# SAVITRIBAI PHULE PUNE UNIVERSITY



**A MINI PROJECT REPORT ON**

# “TESTING ON WEB BASED APPLICATION”

|  |  |
| --- | --- |
| **Submitted by**  **CLASS: BE DIV: A** |  |
| **Name: Anshuman Kalbhor** | **Roll no: 04** |

## Under the Guidance of

Prof. Manisha Desai



**DEPARTMENT OF COMPUTER ENGINEERING**

**RMD SINHGAD SCHOOL OF ENGINEERING**

WARJE, PUNE 411058

### Academic Year 2024 – 25(SEM-I)



## DEPARTMENT OF COMPUTER ENGINEERING

**RMD SINHGAD SCHOOL OF ENGINEERING**

WARJE, PUNE 411058

## CERTIFICATE

This is to certify that the project report entitles

“TESTING ON WEB BASED APPLICATION”

*Submitted by*

Name: Anshuman Kalbhor PRN No : 72217735G

This is a bonafide work carried out by them under the supervision of Prof. Manisha Desai . And it is submitted towards the partial fulfillment of the requirement of Savatribai phule university (SPPU) for Final Year.

|  |  |
| --- | --- |
| **(Mrs. Manisha Desai)** | **(Dr. Vina M. Lomte)** |
| Guide | Head, |
| Department of Computer Engineering | Department of Computer Engineering |

**(Dr. V. V. Dixit)** Principal,

RMD Sinhgad School of Engineering Pune – 58

### Certificate by Guide

This is to certify that Mr.Anshuman Kalbhor has completed the MINI Project work under my guidance and supervision and that, I have verified the work for its originality in documentation, problem statement, implementation and results presented in the Project. Any reproduction of other necessary work is with the prior permission and has given due ownership and included in the references.

Place: Pune

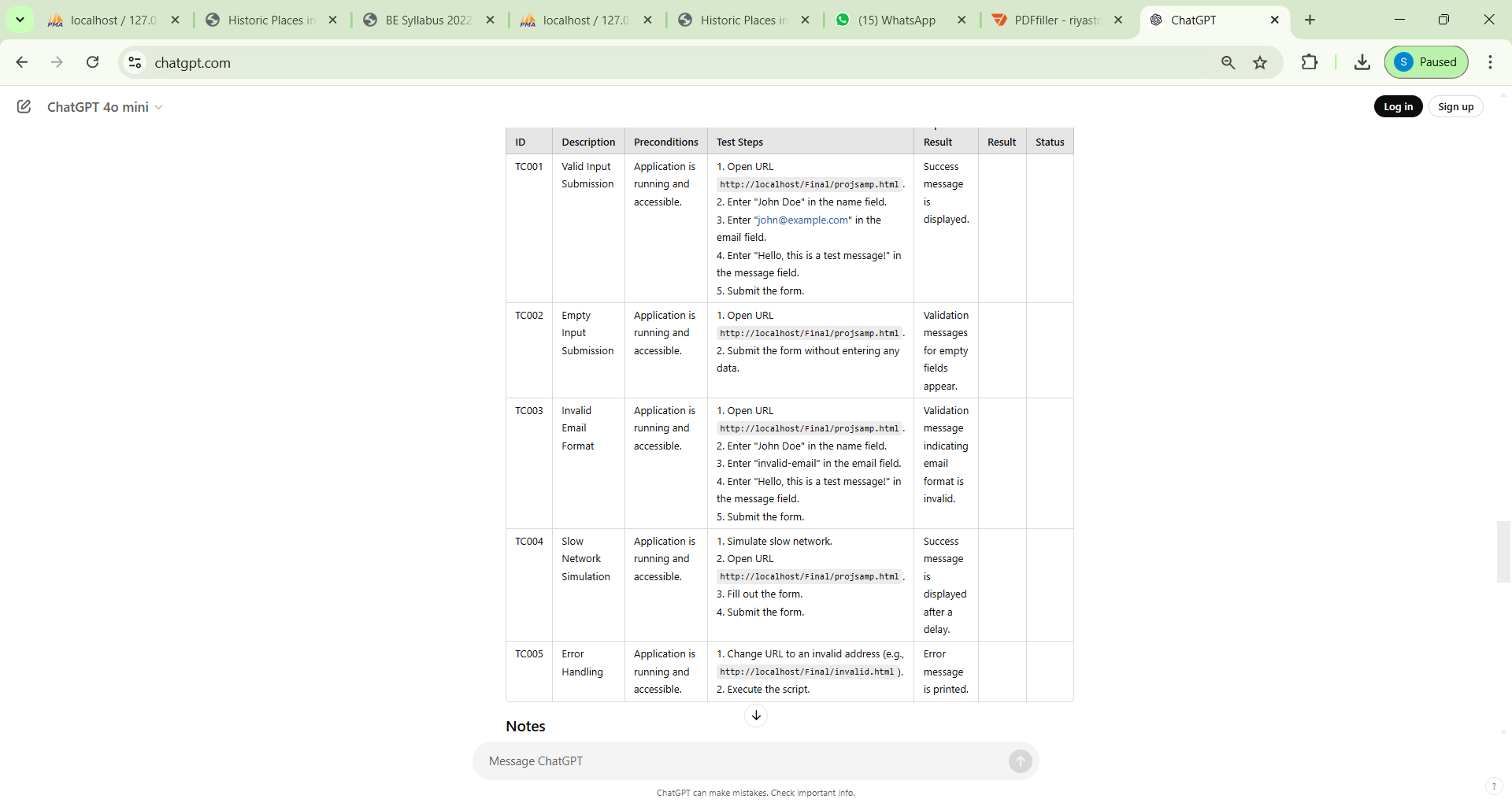
Date:

Signature of Guide

**Mrs. Manisha Desa**

|  |
| --- |
| **Mini Project :Software Testing and Quality Assurance Aim:-**  Software Testing and Quality Assurance Mini Project Dynamic website of form, system information using HTML, CSS, JAVASCRIPT And JAVA, PHP, MySQL database used to store user account, comment, and registration form details. Regular Expression testcases for testing purpose.    **Introduction:-**    In this project, a small web-based application will be developed that includes a login system for admin users, where account details, comments, and form data are stored in a MySQL database. The application will be created using HTML, CSS, JavaScript, Java, PHP, and MySQL, providing a userfriendly interface and secure access. The project includes the implementation of regular expression test cases for validation purposes, as well as Selenium WebDriver for automated testing. The project aims to demonstrate the interaction between client-side and server-side programming along with database management.    **Testing:-**  The testing strategy employed for this project ensures that all functional, security, and  validation requirements are met. By combining manual, automated (Selenium WebDriver), and exploratory testing, the application undergoes comprehensive quality assurance. This multi-faceted approach ensures the robustness of the form system, comment management, and user registration. Automated tests facilitate regression testing, improving test efficiency and reducing the time needed to verify the application after updates.        **Test Plan:-**    **Objectives**  - Validate form submission functionality.  - Ensure proper handling of user input and success messages.  **Test Environment**  - Browser: Google Chrome  - WebDriver: ChromeDriver  - URL: `http://localhost/Final/projsamp.html`  **Test Scenarios**  **1. Valid Input Submission**  - Enter valid data ("John Doe", "john@example.com", "Hello, this is a test message!").  - Verify success message.  **2. Empty Input Submission**  **-** Submit without filling any fields.  **-** Check for validation messages.  **3. Invalid Email Format**  - Enter invalid email ("invalid-email") and valid name/message.  - Verify validation message for email.  **4. Slow Network Simulation**  **-** Simulate slow network, fill form, and submit.  - Check if success message appears eventually.  **5. Error Handling**  **-** Change URL to an invalid one.  - Confirm error message is printed.  **6. Test Data**  **-** Valid:  Name: "John Doe", Email: "john@example.com", Message: "Hello, this is a test message!"  - Invalid:  Email: "invalid-email" |

**Test Cases**



|  |
| --- |
| Test Scenario:-    **What is a Test Scenario?**  A Test Scenario is defined as any functionality that can be tested. It is a collective set of test cases which helps the testing team to determine the positive and negative characteristics of the project.    Test Scenario gives a high-level idea of what we need to test.      **Test Scenario:-**    For Covid19 Dynamic Web Application, a few test scenarios would be |

### 1. Successful Submission

* **Scenario**: Fill in all fields correctly and submit.
* **Expected Result**: The form submits successfully, and a success message appears.

### 2. Missing Name

* **Scenario**: Leave the name field empty and submit.
* **Expected Result**: An error message indicating that the name is required should appear.

### 3. Invalid Email Format

* **Scenario**: Enter an invalid email format (e.g., "johnexample.com").
* **Expected Result**: An error message indicating the email format is invalid should appear.

### 4. Missing Message

* **Scenario**: Leave the message field empty and submit.
* **Expected Result**: An error message indicating that the message is required should appear.

### 5. Long Input Data

* **Scenario**: Enter excessively long strings in the fields (e.g., 1000 characters).
* **Expected Result**: The application should either accept the input or display a relevant error message about character limits.

### 6. JavaScript Validation

* **Scenario**: If the page has JavaScript validations, enter invalid data to see if the script prevents submission.
* **Expected Result**: The form should not submit, and the appropriate error messages should be displayed.

### 7. Multiple Submissions

* **Scenario**: Submit the form multiple times in quick succession.
* **Expected Result**: The application should handle multiple submissions gracefully (either by preventing duplicate submissions or displaying appropriate error messages).

### 8. Field Autofocus

* **Scenario**: Check if the first input field has autofocus when the page loads.
* **Expected Result**: The name input field should be focused on page load.

### 9. Form Reset

* **Scenario**: Use a reset button (if available) to clear all fields.
* **Expected Result**: All input fields should be cleared upon clicking the reset button.

### 10. Successful Submission with Refresh

* **Scenario**: Submit the form, then refresh the page.
* **Expected Result**: The application should either show a confirmation that the form was submitted successfully or warn against resubmission.

### 11. Network Failure Simulation

* **Scenario**: Simulate a network failure while trying to submit the form.
* **Expected Result**: The application should handle the error gracefully and show an appropriate error message.

