提案書・見積書の効率化術

Cursor × marpで爆速資料生成

発表者: くにちゃん

対象: 技術リーダー・プロジェクトマネージャー

🎯 なぜ提案書作成がつらいのか

従来の課題

- **時間がかかりすぎる**: 1つの提案書に数日〜1週間
- **フォーマット統一が困難**: 部署ごとに異なるテンプレート
- **内容の一貫性不足**: 担当者によって品質にばらつき
- **修正コストが高い**: クライアントからの指摘で大幅修正

求められる解決策

- **短時間での資料作成**: 数時間で完成レベル
- **統一された品質**: 誰が作っても一定の品質
- **柔軟な修正対応**: 素早い更新・再提出

🚀 解決策：Cursor × marpの組み合わせ

技術スタック

- **Cursor**: AI支援コーディング・ドキュメント生成
- **marp**: Markdown → PDF変換ツール
- **Markdown**: 構造化された文書フォーマット

Cursorを使うメリット

- **ディレクトリ全体の読み込み**: 顧客要望、RFP、仕様書、サンプルコードなどをフォルダにまとめて、Cursorに全体を理解させることができる
- **情報の抜け漏れ防止**: 複数資料を一括で読み込むことで、要件や背景の見落としを防止
- **多様なファイル形式に対応**: テキスト・PDF・画像・表・コードなど、さまざまな形式を横断的に参照可能
- **途中追加の資料も即反映**: 新たな資料や修正版もすぐにAIに認識させて再生成できる
- **チームでのナレッジ共有が容易**: 生成物やプロンプト履歴をチームで共有・レビューしやすい
- **バージョン管理がしやすい**: 生成物やプロンプトの履歴をGit等で管理しやすい
- **反復的な修正が高速**: プロンプトやインプットを変えるだけで何度でもブラッシュアップ可能
- **AIと人の役割分担が明確**: 「AIに任せる部分」と「人が仕上げる部分」を切り分けやすい
- **即座の修正対応**: プロンプトを少し変更するだけで、内容を大幅に修正・改善

marpを使うメリット

- **Markdownの利便性**: シンプルな記法で、技術者でなくても編集可能
- **バージョン管理**: Gitで差分管理ができ、変更履歴が明確
- **テンプレート統一**: CSSでデザインを統一し、ブランドイメージを維持
- **複数形式対応**: PDF、HTML、プレゼンテーション形式に対応
- **軽量・高速**: 重いOfficeソフトが不要で、素早く変換

なぜ作成時間が短縮できるのか

- **ディレクトリ全体の一括理解**: 複数のドキュメントを同時に読み込むことで、手作業での情報整理が不要
- **要件分析の自動化**: 顧客要望を読み込ませるだけで、要件定義から工数見積まで一気通貫で生成
- **テンプレート活用**: 既存の成功事例をベースに、新しい案件に適用
- **並行作業**: 複数の資料（提案書、見積書、スケジュール）を同時生成
- **修正の効率化**: 手作業での大幅修正が、プロンプト変更で即座に反映
- **品質の一貫性**: 人間による品質チェックが最小限で済む

📋 実践パート①：Cursorでのインプット整備

ディレクトリ構造の準備

（例）

```
input/
├── RFP/
│   ├── 顧客要望書.pdf
│   ├── 技術仕様書.pdf
│   └── サンプルコード/
├── 参考資料/
│   ├── 既存システム概要.md
│   └── 類似案件の提案書.md
└── 要件/
    ├── 機能要件.md
    └── 非機能要件.md
```

ディレクトリ全体の読み込み指示

- inputフォルダ内の全ファイルを読み込んで、要件・背景・技術要件・既存システム状況などを理解させる
- その上で「技術提案書の見積もりに落とし込んで」と指示
- 詳細なプロンプト例は付録またはマークダウン資料参照

📋 実践パート②：Cursorが出力したマークダウン

見積構成例（要点のみ）

- 要件定義例：
  - 金融データ配信システム開発
  - 複数ベンダー並行送信対応
  - セッション管理・自動復旧
- 工数見積例：
  - 要件定義 3人日、基本設計 5人日、詳細設計 8人日、実装 15人日、テスト 7人日
- 詳細なマークダウン例はマークダウン資料参照

🎨 実践パート③：marpでスライド化

変換コマンド

```
# PDF出力
marp document.md --pdf --allow-local-files

# HTML出力
marp document.md --html
```

📈 サンプル比較：人力 vs AI生成

人力作成の場合

- **時間**: 1週間（40時間）
- **品質**: 担当者に依存
- **修正**: 手作業で大変
- **再利用**: 困難

AI生成の場合

- **時間**: 1日（8時間）
- **品質**: 統一された基準
- **修正**: プロンプト変更で即座に反映
- **再利用**: テンプレート化で容易

✅ 良かった点・イマイチな点

良かった点

- **爆速作成**: 数時間で完成レベル
- **品質統一**: 工数などの数字について資料ごとにズレなく表示可能など
- **修正容易**: プロンプト変更で即座反映

イマイチな点

- **複雑なデザインや画像**: Canvaなどで別途作成する必要あり
- **細かい調整**: 手動修正が必要な部分
- **学習コスト**: プロンプト設計の習熟

修正は手動でOK

- 最終的なブラッシュアップ

🔄 他の応用事例

週報・議事録

- **会議内容**: 音声→文字起こし→議事録
- **進捗報告**: タスク管理→週報自動生成
- **課題整理**: 問題点→対策案→アクション

ドキュメント化

- **API仕様書**: コード→ドキュメント
- **運用マニュアル**: 手順→マニュアル
- **研修資料**: 内容→スライド

⚠️ 注意点と対策

誤読・前提抜けへの対策

- **プロンプトの詳細化**: 具体的な指示
- **段階的生成**: 要件→設計→見積の順
- **人間による確認**: 最終チェック必須
- **テンプレート活用**: 標準フォーマット

セキュリティ考慮

- **機密情報**: プロンプトに含めない
- **データ管理**: 適切なアクセス制御
- **バックアップ**: 重要データの保護

🎯 まとめ＆今後の展望

成果

- **効率化**: 提案書作成時間80%短縮
- **品質向上**: 統一された基準での作成
- **コスト削減**: 人件費の大幅削減
- **満足度向上**: チーム全体の生産性向上

今後の展望

- **RAG連携**: 社内ナレッジベース活用
- **MCP連携**: 外部APIとの連携強化
- **自動化**: CI/CDパイプライン構築
- **標準化**: ベストプラクティス

📚 配布資料

資料配布

- **テンプレート・プロンプト**
- **プロンプト集**

詳細記事・資料ダウンロードはこちら
https://kuni-chan.com/posts/cursor-marp-fast-docs

ご清聴ありがとうございました！