VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

DATABASE SYSTEMS (CO2013)
ASSIGNMENT REPORT

QUIZ MANAGEMENT SYSTEM

Instructor: Đỗ Thanh Thái, Ph.D.

Students: Nguyễn Thanh Bình - 2252082
Trần Quốc Bảo Long - 2252453
Đặng Duy Tiến - 2252808

Ho Chi Minh City, December 2024

CONTENTS

MEMBER LIST & WORKLOAD

No.	Full name	Student ID	Task	Percentage of work
1	Nguyễn Thanh Bình	2252082	Sketching, mapping and normalizing diagrams	100%
2	Trần Quốc Bảo Long	2252453	Preparing the SQL files Application development Preparing the report and presentation	100%
3	Đặng Duy Tiến	2252808	Listing the entities and attributes Defining architecture	100%

I. INTRODUCTION

A university wants to develop a system to manage and hold exams with multiple-choice questions and answers. The Quiz Management System (QMS) aims to streamline the process of creating, administering, and evaluating quizzes and exams. The system will cater to the needs of students, instructors, and administrators, ensuring a seamless and efficient examination process. By leveraging modern technology, the QMS will enhance the overall educational experience, providing a robust platform for academic assessments.

The database system for all lecturers in a university includes these information: Citizen ID, name, date of birth, gender, address and phone number. Lecturers assign quizzes and exams to students. *The value for Citizen ID is a string of 12 numbers as per national law, and the phone number is a string of 10 numbers.*

The database also holds students' information such as Student ID, Full name (surname and first name), Date of Birth, Gender, Address, Phone Number and Scores. A student can see and edit their personal information, as well as seeing their scores for past exams.

All lecturers and students may access the system using account information including username and password. All accounts are authenticated by HCMUT-SSO. There are also administrators who manage the system, each of them also have an account to login and access the system.

Each quiz goes with only one name, start date and time, and time limit. Quizzes contain multiple-choice questions with up to five possible answers, in which only one of them is correct.

II. COLLECTING AND ANALYZING REQUESTS

1. Requirement(s)

- Build a system for lecturers to create and assign multiple-choice exams and assign them to students.

2. Database specifications

- The QMS' database manages information such as: Account Information, Lecturers, Students, Quizzes, Scores.

- **Account Information** includes username and password. There are three types of accounts for Administrators, Lecturers and Students.

- **Lecturers** include: Citizen ID, Full name (surname and first name), Date of Birth, Gender, Address, Phone Number. Each lecturer has a unique account information and Citizen ID and can create and manage quizzes.

Requirements: Citizen ID is a string of 12 numbers per national law, and Phone Number is a string of 10 numbers.

- **Students** include: Student ID, Full name (surname and first name), Date of Birth, Gender, Address, Phone Number, Scores. Each student has a unique account information and Student ID. They can enter and submit quizzes. Scores are evaluated and are saved in a student's database.

Requirements: Student ID is a string of 7 numbers. The score is a decimal number and falls between 0 and 10.

- **Quizzes** include Question, up to five answers A, B, C, D, E, and Correct Answer. Lecturers may set start date and time as well as time limit for a quiz.

3. User groups

- Students:
    - View and take assigned quizzes.
    - Check personal scores and performance on completed quizzes.
    - Update personal details such as email, phone number, etc.

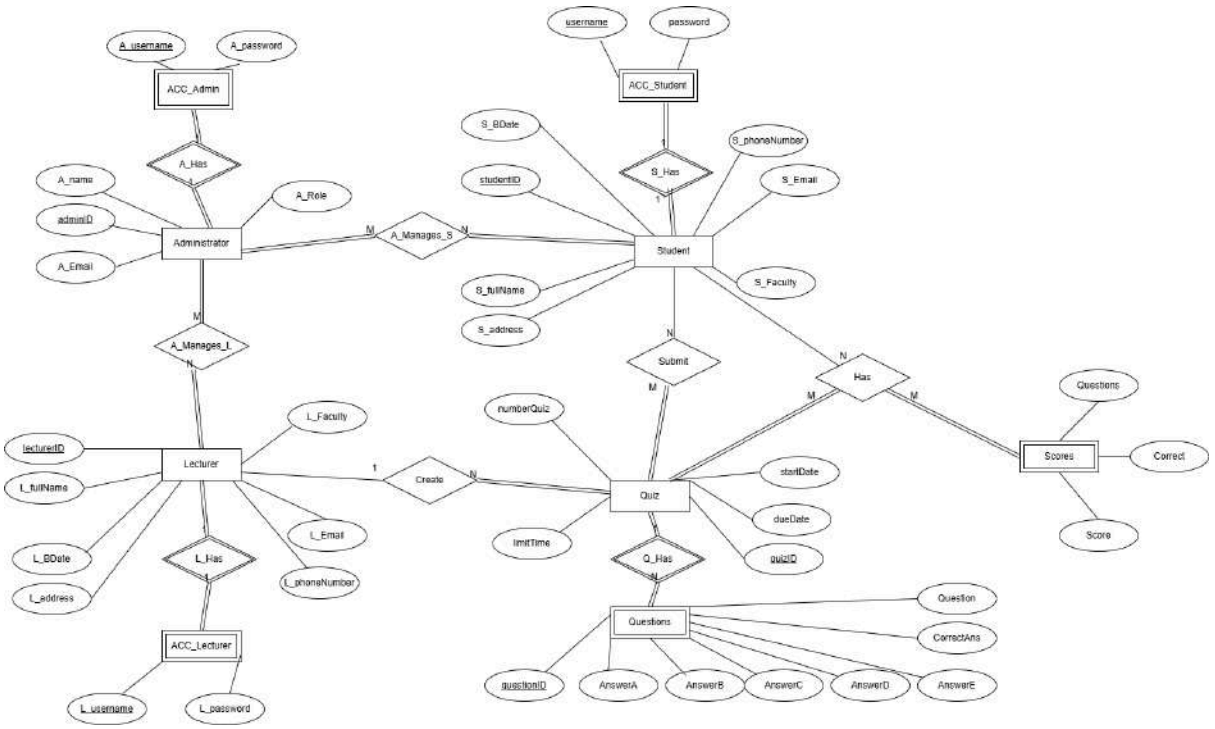
- Lecturers:
    - Add, edit, or delete quizzes, and their questions.
    - Assign specific quizzes to students or groups of students.
    - Access and review quiz responses submitted by students.
    - Analyze students' performance and scores.
- Admins:
    - Add, edit, or delete user accounts (Students, Lecturers, Admins).
    - Assign roles (Lecturer, Student, Admin) to user accounts.
    - View system logs and track user activities.
    - Oversee quizzes and scores across the system.
    - Update or delete quiz and user data if needed.

III. DATABASE DESIGN

- Strong entities include:
+ **Student**{studentID, fullName, Bdate, gender, address, phoneNumber, email, Scores, faculty}
+ **Lecturer**{citizenID, name, Bdate, gender, address, phoneNumber, email, faculty}
+ **Administrator**{adminID, name, email, role}
+ **Quiz**{quizID, Questions, limitTime, startDate, endDate}

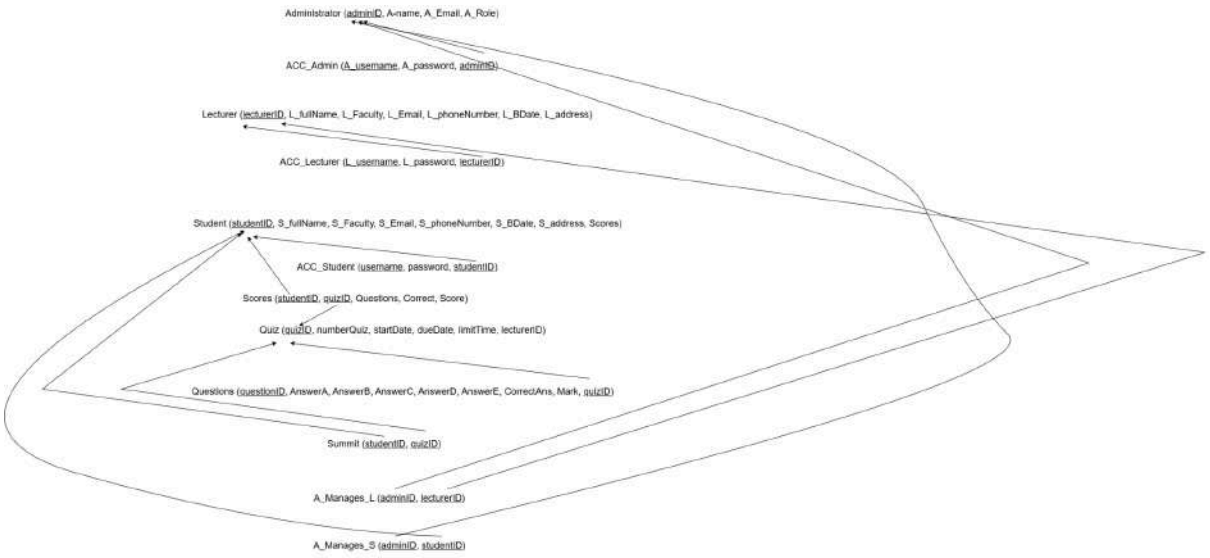
- Weak entities include:
+ **Question** {quizID(FK), questionID, Question, AnswerA, AnswerB, AnswerC, AnswerD, AnswerE, CorrectAnswer}
  (A Question is belonged to the Quiz entity, a Question can not exist without being linked to a specific Quiz)
+ **Scores** {studentID, quizID, Questions, Correct, Score}
  (A score is belonged to the Student and Quiz entity, it cannot exist without being linked to a student and a particular Quiz)
+ **ACC_Lecturer**{userName, password, lecturerID}
+ **ACC_Student**{userName, password, studentID}
+ **ACC_Admin**{userName, password, adminID}
  (A Lecturer/Student/Admin has his/her own account to access the system, an account can be exist without being related to a lecture, student or admin)

1. Draw an Entity-Relationship Diagram (ERD)



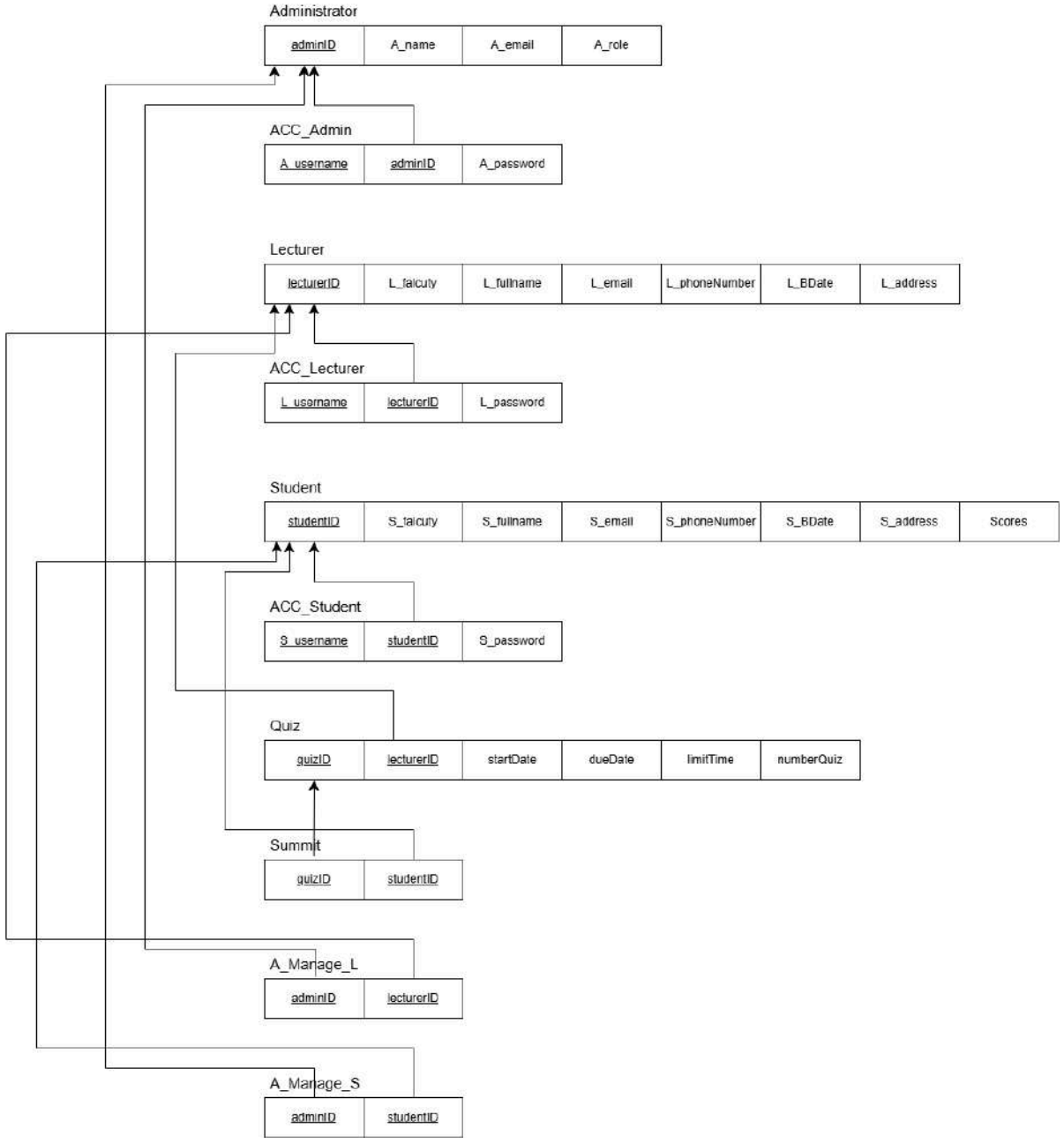
Click here for the detailed image.

2. Mapping ERD Diagram into Relational Schema

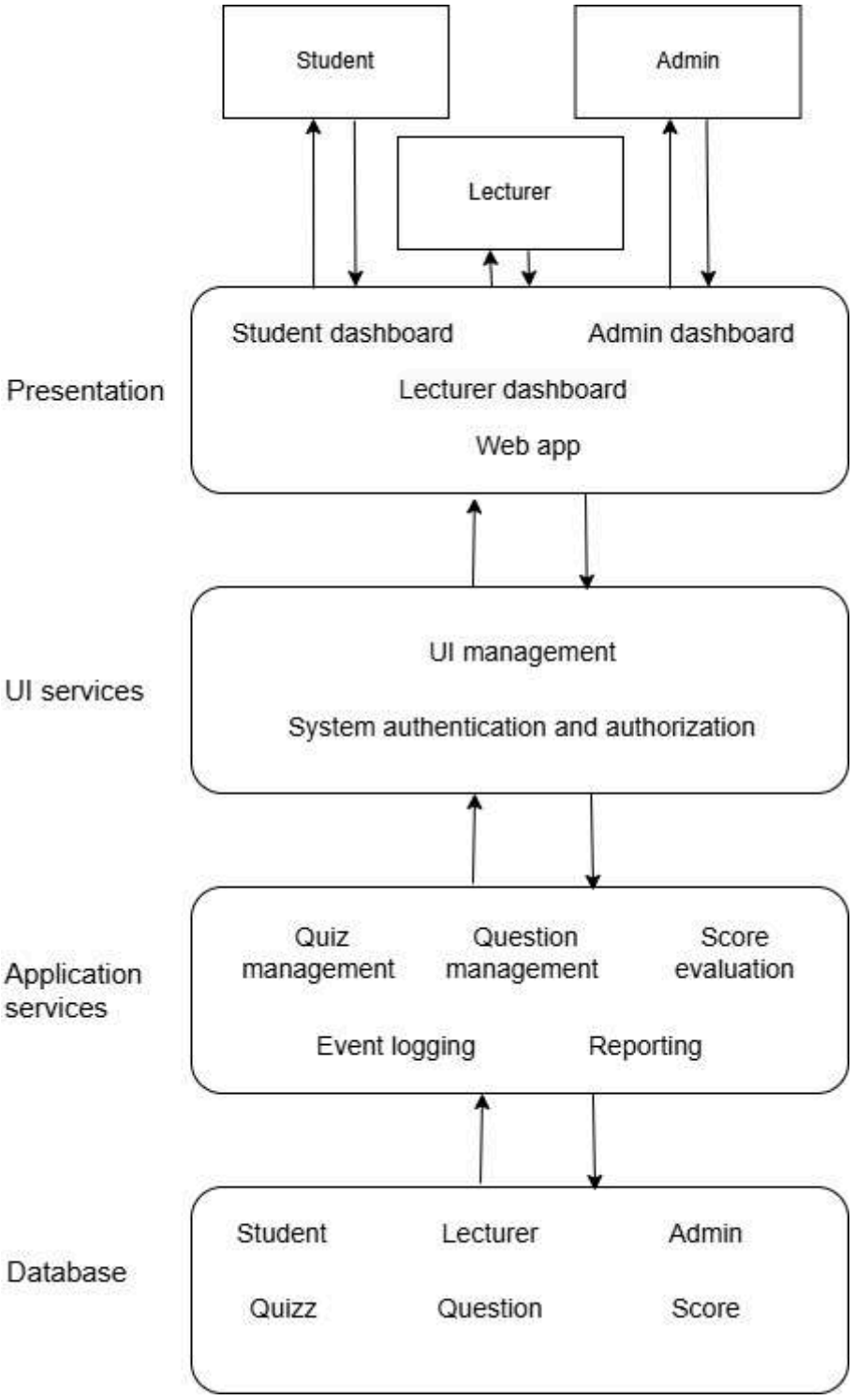


Click here for the detailed image.

