✎ Are performance scores good indicators for energy efficiency of web apps

Miguel Morales Exposito
2618012
m.e.miguel@student.vu.nl

Tanjina Islam
2609513
tanjina012@gmail.com

Christian Valladares
2612946
cvalladares4837@gmail.com

Sanjay Sheombar
2589918
s.sheombar@student.vu.nl

Kwame Chan-Jong-Chu
2583387 Number
kwamsc91@gmail.com

Abstract—Context. **Developers have access to tools like Lighthouse that help them assess the performance of web apps and to guide the adoption of development best practices. When it comes to energy consumption however, these tools seem to be lacking.** *Goal*. **This research investigates if there is correlation between the performance scores from Lighthouse and the energy consumption of web apps. By proving correlation, the performance scores could help developers guide their development in less energy consuming web apps.**
Method. **21 web apps were measured with the Trepn energy profiler and grouped on the performance treatments: Good, Average and Poor. To assess if the energy consumption is related to the fixed performance treatment, we performed the Kruskall Wallis test. We also performed the Spearman's rank correlation test to identify the correlation between the different performance treatments and energy consumption. Furthermore, we performed Dunn's test to identify the pairwise difference among performance treatments. To quantify these differences, we performed Cliff's Delta between the different pair groups to determine the effect size.**
Results. **The Kruskall Wallis test resulted in a significant difference between the performance score cut-offs with a p-value of 3.382e-14. We identified a correlation of -0.324 between performance scores and energy consumption; implying that an increase in performance brings about a decrease in energy consumption. Performing Dunn's test, we discovered that there is a significant difference between Good-Poor and Average-Poor, with an effect size of respectively $d_{Good-Poor} = -0.444$ and $d_{average-poor} = -0.386$. However, Dunn's test did not show a significant difference between good and average performing web-apps with a p-value of 0.499, and this is also corroborated by a negligible delta value of -0.099.**
Conclusions. **We concluded that poor performant web apps consume more energy than web apps with good or average performance levels. However, the performance score alone would not be sufficient in motivating the software developers to build web apps that consume less energy. Performance scores could guide developers in developing less energy consuming web apps, until the treshold for average is reached.**

Index Terms—**Empirical Software Engineering, Green Software, Mobile, Lighthouse, Energy Consumption,**

I. INTRODUCTION

As mobile users increasingly grow to the largest portion of internet users [1], it is important to keep in mind that these consumers of web based applications have different constraints then their personal computer counterparts. One important constraint is the limited battery life of a mobile device [2].

It has been shown that poor energy usage of mobile application contributes to user's negative evaluation, and may even lead to abandonment [3]. Considering the large market share of mobile users, this can affect revenues generated by businesses relying on these applications. This insight can be extended to web applications(web apps) which attempt to produce a native app-like experience in order to fulfill user's growing demand of functionality on web pages, while meeting a developer's requirement of portability across platforms.

Despite of the constraints presented by mobile devices, developers continue to push web technologies to deliver higher functionality while managing performance [4]. Web apps that have a perceived poor performance affect profits and can lead to user abandonment. The BBC lost an additional 10% of users for every additional second their site took to load [5]. Improving the perceived performance is crucial for increasing conversion. Pinterest rebuilt their pages for the sake of performance, realizing a 40% reduction in perceived wait times, which increased both search engine traffic and sign-ups by 15% [6].

To this end, developers have access to tools that help them assess their web app's performance and to guide the adoption of development best practices. Unlike quality requirements such as performance, 'best practices' and development guidelines provided by higher level measurement tools are lacking when it comes to energy efficiency. According to researchers, Application layer research on energy consumption can empower developers to participate in energy conscious development; but, in the context of their research, they identified a knowledge gap and limited tooling that prevented proper assessment of their application [7]. Without these resources, the energy mismanagement that could be introduced in the development cycle may be overlooked.

In contrast, there exist many performance bench-marking tools which help to close the knowledge gap and make performance improvements more accessible. Lighthouse is one of many bench-marking tools that attempt to make performance auditing more accessible to developers [8].

As a tool, Lighthouse, makes assessments according to the

RAIL performance model, which establishes goals based on human perception [9]. Human perception in the RAIL model relates to how an application handles four key actions: the response time to a user's input, the rendering performance for animations, optimal idle-time utilization for the sake of responsiveness, and load impact on a web app. Based on the described model, Fig 1 shows the metrics Lighthouse uses to score a web app:

First Contentful Paint marks the point, when the browser renders the first bit of content and *First Meaningful Paint* marks the point when the page shows its primary content. These metrics provide feedback for users that the page is actually loading. *Time to interactive* measures how long it takes a page to display "useful" content, and the page responds to user interaction within 50 milliseconds.

These metrics are weighed together and ranked in a log-normal distribution of web apps in order to give a score from 0-100.
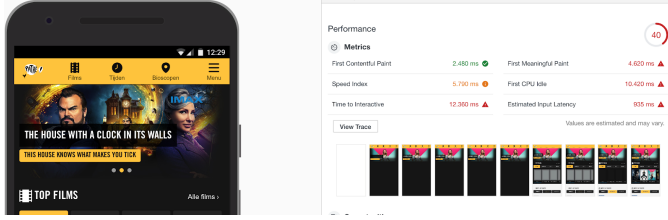

Fig. 1: Pathe.nl Lighthouse performance audit

Energy consumption is overlooked in this assessment. While Web optimizations or any changes to the Web ecosystem are often studied in terms of performance, their effect on energy is not easy to measure.

The purpose of this research is to analyze if there is a correlation between energy consumption of a web application and its measured performance, and if so, to what extent. A correlation between a high measured performance by these tools and higher energy consumption implies that tools such as these can help guide not only the performance, but also help developers take responsibility for the energy consumption of their web applications.

