```c
#include <stdio.h>
#include <stdbool.h>
#include <string.h>

#define MAX_EMPLOYEES 100 // Maximum number of employees

// Define the Employee structure
struct Employee {
    int id;
    char name[50];
    char gender[20];
    char ethnicity[50];
    float salary;
};

//Samuel did this function----------------------------------------
// Add a new employee
void addNewEmployee(struct Employee* employees_value) {
    printf("Enter Employee Name: ");
    scanf("%s", employees_value->name);
    printf("Enter Employee ID: ");
    scanf("%d", &employees_value->id);
    printf("Enter Employee Gender: ");
    scanf("%s", employees_value->gender);
    printf("Enter Employee Ethnicity: ");
    scanf("%s", employees_value->ethnicity);
    printf("Enter Employee Salary: ");
    scanf("%f", &employees_value->salary);
}

//Samuel did this function-----------------------------------------------
// Delete an employee
void deleteEmployee(struct Employee employees[], int* num_employees) {
    int emp_id_del;
    bool delete_flag = true;

    printf("Enter the Employee ID of the employee you want to delete: ");
    scanf("%d", &emp_id_del);

    bool found = false;
    for (int i = 0; i < *num_employees; i++) {
        if (employees[i].id == emp_id_del) {
            // Shift all subsequent employees up by one position
            for (int j = i; j < *num_employees - 1; j++) {
                employees[j] = employees[j + 1];
            }
            (*num_employees)--; // Reduce the employee count
            printf("Employee with ID %d has been deleted.\n", emp_id_del);
            found = true;
            break;
        }
    }

    if (!found) {
        printf("Employee ID %d not found in the database.\n", emp_id_del);
    }
}

//Sehajdeep did this function-----------------------------------------
// Display employees
void displayEmployee(struct Employee* employees, int num_employee) {
    for (int i = 0; i < num_employee; i++) {
        printf("Name: %s\n", (employees + i)->name);
        printf("ID: %d\n", (employees + i)->id);
        printf("Gender: %s\n", (employees + i)->gender);
        printf("Ethnicity: %s\n", (employees + i)->ethnicity);
        printf("Salary: %.2f\n", (employees + i)->salary);
        printf("\n");
    }
}

//Sehajdeep did this function-----------------------------------------
// For searching employee by ID
void searchEmployeeByID(struct Employee* employees, int num_employee) {
    int user_input;
    printf("\nPlease enter employee ID: ");
    scanf("%d", &user_input);

    bool found = false;
    for (int i = 0; i < num_employee; i++) {
        if ((employees + i)->id == user_input) {
            printf("\nEmployee Found:\n");
            printf("Name: %s\n", (employees + i)->name);
            printf("ID: %d\n", (employees + i)->id);
            printf("Gender: %s\n", (employees + i)->gender);
            printf("Ethnicity: %s\n", (employees + i)->ethnicity);
```

```
            printf("Salary: %.2f\n", (employees + i)->salary);
            found = true;
            break;
        }
    }

    if (!found) {
        printf("\nEmployee with ID %d not found.\n", user_input);
    }
}

//Samuel did this function-----------------------------------------------------
//For calculating average salary
void calculateAverageSalary(struct Employee* employees, int num_employees) {
    char gender[20];
    char ethnicity[50];
    int user_choice;

    printf("\nCalculate Average Salary:\n");
    printf("1. Average Salary Based On Gender\n");
    printf("2. Average Salary Based By Ethnicity\n");
    printf("3. Average Salary Based On Both Gender and Ethnicity\n");
    printf("Please choose an option: ");
    scanf("%d", &user_choice);

    float total_salary = 0.0;
    int count = 0;

    if (user_choice == 1) { // To filter by Gender
        printf("Enter Gender to calculate: ");
        scanf("%s", gender);

        for (int i = 0; i < num_employees; i++) {
            if (strcmp((employees + i)->gender, gender) == 0) {
                total_salary += (employees + i)->salary;
                count++;
            }
        }
    } else if (user_choice == 2) {  //To filter by Ethnicity
        printf("Enter Ethnicity to calculate: ");
        scanf("%s", ethnicity);

        for (int i = 0; i < num_employees; i++) {
            if (strcmp((employees + i)->ethnicity, ethnicity) == 0) {
                total_salary += (employees + i)->salary;
