

TEST SEATING MANAGEMENT SYSTEM

A COURSE PROJECT REPORT

By

PASUMARTHI MADHAVA VENKATA PRANAV (RA2011030010103)

KAARTHIK SAI CHARAN AYINENI (RA2011030010104)

Under the guidance of

Dr. Thanga Revathi . S , Assistant Professor

In partial fulfilment for the Course

of

18CSC303J - DB MANAGEMENT SYSTEMS

in Networking and Communications



**FACULTY OF ENGINEERING AND TECHNOLOGY SRM
INSTITUTE OF SCIENCE AND TECHNOLOGY**

Kattankulathur, Chenpalattu District

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this mini project report "**Test Seating Management System** " is the bonafide work of **PASUMARTHI MADHAVA VENKATA PRANAV (RA2011030010103)** and **KAARTHIK SAI CHARAN AYINENI (RA2011030010104)** who carried out the project work under my supervision.

SIGNATURE

Dr. Thanga Revathi. S

Assistant Professor

Networking and Communications

SRM Institute of Science and Technology

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr.C.MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors. We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy**, for his encouragement

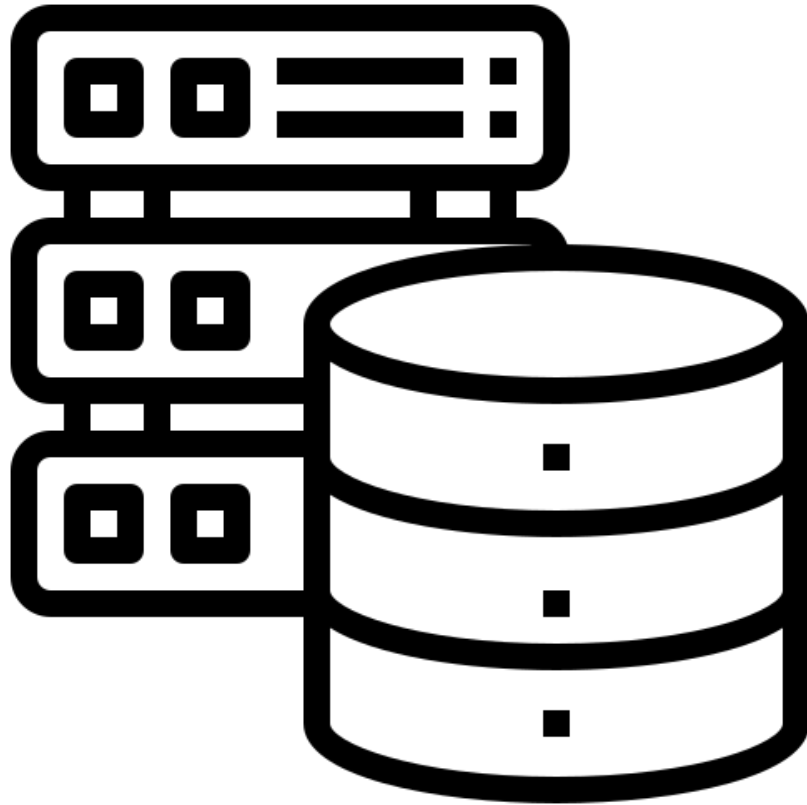
We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal**, for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman**, for imparting confidence to complete my course project We are highly thankful to our my Course project Faculty **Dr. Thanga Revathi . S , Assistant Professor , Networking and Communications** , for his/her assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **HoD Dr. Annapurani Panaiyappan , Professor and Head, Networking and Communications** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

TEST SEATING ALLOCATION



PASUMARTHI MADHAVA VENKATA PRANAV – RA2011030010103

KAARTHIK SAI CHARAN AYINENI – RA2011030010104

INDEX

Serial No	Content	Page No
1.	Abstract	1
2.	Introduction	2
3.	Literature Survey	5
4.	Architecture and Design	8
5.	Modules and Functionality	11
6.	Coding and Testing	14
7.	Conclusion	20
8.	References	21

List of Figures

Serial No	Figure Name	Page No
1.	Architecture Diagram	8
2.	Use Case Diagram	10

List of Tables

Serial No	Table Name	Page No
1.	Literature Survey 1 - Algorithm for Efficient Seating Plan for Centralized Exam System	5
2.	Literature Survey 2 - Web Application based Exam Hall Seating Management System	6
3.	Literature Survey 3 - Automatic Exam Seating & Teacher Duty Allocation System	7

Abbreviations

- **DB** – Database

ABSTRACT

A DB Management System must be designed for Admission test and Seating allocation data. The Student Data Table has a Schema which comprises NAME, EXAM TYPE, CENTRE NO, EXAM CITY, CENTRE ADDRESS, LAB NO, TEST DATE, TEST SHIFT, TEST ID, TEST PASSWORD, SYSTEM NO, EXAM REFEREE NAME.

The Student Enters the TEST ID in the monitor at the centre, where he could identify his SYSTEM NO so that he can reach the allotted lab and attempt his test in the allocated system and slot.

The SYSTEM NO alone is generated independently and stored in the Data tables of the, which could be accessed by the student at the time of appearing for the exam. Whereas the rest of details and credentials are circulated by the College management to the students prior to the exam.

Different tables are generated for different shifts so that it could lead to the least or no confusions in conduction of the tests without any panic and incoherence of coordination.

INTRODUCTION

PROBLEM STATEMENT

SRM has decided to conduct SRMJEE examination for the 2023-2027 intake batch for all the courses in B.Tech, B.Des, M.B.A, Law. The exam cell has decided to conduct the tests in a shift system across different cities in India, for the convenience of students. The Management has approached a testing agency for conducting the exams throughout different cities and managing the test system allocation data of the students who enrolled for the exams.

