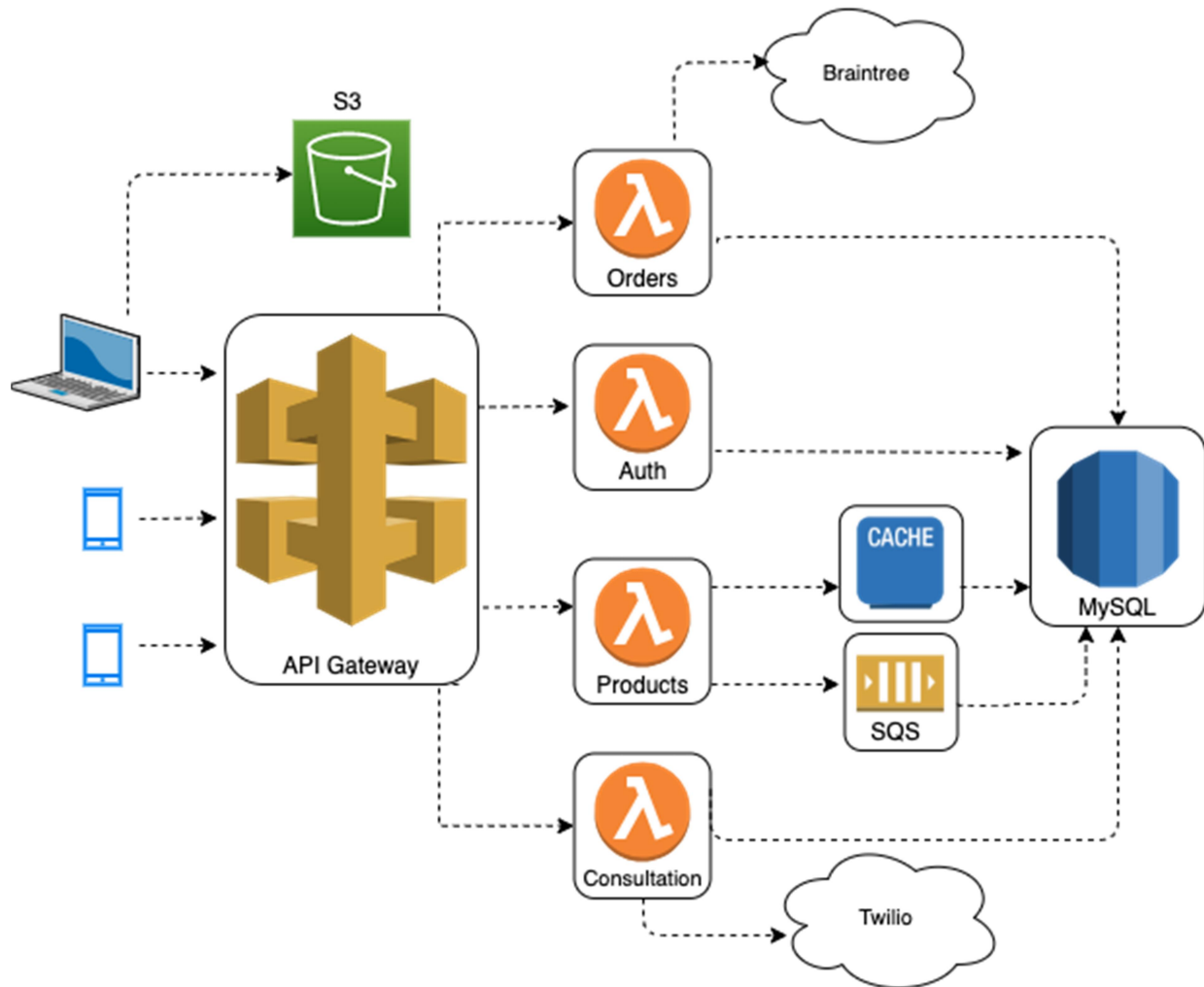
Screenshots:
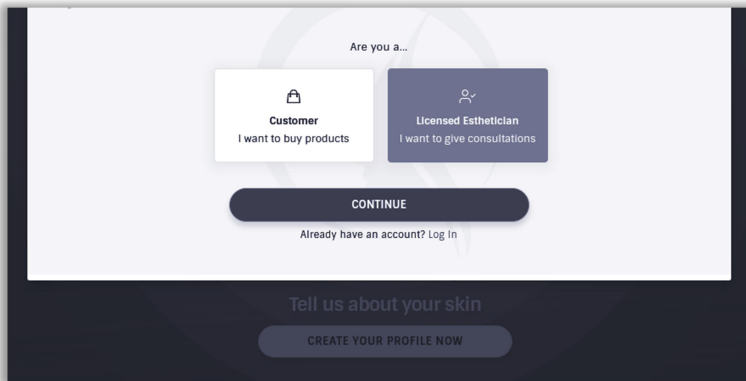
Screenshot 1: Sign Up
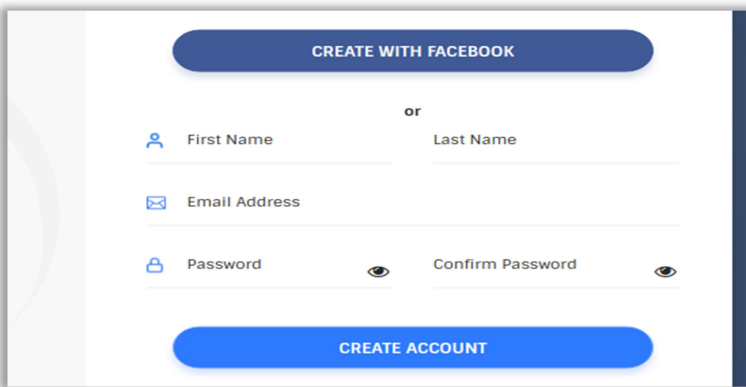
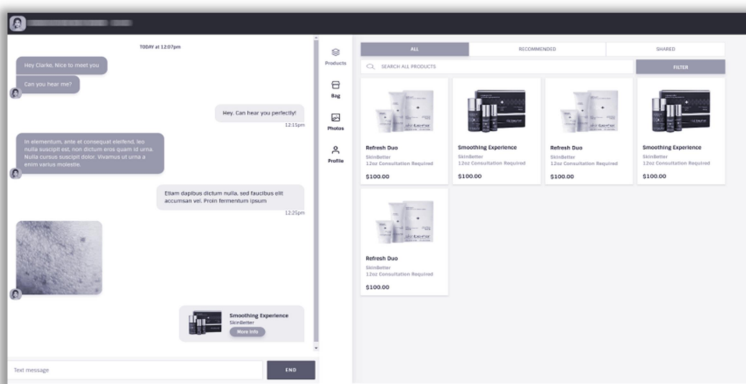**Screenshot 2:** Create Account
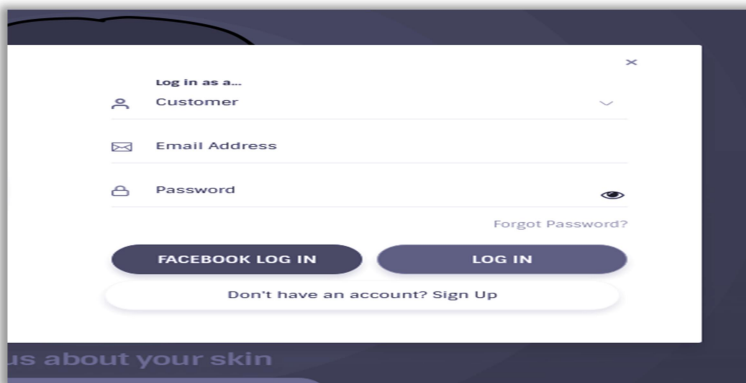
