**LAB PRACTICAL REPORT**

# ON

**ADVANCED WEB DESIGN FRAMEWORKS & DEVELOPMENT**

**22CS402**

### BACHELOR OF ENGINEERING

**in**

# COMPUTER SCIENCE AND ENGINEERING

****

### Submitted by: Supervised By:

**Bhavin Jain (2310990227) Mr. Venkatesh K.**

**Dev Jindal (2310990237) Associate Professor**

**Divyanshu Mehta (2310990244)**

**Garvit Rawal (2310990249)**

### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**CHITKARA UNIVERSITY**

**CHANDIGARH-PATIALA NATIONAL HIGHWAY, RAJPURA, PUNJAB, INDIA**

## CONTENTS

|  |  |
| --- | --- |
| **Title.** | **Page No.** |
| Declaration | 3 |
| Acknowledgement | 4 |
| 1.Introduction to the QR Connect Hub | 5 |
| 2. Frameworks and Development. | 7 |
| 3. Problem Statement. | 11 |
| 3.1 Abstract. | 11 |
| 3.2 Tools and Technology (Spring Boot). | 11 |
| 4. Solutions. | 13 |
| 4.1 Secure QR Code Generation and Management | 13 |
| 4.2 QR Code Scanning and Decoding | 14 |
| 4.3 Real-Time History and Activity Tracking | 14 |
| 4.4 Download and Sharing Options | 15 |
| 4.5 Responsive UI and Theme Customization | 16 |
| 5. Core Features. | 18 |
| 5.1 QR Code Generation for Various Inputs | 18 |
| 5.2 QR Code Scanning from Images | 18 |
| 5.3 History Management and Deletion | 19 |
| 5.4 Download and Sharing Features | 19 |
| 5.5 Responsive Design and Dark/Light Mode | 20 |
| 5.6 Network Accessibility and Device Support | 20 |
| 6. Appendices. | 21 |

**DECLARATION**

I hereby declare that the project work titled “QR Connect Hub - QR Code Generator Application”, submitted as part of the Bachelor’s degree in Computer Science and Engineering at Chitkara University, Punjab, is an authentic record of my own work carried out under the supervision of Mr. Venkatesh K.

**Name**

**Signature(s):**

## ACKNOWLEDGEMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behavior, and acts during the course of my study.

We express our sincere gratitude to all for providing me an opportunity to undergo Integrated Project as part of the curriculum.

We are thankful to **MR. Venkatesh K.** for his support, cooperation, and motivation provided to us during the training for constant inspiration, presence, and blessings.

Lastly, we would like to thank the almighty and our parents for their moral support and friends with whom we shared our day-to-day experience and received lots of suggestions that improve our quality of work.

### Introduction to QR Connect Hub

The QR Code Generator Application is a comprehensive software tool designed to automate and simplify the creation, scanning, and management of QR codes. In today’s digital world, where quick data sharing and access are essential for businesses, marketing, and personal use, manual QR code tools are often limited in functionality and accessibility. This platform provides a centralized digital solution where users can generate QR codes for text, images, and documents, scan them, manage history, and share them seamlessly across devices.

**Background and Need for a QR Code Generator System**

Traditionally, generating QR codes required specialized software or third-party tools, which could be cumbersome, insecure, or incompatible across devices. With the rise of mobile technology and contactless interactions, there is a pressing need for an automated web-based system that handles QR code operations efficiently and securely.

The QR Code Generator addresses these challenges by offering a software-based solution where:

* Users can generate, scan, download, and manage QR codes from any device on the same network.
* The system tracks activity history and allows deletions or regenerations.

**Objectives of the System**

The main objectives of the QR Connect Hub are:

1. Automation of QR Code Processes – Minimize manual efforts and accelerate generation and scanning.
2. User Convenience – Provide easy access to QR code tools from any device with responsive design.
3. Security and Accuracy – Safeguard data embedded in QR codes and ensure error-free operations.
4. Transparency – Offer detailed history and activity records for users.
5. Scalability – Support a growing number of users and QR code types as the platform expands.

