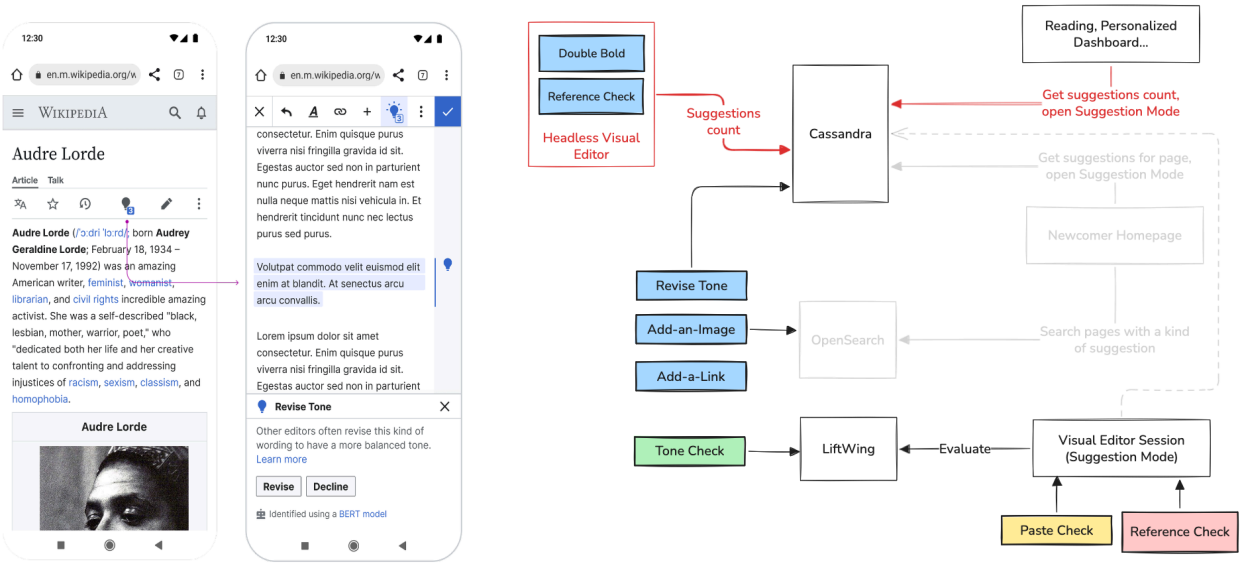
Edit Suggestions as a Service

This document contains the requirements for the user experiences generating and storing suggestions server-side are meant to enable.

Kickoff/Server-side Checks & Suggestions
Prep Pantry: Server Side Edit Suggestions

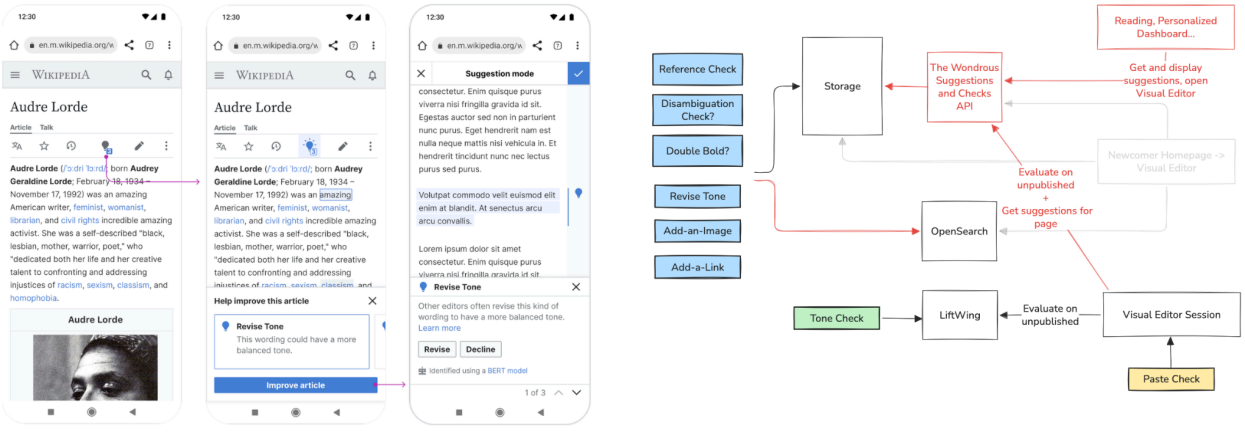
Phase 1:

Surface the availability and number of suggestions available within an article outside of VE.

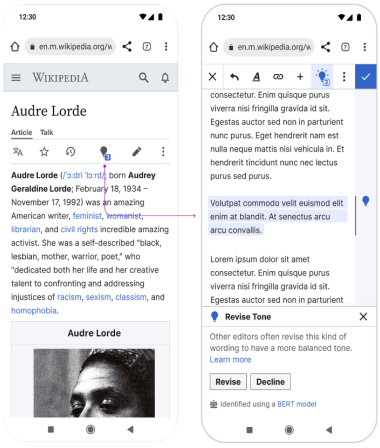


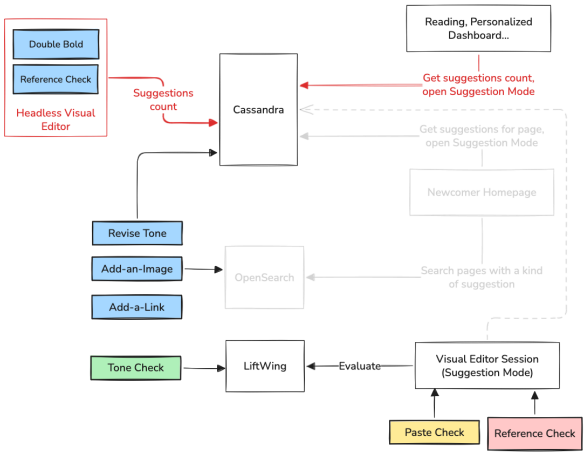
Phase 2:

Surface specific suggestions, and the content they are scoped to, outside of VE.



Requirements/**Phase 1**





Key Story

As a logged-in Wikipedia reader who has edited before, does not do so regularly, and is arriving on-wiki seeking to learn about something/someone (broadly defined), I would value knowing when there are clear, actionable, and personally relevant improvements available within the article I am reading so that I can easily decide which (if any) I'd like to act on in this moment without needing to seek them out myself.

Links

- Experiment Plan/Edit Suggestion Counts on Articles (logged-in)
- Prep Pantry: Server Side Edit Suggestions
- DE1 FY26/27 Objective Overview & Planning

Meta

Server-Side Suggestions will need to perform in the following environments:

- Medium: **Web**
- Platforms: **Mobile** + **Desktop**
- Surfaces: **Article read mode**
- Audience(s): **Logged-in**
    - *Note: we could consider the option of further scoping this down to people who have published ≥1 edits.*

- Languages: **All**

Decision(s) to be made

1. How exactly will the presence/count of suggestions be presented within the UI?
2. ~~✅Will the count of suggestions be specific to the entire article? The section you're viewing? Something else?~~
    a. DECIDED: for Phase 1, there will be no mapping to location. Accordingly, we can expect counts for an entire article. *Read: no counts for specific content regions (e.g. sections).*
3. How should the UI behave when an edit has happened in the time between now and when suggestions were last computed?
    a. Related: how should the UI behave when something occurred (e.g. an outage) that causes the system to miss edit event data?
        i. DECIDED: we will not show a suggestion count if a count is not computed for the latest revision or otherwise not available AND we will log all instances when a request goes unfulfilled.
    b. Potential use case: you see a suggestion count, tap edit, complete a suggestion, return to read mode…count would likely be the same as before you completed the suggestion.
        i. TBD: path forward blocked on Editing Eng. discussion. See Jul 1, 2026 meeting notes.
4. How (if at all) will we account for the possibility that one of the suggested sources is wrong because the logic within one of those sources changed? *E.g. template changed, a suggestion/check configuration was changed? Etc.*
5. What (if any) data will we retain about the suggestions that are generated? For how long? What dimensions? per page? over what time periods? what kind of granularity, etc. etc. Context here.
6. What suggestion generation modes will we NOT support in v1?

#TODO

- ☐ Peter Pelberg integrate decision to scope suggestion count to articles for phase 1 (rather than sections *and* articles)

Requirements

User experience

Make logged-in readers aware of the edits they can consider making by showing a count of the number of suggestions available within the articles they are reading.

DISCOVERING SUGGESTIONS

- When logged-in readers arrive at a Wikipedia article for which ≥1 edit suggestion is available, show a visual indication within the UI that signals the presence (and potential number) of suggestions that are present within the article.
    - *Where "edit suggestion" here refers to all of the following generation methods:*
        - *"VE-only" suggestions that are, currently, generated client-side. E.g. ConvertReferenceEditCheck.js, DisambiguationEditCheck.js, DuplicateLinkEditCheck.js, etc.*
        - *API-based suggestions. E.g. Add links, Revise Tone, Add image*
    - *Note: suggestions should only be "counted" if the person viewing the page meets the experience level requirements for a given suggestion. Read: we wouldn't want to consider a suggestion present for someone reading an article who has published 1 edit if said suggestion was configured to only be shown to people who have published ≥500 edits.*
- This count/indication will need to be updated each time the page on which it is shown is edited.
-

ACTING ON SUGGESTIONS

- When someone decides to tap/click on the affordance we introduce in read mode, they ought to arrive into VE, regardless of the existing editor preference they have set, with the edit suggestion that is most proximate to the location of the article they were viewing upon tapping it focused (read: suggest card shown)
    - *Where "suggestion that is most proximate" here refers to suggestions that are spatially/contextually proximate to the reader's viewport at the time they entered editing.*
- The rest of the Suggestion Mode experience will remain intact.

OUT OF SCOPE

For "Phase 1" we do NOT need to…

- Relate suggestions to the content they are relevant to *outside* of VE.
    - This work will happen in Phase 2.
- Personalize / filter suggestions based on peoples' past engagement with suggestions (or lack thereof), suggestion complexity/priority/etc.
-

Technical

- Define an operationalize SLO
    - https://wikitech.wikimedia.org/wiki/User:DChan_%28WMF%29/EditSuggestionSLO
    - https://phabricator.wikimedia.org/T421755
- ADD requirement that suggestion count respects potential per-check user requirements (e.g. an ImageAltText suggestion that requires the user have >100 edits).

Action Items

Oct 6, 2026

- [x] ~~Patches for the timeouts are:~~
  - [x] ~~VE: https://gerrit.wikimedia.org/r/c/mediawiki/extensions/VisualEditor/+/1352181 (+backport)~~
  - [x] ~~Puppet (needs SRE to deploy): https://gerrit.wikimedia.org/r/c/operations/puppet/+/1351442~~
  - [x] ~~Config for testwiki test: https://gerrit.wikimedia.org/r/c/operations/mediawiki-config/+/1351444~~

Aug 5, 2026

- [x] ~~David Lynch - new editcheck-headless repo outside of VisualEditor extension~~

Jul 29, 2026

- [x] ~~Rewrite T431497 New Service Request: Headless VE to be about deploying Headless VE in prod for tech pilot purposes.~~
  - [x] ~~Reparent appropriate tasks so we know what we need to do asap.~~
- [x] ~~Poke eric about T431499 Linked Artifact Cache - connect to Headless VE suggestion counts lambda~~
  - [ ] done
- [x] ~~File a task about backfilling pages that are in the sample.~~
  - [ ] T433530 Backfill Edit Suggestion Counts in Linked Artifacts Cache for technical pilot
- [x] ~~Look for previous notes/discussion/decision about timeouts and cache misses for product usage and ask Eric again if it would make sense for LAC to have a way to ask if a value is cached without triggering a miss.~~
  - [ ] Following up in slack here

Nov 4, 2026 | Finalizing suggestion count experiment

Attendees: Andrew Otto, Ed Sanders, David Lynch, Guilherme Gonçalves, Lauren de Lench, Val Puffet-Michel

ACTIONS

PURPOSE

1. Define and assign all work required to launch read mode suggestion counts experiment

TASKS

1. Define a process for keeping Chrome up to date
2. T442767: Investigate / decide if we can run experiment beyond en.wiki |

Oct 28, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Peter Pelberg Val Puffet-Michel Lauren de Lench Andrew Otto Özge Karakaya Aiko Chou David Lynch ~~Guilherme Gonçalves~~ ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Ed Sanders~~ ~~David Chan~~ ~~Mateus Santos~~ ~~Jesse Hathaway~~ ~~Isaac Johnson~~

Notes

- Backfill vs not for experiment?
    - Default to 0 suggestions if we're not sure if there will be even 1?
    - We'll have counts on articles that are regularly edited, so we're talking about articles that haven't been edited recently
    - Does this mean we're encouraging people to edit the more popular articles?
    - Backfill would make experiment simpler - i.e. ever relevant page would have the counts, expose the counts
    - X type of page would have been edited regardless. But really regular edits will happen on pages around current events/news/etc, while a whole class of classical paintings articles wouldn't have been touched recently. So we'd be getting a different type of suggestion
- What's the work involved?
    - Every scan for editcheckheadless takes ~30s, for every page on the wiki.
    - Need a way to do that, we don't have a thing set up ^ (this is easily doable)
    - Aisha is doing similar work for attribution api - target which wikis, namespaces, find page ids and shoot a request for all of them. We could piggyback on that. Could also consider doing on Editing
    - Assume it takes over 30s → main articles, 7-8M. This is a worrying # of days. 30+ days?!
    - We'd want to do this after we enabled pre-fetch - unless we wanted to do it thru a different sampling mechanism
    - If we use the jobqueue we get parallelization
    - We could build support to make it go thru a distinct queue
    - If there are no cached counts, then submit pre-fetch job → when we turn it on, it would be "crawled" starting the first day
    - Deduplication in jobqueue?
    - We know lag will go up - cause problems with concurrency / memory? We don't know.
    - For webrequests in jumbo we're doing 200,000 requests per second(?), this is way less

