VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM-590014

A Mini-Project Report

On

"EVENT MANAGEMENT SYSTEM"

A Mini-project report submitted in partial fulfillment of the requirements for the award of the degree of ***Bachelor of Engineering in Computer Science and Engineering*** *of Visvesvaraya Technological University, Belgaum.*

Submitted by:

Menahi Shayan	1AM17CS101
P Vamshi Prasad	1AM17CS127
Nishank Swamy D N	1AM17CS121
Podaralla Revathi	1AM17CS132

Under the Guidance of:
Mrs. V.R. Srividhya

(Asst. Prof., Dept. of CSE)

Department of Computer Science and Engineering
AMC Engineering College,
18th K.M, Bannerghatta Main Road, Bangalore-560 083

2019-2020

AMC Engineering College,

18th K.M, Bannerghatta Main Road, Bangalore-560 083

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the mini-project work entitled **"EVENT REGISTRATION SYSTEM"** has been successfully carried out by Menahi Shayan (1AM17CS101), P Vamshi Prasad (1AM17CS127), Nishank Swamy D N (1AM17CS121), Podaralla Revathi (1AM17CS132) bonafide students of **AMC Engineering College** in partial fulfilment of the requirements for the award of degree in **Bachelor of Engineering in Computer Science and Engineering** of **Visvesvaraya Technological University, Belgaum** during academic year 2019-2020 . It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Guide:

Mrs. SRIVIDHYA V.R.	**Dr. LATHA C A**	**HOD,**
Asst. Prof., Dept. of CSE	**Dept. of CSE**	

Dr. A.G NATARAJ
Principal, AMCEC

Examiners: **Signature with Date**

1.

2.

ACKNOWLEDGEMENT

The joy and satisfaction that accompany the successful completion of any task would be incomplete without the mention of those who made it possible. We are glad to express our gratitude towards our prestigious institution **AMC ENGINEERING COLLEGE** for providing us with utmost knowledge, encouragement and the maximum facilities in undertaking this project.

We wish to express sincere thanks to our respected chairman **Dr. K. R. Paramahamsa** and beloved principal **Dr. A.G Nataraj** for all their support.

We express our deepest gratitude and special thanks to
Dr. Latha C A, **H.O.D, Dept. Of Computer Science Engineering**, for all her guidance and encouragement.

We sincerely acknowledge the guidance and constant encouragement of our mini-project guide, **Mrs. V. R. Srividhya , Assistant Prof. , Dept. Of Computer Science Engineering.**

NAME	USN
Menahi Shayan	**1AM17CS101**
P Vamshi Prasad	**1AM17CS127**
Nishank Swamy D N	**1AM17CS121**
Podaralla Revathi	**1AM17CS132**

ABSTRACT

The entitled project "**EVENT MANAGEMENT SYSTEM**" is made keeping in mind all the aspects of the Events.The system allows only registered users to login and new users are allowed to register on the application. The project provides most of the basic functionality required for an event. It allows the user to select from a list of event types.It will be capable of doing all the necessary operations/functions that are done in any Event for example-registration of events, ticketing, login-logout, transactions ,history of payments ,minimalistic UI ,UPI ,WhatsApp ,etc. Since all the work that is to be done by this software can also be done manually, but this consumes time and internet. So this software will be a relief to those who have to do all this work manually. The knowledge of computers and programming has become a basic skill needed to survive in present information based on society.

The motive is to make such kind of software which is very easy to use. There will not be need of any training and the person who does not have much knowledge of computers can also use this . All the history of the customers will be kept for further enquiries.

CONTENTS

Chapter 1

INTRODUCTION

This application is used by two users:

1) Admin
2) User

The main aim is to automate the entire day to day activities of Events like: Logins, new registrations, number of events, transactions, ticketing, UPI and updating.

The users can login to the login page and can register an event according to his type by providing the necessary details, admin updates the event records, he is an authorized user.

