

Specyfikacja projektu LabVIEW

ATM Machine

Sekcja 2 Podsekcja 2

Revision 1.0

19.03.2021

Skład podsekcji:

Marcin Bochenek

Mateusz Dzieżok

Sebastian Nowak

Piotr Szymczyk

Michał Żurawka

Wstęp

Ludzie od pokoleń szukali możliwości depozytowania pieniędzy, w średniowieczu odpowiednikiem dzisiejszych banków był zakon templariuszy. Pierwszy bankomat został zaprojektowany w 1939 roku przez amerykańskiego wynalazcę, Ormianina z pochodzenia, Luthera G. Simjiana. Zbudowany jako "Bankograph" w 1960r. Niezależnie wynalazcą bankomatu był też John Shepherd-Barron, a pierwsze urządzenie pojawiło się w 1967r. W polsce pierwszy bankomat pojawił się w roku 1987r.

Cel projektu

Celem projektu było udoskonalenie bankomatu działającego w trybie offline. Do bankomatu połączonego z bazą danych, zapewnia to możliwość zmiany funduszu przypisanego do danego klienta o kwotę wypłacaną lub wpłaconą. W bazie danych każdy klient opisywany jest za pomocą dwóch wartości, unikalnego numeru klienta oraz czterocyfrowego kodu pin, który jest znany tylko przez właściciela konta. W bankomacie jest możliwość zmiany wyłącznie salda. Największym wyzwaniem jakie towarzyszyło nam podczas pracy nad projektem było połączenie bazy danych z systemem LabVIEW, w którym została napisana poprzednia wersja maszyny ATM.

Przegląd systemu

Projekt został stworzony w środowisku LabVIEW 2020, z wykorzystaniem przykładu z egzaminu CLD pod tytułem "ATM Machine" zbudowany na szablonie JKI State Machine, który został dostarczony jako baza projektu. Projekt zbudowany został w całości w warstwie systemowej LabVIEW, bez użycia dodatkowego sprzętu lub narzędzi.

Software

Windows 10 - to seria systemów operacyjnych opracowanych przez firmę Microsoft i wydanych jako część rodziny systemów operacyjnych Windows NT.

LabVIEW - to oprogramowanie dla systemów do tworzenia aplikacji wymagających pomiarów i kontroli z szybkim dostępem do sprzętu i danych. LabVIEW oferuje graficzne podejście do programowania, które pomaga wizualizować każdy aspekt aplikacji, w tym konfigurację sprzętu, dane pomiarowe oraz testowanie.

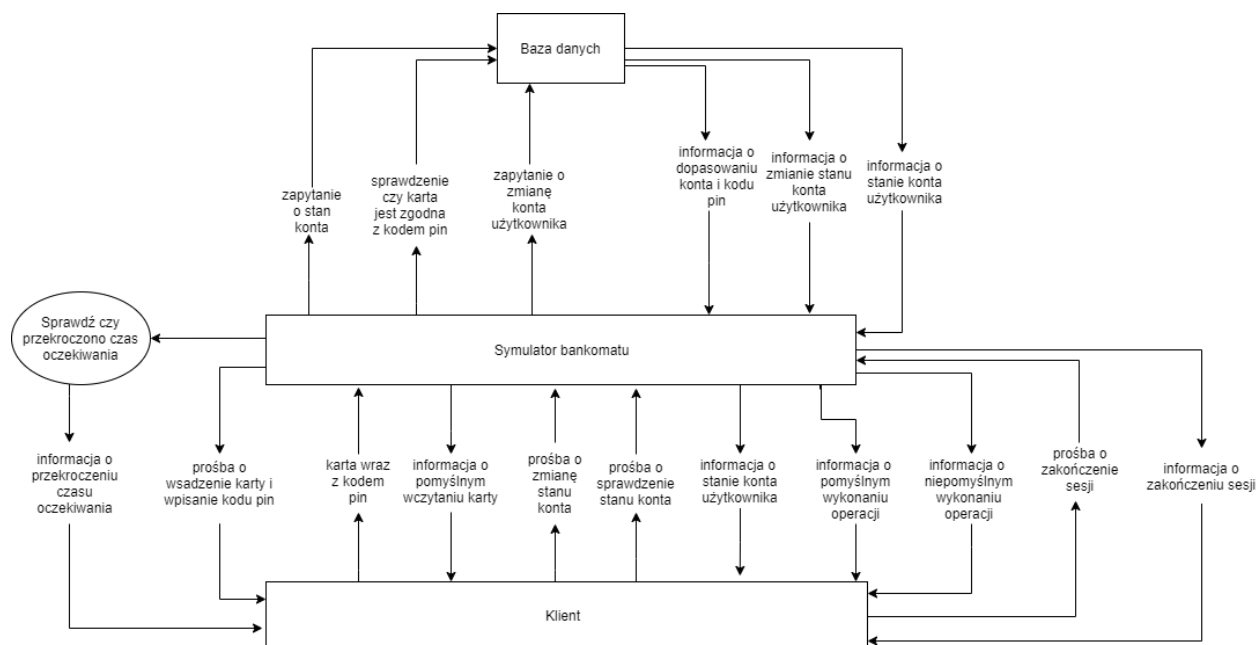
JKI State Machine - Maszyna stanów JKI dla LabVIEW jest to łatwy w użyciu, ale jednocześnie niezawodny szablon maszyny stanów. Obsługuje stany w naszym programie.

