From Tiki 27 to Tiki 30: what's changing, and why you should start preparing your migration now
From Tiki 27 to Tiki 30: what's changing, and why you should start preparing your migration now

 Bernard Sfez - 2026-06-29 11:58

With the arrival of Tiki 30, the new Long Term Support release expected in summer 2026 and supported until 2031, your Tiki 27 enters the final stretch of its lifecycle. While it will keep receiving security fixes until 2029, there will be no more functional fixes or improvements. The time has come to plan your migration, without rushing, but without putting it off indefinitely.

Between the two releases, two versions (28 and 29) have deeply modernised Tiki. The list is very long, too long for a single, simple article, so we offer here a concise summary, along with all the links to read the details published by the Tiki community. Important good news: there is no abrupt technological break comparable to the PHP 7 to PHP 8 jump, because that work was already absorbed by Tiki 27. The developers, mindful of user's feedback after earlier, more "muscular" upgrades, have taken particular care to build bridges and update scripts that make it far more painless. In short, the road is clear.

Why this article now ?

Tiki follows a steady release cycle: a major version every eight months, and an LTS every three versions, supported for five years. That support breaks down into two phases: a full-support period (bug fixes and small improvements), followed by a longer period of security fixes only. Tiki 27 is entering precisely this second phase as Tiki 30 stabilises, which is why moving matters now. This version will remain under security watch until 2029, but its functional-support window is closing, with no more bug fixes or improvements beyond mid-2026. Tiki 28 and 29 are short-term-support (STS) versions that served as a laboratory for modernisation. And Tiki 30 LTS, expected in summer 2026, will become the next reference destination for production… once the first maintenance releases have confirmed its stability.
The question then, is not whether you should migrate, but when and by which path. That is the whole point of this article: to summarise what has changed, feature by feature, from Tiki 27 to Tiki 30, and then to lay out the migration strategy we recommend.

The strategy in one sentence

Don't sit on Tiki 27 waiting for Tiki 30: make a smoother transition by going through Tiki 29 first. Tiki 27 loses its functional support in mid-2026, and Tiki 30 isn't yet ripe for production. Tiki 29 is the most recent stable version: it lays the technical groundwork (the same modernised building blocks that Tiki 30 inherits), and it carries you comfortably until Tiki 30 LTS is ready. Above all, it gives you time to fix your themes, fine-tune your settings, harden security, and inform and train your teams — to run the dress rehearsal before opening night. This is the gradual, bump-free path we take for our clients, and it ultimately saves both time and money.

Overview: the 27 → 30 trajectory

Tiki 27 was an exceptional LTS: the developers packed into it changes usually reserved for intermediate versions — Smarty 4 → 5, a new npm-based build system, PHPUnit 10, and above all the move to PHP 8.1/8.2/8.3 (extended to 8.4 in the maintenance releases), described by the team as one of the hardest PHP version jumps in the project's more than twenty years of existence.
Tiki 28 and 29 are STS versions (Standard Term Support) that carried the modernisation further:

- migration of the database engine to InnoDB for the unified index.
- Replacement of aging libraries.
- Strengthened security.
- Numerous usability improvements.

Tiki 30 will become the next LTS release and, unlike Tiki 27, it returns to the traditional LTS philosophy: a release focused primarily on refinement, with few disruptive changes. The next major architectural and functional changes are expected in Tiki 31.

The changes, feature by feature

Wiki pages and editing tools

This is one of the areas where the changes are most tangible for users day to day, and it's worth clearly distinguishing the two online editors Tiki offers: the Markdown / TOAST UI editor on one side, and the WYSIWYG editor on the other.

On the Markdown side, the big work happened in Tiki 27, with the move to a Markdown-mode text editor (Tiki-compatible or pure Markdown) and to the TOAST UI editor, making the experience more pleasant without breaking compatibility with traditional Tiki syntax. Tiki 28 carried on by fixing ergonomic flaws and improving PDF generation and preview.

But the most structural change comes with Tiki 29, which replaces CKEditor 4 with Summernote as the WYSIWYG editor. The reason is as much technical as strategic, CKEditor 4 no longer being open source. Summernote is a Bootstrap-based editor, themable, with a customisable toolbar and a working CodeMirror integration. Pasting has also been improved. Formatted content (bold, colours, styles) copied from a word processor or a web page now keeps its formatting when converted into a Tiki page, which cuts down on manual cleanup. This is a major point of attention for your migration, because the editing engine changes under the hood.

