| | |
|---|---|
| **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

# 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
```

## Requirements traceability

Some of Moodle's system requirements are as follows:

Moodle should allow an authorized teacher to post a grade to a student's assignment.
Moodle should allow an authorized teacher to post feedback to a student's assignment.
Moodle should allow an authorized user the ability to view a course in which they are enrolled on the Moodle site
Moodle should allow a student or teacher the ability to post a comment to an open forum.
In Moodle 3.6 and above, Moodle should allow the authorized teacher of a course the ability to lock a forum so no user can post to the forum any longer.

**Tested items**

- Posting a grade - On the Moodle site, once a teacher has posted an assignment and a student submits his/her work, moodle allows a teacher to post a grade that can be seen by both the student and the teacher.

```
$this->getDataGenerator()->create_grade_item();

$this->getDataGenerator()-
>create_grade_item(array('itemtype' => $itemtype,
'itemname' => $itemname, 'outcomeid' => $outcomeid,
'scaleid' => $scaleid, 'gradetype' => $gradetype));
```

- Posting feedback - On the Moodle site, after a teacher has posted a grade, they can also post a body of text relating to the assignment and/or submission for the assignment.

```
$this->preventResetByRollback();

$sink = $this->redirectMessages();

//... code that is sending feedback

$messages = $sink->get_messages();

$this->assertEquals(3, count($messages));

//.. test messages were generated in correct order with
appropriate content
```

- View a Moodle class - Once a teacher has created a class on the moodle site, a graphic with the name of the class and a picture can be clicked and the user is taken into the class's information

```
$course1 = $this->getDataGenerator()->create_course();

$category = $this->getDataGenerator()->create_category();

$course2 = $this->getDataGenerator()-
>create_course(array('name'=>'Some course',
'category'=>$category->id));
```

- Creating Course Activities - Once a teacher has created an activity, students who are enrolled in the class can participate.

```
$course = $this->getDataGenerator()->create_course();

$generator = $this->getDataGenerator()-
>get_plugin_generator('mod_page');

$generator->create_instance(array('course'=>$course->id));
```

- Adding Users to Groups - Once a group is created and posted, the teacher has the ability to add and remove users to the group.

```
$this->getDataGenerator()-
>create_group_member(array('userid' => $userid, 'groupid'
=> $groupid));
```

```
$this->getDataGenerator()-
>create_group_member(array('userid' => $userid, 'groupid'
=> $groupid, 'component' => $component, 'itemid' =>
$itemid));
```

**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.