# React Js Source Code Auditor

[1]P.Sai Tharun [2]M.Subhashini

*[1]CSE, NEC, Gudur*

*2Assistant Professor, CSE Department, NEC, Gudur*

***Abstract:*** In software development, two key challenges i.e., typo mistakes and coding consistency across developers takes the highest priority. An introduction for a React.js code auditor is presented here, a framework addressing these key challenges in an organization.When user run the tool after selecting target source code folder then expectation is system scan all the programs line by line, perform code review based on pre-configured parameters and publish the result in an excel file format.

This automation framework helps both developer and code auditor by sharing his work load.From developer perspective, once he/she is done with build then they can run this tool, get the results and fix it by themself without waiting for project code auditor team member time slot. From code auditor perspective everything explicitly documented in coding standards must be followed to align the code in technical standards which will get automated.

Code auditor can review the code only from functional GAP perspective. This approach will indirectly reduce the number of defects getting reduced from SIT, UAT phase.

***Keywords:*** *React JS, source code auditor,UTA phase, automate*

## I. INTRODUCTION

In the world of software development, two major hurdles stand out: typos and maintaining consistency in code among developers. Enter the React.js code auditor a solution designed to tackle these challenges head-on. This framework offers a streamlined approach to ensuring code quality and coherence within an organization. When a user runs the React.js code auditor and selects the target source code folder, the system swings into action. It meticulously scans every line of code, conducting a thorough review based on predefined parameters.

Once this analysis is complete, the results are compiled into an Excel file format, providing an easily digestible overview of the code's quality.

This automation framework serves as a boon for both developers and code auditors alike. From a developer's standpoint, the process is seamless – they can initiate the tool once they've completed their build, receive instant feedback, and address any issues independently, without the need to wait for a slot in the project code auditor's schedule. For code auditors, adherence to coding standards is paramount. The React.js code auditor ensures that all code follows these standards to the letter, aligning it with technical benchmarks automatically. By focusing their efforts on functional gaps within the code-base, auditors can efficiently pinpoint areas for improvement, ultimately leading to a reduction in defects,ultimately enchancing UTA phase.

## II.METHODOLOGY

**1. Define Requirements:**

**- Gather Detailed Requirements:** Collect requirements from users regarding the output format for the error reports, including file types (e.g., Excel, CSV), structure, and specific data to be included (filename, folder path, line number, remarks).

**2. Develop a User Interface (UI):**

**- Design User-Friendly Interface:** Create an intuitive interface for users to select the target folder, and specify the file name and format for the output report.

**- Include Validation Checks:**Implement validation to ensure users enter valid parameters, such as existing folder paths and acceptable file names/formats.

**3. Generate Output:**

**- Programming Languages and Tools:** Use Python with libraries like openpyxl for Excel output and pandas for CSV to generate the desired output based on user specifications.

**- Incorporate Data Processing Logic:** Develop logic to scan React.js files, detect errors, and format the data correctly before writing it to the destination file.

**4. Documentation:**

**- Methodology Documentation:** Detail the process of generating output files, including the supported formats and any dependencies or libraries used**.**

**- User Instructions:** Provide clear instructions on how to use the system to generate and save output files, including screenshots of the UI and step-by-step guidance.

**5. File Handling:**

**- Implement File Handling Mechanisms:** Ensure robust mechanisms for creating, writing, and saving the generated output file to the designated directory.

**- Error Handling:** Include error handling to manage potential issues during file creation or writing, ensuring users are informed of any problems and can take corrective action.

## III.IMPLEMENTATION

### 1. CG_F001- GUI Screen:

GUI stands for Graphical User Interface which is an interface between user and components like buttons, icons etc and it consists of browse button, reference folder path in text box and submit button. As a user, I want to have a GUI screen that allows me to easily process a folder and an Excel file. This screen should provide a straightforward and intuitive interface where I can select the folder and the Excel file, initiate the processing, and view the results in a clear and organized manner.

The goal is to streamline my workflow and make the handling of these files more efficient and user-friendly. With this GUI, I aim to save time and reduce the complexity involved in managing and processing data from multiple sources. In this scenario, the acceptance criteria require that when the parent code is provided and the run button is clicked, a GUI screen should immediately pop up. This GUI screen should be user-friendly and intuitive, facilitating the subsequent tasks the user needs to perform. The process should be seamless, ensuring that upon initiation by clicking the run button, the transition to the GUI screen is smooth and immediate. This functionality is crucial to provide an efficient user experience, enabling the user to proceed with their tasks without any delays or technical issues.


