SymbolSource

# Definicje

|  |  |
| --- | --- |
| **Nazwa** | **Opis** |
| Administrator | Osoba odpowiedzialna za utrzymanie systemu. |
| Klient | Użytkownik końcowy systemu, nie wyróżniamy na tym etapie firmy, która być może zatrudnia tę osobę. |
| System | Całokształt przedstawianego rozwiązania informatycznego. |

# Wymagania biznesowe

|  |  |  |
| --- | --- | --- |
| **ID** | **Opis** | **Odniesienia** |
| BR1 | Klient może zapoznać się z możliwościami systemu na stronie WWW. |  |
| BR2 | Korzystanie z systemu nie wymaga instalowania dodatkowego oprogramowania na komputerze klienta. |  |
| BR3 | Klient może automatycznie pobierać źródła korzystając z Visual Studio. |  |
| BR4 | Klient może jednocześnie korzystać z innych serwerów źródeł. |  |
| BR5 | Klient może sprawdzić, jakie źródła są dostępne w systemie. |  |
| BR6 | Administrator może dodawać dowolne nowe źródła do systemu. |  |
| BR7 | Klient może wpływać na decyzje o dodaniu nowych źródeł do systemu. |  |
| BR8 | System może przechowywać dużą ilość źródeł w efektywny sposób. |  |
| BR9 | System udostępnia publicznie źródła do najważniejszych projektów OSS. |  |
| BR10 | Administrator może śledzić wykorzystanie systemu przez klientów. |  |
| BR11 | System wspiera samodzielne dodawanie źródeł przez autorów projektów. |  |

# Wymagania funkcjonalne

|  |  |  |
| --- | --- | --- |
| **ID** | **Opis** | **Odniesienia** |
| FR1 | Istnieje strona WWW przedstawiająca możliwości systemu oraz korzyści płynące z jego wykorzystania. | BR1 |
| FR2 | Na stronie WWW znajduje się instrukcja konfiguracji Visual Studio w celu korzystania z systemu. | BR2, BR3, BR4 |
| FR3 | Na stronie WWW znajduje się wykaz dostępnych projektów, wersji projektów oraz plików źródłowych powiązanych z daną wersją. | BR5 |
| FR4 | Dla każdej wersji projektu dostępny jest odnośnik do oficjalnej dystrybucji binarnej i źródłowej projektu. | BR5 |
| FR5 | Istnieje narzędzie umożliwiające modyfikację oficjalnych plików PDB tak, aby zawierały informacje niezbędne do automatycznego pobierania plików źródłowych z systemu przez Visual Studio. | BR6 |
| FR6 | Administrator ma możliwość załadowania plików PDB i plików źródłowych do systemu tak, aby było możliwe ich automatyczne dobieranie przez Visual Studio. | BR6 |
| FR7 | Załadowanie źródeł do systemu powoduje automatyczne uaktualnienie publicznych wykazów projektów, wersji projektów i plików źródłowych. | BR5, BR6 |
| FR8 | Pliki PDB i pliki źródłowe są zorganizowane w efektywny sposób, umożliwiający szybki dostęp do nich i zarządzanie nimi, zakładając obecność wielu projektów, wersji projektów i plików źródłowych (w tym plików niezmienionych pomiędzy wersjami projektów). | BR8 |
| FR9 | Klient ma na stronie WWW możliwość głosowania na nowe projekty i wersje projektów do dodania do systemu. | BR7 |
| FR10 | Administrator ma możliwość dodania źródeł z nieoficjalnymi plikami binarnymi. | BR9 |
| FR11 | Klient ma możliwość rejestracji w systemie i otrzymania swojego unikalnego identyfikatora. | BR10, BR11 |
| FR12 | System udostępnia pliki PDB i pliki źródłowe z uwzględnieniem unikalnego identyfikatora klienta. | BR10 |
| FR12 | Instrukcja konfiguracji Visual Studio uwzględnia konieczność wykorzystania unikalnego identyfikatora klienta. | BR10 |
| FR13 | Administrator ma możliwość przeglądania historii i sensownych statystyk wykorzystania systemu przez klientów. | BR10 |
| FR14 | Administrator ma możliwości przypisania klientowi prawa do publikacji źródeł w ramach wybranego projektu. | BR11 |
| FR15 | Istnieje narzędzie umożliwiające wsadowe wysyłanie źródeł do systemu przez uprawnionego klienta. | BR11 |

