Dokumentacja projektu zaliczeniowego

Przedmiot: Inżynieria oprogramowania

Temat: Dziennik elektroniczny

Autorzy: Aleksander Grobicki-Madej, Maciej Wolf

Grupa: I1-222A

Kierunek: informatyka

Rok akademicki: 2016/2017

Poziom i semestr: I/4

Tryb studiów: stacjonarne

Należy pozostawić wszelkie nagłówki tego dokumentu, a umieszczać treść w odpowiednich miejscach zamiast obecnych objaśnień.

Stronę tytułową można sformatować w dowolny sposób, ale należy pozostawić zawartość informacyjną w układzie pokazanym powyżej.

Praca powinna zostać złożona wyłącznie w formacie pdf. Przed wygenerowaniem ostatecznej wersji należy zaktualizować spis treści – wyświetlane dwa poziomy.

Niniejszą informację należy również usunąć z wersji końcowej.

Spis treści

[2 Odnośniki do innych źródeł 4](#_Toc448735363)

[3 Słownik pojęć 5](#_Toc448735364)

[4 Wprowadzenie 6](#_Toc448735365)

[4.1 Cel dokumentacji 6](#_Toc448735366)

[4.2 Przeznaczenie dokumentacji 6](#_Toc448735367)

[4.3 Opis organizacji 6](#_Toc448735368)

[4.4 Analiza SWOT organizacji 6](#_Toc448735369)

[5 Specyfikacja wymagań 7](#_Toc448735370)

[5.1 Charakterystyka ogólna 7](#_Toc448735371)

[5.2 Wymagania funkcjonalne 7](#_Toc448735372)

[5.3 Wymagania niefunkcjonalne 8](#_Toc448735373)

[6 Zarządzanie projektem 9](#_Toc448735374)

[6.1 Zasoby ludzkie 9](#_Toc448735375)

[6.2 Etapy/kamienie milowe projektu 9](#_Toc448735376)

[6.3 Harmonogram prac 9](#_Toc448735377)

[7 Zarządzanie ryzykiem 10](#_Toc448735378)

[7.1 Lista czynników ryzyka 10](#_Toc448735379)

[7.2 Ocena ryzyka 10](#_Toc448735380)

[7.3 Plan reakcji na ryzyko 10](#_Toc448735381)

[8 Zarządzanie jakością 11](#_Toc448735382)

[8.1 Scenariusze testowe 11](#_Toc448735383)

[9 Projekt techniczny 12](#_Toc448735384)

[9.1 Opis architektury systemu 12](#_Toc448735385)

[9.2 Technologie implementacji systemu 12](#_Toc448735386)

[9.3 Diagramy UML 12](#_Toc448735387)

[9.4 Charakterystyka zastosowanych wzorców projektowych 12](#_Toc448735388)

[9.5 Projekt bazy danych 12](#_Toc448735389)

[9.6 Projekt interfejsu użytkownika 12](#_Toc448735390)

[9.7 Procedura wdrożenia 13](#_Toc448735391)

[10 Dokumentacja dla użytkownika 14](#_Toc448735392)

[11 Podsumowanie 15](#_Toc448735393)

[11.1 Szczegółowe nakłady projektowe członków zespołu 15](#_Toc448735394)

[12 Inne informacje 16](#_Toc448735395)

# Odnośniki do innych źródeł

<https://github.com/Grommuald/IOProjekt> - odnośnik do strony repozytorium z projektem i dokumentacją

# Słownik pojęć

Tabela lub lista z pojęciami, które wymagają wyjaśnienia, wraz z tymi wyjaśnieniami – w szczególności synonimy różnych pojęć używanych w dokumentacji.

# Wprowadzenie

## Cel dokumentacji

Dokumentacja zawiera szczegółowe informacje na temat całego projektu. Stanowi opis wszelkich obiektów i czynności występujących w systemie, jak również jako materiał do wglądu przez Prowadzącego z zajęć projektowych z Inżynierii Oprogramowania.

## Przeznaczenie dokumentacji

Dokumentacja kierowana jest zarówno do uczestników projektu, jak i do Prowadzącego przedmiot.

## Opis organizacji

System realizowany jest dla szkoły - jednostki oświatowo-wychowawczej zajmującej się, kształceniem i wychowaniem - zgodnie z podstawą programową i ustawą o oświacie. Szkoła opiera się na relacji nauczyciel-uczeń, co pozwala na systematyczne przekazywanie wiedzy ze strony nauczyciela, jak również relację zwrotną ze strony ucznia, w postaci odbioru przekazywanego materiału. Do prowadzenia zajęć niezbędny jest specjalny dokument zwany dziennikiem. W nim przechowywane są informacje o postępach każdego z uczniów, tj.: oceny, obecności czy uwagi. Pomaga to w weryfikacji postępów w nauce każdego ucznia, zarówno przez rodziców, jak i nauczycieli. Każdy z uczniów przydzielony jest również do określonej klasy – pomaga to w organizacji zajęć, dzieląc wszystkich uczniów uczęszczających do szkoły na mniejsze grupy. Co więcej, istnieje również podział ze względu na zaawansowanie w opanowanej wiedzy i wiek – stąd wyróżniamy podział na roczniki.

