Summary details text

i have a bachelor in engineering from BITS Pilani in computer science and engineering. Apart from that, i have also completed Develop Your Skills as a Marketing Professional certificate on LinkedIn earlier.

okay. so let me give you a broad timeline of my work experience. i started off in 2018 with Capgemini as an associate consultant where i stayed for two years. i moved on to upgrade in 2020 august, staying there till nov 2022. i joined almabetter in dec 2022, where i have been working. the ngo was a volunteer experience between 2019-2022, so three academic terms. we have covered upGrad and NGO till now. let's start with Capgemini now. ask me relevant questions in order to perfectly highlight the skills, especially transferable skills. questions should be relevant to the industry and domain.
I am familiar with Scrum and Agile and have worked with JIRA, Trello, and Basecamp. I have managed a team of up to people indirectly while managing 2 APMs directly. the projects I managed were marketing initiatives, growth endeavors, learning content development, and learning operations. In terms of tools, I am familiar with Excel, Metabase, and Google Sheets - I can work with SQL-based tools too, with some practice. I have carried out descriptive, and diagnostic analyses. I have collected data through surveys, web scraping, and onsite ratings. Consider I am fairly proficient in working with data in Excel. I have taken sessions on Excel for my team members and students too. This has helped me recommend data-backed suggestions to leaders such as redirecting marketing budget, deciding on shoot style for learning content, project management software to use.

at upGrad, I was awarded the Star award for consistently high performance. My courses (edtech products) received consistent ratings of 4.5+ for over a quarter, and also for suggesting process improvement changes wrt content creation that reduced the deviation in timelines by 15%. I am adding my bullet points from resume for upGrad, gauge them for information they convey and ask clarifying questions if required for the role. we will discuss NGO details in a later prompt.
points: Managed a team of 5+, steering cross-functional team collaboration to ensure a holistic learning experience with a customer satisfaction (CSAT) score of over 80
● Oversaw a project budget of Rs 8 Lakh, reducing the expenditure by 10% in the first quarter through vendor selection and contract negotiations
● Collaborated with business leadership to set priorities based on business needs, resource capacity, and risk exposure - reducing deviation by 15%
Product Strategist September 2020 - June 2022 upGrad, Bengaluru
● Owned the product life cycle - gathering product requirements, conceptualizing product format, developing the product road map, cross-functional team collaboration to create the product and post-launch product support
● Performed customer interviews, analyzed platform data and subjective feedback, acted as the SPOC for industry experts and faculties to launch products with a customer rating of 4.51/5
● Created and added a new feature to a product with an international customer base, raising the average customer rating by 5%

So, the team was primarily made of content developers (product strategist role). cross-functionally I worked with the video development team, program managers, language review teams, SME, vendor, and legal teams. it was part of my job to coordinate with all other teams to ease my team's work. In vendor negotiation, I onboarded new vendors at a lower rate than the current vendor with no dip in quality. the product feature that was added was based on user feedback to make the content more interactive. so, i worked with h5p to develop interactive quizzes and online labs for learners to experiment and learn a concept. i want to better frame the vendor onboarding with no dip in quality point.

let's move on to NGO work ex. So, I worked with U&I, which works in underprivileged children's education. I was a volunteer who taught the kids directly - where I designed the lesson plans for the weekly/monthly level. next year, I took on a leadership role of a center leader where I indirectly managed 40+ volunteers while directly managing 10 of them. I was in charge of designing the center's SOP, worked with the fundraising team as a coach, collated and processed data on student progress, worked on community development. Since it was a new center that was started, we had to setup all SOPs and relationships from scratch- although we got assistance from the head office in setting things up. In my third year with U&I, 50% of the selected leaders were from my team that i directly managed.

okay. my work experience at capgemini can be divided into two projects. the first one was with wills towers watson to upgrade their CMS. the role required my team to replicate the content onto the new platform. this was a tedious task that required duplicating each element of the webpage to the new cms. i created an automation using kantu that allowed me to automate the steps taken to create the page in the new CMS. this reduced our delivery time by 2 weeks, out of 5 months. afterwards, i moved on to a different project where the client was walt disney.

since this was my first project as a professional, i focused mainly on execution. some other responsibilities were daily update mails and weekly reports - which i automated through VBA. this project was solely handled by my team. this was not a very technical role. i had regular client interactions including directors at WTW. i self-learned the automation part. my major learning from this project was my first interaction with the concept of automation which i utilised everywhere later in my work, also how to think about reports and stakeholder management/interaction.

great. at walt disney project I was part of the team that handled the system of records for the client. so, it was an application called Team Box that took care of all reservations across Disney parks, hotels and transport. my team was responsible for L3 support (assuming support levels of the service industry) interacting directly with the clients. I worked on issue resolution and bug reporting. i worked with Splunk for triaging. My day-to-day included working on issue tickets that were unresolved by the L2 team, generating performance reports, daily status updates for other stakeholders such as our onshore team and client team members, and preparing documentation for any recurring issues.

the unresolved tickets were assigned by the team lead. I was to pick tickets based on the assigned priority. the ticket usually had screenshots with a short description of the error. The L2 team added some logs from the system that reported the error. since my team had an overall picture, I was able to pull out other relevant logs, tracking the request from the point of origin to failure. this allowed me to trace the API calls made and understand the error in the payload or the system. based on the settings of the application, I was also able to highlight if it was a user-generated error or a system error. Coordinating with the onshore team required sending an EoD mail and a quick catchup to convey high-priority tasks, if any. then there were issue-reporting emails that were sent to the development team and clients to keep them apprised of high-impact issues or recurring issues that required developmental changes. I also worked on product deployments where I was responsible for coordinating with the deployment manager in order to monitor the health of the systems and coordinate with all the other teams involved in the deployment such that SOPs for deployment are being followed. I had prod server access and deployment access. I also worked on AWS as we migrated some servers to AWS - I am aware of cloud logs, how to read them and work with them. One major incident that I can recall was that once there was an issue with the system as it was unable to take bookings for a peak season time. this caused a major issue as it led to a direct loss of revenue and resulted in a major call with all stakeholders. based on the logs, I figured that there was no issue with the API calls. so, next, I went to check the configurable settings for holidays. turns out, that the holiday package team had setup the package dates correctly but the hotel team did not make the rooms available for the dates for the hotels added to the specific packages. there were other factors that were not properly configured too. my major learning from this project was how to interact with top middle management and prepare reports.

moving ahead to my experience at almabetter. I joined as a program manager for the content team where I had two direct APMs reporting to me. The first project was to manage the development of a data engineering track at Almabetter, while my reportees handled two other tracks. this project required me to work in Basecamp where I created the SOP to use Basecamp, content development, reviews, and launches. this made me well-versed with basecamp including its limitations. Since Basecamp does not provide you with detailed data/analysis of project velocity - I suggested moving the projects to Jira. Since this was a new tool for the team to take up, I spent time learning the tool from scratch, documented the learning, and created the SOP for content development and reporting. I set the whole process on a scrum board using workflows in Jira, setup automation for instant reporting of status changes to the concerned stakeholders, and analyzed/created/shared reports on the progress of the project. A few challenges while setting up this project were the lack of knowledge in the team, resistance to a more complex but robust system, and defining the work unit. here's how I tackled each problem: I created a video along with confluence documentation that highlighted the relevant states, transitions and use cases for each type of role in the project (content author, reviewer, language review, program manager review, final quality check). The second issue was the complexity of the system which was tackled through multiple automation allowing the authors/PMs to fill in minimal details while ensuring all relevant details are being filled - through custom/calculated fields. when I took this project up the problem statement was to ensure that we have consistent reporting, along with a reduction in the number of errors. when I moved on from the project, we were able to take a lead of 7 weeks (starting from 1 week) on the live classes for which the content was being prepared. Also, this process led to a reduction in errors by a whopping 90%. this also allowed us to measure the efficiency of the team, through understanding the progress of the project through reports.

okay. my major criteria for choosing Jira were its reporting capabilities and automation. also, I wanted to gain experience in using Jira. since no one in the team had ever worked with Jira before, the gap was evident. As mentioned above, I ensured that each person had the specific knowledge required to perform their roles. apart from that I took KT sessions for my team of APMs to explain the whole workflow, rationale, automations and limitations. my main resistance was due to the fact that it was being tracked on Google Sheets so moving to Jira was a pretty big step. I addressed the limitations such as the lack of info for when a specific thing was completed unless filled in explicitly and with multiple teams/people involved in the process it would not have been feasible to get correct data every time. parallel, the documentation prepared was clear enough to highlight the meaning of each state of the artefact and all possible transitions that the artefact can do from there. these documents were prepared for each type of person or role interaction with the project. since the documentation was highly customized for the specific person who was using the project, it was picked up relatively easily by the team later. for error reduction, the introduction of three levels of quality checks was the major reason for the reduction in errors. Apart from this, I ensured that the team had access to top-notch grammar and plagiarism software such that content was error-free. this initiative was well-received by the leadership as they were able to see regular reports and a sprint retrospective was planned to drill down on any lack in performance. I organized the scrum ceremonies, deciding the flow of the standups, parameters for planning sprints, documentation guidelines, and report formats. getting to the lead part of the project planning, I was able to split the whole curriculum into multiple components, defining the time it takes to complete each of them through inputs from the team. The status of all artefacts in progress was taken up by the respective APMs for their tracks, which was conveyed to me on a bi-weekly basis. This drill-down allowed us to track which component took more time than expected (so we tweaked the framework or time to fit), who was not performing after repeated interventions, quality of work vs time, and team-level efficiency.

