Aplikacja do zarządzania Działem Utrzymania Ruchu.

Chudziaszek Piotr Krzysztof
Darmetko Alan Adrian
Raducha Michał
Promotor: dr inż. Janusz Rafałko
Październik 2023

Spis treści

1. Cel biznesowy:

Celem jest stworzenie aplikacji do wsparcia zarządzaniem pracy Działu Utrzymania Ruchu w zakładzie produkcyjnym. Pracownikom działu ma udostępnić dziennik napraw maszyny oraz wcześniej działające rozwiązania usterek. Pozwolić na wgląd do magazynu części zamiennych oraz planowania cyklu konserwacji. Kierownikowi i szefowi produkcji aplikacja da dostęp do aktualnego stanu maszyn i pokazać potrzeby zaopatrzeniowe, co pozwoli na lepsze planowanie produkcji. Działania te powinny zwiększyć efektywność operacyjną, poprzez zapewnienie szybkiego dostępu do kluczowych informacji. Aplikacja może poprawić komunikację między różnymi poziomami zarządzania, umożliwiając szybkie zgłaszanie problemów, monitorowanie postępów i reagowanie na bieżące wyzwania.

2. Historia zmian:

Data	Autor	Opis	Wersja
15.10.2023r.	Michał Raducha	Cel biznesowy, wymagania funkcjonalne i niefunkcjonalne.	1.0
17.11.2023r.	Piotr Chudziaszek	Technologie, historia zmian	1.1
17.11.2023r.	Alan Darmetko	Harmonogram	1.2
17.11.2023r.	Michał Raducha	Metodyka, podział pracy w zespole	1.3
26.01.2024r.	Piotr Chudziaszek	Model systemu	1.4

3. Wymagania funkcjonalne:

3.1. Logowanie i autoryzacja:

Użytkownicy (automatycy, kierownik, właściciel zakładu) muszą móc zalogować się do aplikacji i być autoryzowani do dostępu tylko do odpowiednich funkcji.

3.2. Stan maszyn:

Automatycy powinni mieć możliwość aktualizacji stanu maszyny (sprawna, w trakcie naprawy, brak części do naprawy, niesprawna) i oznaczania ich odpowiednim kolorem.

3.3. Śledzenie Inwentarza:

System powinien umożliwiać użytkownikom sprawdzić stan magazynu i zlokalizować części potrzebne do naprawy maszyn oraz monitorowanie poziomów zapasów w czasie rzeczywistym, wraz z funkcją automatycznego powiadamiania o niskich stanach zapasów.

3.4. Dziennik napraw maszyn:

Automatycy powinni móc prowadzić dziennik napraw maszyn, zawierający informacje o przeprowadzonych naprawach.

3.5. Raportowanie i Analityka:

Zestaw narzędzi do tworzenia spersonalizowanych raportów dotyczących konserwacji i napraw.

3.6. Różne poziomy dostępu:

Kierownik może sprawdzić aktualny stan maszyn stan magazynu, zapotrzebowanie na części i dodawać nowe części do magazynu. Właściciel zakładu może przeglądać aktualny stan maszyn.

4. Wymagania niefunkcjonalne:

4.1. Wydajność:

Aplikacja powinna działać wydajnie zarówno w wersji mobilnej, jak i desktopowej. Baza danych musi być zoptymalizowana, aby zapewnić płynne działanie.

4.2. Bezpieczeństwo:

Aplikacja musi spełniać wysokie standardy bezpieczeństwa, w tym bezpieczne uwierzytelnianie i dostęp do danych tylko dla uprawnionych użytkowników, wykonywane regularne kopie zapasowe.

4.3. Dostępność:

Aplikacja powinna być dostępna 24/7 z minimalnym czasem przestoju. Aplikacja powinna być dostępna na różnych platformach, w tym na urządzeniach mobilnych i desktopowych.

4.4. Skalowalność:

System powinien być skalowalny, aby mógł obsłużyć rosnącą liczbę maszyn i użytkowników wraz z rosnącymi potrzebami zakładu.

4.5. Użyteczność:

Interfejs użytkownika powinien być intuicyjny i łatwy w użyciu, z minimalnym progiem wejścia dla nowych użytkowników, aby zapewnić efektywną pracę użytkowników.

4.6. Kompatybilność:

Aplikacja powinna być kompatybilna z różnymi urządzeniami i systemami operacyjnymi, w tym komputerami stacjonarnymi, laptopami, tabletami i smartfonami.

5. Technologie

Projekt ten realizowany jest jako aplikacja klient-serwer z wykorzystaniem języka Python. Wybór Pythona jako języka programowania, Oracle Database jako systemu zarządzania bazą danych oraz technologii socket do komunikacji klient-serwer zapewnia solidne fundamenty dla efektywnego funkcjonowania aplikacji. Użycie Visual Studio Code jako narzędzia programistycznego oraz Ubuntu Server w środowisku produkcyjnym na platformie VMware stanowi optymalne rozwiązanie dla potrzeb projektu.

5.1. Język Programowania:

Aplikacja, zarówno w części klienta, jak i serwera, zostanie zaimplementowana w języku Python, ze względu na jego wszechstronność i wsparcie dla aplikacji sieciowych.