On the other hand, if there is no correlation, developers can employ this knowledge and use other tools to measure the energy consumption of their application, or even expand Lighthouse to also include an energy consumption meter.

II. EXPERIMENT DEFINITION

A. Goal

The goal of the experiment is identified in Table I, using the well-established GQM goal definition pattern [10]:

B. Questions

[RQ1]: *To what extent do performance scores correlate to the energy consumption in the context of web apps?*

Analyze	Performance Score
For the purpose of	Evaluating correlation
With respect to their	Energy Consumption
From the point of view of	Software Developers
In the context of	Web Apps
Result	
Analyze performance score for the purpose of evaluating how performance correlates with respect to their energy consumption from the point of view of software developers in the context of Web Apps	

TABLE I: Goal definition

To answer this question we will randomly choose 21 web apps from the Alexa list of top visited web apps and load them on the device to measure the energy consumption of each[11]. Then we will compare the energy consumption with the score obtained from Lighthouse and check whether there is any correlation between these two values or not.

C. Metrics

To answer the aforementioned questions and thus fulfill the objective of our experiment we will need the following metrics:

– Power consumption: measured in micro-watts. It is the base metric that we will use to calculate energy consumption of different web apps.
– Profiling time: The duration of each run of the experiment, based on the Time to interactive lightouse-metric for each web app in milliseconds
– Total energy consumption: It is measured in Joule based on power consumption over time to load the web application. This value is calculated using the following formula: *Energy = Power x Time*

D. GQM-Tree

Figure 2 displays the visual representation of the GQM tree. This diagram illustrates how our experimental goal, the questions related to this goal and the metrics that are going to address those questions are derived from hierarchical model.
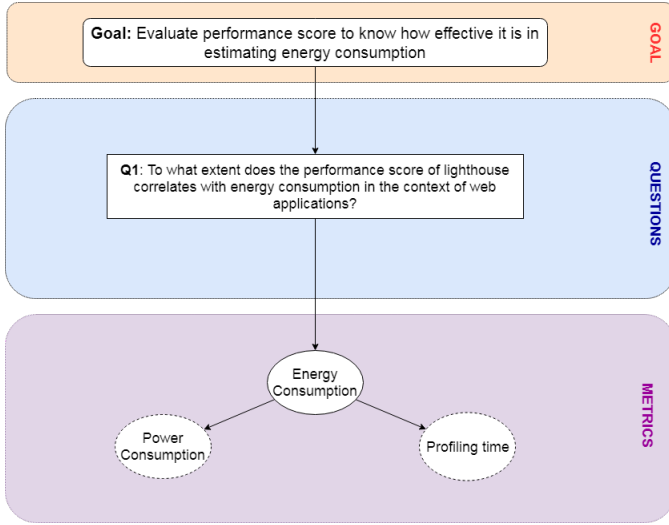
Fig. 2: GQM tree of the Experiment

III. EXPERIMENT PLANNING

A. Context Selection

In this particular section, we cover our context that is the environment under which we are going to perform our experiments. The context has been defined based on four dimensions proposed by Wohlin et al. [10]

The first dimension is Online versus Offline experimentation. The context of our experiment is going to be limited to mobile web apps running on the Google Chrome browser on Android device. This is an offline experiment based on the fact that the measurements are done via tools on web apps that have already been developed and published on the web, and we are not part of the development process.

The second dimension is concerned with students or professionals as subjects. Given that our concern is the energy consumption as a function of a web app's performance score, the subjects of the experiment are developers, who ultimately have a direct impact on both the score and an app's energy efficiency.

The third dimension is the nature of the experiment: toy problems vs real problems. We sampled actual web apps from the Alexa list from different performance categories, thus targeting web apps from a real world context. Furthermore, we will be conducting experiments to address a real world problem - how performance score could be an indication of energy consumption for web apps. If proven so, performance audits might help guide software developers towards developing less energy consuming web apps.

The fourth dimension is whether our experiment is specific versus general. We are specific because we are conducting our experiment using Lighthouse as performance analyzing tool and Chrome browser on Android platform. The web apps used for the experiment are subjects from the top visited web apps with high traffic. Although we are not targeting other supported tools, browsers and platforms, we are using tools made by high regarded developers and have a good representative sample of the population. More specifically, Chrome captures over 60% market share [12]. Lighthouse is developed by Google, under active development to improve its audits. Finally, Trepn was developed by Qualcomm which captures 42 percent of processors market share [13].

B. Variable Selection

For our research question, we consider a web app's performance score as the independent variable. We control and change our independent variables via the selection of web apps on the basis of their scores. The measurement scale for this variable is a composite score of different loading time milestones, measured in milliseconds, which are weighted, and these are compared to a benchmark of real web apps. The outcome of this process is a relative performance score placed in a log normal distribution of benchmark scores. The scoring range is thus between a score of one and one hundred. We concern ourselves in measuring web apps with the following fixed levels based on lighthouse score ranges [14]