3. Normalizing to Boyce-Codd Normal Form (BCNF) for the database schema



IV. ARCHITECTURE OF THE SYSTEM



V. DATABASE IMPLEMENTATION

1. Creating database tables

The following tables shall be created:

- Student
- Lecturer
- Administrator
- Quizzes
- Questions
- Scores
- Accounts
- ACC_Admin
- ACC_Lecturer
- ACC_Student

Creating the **Student** table:

```
CREATE TABLE Student (
    STUDENTID     VARCHAR(7) PRIMARY KEY,
    FULL_NAME     VARCHAR(30) NOT NULL,
    BDATE         DATE,
    GENDER        CHAR(1),
    ADDRESS       VARCHAR(60) NOT NULL,
    PHONE         VARCHAR(10),
    EMAIL         VARCHAR(30) NOT NULL,
    FACULTY       VARCHAR(10)
);
```

Creating the **Lecturer** table:

```
CREATE TABLE Lecturer (
    L_CITIZENID     VARCHAR(12) PRIMARY KEY,
    L_FULL_NAME     VARCHAR(30) NOT NULL,
    L_BDATE         DATE,
    L_GENDER        CHAR(1),
    L_ADDRESS       VARCHAR(60) NOT NULL,
    L_PHONE         VARCHAR(10),
    L_EMAIL         VARCHAR(30) NOT NULL,
    L_FACULTY       VARCHAR(10)
);
```

Creating the **Administrator** table:

```
CREATE TABLE Administrator (
    AD_ID       VARCHAR(10) PRIMARY KEY,
    AD_NAME     VARCHAR(30) NOT NULL,
    AD_EMAIL    VARCHAR(30) NOT NULL,
    AD_ROLE     VARCHAR(10)
);
```

Creating the **Quizzes** table:

```
CREATE TABLE Quizzes (
    QUIZ_ID       VARCHAR(6)  PRIMARY KEY,
    START_TIME    DATE,
    END_TIME      DATE,
    TIME_LIMIT    INT,
    QUESTIONS     INT
);
```

Creating the **Questions** table:

```
CREATE TABLE Questions (
    QUIZ_ID       VARCHAR(6),
    QUESTION_ID   VARCHAR(6) PRIMARY KEY,
    QUESTION      VARCHAR(1000),
    ANSWER_A      VARCHAR(1000),
    ANSWER_B      VARCHAR(1000),
    ANSWER_C      VARCHAR(1000),
    ANSWER_D      VARCHAR(1000),
    ANSWER_E      VARCHAR(1000),
    CORRECT       CHAR(1),

    CONSTRAINT fk_QuizID FOREIGN KEY (QUIZ_ID)
       REFERENCES Quizzes(QUIZ_ID)
       ON DELETE CASCADE
);
```

Creating the **Scores** table:

```
CREATE TABLE Scores (
    STUDENT_ID    VARCHAR(7),
    QUIZ_ID       VARCHAR(6),
    QUESTIONS     INT(3),
    CORRECT       INT(3),
    SCORE         FLOAT,

    CONSTRAINT fk_StuID_Scores FOREIGN KEY (STUDENT_ID)
       REFERENCES Student(STUDENTID)
       ON DELETE CASCADE,
    CONSTRAINT fk_QuizID_Scores FOREIGN KEY (QUIZ_ID)
       REFERENCES Quizzes(QUIZ_ID)
       ON DELETE CASCADE,
```

```
);
```

Creating the **Accounts** table:

```
CREATE TABLE Accounts (
    USERNAME    VARCHAR(20) PRIMARY KEY,
    PASSWORD    VARCHAR(20)
);
```

Creating the **ACC_Student** table:

```
CREATE TABLE ACC_Student (
    USERNAME    VARCHAR(20) PRIMARY KEY,
    STUDENTID   VARCHAR( 7),

    CONSTRAINT fk_AccSTU FOREIGN KEY (USERNAME)
       REFERENCES Accounts(USERNAME)
       ON DELETE CASCADE
);
```

Creating the **ACC_Lecturer** table:

```
CREATE TABLE ACC_Lecturer (
    USERNAME    VARCHAR(20) PRIMARY KEY,
    CITIZENID   VARCHAR( 12),

    CONSTRAINT fk_AccLEC FOREIGN KEY (USERNAME)
       REFERENCES Accounts(USERNAME)
       ON DELETE CASCADE
);
```

Creating the **ACC_Admin** table:

```
CREATE TABLE ACC_Student (
    USERNAME    VARCHAR(20) PRIMARY KEY,

    CONSTRAINT fk_AccADMIN FOREIGN KEY (USERNAME)
       REFERENCES Accounts(USERNAME)
       ON DELETE CASCADE
);
```

After creating the tables, the script output displays:

```
Table STUDENT created.
Table LECTURER created.
Table ADMINISTRATOR created.
Table QUIZZES created.
Table QUESTIONS created.
Table SCORES created.
Table ACCOUNTS created.
```

```
Table ACC_STUDENT created.
Table ACC_LECTURER created.
Table ACC_ADMIN created.
```

2. Adding demo values into the tables

```
--------------------STUDENT
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
INSERT INTO STUDENT VALUES('2252001','NGUYEN VAN A','01-01-2004','M','HO CHI
MINH CITY','0123456789','A.NGUYEN@HCMUT.EDU.VN','CSE');
INSERT INTO STUDENT VALUES('2252002','TRAN THI B','14-03-2004','F','HO CHI
MINH CITY','0123456790','B.TRAN@HCMUT.EDU.VN','SIM');
INSERT INTO STUDENT VALUES('2252003','DAO THANH C','11-01-2004','M','HO CHI
MINH CITY','0123456791','C.DAO@HCMUT.EDU.VN','GEOPET');
INSERT INTO STUDENT VALUES('2252004','LE THANH D','05-06-2004','M','HO CHI
MINH CITY','0123456792','D.LE@HCMUT.EDU.VN','DEE');
INSERT INTO STUDENT VALUES('2252005','NGUYEN VAN A','31-12-2004','M','HO CHI
MINH CITY','0123456793','A.NGUYEN@HCMUT.EDU.VN','CSE');
INSERT INTO STUDENT VALUES('2252006','NGUYEN VAN E','15-08-2004','M','HO CHI
MINH CITY','0123456794','E.NGUYEN@HCMUT.EDU.VN','CSE');
INSERT INTO STUDENT VALUES('2252007','NGUYEN VAN Y','09-01-2004','M','HO CHI
MINH CITY','0123456795','Y.NGUYEN@HCMUT.EDU.VN','CSE');
INSERT INTO STUDENT VALUES('2252082','NGUYEN THANH BINH','13-03-2004','M','HO
CHI MINH CITY','0123456796','BINH.NGUYENTHANH@HCMUT.EDU.VN','CSE');
INSERT INTO STUDENT VALUES('2252453','TRAN QUOC BAO
LONG','02-11-2004','M','HO CHI MINH
CITY','0904051758','LONG.TRAN041102@HCMUT.EDU.VN','CSE');
INSERT INTO STUDENT VALUES('2252808','DANG DUY TIEN','26-10-2004','M','HO CHI
MINH CITY','0123456798','TIEN.DANGDUYTIEN@HCMUT.EDU.VN','CSE');

--------------------LECTURER
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
INSERT INTO LECTURER VALUES('075080000001','DO THANH
THAI','06-06-1980','M','DONG NAI','0901234567','THAI@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075081000002','NGUYEN DUC
DUNG','12-03-1981','M','DONG NAI','0901234568','DUNGNN@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075082000003','LE THANH
VAN','09-10-1982','F','DONG NAI','0901234569','LTV@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075083000004','LE THANH
SACH','04-04-1983','M','DONG NAI','0901234570','SACHLE@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075084000005','NGUYEN VAN
BINH','01-09-1984','M','DONG
NAI','0901234571','BINHNGUYEN@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075085000006','TRAN THANH
BINH','13-12-1985','M','DONG NAI','0901234572','TTBINH@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075086000007','BUI DUC
BAO','08-09-1986','M','DONG NAI','0901234573','BDBAO@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075087000008','TRAN THI
HA','25-11-1987','F','DONG NAI','0901234574','HATRAN@HCMUT.EDU.VN','CSE');
INSERT INTO LECTURER VALUES('075088000009','CHAU VAN
LIEM','31-01-1988','M','DONG NAI','0901234575','CVL@HCMUT.EDU.VN','CSE');
```

```
INSERT INTO LECTURER VALUES('075089000010','PHAN TRONG
NHAN','05-12-1989','M','DONG
NAI','0901234576','PHANNHAN@HCMUT.EDU.VN','CSE');

--------------------ADMINISTRATOR
INSERT INTO ADMINISTRATOR
VALUES('0001','ADMIN_A','ABC@HCMUT.EDU.VN','ADMIN');
INSERT INTO ADMINISTRATOR VALUES('0101','ADMIN_B','ABB@HCMUT.EDU.VN','MOD');
INSERT INTO ADMINISTRATOR VALUES('0102','ADMIN_C','ABE@HCMUT.EDU.VN','MOD');
INSERT INTO ADMINISTRATOR VALUES('0103','ADMIN_D','ABF@HCMUT.EDU.VN','MOD');

----------------------QUIZZES
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
INSERT INTO QUIZZES VALUES('024001','01-11-2024','02-11-2024',30,10);
INSERT INTO QUIZZES VALUES('024002','03-11-2024','04-11-2024',30,10);
INSERT INTO QUIZZES VALUES('024003','05-11-2024','06-11-2024',30,10);

----------------------QUESTIONS
INSERT INTO QUESTIONS
VALUES('024001','000001','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000002','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000003','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000004','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000005','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000006','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000007','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000008','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000009','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024001','000010','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000011','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
```

```
INSERT INTO QUESTIONS
VALUES('024002','000012','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000013','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000014','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000015','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000016','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000017','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000018','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000019','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024002','000020','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000021','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000022','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000023','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000024','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000025','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000026','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000027','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000028','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
```

```
INSERT INTO QUESTIONS
VALUES('024003','000029','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');
INSERT INTO QUESTIONS
VALUES('024003','000030','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_
D','ANSWER_E','A');

----------------------SCORES
INSERT INTO SCORES VALUES('2252001','024001',10,8,8.0);
INSERT INTO SCORES VALUES('2252001','024002',10,9,9.0);
INSERT INTO SCORES VALUES('2252001','024003',10,8,8.0);
INSERT INTO SCORES VALUES('2252453','024001',10,9,9.0);
INSERT INTO SCORES VALUES('2252453','024002',10,10,10.0);
INSERT INTO SCORES VALUES('2252453','024003',10,8,8.0);
INSERT INTO SCORES VALUES('2252003','024001',10,7,7.0);
INSERT INTO SCORES VALUES('2252004','024001',10,10,10.0);

----------------------ACCOUNTS
INSERT INTO ACCOUNTS VALUES('STUDENT','ABCD1234');
INSERT INTO ACCOUNTS VALUES('LECTURER','ABCD1234');
INSERT INTO ACCOUNTS VALUES('ADMIN','ABCD1234');
```

3. Demonstrate desired data using SELECT

List all students:

```
--List all students
SELECT * FROM STUDENT;
```

	STUDENTID	FULL_NAME	BDATE	GENDER	ADDRESS	PHONE	EMAIL	FACULTY
1	2252001	NGUYEN VAN A	01-01-2004	M	HO CHI MINH CITY	0123456789	A.NGUYEN@HCMUT.EDU.VN	CSE
2	2252002	TRAN THI B	14-03-2004	F	HO CHI MINH CITY	0123456790	B.TRAN@HCMUT.EDU.VN	SIM
3	2252003	DAO THANH C	11-01-2004	M	HO CHI MINH CITY	0123456791	C.DAO@HCMUT.EDU.VN	GEOPET
4	2252004	LE THANH D	05-06-2004	M	HO CHI MINH CITY	0123456792	D.LE@HCMUT.EDU.VN	DEE
5	2252005	NGUYEN VAN A	31-12-2004	M	HO CHI MINH CITY	0123456793	A.NGUYEN@HCMUT.EDU.VN	CSE
6	2252006	NGUYEN VAN E	15-08-2004	M	HO CHI MINH CITY	0123456794	E.NGUYEN@HCMUT.EDU.VN	CSE
7	2252007	NGUYEN VAN Y	09-01-2004	M	HO CHI MINH CITY	0123456795	Y.NGUYEN@HCMUT.EDU.VN	CSE
8	2252082	NGUYEN THANH BINH	13-03-2004	M	HO CHI MINH CITY	0123456796	BINH.NGUYENTHANH@HCMUT.EDU.VN	CSE
9	2252453	TRAN QUOC BAO LONG	02-11-2004	M	HO CHI MINH CITY	0904051758	LONG.TRAN041102@HCMUT.EDU.VN	CSE
10	2252808	DANG DUY TIEN	26-10-2004	M	HO CHI MINH CITY	0123456798	TIEN.DANGDUYTIEN@HCMUT.EDU.VN	CSE

List all lecturers in a university:

```
--List all lecturers
SELECT * FROM LECTURER;
```

	L_CITIZENID	L_FULL_NAME	L_BDATE	L_GENDER	L_ADDRESS	L_PHONE	L_EMAIL	L_FACULTY
1	075080000001	DO THANH THAI	06-06-1980	M	DONG NAI	0901234567	THAI@HCMUT.EDU.VN	CSE
2	075081000002	NGUYEN DUC DUNG	12-03-1981	M	DONG NAI	0901234568	DUNGNN@HCMUT.EDU.VN	CSE
3	075082000003	LE THANH VAN	09-10-1982	F	DONG NAI	0901234569	LTV@HCMUT.EDU.VN	CSE
4	075083000004	LE THANH SACH	04-04-1983	M	DONG NAI	0901234570	SACHLE@HCMUT.EDU.VN	CSE
5	075084000005	NGUYEN VAN BINH	01-09-1984	M	DONG NAI	0901234571	BINHNGUYEN@HCMUT.EDU.VN	CSE
6	075085000006	TRAN THANH BINH	13-12-1985	M	DONG NAI	0901234572	TTBINH@HCMUT.EDU.VN	CSE
7	075086000007	BUI DUC BAO	08-09-1986	M	DONG NAI	0901234573	BDBAO@HCMUT.EDU.VN	CSE
8	075087000008	TRAN THI HA	25-11-1987	F	DONG NAI	0901234574	HATRAN@HCMUT.EDU.VN	CSE
9	075088000009	CHAU VAN LIEM	31-01-1988	M	DONG NAI	0901234575	CVL@HCMUT.EDU.VN	CSE
10	075089000010	PHAN TRONG NHAN	05-12-1989	M	DONG NAI	0901234576	PHANNHAN@HCMUT.EDU.VN	CSE

List all questions from a quiz:

```
--List all questions from Quiz ID 024001
SELECT * FROM QUESTIONS
WHERE QUIZ_ID = '024001';
```

	QUIZ_ID	QUESTION_ID	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	CORRECT
1	024001	000001	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
2	024001	000002	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
3	024001	000003	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
4	024001	000004	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
5	024001	000005	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
6	024001	000006	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
7	024001	000007	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
8	024001	000008	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
9	024001	000009	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
10	024001	000010	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A

List scores of students:

```
--List all scores from students
SELECT * FROM SCORES
```

	STUDENT...	QUIZ_ID	QUESTIONS	CORRECT	SCORE
1	2252001	024001	10	8	8
2	2252001	024002	10	9	9
3	2252001	024003	10	8	8
4	2252453	024001	10	9	9
5	2252453	024002	10	10	10
6	2252453	024003	10	8	8
7	2252003	024001	10	7	7
8	2252004	024001	10	10	10

List all students within a faculty, for example, all students in Faculty of Computer Science and Engineering (CSE):

```
--List all CSE students
SELECT STUDENTID, FULL_NAME, EMAIL, FACULTY FROM STUDENT
WHERE FACULTY = 'CSE';
```

STUDENT	FULL_NAME	EMAIL	FACULTY
2252001	NGUYEN VAN A	A.NGUYEN@HCMUT.EDU.VN	CSE
2252005	NGUYEN VAN A	A.NGUYEN@HCMUT.EDU.VN	CSE
2252006	NGUYEN VAN E	E.NGUYEN@HCMUT.EDU.VN	CSE
2252007	NGUYEN VAN Y	Y.NGUYEN@HCMUT.EDU.VN	CSE
2252082	NGUYEN THANH BINH	BINH.NGUYENTHANH@HCMUT.EDU.VN	CSE
2252453	TRAN QUOC BAO LONG	LONG.TRAN041102@HCMUT.EDU.VN	CSE
2252808	DANG DUY TIEN	TIEN.DANGDUYTIEN@HCMUT.EDU.VN	CSE

4. Write Procedures, Functions, Assertions and Triggers

4.1. Create Views

Creating Views containing information from the tables Lecturer and Student.

```
-- Create Views
CREATE OR REPLACE VIEW LECTURER_VIEW AS
SELECT L_CITIZENID, L_FULL_NAME, L_BDATE, L_GENDER, L_ADDRESS, L_PHONE,
L_EMAIL, L_FACULTY
FROM LECTURER;

CREATE OR REPLACE VIEW STUDENT_VIEW AS
SELECT STUDENTID, FULL_NAME, BDATE, GENDER, ADDRESS
FROM STUDENT;
```

4.2. Create Procedures and Triggers to add students

- Add a procedure for adding students.