SCOPE OF PROJECT

Need

Client needs to conduct an exam for his institute where the student accessing the system at the centre must be able to discover his system number and exam referee name.

Scope

The Data that must be displayed to the student is his name, system allocated to him, name of exam referee.

The student data is stored in the DB in the forms of table. The tables are made for different days and different slots for each centre where the test is happening.

The College Management send the raw data obtained from the student applications to the DB admin of the testing agency, to sort the data and structure it properly into tables to avoid sophisticated view of data.

The student will be gaining access from the college management to access his credentials and for knowing the details about exam, venue etc...

OBJECTIVES

1. To Store the raw data obtained from the college into tables.
2. Name the tables based on the date, shift, centre basis.
3. Fix the threshold count of systems based on the availability of the functioning systems in that specific lab.
4. Allocate the systems to the students based on the capacity of the lab.
5. Allow the Exam referee to access the complete details of the student in order to provide
support to the Student in the exam if needed.
6. Student can access the system at the entrance of lab to identify his system no and the name of his exam referee.

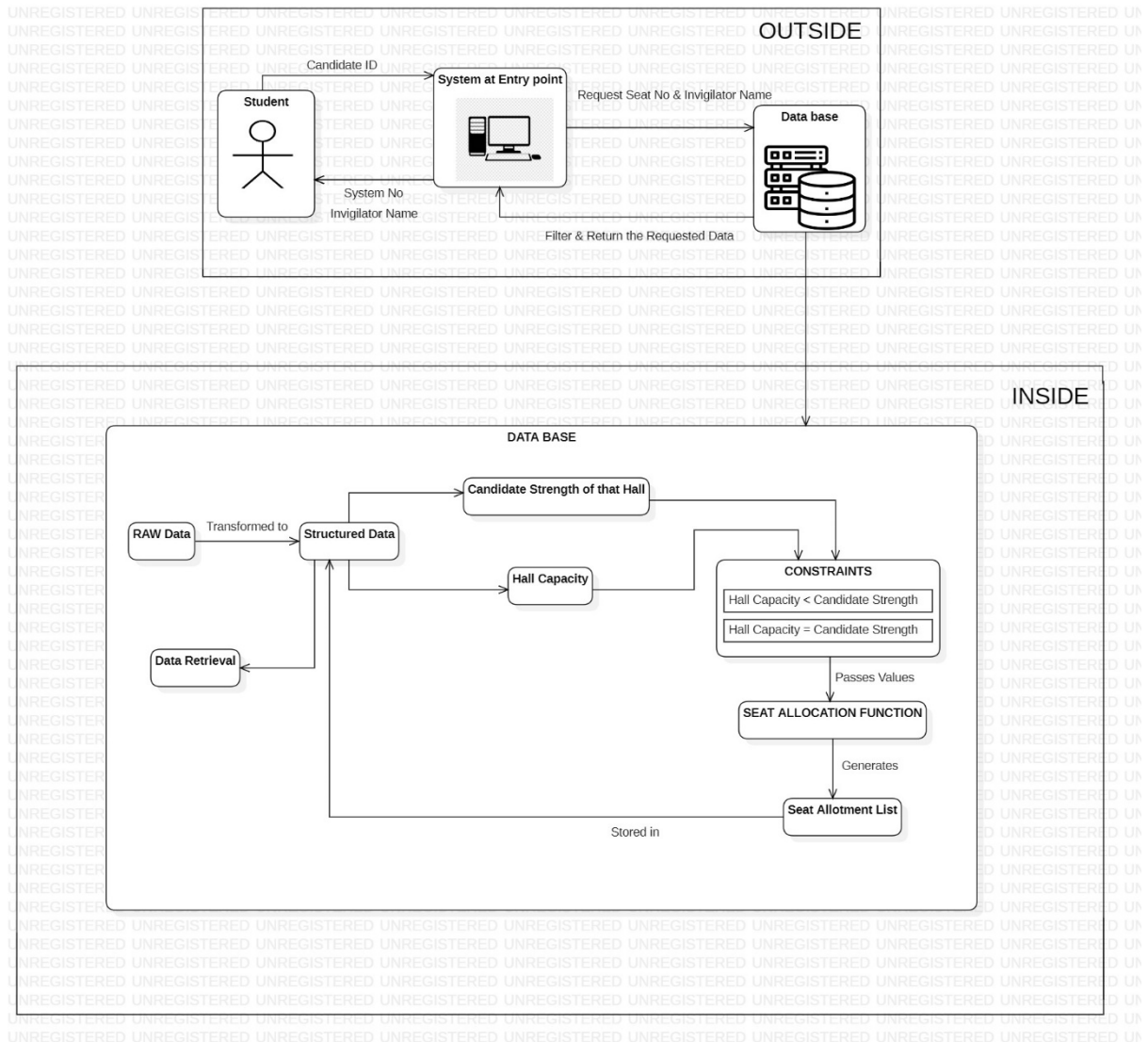
LITERATURE SURVEY

Research Name	Algorithm For Efficient Seating Plan for Centralized Exam System
Problem Identified	Seat allotted when there is huge strength
Objective	Our proposed system is applicable for I shape seat allocation system. The system needs two types of data as input, Room information and Exam information. The total number of students must have to be less than or equal to the total seats
Novelty/Significance	This will reduce a huge number of workloads that have to be given by the employees before exam to prepare an exam seating arrangement plan
Experimental setup and survey	Here we have to take description of specified room, no. of seats and room allocation

