**Aiya Zhakupova SE-2439**

**Web Technologies 1(Frontend)**

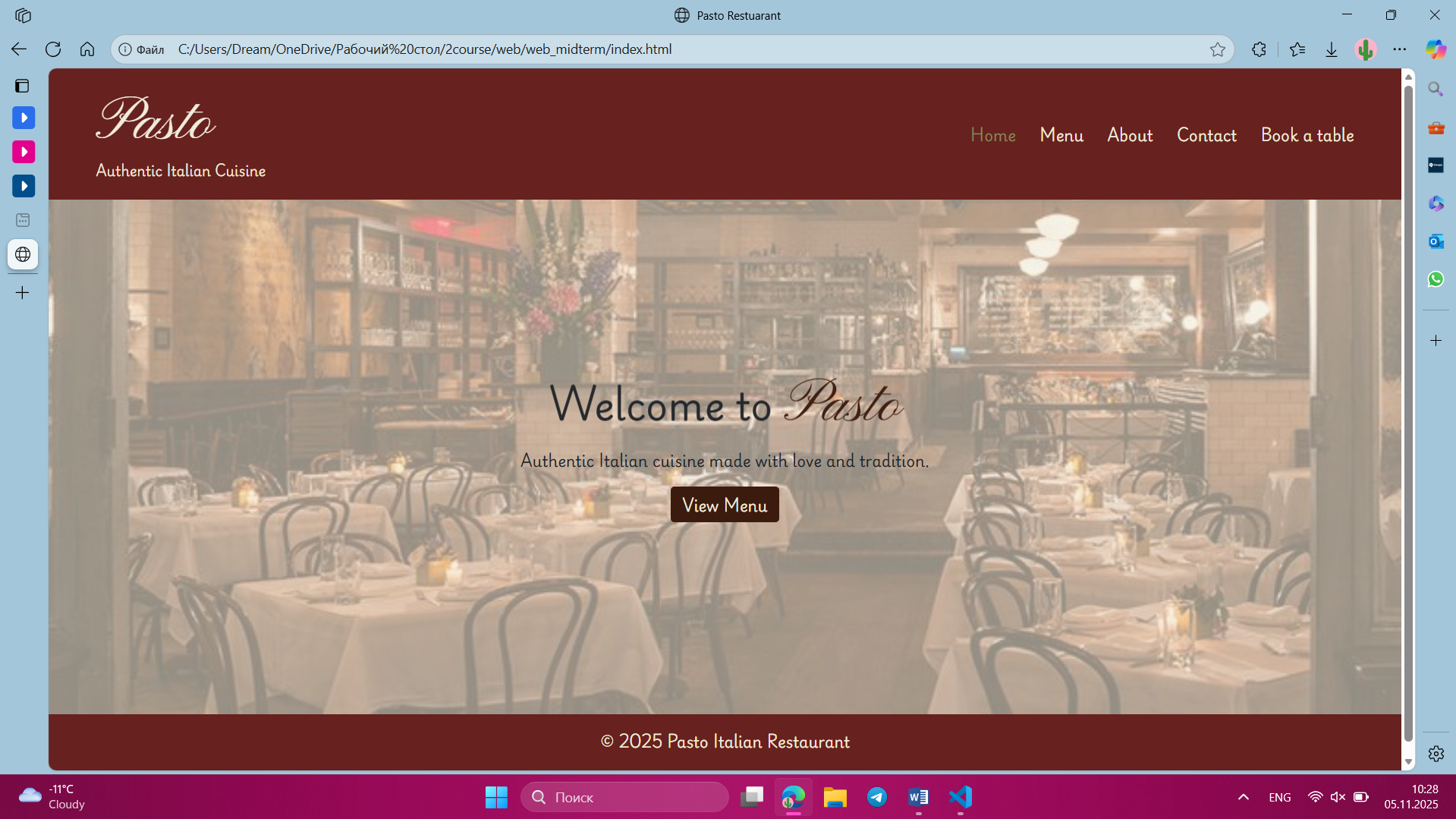
**Final project**

**Link:** [**https://aiya594.github.io/aitu-web-midterm/**](https://aiya594.github.io/aitu-web-midterm/)

**1.Home page**

When the document finishes loading, the script hides the body content and fades it in over one second. Buttons with the class .btn move slightly to the right when hovered over and return when the mouse leaves, creating a sliding effect.  
The navigation section is made dynamic: when a user clicks a navigation link, the script removes the active class from all links and adds it to the clicked one. This visually highlights the current section or page.