since we were able to break down and get times for each component, we were able to figure out how much time was required for each lesson, each chapter, hence each module, hence the whole course (project). the info regarding the time taken for a specific type of content to be developed was taken from a survey sent out to the whole team. few other factors were considered such as leaves, depth of content, and additional materials required. so, i balanced the workload using units. so, a chapter might have a different number of topics but each topic has a fixed framework of components to be created. So, we tracked the components and divided them equally among each member of the team. Also, their individual skills were taken into account when divvying up the chapters/modules. Regarding identifying the risk, since we had sprint retrospectives and daily status updates, we were able to create a risk toleration framework which was essentially that at any point in time, neither of the team members should have less than 2 lessons completed per week. so, as soon as this was a recurring issue for a couple of weeks - we had a discussion with the person to understand the delays. in case, this was an acceptable delay - no changes. Else, if it was mitigable issue, relevant steps were taken. the timelines and project details were communicated over an all-hands call and over a mail. the project timeline was adaptable as there might be some unforeseen challenges as it is a creative project. since we had a baseline of unit work, and minimum work that can be completed per person per week, we were able to meander the project as per our aim. the project was later scrapped as the firm wanted to cut costs and stopped using jira. But, in the longer term i would have wanted to scale the project with the same parameters as the current system of scoping/planning/creating has built in modularity and unitary to scale up. This was a very crucial project in terms of learning as I got to know in and out of Jira software, i understood an performed scrum ceremonies, project planning, identifying risks, planning mitigation, and how to get everyone on the same page.

okay - let's move to another project that I was working on. So, I also handled the video requirements from the learning/marketing team. The video production was a part of the marketing team. I worked as a liaison between all teams and the production team in terms or video production. Whether it be learning videos, intagram reels or website videos for product details - i was the POC. So, all shoot schedule, scripting, editing instructions, upload and publish was managed by me. for metrics, there was never a miss in terms of delivering the videos on time - 100% timely delivery. this was a continuous project - all video requirements were flowed to me and then communicated to the team.

okay - let's move on to the next project. I worked at the intersection of multiple teams - with a larger overlap between the learning team and the marketing team. I headed the technical content development project for the website. So, my team of 4 worked on developing blogs, carousels, reels, pdf notes, and other such items. We also created tutorials - which were like short courses on a particular topic such as data visualization or chatgpt tutorial. my role in this project was to ensure that we delivered a fixed number of artifacts for each type - the number of which was decided based on marketing targets. I set up the process for working on blogs and website content for both technical topics (my team) and non-technical topics. I created a sheet, put up relevant filters, outlined and streamlined the process in the tracker itself, added efficiency calculation, and added MIS for tracking the progress of this charter. I worked with the marketing content head to finalize the list of topics that were conveyed to my team. Through this project, we were able to increase our session times by 30% and decrease our bounce rates by about 40% - as per SEO numbers. Overall, I set this technical marketing vertical up from scratch, introducing new processes, standardizing delivery timelines, and ensuring top-quality work through relevant reviews and rework. to give an idea of how well the project was managed, we were able to build a lead of more than 2 months of content - this was a risk that was foreseen and mitigated as the team also worked with the learning team and would have had to pick some work from there if need be. So, i gave the team targets such that we always had a lead of 1-2 months so if the team was unable to work on any content for 2 months, we still would have been able to publish blogs - and this happened.

let's move on to the next project at Almabetter. At Almabetter, we wanted to create production-quality videos for our courses. currently, those were live class recordings. taking this as a problem statement - I kicked this project off. firstly, there was a decision to be made in terms of video format since the shooting process would differ. I calculated the timeline and budgets required for each format (offline studio shoot with graphics/ online shoot with graphics/ screen recording only) and then finalized the shoot format after discussion with the COO. I managed the search and discussions (Executed by the APM) with SMEs, planned project timelines, and inventory, and set up the process for the project. the project was executed by the APM.

yes, there are more projects at almabetter. I set up a ticket management system. Since Almabetter is an edtech, it is a consumer-facing business that receives quite a few tickets from learners on a daily basis. moreover, some are recurring issues. one other problem that we were facing was that the learning team was being tagged across all kinds of issues and this affected their bandwidths. primary aim for me was to streamline the ticket resolution process. i worked with the support team lead and one of my APMs to set a process for ticket resolution. firstly, i asked for a detailed breakdown on types of tickets that the team saw. then we identified categories that were easily resolvable or redirected to other relevant team. for example, tech tickets were directly assigned to the tech team. one category was of tickets where learners were unclear with the instructions on how to submit the project. we proposed and implemented a few solutions. firstly, an FAQ doc was to be prepared for all issues that were recurring and could be answered by the support team - non-technical queries, process queries, etc. next, channels were created/modified on slack with proper description on when to use which channel. for example, all sales team related doubts were restricted to one channel. further, in order to capture and track curriculum or content errors - they were moved to a specific channel with a set automation. the automation was set using slack's workflow integration. in the error channel, the support executive would trigger the workflow which would pop out a form with the required details such as name, learner name/email, error type, error description, and attachments. these details once submitted, would trigger a message on the channel along with adding a row to a spreadheet. The APM would then keep a track on the issue and followup to its completion.

The next project that I worked on was setting up an assistance program with the placement team. here we aimed to provide enhanced assistance to learners enrolled in our placement assistance program (known as AlmaX). the problem statement here was we needed to decrease the time it needed for a student to get a job. the process was designed such that the learner received guidance from the learning team at each stage of their placement. so, starting off with the assignment till the final stage of the interview - the team provided assistance wherever required. I also provisioned for business communication sessions for HR rounds or non-technical rounds. My main role was to set the process for the system which included defining the steps, stakeholders for each step, how learners were assigned to each person in the team, compensation/reward for the team members, and checks and balances for the process. we also introduced an improvement wherein a Slack message was sent from the spreadsheet that had this data as soon as a row was updated, tagging relevant people. through this process, we were able to reduce the number of interviews it took to land a job from 7 to 5. the stakeholders here refer to the internal team - the placements team and the learning team. I explained the process on a call, created a document, and demarcated the process on the tracker through color coding. apart from that, there was a weekly call setup between the leaders to understand the progress and mitigate any issues. the guidance was in terms of explanation of projects, providing additional reading materials, interview prep, etc. These were correctly tagged and stored in terms of role so that they can be reused later. initially, the team leads were required to assign the learners based on team members' workload. later, it was decided based on individual skills and bandwidth.

the next project that I worked on was integrating webengage usage within the learning domain as well as marketing. within the learning domain, I set up journeys for learners enrolled in our courses. for our bootcamp courses, the journey was setup so that they receive 5 emails throughout their first week with important information regarding the live class schedule, joining process, learning journey details, and other such relevant information to kick off their course. this required me to understand the current WebEngage setup of the organization and suggest relevant modifications to user attributes so as to enable these journeys. the other part was setting up marketing funnels and journeys. I was heading the event's charter at the organisation (we will discuss this project later) and I wanted to understand how our page performed. I checked with the tech team and asked them to introduce a few web events such as when a person lands on the page, clicks on the register now button, and finishes the registration. this helped me understand at what points lead drop off from my system. what i realized was that there was almost a 50% drop just after the users clicked on register now. this was due to the fact that the system asked for a Google login which led them to a different tab and even after login were not redirected to the correct page. this was a very big blocker for registrations. so, based on competition research we suggested that we remove the Google login redirect and add a form that opens as a popup to capture details. this change resulted in a slight reduction of drop-offs, let's say 20%. One more case that I realized was that even if a lead was logged in to the website, they still would have to fill in all these details. based on this suggestion, the product team introduced 1-click registration, which brought down the drop-offs by another 5%. One more usage for WebEngage was to understand the pattern of these API responses to design our communication strategy. so, the day before the event received the highest interaction/registrations. so, we scheduled comms for the highest-intent audience on that day. similarly, channels were also decided.

a note: while working as a program manager with the learning team, I was involved in planning discussions for all the learning ops projects. for example, capacity planning for the upcoming quarter, budget discussions, SME quality reviews, learner feedback action decisions, specialization launch planning - restructuring project timelines and resources as per the launch, and other activities that a program manager office lead does at an edtech firm. please provide a finalised summary for the project including this note.

penultimately, there are a couple of projects that I started but was unable to finish due to a recent round of layoffs. first was a collaboration with the business development team. here, I was working on creating a repository of resources for the team. the resources were built on a use-case basis such as what to send on a first interaction, what should be sent after the first call, what should be sent as a follow-up, what should be sent if the lead is uninterested currently but will rethink later, what should be sent to totally uninterested leads, and a few other use cases. the resources were a mix of videos, blogs, brochures, texts, and other such media. the next project that I was working on was presales emailers or cold emailing. the aim of this project was to generate leads through cold emails. we had a database of about 50lac unclean leads - the cleaning was underway for the data. meanwhile, i charted out the strategy for the campaign in terms of segments, personas, email journey, emails, and other operational details to execute the project. in both these projects, there are no implementation numbers to show but the planning and metrics were all worked on and decided. i want to showcase that i have implementable plan ready for these kinds of projects (presales and sales communication campaigns). if you can interpret your own chat through a link - i can share the two discussions with you to further understand the plans.

this is a major project and I want it to have a higher weightage in my overall profile for a resume. I was leading the event's charter in the organization. I took over the charter in June. this was a cross-functional charter that involved all teams in the organization. starting off I was the sole person responsible for managing this entire charter. I set up the process for organizing events - right from deciding on a topic to finally executing the event. I can detail the process in the next message if required. overall, I was responsible for ensuring that the event's charter functioned smoothly. before I took over the project there was no detailed analysis present on the performance. we only had the enrollment numbers for each event. I set the data analytics for the event working with the analytics team - i worked on metabase to create dashboards relevant to each team and other such metrics. working with the marketing team, I set the content standards for events. this included working on communication styles with them, which channels to use when and how, what to send in each channel, and standardizing utms for all channels so that we understand the volume better and make data-driven decisions,. in terms of operational readiness, the APM who worked with me worked on creating an events calendar based on the theme, type of event (we had AMAs with alumni, AMAs with founders, and masterclasses - all conceptualized by me), topical relevance, course launch plan and SME availability (he sourced the SMEs, to be approved by me). He also took care of other operational readiness on the day of the online event. I have also hosted a few events. the author in my team also worked on the PPT for the event along with the SME. I also worked with the tech team to get improvements within the charter - the evolution of the events registration process and setting up data pipeline and dashboards are such examples. I setup all the trackers, finalized the processes, and metrics to be tracked, tracked the metrics, and suggested changes to spending based on those metrics. I also encouraged people to take ownership of their channels and suggest changes to the strategy. I also decided on the communication strategy based on the target segments - we used lead age as the metric for segmentation. this strategy was continuously tweaked as per the performance to get better results.

