

FY 24-25

My brief work story of FY 24-25

This was a high growth and a lot of modifications year for the company as well as for me. We decided to shift the primary focus from India to the US and also from a polling app, aka poll-based social media, to a poll-based dating app.

Thus, I worked on understanding the dating market, looked at competitors, and understood what the US audience likes (specifically Gen Z), what their preferences are, and what kind of content we can serve them.

I even helped improve the Shortvibes, PollabilityScore, CommentabilityScore, NSFW score and This or That prompts. For this, I understood the backend working of these scores and short vibe, also looked at the prompts and the results they are giving in output, and based on all of these, I wrote the points to work on and optimize these prompts.

Here, I even used my prompting expertise to improve the results of these prompts, make required tweaks in parameters like temperature, model type, etc. and also created an automation system that will take input from Google Sheet, run the GPT as per prompt, and write the output back on Google Sheet. This process helped reduce the manual effort of a few days to manually run the prompt for testing on sample data, send it to tech for further testing via automation, check its result and then tweak or finalize it based on the outcomes. I used Make.com and GPT API for this

These things may look simple and with low impact, but in reality, it's quite impactful as PollabilityScore, CommentabilityScore, and NSFW scores decide what content is good and what should be shown first, and Shortvibes is the most important thing as the more good this 1 line is when the profile of a particular user is shown, the more chances of that user sending the wave to another user and they both getting matched as it's the top thing shown to the user (after the photo of the user) when we recommend them to another user.

I also helped reduce the time to create various reports with the help of the effective and efficient use of GPT. Some of these reports are:

- 1) Competitors' ads reports, which we used to collect top ads, their CTA, target audience, and their transcript,.. etc
- 2) User rating report wherein we collect the ratings users have given on the App Store and Play Store, understand their sentiments, compare them with previous ones and create a summary of what's the overall sentiment, what issues are highlighted by users, what users are liking, etc

I also continued the content experiments to ensure we were providing the best content while also taking care of content diversity. A few things I handled are:

1) Anime Polls:

I primarily selected the Anime category for the experiment as we found that it is one of the top categories users are selecting during the onboarding category selection on the app, and still, the content on the app around this category is very low, so we wanted to push the category by supplying more polls around the same on the app. Also, Gen Zs are more in Anime and pop cult as it was perfect to target them too

We created 5 batches of Anime, and each batch had like 10-15 polls, and they were promoted on the app for 10 days, so in total, the entire campaign ran for ~2 months. Every time I created a new batch, I looked at the performance of the previous batch, like which polls were working, how they were performing, whether there was any specific pattern emerging, etc, and optimised the next batch based on it. We also used good UGC polls so that we can also push their content.

When we concluded the experiment, what we got is:

50% increase in polls published in the Anime category compared to what it was before we started publishing polls and promoting the category

10% increase in category share of Anime

Users who viewed and engaged with Anime polls had 10x more Daily time spent on the app

Users who created and published polls in Anime had 2x better DAU/MAU and had 2.5x more Daily time spent on the app

2) Video Polls

This experiment was primarily planned to increase engagement in terms of comments and shares, as well as the time users spend on the app. And also help increase app content diversity and promote the feature to add video in the description as it was newly launched.

Here I primarily focused on Instagram to find and filter good video content for India, and for the US, I focused on TikTok. After looking at numerous reels and TikTok videos, I filtered out videos that will be used as descriptions in video polls. Also, a bigger issue was if we manually uploaded polls one by one, then it would take a bit of time, and also we would need to create various accounts for the same, so I worked closely with tech and got the script to upload the video polls developed, tested it, got required changes, and finally, we had the script to upload video polls.

I finally uploaded the 1st batch of video polls for India and the US, each having 10-15 video polls and got them fixed and promoted. We got good attraction, especially in India, so we looked at the kind of video polls that were working well, and for the next batch, we doubled down on them. We also used good UGC polls so that we can also push their content.

As we were done with both batches below were the results and few limitations we found:

(1) Number of session per day had increased by ~17%, and time spent per polls had also increased by ~20%, so users were spending more time on the app as other users were also creating video polls

(2) The cost to surface the video polls was a bit high for us, so we had to stop actively promoting and fixing these polls

(3) As it was a new feature, we faced various issues like UGC video polls not getting enough distribution, users facing issues while publishing these polls, etc, which limited our scope to promote the feature

(4) So we stopped actively promoting the video polls but still continued for a small number of polls 3-5 video polls

3) Fashion and Makeup Polls

I continued the experiment to publish polls in Fashion and Makeup as I saw that it was working very well based on the 6 batches I had created till now for this experiment, each batch having 10-15 polls.

Further, I created 5 batches of Fashion and Makeup during the initial months of FY 24-25, and each batch had 15-20 polls, which were promoted on the app for 10 days. Every time I created a new batch, I looked at the performance of the previous batch, like which polls were working, how they were performing, whether there was any specific pattern emerging, etc., within those 10 days and optimised the next batch based on it. Here, we found that image polls with female characters worked very well for the polls, so we doubled down on such polls. We also used good UGC polls so that we can also push their content.