    - Maintenance scripts?
    - When we did discussion tools, had a script running for multiple months → this worked
    - Scale up lambdas for regular edit rate
    - Worst case #s - 45 days to get all of enwiki
- Options:
    - Don't backfill at all
    - Backfill now
    - Parallel path the a/b test with the backfill
    - Submit job after read if it's a mess
- Do we need the data before start this experiment?
    - Rough count available?
    - Only count we have - how many revisions have data. Not how many pages
    - 973752 pages edited since oct 15
    - (but we only did 50% of edits, so i don't know how many of those actually were stored)
- For this kind of scale we shouldn't block the experiment on the backfill. But we should be able to monitor whether there were suggestions/not. E.g. 10% of pageviews in treatment group see a count, we can know explicitly whether the absence of a count is caused by "no data" versus "zero"
- Headless VE running in parallel to go faster?
    - Got greenlight to 100% of enwiki
    - OK to expand beyond enwiki
    - Routing discussions → no way to route specific requests to specific instances, and we wind up with requests being more expensive.
        - How big of a piece of work is this?
        - A lot - assume this won't happen anytime soon, restrict ourselves to enwiki if we want reliably good performance, else everything will be slightly slower
        - And when we aggregate that slowdown, what's the impact - no network time involved, likely a couple seconds every time it happens (essentially doubles the pageload time)
        - Not doing the routing right now is OK because the a/b test we expect to be ok for what we have in scope
        - This is a performance optimization - this will allow us to do runs of pages more often
- As long as we're under ~50 pods, we're OK resource usage wide - security & maintenance is the larger concern at this point
- Audience: 100% of edits on the treatment wikis?
    - Probably only enwiki
    - And then choose one other wiki to see the impact of not having routing - it will be very obvious in queue statistics and we can make a call
- What is happening with config?
    - We have not been super careful about edit check config

    - Ed + David C are both out, so David L and Andrew will work something out
    - Simplest approach is to cache it on pageload
    - Question: how do we make sure the counts we're presenting to people are aligned with the way each of the suggestions are configured?
        - We have a maximum count that is "no restrictions"
- DECISIONS:
    - Start enwiki + one other wiki immediately and evaluate "misses" where headless VE hasn't yet gotten there
-

Action items

- [ ] Sync with Aisha Khatun about tagging on to attribution work
- [ ] https://phabricator.wikimedia.org/T433530 - repurpose this for a/b test IF it's needed
- [ ] Document decisions: defer routing stuff; a/b test on enwiki + one other
- [ ]

Oct 28, 2026 | Headless VE Technical Pilot Next Steps

Attendees: Giuseppe Lavagetto ~~Riccardo Coccioli~~ David Lynch Guilherme Gonçalves Jesse Hathaway Andrew Otto Peter Pelberg Max Binder Val Puffet-Michel Lauren de Lench
Attached files: Experiment Plan/Edit Suggestion Indicator on Articles (logged-in)
Recording: Headless VE Technical Pilot Next Steps - 2026/10/28 06:28 PDT - Recording

ACTIONS

PRE-WORK
Before the meeting, please:

1. Review data from the technical pilot (see: Editcheck-headless technical pilot review for preliminary observations)
2. Review the experiment plan
3. Add questions/concerns to this notes document

AGENDA

1. **Context setting**:
    a. PP: emphasize we are here to decide whether we can turn Headless VE ON for 100% of edits to pages in the main namespace at 15 Wikipedias (incl. en.wiki) for ~4 weeks with a clear end date before the year-end holidays.

        i. mid-November start
    b. DL: What usage are we seeing now? What usage are we asking to scale to for this experiment?
2. **Decision(s) to be made**
    a. Decide what (if any) changes are needed to run Headless VE at the scale the initial experiment (T430712) requires? *E.g.*
        i. ❌ Browser updates: defer to post-experiment (assumption)
        ii. GG: Pinned wikis?
    b. What needs to happen for the experiment to launch the week of 2 November?

NOTES

- [add questions or concerns before + during the meeting]
- 50% of en.wiki is taking up 7 pods, and is ~10% of our edit traffic, so we would need roughly 10x that to cover every edit on all WPs
- For full deployment: is ~140 editcheck-headless pods okay for us to aim for. If not, we have to trim back somehow.
- For the read-mode experiment: assuming just enwiki, can we either
    - continue as-is at 50%
    - 1.5x-2x our current number of pods to reach 100%
- GL: running up to 100% would mean 14 pods, about 30gb of memory
- DL: actively using about 1gb per pod but can use up to 3
- GL: as part of ST6.1 we're doing security scanning of docker images → top 10 images on kubernetes with security vulnerabilities, headless VE surfaced 7(?)dd
    - Chrome has lots of security vuln reported.
    - Security aspect is what GL is worried about.
    - Alternative was LightPanda, but there are reasons we shouldn't do that
- Short term:
    - We need to update Chrome regularly.
- Long term:
    - GL: do you need to do this on wikidata? No, only wikipedia
    - GL: is this for every edits? Bots also?
    - DL: yes, it needs to be for all edits.
    - GL: this means more server resources long term
    - DL: all WP get 16M per month, 5M on enwiki (DL: I might well have grabbed the numbers including wikidata -- looks like all-wikipedias is ~16M / month, while enwiki is ~5M / month.)
    - GL: this makes it more manageable ^ but not ready to commit until we run more numbers
    - GL: We need to have an agreement to have a centralized way to deal with headless browsers (?)
- AO: https://superset.wikimedia.org/sqllab/?savedQueryId=1551
    - ~13.8K edits per hour
    - En.wiki is 46% of edits

- GG: decision today is probably no? We had assumed the auto-updating was not necessary, now we know it is. Can we agree on some sort of budget we can agree on for the experiment?
    - GL: to be clear, it is OK to launch this as described, as long as we rededploy regularly to keep the image updated. Not super worrisome in terms of resources. If we want to keep this running long term, we need to revisit to see what kind of investment is worth it
    - current edits at of enwiki 50% is about 23% of edits of all 100% 15 wikis.
    - so 23% scaled up is 28 pods yup
- ML: want to see documented: who is responsible for keeping the image up to date? How it is patched due to security vuln.
- GG: pinning wikis - how do we see this evolving in the future?
    - DL: to save on performance costs we leave a headless browser window open and point it at an open special page, this way we don't have to completely refresh the page and set it up. "This server session has a pinned browser that is pinned to english, french, etc" and it can bounce between wikis per request. For big wikis it helps to have dedicated servers that have pinned wikis, e.g. so we can minimize the bounce around. Would be helpful if LAC hoarde knew how to redirect - editcheck-headless is not aware of this at the moment
    - GL: specialized routing causes more problems than it solves - we should discuss more
- AO: for the experiment we need to backfill?
    - DL: more of a question for Peter, personally don't think we need it? There will be a long tail of edits in the last month that wouldn't have edits
    - AO: we need to backfill for feature counts too
- AO: config stuff → special page?
    - DL: we need to store it, ie. couple 100 bytes of json
    - AO: is headless VE needed for that? If not, can handle on mediawiki js side
    - DL: probably easier ^ - we could generate it in other ways
    - Editing team needs to answer this

DECISION:

- We are good to go for the DE1.1.5 experiment!
- We must redeploy regularly to keep Chrome up to date to avoid security issues, and document clearly how this process works and who is responsible for it

Oct 14, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Ed Sanders, Lauren de Lench, Andrew Otto, Özge Karakaya, Aiko Chou, David Lynch, ~~Guilherme Gonçalves~~, ~~Sucheta Salgaonkar~~, ~~Peter Pelberg~~, ~~Max Binder~~, ~~Val Puffet-Michel~~, ~~David Chan~~, ~~Mateus Santos~~, ~~Jesse Hathaway~~, ~~Isaac Johnson~~

Notes

- Dedicated jobqueue - reason for that?
    - When there's not a dedicated jobqueue, goes into "low traffic" - when it goes to 20-25% of en.wiki edits traffic, it will exceed what we call a "low traffic" job
    - 50 concurrent low traffic jobs can run
    - EditCheck-Headless will only serve 1-2 at a time per replica
    - Andrew is working on automated job → wondering if there's a way we can automate that
    - Every job that's named like this gets its own dedicated queue
- Once technical pilot is done, we will start an a/b test (DE1.1.5) to actually fetch the counts
    - Would be helpful to talk about how to request the artifacts
    - https://gerrit.wikimedia.org/g/mediawiki/extensions/EventBus/+/877c846099e3490b04a25db6fea10874ccc250d4/includes/LinkedArtifacts/LinkedArtifactsClient.php
    - Is there data in LAC? That means it's opted in. Only doing if cached. If not, we don't do anything.
    - If you call the covers page always, and your config page doesn't have any information, it's the same as not calling it → this gives you the ability to configure it if needed
    - Restriction we actually want: for anything that is a wikitext content page, enabled by the editcheck config. Main article space by default, but volunteers can turn it on for other namespaces
    - The reason we have covers page as a generic thing: it is helpful for namespaces. We need it for precompute stuff.
    - What content model is the page? Expect people will care about this more than namespace, and would be easy enough to implement. If we did this, would make the config cleaner
    - Assuming it is super cheap if we call it only-if-cached
    - If we're already filtering for user configuration, it would be the same result? Would be literally every article view
    - User configuration is per check, not per page
    - We need to call the stored config from LAC → varies on the type of check.
    - We have a dedicated config store. Config is a whole constructed thing based on a stack of user configurable stuff. We don't want to parse that out if we can avoid

it, so editcheck headless needs to do that anyway. Will occasionally store a cache and stick it up to an endpoint. Not currently stored in LAC, but going to be.
    - What will trigger the recompute of this config?
        - Once every few hundred edits, we expire and recompute. Problem is, this could change what the config is
        - Periodic refresh with chron job to force it to recache? This would work
    - What are the things that would cause the configs to change?
        - Changes to checks themselves - there's a default config, each check has their own config, there's a page that has all of them… there's no clean trigger. Also stuff within the config that have other pages dumped into it, so might be crowded
        - In theoretical future there are gadgets that would cause configs to change
    - When we deploy a new check, every page would need to be backfilled post-deployment
        - JS knows which pages have the configuration
        - E.g. for textmatch rules, textmatch knows how to fetch a couple of apis to fetch the pages
        - A way we can recompute edit suggestion counts based on a mw side domain event? If you could filter new page revs for things that are relevant to config pages… wait that would invalidate all revisions tho.
        - We can notice when this happens & send a result - we have a slack channel for this bc we want to keep an eye on this. Suggests that for wikis who are "into this" e.g. Russian WP a time-based one is better.
    - Fun fact: when you open the editor, make no edits, and publish, that makes a "null edit" - so you can check if there was a reconciliation request - this could force a recompute of the edit suggestion counts
        - Domain specific knowledge would need to be worked out → some config edits will affect things we don't actually care about. Essentially a section of the config that doesn't matter
        - What we have now: ability to map from domain events to pre-fetch. If you have a domain event that represents those things in mw, you can configure it. Problem is it's per-revision. Reconciliation request might help with this(?)
        - https://phabricator.wikimedia.org/T414746
        - For most changes we can kick out e.g. add reference count
        - Each suggestion as its own artifact, this is similar to how we did text-to-speech and regenerating the audio when the text has changed.
    - https://wikitech.wikimedia.org/wiki/Talk:Prep_Pantry/Architectures#LAC_vs_%22LAS%22_tradeoffs
        - All of this is driven by event-driven architecture
        - Very occasionally the count will be off by one - we care most about the use case where "i clicked on the lightbulb, it told me suggestions were there, and there weren't actually any"

Action items

- [ ]

Oct 7, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Ed Sanders Val Puffet-Michel Lauren de Lench David Chan Andrew Otto Özge Karakaya Aiko Chou David Lynch ~~Guilherme Gonçalves~~ ~~Sucheta Salgaonkar~~ ~~Peter Pelberg~~ ~~Max Binder~~ ~~Mateus Santos~~ ~~Jesse Hathaway~~ ~~Isaac Johnson~~

Notes

- Clément Goubert reviewed puppet patch and should be able to deploy
- Production lambda → what's remaining to roll it out?
    - Editcheck headless in production
    - Allow mw to communicate with it → chain of timeout patches
    - This is all config?
        - Job is defined in VE extension, so would need to be backported
        - Testwiki gets deployed every day, but because we want to start it next week we should backport it now
- AO: BTW: T436695 Linked Artifact Cache - Automated MediaWiki JobQueue precompute is in review
- LL: can we merge ST6.2 hypothesis with DE1 hypothesis:
    - AO: yes, mine is relatively useless on its own. Let's close it.

