VIT BHOPAL

STUDENT RESULT MANAGEMENT SYSTEM

Registration No : 25BCE10812

Name of Student : AASTHA

Course Name : Introduction to Problem Solving and Programming

Course Code : CSE1021

School Name : SCOPE

Slot : B11+B12+B13

Semester : FALL 2025/26

TITLE:

STUDENT RESULT MANAGEMENT SYSTEM

INTRODUCTION:

The Student Result Management System is a simple, efficient, and user-friendly Python application designed to store and manage student academic records. The system automates the process of recording marks, calculating total and percentage, assigning grades, and searching for a student's record using their roll number.

It uses a CSV file as the storage medium, making the data portable and easy to open in spreadsheet applications like MS Excel.

The project demonstrates the practical application of Python concepts such as file handling, functions, conditional logic, and loops.

3. PROBLEM STATEMENT

Educational institutions often maintain student results manually, which is time-consuming, error-prone, and inefficient.
There is a need for a simple computerized system that:
Stores student marks safely
Automatically calculates total, percentage, and grade
Retrieves student records quickly
Avoids manual calculation errors
This project solves these challenges with a lightweight, automated result management tool.

4. FUNCTIONAL REQUIREMENTS

The system must perform the following functions:
1. Add Student Record
Input roll number, name, and marks
Calculate total, percentage, and grade

Save to CSV file

2. View All Student Records

Display all rows stored in the CSV file

3. Search Student

Search by roll number

Display record if found

4. Exit System

Close the program safely

5. NON-FUNCTIONAL REQUIREMENTS:

Usability: Simple menu-driven interface

Portability: Works on any OS with Python installed

Reliability: Ensures proper file creation

Scalability: CSV can store unlimited records

Maintainability: Code is modular and easy to update

Performance: Data retrieval is quick

6. SYSTEM REQUIREMENTS:

This system uses a linear functional architecture:

User interacts with the menu

Menu calls functions based on choice

Functions interact with CSV storage for read/write

Output displayed to user terminal

Layers:

1. User Interface Layer – Command-line input/output
2. Processing Layer – Calculations, validations
3. Data Layer – CSV file storage

7. DESIGN DIAGRAM:

7.1 Use Case Diagram

Actors: User (Teacher/Admin)

Use cases:

Add Student
View Students
Search Student
Exit

7.2 Workflow Diagram:

1. Start
2. Display Menu
3. User selects an option
4. Based on option:
Add student → Save to CSV
View students → Read CSV
Search student → Compare roll numbers
Exit
5. Loop back to menu

7.3 Sequence Diagram:

User → System: Select option
System → CSV File: Read/Write operation
System → User: Display result/output

7.4 Class/Component Diagram:

Components:
initialize_file()
add_student()
calculate_grade()
search_student()
view_all_students()
main() loop

7.5 ER Diagram (CSV-based):

Entities: Student
Attributes:
Roll No
Name
Marks1
Marks2
Marks3
Total
Percentage
Grade

