CAP.LY BEGINNER GUIDE (STEP BY STEP)

Version: February 13, 2026

This guide explains, in simple terms, everything we set up for your URL shortener.
It is written for a complete beginner.

1) WHAT YOU BUILT

You built a clean URL shortener app called cap.ly.

Tech stack:
- HTML (page structure)
- CSS (styling)
- JavaScript (logic)
- Node.js (simple local server for development)
- Cloudflare Pages (hosting on the internet)
- GitHub (code backup/version control)
- Namecheap + Cloudflare DNS (custom domain: cap-ly.com)

Important: this version does NOT use a database.
Links are saved in the browser's local storage only.

2) PROJECT FILES (WHAT EACH FILE DOES)

In your project folder:

- public/index.html
  The user interface (form, buttons, result area).

- public/styles.css
  App styling (layout, colors, spacing, typography).

- public/script.js
  URL shortener logic in JavaScript.
  - Validates links
  - Generates custom/random short codes
  - Saves mappings to localStorage
  - Redirects when a short path is opened

- index.js
  Small Node server for local testing.
  Serves files from /public.

- package.json
  Project metadata and scripts.
  Main scripts:
  - npm start
  - npm run deploy:cloudflare

- wrangler.toml
  Cloudflare Pages config.
  Project name: surl
  Output folder: public

- DEPLOY.md
  Quick deployment instructions.

3) HOW THE SHORTENER WORKS (CURRENT VERSION)

When user enters a long URL and clicks "Create Short Link":

Step A:
- JavaScript reads the long URL and optional custom code.

Step B:
- It validates URL format (must be http/https).

Step C:
- If custom code was entered:
  - It is sanitized (letters, numbers, - and _ only).
  - If that code already exists, app creates a random code instead.

Step D:
- App builds short URL using your domain:
  https://cap-ly.com/<code>

Step E:
- App stores mapping in browser localStorage.
  Storage key: caply.links.v1

Step F:
- App shows short link and copy button.

When someone opens a short path (example /abc123):
- script.js reads the path.
- If code exists in localStorage of THAT browser, it redirects.
- If not, it shows "code not found".

Very important:
- localStorage is per browser/device.
- Links are NOT shared globally.
- This is fine for demo/learning, not for production SaaS.

4) GITHUB SETUP (WHAT YOU DID)

You connected the project to:
- https://github.com/captaine13/url-shortner

You also cleaned old git history and previous remote relation,
then pushed the current clean project state.

Current branch:
- master

Current remote:
- origin -> git@github.com:captaine13/url-shortner.git

5) CLOUDFLARE PAGES DEPLOY (WHAT YOU DID)

```
You created Cloudflare Pages project:
- surl

Default Pages URL:
- https://surl-eri.pages.dev

Deploy command used by npm script:
- npx wrangler pages deploy public --project-name surl

So every deploy uploads your /public folder as the live website.


============================================================
6) CUSTOM DOMAIN SETUP (NAMECHEAP + CLOUDFLARE)
============================================================

You bought domain:
- cap-ly.com

Then you:

Step 1:
- Added cap-ly.com to Cloudflare.

Step 2:
- Cloudflare gave 2 nameservers.
  (They look like random words, for example: alice.ns.cloudflare.com)

Step 3:
- In Namecheap, replaced default nameservers with Cloudflare nameservers.

Step 4:
- Waited for propagation (can take minutes to 24h).

Step 5:
- In Cloudflare Pages project (surl), added custom domains:
  - cap-ly.com
  - www.cap-ly.com

Step 6:
- SSL became active.


============================================================
7) WWW TO ROOT REDIRECT RULE (WHY IT EXISTS)
============================================================

Rule name:
- www-to-apex

Behavior:
- If user opens https://www.cap-ly.com/*
- Redirect to https://cap-ly.com/${1}
- Status code 301
- Query string preserved

Meaning:
- One canonical domain (cap-ly.com)
- Better consistency and SEO
- Prevents split traffic between www and non-www

Example:
```

- Input: https://www.cap-ly.com/abc?x=1
- Output: https://cap-ly.com/abc?x=1

8) SSL ERROR YOU SAW AND WHAT IT MEANT

Error seen:
- ERR_SSL_VERSION_OR_CIPHER_MISMATCH

Typical reason in your flow:
- DNS/nameservers were not fully active yet
- or certificate was not issued/propagated yet

After nameservers and Cloudflare domain status turned Active,
SSL worked correctly.

9) WHY EDITS SOMETIMES DID NOT REFLECT IMMEDIATELY

Common reasons:
- Changes were local only (not deployed yet)
- Browser cache showed old files
- Cloudflare edge cache had old response briefly

Fix process:
1. Save code changes.
2. Deploy again.
3. Hard refresh browser (Cmd + Shift + R on macOS).

10) SIMPLE DEPLOY WORKFLOW FOR FUTURE (RECOMMENDED)

Every time you change code:

1. Open terminal in project folder.
2. Run:

```
   git add .
   git commit -m "Describe your change"
   git push origin master
   npm run deploy:cloudflare
```

3. Open:
   https://cap-ly.com

If page looks old:
- Hard refresh with Cmd + Shift + R.

11) OPTIONAL AUTO DEPLOY FROM GITHUB

You can make deployments automatic:

In Cloudflare Dashboard:

- Workers & Pages -> surl -> Settings -> Builds & deployments
- Connect your GitHub repo: captaine13/url-shortner
- Production branch: master
- Build command: (empty)
- Build output directory: public

Then future deploy flow becomes:
- git add .
- git commit -m "..."
- git push origin master

Cloudflare deploys automatically.

==
12) WHERE LINKS ARE STORED (IMPORTANT)
==

Right now:
- Stored in localStorage in each browser.
- Not in a backend database.

So if another person opens your site:
- They will NOT see links created on your browser.

To make it like bit.ly, you need:
- Backend API (Node/Express or Cloudflare Worker)
- Database (D1, Supabase, Postgres, etc.)
- Save codes centrally
- Read codes centrally on redirect

==
13) HOW BIT.LY-LIKE LINKS WORK (HIGH LEVEL)
==

For a real global shortener:

1. User sends long URL to backend.
2. Backend creates unique code (for example 4qxdqTz).
3. Backend stores: code -> long URL in database.
4. Short link shared: https://cap-ly.com/4qxdqTz
5. Any device opening that link hits backend.
6. Backend looks up code in DB and sends redirect.

That is the major difference from localStorage demos.

==
14) QUICK TROUBLESHOOTING CHECKLIST
==

If site does not load:
- Check domain status in Cloudflare (Active)
- Check SSL/TLS status in Cloudflare
- Confirm DNS records are correct

If code changes not visible:
- Confirm deployment succeeded
- Hard refresh browser
- Test in incognito window

If short links fail on other devices:
- Expected in current app (no database)

==

15) WHAT YOU HAVE NOW

==

You now have:
- A working beginner-friendly URL shortener UI
- A custom branded domain: https://cap-ly.com
- Working Cloudflare hosting + SSL
- GitHub-connected codebase for version control
- Clear deploy process for future updates

This is a strong base. Next upgrade is adding a real backend + database.