

大規模ゲーム開発を 支えるデータ分析・可 視化基盤の構築

Google
Cloud
Next

Tokyo

Proprietary



高本 真宏

株式会社 カプコン
CS 第二開発統括
システム基盤部
ソリューション開発室

2018 年サーバー エンジニアとして、株式会社カプコン入社。

各種タイトルのサーバー エンジニアやデータ分析基盤構築の担当を経て、現在はデータや AI/ML の活用を軸とした、開発や品質保証 (QA) の効率化業務にエンジニア、PM として従事。

所属プロジェクトについて

最適化
費用
Cost
Optimization

Maximize
Value
価値
最大化

CAPCOM[®]

アジェンダ

- 01 課題とその背景
- 02 解決策
- 03 運用のポイント
- 04 今後の展望
- 05 まとめ

01. 課題とその背景

ゲーム開発は大規模化している

近年のゲーム開発は非常に大規模なプロジェクトです。

そのため、開発の現場において状況を正しく、客観的に把握するために、データによる可視化が必要になってきます。



開発データ収集における課題



オンライン要素 がないタイトル

ゲームサーバーを持たないため、サーバーとの通信は基本的に行っていない。



大量のテストプ レイ機材

テストプレイ用の機材は、QA 繁忙期には 1 タイトル数十台規模の台数に。



サポート用途の ログは...?

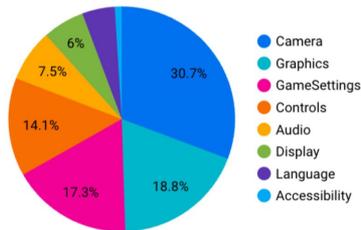
ユーザーサポート目的のログであれば多くの場合で収集の仕組みがあるが、開発状況の把握には適さない。

集計や可視化における課題

表計算ソフトによる集計
※画面はイメージです

		C	D	E	F	G	H	I	J	K	L	M	N
1	カテゴリ												
2	CONTROLS	1064	2402	2267	3114	3716	4317	4919	5520	6122	6723	7325	7926
3	CONTROLS	4	3	5	7	8	9	10	11	12	13	14	15
4	CONTROLS	5	0	1	0	2	3	5	6	9	0	10	12
5	CONTROLS	1063	2405	2271	1608	2212	2362	2512	2662	2812	2963	3113	3263
6	CONTROLS	1063	2400	2256	1608	2205	2354	2503	2652	2801	2950	3099	3248
7	CONTROLS	5	5	16	0	6	5	5	4	4	4	3	3
8	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
9	CONTROLS	1068	2405	2272	1598	2205	2346	2491	2637	2783	2929	3074	3220
10	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
11	CONTROLS	0	0	0	10	10	13	16	19	22	25	28	31
12	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
13	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
14	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
15	CONTROLS	1068	2405	2272	1608	2210	2359	2507	2656	2805	2954	3102	3251
16	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
17	CONTROLS	1068	2405	2272	1608	2210	2359	2507	2656	2805	2954	3102	3251
18	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
19	CONTROLS	1066	2401	2267	1589	2190	2333	2477	2620	2764	2907	3051	3194
20	CONTROLS	2	4	5	19	21	26	31	36	41	47	52	57
21	CONTROLS	1068	2402	2270	1608	2209	2358	2507	2655	2804	2953	3102	3251
22	CONTROLS	0	3	2	0	5	1	1	1	1	1	1	0
23	CONTROLS	5	11	8	0	2	0	4	2	5	3	6	4
24	CONTROLS	1063	2384	2264	1608	2209	2359	2510	2660	2811	2961	3112	3262
25	CONTROLS	1068	2404	2270	1608	2209	2358	2506	2655	2803	2952	3101	3249
26	CONTROLS	0	1	2	0	1	1	1	1	1	1	2	2
27	CONTROLS	1068	2404	2271	2142	2744	3002	3361	3670	3979	4288	4597	4906
28	CONTROLS	0	1	1	0	1	1	1	1	1	1	1	1
29	CONTROLS	1065	2400	2268	123	725	429	133	163	459	755	1040	1345
30	CONTROLS	3	5	4	0	1	3	4	2	5	6	8	1
31	CONTROLS	0	0	0	0	0	0	0	0	0	0	0	0
32	CONTROLS	1065	2398	2258	1606	2203	2351	2499	2647	2796	2944	3092	3241
33	CONTROLS	3	7	14	2	2	2	2	2	2	2	2	2
34	CONTROLS	1067	2401	2258	1608	1608	1608	1608	1608	1608	1608	1608	1608
35	GRAPHICS	1	4	4	6	8	9	11	12	14	15	17	18

BI ツールによる集計
※画面はイメージです



CategoryName	MOptionValue	Record Count
GameSettings	表示しない	1
GameSettings	標準	1
GameSettings	縮小	1
GameSettings	標準	1
GameSettings	縮小	1
GameSettings	大	1
GameSettings	標準	1
GameSettings	大	1
GameSettings	標準	1
GameSettings	標準	1
GameSettings	大	1

表計算ソフトでの集計

- データの入力やまとめに時間がかかる
- セル上に埋め込まれた計算式のメンテナンスが大変
- 関係者への共有の仕方やバージョン管理など、運用上の取り回し

※画面はイメージです

2025-01-01	2025-01-08	2025-01-15	2025-01-22	2025-01-29	2025-02-05	2025-02-12	2025-02-19	2025-02-26	2025-03-06	2025-03-12	2025-03-19	2025-03-26	2025-04-02
1064	2402	2287	3114	3716	4317	4919	5520	6122	6723	7325	7926	8528	9129
4	3	5	7	8	9	10	11	12	13	14	15	16	17
5	0	1	0	2	3	5	6	9	9	10	12	13	12
1063	2405	2271	1608	2212	2362	2512	2662	2812	2963	3113	3263	3413	3563
1063	2400	2256	1608	2205	2354	2503	2652	2801	2950	3099	3248	3397	3546
5	5	16	0	6	5	5	4	4	4	3	3	2	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1068	2405	2272	1698	2200	2344	2491	2637	2783	2929	3074	3220	3366	3511
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	10	10	13	16	19	22	25	28	31	34	37
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1068	2405	2272	1608	2210	2359	2507	2656	2805	2954	3102	3251	3400	3548
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1068	2405	2272	1608	2210	2359	2507	2656	2805	2954	3102	3251	3400	3548
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1066	2401	2267	1599	2190	2333	2477	2620	2764	2907	3051	3194	3338	3481
2	4	5	19	21	26	31	36	41	47	52	57	62	67
1068	2402	2270	1608	2209	2358	2507	2655	2804	2953	3102	3251	3399	3548
0	3	2	0	1	1	1	1	1	1	1	0	0	0
5	11	8	0	2	0	4	2	5	3	6	4	1	0
1063	2394	2264	1608	2209	2359	2510	2660	2811	2961	3112	3262	3413	3563
1068	2404	2270	1608	2209	2358	2506	2655	2803	2952	3101	3249	3398	3546
0	1	2	0	1	1	1	1	1	2	2	2	2	2
1068	2404	2271	2142	2744	3052	3361	3670	3979	4288	4597	4906	5215	5524
0	1	1	0	1	1	1	1	1	1	1	1	1	1
1065	2400	2268	123	725	429	133	163	459	755	1040	1345	1642	1938
3	5	4	0	1	3	4	2	5	6	8	1	5	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1065	2388	2258	1606	2203	2351	2499	2647	2796	2944	3092	3241	3389	3537
3	7	14	2	2	2	2	2	2	2	2	2	2	2
1067	2401	2268	1608	1608	1608	1608	1608	1608	1608	1608	1608	1608	1608
1	4	4	6	6	9	11	12	14	15	17	18	20	21

