Java Based Smart Procurement and Vendor Management System

What are the Real World Problems ?

- **Manual Inefficiency** : Paper-based requisitions and email approvals cause significant delays.
- **Opacity** : Lack of real-time tracking for employees and management.
- **Vendor Fragmentation** : No centralized source of truth for vendor data, leading to inconsistencies.
- **Financial Risk** : Difficulty in monitoring spend against budget in real-time.
- **Audit Gaps** : Manual record-keeping makes historical audits time-consuming and prone to error.

Solution !!

- **Centralized Platform** : A unified digital ecosystem for all procurement activities.
- **Automated Workflows** : Standardized digital processes from Requisition to Purchase Order.
- **Role-Based Transparency** : Dashboards and status indicators for every stakeholder.
- **Data-Driven Accountability** : Automated report generation and financial tracking.
- **Scalable Architecture** : Robust backend designed for enterprise growth.

TECHNOLOGY STACK

- **Backend Framework** : Spring Boot 3.2.5 (Java 17/21)
- **Database** : MySQL 8.0 (Relational Data Storage)
- **Security** : Spring Security & JWT (JSON Web Token)
- **Documentation** : Springdoc OpenAPI / Swagger UI
- **Reporting Engine** : JasperReports & Apache POI
- **Build Tool** : Maven

PROJECT ARCHITECTURE

- **Layered Pattern** : Follows the standard Controller -> Service -> Repository pattern.
- **Clean Separation** : Decoupled business logic from data access and API endpoints.
- **Stateless REST** : Designed for horizontal scalability.
- **Components** :

- **Controllers** : Handle HTTP requests and responses.
- **Services** : Contain core business and workflow logic.
- **Repositories** : Interface with MySQL via Spring Data JPA.

DATABASE DESIGN

- **Relationship Model** : Normalized schema for consistency.

- **Key Entities** :
- Users & Roles : Authentication and access control.
- Vendors : Master data for suppliers.
- Requisitions : Internal purchase requests.
- Purchase Orders : Contractual orders sent to vendors.
- Approvals : Audit logs for every decision.

- **Integrity** : Foreign key constraints ensure data reliability between POs and Vendors.

AUTHENTICATION FLOW

- **Login Request** : User sends credentials (Username/Password).
- **Verification** : Backend validates using `CustomUserDetailsService` and BCrypt.
- **Token Generation** : `JwtTokenProvider` creates a signed JWT.
- **Token Storage** : Client stores token (e.g., in LocalStorage/Session).
- **Authorization** : JwtAuthenticationFilter` intercepts subsequent requests to validate the token in the "Authorization: Bearer" header.

COMPLETE FLOW

- **Initiation** : Employee creates a Requisition (REQ).
- **Review** : Requisition is submitted for Manager approval.
- **Conversion** : Approved REQ is converted into a Purchase Order (PO).
- **Vendor Selection** : Procurement links active vendors to specific PO items.
- **Finalization** : PO is sent to the Vendor; status updated to 'CLOSED' upon fulfillment.

REQUISITION LIFECYCLE

1. CREATED : Draft stage for inputting items.
2. SUBMITTED : Sent for managerial review.
3. APPROVED : Financial validation successful.
4. REJECTED : Sent back for revision (with reason log).
5. CLOSED : Fully processed and linked to a finalized PO.

API STRUCTURE

- **Auth API** : `/auth/login`, `/auth/register`
- **Vendor API** : `/vendor/create`, `/vendor/all`, `/vendor/{id}`
- **Requisition API** : `/procurement/requisition/create`, `/procurement/requisition/all`
- **Purchase Order API** : `/procurement/purchase-order/create`, `/procurement/purchase-order/all`
- **Report API** : `/reports/vendor?format=pdf`, `/reports/vendor?format=excel`

SECURITY IMPLEMENTATION

- **JWT Protection** : Stateless security for all procurement endpoints.
- **BCrypt Hashing** : Strong password encryption at rest.
- **CORS Configuration** : Controlled access for frontend integrations.
- **Role Guards** : Annotations like `@PreAuthorize` to restrict manager endpoints.
- **Validation** : Strict input validation using Jakarta Validation API.

Error Handling

- **Global Controller**

  GlobalExceptionHandler ensures consistent API responses

- **Custom Exceptions**

  BusinessException : For domain-specific logic failures

  ResourceNotFoundException : For missing entities.

  UnauthorizedException : For failed security checks.

- **Standard Body**

  Returns timestamp, status, error, message, and path.

KEY FEATURES

- **Dynamic Reporting** : Instant PDF/Excel generation using Jasper.
- **Audit Trails** : Detailed logs of which manager approved which order.
- **Reliable Calculations** : Automated tax and total amount logic.
- **Responsive API** : High-performance JSON-based endpoints.
- **Role Switching** : Support for Admin, Manager, and Employee personas.

LIMITATIONS

- **External Integration** : Currently no direct payment gateway integration.
- **Mobile Native** : No native iOS/Android application yet (API only).
- **Vendor Portal** : Vendors cannot yet log in to acknowledge orders directly.
- **Scalability Limit** : Single database instance without sharding.

FUTURE ENHANCEMENT

- **AI ANALYTICS** : Predict spending trends and suggest cost saving vendors.
- **SUPPLIER PORTAL** : Enable vendors to self-onboard and update order statuses
- **Mobile App** : Dedicated mobile app for management-on-the-go.
- **Blockchain** : Decentralized immutable logs for critical procurement stages.

DEMO OUTLINE

- **Stage 1** : Login as 'admin' to view the vendor registry.
- **Stage 2** : Create a Vendor and a Requisition.
- **Stage 3** : Show the Approval process (Status update).
- **Stage 4** : Transition to Purchase Order.
- **Stage 5** : Generate a PDF Vendor Report as the finale.

THANK YOU !!