Request for Proposal (RFP) for Back Office Operations System for Pension Services

Issued by: Trust Pension System
Date:23/11/2024

Introduction

Trust Pension System is inviting proposals from qualified software development firms to design and implement a robust Back Office Operations System for pension services. The system, built using **React** (frontend), **Node.js** (backend), and **PostgreSQL** (database), aims to enhance employer and employee onboarding, payment schedule management, and payment allocation processes.

Objectives

1. Automate and streamline employer and employee onboarding processes.
2. Provide efficient tools for uploading and validating payment schedules.
3. Automate payment allocation using system-generated validation codes.
4. Ensure data integrity and compliance with applicable regulations.

Scope of Work

1. Employer and Employee Onboarding

- **Employer Onboarding**:
    - Allow manual onboarding via forms with fields for employer details (e.g., name, contact, tax ID).
    - Enable bulk employer onboarding via Excel file upload.
- **Employee Onboarding**:
    - Allow manual onboarding via forms with fields for personal details (e.g., name, Ghana Card Number, SSNIT number).
    - Enable bulk employee onboarding via Excel file upload.
- **Employee-Employer Linking**:
    - Allow manual linking of employees to employers.
    - Enable bulk linking of multiple employees to a specific employer.

2. Payment Management

- **Payment Upload**:
    - Allow bulk uploads of payments via Excel, capturing transaction date, value date, and payment reference.

- **Payment Allocation**:
    - **Automatic Assignment**:
        - Automatically assign payments to an employer if:
            - The validation code in the payment narration matches the employer's validation code.
            - The payment amount matches the validated payment schedule amount.
            - Allocate payments to employees based on the payment schedule.
    - **Manual Assignment**:
        - Allow back office personnel to manually assign payments to employers if automatic validation fails.

3. Payment Schedule Management

- **Schedule Upload and Validation**:
    - Enable back office personnel to upload payment schedules via Excel.
    - Validate uploaded schedules by checking:
        - Ghana Card Numbers.
        - SSNIT Numbers.
        - Employer-employee linkage.
        - Membership_ID
    - Generate system-generated validation codes after successful validation.
- **Pending Payment Tracking**:
    - Mark validated payment schedules as pending on the employer account until payments are uploaded and allocated.

4. Reporting and Auditing

- Generate reports for:
    - Onboarding statistics for employers and employees.
    - Payment status and allocation details.
    - Validation success and failure summaries.
- Maintain audit logs for all system actions (e.g., onboarding, uploads, validations, assignments).

5. User Management and Permissions

- Role-based access control for back office personnel, ensuring actions are restricted based on roles.

6. Notifications and Alerts

- Send email or system notifications for:
    - Successful uploads, validations, and assignments.
    - Errors during validation or allocation processes.

7. Data Security

- Ensure encryption for data transmission and storage.
- Comply with Ghana Data Protection Act and global best practices.

Technical Requirements

1. Technology Stack

- **Frontend**: React
    - Responsive UI for desktop and mobile.
    - Form validation and dynamic user interaction with React hooks or Redux.
    - Integration with backend APIs.
- **Backend**: Node.js
    - RESTful APIs for frontend communication.
    - File handling for bulk uploads.
    - Business logic for validation, payment allocation, and reporting.
- **Database**: PostgreSQL
    - Schema design to support employer, employee, payment, and validation workflows.
    - Optimized queries for handling bulk uploads and processing.

2. System Architecture

- Use microservices architecture for modularity and scalability.
- Ensure robust error handling and logging mechanisms.
- API-first design for future integrations with external systems (e.g., payroll, banking platforms).

3. Performance Requirements

- The system should handle:
    - Bulk uploads of up to 10,000 records without significant delays.
    - Concurrent use by up to 200 users.

4. Security

- Use SSL/TLS for secure data transmission.
- Implement authentication and authorization using JWT or OAuth2.
- Regular vulnerability assessments to ensure system integrity.

Proposal Requirements

Proposals must include the following:

1. **Company Information**
    - Company name, background, and relevant experience with similar projects.
2. **Technical Proposal**
    - Detailed architecture and technology stack description.
    - Approach to meeting the outlined scope of work.
3. **Project Plan**
    - Detailed project timeline, including milestones and deliverables.
    - Resource allocation plan.
4. **Cost Breakdown**
    - Detailed budget, including development, deployment, and maintenance costs.
5. **Team Composition**
    - Resumes and roles of key personnel assigned to the project.
6. **References**
    - Case studies or testimonials from previous clients.

Evaluation Criteria

Proposals will be evaluated based on:

1. Alignment with technical requirements and objectives.
2. Experience with React, Node.js, and PostgreSQL projects.
3. Scalability and security measures proposed.
4. Cost-effectiveness.
5. Quality of support and training plans.

Submission Details

- **Deadline for Submission:** [Insert Deadline]

- **Submission Method:** [Insert Method - e.g., email or online portal]
- **Contact Information:** [Insert Contact Person and Details]

Questions and Clarifications

Direct all questions to [Contact Person] by [Insert Deadline]. Responses will be shared by23/11/2024.