Secure National ID Management System

Project Report

Department of Computer Science & Engineering
Pundra University of Science and Technology

Course Title: Software Development Project I
Course Code: CSE-3100

Prepared For:

Mst. Sahela Rahman
Lecturer, Dept. of CSE

Prepared By:

Md.Meahadi Hasan (ID: 0322310105101034)
Md.Sohanur Rahman(ID: 0322310105101039)
Arafat Rahman (ID: 0322310105101052)

“Secure National ID Management System”

A Project Report

By

Arafat Rahman, Md.Meahadi Hasan ,Md.Sohanur Rahman

Department of Computer Science and Engineering

Pundra University of Science & Technology

Rangpur Road, Gokul, Bogura, Bangladesh.

May, 2025

DEDICATION

We would like to dedicate this project to our loving parents and our course teacher, who have always believed in us and supported us through every challenge. Their constant encouragement, guidance, and inspiration gave us the strength to complete this project successfully.

Acknowledgement

First and foremost, all praises and thanks to Almighty Allah for His endless blessings, guidance, and mercy throughout the journey of this project, which enabled us to successfully complete our work.

We would like to express our heartfelt gratitude to our course teacher, Mst. Sahela Rahman, Lecturer, Department of CSE, Pundra University of Science & Technology, Bogura, for giving us the opportunity to undertake this project and for her invaluable support and guidance. Her dedication, sincerity, and inspiring vision have been a constant source of motivation for us.

It has been a great privilege and honor to work and learn under her supervision. Her thoughtful suggestions, scholarly guidance, continuous encouragement, and patient mentorship have helped us grow both personally and academically. We are deeply thankful for the time she dedicated to reviewing our work, offering insightful feedback, and guiding us through every stage of the project.

Her empathy, patience, and unwavering support—even while reading and correcting multiple drafts—have played a crucial role in the successful completion of this work. We are truly grateful for all she has done for us.

Finally, we would like to extend our heartfelt thanks to our beloved parents for their unconditional love, prayers, and unwavering support throughout our academic journey. Their constant encouragement has been a pillar of strength for us.

TheAuthors
May 2025

Abstract

The National ID Management System is a secure, database-driven application designed to manage citizen records efficiently [1]. Built using C, SQLite, and OpenSSL, it offers features like registration, data retrieval, updates, deletion, and audit logging [5]. The system incorporates role-based access control for administrators, ensuring secure operations. It aims to streamline national ID management by enhancing data integrity and traceability. This report details the system's design, implementation, and testing processes. It also explores potential future enhancements to improve functionality [2]. The project highlights the system's feasibility and operational benefits for government use.

Table of Contents

Page

Chapter 1
Introduction

1.1 Introduction

The National ID Management System is a software solution that automates citizen record management, addressing inefficiencies in manual systems [1]. It supports essential government services like voting, taxation, and social welfare. Built using C, SQLite, and OpenSSL, the system ensures robust security and scalability [5, 6, 10]. Traditional paper-based or semi-digitized systems often face errors and delays. This project aims to enhance data accuracy and traceability for efficient administration. It provides a framework for secure and reliable ID management [2]. The report outlines the system's development and potential impact.

1.2 Motivation

National identification systems are crucial for accurate citizen data in government operations . Manual systems frequently cause errors, delays, and security breaches, hindering efficiency. The National ID Management System seeks to overcome these challenges with automation. It uses modern database and cryptographic technologies to ensure reliability . The project was motivated by the need to improve data accuracy and accessibility. It aims to streamline administrative processes for governments. This solution addresses critical gaps in existing systems.

1.3 Objectives

The National ID Management System aims to deliver secure citizen record management. It seeks to implement functionalities like registration, search, update, and deletion. Audit logging ensures traceability of all actions. The system uses SQLite and OpenSSL to maintain data integrity and security [6,10]. It provides a user-friendly interface for administrators to operate efficiently. The design supports scalability for future enhancements. These objectives guide the project's development and evaluation.