On the accessibility side, the page structure has been reworked for WAI/508 compliance. As with every version, the developers pay particular attention to this and aim for the best possible scores on W3C validation tests, whether for code structure or visual rendering. This matters if your site has to meet legal accessibility requirements, which are increasingly common in the public sector and large organisations.

Files (File Gallery)

To set the new features in context, recall that Tiki handles files at two levels. There are the classic file galleries, that is the site's document manager, with versioning, drafts, WebDAV access, and list or thumbnail views. And there is the "Files" tracker field, which lets you attach documents directly to a record. It's on this second point that recent developments are the most telling. The standout new feature comes from Tiki 29. The "Files" tracker field can now automatically create a dedicated file gallery for each tracker item. In practice, each record (user profile, article, invoice, contract, signature) gets its own gallery, instead of having all documents pile up in a shared repository. For anyone using trackers as a business database (managing client files, contracts, requests), this is a real organisational gain. Documents stay tied to their context, and the tree remains readable even after thousands of items. The upload interface for this field has also been significantly enriched.

Tiki 29 has also woven a direct link between the webmail and files. When you send an email from Cypht, Tiki can suggest the relevant tracker item and file the sent email there. The mail system thus automatically feeds the document folder of the record concerned, which supports traceability for client support and projects. Tiki 30 extends this logic on the data-entry side. A new preference (email_to_tracker_mode) lets you choose whether emails are moved or copied when used to create tracker items from an IMAP mailbox, a detail that matters in test or pre-production environments, where you often want to keep the original messages.For the rest, the file manager continues to rest on the familiar building blocks. Standard interface or elFinder interface (drag-and-drop from your workstation, right-click context actions), batch upload, and storage of images in galleries that effectively serve as image galleries.

Trackers (the built-in database and forms

Trackers are the application core of Tiki: its database builder and its low-code / no-code framework, the free-software equivalent of a Microsoft Access or a FileMaker. They evolve with every version, and the 28 → 30 series brings both functional new features and a migration point not to be overlooked.

Tiki 28 first worked on usability and mobility. The Sortable.js library now allows drag-and-drop of fields on touch devices, something that until then made configuring a tracker painful on a tablet or smartphone. Above all, offline trackers become usable within the PWA (Progressive Web App) feature: once the PWA is enabled and the Dexie package installed, a selected tracker can be filled in without a connection, the data being synchronised afterwards — useful for field data entry (inventory, surveys, interventions). This feature, long experimental, was picked up and made reliable between Tiki 27 and 28.

But Tiki 28 also carries the main migration point of attention of this whole series: the switch of field keys from fieldID to permName (permanent name), via the unified_trackerfield_keys mechanism. In practice, any configuration that relied on the field's numeric identifier (fieldID) is migrated to the permanent name. For the vast majority of sites, the update scripts handle the

conversion automatically. But if you have custom code, bespoke plugins, exports or integrations that reference fields by their fieldID, those references need to be reviewed: this is exactly the kind of underlying change that warrants migrating in stages and testing on a pre-production environment rather than switching straight to production.Tiki 29 is the richest version on the feature front. It adds a new barcode / QR code field type, valuable as soon as logistics, inventory or asset tracking are involved. It also introduces online audio recording directly within a tracker item, with waveform preview and built-in playback — handy for voice notes, reports or field documentation. On the files side, the "Files" field gains automatic creation of a dedicated gallery per item (see the Files section), and a smart bridge links the Cypht webmail to trackers: when sending an email, Tiki suggests the relevant tracker item and can file the message there, which also lays the groundwork for automatic routing of replies via an "email folder" field and Sieve rules. For anyone running trackers as a management tool (client support, case files, projects), this is a real gain in traceability.

Tiki 30, as a good refinement-focused LTS, consolidates rather than upends. On email-based entry, a new preference (email_to_tracker_mode) lets you choose whether messages are moved or copied when creating items from an IMAP mailbox — a convenience in test environments where you want to keep the originals. The administration interface for tracker-change notifications (tiki-admin_notifications.php) has also been clarified, making it easier to find settings such as copying the activity of an entire tracker to an email. Finally, the new background task queue (the queue:process command) indirectly benefits heavy trackers, by offloading long operations such as certain document generations.

