

# Code Smells

## Project Overview

Our project aims to improve code quality in open-source repositories by detecting code smells in JavaScript and Python. The tool is divided into two releases:

- Release-1 (R1): Code smell detector
- Release-2 (R2): Automatic and semi-automatic refactoring suggestions

Release-1 focuses on detecting various code smells through a Chrome extension that integrates directly into GitHub. This enables developers to identify and analyze problematic code patterns easily.


## Features

- **Browser-Based Detection:** A local Chrome extension that activates on GitHub repository pages.
- **Interactive UI:** A small icon appears next to each folder in a repo. Clicking it triggers smell detection.
- **Language Support:** Supports both JavaScript and Python codebases.
- **Smell Highlighting:** Generates a list of detected smells with file-wise and folder-wise breakdowns.
- **Smell Report Storage:** To view all the smells in a repo, they are stored in `code_smells_report.json` file in the code smells root directory.

## Code Smells Detected

- **Javascript-** Long Functions, Too Many Parameters, Large Files, Console Log Overuse, Deep Nesting, Magic Numbers, Duplicate Code, Unused Variables, Callback Hell, Long Chained Calls, Inconsistent Naming, Low Comment Density, Empty Catch Blocks, Unnecessary Semicolons
- **Python-** High Complexity Functions, Low Maintainability Index, Large Files, Deeply Nested Functions, Large Functions, Feature Envy, Data Clumps, Dead Code (Unused Variables), Shotgun Surgery, Long Lambdas, Useless Exceptions, Duplicate Code, Large Classes, Too Many Returns, Global Variables, Large Parameters

## Running the Tool:

- Run the backend- `python server.py`
- Open google chrome and goto:
  - `chrome://extensions/`
  - Toggle Developer Mode (top right corner)
  - Click "Load Unpacked" (top left)
  - Select the folder named: `code-smell-extension`
- Use the Extension on GitHub:
  - Open any public GitHub repository in a new tab.
  - Click on the extension icon (toggle it ON if needed).
  - A small  (info icon) will appear next to each file in the repo.
  - Click the icon to view all detected code smells for that folder.

## Methodology & Techniques

- **Frontend (Chrome Extension):**
  - Injects a content script on GitHub.
  - Detects folder structure dynamically and attaches a clickable icon.
  - Sends repo URL and file paths to the backend.
- **Backend (Python API):**
  - Clones GitHub repo locally.
  - Parses `.js` and `.py` files using:
    - **AST (Abstract Syntax Tree)** analysis for Python.
    - **ESLint/Custom Parsers** for JavaScript.
  - Applies rule-based logic and heuristics to detect code smells.
  - Aggregates results and sends them back to the extension.
- **Smell Detection Techniques:**
  - **Cyclomatic Complexity** analysis
  - **Line and Character Thresholds** for file/function size
  - **Pattern Matching** for console logs, deep nesting, magic numbers
  - **Name Pattern Checks** for inconsistent naming
  - **Structural Analysis** using AST for nesting, dead code, returns

## **Contributions:(Smells detected and Chrome extension)**

- Akshatha RH (CS22B003)- (for js) Duplicate Code, Too Many Parameters, Unused Variables smells, Low Comment Density, Empty Catch Blocks, Global variables
- A Sai Preethika (CS22B006)- (for py) Long lambdas, Useless exceptions ,Duplicate code ,Large classes, Too many returns, Display of smell info on icon click
- Ch Aarya (CS22B018)- (for py) High complexity functions, Low maintainability, Large file, Globalvariables, Large parameters smells, Integrated code to chrome extension
- K Sanjay Varshith (CS22B029)- (for py) Deeply nested functions, Large functions, Feature envy, Data clumps, Dead code variables, Display of info icon beside smelly files
- K Akhil Solomon (CS22B032)- Callback Hell, Long Chained Calls , Inconsistent Naming smell, Shotgun surgery smell, Unnecessary Semicolons smell, Useless exceptions smell
- M Akash (CS22B037)- Long Function smell, Large File smell, Console Log Overuse smell, Deep Nesting smell, Magic Numbers smell, Dead code variables smell