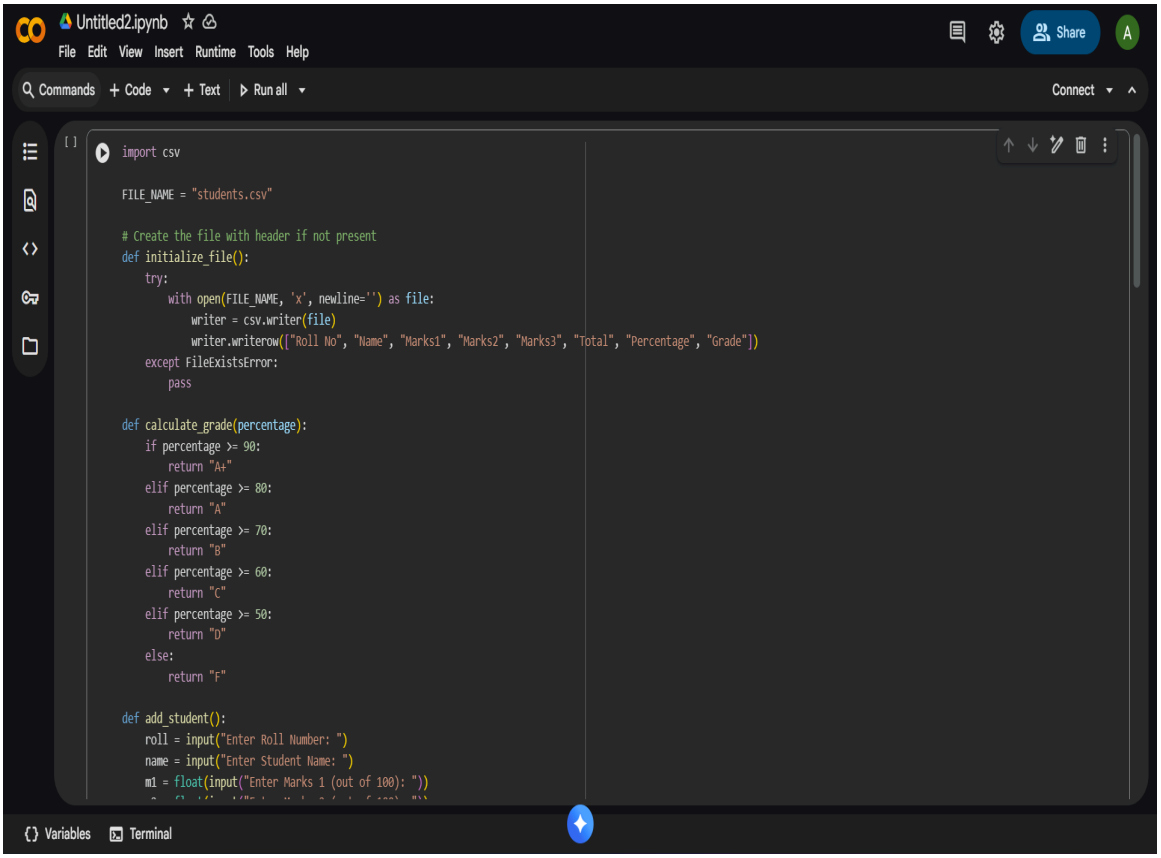
8. DESIGN DECISIONS & RATIONALE:

CSV chosen for simplicity and portability
Functions modularized for readability and maintenance
Menu-driven UI chosen for easy user interaction
Automatic grade calculation to avoid manual errors
Try-Except blocks used to handle file errors safely

