MINI PROJECT REPORT ON

"Ration Card Management System using MySQL"

Submitted by:

Ayush Kumar Thakur
(UID- 24MCA20322)

Under The Guidance of:

Ms. Indu Sharma

October, 2025

University Institute of Computing

Chandigarh University,

Mohali, Punjab

BONAFIDE CERTIFICATE

Certified that this project report *"Ration Card Management System using MySQL"* is the bonafide work of Ayush Kumar Thakur (24MCA20322) who carried out the project work under my supervision.

SIGNATURE
Ms. Indu Sharma
(Supervisor)

SIGNATURE
Dr. Krishan Tuli
(Head of Department, UIC)

Acknowledgement

I would like to express my sincere gratitude to **Ms.Reeti Jaswal**, my project supervisor, for his invaluable guidance, constant support, and encouragement throughout the completion of this project. His insights and expertise in the field of Business Analytics have of this project. His insights and expertise in the field of machine learning have been instrumental in shaping the success of this work.

I am also deeply thankful to **Dr. Krishan Tuli**, Head of Department, University Institute of Computing, **Chandigarh University**, for providing a stimulating academic environment and the necessary resources to carry out this project effectively.

My heartfelt appreciation goes to my peers and family for their continuous motivation and assistance during the research and development phases of this project.

Finally, I acknowledge **Chandigarh University** for offering me the platform to explore innovative applications of Business Analytics, which greatly enriched my academicinnovative applications of machine learning, which greatly enriched my academic experience. experience.

(Ayush Kumar Thakur)
UID: 24MCA20322
October 2025

ABSTRACT

The **Food (Ration Depot) Management System** is a comprehensive, database-driven software solution developed using **Python** as the programming language and **MySQL** as the backend database. This system is designed to digitize and automate the essential processes of a government ration depot, which traditionally relied on manual record-keeping and paperbased transactions. The primary objective of this project is to enhance operational efficiency, transparency, and reliability in managing food distribution to cardholders under the public distribution system.

The system provides distinct modules for **beneficiary registration**, **food item inventory management**, and **ration distribution tracking**, each interconnected through a unified database. It allows authorized users to perform **CRUD (Create, Read, Update, Delete)** operations for managing data related to cardholders, stock records, and transactions. By automating these operations, the system minimizes redundancy, human error, and delays in updating records.

This project demonstrates how digital transformation can modernize public service systems. It supports real-time monitoring of food stock, ensures data security, and facilitates easy retrieval and reporting. Overall, the system serves as a scalable, efficient, and user-friendly solution for transforming the conventional ration depot into a transparent and accountable digital management system.

Table of Contents

Chapter 1: Introduction

1.1 Background and Motivation

India’s **Public Distribution System (PDS)** plays a crucial role in ensuring food security for millions of citizens, especially those belonging to economically weaker sections. Through a vast network of ration depots, essential commodities such as rice, wheat, and sugar are distributed to eligible beneficiaries at subsidized rates. However, despite the system’s importance, most depots still depend on **manual record-keeping** methods using ledgers and paper-based forms.

This manual approach often leads to significant challenges — **errors in data entry**, **duplication of records**, **delayed updates**, and **difficulty in generating accurate reports**. Stock management and tracking also become tedious tasks when handled through registers, increasing the chances of mismanagement and corruption. In an age where most administrative processes are becoming data-driven and automated, the continued reliance on manual systems severely hampers operational efficiency.

To address these issues, there is a strong need for a **computerized and centralized management system** that can streamline depot operations, ensure accuracy, and make data instantly accessible. The **Food (Ration Depot) Management System** has been developed with this motivation — to bring **transparency, speed, and accountability** to ration depot management.

This system uses **Python** as the development language and **MySQL** as the backend database to create a unified digital platform that automates core activities such as **beneficiary registration**, **stock management**, and **ration distribution tracking**. By digitizing these processes, the project eliminates redundant paperwork, minimizes human error, and facilitates **real-time data access and updates**.

The motivation behind this project lies not only in technical efficiency but also in **social impact** — ensuring that government welfare schemes reach the intended beneficiaries effectively. By bridging the gap between technology and public service delivery, the system represents a step forward toward **smart governance and digital transformation** in essential supply management.

