DAFTAR PUSTAKA

Anonim. “https://studylibid.com/doc/146456/bab-ii-tinjauan-pustaka-2.1-website”, diakses pada 25 April 2020 pukul 19.23.

Anonim. “https://123dok.com/document/zgl6p7nq-tinjauan-pengertian-komputer-informasi-pelayanan-berbasis-palembang-repository.html”, diakses pada 25 April 2020 pukul 21.34.

Chandra, Timotius Nugroho dan Inggriani Liem. 2013. *Source Code Editing Evaluator for Learning Programming*. Bandung: Institut Teknologi Bandung.

Dutta, Mala, Kamal K Sethi dan Ajay Khatri. 2014. “Web Based Integrated Development Environment” dalam *International Journal of Innovative Technology and Exploring Engineering (IJITEE) Volume 3, Issue 10* (hlm.56-60). India: Blue Eyes Intelligence Engineering & Sciences Publication Pvt. Ltd.

Jayaraju, Poreddy dan Vijay Prakash. 2015. “Web Based IDE Implementation for C, C++, C#, VB, Java, Perl, Python, Ruby, HTML, CSS, java script” dalam *International Journal of Innovative Science, Engineering & Technology Volume 2, Issue 10* (hlm.263-270). India: RGPV University Bhopal.

Jingwen, Ou, Mahdi Tayarani Najaran dan Mushfiqur Rouf. *Aurora SDK: A Web Based Integrated Development Environment*.

Rasmussen, Christian dan David Åse. 2014. *A Web-Based Code-Editor for Use in Programming Courses*. Norwegia: Norwegian University of Science and Technology.

Setiawan, Ganang Wahyu. 2011. *Pengujian Perangkat Lunak Menggunakan Metode Black Box Studi Kasus Exelsa Universitas Sanata Dharma*. Yogyakarta: Universitas Sanata Dharma.

Umar, Jefri. 2009. Analisis dan *Perancangan Perangkat Lunak IDE (Integrated Development Environment) Fortran G77*. Medan: Universitas Sumatera Utara.

Vu, Hien T. 2016. *Web-based Integrated Development Environment*. San Jose: San Jose State University.

LAMPIRAN

❖ Source Code

❖ Controller

➢ Class.js

```
const User = require('../models/User');
const Class = require('../models/Class');
const Exercise = require('../models/Exercise');
const Work = require('../models/Work');
const Worklog = require('../models/Worklog');
const Run = require('../models/Run');
const Runlog = require('../models/Runlog');
const Testlog = require('../models/Testlog');
const io = require('./socketio');
const Courses = require('../models/Class');
const Course = require('../models/Course');

/**
 * GET /
 * Home page.
 */
exports.index = (req, res) => {
  // console.log('user', req.user);

  if (req.user.role == 'student')
    return res.redirect('/student');

  Class.find(function(err, docs) {
    res.render('class/list', {title: 'Class List', classes: docs});
  });
};

exports.index2 = (req, res) => {
  // console.log('user', req.user);

  if (req.user.role == 'student')
    return res.redirect('/student');
  console.log(req.params.courseId);
  Class.find({courseId : req.params.courseId}, function(err, docs) {
    res.render('class/list', {title: 'Class List', classes: docs, courseId : req.p
arams.courseId});
  });
};
/**
 * GET /classlog/:classId
 * get one class
 */
exports.getById = async (req, res) => {
  if (req.user.role == 'student')
    return res.redirect('/student');

  try {

    var Classest = null
    Class.findById(req.params.classId, function(err, docs) {
      Classest = docs
    });


   const [classData, exerciseData, userData, courseData, classStudent] = await Pro
mise.all([
     Class.findById(req.params.classId).exec(),
     Exercise.find({courseId:req.params.courseId}).sort('title').exec(),
     User.find({}).sort('email').exec(),
     Course.findById(req.params.courseId).exec(),
```

```
     Class.find({courseId:req.params.courseId, _id: { $ne: req.params.classId } })
.exec()
    ]);

    console.log(classStudent)

    var students = userData.filter((s) => s.role === 'student')
      .map((s) => {
        s=Object.assign({},s._doc);
        s.inClass = classData.students.indexOf(s._id)>=0;
        return s;
      })
      .sort((l, r) => {
        if (l.inClass === true && r.inClass === false) {
          return -1;
        }
        if (l.inClass === false && r.inClass === true) {
          return 1;
        }
        return l.email.localeCompare(r.email);
      });

    var assistants = userData.filter((s) => s.role === 'assistant')
      .map((s) => {
        s=Object.assign({},s._doc);
        s.inClass = classData.staff.indexOf(s._id)>=0;
        return s;

      });
    var teachers = userData.filter((s) => s.role === 'teacher')
      .map((s) => {
        s=Object.assign({},s._doc);
        s.inClass = classData.staff.indexOf(s._id)>=0;
        return s;
      });
      // console.log(teachers);
    var exercises = exerciseData
      .map((s) => {
        s=Object.assign({active: false, deadline: ''},s._doc);
        // console.log('s', s);
        try {
          var d = (({active, deadline}) => ({active, deadline}))(classData.exercis
es.find((k)=> {return String(s._id) == String(k.exId)}));
          s = Object.assign(s, d);
          // s = Object.assign(s, (({active, deadline}) => {active, deadline})(cla
ssData.exercises.find((k)=> {return String(s._id) == String(k.exId)})) );
          // console.log('match', s, d);//classData.exercises.find((k)=> {return S
tring(s._id) == String(k.exId)}));
        }
        catch(err) {}
        return s;
      });


      var idStud = []
      classStudent.forEach(element => {
        element.students.forEach(s => {
          var foo = s;
          var bar = '' + foo;
          (idStud.push(bar))
        });
      });
    return res.render('class/edit', { title: 'Edit Class',
      courseData,classData, teachers, assistants, students, exercises, alreadyAssi
gn:idStud
    });
  } catch(err) {
    return res.status(500);
  }
}
```

```
exports.getAllClassCourse = async (req, res) => {

  if (req.user.role != 'teacher')
    return res.redirect('/');

  var classes = await Class.find({courseId: req.params.courseId}).exec();
  var thisCourse = await Course.findById(req.params.courseId).exec();

  return res.render('class/allClass', {title: thisCourse.title, classes: classes,
courseId : req.params.courseId});

}

exports.publish = (req, res) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  // var classData = Class.findById(req.params.classId)
  Class.findByIdAndUpdate(req.params.classId, { status: 1 }, function (err, doc) {
    if (err);

    return res.redirect('/course/'+doc.courseId);

  });
}

exports.archived = (req, res) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Class.findByIdAndUpdate(req.params.classId, { status: 0 }, function (err, doc) {
    if (err);

    return res.redirect('/course/'+doc.courseId);

  });
}

exports.logIndex = (req, res) => {
  if (req.user.role == 'student')
    return res.redirect('/student');

  Class.find(function(err, docs) {
    res.render('class/loglist', {title: 'Class Log List', classes: docs});
  });
};

exports.deleteById = (req, res) => {
  //if (req.user.role !== 'teacher')
    //return res.status(401).json({code:401, error: 'Wrong privilege'});
  var Classest = null
  Class.findById(req.params.classId, function(err, docs) {
    Classest = docs
  });
  Class.findByIdAndRemove(req.params.classId, function(err,doc) {
    if (err);

    return res.redirect('/course/');

  });
}

exports.getLogsById = async (req, res) => {
  if (req.user.role == 'student')
    return res.redirect('/student');

  try {
    // console.log(req.params.classId);
    var classData = await Class.findById(req.params.classId).exec();
    return res.render('class/logstats', {
      title: `Class Log Stats ${classData.classId}`,
```

```
      classId: req.params.classId,
      classTitle: classData.classId,
      userId: req.user._id,
    });
  } catch(err) {
    return res.status(500);
  }
    }

exports.getStaffworkById = async (req, res) => {
  if (req.user.role == 'student')
    return res.redirect('/student');

  try {
    // console.log(req.params.classId);
    var classData = await Class.findById(req.params.classId).exec();
    return res.render('class/staffwork', {
      title: `Class ${classData.classId}`,
      classId: req.params.classId,
      classTitle: classData.classId,
      userId: req.user._id,
    });
  } catch(err) {
    return res.status(500);
  }

exports.getLogsStatsById = async (req, res) => {
  // if (req.user.role == 'student')
  //   return res.redirect('/student');

  try {
    const classData = await Class.findById(req.params.classId)
        .populate('exercises.exId', 'title')
        .populate('staff')
        .populate('students')
        .exec();

    const threeHoursAgo = Date.now() - 3*3600000;
    const studentIdList = classData.students.map((s) => s._id);

    const timestamp = Number(req.params.timestamp) || 0;

    const [worklogData, runlogData, testlogData] = await Promise.all([
        Worklog.find().in('student', studentIdList)
          .gt('updatedAt', timestamp)
          .sort('updatedAt')
          // .populate('work', 'hash')
          .exec(),
        Runlog.find().in('student', studentIdList)
          // .gt('updatedAt', threeHoursAgo)
          .gt('updatedAt', timestamp)
          // .populate('run', 'hash')
          .sort('updatedAt')
          .exec(),
        Testlog.find().in('student', studentIdList)
          // .gt('updatedAt', threeHoursAgo)
          .gt('updatedAt', timestamp)
          // .populate('run', 'hash')
          .sort('updatedAt')
          .exec(),
    ]);

    worklogs.init(worklogData)
    runlogs.init(runlogData)
    testlogs.init(testlogData)

    return res.json({classData,
                     worklogs:(({latestWork, stats}) => ({latestWork, stats}))(work
logs),
                     runlogs:(({latestRun, stats}) => ({latestRun, stats}))(runlogs
),
```

```
                    testlogs:(({latestTest, stats}) => ({latestTest, stats}))(test
logs)
    });

  } catch(err) {
    return res.status(500);
  }

}

exports.create = (req, res, next) => {

  var newClass = new Class(req.body);
  newClass.save(function (err) {
      if (err)
        console.log(err);
        return res.status(500).json({code:500, error: err});
      return res.json({code:200});
  });
}

exports.updateById = (req, res, next) => {
  if (req.user.role !== 'teacher')
    return res.status(401).json({code:401, error: 'Wrong privilege'});

  Class.findByIdAndUpdate(req.params.classId, req.body, function(err, doc) {
    // console.log(err, doc);
    if (err)
      return res.status(500).json({code:500, error: err});
    return res.json({code:req.params.classId});
  });
}

exports.editById = (req, res, next) => {
   if (req.user.role !== 'teacher')
    return res.status(401).json({code:401, error: 'Wrong privilege'});

   return res.render('home',{
      title: 'Home'
    });
}
let worklogs = {
  data: {},
  stats: {},
  latestWork: {},
  initialized: false,
  updated: false,
  init: function(wl) {
      this.data = {};
      this.stats = {};
      this.latestWork = {};
      wl.forEach((w) => this.add(w));
      this.initialized = true;
      return this.data;
  },
  add: function(w) {
      if (!this.data[w.student]) {
          this.data[w.student] = {};
          this.stats[w.student] = {};
      }

      this.latestWork[w.student] = (({exercise, clientIp, updatedAt}) => ({exercis
e, clientIp, updatedAt}))(w);

      if (!this.data[w.student][w.exercise]) {
          this.data[w.student][w.exercise] = [w];
          this.stats[w.student][w.exercise] = {
              savecount: 0, // number of saves
              pastecount: 0, // number of pastes
              pasteChars: 0, // total characters of pastes
              clientIp: 0, //w.clientIp, // last ip address
```

```
              updatedAt: 0, //w.updatedAt, // last save time
              firstSavedAt: 0,
            };
      }

      this.stats[w.student][w.exercise].savecount = w.savecount >= 0 ? w.savecount
 : this.stats[w.student][w.exercise].savecount; // number of saves
      this.stats[w.student][w.exercise].clientIp = w.clientIp;
      if (w.savecount == -1) {
          var pastes = w.logs.filter((l) => l.type == 'paste')
          var pasteLength = pastes.reduce((acc,cur) => acc + cur.text.length, 0)
          this.stats[w.student][w.exercise].pastecount += pastes.length; // number
 of pastes
          this.stats[w.student][w.exercise].pasteChars += pasteLength; // total ch
aracters of pastes
      }
      else {
          this.stats[w.student][w.exercise].updatedAt = w.updatedAt;
          if (this.stats[w.student][w.exercise].firstSavedAt == 0)
            this.stats[w.student][w.exercise].firstSavedAt = w.updatedAt;
}
      // console.log('stats', this.stats[w.student][w.exercise], w);
      this.updated = true;
      return w;
  },
}

let runlogs = {
  data: {},
  stats: {},
  latestRun: {},
  initialized: false,
  updated: false,
  init: function(rl) {
    this.data = {};
    this.stats = {};
    this.latestWork = {};
    rl.forEach((r) => this.add(r));
      this.initialized = true;
      return this.data;
  },
  add: function(r) {
      // r : {... logs: [...] }
      if (!this.data[r.student]) {
          this.data[r.student] = {};
          this.stats[r.student] = {};
      }

      this.latestRun[r.student] = (({exercise, clientIp, updatedAt}) => ({exercise
, clientIp, updatedAt}))(r);

      let errors = r.logs.filter((l) => l[1].search('Error') >= 0).length;
      let functionCalls = r.logs.filter((l) => l[1] === 'element' && l[3] === 'sta
rt').length;

      if (!this.data[r.student][r.exercise]) {
          this.data[r.student][r.exercise] = [r];

          this.stats[r.student][r.exercise] = {
              runs: 1, // number of runs
              errors: errors, // number of errors on the last run
              functionCalls: functionCalls, // number of function calls on the las
t run
              clientIp: r.clientIp, // last ip address
              updatedAt: r.updatedAt, // last run update time
              firstRunAt: r.updatedAt, // first run time
          }
      }
      else {
          this.stats[r.student][r.exercise].clientIp = r.clientIp;
          this.stats[r.student][r.exercise].updatedAt = r.updatedAt;
```

```
          var studentLog = this.data[r.student][r.exercise];
          if (studentLog[studentLog.length-1].run == r.run) {
              // studentLog[studentLog.length-
1].logs = studentLog[studentLog.length-1].logs.concat(r.logs);

              this.stats[r.student][r.exercise].errors += errors;
              this.stats[r.student][r.exercise].functionCalls += functionCalls;
          }
          else {
              studentLog.push(r);
              if (studentLog.length > 1)
                  studentLog.shift();
              // new run; add run counts,
              this.stats[r.student][r.exercise].runs++;
              this.stats[r.student][r.exercise].errors = errors;
              this.stats[r.student][r.exercise].functionCalls = functionCalls;
          }
      }
      // console.log('stats', this.stats[r.student][r.exercise], errors, functionC
alls, r.logs)
      this.updated = true;
      return r;
  },
}

let testlogs = {
  data: {},
  stats: {},
  latestTest: {},
  initialized: false,
  updated: false,
  init: function(tl) {
    this.data = {};
    this.stats = {};
    this.latestWork = {};
    tl.forEach((r) => this.add(r));
      this.initialized = true;
      return this.data;
  },
  add: function(t) {
      // t : {... logs: [...] }
      if (!this.data[t.student]) {
          this.data[t.student] = {};
          this.stats[t.student] = {};
      }

      this.latestTest[t.student] = t;

      let testResults = t.logs[2]
      let errors = Object.keys(testResults).filter((l) => testResults[l] == false)
;

      if (!this.data[t.student][t.exercise]) {
          this.data[t.student][t.exercise] = [t];

          this.stats[t.student][t.exercise] = {
              tests: 1, // number of tests
              errors: errors, // errors on the last test
              clientIp: t.clientIp, // last ip address
              updatedAt: t.updatedAt, // last test update time
              firstPassAt: errors.length ? 0 : t.updatedAt, // test passes the fir
st time
          }
      }
      else {
          this.stats[t.student][t.exercise].clientIp = t.clientIp;
          this.stats[t.student][t.exercise].updatedAt = t.updatedAt;
          this.stats[t.student][t.exercise].tests++;
          this.stats[t.student][t.exercise].errors = errors;
```

```
          this.stats[t.student][t.exercise].firstPassAt = errors.length ? 0 : t.up
datedAt;
      }
      // console.log('stats', this.stats[t.student][t.exercise], errors, functionC
alls, t.logs)
      this.updated = true;
      return t;
  },
}
```

