MAPS PLAGIARISM DETECTOR

Team-114

Sakshi Tonwer

Preethi Anbunathan

Mateo Kockaj

Manthan Thakker

Problem

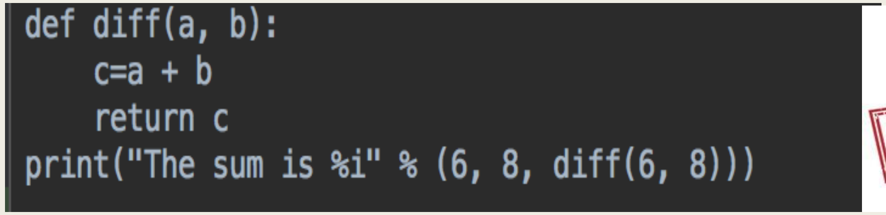
- Traditional Systems Fail at Detecting plagiarism in programming classes.
- Not so many ways to detect plagiarism for CS courses especially in coding assignments.
- Algorithms exist on the web but no such specific user friendly systems accessible to professors to manage as well as detect plagiarism in real time.

System Functionality – Achieved

- Plagiarism in different python files are detected.
- The system has been compared against MOSS and a weighted polynomial function is proposed.
- The following types of plagiarism in the code are identified:
  - ❑ *Method name Check*
  - ❑ *Variable name Check*
  - ❑ *Class name Check*
  - ❑ *Code Shift Comparison*

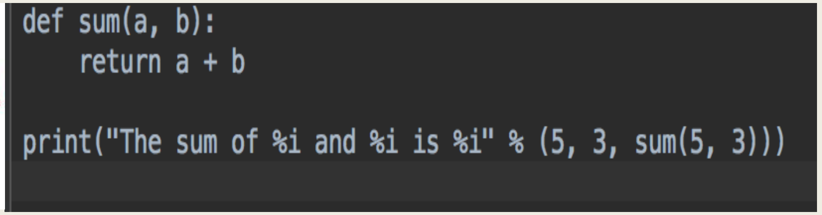
Types of Plagiarism

Variable name Changes:



```
def diff(a, b):
    c=a + b
    return c
print("The sum is %i" % (6, 8, diff(6, 8)))
```

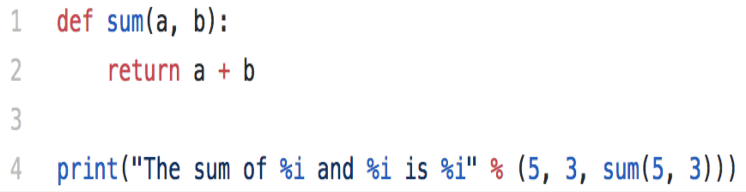
PLAGIARISM



```
def sum(a, b):
    return a + b

print("The sum of %i and %i is %i" % (5, 3, sum(5, 3)))
```

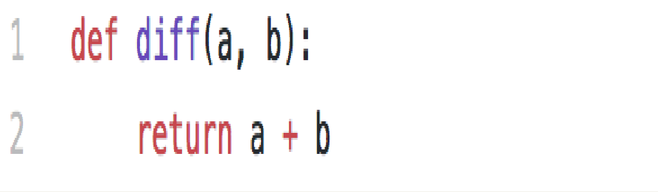
Function name changes:



```
def sum(a, b):
    return a + b

print("The sum of %i and %i is %i" % (5, 3, sum(5, 3)))
```

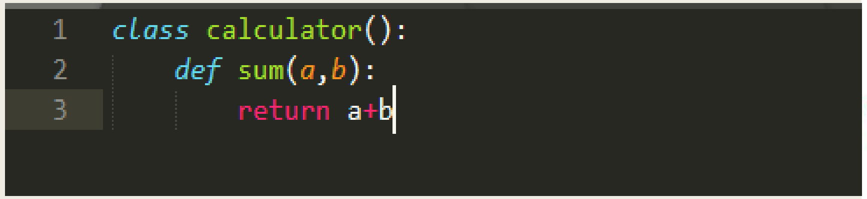
PLAGIARISM



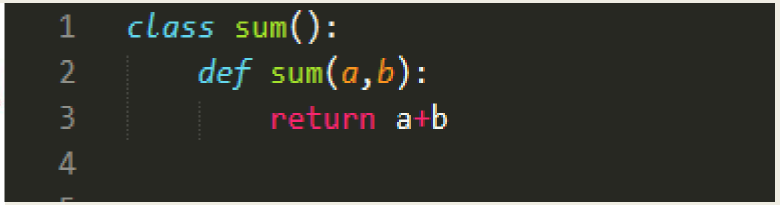
```
def diff(a, b):
    return a + b
```

Types of Plagiarism

Class name Changes:



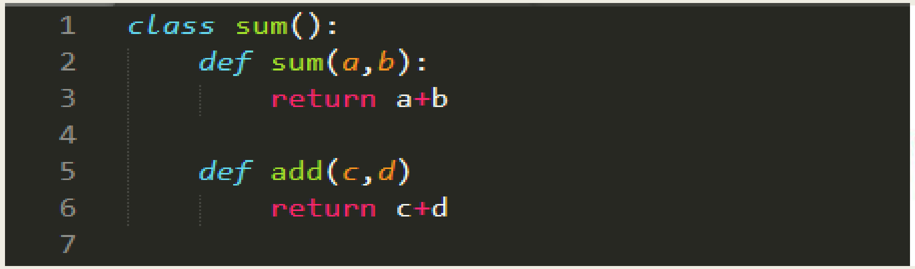
```python
class calculator():
    def sum(a,b):
        return a+b
```



```python
class sum():
    def sum(a,b):
        return a+b

```

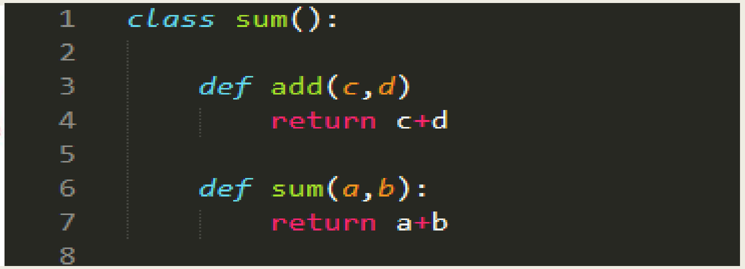
Code Shift:



```python
class sum():
    def sum(a,b):
        return a+b

    def add(c,d)
        return c+d

```



```python
class sum():

    def add(c,d)
        return c+d

    def sum(a,b):
        return a+b

```

System Functionality - Achieved

- System is integrated with GitHub, the python files can be fetched using the GitHub repository URL.
- In addition, three thresholds have been set to categorize the plagiarized files into:
  - ❑ High plagiarized – x >= 90%
  - ❑ Medium plagiarized - 75% <= x < 90%
  - ❑ Low plagiarized – x < 75%
- Side by side comparison of the plagiarized files are done, which will highlight the plagiarized contents in python files.
- History of all generated results are stored in a database and hence the report can be viewed at any point.

