DAFTAR PUSTAKA

A query language for your API. (n.d.). Retrieved October 22, 2022, from https://graphql.org/

Ala-Laurinaho, R., Mattila, J., Autiosalo, J., Hietala, J., Laaki, H., & Tammi, K. (2022). Comparison of REST and GraphQL Interfaces for OPC UA. *Computers*, *11*(5), 65. https://doi.org/10.3390/computers11050065

Apa itu Amazon Relational Database Service (Amazon RDS)? - Amazon Relational Database Service. (n.d.). Retrieved October 23, 2022, from https://docs.aws.*Amazon*.com/id_id/*Amazon*RDS/latest/UserGuide/Welcome.html

Apache JMeter—User's Manual: Glossary. (n.d.). Retrieved October 23, 2022, from https://jmeter.apache.org/usermanual/glossary.html

Chrome DevTools. (n.d.). Chrome *Developer*s. Retrieved October 23, 2022, from https://*developer*.chrome.com/docs/*DevTools*/

Brito, G., Mombach, T., & Valente, M. T. (2019). *Migrating to GraphQL: A Practical Assessment*. https://doi.org/10.1109/SANER.2019.8667986.

DBaas Comparison. (n.d.).

Dustdar, S., & Schreiner, W. (2005). A survey on web *Services* composition. In *Int. J. Web and Grid Services* (Vol. 1, Issue 1). http://www.infosys.tuwien.ac.at/Staff/sd

Erlandsson, P., & Remes, J. (2020). *Performance comparison: Between GraphQL, REST & SOAP (Dissertation)*. Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-18713 .

Gottschalk, K. (2002). *Introduction to web Services architecture Cite this paper Introduction to Web Services architecture*.

GraphQL is better than Rest. (n.d). Retrieved March 6, 2022, from https://www.howtographql.com/basics/1-graphql-is-the-better-rest/

Halili, E. H. (2008). *Apache JMeter : a practical beginner's guide to automated testing and performance measurement for your websites*. PacktPub.

Helgason, A. F. (2017). *Performance analysis of Web Services: Comparison between RESTful & GraphQL web Services (Dissertation)*. Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-13683 .

Introduction to GraphQL. (n.d.). Retrieved March 7, 2022, from https://graphql.org/learn/.

Komputer, W. (2010). *Panduan Belajar MySQL Database Server*. MediaKita. https://books.google.co.id/books?id=rR1VNYpc08wC

Kumar Mungara, K. (n.d.). *Basic Components of Web Service Basic Components of Web Service Under Graduate, Guru Nanak Institutions Technical Campus*. https://doi.org/10.46647/ijetms.2020.v04i04.00

Khannedy, E, K. (2021, July 24). BELAJAR *RESTFUL API* (BAHASA INDONESIA). https://www.youtube.com/watch?v=9ed3b0tSRvI

The PHP Frameworks for Web Artisans. (n.d.). Retrieved October 22, 2002, from https://laravel.com/

Laravel—The PHP Framework For Web Artisans. (n.d.). Retrieved October 23, 2022, from https://laravel.com/

Lawi, A., Panggabean, B. L. E., & Yoshida, T. (2021). Evaluating graphql and rest api *Services* performance in a massive and intensive accessible information system. *Computers*, *10*(11). https://doi.org/10.3390/computers10110138.

Lerdorf, R., Tatroe, K., Kaehms, B., McGredy, R., Torkington, N., & Ferguson, P. M. (2002). *Programming PHP*. O'Reilly Media, Incorporated. https://books.google.co.id/books?id=7OjvOmol3CcC

Madeja, M. (2022). Simple and practical: Laravel with GraphQL. International PHP Conference. https://phpconference.com/blog/simple-and-practical-laravel-with-graphql/

Maldonado, FreeCodeCamp (2019). *Why GraphQL is the future of APIs*. Retrieved March 5, 2022, from https://www.freecodecamp.org/news/why-graphql-is-the-future-of-apis-6a900fb0bc81/

Massé, M. (n.d.). *REST API Design Rulebook*. www.allitebooks.com.

Murty, J. (2008). *Programming Amazon Web Services: S3, EC2, SQS, FPS, and SimpleDB*. O'Reilly Media. https://books.google.co.id/books?id=NtqbAgAAQBAJ

National University of Sciences and Technology (Islāmābād, P. M. C. of S., Institute of Electrical and Electronics Engineers. Islamabad Section, & Institute of Electrical and Electronics Engineers. (n.d.). *International Conference on Communication Technologies (ComTech - 2017): 19-21 April 2017, Military College of Signals, National University of Sciences & Technology*.

Prayogi, A. A., Niswar, M., Indrabayu, & Rijal, M. (2020). Design and Implementation of REST API for Academic Information System. *IOP Conference Series: Materials Science and Engineering*, *875*(1). https://doi.org/10.1088/1757-899X/875/1/012047

Putra, Dicoding. (2019). GraphQL vs REST API: Apa bedanya?. Retrieved March 5, 2022, from https://www.dicoding.com/blog/graphql-api-vs-rest-api-apa-bedanya/

Gupta, REST API Tutorial. (2022). What is REST?. Retrieved October 22, 2022, from https://restfulapi.net/

Quiña-Mera, A., Fernandez, P., García, J. M., & Ruiz-Cortés, A. (2022). GraphQL: A Systematic Mapping Study. *ACM Computing Surveys*. https://doi.org/10.1145/3561818

Ragam Peran dan Tugas Pengguna di Aplikasi SISTER. (n.d.). Retrieved October 20, 2022, from https://blog.ecampuz.com/ragam-peran-dan-tugas-pengguna-di-aplikasi-sister/

Request Lifecycle. (n.d). Retrieved October 21, 2022, from https://laravel.com/docs/9.x/*Lifecycle*

Request Lifecycle Lighthouse. (n.d). Retrieved October 21, 2022, from https://lighthouse-php.com/5/concepts/*request-Lifecycle*.html#*request-Lifecycle*

REST API - N+1 *Request*. (n.d). Retrieved October 22, 2022 from https://restfulapi.net/rest-api-n-1-problem/

Rizvandi, N. B., Lee, Y.-C., Zomaya, A., Choon, Y., Zomaya, A. Y., & Lee, Y. C. (2014). *Preliminary Results on Modeling CPU Utilization of MapReduce Programs Routing Protocol for Wireless Sensor Networks View project Localization Techniques using Sensor Networks in IoT Infrastructure View project SCHOOL OF INFORMATION TECHNOLOGIES PRELIMINARY RESULTS ON MODELING CPU UTILIZATION OF MAPREDUCE PROGRAMS TECHNICAL REPORT 665 Preliminary Results on Modeling CPU Utilization of MapReduce Programs*. https://www.researchgate.net/publication/228409939

Shi, M., Liu, J., Zhou, D., Tang, M., & Cao, B. (2017). WE-LDA: A Word Embeddings Augmented LDA Model for Web *Services* Clustering. *Proceedings - 2017 IEEE 24th International Conference on Web Services, ICWS 2017*, 9–16. https://doi.org/10.1109/ICWS.2017.9.

Subecz, Z. (2021). *Web-development with Laravel framework*. *Gradus*, *8*(1), 211–218. https://doi.org/10.47833/2021.1.csc.006

Sultanov, M. M., Gorban, Y. A., Smirnov, A. A., & Yurov, V. A. (2021, March 11). Development of a centralized system for data storage and processing on operation modes and reliability indicators of power equipment. *Proceedings of the 3rd 2021 International Youth Conference on Radio Electronics, Electrical and Power Engineering, REEPE 2021*. https://doi.org/10.1109/REEPE51337.2021.9388022

Vogel, M., Weber, S., & Zirpins, C. (2018). Experiences on Migrating RESTful Web *Services* to GraphQL. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10797 LNCS*, 283–295. https://doi.org/10.1007/978-3-319-91764-1_23

What is Amazon EC2? - Amazon Elastic Compute Cloud. (n.d.). Retrieved October 22, 2022, from https://docs.aws.*Amazon*.com/AWSEC2/latest/UserGuide/concepts.html

What is throughput? - Definition from WhatIs.com. (n.d.). SearchNetworking. Retrieved November 30, 2022, from https://www.techtarget.com/searchnetworking/definition/throughput

LAMPIRAN

Lampiran 1 Hasil Pengukuran *Response Time Concurrent Request*

Fetch Data Sumber Daya Manusia

Sample	*Response Time (milliseconds)*							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	143141	99364	149038	187083	109228	132068	164854	178959
2	123935	123867	158635	154995	111222	135200	170191	183062
3	83933	147329	148414	176434	120370	131174	168979	184311
4	109840	115166	134984	172240	123409	140045	168525	251378
5	109154	110964	151120	182088	118989	135832	168902	222715
Average	**114001**	**119338**	**148438**	**174568**	**116644**	**134864**	**168290**	**204085**

Fetch Data Pendidikan Formal SDM

Sample	*Response Time (milliseconds)*							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	3028	3312	3808	4293	4276	5266	5891	6830
2	2861	3213	3834	4135	4552	5182	5947	6892
3	2765	3257	3768	4197	4424	5110	6020	6844
4	2841	3340	3909	4532	4551	5165	5964	6918
5	2802	3361	3846	4206	4805	5158	6072	6971
Average	**2859,4**	**3296,6**	**3833**	**4272,6**	**4521,6**	**5176,2**	**5978,8**	**6891**

Fetch Data Dokumen SDM

Sample	*Response Time (milliseconds)*							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	3617	4329	4731	5055	6304	7371	8861	9594
2	3479	3962	5105	5082	6281	7783	9905	9713
3	3606	4040	4479	5138	6391	7578	8706	9586
4	3441	4621	4690	5116	6385	7471	8681	10086
5	3827	4067	4591	5039	6406	7478	8570	9720
Average	**3594**	**4203,8**	**4719,2**	**5086**	**6353,4**	**7536,2**	**8944,6**	**9739,8**

Upload Data Dokumen SDM

Sample	*Response Time (milliseconds)*							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	80294	97837	111144	123711	78333	94337	108936	120683
2	80287	99112	105873	131389	76880	94414	108706	124702
3	79698	96833	111008	129240	77558	91673	109270	121566
4	77462	96661	108160	127728	79672	91828	106517	121443
5	80110	95012	108284	125706	78280	93417	109274	118337
Average	**79570,2**	**97091**	**108894**	**127555**	**78144,6**	**93133,8**	**108541**	**121346**

Lampiran 2 Hasil Pengukuran *Throughput Concurrent Request*

Fetch Data Sumber Daya Manusia

Sample	*Throughput* (bits/min)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	22	37,5	28,3	23,7	28	26,8	27,8	31,9
2	22,6	29,7	29,4	32	26,9	26,9	27,8	31,8
3	28,8	25,9	27,9	25,2	25,4	27,4	27,8	31,8
4	26,7	33,6	31,7	28,1	24,9	26,8	27,9	19,1
5	28,1	29,9	28,3	26,6	25,9	27,1	27,9	21,8
Average	**25,64**	**31,32**	**29,12**	**27,12**	**26,22**	**27**	**27,84**	**27,28**

Fetch Data Pendidikan Formal SDM

Sample	*Throughput* (*bits*/sec)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	18,9	20	20,1	20	12,3	12	12,4	12,3
2	19,4	20,1	19,6	20,5	11,5	12,1	12,3	12,2
3	19,9	18,7	20	20,2	12	12,4	12,2	12,1
4	19,4	19,8	19,3	19,2	11,6	12,2	12,2	11,9
5	19,7	19,8	18,5	20,4	10,9	12,2	12,1	12,2
Average	**19,46**	**19,68**	**19,5**	**20,06**	**11,66**	**12,18**	**12,24**	**12,14**

Fetch Data Dokumen SDM

Sample	*Throughput* (*bits*/sec)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	15,7	16,3	16,3	17,1	8.1	8,1	8,3	8,5
2	16,1	16,6	15,5	17	8.4	8,2	6,8	8,6
3	16,1	16,5	16,9	17	7.9	8,4	8,5	8,7
4	15,9	14,2	15,9	14,9	8.4	8,6	8,5	8,5
5	14,2	16,4	16,7	17	8.4	8,5	8,6	8,6
Average	**15,6**	**16**	**16,26**	**16,6**	**8.24**	**8,36**	**8,14**	**8,58**

Upload Data Dokumen SDM

Sample	*Throughput* (*bits*/min)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	43,7	37,1	47,5	54,3	34,6	31,5	41,6	48,5
2	40,5	32,4	52,5	54	52,6	43,6	47,4	30
3	47,7	57,9	46,1	36,3	45,2	51,9	29,8	34,5
4	49,2	55,4	33,1	36,6	44,2	45,5	45,6	34,2
5	46,9	28,1	34,9	39,3	56,1	52,3	43,7	38,4
Average	**45,6**	**42,18**	**42,82**	**44,1**	**46,54**	**44,96**	**41,62**	**37,12**

Lampiran 3 Hasil Pengukuran CPU Utilization Concurrent Request

Fetch Data Sumber Daya Manusia

Sample	*CPU Utilization* (%)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	9,40769	11,0818	10,7083	10,7083	84,3194	84,4137	82,8787	79,4893
2	12,0083	12,3889	12,2273	12,2273	84,0033	83,9512	73,0204	82,9009
3	12,6118	11,0714	10,1	10,1	87,8058	83,6309	82,785	81,6314
4	11,225	10,7727	9,25556	9,25556	89,4163	87,82	85,8255	82,47
5	10,9444	10,9444	10,35	10,35	88,5598	83,1101	80,7427	85,6461
Average	**11,2394**	**11,2519**	**10,5282**	**10,5282**	**86,8209**	**84,5852**	**81,0505**	**82,4275**

Fetch Data Pendidikan Formal SDM

Sample	*CPU Utilization* (%)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	30	26,7	31,8	27,375	42,6667	39,76	39,08	36,6
2	25,85	28,1	27,7667	27,8	45,175	41,52	41,78	42,1167
3	22,2333	34,35	31,2333	25,4	38,75	38,68	36,4667	40,75
4	29,15	19,2	31,1	31,25	40,5667	37,075	40,26	44,1667
5	33,3	17,3667	31,9	30,4	38,3	40,425	42,28	39,1
Average	**28,1067**	**25,1433**	**30,76**	**28,445**	**41,0917**	**39,492**	**39,9733**	**40,5467**

Fetch Data Dokumen SDM

Sample	*CPU Utilization* (%)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	26,65	17,58	21,6667	17,92	46,1667	53,3333	49	50,425
2	29,15	25,5667	24,475	26,975	55,76	50,9	41,5222	50,3625
3	22,5	28,2	26,85	24,6	48,16	46,6167	52,8125	52,7
4	26,7	17,6333	26,05	24,725	51,24	52,15	51,7714	46,5667
5	19,4667	31,2	28,35	26,275	56,98	52,7	51,4143	49,9875
Average	**24,8933**	**24,036**	**25,4783**	**24,099**	**51,6613**	**51,14**	**49,3041**	**50,0083**