- course.js

```
const User = require('../models/User');
const Class = require('../models/Class');
const Exercise = require('../models/Exercise');
const Work = require('../models/Work');
const Course = require('../models/Course');
const Module = require('../models/Module');
const Quis = require('../models/Quis');
const QuisStudent = require('../models/QuisStudent');
const Discussion = require('../models/Discussion');
const Forum = require('../models/Forum');
const formidable = require('formidable');
const multer = require('multer');
const path = require('path');
var url = require('url') ;
const { WorkspaceContext } = require('twilio/lib/rest/taskrouter/v1/workspace');

const storage = multer.diskStorage({
  destination: 'public/image/',
  filename: function(req, file, cb){
    cb(null, file.fieldname + '-' + Date.now() + path.extname(file.originalname));
  }
});

//init upload
const upload = multer({
  storage : storage,
  limits:{filesize: 1000000},
  filefilter:function(req,file,cb){
    checkFileType(file, cb)
  }
}).single('myImage');

function checkFileType(file, cb){
  const filetypes = /jpeg|jpg|png|gif/
  const extname = filetypes.test(path.extname(file.originalname).toLocaleLowerCase
());
  const mimetype = filetypes.test(file.mimetype)

  if(mimetype && extname){
    return cb(null,true)
  } else {
    cb('Error : image only')
  }
}

exports.upload = async (req, res) => {
  upload(req,res, (err) => {
    if(err){
      res.render('/course/edit', {
        msg: err
      })
    } else {
      // findOneAndUpdate({_id: id}, {$set: body}, {new: true, useFindAndModify: f
alse})
      if(req.file){
        Course.findByIdAndUpdate(req.params.courseId, { image: req.file.filename }
, {new: true, useFindAndModify: false}, function (err, doc) {
          if (err)
            return res.status(500).json({ code: 500, error: err });
```

```
        })
      }
      req.flash('success', { msg: 'Success! Picture uploaded.' });
      return res.redirect('/course');
    }
  })
  // return res.redirect('/course');
}


exports.indexByStudent = async (req, res) => {

  const [workData, classData] = await Promise.all([
    Work.find({ student: req.user._id }).exec(),
    Class.find({ students: { $all: [req.user._id] } })
      .populate('courseId').exec()
  ]);

  const coursesData = classData.map((e) => { return e.courseId; });
  console.log(coursesData);

  var d = new Date();
  var n = d.getMonth();
  var currentSemester = n > 8 ? 'awal' : 'akhir'
  var courseList = await Course.find({$or: [{semester: currentSemester}, {status:
1}]}, {}, { sort: { order: 1 } }).exec();
  return res.render('course/studentlist', { courses: coursesData, title: 'My Cours
es' });
}

exports.getByStudent = async (req, res) => {

}

exports.livecourses = async (req, res) => {

  var thisCourse = await Course.findById(req.params.courseId).exec();

  return res.render('course/livecourses', {
    title: thisCourse.title
  });
};

exports.previewModuleById = async (req, res) => {
  var thisCourse = await Course.findById(req.params.courseId).exec();
  var thisModules = await Module.findById(req.params.moduleId).exec();
  var Modules = await Module.find({ courseId: req.params.courseId }).exec();
  var Quises = await Quis.find({ courseId: thisCourse.id , status:1}).exec();
  var QuisesStudent = await QuisStudent.find({studentId:req.user.id}).exec();
  var isAlreadyAnswer = QuisesStudent.length > 0
  var thisDiscussion = await Discussion.find({moduleId : req.moduleId}).exec();
  var thisForum = await Forum.find({courseId : req.params.courseId},{},{ sort: { c
reatedAt: -1 } }).exec();
  var thisExercise = await Exercise.find({courseId: req.params.courseId},{},{ sort
: { createdAt: -1 } }).exec();
  // console.log(thisModules)
  const [workData, classData] = await Promise.all([
    Work.find({student: req.user._id}).exec(),
    Class.findOne({students:{$all: [req.user._id]}})
      .populate('exercises.exId').exec()
  ]);


  var Classest = null
  Class.find({ courseId: req.params.courseId }, function (err, docs) {
    Classest = docs
  });
  console.log("image : " + thisCourse.image)
  return res.render('course/studentmodulelist', { thisForum:thisForum,thisModules
: thisModules, ex:thisCourse, idCourse: thisCourse.id, title: thisCourse.title, mo
```

```
dules: Modules, classest: Classest, quisest: Quises, isAlreadyAnswer:isAlreadyAnsw
er, discussion:thisDiscussion, exercise:thisExercise, classExercise:classData });
}

exports.previewModuleById2 = async (req, res) => {

  try {
    console.log(req.params.moduleId)
    var thisDiscussion = await Discussion.find({parentId : req.params.moduleId},{}
,{ sort: { createdAt: -1 } }).exec();
    var thisCourse = await Course.findById(req.params.courseId).exec();
    var thisModule = await Module.findById(req.params.moduleId).exec();
    var thisForum = await Forum.find({courseId : req.params.courseId},{},{ sort: {
 createdAt: -1 } }).exec();
    return res.render('course/preview', {thisForum:thisForum, discussion:thisDiscu
ssion,title: thisModule.titleModule, ex: thisModule, course:thisCourse });
  } catch (err) {
    console.log(thisCourse);
    res.status(500);
  }
}

exports.previewModuleList = async (req, res) => {

  console.log(req.params.courseId)
  // console.log("llllll")
  var Modules = await Module.find({ courseId: req.params.courseId }).exec();
  var thisCourse = await Course.findById(req.params.courseId).exec();

  return res.render('course/previewModuleList', {modules:Modules, ex:thisCourse});
}

exports.index = async (req, res) => {
  if (req.user.role == 'student')
    return res.redirect('/studentcourses');

  var d = new Date();
  var n = d.getMonth();
  var currentSemester = n > 8 ? 'awal' : 'akhir'
  var hostname = req.headers.host; // hostname = 'localhost:8080'
  var pathname = url.parse(req.url).pathname; // pathname = '/MyApp'


  // var courseList = await Course.find({semester:currentSemester, status:1}, {},
{ sort: { order: 1 } }).exec();
  var courseList = await Course.find({}, {}, { sort: { status: -1 } }).exec();

  return res.render('course/list', { courses: courseList, title: 'Course List', fi
le:`image/` });




exports.create = (req, res, next) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  var newCourse = new Course(req.body);
  newCourse.save(function (err) {
    if (err)
      return res.status(500).json({ code: 500, error: err });
    return res.json({ code: 200 });
  });
}

exports.createModule = (req, res, next) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  var newModule = new Module(req.body);
  newModule.save(function (err) {
```

```
    if (err)
      return res.status(500).json({ code: 500, error: err });
    return res.json({ code: 200 });
  });
}

exports.createExercise = (req, res, next) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  var newExercise = new Exercise(req.body);
  newExercise.save(function (err) {
    if (err)
      return res.status(500).json({ code: 500, error: err });
    return res.json({ code: 200 });
  });

}

exports.getById = async (req, res) => {
  // console.log(req);
  if (req.user.role == 'student')
    return res.redirect('/studentcourses');

  const [classData, exerciseData, userData, moduleData, classList, exerciseList, q
uisList] = await Promise.all([
    Class.findById(req.params.classId).exec(),
    Exercise.find({}).sort('title').exec(),
    User.find({}).sort('email').exec(),
    Module.find({ courseId: req.params.courseId }).exec(),
    Class.find({ courseId: req.params.courseId }).exec(),
    Exercise.find({ courseId: req.params.courseId }).exec(),
    Quis.find({ courseId: req.params.courseId }).exec(),
  ]);
  console.log("classData");
  var thisForum = await Forum.find({courseId : req.params.courseId},{},{ sort: { c
reatedAt: -1 } }).exec();
  //
  console.log(req.params.courseId);
  var teachers = userData.filter((s) => s.role === 'teacher')
    .map((s) => {
      s = Object.assign({}, s._doc);
      if (classData != null)
        s.inClass = classData.staff.indexOf(s._id) >= 0;
      a = JSON.stringify(s._id);
      b = JSON.stringify(req.user._id);
      if (a == b) s.disabled = true;
      return s;
    });
  var modules = moduleData;
  var classes = classList;
  var exercises = exerciseList;
  var quis = quisList;
  var currentDate = new Date()
  var currentYear = currentDate.getFullYear()
  var lastTeenYear = currentYear - 10
  var listYear = []
  for(i=lastTeenYear;i<=currentYear+10; i++){
    listYear.push(i+1)
  }

  var d = new Date();
  var n = d.getMonth();
  var currentSemester = n > 8 ? 'awal' : 'akhir'

  var classBySemesterandYear = await Class.find({$and: [{courseId: req.params.cour
seId}, {status:1}]}, {}, { sort: { order: 1 } }).exec();

  try {
    var thisCourse = await Course.findById(req.params.courseId).exec();
    var author = await User.findById(thisCourse.author).exec();
```

```
    console.log(thisCourse)
    return res.render('course/edit', { forum:thisForum,title: thisCourse.title, ex
: thisCourse, teachers, modules, idCourse: req.params.courseId, author: author.pro
file.name, classes, exercises, quis, listYear:listYear, currentYear:currentYear, c
lassBySemesterandYear:classBySemesterandYear, imageCourse:thisCourse.image });
  } catch (err) {
    console.log(thisCourse);
    res.status(500);
  }
};

exports.getModuleById = async (req, res) => {
  // console.log(req.params.moduleId);
  try {

    var thisModule = await Module.findById(req.params.moduleId).exec();
    var thisCourse = await Course.findById(req.params.courseId).exec();
    console.log(thisModule);
    return res.render('course/editmodule', { title: thisModule.titleModule, ex: th
isModule, courseTitle: thisCourse.title, courseId: req.params.courseId, moduleId:
req.params.moduleId });
  } catch (err) {
    console.log(err);
  }
}

exports.getExerciseByCourse = async (req, res) => {
  try {
    // console.log(req.params);
    var thisCourse = await Course.findById(req.params.courseId).exec();
    var thisExercises = await Exercise.find({ courseId: req.params.courseId }).pop
ulate('parentId').exec();

    // Query for add data module
    var thisModules = await Module.find({ $or: [{ courseId: req.params.courseId },
 { titleModule: 'None' }] }).exec();

    var Modules = await Module.find({ courseId: req.params.courseId }).exec();
    // console.log(Modules);
    console.log(thisExercises);
    return res.render('course/exerciseslist',
      {
        title: thisCourse.title,
        ex: thisCourse,
        exercises: thisExercises,
        thisModules: thisModules
      });
  } catch (err) {
    console.log(err);
  }
}

exports.getExerciseById = async (req, res) => {
  try {
    var thisCourse = await Course.findById(req.params.courseId).exec();
    var thisExercise = await Exercise.findById(req.params.exerciseId).exec();
    var thisModules = await Module.find({ $or: [{ courseId: req.params.courseId },
 { titleModule: 'None' }] }).exec();
    var modules = await Module.find({courseId : req.params.courseId}).exec();
    // var thisCourse = await Course.find(req.params.courseId).exec();


    return res.render('course/editexercise', {
      title: '', ex: thisExercise, thisCourse, thisModules, courseId: req.params.c
ourseId, exerciseId: req.params.exerciseId
    });
  } catch (err) {
    console.log(err);
  }
}
```

```
exports.getAllExerciseStudentById = async (req, res) => {
  try {
    // console.log(req.params.exerciseId)
    var thisCourse = await Course.findById(req.params.courseId).exec();
    var thisExercise = await Exercise.findById(req.params.exerciseId).exec();
    var thisModules = await Module.find({ $or: [{ courseId: req.params.courseId },
 { titleModule: 'None' }] }).exec();
    var modules = await Module.find({courseId : req.params.courseId}).exec();
    // var thisCourse = await Course.find(req.params.courseId).exec();
    var thisExerciseListByStudent = await Work.find({exercise : req.params.exercis
eId}).populate('student').exec()
    var u = Work.find({ exercise: { $all: [req.params.exerciseId] } })
      .populate('student').exec()
    console.log(thisExerciseListByStudent)
    return res.render('exercise/exerciseList', {
      title: '', ex: thisExercise, thisCourse, thisModules, courseId: req.params.c
ourseId, exerciseId: req.params.exerciseId, exercisesList:thisExerciseListByStuden
t
    });
  } catch (err) {
    console.log(err);
  }
}

exports.previewById = async (req, res) => {
  console.log(req);
  // if (req.user.role == 'student')
  //   return res.redirect('/studentcourses');

  try {
    var thisCourse = await Course.findById(req.params.courseId).exec();
    return res.render('course/preview', { title: thisCourse.title, ex: thisCourse
});
  } catch (err) {
    console.log(thisCourse);
    res.status(500);
  }
};

exports.deleteById = (req, res) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  /*Delete all customers where the address starts with an "O":*/
  var myquery = { courseId: req.params.courseId };
  Class.deleteMany(myquery, function(err, obj) {
    if (err) throw err;
    console.log(" document(s) deleted");
    // db.close();
  });


  Course.findByIdAndRemove(req.params.courseId, function (err, doc) {
    if (err);

    return res.redirect('/course');

  });
}

exports.archieved = (req, res) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Course.findByIdAndUpdate(req.params.courseId, { status: 0 }, function (err, doc)
 {
    if (err);

    return res.redirect('/course');

  });
```

```
}

exports.publish = (req, res) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Course.findByIdAndUpdate(req.params.courseId, { status: 1 }, function (err, doc)
 {
    if (err);

    return res.redirect('/course');

  });
}

exports.deleteModuleById = (req, res) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Module.findByIdAndRemove(req.params.moduleId, function (err, doc) {
    if (err);
    return res.redirect('/course/' + req.params.courseId);
  });
}

exports.deleteExerciseById = (req, res) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Exercise.findByIdAndRemove(req.params.exerciseId, function (err, doc) {
    if (err);
    return res.redirect('/course/' + req.params.courseId);
  });
}

exports.updateExerciseById = (req, res, next) => {
  if (req.user.role !== 'teacher')
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Exercise.findByIdAndUpdate(req.params.exerciseId, req.body, function (err, doc)
{
    console.log(req.body);
    if (err)
      return res.status(500).json({ code: 500, error: err });
    return res.json({ code: req.body });
  });
}

exports.updateModuleById = (req, res, next) => {
  if (!(req.user.role === 'teacher' || req.user.role === 'admin'))
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Module.findByIdAndUpdate(req.params.moduleId, req.body, function (err, doc) {
    console.log(req.body);
    if (err)
      return res.status(500).json({ code: 500, error: err });
    return res.json({ code: req.params.moduleId });
  });
}

exports.updateById = (req, res, next) => {
  if (req.user.role !== 'teacher')
    return res.status(401).json({ code: 401, error: 'Wrong privilege' });

  Course.findByIdAndUpdate(req.params.courseId, req.body, function (err, doc) {
    console.log(req.body);
    if (err)
      return res.status(500).json({ code: 500, error: err });
    return res.json({ code: req.params.courseId });
  });
}
```