                count++;
            }
        }
    } else if (user_choice == 3) { // Filter by Both Gender and Ethnicity
        printf("Enter Gender to calculate: ");
        scanf("%s", gender);
        printf("Enter Ethnicity to calculate: ");
        scanf("%s", ethnicity);

        for (int i = 0; i < num_employees; i++) {
            if (strcmp((employees + i)->gender, gender) == 0 &&
                strcmp((employees + i)->ethnicity, ethnicity) == 0) {
                total_salary += (employees + i)->salary;
                count++;
            }
        }
    } else {
        printf("Invalid option.\n");
        return;
    }

    if (count > 0) {
        printf("\nAverage Salary: %.2f\n", total_salary / count);
    } else {
        printf("\nNo employees match the given criteria.\n");
    }
}

//Sehajdeep did this function-------------------------------------------------
//for displaying comparision results depending on gender or ethnicity
void displayComparisonResults(struct Employee* employees, int num_employees) {
    char filter_gender[20];
    char filter_ethnicity[50];
    int user_choice;

    printf("\nDisplay Comparison Results:\n");
    printf("1. Compare Salaries by Gender\n");
    printf("2. Compare Salaries by Ethnicity\n");
    printf("Choose an option: ");
    scanf("%d", &user_choice);

    float total1 = 0.0, total2 = 0.0;
```

```c
    int count1 = 0, count2 = 0;

    if (user_choice == 1) { // To compare by Gender
        printf("Enter Gender to filter (e.g., Male, Female, Non-Binary): ");
        scanf("%s", filter_gender);

        for (int i = 0; i < num_employees; i++) {
            if (strcmp(employees[i].gender, filter_gender) == 0) {
                total1 += employees[i].salary;
                count1++;
            } else {
                total2 += employees[i].salary;
                count2++;
            }
        }

        printf("\nComparison Results by Gender:\n");
        if (count1 > 0) printf("Average Salary for %s: %.2f\n", filter_gender, total1 / count1);
        if (count2 > 0) printf("Average Salary for Other Genders: %.2f\n", total2 / count2);

    } else if (user_choice == 2) { // To compare by Ethnicity
        printf("Enter Ethnicity to filter (e.g., Asian, Black, Caucasian): ");
        scanf("%s", filter_ethnicity);

        for (int i = 0; i < num_employees; i++) {
            if (strcmp(employees[i].ethnicity, filter_ethnicity) == 0) {
                total1 += employees[i].salary;
                count1++;
            } else {
                total2 += employees[i].salary;
                count2++;
            }
        }

        printf("\nComparison Results by Ethnicity:\n");
        if (count1 > 0) printf("Average Salary for %s: %.2f\n", filter_ethnicity, total1 / count1);
        if (count2 > 0) printf("Average Salary for Other Ethnicities: %.2f\n", total2 / count2);

    } else {
        printf("Invalid option. Returning to the menu.\n");
    }
}


int main() {
    struct Employee employees[MAX_EMPLOYEES];
    int num_employees = 0;
    int menu;

    do {
        printf("\nMenu:\n");
        printf("1. Add a new employee\n");
        printf("2. Delete an employee\n");
        printf("3. Display employees\n");
        printf("4. Search employee by ID\n");
        printf("5. Calculate average salaries based on gender and/or ethnicity\n");
        printf("6. Display comparision results");
        printf("0. Exit\n");
        printf("Choose an option: ");
        scanf("%d", &menu);

    if (menu == 1) {
            // For adding a new employee
            if (num_employees < MAX_EMPLOYEES) {
                addNewEmployee(&employees[num_employees]);
                num_employees++;
            } else {
                printf("Employee list is full!\n");
            }
        } else if (menu == 2) {
            deleteEmployee(employees, &num_employees);
        } else if (menu == 3) {
            displayEmployee(employees, num_employees);
        } else if (menu == 4) {
            searchEmployeeByID(employees, num_employees);
        } else if (menu == 5) {
            calculateAverageSalary(employees, num_employees);
        } else if (menu == 6) {
            displayComparisonResults(employees, num_employees);
        } else if (menu != 0) {
            printf("Invalid option. Try again.\n");
        }
    } while (menu != 0);


    return 0;
}
```