# Wymagania pozafunkcjonalne

|  |  |  |
| --- | --- | --- |
| **ID** | **Opis** | **Odniesienia** |
| NFR1 | System podzielony jest na niezależne komponenty umożliwiające rozdzielenie obowiązków pomiędzy osobne fizyczne maszyny. |  |
| NFR2 | System umożliwia efektywne przetwarzanie liczby źródeł na poziomie 450 tysięcy plików. | **BR8** |

# Ograniczenia

|  |  |  |
| --- | --- | --- |
| **ID** | **Opis** | **Odniesienia** |
| C1 | System wykorzystuje platformę .NET. |  |
| C2 | Moduły serwerowe są kompatybilne z systemem operacyjnym Linux i środowiskiem wykonawczym Mono. |  |

# Pogląd na system



# Decyzje architektoniczne

|  |  |  |
| --- | --- | --- |
| **ID** | **Opis** | **Odniesienia** |
| D1 | **Baza danych zostanie rozdzielona na część z plikami i metadanymi.**  Uzasadnienie: możliwe jest wykorzystanie różnych technologii utrwalania danych, dopasowanych dokładnie do różnych potrzeb tych dwóch grup. | BR6, NFR1 |
| D2 | **Przechowywanie metadanych będzie odbywać się w relacyjnej bazie danych PostreSQL.**  Uzasadnienie: metadane zawierają relacyjne i hierarchiczne powiązania oraz wymagają wydajnego pobierania wielu rekordów i agregacji danych. | NFR1, NFR2 |
| *Inne możliwości*: większość z potencjalnych rozwiązań jest płatna (jak np. OracleDB), bądź zbyt mało wydajna w większych projektach. |
| D3 | **Przechowywanie danych (plików) odbywać się będzie w Amazon S3.**  Uzasadnienie: usługa przechowania danych typu klucz-wartość jest wystarczająca do zapisywania i pobierania pojedynczych plików. Wykorzystanie *cloud storage* zmniejsza wymagania na powierzchnię dyskową i moc obliczeniową głównego serwera oraz zapewnia większą dostępność danych przez replikację. Pobieranie pojedynczych plików zdecydowaną stanowi większość przewidywanego obciążenia systemu. | NFR1 |
| *Inne możliwości*: wykorzystanie własnego serwera wiąże się nie tylko z wydatkiem dodatkowych pieniędzy, ale także kwestia jego utrzymania – dlatego też podjęto decyzję o wykorzystaniu Amazon S3. Na początku próbowano wykorzystać własny serwer, lecz szybko zauważono problem z dostępnością miejsca dyskowego oraz procesu wprowadzania dodatkowych dysków, jako zbyt pracochłonnego. |
| D4 | **Interfejs użytkownika końcowego zostanie wykonany w technologii ASP.NET MVC.**  Uzasadnienie: aplikacja webowa najlepiej spełnia wymagania dotyczące swobodnego dostępu do opisu projektu oraz katalogu udostępnianych źródeł. Model MVC gwarantuje sprawne tworzenie tego typu aplikacji oraz zapewnia łatwość testowania i dalszego rozwoju oprogramowania. | C1 |
| *Inne możliwości:* Doświadczenie projektantów pozwoliło szybko zadecydować o wykorzystaniu technologii wg. Modelu MVC – by ułatwić późniejszy rozwój aplikacji. |
| D5 | **Możliwość publikowania źródeł będzie oparta o usługi sieciowe SOAP.**  Uzasadnienie: usługi SOAP umożliwiają integrację z poziomu wielu różnych technologii i zapewniają oddzielenie implementacji serwera od publicznego interfejsu. Decyzja wspiera implementację różnych narzędzi do publikacji i eliminuje duplikację kodu obsługującego publikowanie. | C2, BR4, BR5, BR6 |
| D6 | **Komunikacja portalu WWW z bazą danych będzie odbywać się również za pomocą usług SOAP.**  Uzasadnienie: celem decyzji jest pełne odseparowanie serwera źródeł od opcjonalnych elementów zewnętrznych, takich jak interfejs użytkownika i narzędzia do zarządzania. Szczegóły implementacyjne serwera pozostają w pełni ukryte. Istnieje możliwość fizycznego rozdzielenia poszczególnych modułów systemu. | C2 |
| *Inne możliwości:* Od razu zadecydowano, że system będzie budowany w taki sposób, by maksymalnie ułatwić późniejszy proces jego rozbudowy. Dlatego też zadecydowano o odseparowaniu poszczególnych modułów oraz ukryciu informacji o poszczególnych komponentach składowych systemu. |
| D7 | **Komunikacja z bazą danych będzie odbywać się za pomocą NHibernate.**  Uzasadnienie: NHibernate jest dojrzałą i bogatą w funckje biblioteką do obsługi mapowania obiektowo-relacyjnego. Ułatwia korzystanie z bazy danych i eliminuje wiele potencjalnych błędów, np. ręczne sklejanie zapytań. | C1 |
| *Inne możliwości*: główny rywal to Entity Framework – z testów dostępnych w internecie, a także konsultacjach z ekspertami, wynika, że EF nie jest jeszcze na tyle rozbudowanym środowiskiem, by spełniać wymagania tak złożonego systemu. |
| D8 | **Implementacja usług SOAP i protokołu Visual Studio zostanie wykonana w osobnych modułach.**  Uzasadnienie: mimo korzystania z tych samych danych i wykonania w tej samej technologii (ASP.NET MVC), moduły zostaną rozdzielone ze względu na możliwość równoległej pracy programistów oraz lepszą separację zakresu obowiązków. | C11 |
| D9 | **Publikowanie źródeł przez autorów projektów będzie obsługiwane przez narzędzia OpenWrap i NuGet.**  Uzasadnienie: są to dwa główne rozwiązania problemu zarządzania zależnościami i paczkami na platformie .NET, więc jest duże prawdopodobieństwo, że będą dostępne w środowisku klienta. | C1, C2, BR3 |
| D10 | **Zostaną zaimplementowane niezależne bramki do przyjmowania paczek OpenWrap i NuGet, korzystające z usług SOAP głównego serwera.** | C2 |
| D11 | **Za obsługę przychodzących paczek i ładowanie źródeł do serwisu odpowiedzialny będzie wyspecjalizowany demon działający na serwerze.**  Uzasadnienie: załadowanie kilku tysięcy plików jest długotrwałą operacją, więc nie powinno blokować krótkotrwałych z definicji wywołań usług sieciowych SOAP. Synchroniczna będzie natomiast walidacja paczek, gwarantująca sukces późniejszej pracy demona. |  |