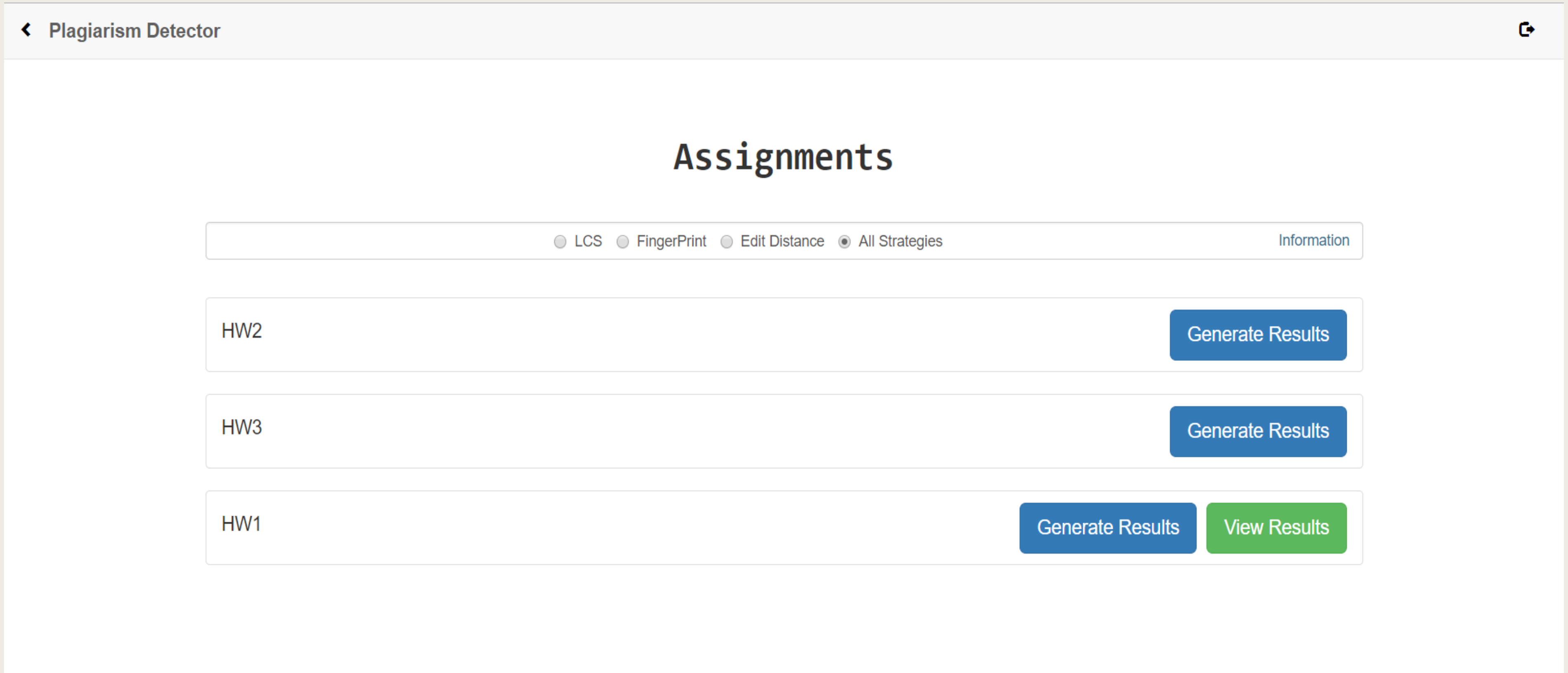
System Functionality – Not Achieved

- We plan on implementing the feature of downloading the generated report.
- Yet to implement a feature of deleting reports.

Application Usage

- Clients can install the system and detect plagiarism across python files using it.
- The reports generated are accurate as the weighted polynomial function matches the MOSS standards.
- The results are color coded based on the plagiarized contents present, facilitating ease of use.
- Client is guided throughout the use of application, as we have provided proper help information on every page.

Achieved System Functionalities - Screenshots



Plagiarism Detector

Assignments

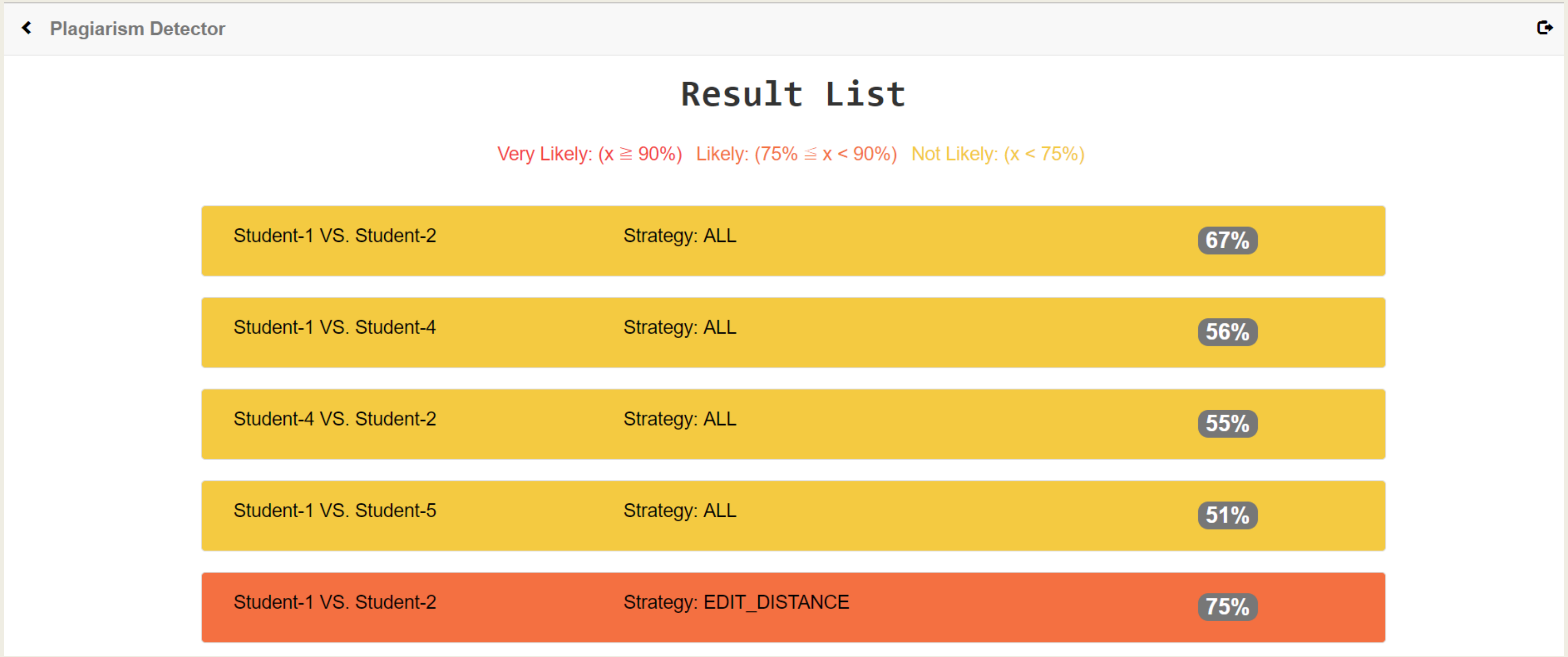
○ LCS ○ FingerPrint ○ Edit Distance ● All Strategies

