International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

Novel Approach for Job Offloading Technique in Mobile Cloud Computing

Nitinkumar Rajnikant Pandya [1]*, Dr. Ankit Shah [2]
[1]Research Scholar, Sankalchand Patel University, Visnagar, Gujarat, India.
[1]Assistant Professor, Parul Institute of Engineering & Technology (PIET),
Parul University, Vadodara, Gujarat, India.
email: nitin.pandya85@gmail.com, nitinkumar.pandya19192@paruluniversity.ac.in
[2.] Associate Professor, Computer Engineering Department,
Gandhinagar Institute of Technology, Gandhinagar University,
Gujarat, India.
email: shah_ankit101@yahoo.co.in

Abstract*:* User preferences for computing have evolved as a result of recent advancements in mobile computing technologies. Mobile devices (SMDs) are still low potential computer platforms with capped memory sizes, CPU speeds, and battery life. Computationally heavy mobile apps may be run on SMDs thanks to Mobile Cloud Computing (MCC), which uses computational offloading. In both the business and academic worlds, there is rising interest in the mobile cloud computing is a new computing paradigm. Lack of a comprehensive experimental framework to use in their experiments and to evaluate their proposed work is a serious issue for mobile cloud computing researchers. Through MCC, mobile devices will be able to serve a wider variety of resource-intensive tasks while maintaining and growing their resource pools. In order to improve the mobile user experience, it places a high priority on improving energy efficiency, storage capacity, computational power, and data security. Since both the mobile device and the Cloud must determine energy-time trade-offs and decisions made on one side have an influence on the performance of the other, designing MCC systems is a challenging task. According to an examination of the MCC literature, all present models are centred on mobile devices, with the Cloud viewed as a system with infinite resources. Furthermore, no MCC-specific simulation tool is currently known to exist. To fill this need, we present in this study a Novel Approach for Job Offloading in Cloud Environments such as Google Cloud, using OCR application, while attempting to reduce energy use (Power). We are measuring the results of this experiment on both Cloud Computing and Mobile Device Computing.

Keywords*:* Computational offloading, mobile cloud computing, Service Level Agreement, energy efficiency, and portability

I. INTRODUCTION

The most recent advancements in mobile computing technologies have transformed consumer computing choices. Recently, smartphones have displaced a variety of
computer and communication equipment in favour of all-in-one ubiquitous computing devices such as PDAs, digital cameras, Internet surfing devices, and Global Positioning Systems (GPS). In many areas, including business, e-learning, entertainment, gaming, management information systems, and healthcare, people are becoming more and more dependent on modern cell phones. Mobile devices are expected to be the dominating future computing devices, with high user expectations for access to computationally intensive apps comparable to powerful stationary computing equipment. [1][2][3].

SMDs are still low-potential computing devices, and the battery life, CPU capability, and memory capacity of the newest smartphones and tablets are their limitations. MCC processes computationally expensive mobile apps using the application processing capabilities of computational clouds. Recently, a variety of computational offloading frameworks have been suggested for MCC to process demanding mobile applications. [4][5][6].

In comparison to traditional personal computers, smartphones and tablet PCs are increasingly common and are being used more frequently. We have entered what is known as the era of mobile computing as a result of this, which has caused the market for mobile devices to see unheard-of growth. Because of their constrained resources, mobile devices have significant drawbacks. There are a number of restrictions on their use, including as poor connection, constrained bandwidth, security flaws, incompatibility with some applications, and a limited range of power sources because batteries are their primary power source. [7][8].

Today's mobile service providers and consumers are shown a strong interest in the new MCC idea. It is built on the merging of cloud computing providers with mobile service systems. The ability to transfer mobile data and intense compute requirements to cloud computing infrastructures enables this integration. [9].

As a result, the mobile device may move computationally demanding activities to cloud infrastructures while preserving its own resources. Due to the possibility of storing user data in a cloud storage system, mobile devices would no longer need a lot of internal storage space. When needed, the user can quickly recover his or her data. MCC research is rapidly developing, with numerous academics focusing on subjects like as data offloading, mobile-cloud subscription, and security [10][11].

MCC system design is a difficult challenge since both the mobile device and the cloud must identify energy-time trade-offs, and decisions made by one impact the performance of the other. According to the report, all MCC models are centred on mobile devices, with the Cloud viewed as a system with infinite resources. Furthermore, as far as we know, no MCC-specific simulation tool exists. [12][13].

We suggest a modelling and Live Cloud Environment for the design and study of MCC systems in order to help close this gap.[14][15]. Job offloading for the mobile device is the first pillar. Energy and performance concerns influence offloading options. One strategy is to analyse the energy costs for each work under the two different execution scenarios on the mobile device or in the cloud for each job.

The Service Level Agreement (SLA) for the MCC is the second pillar. At various MCC levels, there are several Service Level Agreements, such as IAAS, PAAS, and SAAS. At PAAS, we have put a lot of emphasis on numerous SLA Parameters including integration, scalability, pay as you go, deployment environments, browser, etc. The third pillar is integration with Live Cloud in MCC. There are many Cloud (Public or Private) are available for integration with MCC like, Microsoft Azure, AWS, Google Cloud, etc. we have integrated with Google Cloud in our proposed work using Firebase services [16][17][18].

The following portions of the paper are categorised. The second section goes about computational offloading frameworks for MCC Related Work. The working method of Energy Efficiency employing SLA parameters is discussed in Section 3 along with the suggested framework, which also exhibits its unique qualities. Section 4 goes through the techniques used to evaluate the proposed framework. The results are presented and the experimental findings are covered in Section 5. Section 6 concludes with some last thoughts and

II. RELATED WORK

Muhannad, et al. [1] and Amoretti, Michele et al. [2] have derived that, the primary question on the mobile platform is always: to offload or not to offload? This is true even if MCC may be implemented using several methodologies that vary in the compute granularity addressed by the offloading process (ranging from device cloning to application segmentation and migration) and in the degree of engagement of the Cloud.

Shiraz, Muhammad, et al [3] have worked on Mobile Assistance Using Infrastructure (MAUI) focuses on SMD energy savings. The local and remote components of an application are identified by application developers during the design phase. The MAUI profiler decides if a technique for offload processing that is remotely annotated is feasible. When a method is invoked, the profiler component evaluates it for energy savings, which uses more computational resources (CPU, battery) on SMD.

