Spatial Database for Item Management

Name: Saamer Masood

Student ID: 082240793

From: Automotive Engineering G30

Course Name: Computer Software II 2024

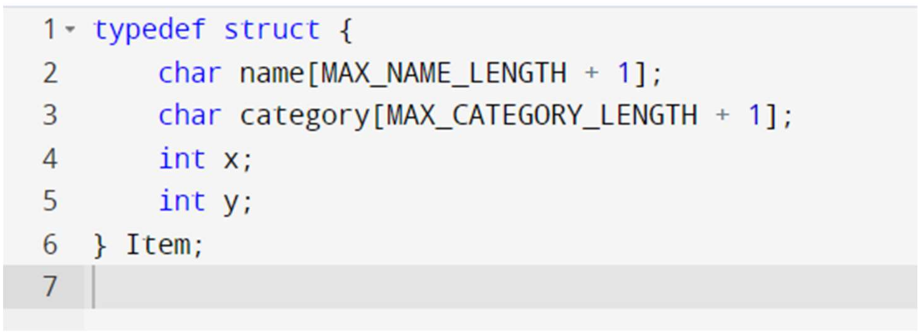
Report for Final Project

Introduction

This is the final report for Computer Software II 2024. The project involves developing and designing a spatial database management system that allows users to search up specific locations along with multiple other features. Users can perform tasks such as inserting new items, loading and saving data from files, searching for items by name, category, and spatial criteria, and generating a visual map on a HTML site. This project uses C and focuses on utilizing spatial queries and various data operations related to it.

Explanations of Design, Approach, and Algorithm

We declare the 'Item' structure, which essentially stores information about each item like its name, category and coordinates. It is essentially the building block of the whole program. It is implemented as the following in **Item.h** file.



```c
typedef struct {
    char name[MAX_NAME_LENGTH + 1];
    char category[MAX_CATEGORY_LENGTH + 1];
    int x;
    int y;
} Item;

```

Program Design

The program primarily utilizes the modular approach wherein it divides the functions into different modules. The main ones are:

Main Menu (main_menu.c): Provides the user interface and handles user inputs for various operations. The **main_menu.c** module is dependent on the **database.c** as it fetches the function's core code from **database.c**. So, for instance, if the user inputs 'i' key, **main_menu.c** will execute the **insert_item** function in **database.c**.

Database Operations (database.c): Manages functionality for each operation. The results from **database.c** are then passed onto the **print_page** function in **page.c** to display the output.

Page Display (page.c): Handles the generation of HTML pages for displaying the search results and the map.

As you can see, in a modular design, each function serves a different purpose but, in the end, they are all connected to each other and cannot function if there is an error in one of them. Now let's look at the function of the other files.

Database.h: This file declares the function's prototypes

Item.h: As we saw earlier, this file is used to define the 'Item' structure and also the **MAX_NAME_LENGTH** and **MAX_CATEGORY_LENGTH** to limit large memory usage.

Makefile: This file is used to compile the program. Essentially it builds the executables, cleans up object files and ensures correct build order.

Page.h: This file is like the backbone of **page.c** file. Similar to **database.h**, it ensures **page.c** executes properly.

Data.csv: Storage file for the data items. Stores information such as item name, category and coordinates.

Brief summary of the core functions in database.c:

insert_item(): Prompts the user to input item details and stores the item in the database.

print_all_items(): Prints all stored items to the console.

load_items(): Loads item data from the csv file.

save_items(): Saves the current item data to the csv file.

find_items_by_name(): Searches for items matching a given name by the user.

find_items_by_category(): Searches for items matching a given category.

find_items_by_range(): Searches for items within a specified distance from a point.

find_nearest_neighbor(): Finds the item closest to a given point.

find_items_by_partial_name(): Searches for items partially matching a given name.

print_page(): Generates an HTML page displaying the search results.

Discussion and Limitations

In terms of limitations of this program, as we can see, it does not use dynamic memory management. This can lead to potential inefficient memory usage. In real world applications, the majority of the applications use dynamic memory management and allocation.

Future Improvements

Certainly, the most impactful improvement would be to implement dynamic data structures. This could include the use of linked lists and hash tables to handle larger datasets more efficiently. Besides that, the GUI can be improved to make it more user-friendly and easier to navigate for anyone. Advanced search features can also be added to the core functions which can allow sophisticated and complex operations.

Conclusion

For a basic spatial database system, this program perfectly implements the essential features required such as data insertion, item search and etc. It builds the foundation for how spatial database management and modular design work. Upon these foundations, more complex programs are then built.