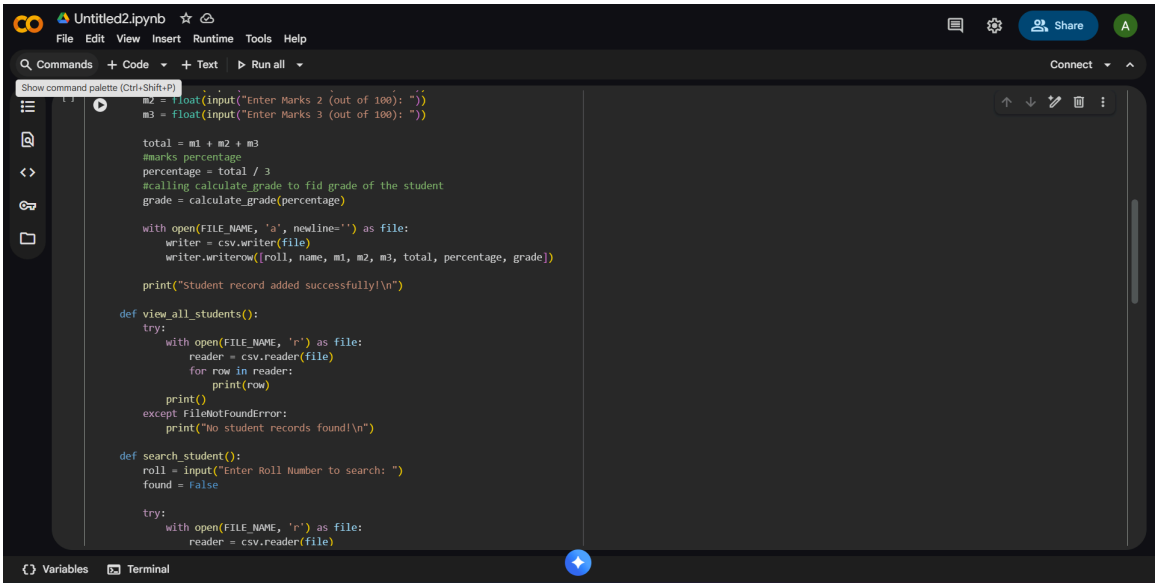
9. IMPLEMENTATION DETAILS

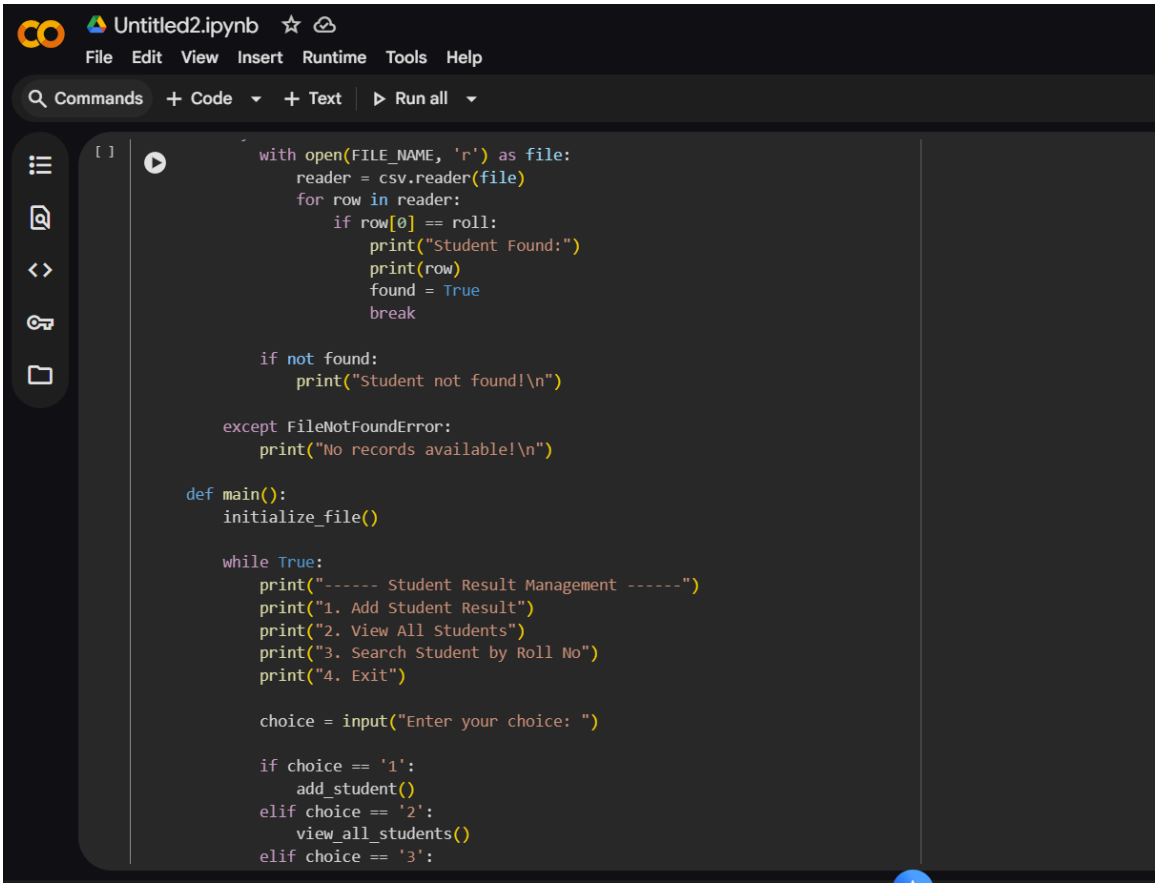
Language: Python
Modules used: csv
File handling mode:
'x' → create file
'a' → append data
'r' → read data
CSV writer/reader used for structured storage