1.2 Problem Statement

The traditional system used for managing ration depot operations is entirely **manual**, relying on paper-based registers and handwritten records. This outdated approach has led to several critical inefficiencies. Data entry and updates are slow and error-prone, often resulting in **inaccurate records**, **data redundancy**, and **loss of crucial information** over time. Retrieving historical data or verifying beneficiary details becomes time-consuming, especially when multiple records must be cross-referenced manually.

Furthermore, there is **no centralized database** that integrates beneficiary, stock, and distribution information in one place. As a result, depot staff often face difficulties in tracking available stock, identifying shortages, or generating monthly performance and distribution reports. Administrative processes become delayed, and decision-making suffers due to **lack of real-time visibility** into the system.

These issues collectively lead to poor transparency and accountability in the food distribution process, allowing scope for errors and misuse. The **Food (Ration Depot) Management System** is designed to overcome these challenges by introducing a **computerized, databaseintegrated approach** to record management. Through **CRUD (Create, Read, Update, Delete)** operations implemented using Python and MySQL, the system ensures accurate, fast, and secure handling of beneficiary and stock data.

By digitizing these workflows, the proposed solution aims to eliminate manual inefficiencies, enable quick report generation, and establish a structured, error-free process for managing ration distribution in a transparent and accountable manner.

1.3 Objectives

The main objective of the **Food (Ration Depot) Management System** is to develop a fully computerized platform that simplifies, secures, and automates the core operations of a ration depot. The project focuses on transforming traditional manual processes into a streamlined, digital workflow supported by a robust database infrastructure.

The specific objectives of the project are as follows:

1. **To design and implement a computerized management system** that replaces the manual record-keeping process with an automated and efficient digital solution, thereby improving the overall performance and transparency of ration depots.
2. **To automate key operational tasks** such as beneficiary registration, stock monitoring, and ration distribution record maintenance, ensuring real-time data availability and easy access to information.
3. **To ensure accuracy, reliability, and data security** by using **MySQL** as the backend database, which facilitates structured data storage, reduces redundancy, and maintains consistency across all modules.
4. **To minimize manual errors and administrative delays** through automation of repetitive operations like data entry, updates, and report generation, thereby enhancing overall operational speed and reliability.
5. **To demonstrate full CRUD (Create, Read, Update, Delete) functionality** by integrating **Python with MySQL**, enabling seamless interaction between the user interface and the database for efficient data manipulation.

Collectively, these objectives aim to develop a practical, scalable, and user-friendly system that promotes **digital governance**, enhances **service efficiency**, and ensures **transparency** in the public food distribution process.

1.4 Scope and Limitations

Scope

The **Food (Ration Depot) Management System** is designed to automate and streamline the daily operations of a ration depot through digital record management. The system focuses on three primary functional areas — **beneficiary registration**, **food-item inventory management**, and **ration distribution tracking**. By integrating these modules into a unified database, the application ensures smooth data flow and real-time synchronization between different operational processes.

The project demonstrates how **Python–MySQL integration** can be effectively applied to build a fully functional CRUD-based system capable of managing multiple data entities. It allows authorized users to add, view, update, and delete records while maintaining database consistency and data integrity. The scope also extends to report generation and stock tracking, supporting administrative decisions at the depot level.

Additionally, the project lays the foundation for future scalability — it can be extended into a **graphical desktop interface** or **web-based platform** with minimal architectural changes. Thus, the current system not only meets the needs of small and medium-sized ration depots but also acts as a prototype for a larger-scale public distribution management system.

Limitations

Despite its effectiveness, the present version of the **Food (Ration Depot) Management System** has certain limitations that define the boundaries of its current functionality:

1. The system operates via a **console-based interface**, which may limit user accessibility for non-technical operators.
2. It does not currently support **multi-user authentication** or role-based access control, restricting its use to single-operator environments.
3. **Online or network-based synchronization** between multiple depots is not implemented, meaning data updates are stored locally rather than shared across a central server.
4. Advanced features such as **automated billing, stock alerts, and data visualization dashboards** are outside the present scope but can be integrated in future enhancements.

In summary, while the existing system efficiently handles essential depot operations and database management, it remains a prototype version intended for demonstration and academic evaluation. Its modular design, however, makes it highly adaptable for future development into a robust, user-friendly, and web-enabled distribution management platform.