App name	URL	Lighthouse performance score	average energy consumption	L.o.c. HTML	L.o.c. CSS	L.o.c. JavaScript	Time to interactive
awsamazoncom	http://www.amazonaws.com	0.13	3078.0682	11642	21519	42225	
apple	http://www.apple.com	0.45	175.86779	992	34967	45516	
ask	http://www.ask.com	0.94	39.6393	729	557	11977	
china*	http://www.china.com	0.08	206.62427	2131	3617	19791	
cnn	http://www.cnn.com	0.01	2865.62333	44803	866	153037	
coccoc	http://www.coccoc.com	0.22	537.66665	1121	16149	154126	
ettoday	http://www.ettoday.net	0.04	477.83588	1599	15989	111295	
hao123	http://www.hao123.com	0.17	323.1916	14582	56	25905	
instagram	http://www.instagram.com	0.69	141.85656	12078	0	83393	
microsoft	http://www.microsoft.com	0.86	55.14962	3275	94	5548	
paypal	http://www.paypal.com	0.63	91.65997	991	8995	20302	
popads	http://www.popads.net	0.94	0.65731	434	553	7253	
quora	http://www.quora.com	0.59	197.19694	4377	51492	25800	
theguardian	http://www.theguardian.com	0.43	780.81904	1337	10785	48699	
tianya	http://www.tianya.cn	0.52	61.03795	347	1409	7413	
twitter	http://www.twitter.com	0.48	191.28732	1616	30451	54336	
whatsapp	http://www.whatsapp.com	0.63	224.84987	20532	44333	129280	
xnxx	http://www.xnxx.com	0.8	124.76117	4350	10635	11023	
xvideos	http://www.xvideos.com	0.76	223.455	5381	25369	23090	
yandex	http://www.yandex.ru	0.83	94.20127	20125	5054	62952	
youtube	http://www.youtube.com	0.75	170.90367	25540	12318	163102	

TABLE II: Balanced design

- Poor: (0 - 44)
- Average: (45 - 74)
- Good: (75 - 100)

We consider the energy consumption as the dependent variable. We measure the consumed energy of a web app while loading it. Energy consumption is a variable calculated as the power consumed by Google Chrome to load one subject multiplied by the time the profiler has been tracking it.

C. Hypothesis Formulation

The following hypothesis were formulated to address the main research question. To assess if the outcome of the energy consumption is related to the fixed performance levels: Poor, Average, Good - we have the following hypothesis:

We test the null hypothesis that the energy consumption mean of web apps in the category low, average and high are the same:

$$H_0^1 : \mu_{low} = \mu_{average} = \mu_{high}$$

versus the alternative that they are not the same for at least one pair:

$$H_a^1 : \exists (i, j) | \mu_i ! = \mu_j$$

for at least one pair (i,j). proving that they are not the same gives an idea of dependency.

D. Subject Selection

The subject selection is done by a python script [15] which reads a csv containing the top 1 million web apps from Alexa. The script starts from the first web apps (the most visited measured by Alexa) and goes down the list until it has 100 web apps. We discard domain names where the only difference is their extension. For example if the list contains http://www.google.com and then http://www.google.ru will not be added to the list. We use Lighthouse batch reporter to run Lighthouse tests on the list of 100 web apps, which generates 100 JSON files with Lighthouse scores [16]. We then place the web apps in categories based on their scores, these categories being poor 0-44, average 45-74, good 75-100. From each of these categories we randomly selected 7 web apps (yielding a total of 21 web apps).

E. Experiment Design

In order to determine if there is any statistically significant difference between the performance scores, the design will guide the statistical test needed. The statistical test is used to understand whether the energy consumption differed based on the performance score levels.

The process of selecting the right test involves considering the dependent, independent variable and proving the assumptions.

-Energy consumption of the web app is our dependent variable which is a continuous ratio level of measurement.

-Performance score is our only independent variable with 3 treatments (Good, Average and Poor).

We consider a balanced design of subjects over the different treatments in Table III:

Treatment level	Web Apps						
Good	w_1	w_2	w_3	w_4	w_5	w_6	w_7
Average	w_8	w_9	w_{10}	w_{11}	w_{12}	w_{13}	w_{14}
Poor	w_{15}	w_{16}	w_{17}	w_{18}	w_{19}	w_{20}	w_{21}

TABLE III: Balanced design

Every Treatment level contains 7 unique web apps denoted by w_i. Based on this design, we can consider the following tests which rely on the probability distribution of the data.

When the data meets the assumption of the outcome being sampled independently from normal populations, with possibly different population means, and with equal populations variances, we can consider one-way ANOVA.

When the data does not meet these assumptions, Kruskal Wallis can be considered.

F. Instrumentation

Hardware

For our experiment, we are using Nexus 9 tablet running on Android 5.1.1. The device is equipped with a 2.3 GHZ Dual Core processor (Denver), 2GB of RAM, a Kepler DX1 GPU and a 802.11 a/b/g/n/ac WiFi interface. Furthermore the device has a 6700 mAh Lithium-Polymer battery installed.

R and R studio

R Studio is an IDE for R. It will be used to perform the statistical tests [17].

Lighthouse

Lighthouse is used for performance measuring. This tool simulates an Android device and loads a web app on it measuring different audits. It gives a performance score from 0 to 100 based on the specified audits.

Trepn

To measure the energy consumption we use a power and performance profiling application called 'Trepn'. This profiler estimates the energy consumed on the mobile device. To get a better indication of energy consumption of the web app, we selected the delta power consumption value in μW which removes the drainage cause by the Android OS and the Trepn profiler itself. From Trepn we will also get the Profiling time in milliseconds. Then, Power consumption and Profiling time will be used to measure the total Energy consumption of each web apps. Web apps are run within the Google Chrome browser (version: 40.0.2214.89) with Javascript V8 3.30.33.15.

Android Runner

For the automation of the experiment, we are using the software tool Android Task runner [18]. This tool will load each web app on Google Chrome and start the power profiler. After 60 seconds, the browser is closed and Trepn stops profiling. For each web app the test is performed 25 times.

The energy consumption is measured for the initial load of the web app, no user interactions will be done on a web app.

Lighthouse Batch Reporter

This tool is used to perform a Lighthouse analysis on several web apps sequentially. We used to get the performance score of each of our selected subjects [16].