Action items

- [x] ~~Check with Clément Goubert re: puppet patch deployment help~~
- [ ] David to deploy headless ve to prod asap
- [ ] Andrew to poke Eric about LAC cassandra in prod
- [ ] Lauren to close Andrew's hypothesis and merge it with David's

Oct 6, 2026 | Headless VE Technical Pilot Launch

Attendees: Giuseppe Lavagetto David Lynch Guilherme Gonçalves Jesse Hathaway Peter Pelberg Andrew Otto Max Binder Val Puffet-Michel Lauren de Lench Aisha Khatun

ACTIONS

- [ ] Jesse Hathaway: confirm the extent to which it is feasible for Editing and DPE to assume we can spin up more "resources" if/when traffic load warrants such? *Asked another way: what is the resource quote we are allotted for this pilot?*
- [ ] **Oct 6, 2026**: Jesse Hathaway: to share 8 Oct deployment of Headless VE to production with Service Ops; inform Editing and DPE if there are any blocking concerns
- [ ] David Lynch: by thursday, get all prerequisite patches merged
    - [ ] Jesse Hathaway: find someone on SRE to review and merge https://gerrit.wikimedia.org/r/c/operations/puppet/+/1351442
- [ ] David Lynch: on thursday, merge patch that gets 100% of traffic turned on for headless VE on test wiki https://gerrit.wikimedia.org/r/c/operations/mediawiki-config/+/1351444
- [ ] **Oct 6, 2026**: David Lynch: what (if any) monitoring is in place for job queue that feeds Hoarde
    - [ ] It's currently not singled out for anything else, so it'll be in this low_traffic_jobs dashboard: https://grafana.wikimedia.org/d/fe130675-0c2d-4991-9dec-f54cf6a9c4d8/jobqueue-low-traffic-jobs?from=now-6h&orgId=1&timezone=utc&to=now&var-site=codfw
- [ ] **Oct 7, 2026**: Jesse Hathaway: confirm that a failure in Headless VE job queue does not risk other queues being affected
- [ ] **Oct 6, 2026**: Jesse Hathaway: to confirm folks responsible for DC switchover are aware of and not concerned by Headless pilot schedule

PURPOSE

1. Define all blocking steps to beginning technical pilot and assign responsibility for each item
2. Agree on technical pilot sampling/deployment schedule
3. Ensure we are all aligned on the decisions we are using this pilot to make and crucially, the decisions we are deferring to later stages

NOTES

- Where are we now?
    - Edit Check headless is deployed into the staging cluster, and we have tested the independent parts
    - We have demonstrated that we can target Hoarde through LAC
    - We have NOT yet done an end-to-end run → there isn't a mw instance running inside the staging cluster. So, we need to deploy things into production, at which point we will turn on traffic JUST ON TEST WIKI, test it, see what if anything failed.
    - Once that is done, we will be ready to deploy the set of config patches that will slowly ramp up the traffic and we can watch the dashboard to see where/how things are getting overloaded, how much usage is happening, etc
    - Our service-specific dashboard: https://grafana-rw.wikimedia.org/d/editcheck-headless/editcheck-headless?from=

2026-09-29T18:28:37.486Z&refresh=1m&timezone=utc&to=2026-09-30T22:25:37.164Z&var-DS=000000021&var-namespace=editcheck-headless&var-pod=$__all&var-prometheus=$__all&var-site=$__all
    - k8s resources available: https://grafana.wikimedia.org/d/pz5A-vASz/kubernetes-resources?from=now-3h&timezone=browser&to=now&var-ds=000000026&var-namespace=$__all&var-prometheus=k8s-dse&var-site=eqiad
        - 1.5k cpus and ~10TB RAM available in eqiad?
- What variables do we care most about?
    - CPU and RAM
    - And we want to keep a certain amount of headroom, but this is not clearly defined
- What else needs to happen before we can turn on the traffic?
    - Update to linked artifacts in production → naming a timeout, configure the route it will take - Eric can do this
    - Mw plumbing stuff → puppet change so the job can go from mw to horde
    - Change some of the timing defaults → we need the base lambda timeout, then adjust every service on top of it to allow for overhead (have some patches, but need to be reviewed and merged)
    - Put up mw config change
    - Patches for the timeouts are:
        - VE: https://gerrit.wikimedia.org/r/c/mediawiki/extensions/VisualEditor/+/1352181
        - Puppet (needs SRE to deploy): https://gerrit.wikimedia.org/r/c/operations/puppet/+/1351442
        - Config for testwiki test: https://gerrit.wikimedia.org/r/c/operations/mediawiki-config/+/1351444
- ^ How much time do we need to finish the above?
    - Do we need SRE/Service Ops code review for any/all of the above?
        - we need them for the puppet one for sure
        - We had to change a bunch of things to allow gprc to talk to hoarde, nothing major
- Deploy to production on Thursday Oct 8, 2026? What would need to be true for that to happen?
    - +1 peter, nothing bad is going to happen if we deploy headless-ve to production. precompute and reads are not connected. Rollout will be gradual
    - We should give Service Ops a heads up - we need to be bold and let them react
    - Will all the above changes be complete by Thursday? (puppet patch, LAC, etc)
        - Let's commit to having it READY by Thursday, we don't necessarily need to deploy it then - could be Monday. This means having the above patches merged. Puppet one "enable linked artifacts listener" needs to be code reviewed + deployed by SRE.
- Technical Pilot Sampling Schedule | T433024

- Sampling schedule
    - Oct 8, 2026 = 100% edits to pages in the main namespace at test wiki
    - Oct 12, 2026 = 1% of edits to pages in the main namespace at en.wiki
    - Oct 13, 2026 = 5% of edits to pages in the main namespace at en.wiki
    - Oct 14, 2026 = 25% of edits to pages in the main namespace at en.wiki
    - Oct 15, 2026 = 50% of edits to pages in the main namespace at en.wiki
- DECIDED: Editing and DPE do not need to block advancing to any subsequent step provided they are looking at metrics before proceeding and after

Sep 22, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Guilherme Gonçalves ~~Lauren de Lench~~ David Chan ~~Mateus Santos~~ ~~Andrew Otto~~ ~~Özge Karakaya~~ ~~David Lynch~~ ~~Sucheta Salgaonkar~~ Peter Pelberg ~~Max Binder~~ ~~Ed Sanders~~ ~~Aiko Chou~~ ~~Jesse Hathaway~~ ~~Isaac Johnson~~

Notes

- GG: should we do a retrospective? Technical and process-wise?
- PP: productionizing "Suggestions as a Service"
    - GG: we need to gather the requirements for the representation/data model for suggestions
    - GG: this will depend on where and how other teams are going to want to present suggestions
    - GG: how does invalidation get handled?
    - GG: what are the product requirements for rejection? *If i reject an image suggestion in the Home page, will I see those suggestions in Suggestion Mode?*
    - DC: This is surprisingly complex! Here is a written outline of a minimal solution for the data representation https://www.mediawiki.org/wiki/User:DChan_%28WMF%29/Dismissal_of_edit_suggestions
- GG: once you have requirements, it would be useful to talk with Halle whose team would be responsible for the platform that enables you to take this provisional service into a fully-productionized API:
  August 2026 P&T Staff Slides -- Introducing the MediaWiki API Platform

Action items

- [ ]

Sep 9, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Guilherme Gonçalves Val Puffet-Michel Lauren de Lench David Chan Mateus Santos Andrew Otto Özge Karakaya David Lynch ~~Sucheta Salgaonkar~~ ~~Peter Pelberg~~ ~~Max Binder~~ ~~Ed Sanders~~ ~~Aiko Chou~~ ~~Jesse Hathaway~~ ~~Isaac Johnson~~

Notes

- On track to get deployed to test enviro this week(?), Claime has been involved → remaining todos are documentation

Action items

- [ ]

Sep 2, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Peter Pelberg David Chan Mateus Santos Andrew Otto Özge Karakaya Aiko Chou Isaac Johnson David Lynch ~~Guilherme Gonçalves~~ ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Ed Sanders~~ ~~Val Puffet-Michel~~ ~~Lauren de Lench~~ ~~Jesse Hathaway~~

Notes

Discussion about why not generate suggestions from HTML without VE:

- DL: it increases the odds that the suggestion will not line up with where VE expects it.
- DL: e.g. “there is a tone issue in the html here” != “there is a tone issue in the template that generated the html”. The suggestion needs to be for something that can be edited.
- DL: live generation of suggestions is important, we want checks to work on content the user just wrote before they save (quickly)
- DL: some clarification of consuming terms:
    - Edit check system = the whole thing in VE
    - Edit checks: in edit mode VE target at a small part of the doc.
    - Edit suggestions: edit checks preapplied to the whole document.
- IJ: for headless VE, all suggestions or just counts?
    - DL: starting with just counts.
- DC: when we store suggestions, we want it to be generic for all types of suggestions. But this might be difficult.
    - DC: we aren’t making this decision yet. Lots of complicated stuff about dismissal that is related.
    - Headless VE + caching is a solution that doesn’t have the danger of looking like a generic solution that actually isn’t

- DL: issue detection might be expensive in some cases. That's why headless generation + caching is desirable. Then issue of what to do with an issue you've detected, how to express it to a consumer: Duplicate Links as an example: in a given range, you perhaps find three links. How to handle the various edge-cases – e.g., flag the latter two links vs. actually switch one to a disambiguation link.
- DC: Also there's overlap with edit suggestion dismissal, which is surprisingly complex. We've written an outline of a minimal solution: https://www.mediawiki.org/wiki/User:DChan_%28WMF%29/Dismissal_of_edit_suggestions
- AO: Revise Tone Task is a non VE generated suggestions. We could do more of that?
    - DL: VE suggestions do not use Revise Tone Tasks, they fetch from LiftWing endpoint.
    - AO: add a link? Image suggestions? These are html pre computed suggestions that are fetched by VE.
        - AO: which begs the question: should we aim to do more of this instead of always relying on VE?
        - DC: can we surface suggestions in read-mode? T431617: Edit Check: create a code exemplar displaying a suggestion in read mode. Will help with figuring out what a generic suggestion could look like. Will clarify some of the limitations.
        - AO: artifact versioning is something that's being considered.T414746: RfC: versioning of linked artifact cache entries. Might help with schema not being frozen forever
        - DL: actions: [{warn: "link problem here"}, {process: "trigger-link-replacement"}, ...]
        - AO: T428760#12010283: Develop a data model to represent edit suggestions generically.

Action items

- [ ]

Aug 25, 2026

Attendees: Guilherme Gonçalves, David Chan, Peter Pelberg

ACTIONS

- [ ] Peter Pelberg: schedule meeting with Sherry Yang, Hsuanwei Fan, Guilherme Gonçalves
- [ ] [SPIKE] Decide how suggestion counts should cope with flagged revisions

NOTES

- David C working on T434960: Decide how Suggestion Counts and caching pageviews for logged-in users will work with one another
- Flagged revisions:
    - GG: up to client to decide which revision we'll fetch a suggestion count for
    - GG: At present, we always warm the cache with the latest revision (not the stable revision)

Jul 22, 2026

- Peter Pelberg
    - ☑ ~~FILE ticket for specifying sampling requirement for headless VE technical pilot~~
        - ~~DECIDED: deterministic random sampling of pages at a subset of pages on multiple wikipedia, limited to pages in the main namespace~~
        - ~~OPEN: what % at how many wikis~~
        - ~~OPEN: what will the scaling path look like from start to full scope of experiment~~
        - ~~OPEN: what metrics/criteria we will use to evaluate experiment (SRE)~~
        - ~~OPEN: what performance will we need to see in order to move forward with controlled experiment?~~
    - ☑ ~~COMMENT on T431497 requesting new service~~
        - *See: T431497#12151902*
    - ☑ ~~FOLLOW UP with Riku RE security review~~
- Andrew Otto
    - ☑ ~~FILE ticket and work on for "Create headless VE docker image"~~
        - T433514 Headless VE - Docker image pipeline
    - ☑ ~~WORK on T431499~~
        - For Eric Evans once T431497 New Service Request: Headless VE is done.
- Guilherme Gonçalves
    - ☑ ~~ASK SRE about priority and sequencing of T431499~~

Jul 15, 2026

- ☑ ~~Guillherme: share decision to move forward with guilherme's pilot~~
- ☐ Peter: define experiment scale so that SRE can estimate resource requirements based on initial experiment: T432427
- ☐ Peter: talk with Julie to learn if proposed experiment would work with TestKitchen
    - How might we tell the experiment that a page has or has not been sampled as part of the suggestion count generation
    - Can we omit all pages / sessions from the experiment where no suggestion counts are available?

Topics

- AO: Headless VE per wiki deployments + mediawiki-config changes.
    - Headless VE is per wiki. Do we need 1000 deployments of it? Can we make the a single service deployment somehow respond for any wiki?
    - Do we need Headless VE to be deployed along with mediawiki-config backports and the MW train?
- Early explorations with lightpanda.io https://phabricator.wikimedia.org/T431726#12109228
    - ES: Early experiments are showing a 80-90% drop in CPU/memory usage which is very promising.
- AO: LAC cache misses on reads.
    - Related thread

Meeting Notes

Aug 19, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Peter Pelberg David Chan Mateus Santos Andrew Otto Özge Karakaya Aiko Chou David Lynch ~~Guilherme Gonçalves~~ ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Ed Sanders~~ ~~Val Puffet Michel~~ ~~Lauren de Lench~~ ~~Jesse Hathaway~~ ~~Isaac Johnson~~

Notes

- Headless VE help?
    - AO: image is published! https://docker-registry.wikimedia.org/repos/mediawiki/services/editcheck-headless/tags/
    - DL: made helm chart.
    - DL: LAC needs hooked up. T431499 Linked Artifact Cache - connect to Headless VE suggestion counts lambda
        - AO: will keep poking this one.
    - DL: will lambda timeout stuff?
- DC: how might we do this in node instead of headless VE?
    - DC: nodejs code that can pull from ResourceLoader, and run MW JS modules.
    - DC: Fiddling to do to make sure config is pulled from appropriate wikis.
    - DC: would we run against live enwiki? Or release the resourceloader code itself. Cache it? Trying not to answer these yet, but just seeing if it is even possible.
    - DC: looks possible, fiddly stuff but I think we can do it. Still unknowns so not sure.
- https://www.mediawiki.org/wiki/User:DChan_(WMF)/Dismissal_of_edit_suggestions

        - Suggestion id needs to be fuzzy, depend on stuff not specific to individual suggestion.
- https://phabricator.wikimedia.org/T430676#12168562
    - Discussion about relation to edit suggestion
- https://en.wikipedia.org/wiki/Wikipedia:Village_pump_(WMF)#AI-generated_edit_suggestions
    -

Action items

- [ ]

Aug 13, 2026 |

ACTIONS

- [ ] Kavitha Appakayala: confirm with Chris Danis and Giuseppe Lavagetto

PURPOSE

1. Define SRE point of contact for Editing/DPE while Jesse out
2. Set preliminary deployment date

NOTES

- Jesse as main point of contact
-
- .
- .
- .