# Analiza decyzji

W tej sekcji przedstawione zostaną wnioski z podjętych decyzji w odniesieniu do każdej z nich.

### Technologia .NET

|  |  |  |
| --- | --- | --- |
|  | **POZYTYWNE** | **NEGATYWNE** |
| **WEW** | **MOCNE STRONY** - dobra znajomość środowiska przez programistów oraz doświadczenie w tworzeniu projektów na tej platformie, - dobrze rozwinięta struktura forów internetowych z pomocą dla programistów, - idealne do zastosowań w dużych projektach informatycznych, - stale rozwijany język zawierający najnowsze wynalazki programistyczne, - możliwość integracji z językami funkcjonalnymi (np. F#) oraz innymi składnikami CLR (*Common Language Runtime*). | **SŁABE STRONY** - duży koszt zakupu oprogramowania po wykorzystaniu możliwości programu BizSpark, - klient musi mieć zainstalowany .NET Framework by obsługiwać narzędzie. |
| **ZEW** | **SZANSE** - wykorzystanie zniżek i oprogramowania z programu BizSpark (dla młodych przedsiębiorstw). | **ZAGROŻENIA** - decyzja raz podjęta nie może być zmieniona. |

Łatwo zauważyć, że wybór technologii programistycznej był słusznie dokonany.

### Komunikacja z bazą danych - nHibernate

|  |  |  |
| --- | --- | --- |
|  | **POZYTYWNE** | **NEGATYWNE** |
| **WEW** | **MOCNE STRONY** - zbudowane na podstawie chwalonej bibliloteki – Hibernate dla języka Java,  - łatwe w integracji z praktycznie każdą bazą danych – możliwość podmiany na dowolną inną. | **SŁABE STRONY** - trudna konfiguracja (duża praca wejścia potrzebna do uruchomienia narzędzia), - potrzebna dosyć zaawansowana wiedza związana z bazami danych. |
| **ZEW** | **SZANSE** - stale rozwijany projekt, - częste aktualizacje źródeł na stronach, - projekt Open Source. | **ZAGROŻENIA** - duże straty czasu przy wdrażaniu nowych wersji biblioteki (szczególne, gdy aktualizowane źródła są często), - możliwość zablokowania działania systemu przy błędnym wdrażaniu nowej wersji biblioteki. |

# Architektura logiczna

Patrz plik Architektura.vsd.

# Architektura fizyczna

Patrz plik Architektura.vsd.