



**WYŻSZA SZKOŁA
INFORMATYKI i ZARZĄDZANIA**
z siedzibą w Rzeszowie

KOLEGIUM INFORMATYKI STOSOWANEJ

Kierunek: INFORMATYKA

Specjalność: Wpisujemy po jej wybraniu

Jan Nowak
Nr albumu studenta w9699

System do raportowania przewinień w biurze

Prowadzący: mgr inż. Ewa Żesławska

Praca projektowa programowanie obiektowe C#

Rzeszów 2023

Spis treści

Wstęp	4
1 Opis założeń projektu	5
1.1 Cele projektu	5
1.2 Wymagania funkcjonalne i нефункционалне	5
1.2.1 Przkłady	8
2 Opis struktury projektu	10
3 Harmonogram realizacji projektu	11
4 Prezentacja warstwy użytkowej projektu	12
5 Podsumowanie	13
Bibliografia	14
Spis rysunków	15
Spis tablic	16

Wstęp

Przykład wstępu:

Na co dzień wielu współpracowników w biurach spotyka się z pewnymi niedociągnięciami pojawiającymi się na tle drobnych obowiązków domowych, z którymi nie chcieliby spotykać się jeszcze dodatkowo w pracy, a które wynikają z zaniedbań na tym tle innych pracowników. Do takich małych niedociągnięć i zaniedbań obowiązków należy dla przykładu pozostawianie na dłuższy czas naczyń w zlewozmywaku, nie wynoszenie na czas śmieci, które zalegają w koszu przez dłuższy czas. Albo też nieumyty po używaniu ekspres do kawy, niewymienione po wykorzystaniu pojemniki z tuszem w drukarce. Problemem istotnym, a niebędącym bezpośrednio związanym z przedmiotem działalności firmy, jest dbanie o takie szczegóły, a które mogą wpływać na samopoczucie pracowników i atmosferę w firmie. Dlatego, wbrew pozorom, ważne jest odpowiednie zmotywowanie pracowników aby o takie, mimo że drobne obowiązki, dbali na równi z własnym domem, dzięki czemu każdemu współpracownikowi będzie się lepiej pracowało. I tutaj pojawia rozwiązanie naszego zespołu – system, który jednocześnie w jakiś sposób rozwiązuje powyższe zagadnienie, a z drugiej strony jest także formą urozmaicenia i rozrywki w pracy.

Przykłady wstępów do prac projektowych/naukowych/dyplomowych można znaleźć w artykułach/pracach naukowych/dyplomowych dostępnych w sieci (artykuły/prace naukowe) lub w repozytorium uczelni (biblioteka WSIZ Kielanrowa).

Rozdział 1

Opis założeń projektu

1.1 Cele projektu

Tutaj powinien zostać umieszczony tekst zawierający następujące informacje:

- Jaki jest cel projektu?
- Jaki jest problem, który będzie rozwiązywany oraz proszę wskazać podstawowe źródło problemu?
- Dlaczego ten problem jest ważny oraz jakie są dowody potwierdzające jego istnienie?
- Co jest niezbędne, aby problem został rozwiązany przez Zespół i dlaczego?
- W jaki sposób problem zostanie rozwiązany? Jak krok po kroku będzie przebiegała realizacja projektu? Co będzie wynikiem prac. (Wynikiem może być np.: aplikacja, system, sposób, metoda, program komputerowy).

1.2 Wymagania funkcjonalne i нефункционалне

Definicja:

Wymagania funkcjonalne

- opisują funkcje (czynności, operacje, usługi) wykonywane przez system
- Często stosowany sposób opisu wymagań – język naturalny
- Liczba wymagań funkcjonalnych może być bardzo duża; konieczne jest pewnego rodzaju uporządkowanie tych wymagań, które ułatwi pracę nad nimi (złożoność !)
- Opisują, jak funkcja powinna działać.
- Skupiają się na wyniku działania użytkownika.
- Definiuje wymagania użytkownika.
- Posiada funkcje uwzględnione w przypadkach użycia.
- Weryfikuje funkcjonalność system

Wymagania нефункционалне

- opisują ograniczenia, przy zachowaniu których system powinien realizować swe funkcje.
- Opisują, jakie właściwości sprawiają, że funkcja będzie działać.

- Skupiają się na uproszczeniu procesu i wykonania wyniku.
- Definiują oczekiwania i doświadczenia użytkownika działania użytkownika.
- Posiadają ograniczenia, które pomogą zredukować czas i koszty rozwoju.
- Weryfikują wydajność systemu.

Wymagania funkcjonalne przykłady:

Lista przykładów wymagań funkcjonalnych obejmuje każde zachowanie systemu IT, zmieniające się pod wpływem zastosowanej funkcji. Jeżeli wymagania funkcjonalne nie zostaną potwierdzone, system nie będzie działał.

