Fundamentals of Software Architecture

Assignment 4: Architectural Styles and their Applications

Rebekka Wohlrab

In Assignment 4, you will work with three separate types of systems. One system is constructed in a Pipe-And-Filter style architecture (Image Manipulation Filters), the second in a Web-Services style, and the other made in the form of Microservices architectural style.

You implement a list of features, create a screencast, and answer questions about your architectural choices.

Grading Criteria

The assignment is worth 100 points. The point breakdown is as follows.

- Module 1: 30 points
- Module 2: 35 points
- Module 3: 35 points

To pass this assignment, you need to pass each module (i.e., you need at least 50% of the points in each module). We will evaluate your submission using your Gitlab commits and using your screencast videos (see below).

Evaluation

Your implementation will be evaluated based upon the following criteria:

General

- Make sure to create the solutions according to the descriptions in the README files located in each module.
- Fill out the Assignment 4: Questions and Diagrams template and provide answers to all questions.
- The features should be implemented while adhering to the architectural style in the particular module.
- The quality of code written to meet the features should be clear, well documented, and easy to follow.
- The code should compile successfully, without crashing due to unhandled errors. Any exceptions should be handled accordingly.
- **Mandatory screencast**: Create one video per module and demonstrate all of the implemented features in the module, each with an example.
- **Mandatory session:** In order to pass, you are required to attend the mandatory TA session in which your team will have to answer questions about your solutions and the general architectures. Further details about the date and time can be found on the Canvas homepage in the Group Assignments section.

Screencast guidelines

- For each module, create a 2-5 min video.

- The screencasts are mandatory, and you have to explain how you solved the different parts of each module. Make sure that you show that you understand the systems you have implemented.
- In the screencasts, briefly showcase the functionalities of the sections you have developed. Navigate through different features, use cases, and scenarios to highlight how users interact with the system.
- Briefly explain the underlying software architecture of your system. Describe the high-level components, their interactions, and how they contribute to the system's functionality.
- Narration: Provide clear and articulate narration while walking through the system. Explain each feature and interaction step-by-step.
- Visual Clarity: Ensure that the screencast captures the system interface and interactions clearly. Use appropriate zoom-ins and callouts to focus on specific elements when necessary. You can speed up the video to meet the 2-5 min constraint.

Evaluation Criteria:

- Clarity and Presentation: The clarity of your explanations and the effectiveness of your demonstration.
- Functionality Showcase: The thoroughness and relevance of the system functionalities you showcase.
- Architecture Understanding: The accuracy and comprehensiveness of your software architecture overview.

How to Submit:

- **Code:** The code you will develop for the different modules is to be pushed to Gitlab. The CI system will give you automatic feedback on the correctness of your system.
  Obviously, you are not supposed to change anything in the gitlab-ci.yml file. You are not allowed to make any changes to the test repository.
- **Questions & Diagrams:** The questions and diagrams are to be added to the template document in the Assignment 4 page (A4 Questions and Diagrams) and submitted to Canvas.

How to Submit

Module 1: Pipe-And-Filter – 30 points

Part 1: Completion of filters – 20 pts

- Filter 1: Flipping image horizontally 6pts

● This filter should mirror the image horizontally. It is expected and written in the documentations that you are not to use the premade OpenCV functions and instead create your own algorithm. E.g., use nested loops to store flipped pixels on a secondary image and then return the new flipped image.

- Filter 2: Flipping image vertically 6 pts

● This filter should mirror the image vertically. It is expected and written in the documentations that you are not to use the premade OpenCV functions and instead create your own algorithm. E.g., use nested loops to store flipped pixels on a secondary image and then return the new flipped image.

- Filter 3: Greyscale image 8 pts

● This filter should turn the image to greyscale. It is expected and written in the documentations that you are not to use the premade OpenCV functions and instead create your own algorithm. E.g., for every pixel, extract pixel information for RGB channels, and perform calculations to turn it to a shade of gray.

Partial solutions are worth 0 points. Automated CI and the screencast will be used to determine the correctness of your solution. Make sure to run your pipeline and show the generated images.