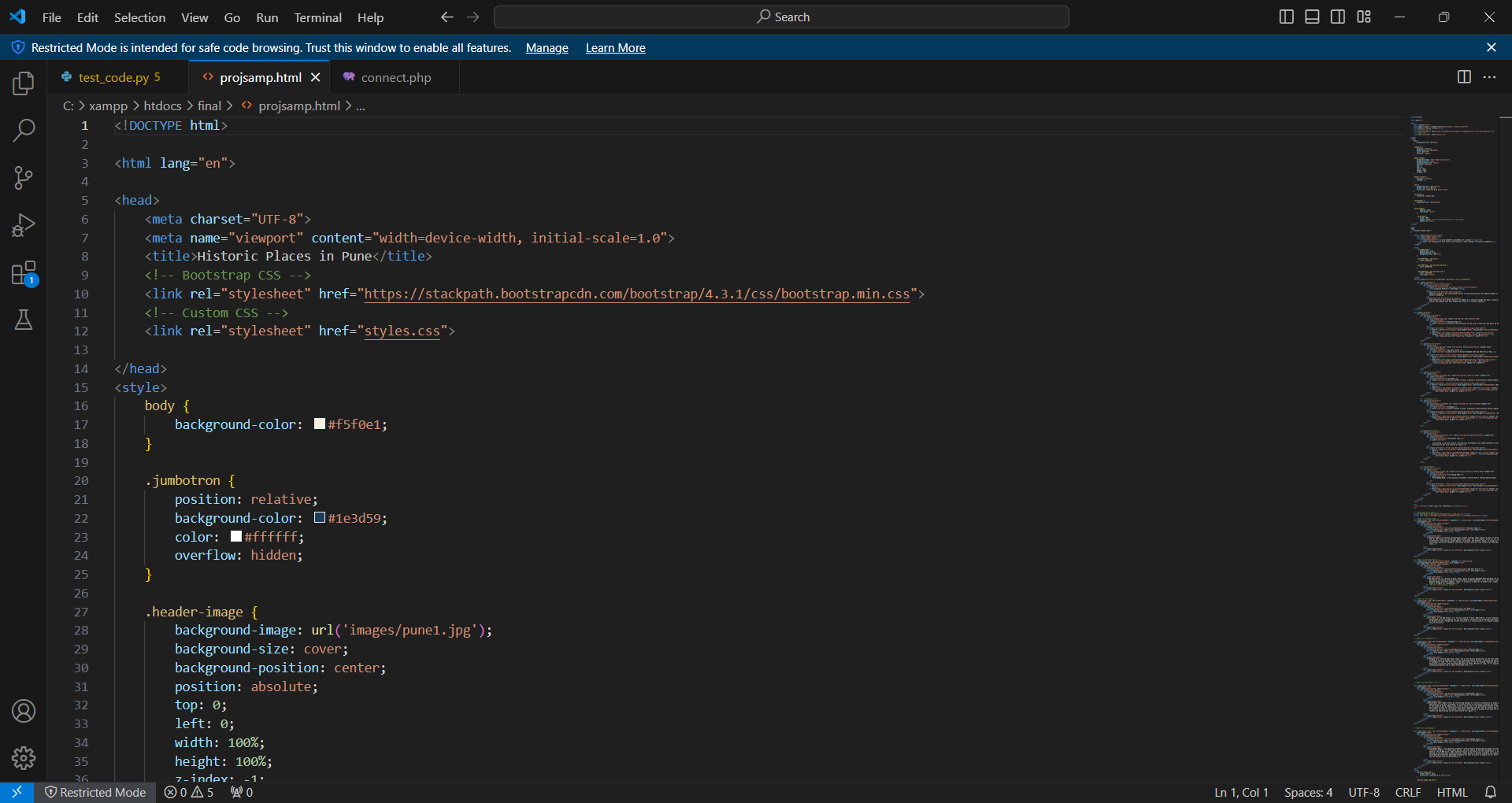
### 12. Accessibility Check

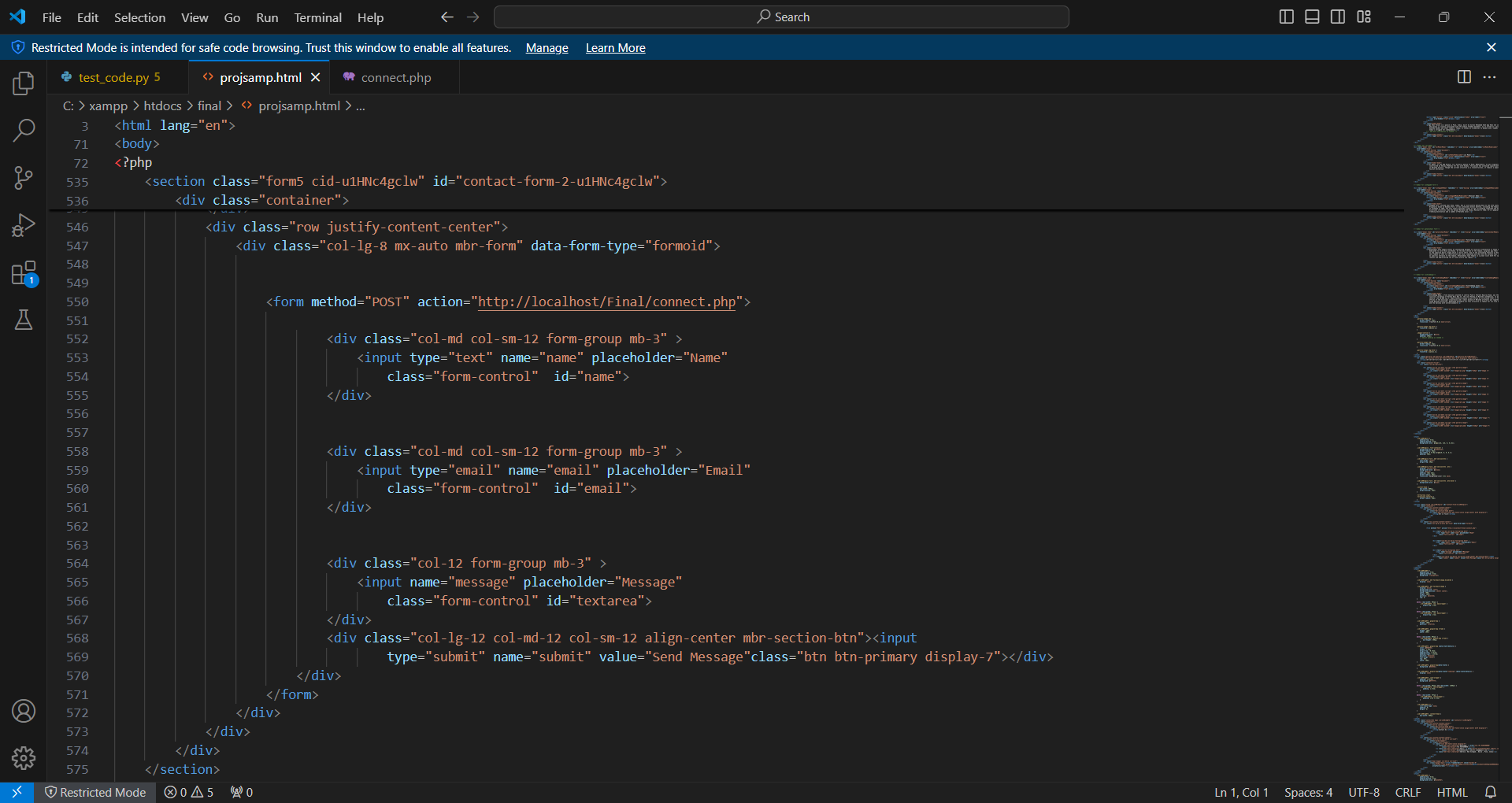
* **Scenario**: Test whether all fields are accessible via keyboard navigation.
* **Expected Result**: All fields should be focusable and accessible via keyboard shortcuts.

