Politechnika Poznańska
Wydział Informatyki
Instytut Informatyki

Praca dyplomowa inżynierska

TYTUŁ PRACY INŻYNIERSKIEJ

Jakub Rojek, 12345
Dawid Neumann, 12345
Wiesław Nowak, 12345
Wiktor Kierzek, 12345

Promotor
dr hab. inż. Imię Nazwisko

Poznań, 2013 r.

Tutaj przychodzi karta pracy dyplomowej;
oryginał wstawiamy do wersji dla archiwum PP, w pozostałych kopiach wstawiamy ksero.

Spis treści

Rozdział 1

Wprowadzenie

1.1 Opis problemu i koncepcja jego rozwiązania

Zgodnie z rozporządzeniem Ministerstwa Nauki i Szkolnictwa Wyższego [1,2] każda uczelnia wyższa w Polsce jest zobowiązana do monitorowania karier swoich absolwentów, w celu pozyskania informacji o ich aktualnej sytuacji zawodowej. Oprócz spełnienia wymogów formalnych, systematyczne gromadzenie i analizowanie danych o zatrudnieniu absolwentów umożliwia uczelniom weryfikację jakości i efektywności kształcenia na poszczególnych wydziałach i kierunkach. Zebrane informacje stanowią cenną wskazówkę w ciągłym procesie doskonalenia oferty dydaktycznej uczelni, pomagając w dostosowywaniu kierunków studiów i programów kształcenia do potrzeb rynku pracy. Prowadzenie rzetelnych badań na temat losów zawodowych absolwentów i prezentowanie statystyk zatrudnienia jest również istotne z punktu widzenia wizerunku uczelni.

Ministerstwo Nauki i Szkolnictwa Wyższego nie narzuca uczelniom sposobu realizacji procesu monitorowania karier absolwentów. Większość uczelni wyższych, w tym Politechnika Poznańska, wywiązuje się z tego obowiązku za pomocą badania ankietowego. Opracowane ankiety są rozsyłane do absolwentów w formie elektronicznych lub drukowanych formularzy bądź przeprowadzane za pośrednictwem rozmów telefonicznych. Rozwiązania te są nie tylko kosztowne i czasochłonne, lecz charakteryzują się także niewielką efektywnością. Z szacunków Centrum Praktyk i Karier Politechniki Poznańskiej wynika„ że z możliwości dobrowolnego wypełnienia ankiety absolwenckiej korzysta poniżej 10

W związku z wymienionymi wadami dotychczasowych metod zaproponowano stworzenie systemu informatycznego, w postaci aplikacji internetowej, który monitorowałby kariery zawodowe absolwentów wykorzystując dane udostępniane przez nich w serwisie LinkedIn. Serwis LinkedIn stanowi jedną z największych sieci zawodowych, łącząc ponad 250 mln. użytkowników w 200 krajach i terytoriach na całym świecie. Stworzony system powinien w założeniach zautomatyzować i uskutecznić proces monitorowania, pobierać dane o zatrudnieniu absolwentów z serwisu LinkedIn oraz prezentować je w formie raportów o z góry określonej strukturze. Należy przewidzieć również możliwość integracji z innymi serwisami mogącymi posłużyć jako źródło danych o sytuacji zawodowej absolwentów.

System został zrealizowany na Wydziale Informatyki Politechniki Poznanskiej w ramach zajęć Studia Rozwoju Oprogramowania. Wykonanie systemu zostało zlecone przez rzeczywistego klienta, w postaci przedstawicieli władz wydziału i uczelni. Prace były prowadzone według przyjętej metodyki i harmonogramu.

1.2 Omówienie pracy

Niniejsza praca opisuje otwarty system monitorowania karier zawodowych absolwentów LinkedInGrads (ang. LinkedInGrads: graduate career tracking system), zwany dalej Systemem, realizujący koncepcję przedstawioną w punkcie 1.1. Praca stanowi dokumentację techniczną systemu, a także wyjaśnia idee stojące za poszczególnymi decyzjami projektowymi. Powinna być przydatna zarówno dla użytkowników końcowych systemu, jak i dla osób, które zamierzają go wdrożyć, utrzymywać bądź rozwijać. Jako praca dyplomowa inżynierska jest również skierowana do członków komisji egzaminacyjnej.

W rozdziale 2. przedstawiono aktorów, obiekty biznesowe oraz przypadki użycia występujące w systemie. W rozdziale 3. opisano wymagania funkcjonalne, a w rozdziale 4. wymagania pozafunkcjonalne, wraz z informacją, które z nich zostały zrealizowane. W rozdziale 5. omówiono ogólną architekturę systemu. Rozdział 6. zawiera szczegóły implementacji systemu oraz opis wykorzystanych koncepcji i technologii. W rozdziale 7. przedstawiono metody i narzędzia wspomagające zapewnienie jakości systemu. W rozdziale 8. opisano zebrane wnioski i doświadczenia. Rozdział 9. zawiera podsumowanie całości projektu oraz propozycje dalszego rozwoju systemu. W skład dokumentu wchodzi również bibliografia pracy oraz dodatki, obejmujące informacje uzupełniające, prezentację wyglądu aplikacji, instrukcję instalacji oraz scenariusze manualnych testów akceptacyjnych.

Rozdział 2

Opis procesów biznesowych

Niniejszy rozdział przedstawia otoczenie systemu LinkedInGrads. Wyszczególnieni zostali aktorzy: użytkownicy oraz zewnętrzne systemy. Zaprezentowano obiekty biznesowe będące rzeczywistością, w jakiej porusza się użytkownik systemu. Każdy obiekt opatrzono krótkim opisem oraz wykazem atrybutów. Ostatni podrozdział prezentuje biznesowe przypadki użycia.