HW2

HW3

HW1

Achieved System Functionalities - Screenshots

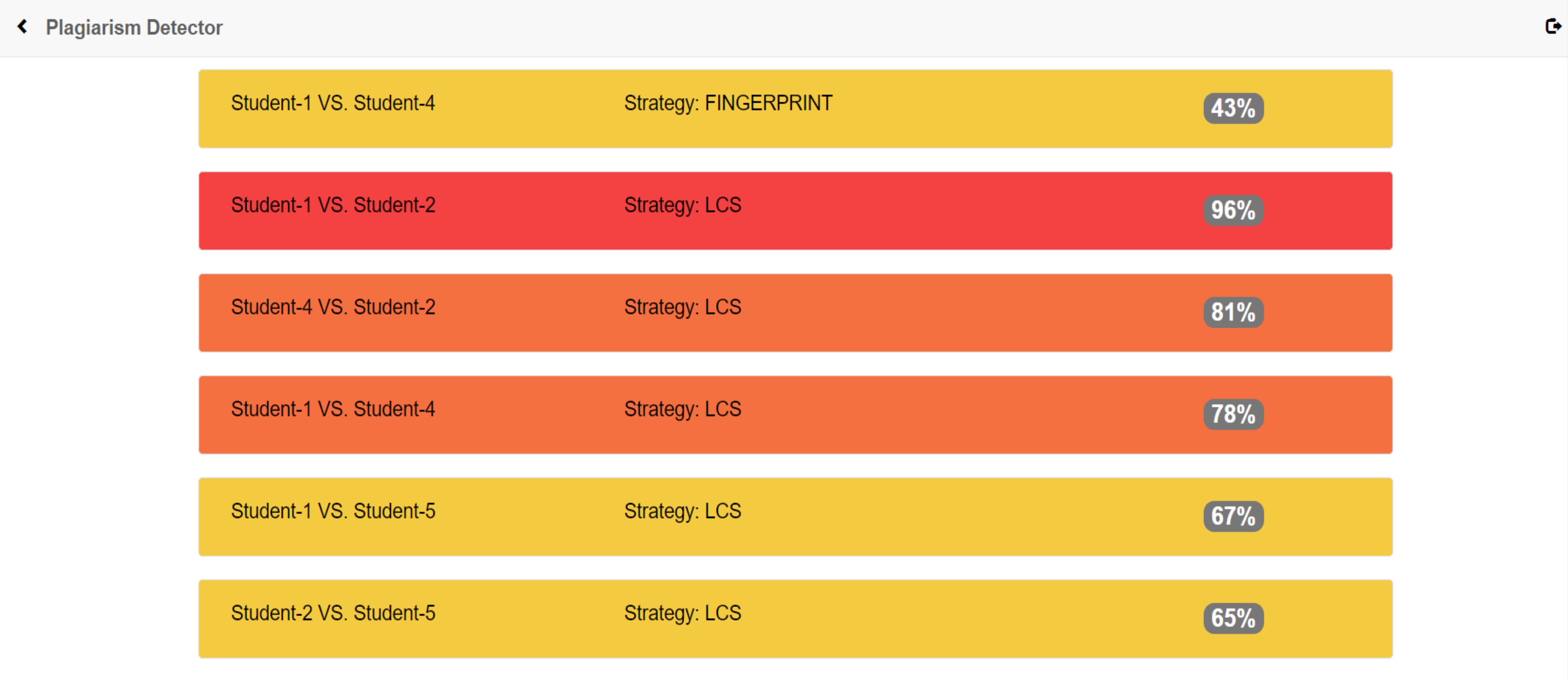


Result List

Very Likely: (x ≧ 90%) Likely: (75% ≦ x < 90%) Not Likely: (x < 75%)

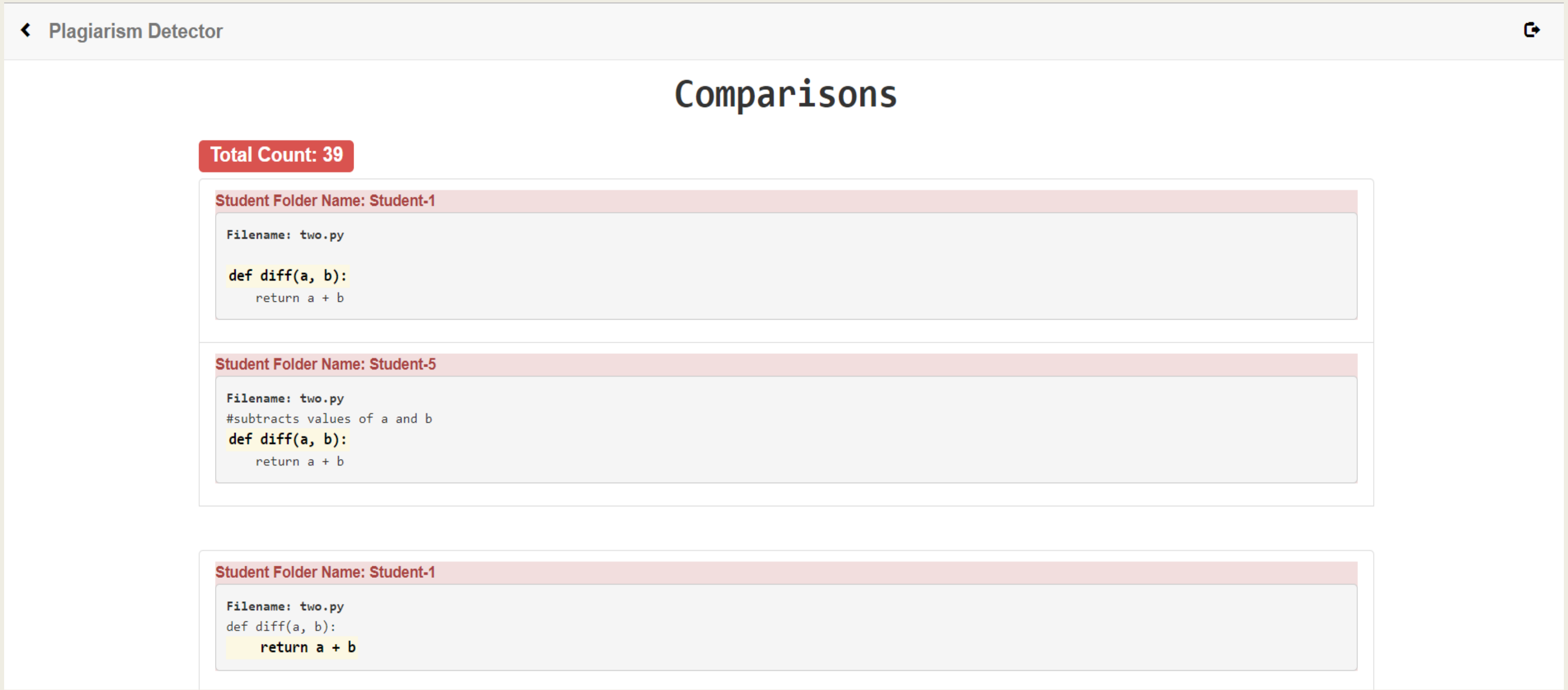
Student-1 VS. Student-2	Strategy: ALL	67%
Student-1 VS. Student-4	Strategy: ALL	56%
Student-4 VS. Student-2	Strategy: ALL	55%
Student-1 VS. Student-5	Strategy: ALL	51%
Student-1 VS. Student-2	Strategy: EDIT_DISTANCE	75%

