

CRM System Documentation

Author: Pleșca Alexandru

Faculty of Economic Sciences, Business Informatics, Year II

Table of Contents

[Project theme and purpose: 2](#_Toc187586500)

[Why did I choose this theme? 2](#_Toc187586501)

[Functionalities of the project: 2](#_Toc187586502)

[Code explanation: 3](#_Toc187586503)

[Data Exporting and Saving: 5](#_Toc187586504)

[Challenges Faced: 6](#_Toc187586505)

[Conclusion 6](#_Toc187586506)

## Project theme and purpose:

### Why did I choose this theme?

I chose the Customer Relationship Management (CRM) system as the theme for this project because CRM systems play a vital role in modern businesses, especially in organizing customer data, managing employees, and tracking orders. CRM system directly impact business decision-making and enhance customer satisfaction. As a business informatics student, I believe this project allowed me to integrate programming skills with business concepts, providing a practical application for what I’ve learned so far.

### Functionalities of the project:

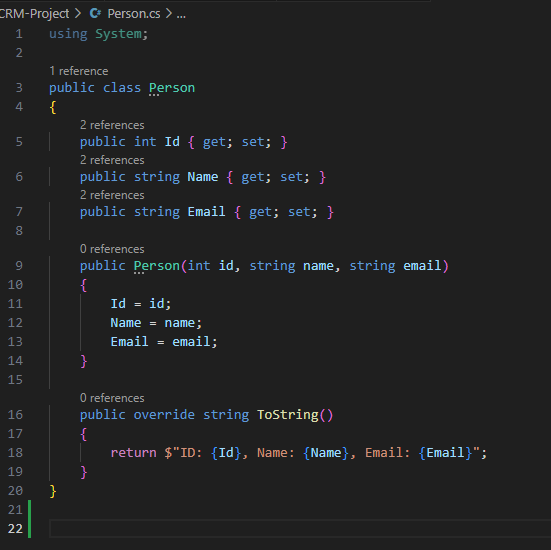
1. Client management: Add, view and save client information
2. Employee management: Add and display employee details
3. Order tracking: Record and display orders associated with clients and employees
4. Data persistence: Save and load data to and from CSV files
5. Validation: Ensures correct input formats to avoid errors during runtime

## Code explanation:

1. Superclass – Persons.cs

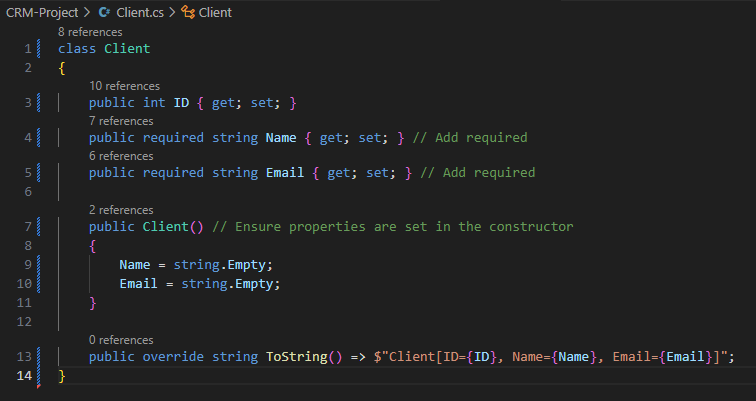
This class serves as the parent class for **Client** and **Employee**

It has: Private fields (id, name, email), public properties (encapsulate access to the fields), constructor (initializes values upon object creation)



