Design document: FoodBridge

By: Simeon Markov

Institution Name: Fontys UAS

Course/Class: ICT/EN08

Date: 2025-12-11

Introduction

The purpose of this document is to set a clear design plan, illustrating incorporated layout principles. It includes the wireframing process, explaining each wireframe and the user flow as well as the design of the database (shown in EER diagram) explaining fields and relations between tables.

Design Guide (UI/UX Principles)

This section presents the initial design plan (incorporating layout principles, accessibility, etc.)

User-Centric: Interfaces should align with the organizational needs and expectations (key features).

Visual Hierarchy: Having a strategic application of layout, color, typography, and spacing to emphasize high-priority data while minimizing cognitive load.

Simplicity and clarity: Presenting complex information in digestible formats without overwhelming users with excessive visual elements (for example adhering to the 5-second rule).

Consistency: Consistency across typography, color schemes, button styles, and interactive behaviors.

Appropriate Visualization Selection: Matching chart types to data characteristics and analytical objectives. Line charts excel at temporal trend representation, bar charts facilitate categorical comparisons.

Real-Time Data Presentation and Contextual Feedback: Dashboards should provide real-time data visualization in formats that remain clear yet unobtrusive. Users require constant awareness of system status through loading indicators, success and error messages, and visual cues guiding interactions.

Big As Numbers (BANs): Providing context through comparisons, trend lines, or sparklines rather than isolated numerical values.

Wireframing

The wireframing was done on Figma: Popular software for designing and prototyping.

Login system

The login system is the entry-point of the foodbank app. The user is navigated to the login/registration page for authentication and afterwards he/she could access the dashboard.

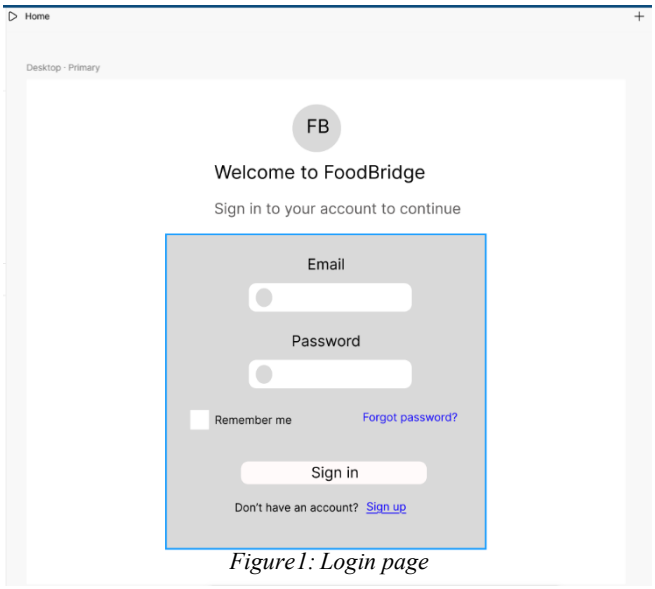

Figure1: Login page

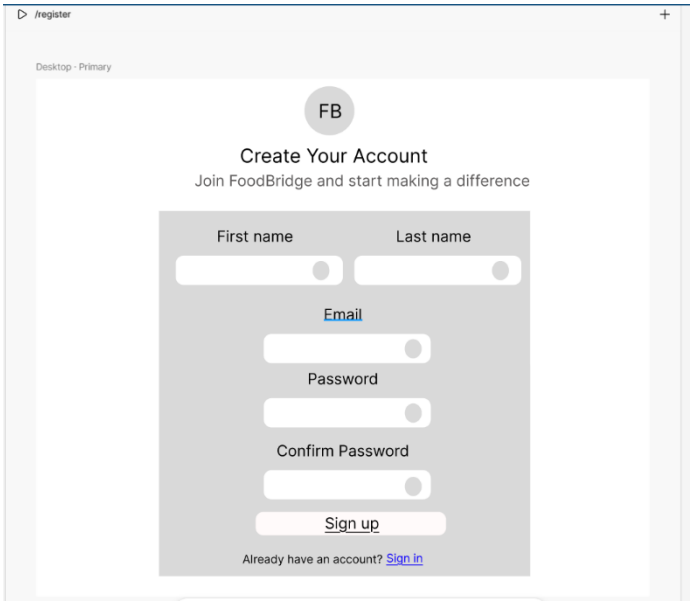

Figure 2: Register page

Index page

This page contains dashboard content (managing donors, food supplies, etc.). It is the first page the authenticated user is going to see when logged in.