## Analiza SWOT organizacji

Wystarczy sama tabela 2x2 (silne-słabe-szanse-zagrożenia)

|  |  |
| --- | --- |
| Silne strony: cyfryzacja w dziedzinie edukacji | Szanse: pojawianie się nowych technologii na rynku edukacji |
| Słabe strony: zastój w technologii, w pełni analogowe zarządzanie placówką | Zagrożenia: reforma edukacyjna |

# Specyfikacja wymagań

## Charakterystyka ogólna

### Definicja produktu

System służy do komunikacji między uczniami a nauczycielami oraz rodzicami a nauczycielami. System dokumentuje w formie elektronicznej zajęcia lekcyjne w szkole.

### Cel biznesowy

- ułatwienie komunikacji między instytucją szkoły a rodzicami  
- ułatwiony wgląd w dokumenty  
- ułatwiony wgląd w postępy w edukacji uczniów  
- ułatwienie wypełniania obowiązków przez nauczycieli

### Użytkownicy

- **uczeń** - sprawdza oceny, komunikuje się z nauczycielem  
- **nauczyciel** - wystawia oceny, komunikuje się z uczniem i rodzicem, usprawiedliwia nieobecności ucznia, sprawdza obecność, wpisuje uwagi  
- **administrator systemu** - udostępnia plan lekcji, przydziela nauczycieli do klas, zarządza systemem  
- **rodzic** - sprawdza oceny, komunikuje się z nauczycielem, usprawiedliwia nieobecności ucznia

### Korzyści z systemu

1. Korzyści  
 1.1 - uczeń - wie, kiedy ma odwołane zajęcia, zna swoje oceny, zawsze zna swoją aktualną średnią ocen  
 1.2 - rodzic - może na bieżąco kontrolować oceny swojego dziecka  
 1.3 - nauczyciel - ma łatwiejszy kontakt z uczniami i rodzicami, łatwiej prowadzi statystykę uczniów

### Ograniczenia projektowe i wdrożeniowe

TODO:  
- sale lekcyjne muszą być wyposażone we własne komputery  
- kadra musi być przeszkolona w zakresie korzystania z komputera i systemu

## Wymagania funkcjonalne

### Lista wymagań

lista numerowana – czyli lista przypadków użycia

### Diagramy przypadków użycia

Tutaj same diagramy – bez specyfikacji, ale każdy diagram z tytułem i na osobnej stronie

### Szczegółowy opis wymagań

każde na nowej stronie wg następujących punktów:

1. System

* Numer – jako ID
* Nazwa
* Uzasadnienie biznesowe – odwołanie (-a) do elementów wymienionych w 5.1.4. (id i treść elementu, do którego się odwołujemy)
* Użytkownicy
* Warunki początkowe
* Przebieg działań
* Efekty – warunki końcowe
* Wymagania niefunkcjonalne – szczegółowe wobec poszczególnych wymagań funkcjonalnych
* Częstotliwość - na skali 1-5 lub BN-BW
* Istotność – inaczej: zależność krytyczna, znaczenie - na skali 1-5 lub BN-BW

***Ważne!***

*Elementy od warunków początkowych do końca mogą być grupowane, tj. specyfikacja pojedynczego przypadku użycia może zawierać:*

*- pojedynczy przebieg działań (scenariusz główny) oraz ew. scenariusze alternatywne, albo*

*- wiele przebiegów głównych wraz z ew. scenariuszami alternatywnymi – wtedy każdy z przebiegów głównych powinien być opisany wg tych punktów (od warunków początkowych do końca).*

## Wymagania niefunkcjonalne

1. Wydajność   
   **-** system powinien zapewnić́ stały i stabilny dostęp do elektronicznego dziennika
2. Bezpieczeństwo  
   - system powinien zapewnić tworzenie kopii zapasowych informacji w razie błędów związanych z niepoprawnym wprowadzeniem informacji przez osobę z niego korzystającą
3. Zabezpieczenia  
   -- system powinien zapewnić tworzenie kopii zapasowych informacji w razie nieprzewidzianej awarii związanej z działaniem czynników zewnętrznych
4. Inne cechy jakości  
   - system powinien być dostępny z każdego urządzenia z dostępem do Internetu

# Zarządzanie projektem

## Zasoby ludzkie

(rzeczywiste lub hipotetyczne) – przy realizacji projektu

## Etapy/kamienie milowe projektu

wg kaskadowego, przyrostowego lub innego cyklu życia

## Harmonogram prac

wraz ze wskazaniem, co jest warunkiem odbioru danego etapu i przejścia do następnego

obejmuje również Harmonogram wdrożenia projektu – np. szkolenie, rozruch, konfiguracja, serwis – może obejmować różne wydania (tj. o różnej funkcjonalności – personal, professional, enterprise) i wersje (1.0, 1.5, itd.)