IV. EXPERIMENT EXECUTION

Our experiment starts with the setup. Android Runner is executed on a laptop with Ubuntu 18.04.1 LTS, Intel i7-4702MQ and 8GB RAM. With the laptop connected to the Android devices via USB, all the runs will be executed from the laptop.The runs of the experiment are done on the same wifi network with a speed of 100 Mbps. To ensure that the wifi conditions do not alter the experiment output, the mobile device is always placed 5 meters from the router.

We picked the first 100 web apps from the top 1 million Alexa list. We used Lighthouse-batch to get the JSON files with the report of Lighthouse. This returns JSON files which contain the scores of the performance test. We enforce the simulation of 3G connectivity and we also force the simulation of first-time loads (ignoring the cache). This simulation delivers a full report: an aggregated score for each performance category, and also the raw values for each individual load time milestone, in addition to the settings and parameters that were requested for a given test run. The aggregate performance score is the relevant independent variable, thus it is extracted from these reports.

From these 100 web apps, we randomly select a web app and fill it in a category: Good, Average, Poor, based on its score. This is done until each category reaches 7 web apps. If the randomly selected web app fits a category that already reached its limit of 7, it is discarded. The web apps are ready to use when a total of 21 subjects are selected

From the selected 21 web apps, we run the tests using Android runner which will execute each web app on Google Chrome. We make the tests on intervals of 2 minutes each until the Time to Interactive. We collect the energy readings using an automated script to execute 21 web apps for 25 trials each. On each run android runner will also start the Trepn profile tool and generate a csv file with the power consumed.

Given the design of the experiment, we must take statistical tests for one factor and more than two treatments. The measurements are independent from each other, and energy consumption is a continuous function. We must then analyze the distribution of these variables in order to check whether a parametric or non-parametric test will hold. We start with identifying outliers by using a box plot. Furthermore, we intend to use Q-Q Plots with respect to the normal distribution in conjunction with the Shapiro - Wilks test to verify the normality of the dependent variable. We will also verify the normality of the residuals. We will then verify the whether or not an equal variance exists between all sample groups. The parametric checks may not hold initially. In some cases, transforming the data may fit the assumptions better.
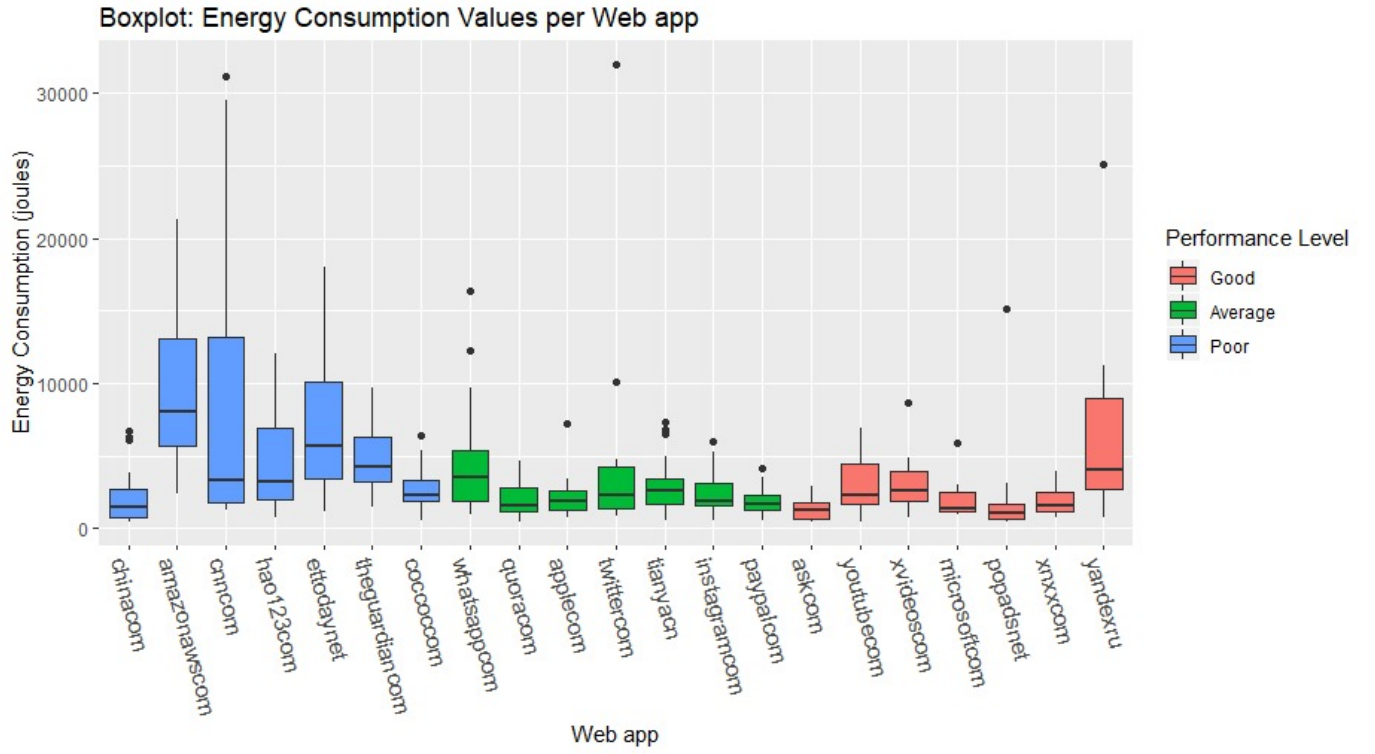
Fig. 3: Boxplots

Upon verifying these assumptions, we will perform either the one-way ANOVA test or Kruskal-Wallis on the collected data. This test will verify whether the means of the different performance score groups stem from unequal population variances. If indeed the population variances differ, we intend to perform Tukey's or Dunn's test to identify which scoring groups differ from each other. Furthermore, we would like to understand the effect size if indeed an effect exists. This will be done by applying Cohen's d.

