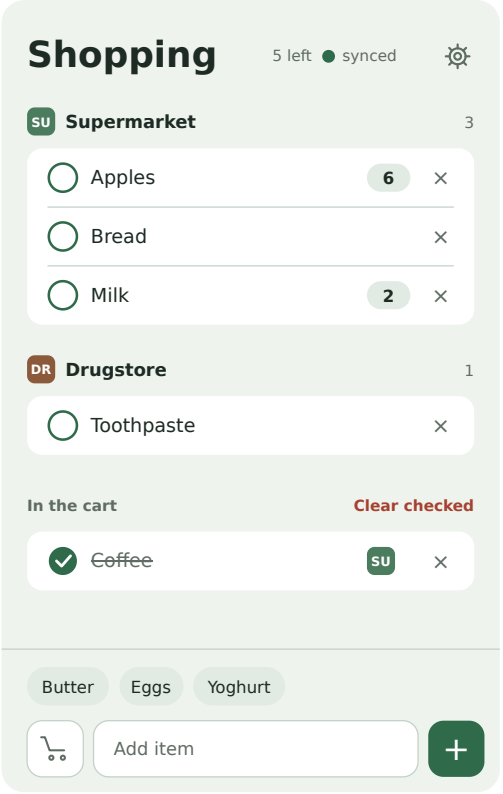
WEB APP · VERSION 1.6.1

Shopping List

Installation and user manual for the lean, privacy-friendly shopping list with shops, quantities and sync through your own web space.



it-und-mehr · Paul Drieduite · it-und-mehr.at

October 2026 · Licence: MIT

Overview

The shopping list is a single HTML file that you host on your own web space. On your phone you add it to the home screen, after which it behaves like a regular app. A small PHP script keeps your phone and computer in sync.

Contents

1. What the app does
2. Requirements
3. Installing on your web space
4. Setting it up on your phone
5. More devices and updates
6. Everyday use
7. Shops and icons
8. Settings
9. Restoring a backup
10. How the sync works
11. Troubleshooting
12. Technical details and licence

1. What the app does

- **Items and quantities** in one field, for example `3 apples` or `2x milk`.
- **Suggestions from your history**, most frequent first – most items go on the list with a single tap.
- **Shops with your own icons**; the list is grouped by shop automatically.
- **Sync** between your devices through your own server.
- **Works offline**: the app starts without a network, changes are sent later.
- **Automatic backups** of the last 50 versions, each one restorable.
- **Screen stays on** while the app is open (can be switched off).
- **Six languages** (English, German, Dutch, French, Italian, Spanish), light and dark mode.

> **Privacy**
> The app loads no external fonts, scripts or trackers, and it needs no account. Without sync your list stays on your device only; with sync it also lives on your own server – nowhere else.

2. Requirements

Web space	HTTPS and PHP 7.4 or newer, Apache (for the included `.htaccess`). No database required.
Access	FTP or SFTP access, for example with FileZilla.
Devices	Current browsers: Safari, Firefox, Chrome, Edge. “Keep screen on” on iPhone requires iOS 18.4.
Package	`index.html`, `api.php`, `sw.js`, `config.php`, `data/.htaccess`, `README.md`, `LICENSE`.

The app also works without sync, but then the list stays on a single device. In that case uploading `index.html` is enough.

3. Installing on your web space

1. **Unpack the package**
   Unzip the download. You get a folder called `shopping-list`.

2. **Generate a key**
   This key protects your list from outside access. Generate it in a terminal:

   ```
   openssl rand -hex 24
   ```

   Keep the output – you will need it on every device later.

3. **Enter the key**
   Open `config.php` in an editor and replace the placeholder. The single quotes and the semicolon must stay:

   ```
   <?php
   // Generate a key with: openssl rand -hex 24
   const TOKEN = 'your-48-character-key';
   ```

4. **Upload**
   Upload the `shopping-list` folder with all its files to your web space, for example to `https://example.com/shopping-list/`. The `data` subfolder and the `.htaccess` inside it must be uploaded too. In FileZilla, show hidden files via *Server → Force showing hidden files*.