**Importance of the System**

The significance of a QR Code Generator Application lies in its ability to integrate multiple services into a single platform:

* For Users: It simplifies data encoding, saves time, and provides instant access to QR code management.
* For Businesses: It reduces dependency on external tools, improves marketing efficiency, and minimizes costs.
* For the Digital Industry: Such platforms promote quick information sharing and enhance user experiences in various sectors like retail and events.

**Role of Technology**

The system leverages modern web technologies to deliver a robust solution. Frontend tools (like Thyme leaf, HTML5, CSS3, JavaScript) create an intuitive user interface, while backend technologies (Spring Boot, Java, MySQL) ensure smooth data handling and secure storage. Security measures protect user data, and responsive design supports multi-device access.

**Conclusion**

In conclusion, the  QR Code Generator Application is a vital step toward modernizing data sharing tools. It enhances efficiency, transparency, and user satisfaction by combining automation with security. The system ensures reliable QR code services that can scale to meet increasing demands, ultimately improving digital interactions for all users.

### 2. Frameworks and Development

The QR Code Generator Application has been developed using the Spring Boot stack, a modern, full-stack Java framework consisting of Spring Boot, MySQL, Thyme leaf, and supporting libraries. This combination provides an efficient environment for building secure, scalable, and interactive web applications, allowing developers to use Java for backend logic and integrated frontend rendering.

**Backend Frameworks**

1. **Spring Boot :**

* Spring Boot is a framework that simplifies Java application development with auto-configuration and embedded servers.
* Its modular design makes it lightweight and highly scalable - ideal for handling multiple QR code generations and user requests simultaneously.

1. **Java :** 
   * Java is the core programming language used for server-side logic.
   * It provides robust features for file handling, QR code encoding/decoding, and database interactions.

Together, Spring Boot and Java form the backbone of the server-side logic for the QR Code Generator Application.

**Frontend Framework**

1. **Thymeleaf** 
   * Thyme leaf is a Java-based template engine for rendering dynamic HTML pages.
   * It integrates seamlessly with Spring Boot, allowing for server-side rendering of UI components such as QR code forms, history lists, and scan result.
   * Thyme leaf ensures real-time updates and a responsive interface.
2. **HTML5, CSS3, JavaScript**
   * These are used along with thymeleaf to structure, style, and add interactivity to the system.
   * CSS frameworks and custom styles help create a clean, responsive design with dark/light mode support suitable for both mobile and desktop users.

**Database Framework**

1. **MySQL**
   * MySQL is a relational database that stores data in structured tables.
   * It is used for persisting QR code history, user activities, and generated codes.
   * MySQL supports scalability and efficiently handles large amounts of data, making it well-suited for a QR code management platform.
   * It integrates seamlessly with Spring Boot via Spring Data JPA for ORM (Object-Relational Mapping).
2. **Spring Data JPA**
   * Spring Data JPA is used to define entities and repositories for MySQL tables.
   * It simplifies database operations like creating new bookings, fetching hotel details, or updating user profiles saving QR codes, fetching history, or deleting entries.
   * It also ensures **data validation and consistency**, which are critical for reliable booking operations.

**Additional Supporting Tools**

1. **Zxing (Zebra Crossing) :**
   * A library for QR code generation and scanning in Java.
   * Ensures accurate encoding for text, images, and documents.
2. **Maven :**
   * Used for dependency management and building the project.
3. **Thymeleaf Extras:**
   * For enhanced template features like conditional rendering.
4. **CSS/JS Libraries:**
   * For responsive design and theme toggling.
5. **GitHub / Git:**
   * Provides version control and enables collaborative development.

