5 Group Work Assignment in Software Testing

Table of Contents

5.0 Introduction

The group work aims to cover most of the objectives from the course syllabus: https://www.du.se/sv/Utbildning/kurser/kursplan/?code=GMI2J3 and is graded (G / U).

Formally the project work is stated as: “A project work in a group is carried out where you report a work where you have immersed yourself in a development project, some specific technology or aids that are used in connection with software testing.”

The exercise contains elements that either can be done by a single student or a group of up to maximum 3 students. Note that if you are working in a group, the work should be a team effort!
For example, if software testing is the majority of the activity in the project (the work will probably contain some software development as well). The recommended structure is pair or “mob” work where one take turn in leading the various project work as creating user stories and test cases, write code, unit tests test and test documents etc.
It is not important to show which student makes the specific work (for example commits to Git) but it is important that all students are equally active in the work.

Talk with the teacher before deciding and/or when handing in the project charter for the group work. The project charter needs to be approved by the teacher before you formally can start with the project. The project charter is small, probably just 1-3 pages of effective content in this case.

The calculated time to spend for each student doing the group work is minimum around 40h of effective time. Since the teachers value the project around 1.5p and the labs around 3p => 4.5p.
Formula: A course is 10 weeks at 20h/week => 200h/7.5p => 27h for every point. (1.5p * 27h) = 40.5h.

5.1 Project ideas

The group work assignment allows much freedom to choose how and what you want to do to a certain extent as long as you learn the subject in question. Preferable it is existing code with something your group find interesting, something your group think is missing in the course or something you have been wanting to learn more about for a while etc.

Högskolan Dalarna
Röda vägen 3
781 88 BORLÄNGE
Telefon: 023-77 80 00
Telefax: 023-77 80 50
URL: http://www.du.se/

In lack of ideas a simple game (that already is coded) which you create tests for can be a good choice. Or maybe fork off a simple/small existing open source project of some kind that the group create tests for and eventually "develops" further. With develop in this case I mean mock some function, fix errors and handle exceptions.

Regarding existing projects it can also be a software group work from earlier courses which you may want to extend and add software testing capabilities into. For example lab 4 (or the project work) you did in: https://www.du.se/sv/Utbildning/kurser/kursplan/?code=GIK2F7 or lab in: https://www.du.se/sv/Utbildning/kurser/kursplan/?code=GMI2BT or lab done in another program course you have participated in etc.

One objective can be to perform some kind of a small "software project" (the whole V-model) as well. The focus should however lean heavily towards source code testing, the testing documentation and proper testing methods and tools usage. In short activities in the right side of the V-model.

Note that if you are "taking over" an existing source code project, you need to create User Stories or PBIs (Product Backlog Items) which are the requirements in Agile (and Scrum) development to already the created code in order to write test cases.

Some proposals which can be combined with the objectives above (if possible and time permits) is:

- Software development in other languages than mainly used during labs
  - Unit testing, unit test frameworks and Mocking in other popular programming languages than the ones in the course as: Golang, Rust, PHP, JavaScript, Java, ASP.NET Core etc.
- Agile project management systems evaluation
  - DevOps tool chains as Azure, JIRA, GitLabs, Trello etc. … see links in Learn
- Software test management and bug/issue tracking systems evaluation
  - Azure Test Plans, JIRA, GitLab, Trello etc. … see links in Learn
- Technical testing tools benefits and evaluation
  - Non-functional test tools, test automation (UI) tools as Selenium, Eggplant etc. … see links in Learn
- Technical code quality tools benefits and evaluation
  - Various test and code quality tools in IDEs and other software as http://mccabe.com/, etc. … see links in Learn
- Participate in an open source project of interest (update, fork off, extend or create one yourself) and do some development, testing, report issues etc. Treat your work as an iteration or a module in the larger project in question. Example of a community game project: https://github.com/OpenRA/OpenRA
- Your own proposal … talk with the teacher

There are some minimum requirements that the teachers want you to achieve in order to be approved for the "software project". You should see these as guidelines to conduct a project of the right size and type.