5.2. Baza Danych:

Wykorzystanie Oracle Database zapewnia wydajne zarządzanie dużymi ilościami danych, wysoki poziom bezpieczeństwa i niezawodności.

5.3. Komunikacja Klient-Serwer:

Komunikacja między klientem a serwerem będzie realizowana przy użyciu technologii socket bezpośrednio zaimplementowanej jako metoda wysokopoziomowa w Python, co umożliwi bezpośrednią i szybką wymianę danych między różnymi komponentami systemu.

5.4. Frontend

Kivy, używane do tworzenia interfejsu użytkownika, umożliwi tworzenie przenośnych i konfigurowalnych UI, kompatybilnych z wieloma platformami, w tym z Androidem.

5.5. Bezpieczeństwo i Kryptografia:

PyCryptodome zostanie wykorzystane do obsługi algorytmów kryptograficznych, co jest kluczowe dla zapewnienia bezpieczeństwa transmisji danych. Jego zaletami są rozbudowane wsparcie dla różnych algorytmów oraz kompatybilność z wieloma platformami.

5.6. Narzędzia Programistyczne:

Visual Studio Code, jako główne narzędzie do tworzenia i edycji kodu, wspiera Pythona, Kivy oraz integrację z narzędziami kryptograficznymi.

5.7. Środowiska Testowe:

Indywidualne środowiska dla członków zespołu umożliwiają niezależne testowanie i rozwój.

5.8. Środowisko Produkcyjne:

a) Wykorzystanie platformy wirtualizacyjnej VMware do postawienia środowiska produkcyjnego gwarantuje łatwość zarządzania, izolację oraz skalowalność systemu.

b) Aplikacja serwerowa zostanie uruchomiona na systemie Ubuntu Server, co zapewnia odpowiednią stabilność i wydajność dla potrzeb produkcyjnych.

c) Aplikacja kliencka zostanie uruchomiona na systemie MS Windows.

d) Środowisko produkcyjne będzie zawierać:

- Oracle Database – dla efektywnego zarządzania danymi,
- Python – jako język programowania dla logiki aplikacji

6. Metodyka

6.1. Spotkania Zespołu:

Spotkania zespołu odbywają się raz w tygodniu online, korzystając z platformy Zoom. Podczas spotkania każdy członek zespołu przedstawia raport dotyczący postępów oraz ewentualnych problemów. Następnie, wspólnie podejmujemy decyzje w sprawie rozwiązania napotkanych trudności, planujemy zadania na najbliższy tydzień, a także ustalamy wstępne założenia na kolejne dwa tygodnie. Spotkanie kończymy aktualizacją tablicy zadań w projekcie na platformie GitHub.

6.2. Decyzje i Postępy, Raportowanie:

Decyzje są podejmowane przez lidera zespołu po wysłuchaniu i konsultacji z członkami zespołu. Zespół, składający się z trzech osób, korzysta z tablicy zadań na GitHubie(https://github.com/users/Piotr029/projects/3) do śledzenia postępów i koordynacji pracy. Informacje o postępach, problemach oraz zadaniach znajdują się w tej tablicy, co ułatwia bieżące śledzenie działań zespołu.

6.3. Rola Lidera:

Lider zespołu, którym jest Raducha Michał odpowiada za koordynację zadań i tworzenie ogólnej wizji projektu, zachowując otwartość na pomysły i problemy zgłaszane przez pozostałych członków zespołu.

6.4. Problemy:

Napotkane problemy są zgłaszane na platformie Discord projektu oraz w sekcji Issues na GitHubie. Precyzyjny opis problemu umożliwia szybkie zrozumienie sytuacji, a rozwiązanie jest ustalane podczas dodatkowego spotkania online.

6.5. Zmiany Planu:

Dzięki niewielkiemu rozmiarowi zespołu, jesteśmy elastyczni i otwarci na szybkie zmiany w planie pracy.

6.6. Technologie wykorzystane w Metodyce Pracy:

Do komunikacji i zarządzania projektem wykorzystujemy platformy Zoom, Discord oraz GitHub.

6.7. Metody modelowania:

a) UML (Unified Modeling Language):

Wykorzystanie diagramów klas, diagramów przypadków użycia, diagramów sekwencji itp. do reprezentowania struktury i zachowania aplikacji.

b) SQL Data Modeler:

Projektowanie struktury bazy danych

c) Figma:

Stworzenia mockupów interfejsu użytkownika, aby wizualizować planowany wygląd aplikacji.

6.8. Procesy wytwarzania oprogramowania

Projekt na platformie GitHub (https://github.com/Piotr029/AL_PZ1_2023-24_nst_zespol_nr2) został starannie podzielony na poszczególne katalogi, co znacznie ułatwia nawigację po projekcie i umożliwia efektywne wspólne działanie wielu osób na jednym projekcie. Kod został zorganizowany w łatwo importowane moduły, co nie tylko zwiększa czytelność, ale również ułatwia zarządzanie funkcjonalnością projektu. W ramach implementacji, projekt konsekwentnie przestrzega zasad określonych w PEP8, co pozwala utrzymać spójność i czytelność kodu. Dodatkowo, w celu ułatwienia zrozumienia trudnych fragmentów kodu oraz kluczowych decyzji projektowych, w kodzie umieszczono liczne i czytelne komentarze.

