|  |  |  |  |
| --- | --- | --- | --- |
| **Wydział Matematyki Stosowanej**  **Politechnika Śląska** | | | |
| **Kierunek:** | | | Informatyka |
| **Stopień studiów:** | | | pierwszy |
| **Semestr:** | | | IV |
| **Przedmiot:** | | | Programowanie obiektowe i graficzne |
| **Prowadzący zajęcia:** | | | dr inż. Adam Zielonka |
| **Dokumentacja techniczna**  **realizacji projektu zespołowego** | | | |
| **Zespół projektowy** | | | |
| **Lider zespołu:** | | Arkadiusz Duliban arkadiusz.duliban@gmail.com | |
| **Wykonawca 1:** | | Dominika Bartek | |
| **Wykonawca 2:** | | Patrycja Chmielewska | |
| **Projekt** | | | |
| **Tytuł projektu:** | Program do fakturowania "Fakturka" | | |
| **Repozytorium:** | <https://github.com/dulibanarkadiusz/faktury> | | |
| **Cel projektu:** |  | | |
| **Opis projektu:** |  | | |
| **Użyte narzędzia** | | | |
| Nazwa: | Visual Studio 2013 Professional | | |
| Licencja: | Licencja DreamsPark | | |
| Nazwa: | gitHub, MySQL | | |
| Licencja | - , GPL | | |

**Historia zmian dokumentu**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LP | Opis zmiany | Typ | Data zmiany | Autor | Uwagi |
| 1 | Przygotowanie szablonu dokumentacji technicznej | E | 24.05.2016 | Adam Zielonka | Proszę nie zmieniać wielkości czcionek  w dokumencie |
| 2 | Uzupełnienie dokumentacji o dane oraz opis projektu. | D | 26.05.2016 | Arkadiusz Duliban |  |
| 3 | Sprecyzowanie wymagań projektu; sporządzenie modelu danych | D | 28.05.2016 | Arkadiusz Duliban |  |
| 4. | Opis klasy Article | D | 31.05.2016 | Dominika Bartek |  |
| 5. |  |  |  |  |  |
| 6. | Opis klasy EditorXML | D | 13.06.2016 | Dominika Bartek |  |
| 7. | Opis klasy OurCompany | D | 18.06.2016 | Dominika Bartek |  |

Typ: E - edycja dokumentu, D - dodanie nowych treści, M - korekta merytoryczna

**1. Opis projektu**

*Program skierowany jest do małych i średnich firm lub działalności jednoosobowych, u których istnieje konieczność generowania faktur VAT. Aplikacja umożliwia generowanie nowych faktur oraz ich automatyczne archiwizowanie, umożliwiając późniejsze przeglądanie, wyszukiwanie czy księgowanie wpłat. Tworzenie faktury jest niezwykle łatwe dzięki przestępnemu interfejsowi graficznemu: wystarczy jednorazowo wprowadzić dane o artykułach (lub usługach) do bazy danych, by móc niskim nakładem pracy dodawać towary do faktury. Raz zapisane dane o kontrahentach również mogą być odtwarzane i edytowane w przyszłości.   
Gotowa faktura VAT może zostać wygenerowania w formie PDF i automatycznie wysyłana jako załącznik na adres e-mail kontrahenta.   
Istotną aspektem programu jest jego bezpieczeństwo: każdy pracownik ma możliwość pracy na swoim indywidualnym koncie użytkownika z prawami dostępu określonymi przez administratora systemu.   
W ten sposób - przykładowo - użytkownik odpowiedzialny za dodawanie artykułów nie będzie w stanie generować faktur (a także ich wyświetlać), a osoba wystawiająca faktury nie będzie miała możliwości edytowania cen produktów. Ponadto wszelkie zdarzenia dokonywanie na bazie danych (edycja danych czy ich utworzenie) są rejestrowane w formie logów.*

**2. Wymagania**

Członkowie zespołu prowadząc wspólnie analizę zebranych wymagań dotyczących realizacji projektu pogrupowali je w następujący sposób:

