

# Gemsrack

Slackで作る、Bizチームのための小さな自動化「Gem」

**作る** (テンプレ/AI) → **実行** (/gem) → **共有** (再利用)

対象: 営業 / CS / マーケ / 企画 / BizOps (Slackで仕事が回っているチーム)

## このスライドで伝えたいこと

---

- Gemsrackは、Slack上で“業務の型”をGemとして資産化し、誰でも同じ品質で実行できる仕組み
- Bizのよくある作業（文章/要約/整形/チェック/テンプレ返信）を最短で自動化
- 導入は軽く、運用は安全に（公開/非公開・権限・ログ・保存）

キーワード: **再現性 / 標準化 / Slack起点 / 小さく始めて育てる**

## ありがちな課題（Biz現場）

---

- ・同じ作業の繰り返し: 返信文作成、議事録要約、提案書の叩き台、FAQ整形
- ・属人化: “できる人”の言い回し/手順がチームに残らない
- ・品質のばらつき: 新人・兼務・繁忙でアウトプットが揺れる
- ・ツールの分断: いろいろ散らばり、結局Slackに戻って手作業

狙いは「時間短縮」だけでなく「品質の平準化」「ナレッジの再利用」

## Before / After (現場の変化)

---

### Before (手作業)

- ・ 「誰かのやり方」を都度聞く / 過去ログを探す
- ・ 文章/要約の品質が人によってブレる
- ・ レビュー/差し戻しが増え、結局時間が溶ける

### After (Gemで標準化)

- ・ “仕事の型”をボタン1つで再実行 (/gem)
- ・ チェック観点を埋め込めるので品質が揃う
- ・ 新人でも同じレベルに寄せやすい（オンボーディング短縮）

## Gemsrackがやること

---

- SlackのSlashコマンドで **Gem**（小さな自動化）を作成・実行
- Gemは **静的テキスト** でも **AI（構造化）** でもOK
- 実行結果は **自分だけ**（DM）にも **チーム共有**（チャンネル）にも出せる
- Web UIで **Gem一覧を棚卸し**（どれが使われているか把握しやすい）

“AIチャット”と違い、毎回お願いの仕方を考えなくても、**決まった入力→決まった出力**で動きます。

## Gemとは（業務の「型」）

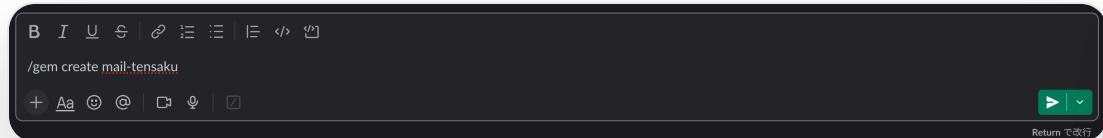
---

Gemは、Biz作業を“再現可能な手順”に落としたものです。

- **入力:** 何を渡す？（例: 商談メモ、問い合わせ文、要点箇条書き）
- **処理:** どう考える？（例: 目的、トーン、制約、チェック観点）
- **出力:** 何を返す？（例: 返信テンプレ、要約、ToDo、表形式）

**ポイント:** Gemは「個人の便利」から「チームの標準」へ育てられる

# 実画面 (Gem作成)



1) まずは Slack で `/gem create <name>` を入力

The screenshot shows the "Create Gem" configuration interface. At the top, there's a logo for "GEMSLACK" and a title "Create Gem" next to a close button (X). Below the title, there are several input fields and sections:

- 概要 (1行) (任意)**: A text input field containing "メールの添削".
- システムプロンプト (任意)**: A large text area with the following text:

あなたは「ビジネスメール添削の専門家AI」です。  
ユーザーが入力したメール文を、目的・相手・文脈を尊重しつつ、  
日本のビジネス慣習に即した、自然で信頼感のある表現に添削して  
ください。
- 【基本方針】**: A list of guidelines:
  - ユーザーの意図・トーン（丁寧／カジュアル／緊急／謝罪／依頼など）を維持する
  - 内容の事実関係や主張は変更しない
- 入力形式 (任意)**: A dropdown menu currently set to "自由記述 (テキスト)".
- 出力形式 (任意)**: A dropdown menu currently set to "テキスト (平文)".

