



# チーム開発の進め方と 便利ツール

2021年度 株式会社サイバーエージェント エンジニア 新卒研修  
AI事業本部 コピー素材開発 脇本 宏平

# 脇本 宏平

19新卒

AI事業本部 > 極事業部 > コピー素材開発

Data Scientist

 @KoheiWakimoto



💡 このスライドはNotionで作成した新卒向け講義資料をほぼそのまま添付したものです。  
公開にあたって実例や社内情報など一部内容を省いてます。

# この資料について

- 💡 このドキュメントは21エンジニアの1ヶ月のチーム開発研修の最初に行う5つの講座のパート2「チーム開発の進め方と便利ツール」の資料です。

## 講座スケジュール

### 1日目

- 講座1. チームビルド
- 講座2. チーム開発の進め方と便利ツール ← ここ
- 講座3. 去年の新卒研修を終えて

### 2日目

- 講座3 DevOps
- 講座4 良いコードとは何か

### 8日目

- 講座5 エンジニアとしてのプロダクト貢献

# この講座の目的

1. 全員がスムーズにチーム開発に入れる状態にする
2. 良いチームづくりについて考える際の助けになる
3. 研修終了後の振り返りの反省材料になる

# チーム開発の進め方

## 2部構成

1. チーム開発の流れ
2. チーム開発の要素と便利なツール

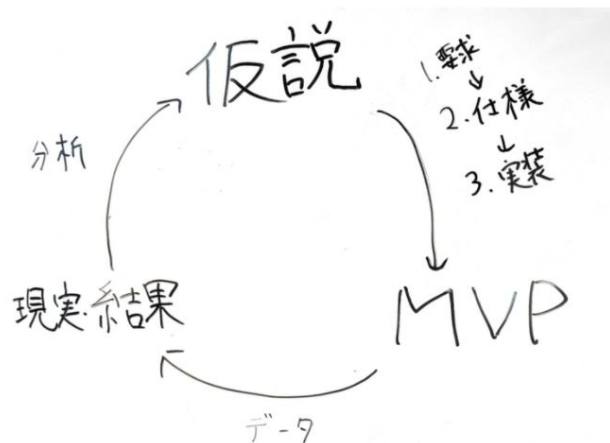


AI×広告×クリエイティブ制作の開発者の主観をたくさんいれます

# チーム開発の流れ

1. 仮説を立てる(要求分析・要件定義)
2. 準備する(仕様定義, 技術選定, 設計, 役割分担, タスク設定, 工数見積もり)
3. 実行する(実装→テスト→レビュー→リリース)

→ 1.にもどる



# 1. 仮説を立てる(要求分析・要件定義)

開発の目的 = プロダクトの価値を上げる。新規なら0から価値を作る。

仮説 = プロダクトの価値を上げる可能性のあるもの。市場に投下しフィードバックを得て、価値の変化を観測するまで確定しない。

## プロセス

要求を分析

- ビジネス課題やビジネスチャンス

要件を決める

- 要求に対するアプローチ
- 顧客の要望、市場調査、データ分析、課題分析などに基づき「どんなものを」「なぜ」つくるのか決める
- 複数ある要求の中からビジネスインパクトや工数に基づく優先度を考えて「いつ」つくるのか決める

## 仮説設定からエンジニアが参加する必要性

- 技術が絡む領域について、何が仮説になり得るか網羅的に発想し判断
- 現実には仮説の数に対してエンジニアが足りている状態はほぼない  
→ 無数にある仮説の中からコストやクオリティなどを考慮して絞り込む



・ 本研修では必須ではないですが、要件を考える際には是非意識してみてください



## 2. 準備する(仕様定義, 技術選定, 設計, 役割分担, タスク設定, 工数見積もり)

- 要件を「どう」実現するか具体化するフェーズ
- 以下の作業を並行して行う
- **仕様定義**
  - UI/UX、作る機能(利益>コスト)、作らない機能(利益<コスト)など
- **技術選定・設計**
  - アーキテクチャ、インフラ、言語など
- **役割分担**
  - 開発の参加者を決める。必要に応じてアサインする。

- **タスク設定**

- 誰がいつ何をするか

- **工数見積もり**

- 実現可能かどうか作る前に判断するために行う→徹夜しなくて済む。(自戒)
  - 工数=人×時間。5人で10営業日→50人日以上タスクはこなせない。
- 実行する価値があるか判断する
  - 利益 > コスト → やる
  - 利益 < コスト → やらない