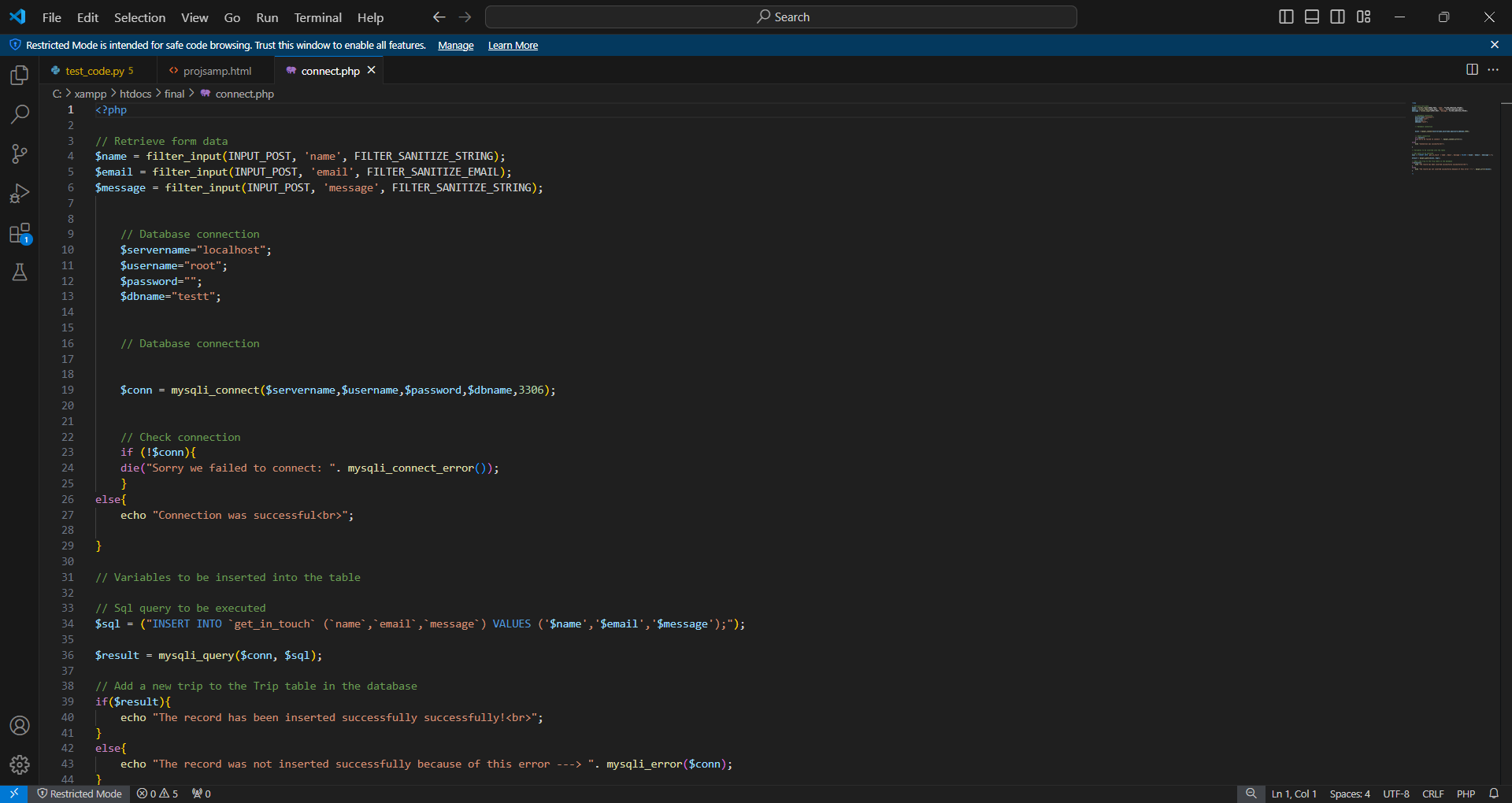
### 13. Browser Compatibility

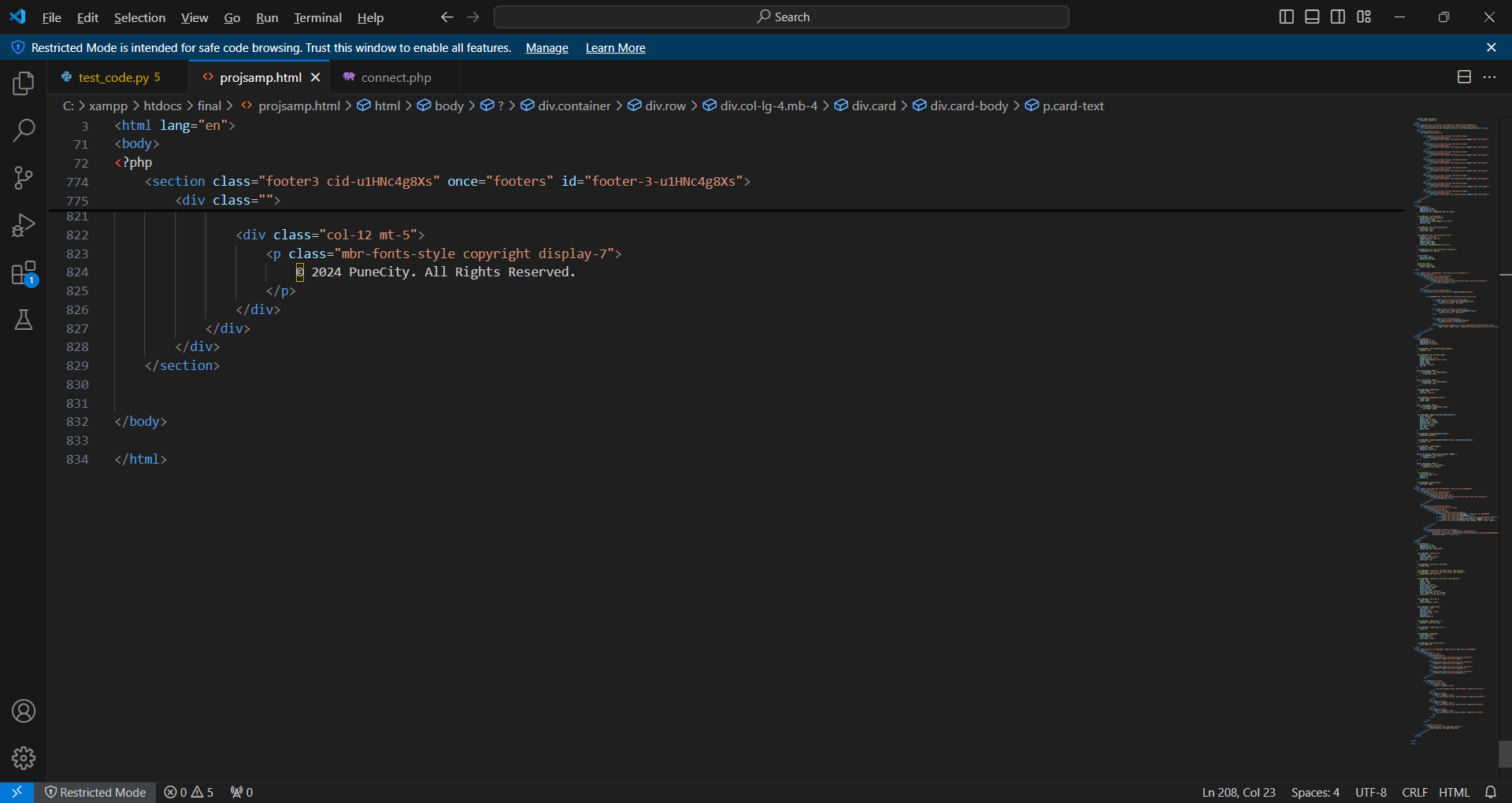
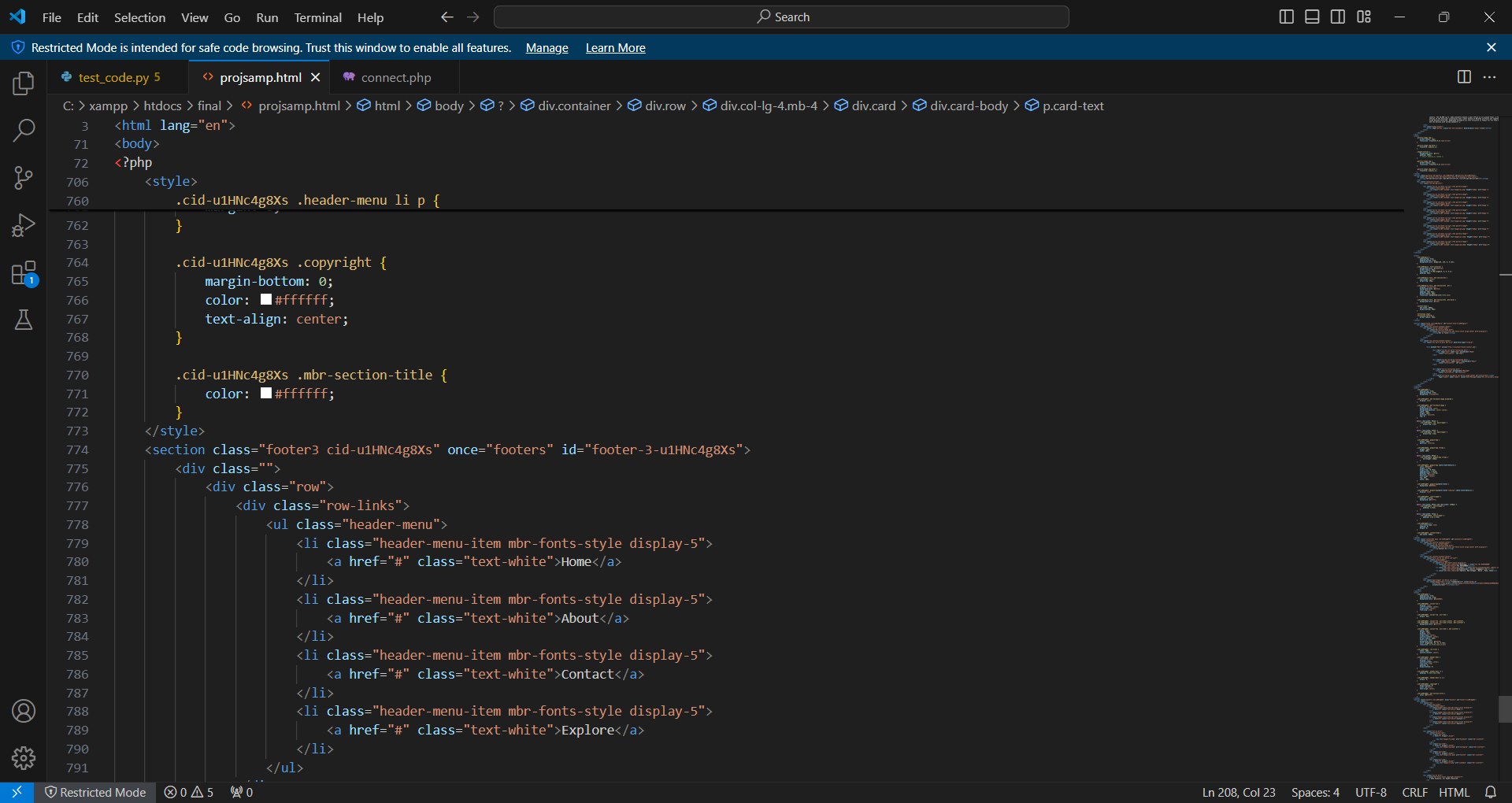
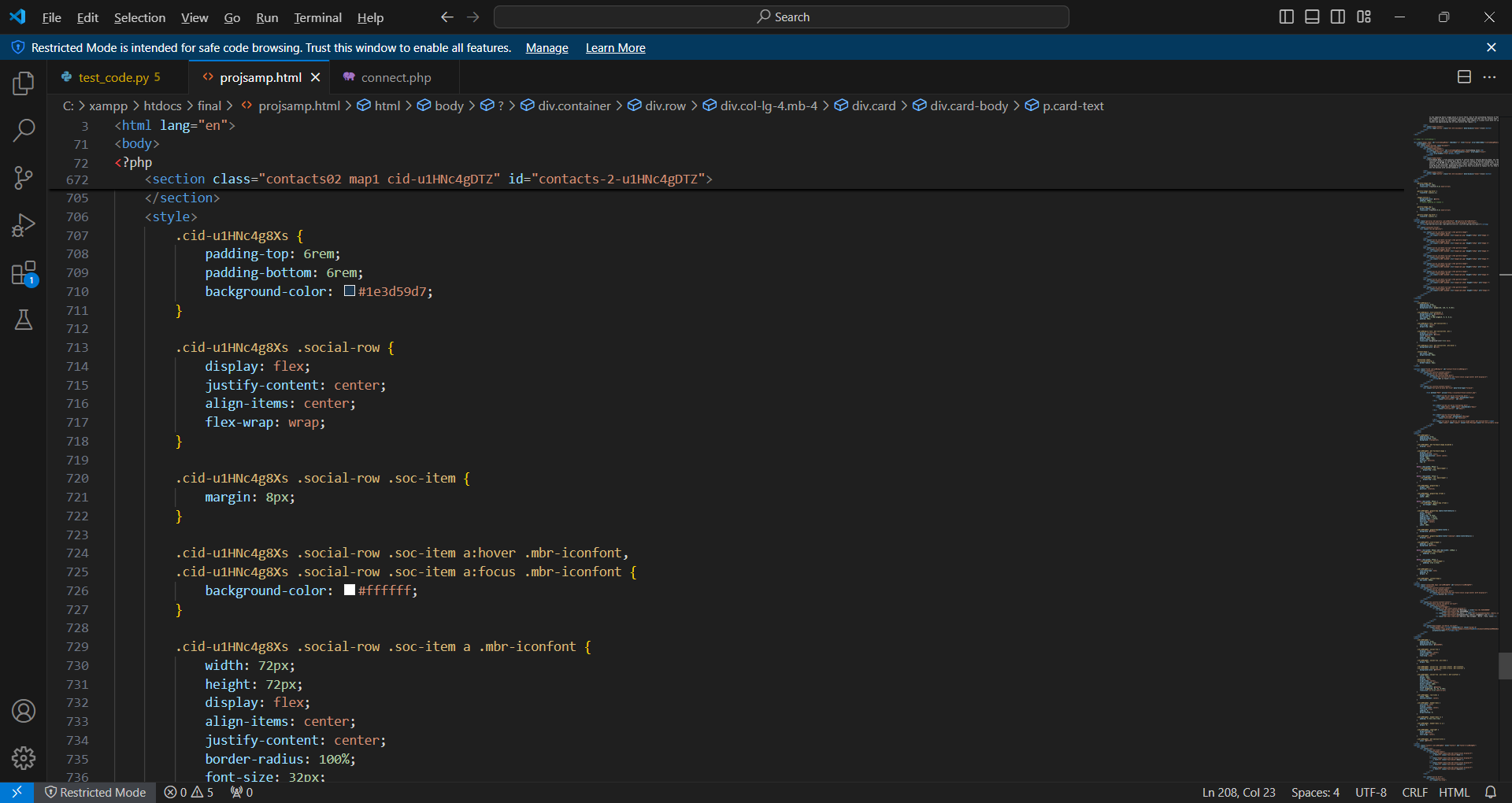