### 2.CG_F002- Option menu for source type:

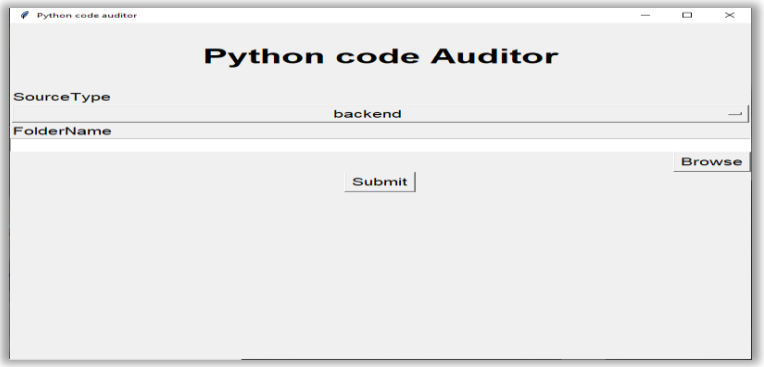
An Option menu is a feature that is used to display the list of use cases from the available drop down menu. As a user, I want to have an option menu integrated into the system so that I can easily select the desired use case. This menu should present a list of available options in a clear and organized manner, allowing me to quickly choose the specific task or function I need to perform.

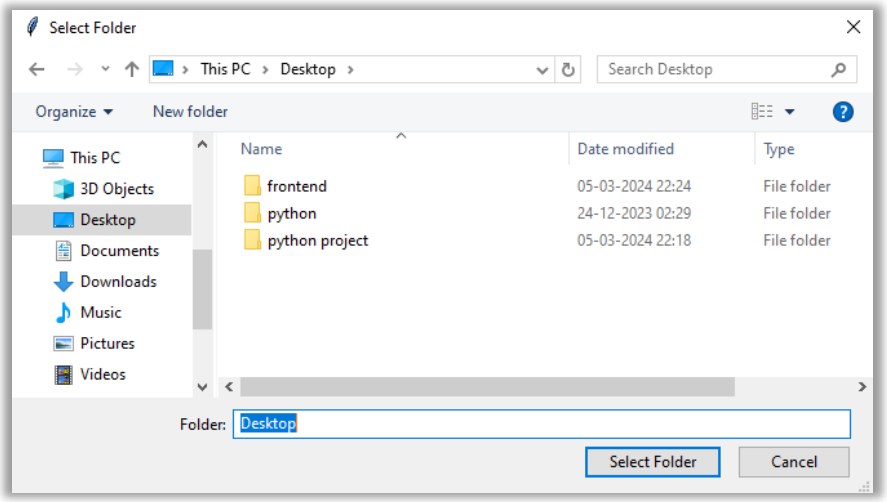
By providing a straightforward way to navigate between different use cases, the option menu will enhance my overall experience, making it more efficient and user-friendly. This feature is essential for streamlining my workflow and ensuring that I can access the necessary functions with minimal effort. In this scenario, the acceptance criteria specify that when the GUI screen is active and the option menu is clicked, the use case options should be displayed promptly. The option menu must effectively showcase a list of available use cases in a clear and accessible manner.

**3.CG_F003- Browse and Select Folder:**

• This browse button is to browse a folder in the local system

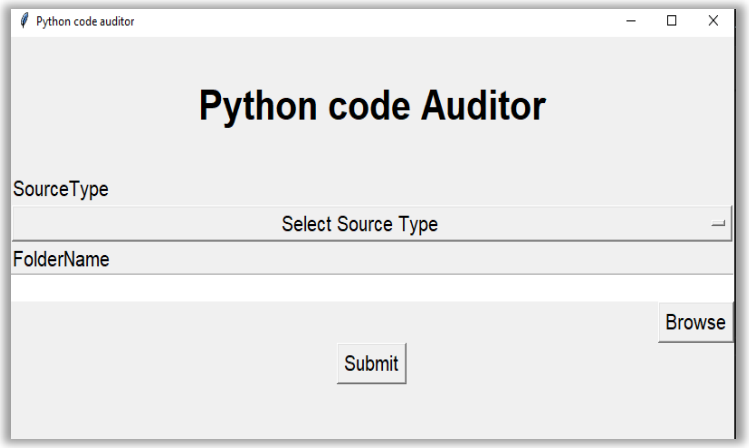
As a user, I want to have the ability to browse a folder from my local file system and display the folder path in a "Source" name entry label. This functionality will enable me to select and process the necessary folder and associated Excel file efficiently. By integrating a folder browsing feature with a clear and visible path display, I can ensure that I am working with the correct files
. This will streamline my workflow, reduce errors, and make the task of managing and processing files more user-friendly and straightforward. In this scenario, the acceptance criteria outline the requirement for a GUI screen that facilitates browsing the local file system through a file dialog window.
The user should be able to initiate this action seamlessly. When the user wishes to browse a folder from their local file system, the interface must respond promptly by displaying a file dialog window. This window will allow the user to navigate through their directories and select the desired folder. The functionality ensures that the user can easily and efficiently locate and choose the necessary folder, which is critical for subsequent processing tasks.



