Project Name:	Moodle LMS
Product Name:	Moodle Testing 362
Product Release Version:	Moodle Version 3.7.2

Document Version 1.0

Created by: Chandler Long, Alex Laughlin, and Andrew Nesbett

Test Plan

Approach:

Test Execution

Test cases will be initialized and run with PHPUnit, a PHP testing framework required by Moodle to run unit tests. PHPUnit can be initialized from the root directory using

```
vendor/bin/phpunit
```

Running Single Tests

Running individual test cases can be achieved by running PHPUnit concurrently with the file we would like to test.

Ex.

```
vendor/bin/phpunit cohort/tests/cohortlib_test.php
```

Automated Testing Framework

1. Call a bash that loops through all our test cases files
2. The necessary data is aggregated and assigned to variables
3. The expected output is then written to the oracle
4. The bash script navigates to the directory in Moodle to run the method
5. The method being tested is run with the test cases input
6. The output of the method is written to a text file
7. The expected output, actual output, and the result of comparing the two are written to and saved in an HTML file

Test Cases

001
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
@@@
Array ( )

002
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
" "
Array ( )

003
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
""
Array ( )

004
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
Hey! :) Whats up!
Array ([0] => Hey [1] => Whats [2] => up)

005
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
:():()@!Hey!*
Array ([0] => Hey)

006

The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
Welcome to the framework!
Array ([0] => Welcome [1] => to [2] => the [3] => framework)

007
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
Moodle is... fun!
Array ([0] => Moodle [1] => is [2] => fun)

008
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
0101100
Array ([0] => 0101100)

009
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
"0101100"
Array ([0] => 0101100)

010
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
"The" "College" "of" "Charleston"
Array ([0] => The" "College" "of" "Charleston)

011
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
(3,3),(5,9)
Array ( )

012
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
1000 * 2 = 2000
Array ([0] => 1000 [1] => 2000)

013
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
"200 / 2 = 200"
Array ([0] => 200 / 2 = 200)

014
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
www.moodle.com
Array ([0] => www [1] => moodle [2] => com)

015
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
tokenize($needle)
Array ([0] => tokenize [1] => needle)

016
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
0.0021 is a decimal
Array ([0] => 0021 [1] => is [2] => decimal)

017
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize

"00.0021" is a string!
Array ([0] => 00 [1] => 0021 [2] => is [3] => string)

018
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
~/$HOME/Desktop/CSCI362
Array ([0] => ~/$HOME/Desktop/CSCI362)

019
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
moodle -> Framework
Array ([0] => moodle [1] => Framework)

020
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
"--------"
Array ([0] => --------)

021
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
" "
Array ( )

022
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
Adding y0u t0 the class
Array ([0] => Adding [1] => y0u [2] => t0 [3] => the [4] => class)

023
The tokenize method takes in a string and returns an array with the words indexed

projects/moodle/grade/grading/lib.php
tokenize
Hello world
Array ([0] => Hello [1] => world)

024
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
Test case
Array ([0] => Test [1] => case)

025
The tokenize method takes in a string and returns an array with the words indexed
projects/moodle/grade/grading/lib.php
tokenize
Nothing
Array ([0] => Nothing)

Test recording procedures

In order to confirm that out tests have completed successfully, the tests will be writing to .txt files with their results. This will provide useful insight into what type of test cases provide valuable feedback. This will also allow us to confirm that Moodle software is working as intended.

Hardware and software requirements

Moodle recommends the following hardware requirements:

- Disk space: 200MB for the Moodle code, plus as much as you need to store content. 5GB is probably a realistic minimum.
- Processor: 1GHz (min), 2GHz dual core or more recommended.
- Memory: 512MB (min), 1GB or more is recommended. 8GB plus is likely on a large production server

- Consider separate servers for the web "front ends" and the database. It is much easier to "tune"

Moodle recommends the following software requirements:

- Moodle upgrade: Moodle 3.2 or later

- PHP version: minimum PHP 7.1.0 *Note: minimum PHP version has increased since Moodle 3.6*. PHP 7.2.x and 7.3.x are

supported too. PHP 7.x could have some engine limitations.

- PHP extension **intl** is required since Moodle 3.4 (it was recommended in 2.0 onwards)

To run these tests, we will use:

- OS: Ubuntu OS
- Packages: The moodle repository as found on GitHub
- Processor: Moodle suggests 1GHz as a minimum but a mid-range, modern CPU (>3.0GHz) will drastically increase run time of our tests
- Memory: 8gb of RAM should be sufficient to run these tests in a timely manner
- Storage: The results of the tests should take up no more than 1GB of memory in addition to what Moodle recommends

Constraints

Constraints that affect the testing process: staff shortages, time allocation, deadlines

Staff shortages - due to our small team, testing all available methods in moodle would be impractical.

Time allocation - Due to the scope of the Moodle application, the test cases that the team will examine are limited.

Deadlines - Since the team is only working on the project for a semester, limitations are imposed by the time allotted