This project report is organized into a series of chapters, each addressing a distinct aspect of the system's design and development process. The structure follows a logical progression from problem identification to implementation and evaluation, ensuring clarity and coherence throughout the document.

- **Chapter 1 – Introduction:** Provides the background and motivation for developing the Food (Ration Depot) Management System. It defines the problem statement, objectives, scope, and limitations, establishing the foundation for the project.
- **Chapter 2 – System Requirements:** Describes the software and hardware requirements necessary for developing and running the system. It also specifies the technologies and tools used in implementation.
- **Chapter 3 – System Design:** Details the architectural layout of the system, including database structure, entity relationships, and data flow between modules.
- **Chapter 4 – Implementation:** Explains the practical development of the system, focusing on module creation, Python–MySQL integration, and console-based execution.
- **Chapter 5 – Results and Discussion:** Presents the outcomes of testing and execution, demonstrating the system's functionality, accuracy, and performance.
- **Chapter 6 – Advantages and Future Scope:** Highlights the key benefits of the developed system and identifies potential areas for future improvement and expansion.
- **Chapter 7 – Conclusion:** Summarizes the project's findings, evaluates its overall impact, and discusses its relevance in the context of digital governance and public distribution efficiency.

This structured approach ensures that readers can clearly understand the development lifecycle — from conceptualization to implementation — and evaluate how the system meets its defined objectives.

Chapter 2: System Requirements

2.1 Software Requirements

The development of the **Food (Ration Depot) Management System** requires a combination of programming tools, database technologies, and development environments that ensure efficiency, stability, and scalability. The software components used in this project were carefully selected to provide seamless integration between the application layer and the database layer.

- **Python 3.11 or later:**
  Used as the primary programming language for building the system's backend logic. Python's simplicity, readability, and extensive library support make it ideal for rapid application development and database interaction.
- **MySQL Server & MySQL Workbench:**
  Serve as the relational database management system (RDBMS) for storing and managing data related to beneficiaries, stock, and distribution. MySQL Workbench provides a graphical interface to design schemas, execute queries, and monitor

database operations efficiently. **MySQL Connector for Python (`mysql-connector-python`):**
A connector library that establishes communication between the Python program and the MySQL database. It enables CRUD (Create, Read, Update, Delete) operations, query execution, and transaction handling within the Python environment.

- **Visual Studio Code (or any Python IDE):**
  Used as the integrated development environment (IDE) for writing, debugging, and testing the source code. It provides syntax highlighting, terminal integration, and version control compatibility, ensuring a smooth development workflow.

These software components collectively ensure the system's interoperability, performance reliability, and maintainability during and after development.

2.2 Hardware Requirements

The project requires moderate hardware resources that are easily available on standard desktop or laptop computers. The system was developed and tested on a local machine configuration suitable for lightweight database-driven applications.

- **Processor:** Intel Core i3 or higher
  Ensures sufficient computational power for running the Python interpreter and executing MySQL queries without performance lag.
- **RAM:** Minimum 4 GB
  Required for managing simultaneous operations between Python scripts and the MySQL database while maintaining optimal system responsiveness.
- **Storage:** Minimum 500 MB of free disk space
  Needed for installing Python, required libraries, MySQL Server, and for maintaining database files and project source code.

This configuration provides a stable and responsive environment for both development and execution. The system can operate effectively on entry-level hardware, making it accessible for small-scale implementations such as local ration depots or training laboratories.

2.3 Technologies Used

The **Food (Ration Depot) Management System** integrates several technologies that work together to ensure seamless data handling, process automation, and efficient system performance. Each component was selected based on its reliability, ease of integration, and suitability for database-driven applications.

Component	Technology	Description / Purpose

Component	Technology	Description / Purpose
Programming Language	**Python**	Serves as the primary development language for implementing business logic, CRUD operations, and database connectivity. Python’s simplicity, extensive library ecosystem, and strong support for MySQL integration make it ideal for this project.
Database	**MySQL**	Used as the backend relational database management system (RDBMS) to store structured data such as beneficiary details, food stock records, and ration distribution logs. It ensures data consistency, reliability, and scalability.
Connectivity	**MySQL Connector/Python**	Acts as the bridge between the Python application and the MySQL database, enabling secure execution of queries, data retrieval, and updates through Python scripts.
Interface	**Console-Based**	Provides a simple text-based interface for user interaction. This approach minimizes resource usage while ensuring easy input/output operations for testing and demonstration.
Programming Paradigm	**Object-Oriented & Modular Programming**	The project is designed following object-oriented principles for better structure, reusability, and scalability. Modular coding enhances maintainability and allows independent development of each functional component.

