ТЕМА НА ПРОЕКТА:

„CWallet“-софтуерни приложения

Автор:

Борис Златев, 8б клас,

Михаил Маринов 9а клас

ученици в МГ „Баба Тонка“, гр. Русе

Научен ръководител (консултант):

Снежана Тодорова, старши учител по информационни технологии в МГ „Баба Тонка“, гр. Русе

**Резюме:**

„CWallet“ е приложение, чиято цел е да улесни проследяването на финансите на потребителя.

**Цел на приложението:**

Целта на проекта е да проследява приходите и разходите на отделен човек или цяло семейство. С интуитивен интерфейс подпомага индивидуалното разглеждане на различни видове финанси . Всеки човек може да използва приложението без значение от възрастта му.

„CWallet“ е разработен на C#. Изображенията са обработени в: Autodesk Sketchbook; Figma. Базата данни се поддържа на MySQL Client качен на сървър и се разработва на MySQL.

**Основни етапи в реализирането:**

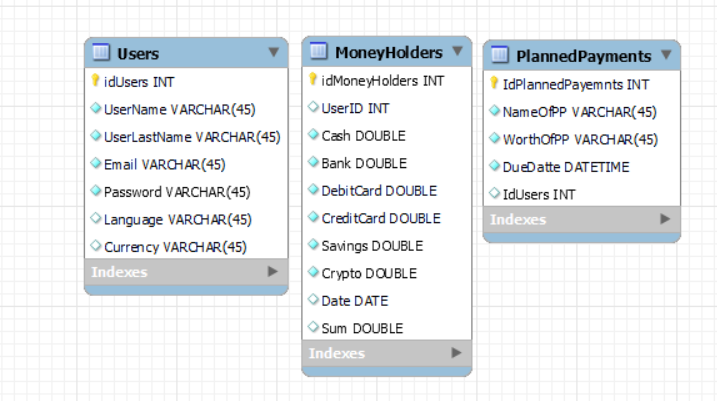
Идеята за проекта дойде, когато и двамата се чудехме как да улесним работата на наш приятел, който не можеше да организира финансите си. Ние решихме да направим програма, която обединява всякакъв тип разходи без значение от типа им. Като следващ етап на реализация преди популяризиране, сме планирали добавяне на повече функционалности и възможности за персонализация.

**Ниво на сложност:**

Реализирахме приложението, чрез база данни, качена на сървър, но срещнахме затруднения в настройването и инсталацията на Mysql Client на сървър и връзката на сървъра с приложението. Успяхме да решим проблема, като променихме системните настройки на cloud-а.

**Логическо и функционално описание на решението:**

Програмата използва сървър където да записва информация от различните типове, които предлага приложението: видовете финанси, планирани покупки и плащания, както и настройки, които са в процес на разработка. Не всички полета в базата трябва да бъдат запълнени, като те или имат стойности зададени по под разбиране, или са със стойност от друга колона в друга таблица. На *фиг. 1* се вижда схемата на цялата база данни.



*Фиг. 1*

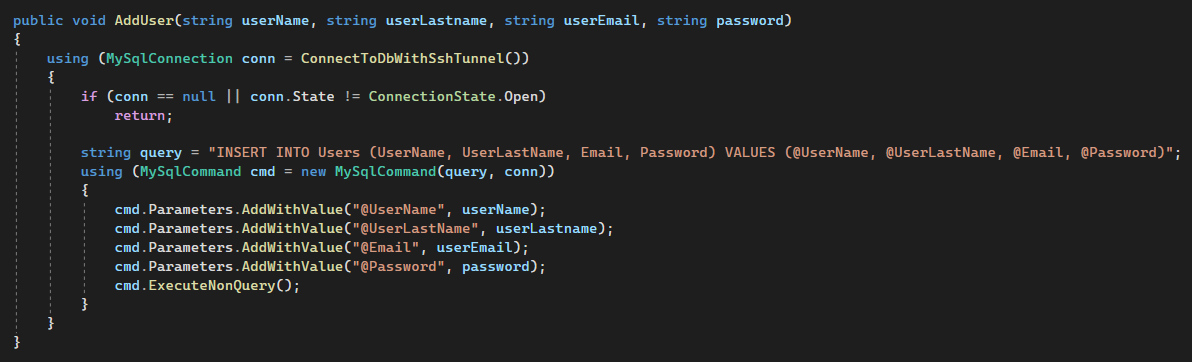
Имаме клас наречен DbConnector, в който имаме методи и инстанции, чрез които да извършваме определени операции в приложението. Връзката се осъществява чрез метода ConnectToDbWithSshTunnel(). Този метод ще бъде използван в абсолютно всяка операция свързана с комуникация до сървъра.

