# Зміст

# ВСТУП

У курсовій роботі потрібно розробити програмний додаток згідно вимог MVC  для управління навчальним процесом в університеті (реалізувати функціонал для адміністратора).

У роботі розглядаються основні принципи MVC, принципи побудови графічного інтерфейсу користувача, основні елементи інтерфейсу, організація обробки подій, система відлову помилок і організація багатопоточності.

Розробка модулю  виконується на мові програмування Java. Робота містить повну документацію, а також програмний код проекту та UML діаграму класів.

# 1 ТЕХНІЧНЕ ЗАВДАННЯ

## 1.1 Загальне завдання

Необхідно розробити програмний додаток із використанням MVC для управління навчальним процесом в університеті.

## 1.2 Вимоги до функціональності

Для адміністрації університету система надаватиме наступні можливості :

- Внести зміни до розкладу
- Отримати статистику успішності/відвідування/атестацій
- Залишити повідомлення будь-якому працівнику/студенту
- Отримати інформацію про будь-якого працівника/студента
- Повідомити батьків про успіхи/неуспіхи студента
- Ефективно вести контроль навчального процесу
- Переглянути/залишати новини
- Редагувати/видалити акаунт

## 1.3 Вимоги до реалізації

- забезпечити надійність системи
  - резервне копіювання та відновлення даних
  - захист від зловмисних атак
  - робота в режимі великих навантажень
- мова програмування Java;
- Відповідність вимогам MVC
- інтерфейс адміністратора повинен забезпечувати контроль над всією функціональністю системи
- проект має бути повністю задокументований за допомогою JavaDoc;
- проект має повністю відповідати правилам CheckStyle;

## 1.4 Огляд парадигми MVC

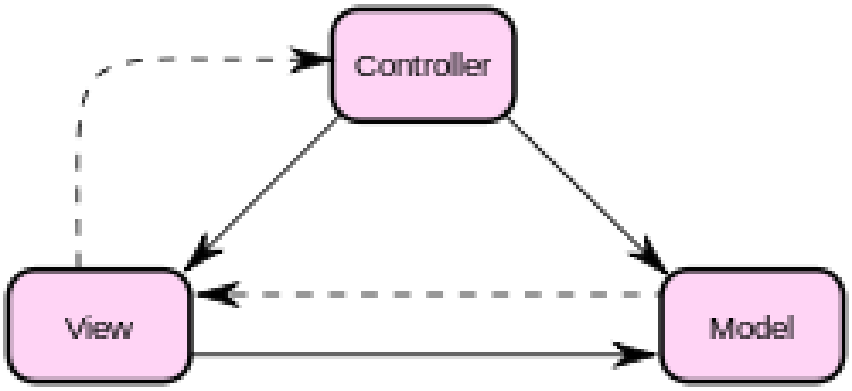


Рис. 1.1 Діаграма взаємодії між компонентами шаблону

Моде́ль-вид-контро́лер (*Model-view-controller*, MVC) — архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.

Цей шаблон поділяє систему на три частини: модель даних, вигляд даних та керування. Застосовується для відокремлення даних (модель) від інтерфейсу користувача (вигляду) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

Мета шаблону — гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах призводить до певної впорядкованості їх структури і робить їх зрозумілішими завдяки зменшенню складності.

## 1.5 Компоненти MVC

Архітектурний шаблон Модель-Вид-Контролер (MVC) поділяє програму на три частини. У тріаді до обов'язків компоненту Модель (Model) входить зберігання даних і забезпечення інтерфейсу до них. Вигляд (View) відповідальний за представлення цих даних користувачеві. Контролер (Controller) керує

компонентами, отримує сигнали у вигляді реакції на дії користувача, і повідомляє про зміни компоненту Модель. Така внутрішня структура в цілому поділяє систему на самостійні частини і розподіляє відповідальність між різними компонентами.

MVC поділяє цю частину системи на три самостійні частини: введення даних, компонент обробки даних і виведення інформації. Модель, як вже було відмічено, інкапсулює ядро даних і основний функціонал з їх обробки. Також компонент Модель не залежить від процесу введення або виведення даних. Компонент виводу Вигляд може мати декілька взаємопов'язаних областей, наприклад, різні таблиці і поля форм, в яких відображається інформація. У функції Контролера входить моніторинг за подіями, що виникають в результаті дій користувача (зміна положення курсора миші, натиснення кнопки або введення даних в текстове поле).

Зареєстровані події транслюються в різні запити, що спрямовуються компонентам Моделі або об'єктам, відповідальним за відображення даних. Відокремлення моделі від вигляду даних дозволяє незалежно використовувати різні компоненти для відображення інформації. Таким чином, якщо користувач через Контролер внесе зміни до Моделі даних, то інформація, подана одним або декількома візуальними компонентами, буде автоматично відкоригована відповідно до змін, що відбулися.

У мові програмування Java концепція MVC підтримується на рівні стандартних класів-бібліотек. В результаті використання парадигми MVC програміст отримує в своє розпорядження могутню структуру об'єктів-компонентів, функції яких чітко розмежовані, що гарантує надійність і розширюваність системи, що розробляється.