Proces doskonalenia jakości kodu jest wspierany poprzez regularne przeglądy kodu przez zespół. Ten praktykowany mechanizm pozwala uzyskać różnorodne perspektywy na kod projektu, co z kolei przyczynia się do poprawy jakości implementacji. W ten sposób, projekt nie tylko skupia się na efektywnej organizacji i implementacji, ale także na utrzymaniu wysokich standardów jakości kodu i współpracy zespołowej.

7. Podział Pracy w Zespole

Nasz zespół składa się z trzech zaangażowanych członków, z każdym z nas pełniącym specyficzną rolę w projekcie:

Raducha Michał - Lider Zespołu

Zakres Pracy:

- Odpowiedzialny za koordynację działań zespołu oraz utrzymanie ogólnej wizji projektu.
- Backend aplikacji zarówno po stronie serwera, jak i klienta.
- Aktywny udział w procesie decyzyjnym i planowaniu rozwoju projektu.

Chudziaszek Piotr - Specjalista ds. Serwera i Bazy Danych

Zakres Pracy:

- Implementacja i utrzymanie serwera aplikacji.
- Projektowanie i zarządzanie bazą danych.
- Współpraca z zespołem w celu zoptymalizowania działania backendu.

Darmetko Alan - Frontend/UI Developer

Zakres Pracy:

- Tworzenie interfejsu użytkownika (UI) oraz odpowiedzialność za frontend aplikacji.
- Współpraca z zespołem w celu zapewnienia spójności między interfejsem a funkcjonalnościami backendu.
- Odpowiedzialny za atrakcyjny i intuicyjny design interfejsu użytkownika.

Nasz zespół, złożony z różnorodnych umiejętności i doświadczeń, dąży do efektywnej współpracy, zapewniając kompleksową realizację projektu. Podział zadań uwzględnia specjalizacje każdego członka zespołu, co przyczynia się do skutecznej i efektywnej implementacji poszczególnych komponentów projektu.

8. Harmonogram

Zadanie	Termin	Wykonawca
Planowanie i tworzenie dokumentacji wstępnej	Koniec października	Wszyscy
Dobór systemu operacyjnego serwera	Koniec października	Chudziaszek Piotr
Modelowanie systemu	Koniec października	Raducha Michał
Wykonanie Modelu Logicznego Bazy Danych	Koniec października	Chudziaszek Piotr
Stworzenie rozszerzonych diagramów UML	Koniec października	Darmetko Alan
Tworzenie skryptów bazy danych	Listopad	Chudziaszek Piotr
Kod Pl/SQL obsługi BD	Listopad	Chudziaszek Piotr
Kod polacznie klient-serwer	Listopad	Raducha Michał
Kod aplikacji serwera	Listopad	Raducha Michał
Tworzenie Szkicu UI	Listopad	Darmetko Alan
Przygotowanie UI	Listopad	Darmetko Alan
Integracja komponentów aplikacji	Grudzień	Raducha Michał
Testy integracyjne	Grudzień	Darmetko Alan
Przygotowanie środowiska produkcyjnego	Grudzień	Chudziaszek Piotr
Integracja z zewnętrznymi systemami	Styczeń	Darmetko Alan
Wdrożenie systemu do środowiska produkcyjnego	Styczeń	Chudziaszek Piotr
Zbieranie i analiza opinii użytkowników	Styczeń	Raducha Michał
Refaktoryzacja i optymalizacja kodu	Styczeń	Raducha Michał

9. Model systemu.

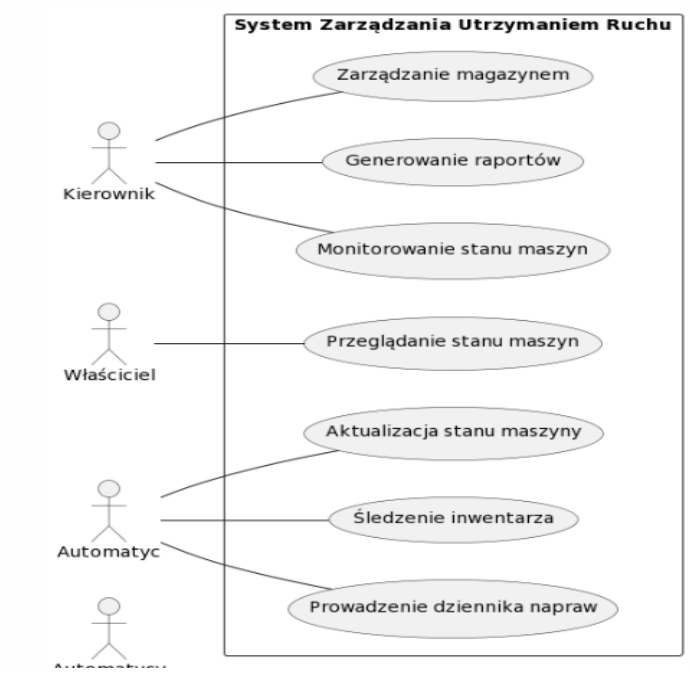
9.1. Ogólna architektura systemu

System aplikacji do zarządzania Działem Utrzymania Ruchu składa się z dwóch głównych komponentów: serwera i klienta. Serwer, zaimplementowany w Pythonie, zarządza bazą danych, obsługuje żądania klienta i realizuje logikę. Klient, również stworzony w Pythonie, zapewnia interfejs użytkownika i komunikuje się z serwerem za pomocą protokołu socket.