2.1 Aktorzy i zewnętrzne systemy

- Administrator - osoba odpowiedzialna za aktualizację danych w systemie: dodawanie nowych absolwentów, znajdowanie adresów profili absolwentów.
- Pracownik dziekanatu - główny użytkownik systemu, grupuje pojęcia dotyczące absolwentów, generuje raporty.
- eLogin - zewnętrzny system Politechniki Poznańskiej służący do uwierzytelniania użytkowników.
- LinkedIn - sieć społecznościowa zrzeszająca specjalistów, służąca nawiązywaniu kontaktów i rozwojowi kariery.

2.2 Obiekty biznesowe

2.2.1 Raport

Raport jest obiektem biznesowym reprezentującym wycinek danych przetworzonych przez system. Dodatkowo każdy z raportów uwzględnia nagłówek zawierający podstawowe informacje na temat raportu oraz zestawienie stosunku ilości danych pochodzących z różnych źródeł. Raport może również zawierać uzasadnienie prezentowanych danych prezentując dodatkowo listę absolwentów wraz z wartością użytych do wygenerowania raportu.

- Analiza pracodawców - prezentuje ilość absolwentów zatrudnionych w danych firmach
- Analiza zamieszkania - prezentuje miejsca zamieszkania absolwentów
- Analiza umiejętności - prezentuje umiejętności nabyte przez studentów
- Analiza zatrudnienia - prezentuje stosunek osób zatrudnionych do bezrobotnych
- Analiza stanowisk - prezentuje ilość studentów na danych stanowiskach, które mogą zostać zgrupowane do bardziej uniwersalnego nazewnictwa. Dodatkowo uwzględnia historyczne stanowiska absolwentów.

Atrybuty:

- Typ raportu,
- Rok ukończenia studiów,
- Uzasadnienie danych.

2.2.2 Grupa

Grupa jest obiektem biznesowym reprezentującym alias dla nazewnictwa użytego w danych pobranych z zewnętrznych źródeł. Obiekt ten wykorzystywany jest wewnątrz raportu w celu unifikacji nazewnictwa użytego w zewnętrznych źródłach.

Atrybuty:

- zgrupowana nazwa,
- nazwa pierwotna.

2.3 Biznesowe przypadki użycia

2.3.1 Generowanie raportu o absolwentach

Przypadek użycia: BUC1: Generowanie raportu o absolwentach
Aktorzy: Pracownik dziekanatu
Pre: Istnieją dane użytkowników pobrane z zewnętrznych źródeł i wzorce raportów
Post: Raport
Scenariusz Główny
1. Pracownik wybiera typ raportu. 2. Pracownik wybiera filtry raportu. 3. Pracownik analizuje otrzymany raport.

2.3.2 Łączenie pojęć w grupy

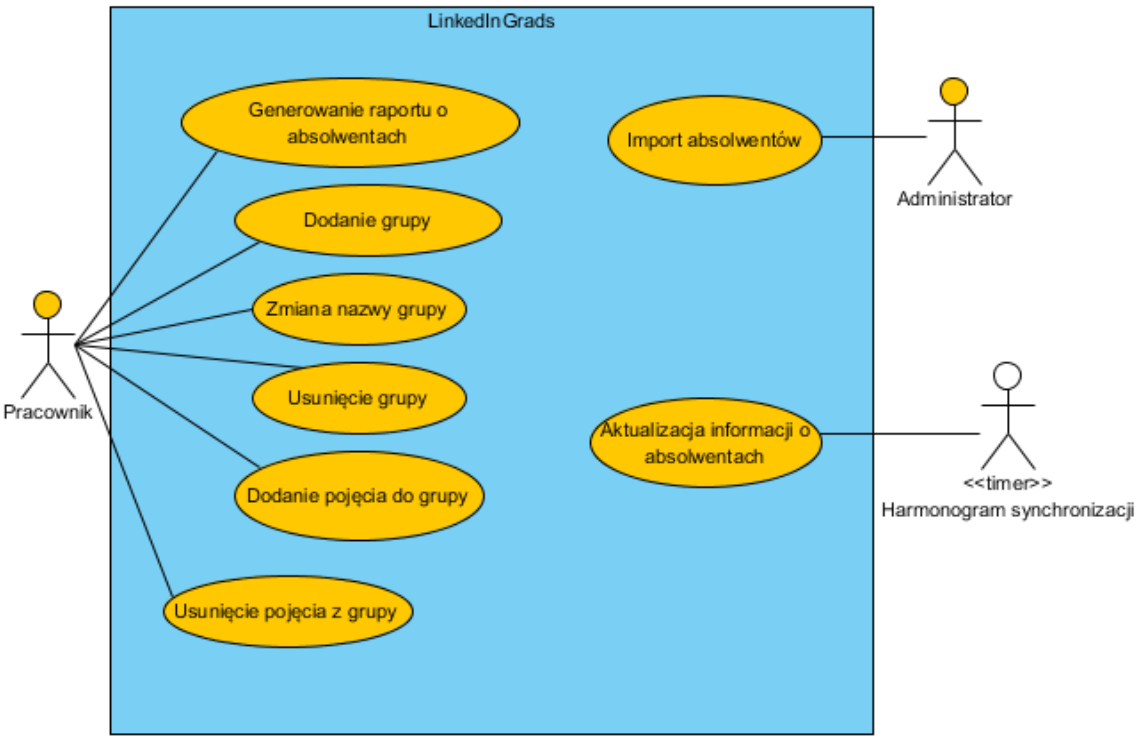
Przypadek użycia: BUC2: Łączenie pojęć w grupy
Aktorzy: Pracownik dziekanatu
Pre: Istnieją dane użytkowników pobrane z zewnętrznych źródeł
Post: Grupa
Scenariusz Główny
1. Pracownik tworzy nową grupę. 2. Pracownik przypisuje pojęcia do grupy.

Rozdział 3

Wymagania funkcjonalne

Wymagania funkcjonalne (ang. functional requirements) określają co system powinien oferować użytkownikowi, to jest jakie operacje można na nim wykonać. Ważne jest, by utrzymać kompletny zbiór poprawnie zdefiniowanych wymagań, pomaga to w zrozumieniu w jaki sposób powinien działać system, nawet dla osób które nie mają wiedzy technicznej oraz pomaga podczas jego projektowania.

