Specyfikacja wymagań

Aplikacja “Ogar”

Autorzy: Dominik Olejarz & Szczepaniak

Niech ​**EndUser**​ będzie aliasem dla klienta końcowego aplikacji, czyli osoby która zgłasza problem do konsultanta danej firmy.

Gdy w zadaniu pojawia się słowo użytkownik, to mówimy o funkcjach wspólnych dla konsultanta i EndUsera.

**Wymagania funkcjonalne:**

1. **Funkcje integracji z innymi systemami:**

* System powinien umożliwiać integrację z istniejącymi systemami związanymi z projektami, takimi jak systemy zarządzania zasobami ludzkimi, narzędziami do analizy danych, itp.

1. **Funkcje analizy danych związanych z projektami:**

* System powinien oferować zaawansowane funkcje analizy danych związanych z projektami, umożliwiając EndUserowi i przedstawicielom firmy śledzenie postępu, identyfikowanie trendów i podejmowanie informowanych decyzji.

1. **Terminarz:**

* Aplikacja powinna zawierać terminarz umożliwiający zarządzanie zadaniami, spotkaniami, i innymi wydarzeniami związanymi z projektem.

1. **Funkcje współpracy – udostępnianie plików, komunikacja:**

* System powinien umożliwiać łatwe udostępnianie plików pomiędzy EndUserem a konsultantem, oraz wspieranie efektywnej komunikacji między użytkownikami.

1. **Automatyzacja raportów, generowanie prognoz:**

* Aplikacja powinna zawierać funkcje automatyzacji procesu generowania raportów związanych z projektami, a także umożliwiać prognozowanie na podstawie dostępnych danych.

1. **Wybór konkretnego projektu:**
   * EndUser powinien mieć łatwy dostęp do listy projektów i możliwość wyboru interesującego go projektu spośród dostępnych opcji.
2. **Odbieranie wiadomości:**
   * Użytkownik powinien mieć łatwy dostęp do wiadomości, a także przesyłanych plików wraz z komentarzem, zarówno po stronie EndUsera, jak i konsultanta.
3. **Dostęp do historii konwersacji:**
   * Użytkownik powinien mieć możliwość przeglądania historii konwersacji z daną firmą, umożliwiając śledzenie wcześniejszych interakcji.

2. User stories

**1. Historia: Tworzenie Nowego Projektu**

*Jako Project Manager, chcę móc utworzyć nowy projekt, aby zacząć planowanie i delegowanie zadań.*

* Akcja: Użytkownik otwiera aplikację.
* Akcja: Przechodzi do sekcji "Nowy Projekt".
* Akcja: Wprowadza podstawowe informacje o projekcie (nazwa, opis, data rozpoczęcia, data zakończenia).
* Akcja: Przypisuje członków zespołu do projektu.
* Wynik: Nowy projekt jest utworzony i widoczny dla wszystkich członków zespołu.

**2. Historia: Dodawanie Zadania do Projektu**

*Jako Project Manager, chcę mieć możliwość dodawania zadań do projektu, aby precyzyjnie określić cele i etapy realizacji.*

* Akcja: Użytkownik wybiera istniejący projekt.
* Akcja: Przechodzi do sekcji "Dodaj Zadanie".
* Akcja: Wprowadza opis zadania, priorytet, termin wykonania, i przypisuje odpowiednich członków zespołu.
* Wynik: Zadanie jest dodane do projektu, a członkowie zespołu są poinformowani o nowym zadaniu.

**3. Historia: Śledzenie Postępu Zadania**

*Jako Project Manager, chcę mieć możliwość monitorowania postępu zadań, aby ocenić efektywność realizacji projektu.*

* Akcja: Użytkownik otwiera projekt.
* Akcja: Przechodzi do sekcji "Zadania".
* Akcja: Sprawdza status i postęp każdego zadania.
* Akcja: Aktualizuje status zadania na podstawie postępu prac.
* Wynik: Zespół ma aktualne informacje o postępie projektu, co ułatwia podejmowanie decyzji i reagowanie na ewentualne opóźnienia.