Aug 12, 2026 | Headless VE Technical Pilot

Attendees: Giuseppe Lavagetto David Lynch Guilherme Gonçalves Chris Danis Peter Pelberg Andrew Otto Val Puffet-Michel David Chan Caro Medelius ~~Kavitha Appakayala~~ ~~Jesse Hathaway~~ ~~Lauren de Lench~~ ~~Karen Hernandez~~ ~~Max Binder~~
Attached files: Prep Pantry: Server Side Edit Suggestions

ACTIONS

- [ ] Service Ops

- Chris Danis to work with SRE to create dashboard needed to monitor headless VE technical pilot
    - *Will pull in a few graphs from consoles that already exist and locate them in a centralized place*
- Deploy pilot
    - Add entry in kubernetes cluster and load balancer (puppet)

☐ Editing
- to create helm charts; SRE to review
- to finish security review
- Send SRE SCROLL steps intentionally skipping for pilot, on ticket | T432715
- Ask service ops (NAME) to review helm chart(s)
- ~~Talk with Kavitha Appakayala about timing and point of contact over the next ~2 weeks~~

PURPOSE

1. ~~**Align** on the sampling requirements for the Headless VE technical pilot. See proposal.~~
2. ~~**Assess** state of SCROLL/EditCheckHeadless~~
3. **Define** technical pilot start date and resolution for any remaining blockers.

LINKS

- Prep Pantry: Server Side Edit Suggestions
- SCROLL/EditCheckHeadless
-

NOTES

- GL: that scaling schedule seems to be aggressive.
- CD: be careful of traffic over the weekend.
- DL: Broadly: we don't care about the percentages or timing, we just want to make sure we gather whatever information we need.
    - DL: (And for the pilot purposes, this isn't us getting to the end and just leaving it turned on -- once we have the data this gets shut down and we go actually finish all the SCROLL stuff.)
- GG: if something goes wrong on day3, we will roll back
- Schedule
    - Monday = 1% of edits to pages in the main namespace at en.wiki
    - Tuesday = 5% of edits to pages in the main namespace at en.wiki
    - Wednesday = 10% of edits to pages in the main namespace at en.wiki
    - Following Monday = 25% of edits to pages in the main namespace at en.wiki
    - Following Wednesday = 50% of edits to pages in the main namespace at en.wiki
    - *Note: SRE will decide whether resource consumption is sustainable enough to move to each subsequent scaling step.*
- SCROLL

    - DL: what we'll need to get done (before the pilot?): the security review, dependencies, monitoring (we have an MR for this), logging
- .
-

Aug 11, 2026 | Edit Suggestions Fortnightly Product Sync

Attendees: Guilherme Gonçalves Peter Pelberg Lauren de Lench David Chan Mateus Santos Andrew Otto ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Ed Sanders~~

Notes

- Potential "Phase 2" workstreams
    - Surface Edit Check-powered suggestions within Home *without* any specifics about the content the suggestion is relevant to. *Read: this article has ≥1 of these suggestions within it.*
    - Show "fully described" suggestions on any surface. Where "fully described":
        - Suggestion type
        - Suggestion description
        - Content suggestion is relevant to/an annotation on
        - Suggestion actions
    - Enable people to hide suggestion for their future selves and other people
    - Invalidating suggestions
    -

Action items

- ☐ Peter Pelberg : spell out potential areas of investment for phase 2

Aug 5, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Peter Pelberg Lauren de Lench David Chan Mateus Santos Andrew Otto Aiko Chou Isaac Johnson David Lynch ~~Guilherme Gonçalves~~ ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Ed Sanders~~ ~~Val Puffet-Michel~~ ~~Özge Karakaya~~ ~~Jesse Hathaway~~

Notes

- Andrew Otto list work that needs to be done for DE1.1.2
    - T433514 Headless VE - Docker image pipeline - Owner: Editing. But, Andrew is doing this.
        - Jesse already +1, just needs a new repo home.

- **T434109: Create Headless VE Helm chart & deployment** - Owner: Editing, with heavy consultation from SRE
  - T428174 Standard helm chart for simple service-utils nodejs apps would help.
  - Editing + Service Ops (Jesse Hathaway) should work closely on this
  - Is this required for the SCROLL process?
    - Yes
- T431499 Linked Artifact Cache - connect to Headless VE suggestion counts lambda - Owner: Data Persistence. Eric Evans TODO (he is OOO until mid August)
- Enable T432733 Server Side Edit Suggestions - Linked Artifacts Cache precompute job - Owner: DPE (code done)
- T433024 Implement sampling requirements for Headless VE technical pilot - Owner: Editing.
  - Proposal:
    - Day 0 = 100% edits at test wiki
    - Day 1 = 1% of edits at en.wiki
    - Day 2 = 5% of edits at en.wiki
    - Day 3 = 25% of edits at en.wiki
    - Day 4 = 50% of edits at en.wiki
  - Metrics of evaluation
    - Kubernetes cluster load
    - Job queue drain rate (remains faster than it fills)
    - LAC resource usage?
  - Open Q: how to set/scale headless VE replicas for evaluation? Up to SRE.
  - Possible to “synthesize” a production environment instead of doing it in production?
    - Latency and CPU usage of a single replica → but do we need to hook up the whole jobqueue…
    - Not hooking up reads
    - We don’t yet know which combinations of articles + suggestions would be more resource intensive - wouldn’t be comfortable vouching for this approach representing reality
- T433530 Backfill Edit Suggestion Counts in Linked Artifacts Cache for technical pilot - Owner: DPE. Andrew TODO.
  - This is needed before any read mode use, but not really before.
  - How long would we need to wait without a backfill?
    - Counts are determined when the article is edited - so could be months, but for most popular articles would be pretty much immediately (days, hours, or minutes)
  - How complex would it be to scope the backfill to meet suggestions of the sort we just sketched out?
    - Not relevant to technical pilot

        - Not too technically difficult, we should just do it → a lot of pages out there that aren't edited often but the presence of suggestions could have a high likelihood of being actioned
        - Need to think about how fast/slow we do it
    - This should be a subtask of the read mode epic
    - Worth scoping to articles that haven't been edited in x timeframe?
        - Simpler to scan over all pages
        - If we hit pages that are already computed, it will just hit the cache and we skip backfill
-

Action items

- [ ] David Lynch - new editcheck-headless repo outside of VisualEditor extension
- [ ] Schedule meeting (early week of 10 Aug) with Andrew, DL, DC, LDL, Peter, Jesse, Giuseppe Lavagetto (or someone else explicitly delegated to make decision) re proposal + eval criteria for T433024
    - [ ] Define experiment evaluation criteria T432286
    - [ ] Commit to scaling schedule T433024
- [ ] Peter Pelberg update T433024 with proposed scaling schedule and metrics for evaluation

Jul 29, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Guilherme Gonçalves Peter Pelberg Ed Sanders Mateus Santos Andrew Otto Özge Karakaya Aiko Chou Isaac Johnson ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Val Puffet-Michel~~ ~~Lauren de Lench~~ ~~David Chan~~ ~~Jesse Hathaway~~

Notes

- T431497 New Service Request: Headless VE
    - This is between Editing team and ServiceOps. DPE is trying to keep convo going, but those teams need to drive it.
- T432733 Server Side Edit Suggestions - Linked Artifacts Cache precompute job
    - Patch ready for merge. One comment nit. Editing will merge, and we can enable with config once the other services are deployed.
- T433514 Headless VE - Docker image pipeline
    - Otto will try to push this forward asap.
- ES: we would prefer to know if something is cached to decide to show it. We can use a short timeout which would trigger a cache miss, but really it would be better to avoid triggering the cache miss and just fetch and get a 404 or whatever if the value is not cached.

Jul 22, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Guilherme Gonçalves Peter Pelberg Ed Sanders Lauren de Lench Mateus Santos Andrew Otto Özge Karakaya Jesse Hathaway ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Val Puffet-Michel~~ ~~David Chan~~ ~~Aiko Chou~~ ~~Isaac Johnson~~

ACTIONS

- Peter Pelberg
  - [ ] FILE ticket for specifying sampling requirement for headless VE technical pilot
    - DECIDED: deterministic random sampling of pages at a subset of pages on multiple wikipedia, limited to pages in the main namespace
    - OPEN: what % at how many wikis
    - OPEN: what will the scaling path look like from start to full scope of experiment
    - OPEN: what metrics/criteria we will use to evaluate experiment (SRE)
    - OPEN: what performance will we need to see in order to move forward with controlled experiment?
  - [ ] COMMENT on T431497 requesting new service
  - [x] ~~FOLLOW UP with Riku RE security review~~
- Andrew Otto
  - [ ] FILE ticket and work on for "Create headless VE docker image"
  - [ ] WORK on T431499
- Guilherme Gonçalves
  - [ ] ASK SRE about priority and sequencing of T431499

PURPOSE

- Review state of past actions
  - Page sampling vs user sampling
    - We learned in 1st controlled experiment of suggestion mode: only 15% of edit suggestions caused someone to see a suggestion, which made it difficult for us to see the broader effect of suggestion mode/reach statistical power, exposure was low.
    - So: how do we make sure the treatment group doesn't have that same shortcoming?
    - We're doing article-level sampling. If you're outside the article sample, you'll just get 0 results. If you're in the control group, you won't see any suggestions → regardless of whether there *were no suggestions available* or if they were suppressed to the user (because they're in the control group)
    - What if we're doing pageID sampling, and we realize we can only compute 10% of suggestions available bc of resources available?

        - just don't emit the instrumentation event if the page is not in sample.
        - We would only consider pages that have a suggestion count available
        - Are you sampling randomly or within a subset of highly viewed pages?
        - The experiment wants as many pages sampled as possible, so is it the best use of time to start estimating how much it can handle?
        - We need to sample on page because we want the page count to be deterministic and consistent - how does precompute give us information about the resource constraints? E.g. if the articles we sample just happen to have no suggestions
        - If we're choosing 1% of pages and it's random… it's a big number for statistical sampling
        - Computed based on pageID, but needs to be pseudo-random
        - If we sample randomly on pageID, this will give us a representative sample that will give us the information we need to answer: How greedy is this thing?
        - To what extent do we go bottom-up, versus trying to estimate ahead of time? We asked SRE for a budget but they don't know, that's why we're going with a bottom-up approach. Let's not try to estimate ahead of time, ideally we would have 100% for one wiki and see where we land (ideally it will be much more than that)
        - OTOH the total # of edits per second is <10 - we're not doing wikidata, it will only be a handful per second.
        - the assumption is that a deterministic random sampling of pages will be representative enough in edit frequency to give us a good measure of HeadlessVE greediness
        - Peter's preference: do a subset of pages from multiple wikis, because that will most closely mirror what we want to test
        - Takeaway: page sampling can't be done in Test Kitchen but we can set it up ourselves
- Align on headless VE pilot launch (steps and timing) | T432286
    - Headless VE docker image → new requirement, Andrew Otto can create task
    - https://phabricator.wikimedia.org/T431497#12102323
        - Scott French noted this is blocked??
        - According to Roan Kattouw the security review does block the technical pilot
        - Because editing owns headless VE, it's on editing to keep poking
        - We aligned to move forward with SRE, what's next → give them some indication on timeline
-

Jul 20, 2026 | Headless VE Security Conversations

Attendees: Andrew Otto, Ed Sanders, Roan Kattouw, Peter Pelberg

ACTIONS

- Ed
  - ☑ ~~File task headless VE for security review (architecture)~~
    - https://gerrit.wikimedia.org/r/c/mediawiki/extensions/VisualEditor/+/1308570/13
    - Emphasize review of architecture
    - T432708
  - ☐ [Maybe] submit lightpanda review
- Peter
  - ☑ ~~Reach out to Riku Silvola~~

PURPOSE

July 14th instance of Edit Suggestions Fortnightly Product Sync surfaced a need to determine needs and timing for security review. Folks wanted to sync with Roan first, while Lauren and Ed put together an L3SC request.

Timezones for this meeting offer a narrow overlap, so this seemed the best slot. If you see a better one, please move it!

