Course ID: 5364DCBS6Y

DevOps and Cloud-based software—
Course project and literature study guideline

Program: Software Engineering

Course Coordinator: Zhiming Zhao
Contact: z.zhao@uva.nl

Lecturers:
Zhiming Zhao (z.zhao@uva.nl)
Yuri Demchenko (y.demchenko@uva.nl)

Teaching assistants:
Georgia Samaritaki g.samaritaki@uva.nl

Course projects:
Siamak Farshidi (UvA) s.farshidi@uva.nl
Nafiseh Soveizi (UvA) n.soveizi@uva.nl

Spiros Koulouzis (UvA/Lifewatch VLIC) S.Koulouzis@uva.nl
Gabriel Pelouze (UvA/Lifewatch VLIC) gabriel.pelouze@lifewatch.eu
Peide Zhu (UvA/Lifewatch VLIC) peide.zhu@lifewatch.eu

Maurice Mouw (SUE) maurice.mouw@sue.nl
Nathan Keyaerts (SUE)nathan.keyaerts@sue.nl

1. Group project organization

Students will work on a software project in groups using the methods and technologies learned from the course, including web services, DevOps pipeline, Cloud computing technologies, and Agile practices. Students should choose a project from a given list (see section 4), and each project can be chosen by maximally 2 groups (3 as an exception).

project group should consist of 4 members (5 as an exception), including one product manager (contact person of the group), one scrum manager, and several developers. Note that one student can act in multiple roles in a team. Each group member is expected to offer an equal and significant contribution to the development of the project. The division of tasks within a group is such that it ensures the most effective final development powered by Agile team cooperation. Each group organizes its work independently.

Project providers ("the **customers**" of project groups) can be reached during the course. A project group should actively reach out to the customer to discuss the requirements, negotiate the milestones, and make an agreement on a specific solution. But note that the project group should not treat the customer as the technical support for solving technical problems.

Within **each project group**, students should pair up as **state-of-the-art (SOTA) teams** (2 members) to review relevant project topics. SOTA team members must be from the same project group. Detailed requirements will be explained in section 3.

Each student is required to join one **project group** and one **SOTA team**. Students can establish their project group and SOTA team via Canvas by 9/Feb. After 9/Feb, we will assign teamless students.

Each SOTA team will present the SOTA study in week six and submit a SOTA report (as a team) in week seven. Each project group should pitch the project in week three, the final results in week seven, and submit a project report (as a group) in week seven. Additional information and instruction will be provided during the lectures and via Canvas.

2. Timeline and Schedule

The following schedule is a guideline. The actual schedule may vary by group. Plan within your group.

Table 1. Recommended timeline project development.

Week	Objectives	Possible activities	Output
1	Establish project group (4 members)* and SOTA team.	1. Find project group partners and set up SOTA team pair; 2. Discuss project topics and select one; 3. Plan weekly group activities, including the SOTA, project development, reporting, milestones, etc.; 4. Configure GitLab project for group**; 5. Prepare the technologies based on an individual assignment; **6. Course project providers will be available for discussion.**	1. Fill in the Canvas project group and SOTA team; 2. Fill in the Google doc of your team members, **a product owner, a scrum manager**, and the project subject; 3. Put the Git project URL on the Google doc (find the link from CANVAS).
2	1. Agrees on the SOTA topic; 2. Agree with the project "customers" on the milestone and output; 3. Have product backlog defined.	1. Plan the SOTA activities based on the topic, e.g., scope, methods, etc.; 2. Discuss the project and collect requirements; 3. Continue with the individual assignment;	Report the SOTA topic in the Google doc (find the link from CANVAS).
3 4 5	Work on the project.	1. Work on the development backlog; 2. Review the backlog; 3. Update the project providers when necessary.	Project pitch (highlight the problem, challenge, and expected solution).
6	Finalize the SOTA reports.	1. Present SOTA results. 2. Prepare the project demo.	1. Version one project solution. 2. Initial SOTA report. 3. Present the SOTA study during lecture time.
7	Finalize and submit the project report.	1. Finalize the project; 2. Finalize the report.	1. Test report of version one; 2. Finalize all reports and submit them; 3. Present the project during the lecture

			time.