When we concluded the experiment, what we got is:

80% increase in polls published in the Fashion and Makeup category compared to what it was before we started publishing polls and promoting the category

100% increase in category share of Fashion and Makeup

Users who viewed and engaged in Fashion and Makeup Polls had 300% better Median Message Sent, 680% better D30 Retention rate,

850% better DAU/MAU ratio, were spending 470% more daily time spent on the app compared to the ones who didn't view and engaged in Fashion and Makeup Polls

Users who created and published polls in the Fashion and Makeup category had 62% better DAU/MAU and 20% better daily time spent compared to ones who didn't create and publish polls in the Fashion and Makeup category

4) Dating Advice and Fake Conversations Polls

This experiment was basically planned to enhance engagement and user interaction, inspired by subreddits like Bumble Girls and Hinge Stories. These were designed to resemble real conversations that users typically do on dating apps, which are hooky, spicy, and interactive.

For this experiment, I looked at various subreddits like Bumble Girls, Hinge Stories, etc. and also Pinterest to collect some hooky and spicy conversations between users and further modified them to refine them as per how we can use them as poll descriptions and even made them a bit spicier. I then used two internal accounts to simulate realistic dating conversations. Messages were sent between these accounts to create a fake yet engaging and relatable conversation that could prompt reactions from users. These conversations were then converted into polls with attached images, aiming to encourage users to expand, comment, and share more actively. For this 1st batch, we created 10-15 image-based polls and also found some TikTok videos and stories from subreddits around dating and dating advice, using which we created 5 video-based polls and 10 text description-based polls that had stories and were asking for advice.

I got these polls promoted and fixed for ~1 month and had also created a Mixpanel Board to track their performance. After 1 month, I created an analysis report comparing with other Fixed and promoted polls and also normal polls getting published on the app. Here, it was found that key metrics like Comments to Impression, Share to Impression and Expand Count to Impression

were 4x better for Fake Conversation polls compared to other Fixed and Promoted Polls.

Thus, based on this, I again worked on creating a new batch of polls around the same lines, and this time, we even optimized things based on the learnings of the previous batch, like what kind of polls worked. This batch is being reviewed and will soon go live.

Moreover, the work of continuously updating fixed and promoted polls is as it used to be so that the NUs and RUs (new and returning users) user experience is top-notch. Also, I made sure that as we are moving from a poll-based social app to a dating app, the majority of questions are around dating and relationships and others also close to them, so it's no longer quirky or random stuff that has no connection to users' lifestyles, behaviour, hobbies, psychology and dating preferences. Some of the fixed polls experiments that had a direct correlation here are personality-based fixed polls and category-based fixed polls wherein there were polls from each and every category so 2 users could better understand what the other one was thinking for XYZ, etc.

Further, I also continued some of the works this year too like:

1) Supporting marketing team with PR and Appstore campaigns:

I continued work on this work item this year, too. Here, I created and promoted polls on the app to collect responses from users for PR work while making sure they seem like just another poll on the app to make sure the responses are genuine.

Also, I created hashtags and polls to be fixed on the app to support our Appstore campaign, which will help us increase organic visibility. Two such campaigns were the Valentine's Day

campaign, wherein we ran the #Valentine and created polls around it, and another was around Pets, wherein we ran the #PawfectMatch and created polls around it. Polls for both of these campaigns worked quite well as they had 4x better Votes to View and Engagement Rate.

2) Trending Topics Work:

I continued work on this work item this year, too, in the initial months, and as we shifted to dating, I stopped this work item as it was no longer needed and didn't make more sense to work extensively on this. Here, I researched trending topics every week, found relevant keywords related to them, created polls around them, and fixed these keywords and polls on the app so that users could engage with them and create more polls around them.

I created about 30 batches this FY 24-25 for this work item, which basically meant finding 5 trending keywords and topics every week for India and the US and creating 10 polls each for India and the US (2 polls per keyword/topic). Finally, fixing their keywords, getting them promoted on the app and creating Mixpanel Boards to track keywords as well as polls.

Finding the trending keywords and topics every week was also not that easy as we had to look at Google trends, Instagram, YouTube, TikTok and Reddit trending things and find relevant topics to our target audience, ones for which we can create polls and these topics should have at least some presence on the app so that we can fix the keywords. Based on the Mixpanel Board, we were able to track which keywords were getting higher attraction, so we were every time refining the selection of keywords based on what kind of keywords and topics have performed till now.

3) Doing Adhock tasks:

(1) A few tasks that we regularly do is getting the Shortvibes for campaign polls updated wherein we run the optimized prompt for shortvibes and get the shortvibes generated for each and

every option manually (I have made the automation now in March 2025, but till then, things used to be done manually), also getting the PollabilityScore, CommentabilityScore and NSFW tags updated so that the campaign polls gets there required visibility.

(2) Constantly creating new content for the app so that we can daily push 30 new polls on the app, which makes the content on the app fresh.

(3) Constantly keep track of fixed polls performance and flag things as well as get it resolved as 1st priority to ensure minimum impact on NUs of the app.

(4) Create Mixpanel Board and Analysis report for each and every campaign going live on the app to measure its impact, get important learning from them and also roll back things if it's not working as expected, rather negatively impacting user performance.

(5) Maintained and got the required modification made in Coda to handle and upload new content types like GIF and video-based polls on the app.

(6) Got the scripts developed to upload various types of content on the app, i.e. text-based, video-based, image-based, etc, to ensure that we don't have to upload content of polls one by one via Coda.

And many more things can be seen in the stories of Adhock's work item.