Upload Data Dokumen SDM

Sample	*CPU Utilization* (%)							
	REST				GraphQL			
	100	120	140	160	100	120	140	160
1	7,33448	10,7542	10,5	8,8383	17,0267	10,8478	10,9793	10,3432
2	10,775	10,3091	10,3091	9,73889	16,9045	14,8739	11,2107	15,1049
3	8,27368	9,53846	9,41481	9,80227	17,24	14,728	14,2897	13,3027
4	8,82273	8,91538	9,07778	11,5767	14,5737	16,8037	12,7469	12,393

5	10,3667	10,0227	9,86667	9,08421	15,9	11,904	10,2237	13,9316
Average	**9,11451**	**9,90797**	**9,83367**	**9,80807**	**16,329**	**13,8315**	**11,8901**	**13,0151**

Lampiran 4 Hasil Pengukuran *Response Time* Interval 5 menit

Fetch Data Sumber Daya Manusia 100 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:03	2787	4175	02:33	2816	27006
00:06	2829	4140	02:36	2836	26415
00:09	2774	4208	02:39	2804	25790
00:12	3085	4079	02:42	2838	25088
00:15	2842	10482	02:45	2777	24485
00:18	2731	9124	02:48	2798	23804
00:21	2741	8551	02:51	2831	23437
00:24	2744	7883	02:54	2881	23043
00:27	2794	12416	02:57	2822	22405
00:30	2741	11848	03:00	2865	23731
00:33	2785	15405	03:03	2879	22915
00:36	2739	12223	03:06	2826	22257
00:39	2728	11634	03:09	2822	23367
00:42	2810	11210	03:12	2885	26782
00:45	2811	15645	03:15	2777	22909
00:48	2769	14433	03:18	2814	23268
00:51	2742	13812	03:21	2829	23119
00:54	2812	14724	03:24	2876	23014
00:57	2803	14018	03:27	2772	23393
01:00	2832	13479	03:30	2767	29053
01:03	2779	12874	03:33	2773	27920
01:06	2744	20467	03:36	2733	27788
01:09	2778	18088	03:39	2780	32223
01:12	2742	17704	03:42	2829	31225
01:15	2797	17066	03:45	2796	36168
01:18	2783	16438	03:48	2823	33182
01:21	2839	15816	03:51	2787	32824
01:24	2782	15197	03:54	2774	32231

01:27	2776	14582	03:57	2774	31817
01:30	2747	14380	04:00	2823	32690
01:33	2972	16966	04:03	2872	32570
01:36	2780	16304	04:06	2843	31887
01:39	2737	15948	04:09	2819	31543
01:42	2745	15287	04:12	2922	34659
01:45	2734	21313	04:15	2822	38336
01:48	2736	24192	04:18	2775	37989
01:51	2778	21814	04:21	2790	37378
01:54	2838	21160	04:24	2744	36802
01:57	6119	20532	04:27	2812	36140
02:00	4900	19828	04:30	2773	35521
02:03	2940	26162	04:33	2775	35396
02:06	2782	25559	04:36	2865	35785
02:09	2790	24901	04:39	2837	41855
02:12	2976	24504	04:42	2782	40000
02:15	2829	31003	04:45	2845	39404
02:18	2806	28919	04:48	2767	38972
02:21	2838	28529	04:51	2786	39088
02:24	2814	28138	04:54	2769	38448
02:27	2783	27529	04:57	2774	37797
02:30	2828	27182	05:00	2766	37226

Fetch Data Sumber Daya Manusia 120 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:50	4997	4191	02:32:50	3628	39792
00:05:00	8444	4110	02:35:00	6319	45680
00:07:50	5774	4777	02:37:50	5599	50206
00:10:00	4404	5589	02:40:00	8547	49243
00:12:50	2953	5503	02:42:50	4672	49134
00:15:00	10469	5347	02:45:00	4160	54535
00:17:50	2886	5449	02:47:50	3250	56485
00:20:00	2785	5305	02:50:00	2768	56054
00:22:50	2772	5176	02:52:05	2786	59362
00:25:00	2734	9683	02:55:00	2872	59235
00:27:50	3031	9092	02:57:50	5865	59105
00:30:00	2749	12263	03:00:00	8450	58934

00:32:50	3011	14528	03:02:50	4386	58769
00:35:00	2807	12180	03:05:00	3667	58623
00:37:50	2776	12090	03:07:50	3201	62689
00:40:00	2781	12225	03:10:00	2776	61628
00:42:50	3031	12210	03:12:50	2881	65426
00:45:00	2830	12320	03:15:00	2782	64315
00:47:50	2788	12264	03:17:50	2723	69883
00:50:00	2783	12157	03:20:00	2829	69455
00:52:05	3080	12003	03:22:50	2741	69374
00:55:00	2776	11914	03:25:00	8790	69506
00:57:50	5814	11757	03:27:50	3811	69721
01:00:00	8498	17347	03:30:00	2880	69812
01:02:50	4320	17416	03:32:50	4010	70677
01:05:00	2912	17336	03:35:00	9054	70808
01:07:50	2837	17180	03:37:50	2772	70664
01:10:00	2792	16995	03:40:00	9013	72579
01:12:50	3115	18794	03:42:50	2982	77201
01:15:00	2743	18388	03:45:00	2736	77990
01:17:50	3019	18321	03:47:50	5557	75737
01:20:00	2775	18207	03:50:00	8442	80170
01:22:50	3024	24568	03:52:05	4095	79295
01:25:00	2778	22961	03:55:00	2878	79888
01:27:50	6727	23577	03:57:50	2836	79761
01:30:00	4545	23870	04:00:00	2773	79627
01:32:50	4118	24462	04:02:50	2832	84809
01:35:00	3192	24631	04:05:00	2773	82344
01:37:50	3076	24572	04:07:50	4497	82312
01:40:00	2741	24419	04:10:00	8075	82137
01:42:50	2979	26917	04:12:50	3319	82732
01:45:00	2775	27883	04:15:00	2785	82700
01:47:50	4577	32031	04:17:50	2719	82552
01:50:00	8177	27741	04:20:00	2782	82862
01:52:05	3845	28037	04:22:50	3013	82722
01:55:00	2824	27978	04:25:00	2788	82630
01:57:50	8819	28408	04:27:50	2770	82775
02:00:00	10498	29012	04:30:00	2821	86918
02:02:50	14704	28824	04:32:50	3027	86806
02:05:00	12760	32955	04:35:00	2736	86927
02:07:50	11309	32112	04:37:50	3074	86804

02:10:00	10140	31991	04:40:00	2776	86639
02:12:50	8659	32061	04:42:50	2783	87234
02:15:00	7660	32436	04:45:00	2878	88115
02:17:50	6257	32305	04:47:50	3056	88243
02:20:00	4802	38251	04:50:00	2771	88337
02:22:50	3377	36538	04:52:05	2808	88166
02:25:00	2767	40956	04:55:00	2808	88094
02:27:50	2972	39327	04:57:50	8185	87948
02:30:00	2742	39957	05:00:00	2780	87828

Fetch Data Sumber Daya Manusia 140 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:10	2781	5524	02:29:10	2808	66230
00:04:20	2866	6116	02:31:20	2771	66576
00:06:30	2801	6407	02:33:30	3067	66687
00:08:40	2826	6888	02:35:40	2962	71765
00:10:50	2810	7548	02:37:50	2787	70936
00:13:00	2741	8131	02:40:00	2744	72495
00:15:10	2785	8623	02:42:10	2833	72907
00:17:20	2724	9077	02:44:20	2742	75394
00:19:30	2812	10198	02:46:30	2791	79418
00:21:00	5672	10374	02:48:00	2770	75620
00:23:10	7971	10511	02:50:10	3998	76560
00:25:20	5574	10634	02:52:20	3397	76848
00:27:30	5252	11265	02:54:30	2820	77793
00:29:40	6037	11384	02:56:40	2776	79999
00:31:50	4724	11529	02:58:50	2821	84034
00:34:00	3662	11888	03:01:00	2785	80096
00:37:10	2807	14599	03:03:10	4335	80551
00:38:20	2821	14864	03:05:20	3971	81723
00:40:30	2819	15335	03:07:30	9456	81978
00:42:00	2779	15645	03:09:00	10455	82303
00:44:10	2790	16027	03:12:10	9397	82544
00:46:20	2807	16293	03:13:20	8275	82979
00:48:30	2743	16404	03:15:30	7535	83300
00:50:40	2803	16837	03:17:40	6402	83501
00:52:50	5181	17401	03:19:50	5328	83835

00:55:00	4809	17473	03:22:00	8501	84225
00:57:10	4480	17737	03:24:10	6634	85938
00:59:20	3374	26281	03:26:20	6493	86854
01:01:30	2774	24386	03:28:30	6139	87130
01:03:00	2793	24430	03:30:00	5761	87295
01:05:10	5989	24602	03:32:10	5912	87950
01:07:20	3940	25234	03:34:20	5912	87769
01:09:30	3827	25484	03:36:30	5328	88240
01:11:40	2820	32398	03:38:40	4262	88796
01:13:50	2783	31049	03:40:50	3398	89461
01:16:00	2792	31316	03:43:00	3078	89619
01:18:10	2786	35336	03:45:10	2771	89803
01:20:20	3120	38245	03:47:20	2802	89953
01:22:30	3562	35566	03:49:30	2976	90317
01:24:00	3243	36117	03:51:00	2779	91390
01:26:10	2795	36300	03:53:10	2778	91985
01:28:20	2824	42561	03:55:20	2781	91912
01:30:30	2728	42845	03:57:30	2724	92333
01:32:40	2838	43100	03:59:40	2788	92496
01:34:50	3422	46869	04:01:50	3049	92674
01:37:00	2764	46630	04:04:00	6119	93040
01:39:10	8320	46959	04:06:10	5251	93148
01:41:20	2741	55253	04:08:20	4143	95137
01:43:30	2813	53608	04:10:30	3041	98566
01:45:00	2824	53519	04:12:00	2822	95394
01:47:10	2765	53961	04:14:10	2789	96189
01:49:20	4013	54416	04:16:20	2745	97219
01:51:30	2878	57942	04:18:30	4909	97405
01:53:40	7866	57335	04:20:40	3442	97632
01:55:50	2768	57662	04:22:50	2779	103467
01:59:00	2834	57664	04:25:00	2783	102325
02:00:10	2829	58199	04:27:10	2829	102531
02:02:20	2773	58473	04:29:20	2731	102977
02:04:30	2736	59687	04:31:30	2730	103206
02:06:00	2733	61876	04:33:00	2769	103311
02:08:10	2738	60255	04:35:10	2780	103703
02:11:20	2725	60366	04:37:20	2769	104433
02:12:30	2775	60630	04:39:30	2782	105009
02:14:40	2818	60922	04:41:40	2722	105816

02:16:50	2821	61245	04:43:50	2771	106434
02:19:00	2844	61364	04:46:00	3058	106541
02:21:10	3262	62687	04:47:10	2956	106647
02:23:20	2839	65054	04:50:20	2731	106915
02:25:30	2821	65749	04:52:30	2822	107137
02:27:00	2837	66073	04:54:00	2774	107749

Fetch Data Sumber Daya Manusia 160 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:20	3527	4076	02:26:20	2826	109758
00:04:00	3445	4550	02:28:00	2976	110531
00:05:40	2796	5831	02:29:40	2736	111077
00:07:20	2746	9019	02:31:20	2783	111523
00:09:00	2803	11800	02:33:00	2779	113047
00:11:20	2776	15583	02:35:20	2779	114306
00:13:00	2774	12904	02:37:00	2827	114817
00:14:40	2730	15284	02:38:40	2781	115282
00:16:20	2727	16294	02:40:20	2785	116042
00:18:00	2736	16768	02:42:00	2778	116778
00:20:20	2742	17264	02:44:20	2724	118082
00:22:00	2819	17762	02:46:00	2741	118517
00:23:40	2776	25289	02:47:40	2786	119275
00:25:20	2829	30365	02:49:20	2780	119786
00:27:00	2765	30722	02:51:00	2769	125436
00:29:20	2813	31609	02:53:20	2776	125264
00:31:00	2977	32096	02:55:00	2775	125685
00:32:40	8639	32619	02:56:40	3034	126612
00:34:20	7619	33051	02:58:20	4626	127171
00:36:00	6964	33868	03:00:00	3831	133244
00:38:20	6186	34567	03:02:20	2982	133165
00:40:00	5547	35363	03:04:00	8657	133690
00:41:40	4702	35847	03:05:40	2732	134219
00:43:20	4573	36824	03:07:20	2827	134746
00:45:00	4824	38261	03:09:00	3918	135582
00:47:20	3974	39736	03:11:20	3398	136865
00:49:00	3083	40232	03:13:00	2770	137329
00:50:40	2825	42601	03:14:40	9039	139150

00:52:20	2766	42589	03:16:20	2809	140084
00:54:00	2835	43094	03:18:00	2804	141323
00:56:20	2775	43596	03:20:20	2772	141821
00:58:00	2787	44362	03:22:00	2771	142517
00:59:40	4882	45605	03:23:40	2813	143014
01:01:20	8334	46081	03:25:20	3077	143625
01:03:00	5084	48623	03:27:00	6096	144058
01:05:20	4225	51141	03:29:20	4827	144587
01:07:00	3427	49135	03:31:00	4008	148421
01:08:40	2878	49616	03:32:40	3151	148732
01:10:20	2734	50357	03:34:20	2818	149432
01:12:00	2768	55931	03:36:00	2734	155708
01:14:20	3012	57253	03:38:20	7211	155457
01:16:00	2930	63180	03:40:00	9339	156721
01:17:40	2781	63370	03:41:40	13702	157479
01:19:20	2831	63978	03:43:20	12852	164284
01:21:00	2784	64463	03:45:00	12280	163460
01:23:20	2785	67977	03:47:20	14173	164264
01:25:00	2783	70668	03:49:00	13634	165753
01:26:40	2739	68423	03:50:40	12784	166482
01:28:20	2731	69158	03:52:20	12012	168029
01:30:00	2987	69676	03:54:00	11395	168459
01:32:20	4624	70383	03:56:20	10520	169016
01:34:00	3774	71378	03:58:00	10629	169593
01:35:40	2944	71883	03:59:40	10391	174816
01:37:20	2795	72614	04:01:20	15640	174340
01:39:00	2816	75629	04:03:00	16553	174952
01:41:20	2781	79326	04:05:20	14575	175668
01:43:00	8333	76876	04:07:00	17371	176172
01:44:40	7352	77425	04:08:40	14079	176843
01:46:20	6715	84777	04:10:20	16404	177448
01:48:00	5920	84029	04:12:00	17770	177825
01:50:20	5363	84589	04:14:20	20353	178338
01:52:00	5323	84994	04:16:00	17418	178910
01:53:40	5486	85489	04:17:40	16580	180256
01:55:20	4859	89036	04:19:20	15802	181452
01:57:00	4146	92505	04:21:00	16743	181945
01:59:20	3312	89774	04:23:20	21562	187030
02:01:00	2770	91824	04:25:00	19765	186294

