* Nazwa kodowa: **Ledikom**
* Temat pracy: **Zarządzanie siecią aptek**
* Skład zespołu:
  + Tomasz Kasprzak (Team Leader)
  + Adrianna Leśna
  + Arkadiusz Skiba
  + Bartosz Piętka
  + Hleb Shypula
  + Kateryna Andrusiak
  + Vadym Semkovych

**Raport** z przeprowadzonej rozmowy na żywo z właścicielem sieci aptek Ledikom:

Na podstawie rozmowy przeprowadzonej z właścicielem sieci aptek ustalono podstawowe właściwości i funkcjonalności systemu.

Właściciel wyraził konieczność możliwości dodawania pracowników do systemu, uzupełniania informacji o nich, dodawania aptek, ewidencji inspekcji i naruszeń oraz ewidencji dostaw.

Zaznaczył także potrzebę ustalania harmonogramu pod kątem świąt oraz dodawania kursów do pracowników.

Dodatkowo do wymagań klienta dodano potrzebne funkcje nieuwzględnione w ramach pierwotnej, nieszczegółowej rozmowy.

* **Definicja problemu:**
  + Archiwizacja danych, automatyzacja obliczeń, płatności i zarządzania siecią aptek.
  + Firma ma oprogramowanie do księgowości, ale nie ma programu do przechowywania i odczytu różnych informacji związanych ogólnie z firmą (nie księgowość).
* **Wizja projektu:**
  + Celem jest napisanie programu do zarządzania siecią aptek. Program powinien automatyzować rozwiązania problemów związanych ściśle z działalnością farmaceutyczną oraz różnych problemów ogólnie typowych dla biznesu farmaceutycznego.
  + Potencjalny klient – firma zajmująca się siecią aptek oraz jej rozwojem.
  + Wymagania:  
    Pracownik, obiekt (apteka, biuro, magazyn), dostawca oraz powiązania pomiędzy tymi aktorami (kursy, praca w święta, inspekcje, dostawy itd.).   
    Zdefiniowanie algorytmów do interpretacji wprowadzonych danych i odpowiednich obliczeń na podstawie tych danych (np. terminy kursów oraz inspekcji).
* Koncepcja systemu:
  + Architektura logiczna:
    - Na architekturę systemu składają się poniższe tabele połączone relacjami:

**Employee** – tabela pracowników  
**Employee\_info** – dane pracownika

* Dob – data urodzenia
* Position – stanowisko w firmie
* Category – prawnie wymagana, według ustawodawstwa farmaceutycznego na Białorusi, kategoria pracownika farmaceutycznego (1, 2 lub wyższa)
* PPE – według ustawodawstwa farmaceutycznego na Białorusi co rok pracodawca ma wydać komplet środków ochrony osobistej (fartuch etc)
* Course\_hours\_sum – według ustawodawstwa farmaceutycznego na Białorusi każdy pracownik farm. ma w ciągu każdych 5 lat odbyć 100 godzin kursów (5 lat liczonych jest od *hiring\_date*)

**Employee\_data**

* Role – stanowiska (uprawnienia) (np. szef, pracownik, kierownik apteki)

**Course** – kursy dla pracowników

**Facility** – obiekt

**Facility\_info**

* Status – (biuro, apteka)

**Employee\_facility** – N do M, bo pracownik może pracować jednocześnie na kilku obiektach.

**Contract** – umowa zawierana jest na jednego pracownika na jednym obiekcie.

* Type – typ umowy, dokładniej pełen/niepełen etat oraz czy pracownik pracuje poza firmą (W głównym miejscu pracy - tylko na tym obiekcie;  
  Wewnętrznie w niepełnym wymiarze godzin - w firmie na więcej niż jednym obiekcie;

Zewnętrznie w niepełnym wymiarze godzin - pracuje również poza firmą.

**Holiday** – święta i dni wolne od pracy (Wielkanoc, Boże Narodzenie itd.)

**Facility\_holiday**

* Proceeds – dochód w to święto.

