Simplify: Beta Release

GitHub Repository link:

https://github.com/HeinHtutZaw19/simplify

Website link:

https://simplify-e3px.onrender.com/welcome

Contact us at hein.zaw@stonybrook.edu to restart the server - valid for 15 minutes of no-user period

Individual Progress Update

Hein Zaw (115344093)

1) Scheduled

Milestone 1
- Web Scraping of the products + articles
    - Adding documents on supabase

Milestone 2
- Chatbot API Implementation via Langchain
    - Langchain API test cases
    - Connection with frontend + backend
    - Deleting chat history

Milestone 3
- Vision API Implementation via OpenAI
    - Adding the photo via supabase backend
    - Analyzing the public url via OpenAI Vision
    - Connecting with frontend+backend
- Product Recommendation API Implementation via OpenAI
    - Making AI API implementation to recommend the product based on survey results

Milestone 4
- Deployment
    - Production Mode Implementation
    - Deploying the server on Render

2) Completed

Milestone 1

- Successfully implemented web scraping scripts to collect product details and related articles from various sources.
- Structured and uploaded the scraped product data and articles—over 1,000 documents in total—to Supabase, storing them in PostgreSQL format for efficient querying and organization.
- Additionally, predefined prompt-answer pairs were transformed into vector embeddings and stored to support semantic search and intelligent responses from the chatbot.

Milestone 2

- Integrated LangChain's API to enable dynamic conversational interaction with the ingested product and article data.
- Created and ran test cases to validate the performance and accuracy of the chatbot responses using LangChain.
- Connected the chatbot seamlessly with both frontend and backend components to allow real-time communication.
- Implemented functionality to delete previous chat histories, ensuring privacy and reducing storage bloat for inactive sessions.

Milestone 3

- Integrated OpenAI's Vision API to enable image-based analysis. Users can upload product photos via the Supabase backend.
- Analyzed publicly accessible URLs of uploaded images using the Vision API, extracting relevant insights for the chatbot.
- Connected the Vision API's analysis results to the overall app flow, including frontend display and backend processing.
- Developed a product recommendation system using OpenAI API, generating personalized product suggestions based on user survey inputs and inferred preferences.

Milestone 4

- Finalized the application in Production Mode, ensuring stable and optimized performance.
- Successfully deployed the backend + frontend server and APIs to Render, making the application accessible for live usage and testing.

Every milestone task is completed, but deployment on AWS is still required for persistent and scalable long-term hosting.

Minji Kim (115242397)

Scheduled:

Milestone 1

- Frontend Pages (Completed)
- Leaderboard Scoring Logic (Completed)

Milestone 2

- Admin Page (Completed)
- Skin-related API prompt design (Other member Completed)
- Frontend State Management (Completed)
- Integration with Backend (Completed)

Milestone 3

- Dall-E vision API for Skin Lab (Partially Contributed)
- Leaderboard System (Completed)

Milestone 4

- Deployment (Other Member Completed)
- Testing Bug Fixing (In Progress)

Completed:

Milestone 1

- Frontend Pages - Frontend revision, fixing prominent bugs of previously made pages. Some simple bugs took a significantly long time, but this helped in understanding bugs further into the project.
- Leaderboard Scoring Logic - Made concrete streak and point definitions and their logic system. How they would be stored and how they would be updated was decided.

Milestone 2

- Color Design and Theme - Dived into the specifications of industry standards of UX and UI. Created color schemes using the color of our Simplify logo. Added specific text colors that have at least 4.5:1 contrast with the background and other specifications for the css of components.
- Profile Page - Frontend. Was able to make note of what data sets from users the profile page needed later for backend integration.
- Admin Page - Frontend. Was able to do the same as the profile page.
- Webcam / File upload - Component. This component was placed for survey and Skin Labs.

Milestone 3

- Frontend State Management - From the bug reports we have done this week, identified bugs were able to be fixed.
- Worked on the details of the vision API implementation on the backend.
- Data Design Revision - We discussed and decided to change the data design to optimize the current use of classes and fit the requirements of streaks, points, and leaderboard without needing too many functions added. New attributes: point, days, routineDate added to the user class. Forum and Leaderboard classes deleted.
- Redesign of the Leaderboard, Streaks, Points related API as the data design changed.