Högskolan Dalarna
Röda vägen 3
781 88 BORLÄNGE

Telefon: 023-77 80 00
Telefax: 023-77 80 50
URL: http://www.du.se/

5.2 Software testing project requirements

1. After deciding what to do via the
   - Project charter
     - In short, the project charter defines what needs to be done and gives the project manager the authority to do so. Instead, the project plan is created afterwards, and defines how activities will actually be executed.
     - https://www.google.com/search?q=project+charter+template

2. you need to begin writing
   - A limited project plan (if needed and applicable)
     - https://www.google.com/search?q=project+plan+template
   - User Stories which are the requirements in Agile development
     - A User Story is really just a well-expressed requirement
     - The 15 minute video "Agile Product Ownership in a Nutshell" is possibly the best explanation available to get an overview how working in Agile projects is done. A must view! https://www.youtube.com/watch?v=502ILHjX9EE
     - The group in this case takes all the roles in the project.
   - See point 5 below for some acceptance criteria of the project work

3. During the software testing show usage of
   - DevOps solutions (if possible)
     - Agile methods
     - Software test management tools
   - Git with commits
   - TDD (if possible)

4. Understand unit-testing, mocking, code complexity and code coverage.
   - Show usage of unit testing and mocking frameworks
   - Show usage of code coverage and code complexity tools
   - Show the use of different type of mocking (if possible)
     - Testing behavior using mocks
     - You may have needed to mock or isolate dependencies so that only bugs in the class under test trigger a failing test and is not propagating / ripples off to cause a bug in other classes or modules

5. At the delivery (group presentation and demonstration) of the solution
   - Show a project of proper size and complexity – at least 4 System Under Test (SUT) classes with dependencies
   - You project have used Dependency Injection
     - https://medium.freecodecamp.org/a-quick-intro-to-dependency-injection-what-it-is-and-when-to-use-it-7578c84fa88f
   - Objects of classes are created, not only during setup / start of app.
   - The SUT should have some form of User Interface (console, GUI, or web)
   - Software Testing Life Cycle (STLC) documents

Högskolan Dalarna
Röda vägen 3
781 88 BORLÄNGE
Telefon: 023-77 80 00
Telefax: 023-77 80 50
URL: http://www.du.se/

5.3 Software Testing Life Cycle (STLC)

After the first iteration of the

- user stories (the applications requirements) are finished (which is the basis for your testing), begin planning for tests

This means

- writing a test strategy for the project (this may be done once)

Then

- begin writing a test plan for the projects current iteration
- design and write test cases
- test environment setup
- test the software
- collect results and iterate back to "Then", or add more user stories and continue from the beginning with a full new iteration

And finally

- report the result of the testing

Help-resources for this work may be: https://www.softwaretestinghelp.com/free-online-software-testing-qa-training-course/ and some of the templates found in Learn, Course material > Various documents of value.
A detailed description of the phases in STLC: https://www.google.com/search?q=stlc+phases.
It is however not mandatory to follow every phase in the STLC depending on the projects properties.

5.4 Hand-in schedule and group presentation

Handed in for approval at latest the 10th of May 12.00 in Learn.

- Project charter of what "software testing project" the group is planning to do and who are participating in the group. Around 2-3 pages of effective content.

Handed in before the project group presentation, which means the 30th of May in Learn.
Documents and source code produced in a single zip file as

- Project plan (if needed and applicable)
- User stories
- Links to repositories with source code
- Test strategy
- Test plan
- Test cases
- Test report

Also hand in the

- Group presentation documents

Notes

- **All students must participate in the presentation even if work haven't been done**
- The project needs to ensure that time is allotted so all activities can be performed in the STLC
- If a DevOps software test management solution have been used you may have to print the web pages or export the documents to a suitable format

The allocated time to present the work and answer questions may be up to 30 minutes, but not less than 20 minutes.

Högskolan Dalarna
Röda vägen 3
781 88 BORLÄNGE
Telefon: 023-77 80 00
Telefax: 023-77 80 50
URL: http://www.du.se/