5. **Set write permissions**
   PHP must be allowed to write into the `data` folder. Usually `755` is enough. If PHP runs as a different user than your FTP account, you need `777`. The folder still stays blocked from the web, which is what the `.htaccess` is for.

6 **Check the protection**

Both commands must return exactly the codes shown:

```
curl -s -o /dev/null -w "%{http_code}\n" https://example.com/shopping-list/api.php   #
401
curl -s -o /dev/null -w "%{http_code}\n" https://example.com/shopping-list/data/     #
403
```

`401` means the interface demands the key. `403` means the data folder is blocked from outside.

7 **Test write access**

This command creates an empty state and answers the question whether PHP may really write. Expect `{"rev":1}` and `200`:

```
K=your-key
curl -s -w "\n%{http_code}\n" -H "X-Token: $K" -H "Content-Type: application/json" \
     -d '{"data":{"items":[],"hist":{},"shops":[]}}' \
     https://example.com/shopping-list/api.php
```

> **Common pitfall**
>
> If the server reports `Could not acquire lock` or `Write failed`, the write permissions on `data` are missing. If it reports `Wrong key`, the key in `config.php` does not match the one you sent – often the single quotes are missing.

4. Setting it up on your phone

1 **Open the address**

Go to `https://example.com/shopping-list/` in Safari, with the trailing slash.

2 **Add to home screen**

Tap *Share*, then *Add to Home Screen*. From now on, start the app from that icon only.

3 **Enter the key**

Open the app from the icon, tap the gear in the top right and enter the key under *Sync key*. After saving, the dot turns green and reads *synced*.

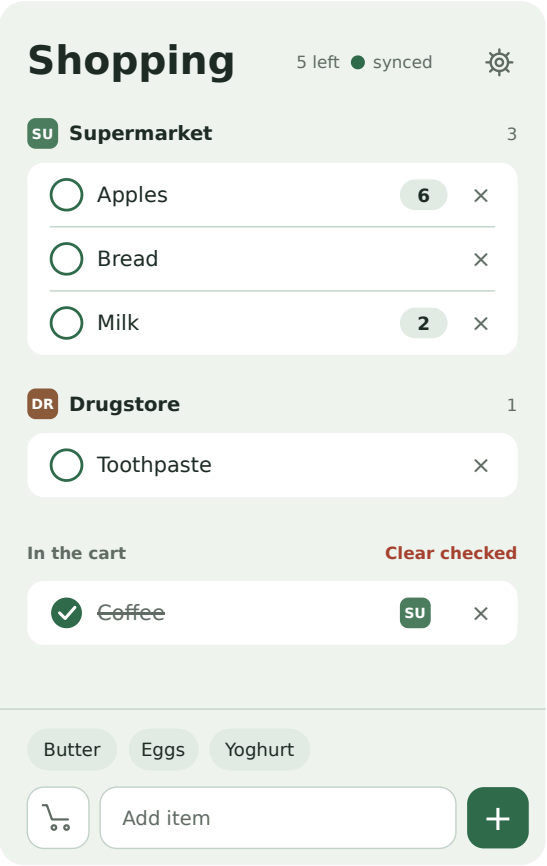
> **Why start from the icon?**
>
> Safari and the home-screen app have separate storage. Only the home-screen app runs full screen and keeps its data permanently; for ordinary websites, Safari clears the storage after a few weeks without use.

5. More devices and updates

On every other device, open the same address and enter the same key. All devices then show the same list. Language and the screen switch are stored per device and are not synced.

To update, replace only `index.html`, `api.php` and `sw.js`. Leave your `config.php` and the `data` folder untouched – otherwise you lose your key and your list. Close the app completely afterwards and open it again. The version number at the bottom of the settings tells you whether the new version is active.

6. Everyday use



The list, grouped by shop. At the bottom, the suggestions and the input row.