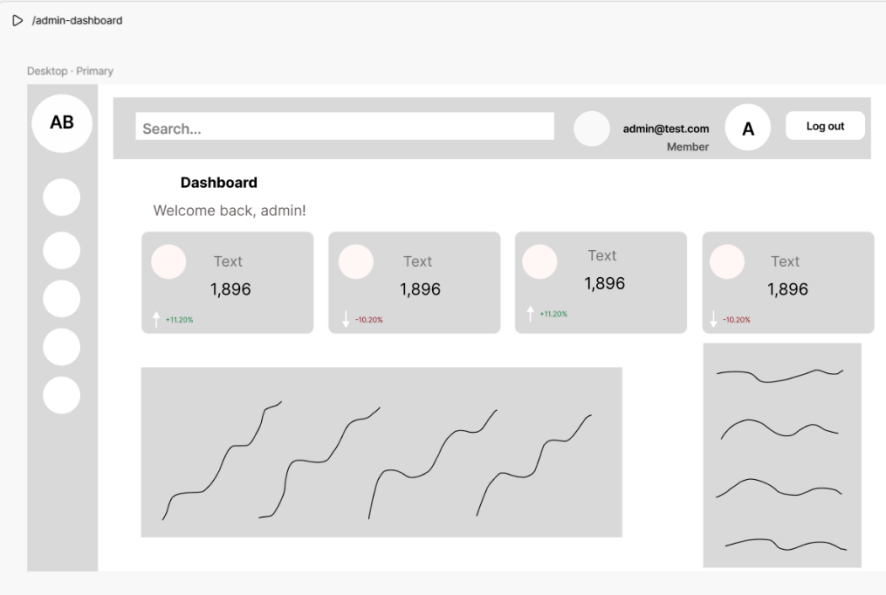

Figure 3: Dashboard page

<u>*Breakdown:*</u> The page consists of several containers: *nav* (containing all the links to the different internal pages within the dashboard), *search* (global search) and *main* container, which *contains data analytics (e.g. charts)*.

Manage Resource page

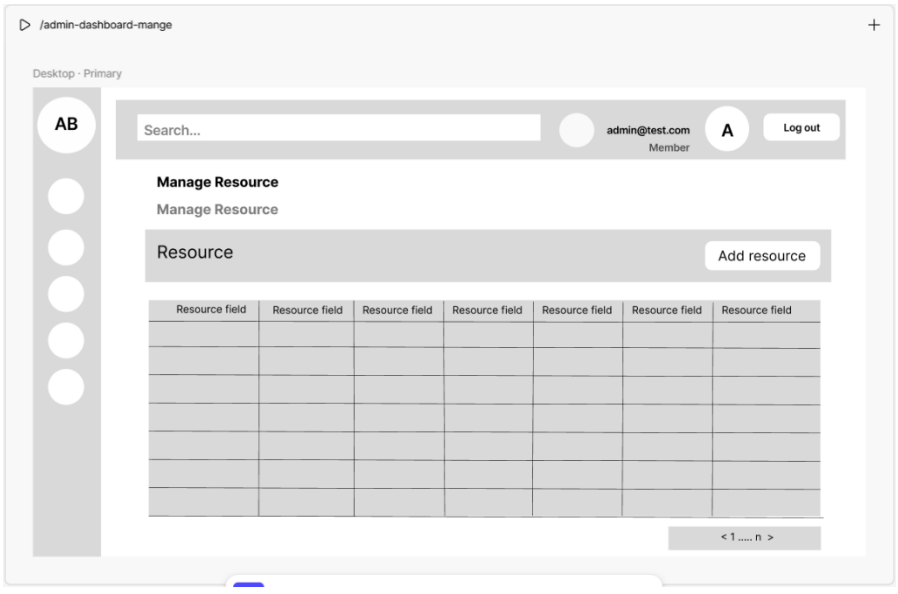

Figure 4: Manage resource page

<u>*Breakdown:*</u> The page consists of dynamic components for the different sections of the management resources page. It displays the resource that is to be manipulated (create, update, delete) and a table showing the records for each resource fetched from the database. On the bottom right is positioned the pagination.

