TheAiEdge.io

HOW TO MANAGE MACHINE LEARNING PROJECTS FOR MAXIMUM ROI

ASSESS - PLAN - EXECUTE - ITERATE

1. Product Discovery

What is the size of your market and for what ML solution?

Like any business endeavor, building a new software requires a deep understanding of your **customers' problems** and how likely those problems are to be addressed by a **Machine Learning solution**.

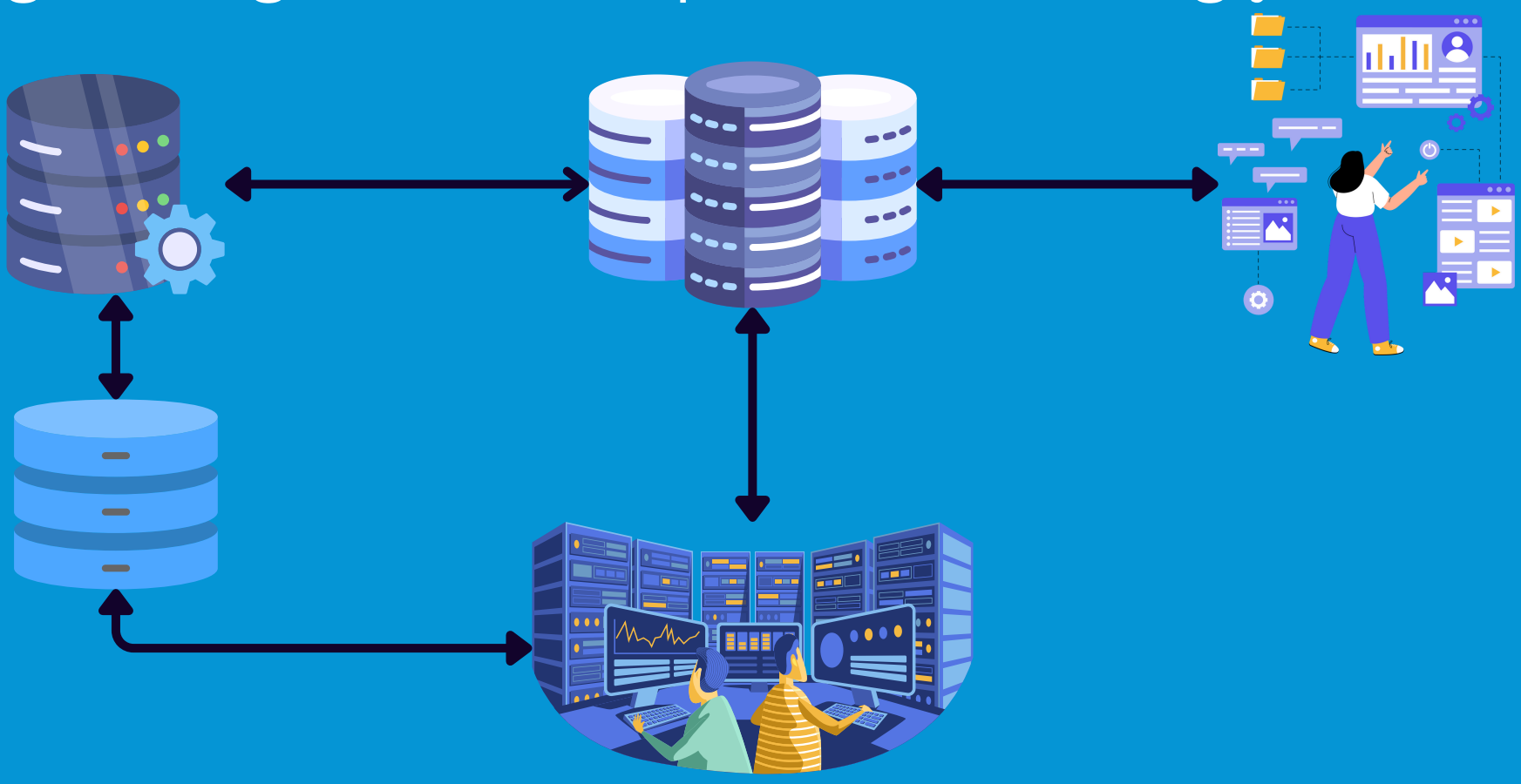
This understanding of the problem is critical to estimate the potential **monetary gain** that such new product or feature is likely to generate if implemented. It is also important to ideate if there is a feasible ML method that is likely to solve the established problem.

