```php
<?php
require(dirname(dirname(dirname(__FILE__))) . '/config.php');
require_once $CFG->libdir . '/formslib.php';
require_once $CFG->dirroot.'/grade/lib.php';
require_once($CFG->dirroot.'/local/offline_attendance/classes/autoloader.php');
require_once $CFG->dirroot . '/local/offline_attendance/lib.php';
require_once($CFG->dirroot.'/lib/grade/grade_grade.php');

$id = required_param('id', PARAM_INT);    // course id

$course = $DB->get_record('course', array('id'=>$id), '*', MUST_EXIST);

require_course_login($course, true);
$context = context_course::instance($course->id);
$PAGE->set_context($context);

$attendance = new offline_attendance($id);

//총점 및 감점
$maxscore = (int)$attendance->maxscore;
$minscore = (int)$attendance->minscore;
$subtract = array(
              2 => $attendance->absent,
              3 => $attendance->late,
              4 => $attendance->early
            );

$fail_flag = get_config('local_offline_attendance', 'absentfail');

$roleobjs = $DB->get_records('role', array('archetype' => 'student'));
$roles = array_keys($roleobjs);
list($sql_in, $sql_params) = $DB->get_in_or_equal($roles, SQL_PARAMS_NAMED, 'roleid');

$sql_select = "SELECT  ur.* ";
$sql_from = " FROM {user} ur
              JOIN (
                SELECT userid
                FROM {role_assignments}
                WHERE contextid = :contextid AND roleid $sql_in
                GROUP BY userid
                ) ra ON ra.userid = ur.id ";

$sql_conditions = array('ur.deleted = :deleted');

$sql_params['contextid'] = $context->id;
$sql_params['deleted'] = 0;

$sql_where = ' WHERE '.implode(' AND ', $sql_conditions);
$sql_orderby = ' ORDER BY firstname, lastname ASC ';

$users = $DB->get_records_sql($sql_select.$sql_from.$sql_where.$sql_orderby, $sql_params);

$userids = array_keys($users);
list($sql, $params) = $DB->get_in_or_equal($userids, SQL_PARAMS_NAMED, 'userid');
$sql_select = ' SELECT * FROM {local_off_attendance_status} ';
```

```
$sql_where = ' WHERE userid '. $sql . ' and courseid = :courseid';
$params['courseid'] = $id;
$attendance_books = $DB->get_records_sql($sql_select.$sql_where, $params);

$grades = array();
foreach($attendance_books as $atttendance_book) {
    $userid = $atttendance_book->userid;
    $status = $atttendance_book->status;

    if(empty($grades[$userid])) {
        $grades[$userid] = $maxscore;
    }
    $currentgrade = $grades[$userid];

    // 결석이면서 결석일 경우 minscore 활성화인 경우
    if($status == 2 && $fail_flag) {
        $grades[$userid] = $minscore;
    } else {
        $tempgrade = $currentgrade +  $subtract[$status];
        if($tempgrade >= $minscore) {
            $grades[$userid] = $tempgrade;
        } else {
            $grades[$userid] = $minscore;
        }
    }
}

$count = 0;
foreach($users as $user) {
    $userid = $user->id;
    $gradegrade = grade_grade::fetch(array('itemid' => $attendance->itemid, 'userid' => $userid))
;
    if(empty($gradegrade)) {
        $gradegrade = new grade_grade();
        $gradegrade->itemid = $attendance->itemid;
        $gradegrade->userid = $userid;
        $gradegrade->finalgrade = $grades[$userid];
        $gradegrade->rawgrademax = floor($attendance->maxscore);
        $gradegrade->rawgrademin = floor ($attendance->minscore);
        $gradegrade->timecreated = time();
        $gradegrade->timemodified = time();
        $gradegrade->insert();
    }else {
        $gradegrade->finalgrade = $grades[$userid];
        $gradegrade->update();
    }
        $count++;
}

$returnvalue = new stdClass();
$returnvalue->status = 'success';
$returnvalue->count = $count;
$returnvalue->text = get_string('book:alert2', 'local_offline_attendance', $returnvalue);

@header('Content-type: application/json; charset=utf-8');
```

```
echo json_encode($returnvalue);
```