First assignment

The main **goal** of the first assignment is to explore the different types of design decisions in a project, and to focus specifically on the components and connectors of the respective project. Specifically, we aim to answer two research questions:

1. What types of design decisions are discussed in the issue tracker of a project?
2. What are the differences between architectural components, connectors and configurations in issue trackers versus source code?

Each group will focus on a specific project. The table below shows which project belong to which group and provides the required information for each project.

Group	**Project**	Domain	Documentation & code
Group 1	Struts2	Web	https://struts.apache.org/ https://github.com/apache/struts
Group 2	ActiveMQ	Middleware	activemq.apache.org/ github.com/apache/activemq
Group 3	ActiveMQ	Middleware	activemq.apache.org/ github.com/apache/activemq
Group 4	Axis2	Middleware	axis.apache.org/axis2/java/core/ github.com/apache/axis-axis2-java-core
Group 5	Axis2	Middleware	
Group 6	JClouds	Cloud	jclouds.apache.org github.com/apache/jclouds
Group 7	ActiveMQ	Middleware	activemq.apache.org/ github.com/apache/activemq
Group 8	Tika	Content	tika.apache.org/ github.com/apache/tika
Group 9	Wicket	Web	wicket.apache.org github.com/apache/wicket
Group 10	Derby	Data	db.apache.org/derby/ github.com/apache/derby
Group 11	Derby	Data	
Group 12	Log4j2	Dev tools	logging.apache.org/log4j/2.12.x/

To achieve our goal and answer the research questions, we will follow multiple steps in the next weeks:

Week 1:

- Step 1: Explore architecturally significant requirements of a project.

Skim the user guide and websites documentation for functional requirements and quality attributes of the project. List the functional requirements and quality attributes of the project in max 2 pages. In listing the requirements and quality attributes, please try to be specific and focus on main features rather than detailed features. The aim of this step is to provide you with initial understanding about the project. Do not read every documentation part, otherwise it can take you longer period. Try to focus on the main features. Please note that this list might not be perfect, but it will give you an initial overview about the architectural requirements of this project.

- Step 2: Explore types of design decisions from issue trackers.

Given the provided list of issues, apply the steps of qualitative content analysis to classify issues based on the types of design decisions in each issue description. We focus on the types: Existence, property and executive (see slides in lecture). An issue description can contain zero, one, two or the three types of design decisions. For each issue, we should provide the following:

- If the issue is architectural or not (Yes/No): This can be determined if the issue is significant. You can determine this either from the issue description directly or going to the Github repository and search using the issue ID. If the issue is not significant, then it is non-architectural, and other below points can be ignored.
- If the issue is architectural, then what types of design decisions does the issue description contains? (Existence/Property/Executive). If the issue description has no architectural decisions, then it is non-architectural and can be ignored.
- For existence issues, does the issue contains descriptions of components design, (Yes/No) (see slides lecture). Here please check issue description and comments as well.
- For property issues, what quality attributes are discussed in an issue.
- Justification of the classification.

You can access each issue by concatenating the issue ID with the URL https://issues.apache.org/jira/browse/

Before starting with the classification, please read the existing coding book (attached). The analysis should follow the following qualitative process: At the beginning, group members should classify few issues together to reach clear understanding of the three types of design decisions and formulate coding rules. These rules should be added to a coding book. After this, each member should classify part (not all) of the issues alone. This should be followed by a small check with other team members for some issues. After this, the group members should classify the rest of the issues. By the end of the classification, the three members of the group should calculate Kappa coefficient for a random sample of issues, around 10% to ensure the agreement on the classification.

By the end of Step 2, you can answer the research question:
What types of design decisions are discussed in issues?

Week 2:

- Finalize step 2.

- *Step 3:* Based on your analysis from Step 2, select max 15 issues that involve the biggest descriptions of components, connectors, and configurations. These descriptions could be only in the descriptions or in both the descriptions and comments. These issues should have been classified as architectural existence decisions. One property that can guide in selecting issues is the size of description and comments.

  After selecting issues perform the following two steps for each issue:

  1. *Analyze source code changes to determine added components, connectors and configuration changes per issue*: To achieve this step, follow these sub-steps:

a. Determine Github commits that implement the code changes of each issue.
b. Determine the parent of each commit.
c. Use the tool Understand (https://scitools.com/) to analyze the differences between each commit and its parent commit. The tool can show added, removed, modified classes and packages. It also provides models that can help to show the changes. Note: The tool is free for students and researchers. You need to request an academic license to use the tool.
d. Using Understand, determine added, removed and modified components, connectors and configurations.

2. *Analyze components descriptions in issue description and comments*: Determine the components, connectors and configurations discussed in the issues. Components could be class names or general terms from the project, while connectors appear as verbs that connect components. See example in lecture slides. Draw a simple model that shows the components, connectors, and their configurations.

It is expected for each issue to have an analysis that involve:
a) A model that reflects added, removed, and modified components, connectors and configurations in source code for each issue.
b) A model with the components, connectors and configurations in each issue.
Using both models, you can compare the components design from issues and source code to determine differences in components, connectors and configurations.

Week 3:

- Finalize step 3.

By the end of Step 3, you can answer the research question:
What are the differences between architectural components, connectors and configurations in issues and source code?

- Write the report and prepare presentation.