

# Group 22B - Assignment 2

## Task 1

### Class level metrics:

We will be using the following class level metrics to identify which classes should be refactored:

#### ATFD - Access to Foreign Data:

As per - M. Lanza, R. Marinescu. Object-Oriented Metrics in Practice. Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems. Springer-Verlag Berlin Heidelberg, 2006. The desirable value of this metric is less than 6.

#### CBO - Coupling Between Objects:

According to - R. Shatnawi, W. Li, J. Swain, and T. Newman. Finding software metrics threshold values using ROC curves. J. Soft. Maint. Evol., 22(1):1–16, 2010. The threshold value of this metric is 13.

#### NPA - Number of Public Attributes:

As per - M. Lanza, R. Marinescu. Object-Oriented Metrics in Practice. Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems. Springer-Verlag Berlin Heidelberg, 2006. Desirable value of this metric is less than 4.

#### RFC - Response for a Class:

According to - R. Shatnawi, W. Li, J. Swain, and T. Newman. Finding software metrics threshold values using ROC curves. J. Soft. Maint. Evol., 22(1):1–16, 2010. The threshold value of this metric is 44.

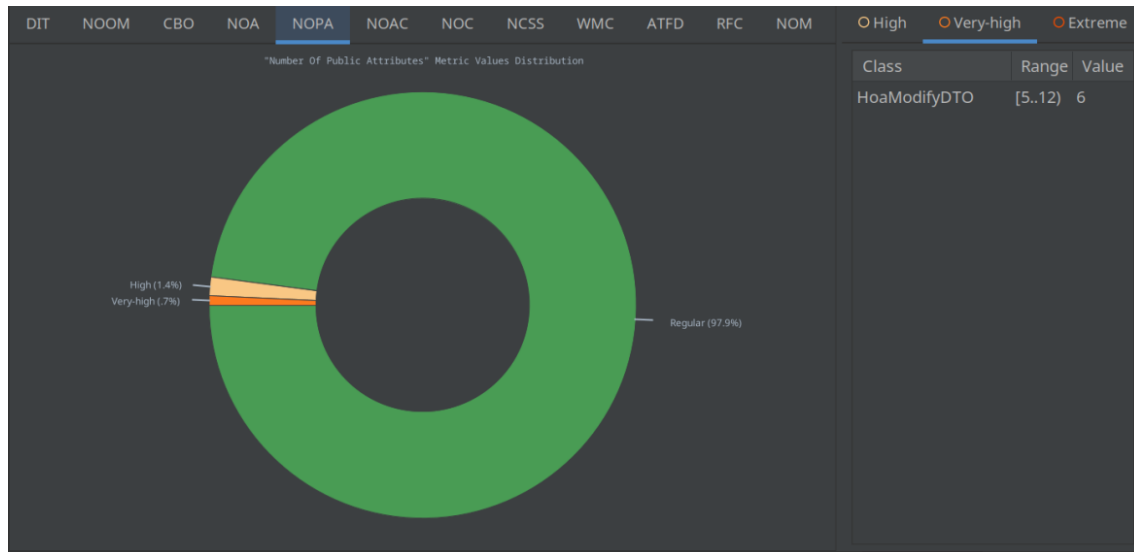
#### WMC - Weighted Methods per Class:

In line with - Filó, T. G., Bigonha, M., & Ferreira, K. 2015. A catalogue of thresholds for object-oriented software metrics. Proc. of the 1st SOFTENG, 48–55.