Profile component

Contains personal information.

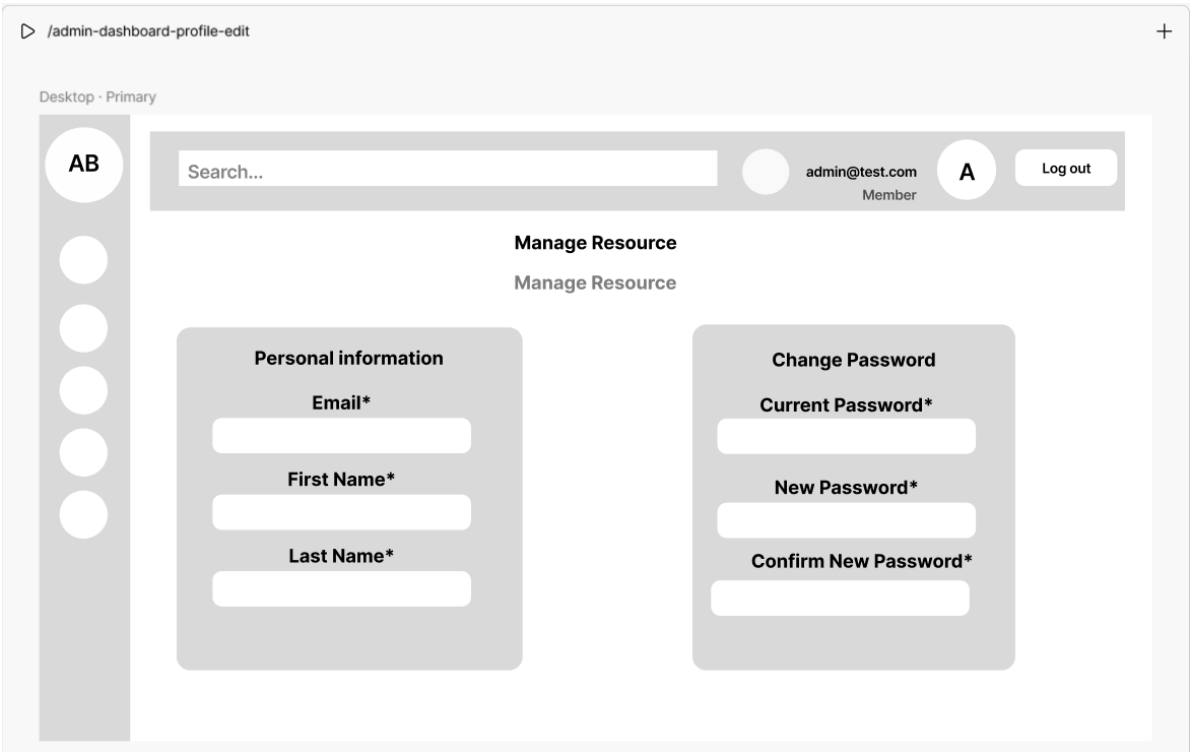

Figure 5:Profile page

Breakdown: Users can change their personal information by updating the related fields.

Techstack decision

Frontend Architecture

- **React 18** delivers substantial performance improvements through concurrent rendering, which enables the framework to pause, resume, and prioritize rendering work (UI remains responsive even during complex operations).
- **JavaScript:** The standard language for web development, ensuring broad compatibility and a vast ecosystem of libraries and tools.
- **Tailwind CSS:** Its utility-first approach eliminates the bloat of unused styles. Tailwind's JIT (Just-in-Time) mode generates styles on-demand during development while ensuring production builds contain only the classes used in the codebase.
- **Recharts:** integrates seamlessly with React's component model, following declarative patterns familiar to React developers. Built on D3.js submodules, it provides a composable, lightweight charting solution with native SVG support.

Backend Architecture

- **C#**: offers type safety, mature tooling, and exceptional performance for backend operations.
- **ASP.NET Core 8:** Delivers exceptional performance. Enhanced features like improved form binding in Minimal APIs, better Blazor support, and dynamic Profile-Guided Optimization (PGO) enabled by default deliver approximately 15% average performance improvements.
- **SQL Server** provides enterprise-grade reliability with features specifically designed for complex workloads. In-Memory OLTP and Columnstore Indexes enhance transaction processing and query

performance, enabling faster data access and real-time analytics. The database integrates seamlessly with the Microsoft ecosystem—including SSMS, Azure, Power BI.