💡 作って終わりではないので、運用にかかるコストも見積もった上で設計や技術選定を行う

### 3. 実行する(実装→テスト→レビュー→リリース)

- 実装
  - コーディングに全集中
- テスト
  - 実装したものが想定通りか確認
  - テストについては **講座4 品質（テスト・より良いコードとは）** で詳しく紹介
- レビュー
  - 実装したものが想定通りのものか開発者同士でお互いに確認し合う
  - 何かしらの問題があれば、実装に戻り修正をする
  - 問題がなくなるまでこれを繰り返す
- リリース・運用
  - MVPを市場に解き放ち、やきとりセンターで打ち上げをし、1.に戻る

💡 ・実際には実装の過程で設計の問題が発覚したり、競合のリリースや法律の改定などで市場が変化したりするので柔軟な対応が必要  
1,2が正しいか常に疑う(詳しくは講座5で)

・TDD(テスト駆動開発)なチームでは実装とテストが同時に進むor逆になることもある(弊社チームではこれ)

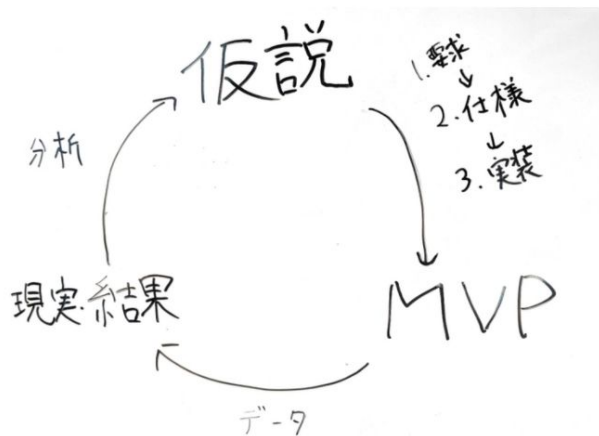
・1~3のサイクルのプラクティスは講座3 DevOpsで詳しく紹介

# MVPをリリースする

MVP（Minimum Viable Product）：要件を満たす最小限のプロダクト

1~3のプロセスを最短で実行し素早く「結果」を観測

より低コストにより良い「仮説」にアップデートする



# チーム開発の要素と便利なツール



研修で意識して欲しい要素とツールを紹介

## 要素たち

- Git運用とレビュー
- 見積もり
- マネジメント
- コミュニケーション
- ドキュメント

# Git運用とレビュー

## 心構え

- 1分でもはやく失敗/認識ズレに気づく
  - そのためにチームでルールを決めておく
    - こまめにレビューしてもらう
      - Reviewerは批判でなく改善のためのコメントを
      - 語尾に「かも」をつけた方が陰悪になりにくいかも (by ベストエンジニア)
  - Reviewerの負担を最小化するブランチ名・コミットメッセージ・PR
    - テンプレート機能。意図を反映したチェックリスト
  - PRのスコープを小さく
- トランクベース開発で実装→レビューのサイクルを速く(2日目の講座で触れます)

# 見積もり

## 個々のタスクの妥当な見積もり

- 例: PERT法 (楽観値+4×標準値+悲観値)/6
- バッファは必要
  - 特に新規チーム不確実性が高いので多めにとる(研修では5日くらいはバッファにしておくのが良さそう)
- 経験・失敗しながら精度を上げる

## クリティカルパスを意識する

クリティカルパス：タスクの依存関係グラフの最長経路→プロジェクト全体のスケジュールを支配する

# マネジメント

## マネージャーの役割(エンジニアリングマネージャー)

チームの生産性を高く保つためのあらゆる行動

- 目標設定
- タスク管理・スケジュール管理
- メンバーの評価(今回は触れない)
- メンタルケア(今回は触れない)

