# Simple Student Management System

MY PROFILE
NAURAN S.

MASTERING EMBEDDED SYSTEM ONLINE DIPLOMA
WWW.LEARN-IN-DEPTH.COM

# 1. Project Overview

*The Simple Student Management System is a C-based application designed to manage student records using a queue data structure. It allows users to add, search, delete, and update student information effectively. The system is structured to handle student data such as ID, name, GPA, and courses enrolled.*

*This report explains the project's overall structure, functionality, and the purpose of each function implemented in the code.*

# 2. Project Structure

*The project is divided into multiple files, each serving specific roles:*

- ***main.c**: This file contains the entry point of the program and initializes the system.*
- ***Queue.c**: Implements the queue data structure functionalities for managing student data.*
- ***Queue.h**: Contains the declarations for the queue operations and the student data structure.*
- ***System.c**: Implements the system management functionalities such as user options and handling student records.*
- ***System.h**: Contains declarations for system functions.*
- ***File_System.c**: file implements the functionalities declared in the header file. It contains two primary functions for file handling.*
- ***File_System.h**: file serves as the interface for the file system module, defining the necessary types and function prototypes for file operations.*

# 3. Detailed Function Descriptions

## main.c

- ***int main():***

  *Calls System_INIT to initialize the queue with a specified length (50).*

  *Invokes System_Options to present the user with available operations.*

## *Queue.c*

- ***Queue_State_t Queue_INIT(Queue_t* Queue, elementType* buffer, uint32_t length)****:*
  - *Initializes the queue structure, setting the head, tail, base, count, and length.*
  - *Returns an error state if the buffer is null.*
- ***Queue_State_t Queue_Enqueue(Queue_t* Queue, elementType item)****:*
  - *Adds a new student record to the end of the queue.*
  - *Checks if the queue is full before adding the item.*
- ***Queue_State_t Queue_Dequeue(Queue_t* Queue, elementType* item)****:*
  - *Removes a student record from the front of the queue and stores it in item.*
  - *Ensures the queue is not empty before performing the operation.*
- ***Queue_State_t Queue_ISFull(Queue_t* Queue)****:*
  - *Checks if the queue is full based on the current count and length.*
- ***Queue_State_t Queue_ISEmpty(Queue_t* Queue)****:*
  - *Checks if the queue is empty by verifying the count.*

- ***void Queue_Print(Queue_t* Queue)****:*
  - *Displays all student records in the queue.*

    - *Iterates through the queue and prints each student's details.*
- ***void Queue_Print_Item(elementType* item)**:*
    - *Prints the details of a specific student record.*
- ***elementType Fill_Data()**:*
    - *Collects data for a new student from user input and returns it as an elementType.*
- ***Queue_State_t Queue_FindByID(Queue_t* Queue, int id)**:*
    - *Searches for a student by their ID and prints the record if found.*
- ***Queue_State_t Queue_FindByName(Queue_t* Queue, char* name)**:*
    - *Searches for students by first or last name and prints records if found.*
- ***Queue_State_t Queue_FindByCourse(Queue_t* Queue, int courseID)**:*
    - *Finds students enrolled in a specific course and prints their records.*
- ***Queue_State_t Queue_DeleteByID(Queue_t* Queue, int id)**:*
    - *Deletes a student record from the queue by their ID.*
- ***Queue_State_t Queue_UpdateByID(Queue_t* Queue, int id)**:*
    - *Updates the data of a student found by ID.*

## *System.c*

- ***System_State_t System_INIT(int length)**:*
    - *Initializes the student management system and the queue with a specified length.*
    - *Checks for length constraints against a maximum value.*
- ***void System_Options()**:*

- *Displays available options for user interactions and captures the user's choice.*
- *Calls System_Call with the selected option.*

- ***void System_Call(int option)**:*
  - *Handles user-selected operations such as adding students, searching, deleting, and updating records.*
  - *Implements logic based on user input.*
- ***void System_Update(elementType* selected)**:*
  - *Updates the details of a selected student record.*
- ***void System_HandleFileData(char *buffer, elementType* data)**:*
  - *Processes data read from a file and populates a student record.*
- ***System_State_t System_IsIDUnique(Queue_t* Queue, int ID)**:*
  - *Checks for uniqueness of student IDs within the queue before adding new records.*

## *File_System.c*

- ***File_State_t File_Create(char *filename)**:*
  - *This function attempts to create a new file with the specified filename. If the file already exists or cannot be created due to permission issues, it will return an error.*
- `File_State_t` **`File_Read`**`(`**`char`** `*filename,` **`char`**`* buffer)`:
  - *This function attempts to read the first line of a specified file into the provided buffer. If the file does not exist, it calls File_Create to create it. It handles errors related to file opening and reading.*

## 4. Some Screenshots: *github*

```
************************************************************
 0: Exit
 1: Add From File
 2: Add Manually
 3: Find By ID
 4: Find By Name
 5: Find By Course ID
 6: Number Of Students
 7: Delete 1st Student
 8: Delete All
 9: Update Student Data
10: Display Student Data
Option: 1
************************************************************
               *** Add From File (only one student at a time) .. ***
        Input Format:ID,FirstName,LastName,GPA,course_1,course_2,course_3,course_4,course_5
        Note: courseID = 0 (null)

        File Name: data.txt
```

```
************************************************************************
                 *** Add Manually (only one student at a time) .. ***
        ID: 200
        First Name: Nouran
        Last Name: S.
        GPA: 3.7
        No. of Courses: 2
                Courses No.1: 1
                Courses No.2: 1

************************************************************************

        ** Duplicate course ID found **
************************************************************************
```

```
************************************************************************
                 *** Find By ID .. ***
        ID: 200
        ********** Found A Student Has The Same ID **********
                ID: 200
                First Name: Nouran
                Last Name:
                GPA: 3.70
                Courses: 3
        ********************************
************************************************************************
```

```
**************************************************************
                *** Find By Name .. ***
        Name: Nouran
        *********** Found A Student Has The Same Name ***********
                ID: 100
                First Name: Nouran
                Last Name: Luna
                GPA: 3.10
                Courses: 1      2       30
        *********************************
        *********** Found A Student Has The Same Name ***********
                ID: 200
                First Name: Nouran
                Last Name:
                GPA: 3.70
                Courses: 3
        *********************************
**************************************************************
```

```
**************************************************************
                *** Find By Course ID .. ***
       Course ID: 2
       *********** Found A Student Enrolls In The Same Course ID ***********
                ID: 100
                First Name: Nouran
                Last Name: Luna
                GPA: 3.10
                Courses: 1      2       30
       *********************************
**************************************************************
```

```
 0: Exit
 1: Add From File
 2: Add Manually
 3: Find By ID
 4: Find By Name
 5: Find By Course ID
 6: Number Of Students
 7: Delete 1st Student
 8: Delete All
 9: Update Student Data
10: Display Student Data
Option: 6
**************************************************************
                *** Number Of Students: 2 .. ***
**************************************************************
```