Система організації навчального процесу в університеті “KPI City”

Виконав
Студент 2-курсу
Групи ІО-22
Коломієць Олег
Перевірив
Болдак А. О.

Київ 2013

1. Запити зацікавлених осіб

1.1 Вступ

У даному документі описуються запити зацікавлених осіб котрими виступає замовник – Національний технічний університет України – «Київський політехнічний інститут», для якого розроблюється система організації навчального процесу в університеті “KPI City”.

1.1.1 Мета

Метою документу є визначити головні вимоги до функціональності, продуктивності, надійності, зручності, доступності, а також визначити бізнес-правила та технологічні обмеження, що накладаються на предмет розробки.

1.1.2 Контекст

Перелік вимог, зазначених у даному документі, є основою технічного завдання для розробки системи організації навчального процесу в НТУУ «КПІ» “KPI City”.

2.1 Короткий огляд продукту

Система організації навчального процесу в НТУУ «КПІ» “KPI City” – це база даних всіх студентів та працівників університету разом із програмними продуктами, надають 3 різні рівні доступу. Студенти (перший рівень) можуть швидко знайти інформацію про предмети, викладачів. Також можна залишити повідомлення чи отримати консультацію, вести моніторинг своєї успішності. Викладачі(другий рівень) зможуть швидко виставляти оцінки, атестації, отримати інформацію про студентів чи розклад, робити оголошення. Адміністрація університету(третій рівень) зможе ефективно організувати та контролювати навчальний процес, вносити зміни, швидко отримувати інформацію та вести статистику.

3.1 Ділові правила

3.1.1 Призначення системи “KPI City”

Система призначена для збору інформації про студентів та викладачів із подальшим її використанням на різних рівнях доступу для полегшення організації навчального процесу та економії часу та зусиль студентів та викладачів університету.

3.1.2 Політика взаємовідносин із клієнтом

Клієнтами системи “KPI City” можуть бути студенти, аспіранти, спеціалісти, професори, працівники адміністрації, що навчаються та/або працюють у даному університеті.

Політика взаємовідносин із клієнтом системи “KPI City” полягає в наданні йому різного роду інформації, в основному із допомогою веб-сервісів, із можливістю чи неможливістю внесення змін до неї.

3.1.3 Характеристика ділового процесу

Керівництво системою “KPI City” здійснюється адміністраторами, що взаємодіють із керівництвом університету.

Адміністратори системи “KPI City” несуть відповідальність за створення, збереження та видалення інформації будь-якого роду.

Керівництво університету має доступ на рівні адміністраторів й також несе відповідальність за створення, збереження, видалення чи внесення змін до інформації будь-якого роду.

Викладачі мають доступ до інформації про розклад та групи, у яких викладають, й також несуть відповідальність за створення, збереження, видалення чи внесення змін до даної інформації.

Студенти системи мають доступ лише до ознайомлювальної частини системи (оцінки, атестації, відвідування, розклад, екзамени) й несуть відповідальність за використання особистих даних інших користувачів системи.

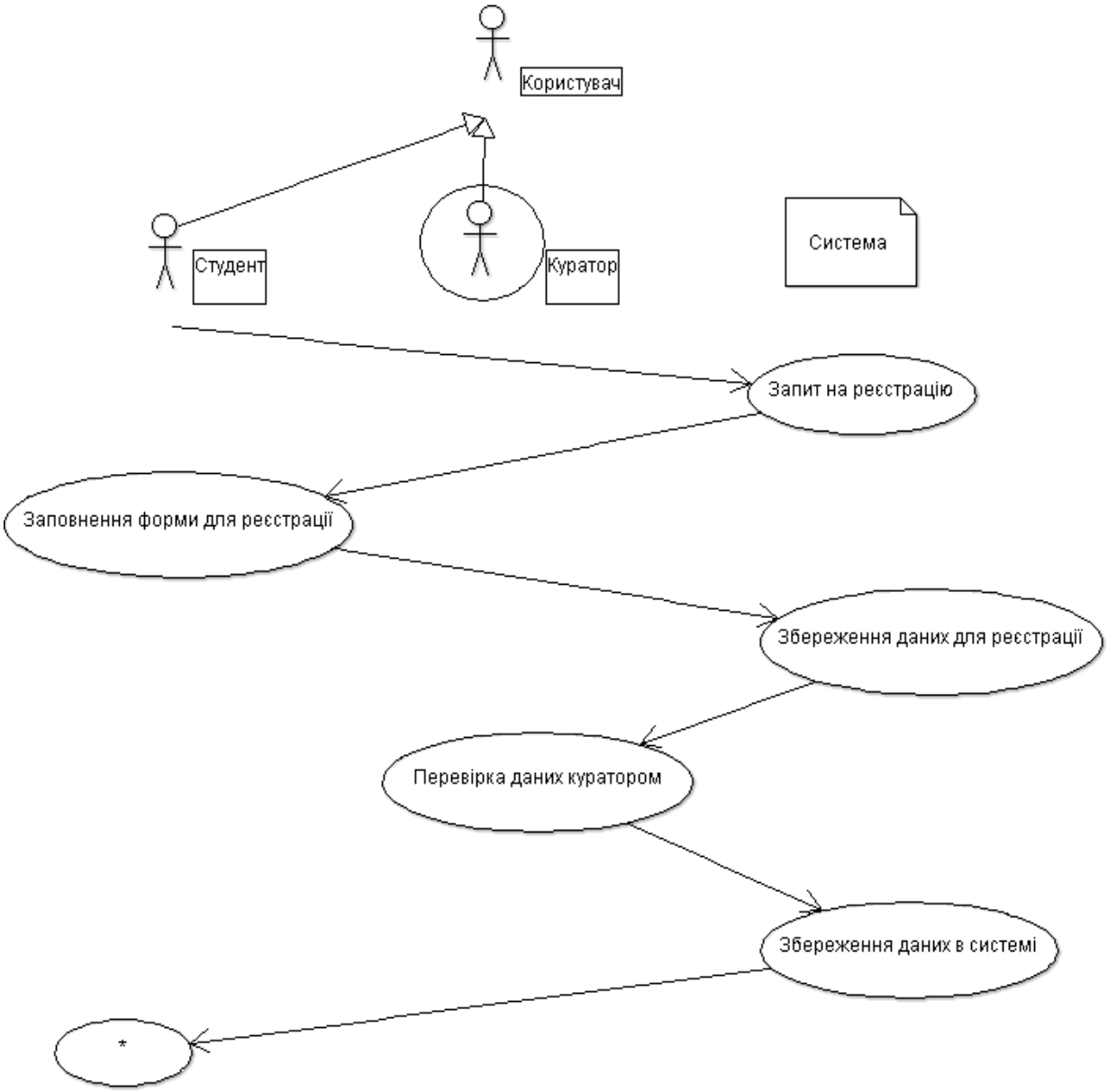
3.1.4 Сценарій реєстрації нового користувача

Студент реєструється самостійно, заповнюючи реєстраційну форму на веб-сайті, вводить усі необхідні дані (логін, пароль, підтвердження паролю, ім’я, прізвище, по-батькові, e-mail, дійсний контактний телефон, контакти батьків, адресу проживання, ін.). Зі створеного акаунту студент відправляє запит куратору своєї групи, куратор перевіряє дані та включає студента до списків відповідної групи.