"EVENT MANAGEMENT SYSTEM" has been designed to computerize the following functions that are performed by the system:

- Event details functions
- Registration details
- Login -Logout details
- Event Registration facility
- M-ticket generation
- Security & Encryption
- Responsive UI/UX
- Encrypted Database
- Secure Authentication

1.1 OBJECTIVES

During several decades personnel function has been transformed from a relatively obscure record keeping staff to a central and top level management function. There are many factors that have influenced this transformation like technological advances, professionalism and general recognition of human beings as most important resources.

- A user based management system is designed to handle all the primary information required to calculate monthly statements. Separate database is maintained to handle all the details required for the correct statement and generation.
- The intention is to introduce more user friendliness in the various activities such as record updating, maintenance, searching.
- The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying the identification of the customer.
- Similarly, record maintenance and updating can also be accomplished by using the identification of customer with all details being automatically generated. These details are also being automatically updated in the file thus keeping record up-to-date.
- The entire information has been maintained in the database or files and whoever wants to retrieve can’t retrieve, only authorized (admin) user can retrieve the necessary information which can be easily accessible from the file.

1.2 SCOPE

PERFORMANCE: Manual handling of the record is time consuming and highly prone to error. Hence to improve the performance, computerized and user friendly system is undertaken.

EFFICIENCY: Basic need is efficiency. So whenever the new user submits his/her details, it has to be updated automatically.It works on all platforms like Mac, Windows, Linux, iOS, Android etc..

CONTROL: The complete control is under the admin who has the password to access and illegal access is not supported. All controls is under the administrator and has the rights just to see the entry and not to change the records or any entries.

SECURITY: This is the main criteria of the proposed system. Since illegal access may cause the generation of tickets, So security has been given. Here we are using AES256 security.

Chapter 2

SYSTEM SPECIFICATION

2.1 HARDWARE REQUIREMENTS

PROCESSOR: i9 10^{th} gen

RAM: 32GB

HARD DISK: 4TB

2.2 SOFTWARE REQUIREMENTS

OS: Cross platform, 64 bit

CODING LANGUAGE:

- FRONT END: NODE-JS

  It is an open source ,cross-platform,runtime environment that allows developers to create all kinds of server-side tools and applications in JavaScript. It uses an event-driven,non-blocking I/O model that makes it lightweight and efficient ,perfect for data-intensive real-time applications that run across distributed devices.

- BACK END: SQL

  It is a relational database whose components (tables, forms, queries) are linked (related). The linkages between database components are created by making relationship links between them. The relationship can be:

  - One component and another(one-one relationship)
  - One component related to several other components(one-many)
  - Several database components(many-many)

Creation of relationships between the database components reduces data redundancies and enhances ease of access of the information.

Chapter 3

DESIGN

3.1 Description of Event Management Database System

- The details of events is stored into the Event tables respective with all tables.

Each entity (USN, R_ID, E_ID, Name, Section, Department, Coordinators) contains primary keys, foreign keys and composite keys.

- The above mentioned keys combination is used as composite keys here.
- There is one-to-one and one-to-many relationships available between USN,.
- All the entities are normalized and reduce duplicity of records.
- Indexing is implemented on each tables of Event Management System tables for fast query execution.

There will be many events conducted in a country. Each event is identified by its E_ID. Each event provides many registrations.Events

are identified by unique E_ID and event name . Each event is performed on the mentioned venue and the mentioned time. An event has many number of rules . Each student can register multiple events.

Each users name, USN, E_ID are stored in registration database. Each student is provided with a unique USN. Each event is assigned with a coordinator and the users who are registered to the events will be updated regarding the event by the assigned coordinator to the event.

The coordinator login to his profile and registers the event according to the student requirement. The user is needed to give details to register for the events. After registering for the event the users are provided with a unique R_ID known as reference id. The payment is done based on the tariff (price) of the event via any payment mode that is using UPI or CASH. The M-ticket is generated with the QR CODE based on the R_ID of the event.