I learned about SQL so that we no longer have a greater dependency on the data and analytics team, and we can run queries for the analytics task and uncover various details about users and their preferences. I even downloaded Dbeaver and learned to run queries on Postgresql and Redshift. I even helped my coworkers learn it and what I primarily did was:

1) As my manager gave me a task to extract the data for XYZ from the database, I break the requirement into what logic needs to be applied and what we are expecting as output

2) Based on this, I look for tables that will have this data. Once I get the table/s that contains this data, I give GPT the requirement, logic, tables and their columns so it can create an SQL query for the same.

3) If I get error, I again give it to GPT, and if it's something clearly visible error, I solve it myself

I taught this method to my coworkers, and now they are able to handle things by replicating this blueprint. Finally, I'm becoming less dependent on GPT and am able to write at least simple queries myself.

I also handled some of the most important POCs for the app, which will transform the direction and play a key role in the future success of the app. Some of them include:

1) Voice-Based User Onboarding:

Here, the plan was to build voice-based AI agents that will help do the entire user onboarding on call rather than the user needing to open the app, go through various screens provided and fill in all the details.

For this work, I first researched the tools we can use to create an onboarding flow, add an LLM on top of it and a voice agent to handle the entire conversation. I tried tools in which I needed to create a complete user flow like Synthflow, Voiceflow, etc., but here the issue was each block should work properly and as expected, but as I had limited time to complete the POC, I couldn't learn these tools by going through their long tutorials so I took help from GPT to understand how to create blocks, add logic, etc. but I was continuously running into errors which weren't getting solved.

The next day, I tried to give a shot at tools wherein I could just give the entire onboarding flow as a system prompt, like PhoneCallBot, Retell AI, and ElevenLabs. First of all, I tried

tools like PhoneCallBot and Retell AI and explored the system prompt and tweaked basic parameters like speed, voice, background voice, latency, temperature, llm model, etc. After doing it for various calls, I got the basic idea of what to fix and was confident that these tools would work perfectly for us.

Further, I created an initial document to submit my initial findings and sample voice calls I had done with the AI voice agent. After that, I jumped to make them much more reliable and advanced by storing the data of onboarding details in their respective variables like name, age, gender, etc. For this, we needed to write custom functions and add them to the call so that they could be executed properly. Initially, with the help of GPT and a few videos on YouTube, I wrote a custom function to handle age wherein if the user is below 18 years old, the AI voice bot will gracefully decline their entry on the app; however, if they are 18 or above they will continue with the onboarding flow.

Further, I improved the system prompt based on the feedback I got from stakeholders from the initial document I submitted. I also now went one step further and created a custom function to store values in variables for all the details we were asking the user so that we can easily send it to the backend and they can create the user profile on the app.

However, as we were doing POC, rather than sending it to the backend, I planned to use Webhook and Google Sheets. I learned to connect Retell AI with Webhook and then Webhook to Google Sheets via Make.com. For this, I looked at a few YouTube videos and also did some trial and testing to fully connect everything. And it worked wonders, and efforts paid off

I tested this multiple times and also optimized the system prompt for various edge cases, such as how to handle users' silence, how to handle wrong info collected and override it, how to handle questions other than onboarding, etc.

I created the final document with all the learnings, tool details, prompts, custom functions info, sample recordings, Google Sheets and Make.com details, etc. and handed over the work to the Product Manager so they can now take it to the next level of implementation on the app.

2) Voice-Based Match Proposal:

Here, the plan was to build 2 voice-based AI agents, i.e. single and multiple proposals. A single proposal-based voice agent will be used as a premium offering to users wherein they can use it to propose to users who will want to get matched with, and then the agent will call the potentially targeted user and try to convince them why they both can be a great match. The multiple proposal-based voice agent will be used to send the top 5 matches of the week to the user and try to convince them why these are the top users to match for them. Both these agents are voice-based, so the user will talk with them, increasing the chances of considering a match with the proposed users

First of all, I started working for a single proposal agent, and here, I created a Mixpanel Board to extract the user info of the users who have recently received a good number of waves on the app and have also voted on good enough polls so that I can extract their data and use as a sample for single proposal agent. Here, extract a female user to whom various male users had sent the wave, and further from Postgresql, extract their poll data, such as what polls they voted, what all they voted, etc.

Further, I created a GPT prompt that will take the user info and polls data and help me transform it into a structure that contains like what these 2 users have in common, what is their personalities, why they both are a good match, etc., for each pair of male-female from the sample data set I've taken for this work. After multiple trials, I was good to move ahead as I had got the structure I wanted.

Now I jumped to ways in which I can execute this task and tools to use. I was clear about using Retell AI and Eleven Labs, so I

focused on them and explored the ways in which I could execute this work. Primarily, I found 2 ways: Either add user data at the end of the system prompt or add it to the knowledge base.

I started with the way to add user data at the end of the system prompt. So I started working on the system prompt for this and covered all important edge cases I can initially think of like what if user asks questions of the prompted user around something we don't have data of, what if user asks questions of other things which have no relation to Hunch or this proposal call, what if user isn't sure about the proposed match, how to nudge them 1-2 times if they are unsure, etc. and also handle things like not force them if they aren't interested, how to understand users said and unsaid words, tone and voice of agent is energetic, agent doesn't repeat same things multiple times, etc.