- exercise.js

```
const User = require('../models/User');
const Class = require('../models/Class');
const Exercise = require('../models/Exercise');
const Work = require('../models/Work');
const Course = require('../models/Course');

exports.index = async (req, res) => {
  if (req.user.role == 'student')
    return res.redirect('/student');

  var exerciseList = await Exercise.find({}).exec();

exports.getById = async (req, res) => {
  if (req.user.role == 'student')
    return res.redirect('/student');

  if (req.params.exerciseId === 'new') {
    return res.render('exercise/new', {
    });
  }
  try {
      var thisExercise = await Exercise.findById(req.params.exerciseId).exec();
      return res.render('exercise/edit', {title: thisExercise.title, ex: thisExerc
ise});
  } catch(err) {
      console.log(err);
      res.status(500);
  }
};

exports.create = (req, res, next) => {
  if (req.user.role == 'student')
    return res.status(404).json({code:404});

  var newExercise = new Exercise(req.body);
  newExercise.save(function (err) {
      if (err)
        return res.status(500).json({code:500, error: err});
      return res.json({code:200});
  });
};

exports.updateById = (req, res, next) => {
  if (req.user.role == 'student')
    return res.status(404).json({code:404});

  Exercise.findByIdAndUpdate(req.params.exerciseId, req.body, function(err, doc) {
    // console.log(err, doc);
    if (err)
      return res.status(500).json({code:500, error: err});
    return res.json({code:200});
  });
};

exports.indexByStudent = async (req, res, next) => {
  // console.log(req.user.id)
  if (req.user && req.user.role === 'student') {
    try {
      const [workData, classData] = await Promise.all([
        Work.find({student: req.user._id}).exec(),
        Class.find({students:{$all: [req.user._id]}})
          .populate('exercises.exId').populate('courseId').exec()
      ]);


      var thisExercise = await Exercise.find({courseId: req.user._id},{},{ sort: {
 createdAt: -1 } }).exec();
      // console.log('exerciseGroup', exerciseGroups);
```

```
      var ex = classData.map((e)=> {return e.exercises})
      var eex = ex.map((e)=>{return e})



      return res.render('exercise/studentlist', {title: 'Exercises', inClass: true
, exercise:classData});
  }
  catch(e) {
    console.log('err', e);
    return res.render('exercise/studentlist', {title: 'Exercises', exerciseGroups:
 null, inClass: false});
  }

  }

  return res.status(404).json({code:404});
}
// middleware
exports.indexByStudentMiddleware = async (req, res, next) => {
  if (req.user && req.user.role === 'student') {
    const [workData, classData] = await Promise.all([
      Work.find({student: req.user._id}).exec(),
      Class.findOne({students:{$all: [req.user._id]}})
        .populate('exercise').exec()
    ]);
    const exerciseData = classData.exercises;

    res.locals.exercises = exerciseData;
    res.locals.work = workData;

    return next();
  }

  next();
}
```

