AI Marketer Development Journal

This journal documents the week-by-week progress of the AI Marketer platform development during my industry placement at AKA Studio from March to May 2025.

Note about documentation references: Throughout this journal, there are references to various planning documents, technical specifications, and guides. All these documents can be found in the `/docs` folder of the project repository. The actual filenames in the repository may differ from those mentioned in this journal, but the content remains the same.

Table of Contents

Week 1

- Discuss with the team whether the framework used for the current prototype is suitable for us @Monday
  - Conclusion: Keep the current setup (Next.js for Frontend, Django for Backend)
- Share the distributed link to the prototype with the team so they can check it out and get a better understanding of the overall concept before working on the actual code. @Monday
- Create a Features Document and share it with the team for the mind map @Tuesday
  - [AI Marketer v2] Features
- Clean up the existing prototype code, create a new Git repository for the backend, and share it with guidelines @Tuesday
  - https://github.com/heeran-kim/ai-marketer-v2-backend.git
- Clean up the existing prototype code, create a new Git repository for the frontend, and share it with guidelines @Wednesday
  - https://github.com/heeran-kim/ai-marketer-v2-frontend.git

- Create a draft of the system architecture and share it with the team @Wednesday
  - System Architecture & Requirements
- Review the mind map Daniel and Drishti made and provide feedback @Friday
- Review the system architecture document revised by Daniel and provide feedback @Friday

Week 2

GitHub Setup & Weekly Release Plan

- Had a meeting regarding GitHub setup. @Wednesday
- Created Aka Studio's GitHub repository and uploaded the existing code as the initial setup.
- Sent a follow-up email with step-by-step instructions on how to download the repository without requiring installations. @Wednesday

Updated Features Document (v2)

- Created Features v2 document as an updated version of v1. @Friday
  - [AI Marketer] Features v2 (16.Mar)
  - Added Potential Assignee section, suggesting a division of tasks into three main areas so that each of the three team members can take ownership of one area, ensuring better workload distribution based on task relevance.
  - Prioritized features within each category so lower-priority tasks can be skipped or adjusted as needed.
  - Documented API Endpoints for backend services that the frontend will call.

Frontend & Backend Integration

- Connecting frontend to backend, replacing previously used frontend mock data with actual backend API calls @Monday~Sunday
- Uploaded the updated code to AKA Studio's GitHub repository @Sunday

Week 3

[Feature 1] User Authentication

1.1. Authentication Guide Documentation

I created an Authentication Implementation Guide outlining the authentication strategy.

It includes our requirements, chosen authentication method, the reasoning behind it, common issues and solutions, and future refactoring suggestions.

Link: [Feature][1] Authentication Implementation Guide.pdf

1.2. Fix for UI Flickering

Previously, the app had flickering issues during authentication state updates.
This issue was described in items #4 and #5 of the guide. I modified the authentication state logic to prevent unnecessary re-renders. Fix implemented in this commit: [9439756]

1.3. Authentication Code Refactoring

I refactored the authentication logic for better extensibility (as described in #6 of the guide). The updated structure supports adding new features beyond traditional login/register. I made sure the code structure supports easy addition of new features. I also created the necessary components for those features with clear explanations of what each component's role should be to help team implement additional features easily. Implemented in this commit: [2e7f666]

1.4. Unified Authentication Flow & UI Improvements

I simplified the user authentication experience by combining login and registration into a single flow.
The UI now includes clearer error handling and a unified `Get Started` entry point. Social login and passkey support have also been prepared for future use. Implemented in this commit: [080b569][efec813] => Feature 1.a. User Authentication - Register, Login, Logout (Traditional) is now fully complete. A demonstration video link is attached below: [AI Marketer v2][Feature 1a] Traditional Auth.mov

2. [Feature 2.a.] Business Management - General Settings

2.1. Save Functionality

I implemented per-field saving on the General Settings page. I created and connected the necessary backend API endpoints to handle business information updates. Implemented in this commit: [9728b38]

2.2. Add User Feedback Mechanisms

Enhanced the user interface with comprehensive feedback mechanisms including field-specific loading states, success notifications after successful saves, and validation error messages. Implemented in this commit: [5e75c47]

2.3. Handle Logo Upload Functionality

Implemented automatic logo upload that saves immediately upon file selection. Implemented optimistic UI updates with temporary URLs for immediate feedback.
Added proper error handling to restore the previous logo if upload fails, and included file validation for size and type. Implemented in this commit: [d58a453]

2.4. Improve Settings Page for Responsive Design

Enhanced the settings page with proper responsive design to provide optimal experiences on both mobile and desktop devices. Implemented in this commit: [3ee8efd] => This feature is now fully complete. A demonstration video link is attached below: [AI Marketer v2][Feature 2a] General Settings.mov

3. Team Leading

As team lead, I supported both Daniel and Drishti this week.