Based on numerous trials, I arrived at the system prompt, which works effectively along with other parameters like the LLM model, voice speed, latency, etc. Further, I also tried another way of adding user data in the knowledge base rather than in the system prompt and also made the required changes in the system prompt. After a few trials, it worked as equivalent to the one wherein we had added user data to the system prompt.

Now, I document everything, including both the ways, system prompt, parameters settings, sample recordings, etc., and share them with my manager. Based on his feedback, I added a few other user details like their zodiac sign, dating preferences, etc. I made the required modifications in the system prompt, tested it, and further worked on generalizing the system prompt so that we could easily scale it. The examples included in the prompt to make the agent understand what to do and how to behave were made generic using placeholders. Finally, after making tweaks here and there based on responses from the agent, I was ready to send it over, so I shared the final sample recordings and updated document with him, as well as handed it over to the product manager so they could take it forward for implementation.

I now started working on multiple proposal agents, and here I used the same data extracted for a single proposal agent but worked on modifying the structure of how the user data will be included in the system prompt for proposal as this time it was no longer 1 user getting proposed to another user rather we wanted to proposed top 5 users to the particular user every week. This time, I made things much more extensive by even including the compatibility score and date ideas for the two users, a similar way done for other pairs, so this time, the agent has more things to share and as there are more folks to compare the user doesn't feel that they aren't getting enough info around the proposed matches and why they both are a good match.

Once I was confident about the user proposal structure, I created with the help of GPT for each pair from the sample data, I moved to the system prompt. Here, too I couldn't just copy the same as single proposal agent as in multiple proposal agent we were proposing 5 users to a particular user which can be quite overwhelming if not handled well so looked at ways in which we can make it less overwhelming like giving more control to the user on if they want to hear more about a specific user or if they want an agent to start based on compatibility score, if they have liked a specific user based on the conversation with an agent do they still want to hear about other users or not, etc. Also, we will need to handle more nuanced edge cases; thus, keeping all of these in mind, I started working on creating a system prompt and went ahead with the way of adding the user proposal structure data at the end of the system prompt.

After numerous trials and refinements of the system prompt and parameters like voice, llm model, latency, speed, temperature, etc, I was to the point wherein it seemed the best in class. I finally made the generalized version of the system prompt, so it's scalable, and after final tests, like every other time, I shared the voice samples and created a detailed document covering everything with my manager and finally handed it over to the fellow product manager for further implementation on the app.

3) Hunch IP: Onboarding to Date:

Here, we are trying to be independent of the Hunch app and do the voice-based onboarding of the user, take a fee to find a date for them, send them a top proposal-based on the user details like age, gender, dating preference, smoking, drinking, etc habits and also their MBTI type which will be generated based on new questions we will ask them around it and finally as 2 users accept each others proposal it's a date. In the entire process, voice calling is the main focus area, so we will be using voice-based AI agents to handle everything.

Basically, we are trying to reduce friction, make things much faster and more action-oriented, and use AI voice agents to make the entire thing scalable.

This POC is currently in progress, and what we have done till now is explore the other apps that are doing something similar, plan and map down the working of this new IP and tools to be used and also create the flow chart of the user journey. I also looked at what technical difficulties we might face as our team has no one from the tech team, so we are trying our best to optimize things. I've also explored new tools that will take photos or the name and city of the user and extract their publically available info from the internet, which will further help us reduce the onboarding time and also make the user journey from onboarding to date much faster.

4) Personality Trait Model:

Here, the plan was to research various personality type models and try to understand their working logic and further look into how we can implement the personality type model on Hunch. This was primarily inspired by dating apps like Boo and famous personality test models like 16personality test, which provides users with a personality type based on their responses to a set of questions they are asking and further also gives details of what this type means; other personality traits based on the type, their compatible types, etc.

For this work, I started with researching the prominent personality type models, what they mean, where they are being used, whether they are open-sourced, what's their working logic and what kind of content they are covering to derive the personality type they are giving, etc. I initially used AI tools like GPT search, Perplexity, Gemini and Claude for this and also went through various articles and papers published around them. Doing this entire exercise gave me a fair idea of what prominent personality type models are present, what all are used in different dating apps, what their pros and cons are, etc. I created a document based on all the things I got from my initial research so that we can narrow down to take a call on what personality type models we want to focus on so that I can then go deeper into understanding them. I even proposed a hybrid model to cover the pros of the top models and avoid the cons so we can make it more efficient and optimized for accuracy.

Based on the discussion around the document and my initial research, we planned to initially avoid a hybrid model as we will then need to make users aware of it and rather than it initially, we should use a prevalent model as users are already aware of it. So we went ahead to finalise exploring the MBTI, Big Five and HEXCO models in depth and then again discuss which one suits us.

I started giving personality tests, which were designed around MBTI, Big Five and HEXCO models and also explored the apps that were using the models. I even looked at their history and official websites to understand if I could get the extracted working and the framework that we could replicate. Although I couldn't find the exact framework and working open source, except for Big Five, I still had enough info that I could go through, backtrack the tests and connect everything to get to the working framework of these models. I also found that if an app like Boo is using the MBTI model, then the MBTI results of Boo won't be the same as that for the 16personality test MBTI as every platform or app has made their own modification and tweaks in MBTI.

