Integration Options

Replyke offers two primary ways to integrate into your application, depending on your tech stack. Whether you're using React, React Native, or another framework, you can leverage Replyke's functionality in the way that best suits your needs.

⚠ To use Replyke in production from a web application, make sure to whitelist your domain in the dashboard of your project.

Using the React or React Native Client Packages (Recommended)

If your application is built with **React or React Native**, the easiest and most efficient way to integrate Replyke is through the provided client packages. These packages handle much of the complexity for you, including:

- **Authentication Handling** – Handles authentication by managing tokens and automatically refreshing them when necessary.
- **State Management** – Seamless updates and reactivity with minimal effort.
- **Optimized API Calls** – Abstracts away API interactions for a smoother developer experience.
- **UI Components (Optional)** – Includes ready-to-use UI components that can be customized.

For most developers working with React or React Native, using the client packages is the best approach as it drastically simplifies the integration process and reduces the amount of work required.

Using the API Directly

If your application is built with a framework other than React or React Native (such as Vue, Angular, Svelte, or a backend-only implementation), you can integrate Replyke via the **API**.

This approach requires handling authentication, state management, and API calls manually, but it allows full flexibility to integrate Replyke into any tech stack.

- **Client-Side API Usage** – For frameworks other than React/React Native, implement Replyke by making direct API calls from your client.
- **Server-Side API Usage** – If you need Replyke functionality from the backend (e.g., syncing comments, moderating content, handling reports), your server should also communicate with Replyke via the API.
- **Authentication Handling** – Developers need to manage authentication manually, including storing tokens securely and refreshing them when necessary.

Which Option Should You Choose?

Scenario	Recommended Approach
Using React or React Native	Use the client packages (Recommended)
Using Vue, Angular, Svelte, or other frameworks	Use the API
Server-to-server communication	Use the API

More framework integrations are currently under development.

For details on each approach, check out the **Client SDK Guide** for React/React Native or the **API Documentation** for direct API usage.

Last updated on May 6, 2025