- Reguły biznesowe.
- Poziomy autoryzacji.
- Śledzenie audytów.
- Interfejsy zewnętrzne.
- Funkcje administracyjne.
- Generowanie danych historycznych.
- Uwierzytelnianie użytkownika na żądanie.
- Logi serwera wszystkich istniejących danych.
- Generowanie raportów w określonym czasie.
- Definiowanie poziomów autoryzacji systemu.

Wymagania niefunkcjonalne przykłady:

Na liście wymagań niefunkcjonalnych znajdują się,

- Pojemność.
- Wydajność.
- Środowisko.
- Użyteczność.
- Skalowalność.
- Niezawodność.
- Odzyskiwalność.
- Bezpieczeństwo.
- Utrzymywalność.
- Interoperacyjność.
- Integralność danych.
- 2-poziomowe uwierzytelnianie.

Rozwinięcie wymagań niefunkcjonalnych:

- Aplikacja IT powinna mieć kolor tła wszystkich ekranów #fffaaa.
- Aplikacja IT powinna przestrzegać wymagań regulatora.
- Aplikacja IT powinien rejestrować każdą nieudaną próbę logowania;
- Użytkownicy powinni zmienić hasło po pierwszym udanym logowaniu.
- Dashboard powinien pojawić się w ciągu 3 sekund po zalogowaniu użytkownika.
- Aplikacja IT powinien być w stanie obsłużyć XYZ liczbę użytkowników, zapewniając płynne działanie.

Jak zdefiniować wymagania funkcjonalne?

Jeśli twoje podejście do rozwoju oprogramowania jest zwinne (Agile), prawdopodobnie zdefiniujesz wymagania w dokumencie. Dokument wymagań funkcjonalnych będzie zawierał historie użytkowników, przypadki użycia, a także następujące sekcje.

- Cel: Ta sekcja będzie zawierała całe tło, definicje i przegląd systemu;
- Zakres aplikacji, oczekiwania i zasady biznesowe;
- Wymagania dotyczące bazy danych, atrybuty systemu i wymagania funkcjonalne;
- Przypadki użycia, czyli opisywać, w jaki sposób użytkownik będzie wchodził w interakcję z systemem. Zdefiniuj rolę każdego aktora biorącego udział w interakcji;
- Napisz jasno cel wdrożenia systemu IT.
- Wspomnij o użytkownikach aplikacji, którzy szczegółowo opisz, jak krok po kroku będą się angażowali w tworzenie aplikacji.
- Opracuj klikalny prototyp aplikacji. To pomoże Ci reprezentować produkt w lepszy i przekonujący sposób dla interesariuszy. Możesz wybrać prototypy do wyrzucenia lub prototypy interaktywne dla swojego projektu.

Jak zdefiniować wymagania niefunkcjonalne?

Teraz nadchodzi część, w której definiujesz oczekiwania jakościowe aplikacji dedykowanej. Te atrybuty opisują sposoby, w jakie oczekujesz, że aplikacja będzie się zachowywała.

- Zdefiniuj oczekiwania dotyczące użyteczności produktu.
- Opisz, do jakich praw i regulacji aplikacja powinna spełniać.
- Zdefiniuj dostępność aplikacji, czyli czy będzie ona funkcjonować 24/7/365?
- Określ wydajność systemu IT dla różnych funkcjonalności. To znaczy, w jakim czasie użytkownik powinien zobaczyć listę, jak długo użytkownik będzie połączony z aplikacją w przypadku braku połączenia z internetem, itp.
- Zdefiniuj wymagania dotyczące bezpieczeństwa systemu IT.
- Użyj narzędzi do automatycznego testowania, aby upewnić się co do wydajności aplikacji dedykowanej.

1.2.1 Przykłady

Przykłady wymagań funkcjonalnych aplikacji webowej:

- Rejestracja i logowanie użytkowników.
- Bezpieczne uwierzytelnianie i autoryzacja użytkowników.
- Zarządzanie profilami użytkowników.
- Możliwość wyszukiwania treści w aplikacji.
- Funkcjonalność e-commerce, taka jak koszyk i proces kasowy.
- Treści generowane przez użytkowników, takie jak komentarze i oceny.
- Integracja z usługami stron trzecich, takimi jak media społecznościowe i bramki płatności.
- Dynamiczne aktualizacje treści i powiadomienia.
- Pulpit administracyjny do zarządzania aplikacją.