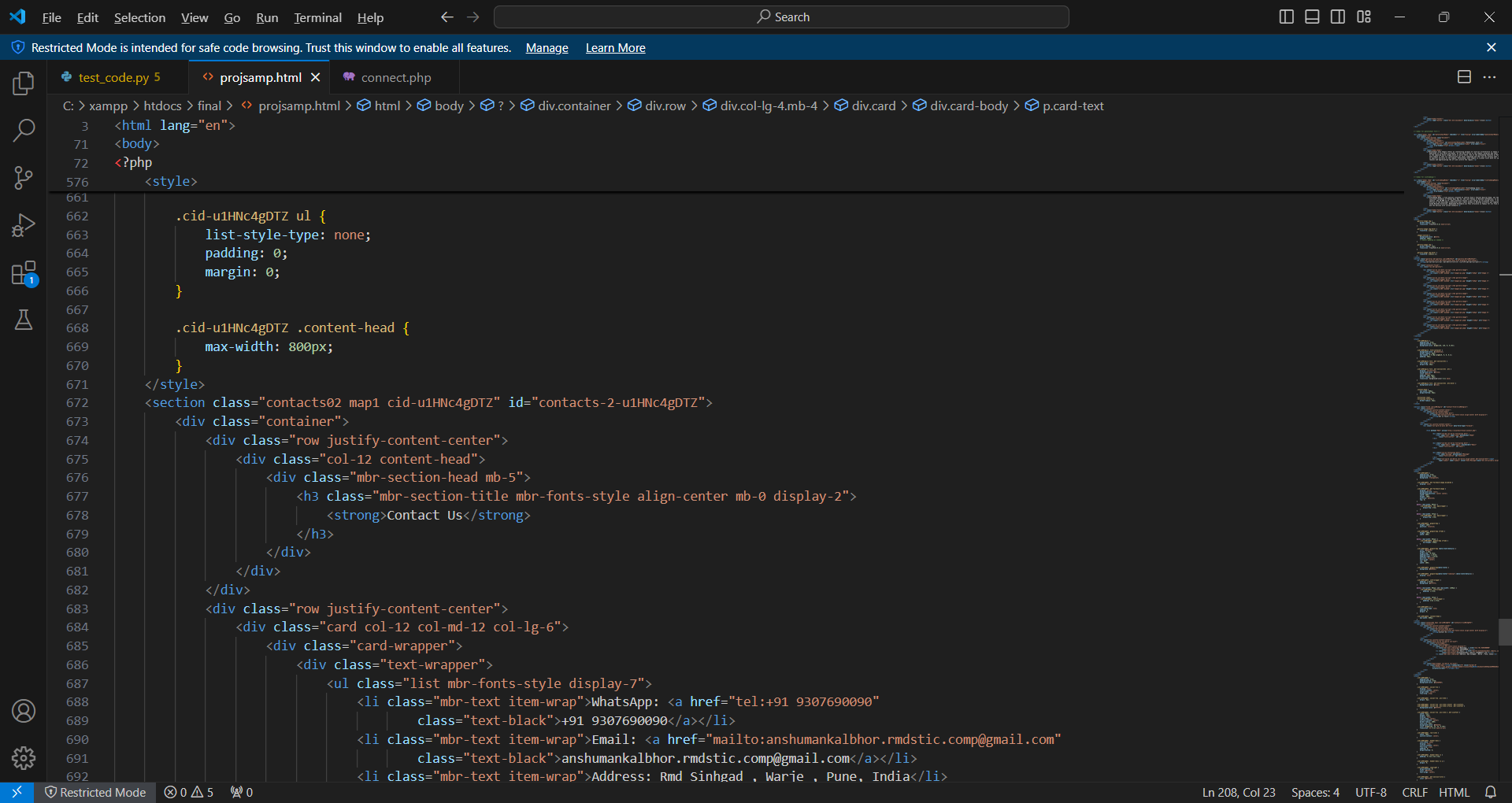
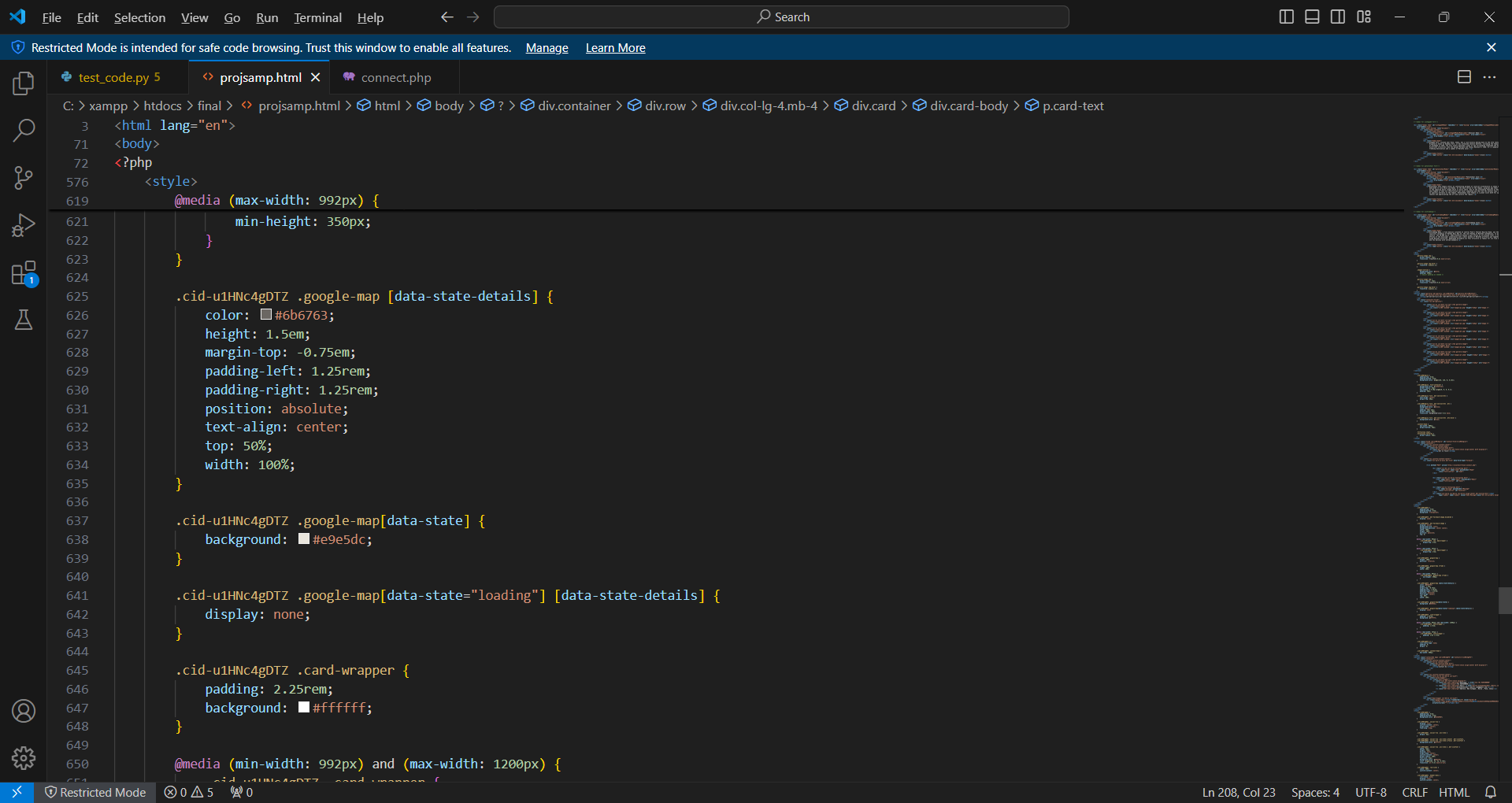
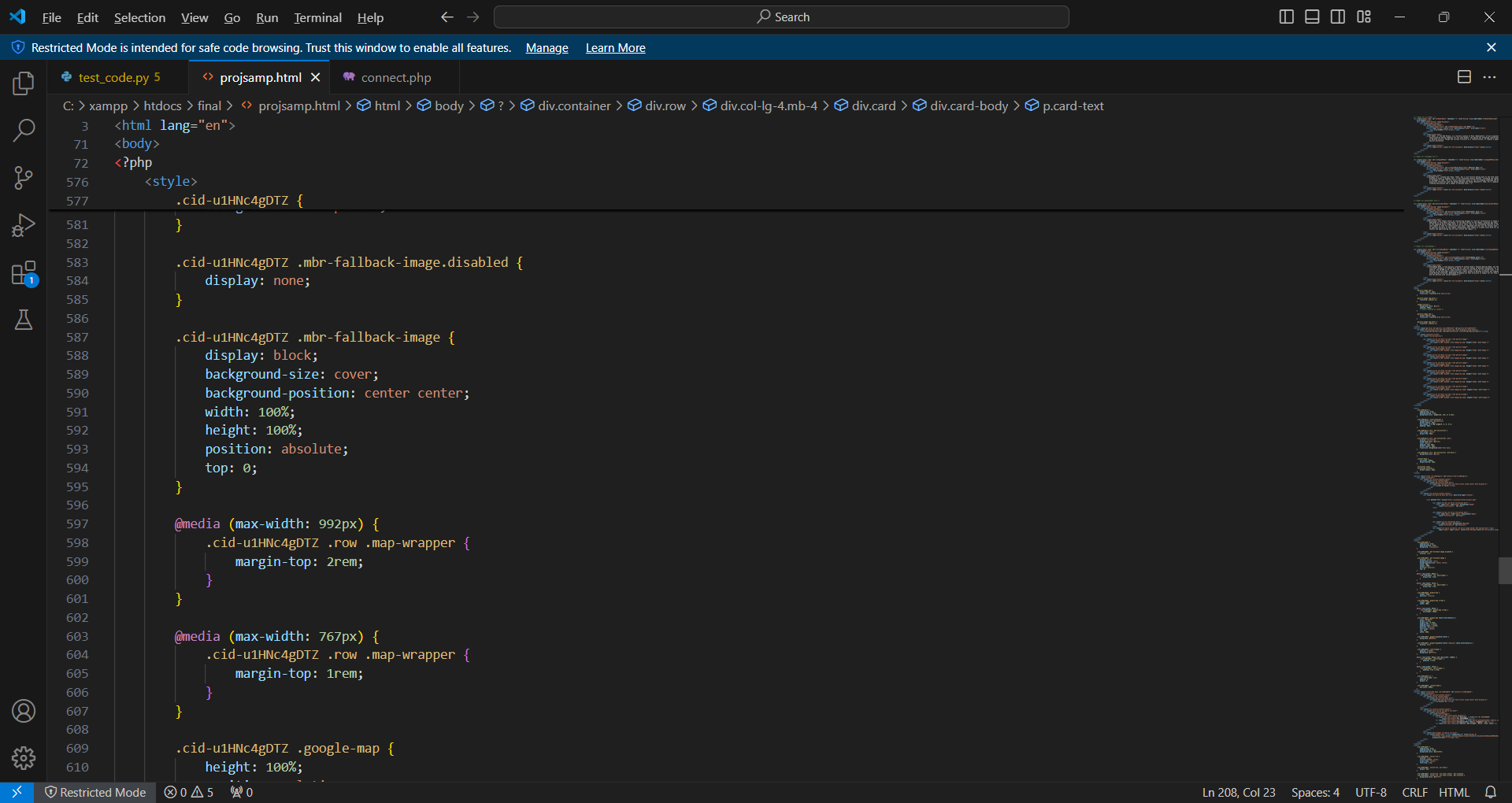
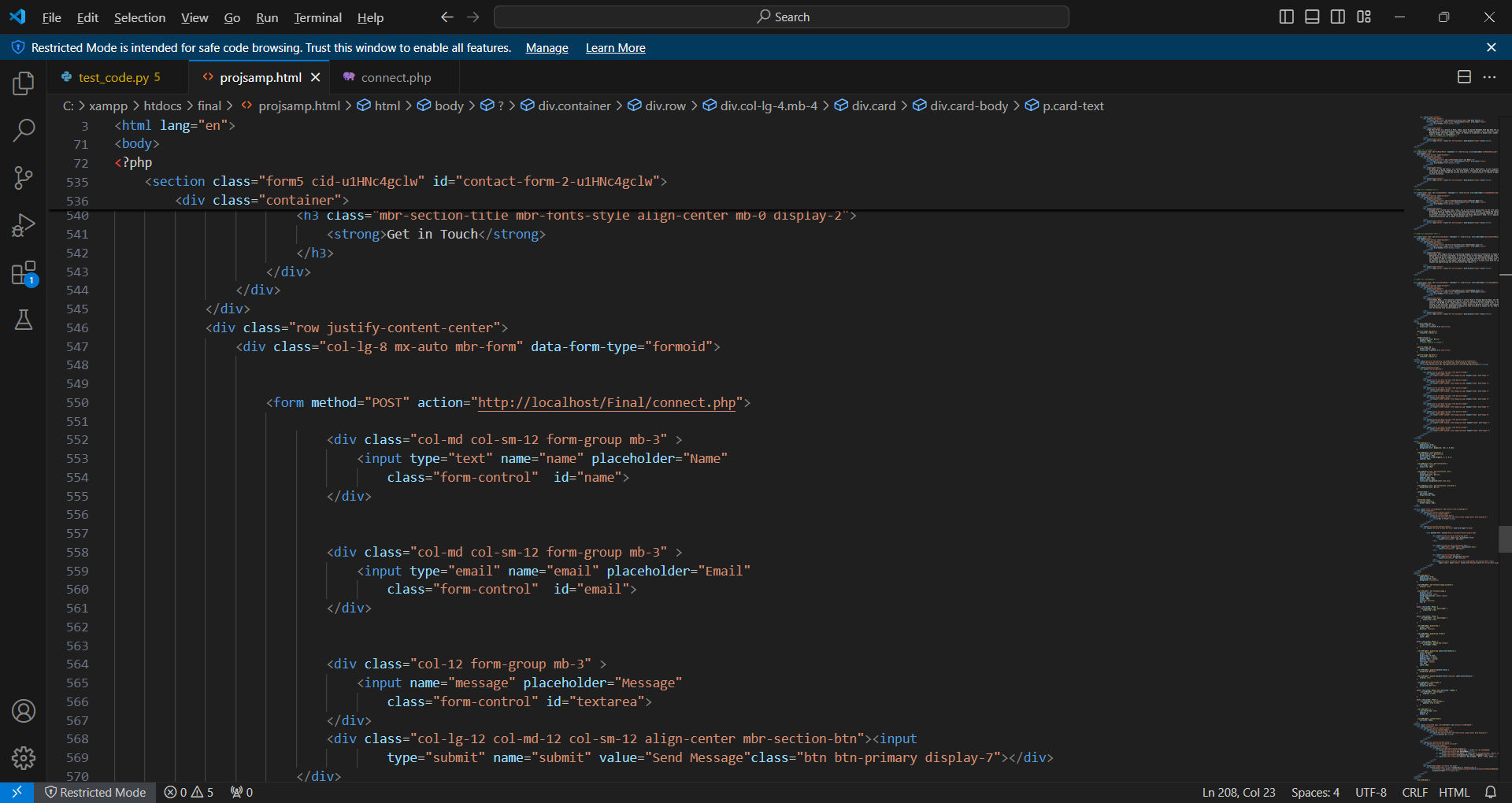
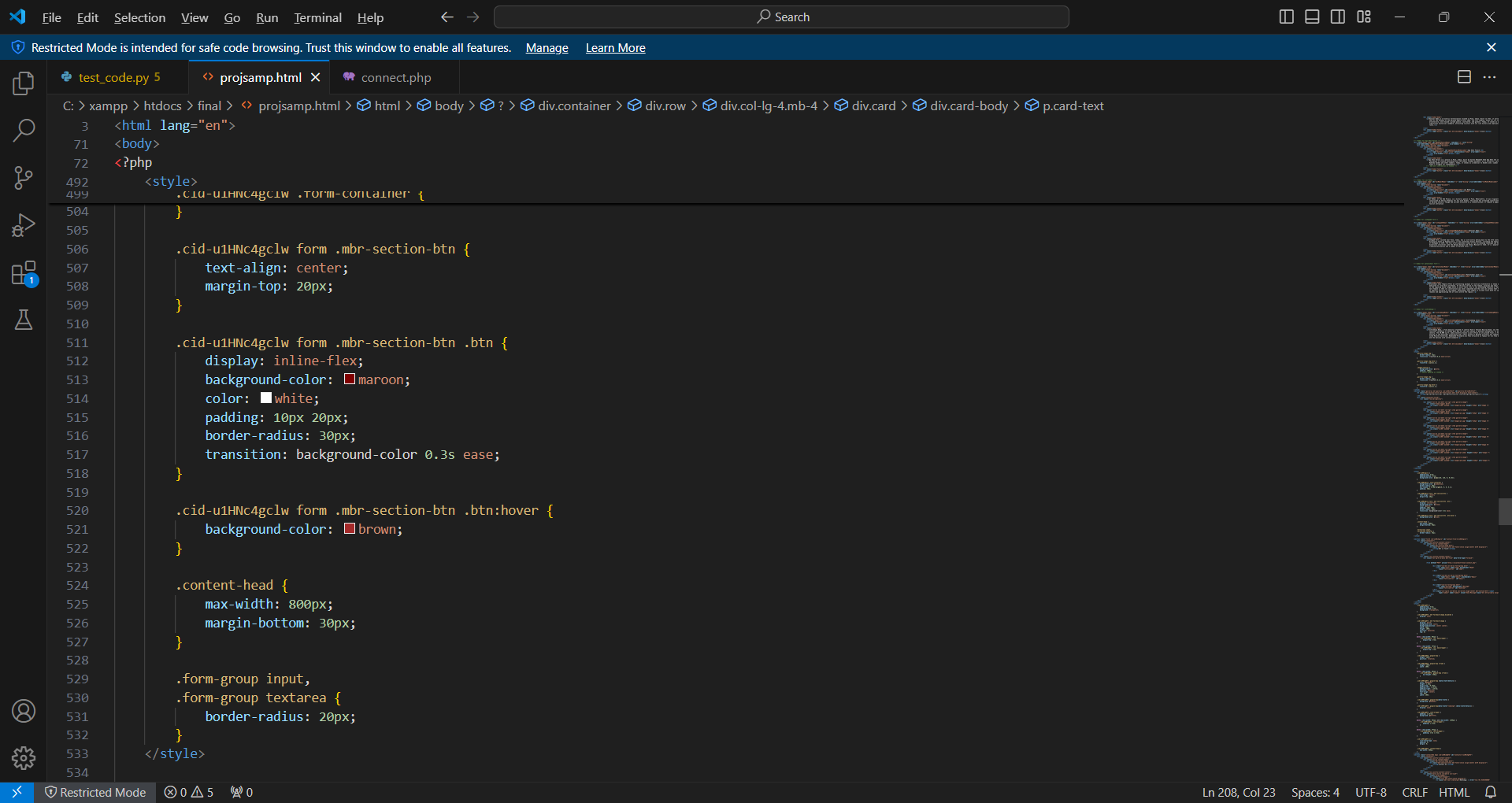