Shiraz, Muhammad, et al [3] and Mukherjee, Islam [4] have find out the main three stages or ways of offloading. Three stages for offloading initialization, computational offloading, and remote application execution make up the processing of cloud-based applications in conventional computational offloading frameworks. (a) During the start-up phase, reports of the network and mobile context are gathered from the respective sensor modules, and the services that are available on the cloud server node are found. (b) Decisions are made during the computational offloading process, and parameter setups are carried out. Application offloading and partitioning decisions, user authentication, and permission decisions are all part of this step. (c) Following configuration, the delegated application is performed on the remote server node and its running state is restarted on the remote virtual device instance.

Asharul, et al [5], have identified the traditional code offloading uses computational offloading at runtime, which necessitates virtual device instances on cloud server nodes by utilising IaaS services of computational. When an application is operating, its binary files and data states are offloaded, increasing the energy cost of computational offloading for MCC.

III. PROPOSED FRAMEWORK FOR JOB ALLOCATION MECHANISM

The Proposed algorithm for job Offloading is as below.

Novel Algorithm for job Offloading (Algorithm)

```
Step 1: Start
Step 2: Take the Image /Predefined Image (Job arrival)
Step 3: the job will be executed by 3 ways,
If Local Device Execution then
        the job will be executed on Local Devices only
If Google Cloud Computing Execution then
        the job will be executed on google Cloud
If Autonomous then,
        it will decide based on the image
            if the words <20 then the job will be executed on Local Devices
            else the job will be executed on google Cloud
Step 4: collect the result and compare the CPU, Memory, Energy Efficiency in the Mobile
Step 5: End
```

The Nobel Algorithm for job Offloading is also described as Flowchart as below.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

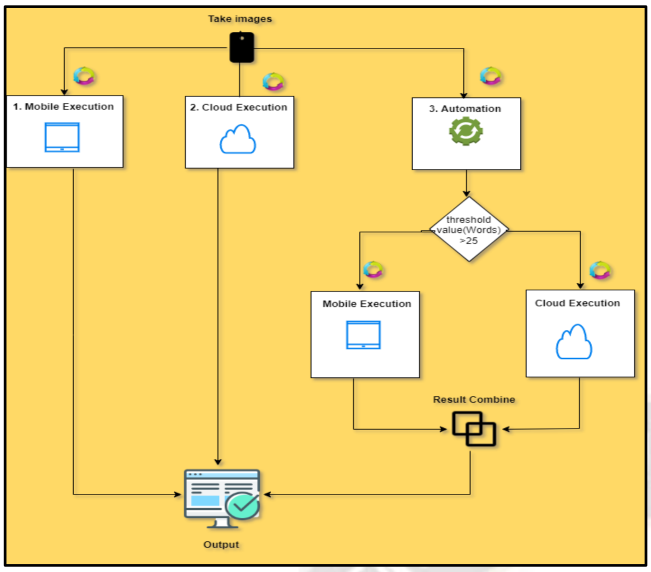


Figure 1. Proposed Algorithm for Novel Job Allocation Methodology in MCC

IV. IMPLEMENTATION & EXPERIMET SETUP

We have created following experimental Setup for our proposed algorithm and implementation,

- **Software:** Android Studio 2021.2.1 Version with kotlin
- **Application:** Optical Character Recognition
- **Cloud:** Google Cloud (Firebase Service)
- **Mobile Device:** Any Android Device (higher API level 20)
- **Mode of Execution:** Using USB/ Apk/Emulation
- Measure the Parameter for Cloud and Mobile Device:
  1. CPU usage
  2. Energy Consumption
  3. Memory Usage
- **Power Tutor:** Measure the Parameter in Mobile Device

The different steps for making experiment we have considered OCR application in MCC, there are some steps we described for Mobile Cloud Computing and Mobile Device Computing. We have also taken some SLA parameters for Energy Efficiency in MCC which is described as below.

A. Steps for Mobile Cloud Computing Execution

1. In device, we will integrate firebase authentication and firebase function SDK.
2. From the device, Firebase function will be called through Firebase SDK, In that, image base 64 data will be passed
3. Firebase function will call google cloud vision api with base 64 of image
4. Cloud vision API will process the Base 64 string of the image and return the Text to firebase function
5. Firebase function will return data to mobile

B. Steps for Local Mobile Device Computing

1. In Device, Google ML kit SDK will be integrated function
2. from the device, Image will be processed to extract the text through ML kit
3. That SDK function will return the text

C. SLA Parameters

We have considered the following SLA Parameter at PASS Level for job offloading with its description in our Experimental setup using MCC.

TABLE I. SLA PARAMETERS OF PAAS FOR JOB OFFLOADING ON GOOGLE CLOUD

Parameter	Description
Integration	Integration with Google Cloud Fire base Services
Scalability	No of Images that we can increase up to 1000 (Predefined)
Pay as you go billing	Charging based on resources or time of service
Environments of deployment	Supporting offline (Mobile Devices) and cloud systems (Google Cloud Fire base Services) developed in Android studio
Browsers	Google Chrome, Firefox, Explorer,

The SLA Parameters we have considered according to the requirement of application which will be executed on the Cloud Service provider as well as user requirement. Here we have taken the SLA Parameter with its specification / Description which we had used in Google Cloud (Firebase Services) and Android Studio (Software) and Browser like Google Chrome.

D. Experiments Screens

We have Perform the Experiments in the Android studio (2021.2.1) using USB debugger by enabled the Developer option in the Mobile Devices. We have installed the application directly by creating the APK file of the application in any Smart Phone (mobile devices) who have higher API level 20.

When we start the Application, we will find the application start view as shown in below Fig. 2 which shows the First Camera View and we can click the images and then it shows the option from where we can go for the execution like local text Detection, Cloud Text Detection, automated text detection (some images go for a local execution and some images go for a cloud execution).