**Importance of Frameworks in the Project**

1. **Full Java Stack:** Using Java throughout with Spring Boot provides consistency and simplifies development.
2. **Scalability:** Spring Boot and MySQL handle high loads efficiently, making the system scalable for thousands of QR codes and users.
3. **Security:** Built-in Spring Security can be extended for future authentication.
4. **User Experience:** Thymeleaf and JavaScript ensure a dynamic and responsive interface for smooth operations.
5. **Flexibility:** MySQL’s structured design adapts easily to new features (e.g., adding new QR code types).

**Development Approach**

The development of the **QR Connect Hub** follows a **modular approach**:

* **Generation Module:** Handles QR code creation for various inputs.
* **Scanning Module:** Processes uploaded images to decode QR codes.
* **History Module:** Tracks activities, allows deletions, and regenerations.
* **UI Module:** Manages responsive design and theme switching.
* **Network Module:** Ensures accessibility across devices on the same network.

Each module works independently but integrates seamlessly with others, ensuring the application remains robust, modular, and scalable.

**Conclusion**

The choice of the S**pring Boot** stack makes the QR Code Generator Application modern, flexible, and efficient. With MySQL managing data, Spring Boot handling backend logic, and Thymeleaf providing a smooth user interface, the system achieves a balance of performance, scalability, and user-friendliness. This framework selection ensures that the QR Code Generator Application can not only handle current requirements but also adapt to future enhancements, making it a sustainable digital solution for data sharing**.**

### Problem Statement

Traditional QR code tools often present significant challenges for users and developers. Many rely on standalone apps or online services that are limited in scope, insecure, or not accessible across devices. Users may struggle to generate custom QR codes for images/documents, scan them reliably, or manage history without multiple tools. Key problems include lack of integration, poor mobile support, and no persistent storage for activities.

Key problems include:

1. **Inefficient QR Management**: Manual generation and scanning are time-consuming and prone to errors.
2. **Lack of Real-Time History**: Users cannot easily track or manage past QR codes.
3. **Security Concerns**: Embedding sensitive data without proper handling risks exposure.
4. **Limited Accessibility**: Tools are often device-specific or require internet for basic functions.

There is a clear need for a centralized, secure, and user-friendly digital **QR Code Generator Application** that automates core operations, provides real-time management, and ensures multi-device access.

**3.1 Abstract**

Managing QR codes through traditional methods is often inefficient and lacks the flexibility demanded by modern users. This project proposes the development of a web-based QR Code Generator Application designed to address these challenges. The system provides a centralized platform for generating, scanning, and managing QR codes, with history tracking and responsive design, thereby improving efficiency, accessibility, and user satisfaction.

**3.2 Tools and Technology**

The QR Code Generator Application is built using the Spring Boot stack, a combination of modern technologies for scalability, security, and user experience:

* **Frontend Technologies**:
  + **HTML5, CSS3, JavaScript**: For structure, styling, and interactivity.
  + **Thymeleaf**: For dynamic server-side rendering.
* **Backend Technologies**:
  + **Java**: As the core JavaScript runtime environment.
  + **Spring Boot**: A framework for building robust REST APIs and applications.
  + Zxing: For QR code generation and scanning.
  + **Spring Data JPA**: For database interactions.
* **Database**:
  + **MySQL**: A relational database for storing history and QR data.
* **Development & Version Control**:
  + **Visual Studio Code/IntelliJ**: IDE for frontend and backend development.
  + **Maven** : For build and dependency management.
  + **GitHub**: For source code management and version control.

# 4. Solutions

The QR Code Generator Application provides a comprehensive set of solutions designed to modernize QR code handling. Leveraging the Spring Boot stack (Java, Spring Boot, MySQL, Thymeleaf), the system automates critical functions, enhances security, and delivers a superior user experience.

**4.1 Secure QR Code Generation and Management**

This foundational solution digitizes the process of creating and managing QR codes, replacing manual tools with an efficient workflow.

