1. Faza konceptualna

Streszczenie:

Celem przedsięwzięcia jest zaprojektowanie bazy danych umożliwiającej zarządzanie systemem akademików. Baza powinna umożliwiać gromadzenie danych dotyczących akademików, pokoi wraz z wyposażeniem oraz informacji na temat studentów (informacji osobistych, danych kontaktowych, statusów akademickich, historii zamieszkania) i pracowników (informacji osobistych, danych kontaktowych, stanowiska itd.). Dodatkowo system będzie umożliwiał logowanie, zarządzanie zakwaterowaniami, składanie i rozpatrywanie wniosków, podpisywanie i rozwiązywanie umów, monitorowanie opłat oraz zgłaszanie usterek.

W projekcie należy uwzględnić powiązania między poszczególnymi encjami, przykładowo pracownika z akademikiem lub studenta z pokojem. W zależności od roli, każdy użytkownik będzie miał dostęp do innych funkcjonalności. System powinien również umożliwiać generowanie raportów oraz monitorowanie statystyk.

Wymagania funkcjonalne:

1. Użytkownik:
    - Użytkownicy (studenci, pracownicy, administratorzy) powinni mieć możliwość rejestracji i logowania do systemu. W zależności od poziomu dostępu użytkownikom jest przypisywana rola: admin, student, employee, accountant, manager.
    - Każdy użytkownik ma możliwość zarządzania swoimi danymi (zmiana hasła, aktualizacja danych kontaktowych).
2. Student:
    - Student ma możliwość złożenia wniosku o miejsce w akademiku.
    - Jeśli student wynajmuje pokój może złożyć wypowiedzenie najmu pokoju.
    - Studenci mają możliwość zgłaszania usterek technicznych dotyczących ich pokoi.
    - Student posiada możliwość przeglądania i opłacania rachunków związanych, które jego dotyczą.
    - Student ma możliwość podglądu wyposażenia pokoju.
    - Student ma możliwość podglądu regulaminu akademiku.
    - Student może wynajmować tylko jeden pokój.
3. Pracownik:
    - Pracownik może mieć wiele umów o pracę.
    - Pracownik może generować raporty dotyczące usterek i pokoi.
    - Pracownik może zarządzać pokojami.
    - Może zarządzać wnioskami studentów.
    - Zarządzanie umowami pracowników.
    - Obsługa zgłoszeń usterek.
    - Zarządzanie miejscami parkingowymi.
    - Zarządzanie regulaminem.
4. Akademik:
    - Administrator systemu lub manager mają możliwość zarządzania informacjami o akademiku np. liczba pokoi, lokalizacja.
5. Pokój:

- Każdy pokój powinien mieć przypisane wyposażenie. System umożliwia podgląd i edycję wyposażenia pokoju.
- Możliwość zarządzania statusem pokoju (dostępny, zajęty, naprawa).
- Dane na temat pokojów może aktualizować administrator i manager akademiku.
- Pokój może być wynajmowany przez wielu studentów.

6. Umowa pracownika:
    - Możliwość tworzenia i zarządzania umowami o pracę.
    - System powinien umożliwiać złożenie wypowiedzenia umowy przez pracownika lub pracodawcę.
7. Wniosek:
    - Studencie mogą składać wnioski o wynajem pokoju poprzez system.
    - Możliwość śledzenia, czy wniosek został zaakceptowany, odrzucony czy jest w trakcie przetwarzania.
8. Opłata:
    - Możliwość podglądu rachunków za wynajem pokoju oraz innych opłat.
    - Integracja z systemem płatności, umożliwiająca opłacanie rachunków.
9. Zgłoszenie usterki:
    - Studenci oraz pracownicy mogą zgłaszać usterki w pokojach lub innych częściach akademika.
    - Możliwość śledzenia statusu zgłoszonych usterek i napraw.
10. Regulamin:
    - Studenci i pracownicy mogą przeglądać regulamin akademika w systemie.
    - Przy rejestracji użytkownicy muszą zaakceptować regulamin akademika.
11. Parking:
    - Podgląd szczegółów miejsca parkingowego przez użytkownika.
    - Administrator ma możliwość dodawania, usuwania i edytowania miejsc parkingowych.
12. Umowa najmu:
    - Administrator może ręcznie edytować szczegóły umowy.
    - Student może zgłosić wypowiedzenie umowy najmu przez system.
    - Wypowiedzenie jest rejestrowane, a administrator otrzymuje powiadomienie.
    - Użytkownicy mają dostęp do aktualnych i archiwalnych umów najmu.
    - System umożliwia pobranie umowy w formacie PDF.
13. Sprzęt w akademiku:
    - System umożliwia aktualizowanie sprzętu należącego do akademika.
    - Administrator może zarządzać historią napraw i zgłoszeń dotyczących sprzętu.
    - Administrator ma dostęp do historii sprzątania poszczególnych pokoi i może generować raporty o realizacji grafików.