Googleのre:Workは参考になります

### Google re:Work - マネージャー

マネージャーは非常に重要な役割であり、従業員の業績に大きな影響を与えます。優れたマネージャーの条件を共有し、能力開発の機会を提供し、優れたマ

 <https://rework.withgoogle.com/jp/subjects/managers/>

# re:Work

「働く」をもっと良くする先進事例・研究・アイデア

Google



# 目標設定

- 「生産性が高い」とは→チームが目標達成に近づいている
- 目標設定
  - 計測可能な目標がない状態で生産性は存在し得ない

## OKR(目標と成果指標: Objectives and Key Results)

Google re:Work - ガイド: OKRを設定する

OKR の策定方法は状況により異なりますが、最初に組織の目標を表明しておく  
と、チームや個人がそれを考慮して自分たちの目標を設定できます。この方法

 <https://rework.withgoogle.com/jp/guides/set-goals-with-okrs/steps/introduct...>

re:Work

「働く」をもっと良くする先進事例・研究・アイデア

Google

# 目標設定

- 「生産性が高い」とは→チームが目標達成に近づいている
- 計測可能な目標設定が必要

## OKR(目標と成果指標: Objectives and Key Results)

Google re:Work - ガイド: OKRを設定する

OKR の策定方法は状況により異なりますが、最初に組織の目標を表明しておく  
と、チームや個人がそれを考慮して自分たちの目標を設定できます。この方法

 <https://rework.withgoogle.com/jp/guides/set-goals-with-okrs/steps/int...>

re:Work

「働く」をもっと良くする先進事例・研究・アイデア



**Objective:** ストレッチ目標

**Key Result:** 目標へ近づいたことを確認する指標となる行動や状態

決める順番：

1. チームのObjective
2. チームのKey Result
3. チームのKey Result達成に関わる個人のObjective
4. 個人のKey Result

**Objective:** ストレッチ目標(厳しい目標)

**Key Result:** 目標へ近づいたことを確認する指標となる行動や状態

決める順番：

1. チームのObjective
2. チームのKey Result
3. チームのKey Result達成に関わる個人のObjective
4. 個人のKey Result

