Personal Financial Management System

Introduction

Scope

The system is designed to help managing and accessing data regarding all expanses and incomes of the user, using a universally-compatible and fail-safe database, which may be accessed in the future by various network machines and applications in case this system is not used anymore.

The system will also generate monthly reports based in the inputs, and data will be backed up on a remote server.

The system will run as both a client and a server from a single user , operated by the user through a minimalistic and straightforward interface.

The system will be able to generate weekly and monthly spending reports with QR codes that can be easily stored and shared.

This will enforce the storage of all latest data on the same local machine, allowing for continuous offline service in case of a network fail, alongside with the availability of online backup.

add softwares requirements

Overview

This document contains a preliminary design for the Personal Fianacial Management System, described in the next section (Software Design Description). # add figures

In addition, it contains a general outline for the testing and logging of the program's activity (Tests and Monitor).

Terminology

- Incomes, Outcomes - names refer to either the program components or their related database (see overview).
- Personal Financial Management System - refers either to the main component of the program or to The entire system.

Software design description

General Flow

The Personal Financial Management System will be composed of two independent sub-systems, each connected to its own database on a local server:

1. Incomes – offers the functionality of adding incomes entries into a database, removing and updating it, displaying incomes by its common fields, by current month or by selected days (current or previous).
2. Outcomes - offers the functionality of adding outcomes entries into a database, removing and updating it, displaying outcomes by its common fields, by current month or by selected days (current or previous).

The user will be able to work with one sub-item at a time, and switch between them from the main system menu. Each one of the sub-system will be visible to select from a calendar, represent the current month and day.

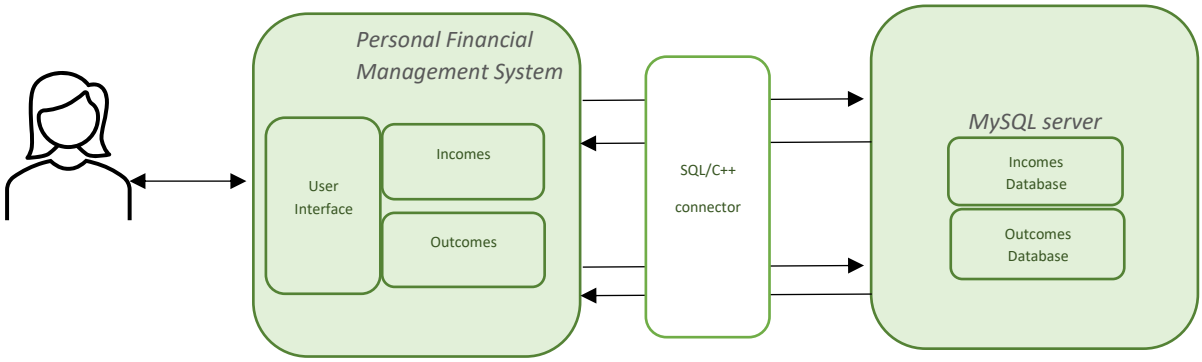

Fig. 1: General flow of the Personal financial management system. User input will be chosen by the user interface GUI, making function call choices at runtime, producing SQL statements and executing them for the relevant database. Result sets returned from the SQL server will be interpreted and displayed to the user.

Software Architecture

The Personal Financial Management System, including the Incomes, Outcomes, and user interface (via QT GUI) will be implemented with C++, and will communicate with the SQL server via a MySQL/C++ Connector library.

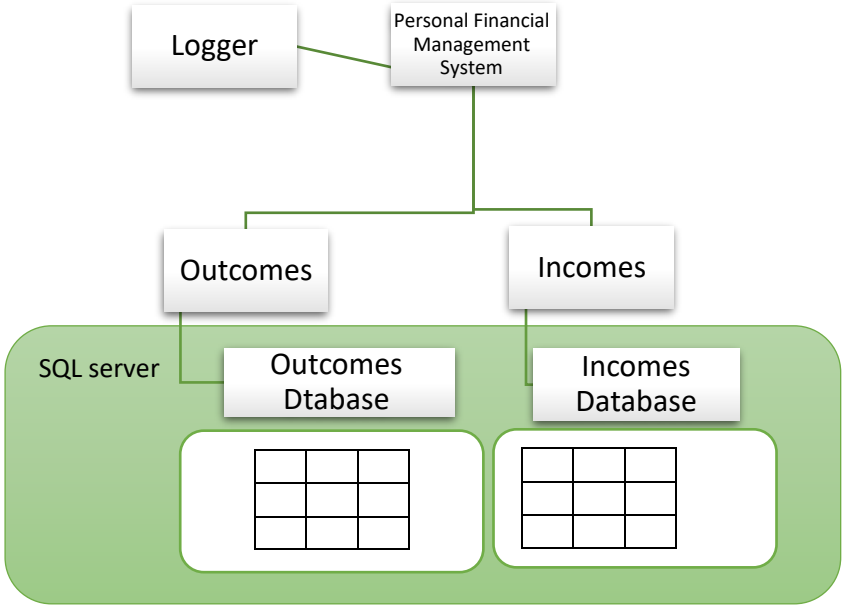

Fig. 2: General preliminary class diagram of the Animal Shelter System. See additional Fig. 2.a for the fully detailed UML.

The two sub-systems will operate independently from each other and in single-threaded manner. As they live in the same scope in the main window, the two-sub system wont affect each other while updating and visiting each one of them.

In the main window, there will be an assessing of the incomes – outcomes of the user, as received as input in the two sub-systems.

User Interface

The current design of the system rely on the QT GUI functionality.
When entering the System, the user must provide details in order to connect his SQL server. Later, The user will be able to choose the action he would prefer to perform. The options are as follow:

- Add new outcome in specific date
- Add new income in specific date
- Search outcome
- Search income
- Show incomes – outcomes

In each option menu, the user will be able to go back to the last screen, go to main screen, or exit the program.

Data Handling

All databases will be stored on local storage, and on remote cloud directory, which will be done periodically updated with the changes made. This will be done using a tool such as Dropbox.

Integration with other systems

Connection with the local SQL server is done over TCP connection, with the local host IP and port, using a root user and password. Such connection is also available to other authorized network devices as clients.
Exporting data from the databases should be available via the user interface to a .csv format copy, which will allow to view and edit the copy with commonly used softwares such as Windows Excel.

Tests and Monitor

Tests

1. Unit testing for every module of the system through the development.
2. Testing the finished system with various use cases and edge cases.
3. Testing across other Linux distributions and architectures

Logs

In the release version, scenarios to be logged will be:
- Startup and exit from the system
- Errors and Warnings generated during initiation or exit of the system
- Exceptions generated in case of failure to update changes in the databases
- Uninitiated loss of connection with the SQL server during runtime
- An external device attempting to connect to the SQL server