Once we taken the number of images or photos from camera and go for the any one of the options like Cloud text detection or local text detection or automated text detection the images will be converted into text and the performance parameter like CPU usage, Memory usage and Energy (Power) usage, of the Mobile devices (using Power Tutor App) as well as the Cloud computing (using Android Studio Profiler) we will measure.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

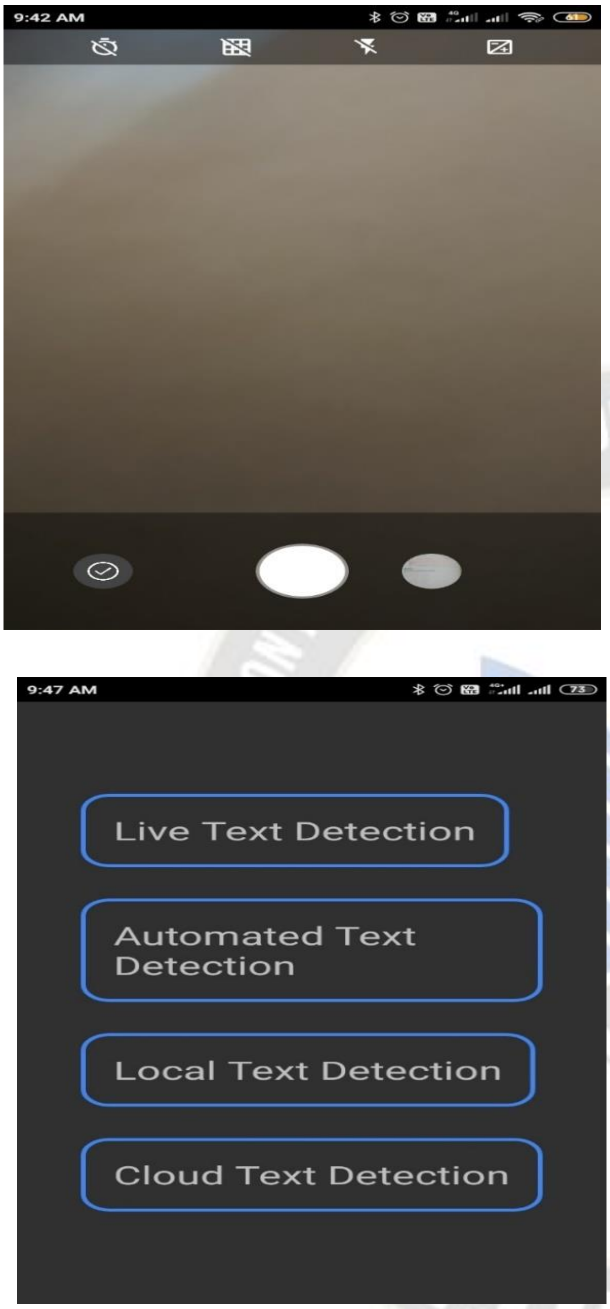


Figure 2. (a) Application Start Camera view (b) After taking images the Job Offloading options screen

First, we had executed the application in the Google Cloud Environment and measure the parameters as shown in Fig. 3.

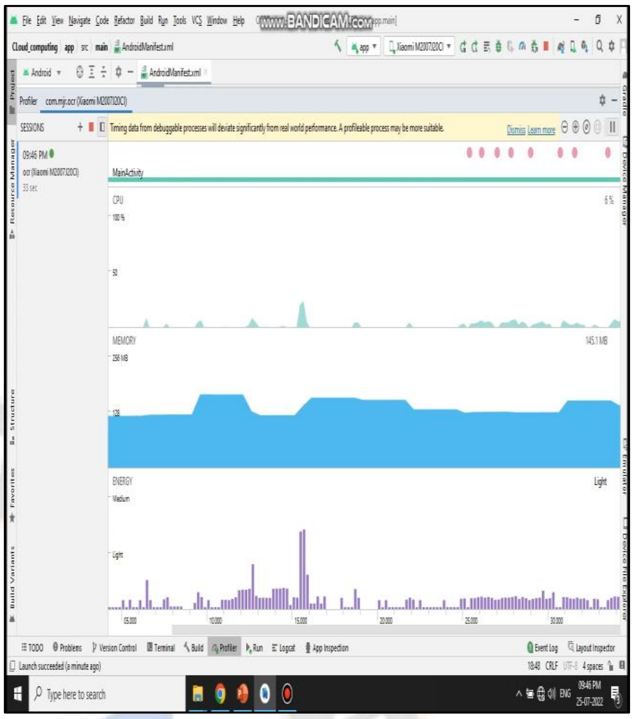
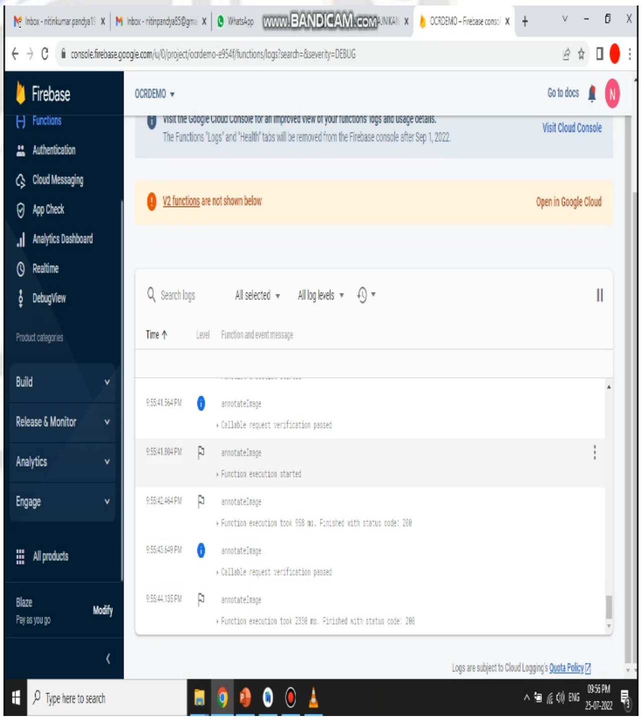


Figure 3. Job Offloaded on Cloud Computing through Mobile Device shows CPU, Memory and Power (Energy) Utilization in Computer

Figures 4 and 5 demonstrate how the task was offloaded to the Google Cloud, where the photographs were uploaded and the outcome was shown on mobile devices.

