

Project Overview

The Sundaram Portal is a data management system with role-based access control (MAKER-CHECKER-ADMIN architecture) designed for secure data handling and validation. The system implements a multi-step approval process for data modifications with built-in security features and audit trails.

High-Level Architecture Overview

1. Three-Tier Architecture

- Presentation Layer (Frontend)
- Application Layer (Backend Express.js Server)
- Data Layer (PostgreSQL Database)

2. Role-Based Access Control (RBAC)

Three distinct user roles:

- **Maker:** Initiates data changes
- **Checker:** Reviews and approves/rejects changes
- **Admin:** Manages system configuration and user access

Core Roles :-

➤ **Maker**

- Primary data entry/modification role
- Can view and edit tables based on permissions
- Can only edit columns that are configured as editable by admin
- Can see their pending changes highlighted in tables
- Receives notifications about checker approvals/rejections
- Can perform bulk updates through CSV uploads
- Can only edit data through dropdown values for restricted columns

➤ **Checker**

- Reviews and approves/rejects changes made by makers
- Can see all pending change requests
- Must provide comments when rejecting changes
- Can approve/reject changes individually or in bulk
- Can view change history
- Receives notifications about pending approvals
- Can see grouped and ungrouped tables with pending changes

➤ Admin

- **Highest level of system configuration**
- **User Management:**
 - Create/update maker, checker, and admin accounts
 - Disable users (no deletion allowed)
- **Column Configuration:**
 - Define which columns are editable for each table
 - Configure dropdown values for specific columns
- **Group Management:**
 - Create table groups (e.g., "Sales Tables", "HR Tables")
 - Assign tables to groups
 - Manage ungrouped tables

Key Features:-

1. Table Management

- Tables are organized into groups and ungrouped categories
- Each table has configurable column permissions
- Support for dynamic dropdowns in editable columns
- Change tracking for all modifications
- Bulk upload capabilities with validation

2. Change Tracking System - Tracks all changes with before/after states

- Maintains approval status
- Records maker/checker information
- Supports comments for rejections
- Notification system for both makers and checkers

3. Authentication & Security

- JWT-based authentication
- Access token and refresh token mechanism
- Single device login enforcement
- Session management with expiration
- Role-based access control
- Input sanitization and validation

4. Database Structure

- **Separate schemas for application data**
- **Comprehensive tracking tables:**
 - User management (app.users)

- Group management (app.group_table)
- Dropdown configurations (app.dynamic_dropdowns)
- Column permissions (app.column_permission)
- Change tracking (app.change_tracker)
- Row additions (app.add_row_table)
- Session management (app.user_sessions)
- Table metadata (app.table_metadata)

5. Notification System

- Real-time notifications for:
- Pending approvals (for checkers)
- Approval/rejection status (for makers)
- Change request updates
- Notification tracking with seen/unseen status

6. Bulk Operations

- Bulk data upload with validation
- Bulk approval/rejection for checkers
- Transaction management for data integrity
- Error handling and reporting

7. UI Features

- Cell highlighting for pending changes
- Grouped/ungrouped table views
- Dynamic dropdown menus
- Change history views
- Notification indicators

System Documentation:-

1. Authentication Flow

1. User Accesses URL:

- The user hits the login page URL.

2. Email Submission:

- User enters a registered email.
- Backend validates the email format and checks its existence in the database.

3. OTP Generation & Delivery:

- A 6-digit OTP is generated and sent via Gmail SMTP.

- OTP is stored temporarily with a 30-seconds expiry.

4. OTP Verification:

- User enters OTP → backend verifies it against the stored value.
- If valid, proceed; else, reject with an error.

2. Token Management

- **Access Token (JWT):**

- Short-lived (e.g., 5 mins), stored in **HTTP-only cookies**.
- Contains user role (admin, maker, checker) and session ID.

- **Refresh Token:**

- Long-lived (e.g., 20 mins), stored securely in the database.
- Used to generate new access tokens upon expiry.

- **Token Refresh Flow:**

- When the access token expires, the client sends the refresh token to /refresh-token.
- Backend issues a **new access token + new refresh token** (rotation for security).

3. Role-Based Access Control (RBAC)

Role	Permissions
Admin	User management, all access
Maker	Create/Edit records (pending approval)
Checker	Approve/Reject records from Makers

4. Security Features

- **Rate Limiting:**

- Limits OTP requests (e.g., 5 attempts/min) to prevent brute force.

- **SQL Injection/XSS Prevention:**

- Input sanitization + parameterized queries.
- CSP (Content Security Policy) headers.

- **Session Timeout:**
 - 15-minute inactivity → automatic logout + cookie clearance.
- **Cookies:**
 - Secure, HttpOnly, SameSite=Strict flags enabled.

1. User Journey & Data Flow (Low Level Design)

1.1 User Login Process

When a user opens the application:

- **Login Screen**
 - User types in their email (like john.doe@sundaram.com)
 - User enters their password
 - The system also quietly collects information about:
 - * What web browser they're using (like Chrome or Firefox)
 - * What computer system they're on (like Windows or Mac)
 - * What type of device they're using (computer, phone, etc.)
 - * The exact time they're trying to log in
- **What the System Does**
 - Checks if the email and password are correct
 - If successful:
 - * Creates a special pass (access token) that's valid for 5 minutes
 - * Sends back user information (name, role, etc.)
 - If unsuccessful:
 - * Sends back an error message explaining what went wrong

1. Login Screen

// What user sees and enters

```
interface LoginForm {  
    email: string;    // e.g., "john.doe@sundaram.com"  
    password: string; // e.g., "*****"  
}
```

// What frontend sends to backend

```
interface LoginRequest {  
    email: string;  
    password: string;  
    deviceInfo: {  
        browser: string;    // e.g., "Chrome 98.0.4758.102"  
        os: string;         // e.g., "Windows 10"  
        deviceType: string; // e.g., "Desktop"  
        timestamp: string;  // e.g., "2024-03-20T10:30:00Z"  
    }  
}
```

// What backend sends back

```
interface LoginResponse {  
    success: boolean;  
    accessToken: string;    // JWT token valid for 5 minutes  
    user: {  
        userId: string;  
        firstName: string;  
        lastName: string;  
        role: "MAKER" | "CHECKER" | "ADMIN";  
        email: string;  
    };  
    message?: string;    // Error message if login fails  
}
```

2. Session Creation

- Backend creates new session in `app.user_sessions`
- Terminates any existing sessions for the user
- Sets HTTP-only cookie with refresh token

1.2 Maker Dashboard

- **Initial View**

- Shows all tables the user can access
- Tables are organized in two ways:
 - * Grouped tables (organized in categories)
 - * Ungrouped tables (standing alone)
- For each table, shows:
 - * Table name
 - * Description
 - * When it was last changed
 - * How many changes are waiting for approval

- **When Viewing a Table**

- User can choose how many rows to see at once
- Can filter data (like showing only certain dates)
- Can sort data (like A-Z or newest first)
- System shows which columns can be edited
- Shows any dropdown menus available

Table Selection Process

1. Initial Load

// **Frontend request**

GET /api/tables/available

// Backend response

```
interface TableListResponse {  
  grouped: {  
    [groupName: string]: {  
      groupId: string;  
      tables: Array<{  
        tableName: string;  
        displayName: string;  
        description: string;  
        lastModified: string;  
        pendingChanges: number;  
      }>  
    }  
  };  
  ungrouped: Array<{  
    tableName: string;  
    displayName: string;  
    description: string;  
    lastModified: string;  
    pendingChanges: number;  
  }>;  
}
```

2. Table Data Loading

// Frontend request

```
interface TableDataRequest {  
  tableName: string;  
  page: number;  
  pageSize: number;  
  filters?: {  
    [columnName: string]: string | number | boolean;  
  }  
}
```



```
};

sort?: {
    column: string;
    direction: "ASC" | "DESC";
};
}
```

// Backend response

```
interface TableDataResponse {
    data: Array<Record<string, any>>;
    totalRows: number;
    editableColumns: Array<{
        name: string;
        type: string;
        hasDropdown: boolean;
        dropdownValues?: string[];
    }>;
    primaryKeyColumn: string;
}
...
```

1.3 Data Editing Process

- Single Cell Edit
 - When a maker changes one piece of data:
 - * System records the table name
 - * Records which row was changed
 - * Saves what the old value was
 - * Saves what the new value is
 - * Records when the change was made
 - * Creates a unique request ID for tracking

- Bulk Edit
 - When changing multiple items at once:
 - * Can update many rows together
 - * System checks each change for errors
 - * Provides a summary of:
 - How many changes were attempted
 - How many were successful
 - How many failed
 - Any error messages

- **Single Cell Edit**

// When maker clicks to edit a cell

```
interface CellEditRequest {
  tableName: string;
  rowId: string;
  column: string;
  oldValue: any;
  newValue: any;
  timestamp: string;
}
```

// Backend creates entry in change_tracker

```
interface ChangeTrackerEntry {
  requestId: string;
  tableName: string;
  rowId: string;
  oldData: {
    [column: string]: any;
  };
};
```

```
newData: {  
  [column: string]: any;  
};  
maker: string;  
status: "PENDING";  
created_at: string;  
}
```

// Backend response

```
interface CellEditResponse {  
  success: boolean;  
  requestId: string;  
  message: string;  
}
```

• Bulk Edit Process

// Frontend bulk edit request

```
interface BulkEditRequest {  
  tableName: string;  
  changes: Array<{  
    rowId: string;  
    updates: {  
      [column: string]: any;  
    };  
  }>;  
}
```

// Backend response

```
interface BulkEditResponse {  
  success: boolean;  
  results: Array<{
```

```

rowId: string;
requestId: string;
status: "PENDING" | "ERROR";
error?: string;
}>;
summary: {
total: number;
successful: number;
failed: number;
};
}

```

1.4 Checker Workflow

- **Seeing Pending Changes**

- Checkers can see:

- * All changes waiting for approval
- * Who made each change
- * When changes were made
- * What exactly was changed
- * Summary of how many changes need review

- **Pending Changes View**

// Frontend request for pending changes

GET /api/checker/pending-changes

// Backend response

```

interface PendingChangesResponse {
changes: Array<{

```

```
requestId: string;
tableName: string;
displayName: string;
maker: {
  name: string;
  email: string;
};
timestamp: string;
changes: Array<{
  column: string;
  oldValue: any;
  newValue: any;
}>;
status: "PENDING";
};
summary: {
  totalPending: number;
  byTable: {
    [tableName: string]: number;
  };
};
}
```

1.5 Checker Review Process

- **Reviewing Changes**
 - When looking at a specific change:
 - * Can see the complete before and after
 - * Sees who made the change and when
 - * Views the history of that item
 - * Can approve or reject with comments

- **Reviewing Changes**

// When checker views a specific change

```
interface ChangeDetailRequest {
  requestId: string;
}

interface ChangeDetailResponse {
  requestId: string;
  tableName: string;
  rowDetails: {
    before: Record<string, any>;           // Complete row data before changes
    after: Record<string, any>;           // Complete row data after changes
    changedFields: string[];              // List of modified columns
  };
  maker: {
    name: string;
    email: string;
    timestamp: string;
  };
  auditTrail: Array<{
    action: string;
    timestamp: string;
    user: string;
    details: string;
  }>;
}
```

// Approval/Rejection Request

```
interface CheckerDecisionRequest {  
    requestId: string;  
    action: "APPROVE" | "REJECT";  
    comments?: string; // Required for rejection  
    timestamp: string;  
}  
  
interface CheckerDecisionResponse {  
    success: boolean;  
    message: string;  
    updatedStatus: string;  
    notificationSent: boolean;  
}
```

1.6 Admin Configuration Workflows

User Management

- **Managing Users**
 - Creating new users:
 - * Set up email and name
 - * Assign role (Maker/Checker/Admin)
 - * Set what they can access
 - * System creates temporary password
 - Updating users:
 - * Can change names
 - * Can change roles
 - * Can turn accounts on/off
 - * Can modify what they can access

// Creating new user

```
interface CreateUserRequest {  
    email: string;  
    firstName: string;  
    lastName: string;  
    role: "MAKER" | "CHECKER" | "ADMIN";  
    permissions?: {  
        tableAccess?: string[];  
        groupAccess?: string[];  
    }; };
```

// Response

```
interface CreateUserResponse {  
    success: boolean;  
    userId: string;  
    temporaryPassword: string;    // One-time password  
    message: string;  
}
```

// Updating user

```
interface UpdateUserRequest {  
    userId: string;  
    updates: {  
        firstName?: string;  
        lastName?: string;  
        role?: string;  
        active?: boolean;  
        permissions?: {  
            tableAccess?: string[];  
            groupAccess?: string[];  
        }; };
```


Column Permission Configuration

- **Setting Up Permissions**

- For each table:
 - * Choose which columns can be edited
 - * Set up dropdown menus
 - * Add validation rules (like minimum/maximum values)
 - * Make certain fields required

// Setting column permissions

```
interface ColumnPermissionRequest {  
    tableName: string;  
    columns: Array<{  
        name: string;  
        editable: boolean;  
        dropdownEnabled: boolean;  
        validations?: {  
            type: string;  
            min?: number;  
            max?: number;  
            pattern?: string;  
            required?: boolean;  
        }; }>; };
```

// Database storage in app.column_permission

```
{  
    "table_name": "customer_data",  
    "column_list": {  
        "customer_name": {  
            "editable": true,  
            "dropdown_enabled": false,
```

```

"validations": {
  "type": "string",
  "required": true
}},
"status": {
  "editable": true,
  "dropdown_enabled": true,
  "dropdown_values": ["Active", "Inactive", "Pending"]
}}

```

Group Configuration

- **Creating Groups**
 - Organize tables into categories
 - Give groups names and descriptions
 - Decide order of tables within groups
 - Track who made changes and when

// Creating/Updating groups

```

interface GroupConfigRequest {
  groupName: string;
  description?: string;
  tables: Array<{
    tableName: string;
    displayOrder: number;
  }>;
}