Database design

(SSMS) SQL Server Management Studio (RDBMS) was used for generating the EER diagram. The diagram was generated based on the migrations (code-first approach).

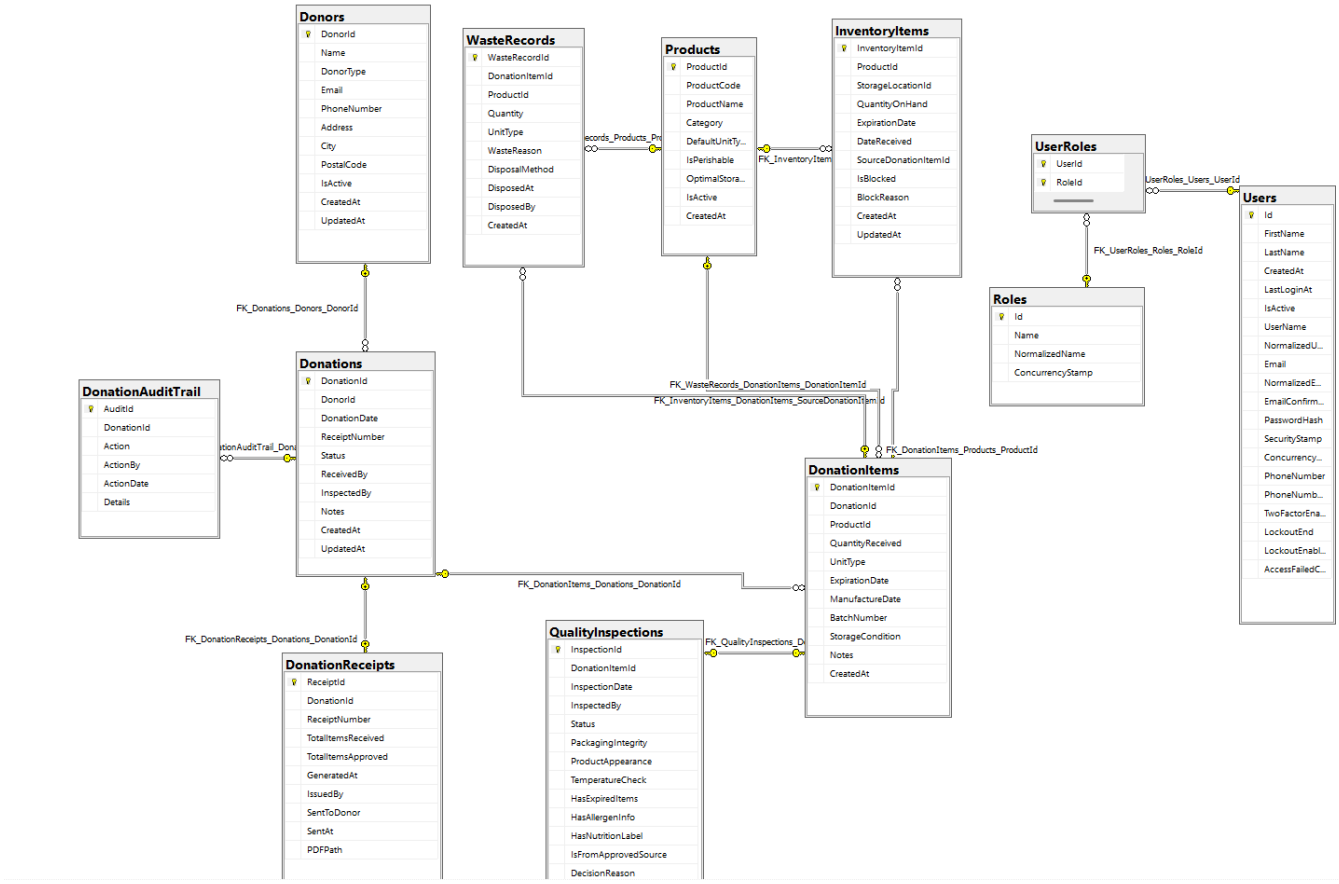

Figure 9: Database diagram

Overview

List of all entities:

- Donation
- DonationAuditTrail
- DonationItem
- DonationReceipt
- Donor
- InventoryItem
- Products
- QualityInspection
- WasteRecord