Niniejszy rozdział przedstawia wymagania funkcjonalne za pomocą dwóch najpopularniejszych sposobów ich opisu, są to: przypadki użycia (ang. use cases) oraz opowieści użytkownika (ang. user stories). Pierwsza z metod polega na określeniu listy kroków. Reprezentuje ona interakcję między aktorem a systemem. Wykonanie kroków skutkuje osiągnięciem celu, który ma zapewnić system. Druga metoda - Opowieści użytkownika przedstawia w kilku zdaniach potrzebę użytkownika, którą ma realizować system.



Rysunek 3.1: Diagram przypadków użycia

3.0.3 Generowanie raportu o absolwentach

Przypadek użycia: UC1: Generowanie raportu o absolwentach
Aktorzy: Pracownik dziekanatu, System
Scenariusz Główny
1. Pracownik wybiera opcję generowania raportu. 2. System prezentuje typy raportów do wyboru. 3. Pracownik wybiera typ raportu. 4. System prezentuje rok ukończenia studiów do wyboru. 5. Pracownik wybiera rok ukończenia studiów. 6. Pracownik inicjuje generowanie raportu. 7. System prosi o wybranie ścieżki zapisu raportu. 8. Pracownik wybiera ścieżkę zapisu.

3.0.4 Dodanie grupy

Przypadek użycia: UC2: Dodanie grupy
Aktorzy: Pracownik dziekanatu, System
Scenariusz Główny
1. Pracownik wybiera opcję dodania grupy. 2. System prosi o podanie nazwy grupy. 3. Pracownik podaje nazwę grupy. 4. System informuje o pomyślnym dodaniu grupy.
Rozszerzenia
3.A. Grupa o podanej nazwie już istnieje. 3.A.1. System informuje o błędzie. 3.A.2. Powrót do kroku 3.

3.0.5 Zmiana nazwy grupy

Przypadek użycia: UC3: Zmiana nazwy grupy
Aktorzy: Pracownik dziekanatu, System
Scenariusz Główny
1. Pracownik wybiera opcję zmiany nazwy grupy. 2. System prezentuje listę istniejących grup. 3. Pracownik wybiera grupę. 4. System prosi o podanie nowej nazwy grupy. 5. Pracownik podaje nową nazwę grupy. 6. System informuje o pomyślnej zmianie nazwy grupy.
Rozszerzenia
5.A. Grupa o podanej nazwie już istnieje. 5.A.1. System informuje o błędzie. 5.A.2. Powrót do kroku 3.

3.0.6 Usunięcie grupy

Przypadek użycia: UC4: Usunięcie grupy
Aktorzy: Pracownik dziekanatu, System
Scenariusz Główny
1. Pracownik wybiera opcję usunięcia grupy. 2. System prezentuje listę istniejących grup. 3. Pracownik wybiera grupę do usunięcia. 4. System informuje o pomyślnym usunięciu grupy.

3.0.7 Dodanie pojęcia do grupy

Przypadek użycia: UC5: Dodanie pojęcia do grupy
Aktorzy: Pracownik dziekanatu, System
Scenariusz Główny
1. Pracownik wybiera opcję grupowania pojęć. 2. System prezentuje listę istniejących grup. 3. Pracownik wybiera grupę. 4. System prezentuje listę pojęć niezgrupowanych. 5. Pracownik wybiera pojęcie do dodania. 6. System informuje o pomyślnym dodaniu pojęcia do grupy.
Rozszerzenia
3.A. Grupa została w międzyczasie usunięta. 3.A.1. System informuje o błędzie. 3.A.2. Powrót do kroku 2. 5.A. Grupa została w międzyczasie usunięta. 5.A.1. System informuje o błędzie. 5.A.2. Powrót do kroku 2.

3.0.8 Usunięcie pojęcia z grupy

Przypadek użycia: UC6: Usunięcie pojęcia z grupy
Aktorzy: Pracownik dziekanatu, System
Scenariusz Główny
1. Pracownik wybiera opcję grupowania pojęć. 2. System prezentuje listę istniejących grup. 3. Pracownik wybiera grupę. 4. System prezentuje listę pojęć przypisanych do grupy. 5. Pracownik wybiera pojęcie do usunięcia. 6. System informuje o pomyślnym usunięciu pojęcia z grupy.
Rozszerzenia
3.A. Grupa została w międzyczasie usunięta. 3.A.1. System informuje o błędzie. 3.A.2. Powrót do kroku 2.

3.0.9 Aktualizacja informacji o absolwentach

Opowieść użytkownika: US1
Opis: Przebieg aktualizacji informacji o absolwentach. Odbywa się automatycznie, w tle działającego systemu.
Treść: Jako Pracownik chcę aby System, co ustalony czas pobierał informacje o absolwentach z portalu LinkedIn, a następnie aktualizował swoją bazę danych.

3.0.10 Import absolwentów

Przypadek użycia: UC7: Import absolwentów
Aktorzy: Administrator, System
Scenariusz Główny
1. Pracownik wywołuje skrypt importu absolwentów. 2. System prezentuje listę dodanych absolwentów.
Rozszerzenia
1.A. Niepoprawne dane wejściowe. 1.A.1. System przerywa import i informuje o błędzie. 1.A.2. Powrót do 1.

3.0.11 Grupowanie pojęć

Opowieść użytkownika: US2
Opis: Generowanie raportu z użyciem mechanizmu grupowania pojęć.
Treść: Jako Pracownik chcę aby pojęcia (umiejętności, stanowiska, organizacje) w wygenerowanym raporcie były pogrupowane (np. za pomocą metody LDA) w bardziej ogólne zbiory.