```

// Database storage in app.group_table

```

{
  "group_name": "Sales Data",
  "table_list": {
    "tables": [

```

```
{  
  "name": "sales_transactions",  
  "order": 1  
},  
{  
  "name": "customer_accounts",  
  "order": 2  
}  
],  
"metadata": {  
  "created_by": "admin@sundaram.com",  
  "last_updated": "2024-03-20T10:30:00Z"  
}}}
```

1.7 Notification System

- How Notifications Work
 - Different types:
 - * Change requests
 - * Approvals
 - * Rejections
 - * System messages
 - Each notification includes:
 - * What happened
 - * When it happened
 - * How important it is
 - * Any actions needed

- **Real-time Notifications**

// WebSocket message structure

```
interface NotificationMessage {  
  type: "CHANGE_REQUEST" | "APPROVAL" | "REJECTION" | "SYSTEM";  
  timestamp: string;  
  data: {  
    requestId?: string;  
    message: string;  
    priority: "HIGH" | "MEDIUM" | "LOW";  
    actionRequired?: boolean;  
    link?: string;  
  };  
  recipient: {  
    userId: string;  
    role: string;  
  }; };
```

// Notification storage in database

```
interface NotificationRecord {  
  id: string;  
  userId: string;  
  message: string;  
  type: string;  
  read: boolean;  
  created_at: string;  
  expires_at?: string;  
  related_request?: string;  
}
```

1.8 Error Handling & Validation

- **Checking for Mistakes**

- Forms check for:

- * Required fields
- * Correct length
- * Valid formats
- * Custom rules

- **Error Messages**

- Clear explanations of what went wrong

- Different types of errors:

- * Login problems
- * Invalid data
- * Missing information
- * Permission issues

- **Frontend Validation**

// Form validation rules

```
interface ValidationRules {  
  required?: boolean;  
  minLength?: number;  
  maxLength?: number;  
  pattern?: RegExp;  
  custom?: (value: any) => boolean;  
}
```

// Error message structure

```
interface ValidationError {  
  field: string;
```

```
message: string;
code: string;
details?: any;
}
```

// Backend Error Responses

```
interface ErrorResponse {
  success: false;
  error: {
    code: string; // e.g., "AUTH001", "VAL002"
    message: string; // User-friendly message
    details?: any; // Technical details (dev mode only)
    timestamp: string;
  };
  requestId?: string; // For tracking/debugging
}
```

// Common error codes

```
const ERROR_CODES = {
  AUTH001: "Invalid credentials",
  AUTH002: "Session expired",
  VAL001: "Invalid input data",
  VAL002: "Required field missing",
  DB001: "Database operation failed",
  PERM001: "Insufficient permissions"
};
```

1.9 Audit Trail

- **What's Recorded**

- Every change is logged with:

- * Who made it
 - * When it happened
 - * What was changed
 - * From where it was done
 - * Complete before and after

- **Tracking Changes**

// Every change is logged

```
interface AuditLogEntry {  
    id: string;  
    timestamp: string;  
    userId: string;  
    action: string;  
    tableName: string;  
    rowId?: string;  
    oldValue?: string;  
    newValue?: string;  
    requestId?: string;  
    ipAddress: string;  
    userAgent: string;  
}
```

// Sample audit query response

```
interface AuditTrailResponse {  
    entries: Array<AuditLogEntry>;  
    pagination: {  
        currentPage: number;
```

```

totalPages: number;
totalEntries: number;
};
filters?: {
dateRange?: [string, string];
users?: string[];
actions?: string[];
}; }

```

1.10 Data Export/Import

- **Exporting Data**

- Can get data in different formats:
 - * Excel
 - * CSV (like Excel but simpler)
 - * JSON (for technical use)
- Can choose what to include

- **Importing Data**

- System checks all data before accepting
- Shows summary of:
 - * How many rows are valid
 - * Any errors found
 - * Preview of data

- **Export Format**

```

interface ExportRequest {
tableName: string;
format: "CSV" | "EXCEL" | "JSON";
filters?: Record<string, any>;
columns?: string[];
}

```

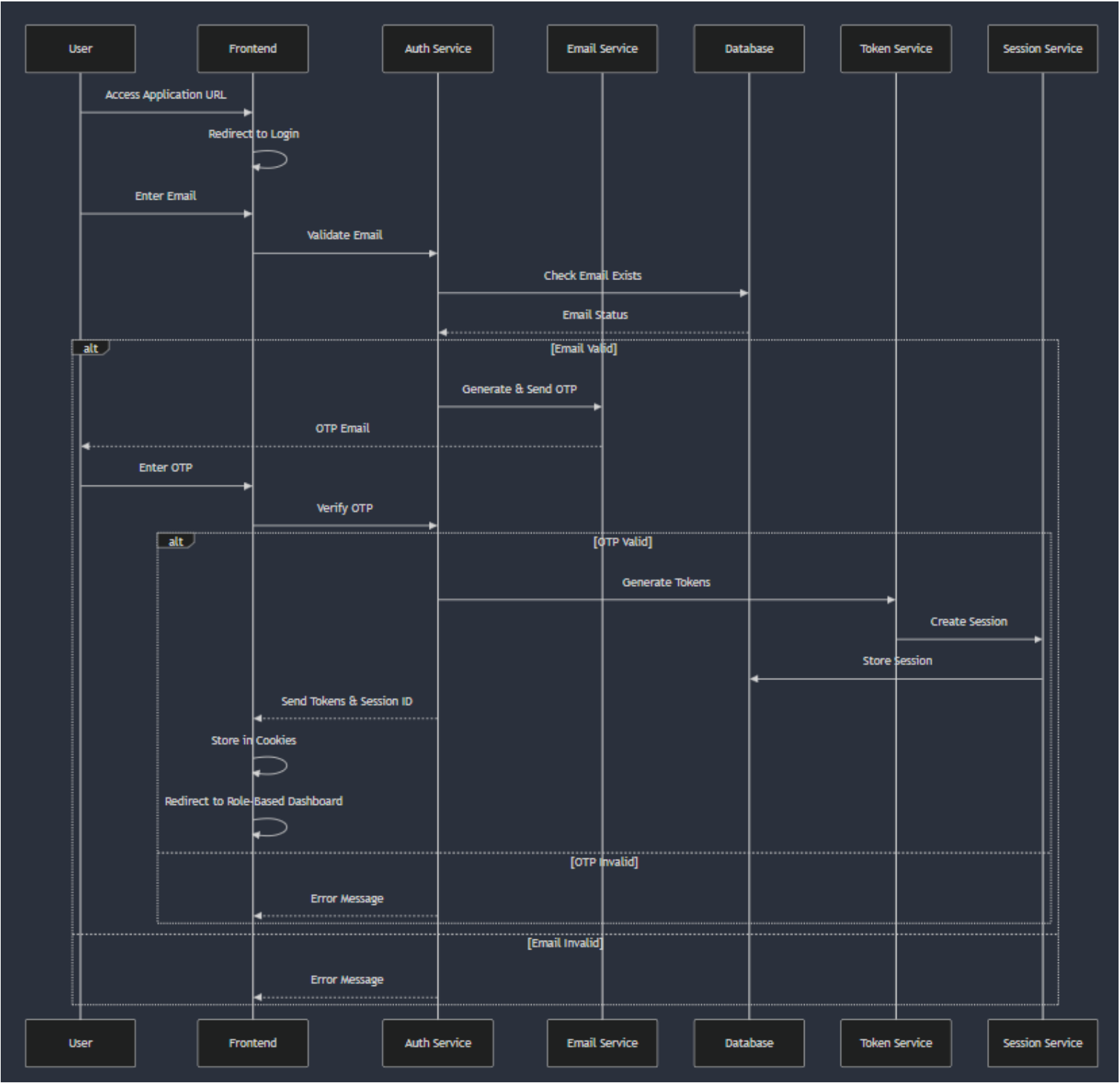

// Export file structure (CSV example)

```
"ID","Name","Status","Last Modified","Modified By"
"001","John Doe","Active","2024-03-20","admin@sundaram.com"
"002","Jane Smith","Pending","2024-03-19",maker1@sundaram.com
```

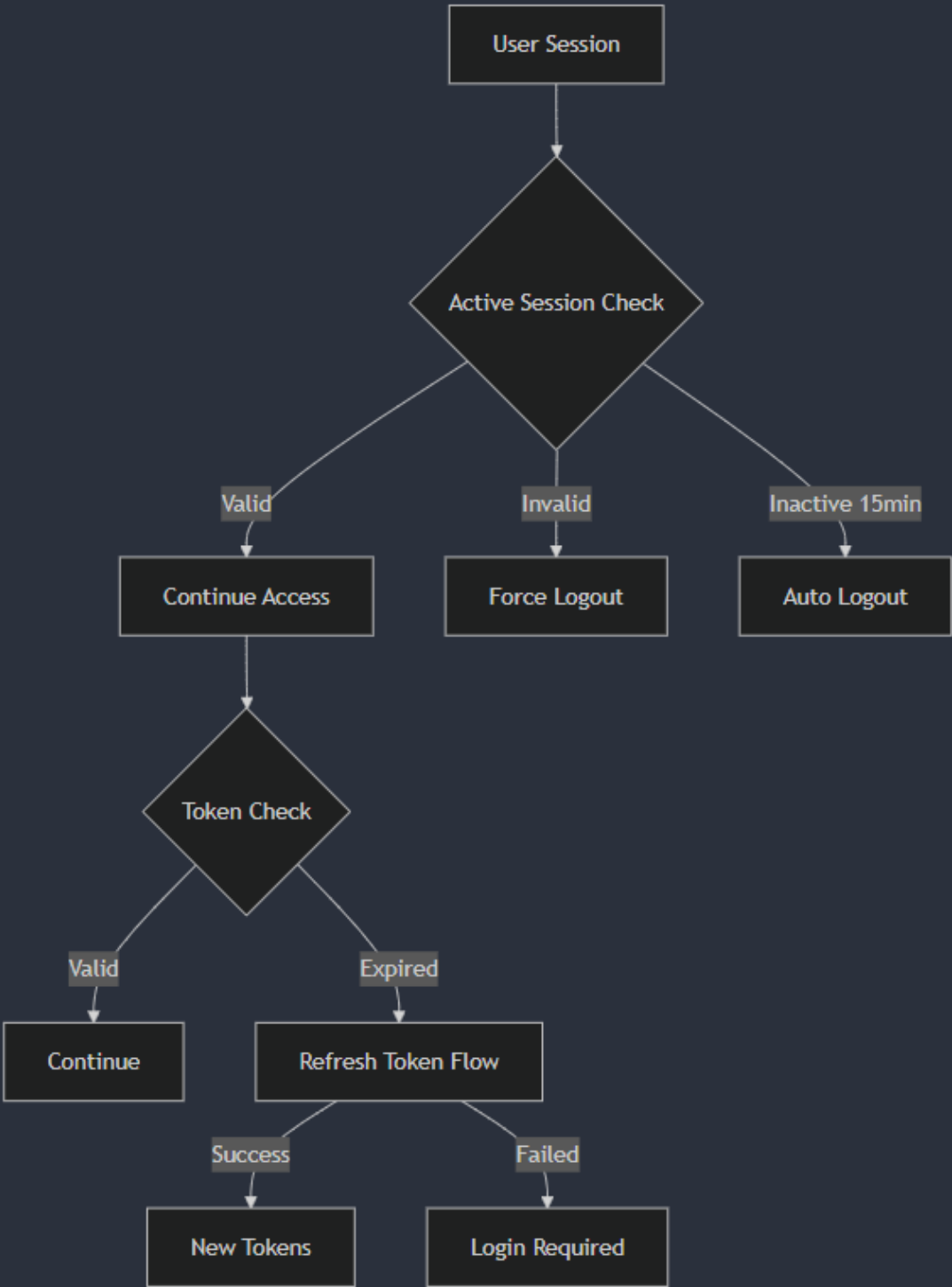
// Import Validation

```
interface ImportValidationResult {
    valid: boolean;
    errors: Array<{
        row: number;
        column: string;
        value: any;
        error: string;
    }>;
    summary: {
        totalRows: number;
        validRows: number;
        errorRows: number;
    };
    preview?: Array<Record<string, any>>;
}
```