1.4 Existing Work

We chose this project to address inefficiencies in manual ID systems, drawing inspiration from global efforts in national identity management. India's Aadhaar system uses a centralized database with biometric integration, achieving high identification accuracy but raising privacy concerns. Estonia's e-ID system employs secure digital signatures, enhancing accessibility, though it requires advanced infrastructure. The UAE Pass system seamlessly integrates digital identities across sectors like transportation, improving public service efficiency while addressing cybersecurity risks. Kenya's National Integrated Identity Management System (NIIMS) incorporates biometric data but faced legal challenges over privacy and inclusion. Nigeria's National Identity Management Commission (NIMC) focuses on inter-agency collaboration, deploying user-friendly enrollment interfaces, though it struggles with scalability. These works highlight the need for balancing security, scalability, and privacy in our system design.

1.5 Problems of the Existing System

Manual or semi-automated ID systems suffer from significant drawbacks [4]. Data inaccuracies arise from manual entry errors, such as incorrect names or addresses. Security vulnerabilities expose records to unauthorized access or tampering. Retrieving or updating records is slow, delaying government services. The lack of audit trails prevents tracking changes, reducing accountability. Scalability issues limit handling large populations [4]. These problems necessitate a modern, automated solution.

1.6 Project Timeline

The project timeline spans from January 2025 to May 2025, following key milestones provided for the National ID Management System development. It includes phases for topic selection, design, development, testing, and final submission, aligned with the course schedule. Each phase ends on a specified date, ensuring structured progress. The final documentation phase extends to May 9, 2025, for report completion. The table below details each phase with corresponding start and end dates. This timeline reflects our planned approach to meet project goals.

Phase	Date	Description
Orientation & Topic Selection	4-Jan-25	Selected project topic and reviewed scope .
Literature Review & Related Works	11-Jan-25	Analyzed related works in ID management .
Requirement Collection and Analysis	18-Jan-25	Collected and analyzed system requirements .
Project Planning & Timeline	1-Feb-25	Planned project phases and created timeline .
System Design – Use Case & Architecture ,Frontend UI/UX Design	8-Feb-25	Designed use case and architecture diagrams .Designed command-line interface for admins .
Database Design and ER Diagram	22-Feb-25	Developed ER diagram and database schema .
Midterm Project Progress Presentation	1-Mar-25	Presented project progress and demo .
Backend Development & API Planning	19-Apr-25	Implemented backend using C and SQLite .
Implementation Phase	26-Apr-25	Conducted group coding and integration .
Testing and Debugging	31-May-25	Performed unit and integration testing .
Final Submission & Docs	3-May-25	Submitted final code and documentation .
Final Analysis and Review	9-May-25	Reviewed project outcomes .
Final Presentation & Viva	17-Mar-25	Delivered final Project, viva, and finalized report .

1.7 Expected Outcome

The system will provide secure data storage for citizen records. It will enable efficient management through registration and updates. Audit capabilities will ensure traceability of actions. Deployable on Ubuntu with SQLite and C, it guarantees reliability [6, 7]. The solution will streamline national ID administration processes. It will support government operations effectively. Future enhancements can build on this foundation.

Chapter 2
Feasibility Study

2.1 Introduction
This feasibility study evaluates the National ID Management System's viability across technical, operational, and economic dimensions [1]. It ensures the system aligns with stakeholder requirements. The analysis identifies potential benefits, such as improved efficiency. Challenges like resource constraints are also considered. The study guides project planning and execution. It confirms the system's practicality for deployment.

2.2 Related Works
Existing systems like Aadhaar in India use centralized databases with biometric integration [3]. These systems are robust but resource-intensive. Our system employs SQLite for a lightweight, serverless solution [6]. It is designed for small to medium-scale deployments. This approach contrasts with complex, proprietary systems. It prioritizes cost-effectiveness and simplicity.

2.3 Comparative Studies
The C-based National ID Management System is lightweight compared to web-based alternatives [1]. It leverages open-source tools to minimize costs. This makes it ideal for low-resource environments. Unlike proprietary systems, it avoids expensive licensing fees. The system's design ensures efficient performance. It suits small-scale government applications.

2.4 Operational Feasibility
The system is tailored for administrators with basic computer literacy [4]. It operates on Ubuntu within a VirtualBox environment [7, 8]. This setup ensures compatibility with modest hardware. The interface is intuitive, reducing training needs. Operational feasibility is enhanced by its simplicity. It supports practical deployment in various settings.