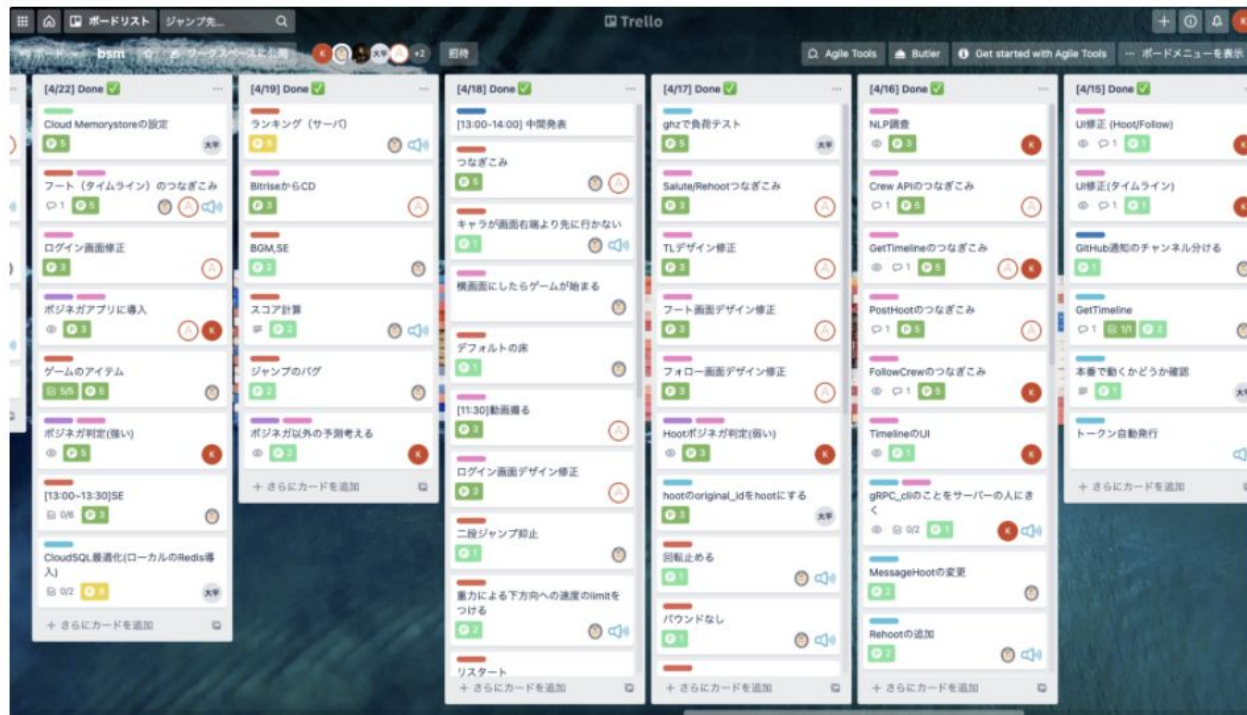
# タスク管理・スケジュール管理

何もしなければ見積もった計画からどんどんずれていくので、ずれないための努力が必要  
便利ツールを紹介

# Trello

## チケット(タスク)管理ツール

「公開」に注意



タイムラインのアイコン表示

in list Backlog

MEMBERS

UI

DUE DATE

Apr 11 at 9:46 AM

STORY POINTS

2

ADD TO CARD

Members

Labels

Checklist

Due date

Attachment

Cover

POWER-UPS

Story Points

Add Power-Ups

BUTLER

Add button

ACTIONS

Move

Description

Edit

完了条件

- タイムラインにユーザーアイコンを表示する

Attachments

LINK

http://UIイメージ

Added 2 minutes ago - Comment - Remove - Edit

Add an attachment

進行状況

Hide checked items

Delete

50%

ダミー画像表示

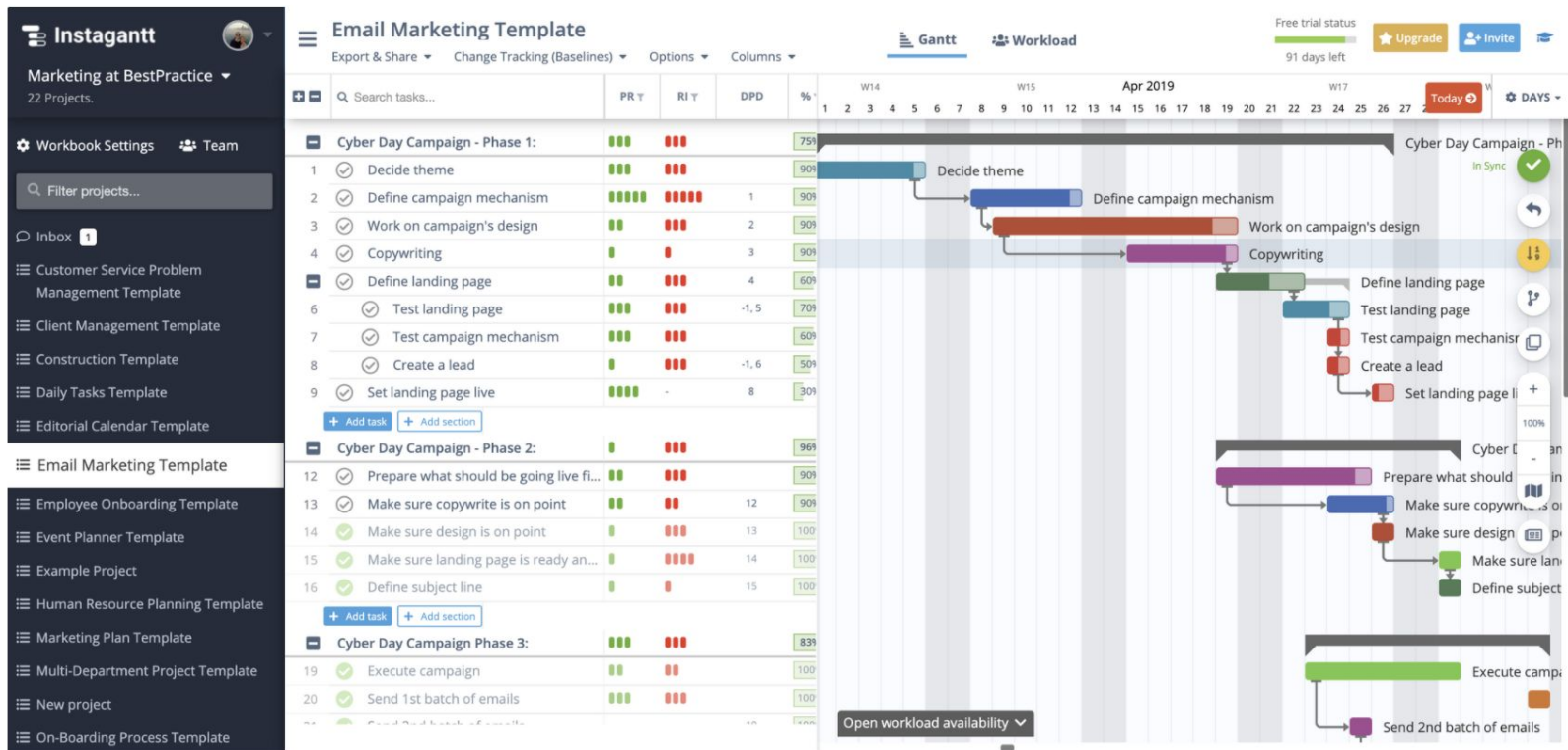
ユーザーアイコン表示

## おすすめの拡張機能 Story Points(from Agile Tools)

- Point→工数をざっくり表したもの
- 毎日・毎週の振り返りでどれくらいタスクを消化したかわかりやすい→次のイテレーションを見積もりやすい  
「1週目で50ポイント。3週目までに残り200くらいか。スケジュール厳しいからスコープ見直そ」
- 長く続けていくうちに1ポイントの価値が変動していくので注意

# Asana + Instagantt

チケット管理とガントによる可視化のUXが優秀(普段はこれつかってます)



## Notion

ドキュメントツールとしてNotionを使う場合はNotionのガント( `/timeline` )を使っても良い

### Copy of スケジュール

			<div>March 2021</div>		<div>April</div>															<div>Month</div>		<div>&lt;</div>
<div>Aa</div> Name	<div>Assign</div>	<div>Status</div>	29	30	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		
機能1 実装	<div>宏平 脇本</div>	Completed			機能1 実装	<div>Completed</div>																
機能2 実装	<div>宏平 脇本</div>	In progress						機能2 実装	<div>In progress</div>													