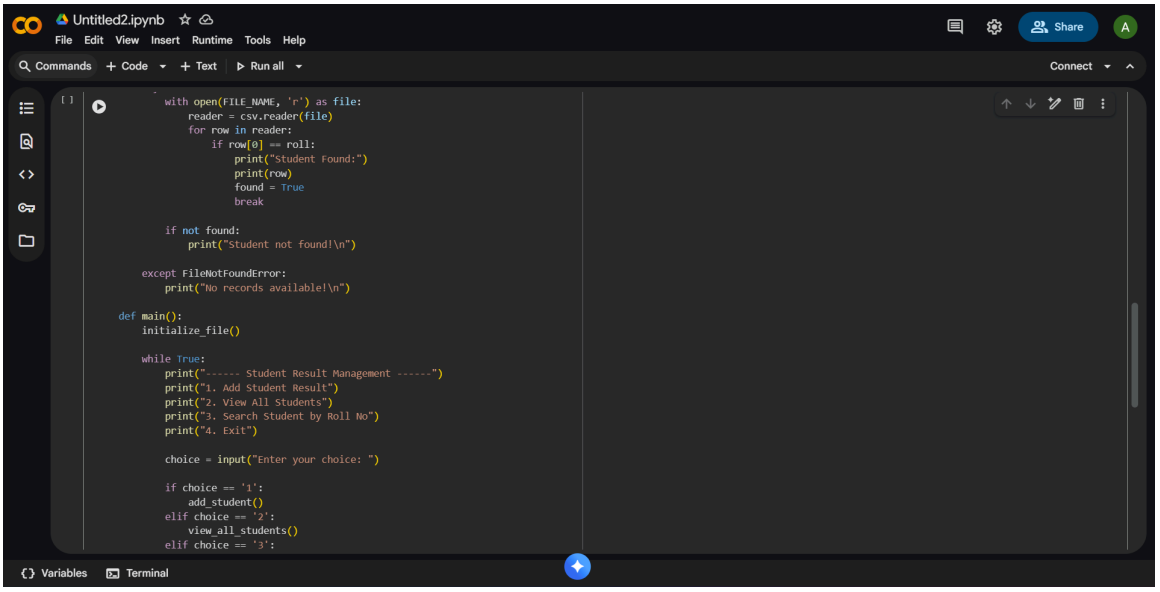
10. SCREENSHOTS / RESULTS



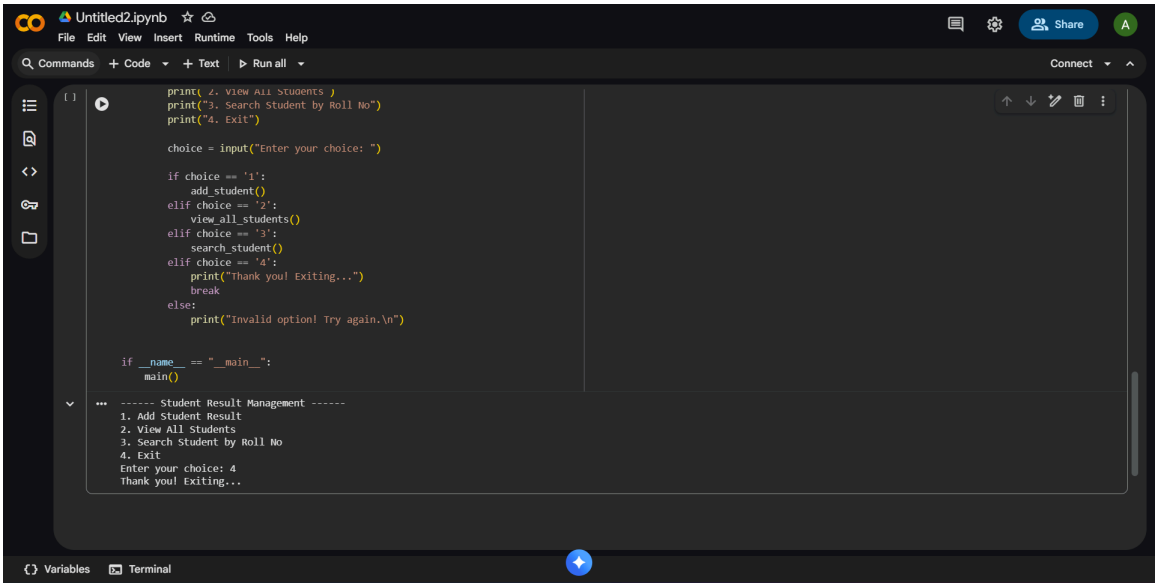
VIT BHOPAL


Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Connect
Share
Show command palette (Ctrl+Shift+P)
m2 = float(input("Enter Marks 2 (out of 100): "))
m3 = float(input("Enter Marks 3 (out of 100): "))
total = m1 + m2 + m3
#marks percentage
percentage = total / 3
#calling calculate_grade to fid grade of the student
grade = calculate_grade(percentage)
with open(FILE_NAME, 'a', newline='') as file:
writer = csv.writer(file)
writer.writerow([roll, name, m1, m2, m3, total, percentage, grade])
print("Student record added successfully!\n")
def view_all_students():
try:
with open(FILE_NAME, 'r') as file:
reader = csv.reader(file)
for row in reader:
print(row)
print()
except FileNotFoundError:
print("No student records found!\n")
