# Mini Project : STUDENT DATABASE

## Objective:

Your app should be used for all elementary work of a database application such as Input Records, deletion of records, Display records, sort records, save , search record etc..

## User Interface:

The application when executed, should display a menu as given below.

$ ./dataBase

----------*MENU*----------

a/A : Add a new student record
p/P : Print the records from database
s/S : Save the database in a file
d/D : Delete a record
e/E : Edit a record ( **display sub-menu** )
**1) For name edit 2) For DOB 3) For percentage**
**4) For contact 6) For email id**
f/F : Find a student
r/R : reverse the records of current display(No changes in database file)
q/Q : Quit from app

---

## Requirements:

- ➔ Each student record should contain StudentId, Student Name, D.O.B, marks(percent in Matriculation, Inter/diploma, graduation), contact number , email id.
  - If student Id is **V24HE5A1**, It indicates: **Vector2024HyderabadEmbedded5AkashSeq1.**
- ➔ Make sure that , duplicates IDs should not be store in DataBase.

## DELIVERABLES:

1. This app should contain user-defined functions for each and every task.
   **Ex:** add_record() , find_record() , etc....
2. Use makefile and make tool to manage the project.
3. Use **readme.txt** to explain the usage of the project, how to compile, execute etc..
4. If we **re-launch** the app , old / previows data should be available.
5. Deliver the project, in a folder(**named your ID**), containing all source files, headerfiles, makefile, and readme.txt.

## PROJECT approach :

1. USE of structure pointer with dynamic memory allocation.
2. FileHandling : File based functions like **save(), syncfile()**, should use to store data in file and to get old data.
3. For every function , separate file should be implemented.

---

## Project Version1

1. Use structure pointer.
2. DMA require to store records info.
3. No file handling functions.

## Project Version2

1. Use structure pointer and implement by using SLL.
2. File handling functions require to store data.

## SUGGESTION:

A) Use typedef, enum, union where-ever applicable.
B) Use separate header file to keep all structure, union , and typedefs.