Part 2: Questions - 5 pts

Edit the A4 Questions and Diagrams template (on Canvas) and provide answers to the questions. It is expected of all answers to be justified, using argumentation and, if necessary, examples. 1 pt per correct answer.

Part 3: Component diagram – 5 pts

Create a component diagram of the system you have created where you show in detail the different components of your pipe and filter system and how they interface.

Module 2: Web-Services – 35 points

● Part 1: Modifications to the System - 23 pts

To refresh your knowledge on web services, check out the Web Services – An Overview page on Canvas: https://canvas.gu.se/courses/68673/pages/optional-web-services-an-overview?module_item_id=950851

- **Feature 1: Send request for creating a student under a course (Frontend) 7pts**

  Get the student's information from the form and add the student under the correct course. In case of success, the web page should prompt the user with a success message (of any kind). Otherwise, an informative message regarding the issue shall be displayed. All possible crashes/exceptions/errors shall be handled. The student model is pre-made and its relationship to the course model can be seen under the models in backend. Create the students based on the models and according to the relationship between a student and a course. The endpoint for this function is also missing in the backend.

- **Feature 2: Send request for retrieving the list of students of a course (Frontend) 5pts**

  Get the list of all students enrolled in this course. All possible crashes/exceptions/errors shall be handled. The student model is pre-made and its relationship to the course model can be seen under the models in backend. The endpoint for this function is also missing in the backend.

- **Feature 3: Endpoint for posting new students under a course (Backend) 6pts**

  Implement the endpoint for creating/adding a student to the list of students of a course. For the most important status/response codes (such as 500, 404, 201), appropriate messages shall be shown. The created student object shall exist independently of the course and shall also be found in the list of all students (not just connected to a course).

- **Feature 4: Endpoint for getting the list of all students of a course (Backend) 5pts**

  Implement the endpoint for getting the list of all students of a course. For the most important status/response codes (such as 500, 404, 201), appropriate messages shall be shown.

Partial solutions are worth 0 points. Automated CI and the screencast will be used to determine the correctness of your solution. A solution is considered complete when all the tests made for it pass and it covers all the other requirements stated for that feature.

● Part 2: Questions - 6 pts

Edit the A4 template (A4 Questions and Diagrams on Canvas) and provide answers to the questions. It is expected of all answers to be justified, using argumentation and, if necessary, examples. 1 pt per correct answer.

● Part 3: Component diagram – 6 pts

Create a component diagram of the system you have developed. Show in detail the different components of your system and how they interface.

Module 3: Microservices – 35 points

● Part 1: Modifications to the system – 15 pts

For all modifications, keep track of the following measures – you will need it for Part 2:

- The number of modules affected
- The number of new modules that were required
- Time to make the new changes
- Extent to which the changes affect other elements of the system

● Feature 1: Delete an order by order_id

● With accordance to the architectural style, the feature to remove an order from the orders table shall be implemented, taking the id as a parameter. Any unhandled exceptions should be handled (i.e., the system should not crash). Make sure you don't violate the microservices style.

● Feature 2: User Registration and Authentication

● Before any access to the variety of services in the application, a user shall be able to log in with a **username** and a **password.** If the credentials are not present, users should be able to sign up. Adhering to the architectural style, the **feature** should be implemented so that access is only granted to signed-in users.

● Feature 3: Logging Activity and Faults

● Adhering to the architectural style, all important actions should be logged. Concretely, each action shall be stored with the username that conducts the operation, as well as which action was conducted (Create/Retrieve/Delete). Any exceptions should also be recorded.

● Part 2: Data Collection and Reporting – 10 pts

● The content and quality of your write-up. You should reason about the difficulty of making modifications to this system. Describe the modification measures you collected for each implemented feature and discuss the process of making the modifications to the provided system. Reflect about:

● The number of modules affected

● The number of new modules that were required

● Time to make the new changes

● Extent to which the changes affect other elements of the system

● Part 3: Architectural Discussion – 10 pts

● **Choices** made in the implementation should be justified appropriately, along with the **impacts** of those decisions on the quality and functionality of the system, as well as on the refactoring process. Be clear, concise, and complete (make sure your discussion is **architectural and not code-centric**).