This procedure receives information of a new student, then inserts into the Students table a new value containing information of the newly added student.

```
-- Procedure to add students
CREATE OR REPLACE PROCEDURE INSERT_STUDENT (
      STUDENTID IN STUDENT_VIEW.STUDENTID%TYPE,
      FULL_NAME IN STUDENT_VIEW.FULL_NAME%TYPE,
      BDATE     IN STUDENT_VIEW.BDATE%TYPE,
      GENDER    IN STUDENT_VIEW.GENDER%TYPE,
      ADDRESS   IN STUDENT_VIEW.ADDRESS%TYPE,
      PHONE     IN STUDENT_VIEW.PHONE%TYPE,
      EMAIL     IN STUDENT_VIEW.EMAIL%TYPE,
      FACULTY   IN STUDENT_VIEW.FACULTY%TYPE,
)
AS
BEGIN
      INSERT INTO STUDENT_VIEW
```

```
      VALUES (STUDENTID, FULL_NAME, BDATE, GENDER, ADDRESS, PHONE, EMAIL,
FACULTY);
END INSERT_STUDENT;
```

Add a Trigger to implement semantic constraints for the added data.

```
-- Trigger to check constraints
CREATE OR REPLACE TRIGGER CHECK_INSERT_STUDENT
BEFORE INSERT ON STUDENT
FOR EACH ROW
BEGIN
      IF (regexp_like(:NEW.STUDENTID,'[[:alpha:] *!?@#$&+()/]')) THEN
      raise_application_error(-20000,'STUDENT ID CAN ONLY CONTAIN NUMBERS');
      END IF;

      IF (regexp_like(:NEW.FULL_NAME, '[0123456789*!?@#$&+()/]') THEN
      raise_application_error(-20000,'FULL NAME CANNOT CONTAIN NUMBERS OR
SPECIAL CHARACTERS');
      END IF;

      IF(:NEW.GENDER != 'M' AND :NEW.GENDER != 'F') THEN
      raise_application_error(-20000,'GENDER MUST BE M OR F');
      END IF;

      IF (regexp_like(:NEW.PHONE,'[[:alpha:] *!?@#$&+()/]')) THEN
      raise_application_error(-20000,'INVALID PHONE NUMBER');
      END IF;
END;
```

Testing the procedure:

- Adding valid information:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
EXECUTE INSERT_STUDENT('2252286','LE MINH QUOC','17-AUG-04','M','HO CHI MINH
CITY','0901010101','QUOC.LEMINH@HCMUT.EDU.VN','FME');
EXECUTE INSERT_STUDENT('2311373','TRAN PHONG HAO','25-MAR-05','M','BINH
DUONG','0341125538','HAO.TRAN2503@HCMUT.EDU.VN','FCE');
```

After that, new values are added to the table:

	STUDENTID	FULL_NAME	BDATE	GENDER	ADDRESS	PHONE	EMAIL	FACULTY
1	2252001	NGUYEN VAN A	01-JAN-04	M	HO CHI MINH CITY	0123456789	A.NGUYEN@HCMUT.EDU.VN	CSE
2	2252002	TRAN THI B	14-MAR-04	F	HO CHI MINH CITY	0123456790	B.TRAN@HCMUT.EDU.VN	SIM
3	2252003	DAO THANH C	11-JAN-04	M	HO CHI MINH CITY	0123456791	C.DAO@HCMUT.EDU.VN	GEOPET
4	2252004	LE THANH D	05-JUN-04	M	HO CHI MINH CITY	0123456792	D.LE@HCMUT.EDU.VN	DEE
5	2252005	NGUYEN VAN A	31-DEC-04	M	HO CHI MINH CITY	0123456793	A.NGUYEN@HCMUT.EDU.VN	CSE
6	2252006	NGUYEN VAN E	15-AUG-04	M	HO CHI MINH CITY	0123456794	E.NGUYEN@HCMUT.EDU.VN	CSE
7	2252007	NGUYEN VAN Y	09-JAN-04	M	HO CHI MINH CITY	0123456795	Y.NGUYEN@HCMUT.EDU.VN	CSE
8	2252082	NGUYEN THANH BINH	13-MAR-04	M	HO CHI MINH CITY	0123456796	BINH.NGUYENTHANH@HCMUT.EDU.VN	CSE
9	2252453	TRAN QUOC BAO LONG	02-NOV-04	M	HO CHI MINH CITY	0904051758	LONG.TRAN041102@HCMUT.EDU.VN	CSE
10	2252808	DANG DUY TIEN	26-OCT-04	M	HO CHI MINH CITY	0123456798	TIEN.DANGDUYTIEN@HCMUT.EDU.VN	CSE
11	2252286	LE MINH QUOC	17-AUG-04	M	HO CHI MINH CITY	0901010101	QUOC.LEMINH@HCMUT.EDU.VN	FME
12	2311373	TRAN PHONG HAO	25-MAR-05	M	BINH DUONG	0341125538	HAO.TRAN2503@HCMUT.EDU.VN	FCE

- Adding invalid information: Student ID already existed in the table.

```
EXECUTE INSERT_STUDENT('2252001','NGUYEN VAN H','16-DEC-2004','M','HO CHI
MINH CITY','0123400229','H.NGUYEN@HCMUT.EDU.VN','CSE');
```

Result: Unique constraint (primary key) violated, no new values are added to the table.

```
BEGIN INSERT_STUDENT('2252001','NGUYEN VAN H','16-DEC-2004','M','HO CHI MINH CITY','0123400229','H.NGUYEN@HCMUT.EDU.VN','CSE'); END;
*
ERROR at line 1:
ORA-00001: unique constraint (SYSTEM.SYS_C008317) violated
ORA-06512: at "SYSTEM.INSERT_STUDENT", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-00001/


More Details :
https://docs.oracle.com/error-help/db/ora-00001/
https://docs.oracle.com/error-help/db/ora-06512/
```

- Adding invalid information: Student's ID containing characters other than numbers.

```
EXECUTE INSERT_STUDENT('2311abc','LE DUY MANH','20-APR-05','M','HO CHI MINH
CITY','0921345366','MANH.LE1@HCMUT.EDU.VN','FAS');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_STUDENT('2311abc','LE DUY MANH','20-APR-05','M','HO CHI MINH CITY','0921345366','MANH.LE1@HCMUT.EDU.VN','FAS'); END;
*
ERROR at line 1:
ORA-20000: STUDENT ID CAN ONLY CONTAIN NUMBERS
ORA-06512: at "SYSTEM.CHECK_INSERT_STUDENT", line 3
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_STUDENT'
ORA-06512: at "SYSTEM.INSERT_STUDENT", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/


More Details :
https://docs.oracle.com/error-help/db/ora-20000/
```

- Adding invalid information: Student's full name containing numbers.

```
EXECUTE INSERT_STUDENT('2311388','LE DUY MANH1','20-APR-05','M','HO CHI MINH
CITY','0921345366','MANH.LE1@HCMUT.EDU.VN','FAS');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_STUDENT('2311388','LE DUY MANH1','20-APR-05','M','HO CHI MINH CITY','0921345366','MANH.LE1@HCMUT.EDU.VN','FAS'); END;
*
ERROR at line 1:
ORA-20000: FULL NAME CANNOT CONTAIN NUMBERS OR SPECIAL CHARACTERS
ORA-06512: at "SYSTEM.CHECK_INSERT_STUDENT", line 7
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_STUDENT'
ORA-06512: at "SYSTEM.INSERT_STUDENT", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

- Adding invalid information: Student's gender is neither male (M) nor female (F).

```
EXECUTE INSERT_STUDENT('2311388','LE DUY MANH','20-APR-05','N','HO CHI MINH
CITY','0921345366','MANH.LE1@HCMUT.EDU.VN','FAS');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_STUDENT('2311388','LE DUY MANH','20-APR-05','N','HO CHI MINH CITY','0921345366','MANH.LE1@HCMUT.EDU.VN','FAS'); END;
*
ERROR at line 1:
ORA-20000: GENDER MUST BE M OR F
ORA-06512: at "SYSTEM.CHECK_INSERT_STUDENT", line 11
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_STUDENT'
ORA-06512: at "SYSTEM.INSERT_STUDENT", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/


More Details :
https://docs.oracle.com/error-help/db/ora-20000/
```

- Adding invalid information: Invalid phone number.

```
EXECUTE INSERT_STUDENT('2311388','LE DUY MANH','20-APR-05','M','HO CHI MINH
CITY','09213a5366','MANH.LE1@HCMUT.EDU.VN','FAS');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_STUDENT('2311388','LE DUY MANH','20-APR-05','M','HO CHI MINH CITY','09213a5366','MANH.LE1@HCMUT.EDU.VN','FAS'); END;
*
ERROR at line 1:
ORA-20000: INVALID PHONE NUMBER
ORA-06512: at "SYSTEM.CHECK_INSERT_STUDENT", line 15
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_STUDENT'
ORA-06512: at "SYSTEM.INSERT_STUDENT", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

4.3. Create Procedures and Triggers to add lecturers

- Add a procedure for adding lecturers.

This procedure receives information of a new lecturer, then inserts into the Lecturer table a new value containing information of the newly added lecturer.

```
--Trigger to add lecturers
CREATE OR REPLACE PROCEDURE INSERT_LECTURER (
      L_CITIZENID IN LECTURER_VIEW.L_CITIZENID%TYPE,
      L_FULL_NAME IN LECTURER_VIEW.L_FULL_NAME%TYPE,
      L_BDATE     IN LECTURER_VIEW.L_BDATE%TYPE,
      L_GENDER    IN LECTURER_VIEW.L_GENDER%TYPE,
      L_ADDRESS   IN LECTURER_VIEW.L_ADDRESS%TYPE,
      L_PHONE     IN LECTURER_VIEW.L_PHONE%TYPE,
      L_EMAIL     IN LECTURER_VIEW.L_EMAIL%TYPE,
      L_FACULTY   IN LECTURER_VIEW.L_FACULTY%TYPE
)
AS
BEGIN
      INSERT INTO LECTURER_VIEW
      VALUES (L_CITIZENID, L_FULL_NAME, L_BDATE, L_GENDER, L_ADDRESS,
L_PHONE, L_EMAIL, L_FACULTY);
END INSERT_LECTURER;
```

Add a Trigger to implement semantic constraints for the added data.

```
-- Trigger to check constraints
CREATE OR REPLACE TRIGGER CHECK_INSERT_LECTURER
BEFORE INSERT ON LECTURER
FOR EACH ROW
BEGIN
    IF (regexp_like(:NEW.L_CITIZENID,'[[:alpha:] *!?@#$&+()/]')) THEN
    raise_application_error(-20000,'CITIZEN ID CAN ONLY CONTAIN NUMBERS');
    END IF;

    IF (regexp_like(:NEW.L_FULL_NAME, '[0123456789*!?@#$&+()/]')) THEN
    raise_application_error(-20000,'FULL NAME CANNOT CONTAIN NUMBERS OR
SPECIAL CHARACTERS');
    END IF;

    IF(:NEW.L_GENDER != 'M' AND :NEW.L_GENDER !='F') THEN
    raise_application_error(-20000,'GENDER MUST BE M OR F');
    END IF;

    IF (regexp_like(:NEW.L_PHONE,'[[:alpha:] *!?@#$&+()/]')) THEN
    raise_application_error(-20000,'INVALID PHONE NUMBER');
    END IF;
END;
```

Testing the procedure:

- Adding valid information into the table:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
EXECUTE INSERT_LECTURER('079088000001','LE ANH SON','12-04-1988','M','BINH
THUAN','0921357913','SONLE@HCMUT.EDU.VN','DEE');
```

After that, new values are added to the table:

	L_CITIZENID	L_FULL_NAME	L_BDATE	L_GENDER	L_ADDRESS	L_PHONE	L_EMAIL	L_FACULTY
1	075080000001	DO THANH THAI	06-06-1980	M	DONG NAI	0901234567	THAI@HCMUT.EDU.VN	CSE
2	075081000002	NGUYEN DUC DUNG	12-03-1981	M	DONG NAI	0901234568	DUNGNN@HCMUT.EDU.VN	CSE
3	075082000003	LE THANH VAN	09-10-1982	F	DONG NAI	0901234569	LTV@HCMUT.EDU.VN	CSE
4	075083000004	LE THANH SACH	04-04-1983	M	DONG NAI	0901234570	SACHLE@HCMUT.EDU.VN	CSE
5	075084000005	NGUYEN VAN BINH	01-09-1984	M	DONG NAI	0901234571	BINHNGUYEN@HCMUT.EDU.VN	CSE
6	075085000006	TRAN THANH BINH	13-12-1985	M	DONG NAI	0901234572	TTBINH@HCMUT.EDU.VN	CSE
7	075086000007	BUI DUC BAO	08-09-1986	M	DONG NAI	0901234573	BDBAO@HCMUT.EDU.VN	CSE
8	075087000008	TRAN THI HA	25-11-1987	F	DONG NAI	0901234574	HATRAN@HCMUT.EDU.VN	CSE
9	075088000009	CHAU VAN LIEM	31-01-1988	M	DONG NAI	0901234575	CVL@HCMUT.EDU.VN	CSE
10	075089000010	PHAN TRONG NHAN	05-12-1989	M	DONG NAI	0901234576	PHANNHAN@HCMUT.EDU.VN	CSE
11	079088000001	LE ANH SON	12-04-1988	M	BINH THUAN	0921357913	SONLE@HCMUT.EDU.VN	DEE

- Adding invalid information: Citizen ID already available:

```
EXECUTE INSERT_LECTURER('075088000009','LE ANH SON','12-04-1988','M','BINH
THUAN','0921357913','SONLE@HCMUT.EDU.VN','DEE');
```

Result: Unique constraint violated, no new values are added to the table.

```
BEGIN INSERT_LECTURER('075088000009','LE ANH SON','12-04-1988','M','BINH THUAN','0921357913','SONLE@HCMUT.EDU.VN','DEE'); END;
*
ERROR at line 1:
ORA-00001: unique constraint (SYSTEM.SYS_C008321) violated
ORA-06512: at "SYSTEM.INSERT_LECTURER", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-00001/


More Details :
https://docs.oracle.com/error-help/db/ora-00001/
https://docs.oracle.com/error-help/db/ora-06512/
```

- Adding invalid information: Invalid Citizen ID

```
EXECUTE INSERT_LECTURER('07508a000009','LE ANH SON','12-04-1988','M','BINH
THUAN','0921357913','SONLE@HCMUT.EDU.VN','DEE');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_LECTURER('07508a000009','LE ANH SON','12-04-1988','M','BINH THUAN','0921357913','SONLE@HCMUT.EDU.VN','DEE'); END;
*
ERROR at line 1:
ORA-20000: CITIZEN ID CAN ONLY CONTAIN NUMBERS
ORA-06512: at "SYSTEM.CHECK_INSERT_LECTURER", line 3
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_LECTURER'
ORA-06512: at "SYSTEM.INSERT_LECTURER", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

- Adding invalid information: Full name containing numbers or special characters

```
EXECUTE INSERT_LECTURER('079190001234','NGUYEN TH1 HOANG
MAI','19-06-1990','F','HO CHI MINH
CITY','0123455432','MAINGUYEN@HCMUT.EDU.VN','SIM');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_LECTURER('079190001234','NGUYEN TH1 HOANG MAI','19-06-1990','F','HO CHI MINH CITY','0123455432','MAINGUYEN@HCMUT.EDU.VN','SIM'); END;
*
ERROR at line 1:
ORA-20000: FULL NAME CANNOT CONTAIN NUMBERS OR SPECIAL CHARACTERS
ORA-06512: at "SYSTEM.CHECK_INSERT_LECTURER", line 7
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_LECTURER'
ORA-06512: at "SYSTEM.INSERT_LECTURER", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

- Adding invalid information: Gender is neither Male (M) or Female (F)

```
EXECUTE INSERT_LECTURER('079190001234','NGUYEN TH1 HOANG
MAI','19-06-1990','E','HO CHI MINH
CITY','0123455432','MAINGUYEN@HCMUT.EDU.VN','SIM');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_LECTURER('079190001234','NGUYEN THI HOANG MAI','19-06-1990','E','HO CHI MINH CITY','0123455432','MAINGUYEN@HCMUT.EDU.VN','SIM'); END;
*
ERROR at line 1:
ORA-20000: GENDER MUST BE M OR F
ORA-06512: at "SYSTEM.CHECK_INSERT_LECTURER", line 11
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_LECTURER'
ORA-06512: at "SYSTEM.INSERT_LECTURER", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

- Adding invalid information: Invalid phone number

```
EXECUTE INSERT_LECTURER('079190001234','NGUYEN THI HOANG
MAI','19-06-1990','F','HO CHI MINH
CITY','01XX455432','MAINGUYEN@HCMUT.EDU.VN','SIM');
```

Result: Semantic constraint violated, no new values are added to the table.