SQLite Library - biblioteka procesów, która implementuje niezależny, bezserwerowy silnik bazy danych SQL bez konfiguracji. W naszym programie używamy tej biblioteki do obsługi bazy danych z poziomu aplikacji w LabVIEW.

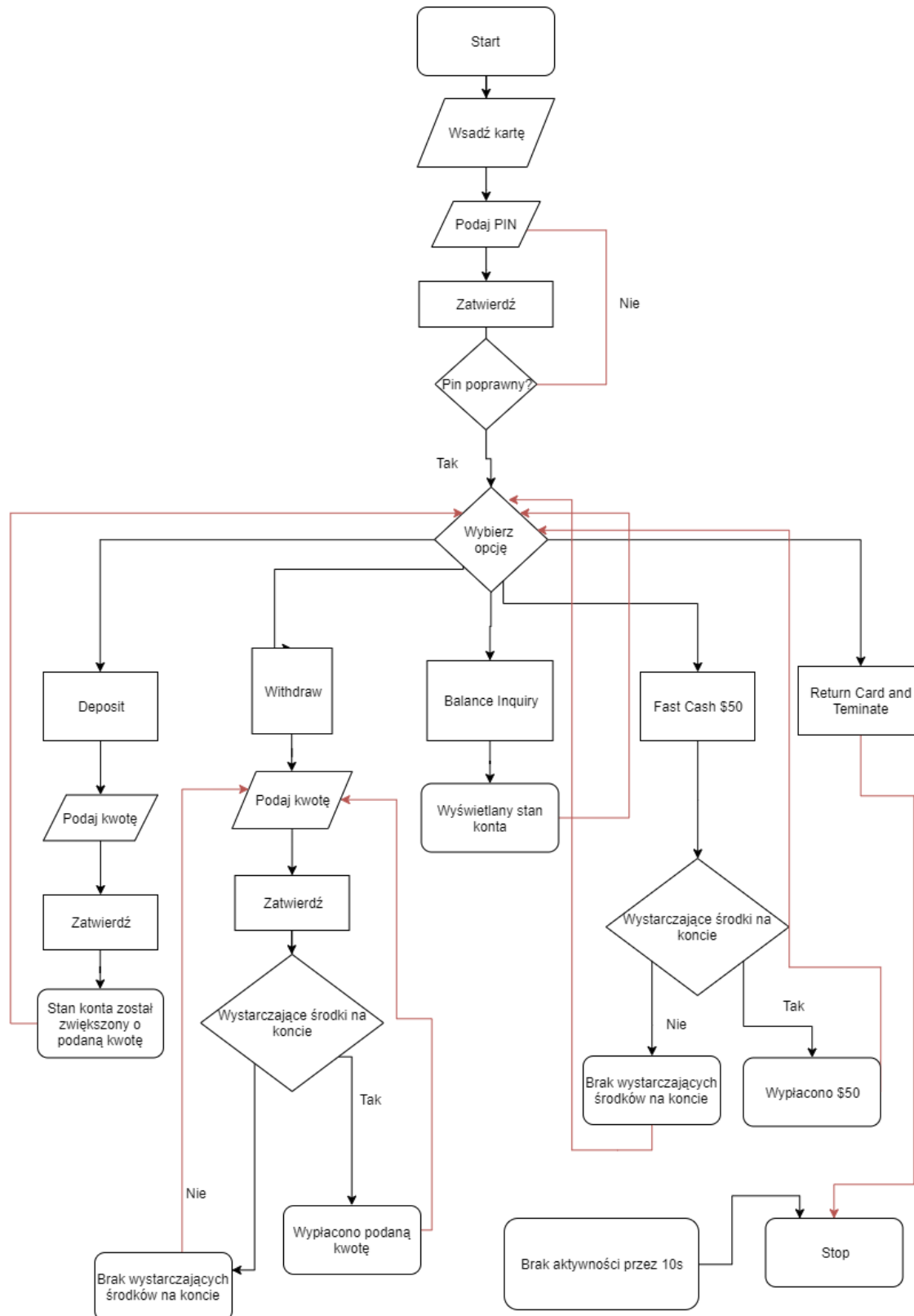
SQLiteStudio - program do obsługi baz danych. Używaliśmy tego programu do sprawdzania poprawności wykonywanych poleceń z biblioteki SQLite w programie LabVIEW.