Milestone 4

- Leaderboard Page - Implementation of backend and frontend APIs. Redesign of the page which was a change to the Requirements.
- Streaks/Points - Backend operations for all end cases. Backend and frontend API implementation.
- Admin Page - Implementation of backend and frontend APIs. Additional frontend design and functionality added.

Dowon Kim (114001605)

Scheduled:

- Milestone 1
  - API design
  - API early implementation
  - Authentication error handling
  - Frontend responsiveness
  - Google Cloud Console setup
- Milestone 2
  - Chatbot history
  - Google OAuth login
- Milestone 3
  - Cloudinary image hosting API connection
  - Home page user data integration (product list, survey summary)
- Milestone 4
  - User feedback
  - Alpha/Beta testing
  - AI prompt improvements (In-progress)
  - Bug fixes for assigned tasks (In-progress)

Completed Tasks (Milestones 1 to 4):

- Frontend
    - Signup page
        - Implementation
        - Error handling
    - Login page
        - Implementation
        - Google OAuth (completed, but buggy)
        - Error handling
    - Home page
        - Product list data
        - Skin analysis data
            - User's survey image
            - AI generated feedback
    - Profile page
        - Change profile picture
    - API middleware implementation
        - get user's routine
        - get user's skin analysis summary
        - get user's chat list
        - get AI recommendation from survey data
        - upload image
        - update user's profile picture
    - Minor optimizations & quality of life changes
        - Only load components after user data received from backend
        - Auto focus on input fields
- Backend
    - API design & implementation
        - google login/callback
        - get user's chat
        - get user's routine
        - get user's skin analysis summary
        - upload to Cloudinary
        - get AI recommendation
        - update user's profile picture
    - Google Cloud Console setup
    - Cloudinary setup

In-progress Tasks:

1. AI prompt improvement

The AI routine/product recommendation in the survey and skinlab processes can be significantly improved. I have been experimenting with lots of different styles of prompts for the AI, but I still need to make sure that the responses are properly structured and have no hallucinations. More specifically, in the survey section, I observed that it sometimes responds with only 3 products, even when I explicitly ask to generate 4. The prompt modification is at its final stages (around 80% done), but I will need a few more days to try some new ideas out.

2. Bug fixes for assigned tasks

There were some minor bugs for the tasks that were assigned to me. For example, the Google OAuth login can sometimes redirect to the welcome page after a successful login, requiring the user to refresh the browser to use the app. Another minor bug is with signup's error handling process, where it doesn't reset the page when an error occurs and the user gets stuck. I have been fixing a lot of my bugs constantly throughout our development process, so I would say that this process is also close to being finished at around 90%.

Group Progress Update

General Progress

As of the beta release, our development is on track with our initial schedule written in the software design document. We have finished all of our main use cases described in the Software Requirements and Software Design documents, and only have minor adjustments and fixes left to release our web application. The last week leading up to the final release will be focusing on the topics discussed during our beta release presentation, which includes other deployment service considerations, AI prompt improvements, and lots of bug fixes.

Self-Grading

We have decided to give ourselves a grade of **A** at this point of our project. This is because overall, our development has been smooth and consistent throughout the semester, and managed to finish the important functionalities in time for the beta release with no major schedule issues. We've had no problems with communication, had no disputes with each other, and every member of the team contributed meaningfully at each of the four milestones.

Schedule Issues and Adjustments

The biggest issue that our team is currently facing is **deployment**. We initially decided on Render's free service for its simplicity and ease of use, but quickly realized that it has too many drawbacks. One of these downsides is the pausing of the servers after *15 minutes of inactivity*. This was too much for us, and even though the deployment on Render works well right now, we need to quickly find and move our code base before the final release date. Other than deployment, we have some small issues regarding minor bugs not being fixed and additional quality-of-life features. We believe these problems to be solved by the weekend before our release date, so we are not worried about it affecting our schedule.