2. Assess the data Infrastructure

This step will really define the complexity of the project

This is critical to assess the quality of the **data** and the **data infrastructure** as soon as possible. This step will have a heavy influence on the **cost of the project** or even its feasibility.

The faster this assessment is done, the faster engineering work can be planned accordingly.

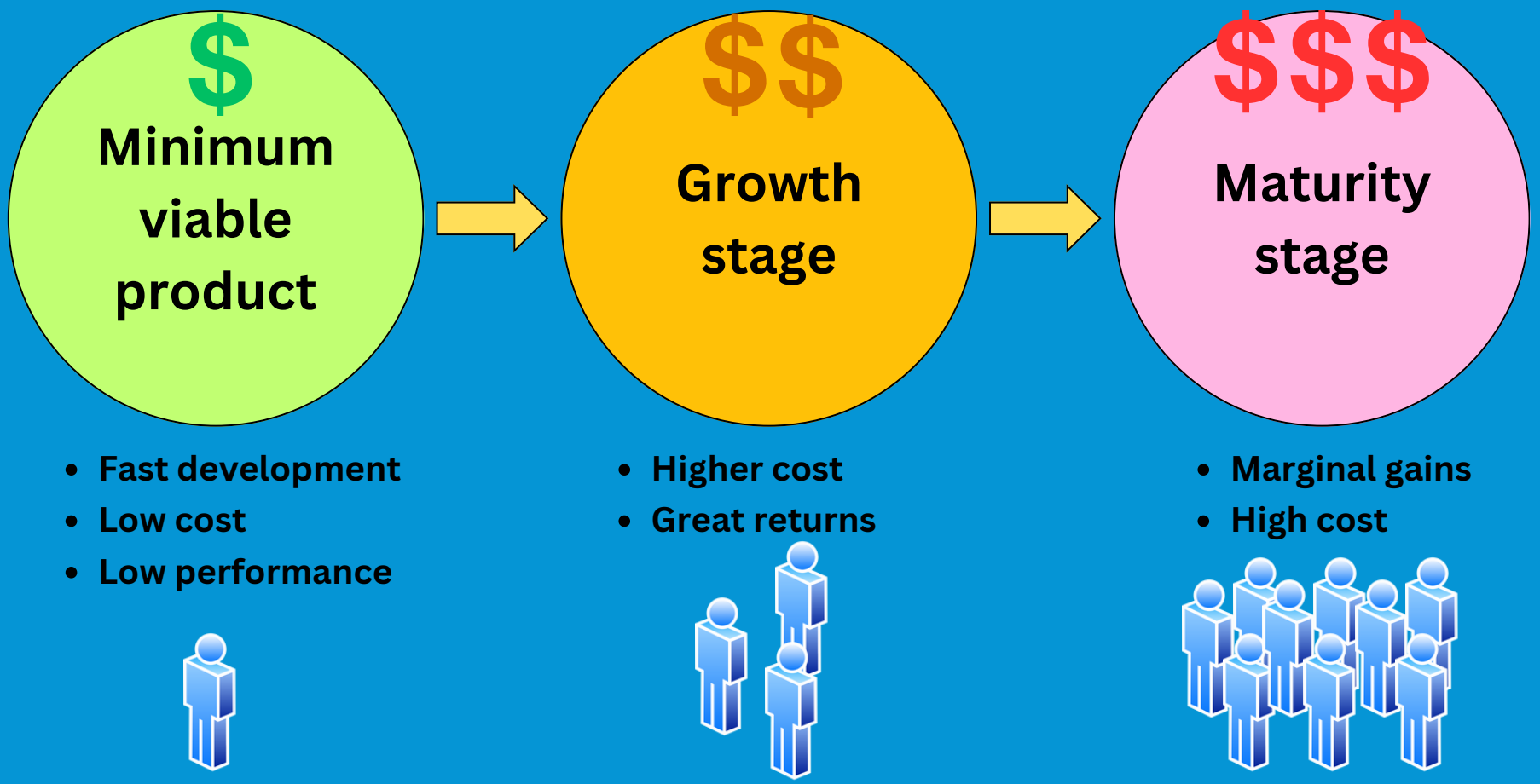


3. Work in Stages

Reduce the risk

A Machine Learning project can easily be broken in 3 stages:

1. The **Minimum Viable Product** (MVP): fast development, low cost, low performance. To assess the viability of such project.
2. The **Growth stage**: higher cost, great returns. To establish the foundations of a successful product.
3. The **Maturity stage**: Marginal gains, high cost. To squeeze as much gain as possible.