Pozyskiwanie danych

Schemat przepływu danych



Schemat blokowy



Prezentacja

Interfejs użytkownika

Zgodnie z założeniem, chcieliśmy aby interfejs naszej aplikacji był przyjazny dla potencjalnego użytkownika, w związku z tym stworzyliśmy go tak, aby był przestronny i prosty w użyciu. Znajdziemy tutaj przycisk symulujący włożenie naszej karty bankowej, a także pole do wpisania naszego numeru PIN. Do potwierdzenia PIN'u służy przycisk ENTER indeksowany jako 'E'. Po prawidłowym wprowadzeniu danych, na środku naszego interfejsu w polu ATM Messages, będą pojawiały się informacje, które pomagają użytkownikowi w podejmowaniu odpowiednich akcji. Z lewej strony znajdują się trzy przyciski indeksowane jako 'Left Buttons', a okienka oznaczone jako 'Left Menu' są przypisane do sąsiadującego przycisku i informują nas o rezultacie jego naciśnięcia. Sytuacja wygląda bardzo podobnie z prawej strony, z tą różnicą, że przyciski oznaczone jako 'Right Buttons' pełnią inną funkcję.

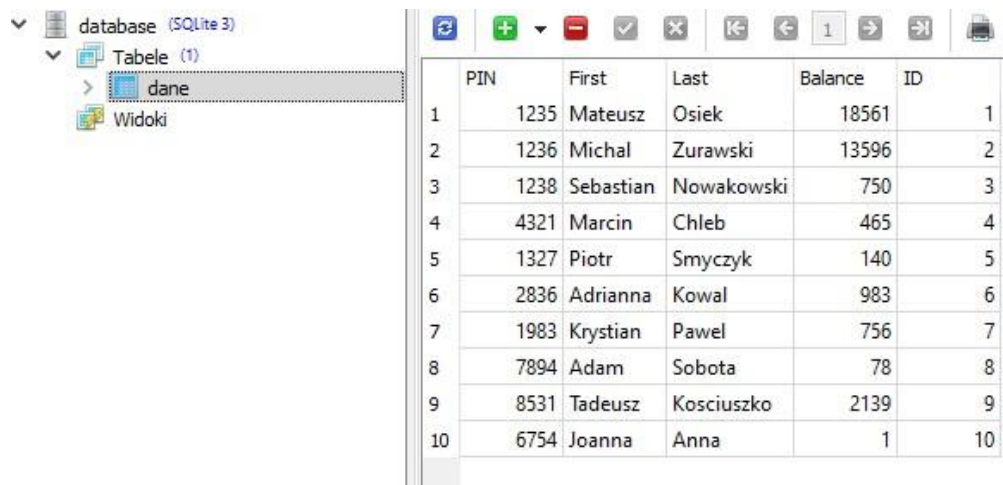


Pliki danych

Miejszem przechowywania wszystkich danych od użytkowników jest baza danych napisana w programie SQLiteStudio za pośrednictwem języka SQL. Umiejscowiona jest ona na dysku, jednak nie ma żadnych przeciwwskazań aby umieścić ją na serwerze. W celu weryfikacji klienta baza danych posiada osobną kolumnę z numerem identyfikacyjnym klienta oraz kodem pin do konta. Dopiero po poprawnym uzupełnieniu pinu użytkownik dostaje dostęp do salda konta, co za tym idzie również jego zmiany poprzez wpłatę lub wypłatę. Jako że podchodzimy indywidualnie do każdego klienta, w naszej bazie danych widnieje również jego imię oraz nazwisko.

Program obsługuje bazę danych za pośrednictwem wysyłania poleceń do SubVi "Manage Accounts.vi" które odpowiada wartościami takimi jak dane użytkownika albo ilość pieniędzy na koncie użytkownika.