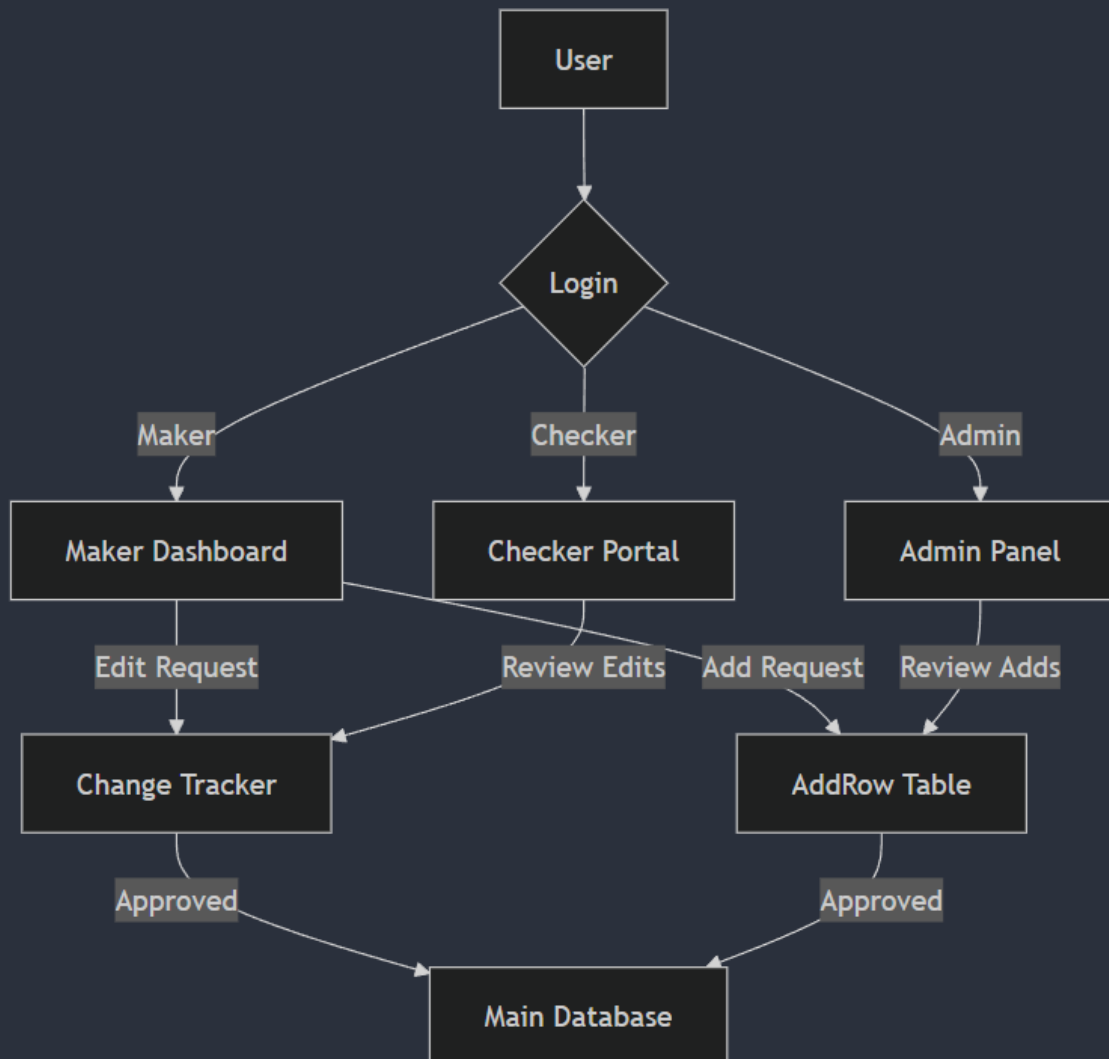
Login Working flow



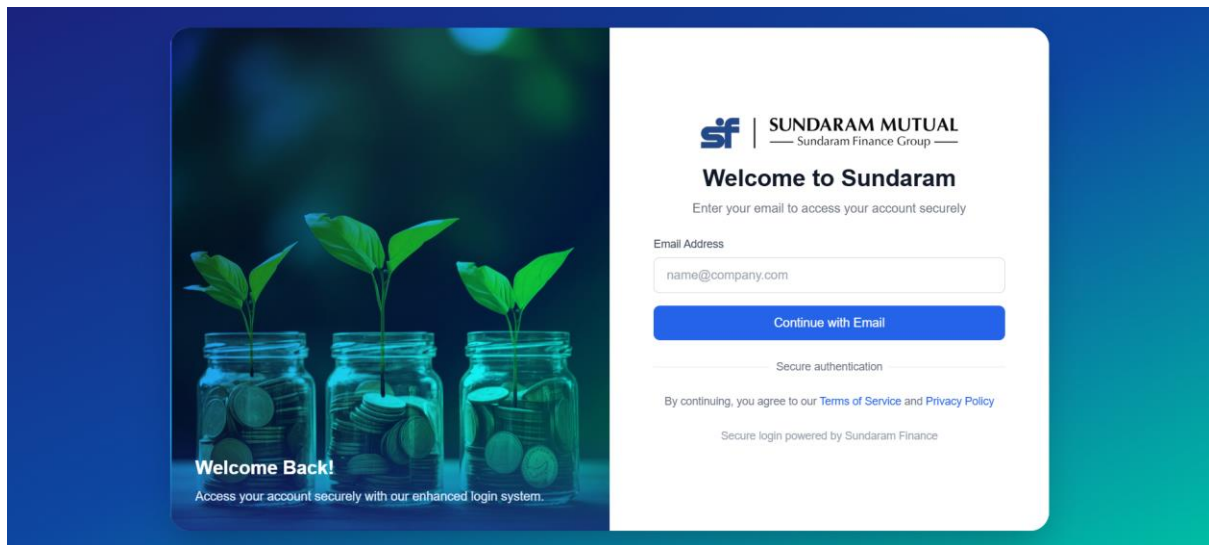
Session management working flow



Edit and Add row requests Flow



Login Page UI Interface



The login page features a dark blue background with a teal gradient on the right. On the left, there is a vertical image of three glass jars containing coins and small green plants growing from them. The text 'Welcome Back!' is overlaid on this image, followed by the subtitle 'Access your account securely with our enhanced login system.'.

Sf | SUNDARAM MUTUAL
— Sundaram Finance Group —

Welcome to Sundaram
Enter your email to access your account securely

Email Address

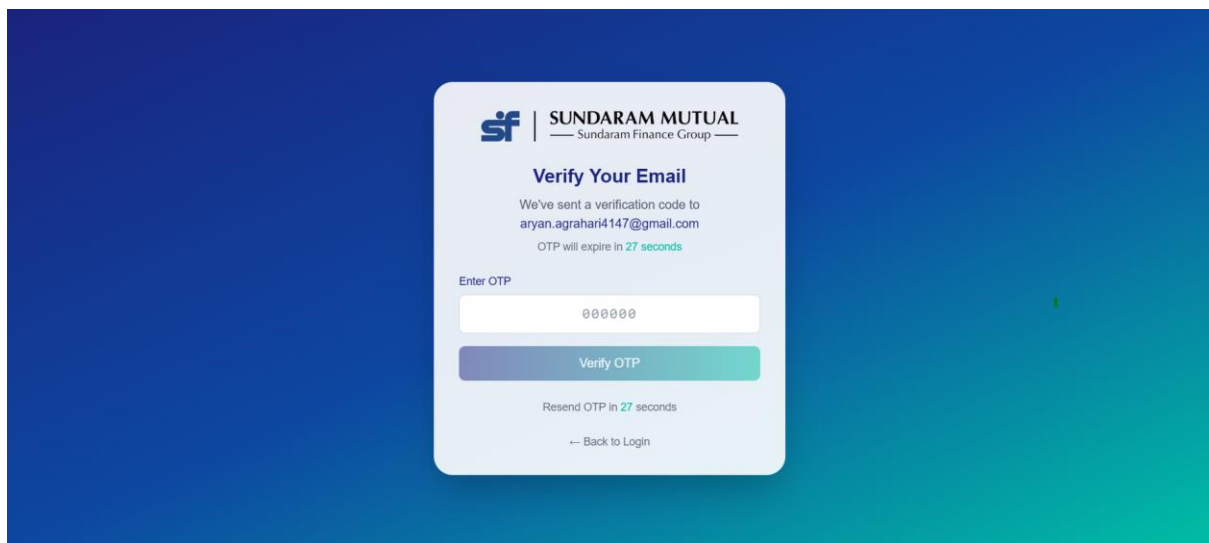
[Continue with Email](#)

Secure authentication

By continuing, you agree to our [Terms of Service](#) and [Privacy Policy](#)

Secure login powered by Sundaram Finance

Login page:- Enter registered email



The verification page has a dark blue background with a teal gradient on the right. It features a light blue rounded rectangle containing the Sundaram Mutual logo and the title 'Verify Your Email'. The text indicates a verification code has been sent to 'aryan.agrahari4147@gmail.com' and that the OTP will expire in 27 seconds. Below this is a text input field for the OTP, a 'Verify OTP' button, and a 'Resend OTP' link. At the bottom is a 'Back to Login' link.

Sf | SUNDARAM MUTUAL
— Sundaram Finance Group —

Verify Your Email
We've sent a verification code to
aryan.agrahari4147@gmail.com
OTP will expire in 27 seconds

Enter OTP

[Verify OTP](#)

[Resend OTP in 27 seconds](#)

[← Back to Login](#)

Login page:- Enter OTP

MAKER

1. Maker Role

- View database tables
- Edit existing rows (requires Checker approval)
- Add new rows (requires Admin approval)
- View change history
- Track request status

Features Breakdown:-

1. Table Management

- Group-based organization
- Ungrouped tables support
- Dynamic column permissions
- Configurable dropdown values

2. Change Management

- Cell-level tracking
- Change history
- Status tracking
- Notification system

3. Approval Workflow

- Two-step verification
- Comment system
- Bulk approval support

4. User Interface

- Grouped/Ungrouped view
- Highlighted changed cells
- Status indicators
- Notification alerts

Database Structure:-

Main Tables

- app.users - User management
- app.change_tracker - Edit request tracking
- app.add_row_table - New row requests
- app.column_permission - Column-level permissions
- app.group_table - Table grouping
- app.dynamic_dropdowns - Dropdown configurations

Error Handling

1. Frontend

- Input validation errors
- Network error handling
- User feedback system

2. Backend

- Transaction management
- Error logging
- Rollback mechanisms
- Status codes

Interface:-

➤ **Initial View**

- Default shows grouped tables
- Toggle between grouped/ungrouped
- Search functionality available

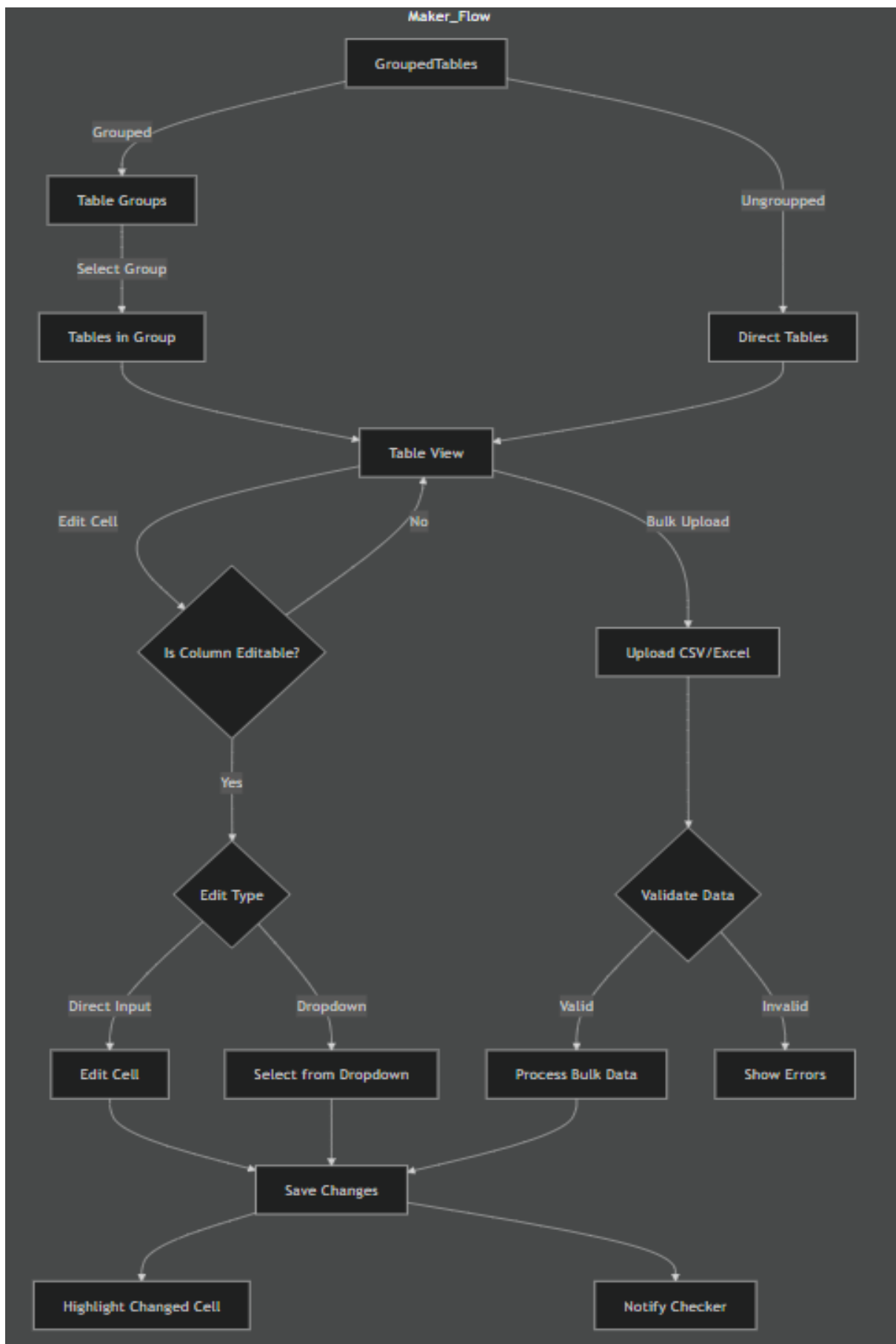
➤ **Group Selection** (if in grouped view)

- List of available groups
- Group description and table count
- Quick search within groups