I successfully executed 26 events in the past 4 months - to be fair, I had an APM reporting to me for 20 of them to execute my strategic decisions. when I took the charter, the average enrollment per event stood at 400, and now it stands at 760. the average attendance numbers have also increased from 50 to 120 through these months. there was no process set before I came in. I set up the process from scratch and now we are able to close content development for each event in two days. our calendar is set up 1 month in advance (earlier it used to be decided 5 days before the event). this process led to enormous gains in team efficacy as everyone was aware of the calendar well in advance, and team leads could plan the deliverables as per the deadlines given. and all of this was being maintained on the tracker with fixed deadlines for each step so everyone was aware of where each event was in the pipeline. also, all updates were communicated on a Slack channel. moving on to metrics being tracked, since we were working for an online event we tracked each channel through utms. I defined the utms for the charter based on the already set framework within the marketing team. based on these numbers, I presented my findings to the leadership. one instance is when we looked at the numbers for registration and enrollments for channels we realized that although ads are bringing in leads they have no ROI. so, we stopped ads for a bit and utilized the amount to run experiments on LinkedIn and Facebook - which resulted in a 10% increase in the number from those channels. so, we had UTM data, enrollment data, and registration data on a minute level to influence our strategy. I also worked on standardizing the data and creating baselines to understand the performance - sort of creating 'a truth'. in terms of revenue, the charter directly impacted revenue of 9 million within the past quarter. in terms of ROI, we spent an average of 45k per event including marketing costs with an impact of 9 million. the impact of empowering team members was apparent in their dedication to improvement - everyone presented their ideas on an all-hands forum and shared the results of their experiments. One such example is getting influencers onboarded to promote events. earlier, we used to just get videos from them for our social media page. now, we also ask them to post for our events. this has increased enrollments by 4%. The collaboration between the departments was very smooth as we progressed. like clockwork, each team posted updates and picked up the relevant artifacts/tasks to be done. I did not need to monitor the D-day execution anymore. I have mentioned the decreased dropoff earlier - please use that here. also, we introduced banners across the website - all with the simplified registration flow too. this led to a significant increase in our average enrollments from 600 to 760. we also enhanced the quality of SMEs that we roped in to take the masterclasses. standardizing the utms allowed me to create a baseline to start analyzing. earlier, there was no tracking and we were unable to figure out organic vs inorganic traffic. this solved this problem primarily. at one point after this data was available, we realized that the inorganic traffic was not contributing much so we slashed our spending and worked on organic traffic through Discord, social media, telegram, email, and WhatsApp. after we stabilized the organic traffic (consistently hitting 450 for 3 events), we opened our spending budget for ads. this turned out well as then we were able to also increase our total registration numbers to 750+ within a couple of months through all the other changes too. from what I believe, the event's charter helped establish almabetter as a consistent organization that provided a learning opportunity every week through an interesting, innovative, and useful topic. this also helped almabetter position itself as an industry leader associated with a ton of SMEs working in top companies who also trust the learning experience at almabetter.

<Miles Education>

At Miles Education (Miles), I joined as a senior program manager with the CEOs Office. I have to switch between multiple hats.

so i joined at the itersection of marketing and sales team. starting off, I setup the utm tracking params and process, created trcker for the marketing team to store the data which would be used for tracking marketing efforts of over 2 cr per month. next, I was aligned more closely to the sales operations team. here, I was required to work with the marketing PM and the sales team to understand the whole funnel and optimize the same. i was responsible for generating reports, monthly analysis, daily analysis and reporting of sales metrics (called %, connected %, office visit %, among other relevant metrics). i worked with the team for about 3 months, during which I worked on automating reports through metabase. i discussed with my CTO to get metabase as our reporting tool so that we can create these reports as a public dashboard to be shared. once taken, I worked with the dev team to create dashboards on metabase, giving them detailed requirements, verifying data by working on excel, and setting up subscriptions. This dashboard saved about 2 hours per day of manual effort. Next, I standardised the report formats with the stakeholders. Since I had to work with two separate dbs to produce the final report, it was mostly done on excel.

After 3 months, I changed teams to move into the tech team - currently, CTOs office. Here I continued my work with reporting and analytics, primarily working with metabase along with a team of 2 analysts. i gathered requirements fro marketing, sales and tech team in terms of what do they want to see. Then I collated the requirements to condense them into a couple of dashboards with relevant metrics to be shown to business leaders. Next I created documentation (PRD) to share with the team mentioning the tables, db, and logic to be used across the dashboards. I managed the team keeping checks and balances to ensure that the work is on track and correctly presented. Post this activity, the team updated documentation for all the reports, questions and dashboards created which is now the Bible for all reports on metabase.

Once this was closed I picked up marketing automation tasks. Since I was the only one in the tech team with exposure to sales, marketing and tech, I became the natural POC for any tech developments for marketing /sales team. I took up the comms to be sent out through WhatsApp and email. I designed the journeys for business WhatsApp communication when the user interacts with the company website or through ads. I also worked with the team to setup the email drip campaigns based on the business logic.

Next I took up getting a marketing automation tool into our system in order to send automated comms and other tasks. I researched the set of available tools and finally we decided to work with clever tap. I led the clever tap project - taking lead on integrations, workflows and reporting. I setup triggers, events and what and how data should be captured. I created a process document to track the progress of the whole project. I setup integrations, created events through API push, created reporting structures to be created on CT (which were then moved to metabase). I created email journeys, based on the triggers - integrating emails and WhatsApp messages. This activity also required state change data push from a particular db, so, I shared the requirements to the tech team to configure this.
I also manage the automation tool called n8n. this tool manages workflows that get our data from websites and other platforms into our database. I work on the tool to do WhatsApp shootouts too for our webinars. I understood the workflows setup and modified them to suit the business needs as and when required. I also worked with fireflink where I managed a guy to create automation test scripts for our website forms. Basically, we created an automation that fills the form on our website iterating through all combinations of values on the form in order to test all branches of the workflow. it sent out a report at the end highlihtin what failed and why. I am also heading the hubspot arm implementation. I manage the whole project right from integrations to business requirement translation. I am also working on the tech part of the integration in order to setup tracking on the incoming leads. i work with a couple of folks who handle the day to day of hubspot while I manage the activities, development, and strategic planning for the project. I work with business heads to translate business requirements into technical requirements that the team can then implement.

I started with the marketing team and considering my previous experience with establishing tracking and maintaining performance of marketing campaigns. I established the UTM tracking system with guidelines. This was also one of my first forays into martech as i worked closely with the tech team to implement the system. I worked with the performance team to understand the parameters that they would want to capture in order to optimize their campaigns. We then built a platform that could be used to create campaigns in the required structure. We could then pull data from GA to understand what worked and what did not. I also setup the base reporting structure for the marketing team, in terms of how campaigns are tracked. I automated spends data mapped to campaign details, which was then used to track how much a lead costed.
I was then moved to a separate team that worked at t the intrersectiion of marketing and sales. I led the efforts for reporting structures - daily/weekly/monthly reports. I also worked with the tech team to ensure no lead leakages between our multiple systems. This involved my working with data at the base level and mapping the systems to tally the leads for the day. This led me to delve deeper into the company architecture and how tech worked here. I worked on the automated report creation and sharing. The firm we were working with to manage our CRM, I put in requests for a few dashboards that would enable us to track real-time lead data - the reports for which we prepared manually by me earlier. We worked on metabase to create a lead funnel that would show the breakdown of lead quality and current calling status of the queue.
I prepared sales and marketing report that highlighted the flow of leads, ROI, and funnel and other such metrics that the sales team then used to track performance of the team. For example, i prepared reports that allowed sales team to understand the sales lifecycle, what is our acquisition cost, what is the trend of uncalled leads per day, trend of lead inflows, source distribution, quality of leads, etc. These reports were then used by both marketing and sales team to course correct. For example, linkedin leads were considered to be premium leads considering the TG to be “educated” but upon analysis, the conversion numbers were similar to

YT leads - which were considered junk, and are cheaper to acquire. This led us to reduce the spends on linkedin while redirecting efforts to remakrting. This was also supported by the fact that there was an uptick in lead inflow once WhatsApp was introduced as a touchpoint, both ion website and in comms journeys. One more instance where data reinforced a business decision was when there was an emphasis on upselling the flagship (Miles US Pathway) to our existing customers, and data confirmed that the student relations team was more successful in converting existing candidates to the flagship.

I also led daily sales calls wherein pan india team would discuss daily progress and blockers. Since i was kind of working with the tech team, i started picking up our CRM as a tool to learn. A couple of months later, I moved into the tech team to explore more technical side of things, rather than just dashboarding. As i had experience with dashboarding in the previous stint with the sales team, i was brought in to setup the reporting module of the CRM. As a stopgap, we picked up metabase, since the teams had already worked with metabase. Since i was also looking at the Whatsapp comms setup for the org, we also built the tracking for it.

I led a team of data analysts who worked with me to setup reports/dashboards on metabase. As per my experience, and after discussing with the business we setup dashboards for the teams to visualize. We run whatsapp on n8n, an automation tool. So we made the whole architecture of the flows in order to track relevant metrics such as delivery rate, read rate, send rate, campaign-wise performance. We were also able to run whatsapp redirect campaigns from meta, and google. Since we captured all incoming messages, we were able to associate campaigns with performance. This was achieved by dynamic whatsapp text based on the url the text is coming from. We were also able to create a tracking of ga_ids on WhatsApp using invisible characters. The tracking helped us in determining the optimal length of a user journey. We started with a reminder/callback journey of 10 messages, but as the setup went live, we could see the delivery rate of the end messages were not exceeding 1%. This observation was then used to revamp the journeys and mke them shorter. The content was also tweaked to be gentler.

