Business Directory Project:

Requirements Document

**Subtitle:**  
Detailed Requirements for Business Directory Website

**Prepared by: Yeabsira Behailu**  
Backend developer

**Date: 20/11/2016**

Table of Contents

[1. Introduction 1](#_Toc172822263)

[1.1 Purpose 1](#_Toc172822264)

[1.2 Scope 1](#_Toc172822265)

[2. System Overview 1](#_Toc172822266)

[2.1 Description 1](#_Toc172822267)

[2.2 Key Features 2](#_Toc172822268)

[3. Functional Requirements 2](#_Toc172822269)

[3.1 User Management 2](#_Toc172822270)

[3.1.1 Registration and Login 2](#_Toc172822271)

[3.1.2 Roles 2](#_Toc172822272)

[3.1.3 Password Management 3](#_Toc172822273)

[3.2 Business Management 3](#_Toc172822274)

[3.2.1 Business Registration 3](#_Toc172822275)

[3.2.2 Business Details 3](#_Toc172822276)

[3.3 Tagging and Photos 3](#_Toc172822277)

[3.3.1 Tagging 3](#_Toc172822278)

[3.3.2 Photos 3](#_Toc172822279)

[3.4 Ratings and Favorites 3](#_Toc172822280)

[3.4.1 Ratings 3](#_Toc172822281)

[3.4.2 Favorites 4](#_Toc172822282)

[3.5 Roles and Permissions 4](#_Toc172822283)

[3.5.1 Role Management 4](#_Toc172822284)

[3.5.2 Permissions 4](#_Toc172822285)

[3.6 Security and Logging 4](#_Toc172822286)

[3.6.1 Activity Logging 4](#_Toc172822287)

[3.6.2 Password Security 4](#_Toc172822288)

[3.6.3 Two-Factor Authentication 4](#_Toc172822289)

[4. Non-Functional Requirements 5](#_Toc172822290)

[4.1 Performance 5](#_Toc172822291)

[4.2 Usability 5](#_Toc172822292)

[4.3 Reliability 5](#_Toc172822293)

[4.4 Security 5](#_Toc172822294)

[5. Database Schema 6](#_Toc172822295)

[5.1 Overview 6](#_Toc172822296)

[5.2 Tables 6](#_Toc172822297)

[6. Implementation Considerations 6](#_Toc172822298)

[6.1 Technology Stack 6](#_Toc172822299)

[6.2 Development Process 6](#_Toc172822300)

[7. Appendices 7](#_Toc172822301)

[7.1 Glossary 7](#_Toc172822302)

[7.2 References 7](#_Toc172822303)

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to define the requirements and specifications for developing a business directory system. It provides a comprehensive overview of what the system will do, the functionalities it will offer, and how it will meet user and business needs. This document ensures that all team members have a clear understanding of the project's scope, objectives, and constraints.

## 1.2 Scope

The business directory system is a web application that enables businesses to list their profiles and interact with users. It includes features for user and business management, tagging, rating, and security. This document covers all aspects of the system, from functional requirements to non-functional requirements, and defines the database schema and implementation considerations.

# 2. System Overview

## 2.1 Description

The system will be built using the MERN stack, which includes:

* **MongoDB:** A NoSQL database for flexible data storage.
* **Express.js:** A web application framework for Node.js, used for building the server-side logic.
* **React.js:** A front-end library for building interactive user interfaces.
* **Node.js:** A runtime environment for executing JavaScript code on the server.

The application will provide a platform for business owners to register and manage their businesses, while users can rate and favorite businesses. Administrators will have oversight capabilities to manage system content and user roles.

## 2.2 Key Features

* **User Management:** Users can register, log in, and manage their accounts. Different roles (business owner, regular user, admin) will have varying levels of access.
* **Business Management:** Business owners can create, update, and manage their business profiles, including adding details and photos.
* **Tagging and Photos:** Businesses can be tagged with keywords, and photos can be uploaded to enhance their profiles.
* **Ratings and Favorites:** Users can rate businesses and mark them as favorites for quick access.
* **Role-Based Access Control:** Different roles will have specific permissions, controlling what actions they can perform within the system.
* **Security Features:** The system will employ password hashing, two-factor authentication, and activity logging to ensure user data and activities are secure.

# 3. Functional Requirements

## 3.1 User Management

### 3.1.1 Registration and Login

* **Functionality:** Users must be able to sign up by providing a username, email address, and password. After registration, they should be able to log in using these credentials.
* **Validation:** The system will validate email addresses to ensure they are correctly formatted and unique. Passwords must meet strength requirements (e.g., minimum length, mix of characters).
* **Password Security:** Passwords will be hashed using a secure algorithm like bcrypt before being stored in the database, protecting them from unauthorized access.

