Payroll System: Functional Requirements Report

Project Overview

This system is a Blazor WebAssembly + ASP.NET Core API solution for managing employee master data and monthly payroll uploads through Excel files. The goal is to provide a validated, report-driven platform with tight error checks before committing any financial or statutory data.

Technologies

- **Frontend**: Blazor WebAssembly (Web & MAUI Hybrid)
- **Backend**: ASP.NET Core Web API
- **Database**: SQL Server (EF Core)
- **Excel Parser**: ClosedXML
- **Hosting**: Azure (App Service + SQL Database)

Phase 1: Employee Master Upload

Upload Fields (Excel)

- Name *(Required)*
- Joining Date *(Required)*
- PF Number *(Required if company setting enables it)*
- ESI Number *(Required if company setting enables it)*
- Aadhaar Number *(Always Required)*

Validation Rules

- Name and Joining Date must be present
- Aadhaar is mandatory for all employees
- PF and ESI fields are conditionally required per company setting
- PF and ESI must be unique:
- Within the uploaded file
- Compared against existing `Employee` records in DB

Upload Behavior

- Uploads are saved to a staging table: `EmployeeUploadStaging`
- Each upload is identified by a unique `UploadId`
- Previous uncommitted data for the same company is removed on new upload
- Each row is annotated with validation errors (if any)
- On new upload and successful commit, **previous Employees records for that company are wiped** and replaced by the new data.

Commit Logic

- Allowed only if all rows have no error

- Valid records are moved to the `Employees` table
- Staging rows are marked `Committed = true`

Additional Features

- Full CRUD operations must be implemented for **Employee** and **Company** management

---

Phase 2: Payroll Upload (Per Month, Per Company)

Upload Fields (Excel)

- PF/ESI Number *(Must match an employee)*
- Working Days *(Required)*
- Basic + DA *(Required)*
- Gross Salary *(Required)*
- Reason *(Required if Working Days = 0)*

Validation Rules

- PF/ESI must exist in the committed `Employees` table
- Each employee must appear once per month
- Missing employees (present in DB, not in Excel, no leaving date) are flagged
- NCP = Payroll Month Days - Working Days (auto-calculated)
- Reason is required when Working Days = 0

Upload Behavior

- Uploads are saved to a staging table: `PayrollUploadStaging`
- Each row includes error messages (if any)
- Only clean data can be committed to `PayrollEntry` table
- Previous uncommitted records for the same company/month are deleted on new upload
- Payroll process should also identify and include **missing employees** from DB who are not in Excel and have no leaving date

---

Reporting Requirements

PF and ESI Reports

- Generated from committed payroll data
- Only available if **no errors exist** in the latest upload for that company/month
- Errors in payroll staging block report generation

Blocking Conditions

- **Payroll upload** is blocked if `EmployeeUploadStaging` has any uncommitted errors
- **PF/ESI Report Download** is blocked if `PayrollUploadStaging` has uncommitted errors

---

Summary of Key Tables

`EmployeeUploadStaging`

- UploadId, CompanyId, RowNumber
- Name, JoiningDate, PFNumber, ESINumber, Aadhaar
- Error, Committed, UploadedAt

`Employees`

- EmployeeId, CompanyId, Name, JoiningDate
- PFNumber, ESINumber, Aadhaar

`PayrollUploadStaging`

- UploadId, CompanyId, PayrollMonth
- RowNumber, PF/ESI, WorkingDays, BasicDA, GrossSalary
- NCP, Reason, Error, Committed

`PayrollEntry`

- EmployeeId, CompanyId, PayrollMonth
- WorkingDays, BasicDA, GrossSalary, NCP, Reason

Next Steps

- Add Aadhaar field validation to employee upload logic
- Implement `PayrollUploadStaging` table and controller
- Enforce PF/ESI report download blocking logic based on error-free payroll
- Implement CRUD endpoints for Employee and Company entities
- Ensure `Employees` table is purged and refreshed on new committed upload