**Wrocławska Wyższa Szkoła Informatyki Stosowanej**

Wydział Informatyki

Krzysztof Hadaś

Nr albumu: 5282

Integracja aplikacji bazodanowej w chmurze MS Azure

Praca inżynierska

Kierunek: Informatyka

Specjalność: Programowanie

Praca wykonana pod kierunkiem:

Mgr Stanisław Lota

Wrocław 2020

Spis treści

1. Wstęp
2. Cel i założenia projektu
3. Zakres pracy

Wstęp

Przedmiotem niniejszej pracy jest propozycja platformy wspomagającej monitorowanie i zarządzenie kontrolą jakości części samochodowych opartej na architekturze bazodanowej klient-serwer. Funkcjonalność i rozwiązania które w pracy będą omawiane będą oparte na realnej potrzebie firmy w której miałem przyjemność pracować na stanowisku kierowniczym.

Firma świadczyła usługi z zakresu kontroli jakości i napraw dla swojego klienta z branży automotive. Codzienne tworzenie raportów z przeprowadzanych dzień wcześniej sortowań stworzyło problem natury logistycznej. Pojawił się problem z przechowywaniem i archiwizowaniem raportów z każdego dnia tak aby w miarę potrzeby można było łatwo i szybko otrzymać konkretny raport z danego dnia. Firma dla każdego sortowania tworzyła pojedynczy raport w formacie arkusza kalkulacyjny excel, dzienny zbiór raportów archiwizowała programem 7zip a zarchiwizowaną paczkę przechowywała na zdalnym serwerze.

Cel pracy

Celem pracy będzie stworzenie od podstaw aplikacji komputerowej umożliwiającej nadzór procesu sortowania części samochodowych. Pomysł narodził się w mojej głowy podczas pracy w firmie zajmującą się kontrolą jakość. Pewnego dnia zauważyłem, że za dużo czasu tracę na analizowanie statystyk danego sortowania, porozmawiałem o tym z moim przełożonym i poszukałem ewentualnych rozwiązań innych firm. Niestety żadne rozwiązanie które znalazłem nie spełniło moich oczekiwań. Dlatego znając problem i strukturę firmy zdecydowałem się na stworzenie swojego rozwiązania.

Zawartość pracy

Lorem ipsum itp.

Systemy kontroli wersji

Nieodłącznym elementem tworzenia aplikacji komputerowej jest wprowadzanie zmian w postaci nowego kodu lub jego modyfikacji. Podczas długotrwałej pracy nad projektem może dojść do sytuacji w której stwierdzimy, że nie rozwijamy dalej danej funkcjonalnośći i należy prace cofnąć do pewnego momentu. Może też dojść do sytuacji w której programista wprowadzi zmiany w niewłaściwym pliku, usunie ważną funkcję lub jego zadanie zostanie przekazane innemu programiście. Aby ułatwić programistom pracę z kodem w rozbudowanych aplikacjach stosuje się systemy kontroli wersji (ang. version/revision control system).

# „Systemem Kontroli Wersji jest repozytorium plików, zazwyczaj plików kodu źródłowego programów komputerowych, z kontrolą dostępu. Każda zmiana wykonana na kodzie źródłowym jest śledzona razem z tym kto zrobił tą zmianę, dlaczego ją zrobił oraz adnotacjami do naprawionych problemów albo wprowadzonych ulepszeń. Systemy kontroli wersji są niezbędne dla wszelkich form współpracy nad rozwojem oprogramowania.”

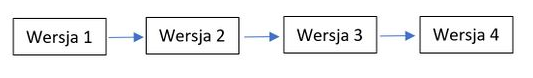
Dzięki zastosowaniu w projekcie systemu kontroli wersji projekt może być rozwijany w jednym czasie przez wielu programistów i w wielu miejscach aplikacji. Przy każdej modyfikacji dostajemy informację o zmodyfikowanym kodzie, jego autorze a także możliwość wprowadzenia tych zmian do głównej aplikacji i opcję przywrócenia stanu sprzed dodania poprawki.

Główne zalety korzystania z systemu kontroli wersji:

* Śledzenie wersji – wraz z rozbudową kodu mamy możliwość śledzenia dodawanego kodu z informacjami o autorze, dacie dodania, uwagami. Mamy możliwość obserwowania co się zmieniło na przestrzeni rozwijania całego projektu.
* Koordynowanie pracy zespołów – możemy zlecić wielu zespołom pracę na tej samej wersji ale lokalnych kopiach kodu nie powodując konfliktów w czasie tworzenia kodu.
* Kontrola zawartości oraz monitorowanie – możemy poznać autora danej części kodu oraz liczbę osób rozwijających aplikację.

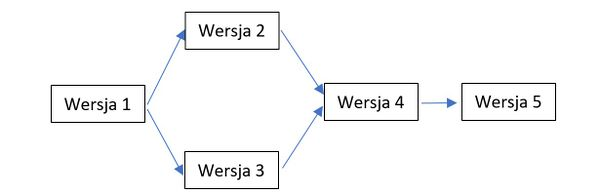
Schemat działania systemu kontroli wersji

Mechanizm wykorzystywany przez systemy kontroli wersji opiera się na bazach zmian zwanych repozytoriami (ang. Repository) w których zapisywane są informację o zmianach w śledzonych plikach oraz na kopiach roboczych na których programista wprowadza zmiany. Modyfikowanie kodu na kopii roboczej umożliwia programiście dowolną modyfikacje kodu. Gdy osoba monitorująca rozwój kodu jest zadowolona z wprowadzonych zmian następuje proces łączenia zmian z głównym repozytorium zwany mergowaniem.



## Rysunek 1 Schemat historii bazy w wersji liniowej.

Powyższy schemat przedstawia najprostszy rozwój oprogramowania w wersji liniowej, każda zmiana następuje po następnej. Widzimy jedną ciągłą linię rozwoju kodu która wyglądem przypomina oś czasu w której widzimy jakie zmian zostały wprowadzone.



## Rysunek 2 Schemat historii bazy w wersji rozgałęzionej.

W drugim schemacie programiści pracują jednocześnie na lokalnych kopiach głównego repozytorium tworząc tzw. rozgałęzienia (ang. Branches). Utworzone rozgałęzienie jest alternatywną wersją zmian w głównym repozytorium nazywaną gałęzią. Rozgałęzienie wykorzystujemy

Coś jeszcze i podział na jakie dzielimy

Coś konkretnego o gicie

Zakładanie repozytorium na bitbackecie

Podsumowanie

Spis rysunków

Bibliografia

Linki

Załączniki