9.2. Środki implementacji

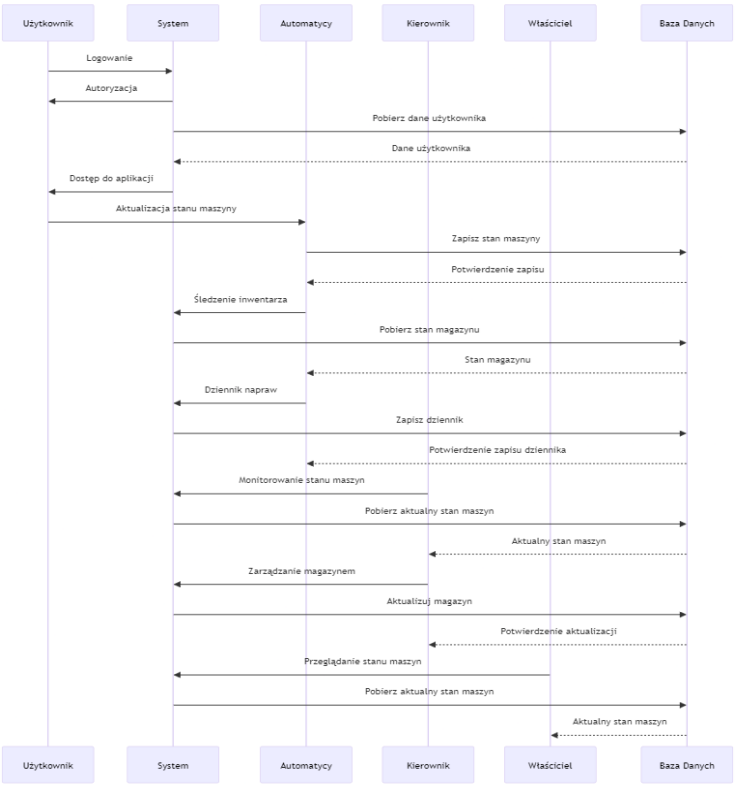
Język programowania: Python.
Baza danych: Oracle Database.
Technologie: Socket dla komunikacji sieciowej, Tkinter i Kivy dla GUI, PyCryptodome dla kryptografii.
Narzędzia: Git dla kontroli wersji, Zoom i Discord dla komunikacji w zespole.
System operacyjny serwera: CentOS 7 x64
System operacyjny klienta: MS Windows