02:02:40	2825	93797	04:26:40	19209	187036
02:04:20	2782	95011	04:28:20	19122	187490
02:06:00	2825	95501	04:30:00	18557	188010
02:08:20	2869	96220	04:32:20	20086	188775
02:10:00	2809	97460	04:34:00	23518	189477
02:11:40	2768	97994	04:35:40	24500	189927
02:13:20	3070	98725	04:37:20	23215	193809
02:15:00	3174	99247	04:39:00	22362	193138
02:17:20	3376	100017	04:41:20	21522	193588
02:19:00	2823	101574	04:43:00	20741	194558
02:20:40	2884	108335	04:44:40	19874	195050
05:22:20	2829	108529	04:46:20	19083	201425
02:24:00	2815	109312	04:48:00	23888	201468

Fetch Data Pendidikan Formal SDM 100 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:03	553	601	02:33	553	595
00:06	604	609	02:36	544	636
00:09	576	594	02:39	553	597
00:12	539	597	02:42	546	641
00:15	573	601	02:45	551	637
00:18	562	589	02:48	606	604
00:21	546	578	02:51	560	588
00:24	612	577	02:54	553	639
00:27	570	590	02:57	541	621
00:30	549	597	03:00	549	585
00:33	579	587	03:03	550	592
00:36	537	586	03:06	554	588
00:39	583	620	03:09	534	609
00:42	631	589	03:12	548	598
00:45	587	631	03:15	773	601
00:48	630	592	03:18	609	655
00:51	578	618	03:21	577	590
00:54	559	587	03:24	571	600
00:57	552	627	03:27	625	589
01:00	571	595	03:30	544	600
01:03	647	581	03:33	553	589

01:06	543	682	03:36	545	604
01:09	585	591	03:39	550	625
01:12	553	586	03:42	555	586
01:15	554	599	03:45	537	589
01:18	551	602	03:48	554	582
01:21	552	590	03:51	581	587
01:24	597	579	03:54	545	586
01:27	585	607	03:57	542	594
01:30	553	593	04:00	592	599
01:33	545	597	04:03	554	629
01:36	554	619	04:06	535	597
01:39	546	599	04:09	535	594
01:42	553	580	04:12	537	585
01:45	535	598	04:15	542	592
01:48	545	578	04:18	575	593
01:51	545	594	04:21	541	598
01:54	543	593	04:24	604	588
01:57	541	589	04:27	539	589
02:00	597	585	04:30	596	590
02:03	555	587	04:33	538	587
02:06	552	595	04:36	542	587
02:09	597	595	04:39	609	604
02:12	545	587	04:42	546	591
02:15	551	670	04:45	561	592
02:18	550	576	04:48	589	581
02:21	582	600	04:51	549	589
02:24	561	636	04:54	541	585
02:27	544	587	04:57	548	588
02:30	542	592	05:00	599	596

Fetch Data Pendidikan Formal SDM 120 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:50	649	641	02:32:50	617	643
00:05:00	686	635	02:35:00	585	668
00:07:50	635	621	02:37:50	797	626
00:10:00	601	644	02:40:00	673	692
00:12:50	601	618	02:42:50	653	619

00:15:00	593	628	02:45:00	658	643
00:17:50	589	623	02:47:50	741	653
00:20:00	592	659	02:50:00	581	645
00:22:50	609	626	02:52:05	842	617
00:25:00	600	639	02:55:00	585	635
00:27:50	612	630	02:57:50	602	637
00:30:00	606	635	03:00:00	589	637
00:32:50	599	641	03:02:50	603	629
00:35:00	649	638	03:05:00	596	687
00:37:50	636	649	03:07:50	643	629
00:40:00	611	639	03:10:00	605	695
00:42:50	599	626	03:12:50	633	633
00:45:00	622	650	03:15:00	653	633
00:47:50	619	630	03:17:50	628	623
00:50:00	590	625	03:20:00	617	633
00:52:05	603	633	03:22:50	690	634
00:55:00	611	680	03:25:00	592	673
00:57:50	637	645	03:27:50	653	631
01:00:00	618	645	03:30:00	590	667
01:02:50	608	619	03:32:50	595	643
01:05:00	629	645	03:35:00	627	687
01:07:50	602	628	03:37:50	636	648
01:10:00	631	649	03:40:00	653	639
01:12:50	601	639	03:42:50	598	621
01:15:00	587	645	03:45:00	596	643
01:17:50	626	622	03:47:50	602	700
01:20:00	593	645	03:50:00	587	649
01:22:50	605	705	03:52:05	645	719
01:25:00	601	624	03:55:00	599	677
01:27:50	646	631	03:57:50	601	622
01:30:00	619	632	04:00:00	583	630
01:32:50	663	677	04:02:50	624	651
01:35:00	633	633	04:05:00	705	634
01:37:50	636	628	04:07:50	615	631
01:40:00	590	644	04:10:00	636	757
01:42:50	633	624	04:12:50	641	655
01:45:00	588	643	04:15:00	600	652
01:47:50	663	641	04:17:50	604	674
01:50:00	608	644	04:20:00	601	646

01:52:05	600	633	04:22:50	617	626
01:55:00	587	653	04:25:00	598	635
01:57:50	597	639	04:27:50	605	622
02:00:00	592	642	04:30:00	617	626
02:02:50	611	624	04:32:50	612	631
02:05:00	603	647	04:35:00	642	644
02:07:50	611	621	04:37:50	636	623
02:10:00	662	679	04:40:00	636	657
02:12:50	598	624	04:42:50	656	682
02:15:00	593	657	04:45:00	598	639
02:17:50	598	640	04:47:50	598	630
02:20:00	587	638	04:50:00	621	632
02:22:50	608	664	04:52:05	608	640
02:25:00	688	644	04:55:00	646	641
02:27:50	609	631	04:57:50	597	648
02:30:00	650	637	05:00:00	584	644

Fetch Data Pendidikan Formal SDM 140 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:10	565	622	02:29:10	570	586
00:04:20	542	610	02:31:20	550	590
00:06:30	555	582	02:33:30	545	575
00:08:40	561	598	02:35:40	591	581
00:10:50	549	694	02:37:50	552	593
00:13:00	549	689	02:40:00	638	577
00:15:10	551	577	02:42:10	549	602
00:17:20	604	585	02:44:20	575	587
00:19:30	557	576	02:46:30	558	625
00:21:00	553	583	02:48:00	595	628
00:23:10	552	586	02:50:10	555	598
00:25:20	576	578	02:52:20	554	616
00:27:30	563	605	02:54:30	560	601
00:29:40	547	618	02:56:40	563	594
00:31:50	597	623	02:58:50	554	579
00:34:00	566	581	03:01:00	559	615
00:37:10	561	608	03:03:10	549	588
00:38:20	556	573	03:05:20	640	593

00:40:30	558	589	03:07:30	554	601
00:42:00	601	592	03:09:00	541	684
00:44:10	547	582	03:12:10	616	603
00:46:20	563	583	03:13:20	549	645
00:48:30	552	598	03:15:30	557	577
00:50:40	542	584	03:17:40	571	646
00:52:50	560	587	03:19:50	550	583
00:55:00	547	648	03:22:00	687	578
00:57:10	565	577	03:24:10	557	583
00:59:20	560	587	03:26:20	548	578
01:01:30	554	588	03:28:30	564	618
01:03:00	559	593	03:30:00	539	617
01:05:10	557	614	03:32:10	577	596
01:07:20	548	592	03:34:20	541	598
01:09:30	574	619	03:36:30	554	595
01:11:40	576	596	03:38:40	634	574
01:13:50	555	601	03:40:50	574	581
01:16:00	552	597	03:43:00	545	581
01:18:10	545	614	03:45:10	543	579
01:20:20	550	719	03:47:20	556	591
01:22:30	560	595	03:49:30	559	638
01:24:00	541	580	03:51:00	564	644
01:26:10	615	578	03:53:10	546	594
01:28:20	562	584	03:55:20	552	594
01:30:30	550	586	03:57:30	567	593
01:32:40	554	593	03:59:40	549	591
01:34:50	542	582	04:01:50	563	578
01:37:00	549	579	04:04:00	566	589
01:39:10	592	626	04:06:10	592	597
01:41:20	549	586	04:08:20	550	595
01:43:30	613	586	04:10:30	541	593
01:45:00	539	624	04:12:00	542	577
01:47:10	557	599	04:14:10	551	580
01:49:20	551	609	04:16:20	558	594
01:51:30	557	583	04:18:30	621	599
01:53:40	551	589	04:20:40	552	587
01:55:50	554	579	04:22:50	549	669
01:59:00	546	597	04:25:00	568	585
02:00:10	554	586	04:27:10	577	583

02:02:20	628	665	04:29:20	593	613
02:04:30	556	594	04:31:30	545	651
02:06:00	543	581	04:33:00	551	630
02:08:10	558	578	04:35:10	599	613
02:11:20	560	633	04:37:20	551	620
02:12:30	545	592	04:39:30	722	592
02:14:40	579	587	04:41:40	554	586
02:16:50	547	575	04:43:50	555	583
02:19:00	588	604	04:46:00	549	606
02:21:10	568	605	04:47:10	668	595
02:23:20	553	581	04:50:20	552	596
02:25:30	549	585	04:52:30	631	576
02:27:00	549	593	04:54:00	545	583

Fetch Data Pendidikan Formal SDM 160 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:20	552	649	02:26:20	546	674
00:04:00	542	576	02:28:00	565	585
00:05:40	582	590	02:29:40	553	577
00:07:20	632	578	02:31:20	553	580
00:09:00	571	571	02:33:00	557	584
00:11:20	551	658	02:35:20	568	592
00:13:00	546	588	02:37:00	554	581
00:14:40	567	576	02:38:40	568	585
00:16:20	554	577	02:40:20	553	587
00:18:00	604	592	02:42:00	596	575
00:20:20	554	591	02:44:20	618	596
00:22:00	548	570	02:46:00	560	586
00:23:40	569	575	02:47:40	550	573
00:25:20	589	581	02:49:20	593	585
00:27:00	547	587	02:51:00	536	656
00:29:20	553	578	02:53:20	636	576
00:31:00	550	583	02:55:00	568	595
00:32:40	543	598	02:56:40	559	583
00:34:20	556	591	02:58:20	562	608
00:36:00	597	595	03:00:00	563	581
00:38:20	559	587	03:02:20	659	585

00:40:00	563	601	03:04:00	542	583
00:41:40	603	578	03:05:40	558	591
00:43:20	558	584	03:07:20	562	580
00:45:00	550	601	03:09:00	557	622
00:47:20	614	578	03:11:20	557	569
00:49:00	557	587	03:13:00	595	583
00:50:40	545	568	03:14:40	561	608
00:52:20	595	590	03:16:20	549	639
00:54:00	542	589	03:18:00	556	590
00:56:20	630	591	03:20:20	568	593
00:58:00	553	588	03:22:00	562	569
00:59:40	576	613	03:23:40	570	588
01:01:20	588	574	03:25:20	577	583
01:03:00	620	590	03:27:00	568	579
01:05:20	564	589	03:29:20	537	575
01:07:00	550	607	03:31:00	554	582
01:08:40	559	575	03:32:40	536	622
01:10:20	593	591	03:34:20	564	589
01:12:00	556	579	03:36:00	557	575
01:14:20	554	586	03:38:20	555	580
01:16:00	556	571	03:40:00	539	570
01:17:40	553	585	03:41:40	584	591
01:19:20	551	577	03:43:20	555	582
01:21:00	559	588	03:45:00	551	574
01:23:20	543	583	03:47:20	545	593
01:25:00	579	589	03:49:00	590	584
01:26:40	544	584	03:50:40	556	611
01:28:20	546	606	03:52:20	596	611
01:30:00	539	644	03:54:00	566	582
01:32:20	576	587	03:56:20	560	585
01:34:00	554	572	03:58:00	543	589
01:35:40	581	595	03:59:40	576	600
01:37:20	545	645	04:01:20	704	627
01:39:00	563	594	04:03:00	603	588
01:41:20	548	577	04:05:20	548	650
01:43:00	576	587	04:07:00	546	597
01:44:40	574	574	04:08:40	541	575
01:46:20	544	574	04:10:20	555	592
01:48:00	554	593	04:12:00	544	649

01:50:20	562	586	04:14:20	624	581
01:52:00	591	597	04:16:00	540	596
01:53:40	542	618	04:17:40	558	587
01:55:20	564	579	04:19:20	558	585
01:57:00	642	594	04:21:00	549	598
01:59:20	546	658	04:23:20	547	658
02:01:00	553	643	04:25:00	556	582
02:02:40	542	631	04:26:40	556	583
02:04:20	561	580	04:28:20	553	592
02:06:00	547	695	04:30:00	550	591
02:08:20	547	588	04:32:20	551	589
02:10:00	553	675	04:34:00	558	579
02:11:40	564	583	04:35:40	556	605
02:13:20	584	577	04:37:20	557	630
02:15:00	589	586	04:39:00	567	592
02:17:20	560	588	04:41:20	576	602
02:19:00	664	598	04:43:00	558	616
02:20:40	547	594	04:44:40	580	612
05:22:20	553	596	04:46:20	582	583
02:24:00	559	577	04:48:00	545	576

Fetch Data Dokumen SDM 100 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:03	882	3387	02:33	794	956
00:06	949	925	02:36	813	888
00:09	917	935	02:39	817	891
00:12	1180	1016	02:42	859	966
00:15	803	1011	02:45	800	2869
00:18	794	914	02:48	839	876
00:21	826	890	02:51	828	979
00:24	963	901	02:54	834	881
00:27	826	893	02:57	819	1136
00:30	795	1079	03:00	881	2013
00:33	829	946	03:03	1277	903
00:36	860	1017	03:06	820	880
00:39	917	2156	03:09	871	874
00:42	868	892	03:12	1091	857