2.1 Wymagania funkcjonalne

* formularz logowania umożliwiający dostęp do bazy; administrator ma mieć możliwość dodawania użytkowników i przypisywania im uprawnień
* możliwość dodawania, edytowania i usuwania danych dotyczących towarów i kontrahentów
* krotki artykułów i kontrahentów umożliwiają skonstruowanie faktury o dynamicznie generowanym numerze. Artykuł występujący na fakturze zmienia się w "pozycja faktury" i również może być edytowalny po dodaniu.
* opcja eksportu obiektu faktury do pliku PDF
* możliwość wysłania faktury PDF jako wiadomość e-mail z poziomu aplikacji
* możliwość łatwego i szybkiego wyszukiwania istniejących informacji z poziomu aplikacji.
* system logów rejestrujący wszystkie wykonywane akcje; możliwe do przeglądania przez administratora

2.2 Wymagania niefunkcjonalne

* wielodostępowość bazy danych
* aplikacja ma być łatwa w użytkowaniu dla osób nieznających MySQL
* zapewnienie bezpieczeństwa dla danych przechowywanych w bazie i zarządzania nimi przez więcej niż 1 użytkownika na różnych stanowiskach
* ochrona przed błędami użytkownika - w szczególności nadzorowanie wprowadzania danych we właściwym formacie, itp.
* respektowanie podstawowych przepisów prawa w zakresie wystawiania faktur VAT (terminy, sposób budowy dokumentu standardowej faktury VAT i in.)

**3 Projekt interfejsu graficznego**

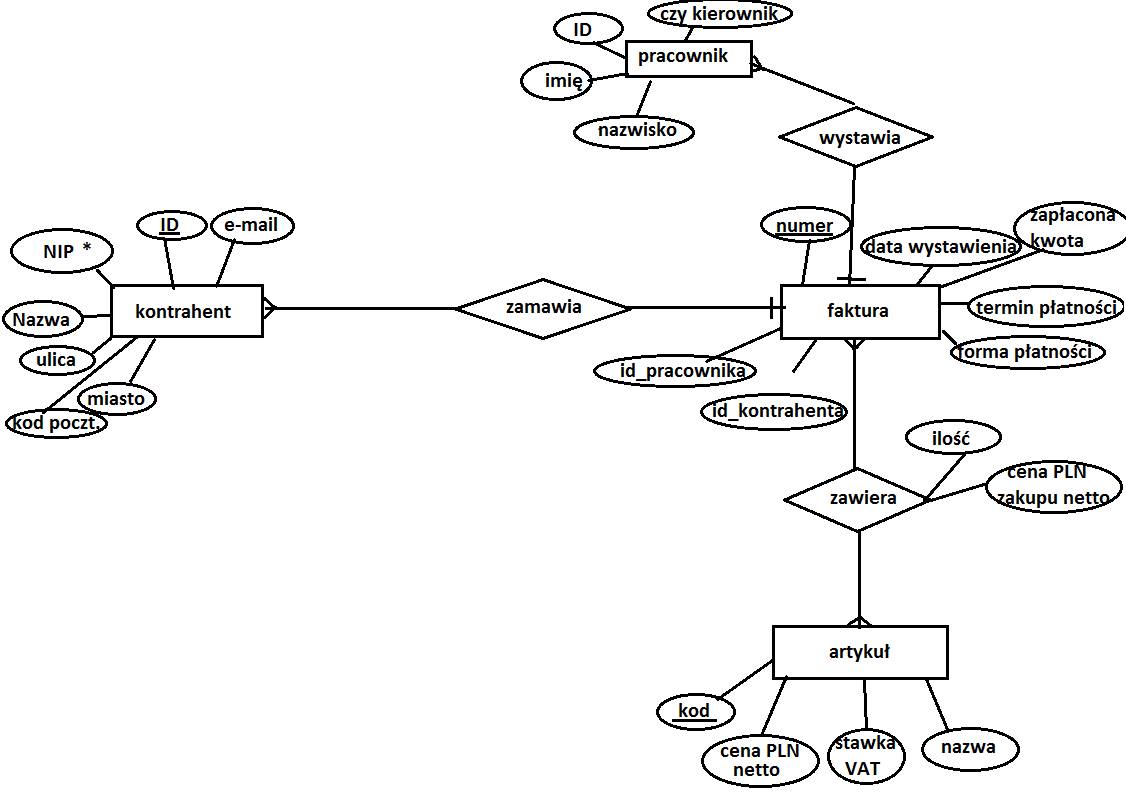
*Szkice wyglądu interfejsu publicznego aplikacji w odniesieniu do zebranych wymagań*

