

JEWELRY MANAGEMENT SYSTEM

TABLE OF CONTENTS

1. SUMMARY OF THE
PROJECT...
2. PROJECT DETAILS AND
WORKING ...
3. RESULTS AND ANALYSIS ...



CHAPTER: 1

SUMMARY:

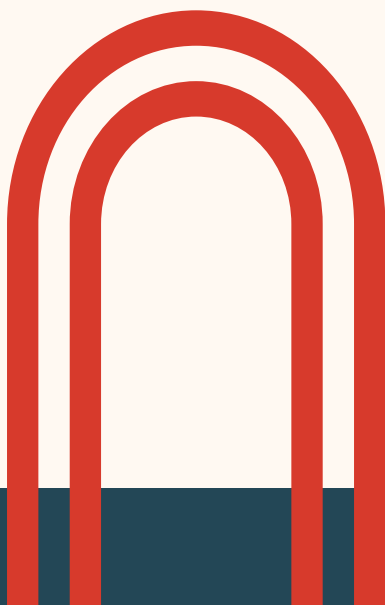
The Jewelry Inventory Management System (JMS Database) is developed to enhance the efficiency and accuracy of managing inventory and related operations in a business environment. This project focuses on streamlining inventory management, tracking supplier and worker details, and automating various related processes.

OBJECTIVES:

- ❖ **EFFICIENCY:** To automate inventory and worker management processes.
- ❖ **ACCURACY:** To reduce errors through database constraints and validations.
- ❖ **AUTOMATION:** To provide automated processing of data to reduce manual workload.

SIGNIFICANCE:

The system is designed to address common issues in inventory and workforce management, such as tracking stock levels, managing supplier interactions, recording worker details, and automating payroll and inventory operations. This project aims to create a robust system that minimizes human error and maximizes operational efficiency. The AIMS project addresses common challenges faced by retail businesses in managing their inventory, such as stockouts, overstocking, and discrepancies in inventory records. By automating these processes, businesses can focus more on customer service and sales strategies.



CHAPTER 2: PROJECT DETAILS AND WORKING

SECTION 2.1: INTRODUCTION

This section details the system components, their architecture, and how they work together to achieve the project objectives.

SECTION 2.2: SYSTEM ARCHITECTURE

The system architecture of the Automated Inventory Management System includes the following layers:

- **Presentation Layer:** Interface for users to interact with the system.
- **Logic Layer:** Core functionality that processes data and business rules.
- **Data Layer:** Database where all data is stored and managed in form of tables.

SECTION 2.3: DATABASE DESIGN

The database is designed to manage various aspects of the inventory and workforce, using multiple interconnected tables.

SECTION 2.4: TABLE DETAILS AND CONSTRAINTS

SUPPLIER TABLE

- **Description:** Stores information about suppliers, including their contact details, items supplied, and transaction details.
- **Constraints:** Ensures **quantity** and **total_money** are non-negative values.

WORK TABLE

- **Description:** Records details of the work performed, such as the type of work, materials used, and the worker responsible for the task.
- **Foreign Key:** Links **worker_id** to the **workers** table to ensure data integrity.

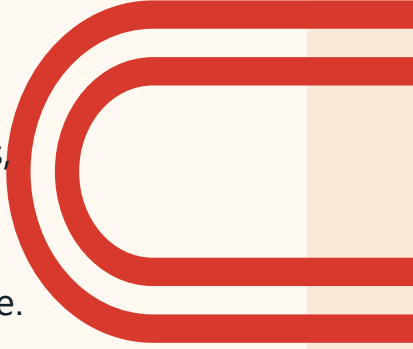
WORKERS TABLE

- **Description:** Contains personal and job-related details of workers, including their contact information, job responsibilities, and salary.
- **Constraints:** Ensures **salary** is non-negative.
- **Unique:** Ensures **phone** and **cnic** (National Identity Card) are unique.

WORKER SALARY TABLE

- **Description:** Manages salary payments for workers, including the amount paid, the period covered, and payment dates.
- **Constraints:** Ensures **amount** is non-negative.
- **Foreign Key:** Links **worker_id** to the **workers** table to ensure data integrity.

WORKER TABLE FOR AUTHENTICATION

- 
- **Description:** Holds authentication details for workers, such as usernames and passwords, ensuring secure access to the system.
 - **Unique:** Ensures **username** and **password** are unique.

ADD INVENTORY TABLE

- **Description:** Tracks inventory additions, including purchase dates, types of goods, quantities, and prices.
- **Constraints:** Ensures **weight** and **price** are positive values.
- **Unique:** Ensures **distributorId** is unique.

RETURN WORK TABLE

Description: Records details of returned work, including the weight of returned items, payment to workers, and related work IDs.

Constraints: Ensures **return_weight**, **item_weight**, and **money** are positive values.

Foreign Key: Links **work_id** to the **work** table to ensure data integrity.

POLISH SEND TABLE

- **Description:** Manages the sending of items for polishing, including the date sent, item details, and weight.
- **Constraints:** Ensures **weight** is positive.

POLISH RETURN TABLE

- **Description:** Tracks the return of polished items, including the return date, item details, and weight.
- **Constraints:** Ensures **return_weight** is positive.
- **Foreign Key:** Links **polish_id** to the **polish_send** table to ensure data integrity.

RECEIPTS TABLE

- **Description:** Stores receipt information for transactions, including customer details, transaction amounts, and quantities.
- **Constraints:** Ensures **amount** and **quantity** are positive values.
- **Unique:** Ensures **phone** is unique.

DELETED WORKERS DETAIL TABLE

- **Description:** Maintains a log of details for workers who have been removed from the system, including their personal information and the date they were deleted.

SECTION 2.5: IMPLEMENTATION DETAILS

The implementation involved setting up the database with necessary constraints and relationships, ensuring data integrity, and providing validation checks to maintain accuracy.

- **Database Setup:** The database schema was created, and initial data was populated as needed.
- **Validation:** Various checks and constraints were added to ensure data integrity.
- **Foreign Keys:** Established relationships between tables to maintain consistency.

Chapter 3: Results and Analysis



The Automated Inventory Management System proved to be an effective tool in streamlining inventory and workforce management processes, leading to better resource management and operational efficiency.

EFFICIENCY

The system demonstrated increased efficiency by automating manual processes and reducing the time required to manage inventory and worker details.

ACCURACY

By implementing constraints and validations, the system significantly reduced errors in data entry and management.

DATA INTEGRITY

The use of foreign keys and unique constraints ensured data consistency and integrity across the database.