NUTECH Lab Equipment Clearance System (ECMS)

Comprehensive Technical Documentation

Table of Contents

System Overview

The **NUTECH ECMS** is a comprehensive lab equipment management system designed for educational institutions. It handles:

- Equipment inventory tracking across multiple labs
- Multi-stage equipment borrowing requests
- Role-based access control with 7 user levels
- Clearance certificate generation for graduating students

Current Version: V3.0

Technology Stack

Layer	Technology	Purpose
Frontend	Next.js 16.1.1 (App Router)	React-based UI framework
Styling	Tailwind CSS	Utility-first CSS
Database	PostgreSQL (via Supabase)	Relational data storage
Authentication	Supabase Auth	User login/registration
ORM/Client	Supabase JS Client	Database queries
Hosting	Vercel/Self-hosted	Deployment platform

Database Architecture

Database Provider

Supabase PostgreSQL - A cloud-hosted PostgreSQL database with:

- Row Level Security (RLS)
- Real-time subscriptions

- Built-in authentication (`auth.users`)
- Auto-generated REST API

Schema: `public`

```
erDiagram
    auth_users ||--|| profiles : "1:1"
    labs ||--o{ profiles : "assigned_lab_id"
    labs ||--o{ inventory : "lab_id"
    labs ||--o{ borrow_requests : "lab_id"
    profiles ||--o{ borrow_requests : "user_id"
    borrow_requests ||--o{ borrow_request_items : "request_id"
    inventory ||--o{ borrow_request_items : "inventory_id"
    profiles ||--o{ issues : "user_id"
    inventory ||--o{ issues : "inventory_id"
    inventory ||--o{ maintenance_logs : "inventory_id"
    labs ||--o{ procurement_requests : "lab_id"
    profiles ||--o{ notifications : "user_id"
    profiles ||--o{ clearance_requests : "user_id"
```

Database Tables

1. `public.labs` - Laboratory Management

Column	Type	Constraints	Description
`id`	UUID	PK, DEFAULT gen_random_uuid()	Unique lab identifier
`name`	VARCHAR(100)	NOT NULL, UNIQUE	Display name (e.g., "Robotic Lab")
`code`	VARCHAR(20)	NOT NULL, UNIQUE	Short code (e.g., "ROBO", "DLD")
`description`	TEXT		Lab details
`location`	VARCHAR(100)		Room/Block location
`image_url`	TEXT		Lab image URL
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	Creation timestamp

Seed Data:

- ROBO (Robotic Lab)
- DLD (DLD Lab)
- IOT (IOT Lab)
- EMB (Embedded Design Lab)
- CNET (Computer & Network Lab)

Usage: Lab cards in dashboard, inventory filtering, request routing.

---

2. `public.profiles` - User Management

Column	Type	Constraints	Description
`id`	UUID	PK, FK → auth.users(id) ON DELETE CASCADE	Links to Supabase Auth
`email`	VARCHAR(255)	NOT NULL	User email
`full_name`	VARCHAR(255)	NOT NULL	Display name
`role`	user_role (ENUM)	NOT NULL, DEFAULT 'student'	Primary role
`reg_no`	VARCHAR(50)		Student registration number
`department`	VARCHAR(100)		Department affiliation
`contact_no`	VARCHAR(20)		Phone number
`avatar_url`	TEXT		Profile picture URL
`reliability_score`	INT	DEFAULT 100, CHECK 0-100	Borrower reliability metric
`assigned_lab_id`	UUID	FK → labs(id)	For lab staff only
`secondary_role`	user_role		Dual-role support
`secondary_lab_id`	UUID	FK → labs(id)	Secondary lab assignment
`is_active`	BOOLEAN	DEFAULT true	Account status
`notification_preferences`	JSONB	DEFAULT '{"email": true, "in_app": true}'	Notification settings
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	Account creation
`updated_at`	TIMESTAMPTZ	DEFAULT NOW()	Last update

Triggers:

- `on_profile_update` : Auto-updates `updated_at` timestamp
- `on_auth_user_created` : Creates profile when user registers

Usage: Authentication, role-based access, user management, lab assignments.