4. Prioritize

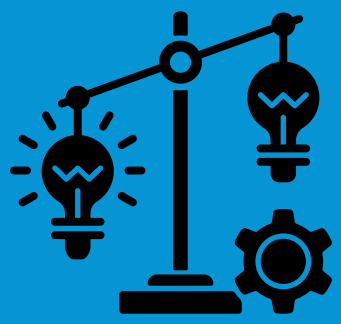
Maximize gains, Minimize cost

Not all Machine Learning projects are born equal! Utilize the product discovery step and the infrastructure assessment step to estimate the potential gains and cost.

Optimize for **high gain**, **low complexity/cost**. Prioritize the projects that have a high propensity for success and discard the others!

Maximize Impact

- It is a project well solved by Machine Learning
- It maximizes business metrics (revenue, user utility, ...)



Minimize Cost

- Complexity
- Cost of the data
- Cost of ML errors

5. Architect the Solution

Get the business requirements, translate them into technical requirements

That is the time to dig into the details and refresh your **System Design** skills!

Get the **business requirements**:

- How many inference requests / day or / seconds?
- How many users?
- Minimum predictive performance?
- Acceptable latency?
- ...

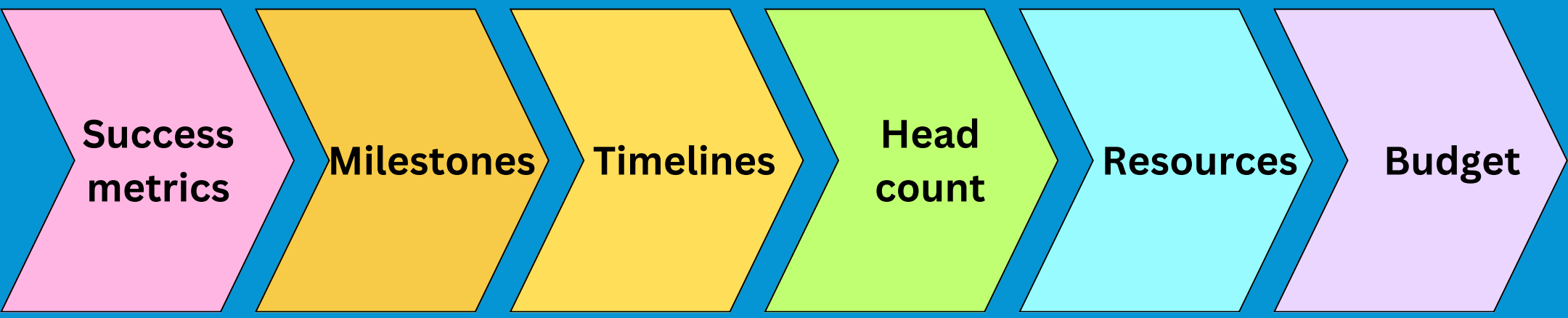
Translate into **technical requirements**:

- Batch or real-time?
- How many servers?
- Fallback mechanisms?
- Do we need databases or queues to store resulting data?
- ...

6. Strategic Planning

Setup the long term vision

- Establish **success metrics**: those are your North Star goals!
- Establish **milestones**: what are the mandatory steps to achieve such that you reach the North Star goals?
- Establish **timelines**: make sure those steps are feasible in a finite amount of time! Optimize for speed to market.
- Establish **head count**: how many people and with what skills do you need to succeed in those steps?
- Establish the **required resources**: what tools do you need to succeed?
- And more importantly, putting everything together, establish a **budget**: how much money do you need to implement the solution in a time boxed fashion?



7. Tactical Planning

How do you actually execute!

That is where Agile development strategies can be useful!

- When the model needs to be developed and by whom?
- When the data pipelines need to be built and by whom?
- When does the model need to be deployed and by whom?
- ...

How do we **coordinate** all the different people **across teams** such that the project moves smoothly without delays?