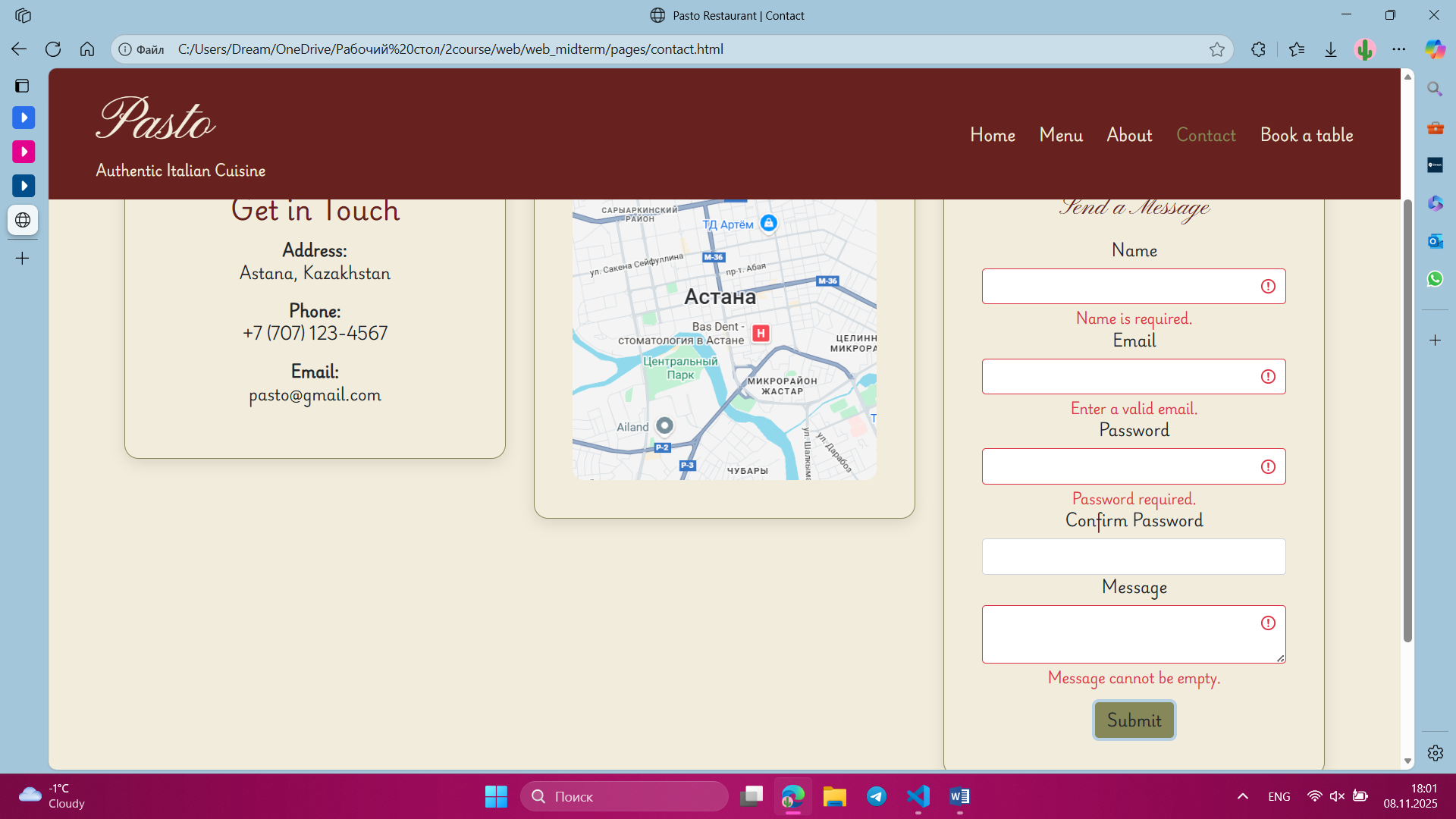
* **$(document).ready()** ensures the code runs only after the HTML structure is fully loaded.
* **$("body").fadeIn(1000)** creates a smooth page entrance.
* **.hover()** defines two animations: one for mouse-enter and one for mouse-leave.
* **.addClass() / .removeClass()** manage which navigation link appears active.