Research Name	Web Application based Exam Hall Seating Management System
Problem Identified	To manage the huge analysis of student
Objective	The main motive of growing this seating association gadget is to offer a manner to allocate exam hall for every scholar without any clash. Guide Excel sheet and office work are automated. For arranging hall allocation for students, we are mainly using the department and registration number of students
Novelty/Significance	A web-primarily based interface for displaying hall details and examination details for student was evolved, which makes students to peer their respective corridor effortlessly. They don't need to login. Students can get their exam and hall details by using branch and roll number only. Therefore, they can easily get the information's and it will reduce the time delay
Experimental setup and survey	Here we have to take description of specified room,no. of seats and room allocation

Research Name	Automatic Exam Seating & Teacher Duty Allocation System
Problem Identified	A time-consuming task for manually allocate classrooms, segregate the students according to their requirements,
Objective	It assigns the classrooms and the duties to the teachers in any institution. An input-output data is obtained from the real system which is found out manually by the organizers who set up the seating arrangement and chalk out the supervision duties.
Novelty/Significance	This software helps the Exam Coordinators to allocate the duties to the respective teachers and also to develop a student seating allocation plan for examinations. The project aims at allocating the duties with much greater effectiveness.

ARCHITECTURE DIAGRAM



Description:

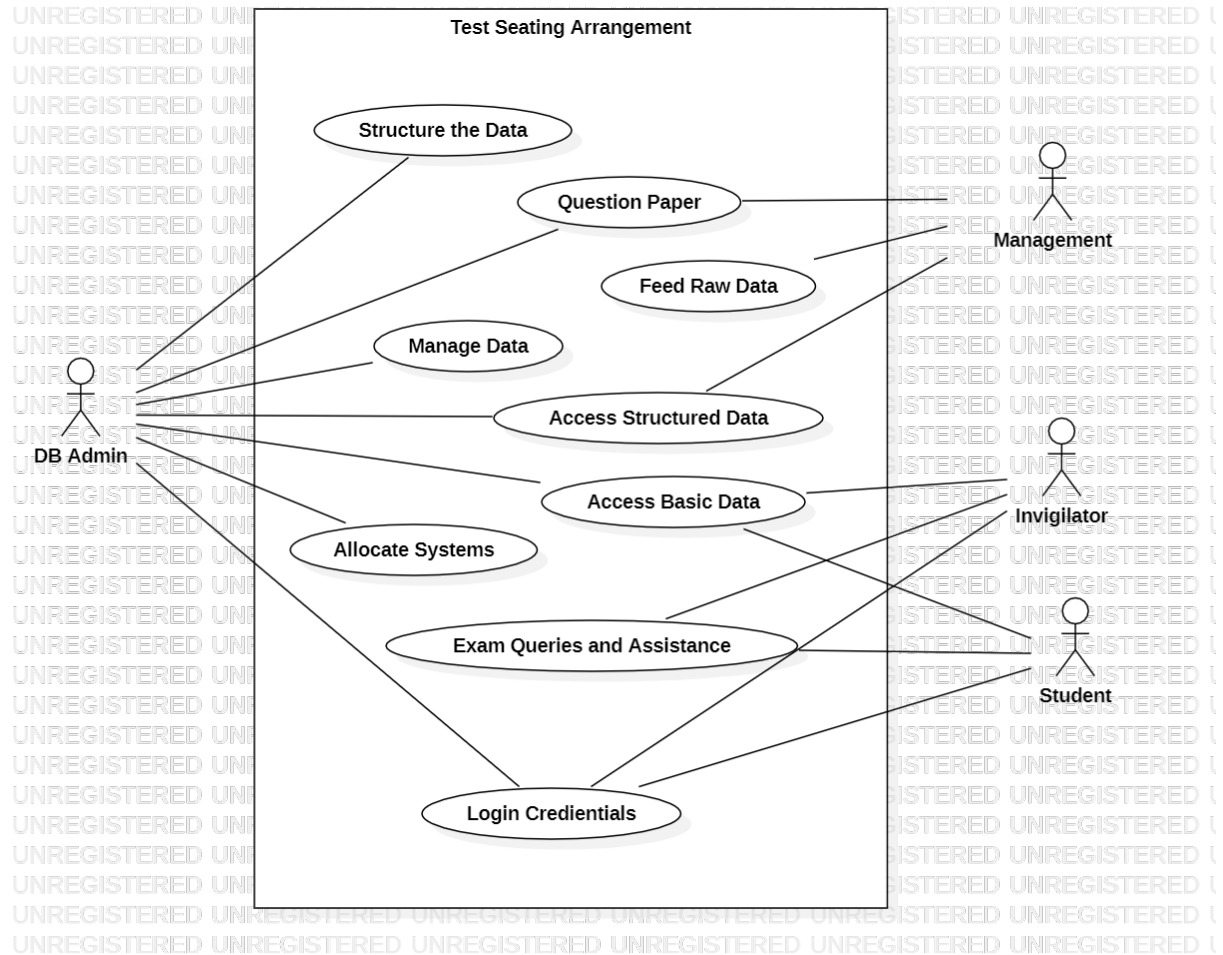
The student requests the Seat no and Invigilator name from the system which was placed at the entry of the exam centre, the DB reverts the data requested from the entry system.

Initially the Raw Data is converted into structured data, and it is allowed for the data retrieval at any point when needed.

The Allocation happens such that, the Students appearing and the capacity of the exam hall are identified from the DB, some necessary constraints are setup to allocate a unique seat number to the students without the clash of question papers.

If Student strength is less than or equal to the hall capacity allocation is allowed to be performed. If the above rule is violated the allocation comes to an end. When followed the entire data is entered along with the allocation into the assigned DB.

USE CASE DIAGRAM



MODULES AND FUNCTIONALITY

MODULES:

Tkinter:

Tkinter is a standard GUI (Graphical User Interface) toolkit for creating graphical desktop applications in Python. It is included in most Python installations, and is based on the Tcl/Tk GUI toolkit.