00:45	843	874	03:15	857	874
00:48	825	871	03:18	951	875
00:51	788	968	03:21	859	868
00:54	837	995	03:24	931	1055
00:57	853	885	03:27	851	923
01:00	812	864	03:30	823	925
01:03	835	894	03:33	884	909
01:06	793	893	03:36	966	975
01:09	927	2566	03:39	896	863
01:12	831	1721	03:42	1021	974
01:15	823	2604	03:45	984	893
01:18	2415	875	03:48	841	874
01:21	3215	867	03:51	896	884
01:24	802	941	03:54	947	881
01:27	951	861	03:57	822	878
01:30	901	920	04:00	833	2718
01:33	821	953	04:03	912	880
01:36	833	865	04:06	796	880
01:39	839	993	04:09	812	2534
01:42	855	2532	04:12	1465	906
01:45	828	907	04:15	805	1037
01:48	806	889	04:18	1044	3005
01:51	816	3326	04:21	929	2051
01:54	892	873	04:24	896	918
01:57	819	1318	04:27	848	1133
02:00	842	946	04:30	827	2458
02:03	812	926	04:33	804	980
02:06	814	2558	04:36	797	936
02:09	798	2929	04:39	831	931
02:12	1005	910	04:42	884	2101
02:15	886	1006	04:45	947	2489
02:18	944	989	04:48	858	935
02:21	2331	943	04:51	844	866
02:24	832	2480	04:54	841	2768
02:27	809	871	04:57	883	934
02:30	859	898	05:00	835	870

Fetch Data Dokumen SDM 120 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:50	1932	877	02:32:50	897	849
00:05:00	907	952	02:35:00	930	958
00:07:50	800	861	02:37:50	836	877
00:10:00	958	961	02:40:00	793	903
00:12:50	833	856	02:42:50	858	902
00:15:00	813	883	02:45:00	971	876
00:17:50	865	882	02:47:50	827	847
00:20:00	816	880	02:50:00	808	923
00:22:50	878	1001	02:52:05	796	848
00:25:00	855	977	02:55:00	860	876
00:27:50	833	868	02:57:50	958	885
00:30:00	839	1000	03:00:00	820	900
00:32:50	841	963	03:02:50	855	902
00:35:00	806	954	03:05:00	821	903
00:37:50	868	890	03:07:50	865	920
00:40:00	885	902	03:10:00	798	904
00:42:50	889	915	03:12:50	861	961
00:45:00	836	1035	03:15:00	1068	936
00:47:50	862	982	03:17:50	806	852
00:50:00	1013	872	03:20:00	859	1124
00:52:05	838	925	03:22:50	824	2296
00:55:00	881	874	03:25:00	807	853
00:57:50	846	926	03:27:50	796	899
01:00:00	898	858	03:30:00	892	894
01:02:50	829	848	03:32:50	942	956
01:05:00	923	839	03:35:00	910	886
01:07:50	792	1040	03:37:50	905	2254
01:10:00	808	1071	03:40:00	861	902
01:12:50	808	912	03:42:50	808	964
01:15:00	799	955	03:45:00	878	916
01:17:50	794	854	03:47:50	800	849
01:20:00	976	903	03:50:00	900	895
01:22:50	814	1012	03:52:05	840	901
01:25:00	829	890	03:55:00	914	946
01:27:50	815	882	03:57:50	847	869

01:30:00	910	870	04:00:00	867	901
01:32:50	816	836	04:02:50	815	912
01:35:00	789	856	04:05:00	789	902
01:37:50	822	951	04:07:50	911	932
01:40:00	811	944	04:10:00	813	920
01:42:50	826	895	04:12:50	885	894
01:45:00	803	886	04:15:00	882	936
01:47:50	997	926	04:17:50	808	1071
01:50:00	840	896	04:20:00	899	952
01:52:05	938	926	04:22:50	842	927
01:55:00	827	965	04:25:00	822	969
01:57:50	787	863	04:27:50	831	851
02:00:00	907	877	04:30:00	828	869
02:02:50	837	978	04:32:50	990	883
02:05:00	807	961	04:35:00	810	845
02:07:50	993	869	04:37:50	1011	1161
02:10:00	812	854	04:40:00	833	998
02:12:50	787	862	04:42:50	903	850
02:15:00	849	879	04:45:00	821	871
02:17:50	809	874	04:47:50	811	932
02:20:00	817	1901	04:50:00	910	962
02:22:50	915	922	04:52:05	810	881
02:25:00	902	909	04:55:00	815	886
02:27:50	889	954	04:57:50	893	866
02:30:00	809	982	05:00:00	815	922

Fetch Data Dokumen SDM 140 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:10	873	1032	02:29:10	851	885
00:04:20	841	871	02:31:20	861	856
00:06:30	811	855	02:33:30	799	936
00:08:40	853	875	02:35:40	803	892
00:10:50	798	862	02:37:50	819	917
00:13:00	833	1172	02:40:00	808	920
00:15:10	828	875	02:42:10	813	860
00:17:20	800	926	02:44:20	841	876
00:19:30	893	875	02:46:30	877	900

00:21:00	913	926	02:48:00	827	885
00:23:10	787	890	02:50:10	814	943
00:25:20	796	935	02:52:20	804	916
00:27:30	787	931	02:54:30	925	913
00:29:40	799	989	02:56:40	800	925
00:31:50	878	926	02:58:50	887	917
00:34:00	884	864	03:01:00	812	924
00:37:10	815	934	03:03:10	798	876
00:38:20	799	965	03:05:20	860	889
00:40:30	1034	960	03:07:30	813	863
00:42:00	818	899	03:09:00	829	991
00:44:10	864	903	03:12:10	819	842
00:46:20	849	896	03:13:20	816	898
00:48:30	834	990	03:15:30	867	888
00:50:40	816	1014	03:17:40	811	865
00:52:50	834	1059	03:19:50	827	883
00:55:00	795	880	03:22:00	990	950
00:57:10	1048	881	03:24:10	821	935
00:59:20	809	999	03:26:20	915	862
01:01:30	906	871	03:28:30	804	866
01:03:00	837	844	03:30:00	810	925
01:05:10	950	867	03:32:10	881	858
01:07:20	818	991	03:34:20	875	997
01:09:30	813	904	03:36:30	789	837
01:11:40	826	890	03:38:40	813	958
01:13:50	793	860	03:40:50	880	855
01:16:00	881	872	03:43:00	887	883
01:18:10	820	976	03:45:10	808	907
01:20:20	861	847	03:47:20	841	1047
01:22:30	808	870	03:49:30	926	890
01:24:00	907	867	03:51:00	861	881
01:26:10	786	861	03:53:10	1015	876
01:28:20	808	908	03:55:20	862	971
01:30:30	804	877	03:57:30	817	901
01:32:40	820	852	03:59:40	844	1065
01:34:50	843	853	04:01:50	897	901
01:37:00	966	893	04:04:00	820	904
01:39:10	813	950	04:06:10	808	894
01:41:20	819	935	04:08:20	826	906

01:43:30	863	876	04:10:30	828	863
01:45:00	809	887	04:12:00	825	904
01:47:10	817	880	04:14:10	867	986
01:49:20	836	900	04:16:20	796	891
01:51:30	854	850	04:18:30	811	1000
01:53:40	789	900	04:20:40	819	882
01:55:50	795	893	04:22:50	930	1015
01:59:00	811	927	04:25:00	810	849
02:00:10	935	877	04:27:10	813	900
02:02:20	1048	863	04:29:20	819	856
02:04:30	922	881	04:31:30	804	872
02:06:00	920	975	04:33:00	826	929
02:08:10	822	943	04:35:10	902	904
02:11:20	837	875	04:37:20	816	866
02:12:30	835	839	04:39:30	825	917
02:14:40	847	933	04:41:40	987	923
02:16:50	906	879	04:43:50	833	941
02:19:00	825	1053	04:46:00	837	994
02:21:10	842	870	04:48:10	871	901
02:23:20	846	1041	04:50:20	914	891
02:25:30	816	852	04:52:30	916	871
02:27:00	790	898	04:54:00	798	874

Fetch Data Dokumen SDM 160 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:20	858	873	02:26:20	798	886
00:04:00	873	837	02:28:00	805	897
00:05:40	950	888	02:29:40	849	898
00:07:20	825	863	02:31:20	813	841
00:09:00	830	904	02:33:00	941	870
00:11:20	822	864	02:35:20	910	911
00:13:00	834	902	02:37:00	928	1010
00:14:40	815	873	02:38:40	809	948
00:16:20	907	919	02:40:20	806	861
00:18:00	828	915	02:42:00	838	953
00:20:20	828	864	02:44:20	816	932
00:22:00	844	858	02:46:00	864	964

00:23:40	816	887	02:47:40	828	979
00:25:20	833	843	02:49:20	846	837
00:27:00	919	946	02:51:00	825	967
00:29:20	795	868	02:53:20	847	892
00:31:00	804	974	02:55:00	791	908
00:32:40	852	959	02:56:40	859	850
00:34:20	812	884	02:58:20	851	893
00:36:00	804	855	03:00:00	808	911
00:38:20	856	940	03:02:20	830	1112
00:40:00	853	963	03:04:00	820	926
00:41:40	851	969	03:05:40	877	876
00:43:20	837	916	03:07:20	890	964
00:45:00	807	882	03:09:00	795	915
00:47:20	924	957	03:11:20	805	964
00:49:00	791	876	03:13:00	849	892
00:50:40	1151	897	03:14:40	860	890
00:52:20	841	874	03:16:20	889	883
00:54:00	813	883	03:18:00	827	935
00:56:20	828	885	03:20:20	937	968
00:58:00	849	937	03:22:00	837	891
00:59:40	839	862	03:23:40	846	865
01:01:20	883	866	03:25:20	809	867
01:03:00	823	1117	03:27:00	904	1022
01:05:20	829	873	03:29:20	833	842
01:07:00	833	1076	03:31:00	842	863
01:08:40	794	892	03:32:40	833	953
01:10:20	801	947	03:34:20	841	906
01:12:00	840	933	03:36:00	895	846
01:14:20	839	993	03:38:20	812	1033
01:16:00	879	877	03:40:00	889	888
01:17:40	829	911	03:41:40	819	888
01:19:20	776	855	03:43:20	787	921
01:21:00	884	911	03:45:00	851	873
01:23:20	803	927	03:47:20	933	973
01:25:00	928	849	03:49:00	840	1335
01:26:40	888	1027	03:50:40	806	876
01:28:20	810	863	03:52:20	855	961
01:30:00	795	861	03:54:00	943	912
01:32:20	840	925	03:56:20	861	1237

01:34:00	938	958	03:58:00	829	959
01:35:40	862	1001	03:59:40	827	982
01:37:20	881	894	04:01:20	788	956
01:39:00	825	1012	04:03:00	850	978
01:41:20	797	857	04:05:20	808	857
01:43:00	838	909	04:07:00	803	1132
01:44:40	823	886	04:08:40	841	917
01:46:20	832	940	04:10:20	851	981
01:48:00	787	920	04:12:00	815	2578
01:50:20	1128	879	04:14:20	844	896
01:52:00	820	872	04:16:00	865	957
01:53:40	822	854	04:17:40	874	1035
01:55:20	806	914	04:19:20	850	875
01:57:00	900	999	04:21:00	859	1018
01:59:20	816	1063	04:23:20	801	911
02:01:00	862	880	04:25:00	909	848
02:02:40	917	891	04:26:40	834	891
02:04:20	799	872	04:28:20	813	886
02:06:00	904	889	04:30:00	918	966
02:08:20	831	956	04:32:20	784	869
02:10:00	851	1160	04:34:00	825	999
02:11:40	863	873	04:35:40	790	930
02:13:20	826	936	04:37:20	832	941
02:15:00	835	898	04:39:00	847	921
02:17:20	800	944	04:41:20	931	846
02:19:00	837	871	04:43:00	866	904
02:20:40	798	1227	04:44:40	808	877
05:22:20	821	899	04:46:20	882	859
02:24:00	816	882	04:48:00	791	898

Upload Data Dokumen SDM 100 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:03	5158	5303	02:33	5128	4844
00:06	4957	5019	02:36	4909	4921
00:09	8135	5052	02:39	4912	5409
00:12	4999	5224	02:42	4956	5160
00:15	5232	9599	02:45	5008	5267

00:18	5188	5094	02:48	4959	5165
00:21	5112	5040	02:51	5030	5197
00:24	5172	5479	02:54	4649	4942
00:27	5381	5459	02:57	4980	5139
00:30	4994	5279	03:00	5166	4949
00:33	5076	4988	03:03	5231	5024
00:36	4940	5188	03:06	5057	4958
00:39	5061	4994	03:09	4744	5202
00:42	5401	5208	03:12	4682	5717
00:45	4753	5243	03:15	5594	5467
00:48	4805	5164	03:18	4968	4914
00:51	4849	5017	03:21	5030	5248
00:54	5073	5131	03:24	4751	5225
00:57	4749	5220	03:27	4809	5792
01:00	4810	4880	03:30	4838	4941
01:03	4938	5167	03:33	4981	5017
01:06	5289	5107	03:36	5128	5173
01:09	5134	5659	03:39	4931	5165
01:12	4898	5074	03:42	5211	5137
01:15	5165	4811	03:45	4818	4907
01:18	5016	4922	03:48	4834	5131
01:21	9192	5317	03:51	4867	4874
01:24	4903	5027	03:54	5077	5032
01:27	5078	5051	03:57	5043	5110
01:30	4901	5350	04:00	5288	4923
01:33	5101	5119	04:03	5259	5100
01:36	5710	5201	04:06	5069	5149
01:39	4848	5050	04:09	4965	5208
01:42	5161	4914	04:12	5105	4927
01:45	5185	5199	04:15	4862	5131
01:48	4885	5083	04:18	4963	4950
01:51	5218	5374	04:21	5297	4878
01:54	5175	5136	04:24	5178	4986
01:57	5092	5243	04:27	5051	5619
02:00	5216	5276	04:30	5200	5389
02:03	5519	5172	04:33	4897	5495
02:06	4982	4946	04:36	4971	5814
02:09	4943	5454	04:39	4952	5222
02:12	5249	5888	04:42	4924	5171

02:15	4976	4916	04:45	4897	4960
02:18	4942	5280	04:48	5240	4859
02:21	4978	5179	04:51	5150	5003
02:24	4892	4868	04:54	5235	5161
02:27	5042	5238	04:57	5376	5055
02:30	5156	5294	05:00	4897	5175

Upload Data Dokumen SDM 120 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:50	4856	4951	02:32:50	5108	5007
00:05:00	4897	4941	02:35:00	5007	5110
00:07:50	6963	5141	02:37:50	4938	5022
00:10:00	4988	5303	02:40:00	4883	5152
00:12:50	5351	5259	02:42:50	4924	4873
00:15:00	5342	4935	02:45:00	5285	4920
00:17:50	5415	4910	02:47:50	4920	5587
00:20:00	5212	4961	02:50:00	4967	4965
00:22:50	5346	4954	02:52:05	4939	4798
00:25:00	5009	4939	02:55:00	4944	5324
00:27:50	5148	5354	02:57:50	4902	5450
00:30:00	4908	5271	03:00:00	4944	4969
00:32:50	5338	4970	03:02:50	4910	5011
00:35:00	4936	5223	03:05:00	4930	4898
00:37:50	4968	4996	03:07:50	4849	4992
00:40:00	5569	5487	03:10:00	4949	5374
00:42:50	5964	4845	03:12:50	5274	5215
00:45:00	5772	7701	03:15:00	5222	4990
00:47:50	5471	4928	03:17:50	5212	5308
00:50:00	4922	5046	03:20:00	4949	5337
00:52:05	4880	4960	03:22:50	4792	5023
00:55:00	9497	4998	03:25:00	5026	5309
00:57:50	4842	5175	03:27:50	5187	5379
01:00:00	4798	4952	03:30:00	4987	5400
01:02:50	4635	4902	03:32:50	5067	5561
01:05:00	4808	5125	03:35:00	4839	5650
01:07:50	4982	5326	03:37:50	5019	5836
01:10:00	5511	5335	03:40:00	5228	5653

