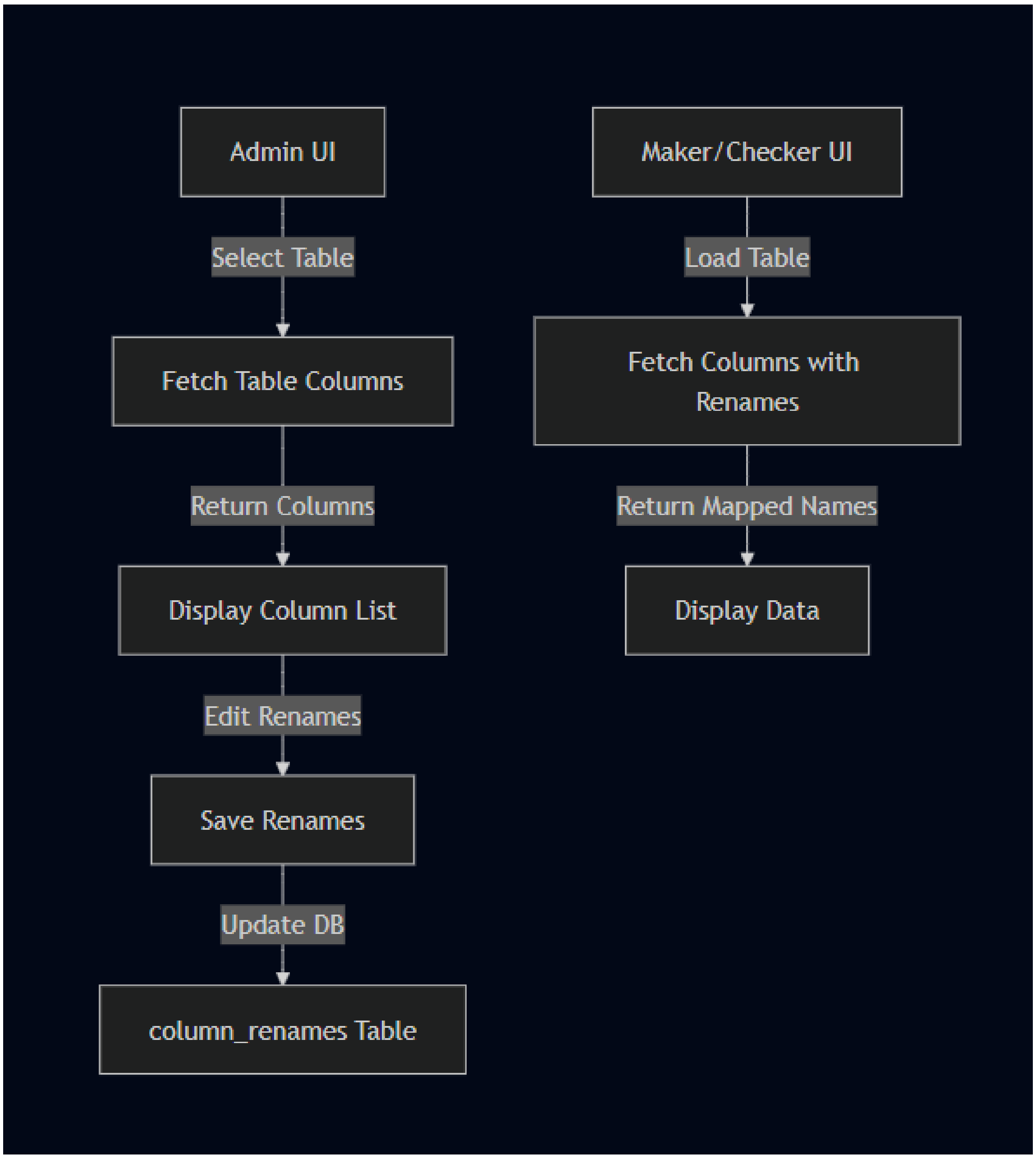
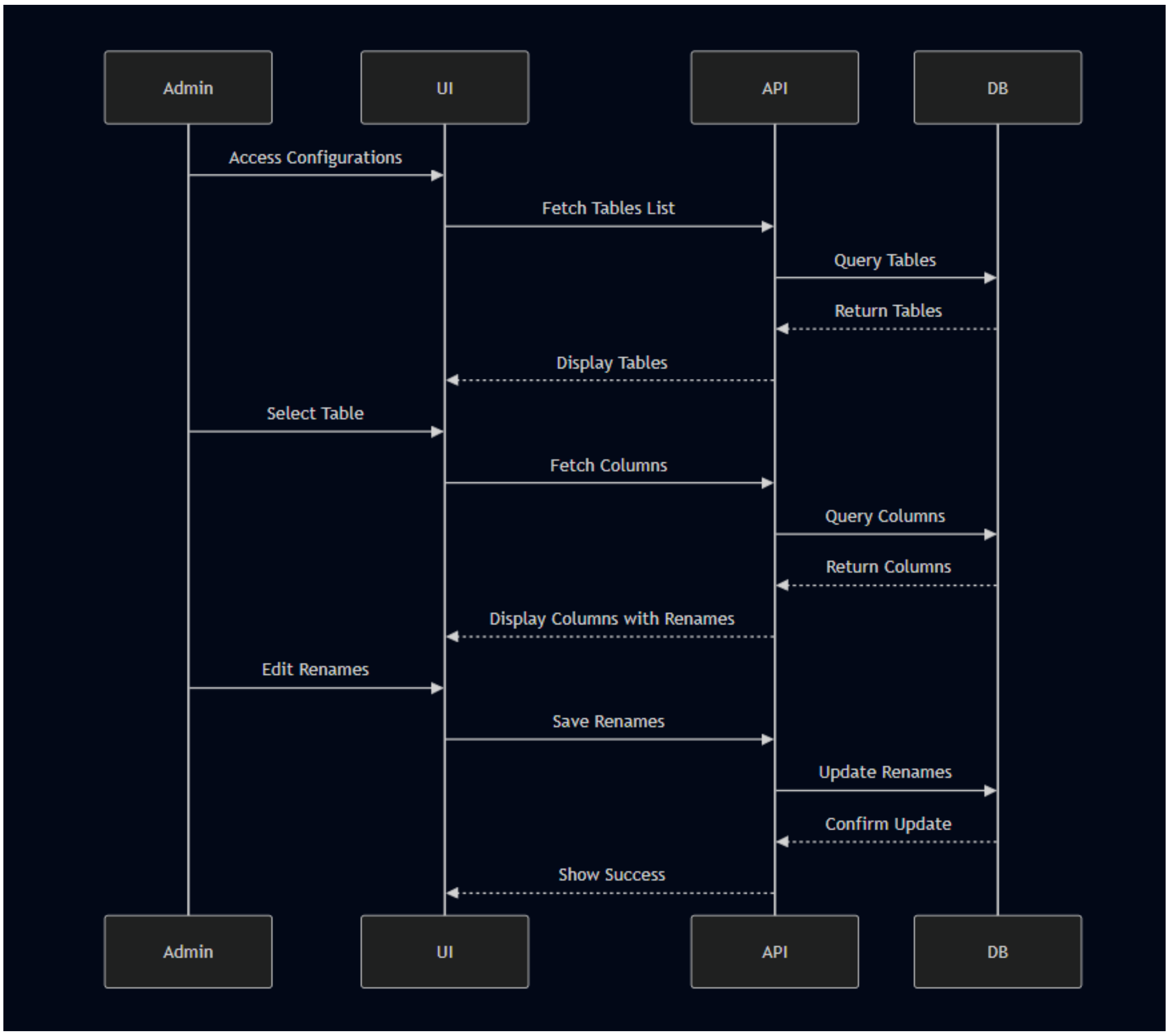