Achieved System Functionalities - Screenshots



Student-1 VS. Student-4	Strategy: FINGERPRINT	43%
Student-1 VS. Student-2	Strategy: LCS	96%
Student-4 VS. Student-2	Strategy: LCS	81%
Student-1 VS. Student-4	Strategy: LCS	78%
Student-1 VS. Student-5	Strategy: LCS	67%
Student-2 VS. Student-5	Strategy: LCS	65%

Achieved System Functionalities - Screenshots



Comparisons

Total Count: 39

Student Folder Name: Student-1

```
Filename: two.py

def diff(a, b):
    return a + b
```

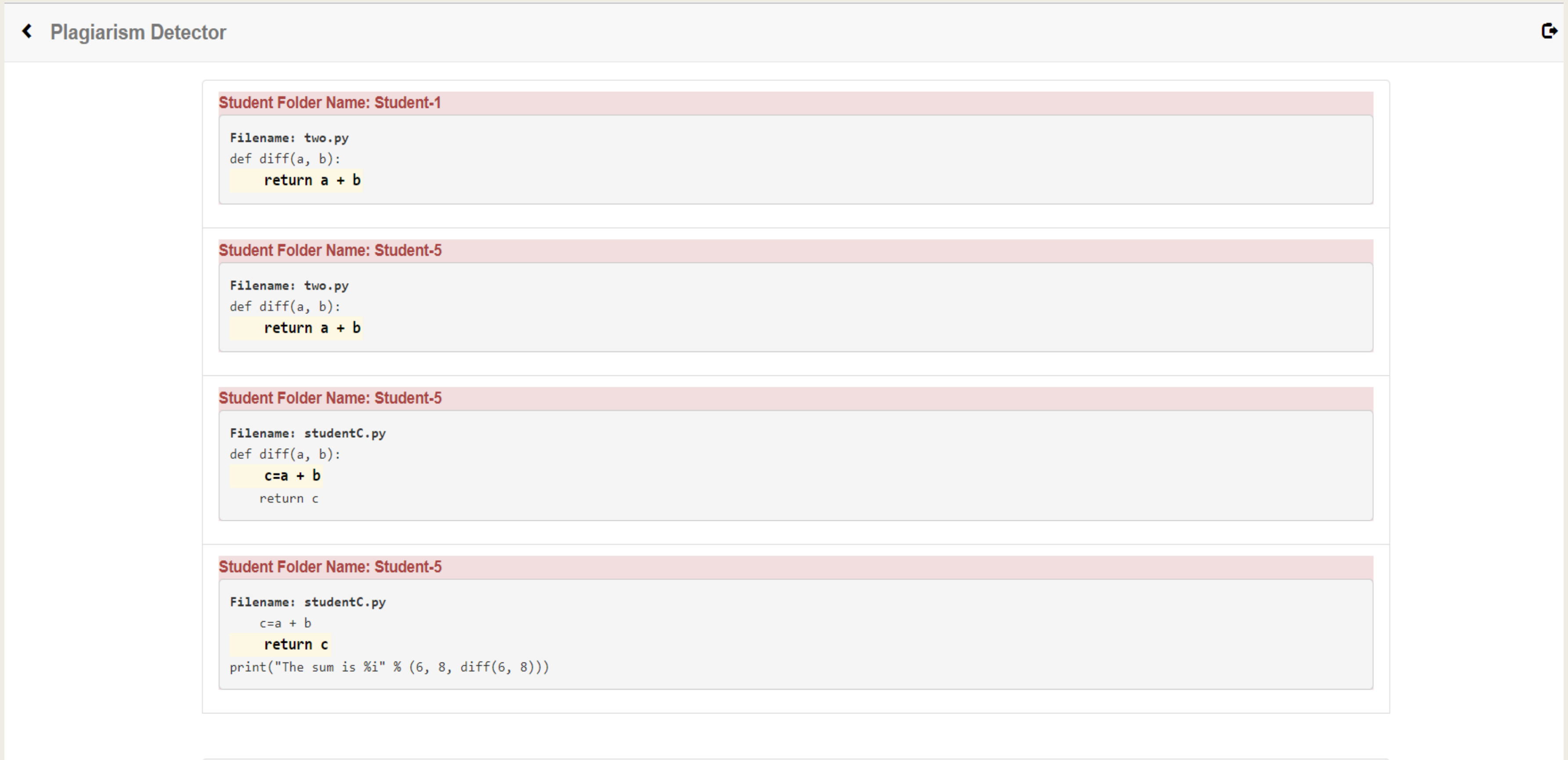
Student Folder Name: Student-5

```
Filename: two.py
#subtracts values of a and b
def diff(a, b):
    return a + b
```

Student Folder Name: Student-1

```
Filename: two.py
def diff(a, b):
    return a + b
```

Achieved System Functionalities - Screenshots



Student Folder Name: Student-1

```
Filename: two.py
def diff(a, b):
    return a + b
```

Student Folder Name: Student-5

```
Filename: two.py
def diff(a, b):
    return a + b
```