---

3. `public.inventory` - Equipment Asset Tracking

Column	Type	Constraints	Description
`id`	UUID	PK, DEFAULT gen_random_uuid()	Equipment ID
`lab_id`	UUID	NOT NULL, FK → labs(id) ON DELETE CASCADE	Owning lab

`name`	VARCHAR(255)	NOT NULL	Equipment name
`model`	VARCHAR(255)		Brand/model
`serial_no`	VARCHAR(100)		Serial number
`asset_tag`	VARCHAR(50)	UNIQUE	Internal asset tag
`qr_code`	TEXT	UNIQUE	QR code data
`status`	inventory_status (ENUM)	DEFAULT 'available'	Current status
`condition`	condition_type (ENUM)	DEFAULT 'good'	Physical condition
`quantity`	INT	DEFAULT 1	Quantity available
`purchase_date`	DATE		Acquisition date
`price`	DECIMAL(10,2)		Purchase price
`supplier`	VARCHAR(255)		Vendor name
`maintenance_interval_days`	INT		Maintenance frequency
`last_maintenance_date`	DATE		Last service date
`next_maintenance_date`	DATE		Next due date
`image_url`	TEXT		Equipment photo
`is_active`	BOOLEAN	DEFAULT true	Active status
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	Record creation

ENUMs:

- `inventory_status` : available, borrowed, maintenance, lost, retired
- `condition_type` : excellent, good, fair, poor, damaged

Usage: Equipment inventory pages, request items, lab dashboards.

---

4. `public.borrow_requests` - Equipment Request Pipeline

Column	Type	Constraints	Description
`id`	UUID	PK	Request ID
`user_id`	UUID	NOT NULL, FK → profiles(id)	Requester
`lab_id`	UUID	NOT NULL, FK → labs(id)	Target lab
`request_type`	VARCHAR(20)	CHECK IN ('university', 'home')	Usage location

`purpose`	TEXT	NOT NULL	Reason for borrowing
`start_time`	TIMESTAMPTZ	NOT NULL	Requested start
`end_time`	TIMESTAMPTZ	NOT NULL	Requested end
`return_time`	TIMESTAMPTZ		Actual return time
`is_group_project`	BOOLEAN	DEFAULT false	FYP/Group project flag
`group_members`	JSONB	DEFAULT '[]'	Team member details
`supervisor_name`	VARCHAR(255)		Project supervisor
`status`	request_status (ENUM)	DEFAULT 'submitted'	Pipeline stage
`current_stage`	INT	DEFAULT 1	Current approval stage (1-3)
Stage 1 Tracking			
`stage1_approved_by`	UUID	FK → profiles(id)	Lab Engineer/Assistant
`stage1_approved_at`	TIMESTAMPTZ		Approval timestamp
Stage 2 Tracking			
`stage2_oic_approved_by`	UUID	FK → profiles(id)	OIC approval
`stage2_oic_approved_at`	TIMESTAMPTZ		
`stage2_asst_approved_by`	UUID	FK → profiles(id)	Asst OIC approval
`stage2_asst_approved_at`	TIMESTAMPTZ		
Stage 3 Tracking			
`stage3_approved_by`	UUID	FK → profiles(id)	HOD/Pro-HOD
`stage3_approved_at`	TIMESTAMPTZ		
Handover Tracking			
`handed_over_by`	UUID	FK → profiles(id)	Staff who handed over
`handed_over_at`	TIMESTAMPTZ		
Return Tracking			
`returned_received_by`	UUID	FK → profiles(id)	Staff who received return
`returned_at`	TIMESTAMPTZ		
Rejection Tracking			
`rejected_by`	UUID	FK → profiles(id)	Rejector
`rejected_at`	TIMESTAMPTZ		

`rejection_stage`	INT		Stage at which rejected
`rejection_reason`	TEXT		Reason text
`admin_notes`	TEXT		Staff notes
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	
`updated_at`	TIMESTAMPTZ	DEFAULT NOW()	

Usage: Multi-step request form, approval pipeline, request history.

---

5. `public.borrow_request_items` - Request ↔ Inventory Junction

Column	Type	Constraints	Description
`request_id`	UUID	FK → borrow_requests(id) ON DELETE CASCADE	Parent request
`inventory_id`	UUID	FK → inventory(id)	Equipment item
`quantity_requested`	INT	DEFAULT 1	Quantity needed

Primary Key: (request_id, inventory_id)

Usage: Links multiple equipment items to a single request.

---

6. `public.issues` - Damage & Issue Tracking

Column	Type	Constraints	Description
`id`	UUID	PK	Issue ID
`lab_id`	UUID	NOT NULL, FK → labs(id)	Related lab
`user_id`	UUID	NOT NULL, FK → profiles(id)	Responsible user
`inventory_id`	UUID	FK → inventory(id)	Related equipment
`title`	VARCHAR(255)	NOT NULL	Issue title
`description`	TEXT		Details
`severity`	issue_severity (ENUM)	DEFAULT 'medium'	low/medium/high/critical
`fine_amount`	DECIMAL(10,2)	DEFAULT 0.00	Fine if applicable
`status`	VARCHAR(20)	DEFAULT 'open'	open/resolved
`reported_by`	UUID	FK → profiles(id)	Reporter
`evidence_image_url`	TEXT		Photo evidence
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	

Usage: Damage reporting, fine management, reliability scoring.