Performance and error feedback are not left behind either, with continuous improvement from version to version and the addition of a parameter that lets plugins fetching tracker items read only the fields that actually need to be retrieved, which markedly lightens lists drawn from large trackers.

In short: the functional value is mostly in Tiki 29 (new fields, audio, webmail link), but the administrative work to anticipate is in Tiki 28 (fieldID → permName migration).

Search system (Unified Index — MySQL, Elasticsearch and Manticore)

This is one of the areas where the underlying work is most significant across this series of versions, and it's worth setting the scene. For years, Tiki has relied on a unified index: a central index that powers the site's search, but also PluginList, PluginCustomSearch and most of the features that list or filter content. This index can draw on several engines of your choosing — the native full-text search of MariaDB/MySQL (the default, built in, ideal up to medium-sized projects), or more powerful external engines such as Manticore Search (fast, light on RAM) or Elasticsearch/OpenSearch (for very large volumes). As a rule, you can switch engines and rebuild the index without touching the rest of the configuration.

Tiki 28 brings structural changes such as switching the unified index from MyISAM FULLTEXT to InnoDB FULLTEXT. Beyond InnoDB's robustness, this change has a very concrete consequence for heavy tracker users: it removes the historical limit on the number of indexable fields.

On search comfort, Tiki 28 introduces two highly visible features for the end user. First, "Did You Mean?", which suggests a spelling correction when no exact match is found. Second, fuzzy search, which directly tolerates typos and slight variations. These functions work fully with the Manticore engine, and Tiki 28 has also made them available for MySQL-based searches, so they're

accessible even to the simplest installations.

Tiki 29 continues on the index-administration side. The index:cleanup command, introduced to prevent the indexing file system from ballooning, becomes safer: it detects whether an index is shared between several Tiki instances before any deletion, to prevent accidental losses. A new preference (unified_check_unused_indexes) makes it possible to spot unused indexes, and additional options (--all, --dry-run, targeted deletion) give finer control. Elasticsearch management has also been reworked for greater reliability.

A migration point of attention. Search is precisely the kind of subsystem you need to test after an upgrade. The move to the InnoDB index (Tiki 28) means rebuilding the unified index after migration, and that's the moment to check the chosen engine and the rebuild time (rule of thumb: if the rebuild takes less than thirty minutes, you generally have nothing to worry about). A reminder inherited from Tiki 27: the old legacy Tiki search engine, deprecated since 2018, has been completely removed. If an older configuration still relied on it, the unified index is now the only path.

Calendar

The calendar is one of the areas where Tiki has invested the most across this series of versions, and you need to understand the architecture to gauge the scope of the changes. Since Tiki 21, the calendar has relied on a built-in CalDAV server (via SabreDAV), which enables synchronisation with external applications (desktop clients, mobiles, other sites) and the handling of email invitations in connection with the Cypht webmail.

Tiki 27 delivered the deepest overhaul. Operations on events (creation, modification, deletion) now always go through the internal CalDAV server, which ensures that scheduling, the generation of iTIP invitation messages and the associated plugins react correctly to every change. In practice, this unlocks several important features: near-complete support for the recurrence rules of the RFC 5545 standard (beyond the simple "every X days/weeks/months", it handles cases such as "the last Friday of each month"); the definition of availability blocks per user, which feed the availability reports and the "check availability" button; appointment slots, where an availability block becomes a bookable time range — a visitor, or even an anonymous user if the permission allows, picks a slot and creates the appointment, the whole thing embeddable in an external site via iframe; and finally personal private calendars, visible only to their creator and to calendar administrators, with a dedicated permission. This is, in practice, an appointment-booking building block comparable to specialised services, integrated natively.Tiki 28 added an expected touch of usability: the ability to enter a text note when replying to an invitation, and to see those notes on incoming confirmations — behaviour familiar to users of services like Roundcube.

Tiki 29 made a technical change under the hood: the FullCalendar library was replaced by Event-Calendar (@event-calendar), both in the main calendar and in the Tracker Calendar plugin. The reason is, once again, a licence change in the original library; the essential scheduling features are preserved and maintainability improves. On the functional side, Tiki 29 introduces private invitations where the participant list stays hidden: only the organiser sees the full list, guests receive their invitation without seeing the other attendees, while keeping RSVP and calendar integration — useful for large invitations where you don't want to expose the addresses.