**4. Historia: Zarządzanie Harmonogramem Projektu**

*Jako Project Manager, chcę móc zarządzać harmonogramem projektu, aby skoordynować działania zespołu i unikać konfliktów terminowych.*

* Akcja: Użytkownik otwiera projekt.
* Akcja: Przechodzi do sekcji "Harmonogram".
* Akcja: Edytuje daty rozpoczęcia i zakończenia projektu oraz poszczególnych etapów.
* Akcja: Otrzymuje powiadomienia o potencjalnych konfliktach terminowych.
* Wynik: Zespół ma aktualny harmonogram, co ułatwia planowanie i zminimalizowanie ryzyka opóźnień.

**5. Historia: Komunikacja w Zespole**

*Jako Project Manager, chcę mieć narzędzie do efektywnej komunikacji z zespołem, aby ułatwić wymianę informacji i rozwiązywanie problemów.*

* Akcja: Użytkownik przechodzi do sekcji "Wiadomości" w ramach projektu.
* Akcja: Tworzy nową wiadomość, wybiera odbiorców (cały zespół lub konkretni członkowie).
* Akcja: Dołącza pliki, jeśli to konieczne, i wysyła wiadomość.
* Wynik: Zespół ma dostęp do istotnych informacji, a wszelkie dyskusje i decyzje są zarejestrowane w ramach projektu.

**6. Historia: Raportowanie Postępu Projektu**

*Jako Project Manager, chcę mieć możliwość generowania raportów dotyczących postępu projektu, aby dostarczyć informacje interesariuszom.*

* Akcja: Użytkownik przechodzi do sekcji "Raporty".
* Akcja: Wybiera kryteria raportowania (np. postęp projektu, zużycie zasobów, harmonogram).
* Akcja: Generuje raport i eksportuje go w odpowiednim formacie (np. PDF, Excel).
* Wynik: Project Manager ma gotowy raport do udostępnienia interesariuszom, co ułatwia śledzenie postępu projektu i podejmowanie strategicznych decyzji.

3. Wymagania niefunkcjonalne

1. **Wydajność:**
   * Aplikacja powinna obsługiwać równoczesne zapytania od wielu użytkowników (co najmniej 100 aktywnych użytkowników jednocześnie) bez utraty responsywności.
   * Czas ładowania strony i operacji w aplikacji nie powinien przekraczać 2 sekund.
2. **Dostępność:**
   * Dostępność systemu powinna być utrzymana na poziomie 99,9% w ciągu miesiąca, z wyłączeniem planowanych przerw technicznych.
   * Mechanizmy backupu i przywracania danych powinny być skonfigurowane i testowane regularnie, aby zminimalizować ryzyko utraty danych.
3. **Bezpieczeństwo:**
   * Wszystkie dane przesyłane między aplikacją a użytkownikami powinny być szyfrowane przy użyciu najnowszych standardów kryptograficznych.
   * System uwierzytelniania i autoryzacji powinien być zaimplementowany zgodnie z najwyższymi standardami bezpieczeństwa.
4. **Zgodność z Przeglądarkami:**
   * Aplikacja powinna być w pełni kompatybilna z najnowszymi wersjami popularnych przeglądarek internetowych, takich jak Google Chrome, Mozilla Firefox, Safari i Microsoft Edge.
5. **Skalowalność:**
   * System powinien być elastyczny i łatwo skalowalny w celu dostosowania się do wzrostu liczby użytkowników i projektów.
   * Skalowanie powinno być możliwe zarówno wertykalnie (więcej zasobów na jednym serwerze) jak i horyzontalnie (dodawanie dodatkowych serwerów).
6. **Testowanie i Zapewnienie Jakości:**
   * Wszystkie zmiany w kodzie aplikacji powinny być poddawane kompleksowym testom jednostkowym, integracyjnym i akceptacyjnym przed wprowadzeniem ich do produkcji.
   * Wprowadzenie automatyzacji testów, w tym testów jednostkowych, integracyjnych i wydajnościowych, jest zalecane dla zapewnienia stabilności systemu.
7. **Interfejs Użytkownika:**
   * Interfejs użytkownika powinien być intuicyjny i łatwy w obsłudze, z minimalną ilością kroków potrzebnych do wykonania kluczowych zadań.