V. RESULTS

A. descriptive statistics

The summary presented in Table IV and histogram in Fig 4 show a better understanding of the energy consumption data of web apps.

	Energy consumption
Min	384.1
1st Qu.	1438.7
Median	2324.4
Mean	3712.2
3rd Qu.	4221.0
Max	32054
St. deviation	4189.437

TABLE IV: Summary energy consumption.
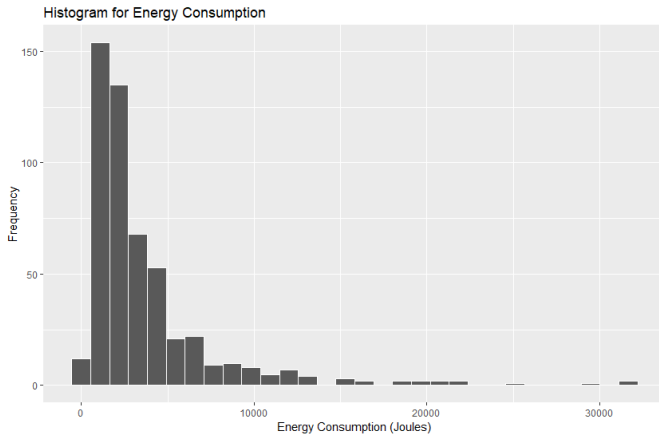
Fig. 4: Histogram energy consumption

Fig. 6: Scatter-plot performance vs energy consumption

From the diagrams, we could see that the energy consumption measurements for Good and Average are pretty similar; both of these groups are centered around the same energy consumption value, and their outliers are quite similar. Comparing both the good and average performance values to the Poor performance level, we see a bigger difference. The median energy consumption from poor performing web apps is higher when compared to good and average performance web apps.

The energy consumption is between 384.1 joules and 32054.2 joules. The central tendency based on the median lies around 2324.4 joules. Using the skewness formula [19], we are able to quantify the skewness of the data. We tested for skewness and achieved a result of 3.23. This value demonstrates a positive skewness as values that deviate from zero implies that the data is not symmetrical. This is further corroborated by the shape of fig 4, with the data showing a long tail towards the higher energy consumption values. A standard deviation of 4189.4 joules shows that there is a big spread in the data.

To get a clearer sense of the observations, we zoomed in on the energy consumption per level in Fig. 5 and Fig. 6. We created energy-consumption box plots diagrams and scatter-plots for the performance levels Good, Average and poor.
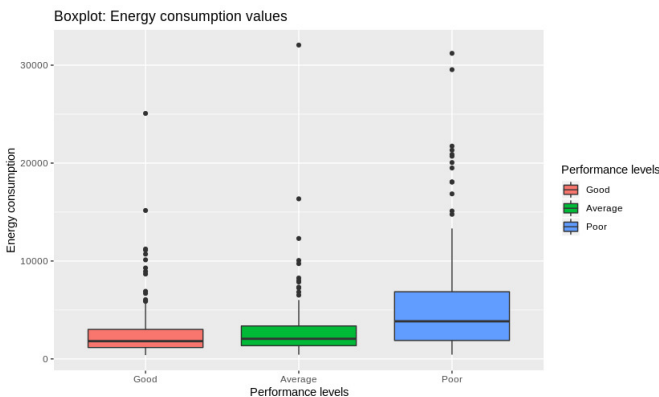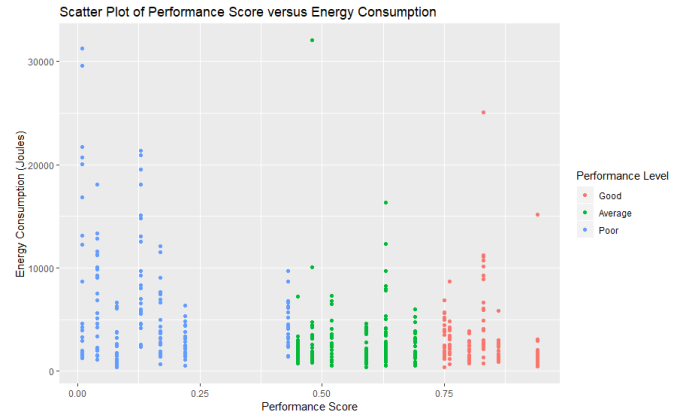
The histogram for energy consumption in Fig. 4 indicates that the energy consumption data is not normal. To justify our assumption we performed a Q-Q plot against a random sample of the normal distribution in Fig. 7.
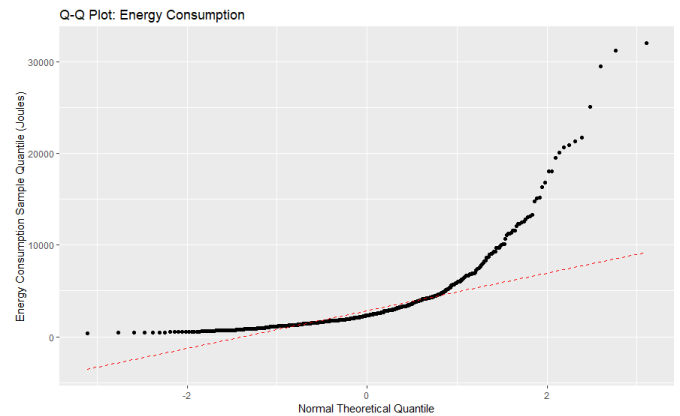