Przykłady wymagań niefunkcjonalnych aplikacji webowej:

- Użyteczność aplikacji i dostępność, takie jak responsywny design i dostępność klawiatury.
- Wydajność aplikacji i skalowalność, np. szybkie czasy ładowania i zdolność do obsługi dużej liczby użytkowników jednocześnie.
- Bezpieczeństwo aplikacji i prywatność, takie jak szyfrowanie wrażliwych danych i ochrona przed atakami.
- Niezawodność aplikacji i dostępność, np. kopie zapasowe i plany odzyskiwania danych po awarii.
- Zgodność aplikacji z wymogami prawnymi i regulacyjnymi, takimi jak GDPR i przepisy dotyczące dostępności.
- Interoperacyjność aplikacji, taka jak zgodność z różnymi przeglądarkami i systemami operacyjnymi.
- Utrzymanie i wsparcie aplikacji, takie jak łatwość aktualizacji i dokumentacja dla programistów.
- Efektywność kosztowa aplikacji, taka jak minimalizacja kosztów serwera i hostingu.

System zarządzania treścią (CMS) umożliwiający edycję i usuwanie treści.

Przykłady wymagań funkcjonalnych aplikacji webowej:

- Integracja aplikacji z zewnętrznymi API w celu wymiany danych lub rozszerzenia funkcjonalności.
- Funkcje optymalizacji aplikacji pod kątem wyszukiwarek (SEO) w celu poprawy widoczności w wyszukiwarkach internetowych.
- Obsługa wielu języków w aplikacji w celu dostosowania do użytkowników posługujących się różnymi językami.
- Narzędzia współpracy, takie jak czat w czasie rzeczywistym i udostępnianie plików dla zespołów.
- Narzędzia analizy danych do śledzenia zachowań użytkowników i wydajności aplikacji.

Projektowanie doświadczeń użytkownika (UX), takich jak łatwy w użyciu interfejs i przejrzysta nawigacja.

Przykłady wymagań niefunkcjonalnych aplikacji webowej:

- Optymalizacja mobilna aplikacji, np. responsywny design i podejście mobile-first.
- Integracja systemu, np. kompatybilność z dotychczasowymi systemami i narzędziami stron trzecich.
- Bezpieczeństwo aplikacji i prywatność danych, takie jak szyfrowanie danych, kopie zapasowe i kontrola dostępu.
- Zgodność aplikacji z normami branżowymi, takimi jak PCI-DSS dla aplikacji e-commerce.
- Wsparcie dla użytkowników aplikacji, takie jak help desk, podręczniki użytkownika i samouczki.
- Monitorowanie aplikacji i raportowanie, takie jak śledzenie błędów i metryki wydajności.

To tylko kilka przykładów funkcjonalnych i niefunkcjonalnych wymagań aplikacji internetowych. Konkretnie wymagania będą zależały od celu i charakteru aplikacji internetowej, a także potrzeb użytkowników i zainteresowanych stron.

Rozdział 2

Opis struktury projektu

W tym rozdziale mają pojawić się informacje odnośnie zaprojektowanej struktury oraz jej opis wraz z opisem technicznym. Należy umieścić informacje odnośnie wykorzystywanego języka, narzędzi oraz minimalnych wymagań sprzętowych. Opisać zarządzanie danymi oraz BD, umieścić informacje odnośnie zaprojektowanej hierarchii klas wraz z krótkim opisem najważniejszych metod.

Rozdział 3

Harmonogram realizacji projektu

W rozdziale tym należy umieścić harmonogram realizacji projektu – diagram Ganta. Rysunek oraz krótki opis do niego. Można napisać jakie problemy trudności wystąpiły w trakcie realizacji projektu. Rozdział ten musi zawierać informacje o repozytorium i systemie kontroli wersji.

UWAGA!!!

Należy pamiętać że pliki do projektu na repozytorium muszą być dostępne przed rok od dnia złożenia końcowej pracy. W przypadku gdy ktoś chciałby usunąć pliki z repozytorium do projektu należy dołączyć załącznik z plikami źródłowymi.

Rozdział 4

Prezentacja warstwy użytkowej projektu

W rozdziale tym należy przedstawić opis warstwy użytkowej projektu w tym celu należy umieścić opis aplikacji oraz PrtSc o których jest mowa.

Rozdział 5

Podsumowanie

W rozdziale należy opisać zrealizowane prace oraz ewentualne planowane dalsze prace rozwojowe projektu.

Bibliografia

- [1] <http://www.potaroo.net/tools/ipv4/> z dnia 8.12.2018
- [2] <http://bgp.potaroo.net/iso3166/v6dcc.html> z dnia 8.12.2018
- [3] Autorzy1, *Tytuł1*, Wydawnictwo1, Miasto1 Rok1.
- [4] Autorzy2, *Tytuł2*, Wydawnictwo2, Miasto2 Rok2.
- [5] Autorzy3, *Tytuł3*, Wydawnictwo3, Miasto3 Rok3.
- [6] Autorzy4, *Tytuł4*, Wydawnictwo4, Miasto4 Rok4.

Spis rysunków

Spis tabel