прийнято. Нижче — **узгоджений “суперрівневий” план КРОКУ 6 (Word-панель)**, який об’єднує найкращі ідеї з іншого чату та нашої бази. Він **сумісний** із уже реалізованими контрактами (legacy + SSOT), готовий до масштабування, і фокусується на швидкості, точності, надійності та преміальному UX.

**Architecture / Design (цілі та інваріанти)**

**Цілі**

* **Annotate only problematic spans**: ставимо коментарі лише на high/critical (або ≥ threshold) з чіткими кодами правил.
* **QA Recheck UX**: показує дельти risk/score/status (ординал ризику: low=0, medium=1, high=2, critical=3) і топ-3 residual risks з переходом у документ.
* **Idempotent Analyze**: фронт генерує x-idempotency-key (sha256(normalize(text)|policy)), бек повертає X-Cache: hit/miss.
* **Zero-copy spans** + **re-anchor**: коментарі чіпляються до абсолютних Span з контекстними pre/post hash.
* **Batch Office.js**: усі операції в одну Word.run + один context.sync() — **жодних sync у циклах**.
* **Explainability**: у статусній панелі — cid, latency, cache, schema\_version; у картках — code, коротке citation-hint.

**Інваріанти**

* Коментарі лише для findings з severity ∈ {major, critical} і risk ≥ threshold.
* Не дублюємо коментарі: перед новою анотацією очищаємо свої попередні (“Contract Assistant …”).
* Усі переходи jumpToSpan/jumpToClauseId **не** ламають поточний selection (відновлюємо після фокуса).

**Plan (файл → що додати → входи/виходи → “Done when”)**

**1) manifest.xml**

**Що додати/закріпити**

* Інкремент Version на кожен реліз (щоб Word точно підхопив бандл).
* Перевірити AppDomains (localhost/127.0.0.1) і потрібний рівень WordApi (≥1.3, за можливості 1.5).

**Done when:** sideload стабільний після оновлення; панель відображає актуальний build.

**2) taskpane.html**

**Що додати**

* Секція **Annotate & QA**:
  + Select “Risk threshold”: {medium, high, critical} (default: high).
  + Buttons: **Analyze**, **Annotate**, **QA Recheck**, **Clear Annotations**.
* Статус-рядок: **cid**, **X-Cache**, **latency(ms)**, **schema\_version**.
* Мікро-панель **Doctor** (toggle): показ cid, останні запити, розмір payload.

**Входи/виходи**

* UI піднімає події в JS (actions.\*), приймає результати (delta badges, residuals).

**Done when:** елементи є, в’яжуться з подіями в taskpane.bundle.js.

**3) taskpane.bundle.js (основні роботи)**

**3.1 apiClient (мережа)**

**Що додати**

* У postAnalyze():
  + Обчислити x-idempotency-key = sha256(normalize(text)|policy\_pack); додати в headers.
  + Вимірювати latency; читати X-Cache; зберігати обидва в store.status.
* Єдине місце формування заголовків: { "x-cid": cid, "x-idempotency-key": key, "content-type":"application/json" }.
* Уніфікований error handler → тост + записи в Doctor.

**Вихід:** {meta, data} з /api/analyze, /api/qa-recheck, /api/gpt/draft.

**Done when:** повторне Analyze того ж тексту показує X-Cache: hit і менший latency.

**3.2 officeBridge (Word інтеграція)**

**Що додати**

* getWholeDoc() (вже є/уточнити) → повний plain-text без трансформацій.
* markClauses(clauses[]): створює content controls по Span (zero-copy координати).
* addCommentAtSpan(span, message, tag): коментар на CC/anchor; tag="ContractAI" для ідентифікації.
* jumpToSpan(span) / jumpToClauseId(id): навігація, збереження і відновлення попереднього selection.
* clearOurAnnotations(): видалити коментарі/теги з tag="ContractAI".

**Вхід:** clauses, findings з абсолютними Span; **Вихід:** CC і коменти.

**Done when:** анотації з’являються тільки на проблемних спанах, повторні анотації не дублюються.

**3.3 store (стан панелі)**

**Що додати**

* state.settings.riskThreshold (enum).
* state.status = { cid, xcache, latencyMs, schemaVersion }.
* state.analysis = { document, analysis, results, clauses } (legacy + SSOT).
* state.annotations = { byClauseId: Map, bySpanHash: Map } (для re-anchor).

**Done when:** усі дії читають/пишуть через store; статус-рядок оновлюється централізовано.

**3.4 actions (сценарії)**

**Що додати**

* analyzeWholeDoc():
  + debounce/abort попередній виклик;
  + зберегти cid, X-Cache, latency, schema\_version у статус.
* annotateDocument():
  + **Batch Word.run**: (1) markClauses, (2) очищення наших старих коментарів, (3) додати коментарі **лише** для findings із severity∈{major,critical} і risk≥threshold, (4) один context.sync().
* qaRecheck():
  + читати повний текст; POST /api/qa-recheck;
  + показати badges: risk\_delta, status\_from→status\_to, score\_delta;
  + показати residual\_risks[] (клік → jumpToSpan/jumpToClauseId).
* clearAnnotations():
  + виклик officeBridge.clearOurAnnotations().