A computer screen shot of a computer code

Description automatically generated

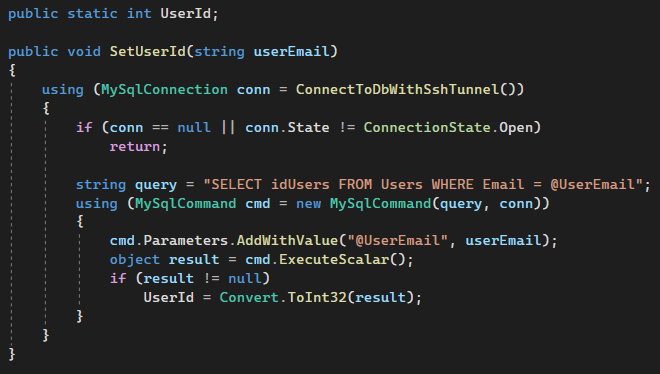
*Фиг. 2*

Вторият метод в този клас се нарича AddUser(). Този метод се използва за добавяне на информация за нов потребител в таблицата `Users` в MySQL база данни. Методът започва с установяване на връзка с базата данни чрез `ConnectToDbWithSshTunnel()`. Използва се `using` оператор, за да се гарантира, че връзката ще бъде правилно затворена и освободена след употреба. Проверява се дали връзката не съществува, ако не е активна то кода спира до там. Дефинира се низ за SQL заявка, който съдържа INSERT INTO заявка за добавяне на записи в таблицата `Users`. Тя специфицира четири колони, `UserName`, `UserLastName`, `Email`, и `Password`, и използва параметризирани заявки (`@UserName`, `@UserLastName`, `@Email`, `@Password`) за предотвратяване на SQL инжекционни атаки. Чрез друг `using` оператор се инициира нова MySQL команда с дефинирания низ за заявка и връзка с базата данни. Към командата се добавят параметри за UserName, UserLastName, Email, и Password със съответните им стойности. Накрая се изпълнява тази non-query команда, което означава, че тя не връща данни, но извършва действие върху базата данни - в този случай, добавя нов запис.



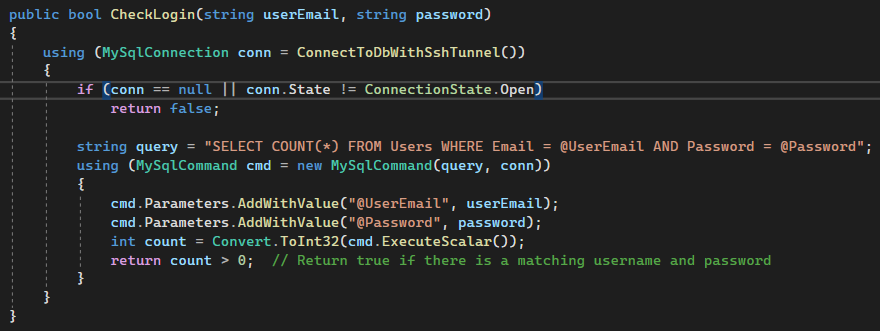
*Фиг. 3*

Третият метод се нарича SetUserId(). Този метод се използва за добавяне на информация за нов потребител в таблицата `Users` в MySQL база данни. Методът започва с установяване на връзка с базата данни чрез `ConnectToDbWithSshTunnel()`. Дефинира се низ за SQL заявка, който съдържа INSERT INTO заявка за добавяне на записи в таблицата `Users`. Заявката специфицира четири колони, `UserName`, `UserLastName`, `Email`, и `Password`, и използва параметризирани заявки (`@UserName`, `@UserLastName`, `@Email`, `@Password`) за предотвратяване на SQL инжекционни атаки. Чрез друг `using` оператор се инициира нова MySQL команда с дефинирания низ за заявка и връзка с базата данни. Към командата се добавят параметри за UserName, UserLastName, Email, и Password със съответните им стойности. Накрая се изпълнява тази команда, което означава, че тя връща данни и добавя стойност в споделена променлива за цялото приложение.