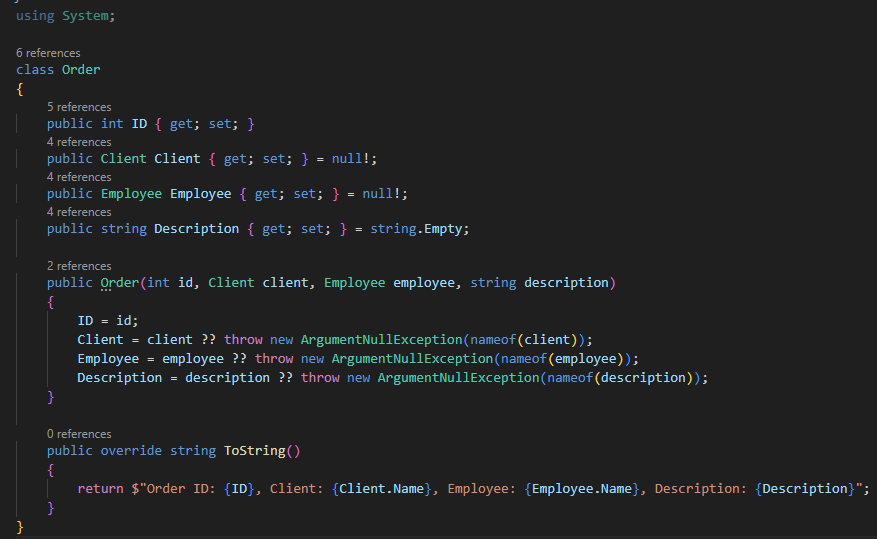
1. Subclass – Client.cs

The **Client** class inherits from **Person** and adds a unique property **Phone** to store phone number



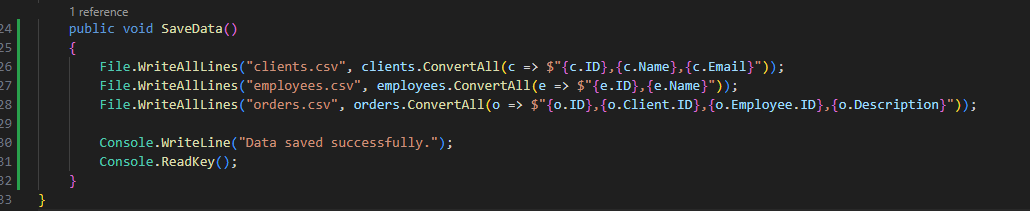
1. Additional Class – Order.cs

The **Order** class is independent and manages order information, including IDs for the client and employee associated with the order.



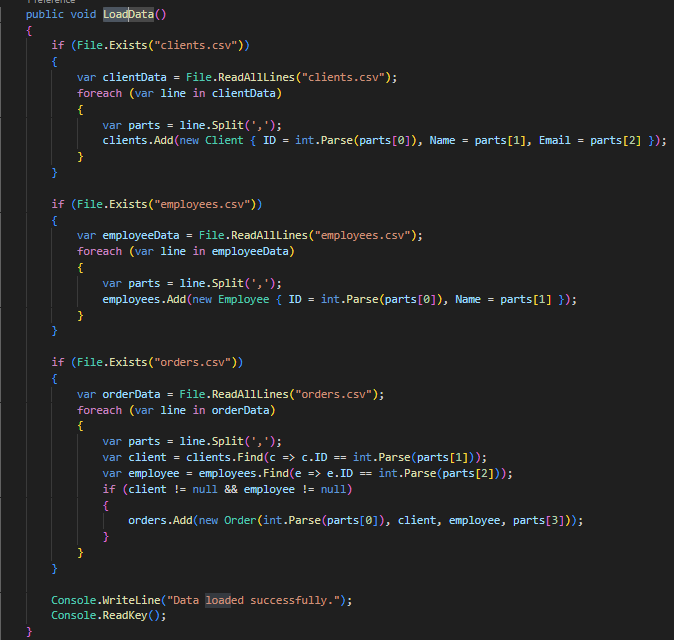
## Data Exporting and Saving:

1. Saving data:



Each line represents one entry and data is stored as comma-separated values

1. Loading data:



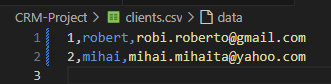
## Challenges Faced:

1. Implementing file handling with CSV required careful handling of data formats and error checking.
2. Ensuring input data is valid was time-consuming, especially for numerical fields.
3. During data loading, handling null values and default assignments took effort to debug and resolve.

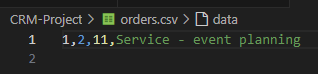
## Conclusion

This project provided practical implementation of object-oriented programing principles such as inheritance, encapsulation and polymorphism in a business application. It demonstrated the value of data persistence and validated input handling. While some parts were challenging, they offered valuable learning experiences that will support future projects.

Here we can see the saved data in the CSV files that will be loaded when the application will be used by the user.







Here we can see the UI of the app when we run it. We can choose what we want to do with the app by entering one number from 1 to 6.