- latihan.js

```
exports.latihan= (req, res) => {
//  if (!req.user) {
      return res.render('latihan/latihan', {
        title: 'Login'
      });
 // }
};
```

- Work.js

```
const mongoose = require('mongoose');

const workSchema = new mongoose.Schema({
    // exerciseTitle: String, // relationship: Exercise.title XXX howto?
    class: {type:mongoose.Schema.Types.ObjectId, ref: 'Class'},
    student: {type:mongoose.Schema.Types.ObjectId, ref: 'User'},
    exercise: {type:mongoose.Schema.Types.ObjectId, ref: 'Exercise'},
    deadline: Date,
    html: {type: String, default: ''},
    css: {type: String, default: ''},
    js: {type: String, default: ''},
    hash: String, // hash of html+css+js
    savecount: Number
}, { timestamps: true });

const Work = mongoose.model('Work', workSchema);

module.exports = Work;
```

- Model

- Class.js

```
const mongoose = require('mongoose');

const subdocSchema = new mongoose.Schema({
    exId: { type: mongoose.Schema.Types.ObjectId, ref: 'Exercise' },
    active: Boolean,
    deadline: Date,
});

subdocSchema.pre('validate', function(next) {
    next();
});

const classSchema = new mongoose.Schema({
    courseId : { type:mongoose.Schema.Types.ObjectId, ref: 'Course' },
    classId: {type: String, unique: true},
    staff: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
    students: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
    exercises: [{type: subdocSchema}],
    semester: String,
    tahun:Number,
    status:Boolean
}, { timestamps: true });

const Class = mongoose.model('Class', classSchema);

module.exports = Class;
```

- Course.js

```
const { text } = require('body-parser');
const mongoose = require('mongoose');

const courseSchema = new mongoose.Schema({
    title: String,
    course: String,
    order: Number,
    author: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
    contributors : [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
    modules : [{ type: mongoose.Schema.Types.ObjectId, ref: 'Modul' }],
    active: Boolean,
    status:Boolean,
    semester:String,
    status:Boolean,
    image:String,
    description: String
}, { timestamps: true });

const Course = mongoose.model('Course', courseSchema);

module.exports = Course;
```

- Exercise.js

```
const mongoose = require('mongoose');

const exerciseSchema = new mongoose.Schema({
  // exerciseId: String, // when updating an exercise, keep the exerciseId. viewed
 based on the timestamp
  courseId :  { type: mongoose.Schema.Types.ObjectId, ref: 'Course' },
  parentId : { type: mongoose.Schema.Types.ObjectId, ref: 'Module' },
  title: String, // exercise title must follow exDD-DD format
  instruction: String,
  html: String,
  js: String,
  css: String,
  testcode: String,
  enablerunraw: Boolean
}, { timestamps: true });

const Exercise = mongoose.model('Exercise', exerciseSchema);
```

```
module.exports = Exercise;
```

- Work.js

```
const mongoose = require('mongoose');

const workSchema = new mongoose.Schema({
    // exerciseTitle: String, // relationship: Exercise.title XXX howto?
    class: {type:mongoose.Schema.Types.ObjectId, ref: 'Class'},
    student: {type:mongoose.Schema.Types.ObjectId, ref: 'User'},
    exercise: {type:mongoose.Schema.Types.ObjectId, ref: 'Exercise'},
    deadline: Date,
    html: {type: String, default: ''},
    css: {type: String, default: ''},
    js: {type: String, default: ''},
    hash: String, // hash of html+css+js
    savecount: Number
}, { timestamps: true });

const Work = mongoose.model('Work', workSchema);

module.exports = Work;
```

- Worklog.js

```
const mongoose = require('mongoose');

const worklogSchema = new mongoose.Schema({
    exercise: {type:mongoose.Schema.Types.ObjectId, ref: 'Exercise'},
    work: {type:mongoose.Schema.Types.ObjectId, ref: 'Work'},
    student: {type:mongoose.Schema.Types.ObjectId, ref: 'User'},
    clientIp: String,
    html: String,
    css: String,
    js: String,
    hash: String,
    savecount: {type: Number, default: -1},
    logs: {type: Array, default: []}
}, { timestamps: true });

const Worklog = mongoose.model('Worklog', worklogSchema);

module.exports = Worklog;
```