.



console.firebase.google.com/u/0/project/ocrdemo-e954f/functions/logs?search=&severity=DEBUG

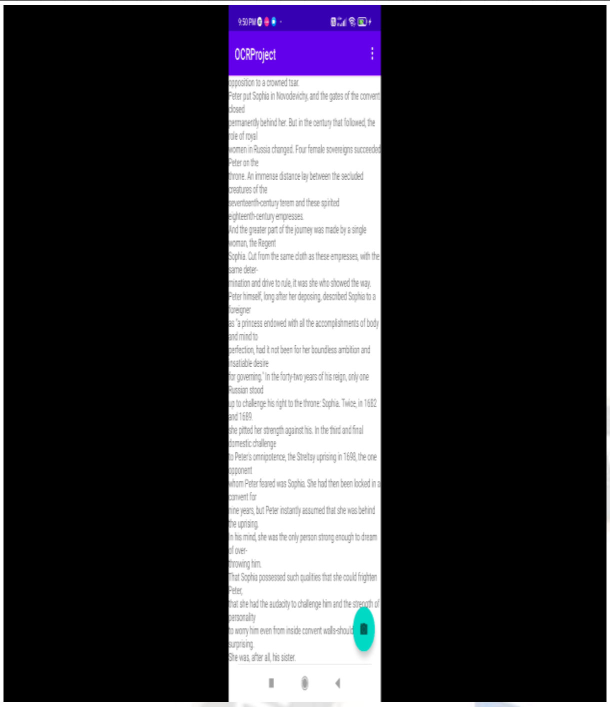
Visit the Google Cloud Console for an improved view of your functions' logs and usage details.
The Functions "Logs" and "Health" tabs will be removed from the Firebase console after Sep 1, 2022.

V2 functions are not shown below

```
Time    Level  Function and event message
9:55:41.564 PM   annotateImage
                 Callable request verification passed
9:55:41.804 PM   annotateImage
                 Function execution started
9:55:42.464 PM   annotateImage
                 Function execution took 958 ms. Finished with status code: 200
9:55:43.649 PM   annotateImage
                 Callable request verification passed
9:55:44.135 PM   annotateImage
                 Function execution took 2330 ms. Finished with status code: 200
```

Logs are subject to Cloud Logging's Quota Policy

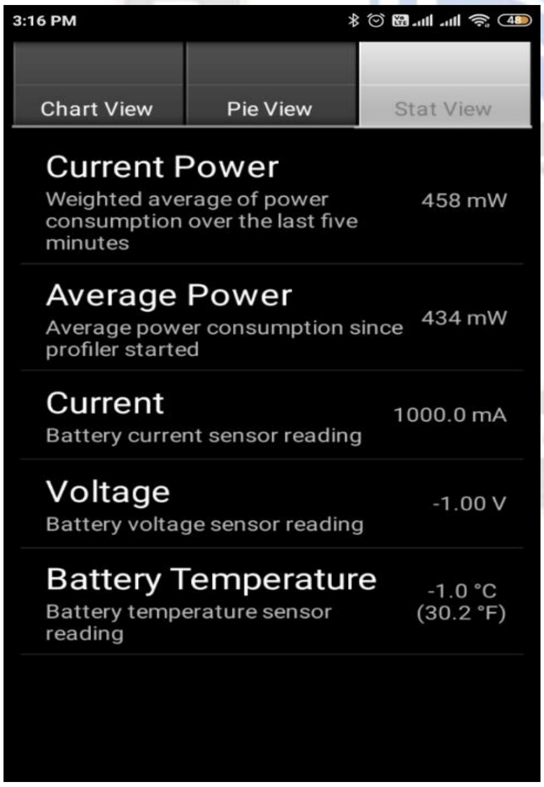
Figure 4. Job Offloaded on Cloud Computing through Mobile Device shows the images offloaded on google Cloud (using Firebase services)



opposition to a crowned tsar.
Peter put Sophia in Novodevichy, and the gates of the convent closed
permanently behind her. But in the century that followed, the role of royal
women in Russia changed. Four female sovereigns succeeded Peter on the
throne. An immense distance lay between the secluded creatures of the
seventeenth-century terem and these spirited
eighteenth-century empresses.
And the greater part of the journey was made by a single woman, the Regent
Sophia. Cut from the same cloth as these empresses, with the same deter-
mination and drive to rule, it was she who showed the way.
Peter himself, long after her deposing, described Sophia to a foreigner
as "a princess endowed with all the accomplishments of body and mind to
perfection, had it not been for her boundless ambition and insatiable desire
for governing." In the forty-two years of his reign, only one Russian stood
up to challenge his right to the throne: Sophia. Twice, in 1682 and 1689,
she pitted her strength against his. In the third and final domestic challenge
to Peter's omnipotence, the Streltsy uprising in 1698, the one opponent
whom Peter feared was Sophia. She had then been locked in a convent for
nine years, but Peter instantly assumed that she was behind the uprising.
In his mind, she was the only person strong enough to dream of over-
throwing him.
That Sophia possessed such qualities that she could frighten Peter,
that she had the audacity to challenge him and the strength of personality
to worry him even from inside convent walls-should
surprising.
She was, after all, his sister.

Figure 5. Job Offloaded on Cloud Computing through Mobile Device (using USB Debugging) Shows the Result in Mobile Devices

Figure 6 shows the different view (like pie view, chart view) of Energy/Power usage, CPU usage, LCD, Wi-Fi, etc. parameters measures in the mobile devices



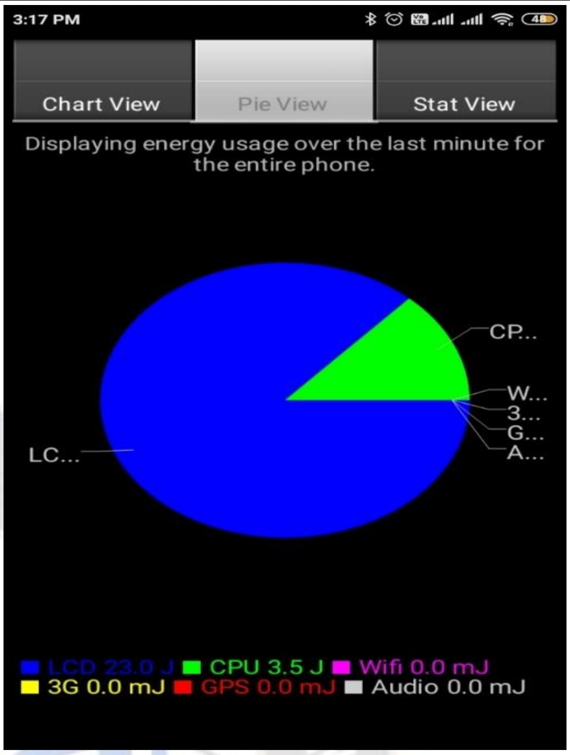
Current Power
Weighted average of power consumption over the last five minutes 458 mW