2.5 Scope of the Problem
Manual ID systems face challenges like data duplication and unauthorized access [4]. These issues disrupt government services and data reliability. The National ID Management System automates processes to enhance efficiency. It incorporates security measures to protect records. Audit trails ensure accountability for all actions. The system addresses scalability for growing datasets [4]. This defines the problem's scope and solution.

2.6 Challenges

Developing the system requires robust data validation to prevent errors. Secure password hashing and salt generation are critical for protection [10]. Database performance must scale with increasing records. Ensuring portability across platforms is a technical hurdle. Resource constraints may limit implementation. These challenges guide the project's technical focus.

Chapter 3
Requirement Specification

3.1 C Language
The system is developed using the C programming language for its efficiency [5]. C offers low-level control for database operations. It supports cryptographic functions critical for security. The language's performance suits the system's needs. This choice ensures fast and reliable execution. It aligns with the project's technical goals [1].

3.2 SQLite
SQLite is the chosen database for its lightweight, serverless architecture [6]. It provides robust SQL support for data management. Ideal for small to medium-scale applications, it ensures efficiency. SQLite integrates seamlessly with C and Ubuntu [5, 7]. It supports secure data storage. This makes it suitable for the system.

3.3 Ubuntu
Ubuntu serves as the development and deployment platform [7]. It is a stable, open-source operating system. Ubuntu supports SQLite and OpenSSL libraries effectively [6, 10]. Its compatibility ensures reliable system performance. The platform is accessible for developers. It enhances the system's operational feasibility.

3.4 VirtualBox
VirtualBox creates a virtualized Ubuntu environment for development [8]. It ensures consistent testing across hardware configurations. This supports reliable system deployment. VirtualBox is open-source and user-friendly. It aligns with the project's resource constraints. It facilitates efficient development workflows.

3.5 Visual Studio Code
Visual Studio Code is the primary IDE for C programming [9]. It offers debugging, code completion, and Git integration. These features enhance development efficiency. The IDE supports project management and collaboration. It is widely used and reliable. This choice streamlines the coding process.

3.6 GitHub
GitHub serves as a critical tool for managing the National ID Management System's codebase throughout its development. It provides version control, allowing the team to track changes, revert to previous states, and manage multiple contributors effectively. The platform enables collaboration through features like pull requests and issue tracking, ensuring smooth integration of code from different team members. Repositories host the project's source code, documentation, and resources, making it accessible for review and backup. Its integration with Visual Studio Code enhances workflow efficiency by supporting seamless commits and pushes. This tool supports the project's scalability and maintainability as it evolves.

Chapter 4
Methodology

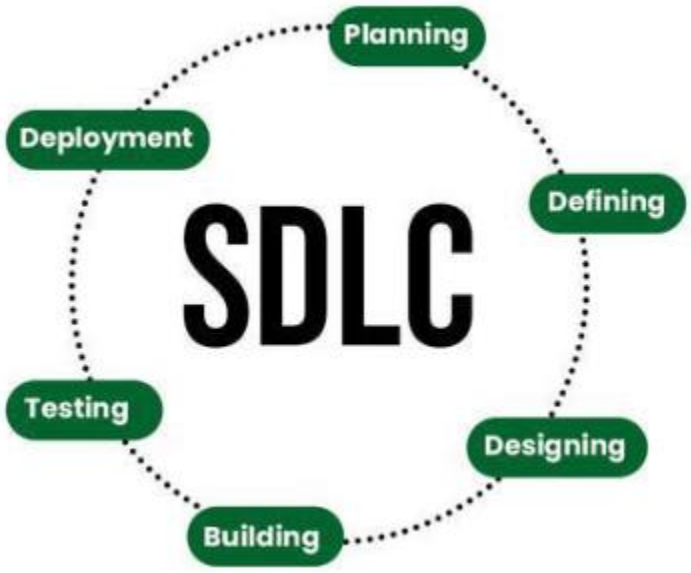
4.1 Introduction

The methodology provides a systematic approach to developing the system [1]. It ensures quality, reliability, and alignment with objectives. A structured process minimizes risks during development. The methodology guides all project phases effectively. It supports stakeholder expectations. This approach is critical for success.