Screenshot 3: Login View
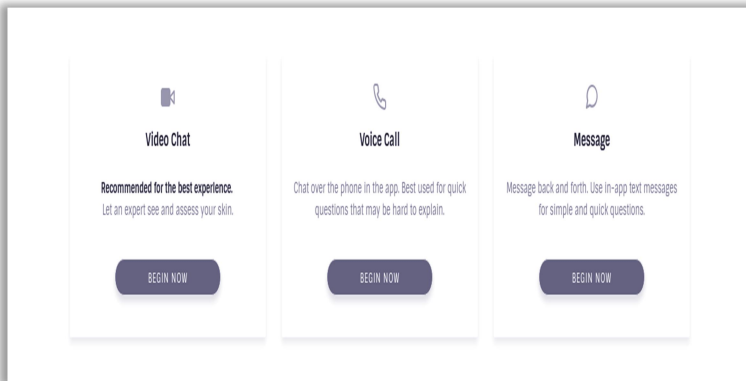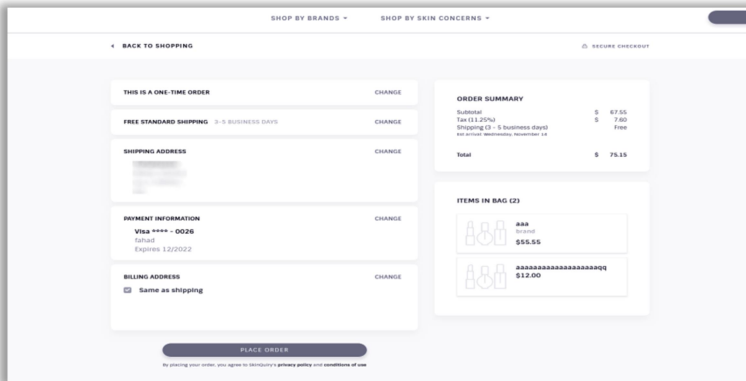
Screenshot 4: Select Consultation
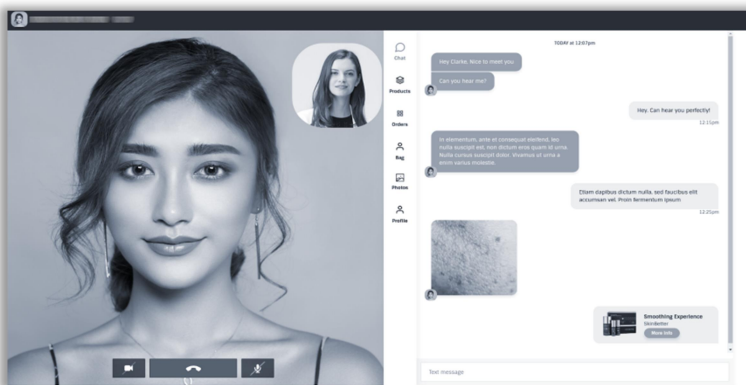
Screenshot 5: Video Consultation

Screenshot 6: Chat Consultation
