**Dokumentacja techniczna**

**Cel projektu:**

**Aplikacja do zgłaszania usterek mechanicznych w pojeździe samochodowym**

**Charakterystyka systemu**

**Specyfikacja i analiza wymagań**

***Aplikacja webowa Usterki Mechaniczne***

***Użytkownicy – aktorzy role :***

1. ***Kierownik***
2. ***Logistyk (sekretarka)***
3. ***Księgowa***
4. ***Mechanik (nie więcej niż 10)***

***Wewnętrzny system dla pracowników serwisu mechanicznego, ma na starcie panel logowania - login, hasło szyfrowane. Jeden administrator – „kierownik” ma możliwość dodawania i edycji kont. Ma pełny podgląd aktywności pracowników. Logistyk posiada możliwość operowania trybem zgłaszania usterek. Mechanik (zwykły użytkownik) widzi po zalogowaniu swój harmonogram. Każdy z użytkowników ma wysublimowany kalendarz połączony z bazą danych pgSql. Ujmuje on dni oraz godziny pracy dla poszczególnych mechaników oraz obsługę klienta. Jest połączony z odpowiednią reakcją na zdarzenia na panelu. Szata graficzna ma w łatwy sposób sugerować dostępność w pracy mechaników (zajęte godziny wizytami klientów czerwone okienko, wolne zielone) oraz analogicznie pokazywać mechanikom na ich panelu dni wolne oraz robocze. Każdy z aktorów ma panel wyposażony w odpowiednią wyszukiwarkę klientów i usterek. Dodatkowo dostępna jest ogólna wyszukiwarka danych usług po dacie oraz obsłużonych pojazdach [nr rejestracyjne].***

***Język projektu:***

***Java***

***Wykorzystywana technologia:***

***- Java Spring Framework***

***- pgSql***

***Oprogramowanie:***

***Spring Tool Suite***

Korzystamy z wersji SPRING BOOT SECURITY (m.in. ze względu na możliwość użycia specjalnego pola roles które jest bardzo funkcjonalne przy różnych uprawnieniach dla różnych kont).

***W projekcie trzymamy się zasad:***

* ***Solid***
* [Zasada jednej odpowiedzialności](http://pl.wikipedia.org/wiki/Zasada_jednej_odpowiedzialno%C5%9Bci) (S)
* [Zasada otwarte-zamknięte](http://pl.wikipedia.org/wiki/Zasada_otwarte-zamkni%C4%99te) (O)
* [Zasada podstawienia Liskov](http://pl.wikipedia.org/wiki/Zasada_podstawienia_Liskov) (L)
* [Zasada segregacji interfejsów](http://pl.wikipedia.org/w/index.php?title=Zasada_segregacji_interfejs%C3%B3w&action=edit&redlink=1) (I)
* [Zasada odwrócenia zależności](http://pl.wikipedia.org/w/index.php?title=Zasada_odwr%C3%B3cenia_zale%C5%BCno%C5%9Bci&action=edit&redlink=1) (D)
* ***Dry***

Don’t Repeat Yourself – reguła zalecająca unikanie różnego rodzaju powtórzeń wykonywanych przez programistów - na przykład unikanie tych samych czynności podczas kompilowania, unikanie wklejania (lub pisania) tych samych (lub bardzo podobnych) fragmentów kodu w wielu miejscach

* ***Kiss***

Keep It Simple, Stupid – zasada utrzymania projektu w sposób prosty i zrozumiały dla każdego, jej istotą jest dążenie do utrzymania eleganckiej i przejrzystej [struktury](http://pl.wikipedia.org/wiki/Struktura)

***metod:***

* ***Refactoringu***
* ***Continous Integration.***

***Korzystamy z metodyki projektowania systemów informatycznych Scrum przechodząc przez poszczególne fazy projektowania:***

1. ***Analiza wymagań użytkownika i modelowanie systemu.***
2. ***Projektowanie systemu (zgodnie z rozpiską sprintów).***
3. ***Implementacja systemu.***
4. ***Testowanie systemu.***
5. ***Pielęgnacja i dalszy rozwój.***

**Kierownik**

|  |
| --- |
| Dodawanie kont wszystkich pracowników |
| Edycja kont, ustawianie haseł i loginów pracowników |
| Ustawia: imię, nazwisko, stanowisko, godziny pracy, urlopy, dostępność |
| Może podglądnąć wszystkie operacje wykonywane przez każdego pracownika |
| Widzi wszystkie opcje z każdego konta |
| Po wejściu na stronę jest panel logowania i wyniki zależne od stanowiska |

**Logistycy (sekretarki)**

|  |
| --- |
| Po zalogowaniu widzi 2 tryby |
| 1. tryb - dane: imię, nazwisko, stanowisko, kalendarz z godzinami pracy, urlopami |
| 2. tryb - zgłaszanie danej awarii : |
| Wyszukiwarka klientów po nazwisku i nr. rejestracyjnym pojazdu |
| Gdy jest klient, z jego konta ustala się nową wizytę, gdy nie jest, dodaje się go |
| Z konta klienta ustala nową wizytę (pole tekstowe) |
| Wybiera się z kalendarza dzień, w którym ma być wizyta |
| W wybranym dniu jest lista mechaników (Zielony-wolny, Czerwony-Zajęty) |
| Wybiera się mechanika, ten automatycznie w bazie robi się Czerwony i akceptuje wizytę |

**Mechanicy**

|  |
| --- |
| Widzi harmonogram (kalendarz), dni kiedy ma wolne (czerwone) i kiedy ma być w pracy(zielone) |
| Zielony dzień jest klikalny, widzi tam rozpiskę godzinową oraz: |
| \*nieedytowalne pole z notatkami sekretarki |
| \*pola do uzupełnienia: cena swojej naprawy, cena za części, textbox- co zrobił |
| \*Dwa przyciski: Zapisz i Nie udało się wykonać naprawy |

**Księgowe**

|  |
| --- |
| Dokładna wyszukiwarka usług, sortująca po: |
| Nazwisku, nr. rejestracyjnym, po cenie usługi, za części i sumie, operatory: < >= |
| Jak wyszuka dane, pokazuje jej się opcja zaznaczenia tych znalezionych osób |
| Po zaznaczeniu uaktywnia się przyciski: Zapisz do pdf i zapisuje się na komputerze |

**Wszyscy**

|  |
| --- |
| Każdy ma zakładkę z wyszukiwarką wykonanych usług i może szukać po: |
| dacie (od -do), nr. rejestracyjnym, nazwisku klienta, nazwisku mechanika (Można łączyć) |

**Sprawy techniczne**

|  |
| --- |
| Stworzenie panelu logowania |
| Stworzenie tabel w bazie danych |
| Stworzenie bazy danych, podpięcie jej |

**User Stories**

* Jako niezalogowany użytkownik mogę wyświetlić panel logowania.
* Po zalogowaniu się do systemu widzę swój panel użytkownika.
* Jako administrator mogę przejść do panelu dodawania nowych użytkowników, aby zarejestrować konta pracowników.
* Po podaniu odpowiednich danych mogę dodać nowego użytkownika lub anulować dodawanie pracownika.
* Jako dowolny aktor widzę na panelu po zalogowaniu odpowiednio swój kalendarz
* Jako sekretarka mogę wyszukiwać klientów, dzięki temu mogę dokonać weryfikacji czy dana osoba jest już klientem w bazie
* Jako administrator mogę dodawać godziny pracy mechanika