- I reviewed Daniel's pull request and provided feedback on his 2FA implementation.
- I discussed with Drishti how her tasks align with the overall project and what she plans to work on this week.

Suggestions I made during the discussion:

- I recommended prioritising other features that better align with the current direction of the project, instead of proceeding with Feature 7.
- I proposed adding Feature 1.f (Account Settings), which would allow users to manage their connected accounts or delete their profiles. I also suggested expanding social login support beyond Google, such as including Apple or Facebook.
- I encouraged reviewing the current code and creating use cases first, to clarify how the feature should be implemented. I also reviewed her use case drafts and gave feedback to help align them with our implementation plan.

Week 4

Feature 2.c Businesses Management - Sales Data

Plan

In a real-world setup, we would fetch sales data using Square's API. However, since we don't have access to actual POS hardware or merchant credentials at this stage, I planned to use the built-in export feature provided by most POS platforms (such as Square) to obtain sales data in CSV format.

This approach allows us to simulate a realistic sales data workflow while staying within current development constraints.

Demo Video Link

[AI Marketer v2][Feature 2c] Sales Data.mp4

Feature 2.d Businesses Management - Display Dashboard

1. Demo Video Link [AI Marketer v2][Feature 2d] Display Dashboard.mp4

Feature 3.e Post Management - Edit/Delete Scheduled Post

Plan Document

[Feature][3e] Post Editing Feature Implementation Plan.pdf

Demo Video Link

[AI Marketer v2][Feature 3e] EditDelete Scheduled Post.mp4

Team Leading

1) Suggestion to Improve Git Workflow Using Squash and Merge

I noticed that our commit history included many small and repetitive messages, which made it harder to follow the project's progress. We were using the default merge strategy that keeps all individual commits.

To improve clarity, I suggested we use "Squash and Merge", which combines all commits from a pull request into a single, clean message. This helps summarize the work more clearly and keeps the history easier to review.

Updated Git Workflow Proposal:

1. Check the size and type of the task. If it's more than a small bug fix, a new branch is created.
2. Work is done in a separate feature branch.
3. Submit a Pull Request. All commits can be reviewed and discussed here.
4. After code review is complete, use Squash and Merge. This combines all commits into a single message before merging into dev or main.

→ This avoids cluttering the commit history with too many small commits.

2) Feedback and Coordination During Frontend Collaboration

This week, Drishti shared progress on a frontend component that was based on an outdated version of the code. I had already submitted a pull request with updates to the Login Page UI, which were not reflected in her work.

Despite the version mismatch, I appreciated Drishti's proactive effort in trying to enhance the UI and her initiative in asking for feedback before pushing any changes. She also tested the updates locally first, showing careful consideration for quality.

I provided feedback to:

- Prevent code conflicts by encouraging her to check the latest version before continuing.
- Maintain design consistency by suggesting the use of the shared Card component used across settings pages.
- Reinforce the importance of following our updated Git workflow and reviewing merged PRs to avoid duplicate or inconsistent work.

This helped align our work and ensured that all team members were on the same page regarding both design and version control.

3) Testing and Validating Team Features

I reviewed the 2FA (two-factor authentication) feature implemented by Daniel before merging. I also reminded him to run `npm run build` to ensure the project builds successfully without errors — an important step to catch issues that may not appear during development.

Week 5

Follow-up on Last Week's Feedback

I logged in with the prototype account and confirmed that I could access the Square Developer tool without any issues. Although I haven't explored it in depth yet, this step helped ensure that the integration works as expected and brought the project one step closer to a production-ready state. I plan to work with the tool more actively once the sales data is needed for AI-based analysis tasks.

