CUSTOMER DATA MANAGEMENT SYSTEM

Submitted by

ANISH PRANAV RA2211003011624
HARI PRASANTH RA2211003011623

Under the Guidance of

Dr.S.SARAVANAN

ASSISTANT PROFESSOR, CTECH

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

MAY 2023

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this Project Report titled **Customer Data Management System** is the bonafide work done by ANISH PRANAV RA2211003011624 and HARI PRASANTH RA2211003011623 who completed the project under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

SIGNATURE
Dr.S.Saravanan
OODP – Course Faculty
Assistant Professor
Department of CTECH
SRMIST

SIGNATURE
Dr. Pushpalatha
Head of the Department
Department of CTECH
SRMIST

TABLE OF CONTENTS

INTRODUCTION

A customer management system in C++ is a program that can be used to manage customer data for a business. Customer data management refers to the processes, strategies, and technologies used to collect, store, organize and analyse customer data. The system should be able to store and manage customer data such a name, address, phone number, email and other relevant information. This data can be stored in data structures such as arrays, vectors or linked lists.

PROBLEM STATEMENT

Coding a customer data management system can come with some challenges. Here are some of the common problems faced :

- Data Security: One of the major concerns with any customer data management system is data security.
- Data Integration: Customer data may come from different sources, such as social media, emails, and phone calls. Integrating all of this data into a single system can be challenging and requires careful planning.
- Scalability: As the number of customers and data grows, the system must be able to scale up to handle the increased load.
- User Adoption:Encouraging user adoption and training users on how to use the system effectively can be a challenge.

MODULES OF PROJECT

Customer Relationship Management (CRM) systems with security measures and restricted data sharing play a crucial role in protecting customer information and ensuring data privacy. Here's some theoretical information about such systems:

Authentication and Authorization: A secure CRM system requires robust authentication and authorization mechanisms. Customers should be required to provide valid credentials (e.g., username/password, two-factor authentication) to access their details. Authorization mechanisms should grant access only to authenticated customers and ensure that they can view their data and perform authorized actions based on their role or permissions.

Secure Communication: The CRM system should utilize secure communication protocols such as HTTPS to encrypt data transmitted between the customer's device and the CRM server. This protects sensitive information from unauthorized interception and ensures data integrity during transit.

Data Encryption: To protect customer details stored in the CRM system, sensitive data such as passwords, financial information, or personal identification should be encrypted. Encryption algorithms (e.g., AES, RSA) can be used to encrypt data at rest, ensuring that even if the data is accessed unlawfully, it remains unintelligible.

Access Controls and Privileges: Access controls should be implemented to limit access to customer data within the CRM system. Fine-grained privileges and role-based access control (RBAC) can be used to define who can view, modify, or delete customer information. Access should be restricted to authorized personnel and monitored to detect any unauthorized activity.

Auditing and Logging: Implementing auditing and logging mechanisms allows for the tracking and recording of activities within the CRM system. This includes logging login attempts, access to customer details, and modifications made to customer records. By monitoring and analyzing these logs, suspicious or unauthorized activities can be identified and investigated promptly.

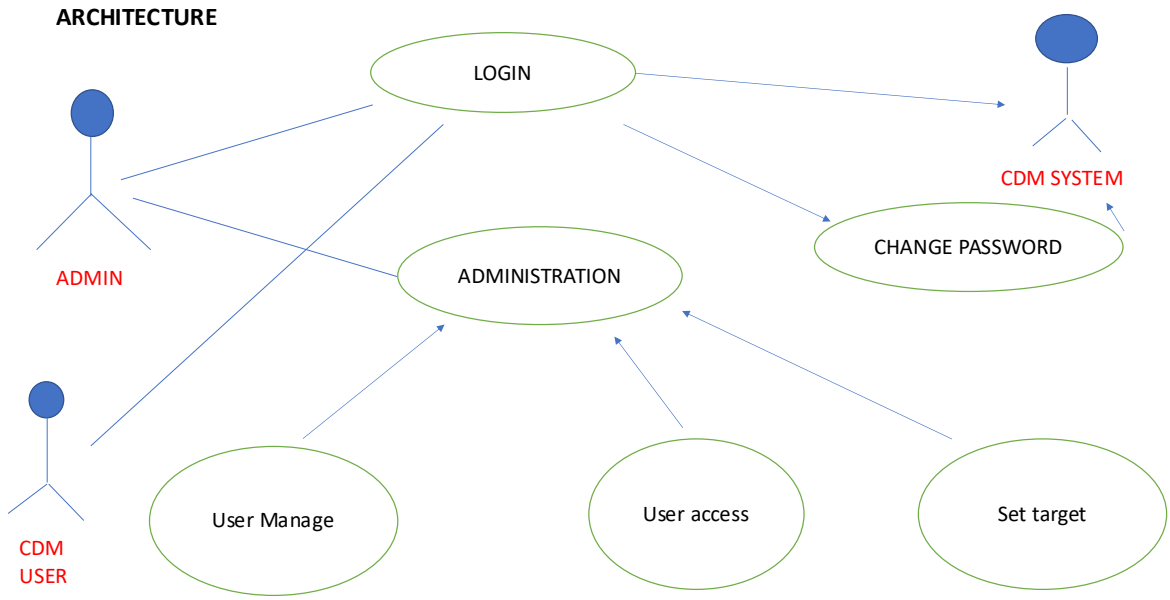
Data Sharing Controls: To ensure customer details are provided only to specific websites or service providers, the CRM system can implement access control lists (ACLs) or similar mechanisms. This allows the CRM system to verify the identity of the requesting website or service provider and determine whether it has permission to access customer data. The CRM system should maintain a trusted list of authorized recipients and validate requests against this list before sharing customer details.