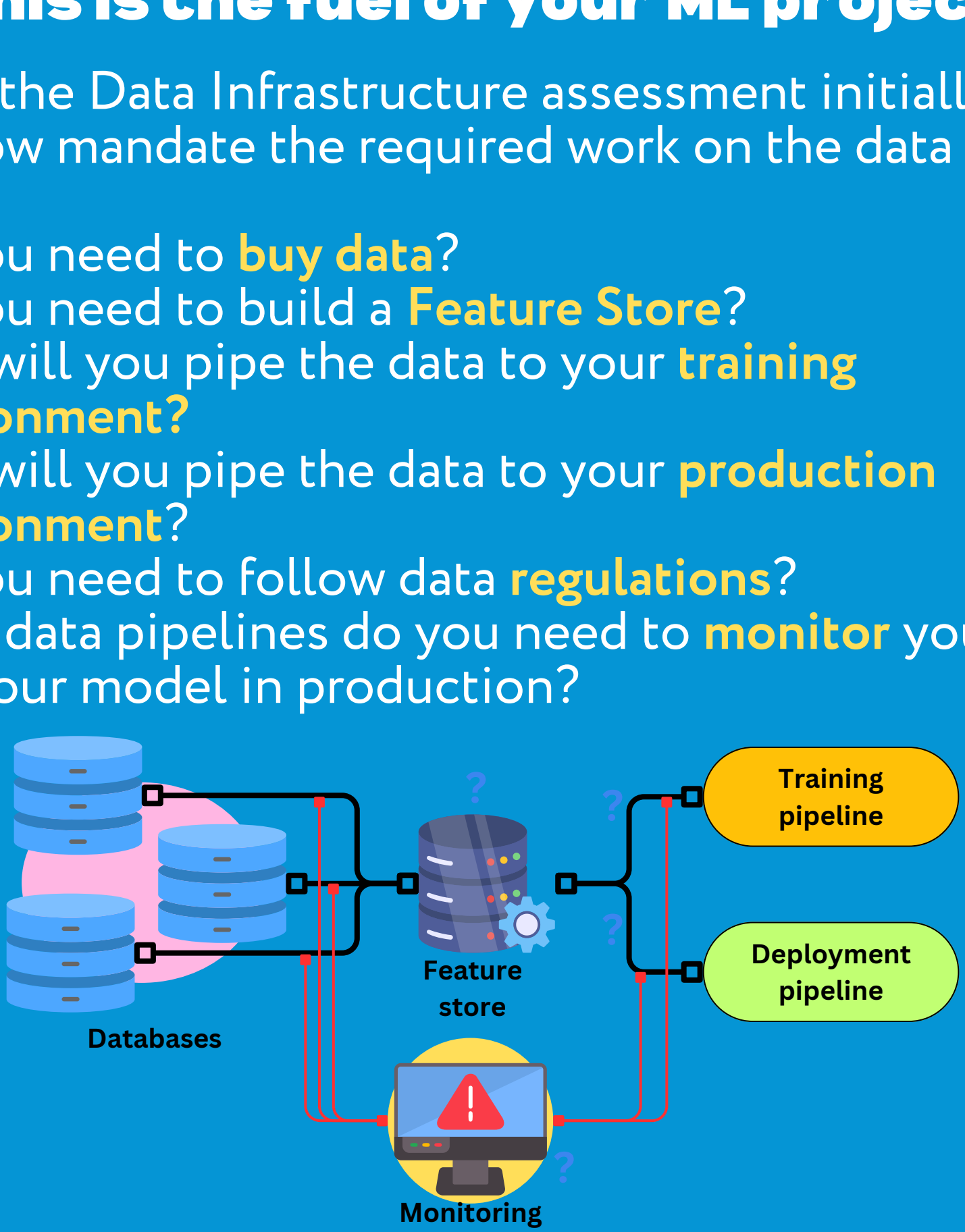
- **When the model needs to be developed and by whom?**
- **When the data pipelines need to be built and by whom?**
- **When does the model need to be deployed and by whom?**
- **...**

8. Data Pipeline work

This is the fuel of your ML project

Based on the Data Infrastructure assessment initially done, we can now mandate the required work on the data side.

- Do you need to **buy data**?
- Do you need to build a **Feature Store**?
- How will you pipe the data to your **training environment?**
- How will you pipe the data to your **production environment**?
- Do you need to follow data **regulations**?
- What data pipelines do you need to **monitor** your data and your model in production?
- ...