Fig. 7: Q-Q-plot energy consumption

Fig. 5: Box-plot energy consumption per treatment

The assumption of non normality for the energy consumption data could be caused by the rather small sample size of 21 web apps, or extreme values caused by the distortion of the data. We would prefer to use parametric statistics based on the higher statistical power of these tests. We performed various data transformations on the energy consumption data, including the squared (Fig. 8), reciprocal (Fig. 9), and the log transformation of the collected data (Fig. 10). The most promising transformation is the log operation on the energy consumption sample:

Fig. 8: Squared transformation

The square transformation of the energy consumption exacerbated the skewness of the data to a skewness value of 6.669, and this is clearly seen from both the histogram and the Q-Q plot. The Histogram does not show the characteristic bell-shape of normal data and the Q-Q plot demonstrates a more pronounced curvature.
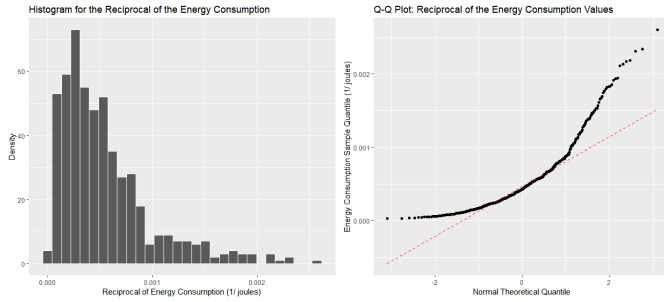

Fig. 9: Reciprocal transformation

The reciprocal transformation was able to better approximate our data to a normal distribution. Skewness decreased to a value of 1.646. However, the curvature of the Q-Q plot is still quite pronounced.
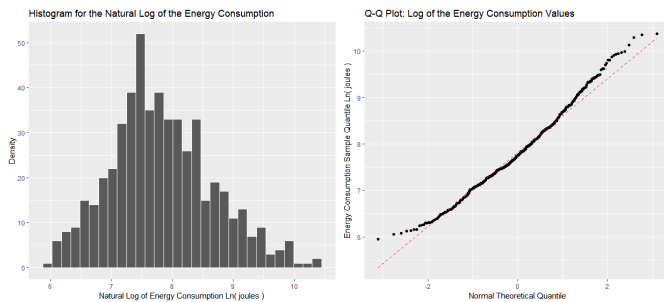

Fig. 10: Log transformation

We check the histogram of the log of the data, and we see that the skewness is dramatically improved. However the data is still positively skewed with an estimated value of 0.421, and the Quantile-Quantile plot shows a slight curvature. Furthermore, we performed the Shapiro - Wilks test, and received a p-value of 1.111e-4, thus there exists evidence to deny the null hypothesis that the transformed data lies within the normal distribution.

B. Hypothesis testing

As the assumption that the data comes from a normal distribution was not met, we performed the Kruskal Wallis non-parametric test based on our experiment design. The Kruskal Wallis rank based non-parametric test is used to determine if there are statistically significant differences between the energy consumptions at the different treatment levels: Good, Average, Poor. Employing the grouping technique from Lighthouse ensures a representative design, given that the performance scores themselves are defined in relation to the scores of other web apps.

In order to perform the test we made a numeric vector for our treatment levels by using a value of 3 for high performing web apps whereas a value of 1 for poor performance level. We got a p-value of 3.382e-14. Therefore, we can reject the null hypothesis that the means of energy consumption for each performance levels are equal.

Then we checked the correlation between the performance score of a web app and its energy consumption by using the spearman non-parametric correlation test. We obtained a negative correlation value of -0.324 for energy consumption with respect to performance, which infers that the energy consumption increases as the level of performance drops.

After that we performed a post-hoc pairwise comparison using Dunn tests due to the non-parametric nature of the data. Given that multiple pairwise comparisons are performed, the p-value must be corrected to avoid the higher probability of getting statistically significant results by chance. We used the Bonferroni correction technique for the pairwise comparison test. We got a p-value of 3.888e-13 for comparison between good and poor and a p-value of 5.119e-09 between average and poor performance levels. This results show that there is significant difference between good versus poor and average versus poor performance levels. On the other hand, we got a p-value of 0.499 between good and average levels, therefore we cannot assume a significant difference for these two treatments.

Next we applied Cliff's delta to investigate the effect size. Effect size will help us to know the quantified measure of the difference between two groups as it emphasizes more on the size of the difference rather than mixing it up with sample size. By looking at the delta estimate of -0.099 we found that the effect size is negligible when comparing good and average levels. For the comparison between good-poor and average-poor we get delta estimates of -0.444 and -0.386 respectively. These results show that the effect size for those treatment levels is medium.
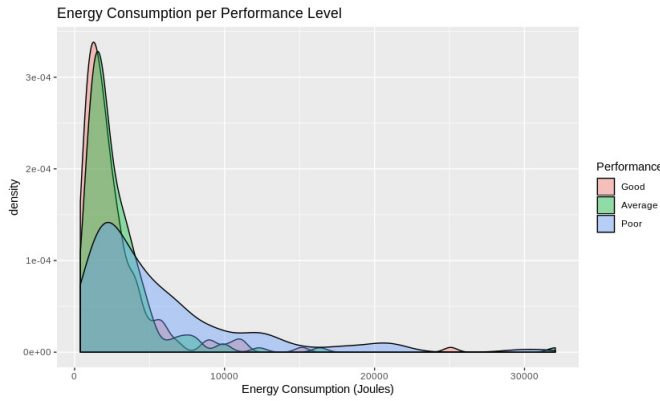
Fig. 11: Log transformation

In order to support the results given by Cliff's delta test we depicted in Fig. 11 a density graph with different levels. In this graph we can see how the density of good and average score are very similar and show a high frequency on low energy consumption values. Contrarily, we see smaller frequencies for low energy consumption for the web apps with poor performance score.