Adding items

Type the item at the bottom and confirm with the return key or the plus button. A quantity is simply a number you type along, before or after the name:

Input	Result
`3 apples`	Apples with quantity 3
`apples 3`	the same
`2x milk`	Milk with quantity 2
`500g flour`	stays as plain text, because “500g” is not a plain number

Without a number no quantity is shown, which means one. To change a quantity, simply enter the item again with a new number. An item already on the list is not added twice.

Suggestions

Everything you have entered once appears above the input row as a suggestion. The most frequent ones come first, and typing filters them. A tap adds the suggestion. Press and hold a

suggestion to remove it from your history.

Checking off and clearing

A tap on an item checks it off and moves it down into *In the cart*. Another tap undoes that. The cross deletes a single item, *Clear checked* removes all checked ones at once. Both can be undone within five seconds. Checked items disappear by themselves after twelve hours anyway.

7. Shops and icons

Shops are what gives the list its order: items are grouped by shop automatically, in the order you define. Inside a shop you therefore see at a glance what you need there.

1. **Add a shop**
   Gear → *Edit shops*, type a name, tap *Add*.

2. **Choose an icon**
   Tap the image field in the shop's row and pick an image from your device. The app resizes it itself and syncs it along; you do not need to upload anything by FTP. Square images work best. Without an image, the shop gets a coloured circle with its initials.

3. **Set the order**
   Use the arrows to sort the shops the way you actually shop. The groups appear on the list in exactly that order.

When adding an item, you pick the shop with the icon to the left of the input field. The choice stays active until you change it. Set to *Anywhere*, the app automatically uses the shop where you last entered that item – so most of the time you need not choose at all. To move an item to a different shop, select the new shop and enter the item again.

If you delete a shop, its items stay on the list and move into the *Anywhere* group.

8. Settings

The gear in the top right opens the settings:

Language	Your device language by default, switchable to any of the six languages. Applies to this device only.
Edit shops	Add, rename, sort, delete shops or give them an icon.
Keep screen on	Stops the display from going dark while you shop. Applies only while the app is in the foreground.
Sync key	The key from `config.php`. Above it you see the status: *synced*, *offline*, *local* or an error message.
Restore earlier version	Opens the list of backups held on the server.

At the very bottom you find the version number. After an update it tells you immediately whether the new version is really running.