Викладач або працівник адміністрації реєструється також самостійно, згодом відсилаючи запит адміністраторам системи, які повинні перевірити введену інформацію та надати відповідні права доступу.

3.1.5 Сценарії різних послуг системи

3.1.5.1 Реєстрація нового користувача



4.1 Функціональність

Головні вимоги щодо функціональності, пред’явлені зацікавленими особами до предмету розробки, відносяться до чотирьох категорій :

- Студент
- Викладач, викладач-куратор
- Керівництво університету
- Адміністратори системи

4.1.1 Послуги для студента

Для студентів система надаватиме такі послуги :

- Переглянути успішність
- Знайти викладача

- Залишити повідомлення для викладача
- Залишити повідомлення для всієї групи
- Замовити довідку
- Переглянути останні новини, повідомлення
- Редагувати акаунт

4.2.1 Можливості для викладача, куратора

Для викладачів система надаватиме наступні можливості :

- Отримати розклад
- Виставити оцінки/атестації
- Оголосити групі оцінки/атестації
- Залишити повідомлення групі/окремому студенту
- Отримати інформацію про студента
- Переглянути/залишати новини, повідомлення
- Редагувати акаунт
- Перевірити правильність реєстраційних даних студента

4.3.1 Можливості для адміністрації університету

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

4.4.1 Можливості для адміністраторів системи

Для адміністраторів система повинна надавати наступні можливості:

- Створити/редагувати/видалити акаунт за згодою керівництва університету
- Розширити/обмежити права користувачів
- Додати/видалити теми на стрічці новин
- Залишити новини
- Обмежувати можливості користувача за порушення правил (нецензурну лексику, неправдиві повідомлення, неправильні дані)

5.1 Практичність

5.1.1 Мобільність

Веб-сайт повинен бути оптимізованим для роботи не тільки із комп’ютера, а також із мобільних пристроїв.

