Software Requirement Specification (SRS)

Employee Management Module

Product

Pippa Enterprise Resource Planning Software

Version

1.0

Prepared By

Md. Zannatul Haque Siam
Project Lead

1. Introduction

1.1 Purpose

The purpose of this document is to define the functional and non-functional requirements for the Employee Management Module.

The module will serve as the central repository for all employee-related information throughout the employee lifecycle.

1.2 Scope

The Employee Management Module shall support:

- Employee Registration
- Employee Profile Management
- Organization Assignment
- Reporting Structure
- Education Management
- Experience Management
- Document Management
- Asset Tracking

- Employee Lifecycle Management
- Employee Timeline
- Audit Trail

2. Business Objectives

The system shall:

- Centralize employee records
- Reduce manual HR operations
- Improve employee data accuracy
- Enable employee lifecycle tracking
- Provide complete auditability
- Support configurable company structures

3. User Roles

Super Admin

Can access all companies.

HR Admin

Can manage all employees within the assigned company.

HR Executive

Can create and update employee records.

Department Head

Can view employees within the department.

Team Lead

Can view assigned team members.

Employee

Can view my own profile.

Epic: Employee Management Module

Story 1: Employee List Page

Story Title:
As an HR user, I want to view all employees in a centralized list so that I can easily search, filter, and manage employee records.

Description:
The system will provide an employee list page where authorized users can view employee records with key information, search employees, apply filters, sort columns, and perform quick actions.

Functional Requirements:

- Show employee list in table format.
- Display Employee ID, Name, Photo, Department, Designation, Branch, Reporting Manager, Employment Type, Status, Joining Date.
- Support server-side pagination.
- Support search by employee name, employee ID, official email, mobile number.
- Support filters by company, branch, department, team, designation, status, employment type, reporting manager, joining date range.
- Support sorting by Employee ID, Name, Joining Date, Department.
- Show action buttons based on permission.

Permissions:

- employee.view
- employee.create
- employee.update
- employee.export

Acceptance Criteria:

- User can see employee list after opening the page.
- Search returns correct employees.
- Multiple filters work together.
- Pagination works correctly.
- Sorting works correctly.
- User without permission cannot see restricted actions.
- Export action is logged in audit trail.

Story 2: Create Employee Wizard

Story Title:
As an HR user, I want to create a new employee using a step-by-step form so that employee information can be entered in a structured way.

Description:
System will provide a multi-step employee creation wizard. HR can enter personal, contact, employment, organization, bank, education, experience, and document information.

Form Steps:

1. Personal Information
2. Contact Information
3. Address Information
4. Employment Information
5. Organization Assignment
6. Reporting Structure
7. Bank Information
8. Education
9. Experience
10. Documents
11. Review & Submit

Functional Requirements:

- Employee can be saved as draft.
- Employee ID will be auto-generated based on configuration.
- Required fields will be configurable.
- Duplicate official email, mobile number, employee ID, and NID should be prevented.
- User can move back and forth between steps.
- Final review page should show all entered data before submission.

Acceptance Criteria:

- HR can create employee successfully.
- HR can save incomplete employee as draft.
- System prevents duplicate official email.
- System prevents duplicate mobile number.
- Employee ID generates automatically.
- Validation messages are clear.
- Audit log is created after employee creation.

Story 3: Employee Personal Information

Story Title:
As an HR user, I want to manage employee personal information so that employee identity data is stored properly.

Fields:

- Employee ID
- First Name
- Last Name
- Full Name
- Preferred Name
- Gender
- Date of Birth
- Blood Group
- Marital Status
- Nationality
- Religion optional
- Employee Photo

Validation Rules:

- First name is required.
- Last name is required.
- Gender is required.
- Date of birth cannot be future date.
- Minimum age should be configurable.
- Photo format: JPG, JPEG, PNG, WEBP.
- Photo size limit should be configurable.

Acceptance Criteria:

- HR can add personal information.
- HR can update personal information.
- Invalid DOB is blocked.
- Invalid photo format is blocked.
- Every update creates audit log with old and new value.

Story 4: Employee Contact Information

Story Title:
As an HR user, I want to store employee contact information so that company can communicate with the employee properly.

Fields:

- Official Email
- Personal Email
- Mobile Number
- Alternative Mobile Number
- Emergency Contact Name
- Emergency Contact Phone
- Emergency Contact Relation

Validation Rules:

- Official email must be unique.
- Mobile number must be unique.
- Email format must be valid.
- Phone number format should follow configured country rules.
- Emergency contact phone cannot be same as employee mobile if company restricts it.

Acceptance Criteria:

- HR can save contact information.
- Duplicate official email is blocked.
- Duplicate mobile number is blocked.
- Invalid email format is blocked.
- Contact update is logged.

Story 5: Employee Address Management

Story Title:
As an HR user, I want to store employee present and permanent address so that employee location information is properly maintained.

Address Types:

- Present Address
- Permanent Address

Fields:

- Address Line 1
- Address Line 2
- Country
- Division
- District

- Upazila/City
- Postal Code

Functional Requirements:

- User can select “Same as Present Address”.
- System should copy present address into permanent address.
- Country/division/district dropdown should be configurable.
- Address fields can be required based on company configuration.

Acceptance Criteria:

- HR can save present address.
- HR can save permanent address.
- “Same as Present Address” works correctly.
- Address update creates audit log.

Story 6: Employee Identity Information

Story Title:
As an HR user, I want to store employee identity documents so that legal and verification information is available.

