Team Contributions: POC
Software Engineering

Team 4, EcoOptimizers

Nivetha Kuruparan
Sevhena Walker
Tanveer Brar
Mya Hussain
Ayushi Amin

This document summarizes the contributions of each team member up to the POC Demo. The time period of interest is the time between the beginning of the term and the POC demo.

1 Demo Plans

For our proof of concept demonstration, we will showcase the core functionality of our energy-efficient Python code refactoring tool. The demonstration will focus on the following key aspects:

1. **Code Smell Detection:** We will show how we used Pylint to identify inefficient code patterns (code smells) in Python source code that may lead to higher energy consumption.

2. **Refactoring:** Using the Rope library, we'll demonstrate how our tool will apply refactorings to address the detected code smells.

3. **Energy Consumption Measurement:** We will show how we utilized CodeCarbon to measure and compare the energy consumption of the original code versus the refactored version.

4. **Functionality Preservation:** We will demonstrate that the refactored code maintains its original functionality by running the original test suite against both versions of the code.

5. **Performance Metrics:** We will display performance reports comparing the original and refactored code, highlighting improvements in energy efficiency.

2 Team Meeting Attendance

Student	Meetings
Total	11
Sevhena Walker	11
Nivetha Kuruparan	11
Tanveer Brar	11
Mya Hussain	11
Ayushi Amin	11

We aim to have all team members present for any team meetings we hold. If one person cannot attend the meeting, we usually reschedule to a different time.

3 Supervisor/Stakeholder Meeting Attendance

Student	Meetings
Total	4
Sevhena Walker	4
Nivetha Kuruparan	4
Tanveer Brar	4
Mya Hussain	4
Ayushi Amin	4

We aim to have all team members present for any meetings we have with our supervisor. If one person cannot attend the meeting, we usually reschedule to a different time.

4 Lecture Attendance

Student	Lectures
Total	12
Sevhena Walker	11
Nivetha Kuruparan	8
Tanveer Brar	8
Mya Hussain	6
Ayushi Amin	6

We aim to have at least one team member present for lectures to make sure we don't miss any critical information regarding deliverables.

5 TA Document Discussion Attendance

Student	Lectures
Total	3
Sevhena Walker	3
Nivetha Kuruparan	3
Tanveer Brar	3
Mya Hussain	3
Ayushi Amin	3

6 Commits

Student	Commits	Percent
Total	396	100%
Ayushi Amin	89	23%
Tanveer Brar	44	11%
Mya Hussain	59	15%
Sevhena Walker	156	39%
Nivetha Kuruparan	48	12%

Some people might have higher commit counts because they commit more frequently or make smaller, more granular changes, whereas others might commit less often with larger, consolidated changes. Additionally some group members squash and merge when merging PRs and others forget sometimes.

7 Issue Tracker

Student	Authored (O+C)	Assigned (C only)
Sevhena Walker	25	24
Mya Hussain	10	19
Nivetha Kurparan	18	23
Tanveer Brar	9	20
Ayushi Amin	12	21

The numbers in the **Assigned** column give a better picture of each team members contribution. Many commits were sometimes authored by the same person due to differences in team responsibilities (logistics and management). Furthermore, the issues here refer to what we can call "work" issues. Issues with the `lecture`, `team-meeting`, `sup-meeting`, and `ta-meeting` labels are not included in this tally.

8 CICD

The section outlines the plan to include CI/CD for this project. The plan will streamline development, testing and deployment processes, while ensuring consistent performance improvements.

8.1 Source Control and Branching Strategy

- **Repository Setup**: Code is hosted on GitHub for version control and collaboration.
- **Branching Strategy**:
  - `main`: Production-ready code.
  - `dev`: Primary development branch.
  - `docs`: Feature branch of dev that is meant for documentation commits.

  Based on deliverables, temporary branches are created on team and individual level and discarded once merged into one of the above branches. In future, `dev` will be diverged into multiple feature branches for initial commits that are eventually merged into it. These include component specific branches such as `refactoring`, `analyser`, `energy`, `test` and `plugin`.
- **Merging Policy**: All pull requests should have at least two reviews before merging, as outlined in the Development Plan.

8.2 Build and Testing Pipeline

GitHub Actions will be used for CI/CD to automate testing and code analysis on pull requests. They will include the following:

- **Build Steps**
  - **Static Code Analysis & Linting**: `PyLint` will be used to handle both code smells for static analysis and enforce PEP 8 style guide.
- **Testing**:
  - **Unit Tests**: Unit tests will be written using `PyTest`.

  - **Code Coverage**: Test code coverage will be tracked using `coverage.py`.
  - **Performance Testing**: Metrics such as memory usage and execution time will be tracked using `cProfile`.

8.3 Continuous Deployment

With every stable version, the product will need to be continuously deployed.

- **Environment Setup**: To standardize environment settings across platforms, Docker containers will be used.
- **Deployment**:
  - **Refactoring Library**: The library will be rebuilt and updated on its public facing source.
  - **VS Code Extension**: With each update to main branch, the VS Code extension will automatically be built and updated on its public facing link.

9 Additional Productivity Metrics

The team does not have any additional metrics of productivity.