4.2 Necessity of Methodology

A structured methodology clarifies project objectives and scope [2]. It ensures efficient use of resources and time. Systematic progress tracking reduces errors and delays. The approach enhances team coordination and focus. It aligns development with stakeholder needs. This necessity drives the project's execution. It ensures consistent outcomes.

4.3 Software Development Life Cycle (SDLC)



The SDLC guides the project through structured phases [1]. It includes analysis, design, development, testing, implementation, and maintenance. Each phase addresses specific project requirements. The SDLC ensures a robust and reliable system. It supports iterative improvements. This framework is essential for success. It aligns with industry standards.

4.3.1 Analysis

The analysis phase identifies user requirements and system functionalities [1]. It defines the project's scope and constraints. Stakeholder input shapes the system's features. This phase ensures alignment with government needs. It lays the foundation for design. Analysis is critical for clarity.

4.3.2 System Design

System design creates database schemas and flow diagrams [12]. It outlines the system's architecture and modules. This phase ensures scalability and maintainability. Design decisions guide development efforts. It incorporates security and efficiency [10]. The process is iterative and detailed.

4.3.3 Development

Development involves coding with C, SQLite, and OpenSSL [5, 6, 10]. It builds modules for registration and authentication. The phase ensures functional implementation. Developers follow design specifications closely. Code reviews enhance quality. This stage is central to the project.

4.3.4 Testing

Testing validates system functionalities through unit and integration tests [13]. It ensures reliability and security. Errors are identified and resolved promptly. Testing covers all modules and scenarios. This phase is crucial for deployment readiness. It aligns with quality standards.

4.3.5 Implementation

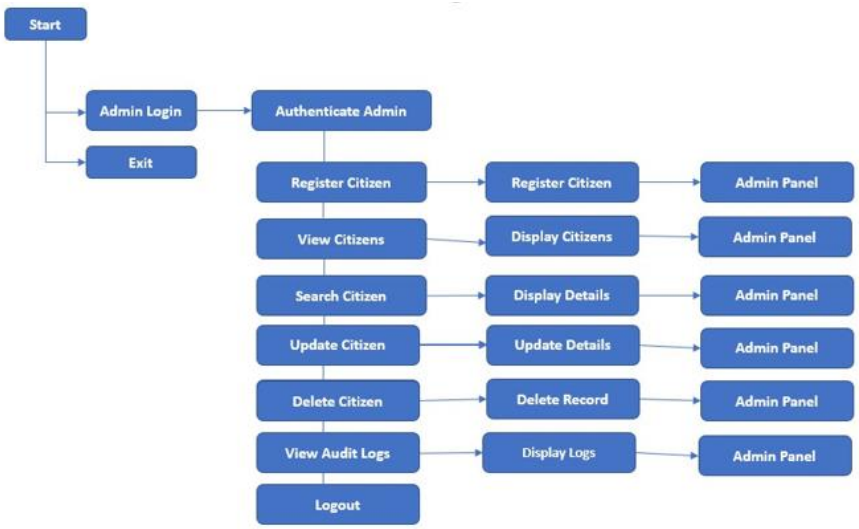
Implementation deploys the system on Ubuntu [7]. It ensures operational functionality for administrators. User training may be provided for efficiency. The phase confirms system readiness. It addresses deployment challenges. Implementation marks the project's completion.

4.3.6 Maintenance

Maintenance addresses bugs and updates features post-deployment [1]. It ensures long-term usability and security. New functionalities, like multi-user support, may be added. Maintenance adapts to evolving needs. It supports system sustainability. This phase is ongoing.

4.4 Software Process Model

4.4.1 Incremental Model



The Incremental Model structures the development process into manageable stages, such as database initialization and user authentication. Each increment undergoes iterative testing, which promotes early validation and continuous feedback. This phased approach minimizes the likelihood of critical failures and allows for ongoing improvements throughout development. Given the project's complexity, this model enhances both the system’s reliability and its scalability.

Chapter 5
System Operations

5.1 System Module

The system comprises modules for citizen management and authentication [6]. These integrate with a SQLite database for efficiency. Audit logging ensures traceability of actions. Modules are designed for modularity and scalability. They support secure and reliable operations. This structure enhances system performance [10].