9. Restoring a backup

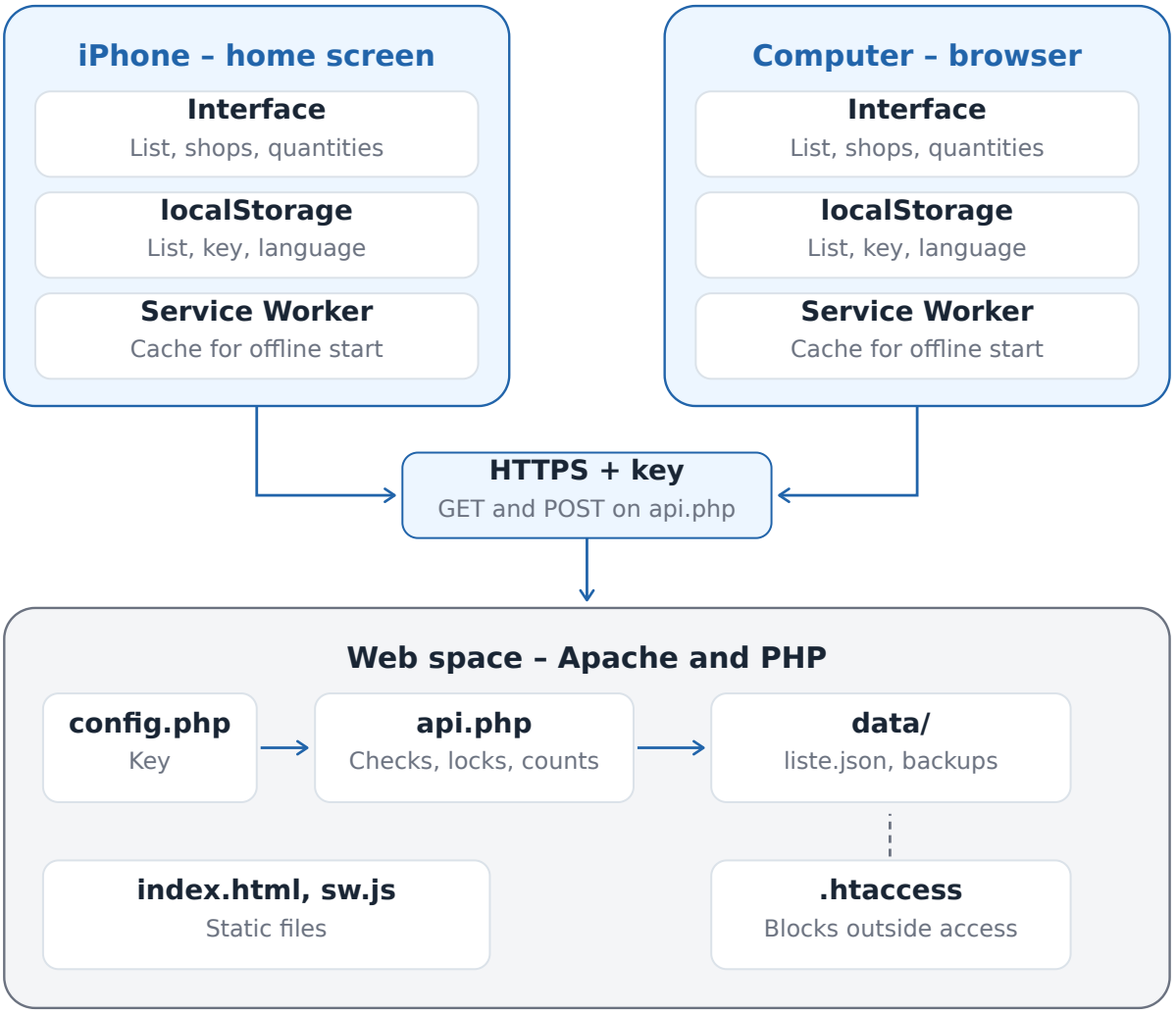
Before every change, the server sets the previous state aside and keeps the last 50 of them. This is your safety net if something was overwritten by accident.

1. **Open the list**
   Gear → *Restore earlier version*. Each entry shows date, time and the number of items.

2. **Pick a version**
   Tap the entry you want and confirm. The current list is backed up first, so nothing is lost for good.

