Specyfikacja wymagań dotyczących oprogramowania

projektu

<nazwa projektu>

Wersja 1.0 zatwierdzona

Przygotowany przez <autor>

<organizacja>

<data utworzenia >

Spis treści

Spis treści ii

Historia zmian ii

1. Wstęp 1

1.1 Cel 1

1.2 Konwencje przyjęte w dokumencie 1

1.3 Zakres projektu 1

1.4 Źródła 1

2. Informacje ogólne 1

2.1 Perspektywa produktu 2

2.2 Klasy użytkowników i ich charakterystyki 2

2.3 Środowisko robocze 2

2.4 Ograniczenia projektowe oraz implementacji 2

2.5 Założenia i zależności 2

3. Funkcje systemu 2

3.1 Funkcja systemu 1 3

3.1.1 Opis 3

3.1.2 Sekwencje bodziec/reakcja 3

3.1.3 Wymagania funkcjonalne 3

3.2 Funkcja systemu 2 (itd.) 3

4. Wymagania dotyczące danych 3

4.1 Logiczny model danych 3

4.2 Słownik danych 3

4.3 Raporty 4

4.4 Pozyskiwanie, integralność, przechowywanie i usuwanie danych 4

5. Wymagania dotyczące interfejsów zewnętrznych 4

5.1 Interfejsy użytkownika 4

5.2 Interfejsy programowe 4

5.3 Interfejsy sprzętowe 5

5.4 Interfejsy komunikacyjne 5

6. Atrybuty jakościowe 5

6.1 Użyteczność 5

6.2 Wydajność 5

6.3 Bezpieczeństwo 5

6.4 Ochrona 6

6.5 [Pozostałe, jeśli występują] 6

7. Wymagania dotyczące internacjonalizacji oraz lokalizacji 6

8. Inne wymagania 6

Historia zmian

|  |  |  |  |
| --- | --- | --- | --- |
| **Nazwa** | **Data** | **Powód zmiany** | **Wersja** |
|  |  |  |  |
|  |  |  |  |

# Wstęp

<We wstępie przedstawione są informacje ogólne, które pomogą czytelnikowi zrozumieć organizację specyfikacji SRS i sposób korzystania z niej.>

## Cel

<Zidentyfikuj produkt, którego wymagania dotyczące oprogramowania są wyspecyfikowane w niniejszym dokumencie, w tym numer wersji albo wydania. Opisz grupy czytelników, dla których przeznaczony jest dokument, takie jak programiści, menedżerowie projektu, marketingowcy, użytkownicy, testerzy albo autorzy dokumentacji.>

## Konwencje przyjęte w dokumencie

<Opisz wszystkie standardowe oraz typograficzne konwencje użyte w dokumencie, w tym znaczenie stylów, wyróżnień i notacji. Jeśli niepowtarzalne identyfikatory wymagań są nadawane ręcznie, możesz opisać ich format, co pozwoli innym osobom dodawać własne wymagania.>

## Zakres projektu

<Zamieść krótki opis specyfikowanego oprogramowania oraz jego cel. Powiąż oprogramowanie z celami użytkowników albo firmy, a także z celami i strategiami biznesowymi. Jeśli istnieje odrębny dokument wizji i zakresu (lub podobny), nie powielaj jego treści, tylko odwołaj się do niego. Specyfikacja SRS, która definiuje wydanie przyrostowe albo produkt ewoluujący, może zawierać własną deklarację zakresu stanowiącą podzbiór długoterminowej, strategicznej wizji produktu. Możesz zamieścić wysokopoziomowe podsumowanie głównych funkcji uwzględnionych w bieżącym wydaniu albo opisać istotne funkcje, które produkt realizuje.>

## Źródła

<Wymień wszelkie dokumenty oraz inne zasoby, do których odwołuje się SRS. Jeśli istnieją one w trwałej lokalizacji, zamieść do nich hiperłącza. Informacje te mogą obejmować opisy stylów, kontrakty, standardy, specyfikacje wymagań systemowych, specyfikacje interfejsów albo dokumenty SRS powiązanych produktów. Zamieść informacje, na podstawie których Czytelnik będzie mógł dotrzeć do każdego ze źródeł, tj. tytuł, autora, numer wersji, datę, miejsce przechowywania albo adres URL.>