6.1 Надійність

6.1.1 Резервне копіювання та відновлення даних

Повинно проводитись резервне копіювання баз даних

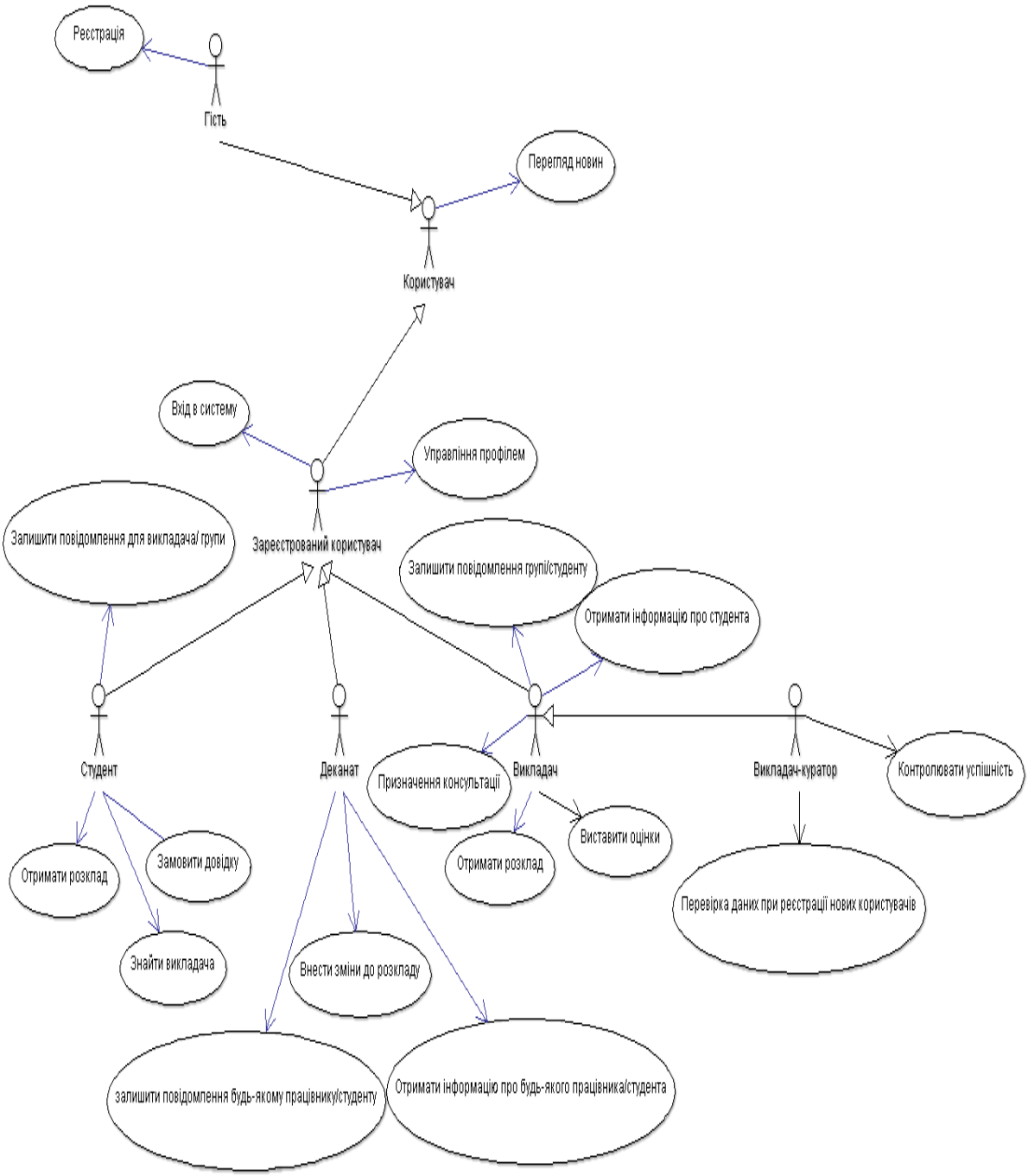
6.1.2 Захист від зловмисних атак

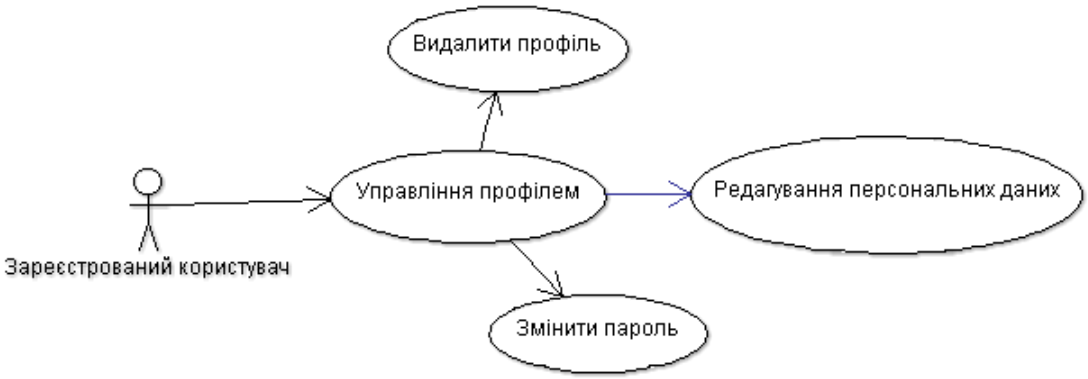
Система повинна бути добре захищена від різного роду зловмисних атак із метою заволодіння інформації чи атак типу DDoS.

6.1.3 Великі навантаження

Система повинна витримувати великі навантаження, обслуговуючи значну кількість користувачів.

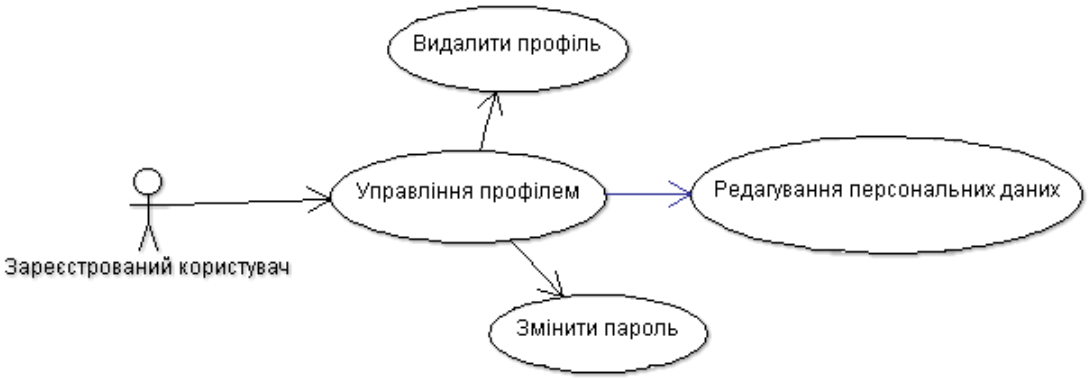
Use-case діаграми


Реєстрація
Гість
Перегляд новин
Користувач
Вхід в систему
Управління профілем
Зареєстрований користувач
Залишити повідомлення для викладача/ групи
Залишити повідомлення групі/студенту
Отримати інформацію про студента
Студент
Деканат
Викладач
Викладач-куратор
Контролювати успішність
Призначення консультації
Виставити оцінки
Отримати розклад
Замовити довідку
Отримати розклад
Знайти викладача
Внести зміни до розкладу
Перевірка даних при реєстрації нових користувачів
залишити повідомлення будь-якому працівнику/студенту
Отримати інформацію про будь-якого працівника/студента


Видалити профіль
Управління профілем
Редагування персональних даних
Зареєстрований користувач
Змінити пароль

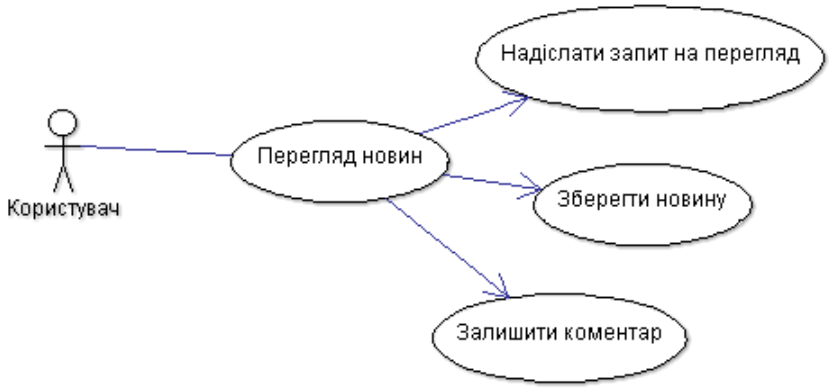

Ввести логін та пароль
Вхід в систему
Зареєстрований користувач
Згадати пароль

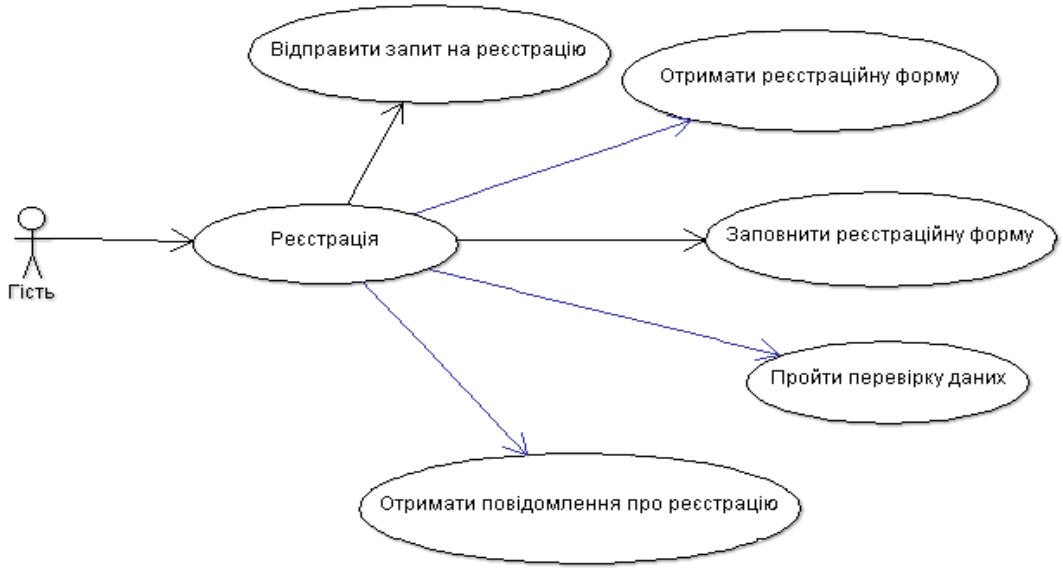

Перевірити стан виконання
Обрати тип довідки
Замовити довідку
Заповнити форму для довідки
Студент
Відмінити замовлення