```
BEGIN INSERT_LECTURER('079190001234','NGUYEN THI HOANG MAI','19-06-1990','F','HO CHI MINH CITY','01XX455432','MAINGUYEN@HCMUT.EDU.VN','SIM'); END;
*
ERROR at line 1:
ORA-20000: INVALID PHONE NUMBER
ORA-06512: at "SYSTEM.CHECK_INSERT_LECTURER", line 15
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_LECTURER'
ORA-06512: at "SYSTEM.INSERT_LECTURER", line 13
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

4.4. Create Triggers when updating students' or lecturers' information

Create two Triggers to check semantic constraints when updating information in tables Student and Lecturer:

```
-- Creating triggers to update students' and lecturers' information
CREATE OR REPLACE TRIGGER CHECK_UPDATE_STUDENT
BEFORE UPDATE ON STUDENT
FOR EACH ROW
BEGIN
      IF (regexp_like(:NEW.STUDENTID,'[[:alpha:] *!?@#$&+()/]')) THEN
      raise_application_error(-20000,'STUDENT ID CAN ONLY CONTAIN NUMBERS');
      END IF;

      IF (regexp_like(:NEW.FULL_NAME, '[0123456789*!?@#$&+()/]')) THEN
      raise_application_error(-20000,'FULL NAME CANNOT CONTAIN NUMBERS OR
SPECIAL CHARACTERS');
      END IF;

      IF(:NEW.GENDER != 'M' AND :NEW.GENDER != 'F') THEN
      raise_application_error(-20000,'GENDER MUST BE M OR F');
      END IF;

      IF (regexp_like(:NEW.PHONE,'[[:alpha:] *!?@#$&+()/]')) THEN
      raise_application_error(-20000,'INVALID PHONE NUMBER');
      END IF;
END;

CREATE OR REPLACE TRIGGER CHECK_UPDATE_LECTURER
BEFORE UPDATE ON LECTURER
FOR EACH ROW
BEGIN
    IF (regexp_like(:NEW.L_CITIZENID,'[[:alpha:] *!?@#$&+()/]')) THEN
    raise_application_error(-20000,'CITIZEN ID CAN ONLY CONTAIN NUMBERS');
    END IF;

    IF (regexp_like(:NEW.L_FULL_NAME, '[0123456789*!?@#$&+()/]')) THEN
    raise_application_error(-20000,'FULL NAME CANNOT CONTAIN NUMBERS OR
SPECIAL CHARACTERS');
    END IF;

    IF(:NEW.L_GENDER != 'M' AND :NEW.L_GENDER !='F') THEN
    raise_application_error(-20000,'GENDER MUST BE M OR F');
    END IF;

    IF (regexp_like(:NEW.L_PHONE,'[[:alpha:] *!?@#$&+()/]')) THEN
    raise_application_error(-20000,'INVALID PHONE NUMBER');
    END IF;
END;
```

- Testing the triggers:

Updating student's information failed: Student ID already existed in the table.

```
UPDATE STUDENT SET STUDENTID = '2252007' WHERE STUDENTID = '2252453';
```

Result: Unique constraint violated, data remains unchanged.

```
Error starting at line : 2 in command -
UPDATE STUDENT SET STUDENTID = '2252007' WHERE STUDENTID = '2252453'
Error report -
ORA-00001: unique constraint (SYSTEM.SYS_C008317) violated

https://docs.oracle.com/error-help/db/ora-00001/

More Details :
https://docs.oracle.com/error-help/db/ora-00001/
```

Updating student's information failed: Invalid student ID.

```
UPDATE STUDENT SET STUDENTID = '2252x10' WHERE STUDENTID = '2252453';
```

Result: Semantic constraint violated, data remains unchanged.

```
Error starting at line : 2 in command -
UPDATE STUDENT SET STUDENTID = '2252x10' WHERE STUDENTID = '2252001'
Error at Command Line : 2 Column : 8
Error report -
SQL Error: ORA-20000: STUDENT ID CAN ONLY CONTAIN NUMBERS
ORA-06512: at "SYSTEM.CHECK_UPDATE_STUDENT", line 3
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_UPDATE_STUDENT'
```

Updating student's information failed: Student's full name containing numbers or special characters.

```
UPDATE STUDENT SET FULL_NAME = 'NGUYEN VAN 2' WHERE STUDENTID = '2252001';
```

Result: Semantic constraint violated, data remains unchanged.

```
Error starting at line : 2 in command -
UPDATE STUDENT SET FULL_NAME = 'NGUYEN VAN 2' WHERE STUDENTID = '2252001'
Error at Command Line : 2 Column : 8
Error report -
SQL Error: ORA-20000: FULL NAME CANNOT CONTAIN NUMBERS OR SPECIAL CHARACTERS
ORA-06512: at "SYSTEM.CHECK_UPDATE_STUDENT", line 7
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_UPDATE_STUDENT'
```

Updating student's information failed: Student's new gender is not either Male (M) or Female (F).

```
UPDATE STUDENT SET GENDER = 'E' WHERE STUDENTID = '2252001';
```

Result: Semantic constraint violated, data remains unchanged.

```
Error starting at line : 2 in command -
UPDATE STUDENT SET GENDER = 'E' WHERE STUDENTID = '2252001'
Error at Command Line : 2 Column : 8
Error report -
SQL Error: ORA-20000: GENDER MUST BE M OR F
ORA-06512: at "SYSTEM.CHECK_UPDATE_STUDENT", line 11
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_UPDATE_STUDENT'
```

Updating student's information failed: Invalid phone number.

```
UPDATE STUDENT SET PHONE = '01234567xx' WHERE STUDENTID = '2252001';
```

Result: Semantic constraint violated, data remains unchanged.

```
Error starting at line : 2 in command -
UPDATE STUDENT SET PHONE = '01234567xx' WHERE STUDENTID = '2252001'
Error at Command Line : 2 Column : 8
Error report -
SQL Error: ORA-20000: INVALID PHONE NUMBER
ORA-06512: at "SYSTEM.CHECK_UPDATE_STUDENT", line 15
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_UPDATE_STUDENT'
```

We can also use similar methods like above to test the trigger when updating information in the Lecturer table.

4.5. Create Procedures and Triggers to add students' scores

- Add a procedure to add scores

The procedure ADD_SCORES receives information about a student's result after finishing a quiz, then saves it into the Scores table. The score will be automatically calculated by dividing the number of correct answers by the number of questions, then converted to a scale of 10.

```
CREATE OR REPLACE PROCEDURE ADD_SCORES (
    nSTUDENT_ID IN SCORES.STUDENT_ID%TYPE,
    nQUIZ_ID    IN SCORES.QUIZ_ID%TYPE,
```

```
    nQUESTIONS  IN SCORES.QUESTIONS%TYPE,
    nCORRECT    IN SCORES.CORRECT%TYPE
)
AS
BEGIN
    INSERT INTO SCORES (STUDENT_ID, QUIZ_ID, QUESTIONS, CORRECT, SCORE)
    VALUES (nSTUDENT_ID, nQUIZ_ID, nQUESTIONS, nCORRECT, ROUND(nCORRECT /
nQUESTIONS * 10, 2));
END ADD_SCORES;
```

- Create a trigger to check the new information added or updated. The number of correct answers must be positive and no greater than the number of questions in a quiz.

```
CREATE OR REPLACE TRIGGER CHECK_SCORES
BEFORE INSERT ON SCORES
FOR EACH ROW
BEGIN
    IF (:NEW.CORRECT < 0 OR :NEW.CORRECT > :NEW.QUESTIONS) THEN
        raise_application_error(-20000,'CORRECT ANSWERS MUST BE BETWEEN 0 AND
NUMBER OF QUESTIONS');
    END IF;
END;

CREATE OR REPLACE TRIGGER CHECK_SCORES_UPDATE
BEFORE UPDATE ON SCORES
FOR EACH ROW
BEGIN
    IF (:NEW.CORRECT < 0 OR :NEW.CORRECT > :NEW.QUESTIONS) THEN
        raise_application_error(-20000,'CORRECT ANSWERS MUST BE BETWEEN 0 AND
NUMBER OF QUESTIONS');
    END IF;
END;
```

Testing the procedures:

- Adding valid information:

```
-- Adding score of a student
EXECUTE ADD_SCORES('2252002','024001',10,9);
-- Updating score of a student
UPDATE SCORES
SET CORRECT = 9,
    SCORE = (9 / QUESTIONS) * 10
WHERE STUDENT_ID = '2252003' AND QUIZ_ID = '024001';
```

When valid information is added or updated, new values are added to the table.

	STUDENT...	QUIZ_ID	QUESTIONS	CORRECT	SCORE
1	2252001	024001	10	9	9
2	2252001	024002	10	9	9
3	2252001	024003	10	8	8
4	2252453	024001	10	9	9
5	2252453	024002	10	10	10
6	2252453	024003	10	8	8
7	2252003	024001	10	7	7
8	2252004	024001	10	10	10
9	2252002	024001	10	9	9

Because Student_ID and Quiz_ID serve as foreign keys in table Scores, referring to the said attributes in tables Student and Quizzes respectively, newly added or updated information can only be done if the student's ID number and the quiz's ID number already existed in their respective tables.

- Adding invalid information: Foreign key not found

```
EXECUTE ADD_SCORES('2252103','024001',10,8);
EXECUTE ADD_SCORES('2252002','024004',10,9);
```

Result: Foreign key not found, no new values are added to the table.

```
BEGIN ADD_SCORES('2252103','024001',10,8); END;
*
ERROR at line 1:
ORA-02291: integrity constraint (SYSTEM.FK_STUID_SCORES) violated - parent key not found
ORA-06512: at "SYSTEM.ADD_SCORES", line 9
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-02291/


More Details :
https://docs.oracle.com/error-help/db/ora-02291/
https://docs.oracle.com/error-help/db/ora-06512/
BEGIN ADD_SCORES('2252002','024004',10,9); END;
*
ERROR at line 1:
ORA-02291: integrity constraint (SYSTEM.FK_QUIZID_SCORES) violated - parent key not found
ORA-06512: at "SYSTEM.ADD_SCORES", line 9
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-02291/
```

- Adding invalid information: Correct answers are greater than number of questions

```
EXECUTE ADD_SCORES('2252002','024002',10,11);
```

Result: Semantic constraints violated, no new values are added to the table.

```
BEGIN ADD_SCORES('2252002','024002',10,11); END;
*
ERROR at line 1:
ORA-20000: CORRECT ANSWERS MUST BE BETWEEN 0 AND NUMBER OF QUESTIONS
ORA-06512: at "SYSTEM.CHECK_SCORES", line 3
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_SCORES'
ORA-06512: at "SYSTEM.ADD_SCORES", line 9
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

- Updating invalid information: Correct answers are greater than number of questions.

```
UPDATE SCORES
SET CORRECT = 11,
    SCORE = (11 / QUESTIONS) * 10
WHERE STUDENT_ID = '2252003' AND QUIZ_ID = '024001';
```

Result: Semantic constraints violated, the values remain unchanged.

```
Error starting at line : 1 in command -
UPDATE SCORES
SET CORRECT = 11,
    SCORE = (11 / QUESTIONS) * 10
WHERE STUDENT_ID = '2252003' AND QUIZ_ID = '024001'
Error at Command Line : 1 Column : 8
Error report -
SQL Error: ORA-20000: CORRECT ANSWERS MUST BE BETWEEN 0 AND NUMBER OF QUESTIONS
ORA-06512: at "SYSTEM.CHECK_SCORES_UPDATE", line 3
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_SCORES_UPDATE'

https://docs.oracle.com/error-help/db/ora-20000/20000. 00000 -  "%s"
*Cause:    The stored procedure 'raise_application_error'
           was called which causes this error to be generated.
*Action:   Correct the problem as described in the error message or contact
           the application administrator or DBA for more information.
```

4.6. Add Procedures and Triggers to create and edit quizzes

- Add a procedure for creating quizzes.

This procedure receives information of a new quiz, then inserts into the Quizzes table a new value containing information of the newly added quiz.

```
CREATE OR REPLACE PROCEDURE INSERT_QUIZ (
      QUIZ_ID    IN QUIZZES.QUIZ_ID%TYPE,
      START_TIME IN QUIZZES.START_TIME%TYPE,
      END_TIME   IN QUIZZES.END_TIME%TYPE,
      TIME_LIMIT IN QUIZZES.TIME_LIMIT%TYPE,
      QUESTIONS  IN QUIZZES.QUESTIONS%TYPE
)
AS
BEGIN
      INSERT INTO QUIZZES
      VALUES (QUIZ_ID, START_TIME, END_TIME, TIME_LIMIT, QUESTIONS);
END INSERT_QUIZ;
```

- Adding triggers

```
-- Creating triggers to add quizzes' information
CREATE OR REPLACE TRIGGER CHECK_INSERT_QUIZ
BEFORE INSERT ON QUIZZES
FOR EACH ROW
BEGIN
    IF (NOT regexp_like(:NEW.QUIZ_ID, '^\d+$')) THEN
        raise_application_error(-20000, 'QUIZ ID CAN ONLY CONTAIN NUMBERS');
    END IF;

    IF (:NEW.END_TIME <= :NEW.START_TIME) THEN
        raise_application_error(-20000, 'END DATE MUST BE AFTER START DATE');
    END IF;

    IF (:NEW.TIME_LIMIT <= 0) THEN
        raise_application_error(-20000, 'TIME LIMIT MUST BE POSITIVE');
    END IF;

    IF (:NEW.QUESTIONS <= 0) THEN
        raise_application_error(-20000, 'QUIZZES MUST HAVE AT LEAST ONE
QUESTION');
    END IF;
END;


-- Creating triggers to update quizzes' information
CREATE OR REPLACE TRIGGER CHECK_UPDATE_QUIZ
BEFORE UPDATE ON QUIZZES
FOR EACH ROW
BEGIN
    IF (NOT regexp_like(:NEW.QUIZ_ID, '^\d+$')) THEN
    raise_application_error(-20000, 'QUIZ ID CAN ONLY CONTAIN NUMBERS');
    END IF;

    IF (:NEW.END_TIME <= :NEW.START_TIME) THEN
    raise_application_error(-20000, 'END DATE MUST BE AFTER START DATE');
    END IF;
```

```
    IF (:NEW.TIME_LIMIT <= 0) THEN
    raise_application_error(-20000, 'TIME LIMIT MUST BE POSITIVE');
    END IF;

    IF (:NEW.QUESTIONS <= 0) THEN
    raise_application_error(-20000, 'QUIZZES MUST HAVE AT LEAST ONE
QUESTION');
    END IF;
END;
```

Testing the procedure:

- Adding valid information.

Creating a new quiz:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
EXECUTE INSERT_QUIZ('024004','01-12-2024','01-01-2025',45,30);
```

After valid information is typed in, the new values are added to the table:

	QUIZ_ID	START_TIME	END_TIME	TIME_LIMIT	QUESTIONS
1	024001	01-11-2024	02-11-2024	30	10
2	024002	03-11-2024	04-11-2024	30	10
3	024003	05-11-2024	06-11-2024	30	10
4	024004	01-12-2024	01-01-2025	45	30

Editing a quiz:

```
UPDATE QUIZZES
SET TIME_LIMIT = 20 WHERE QUIZ_ID = ‘024002’;
```

The table becomes:

	QUIZ_ID	START_TIME	END_TIME	TIME_LIMIT	QUESTIONS
1	024001	01-11-2024	02-11-2024	30	10
2	024002	03-11-2024	04-11-2024	20	10
3	024003	05-11-2024	06-11-2024	30	10
4	024004	01-12-2024	01-01-2025	45	30

- Adding invalid information: Quiz ID already existed in the table.

```
EXECUTE INSERT_QUIZ('024003','01-12-2024','01-01-2025',45,30);
```

Result: Unique constraint violated, no new values are added.

```
BEGIN INSERT_QUIZ('024003','01-12-2024','01-01-2025',45,30); END;
*
ERROR at line 1:
ORA-00001: unique constraint (SYSTEM.SYS_C008325) violated
ORA-06512: at "SYSTEM.INSERT_QUIZ", line 10
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-00001/
```

- Adding invalid information: End date before start date.

```
EXECUTE INSERT_QUIZ('024005','01-12-2024','30-11-2024',45,30);
```

Result: Semantic constraint violated, no new values are added.

```
BEGIN INSERT_QUIZ('024005','01-12-2024','30-11-2024',45,30); END;
*
ERROR at line 1:
ORA-20000: END DATE MUST BE AFTER START DATE
ORA-06512: at "SYSTEM.CHECK_INSERT_QUIZ", line 7
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_QUIZ'
ORA-06512: at "SYSTEM.INSERT_QUIZ", line 10
ORA-06512: at line 1

https://docs.oracle.com/error-help/db/ora-20000/
```

- Adding invalid information: Negative time limit.

```
EXECUTE INSERT_QUIZ('024005','01-12-2024','04-12-2024',-5,30);
```

Result: Semantic constraint violated, no new values are added.

```
BEGIN INSERT_QUIZ('024005','01-12-2024','04-12-2024',-5,30); END;
*
ERROR at line 1:
ORA-20000: TIME LIMIT MUST BE POSITIVE
ORA-06512: at "SYSTEM.CHECK_INSERT_QUIZ", line 11
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_QUIZ'
ORA-06512: at "SYSTEM.INSERT_QUIZ", line 10
ORA-06512: at line 1
```

- Adding invalid information: Quiz with no questions

```
EXECUTE INSERT_QUIZ('024005','01-12-2024','04-12-2024',30,0);
```

Result: Semantic constraint violated, no new values are added.

```
BEGIN INSERT_QUIZ('024005','01-12-2024','04-12-2024',30,0); END;
*
ERROR at line 1:
ORA-20000: QUIZZES MUST HAVE AT LEAST ONE QUESTION
ORA-06512: at "SYSTEM.CHECK_INSERT_QUIZ", line 15
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_INSERT_QUIZ'
ORA-06512: at "SYSTEM.INSERT_QUIZ", line 10
ORA-06512: at line 1
```

4.7. Add Procedures and Triggers to create and edit questions

- Add a procedure for creating questions.