# Informacje ogólne

<Niniejsza sekcja zawiera wysokopoziomowe omówienie produktu oraz środowiska, w którym będzie on używany, a także opis jego docelowych użytkowników, znanych ograniczeń, założeń i zależności.>

## Perspektywa produktu

<Opisz kontekst oraz pochodzenie produktu. Czy jest on kolejnym członkiem rozrastającej się rodziny produktów, kolejną wersją dojrzałego systemu, następcą istniejącej aplikacji, czy może zupełnie nowym produktem? Jeśli niniejszy dokument SRS definiuje składnik większego systemu, omów, w jakiej relacji pozostaje on wobec całości systemu, i zidentyfikuj główne interfejsy, które je łączą. Zastanów się nad dołączeniem modeli wizualnych, takich jak diagram kontekstowy albo mapa ekosystemu, które pokażą związki produktu z innymi systemami.>

## Klasy użytkowników i ich charakterystyki

<Zidentyfikuj różne klasy użytkowników, które zgodnie z Twoimi oczekiwaniami będą korzystać z produktu, i opisz ich cechy. Niektóre wymagania mogą odnosić się tylko do pewnych klas użytkowników. Zidentyfikuj klasy uprzywilejowane. Klasy użytkowników stanowią podzbiór interesariuszy opisanych w dokumencie wizji i zakresu. Opisy klas użytkowników są zasobem wielokrotnego użycia. Jeśli masz taką możliwość, możesz po prostu wskazać opisy klas użytkowników w głównym katalogu z klasami użytkowników, zamiast powielać w tym miejscu istniejące już informacje.>

## Środowisko robocze

<Opisz środowisko, w którym będzie działać oprogramowanie, w tym platformę sprzętową, systemy operacyjne i ich wersje, położenie geograficzne użytkowników, serwery, bazy danych, organizacje hostujące te bazy, serwery oraz strony internetowe. Wymień wszystkie pozostałe składniki oprogramowania oraz aplikacje, z którymi system musi współpracować. Jeśli łącznie z opracowaniem nowego systemu muszą zostać przeprowadzone rozległe prace w zakresie infrastruktury, weź pod uwagę możliwość opracowania odrębnej specyfikacji wymagań infrastrukturalnych, w której prace te zostaną szczegółowo omówione.>

## Ograniczenia projektowe oraz implementacji

<Omów wszelkie czynniki, które ograniczają dostępne dla programistów opcje. Mogą to być polityki firmowe albo przepisy prawne, ograniczenia sprzętowe (wymagania czasowe albo pamięciowe), interfejsy do innych aplikacji, określone technologie, narzędzia albo bazy danych, z których należy korzystać, wymagania albo ograniczenia języków programowania.>

## Założenia i zależności

<Wymień wszelkie przyjęte założenia, które (w odróżnieniu od znanych faktów) mogą wywierać wpływ na wymagania wymienione w dokumencie SRS. Można wśród nich wymienić pochodzące od zewnętrznych dostawców lub komercyjne komponenty, wymagania wielokrotnego użycia, kwestie związane ze środowiskiem programistycznym i roboczym albo ograniczenia. Założenia te mogą mieć wpływ, gdy są niewłaściwe, nie są współdzielone albo ulegną zmianie. Zidentyfikuj także wszelkie zewnętrzne czynniki, które znajdują się poza kontrolą projektu.>

# Funkcje systemu

<Niniejszy szablon ilustruje organizację wymagań funkcjonalnych produktu w podziale na funkcje systemu, tj. główne usługi realizowane przez produkt. Być może wolisz zorganizować niniejszą sekcję według przypadków użycia, trybów działania, klas użytkowników, klas obiektów, hierarchii funkcjonalnej, bodźców, reakcji (lub kombinacji tych dwóch elementów) albo jakiegokolwiek innego podziału, który ma sens w przypadku Twojego produktu.>

## Funkcja systemu 1

<Tylko nie pisz dosłownie „Funkcja systemu 1”; po prostu wyraź kilkoma słowami nazwę funkcji.>

### Opis

<Podaj krótki opis funkcji oraz wskaż, czy ma ona wysoki, średni, czy niski priorytet.>

### Sekwencje bodziec/reakcja

<Podaj sekwencje działań użytkowników i reakcji systemu, które definiują działanie danej funkcji.