BI ツールでの集計

- SQL 等、問い合わせ言語の知識が必要
- 高機能なゆえに使いこなすのに一定のスキルが必要
- 汎用的なシステムなので痒いところに※画面はイメージです
手が届きにくい

```

95 SELECT
96   m.*,
97   p.OptionID,
98   p.OptionValue,
99   CASE
100     WHEN m.IsIdentifier THEN CASE
101       WHEN p.OptionValue = m.DefaultValue THEN 'フック'
102       WHEN p.OptionValue = m.DefaultValue THEN 'フック2ユT'
103       WHEN p.OptionValue = m.DefaultValue THEN 'フック2ユ2'
104     END
105   | ELSE CASE
106     WHEN SAFE_CAST(p.OptionValue AS INT64) >> ARRAY_LENGTH(m.OptionValues) THEN NULL
107     ELSE SAFE_CAST(m.OptionValues[SAFE_CAST(p.OptionValue AS INT64)] AS STRING)
108   END
109 END AS SupOptionValue
110 FROM
111   _connect_master AS m
112 LEFT JOIN
113   _option_rec AS p
114 ON
115   p.OptionID = m.ID
116 AND SAFE_CAST(m.OptionValue AS STRING) =
117 CASE
118   WHEN m.IsIdentifier THEN CASE
119     WHEN p.OptionValue = m.DefaultValue THEN 'フック'
120     WHEN p.OptionValue = m.DefaultValue THEN 'フック2ユT'
121     WHEN p.OptionValue = m.DefaultValue THEN 'フック2ユ2'
122   END
123   | ELSE CASE
124     WHEN SAFE_CAST(p.OptionValue AS INT64) >> ARRAY_LENGTH(m.OptionValues) THEN NULL
125     ELSE SAFE_CAST(m.OptionValues[SAFE_CAST(p.OptionValue AS INT64)] AS STRING)
126   END
127 END |
128 LEFT AS (
129 SELECT
130   DISTINCT CategoryID,
131   CategoryName,
132   ID,

```

Name	MOptionValue	Record Count
1.	タイプ3	1
2.	タイプ1	1
3.	タイプ5	1
4.	タイプ4	1
5.	悪い	1
6.	標準	1
7.	悪い	1
8.	すべて初期で設定	1
9.	タイプ1	1
10.	タイプ2	1
11.	タイプ2	1
12.	タイプ2	1
13.	タイプ1	1