Видалити профіль
Управління профілем
Редагування персональних даних
Зареєстрований користувач
Змінити пароль


Ввести логін та пароль
Вхід в систему
Зареєстрований користувач
Згадати пароль

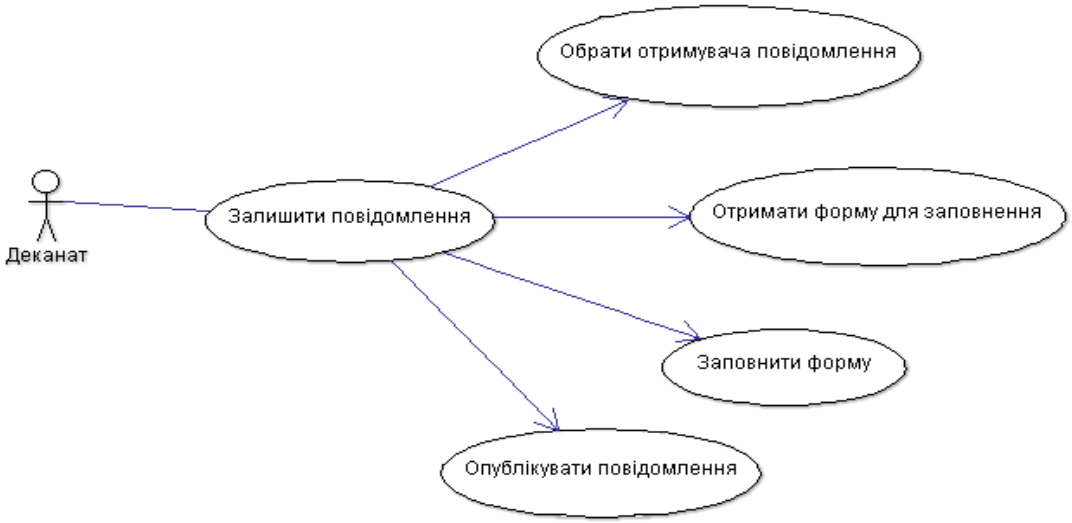

Перевірити стан виконання
Обрати тип довідки
Замовити довідку
Студент
Заповнити форму для довідки
Відмінити замовлення

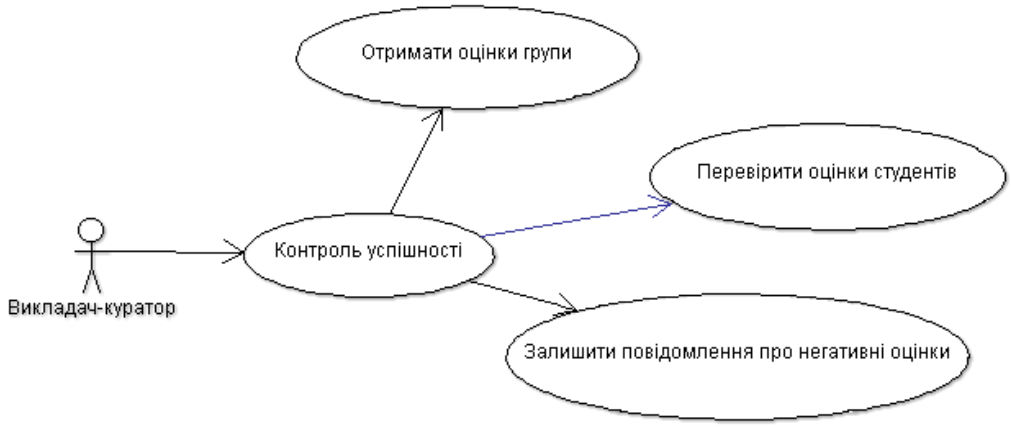

Надіслати запит на перегляд
Перегляд новин
Користувач
Зберегти новину
Залишити коментар

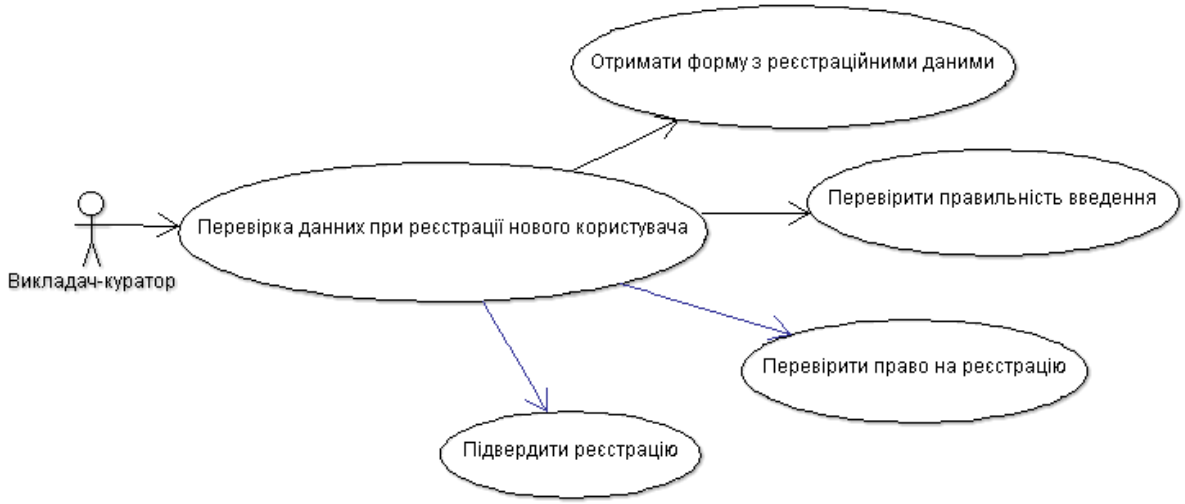

Відправити запит на реєстрацію
Отримати реєстраційну форму
Гість
Реєстрація
Заповнити реєстраційну форму
Пройти перевірку даних
Отримати повідомлення про реєстрацію