We were also able to reduce costs by converting marketing templates to utility and enhancing deliverability too.

This allowed us to decrease SPAM rates, reduce costs and better segment the customers. The marketing team was then able to plan webinar campaigns based on WhatsApp reports. In terms of conversion, we started with initially tracking office visits, which was the primary KPI with all journeys being pushed towards a physical visit. But, as we progressed there was a dip in deliverability of office visits. As an experiment, we added virtual meeting as an option on the website - with google meet as the host. Hypothesis was correct and people started booking virtual meetings. We then tweaked sales processes to also account for google meets. This prompted us to tweak messaging across all channels. We went from “visit office” to “schedule meeting” to make it more non-geographical. This also provided one more touchpoint for the sales person - with various options of physical as well as virtual meeting.

I also setup a automated status checker for the flow. I worked with fireflink to use their automation tool to trigger the flows from the website and whatsapp. Also provided them with a testing matrix, based on which there was an autiomated report shared with the relevant stakeholders in terms of which flow worked, which did not. We also setup error handlers throughout the system. There are alerts setup for all flows to individual mobiles - if there is a template block, payment duee and other such issues.

I am the first person to try out any new tools that we might need to implement from the tech team. The B2B business wanted to work wih thubspot as their CRM. Since no one in the organisation had experience with hubspot, i was asked to pick it up and help setup the system for the team. I worked with the business team to understand their requirements - they had contacts that were related to events and companies and societies. I setup object type relations on hubspot in order for the business to be able to fetch all info for the contact. I take care of the forms that are to be created and embedded onto the website. Since i was also working with the analytics team, i took care of how the forms have to be embedded so that we can ensure tracking for the funnel. We built custom thank you pages for each page, which would then be useful in funnel tracking. I also setup workflow automations such as adding leads to lists, journey for leads that have done some activity like attended an event hosted by us, or remove contacts from marketing contact if they fulfilled some criteria. I also setup ringcentral for the organisation, integrating it with hubspot. I setup workflow using ringcenral activity in hubspot for the spoc to track and take followup steps.

I then moved onto manage website development as a program manager. Here, i acted as a bridge between the business and the tech team. The problem statement was to streamline the requirement gathering process and ensuring timely deliverability. There was a lot of friction between the dev team and the business team. There were issues raised regarding the quality, timeline and lack of visibility for the team. I joined the team and understood the current processes. I sat with the team to understand the current process and the pain points. Tihs helped me to build a few practices which streamlined the development process, reducing the escalations and allowing the team to deliver at 2x speed.

I proposed a process wherein each team had a well-defined role to play. Starting off with the business team in providing the requirements, I devised a format which would require them to put in all the required information in order to add a requirement for any new development. This was then shared with the designers who had TATs to share LoFi designs first and then after getting an approval from the business, they were to start developing the HiFi designs. To add to the development speed, I made the modularity of the page design efficient. I asked the designers to create a repository of designs that can be used to quickly churn new pages, with reduced TAT. The designers would create LoFi designs that would mention the type of component that was to be used in the webpage. Due to componentization of the webpage, now they were quickly able to show multiple options to the business team and get quicker approval. IT also reduced the time to HiFi design. This enabled quicker, smooter delivery of design to the develoment team which was essential to quick turnabouf. Once the design was received, I worked with the technical lead to assign efforts to the page. We used JIRA to track weekly sprints where a daily standup was conducted to check in with the team on previous days progress while also planning out the work for the day. I worked with the technical lead to streamline development workflow too. We faced an issue wherein there was a conflict and one of the devs accidentally deleted a code snippet. I introduced a code_Review stage which would essentially be the Team leads responsiibiltiy to manage merges to prod. Everyone would commit their code to a branch at EoD, which were then reviewed by the team lead, before merging all into a final branch, which would be the one deployed to live. This ensured that the team lead was accountable and responsible for the code being pushed to live. This allowed us to maintain the quality of the code, and ensure single point of control. For reporting, I created custom reports on top of JIRA reports that would be shared with the business team, such as, a website nav bar highlighted report that showed a quick view to the CEO regarding development status for all pages and proposed pages for the website. We were also able to track the number of revisions to the design and content throughout the development cycle, allowing me to intervene at relevant points and reduce average design iterations to 2 per page, from like 11 earlier for some designs.

Let me detail out the workflow that was setup in JIRA eventually, after taking into considerations such as business approval, design process, issues/bugs/hotfixes, code merges and deployment.

this workflow is for website development. starting off the first stage is business requirement doc, which is shared by the business team. once the doc is shared, the requirement is added tot he jira board and assinged to the designer. the page, which was created as a task, is not in lofi design phase. once design is competed, it moves to business approval phase - with an email getting triggered to the business team. once, approved it then moves to hifi design phase. if rejected, it moves back to the lofi design phase, with an email getting triggered to the designer who will rework and submit. once the hifi design is completed, it then again goes to business approval, alongwith content for final design approval. Once approved, it gets pushed to development. Here, once the task arrives, as soon as a developer is assigned, it created a github branch, with the naming convention as that of the ticket. ticket types were divided into new page development, enhancement, or hotfix. each had differnet timeline ranges associated with each in order to give a better estimate of timelines to the business team. once the developer is finsihed with the ticket, they move it to the review stage. this triggers an email to the team lead who will then review the code, and merge it to the "code_review" branch. if the merge is successful, then the ticket is moved to the QC and the QC team receives an email. if the build fails, then the ticket is reverted back to the developer with an email highlighting the issues found during the merge. there was linter introduced to ensure all code compliance issues are taken care of. once the code merge is successful, the QC team then reviews the figma file and a checjkist provided to them to ensure the deliverable matches the business and tech standards. once it gets approved by the QC team, the UAT is then shared with the business team who will then review the page in case there were some misses. in both cases, if rejected - an email is delivered to the developer and the team lead highlighting the issues stated on the ticket.i was also a part of the review process since i was handling the form and tracking for the organisation. i was involved in the page design and also performed QC before delivering the page to the business. this check involved tag manager, analytics, form submissions, campaign tagging, amongst other hygiene checks.

Jira Sample Process Doc

The design elements of the page lied with the business but the functional aspects were taken care of by me. I worked with the business team to work on form placements. During one of the designs, there were no forms on several pages that were to be developed. Based on common sense and data from clarity, we were able to show that there is over 60% drop in the users moving to second fold. So, there has to be a a form or a CTA present in the first fold. While this was not possible in each page, we integrated forms on most pages in the first fold. One other major data-backed change in direction we made in development was when we presented the data that we have over 90% users from mobile devices and rest are split between desktop and tablets, we changed the creative direction towards a mobilefirst design. We also made changes to the website to increase our page speed.

I also worked with the analytics team to rework the GA/GTM setup of the website. As part of website efforts we tracked the client_id, session_count and session_id for users that we could capture. This data was then sent back to GA, as part of conversion metrics and measurement protocol. Based on this setup, we ran a campaign where we targeted the users who registered greater than 5 visits to our website. The sales team were then given these leads to be called. The hypothesis was that these leads would have a higher conversion than the usual leads. They had a higher conversion rate than the leads who have just filled the forms - possibly through other channels.

I then moved towards platform product manager kind of a role. I built the blogs platform and architecture with the marketing team, working with the development team. I worked with the marketing team to understand the requirements. I sat with the designers to understand their designs before pushing it to development. The build was in two parts, admin panel for the internal users to upload blogs and website. We took inspiration from hubspot blogs to design the current setup of blogs. The admin panel was created from django, incorporating all elements the team would require to upload the blogs. I designed the admin panel, modulariaing the structure and enabling the seo team. The admin panel tied up with the front end generated each component of the blog dynamically. All aspects of the page could be controlled by the admin panel: title, banners, meta data, content and ads. This enabled the team to quickly scale up their seo content efforts, driving up our traffic by about 10%. There were a few challenges along the way. The CEO asked us to change the URL structure, which would have led to a development effort. But, instead we had to land on bad code structure to deliver the product within the timeline with all requested features. I also oversaw the migration of old blogs to new blogs: took care of internal link replacements, redirects on DNS level and other internal routing to ensure there is no significant dip in traffic because of the transition to a new domain. We were able to successfully transition to the new website with a minimal drop in traffic - 5%. Currently, there is 3x traffic on the blogs with 10x impressions as compared to the point in thime when blogs was launched. There was a dip in blog traffic, when we started to build the CMS. We arrested the decline, and then got it to 3x traffic.

One more thing i designed was a Scheduling micro service for the organisation. This service essentially became the base for all meeting-based services that Miles would provide. This service could also be used as a standalone single page application, which was also connected to our database. This started off as a micro-service to be integrated with other applications, so we worked on schema and APIs first. Then the requirement was updated so that we can add information into the admin panel to run the service. The admin panel was pretty detailed, There is a provision to add domain, category, spoc, available slots, reminder frequency, email setup, and role-based access based on login. There were also reports that showed available slots for each SPOC, bookings done alognwith their recordings and status for each spoc. The filters were variable and optional, including domain, vertical and category name. One other feature was the marketing team could generate a category/SPOC level link which they can use to promote a category or a specific individual. This was built entirely on nextJS.

Then, i built a couple of CMSs for the marketing team to enable them. First is a learn CMS that was built to be a information repository for mileseducation website. Here, the team would upload topics and content relevant to our products, thereby increasing our SEO rankings. The design of the page was shared by the seo team based on which i created the schema and the admin panel. I gave the option of a fully customizable page content, meta data which the team can customize and use as per their marketing campaigns. The website also now has dynamic sitemaps. This CMS was built on django which wsa then integrated with nextJS.