Now as MBTI was the top most priority, I started backtracking the MBTI test of Boo and found that it was asking questions around all the four types i.e. Introversion (I) vs Extraversion (E), Sensing (S) vs Intuition (N), Feeling (F) vs Thinking (T) and Judging (J) vs Perceiving (P) and each question belongs of one of these type and gets scored for the type based on what we selected on a scale of 1-7 (Disagree to Agree). Based on this, I created a detailed document covering the working and framework of MBTI, Big Five and HEXCO models, their pros and cons, which one would be best for Hunch, what other apps have employed, etc.

Further, I worked on whether we would get the personality types based on the current polls for MBTI, Big Five and HEXCO models so that we can take a better call on what all modifications we will need to make to ensure that the new personality type models can be deployed properly. For this, I first of all downloaded the data of internal users from the app, like their user info and poll data and based on it, I generated their personality types for all three models, i.e. MBTI, Big Five and HEXCO, wherein I explained GPT about these model types and also gave the detailed document so it can get the best results for these polls. Here, as we got all the personality types for these 3 models for all internal users along with the explanation of why this type was assigned, we reached out to internal users, explained to them about each type and asked them if they felt it was the right personality type they had. Based on the feedback of users, we came to the conclusion that as per the current polls in MBTI, the biggest issue was in identifying Introvert and Extrovert personalities and in general, for all model types, the accuracy was ~50-75% which is bit low thus it became clear that we will also need to change the type of polls we are currently using. I even created a detailed documentation for this and shared it with stakeholders and my manager so they can better take a call on what to do next and how to proceed.

Meanwhile, stakeholders were reviewing these documents, one more task came in hand and it was to look at Match.box, which allows users to host events and use it to match 2 attendees of the event based on the questions they ask during the entry of the event and once all attendees are in the host can run their algorithm via just a click on their app and then Match.box will create a group of users who can be great with each other and

finally after some time will drop the matches among the groups. This concept was getting a lot of attention and was also fun to do, so we wanted to understand how they were doing this thing and if we could do something similar for personality types or our offline events.

Again, for this task, I looked at their website, read articles written around them, and read articles written by attendees stating their experiences. Further, I again used GPT search to understand more about them and finally collected good enough info like what kind of questions they were asking and in what buckets these questions were mainly broken down into, like religion, hobbies, dealbreakers, etc. I also learned that they add bonuses to scores and impose penalties for dealbreakers. An important breakthrough was when I got a connection between Marriage Pact and Match.box and how the Marriage Pact Algorithm is being used by Match.box as it's their parent company, like Marriage Pact is running Match.box from the backend.

After this, I now started exploring more about Marriage Pact and the algorithm they use, questions they use, working and backend framework etc., but it's all proprietary, however I did get to know about their broader working. I read some of their old interviews and articles wherein it was mentioned, like what they used to follow during that time, which further helped me better understand what might be happening at the backend.

Now, to backtrack the work, I planned to pay Match.box and do a test event in a way that they don't catch us for this work as we can be direct competitors. I got the premium event for 20 people and took help from the folks working in Hunch to give the quiz so we could capture what X person answered on a particular question, run the algorithm once everyone was done and then, based on the answered on every question and the results of the algorithm we can backtrack things. I got the quiz filled by 10 males, 8 females and 2 non-binary folks so we can understand how friend and love matches are given by Match.box

Now that I had the results and details about what was filled out by everyone, I collated everything in a single sheet and gave the complete details to GPT around Match.box and Marriage Patch and instructed it to help backtrack things. I started to get the breakdown of things, and I was constantly prompted to make sure that it was not just making things up but also understanding and giving the best version of what could be happening at the backend. Once I was ensured that it was able to move and reveal the correct direction, I started off to test the formulas and ways it gave me, which might be used at the backend by Marriage Pact. There were few assumptions we needed to make, such as getting the original weights, how and what all questions are dealbreakers, what is the penalty for dealbreakers, etc. were hard to crack and rather needed a constant trial and error process of changing various parameters in various combinations to get the perfect answer however even if I was able to generate the same output based on assumptions it would be a win for us as further things could be handled by data and analytics team.

For the test, I extracted the internal users' user info and poll data and, first of all, converted it to the scale format, which is used by Match.box. Further, I tried to run the same steps via GPT, which the Match.box would do on the backend. This was a bit of a lengthy and complex process, so once I had all the assumptions being made, I extracted all the data in Google Sheets and added the required formulas to make the calculations. So this way, I manually checked the main data points of users to ensure that there were no errors and the calculations were done properly. Repeated the process for all user pairs and then gave it as input to GPT to do the final step of doing the matching and generating the matching reason and found that it followed the same pattern as Match.box to generate the matching reason and also provided the match pairs.

Further, to ensure it is working well, I gave the user data of the quiz we did in Match.box, but this time I didn't give the results and followed the same process of letting GPT make the assumptions of weights, dealbreaker questions, etc and I did the complex calculation on Google Sheet, and finally, as I gave the calculated scores as input it generated Match % and match reason and gave the match pairs which were the same as Match.box so it was clear that we need to do trial and error to overcome the

difference in Match % of our algorithm which we cracked and the one Marriage Pact uses else results of match pairs are same.

Finally, I documented all the details of the Marriage Pact and Match.Box also covered the entire process of how we backtracked things and what we found along with the formulas and working logic and structure. I even shared the 2 sample tests I did to check it was working. Till the time I was working on this, a decision was made to go ahead with the MBTI personality type model, and we will include it on our app.