Sekwencje te będą odpowiadać elementom dialogu przypisanym do przypadków użycia.>

### Wymagania funkcjonalne

<Wymień specyficzne wymagania funkcjonalne dotyczące danej funkcji. Są to możliwości systemu, które powinny zostać zaimplementowane, aby użytkownik mógł korzystać z usług udostępnianych przez funkcję albo realizować przypadek użycia. Opisz, jak produkt powinien reagować na spodziewane błędy. Jeśli niezbędne informacje są jeszcze niedostępne, skorzystaj z symbolu TBD (do ustalenia).>

## Funkcja systemu 2 (itd.)

# Wymagania dotyczące danych

<W sekcji tej opisane są różne aspekty danych, które system będzie pobierać na wejściu, przetwarzać je albo tworzyć na wyjściu.>

## Logiczny model danych

<Model danych to wizualna reprezentacja obiektów danych i ich zbiorów, które będą przetwarzane przez system, a także zachodzących między nimi relacji. Zamieść tu model danych operacji biznesowych realizowanych przez system albo logiczną reprezentację danych, które system będzie przetwarzać. Modele danych są najczęściej tworzone w postaci diagramów związków encji.>

## Słownik danych

<Słownik danych definiuje skład struktur danych oraz znaczenie, typy danych, długość, format i dozwolone wartości elementów danych tworzących te struktury. W wielu przypadkach lepiej będzie przechowywać słownik danych jako odrębny dokument, zamiast umieszczać go w obrębie specyfikacji SRS. Dzięki temu zwiększa się również potencjał ponownego użycia słownika w innych projektach. >

## Raporty

<Jeśli Twoja aplikacja będzie generować raporty, zidentyfikuj je i opisz ich cechy. Jeśli raport musi być zgodny ze wstępnie zdefiniowanym układem, możesz taki układ zdefiniować jako ograniczenie i być może podać jego przykład. W innym przypadku skup się na logicznym opisie zawartości raportu, kolejności sortowania, poziomach sum częściowych itd., odkładając szczegółowe opracowanie układu raportu do momentu prac projektowych.>

## Pozyskiwanie, integralność, przechowywanie i usuwanie danych

<Opisz, jak są pozyskiwane i przechowywane dane. Określ wszelkie wymagania dotyczące potrzeby chronienia integralności danych systemu. Zidentyfikuj potrzebne techniki, takie jak tworzenie kopii bezpieczeństwa, stosowanie punktów kontrolnych, mirroring albo weryfikacja poprawności danych. Określ polityki, jakie będą wymuszane w systemie związane z przechowywaniem lub usuwaniem danych, w tym danych tymczasowych, metadanych, danych rezydualnych (takich jak usunięte rekordy), danych w pamięci podręcznej, kopii lokalnych i tymczasowych kopii bezpieczeństwa.>

# Wymagania dotyczące interfejsów zewnętrznych

<W tej sekcji znajdują się informacje gwarantujące, że system będzie się poprawnie komunikować z użytkownikami, sprzętem zewnętrznym i elementami oprogramowania.>

## Interfejsy użytkownika

<Opisz cechy logiczne każdego z interfejsów łączących produkt z jego użytkownikami. Taki opis może obejmować przykładowe ekrany, standardy graficznych interfejsów użytkownika albo wskazówki dotyczące stylów stosowanych w rodzinie produktów, ograniczenia dotyczące układu ekranów, standardowe przyciski i funkcje (np. pomoc) pojawiające się na wszystkich ekranach, skróty klawiaturowe, standardy wyświetlania komunikatów o błędach itd. Zdefiniuj składniki oprogramowania, dla których wymagany jest interfejs użytkownika. Szczegóły interfejsu użytkownika powinny zostać udokumentowane w odrębnej specyfikacji interfejsu użytkownika.>

