**Software Requirements Specification (SRS) Document of Order Management System (OMS)**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to specify the requirements for the multiple-company subscription management system, which will allow multiple companies to subscribe, manage their accounts, and use the software.

**1.2 Scope**

The system will support subscription management, company-specific data management, user roles, and permissions, as well as reporting functionalities tailored to each company.

**1.3 Definitions, Acronyms, and Abbreviations**

* **MCSMS:** Multi-Company Subscription Management System
* **API:** Application Programming Interface
* **OMS:** Order Management System

**1.4 References**

* "Software Requirements Specification," IEEE Standard 830-1998.
* "ISO/IEC 27001:2013," Information security management standards.

**2. Overall Description**

**2.1 Product Perspective**

The OMS is a web-based application designed to support multiple companies subscribing to the service. Each company will have isolated data management within the system.

**2.2 Product Functions**

* Manage company subscriptions
* Company-specific data isolation
* User roles and permissions
* Reporting and analytics
* API for external integrations

**2.3 User Classes and Characteristics**

* **System Administrator:** Manages overall system settings, subscriptions, and support.
* **Company Administrator:** Manages company-specific settings, users, and data.
* **Company User:** Uses the software functionalities as per their role within the company.

**2.4 Operating Environment**

* The software will be accessible via modern web browsers on desktop and mobile devices.
* The backend will run on a server supporting database and web technologies.

**2.5 Design and Implementation Constraints**

* Must support multiple companies with isolated data environments.
* Must comply with data protection regulations (e.g., GDPR, CCPA).

**2.6 Assumptions and Dependencies**

* Internet connection is required for accessing the system.
* The system will use a cloud-based database solution.

**3. Specific Requirements**

**3.1 External Interface Requirements**

**3.1.1** User Interfaces

* **Login Screen:** Username, password fields, and "Login" button.
* **Company Website:** Landing page of blurock Innovation
* **Dashboard:** Overview of subscription status, company data, and quick links to functionalities.
* **Subscription Management:** Interface for viewing and managing subscription details.
* **Checkout:** Interface for checkout the product subscription details and taxes.
* **Payment Options:** Interface for payment processing

**3.1.2** Hardware Interfaces

* **None specified:** The system is web-based and does not interface directly with hardware.

**3.1.3** Software Interfaces

* **Payment Gateway:** API for handling subscription payments.
* **Email Service:** SMTP server for sending notifications.

**3.1.4** Communication Interfaces

* **REST API:** For integration with external systems and services.

**3.2 Functional Requirements**

**3.2.1** Subscription Management

* + **FR1.1:** The system shall allow system administrators to create, update, and delete company subscriptions.
  + **FR1.2:** The system shall allow company administrators to manage their subscription details.
  + **FR1.3**: The system shall allow company administrators to generate the history of payment and invoices.
  + FR1.4: The system shall allow company administrators to print the invoice details

**3.2.2** Company Data Management

* + **FR2.1:** The system shall allow company administrators to add, edit, and delete order details data.
  + **FR2.2:** The system shall ensure data isolation between companies.
  + **FR2.3:** The system shall allow company administrators to add, edit and delete inventory items.

**3.2.3** User Roles and Permissions

* + **FR3.1:** The system shall support multiple user roles including System Administrator, Company Administrator, and Company User.
  + **FR3.2:** The system shall allow Company Administrators to manage user roles and permissions within their company.
  + **FR3.3:** The system shall restrict access to data and functionalities based on user roles.

**3.2.3** Reporting and Analytics

* + **FR4.1:** The system shall generate reports on subscription status, usage statistics, and other relevant metrics for each company.
  + **FR4.2:** The system shall allow Company Administrators to generate custom reports based on company-specific data.

**3.2.3** Notifications and Alerts

* + **FR5.1:** The system shall send email notifications for subscription renewals, upcoming expirations, and payment confirmations.
  + **FR5.2:** The system shall allow users to configure notification preferences.
  + **FR5.3:** The system shall send WhatsApp notification for subscription renewals, upcoming expirations, payment confirmation, order details and upcoming events.

**3.2.3** API Integrations

* + **FR6.1:** The system shall provide a REST API for external integrations with third-party systems.
  + **FR6.2:** The system shall support API authentication using API keys.

**3.3 Non-Functional Requirements**

3.3.1 Performance Requirements

* + **NFR1**: The system should support up to 500 concurrent users without performance degradation.
  + **NFR2**: Page load times should not exceed 2 seconds under normal conditions.

3.3.2 Security Requirements

* + **NFR3:** The system shall use encryption (SSL/TLS) for data transmission.
  + **NFR4:** The system shall ensure data isolation between companies using logical separation.
  + **NFR5:** The system shall implement multi-factor authentication for all user logins.

3.3.3 Usability Requirements

* + **NFR6:** The user interface shall be intuitive and user-friendly.
  + **NFR7:** The system shall provide context-sensitive help and documentation.