Tiki 30 refines the link between email and calendar via Cypht. When an invitation received by email is confirmed and added to the calendar, the original email is now archived rather than

deleted, which keeps the invitation and its attachments for later reference alongside the event. And when creating an event from an email invitation, Tiki now favours a clean HTML version of the message body (sanitised for security), which yields more faithful event descriptions and avoids the technical artefacts of calendar attachments. A point of attention for administrators: the old MiniCal (the legacy personal mini-calendar) is deprecated and slated to disappear; Tiki 29 has in fact introduced a calendar:minical:migrate command to move that data over to the modern private-calendar infrastructure.

Articles

Tiki's CMS building block, distinct from wiki pages — dated, published content, organised by topics and subject to a validation workflow (submissions) — has not seen a major overhaul across the 27 → 30 series. It's a mature module, and most of the changes touch it indirectly, through cross-cutting improvements it benefits from: the new editor (Markdown/TOAST UI and WYSIWYG Summernote), the reworked accessibility, search, and notifications.

The most concrete change goes back to Tiki 27, with the introduction of the tiki_p_export_pdf permission, which allows fine control over who can generate an article's PDF. In practice, it's mainly used to remove that capability from anonymous visitors: PDF generation is resource-intensive, and indexing bots are all too happy to abuse it, to the point of weighing on server load. Being able to restrict PDF export to registered users is a small setting that avoids big headaches. On Tiki 30, the new background task queue (queue:process) also benefits this kind of operation, by offloading PDF generation out of web requests.

For the rest, articles benefit from the general care given to the consistency of the administration interface across versions, without any upheaval in how they work.

Native articles or "home-grown" articles via trackers: an architectural choice

Here we need to open a parenthesis that goes beyond the simple version log, because it shapes the way we read the evolution of articles.
In Tiki, the native Articles module (with its topics, its submissions, its publication date) is not the only way to manage editorial content. Many somewhat advanced sites actually manage their "articles" via trackers: a tracker plays the role of an article table (title, standfirst, body, date, author, category, featured image…), and the display is built with the plugins of the List family. The mechanism is simple to describe. Entry is done in the trackers, the articles are displayed via PluginList or PluginTrackerList (with sorting, filters, pagination, display template), and the move from the list to the detailed record happens via the URL parameter that passes the itemId to a detail page. The result is a fully bespoke publishing system: free field structure, status-based validation workflow, fine-grained per-group rights, and presentation controlled down to the pixel — where the Articles module imposes its own data model.Why this matters in the context of this migration: it's precisely the tracker-based approach that captures the full benefit of the 27 → 30 changes. The native Articles module evolves little across this series (see above), but if you manage your editorial content with trackers, you directly inherit everything described in the Trackers section and the Search section: InnoDB index with no field limit, new field types, "Did You Mean?" and fuzzy search on your article lists, per-item file galleries for attachments, and so on. In other words, the question "have articles improved?" has two opposite answers depending on your architecture — modest for the native module, substantial for the tracker approach.

This also opens up an option: if a site is still on the native Articles module and feels cramped

there (rigid workflow, fixed fields), a version migration is the right moment to reassess a move toward tracker-based management, which is where Tiki invests the most today.

Blogs

Blogs are one of Tiki's most stable modules, and the 27 → 30 series brings them no standout, blog-specific new feature. That doesn't mean they're standing still: like articles, they automatically inherit the cross-cutting work that matters day to day — the modernisation of the editors (pasting formatted content from a word processor works better with Summernote), the accessibility overhaul (heading structure, ARIA), the improvements to search and suggestions, and the notification-on-your-own-comments preference introduced in Tiki 28.
In other words, if your site relies on blogs, the value of a migration isn't read in a list of "blog" features but in the overall quality of use that goes up a notch with each version.