NOTES

- Internal proxy settings needed b/c browser won't understand en.wikipedia.org
- Formal security review needed; need to make sure it can't do scary things
- ES: runs logged out and does account-based filtering later
- AO: what are the security surfaces/risks?
- RK: comes down to this being new code deployed in a new way…more so about maintainability, dependencies, etc.
- RK: standard vendor review needed
- RK: main thing i'm worried about is this thing going haywire and DOS from within
- RK: we probably want to disable site and user scripts on this special page…we don't want people on-wiki to dictate the code this special page can run
- RK: you have to give this browser relatively wide scope to fetch things
- AO: ok, so in essence, the scope of the browser is broad enough to create uncertainty around what code this does / does not execute which I can see would warrant review
- RK: main thing i'm scared of is you're deploying a general purpose thing which makes it difficult to put clear lines around it. So i want to think about what a sufficiently secure box/contianer might be
- ES: we could also learn from https://wikitech.wikimedia.org/wiki/Proton

- RK: code is not the most interesting thing, it's the diagram and the measures we take to try keep the headless instance contained and prevent it from doing things that are bad

Jul 15, 2026 | Edit Suggestions Fortnightly Product Sync

Attendees: Guilherme Gonçalves Peter Pelberg Ed Sanders David Chan Andrew Otto Jesse Hathaway Aiko Chou

Prep Pantry: Server Side Edit Suggestions

ACTIONS

- [ ] Guillherme: share decision to move forward with guilherme's pilot
- [ ] Peter: define experiment scale so that SRE can estimate resource requirements based on initial experiment
    - Proportion of logged in users
    - Proportion of pages
    - # of wikis
- [ ] Peter: talk with Julie to learn if proposed experiment would work with TestKitchen

NOTES

- JH: it seems to me the test will suss out the "resource requirements" and not "maintainability" or "security"
- JH: on the maintainability piece, a colleague releases Proton 2x month so that's a benchmark to keep in mind.
- JH: RE security, i don't have a good sense for this. I think we'll want to do the chromium updates regardless.
- JH: I think someone else needs to comment on the security piece
- GG: sounds like we generally agree on this rollout schedule to assess resource requirements
- ES: you mentioned Proton…this is the PDF service? (Uses Chromium to render the page) Do we know the sort of volume that's seeing?
    - JH: no…
    - ES: it seems like a direct comparison and perhaps validates the approach we're proposing
- ES: RE running arbitrary wikitext…let's assume we're using Parsoid HTML. I think we need to assume that's safe. i.e. if there's a vulnerability in Parsoid HTML that's a bigger problem than what we're building.

Jul 14, 2026 | Edit Suggestions Fortnightly Product Sync

Attendees: Guilherme Gonçalves, Sucheta Salgaonkar, Peter Pelberg, Max Binder, Ed Sanders, Lauren de Lench, David Chan, Mateus Santos, Andrew Otto

Notes

- Headless VE
  - We're blocked without SRE sign-off
  - Consider deploying only a certain subset of suggestions?
  - Review Headless VE pilot
  - PP: we need Jesse's feedback on the security thing. Also SRE giving us a checklist to use before we launch - this is a reaction to past experiments that were sprung on SRE and kept in an "MVP" status
  - ES: looks good to me
  - GG: re security review - SRE can flag operational constraints, but for something like lightpanda.io it would need a full review
  - MS: To easy in the conversation with SRE you might want to consider starting fresh with SLOs as soon as possible.
  - AO: why is the security point such a big deal? It's because we're doing LAC compute in real time. If suggestions were to be isolated from any online request path - e.g. if batch compute were sufficient, we would need less scrutiny. Don't totally understand the security constraints. Generally understand headless browser check for concerns, but we don't security review every library we use…
  - ES: +1, we're using it like a library. It doesn't have to be visible to users at all, a completely sandboxed environment. Just calling one URL. User input is literally "which article should we process?"
- GG/AO: Review recompute triggers.
  - Prep Pantry: Server Side Edit Suggestions
  - Right now we're pretty handwavy re: suggestions going out of date, but are trying to map out user actions that can cause them to go out of date. What is the current behavior in phase 1, accept what we're able, and prioritize fixes for phase 2.
  - PP: this looks awesome. Biggest thing I want to do is step through this with Ed + Editing eng. Want to understand impact scenarios where recomputation would not affect…. Don't see change in category. Taht would be in revision ID?
  - ES: categories are often within the page itself, so yes that would be in revision ID… templates will not be considered because they're unlikely to change(?)
    - Missing template parameters
    - Categories that are included via templates
- Experiment task breakdown
  - Data we have is sufficient for phase 1

    - Data Eng owns precompute mechanism, and is concerned with 1) does a task exist already; 2) which task do I use as the parent?
    - [made some changes to the task tree in meeting]
    - At discord call people were concerned about showing suggestions that were invalid/unhelpful to large #s of people.
    - Definitions?
        - Invalidated suggestion =
        - Rejected suggestion =
    - Andrew will ignore the phase 2 task tree for now!
- GG: Should we further break up Phase 2?
    - EditCheck suggestions vs non-Edit Check suggestions?

Action items

- [x] ~~Guilherme Gonçalves: Share proposal with Jesse with specific ask to review~~
- [ ] Ed Sanders: Fill out and submit security intake form (+ Lauren)
- [x] ~~Max Binder: Schedule meeting with Security: Ed, Andrew, Roan, Guilherme (optional), Peter (optional)~~
- [ ] Ed Sanders and Peter Pelberg to step through recompute scenarios and update impact column

Jul 9, 2026 |

Discuss about T428491 - editcheck headless mode options

Attendees: Janis Meybohm Eric Evans Chris Danis David Lynch Caro Medelius Andrew Otto Ed Sanders Giuseppe Lavagetto Kavitha Appakayala Jesse Hathaway Guilherme Gonçalves Val Puffet-Michel Peter Pelberg Lauren de Lench ~~David Chan~~ ~~Sonja Perry~~ ~~Moritz Mühlenhoff~~
[transcript in calendar event]
Notes

- Context: Editing team has built a framework for creating suggestions & checks that runs within VE
    - https://www.mediawiki.org/wiki/Edit_check
    - Now Editing is trying to expose suggestions *outside of VE*
    - Working with DPE re: building data pipeline, storage, processing, and serving
    - Prep Pantry: Server Side Edit Suggestions
    - AO: some terms:
        - **edit check** - in VE on new / unpersisted text the user is typing in the editor
        - **suggestion mode**: opening VE with edit checks evaluated on existing article text.

            - **server side edit suggestions** - precomputing edit suggestions, persisting them, and serving them outside of VE, e.g. on page read or user dashboards.
        - DL: (There's a system called "edit check" which lets you create "checks" which look at the document and see whether they apply. By default, checks are only evaluating changes the user has made while editing. "Suggestion mode" applies checks to the entire document, not just the changes.)
        - ES: main use case for this new service is showing suggestion counts on read mode, not just edit mode Headless VisualEditor for EditCheck
        - [PP notes] Key concerns
            - Maintainability
            - Security concerns
            - Resource requirements
        - Potential path forward
            - Finite lifespan experiment
            - Smaller scale
                - Subset of wikis
                - Sample of articles at all or large wikis
                    - Randomize based on revision
                    - Randomize based on article title
                - Progressively scale
- Review Alternatives to Headless VE considered
-

Action items

- [x] ~~PP to start conversation with marshall, sonja, kavitha RE hypothesis~~
    - [ ] Hypothesis home
- [ ] Editing + DPE: propose experiment design in collaboration with SRE ambassador (Jesse)

Jul 8, 2026 | Edit Suggestions Weekly Engineering Sync

Attendees: Jesse Hathaway Guilherme Gonçalves Peter Pelberg Max Binder Ed Sanders Lauren de Lench David Chan Mateus Santos Andrew Otto Özge Karakaya Aiko Chou ~~Sucheta Salgaonkar~~ ~~Val Puffet-Michel~~

Action Item follow ups:

- Andrew Otto : list the things that can cause suggestion counts to be invalid.

    - Done here.
- Andrew Otto: investigate how to incorporate config more. Possibly store config per wiki in LAC?
    - Are we sure this config should be stored in LAC and considered 'derived data'?
    - A pragmatic definition of 'derived data' scope is: a data pipeline that MW itself is not good at handling.
    - Idea: could MWEditing store/cache per wiki config using an existent MW mechanism? Perhaps WANCache?
    - Ed has a working prototype of the headless VE API endpoint
- Andrew Otto: think about/offer proposal for local development workflow for interacting/testing with LAC
    - AO: how much do we really need this?
    - LAC in beta might be useful for integration testing of the whole pipeline. Would be some burden for DPE + Data Persistence, but we could do it.
    - Do we need local LAC for development? Two use cases:
        - Product dev HTTP fetching from LAC frontend endpoint.
            - This can be mocked.
            - /edit_suggestions/{wiki_id}/{page_id}/{rev_id}
        - Developing gRPC lambda (Headless VE).
            - Really just need to validate that the service conforms to the LAC lambda gRPC contract.
    - OTOH, it would be nice to do local MW development and see the whole flow. We'll have a Job that warms the cache, and it would be nice to test this.
    - Perhaps we could make a docker compose recipe for hoarde. It would be nice if Cassandra was not needed for local dev.
    - Created T431343 Linked Artifact Cache - Local MediaWiki development
- Guilherme Gonçalves: Welcome Jesse!
- Headless VE & ServiceOps?
    - Thread with SRE
    - ES: Headless VE is wild, I understand why folks on first glance object. Needs more discussion.
    - GG: SRE ambassador is point of entry for SRE.
    - GG: Also saw mention of SCROLL process.
    - GG: DPE & Data Persistence are a little agnostic to the lambda compute mechanism. Editing needs to work with ServiceOps for service deployment.
    - GG: Def a process gap here. Maybe a month ago we should have involved SRE.
    - AO: should we dive into technicalities of Headless VE?
    - JH: Would want Giuseppe involved in convo.
        - https://phabricator.wikimedia.org/T428491#12098835
        - Joe: Hi, sorry, let's take a step back.
        - Running headless browsers in production is highly problematic from multiple points of view, hard to maintain, and very resource hungry. If we need to do this at all, we are going to have to pick up the concept in T424872 from where it stopped being worked on.

        - I think we should really look at alternative approaches, if at all possible - as we're already struggling to maintain the only service that uses a headless browser in production.
        - So, can you please explain clearly why a headless browser is needed in this process? Is there a reason we can't obtain the same results using a pure nodejs service?
        - If at all possible, we should avoid using full browsers in production.
    - Riccardo Coccioli is the primary SRE ambassador for DE1, and he is on sabbatical
    -
    - ES: understand there will be performance concerns, when we initially pitched we thought “is it crazy, will it actually run?” and have seen in experiments how the performance goes up/down. Seeing how sessions respond to requests that come in. Certainly things we can do to get to a point where we can handle the volume of events we need to (e.g. queueing up requests). On the surface it looks like 1 session per edit and I understand why this would look like a problem. We need to get into the details to work out whether it’s expensive, or *too expensive*
    - JH: and since we run our own hardware we’re concerned with additional strain. Giuseppe worried about keeping chromium up to date - it gets out of date really quickly with wm-stable(?) - big dependency to wrestle with from a container perspective. I suspect Giuseppe will have other concerns than just the raw implementation
    - ES: not a public-facing API, and not running arbitrary code. It’s a private api that’s being used to fill a cache, on a very distinct subset of code. Once it’s running the Editing team can respond, e.g. if there’s a version of chromium that’s 10 years old we can work around it
    - JH: volume and # of requests/second will be interesting to know
    - AO: roughly a request on edits to main content pages (10 per second or so)
    - ES: doesn’t necessarily translate… leave chromium instance open
    - GG: possible we come out of this with an agreement with long-term vs short-term direction. We have a product roadmap, and phase 1 is essentially: we only care about headless VE to expose the count of suggestions on a page. In this stage we accept a lot more technical debt, but in phase 2 we need to break it out and build it in a way that can handle a lot more data
    - AO: Do we have an idea how long it would be to take edit compute suggestions… ? A year? A quarter?
    - ES: to me it feels quite intractable. We’ve experimented with pulling some of VE core in a node service, plus mediawiki integration - it need sto understand the whole context of mediawiki and various extensions, config….
    - AO: like running a mw JS client but not in the brower, yeah.
    - ES why headless became a solution. Somehow rewriting all the checks, duplicating the logic, and gettin gthat to sync up. We’re looking at gadget-based checks too. All checks are written against the VE data model, need to figure out how to work them through the parser html and that will be really messy

    - DC: running the model separately from VE and node, would be a long-term thing and not something we could deliver in a couple of weeks.
    - AO so what is the long-term? A couple quarters?
    - DC: all the issues Ed mentioned means it would be less than trivial
    - AO: all this mw js, resourceloader, and mw config, all of which happens client side, it's all very coupled
    - MS: curious as to why parsoid html wouldn't be enough?
        - ES: it isn't lack of information in html, the html it transformed into VE data model for editing purposes.
        - ES: we built edit check on VE data model. That is where the live doc lives. We can respond to keystrokes.
- PP: who do we need to talk to, and when, so we can make sure everyone has what they need to make a sound decision? And we can start thinking about fallback options where needed (even if it means major changes to the product requirements)? First piece is educating folks about the thought put into it, what the constraints are, make sure it's legible
    - SRE: Jesse, Giuseppe, Kavitha, Eric
        - What types of information would equip you with what you need to know to work with us?
            - JH SCROLL document is brand new; to make sure new production services have SRE support and ready to go to production - but supposed to happen after architectural decisions are made. Worth reading over, but not blocking the initial architectural pieces we're discussing now.
- PP: am I assuming correctly that there *is no fallback option?* I.e. we've arrived at our proposed architecture for a reason and other paths forward are not viable?
    - AO: we're using VE as precompute mechanism, but there are checks that don't use VE as precompute (e.g. revise tone, add an image, add link) - so include only the ones that are service-based. Thinking about how we do the LLM ones, consider a mechanism that is *not a browser*. For some suggestions we use VE as "the source" but…
    - E: maybe a subset of checks that are easier to convert to … so we're going to write a bespoke server side suggestion for reference check, or some similar subset of checks. If we could never do headless VE, we could go in that direction - but we'd have more situations where a suggestion is reported, or missing. There are ways we can get more out without headless VE
    - PP: so biggest assumption → we need a count of all available suggestions regardless of their source.
        - Path 1 is keep doing what we're doing, but that's dependent on SRE.
        - Path 2 is "everything available in the client" and scope it to what we have available there. Concern is supply, and ensuring there are enough suggestions across enough articles.
        - Path 3 is a specific subset of suggestions stored elsewhere(?)