Feature 5. Promotion Management

Demo Video Link

Update Summary

Related Post Features

- Auto-Scroll to Post: When accessed via promotion or external link, page auto-scrolls to the selected post.
- Post Creation: Fully integrated with backend. Users can:
  - Add image, caption, platform, schedule, and promotion.
  - Retry caption generation on error (with a "Retry" button).
  - Use “Don’t Post” option per platform in Review step.
  - Get success/failure messages without losing input on failure.
  - Return to the originating page (Post/Promotion Dashboard) after success.
- Promotion Management:
  - Tabbed UI: My Promotions and Suggestions.
  - My Promotions
    - Promotion Cards show:
      - Date range, status (computed from current date), categories, description.
      - Connected post info: image, status, platform icon, caption, reactions.
      - Sold count (placeholder for future sales data integration).
      - Action buttons: Create Post, Edit, Duplicate, Delete.
      - Filtering Bar: Search by description, category, or status.
      - Promotion Actions:
        - Delete: Confirmation required; deletes promotion and linked posts.
        - Edit: Update promotion date range.
        - Duplicate: Copy promotion with adjustable date range.
        - Create Post: Opens post editor with promotion details pre-filled.
  - Suggestions (Mock Data)
    - Create Promotion from Suggestion:
      - Adjust date range.
      - Auto-scroll to new promotion in `My Promotions`
- Promotion status: Instead of storing a fixed status, the promotion status (upcoming, ongoing, ended) is now computed based on the current date, ensuring the information is always accurate.

Feature 4.a AI-Powered Post Creation – Image Analysis

I’ve completed the research and analysis for the image analysis approach in the AI-powered post creation feature. After evaluating various options, I found that paid commercial APIs like AWS Rekognition and Google Vision, along with BLIP (Replicate), offer the best balance of quality and implementation effort for food detection. The options and next steps are outlined in the document linked below. [Feature][4a] Image Analysis Plan.pdf

Team Leading

To support smoother collaboration during development, I created a debugging guide outlining how to check API connectivity between the frontend and backend using Docker, browser tools, and log analysis. [General] API Debugging Guide.pdf

Additionally, I recommended some troubleshooting steps to address the CORS error that was preventing frontend-backend communication. These steps included:

- Trying a different browser (since CORS is a browser security setting)
- Disabling CORS settings directly in the browser
- Re-cloning the repository to reset local settings

Unfortunately, none of these solutions worked.

Now, as a potential solution, I am considering running the code on a deployed server (such as Vercel or Render) instead of the local server. If the CORS issue is indeed the root cause, this should resolve it. However, if it's not a CORS issue, we'll need to explore other potential solutions.

Week 6

Feature 4. AI-Powered Post Creation

[Feature][4] Performance and Cost Analysis Report.pdf

I completed a detailed performance and cost analysis of various input methods.

Test Focus: Evaluating cost-efficiency and content quality for product promotion, time-limited discount promotion, and product bundle promotion.

Key Highlights:

- Cost Efficiency: GPT-4o-mini outperforms GPT-4o with 90% cost savings, showing minimal performance difference.
- Input Methods:
  - Additional Prompts & Item Descriptions: Best cost-to-performance ratio.
  - Image Input: Show limited improvements at a higher cost, but are effortless for users and have room for performance enhancement.
  - Object Detection: Not feasible due to excessive user effort.
- Based on test findings, I suggest splitting users into two groups:
  - Basic Users: Users who prioritise lower costs and are willing to put in some effort.
    - Using additional prompts and item descriptions methods
    - Approx. $0.60 per 1000 caption sets
    - Demo Video: [AI Marketer v2][Feature 4] AI-Powered Post Creation - Basic.mp4
  - Premium Users: Users who prioritise a no-effort experience, even at a higher cost.
    - Using additional prompts and image input methods
    - Approx. $1.20 per 1000 caption sets, excluding training (TBD).

Next Steps:

- Implement Square API integration for basic users
- Develop a plan to enhance the image input method using the Fine-tuning API for premium users
  - Challenge: Requires training data
    - How much data is needed for acceptable quality?
    - Where can I collect images for training? Collect public images from social media or Google Maps reviews?
    - Are there any ethical or legal concerns with using those images without explicit consent?
    - ...
  - Given these uncertainties, it may be more efficient to just maintain the current caption generation feature for Basic users (which already works with some user input), and prioritise development of the promotion suggestion feature using sales data.

Team Leading

Follow-up on CORS issue on Drishti's environment

< Last week >

We encountered CORS errors when Drishti attempted to run both the frontend and backend locally. To troubleshoot, I suggested the following steps:

1. Disabling CORS settings in the browser → ❌ No effect
2. Trying a different browser → ❌ No effect
3. Re-cloning the repository → ❌ No effect

Since none of these worked, I proposed an alternative: running either the frontend or the backend on a deployed server (e.g., Vercel or Render) to potentially bypass local CORS issues caused by misconfiguration.