1-80/85 < >

データの活用が限定的な状態

従来の方法では、そういった課題があり、特に導入や運用のハードルやコストが高かった。

そのため、データ活用自体は行っているも、一過性のものだったり限定的なものだった。

02. 解決策

開発や QA の現場でもデータの活用ニーズはある

大規模化しているゲーム開発・QAにおいて、客観的かつ俯瞰的に
状況把握するため。

そして、それは一過性の取り組みではなく、継続的なものにしたい。

行った取り組み

データを集める

タイトル問わず、共通した方法で、データを収集するための基盤を構築。

データを可視化する

収集したデータを手軽に可視化できる基盤。

そして、共通利用だけでなく、様々な現場のニーズに答えられる用途別の専用アプリケーション。

データを活用する

データ分析の専門家でなくても、容易にデータから意味を読み解くことができる仕組み。

データを集める

データを集めるにもログの送信を仕込んだり、一定のコストがかかる。

極力シンプルかつ統一的な方法で実現。



データを可視化する

大きく分けて2つ。

汎用的で誰でも GUI で可視化できる
仕組み。

特定の用途に応じた専用の仕組み。



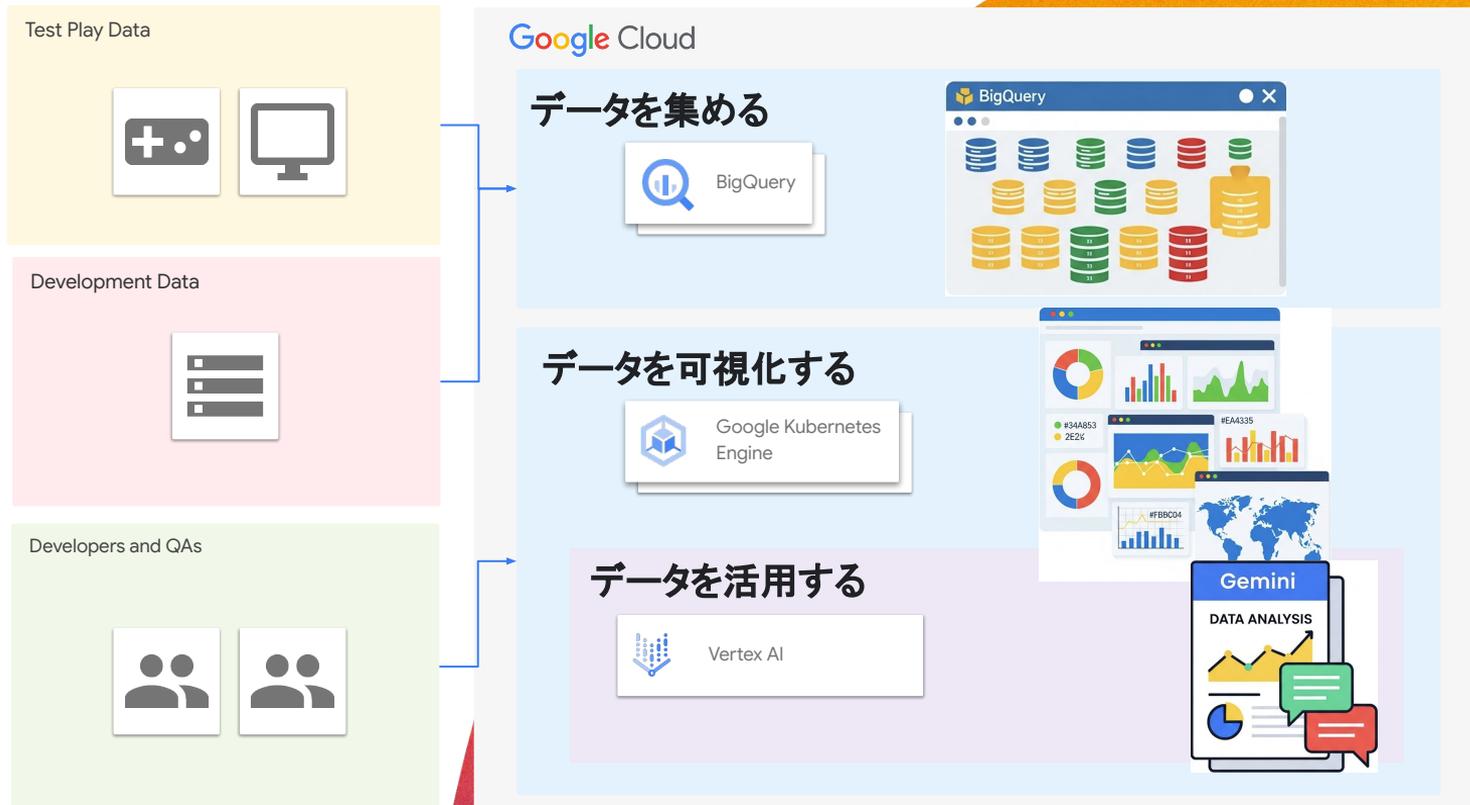
データを活用する

「データ分析の専門家でなくても、データ活用できる」ように、状況判断や意思決定ができる情報に落とし込む。

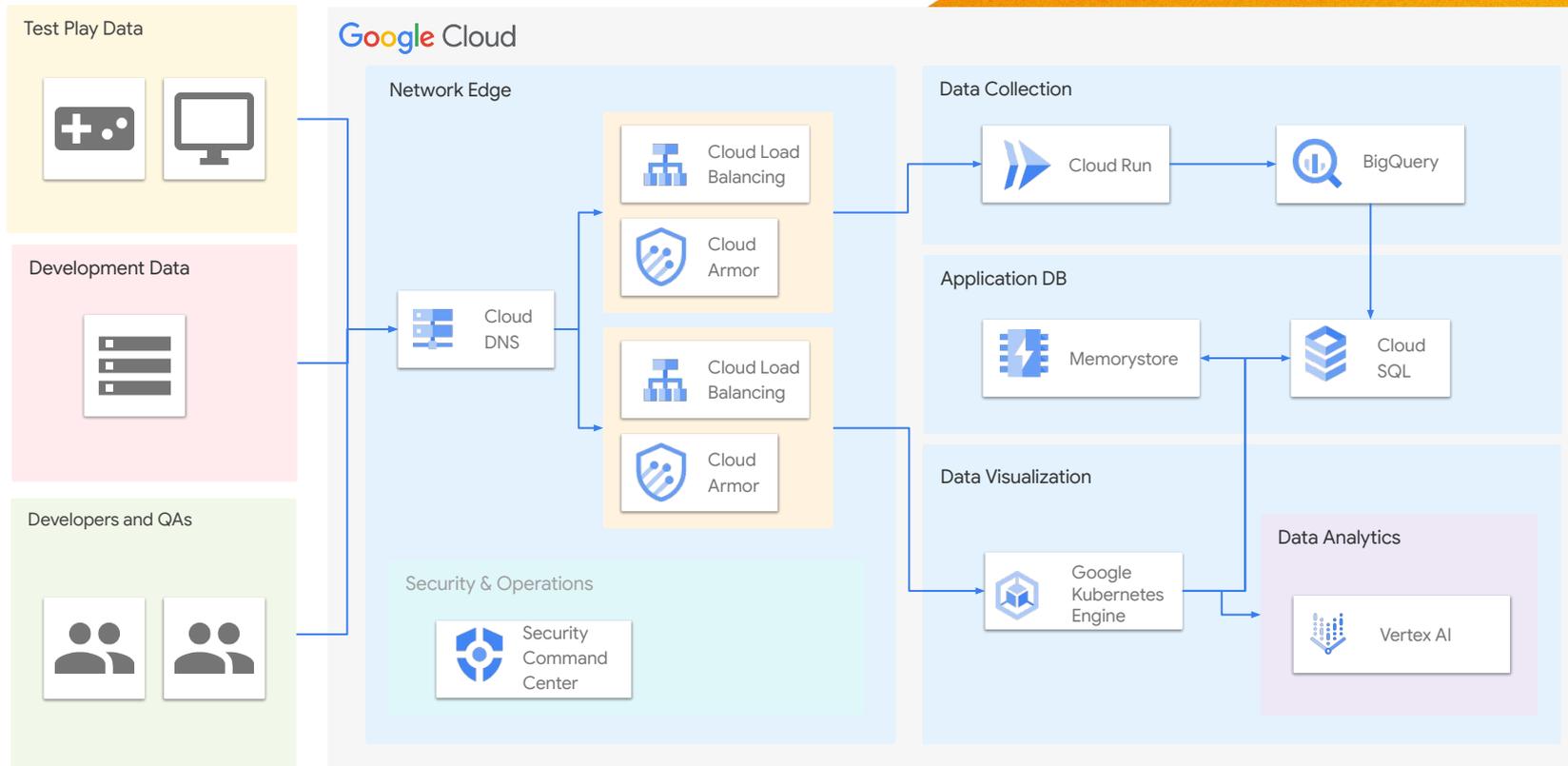


全体イメージと主要プロダクト

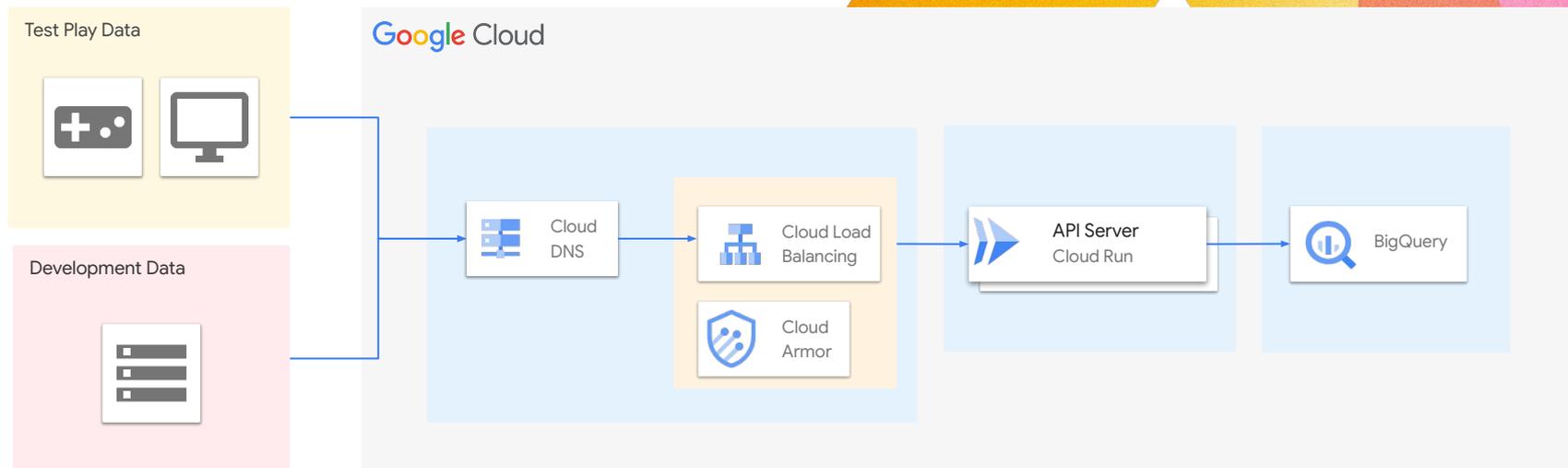
Overall view



Architecture: Capcom Development Data Platform

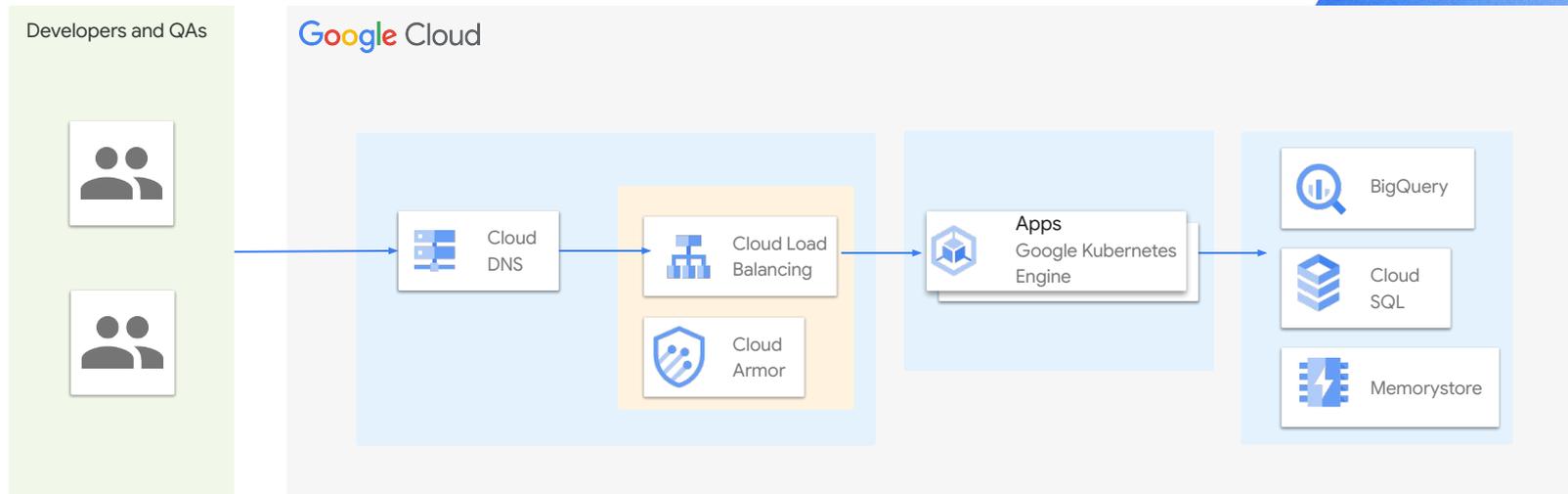


データ収集部分



社内のみで発生するデータとはいえ膨大なデータ量となるため、BigQuery を選択。
そして、API サーバーは構築や運用の手間を少なくするため、Cloud Run を選択。