# Zarządzanie ryzykiem

## Lista czynników ryzyka

## Ocena ryzyka

prawdopodobieństwo i wpływ

## Plan reakcji na ryzyko

# Zarządzanie jakością

## Scenariusze testowe

szczegółowy plan testowania systemu – głównie testowanie funkcjonalności; każdy scenariusz od nowej strony wg następujących punktów:

* numer – jako ID
* nazwa scenariusza – co test w nim testowane (max kilka wyrazów)
* opis – dodatkowe opcjonalne informacje, które nie zmieściły się w nazwie
* tester - konkretna osoba lub klient/pracownik,
* termin – kiedy testowanie ma być przeprowadzane,
* narzędzia wspomagające – jeśli jakieś są używane przy danym scenariuszu
* przebieg działań – tabela z trzema kolumnami: lp. oraz opisującymi działania testera i systemu
* zestaw danych testowych – najlepiej w formie tabelarycznej – jakie konkretnie dane mają być użyte przez testera i zwrócone przez system w poszczególnych krokach przebiegu działań
* *przebieg lub zestaw danych testowych musi zawierać jawną informację o warunku zaliczenia testu*

# Projekt techniczny

## Opis architektury systemu

Architektura będzie opierać się na wzorcu MVC (Model-View-Controller), co ułatwi oddzielenie warstwy front-end od warstwy back-end systemu.

## Technologie implementacji systemu

1. Java – ponieważ jest to bardzo przenośne rozwiązanie. Poza tym oferuje szeroką bibliotekę potrzebnych narzędzi, tj. Java FX czy rozwiązania sieciowe. Jest również bardzo szeroko stosowane w projektach wykorzystujących programowanie zorientowane obiektowo.
2. Heroku – proste rozwiązanie dla małych projektów; oferujące wsparcie dla baz danych; działa na zasadzie chmury.

## Diagramy UML

każdy diagram ma mieć tytuł oraz ma być na osobnej stronie

diagramy przypadków użycia umieszczone w punkcie 5.2.2, a nie tutaj.

### Diagram(-y) klas

### Diagram(-y) czynności

### Diagramy sekwencji

co najmniej 5, w tym co najmniej 1 przypadek użycia zilustrowany kilkoma diagramami sekwencji

### Inne diagramy

co najmniej trzy – komponentów, rozmieszczenia, maszyny stanowej itp.

## Charakterystyka zastosowanych wzorców projektowych

**Model-View-Controller** – wzorzec służący do organizowania struktury aplikacji posiadających graficzne interfejsy użytkownika.   
**Model** – jest pewną reprezentacją problemu bądź logiki aplikacji,  
**View** – opisuje, jak wyświetlić pewną część modelu w ramach interfejsu użytkownika,  
**Controller** – przyjmuje dane wejściowe użytkownika i reaguje na jego poczynania, zarządzając aktualizacje modelu oraz odświeżanie widoków.

## Projekt bazy danych

### Schemat

w trzeciej formie normalnej; jeśli w innej to umieć uzasadnić wybór

### Projekty szczegółowe tabel

## Projekt interfejsu użytkownika

### Lista głównych elementów interfejsu

okien, stron, aktywności (Android)

### Projekty szczegółowe poszczególnych elementów

każdy element od nowej strony z następującą minimalną zawartością:

* numer – ID elementu
* nazwa – np. formularz danych produktu
* opis – dodatkowe opcjonalne informacje o przeznaczeniu, obsłudze – jeśli nazwa nie będzie wystarczająco czytelna
* projekt graficzny – wystarczy schemat w narzędziu graficznym lub zrzut ekranu – z przykładowymi informacjami (nie pusty!!!)
* wykorzystane dane – jakie dane z bazy danych są wykorzystywane
* opis działania – tabela pokazująca m.in. co się dzieje po kliknięciu przycisku, wybraniu opcji z menu itp.

## Procedura wdrożenia

jeśli informacje w harmonogramie nie są wystarczające (a zapewne nie są)

# Dokumentacja dla użytkownika

4-6 stron z obrazkami (np. zrzuty ekranowe, polecenia do wpisania na konsoli, itp.)

* pisana językiem odpowiednim do grupy odbiorców – czyli najczęściej nie do informatyków
* może to być przebieg krok po kroku obsługi jednej głównej funkcji systemu, kilku mniejszych, instrukcja instalacji lub innej pomocniczej czynności.

# Podsumowanie

## Szczegółowe nakłady projektowe członków zespołu

tabela (kolumny to osoby, wiersze to działania) pokazująca, kto ile czasu poświęcił na projekt oraz procentowy udział każdej osoby w danym zadaniu

# Inne informacje

przydatne informacje, które nie zostały ujęte we wcześniejszych punktach