**DTG-POC: Dynamic Table Generator**

**Project Overview**

The **Dynamic Table Generator (DTG-POC)** is a proof-of-concept application designed to streamline the process of creating and managing database tables dynamically from uploaded CSV files. The app was developed to address the inefficiencies and potential errors associated with manually creating database tables for each CSV dataset, a task that becomes increasingly challenging with large data volumes.

**Why This App Was Built**

The primary motivation for building DTG-POC was to automate and simplify the ingestion and management of CSV data into a database. The app achieves this by:

1. Allowing users to upload CSV files via a frontend interface.
2. Analyzing the CSV data to infer column types and suggest primary keys.
3. Obtaining user approval for the inferred types and keys.
4. Dynamically generating tables in the backend database.
5. Providing a frontend dashboard for managing the created tables.
6. Implementing a queuing system to efficiently process large datasets in batches.

This solution enhances productivity, reduces errors, and ensures scalability, making it ideal for handling dynamic datasets of varying sizes.

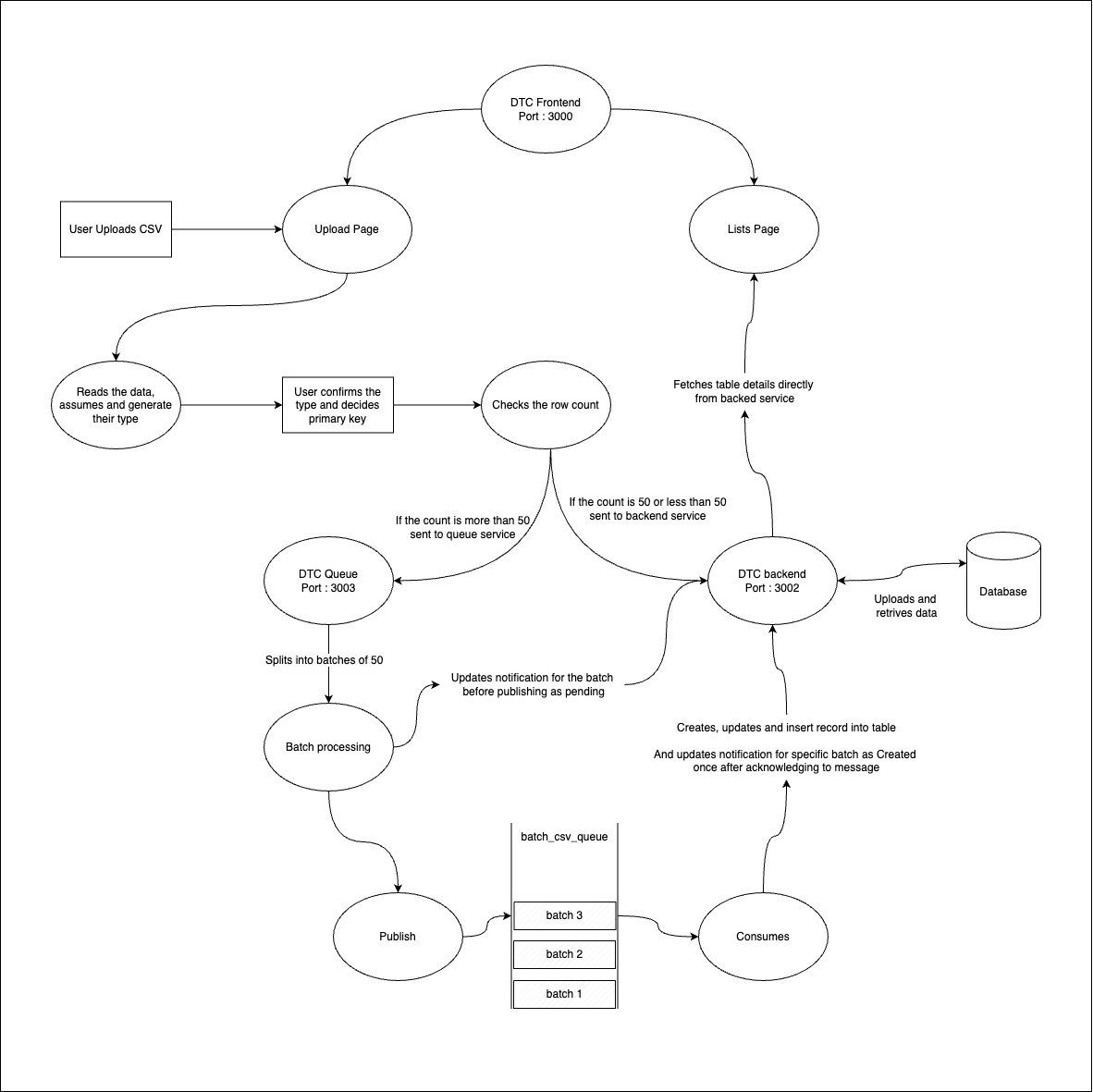
**Features**

* **CSV Upload and Analysis**: Automatically reads and analyzes CSV files to determine column types and suggest primary keys.
* **User Approval**: Enables users to confirm or adjust the inferred types and keys before table creation.
* **Dynamic Table Creation**: Generates tables in the backend database based on the approved structure.
* **Frontend Dashboard**: Offers a user-friendly interface to view and manage created tables.
* **Batch Processing**: Uses a queuing system to split large datasets into manageable batches for processing.
* **Notifications**: Keeps users informed with updates during table creation and batch processing.