< Work on Monday >

1. Deployment Guide and Setup To formalize this approach, I created a guide to help Drishti set up development using a combination of local and deployed environments. [General] Deployment Guide.pdf

As part of this setup, I deployed the frontend to Vercel and the backend to Render to create a stable environment for testing.

2. Deployment Testing Daniel and I tested the deployed setup on both of our local environments and confirmed that the system was functioning as expected.

3. Video Call with Drishti: Deployment Testing

I conducted a video call with Drishti to test the setup on her machine. We tested various combinations of frontend and backend environments, starting with the ones most likely to succeed:

- ☑ Frontend on Vercel & Backend on Render → Working
- ☑ Frontend running locally & Backend on Render → Working
- ❌ Frontend on Vercel & Backend running locally → Not tested (Docker issues)
- ❌ Frontend and Backend both running locally → Not tested (Docker issues)

4. Docker Issue on Drishti’s Machine

We couldn't proceed with local backend testing because Docker was not functioning properly on Drishti’s computer. This is likely due to the high resource usage of Docker, which her system couldn’t handle efficiently.

Note: Docker ensures consistent environments across developers by containerizing the backend, but can be too resource-heavy for lower-spec machines.

5. Proposed Alternatives To move forward, I suggested two options:

- (a) Install Django and dependencies locally without Docker
  - ☑ Pros: Avoids the performance issues caused by Docker
  - ⚠ Cons: Requires manual setup, and could lead to inconsistent environments across the team; developers must be cautious when syncing environments
- (b) Continue using the backend on Render
  - ☑ Pros: Already tested and working
  - ⚠ Cons: Any code changes must be pushed to GitHub and manually redeployed on Render to take effect

< Work on Thursday >

Docker Performance Issues on 8GB RAM

After reviewing the team's hardware specifications, it was found that Docker does not perform smoothly on systems with 8GB of RAM.

[AI Marketer V2] Team HW Spec.xlsx

According to a Reddit thread, the issue appears to be related to memory:

"Your problem is memory. Docker on macs runs in a QEMU VM and uses per default 2GB for that with chrome already maybe eating 4GB (you know chrome is quite hungry especially with each ne tab) you are already at 6GB used and when you start some x86 container that basically mean another qemu VM is starting which emulates x86 hardware. Even more memory is eaten up then. 8GB isn’t enough. I have a 16 GB and you can work with it and I would recommend that as bare minimum. Better to have a MacBook Pro with 32GB."

I think the reason Docker wasn't working when I was accessing Drishti's computer remotely via a Teams video call is that video calls and remote access consume significant system resources, which added additional load on the system.

In conclusion, Docker development is possible with 8GB of RAM, but when running a web browser or additional servers simultaneously, the system may struggle. If Drishti runs Docker for development without remote access, it might be possible with 8GB of RAM.

< Work on Friday >

Current Challenges

- Drishti is struggling to set up and debug the existing code, as it was not written by her.
- Due to limitations on her computer (inability to run Docker and Teams simultaneously), it is difficult for me to assist her remotely.

Proposal

- I suggest that Drishti start her assigned part as a new, separate project from scratch

Considerations

- Integration feasibility: Since Drishti is responsible for the authentication part, integration should be relatively straightforward
- Ease of development and debugging: Since she will be working with either the existing code that she understands or new code she writes herself, debugging is expected to be more manageable
- Integration support: Once the authentication feature is confirmed to be working, I will support the integration into the main project
- Potential redundancy in work: Although she will need to rebuild parts of the existing code in her own project, this process will help her gain a deeper understanding and ownership of the code, ultimately benefiting the overall project.

Week 7

Gantt Chart Update

- Week 4: The caption generation feature, originally scheduled from Week 4 to Week 8, was split into two parts: one for basic users (Feature 4.b) and one for premium users (Feature 4.c). The development for basic users was completed during Week 4. The premium version, which involves enhancement using fine-tuning APIs, will be handled by Daniel.
- Week 5: I implemented newly added settings features related to Square integration (features 2.c, 2.f, and 2.g). These tasks were not part of the original plan but were added based on performance testing conducted in the previous week, which showed that using menu item descriptions retrieved from Square greatly enhanced the quality of caption generation. Further details are provided below.
- Week 6, 7, 8: I will proceed with working on features 6.a and 6.b, which previously had no assignee.

