Steps to Add PWA Functionality in laravel website

To add PWA functionality to a Laravel website, the most straightforward approach is using a Composer package like `silviolleite/laravel-pwa` or `erag/laravel-pwa`. This automates the creation of the necessary **manifest file** and **service worker** script.

Prerequisites

- A running Laravel application (version 8.x or higher is typically supported by recent packages).
- Your application must be served over **HTTPS** (required for service workers to function correctly), even in local development using tools like Laravel Valet or Homestead.
- Composer installed.

Steps to Implement PWA Functionality

1. **Install the PWA Package**
   Install the desired package via Composer in your project's root directory. The following steps use `silviolleite/laravel-pwa` as an example:
   bash

```
composer require silviolleite/laravelpwa
```

Alternatively, you can use the `erag/laravel-pwa` package:

bash

- `composer require erag/laravel-pwa`

- **Publish Configuration Files and Assets**
  After installation, publish the vendor assets to make the configuration files and default service worker/manifest files available in your project's `config/` and `public/` directories:
  bash

```
php artisan vendor:publish --provider="LaravelPWA\\Providers\\LaravelPWAServiceProvider"
```

For the `erag/laravel-pwa` package, you would run:

bash

- ```
  php artisan erag:install-pwa
  ```

- **Configure the Manifest**
  A new configuration file (`config/laravelpwa.php` or `config/pwa.php`) will be created. Customize this file to define your app's metadata:
  - `name` and `short_name`: The name displayed on the user's home screen.
  - `start_url`: The URL the PWA should load when launched (usually `/`).
  - `display`: Set to `standalone` to give it a native app feel without a browser address bar. Other options include `fullscreen`, `minimal-ui`, or `browser`.
  - `theme_color` and `background_color`: Define the colors for the OS UI elements and the splash screen.
  - `icons`: Specify the paths and sizes for various application icons and splash screens. You need to place your own images in the `public/images/icons` directory, ensuring sizes like 192x192px and 512x512px are available.
- **Include Blade Directive in Layout**
  To link the manifest file and register the service worker in the browser, add the provided Blade directive within the `<head>` section of your main layout file (e.g., `resources/views/layouts/app.blade.php`):
  html
- ```
  <!DOCTYPE html>
  <html lang="en">
  <head>
      @laravelPWA <!-- For silviolleite package -->
      {{-- or use @PwaHead for erag package --}}
      ...
  </head>
  <body>
      ...
      {{-- For erag package, add this before the closing </body> tag --}}
      {{-- @RegisterServiceWorkerScript --}}
  </body>
  </html>
  ```

- **Customize the Service Worker (Optional but Recommended)**
  The published assets include a default service worker file (usually `public/serviceworker.js` or `public/sw.js`). This file controls caching strategies for offline functionality. You can customize it to pre-cache specific assets (CSS, JS, images, an offline page) or set up runtime caching for dynamic content:
  - Define an offline route in `routes/web.php` that serves a custom offline view in case of a network failure.
- **Test Your PWA**

- Deploy your application to a hosting environment that uses HTTPS.
- Open your website in a Chromium-based browser (like Chrome or Edge).
- Open **Chrome DevTools** (right-click and select "Inspect") and navigate to the **Lighthouse** tab. Generate a report for "Progressive Web App" to check if all criteria are met.
- You should see an "Install App" or "Add to Home screen" prompt in the browser's address bar or menu, confirming installability

---

Revision #1
Created 13 December 2025 16:24:58 by AI Channel
Updated 13 December 2025 16:25:34 by AI Channel