|  |
| --- |
|  |
| Home Affairs System Description |
| System Description |
|  |
| **Weylin Renison 12009634** |
| **11/13/2013** |

|  |
| --- |
|  |

Table of Contents

[Introduction 3](#_Toc372487923)

[Document Purpose 3](#_Toc372487924)

[Summary 3](#_Toc372487925)

[Problem Statement 3](#_Toc372487926)

[System Overview 4](#_Toc372487927)

[Architectural Strategies 4](#_Toc372487928)

[Program Flow and Features 4](#_Toc372487929)

[Application Specifications 5](#_Toc372487930)

Hold the Ctrl key and click on a chapter if you wish to jump to that chapter straight away.

# Introduction

## Document Purpose

This application was written in response to an early 2nd year college assignment.

The purpose of this document is to describe the problem was to be solved with this software and how the solution software has been achieved and developed.

The document covers a problem description, the project specifications, system overview and architectural strategies.

The document is intended to give the any new developers and companies an over view of the application and how it all fits together.

There are numerous other documentation associated with Home Affairs v1.0 that can be found on the website.

Other documents include:

* ERD Diagrams
* Use Case Diagrams
* User Manuals
* System and Program Flowcharts
* Test Data used for debugging and demoing
* Assignment Brief

# Summary

The program is to generate home affairs documents using a C# ADO.NET application that demonstrates reading and writing of text files, exception handling, validation, object oriented programming, inheritance, event handling and graphical user interfaces.

# Problem Statement

Home affairs office in South Africa are often heavily congested, in an effort to reduce this an application is required to help with the process of generating ID documents, Passports and Death certificates for individuals willing to use the home affairs application.

# System Overview

## Architectural Strategies

The system uses the C# ADO.NET framework for this application, this is part of the specifications provided.

Text files where used to record any necessary data:

* User.txt
  + Username
  + Password
  + User particulars(address ect)
* IDUser.txt
  + ID Document details
* PassUser.txt
  + Passport details

User.txt is generated when a new user is registered, as soon as this user creates an ID document the details are recorded to the IDUser.txt and subsequently PassUser.txt is recorded.

Objects are used to represent the user and the documents, the object attributes are then recorded to the data text files for future reference.

# Program Flow and Features

The user starts off with the login form, from here they can either log in or create a new user. When the user registers a new account a user object is created and the data is store to the appropriate text file. The user is now returned to the log in form where they can now log in to the application.

The application now search the IDUser.txt file to determine if the user has pervious generated an id before, if they have their ID number is displayed on the ID Document tab that is focused first on the main form. If they have not they can now enter their details to generate an ID Document. An IdDoc object is created and the details are written to the appropriate text file.

The user cannot generate a passport document until they have an ID Document therefore the passport tab is disabled till such time. When the user generates a passport document a passDoc object is created and written to the appropriate text file.

When the user generates a death certificate an object is created but not written to file. The user can save the document to their hard drive instead.

All the above documents can be printed.

# Application Specifications

1. Your application must make use of event handling and include debugging and exception handling.
2. Create a graphical user interface (GUI) login page for your project. The login page must:
   1. Allow the user to either register or login to the application. A first time user will need to register first before logging into the system.
   2. Accept a user name and password.
3. The application you develop should enable a user to apply at the Home Affairs department for biographical documentation (i.e. ID Book, Passport, Birth Certificate, marriage certificate, death certificate etc.). Your application should offer a selection of available documentation (minimum of 3 documents) that the user can apply for. Use Google™ to get examples of these documents and the types of information required for each document. The user will enter the additional information.
4. The base class will contain the most basic information about the user (Birth date, name and address) and the more document specific information will be stored in then derived (sub) classes.
5. Derive subclasses for the different documents you need to generate.
6. You will need to generate an ID number according to the following rules so you need to determine what data is input by the user. An example of C# programming code used to validate an ID number can be found here: http://www.dirkstrauss.com/programming/south-african-id-number-validation-in-c. Use it to determine how to generate an ID number. The rules for creating ID numbers are:  The first six (6) numbers are the birth date of the person in YYMMDD format;  The next four (4) are a gender, 5000 and above is male and below 5000 is female;  The next number is the country ID, 0 is South Africa and 1 is a foreigner;  The second last number used to be a racial identifier but now means nothing and is usually an eight (8);  The last number is a check bit. This verifies the rest of the number.
7. Ensure that the documents you generate contain all the information required in an official document issued by Home Affairs. Please note these official documents are not representative of real people (they are in actual fact fraudulent forged documents)