Tkinter provides a set of Python modules that allow developers to create GUI applications using a drag-and-drop interface. It supports a wide range of GUI widgets such as buttons, labels, text boxes, canvas, and menus.

To use Tkinter, you need to import the Tkinter module in your Python script. You can then create and customize GUI widgets and handle user events using event-driven programming.

Random:

The random module in Python is used to generate random numbers and data. It provides a suite of functions for working with pseudo-random number generators (PRNGs).

Note that the random module generates pseudo-random numbers, which means that the sequence of numbers generated can be predicted if the starting seed value is known.

You can set the seed value using the `seed()` function to generate a predictable sequence of random numbers, which can be useful for testing and debugging purposes.

MYSQL.CONNECTOR:

`mysql.connector` is a Python module that provides a standardized way to work with the MySQL DB from Python. It allows Python programs to interact with MySQL DBs using SQL queries.

`mysql.connector` provides many other features for working with MySQL DBs, including inserting, updating, and deleting data, as well as managing transactions and handling errors.

FUNCTIONALITY:

Initially all the required modules are installed into the working environment or IDE, the modules used in the functional design are `tkinter`, `random`, `mysql.connector`. These modules govern the functional design and the flow of the statement executions from the written code.

The Connection is established with the MYSQL DB server by using the `connector.connect()` function, and the graphical interfaces are created by using the `tkinter` module for the labels ID, Username, Password. The crucial function to insert the data into the DB is written as the part of this functional design.

The capacity and the user attributes are identified initially and random shuffling code fragment is written to generate the unique seat no and allocate the set of the question paper to the Student.

The obtained data is embedded into the insert statement and then a `connection.commit()` function is called to bind and store them into the specific table created in the DB.

Buttons for the insertion are created at the end of the functional design to commit and push the data into the storage tables in the DB, and then the buttons are assigned with `insert_data()` functionality to fulfil the purpose, and then a message box is called through tkinter to display the seat no which was allocated to the latest entry that was made a while ago.

Logging into the MYSQL workbench, allows us to cross verify and obtain the access to the entered data.

CODING AND TESTING

CODE:

```
import tkinter as tk
from tkinter import *
from tkinter import ttk
import mysql.connector
import random
from tkinter import messagebox

root = Tk()
root.title("TEST SEATING")

connection = mysql.connector.connect(host='localhost',
user='root', password='dbmsproject',port='3306',
DB='new_schema')
c = connection.cursor()

bkg = "#636e72"

frame = tk.Frame(root, bg=bkg)

label_firstname = tk.Label(frame, text="ID : ",
font=('verdana',12), bg=bkg)
entry_firstname = tk.Entry(frame, font=('verdana',12))

label_lastname = tk.Label(frame, text="USERNAME : ",
font=('verdana',12), bg=bkg)
entry_lastname = tk.Entry(frame, font=('verdana',12))

label_email = tk.Label(frame, text="PASSWORD : ",
font=('verdana',12), bg=bkg)
entry_email = tk.Entry(frame, font=('verdana',12))

def insertData():
    firstname = entry_firstname.get()
    lastname = entry_lastname.get()
    email = entry_email.get()
    labcapacity = 50

    a = 1
    x = [i for i in range(a, labcapacity+1)]
    random.shuffle(x)
```

```

    for i in range(0, 50):
        seat = x[i]
        if seat%2==0:
            Setno="SET-1"
        else:
            Setno="SET-2"

        insert_query = "INSERT INTO new (ID, Username, Password,
Sysno, Setno) VALUES(%s,%s,%s,%s,%s)"
        vals = (firstname, lastname, email, seat, Setno)
        c.execute(insert_query,vals)
        connection.commit()
        res = seat
        messagebox.showinfo("YOUR SEAT NO", seat)

button_insert = tk.Button(frame, text="Insert",
font=('verdana',14), bg='orange',command = insertData)

label_firstname.grid(row=0, column=0)
entry_firstname.grid(row=0, column=1, pady=10, padx=10)

label_lastname.grid(row=1, column=0)
entry_lastname.grid(row=1, column=1, pady=10, padx=10)

label_email.grid(row=2, column=0, sticky='e')
entry_email.grid(row=2, column=1, pady=10, padx=10)

button_insert.grid(row=4,column=0, columnspan=2, pady=10,
padx=10, sticky='nsew')

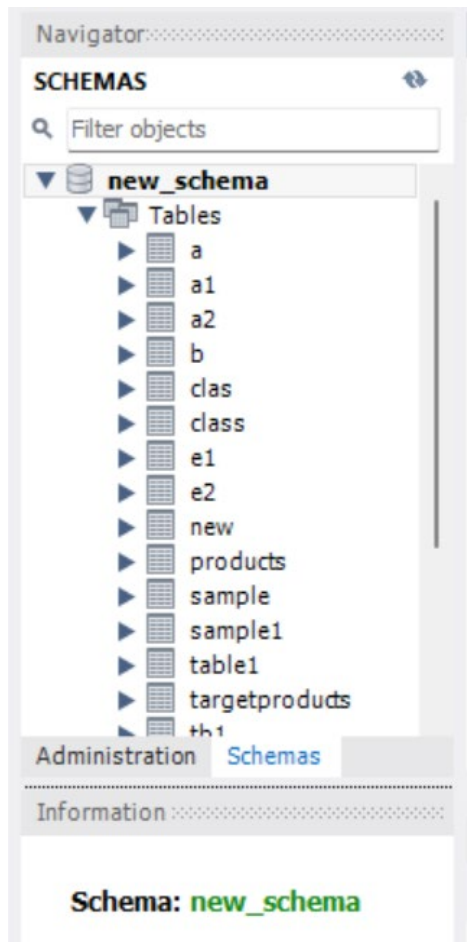
frame.grid(row=0, column=0)

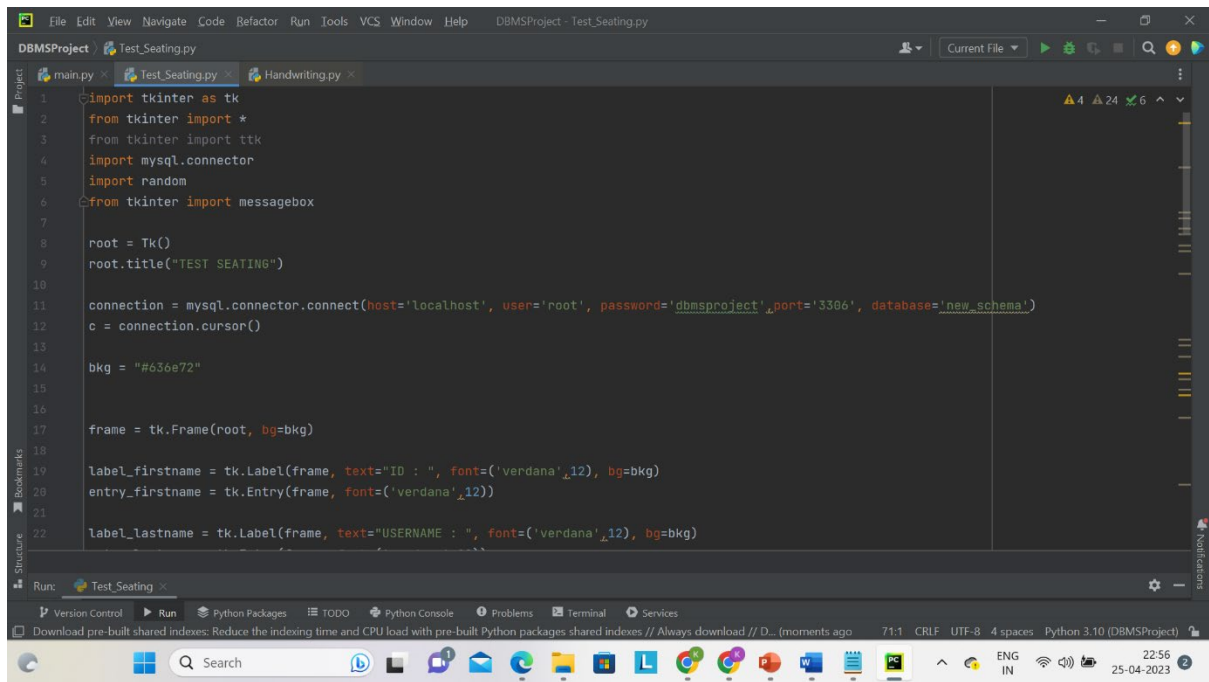
root.mainloop()