Supported Identity Types:

- NID
- Passport
- Birth Certificate
- Driving License
- TIN

Fields:

- Document Type
- Document Number
- Issue Date
- Expiry Date
- Attachment
- Remarks

Functional Requirements:

- Multiple identity records can be added.
- Identity document number uniqueness should be configurable.
- Expiry date should trigger notification if configured.

- Attachment upload should be supported.

Acceptance Criteria:

- HR can add identity document.
- HR can update identity document.
- HR can upload attachment.
- Duplicate identity number is blocked if uniqueness is enabled.
- Expiry reminder works if enabled.

Story 7: Employment Information

Story Title:
As an HR user, I want to store employment details so that employee service information can be tracked.

Fields:

- Joining Date
- Confirmation Date
- Employment Type
- Employee Category
- Probation Period
- Probation End Date
- Work Mode
- Employment Status

Functional Requirements:

- Probation end date can be auto-calculated from joining date.
- Employment type should come from dynamic configuration.
- Status should follow employee lifecycle.
- Confirmation date cannot be before joining date.

Acceptance Criteria:

- HR can save employment information.
- Probation end date auto-calculates.
- Invalid confirmation date is blocked.
- Status change creates timeline event.
- Employment update is logged.

Story 8: Organization Assignment

Story Title:
As an HR user, I want to assign employee to company structure so that organization hierarchy is maintained.

Fields:

- Company
- Branch
- Division
- Department
- Team
- Section
- Designation
- Grade
- Cost Center
- Work Location

Functional Requirements:

- Dropdown values should come from organization configuration.
- Department selection should filter team list.
- Designation should be linked with grade if configured.
- Previous assignment history should be stored.
- Transfer should create timeline event.

Acceptance Criteria:

- HR can assign employee to department.
- Team list changes based on department.
- Previous assignment history is stored.
- Organization change creates audit log.
- Organization change creates timeline event.

Story 9: Reporting Structure

Story Title:
As an HR user, I want to assign reporting managers and team leads so that approval and hierarchy workflows work properly.

Manager Types:

- Reporting Manager

- Team Lead
- Functional Manager
- Matrix Manager

Functional Requirements:

- One employee can have one reporting manager.
- One employee can have multiple team leads if enabled.
- Manager cannot be the same employee.
- Circular reporting should be prevented.
- Manager change should create history.

Acceptance Criteria:

- HR can assign reporting manager.
- HR can assign multiple team leads.
- Employee cannot report to himself.
- Circular hierarchy is blocked.
- Manager change creates timeline and audit log.

Story 10: Employee Bank Information

Story Title:
As an HR/payroll user, I want to manage employee bank information so that salary can be disbursed correctly.

Fields:

- Bank Name
- Branch Name
- Account Name
- Account Number
- Routing Number
- Mobile Banking Provider
- Mobile Banking Number
- Is Primary

Functional Requirements:

- Bank information visibility must be permission based.
- Sensitive changes may require approval.
- Only one primary bank account should be allowed.
- Account number should be encrypted or masked in UI if configured.

Acceptance Criteria:

- Authorized user can add bank information.
- Unauthorized user cannot view bank information.
- Only one primary account exists.
- Bank change can trigger approval workflow.
- Bank change is logged in audit trail.

Story 11: Employee Tax Information

Story Title:
As an HR/payroll user, I want to store employee tax information so that payroll tax calculation can use correct data.

Fields:

- TIN
- Tax Zone
- Tax Circle
- Tax Exemption Status
- Tax Remarks

Functional Requirements:

- TIN should be unique if configured.
- Tax fields should be visible only to authorized users.
- Tax data should be available for payroll module.

Acceptance Criteria:

- Authorized user can add tax information.
- Unauthorized user cannot view tax information.
- Duplicate TIN is blocked if configured.
- Tax update creates audit log.

Story 12: Employee Education Management

Story Title:
As an HR user, I want to manage employee education records so that academic qualification history is maintained.

Fields:

- Degree

- Institution
- Board/University
- Passing Year
- Result
- Major/Group
- Certificate Attachment

Functional Requirements:

- Multiple education records allowed.
- Certificate upload allowed.
- Passing year cannot be future year.
- User can add/edit/delete education records.

Acceptance Criteria:

- HR can add multiple education records.
- HR can edit education record.
- HR can delete education record.
- Future passing year is blocked.
- Certificate upload works.

Story 13: Employee Experience Management

Story Title:
As an HR user, I want to manage employee previous experience so that professional history is maintained.

Fields:

- Company Name
- Designation
- Department
- Joining Date
- Resignation Date
- Responsibilities
- Experience Certificate

Functional Requirements:

- Multiple experience records allowed.
- Resignation date cannot be before joining date.
- System should calculate experience duration.
- Attachment upload supported.

Acceptance Criteria:

- HR can add experience.
- HR can edit experience.
- HR can delete experience.
- Duration is calculated automatically.
- Invalid date range is blocked.

Story 14: Employee Document Management

Story Title:
As an HR user, I want to upload and manage employee documents so that employee files are stored digitally.

Document Types:

- CV
- NID
- Passport
- Certificate
- Offer Letter
- Appointment Letter
- NDA
- Contract
- Other

Functional Requirements:

- Document types should be dynamic.
- File upload, preview, download, replace, delete supported.
- Version history should be maintained.
- Expiry date should be supported.
- Document access should be permission based.

Acceptance Criteria:

- HR can upload document.
- HR can preview document.
- HR can replace document.
- Previous version is stored.
- Unauthorized user cannot access restricted document.
- Expiry notification works if configured.

Story 15: Employee Asset Assignment

Story Title:
As an HR/Admin user, I want to assign assets to employees so that company assets can be tracked.