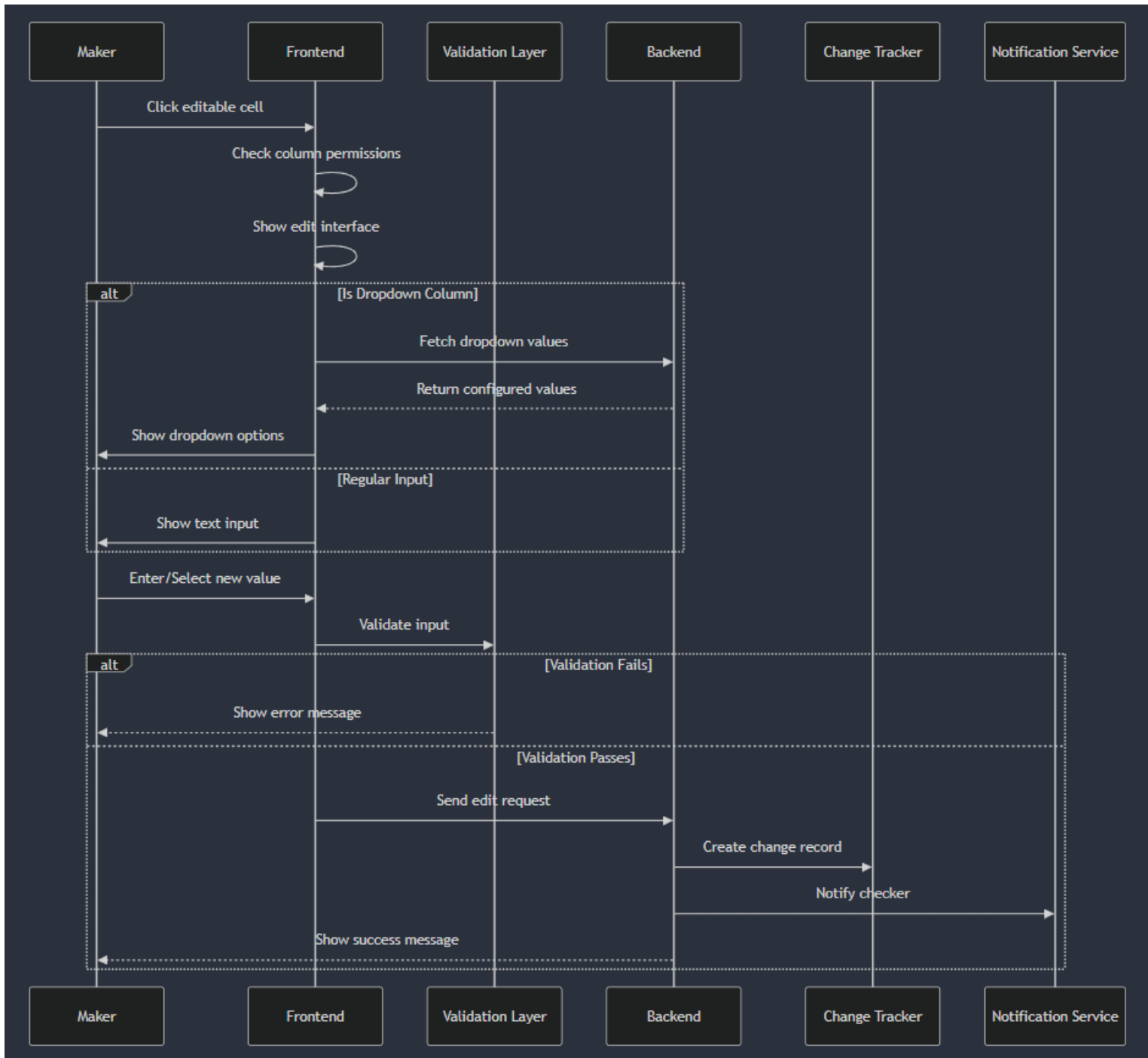
➤ **Table Selection**

- Display table name and description
- Show last modified date
- Indicate pending changes if any

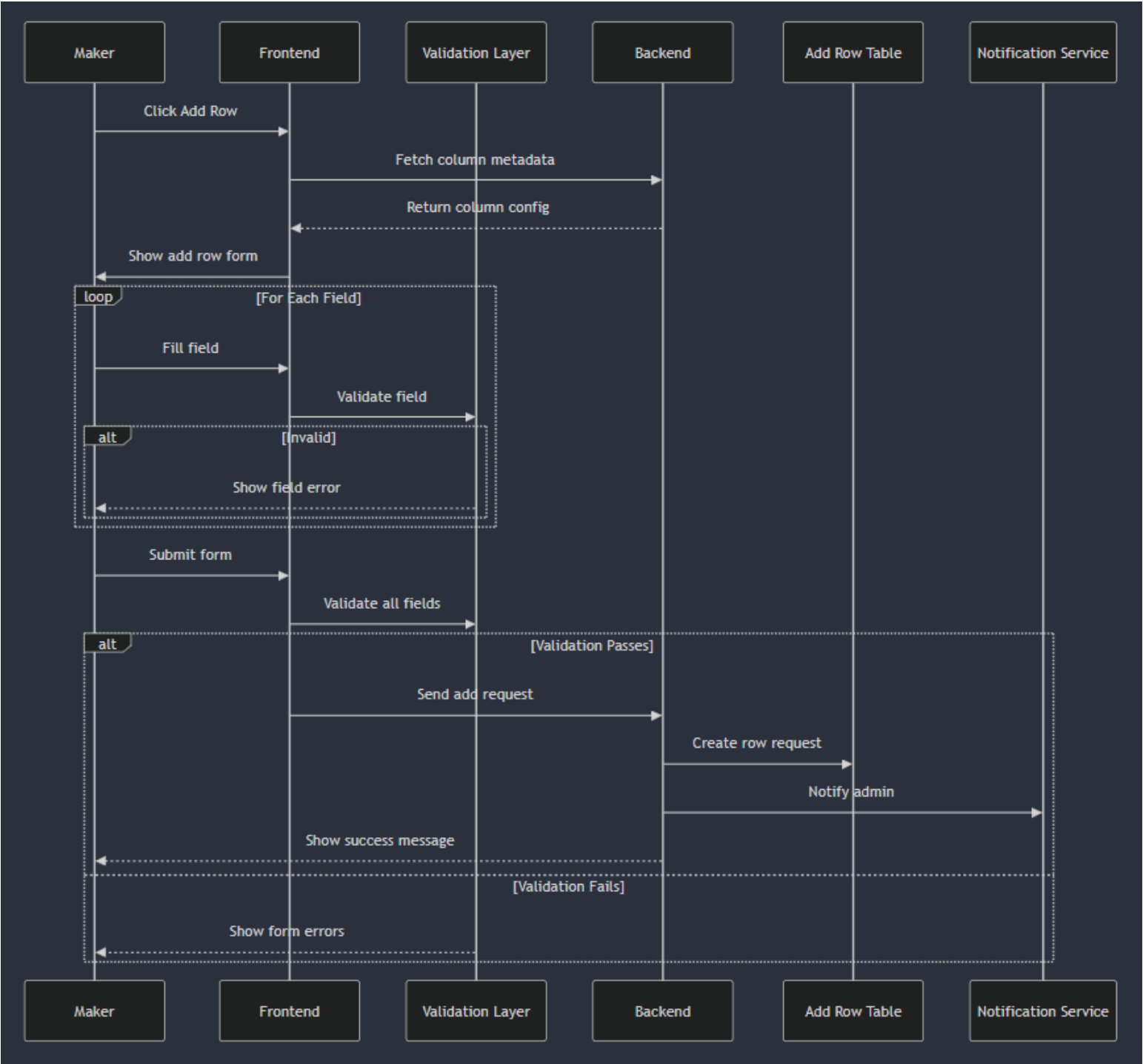
Maker flow



Edit Row Working Flow



Add row working Flow



Maker's UI Interface

Hello, **maker-**

Access powerful tools and insights to manage your operations efficiently and effectively



Data Management

Access and manage fund data with ease



Maker: Maker Dashboard Page

← Home / Data Management

All Groups ▾

Search table here...

ria_master



dim_branch



dim

employee data



stg_call_schememaster_new_date_5



target_branch



stg_offline_point_of_sales



target_region



stg_online_point_of_sales



rm_weightage



change_tracker



add_row_table



stg_stg_hr_master



target_rm



stg_pan_based_rule



targetrm



Maker: View Tables – All tables

← Home / Data Management

All Groups ▾

Search table here...

ria_master →	dim_branch →	dim → employee data →
stg_call_schememaster_new_date_5 →	target_branch →	stg_offline_point_of_sales →
target_region →	stg_online_point_of_sales →	rm_weightage →
change_tracker →	add_row_table →	stg_stg_hr_master →
target_rm →	stg_pan_based_rule →	targetrm →

← Page 1 of 4 →

Maker: View Tables – Groups selection

← Home / Data Management / dim

dim ▾

Search table here...

dim_branch → dim	dim_rm_broker → dim	dim_org → dim
dim_product → dim		

Maker: View table – Grouped View

[← Back to Tables](#)

Q Search in all columns...

Search

Download Template

Upload CSV

Action	ria_master_sk	riacode	name	arn
	1	INA000012351	JM FINANCIAL SERVICES LTD	ARN-0002
	2	INA000009214	ADITYA BIRLA MONEY LIMITED	ARN-0003
	3	INA000000920	HDFC BANK LTD	ARN-0005
	4	INA000001589	HDFC PROPERTY VENTURES LIMITED	ARN-0005
	5	INA100001398	BAJAJ CAPITAL INVESTMENT ADVISER PVT LTD	ARN-0010

Rows per page: 10

Page 1 of 180 < 1 2 ... 180 >



Maker: View Table Data

[← Back to Tables](#)

Q Search in all columns...

Search

Action	dim_branch_sk	branch_code	branch_name	branch_address1	branch_address2
	4663764				
	1234	test01	test01	katniii	test01
	456768	akki55	mumbai	mumbai	mumbai
	444326	3		fdsaajafghfghj	
	213487			hsdafsd	

Rows per page: 10

Add Row

Only editable fields can be changed

Dim Branch Sk

Branch Code

Branch Name

Branch Address1

Branch Address2

Branch Address3

Branch City

Maker: Add row data form

SfSUNDARAM MUTUALSundaram Finance Group

← Back to Tables

Q Search in all columns...

Search

Action	dim_branch_sk	branch_code	branch_name	branch_address1	branch_address2
	4663764				
	1234	test01	test01	katniii	test01
	456768	akki55	mumbai	mumbai	mumbai
	444326	3		fdsaajafghfghj	
	213487			hsdafsd	

Rows per page: 10

Edit Row

Only editable fields can be changed

Branch Category

Created By

Modified By

Created On

dd-mm-yyyy

Modified On

dd-mm-yyyy

Row Id

Cancel

Save Changes

Maker: Edit row data form

SfSUNDARAM MUTUALSundaram Finance Group

← Back to Tables

Q Search in all columns...

Search

Download Template

Upload CSV

Action	dim_branch_sk	branch_code	branch_name	branch_address1	branch_address2	branch_address3	branch_c
	4663764						
	1234	test01	test01	katniii	test01	test01	test01
	456768	akki55	mumbai	mumbai	mumbai	mumbai	mumbai
	444326	3		fdsaajafghfghj			
	213487			hsdafsd			

Rows per page: 10

Success

Changes requested successfully

Maker: Highlighted cell and success toast after edit

← Back to Tables

🔍 Search in all columns...

Search

📄 Download Template

📤 Upload CSV

Action	dim_branch_sk	branch_code	branch_name	branch_address1	branch_address2	branch_address3
	2	ab40	agra	k	3492ndfloornicholsonroad	adjacentko
	2	ab40	agra	k	3492ndfloornicholsonroad	adjacentko
	4	ac99	agra	3	ldgno.7kernagar	bandra
	5	ad99	aandadukaanfinancialservicesprivat	n	mumbai	mumbai
	6	ae99	americanexpressbank			

Filter Type

Contains ▾

Filter Value

Enter value...

Clear

Apply

Rows per page: 10 ▾

Page 1 of 57 < 1 2 ... 57 >

Maker: Sorting and Filtering application

← Home / Data Management

All Group

ria_master	→	dim_branch	→	dim
stg_call_schememaster_new_date_5	→	target_branch	→	stg_of
target_region	→	stg_online_point_of_sales	→	rm_wa
change_tracker	→	add_row_table	→	stg_st
target_rm	→	stg_pan_based_rule	→	target

← Page 1 of 4 →

Notifications

Unread (1)

Approved (94)

Rejected (148)

🚫 Rejected

Mar 24, 25, 09:27 PM

Update in dim employee

Rejector: aryan.agrahan.5266@gmail.com

🚫 Rejection Comment

ghff

3/24/2025

View Changes

Maker: Notification Drawer

Notifications

Unread (1)

Approved (94)

Rejected (148)

Rejected

Mar 24, 25, 09:27 PM

Update in dim employee

Rejector: aryan.agrahari.5266@gmail.com

Rejection Comment

ghff

3/24/2025

Hide Changes

Previous Value	New Value
code:	code:
3192788	\$%^&*

Notifications

Unread (1)

Approved (94)

Rejected (148)

From Date

dd-mm-yyyy

To Date

dd-mm-yyyy

Approved

Mar 11, 25, 06:11 PM

Update in dim employee

Approver: aryan.agrahari.5266@gmail.com

Hide Changes

Previous Value	New Value
period_from:	period_from:
-	12-08-2024

Approved

Mar 11, 25, 05:40 PM

Update in dim employee

Approver: aryan.agrahari.5266@gmail.com

View Changes

Approved

Mar 11, 25, 05:38 PM

Update in dim employee

Approver: aryan.agrahari.5266@gmail.com

Notifications

Unread (1)

Approved (94)

Rejected (148)

From Date

dd-mm-yyyy

To Date

dd-mm-yyyy

Rejected

Mar 24, 25, 09:27 PM

Update in dim employee

Rejector: aryan.agrahari.5266@gmail.com

Rejection Comment

ghff

3/24/2025

Hide Changes

Previous Value	New Value
code:	code:
3192788	\$%^&*

Rejected

Mar 11, 25, 04:54 PM

Update in dim employee

Rejector: aryan.agrahari.5266@gmail.com

Rejection Comment

kk

3/11/2025

View Changes

Notification Drawer:- Different sections

Checker

The Checker role is a critical part of the maker-checker system, responsible for reviewing and approving/rejecting changes made by makers to ensure data integrity and maintain a four-eyes principle in data modifications.