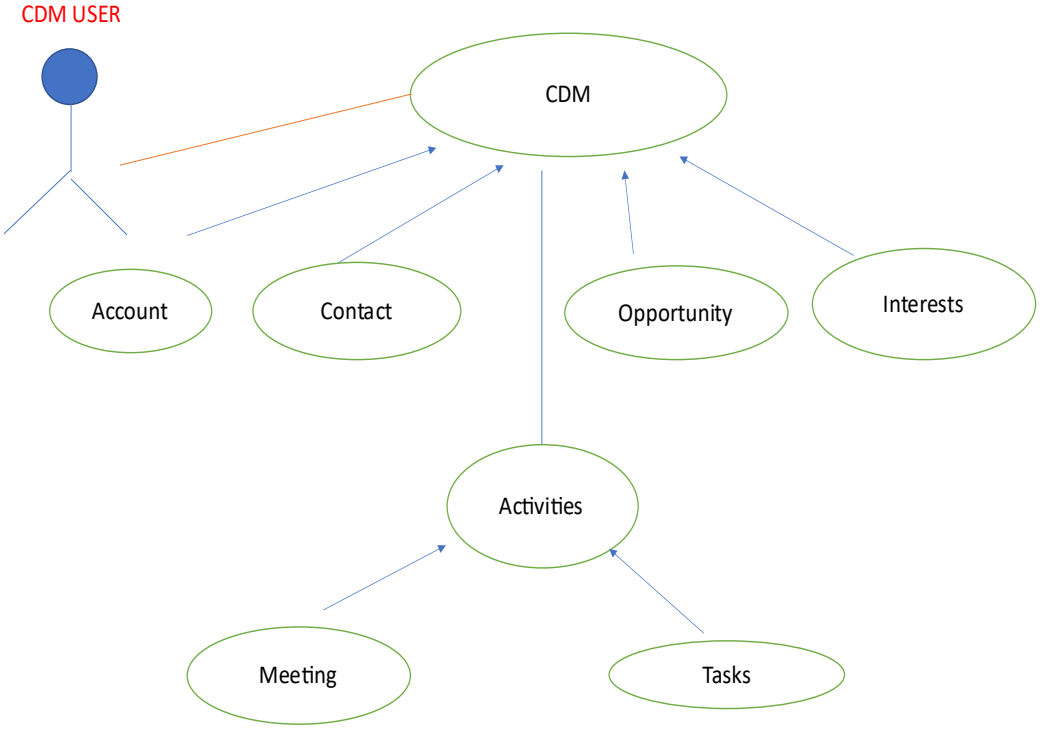
Compliance with Data Privacy Regulations: It is crucial for a secure CRM system to comply with relevant data privacy regulations, such as the General Data Protection Regulation (GDPR) or the California Consumer Privacy Act (CCPA). Compliance involves obtaining customer consent for data processing, providing mechanisms for data subject rights (e.g., data access, rectification, erasure), and implementing necessary technical and organizational measures to protect customer data.

These theoretical considerations highlight some of the important aspects of building a secure CRM system with restricted data sharing. Actual implementation and customization will depend on specific business requirements, industry regulations, and the technologies employed. It's recommended to consult with security and legal experts to ensure compliance and best practices in data protection.