5.2 Features of Our System

The system enables secure citizen registration with unique NID generation [10]. It supports record search, update, and deletion functionalities. Role-based authentication protects against unauthorized access. Audit logging tracks all activities for accountability. Data validation ensures accuracy for inputs like dates. These features address key administrative needs [6]. They ensure a robust solution.

5.3 User Panel

The admin panel provides a command-line interface for operations [6]. It allows citizen registration and record management. Administrators can view audit logs for tracking. Secure logout protects system integrity. The interface is intuitive for basic users. It supports efficient administrative tasks. This enhances usability [10].

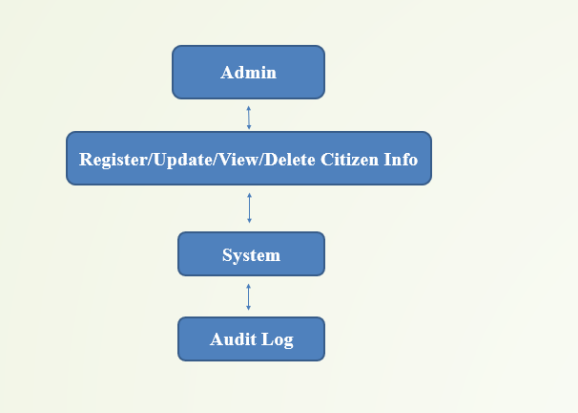
Chapter 6
System Design

6.1 Software Design Description
The system adopts a modular design for maintainability [11]. It separates database operations, authentication, and citizen management. This structure supports easy updates and scalability. Security is integrated using OpenSSL [10]. The design ensures efficient performance. It aligns with project objectives [1]. Modularity enhances long-term usability.

6.2 Traceability
Traceability maps requirements to functions like admin_register_citizen() [12]. This ensures all objectives are addressed. It supports validation during testing. Traceability enhances project transparency. It confirms alignment with stakeholder needs. This process is critical for quality assurance.

6.3 Architecture Design Process
The system uses a client-server model with SQLite as the database [11]. The C application serves as the client for operations. OpenSSL provides cryptographic security [10]. This architecture ensures efficient data handling. It supports scalability and reliability. The design balances simplicity and functionality.

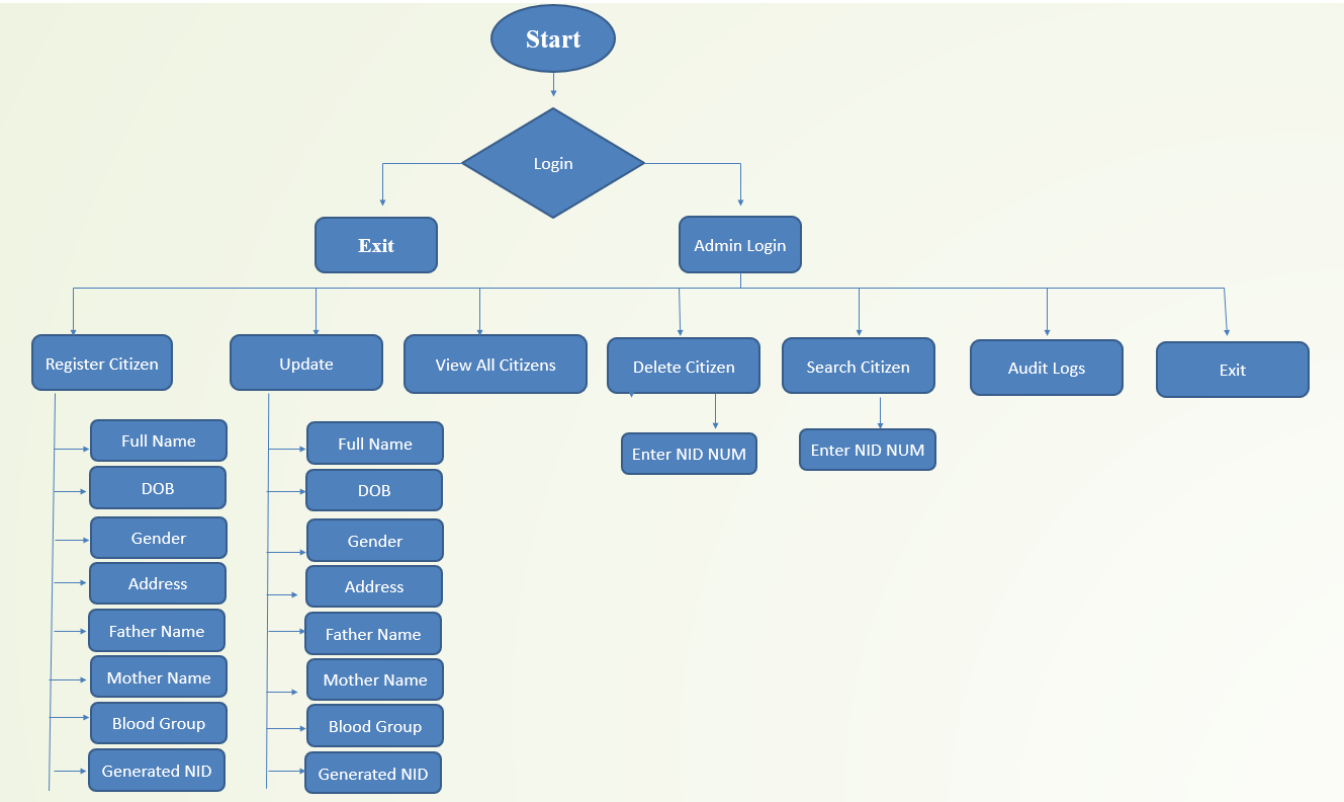
6.4 System Design
6.4.1 Data Flow Diagram of Our System