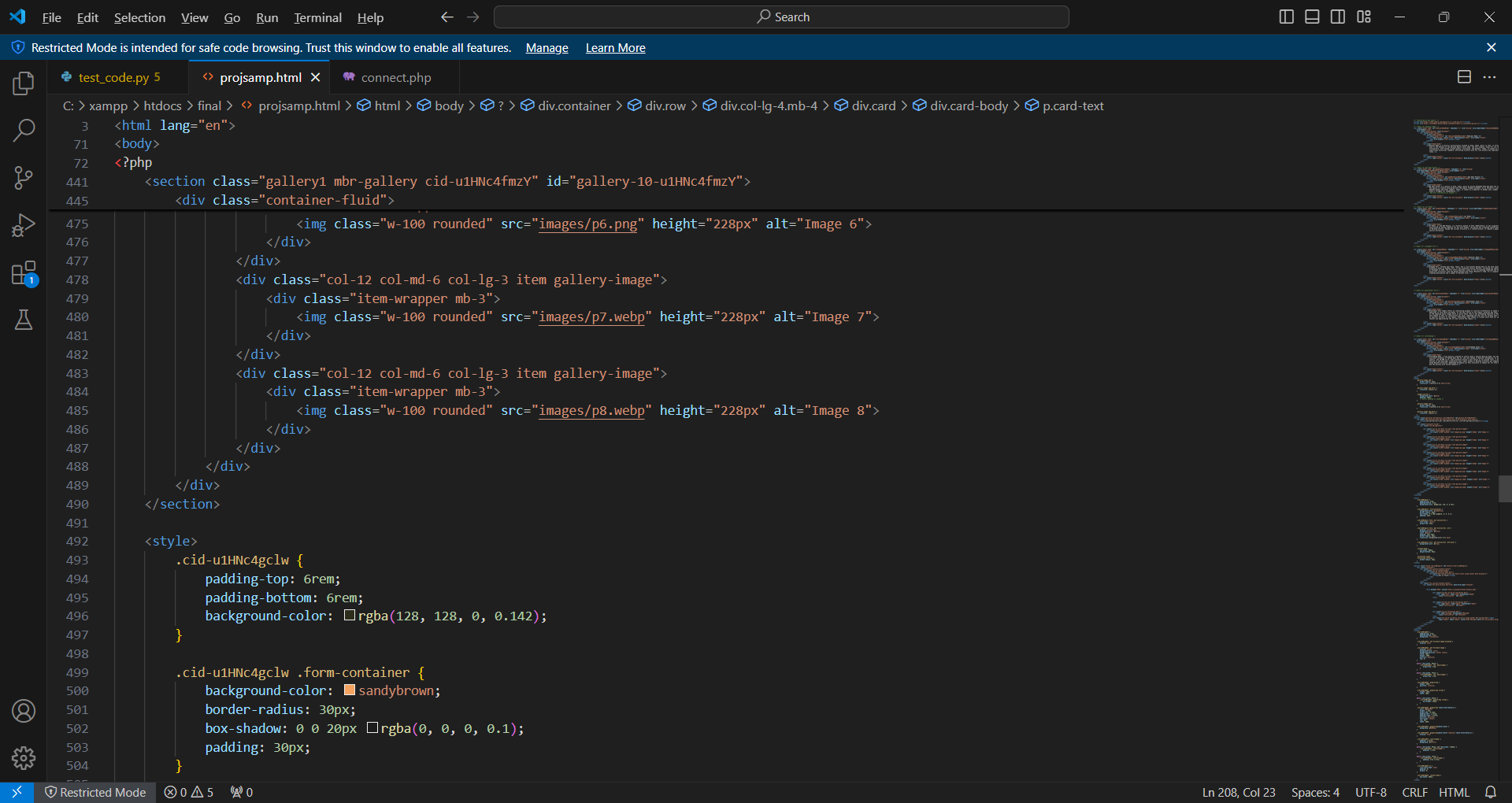
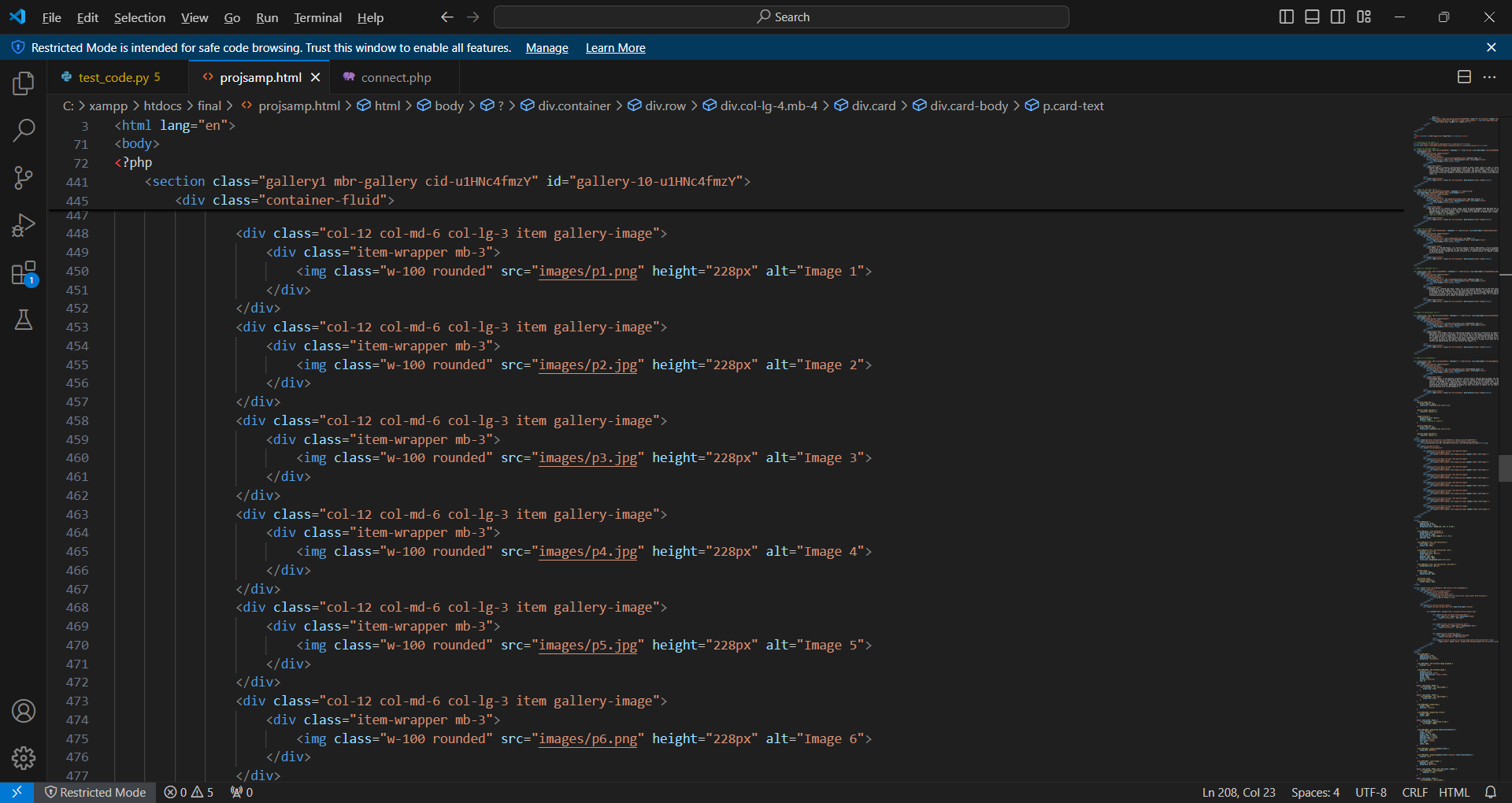
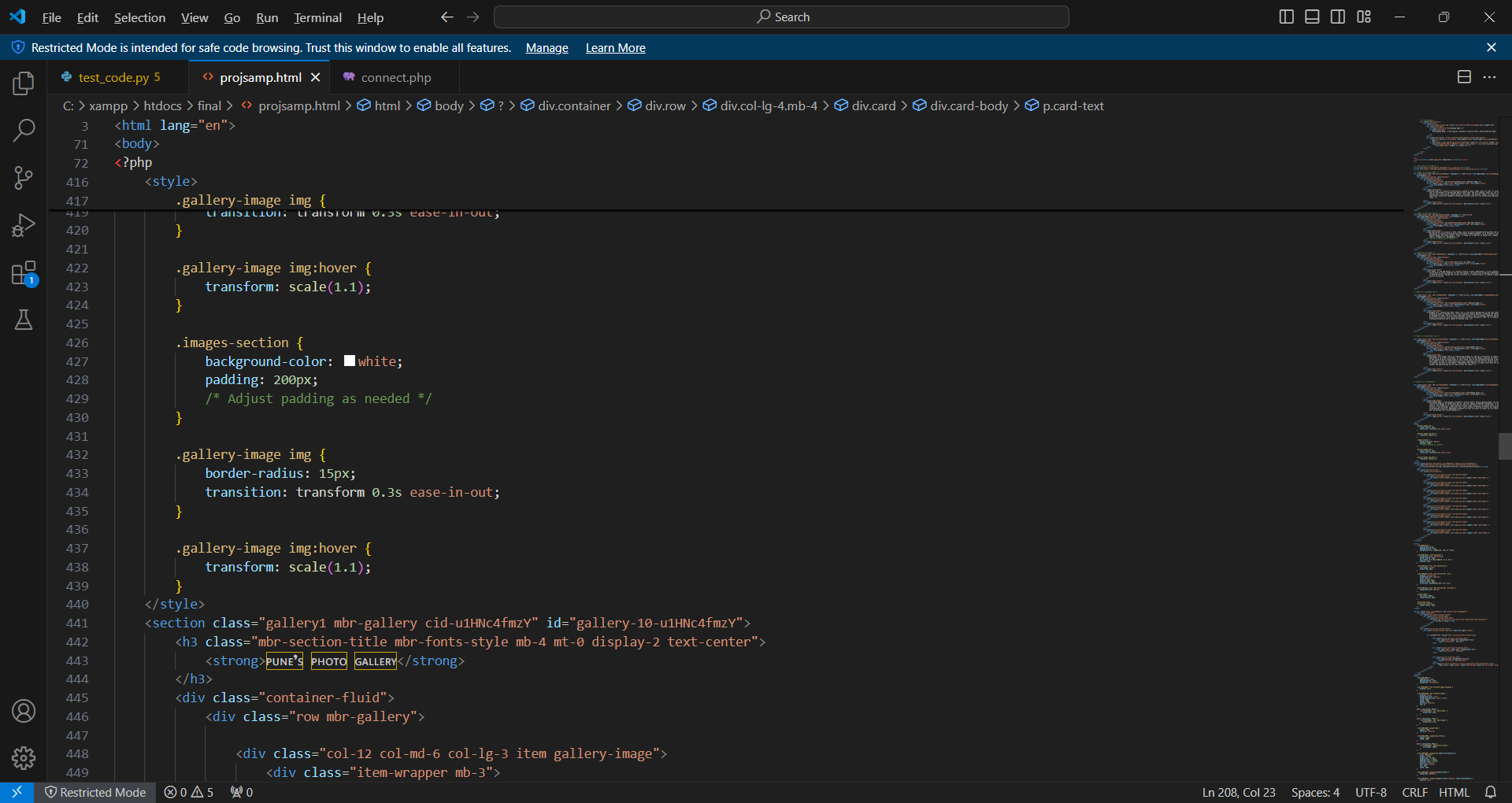
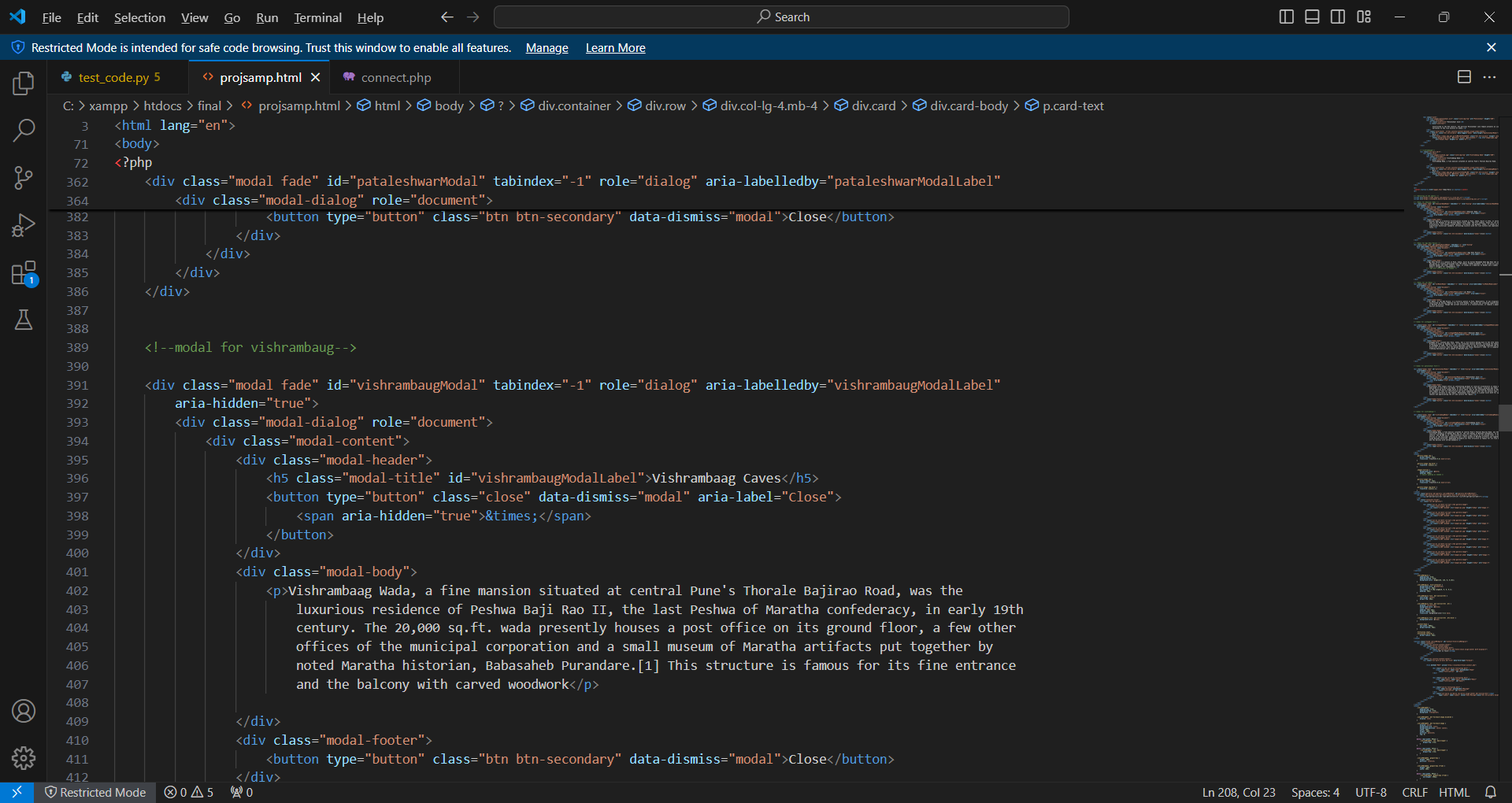
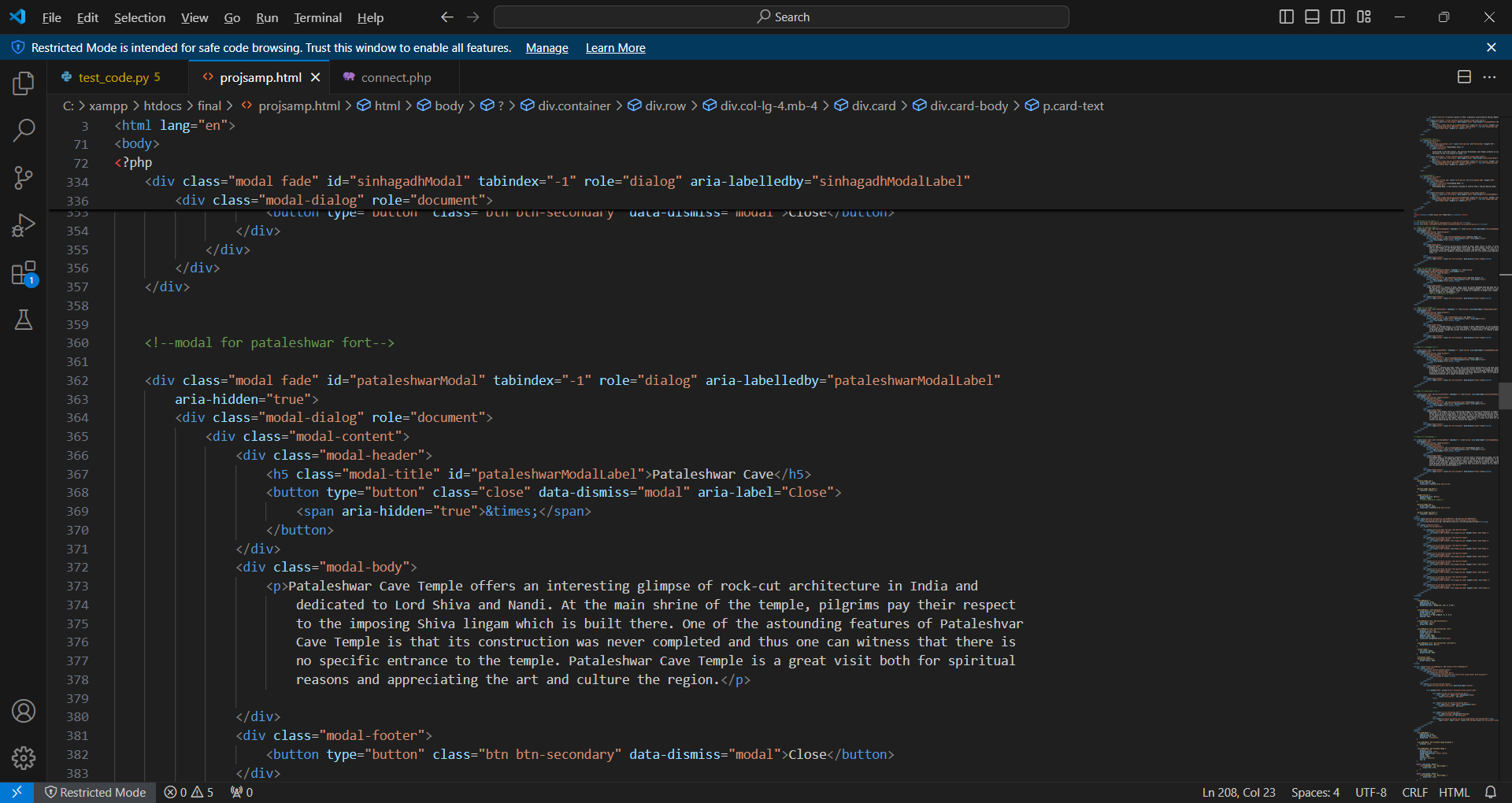