01:12:50	5819	5016	03:42:50	5081	4929
01:15:00	5335	5065	03:45:00	5079	4961
01:17:50	4902	5243	03:47:50	5112	5290
01:20:00	5007	5005	03:50:00	5107	5004
01:22:50	4833	5133	03:52:05	5047	4975
01:25:00	4823	5285	03:55:00	5117	4993
01:27:50	4805	5237	03:57:50	4900	5214
01:30:00	4906	5043	04:00:00	5891	5019
01:32:50	5188	5240	04:02:50	5288	7126
01:35:00	4807	5351	04:05:00	5298	5169
01:37:50	4932	4968	04:07:50	5180	5223
01:40:00	4876	5297	04:10:00	4848	4991
01:42:50	4867	5472	04:12:50	5278	5099
01:45:00	4900	5172	04:15:00	5889	5043
01:47:50	4852	5032	04:17:50	5460	5358
01:50:00	4941	5010	04:20:00	5005	5258
01:52:05	4866	5072	04:22:50	5256	4987
01:55:00	4967	4953	04:25:00	5363	4832
01:57:50	5097	4968	04:27:50	5135	4938
02:00:00	4834	4943	04:30:00	4930	4736
02:02:50	4969	4952	04:32:50	5012	5237
02:05:00	5124	4856	04:35:00	4794	5453
02:07:50	4923	4935	04:37:50	4836	5014
02:10:00	5483	4997	04:40:00	5043	5214
02:12:50	5168	4855	04:42:50	5215	5123
02:15:00	5128	4942	04:45:00	5096	5025
02:17:50	5356	5125	04:47:50	4962	5286
02:20:00	5174	5079	04:50:00	4899	5257
02:22:50	4930	5195	04:52:05	5202	5227
02:25:00	5101	4915	04:55:00	4887	5178
02:27:50	4827	5089	04:57:50	4963	5531
02:30:00	4739	5357	05:00:00	5077	4987

Upload Data Dokumen SDM 140 *Number of Threads*

Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:10	5062	4964	02:29:10	4836	4940
00:04:20	5120	4943	02:31:20	5066	4936

00:06:30	4957	5005	02:33:30	4960	4829
00:08:40	4967	4978	02:35:40	4836	4934
00:10:50	4830	4835	02:37:50	5206	5147
00:13:00	5132	4851	02:40:00	5123	4826
00:15:10	5131	4935	02:42:10	5103	5099
00:17:20	5104	5189	02:44:20	5374	4948
00:19:30	5090	5190	02:46:30	5395	5273
00:21:00	5006	5367	02:48:00	5163	5040
00:23:10	4779	5230	02:50:10	5266	4893
00:25:20	4787	4984	02:52:20	5160	4897
00:27:30	5076	5053	02:54:30	5218	4912
00:29:40	5177	4850	02:56:40	4853	5108
00:31:50	5073	4983	02:58:50	4981	5093
00:34:00	4908	4990	03:01:00	4852	5158
00:37:10	5112	4864	03:03:10	5110	5019
00:38:20	5054	5415	03:05:20	5033	4895
00:40:30	5060	5394	03:07:30	4990	4808
00:42:00	5144	5084	03:09:00	4778	4809
00:44:10	5893	4991	03:12:10	4874	5192
00:46:20	5700	4835	03:13:20	5395	5094
00:48:30	5410	4909	03:15:30	5460	4857
00:50:40	4976	5167	03:17:40	5473	4810
00:52:50	5171	4965	03:19:50	4804	4894
00:55:00	5134	4891	03:22:00	4900	5220
00:57:10	5155	5136	03:24:10	5043	5175
00:59:20	5000	5124	03:26:20	5407	4857
01:01:30	4850	5144	03:28:30	5055	5040
01:03:00	5271	4943	03:30:00	5132	4878
01:05:10	5182	5648	03:32:10	5022	5018
01:07:20	4909	5266	03:34:20	5181	4907
01:09:30	4958	5259	03:36:30	5269	5643
01:11:40	5155	5107	03:38:40	5633	5161
01:13:50	5168	5080	03:40:50	5178	5133
01:16:00	5058	4984	03:43:00	4946	5247
01:18:10	5040	5081	03:45:10	5136	5735
01:20:20	5409	4964	03:47:20	5080	4916
01:22:30	5356	5167	03:49:30	5088	4918
01:24:00	5051	4936	03:51:00	5109	4795
01:26:10	4897	4938	03:53:10	5093	5246

01:28:20	4968	5064	03:55:20	4858	5181
01:30:30	4844	4872	03:57:30	4868	5085
01:32:40	5093	4944	03:59:40	4903	4970
01:34:50	5136	4832	04:01:50	5197	5155
01:37:00	5403	5125	04:04:00	5214	4845
01:39:10	5392	5156	04:06:10	6280	4995
01:41:20	5085	5466	04:08:20	8747	5256
01:43:30	5010	5546	04:10:30	6002	4852
01:45:00	5016	5214	04:12:00	5030	4921
01:47:10	4826	5310	04:14:10	4889	5191
01:49:20	4841	5401	04:16:20	5336	4934
01:51:30	4776	4998	04:18:30	6057	4882
01:53:40	4909	4928	04:20:40	5562	5852
01:55:50	4799	4955	04:22:50	5149	5299
01:59:00	4791	5224	04:25:00	4909	5394
02:00:10	5181	4895	04:27:10	5353	4992
02:02:20	5144	4944	04:29:20	5276	5089
02:04:30	5030	5207	04:31:30	5509	4881
02:06:00	5082	4927	04:33:00	5153	5299
02:08:10	5116	4903	04:35:10	5129	5359
02:11:20	5879	4958	04:37:20	5219	5385
02:12:30	5080	4824	04:39:30	5153	5124
02:14:40	4977	4871	04:41:40	4794	5200
02:16:50	4992	5014	04:43:50	4766	4995
02:19:00	4871	5210	04:46:00	4804	4844
02:21:10	5215	5261	04:47:10	4983	5127
02:23:20	5209	5128	04:50:20	4841	5075
02:25:30	5035	5115	04:52:30	5011	5168
02:27:00	4805	5179	04:54:00	4878	5410

Upload Data Dokumen SDM 160 *Number of Threads*

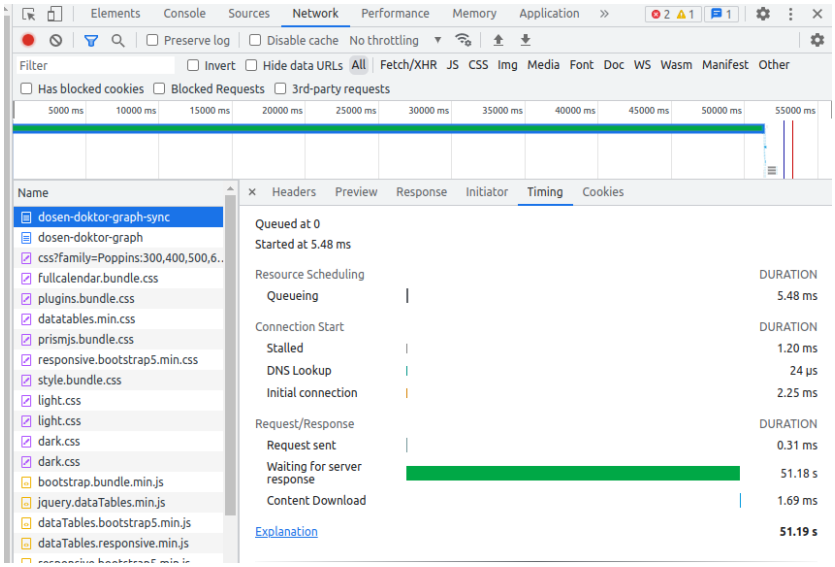
Start time	*Response Time* (ms)		*Start time*	*Response Time* (ms)	
	REST	GraphQL		REST	GraphQL
00:02:20	5230		02:26:20	4805	5439
00:04:00	4794	5166	02:28:00	4795	5216
00:05:40	4849	5178	02:29:40	4906	5200
00:07:20	4843	5056	02:31:20	4919	5107
00:09:00	4853	4872	02:33:00	4942	5131

00:11:20	4835	4855	02:35:20	5111	5068
00:13:00	4821	4865	02:37:00	5102	5131
00:14:40	4905	4781	02:38:40	5152	5164
00:16:20	4851	4829	02:40:20	5235	5086
00:18:00	4829	4967	02:42:00	5484	5231
00:20:20	5212	4890	02:44:20	5474	5258
00:22:00	5278	5662	02:46:00	5123	4942
00:23:40	5183	6091	02:47:40	4853	5007
00:25:20	5113	5984	02:49:20	4825	4757
00:27:00	4806	5677	02:51:00	5086	4843
00:29:20	5103	5275	02:53:20	4955	5211
00:31:00	5065	5050	02:55:00	5180	5150
00:32:40	5344	5389	02:56:40	4869	5188
00:34:20	5724	5277	02:58:20	4919	5092
00:36:00	5752	5227	03:00:00	4693	4899
00:38:20	5315	5289	03:02:20	4820	5194
00:40:00	4953	5154	03:04:00	4906	4957
00:41:40	4752	5284	03:05:40	4993	4766
00:43:20	4794	4788	03:07:20	4961	5189
00:45:00	5131	4922	03:09:00	4831	5168
00:47:20	5107	5202	03:11:20	5159	5133
00:49:00	5022	4991	03:13:00	5105	5146
00:50:40	4911	4780	03:14:40	5201	5172
00:52:20	5026	4908	03:16:20	5266	5001
00:54:00	4824	5289	03:18:00	4909	4997
00:56:20	4834	5087	03:20:20	4872	4853
00:58:00	5083	4860	03:22:00	4845	4913
00:59:40	4861	4839	03:23:40	4803	4908
01:01:20	4830	5181	03:25:20	4707	5016
01:03:00	4831	5037	03:27:00	9203	4837
01:05:20	4907	4882	03:29:20	4683	4961
01:07:00	5151	4792	03:31:00	4856	4905
01:08:40	5211	5167	03:32:40	5257	4903
01:10:20	4814	4966	03:34:20	5098	4826
01:12:00	5170	4971	03:36:00	4931	4904
01:14:20	5147	4845	03:38:20	4794	5113
01:16:00	5129	5283	03:40:00	4911	5431
01:17:40	5072	5464	03:41:40	5196	5498
01:19:20	5061	5500	03:43:20	5199	5347

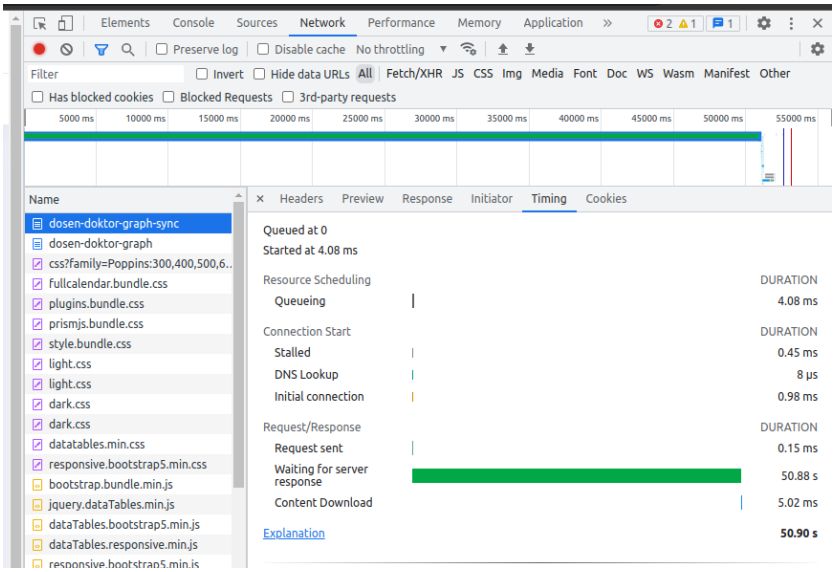
01:21:00	4833	5277	03:45:00	5149	4927
01:23:20	5087	5205	03:47:20	4666	4877
01:25:00	5151	5119	03:49:00	4862	4872
01:26:40	4892	4795	03:50:40	5145	4905
01:28:20	4764	5470	03:52:20	4872	5890
01:30:00	4732	5413	03:54:00	5513	5461
01:32:20	5087	5257	03:56:20	5200	5134
01:34:00	5011	5160	03:58:00	5156	5224
01:35:40	4849	5249	03:59:40	4890	5145
01:37:20	4761	5154	04:01:20	5113	4913
01:39:00	5109	9531	04:03:00	5217	4859
01:41:20	5099	4960	04:05:20	5277	5209
01:43:00	5114	5324	04:07:00	5143	5393
01:44:40	4958	4858	04:08:40	5118	5272
01:46:20	5185	4908	04:10:20	5222	4942
01:48:00	5087	4873	04:12:00	5214	5075
01:50:20	5153	4821	04:14:20	5438	4916
01:52:00	5000	5221	04:16:00	4988	5204
01:53:40	4895	5182	04:17:40	4899	4868
01:55:20	4796	5349	04:19:20	4973	4839
01:57:00	5105	5417	04:21:00	4779	9320
01:59:20	5120	5703	04:23:20	4837	4924
02:01:00	5145	5401	04:25:00	4833	4884
02:02:40	5034	5333	04:26:40	4827	4816
02:04:20	5011	5104	04:28:20	5024	4881
02:06:00	4878	4945	04:30:00	5989	4979
02:08:20	4914	4858	04:32:20	5330	4950
02:10:00	5263	5156	04:34:00	5139	4909
02:11:40	5492	5042	04:35:40	4783	5291
02:13:20	4787	4895	04:37:20	4827	4933
02:15:00	4992	6464	04:39:00	5121	5028
02:17:20	4992	5303	04:41:20	5162	5342
02:19:00	5132	5244	04:43:00	5126	5345
02:20:40	5261	4818	04:44:40	5799	5219
05:22:20	4949	4885	04:46:20	5248	5132
02:24:00	4760	5671	04:48:00	5455	5289