The R_ID is responsible for keeping track of the status of the students and registration of the events. The details of the students and the event registration are accessible only by the coordinator. The payment is handled by the coordinator in the desk. After the user registered , the coordinators retrieves the student details, number of events, accordingly and M-TICKET is generated.

STEP 1:

Entities are:

1. Students
2. Events
3. Registration
4. Coordinators
5. Transaction
6. Authorization

STEP 2:

RELATIONS:

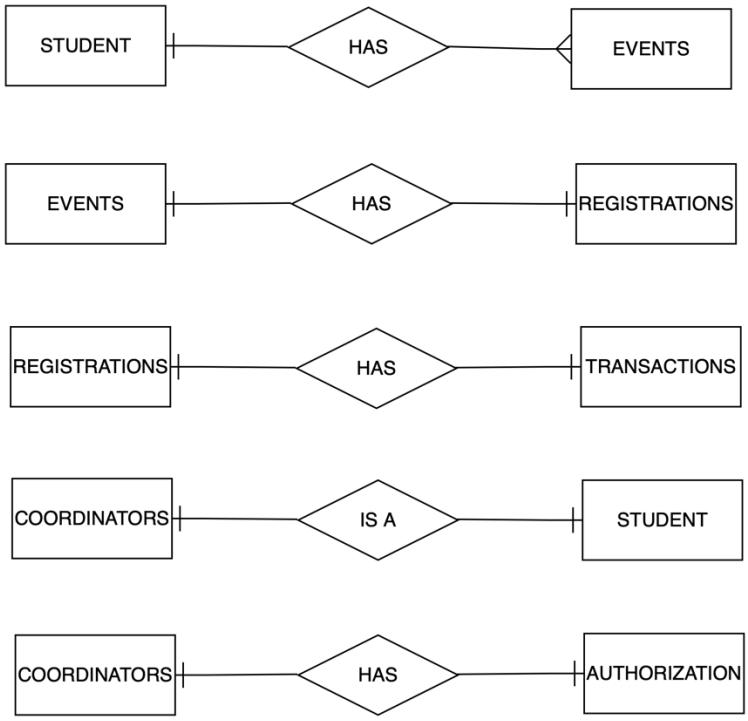

Figure 3.1

STEP 3:

Key attributes:

- STUDENTS - USN
- EVENTS – E_ID
- COORDINATOR - USN
- REGISTRATION – R_ID

- RULES – E_ID, RULE_NO
- TRANSACTION - R_ID
- AUTHORIZATION - USN

STEP 4:

Other attributes

- STUDENTS - USN, DEPT, PHONE, SECTION, SEM, NAME
- EVENTS - TEAM COUNT, E_ID, NAME, COLOR, VENUE, DURATION, TIME, DATE, CATEGORY, PRICE, RULES
- REGISTRATION - R_ID, E_ID, DESK_USN, TIMESTAMP, USN
- TRANSACTION - AMOUNT, R_ID, MODE, STATUS
- COORDINATORS - ROLE, USN, EVENT
- AUTHORIZATION - PASSWORD, USN