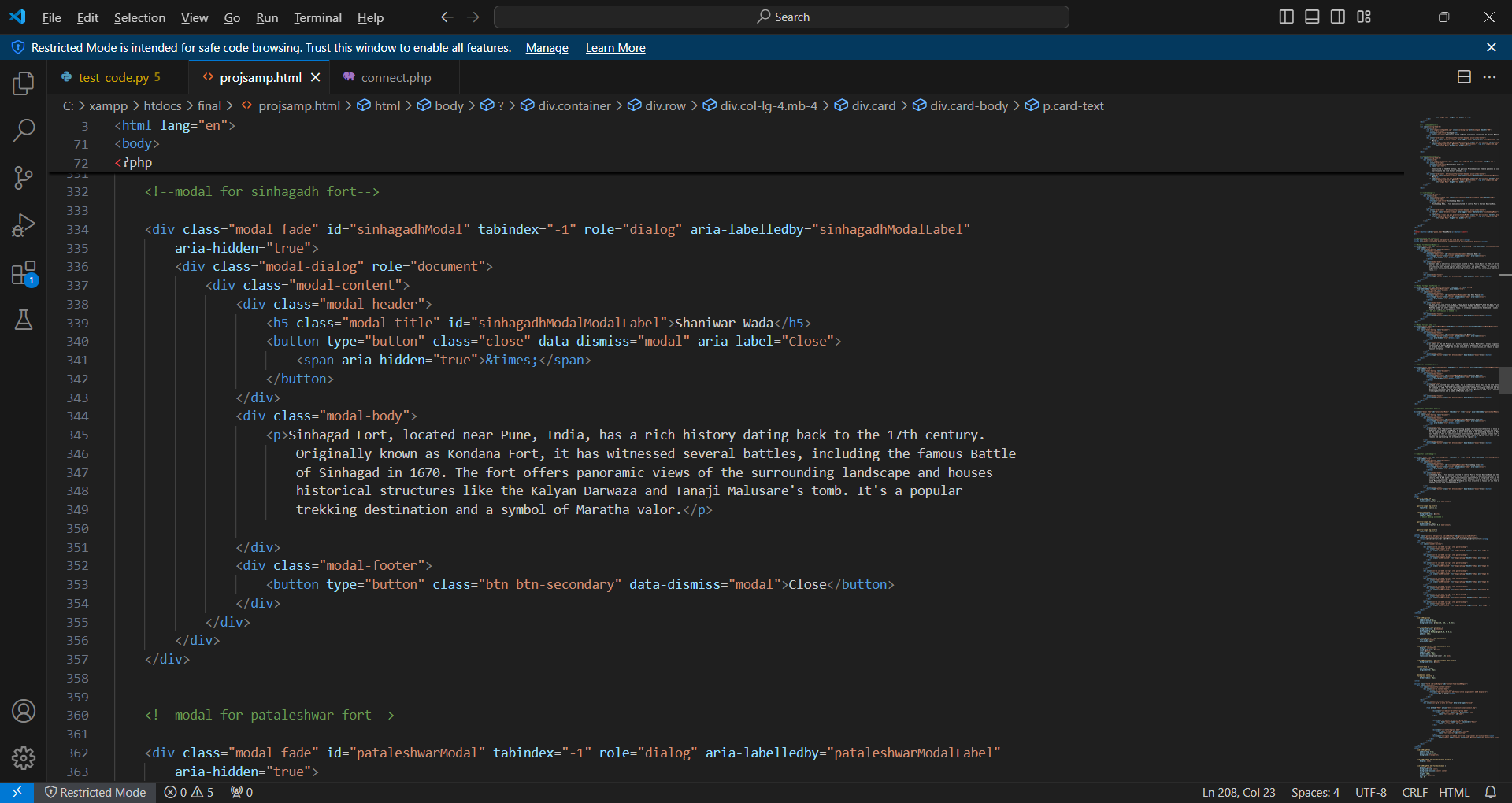
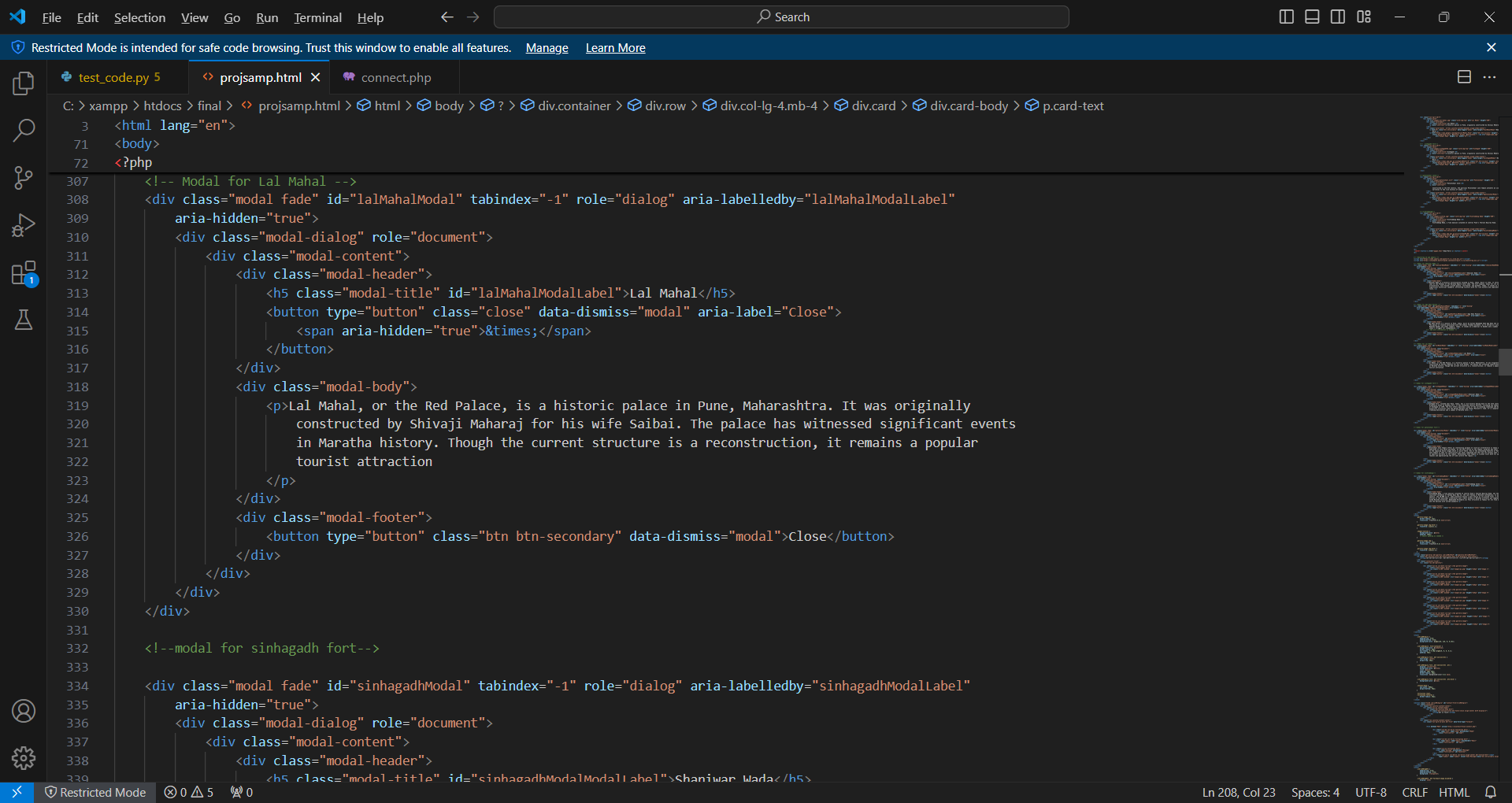
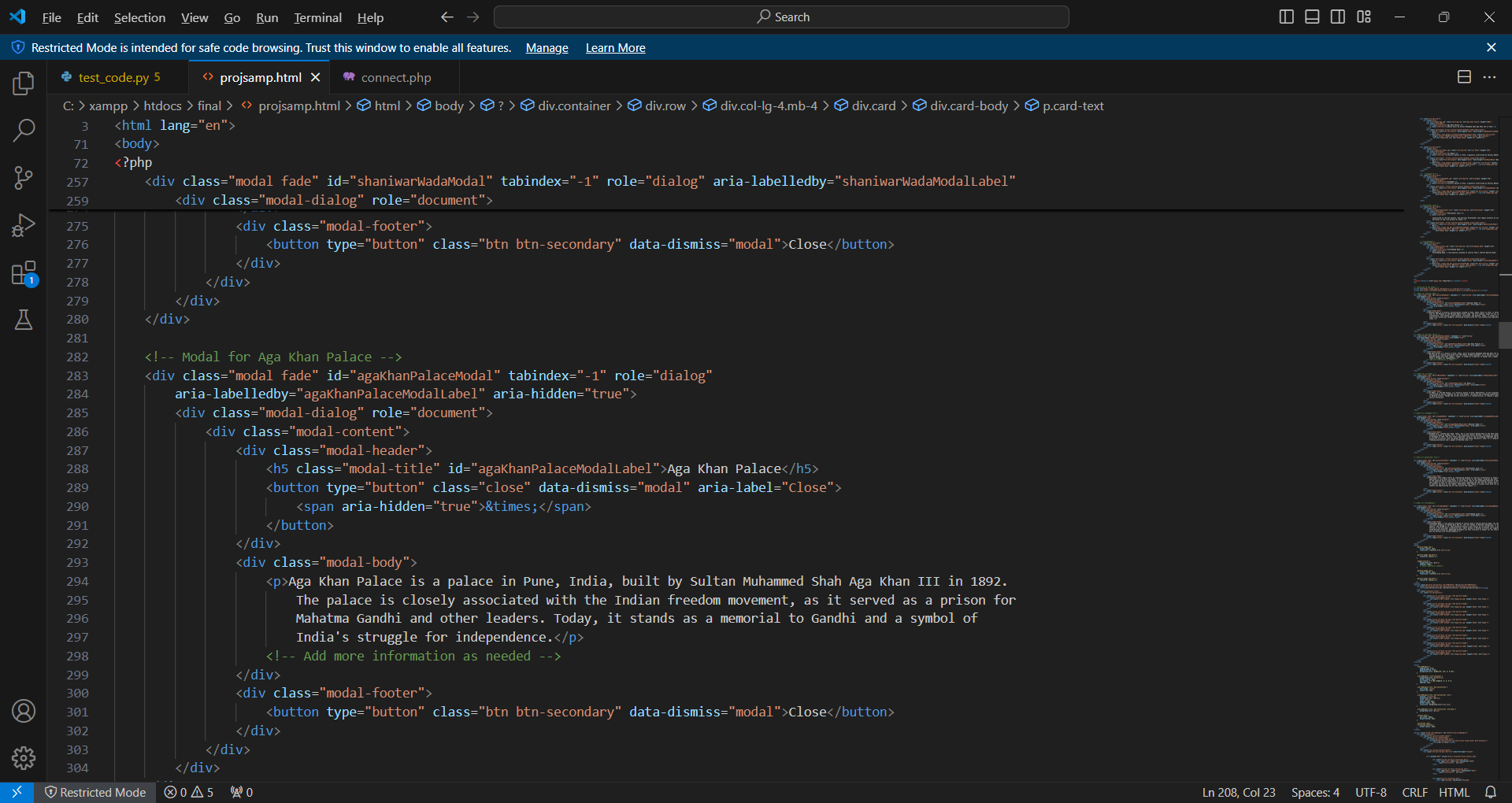
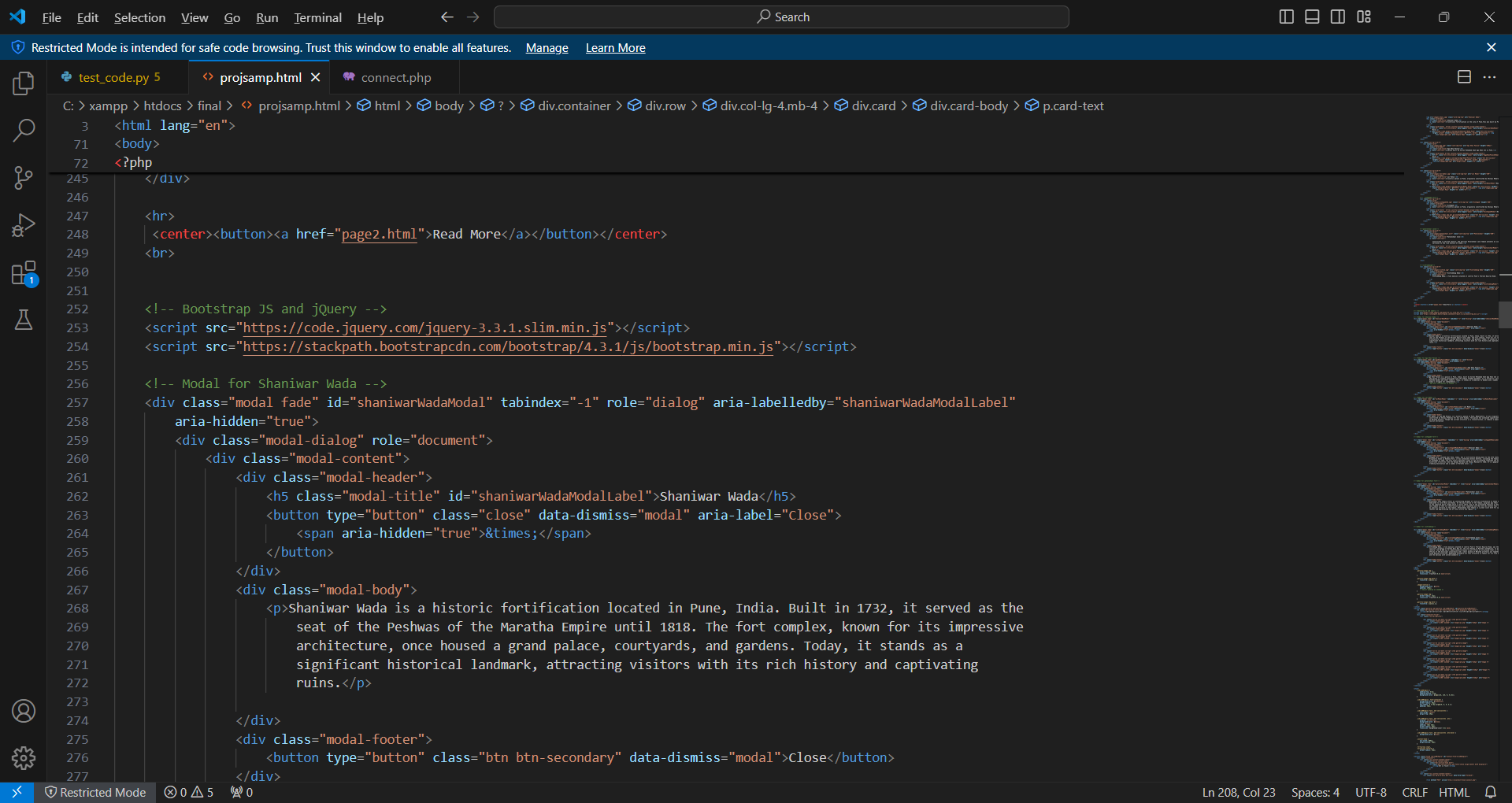
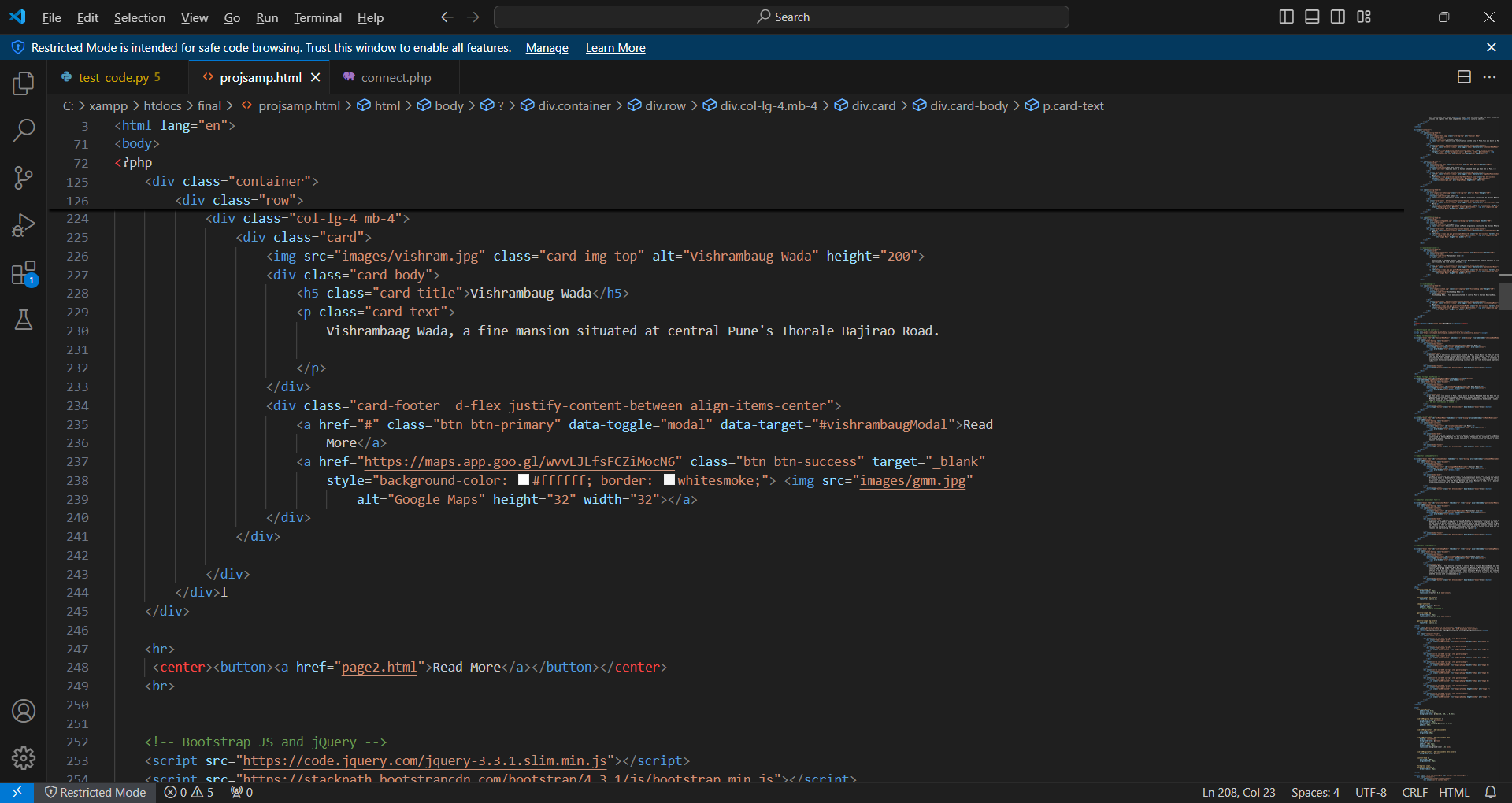
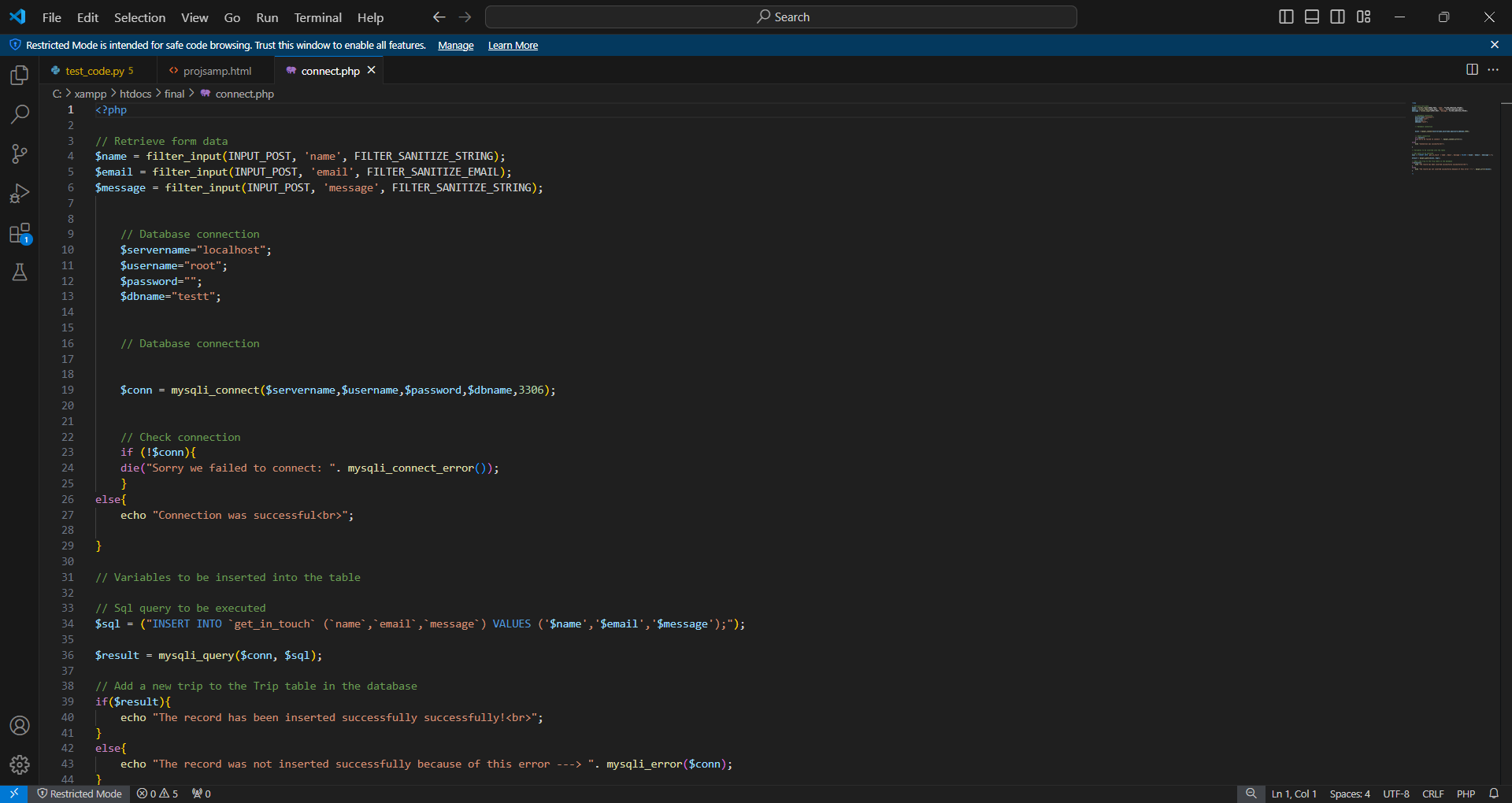
