**Explanation: Student Management System GUI**

**How GTK Works in C (Quick Primer)**

GTK (**GIMP Toolkit**) is a C library used to create GUI applications.  
The general structure of a GTK C program is:

1. **Initialize GTK**
2. gtk\_init(&argc, &argv);
3. **Create Widgets (UI Elements)**  
   Examples:
   * GtkWidget \*window = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);
   * GtkWidget \*button = gtk\_button\_new\_with\_label("Click Me");
4. **Pack Widgets into Containers**  
   Example:
5. gtk\_container\_add(GTK\_CONTAINER(window), button);
6. **Connect Signals (Events)**  
   Example:
7. g\_signal\_connect(button, "clicked", G\_CALLBACK(on\_button\_click), NULL);
8. **Show Widgets & Enter Main Loop**
9. gtk\_widget\_show\_all(window);
10. gtk\_main();

GTK follows an **event-driven model** — your functions run when signals (like button clicks) occur.

**Structure of Your Student Management System Code**

Your program is basically a **menu-based Student Record Manager** wrapped inside a GTK GUI.  
It consists of:

* **Data structure**: Holds student info
* **File handling functions**: To save/load records
* **GUI functions**: To display menus and forms
* **Event handlers**: Functions that respond to button clicks

**Main Components of the Code**

**A. Data Structure**

typedef struct {

int roll;

char name[50];

float marks;

} Student;

* **roll** → Roll number (unique ID)
* **name** → Student’s name
* **marks** → Marks obtained

**B. File Handling**

Your records are stored in **student.dat** as binary data.

Functions:

* **add\_student()** → Opens a form in the GUI, takes input, writes to file.
* **display\_students()** → Reads from file and shows all records in a GTK text view.
* **search\_student()** → Reads from file, matches roll number, shows details.
* **sort\_students()** → Reads into array, sorts (by roll or name), displays.
* **update\_student()** → Finds a record, edits it, writes back.
* **delete\_student()** → Removes a record from the file.

**C. GUI Setup (Main Menu Window)**

GtkWidget \*window = gtk\_window\_new(GTK\_WINDOW\_TOPLEVEL);

gtk\_window\_set\_title(GTK\_WINDOW(window), "Student Management System");

gtk\_window\_set\_default\_size(GTK\_WINDOW(window), 400, 300);

gtk\_window\_set\_position(GTK\_WINDOW(window), GTK\_WIN\_POS\_CENTER);

* Creates a **main application window**.
* Sets title, size, and position.

**D. Layout (VBox + Buttons)**

GtkWidget \*vbox = gtk\_vbox\_new(FALSE, 5);

gtk\_container\_add(GTK\_CONTAINER(window), vbox);

* A **VBox** arranges buttons vertically.
* Buttons for:
  + Add Student
  + Display Students
  + Search
  + Sort
  + Update
  + Delete
  + Statistics
  + Quit

**E. Button Event Binding**

GtkWidget \*add\_btn = gtk\_button\_new\_with\_label("Add Student");

g\_signal\_connect(add\_btn, "clicked", G\_CALLBACK(add\_student\_gui), NULL);

gtk\_box\_pack\_start(GTK\_BOX(vbox), add\_btn, FALSE, FALSE, 0);

* Creates **"Add Student"** button.
* Connects "clicked" event to add\_student\_gui() function.

**F. Example: Add Student GUI**

void add\_student\_gui(GtkWidget \*widget, gpointer data) {

GtkWidget \*dialog, \*content\_area;

GtkWidget \*roll\_entry, \*name\_entry, \*marks\_entry;

dialog = gtk\_dialog\_new\_with\_buttons("Add Student",

GTK\_WINDOW(widget),

GTK\_DIALOG\_MODAL,

"Save", GTK\_RESPONSE\_ACCEPT,

"Cancel", GTK\_RESPONSE\_REJECT,

NULL);

content\_area = gtk\_dialog\_get\_content\_area(GTK\_DIALOG(dialog));

roll\_entry = gtk\_entry\_new();

gtk\_entry\_set\_placeholder\_text(GTK\_ENTRY(roll\_entry), "Enter Roll No");

name\_entry = gtk\_entry\_new();

gtk\_entry\_set\_placeholder\_text(GTK\_ENTRY(name\_entry), "Enter Name");

marks\_entry = gtk\_entry\_new();

gtk\_entry\_set\_placeholder\_text(GTK\_ENTRY(marks\_entry), "Enter Marks");

gtk\_box\_pack\_start(GTK\_BOX(content\_area), roll\_entry, FALSE, FALSE, 5);

gtk\_box\_pack\_start(GTK\_BOX(content\_area), name\_entry, FALSE, FALSE, 5);

gtk\_box\_pack\_start(GTK\_BOX(content\_area), marks\_entry, FALSE, FALSE, 5);

gtk\_widget\_show\_all(dialog);

if (gtk\_dialog\_run(GTK\_DIALOG(dialog)) == GTK\_RESPONSE\_ACCEPT) {

Student s;

s.roll = atoi(gtk\_entry\_get\_text(GTK\_ENTRY(roll\_entry)));

strcpy(s.name, gtk\_entry\_get\_text(GTK\_ENTRY(name\_entry)));

s.marks = atof(gtk\_entry\_get\_text(GTK\_ENTRY(marks\_entry)));

FILE \*fp = fopen("student.dat", "ab");

fwrite(&s, sizeof(Student), 1, fp);

fclose(fp);

}

gtk\_widget\_destroy(dialog);

}

This function:

1. Opens a **dialog box** with entry fields.
2. Gets roll number, name, marks from user.
3. Saves the record in student.dat.

**G. Display Students**

* Reads all students from file.
* Shows them in a **GTK Text View** widget.
* Example usage of GtkTextBuffer to insert text dynamically.

**H. Program Flow**

1. main() creates **main window**.
2. Creates **buttons** for each feature.
3. **User clicks a button** → Calls respective function.
4. **Function updates file** or **reads file** and shows data in GUI.

**Usage Flow (For End User)**

1. Run the program.
2. Main window appears with menu buttons.
3. Click:
   * **Add Student** → Enter details → Save.
   * **Display Students** → View all records.
   * **Search** → Enter roll number → Get result.
   * **Sort** → Choose sorting type → View sorted list.
   * **Update** → Change marks or name of a student.
   * **Delete** → Remove a student.
   * **Statistics** → See number of students and average marks.
4. Data is **saved persistently** in student.dat.