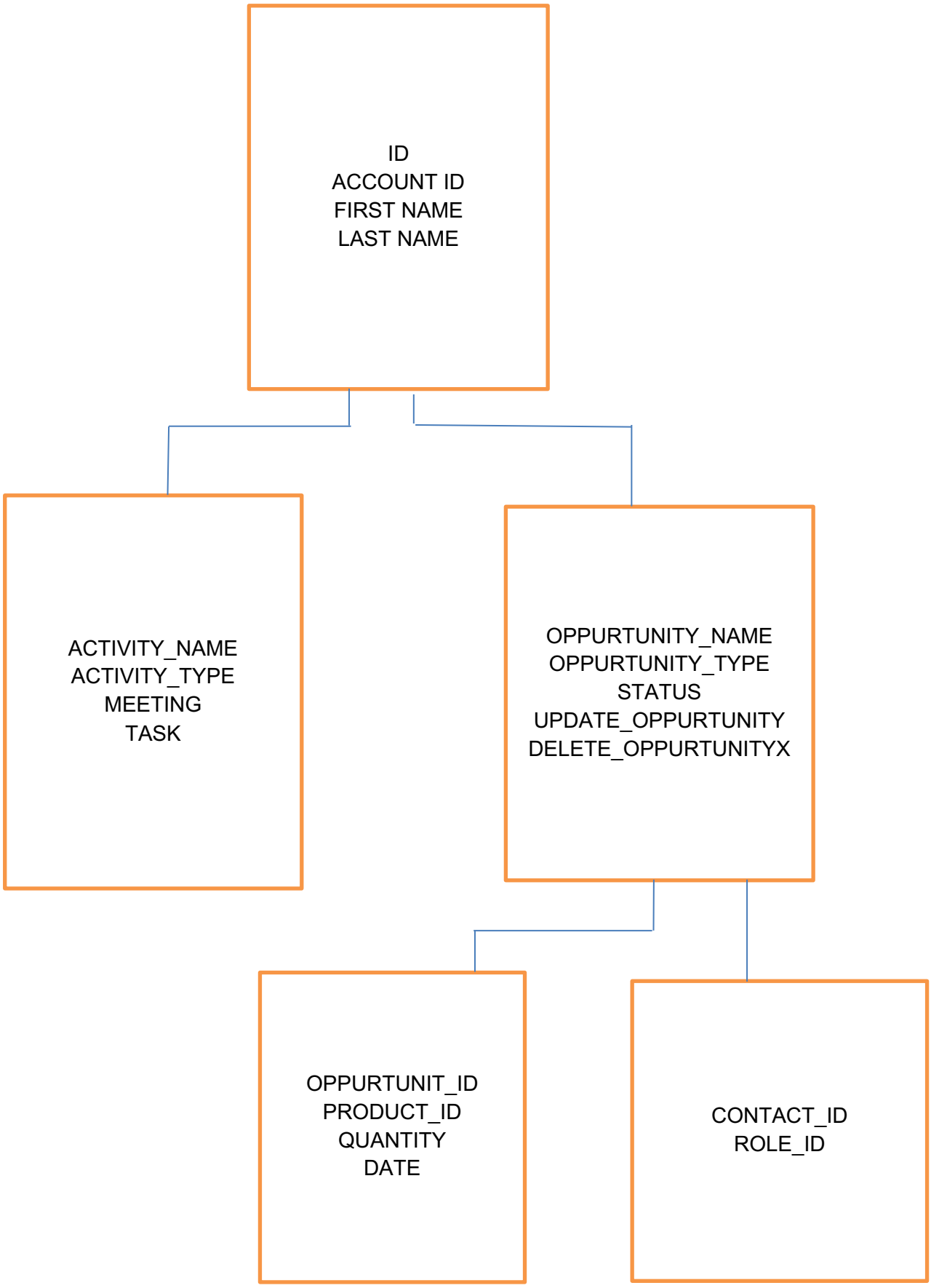
DIAGRAMS

USE CASE DIAGRAMS

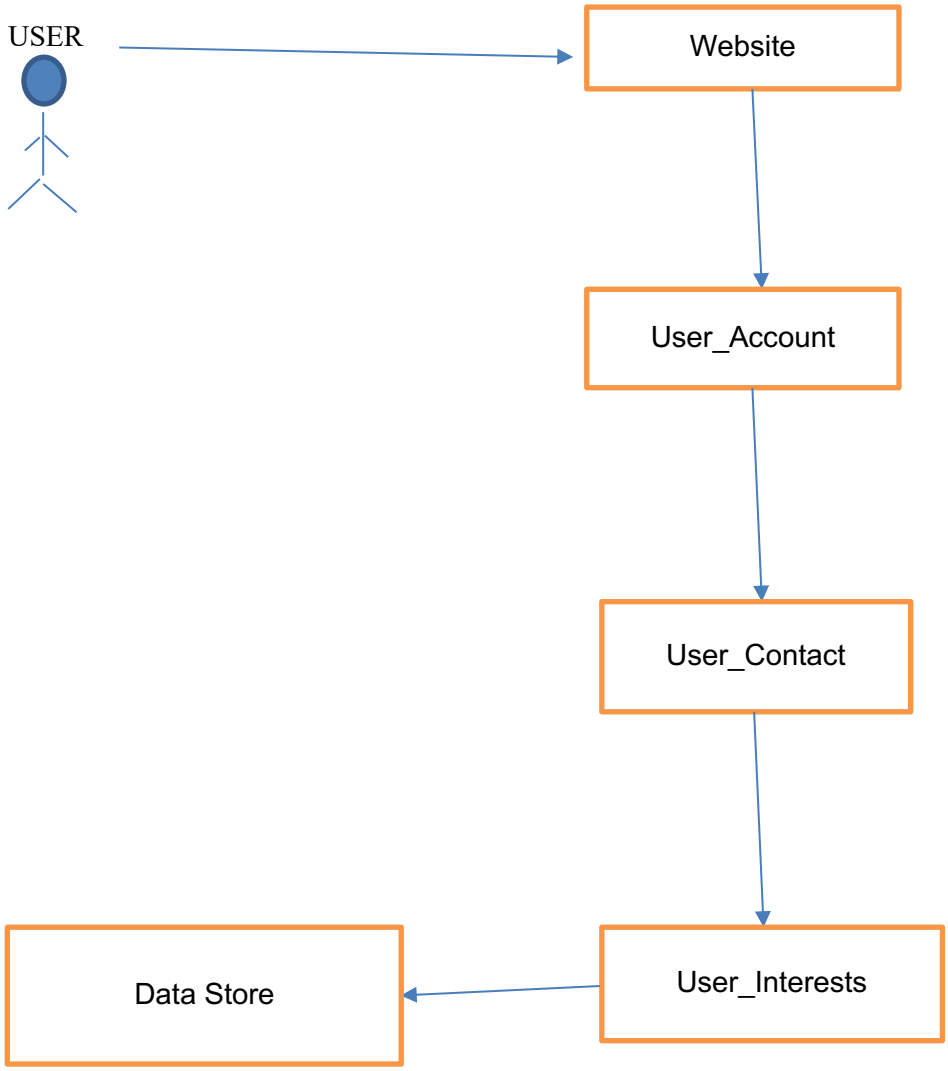