```

Name of the DB: new_schema

Name of the Table: new

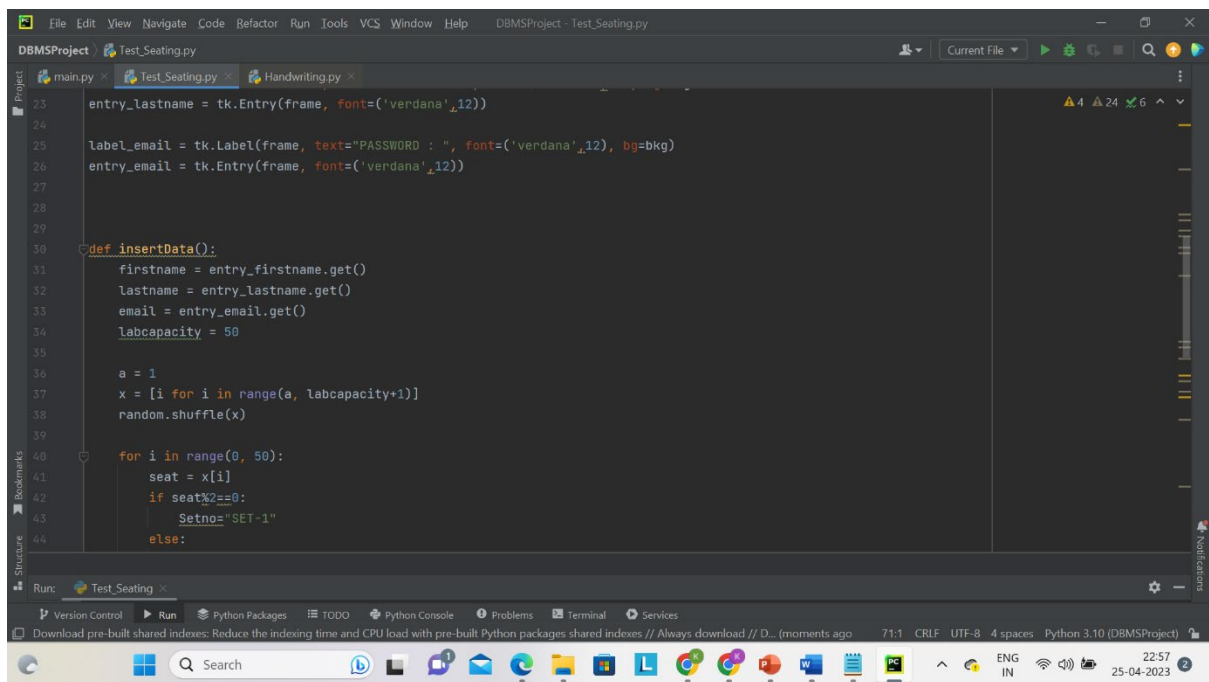




The screenshot shows an IDE window titled "DBMSProject - Test_Seating.py". The code in the editor is as follows:

```
1 import tkinter as tk
2 from tkinter import *
3 from tkinter import ttk
4 import mysql.connector
5 import random
6 from tkinter import messagebox
7
8 root = Tk()
9 root.title("TEST SEATING")
10
11 connection = mysql.connector.connect(host='localhost', user='root', password='dbmsproject', port='3306', database='new_schema')
12 c = connection.cursor()
13
14 bkg = "#636e72"
15
16
17 frame = tk.Frame(root, bg=bkg)
18
19 label_firstname = tk.Label(frame, text="ID : ", font=('verdana', 12), bg=bkg)
20 entry_firstname = tk.Entry(frame, font=('verdana', 12))
21
22 label_lastname = tk.Label(frame, text="USERNAME : ", font=('verdana', 12), bg=bkg)
```