This procedure receives information of a new question corresponding to a quiz, then inserts into the Questions table a new value containing information of the newly added question.

```
-- Procedure to add questions
CREATE OR REPLACE PROCEDURE ADD_QUESTION (
    QUIZ_ID     IN QUESTIONS.QUIZ_ID%TYPE,
    QUESTION_ID IN QUESTIONS.QUESTION_ID%TYPE,
    QUESTION    IN QUESTIONS.QUESTION%TYPE,
    ANSWER_A    IN QUESTIONS.ANSWER_A%TYPE,
    ANSWER_B    IN QUESTIONS.ANSWER_B%TYPE,
    ANSWER_C    IN QUESTIONS.ANSWER_C%TYPE,
    ANSWER_D    IN QUESTIONS.ANSWER_D%TYPE,
    ANSWER_E    IN QUESTIONS.ANSWER_E%TYPE,
    CORRECT     IN QUESTIONS.CORRECT%TYPE
)
AS
BEGIN
    INSERT INTO QUESTIONS
    VALUES (QUIZ_ID, QUESTION_ID, QUESTION, ANSWER_A, ANSWER_B, ANSWER_C,
ANSWER_D, ANSWER_E, CORRECT);
END ADD_QUESTION;
```

- Add triggers

```
-- Trigger for adding/updating questions
CREATE OR REPLACE TRIGGER CHECK_ADD_QUESTION
BEFORE INSERT ON QUESTIONS
FOR EACH ROW
BEGIN
    IF (NOT regexp_like(:NEW.QUESTION_ID, '^\d+$')) THEN
    raise_application_error(-20000, 'QUESTION ID CAN ONLY CONTAIN NUMBERS');
    END IF;
```

```
    IF (:NEW.CORRECT != 'A' AND :NEW.CORRECT != 'B' AND :NEW.CORRECT != 'C'
AND :NEW.CORRECT != 'D' AND :NEW.CORRECT != 'E') THEN
    raise_application_error(-20000, 'CORRECT ANSWER MUST BE BETWEEN A AND
E');
    END IF;
END;

CREATE OR REPLACE TRIGGER CHECK_UPDATE_QUESTION
BEFORE UPDATE ON QUESTIONS
FOR EACH ROW
BEGIN
    IF (NOT regexp_like(:NEW.QUESTION_ID, '^\d+$')) THEN
    raise_application_error(-20000, 'QUESTION ID CAN ONLY CONTAIN NUMBERS');
    END IF;

    IF (:NEW.CORRECT != 'A' AND :NEW.CORRECT != 'B' AND :NEW.CORRECT != 'C'
AND :NEW.CORRECT != 'D' AND :NEW.CORRECT != 'E') THEN
    raise_application_error(-20000, 'CORRECT ANSWER MUST BE BETWEEN A AND
E');
    END IF;
END;
```

Testing the procedure:

- Adding valid information:

```
EXECUTE
ADD_QUESTION('024004','000031','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','A
NSWER_D','ANSWER_E','A');
EXECUTE
ADD_QUESTION('024004','000032','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','A
NSWER_D','ANSWER_E','B');
EXECUTE
ADD_QUESTION('024004','000033','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','A
NSWER_D','ANSWER_E','C');
EXECUTE
ADD_QUESTION('024004','000034','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','A
NSWER_D','ANSWER_E','D');
EXECUTE
ADD_QUESTION('024004','000035','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','A
NSWER_D','ANSWER_E','E');
```

New questions get added to the table.

	QUIZ_ID	QUESTION_ID	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	CORRECT
17	024002	000017	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
18	024002	000018	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
19	024002	000019	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
20	024002	000020	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
21	024003	000021	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
22	024003	000022	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
23	024003	000023	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
24	024003	000024	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
25	024003	000025	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
26	024003	000026	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
27	024003	000027	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
28	024003	000028	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
29	024003	000029	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
30	024003	000030	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
31	024004	000031	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
32	024004	000032	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	B
33	024004	000033	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	C
34	024004	000034	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	D
35	024004	000035	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	E

- Updating valid information:

```
UPDATE QUESTIONS SET CORRECT = 'B' WHERE QUESTION_ID = '000001';
```

	QUIZ_ID	QUESTION_ID	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	CORRECT
1	024001	000001	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	B
2	024001	000002	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A
3	024001	000003	QUESTION	ANSWER_A	ANSWER_B	ANSWER_C	ANSWER_D	ANSWER_E	A

- Adding invalid information: Correct answer is out of scope

```
EXECUTE
ADD_QUESTION('024004','000036','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','A
NSWER_D','ANSWER_E','F');
```

Result: Semantic constraint violated, no new values are added.

```
BEGIN ADD_QUESTION('024004','000036','QUESTION','ANSWER_A','ANSWER_B','ANSWER_C','ANSWER_D','ANSWER_E','F'); END;
*
ERROR at line 1:
ORA-20000: CORRECT ANSWER MUST BE BETWEEN A AND E
ORA-06512: at "SYSTEM.CHECK_ADD_QUESTION", line 7
ORA-04088: error during execution of trigger 'SYSTEM.CHECK_ADD_QUESTION'
ORA-06512: at "SYSTEM.ADD_QUESTION", line 14
ORA-06512: at line 1
```

4.8. Add functions to calculate total of students and lecturers

- Add a function to calculate total of students

```
-- Function to calculate total of students
CREATE OR REPLACE FUNCTION SUMOFSTUDENTS
RETURN INT
AS
    SUM_STUDENTS INT;
BEGIN
    SELECT COUNT(*) INTO SUM_STUDENTS
    FROM STUDENT;

    RETURN SUM_STUDENTS;
END;
```

- Add function to calculate number of students in a faculty:

```
CREATE OR REPLACE FUNCTION SUMSTUDENTSFACULTY (FACULTYNAME IN
STUDENT.FACULTY%TYPE)
RETURN INT
AS
    SUM_STUDENTS INT;
BEGIN
    SELECT COUNT(*) INTO SUM_STUDENTS
    FROM STUDENT
    WHERE FACULTY = FACULTYNAME;

    RETURN SUM_STUDENTS;
END;
```

- Add function to calculate number of lecturers:

```
CREATE OR REPLACE FUNCTION SUMOFLECTURERS
RETURN INT
AS
    SUM_LECTURERS INT;
BEGIN
    SELECT COUNT(*) INTO SUM_LECTURERS
    FROM LECTURER;

    RETURN SUM_LECTURERS;
END;
```

- Add function to calculate number of lecturers in a faculty:

```
CREATE OR REPLACE FUNCTION SUMLECTURERSFACULTY (FACULTYNAME IN
LECTURER.L_FACULTY%TYPE)
RETURN INT
AS
    SUM_LECTURERS INT;
BEGIN
    SELECT COUNT(*) INTO SUM_LECTURERS
    FROM LECTURER
    WHERE L_FACULTY = FACULTYNAME;
```

```
    RETURN SUM_LECTURERS;
END;
```

4.9. Create Assertions

Since many relational databases (e.g., Oracle, MySQL, and PostgreSQL) do not directly support assertions, the assertions are typically enforced through triggers or application-layer validation.

Here on Oracle SQL Developer, we created this trigger as an assertion to enforce that the number of questions of a quiz in table Questions cannot exceed the number of questions of a quiz specified in table Quizzes:

```
CREATE OR REPLACE TRIGGER VALIDATE_QUESTION_COUNT
BEFORE INSERT OR UPDATE ON QUESTIONS
FOR EACH ROW
DECLARE
    MAX_QUESTIONS INT;
    CURRENT_COUNT INT;
BEGIN
    SELECT QUESTIONS INTO MAX_QUESTIONS
    FROM QUIZZES
    WHERE QUIZ_ID = :NEW.QUIZ_ID;

    SELECT COUNT(*) INTO CURRENT_COUNT
    FROM QUESTIONS
    WHERE QUIZ_ID = :NEW.QUIZ_ID;

    IF (CURRENT_COUNT + 1 > MAX_QUESTIONS) THEN
        RAISE_APPLICATION_ERROR(-20001, 'Exceeding the allowed number of
questions for this quiz.');
    END IF;
END;
```

VI. DEVELOPING THE APPLICATION - CONNECTING THE DATABASE

Within this assignment, we are implementing some working interfaces for all three user groups, along with some simple features to demonstrate the feasibility of the designed database:

Lecturer management:

- View the list of lecturers.
- Add lecturers.
- Edit lecturers' information.
- Delete lecturers.
- Search lecturers' information.

Student management:

- View the list of students.
- Add students.
- Edit student's information.
- Delete students.
- Search student's information.

Quizzes management:

- View the list of quizzes.
- Create quizzes.
- Edit quizzes' information.

Questions management.

- View the list of questions.
- Add questions.
- Edit questions' information.

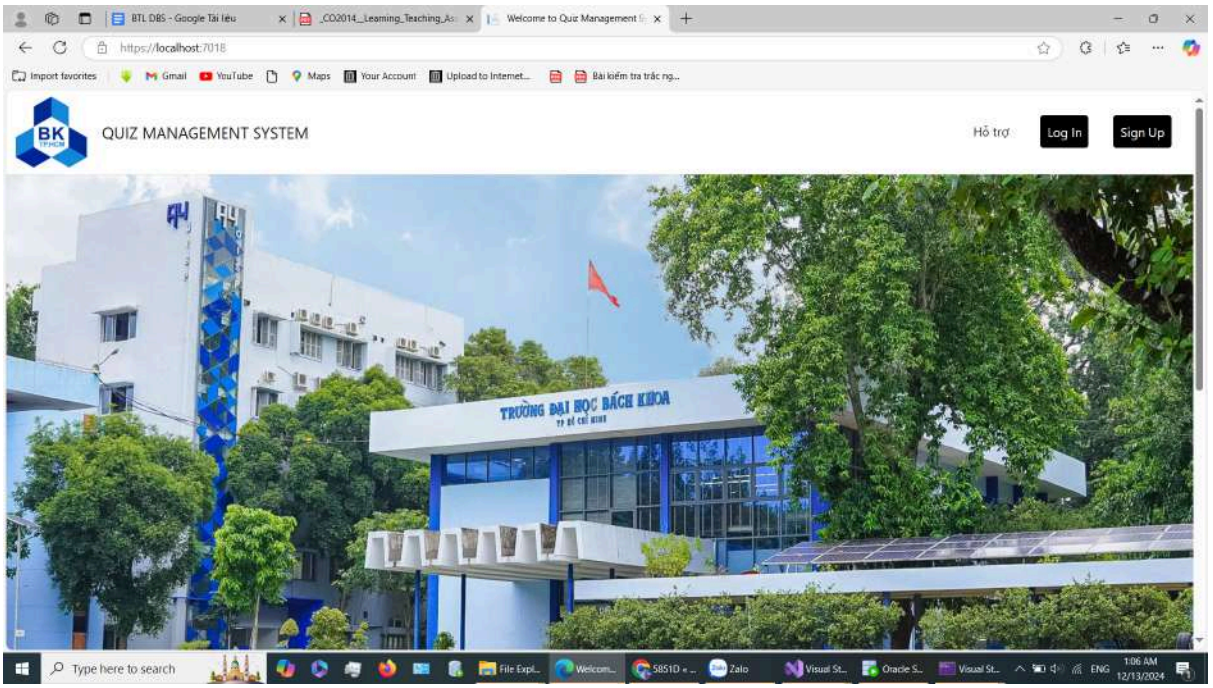
Scores management:

- View the list of scores.
- Search scores of students
-

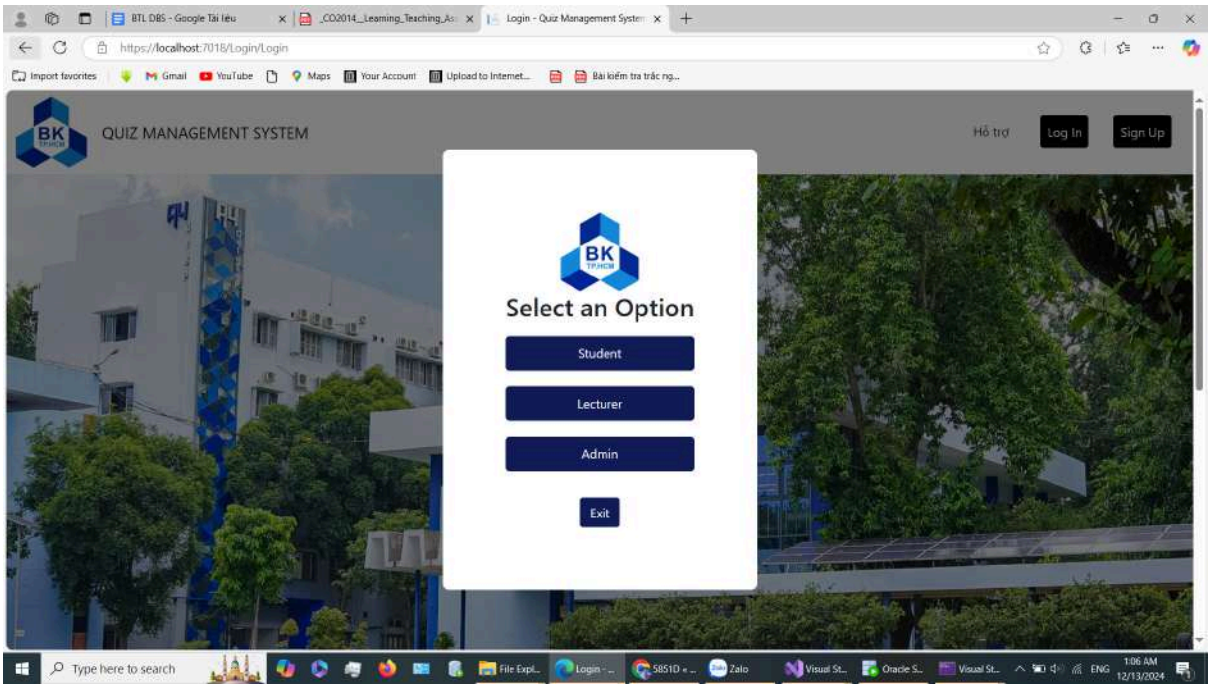
The main technology(ies) for the application is ASP.NET Core MVC for the framework. We used ODP.NET to connect to the implemented database.