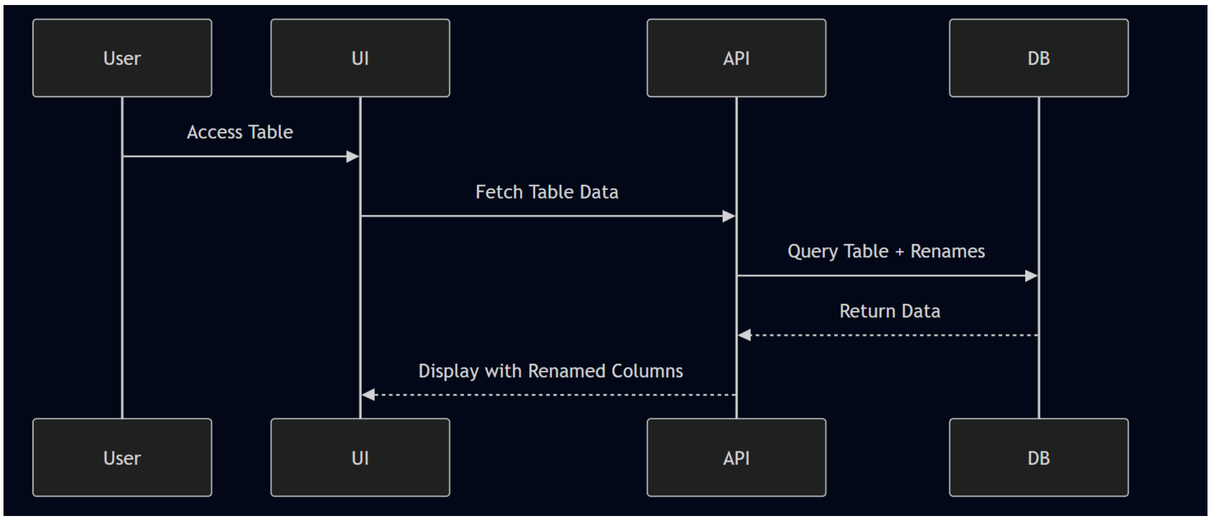
Sundaram Portal – Rename Table Columns Documentation