- Architectural diagram (e.g., component diagram and/or module diagram) should be **correctly labelled** and well-structured, providing a clear representation of the topological **structure** of the components, and how they **communicate**.
- The architectural diagram should also be consistent with the architecture style that they concern.

Upon completing the modules, your group must take part in a mandatory TA demonstration session, where you will be asked about the recorded screencast, as well as your development process and architectural thought-process. If you do not schedule a meeting with your teaching assistant, then this will result in 0 points for the assignment.

Criteria	**Ratings**	**Pts**
Module 1	**+1 pt** Per correct question **+5 pts** Component diagram follows UML guidelines and shows all components of the system and how they interface **20 pts:** CI declares images to be filtered correctly, no one-line functions are used, Screencast showcases pipeline running and filtered images being generated. **18 pts:** Solutions appear correct, no one-line functions are used, but project pipeline does not	30

Criteria	Ratings	Pts
	pass. Screencast showcases pipeline running and filtered images being generated. **14 pts:** Students don't complete one of the "flip filters", no one-line functions are used, but the screencast shows that the other two filters work. CI does not pass. **12 pts:** Students implement both flip filters but not the greyscale, no one-line functions are used, screencast shows both filters working. **8 pts:** Only greyscale filter is implemented, no one-line functions are used, shown with screencast. **6 pts:** Only one of the flip filters is implemented, no one-line functions are used, shown with screencast. **0 pts:** No filters are completed OR filters are not separated according to pipe and filter architecture OR one-line functions are used.	
Module 2	**Per Question** **+1 pt Clear Understanding:** The answer demonstrates a clear understanding and justification of the concept for the question. **+0.5pt Partial Understanding:** The answer demonstrates a partial understanding of the concepts but lacks clear and coherent explanations. **+0pt Inadequate/No Response:** The answer does not address the question at all or provides extremely vague, irrelevant, or incorrect information. **Feature 1** **7 pt Full implementation:** Full and correct implementation of the feature is provided. **0pt Partial/Inadequate Implementation:** The implementation does not address all the requirements. **Feature 2** **5 pt Full implementation:** Full and correct implementation of the feature is provided. **0pt Partial/Inadequate Implementation:** The implementation does not address all the requirements. **Feature 3** **6 pt Full implementation:** Full and correct implementation of the feature is provided. **0pt Partial/Inadequate Implementation:**	35

Criteria		Ratings					Pts
		The implementation does not address all the requirements. **Feature 4** **5 pt Full implementation:** Full and correct implementation of the feature is provided. **0pt Partial/Inadequate Implementation:** The implementation does not address all the requirements. **Component diagram** **+5 pts** Component diagram follows UML guidelines and shows all components of the system and how they interface					
Module 3	Pt. 1	15 - General requirements met - All **3 modifi-cations** added to system	12 - General requirements met - All **3 modifications** implemented in the system	10 - General requirements met - Modifications only done for **2 of 3 sub-parts**	5 - General requirements met - Modifications only done for **1 of 3 sub-parts**	0 - General requirements **not met** **OR** - Implementation **missing** for all of the required sub-parts **OR** - Does not follow microservice **architecture's common practices**	35
	Pt. 2	10 - Data regarding **modified modules** (new/modified) - Time taken to modify as well as - Coupling with other modules, **analyzed and presented**, for each implemented feature from Part 1.		5 - Incomplete data collected - Minimal reflection about the modifiability of the system, a		0 - Not answered in the A4 Questions and Diagrams template.	
	Pt. 3	10 - **Analyzed thoroughly** about architectural choices made, with their impact on the system	8 - **Described** architectural choices with good amount of justification relevant to the system	5 - **Described** architectural choices with satisfactory amount of justification behind them	2 - **Outlined** few architectural choices with minimal to no justifications	0 - No screencast OR - No architecture diagram provided OR - No description of architectural choices	

Criteria		**Ratings**					**Pts**
		AND - Provided architecture diagram **accurately represents a microservice-oriented topology** of the system with minimal anti-patterns.	AND - Provided architecture diagram **partially shows** a microservice-oriented topology.				