テスト	<div>宏平 脇本</div>	Not started										テスト	<div>Not started</div>									
<div>+ New</div>																						

COUNT 3



# コミュニケーション

## 相互理解

- 互いのスキルセットを把握することで
  - 相談相手を見つけやすくなり課題解決が加速
  - 思いついたアイデアについてチームで実現可能かどうか素早く判断できる
- 互いの目標・興味を把握することで
  - よりモチベーションの高いタスク分担ができる→質とスピードの向上
  - 互いにより良いアドバイス・フィードバックができ、成長につながる

性格が合うかに関係なく 意図的に興味を持つ (by SOYAMAN)

## 心理的安全性を高める

心理的安全性の高さで生産性は全く違う

「対人関係においてリスクのある行動をしてもこのチームでは安全であるという、チームメンバーによって共有された考え」

- 「わからない」「できない」「ミスをした」とメンバーに伝えても非難されない(指摘と非難は違う)
- メンバーやチームの方針や意見に指摘をしても非難・拒絶されない(互いに本音で議論できる)

相互理解が甘い段階だと指摘のつもりが非難に聞こえることもある

コツ：1. **まず肯定**し、2. **相手のために助言**する

Google re:Work - ガイド:「効果的なチームとは何か」を知る

Google のリサーチチームが発見した、チームの効果が高いチームに固有の 5 つの力学のうち、圧倒的に重要なのが心理的安全性です。リサーチ結果によると、心理的安全性の高いチームのメンバーは、Google からの離職率が低く、他のチームメ

 <https://rework.withgoogle.com/jp/guides/understanding-team-effectiveness/steps/foster-psychological-safety/>

# re:Work

「働く」をもっと良くする先進事例・研究・アイデア

Google

## 毎週金曜に確認してみよう

自分のチームは1~7のどれか？

1. チームの中でミスをする、と、たいてい非難される。
2. チームのメンバーは、課題や難しい問題を指摘し合える。
3. チームのメンバーは、自分と異なるということを理由に他者を拒絶することがある。
4. チームに対してリスクのある行動をしても安全である。
5. チームの他のメンバーに助けを求めることは難しい。
6. チームメンバーは誰も、自分の仕事を意図的におとしめるような行動をしない。
7. チームメンバーと仕事をするとき、自分のスキルと才能が尊重され、活かされていると感じる。

## 他チームとのコミュニケーション

- CAでは他チームに知見がころがってる
- Slack、Notionを活用してチームの課題と近いことをやってるエンジニアを探してみよう

## Technology Mapを使って見よう

### Technology Map

好きな技術や気になるプロダクトについて調べてみましょう(2分)

