Requirements Specification Document

Project: InvoiceBox Inc - Web-Based Invoicing Platform
Version: 1.1
Date: October 31, 2025

1. Introduction

1.1 Project Overview

InvoiceBox Inc is a web-based platform designed to streamline the invoicing and payment process between service providers and their purchasers. The system provides a clear, role-based interface for creating, managing, and paying invoices, along with analytical dashboards for financial insights.

1.2 System Overview

The system will support two distinct user types: Providers (who create and send invoices) and Purchasers (who review and pay invoices). It will feature a JWT-based authentication system, full invoice lifecycle management, and comprehensive dashboards with analytics and reporting.

2. Functional Requirements

FR1: User Authentication and Authorization

FR1.1: The system shall allow users to register an account by providing email, password, and selecting a role (Provider or Purchaser).

FR1.2: The system shall allow registered users to log in using their email and password.

FR1.3: The system shall implement a JWT-based authentication system to secure API endpoints.

FR1.4: The system shall enforce role-based permissions, granting access to features based on the user's role (Provider/Purchaser).

FR2: Invoice Management

FR2.1: Providers shall be able to create new invoices with the following mandatory fields: Title, Description, Amount, Currency (USD, UGX, LYD), and a selection from a list of registered Purchasers.

FR2.2: The system shall manage an invoice lifecycle with the following statuses: Pending → Payment Submitted → Paid / Defaulted.

FR2.3: Purchasers shall be able to view invoices assigned to them.

FR2.4: Purchasers shall be able to update the status of an invoice assigned to them from Pending to Payment Submitted.

FR2.5: Providers shall have the ability to update the invoice status from Payment Submitted to Paid or Defaulted.

FR3: Dashboard and Analytics

FR3.1: The system shall provide a main dashboard for Providers displaying key summary metrics.

FR3.2: The dashboard shall display, at a minimum: Total Invoices, Total Amount, Paid Amount, Pending Count, and Payment Submitted Count.

FR3.3: The system shall provide an Analytics section for Providers with visual charts.

FR3.4: The analytics shall include:

A pie chart showing the distribution of invoices by status.

A bar chart showing total revenue grouped by currency (USD, UGX, LYD).

A line chart showing monthly revenue trends over time.

3. Non-Functional Requirements

NF1 (Performance): The web application shall have an initial page load time of under 3 seconds. API responses for dashboard and analytical data shall be returned in under 2 seconds.

NF2 (Usability): The user interface shall be intuitive and require minimal training. Role-based navigation shall be clear and prevent user confusion. Common tasks (e.g., creating an invoice) shall be achievable in three clicks or less.

NF3 (Security): All user passwords shall be hashed using a strong, industry-standard algorithm before storage. JWT tokens shall have a reasonable expiration time and be transmitted securely. API endpoints shall be protected against unauthorized access based on user roles.

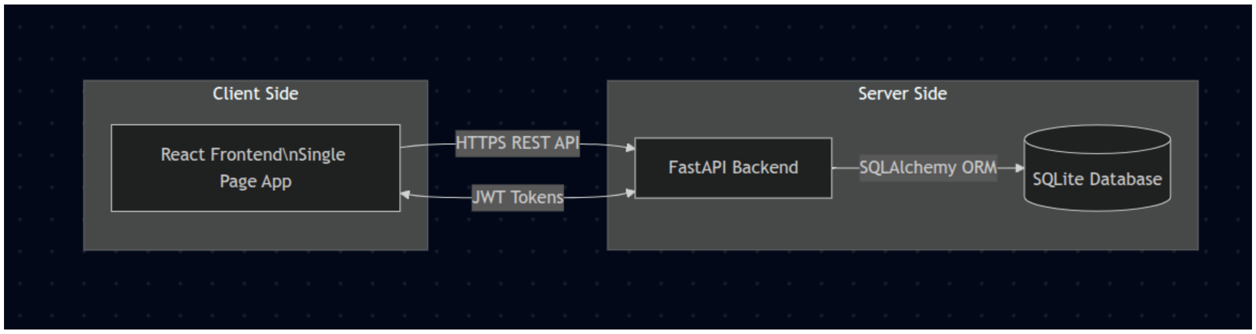
NF4 (Compatibility): The frontend application shall be fully functional and provide a consistent experience on the latest versions of Chrome, Firefox, and Safari.

NF5 (Data Integrity): The system shall ensure that all financial calculations (total amount, paid amount, etc.) are accurate and consistent across the application.