SCHEMA DIAGRAM

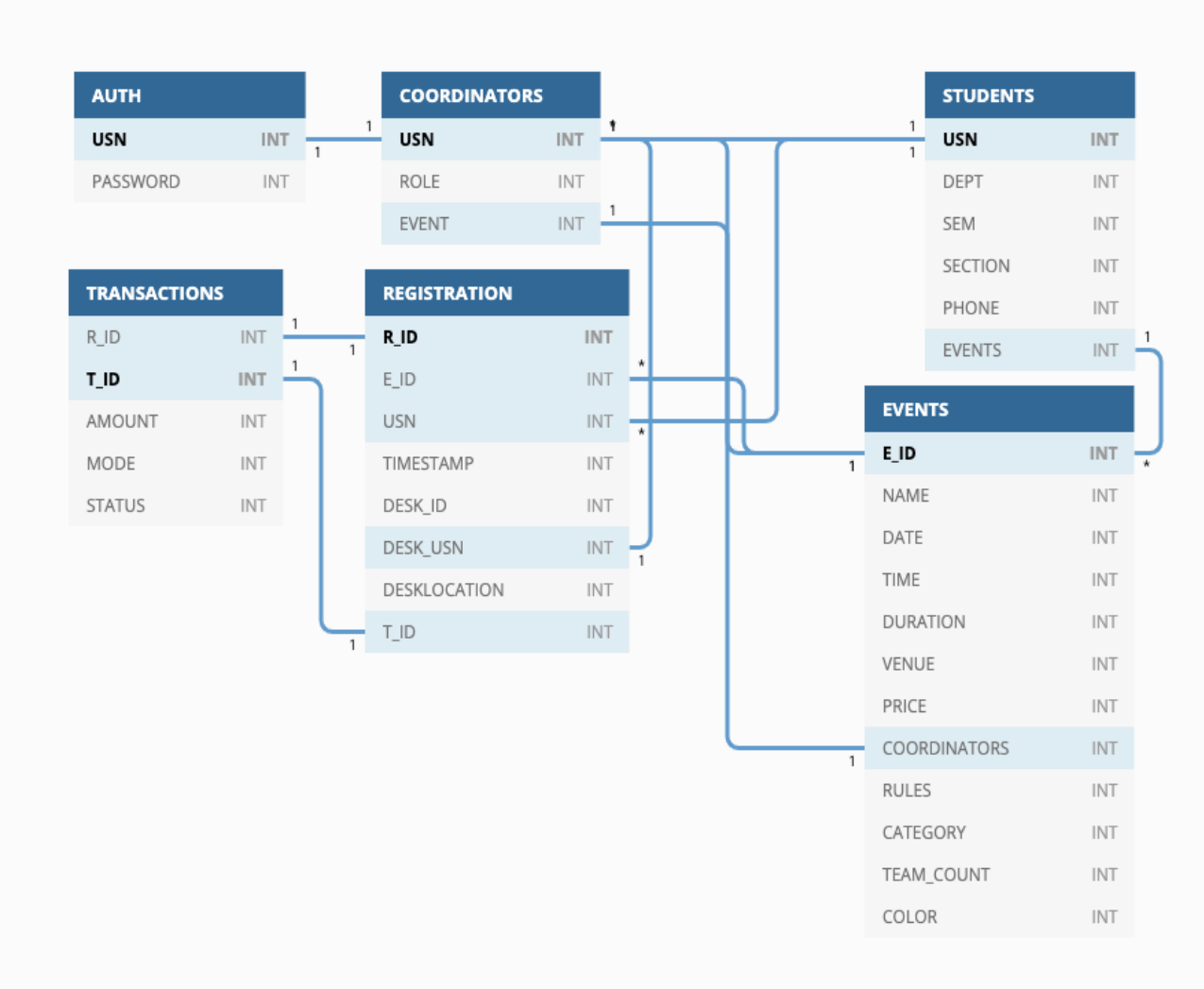

Figure 3.2

STEP 5:

ER DIAGRAM

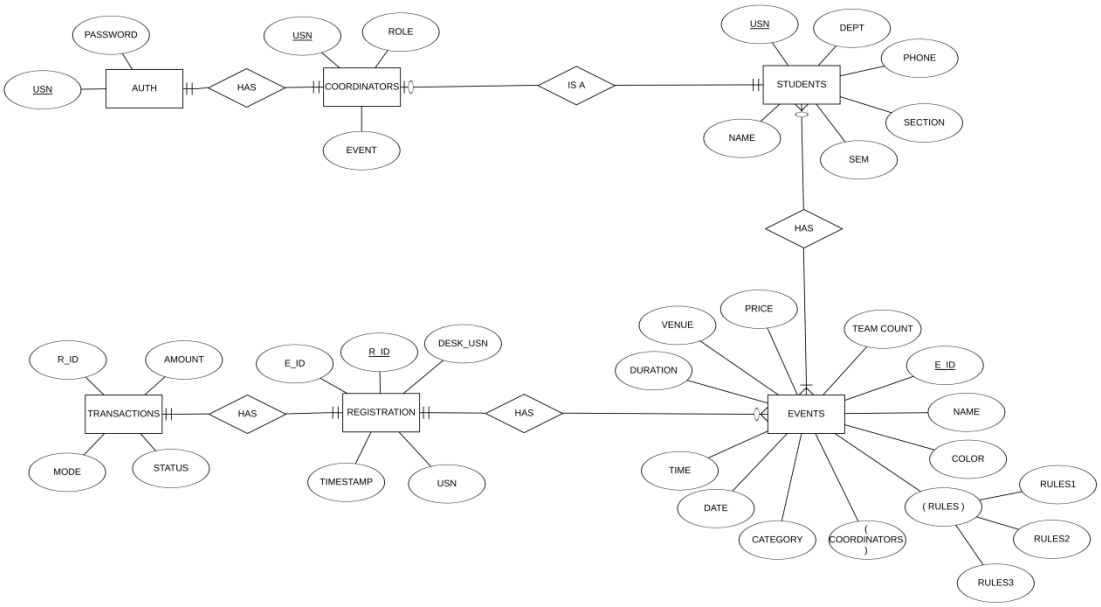

Figure 3.3

3.2 NORMALISATION

The basic Objectives of normalization are to reduce redundancy, which means that information is to be stored only once. Storing information several times leads to wastage of storage space and increase in the total size of data stored. Relations are normalized so that when relation in the database are to be altered during the lifetime of the database, information is not lost or introduces inconsistencies. The type of alterations normally needed for relation is:

- Insertion of new data values to relation. This should be possible without being forced to leave blank fields for some attributes.
- Deletion of a tuple, namely, a row of a relation . This should be possible without losing vital information unknowingly.

Functional Dependency:

As the concept of dependency is very important, it is essential that it should be understood first and then proceed to the idea of normalization. There is no fool-proof algorithmic method of identifying dependency.

Properties of normalized relations:

Ideals relation after normalization should have the following properties:

- No data values should be duplicated in different rows unnecessarily.
- A value must be specified (and required) for every attribute in a row.
- Each relation should be self-contained. In other words, if a row from a relation is deleted, important information should not be accidentally lost.
- When a row is added to a relation, other relations in the database should not be affected.
- A value of an attribute in a tuple may be changed independent of other tuples in the relation and other relations.

Consider the EVENTS table (refer to the schema diagram on figure 3.2).

The prime attributes identified are the attributes which is part of candidate key.

The non-prime attributes are not part of primary key.

When the EVENTS table is split into two tables having the following attributes:

Before Normalization-

Attributes rules was present in the event table before the normalization.(see table 3.1)

Rules attribute stores all the description about the event with multiple points which was containing multiple values.

Table 3.1

E_ID	NAME	DATE	TIME	DURATION	VENUE	PRICE	COORDINATORS	RULES	CATEGORY	TEAM_COUNT	COLOR
E1001	Code it	2019-11-15	10:00	60	LH-201	80	1AM17CS102	The team should complete the task in given time	TECHNICAL	2	YELLOW
E1002	Fashion show	2019-11-15	17:00	15	MAIN STAGE	800	1AM17CS103	The team must not use any vulgar clothes	MAIN STAGE	8	RED
E1003	Group singing	2019-11-15	10:00	7	MAIN STAGE	400	1AM17CS104	To be completed in given time	MAIN STAGE	5	RED
E1004	Group dance	2019-11-15	11:00	7	MAIN STAGE	500	1AM17CS105	To be completed in given time	MAIN STAGE	6	RED
E1005	Solo singing	2019-11-15	12:00	5	MAIN STAGE	100	1AM17CS106	To be completed in given time	MAIN STAGE	1	RED

1NF

- Each table cell should contain a single value.
- Each record needs to be unique.