Asset Types:

- Laptop
- Desktop
- Mobile
- SIM
- ID Card
- Access Card
- Monitor
- Other

Functional Requirements:

- Employee can have multiple assets.
- Asset assignment date must be stored.
- Asset return date must be stored.
- Current asset status should be visible.
- Asset history should be maintained.

Acceptance Criteria:

- User can assign asset to employee.
- User can return asset.
- Asset history is visible.
- Returned asset no longer appears as active.
- Asset assignment creates timeline event.

Story 16: Employee Profile Page

Story Title:
As an authorized user, I want to view a complete employee profile so that I can see all employee information from one place.

Profile Tabs:

- Overview
- Personal
- Contact
- Address

- Employment
- Organization
- Reporting Structure
- Education
- Experience
- Bank
- Tax
- Documents
- Assets
- Notes
- Timeline
- Audit Logs

Functional Requirements:

- Tabs should show based on permission.
- Sensitive tabs should be hidden for unauthorized users.
- Overview should show key summary.
- Profile completion percentage should be displayed.

Acceptance Criteria:

- Employee profile loads correctly.
- User sees only permitted tabs.
- Sensitive information is hidden.
- Profile completion percentage is calculated.
- Recent activities are visible.

Story 17: Employee Timeline

Story Title:
As an HR user, I want to view employee timeline so that I can track all important employee lifecycle events.

Timeline Events:

- Joined
- Confirmed
- Promoted
- Transferred
- Manager Changed
- Salary Changed
- Document Uploaded
- Asset Assigned

- Warning
- Resigned
- Terminated

Functional Requirements:

- Timeline should be chronological.
- Timeline can be filtered by event type.
- Timeline can be filtered by date range.
- System events should be auto-created.

Acceptance Criteria:

- Timeline displays events correctly.
- New employee creation creates joining event.
- Transfer creates transfer event.
- Promotion creates promotion event.
- Filters work correctly.

Story 18: Employee Notes

Story Title:
As an HR user, I want to add internal notes to employee profile so that HR-specific observations can be recorded.

Fields:

- Note
- Visibility
- Attachment
- Mentioned Users

Visibility Options:

- Private
- HR Only
- Manager Visible

Functional Requirements:

- Notes should be permission based.
- User can add attachment.
- User can mention another user.
- Note edit/delete permission should be configurable.

Acceptance Criteria:

- HR can add note.
- HR can edit own note if allowed.
- HR can delete note if allowed.
- Unauthorized users cannot see private notes.
- Note creation is logged.

Story 19: Employee Audit Log

Story Title:
As an admin/auditor, I want to view employee audit logs so that all changes can be tracked for compliance.

Tracked Events:

- Create
- Update
- Delete
- Status Change
- Manager Change
- Bank Change
- Tax Change
- Document Upload
- Document Delete
- Export

Log Fields:

- Employee ID
- Action
- Module
- Field Name
- Old Value
- New Value
- Performed By
- Performed At
- IP Address

Acceptance Criteria:

- Every important change creates audit log.
- Audit log shows old and new values.
- Audit log cannot be edited from UI.

- Authorized user can filter audit logs.
- Unauthorized user cannot access audit logs.

Story 20: Employee Import

Story Title:
As an HR user, I want to import employees from Excel/CSV so that bulk employee creation becomes easier.

Functional Requirements:

- Support XLSX and CSV.
- Provide sample template.
- Allow column mapping.
- Validate data before import.
- Show validation errors.
- Prevent duplicate records.
- Support partial success or full rollback based on configuration.

Acceptance Criteria:

- User can download sample template.
- User can upload Excel/CSV.
- System validates data before import.
- Error report is generated.
- Valid employees are imported.
- Import action is logged.

Story 21: Employee Export

Story Title:
As an HR user, I want to export employee data so that I can use employee information outside the system.

Export Formats:

- Excel
- CSV
- PDF

Functional Requirements:

- User can select columns.
- Export should respect current filters.

- Sensitive fields should export only with permission.
- Export action must be logged.

Acceptance Criteria:

- User can export employee list.
- Exported file respects applied filters.
- Unauthorized user cannot export salary/bank/tax data.
- Export audit log is created.

Story 22: Employee Advanced Search

Story Title:
As an HR user, I want advanced search and filters so that I can quickly find specific employees.

Searchable Fields:

- Employee Name
- Employee ID
- Official Email
- Mobile Number
- NID
- Department
- Designation
- Reporting Manager

Filter Options:

- Company
- Branch
- Department
- Designation
- Status
- Employment Type
- Joining Date
- Confirmation Date
- Work Mode

Acceptance Criteria:

- Search works with partial text.
- Multiple filters work together.
- Search result updates correctly.
- Filter reset works.

- Query performance remains acceptable.

Story 23: Saved Filters

Story Title:
As an HR user, I want to save frequently used employee filters so that I can reuse them later.

Functional Requirements:

- User can save current filters.
- User can rename saved filters.
- User can delete saved filters.
- User can set default filter.
- User can share filter if permission exists.

Acceptance Criteria:

- User can save filter.
- Saved filter can be applied.
- Default filter loads automatically.
- User can delete saved filter.
- Shared filter respects permissions.

Story 24: Employee Dynamic Custom Fields

Story Title:
As an admin, I want to create custom employee fields so that company-specific employee data can be stored without code changes.

Supported Field Types:

- Text
- Number
- Date
- Dropdown
- Radio
- Checkbox
- Multi-select
- File Upload

Functional Requirements:

- Admin can create custom field.
- Admin can set field required/optional.

- Admin can define dropdown options.
- Admin can decide section/tab placement.
- Custom field values should appear in employee profile.