Core Features

1. Change Request Management

Pending Request Viewing

- Checkers can view all pending change requests through the /fetchCheckerRequest endpoint
- Requests are displayed in chronological order (newest first)
- Each request contains:
 - Table name
 - Old data
 - New data
 - Maker's information
 - Request timestamp
 - Request ID
 - Row ID

Group-Based Request Organization

- Requests are organized by table groups via /fetchCheckerGroupRequest
- Groups are configured by admins
- Two main views:
 - Grouped tables view
 - Ungrouped tables view
- Each group shows:
 - Total pending requests count
 - List of tables with changes
 - Last update timestamp

2. Request Review Capabilities

Individual Request Review - Review workflow

1. Checker views change details
2. Compares old and new values
3. Can approve or reject with comments
4. System tracks the checker's ID and timestamp

Bulk Operations

- Mass approval via approveAll endpoint
- Mass rejection via allReject endpoint
- Transaction management ensures data consistency
- Partial success handling (some approved, some failed)

3. Notification System

Real-time Notifications

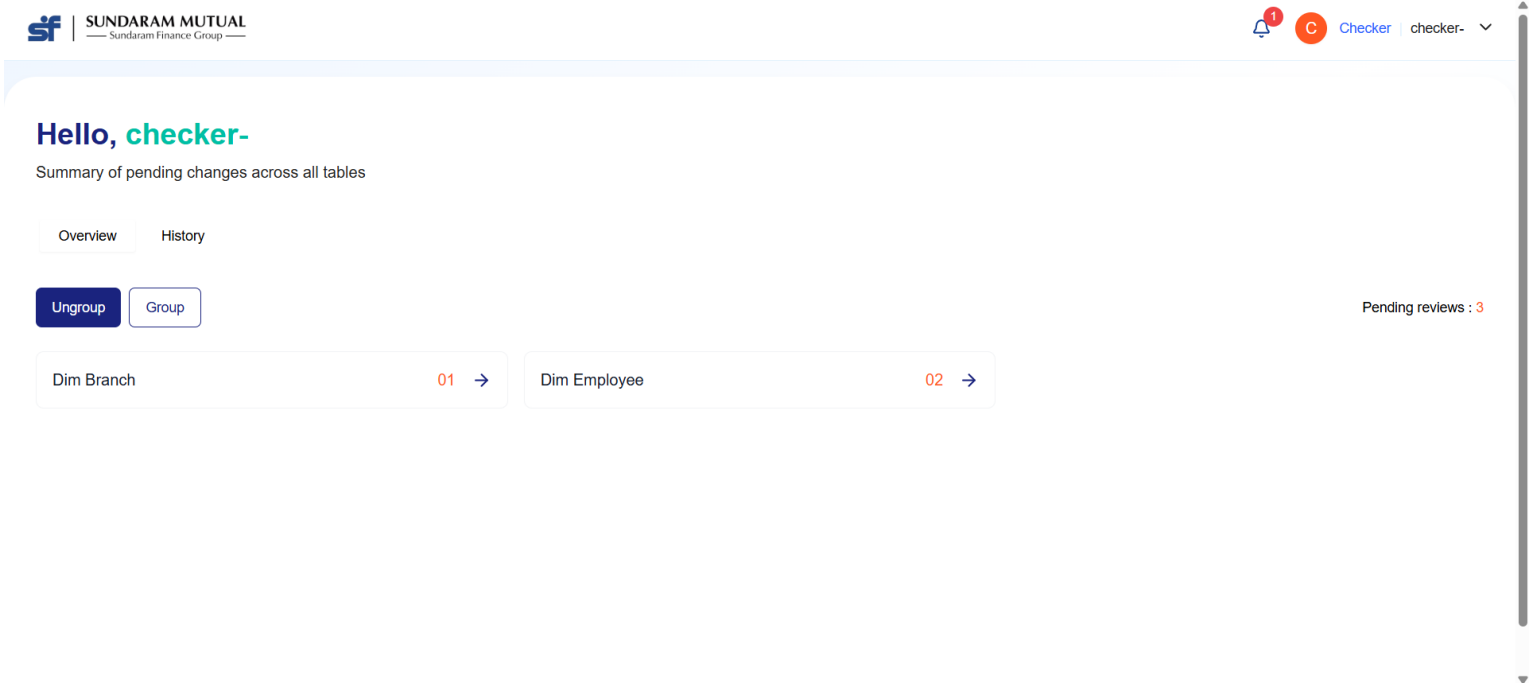
- Checkers receive notifications for:
- New change requests
- Urgent requests
- System updates
- Notification tracking via checkerseen flag

History Tracking

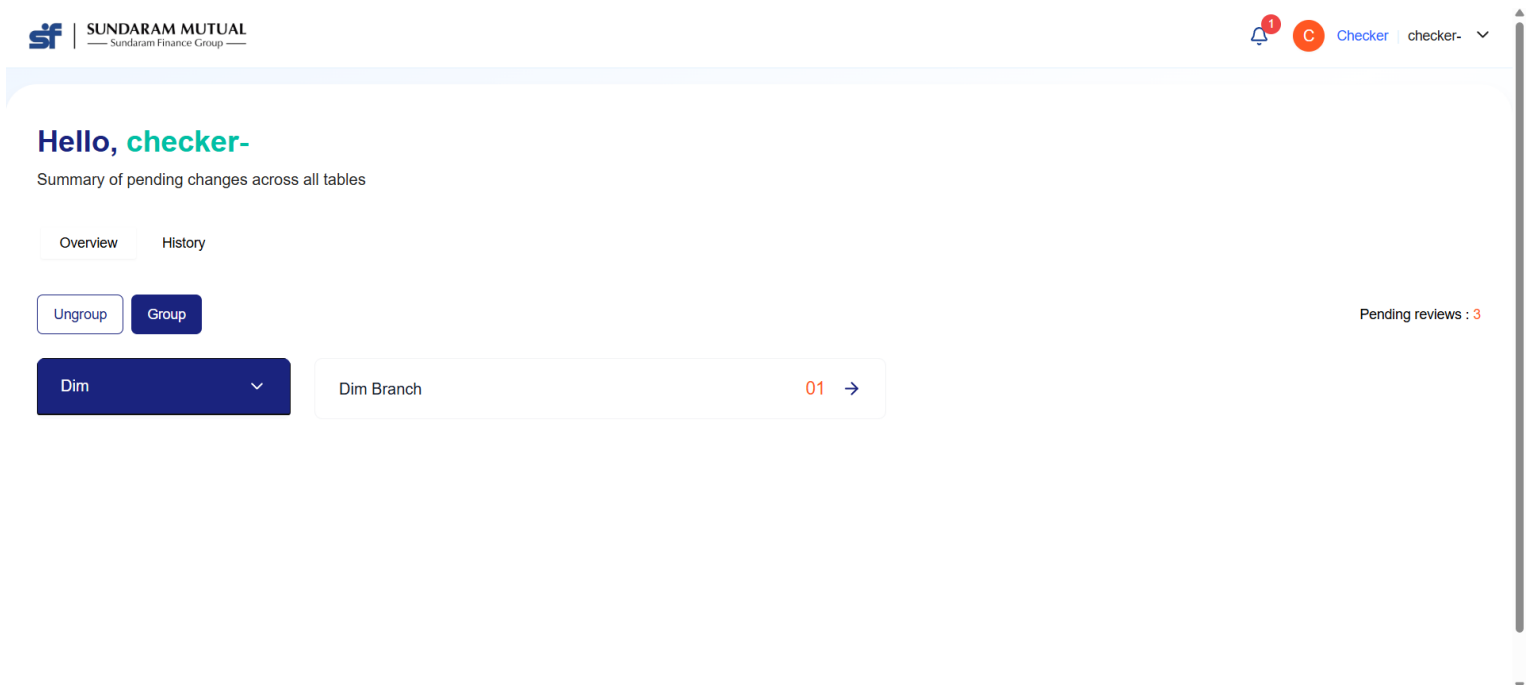
- Complete audit trail of:
- Approved requests
- Rejected requests
- Comments history
- Timestamps
- Action taken



Checker's UI Interface



Checker: Dashboard page (Ungroup table view)



Checker: Dashboard page (Group table view)

SUNDARAM MUTUAL
 — Sundaram Finance Group —

Hello, checker-

Summary of pending changes across all tables


Overview
History

All
Approved
Rejected

Clear Filters
Search



Table Name	Status ↑↓	Maker	Updated
Dim Employee	⊗ Rejected	aryan.agrahari4147@gmail.com	24 Mar 2025 21:27
Dim Employee	⊗ Rejected	avdsheshbhardwaj2004@gmail.com	24 Mar 2025 21:27
Dim Employee	⊗ Rejected	aryan.agrahari4147@gmail.com	11 Mar 2025 16:54
Dim Employee	⊗ Rejected	aryan.agrahari4147@gmail.com	11 Mar 2025 16:54

Checker: Notification Drawer



SUNDARAM MUTUAL

Sundaram Finance Group

Checker

checker-

Hello, checker-

Summary of pending changes across all tables

Overview

History










All

Approved

Rejected

Clear Filters

Search

Table Name	Status 	Maker	Updated	Comments	Actions
Dim Employee	 Rejected	aryan.agrahari4147@gmail.com	24 Mar 2025 21:27	ghff	
Dim Employee	 Rejected	avdsheshbhardwaj2004@gmail.com	24 Mar 2025 21:27	dfxc	
Dim Employee	 Approved	aryan.agrahari4147@gmail.com	11 Mar 2025 18:11	-	
Dim Employee	 Approved	aryan.agrahari4147@gmail.com	11 Mar 2025 17:40	-	

Checker: History page for status of requests

Hello, checker-

Summary of pending changes across all tables

Overview

History

All

Approved





Rejected

Q Search table, maker or comments...

dd-mm-yyyy ▾ to dd-mm-yyyy ▾

Clear Filters

Search

Table Name	Status ↑↓	Maker	Updated	Comments	Actions
Dim Employee	✔ Approved	aryan.agrahari4147@gmail.com	11 Mar 2025 18:11	-	
Dim Employee	✔ Approved	aryan.agrahari4147@gmail.com	11 Mar 2025 17:40	-	
Dim Employee	✔ Approved	aryan.agrahari4147@gmail.com	11 Mar 2025 17:38	-	
Dim Employee	✔ Approved	aryan.agrahari4147@gmail.com	11 Mar 2025 17:37	-	

Checker: Approved request list

Hello, checker-

Summary of pending changes across all tables

Overview

History

All

Approved





Rejected

Q Search table, maker or comments...