9.3. Przypadki użycia

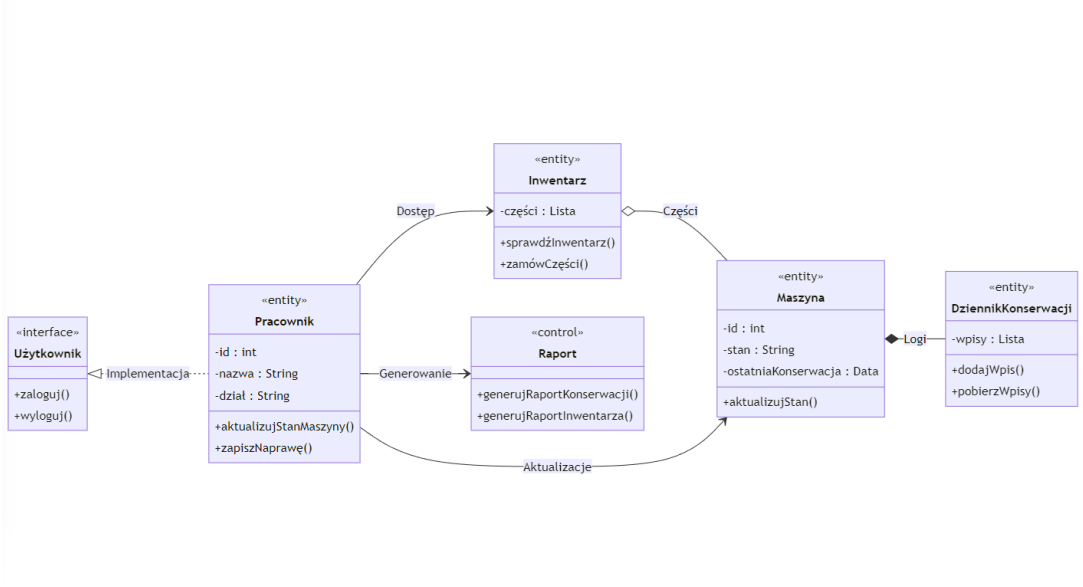


Rysunek 1 - Diagram przypadków użycia

9.4. Diagramy

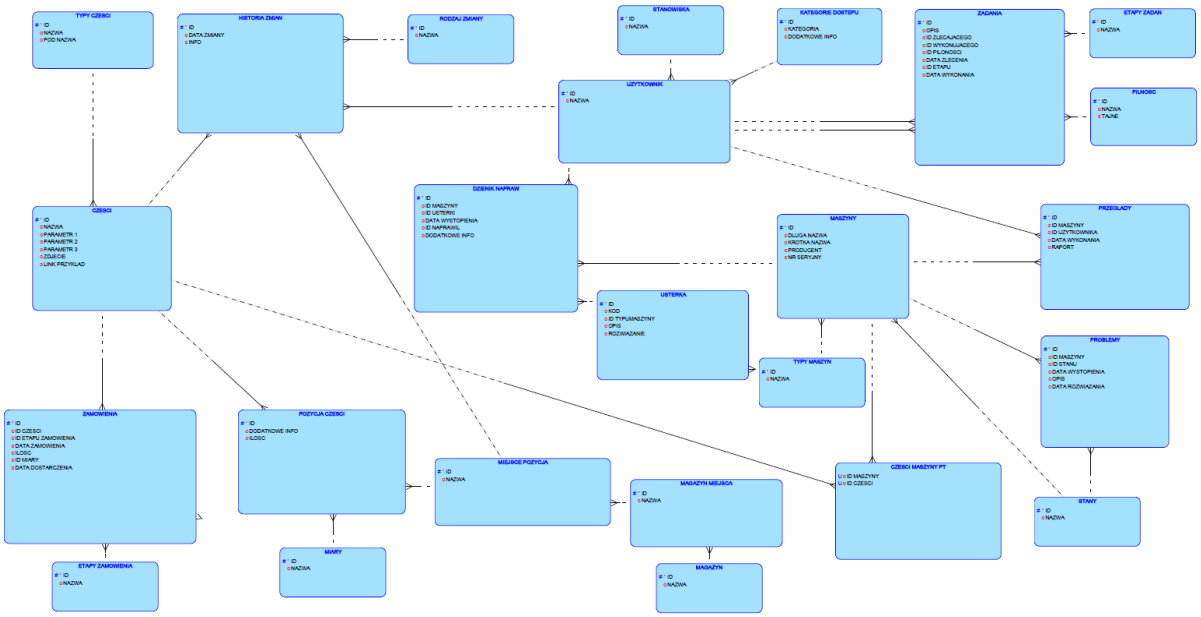


Rysunek 2 - Diagram sekwencji

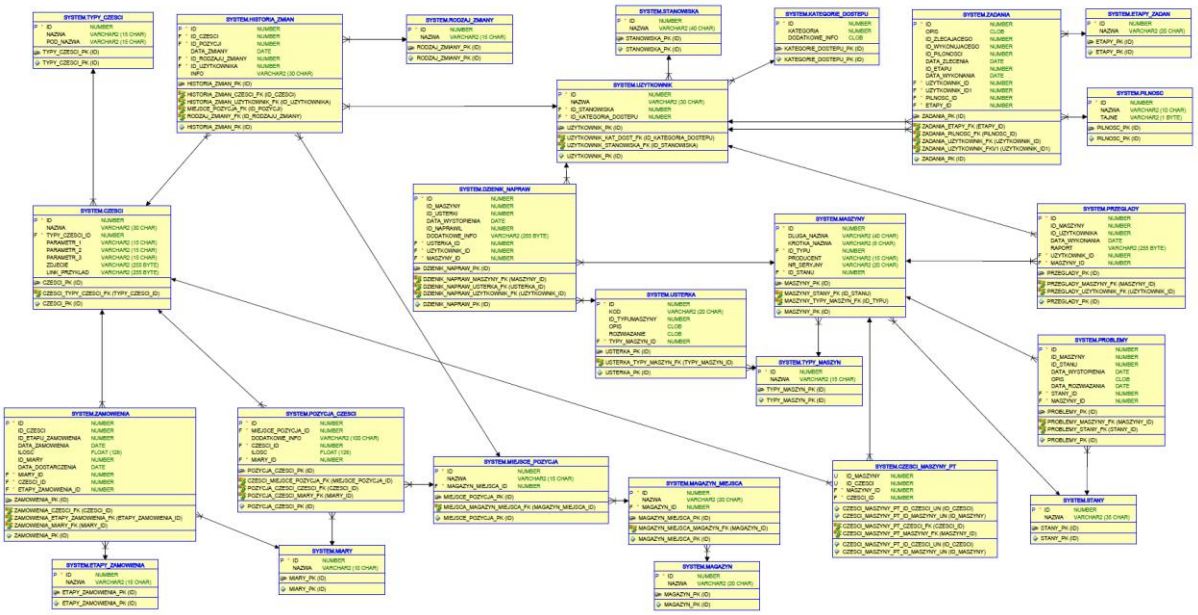


Rysunek 3 - Diagram klas

9.5. Struktura bazy danych



Rysunek 4 - Model logiczny bazy danych

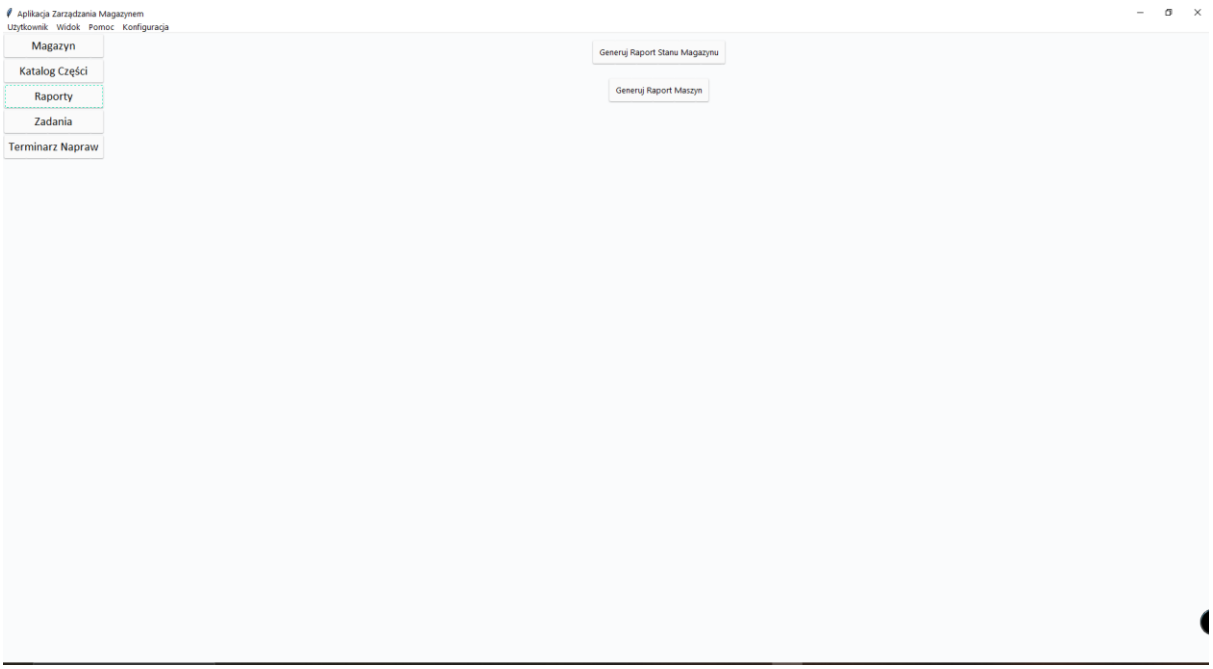


Rysunek 5 - model relacyjny bazy danych

9.6. Opis poszczególnych modułów.

9.6.1. Okno główne aplikacji