- View
  - Course

- exerciselist.pug

```
extends ../layout

block content
  .container
    .page-header(data-id=ex._id)
      h3 #{title}'s Exercises
      h5 Created by #{user.email}

    nav(aria-label="breadcrumb")
      ol.breadcrumb
        li.breadcrumb-item(aria-current="page")
          a(href="/course")
            | All Courses
        li.breadcrumb-item(aria-current="page")
          a(href="/course/"+ex._id)
            | #{title}'s Course
        li.breadcrumb-item.active(aria-current="page")
          | Add Exercises

    if (user.role == 'teacher')
    form.form-horizontal
      .form-group
        label.col-sm-2.control-label(for='title') Exercise Title
```

```
        .col-sm-7
          input.form-
control(type='text', name='title', id='title', placeholder=' Title',autocomplete='
off')
          // ---------------------------------------------
          // Add new select form for add module exercise
      .form-group
        label.col-sm-2.control-label(for='title') Exercise Module
        .col-sm-7
          select#parentId.form-control
            each c in thisModules
              if(JSON.stringify(c._id)==JSON.stringify(ex.parentId))
                option(value=c._id, selected='selected') #{c.titleModule}
              else
                option(value=c._id) #{c.titleModule}

        button#bt-new.col-sm-2.btn.btn-primary
          i.fa.fa-plus-circle
          | Add Exercise

append end-script
  script.
    function deleteId(id){
      var r = confirm("Are you sure want to delete this exercise?");
      if(r==true){
         window.location.href = '/course/exercise/delete/#{ex._id}/' + id;
      }
    }
    $('#bt-new').click(function() {
        var newCourse = {
            courseId : '#{ex._id}',
            exerciseId : '#{ex._id}',
            title : $('#title').val(),
            parentId : $('#parentId').val(), // Hard Code
        };

        fetch('/course/exercise/#{ex._id}', {
            method: 'POST', // or 'PUT'
            body: JSON.stringify(newCourse),
            credentials: 'include',
            headers: new Headers({
            'Content-Type': 'application/json',
            'X-CSRF-Token': $('meta[name="csrf-token"]').attr('content'),
            })
        }).then(res => res.json())
        .catch(error => console.error('Error:', error))
        .then(response => {console.log('Success:', response); window.location.href
 = '/course/#{ex._id}';});
    return false;
    })


list.pug
extends ../layout

block content
  .container
    .page-header

  .container
    .row
      .col-md-8
        h3 Course List
      if (user.role == 'teacher' || user.role == 'admin')
      .col-md-4
        button.col-sm-5.btn.btn-primary(style="margin-
left:50%" type='button' data-toggle='modal' data-target='#exampleModalCenter')
          i.fa.fa-plus-circle
          | Add Course
    each c in courses
      .col-md-4(style="padding:1%")
```

```
link(href='/course/delete/' + c.id)  Delete
        .card(style="border:solid 2px grey; border-radius:10px")
          .card-header
            if(c.status == true)
              span.badge.badge-success(style="color:#25e622") Published
            else
              span.badge.badge-danger Archieved
            if(c.image)
              img.bd-placeholder-img.card-img-
top(width='100%' height='180' src='image/'+ c.image)
            else
              img.bd-placeholder-img.card-img-
top(width='100%' height='180' src='https://img.jakpost.net/c/2018/08/09/2018_08_09
_51037_1533802003._large.jpg')
          .card-body
            h4.card-title.text-dark #{c.title}
            br
            //- p.card-text With supporting text below as a natural lead-
in to additional content.

            center
            a.btn.btn-sm.btn-primary(href='/course/' + c.id)
              | Go To Course
              span.fa.fa-arrow-right
            //- .col-md-6
            a.btn.btn-sm.btn-danger(href='#',style="margin-
left:1px;", onclick='deleteId(\'' + c.id + '\')')
              | Delete
              span.fa.fa-trash
            if (c.status == 1)
              a.btn.btn-sm.btn-info(href='#',style="margin-
left:1px;", onclick='archieved(\'' + c.id + '\')')
                | Archived
                span.fa.fa-archive
            else
              a.btn.btn-sm.btn-success(href='#',style="margin-
left:1px;", onclick='publish(\'' + c.id + '\')')
                | Publish
                span.fa.fa-upload


  #exampleModalCenter.modal.fade(tabindex='-1' role='dialog' aria-
labelledby='exampleModalCenterTitle' aria-hidden='true')
   form.form-horizontal(enctype='multipart/form-data')
    .modal-dialog.modal-dialog-centered.modal-add-course(role='document')
      .modal-content
        .modal-header
          h5#exampleModalLongTitle.modal-title Course Title :
          button.close(type='button' data-dismiss='modal' aria-label='Close')
            span(aria-hidden='true') &times;
        .modal-body
          form.form-horizontal
          .form-group
            .col-sm-12
              input.form-
control(type='text', name='title', id='title', placeholder='Course Title',autocomp
lete='off')
          .form-group
            .col-sm-12
              textarea.form-
control(name='description', id='quis_desc', placeholder='Course Description', auto
complete='off', cols='50', rows="15")
          .form-group
            .col-sm-2
              label(for="inlineCheckbox1" class="form-check-label") Semester:
            .col-sm-6
              input(type='checkbox' id="js-semester" checked='' data-
toggle='toggle' data-on='Awal' data-off='Akhir' data-onstyle='success' data-
offstyle='primary' data-width="50%" data-height="40px")
```

```
        .modal-footer
          button.btn.btn-secondary(type='button' data-dismiss='modal') Close
          button#bt-new.btn.btn-primary Save changes

append end-script
  script.
    function deleteId(id){
      var r = confirm("Are you sure want to delete this course?");
      if(r==true){
         window.location.href = '/course/delete/' + id;
      }
    }

    $('#bt-new').click(function() {
        var d = new Date();
        var n = d.getMonth();
        var currentSemester = n > 8 ? 'awal' : 'akhir'
        var semesterValue = 'akhir'
        //- $(".modal-add-course").find('#js-semester').on('change', function() {
        if ($(".modal-add-course").find('#js-semester').is(':checked')) {
            semesterValue = 'akhir'
        }

        var statusValue = 0
        if(currentSemester === semesterValue){
          statusValue = 1
        }

        var formData = new FormData()

        var newCourse = {
            title: $(this).closest('.modal-add-course').find('#title').val(),
            description: $(this).closest('.modal-add-
course').find('#quis_desc').val(),
            author : '#{user._id}',
            order: 100,
            status: statusValue,
            semester : semesterValue
        };
        fetch('/course', {
            method: 'POST', // or 'PUT'
            body: JSON.stringify(newCourse),
            credentials: 'include',
            headers: new Headers({
            'Content-Type': 'application/json',
            'X-CSRF-Token': $('meta[name="csrf-token"]').attr('content'),
            })
        }).then(res => res.json())
        .catch(error => console.error('Error:', error))
        .then(response => console.log('Success:', response));
    })

    function archieved(id){
      var r = confirm("Are you sure want to archieved this course?");
      if(r==true){
         window.location.href = '/course/archieved/' + id;
      }
    }

    function publish(id){
      var r = confirm("Are you sure want to publish this course?");
      if(r==true){
         window.location.href = '/course/publish/' + id;
      }
    }

preview.pug
extends ../layout

block content
  .container
```

```
    .page-header(data-id=ex._id)
      h3 #{ex.titleModule} #{idSuccess}

      - var text=ex.module
      p!=text

    input.parentId(type="hidden" name="parentId")
    //- input(type="text" name="comment")

  .container

    .row
      h3 Comment
      .card
          #headingFour.card-header
            h5.mb-0.col-md-9
              button.btn.btn-link.collapsed(data-toggle='collapse' data-
target='#collapseFive' aria-expanded='false' aria-controls='collapseThree')
                | Komentar
                i.fa.fa-angle-down

          #collapseFive.collapse(aria-labelledby='headingFour ' data-
parent='#accordion')
              .card-body
                .form-group.shadow-textarea
                  label(for='exampleFormControlTextarea6')
                  textarea#exampleFormControlTextarea6.form-control.z-depth-
1(name="comment",rows='3' placeholder='Put your comment here')
                  br
                  button.btn.btn-success.add-comment(type='button') Submit
                  hr.solid
                ul.list-group
                  if(discussion == 0)
                    p Data not exist
                  each disc in discussion
                    li.list-group-item
                      // i.fa.fa-circle
                      .geser
                        .media
                          .media-left
                            img.avatar(src=user.gravatar(60))
                          .media-body
                            h4.media-heading.title=disc.userName
                            p.komen=disc.comment
                              br
                              p=prettyDate(disc.createdAt)
                      hr.dashed
                  -function prettyDate(dateString){
                    //if it's already a date object and not a string you don't nee
d this line:
                    -var date = new Date(dateString);
                    -var d = date.getDate();
                    -
var monthNames = [ "Jan", "Feb", "Mar", "Apr", "May", "Jun","Jul", "Aug", "Sep", "
Oct", "Nov", "Dec" ];
                    -var m = monthNames[date.getMonth()];
                    -var y = date.getFullYear();
                    -var h = date.getHours();
                    -var mi = date.getMinutes();
                    -return d+' '+m+' '+y +' '+h+':'+mi;
                -}

    #popUpScore.modal.fade(tabindex='-1' role='dialog' aria-
labelledby='exampleModalLabel' aria-hidden='true')
        .modal-dialog(role='document')
            //- .modal-content
            .alert.alert-success.fade.in
              button.close(type='button', data-dismiss='alert')
                i.fa.fa-times-circle-o
              //- for success in messages.success
              div Komentar Berhasil di tambahkan
```

```
append end-script
  script.
    var courseIds = '#{course.id}'
    var moduleIds = $('.page-header').data('id')
    console.log('/course/preview' + courseIds + '/' + moduleIds)
    $('.add-comment').on('click',function(){
      var _this = $(this);
      var _val = _this.siblings('textarea[name="comment"]').val()
      console.log(_val)

      var newQuis = {
            courseId : '#{course.id}',
            parentId : '#{ex._id}',
            userId : '#{user.id}',
            userName:'#{user.profile.name || user.email}',
            comment:_val
        };
        fetch('/discussion', {
            method: 'POST', // or 'PUT'
            body: JSON.stringify(newQuis),
            credentials: 'include',
            headers: new Headers({
            'Content-Type': 'application/json',
            'X-CSRF-Token': $('meta[name="csrf-token"]').attr('content'),
            })
        }).then(res => res.json())
        .catch(error => console.error('Error:', error))
        .then(response => {
            var courseId = $('.courseId').val()
            console.log('Success:', response);
            $('#popUpScore').modal('show');
            setTimeout(function(){
              window.location.href = '/course/preview/' + courseIds + '/' + module
Ids + '/success';
            }, 1000);

        });
    return false;
    })

  style.
    .title {
    font-size: 14px;
    font-weight:bold;
    }
    .komen {
    font-size:14px;
    }
    .geser {
    margin-left:55px;
    margin-top:5px;
    }
    hr.dashed {
    border-top: 2px dashed #999;
    }
    hr.solid {
    border-top: 2px solid #999;
    }


studentlist.pug
extends ../layout

block content
  .container
    .page-header
      if(courses.length>0)
        h3 My Courses
      else
```

```
        .alert.alert-warning(role='alert')
                    | Oops... You are not ass igned in a class, yet.
                    | Please wait a moment...

    .row
      each c in courses
        .col-md-4
          .card.mb-4.box-shadow(style='height: 100% !important; margin-top:5%')
            if(c.image)
              //- img.bd-placeholder-img.card-img-top(width='100%' height='180'
              img.bd-placeholder-img.card-img-
top(width='100%' height='180',src='/image/'+c.image, alt='Card image cap')
            else
              img.bd-placeholder-img.card-img-
top(width='100%' height='180',src='/image/dumb_image.jpg', alt='Card image cap')
            .card-body
              p.card-text(style="font-size:20px")
                | #{c.title}
              .d-flex.justify-content-between.align-items-center
                .btn-group
                  a.btn.btn-xs.btn-outline-
secondary(type='button',href='/course/preview/' + c.id, style="margin-
right:5px;font-size:15px;padding-bottom:2%") Continue Learning
                  a.btn.btn-xs.btn-outline-
secondary(type='button',href='/course/livecourses/' + c.id, style="margin-
right:5px;font-size:15px;padding-bottom:2%") Live Class
                //- a.btn.btn-link(href='/course/delete/' + c.id)  Delete

Exercise
edit.pug
extends ../layout

block content
  .page-header(data-id=ex.id)
    h3 Edit Exercise

  .container
    .row
    .editor-container
      .form-group
        label.col-sm-3.control-label(for='title') Exercise Name
        .col-sm-7
          input.form-
control(type='text', name='title', id='title', placeholder='Name', value=ex.title)
      .form-group
        label.col-sm-3.control-label(for='enablerunraw') Enable run raw code
        .col-sm-7
          input.form-
control(type='checkbox', name='enablerunraw', id='enablerunraw', checked=ex.enable
runraw)

      #exercise-editor.editor
        #instruction-editor-tab.tab.editor-2
          .editor-header Instruction
          .editor-left
            .editor-left-text.vertical-text.hide
              i.fa.fa-angle-up.fa-lg
              span Instruction
          #instruction-editor.spell-editor= ex.instruction

        #testcode-editor-tab.tab.editor-2
          .editor-header Test Code Template
          .editor-left
            .editor-left-text.vertical-text.hide
              i.fa.fa-angle-up.fa-lg
              span Test Code Template
          #testcode-editor.spell-editor= ex.testcode

  .editor-container
    #init-editor.editor
      #html-editor-tab.tab.editor-3
```

```
        .editor-header HTML
        .editor-left
          .editor-left-text.vertical-text.hide
            i.fa.fa-angle-up.fa-lg
            span HTML
        #html-editor.spell-editor= ex.html

      #css-editor-tab.tab.editor-3
        .editor-header CSS
        .editor-left
          .editor-left-text.vertical-text.hide
            i.fa.fa-angle-up.fa-lg
            span CSS
        #css-editor.spell-editor= ex.css

      #js-editor-tab.tab.editor-3
        .editor-header JavaScript
        .editor-left
          .editor-left-text.vertical-text.hide
            i.fa.fa-angle-up.fa-lg
            span JavaScript
        #js-editor.spell-editor= ex.js

  #command
    .form-group
      button#bt-update.col-sm-3.btn.btn-primary
        i.fa.fa-save
        | Update

    .form-group
      button#bt-new.col-sm-3.btn.btn-primary.left-space-1rem
        i.fa.fa-plus
        | Save New
  #run-result

append end-script
  script(src='/ace/src-noconflict/ace.js')
  script(src='/js/resizable.js')
  script.
    $('#instruction-editor-tab').resizable({handles:'e', minWidth:25});
    $('#testcode-editor-tab').resizable({handles:'xoxo', minWidth:25});
    $('#css-editor-tab').resizable({handles:'e', minWidth:25});
    $('#html-editor-tab').resizable({handles:'e', minWidth:25});
    $('#css-editor-tab').resizable({handles:'e', minWidth:25});
    $('#js-editor-tab').resizable({handles:'xoxo', minWidth:25});

    var htmlEditor = ace.edit('html-editor');
    htmlEditor.setTheme('ace/theme/monokai');
    htmlEditor.session.setMode('ace/mode/html');

    var cssEditor = ace.edit('css-editor');
    cssEditor.setTheme('ace/theme/monokai');
    cssEditor.session.setMode('ace/mode/css');

    var jsEditor = ace.edit('js-editor');
    jsEditor.setTheme('ace/theme/monokai');
    jsEditor.session.setMode('ace/mode/javascript');

    var testEditor = ace.edit('testcode-editor');
    testEditor.setTheme('ace/theme/monokai');
    testEditor.session.setMode('ace/mode/javascript');

    var instEditor = ace.edit('instruction-editor');
    instEditor.setTheme('ace/theme/monokai');
    //- testEditor.session.setMode('ace/mode/javascript');

    $('#instruction-editor-tab').resize(
        {parent: '#exercise-editor', peers: ['#instruction-editor-
tab', '#testcode-editor-tab'], minsize: 25, dir: 'horizontal'}, resizeTabs);
    $('#html-editor-tab, #css-editor-tab, #js-editor-tab').resize(
```