### 3.1.2 Roles

* **Functionality:** Users will be assigned roles such as business owner, regular user, admin, viewer, or editor. Each role will have specific capabilities:
  + **Business Owner:** Manages their own business profiles.
  + **Regular User:** Rates and favorites businesses.
  + **Admin:** Oversees the entire system, manages user roles and permissions.
  + **Viewer:** Can view business profiles and ratings.
  + **Editor:** Can edit business profiles and manage content.
* **Role-Specific Access:** Access to features and data will be controlled based on the user’s role.

### 3.1.3 Password Management

* **Reset:** Users can request a password reset by receiving a link sent to their registered email address. This link will allow them to set a new password.
* **Two-Factor Authentication:** Users can enable two-factor authentication, which requires a second form of verification (e.g., a code sent to their phone) in addition to their password.

## 3.2 Business Management

### 3.2.1 Business Registration

* **Functionality:** Business owners can create a profile for their business, including essential information like name, description, category, and more.
* **Validation:** The system will check that all required fields are filled out and validate that the information provided is accurate and complete.

### 3.2.2 Business Details

* **Functionality:** Business owners can update their business details such as operating hours, address, phone number, and website.
* **Premium Status:** Admins can promote businesses to premium status, which may provide them with additional visibility or features.

## 3.3 Tagging and Photos

### 3.3.1 Tagging

* **Functionality:** Businesses can be associated with multiple tags to help categorize and improve searchability. Tags might include keywords related to the business's services or industry.
* **Tag Management:** Admins can create, edit, and delete tags as needed.

### 3.3.2 Photos

* **Functionality:** Business owners can upload photos of their business and add descriptions to provide more information and attract users.
* **Photo Management:** The system should display photos in a user-friendly format and allow business owners to manage (add, edit, delete) their photos.

## 3.4 Ratings and Favorites

### 3.4.1 Ratings

* **Functionality:** Users can rate businesses on a scale of 1 to 5 stars and leave comments about their experience.
* **Validation:** The system will ensure that ratings are within the specified range and handle comments appropriately (e.g., moderation for inappropriate content).

### 3.4.2 Favorites

* **Functionality:** Users can save businesses to their list of favorites for easy access later.
* **Management:** Users can view and manage their list of favorite businesses, including removing businesses from their favorites.

## 3.5 Roles and Permissions

### 3.5.1 Role Management

* **Functionality:** Admins can create and manage user roles, assigning specific permissions to each role to control access to system features.
* **Role-Based Access:** Different roles will have different levels of access to features such as editing business profiles, viewing ratings, or accessing audit logs.

### 3.5.2 Permissions

* **Functionality:** Permissions define what actions users can perform. For example, an admin might have permissions to manage all businesses, while a viewer only has permissions to view business details.
* **Assignment:** Permissions are assigned to roles through the RolePermissions table, allowing for flexible and scalable access control.

## 3.6 Security and Logging

### 3.6.1 Activity Logging

* **Functionality:** The system will log key user activities, such as login attempts, changes to business information, and other significant actions.
* **Retention:** Logs will be retained according to defined policies to ensure compliance and manage storage efficiently.

### 3.6.2 Password Security

* **Functionality:** Passwords will be hashed before storage using a secure hashing algorithm like bcrypt, preventing exposure of plaintext passwords.
* **Encryption:** Sensitive data such as user credentials and business information may be encrypted to enhance security.

### 3.6.3 Two-Factor Authentication

* **Functionality:** Two-factor authentication adds an additional layer of security by requiring a second form of verification beyond just the password.

# 4. Non-Functional Requirements

## 4.1 Performance

* **Scalability:** The system should be able to handle increasing numbers of users and businesses without performance degradation. This includes scaling both the server and database components.
* **Response Time:** The system should provide quick responses to user interactions, such as loading business profiles or submitting ratings.

## 4.2 Usability

* **User Interface:** The application should have an intuitive and easy-to-navigate interface. It should be designed to be user-friendly and accessible on various devices (desktops, tablets, smartphones).
* **Accessibility:** The system should comply with Web Content Accessibility Guidelines (WCAG) to ensure it is usable by people with disabilities.

## 4.3 Reliability

* **Availability:** Aim for high system availability, with minimal downtime for maintenance and updates.
* **Backup:** Regular backups of data should be implemented to prevent data loss in case of system failures or other issues.

## 4.4 Security

* **Data Protection:** Implement measures to protect sensitive data from unauthorized access and breaches. This includes using encryption, access controls, and regular security audits.
* **Access Control:** Ensure that access to system features and data is restricted based on user roles and permissions.