Rozdział 4

Wymagania pozafunkcjonalne

Wymagania pozafunkcjonalne (ang. non-functional requirements) określają jakość sposobu realizacji funkcji przez system. Mimo, że wymagania funkcjonalne uda się spełnić, to nie zawsze realizacja celu jest w stanie usatysfakcjonować odbiorcę systemu. Dobrze sprecyzowane wymagania pozafunkcjonalne pozwalają na uniknięcie niskiej użyteczności systemu oraz mogą mieć wpływ na sam sposób realizacji projektu. Stworzenie i weryfikacja tych wymagań jest ważna zarówno dla zlecającego projekt jak i wykonawcy projektu, gdyż doprecyzowują one użyteczność korzystania z systemu, co pozwala na uniknięcie sporów podczas odbioru projektu, w przypadku niskiej jego jakości. Jakość uzyskanego systemu ma również wpływ na jego późniejsze utrzymanie po stronie klienta, stąd tak ważne rozsądne opisanie wymagań pozafunkcjonalnych, które nie zawsze potrafią być oczywiste dla użytkowników końcowych.

4.1 Standard ISO/IEC FDIS 25010

Model jakości oprogramowania standardu ISO/IEC FDIS 25010 [3] wyróżnia następujące charakterystyki i podcharakterystyki jakości oprogramowania:

- Funkcjonalne dopasowanie (ang. Functional suitability)
  - Funkcjonalna kompletnosc (ang. Functional completeness)
  - Funkcjonalna odpowiedniość (ang. Functional correctness)
  - Funkcjonalna poprawność (ang. Functional appropriateness)
- Wydajność (ang. Performance efficiency)
  - Charakterystyka czasowa (ang. Time behaviour)
  - Zużycie zasobów (ang. Resource utilization)
  - Oczekiwana wydajność (ang. Capacity)
- Kompatybilność (ang. Compatibility)
  - Współistnienie (ang. Co-existence)
  - Interoperacyjność (ang. Interoperability)
- Użyteczność (ang. Usability)
  - Rozpoznawalność zastosowania (ang. Appropriateness recognizability)
  - Łatwość nauczenia się (ang. Learnability)

  - Łatwość operowania (ang. Operability)
  - Ochrona użytkownika przed błędami (ang. User error protection)
  - Estetyka interfejsu użytkownika (ang. User interface aesthetics)
  - Dostępność personalna (ang. Accessibility)
- Niezawodność (ang. Reliability)
  - Dojrzałość (ang. Maturity)
  - Tolerancja uszkodzeń (ang. Availability)
  - Odporność na wady (ang. Fault tolerance)
  - Odtwarzalność (ang. Recoverability)
- Bezpieczeństwo (ang. Security)
  - Poufność (ang. Confidentiality)
  - Integralność (ang. Integrity)
  - Niezaprzeczalność (ang. Non-repudiation)
  - Identyfikowalność (ang. Accountability)
  - Autentyczność (ang. Authenticity)
- Łatwość utrzymania (ang. Maintainability)
  - Modułowość (ang. Modularity)
  - Łatwość ponownego wykorzystania (ang. Reusability)
  - Łatwość analizy (ang. Analysability)
  - Łatwość zmiany (ang. Modifiability)
  - Łatwość testowania (ang. Testability)
- Przenośność (ang. Portability)
  - Łatwość adaptacji (ang. Adaptability)
  - Łatwość instalacji (ang. Installability)
  - Łatwość zamiany (ang. Replaceability)

Wymienione podcharakterystyki posłużyły jako kategorie wymagań pozafunkcjonalnych w projekcie. Ze względu na specyfikę systemu - aplikacji internetowej niektóre kategorie tego standardu nie zostały wykorzystane podczas prac nad wymaganiami pozafunkcjonalnymi.

4.2 Wymagania pozafunkcjonalne i ich weryfikacja

W kolejnych tablicach przedstawiono wymagania pozafunkcjonalne systemu LinkedInGrads, określonych przy pomocy standardu ISO/IEC FDIS 25010. Priorytet wymagań określono za pomocą notacji:

- Would - System może spełniać dane wymaganie,
- Should - System powinien spełniać dane wymaganie,

- Must - System musi spełniać dane wymaganie. Dodatkowo określono złożoność każdego z wymagań wykorzystując następującą notację:
- Low - Niski stopień złożoności wymagania pozafunkcjonalnego,
- Medium - Średni stopień złożoności wymagania pozafunkcjonalnego,
- High - Wysoki stopień złożoności wymagania pozafunkcjonalnego. Metryki te pozwalały podczas prac nad projektem na realizację najważniejszych wymagań, uwzględniając ich złożoność czasową skonfrontowaną z dostępnym czasem na realizację projektu. Ostatnia kolumna tabeli określa czy w projekcie LinkedInGrads udało się spełnić dane wymaganie pozafunkcjonalne.

4.2.1 Funkcjonalne dopasowanie: Funkcjonalna kompletność

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Współpraca z przeglądarką Firefox (dla dwóch najnowszych wersji).	Must	Low	Zrealizowano

Tablica 4.1: Funkcjonalne dopasowanie: Funkcjonalna kompletność

System był tworzony wykorzystując do testów najnowszą wersję przeglądarki Mozilla Firefox, oraz dodatkowo Google Chrome. Zapewniło to kompatybilność systemu z tymi przeglądarkami, a funkcje użyte wewnątrz systemu nie wykraczają poza zakres możliwości poprzednich kilku wersji tych przeglądarek.

4.2.2 Wydajność: Charakterystyka Czasowa

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Dla 20000 absolwentów aktualizacja danych z LinkedIn nie powinna trwać dłużej niż tydzień.	Must	Medium	Zrealizowano
System powinien estymować czas aktualizacji danych.	Would	High	Pominęto
Generowanie raportu nie może trwać dłużej niż 10 sekund - w przeciwnym wypadku informacja o czasie.	Should	Medium	Zrealizowno

Tablica 4.2: Wydajność: Charakterystyka Czasowa

Wymagania tej kategorii udało się spełnić, a dzięki aktualizacji danych w tle, estymacja czasu zakończenia aktualizacji okazała się zbędna w finalnej wersji produktu, co również spowodowało niski czas generowania raportu.