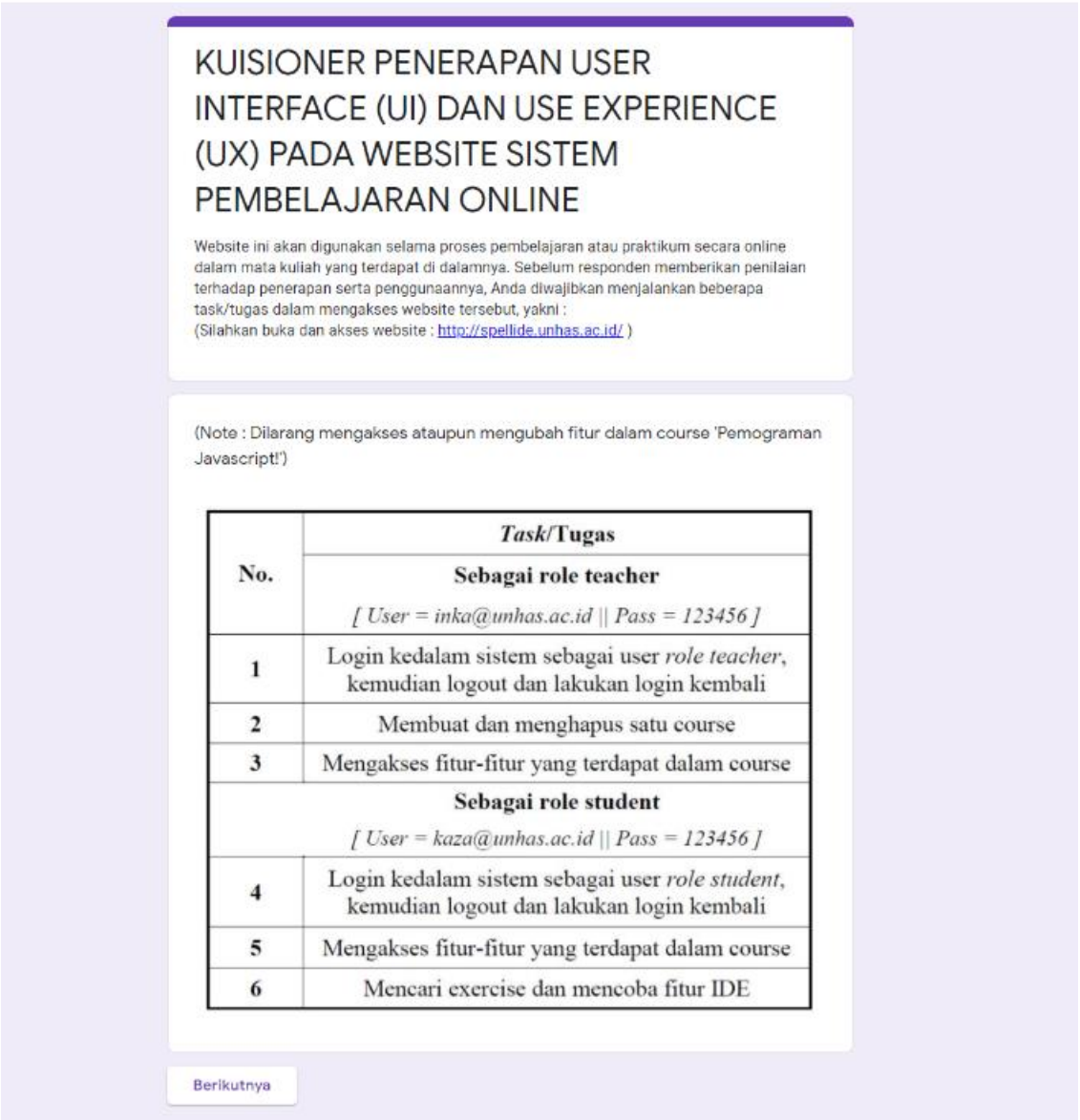
```
        {parent: '#init-editor', peers: ['#html-editor-tab', '#css-editor-
tab', '#js-editor-tab'], minsize: 25, dir: 'horizontal'}, resizeTabs);

    $('#bt-new').click(function() {
      var exercise = {
        title: 'Copy of '+$('#title').val(),
        enablerunraw: !!$('#enablerunraw:checked').val(),
        instruction: instEditor.getValue(),
        testcode: testEditor.getValue(),
        html: htmlEditor.getValue(),
        css: cssEditor.getValue(),
        js: jsEditor.getValue()
      };

      fetch('/exercise', {
        method: 'POST', // or 'PUT'
        body: JSON.stringify(exercise),
        credentials: 'include',
        headers: new Headers({
          'Content-Type': 'application/json',
          'X-CSRF-Token': $('meta[name="csrf-token"]').attr('content'),
        })
      }).then(res => res.json())
      .catch(error => console.error('Error:', error))
      .then(response => console.log('Success:', response));
    });

    $('#bt-update').click(function() {
      var exercise = {
        title: $('#title').val(),
        enablerunraw: !!$('#enablerunraw:checked').val(),
        instruction: instEditor.getValue(),
        testcode: testEditor.getValue(),
        html: htmlEditor.getValue(),
        css: cssEditor.getValue(),
        js: jsEditor.getValue()
      };

      fetch('/exercise/'+$('.page-header').data('id'), {
        method: 'PUT',
        body: JSON.stringify(exercise),
        credentials: 'include',
        headers: new Headers({
          'Content-Type': 'application/json',
          'X-CSRF-Token': $('meta[name="csrf-token"]').attr('content'),
        })
      }).then(res => res.json())
      .catch(error => console.error('Error:', error))
      .then(response => console.log('Success:', response));
    });


exerciseList.pug
cas
extends ../layout

block content
  .container
    if exercisesList
    .page-header
      h3 Exercise Student List

    ul.list-group
      each ex in exercisesList
        li.list-group-item(style="height:50px")
          // i.fa.fa-circle
          a.nounderline(href='/course/exercise/getList/'+ courseId +'/'+ ex.exerci
se + '/' + ex.student.id) #{ex.student.profile.name ? ex.student.profile.name: ex.
student.email} 
            span.pull-right
```

```
append end-script
  script.
    function deleteId(id){
      var r = confirm("Are you sure to delete class ?");
      if(r==true){
         window.location.href = '/class/delete/' + id;
      }
    }
    $('#bt-new').click(function() {
        var newClass = {
            classId: $('#classId').val(),
            courseId: '#{courseId}'
        };
        fetch('/class', {
            method: 'POST', // or 'PUT'
            body: JSON.stringify(newClass),
            credentials: 'include',
            headers: new Headers({
            'Content-Type': 'application/json',
            'X-CSRF-Token': $('meta[name="csrf-token"]').attr('content'),
            })
        }).then(res => res.json())
        .catch(error => console.error('Error:', error))
        .then(response => console.log('Success:', response));

    })

studentlist.pug
extends ../layout

append page-style
    style.
        .deadline {
            display: inline-block;
            margin-left: .5rem;
            font-weight: normal;
            font-size: 75%;
        }

block content
    .container
        .page-header
            .alert.alert-danger.alert-dismissible(role='alert')
                button.close(type='button' onclick='this.parentNode.parentNode.rem
oveChild(this.parentNode);' data-dismiss='alert')
                    span(aria-hidden='true') &times;
                    span.sr-only Close
                strong
                    center
                        i.fa.fa-warning
                        |  Info!
                //- marquee
                hr
                p(style='font-family: Impact; font-size: 12pt')
                | Spell IDE's Survey !
                a(href='https://docs.google.com/forms/d/e/1FAIpQLSdPMIrk-
GgOh3ipR1_X3uEkn2PHGec1eju0kf6XzjL_DA44Dg/viewform')
                    | click here.
                |        We are collecting your edit and run data for the instruct
ors dashboard. We may use the anonymized data for later research and analytics.


            if inClass
                if exercise
                    hr
                    h3 Exercise List
                    hr
                else
                  .alert.alert-warning(role='alert')
                    | No exercise, yet.
                    | Please wait for a moment...
```

```
            else
                .alert.alert-warning(role='alert')
                  | Oops... You are not assigned in a class, yet.
                  | Please wait a moment...

                   div.exercise-text!= ex.instruction
        if exercise
            each exer in exercise
                if exer.exercises.length > 0
                    .col-md-4
                        .card
                            .card-header.bg-primary
                                h4(style="color:white")=exer.courseId.title
                            .exercise-group(id=exer.id)
                                each ex in exer.exercises
                                    .card-header
                                        .exercise-title(data-id=ex.exId._id, data-
deadline=ex.deadline, style="padding-bottom:5%")
                                            hr
                                            a(href='/work/exercise/'+ex.exId._id)=
ex.exId.title
                                            span.deadline.badge.badge-
primary.badge-pill(style="float:right;background-color:red;")
                                                |
                                        div.exercise-text!= ex.exId.instruction
                        br
            br

append end-script
    script(src='https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.22.1/moment.js'
)
    script.
        function drawExpire(idx, el) {
            const deadlineTime = moment(this.dataset.deadline)
            this.querySelector('.deadline').setAttribute('title', deadlineTime.for
mat('llll'))
            this.querySelector('.deadline').innerHTML = 'deadline: '+deadlineTime.
from()
            if (deadlineTime.diff() < 0){
                this.querySelector('a').removeAttribute("href");
                this.querySelector('a').style.color = 'gray';
            }
        }
        $('.exercise-title').each(drawExpire)
        setInterval(function() {
            $('.exercise-title').each(drawExpire);
        }, 60000);

work
edit.pug
sac
extends ../layout

block content
  #editor-container
    #work-editor-header.page-header(data-work-id=work._id, data-
student=work.student, data-exercise-id=exercise._id, data-exercise-
deadline=exercise.deadline, data-savecount=work.savecount).wide-container
        h3 Exercise: #{exercise.title}
        div!= exercise.instruction

    #command.editor-command
      col-12.col-sm-6.col-md-8
        button#save.btn.btn-primary(role='button',style="padding:1%")
          i.fa.fa-save
          | Save
        button#run.btn.btn-primary.left-space-1rem(style="padding:1%")
          i.fa.fa-play
            |  Run
        if exercise.enablerunraw
          button#run-window.btn.btn-primary.left-space-1rem(style="padding:1%")
```

```
            i.fa.fa-play
            | Run in New Window
        if exercise.testcode
          button#run-test.btn.btn-primary.left-space-
1rem.disabled(style="padding:1%")
            i.fa.fa-puzzle-piece
            | Test
      .col-6.col-md-4
        .col-md-6(style="margin-left:12%;padding-left:0%")
          select.form-control.select-theme(name="theme",id='theme')
              each val, index in themeList
                option(value=Object.keys(val)[0])
                  | #{val[Object.keys(val)[0]]}
        .col-md-3(style="padding-left:0%")
          button#change-background.btn.btn-primary
            i.fa.fa-play
            | Change Background

    #work-editor-container.editor-resizable.wide-container
      #work-editor.editor
        #html-editor-tab.tab.editor-3(style="background-color:#D9D9D9;")
          .editor-header HTML
          .editor-left
            .editor-left-text.vertical-text.hide
              i.fa.fa-angle-up.fa-lg
              span HTML
          #html-editor.spell-editor= work.html

        #css-editor-tab.tab.editor-3(style="background-color:#D9D9D9;")
          .editor-header CSS
          .editor-left
            .editor-left-text.vertical-text.hide
              i.fa.fa-angle-up.fa-lg
              span CSS
          #css-editor.spell-editor= work.css

        #js-editor-tab.tab.editor-3(style="background-color:#D9D9D9;")
          .editor-header JavaScript
          .editor-left
            .editor-left-text.vertical-text.hide
              i.fa.fa-angle-up.fa-lg
              span JavaScript
          #js-editor.spell-editor= work.js

    button#logevent.btn.btn-default.left-space-1rem
          #log-text Log
          #log-linecount

  .col-sm-12.js-iframe(style="margin-bottom:5%")
    .col-sm-6
      h3 HTML CSS Result
    .col-sm-5.js-title-console
      h3.js-title
    .col-sm-12
      #run-result
        iframe#run-iframe
        #run-logresult
```