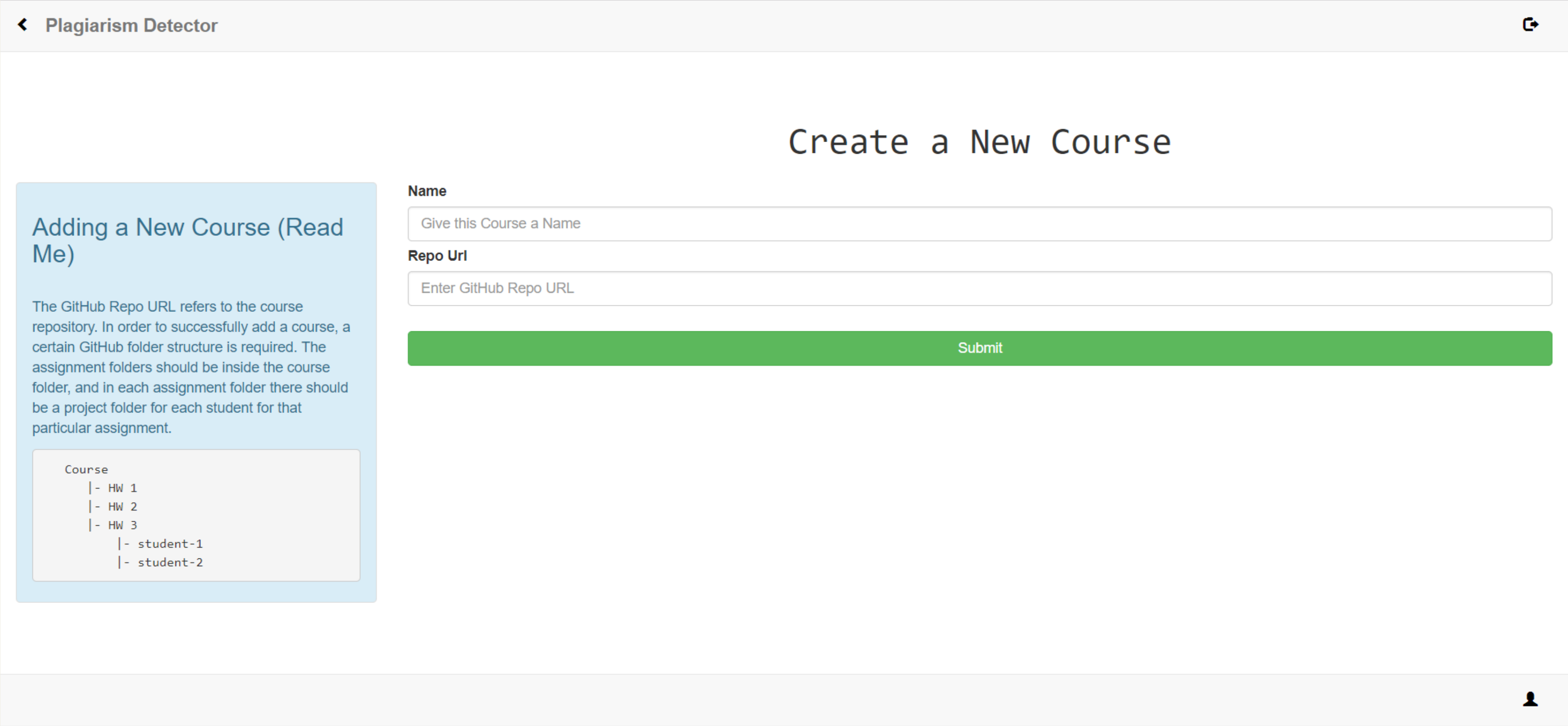
Student Folder Name: Student-5

```
Filename: studentC.py
def diff(a, b):
    c=a + b
    return c
```

Student Folder Name: Student-5

```
Filename: studentC.py
    c=a + b
    return c
print("The sum is %i" % (6, 8, diff(6, 8)))
```

Achieved System Functionalities - Screenshots



Create a New Course

Name

Give this Course a Name

Repo Url

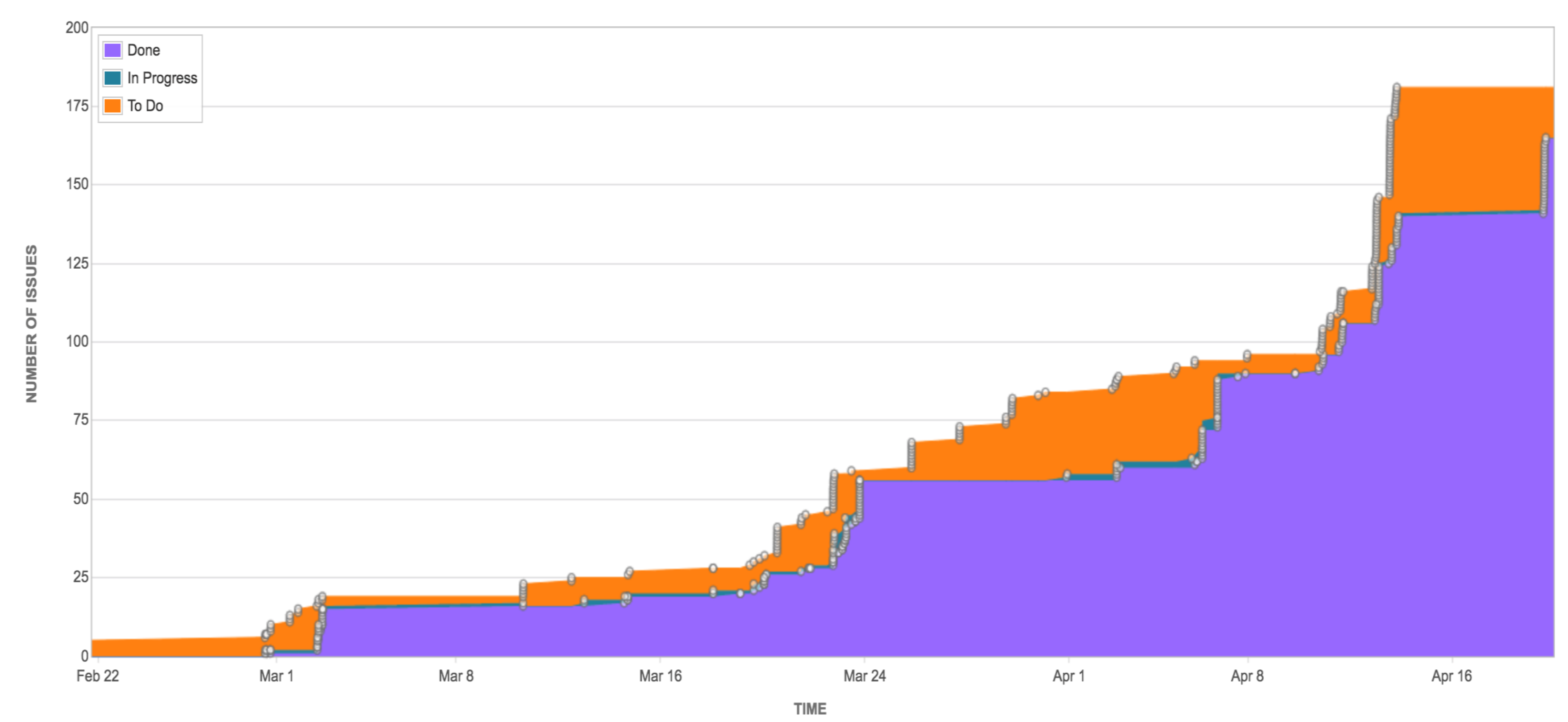
Enter GitHub Repo URL

Adding a New Course (Read Me)