VI. DISCUSSION

From the results of the statistical tests performed in the previous section we can elaborate on our research question. The outcome of the Kruskal Wallis test shows that we can reject our null hypothesis. Therefore, we can say that there is a difference in the means of energy consumption of the different performance levels.

Given that Kruskal Wallis is an omnibus statistical test, the nature of these differences is best explained with the Dunn test. With the results from the Dunn test we can see that there is no significant differences of energy consumption between the good and average performance levels of web apps. However, the test shows that the difference in energy consumption of web apps with poor performance levels is quite relevant.

Finally we try to see the degree of overlap between two distributions of performance levels. With Cliff's Delta test we know that the effect size of a poor performance level against good and average levels is moderate as described by the Cliff's delta guidelines. However, comparing good and average performance levels yields negligible effect sizes.

With the help of data analysis and the obtained results from our experiment we can answer our *research question: To what extent does the performance score of lighthouse correlate with energy consumption in the context of web applications?*

We can conclude that performance score of lighthouse can be used as an indication of how much energy a certain web app consumes. More precisely, we can say that when a web app has a performance score between 0 and 44, the energy consumption is going to be higher than a web app with more than 44 as performance score.

This information can be useful for the web developers who are interested in reducing the energy consumption from their web applications. While developing web apps they can take into account the performance score from lighthouse's performance audit as a guide to estimate how much energy is going to be consumed by that particular web app. And based on this score, they can adopt changes or optimize the codes so that their web app consumes less energy.

VII. THREATS TO VALIDITY

The validity of the experiment has been analyzed based on the four types of classification as defined by Cook and Campbell [20].The four different types of threats to the validity of our experiment are described in the following sections:

A. Internal Validity

History
The way we performed the experiments to collect the energy consumption data was incremental. The first three data collection iterations collected energy consumption data on increments of 5 tests per day with two minutes between test. The last iteration was performed 4 days later with a repetition of 10 runs with two minutes between runs. This way we obtained the energy consumption of each web app 25 times. Furthermore, we performed the subject selection based on performance scores a few days before the energy consumption data was collected. Given the time gap between the subject selection and the data collection process, history may be an internal threat to validity because a web app's performance score could have changed all throughout the data collection process. This is also true for the Performance score for a web app. Given the time constraints, the motivation behind performing a more iterative data collection was to ensure that a balanced data set was produced; we opted for smaller increments so that every web app ended up with an equal number of runs per web app while reducing the risk of not collecting enough data on time. Furthermore, we tried to collect the energy consumption data on iterations as close to each other as possible to mitigate this threat.

Maturation
Maturation might play a role if a trial is repeated multiple times over same object. In order to reduce the effect of it, we made sure that Android runner takes intervals of 2 minutes with a duration of 60 seconds between each test execution. After each execution, it also clears up the cache from the web browser. Thereby, even though the trials are applied in different time-frames we made sure that there was not any cached data to influence the second run.

Reliability of measures
There are several factor that can affect the reliability of the measures i.e. brightness of the devices, distance to the router

and interference with other processes consuming energy. To mitigate this, we always set the brightness of the screen to the minimum and always set the device at the same distance from the router. To ensure that the energy measured is only consumed by the web app under test, we used the delta values for power consumption of Trepn which does not consider the energy consumed by the profiler and the OS.

B. External Validity

Interaction of selection and treatment

Since convenience sampling has been chosen as the sampling technique, there is a high chance of introducing biases and therefore it will make the outcome difficult to be generalized. This threat deals with the situation when the population of subjects is not representative of the one for which we would like to generalize our results. The population of the subjects (web apps) was chosen from the Alexa top 1 million web apps in terms of high traffic. We randomly selected 21 out of the 100 popular most visited web apps as our representative sample. We sampled the top 100 subset due to their high traffic–thus we select web apps of interest to a general population. Furthermore, the randomization of the selection process allows us to not introduce biases based on the type of web apps selected. Although the specific selection of samples affects the external validity and thus the results obtained from it may not be generalized but however, specific sampling helps to represent the scope of our defined experiment. Which in turn, results in performing the analysis of outcomes much more easier and thus increase the conclusion validity.

Interaction of setting and treatment

This threat deals with the situation when the experiments are performed in such a testing environment which is not realistic. For our experiment we used a relatively new device connected via wifi which representative of the way most users browse the internet. Google Chrome was the mobile browser of choice which captures 60% of market share and Lighthouse which is supported by Google. By using these tools and materials we made sure to generate a realistic experimental setting environment that is representative of a real world problem.

C. Construct Validity

Definition of constructs

In order to mitigate inadequate pre-operational explanation of constructs, we defined our constructs quite early before even performing the experiments using the standard GQM method. The goal of our experiment as well as the questions related to this goal and the metrics that are relevant to address those questions have been derived from this well-established GQM- tree. Using the GQM approach, we also formulated the hypotheses to address the main research question and identified the independent and dependant variables for our experiment.

Mono-operation bias

Our experiment is based on one factor, performance score which is the independent variable of our experiment design. Since we have only one independent variable, therefore the experiment might face mono-operation bias. But, we performed our experiments using 25 trials on single factor with 3 treatments which will help to mitigate the mono-operation bias from our measurements.