3.3.4 Reliability Requirements

* + **NFR8:** The system shall have an uptime of 99.9%.
  + **NFR9:** The system shall provide automatic data backups and recovery options.

3.3.4 Scalability Requirements

* + **NFR10:** The system shall scale to support up to 1,000 companies.
  + **NFR11:** The system shall handle up to 10,000 active users across all companies.

**4. Use Case**

**4.1** **Use Case: Company Subscription Management**

* **Use Case Name:** Company Subscription Management
* **Actors:** System Administrator, Company Administrator
* **Preconditions:** The system administrator is logged into the system.
* **Postconditions:** A new company subscription is created, updated, or deleted.

**4.2 Normal flow**

1. The system administrator navigates to the subscription management screen.
2. The administrator clicks on "Add New Subscription."
3. The system displays a form to enter company details (company name, contact information, subscription plan).
4. The administrator fills in the details and clicks "Save."
5. The system validates the input and creates a new subscription.
6. The company administrator receives a notification and login credentials.

**5. System Architecture**

**5.1 Overview**

The system architecture includes a web frontend, a backend server, and a database. The frontend will be built with HTML, CSS, JavaScript and React.js, while the backend will use a RESTful API built with Node.js or similar technology. The database will be MongoDB.

**5.2 Components**

* **Web Application:** User interface for accessing the system (developed using React.js).
* **Backend Application:** Handles business logic, user authentication, and database operations (developed using Node.js and Express).
* **Database:** Cloud-based SQL database to store all data related to the system.

**5.3 Interfaces**

* **API Endpoints**: For frontend-backend communication.
* **Database Connections**: For backend-database interactions.

**5. Detailed Design**

**5.1 Database Design**

* **Tables**: Company Schema, User schema, Subscription schema, Payment schema
* **Relationships**: One-to-many between Company User, Subscription and Payment

**5.2 User Interface Design**

* **Wireframes**: Copany page, subscription plan page, create account page, payment information page, payment confirmation page, login page and company dashboard.
* **Navigation Flow**: Visit MCSMS Website → select subscription plan→ create account page --> payment information page → sign in page --> company dashboard.

**5.3 API Design**

**Endpoints**:

* POST/GET /api/users: User authentication.
* POST/GET/DELETE/PUT /api/companies: Company registration
* POST/GET/DELETE/PUT /api/subscriptions: Company subscription details
* POST/GET /api/payments: Company payment details
* POST/GET/DELETE/PUT /api/com/address: Company address details
* POST/GET/DELETE/PUT /api/com/industry: Company industry details
* POST/GET/DELETE/PUT /api/com/contact: Company contact details

**6. Project Plan**

**6.1 Work Breakdown Structure (WBS)**

* + **Phase 1**: Requirements Gathering
  + **Phase 2**: System Design
  + **Phase 3**: Implementation
    - **Module 1**: Company Management
    - **Module 2**: Subscription plan payment management
    - **Module 3**: User management
  + **Phase 4**: Testing
  + **Phase 5**: Deployment
  + **Phase 6**: Maintenance

**6.2 Project Schedule**

* + **Milestone 1**: Completion of requirements gathering (2 weeks).
  + **Milestone 2**: Completion of system design (3 weeks).
  + **Milestone 3**: Completion of implementation (8 weeks).
  + **Milestone 4**: Completion of testing (3 weeks).
  + **Milestone 5**: Deployment (1 week).

**6.3 Resource Allocation**

* + **Team Members**: Project manager, system architect, frontend developer, backend developer
  + **Tools**: Project management software (e.g., Jira), version control (e.g., Git).

**6.4 Risk Management Plan**

* + **Risk 1**: Scope creep - Mitigation: Strict change control process.
  + **Risk 2**: Security breaches - Mitigation: Regular security audits and testing.

**7. Testing Plan**

**7.1 Testing Objectives**

* + Verify that all functional requirements are met.
  + Ensure the system is secure and performs well under load.

**7.2 Test Cases**

* + **TC1**: Verify company and user login functionality.
  + **TC2**: Verify subscription payment functionality.
  + **TC3**: Verify add user functionality.
  + **TC4**: Verify email notification functionality.
  + **TC3**: Verify cancel subscription functionality.
  + **TC3**: Verify WhatsApp notification functionality.

**7.3 Testing Schedule**

* + Unit Testing: Throughout implementation phase.
  + Integration Testing: After individual modules are complete.
  + System Testing: After all modules are integrated.
  + User Acceptance Testing (UAT): Before deployment.

**7.4 Tools and Environments**

* + Testing Tools: Selenium, JUnit, Postman.
  + Environments: Development, Staging, Production.

**8. Appendices**

**8.1 Glossary**

* + **API**: Application Programming Interface
  + **ERD**: Entity-Relationship Diagram
  + **UI**: User Interface
  + **UAT**: User Acceptance Testing

**8.2 Supplemental Information**

* + Detailed wireframes and mock-ups.
  + Additional user stories and use cases.

**8.3 Document History**

* + Version 1.0: Initial draft