❖ Kuisioner



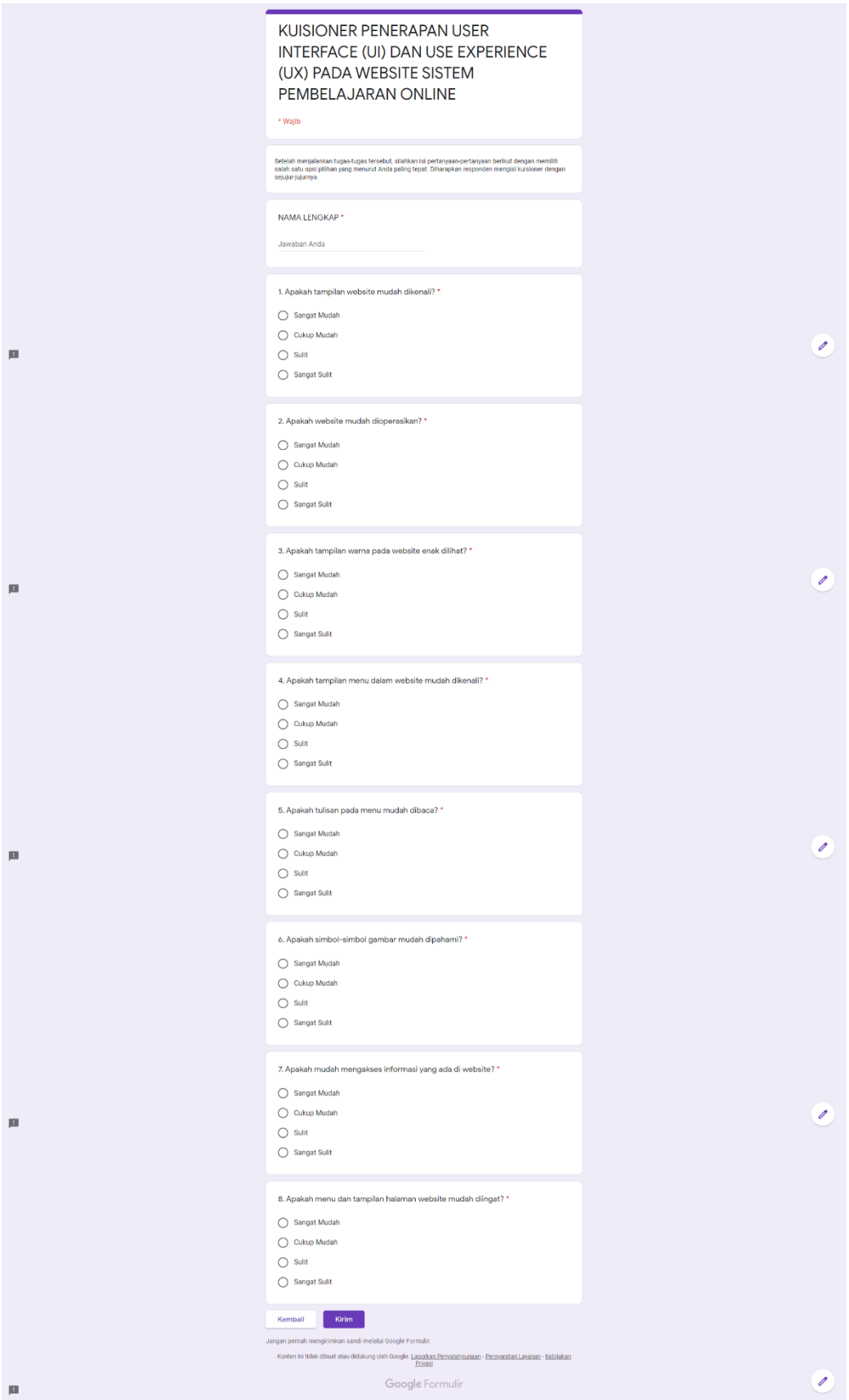
KUISIONER PENERAPAN USER INTERFACE (UI) DAN USE EXPERIENCE (UX) PADA WEBSITE SISTEM PEMBELAJARAN ONLINE

Website ini akan digunakan selama proses pembelajaran atau praktikum secara online dalam mata kuliah yang terdapat di dalamnya. Sebelum responden memberikan penilaian terhadap penerapan serta penggunaannya, Anda diwajibkan menjalankan beberapa task/tugas dalam mengakses website tersebut, yakni :
(Silahkan buka dan akses website : http://spellide.unhas.ac.id/)

(Note : Dilarang mengakses ataupun mengubah fitur dalam course 'Pemograman Javascript!')