Average Power
Average power consumption since profiler started 434 mW

Current
Battery current sensor reading 1000.0 mA

Voltage
Battery voltage sensor reading -1.00 V

Battery Temperature
Battery temperature sensor reading -1.0 °C (30.2 °F)



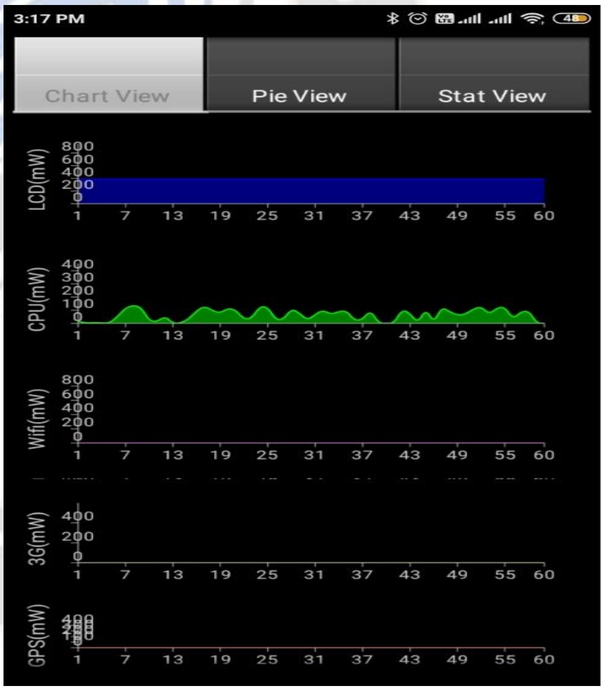


Figure 6. Job Offloaded on Cloud Computing through Mobile Device shows CPU and Power Utilization

The parameters were measured after we had run the programme in the local mobile device environment, as shown in Fig. 7.

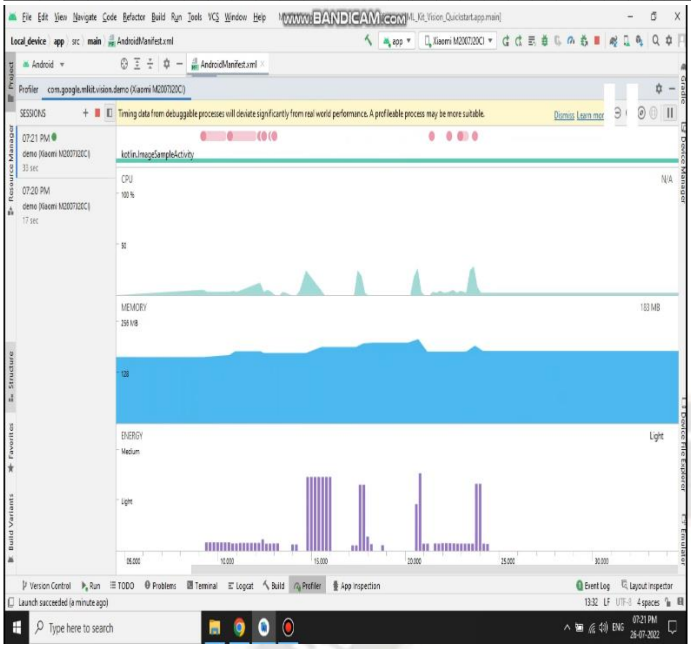
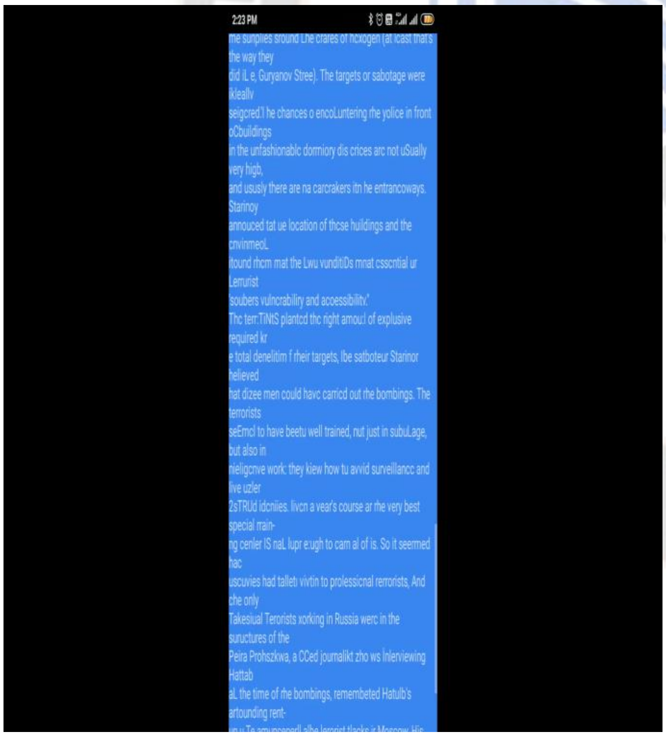

Profiler com.google.mlkit.vision.demo (Xiaomi M2007J20CI)
SESSIONS
Timing data from debuggable processes will deviate significantly from real world performance. A profileable process may be more suitable.
07:21 PM
demo (Xiaomi M2007J20CI)
07:20 PM
kotlin.ImageSampleActivity
CPU
100 %
MEMORY
183 MB
256 MB
ENERGY
Medium
Light

Figure 7. Job Offloaded through Mobile Device shows CPU, Memory and Power (Energy) Utilization in Computer