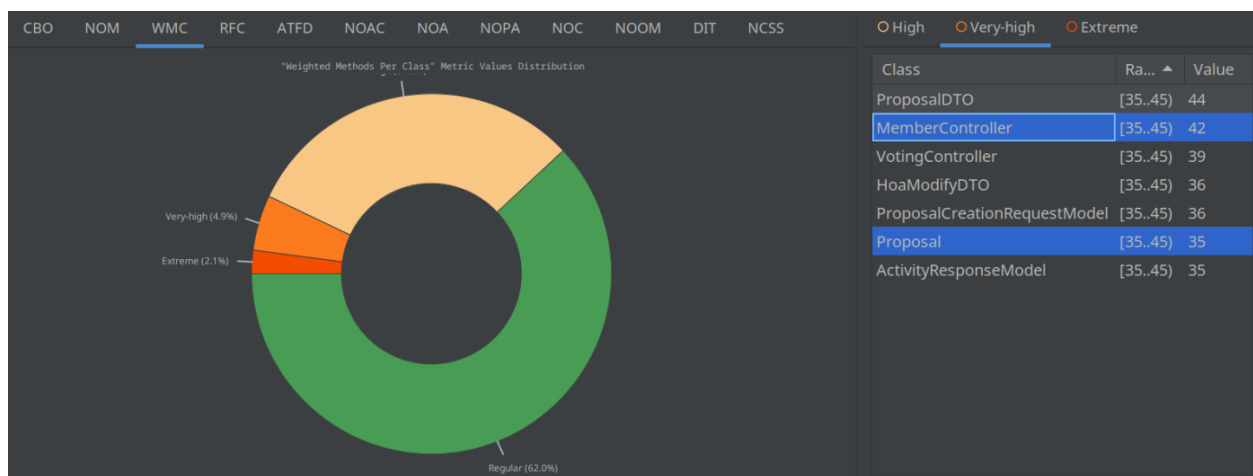
Common value for this metric is less than 12, casual - between 12 and 34, uncommon is greater than 34. Hence we'll be taking the threshold to be 34.

Class	ATFD	CBO	DIT	NCSS	NOAC	NOA	NOC	NOM	NOOM	NOPA	RFC	TCC	WOC	WMC
VotingController	10	30	1	146	0	2	0	9	0	0	83	0.75	1.0	39

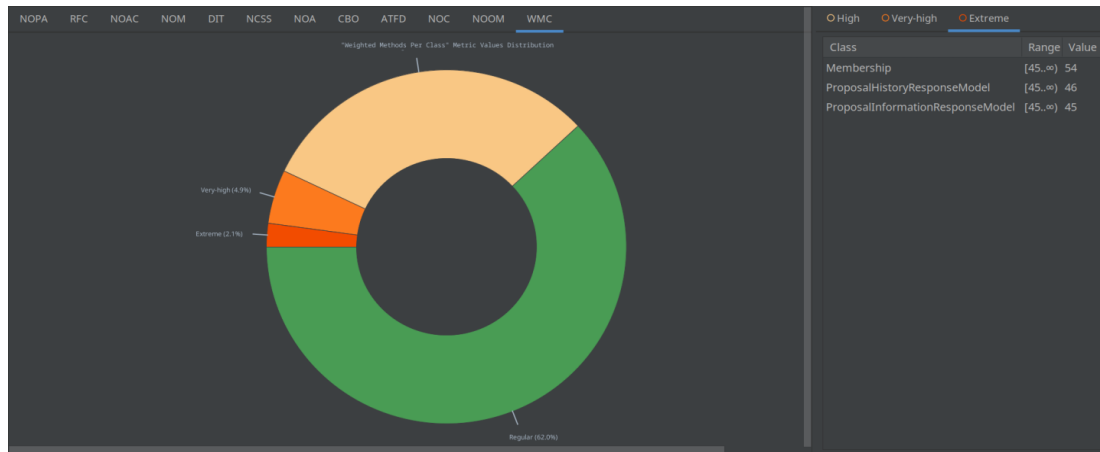
We found one large class (**VotingController**) which fell into the extreme category for multiple metrics, namely: **ATFD**, **CBO** and **RFC**.



Using the **NPA** metric, we have identified a class which goes over the thresholds and hence discovered that **HoaModifyDTO** has getters and setters for every attribute, while also having all its attributes set to public and accessed directly from many methods.



Using the weighted methods per class metric, we have identified two classes which go over the threshold ( $>34$ ), namely **MemberController**, and **Proposal**.