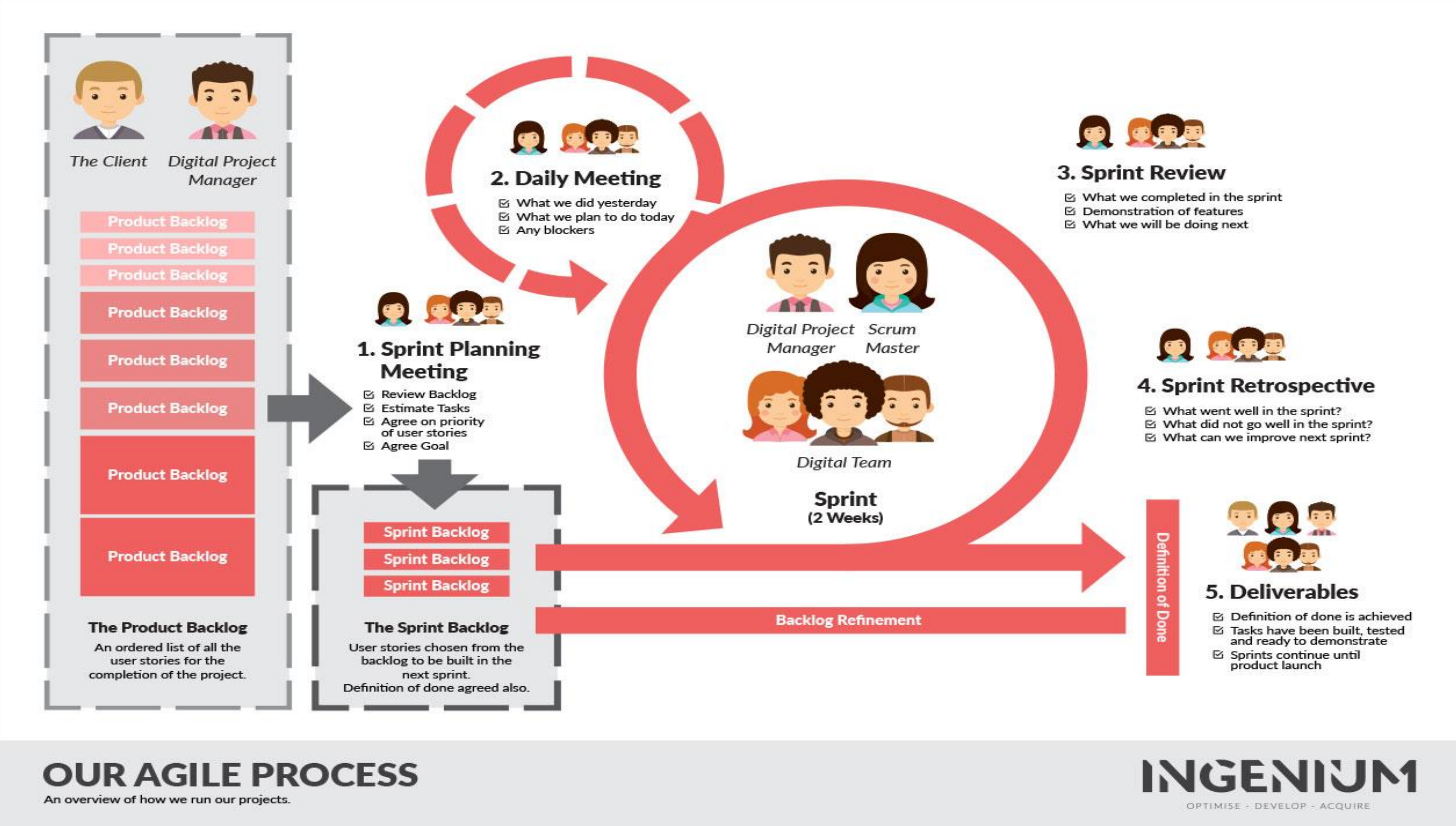
The GitHub Repo URL refers to the course repository. In order to successfully add a course, a certain GitHub folder structure is required. The assignment folders should be inside the course folder, and in each assignment folder there should be a project folder for each student for that particular assignment.

```
Course
   |- HW 1
   |- HW 2
   |- HW 3
       |- student-1
       |- student-2
```

Jira – Backlog Status



Development Process : Agile



The Product Backlog

An ordered list of all the user stories for the completion of the project.

1. Sprint Planning Meeting

- ☑ Review Backlog
- ☑ Estimate Tasks
- ☑ Agree on priority of user stories
- ☑ Agree Goal

The Sprint Backlog

User stories chosen from the backlog to be built in the next sprint.
Definition of done agreed also.

2. Daily Meeting

- ☑ What we did yesterday
- ☑ What we plan to do today
- ☑ Any blockers

3. Sprint Review

- ☑ What we completed in the sprint
- ☑ Demonstration of features
- ☑ What we will be doing next

4. Sprint Retrospective

- ☑ What went well in the sprint?
- ☑ What did not go well in the sprint?
- ☑ What can we improve next sprint?

5. Deliverables

- ☑ Definition of done is achieved
- ☑ Tasks have been built, tested and ready to demonstrate
- ☑ Sprints continue until product launch

OUR AGILE PROCESS

An overview of how we run our projects.