Po prawidłowym zalogowaniu użytkownikowi wyświetla się główne okno aplikacji. Główne okno aplikacji pozwala użytkownikowi na przejście do odpowiedniego modułu aplikacji takiego jak: magazyn, katalog części, raporty, zadania, terminarz napraw. Dodatkowo są funkcjonalności raportów stanu magazynu.



Rysunek 6 - zrzut ekranu - Okno główne aplikacji

9.6.2. Magazyn

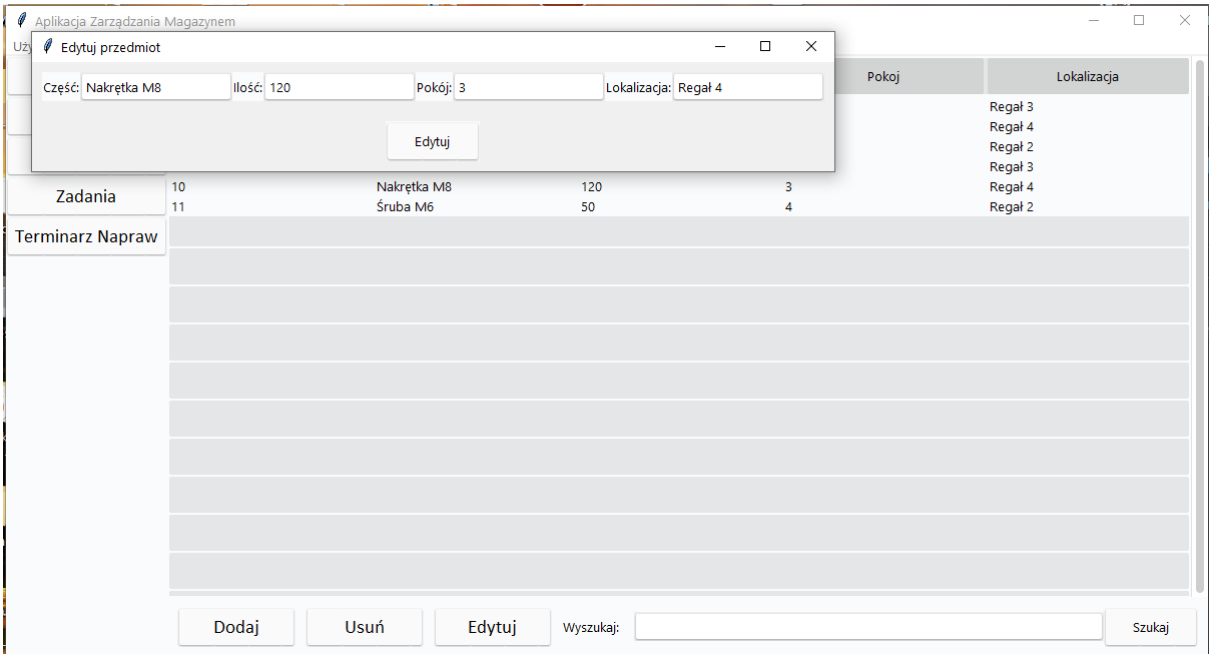
Po przejściu do modułu magazyn, użytkownikowi wyświetla się w oknie głównym spis przedmiotów wraz z ich umiejscowieniem w magazynie. Funkcjonalności jakie oferuje ten moduł to:

Dodaj – użytkownik ma możliwość dodania przedmiotu.