データ可視化基盤



様々なアプリケーションを柔軟に運用したい。

運用の手間を少なくするため、Google Kubernetes Engine (GKE) を採用。

データベースについての補足

アプリケーションの要件に応じて
CloudSQL や Memorystore も使用し
ています。

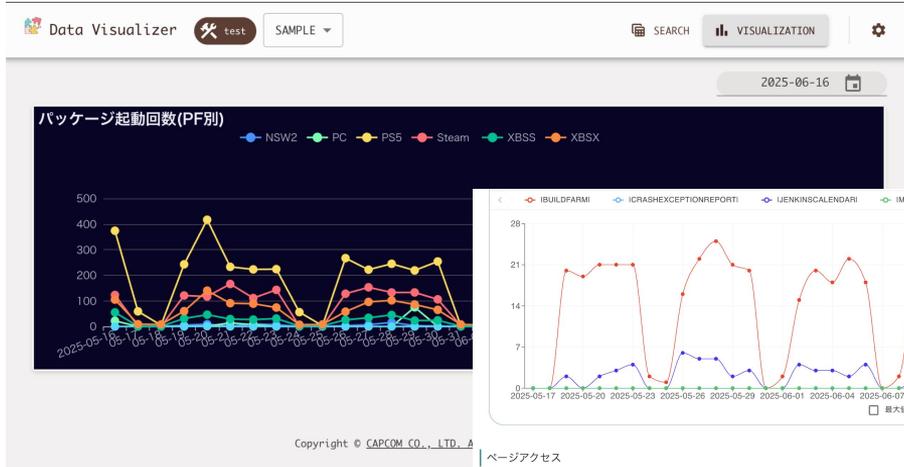


Cloud SQL



Memorystore

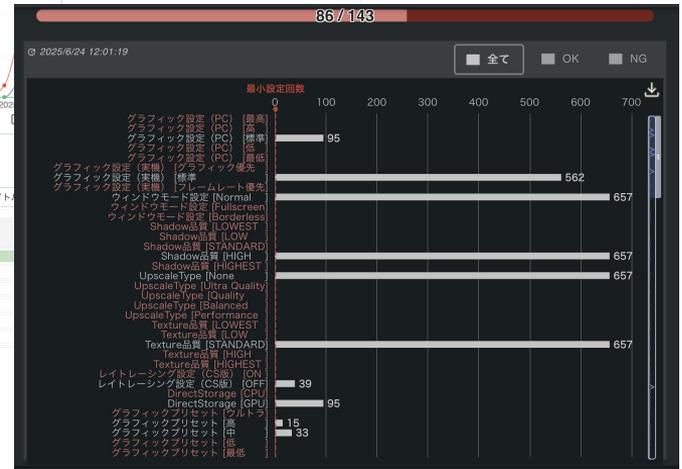
アプリケーションの例



ページアクセス

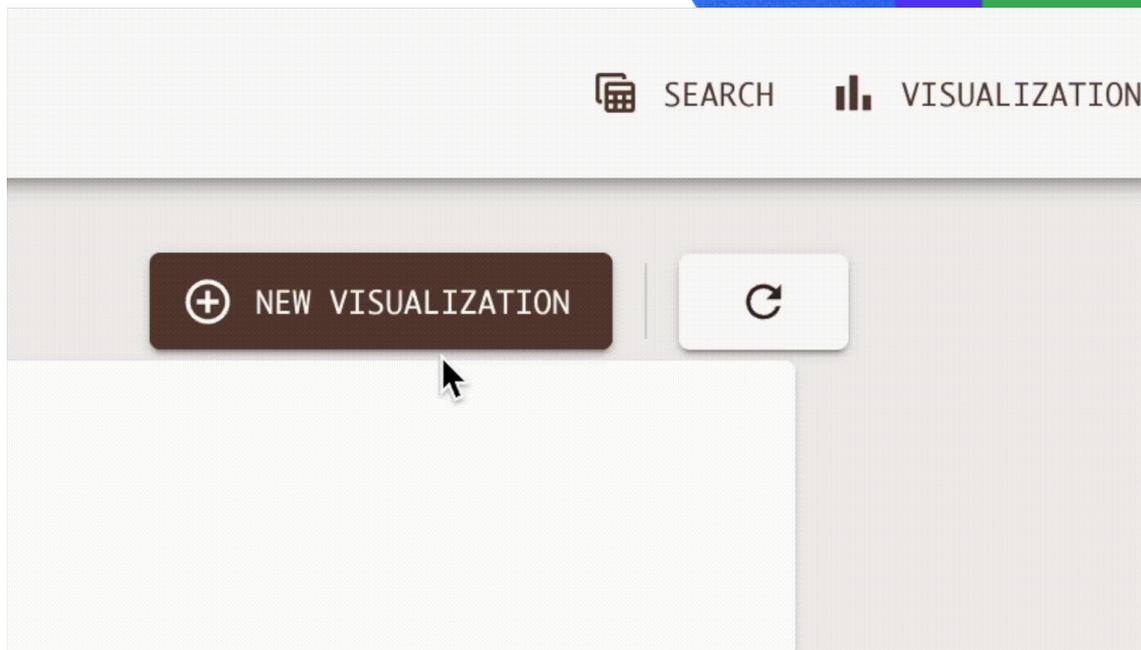
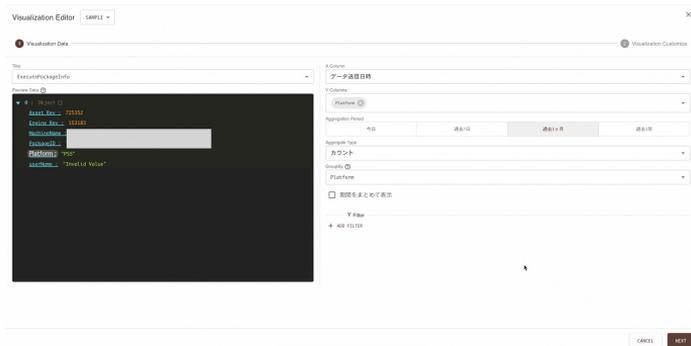
ページアクセスの遷移元、遷移先 (開閉中の合計) (サービスの動作への影響が結果が5%以上のレコードのみ表示) 全タイトル

サイト名	遷移元ページURL	遷移先ページURL	遷移先ページ名	結果	割合
CrashExceptionReport	/exception-crash-repo...	詳細情報ページ		233738	100%
MapService	https://mapservice@...	マップビューシ...		22991	9.67%
MapService	https://mapservice@...	マップビューレイ...		1931	0.78%
CrashExceptionReport	/exception-crash-repo...	詳細情報ページ		787	0.34%
Buildfarm	/login	名前未設定ページ		630	0.27%
CrashExceptionReport	/exception-crash-repo...	詳細情報ページ		448	0.19%
CrashExceptionReport	/exception-crash-repo...	トップページ(通常)		443	0.19%



汎用アプリケーション

GUIによりセルフサービスで簡単なグラフが作成できる。



専用アプリケーション

目的に特化した専用アプリケーション群

ヒートマップフィルタ

計算方法: 定式

マップ選択: [選択済み]

サンプリング: 全てのマシン

プラットフォーム: 全てのプラットフォーム

表示するパルメータ: FPS

グリッドサイズの調整: 26

詳細度の設定: 0.30

スクリーンショットの取得(pps): 90

ヒートマップ情報

詳細

座標: (64.156, 8.22929, 885.5)

FPS: 51.6534FPS

RFPS: 52.2985RFPS

カメラの向き: (0.707107, 0, 0.707106)

MEMORY: 598MB

RESOURCEMEMORY: 12449MB

リビジョン

アセットリビジョン: 654565

エンタナリリビジョン: 147094

マップアクション

目標へ飛ぶ

FPS	RFPS	Mem-Default	Mem-Resource	座標
51.8415	52.5467	60241248	1308991104	(64.156, 10.959, 829.83)
50.9078	51.9827	623410208	1311910192	(64.156, 8.20956, 883.13)
50.9078	51.9827	623410208	1311910192	(64.156, 8.20956, 883.13)
50.9033	51.9065	624231004	1311644224	(64.156, 9.40205, 887.5)
50.9033	51.9065	624231000	1311644224	(64.156, 9.40205, 887.5)
52.5231	53.201	620500008	1311812608	(64.156, 8.67918, 871.5)
52.5231	53.201	620500008	1311812608	(64.156, 8.67918, 871.5)
51.6834	52.2985	627044096	1305603332	(64.156, 8.22929, 885.5)