4.2.3 Kompatybilność: Współistnienie

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Na jednym serwerze może być uruchomionych wiele instancji aplikacji.	Would	Medium	Zrealizowano

Tablica 4.3: Kompatybilność: Współistnienie

Wymaganie to zostało zrealizowane na poziomie architektury poprzez wykorzystanie aplikacji internetowej uruchamianej w środowisku serwera WWW.

4.2.4 Kompatybilność: Interoperacyjność

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
System ma udostępniać dane absolwenta w trybie odczytu w formacie JSON przez webservice.	Should	High	Pominięto
System ma generować raporty w formacie CSV.	Should	Low	Pominęto
System ma generować raporty w formacie XLS.	Must	Low	Zrealizowano

Tablica 4.4: Kompatybilność: Interoperacyjność

Ze względu na duży narzut czasowy wymagań pozafunkcjonalnych z tej kategorii udało się zrealizować tylko generowanie raportu w formacie XLS. Pominięte wymagania funkcjonalne nie były kluczowe dla systemu, co spowodowało, że ich realizacja nie została przydzielona do żadnego ze sprintów.

4.2.5 Użyteczność: Łatwość nauczenia się

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Interfejs użytkownika powinien być skonsultowany ze wszystkimi potencjalnymi użytkownikami.	Must	Low	Zrealizowano
Instrukcja użytkownika powinna być zorientowana na cele poszczególnych aktorów.	Must	Low	Zrealizowano

Tablica 4.5: Użyteczność: Łatwość nauczenia się

Zespół był wysoce zorientowany na potrzeby użytkownika końcowego, co już w samej fazie projektowania interfejsów użytkownika zapewniło wysoką jakość systemu. Po skonsultowaniu z użytkownikami, wdrożono udoskonalenia co pozwoliło na zrealizowanie wymagania pozafunkcjonalnego.

4.2.6 Użyteczność: Ochrona użytkownika przed błędami

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
System musi być przygotowany na rozszerzenie list atrybutów.	Must	Low	Zrealizowano
System powinien logować błedy (wyjątki) do logu systemowego.	Must	Low	Zrealizowano
System powinien informować o błędnej konfiguracji natychmiast po jej wprowadzeniu.	Would	Medium	Pominięto
System ma potwierdzać powiązanie/odwiązanie użytkownika.	Would	Low	Pominięto
System powinien zapewnić kontrolę nieprzewidzianych wyjątków.	Must	Low	Zrealizowano częściowo.
Po wykonaniu importu danych system informuje ilu absolwentów zaimportowano, a ile jest konfliktów.	Must	Low	Pominięto

Tablica 4.6: Użyteczność: Ochrona użytkownika przed błędami

System został przygotowany mając od początku na uwadze możliwość rozszerzenia go o dodatkowe dane jak i dodatkowe źródła danych, co zapewniło realizację pierwszego z wymagań. Wszystkie błędy które pojawią się w aplikacji automatycznie zapisywane są do logów, co częściowo realizuje również wymaganie dotyczące kontroli nieprzewidzianych wyjątków, które zostają w momencie wystąpienia ukryte przed użytkownikiem końcowym. Podczas prac nad systemem zrezygnowano z możliwości konfiguracji parametrów jego pracy, ze względu na brak zasadności jakichkolwiek zmian. Wymaganie to powstało mając na uwadze możliwe ograniczenia narzucone na system pochodzące z zewnętrznych źródeł danych. Również w trakcie prac zrezygnowano z automatycznego wyszukiwania użytkowników wewnątrz sieci społecznościowej LinkedIn i wiązania znalezionych kont z danymi osób dostarczonymi z dziekanatu. Wyeliminowało to potrzebę potwierdzania powiązania/odwiązania użytkownika, gdyż dane te wprowadzane są manualnie przez użytkownika. Ostatnie z wymagań nie zostało zrealizowane ze względu na dołączenie go do wymagań pozafunkcjonalnych w późnej fazie projektu i nie uwzględnienie jego realizacji w żadnym ze sprintów.

4.2.7 Użyteczność: Estetyka interfejsu użytkownika

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Interfejs powinien zawierać logo Politechniki Poznańskiej oraz opis przeznaczenia systemu.	Must	Low	Zrealizowano

Tablica 4.7: Użyteczność: Estetyka interfejsu użytkownika

4.2.8 Użyteczność: Dostępność personalna

Wymaganie	Priorytet	Złożoność	Realizacja
System dostępny tylko w jednej wersji językowej - Polskiej	Must	Low	Zrealizowano

Tablica 4.8: Użyteczność: Dostępność personalna

4.2.9 Niezawodność: Tolerancja uszkodzeń

Wymaganie	Priorytet	Złożoność	Realizacja
Przerwy serwisowe możliwe są w godzinach 18-6 lub po wcześniejszym ustaleniu.	Should	Low	Pominięto
Niekontrolowana awaria może trwać maksymalnie 2 dni.	Should	Low	Pominięto

Tablica 4.9: Niezawodność: Tolerancja uszkodzeń

Wymagania należące do tej kategorii nie zostały zrealizowane. Wynika to z faktu, że dotyczą one utrzymania systemu po jego wdrożeniu, więc weryfikacja tych wymagań była niemożliwa do przeprowadzenia.