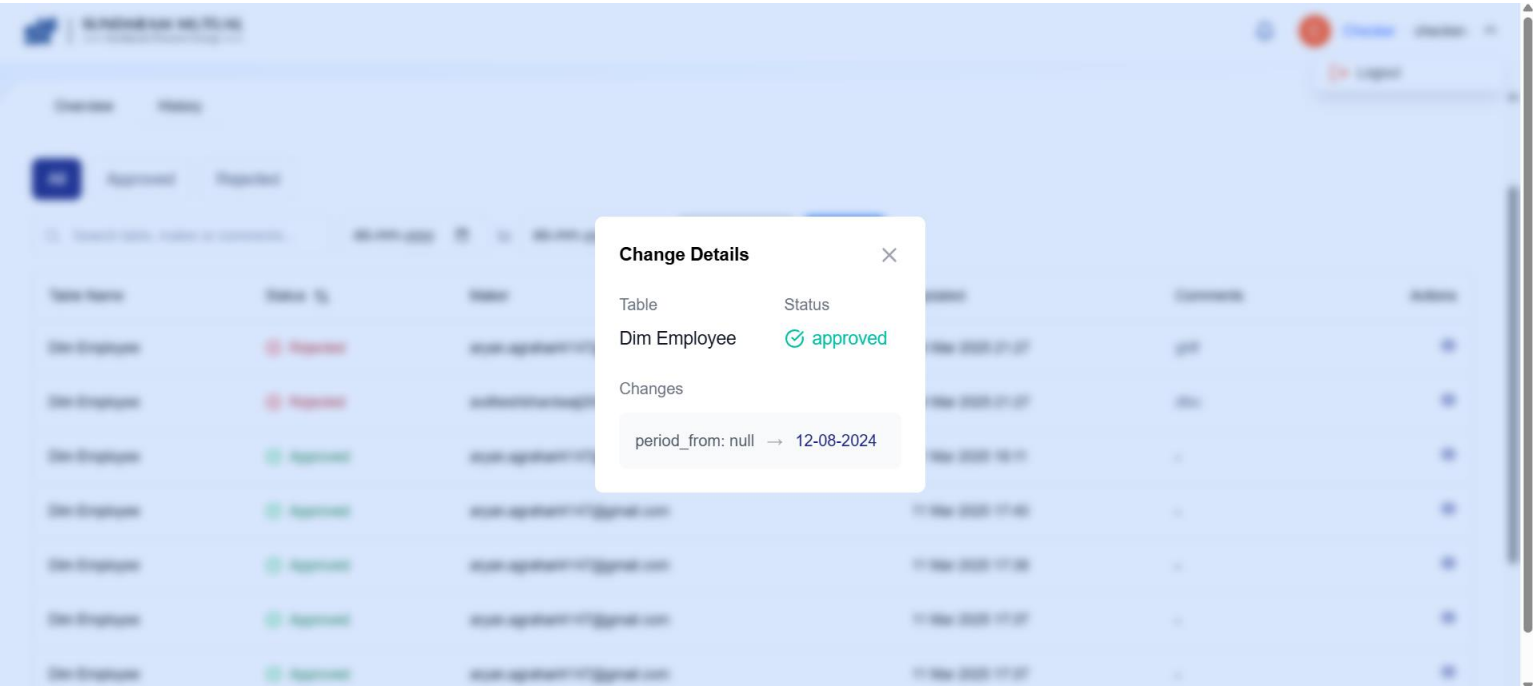
dd-mm-yyyy ▾ to dd-mm-yyyy ▾

Clear Filters

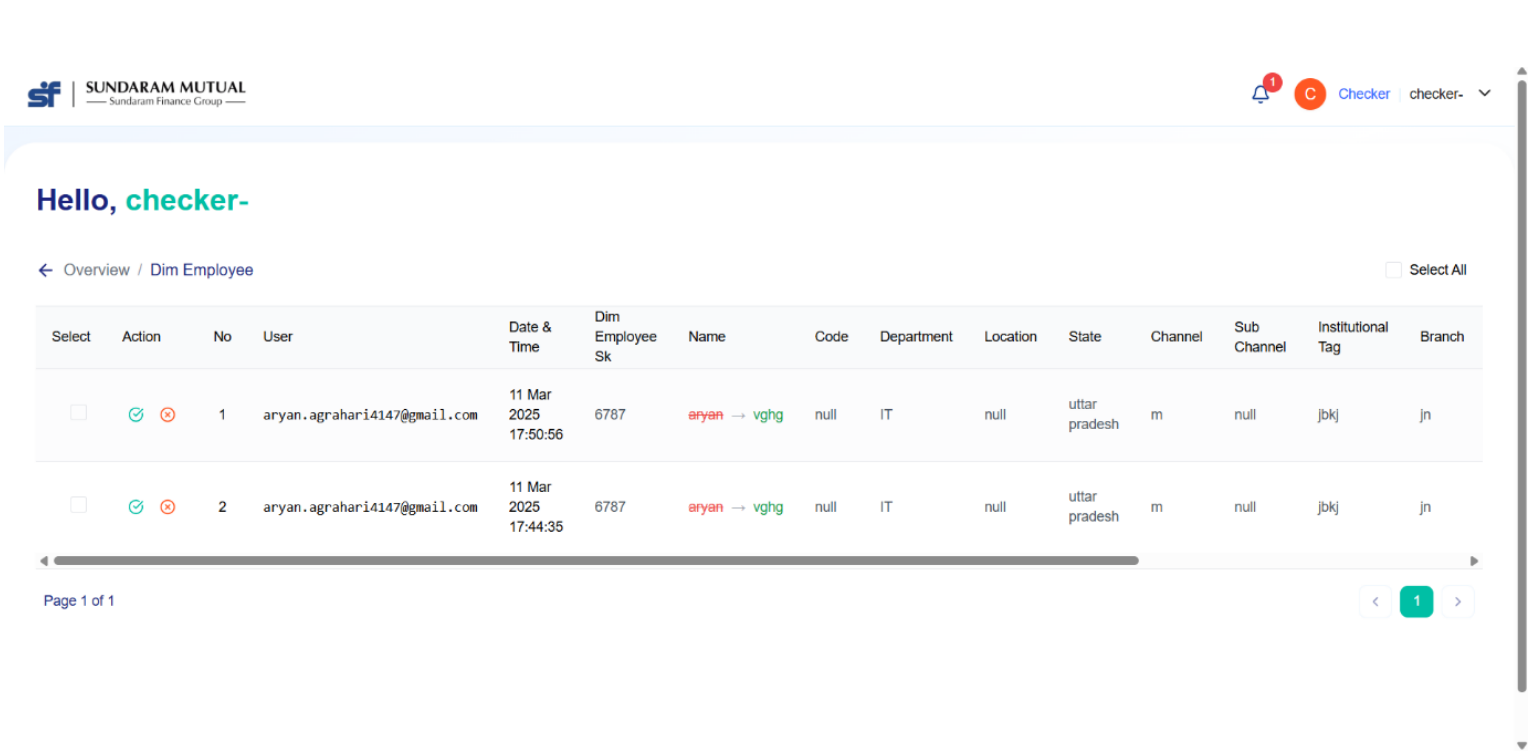
Search

Table Name	Status ↑↓	Maker	Updated	Comments	Actions
Dim Employee	✖ Rejected	aryan.agrahari4147@gmail.com	24 Mar 2025 21:27	ghff	
Dim Employee	✖ Rejected	avdheshbhardwaj2004@gmail.com	24 Mar 2025 21:27	dfxc	
Dim Employee	✖ Rejected	aryan.agrahari4147@gmail.com	11 Mar 2025 16:54	kk	
Dim Employee	✖ Rejected	aryan.agrahari4147@gmail.com	11 Mar 2025 16:54	kk	

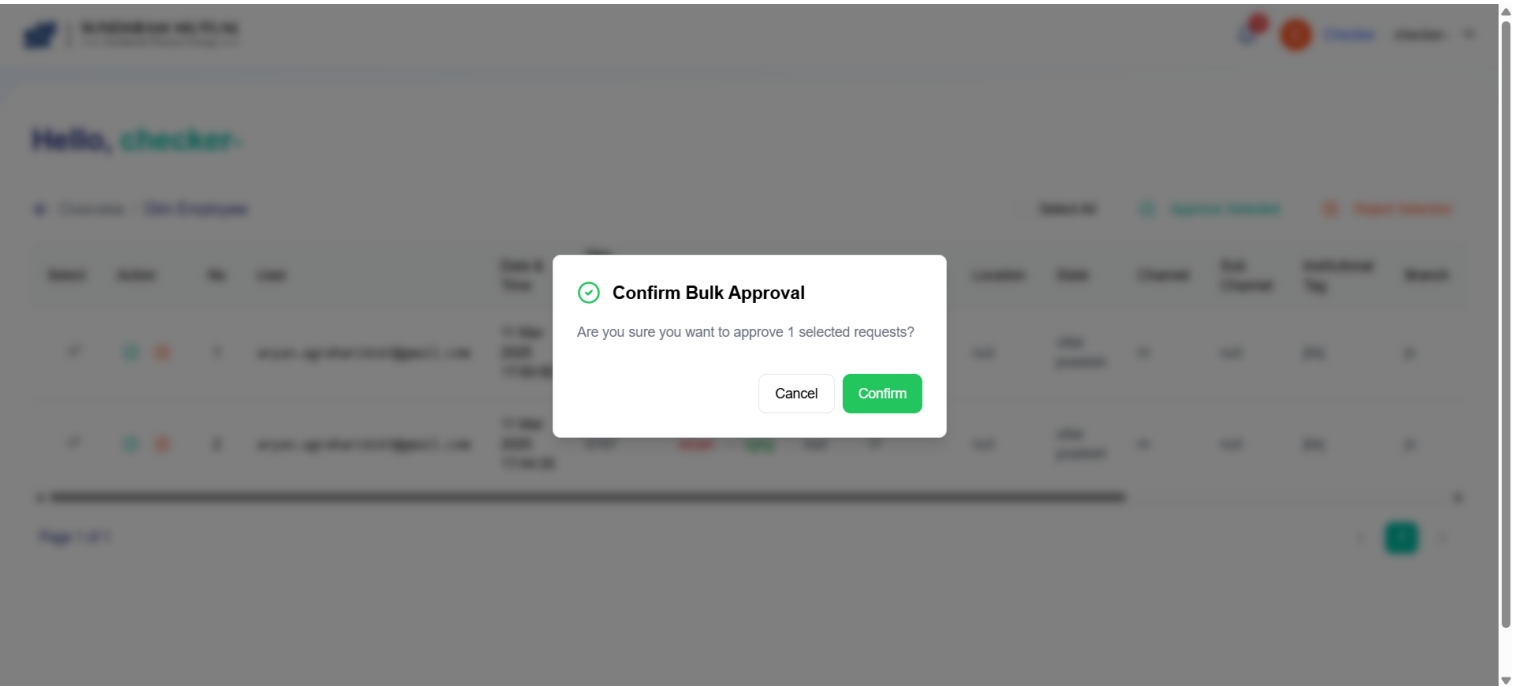
Checker: Rejected Requests list



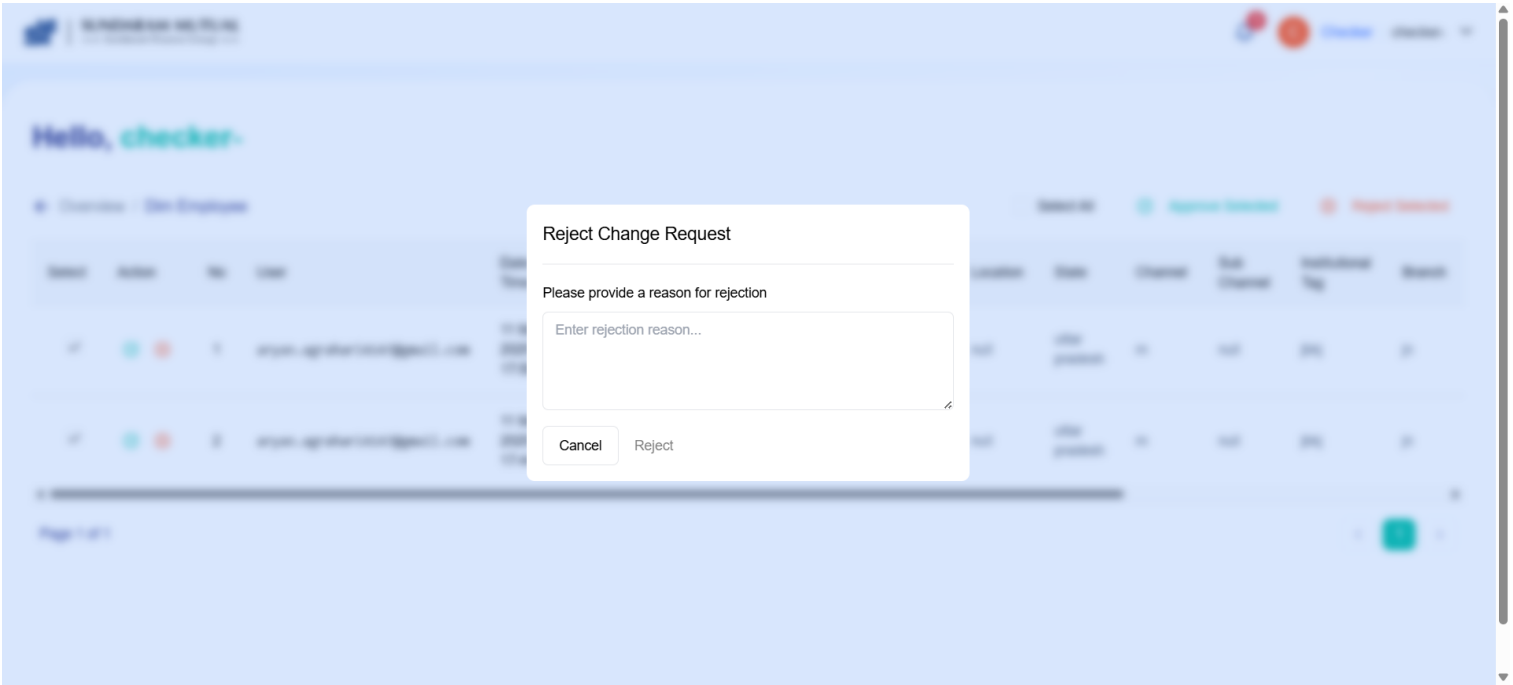
Checker: Eye icon view details of changed data



Checker: Changed row requests



Checker: Approve Dialog for approving requests



Checker: Reject Dialog for rejecting requests

Admin

Core Features

1. User Management

- Create new users (makers, checkers, and admins)
- Update existing user details
- Disable/Enable user accounts
- Cannot delete users (for audit purposes)
- View user activity logs

2. Column Permission Management

- Configure which columns are editable for each table
- Set column-level permissions
- Update existing column permissions
- View current permission configurations

3. Dynamic Dropdown Management

- Configure dropdown options for specific columns
- Add/Update/Remove dropdown values
- Associate dropdowns with specific tables and columns

4. Table Group Management

- Create and manage table groups
- Add/Remove tables from groups
- Configure group-level permissions
- Manage ungrouped tables

5. Table Metadata Management

- Manage display names for tables
- Add/Update table descriptions
- Configure table visibility

Workflow

1. User Creation Process

- Admin creates new user account
- System generates temporary password
- User receives email with credentials
- User must change password on first login

2. Permission Configuration

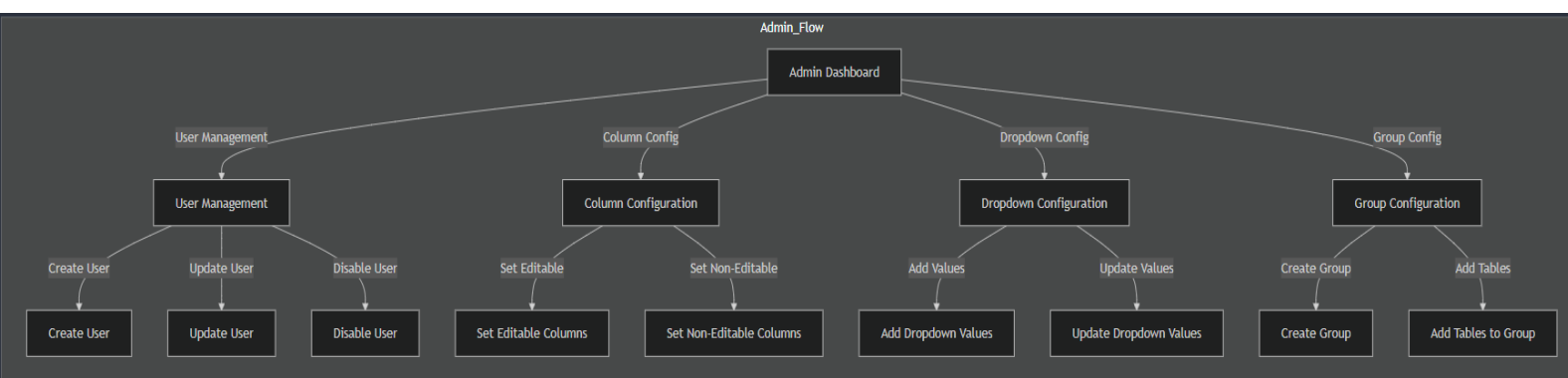
- Admin selects target table
- Views current column permissions
- Updates editable/non-editable status
- System saves and applies changes immediately