# 5. Database Schema

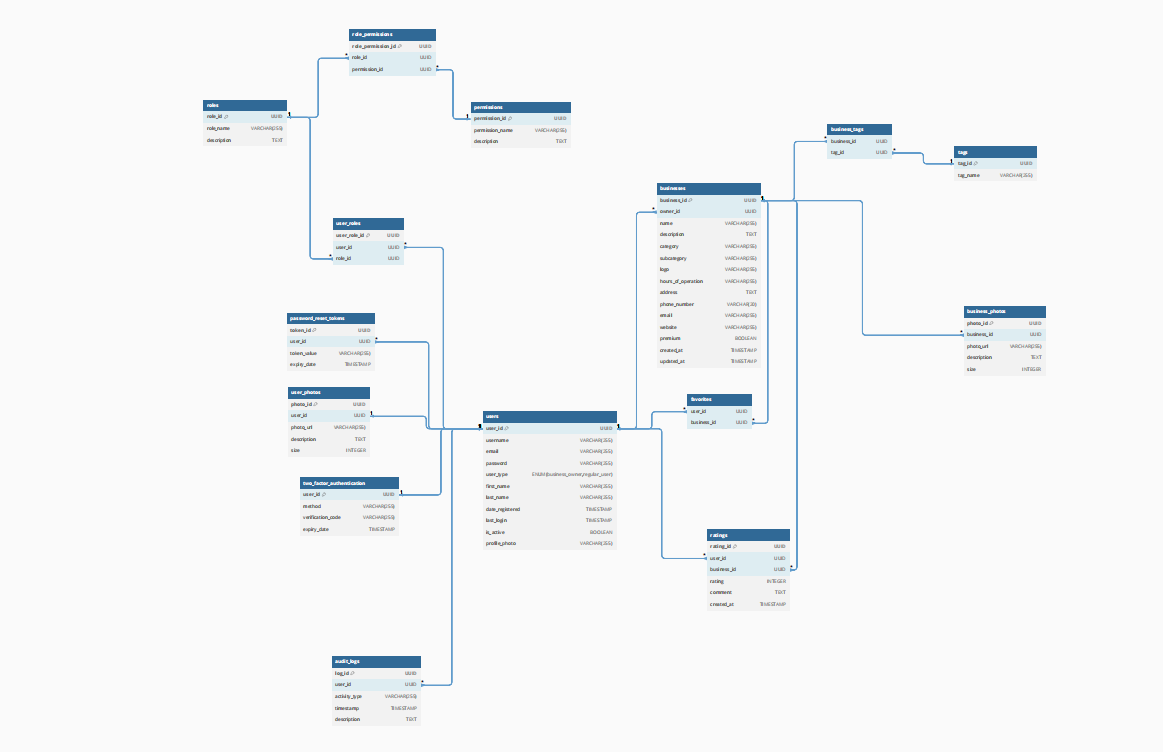
## 5.1 Overview

The database schema defines the structure of the database, including tables, columns, and relationships. It is designed to store and manage data efficiently and securely.

## 5.2 Tables

Each table in the schema has specific responsibilities:

* **Users:** Stores user information and credentials.
* **Businesses:** Contains business details and links to business owners.
* **Tags:** Manages tags for categorizing businesses.
* **BusinessTags:** Links businesses to tags for many-to-many relationships.
* **Photos:** Stores photos related to businesses.
* **Ratings:** Records user ratings and comments for businesses.
* **Favorites:** Keeps track of businesses marked as favorites by users.
* **Roles:** Defines different user roles within the system.
* **UserRoles:** Associates users with their roles.
* **Permissions:** Details the permissions available in the system.
* **RolePermissions:** Maps roles to permissions.
* **AuditLogs:** Logs user activities for auditing purposes.
* **PasswordResetTokens:** Manages tokens for password reset requests.
* **TwoFactorAuthentication:** Stores information related to two-factor authentication setups.



# 6. Implementation Considerations

## 6.1 Technology Stack

* **Backend:** Node.js and Express.js will handle server-side logic and API endpoints.
* **Frontend:** React.js will be used to build the user interface and interact with the backend.
* **Database:** MongoDB will be used for flexible and scalable data storage.

## 6.2 Development Process

* **Version Control:** Use Git to track changes, collaborate with team members, and manage different versions of the codebase.
* **Testing:** Implement various types of testing to ensure system functionality:
  + **Unit Tests:** Test individual components or functions.
  + **Integration Tests:** Ensure different components work together as expected.
  + **End-to-End Tests:** Validate the complete user experience and interactions.

# 7. Appendices

## 7.1 Glossary

* **User Types:** Definitions of roles such as business owner, regular user, admin, viewer, and editor.
* **Technical Terms:** Explanations of technical terms used in the document, such as "hashing," "two-factor authentication," and "NoSQL."

## 7.2 References

* **Documentation:** Links to official documentation for technologies and tools used in the project, such as MongoDB, Express.js, React.js, and Node.js.