*As we've been recently informed, lombok annotations should not be regarded as code smells, therefore we have replaced the below three methods with other refactoring above. We decided to leave this in the document nonetheless, but we do not count it towards the 5 classes refactored.*

Using the weighted methods per class metric, we have identified three classes in the extreme category (>45) that needed refactoring. The **Membership** class and the response models **ProposalHistory** and **ProposalInformation**.

## Method level metrics:

We will be using the following class level metrics to identify which methods should be refactored:

### Lines of code (LOC):

According to Filó, T. G., Bigonha, M., & Ferreira, K. 2015. A catalogue of thresholds for object-oriented software metrics. Proc. of the 1st SOFTENG, 48–55. The desired metric is 30.

### McCabe Cyclomatic Complexity (CC):

According to Filó, T. G., Bigonha, M., & Ferreira, K. 2015. A catalogue of thresholds for object-oriented software metrics. Proc. of the 1st SOFTENG, 48–55. The desired metric is around 4.

The MetricTree plugin was used to identify 5 methods which needed refactoring, namely

<div><div>▼</div><div>m</div><div>registerUser(Username, Password, FullName)</div><div><div>M</div><div>○</div>Condition Nesting Depth: 2</div><div><div>M</div><div>○</div>Halstead Difficulty: 21.0</div><div><div>M</div><div>○</div>Halstead Effort: 8792.379</div><div><div>M</div><div>○</div>Halstead Errors: 0.142</div><div><div>M</div><div>○</div>Halstead Length: 83</div><div><div>M</div><div>○</div>Halstead Vocabulary: 33</div><div><div>M</div><div>○</div>Halstead Volume: 418.6847</div><div><div>M</div><div>○</div>Lines Of Code: 35</div><div><div>M</div><div>○</div>Loop Nesting Depth: 0</div><div><div>M</div><div>○</div>Maintainability Index: 47.6417</div><div><div>M</div><div>○</div>McCabe Cyclomatic Complexity: 11</div><div><div>M</div><div>○</div>Number Of Loops: 0</div><div><div>M</div><div>○</div>Number Of Parameters: 3</div></div>
---

As you can see by the materials provided these 5 methods all go over the metric set for acceptable Lines of code (30) which is a method level code smell.

## Task 2

### Class level metrics

We split the **VotingController** class into three separate classes as it was quite bloated with long and complex methods. This drastically improved many metrics, namely: **ATFD**, **CBO**, **NOM**, **RFC**, **WMC**.

Before:

Class	ATFD	CBO	DIT	NCSS	NOAC	NOA	NOC	NOM	NOOM	NOPA	RFC	TCC	WOC	WMC
VotingController	10	30	1	146	0	2	0	9	0	0	83	0.75	1.0	39

After:

Class	ATFD	CBO	DIT	NCSS	NOAC	NOA	NOC	NOM	NOOM	NOPA	RFC	TCC	WOC	WMC
VotingManagerController	4	17	1	54	0	2	0	2	0	0	33	0.0	1.0	14
VotingInteractionController	5	11	1	39	0	2	0	4	0	0	39	0.3333	1.0	12
VotingInformationController	6	12	1	57	0	2	0	5	0	0	39	1.0	1.0	15