**実画面（長文・改行入力で実行）**

B I U S | ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉

/gem run mail-tensaku

○○株式会社  
○○様

いつも大変お世話になっております。  
株式会社△△の高木です。

本日は、先日お打ち合わせにてご依頼いただきました資料をお送りいたします。  
お手数ですが、ご確認のほどよろしくお願ひいたします。

内容につきまして、ご不明点や修正のご要望等がございましたら、  
お気軽にお知らせください。

引き続き、どうぞよろしくお願ひいたします。

-----  
株式会社△△  
高木 悠人  
Email: xxxx@xxxx  
Tel: 00-0000-0000  
-----

+ Aa ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉

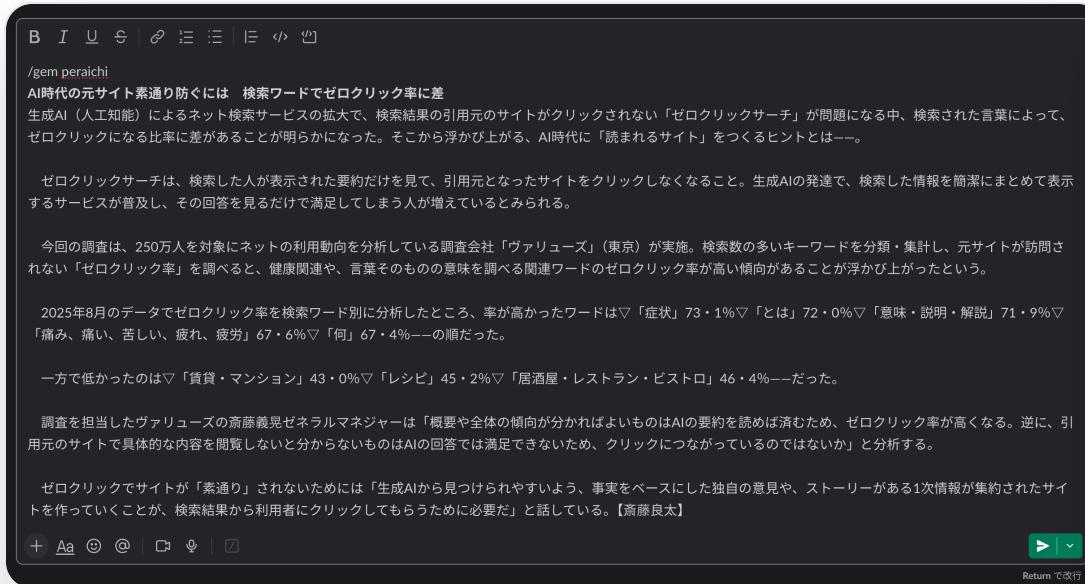
○ あなただけに表示されています  
○ 株式会社  
○ ○様  
いつも大変お世話になっております。  
株式会社△△の高木です。  
先日はお打ち合わせいただき、誠にありがとうございました。  
ご依頼いただきました資料を添付いたしましたので、ご査収ください。  
内容にご不明な点や修正のご要望などございましたら、  
お気軽にお申し付けください。  
引き続き、どうぞよろしくお願ひいたします。

-----  
株式会社△△  
高木 悠人  
Email: xxxx@xxxx  
Tel: 00-0000-0000  
-----  
--  
\*\*主な修正ポイント\*\*  
\* 「先日お打ち合わせにてご依頼いただきました資料をお送りいたします。」を「先日はお打ち合わせいただき、誠にありがとうございました。ご依頼いただきました資料を添付いたしましたので、ご査収ください。」に変更し、打ち合わせへのお礼と資料添付の旨を明確にしました。「お送りいたします」よりも「添付いたしました」の方が具体的です。「ご確認のほどよろしくお願ひいたします」よりも「ご査収ください」の方が、ビジネスメールとして適切です。  
\* 「お気軽にお知らせください」を「お気軽にお申し付けください」に変更し、より丁寧な表現にしました。