According to table (3.2 and 3.3) rules attribute is made into a different table by this each table contains single values and each record is unique.

Therefore, the given tables are in 1NF (1st Normal Form)

2NF

- Be in 1NF
- Single Column Primary Key

According to table (3.2 and 3.3) rules attribute is made into a different table by this each column will be having single column primary key.

Therefore, the given tables are in 2NF (2nd Normal Form)

3NF

- Be in 2NF
- Rule 2- Has no transitive functional dependencies

According to table (3.2 and 3.3) rules attribute is made into a different table by this the two tables does not have transitive functional dependencies.
Therefore, the given tables are in 3NF (3rd Normal Form)

BCNF

- Be in 3NF
- Should not have more than one Candidate Key

According to table (3.2) rules attribute is made into a different table by this there are no multiple candidate keys .There are no other attributes in event table 3.2 which can be a candidate key.
Therefore, the given tables are in BCNF (BCNF Normal Form)

Table 3.2

E_ID	NAME	DATE	TIME	DURATION	VENUE	PRICE	COORDINATORS	CATEGORY	TEAM_COUNT	COLOR
E1001	Code it	2019-11-15	10:00	60	LH-201	80	1AM17CS102	TECHNICAL	2	YELLOW
E1002	Fashion show	2019-11-15	17:00	15	MAIN STAGE	800	1AM17CS103	MAIN STAGE	8	RED
E1003	Group singing	2019-11-15	10:00	7	MAIN STAGE	400	1AM17CS104	MAIN STAGE	5	RED
E1004	Group dance	2019-11-15	11:00	7	MAIN STAGE	500	1AM17CS105	MAIN STAGE	6	RED
E1005	Solo singing	2019-11-15	12:00	5	MAIN STAGE	100	1AM17CS106	MAIN STAGE	1	RED

Table 3.3

E_ID	RULE_NO	RULES
1	1	The team should complete the task in given time
1	2	No usage of internet
2	1	The team must not use any vulgar clothes
2	2	Should be based on some theme
2	3	Maximum of 8 persons in a team
3	1	To be completed in given time
3	2	Maximum of 6 persons in a team
3	3	To get there own bgms
4	1	To be completed in given time

3.1 TRIGGERS

A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

BEFORE and AFTER of Trigger:

BEFORE triggers run the trigger action before the triggering statement is run.
AFTER triggers run the trigger action after the triggering statement is run.

```
CREATE TRIGGER `ENCRYPT_INSERT` BEFORE INSERT ON `auth`
 FOR EACH ROW SET new.PASSWORD = AES_ENCRYPT(new.PASSWORD, 'nish')
```

The above trigger is executed after a new record is added to the auth table. It executes AES_ENCRYPT() function on the entered password using the given encryption key and stores the encrypted result in the table.

```
CREATE TRIGGER `TRANSACT` AFTER INSERT ON `registration`
 FOR EACH ROW INSERT INTO transactions(R_ID, AMOUNT) VALUES(NEW.R_ID,
(SELECT PRICE FROM events WHERE events.E_ID=NEW.E_ID))
```

The above trigger is executed when a new record in the registration table is added**.** This trigger inserts a new record in the transactions table containing the same R_ID as the new registration and Registration fee from events table for the respective E_ID.

Procedure

Stored Procedures are created to perform one or more DML operations on Database. It is nothing but the group of SQL statements that accepts some input in the form of parameters and performs some task and may or may not returns a value.