3. Dropdown Configuration

- Admin selects table and column
- Configures dropdown values
- Sets validation rules (if any)
- Changes apply to maker's interface

4. Group Management

- Admin creates new group
- Assigns tables to group
- Configures group-level permissions
- Changes reflect in maker/checker interfaces



Admin's UI Interface

SUNDARAM MUTUAL

Sundaram Finance Group

A

Admin | Aryan

Hello, Aryan

Admin dashboard

Row Requests

Configuration

User Management

Dim Branch

Dim Employee

<input type="checkbox"/>	Action	Table	Maker	Status	Created At	View
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 10, 25, 13:17:27	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 08, 25, 18:10:46	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 08, 25, 18:03:03	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 07, 25, 13:44:35	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 06, 25, 21:34:32	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 03, 25, 15:15:53	<div></div>

Admin: Add row requests list

SUNDARAM MUTUAL

Sundaram Finance Group

A

Admin | Aryan

Hello, Aryan

Admin dashboard

Row Requests

Configuration

User Management

Dim Branch

Dim Employee

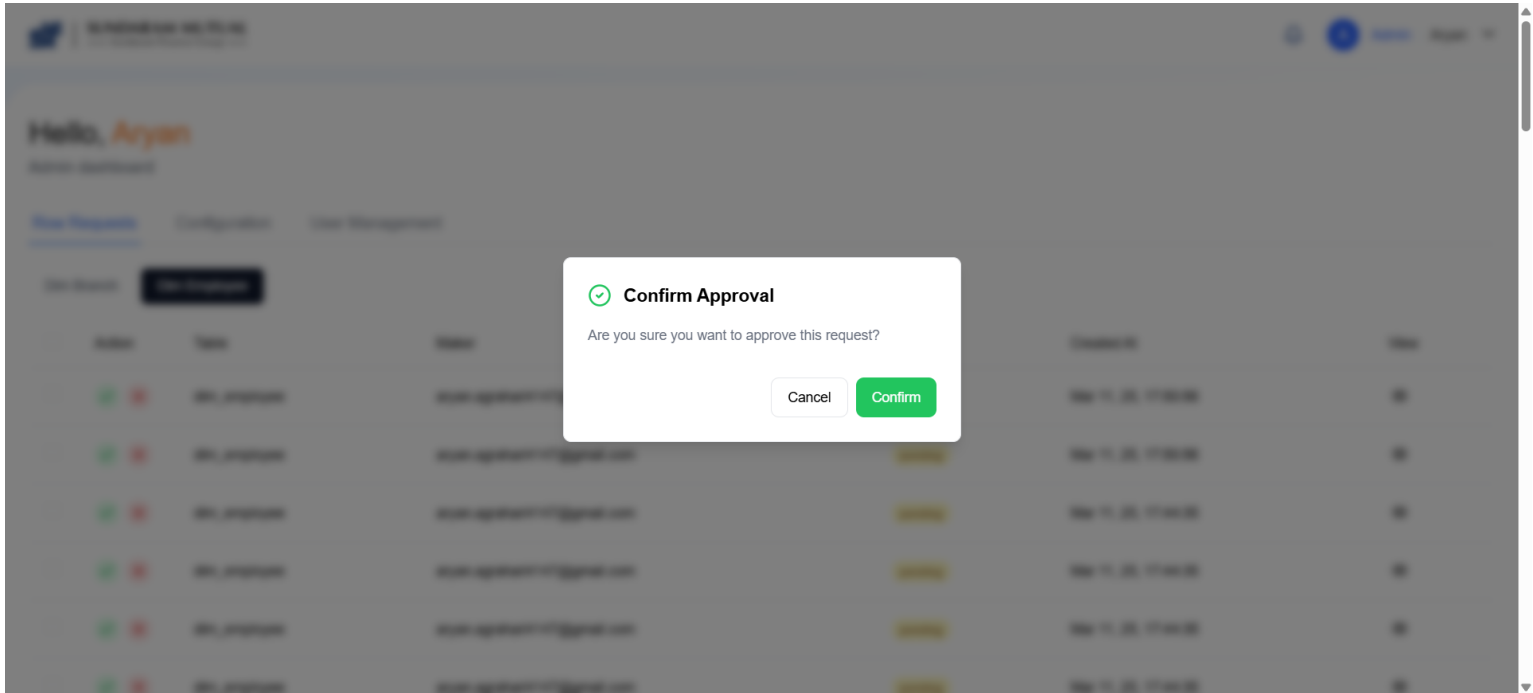
<input type="checkbox"/>	Action	Table	Maker	Status	Created At	View
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 10, 25, 13:17:27	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 08, 25, 18:10:46	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 08, 25, 18:03:03	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 07, 25, 13:44:35	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 06, 25, 21:34:32	<div></div>
<input type="checkbox"/>	<div><div></div><div></div><div></div></div>	dim_branch	aryan.agrahari4147@gmail.com	pending	Mar 03, 25, 15:15:53	<div></div>

Admin: Add row request data

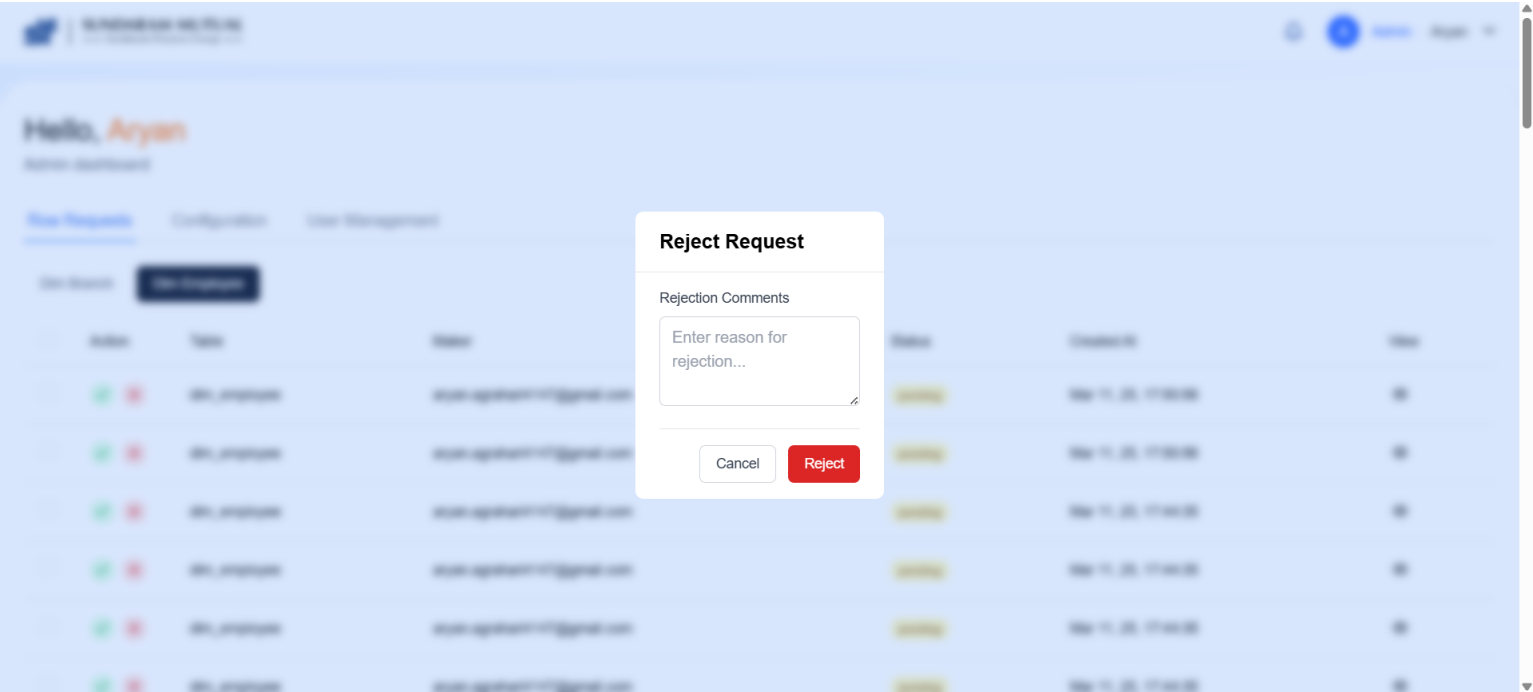
Row Data Details

Field	Value
code	aryan

Close



Admin: Accept Dialog for accepting requests



Admin: Reject Dialog for rejecting requests

Hello, **Aryan**

Admin dashboard

Row Requests [Configuration](#) User Management

Column Configuration

Dropdown Management

Group Configuration

Table Configuration

ria_master ▾

ria_master_sk

Non-editable ☐

riacode

Non-editable ☐

name

Non-editable ☐

arn

Non-editable ☐

Reset

Save

Admin: Column Configuration (editable permission)

Hello, **Aryan**

Admin dashboard

Row Requests [Configuration](#) User Management

Column Configuration

Dropdown Management


Group Configuration

Table Configuration

Dropdown Configuration

 Select Table

ria_master ▾

 Select Column

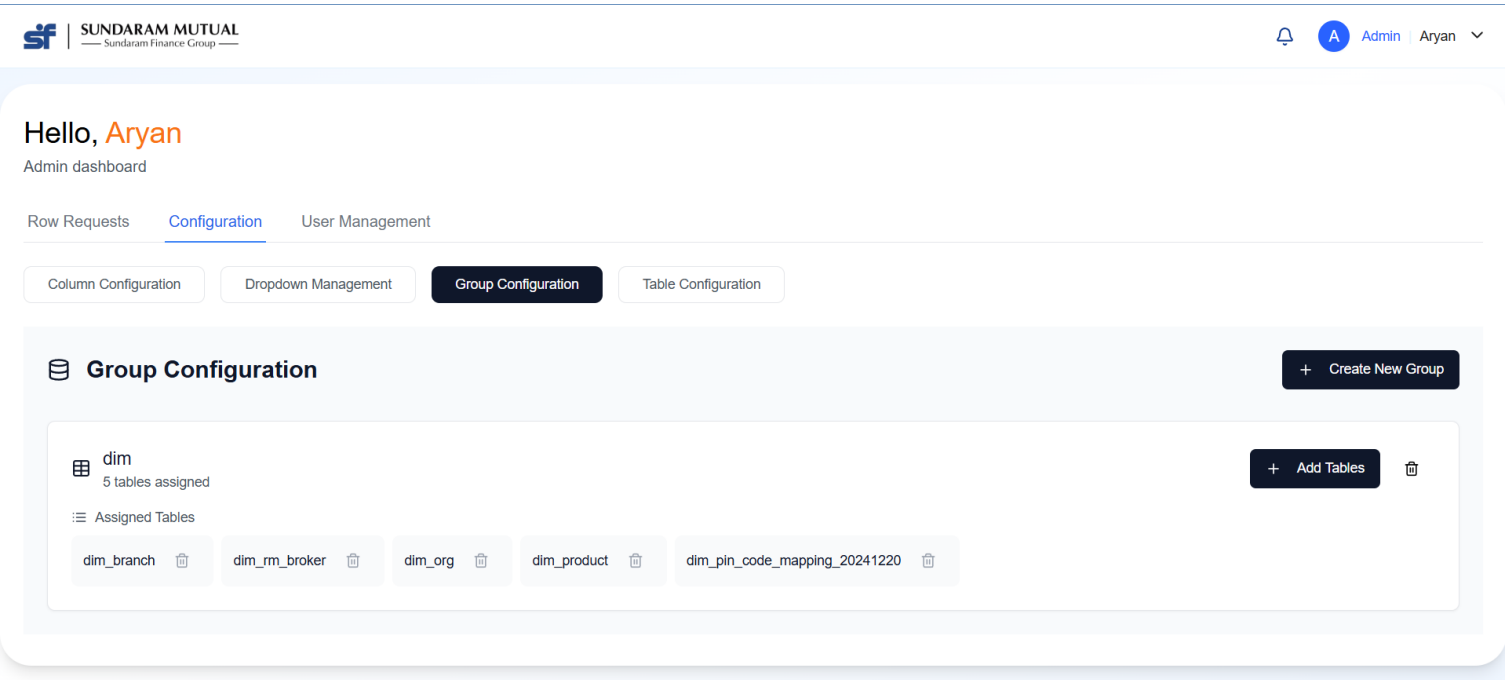
name ▾

Add new option

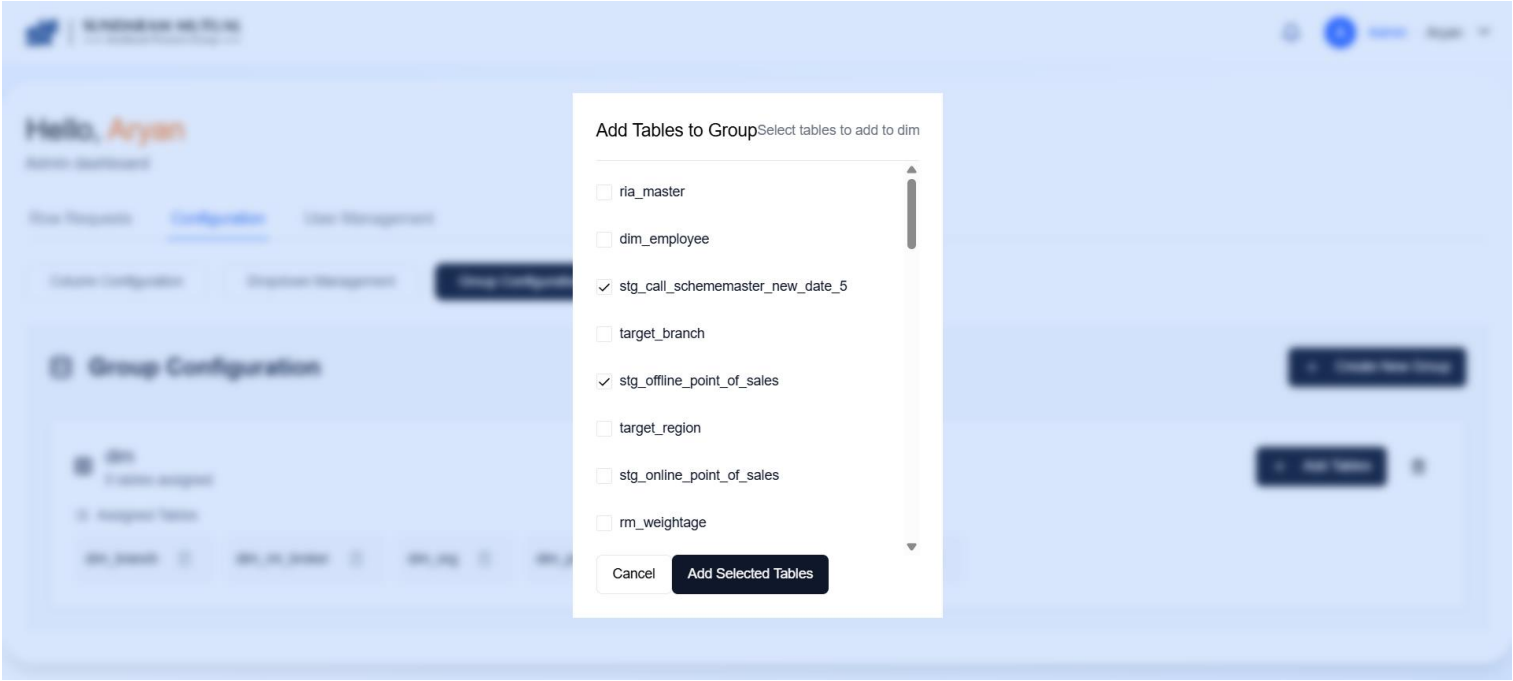
+ Add Option

1 option available

Admin: Dropdown Configuration (for options in dropdown or converting a column into dropdown from input)



