Apollo Group
Engineering – Case Study

On a mission to unlock the potential of technology in the world, Apollo is building consumer marketplaces that bring new types of services to consumers. Our first project is Pluto, a marketplace for consumers to hire trained blue-collar service providers. Customers will be able to hop onto our mobile application, select one of the services being offered and request a fully trained service provider within hours.

In order to build the app, our product team has decided that three components are crucial:

- A web-based admin portal
- A customer mobile application
- A service provider mobile application

Each of the above will offer a set of key functionalities required for various stakeholders in the marketplace. As an example, the admin portal will help our corporate team set the right configurations and settings on the platform. As part of our brainstorming sessions, the product team has requested your input on the following questions:

1. In building our product, in your view, which functionalities would be the most important to get right early on? Please provide the rationale behind your answer.

2. What should be the guiding principle that informs our thinking around which functionalities to include in the MVP version of the consumer application?

3. In order to ship the MVP quickly, our product team has identified that we can ship either the consumer application or the service provider application. In other words, due to time constraints, we need to pick one of the two and then work on launching that in the MVP. Which of the two applications would you consider more critical to include in the MVP and why?

4. How would you design the system architecture and choose appropriate technologies to accommodate the platform's expansion from an MVP to serving 50k service providers, 2M customers, and 500 admins/operational staff? In addition to addressing scalability, performance, and efficient workload management, how would you ensure the system's robustness, fault tolerance, high availability, and seamless user experience in the face of increasing demands?

Please note that there are no right or wrong answers. We are most curious to understand your thinking behind the responses and your unique approach toward problem-solving. We also anticipate candidates using AI tools to help answer and we don't see that as a problem as long as independent thinking is incorporated. During the case study, you may reach out to hussain.akbar@apollo-group.io if you have any questions.

In building our product, in your view, which functionalities would be the most important to get right early on? Please provide the rationale behind your answer.

Answer:

Context:

In general, any application has following type of functionalities:

1. Core Business Functionalities (e.g., able to send or receive money)
2. Functionalities that support / enhance core business functionalities (e.g., doing bill payments)
3. Functionalities that make your product attractive (e.g., rewards on making transactions of XXXXXXX amount in month)
4. Functionalities that add convenience to the product (e.g., notifications, alerts, redirections, payment using QR code instead of entering account number etc.)
5. And then we have functionalities that builds customer's trust (e.g., complaint section with good service, maybe a fraud detection system etc.)
6. And others as well but the basic ones I have covered.

(Note: The examples given above are of a banking application)

The statement ***"Which functionalities would be the most important to get right early on"*** is very vague and subjective because **early** doesn't define how early? Is it the early that means what functionalities should be there when the first iteration (first version of MVP) launched? Or when functionalities should be there when our MVP (after multiple iterations) has validated our idea?

The Actual Answer:

In the early stages, the **core business functionalities** should be implemented. And by **"core business functionalities"**, I refer to the functionalities that add some monetary value to the business. And in my opinion following are the core business functionalities given our vision for this application:

- #### User Registration & Authentication (Sign Up and Login)

  Rationale:

  It's all about building a customer base. So, when a customer is registered, only then you can have the **required data to suggest** him the kind of services that he needs and the service providers that can meet his demand. This is a very standard feature, and every application has it. The registration form and user's profile (especially for service providers) will be very critical here and our listing will depend on it.

- #### Post a Blue-Collar Service (Service Provider Application)

  Rationale:

  As the idea of the application revolves around a service that is provided and consumed so to make that happen, a service provider should be able to post a service

that he offers. A challenge here would be the rates and prices as the fixes to blue collar services can be very unpredictable so the rates will be hard to standardize.

- List Blue-Collar Services (Consumer Application)

Rationale:

As a consumer, of course I would want to see the services that are provided by the providers so that I know what are the services for which I can use this app. And even to utilize a service, I need to see the listing so that I can choose the provider that can fulfill my demand.

- Search for a required Blue-Collar Service (Consumer Application)

Rationale:

Let's say that my Power Back Up (UPS and Solar system), stopped working so I should not want to scroll the listing hoping that I would find a fixer that I need. Instead, I would want a search bar where I can just type "UPS and Solar fixers" and it should list me all technicians with this skillset.

- Sort Blue-Collar Services by different filters (Service provider's reputation, no of jobs completed, distance, avg response time)

Rationale:

Now that I have searched for Solar Panel fixer, obviously I wouldn't want to live in this hot weather without a cooling system (fan, AC etc). So, my top priority here would be to see all those providers who can come asap so that I can get it fixed. Therefore, the sorting is very important as it gives the customer a flexibility to prioritize the provider according to his need.

- Request for a Service (Consumer Application)

Rationale:

After sorting, luckily, I found 1 service provider who can come to my home within an hour so now there should be an option that I can use to ask that provider to come and fix my problem. So, I should be able to request a service.

- Cancel a Service Request (Both Provider & Consumer Application)

Rationale:

(Hypothetically speaking) My son was checking the batteries as he's really interested in technology, so he removed the terminals of the batteries and then checked water (battery acid), added some water and then he connected terminals again and it started working. Now, I should be able to cancel the request as my solution is fixed.

- Alerts (for both service provider and consumers)

Rationale:

As a service provider, I want to receive an alert that some consumer wants to avail my service. Also, as a consumer, I want to see alerts that my service provider has cancelled the request or he will be late or he is here so that I can be at home by the time he reaches.

