Kapittx Upstream Data Sync Statistics API

In Business-to-Business (B2B) firms, Accounts Receivable remains among the top three tangible assets on the books; yet it is highly under-managed, leading to cash getting locked in the balance sheet. Businesses end up spending hours on manual, repetitive tasks, updating multiple spreadsheets, preparing cash flow reports, handling invoice disputes, or manually reconciling invoices. Using Accounts Receivable automation integrated with B2B payments, Kapittx is empowering collection and customer-facing teams to collect payments faster and more efficiently than before.

While the Kapittx data ingestion systems captures the sync statistics regarding the records for a customer’s or invoice’s syncs, the platform is unable to proactively indicate the issues in these sync batches. During a sync process, the data ingestion layer that sources data from the Enterprise Resource Planning (ERP) layer, stores information about how the requests were received and how many records during the sync passed/failed or were processed/unprocessed. The goal was to build a solution that is automated and proactively highlights issues in the sync batches.

We engineered a Sync Statistics API that will proactively provide the details on the sync batches, aiding Kapittx’s customer success functions in dealing with issues in sync. The tech stack consisted of the API which was built using Java, Spring Boot and MySql. Later, Postman was used to test the API.

The inputs to the API consists of:

1. A Kapittx subscriber company
2. The date of sync
3. The sync operation type (‘create invoices’, ‘update invoices’, create customer’, ‘update customer’, etc.)

The Processing logic consists of three layers:

1. The Controller component that receives the API request and delegates the request to the Service component.
2. The Service component preprocesses and validates the request and delegates the messaged data to the Repository component.
3. The Repository component then connects to the MySQL database and fetches the required information back to the service component and eventually to the Controller component.

Overall, this experience has provided me with valuable insight into the entire process of the software development cycle and a higher understanding of API architecture. By getting a hands-on experience with databases and the various API layers, I received the necessary skills required to further contribute to the advancement of software engineering.