The other CMS that i built was an event CMS which is used by the marketing team to create and manage events that are on the live website through an admin panel. The event creation was also linked to the CRM. Each webinar that was created in the CMS was also populated back into the CRM, The lead form on the page was then linked to the webinar and all the leads filling the form were then registered to the webinar. The CMS also gave the control of the home page for events to the marketing team allowing them to pin any webinar they wanted to prioritize. The CMS was also integrated with zoom which allowed it to select all available zoom accounts and then give the option of accounts to the marketing team to select the account in which they wanted to hoist the webinar. There were enhancements made to the CRM, to show reports regarding the webinar such as registrations per source, confirmed by the sales team, live attendance count, etc. I used strapi to build this on, which was then integrated with nextJS as our website if built on nextJS.

I take care of any requests that come in from the business team and the marketing team. These are open ended requests such as “we are launching a new business for which we have to start collecting leads”. I then figure out the next steps wrt business, tech and marketing as to what each team would need in order to run their day-to-day. I setup the form capture, tracking, lead flow and reporting for the process. This involves understanding the business requirements and determining what information has to be captured, how it has to be captured, where should it be stored, how it should be stored, who should have access to it, how they should have access to it, what to track, how to track, how to store the tracking information, what to display in rports, how to display reports. This requires me to discuss with the business what is the expected outcome of the activity, hypothesize the outputs, run experiments and verify what matches and what does not. I sit with the developers to understand what is being built, what is the logic, how is the code optimisation been done, what are the APIs being built, what are the logging and tracking mechanisms for the product/platform that i am building.

One of my major projects was rebuilding the entire marketing comms architecture to support user journeys across apps throughout the ecosystem. I spoke with all relevant stakeholders - brand, acads, marketing, lms, growth to understand their requirements in order to build the new architecture. Since i am fluent with n8n and we had a crunch in terms of developer bandwidth, i decided to build the system in n8n using netcore, CRM and WATI. this can then be replaced piece by piece with coded systems. The entire architecture is modular and enabled respective teams to use just the piece they wanted without utizling extra resources. For example, if the LMS team wants to send out a communication through email when the user logged in, but not Watsapp - they can. Vice-versa, if the growth team wants to send both, they can.
This was an event-based architecture which enabled any system to call the service with a defined event and the workflow would proceed as directed. Netcore became my event-lake (storing all information regarding user engagement and activities, enabling hypersegmentation) and is used to send out email. WATI was the WhatsApp aggregator, enabling us to send out messages and continue conversations with users. It also gave the ability to build chatbots, which were then integrated into the website. I added invisible character tracking to ensure all marketing efforts are properly tracked. The flow of info from the chatbot was to this marcomm architecture also. n8n was the orchestrator, enabling me to stitch everything together. I used subworkflows, custom settings, workers when load was high, error logging and retries for all critical workflows. I also setup dashboards to monitor the performance of the system - logging sends/read/delivers/blocks, mapping template performance enabling teams to understand what worked, cost and ROI visibility.
II also built a frontend where the marketing team could now run this entire thing themselves - fetch templates, update templates, define events, map events to comms, track performance. This reduced depdendecy on tech by a significant portion allowing tech to work on bigger problem statements. The next version of this was started using pub/sub, kafka, and google cloud - but was never deployed to prod.

I taught n8n to a lot of people in the org, took sessions, resolved doubts. As a self learner, i was also fluent with enough devops to fix things in my developments.

I also was the AI person for the organisation. I enhanced the marcomm stack to add AI generated content to be sent out based on users intent and past interactions. The chatbot on the website was gen AI enabled and based on the knowledge base it could respond to user queries. The next gen AI initiative was to help marketing team sort out the collaterals. I worked with the team to enable them to quickly generate and reiterate on marketing collaterals - like videos, news, images - reducing go live time significantly. All these are now set as process workflows with a fronend where the user can just go in, pick the tool they want to use and generate the required.
One workflow uses rss feeds to fetch news content for relevant topics, maps it against db to remove duplicates, generates new supportive content for the news, pushes it to push notification queue for app to pick up.

Next, video generation is tough. Before veo3, there was no audio+video present. So we used two different tools stringed together with workflows. Lyria and hunyan were chosen. Hosted the llm on our local server, exposed api end points for n8n workflows. Based on this prompt, a LLM plans how to execute this - understand prompt, figure storyline, write script, breakdown into 10 sec length snippets, generate audio, generate video, wait for them to finish generation, stitch them all together.

Veo3, however amazing only gives you 8 secs of video. For our content, we needed longer videos. So, i built a workflow that takes in user input regarding what they want. For example, give me a video of an rollerskater through the city for 40secs. Based on this prompt, a LLM plans how to execute this - understand prompt, figure storyline, write script, breakdown into 8 sec length snippets, wait for them to finish generation, stitch them all. Once figured the tasks were delegated to sub workflows, reducing resource locking.

I also worked with the data team to enable quicker reporting using AI. developed nlp2sql workflows integrated with a frontend to query in natural language. The backend was powered by a n8n workflow where the user queries were posted through webhook. This was then connected to a sql agent that understood the query, had context of schema, fetched relevant data by constructing the sql, and displayed it as the user asked. It supported charts, tables and counts.

Apart from Miles education work, i myself have started tinkering with products with AI.
One of the first challenges that i took was to build a full stack ecommerce app that used GenAI in some part. This led to an idea of merekapade, which is an ecommerce platform that uses genAI to generat tshirt designs based on users answers. Details below:

Based on the project documentation, here's a detailed overview of the **Custom T-Shirt Hub** (branded as "MereKapade"):

## Project Vision

The Custom T-Shirt Hub is a **personalized t-shirt design and ordering platform** that enables users to create custom apparel through a dynamic, AI-assisted design flow with editable previews. It's built for manual fulfillment today but designed with a modular architecture to enable automated printing and shipping in the future.

## Core Problem & Solution

Problem: Users lack an intuitive platform to easily design, personalize, and order printed t-shirts with a smooth design experience and reliable fulfillment.

Solution: An AI-powered design platform that guides users through a theme-based questionnaire to generate personalized designs, which they can then customize using a visual editor before ordering.

## Key Features & User Journey

### 1. **Landing & Inspiration**
- Showcases pre-designed t-shirt templates
- T-shirt color preview functionality (black, white, grey, etc.)
- Inspirational gallery to spark creativity

### 2. **AI-Powered Design Flow**
- **Theme Selection**: ~15 predefined themes presented as cards
- **Question Flow**: Step-by-step questionnaire (one question at a time with progress bar)
- **AI Generation**: Context-relevant questions generate personalized designs via AI/API
- **Confirmation**: Review responses before proceeding to design

### 3. **Visual Design Editor**
- Built with `fabric.js` for advanced customization
- Edit AI-generated designs with text, shapes, and image uploads
- Question response sidebar for reference during editing
- Stores both original AI-generated and final user-edited images

### 4. **E-commerce & Fulfillment**
- Shopping cart with product management
- Shipping details collection (phone, address, pin code)
- Payment integration (PhonePe primary, Razorpay/Stripe fallback)
- Order tracking (pending → printing → shipped → delivered)
- Manual fulfillment workflow for admins

## Technical Architecture

### **Frontend Stack**
- **Vite + React + TypeScript** for the main application
- **Tailwind CSS** for styling with **shadcn/ui** components
- **Fabric.js** for the design canvas editor

### **Backend & Data**
- **Supabase** for authentication, database, and storage
- **Supabase Edge Functions** (Deno) for serverless backend logic
- **OpenAI DALL-E 3** for AI image generation

### **Infrastructure**
- **Vercel** for deployment and hosting
- **Modular architecture** allowing easy swapping of services (payment processors, design engines, etc.)

## Current Implementation Status

### ✅ **Completed Features**
- User authentication (email/password, OAuth with Google/Facebook)
- Theme selection and question flow
- AI design generation with OpenAI integration
- Visual design editor with fabric.js
- Design saving and user dashboard
- Basic e-commerce structure

### 🚧 **Partially Implemented**
- Checkout flow (missing payment integration completion)
- Order management system
- Admin dashboard

### ❌ **Planned Features**
- Complete payment gateway integration
- Order fulfillment tracking
- Design sharing capabilities
- Advanced admin panel
- Social media integration

## Success Metrics

The project tracks:
- **Design completion rate**: % of users who complete a design
- **Conversion rate**: % design → order conversions
- **Fulfillment efficiency**: Manual fulfillment turnaround time
- **Customer satisfaction**: Order feedback (CSAT)

## Business Model

- **Current**: Manual fulfillment with admin order management
- **Future**: Automated printing and shipping integration
- **Revenue**: Custom t-shirt sales with premium quality materials
- **Target Market**: Users seeking personalized apparel with unique AI-generated designs

The project represents a modern approach to custom apparel, combining AI creativity with user customization in an intuitive, mobile-responsive platform designed for the Indian market (INR currency, PhonePe payments).

Next, i started talking to a nearby cafe owner who expreseed interest in developing a community around the cafe. We spoke and developed ‘mythirdplace’, details below:

# My Third Place - Project Overview

My Third Place is a community-driven platform designed to combat urban loneliness by helping people in metro cities discover and connect with local communities and events. The name refers to the sociological concept of "third places" - essential social spaces beyond home (first place) and work (second place) where communities naturally form and thrive.

## 🎯 Core Mission
Help people discover meaningful communities and participate in local events, reducing isolation in urban environments through authentic social connections.