---

7. `public.maintenance_logs` - Equipment Servicing

Column	Type	Constraints	Description
`id`	UUID	PK	Log ID
`inventory_id`	UUID	NOT NULL, FK → inventory(id) ON DELETE CASCADE	Equipment
`maintenance_type`	maintenance_type (ENUM)	NOT NULL	preventive/corrective/calibration
`performed_by`	VARCHAR(255)		Technician name
`service_date`	DATE	NOT NULL, DEFAULT CURRENT_DATE	Service date
`cost`	DECIMAL(10,2)	DEFAULT 0.00	Service cost
`technician_notes`	TEXT		Notes
`next_due_date_set`	DATE		Next maintenance
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	

Usage: Maintenance scheduling, service history.

---

8. `public.procurement_requests` - Purchase Requests

Column	Type	Constraints	Description
`id`	UUID	PK	Request ID
`lab_id`	UUID	NOT NULL, FK → labs(id)	Requesting lab
`requester_id`	UUID	NOT NULL, FK → profiles(id)	Staff requester
`item_name`	VARCHAR(255)	NOT NULL	Equipment name
`specification`	TEXT		Technical specs
`quantity`	INT	DEFAULT 1	Quantity needed
`estimated_cost_per_unit`	DECIMAL(10,2)		Unit price
`justification`	TEXT	NOT NULL	Business case
`status`	procurement_status (ENUM)	DEFAULT 'pending_hod'	Approval status
`admin_comments`	TEXT		HOD comments
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	
`updated_at`	TIMESTAMPTZ	DEFAULT NOW()	

ENUM: pending_hod, approved, rejected, ordered, delivered

Usage: New equipment procurement workflow.

---

9. `public.notifications` - User Notifications

Column	Type	Constraints	Description
`id`	UUID	PK	Notification ID
`user_id`	UUID	NOT NULL, FK → profiles(id) ON DELETE CASCADE	Recipient
`title`	VARCHAR(255)	NOT NULL	Notification title
`message`	TEXT	NOT NULL	Body content
`type`	VARCHAR(50)	DEFAULT 'info'	info/warning/success/error
`link`	TEXT		Action URL
`is_read`	BOOLEAN	DEFAULT false	Read status
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	

Usage: In-app notification system.

---

10. `public.clearance_requests` - Graduation Clearance

Column	Type	Constraints	Description
`id`	UUID	PK	Clearance ID
`user_id`	UUID	NOT NULL, FK → profiles(id)	Student
`status`	VARCHAR(20)	DEFAULT 'pending'	Clearance status
`approvals`	JSONB	DEFAULT '{}'	Lab-wise approvals
`certificate_code`	VARCHAR(50)	UNIQUE	Certificate ID
`created_at`	TIMESTAMPTZ	DEFAULT NOW()	

Usage: Final clearance before graduation.

Role Hierarchy & RBAC

7-Level Role System

Level	Role	Label	Capabilities
1	`hod`	Head of Department	Full system access, final approver
2	`pro_hod`	Pro HOD	Can't manage HOD, stage 3 approver
3	`oic_cen_labs`	OIC CEN Labs	Manages all labs, stage 2 approver

4	`asst_oic_cen_labs`	Asst. OIC	Assists OIC, stage 2 approver
5	`lab_engineer`	Lab Engineer	Manages assigned lab, stage 1 approver
6	`lab_assistant`	Lab Assistant	Assists lab engineer, stage 1 approver
7	`student`	Student	Can submit requests only

Role Management Function (Database)

```
CREATE OR REPLACE FUNCTION public.can_manage_role(manager_role user_role, target_role
user_role)
RETURNS BOOLEAN
AS $$
  SELECT CASE
    WHEN manager_role = 'hod' THEN true
    WHEN manager_role = 'pro_hod' AND target_role != 'hod' THEN true
    WHEN manager_role = 'oic_cen_labs' AND get_role_level(target_role) > 2 THEN true
    WHEN manager_role = 'asst_oic_cen_labs' AND get_role_level(target_role) > 3 THEN true
    ELSE false
  END;
$$;
```

TypeScript Implementation

```
// src/types/clearance.ts
export function canManageRole(managerRole: UserRole, targetRole: UserRole): boolean {
  if (managerRole === 'hod') return true;
  if (managerRole === 'pro_hod' && targetRole !== 'hod') return true;
  if (managerRole === 'oic_cen_labs' && ROLE_LEVELS[targetRole] > 2) return true;
  if (managerRole === 'asst_oic_cen_labs' && ROLE_LEVELS[targetRole] > 3) return true;
  return false;
}
```