Implementation

1. Feature 2.f Businesses Management - Square Integration

Implemented OAuth-based Square account linking via Connect/Disconnect buttons.

- On Connect, users are redirected to the Square Auth URL to log in.
- Depending on the result, displays the appropriate message.
  - On error due to user denial, shows: "You chose not to connect your Square account. You can try again anytime."
  - On other errors, shows: "It seems something went wrong. Please try connecting again."
  - On success, displays a toast notification.
- Access tokens are stored in the backend Business model.
- Initial sync fetches transaction-level sales data from Square, aggregates it into daily revenue records, and updates the last sync timestamp.
- On Disconnect, removes the relevant data including the access token, resets the UI to the disconnected state, and displays a toast confirming disconnection.

2. Feature 2.c Businesses Management - Sales Data Management

Developed workflows for retrieving sales data from Square, in addition to supporting CSV file uploads.

- If no sales data is available, users are prompted to either connect their Square account or upload a CSV file.
- CSV uploads automatically save data to the database and update the current day's revenue.
- A Refresh button (displaying a spinner while loading) fetches only new data since the last sync, stores it as daily revenue, and shows a success toast upon completion.

3. Feature 2.g Businesses Management - Menu Items Management

Added view/edit for menu items (name, description, category, variation name, price).

- Users can search menu items by name, description keywords, or category.
- Edit opens a modal to modify fields; upon submit, changes auto-sync to Square.
- This information is used for caption generation.

4. Demo Video Link

[AI Marketer v2][Feature 2cfg] Square-realted Settings.mp4

Team Leading

Features Document Update: I updated the Features Document by reflecting added/removed features, updating developer assignments, standardising feature descriptions, and adding a Progress Date column in line with the Gantt chart. Team members were encouraged to fix any date mismatches. [AI Marketer] Features v3

Week 8

Implementation

Feature 6 AI-driven Promotion Suggestions

This week, I created a comprehensive feature plan document that outlines the complete workflow:

1. Sales Data Collection - Collecting sales data through CSV file uploads and Square integration
2. Data Processing & Storage - Processing and storing sales data with product-level details
3. Data Analysis & Presentation
   a. Display Top/Bottom Product Performance Charts - Visualising best and worst performing products b. Implement AI-based Analysis Logic - Analysing sales patterns for promotion opportunities
4. Promotion Suggestions - Generating targeted promotion ideas based on sales analysis
5. Promotion Tracking - Measuring promotion effectiveness by comparing sales data

I successfully implemented steps 1, 2, and 3.a of this workflow:

- Enhanced the existing Sales Data Collection by extending it to capture product-level information instead of just daily totals

- Improved the Data Processing model by adding `product_name`, `product_price`, and `units_sold` fields to the `SalesDataPoint` model
- Refined the Square API integration to extract detailed product information from order line items
- Expanded the data visualisation endpoints to return not only overall sales data but also top and bottom performing product charts

These changes transform our system from simply tracking daily revenue totals to providing detailed product performance analytics that will serve as the foundation for our AI-driven promotion suggestions.

For the next week, I'll implement steps 3.b and 4, developing the AI analysis logic and promotion suggestion generation.

Team Leading

- Support Request: Drishti asked for help with frontend-backend integration and a database issue, after getting Docker running on her machine.

- Guidance Provided:

  - She mentioned Docker, so I explained that Docker isn't essential for solo development. It's more useful for team consistency or deployment.
  - She mentioned a database issue, so I explained that Postgres isn’t necessary for her features (social login, forgot password) and suggested SQLite instead.
    - Postgres: The better choice for production-level applications, especially when handling larger datasets such as sales data or needing scalability.
    - SQLite: Easier to set up and works well for smaller, educational projects, but not ideal for production environments.
  - She mentioned frontend-backend integration issues, so I encouraged her to share more details so I could better assist with troubleshooting.

- Follow up: Drishti informed me she switched from Postgres to SQLite.

- Support Request: Drishti asked for an explanation of the backend code, particularly the structure due to the large number of files, and inquired about whether user account data is stored in the database.

- Guidance Provided:

  - I provided a detailed overview of the backend structure, including the roles of all apps in the project, the files in the `users` app and their respective functions, the authentication workflow, and the authentication settings.
  - [Feature][1] Authentication System.pdf

- Follow up: Drishti informed me she started working on implementing the `delete account` feature and shared `setup report` documentation.