Usuń – użytkownik ma możliwość usunięcia przedmiotu.

Edytuj – użytkownik ma możliwość edycji przedmiotu.

Wyszukaj – użytkownik ma możliwość wyszukania przedmiotu.

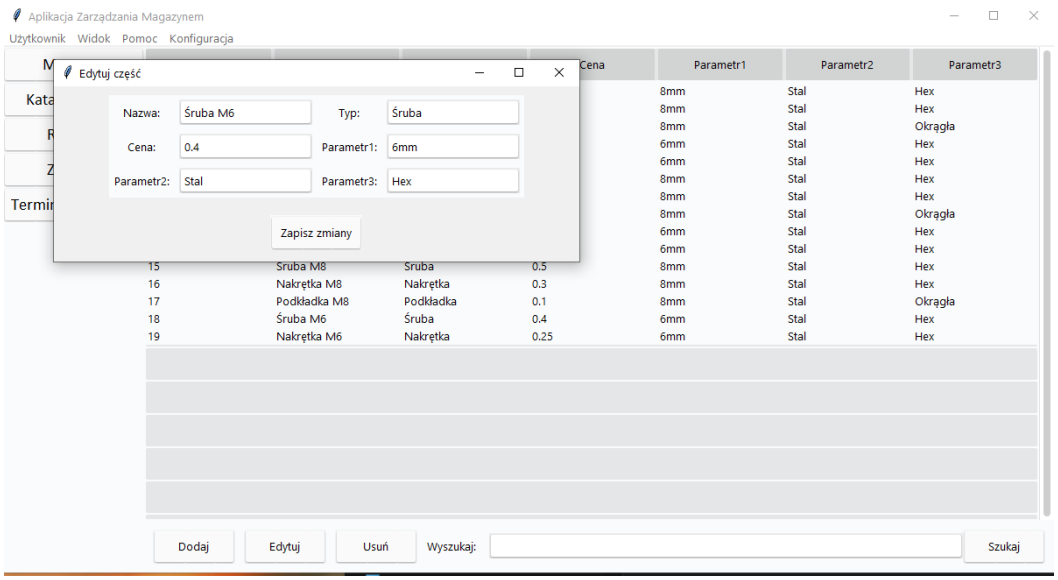


Rysunek 7 - zrzut ekranu - Edycja przedmiotu w module magazyn

9.6.3. Katalog części

Po przejściu do modułu katalog części, użytkownikowi wyświetla się w oknie głównym spis części wraz z ich nazwą, typem, ceną. Funkcjonalności jakie oferuje ten moduł to:

Dodaj – użytkownik ma możliwość dodania części.
Usuń – użytkownik ma możliwość usunięcia części.
Edytuj – użytkownik ma możliwość edycji części.
Wyszukaj – użytkownik ma możliwość wyszukania części.