Core Entities:

- ***Users:*** *This table stores user information, including their name, email, password.*
- ***Donors:*** *This table the credentials of the donors like name, email, etc.*
- ***Donations:*** *This table defines the donation's status, donated date, inspected by, etc. It establishes a many-to-one relationship with the* ***Donors*** *table* (one donor could make multiple donations).
- ***Products:*** *This table stores product's data like name, category, quantity, etc.*
- ***InventoryItems:*** *This is the table for keeping track of storing the products (storage location, which is foreign key, expiration date). The established relationship between table* ***Products*** *is many-to-one (many products could belong to one storage).*
- *Key Relationships:*
- *One donor could make multiple donations (one-to-many).*
- *Products could have one storage location (many-to-one).*
- *A product of donation could pass several inspections (many-to-one).*

Tables description

'DonationAuditTrail:

Column	***Type***	***Constraint***	***Description***
AuditId	*INT*	*PRIMARY KEY, AUTO_INCREMENT*	*Unique record id*
DonationId	*INT*	*NOT NULL, FOREIGN KEY*	*Links to Donations table*
Action	*NVARCHAR(100)*	*NOT NULL*	*Type of action (edit, approve, etc.)*
ActionBy	*INT*	*NOT NULL*	*User who performed the action*
ActionDate	*DATETIME*	*NOT NULL*	*When the action occurred*
Details	*LONGTEXT*	*NOT NULL*	*Details of the activity*

Table 1: DonationAuditTrail

'DonationItems:

Column	**Type**	**Constraint**	**Description**
DonationItemId	INT	PRIMARY KEY, AUTO_INCREMENT	Unique item id
DonationId	INT	NOT NULL, FOREIGN KEY	Links to Donations table

Column	Type	Constraint	Description
ProductId	INT	NOT NULL, FOREIGN KEY	Links to Products catalog
QuantityReceived	INT	NOT NULL	Received quantity
UnitType	NVARCHAR(50)	NOT NULL	Unit of measurement
ExpirationDate	DATETIME	NULL	Expiry date if applicable
ManufactureDate	DATETIME	NULL	Date made
BatchNumber	NVARCHAR(100)	NOT NULL	Batch identification number
StorageCondition	NVARCHAR(100)	NOT NULL	Required storage conditions
Notes	LONGTEXT	NULL	Additional remarks
CreatedAt	DATETIME	NOT NULL	Record creation time

Table 2: DonationItems

‘DonationReceipts:

Column	Type	Constraint	Description
ReceiptId	INT	PRIMARY KEY, AUTO_INCREMENT	Unique receipt id
DonationId	INT	NOT NULL, FOREIGN KEY	Links to Donations table
ReceiptNumber	NVARCHAR(50)	NOT NULL, UNIQUE	Unique receipt code
TotalItemsReceived	INT	NOT NULL	Number of items received
TotalItemsApproved	INT	NOT NULL	Number of items approved
GeneratedAt	DATETIME	NOT NULL	Date receipt was generated
IssuedBy	INT	NOT NULL	User who created the receipt
SentToDonor	BOOLEAN	NOT NULL, DEFAULT FALSE	Whether sent to donor

Column	Type	Constraint	Description
SentAt	DATETIME	NULL	When sent to donor
PDFPath	NVARCHAR(500)	NOT NULL	File path for receipt PDF

Table 3: DonationReceipts.

'Donations':

Column	Type	Constraint	Description
DonationId	INT	PRIMARY KEY, AUTO_INCREMENT	Unique donation id
DonorId	INT	NOT NULL, FOREIGN KEY	Links to Donors table
DonationDate	DATETIME	NOT NULL	Date received
ReceiptNumber	NVARCHAR(50)	NOT NULL, UNIQUE	Receipt code
Status	NVARCHAR(50)	NOT NULL	Current status of donation
ReceivedBy	INT	NOT NULL	User who accepted donation
InspectedBy	INT	NULL	User who inspected (if any)
Notes	LONGTEXT	NOT NULL	Remarks
CreatedAt	DATETIME	NOT NULL	Record created
UpdatedAt	DATETIME	NOT NULL	Last modification

Table 4: Donations

'Donors':

Column	Type	Constraint	Description
DonorId	INT	PRIMARY KEY, AUTO_INCREMENT	Unique donor id
Name	NVARCHAR(255)	NOT NULL	Donor organization/person name
DonorType	NVARCHAR(50)	NOT NULL	Type (individual, company, etc.)