4.2.10 Niezawodność: Odporność na wady

Wymaganie	Priorytet	Złożoność	Realizacja
System należy poddać analizie z wykorzystaniem metody ATAM.	Must	Low	Zrealizowano
Wszystkie funkcje głównego scenariusza powinny być możliwe do przetestowania w sposób automatyczny.	Should	Low	Zrealizowano
W przypadku problemów z połączeniem, proces aktualizacji danych powinien być wznawiany (od stanu w którym nastąpiło przerwanie).	Would	Low	Zrealizowano
System powinien posiadać testy jednostkowe oraz testy akceptacyjne.	Must	High	Zrealizowano
W przypadku problemów z połączeniem proces aktualizacji danych powinien być ponawiany.	Must	Low	Zrealizowano

Tablica 4.10: Niezawodność: Odporność na wady

4.2.11 Niezawodność: Odtwarzalność

Wymaganie	Priorytet	Złożoność	Realizacja
System powinien tworzyć kopie zapasowe danych codziennie.	Must	Low	Zrealizowano

Tablica 4.11: Niezawodność: Odtwarzalność

4.2.12 Bezpieczeństwo: Poufność

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
System zapewnia szyfrowane połączenie z użytkownikiem (nie wymaga zaufanego certyfikatu).	Should	Low	Zrealizowano
System powinien korzystać z systemu eLogin do uwierzytelniania użytkowników.	Must	Medium	Zrealizowano

Tablica 4.12: Bezpieczeństwo: Poufność

4.2.13 Bezpieczeństwo: Integralność

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Dane przechowywane w systemie powinny być chronione przed nieuprawnionym odczytem, modyfikacją, usunięciem.	Must	Low	Zrealizowano
System powinien być odporny na SQL injection.	Must	Low	Zrealizowano

Tablica 4.13: Bezpieczeństwo: Integralność

Wymagania zawarte w tej kategorii zostały zrealizowane na poziomie architektury, poprzez wykorzystanie przygotowanych zapytań (ang. prepared statements), oraz wymaganie uwierzytelnienia użytkownika.

4.2.14 Bezpieczeństwo: Niezaprzeczalność

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Import danych powinien mieć stempel czasowy – w raporcie powinna się znaleźć data aktualizacji danych na podstawie których będzie generowany raport.	Must	Low	Zrealizowano

Tablica 4.14: Bezpieczeństwo: Niezaprzeczalność

4.2.15 Łatwość utrzymania: Łatwość analizy

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Należy zapewnić dokumentację zgodną z wymaganiami DRO.	Must	Medium	Zrealizowano
Kod źródłowy systemu musi przestrzegać konwencji wymaganej przez DRO.	Must	Medium	Zrealizowano

Tablica 4.15: Łatwość utrzymania: Łatwość analizy

4.2.16 Łatwość utrzymania: Łatwość zmiany

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Architektura powinna zapewnić rozdzielenie warstwy backendowej od frontendowej.	Must	Low	Zrealizowano
System musi być przygotowany na zmianę generowania raportów i udostępniania danych.	Must	Low	Zrealizowano
Model danych systemu powinien być przygotowany na zmiany: integrację z nowymi systemami, adaptację na potrzeby innych uczelni.	Must	Low	Zrealizowano

Tablica 4.16: Łatwość utrzymania: Łatwość zmiany

Rozdzielenie warstwy frontendowej od backendowej zostało zrealizowane z pomocą frameworka Ruby on Rails implementującego wzorzec projektowy MVC (Model-view-controller). System zapewnia również możliwość tworzenia nowych raportów, oraz integrację z innymi systemami ze względu na modularną architekturę systemu.

4.2.17 Łatwość utrzymania: Łatwość testowania

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Przewidzenie dwóch trybów - produkcyjny i testowy.	Should	Low	Zrealizowano

Tablica 4.17: Łatwość utrzymania: Łatwość testowania

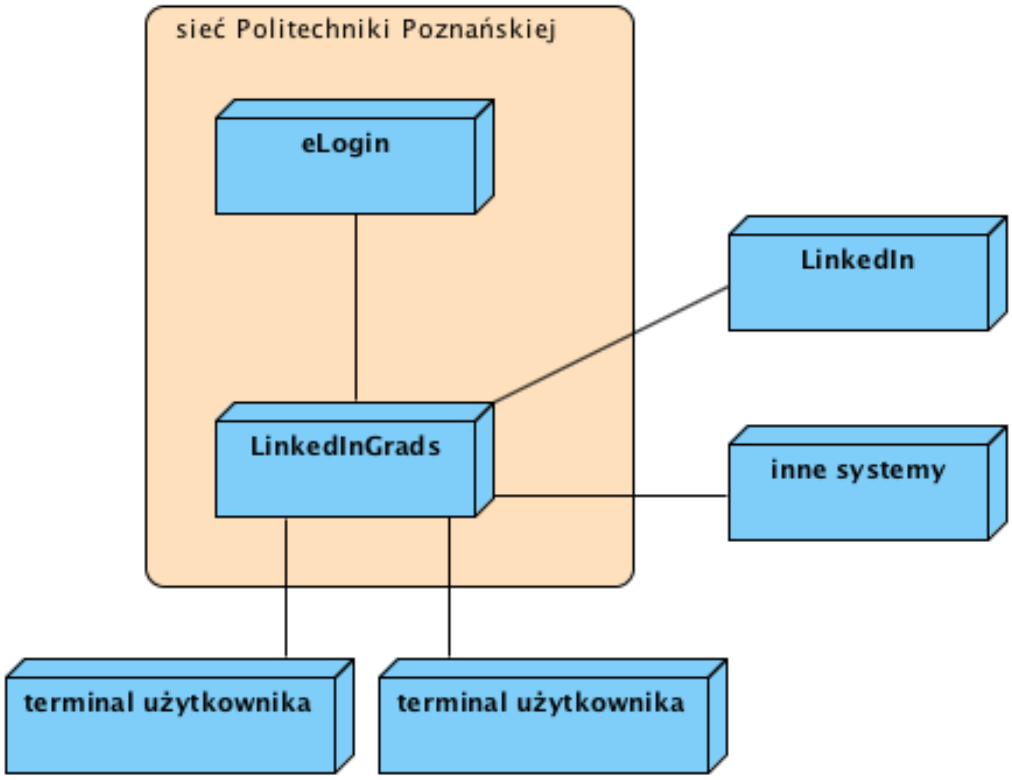
4.2.18 Przenośność: Łatwość instalacji

Wymaganie	**Priorytet**	**Złożoność**	**Realizacja**
Należy dostarczyć szczegółową instrukcję instalacji oraz konfiguracji środowiska wykonawczego.	Must	Medium	Zrealizowano

Tablica 4.18: Przenośność: Łatwość instalacji

Rozdział 5

Architektura systemu



Rysunek 5.1: Architektura systemu LinkedInGrads

W rozdziale przedstawione zostaną zagadnienia związane z architekturą systemu LinkedInGrads. Ukazane zostaną zastosowane podejścia architektoniczne, podjęte decyzje (razem z ich uzasadnieniami oraz zależnościami), zobrazowany będzie przyjęty schemat bazy danych. W celu lepszego przedstawienia tematu architektury, ze względu na jego złożoność, w podrozdziale 5.4. Perspektywy architektoniczne wykorzystano model 4+1 Views.

