Ok. I am going to write down all of the components of this application so that you can understand what we are going to build here. KIMBALL will be a full stack application where the front-end will be REACT (but we will probably test with a simple Streamlit testing front end) AND it will use a FASTAPI backend. Everything that we want the user to be able to do will be done via API calls to our FastAPI server and there will be little to no business logic code in the front-end. The front-end just needs to know how to use the APIs and in what order to call them. So we will have a robust set of backend services/APIs in FastAPI.

KIMBALL will consist of four primary phases. See below…

1. Acquire – The Acquire phase will be to acquire data from a VARIETY of different sources. These sources can be applications (e.g. ERP, CRM, EPM), databases (e.g. Oracle, SQL Server, Postgres, Redshift, MongoDB, etc), APIs (to pull data through and even raw storage containers (e.g. S3, Google Storage, Dropbox, Azure Storage, etc…). We will need to be able to define properties for the different types so that we can connect and pull data through one or multiple of these “types”. When we acquire the data, we will flatten the data into Clickhouse tables structures in our bronze database (bronze layer). Bronze data will simply be direct pulls with little to no transformation. The only transformation that may need to occur would be if we are pulling nested hierarchical data from something like MONGODB. But we will tackle that one later. We should be able to create the appropriate “extraction” language (e.g. SQL queries, REST API calls, MQL queries, etc) to get the data. But the guiding principle for this Acquire phase is to do as little transformation of the data as possible so that our acquire step is not overly complex and we don’t have to create a ton of different logic in different languages. We want to get the data as quickly as possible and put it into our bronze schema inside of clickhouse.
2. Discover – The Discover Phase is where we will go through and document all of the tables and columns that are in the bronze schema. You have much of this code already written and we are creating the metadata\_catalog.json file right now. This is the final result of the discovery phase.
3. Model – After we have Discovered the data in the bronze layer then we want to Model the data. There are two major things that we want to do during the Model Phase. First, we want to build a potential ERD that the user can view and then change/edit. This would allow us to then potentially join our data to write then into our silver schema. The Silver schema (layer) within clickhouse will basically be the correct combination of the data (and hopefully a simple reduction of tables) that will likely be more like a third normal form (3NF) model. So the user will need to view the ERD, make changes and then from that we need to be able to generate the SQL to then combine the data properly into fewer tables (than bronze) in our silver layer. Then there is a second part of the Model phase where we will then discover the hierarchies (like we are doing now in the code) and we need to also create a json file with the proper potential hierarchies and their levels. Once again this will be presented back to the user where he or she can modify the hierarchies (e.g. delete siblings not wanted, change leveling, names of the hierarchies, etc…). Then once the user makes those changes and “submits” then he or she can see the updated hierarchies and move forward or refine. Once the user is satisfied with the hierarchies, then we need to generate a second ERD but this time it will be in the form of a star schema. We will need to separate our dimensions into proper dimension tables with our hierarchies and the facts into fact tables. And we need to make sure that we have our key columns so that we can join properly. This ERD will then be used to construct our final schema (layer) in clickhouse which will be the Gold layer (schema). Then we need to generate the proper SQL to be able to take the data in silver and the properties that we have generated in terms of the hierarchies and which dimensions and facts and we need to then build the gold tables and the sql to populate them. In the gold layer our naming convention will always be [table\_name]\_dim for dimension tables and [table\_name]\_fact for our fact tables.
4. Build Phase – the final phase is our Build phase. With our BUILD phase we are going to construct the DAGs (<https://www.astronomer.io/blog/what-exactly-is-a-dag/>) that we can then “productionalize” as part of a permanent data pipeline. Essentially we need to be able to take all of the “stuff” that we have done from Acquire through Discover through Model to then construct the proper transformation logic (SQL based with Python if/when needed) and the be able to schedule and run these integrations with robust logging.

Again, everything we build will need to be enabled via FastAPI. We need to create a proper python class structure for each of these phases and major components.

To get started here is what we want to do… First, let’s create a proper scaffolding structure for our code and we can start to move things around. Keep in mind that our code and APIs should be organized around those four major phases – Acquire, Discover, Model, Build – Also let’s add a test folder for any python testing programs that we build so that we keep our code structure clean. Then we will build a frontend folder where we can use Streamlit to test our flow with some user interactions. Again, ultimately we will use REACT as our front-end but that will be build ONLY AFTER we have all of our APIs working properly and we have thoroughly tested everything with Streamlit.

Also, I have created the remote github repository and we should regularly add and commit and push our code (<https://github.com/kainam-jg/kimball.git>). Also, let’s make sure we keep copious documentation on the KIMBALL application.