Escape startup work hell, build products and find work life balance.

Practical Strategy

Problem

Working on multiple products at the same time is quite a challenge. Especially when you have a small team and tight deadlines, and this is such a typical experience when working at a startup or as part of a small team in a larger organization.

You know the drill, lots of overtime, competing priorities, crazy deadlines, endless pivots, no work life balance.

There is a another way that yields good results and leads to order out of chaos.

Solution

Let me share with you how to think about prioritizing tasks to :

- reduce risks
- improve the probability of a successful outcome
- and find ways to accelerate the deliverables

The key to success is to :

- identify overlapping capabilities or components across multiple products being developed
- tackle these components first
- but build only the minimum of what's needed to build an entire end to end solution
- first iteration has to move fast to build the momentum

Outcomes

I understand that the outcomes won't be perfect, however, building first iteration of an end state solution enables gathering feedback from the users and will result in a strong foundation for subsequent iterations.

This approach greatly reduced various risks typically associated with product building.

But the most important outcome is that you and your team may be able to **reduce the overtime**, **lower stress levels** and begin your journey towards **some sort of work life balance**.

End users will appreciate driving product vision and get excited to see it take shape.

There is no universal panacea for startup work hell, however, applying small baby steps consistently over time will make things a little bit better one iteration at a time.

Example

Let’s demonstrate how to apply the solution while building the following three products.

- Product A - use case to build a generative AI based integration
- Product B - use case to build recommender
- Product C - use case to build mobile/web application to recommend content

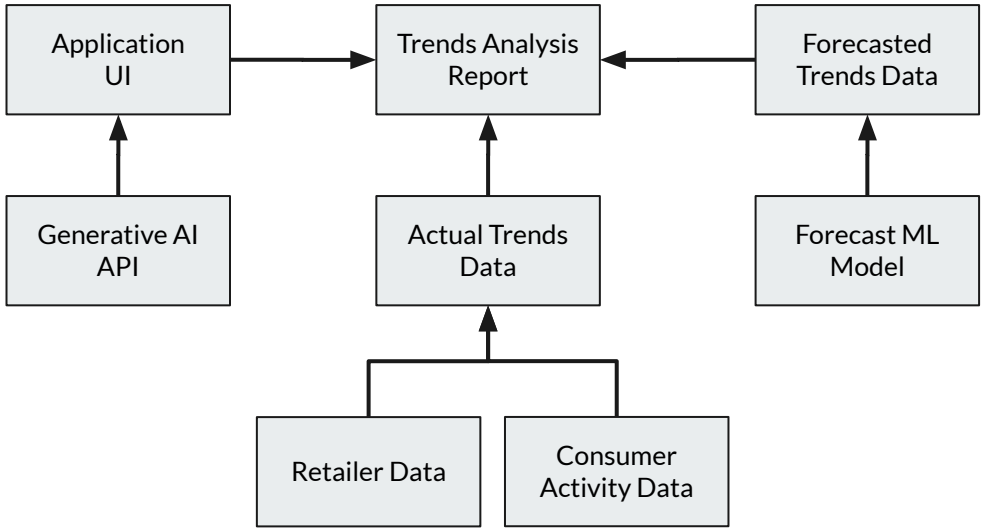
I will share the thought process for how to identify the common components, reduce the scope to a bare minimum, construct and end to end solution in rapid fashion all during a single iteration.

SHARED CAPABILITIES | REDUCE RISK | SPEED | BARE BONES

Guiding Principles

Product A - Generative AI integration

- Ask which components can be built first to test end to end? ***(bare bones UI, actual trends, report)***
- Any components needed for other products? ***(actual trends)***
- Ask which components can be simulated? ***(forecasted trends, generative AI API)***
- Ask what can we build in 2 weeks? ***(ui, report, actual trends)***
- Can you identify risks? ***(generative AI API integration, forecast ML model, consumer activity data)***