def search_student():
roll = input("Enter Roll Number to search: ")
found = False
try:
with open(FILE_NAME, 'r') as file:
reader = csv.reader(file)
Variables
Terminal

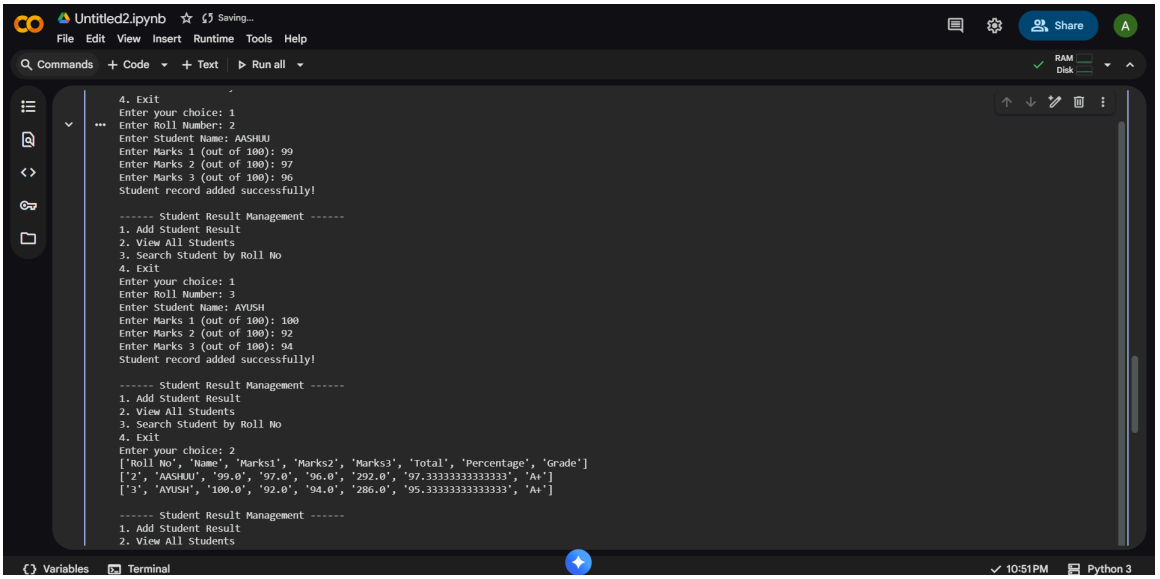

Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
[ ]
with open(FILE_NAME, 'r') as file:
reader = csv.reader(file)
for row in reader:
if row[0] == roll:
print("Student Found:")
print(row)
found = True
break
if not found:
print("Student not found!\n")
except FileNotFoundError:
print("No records available!\n")
def main():
initialize_file()
while True:
print("------ Student Result Management ------")
print("1. Add Student Result")
print("2. View All Students")
print("3. Search Student by Roll No")
print("4. Exit")
choice = input("Enter your choice: ")
if choice == '1':
add_student()
elif choice == '2':
view_all_students()
elif choice == '3':