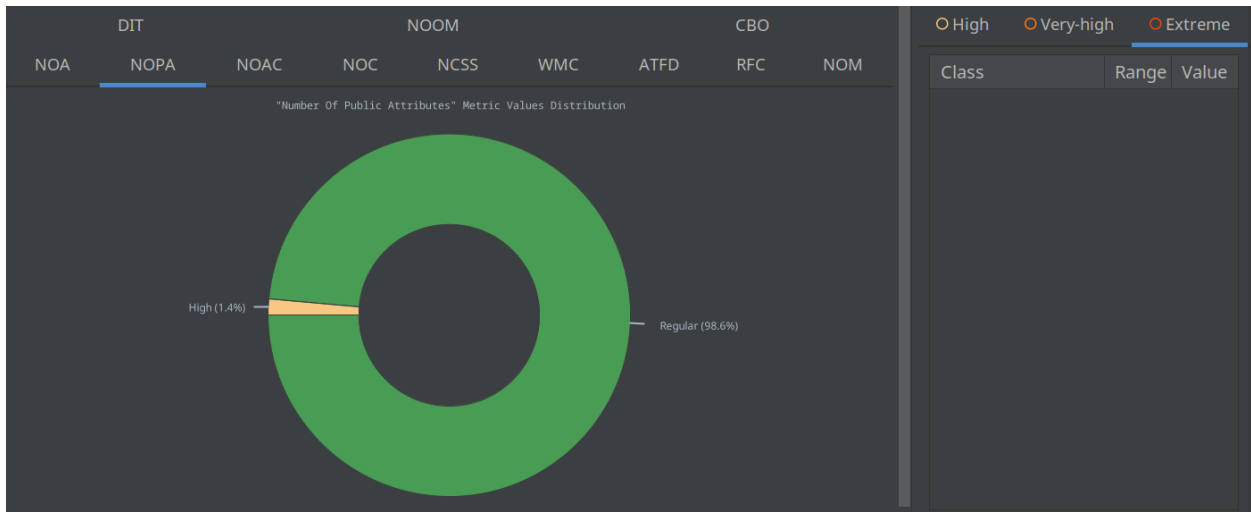
We refactored the **MemberController** class similarly, to improve the WMC metric, as it contained a lot of complex methods. The class was split into two, **MemberAttributeController** and **MemberInfoController**.

**MemberController**    ->    **MemberAttributeController**    **MemberInfoController**

<ul style="list-style-type: none"><li>○ Data Abstraction Coupling: 5</li><li>○ Depth Of Inheritance Tree: 1</li><li>○ Lack Of Cohesion Of Methods: 1</li><li>○ Message Passing Coupling: 127</li><li>○ Non-Commenting Source Statements: 116</li><li>○ Number Of Added Methods: 9</li><li>○ Number Of Attributes: 8</li><li>○ Number Of Attributes And Methods: 31</li><li>○ Number Of Children: 0</li><li>○ Number Of Methods: 10</li><li>○ Number Of Operations: 23</li><li>○ Number Of Overridden Methods: 0</li><li>○ Response For A Class: 43</li><li>○ Weighted Methods Per Class: 42</li></ul>	<ul style="list-style-type: none"><li>○ Data Abstraction Coupling: 5</li><li>○ Depth Of Inheritance Tree: 1</li><li>○ Lack Of Cohesion Of Methods: 1</li><li>○ Message Passing Coupling: 38</li><li>○ Non-Commenting Source Statements: 38</li><li>○ Number Of Added Methods: 3</li><li>○ Number Of Attributes: 5</li><li>○ Number Of Attributes And Methods: 22</li><li>○ Number Of Children: 0</li><li>○ Number Of Methods: 4</li><li>○ Number Of Operations: 17</li><li>○ Number Of Overridden Methods: 0</li><li>○ Response For A Class: 26</li><li>○ Weighted Methods Per Class: 13</li></ul>	<ul style="list-style-type: none"><li>○ Data Abstraction Coupling: 5</li><li>○ Depth Of Inheritance Tree: 1</li><li>○ Lack Of Cohesion Of Methods: 1</li><li>○ Message Passing Coupling: 89</li><li>○ Non-Commenting Source Statements: 82</li><li>○ Number Of Added Methods: 6</li><li>○ Number Of Attributes: 8</li><li>○ Number Of Attributes And Methods: 28</li><li>○ Number Of Children: 0</li><li>○ Number Of Methods: 7</li><li>○ Number Of Operations: 20</li><li>○ Number Of Overridden Methods: 0</li><li>○ Response For A Class: 33</li><li>○ Weighted Methods Per Class: 30</li></ul>
---	--	--

As pointed out earlier, the **HoaModifyDTO** class had both getters and setters, while having all its attributes set to public. We have changed all of them to private, and since many other methods

accessed these attributes via public access instead of getters and setters, we had to refactor many other methods in other classes to be using the getters and setters properly. After that, the NOPA metric for the HoaModifyDTO class decreased to 0 which is within the regular limits.