Acceptance Criteria:

- Admin can create custom field.
- Custom field appears in employee form.
- Required custom field validation works.
- Custom field value saves correctly.
- Custom field update creates audit log.

Story 25: Employee Custom Tabs

Story Title:
As an admin, I want to create custom tabs in employee profile so that company-specific sections can be maintained.

Functional Requirements:

- Admin can create tab.
- Admin can edit tab name.
- Admin can reorder tabs.
- Admin can assign custom fields to tab.
- Admin can hide/show tab.

Acceptance Criteria:

- Custom tab appears in employee profile.
- Tab order follows configuration.
- Hidden tab is not visible.
- Custom fields show under assigned tab.

Story 26: Employee Approval Queue

Story Title:
As an approver, I want to review employee-related changes so that sensitive updates are approved before applying.

Approval Items:

- New employee approval
- Bank information change
- Department change

- Designation change
- Reporting manager change
- Salary-related change

Functional Requirements:

- Pending requests should appear in approval queue.
- Approver can approve.
- Approver can reject with remarks.
- Approver can return for correction.
- Approved changes should apply to employee profile.
- Rejected changes should not update master data.

Acceptance Criteria:

- Pending request appears in queue.
- Approver can approve request.
- Approver can reject request.
- Rejection remarks are required.
- Approved change updates employee profile.
- Approval action creates audit log.

Story 27: Employee Permission Matrix

Story Title:
As an admin, I want role-based permissions for employee module so that data access can be controlled properly.

Permission Types:

- View Employee
- Create Employee
- Edit Employee
- Delete Employee
- Export Employee
- Import Employee
- Approve Employee
- View Bank Info
- View Tax Info
- View Documents
- View Audit Logs
- Manage Custom Fields

Functional Requirements:

- Permissions should be role based.
- Sensitive fields should support field-level permissions.
- Action buttons should hide if permission missing.
- Backend API should also enforce permission.

Acceptance Criteria:

- User sees only allowed actions.
- Unauthorized API request is blocked.
- Sensitive fields are hidden.
- Permission changes reflect immediately after login/session refresh.

Story 28: Employee Dashboard

Story Title:
As an HR user, I want an employee dashboard so that I can quickly understand employee statistics.

Dashboard Widgets:

- Total Employees
- Active Employees
- Probation Employees
- Confirmed Employees
- Resigned Employees
- Department-wise employee count
- Branch-wise employee count
- Gender ratio
- Monthly joiners
- Monthly resignations

Acceptance Criteria:

- Dashboard loads employee statistics.
- Data respects company/branch permission.
- Charts show correct data.
- Filters update dashboard data.

Story 29: Employee Module Configuration

Story Title:
As an admin, I want to configure employee module settings so that the system can support different company policies.

Configuration Areas:

- Employee ID format
- Employment types
- Employee statuses
- Document types
- Required fields
- Custom fields
- Custom tabs
- Profile layout
- Validation rules
- Approval rules

Acceptance Criteria:

- Admin can update employee ID format.
- Admin can manage employment types.
- Admin can manage employee statuses.
- Admin can manage document types.
- Required field configuration works.
- Configuration changes affect employee form dynamically.

Database design Specification

Database design Specification

1. Purpose

This chapter defines the database architecture for the Employee Management Module. It covers table design, key columns, relationships, indexes, multi-company data isolation, audit trail, custom field architecture, soft delete strategy, and future scalability considerations.

2. Design Principles

The database must be normalized, scalable, auditable, and suitable for a multi-company SaaS HRMS. Employee data should not be stored in one very large table. Instead, data should be split by domain so that security, performance, and maintainability remain strong.

3. Core Database Principles

- Every employee must belong to one company using company_id.
- Company-level data isolation is mandatory for SaaS architecture.
- Employee master table should store only core identity/profile data.
- Separate tables should be used for contacts, addresses, employment, reporting, documents, assets, education, experience, bank and tax records.

- Historical records must be maintained for organization assignment, manager assignment, lifecycle events and critical changes.
- Soft delete should be used instead of physical delete for employee master data.
- Critical updates must create immutable audit logs.
- Custom fields must use metadata/value architecture instead of altering employee table columns dynamically.
- Frequently filtered fields must be indexed for performance.

4. Recommended Entity List

Table	Purpose
employees	Main employee master profile.
employee_contacts	Employee email, phone and emergency contact.
employee_addresses	Present and permanent addresses.
employee_identities	NID, passport, TIN, driving license and other legal identity records.
employee_employments	Employment and job service information.
employee_organization_assignments	Current and historical organization placement.
employee_reporting_managers	Reporting manager, team lead, functional manager and matrix manager relationships.
employee_educations	Education history.
employee_experiences	Previous employment history.
employee_bank_accounts	Payroll bank details.
employee_tax_profiles	Tax information.
employee_documents	Employee document metadata and versions.
employee_assets	Asset assignment and return tracking.
employee_notes	Internal HR notes.
employee_timelines	Immutable employee lifecycle timeline.
employee_custom_fields	Custom field metadata.
employee_custom_field_options	Dropdown/radio/multi-select options.
employee_custom_field_values	Custom field values.
employee_audit_logs	Immutable audit trail.
employee_status_histories	Employee status change history.
employee_import_batches	Import operation batch header.
employee_import_errors	Import validation errors.