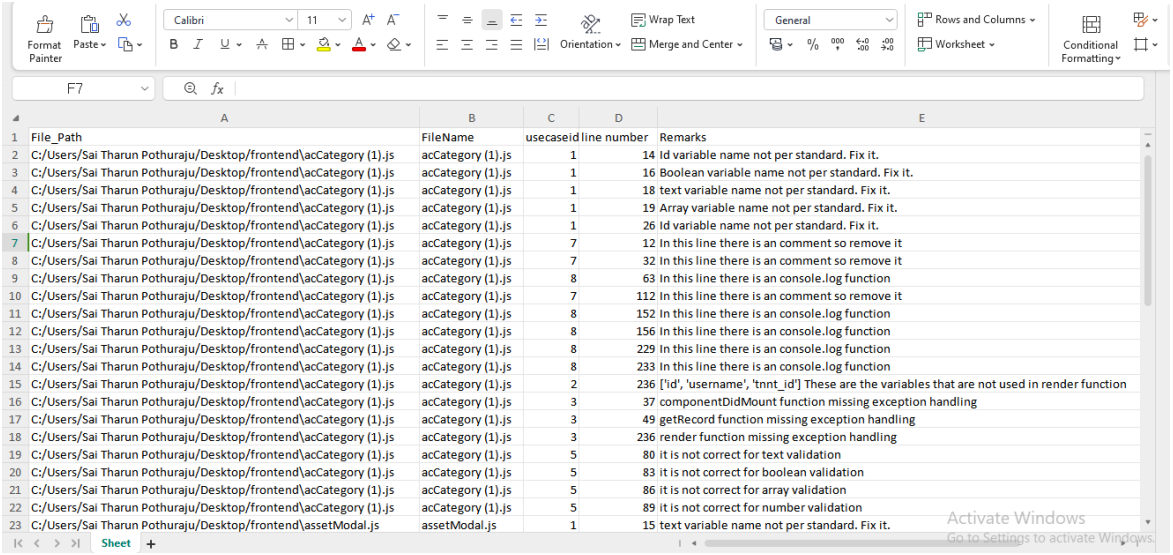

**4.CG_F004- Validation for Submit Button:**

Submit button is a feature, which helps the user for validating the input files(folder, excel file and values in the input excel, etc. In this scenario, the acceptance criteria outline the need for the GUI to handle situations where a user attempts to submit a selected folder without having chosen an Excel file
.
When the user provides a folder but does not select an Excel file, and then clicks on the submit button, the system should respond by displaying a pop-up message box. This message box should clearly inform the user that no Excel file has been selected, guiding them to rectify the oversight before proceeding.




## IV.RESULT

If Source type and folder name are same and folder having js files then the python code editor should read the code in the selected folder having js files and it need to generate an excel having fields of filepath,filename,line number,usecase id, remarks.



| File_Path | FileName | usecaseid | line number | Remarks |
|---|---|---|---|---|
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 1 | 14 | Id variable name not per standard. Fix it. |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 1 | 16 | Boolean variable name not per standard. Fix it. |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 1 | 18 | text variable name not per standard. Fix it. |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 1 | 19 | Array variable name not per standard. Fix it. |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 1 | 26 | Id variable name not per standard. Fix it. |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 7 | 12 | In this line there is an comment so remove it |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 7 | 32 | In this line there is an comment so remove it |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 8 | 63 | In this line there is an console.log function |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 7 | 112 | In this line there is an comment so remove it |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 8 | 152 | In this line there is an console.log function |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 8 | 156 | In this line there is an console.log function |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 8 | 229 | In this line there is an console.log function |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 8 | 233 | In this line there is an console.log function |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 2 | 236 | ['id', 'username', 'tnnt_id'] These are the variables that are not used in render function |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 3 | 37 | componentDidMount function missing exception handling |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 3 | 49 | getRecord function missing exception handling |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 3 | 236 | render function missing exception handling |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 5 | 80 | it is not correct for text validation |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 5 | 83 | it is not correct for boolean validation |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 5 | 86 | it is not correct for array validation |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\acCategory (1).js | acCategory (1).js | 5 | 89 | it is not correct for number validation |
| C:/Users/Sai Tharun Pothuraju/Desktop/frontend\assetModal.js | assetModal.js | 1 | 15 | text variable name not per standard. Fix it. |

## V.CONCLUSION

The React.js code auditor effectively addresses typographical errors and coding consistency in software development. By automating code reviews based on predefined parameters, it ensures high-quality code and generates comprehensive Excel reports. Developers benefit from immediate feedback, allowing for quick corrections, while code auditors can focus on functional discrepancies. This tool enhances overall development efficiency and reduces defects during System Integration Testing (SIT) and User Acceptance Testing (UAT).

.

## VI.REFERENCES

[1] Shari Lawrence Pfleeger. Software Engineering: Theory and Practice[M]. 2nd edition. Prentice Hall, Inc. 2001
[2] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software[M]. Addison Wesley, 1995.
[3] Krzysztof Czarnecki, Hlrich W.Eisenecker. Generative Programming Methods, Tools, and Applications[M]. Addison Wesley, 2000.
[4] Stewart Baird. Teach Yourself Extreme Programming in 24 Hours[M]. SAMS, 2003.
[5] Andrew Haigh. Obejct-Oriented Analysis & Design[M]. McGraw-Hill, 2001.
[6] John Swanco. COM Programming by Example[M]. R&D Books, 2000.
[7] Pradeep Tapadiya. COM+ Programming[M].Prentice Hall, 2002
[8] Bruce Eckel. Thinking in C++[M]. Prentice Hall, 1995.