5.1 Zastosowane podejście architektoniczne

Rozdział dotyczy wzorca projektowego, w oparciu o który zaprojektowano architekturę systemu.

5.2 Analiza SWOT

Analiza SWOT przyjętego podejścia architektonicznego.

5.3 Perspektywy architektoniczne

5.3.1 Perspektywa fizyczna

Rysunek wraz z opisem.

5.3.2 Perspektywa logiczna

Rysunek wraz z opisem.

This view of the architecture addresses the functional requirements of the system, in other words, what the system should do for its end users. It is an abstraction of the design model and identifies major design packages, subsystems, and classes.

5.3.3 Perspektywa implemetancyjna

Rysunek wraz z opisem. Można tutaj też umieścić perspektywę kodu.

This view describes the organization of static software modules (source code, data files, components, executables, and other accompanying artifacts) in the development environment in terms of packaging and layering and in terms of configuration management (ownership, release strategy, and so on). It addresses the issues of ease of development, management of software assets, reuse, subcontracting, and off-the-shelf components.

5.3.4 Perspektywa procesu (równoległości)

Rysunek wraz z opisem.

This view addresses the concurrent aspects of the system at runtime—tasks, threads, or processes as well as their interactions. It addresses issues such as concurrency and parallelism, system start-up and shutdown, fault tolerance, and object distribution. It deals with issues such as deadlock, response time, throughput, and isolation of functions and faults. It is concerned with scalability. Examples are a flight management process, flight plan entry processes, and an airspace management process.

5.4 Decyzje projektowe

Tutaj piszemy o decyzjach projektowych, związkach pomiędzy nimi oraz innych związanych sprawach.

5.5 Wykorzystane technologie

Opis wykorzystywanych technologii (COTS).

5.6 Schemat bazy danych

Schemat bazy danych, ze względu na objętość, można umieścić w którymś dodatku.

Rozdział 6

Opis implementacji

6.1 Wstęp

Wprowadzenie. Struktura tego rozdziału nie jest z góry określona, gdyż mocno zależy to od specyfiki projektu. Generalnie w poszczególnych podrozdziałach każdy powinien opisać swoją część z takiego technicznego punktu widzenia. Piszecie, jak zrealizowaliście poszczególne wymagania, jak to wygląda „pod maską", oczywiście też trzeba przyjąć jakiś poziom szczegółowości. W bardzo szczególnych przypadkach chyba może się zdarzyć, że trzeba będzie załączyć fragment jakiegoś kodu źródłowego czy konfiguracji – generalnie ma to być opisane w taki sposób, że jako osoba nieznająca systemu siadam i wiem, jak i co zrobiliście.

Ten rozdział ma umożliwić innym osobom modyfikację kodu. Warto zatem opisać najbardziej prawdopodobne scenariusze zmian. Np. jesteśmy świadomi, że w naszym systemie może zaistnieć konieczność zmiany wizualnej jakiegośraportu. Wówczas należałoby wyjaśnić w jaki sposób należy zmodyfikować kod, aby tego typu zmiany wcielić w życie.

Podsumowując można tutaj umieszczać dwa typy podrozdziałów: - tłumaczące działania określonych mechanizmów - opisujące pewne scenariusze zmiany - tego typu rozdział powinien mieć jasno określony scenariusz zmiany (warto także nawiązać do wymagań) oraz rozwiązanie:

6.2 Scenariusz zmiany: modyfikacja raportu

Opis problemu

System X umożliwia generowanie raportów o stanie finansowym (UC1). Istnieje konieczność zmiany wizualnej w układzie graficznym raportu.

Rozwiązanie

...

6.3 Mechanizm generowania raportu

Rozdział o charakterze opisowym prezentujący implementację konkretnego mechanizmu.

6.4 Sekcja ...

Dalsze opisy.

Rozdział 7

Zapewnianie jakości i konserwacja systemu

7.1 Testy i weryfikacja jakości oprogramowania

7.1.1 Testy jednostkowe

Tutaj piszemy o testach jednostkowych – jak je zrobiliśmy oraz jakie były problemy, wnioski, które się pojawiły.

7.1.2 Testy integracyjne

Podrozdział opcjonalny (bo nie wiem, czy wszyscy robią takie testy). Tutaj piszemy o testach integracyjnych – jak je zrobiliśmy oraz jakie były problemy, wnioski, które się pojawiły.

7.1.3 Testy akceptacyjne

Tutaj piszemy o testach akceptacyjnych – jak je zrobiliśmy oraz jakie były problemy, wnioski, które się pojawiły. Tutaj zwykle pojawią się testy akceptacyjne przygotowane przez Waszych kierowników, ale także różne opisy, w tym – jeśli jest to zrobione – automatycznych testów.

Poniżej przedstawiono Manualne Testy Akceptacyjne:

MAT01 : Nazwa testu pierwszego		
Warunki początkowe		
• **Warunek początkowy 1** • **Warunek początkowy 2**		
Krok	**Akcja**	**Oczekiwana odpowiedź**
1.	Polecenie pierwsze	Odpowiedź systemu
2.	Polecenie drugie	Odpowiedź systemu
3.	Polecenie trzecie	Odpowiedź systemu
Uwagi		
Jakaś uwaga.		