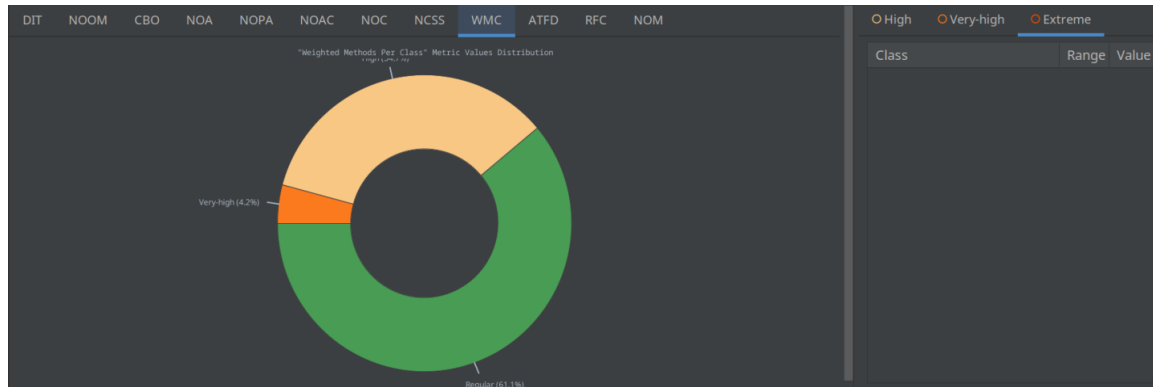
Similarly, to reduce the WMC metric of the **Proposal** class, some of its methods have been moved to a helper class, converting them to static functions. This allows the class to be more readable without hampering the functionality.

Proposal	->	Proposal (updated)	ProposalHelper
<div><div>M</div>○ Number Of Attributes: 10</div> <div><div>M</div>○ Number Of Attributes And Methods: 44</div> <div><div>M</div>○ Number Of Children: 1</div> <div><div>M</div>○ Number Of Methods: 22</div> <div><div>M</div>○ Number Of Operations: 34</div> <div><div>M</div>○ Number Of Overridden Methods: 0</div> <div><div>M</div>○ Response For A Class: 43</div> <div><div>M</div>○ Weighted Methods Per Class: 35</div>		<div><div>M</div>○ Number Of Attributes: 10</div> <div><div>M</div>○ Number Of Attributes And Methods: 45</div> <div><div>M</div>○ Number Of Children: 1</div> <div><div>M</div>○ Number Of Methods: 23</div> <div><div>M</div>○ Number Of Operations: 35</div> <div><div>M</div>○ Number Of Overridden Methods: 0</div> <div><div>M</div>○ Response For A Class: 37</div> <div><div>M</div>○ Weighted Methods Per Class: 30</div>	<div><div>M</div>○ Number Of Attributes: 0</div> <div><div>M</div>○ Number Of Attributes And Methods: 14</div> <div><div>M</div>○ Number Of Children: 0</div> <div><div>M</div>○ Number Of Methods: 2</div> <div><div>M</div>○ Number Of Operations: 14</div> <div><div>M</div>○ Number Of Overridden Methods: 0</div> <div><div>M</div>○ Response For A Class: 19</div> <div><div>M</div>○ Weighted Methods Per Class: 8</div>

For the fifth class to refactor, we have not found anything else going over our thresholds. Instead, as per the assignment description, we would like to point to refactoring we did in previous weeks, specifically **commits 4fa4f52e972bb811a3ef079d67789c039faf857 and 917d06f5cf9dd93bfb543ebcf4e6ed19eca03dc9**, also tagged on git as ‘refactoring’. In these commits, we spread out methods from AuthenticationController into UserInfoControllers, as the class was getting too long, and some new endpoints were created which were originally in AuthenticationController were also put into UserInfoController instead.