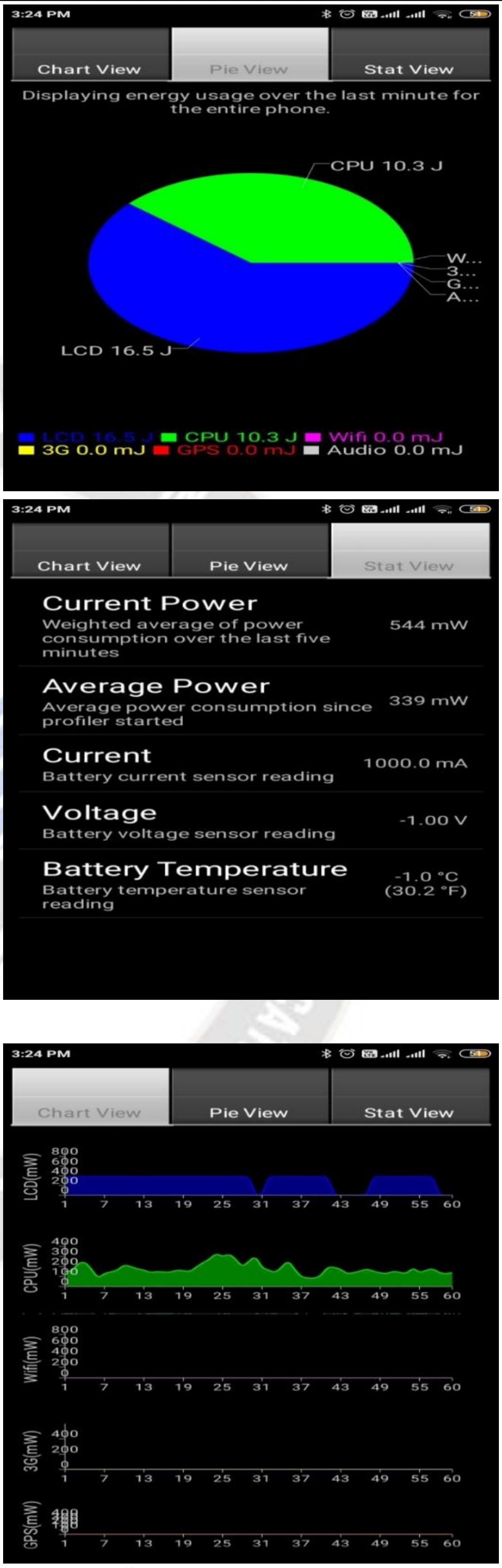
Figure 8 shows that the job offloading done in the Local Mobile Devices and shows the result back in to the Mobile devices.


2:23 PM

me sunplies sround Lhe crares of ncxogen (at lcast that's
the way they
did iL e, Guryanov Stree). The targets or sabotage were
ikleallv
seigcred.'I he chances o encoLuntering rhe yolice in front
oCbuildings
in the unfashionablc dormiory dis crices arc not uSually
very higb,
and ususly there are na carcrakers itn he entrancoways.
Starinoy
annouced tat ue location of thcse huildings and the
cnvinmeoL
itound rhcm mat the Lwu vunditiDs mnat csscntial ur
Lerrurist
'soubers vulncrabiliry and acoessibility.'
Thc terr:TiNtS plantcd thc right amou:l of explusive
required kr
e total denelitim f rheir targets, lbe satboteur Starinor
helieved
hat dizee men could havc carricd out rhe bombings. The
terrorists
seEmcl to have beetu well trained, nut just in subuLage,
but also in
nieligcnve work: they kiew how tu avvid surveillancc and
live uzler
2sTRUd idcniies. livcn a vear's course ar rhe very best
special main-
ng cenler IS naL lupr e:ugh to cam al of is. So it seemed
hac
uscuvies had talleti vivtin to prolessicnal rerrorists, And
che only
Takesiual Terorists xorking in Russia werc in the
suructures of the
Peira Prohszkwa, a CCed journalikt zho ws Inlerviewing
Hattab
aL the time of rhe bombings, remembeted Hatulb's
artounding rent-

Figure 8. Job Offloaded through Mobile Device (using USB Debugging) Shows the Result in Mobile Devices

Figure 9 shows the different view (like pie view, chart view) of Energy/Power usage, CPU usage, LCD, Wi-Fi, etc. parameters measures in the mobile devices



Current Power
Weighted average of power consumption over the last five minutes — 544 mW

Average Power
Average power consumption since profiler started — 339 mW

Current
Battery current sensor reading — 1000.0 mA

Voltage
Battery voltage sensor reading — -1.00 V

Battery Temperature
Battery temperature sensor reading — -1.0 °C (30.2 °F)

Figure 9. Job Offloaded on Local Mobile Device shows CPU and Power Utilization

V. RESULT AND DISCUSSION

As we have observed mainly 3 parameters like CPU, Memory and Energy (Power) using Power Tutor application in Mobile Devices for measuring the performance of the Mobile Devices.

When we execute the application in both Mobile devices as well as the Cloud computing, we have also observed the same parameters in computer also using Android studio Profiler. Then we have compared it with the Local Devices Computing and Computer computing and compare the result as shown in Table 2.

Table 2 allows us to state that, CPU usage is 63 % better than the local mobile devices computing and memory usage is 26% less than the Mobile devices.