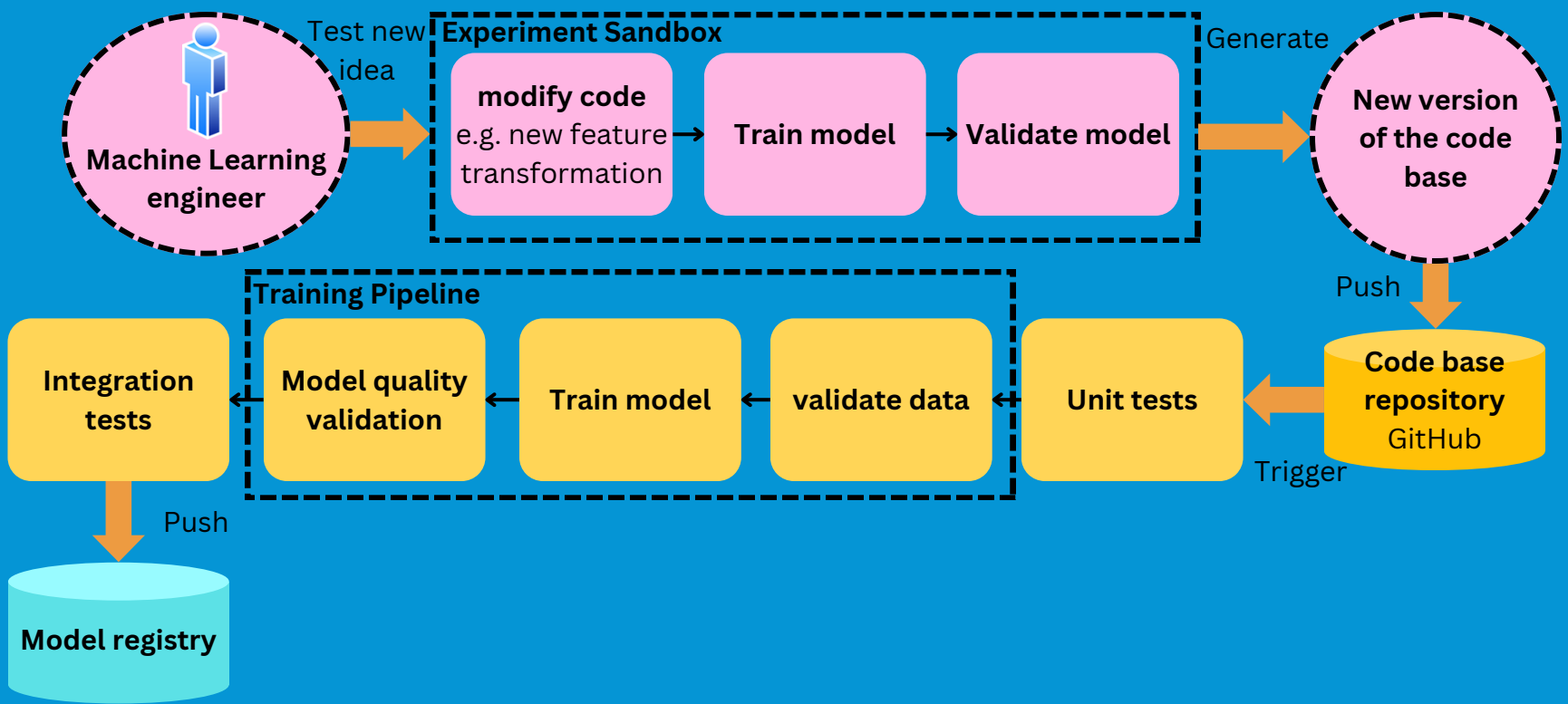


9. Build the training Pipeline

Drop the notebooks!

In this Era of **automation**, we need to ensure that a model can be retrained or fine-tuned at will or on a schedule with a high level of control and then productionized at a simple click of a button.