DASHBOARD

CHECK RESULT

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

86 / 114

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

47 / 83

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

83 / 107

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

28 / 132

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

32 / 47

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

108 / 143

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

3 / 8

2025/6/17 9:59:42

オプション設定回数: 1

最小設定回数: 1

4 / 28

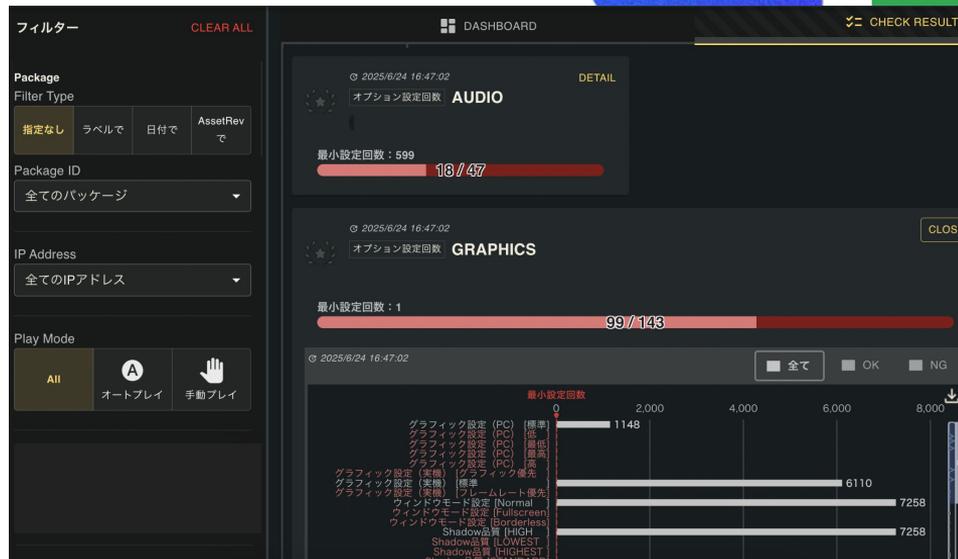


専用アプリケーションの一例

全てのゲーム中の要素が偏りなく
チェックを通っていることを確認する目的のアプリケーション。

実績を集計し、既定値以下の項目を分かりやすく表示。

テストが不十分な箇所を特定、計画の修正などに活かす。



集計結果のレポート生成

集計結果をレポートから大まかな傾向を掴む

LLM のモデルは

Gemini 2.0 Flash を使用。

100 万トークンの入力が可能、数学にも強い、ということで、データの集計結果からレポートを生成する、ということも容易に実現。

DASHBOARD

CHECK RESULT

チェック項目
全体のまとめ

全体のまとめ

- 全体の傾向
今回のテストプレイ結果の傾向としては、各チェック項目において、ある程度の回数がカウントされているものが多いようです。
しかし、一部のチェック項目では、回数が極端に少ない項目や、0回の項目が見受けられるため、今後これらのチェックを網羅的に行うことを検討してみましょう。
特に、回数が0回の項目については、除外対象となっているものも含まれているため、念のため全て除外対象となっているか確認することを推奨します。
- 特に回数が0回のもが目立つチェック項目

項目	詳細	0回の項目数	備考
会話再生チェック	会話再生回数	83件	

【補足】
他にも回数が0回のは見られましたが、詳細は各チェック項目ごとのレポートをご覧ください。

分析レポートについて
質問する
選択されたチェック項目について回答します

気になる内容がありますか？

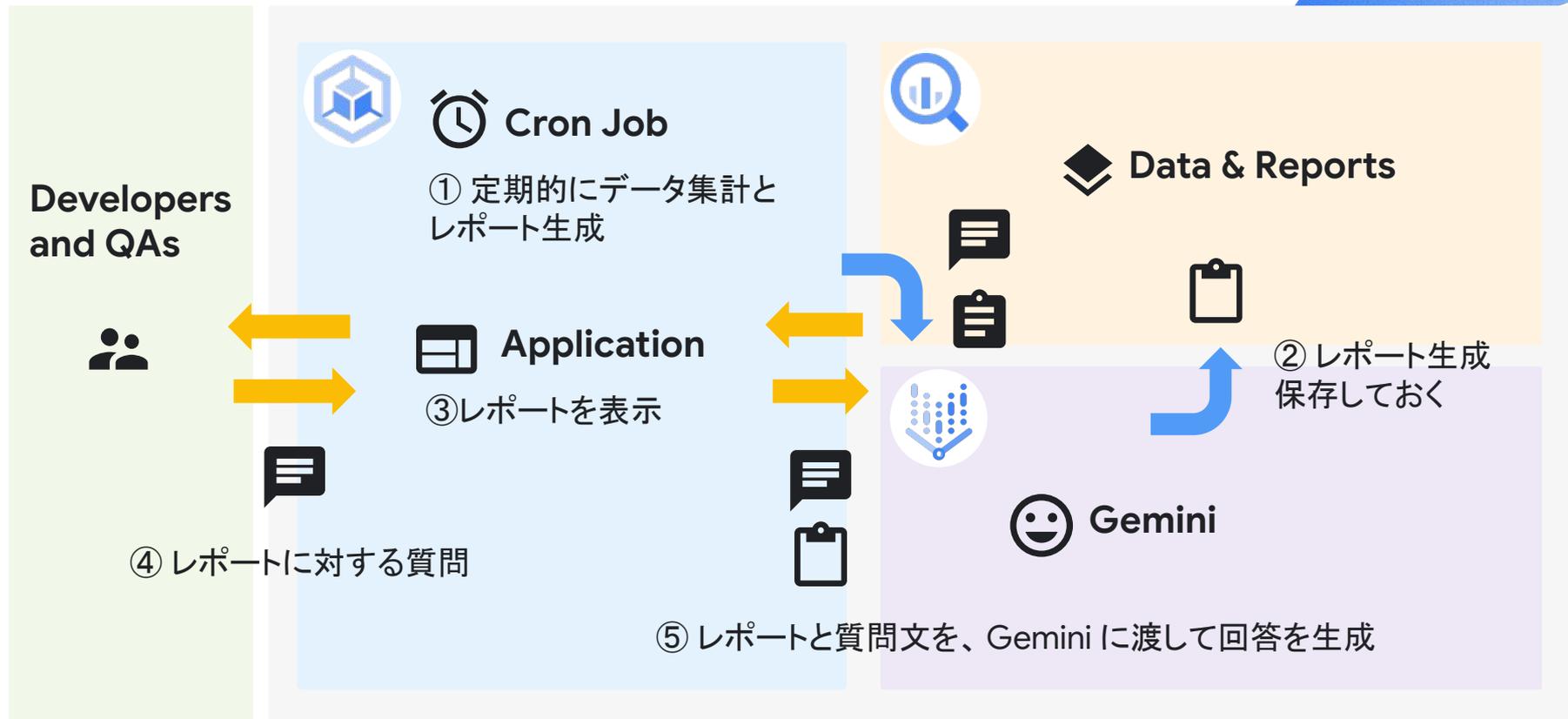
回数が0件のものだけを教えてください

回数が多いものだけを教えてください

回数が少ないものだけを教えてください

質問を入力してください

レポート生成と質疑応答の内部構成



プロンプトで工夫した点



マークダウン形式

マークダウン形式で、構造化して指示する。



フォーマット指定

入出力データのフォーマットを指定。具体例も記載。



ペルソナ設定

レポートを誰に対して提出する立場なのか、ペルソナを設定する。

03. 運用のポイント

活用に至ったポイント

- フルマネージド サービスを活用し、運用構築に割く時間を最低限にできたこと
- その分現場のニーズをヒアリングし、目的に応じたサービスを開発することに注力できたこと
- アプリ部分は内製化し、小回りの効く運用ができたこと