4. System Architecture and Flowcharts

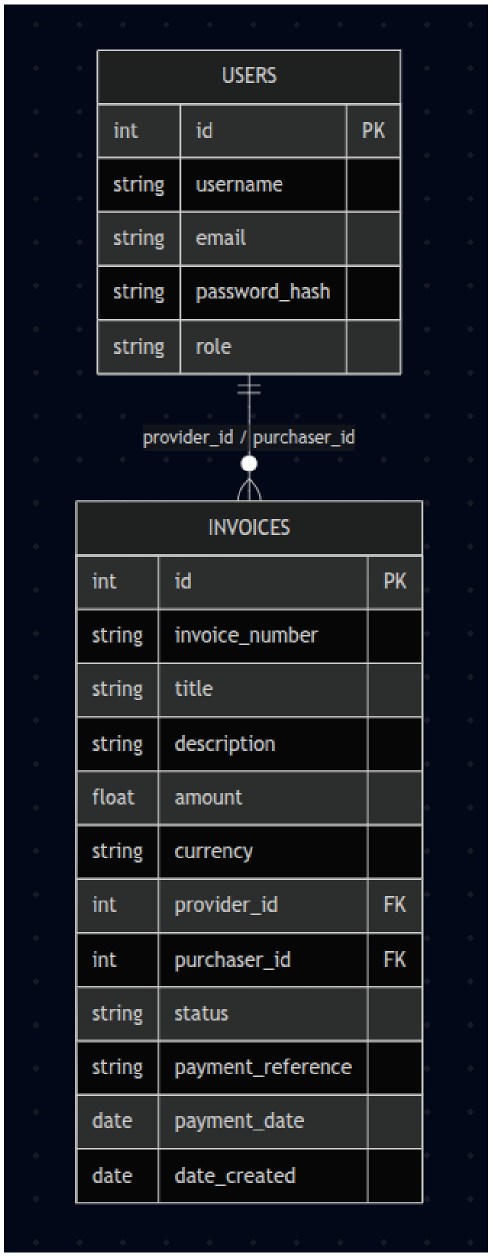
4.1 High-level System architecture

- **Frontend:** React
- **Backend:** FastAPI REST API
- **Database:** SQLite
- **Auth:** JWT (Access Token)



4.2 Database ER Diagram

- Tables: Users, Invoices
- Relationships: One Provider → Many Invoices / One Purchaser → Many Invoices