Lampiran 5 Hasil *Fetch* Data *Nested* saat *Integrasi* dengan Sinkron Sister

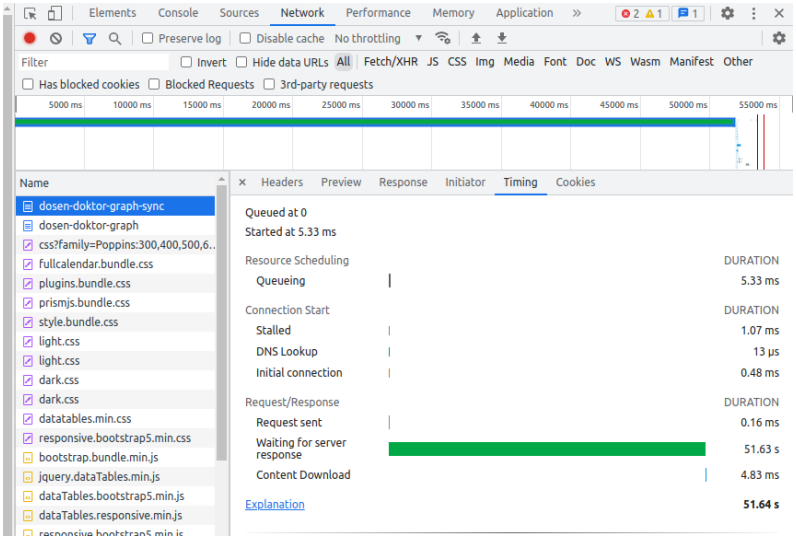
Tangkapan Layar GraphQL – 1



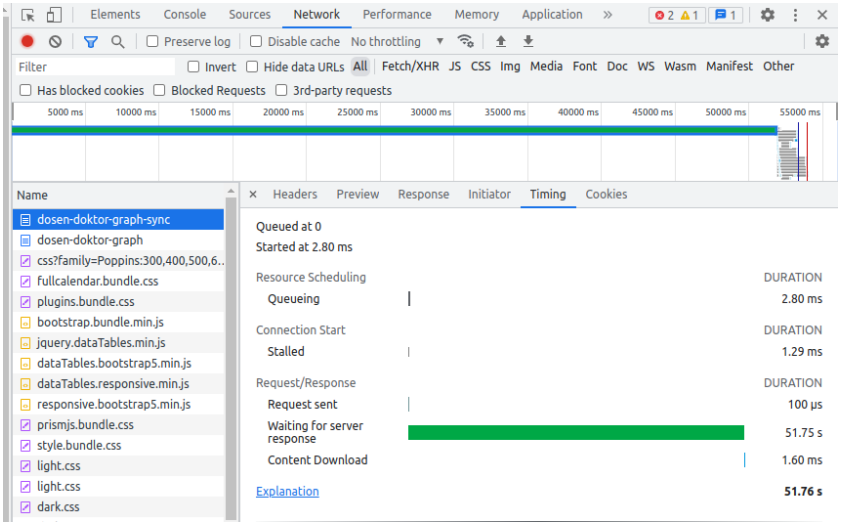
Tangkapan Layar GraphQL – 2



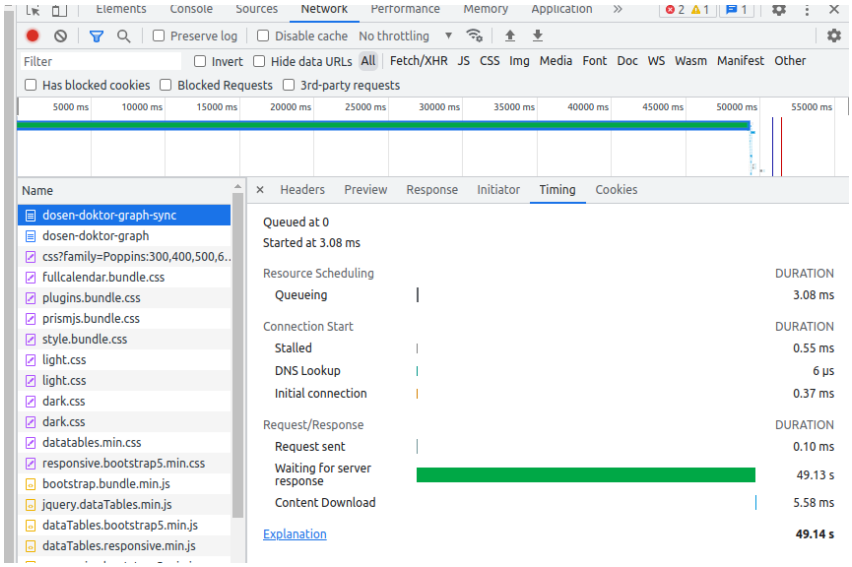
Tangkapan Layar GraphQL - 3



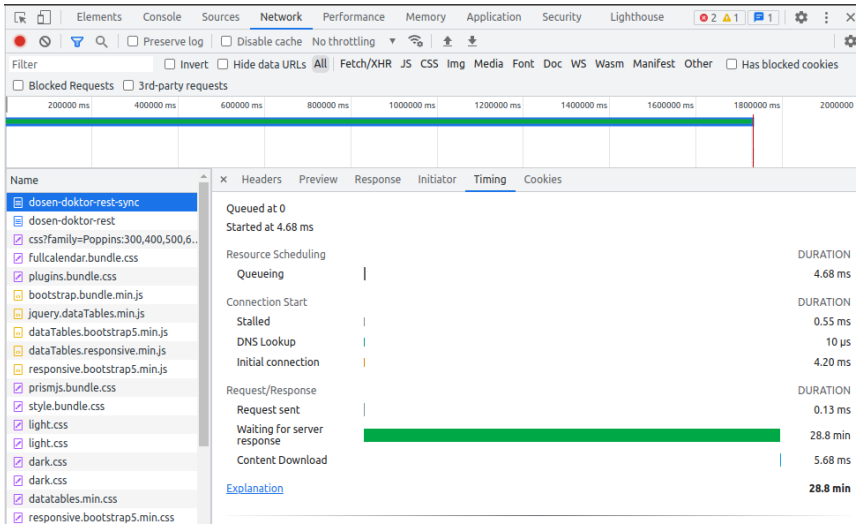
Tangkapan Layar GraphQL – 4



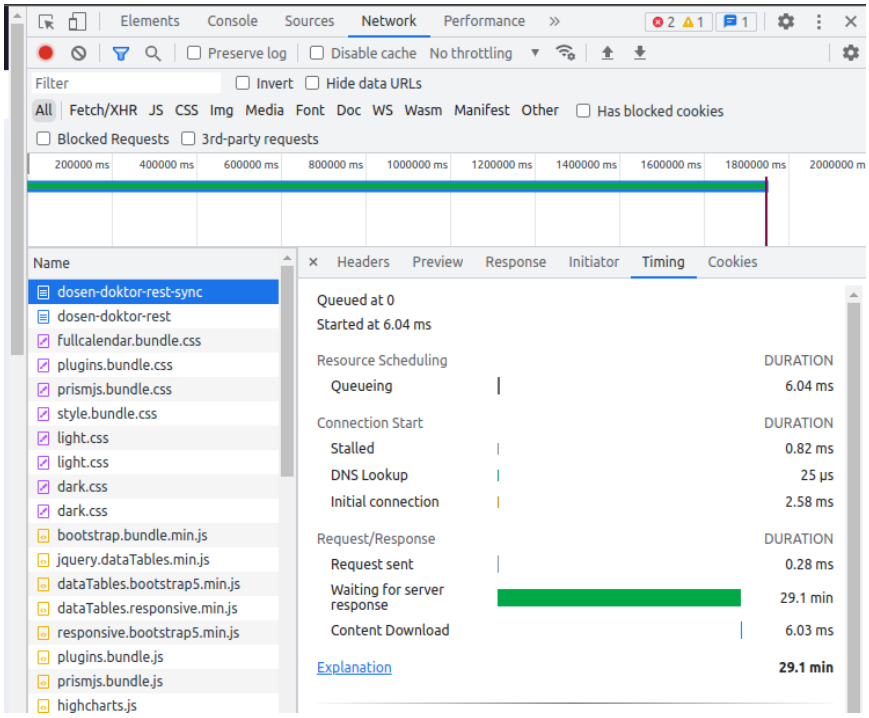
Tangkapan Layar GraphQL - 5



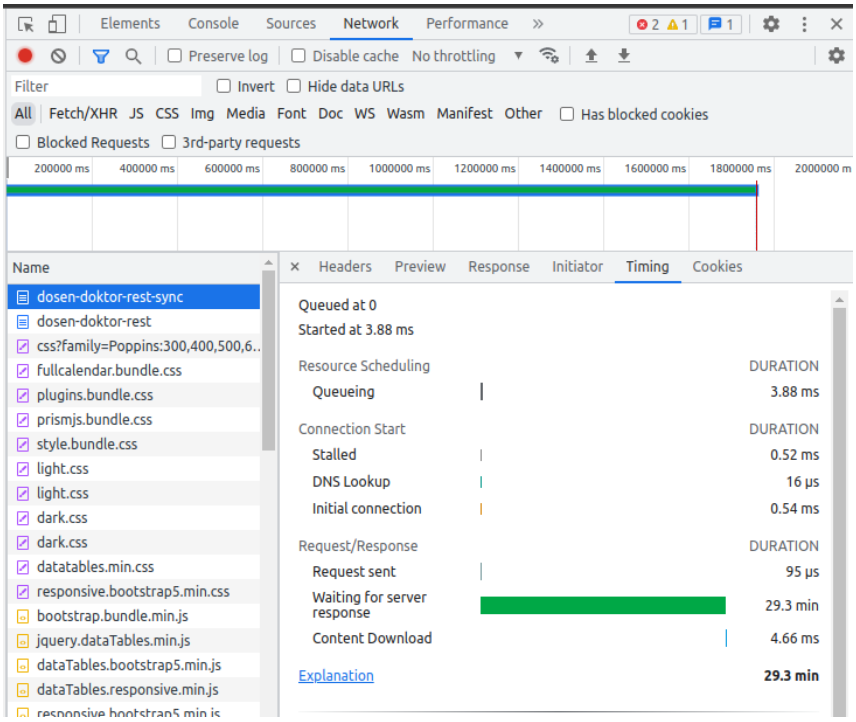
Tangkapan Layar REST – 1



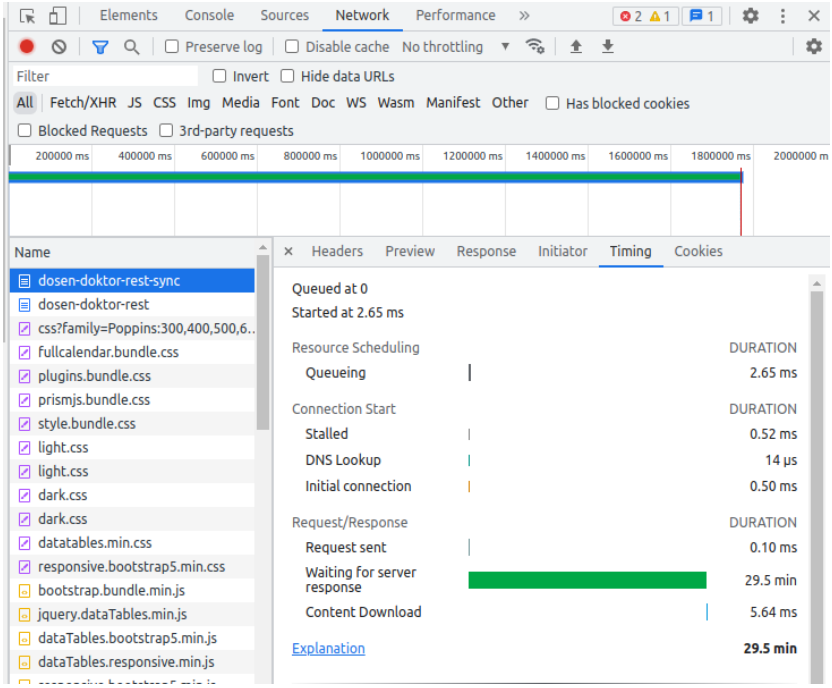
Tangkapan Layar REST - 2



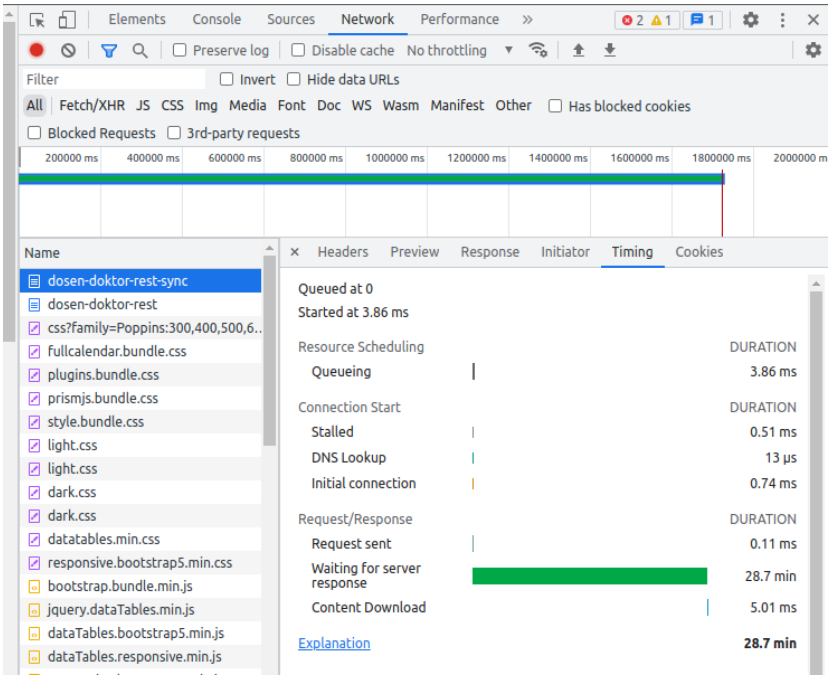
Tangkapan Layar REST - 3



Tangkapan Layar REST – 4



Tangkapan Layar REST - 5



Page Load Time (Second)					
Sample	1	2	3	4	5
REST	1728	1746	1758	1770	1722
GraphQL	51,19	50,9	51,64	51,76	49,14

Lampiran 6 *Source Code REST API* pada *Service Sister*

File api.php

```
<?php

use Illuminate\Http\Request;
use App\Models\HumanResource;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\LoginController;
use App\Http\Controllers\DokumenController;
use App\Http\Controllers\HumanResourceController;
use App\Http\Controllers\FormalEducationController;

Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});

Route::post('login', [LoginController::class, 'login']);
Route::post('logout', [LoginController::class, 'logout']);
Route::get('sumberDayaManusia', [HumanResourceController::class, 'getAll']);
Route::get('pendidikanFormal/{id_sdm}', [FormalEducationController::class,
'getByIDSdm']);
Route::get('dokumen/{id_sdm}', [DokumenController::class, 'getByIDSdm']);
Route::post('dokumen/{id_sdm}', [DokumenController::class,
'uploadDokumenSDM']);
```