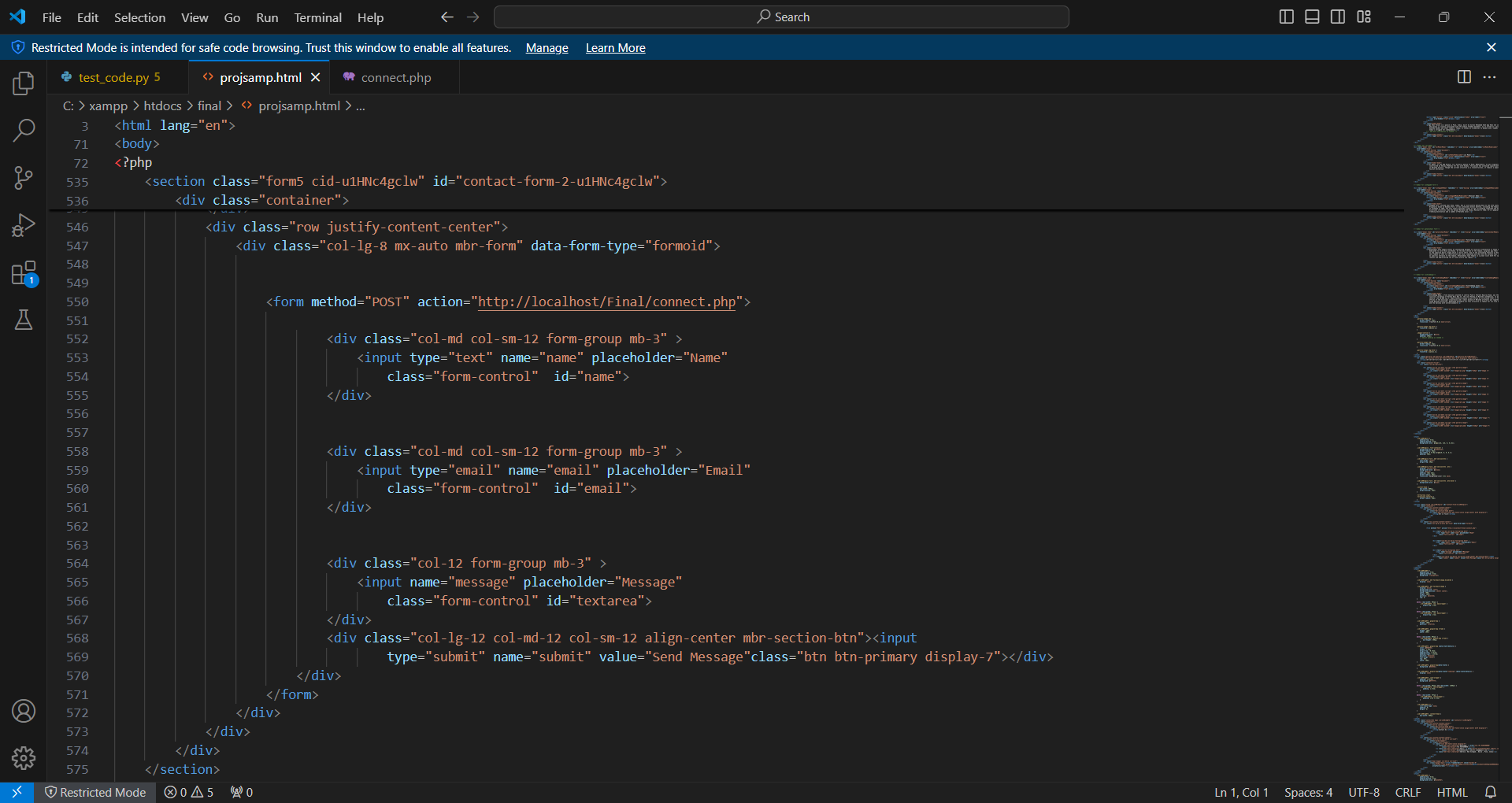
* **Scenario**: Run the script in different browsers (e.g., Firefox, Edge).
* **Expected Result**: The form behaves consistently across browsers.