    - Should we run down headless VE to make sure it's viable, or shift from there? Or try an alternate path in parallel?
    - ES: solution should plug into other architecture we're planning
    - DC: and if we have headless VE running in nodejs, we have the option of scaling up in the future. Current headless VE runs inside a browser. We're talking about running it in a node js server…. And then we have the option to

Jul 1, 2026 | Edit Suggestions Fortnightly Engineering Sync

Attendees: Guilherme Gonçalves Peter Pelberg Ed Sanders David Chan Andrew Otto ~~Sucheta Salgaonkar~~ ~~Max Binder~~ ~~Val Puffet-Michel~~ ~~Lauren de Lench~~ ~~Mateus Santos~~

Notes

- Roadmap the timeline (see Requirements/Server-Side Suggestions)
- Ownership and outline for XXX (Edit Check: create a mechanism to initiate runs of VE headless mode)
- Ownership and outline for T430077 (Edit Check: create a way to store and serve suggestions from VE headless mode)
- Editing: what (if any) ability do we have to count the number of suggestions acted upon within an edit session for the repurpose of landing someone who just publishes an edit back in read mode with the suggestion count decremented by the number of suggestions they actioned.
- Prep Pantry: Server Side Edit Suggestions Phase 1 counts discussion
    - ES: HVE + LAC interface. Product requirement: being aware of config for every check. Suggest a link task. E.g. only show to users with >100 edits.
    - AO: google comment discussion
    - ES: Idea: store relevant configs with suggestions.
    - AO: can we fetch relevant configs at read time rather than compute and store it?
    - ES: maybe, but it would be easier to just store it
- ES: three places to configure checks/suggestions
    - Local settings
    - Within the code (where defaults are set) this is hard because we need to load edit chekc to know what that value is
    - on-wiki .json file
- GG: three threads
    - LAC development
    - Storing configs
    - Invalidation – for how long can we assume only new revisions invalidate articles

Jun 30, 2026 | Edit Suggestions Fortnightly Product Sync

Attendees: David Chan Ed Sanders Guilherme Gonçalves Lauren de Lench Max Binder Mateus Santos Peter Pelberg Sucheta Salgaonkar

Agenda

- Review decisions / requirements for phase 1
    - Any time the UI requests a suggestion and there is an outage, OR we have determined there is a failure by way of an established SLO
        - We need to log something
        - We need to take an action
        - https://phabricator.wikimedia.org/T421755
- Roadmap the timeline (see Requirements/Server-Side Suggestions)
- Ownership and outline for XXX (Edit Check: create a mechanism to initiate runs of VE headless mode)
- Ownership and outline for T430077 (Edit Check: create a way to store and serve suggestions from VE headless mode)

Action items

- [ ]

Jun 17, 2026 | Edit Suggestions sync | ML <--> Editing <--> Product Derived Data

Max Binder , Guilherme Gonçalves , Lauren de Lench , Sucheta Salgaonkar , David Chan

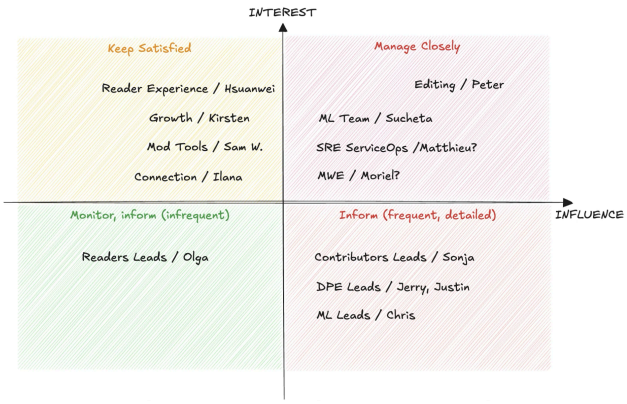
ACTIONS

- [x] ~~**MB**~~
    - [x] ~~Schedule recurring bi-weekly meeting for product (roadmapping, etc)~~
        - [x] ~~Add agenda item for roadmapping the timeline~~
    - [x] ~~Schedule recurring meetings for engineers (Add David Chan, Ed, Peter, Guilherme to technical forum)~~
        - [x] ~~Later expand to CTT~~
    - [x] ~~Add folks to ST6.2 steerco meeting and Slack~~
        - [x] ~~Sucheta, Guilherme, Mateus (optional)~~
    - [x] ~~Retitle channel from #server-side-suggestions → #edit-suggestions~~

- [ ] LD
    - [x] ~~Make sure Sucheta, Guilherme are invited to DE1.1 steerco, Mateus optional~~

PURPOSE

1. Nature of the work
    a.
2. RACI: roles and responsibilities
3. Timeline: what is needed when?
    a. Launch controlled experiment (reader experiment part 1 - suggestion count)
        i. August 1st (aggressive? Almost certainly)
            1. July 1st is the first draft
        ii. Highest traffic experiment the Editing team has done
    b. Deploy end-to-end PoC as Beta Feature
    c. Start running Headless VE to start backfilling pipeline
    d. Align on design for PoC data pipeline + serving.
    e. July 1: first draft of data pipeline
    f. Headless VE API
    g.
4. Dependencies:



    a. Search
    b. DPE
    c. Editing
    d. ML - no dependency, but keep informed
5. Communication: what might we need to keep stakeholders informed?
    a. *E.g. recurring standups? Eng to work through technical details/problems. Eng + Product + Design (?) to make tradeoff decisions. Potentially repurpose "Tone Check"*
    b. One default option is to have folks from each others' steercos
        i. We also need a place for ML to stay in touch with us, steercos are a good place for that, perhaps have a representative at the other meetings with engs.

    c. If we want to have deeper technical discussions, we could have dedicated meetings for this specific work so that we don't take over OKR steercos. A place for the engineers to review things.
    d. Slack: server-side-suggestions?
        i. Perhaps a rename is needed → what we're talking about is the "shelf" re: how we're generating suggestions at scale, suggestion agnostic
        ii. Suggestions-as-a-service = anyone, from any experience can pull suggestions
        iii. or simply #edit-suggestions ?
    e. Use the non-steerco / unified channel to talk about work, steercos for OKR specifics
6. Roadmap: what do y'all need from PMs in order to identify blockers, dependencies, etc.?
7. Define processes/practices that will help us remain in sync.
8. Artifacts
    a. *What are our definitive sources of truth?*
    Prep Pantry: Server Side Edit Suggestions ?
    Requirements/Server-Side Suggestions ?
9.

NOTES

-

May 18, 2026 |
LLM Generated Editing Suggestions Architecture

Attendees: Aiko Chou Andrew Otto Guilherme Gonçalves Ilias Sarantopoulos Özge Karakaya David Lynch

Model-Generated Editing Suggestions
Guilherme's diagrams:
https://excalidraw.com/#room=d892cc328dde706dc4c7,t-a4_7zMNyT4yF_VQLpA7A

Notes

- [OK] Initial plan for enwiki.
    - Find 5 categories that are good for generating suggestions.
    - Trying to generate suggestions using LLMs.
- [OK] What is the current/target architecture for:
    - Pre-defined edit suggestions.
    - On-the-fly edit suggestions.
    - Which edit types are currently available in each category?
        - AO: related: Derived Data Entity Use Case Categories
        - Edit type: Semantic category of an edit e.g. MOS:GEO, simplify_language, NPOV.
        - AO: are these all in https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style ?
        - AO: answering myself: no: some are but not all.
        -
        -
- [OK] Preferred way (pre-defined/on-the-fly) for LLM Generated Editing Suggestions.
    - Offline**:
        - Data integration: we use html and we will need revision html in the datawarehouse e.g. daily, hourly
        - Invalidated/repopulated based on the most recent revision id.
        - Easier to control generated suggestions based edit types, number of suggestions to generate per day, surfacing the best suggestions etc. Let's say we want to generate N suggestions per day. We can create N*2 per day, score by quality and pick the top N. It should help to increase the quality of the suggestions and we have an additional filter.
        - Where should we store them?
        - Expected number of suggestions per day?
    - Online:
        - Slow to generate (even slower in thinking mode).
        - how to show a number of pending suggestions on the page.
        - No filtering.
        - May not be easy to personalize in the future.
        - No need to bother with invalidation.
- [OK] Editing suggestion data structure: What should be the schema?
    - Prior (location. How do we currently identify location? We have sentences from the given revision id.)
    - Suggestion
    - Edit type
- [OK] Edit types: do we need to define an editing template for each edit type? I see some edit types are yes/no, some lead to editing screen etc. Some of our edit types are only for section names.

- [OK] Can we deliver a list of suggestions in a file or save it to a preferred DB for the initial phase? So that the implementation does not block online evaluation and the architecture implementation can go in parallel.
- [OK] Where do we save the data to calculate the success rate of the suggestions?

Action items:

- [ ] Check with David and Eric for the search part getting something out for evaluation with users?
- [ ] Check with Isaac and C Scott etc. about standardizing suggestion schema/outputs.
    - https://en.wikipedia.org/wiki/User:Cscott/Ideas/A_Dozen_Visions_for_Wikitext/Annotations/Design_Space
    - https://wikimedia.slack.com/archives/C0B350MRBBM/p1778763098643019?thread_ts=1778683907.272089&cid=C0B350MRBBM

May 12, 2026 | Server-Side Suggestion Kickoff

Attendees: David Lynch, Caro Medelius, Ilias Sarantopoulos, Andrew Otto, Ed Sanders, Özge Karakaya, Andreas Hoelzl, Guilherme Gonçalves, Sucheta Salgaonkar, Peter Pelberg, Val Puffet-Michel, C. Scott Ananian, Bárbara Martinez, David Chan, David Causse, Justin Moore, Max Binder, Peter Fischer, Georgios Kyziridis, Eric Evans, Halley Coplin, Fabian Kaelin, Isaac Johnson, Lu Wilson, Nick “Quiddity” Wilson, ~~Mateus Santos~~, ~~Lauren de Lench~~, ~~Sonja Perry~~, ~~Aiko Chou~~, ~~Miriam Redi~~, ~~Amin Al Hazwani~~
Attached files: Kickoff/Server-side Checks & Suggestions

ACTIONS

- [ ] Editing Team (Val + Peter)
    - [ ] Share finalized UX requirements (Requirements/Server-Side Suggestions) with DPE for review.
    - [ ] Draft technical requirements via Prep Pantry, collaborating with DPE.
        - Template: (TEMPLATE) Prep Pantry: Intake for <Project>
        - Example: Prep Pantry: Intake for Attribution API Signals
    - [ ] Ensure all requirement-related questions (UX, technical) from slides are addressed.
- [ ] DPE
    - [ ] Review and offer feedback on requirements once Editing shares
    - [ ] Propose initial storage and serving architecture once aligned with Editing on requirements
- [ ] PMs (Guillherme, Sucheta, Peter)

    - [ ] Decide what hypotheses will house Phase 1 work and what KRs these hypotheses will be homed within
    - [ ] Sketch out delivery timeline

- [ ] EMs (Andreas, Ilia, Val)
    - [ ] Define eng leads for respective teams

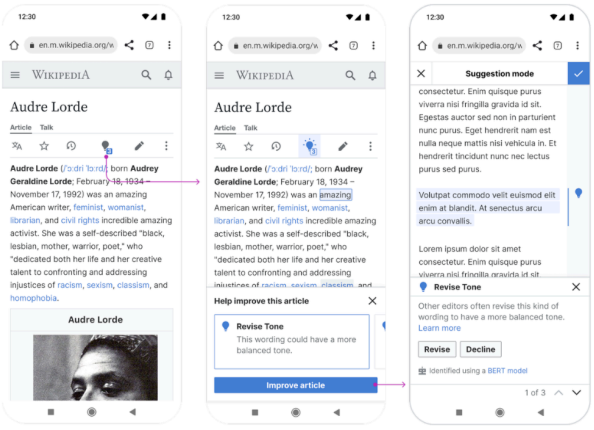
NOTES

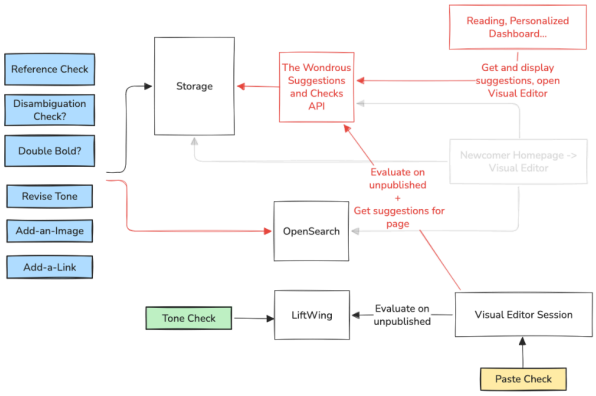
- DChan: Re: how our suggestion storage annotates the range to which the suggestion applies - the choice we make about this technical decision will have far-ranging implications on what kind of issues we can represent. It's a technical decision with product impact.
    - CScott: There are parts of an article that are hard to identify (due to templates etc.) - so from a higher level product perspective, if you can identify a specific wikitext region corresponding to your suggestion, it ought to be "easy" to represent in a portable way. Something that is sourced from wikidata via a lengthy collection of Scribunto modules, etc, and doesn't appear literally in the wikitext, will be harder to represent portably. (Just trying to give some non-technical insight about hard-vs-easy here.)
- [OK] What are our business goals?
    - [PP] At the highest level, trying to grow the population of people who are actively contributing to the wikis. This is the broader direction for all Contributor teams.
    - [PP] We have a few approaches to fulfill this goal, and this one focuses on reducing the barrier to entry for editing. We are trying to flip the current open-ended paradigm, leading with an experience that is explicit and structured and certain, giving people clarity about what's worth doing on the project.
    - [PP] Experienced volunteers are a big part of this too - we want to reduce toil, and one of the biggest challenges for them is figuring out what to work on. Some editing actions also have multiple steps. So we are also trying to figure out which suggestions are best fit for newcomers vs. which ones reduce toil for experienced volunteers.
    - [PP] Eg. Reference Check caused a 17.5% increase in newcomers adding references; but now we need to equip experienced volunteers to moderate all of these new reference additions.
    - [PP] Also thinking about how to make suggestions "portable" so we can show them in a variety of surfaces. We are trying to do some kind of match-making between users and suggestions.
- [PP] Talking about roles and collaboration
    - What is each team's role?
        - [GG] Editing: Product owners for the actual suggestions, technical owners for the business logic. There will be a platform with a missing piece and you get suggestions on the other side - Editing is the missing piece. Business logic doesn't necessarily include the *execution* of the logic, but instead defining the logic (at least in some cases) and DPE/SRE fulfills it.