## Interfejsy programowe

<Opisz połączenia istniejące między produktem a innymi składnikami oprogramowania (zidentyfikowanymi za pomocą nazwy i wersji), w tym innymi aplikacjami, bazami danych, systemami operacyjnymi, narzędziami, bibliotekami, stronami internetowymi oraz zintegrowanymi składnikami komercyjnymi. Określ cel, formaty i zawartość komunikatów, danych i wartości kontrolnych, które są wymieniane między składnikami oprogramowania. Wyspecyfikuj mapowania danych wejściowych i wyjściowych wymienianych między systemami oraz konwersje, które należy przeprowadzić, aby przesyłać dane z jednego systemu do drugiego. Opisz usługi wymagane przez zewnętrzne składniki oprogramowania lub przez nie dostarczane oraz naturę komunikacji między poszczególnymi składnikami. Zidentyfikuj dane, które będą wymieniane między składnikami oprogramowania lub, z których składniki te będą wspólnie korzystać. Wyspecyfikuj wymagania pozafunkcjonalne mające wpływ na interfejs, takie jak poziom usług związany z czasem i częstotliwością reakcji lub ograniczenia i metody kontroli dotyczące bezpieczeństwa.>

## Interfejsy sprzętowe

<Opisz cechy wszystkich interfejsów łączących programistyczne i sprzętowe składniki systemu (jeśli istnieją). Opis może obejmować uwzględnione typy urządzeń, interakcje danych zachodzące między oprogramowaniem i sprzętem, a także protokoły komunikacyjne, z których należy korzystać. Wymień dane wejściowe i wyjściowe, ich formaty, dozwolone wartości albo zakresy oraz ograniczenia czasowe, o których powinni wiedzieć programiści. Jeśli tego rodzaju informacje są obszerne, weź pod uwagę możliwość utworzenia odrębnego dokumentu ze specyfikacją interfejsów.>

## Interfejsy komunikacyjne

<Określ wymagania dotyczące wszelkich funkcji komunikacyjnych, które zostaną użyte w produkcie, w tym wiadomości e-mail, przeglądarki internetowe, protokoły sieciowe i formularze elektroniczne. Zdefiniuj formaty wiadomości. Wyspecyfikuj kwestie związane z bezpieczeństwem komunikacji, szyfrowaniem, prędkościami transmisji danych, handshakingiem i mechanizmami synchronizacji. Określ ograniczenia dotyczące tych interfejsów, np. dopuszczalność dodawania załączników w wiadomościach e-mail.>

# Atrybuty jakościowe

## Użyteczność

<Określ wszelkie wymagania dotyczące cech, które sprawią, że oprogramowanie będzie wyglądać na „przyjazne dla użytkownika”. Użyteczność obejmuje prostotę użycia, łatwość uczenia się, zapamiętywalność, unikanie błędów, obsługę, przywracanie po awarii, skuteczność interakcji, dostępność i ergonomię. Czasami cechy te mogą być ze sobą sprzeczne, jak to ma miejsce w przypadku prostoty użycia i łatwości uczenia się. Wskaż wszelkie standardy i wytyczne projektowania interfejsu użytkownika, z którymi aplikacja musi być zgodna.>

## Wydajność

<Zdefiniuj wymagania wydajnościowe dotyczące rozmaitych operacji wykonywanych w systemie. Jeśli różne wymagania funkcjonalne albo funkcje charakteryzują się różnymi wymaganiami wydajnościowymi, stosowne będzie wyspecyfikowanie ich łącznie z odpowiednimi wymaganiami funkcjonalnymi i pominięcie ich w niniejszej sekcji.>