*see group guidelines in section 1.
** GitLab project group will be created for groups

Every week, we will have two times four hours of laptop college (for lab assignments, SOTA, course projects, and self-study). **Each project group** should create a git/GitHub/GitLab project and deploy the final project (e.g., web application) on the cloud (e.g., AWS, Azure, or others).

3. State-Of-The-Art (SOTA) study

In the group project, members must do a **SOTA study** to study the theoretical knowledge required by the project work. It is an important research-based learning activity in the course.

The following are the criteria for reporting the SOTA study:

1. Try to choose a topic close to the provided subject list (see 3.1)
2. Each **project group** decides how to split members into two SOTA teams and study different topics.
3. The SOTA will be reported per the SOTA team and evaluated per the SOTA team (not per project group)**.**
4. Follow the guidelines (see 3.2) for the SOTA study.

3.1. Recommended topics

The following topics for the literature review are proposed but not limited to:

1. Agile requirements engineering: best practices and risks;
2. User story quality control and analysis;
3. Backlog and sprint risk analysis and quality control;
4. Architecture sustainability and Agile practices;
5. System and Software Engineering architecture styles and use cases
6. User story-based testing in Agile and DevOps: best practices and tools;
7. How to measure and manage technical debt in Agile practices?
8. The quality metrics and measurement in the microservice applications;
9. Microservice modeling: challenges and state of the art;
10. Challenges and trends in AIOps, SecDevOps, DataOps …

3.2. Recommended steps in the SOTA study

The following best reflects the literature process to achieve the best value for your study and project:

1. Organize the SOTA study as a research process;
2. Define research questions based on curiosity;
3. Identify a list of keywords for searching the literature;
4. Search literature on the web (e.g., from IEEE/ACM digital libraries or Google Scholar). Try to search high-quality publications, e.g., based on the rank of journals and conferences or citation numbers.
5. Start from 1-2 relevant papers (presumably academic papers);
6. Understand paper contribution, find gaps, and identify new research or implementation questions;
7. Follow with finding new sources to clarify questions, gaps, and technology;

8. Discuss in a SOTA team;
9. Provide your research questions, methods for searching literature answers to the research questions, insights, etc. in a literature study report.
10. You will present the SOTA report separately from the project report.

A SOTA study takes time, so start it earlier.

3.3 SOTA presentation, report, and assessment

Each SOTA team should present the SOTA in week six during lecture time and submit a SOTA study report by the end of the course.

Each SOTA presentation will be 10 minutes of talk plus 2 minutes of discussion. **However, depending on the total number of groups, we may change the length of each slot.** Please wait for the instructions from the teachers. The presentation will be evaluated based on:

1. Topic and research questions (20%),
2. SOTA method (10%),
3. Quality of findings (40%),
4. Presentation: media usage, Q/A, and time control, etc (30%).

The SOTA report should include the key elements of the study; a sample template can be (students may adjust it):

1. **The title** of the study (clearly indicate the topic and scope of your study);
2. **Introduction** (why did you choose this topic, what questions did you want to answer from the study, how was your report structured);
3. **Related work** (are there any existing SOTA studies on the same topic? What will be the differences between your study and the existing ones?);
4. **Methodology** (how did you do the study? How/where did you find/select the literature? …);
5. **Analysis** (what did you find from the literature? How will you answer your research questions? What are the highlights? What can you derive or discover from those articles?);
6. **Discussion** (what can you learn from the analysis? What can you summarize/recommend/etc.? Weakness of the study?);
7. **Conclusions** (Key takeaways, e.g., a summary of the answers to research questions, any new/novel findings from the study, etc.);
8. **References**. Please include all literature and external sources (including websites, git repositories) that your report has cited. DO NOT simply copy any materials (including text and images) from external sources without referring to them. All reports will have to pass the similarity check. **Using generative AI, e.g., ChatGPT, for writing is restrictively forbidden.**
9. **Contributions from members.** The contribution of each member should be clearly stated in the report. We assume both team members will equally contribute to the work. However, a team can also agree on different portions of contributions if there are big differences among individual efforts. Each team should explicitly state the distribution (i.e., equally or different portions) in the report.
10. **You are encouraged to choose** IEEE two-column style[1] to write the report.