Column	Type	Constraint	Description
Email	NVARCHAR(255)	NOT NULL, UNIQUE	Email contact
PhoneNumber	NVARCHAR(20)	NOT NULL	Phone
Address	NVARCHAR(500)	NOT NULL	Full address
City	NVARCHAR(100)	NOT NULL	City
PostalCode	NVARCHAR(10)	NOT NULL	Postal/Zip code
IsActive	BOOLEAN	NOT NULL, DEFAULT TRUE	Donor account enabled or not
CreatedAt	DATETIME	NOT NULL	When created
UpdatedAt	DATETIME	NOT NULL	Last update

Table 5: Donors'

'InventoryItems':

Column	Type	Constraint	Description
InventoryItemId	INT	PRIMARY KEY, AUTO_INCREMENT	Inventory item id
ProductId	INT	NOT NULL, FOREIGN KEY	Product catalog reference
StorageLocationId	INT	NULL	Storage location (if tracked)
QuantityOnHand	INT	NOT NULL	Stock quantity available
ExpirationDate	DATETIME	NULL	Best before date
DateReceived	DATETIME	NOT NULL	Date received
SourceDonationItemId	INT	NULL, FOREIGN KEY	Links to original donation item
IsBlocked	BOOLEAN	NOT NULL, DEFAULT FALSE	Flag if inventory blocked
BlockReason	LONGTEXT	NOT NULL	Reason for blocking (required)
CreatedAt	DATETIME	NOT NULL	Record creation time
UpdatedAt	DATETIME	NOT NULL	Last update

Table 6: InventoryItems'

'Products':

Column	Type	Constraint	Description
ProductId	INT	PRIMARY KEY, AUTO_INCREMENT	Unique catalog id
ProductCode	NVARCHAR(50)	NOT NULL, UNIQUE	SKU/product code
ProductName	NVARCHAR(255)	NOT NULL	Product name
Category	NVARCHAR(100)	NOT NULL	Product category
DefaultUnitType	NVARCHAR(50)	NOT NULL	Default unit type
IsPerishable	BOOLEAN	NOT NULL	If product can expire
OptimalStorageCondition	NVARCHAR(100)	NOT NULL	Best storage condition
IsActive	BOOLEAN	NOT NULL, DEFAULT TRUE	Product enabled for transactions
CreatedAt	DATETIME	NOT NULL	When product was added

Table 7: Products

'QualityInspections':

Column	Type	Constraint	Description
InspectionId	INT	PRIMARY KEY, AUTO_INCREMENT	Quality inspection record id
DonationItemId	INT	NOT NULL, FOREIGN KEY	Links to DonationItems
InspectionDate	DATETIME	NOT NULL	Date of inspection
InspectedBy	INT	NOT NULL	User who inspected
Status	NVARCHAR(50)	NOT NULL	Status (passed, failed, etc.)
PackagingIntegrity	NVARCHAR(50)	NOT NULL	Packaging quality

Column	Type	Constraint	Description
ProductAppearance	NVARCHAR(50)	NOT NULL	Appearance assessment
TemperatureCheck	DECIMAL(18,2)	NULL	Temperature (if tracked)
HasExpiredItems	BOOLEAN	NOT NULL	Expiry indicator
HasAllergenInfo	BOOLEAN	NOT NULL	Allergen info check
HasNutritionLabel	BOOLEAN	NOT NULL	Nutrition label present
IsFromApprovedSource	BOOLEAN	NOT NULL	Source validity check
DecisionReason	LONGTEXT	NULL	Inspector's rationale
CreatedAt	DATETIME	NOT NULL	Creation time

Table 8: QualityInspections

'Roles':

Column	Type	Constraint	Description
Id	NVARCHAR(450)	PRIMARY KEY	Role id
Name	NVARCHAR(256)	NULL	Role name (friendly)
NormalizedName	NVARCHAR(256)	NULL	Normalized name (standard comparison)
ConcurrencyStamp	LONGTEXT	NULL	Stamp for concurrency check

Table 9: Roles

'UserRoles':