File LoginController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Repositories\AuthSister;
use Illuminate\Support\Facades\Http;

class LoginController extends Controller
{
    protected $authSister;

    public function __construct()
    {
        $this->authSister = new AuthSister;
    }
```

```
    public function login(Request $request)
    {
        $response = Http::post('http://10.0.0.108/api/login', [
            'email' => $request->email,
            'password' => $request->password,
        ]);

        return response()->json($response->json(), $response->status());
    }

    public function logout(Request $request)
    {
        if ($request->header('Authorization') != null) {
            $token = explode(' ', $request->header('Authorization'));
            $valid = $this->authSister->verifyTokenRest($token[1]);

            if ($valid) {
                $response = Http::post('http://10.0.0.108/api/logout', [
                    'token' => $token[1],
                ]);

                return handleResponse($response->json(), 'success');
            } else {
                return handleError('Unauthorized', [], 401);
            }
        } else {
            return handleError('Unauthorized', [], 401);
        }
    }
}
```

File HumanResourcesController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Repositories\AuthSister;
use Illuminate\Support\Facades\DB;

class HumanResourceController extends Controller
{
    protected $authSister;
```

```
    public function __construct()
    {
        $this->authSister = new AuthSister;
    }

    public function getAll(Request $request)
    {
        if ($request->header('Authorization') != null) {
            $token = explode(' ', $request->header('Authorization'));
            $valid = $this->authSister->verifyTokenRest($token[1]);

            if ($valid) {
                $humanResources = DB::table('human_resources')->get();
                return handleResponse($humanResources, 'success');
            }else{
                return handleError('Unauthorized', [], 401);
            }
        }else{
            return handleError('Unauthorized', [], 401);
        }
    }
}
```

File FormalEducationController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Repositories\AuthSister;
use Illuminate\Support\Facades\DB;

class FormalEducationController extends Controller
{
    protected $authSister;

    public function __construct()
    {
        $this->authSister = new AuthSister;
    }

    public function getByIDSdm(Request $request, $id_sdm)
    {
        if ($request->header('Authorization') != null) {
```

```
            $token = explode(' ', $request->header('Authorization'));
            $valid = $this->authSister->verifyTokenRest($token[1]);

            if ($valid) {
                $formalEducations = DB::table('formal_education')->where('id_sdm',
'=', $id_sdm)->get();
                return handleResponse($formalEducations, 'success');
            } else {
                return handleError('Unauthorized', [], 401);
            }
        } else {
            return handleError('Unauthorized', [], 401);
        }
    }
}
```

File DokumenController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Support\Str;
use Illuminate\Http\Request;
use App\Repositories\AuthSister;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Storage;

class DokumenController extends Controller
{
    protected $authSister;

    public function __construct()
    {
        $this->authSister = new AuthSister;
    }

    public function getByIDSdm(Request $request, $id_sdm)
    {
        if ($request->header('Authorization') != null) {
            $token = explode(' ', $request->header('Authorization'));
            $valid = $this->authSister->verifyTokenRest($token[1]);

            if ($valid) {
                $dokumens = DB::table('dokumens')->where('id_sdm', '=', $id_sdm)-
>get();
```

```
            return handleResponse($dokumens, 'success');
        } else {
            return handleError('Unauthorized', [], 401);
        }
    } else {
        return handleError('Unauthorized', [], 401);
    }
}

public function uploadDokumenSDM(Request $request, $id_sdm)
{
    if ($request->header('Authorization') != null) {
        $token = explode(' ', $request->header('Authorization'));
        $valid = $this->authSister->verifyTokenRest($token[1]);

        if ($valid) {
            $fileName = $request->file('file')->getClientOriginalName();
            $path = 'storage/rest/'. $fileName;

            $id_jenis_dokumen = 0;
            $jenis_dokumen = '';
            $nama = '';
            $keterangan = '';

            if ($request->id_jenis_dokumen) {
                $id_jenis_dokumen = $request->id_jenis_dokumen;
            }

            if ($request->jenis_dokumen) {
                $jenis_dokumen = $request->jenis_dokumen;
            }

            if ($request->nama) {
                $nama = $request->nama;
            }

            if ($request->keterangan) {
                $keterangan = $request->keterangan;
            }

            Storage::disk('public')->put('rest/' . $fileName,
file_get_contents($request->file('file')->getRealPath()));
            DB::table('dokumens')->insert([
                'id_dokumen' => Str::uuid(),
                'id_sdm' => $id_sdm,
                'tautan' => $path,
```

```
                'id_jenis_dokumen' => $id_jenis_dokumen,
                'jenis_dokumen' => $jenis_dokumen,
                'nama' => $nama,
                'keterangan' => $keterangan
            ]);
            return handleResponse(['path' => $path], 'success');
        } else {
            return handleError('Unauthorized', [], 401);
        }
    } else {
        return handleError('Unauthorized', [], 401);
    }
  }
}
```

Lampiran 7 *Source Code GraphQL* pada *Service Sister*

File schema.graphql

```
scalar Upload
  @scalar(class: "Nuwave\\Lighthouse\\Schema\\Types\\Scalars\\Upload")

type Query {
  sumberDayaManusia: [HumanResource] @field(resolver:
"SDM@sumberDayaManusia")
  pendidikanFormal(id_sdm: String!): [FormalEducation] @field(resolver:
"FormalEducation@byID")
  dokumen(id_sdm: String!): [Dokumen] @field(resolver:
"\\App\\GraphQL\\Mutations\\Dokumen@byIDSdm")
}

type Mutation{
  uploadDokumen(
    file: Upload!,
    id_sdm: String!,
    tautan: String,
    id_jenis_dokumen: Int,
    jenis_dokumen: String,
    nama: String,
    keterangan: String,
  ): String @field(resolver: "Dokumen@uploadDokumen")
}
```

```
type HumanResource {
    id: ID
    id_sdm: String
    nama_sdm: String
    nidn: String
    nip: String
    nama_status_aktif: String
    nama_status_pegawai: String
    jenis_sdm: String
    created_at: String,
    updated_at: String,
    pendidikan_formal: [FormalEducation] @hasMany
    dokumen: [Dokumen] @hasMany
}

type FormalEducation {
    id: ID
    id_sdm: String
    id_pendidikan: String
    jenjang_pendidikan: String
    gelar_akademik: String
    bidang_studi: String
    nama_perguruan_tinggi: String
    tahun_lulus: Int
    created_at: String,
    updated_at: String
}

type Dokumen {
    id: ID
    id_sdm: String
    id_dokumen: String
    tautan: String
    id_jenis_dokumen: Int
    jenis_dokumen: String
    nama: String
    keterangan: String
    tanggal_upload: String
    nama_file: String
    jenis_file: String
    created_at: String
    updated_at: String
}

type User {
    id: ID
```

```
    nama: String
    email: String
}
#import auth.graphql
```

File auth.graphql

```
extend type Mutation{
    login(email: String!, password: String!): String! @field(resolver:
"Authentication@login")
    logout: User @field(resolver: "Authentication@logout")
}
```

File Authentication.php

```
<?php

namespace App\GraphQL\Mutations;

use Softonic\GraphQL\ClientBuilder;
use GraphQL\Type\Definition\ResolveInfo;
use Nuwave\Lighthouse\Support\Contracts\GraphQLContext;
use Symfony\Component\HttpKernel\Exception\HttpException;

use App\Repositories\AuthSister;

final class Authentication
{
    protected $authSister;

    public function __construct()
    {
        $this->authSister = new AuthSister;
    }

    public function login($_, array $args): String
    {
        $client = ClientBuilder::build('http://10.0.0.108/graphql');

        $query = '
        mutation login($email: String!, $password: String!){
            login(email: $email, password: $password)
        }
        ';
```

```
        $variables = [
            'email' => $args['email'],
            'password' => $args['password']
        ];

        $response = $client->query($query, $variables);
        if (property_exists($response, 'errors') && $response->getErrors() != []) {
            return throw new HttpException(401, 'The provided credentials are
incorrect.');
        }else{
            $result = $response->getData();
            return $result['login'];
        }
    }

    public function logout($rootValue, array $args, GraphQLContext $context,
ResolveInfo $resolveInfo)
    {
        if (array_key_exists('HTTP_AUTHORIZATION', $context->request()-
>server())) {
            $arrToken = explode(' ', $context->request()-
>server()['HTTP_AUTHORIZATION']);

            $valid = $this->authSister->verifyToken($arrToken[1]);

            if ($valid) {
                $client = ClientBuilder::build('http://10.0.0.108/graphql');

                $query = '
                mutation logout($token: String!){
                    logout(token: $token){
                        id,
                        nama,
                        email
                    }
                }
                ';

                $variables = [
                    'token' => $arrToken[1]
                ];

                $response = $client->query($query, $variables);
                $result = $response->getData();
                return $result['logout'];
            } else {
```

```
            return throw new HttpException(401, 'Unauthorize');
        }
    } else {
        return throw new HttpException(401, 'Unauthorize');
    }
  }
}
```

File SDM.php

```
<?php

namespace App\GraphQL\Queries;

use App\Models\HumanResource;
use App\Repositories\AuthSister;
use GraphQL\Type\Definition\ResolveInfo;
use Nuwave\Lighthouse\Support\Contracts\GraphQLContext;
use Symfony\Component\HttpKernel\Exception\HttpException;

final class SDM
{
  protected $authSister;

  public function __construct()
  {
    $this->authSister = new AuthSister;
  }

  public function SumberDayaManusia($rootValue, array $args,
GraphQLContext $context, ResolveInfo $resolveInfo)
  {
    if (array_key_exists('HTTP_AUTHORIZATION', $context->request()-
>server())) {
        $arrToken = explode(' ', $context->request()-
>server()['HTTP_AUTHORIZATION']);

        $valid = $this->authSister->verifyToken($arrToken[1]);

        if ($valid) {
          return HumanResource::all();
        } else {
          return throw new HttpException(401, 'Unauthorize');
        }
    } else {
        return throw new HttpException(401, 'Unauthorize');
```

```
        }
    }
}
```

File FormalEducation.php

```
<?php

namespace App\GraphQL\Queries;

use App\Repositories\AuthSister;
use Illuminate\Support\Facades\DB;
use Nuwave\Lighthouse\Support\Contracts\GraphQLContext;
use Symfony\Component\HttpKernel\Exception\HttpException;

final class FormalEducation
{
    protected $authSister;

    public function __construct()
    {
        $this->authSister = new AuthSister;
    }

    public function byID($_, array $args, GraphQLContext $context)
    {
        if (array_key_exists('HTTP_AUTHORIZATION', $context->request()->server())) {
            $arrToken = explode(' ', $context->request()->server()['HTTP_AUTHORIZATION']);

            $valid = $this->authSister->verifyToken($arrToken[1]);

            if ($valid) {
                return DB::table('formal_education')->where('id_sdm', '=', $args['id_sdm'])->get();
            } else {
                return throw new HttpException(401, 'Unauthorize');
            }
        }else{
            return throw new HttpException(401, 'Unauthorize');
        }
    }
}
```

File Dokumen.php

```
<?php

namespace App\GraphQL\Mutations;

use Illuminate\Support\Str;
use Illuminate\Http\Request;
use App\Models\Dokumen as Model;
use App\Repositories\AuthSister;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Storage;
use GraphQL\Type\Definition\ResolveInfo;
use Nuwave\Lighthouse\Support\Contracts\GraphQLContext;
use Symfony\Component\HttpKernel\Exception\HttpException;

final class Dokumen
{
   protected $authSister;

   public function __construct()
   {
      $this->authSister = new AuthSister;
   }

   public function byIDSdm($rootValue, array $args, GraphQLContext
$context, ResolveInfo $resolveInfo)
   {
      if (array_key_exists('HTTP_AUTHORIZATION', $context->request()-
>server())) {
         $arrToken = explode(' ', $context->request()-
>server()['HTTP_AUTHORIZATION']);

         $valid = $this->authSister->verifyToken($arrToken[1]);

         if ($valid) {
            $dokumen = DB::table('dokumens')->where('id_sdm', '=',
$args['id_sdm'])->get();
            return $dokumen;
         } else {
            return throw new HttpException(401, 'Unauthorize');
         }
      } else {
         return throw new HttpException(401, 'Unauthorize');
      }
   }
```

```
   public function uploadDokumen($rootValue, array $args, GraphQLContext
$context, ResolveInfo $resolveInfo)
   {
      if (array_key_exists('HTTP_AUTHORIZATION', $context->request()-
>server())) {
         $response = $context->request()->server('HTTP_AUTHORIZATION');
         $token = substr($response, strpos($response, " ") + 1);

         $valid = $this->authSister->verifyToken($token);

         if ($valid) {
            $fileName = $args['file'];
            $path = 'storage/graphql/' . $fileName->getClientOriginalName();

            $id_jenis_dokumen = 0;
            $jenis_dokumen = '';
            $nama = '';
            $keterangan = '';

            if (array_key_exists('id_jenis_dokumen', $args)) {
               $id_jenis_dokumen = $args['id_jenis_dokumen'];
            }

            if (array_key_exists('jenis_dokumen', $args)) {
               $jenis_dokumen = $args['jenis_dokumen'];
            }

            if (array_key_exists('nama', $args)) {
               $nama = $args['nama'];
            }

            if (array_key_exists('keterangan', $args)) {
               $keterangan = $args['keterangan'];
            }
            Storage::disk('public')->put('graphql/' . $fileName-
>getClientOriginalName(), file_get_contents($args['file']));
            DB::table('dokumens')->insert([
               'id_dokumen' => Str::uuid(),
               'id_sdm' => $args['id_sdm'],
               'tautan' => $path,
               'id_jenis_dokumen' => $id_jenis_dokumen,
               'jenis_dokumen' => $jenis_dokumen,
               'nama' => $nama,
               'keterangan' => $keterangan
            ]);
```

```
            return $path;
        } else {
            return throw new HttpException(401, 'Unauthorize');
        }
    } else {
        return throw new HttpException(401, 'Unauthorize');
    }
  }
}
```

Lampiran 8 *Source Code Repository Pattern* dan *Helper* pada *Service Sister*

File AuthSister.php

```
<?php

namespace App\Repositories;

use Softonic\GraphQL\ClientBuilder;
use Illuminate\Support\Facades\Http;

class AuthSister
{
   public function verifyToken(string $token)
   {
      $client = ClientBuilder::build('http://127.0.0.1:8001/graphql');

      $query = '
      mutation verify($token: String!){
         verify(token: $token)
      }
      ';

      $variables = [
         'token' => $token,
      ];

      $response = $client->query($query, $variables);

      $result = $response->getData();

      return $result['verify'];
   }
```

```
    public function verifyTokenRest(string $token)
    {
        $response = Http::post('http://10.0.0.108/api/verify', [
            'token' => $token,
        ]);


        if ($response->status() == 200) {
            $result = $response->json();
            return $result['verify'];
        }
    }
}
```