## Bezpieczeństwo

<Wyspecyfikuj wszelkie wymagania dotyczące bezpieczeństwa i prywatności, które ograniczają dostęp do oprogramowania lub możliwości korzystania z niego. Wymagania te mogą dotyczyć bezpieczeństwa fizycznego, danych lub oprogramowania. Wymagania związane z bezpieczeństwem często wywodzą się z reguł biznesowych, w związku z czym należy zidentyfikować wszelkie dotyczące bezpieczeństwa polityki i przepisy, z którymi produkt musi być zgodny. Jeśli reguły te są już udokumentowane w repozytorium reguł biznesowych, wystarczy zamieścić odwołanie do nich.>

## Ochrona

<Wyspecyfikuj wymagania związane z możliwością powstania strat, zniszczeń lub urazów wynikających z korzystania z produktu. Zdefiniuj zabezpieczenia albo działania, które należy podjąć, a także potencjalnie niebezpieczne czynności, których należy unikać. Zidentyfikuj związane z ochroną certyfikaty, polityki i przepisy, z którymi produkt musi być zgodny.>

## [Pozostałe, jeśli występują]

< Dla każdego dodatkowego atrybutu jakościowego utwórz odrębną sekcję specyfikacji SRS, w której opiszesz cechy produktu ważne dla klientów albo programistów. Takimi cechami mogą być dostępność, wydajność, łatwość instalacji, integralność, interoperacyjność, modyfikowalność, przenośność, niezawodność, wielokrotność użycia, odporność, skalowalność albo weryfikowalność. Należy je opisać w sposób jednoznaczny, ilościowy i weryfikowalny. Określ relatywne priorytety poszczególnych atrybutów, jak na przykład bezpieczeństwo ponad wydajnością.>

# Wymagania dotyczące internacjonalizacji oraz lokalizacji

<Wymagania dotyczące internacjonalizacji i lokalizacji gwarantują, że produkt znajdzie zastosowanie w różnych krajach, kulturach i lokalizacjach geograficznych innych niż te, w których został opracowany. Tego rodzaju wymagania mogą dotyczyć różnic w walutach, formatowaniu dat, liczb, adresów, numerów telefonicznych, języków (w tym różnych konwencji w obrębie jednego języka, jak na przykład angielski brytyjski i amerykański), stosowanych symboli i zestawów znaków, kolejności pisania imion i nazwisk, stref czasowych, praw i przepisów, zagadnień kulturowych i politycznych, rozmiarów papieru, wag i miar, napięć elektrycznych i kształtów wtyczek itp.>

# Inne wymagania

<Wśród przykładów można wymienić zgodność z uregulowaniami lub zasadami finansowymi, wymagania dotyczące standardów, wymagania związane z instalacją konfiguracją, rozruchem i wyłączaniem produktu, a także wymagania dotyczące logowania, monitorowania i śledzenia. Zamiast wymieniać wszystkie te wymagania w punkcie „Inne”, dodaj do szablonu nowe sekcje, które będą mieć związek z Twoim projektem. Pomiń niniejszą sekcję, jeśli wszystkie wymagania znalazły swoje miejsce w pozostałych sekcjach.>

Dodatek A. Glosariusz

<Zdefiniuj wszelkie wyspecjalizowane terminy, które w celu zrozumienia specyfikacji SRS powinien znać czytelnik, w tym akronimy i skróty. Rozwiń każdy akronim i podaj jego definicję. Weź pod uwagę możliwość utworzenia na poziomie całego przedsiębiorstwa glosariusza wielokrotnego użycia, który obejmowałby wiele projektów. Wszystkie terminy związane z danym projektem byłyby wyjaśniane za pomocą odsyłaczy do tego glosariusza.>

Dodatek B. Modele analityczne

<Ta nieobowiązkowa sekcja zawiera albo wymienia modele analityczne, takie jak diagramy przepływu danych, drzewa funkcjonalności, diagramy przejść stanów albo diagramy związków encji. Zamiast umieszczać tego rodzaju modele pod koniec dokumentu, możesz wstawić niektóre z modeli w odpowiednich sekcjach specyfikacji.>