Column	Type	Description	Constraints
id	VARCHAR(255)	Primary key, unique identifier for the job batch.	Primary Key
name	VARCHAR(255)	A descriptive name for the job batch.	Not Null
total_jobs	INT	The total number of jobs in the batch.	Not Null
pending_jobs	INT	The number of jobs in the batch that are still pending.	Not Null
failed_jobs	INT	The number of jobs in the batch that have failed.	Not Null
failed_job_ids	LONGTEXT	A serialized list of IDs of the failed jobs within the batch.	Not Null

options	MEDIUMTEXT	Configuration options for the job batch.	Nullable
cancelled_at	INT	Unix timestamp indicating when the job batch was cancelled.	Nullable
created_at	INT	Unix timestamp indicating when the job batch was created.	Not Null
finished_at	INT	Unix timestamp indicating when the job batch was completed.	Nullable

Table 10: UserRoles (pivot table)

'Users':

Column	Type	Constraint	Description
Id	NVARCHAR(450)	PRIMARY KEY	User id
FirstName	LONGTEXT	NULL	First name
LastName	LONGTEXT	NULL	Last name
CreatedAt	DATETIME	NOT NULL	Creation date
LastLoginAt	DATETIME	NULL	Last login
IsActive	BOOLEAN	NOT NULL, DEFAULT TRUE	User enabled/disabled
UserName	NVARCHAR(256)	NULL	Username
NormalizedUserName	NVARCHAR(256)	NULL	Normalized name for login
Email	NVARCHAR(256)	NULL	Email address
NormalizedEmail	NVARCHAR(256)	NULL	Normalized email for search/login
EmailConfirmed	BOOLEAN	NOT NULL	Email verification status
PasswordHash	LONGTEXT	NULL	Encrypted password
SecurityStamp	LONGTEXT	NULL	Security reset stamp
ConcurrencyStamp	LONGTEXT	NULL	Stamp for concurrency checks
PhoneNumber	LONGTEXT	NULL	User phone number
PhoneNumberConfirmed	BOOLEAN	NOT NULL	Phone number verification status
TwoFactorEnabled	BOOLEAN	NOT NULL	Whether 2FA is active
LockoutEnd	DATETIME	NULL	Account lockout end date

Column	Type	Constraint	Description
LockoutEnabled	BOOLEAN	NOT NULL	Whether account can be locked
AccessFailedCount	INT	NOT NULL	Failed login attempts

Table 11: Users

'WasteRecords':

Column	Type	Description	Constraints
id	BIGINT	Primary key, unique identifier for the attachment.	Primary Key
post_id	BIGINT	Foreign key, references the ID of the post the attachment belongs to.	Foreign Key (posts.id)
name	VARCHAR(255)	The original name of the attached file.	Not Null
file_path	VARCHAR(255)	The path where the attached file is stored.	Not Null
mime_type	VARCHAR(20)	The MIME type of the attached file (e.g., 'image/jpeg', 'video/mp4').	Not Null
created_by	BIGINT	Foreign key, references the ID of the user who uploaded the attachment.	Foreign Key (users.id)
created_at	TIMESTAMP	Timestamp indicating when the attachment was created.	Not Null

Table 12: WasteRecords'

References

- Vishwark. (2025). React 18 Features You Must Understand in 2025 (Deep Explanation + Real Insights). https://dev.to/vishwark/react-18-features-you-must-understand-in-2025-deep-explanation-real-insights-8e1
- Kacper, Rafalski (2025). The Future of React: Top Trends Shaping Frontend Development in 2025. https://www.netguru.com/blog/react-js-trends
- Why is Tailwind CSS better than other CSS framework. https://windframe.dev/blog/why-tailwind-css-is-good
- TailwindCSS Documentation. https://v3.tailwindcss.com/docs/optimizing-for-production
- Paul, Bratslavsky (2025). Top 5 Chart Libraries to use in Your Next Project. https://strapi.io/blog/chart-libraries
- Richa (2025). .NET 8: A Comprehensive Overview and New Features. https://eluminoustechnologies.com/blog/net-8-features/
- Nisarg, Rami (2025). Benefits of Using React with ASP .Net Core. https://www.c-metric.com/blog/using-react-with-asp-net-core/
- Stephen, Planck (2025). Why SQL Server Provides Value for the Enterprise: A Comparative Look at PostgreSQL. https://www.sqltabletalk.com/?p=722