Forums

The change to note on the forums side is in Tiki 29, and it requires concrete action from the administrator: the retrieval of messages posted by email (forum "mail-in") now goes through IMAP with TLS, replacing the legacy POP3 protocol — support for which has been removed. On the substance, this is a good change: IMAP with TLS offers encrypted access and stays compatible with modern email providers, many of which have begun closing off unencrypted POP3 access. But there's an operational consequence to anticipate: IMAP does not delete messages from the mailbox by default, unlike POP3 behaviour where the message was typically removed after retrieval. So if you use forum mail-in, you'll need to review the retention settings of the dedicated mailbox, otherwise it will fill up indefinitely and the same messages risk being reprocessed. This is exactly the kind of detail that takes five minutes to handle when you anticipate it, and becomes an incident when you discover it in production.

For the rest, forums benefit from the cross-cutting improvements (editor, search, accessibility, notifications) without any rework of their own logic.

Mail and webmail

Email has been one of Tiki's strategic priorities for several versions now, under the banner of "email as a first-class citizen": the idea that managing emails should be as integrated into the rest of Tiki as wiki pages, files or trackers are. In concrete terms, this runs through Cypht, the open-source webmail client embedded in Tiki (an aggregator able to bring several mailboxes together), enriched within Tiki with functions it doesn't have on its own, such as calendar integration (CalDAV) and the link with trackers. Across the 27 → 30 series, the work splits between updating the send-and-read plumbing and integrating email into workflows.

On the plumbing side, two changes that matter to administrators. First, Tiki 27 moved Cypht from the 1.4 branch to the 2.0 branch, with its set of improvements, and the following versions continued the upgrade (Cypht 2.4.x in Tiki 29). Second, and this is the point to know for custom configurations: Tiki 29 replaced the email-sending library. The old laminas-mail component (inherited from the Zend Framework era) was abandoned by its maintainer, so Tiki replaced it with symfony-mailer.

In the same move, Tiki 29 modernised the parsing of incoming emails (a version bump of the MIME parser for better PHP 8.2+ compatibility and more reliable handling of encoded headers and

attachments).On the workflow side is where the "email first-class citizen" vision becomes tangible. Tiki 29 introduces a smart integration between the Cypht webmail and trackers: when sending an email, Tiki automatically suggests the relevant tracker items, and the user can choose to file the sent email in the folder of the item concerned. The email thus stops living in an isolated mailbox and joins the folder of the project, client or support case it relates to — a real gain in traceability. This building block also lays the foundations for automatic routing of replies, via an "email folder" field and Sieve rules, and ties in with the email search already fed into the unified index. Tiki 30 extends the logic on the calendar side: an invitation received by email and confirmed has its original message archived rather than deleted (the invitation and its attachments remain available alongside the event), and descriptions of events created from an email now favour a clean, sanitised HTML version.

If your use of Tiki revolves around project collaboration or client relations, this is probably the area where the value of a migration to Tiki 29/30 is the most concrete, because it turns the webmail from a simple email reader into a part of the collaborative setup.

Plugins, new options and technology changes

This section brings together what concerns the cross-cutting technical building blocks: the new features on the plugin side (the wiki-syntax extensions that add functions inside pages), the new configuration options, and the technology changes under the hood. This is a Tiki strong point worth highlighting: since almost all functionality lives in the core of the software, you don't have to wait for an ecosystem of third-party plugins to update before migrating — a decisive advantage compared with CMSes where a version upgrade is regularly held up by an unmaintained extension.

New plugins and options

Tiki 29 introduces PluginFancyLink, which generates a rich preview of a URL directly in the page (title, description and image of the linked content extracted automatically), activatable via a dedicated preference — handy for monitoring or curation pages. Tiki 30 continues in this vein with PluginModel3DViewer (displaying 3D objects in a page), modern-CSS image reflections via the PluginImg options, and above all an awaited enhancement to PluginList, which now handles advanced filtering with AND / OR / NOT logic on multiple selections — a real gain for sites that rely on tracker lists (and therefore, as we've seen, for managing articles via trackers). Going back to Tiki 27, there's also the arrival of PluginList Sublist (nested sublists) and facet output for PluginList and PluginCustomSearch, which beefed up the in-page search capabilities.

