Specyfikacja wymagań

Software Requirements Specification

# Wersja

|  |  |  |  |
| --- | --- | --- | --- |
| Wersja | Data | Autor | Zakres zmian |
| 0.1.0 | 2020-10-07 | Artur Karczmarczyk | Inicjalizacja szablonu dokumentu |

# Spis treści

[Wersja 1](#_Toc52991313)

[Spis treści 1](#_Toc52991314)

[1. Wprowadzenie 2](#_Toc52991315)

[1.1 Cel 2](#_Toc52991316)

[1.2 Przyjęte konwencje 2](#_Toc52991317)

[1.3 Zakres projektu 2](#_Toc52991318)

[1.4 Odwołania 2](#_Toc52991319)

[2. Opis ogólny 2](#_Toc52991320)

[2.1 Perspektywa produktu 2](#_Toc52991321)

[2.2 Klasy i charakterystyki użytkowników 2](#_Toc52991322)

[2.3 Środowisko działania 3](#_Toc52991323)

[2.4 Ograniczenia projektowe i wykonawcze 3](#_Toc52991324)

[2.5 Założenia i zależności 3](#_Toc52991325)

[3. Funkcjonalności systemu 3](#_Toc52991326)

[3.x Funkcjonalność XYZ 3](#_Toc52991327)

[3.x.1 Opis 3](#_Toc52991328)

[3.x.2 Wymagania funkcjonalne 3](#_Toc52991329)

[4. Wymagania dotyczące danych 3](#_Toc52991330)

[4.1 Logiczny model danych 3](#_Toc52991331)

[4.2 Raporty 3](#_Toc52991332)

[4.3 Pozyskiwanie, integralność przechowywanie I usuwanie danych 4](#_Toc52991333)

[5. Wymagania dotyczące interfejsu zewnętrznego 4](#_Toc52991334)

[5.1 Interfejsy użytkownika 4](#_Toc52991335)

[5.2 Interfejsy programowe 4](#_Toc52991336)

[5.3 Interfejsy sprzętowe 4](#_Toc52991337)

[5.4 Interfejsy komunikacyjne 4](#_Toc52991338)

[6. Cechy jakości 5](#_Toc52991339)

[6.1 Użyteczność 5](#_Toc52991340)

[6.2 Wydajność 5](#_Toc52991341)

[6.3 Zabezpieczenia 5](#_Toc52991342)

[6.4 Bezpieczeństwo użytkowania 5](#_Toc52991343)

[6.x Inne 5](#_Toc52991344)

[7. Wymagania internacjonalizacji i lokalizacji 5](#_Toc52991345)

[8. Inne wymagania 5](#_Toc52991346)

[Suplement A: Słownictwo 5](#_Toc52991347)

[Suplement B: Modele analityczne 5](#_Toc52991348)

# 1. Wprowadzenie

We wprowadzeniu znajduje się przegląd, który ma pomóc czytelnikowi zrozumieć, jak zorganizowany jest ten dokument i jak z niego korzystać.

## 1.1 Cel

Aplikacja będzie nosić nazwę Clocker. Jej głównym zadaniem będzie śledzenie roboczogodzin spędzonych przez użytkowników systemu nad poszczególnymi zadaniami w projekcie. Klienci będą mogli być przypisywani do projektów przez Project Ownerów gdzie będą oni mieli możliwość uruchamiania i zatrzymywania liczników czasu danego zadania (timer). Takim sposobem pod koniec realizacji przedsięwzięcia PO będą mogli sprawdzić na jakie segmenty pracownicy poświęcili jaką ilość czasu, aby móc w przyszłości lepiej zarządzać projektami.

Głównymi odbiorcami będą kierownicy projektów oraz pracownicy. Uniwersalność zastosowania aplikacji pozwala na użycie jej w różnorakich typach projektów.

## 1.2 Przyjęte konwencje

Poprzez skrót PO rozumiany jest Project Owner

# 1.3 Zakres projektu

Zamysł oraz główne zadanie aplikacji zostało opisane w punkcie 1.1

## 1.4 Odwołania

Odwołania do wszystkich przydatnych dokumentów, do których odwołuje się SRS.

# 2. Opis ogólny

Ta sekcja przedstawia ogólny przegląd produktu i środowiska, w którym będzie używany, przewidywanych użytkowników oraz znanych ograniczeń, założeń i zależności

Przewidywanym środowiskiem pracy aplikacji są firmy, w których menedżerowie potrzebują określenie spędzonych roboczogodzin na poszczególnych projektach oraz ich zadaniach.