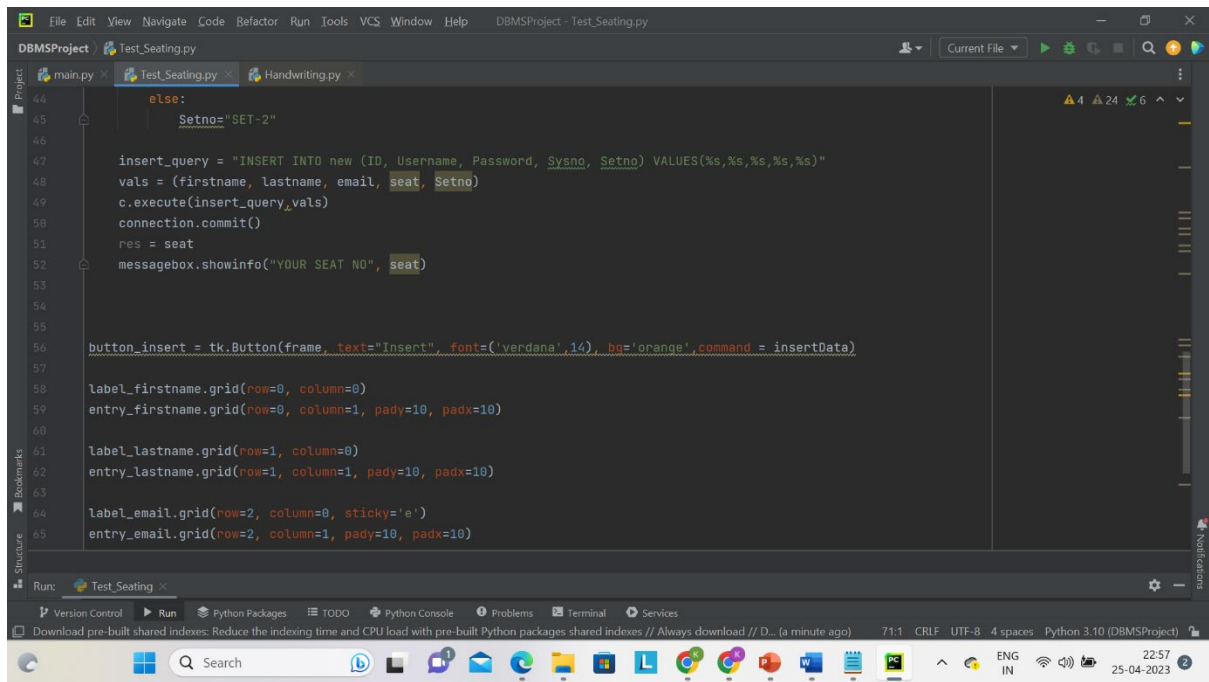
The IDE interface includes a menu bar (File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help), a toolbar with icons for file operations and running, and a status bar at the bottom showing settings like "71:1 CRLF UTF-8 4 spaces Python 3.10 (DBMSProject)".



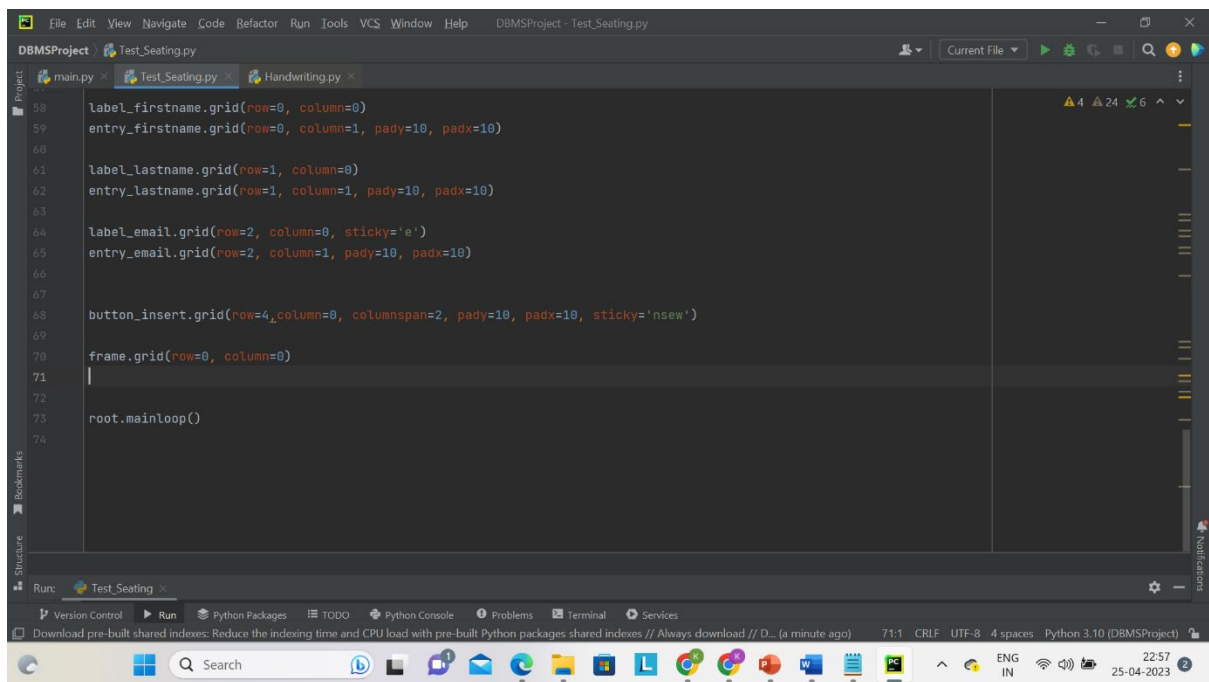
The screenshot shows the same IDE window, now displaying lines 23 through 44 of the script:

```
23 entry_lastname = tk.Entry(frame, font=('verdana', 12))
24
25 label_email = tk.Label(frame, text="PASSWORD : ", font=('verdana', 12), bg=bkg)
26 entry_email = tk.Entry(frame, font=('verdana', 12))
27
28
29
30 def insertData():
31     firstname = entry_firstname.get()
32     lastname = entry_lastname.get()
33     email = entry_email.get()
34     labcapacity = 50
35
36     a = 1
37     x = [i for i in range(a, labcapacity+1)]
38     random.shuffle(x)
39
40     for i in range(0, 50):
41         seat = x[i]
42         if seat%2==0:
43             Setno="SET-1"
44         else:
```