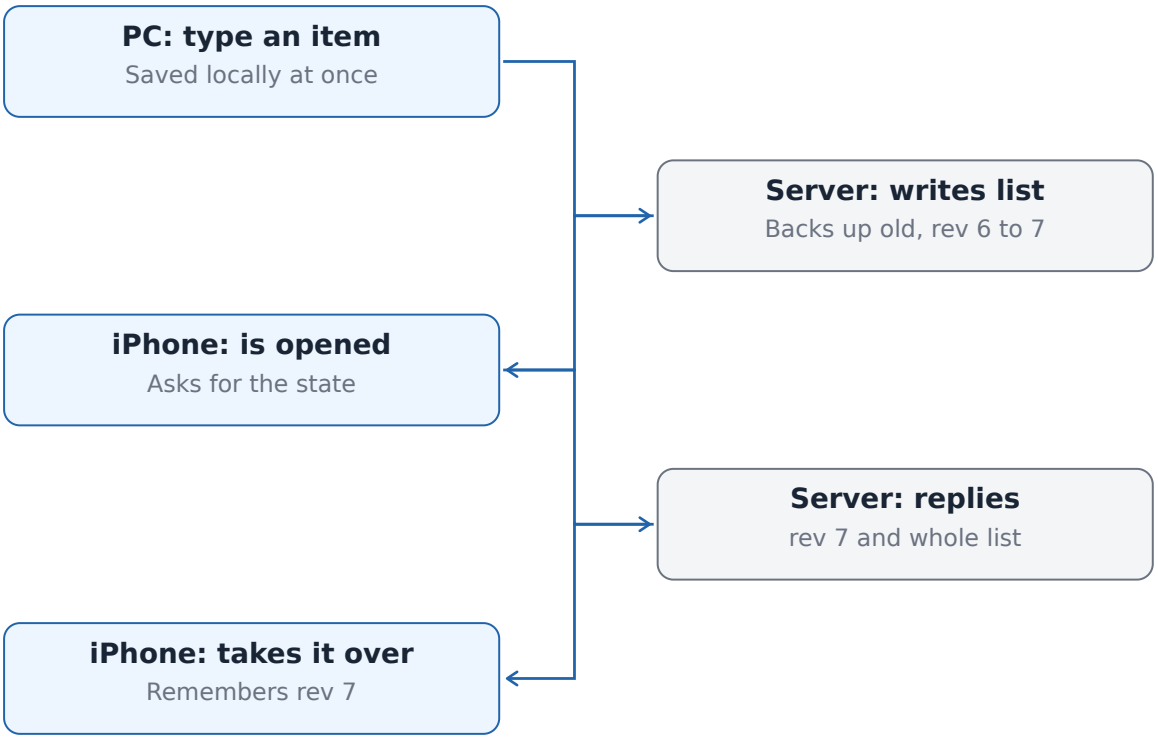
10. How the sync works

This section is here for understanding; you do not need it for everyday use.



Structure: every device holds a full copy of the list, the server is the shared store.

Every device stores the list locally first. That is why the app works without a network. The service worker serves the cached interface, so the app even starts where there is no signal. On the server, `api.php` checks the key against `config.php`, locks the file while writing and increases a revision number.



Sequence: the revision number decides whether a device takes over the server state.

The revision number sets the beat. If the phone still knows 6 and the server reports 7, the phone takes over the server state. If both match, nothing happens and almost no traffic goes over the line. Syncing happens when the app is opened, after every change, every 20 seconds, and as soon as the device comes back online.

> **Limits of the sync**
>
> The change sent last wins. The app is built for one person with several devices, not for several people at the same time. If you edit offline on one device and in parallel on another, the device that syncs last overwrites the other state. That is what the backups in section 9 are for.

11. Troubleshooting

Symptom	Cause and fix
Status reads *key?*	The key entered does not match `config.php`. Copy the key again and check for missing quotes in `config.php`.
Status reads *error*	The key is fine, but the server reports a problem – almost always missing write permissions on `data`. Run the write test from section 3; the message names the cause in plain text.
Status reads *local*	No key has been entered. The list stays on this device. Gear → *Sync key*.
Status stays *offline*	No connection, or the address is not reachable over HTTPS. Check the address in a browser.
Changes do not reach the other device	Both devices must use the same key. Close the app once and reopen it to force a sync.
Old version runs after an update	The app was still in memory. Close it completely (swipe it away in the app switcher) and reopen, then check the version number in the settings.
List is empty although the server has data	The app was probably opened in Safari instead of from the home-screen icon. Start it from the icon and enter the key there.
Overwritten by accident	Gear → *Restore earlier version* and pick the last good state.

12. Technical details and licence

Footprint	One HTML file, one PHP script, one service worker. No dependencies, no database, no build step.
Storage	In the browser via `localStorage`; on the server as `data/liste.json` plus up to 50 backups in `data/backup/`.
Interface	`GET` reads the state, `POST` writes it. The key travels in the `X-Token` header and is checked with `hash_equals`.
Server	PHP 7.4 or newer with Apache. Writes are guarded against overlap with `flock`.
Limits	Designed for one person with several devices. No collaborative editing, no user management.
Licence	MIT – free to use, modify and pass on, without warranty.

Questions, bugs or suggestions are welcome at info@it-und-mehr.at. You will find the current version at it-und-mehr.at/downloads-en.html.