2. Faza logiczna

Definicja schematów relacji:

1. Student

Nazwa atrybutu	Typ atrybutu
id	int
nazwisko	varchar
imie	varchar
pesel	varchar
data_ur	date
adres_zam	varchar
plec	varchar
nr_albumu	int
status	enum(‘aktywny’, ‘nieaktywny’)
email	varchar
haslo	varchar
nr_telefonu	varchar

Student(**id**, nazwisko, imie, pesel, data_ur, adres_zam, plec, nr_albumu, status, email, haslo, nr_telefonu)

2. Pracownik

Nazwa atrybutu	Typ atrybutu
id	int
nazwisko	varchar
imie	varchar
pesel	varchar
data_ur	date
adres_zam	varchar
plec	varchar
email	varchar
haslo	varchar
nr_telefonu	varchar

Pracownik(**id**, nazwisko, imie, pesel, data_ur, adres_zam, plec, email, haslo, nr_telefonu)

3. Stanowisko

Nazwa atrybutu	Typ atrybutu
id	int
nazwa	varchar
obowiazki	varchar

Stanowisko(**id**, nazwa, obowiazki)

4. Umowa pracownika

Nazwa atrybutu	Typ atrybutu
id	int
data_zawarcia	date
data_rozwiazania	date

okres_trwania	varchar
rodzaj_umowy	varchar
wymiar_czasu_pracy	int
pensja	float

Umowa pracownika(**id**, data_zawarcia, data_rozwiazania, okres_trwania, rodzaj_umowy, wymiar_czasu_pracy, pensja, **#id_pracownika**, **#id_stanowiska**, **#id_akademika**)

5. Akademik

Nazwa atrybutu	Typ atrybutu
id	int
nazwa	varchar
adres	varchar

Akademik(**id**, nazwa, adres, **#id_regulaminu**)

6. Regulamin

Nazwa atrybutu	Typ atrybutu
id	int
tresc	varchar

Regulamin(**id**, tresc)

7. Parking

Nazwa atrybutu	Typ atrybutu
id	int
adres	varchar
ilosc_miejsc	int

Parking(**id**, adres, ilość_miejsc, **#id_akademika**)

8. Pokój

Nazwa atrybutu	Typ atrybutu
id	int
numer	int
max_ilosc_osob	int
status_pokoju	varchar

Pokój(**id**, numer, max_ilosc_miejsc,status_pokoju, **#id_akademika**)

9. Sprzęt

Nazwa atrybutu	Typ atrybutu
id	int
data_zakupu	date
stan	varchar

kategoria	varchar

Sprzęt(**id**, data_zakupu, stan, **#id_kategorii, #id_pokoju**)

10. Kategoria sprzętu

Nazwa atrybutu	Typ atrybutu
id	int
nazwa	varchar

Kategoria sprzętu(**id**, nazwa)

11. Zgłoszenie usterki

Nazwa atrybutu	Typ atrybutu
id	int
data_zgloszenia	date
status	enum('wysłane', 'w trakcie naprawy', 'zakończone')

Zgłoszenie usterki(**id**, data_zgloszenia, status, **#id_studenta, #id_sprzetu**)

12. Umowa najmu

Nazwa atrybutu	Typ atrybutu
id	int
data_zawarcia	date
data_rozwiazania	date
oplata	float

Umowa najmu(**id**, data_zawarcia, data_rozwiazania, oplata, **#id_pracownika, #id_akademika, #id_studenta, #id_pokoju**)

13. Wniosek

Nazwa atrybutu	Typ atrybutu
id	int
data_zlozenia	date
status	enum('zarejestrowany', 'gotowy do rozpatrzenia', 'rozpatrzony pozytywnie', 'odrzucony')
dochod	float

Wniosek(**id**, data_zlozenia, status, dochod, **#id_studenta**)

14. Opłata

Nazwa atrybutu	Typ atrybutu
id	int

wartosc	float
rodzaj	varchar
status	enum(‘oczekująca’, ‘w trakcie przetwarzania’, ‘zaksięgowana’, ‘zaległa’)

Opłata(**id**, wartosc, rodzaj, status, **#id_studenta**)

Pierwsza postać normalna (1NF): Każda kolumna zawiera wartości atomowe (niepodzielne) - warunek spełniony.

Druga postać normalna (2NF): Tabele w 1NF, bez zależności częściowych (wszystkie atrybuty niekluczowe są zależne od całego klucza głównego) - warunek spełniony.

Trzecia postać normalna (3NF): Tabele w 2NF, bez zależności przechodnich (żaden atrybut niekluczowy nie zależy od innych atrybutów niekluczowych) - warunek spełniony.

Po przeanalizowaniu stwierdzamy, że schematy są w III postaci normalnej.