Link to our GitHub repository: https://github.com/longtqb04/QuizManagement

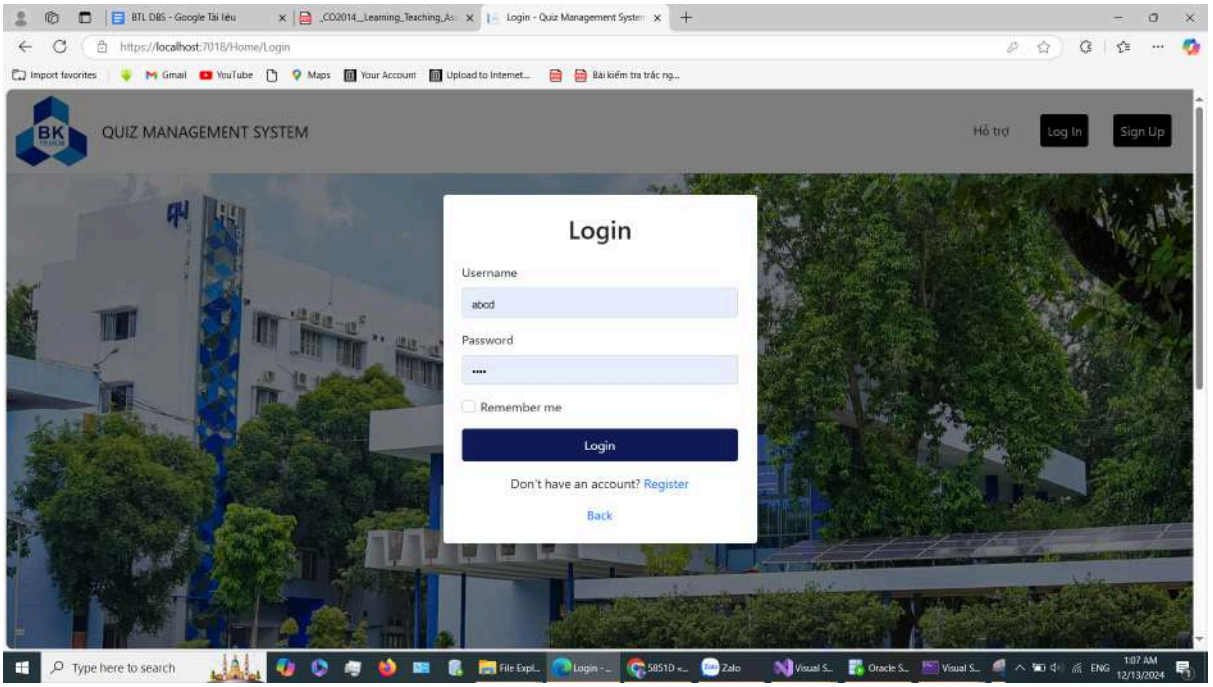
1. Home Page



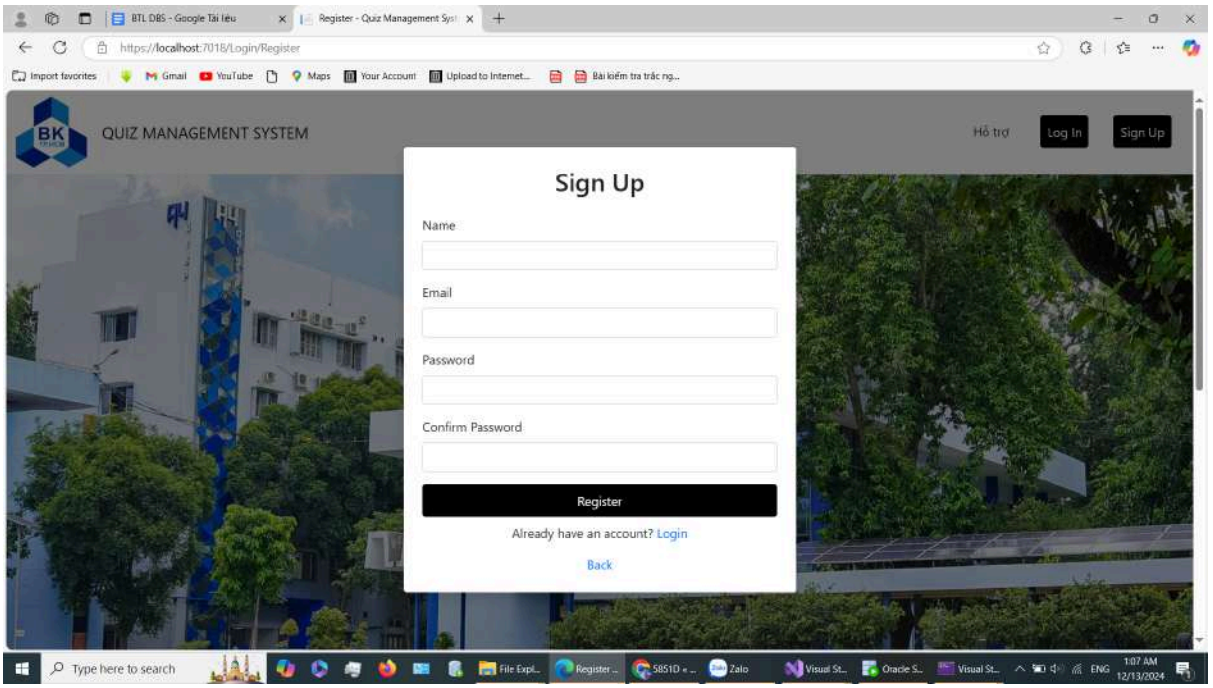
2. Login/Register Page



Selecting options

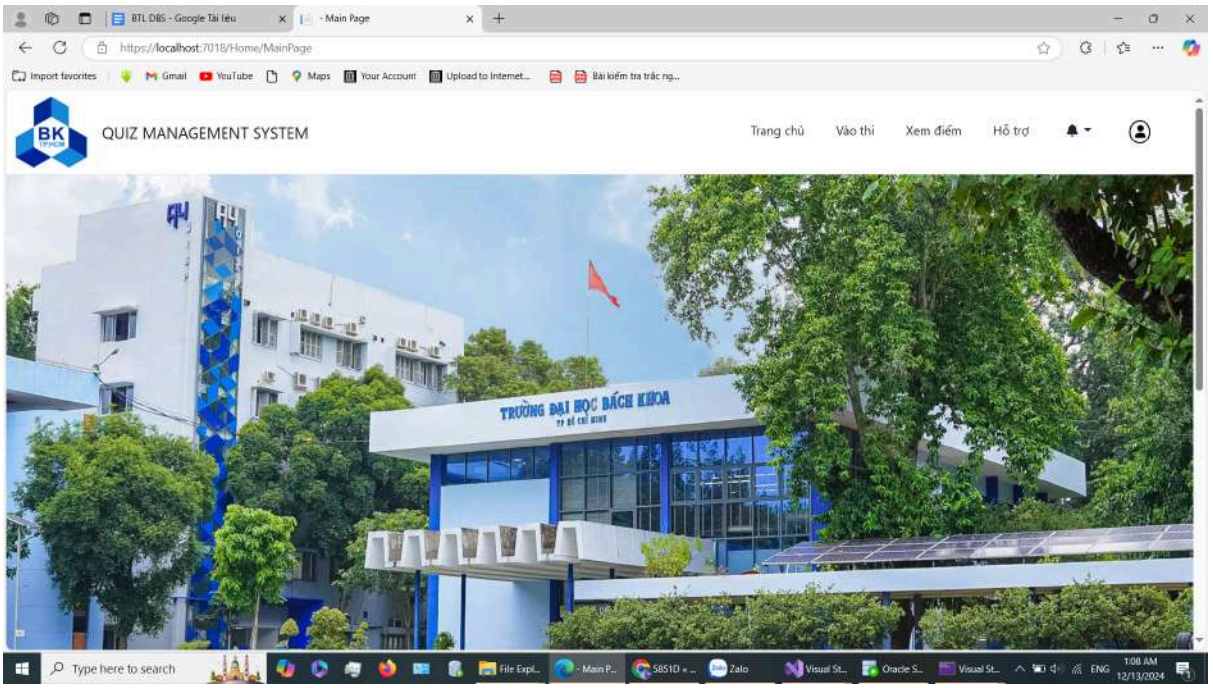


Login page

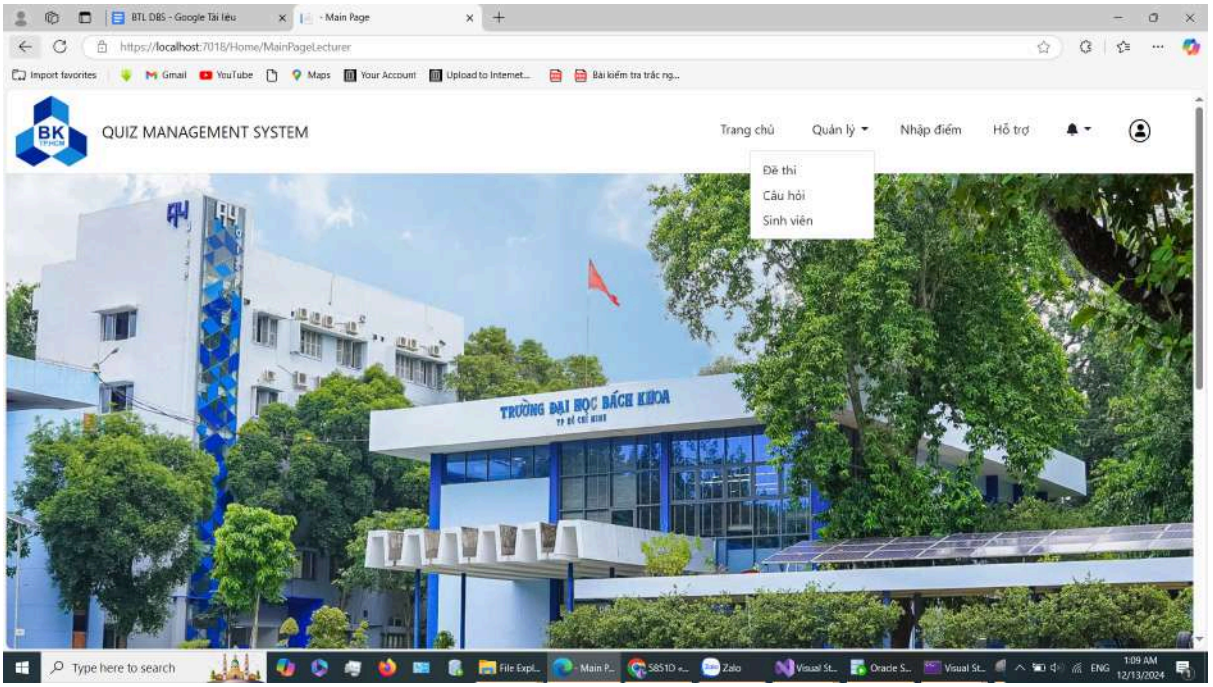


Register page

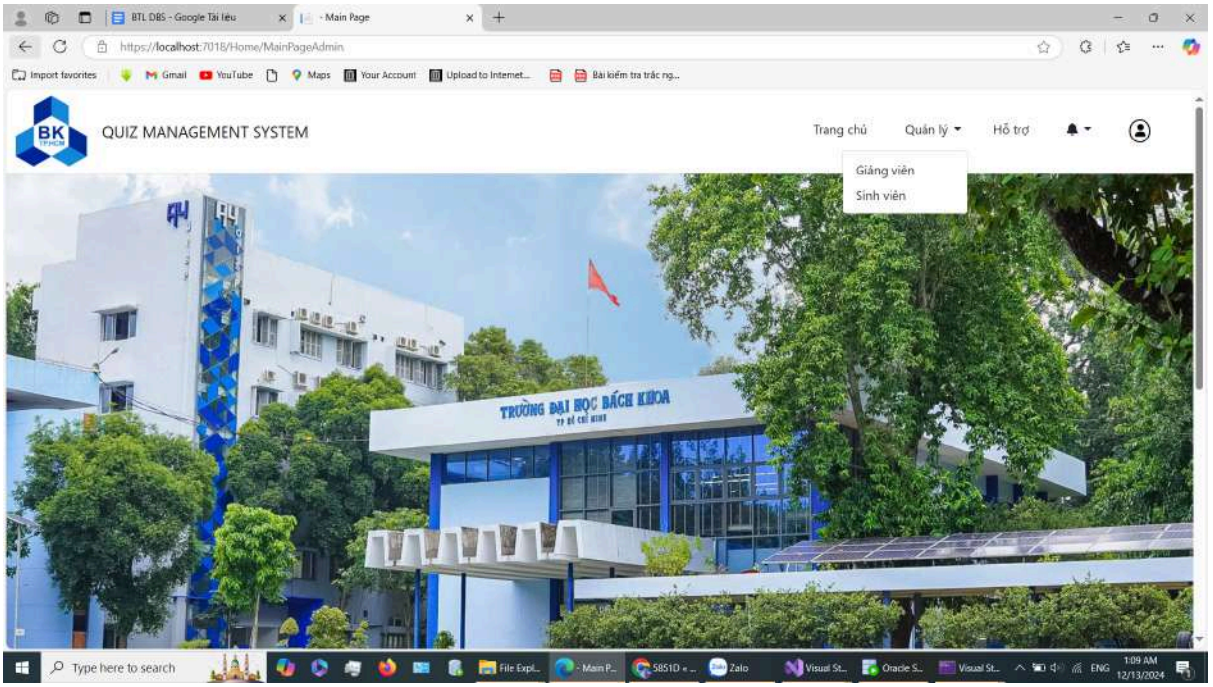
3. Navigation Bars



Student's view

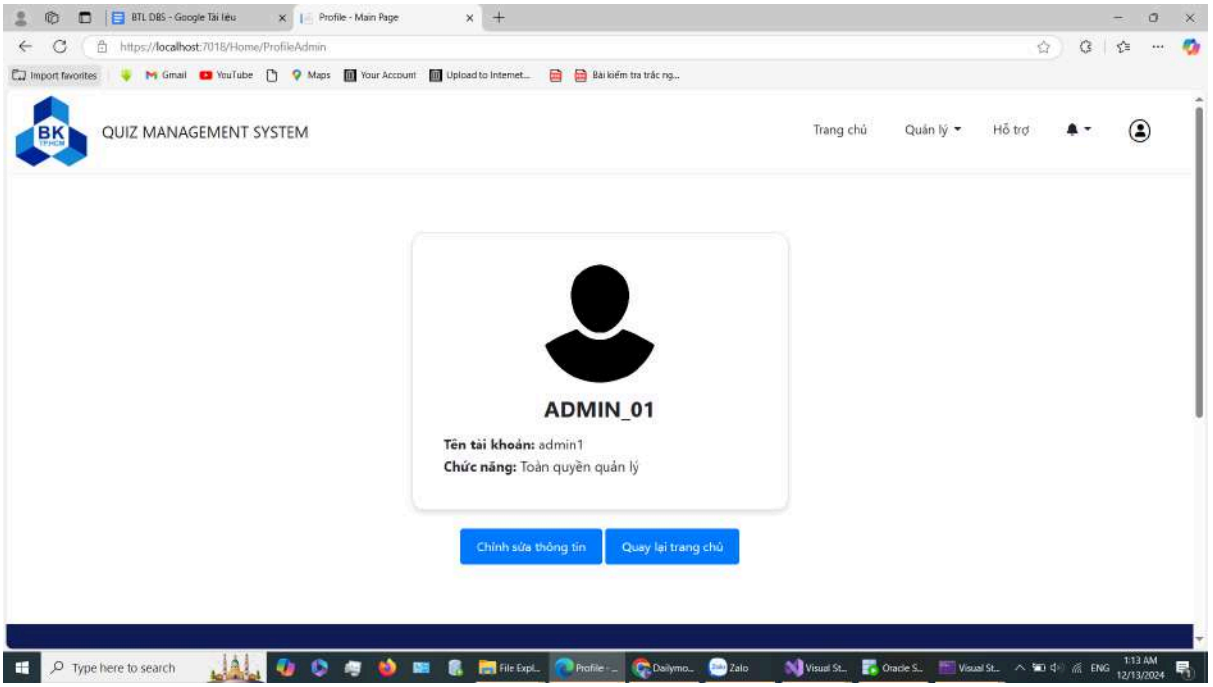


Lecturer's view

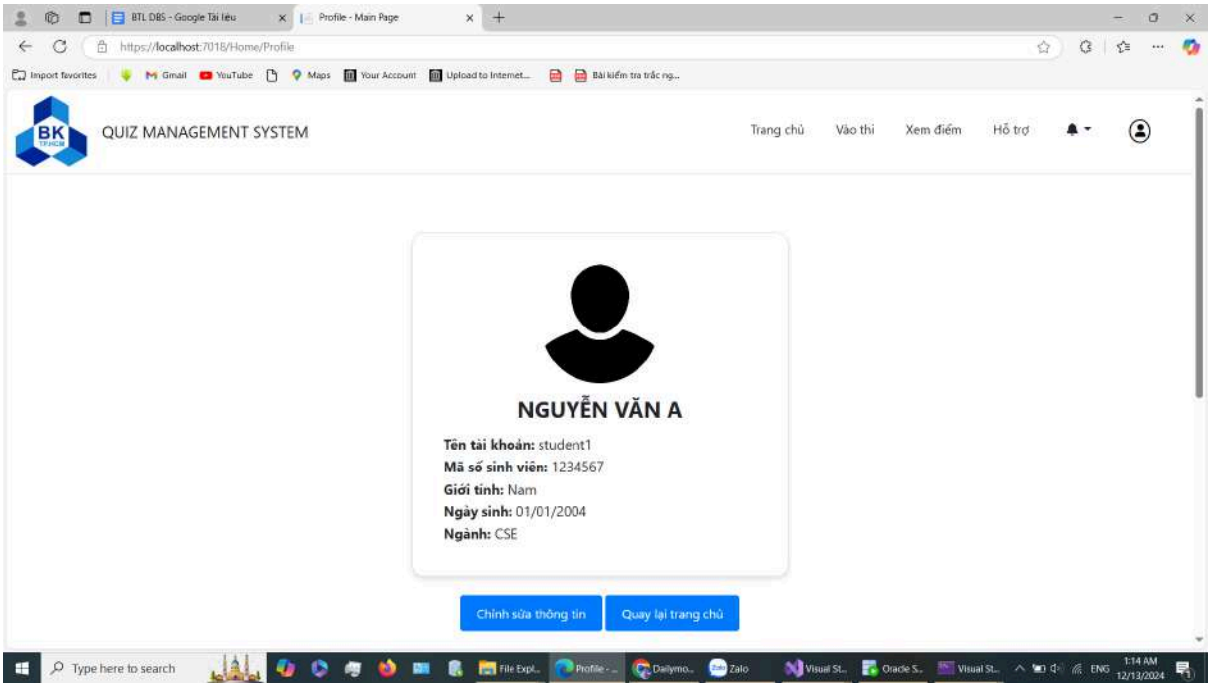


Admin's view