## 🏗️ What It Does Today

### **Community Discovery & Engagement**
- **Browse Communities**: Mobile-first discovery interface with Material UI navigation
- **Join Communities**: Free membership with instant feedback and optimistic UI updates
- **Community Details**: View member counts, events, discussions, and location information

### **Event Management**
- **Event Discovery**: Browse events across all communities with search and filtering
- **Event Registration**: Support for both free and paid events
- **Payment Integration**: Cashfree payment gateway for paid events (in progress)
- **Capacity Tracking**: Real-time attendee counts and registration status

### **Discussion System**
- **Community Discussions**: Admin-created, time-bound discussion threads
- **Member-Only Commenting**: Discussion participation gated to community members
- **Real-time Updates**: Live comment updates with optimistic rendering

### **User Features**
- **Google OAuth Authentication**: Seamless sign-in with session persistence
- **Personal Dashboard**: "My Events", "My Communities" overview with quick stats
- **Profile Management**: User profiles with notification preferences
- **Referral System**: Code-based referral flow with tracking and analytics

### **Advanced Systems**
- **Email Automation**: Welcome email system via Supabase Edge Functions + Resend

- **Webhook Infrastructure**: Event dispatching system for external integrations (n8n compatible)
- **Activity Logging**: Comprehensive user activity tracking (selectively enabled)

## 🛠️ Technical Architecture

### **Frontend Stack**
- **React 18** with TypeScript for type safety
- **Tailwind CSS** + Material UI patterns for mobile-first responsive design
- **Vite** for fast development and optimized builds
- **React Query** for efficient data fetching and caching

### **Backend Infrastructure**
- **Supabase** as the complete backend solution:
  - PostgreSQL database with 14 tables
  - Row Level Security (RLS) policies for data protection
  - Authentication with Google OAuth
  - Edge Functions for serverless logic
  - Real-time subscriptions

### **Key Integrations**
- **Payment Processing**: Cashfree gateway for event payments
- **Email Service**: Resend API for transactional emails
- **External Webhooks**: n8n-compatible webhook system for automation

## 🎨 Design Philosophy

### **Mobile-First Approach**
- Responsive design optimized for mobile devices
- Bottom navigation for easy thumb navigation
- Touch-friendly interactions and button sizing

### **Security & Privacy**
- Comprehensive RLS policies on all database tables
- Role-based access control (User, Host, Admin)
- Privacy by default (e.g., participant lists are backend-only)

### **User Experience Principles**
- **Optimistic UI**: Instant feedback for all user actions
- **Progressive Enhancement**: Works without authentication for discovery
- **Graceful Degradation**: Comprehensive error handling and loading states
- **Accessibility**: WCAG 2.1 AA compliance target

## 🚀 Current Development Status

### **Completed Phases**
1. ✅ **Core Architecture**: Database schema, authentication, mobile-first layout
2. ✅ **User Features**: Community discovery, event registration, discussions, referrals
3. ✅ **UX Polish**: Optimistic updates, responsive design, navigation optimization
4. ✅ **Infrastructure**: Email system, webhook framework, activity logging

- 🟡 **Payment Integration**: Edge functions implemented, signature verification pending
- 🟡 **Webhook System**: Infrastructure complete, admin UI pending in separate Admin Panel

## 🧪 Quality Assurance

### **Comprehensive Testing**
- **230+ passing tests** across unit, integration, and RLS policy validation
- **Mandatory test validation** before commits via `node src/scripts/test.js`
- **CI/CD Integration** with GitHub Actions and Vercel deployments

### **Test Coverage Areas**
- Authentication flows and protected routes
- Database integrity and RLS enforcement
- Component logic and user interactions
- API endpoints and error handling

## 🌟 Unique Value Propositions

1. **Community-Centric**: Focus on authentic local communities rather than generic social networking
2. **Event-Driven Engagement**: Events as the primary mechanism for community participation
3. **Privacy-First**: Strong data protection with granular access controls
4. **Mobile-Optimized**: Built specifically for mobile-first urban users
5. **Extensible Architecture**: Webhook system enables integration with external tools and automation platforms

## 🎯 Target Outcome
Create vibrant local communities where people can discover their "third place" - that essential social space where they belong, contribute, and form meaningful connections beyond their home and workplace.

The platform aims to transform how people discover and engage with their local communities, making it easier to find like-minded individuals and participate in activities that enrich their social lives and reduce urban isolation.

Mythird place also has an admin panel that has most of the frontend pieces integrated into it. Here are the details of the project:

Based on the project documentation, here's a detailed overview of **MyThirdPlace Admin Panel**:

## Project Overview

MyThirdPlace Admin Panel is a comprehensive administrative dashboard for managing a community platform. It's built as a modern web application using React, TypeScript, and Supabase, designed to provide administrators with complete control over their community ecosystem.

## Core Purpose & Objectives

The platform aims to be a **"third place"** - a social environment separate from home and work where people can gather, connect, and build community. The admin panel serves as the central command center for managing this digital community space.

### Key Management Areas:

1. **User Management**
   - Complete user lifecycle management (CRUD operations)
   - Role-based access control (user/admin roles)
   - Ban/unban functionality with audit trails
   - Referral system tracking
   - Bulk operations for efficient user management

2. **Community Management**
   - Create and manage local communities organized by city
   - Community branding with image uploads
   - Member relationship tracking
   - Community analytics and engagement metrics

3. **Event Management**
   - Comprehensive event creation and scheduling
   - Registration and capacity management
   - Payment integration and revenue tracking
   - Event media management
   - Host assignment and event cancellation controls

4. **Discussion & Content Moderation**
   - Community-based discussion monitoring

   - Content flagging and moderation tools
   - Expiration systems for time-sensitive discussions
   - Visibility controls and comment management

5. **Analytics & Business Intelligence**
   - Real-time dashboard with key metrics
   - Growth trend analysis and user engagement tracking
   - Revenue analytics and registration monitoring
   - Interactive data visualizations

## Technical Architecture

### Frontend Stack:
- **React 18 + TypeScript** for type-safe component development
- **Vite** for fast development and optimized builds
- **Tailwind CSS + Shadcn/ui** for professional, accessible UI components
- **React Query** for efficient server state management
- **React Router** for protected admin routing

### Backend & Database:
- **Supabase** as the complete backend solution
- **PostgreSQL** with Row Level Security (RLS) policies
- **18 core database tables** covering all business entities
- **Advanced role and permission system** with granular access control
- **Real-time subscriptions** for live data updates

### Security Features:
- **Role-based authentication** with session management
- **Comprehensive RLS policies** for data protection
- **Audit logging** for all administrative actions
- **Protected routes** and component-level security
- **Granular permission system** with inheritance

## Business Value & Use Cases

### For Platform Administrators:
- **Centralized Control**: Single dashboard for all platform operations
- **Data-Driven Decisions**: Comprehensive analytics and reporting
- **Efficient Moderation**: Streamlined content and user management
- **Revenue Tracking**: Complete financial oversight of events and registrations

### For Community Managers:
- **Community Growth**: Tools to foster engagement and expansion
- **Event Success**: End-to-end event management capabilities

- **User Engagement**: Discussion facilitation and moderation tools
- **Performance Monitoring**: Real-time metrics and trend analysis

## Current Status & Achievements

The project has completed **11 major development phases** and includes:

- **40+ reusable components** with consistent design system
- **9 specialized admin pages** plus public landing pages
- **110+ individual features** implemented and tested
- **Enterprise-grade security** with advanced role management
- **Mobile-responsive design** with progressive enhancement
- **Performance optimizations** including code splitting and lazy loading

## Future Roadmap

Planned enhancements include:
- **Mobile app development** (PWA implementation)
- **Advanced API integrations** for third-party services
- **Enhanced analytics** with custom reporting
- **Automated moderation** using AI/ML capabilities
- **Multi-language support** for global communities

The platform represents a complete solution for managing modern digital communities, combining the social aspects of community building with the administrative tools needed to scale and maintain a thriving platform.

Now, in my new role at tribeca, i am their first tech hire. So essentially, i am gauging across the organisation where all can i integrate tech optimisations. Started off with legal, in talks withmultiple vendors to land a contract management and legal research tools. Did research, prepared metrics for evaluation, took discussion with vendors, now on to next steps.
Next, worked with Hr to understand their pain points. Suggested tools to inculcate AI, speaking to HRMS vendor for API access to sort data and prep dashboards for leadership.
Working on dahbaords for CXO visibility. Based on a downloadable report, developed dashboards on streamlit. Speaking to salesforce internal teams to understand the data architecture. Planning to move all reporting as API based data fetch from salesforce.

One major thing i am working on there is a RAG app that can be shared with anyone across the organisation to enable them to build their own personal chatbots or i can make some standard chatobots and allow peopl to chat - hr policy bot, sales process bot, regulations bot, compliance bot etc. below is the detail for the app:

# 📄 PRD — RAG Framework App (Final with Repo Scaffold)

## 1. Problem

Users need a **secure, session-aware RAG platform** where they can:

* Log in with Google/Microsoft.
* Create and manage **agents** configured with prompts, models, and documents.
* Chat with selected agents in a conversational UI with **session memory**.
* Upload/manage documents (with versioning and summaries), view them globally, and see which agents they're linked to.
* Receive **confidence-scored, cited answers**, with fail-safe when low confidence.
* Ensure a **test-driven development workflow**, structured logging, and integration with Linear for tracking.

## 2. Goals

1. **Auth & Accounts**: Support Google and Microsoft login.
2. **Agent Management**: Create/edit/delete agents with configurable prompts, models, and docs.
3. **Document Management**: Global pool of docs, visible in a “Documents” tab, with summaries, versioning, and links to agents.
4. **Chat Experience**: Session-based conversations with context memory + RAG retrieval, confidence scoring, and citations.

5. **Profile & Dashboard**: Show user info and usage stats (#agents, #docs).
6. **Infrastructure**: Supabase (DB/Auth/Storage), Pinecone (embeddings), OpenAI (embeddings + generation), Linear (tracking).
7. **Engineering Workflow**: Test-driven development (TDD) enforced, with logs for queries/answers/evals/tokens/latency.

## 3. Functional Requirements

### 3.1 Authentication

* Login with **Google or Microsoft** via Supabase Auth.
* JWT-based session persistence.

### 3.2 Agents

* Users can create agents with:

  * Name, description.
  * System prompt.
  * Attached documents (upload new or select from global pool).
  * Embedding model (default: `text-embedding-3-small`).
  * Generation model (default: `gpt-4o-mini`).
  * Last K messages (default 6).
  * Similarity threshold (default 0.75).
  * Fail-safe toggle (on/off).
* Users can remove a doc from an agent's knowledge base.
* Docs uploaded here are also added to the global pool.