*As we’ve been recently informed, lombok annotations should not be regarded as code smells, therefore we have replaced the below three methods with other refactoring above. We decided to leave this in the document nonetheless, but we do not count it towards the 5 classes refactored.*

Refactoring for the three classes: **Membership**, **ProposalHistoryResponseModel** **ProposalInformationResponseModel** as per the **WMC** metric, results:



Upper row - before, Lower row - after

**Membership** **ProposalHistoryResponseModel** **ProposalInformationResponseModel**

<ul style="list-style-type: none"> <li>○ Data Abstraction Coupling: 7</li> <li>○ Depth Of Inheritance Tree: 1</li> <li>○ Lack Of Cohesion Of Methods: 5</li> <li>○ Message Passing Coupling: 26</li> <li>○ Non-Commenting Source Statements: 51</li> <li>○ Number Of Added Methods: 21</li> <li>○ Number Of Attributes: 10</li> <li>○ Number Of Attributes And Methods: 45</li> <li>○ Number Of Children: 1</li> <li>○ Number Of Methods: 22</li> <li>○ Number Of Operations: 35</li> <li>○ Number Of Overridden Methods: 0</li> <li>○ Response For A Class: 43</li> <li>○ Weighted Methods Per Class: 35</li> </ul>	<ul style="list-style-type: none"> <li>○ Data Abstraction Coupling: 5</li> <li>○ Depth Of Inheritance Tree: 1</li> <li>○ Lack Of Cohesion Of Methods: 9</li> <li>○ Message Passing Coupling: 12</li> <li>○ Non-Commenting Source Statements: 12</li> <li>○ Number Of Added Methods: 17</li> <li>○ Number Of Attributes: 8</li> <li>○ Number Of Attributes And Methods: 42</li> <li>○ Number Of Children: 0</li> <li>○ Number Of Methods: 21</li> <li>○ Number Of Operations: 34</li> <li>○ Number Of Overridden Methods: 3</li> <li>○ Response For A Class: 33</li> <li>○ Weighted Methods Per Class: 45</li> </ul>	<ul style="list-style-type: none"> <li>○ Data Abstraction Coupling: 4</li> <li>○ Depth Of Inheritance Tree: 1</li> <li>○ Lack Of Cohesion Of Methods: 8</li> <li>○ Message Passing Coupling: 13</li> <li>○ Non-Commenting Source Statements: 13</li> <li>○ Number Of Added Methods: 18</li> <li>○ Number Of Attributes: 8</li> <li>○ Number Of Attributes And Methods: 43</li> <li>○ Number Of Children: 0</li> <li>○ Number Of Methods: 22</li> <li>○ Number Of Operations: 35</li> <li>○ Number Of Overridden Methods: 3</li> <li>○ Response For A Class: 34</li> <li>○ Weighted Methods Per Class: 46</li> </ul>
<ul style="list-style-type: none"> <li>○ Data Abstraction Coupling: 2</li> <li>○ Depth Of Inheritance Tree: 1</li> <li>○ Lack Of Cohesion Of Methods: 8</li> <li>○ Message Passing Coupling: 4</li> <li>○ Non-Commenting Source Statements: 9</li> <li>○ Number Of Added Methods: 10</li> <li>○ Number Of Attributes: 10</li> <li>○ Number Of Attributes And Methods: 37</li> <li>○ Number Of Children: 0</li> <li>○ Number Of Methods: 14</li> <li>○ Number Of Operations: 27</li> <li>○ Number Of Overridden Methods: 2</li> <li>○ Response For A Class: 17</li> <li>○ Weighted Methods Per Class: 18</li> </ul>	<ul style="list-style-type: none"> <li>○ Data Abstraction Coupling: 5</li> <li>○ Depth Of Inheritance Tree: 1</li> <li>○ Lack Of Cohesion Of Methods: 8</li> <li>○ Message Passing Coupling: 19</li> <li>○ Non-Commenting Source Statements: 21</li> <li>○ Number Of Added Methods: 16</li> <li>○ Number Of Attributes: 8</li> <li>○ Number Of Attributes And Methods: 40</li> <li>○ Number Of Children: 0</li> <li>○ Number Of Methods: 19</li> <li>○ Number Of Operations: 32</li> <li>○ Number Of Overridden Methods: 2</li> <li>○ Response For A Class: 36</li> <li>○ Weighted Methods Per Class: 30</li> </ul>	<ul style="list-style-type: none"> <li>○ Data Abstraction Coupling: 4</li> <li>○ Depth Of Inheritance Tree: 1</li> <li>○ Lack Of Cohesion Of Methods: 7</li> <li>○ Message Passing Coupling: 21</li> <li>○ Non-Commenting Source Statements: 22</li> <li>○ Number Of Added Methods: 17</li> <li>○ Number Of Attributes: 8</li> <li>○ Number Of Attributes And Methods: 41</li> <li>○ Number Of Children: 0</li> <li>○ Number Of Methods: 20</li> <li>○ Number Of Operations: 33</li> <li>○ Number Of Overridden Methods: 2</li> <li>○ Response For A Class: 35</li> <li>○ Weighted Methods Per Class: 31</li> </ul>