        - [DC] Architecture of how we pass the business logic and suggestions is a shared responsibility.
        - [GG] How DPE gathers requirements:
            - https://wikitech.wikimedia.org/wiki/Prep_Pantry
            - This will be the next step for PP + GG
        - [SS] ML's role is to figure out how to make LLM-generated suggestions available in a way that is roughly aligned with the direction that DPE/SRE and Editing are going for non-ML suggestions
        - [GG] Search team will be involved because we have this filtering suggestions mode of access ("find all iterations of X suggestion"). But even at a step below that - how can we load all of the suggestions we're generating into OpenSearch? We'll need to know how to use Search infra for this project and also need to know Search SLOs that Editing can rely on
- Next steps:
    - DONE - PMs and EMs to figure out where this work is happening besides Slack
    - PMs will figure out requirements (Prep pantry) that will guide the technical discussions
    - PMs f

☐

Requirements/**Phase 2**





Key Stories

A range of workstreams could come to make up Phase 2. What we're aware of is documented below.

PHASE 1		PHASE 2		
Outcome	Explanation(s)	Impact	Workstream(s)	Notes
Low engagement: suggestion count doesn't draw people in	1) Suggestion count might not be discoverable or compelling 2) too few articles of		1) Improve discoverability 2) Increase suggestion supply	
High engagement, low completion: people click and drop-off before publishing	1) Publishing path is too effortful 2) Suggestions are unclear too onerous			

High engagement, high completion, low quality: people publish bad edits	1) Suggestions are low quality (high false positives) 2) People are acting too quickly			
High engagement, high completion, high quality: People publish quality suggestions	1) CTA works 2) Suggestions are compelling 3) Publishing path is clear	Scale impact	1) Show suggestion counts to more people (e.g. temp accounts, logged out) 2) Show suggestions in more surfaces	
TBD				

Meta

Server-Side Suggestions will need to perform in the following environments:

- [hard] Medium: **Web/Browser**
- [hard] Platforms: **Mobile** + **Desktop**
- [hard] Interfaces: **Visual editor**
- [hard] Languages: **All**

Requirements

Inbox

- Guilherme Gonçalves : I could see us separating computing suggestions (server-side) and deciding when to include them in a count/show them (client-side, dependent on Wiki- or user configs). | slack
- Invalidating suggestions
- Invalidating suggestions when editcheck community config is updated
    - Prep Pantry: Server Side Edit Suggestions
- IF we released a new Revise Tone model, how long are we willing to serve the suggestions generated by the old model?

User experience

.

Discoverability

.

Engineering

Meeting Notes/**Phase 2**

Aug 11, 2026 Edit Suggestions Fortnightly Product Sync

- GG: roadmap questions → any big takeaways re: dismissing suggestions?
    - This is a big problem, but solving this piece is premature right now
    - We're not hearing a lot of complaints re: suggestions are bad, or I want to get rid of them
    - Finish phase 1 and work on the atomic unit re: showing suggestions in other places
    - We are stepping away from page editing → so what are the mechanics around accountability and moderation?
    - Committing to an architecture prematurely has risks
    - Concerns about deploying everywhere without a mechanism for dismissal?
        - We need to offer a way for communities to turn things on/off
        - But we can't allow ourselves to succumb to all the hypotheticals!
        - Thinking back to 2013 and pushback on VE:
            - there was a strict deadline from mgmt, and low amount of openness to anything other than VE being deployed by a date, and there was no room for iteration or delay. Also some important feedback from volunteers that hadn't been considered.
            - Perception that VE was a lot more buggy than we sold it as → this sowed distrust that we're still seeing
        - We would be saying:
            - Suggestion mode doesn't currently offer a mechanism for dismissing → but we aren't aligned on what it even means to dismiss! It's WIP
            - Here's what you can consider putting in place in the meantime → equip them with information they need to decide whether they should block it or not
            - Be forthright about things that are "buggy" or "not fine" → don't pretend that it's not there
        - What's different now?
            - We have a lot of experienced editors using suggestion mode (few used or cared to use VE in 2013)
                - 15k experienced editors currently using
            - Experienced editors have seen newcomers using suggestions and their hearts are changed
            - Existential moment where editor counts are dropping - we need to make a change
            - EditCheck fundamentally changed the value proposition of VE → it not only lowers the barriers to entry, but also protects against things like copyright infringement

- Conversation so far has been very solutions-oriented, not problems-oriented → this signals some amount of drift from what's happening in the present
    - Pre-mortem: look at risks together and potential mitigations, from very small and specific to larger and longer-term
    - DavidC advocates for making space so that we don't run into a dead end, even if we don't travel the whole road re: dismissals
- Phase 2 definition → other workstreams have entered of a similar complexity
    - Data modeling is a complicated problem, so would be helpful to split what we want for phase 2 vs phase 3/beyond
    - "Invalidation" = When does a suggestion that was previously computed become invalid?
- Several related scenarios related to engagement volume:
    - Too few people engage with the counts. Insufficiently discoverable. Need to experiment with a different way of surfacing them
    - Too many people engage with the counts. Swamped with data of insufficient quality.
    - The supply of precomputed suggestions gets exhausted
    - Highest traffic surface: the read-mode article

Aug 11, 2026 Dismissing Suggestions

ACTIONS

PURPOSE

1. Align on the issues/risks we can expect without a way to "dismiss" suggestions
2. Discuss approaches for how we might develop a framework for invalidating suggestions that is capable of scaling to accommodate various use cases
3. Decide if/when (roughly) we think we need to prioritize work on this effort

LINKS

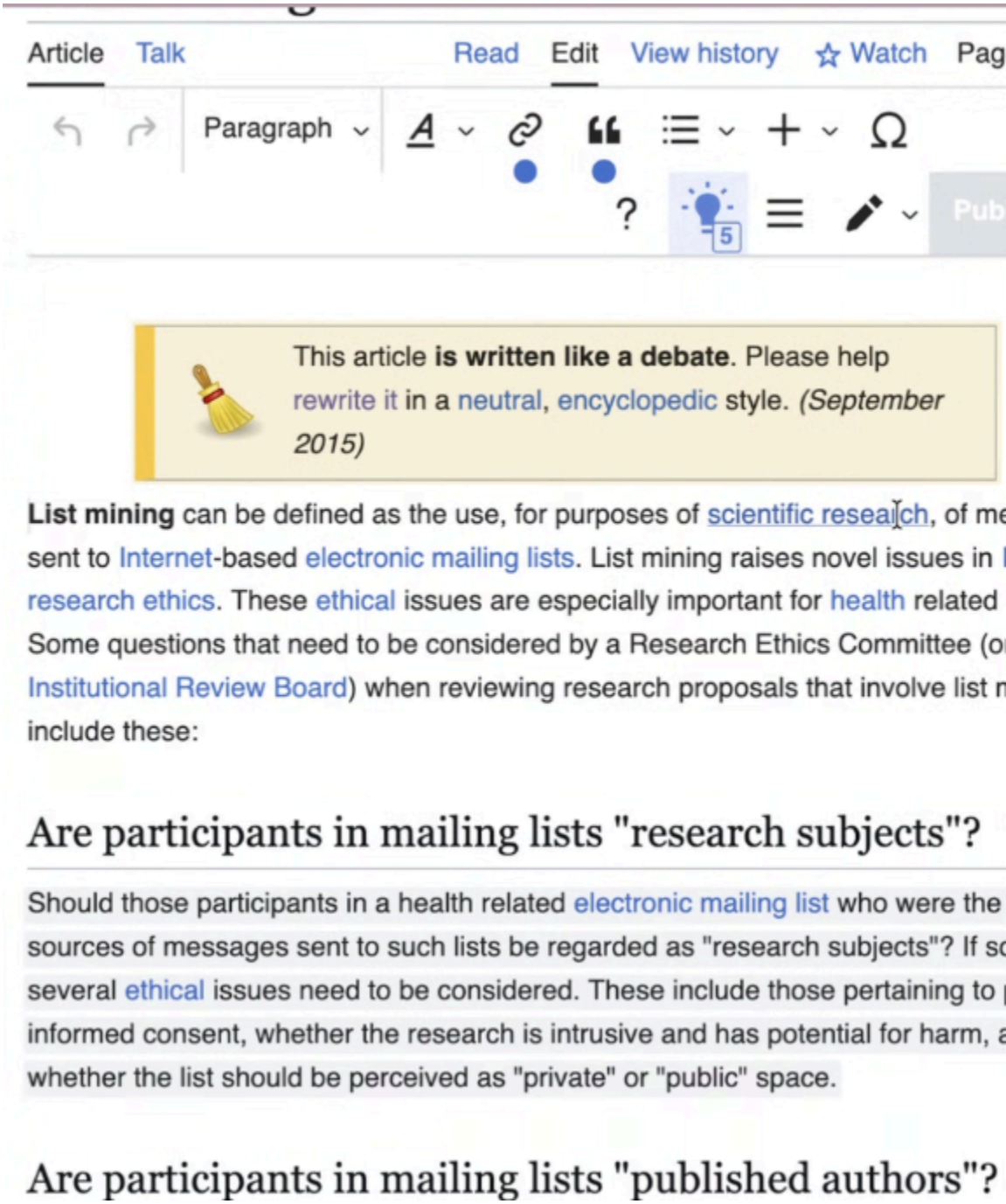
1. T429426: [Epic] Develop a framework for invalidating edit suggestions
2. T433147: [FEATURE] Hide suggestions once defined criteria are met
3. T427236: [FEATURE] Enable people to prevent dismissed suggestions from showing in their future edit sessions
4. Dismissal_of_edit_suggestions

NOTES

- This not a decision-making meeting - we know we have ideas/opinions about this and we want to share understanding for phase 2

- We're in a moment of starting to scale, and we want to be intentional about shortcomings of this feature so we don't commit to scaling something unsustainable → we know dismissal needs to be addressed, BUT we're not letting it block our initial hypotheses
    - Counterpoint: this *should* block our initial release, if we release it as-is we expect pushback from community
    - Defer this decision today
- What happens when a user dismisses a suggestion? 3 streams of work:
    - Overarching epic T429426
        - Two key use cases
            - T433147 - "remove from pool if criteria is met"
            - T427236 - "turn them off for me"
                - E.g. don't show me tone checks ever again / don't show me X checks on this article any more / don't show this-specific-check again
                - This doesn't mean the article doesn't have a suggestion, it just isn't showing up for that user - similar to how you can unsubscribe from a marketing email
- Reviewing https://www.mediawiki.org/wiki/User:DChan_(WMF)/Dismissal_of_edit_suggestions
    - DavidC looking into data complexities that would go into tracking dismissals
    - One big issue - if you dismiss a suggestion on some version of an article, and someone is editing a later version of that article → what does "the same" mean?
    - What does it mean for 2 suggestions to be the same?
    - We probably mean something like: the matched text is still the same, and the context around it hasn't changed much. What happens if you've changed the paragraph a little bit? Some changes would invalidate the suggestion; some changes would not
    - An efficient way to sort out what has/hasn't been dismissed is technically nuanced and a hard problem in terms of data → how can we cut corners so we can make this manageable?
    - What makes it hard? We can't draw a neat line around what's invalidated?
        - "Similar" in text is very hard to code
    - Governance side → there's no "right" answer for when something should be dismissed. We will *always* run into scenarios where people disagree. So what options do we have? Bc dismissals are handled on the backend, editors won't necessarily be able to see and/or curate them. If someone gets annoyed and dismisses a bunch of suggestions, those are "bad faith" dismissals and we want to show them → do we override the editor? It puts a lot of governance in our hands that we don't want. Newcomer tasks e.g. 'add a link' is an example → en.wiki community blocked it for a while, now they're ok with rolling it out. We're gonna see more of these situations. Worry that we'll run into a situation where people turn it off because it's complex to deal with and it won't turn it on again until you "fix it."
    - How can we scale without sidestepping completely?