**Architecture and Design**

**Data Flow Diagram (DFD)**

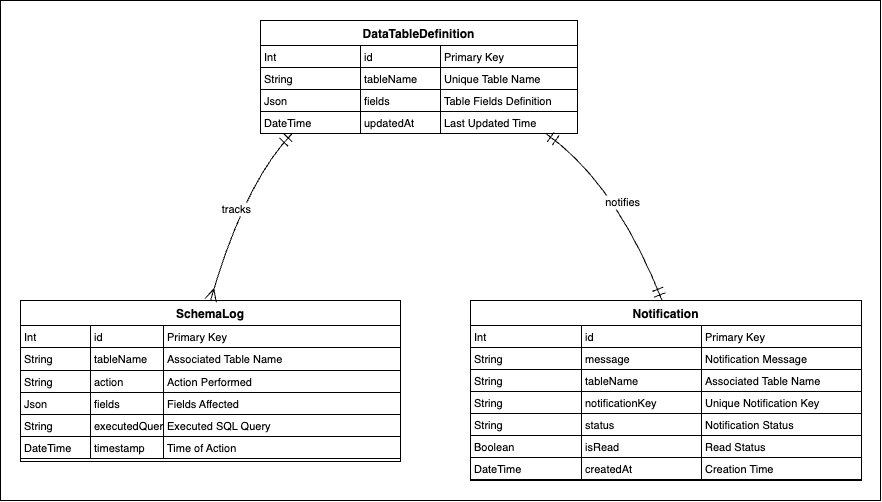
The DFD illustrates the flow of data through the DTG-POC application:



* **Frontend (Port: 3000)**:
  + Users upload CSV files via the **Upload Page**, where the system infers column types and allows key selection.
  + The **Lists Page** displays created tables and fetches details from the backend.
  + Small datasets (≤50 rows) are sent directly to the backend; larger datasets (>50 rows) go to the queue.
* **Queuing System (Port: 3003)**:
  + The **DTC Queue** splits large CSV data into batches of 50 rows, stored in batch\_csv\_queue.
  + Batches are processed, published, and consumed before being sent to the backend.
* **Backend (Port: 3002)**:
  + The **DTC Backend** creates tables, inserts records into the **Database**, and sends notifications (e.g., "Created").

**Entity-Relationship Diagram (ERD)**

The ERD defines the database schema for DTG-POC:



* **DataTableDefinition**:
  + id (int, Primary Key): Unique table identifier.
  + tableName (String, Unique Key): Table name.
  + fields (JSON): Column structure.
  + createdAt, updatedAt (DateTime): Timestamps.
* **SchemaLog**:
  + id (int, Primary Key): Log entry identifier.
  + tableName (String): Linked table name.
  + action (String): Action type (e.g., "CREATE").
  + fields (JSON), executedQuery (String), timestamp (DateTime): Log details.
  + **Relationship**: tracks links to DataTableDefinition via tableName.
* **Notification**:
  + id (int, Primary Key): Notification identifier.
  + message, tableName, notificationKey (String, Unique Key), status (String), isRead (Boolean), createdAt (DateTime): Notification details.
  + **Relationship**: notifies links to DataTableDefinition via tableName.

**Services**

The following services drive the DTG-POC application:

* **DTC Frontend (Port: 3000)**: Manages CSV uploads, user approvals, and table management dashboard.
* **DTC Backend (Port: 3002)**: Handles table creation and database interactions.
* **DTC Queue (Port: 3003)**: Processes large datasets via batch queuing.

*Note: Service names are inferred from the DFD; replace with actual names from your .md files if different.*

**Getting Started**

1. Clone the repository:
2. git clone https://github.com/johnsaferio-tsi/DTG-POC.git
3. Navigate to the project directory:
4. cd DTG-POC
5. Install dependencies:
6. npm install
7. Configure the database and environment variables (e.g., .env file).
8. Start the services:
   * Frontend: npm start (Port: 3000)
   * Backend: (Port: 3002)
   * Queue: (Port: 3003)
9. Access the dashboard at http://localhost:3000.

*Note: Update these steps with specific instructions for your setup.*