The **@Data** annotation increased the cyclomatic complexity by a large amount and added a lot of extra methods for classes with a large amount of attributes. We have removed this annotation and replaced it with more specific annotations/implementations for methods used, thereby improving on the weighted methods per class metric, with minor improvements in a few other metrics as well for these three classes.

## Method level metrics:

For the method `addActivity()` i moved all the logic that checks if the given variables are correct in size and by the specifications to a separate method that used to only check the name, that method was then changed from `checkNameValidity()` to `checkParamValidity()` and resulted in `addActivity()` having a lower Cyclomatic complexity and lines of code without impacting the helper method too much

addActivity(int, String, Date, String, String)	addActivity(int, String, Date, String, String)
Condition Nesting Depth: 1	Condition Nesting Depth: 1
Halstead Difficulty: 18.6875	Halstead Difficulty: 15.4062
Halstead Effort: 8719.3158	Halstead Effort: 4429.7662
Halstead Errors: 0.1412	Halstead Errors: 0.0899
Halstead Length: 84	Halstead Length: 57
Halstead Vocabulary: 47	Halstead Vocabulary: 33
Halstead Volume: 466.5855	Halstead Volume: 287.5305
Lines Of Code: 35	Lines Of Code: 25
Loop Nesting Depth: 0	Loop Nesting Depth: 0
Maintainability Index: 47.354	Maintainability Index: 52.1088
McCabe Cyclomatic Complexity: 8	McCabe Cyclomatic Complexity: 4
Number Of Loops: 0	Number Of Loops: 0
Number Of Parameters: 5	Number Of Parameters: 5

For the method `registerUser()` i once again moved the logic for checking the parameters to a separate method which resulted in a sizable difference in Cyclomatic complexity and Lines of code once again, both of which were bad smells over the limit

registerUser(Username, Password, FullName)	registerUser(Username, Password, FullName)
Condition Nesting Depth: 2	Condition Nesting Depth: 1
Halstead Difficulty: 21.0	Halstead Difficulty: 12.8571
Halstead Effort: 8792.379	Halstead Effort: 1529.2539
Halstead Errors: 0.142	Halstead Errors: 0.0442
Halstead Length: 83	Halstead Length: 28
Halstead Vocabulary: 33	Halstead Vocabulary: 19
Halstead Volume: 418.6847	Halstead Volume: 118.942
Lines Of Code: 35	Lines Of Code: 20
Loop Nesting Depth: 0	Loop Nesting Depth: 0
Maintainability Index: 47.6417	Maintainability Index: 57.0188
McCabe Cyclomatic Complexity: 11	McCabe Cyclomatic Complexity: 2
Number Of Loops: 0	Number Of Loops: 0
Number Of Parameters: 3	Number Of Parameters: 3



