Part 1: Strategy & Architecture (40%)
Task 1: Root Cause Memo

1. What are the top 3 reasons for 50% error?
   Rank by impact. For each, explain: what's wrong, why it matters, rough impact
   Estimate.
   - Memory issues: Depending on the severity of the crashes (I'm assuming there's no escaping the crashes and they are common) there would be forecasts that are simply not done. Any kind of software error of this gravity needs to be resolved first as the rest of the pipeline after the batch processing will suffer from wrong data.

   - Dead stock and wrong assortments: It is said in the description that the system forecasts 12.6M products that are mostly dead stock or wrong assortments. This is a fundamental problem that needs to be resolved first because the data seen during inference needs to be representative of the data used during training. I.e., maybe the trained models weren't trained on dead stock as this type of inventory isn't sellable, or at least we shouldn't prioritize these products. Either way, if these items are dead stock with 0 sales and it's still possible to forecast them, this will drive the error significantly. Wrong assortments are products which have their data wrongly processed which means the model will forecast them wrongly.

   - Under-represented categories: Much worse error for footwear and sportswear which lead to a higher overall WMAPE. Maybe these two categories were under-represented during model training when they actually have a significant weight in the WMAPE.

2. Why does Footwear have 233% error while Made-to-Measure has 40%?
   What would you investigate? What hypotheses do you have?
   - Under-represented category: It could be that the category Footwear was under-represented when the system or model was trained. I would make sure data for Footwear is in quantity and quality, meaning the inference data is preprocessed the same way it was during training. I.e., footwear can be seasonal: Was the model retraining taking into consideration seasonal data? Made-to-Measure could be more stable throughout the year.

3. One critical decision you'd make in Week 1
   If you could only fix one thing first, what would it be and why?
   - OOM crashes: This is probably the most important driver of errors while also probably being the easiest fix. I'd investigate if there are improvements in the code that could be made or if using a different compute helps. Also, generally speaking I'll always favor quick wins when time is constrained.

Task 2: 90-Day Plan

Week #	Improvement	Reason	Impact	Risk	WMAPE Metric target
Month 1					
Week 1	Retrain without dead stock	Dead stock isn't useful to forecast if they are not going to be sold either way. Having a model that relies on good, sellable products is more performant. This needs to be done first because it has a high impact and could help with the second task, which is getting rid of OOM errors.	This will improve forecast performance and also memory issues which is important to tackle. This also translates into lower costs.	The logic to categorize a dead stock needs to be well defined because everything coming after this step in the pipeline is dependent on this.	45%
Week 2	OOM error reduction (support from Junior dev)	The infrastructure is the foundation and needs to be robust to train the models and run the pipelines.	Less downtime running and troubleshooting the batch process and pipelines. More products that are correctly preprocessed and forecasted.	There is a risk the system still has too much data and will continue to crash if data management isn't optimized. Plus we can't use bigger and more expensive AWS computes.	40%
Week 3-4	Features assessment and correction (support from Data engineer)	The wrongly assorted products are problematic and in high volume. They need their data to correctly reflect what the models were trained on.	The precision of the forecast should continue improving with this as the inference data will be of higher quality and it will impact a high volume of products.		35%
Month 2					
Week 1	Investigate the Sportwear category	Sportwear was showing a >1000% error in WMAPE. There is probably a fundamental error causing this which could be easy to solve while also having a high impact.	The investigation will enable us to understand the low performance for this category and better preprocess the data or train the model.	Investigation and data analysis are open work that can take more time than anticipated.	35%
Week 2	Retrain the Sportwear category model (support from Junior dev)	The model will need to be retrained following the findings of the previous week.	The retrained model will have a better representation of the Sportwear data and forecasting results.		32%

Week 3-4	Investigate the Footwear category and retrain the model (support from Junior dev)	Same reason for choosing this improvement as the Sportwear category, but Footwear comes second due to a lower WMAPE.	The retrained model will have a better representation of the Footwear data and forecasting results.	Same risk as the previous week 1	30%
Month 3					
Week 1	More investigation on seasonality, promos, holidays impact across all products	If improving performance was possible for two categories, there are probably some findings that can be easily applied to the rest of the inventory.	Can potentially improve all of the inventory by developing new features associated with seasonality and more.	This is additional exploratory work that could help get better performance but there's a risk we see little performance improvement as the bigger gains were made in the previous months.	27%
Week 2	Data drift monitoring (support from Junior dev)	From the description of the task, I'm assuming the forecasting performance is already well monitored but it doesn't mention data monitoring.	Dashboard where we can associate the forecasting performance to data drift (seasonal changes and more).	Running a dashboard will increase the cloud cost.	25%
Week 3-4	Prepare demo for clients (support from Director)	When the forecasting system is debugged, performance has improved, it is time to prepare the demo for the clients.	Impact is high as selling the system to the clients is ultimately the only goal.		25%

Task 3: System Design

1. Inputs/Outputs

Inputs:

- Product data: nb_days_since_launched, current_inventory, sales_last_14d, current_margin, markdown_pct
- Sales data: daily sales, inventory by product-site
- Site config: site_type (ecom or not), active_assortment_list (for ecom)