Admin: Group Configuration (for grouping tables)



Admin: Group Configuration (Selecting tables for adding in a group)

Hello, **Aryan**

Admin dashboard

Row Requests [Configuration](#) User Management

Column Configuration

Dropdown Management

Group Configuration

Table Configuration

Table Configuration

+ Add Table

dim

Original: dim_employee
employee data



Admin: Table Configuration (adding details or description about the table)

Add New Table

Configure a new table display name and description

Original Table Name

Select a table 

Display Name

Enter display name

Description (Optional)

Enter description

Cancel

Add Table

Admin: Add new table form

Hello, **Aryan**

Admin dashboard

Row Requests Configuration User Management

User Management

Search by name or email...

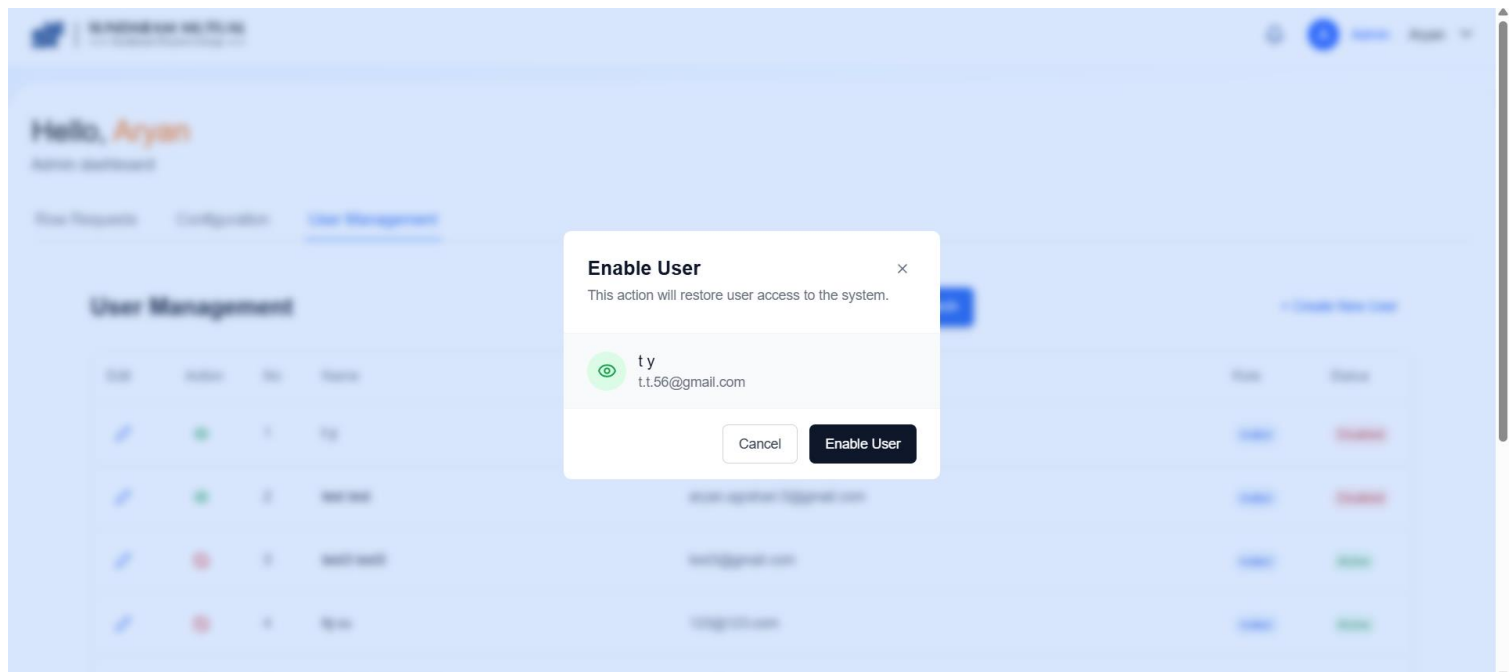
Clear

Search

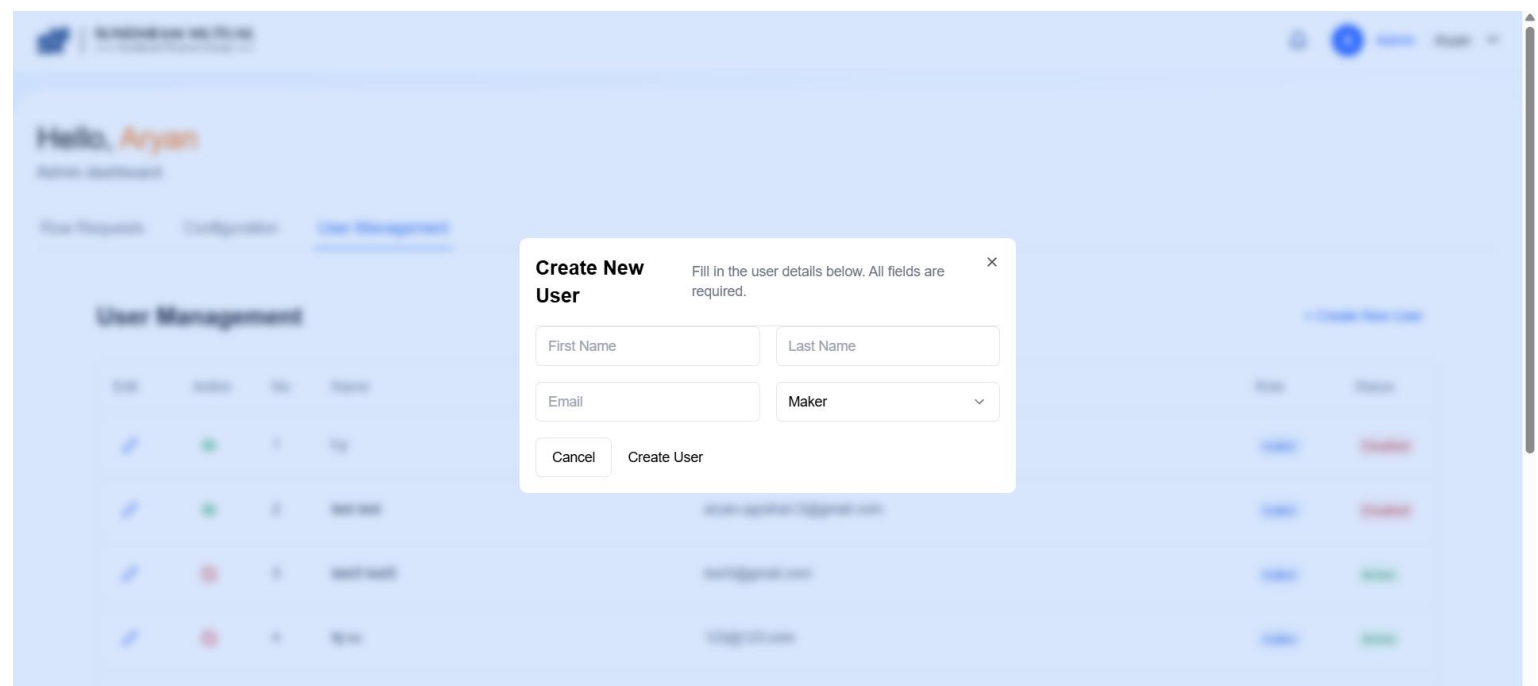
+ Create New User

Edit	Action	No	Name	Email	Role	Status
Edit	Enable	1	t y	t.t.56@gmail.com	maker	Disabled
Edit	Enable	2	test test	aryan.agrahari.5@gmail.com	maker	Disabled
Edit	Disable	3	test3 test3	test3@gmail.com	maker	Active
Edit	Disable	4	hj vu	123@123.com	maker	Active

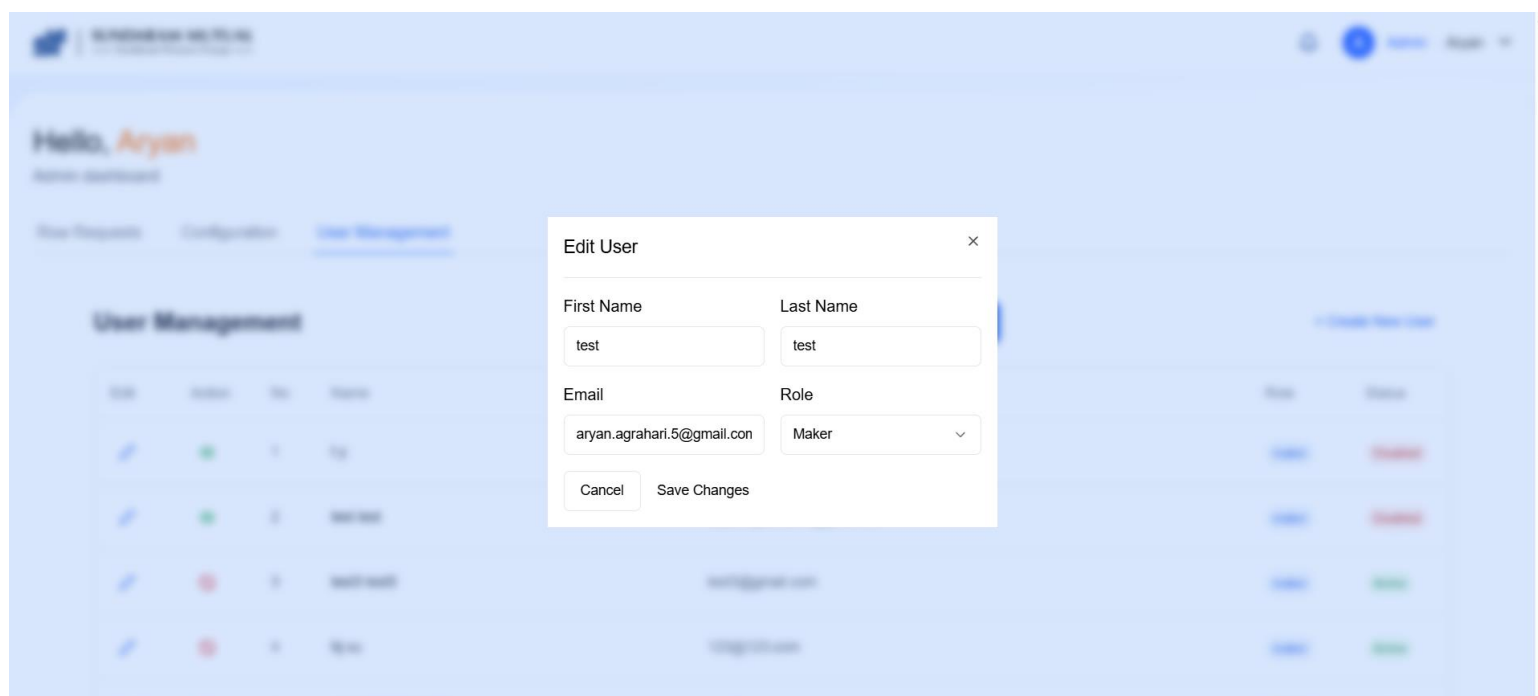
Admin: User Management (list of registered users)



Admin: Enable or disable user



Admin: Create a new user form



Admin: Edit details form of existing user

Sf

SUNDARAM MUTUAL
Sundaram Finance Group

Hello, Aryan

Admin dashboard

Row Requests

Configuration









User Management

User Management

Q Search by name or email...

Clear

Search

Edit	Action	No	Name	Email
		1	t.y	t.t.56@gmail.com
		2	test test	aryan.agrahari.5@gmail.com
		3	test3 test3	test3@gmail.com
		4	hj vu	123@123.com

Admin: Notification Drawer

Notifications

Dim Employee

4 Unread

Maker: aryan.agrahari4147@gmail.com

3/10/2025, 4:36:58 PM

Dim Branch

1 Unread

Maker: aryan.agrahari4147@gmail.com

3/3/2025, 3:15:53 PM