Przewidywani użytkownicy oraz założenia zostały opisane w punkcie 1.1

Głównym ograniczeniem jest poleganie na pracownikach i ich systematyczności w sprawie dokumentowania spędzonego czasu.

## 2.1 Perspektywa produktu

Aplikacja będzie nowym produktem ukazującym kolejne rozwiązanie problemu śledzenia pracy nad zadaniami oraz projektami. Będzie ona odrębnym indywidualnym produktem.

## 2.2 Klasy i charakterystyki użytkowników

Typy użytkowników:  
- Administrator (Project Owner)

- Pracownik

## 2.3 Środowisko działania

Platforma sprzętowa:

- Urządzenia stacjonarne (PC, Mac, Linux)

-Urządzenia mobilne (Android, iOS)

Baza danych:

-SQL (Na czas produkcji – lokalna baza danych)

## 2.4 Ograniczenia projektowe i wykonawcze

Technologie używane przy produkcji aplikacji:  
- HTML

- CSS

- JavaScript

- PHP

- SQLi

- Node (utworzenie środowiska programistycznego)

- XAMPP (Apache, SQL)

## 2.5 Założenia i zależności

Do poprawnego korzystania z końcowego produktu wymagana jest jedynie przeglądarka internetowa co automatycznie poszerza grono odbiorców oraz zdejmuje ograniczenia występujące w natywnych aplikacjach danych platform. Do poprawnego hostowania aplikacji potrzebny będzie serwer Apache oraz lokalna baza danych, co dostarcza pakiet XAMPP.

# 3. Funkcjonalności systemu

Wymagania funkcjonalne można zorganizować według obszaru funkcjonalnego, przebiegu procesu, przypadku użycia, trybu działania, klasy użytkownika itp. Możliwe są również hierarchiczne kombinacje tych elementów, takie jak przypadki użycia w klasach użytkowników. Należy wybrać taką metodę organizacji, która ułatwi czytelnikom zrozumienie zamierzonych możliwości produktu.

## 3.1 Funkcjonalność Logowanie/Rejestracja

Została zaimplementowana funkcjonalność pozwalająca na sprawne logowanie się użytkowników jak i również ich rejestrację.

### 3.1.1 Opis

Po zalogowaniu treść aplikacji jest zależna od typu konta użytkownika.

### 3.1.2 Wymagania funkcjonalne

Po zalogowaniu na serwerze jest tworzona sesja użytkownika, na której opiera się późniejsza komunikacja klienta z serwerem.

## 3.2 Funkcjonalność Zarządzanie projektami, klientami oraz grupami użytkowników

Została zaimplementowana funkcjonalność zarządzania projektami, klientami oraz grupami użytkowników.

### 3.2.1 Opis

Funkcjonalność polega na możliwości dodania projektu wraz z klientem oraz przypisaniu mu grupy, do której również można przypisywać użytkowników (developerzy itp…)

### 3.2.2 Wymagania funkcjonalne

Po dodaniu któregokolwiek z wyżej wymienionych elementów po stronie klienta, baza danych jest aktualizowana.

## 3.3 Funkcjonalność Dodawanie zadań

Została zaimplementowana funkcjonalność pozwalająca na dodawanie zadań do wcześniej utworzonych projektów.

### 3.3.1 Opis

Użytkownik może dodać do konkretnego projektu zadanie, które następnie może zostać wykonywane przez przypisanych do grupy projektu na przykład developerów.

### 3.3.2 Wymagania funkcjonalne

Baza danych otrzymuje aktualizacje każdorazowo po dodaniu zadania do projektu.

## 3.4 Funkcjonalność Zarządzanie czasem zadania

Została zaimplementowana funkcjonalność pozwalająca na sprawne obserwowanie czasu jaki został poświęcony dotychczas na konkretne zadanie w konkretnym projekcie.

### 3.4.1 Opis

Po dodaniu zadania użytkownicy mogą wystartować z nim związany czasomierz. W trakcie wykonywania zadania użytkownicy powinni wznowić konkretny czasomierz a po zakończonej pracy nad danym zagadnieniem go zatrzymać.

### 3.4.2 Wymagania funkcjonalne

Czas jest aktualizowany w bazie danych każdorazowo przy akcji wznawiania lub zatrzymywania czasomierza.

## 3.5 Funkcjonalność Serwowanie treści zależnej od typu konta użytkownika

Została zaimplementowana funkcjonalność pozwalająca serwowanie różnych funkcji zależnie od typu konta zalogowanego użytkownika

### 3.5.1 Opis

Po zalogowaniu treść aplikacji jest dobierana zależnie od typu konta zalogowanego użytkownika (Admin, użytkownik) oraz od przynależności do projektów (Użytkownik widzi tylko te projekty, do których został wcześniej przypisany).