開発スプリント

ヒアリング

利用者の要望を、定例ミーティングなどでヒアリング。

要望の実装

追加要望があった機能の検討、実装。

機能リリース

追加機能のリリース。

ヒアリング

追加機能の使用感、新たな要望などのヒアリング。

繰り返し...

04. 今後の展望

レポート生成やチャット機能の強化

Detailed report creation and Q&A by AI



Make Gemini an MCP client.

By doing so, questions could be answered by chat only from pre-collected results in the future, we think it will be possible for Gemini informator from APIs and generate answers specifying to parameters according to questions.

In the development field, there are times when you want a rough overview, and times when you want to know the precise status of a package created yesterday, and there also needs to know the situation from various perspectives on a case-by-case basis.

MCP サーバー対応

下記の OSS を利用する場合の例

- API サーバーに、FastAPI
- MCP サーバー化に、FastAPI-MCP



出典: <https://fastapi.tiangolo.com/ja/>



出典: https://github.com/tadata-org/fastapi_mcp

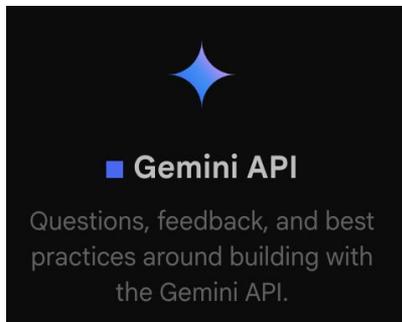
MCP サーバー対応

最低限、追加するコードはこれだけ

```
1 from fastapi import FastAPI
2 from fastapi.middleware.cors import CORSMiddleware
3 from fastapi_mcp import FastApiMCP
4
5 from app.router import router as api_router
6
7
8 def get_application():
9     app = FastAPI(title="AutoPlayVisualizer API", version="1.0.0", docs_url="/autoplay-visualizer/api/docs")
10
11     app.add_middleware(
12         CORSMiddleware,
13         allow_origins=["*"],
14         allow_credentials=True,
15         allow_methods=["*"],
16         allow_headers=["*"],
17     )
18
19     app.include_router(api_router, prefix="/testplay-visualizer/api")
20
21     return app
22
23 app = get_application()
24
25 # MCPサーバーを作成
26 mcp = FastApiMCP(
27     app,
28     name="TestplayVisualizer MCP Server",
29     include_tags=["mcp"],
30 )
31 mcp.mount()
32
```

MCP クライアント

Gemini API SDK を使用



出典: <https://discuss.ai.google.dev/>

出典

: https://ai.google.dev/gemini-api/docs/function-calling?hl=ja&example=weather#model_context_protocol_mcp

Google AI for Developers モデル ソリューション コードアシスタンス もっと見る

検索 /

Gemini API のドキュメント API リファレンス クックブック Community

料金
レート制限
お支払い情報

モデルの機能

- テキスト生成
- 画像の生成
- 動画生成
- 音声生成
- 音楽生成
- 長いコンテキスト
- 構造化出力
- 考えている
- 関数呼び出し
- ドキュメントの理解
- 画像の理解
- 動画理解
- 音声の理解
- コードの実行
- URL コンテキスト
- Google Search

ガイド

- プロンプト エンジニアリング
- ライブ API

Model Context Protocol (MCP)

Model Context Protocol (MCP) は、AI アプリケーションを外部ツールやデータに接続するためのオープン標準です。MCP は、モデルが関数（ツール）、データソース（リソース）、事前定義されたプロンプトなどのコンテキストにアクセスするための共通プロトコルを提供します。

Gemini SDK には MCP のサポートが組み込まれているため、ポイラードコードを削減し、MCP ツールの自動ツール呼び出しを提供できます。モデルが MCP ツール呼び出しを生成すると、Python クライアント SDK と JavaScript クライアント SDK は MCP ツールを自動的に実行し、後続のリクエストでレスポンスをモデルに戻します。このループは、モデルからツール呼び出しがなくなるまで続行されます。

Gemini と mcp SDK でローカル MCP サーバーを使用する方法の例を次に示します。

Python JavaScript

選択したプラットフォームに最新バージョンの mcp SDK がインストールされていることを確認します。

```
pip install mcp
```

★ 注: Python は、ClientSession を tools パラメータに渡すことで、ツールの自動呼び出しをサポートしています。無効にするには、無効な True を指定して automatic_function_calling を指定します。

```
import os
import asyncio
from datetime import datetime
```

MCP クライアント

最小限のサンプルコード

```
1 import asyncio
2 import argparse
3 from mcp import ClientSession
4 from mcp.client.sse import sse_client
5 from google import genai
6 from google.oauth2 import service_account
7
8 credentials = service_account.Credentials.from_service_account_file(
9     scopes=["https://www.googleapis.com/auth/cloud-platform"]
10 )
11 client = genai.Client(
12     vertexai=True,
13     project="developer-datapf-prd",
14     location="us-central1",
15     credentials=credentials
16 )
17
18 server_url = "http://0.0.0.0:8001/mcp"
19
20 async def run(prompt: str):
21     async with sse_client(server_url) as (read, write):
22         async with ClientSession(read, write) as session:
23             await session.initialize()
24
25             response = await client.aio.models.generate_content(
26                 model="gemini-2.0-flash-001",
27                 contents=prompt,
28                 config=genai.types.GenerateContentConfig(
29                     temperature=0,
30                     tools=[session],
31                 ),
32             )
33
34             print("---- Response ----")
35             print(response.text)
36
37
38 if __name__ == "__main__":
39     parser = argparse.ArgumentParser()
40     parser.add_argument("--prompt", type=str)
41     args = parser.parse_args()
42
43     asyncio.run(run(args.prompt))
44
```