The most important part is parameters. Parameters are used to pass values to the Procedure. There are 3 different types of parameters, they are as follows:

1. **IN:**
   This is the Default Parameter for the procedure. It always recieves the values from calling program.
2. **OUT:**
   This parameter always sends the values to the calling program.
3. **IN OUT:**
   This parameter performs both the operations. It Receives value from as well as sends the values to the calling program.

REGISTER

If USN is not present in students table then add student usn, name, phone, sem, section, department.
If the usn, e_id pair does not exist in the registration table then

- add to registration as e_id, usn, desk_usn.
- Trigger will add transaction automatically as pending if not it will show error as duplicate registration.
- The transaction is updated as amount as amount, mode as payment and status as 'paid' where R_ID =_R_ID.
- Display the R_ID for the corresponding registration.

If the above statements become false then it will give error message saying "Duplicate registration".

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `REGISTER`(IN `_E_ID` INT(11),
IN `_USN` VARCHAR(20), IN `_NAME` VARCHAR(20), IN `_PHONE` VARCHAR(20),
IN `_SEM` INT(11), IN `_SECTION` VARCHAR(20), IN `_PAY` VARCHAR(20), IN
`_DESKUSN` VARCHAR(20), IN `_DEPT` VARCHAR(20))
   MODIFIES SQL DATA
BEGIN
DECLARE _COUNT INT(11);
DECLARE _R_ID INT(11);
DECLARE _EXISTCOUNT INT(11);
   SELECT
      COUNT(*)
          INTO _COUNT
   FROM
      students
```

```
    WHERE
        USN = _USN;

IF (_COUNT = 0) THEN
    INSERT INTO students(
        USN,
        NAME,
        PHONE,
        SEM,
        SECTION,
        DEPT
    )
VALUES(
    _USN,
    _NAME,
    _PHONE,
    _SEM,
    _SECTION,
    _DEPT
);
END IF;
SELECT COUNT(*) INTO _EXISTCOUNT FROM registration WHERE USN=_USN AND
E_ID=_E_ID;
IF (_EXISTCOUNT = 0) THEN
INSERT INTO registration(E_ID, USN, DESK_USN)
VALUES(_E_ID, _USN, _DESKUSN);

    SELECT
        R_ID
    INTO _R_ID
    FROM
        registration
    WHERE
        E_ID = _E_ID AND USN = _USN
```

```
      ORDER BY `TIMESTAMP` LIMIT 1 ;

UPDATE
   transactions
SET
   `MODE` = _PAY,
   `STATUS` = 'PAID'
WHERE
   R_ID = _R_ID;
   SELECT _R_ID LIMIT 1;
ELSE
     SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Duplicate registration for USN, E_ID pair';
END IF;
END$$
DELIMITER ;
```

Procedure
Login

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `LOGIN`(IN `_USN`
VARCHAR(11), IN `PASS` VARCHAR(30))
   READS SQL DATA
BEGIN

IF ((SELECT AES_ENCRYPT(PASS, 'nish') AS `PASSWORD`) = (SELECT `PASSWORD`
FROM `auth` WHERE `USN`=_USN)) THEN
SELECT true AS RESPONSE;
ELSE
SELECT false AS RESPONSE;
END IF;
```

```
END$$
DELIMITER ;
```

Procedure

Change password

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `CHANGEPASS`(IN `_USN` VARCHAR(11), IN `OLDPASS` VARCHAR(30), IN `NEWPASS` VARCHAR(30))
   MODIFIES SQL DATA
BEGIN

IF ((SELECT AES_ENCRYPT(OLDPASS,'nish') AS `PASSWORD`) = (SELECT `PASSWORD` FROM auth WHERE USN=_USN)) THEN
    UPDATE auth SET `PASSWORD`= AES_ENCRYPT(NEWPASS,'nish') WHERE USN=_USN;
ELSE
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Incorrect Password. Please check the old password you have entered.';
END IF;
END$$
DELIMITER ;
```

Chapter 4

IMPLEMENTATION AND CODING

4.1 USER DEFINED FUNCTIONS