### 3.3 Documents

* **Global "Documents" tab**: lists all docs uploaded by the user.
* Each doc shows:

  * Name.
  * Auto-generated quick summary.
  * Version history.
  * Agents it's linked to.
* Users can:

  * Upload new docs.
  * Link/unlink docs ↔ agents.
  * Delete a doc (removes all links + vectors + marks tombstone).

### 3.4 Chat

* Chat window requires selecting an agent.
* Each chat session:

  * Maintains **session memory** (rolling summary + last K turns).
  * Pulls relevant context chunks from Pinecone.
* Each answer includes:

  * Response text.
  * **Confidence footer** (displayed if <80%).
  * Citations (doc name + page).
  * Fail-safe response if <50%.
* Queries, answers, evals, tokens, and latency are logged.

### 3.5 Profile

* Basic user info (name, email, org).
* Stats: #agents, #docs, #sessions.

## 4. Non-Functional Requirements

* **Test-Driven Development (TDD)**:

  1. Understand feature/problem.
  2. Define success criteria.
  3. Write functional + technical tests.
  4. Develop feature.
  5. Run tests → fail → debug; pass → commit.
  6. Update Linear issue with results.
* Latency target: <3s p50 per query.
* Scalability: 1000 docs/user, 50 agents/user.
* Security:

  * Supabase RLS policies per user.
  * Signed URLs for docs.
  * Pinecone vectors namespaced `user_id:agent_id`.

## 5. System Design

Frontend (Next.js/Vercel)

* Tabs: Agents, Documents, Chat, Profile.
* State: React Query + Context.

Backend (Supabase + Edge Functions)

* Auth: Google/Microsoft.
* DB: Supabase Postgres (agents, docs, sessions, logs).
* Storage: Supabase buckets for PDFs.
* Edge Functions:

  * `ingest_doc`: parse, chunk, embed, upsert.
  * `create_agent`: save agent + doc mappings.
  * `query_chat`: assemble prompt, run OpenAI call, eval, log.
  * `unlink_doc`: remove agent↔doc mapping.
  * `delete_doc`: purge vectors + tombstone record.

Vector DB (Pinecone)

* Index: cosine similarity.
* Namespace: `user_id:agent_id`.

Models (OpenAI)

* Embeddings: `text-embedding-3-small`.
* Generation: `gpt-4o-mini`.

Tracking (Linear MCP)

* All epics/issues in Linear.
* One PR per issue.
* Test results stored as artifacts.

## 6. Data Models

users

* id, email, name, org, created_at

agents

* id, user_id, name, description, system_prompt, embed_model, gen_model, last_k, sim_threshold, failsafe_enabled, created_at

documents

* id, user_id, name, summary, latest_version, created_at

document_versions

* id, document_id, version_no, checksum, created_at

document_chunks

* id, document_id, version_id, page_start, page_end, chunk_index, token_count, created_at

agent_docs (join table)

* id, agent_id, document_id, version_id

sessions

* id, user_id, agent_id, title, created_at

messages

* id, session_id, role (user/assistant), content, tokens_in, tokens_out, created_at

session_summaries

* id, session_id, summary_text, updated_at

queries

* id, session_id, user_query, latency_ms, token_in, token_out, created_at

answers

* id, query_id, text, confidence, fail_safe_triggered, created_at

evals

* id, query_id, heuristic_max_sim, heuristic_mean_sim, pct_above_thresh, llm_faithfulness, aggregate_confidence, created_at

citations

* id, answer_id, doc_id, version_id, page_start, page_end, chunk_index, sim_score

---

## 7. Workflows

### 7.1 Agent Creation

1. User clicks **New Agent**.
2. Configures prompt/models/thresholds.
3. Attaches docs (upload new → adds to pool, or select existing).
4. Agent saved + Pinecone upserts.

### 7.2 Document Management

* **Documents Tab** lists all user docs.
* User actions:

  * Upload new doc (goes to pool, linkable to agents).
  * View/edit doc summary + versions.
  * Link/unlink agents.
  * Delete doc (purge all).

### 7.3 Chat

1. User selects an agent → start session.
2. User sends query.
3. Backend: fetch session summary + last K → retrieve Pinecone context → call LLM.
4. Run evals → aggregate confidence.
5. Store logs.
6. Display: answer + citations + confidence footer.

## 8. QA / TDD Protocol

* Every feature begins with tests.
* Store tests under `/tests` (mirrors `/src`).
* Types:

  * Unit: chunking, eval aggregation.

  * Integration: agent creation, doc ingestion.
  * E2E: chat with agent, fail-safe flow.
* CI runs all tests before merging.
* Test results stored and viewable.

## 9. Repo Folder Scaffold

```
/ (root)
├── /src
│   ├── /frontend       # Next.js components, pages, hooks
│   ├── /backend        # Supabase edge functions
│   ├── /agents         # Agent config + orchestration logic
│   ├── /docs_ingest    # PDF parsing, chunking, embedding
│   ├── /chat           # Session memory, prompt assembly
│   ├── /evals          # Eval heuristics + judge logic
│   └── /utils          # Shared helpers
│
├── /tests
│   ├── /unit
│   ├── /integration
│   └── /e2e
│
├── /scripts
│   └── /linear         # Linear MCP setup/automation scripts
│
├── /docs
│   ├── PRD.md          # This PRD
│   ├── ARCHITECTURE.md  # High-level diagrams
│   └── SETUP.md        # Dev setup instructions
│
├── /config
│   ├── supabase/       # DB migrations, schema
│   ├── vercel/         # Vercel configs
│   └── .env.example    # Env template
│
├── /infra
│   ├── docker/         # Dockerfiles, compose
│   └── deploy/         # CI/CD configs
│
└── package.json
```

Rules:

* All Linear scripts under `/scripts/linear`.
* All docs under `/docs`.
* All tests mirror `/src`.
* No stray files in root — everything must be slubbed into its folder.

## 10. Metrics

* \#agents/user.
* \#docs uploaded.
* Avg confidence per query.
* % queries <0.5 (fail-safe).
* Latency p50/p95.
* Token usage.

## 11. Risks

* Large doc versions → storage quota.
* Eval latency → longer response time.
* Token drift → trimming last K required.
* Multi-tenant doc-sharing not yet designed.

✅ This PRD now covers:

* Google/Microsoft login
* Global Docs tab with agent mappings
* Session-aware chat with memory + confidence
* TDD workflow baked in
* Repo scaffold for maintainability

One other thing that i developed at Tribeca is a CoS bot. The bot essentially functioned as an EA that would monitor conversations in pariuctlar groups and infer tasks that would then be added to a task list. This was AI enabled allowing the users to natural language querying to provide answers regarding the tasks statuses. Here is a detailed documentation of the bot:

Alright, here's the **proper, end-to-end documentation** for your **CoS (Chief-of-Staff) Bot**.
This is written so you can hand it to:

- a future engineer,
- a stakeholder like Kalpesh,
- or your future self when you've forgotten why half of this exists.

No fluff, no marketing nonsense. Just how the thing actually works.

CoS Bot – System Documentation

WhatsApp → Tasks → Intelligence → Natural Language Control

---

1. Overview

The **CoS Bot** is an AI-powered operational assistant that:

- Listens to WhatsApp group conversations
- Extracts actionable tasks from unstructured messages
- Stores tasks in a structured database
- Embeds tasks for semantic retrieval
- Allows leadership to **query, review, and update tasks in natural language**
- Acts as a lightweight Chief-of-Staff layer over daily execution

In short:

> *People talk casually. The system remembers, structures, and answers later.*

---

2. High-Level Architecture

WhatsApp Groups
↓
Periskope (Webhook)
↓

n8n (Message ingestion + orchestration)
↓
Supabase (Postgres + pgvector)
↓
Edge Functions (Search / Fetch / Update)
↓
ChatGPT (Natural Language Interface)

Each layer has **one job only**. No layer is “smart” unless it absolutely must be.

---

3. Core Principles

1. **WhatsApp is the source of truth**
   - No forced formats
   - No “please fill this template” nonsense
2. **Tasks are immutable facts**
   - Who said what
   - When
   - In which group
   - Confidence score attached
3. **Search is semantic, not keyword-based**
   - “Facade work”, “cladding”, “elevation” → same intent
4. **LLMs reason, databases decide**
   - LLM interprets intent
   - Database enforces structure and truth

---

4. Message Ingestion (WhatsApp → n8n)

4.1 Source

- WhatsApp groups connected via **Periskope**
- Bot number is added to selected groups

4.2 Webhook Payload

Periskope sends message events including:

- `chat_jid` (group ID)

- `sender_phone`
- `message_uid`
- `text`
- `mentioned_ids`
- timestamps

4.3 Filtering Logic (n8n)

In n8n:

- Only groups matching naming rules (e.g. “Tribeca”) are processed
- Media-only messages are ignored
- Duplicate messages are deduped via `message_uid`

---

5. Data Model (Supabase)

5.1 Core Tables

`chats`

Stores WhatsApp group metadata

- `chat_jid`
- `name`
- `description`

`people`

Maps WhatsApp users to people

- `wa_jid`
- `phone`
- `email` (optional)

`messages`

Immutable record of WhatsApp messages

- `message_uid` (unique)
- `chat_id`
- `sender_id`

- `text`
- `timestamp`

`tasks`

Structured tasks extracted from messages

- `title`
- `description`
- `assigner_id`
- `assignee_id`
- `due_date`
- `priority`
- `status`
- `confidence`
- `source_message_id`

`task_vectors`

Vector embeddings for semantic search

- `task_id`
- `content`
- `embedding (pgvector)`
- `metadata`

---

6. Task Extraction (LLM Parsing)

6.1 Trigger

Every new message enters an LLM parsing flow if:

- Message contains intent markers (`@mentions`, “do this”, “please”, “task”, deadlines)
- Or passes confidence threshold via heuristic rules

6.2 Output Schema

LLM **must** output:

output.tasks[]

Each task includes:

- title
- description
- assignee / assigner (IDs + WhatsApp JIDs)
- due date (ISO)
- priority
- confidence score
- raw_message (for traceability)

No task is created without a confidence score.

