Hardware and Software Requirements

Server Hardware: For hosting we will most likely containerize out application in order to host it via a managed Kubernetes cluster on one of the cloud vendors, this will allow us to quickly scale to any kind of capacity demands while keeping costs low since our application will scale down when traffic isn't present. We'll follow the same approach for any kind of internal facing services we may need for doing things like calculating company financial health scores and storing them for later use.

Development Software:

- Programming Language: Python is being used for it's wide range uses and ability to manipulate data, since we will be working with financial data python will work well.
- Web Framework: Django, a high-level Python web framework, facilitates rapid development and clean, pragmatic design.
- Database Management System (DBMS): We'll be using MySQL since it's a tried and true industry standard, and we have already started storing data so moving to something else would slow down development of the MVP.
- Data Analysis Tools: Libraries such as Pandas for data manipulation, NumPy for numerical computations, and Matplotlib for plotting data.
- Financial Data APIs: Integration with financial data sources (for balance sheets, income statements, etc.) is crucial.

Client-Side Hardware and Software: Lightweight, considering the application is web-based. Users need devices with internet access and web browsers.

Network Layout

- Cloud-Based Infrastructure: Using cloud services (AWS, Azure, Google Cloud) can be effective for scalability and reliability.
- Content Delivery Network (CDN): Implementing a CDN can enhance load times for users spread geographically.
- Secure Connections: Utilizing HTTPS to ensure data security during transmission.

Database Design

Company Table:

- companyID (VARCHAR(255) PRIMARY KEY): A unique identifier for each company.
- companyName (VARCHAR(255) NOT NULL UNIQUE): The name of the company, which is unique and not nullable.

QuarterlyFinancialData Table:

- id (INT NOT NULL AUTO_INCREMENT, PRIMARY KEY): A unique identifier for each record.
- companyID (VARCHAR(255)): Refers to the companyID in the Company table.
- fiscalDateEnding (DATE NOT NULL): The ending date of the fiscal quarter.
- Financial metrics fields (BIGINT DEFAULT NULL): Includes various financial metrics such as total assets, liabilities, equity, etc., each stored as a BIGINT. These fields are optional (nullable).
- UNIQUE KEY (companyID, fiscalDateEnding): Ensures the uniqueness of each record based on companyID and fiscalDateEnding.

AnnualFinancialData Table:

- data_id (INT SERIAL PRIMARY KEY): A unique identifier for each annual financial data record.
- company_id (INT SERIAL, FOREIGN KEY): Foreign key that references the companyID in the Company table.
- fiscalDateEnding (DATE NOT NULL): The ending date of the fiscal year.
- reportedCurrency (VARCHAR(5) NOT NULL): The currency in which the financial data is reported.
- Financial metrics fields (BIGINT): Similar to the QuarterlyFinancialData table, this table stores annual financial metrics.

Key Points in Design

Normalization: The design avoids redundancy by separating company data from their financial records. Each company is uniquely identified in the Company table, and their financial data are stored in the QuarterlyFinancialData and AnnualFinancialData tables.

Data Types:

- The choice of VARCHAR(255) for companyID in the Company table allows for alphanumeric identifiers.
- Financial metrics are stored as BIGINT, which accommodates large numerical values typical in financial data.
- The reportedCurrency field in the AnnualFinancialData table is VARCHAR(5) to accommodate currency codes.

- Referential Integrity: The use of foreign keys in the financial data tables ensures referential integrity. It links the financial records back to the respective companies.
- Uniqueness Constraints: The unique constraints on companyID and fiscalDateEnding in the QuarterlyFinancialData table ensure that there are no duplicate records for the same company and fiscal period.

Recommendations for Improvement

- Indexing: For performance optimization, especially in queries involving financial data, indexing key fields like companyID and fiscalDateEnding is recommended.
- Data Validation: Additional constraints might be needed to ensure data accuracy, such as check constraints for financial figures.
- Handling NULL Values: Defining a strategy for handling NULL values in financial metrics is important, as they could represent different scenarios (e.g., data not available, metric not applicable).
- Data Security: Implementing security measures to protect sensitive financial data, especially if the application will be handling user-specific data or confidential company information.

This design provides a robust framework for the TVI application to handle and analyze financial data effectively, ensuring both data integrity and performance.