* **How It Works**: Users input text, images, or documents via a Thymeleaf form. The backend uses ZXing to generate the QR code, stores it in MySQL, and displays it instantly.
* **Benefits**: Users can generate QR codes anytime, with persistent storage for management.
* **Implementation Details**: Spring Boot controllers handle requests, with JPA for database operations.

**Example**: A user generates a QR code for a document and saves it for later use.

* 1. **Booking Management (Create, Update, Cancel)**

This solution allows users to upload images and decode embedded data.

* **How It Works**: Users upload an image; the backend processes it with ZXing to extract data, displaying results in real-time.
* **Benefits**: Provides quick decoding without external apps.
* **Implementation Details**: File upload handled by Spring Boot, with ZXing for scanning.
* **Example**: A user scans a QR code image to retrieve contact details.
  1. **Real-Time History and Activity Tracking**

This feature tracks all generated QR codes and activities.

* **How It Works**: Each generation or scan is logged in MySQL. Users view a list with filters.
* **Benefits**: Users can review and manage past activities easily.
* **Implementation Details**: JPA repositories query the history table.
* **Example**: A user downloads a QR code for printing on marketing materials.
  1. **Download and Sharing Options**

This solution enables downloading QR codes as images for sharing.

* **How It Works**: Generated QR codes can be downloaded via a button, with options to share links.
* **Benefits**:  Facilitates offline use and easy distribution.
* **Implementation Details**: Spring Boot serves files for download.

**Example**: A user downloads a QR code for printing on marketing materials.

* 1. **Responsive UI and Theme Customization**

This ensures the app works on all devices with customizable themes.

* **How It Works**:  CSS and JavaScript handle responsiveness and dark/light mode toggling.
* **Benefits**:  Improves accessibility across networks and devices.
* **Implementation Details**: Thymeleaf renders dynamic UI, with local IP binding for network access.
* **Example**: A user accesses the app from a mobile device and switches to dark mode.

# Core Features

# 5.1 QR Code Generation for Various Inputs

# Purpose: Allows creation of QR codes from text, images, or documents.

# Key Components:

# Input Form: Thymeleaf-based form for user inputs with validation.

# Encoding : ZXing library handles data embedding securely.

# Instant Preview: Displays generated QR code immediately.

# Significance: Makes QR creation versatile and user-friendly.

# 5.2 QR Code Scanning from Images

# Purpose: Decodes QR codes from uploaded images.

# Key Components:

# Upload Interface: Secure file upload for images.

# Decoding Process: Backend uses ZXing to extract data.

# Result Display: Shows decoded information with error handling.

# Significance: Provides a complete QR lifecycle in one app.

# 5.3 History Management and Deletion :

# Purpose: Tracks and manages past QR activities..

# Key Components:

# History Dashboard: Lists all entries from MySQL.

# Delete/Regenerate: Options to remove or recreate QR codes.

# Filtering: Search by date or type.

# Significance: Ensures users can organize and revisit activities.

# 5.4 Download and Sharing Features

# Purpose: Enables exporting and distributing QR codes.

# Key Components:

# Download Button: Generates image files for saving.

# Sharing Links: Network-accessible URLs for QR codes.

# Format Options: Supports PNG or other image formats.

# Significance: Extends usability beyond the app.

# 5.5 Responsive Design and Dark/Light Mode

# Purpose: Ensures optimal viewing on all devices.

# Key Components:

# CSS Media Queries: For mobile/desktop adaptation.

# Theme Toggle: JavaScript for switching modes.

# UI Elements : Consistent across themes.

# Significance: Enhances accessibility and user comfort.

# Network Accessibility and Device Support

# Purpose:  Allows access from any device on the same network.

# Key Components:

# Local IP Binding: Spring Boot listens on all interfaces.

# Cross-Device Testing: Ensures compatibility.

# Firewall Guidance: Setup for inbound connections.

# Significance: Makes the app collaborative and portable

# Appendices