Week 9

Implementation

Frontend Authentication Middleware Implementation

[General] Frontend Authentication Middleware Implementation.pdf

Uploaded a technical document outlining the new authentication middleware system for the frontend. This implementation replaces the previous AuthProvider-based approach with a more secure and performant middleware architecture using Next.js. The document explains how server-side authentication is now handled using HTTP-only cookies, describes routing protection logic, and includes backend integration details. This transition improves user experience, prevents unauthorized access at the routing level, and aligns with best practices for modern web security.

Feature 6 AI-driven Promotion Suggestions

Workflow:

1. Sales Data Collection - Collecting sales data through CSV file uploads and Square integration
2. Data Processing & Storage - Processing and storing sales data with product-level details
3. Data Analysis & Presentation
   a. Display Top/Bottom Product Performance Charts - Visualising best and worst performing products b. Implement AI-based Analysis Logic - Analysing sales patterns to prepare input for AI
4. Promotion Suggestions - Generating targeted promotion ideas based on sales analysis
5. Promotion Tracking - Measuring promotion effectiveness by comparing sales data

3.b: Preparing Input for AI I implemented the process of preparing structured input data for the AI model. This step is crucial to ensure the AI receives high-quality data for generating effective promotion suggestions. The input includes:

- Sales Performance Data
- Business Data
- Feedback History

For more details, please refer the attached file: [Feature][6] Promotion Suggestion Generation.pdf

4: Managing Promotion Suggestions Once the input is prepared, the AI model generates tailored promotion suggestions. The full lifecycle of these suggestions – including how users trigger, view, accept/dismiss suggestions, and leave feedback – is documented with UI screenshots in the following file: [Feature][6] Promotion Suggestion Lifecycle.pdf

For the next week, I'll implement workflow 5 and testing on the deployed server.

Team Leading

- Task: Support integration process for Drishti's work.
- Context:
  - Drishti was using a different development environment (SQLite and no Docker) due to repeated setup issues.
  - Because of this, full integration testing on her local machine was not possible.
- Timeline and Actions:

Day	Description

Day	Description
Monday	Provided Drishti with clear integration guidelines 1. Create a separate branch for each feature 2. Merge her work 3. Submit pull requests one by one
Tuesday	-
Wednesday	Drishti informed me that she submitted a PR for the “forgot password” feature. I provided the following feedback: - Drishti used a generic branch name (drishti-dev), so I advised her to name it after the feature (e.g., forgot-password). This is because the standard Git workflow is to have one branch per feature. - She didn't fill in the title, description, or reviewers, so I explained that these are needed for proper review, as a PR is a way to notify teammates and request a review before merging. - She accidentally created an incomplete PR and asked how to fix it. I clarified that PRs aren't snapshots but reflect the latest changes in the branch, and they can be updated anytime, including titles and descriptions using the “Edit” button. - She set the PR to merge into `main` branch (default), so I advised her to change it to `dev` branch, since we're still in the development phase.
Thursday	I noticed a couple of basic issues with Drishti's PR, so I provided feedback and guidance to help her resolve them: - There was a build error on the frontend server. I advised her to check the error using `npm run build` locally and fix it. - The latest changes weren't merged before her PR was submitted. I advised her to merge the latest updates from the dev branch into her branch.
Friday	Since it's the end of the week, I followed up with Drishti to finalise the review process. - For the backend, her pull request had already been merged before I could complete the review. I asked her to revert the merge commit and open a new PR so I could properly review it. - For the frontend, I requested that she let me know once all review comments have been addressed, so I can do a final check before merge.
Saturday	Since I hadn't heard back from Drishti: - For the backend, I proceeded with reverting the merge commit on the dev branch to address the issue and am waiting for a new PR to be opened. - For the frontend, I am still awaiting confirmation from Drishti once all review comments have been addressed, so I can perform a final check before the merge.

- Task: Sales pitch creation ownership
- Context:
    - During the team weekly meeting, feedback was given about the importance of having materials to introduce new services to people who are unfamiliar with them.
    - A sales pitch for the new service (AI Marketer) was identified as one of these important materials.
- Timeline and Actions:
    - April 28 (Monday):

        - Drishti volunteered to take on the task of creating the sales pitch.
        - She asked for my opinion on the format, and I suggested a video or PowerPoint as options.
        - I researched and sent links to examples of video sales pitches:
            - How-to Video for AI & SaaS Product | LangEase
            - How-To Video for Fintech Product | Praxis
        - Drishti will finalise the sales pitch direction after completing the feature implementation and share it with the team.