Dostęp do bazy danych jest udzielany dopiero po uzyskaniu pinu od użytkownika i na tej podstawie jest obsługiwany wiersz w bazie danych zalogowanego użytkownika.



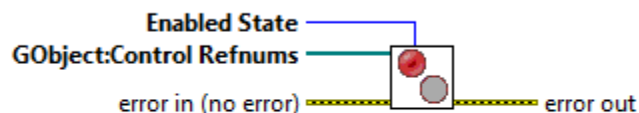
	PIN	First	Last	Balance	ID
1	1235	Mateusz	Osiek	18561	1
2	1236	Michał	Zurawski	13596	2
3	1238	Sebastian	Nowakowski	750	3
4	4321	Marcin	Chleb	465	4
5	1327	Piotr	Smyczyk	140	5
6	2836	Adrianna	Kowal	983	6
7	1983	Krzysztof	Paweł	756	7
8	7894	Adam	Sobota	78	8
9	8531	Tadeusz	Kościuszko	2139	9
10	6754	Joanna	Anna	1	10

Raporty z systemu

System komunikuje się z użytkownikiem za pomocą wiadomości na panelu czołowym, odbywa się to za pośrednictwem wywołania SubVI "Get ATM Message.vi" która zajmuje się obsługą wyświetlania informacji użytkownikowi.

Podprogramy SubVI

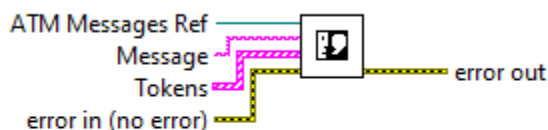
Batch Enable Controls.vi



This function enables, disables, and disables-and-greys-out a set of controls and indicators en masse. Its input is an array of "GObject:Control" refnums.

Ta funkcja zajmuje się włączaniem, wyłączaniem oraz dezaktywacją elementów sterujących i wskaźników. Jego wejście jest tablicą elementów referencji "GObject:Control".

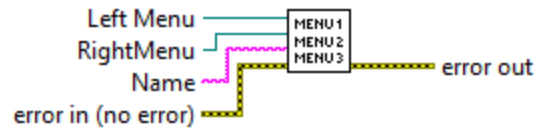
Get ATM Message.vi



This VI accesses and parses a message string for user display. It provides encapsulation of the core defined messages, so the message set can be updated/modified without having to change this function or any of its callers.

Ta funkcja uzyskuje dostęp oraz wyświetla ciąg znaków do wyświetlenia użytkownikowi. Dostarcza ona również zdefiniowane wiadomości aby można było zmieniać w nich fragmenty co pozwala na wywoływanie tej funkcji bez jej zmian.

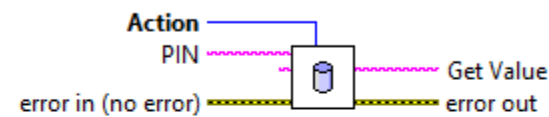
Get Menus.vi



This VI provides access to the defined set of menus, returning whichever is requested by the caller. It provides encapsulation of the menu list, so the list can be updated/modified without having to change this function or its callers.

To VI pozwala na dostęp do zdefiniowanego zestawu kart, zwracając te które są zażądane przez wywołanie. Jej konstrukcja pozwala na zmianę zawartości zwróconych bez konieczności zmiany funkcji bądź wywołania.

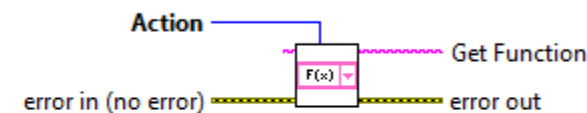
Manage Accounts.vi



This VI manages the account records and the file that stores them on disk. It provides encapsulation for the record data so that the file structure or data set can be changed, and this functions callers will not have to be updated to accommodate those changes.

Ta funkcja zarządza danymi użytkowników i obsługuje bazę danych. Wywołania odnoszące się do danych użytkowników są wywoływane tak aby można było wywołać pojedynczych użytkowników bez zmieniania funkcji. Pozwala na obsługę bazy danych bez potrzeby łączenia się nią poza tą funkcją.

Manage Current Function.vi



This FGV stores the last-selected function from the menu controls. It is a simple memory for the application.

Jest to FGV (tłumaczenie w słowniku) która zachowuje ostatnią funkcję z menu. Jest to prosta pamięć dla aplikacji.

Withdraw.vi



This VI handles withdraw money from user account. Its input is amount of money we want to withdraw, it output proper message as a string.

