**Technical details**

* **Overview of BRM System**
* BRM is a modern .Net Core (.Net 5) Server Application.
* The web pages uses Bootstrap 5 css and does not have any further libraries or third party component dependencies.
* Depends on .Net Core Hosting Bundle for deployment.
* BRM Security is controlled by Active Directory
* The System comprises of 4 Projects:
* Sassa.BRM – Main system SPA
* Sassa.Data – Datalayer using EntityFramework with an Oracle connector.
* Razor.Components – Custom UI components used on the Website (Pager, Tooltip etc.)
* Sassa.Barcoder – Creates SVG barcodes and QR Codes used in the system
* **Sassa.BRM**
* **Services**
* The BRM system implements dependency injection for all components and functions.
* The following Services are defined in the \Services folder
* **BarCodeService** – Implements the barcode creation logic.
* **BRMDbService** – Impliments all the db calls
* All business Logic implemented in this Service.
* All db calls are async.
* Errors are thrown to the UI that will wrap them in a Alert Message.
* **CSService** – Interface with ContentServer to fetch Document Images.
* **EmailClient** – for sending confirmation mails and TDW file requests.
* **MailMessages** – Formats for standard email messages.
* **RawSQLService** – Calls to Socpen data and other ad hoc DB’s.
* **ReportDataService** – SQL used for reports.
* **StaticDataService** – All static data is cached here for access by all clients of the app.
* **SvgRenderer** – Renders SVG components (Barcodes/charts)
* **Timed Service** – An background service that runs on a timer. Currently used to synchronize Kofax scans with the File requests (Picklist Item status)
* **UI**
* The UI consists of Pages, Components, Modals and Print pages inn the \Pages folder with the following Structure:
* Pages
* **Index** – Main landing page with user info.
* **SearchApplicant** – Main page for capturing BRM records based on a list of SOCPEN records for an Applicant Id. Creation of the file Coversheet.
* **Batching**
* LOBatch – batching.
* BatchWaybilll – Waybill list page
* TDW Waybill (print dispatch)
* Receipt
* **EnquiryPage** – General BRM enquiries over BRM, SOCPEN and TDW data
* **Reboxing** - Ready batchs for Scanning and dispatch to TDW
* **FileRequest** – Requesting files form Contentserver(Digital) or TDW(Physical files)
* SearchFiles
* View digital images.
* ViewRequests
* List of active file requests
* Request processing
* Picklist status – Picklist progress page and manual receipt/return functions.
* **ReportPage** – Report request page. All report are prepared async and saved to the report download folder.
* Destruction - Destruction Process
* Exclusions - Capture Id's to be excluded
* Approval - Stakeholders to approve each region's Exclusions to be complete.
* Export Destruction list - For use by TDW in the destruction process.

**Copilot:**

**# Sassa.Brm Technical Overview**

**## Table of Contents**

**1. [Project Overview](#project-overview)**

**2. [Architecture](#architecture)**

**3. [Key Components](#key-components)**

**4. [Dependencies](#dependencies)**

**5. [Logging](#logging)**

**6. [Build & Deployment](#build--deployment)**

**7. [Best Practices](#best-practices)**

**---**

**## Project Overview**

**\*\*Sassa.Brm\*\* is a Blazor-based web application targeting .NET 9.0. It is designed for the South African Social Security Agency (SASSA) to manage, process, and report on BRM (Business Resource Management) data. The application leverages modern .NET technologies, Entity Framework Core, and integrates with various internal and external services.**

**---**

**## Architecture**

**- \*\*Frontend:\*\* Blazor (Server-side), using Razor components for UI.**

**- \*\*Backend:\*\* ASP.NET Core, Entity Framework Core for data access.**

**- \*\*Service Integration:\*\* WCF (via Connected Services), Oracle DB (via Oracle.ManagedDataAccess), and custom service libraries.**

**- \*\*Logging:\*\* Serilog for structured file and console logging.**

**- \*\*Authentication:\*\* Windows Authentication (Negotiate).**

**### Project Structure**

**- \*\*Components:\*\* UI logic and rendering (Razor files).**

**- \*\*Services:\*\* Business logic, data access, and integration with external systems.**

**- \*\*Helpers:\*\* Utility classes for authentication, logging, etc.**

**- \*\*Data:\*\* Entity models and view models.**

**---**

**## Key Components**

**- \*\*Blazor Components:\*\***

**UI elements for data capture, reporting, and user interaction (e.g., `FileCaptureSocpen.razor`, `TdwDispatchHistory.razor`).**

**- \*\*Services:\*\***

**- `CSService`: Handles Content Server document management and integration.**

**- `ApplicationService`: Business logic for application validation and BRM record management.**

**- `SessionService`, `ReportDataService`, `TdwBatchService`: Support user session, reporting, and batch operations.**

**- \*\*Helpers:\*\***

**- `WindowsAuthenticationStateProvider`: Integrates Windows authentication with Blazor.**

**- `Log`: (Legacy) JSON file-based logging utility.**

**---**

**## Dependencies**

**### NuGet Packages**

**- `Microsoft.AspNetCore.Components.QuickGrid` (UI grid)**

**- `Microsoft.EntityFrameworkCore.Relational` (EF Core)**

**- `Serilog`, `Serilog.AspNetCore`, `Serilog.Sinks.File`, `Serilog.Sinks.Console` (Logging)**

**- `Oracle.ManagedDataAccess.Client` (Oracle DB)**

**- `PDFsharp` (PDF generation)**

**- `System.ServiceModel.\*` (WCF support)**

**### Project References**

**- `Sassa.Brm.Common`, `Sassa.BRM.Data`, `Sassa.Barcoder`, `Sassa.CsService`, `Sassa.PdfServices`, `Sassa.QrCoder`, `Sassa.Socpen.Data`, `razor.Components`**

**---**

**## Logging**

**- \*\*Provider:\*\* Serilog (configured in `Program.cs`)**

**- \*\*Sinks:\*\* File (rolling daily logs in `Logs/app-.log`), Console**

**- \*\*Usage:\*\* Inject `ILogger<T>` into services/components for structured logging.**

**- \*\*Legacy:\*\* A custom `Log` class exists but is superseded by Serilog.**

**\*\*Sample Serilog Setup:\*\***