Each team should submit a SOTA report. The report should be clear and readable. Keep the report within **6 pages (if needed can be extended to 10 pages)**.

The report will be evaluated based on:

1. The selection of topic and research questions (20%)

[1] https://www.ieee.org/conferences/publishing/templates.html

2. The quality of the research method (10%)
3. The quality of references (10%)
4. The quality of analysis (e.g., answers to questions, classification of existing work) (40%)
5. The quality of discussions and conclusions (20%)

The SOTA will be graded per SOTA team; the report will be weighted as 15%, and the presentation will be 5% of the final grade.

4. Course project

Project-based learning is an important strategy of the DevOps course. A list of small projects is provided in the following example types:

1. migrate an existing application to the cloud;
2. investigate microservice application performance from an operation perspective;
3. check advanced DevOps technologies;
4. build a cloud-native application from scratch.

Students can implement those projects using the technologies learned from the course.

4.1. Project selection

A project group should read the list and discuss it with the project providers to make a choice. The project providers will be available in the lab sessions of the first week. You can also directly contact the project providers to arrange a meeting.

Please note: during the course project, the project provider will act as your "customer." You can discuss project requirements, expected output, milestones, and measures for success with the provider. But do not expect the project provider to provide you with all technical support for implementing the project; you, as the project group, should actively analyze the requirements, compare technologies, and find technical solutions. You can update your "customer" based on the agreed timeline and ask for feedback.

Your "customer" will join the evaluation of your project.

An example project：

ID	1
Title	Migrate a classical application to the cloud
Contact	Maurice Mouw
Description	A university wants to migrate its attendance system to the cloud. The solution should require minimal maintenance for the engineers who manage the AWS cloud environment, as they already have a staff shortage. One of the engineers suggested a serverless solution to reduce infrastructure maintenance to limit additional work on infrastructure for the IT department.
Requirements	1. An attendance registration system prototype with back and front-end 2. Scalability to handle peak traffic 3. Automated deployment of back and front-end 4. All code must be void of plaintext secrets
Expected output	1. Design of the created solution.

	2. Working attendance registration system. 3. A system that scales out during peak traffic. 4. A solution deployed via DevOps best practices a. Documentation is generated b. Deployed via pipeline(s) c. Secret management is applied
References	Example: https://github.com/Ubaid-Manzoor/AttendanceApp-BackEnd/blob/master/version_II/app/Services/userServices.py

5. Development tools, deployment platform, and automation pipeline

5.1. Cloud infrastructure

You can deploy the final project on AWS, as you did in the individual assignments. You can also use other providers you feel most familiar with, but TAs may not be able to offer the same level of help as AWS.

Please note that lots of public cloud (AWS, Azure, Digital Ocean) providers offer a certain number of free credits for newly registered users and large spectrum of the Free Tier resources sufficiento to perform the project . Please make use of it during your project. Please estimate the cloud cost of your project and talk to the teacher if your project needs extra cloud resources above the free tier. The course has a certain budget for cloud usage.

5.2. Versioning system, Git/GitLab

The use of Git is recommended for the group project.

Each student will receive a GitLab account at the FNWI git service (https://gitlab-fnwi.uva.nl/). Projects at GitLab will be created for you after the group membership is reported.

Optionally, the group can create a GitHub project at https://github.com/ (or use Azure DevOps Repo) As students, you can apply for a student account sufficient to organize your project group of 4-5 collaborators https://education.github.com/students

6. Project presentation, report, and assessment

During the project, each group will present the results and submit the report in week 7.

6.1 Project pitch

Each group should pitch the selected project in week 3.

- What is the project about, e.g., scope, business scenario, user stories, etc.?
- What is your plan, e.g., actions, risk analysis, etc.?
- What are your expected outputs?

6.2. Final project presentation

Each project group will give a final presentation in week 7; each group has 12 minutes, including 8 minutes of talk and 4 minutes of discussion. (Time can be adjusted based on the actual number of groups.)