File helpers.php

```
<?php

function handleResponse($result, $msg)
{
    $res = [
        'success' => true,
        'message' => $msg,
        'datas' => $result,
    ];
    return response()->json($res, 200);
}

function handleError($error, $errorMsg = [], $code = 404)
{
    $res = [
        'success' => false,
        'message' => $error
    ];

    if (!empty($errorMsg)) {
        $res['data'] = $errorMsg;
    }
    return response()->json($res, $code);
}
```

Lampiran 9 *Source Code REST* pada *Service Auth*

File api.php

```
<?php

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\LoginController;

Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});

Route::post('/login', [LoginController::class, 'login']);
Route::post('/verify', [LoginController::class, 'verify']);
```

File LoginController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\User;
use App\Models\UserToken;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;

class LoginController extends Controller
{
    public function login(Request $request)
    {
        $request->validate([
            'email' => 'required|email',
            'password' => 'required',
        ]);

        $user = User::where('email', $request->email)->first();

        if (!$user || !Hash::check($request->password, $user->password)) {
            $error = 'The provided credentials are incorrect.';
```

```
            return handleError('Unprocessable Content', ['error' => $error], 422);
        }

        $token = $user->createToken($request->email)->plainTextToken;

        DB::table('user_tokens')->insert([
            'user_id' => $user->id,
            'token' => $token,
            'created_at' => now(),
            'updated_at' => now()
        ]);

        return handleResponse(['token' => $token], 'success');
    }

    public function verify(Request $request)
    {
        $request->validate([
            'token' => 'required'
        ]);

        $token = DB::table('user_tokens')->where('token', $request->token)-
>first();
        if ($token != null) {
            $datas = ['verify' => true];
            $code = 200;
        }else{
            $datas = ['verify' => false];
            $code = 401;
        }

        return response()->json($datas, $code);
    }

    public function logout(Request $request)
    {
        $request->validate([
            'token' => 'required'
        ]);

        $user = DB::table('user_tokens')->where('token', $request->token)-
>with('user')->first();

        if ($user) {
            $token = UserToken::find($user->id);
            $token->delete();
```

```
            return $user->user;
        } else {
            return null;
        }
    }
}
```

Lampiran 10 *Source Code GraphQL* pada *Service Auth*

File schema.graphql

```
type Query {
    me: User! @auth @guard
}

type Mutation {
    login(email: String!, password: String!): String!@field(resolver:
"Authentication@login")
    verify(token: String!): Boolean! @field(resolver: "Authentication@verify")
    logout(token: String!): User @field(resolver: "Authentication@logout")
}

type User {
    id: ID
    nama: String
    email: String
    created_at: String
    updated_at: String
}
```

File Authentication.php

```
<?php

namespace App\GraphQL\Mutations;

use App\Models\User;
use App\Models\UserToken;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\ValidationException;

final class Authentication
{
```

```
    public function login($_, array $args)
    {

        $user = User::where('email', $args['email'])->first();

        if (!$user || !Hash::check($args['password'], $user->password)) {
            throw ValidationException::withMessages([
                'email' => ['The provided credentials are incorrect.'],
            ]);
        }

        $token = $user->createToken($args['email'])->plainTextToken;

        DB::table('user_tokens')->insert([
            'user_id' => $user->id,
            'token' => $token,
            'created_at' => now(),
            'updated_at' => now()
        ]);

        return $token;
    }

    public function verify($_, array $args)
    {
        $token = DB::table('user_tokens')->where('token', $args['token'])->first();
        if ($token != null) {
            return true;
        }else{
            return false;
        }
    }

    public function logout($_, array $args)
    {
        $user = DB::table('user_tokens')->where('token', $args['token'])-
>with('user')->first();
        if ($user) {
            $token = UserToken::find($user->id);
            $token->delete();
            return $user->user;
        } else {
            return null;
        }
    }
}
```

Lampiran 11 *Source Code Helper* pada *Service Auth*

File helpers.php

```
<?php


function handleResponse($result, $msg)
{
    $res = [
        'success' => true,
        'message' => $msg,
        'datas' => $result,
    ];

    return response()->json($res, 200);
}

function handleError($error, $errorMsg = [], $code = 404)
{
    $res = [
        'success' => false,
        'message' => $error
    ];

    if (!empty($errorMsg)) {
        $res['data'] = $errorMsg;
    }
    return response()->json($res, $code);
}
```

Lampiran 12 *Source Code* Integrasi Sinkron Sister

File web.php

```
<?php
++++++
    Route::group(['prefix' => 'data-tugas-akhir'], function(){
        Route::get('/dosen-doktor-rest', [TugasAkhirController::class,
'getDosenDoktorRest'])->name('dosen-doktor-rest');
        Route::get('/dosen-doktor-rest-sync', [TugasAkhirController::class,
'getDosenDoktorRestSync'])->name('dosen-doktor-rest-sync');
```

```
    Route::get('/dosen-doktor-graph', [TugasAkhirController::class,
'getDosenDoktorGraph'])->name('dosen-doktor-graph');
    Route::get('/dosen-doktor-graph-sync', [TugasAkhirController::class,
'getDosenDoktorGraphSync'])->name('dosen-doktor-graph-sync');
  });
+++++++
});
```

File TugasAkhirController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\PegawaiPerguruanTinggi;
use Error;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Softonic\GraphQL\ClientBuilder;
use Illuminate\Support\Facades\Http;
use App\Repositories\SisterUniversitas\Pegawai;

class TugasAkhirController extends Controller
{
  protected $api;

  public function __construct()
  {
    $this->api = new Pegawai();
  }

  public function getDosenDoktorRest(Request $request)
  {
    $datas = [];
    $rute = route('dosen-doktor-rest');
    $title = 'Data Pegawai/SDM di Universitas';

    $sqlQuery = "SELECT t1.id_sdm, t1.nama_sdm, t2.bidang_studi,
t2.jenjang_pendidikan, t2.gelar_akademik
          FROM tugas_akhir_s_d_m_s AS t1
          JOIN tugas_akhir_pend_formals AS t2 ON t2.id_sdm = t1.id_sdm
          WHERE jenjang_pendidikan = 'S3'
          AND jenis_sdm = 'Dosen'
          AND nama_status_aktif = 'Aktif'
          ORDER BY nama_sdm";
```

```
        $datas = DB::connection('mysql')->select(DB::raw($sqlQuery));

        return view('app.dashboard.tugas-akhir.dosen-doktor', [
            'data' => $title,
            'rute' => $rute,
            'datas' => $datas
        ]);
    }

    public function getDosenDoktorGraph()
    {
        $datas = [];
        $rute = route('dosen-doktor-rest');
        $title = 'Data Pegawai/SDM di Universitas';

        $sqlQuery = "SELECT t1.id_sdm, t1.nama_sdm, t2.bidang_studi,
t2.jenjang_pendidikan, t2.gelar_akademik
                FROM tugas_akhir_s_d_m_graphs AS t1
                JOIN tugas_akhir_pend_formal_graphs AS t2 ON t2.id_sdm =
t1.id_sdm
                WHERE jenjang_pendidikan = 'S3'
                AND jenis_sdm = 'Dosen'
                AND nama_status_aktif = 'Aktif'
                ORDER BY nama_sdm";

        $datas = DB::connection('mysql')->select(DB::raw($sqlQuery));

        return view('app.dashboard.tugas-akhir.dosen-doktor-graph', [
            'data' => $title,
            'rute' => $rute,
            'datas' => $datas
        ]);
    }

    public function getDosenDoktorGraphSync()
    {
        $token = session()->get('tokenTA');

        $options = [
            'headers' => [
                'Authorization' => "Bearer $token",
            ]
            ];

        $client = ClientBuilder::build('http://54.167.250.198/graphql', $options);
```

```
    $query = '
    query{
      sumberDayaManusia{
        id_sdm,
        nidn,
        nama_sdm,
        nama_status_aktif,
        jenis_sdm,
        pendidikan_formal{
          id_pendidikan,
          id_sdm,
          jenjang_pendidikan,
          bidang_studi,
        }
      }
    }
    ';

    $variables = [];

    $response = $client->query($query, $variables);

    $datas = $response->getData();

    $success = $this->savePengawaiGraph($datas['sumberDayaManusia']);
    if ($success) {
      return redirect(route('dosen-doktor-graph'));
    }
  }

  private function savePengawaiGraph($dosens)
  {
    foreach($dosens as $dosen){
      DB::beginTransaction();
      try {
        DB::table('tugas_akhir_s_d_m_graphs')->insertOrIgnore([
          'id_sdm' => $dosen['id_sdm'],
          'nama_sdm' => $dosen['nama_sdm'],
          'nidn' => $dosen['nidn'],
          'nama_status_aktif' => $dosen['nama_status_aktif'],
          'jenis_sdm' => $dosen['jenis_sdm'],
        ]);
        DB::commit();
      } catch (Error $e) {
        DB::rollBack();
        dd($e);
```

```
        }

        foreach($dosen['pendidikan_formal'] as $pend_formal){
            DB::beginTransaction();
            try {
                DB::table('tugas_akhir_pend_formal_graphs')->insertOrIgnore([
                    'id_pendidikan' => $pend_formal['id_pendidikan'],
                    'id_sdm' => $dosen['id_sdm'],
                    'jenjang_pendidikan' => $pend_formal['jenjang_pendidikan'],
                    'bidang_studi' => $pend_formal['bidang_studi'],
                ]);
                DB::commit();
            } catch (Error $e) {
                DB::rollBack();
                dd($e);
            }
        }
    }

    return true;
}

private $id_sdm;

public function getDosenDoktorRestSync()
{
    $token = session()->get('tokenTA');
    $response = Http::withHeaders([
        'Accept' => 'application/json',
        'Authorization' => "Bearer $token"
    ])->get('http://54.167.250.198/api/sumberDayaManusia');

    if ($response->status() == 200) {
        $dosens = $response->json();
    }

    $successSavePegawai = $this->savePegawai($dosens['datas']);
    if ($successSavePegawai) {
        $this->savePendidikanFormal();
    }


    return redirect(route('dosen-doktor-rest'));
}

private function savePegawai($dosens)
```

```
{
    foreach ($dosens as $item) {
        $this->id_sdm[] = $item['id_sdm'];
        DB::beginTransaction();
        try {
            DB::table('tugas_akhir_s_d_m_s')->insertOrIgnore([
                'id_sdm' => $item['id_sdm'],
                'nama_sdm' => $item['nama_sdm'],
                'nidn' => $item['nidn'],
                'nama_status_aktif' => $item['nama_status_aktif'],
                'jenis_sdm' => $item['jenis_sdm'],
            ]);
            DB::commit();
        } catch (Error $e) {
            DB::rollBack();
            dd($e);
        }

    }

    return true;
}

private function savePendidikanFormal()
{
    $idSdmFail = [];
    foreach($this->id_sdm as $sdm){
        $token = session()->get('tokenTA');
        $response = Http::withHeaders([
            'Accept' => 'application/json',
            'Authorization' => "Bearer $token"
        ])->get('http://54.167.250.198/api/pendidikanFormal/'. $sdm);

        $result = $response->json();
        if ($response->status() == 200) {
            foreach($result['datas'] as $item){
                DB::beginTransaction();
                try {
                    DB::table('tugas_akhir_pend_formals')->insertOrIgnore([
                        'id_pendidikan' => $item['id_pendidikan'],
                        'id_sdm' => $item['id_sdm'],
                        'jenjang_pendidikan' => $item['jenjang_pendidikan'],
                        'bidang_studi' => $item['bidang_studi'],
                    ]);
                    DB::commit();
                } catch (Error $e) {
```

```
                DB::rollBack();
                dd($e);
            }
        }
    }else{
        $idSdmFail[] = $sdm;
    }
}

if ($idSdmFail != []) {
    foreach ($idSdmFail as $idSdm) {
        $token = session()->get('tokenTA');
        $response = Http::withHeaders([
            'Accept' => 'application/json',
            'Authorization' => "Bearer $token"
        ])->get('http://54.167.250.198/api/pendidikanFormal/' . $idSdm);

        $result = $response->json();
        if ($response->status() == 200) {
            foreach ($result['datas'] as $item) {
                DB::beginTransaction();
                try {
                    DB::table('tugas_akhir_pend_formals')->insertOrIgnore([
                        'id_pendidikan' => $item['id_pendidikan'],
                        'id_sdm' => $item['id_sdm'],
                        'jenjang_pendidikan' => $item['jenjang_pendidikan'],
                        'bidang_studi' => $item['bidang_studi'],
                    ]);
                    DB::commit();
                } catch (Error $e) {
                    DB::rollBack();
                    dd($e);
                }
            }
        } else {
            $idSdmFail[] = $idSdm;
        }
    }
}else {
    return true;
}
}

public function tarikDokumen()
{
    $sdm = PegawaiPerguruanTinggi::all();
```

```
        foreach($sdm as $item){
            $results = $this->api->dokumen_sdm($item['id_sdm']);
            if ($results['statusCode'] == 200) {

                foreach($results['data'] as $dokumen){
                    DB::beginTransaction();
                    try{
                        DB::table('tugas_akhir_dokumens')->insertOrIgnore([
                            'id_dokumen' => $dokumen['id'],
                            'id_sdm' => $item['id_sdm'],
                            'id_jenis_dokumen' => $dokumen['id_jenis_dokumen'],
                            'jenis_dokumen' => $dokumen['jenis_dokumen'],
                            'nama' => $dokumen['nama'],
                            'keterangan' => $dokumen['keterangan'],
                            'tautan' => $dokumen['tautan'],
                            'tanggal_upload' => $dokumen['tanggal_upload'],
                            'nama_file' => $dokumen['nama_file'],
                            'jenis_file' => $dokumen['jenis_file'],
                        ]);
                        DB::commit();
                    }catch(Error $e){
                        DB::rollBack();
                        dd($e);
                    }
                }
            }
        }
    }
}
```

LEMBAR PERBAIKAN SKRIPSI

"ANALISIS KINERJA REST API DAN GRAPHQL PADA TEKNOLOGI *WEB SERVICES*"

OLEH:

DARUL IKHSAN
D121181017

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 30 November 2022.

Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Dr. Eng. Ir. Muhammad Niswar, S.T., M.InfoTech.	
Sekretaris	Iqra Aswad, S.T., M.T.	
Anggota	Dr. Eng. Adi Wahyudi Paundu, S.T., M.T.	
	Dr. Eng. Ir. Wardi, S.T., M.Eng.	

Persetujuan Perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr. Eng. Ir. Muhammad Niswar, S.T., M.InfoTech.	
II	Iqra Aswad, S.T., M.T.	