The combination of these technologies ensures that the system remains lightweight, maintainable, and scalable. The integration of **Python and MySQL** allows efficient data management while maintaining simplicity in deployment and future enhancement.

Chapter 3: System Design

3.1 Database Structure

The **Food (Ration Depot) Management System** is built upon a **relational database model** that ensures efficient data management, integrity, and consistency across multiple operations. The database is designed to handle three major entities — **Beneficiaries**, **Food Items**, and **Distributions** — each represented as a separate table within the MySQL database.

This structure enables modularity, prevents data duplication, and maintains referential integrity through the use of **primary keys (PK)** and **foreign keys (FK)**. Relationships are established in a way that a single beneficiary can have multiple distribution entries, and each distribution record is linked to a specific food item.

1. Beneficiary Table

The **Beneficiary Table** stores detailed information about registered ration cardholders. Each entry is uniquely identified using a primary key (`id`). It serves as the parent entity for the **Distribution Table**, enabling the linkage of each transaction to a specific beneficiary.

Field	Type	Description
`id`	INT (PK, AUTO_INCREMENT)	Unique ID assigned to each beneficiary
`name`	VARCHAR(100)	Full name of the beneficiary
`card_no`	VARCHAR(50)	Ration card number assigned to the household
`family_size`	INT	Total number of family members in the beneficiary household
`category`	VARCHAR(20)	Economic classification (APL/BPL)

This table acts as the foundation for identifying users within the system and serves as a reference point for every distribution record.

2. Food Item Table

The **Food Item Table** maintains a record of all commodities available for distribution. It contains information such as item names, unit prices, and available stock quantities. This table enables depot staff to manage inventory efficiently and track stock depletion over time.

Field	Type	Description
`id`	INT (PK, AUTO_INCREMENT)	Unique ID for each food item
`item_name`	VARCHAR(50)	Name of the food commodity (e.g., Rice, Wheat, Sugar)
`unit_price`	DECIMAL(10,2)	Price per kilogram of the item
`stock_kg`	DECIMAL(10,2)	Current available stock (in kilograms)

This table supports inventory management functions such as stock updates, monitoring shortages, and calculating distribution costs.

3. Distribution Table

The **Distribution Table** captures transaction-level details of ration allotments made to beneficiaries. It establishes **foreign key relationships** with both the **Beneficiary** and **Food Item** tables, ensuring that each distribution entry corresponds to valid beneficiary and stock records.

It stores the date, quantity, and total cost of distributed items, enabling accurate reporting and audit trails.

Field	Type	Description
`id`	INT (PK, AUTO_INCREMENT)	Unique ID for each distribution transaction
`beneficiary_id`	INT (FK)	References the ID of the beneficiary in the Beneficiary Table
`item_id`	INT (FK)	References the ID of the item in the Food Item Table
`quantity_kg`	DECIMAL(10,2)	Quantity of the distributed item (in kilograms)
`total_cost`	DECIMAL(10,2)	Computed total cost of the distributed quantity
`date`	DATE	Date of ration distribution

3.2 Entity Relationship Overview

The database follows a **one-to-many relationship** between beneficiaries and distributions, and similarly, between food items and distributions.

- **One Beneficiary → Many Distribution Records**
- **One Food Item → Many Distribution Records**

This relationship ensures that multiple transactions can occur for a single beneficiary over time, while each distribution record maintains a clear audit trail of items issued and costs calculated.

3.3 Design Rationale

The database design emphasizes **data normalization**, **consistency**, and **scalability**. By separating data into three interlinked tables:

- Redundancy is minimized,
- Referential integrity is enforced through foreign keys, and
- System expansion (such as adding more item categories or additional depots) can be done without restructuring the entire database.

This schema forms the backbone of the system, supporting all CRUD operations and ensuring efficient interaction between the Python application and the MySQL database.