**4. Model obiektowy**

*Schemat UML klas*

**5. Model danych (w przypadku aplikacji bazodanowej)**

*(tymczasowy)*

**

**6. Implementacja**

*Dla każdej z klas opisujemy dokładnie jej interfejs publiczny*

|  |  |
| --- | --- |
| **Klasa** | NazwaKlasy |
| **Właściwości** | |
| typ NazwaWlasciwosci | *Co reprezentuje i jak działa* |
|  |  |
| Metody publiczne | |
| typ NazwaMetody | *Jak działa i co zwraca* |

|  |  |
| --- | --- |
| **Klasa** | Article |
| **Właściwości** | |
| string Code | *Reprezentuje kod artykułu. Zwraca napis reprezentujący kod.* |
| string Name | *Reprezentuje nazwę artykułu. Zwraca napis reprezentujący nazwę.* |
| decimal PriceNetto | **Reprezentuje** *cenę netto artykułu. Zwraca wartość ceny netto.* |
| decimal PriceBrutto | **Re**prezentuje cenę brutto artykułu. Zwraca wartość ceny brutto. |
| decimal VATvalue | Reprezentuje stawkę vat. Zwraca wartość wartości vat wyrażoną w %. |
| string UnitMeasure | Reprezentuje zdefiniowaną jednostkę miary danego artykułu. Zwraca nazwę jednostki miary. |
| Metody publiczne | |
| string GenerateQueryUpdateArticles() | *Tworzy zapytanie do bazy aktualizujące informacje o artykule. Zwraca treść zapytania.* |
| string GenerateQueryInsertArticles() | *Tworzy zapytanie do bazy dodające nowy artykuł. Zwraca treść zapytania.* |
| GenerateQueryDropArticles() | *Tworzy zapytanie do bazy usuwające dany artykuł. Zwraca treść zapytania.* |

|  |  |
| --- | --- |
| **Klasa** | EditorXML |
| **Właściwości** | |
| typ NazwaWlasciwosci | *Co reprezentuje i jak działa* |
|  |  |
| Metody publiczne | |
| void AddToXML(string name, string contents) | *Generuje element dokumentu XML o nazwie name i wartość elementu contents, lub zastępuje istniejący element o nazwie name nadając mu nową wartość contents. Po czym dopisuje element do pliku. Nie zwraca nic.* |
| String FindInXML(string name) | *Szuka w pliku XML elementu o nazwie name i zwraca jego wartość.* |

|  |  |
| --- | --- |
| **Klasa** | OurCompany |
| **Właściwości** | |
| string Regon | *Reprezentuje numer REGON. Zwraca ciąg znaków reprezentujący REGON.* |
| string CompanyName | *Reprezentuje nazwę firmy. Zwraca nazwę firmy.* |
| string City | *Reprezentuje miasto w którym znajduje się firma. Zwraca nazwę miasta.* |
| string PlaceAddres | *Reprezentuje ulicę na której znajduje się firma. Zwraca nazwę ulicy.* |
| string NIP | *Reprezentuje numer NIP firmy. Zwraca ciąg znaków reprezentujący NIP.* |
| string Code | *Reprezentuje kod pocztowy. Zwraca ciąg znaków reprezentujący kod pocztowy.* |
| string BankAccount1 | *Reprezentuje główny numer konta bankowego firmy. Zwraca ciąg znaków reprezentujący numer konta bankowego.* |
| Metody publiczne | |
| Void EditOurCompany() | *Zapisuje edytowane dane firmy do pliku XML. Nie zwraca nic.* |

**7\* Testy**

**8. Instrukcja użytkownika**