Отримати форму для заповнення
Викладач
Залишити повідомлення
Заповнити форму
Опублікувати повідомлення

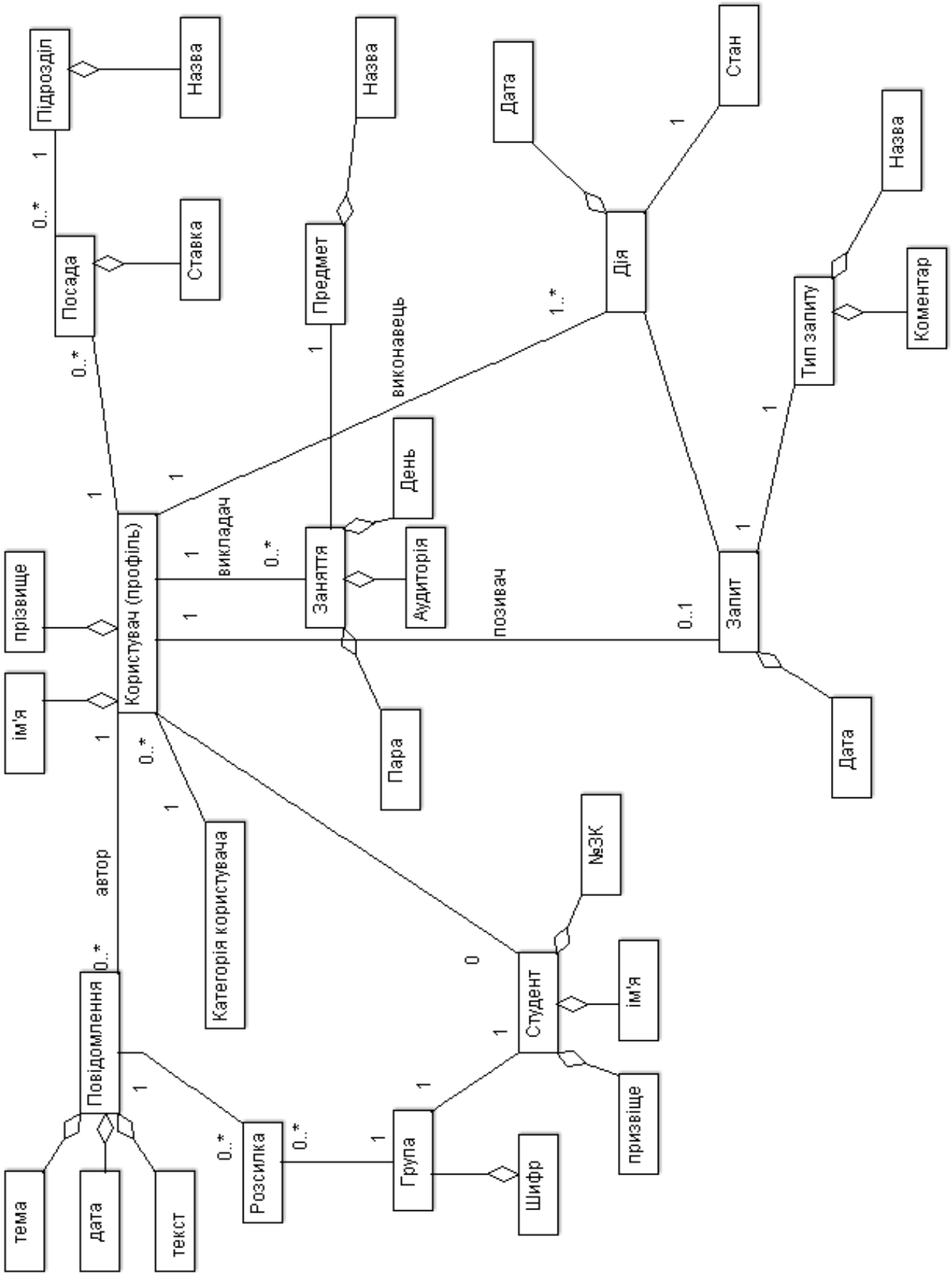

Отримати форму із розкладом
Внести зміни
Деканат
Внести зміни до розкладу
Підтвердити зміни
Опублікувати розклад


Обрати отримувача повідомлення
Деканат
Залишити повідомлення
Отримати форму для заповнення
Заповнити форму
Опублікувати повідомлення