5.1 Table: employees

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Required. Company isolation.
employee_no	VARCHAR(50)	Unique employee ID within company.
first_name	VARCHAR(100)	Required.
last_name	VARCHAR(100)	Required.
preferred_name	VARCHAR(100) NULL	Optional.
gender	VARCHAR(30)	Configured dropdown.
date_of_birth	DATE	DOB validation required.
blood_group	VARCHAR(10) NULL	Optional.
marital_status	VARCHAR(30) NULL	Optional.
nationality	VARCHAR(80) NULL	Optional/default country.
religion	VARCHAR(80) NULL	Optional/configurable.
photo_path	VARCHAR(255) NULL	Employee photo path.
status	VARCHAR(50)	Draft, active, probation, confirmed,

Column	Data Type	Description / Rule
		etc.
profile_completion_percentage	DECIMAL(5,2) DEFAULT 0	Computed/stored profile completion.
created_by	BIGINT UNSIGNED NULL	Creator user ID
updated_by	BIGINT UNSIGNED NULL	Last updater user ID
deleted_at	TIMESTAMP NULL	Soft delete
created_at	TIMESTAMP	Created time
updated_at	TIMESTAMP	Updated time

5.2 Table: employee_contacts

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
official_email	VARCHAR(150)	Unique within company
personal_email	VARCHAR(150) NULL	Optional
mobile_no	VARCHAR(30)	Unique if configured
alternative_mobile_no	VARCHAR(30) NULL	Optional
emergency_contact_name	VARCHAR(150) NULL	Optional/configurable
emergency_contact_phone	VARCHAR(30) NULL	Optional/configurable
emergency_contact_relation	VARCHAR(80) NULL	Configured dropdown
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.3 Table: employee_addresses

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
address_type	VARCHAR(30)	present/permanent
country_id	BIGINT UNSIGNED NULL	Country master
division_id	BIGINT UNSIGNED NULL	Division/state master
district_id	BIGINT UNSIGNED NULL	District/city master
upazila_or_city	VARCHAR(100) NULL	Optional
postal_code	VARCHAR(20) NULL	Postal code
address_line_1	TEXT	Main address
address_line_2	TEXT NULL	Optional
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.4 Table: employee_identities

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
document_type	VARCHAR(50)	nid/passport/tin/etc.
document_number	VARCHAR(100)	Identity number
issue_date	DATE NULL	Optional
expiry_date	DATE NULL	Required for expiry-tracked

Column	Data Type	Description / Rule
		documents
attachment_path	VARCHAR(255) NULL	File path
remarks	TEXT NULL	Optional remarks
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.5 Table: employee_employments

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
joining_date	DATE	Required
confirmation_date	DATE NULL	After joining date
employment_type_id	BIGINT UNSIGNED NULL	Dynamic employment type
employee_category_id	BIGINT UNSIGNED NULL	Dynamic category
work_mode	VARCHAR(50)	office/remote/hybrid/field
probation_period_value	INT NULL	Probation duration value
probation_period_unit	VARCHAR(20) NULL	days/months
probation_end_date	DATE NULL	Auto-calculated
status	VARCHAR(50)	Employment status
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.6 Table: employee_organization_assignments

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
branch_id	BIGINT UNSIGNED NULL	Branch
division_id	BIGINT UNSIGNED NULL	Division
department_id	BIGINT UNSIGNED NULL	Department
team_id	BIGINT UNSIGNED NULL	Team
section_id	BIGINT UNSIGNED NULL	Section
designation_id	BIGINT UNSIGNED NULL	Designation
grade_id	BIGINT UNSIGNED NULL	Grade
cost_center_id	BIGINT UNSIGNED NULL	Cost center
work_location_id	BIGINT UNSIGNED NULL	Office/work location
effective_from	DATE	Effective start date
effective_to	DATE NULL	End date. Null means current.
is_current	BOOLEAN DEFAULT TRUE	Current assignment flag
change_reason	TEXT NULL	Transfer/promotion reason
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.7 Table: employee_reporting_managers

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED	Company isolation

Column	Data Type	Description / Rule
	FK	
employee_id	BIGINT UNSIGNED FK	Employee who reports
manager_id	BIGINT UNSIGNED FK	Manager employee ID
manager_type	VARCHAR(50)	reporting_manager/team_lead/functional_manager/matrix_manager
effective_from	DATE	Effective start
effective_to	DATE NULL	End date
is_current	BOOLEAN DEFAULT TRUE	Current flag
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.8 Table: employee_educations

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
degree	VARCHAR(150)	Degree name
institution	VARCHAR(200)	Institution
board_or_university	VARCHAR(150) NULL	Board/university
major	VARCHAR(150) NULL	Major/group
passing_year	YEAR NULL	Cannot be future
result	VARCHAR(50) NULL	Result/GPA
attachment_path	VARCHAR(255) NULL	Certificate path
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.9 Table: employee_experiences

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
company_name	VARCHAR(200)	Previous company
designation	VARCHAR(150) NULL	Previous designation
department	VARCHAR(150) NULL	Previous department
joining_date	DATE NULL	Start date
resignation_date	DATE NULL	End date
duration_months	INT NULL	Calculated duration
responsibilities	TEXT NULL	Responsibilities
attachment_path	VARCHAR(255) NULL	Experience certificate
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.10 Table: employee_bank_accounts

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference

Column	Data Type	Description / Rule
bank_name	VARCHAR(150)	Bank name
branch_name	VARCHAR(150) NULL	Branch name
account_name	VARCHAR(150)	Account holder name
account_number	VARCHAR(255)	Encrypted/masked in UI
routing_number	VARCHAR(50) NULL	Routing number
mobile_banking_provider	VARCHAR(80) NULL	bKash/Nagad/etc.
mobile_banking_number	VARCHAR(30) NULL	Optional
is_primary	BOOLEAN DEFAULT FALSE	Only one primary
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.11 Table: employee_tax_profiles