Now, my task here was to help create the MBTI polls, which we will use on the app to generate the MBTI type of users. Based on my understanding till now, the way we structure the polls on our app and what questions Boo and other MBTI-based personality tests and apps are using, I created a list of 80 polls with the help of GPT, 20 polls each in 4 buckets i.e. Introversion (I) vs Extraversion (E), Sensing (S) vs Intuition (N), Feeling (F) vs Thinking (T) and Judging (J) vs Perceiving (P). I generated them bucket by bucket and then went through the polls and filtered out the top 5 polls that I felt were the best in that bucket. I even asked GPT to do the same and wherever we both had selected the same questions, they were final, and the ones wherein there were different questions selected, I cross-checked with GPT and asked why XYZ was selected and not ABC and based on it the replies I settled the case of top 5 polls in each bucket and sent it to stakeholders. Based on their feedback, I made a few improvements and sent the new list of Top 20 polls. I also created a test that anyone can give for these polls, and they will get their MBTI type.

Moreover, as stakeholders asked to make the total number of polls to 12 so I further selected just the top 3 polls from each bucket and created a list of the top 12 polls. Also, created its test and shared it with them so they can give both tests and check which one is more accurate. The project manager who was handling this project used the polls I had used in the top 12 polls and gave it a dating tweak, and proposed it along with the test I had created so we could understand which one was working well. Based on the results, almost every test had the same

accuracy, so they went ahead with the dating-based top 12 polls, and got it implemented on the app.

5) ShortVibes based on First Date Ideas, Debate Topics & Netflix, Amazon Prime, Spotify and YouTube Recommendations:

We planned this work to bring diversification and newness to the current shortvibes structure. Wherein we planned to recommend First Date Ideas, Debate Topics & Media Recommendations (Netflix, Amazon Prime, Spotify and YouTube Recommendations) to users just like shortvibes are shown.

Before we planned for this work item, I had done some analysis on what type of shortvibes type are shown more to users i.e. Sun-sign based and poll based. Here, we found that more than 50% of the users were shown sun-sign-based shortvibes, which are based on the zodiac sign and a bit generalized, so as they were shown more to users, it was diluting the impact of shortvibes for users. Thus, to solve that issue, we planned to make things much more personalized and diversified based on First Date Ideas, Debate Topics & Media Recommendations.

Now, for this, I extracted the sample user data from Hunch, such as basic user info and poll data. Then, I started working on creating a GPT prompt for First Date Ideas, Debate Topics & Media Recommendations, wherein it will go through the data of users I have extracted, and generate First Date Ideas, Debate Topics & Media Recommendations, give their explanation of why was this recommendation suggested for a particular pair of user and also cover what all data of these users was used for the same and finally it will also give a confidence score of how confident is GPT on this score. The final prompt was created based on multiple results and optimizations every time a result was generated so that I get the best output from it. A few issues that were tackled were the hallucination of user data and creating data that doesn't exist for the user, repeating the recommendations, and trying to create all the recommendations, i.e. First Date Ideas, Debate Topics & Media Recommendations rather than the one which makes the most sense, etc. I also made

some rules for it to follow to generate these recommendations, like they should have voted on the same option for the recommendation to be generated.

I documented the results and their progress and shared it with my manager. Based on his feedback, I made modifications like creating separate prompts for First Date Ideas, Debate Topics & Media Recommendations, so it is no longer a generalized prompt that contains all of them together. Also, I made some changes in rules and created separate prompts based on them like:

- (1) Users should have voted on the same options in a particular poll
- (2) Users should have voted on the same poll irrespective of whether they voted on same option or not
- (3) No rules on voting on same option or same poll

Again, here, I ran it multiple times and optimized the prompts for each of them. What I found was that for Debate Topics, things weren't working great as we primarily need things wherein they both have opposite views to get better debate topics. Also, in general, as now we were generating them individually, sometimes recommendations were generated even where it wasn't that much required, although the confidence score solved it. This was better managed when the prompt was generalized, and all of the recommendations were included in a single prompt and generated based on it. Also, for the case "No rules on voting on same option or same poll" I expected better output as there was more freedom with GPT to take on calls on when to generate, based on what to generate and which one to generate, but the results were quite the opposite and seems like the worst choice to go ahead with. Thus, I documented all of these points, new prompts for all of them and the results associated with them and shared them with my manager.

Based on the feedback from my manager, I worked to further optimize the Debate Recommendations by adding more samples of debate recommendations within the prompt given to GPT. However, I got no significant improvement but a sweet spot from where things can basically work.

Now, I started writing a detailed document around this, which will cover things like Current State of Shortvibes, Why Are New Shortvibe Recommendation Types Required?, How the Proposed System Solves These Issues, New Shortvibe IPs Logic, Make Shortvibes Clickable, A/B Testing Plan, Success Metrics, Experiments Conducted and Their Results, GPT Prompts of First Date Ideas, Debate Topics & Media Recommendations individual as well as generalized. In this document, I even made the plan to use the explanation given by GPT as a premium feature wherein premium users can click on shortvibes, and they will get a detailed explanation of why this recommendation is given to them. I even explained various cases to be used as logic and how to create these recommendations and further use the confidence level generated by GPT to take a call on which recommendation to serve. As this document reached the stakeholders, they liked this idea and were ready to take it ahead.