Technology changes under the hood

This is where the modernisation is concentrated, largely begun by Tiki 27 (Smarty 5, the new npm-based build system, PHPUnit 10). Tiki 28 carried on by migrating the JavaScript and CSS libraries to this new build system, and by adding a mechanism to validate the version of Composer used with Tiki Manager, to avoid incidents tied to an incompatible version pre-installed on the server. Tiki 29 replaced a series of ageing dependencies: a bump of Swiper (carousels) from version 3 to 11 via npm, the replacement of PhantomJS/CasperJS with Chrome-php for headless browsing (used for exporting wiki diagrams, search snapshots and ChartJS exports), and

a switch of the incoming-email parser to a PHP 8.2+ compatible version. Tiki 30 continued refreshing the building blocks (Converse.js 11 → 12 for instant messaging, DOMPurify 2 → 3 for HTML sanitisation, Monolog 3 for logging) and anticipates compatibility with PHP 8.4 and 8.5.

Point of Attention: Element Plus

Since Tiki 27/28, Tiki has increasingly relied on the Element Plus UI component library for its modern selectors and input widgets. The result is a more attractive and more functional user experience, but it also has an important implication for anyone who customizes their site: these components are more difficult to target with CSS or JavaScript than the older form elements. If your site depends on extensive theme customizations or scripts that manipulate form fields, this is something you should test carefully during migration. This topic is also currently under active review by the developers as part of the "Review all selectors and pickers" project.

Background Task Queue

In Tiki 30, a generic task queue system (queue:process command) now makes it possible to offload long-running operations—such as PDF generation and instance creation—from web requests, while providing complete task lifecycle tracking (pending, running, completed, failed) through both API and command-line interfaces. Although it operates behind the scenes, this is a significant architectural improvement that enhances the perceived responsiveness of large sites and lays the groundwork for more advanced asynchronous processing capabilities in future releases.

Security and Authentication

This is one of the areas that has seen the most significant and consistent improvements throughout the 27 → 30 release series, making it a compelling reason to upgrade. Staying on a version that receives only security fixes means missing out on these ongoing security enhancements. Tiki has long placed a strong emphasis on security and its highly granular permissions system, which extends down to the level of individual features and even individual objects throughout the site.

L'authentification forte progresse à chaque version

Tiki 28 introduced two-factor authentication (2FA) via email, complementing the existing app-based 2FA (such as Google Authenticator). This provides administrators with an alternative that does not require users to install a third-party authentication app.

Tiki 29 marks a major milestone with support for WebAuthn and passkeys, enabling passwordless authentication using a hardware security key, fingerprint, or the device's built-in facial recognition. This is now the industry standard for strong authentication, offering robust protection against phishing attacks. Its introduction brings Tiki in line with modern expectations for secure user authentication.

Tiki 30 takes 2FA management to the next level for organizations. While previous versions

introduced the feature itself, Tiki 30 makes it practical to deploy and manage at scale. Administrators can now reset or disable a user's 2FA configuration—an essential capability for account recovery when someone loses their phone or security key. Mandatory 2FA can also be enforced with configurable grace periods by group or individual user (for example, 3, 7, 14, or 30 days), with reminder messages displayed during the grace period and optional notifications sent to users who have not yet enabled 2FA. A dedicated administration interface allows these grace periods to be granted, revoked, or extended on a per-user basis. For organizations rolling out 2FA to a large user base without disrupting everyone overnight, these management tools fill a critical gap.

Tiki 30 also adds the ability to lock or unlock multiple user accounts in bulk from the user administration interface, along with an experimental mechanism to help protect against brute-force attacks.

Server and Browser Security Hardening

Tiki 30 introduces a comprehensive set of configurable HTTP security headers and CORS support directly from the administration interface, making it possible to implement security best practices—such as Content Security Policy (CSP), clickjacking protection, and allowed-origin controls—without manually editing the web server configuration.

Tiki 29, meanwhile, overhauled the cookie consent system to align with GDPR requirements. Users can now accept, reject, or customize their cookie preferences through a dedicated dialog box, with their choices stored and synchronized with their account. As noted earlier for Webmail, this redesign also renamed many configuration preferences, so sites with customized configurations should test this area carefully during migration.

Auditability and responsible disclosure. Tiki 29 also enhanced logging through Monolog, including improved tracking of failed login attempts, and introduced the security:generate command, which creates a standard security.txt file in the .well-known/ directory. This provides the standardized way to publish a security contact and a vulnerability disclosure policy.