MAT02 : Nazwa testu drugiego		
Warunki początkowe		
• **Warunek początkowy 1** • **Warunek początkowy 2**		
Krok	**Akcja**	**Oczekiwana odpowiedź**
1.	Polecenie pierwsze	Odpowiedź systemu
2.	Polecenie drugie	Odpowiedź systemu
4.	Polecenie trzecie	Odpowiedź systemu
Uwagi		
Brak		

7.1.4 Inne metody zapewniania jakości

Jeśli testowaliście projekt w jakiś inny sposób, mieliście inne formy weryfikacji (np. rzucanie komputerem o ściany czy z drugiego miejsca), to tutaj to opisujecie. Prawdopodobnie dobrze będzie opisać tutaj wszelkie testy dla interfejsu, jeśli np. testowaliście użytkowników i sam system pod tym kątem.

7.2 Sposób uruchomienia i działania systemu

Tutaj można napisać jak przygotować system do działania i jak przeprowadzić konfigurację. Prawdę mówiąc, nie wiem, czy to powinno się tutaj znaleźć, ale być może warto, dlatego na wszelki wypadek to umieszczam.

Rozdział 8

Zebrane doświadczenia

Tutaj znajduje się opis wszystkich Waszych doświadczeń związanych z projektem – zarówno pozytywnych jak i negatywnych, dotyczących organizacji, środowiska czy samych już kwestii technicznych. To ma być zebranie Waszych wniosków, wraz z prawdopodobnymi nauczkami dla przyszłych roczników.

To dobre miejsce na zaaplikowanie zawartości Lessons Learned Log, jeśli tak prowadziliście, ale też miejsce na własne przemyślenia.

Rozdział 9

Zakończenie

9.1 Podsumowanie

Podsumowanie powstałego systemu, czy przedsięwzięcie się udało, czy to się nadaje do czegokolwiek. Taki ładny epilog na koniec pracy inżynierskiej.

9.2 Propozycja dalszych prac

Być może system wymaga jakichś prac w przyszłości lub są jakieś propozycje rozszerzenia funkcjonalności. Dotyczy to zarówno tego, co być może sami będziecie dalej robić (jeśli Wam oczywiście zapłacą) lub mają po Was przejąć inne osoby (z następnymi rocznikami włącznie).

Dodatek A

Informacje uzupełniające

A.1 Wkład poszczególnych osób do przedsięwzięcia

Skład zespołu pracującego nad projektem został przedstawiony w tablicy A.1.

Stanowisko	**Osoba**
Założyciel projektu, klient	Tytuł Imię Nazwisko
Główny użytkownik	Tytuł Imię Nazwisko
Główny dostawca	Tytuł Imię Nazwisko
Dostawca od strony DRO	Tytuł Imię Nazwisko
Starszy konsultant	Tytuł Imię Nazwisko
Konsultant	Tytuł Imię Nazwisko
Kierownik projektu	inż. Imię Nazwisko
Analityk/Architekt	inż. Imię Nazwisko
Programiści	Imię Nazwisko Imię Nazwisko Imię Nazwisko Imię Nazwisko

Tablica A.1: Osoby związane z przedsięwzięciem

Teraz bardzo ważna rzecz – w tym miejscu piszecie, co kto przygotowywał w tekście pracy inżynierskiej. Prawdopodobnie tutaj będziecie musieli wymienić, które rozdziały został dla Was przygotowane przez kierownika projektu, analityka, architekta lub inną osobę. Oczywiście, piszecie też, za które części dokumentu Wy jesteście odpowiedzialni. To jest ważna, aby ta część była tutaj precyzyjnie przygotowana – takie są wymogi uczelni oraz też uwzględnienia pracy innych osób.

Odpowiedzialność za część implementacyjną systemu została przedstawiona poniżej:

Imię i nazwisko pierwszego programisty

- Odpowiedzialność 1
- Odpowiedzialność 2
- Odpowiedzialność 3
- ...

Imię i nazwisko drugiego programisty

- Odpowiedzialność 1

- Odpowiedzialność 2
- Odpowiedzialność 3
- ...

Imię i nazwisko trzeciego programisty

- Odpowiedzialność 1
- Odpowiedzialność 2
- Odpowiedzialność 3
- ...

Imię i nazwisko czwartego programisty

- Odpowiedzialność 1
- Odpowiedzialność 2
- Odpowiedzialność 3
- ...

Ewentualne podziękowania dla innych osób, które Wam pomagały, mowy dziękczynne, itd.

A.2 Wykaz użytych narzędzi

Wprawdzie jest odpowiedni podrozdział w rozdziale 5, ale tutaj można wymienić nawet małe narzędzia i biblioteki, które wykorzystywaliście (np. narzędzie do robienia makiet interfejsu) i które warto wymienić, także dla przyszłych roczników (można też dać linki).

A.3 Zawartość płyty CD

Do dokumentu załączono płytę CD o następującej zawartości:

- Zawartość 1
- Zawartość 2
- Zawartość 3
- ...

Dodatek B

Wygląd aplikacji

Dodatek opcjonalny, tutaj można zamieścić jakieś zrzuty ekranu czy inne materiały dotyczące interfejsu. Jeżeli nie chcecie tego dodatku, zakomentujecie załączenie tego pliku w `thesis-bachelor-polski.tex`.

Dodatek C

Schemat bazy danych

Dodatek opcjonalny, tutaj można zamieścić schematy bazy danych, jeśli nie zmieścił się w rozdziale o architekturze. Może być też tak, że będziecie mieli legen-czekaj-darny schemat o formacie A3 i będziecie go osobno drukować i wklejać w tym miejscu. Jeżeli nie chcecie tego dodatku, zakomentujecie załączenie tego pliku w `thesis-bachelor-polski.tex`.

Literatura

Instytut Informatyki, Wydział Informatyki
Politechnika Poznańska

Skład przy użyciu systemu LaTeX.

BibTeX:

```
@mastersthesis{ key,
    author = "Jakub Rojek \and Dawid Neumann \and Wiesław Nowak \and Wiktor Kierzek",
    title = "{Tytuł pracy inżynierskiej}",
    school = "Poznan University of Technology",
    address = "Pozna{\'n}, Poland",
    year = "2013",
}
```