- Maintenance templates → editors have curated, added. It stays in the article until someone removes it. You're giving them full control again. [Isaac shows a rough prototype]



- We do already support suppressing suggestions per-type based on templates / categories
- There's also T425097: Make a suggestion to highlight maintenance templates which lacks the LLM-usage of the demo, but is a community approach to displaying this kind of guidance

    - How would the maintenance templates help us deal with how to handle dismissals?
        - You already have a template → can set a parameter where editors say “don’t show x suggestion when y template is present”
        - This would avoid similarity model over time and allow more fine grained tuning
    - Suggestions currently designed are bite-sized and easy to tackle, what we’re seeing above is article-level issues. An experienced editor is more likely to use an article-wide template rather than pointing out tiny suggestions. Is there a way this model could support bite-sized suggestions?
        - In-line maintenance tags are bite-sized, so it’s possible we can expand on work we’re already doing
            - But only the biggest wikis have/use the inline cleanup templates. E.g. Enwiki has ~220, but Plwiki only has 19.
        - Ideally we don’t end up in a place where experienced editors are marking up the article to work around the limitations of our dismissals
- Requirements (TODO Isaac do pass)
    - Volunteers need a way to view / audit suggestion dismissals so that they can decide whether a given bit of feedback is relevant/useful…
    - We need a way to define “sameness” of suggestions, so a new suggestion can be suppressed if it is “the same” as a suggestion that has already been dismissed.
- <Discussion in chat around MCR and role it could play in being a better space for handling these Suggestions. Also ability to see dismissed suggestions>
    - IJ/AO: general feeling MCR would work well for a long-term solution with comparing actions between revisions
    - AO: I think MCR is one of those things that has a lot of potential but product use cases keep either turning away or being turned away, so it is never built out further.
    - DL: yeah, my understanding is that the specific usage of MCR (probably the computed revisions that let you attach data after the fact) that we'd want to use got *merged* but is basically untested, such that it'd be a bit of a task to actually rely on it for this.
    - DL: It's something we could absolutely pursue instead of the current LAC approach.
- Samuel: If the revision history (diff) page could show the suggestions a user would have seen and the changes they made, then it would be auditable the same as article text
- -
- PP: we’ve heard volunteers say we need a way to dismiss suggestions /and/ we haven’t heard of any specific suggestions that volunteers are angry with
    - Anecdotal counter example: Suggested link
    - DL: Technically our link suggestion is built on the exact same thing they're annoyed about with the newcomer add-link task. I'm just not sure if they've drawn a connection between them or not.

    - DL: (I could see the presentation of it as one-amongst-many in a full edit making it genuinely different in-practice, in terms of community reaction.)
- PP: how much risk can we take on? Perhaps we have a premortem; we can start to decide how long we can defer this issue
    - One potential solution: (less than ideal) could scope suggestions in the short term; say “for xyz type of article, don’t show abc suggestion”
- Samuel: community has come very close to disabling feature overall. Samuel/Kirsten convinced them to hold off a bit and tweaked config settings instead but if ML can't resolve overlinking tendency, likely to be turned off. Seeing a lot of discussion around Newcomer Task but not so much as its form as a Suggested Edit. Maybe that means they're different because different editor pool? But maybe that just means it hasn't scaled enough?
- Peter: decision around Suggestion Mode as default. Feels like still a ways from "not viable". Also okay if editors decide it's not fit to be scaled. Different projects have different appetites at different times too so doesn't mean we should stop development.

determine the presence/absence of peacock language

- [hard] Within < 500ms of someone finishing *adding* a **new sentence/paragraph**[1] to a Wikipedia article (NS:0) that we are ≥90% sure contains peacock language present a Peacock Check card
    - Where "someone finishing…" will be determined by a set of heuristics detectable through the UX. *E.g. When the focus leaves the paragraph, or based on no activity for a certain time*
    - *See "Awareness" for card details…*
- [later] When someone *modifies* **an existing paragraph or sentence**…

Awareness + Understanding: people are aware of what they have done/are doing that needs their attention and why.

- The *moment* when the interface makes people aware that what they've written warrants additional attention will be the same on mobile and desktop; however, how the Peacock Check card is presented will vary by platform:
    - [hard] On **mobile**, a visual hint will appear in the rail that exists alongside the editable surface. When tapped, the Peacock Check card will be expanded
    - [hard] On **desktop**, the Peacock Check card will appear in the sidebar that exists alongside the editable surface. This card will contain:
        - [hard] Information that the software detected peacock/non-neutral language within the text you added
        - [hard] Information about why the presence of this language/tone is worth you allocating attention to. *E.g. edits without non-neutral tone are ##% more likely to be approved (copy TBD).*
        - [hard] Choices/calls to action to address the issue. *See "Action" below.*
    - [hard] On both platforms, the top of the Peacock Check card will appear at the same "latitude" as the first sentence/paragraph that the system detected peacock language within.
        - [hard] Further, the sentence(s) that the system has detected peacock language within ought to be highlighted.

- [hard] **TECH REQUIREMENT**: Test diffs, i.e. the input is **<original text, modified text>**. Flag only if the content has *become* peacocky.
    - **Rationale:** We don't want to flag a paragraph that is already peacocky in the original text. The current editor is *not* required to fix a peacock violation when modifying the paragraph in some unrelated way.
    - **Assumption:** There is suitable training data for this, since we can gather <original text, modified text> for every diff, and also whether it subsequently was or was not reverted for NPOV violation.
- **TECH REQUIREMENT**: Test paragraphs in isolation, not the whole article.

[1] Where "finish" here is defined as someone beginning a new sentence or beginning a new paragraph, i.e. the focus leaving the added content.

- **Rationale:** Testing paragraphs in isolation drastically simplifies the Edit Check data model and interaction process. The editor can see warnings about individual paragraphs and dismiss them independently. Whereas if our warning applies to the whole article diff, then dismissal can only really be for the whole article.
- **Rationale:** Sending the whole article to be checked repeatedly could cause bandwidth issues.
- **Assumption:** We can train a model to analyse single paragraph changes, without wider article context.
- **Assumption:** There is suitable training data for this, since heuristically a high proportion of diffs are single paragraphs, and each such diff either was or was not reverted for NPOV violation.
- **Assumption:** Peacockiness can be detected on single paragraph changes. Having the entire article as context wouldn't greatly improve precision and recall.

- Otherwise it is difficult during an edit session to show the user what particular content is likely to be suspect.

Action: people are equipped with the tool(s)/affordance(s) to neutralize the tone of what they've written or explicitly decide not to and express why

- Within the mobile and desktop Peacock Check cards, people will be met with the following choices:
    - [hard] A way to **fix/address** the tone issue the software has detected by:
        - [idea] Seeing examples of similar tone issues and how they were successfully improved and an easy way to begin editing the text the model detected peacock language within
    - [hard] A way to **decline** addressing the tone issue the software has detected by electing any of the following:
        - DECLINE REASON #1
        - DECLINE REASON #2
        - DECLINE REASON #3
        - DECLINE REASON #4
    - [hard] A way to **collapse/defocus** the Peacock Check card and bypass taking action on it during the "Mid-Edit" moment.
    - [hard] A way to **expand/refocus** a Peacock Check card you had previously elected to collapse/defocus in the "Mid-Edit" moment
    - [hard] Any Peacock Check(s) that someone has not engaged with[2] during the Mid-Edit moment should be re-presented during the Pre-Save moment with the same calls to action defined above
    - [hard] When someone revises a span of text that had caused the Peacock Check to activate, the model ought to re-check the text to evaluate the extent to which peacock/non-neutral language is still present.

[2] Where "engage" here refers to both attempting to address the peacock tone issue or indicating that you do not think what you've written warrants revision/reconsideration

[hard] **Observability + Improvability**: enable volunteers and staff to see the edits the model detected peacock language within and provide feedback to improve the accuracy of the model.

More in T384651.

[hard] **Impact**: enable the Editing Team to evaluate what – if any – impact presenting Peacock Check(s) have on the likelihood that a new content is: A) reverted and B) written in a non-neutral tone.

More in T383279

Jul 8, 2026

- [x] ~~Lauren de Lench : set up discussion time with SRE re: requirements + risks https://phabricator.wikimedia.org/T428491#12098835~~
  - [x] ~~(done by Kavitha!)~~
- [x] ~~Ed Sanders : to document alternatives pursued and for now, discounted and add to Prep Pantry: Server Side Edit Suggestions~~
- [x] ~~Andrew Otto ALT #1: File ticket: [SPIKE] Investigate surfacing service-based suggestions without VE: T431615 [Spike]: Edit Suggestions - How might we serve computed edit suggestions without VE?~~
- [x] ~~Ed Sanders ALT #2: File ticket: [SPIKE] Identify candidate suggestions that can be extracted from VE | T431602~~

Jul 1, 2026

- [x] ~~Andrew Otto : think about/offer proposal for local development workflow for interacting/testing with LAC~~
  - [x] ~~T431343 Linked Artifact Cache - Local MediaWiki development~~
- [x] ~~Andrew Otto : investigate how to incorporate config more. Possibly store config per wiki in LAC?~~
  - [x] ~~Needs more thought, but not in LAC. Hopefully using an existent MW cache mechanism. TBD how.~~
- [x] ~~Andrew Otto : list the things that can cause suggestion counts to be invalid. Done here.~~

DE1.1.2 Status Reports

Aug 10, 2026 - Aug 14, 2026 Weekly Update

Aug 10, 2026 - Aug 14, 2026 Weekly Update

DE1.1.2 Ready server-side suggestions PoC for use in production:
If we create an Edit Suggestion dataset, we will be able to present counts of the suggestions available within any given Wikipedia article(s) on any on-wiki surface.

- **Status:** At Risk
- **Progress update on the hypothesis for the week**
  - No updates.
- **Any new metrics related to the hypothesis**
  - No.
- **Any emerging blockers or risks**
  - Still blocked on Headless VisualEditor deployment in DE1.1.4.
- **Any unresolved dependencies** - *do you depend on another team that hasn't already given you what you need? Are you on the hook to give another team something you aren't able to give right now?*
  - No.
- **Have there been any new lessons from the hypothesis?**
  - No.
- **Have there been any changes to the hypothesis scope or timeline?**
  - No.

Next steps:
- Wait for Editing and ServiceOps to work out DE1.1.4.

Aug 3, 2026 - Aug 7, 2026 Weekly Update

DE1.1.2 Ready server-side suggestions PoC for use in production:
If we create an Edit Suggestion dataset, we will be able to present counts of the suggestions available within any given Wikipedia article(s) on any on-wiki surface.

- **Status:** At Risk
- **Progress update on the hypothesis for the week**

    - No updates.
- **Any new metrics related to the hypothesis**
    - No.
- **Any emerging blockers or risks**
    - Still blocked on Headless VisualEditor deployment in DE1.1.4.
- **Any unresolved dependencies** - *do you depend on another team that hasn't already given you what you need? Are you on the hook to give another team something you aren't able to give right now?*
    - No.
- **Have there been any new lessons from the hypothesis?**
    - No.
- **Have there been any changes to the hypothesis scope or timeline?**
    - No.

Next steps:

- Wait for Editing and ServiceOps to work out DE1.1.4.

Jul 27, 2026 - Jul 31, 2026 Weekly Update

DE1.1.2 Ready server-side suggestions PoC for use in production:
If we create an Edit Suggestion dataset, we will be able to present counts of the suggestions available within any given Wikipedia article(s) on any on-wiki surface.

This is the first status report for this hypothesis. I will summarize where this work stands today, and then provide a weekly update following the template.

The Editing team has been engaging with the new Prep Pantry process being developed under ST6.2. DPE, Data Persistence, and Editing have reached consensus on a recommended architecture. This recommended architecture consists of 3 main components that need to be deployed:

Headless VisualEditor - DE1.1.4 - this will compute the edit suggestions for a page revision.

Linked Artifact Cache storage and service - this will serve suggestion counts data for use in product features.

Precompute Job - this will trigger edit suggestions to be computed after every edit.

- **Status:** At Risk
- **Progress update on the hypothesis for the week**
  - EditSuggestionCount specific Precompute Job code has been merged. When the other components are deployed, we can turn this on via configuration.
  - Linked Artifact Cache work to store EditSuggestionCounts has begun.
- **Any new metrics related to the hypothesis**
  - No.
- **Any emerging blockers or risks**
  - Headless VisualEditor deployment is at risk.
    - Headless VisualEditor deployment is questioned by SRE ServiceOps. The Editing and ServiceOps teams need to work more proactively directly together to make progress on the proposed technical pilot.
    - My view: neither team currently feels like they know or own the process to push this forward. Hopefully this will be resolved soon.
    - This is tracked explicitly by DE1.1.4, which is a dependency of this DE1.1.2 hypothesis.
- **Any unresolved dependencies** - *do you depend on another team that hasn't already given you what you need? Are you on the hook to give another team something you aren't able to give right now?*
  - The Editing and ServiceOps teams need to work more proactively directly together to make progress on the proposed technical pilot.
- **Have there been any new lessons from the hypothesis?**
  - There are now three overlapping (and competing?) P&T design processes.
    - https://wikitech.wikimedia.org/wiki/Prep_Pantry (ST6.2 from FY2026-2027)
    - https://wikitech.wikimedia.org/wiki/SRE/Data_Persistence/Design_Review (WE6.2.3 from FY2025-2026)
    - https://wikitech.wikimedia.org/wiki/SCROLL (WE6.2.1 from FY2025-2026).
  - How and when teams should engage with each of these is not clear. Also: it is a lot of (well intentioned and valuable!) red tape.
- **Have there been any changes to the hypothesis scope or timeline?**
  - No.

Next steps:

- Get Editing team and ServiceOps to focus together on deploying Headless VE (or not!) for T432286 [MILESTONE] Run technical pilot to evaluate Headless VE resource requirements

Status Report TEMPLATE

Jul 27, 2026 - Jul 31, 2026 Weekly Update

DE1.1.2 Ready server-side suggestions PoC for use in production:
If we create an Edit Suggestion dataset, we will be able to present counts of the suggestions available within any given Wikipedia article(s) on any on-wiki surface.

- **Status:** On Track
- **Progress update on the hypothesis for the week**
  -
- **Any new metrics related to the hypothesis**
  - No.
- **Any emerging blockers or risks**
  - No.
- **Any unresolved dependencies** - *do you depend on another team that hasn't already given you what you need? Are you on the hook to give another team something you aren't able to give right now?*
  -
- **Have there been any new lessons from the hypothesis?**
  -
- **Have there been any changes to the hypothesis scope or timeline?**
  - No.

Next steps:

-

Requirements/**Phase 3**