Week 10

Implementation

Feature 6 AI-driven Promotion Suggestions

Workflow:

1. Sales Data Collection - Collecting sales data through CSV file uploads and Square integration
2. Data Processing & Storage - Processing and storing sales data with product-level details
3. Data Analysis & Presentation
   a. Display Top/Bottom Product Performance Charts - Visualising best and worst performing products b. Implement AI-based Analysis Logic - Analysing sales patterns to prepare input for AI
4. Promotion Suggestions - Generating targeted promotion ideas based on sales analysis
5. Promotion Tracking - Measuring promotion effectiveness by comparing sales data

With the completion of Promotion Tracking, the implementation of Feature 6: AI-Driven Promotion Suggestions is now finished.

- 📎 For detailed information on Promotion Tracking, please see the attached file.
    - [Feature][6] Promotion Tracking Implementation.pdf
- 📽️ Please refer to the demo video below for a full workflow demonstration.
    - [AI Marketer v2][Feature 6] Promotion Suggestions.mp4

Integration

Forgot Password Feature Enhancement and PR Merge Complete

The Forgot Password feature originally implemented by Drishti has been enhanced and successfully merged into the main project. The following updates were made as part of this process:

- Unified environment variable settings for consistency.
- For better consistency, direct `fetch` calls were replaced with `apiClient`.
- The entire password reset flow has been tested and is functioning correctly. The Forgot Password feature is now fully integrated and operational in the main project.

Team Leading

As we approach the final stages of our project, we have around three weeks left to wrap things up. To ensure everything stays on track, I've outlined a plan for the remaining weeks and shared it with the team.

1. Feature Completion

- Deadline: Sunday, May 11 Extended for Drishti to Tuesday, May 13
- Tasks: Roll back and cancel unfinished features
- Note: No feature work after Sunday

2. Deployment

- Deadline: Thursday, May 15
- Tasks
  - Deploy frontend on Vercel, backend on Render (@Heeran)
  - Create tutorial videos for deployment (@Daniel)
  - Finalise documentation and write test scenarios (@Drishti)
- Note: No code uploads during this phase

3. Full Testing

- Deadline: Friday, May 23
- Tasks
  - Test everything on the deployed server following the test scenarios (@All)
  - Fix minor issues and hide/restrict/delete major issues
  - Clean up unnecessary files/code

4. Final Documentation

- Deadline: Friday, May 30
- Tasks: Write the final report for university submission (@All)

Week 11

Deployment Validation and Optimisation

Deployment configuration, Code cleanup, bug fixes, and feature optimisations performed. This includes configuring environment-specific settings, removing unsupported features, addressing cross-environment compatibility issues, implementing workarounds for hosting limitations, and enhancing the user experience by preventing error-prone workflows.

Category	Title	Description	Commit

Category	Title	Description	Commit
Deployment Configuration	Environment-based database initialisation	Database setup logic was updated to run conditionally based on environment variables. In development (DJANGO_ENV=development), the system performs database schema changes by creating migration files (make migrations), applies those changes to the database (migrate), and flushes (clears) the database if FLUSH_DB=True. In production (DJANGO_ENV=production), the system only applies existing database schema (migrate) without creating new migration files or flushing the database, ensuring that live data remains safe and intact during deployment.	Backend
Code Cleanup	Removed Social login support	The Social Login option was removed. Please refer to the Project Timeline and Progress Overview below for further details.	Frontend Backend
Code Cleanup	Removed Passkey support	Removed the Passkey login option following the decision to discontinue support. See this note for details.	Frontend Backend
Code Cleanup	Removed Image Analysis feature	Removed the image analysis feature that identified objects in uploaded images due to discontinued support. Refer to this note for more information.	Frontend Backend
Code Cleanup	Removed Caption Generation with Image feature	Removed the caption generation with image feature that would create social media captions based on image analysis as it could not be completed within the available time and would have required additional costs.	Same as above
Code Cleanup	Removed Dark Mode feature	Removed the Dark Mode feature as part of a design simplification effort, focusing development resources on core features.	Frontend
Code Cleanup	Removed Refresh Token handling	Removed refresh token support to simplify authentication flow and reduce code complexity.	Backend
Code Cleanup	Removed Twitter & Threads support	Removed the Twitter and Threads support following team discussions, as Twitter access requires a paid plan and the focus is on other features. Backend functionality and any non-visible frontend code related to these remain unchanged to allow potential future restoration.	Frontend