Equipment Request Pipeline

3-Stage Approval Flow

```
Student Submits → Stage 1 → Stage 2 → Stage 3 → Handover → Return
                  (Lab)     (OIC)     (HOD)
```

Stage	Approvers	Status Codes
1	Lab Engineer OR Lab Assistant	`stage1_pending → stage1_approved`
2	OIC CEN Labs AND Asst. OIC	`stage2_pending → stage2_approved`
3	HOD OR Pro-HOD	`stage3_pending → approved`

Final	Lab Staff (handover/return)	`handed_over` → `returned`

Status Flow

```
stateDiagram-v2
    [*] --> submitted: Student creates
    submitted --> stage1_pending: Auto-advance
    stage1_pending --> stage1_approved: Lab staff approves
    stage1_pending --> rejected: Lab staff rejects
    stage1_approved --> stage2_pending: Auto-advance
    stage2_pending --> stage2_approved: Both OIC approve
    stage2_pending --> rejected: OIC rejects
    stage2_approved --> stage3_pending: Auto-advance
    stage3_pending --> approved: HOD approves
    stage3_pending --> rejected: HOD rejects
    approved --> handed_over: Equipment given to student
    handed_over --> returned: Student returns equipment
    returned --> [*]
    rejected --> [*]
```

Feature-to-Database Mapping

Feature	Tables Used	Key Relationships
Dashboard	profiles, labs, inventory, borrow_requests	user role determines visible data
Equipment Inventory	inventory, labs	`inventory.lab_id` → `labs.id`
Request Form	borrow_requests, borrow_request_items, inventory	request → items → inventory
Request Pipeline	borrow_requests, profiles	approval columns track each approver
User Management	profiles, labs	`profiles.assigned_lab_id` for lab staff
Edit Equipment	inventory	Direct CRUD operations
Lab Pages	labs, inventory	Filtered by `lab_id`

API & Server Actions

Location: `src/app/actions/`

File	Functions	Database Tables
`auth.ts`	`getCurrentUser`, `signOut`	profiles, auth.users

`inventory.ts`	`getInventory`, `getInventoryItem`, `addInventoryItem`, `updateInventoryItem`, `getLabs`	inventory, labs
`equipment-request.ts`	`createEquipmentRequest`, `approveRequest`, `rejectRequest`, `getMyRequests`, `getAllRequests`	borrow_requests, borrow_request_items, profiles
`users.ts`	`getAllUsers`, `createFacultyUser`, `updateUserRole`, `deleteUser`	profiles, auth.users

Security Implementation

Row Level Security (RLS)

```
-- Labs are publicly readable
CREATE POLICY "Public Read Labs" ON public.labs FOR SELECT USING (true);

-- Users can only view their own profile
CREATE POLICY "Users view own profile" ON public.profiles
  FOR SELECT USING (id = auth.uid());

-- Admins have full access
CREATE POLICY "Admins full access" ON public.profiles
  FOR ALL USING (
    EXISTS (SELECT 1 FROM public.profiles WHERE id = auth.uid() AND role IN ('hod',
'pro_hod'))
  );
```

Application-Level Security

1. **Server Actions**: All data mutations go through server actions with authentication checks
2. **Role Checks**: `canManageRole()` function validates permissions
3. **Lab Assignment**: Lab staff can only access their assigned lab's data

---

Database Connection

Environment Variables

```
NEXT_PUBLIC_SUPABASE_URL=https://your-project.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-anon-key
SUPABASE_SERVICE_ROLE_KEY=your-service-role-key
```

Client Initialization

```
// src/lib/supabase/server.ts
import { createServerClient } from '@supabase/ssr';
import { cookies } from 'next/headers';

export async function createClient() {
  const cookieStore = await cookies();
  return createServerClient(
    process.env.NEXT_PUBLIC_SUPABASE_URL!,
    process.env.NEXT_PUBLIC_SUPABASE_ANON_KEY!,
    { cookies: { getAll: () => cookieStore.getAll(), setAll: (c) => c.forEach(...) } }
  );
}
```

Database Migrations

File	Description
`00_rebuild_database.sql`	Initial schema (V2.1)
`12_fix_rls_recursion.sql`	RLS policy fixes
`13_seed_inventory.sql`	Sample equipment data
`14_v3_migration.sql`	V3 role hierarchy + pipeline
`15_orphan_cleanup_rpc.sql`	User cleanup function
`21_add_quantity_support.sql`	Quantity column for items
`99_MASTER_RESET.sql`	Complete database reset

Document Version

- **Version:** 3.0
- **Last Updated:** January 2026
- **Author:** NUTECH ECMS Development Team