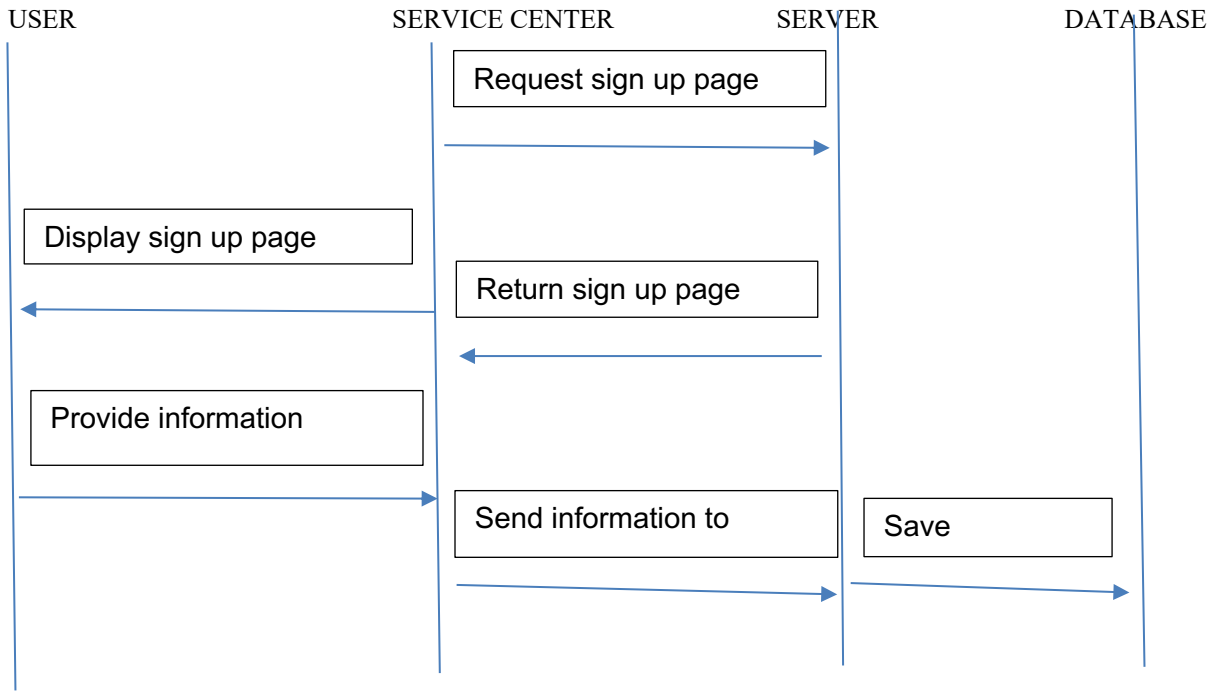
CLASS DIAGRAM



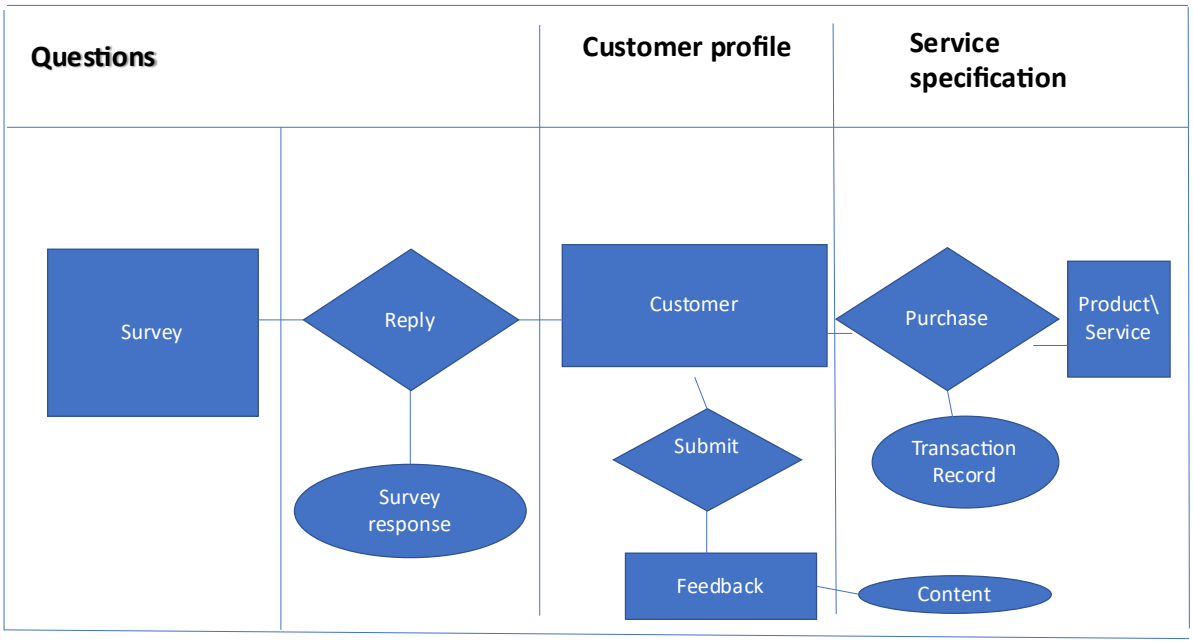