**2. Contact page**

When the form is submitted, the script prevents it from reloading the page and checks each input and textarea. Empty fields turn red, and valid ones remain styled in olive green. The email input is additionally tested against a regular expression to verify its format.  
If all data is correct, the script displays a green success message and resets the form. If not, it shows a red error message.

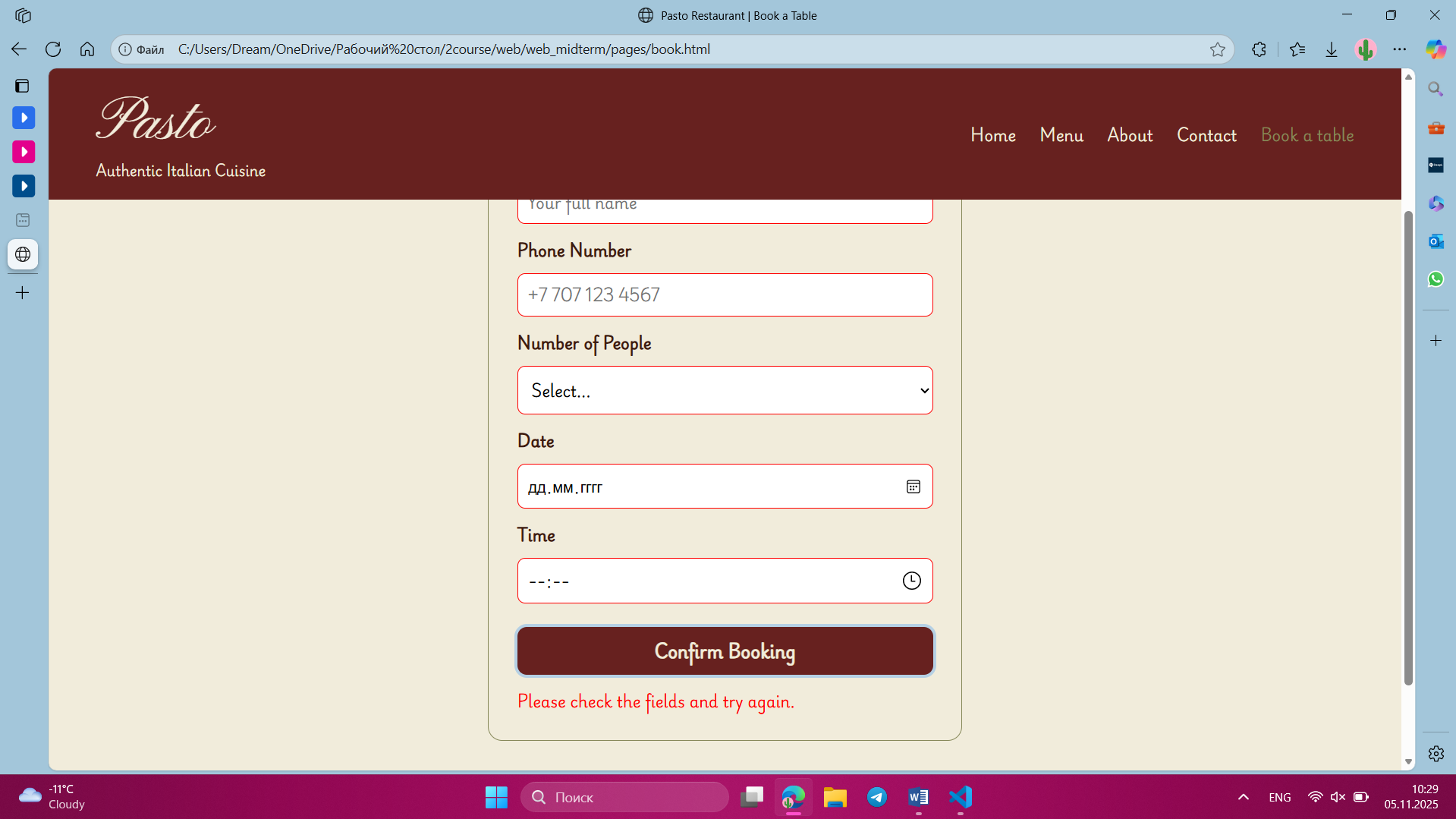
• e.preventDefault() stops the form’s default submission behavior.  
• .each() iterates over all input and textarea elements to check for empty values.  
• Regex pattern /^[^ ]+@[^ ]+\.[a-z]{2,3}$/ ensures the email has a valid format.  
• .fadeIn(), .delay(), .fadeOut() animate the display of feedback messages.  
• [0].reset() clears the form using the native DOM method.