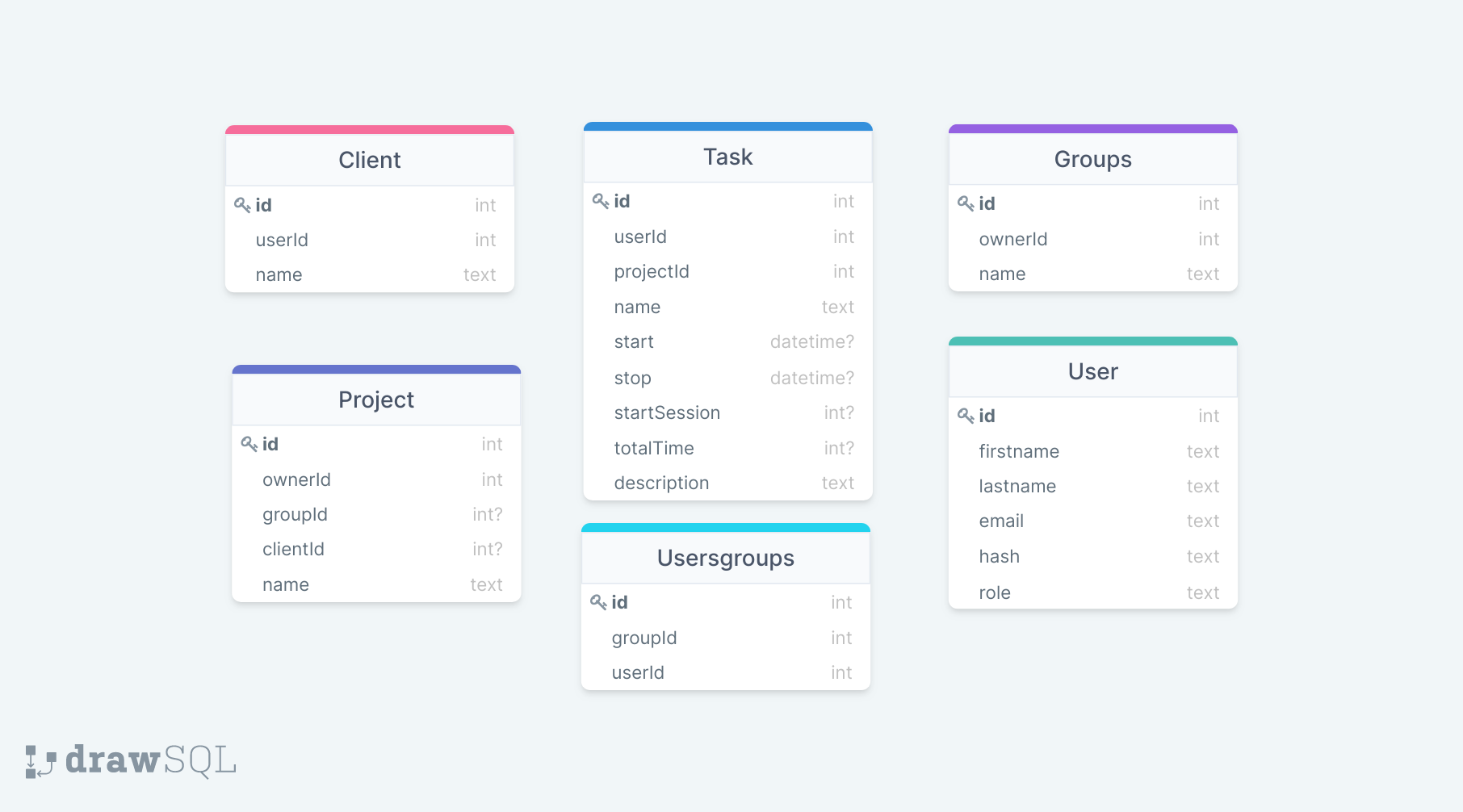
### 3.5.2 Wymagania funkcjonalne

W plikach sesji na serwerze przetrzymywany jest typ konta użytkownika co pozwala na dynamiczne wyświetlanie treści po stronie klienta.

# 4. Wymagania dotyczące danych

W tej sekcji szablonu należy opisać różne aspekty danych, które system będzie wykorzystywał jako dane wejściowe, przetwarzał w jakiś sposób lub produkował jako dane wyjściowe.

## 4.1 Logiczny model danych



## 4.2 Raporty

Aplikacja nie generuje żadnych raportów.

## 4.3 Pozyskiwanie, integralność przechowywanie I usuwanie danych

Sposób w jakim są przechowywane dane znajduje się w punkcie 4.1.