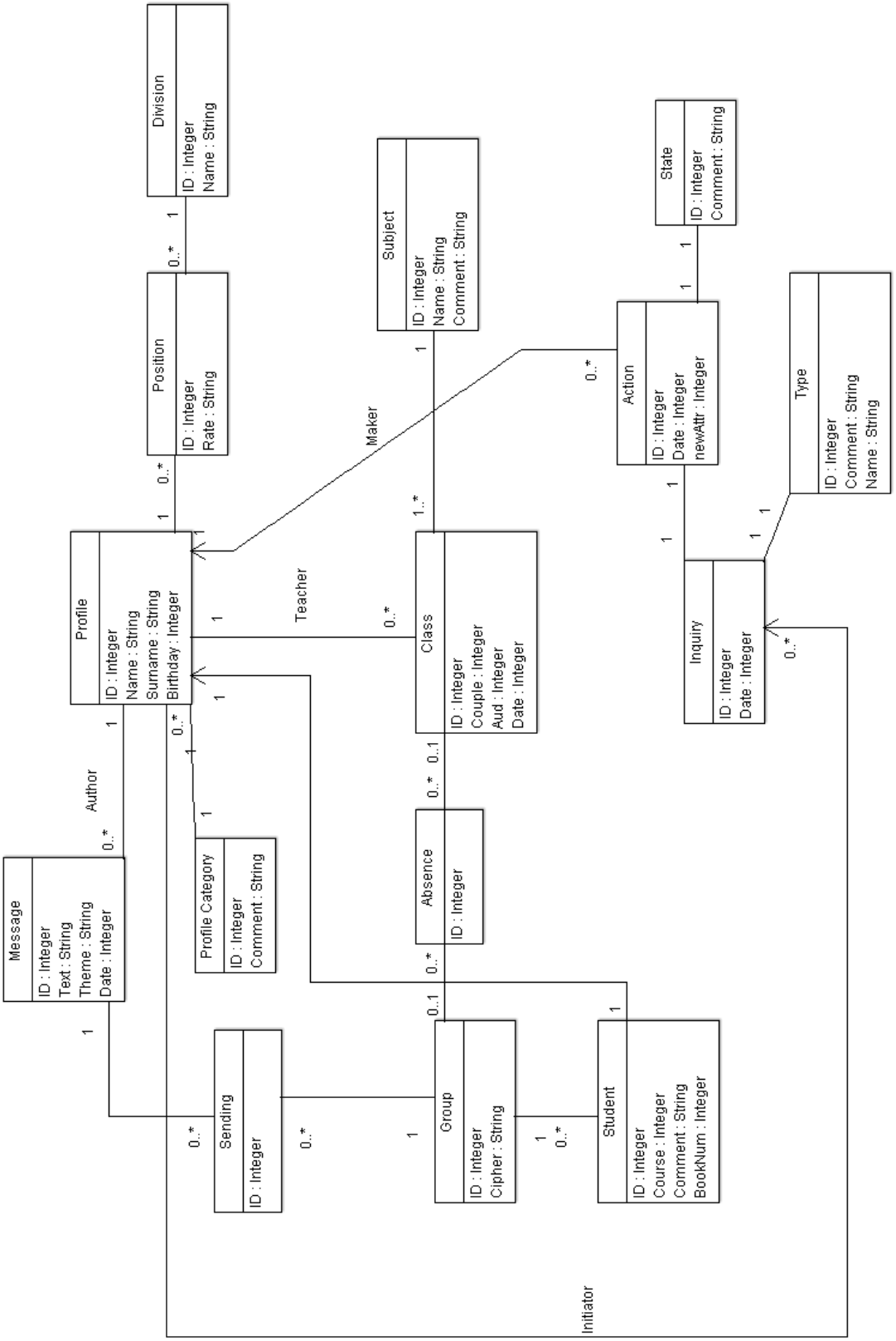

Отримати оцінки групи
Перевірити оцінки студентів
Викладач-куратор
Контроль успішності
Залишити повідомлення про негативні оцінки


Отримати форму з реєстраційними даними
Перевірити правильність введення
Викладач-куратор
Перевірка данних при реєстрації нового користувача
Перевірити право на реєстрацію
Підвердити реєстрацію

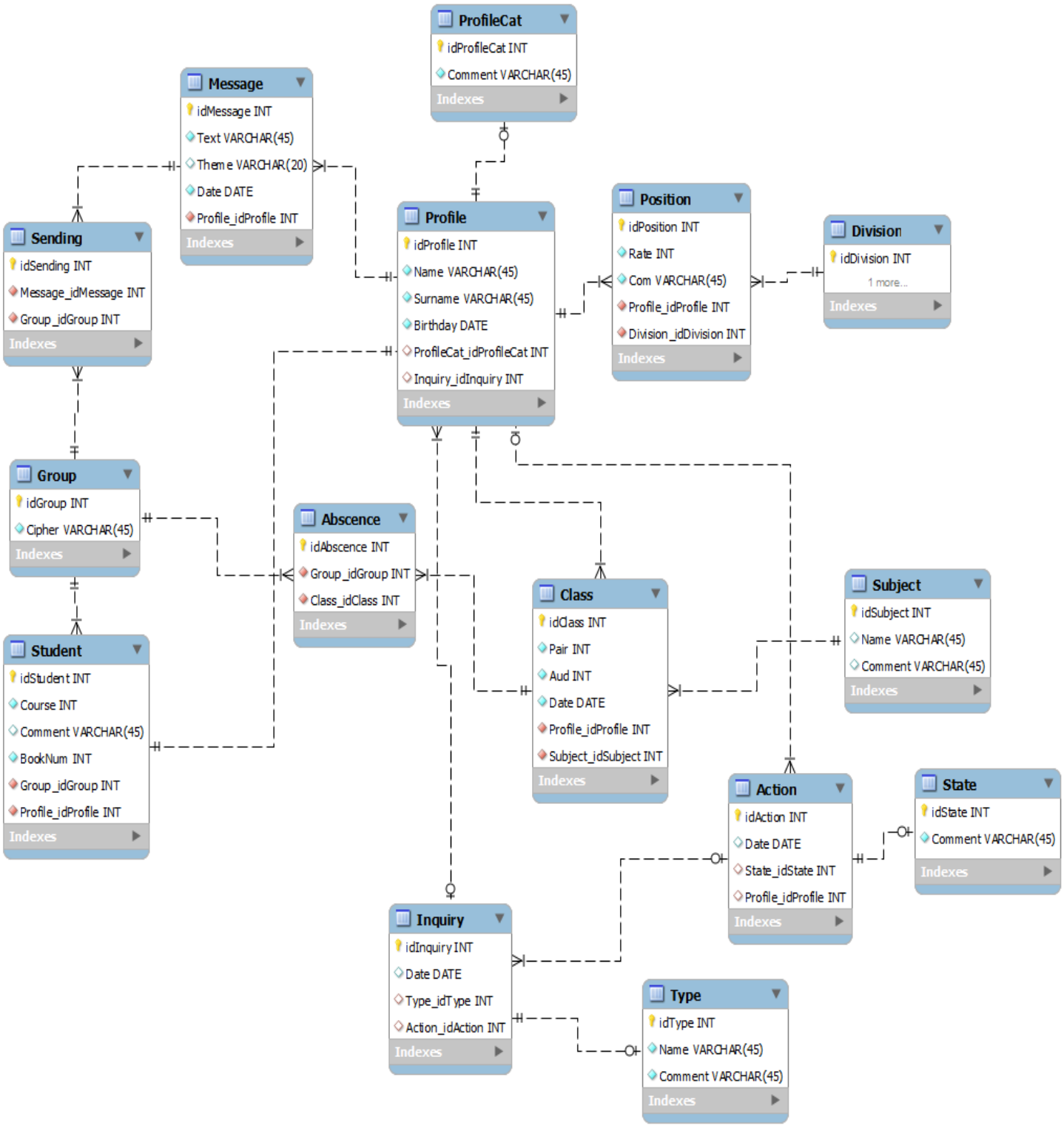
Діаграма бізнес- сутностей



Діаграма класів



Реляційна таблиця бази даних



SQL-код створення таблиці

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci ;