docstring を書く

LLM にとって API の 仕様理解に必要

```
@router.get('/{title}/results', operation_id="get_result_list", tags=["mcp"])
async def get_result_list(
    title: str,
    phase_id: int = Query(default=1),
    qa_group: int = Query(default=0),
    category_id: int = Query(default=10),
    platform: str = Query(default=const.PARAM_PLATFORM_ALL),
    package_id: str = Query(default=const.PARAM_PACKAGE_ALL),
    date_range: str = Query(default=const.PARAM_DATE_RANGE_ALL),
    from_date: str = Query(default=None),
    to_date: str = Query(default=None),
    play_mode: str = Query(default=const.PARAM_PLAY_MODE_ALL),
    ip_address: str = Query(default=None),
):
    """
    # 関数の説明
    指定したカテゴリの集計結果を取得します。
    titleの指定は必須です。
    パラメータを指定しなくてもデフォルト値が適用されますので、必要に応じて変更してください。
    # パラメータの説明
    * from_date
    * 型
    |   str
    * 説明
    |   集計期間の開始日を YYYY-MM-DD 形式で指定します。
    |   date_range が CUSTOM_RANGE の場合に有効です。

    * to_date
    * 型
    |   str
    * 説明
    |   集計期間の終了日を YYYY-MM-DD 形式で指定します。
    |   date_range が CUSTOM_RANGE の場合に有効です。

    * play_mode
    * 型
```

補足のプロンプト

API 使用の精度を上げるため
補足の情報なども与える

```
15 |     credentials=credentials
16 | )
17 |
18 | server_url = "http://0.0.0.0:8001/mcp"
19 |
20 | api_param_rules = """
21 | 以下は、汎用的なMCPサーバーのAPIパラメータルールです。
22 | 全てのAPIに必要なものではありませんが、パラメータとそれが意味するところは共通です。
23 | 適宜解釈し、ツール呼び出しのパラメータ生成の際に参照してください。
24 | 指定されているかどうか、判断ができないパラメータは未指定としてください。
25 | API側でデフォルト値が適用されます。
26 |
27 | ### 判例
28 | * title
29 |   * 型
30 |     str
31 |   * 説明
32 |     データの対象となるプロジェクトのタイトル名。
33 |     例: "XXXXXX"
34 | """
35 |
36 | async def run(prompt: str):
37 |     async with sse_client(server_url) as (read, write):
38 |         async with ClientSession(read, write) as session:
39 |
40 |             await session.initialize()
41 |
42 |             response = await client.aio.models.generate_content(
43 |                 model="gemini-2.0-flash-001",
44 |                 contents=prompt + api_param_rules,
45 |                 config=genai.types.GenerateContentConfig(
46 |                     temperature=0,
47 |                     tools=[session],
48 |                 ),
49 |             )
50 |
51 |             print("----- Response -----")
52 |             print(response.text)
53 |
54 | if __name__ == "__main__":
55 |     parser = argparse.ArgumentParser()
56 |     parser.add_argument("--prompt", type=str)
57 |     args = parser.parse_args()
58 |
```

実行結果

```
$ python sse_app.py --prompt 開発チームの手動プレイによる、オプションチェックの集計結果をまとめてください。対象日は2025/5/26から2/25/5/30までとしてください。
----- Response -----
開発チームの手動プレイによる、オプションチェックの集計結果です。対象日は2025/5/26から2025/5/30までです。

*   オプション設定回数：GAME_SETTING

    *   最小設定回数：20
    *   パスした回数：96
    *   チェック項目の総回数：114
*   オプション設定回数：CONTROLS

    *   最小設定回数：20
    *   パスした回数：60
    *   チェック項目の総回数：85
*   オプション設定回数：CAMERA

    *   最小設定回数：20
    *   パスした回数：114
    *   チェック項目の総回数：187
*   オプション設定回数：DISPLAY

    *   最小設定回数：20
    *   パスした回数：89
    *   チェック項目の総回数：132
*   オプション設定回数：AUDIO

    *   最小設定回数：20
    *   パスした回数：40
    *   チェック項目の総回数：47
```

LLM が自分で判断して、API をコール。
さらにプロンプトの情報から、API のパラメータも生成。
そして得られた結果を要約している。

パラメータの確認

【プロンプト】

開発チームの手動プレイによる、オプションチェックの集計結果をまとめてください。**対象日は 2025 / 5 / 26 から 2025 / 5 / 30 までとしてください。**

API 側で受け取ったパラメータのログ

```
backend_app-1 | INFO: 172.18.0.1:57304 - "POST /mcp/messages/?session_id=c758dacce4f94906b6733ced5017fb97 | HTTP/1.1" 202 Accepted  
backend_app-1 | parameters: title=██████, phase_id=1, qa_group=0, category_id=201, platform=All, package_id=All | date_range=CUSTOM_RANGE, from_date=2025-05-26, to_date=2025-05-30, play_mode=manual
```

拡大

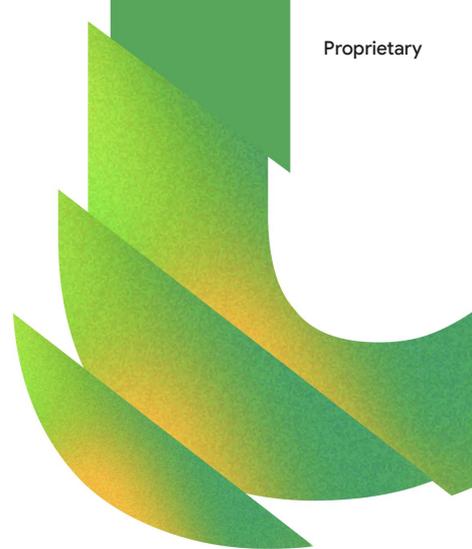
```
TTP/1.1" 202 Accepted  
date_range=CUSTOM_RANGE, from_date=2025-05-26, to_date=2025-05-30
```

より柔軟なやり取りが可能に

- 「昨日作成した」パッケージの動作は安定してる？
- 過去一週間で不具合の件数は「減少傾向」なのか？
- 「バージョン ???」の「〇〇ステージ」の状況を知りたい

等々...

そして、そこから続けて質問をしていくことで
より深掘りした詳細な分析が可能に...



全ての情報へ統合的なアクセスが可能に



05. まとめ

まとめ

1

マネージド サービスを活用し、迅速にアプリを構築。

2

LLM の活用により、専門家でなくてもデータから状況を読み取れるように。

3

今後、さらなる AI の発展に期待。そして、さらにクリエイティブ領域の時間を確保し、より面白いゲームを。