**Supplier ma storage** – dostawę mamy z tego magazynu.

**Facility\_storage**

* Delivery\_day – dzień dostawy, pon, wto, śro itd.

**Delivery\_record** – opisuje już konkretną dostawę.

* Date – data dostawy.
* Payment\_delay – dostawca pozwala na pewne opóźnienie opłaty (liczone w dniach).

**Inspection** – według ustawodawstwa farmaceutycznego na Białorusi, co rok w każdej aptece ma być przeprowadzona samokontrola według ustalonej listy pytań.

* Description – opis w razie potrzeby
* Id\_employee – pracownik odpowiedzialny za inspekcję

**Checkup** – tabela na przechowywanie jednego pytania z ww. listy oraz odpowiedzi na to pytanie (Tak/Nie).

**Violation** – jeśli udzielono odpowiedzi NIE, to zapisujemy naruszenie.

* Id\_employee – pracownik odpowiedzialny za naprawę
* Correction\_term – termin do którego ma czas naprawić
* Correction\_date – data naprawy
* Koncepcja systemu cd:
  + Zakres usług:

Jak zostało powiedziane, system opiera się na istnieniu **pracownika**, który może pracować jednocześnie na kilku **obiektach**. Cały system jest zbudowany wokół tego.

Szef może dodawać pracowników, obiekty oraz pracowników do obiektów. Może to wszystko aktualizować i uzupełniać.

Pracownik po zalogowaniu może przeglądać za co odpowiada (np. naprawa naruszenia po samokontroli).

* + Główne funkcje:

Do pracownika możemy przypisać odbyty kurs, automatycznie uzupełniamy wtedy sumę godzin w bieżących 5 latach dla tego pracownika.

Obsługujemy przypisywanie pracowników do pracy w święta - employee\_holiday\_facility – podczas tworzenia harmonogramu na miesiąc, jeśli jest święto, to patrzymy kto pracował podczas poprzednich świąt, żeby nie przypisać go dwa razy pod rząd. Analizujemy, czy istnieje potrzeba otwierania apteki podczas danego święta na postawie przychodu podczas tego samego święta rok wcześniej. W tabeli *proceeds* przechowujemy wspomniany dochód w danym dniu świątecznym i określamy, czy istnieje potrzeba otwierać aptekę w to święto.

System obsługuje także przypominanie, jedno z nich - dla kontraktu pracownika (np. 3 miesiące przed końcem), żeby przedłużać umowę lub rekrutować nowego pracownika. Drugie - dla daty urodzenia pracownika.

Pracownik może wypisać listę inspekcji, do których jest przypisany. Pracownik odpowiedzialny za inspekcję przypisuje pracowników do naprawy naruszeń i określa maksymalny termin ich naprawy.

Pod kątem opłat za dostawy, przykładowo gdy pracownik zaznaczył dostawę, księgowa widzi, że ma zapłacić z podanym maksymalnym możliwym opóźnieniem.

Robi to z głównego portfelu firmy.

Analogicznie dostawca posiada portfel, do którego wpływają przelewy od księgowej.

* + Założenia technologiczne:

Do podziału ról i przydziału zadań używany jest program **Jira**.

Wykorzystywany jest system kontroli wersji **GitHub**.

System implementowany jest w języku **Java** z użyciem **JavaFX** (dodatkowo **SceneBuilder**).

Używana jest baza danych **ElephantSQL**.

Do automatyzacji budowania oprogramowania używane jest narzędzie **Maven**.

Do implementacji komunikacji z bazą wykorzystywany jest program **DataGrip**, natomiast do implementacji kodu **Intellij**.

* + Organizacja pracy nad projektem:

Rozdzielono prace nad projektem na następujące role:

* + - projektanci architektury – Tomasz Kasprzak, Hleb Shypula
    - analitycy/projektanci UX - Hleb Shypula, Adrianna Leśna
    - koderzy – Arkadiusz Skiba, Bartosz Piętka, Kateryna Andrusiak
    - testerzy – Arkadiusz Skiba, Vadym Semkovych, Kateryna Andrusiak
    - scrum masterzy – Tomasz Kasprzak, Adrianna Leśna