

System Design Document

PAT Phase 2 – Grade 12

Prepared by Marco de Sá

Maragon Mooikloof

3/4/2022

Table of Contents

1. Interface Design 1

2. Program Flow 6

3. Class Design and OOP Principles 8

4. Secondary Storage Design 9

5. Explanation of Secondary Storage Design 10

# Interface Design

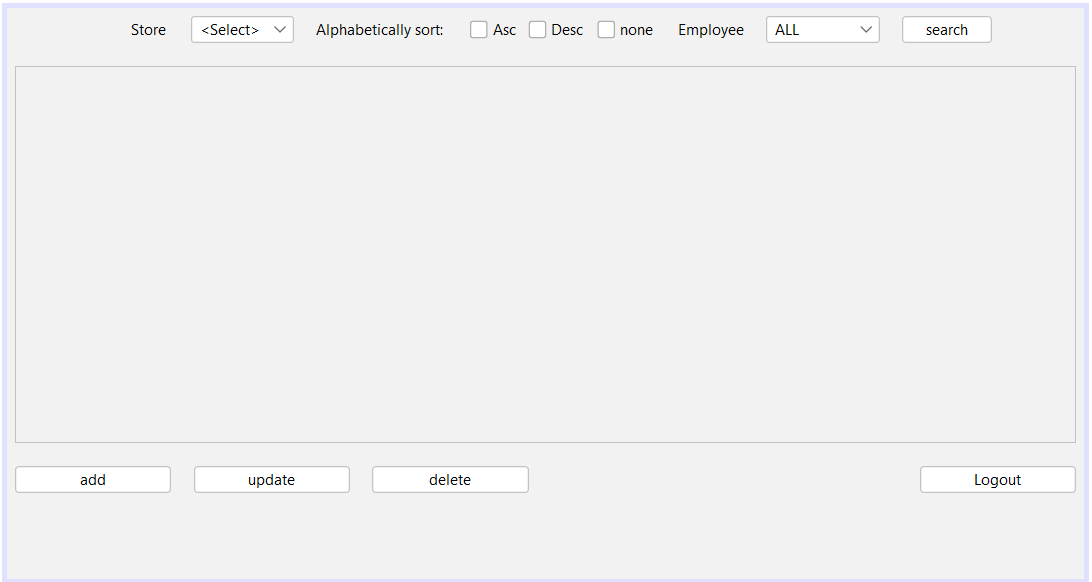
## Graphical user interface, application Description automatically generatedLogin Screen

|  |  |
| --- | --- |
| **Description** | This screen will be use to login. It contains text boxes for the user to enter their username and password into and a Register and Login button. |
| **Data** | The user’s name username and password will be entered. |
| **Actions** | * **Register**   Check if values are filled in. If not, display a message.  Check if user exists. If it does, display a message.  If not, add the user to the text file.   * **Login**   Check if values are filled in. If not, display a message.  Check if user name exists. If not, display a message.  If it does, check if the password is correct. If not, display a message.  If valid, display the Menu screen. |

## Graphical user interface, application Description automatically generatedMenu

|  |  |
| --- | --- |
| Description | This screen can open the database, help document or go back to the login screen |
| Data | n/a |
| Actions | * Database   Opens the document   * Help   Opens the help document   * Logout   Will return to the Login screen |

## Main

****

|  |  |
| --- | --- |
| Description | This screen views and can open the add, update and delete screens as well as return to the login screen |
| Data | n/a |
| Actions | Sort by store, sort by manager or employee and alphabetically sort names |

## Add screen

**Graphical user interface, application

Description automatically generated**

|  |  |
| --- | --- |
| Description | Accepts data to add a new employee |
| Data | users Firstname, Surname, manager, password, Store |
| Actions | Add inputted data to database or hides screen |

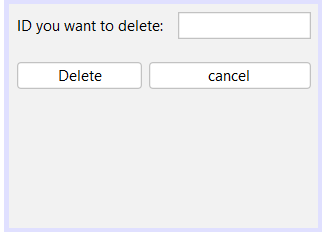
## Update

**Graphical user interface

Description automatically generated**

|  |  |
| --- | --- |
| Description | Accepts data to update the existing employees |
| Data | Users ID, Firstname, Surname, manager, password, Store |
| Actions | updates inputted data to specific database entry or hides screen |

## Delete

****

|  |  |
| --- | --- |
| Description | Accepts data to delete an existing employees |
| Data | Users ID |
| Actions | deletes a specific database entry or hides screen |