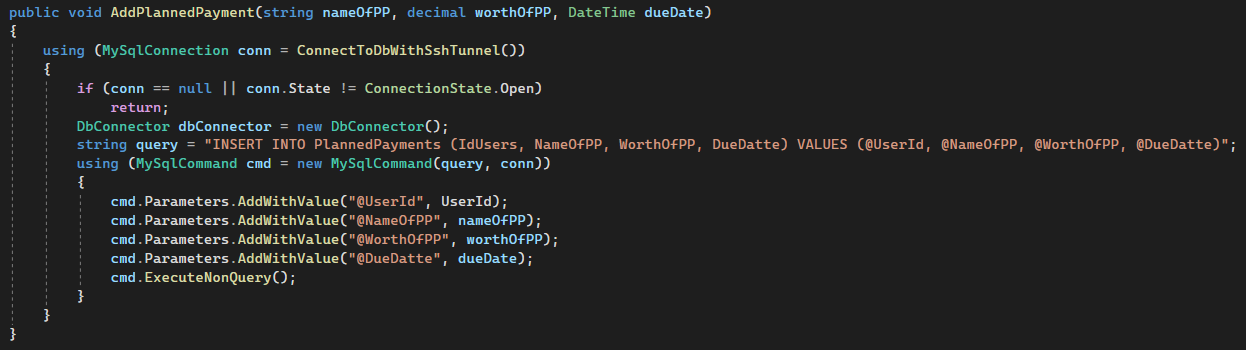
*Фиг. 4*

Четвъртата операция в класа DbConnector носи името CheckLogin(). Той се изпълнява в прозореца LogInWin. Неговата работа е да провери въведените в textbox-ове имейл и парола. Методът работи като чете имейл и парола от текстовите полета в прозореца и проверява за съответствия между двете.



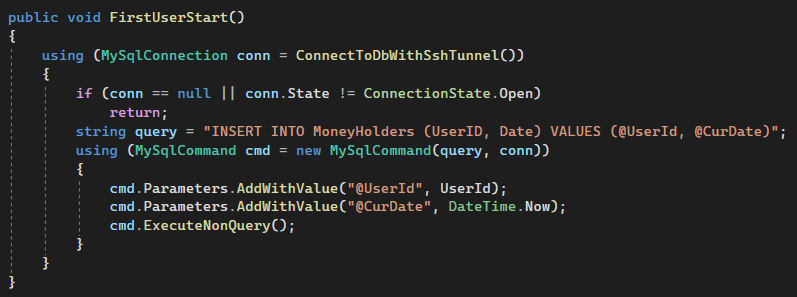
*Фиг. 5*

Следващия метод е от ключово значение за голяма част от сегашната версия на приложението. Неговото име е AddPlannedPayment(). Той изпълнява функцията да добавя видове планирани плащания към таблицата PlannedPayments. Извиква се в прозореца AddPlannedPaymentWin, като взима име, стойност и дата, на която трябва да бъде извършено плащането.



*Фиг. 6*

Шестият метод се казва FirstUserStart(). Този метод се използва за добавяне на информация за нов потребител в таблицата `MoneyHolders` в базата данни. Методът започва с установяване на връзка с базата данни чрез `ConnectToDbWithSshTunnel()`. Използва се `using` оператор, за да се гарантира, че връзката ще бъде правилно затворена и освободена след употреба. Проверява се дали връзката е null или не е отворена. Ако е изпълнено някое от тези условия, методът приключва преждевременно, за да се избегнат грешки свързани с null референции или неотворена връзка. Дефинира се низ за SQL заявка, който съдържа INSERT INTO заявка за добавяне на записи в таблицата `MoneyHolders`. Заявката специфицира две колони, `UserID` и `Date`, и използва параметризирани заявки (`@UserId`, `@CurDate`) за предотвратяване на SQL инжекционни атака. Чрез друг `using` оператор се инициира нова MySQL команда с дефинирания низ за заявка и връзка с базата данни. Към командата се добавят параметри за UserID и Date със съответните им стойности. Стойността за дата се получава от текущата дата и час. Накрая се изпълнява тази non-query команда, което означава, че тя не връща данни, но извършва действие върху базата данни - в този случай, добавя нов запис.



*Фиг 7*

**Реализация:**

Работихме в среда на Visual Studio 2022. Програмата е написана на C#. Базата данни се обработва с MySQL Client, връзката между нея и приложението е установена чрез MySQL Server. Логото и всички други графики са създадени и обработени в Autodesk Sketchbook, лично от нас.

**Описание на приложението:**

В момента приложението не се нуждае от инсталация, тъй като е в процес на усилена разработка и е на изпълним файл. Интерфейсът е графичен, за да е максимално дружелюбен за потребителя.

**Бъдещо развитие:**

В много близък срок планираме и дори сме започнали разработката на нови функционалности и доизглаждане на интерфейса с по-изчистен вид. Ще се стремим да направим още 2 версии на приложението – една онлайн и една за телефони и таблети, за да може потребителят да няма затруднения и да носи своя помощник навсякъде със себе си. След като се измисли как да се направи сигурна защита на личните данни възнамеряваме да добавим възможността за автоматично следене на сметката/те чрез банкова сметка или номер на карта, тай като това е голямо улеснение за потребителя.

**Заключение:**

С помощта на приложението ще се спести време, като улеснено потребителя ще може да проследява финансите си. Структурата на приложението позволява проследяването на финансовото състояние на потребителя за даден период от време.