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
tin	VARCHAR(100) NULL	Unique if configured
tax_zone	VARCHAR(100) NULL	Tax zone
tax_circle	VARCHAR(100) NULL	Tax circle
tax_exemption	BOOLEAN DEFAULT FALSE	Exemption flag
remarks	TEXT NULL	Optional
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.12 Table: employee_documents

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
document_type_id	BIGINT UNSIGNED FK	Dynamic document type
file_name	VARCHAR(255)	Original file name
file_path	VARCHAR(255)	Storage path
mime_type	VARCHAR(100)	File MIME type
file_size	BIGINT UNSIGNED	File size bytes
version	INT DEFAULT 1	Document version
parent_document_id	BIGINT UNSIGNED NULL	Previous/root document
expiry_date	DATE NULL	Expiry tracking
is_current	BOOLEAN DEFAULT TRUE	Current version flag
uploaded_by	BIGINT UNSIGNED NULL	Uploader user ID
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.13 Table: employee_assets

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
asset_id	BIGINT UNSIGNED FK	Asset module reference
assigned_date	DATE	Assignment date

Column	Data Type	Description / Rule
returned_date	DATE NULL	Return date
status	VARCHAR(50)	assigned/returned/damaged/lost
remarks	TEXT NULL	Optional
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.14 Table: employee_notes

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
note	TEXT	Note content
visibility	VARCHAR(50)	private/hr/manager
attachment_path	VARCHAR(255) NULL	Optional attachment
created_by	BIGINT UNSIGNED	Creator
updated_by	BIGINT UNSIGNED NULL	Updater
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.15 Table: employee_timelines

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
event_type	VARCHAR(80)	joined/promoted/transferred/etc.
event_title	VARCHAR(200)	Short title
event_description	TEXT NULL	Details
event_date	DATETIME	Event date
reference_type	VARCHAR(100) NULL	Related model type
reference_id	BIGINT UNSIGNED NULL	Related model ID
created_by	BIGINT UNSIGNED NULL	User/system
created_at	TIMESTAMP	Created

5.16 Table: employee_audit_logs

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED NULL	Target employee
module	VARCHAR(100)	Submodule
action	VARCHAR(80)	create/update/delete/export/etc.
field_name	VARCHAR(150) NULL	Changed field
old_value	LONGTEXT NULL	Previous value
new_value	LONGTEXT NULL	New value
performed_by	BIGINT UNSIGNED NULL	User ID
ip_address	VARCHAR(45) NULL	IPv4/IPv6
user_agent	TEXT NULL	Browser/app info
performed_at	DATETIME	Action time

5.17 Table: employee_custom_fields

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
tab_id	BIGINT UNSIGNED NULL	Custom tab placement
field_key	VARCHAR(100)	System-safe unique key
field_label	VARCHAR(150)	Display label
field_type	VARCHAR(50)	text/number/date/dropdown/etc.
is_required	BOOLEAN DEFAULT FALSE	Required flag
is_active	BOOLEAN DEFAULT TRUE	Active flag
sort_order	INT DEFAULT 0	Display order
validation_rules	JSON NULL	Dynamic validation
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

5.18 Table: employee_custom_field_values

Column	Data Type	Description / Rule
id	BIGINT UNSIGNED PK	Primary key
company_id	BIGINT UNSIGNED FK	Company isolation
employee_id	BIGINT UNSIGNED FK	Employee reference
custom_field_id	BIGINT UNSIGNED FK	Custom field
value_text	TEXT NULL	Text value
value_number	DECIMAL(18,4) NULL	Numeric value
value_date	DATE NULL	Date value
value_json	JSON NULL	Checkbox/multi-select/file value
created_at	TIMESTAMP	Created
updated_at	TIMESTAMP	Updated

6. Relationship Rules

- employees.company_id references companies.id.
- every child employee table references employees.id using employee_id.
- employee_organization_assignments stores current and historical organization placement.
- employee_reporting_managers.manager_id references employees.id and must belong to the same company.
- employee_documents should reference dynamic document type configuration.
- employee_assets.asset_id should reference Asset Module master table when Asset Module exists.
- audit and timeline tables should not be edited from normal UI after creation.

7. Index Strategy

Table	Index	Purpose
employees	company_id, employee_no	Unique employee lookup within company
employees	company_id, status	Status filter
employees	company_id, first_name, last_name	Name search support
employee_contacts	company_id, official_email	Unique official email within company
employee_contacts	company_id, mobile_no	Mobile search/unique if enabled

Table	Index	Purpose
employee_organization_assignments	employee_id, is_current	Current assignment lookup
employee_organization_assignments	company_id, department_id	Department filter
employee_reporting_managers	employee_id, manager_type, is_current	Current manager lookup
employee_documents	employee_id, document_type_id, is_current	Current document lookup
employee_timelines	employee_id, event_date	Timeline sort/filter
employee_audit_logs	employee_id, performed_at	Audit history
employee_custom_field_values	employee_id, custom_field_id	Custom field value lookup

8. Soft Delete Strategy

- Employee master records must use soft delete.
- Critical child records should generally not be physically deleted if they are part of compliance history.
- Document delete should mark document inactive/deleted and keep audit trail.
- Archived employees should remain available in historical reports but hidden from active lists by default.

9. Multi-Company SaaS Strategy

- Every query must include company scope.
- Unique constraints should usually be company scoped, for example unique(company_id, employee_no).
- Cross-company manager assignment must be blocked unless explicitly supported by enterprise group policy.
- Document storage path should include company identifier for isolation.
- Import/export must respect authenticated user company access.

10. Audit Strategy

- Audit logs must be immutable from UI.
- Create audit log for create, update, delete, export, import, approval, reject, status change, bank change, tax change, manager change and document operation.
- Old and new values should be stored as JSON/text where possible.
- Sensitive values may be masked or encrypted in audit logs based on security policy.