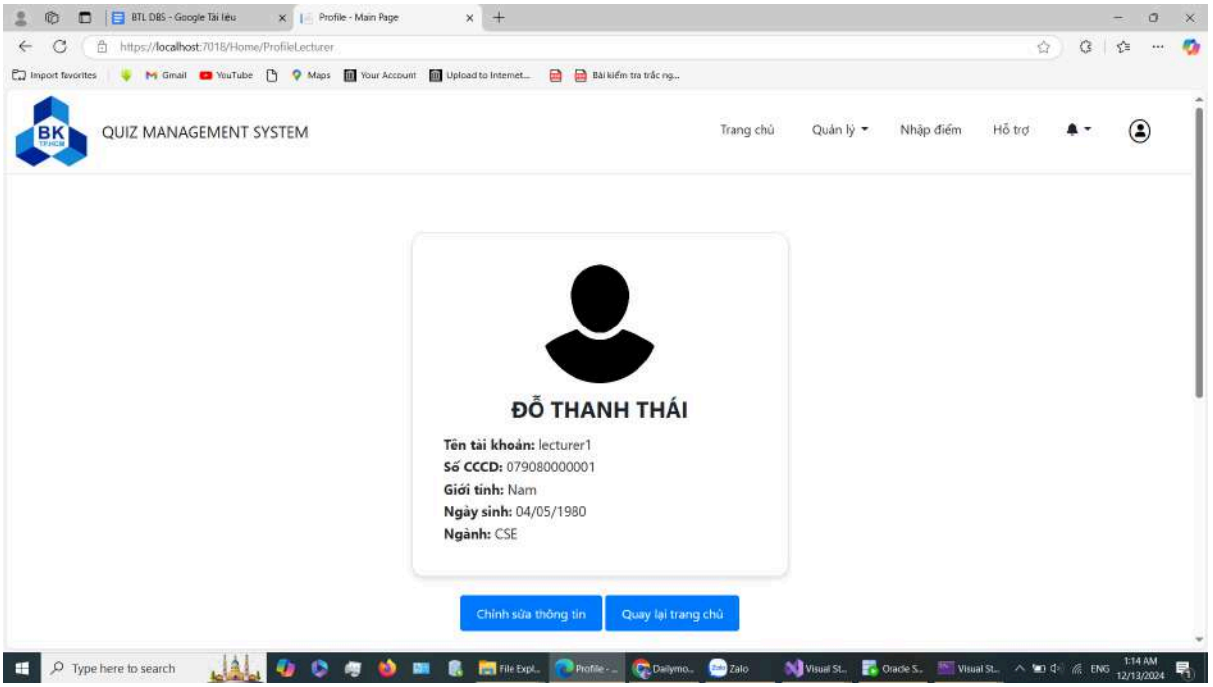
4. Profile pages



Admin's view

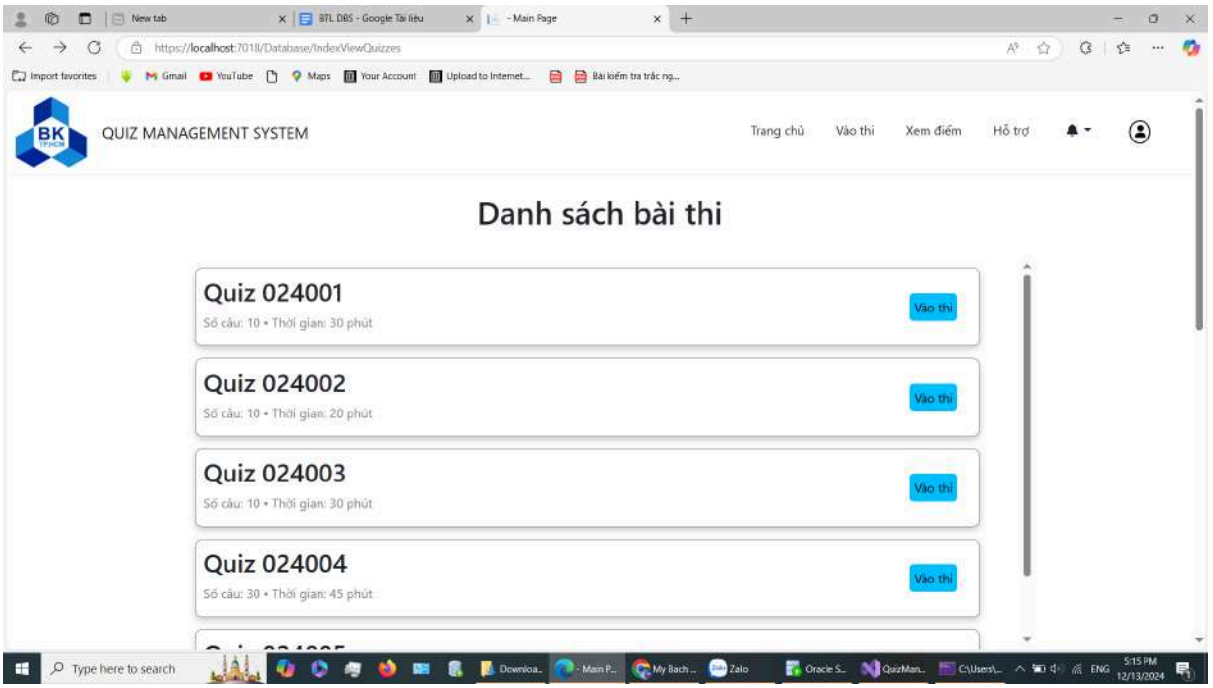


Student's view

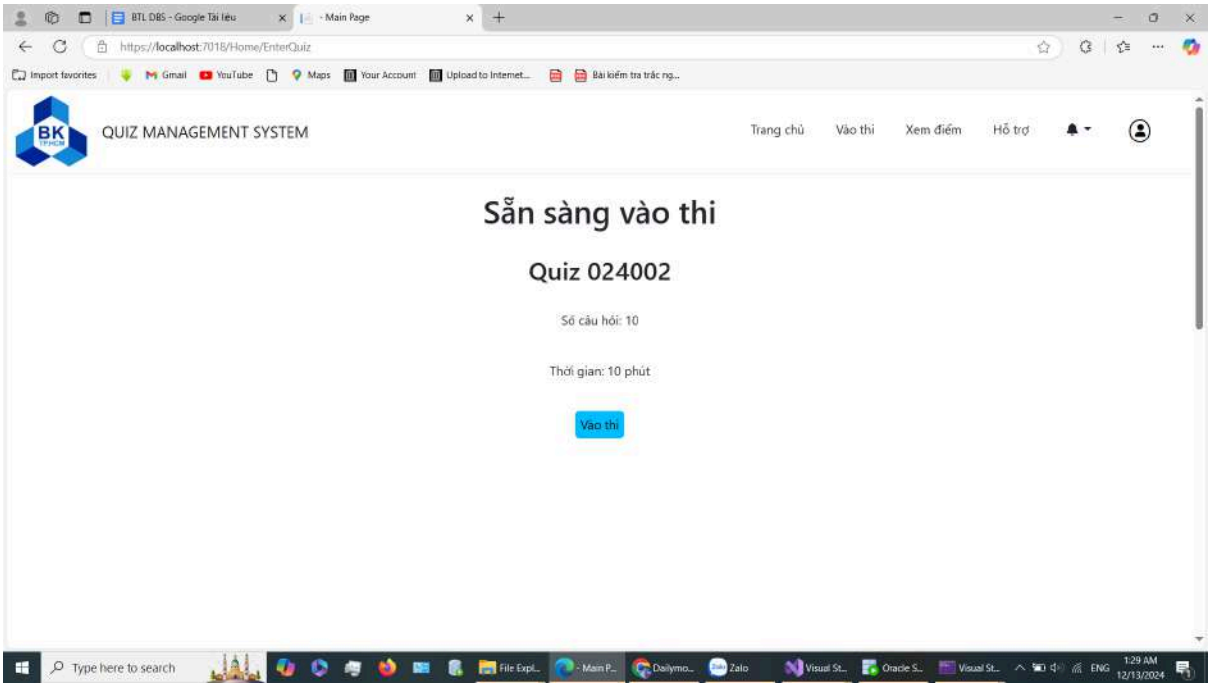


Lecturer's view

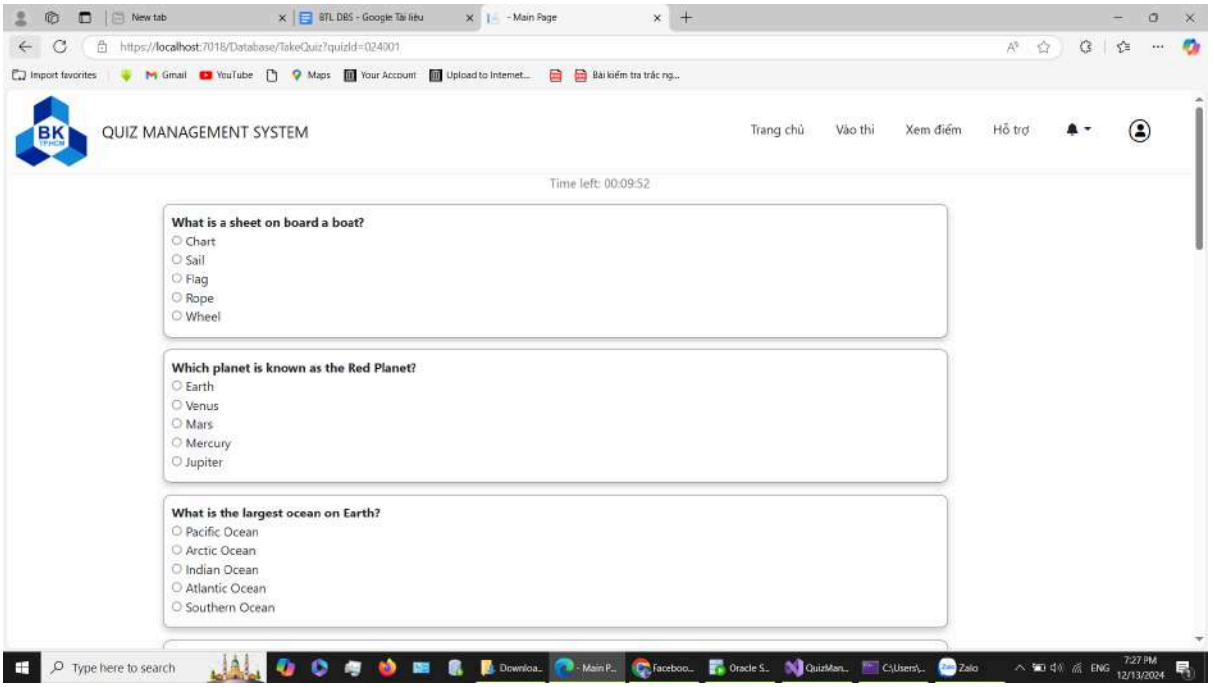
5. Entering a quiz



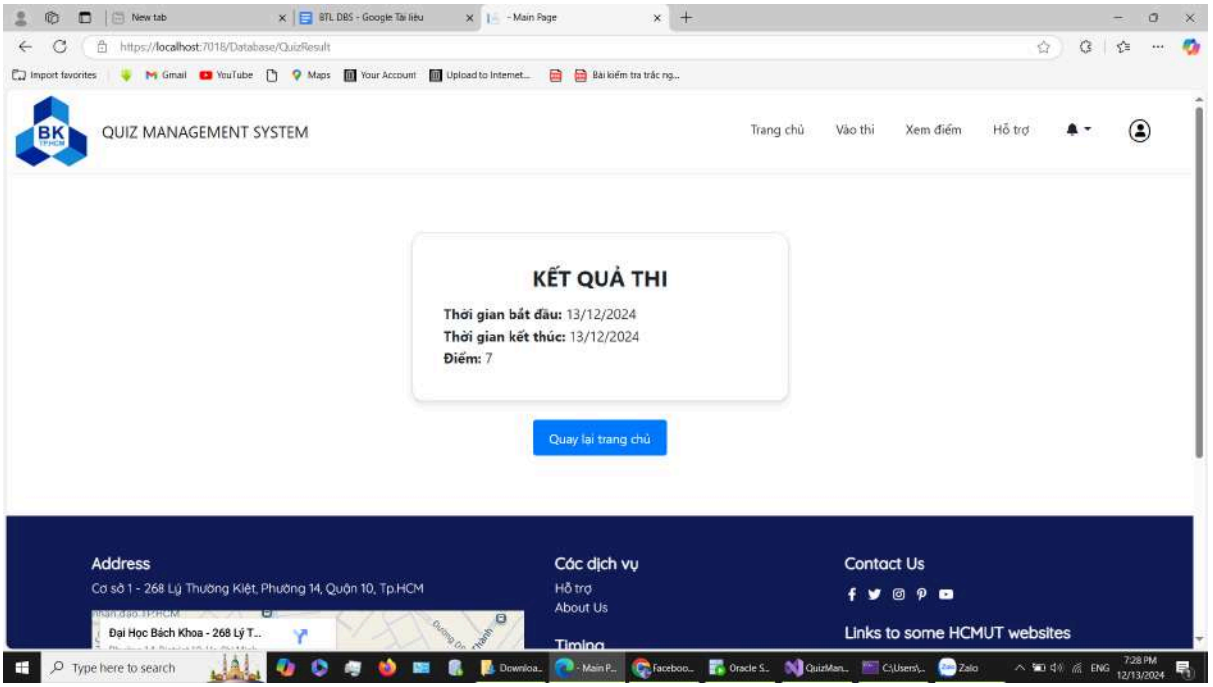
Choosing a quiz



Ready to enter the quiz



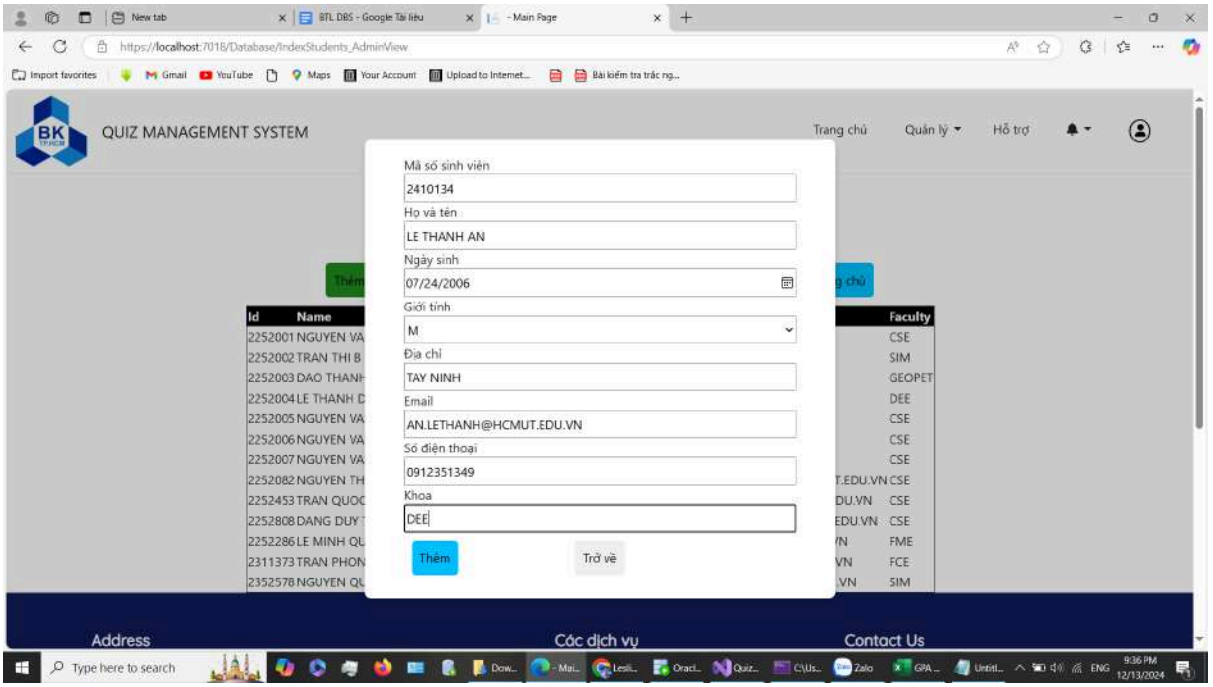
Taking a quiz



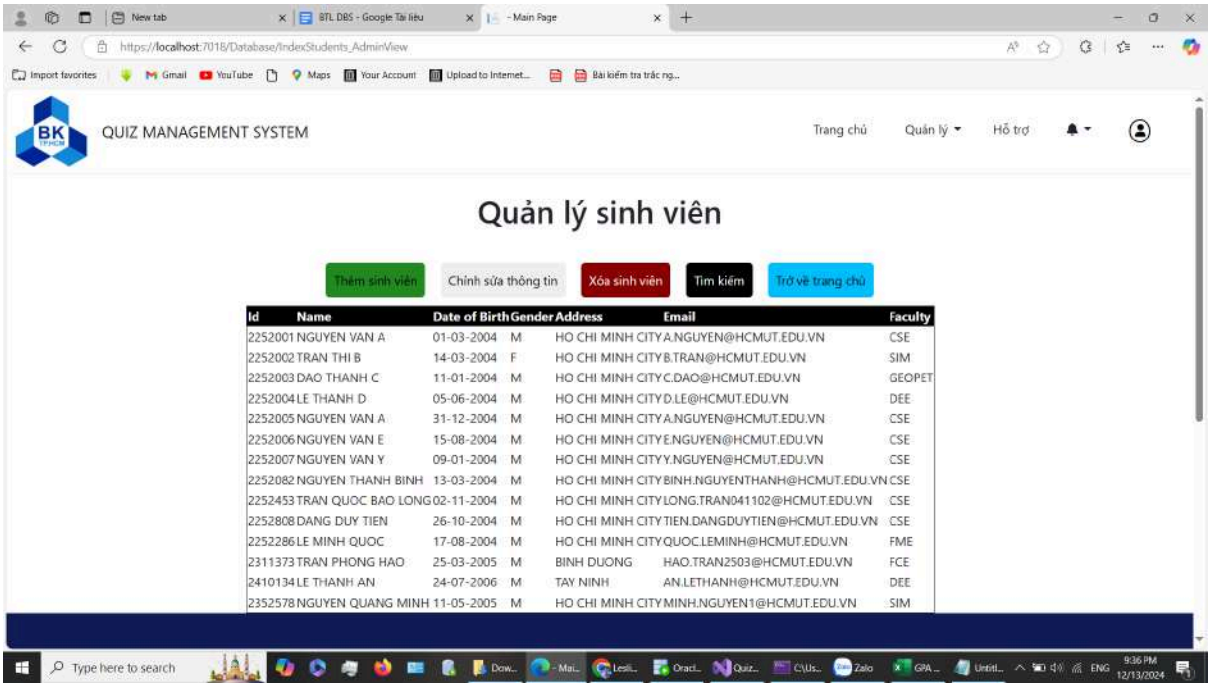
Displaying the results

6. Adding information

For example, adding new students:

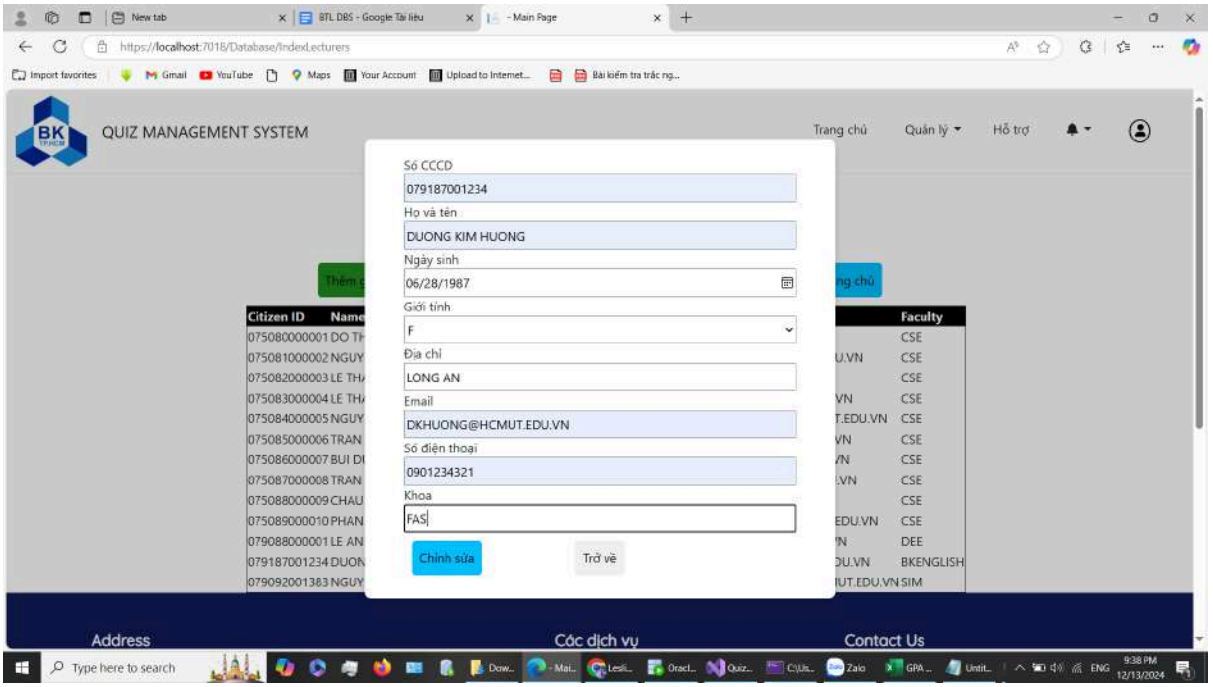


The result:

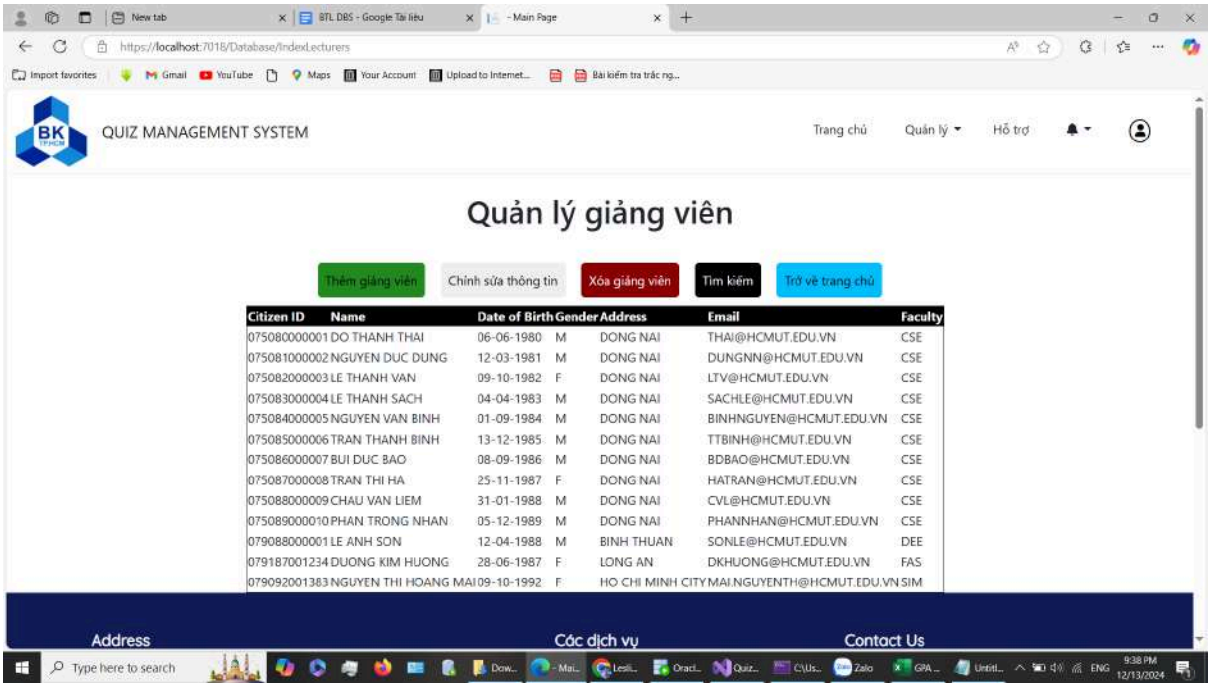


7. Editing information

For example, editing a lecturer's information:

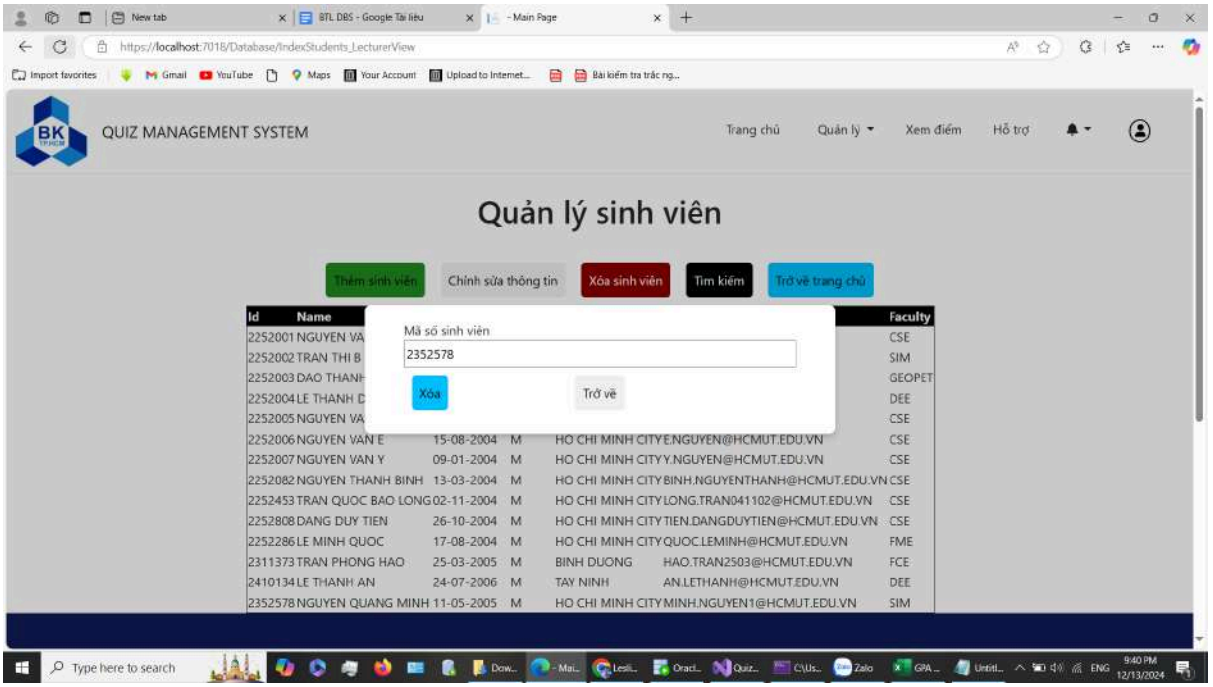


The result:

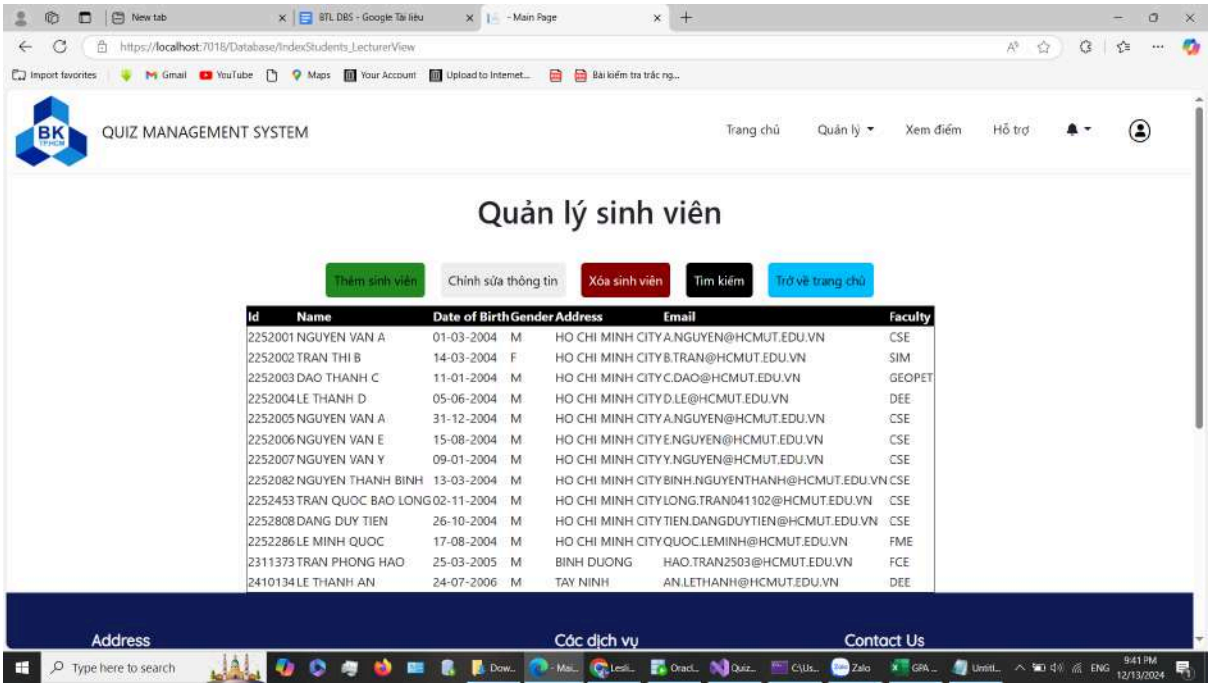


8. Deleting information

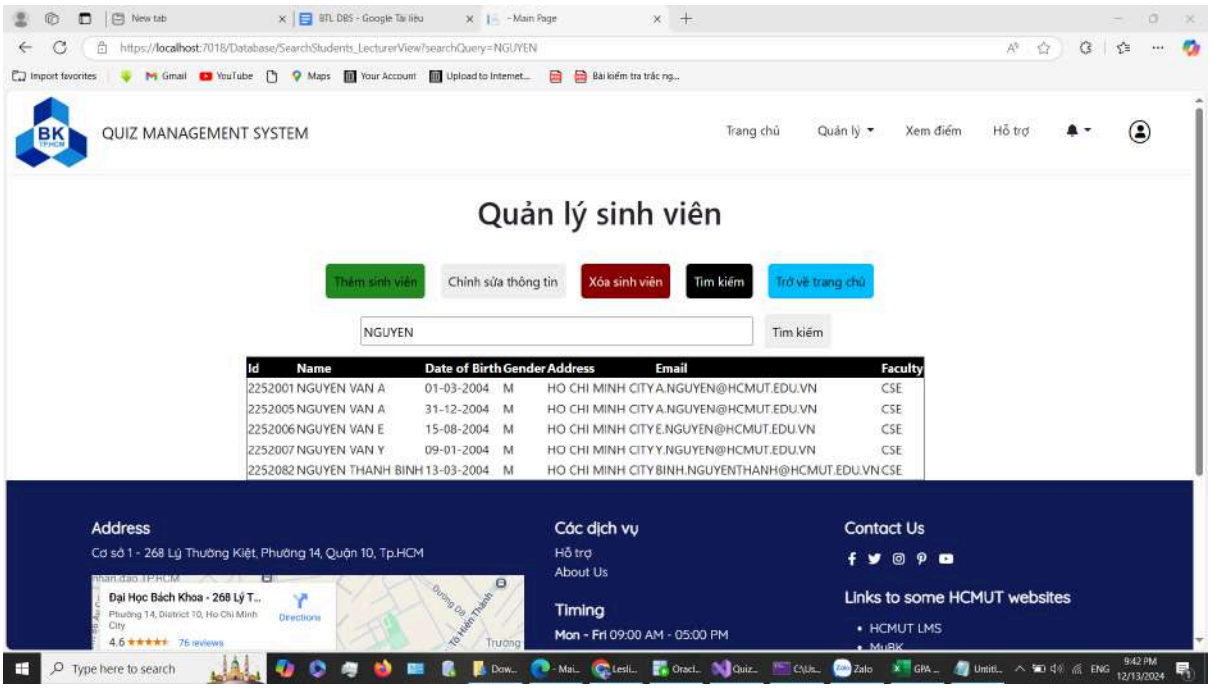
For example, deleting a student by their Student ID:



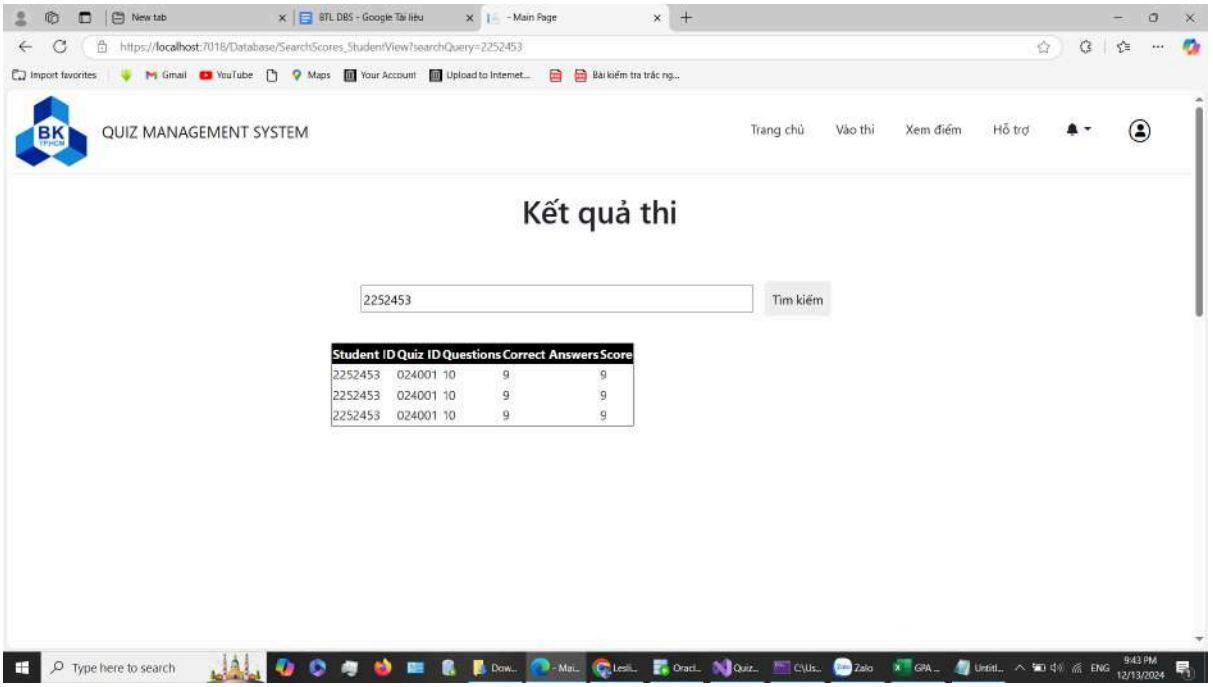
The result:



9. Searching information



Searching all students with surname “NGUYEN”



Searching quiz scores of student ID 2252453

VII. CONCLUSION

To conclude, the development of a simple quiz management system has provided us with a comprehensive understanding of database systems and their practical applications. Throughout this project, we have explored various aspects of database design, including the creation of conceptual schemas, normalization of data, and implementation of efficient query mechanisms.

By addressing the requirements of a quiz management system, we have demonstrated our ability to design and manage a robust database that can handle user interactions, question management, and result tracking.

REFERENCES

1. Elmasri, Ramez, Navathe, Shamkant B. *Fundamentals of Database Systems, 7th Edition*. 2015.

2. W3Schools. *SQL Tutorial*.
   URL: https://www.w3schools.com/SQL/deFault.asp

3. DotNetTutorials. *Introduction to ASP.NET Core MVC Framework*.
   URL:
   https://dotnettutorials.net/lesson/introduction-asp-net-core-mvc/#google_vignette