CREATE TABLE IF NOT EXISTS `mydb`.`Profile` (
  `idProfile` INT(11) NOT NULL AUTO_INCREMENT,
  `Name` VARCHAR(45) NOT NULL,
  `Surname` VARCHAR(45) NOT NULL,
  `Birthday` DATE NOT NULL,
  `ProfileCat_idProfileCat` INT(11) NOT NULL,
  `Inquiry_idInquiry` INT(11) NOT NULL,
  PRIMARY KEY (`idProfile`),
  INDEX `fk_Profile_ProfileCat1_idx` (`ProfileCat_idProfileCat` ASC),
  INDEX `fk_Profile_Inquiry1_idx` (`Inquiry_idInquiry` ASC),
  CONSTRAINT `fk_Profile_ProfileCat1`
    FOREIGN KEY (`ProfileCat_idProfileCat`)
    REFERENCES `mydb`.`ProfileCat` (`idProfileCat`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Profile_Inquiry1`
    FOREIGN KEY (`Inquiry_idInquiry`)
    REFERENCES `mydb`.`Inquiry` (`idInquiry`)
    ON DELETE NO ACTION
```

```
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Message` (
  `idMessage` INT(11) NOT NULL AUTO_INCREMENT,
  `Text` VARCHAR(45) NOT NULL,
  `Theme` VARCHAR(20) NULL DEFAULT NULL,
  `Date` DATE NOT NULL,
  `Profile_idProfile` INT(11) NOT NULL,
  PRIMARY KEY (`idMessage`),
  INDEX `fk_Message_Profile1_idx` (`Profile_idProfile` ASC),
  CONSTRAINT `fk_Message_Profile1`
    FOREIGN KEY (`Profile_idProfile`)
    REFERENCES `mydb`.`Profile` (`idProfile`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`ProfileCat` (
  `idProfileCat` INT(11) NOT NULL AUTO_INCREMENT,
  `Comment` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idProfileCat`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`Sending` (
  `idSending` INT(11) NOT NULL AUTO_INCREMENT,
  `Message_idMessage` INT(11) NOT NULL,
  `Group_idGroup` INT(11) NOT NULL,
  PRIMARY KEY (`idSending`),
  INDEX `fk_Sending_Message1_idx` (`Message_idMessage` ASC),
  INDEX `fk_Sending_Group1_idx` (`Group_idGroup` ASC),
  CONSTRAINT `fk_Sending_Message1`
    FOREIGN KEY (`Message_idMessage`)
    REFERENCES `mydb`.`Message` (`idMessage`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Sending_Group1`
    FOREIGN KEY (`Group_idGroup`)
    REFERENCES `mydb`.`Group` (`idGroup`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Position` (
  `idPosition` INT(11) NOT NULL AUTO_INCREMENT,
  `Rate` INT(11) NOT NULL,
  `Com` VARCHAR(45) NOT NULL,
  `Profile_idProfile` INT(11) NOT NULL,
  `Division_idDivision` INT(11) NOT NULL,
  PRIMARY KEY (`idPosition`),
```

```
  INDEX `fk_Position_Profile_idx` (`Profile_idProfile` ASC),
  INDEX `fk_Position_Division1_idx` (`Division_idDivision` ASC),
  CONSTRAINT `fk_Position_Profile`
    FOREIGN KEY (`Profile_idProfile`)
    REFERENCES `mydb`.`Profile` (`idProfile`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Position_Division1`
    FOREIGN KEY (`Division_idDivision`)
    REFERENCES `mydb`.`Division` (`idDivision`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Division` (
  `idDivision` INT(11) NOT NULL AUTO_INCREMENT,
  `Name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idDivision`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Group` (
  `idGroup` INT(11) NOT NULL AUTO_INCREMENT,
  `Cipher` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idGroup`))
ENGINE = InnoDB
```

```
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Student` (
  `idStudent` INT(11) NOT NULL AUTO_INCREMENT,
  `Course` INT(11) NOT NULL,
  `Comment` VARCHAR(45) NULL DEFAULT NULL,
  `BookNum` INT(11) NOT NULL,
  `Group_idGroup` INT(11) NOT NULL,
  `Profile_idProfile` INT(11) NOT NULL,
  PRIMARY KEY (`idStudent`),
  INDEX `fk_Student_Group1_idx` (`Group_idGroup` ASC),
  INDEX `fk_Student_Profile1_idx` (`Profile_idProfile` ASC),
  CONSTRAINT `fk_Student_Group1`
    FOREIGN KEY (`Group_idGroup`)
    REFERENCES `mydb`.`Group` (`idGroup`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Student_Profile1`
    FOREIGN KEY (`Profile_idProfile`)
    REFERENCES `mydb`.`Profile` (`idProfile`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Abscence` (
  `idAbscence` INT(11) NOT NULL AUTO_INCREMENT,
```

```
  `Group_idGroup` INT(11) NOT NULL,
  `Class_idClass` INT(11) NOT NULL,
  PRIMARY KEY (`idAbscence`),
  INDEX `fk_Abscence_Group1_idx` (`Group_idGroup` ASC),
  INDEX `fk_Abscence_Class1_idx` (`Class_idClass` ASC),
  CONSTRAINT `fk_Abscence_Group1`
    FOREIGN KEY (`Group_idGroup`)
    REFERENCES `mydb`.`Group` (`idGroup`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Abscence_Class1`
    FOREIGN KEY (`Class_idClass`)
    REFERENCES `mydb`.`Class` (`idClass`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Class` (
  `idClass` INT(11) NOT NULL AUTO_INCREMENT,
  `Pair` INT(11) NOT NULL,
  `Aud` INT(11) NOT NULL,
  `Date` DATE NOT NULL,
  `Profile_idProfile` INT(11) NOT NULL,
  `Subject_idSubject` INT(11) NOT NULL,
  PRIMARY KEY (`idClass`),
  INDEX `fk_Class_Profile1_idx` (`Profile_idProfile` ASC),
  INDEX `fk_Class_Subject1_idx` (`Subject_idSubject` ASC),
```

```
  CONSTRAINT `fk_Class_Profile1`
    FOREIGN KEY (`Profile_idProfile`)
    REFERENCES `mydb`.`Profile` (`idProfile`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Class_Subject1`
    FOREIGN KEY (`Subject_idSubject`)
    REFERENCES `mydb`.`Subject` (`idSubject`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;


