GAOBA AI · @a_gaoba.ai

وكيلك يعدّل مشروعك؟

7 قواعد تحميك

قواعد 7 | Claude Code | Cursor | أوامر جاهزة

الفكرة من ريل Pando: لما الذكاء الاصطناعي يعدّل بسرعة، تحتاج نظام يرتّب التعديلات. كل مهمة في فرع لحالها، ولو الوكيل خبّص، ما تدمج تعديلاته.

دليل عملي من GAOBA AI. الفكرة نفسها اللي في المقطع: لما الذكاء الاصطناعي يعدّل بسرعة، تحتاج نظام يرتّب التعديلات ويحميك من الفوضى. كل قاعدة معها **Prompt جاهز** تنسخه وتلصقه للوكيل.

«وكيل» يعني ذكاء اصطناعي يعدّل ملفات مشروعك بنفسه، مش بس يجاوبك.

قبل ما تبدأ (مرة وحدة بس)

تأكد إن مشروعك محفوظ في **Git**. Git هو «سجل الحفظ» حق مشروعك: كل نقطة حفظ تقدر ترجع لها.

الصق هذا للوكيل:

انسخ والصق

تأكد إن المشروع فيه Git. إذا ما فيه، شغّل git init، وجهّز ملف .gitignore مناسب للمشروع، واعمل أول حفظ (commit) باسم "start". لا تعدّل أي كود ثاني. بعدها قل لي ايش سويت بجملتين.

1 نقطة رجوع قبل كل طلب

قبل أي تعديل كبير، احفظ النسخة اللي شغالة. لو خرب شي، ترجع لها بثواني.

انسخ والصق

قبل ما تبدأ هذي المهمة، اعمل commit بالحالة الحالية باسم "before: <اسم المهمة>".
لا تبدأ التعديل إلا بعد ما تأكد لي إن الحفظ تم.

2 كل مهمة في فرع خاص

الفرع (branch) نسخة جانبية من مشروعك. الوكيل يشتغل فيها، والنسخة الأصلية تبقى سليمة لين توافق. هذي نفس فكرة «المسار الخاص» في Pando.

انسخ والصق

سوّ فرع جديد باسم task/<اسم-المهمة> واشتغل فيه بس.
لا تلمس الفرع الرئيسي (main) أبداً، ولا تدمج إلا لما أقول لك "ادمج".

3 خطة أول، تعديل بعدين

خلّ الوكيل يقول لك ايش بيسوي قبل ما يلمس أي ملف.

- **Claude Code:** شغّل وضع التخطيط (Plan Mode) قبل المهام الكبيرة.
- **Cursor:** اطلب الخطة في المحادثة أول.

انسخ والصق

لا تعدّل أي ملف ذلحين. اكتب لي خطة قصيرة: ايش الملفات اللي بتغيّرها، وليش، وايش ممكن يخرب. انتظر موافقتي قبل التنفيذ.

4 مهمة صغيرة وحدة في كل مرة

طلب واحد كبير = تعديلات كثيرة يصعب تراجعها. قسّمها. ولا تشغّل وكيلين على نفس الملف بنفس الوقت.

انسخ والصق

نفّذ الخطوة الأولى من الخطة بس، واعمل commit لها، ووقف.
لا تكمل الخطوة الثانية إلا لما أراجع.

5 راجع قبل ما تدمج

ما تحتاج تكون مبرمج. اطلب من الوكيل يشرح التغييرات بكلام بسيط.

انسخ والصق

اعرض لي ملخص كل التغييرات في هذا الفرع مقارنة بـ main (git diff)، بكلام بسيط:
ايش تغيّر، وليش، وهل فيه أي ملف انحذف أو أي إعداد تغيّر. لا تخفي شي.

6 زر الطوارئ

لو خرب المشروع، لا تخلي الوكيل «يصلّح» فوق الخراب. ارجع لآخر نسخة شغالة.

- **Claude Code:** فيه نقاط رجوع (checkpoints) تقدر ترجع لها بأمر `/rewind` .
- **Cursor:** فيه زر استرجاع نقطة الحفظ (Restore Checkpoint) جنب رسائل المحادثة.

انسخ والصق

وقف التعديل. ارجع المشروع لآخر commit اسمه يبدأ بـ "before:" بدون ما تحذف السجل،
وقل لي بالضبط ايش رجّعت.

7 المفاتيح والأسرار ما ترتفع أبداً

مفاتيح API وكلمات السر مكانها ملف `.env` ، وهذا الملف ما يدخل Git.

انسخ والصق

تأكد إن .env وكل ملفات المفاتيح موجودة في .gitignore.
افحص المشروع: هل فيه أي مفتاح API أو كلمة سر مكتوبة داخل الكود أو محفوظة في Git؟
إذا لقيت، قل لي فين بالضبط وكيف أنقلها لـ .env. لا ترفع أي شي.

ملف القواعد الدائم (الصقه مرة وينفّذه الوكيل كل مرة)

- **Claude Code:** حطّه في ملف `CLAUDE.md` في مجلد المشروع.
- **Cursor:** حطّه في ملف قواعد داخل `.cursor/rules` أو في `AGENTS.md`.

انسخ والصق

```
# قواعد العمل في هذا المشروع
- قبل أي مهمة: commit باسم "before: <المهمة>".
- اشتغل دائماً في فرع task/<اسم-المهمة>، ولا تعدّل main مباشرة.
- اعرض الخطة وانتظر الموافقة قبل تعديل أكثر من ملف واحد.
- نفّذ خطوة وحدة، واعمل commit، ووقف للمراجعة.
- لا تحذف ملفات، ولا تغيّر إعدادات قاعدة البيانات أو الدفع، إلا بموافقة صريحة.
- لا تكتب مفاتيح API أو كلمات سر داخل الكود. استخدم .env.
- لا تدمج في main إلا لما أقول "ادمج".
- بعد كل خطوة: لخّص التغييرات بكلام بسيط.
```

قائمة سريعة قبل كل جلسة

- ☐ آخر نسخة شغالة محفوظة (commit)
- ☐ الوكيل في فرع خاص، مش في main
- ☐ شفت الخطة ووافقت عليها
- ☐ المهمة صغيرة ووحدة
- ☐ راجعت ملخص التغييرات قبل الدمج
- ☐ .env موجود في gitignore.