The presentation will be evaluated based on:

1. Structure and content of the story (70%), the content should cover

        1. Your project, e.g., motivation, value proposition, requirements, technologies, implementation, etc.
        2. Management and DevOps, e.g., DevOps pipeline, Agile/Scrum, Risk management, etc.
        3. The final product, e.g., the status of the software/service, short demo, etc.
    2. Use of media (10%), e.g., using table/plots/animation/etc. to make the story more clear and easy to follow.
    3. Answers to the questions (20%)
        1. How do you respond to the questions

The project presentation will be graded 10% of the final grade. Each group, please upload your slides after your talk.

6.3 Project report

The project report explains the design choices made during the realization of the project. A good report contains at least the following parts:

1. Title of the project.
2. Authors: Group name/ID, group members' names, including student ID.
3. **Introduction**: describe the background of the application area and the objective of the project.
4. **Related work** (are there any existing projects/systems with the same objectives? What will be the differences between your project and the existing ones?) Do not repeat the SOTA; only focus on the work related to your project and the differences.
5. **Development of the project**, including such sections as (1) use cases analysis and requirement analysis and specification, (2) system architecture,(3) design technical solution choices, (4) development and implementation including technical details of each component, component (service) interface, system integration and demonstration, with the reference to the application code (structure and documentation), etc.
6. **Operation of the system**, including details like utilized cloud infrastructure and services, deployment, how the runtime system looks like, monitoring of the system, studies on the system performance, etc.?
7. **Management of the project,** including details like how agile practice is applied, how the team is organized, how activities are managed, lessons learned, etc.
8. **DevOps practices**, including automated testing, integration, and deployment pipeline.
9. **Discussion**: a reflection on the result. Does the development process go as planned? What is not, and why not? Experience with Scrum, Kanban, or other cooperative models? Weakness to be improved, etc.
10. **References**. Please include all literature and external sources (including websites and git repositories) cited in your report. DO NOT simply copy any materials (including text and images) from external sources without referring to them. All reports will have to pass the similarity check.
11. **Contributions from members.** The contribution of each member should be clearly stated in the report. We assume both team members will equally contribute to the work. However, a team can also agree on different portions of contributions if there are big differences among individual efforts. Each group should explicitly state the distribution (i.e., equally or different portions) in the report.

In the report, please include the following information: screen snapshot of the system, backlogs of the development (if any), and location of the external resources, e.g., GitHub (if needed).

The example content described above is a guideline. The report should be clear and at a sufficient level of detail. Keep the report within 15 pages (in IEEE/ 2-column style).

6.4. Assessment

The project will be assessed by presentation (in the last week), final report (including supplementary materials, e.g., source code/docker images, scrum backlogs (e.g., in CSV or JSON), and a video of the demo*) (submitted by the end of week 7). **The project will be graded per project group; the report will be 50%, and the presentation will be 10% of the final grade.** Group members may receive different grades based on their actual contributions stated in the report.

*A video of the final demo is recommended. A high-quality video may get a bonus of up to 5% on the final grade. Video quality will be evaluated based on the demo story, system features, usability and highlighted innovations, etc.

The presentation will be evaluated based on:

1. Structure and content of the story (60%)
2. Use of media (10%)
3. Answers to the questions (30%)

The project report will be assessed by the lecturer(s) and teaching assistant(s). The report will be assessed based on:

1. **Quality of the project (40%)**, including the i) requirement analysis, ii) architecture design, iii) design and implementation, iv) quality of the final system, v) innovation of the project, vi) utilization of the cloud services, vii) technical challenges being tackled, etc.
2. **Quality of project management (agile) aspect (20%)**, i) efficiency, ii) deviation from the original plan, iii) quality of collaboration, iv) Utilization of DevOps practices and technologies, v) the quality of product/sprint backlogs. Students can freely choose the style to present those points but make the key messages explicit and clear.
3. **Quality of the software (20%)**: i) executability, ii) documentation, iii) versioning management of the project, iv) software/docker images.
4. **Quality of the project discussion (10%)**: i) insights, ii) lessons learned
5. **Quality of the report writing (10%):** i) readability, ii) clearness, iii) level of detail

The project will be graded per project group; the report will be 50%, and the presentation will be 10% of the final grade.