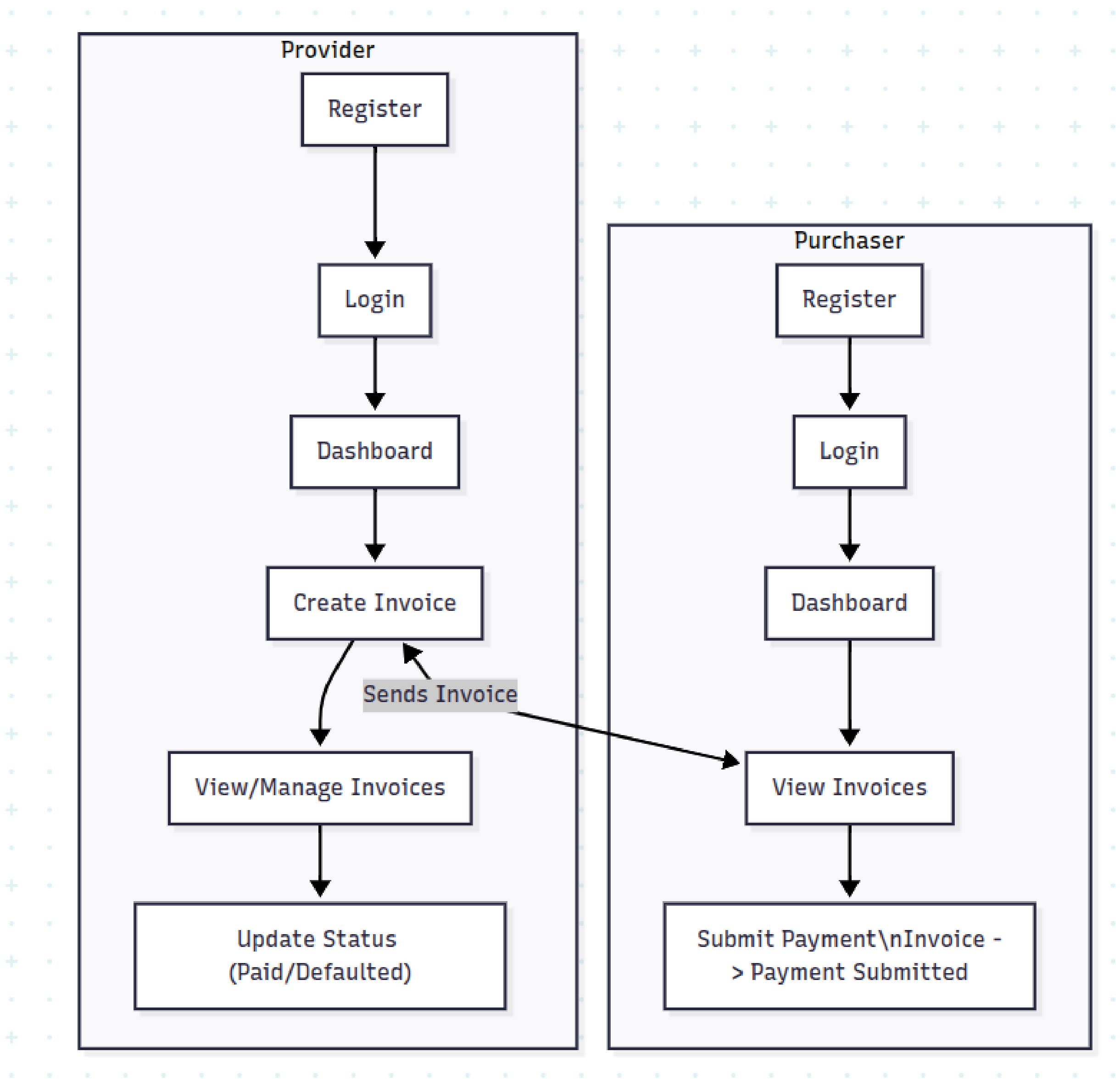


USERS		
int	id	PK
string	username	
string	email	
string	password_hash	
string	role	

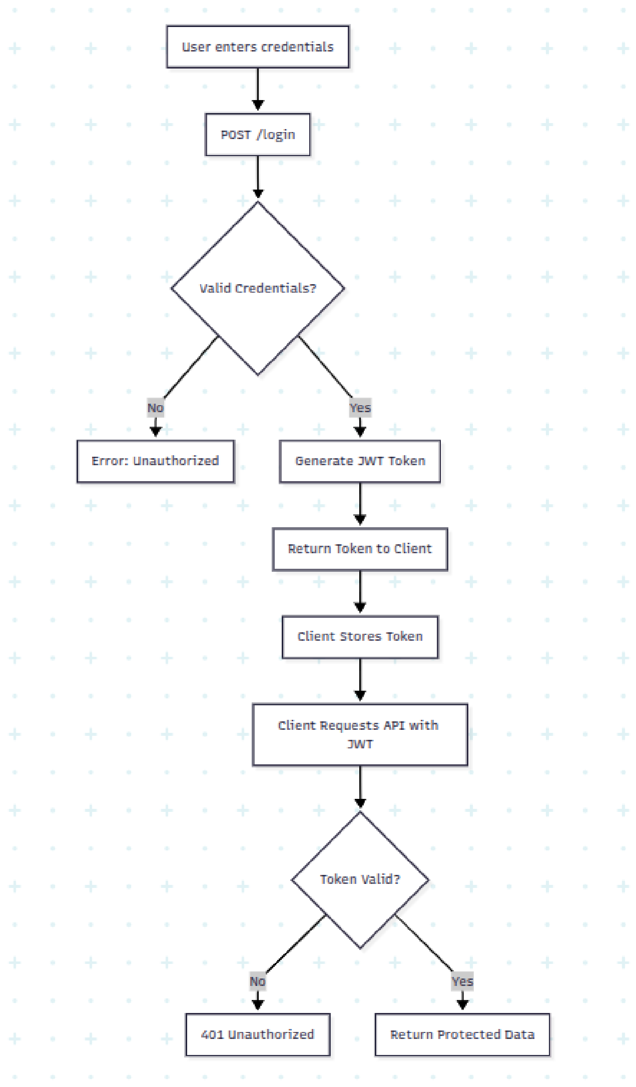
INVOICES		
int	id	PK
string	invoice_number	
string	title	
string	description	
float	amount	
string	currency	
int	provider_id	FK
int	purchaser_id	FK
string	status	
string	payment_reference	
date	payment_date	
date	date_created	

4.3 User flow Diagram

- Provider workflow: Register → Login → Dashboard → Create Invoice → Update Status
- Purchaser workflow: Register → Login → View Invoice → Submit Payment


Provider
Register
Login
Dashboard
Create Invoice
Sends Invoice
View/Manage Invoices
Update Status (Paid/Defaulted)
Purchaser
Register
Login
Dashboard
View Invoices
Submit Payment\nInvoice -> Payment Submitted

4.4 Authentication Flow diagram



5. Technical Stack

Frontend: React (with Vite), Tailwind CSS, Chart.js

Backend: FastAPI, SQLAlchemy (ORM)

Database: SQLite (for development)

Authentication: JWT Tokens