CREATE TABLE IF NOT EXISTS `mydb`.`Inquiry` (
  `idInquiry` INT(11) NOT NULL AUTO_INCREMENT,
  `Date` DATE NOT NULL,
  `Type_idType` INT(11) NOT NULL,
  `Action_idAction` INT(11) NOT NULL,
  PRIMARY KEY (`idInquiry`),
  INDEX `fk_Inquiry_Type1_idx` (`Type_idType` ASC),
  INDEX `fk_Inquiry_Action1_idx` (`Action_idAction` ASC),
  CONSTRAINT `fk_Inquiry_Type1`
    FOREIGN KEY (`Type_idType`)
    REFERENCES `mydb`.`Type` (`idType`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Inquiry_Action1`
    FOREIGN KEY (`Action_idAction`)
```

```
    REFERENCES `mydb`.`Action` (`idAction`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Action` (
  `idAction` INT(11) NOT NULL AUTO_INCREMENT,
  `Date` DATE NOT NULL,
  `State_idState` INT(11) NOT NULL,
  `Profile_idProfile` INT(11) NOT NULL,
  PRIMARY KEY (`idAction`),
  INDEX `fk_Action_State1_idx` (`State_idState` ASC),
  INDEX `fk_Action_Profile1_idx` (`Profile_idProfile` ASC),
  CONSTRAINT `fk_Action_State1`
    FOREIGN KEY (`State_idState`)
    REFERENCES `mydb`.`State` (`idState`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Action_Profile1`
    FOREIGN KEY (`Profile_idProfile`)
    REFERENCES `mydb`.`Profile` (`idProfile`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`Type` (
  `idType` INT(11) NOT NULL AUTO_INCREMENT,
  `Name` VARCHAR(45) NOT NULL,
  `Comment` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idType`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`State` (
  `idState` INT(11) NOT NULL AUTO_INCREMENT,
  `Comment` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idState`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;

CREATE TABLE IF NOT EXISTS `mydb`.`Subject` (
  `idSubject` INT(11) NOT NULL AUTO_INCREMENT,
  `Name` VARCHAR(45) NULL DEFAULT NULL,
  `Comment` VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (`idSubject`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_general_ci;



SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
```

```
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Заповнення таблиць

```
INSERT INTO `kpi`.`profilecat` (`idProfileCat`, `Comment`) VALUES ('4',
'fsfgsdfg');
INSERT INTO `kpi`.`state` (`idState`, `Comment`) VALUES ('4', 'sdgsfg');
INSERT INTO `kpi`.`inquiry` (`idInquiry`, `Date`, `Type_idType`,
`Action_idAction`) VALUES ('1', '1992-01-01', '1', '1');
INSERT INTO
`kpi`.`inquiry` (`idInquiry`, `Date`, `Type_idType`, `Action_idAction`)
VALUES ('2', '1994-01-01', '2', '2');
INSERT INTO `kpi`.`inquiry`
(`idInquiry`, `Date`, `Type_idType`, `Action_idAction`) VALUES ('3', '1993-
03-03', '3', '3');
INSERT INTO `kpi`.`type` (`idType`, `Name`, `Comment`) VALUES ('1', 't1',
'comm');
INSERT INTO `kpi`.`type` (`idType`, `Name`, `Comment`) VALUES ('2',
't2', 'comm');
INSERT INTO `kpi`.`type` (`idType`, `Name`, `Comment`) VALUES
('3', 't3', 'comm');
INSERT INTO `kpi`.`group` (`idGroup`, `Cipher`) VALUES ('1', '123');
INSERT
INTO `kpi`.`group` (`idGroup`, `Cipher`) VALUES ('2', '34234');
INSERT INTO `kpi`.`subject` (`idSubject`, `Name`, `Comment`) VALUES ('1',
'sjkdfngs', 'slnfss');
UPDATE `kpi`.`action` SET `Date`='1000-03-02' WHERE `idAction`='3';
UPDATE `kpi`.`inquiry` SET `Action_idAction`='1' WHERE `idInquiry`='1';
```

```
UPDATE `kpi`.`inquiry` SET `Action_idAction`='2' WHERE `idInquiry`='2';
UPDATE `kpi`.`inquiry` SET `Action_idAction`='3' WHERE `idInquiry`='3';
INSERT INTO `kpi`.`class` (`idClass`, `Pair`, `Aud`, `Date`,
`Profile_idProfile`, `Subject_idSubject`) VALUES ('1', '4', '123', '1994-01
-01', '1', '1');
INSERT INTO `kpi`.`abscence` (`idAbscence`, `Group_idGroup`,
`Class_idClass`) VALUES ('1', '1', '1');
INSERT INTO `kpi`.`message` (`idMessage`, `Text`, `Theme`, `Date`,
`Profile_idProfile`) VALUES ('1', 'sf', 'wrgwrg', '1994-01-01', '1');
INSERT INTO `kpi`.`sending` (`idSending`, `Message_idMessage`,
`Group_idGroup`) VALUES ('1', '1', '1');
INSERT INTO `kpi`.`student` (`idStudent`, `Course`, `Comment`, `BookNum`,
`Group_idGroup`, `Profile_idProfile`) VALUES ('1', '2', 'lol', '2201', '1',
'1');
INSERT INTO `kpi`.`student` (`idStudent`, `Course`, `Comment`,
`BookNum`, `Group_idGroup`, `Profile_idProfile`) VALUES ('2', '2',
'trololo', '32', '1', '2');
INSERT INTO `kpi`.`division` (`idDivision`, `Name`) VALUES ('1', 'div')
INSERT INTO `kpi`.`position` (`idPosition`, `Rate`, `Com`,
`Profile_idProfile`, `Division_idDivision`) VALUES ('1', '2', 'sgdfg', '1',
'1');
```

Приклади таблиць :

idProfile	Name	Surname	Birthday	ProfileCat_idProfileCat	Inquiry_idInquiry
1	Andriy	Bas	1995-01-02	1	1
2	Oleh	Kolomietz	1995-03-03	2	2
3	Andrew	Budash	1997-04-04	3	3

idStudent	Course	Comment	BookNum	Group_idGroup	Profile_idProfile
1	2	Комент1	2201	1	1
2	2	Комент2	32	1	2

idClass	Pair	Aud	Date	Profile_idProfile	Subject_idSubject
1	4	123	2013-12-20	1	1

Запити до таблиць :

вивід імен профілів із конкретни idProfile

select name from profile where (profile.idProfile = 1);

name
Andriy

знайти повідомлення із конкретної дати

select idMessage from message where (message.Date = '1994-01-01');

idMessage
1

#знайти курси, номери книжок із студентів, які є в групах, і шифр останніх > 40

select Course, BookNum from student where (Group_idGroup in (select idGroup from kpi.group where(cipher > 40)));

Course	BookNum
2	2201
2	32