11. Custom Field Architecture

- Do not add new columns dynamically to employees table.
- Use employee_custom_fields for metadata.
- Use employee_custom_field_values for employee-specific values.
- Use typed value columns to improve query performance.
- Use JSON only for complex fields such as multi-select/file metadata.

12. Data Security Strategy

- Bank account and tax data require permission-based access.
- Sensitive values should be encrypted at application level when required.
- Masked display should be used for unauthorized or limited-access users.

- Export must exclude sensitive columns unless user has permission.

13. Migration Development Rules

- Create migrations in dependency order: master tables first, employee master second, child tables after that.
- Add foreign keys after referenced tables exist.
- Use nullable fields for optional future modules such as assets until their modules are complete.
- Add indexes during migration, not later as an afterthought.
- Use enum-like string columns or lookup tables depending on configuration requirement.

14. Future Scalability Considerations

- Employee list API should select only required fields, not full profile.
- Use eager loading carefully to avoid N+1 queries.
- Use separate read models or cached summaries for dashboard if employee count is very large.
- Consider database partitioning or archival strategy for audit logs in very large installations.
- Use queue jobs for import/export and document processing.

15. Laravel Model Relationship Summary

Model	Relationships
Employee	belongsTo Company; hasOne Contact; hasMany Addresses; hasMany Identities; hasOne Employment; hasMany OrganizationAssignments; hasMany ReportingManagers; hasMany Educations; hasMany Experiences; hasMany BankAccounts; hasOne TaxProfile; hasMany Documents; hasMany Assets; hasMany Notes; hasMany Timelines; hasMany AuditLogs.
EmployeeOrganizationAssignment	belongsTo Employee; belongsTo Branch; belongsTo Department; belongsTo Team; belongsTo Designation; belongsTo Grade.
EmployeeReportingManager	belongsTo Employee; belongsTo Manager using manager_id.
EmployeeDocument	belongsTo Employee; belongsTo DocumentType; belongsTo Uploader.
EmployeeCustomFieldValue	belongsTo Employee; belongsTo CustomField.

16. Acceptance Criteria for Database Design

- Database supports complete employee profile without using one overloaded employee table.
- All employee data is isolated by company_id.
- Employee ID is unique within company.
- Official email is unique within company.
- Organization assignment history can be tracked.
- Manager assignment history can be tracked.
- Timeline events can be stored independently.
- Audit logs can store old and new values for critical changes.

- Custom fields can be added without database schema changes.
- Employee documents support version history.
- Bank and tax information can be permission-controlled.
- Indexes exist for common search and filter operations.

API Specification

Base URL: /api/v1/hrms/employees
Auth: Bearer Token
Permission: Role Based
Response Format: JSON

API Groups

1. Employee Master API
2. Employee Profile API
3. Employment Information API
4. Contact & Address API
5. Education API
6. Experience API
7. Bank & Tax API
8. Document API
9. Asset API
10. Timeline API
11. Notes API
12. Audit Log API
13. Import/Export API
14. Configuration API

1. Employee Master APIs

1.1 Employee List

GET /api/v1/hrms/employees

Query Params

Search, company_id
branch_id
department_id
designation_id
team_id
status
employment_type
joining_date_from
joining_date_to
sort_by
sort_direction
per_page
page

Response


```
{
  "data": [
    {
      "id": 1,
      "employee_no": "EMP-0001",
      "name": "Rahim Uddin",
      "official_email": "rahim@company.com",
      "mobile_no": "01700000000",
      "department": "IT",
      "designation": "Software Engineer",
      "reporting_manager": "Karim Ahmed",
      "status": "active",
      "joining_date": "2026-01-01"
    }
  ],
  "meta": {
    "current_page": 1,
    "per_page": 20,
    "total": 150
  }
}
```


1.2 Create Employee

POST /api/v1/hrms/employees

Request


```
{
  "first_name": "Rahim",
  "last_name": "Uddin",
  "gender": "male",
  "date_of_birth": "1995-05-10",
  "official_email": "rahim@company.com",
  "mobile_no": "01700000000",
  "joining_date": "2026-01-01",
  "department_id": 1,
  "designation_id": 2,
  "employment_type": "permanent"
}
```


Rules

employee_no auto-generate hobe
official_email unique hobe
mobile_no unique hobe
joining_date required
department_id required
designation_id required

1.3 View Employee Profile
GET /api/v1/hrms/employees/{employee_id}

Response should include:

basic_info
contact_info
address_info
employment_info
reporting_structure
education
experience
bank_info
documents
assets
timeline
profile_completion

1.4 Update Employee
PUT /api/v1/hrms/employees/{employee_id}

Important
Sensitive fields update হলে approval trigger হবে:

department_id
designation_id
reporting_manager_id
bank_account
salary_structure
status

1.5 Change Employee Status
PATCH /api/v1/hrms/employees/{employee_id}/status

Request
```
{
  "status": "active",
  "effective_date": "2026-01-01",
  "remarks": "Employee activated after approval"
}
```

2. Contact APIs

2.1 Update Contact Information
PUT /api/v1/hrms/employees/{employee_id}/contact

Request
```
{
  "personal_email": "rahim.personal@gmail.com",
  "official_email": "rahim@company.com",
  "mobile_no": "01700000000",
  "alternative_mobile": "01800000000",
  "emergency_contact_name": "Karim Uddin",
  "emergency_contact_phone": "01900000000",
  "emergency_contact_relation": "Brother"
}
```