COLLABORATION DIAGRAM



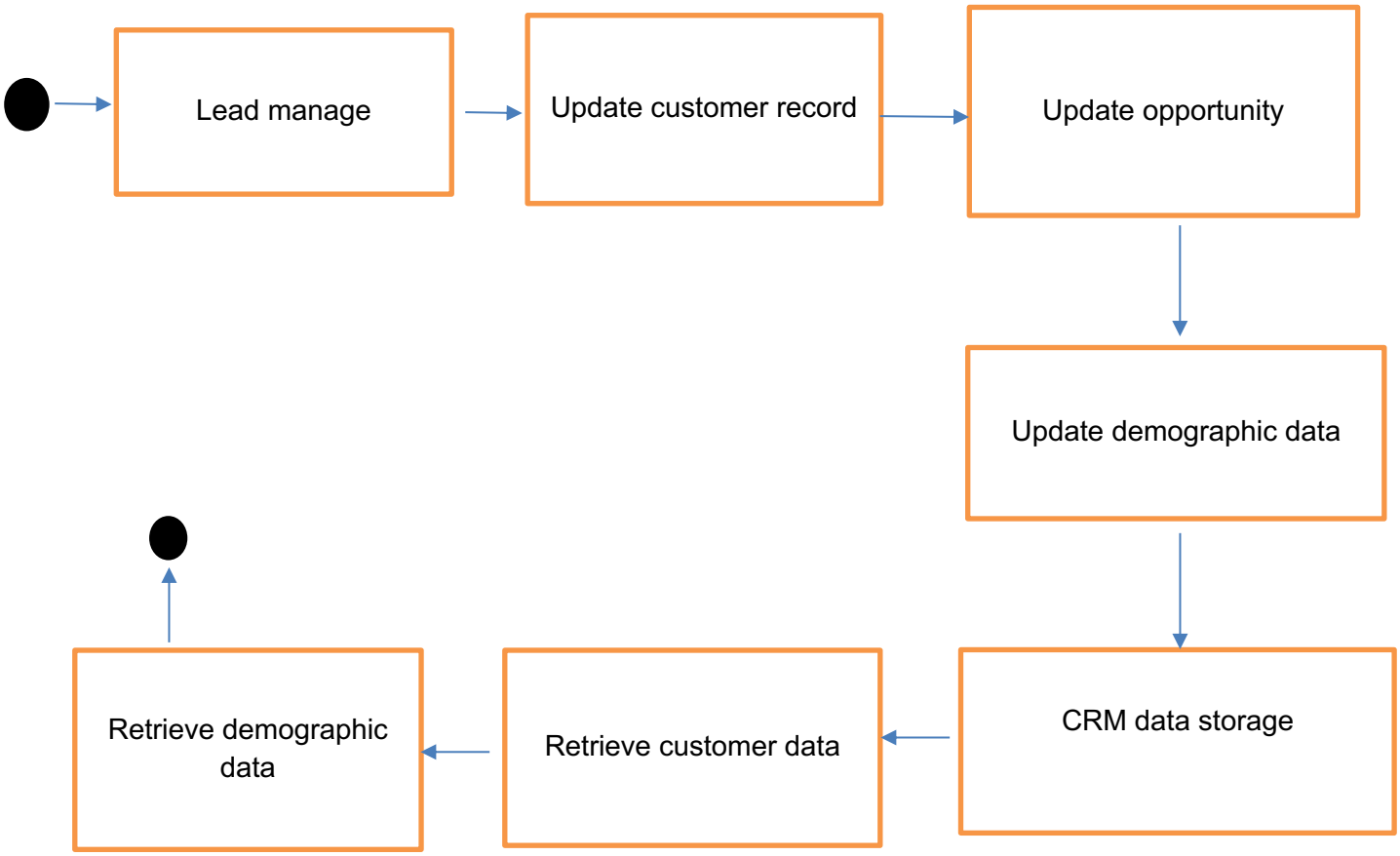
SEQUENCE DIAGRAM:



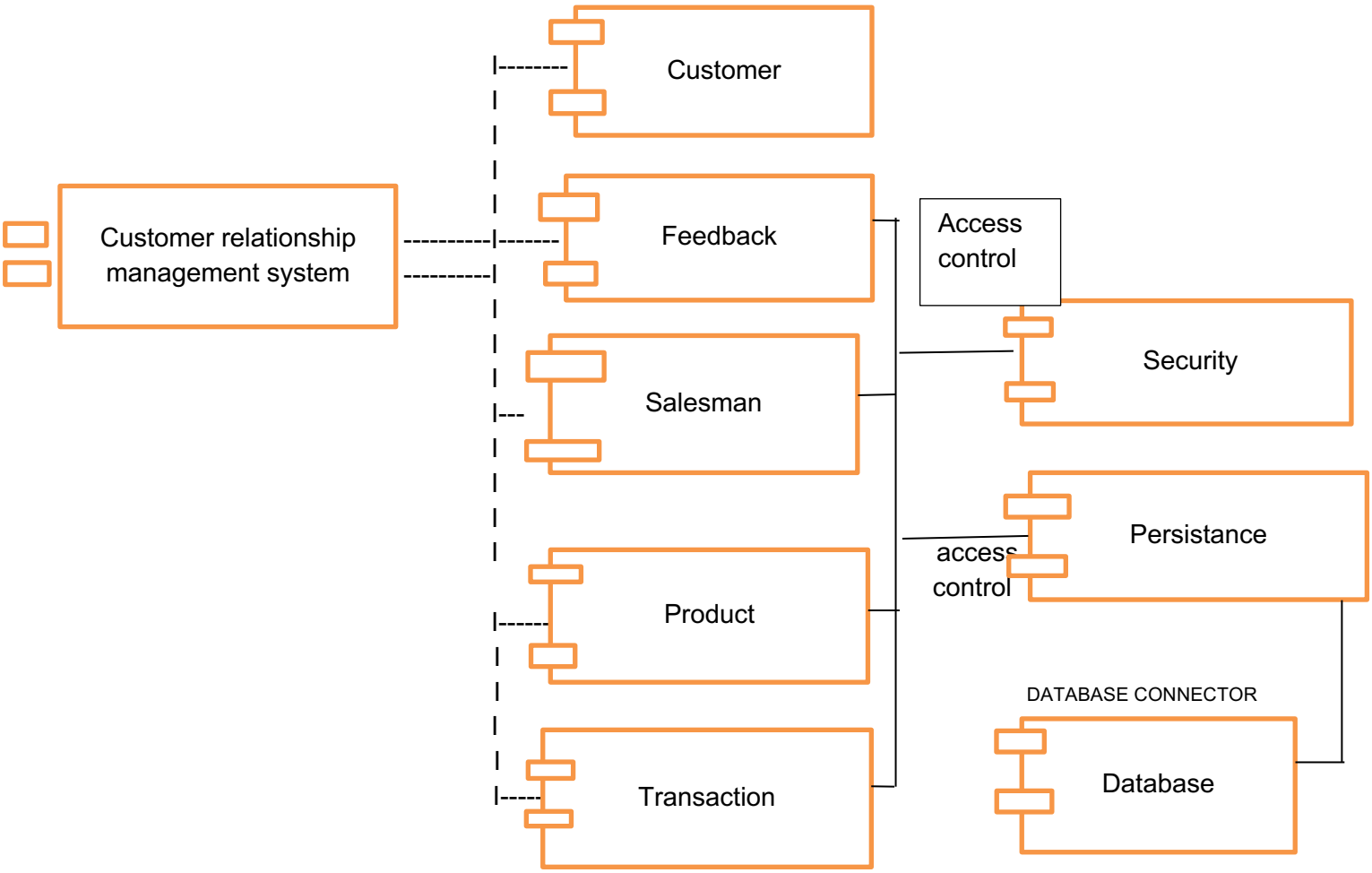
ACTIVITY DIAGRAM



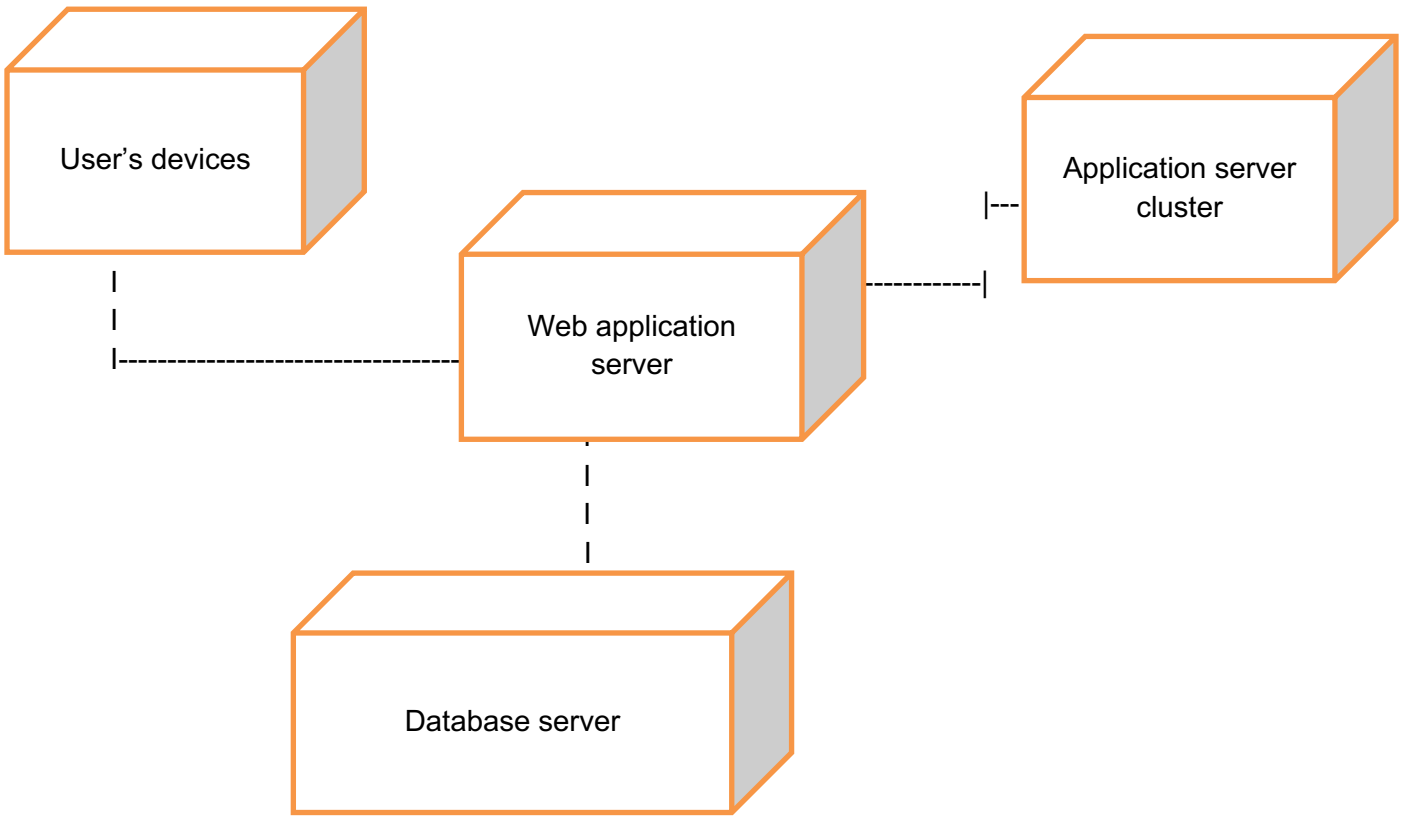
STATE CHART DIAGRAM



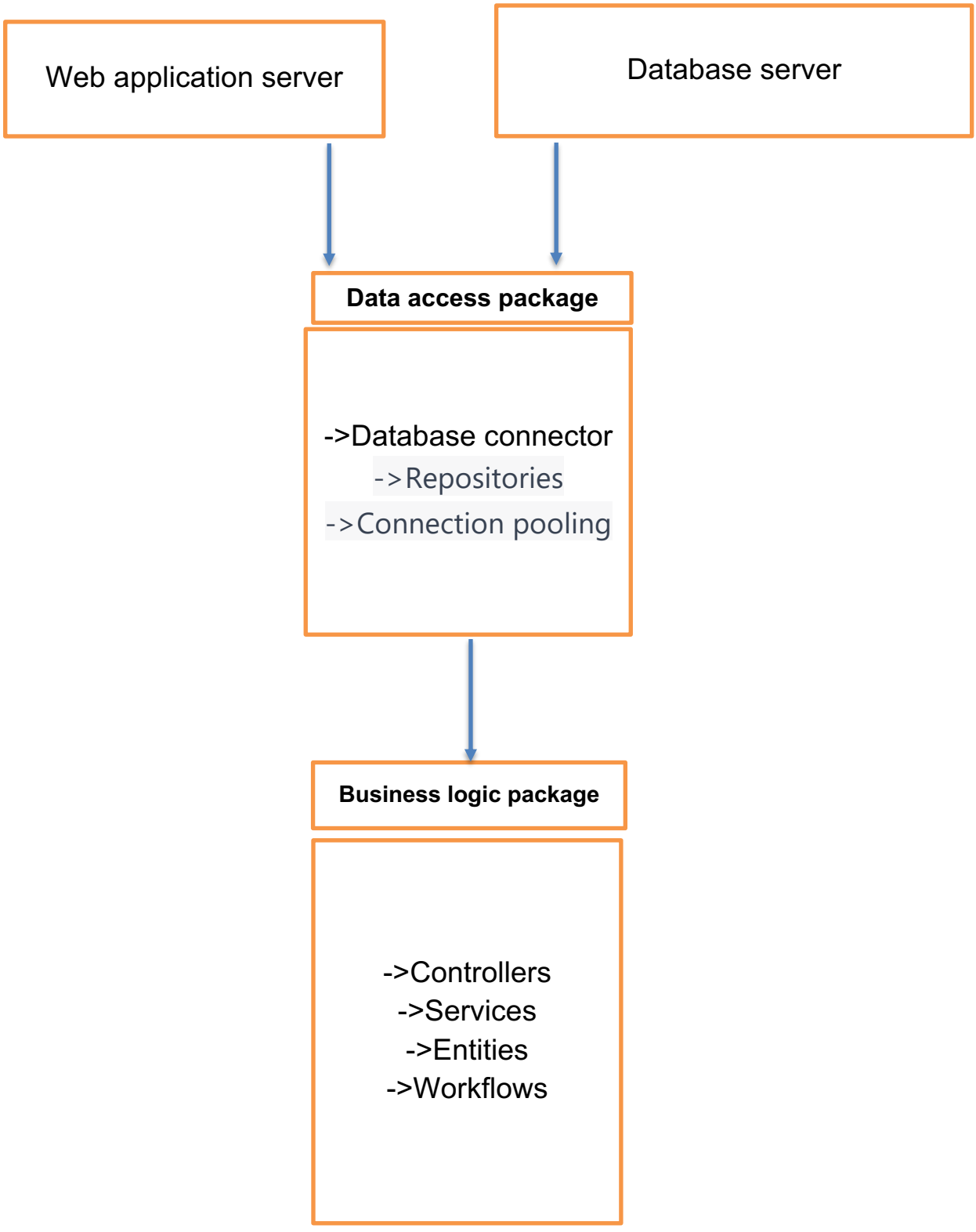
COMPONENT DIAGRAM



DEPLOYMENT DIAGRAM



PACKAGE DIAGRAM



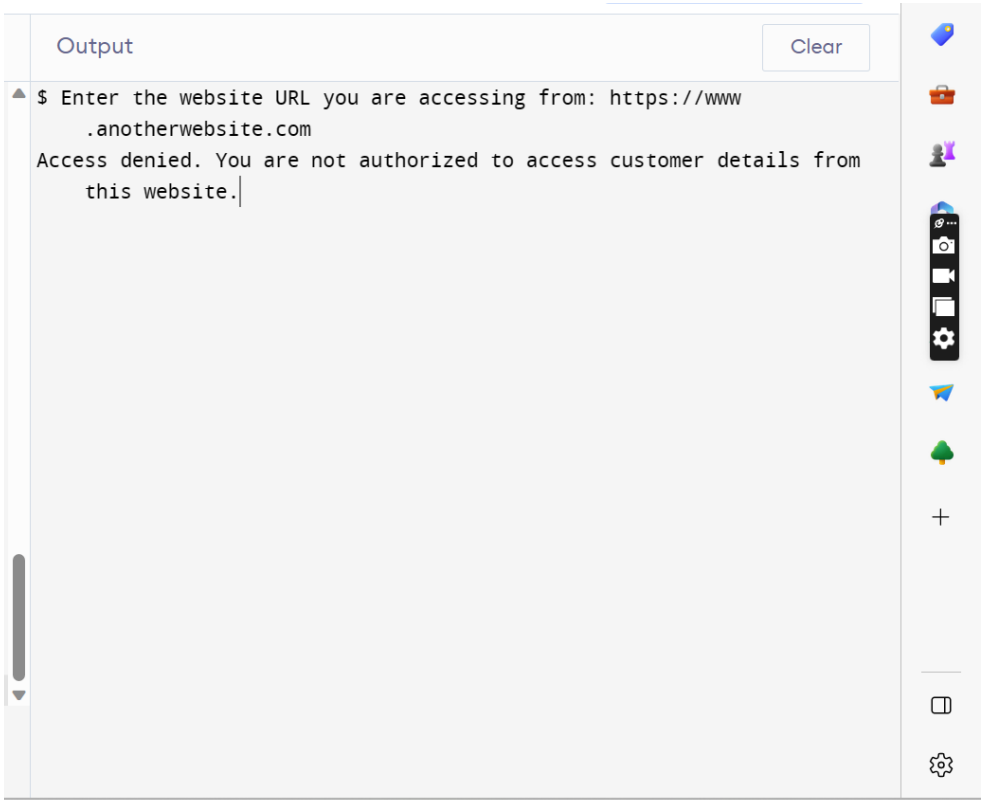
SOURCE CODE

```
#include <iostream>
#include string>
#include <curl/curl.h>
//Function to send HTTP GET request to the server
std: :string sendHttpRequest(const std::string& url, const
        std: string& authToken)
{
        std: string response;
        CURL* curl = curl_easy_init();

        if (curl)
        {

                // Set the URL and Authorization header
                curl_easy_setopt(curl, CURLOPT_URL, url.c_str());
                curl_easy_setopt(curl, CURLOPT_HTTPHEADER,
                        "Authorization: " * authToken);
                // Set the callback to receive the response
                curl_easy_ setopt(curl, CURLOPT_WRITEFUNCTION, []

                // Set the callback to receive the response
                curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, []
                (char* ptr, size_t size, size_t nmemb, std
                : :string* data) {
                data--append(ptr, size * nmemb);
                return size * nmemb;
        });
curl easy setopt(curl, CURLOPT WRITEDATA, &response
        );

// Send the HTTP GET request
CURLcode res = curl_easy_perform(curl);

// Check for errors
if (res != CURLE_OK)
{
        std:: cerr << "Failed to send HTTP request: “ <<
                curl_easy_strerror (res) << std:: endl;
}

}
        // Clean up
```

```
        curl_easy_cleanup(curl)
}
else
{
        std:: cerr << "Failed to initialize CURL. " << std