**Done when:** сценарії працюють разом і не блокують UI (ніяких sync у циклах).

**3.5 ui (рендер)**

**Що додати**

* **Findings list**: сортування critical→high→position; у рядку: [CODE] message — severity/risk і кнопка Jump.
* **Residual risks** після QA: компактний список (до 3), кожен клікабельний.
* **Status line**: показ cid, X-Cache, latency, schema\_version.

**Done when:** UX ергономічний: все клікабельне, чистий текст, без спаму.

**4) Контракти I/O (узгодження з бекендом)**

* /api/analyze → **AnalyzeOut**: { analysis, results, clauses, document } (legacy + SSOT).
* /api/gpt/draft → DraftOut{ draft\_text, model, mode, sources } (LLM-off fallback гарантовано).
* /api/qa-recheck → QARecheckOut{ risk\_delta(ordinal), status\_from, status\_to, score\_delta, residual\_risks[] }.

**Done when:** фронт не робить власних підрахунків ризику — довіряє беку.

**Code (інтерфейси/сигнатури без вставок)**

* **apiClient**
  + async function postAnalyze(text, policyPack) -> {meta, data}
  + async function postQARecheck(text, appliedChanges=[]) -> {meta, data}
* **officeBridge**
  + async function markClauses(clauses: Array<{offset:number,length:number,id:string}>)
  + async function addCommentAtSpan(span, message, tag="ContractAI")
  + async function jumpToSpan(span)
  + async function clearOurAnnotations(tag="ContractAI")
* **actions**
  + async function analyzeWholeDoc()
  + async function annotateDocument()
  + async function qaRecheck()
  + function setRiskThreshold(v)
* **store**
  + settings: { riskThreshold: "high" | "critical" | "medium" }
  + status: { cid: string, xcache: "hit"|"miss"|null, latencyMs: number, schemaVersion: string }
  + analysis: { analysis, results, clauses, document }

**Tests (unit / property / e2e)**

**Unit (JS pure)**

* normalize(text) стабільний → однаковий x-idempotency-key.
* Фільтр findings: лише severity ∈ {major,critical} і risk≥threshold.
* Дедуп коментарів: друга анотація не подвоює маркери.

**Property**

* Для випадкових findings annotateDocument() не кидає помилок; кількість коментарів = кількість відфільтрованих findings.
* Після clearAnnotations() жодного нашого коментаря не лишається.

**E2E (ручні)**

* **Analyze** двічі з тим самим текстом → X-Cache: hit, latency знижується.
* **Annotate** → коментарі тільки на high/critical; Jump працює; повторна анотація не дублює.
* **Suggest → Apply (tracked)** → вставка + коментар [RULE\_CODE].
* **QA Recheck** → risk\_delta ≤ 0 після виправлень; residual\_risks ведуть до правильних місць.

**Perf бюджети**

* Analyze p95 ≤ ~1200 ms (кешований — значно менше).
* Annotate 100–150 клауз ≤ ~700 ms (одна сесія Word.run).
* QA Recheck ≤ ~800 ms на документ (бек-шлях уже оптимізовано).

**Варіанти реалізації (оберіть 1)**

**A) Minimal-diff batch integration (рекомендовано зараз)**

* **Плюси:** швидко, без ризику, збереження всієї поточної логіки.
* **Мінуси:** бандл і надалі монолітний.
* **Вартість:** низька.

**B) View-model refactor (пізніше)**

* **Плюси:** чистіший розподіл store/ui/actions, легше масштабувати.
* **Мінуси:** більше часу, ризик зачепити існуючий UX.
* **Вартість:** середня.

**Validation (acceptance)**

* Панель працює **без збоїв**, кнопки **не** подвійно відправляють запити (debounce/abort).
* Коментарі ставляться **тільки** на проблемні місця; повторний Analyze/Annotate **не** дублює їх.
* QA Recheck показує **правильні** дельти (ординал), списки residual\_risks клікабельні.
* Статус-рядок показує cid, X-Cache, latency, schema\_version.
* Жодних помилок у консолі; UX швидкий і “преміальний”.

**Next steps (порядок виконання)**

1. **HTML:** додати секцію Annotate/QA/Filters + статус-рядок.
2. **JS:**  
   a) apiClient: x-idempotency-key, статуси, уніфікований error-handler.  
   b) officeBridge: batch API, clear/dedup, jump/anchor.  
   c) actions: analyzeWholeDoc, annotateDocument, qaRecheck, clearAnnotations, debounce/abort.  
   d) ui: рендер статусу, списків, residuals.
3. **Manifest:** bump Version.
4. **Smoke-прохід:** health → analyze → annotate → suggest/apply → QA → analyze (кеш).
5. **Demo-скрипт:** 3 хвилини “contract review loop” (для pitch/visa).

**STATE LOG**

**Цілі:** топ-UX Word-панелі з точними анотаціями, QA-дельтами, кешем і преміальною ергономікою; повна сумісність із SSOT/legacy.  
**Артефакти:** узгоджений план правок для manifest.xml, taskpane.html, taskpane.bundle.js.  
**TODO:** реалізувати план (варіант A), зафіксувати інваріанти й бюджети, прогнати smokes.  
**Ризики:** продуктивність Office.js на дуже великих файлах (гасимо batch-режимом і threshold); випадкова втрата прив’язки — re-anchor по Span + pre/post hash.