**UNIVERSIDAD FRANCISCO DE VITORIA**

**ESCUELA POLITECNICA SUPERIOR**

**Object Oriented Programming**

**Practical work I**

**Base-Converter Calculator**

**Hugo Losada Cabeza**

Table Of Contents

[1. Introduction 1](#_Toc199014689)

[2. Description 1](#_Toc199014690)

[2.1. Design Decisions 1](#_Toc199014691)

[2.2. Development Decisions 2](#_Toc199014692)

[2.3. Class Diagram 2](#_Toc199014693)

[3. Problems 3](#_Toc199014694)

[4. Conclusions 3](#_Toc199014695)

# 1. Introduction

This document presents a comprehensive design overview for the PracticalWork 2 application. The application is developed using .NET MAUI and includes user authentication functionalities and a numeric system converter. The main goal of this document is to outline the architecture, design, development decisions, and challenges encountered during the implementation process. This documentation provides a foundational understanding of the application's workings and serves as a reference for future enhancements or similar project developments.

# 2. Description

## 2.1. Design Decisions

The application structure is modular and based on the Shell navigation model provided by .NET MAUI. Each major feature (registration, login, user info display) is implemented as a separate page, enhancing readability, maintainability, and testability.

To ensure consistency and reusability, common layout structure and navigation pattern are used throughout the app. By using MAUI's cross-platform capabilities, the application achieves a uniform look and behavior across multiple devices and operating systems.

User data is persisted using a simple CSV-based storage system, prioritizing ease of implementation and simplicity over scalability or security. This approach allows rapid prototyping and focuses on essential file Input Output mechanisms that are commonly used in educational contexts.

The user interface design prioritizes clarity and responsiveness. The use of MAUI’s VerticalStackLayout and other layout controls ensures that all content is properly aligned and displayed across different screen sizes and orientations.

## 2.2. Development Decisions

* **Shell Navigation**: Chosen for its built-in support for hierarchical navigation, route-based access to pages, and clean URI syntax. It provides a streamlined way to manage navigation between LoginPage, RegisterPage, RecoverPasswordPage and UserInfoPage.
* **CSV Storage**: Selected due to its simplicity. The CSV file stores user information including name, username, email, password, and number of operations. Data is easily readable and modifiable, which suits the scope of the application.
* **Validation**: Validation procedures were used to guarantee that user input follows the correct formats and that necessary fields are filled in.
* **Error Handling**: User-friendly alerts and exception handling are used to notify users of issues such as invalid input or failed file operations. These mechanisms improve usability.
* **Exit Functionality**: The use of Environment.Exit(0) provides a simple way to terminate the app, although it's recognized that this is not always suitable in mobile environments.

## 2.3. Class Diagram

Diagrama

El contenido generado por IA puede ser incorrecto.

# 3. Problems

* **File Path Management**: Initially, hardcoding the CSV file path led to platform-specific issues. This was resolved by verifying the file existence dynamically and standardizing the file naming conventions across devices.
* **User Unique Validation**: Reading and parsing the CSV file to identify existing usernames proved challenging. Edge cases like missing fields or corrupted entries had to be carefully handled to prevent runtime errors.
* **Platform Exit Handling**: Implementing application exit functionality using Environment.Exit(0) raised concerns about compatibility with mobile platforms. Alternative methods may be explored for platform-appropriate termination in future versions.
* **Data Security**: Passwords are stored in plaintext within the CSV file, which is a significant security concern. Although acceptable for a proof-of-concept or educational tool, future iterations must incorporate secure storage and password hashing.
* **Number of Operations Executed**: This function of the program has not been implemented because it wasn not known.

# 4. Conclusions

The development of Practical Work 2 has provided a valuable learning experience in cross-platform mobile application development using .NET MAUI. The project demonstrates the implementation of core features like user registration, login, data persistence, and navigation within a modular architecture.

**Lessons Learned:**

* The critical importance of validating user inputs to maintain application integrity.
* Challenges associated with cross-platform file handling and UI consistency.
* The significance of designing simple yet effective data storage solutions when security is not the primary focus.

**Future Considerations:**

* Migration to a database solution to handle data more efficiently and securely.
* Implementation of encryption and secure hashing algorithms for sensitive information.
* Introduction of unit testing and logging for better maintainability and debugging support.
* Expansion of the numeric system converter with more complex functionality and a more intuitive interface.

This application lays a solid foundation for more complex developments and serves as a successful demonstration of MAUI’s capabilities in delivering functional cross-platform applications.