The IDE interface remains the same, with the status bar at the bottom showing "71:1 CRLF UTF-8 4 spaces Python 3.10 (DBMSProject)".

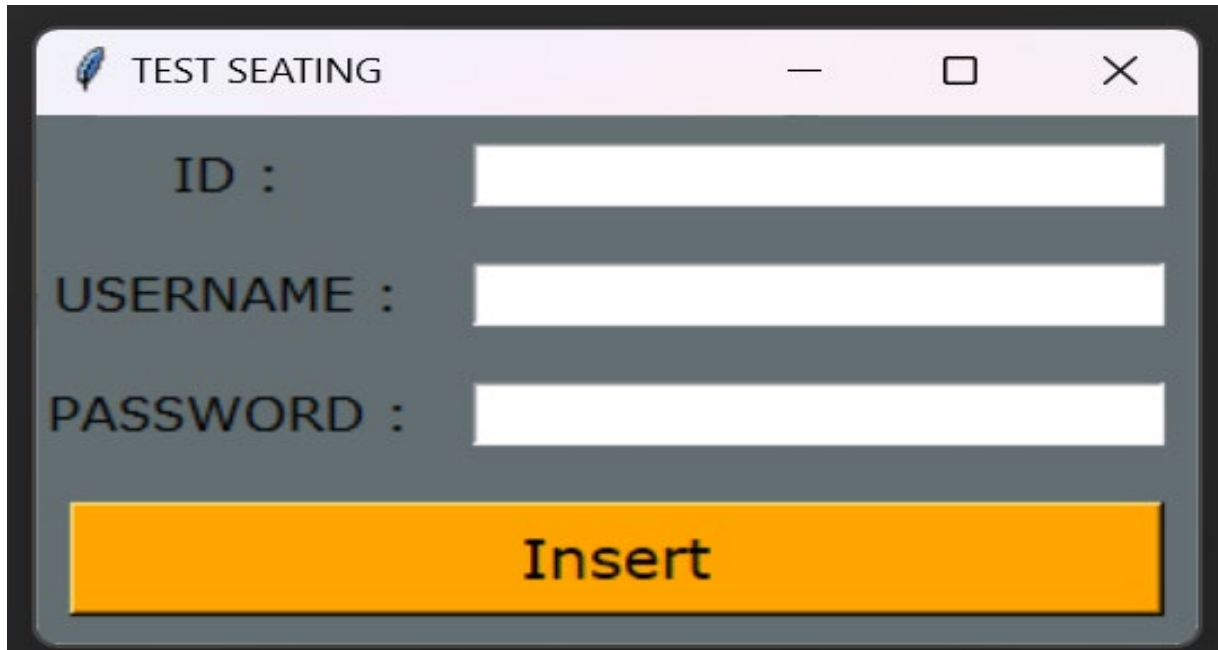


```
DBMSProject - Test_Seating.py
main.py x Test_Seating.py x Handwriting.py x
44 else:
45     Setno="SET-2"
46
47     insert_query = "INSERT INTO new (ID, Username, Password, Sysno, Setno) VALUES(%s,%s,%s,%s,%s)"
48     vals = (firstname, lastname, email, seat, Setno)
49     c.execute(insert_query,vals)
50     connection.commit()
51     res = seat
52     messagebox.showinfo("YOUR SEAT NO", seat)
53
54
55
56 button_insert = tk.Button(frame, text="Insert", font=('verdana',14), bg='orange', command = insertData)
57
58 label_firstname.grid(row=0, column=0)
59 entry_firstname.grid(row=0, column=1, pady=10, padx=10)
60
61 label_lastname.grid(row=1, column=0)
62 entry_lastname.grid(row=1, column=1, pady=10, padx=10)
63
64 label_email.grid(row=2, column=0, sticky='e')
65 entry_email.grid(row=2, column=1, pady=10, padx=10)
```