The Data Flow Diagram (DFD) illustrates key processes of the National ID Management System, including user authentication, citizen data handling, and audit logging. It clearly shows the interaction between users (e.g., Admin) and system components. The diagram highlights how data flows through modules such as registration, updating records, and maintaining audit logs. By incorporating essential data stores for citizen information and system logs, the DFD promotes modular integration and enhances system clarity. This

structured visualization aids both system development and testing, ensuring alignment with functional and security requirements.

6.4.4 Use Case Diagram of Our System



The system's use case diagram illustrates core functionalities such as citizen registration, record management, and system exit. It highlights administrative operations like updating, viewing, deleting, and searching citizen records, as well as accessing audit logs. This diagram effectively maps out the system's workflow, ensuring streamlined design and development. The only actor involved is the administrator, ensuring a secure and role-focused system environment.

Chapter 7
Implementation

7.1 Home Page

```
arafat-rahman@arafat-rahman-VirtualBox:~$ ./national_id_system

NATIONAL ID MANAGEMENT SYSTEM
1.Admin Login
2.Exit
```

The main menu provides options for admin login and system exit, implemented within the main() function of the C program. Its straightforward and user-friendly interface ensures quick access to core system functionalities. This design enhances user efficiency and aligns well with the system's operational objectives.

7.2 Categories

```
ADMIN PANEL
1. Register Citizen
2. View Citizens
3. Search Citizen
4. Update Citizen
5. Delete Citizen
6. View Audit Logs
7. Logout
Choice:
```

The admin panel categorizes functionalities into citizen management and audit logging, as shown in the menu interface. This structured layout offers clear task segmentation, enhancing ease of navigation and operational efficiency. By grouping related actions, the design improves system usability and effectively supports administrators in managing records and monitoring activities.

Chapter 8
System Testing and Maintenance

8.1 Testing

Testing ensures the system meets functional and security requirements [1]. It validates all modules, including registration and logging. Tests identify and resolve errors early. This process ensures reliability for deployment. Testing aligns with quality standards. It is critical for system success [13].

8.2 Testing Methods

8.2.1 Static Testing

Static testing involves code reviews to detect errors [13]. It identifies syntax and logic flaws before execution. This method improves code quality significantly. It is conducted during development. Static testing ensures robust implementation. It supports overall system reliability.

8.2.2 Dynamic Testing

Dynamic testing runs the program to verify inputs and outputs [13]. It tests functionalities like data retrieval. This method ensures operational correctness. It identifies runtime issues effectively. Dynamic testing is essential for validation. It complements static testing efforts.