INGENIUM

OPTIMISE - DEVELOP - ACQUIRE

Independently Working on tasks

- Sprint Planning meetings, Features -> Subtasks -> Interfaces (Functional Specification)
- Teammates independently develop (implements interfaces) and integrates weekly and performs integration testing

- What we achieved?
  - *Lower coupling*
  - *High cohesion*
  - *Faster development*
  - *Well tested modules*

Development Process : Jira and Git

- We used Jira to manage the project lifecycle and assign user stories to developers

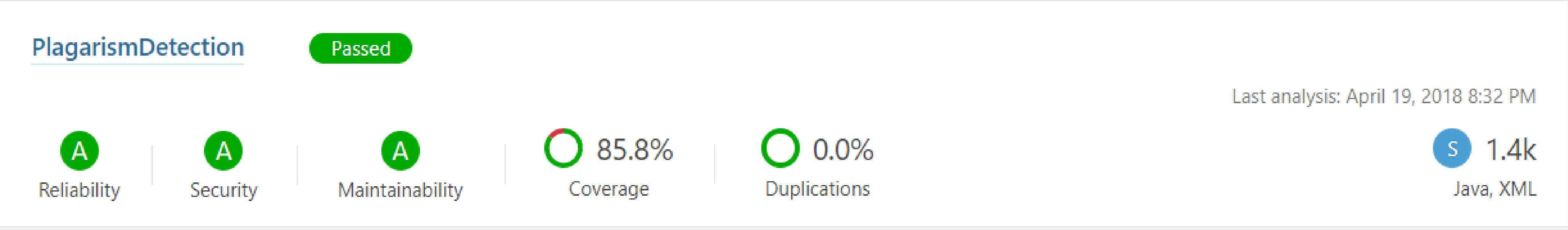
- Git for Version Control:
  - *Smart commits for managing Jira Lifecycle*
  - *GitHub reviewers before merging it to master branch*

Job Quality : Managing throughout the Deployment Process

- Before each feature is merged with master:
  - *Approved by a reviewer (Human errors)*
  - *Builds on Jenkins (System issues)*
  - *Test suite passes for the latest code on the branch (Regression testing)*
  - *SonarQube Test coverage exceeds the minimum Quality Gate of 80% (Testing)*
- Once merged to master, Jenkins automatically builds the master branch and deploys the latest code to the production server.

Job Quality : SonarQube

- Monitored using SonarQube and teammate reviewed pull-requests.
- The results outlined by the SonarQube report are as follows:
  - ❑ Reliability – A
  - ❑ Security – A
  - ❑ Maintainability – A
  - ❑ Code Duplications – 0%
  - ❑ Code Coverage – 85.8%
- This ensures a quality job done.



PlagarismDetection

Passed

Last analysis: April 19, 2018 8:32 PM

A	A	A	85.8%	0.0%	S 1.4k
Reliability	Security	Maintainability	Coverage	Duplications	Java, XML

Team Performance Over Time

- All of us were new to managing Jenkins and SonarQube, as the project progressed we learned and handled them with ease.
- The pace and amount of work done increased gradually over time.
- Rather than constraining a team member to work on one task at hand, multiple tasks were assigned and they were successfully accomplished.
- Overall we had a very good learning experience.

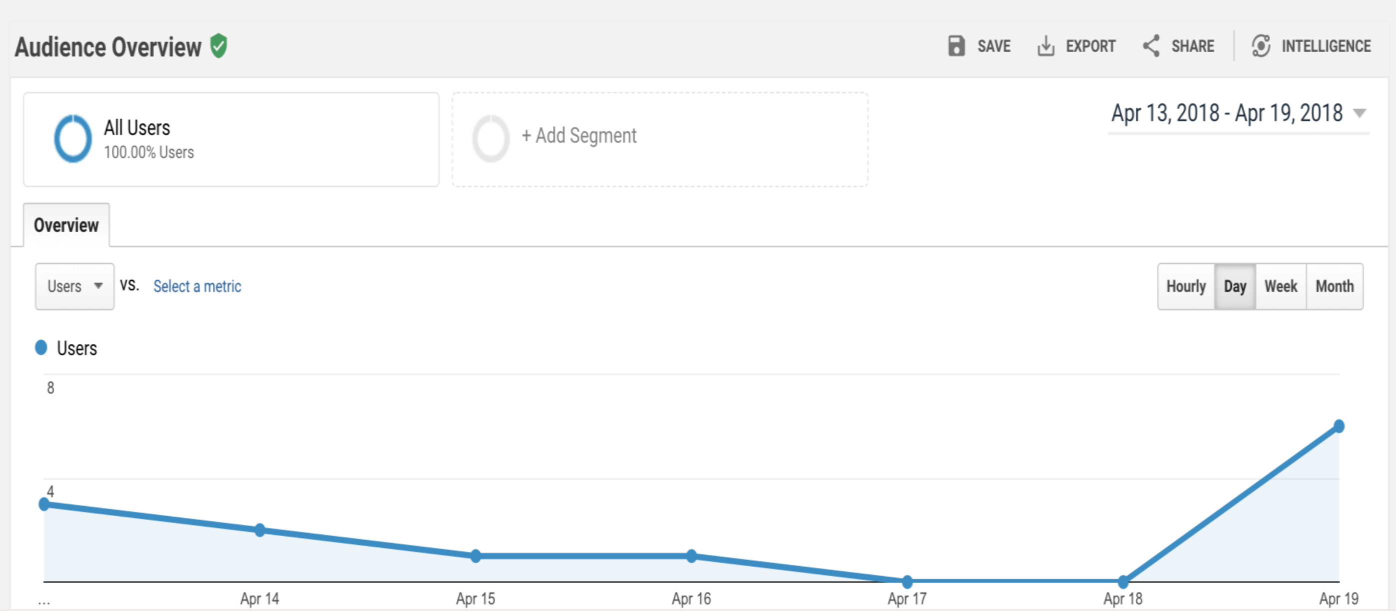
Shortcomings

- Dev-ops issues related to Jenkins and SonarQube.
- AWS builds were expensive.

Future Scope : A step further

- Already launched and live on a domain name ‘www.plagiarismpython.com’
- Integration of Google analytics platform to monitor incoming traffic and help make development decisions.
- Registered with multiple search engines as well as performed search engine optimization to attract traffic.