# 5. Wymagania dotyczące interfejsu zewnętrznego

Ta sekcja zawiera informacje zapewniające prawidłową komunikację systemu z użytkownikami oraz z zewnętrznymi elementami sprzętu lub oprogramowania. Złożony system z wieloma mikroserwisami powinien tworzyć oddzielną specyfikację interfejsu lub specyfikację architektury systemu. Dokumentacja interfejsu może zawierać materiały z innych dokumentów poprzez odwołania.

## 5.1 Interfejsy użytkownika

Głównym elementem, który widnieje na każdym widoku jest pasek nawigacji, który pozwala na sprawne przełączenie pomiędzy interfejsami.

Interfejsy użytkownika:

* Widok „Home”:
  + Wyświetla opis aplikacji
  + Wyświetla statystyki
* Widok Rejestracji:
  + Pozwala na rejestrację nowego użytkownika
* Widok Logowania:
  + Pozwala na zalogowanie się użytkownika
* Widok Panelu Administratora:
  + Pozwala on na:
    - Zarządzanie projektami
    - Zarządzanie grupami
    - Zarządzanie użytkownikami
    - Zarządzanie klientami
* Widok Panelu użytkownika
  + Pozwala on na:
    - Zarządzanie kontem
    - Zarządzanie zadaniami w projektach
* Widok Projektu:
  + Pozwala na:
    - Wznowienie lub zatrzymanie czasomierza danego zadania
    - Przegląd zadań i ich treści

## 5.2 Interfejsy programowe

Jedynym aktywnym połączeniem w aplikacji jest połączenie z bazą danych. Dane są wymieniane za pomocą języka SQL oraz biblioteki PDO. Dane zwrotne z bazy danych występują w postaci obiektowej za sprawą ORM (PDO) a następnie są przetwarzane przez interfejs użytkownika do poprawnego wyświetlania. Cały przepływ danych w aplikacji opiera się o model MVC.

## 5.3 Interfejsy sprzętowe

Do poprawnego działania aplikacji wystarczy jedynie przeglądarka internetowa, która jest niezależna od systemu.

## 5.4 Interfejsy komunikacyjne

Aplikacja korzysta z protokołu HTTP (port 80) używanego przez przeglądarkę. Zapytania do serwera są wysyłane metodą GET oraz POST w celu wyświetlenia informacji z bazy danych po stronie użytkownika.

# 6. Cechy jakości

Aplikacja ładuje się w nie więcej niż 2 sekundy.

Dane wrażliwe są przechowywane wyłącznie po stronie serwera

Strona jest serwowana za pomocą serwera Apache

Z uwagi na hostowanie aplikacji na maszynie hostującej bazę danych pobieranie danych z bazy następuje natychmiastowo

Maksymalne przeciążenia są dyktowane oprogramowaniem XAMPP

## 6.1 Użyteczność

Strona z uwagi na małą ilość funkcjonalności ma bardzo przejrzysty i łatwo w obsłudze interfejs użytkownika. Użytkownik posiadający podstawową wiedzę o funkcjonalności produktu bez problemu automatycznie będzie potrafił się po nim poruszać.

## 6.2 Wydajność

Każda funkcjonalność bazuje na wykonaniu 1-3 podstawowych zapytań do bazy danych co gwarantuje dobrą wydajność.

## 6.3 Zabezpieczenia

Na jednej maszynie jest hostowana strona www oraz baza danych co daje większe bezpieczeństwo z uwagi na brak zewnętrznych zapytań poza maszynę hostującą.

Wszelkie wrażliwe dane użytkownika są przechowywane po stronie serwera co zwiększa bezpieczeństwo.

Hasła użytkowników po stronie serwera są hashowane za pomocą algorytmu Crypt\_BlowFish Algorithm i przechowywane w takiej formie w bazie danych.

## 6.4 Bezpieczeństwo użytkowania

Z uwagi na zastosowanie ORM (PDO) aplikacja jest odporna na ataki typu SQL-Injection. System sesji zapobiega wycieknięciu danych.

# 7. Wymagania internacjonalizacji i lokalizacji

Aplikacja będzie hostowana na jednej maszynie umiejscowionej w Polsce przez co dostęp z innych krajów może wydłużyć czas oczekiwania na zwrot danych. Głównym odbiorcą jest statystyczny Polak-Deweloper.

# Suplement A: Słownictwo

PO - Project Owner

PDO – Php Data Objects

ORM – Object-Relational Mapping

MVC – Model-View-Controller

PHP – Personal Home Page

JS – JavaScript

SQL – Structured Query Language

HTML – HyperText Markup Language