- CAのエンジニアはこわくないのでいきなりDM送っても大丈夫。ランチに誘ってみよう。
- チームの知見はチーム外にも発信して助け合おう

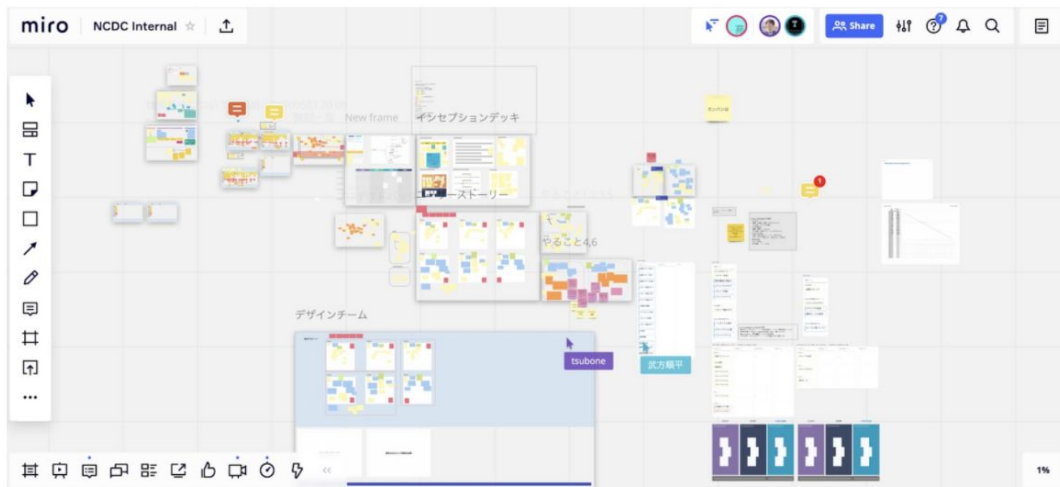
# リモートでのコミュニケーション

チームの一体感が圧倒的に減るので工夫する

弊社チームでは毎日15分の勉強会&昼と夕方の相談会を実施してます

## ツールを活用

- 雑談: [discord](#)(音声主体で気軽)
- ホワイトボード: Miro(広くて機能がリッチなのにサクサク)



拾い物

## テキストベースの会話では感情をオーバーに

- 送り手の想定以上に読み手はネガティブに受け取る
- とりあえず **!** つけとくだけでも良い！
- スタンプを積極的に使おう。追加もできる。



:wakimoto:

## 社内での事例共有イベント「パクリモ」

イベントページにアーカイブがあります

# ドキュメント

- 「人は忘れる」ドキュメントを書こう

- あれ結局どうなったんだっけ？→時間の無駄
- Tips:
  - 棄却されたアイデアも棄却した理由とともに残そう
  - 考慮し忘れたわけではないことがわかる
  - 判断理由をステークホルダーに説明できる
  - 状況が変わり棄却理由がなくなるかも

- 「人は誤解する」ドキュメントを書こう

- MTGでの決定は誤解や抜け漏れがある、関係者間で整合性の指摘ができるように文書化しよう

# テクニカルライティング

- 技術者が良い文書を書く為のテクニック
- Googleエンジニアの記事(1時間で読める)

## Technical Writing | Google Developers

We've aimed these courses at people in the following roles: professional software engineers computer science students engineering-adjacent roles, such as product

 <https://developers.google.com/tech-writing>

Google Developers

- 超要約すると「文書は読まなければならない & 正しく伝わらなければならない。」
  - 読み手の知らない単語や誤解を生む曖昧な表現を使わない
  - 冗長に書かない。箇条書きを活用。ただし簡潔さより明確さを優先
  - などなど

# Notion

## Tips

 [↗公式ヘルプ 日本語版 | Npedia](#) になんでものってる

## 一部紹介

データベース `/table`

- Linked Database

 [定例](#)

- 公式テンプレート集

 [↗Notion Template Gallery](#)

- 自作ページをテンプレートとしてプロジェクトに登録できる



一部紹介

データベース /table

- Linked Database

 定例




開発

 Name	 Tags	 Date
DBスキーマ	db サーバー	Apr 12, 2021
2021/04/10 定例	定例	Apr 10, 2021
2021/04/09 定例	定例	Apr 9, 2021
ログの分析	DS	
UIデザイン	UI フロント	
CI/CD	インフラ	
+ New		

COUNT 6

定例

開発

 Name	 Date	 Tags
2021/04/10 定例	Apr 10, 2021	定例
2021/04/09 定例	Apr 9, 2021	定例

# まとめ

せっかくの機会なので「良いチーム開発」について自分で仮説を立てて色々実験してみてください。

---

著者：脇本 宏平

協力：川本 峻頌

監修：渡部 優