Wymagania niefunkcjonalne:

* Funkcjonowanie w systemie Windows 7 i nowszych
* Umożliwienie obsługi sklepu użytkownikom niezwiązanym z informatyką

Interfejs tekstowy

Jednym z głównych założeń projektu było zrezygnowanie z graficznego interfejsu użytkownika. Należało, więc stworzyć interfejs, który nie tylko pozwoli na sprawną nawigację w sklepie, ale także sprawi, że klienci będą tutaj wracać z uśmiechem na twarzy. W tym celu stworzony został interfejs tekstowy, którego uzupełnieniem są komunikaty głosowe.

Po określeniu wymagań funkcjonalnych stworzono bazę komend. Nawigacja odbywa się poprzez wpisywanie ich w aplikacji klienckiej. Dla użytkowników niezaznajomionych z aplikacją przygotowano pomoc, w której znajdują się wszystkie polecenia wraz z ich opisem oraz przykładem użycia.

Świadomym krokiem było nie stworzenie funkcjonalności poprawiania składni. Spowodowane to było nie tylko brakiem czasu i środków, ale przede wszystkim chęcią nakłonienia klientów do skrupulatnego i poprawnego wpisywania komend, co z całą pewnością przełoży się na sukcesy w życiu zawodowym i prywatnym, dzięki czemu zadowolenie z aplikacji sklepowej wzrośnie jeszcze bardziej.

Brak graficznego interfejsu nie oznacza istnienia jedynie teksu. Po połączeniu z serwerem wyświetlane zostaje niezwykłe logo sklepu, Koń i Samochód wykonane w technologii ASCII ART. Raz ujrzane nie znika już z pamięci podobnie jak piersi 20 letniej, nagiej dziewczyny, która lipcowym porankiem przechadza się po zroszonej łące a obok której idzie o dwa lata młodsza siostra, jednakowoż naga a której wzrok zdaje się mówić „Nas jest dwie a Ty jeden, ale to nic nie szkodzi”. Podobnie komunikaty głosowe sprawiają, że zakupy nie są jedynie monotonną czynnością a rytuałem, który z powodzeniem może zastąpić wizytę w filharmonii czy operze.

ComandParser

CommandParser to narzędzie odpowiedzialne za odczytywanie komend wpisywanych przez użytkownika. Jego główną składową jest klasa CommandParser. Jej elementem jest comandsDictionary, którego kluczami są obsługiwane komendy a wartościami delegaty do odpowiednich metod w klasie StoreSystem. Dzięki takiemu sposobowi możliwe jest łatwe rozszerzanie funkcjonalności systemu. Każda komenda wpisywana przez użytkownika składa się identyfikatora oraz opcjonalnych parametrów. Jej weryfikacja przebiega wieloetapowo. Najpierw sprawdzane jest czy komenda nie jest pusta. Następnie określa się czy komenda jest obsługiwana przez system oraz czy została podana odpowiednia liczba parametrów. Poprawność zawartości parametrów sprawdzania jest dopiero w klasie SystemStore. Commad Parser odpowiedzialny jest też za wyświetlanie komunikatów o nawiązywaniu połączenia oraz o udanym bądź nieudanym połączeniu z serwerem a także wiadomość powitalną.

StoreSystem

Klasa StoreSystem zawiera w sobie całą logikę biznesową sklepu. Odpowiedzialna jest za takie działania jak weryfikacja, autoryzacja i zarządzanie użytkownikami, obsługę koszyka, finalizowanie zamówień, wyszukiwanie i wyświetlanie informacji o produktach w sklepie a także dodawanie nowych i edytowanie już istniejących. W tej klasie dokonuje się też zmian w bazie danych. Ponadto wywołuje ona metody odpowiedzialne za zarówno tradycyjne komunikaty tekstowe jak i zupełnie unikatowe w aplikacjach sklepowych komunikaty głosowe.

ComunikatesFactory

Comunikates Factory to klasa odpowiedzialna za przekazanie do wyświetlenia odpowiedniego komunikatu. Składa się z pola typu dictionary communicatesDictionary oraz metody GetCommunicate. Kluczami w słowniku są rodzaje komunikatów np. LoginAccepted wyświetlany po udanym logowaniu. Typy komunikatów przechowywane są za pomocą typu wyliczeniowego ComunicatesKinds. Wartościami słownika są zmienne typu string, które zostają wyświetlone użytkownikowi. W tym przykładzie jest to „Zalogowano pomyślnie.”. Metoda GetCommunicate zwraca „wyłuskany” ze słownika komunikat o kluczu otrzymanym poprzez parametr.