# Software Requirements Specification (SRS)

## Vendor Billing System Library

### Table of Contents

1. Introduction
2. Overall Description
3. System Features
4. External Interface Requirements
5. Non-Functional Requirements
6. Other Requirements

### 1. Introduction

## 1.1 Purpose

The Vendor Billing System Library is a comprehensive Python package designed to manage multi-vendor billing processes, providing a flexible and extensible solution for order management and payment distribution.

## 1.2 Document Conventions

* Key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119.

## 1.3 Intended Audience

* Software Developers
* System Architects
* Project Managers
* E-commerce Platform Administrators

### 2. Overall Description

## 2.1 Product Perspective

The Vendor Billing System is a standalone library that can be integrated into various e-commerce, marketplace, or multi-vendor platform applications.

## 2.2 Product Functions

* Vendor Registration
* Order Creation
* Item Management
* Payment Distribution
* Error Handling
* Flexible Configuration

## 2.3 User Classes and Characteristics

1. **System Administrators**
   * Manage vendor registrations
   * Oversee billing processes
   * Configure system parameters
2. **Developers**
   * Integrate library into existing systems
   * Extend functionality
   * Implement custom business logic
3. **End Users**
   * Indirect users who benefit from seamless multi-vendor billing

### 3. System Features

## 3.1 Vendor Management

* MUST allow registration of multiple vendors
* MUST generate unique vendor identifiers
* SHOULD support vendor metadata (name, contact, etc.)

**Requirements**:

* Vendor registration method
* Unique vendor ID generation
* Vendor lookup capabilities

## 3.2 Order Management

* MUST support creation of customer orders
* MUST allow adding items from multiple vendors to a single order
* MUST calculate total order amount
* SHOULD support custom order identifiers

**Requirements**:

* Order creation method
* Multi-vendor item addition
* Total amount calculation
* Order tracking

## 3.3 Payment Distribution

* MUST calculate payment amounts for each vendor
* MUST support proportional payment distribution
* SHOULD allow for future extension of payment logic

**Requirements**:

* Vendor payment calculation
* Proportional payment distribution
* Flexible payment processing

## 3.4 Error Handling

* MUST provide clear and specific error messages
* MUST handle scenarios like:
  + Unregistered vendor attempts
  + Empty order processing
  + Invalid item additions

**Requirements**:

* Custom exception classes
* Comprehensive error messaging
* Predictable error handling

### 4. External Interface Requirements

## 4.1 Software Interfaces

* Compatible with Python 3.8+
* No external database dependencies
* Optional integration with payment gateways

## 4.2 Hardware Interfaces

* Minimal system resource requirements
* Supports standard computing environments

## 4.3 Communication Interfaces

* RESTful API design principles
* JSON-compatible data structures

### 5. Non-Functional Requirements

## 5.1 Performance Requirements

* MUST process orders with minimal latency
* SHOULD handle orders with 100+ items efficiently
* SHOULD support concurrent order processing

## 5.2 Security Requirements

* MUST use UUID for secure identifier generation
* SHOULD support encryption for sensitive data
* MUST prevent unauthorized vendor or order modifications

## 5.3 Reliability Requirements

* MUST maintain data integrity during processing
* SHOULD provide logging mechanisms
* MUST handle potential runtime exceptions

## 5.4 Scalability

* MUST support horizontal scalability
* SHOULD allow easy extension of core functionalities
* MUST perform consistently with increasing order complexity

### 6. Other Requirements

## 6.1 Documentation

* Comprehensive README
* Inline code documentation
* Example usage scripts
* API reference

## 6.2 Compliance

* MIT License
* PEP 8 Python Style Guide
* Semantic Versioning

## 6.3 Future Enhancements

* Payment gateway integration
* Advanced reporting
* Tax and discount calculations
* Persistent storage support

### 7. Appendices

* Architecture Diagrams
* Use Case Scenarios
* Development Roadmap

### Revision History

* Version 0.1.0: Initial Draft
* Last Updated: [Current Date]