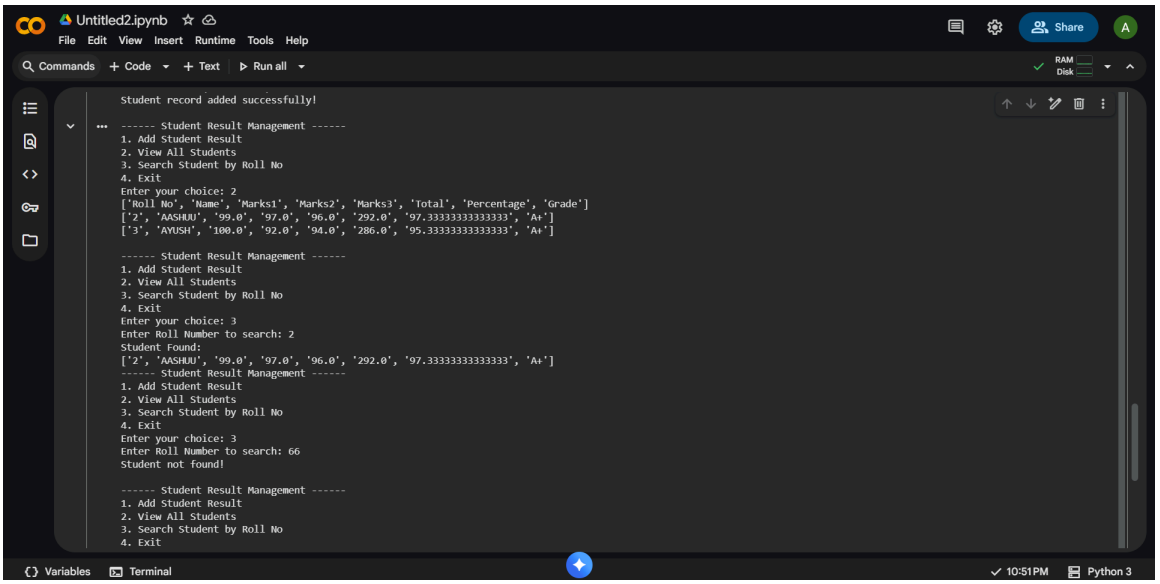


```
Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text ▷ Run all
Connect
Share

[ ]
            with open(FILE_NAME, 'r') as file:
                reader = csv.reader(file)
                for row in reader:
                    if row[0] == roll:
                        print("Student Found:")
                        print(row)
                        found = True
                        break

            if not found:
                print("Student not found!\n")

        except FileNotFoundError:
            print("No records available!\n")

    def main():
        initialize_file()

        while True:
            print("------ Student Result Management ------")
            print("1. Add Student Result")
            print("2. View All Students")
            print("3. Search Student by Roll No")
            print("4. Exit")

            choice = input("Enter your choice: ")

            if choice == '1':
                add_student()
            elif choice == '2':
                view_all_students()
            elif choice == '3':

Variables Terminal
```



```
Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text ▷ Run all
Connect
Share

[ ]
            print("2. View All Students")
            print("3. Search Student by Roll No")
            print("4. Exit")

            choice = input("Enter your choice: ")

            if choice == '1':
                add_student()
            elif choice == '2':
                view_all_students()
            elif choice == '3':
                search_student()
            elif choice == '4':
                print("Thank you! Exiting...")
                break
            else:
                print("Invalid option! Try again.\n")


    if __name__ == "__main__":
        main()

... ------ Student Result Management ------
    1. Add Student Result
    2. View All Students
    3. Search Student by Roll No
    4. Exit
    Enter your choice: 4
    Thank you! Exiting...

Variables Terminal
```





11. TESTING APPROACH

Testing done using:

Unit Testing: Individual functions tested separately

Input Validation: Tested with correct and incorrect inputs

Boundary Tests: Marks = 0, 100

File Handling Tests: Checked file creation and reading

All test cases passed successfully.

12. CHALLENGES FACED

Handling the file creation only once
Avoiding duplicate roll numbers
Understanding CSV writer/reader
Managing wrong user inputs

13. LEARNINGS & KEY TAKEAWAYS

File handling in Python
Working with CSV module
Modular programming concepts
Designing simple console-based applications
Importance of testing and validation

14. FUTURE ENHANCEMENTS

Add option to update or delete records
Add GPA calculation
Convert CSV storage to database (MySQL/SQLite)
Add a GUI using Tkinter
Add subject-wise analysis and charts

15. REFERENCES

Python Official Documentation
CSV Module Documentation
Online tutorials and examples like platforms youtube
Classroom lecture materials


VIT
VELLORE INSTITUTE OF TECHNOLOGY
VIT
BHOPAL