Tiki 28 also improved transparency around the Switch User feature used by administrators. Users can now be notified whenever an administrator assumes control of their account, with this behavior configurable according to the organization's security policy. In short, if there is one compelling reason not to remain on a version that has reached the end of feature support, it is this: Tiki's security has been significantly strengthened between versions 27 and 30, and these improvements are not backported to older releases.

System Requirements Changes

This is the key factor that determines the technical feasibility of any migration, and the good news is that there are no major new system requirement changes between Tiki 27 and Tiki 30.

On the PHP side, the entire 27 → 30 release series supports the same range: PHP 8.1 through 8.4, with Tiki 30 also preparing compatibility for PHP 8.5. The difficult transition—from PHP 7.4 to PHP 8.1, described by the Tiki team as the most challenging upgrade in the project's twenty-year history—was fully absorbed in Tiki 27. In other words, if your server is already running PHP 8.x, the hardest part is behind you and the upgrade path is straightforward. The only significant

obstacle concerns sites still running PHP 7.4: in that case, upgrading the server's PHP version must be your first step. (For reference, Tiki 24 LTS was the last release compatible with PHP 7.4.)

Database requirements also remain unchanged throughout the series: MariaDB 10.5 or later or MySQL 8 or later. The main item to include in your migration plan is not a database version change, but a storage engine change. Tiki 28 migrated the search index to InnoDB, which requires rebuilding the search index after the upgrade (see the Search section).

Finally, regarding development tools, installations from the source code repository (Git) have relied on the npm-based build system since Tiki 27, and Tiki 28 added validation of the Composer version in use. This does not affect installations performed from the standard release packages. However, if you deploy from Git, be sure to include the build step in your migration plan. As a best practice before any upgrade, run Tiki's Server Check script, which quickly verifies that your PHP version, required extensions, and database configuration meet the application's requirements.

What to Do Next

Here is the migration roadmap we recommend, in order.

Start by verifying your server environment. Confirm that you are running PHP 8.1 or later (ideally PHP 8.2 or 8.3) and MariaDB 10.5+ or MySQL 8+. These prerequisites determine everything that follows.

Next, migrate to Tiki 29, beginning with a test environment. It is the latest stable release, includes the modernized components that Tiki 30 builds upon, and moves your site out of the support transition zone. Tiki's migration policy is also clear: when leaving an LTS release, you should always target the latest stable version available, rather than another older release.

During this migration, pay particular attention to the compatibility points discussed throughout this article:

- Update your custom themes to match the version of Bootstrap used by Tiki 27 and later.
- Convert Tracker field references from fieldID to permName if you have custom code or third-party integrations.
- Rebuild the search index after the migration to the InnoDB search engine.
- Thoroughly test the WYSIWYG editor if your site uses it, as CKEditor 4 has been replaced by Summernote.
- Migrate forum mail-in processing from POP3 to IMAP, reviewing your mailbox retention settings.
- Review any customized configuration preferences, particularly those affected by the migration to Symfony Mailer and the redesigned cookie consent system, as many preference names have changed.

Completing these steps will prepare your installation for a smooth transition to Tiki 30 LTS.

Tiki 30 is expected during the summer of 2026. As with every major release, we recommend allowing the initial maintenance releases to mature before upgrading, and targeting Tiki 30.1 or 30.2 once the first round of stability fixes has been delivered. Because Tiki 30 is a refinement-

focused LTS rather than a disruptive release, this final upgrade should be considerably smoother than the major transition introduced with Tiki 27.

At that point, you'll be ready to upgrade your Tiki installation. Depending on your deployment model, several options are available: automated upgrades provided by your hosting provider for smaller sites, upgrading from the official release package, or deploying from Git, which is our preferred approach for larger, professionally managed projects.

If you're unsure where to begin, you can find a Tiki consultant through the Consultants directory on tiki.org, or simply contact our Tiki support team for assistance.

Further Reading: Official Sources

Notes about the Tiki version (communauty documentation)
Tiki 27 - Tiki 28 - Tiki 29 - Tiki 30

Related dev pages (technical and commits)
dev Tiki 27 - dev Tiki 28 - dev Tiki 29 - dev Tiki 30

GitLab repository
Tiki sur GitLab - Commits 27.x - Commits 28.x - Commits 29.x - Commits 30.x

Life Cycle
Tiki Lifecycle