

Code Implementation:

```
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <cstdlib>
#include <direct.h> // For _mkdir

using namespace std;

void getTableDimensions(int& rows, int& cols) {
    cout << "Enter the number of columns: ";
    cin >> cols;
    cout << "Enter the number of rows: ";
    cin >> rows;
}

vector<vector<string>> getTableData(int rows, int cols) {
    vector<vector<string>> table(rows, vector<string>(cols));
    cout << "Enter the table data (headers in the first row):\n";
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            cout << "Enter data for cell [" << i + 1 << "][" << j + 1 << "]: ";
            cin >> ws; // Clear whitespace
            getline(cin, table[i][j]);
        }
    }
    return table;
}

string generateHTMLContent(const vector<vector<string>>& table, int rows, int cols)
{
    string htmlContent = R"(<!DOCTYPE html>
<html>
<head>
<style>
table {
    font-family: arial, sans-serif;
    border-collapse: collapse;
    width: 100%;
}
td, th {
    border: 1px solid #dddddd;
    text-align: left;
    padding: 8px;
}
tr:nth-child(even) {
    background-color: #dddddd;
}
</style>
</head>
<body>
<h2>HTML Table</h2>
<table>
)";
    for (int i = 0; i < rows; i++) {
```

```

        htmlContent += " <tr>\n";
        for (int j = 0; j < cols; j++) {
            if (i == 0)
                htmlContent += " <th>" + table[i][j] + "</th>\n";
            else
                htmlContent += " <td>" + table[i][j] + "</td>\n";
        }
        htmlContent += " </tr>\n";
    }
    htmlContent += R"(</table>
</body>
</html>)" ;
    return htmlContent;
}

string getDesktopPath() {
    char* userProfile = nullptr;
    size_t len = 0;

    // Use _dupenv_s to safely retrieve the USERPROFILE environment variable
    errno_t err = _dupenv_s(&userProfile, &len, "USERPROFILE");
    if (err || userProfile == nullptr) {
        cerr << "Error retrieving the user profile path!" << endl;
        exit(1);
    }

    string desktopPath = string(userProfile) + "\\Documents\\HTML_Table";
    free(userProfile); // Free the allocated memory

    if (_mkdir(desktopPath.c_str()) != 0 && errno != EEXIST) {
        cerr << "Failed to create directory: " << desktopPath << endl;
        exit(1);
    }

    return desktopPath + "\\table.html";
}

void saveHTMLToFile(const string& htmlContent, const string& filePath) {
    ofstream htmlFile(filePath);
    if (htmlFile.is_open()) {
        htmlFile << htmlContent;
        htmlFile.close();
        cout << "HTML file created successfully at: " << filePath << endl;
    }
    else {
        cerr << "Unable to create HTML file." << endl;
    }
}

int main() {
    int rows, cols;
    getTableDimensions(rows, cols);
    vector<vector<string>> table = getTableData(rows, cols);
    string htmlContent = generateHTMLContent(table, rows, cols);
    string filePath = getDesktopPath();
    saveHTMLToFile(htmlContent, filePath);
    return 0;
}

```