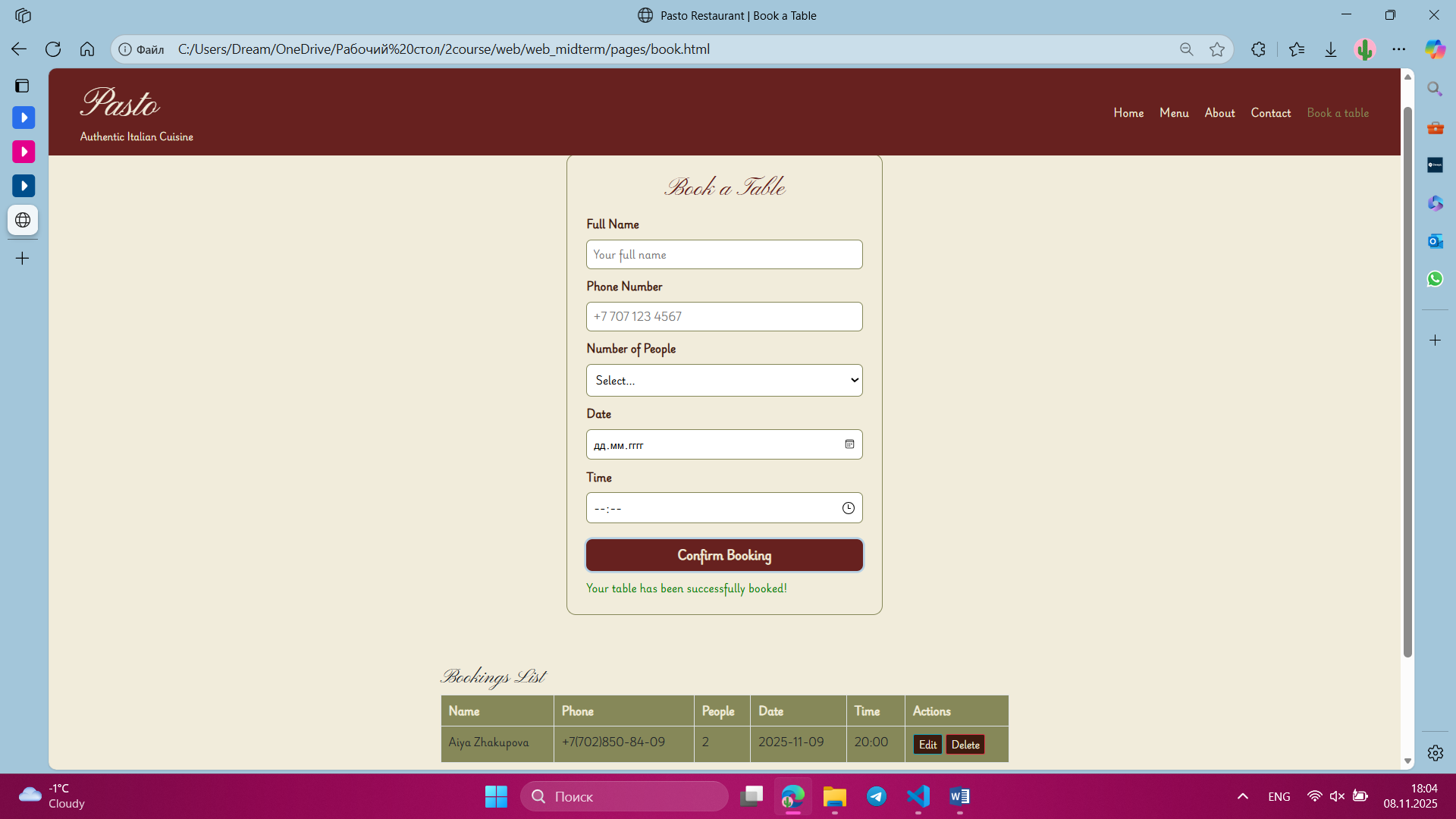


**3. Book page**

When the user submits the form, the script stops the default submission and checks all input and select elements. Empty fields get red borders, while valid fields are highlighted in green.  
It also compares the selected date with today’s date. If the chosen date is before today, the field is marked invalid. Based on the validation, a green success message or a red error message is shown.

• .find("input, select") selects all input and select elements inside the form.  
• new Date() converts the selected date into a JavaScript Date object for comparison.  
• .setHours(0, 0, 0, 0) removes the time portion to compare only the date.  
• Conditional check (selectedDate < today) prevents selecting past dates.  
• Animated feedback uses .fadeIn(), .delay(), .fadeOut() just like the contact form.





**4. Menu page**

The menu page allows users to search and filter menu items dynamically without reloading the page. The script handles user interactions with search input and filter buttons to show relevant menu items.

Search

//search menu item

    $("#searchInput").on("keyup", function () {

        const value = $(this).val().toLowerCase();

        $(".menu-card").each(function () {

            $(this).toggle($(this).text().toLowerCase().includes(value));

        });

    });

* + Purpose: Dynamically filters menu items based on the user’s input.
  + e.keyup event: Triggers every time the user types a character in the search input.
  + .val(): Retrieves the current value of the input field.
  + .toLowerCase(): Converts input and menu text to lowercase for case-insensitive search.
  + .each(): Iterates over each menu card to check if it contains the input text.
  + .toggle(): Shows or hides the menu card depending on whether the text matches.

Category Filtering

//filtering

    $(".menu-filter").click(function () {

        const filter = $(this).data("filter");