---

7. Embedding & Vector Search

7.1 Embedding Strategy

- Text embedded includes:
  - title
  - description
  - due date phrasing (e.g. “Due Oct 25”)
- Model: `text-embedding-3-small` (1536 dims)

7.2 Storage

Embeddings stored in `task_vectors` with:

- `task_id` foreign key
- Metadata (chat, assignee, priority)

7.3 Why embeddings matter

This allows queries like:

- “What’s pending with the facade?”
- “Anything due this week?”
- “Tasks assigned to Tarun”

without brittle filters.

---

8. Edge Functions (Supabase)

All functions require:

Authorization: Bearer <SUPABASE_SERVICE_ROLE_KEY>

8.1 `tasks-search-vec`

Semantic search over tasks.

Input

```
{
  "q": "due this week",
  "limit": 5,
  "threshold": 0.65
}
```

Output

```
{
  "items": [
    { "task_id": 18, "similarity": 0.82 }
  ]
}
```

8.2 `tasks-get`

Fetch full task details.

Input

```
{ "id": 18 }
```

Output

- Task
- Assignee & assigner details
- Chat context
- Source message

8.3 `tasks-update`

Modify task state.

Supports:

- status
- due_date
- priority
- assignee / assigner

Automatically re-queues embedding if text changes.

9. ChatGPT Integration (Natural Language Layer)

9.1 Role

ChatGPT acts as:

- Query interpreter
- Action orchestrator
- Natural language summarizer

It **never** stores state itself.

9.2 Query Flow

Example:
User: “What’s due this week?”

1. GPT → `tasks-search-vec`
2. Receives task IDs
3. GPT → `tasks-get` for top results
4. GPT → Natural language response

9.3 Update Flow

User: “Mark the facade task as done”

1. GPT finds relevant task via search
2. GPT calls `tasks-update`
3. Confirms action in plain English

---

10. Failure Modes & Known Limitations

10.1 Semantic mismatch

- “This week” may not match if due dates aren’t embedded
- Solution: hybrid date filters (planned next)

10.2 Over-extraction

- LLM may hallucinate tasks from casual messages
- Controlled via confidence thresholds

10.3 Missing context

- Some WhatsApp instructions rely on implicit knowledge
- Bot errs on *not creating* tasks unless confident

---

11. Security Model

- **No public endpoints**
- Service Role key only
- RLS bypassed intentionally (system-owned data)
- WhatsApp data never leaves backend except summarized

---

12. What This Enables (Strategically)

- Leadership can ask:
  - “What’s stuck?”
  - “Who’s overloaded?”
  - “What changed today?”
- Zero manual reporting

- Zero follow-up chasing
- Institutional memory from chat chaos

This is not a chatbot.
This is **operational memory + decision surface**.

ChatGPT enhanced version of product management task:

Here's a more detailed and technically rich version of your text, incorporating advanced technical terms and product management concepts:

Blogs Platform Development

As a **Platform Product Manager**, I led the end-to-end design and development of a **Blogs Platform** to support content marketing and SEO strategies. This initiative required close collaboration with the **marketing team** to elicit and prioritize business requirements, the **design team** for UI/UX prototyping, and the **engineering team** to manage technical implementation.

Architecture and Features:

- **Two-tier architecture**:

1. **Admin Panel**: Built on **Django Framework**, the panel incorporated a modular architecture with RESTful APIs for backend logic, enabling dynamic content management. The admin panel was designed to support CRUD operations for the following components:

- **Title and headers**

- **Hero banners and media uploads** with optimized storage using **AWS S3**.

- **SEO metadata** (meta titles, descriptions, and keywords).

- **Content sections**, dynamically generated using **rich text editors (like CKEditor)**.

- **Ad placement controls** to configure ad units dynamically.

2. **Frontend Website**: Developed with a **React-based SPA**, the frontend leveraged **server-side rendering (SSR)** and **static site generation (SSG)** techniques for optimal SEO performance and page load times.

Key Outcomes:

- Empowered the SEO team to scale their efforts, driving a 10% increase in organic traffic by enhancing agility in content updates.

- Seamless integration with the legacy CMS and **automated migration scripts** for legacy blog content using **Python-based ETL pipelines**, ensuring minimal manual intervention.

Challenges and Resolutions:

- **URL structure modification**: A significant request from the CEO to alter URL formats mid-project posed a challenge due to tight deadlines. To balance business needs and timelines, we introduced a **middleware layer** to handle URL remapping, avoiding a full redevelopment effort.

- **SEO traffic preservation**: Managed **301 redirects** at the **DNS and application level** to prevent link breakages. Developed scripts to update internal links in bulk, ensuring continuity in SEO rankings.

Scheduling Microservice

I spearheaded the development of a **Scheduling Microservice**, which became the foundational building block for all meeting-based functionalities within the organization.

Technical Approach:

- **Microservice-first design**: The service was built using **FastAPI** for lightweight, high-performance API development. It adhered to the **12-factor app methodology**, ensuring scalability and maintainability.

- **Database Schema Design**: Leveraged **PostgreSQL** for its ACID compliance and schema flexibility. Tables included:

- **Domains** (to group scheduling contexts).

- **SPOCs** (single points of contact).

- **Availability Slots** (with start and end times stored as **timestamp ranges**).
- **Bookings** (linked with user profiles).
- **Notifications** for reminders.

Admin Panel Features:

Built with **React.js** for a responsive UI and integrated with the microservice APIs, the admin panel offered:

- **Dynamic field management**: Configuration for domains, categories, and SPOCs.
- **Role-Based Access Control (RBAC)**: Integrated with **OAuth2** for secure login and granular permission settings.
- **Reports Dashboard**: Real-time data visualizations using **Chart.js**, supporting filters by vertical, category, and domain.
- **Link Generators**: Allowed the marketing team to generate **parameterized URLs** for targeted campaigns.

Outcomes:

- Introduced a reusable, modular service architecture enabling rapid integration with other products.
- Reduced lead time for deploying new meeting-based features by 30%.

Content Management Systems (CMS)

I designed and delivered multiple CMS platforms, each tailored to specific marketing objectives and leveraging **headless CMS principles** for maximum flexibility.

Learn CMS:

- **Purpose**: A centralized **information repository** to boost SEO rankings and improve content discoverability.
- **Technical Design**:
- **Schema Design**: Tables for topics, categories, and metadata structured to optimize content retrieval.
- **Dynamic Sitemaps**: Built-in **automated sitemap generation** using Django, updating whenever new content was published.

- **Rich Text Editor Integration**: Enabled custom content formatting and image uploads.

- **Outcomes**: Delivered a fully customizable admin interface that aligned content with marketing campaign objectives.

Event CMS:

- **Purpose**: Simplify event creation and management while integrating with existing tools and workflows.

- **Key Features**:

- **Webinar Integration**: Direct API integration with Zoom, fetching available accounts and scheduling meetings seamlessly.

- **CRM Integration**: Automated data synchronization between the CMS and CRM. Each webinar was linked to a lead form that fed registration data back into the CRM pipeline.

- **Dynamic Home Page Management**: Allowed marketing teams to pin high-priority events for increased visibility.

- **Reporting Enhancements**:

- Registrations by source.

- Sales confirmations and follow-up statuses.

- Real-time attendance tracking.

Summary of Core Product Management Contributions

- Applied **Agile methodologies**, managing sprints, backlogs, and stakeholder reviews to ensure timely delivery.

- Developed detailed **product requirement documents (PRDs)** and **user stories**, with acceptance criteria aligned to business goals.

- Used tools like **Jira** for project tracking, **Figma** for design reviews, and **Postman** for API testing.

- Conducted **A/B testing** and performance analysis to validate features post-deployment, ensuring alignment with KPIs.

- Delivered scalable and reusable systems leveraging **API-first design**, **microservices architecture**, and **cloud-based integrations (AWS, Zoom APIs)**.

This experience reflects a strong foundation in **product lifecycle management**, technical understanding of platform architecture, and the ability to align product deliverables with strategic business objectives.

Here's a detailed version of your **Miles Education** experience, incorporating your contributions to product and platform development, with relevant metrics:

Product Manager | CTO's Office
Miles Education
December 2023 – Present | Bangalore Urban, Karnataka, India

Project Management

- Spearheaded the design and implementation of end-to-end workflows for website development, enabling modular content management and faster page deployment by 30%.
- Reduced team coordination efforts by 25% through automated status updates and cross-functional task alignment.

Campaign Tracking and Attribution

- Enhanced marketing attribution and tracking, leading to a 12% increase in qualified leads.
- Optimized campaign execution, achieving a 15% boost in ROI through targeted spending.

HubSpot CRM Workflows

- Automated lead qualification and lifecycle management in HubSpot, increasing sales team efficiency by 20%.

Content Management Workflow

- Designed a modular workflow using n8n, reducing content deployment time by 40%.
- Improved website performance with a 10% growth in organic traffic and a 7% increase in session duration.

Customer Communication Automation

- Deployed n8n-based WhatsApp and email automations, reducing funnel drop-offs by 30%.
- Enhanced communication efficiency, achieving a 25% improvement in response times.

Reporting Dashboards

- Built Metabase dashboards to automate performance tracking, cutting reporting turnaround time by 50%.
- Provided actionable insights for faster decision-making.

SEO Publishing Platform

- Developed a publishing platform for the SEO team, enabling trend-aligned content updates.
- Contributed to a 15% increase in inbound traffic by streamlining content workflows.

This version highlights your leadership in developing platforms, driving product innovation, and quantifying the impact of your contributions. It positions you as a tech-savvy and result-oriented Product Manager, aligning with your career goals. Let me know if you'd like any refinements!