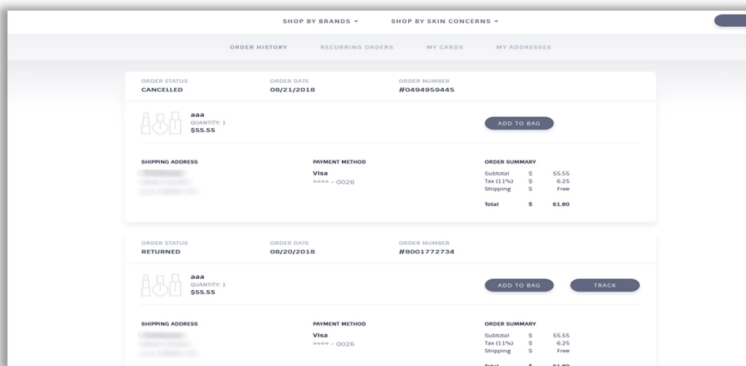
Screenshot 7: Place Order

Screenshot 8: Order Management

Screenshot 9: Manage Cards
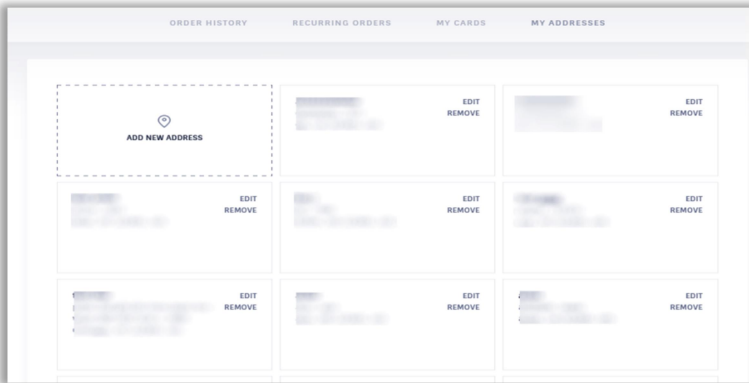
Screenshot 10: Manage Address
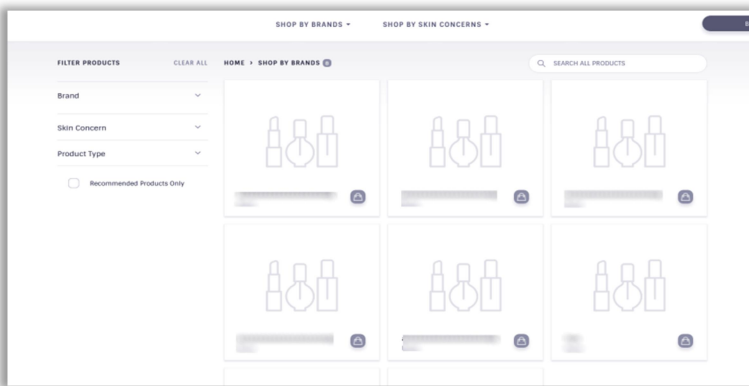
Screenshot 11: Explore Products