3. Address APIs

3.1 Store / Update Address
PUT /api/v1/hrms/employees/{employee_id}/addresses

Request
```
{
  "present": {
    "country": "Bangladesh",
    "division": "Dhaka",
    "district": "Dhaka",
    "postal_code": "1207",
    "address_line_1": "Mirpur",
    "address_line_2": "Dhaka"
  },
  "permanent": {
    "country": "Bangladesh",
    "division": "Dhaka",
    "district": "Gazipur",
    "postal_code": "1700",
    "address_line_1": "Kaliganj",
    "address_line_2": "Gazipur"
  }
}
```

4. Employment APIs

4.1 Update Employment Information
PUT /api/v1/hrms/employees/{employee_id}/employment

Request

```
{
  "joining_date": "2026-01-01",
  "confirmation_date": "2026-04-01",
  "employment_type": "permanent",
  "employee_category": "regular",
  "probation_end_date": "2026-04-01",
  "work_mode": "office"
}
```

4.2 Update Organization Assignment

PUT /api/v1/hrms/employees/{employee_id}/organization

Request

```
{
  "company_id": 1,
  "branch_id": 2,
  "department_id": 3,
  "team_id": 4,
  "designation_id": 5,
  "grade_id": 2,
  "cost_center_id": 1
}
```

System Behavior

Previous assignment history save korte hobe
Timeline event create korte hobe
Approval lagle pending state e rakhte hobe

4.3 Assign Reporting Managers

PUT /api/v1/hrms/employees/{employee_id}/reporting-managers

Request

```
{
  "reporting_manager_id": 10,
  "team_lead_ids": [12, 13],
  "functional_manager_id": 15,
  "matrix_manager_id": null
}
```

5. Education APIs

5.1 Add Education

POST /api/v1/hrms/employees/{employee_id}/educations

```
{
  "degree": "BSc in CSE",
  "institution": "ABC University",
  "board": "Dhaka",
  "passing_year": 2018,
  "result": "3.50"
}
```

5.2 Update Education
PUT /api/v1/hrms/employees/{employee_id}/educations/{education_id}

5.3 Delete Education
DELETE /api/v1/hrms/employees/{employee_id}/educations/{education_id}

6. Experience APIs

6.1 Add Experience
POST /api/v1/hrms/employees/{employee_id}/experiences

```
{
  "company_name": "ABC Software Ltd",
  "designation": "Software Engineer",
  "joining_date": "2020-01-01",
  "resignation_date": "2022-12-31",
  "responsibilities": "Backend development"
}
```

7. Bank & Tax APIs

7.1 Update Bank Information
PUT /api/v1/hrms/employees/{employee_id}/bank

```
{
  "bank_name": "BRAC Bank",
  "branch_name": "Gulshan",
  "account_name": "Rahim Uddin",
  "account_number": "123456789",
  "routing_number": "060260436"
}
```

7.2 Update Tax Information
PUT /api/v1/hrms/employees/{employee_id}/tax

```
{
  "tin": "123456789012",
  "tax_zone": "Zone-05",
```

```
  "tax_circle": "Circle-120",
  "tax_exemption": false
}
```

8. Document APIs

8.1 Upload Document
POST /api/v1/hrms/employees/{employee_id}/documents

Form Data
document_type_id
file
expiry_date
remarks

8.2 Replace Document
POST /api/v1/hrms/employees/{employee_id}/documents/{document_id}/replace

8.3 Download Document
GET /api/v1/hrms/employees/{employee_id}/documents/{document_id}/download

8.4 Delete Document
DELETE /api/v1/hrms/employees/{employee_id}/documents/{document_id}

9. Employee Timeline API
GET /api/v1/hrms/employees/{employee_id}/timeline

Query Params
event_type
date_from
date_to

10. Notes APIs

Add Note
POST /api/v1/hrms/employees/{employee_id}/notes

```
{
  "note": "Employee performed well during probation.",
  "visibility": "hr"
}
```

Get Notes
GET /api/v1/hrms/employees/{employee_id}/notes

11. Audit Log API

GET /api/v1/hrms/employees/{employee_id}/audit-logs

Response Fields

action
module
field_name
old_value
new_value
performed_by
performed_at
ip_address

12. Import / Export APIs

Import Employees

POST /api/v1/hrms/employees/import

Form Data

file

Export Employees

GET /api/v1/hrms/employees/export

Query Params

format=xlsx/csv/pdf
selected_columns[]
filters[]

13. Configuration APIs

Employee Status List

GET /api/v1/hrms/employee-config/statuses

Employment Types

GET /api/v1/hrms/employee-config/employment-types

Document Types

GET /api/v1/hrms/employee-config/document-types

Custom Fields

GET /api/v1/hrms/employee-config/custom-fields

Common API Response Format

Success

```
{
  "success": true,
  "message": "Employee created successfully.",
  "data": {}
}
```

Validation Error

```
{
  "success": false,
  "message": "Validation failed.",
  "errors": {
    "official_email": ["Official email already exists."]
  }
}
```

Permission Error

```
{
  "success": false,
  "message": "You do not have permission to perform this action."
}
```

Backend Layer Suggestion

Laravel structure:

```
Modules/
└── Employee/
    ├── App/
    │   ├── Http/
    │   │   ├── Controllers/
    │   │   ├── Requests/
    │   │   └── Resources/
    │   ├── Services/
    │   ├── Repositories/
    │   ├── Models/
    │   ├── Enums/
    │   └── Actions/
    ├── Database/
    │   ├── Migrations/
    │   └── Seeders/
    └── Routes/
        └── api.php
```

Controller = Request receive + response
Service = Business processing
Repository = All DB query, insert, update, transaction

Request = Validation
Resource = API response formatting
Enum = Status/type constants