Admin flow:



Maker/Checker Flow:



1. Database Changes Required

1. Create a new table `column_renames` in the database:
   - `id` (Primary Key)
   - `table_name` (VARCHAR)
   - `original_column_name` (VARCHAR)
   - `renamed_column_name` (VARCHAR)

2. Backend API Endpoints

1. Get Table Columns with Renames
   - Endpoint: `GET /api/admin/column-renames/:tableName`
   - Purpose: Fetch all columns for a table with their rename information
   - Response: List of columns with original names and their renamed versions

2. Create/Update Column Rename
   - Endpoint: `POST /api/admin/column-renames`
   - Purpose: Create or update column rename mapping
   - Request Body:
     - `table_name`
     - `original_column_name`
     - `renamed_column_name`

3. Delete Column Rename
   - Endpoint: `DELETE /api/admin/column-renames/:id`
   - Purpose: Remove a column rename mapping

4. Get All Tables with Renamed Columns
   - Endpoint: `GET /api/admin/column-renames`
   - Purpose: Fetch all tables that have renamed columns

3. Frontend Components

1. Admin Configuration Section
   - New section "Column Renames" in admin configuration
   - Table selection dropdown
   - Column list with rename functionality

2. Column Rename Interface
   - Table showing:
     - Original column name
     - Renamed column name (editable)
     - Actions (Edit/Delete)
   - Add new rename button
   - Save changes button

3. Maker/Checker View Integration
   - Modify existing table components to use renamed columns
   - Add utility function to map original names to renamed names

4. Implementation Flow

1. Admin Side:
   - Admin selects a table from dropdown
   - System fetches all columns for the table
   - Admin can:
     - View existing renames
     - Add new renames
     - Edit existing renames
     - Delete renames
   - Changes are saved to database

2. Maker/Checker Side:

- When fetching table data, system also fetches column renames
- Table headers show renamed columns instead of original names
- All internal operations continue using original column names
- UI displays renamed columns to users

5. Data Flow

1. Data Fetching:

Frontend → Backend API → Database

- Fetch table columns
- Fetch rename mappings
- Combine data for display

2. Data Saving:

Frontend → Backend API → Database

- Validate rename input
- Save/update rename mapping
- Return success/failure

6. Security Considerations

1. Access Control:

- Only admin users can modify column renames
- Maker/Checker users can only view renamed columns

2. Input Validation:

- Validate table names
- Validate column names
- Prevent duplicate renames
- Sanitize input data

7. Error Handling

1. Validation Errors:
   - Invalid table name
   - Invalid column name
   - Duplicate rename
   - Empty rename value

2. System Errors:
   - Database connection issues
   - Invalid table/column references
   - Permission issues