No.	*Task*/Tugas
	Sebagai role teacher *[User = inka@unhas.ac.id || Pass = 123456]*
1	Login kedalam sistem sebagai user *role teacher*, kemudian logout dan lakukan login kembali
2	Membuat dan menghapus satu course
3	Mengakses fitur-fitur yang terdapat dalam course
	Sebagai role student *[User = kaza@unhas.ac.id || Pass = 123456]*
4	Login kedalam sistem sebagai user *role student*, kemudian logout dan lakukan login kembali
5	Mengakses fitur-fitur yang terdapat dalam course
6	Mencari exercise dan mencoba fitur IDE

Berikutnya



KUISIONER PENERAPAN USER INTERFACE (UI) DAN USE EXPERIENCE (UX) PADA WEBSITE SISTEM PEMBELAJARAN ONLINE

* Wajib

Setelah menjalankan tugas-tugas tersebut, silahkan isi pertanyaan-pertanyaan berikut dengan memilih salah satu opsi pilihan yang menurut Anda paling tepat. Diharapkan responden mengisi kuisioner dengan sejujur-jujurnya.

NAMA LENGKAP *

Jawaban Anda

1. Apakah tampilan website mudah dikenali? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

2. Apakah website mudah dioperasikan? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

3. Apakah tampilan warna pada website enak dilihat? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

4. Apakah tampilan menu dalam website mudah dikenali? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

5. Apakah tulisan pada menu mudah dibaca? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

6. Apakah simbol-simbol gambar mudah dipahami? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

7. Apakah mudah mengakses informasi yang ada di website? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

8. Apakah menu dan tampilan halaman website mudah diingat? *

- Sangat Mudah
- Cukup Mudah
- Sulit
- Sangat Sulit

Kembali Kirim

Jangan pernah mengirimkan sandi melalui Google Formulir.

Konten ini tidak dibuat atau didukung oleh Google. Laporkan Penyalahgunaan - Persyaratan Layanan - Kebijakan Privasi

Google Formulir

Timestamp	NAMA LENGKAP	1. Apakah tampilan website mudah dikenali?
12/2/2020 17:47:30	Andi Besse	Sangat Mudah
12/2/2020 18:10:59	Nur Rahmadani	Sangat Mudah
12/2/2020 18:23:31	Reskita Amelia	Sangat Mudah
12/2/2020 18:28:44	Nurfaindah Julianti Syaharuddin	Sangat Mudah
12/2/2020 23:44:58	Juan Jimmy Dwiangga AL	Cukup Mudah
12/3/2020 17:11:38	Gilbert Hyman Goes	Cukup Mudah
12/3/2020 17:12:31	Muhammad Arfani Asra	Cukup Mudah
12/3/2020 17:12:47	Akram Firmansyah	Cukup Mudah
12/3/2020 17:30:05	Fadhil Khusnul Hakim	Sangat Mudah
12/3/2020 17:55:32	Taufik Arkananta Halyb	Cukup Mudah
12/3/2020 18:09:52	Reinhart Wibisono Soplantila	Sangat Mudah
12/3/2020 18:27:14	Ikhsan Jihadi	Cukup Mudah
12/3/2020 18:45:46	ILHAM	Cukup Mudah
12/3/2020 18:59:10	Farid kemal	Sangat Mudah
12/3/2020 19:44:46	Azzahra Zalzabila Kekes	Cukup Mudah
12/3/2020 20:03:24	Arya saputra	Cukup Mudah
12/3/2020 20:24:00	Alwan Indrawan Azis	Sangat Mudah
12/3/2020 21:11:22	Kemal Idris	Sangat Mudah
12/3/2020 22:06:33	Rachmat Maulana Nur	Sangat Mudah
12/3/2020 22:08:41	Nur Hasana Abunawas	Cukup Mudah
12/3/2020 22:09:44	Muh. Amdar Febriansyah	Cukup Mudah
12/3/2020 22:12:31	ALFIAN ALDY HAMDANI	Cukup Mudah
12/3/2020 22:12:33	Atiqa luthfiah	Cukup Mudah
12/3/2020 22:12:39	Kamtina Musyfirah	Cukup Mudah
12/3/2020 22:15:48	M. Fadhlu Rahman	Cukup Mudah

2. Apakah website mudah dioperasikan?	3. Apakah tampilan warna pada website enak dilihat?
Cukup Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Cukup Mudah	Cukup Mudah
Sulit	Sangat Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Sulit	Sangat Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Sangat Mudah	Cukup Mudah
Sangat Mudah	Sangat Mudah
Sangat Mudah	Sulit
Cukup Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Sangat Mudah

4. Apakah tampilan menu dalam website mudah dikenali?	5. Apakah tulisan pada menu mudah dibaca?
Sangat Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Sulit	Sulit
Sangat Mudah	Cukup Mudah
Sulit	Sangat Mudah
Sangat Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Sangat Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah

6. Apakah simbol-simbol gambar mudah dipahami?	7. Apakah mudah mengakses informasi yang ada di website?
Cukup Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Sangat Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Cukup Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Sangat Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Sulit
Sangat Mudah	Cukup Mudah
Sangat Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Cukup Mudah	Sangat Mudah
Cukup Mudah	Sulit
Cukup Mudah	Sangat Mudah
Cukup Mudah	Cukup Mudah
Sangat Mudah	Cukup Mudah

8. Apakah menu dan tampilan halaman website mudah diingat?
Cukup Mudah
Sangat Mudah
Cukup Mudah
Cukup Mudah
Sulit
Cukup Mudah
Sangat Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Sangat Mudah
Sangat Mudah
Sangat Mudah
Sangat Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah
Cukup Mudah

❖ Data rata-rata respon time

1) Percobaan 10 host

Percobaan	respon time (ms)
1	1315
2	1325
3	1301
4	1393
5	1333
6	1373
7	1316
8	1334
9	1325
10	1306
rata-rata	1332.1

2) Percobaan 20 host

Percobaan	respon time (ms)
1	1491
2	1314
3	1452
4	1426
5	1328
6	1358
7	1367
8	1356
9	1430
10	1490
rata-rata	1401.2

3) Percobaan 30 host

Percobaan	respon time (ms)
1	1579
2	1574
3	1533

4	1504
5	1562
6	1547
7	1516
8	1562
9	1572
10	1505
rata-rata	1545.4

4) Percobaan 40 host

Percobaan	respon time (ms)
1	1694
2	1645
3	1658
4	1602
5	1682
6	1668
7	1640
8	1641
9	1631
10	1657
rata-rata	1651.8

5) Percobaan 50 host

Percobaan	respon time (ms)
1	1683
2	1675
3	1649
4	1626
5	1659
6	1701
7	1750
8	1786
9	1766
10	1721
rata-rata	1701.6

6) Percobaan 60 host

Percobaan	respon time (ms)
1	1790
2	1917
3	1916
4	1753
5	1721
6	1996
7	1731
8	1720
9	1901
10	1808
rata-rata	1825.3

7) Percobaan 70 host

Percobaan	respon time (ms)
1	1900
2	2015
3	1970
4	1946
5	1915
6	1976
7	1982
8	2052
9	2047
10	2004
rata-rata	1980.7

8) Percobaan 80 host

Percobaan	respon time (ms)
1	2047
2	2060
3	2006
4	2108
5	2072
6	2195

7	2111
8	2128
9	2080
10	2186
rata-rata	2099.3

9) Percobaan 90 host

Percobaan	respon time (ms)
1	2274
2	2286
3	2138
4	2125
5	2210
6	2138
7	2164
8	2196
9	2106
10	2210
rata-rata	2184.7

10) Percobaan 100 host

Percobaan	respon time (ms)
1	2257
2	2274
3	2376
4	2347
5	2374
6	2284
7	2388
8	2291
9	2328
10	2279
rata-rata	2319.8

LEMBAR PERBAIKAN SKRIPSI

IMPLEMENTASI *INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)* BERBASIS WEB UNTUK PEMROGRAMAN *JAVASCRIPT*

Oleh:

INKA GUSTIANY MALLISA
D421 14 517

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 28 Juli 2021. Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh Tim Penguji:

	Nama	Tanda Tangan
Ketua	Dr. Eng. Muh. Niswar, S.T., M.IT.	
Sekretaris	Dr. Eng. Zulkifli Tahir, S.T., M.Sc.	
Anggota	Dr. Amil Ahmad Ilham, ST., M.IT	
	A.Ais Prayogi Alimuddin, S.T., M.Eng.	

Persetujuan perbaikan oleh Pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr. Eng. Muh. Niswar, S.T., M.IT.	
II	Dr. Eng. Zulkifli Tahir, S.T., M.Sc.	