Unfortunately, due to a lack of tech bandwidth and other priority work items in hand with product managers, this work item remained unworked. Finally, after months of no work around this work item, I took the responsibility to at least run a small-scale experiment and, based on it, decide further to take things ahead. For this, what we planned is just to replace First Date Ideas and Media Recommendations with the sun-sign-based shortvibes repository, which will lead to minimum effort from tech, and we will also be able to test things.

For this, I talked to the product manager who handled the feature of sun-sign-based short vibes, explored the edge cases we need to take care of, how things should be handled, and what all things we should expect to be changed as we replace the repository of sun-sign-based short vibes with First Date Ideas and Media Recommendations repository. I wrote the prompts to create the First Date Ideas and Media Recommendations repository respectively. For Media Recommendations, I extracted top and trending YouTube videos and creators, Spotify songs and artists, Amazon and Netflix TV shows and movies based on which the Media Recommendations can be created. I optimized the prompts with each run for both of these recommendations so that I get the best results. After that I manually went through the First Date

Ideas and Media Recommendations and filtered the top 250 recommendations (150 First Date Ideas and 100 Media Recommendations) and created this repository. Finally, I wrote the document covering everything required to take this experiment live and handed it over to tech.

Promotion Engine V3

This year FY 24-25, we finally decided to make a new V3 version for Poll Promotion and Fixing Interface from scratch which will be independent from Amazon Personalization as in the initial version, we were constantly facing issues primarily because of 2 things:

(1) Issues from Amazon Personalization wherein whenever it was retrained, we started facing issues with our campaigns, and our fixed and promoted polls didn't work as expected. Also, as it's a black box for us, there was very little we could do here.

(2) Changes in features in feed: With constant changes in feed, like the addition of an NSFW tag to identify 18+ content, PollabilityScore and CommentabilityScore to identify how good the poll is for polling and commenting so that good ones can be shown 1st all of these led to issues as now the Poll Promotion and Fixing Interface needed to be constantly updated for these features and even a small issue in implementation here was leading to impact on reach of our campaign.

(3) The versions developed till now didn't have in-depth capabilities to show if, for a particular user, our campaign was picked or not and if not why did it happen and what things didn't match. So it was getting hard to do an analysis of the same, leading to issues regarding who is responsible for this.

(4) Also advanced capabilities were missing in previous versions like the ability to promote and fix more than 30 polls in a campaign, get the polls within the campaign shuffle (to overcome the issue that just a few polls get most of the impressions), ability to give priority to campaign and run multiple campaigns for the same target audience, etc

Based on all of the above things, I wrote a requirement document to create a new version V3 of Poll Promotion and Fixing Interface, which will be independent of Amazon Personalization and will directly run from the database, which also makes it scalable and provides more control to us. I also made a detailed comparison of the current working and what we want in the new version, and how this new version will be helpful.

I even made sure that the Data and Analytics team is synced here and also aligned based on what we are planning to do as they manage the working of the feed, and the changes proposed will directly impact the feed structure and its working so it was important to get things aligned. After multiple rounds of discussion and a few changes in technical implementation based on recommendations from the Data and Analytics team, we were ready to move forward. I played the role of product manager here and made sure that development was done based on what we planned, I even daily tracked the movement of work to make sure things go out on time; however, even after development was done on time we were stuck on testing as QAs were busy with other priority tasks in hand and also due to shared resources with other team it was bit hard to get dedicated time.

Till the time being, I acted as the QA and wrote the cases I could think of to test this feature and also took help from GPT for the same. This proactiveness helped QA spend less time on creating cases and rather directly testing them. Also, before they did it, I was done testing the cases and raising them with developers so they could start fixing them. I prepared a document around the basic structure of the webpage we will need to build to make sure that the campaign created for fixing and promotion from Poll Promotion and Fixing Interface is easy to operate and has clear visibility of what all campaigns are running, with what all conditions and for what timeframe.

I spent time with them, being on calls, motivating them and helping them with solutions as much as I could. Whether it was a call on which they were solving an issue that arose while testing or QA was facing issues with some testing and needed more clarity on the expected outcome, I was present. We spent

weekends and late nights working on this, as it was quite a big project that would have an impact on the entire feed.

Finally, the hard work paid off, and we launched it successfully along with the admin webpage through which I was able to create, delete and handle all the fixed and promoted campaigns on the app. And it's been like 8 months till it's launched, and it's working as expected.

In order to automate things, use various AI tools efficiently, apply various prompting techniques, etc., I have spent 99% of my weekends constantly learning about them and upscaling myself by looking at YouTube videos and tutorial series, attending webinars, practical sessions, did courses from various places and what not. It's a long journey and a loop of learning, learning, more learning, implementing and learning more of it.

In FY 24-25 we scaled down on the intern side as we were moving quite aggressively, and they weren't able to maintain the same pace and also 2 of them's performances were dropping, so slowly and steadily, we removed them and again started handling things single-handedly until we planned to finally get 1 intern which was Anshika. I managed the entire process of finding her, taking her, and getting her onboarded, and finally, she is currently working with me as a coworker, and I manage her.

Even though I removed 2 of them due to performance, I still get their messages; they send me reels like Send this to your manager who helped you in your job and was always there for you, etc, which I feel is an achievement to have.