                : :endl;
}
        return response:
}

int main()
{
        // Server URL
        std::string serverUrl = "https: //api.example.com
        /customer /details";
// Authorization token for access control
std::string authToken= "Bearer YOUR AUTH TOKEN"

// Customer-specific website URL
std: string customerWebsiteUrl = "https : //ww.example.com";

//Check if the customer is accessing from the specific website
std: string referringWebsite;
std: cout << "Enter the website URL you are accessing from: “;
std: cin >> referringWebsite;

if (referringWebsite customerWebsiteUrl)
{

        // Send HTTP GET request to retrieve customer details
        std: :string customerDetails = sendHttpRequest
                (serverUrl, authToken) :

if (referringWebsite - customerWebsiteUrl)
{
        // Send HTTP GET request to retrieve customer details
        std: string customerDetails = sendHttpRequest
                (serverUrl, authToken) ;

        // Display customer details
        std:: cout << *Customer Details: \n"
                customerDetails << std:: end;
}
slse
{
        std:: cout << "Access denied. You are not authorized
                to access customer details from this website
                << std::endl;
```

```
}
return 0;
```

OUTPUT



APPLICATIONS

CDMS has several applications that can help business improve their customer relationships and drive sales. Some of these are:

- Customer service: CDMS allows business to access customer information quickly and easily.
- Marketing automation: CDMS can automate marketing processes such as email campaigns, social media advertising and lead generation.
- Personalization: With CDMS, businesses can personalize their marketing efforts to individual customers.