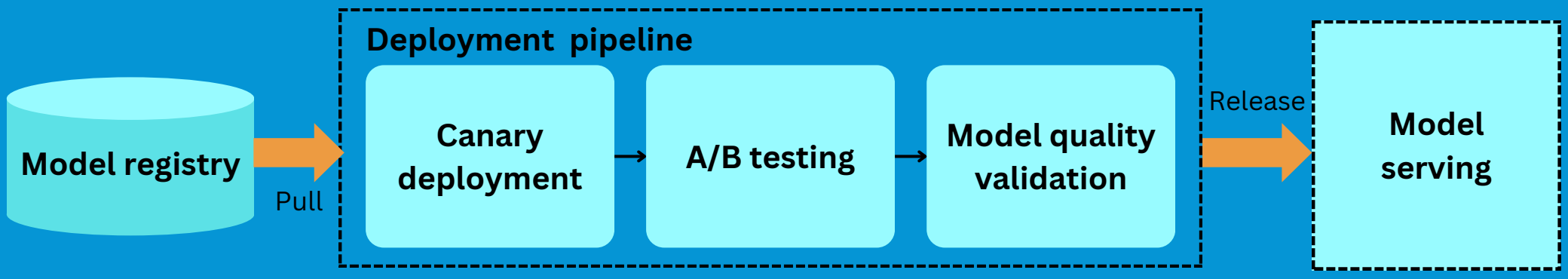
Of course, this level of automation is highly dependent on the current stage of the project development (MVP, growth, maturity). However, at any stage, the training pipeline and the production pipeline should be **highly coupled** and developed by the same people or by people working closely together.



10. Build the production Pipeline

What actually provides value to the users!

The production pipeline is the extension of the training pipeline and is driven by the business use case. How do you actually expose the model to your users? Do you need to build an **access API**? What level of **security** do you need? Do you need a **post processing** module to make the model inference useful?

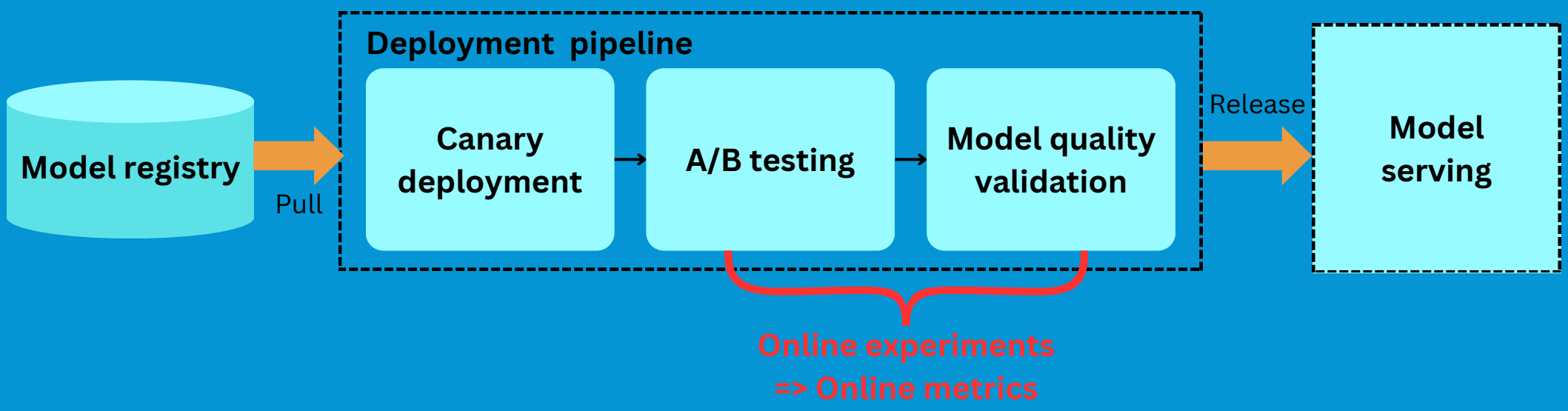


11. Test in Production

Is your model actually good?

Testing in production is the only way to actually know if your model is providing the expected value.