D. Conclusion Validity

Low statistical power

In order to deal with this threat and reduce the impact size of it, we made sure to have enough data to perform the data analysis. We used a fixed number of treatments i.e. 3 for our experiments with 7 subjects each. So in total 21 web apps and we executed the test 25 times per web app. Therefore, we have relatively large sample size with 525 trials in total of the web apps for all three treatment levels. This is done to make sure that we have sufficient data to perform statistical analysis on that. However, for future research, it may be helpful to increase the number of web apps per category to verify if statistically significant differences may be identified between good and average performing web apps.

Violated assumptions of statistical tests

To mitigate this threat to validity, we checked the distribution of data before performing the statistical analysis on that. This is done to make sure that we will select appropriate tests basing on the data distribution type. So that, based on the distribution of data, we can adjust our tests accordingly. While formulating our experiment plan, we decided to use the ANOVA test assuming the energy consumption is normally distributed. In our case this was not true hence we adjusted our tests and migrated to Kruskal-Wallis test.

Fishing and error rate

Since convenience sampling has been chosen as the sampling technique, there is a high chance of introducing biases and therefore it could influence the outcomes of our experiment. However, the specific selection of samples helps to define the scope of the experiments and thus will help to answer our research questions. But apart from that, with regards to mitigate this threat to validity we applied Bonferroni's p-value correction technique to obtain statistical correctness so that we can adapt the significance difference that we found while performing different statistical tests.

VIII. Conclusions

In this article we study the relation between performance score given by lighthouse and the energy consumption of a web app. We designed an experiment with performance score as a factor and three different treatment levels according to the performance category. After performing the experiment we can conclude that web apps with a poor score consume more energy than web apps with good and average scores. The performance score would be a good guiding factor for the

first few optimizations of the web app's energy consumption if the web app's performance is poor. Once the web app's performance crosses this threshold to the average performance level, more specialized methods should be used to guide optimizations for energy consumption while developing web apps.

This experiment could be extended performing an analysis on the score of each of the audits used to give a performance score. This way it would be possible to know which audit affects more to the energy consumption and developers could focus on its improvement to reduce energy consumption.

Another possible extension to the study could be done by using different bench-marking tools. In this way, it would be possible to know which tool gives a more precise score to estimate energy consumption.

REFERENCES

[1] Statscounter, "Desktop vs mobile vs tablet market share worldwide," GlobalStats, Tech. Rep., August 2018. [Online]. Available: http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet

[2] J. L. . M. Q. . J.-W. N. . T. Chen, "Battery-aware task scheduling in distributed mobile systems with lifetime constraint," IEEE, Tech. Rep., 2011. [Online]. Available: https://ieeexplore.ieee.org/document/5722286/

[3] L. Cruz and R. Abreu, "Performance-based guidelines for energy efficient mobile applications," IEEE/ACM, Tech. Rep., 2017. [Online]. Available: https://luiscruz.github.io/papers/cruz2017performance.pdf

[4] H. archive, "State of the web," Http archive, Tech. Rep., 2018. [Online]. Available: https://beta.httparchive.org/reports/state-of-the-web#bytesTotal

[5] M. Clark. (2018, January) How the bbc builds websites that scale. [Online]. Available: https://www.creativebloq.com/features/how-the-bbc-builds-websites-that-scale

[18] S2-Group, "Android task runner," GitHub Repository, 2018. [Online]. Available: https://github.com/S2-group/android-runner

[6] V. A. . J. C. Sam Meder. (2017, March) Driving user growth with performance improvements. [Online]. Available: https://medium.com/@Pinterest_Engineering/driving-user-growth-with-performance-improvements-cfc50dafadd7

[7] G. Pinto and F. Castor, "Energy efficiency: A new concern for application software developers," ACM, Tech. Rep., 2018. [Online]. Available: http://gustavopinto.org/lost+found/cacm2017.pdf

[8] Google. (2018, August) Lighthouse. [Online]. Available: https://developers.google.com/web/tools/lighthouse/

[9] M. K. . A. O. . K. B. . J. Miller. (2018, August) Measure performance with the rail model. [Online]. Available: https://developers.google.com/web/fundamentals/performance/rail

[10] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Kluwer, 2012.

[11] Amazon. (2018) The top 500 sites on the web. [Online]. Available: https://www.alexa.com/topsites

[12] Statscounter, "Browser market share worldwide," GlobalStats, Tech. Rep., 2018. [Online]. Available: http://gs.statcounter.com/browser-market-share

[13] T. S. Portal, "Global market revenue share of leading smartphone applications processor vendors in 2014 and 2017," Statista, Tech. Rep., 2017. [Online]. Available: http://gs.statcounter.com/browser-market-share

[14] Google. (2018) Lighthouse v3 scoring guide. [Online]. Available: https://developers.google.com/web/tools/lighthouse/v3/scoring#perf-scoring

[15] S. Sheombar, "Randomwebsite," GitHub Repository, october 2018. [Online]. Available: https://github.com/JaySheombar/GreenLab-RandomWebsite

[16] (2018, october) Lighthouse batch reporter. [Online]. Available: https://www.npmjs.com/package/lighthouse-batch

[17] R-Studio. (2018) R-studio. [Online]. Available: https://www.rstudio.com/products/rstudio/

[19] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, F. Leisch, C.-C. Chang, and C.-C. Lin, "Misc functions of the department of statistics, probability theory group (formerly: E1071), tu wien," TU Wien, Tech. Rep., 2018. [Online]. Available: https://cran.r-project.org/web/packages/e1071/e1071.pdf

[20] T. D. Cook and D. T. Campbell, *Quasi-Experimentation: Design and Analysis Issues for Field Settings*. Houghton Mifflin, 1979.