# Program Flow

HEADER: FrmLogin

BEGIN

IF “Log In” button is pressed

IF username field is empty OR password field is empty

DISPLAY error message that values are required

ELSE IF user name exists

IF Password correct

Display Menu screen

IF “Logout” button is pressed

Close program

ELSE

DISPLAY error message that the password is incorrect

ELSE

DISPLAY error message that the user name does not exist

END

HEADER: FrmMenu

BEGIN

IF “Database” button is pressed

DISPLAY Main screen

IF “Help” button pressed

Display “Help” Document

IF “Logout” button is pressed

Display Login screen

END

HEADER: FrmMain

BEGIN

IF “Add” button is pressed

Display FrmAdd

IF “Update” button is pressed

Display FrmUpdate

IF “Delete” button is pressed

Display FrmDelete

IF “Logout” button is pressed

Display Login screen

END

HEADER: FrmAdd

BEGIN

If “Add” button is pressed

Add Inputted information into database

If “cancel” button is pressed

Display FrmMain

END

HEADER: FrmUpdate

BEGIN

If “Update” button is pressed

Update Inputted information into database

If “cancel” button is pressed

Display FrmMain

END

HEADER: FrmDelete

BEGIN

If “Delete” button is pressed

Delete Inputted information From database

If “cancel” button is pressed

Display FrmMain

END

# Class Design and OOP Principles

**3.1 User Class**

|  |  |
| --- | --- |
| **User Class** | |
| **Properties** | **Description** |
| - Username: string | Holds the unique username of a user. |
| - Password: string | Holds the password of a user. |
| **Methods** | |
| + Contructor(String inUserName, String inPassword) | Instantiates a user object and assigns values to the fields of the object. |
| + getUserName(): string | Returns the username, |
| + getPassword():String | Returns the password. |
| + setUserName() | Sets Username |
| + setPassword () | Sets Password |

|  |  |
| --- | --- |
| **UserArray class** | |
| **Fields** | **Description** |
| - userArr[]:User | Holds the array of objects of all users. |
| - size: int | Holds the number of objects |
| **Methods** | |
| + Constructor() | Reads data from file creates an object for each user. |
| + userExists: boolean | Returns a value to indicate if the user exists. |
| + userPasswordValid: boolean | Returns a value to indicate if the password is valid. |

3.2 EmployeeManager

|  |  |
| --- | --- |
| **EmployeeManager class** | |
| **Properties** | **Description** |
| - HobbyZA : DB |  |
| - rs : ResultSet |  |
| **Methods** | |
| + constructor() | Reads data from file creates an object for each user. |
| + getEmployeeData() | Return EmployeeData |
| + addEmployeeToDB(int Store, String text, String text0, String text1, boolean Mngr) | Adds Employee to a new entry |
| + changeDB(String inStatement) | Executes query |
| + updateEmployeeData(int Store, String text, String text0, String text1, boolean Mngr, String text2) | Updates entries with new data |
| + deleteEmployeeFromDB(int ID) | Delete entry from database |

|  |  |
| --- | --- |
| **DB** | |
| **Properties** | **Description** |
| - resultSet: ResultSet |  |
| - statement: Statement |  |
| - connection: Connection |  |
| - url: String |  |
| - driver: String |  |
| **Methods** | |
| + constructor() | Reads data from file creates an object for each user. |
| + queryDB(String inStatement) |  |
| + changeDB(String inStatement) |  |

# Secondary Storage Design

**4.1 tblUser**

o Field names

EmployeeID

StoreID

Firstname

Surname

Username

Password

Manager

o Field types

EmployeeID: AutoNumber

StoreID: Number

Firstname: Short Text

Surname: Short Text

Username: Short Text

Password: Short Text

Manager: Yes/No

*o Description of fields*

EmployeeID: A unique number Automatically assigned

StoreID: A number associated with specific stores

Firstname: Firstname of Employee

Surname: Surname of Employee

Username: Username of employee

Password: Password of Employee

Manager: Whether the employee is a manager

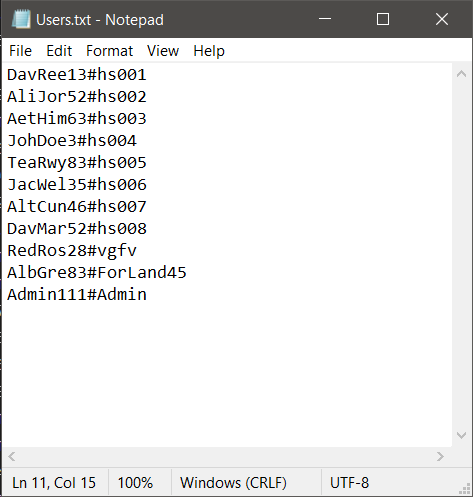
*o Primary key*

*EmployeeID*

*o Sample data for each table*

| **tblEmployees** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **EmployeeID** | **StoreID** | **Firstname** | **Surname** | **Username** | **Password** | **Manager** |
| 1 | 1 | Darius | Reece | DarRee35 | hs001 | Yes |

New data entered will be inserted into the table by a query in the java project

**

Format:

Username#Password

For the login button to work this text document will be used

# Explanation of Secondary Storage Design

There are two primary storage methods that are going to be used. First there is going to be a database that will store the vast majority of data in the program. Secondly there will be a text file called users.txt and it will store the basic login details because of ease of access.