- ## Payments

Rationale:

A payment mechanism (sending and receiving money) should be there. I hired a provider, and he did the job well. Now, I can either pay him cash but then it's hard for the application company to collect payment from provider and not a very good way to track the application's growth.

So, a better way is that I pay using the app so that the application company can deduct it's cut and sends the rest of the amount to the provider.

What should be the guiding principle that informs our thinking around which functionalities to include in the MVP version of the consumer application?

Answer:

The guiding principle is to add all those functionalities which fulfill a complete cycle of our core vision for the product. Here for example, the core vision can be,

"A consumer should be able to avail a required blue-collar service posted by a provider and upon successful provision of the service, the consumer has to pay the provider".

Now, all the functionalities that are mandatory to make that happen should be the part of our MVP.

My Personal Opinion about any new product:

Moreover, every product should have a functionality that uniquely defines it. That functionality basically makes that product stands out from other products of the same nature in the market. So, that functionality should also be the part of MVP because it all starts with that. If there isn't something really unique about a new product (apart from standard NFRs like security, reliability, responsiveness etc), I personally would not want to use it.

For example, Netflix has a lot of series and movies. Amazon Prime has a lot of series and movies but both of these apps don't have all the movies which one would want to watch. Now, right now, at the time of writing this, I wanted to watch a tv series "Faadu" and it's only available on SonyLiv. So, maybe I will get a weekly or daily or maybe 1 month subscription of SonyLiv to make that happen. Now *tomorrow, if there comes a new entertainment streaming platform that contains all those movies and series that are* ***not*** *on Netflix / Prime Video, I am going to have a subscription of it so that I don't have to subscribe for SonyLiv, HBO, ShowTime etc and I get all in one.*

In order to ship the MVP quickly, our product team has identified that we can ship either the consumer application or the service provider application. In other words, due to time constraints, we need to pick one of the two and then work on launching that in the MVP. Which of the two applications would you consider more critical to include in the MVP and why?

Answer:

I believe that **"Consumer Application"** would be more critical.

Rationale:

Because it all starts with the customer. So, if we launch the consumer application then one of our team can also start to work on the branding of the product in parallel while the engineering teams will start working on the service provider application.

Strategy To Make Things Work with just Consumer Application:

What we can do here is to launch the consumer app. The question comes that where will the listing comes from?

So, to do that, we can get in touch with some companies who provide blue collar service providers, and we can get manually add the services provided by each provider of that company in our database manually and once the data is inserted into our database, the consumer app will be fetching data from the same database and show the listing.

The advantage is that with some manual effort, we can make things work with one app. The disadvantage is that there will be chance or errors, manual efforts, resources to do manual work etc.

My Opinion about 2 separate apps:

Given the description of the scope of the product, I believe a single application can be used for both service provider and consumer. What we can do is to add an option at the time of registration where the user can choose his type (Consumer or Provider) and based on that we can show him the feature that he needs to see.

I believe, with some access control (at feature and customer level), we can achieve our idea through one application. And it will be easy to manage as for different applications, we will have a separate repository for each, separate databases (maybe), separate deployment and release cycles etc which can be avoid if possible.

How would you design the system architecture and choose appropriate technologies to accommodate the platform's expansion from an MVP to serving 50k service providers, 2M customers, and 500 admins/operational staff? In addition to addressing scalability, performance, and efficient workload management, how would you ensure the system's robustness, fault tolerance, high availability, and seamless user experience in the face of increasing demands?

Answer:

Before deciding about the non-functional requirements, let's see the constraints / limitations that we have,

Constraints:

- Lack of Time
- It's a start up so a big up-front investment would not be wise.
- The feedback and upgrade cycle should be quick.

Actual Answer

The company needs should go for a **"Cloud-Based Application Development"** approach here.

So, the company can go for AWS Cloud services. Following are the cloud-based services we can use here:

1. DynamoDb (A NoSQL database)
2. AWS Lambdas using Node.JS or Python (For business logic)
3. AWS S3 Glacier (for consumers or providers who deleted their accounts)
4. AWS SNS (for alerts or notifications)
5. AWS Load balancer (For load balancing)
6. Redis (Open source) or AWS Elasticache (For caching of most search services) to avoid delay and quick response.
7. For payments, we can use Google Pay or Apple Pay.

Rationale for choosing Cloud Based Solution:

- Pay as you go basis, so we don't need to make an upfront purchase for infrastructure etc
- No need to hire resources to manage infrastructure.
- A cloud provider takes care of infrastructure level security, disaster recovery, robustness, fault tolerance, scalability so most of the NFRs will be automatically handled.
- With cloud-based deployment solution, the app deployment will be quick and easy.
- No need to set logging, auditing explicitly. Amazon Cloud watch can do that for you.
- We can focus on the actual business instead of spending time on infrastructure that promotes our actual business.
- Given our timeline, it's the quickest way to develop and deploy our application.
- Hiring resources will be easy as we will have to look for resources that has expertise in that specific cloud provider and have few certifications (this advantage doesn't go in favor of me :p as I don't have much cloud certifications in my profile but I am not afraid to mention facts)

The only disadvantage I can see here is that in the longer run let's say that we have 2M customers, then maybe the cloud-based pricing model will cost us more than the on-premises infrastructure so we may have to increase our commission to cover costs of cloud-based services and still have good profits.