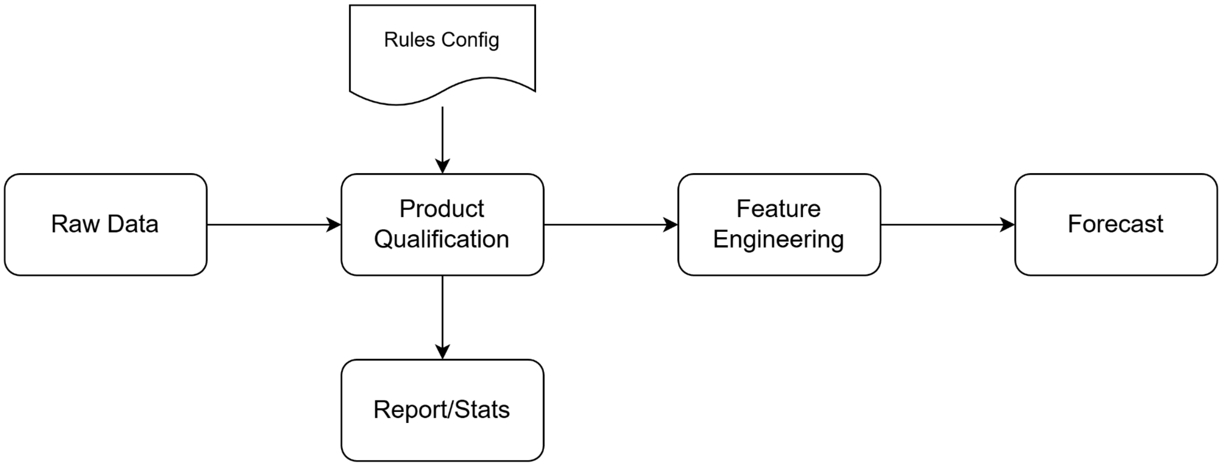
Rules config:

- Min_markdown_pct
- Min_sales_last_14d
- min_current_margin
- Min_nb_days_on_shelf
- Max_nb_days_on_shelf

Outputs:

- Qualified list: style_id, site_id, status (FORECAST/EXCLUDE), reason, confidence
- Summary report: qualified % and more stats

The product qualification step is included in the whole system as follow:



2. Business rules

Key concept I'll consider to define the rules:

- Old products that don't have much sales and margins shouldn't be eligible to be forecasted

- New products should have higher forecast propensity because they need more exposure to let the customers
- Clearance sales should be considered to forecast even older items
- Products not in inventory should automatically be excluded

Rule #1: Inventory cannot be empty

We cannot sell what's not in store so:
EXCLUDE if: current_inventory = 0

To note that it might be possible to still forecast a product that is soon to arrive in store if we want to optimize sales.

Rule #2: New products should be forecasted

New, unknown products should be forecasted more often as they need exposure to drive more sales. This will also make us gather data about their sales quicker which will feed our sales history for these new products and help future forecasting.
INCLUDE if: nb_days_since_launched < min_nb_days_on_shelf days, regardless of sales history

Rule #3: Old products that don't sell shouldn't be forecasted

Older products should have a lower sales propensity, but only if they are not generating revenues enough.
EXCLUDE if: nb_days_since_launched > max_nb_days_on_shelf days AND (sales_last_14d < min_sales_last_14d OR current_margin < min_current_margin)

Rule #4: Products should be forecasted when having a clearance sales

Clearance items need forecasts to optimize revenues and clear inventory, therefore they should have a higher propensity to be forecasted.
INCLUDE if: markdown_pct > min_markdown_pct AND inventory > 0, even if product is old

Rule #5: Store type (ecom vs physical)

There are fundamental differences between physical and ecom stores that would require more rules, but I don't know enough about these to be able to define them. A future improvement to these rules could be to add different thresholds depending on the store type. I.e., an idea could be, if it’s harder to restock physical stores, to also expect them to keep their inventory for a longer time and continue forecasting older products.

3. Scale

- Use Polars lazy or potentially migrating to Spark if the dataset is big enough.
- Reduce the volume of dead stock and wrongly assorted items.
- Divide the dataset into categories (ecom/physical stores, menswear/womenswear, etc.) to train different models with less data. This could potentially help having even better performance.

4. Validation

Here are the metrics I’d use to evaluate the performance of the rules algorithm:

- Qualification rate: qualified / total
- False negative (FP) rate: Products excluded that sold next week (target <1% - critical metric)
- False positive rate (FP): Products included with zero sales (less critical)
- Exclusion reasons: Distribution (dead stock, wrong assortment, low volume, etc.)
- Max memory used, OOM crashes.
- WMAP. We should be able to see an improved WMAPE by removing the undesirable items.

Finally, to make sure the new approach is working as it should be, I’d compare it to the current system using A/B testing.

- Group A: Check metrics for forecasted items using the current approach
- Group B: Check metrics for forecasted items using the new rules

When going to production with the new system, we can also gradually rollout the old approach or do shadow testing beforehand so that we can be sure the new approach is working before having a real impact after deployment.

The deployment will be considered successful if we observe less FPs (items that shouldn’t have been forecasted), lower total WMAPE, and zero OOM crashes.