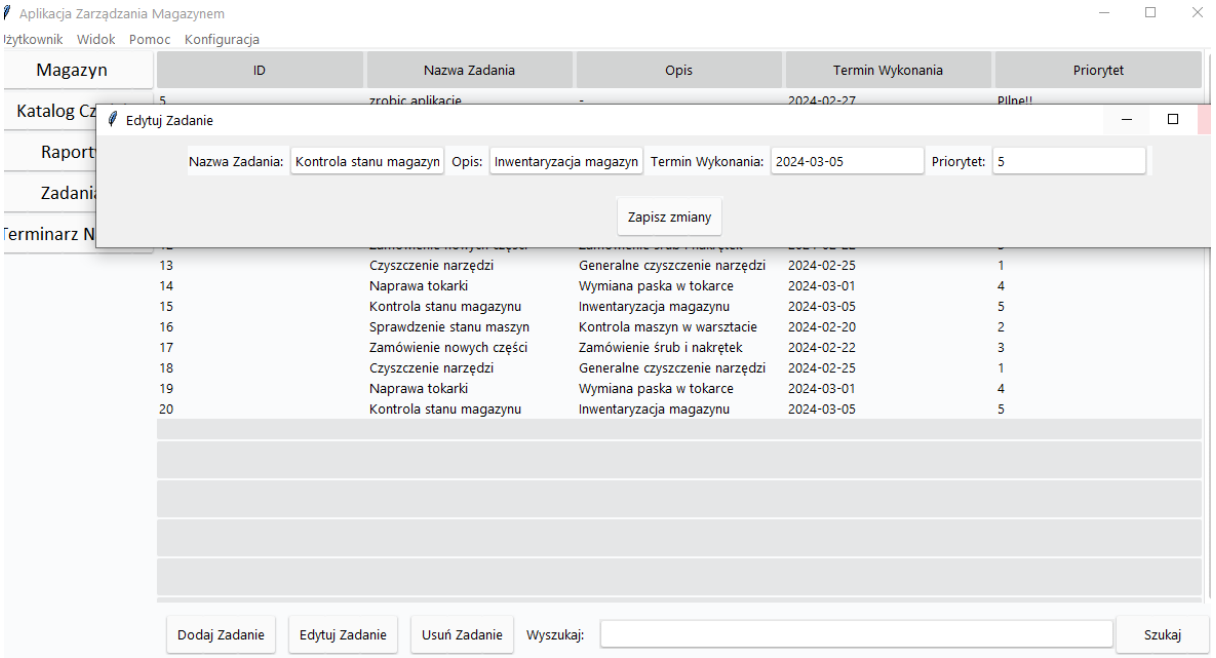


Rysunek 8 - zrzut ekranu - Edycja części w module katalog części

9.6.4. Zadania

Po przejściu do modułu zadania, użytkownikowi wyświetla się w oknie głównym lista zadań wraz z ich nazwą, opisem, terminem wykonania oraz priorytetem. Funkcjonalności jakie oferuje ten moduł to:

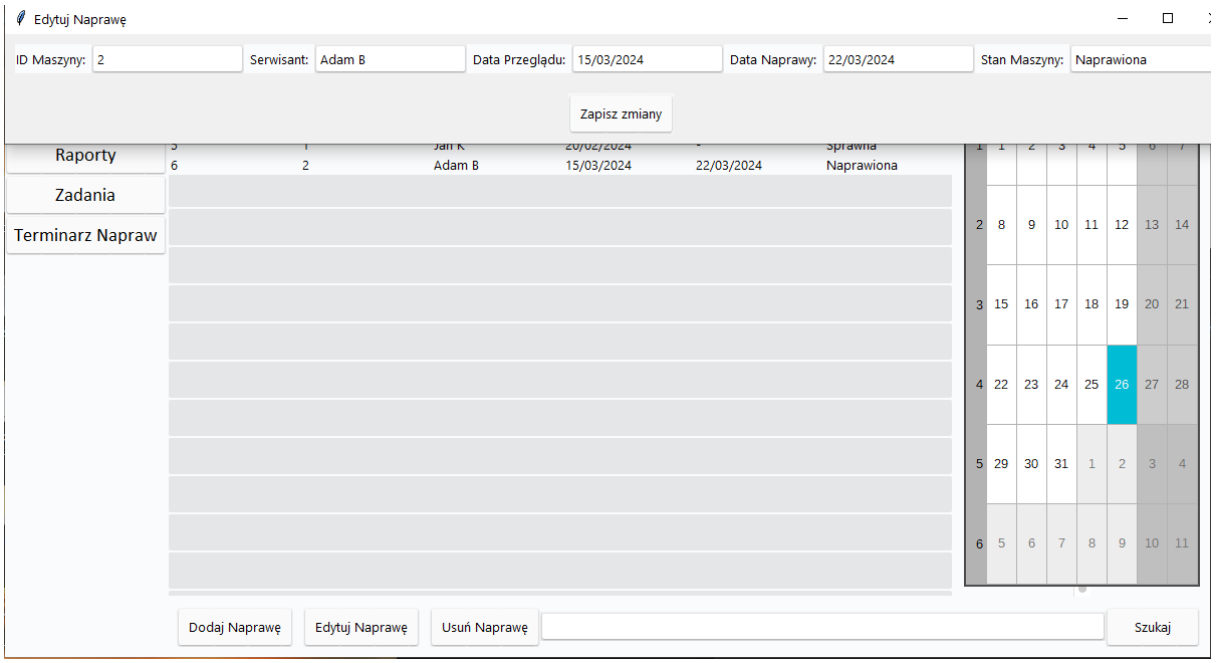
Dodaj – użytkownik ma możliwość dodania zadania.
Usuń – użytkownik ma możliwość usunięcia zadania.
Edytuj – użytkownik ma możliwość edycji zadania.
Wyszukaj – użytkownik ma możliwość wyszukania zadania.



Rysunek 9 - zrzut ekranu - Edycja zadania w module zadania

9.6.5. Terminarz napraw

Po przejściu do modułu terminarz napraw, użytkownikowi wyświetla się w oknie głównym lista przeprowadzonych napraw wraz z id maszyny która wymagała naprawy, serwisant który naprawę przeprowadzał, datą przeglądu i naprawy oraz stan maszyny. Dodatkowo w prawej części okna wyświetla się terminarz w formie graficznej. Funkcjonalności jakie oferuje ten moduł to:
Dodaj – użytkownik ma możliwość dodania naprawy.
Usuń – użytkownik ma możliwość usunięcia naprawy.
Edytuj – użytkownik ma możliwość edycji naprawy.
Wyszukaj – użytkownik ma możliwość wyszukania naprawy.



Rysunek 10 - zrzut ekranu - Edycja naprawy w module terminarz napraw