**ScreenShots:- Html Page code:-**

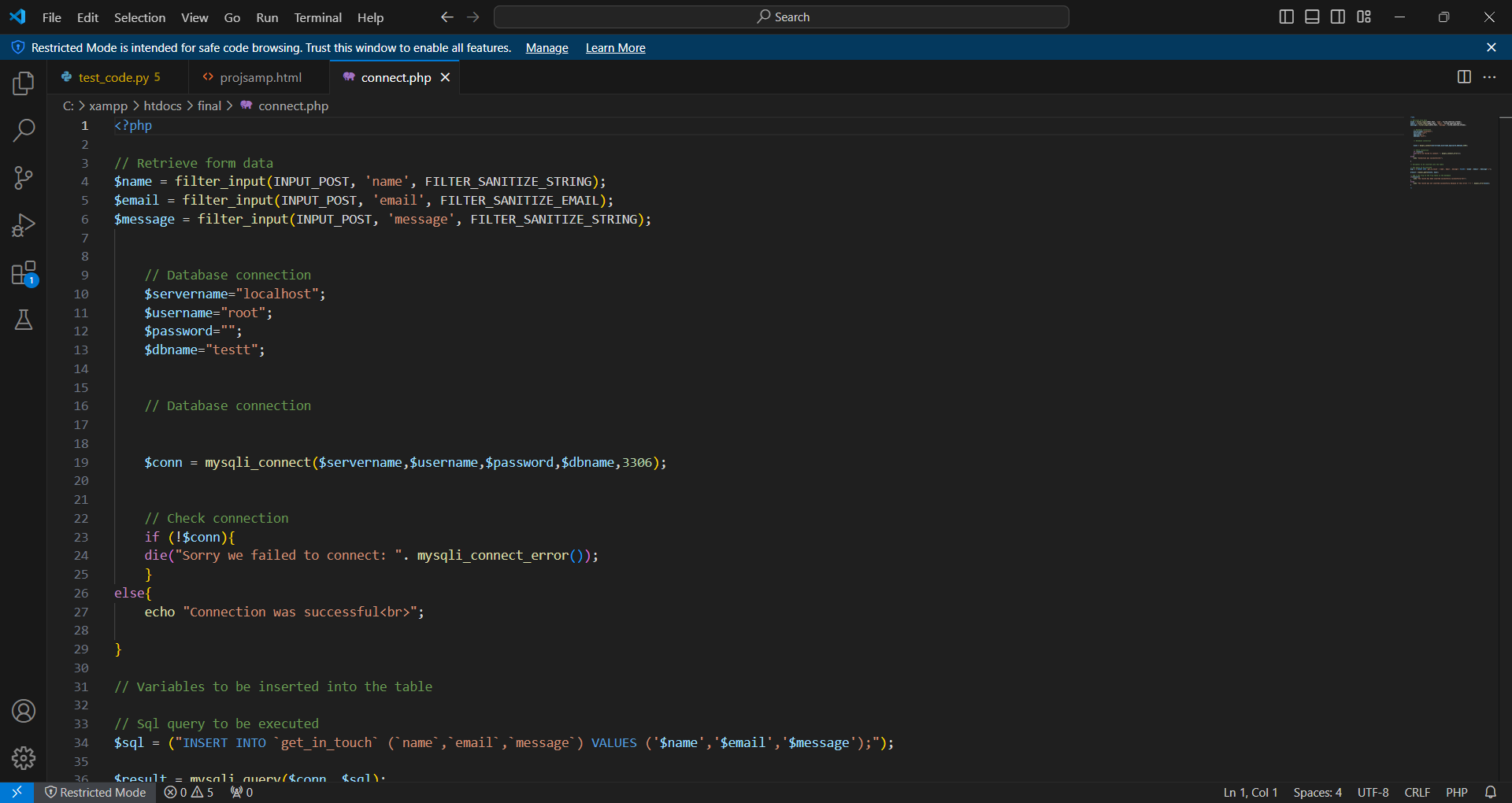


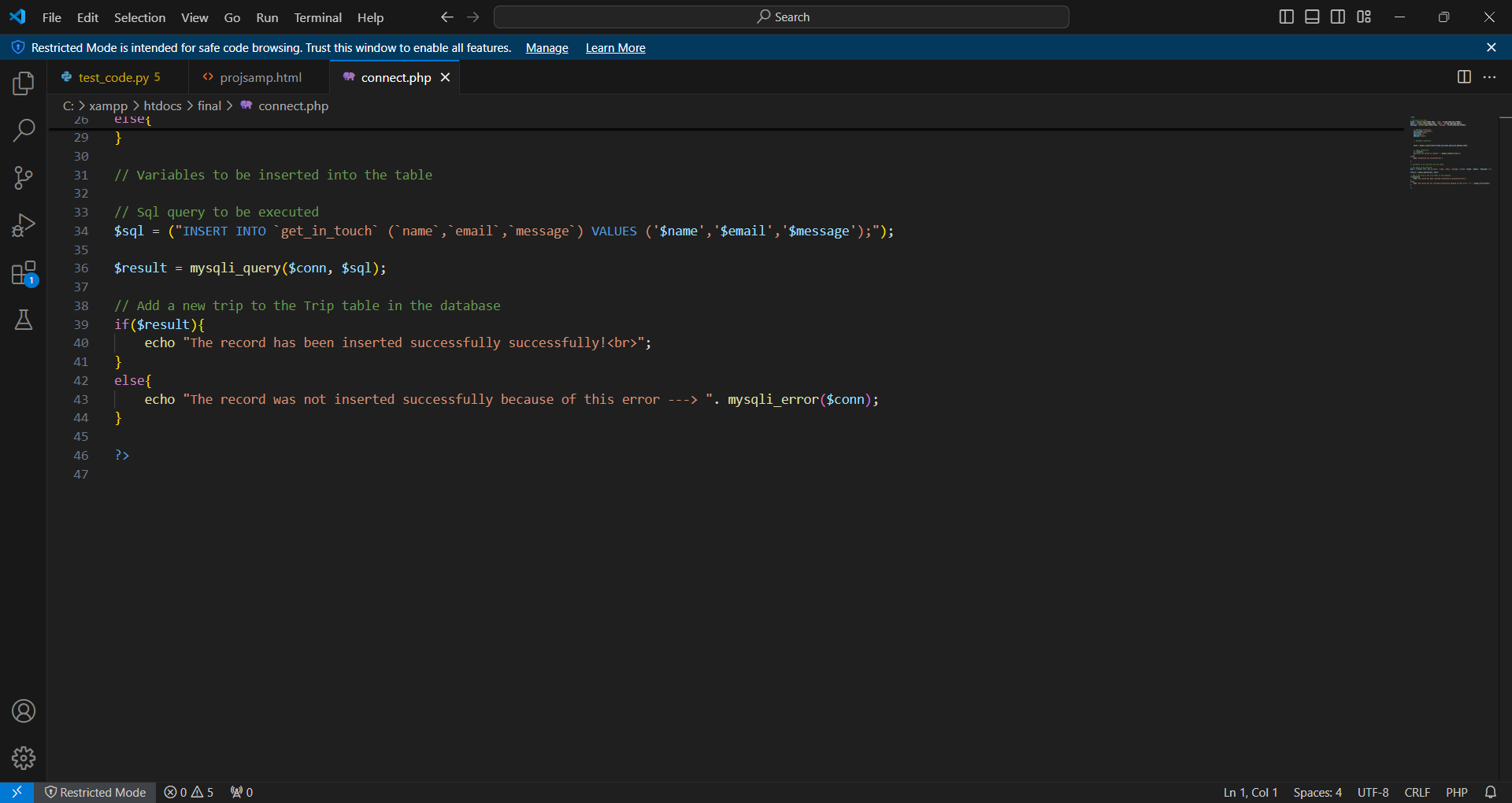




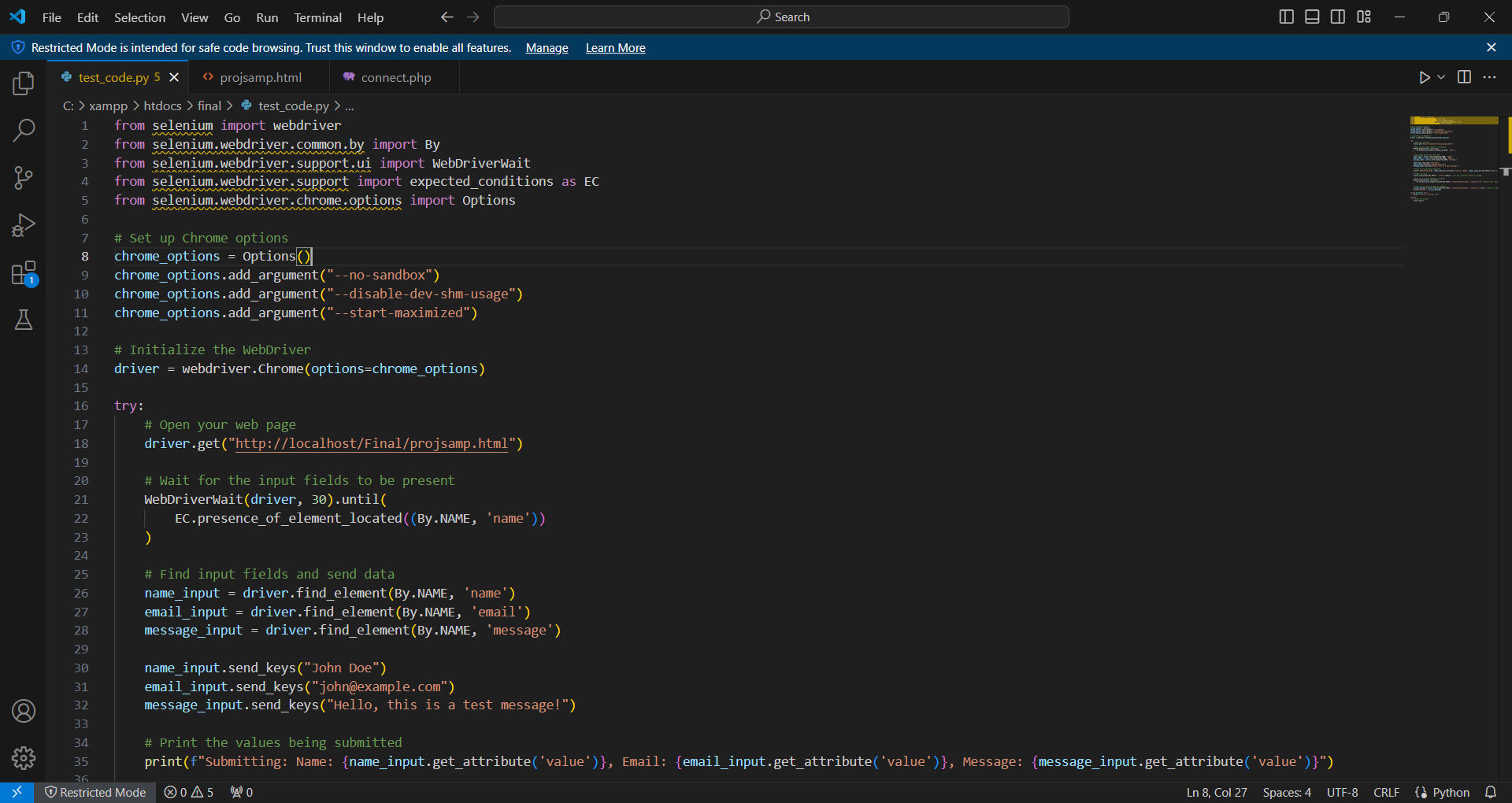
****

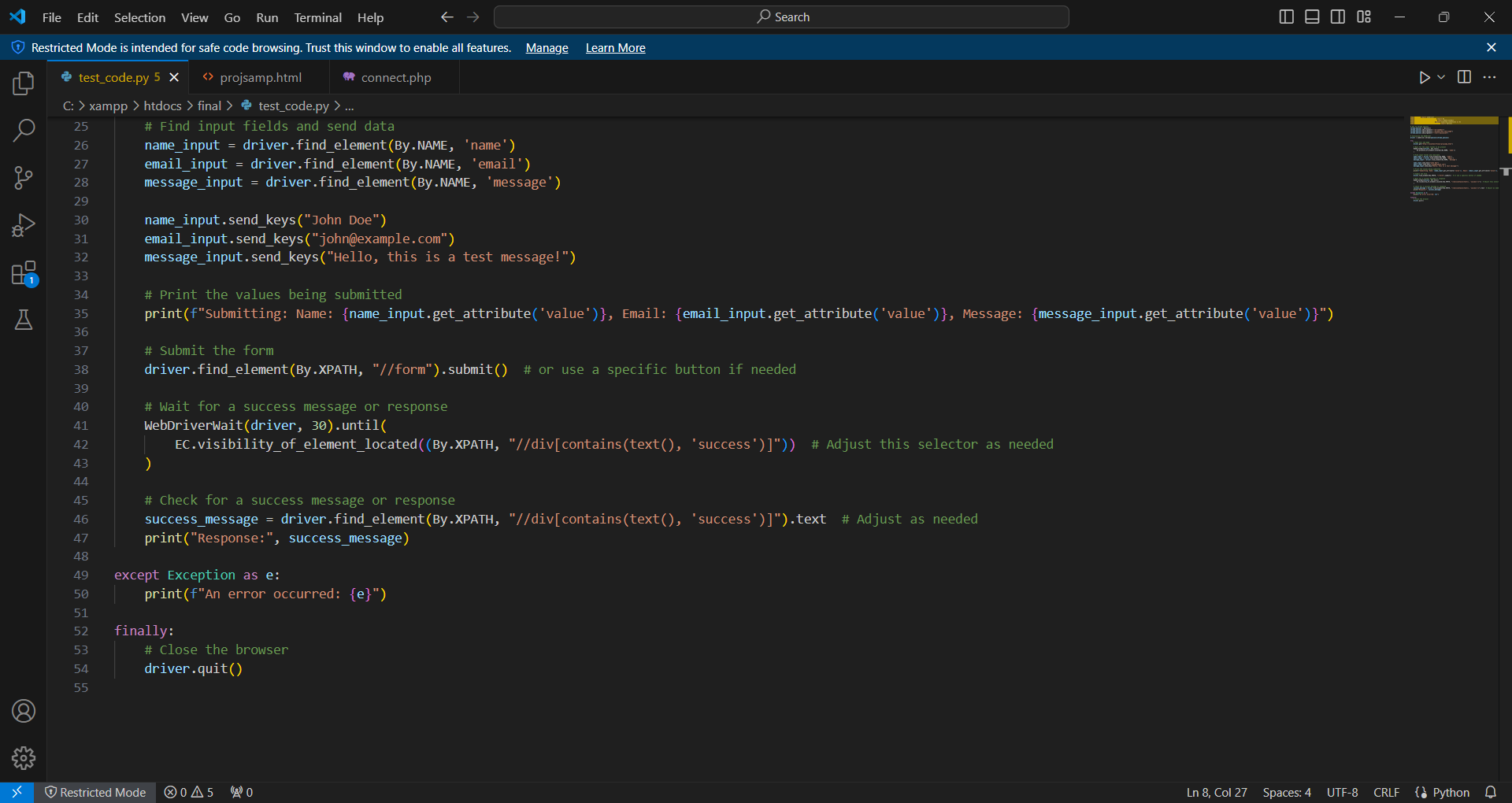
**PHP code:-**

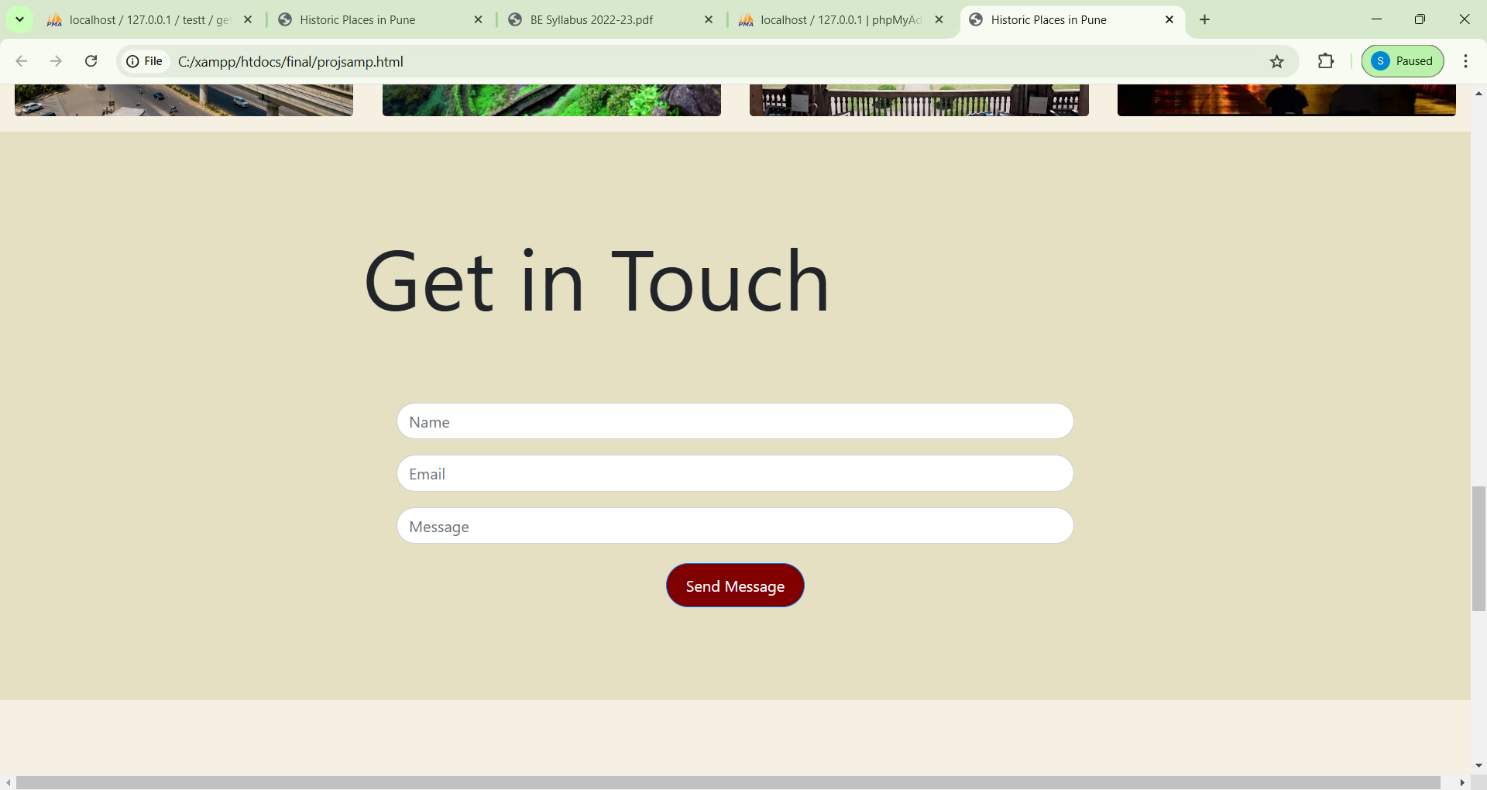


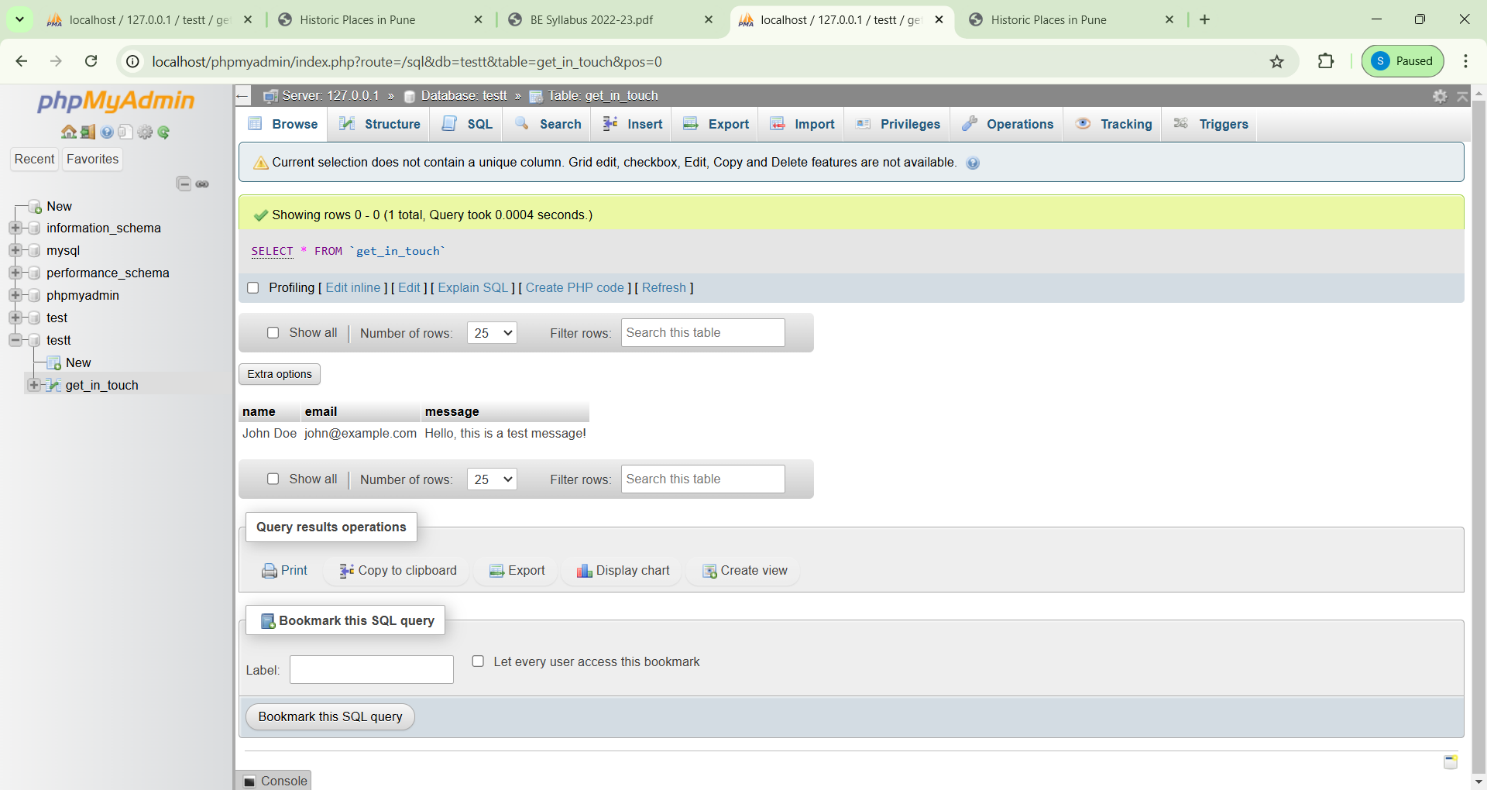


**Senelium Ide code:-**









|  |
| --- |
| **Conclusion:-**    The project successfully integrates multiple technologies to build a web-based application with a feasible working form system,. Testing, both manually and using Selenium WebDriver, ensures the functionality works as expected, identifying key bugs during form operations and database operations. With regular regression testing, the project remains robust and scalable, ensuring high performance and security. |