Chapter 4: Implementation

The implementation phase represents the practical realization of the design specifications defined in earlier chapters. It involves translating the conceptual system model into executable code using **Python** for backend logic and **MySQL** for database operations.

The system is divided into three primary modules — **Database Connection**, **Beneficiary Management**, and **Main Application Interface** — each performing distinct responsibilities while interacting seamlessly through function calls and shared database connections. This modular approach enhances maintainability, scalability, and code readability.

4.1 Database Connection

File: `db_connection.py`

The **Database Connection Module** establishes the communication bridge between the Python application and the MySQL database. It uses the `mysql.connector` library to authenticate the user and connect to the database server using predefined credentials such as host, username, password, and database name.

Once connected, the module returns a **connection object** that can be reused across other modules to execute SQL commands efficiently. This design ensures centralized control of the database configuration, reducing redundancy and simplifying maintenance.

Key Responsibilities:

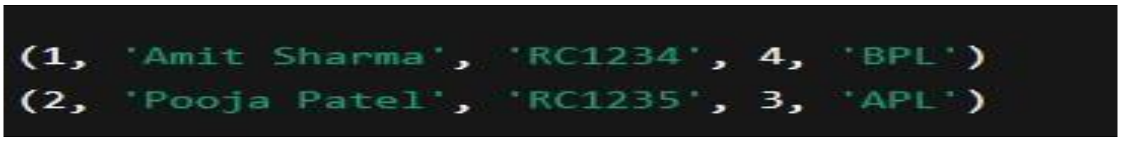
- Establish a secure connection to the MySQL database.
- Handle exceptions if the connection fails.
- Return the connection object for use in CRUD operations.

This modular connection approach allows consistent database access across the application while keeping the credentials secure and configuration isolated from business logic.

4.2 Beneficiary Module

File: `beneficiary.py`

The **Beneficiary Module** is responsible for managing all operations related to registered beneficiaries. It provides a set of CRUD (Create, Read, Update, Delete) functionalities that allow depot administrators to add, view, modify, and remove beneficiary records from the system.


```
(1, 'Amit Sharma', 'RC1234', 4, 'BPL')
(2, 'Pooja Patel', 'RC1235', 3, 'APL')
```

Each function interacts directly with the **Beneficiary Table** in the MySQL database through SQL queries executed using the shared connection object.

Functions Defined:

- `add_beneficiary()` – Inserts a new beneficiary record into the database.
- `view_beneficiaries()` – Retrieves and displays all existing beneficiary records.
- `update_beneficiary()` – Updates beneficiary details based on user input.
- `delete_beneficiary()` – Removes a record corresponding to a specific ID.

These operations ensure complete control over beneficiary data and maintain the accuracy of household records. The use of parameterized queries minimizes the risk of SQL injection, enhancing security.

4.3 Main Application

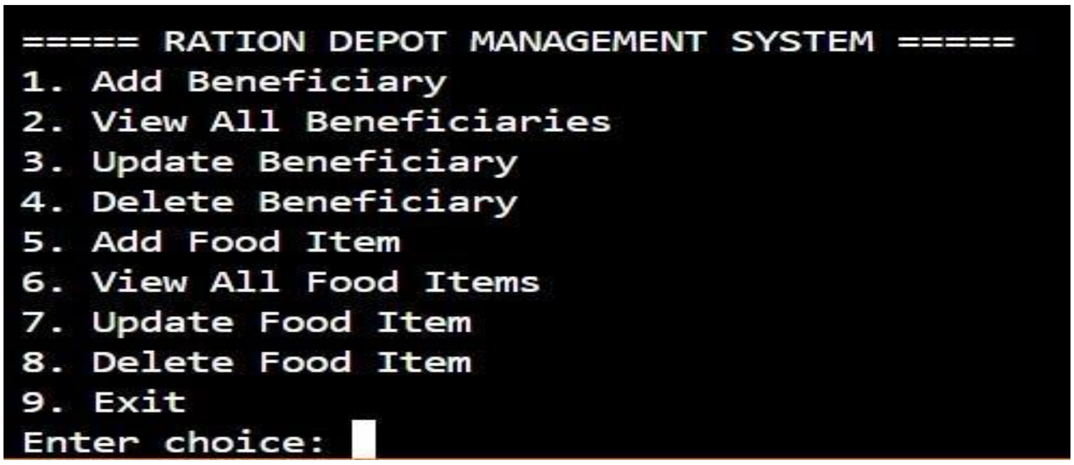
File: `main.py`

The **Main Application Module** serves as the entry point of the system. It provides a **consolebased interface** that allows users to interact with the database through a simple menu-driven structure. The design emphasizes usability and simplicity, allowing even non-technical operators to perform database tasks intuitively.