        if (filter === "all") {

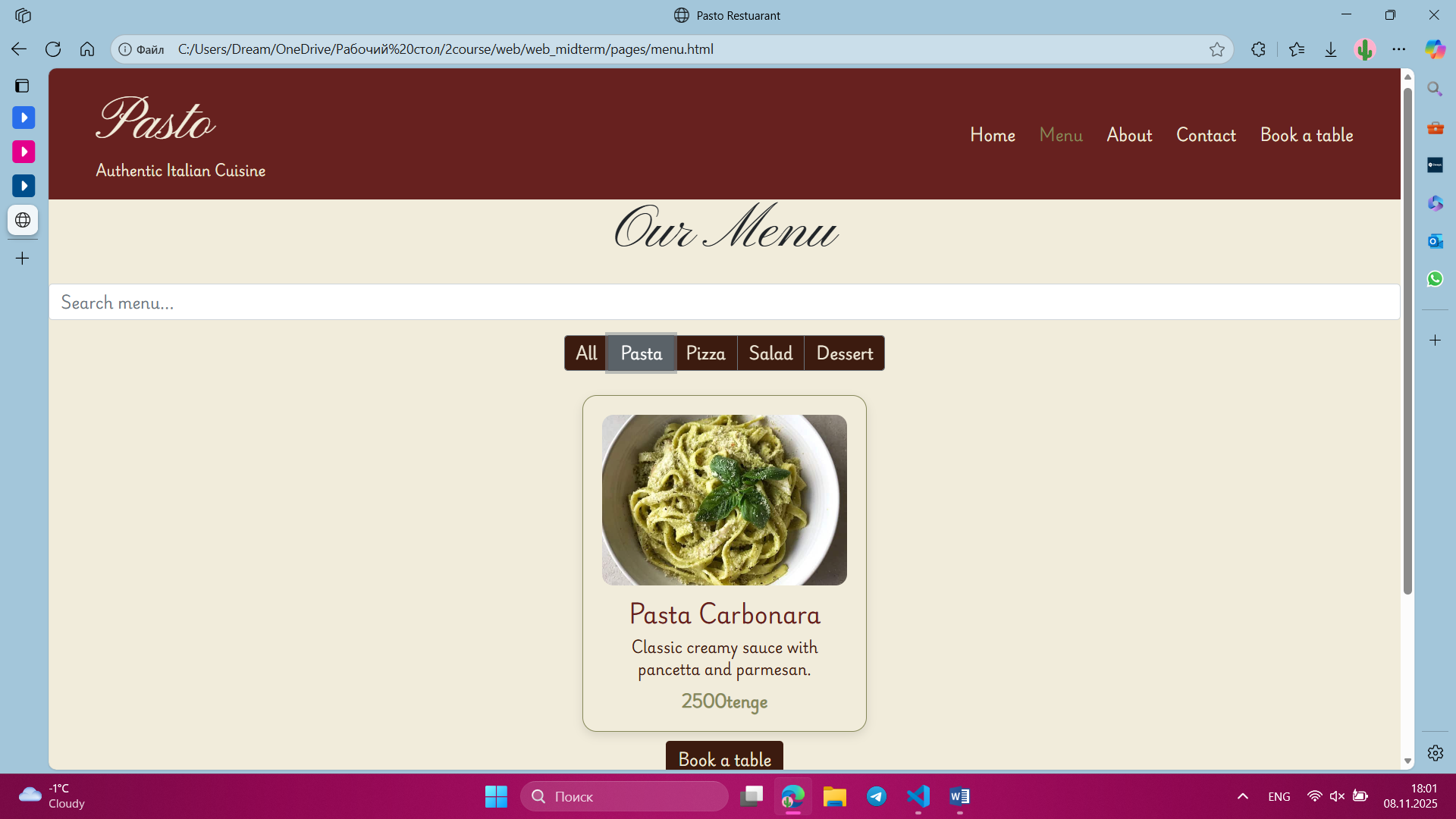
            $(".menu-card").show();

        } else {

            $(".menu-card").each(function () {

                $(this).toggle($(this).data("category") === filter);});}});

* + Purpose: Allows users to filter menu items by categories such as Pasta, Pizza, Salad, or Dessert.
  + .data("filter"): Retrieves the category value from the clicked button.
  + Conditional check:
    - "all" shows all menu cards.
    - Other categories only show menu cards whose data-category matches the selected filter.
  + .toggle(): Dynamically hides or shows each menu card based on the category

## **Summary**

The website provides an interactive and user-friendly experience across all pages. Each page uses jQuery to enhance usability: the Home page features smooth fade-in and dynamic navigation highlights; the Contact and Book pages include form validation with visual feedback and prevention of invalid submissions; the Menu page allows real-time searching and category filtering of dishes without reloading. Overall, the site combines clean design with responsive interactivity, ensuring users can easily navigate, find information, and engage with content efficiently.