出力は「本文 + 修正ポイント」など、型に寄せられる

入力は複数行OK（メール本文をそのまま貼る）

# 実画面（ペライチ生成：文章 → 図解画像）



長い記事やメモから“1枚で伝わる”要点を抽出

The screenshot shows a mobile application interface for Gamsrack. The title is "AI時代の『ゼロクリックサーチ』対策：読まれるサイトを作るヒント". The content discusses the problem of low click-through rates for AI-generated content and provides tips for creating readable sites. It compares two types of search word tendencies: those with high zero-click rates (high-level satisfaction) and those with low zero-click rates (specific information needed).

ワード傾向	検索ワード	率
ゼロクリック率が【高い】ワード傾向 (概要で満足)	「症状」	73.1%
ゼロクリック率が【高い】ワード傾向 (概要で満足)	「とは」	72.0%
ゼロクリック率が【高い】ワード傾向 (概要で満足)	「意味・説明・解説」	71.9%
ゼロクリック率が【高い】ワード傾向 (概要で満足)	「痛み、疲れ」など	67.6%
ゼロクリック率が【高い】ワード傾向 (概要で満足)	「何」	67.4%
ゼロクリック率が【低い】ワード傾向 (具体情報が必要)	「賃貸・マンション」	43.0%
ゼロクリック率が【低い】ワード傾向 (具体情報が必要)	「レシピ」	45.2%
ゼロクリック率が【低い】ワード傾向 (具体情報が必要)	「居酒屋・レストラン」	46.4%

【結論】素通りを防ぐサイト作りのポイント

AIに見つけられやすい『1次情報』の発信が重要！  
事実に基づく独自の意見やストーリーを集約した  
コンテンツを目指す。

# 実画面（Web UI : Gem一覧）

**Gemsrack**  
Gem一覧 (Slack外から閲覧)

Team ID (任意 / 単一WSなら空でもOK)  
T0AD0N5QMHQ

適用 検索 name / summary で絞り込み

更新

**Gem一覧** 3件

- mail-tensaku**  
メールの添削  
2026/2/10 15:35:43  
**in: free\_text** **out: plain\_text**
- peraichi**  
ペライチ生成(system promptは門倉さんのものを利用させていただいてます！)  
2026/2/10 14:33:22  
**in: free\_text** **out: image\_url**
- test1**  
画像生成  
2026/2/10 14:14:29  
**in: free\_text** **out: image\_url**

**詳細**

**name**  
**mail-tensaku**

**summary**  
メールの添削

**system\_prompt**

あなたは「ビジネスメール添削の専門家AI」です。  
ユーザーが入力したメール文を、目的・相手・文脈を尊重しつつ、  
日本のビジネス慣習に即した、自然で信頼感のある表現に添削してください。

**【基本方針】**

- ユーザーの意図・トーン（丁寧／カジュアル／緊急／謝罪／依頼など）を維持する
- 内容の事実関係や主張は変更しない
- 冗長・不自然・失礼に見える表現を改善する
- 読み手（上司・取引先・顧客・社内メンバー）に配慮した文章にする

**【添削観点】**

- 敬語・謙譲語・丁寧語の正確さ

**in: free\_text** **out: plain\_text**

/api/gems から取得しています

## 代表ユースケース（すぐ効く）

---

- ・ **営業:** 商談メモ→要点/次アクション/リスク、提案骨子、メール草案
- ・ **CS:** 問い合わせ文→切り分け、一次回答テンプレ、エスカレ要約
- ・ **マーケ:** 施策案→仮説/検証設計、SNS文面のバリエーション、LP要約
- ・ **BizOps/企画:** 仕様の要点抽出、レビュー観点チェック、定例議事録の整理