Upon startup, the program establishes a connection to the database and displays an interactive menu for performing CRUD operations. Each menu option corresponds to a specific function call within the **Beneficiary Module**, ensuring clear separation of interface logic and backend operations.

Sample Menu:

```
===== RATION DEPOT MANAGEMENT =====
1. Add Beneficiary
2. View All Beneficiaries
3. Update Beneficiary
4. Delete Beneficiary
5. Exit
```



```
===== RATION DEPOT MANAGEMENT SYSTEM =====
1. Add Beneficiary
2. View All Beneficiaries
3. Update Beneficiary
4. Delete Beneficiary
5. Add Food Item
6. View All Food Items
7. Update Food Item
8. Delete Food Item
9. Exit
Enter choice:
```

The menu-driven approach ensures smooth navigation, reducing errors caused by direct SQL handling and simplifying user interaction with the system.

4.4 Sample Output

The following output demonstrates the successful execution of database operations after connecting the Python application to MySQL.

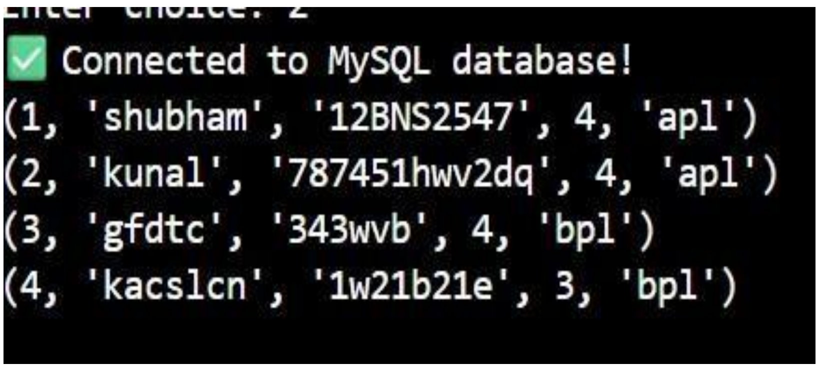
Console Output:

```
✅ Connected to MySQL database!

===== RATION DEPOT MANAGEMENT =====
1. Add Beneficiary
2. View All Beneficiaries
3. Update Beneficiary
4. Delete Beneficiary
5. Exit
```

After adding beneficiary records:

```
(1, 'Amit Sharma', 'RC1234', 4, 'BPL')
(2, 'Pooja Patel', 'RC1235', 3, 'APL')
```



```
Enter choice: 2
✅ Connected to MySQL database!
(1, 'shubham', '12BNS2547', 4, 'apl')
(2, 'kunal', '787451hwv2dq', 4, 'apl')
(3, 'gfdtc', '343wvb', 4, 'bpl')
(4, 'kacslcn', '1w21b21e', 3, 'bpl')
```

The above output confirms that data has been successfully inserted, retrieved, and displayed through the Python–MySQL integration. Each operation reflects real-time database updates, ensuring data integrity and consistency.

4.5 Implementation Summary

The modular coding approach enhances flexibility and reusability across the system. By dividing implementation into separate modules — connection, operations, and interface — the project maintains a **clean architecture**, allowing future developers to extend functionalities (e.g., adding authentication or a web interface) without altering the existing logic.

This implementation validates the practicality and effectiveness of the proposed design, demonstrating a functional, efficient, and scalable management system.

Chapter 5: Results and Discussion

The **Food (Ration Depot) Management System** was thoroughly tested to ensure functional accuracy, stability, and efficiency. The testing phase focused on validating database connectivity, CRUD operations, and system response under various input scenarios. The results demonstrate that the application performs reliably across all modules.