To VI obsługuje wyciąg pieniędzy z konta użytkownika. Jako dane wejściowe przyjmuje ilość pieniędzy które chcemy wybrać, a na wyjściu wypisuje ciąg znaków opisujący wynik operacji.

Sposób testowania

Po odpowiednim zakodowaniu bazy danych oraz połączeniem jej z aplikacją bankomatu napisaną w LabVIEW, przeszliśmy do fazy testów. Zaczęliśmy od jednego użytkownika aby sprawdzić czy bez podawania kodu pin będzie możliwość poznania stanu konta oraz jego zmiany. Co utwierdziło nas w przekonaniu że zabezpieczenie salda użytkownika jest na na należyłym poziomie. Następną fazą testów było zwiększenie ilości klientów do 10. Niestety jako że mamy dostęp jedynie do symulacji w programie, byliśmy zmuszeni do stworzenia 10 innych haseł, gdyż weryfikacja numeru klienta odbywać powinna się przez kartę wsadzoną do bankomatu.

Wnioski

Podsumowując, po licznych spotkaniach oraz wspólnych konsultacjach w grupie, udało nam się zrealizować postawione zadanie - zaprogramowaliśmy symulator bankomatu. Podczas projektu rozwinęliśmy wiedzę nie tylko z zakresu samego LabView, ale także z innych dziedzin jak np. obsługa bazy danych, lub też jej połączenie z naszym oprogramowaniem. Reasumując spełniliśmy wszystkie wymagania dotyczące naszego projektu.

Dodatek A: Słownik

ATM - (z ang. automated teller machine) bankomat

CLD - (z ang. Certified LabVIEW Developer) Certyfikowany Deweloper LabVIEW

JKI - firma tworząca rozwiązania w LabVIEW

PIN - (z ang. Personal Identification Number) osobisty numer identyfikacyjny

Offline - poza siecią

State Machine - maszyna stanu

SQL - (z ang. Structured Query Language) Strukturalny język zapytań

UI - (z ang. user interface) interfejs użytkownika

FGV - (z ang. functional global variable) funkcjonalna zmienna globalna

Deposit - wpłacić

Withdraw - wypłacić

Balance - stan konta

Button - przycisk

Insert card - włóż kartę

Dodatek B: Stany

Default - obsługuje niezdefiniowane stany

Initialize Core Data - determinuje zachowanie panelu podczas wyjścia z programu

Error Handler - obsługuje błędy

Exit - wyłącza program

UI: Initialize - inicjalizuje UI.

UI: Front Panel State - otwiera lub zamyka panel czołowy

Macro: Initialize - dodaje stany inicjalizujące do kolejki stanów

Macro: Exit - wywoływane podczas zamykania programu, sprawdza czy panel czołowy powinien zostać zamknięty, jeśli tak to dodaje odpowiedni stan zamykający do kolejki stanów

Data: Initialize - inicjalizuje wszystkie dane do rejestru przesuwne

Data: Cleanup - czyści obecną referencję do VI

Action: Input Clear - czyści User Input

Action: Terminate - wyświetla komunikat terminacji programu w polu Message ATM, wyłącza User Input i Card Simulator zostaje ustawione na fałsz, następnie dopisuje do kolejki stanów
Action: Shutdown

Action: Balance - wyświetla aktualny stan konta użytkownika w polu Message ATM

Action: Shutdown - zwraca wszystkie kontrolki i wywołuje makro Exit

Action: Withdraw - wypłaca środki z konta użytkownika, wypisuje odpowiedni komunikat w polu Message ATM następnie dodaje stan Action: Input Clear do kolejki stanów

Action: Deposit - wpłaca środki na konto użytkownika

Action: Insert Card - wpłaca środki na konto użytkownika, wypisuje odpowiedni komunikat w polu Message ATM następnie dodaje stan Action: Input Clear do kolejki stanów

Action: Validate - sprawdza czy użytkownik o podanym pinie istnieje w bazie danych, jeśli tak tak następuje zalogowanie na konto użytkownika, jeśli nie, wypisuje odpowiedni komunikat w polu Message ATM by ponownie wpisać PIN

Buttons Events: Right Buttons - obsługuje przyciśnięcie prawych przycisków menu

Buttons Events: Left Buttons - obsługuje przyciśnięcie lewych przycisków menu

Event Structure, Idle - stan bezczynności, podczas niego są wykonywane zdarzenia (wywoływany jest kiedy kolejka stanów jest pusta)