GOOGLE ANALYTICS

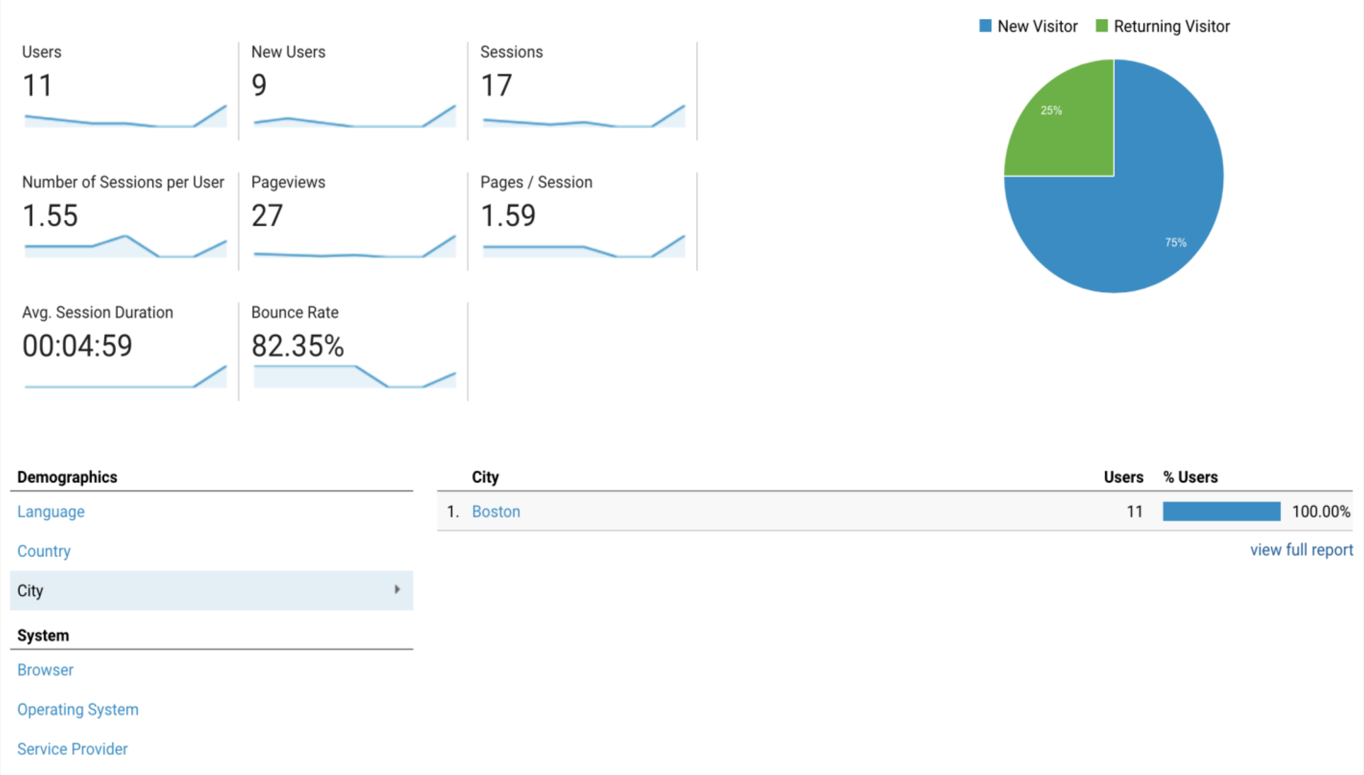


All Users
100.00% Users

Apr 13, 2018 - Apr 19, 2018

Users vs. Select a metric

● Users



Users	New Users	Sessions
11	9	17

Number of Sessions per User	Pageviews	Pages / Session
1.55	27	1.59

Avg. Session Duration	Bounce Rate
00:04:59	82.35%

Demographics

Language

Country

City

System

Browser

Operating System

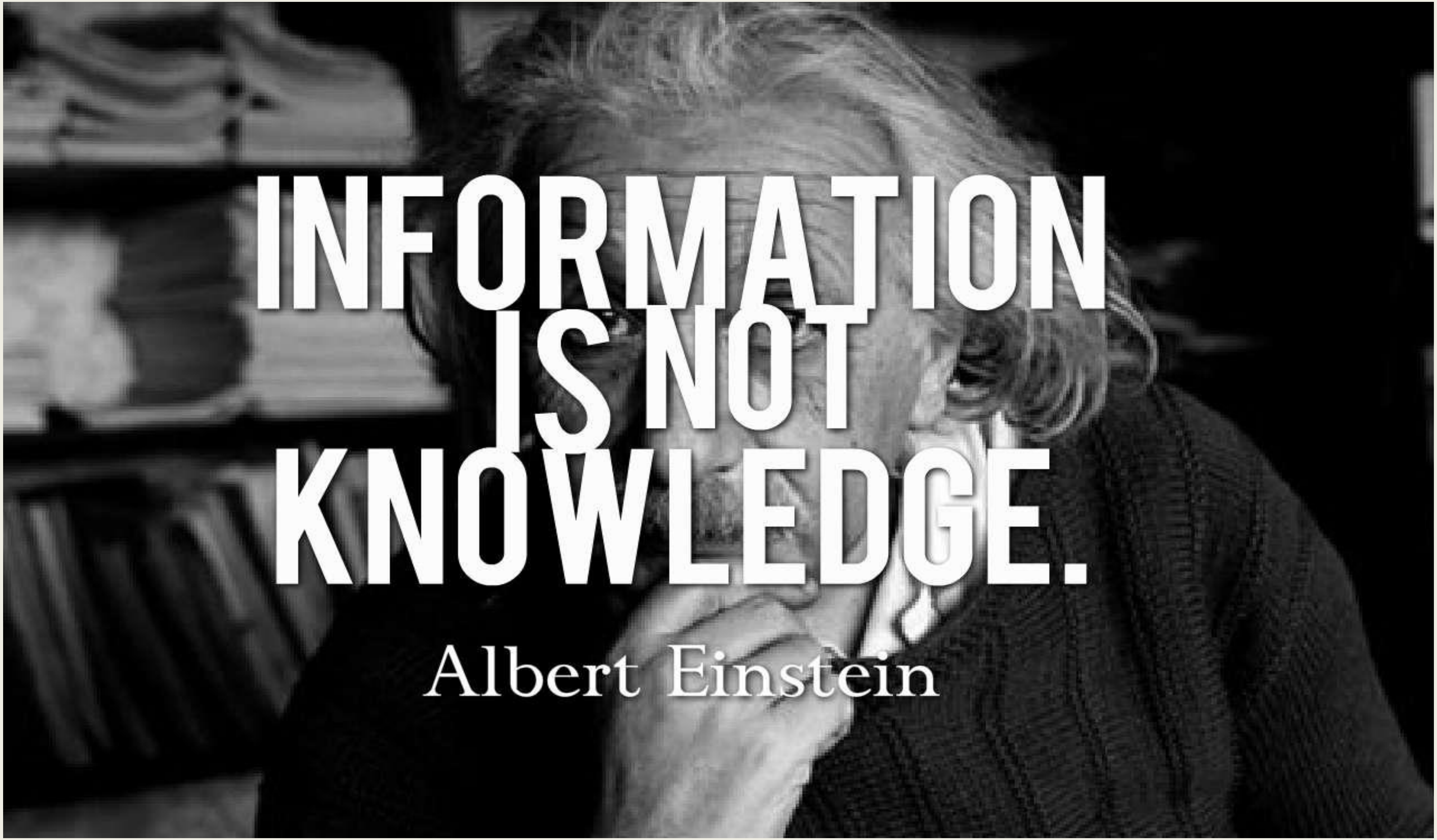
Service Provider

	City	Users	% Users
1.	Boston	11	100.00%

view full report

Future Scope : A step further

- Implementing session management.
- Extended to detect plagiarism among different programming languages.
- Further security improvements.



INFORMATION IS NOT KNOWLEDGE.

Albert Einstein