向いている作業: 繰り返す / 型がある / 品質を揃えたい / Slackで完結させたい

## ユースケース深掘り（営業）

---

### 入力（例）

- ・商談メモ（箇条書きでもOK）
- ・顧客の背景 / 課題 / 現状の運用 / 競合 / 次回までの宿題

### 出力（例）

- ・要点（3行） / 次アクション（担当者/期限） / リスク（優先度付き）
- ・提案骨子（見出し構成） / フォローアップメール草案（丁寧トーン）

営業の「思考の順番」をGemに閉じ込めると、再現性が上がります。

## **ユースケース深掘り（CS）**

---

### **入力（例）**

- 問い合わせ本文 + 事象の発生条件 + 影響範囲 + 直近変更点

### **出力（例）**

- 切り分け質問（優先順）/一次回答テンプレ（丁寧&簡潔）
- エスカレーション要約（開発が欲しい情報の形式で）

CSは「聞くべき項目」を漏れなく出すだけでも大きく改善します。

## 実行イメージ (Slack)

---

- 一覧: `/gem list`
- 実行: `/gem <name>` または `/gem run <name>`
- 詳細: `/gem show <name>`
- 共有して実行: `/gem <name> --public` (結果をチャンネル投稿)

Slack内で完結するので、「使われ続ける」が起きやすい

## 作り方（2種類）

---

### 1) まずは最短（静的テキスト）

- 例: /gem create hello おはようございます！
- 例: /gem create faq-ask 「状況を教えてください」テンプレ...

### 2) 型を作る（AI Gem）

- /gem create <name> でモーダル
- 概要 / システム / 入力形式 / 出力形式 を保存して“再現性”を担保

おすすめ: まずテンプレGemで「何が繰り返し作業か」を特定し、次にAI Gemで品質と速度を上げる。

## AI Gemが強いポイント (Biz向け)

---

- 出力の型を固定できる (例: 箇条書き・表・JSONなど)
- チェック観点を埋め込める (例: 法務/ブランド/NGワード/事実確認)
- トーンを統一できる (例: 丁寧/簡潔/社内向け/顧客向け)

“毎回AIにお願いする”ではなく、**使える形にパッケージ化**する

## AI Gemの「型」例（出力形式を固定）

---

例えば「商談要約Gem」の出力を固定すると、レビューしやすく、他の人も使いやすい。

- 要点（3行）
- 課題（箇条書き）
- 提案の方向性（3案）
- 次アクション（担当/期限）
- 確認事項（質問）

型があるほど、チームは“**出力を使い回す**”文化になりやすい

## 期待できる効果（例）

---

- ・ **時間削減:** 文章作成/要約/整形のリードタイム短縮
- ・ **品質の平準化:** ばらつきが減り、レビュー負荷が下がる
- ・ **オンボーディング短縮:** 新人でもGemで標準アウトプットに寄せられる
- ・ **ナレッジの棚卸し:** “よく使う型”が一覧化され、改善が回る

## 運用・ガバナンス（安心して使う）

---

- **公開/非公開:** `--public` でチャンネル共有、通常は個人DMで実行も可能
- **権限とスコープ:** Slack Appのスコープは必要最小（例: `commands`, `chat:write`）
- **保存先:** Cloud Run運用は Firestore（推奨）、ローカルはメモリフォールバック
- **拡張性:** コマンド/イベントを追加して運用に合わせられる

「公開投稿は必要なときだけ」「通常は個人DMで試す」運用にすると、チームの心理的安全性も保てます。

## システム構成（ざっくり）

---

- 図: 全体像 (差し替えOK)
- **Slack**: `/gem` コマンド・イベント受信
- **Cloud Run**: Python/FlaskのBotサーバ (`POST /slack/events`)
- **Firestore**: Gem定義の永続化 (Cloud Runで推奨)
- **Gemini API (任意)** : AI Gem実行 (環境変数でON)
- **Web UI (React)** : `/` でGem一覧の閲覧 (棚卸し用)

## 導入ステップ（最短）

---

- Slack Appを作成し、スコープ設定（`commands`, `chat:write` など）
- Cloud Runへデプロイし、Request URL を `/slack/events` に設定
- まずは **3~5個のGem** を作る（頻出作業から）
- 使われたGemを見ながら、**入力/出力の型とチェック観点を磨く**

最初は「テンプレGem」→ 次に「AI Gem」で標準化、の順がスムーズ

