### 5. TESTING AND IMPLEMENTATION

#### 5.5.1 Tools and Techniques Used

To ensure the accuracy and reliability of the application, the following tools and techniques were employed during the testing phase:

* **Manual Testing**: We thoroughly tested the form manually to verify that all input fields accept the correct data types. We implemented form validations using HTML5 and JavaScript to ensure that users provide correct values. This included checking for required fields, numeric inputs, valid email formats, and preventing submission when any field contained invalid data.
* **Browser DevTools**: We used browser developer tools (Chrome DevTools) to inspect and debug the front-end form. This helped us identify issues with the HTML structure, CSS styles, and JavaScript validation functions. It allowed us to test how the form behaves under different scenarios (e.g., submitting invalid data, leaving fields empty).
* **Flask Debugger**: Flask’s built-in debugging tool was used to monitor server-side operations during form submissions. This allowed us to identify and resolve any issues related to the backend processing of customer data and company recommendations.

By utilizing these testing tools and techniques, we ensured that the application’s user inputs and backend processes work correctly in real-time, improving both functionality and user experience.

#### 5.5.2 Implementation and Results

The implementation phase involved connecting the front-end form to the Flask backend and ensuring that data was processed as intended. Below are the key elements and outcomes of this phase:

* **Form Validation**: We implemented robust form validation using HTML5 and JavaScript to prevent incorrect or incomplete data from being submitted. This included specifying required fields, setting input types (such as numbers or text), and using JavaScript for additional checks (e.g., valid email formats). Error messages were displayed to users in real-time if invalid data was entered, enhancing the usability of the form.
* **Backend Processing**: Once the form was successfully submitted, the backend processed the customer’s risk profile and matched it with the appropriate companies from our Excel file. The Flask application was configured to read customer inputs, compute their risk score using pandas, and return a table of recommended companies based on the customer’s risk profile.
* **Results**: The results of the testing phase showed that all validation rules were functioning as expected. Screenshots were included in the report to demonstrate the error messages for incorrect inputs and successful submissions. After form submission, the application returned the expected results— a list of recommended companies matching the user's risk profile—demonstrating the reliability of the system.

These sections reflect the actual tools and techniques used in your project, emphasizing the successful implementation of form validation and backend processing.