8.2.3 Black Box Testing

Black box testing evaluates functionalities without code knowledge [13]. It tests features like citizen registration. This ensures the system meets user needs. It focuses on input-output behavior. Black box testing enhances usability. It is critical for user acceptance.

8.2.4 White Box Testing

White box testing examines internal logic, such as derive_key() [13]. It ensures code correctness and security. This method requires deep code understanding. It identifies vulnerabilities effectively. White box testing strengthens system integrity. It supports secure deployment.

8.3 System Testing

8.3.1 Unit Testing

Unit testing validates individual functions like validate_date() [1]. It ensures each component works correctly. Errors are isolated and fixed early. This method enhances module reliability. Unit testing is foundational for quality. It supports system integration.

8.3.2 Integration Testing

Integration testing verifies module interactions, such as registration and logging [1]. It ensures cohesive system performance. This method identifies interface issues. It confirms seamless data flow. Integration testing is vital for deployment. It ensures system unity.

8.4 Testing Types

The system undergoes functional, security, and performance tests [13]. Functional tests verify core features like updates. Security tests protect against unauthorized access. Performance tests ensure scalability. These tests guarantee reliability. They align with project goals.

8.5 Testing Inputs

8.5.1 Google Search Operators

Google search operators aided research on testing strategies [13]. They helped optimize SQLite performance. This informed effective test planning. Operators streamlined information gathering. They supported robust testing processes. This enhanced system quality.

8.6 Importance of Testing

Testing ensures data accuracy and system security [13]. It prevents errors in real-world applications. This process validates user requirements. Testing enhances system reliability significantly. It is essential for government deployment. It supports stakeholder trust.

8.7 Software Maintenance

Maintenance addresses bugs and enhances features post-deployment [1]. It includes adding multi-user support. Updates ensure compatibility with new technologies. Maintenance sustains system usability. It responds to user feedback. This process ensures long-term success.

8.8 Importance of Software Maintenance

Maintenance keeps the system secure and functional [1]. It adapts to evolving technological needs. Regular updates prevent performance degradation. Maintenance ensures user satisfaction. It supports scalability for future growth. This process is critical for sustainability. It aligns with project longevity.

Chapter 9
Conclusion and Future Work

9.1 Conclusion

The National ID Management System automates citizen record management effectively [11]. It achieves objectives with secure authentication and logging. The system enhances data integrity and traceability. It is reliable for government use. Deployment on Ubuntu ensures accessibility [7]. This project demonstrates practical innovation.

9.2 Limitations of the System

The system uses a command-line interface, limiting accessibility. It supports only administrators, not multiple user roles. Biometric integration is currently absent. Web-based access is not available. These limitations restrict broader adoption. They guide future improvements.

9.3 Future Plan

Future enhancements include developing a graphical user interface [10]. Multi-user support will expand access. Biometric data integration will enhance security. Cloud storage will improve scalability. These upgrades will address current limitations. They will support large-scale deployments.

References

[1] Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach*. McGraw-Hill.
[2] Sommerville, I. (2015). *Software Engineering*. Pearson.
[3] UIDAI. (2023). *Aadhaar: Unique Identification Authority of India*. https://uidai.gov.in/
[4] Pfleeger, C. P., & Pfleeger, S. L. (2007). *Security in Computing*. Prentice Hall.
[5] Kernighan, B. W., & Ritchie, D. M. (1988). *The C Programming Language*. Prentice Hall.
[6] SQLite Documentation. (2023). https://www.sqlite.org/docs.html
[7] Ubuntu Official Documentation. (2023). https://ubuntu.com/documentation
[8] VirtualBox User Manual. (2023). https://www.virtualbox.org/manual/
[9] Visual Studio Code Documentation. (2023). https://code.visualstudio.com/docs
[10] OpenSSL Documentation. (2023). https://www.openssl.org/docs/
[11] Sommerville, I. (2015). *Software Engineering*. Pearson.
[12] Yourdon, E., & Constantine, L. L. (1979). *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design*. Prentice Hall.
[13] Myers, G. J., Sandler, C., & Badgett, T. (2011). *The Art of Software Testing*. Wiley.