TABLE II. CPU, MEMORY AND ENERGY UTILIZATION FOR CLOUD AND MOBILE DEVICES SHOWS IN COMPUTER LAPTOP (ANDROID STUDIO)

Parameters	**Cloud Computing (Computer)**	**Local Device Computing (Computer)**	**% better in Cloud Computing**
CPU	9%	24%	**63**
Memory	169.1 MB	228.3 MB	**26**
Energy (Power)	Light	Medium	-

Figure 10 shows the comparison of CPU and Memory utilization in the computer when we offload the job in the Cloud Computing

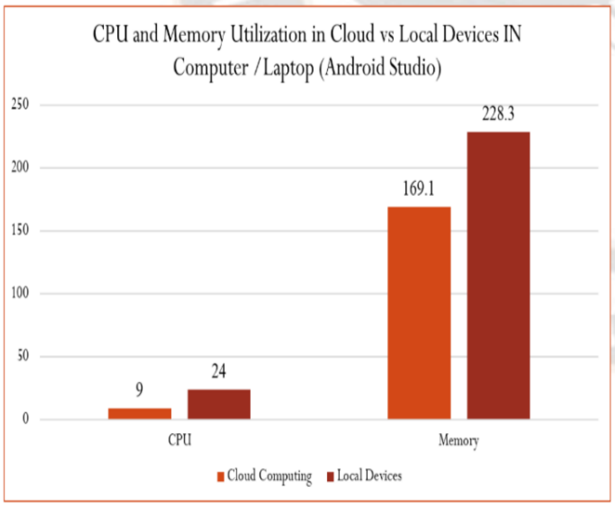


Figure 10. CPU and Memory Utilization in Cloud vs Local Devices IN Computer /Laptop (Android Studio)

Table 3 shows the Comparison of CPU, Memory and Energy (Power) in Mobile devices using Power Tutor app in J/M when we offload the Job on the Local Mobile Devices.

From the Table 3 we can say that, CPU usage is 64 % better than the local mobile devices computing, memory usage is 19% less than the Mobile devices and Energy is 16% reduced or less than the mobile devices.

TABLE III. CPU, MEMORY AND ENERGY UTILIZATION FOR CLOUD AND MOBILE DEVICES SHOWS IN MOBILE USING POWER TUTOR J/M

Parameters	**Cloud Computing (Mobile Utilization) in J/M**	**Avg. Utilization**	**Local Device Computing (Mobile Utilization) J/M**	**Avg. Utilization**	**% better in Cloud Computing**
CPU	3.5 J/M to 4.1 J/M	3.8 J/M	6.3 J/M to 14.7 J/M	10.5 J/M	**64**
Memory	1870 MB to 1902 MB	1886 MB	2280 MB to 2392 MB	2336 MB	**19**
Energy (Power)	$32.58*10^9$ J/M	$32.58*10^9$ J/M	$38.94*10^9$ J/M	$38.94*10^9$ J/M	**16**

Figure 11 shows the comparison of CPU and Energy (Power) utilization in the Mobile when we offload the job in the Local Mobile Devices and google Cloud Computing in J/M

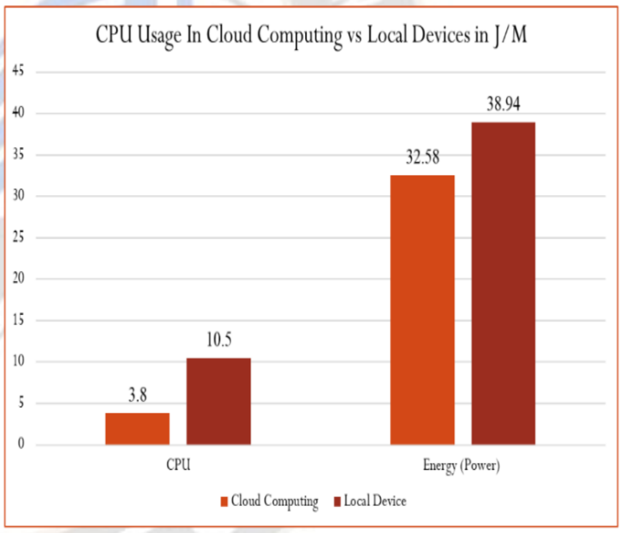


Figure 11. CPU and Energy Utilization in Cloud vs Local Devices in Mobile Devices using Power Tutor (J/M)

Figure 12 shows the memory usage comparison in Mobile Devices and Cloud Computing when we offload the job, the Cloud Computing is memory usage is 16% less compared to Mobile Devices.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

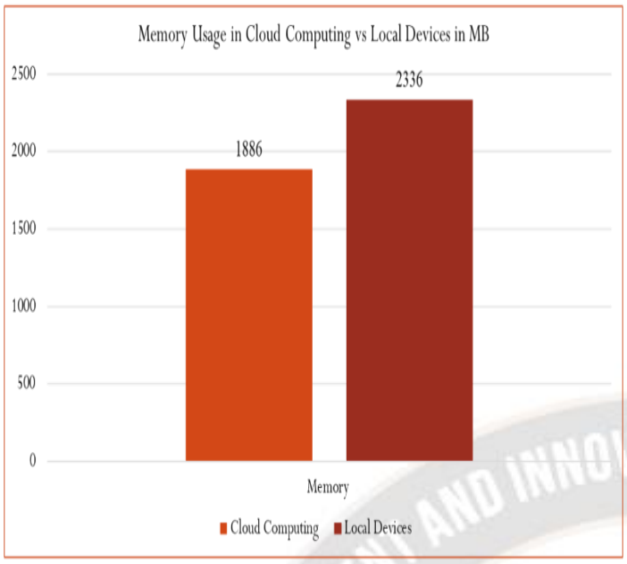


Figure 12. Memory Usage in Cloud Computing vs Local Devices (MB) in Mobile Devices using Power Tutor

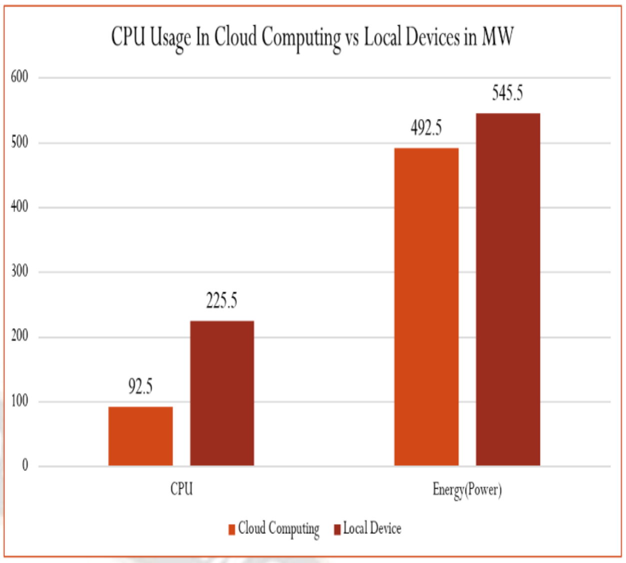


Figure 13. CPU and Energy Utilization in Cloud vs Local Devices in Mobile Devices using Power Tutor (MW)

Table 4 shows the Comparison of CPU, Memory and Energy (Power) in Mobile devices using Power Tutor app in MW when we offload the Job on the Local Mobile Devices as well as Cloud Computing.