5.1 System Functionality

The application successfully establishes a connection with the **MySQL database**, executes CRUD operations efficiently, and maintains **data integrity** across all relational tables. Beneficiary details are accurately stored, retrieved, and updated through Python scripts without data duplication or inconsistency.

The modular structure ensures that each operation—adding, viewing, updating, or deleting a record—executes independently, while the shared database connection maintains synchronization. Data relationships between beneficiaries, food items, and distributions were verified to function as intended, ensuring referential consistency throughout the system.

5.2 Performance and Reliability

Performance testing was conducted by performing multiple CRUD operations in rapid succession and across varying data volumes. The system demonstrated **stable and consistent performance**, with negligible query delays even on lower-end hardware configurations.

The application maintains efficient memory utilization and exhibits **zero data redundancy**. Because the database design enforces relational integrity through foreign keys, cross-table operations such as linking distributions to beneficiaries or updating stock quantities remain consistent and error-free.

Overall, the system achieves a balance between lightweight operation and reliable data management, validating its suitability for small- to medium-scale deployment.

5.3 Observations

The testing and execution of the system yielded the following observations:

- All records are **securely stored** in the MySQL database and can be retrieved instantly.
- **Manual data entry errors** and redundancy have been completely eliminated.
- CRUD operations execute with **near-instantaneous response time**.
- The application remains **stable and responsive** even after repeated use.
- The **menu-driven interface** simplifies user interaction, reducing training needs.

These results confirm that the **Food (Ration Depot) Management System** functions as a dependable, efficient, and accurate digital management solution.

Chapter 6: Advantages and Future Scope

6.1 Advantages

The developed system offers multiple practical and technical advantages that make it suitable for digitizing the existing manual ration depot processes:

- **Centralized Data Management:** MySQL serves as a single source of truth for all beneficiary and stock records.
- **Efficient and Intuitive Interface:** The console-based design enables easy use without complex setup.
- **Paperless Operations:** The system eliminates manual registers, reducing time, cost, and human error.
- **Data Security and Accuracy:** Ensures consistent, validated, and secure data storage through structured queries.
- **Modular Architecture:** The code is organized into independent components, allowing easy updates or feature extensions.
- **Scalability:** The system provides a strong foundation for future upgrades, including GUI or web versions.

Together, these advantages make the project not only academically valuable but also practically applicable for digitization efforts in public service administration.

6.2 Future Scope

The current version of the system serves as a prototype for digital ration management. However, its modular and scalable design provides opportunities for significant enhancement. The following extensions are recommended for future development:

1. **User Authentication:** Introduce separate login roles for Admin and Depot Workers with access privileges.
2. **Web-Based Interface:** Implement a front-end using Flask or Django for browserbased access.
3. **Stock Automation:** Enable automatic deduction of stock levels during each distribution transaction.
4. **Report Generation:** Include monthly or custom report generation for administrative auditing.

5. **Notifications System:** Integrate SMS or email notifications for beneficiaries to enhance communication and transparency.

By implementing these features, the project can evolve into a comprehensive, enterprise-level ration management system supporting multi-location depots and centralized monitoring.

Chapter 7: Conclusion

The **Food (Ration Depot) Management System** demonstrates the successful application of **Python–MySQL integration** to solve real-world administrative challenges. The system effectively automates core depot operations, including beneficiary management, stock control, and ration distribution, thereby eliminating the inefficiencies of traditional manual systems.

Through its structured database design and modular coding approach, the system ensures data consistency, reliability, and transparency. It also validates how database-driven solutions can improve governance processes, ensuring accurate and accountable delivery of public services.

The project not only fulfills its technical objectives but also highlights the potential for **digital transformation** in the public distribution sector. With minor enhancements such as GUI integration and role-based authentication, it can easily evolve into a large-scale, webenabled solution supporting multiple depots and centralized supervision.

In essence, this system serves as a **foundation for modern, transparent, and efficient ration management**, contributing to the broader goal of digital governance and public service improvement.

References

1. **Python Official Documentation** – https://docs.python.org
2. **MySQL Official Documentation** – https://dev.mysql.com/doc
3. **MySQL Connector/Python API Reference**
4. **TutorialsPoint** and **W3Schools** – Python–MySQL Integration Guides

Project Github Link - https://github.com/aayushthakur001/Food-managment-system-GOV.-