- Cross-selling and upselling: CDMS can suggest complementary products and services to customers based on their purchase history, behaviour and preferences.

FUTURE SCOPE

The future scope of Customer Data Management System (CDMS) is quite promising, as businesses continue to focus on providing personalized customer experiences. Here are some potential future developments:

- Artificial Intelligence and Machine Learning: CDMS will increasingly leverage Artificial Intelligence(AI) and Machine Learning (ML) algorithms to analyse customer data and generate insights.
- Real-Time data analysis: CDMS will move towards real-time data analysis, enabling businesses to respond to customer behaviour and preferences in real time.
- Increased adoption: As businesses of all sizes recognize the value of customer data, CDMS will become more widely adopted across industries.

RESULTS

- Implementing a customer data management system (CDMS) can yield a range of results that can benefit businesses.
- CDMS can help businesses to gain a better understanding of their customers and their needs.
- CDMS can automate many data management processes, such as data entry, data cleaning, and data analysis. This can save businesses time and resources
- CDMS can help businesses to identify and address customer issues proactively. By resolving customer issues quickly, businesses can reduce churn rates and improve customer retention.
- Implementing a CDMS can lead to a range of positive outcomes for businesses.

CONCLUSION

CDMS is mainly used for :

- Data collection
- Data storage
- Data quality
- Data analytics
- Data privacy

Therefore we implemented a C++ code to analyse the customer's data.