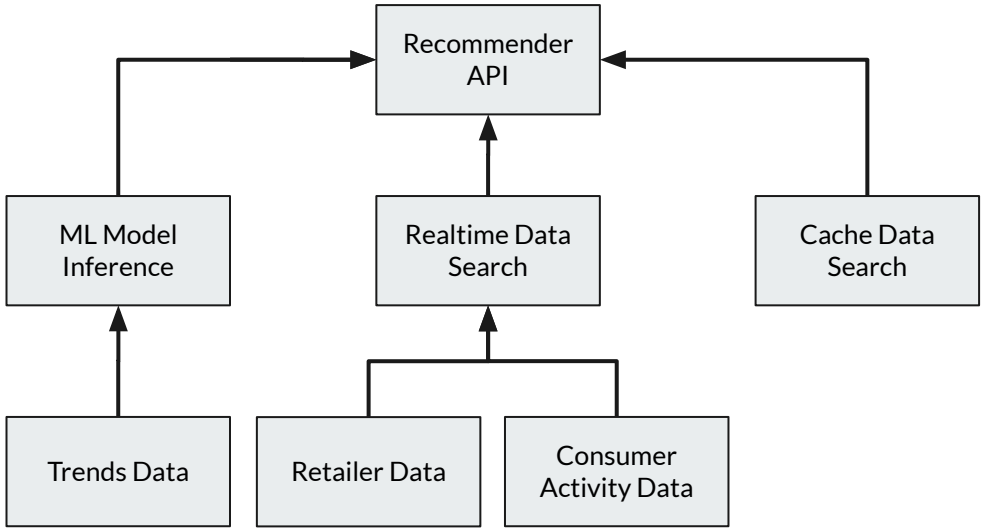


SHARED CAPABILITIES | REDUCE RISK | SPEED | BARE BONES

Guiding Principles

Product B - Recommender

- Ask which components can be built first to test end to end? ***(recommender API)***
- Any components needed for other products? ***(recommender API)***
- Ask which components can be simulated? ***(trends data, ML model, cache data search, realtime data search)***
- Ask what can we build in 2 weeks? ***(recommender API, realtime data search)***
- Can you identify risks? ***(ML model inference, trends data, cache data search, consumer activity data)***

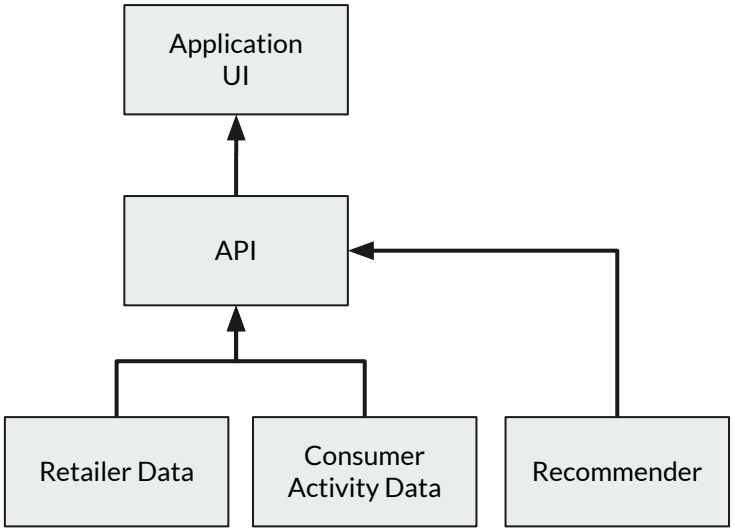


SHARED CAPABILITIES | REDUCE RISK | SPEED | BARE BONES

Guiding Principles

Product C - Mobile/Web application

- Ask which components can be built first to test end to end? ***(bare bones UI, api)***
- Any components needed for other products? ***(consumer activity data)***
- Ask which components can be simulated? ***(retailer data, consumer activity data)***
- Ask what can we build in 2 weeks? ***(ui, api, consumer activity data)***
- Can you identify risks? ***(recommender, retailer data)***



The Build

We bring common dependencies identified from all three products together into our first engineering sprint.

- Product A - use case to build a generative AI based integration **=> trends data, report**
- Product B - use case to build recommender **=> recommender API**
- Product C - use case to build mobile/web application **=> bare bones ui, consumer activity data**

What has emerged is a set of core components that enable the rest of the build. We begin with focus on developing datasets needed across all the products for reporting and machine learning activities, the recommendation API and UI needed to collect and interact with data.

Data, API, UI

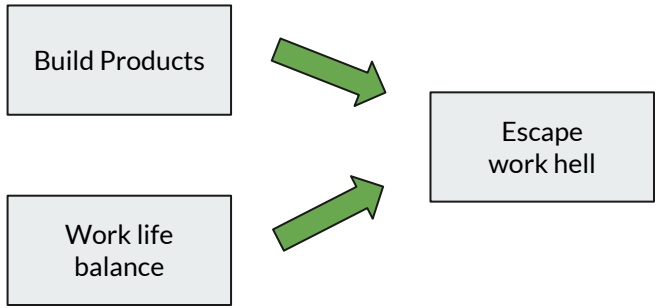
Seems obvious, but it's far from it.

Now we celebrate!

How does this help me?

This approach brings about a WIN-WIN-WIN situation for the engineering team, end users and management.

- Engineers can work quickly without analysis paralysis and materialize the grand product vision.
- End users get to quickly interact with the product and drive product direction with feedback.
- Management can quickly observe forward momentum, guide product development and user community and validate ROI.



Questions?

Filip Szalewicz
fszale@gmail.com