Category	Title	Description	Commit
Code Cleanup	Refactored imports to follow PEP 8 style	Standardised all import statements across the backend to align with PEP 8 guidelines. These changes improve readability, maintainability, and help enforce consistent code quality throughout the project.	Backend
Bug Fix	Fixed logo image rendering issue on Vercel	Resolved image distortion on the production environment by replacing zero width and height attributes with appropriate values. This issue occurred because local development environments rely primarily on CSS for image sizing, allowing images with zero width and height attributes' to display correctly. However, in production, Next.js's image optimisation requires explicit width and height values; setting these to zero causes improper rendering and image distortion.	Frontend
Bug Fix	Fixed incorrect redirection issue	Resolved an issue where users were not properly redirected after authentication due to middleware's inability to access cross-domain cookies. Fixed by moving the authentication check and redirect logic from server-side middleware to the client side. This workaround is necessary because the frontend and backend are hosted on different domains using free services (Vercel and Render). In the future, using a unified domain for both frontend and backend will allow restoring middleware-based authentication handling.	Frontend Backend
Bug Fix	Fixed image loading issue on Render	Implemented temporary media handling via Discord to resolve image hosting issues on Render. Render's free plan does not provide persistent disk storage, which limits direct media hosting. As a workaround, images are temporarily uploaded to a Discord server via webhook (`TEMP_MEDIA_DISCORD_WEBHOOK`). To support this, I configured `NEXT_PUBLIC_IMAGE_DOMAINS` to allow Discord CDN URLs for image optimisation in Next.js. In the future, if media is served directly from the backend (with a paid Render plan) or through a dedicated service like AWS S3, the `NEXT_PUBLIC_IMAGE_DOMAINS` value will need to be updated accordingly.	Frontend Backend
Feature Enhancement	Restricted post creation w/o linked account	Prevented users from accessing the post creation flow when no social media account is linked, providing clearer feedback and guiding users to connect an account first. This ensures a more consistent and error-free user experience by avoiding failed post attempts due to missing account connections.	Frontend Backend

Deployment Tutorial Video for Team

This brief guide is intended for the team and assumes technical knowledge. It is not designed for non-technical users. Daniel will create a separate tutorial video for non-technical users, starting from the sign-up process and covering all necessary steps.

Deployment Videos

- [AI Marketer v2][Deployment Guide] Vercel.mp4
  Covers the full deployment process using Vercel, including repo setup and project configuration.
- [AI Marketer v2][Deployment Guide] Render.mp4 Explains the deployment process on Render, including how to update environment variables.
- [AI Marketer v2][Deployment Guide] Vercel 2.mp4 Focuses on how to manage and update environment variables on Vercel post-deployment.

Creating Environment Variables

- [AI Marketer v2][Deployment Guide] Discord.webm

Project Timeline and Progress Overview

1. Feature Completion (Completed)

- Deadline: Sunday, May 11 --> Friday, May 16 --> Saturday, May 17
- Tasks: Roll back and cancel unfinished features
- Although both the Delete Account feature and the Google Places integration were functioning locally, neither had been shared to the main project repository by the Friday deadline. On Saturday, it was determined that integrating both features was not feasible given the time constraints, and the team agreed to proceed with the essential Delete Account feature only. This feature was successfully integrated by 12 PM, marking the completion of the Feature Completion phase.

2. Deployment (Ongoing)

- Deadline: Thursday, May 15 --> Tuesday, May 20
- Tasks
  - Deploy frontend on Vercel, backend on Render (@Heeran)
  - Create tutorial videos for deployment (@Daniel)
  - Finalise documentation and write test scenarios (@Drishti)
- I completed deployment validation and optimisation first (see details in the table above), then proceeded with deploying the project. While deploying, I recorded the entire process and shared the videos with Daniel as a reference to help him create a clear tutorial guide for new app owners (see video links above). I plan to verify that the deployment was successful by Tuesday.

3. Full Testing (Upcoming)

- Deadline: Friday, May 23
- Tasks
  - Test everything on the deployed server following the test scenarios (@All)
  - Fix minor issues and hide/restrict/delete major issues

    - Clean up unnecessary files/code