For the method `beginVoting()` for refactoring i removed some unnecessary checks, added the complex logic to the try/catch block as well as created constructors for a parameter which was set using setters which greatly reduced the Lines of code of the method

beginVoting(ProposalGenericRequestModel)	beginVoting(ProposalGenericRequestModel)
(M) Condition Nesting Depth: 1	(M) Condition Nesting Depth: 1
(M) Halstead Difficulty: 34.1786	(M) Halstead Difficulty: 27.0
(M) Halstead Effort: 15022.4202	(M) Halstead Effort: 9981.1194
(M) Halstead Errors: 0.2029	(M) Halstead Errors: 0.1545
(M) Halstead Length: 81	(M) Halstead Length: 69
(M) Halstead Vocabulary: 43	(M) Halstead Vocabulary: 41
(M) Halstead Volume: 439.5274	(M) Halstead Volume: 369.6711
(M) Lines Of Code: 38	(M) Lines Of Code: 31
(M) Loop Nesting Depth: 0	(M) Loop Nesting Depth: 0
(M) Maintainability Index: 46.7951	(M) Maintainability Index: 49.2767
(M) McCabe Cyclomatic Complexity: 6	(M) McCabe Cyclomatic Complexity: 5
(M) Number Of Loops: 0	(M) Number Of Loops: 0
(M) Number Of Parameters: 1	(M) Number Of Parameters: 1

For the method `addOption()` for refactoring i once again removed unnecessary checks in favor of the try/catch block as well as moving the main logic inside the block as well

addOption(AddOptionRequestModel)	addOption(AddOptionRequestModel)
(M) Condition Nesting Depth: 1	(M) Condition Nesting Depth: 0
(M) Halstead Difficulty: 37.0556	(M) Halstead Difficulty: 32.0625
(M) Halstead Effort: 12056.8219	(M) Halstead Effort: 7893.9666
(M) Halstead Errors: 0.1753	(M) Halstead Errors: 0.1322
(M) Halstead Length: 62	(M) Halstead Length: 48
(M) Halstead Vocabulary: 38	(M) Halstead Vocabulary: 35
(M) Halstead Volume: 325.3715	(M) Halstead Volume: 246.2056
(M) Lines Of Code: 33	(M) Lines Of Code: 27
(M) Loop Nesting Depth: 0	(M) Loop Nesting Depth: 0
(M) Maintainability Index: 49.1005	(M) Maintainability Index: 51.9417
(M) McCabe Cyclomatic Complexity: 4	(M) McCabe Cyclomatic Complexity: 2
(M) Number Of Loops: 0	(M) Number Of Loops: 0
(M) Number Of Parameters: 1	(M) Number Of Parameters: 1

For the method `castVote()` I once again removed some unnecessary changes as well added the main logic to a try/catch block which decreased Lines of Code and Cyclomatic complexity once again

castVote(CastVoteRequestModel)	castVote(CastVoteRequestModel)
Condition Nesting Depth: 1	Condition Nesting Depth: 1
Halstead Difficulty: 28.0	Halstead Difficulty: 21.0
Halstead Effort: 9698.7793	Halstead Effort: 5614.4107
Halstead Errors: 0.1516	Halstead Errors: 0.1053
Halstead Length: 67	Halstead Length: 53
Halstead Vocabulary: 36	Halstead Vocabulary: 33
Halstead Volume: 346.385	Halstead Volume: 267.3529
Lines Of Code: 35	Lines Of Code: 29
Loop Nesting Depth: 0	Loop Nesting Depth: 0
Maintainability Index: 48.2981	Maintainability Index: 50.9224
McCabe Cyclomatic Complexity: 6	McCabe Cyclomatic Complexity: 4
Number Of Loops: 0	Number Of Loops: 0
Number Of Parameters: 1	Number Of Parameters: 1