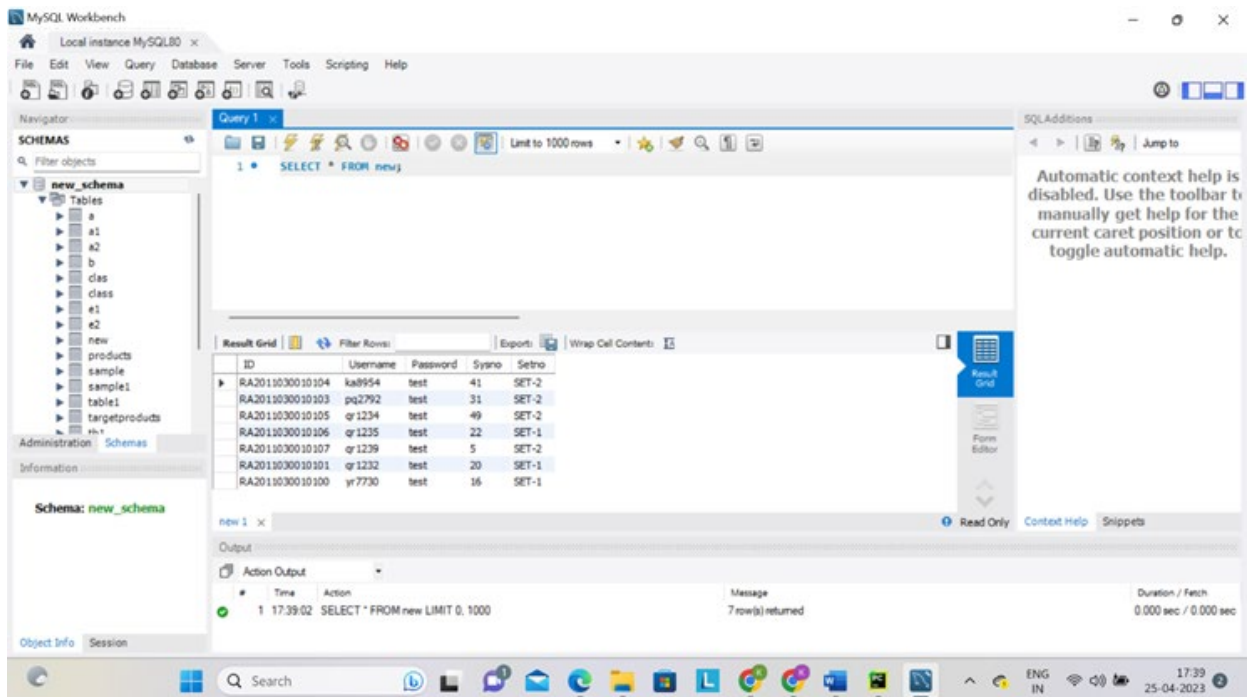
```
58 label_firstname.grid(row=0, column=0)
59 entry_firstname.grid(row=0, column=1, pady=10, padx=10)
60
61 label_lastname.grid(row=1, column=0)
62 entry_lastname.grid(row=1, column=1, pady=10, padx=10)
63
64 label_email.grid(row=2, column=0, sticky='e')
65 entry_email.grid(row=2, column=1, pady=10, padx=10)
66
67 button_insert.grid(row=4,column=0, columnspan=2, pady=10, padx=10, sticky='nsew')
68
69 frame.grid(row=0, column=0)
70 |
71
72
73 root.mainloop()
74
```


Frontend USER INTERFACE:



A screenshot of a web application window titled "TEST SEATING". The window has a light purple header bar with a feather icon on the left and standard window controls (minimize, maximize, close) on the right. The main content area has a dark gray background. It contains three white input fields stacked vertically, each preceded by a label: "ID :", "USERNAME :", and "PASSWORD :". Below these fields is a large, bright orange button with the word "Insert" in black text.

Backend Workforce in MYSQL:



A screenshot of the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar shows a "Navigator" pane with "SCHEMAS" expanded, listing various databases like "new_schema", "products", "sample", "table1", and "targetproducts". The main workspace shows a "Query 1" editor with the SQL statement: `SELECT * FROM new;`. Below the editor is the "Result Grid" showing 7 rows of data. The columns are ID, Username, Password, Syno, and Setno. The bottom pane shows the "Output" tab with a message: "1 17:39:02 SELECT * FROM new LIMIT 0, 1000" and "7 row(s) returned". The status bar at the bottom indicates the time is 17:39 on 25-04-2023.

ID	Username	Password	Syno	Setno
RA2011030010104	ka954	test	41	SET-2
RA2011030010103	pq2792	test	31	SET-2
RA2011030010105	qr1234	test	49	SET-2
RA2011030010106	qr1235	test	22	SET-1
RA2011030010107	qr1239	test	5	SET-2
RA2011030010101	qr1232	test	20	SET-1
RA2011030010100	yr7730	test	16	SET-1

CONCLUSION

In this project “Test Seating Allocation system” has been implemented and it can allocate the seat no to the Students appearing for the test and identifies the set of the question that the Student will face is also displayed. MYSQL is a most popular DB management system which was used to handle the data of the test seating arrangement. Python modules such as tkinter was used to build the GUI interface for user-friendly interaction and mysql.connector to interact eventually to perform operations on the data that is being obtained at a regular basis. Random module holds the event of allocating a unique system number without any duplicates to each Student. This project shows the capability and continuity portrayed by MYSQL DB and the python modules in obtaining the data and performing the user intended data management operations. This concludes that “Test Seating Allocation System” is capable of allocating unique seat number to the Students appearing for a specific test and also allocate the question paper from the existing sets based on the seating position of the Student, and also the integration between the MYSQL and Python could potentially bring many innovations in the upcoming days.

REFERENCES

- <https://ieeexplore.ieee.org/abstract/document/7514601/>
- <https://blog.coeo.com/importance-of-bringing-python-to-sql-server>
- <https://docs.databricks.com/dev-tools/python-sql-connector.html>
- <https://blog.devgenius.io/how-to-integrate-sql-server-with-python-and-using-sql-script-in-python-a20bcbea6def>
- <https://docs.snowflake.com/en/user-guide/python-connector-example.html>
- <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7514601>
- <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9792670>
- <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8473145>