From the Table 4 we can say that, CPU usage is 59% better than the local Mobile Devices computing, memory usage is 19% less than the Mobile Devices and Energy is 10% reduced or less than the Mobile Devices.

TABLE IV. CPU, MEMORY AND ENERGY UTILIZATION FOR CLOUD AND MOBILE DEVICES SHOWS IN MOBILE USING POWER TUTOR MW

Parameters	**Cloud Computing (Mobile tilization) in MW**	**Avg. tilization**	**Local Device Computing (Mobile Utilization) MW**	**Avg. tilization**	**% better in Cloud Computing**
CPU	87 MW to 98 MW	92.5 MW	182 MW to 269 MW	225.5 MW	**59**
Memory	1870 MB to 1902 MB	1886 MB	2280 MB to 2392 MB	2336 MB	**19**
Energy (Power)	442 MW to 543 MW	492.5 MW	442 MW to 649 MW	545.5 MW	**10**

Figure 13 shows the CPU and Energy (Power) usage comparison in Mobile Devices and Cloud Computing when we offload the job, the Cloud Computing is CPU usage is 59% less compared to Mobile Devices and Energy (Power) usage is 19 % better than the Mobile Devices.

VI. CONCLUSION AND FUTURE WORK

As the Experimental result, we had done in the Mobile Cloud Computing environment using android studio and the power tutor application (in Mobile Devices), we can conclude that the job offloading Mechanism we applied for the MCC environment is better than the many of existing algorithm for job offloading in MCC.

From the comparisons, we can conclude that the job offloaded in the Mobile Devices and the job offloaded in the Cloud Computing, the Decision made by our algorithm is better than the existing algorithms and the approach we have taken it's a Novel approach for job offloading.

In future, we will try to optimize our offloads decision algorithm using AI&ML and try to execute it from the live text detection techniques using AI&ML.

ACKNOWLEDGMENT

This research was supported by my Guide Dr. Ankit Shah, we are thankful to my colleagues who provided expertise that greatly assisted the research, although they may not agree with all of the interpretations provided in this paper.

REFERENCES

[1] Quwaider, Muhannad, et al. "Experimental framework for mobile cloud computing system." Procedia Computer Science 52 (2015): 1147-1152.

[2] Amoretti, Michele, Alessandro Grazioli, and Francesco Zanichelli. "A modeling and simulation framework for mobile cloud computing." Simulation modelling practice and theory 58 (2015): 140-156.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

[3] Shiraz, Muhammad, et al. "Energy efficient computational offloading framework for mobile cloud computing." Journal of Grid Computing 13.1 (2015): 1-18.

[4] Mukherjee, Anwesha, and Debashis De. "Low power offloading strategy for femto-cloud mobile network." Engineering Science and Technology, an International Journal 19.1 (2016): 260-270.

[5] Islam, Asharul, et al. "Efficient resourceful mobile cloud architecture (mRARSA) for resource demanding applications." Journal of Cloud Computing 9.1 (2020): 9.

[6] Aliyu, Ahmed, et al. "Mobile cloud computing: taxonomy and challenges." Journal of Computer Networks and Communications 2020 (2020).

[7] Rahman, Mazedur, Jerry Gao, and Wei-Tek Tsai. "A Survey of Energy Saving Strategies in Mobile Cloud."

[8] Pramanik, Pijush Kanti Dutta, et al. "Power consumption analysis, measurement, management, and issues: a state-of-the-art review of smartphone battery and energy usage." IEEE Access 7 (2019): 182113-182172.

[9] Singh, Ajmer. "Taxonomy of SLA violation minimization techniques in cloud computing." 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT). IEEE, 2018.

[10] Yang, Tinging, et al. "Multivessel computation offloading in maritime mobile edge computing network." IEEE Internet of Things Journal 6.3 (2018): 4063-4073.

[11] Justino, Tiago, and Rajkumar Buyya. "Outsourcing resource-intensive tasks from mobile apps to clouds: Android and aneka integration." 2014 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM). IEEE, 2014.

[12] Tang, Wenyi, et al. "A novel task allocation algorithm in mobile crowdsensing with spatial privacy preservation." Wireless Communications and Mobile Computing 2019 (2019).

[13] Mastroeni, Loretta, Alessandro Mazzoccoli, and Maurizio Naldi. "Service Level Agreement Violations in Cloud Storage: Insurance and Compensation Sustainability." Future Internet 11.7 (2019): 142.

[14] Isaac, Odun-Ayo, Blessing Udemezue, and Abiodun Kilanko. "Cloud Service Level Agreements and Resource Management." Advances in Science, Technology and Engineering Systems Journal 4.2 (2019): 228-236.

[15] Shen, Chao, Shengjun Xue, and Shucun Fu. "ECPM: an energy-efficient cloudlet placement method in mobile cloud environment." EURASIP Journal on Wireless Communications and Networking 2019.1 (2019): 141.

[16] Liu, Li, and Qi Fan. "Resource allocation optimization based on mixed integer linear programming in the multi-cloudlet environment." IEEE Access 6 (2018): 24533-24542.

[17] Peng, Hua, et al. "Joint optimization method for task scheduling time and energy consumption in mobile cloud computing environment." *Applied Soft Computing* 80 (2019): 534-545

[18] Guo, Kai, et al. "Efficient resource assignment in mobile edge computing: A dynamic congestion-aware offloading approach." Journal of Network and Computer Applications 134 (2019): 40-51.

[19] Babu, KR Remesh, and Philip Samuel. "Energy aware clustered load balancing in cloud computing environment." International Journal of Networking and Virtual Organisations19.2-4 (2018): 305-320.

[20] Pandya, Nitinkumar Rajnikant, and Ankit Shah. "Job Offloading Techniques for Increasing Mobile Cloud Computing's Energy Efficiency." International Journal of Intelligent Systems and Applications in Engineering 10.4 (2022): 327-333.