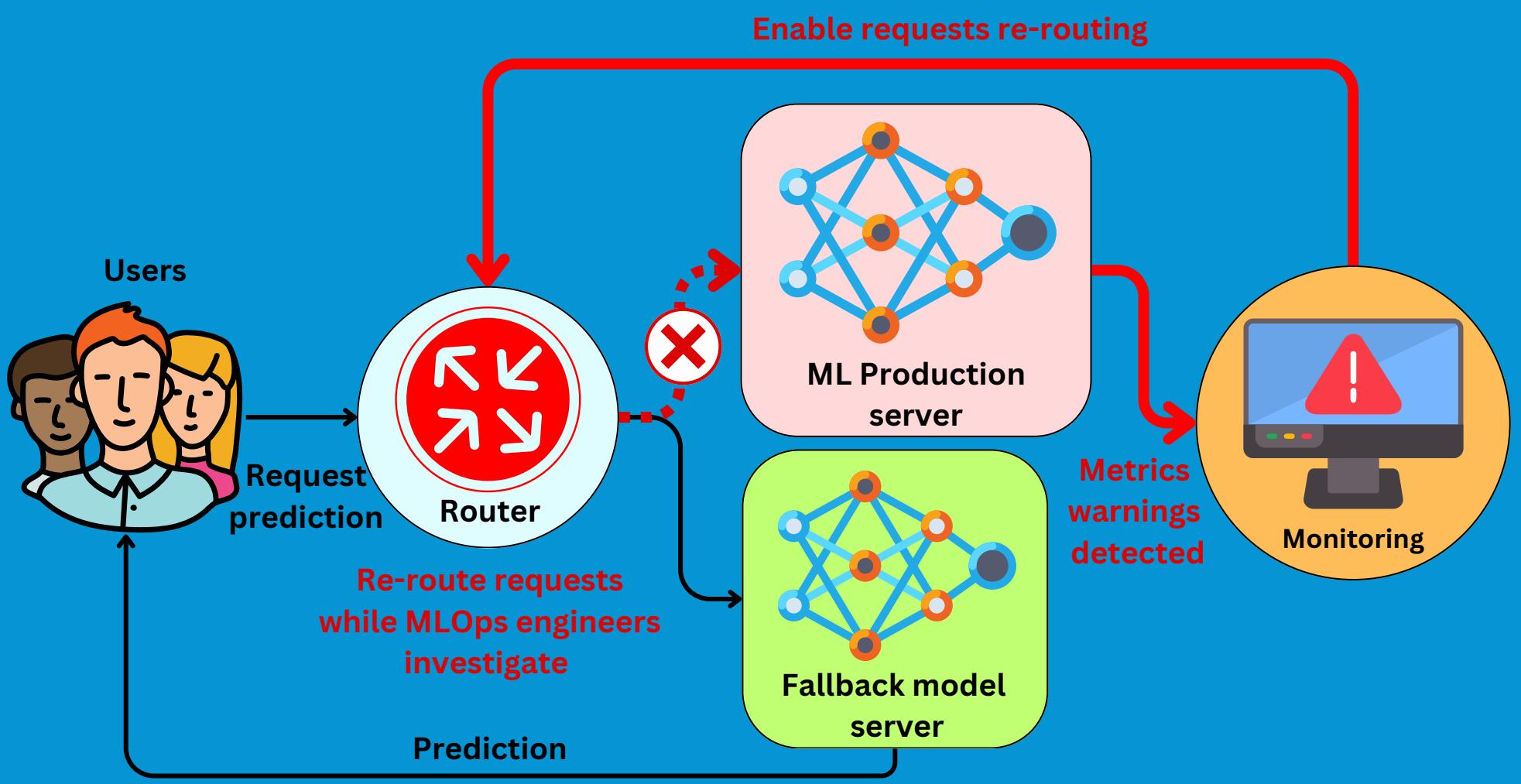
The pipelines to perform **A/B testing** and **Canary deployments** need to be carefully designed to quickly detect if a model is unexpectedly underperforming and remove it without harming the user experience nor the revenue.



12. Monitoring in Production

What about your model becomes bad tomorrow?

In production everything can happen! **Data or Concepts can shift**. Servers can **break**. Softwares can have undetected **bugs**. You need to monitor those different components and implement plans of action (automated or not) as fallback mechanisms in case something goes wrong.



13. Document

The most important step that nobody likes!

Along with writing unit tests, this part of the work might be the most **underrated** effort that contribute to the success of a project in the long run!

Update wiki

Comment code and API

Educate team members

14. Iterate

TheAiEdge.io

Get back to step 1!

If more improvements **make sense financially**, you are just getting started! The field is moving so fast that by the time you done with one iteration, you likely have new tools and techniques that you haven't had the time to test nor implement yet!

You may need to invest in the next stage of development or simply update the current one.