During the initial 3 months of FY 24-25 during which these 4 interns were working with me, we worked on various things like:

(1) Creating and filtering new content for circles, which was a feature we had launched wherein you can create your own circles with friends, and instead of options, there are names of friends in your circle, which you need to select who you feel perfectly suits for the poll question. You can even be part of other circles, obviously, and it's like a game as you have limited reveals to get to know who selected your name in a particular poll. We worked quite hard to make things work here and also made efforts to Indianize the content for India.

(2) Creating and filtering good UGC polls for Fashion and Makeup experiments. As we saw that polls with real human images as options were working quite well, especially of females, so it was quite an effort to find good images for these polls while also making sure we didn't use copyrighted content. We explored Pinterest, Instagram, TikTok, and whatnot to make sure we are serving the topmost content and tracking it continuously to ensure we are on the right track.

Further, I got them to explore the feed for both India and the US feeds and told them to engage in F&M and also categories they feel to have a good grip on and post comments and interact with other users & their polls. This way, they understand more about the users and content on the platform and figure out things that we can improve from our end. Moreover, they also ended up getting vibe score, followers,... They also created a report on this task, which also brought in some good insights.

(3) In order to increase Hunch's social presence on other platforms and organic visibility, we planned to help marketing on this front. We looked at our competitors' content, how other social apps scaled themselves and made themselves visible in their initial days, etc. For now, we are planning things for India, so we went ahead with Instagram as a platform wherein we will put our efforts. Further, we looked at what could be done, like a broader idea or plan we could execute. Everyone brought some great ideas like:

Vox Pops - Asking questions (like on Hunch) to people at different locations, CP, Khan Market, Colleges. Getting their opinion on taboo topics or embarrassing topics + Asking

embarrassing questions to opposite genders + Office Reel, creating content in office, asking their opinions on different taboo topics

Having a Hunch Character (The logo) and making reel content around it. For eg - That character will be reading the controversial polls, sharing opinion on controversial polls or be reacting on the viral polls on Hunch

Reacting to polls on Hunch

Rating the answers of a question or a poll

Using the template to create a meme reel, For eg - World before posting that embarrassing poll, World after posting that embarrassing poll.

Day in the life on hunch(the app), For eg - What type of polls people upload etc

Polls will decide what we will do in a day - decide by polls. The Hunch interns (acting as hunch users) will post polls about what he/she should do, and the users of Hunch will vote on those polls. Interns has to do the task which that have maximum votes and the list goes on

Out of these ideas based on a detailed discussion we went ahead with 2 ideas for initial POC:

Polls will decide what we will do in a day - decide by polls. The Hunch interns (acting as hunch users) will post polls about what he/she should do, and the users of Hunch will vote on those polls. Interns has to do the task which that have maximum votes

Vox Pops - Asking questions (like on Hunch) to people at different locations, CP, Khan Market, Colleges. Getting their opinion on taboo topics or embarrassing topics + Asking embarrassing questions to opposite genders + Office Reel, creating content in office, asking their opinions on different taboo topics

Once the topic was finalized, interns started working on creating rough scripts for both of the ideas, and after I reviewed them, I provided feedback on what to improve, what more could be done, etc, and finally, they started storyboarding it.

After multiple rounds of review and further detailing, 4 interns were divided into groups of 2, and they started doing a POC wherein 1 would shoot things, and the other would be in front and vice versa for their videos. This was like a test for them to make themselves comfortable talking to strangers, in the case of Voxpop, and also to face the camera and shoot the videos. This will help them as they officially shoot with better equipment like professional mics and cameras.

Here 1 group did well while the other one didn't, nor did they submit the samples on time, and also, their samples weren't that great. Also, we were seeing their drop in performance for some time, for which I had also warned them, so finally, we decided to drop them off. So now we were left with just 2 interns, and they too left in some time as they wanted a full-time position; however, we weren't completely ready for it as there are plans ongoing to move to date from polling. We then started looking for a new intern, and that's when we got Anshika.

As Anshika joined as an intern to support me and take up the daily routine tasks like creating new content to be uploaded on the app, creating and planning content for new experiments like Anime polls, video polls, etc and supporting me with new IPs like Voice AI Agent, Personality Traits, etc., she has grown from an intern to full-time employee.

I played a pivotal role in guiding Anshika Gupta's transition from an intern to a full-time content strategist. Beyond just assigning tasks, I ensured she had the space to explore, learn, and develop critical skills across multiple domains, including AI tools, automation, prompt engineering, SQL, analytics, and psychological research.

I encouraged her to step beyond her initial responsibilities of poll creation, introducing her to advanced tools like Mixpanel, PostgreSQL, AI-powered voice agents, and automation platforms like Make.com. By providing structured guidance, sharing insights on industry best practices, and fostering a culture of

continuous learning, I helped her build confidence in handling complex tasks.

My leadership extended beyond technical mentorship—I supported her in navigating corporate challenges, refining her content strategy approach, and optimizing workflows for efficiency. Her journey from an intern to a full-time contributor reflects my commitment to fostering growth, ensuring interns are not just completing tasks but evolving into valuable long-term assets for the company.

This success underscores my ability to mentor, empower, and guide talent, ensuring they contribute meaningfully while developing essential professional skills.