

# Auto-create demo video



I have a React app.  
How to auto-create demo video? I can mention which buttons to click and what to enter.

Got it 👍 You want to **auto-generate a demo video of your React app**, where instead of manually screen-recording, you just describe interactions (like “click this button” or “type this text”), and it produces a polished video.

There are a few approaches you can take:

## Option 1: Scripted Browser Automation + Screen Recording

1. Use **Playwright** or **Puppeteer** to script interactions with your React app:

- Click buttons, type in fields, navigate, etc.
- Example:

```
js

// demo.js
const { chromium } = require('playwright');

(async () => {
  const browser = await chromium.launch();
  const context = await browser.newContext({
    recordVideo: { dir: 'videos/', size: { width: 1280, height: 720 } }
  });
  const page = await context.newPage();

  await page.goto('http://localhost:3000');

  // Example actions
  await page.click('#login-btn');
  await page.fill('#username', 'demoUser');
  await page.fill('#password', 'secret123');
  await page.click('#submit');

  await page.waitForTimeout(3000); // wait so video captures result

  await browser.close();
})();
```

- This auto-generates a video (`videos/demo.mp4`).

## Option 2: Script to Storyboard → Video

- You define a **script/DSL** like:

```
yaml

steps:
  - goto: "http://localhost:3000"
```

```
- click: "#login-btn"
- type: { selector: "#username", text: "demoUser" }
- type: { selector: "#password", text: "secret123" }
- click: "#submit"
```

- Then a small Node.js tool reads this and runs Playwright/Puppeteer to record.

## Option 3: Use Browser Recording Plugins

- Tools like [Replay.io](#) ↗, [Loom API](#) ↗, or [Screenity](#) ↗ let you **programmatically capture sessions**, but they're more manual. If you want automation, Playwright's built-in video capture is better.

## Option 4: Add Narration/Text Overlays

- After Playwright records the raw video:
  - Use **FFmpeg** to overlay captions like "Clicking Login button".
  - Or automatically add voice narration using **text-to-speech (TTS)** (e.g., Google Cloud TTS or AWS Polly).

🔑 The most automated and polished approach is:

1. Write interactions in a YAML/JSON "script".
2. Run Playwright → record video.
3. (Optional) Post-process with FFmpeg → captions + TTS narration.

👉 Do you want me to draft a **ready-to-run template** where you just list the steps (like "click X, type Y"), and it spits out a video automatically?



Any npm packages to implement Option 2: Script to Storyboard → Video?

Yes — there *are* several npm packages / services that can help with implementing **Option 2: Script → Storyboard → Video**. Depending on how much of the pipeline you want to build vs use a service, you can mix & match. Here are some good ones + ideas how to put them together.

## 🔍 Useful npm packages / services

Package / Service	What it does / Pros	What you'll need to build or integrate to use for your "scripted storyboard → video" flow
@puppeteer/recorder	Records user interactions in Chrome / Chromium (via Recorder) and produces a Puppeteer script. <a href="#">GitHub</a>	You can use this to capture the steps of the flow (click, type etc.) into a script that you can run. But doesn't itself record video output; you'd combine it with a video

Package / Service	What it does / Pros	What you'll need to build or integrate to use for your "scripted storyboard → video" flow
		recorder step (e.g. Puppeteer + a screen recorder) to get visuals.
<b>puppeteer-screen-recorder</b>	Allows recording the browser screen during a Puppeteer session. Can record actions etc. <a href="#">npm</a>	If you replay the script (from recorder or handcrafted), then wrap in <b>puppeteer-screen-recorder</b> (or Puppeteer's built-in video/capture API) to generate the video. Might need to install FFmpeg etc depending on formats.
<b>puppeteer-video-recorder</b>	A simpler video capture plugin for Puppeteer. <a href="#">npm</a>	Same: plays well when you have scripted automation. Might lack advanced editing (overlay, transitions etc.).
<b>@airnub/puppeteer-remote-recorder</b>	Captures screen (video) and audio, works over remote Puppeteer connections. <a href="#">npm</a>	Helpful if your demo setup is remote or distributed; you still need to feed in the script of interactions.
<b>Shotstack (via shotstack-sdk)</b>	A cloud video editing API + SDK. Combine images/video/text/assets via JSON → rendered video with transitions, overlays etc. <a href="#">Shotstack+2</a>	Could use this to <i>post-process</i> or build polished video: e.g. your recorded video clips + overlays/text/titles, or even to stitch together snapshots/screenshots rather than full video, depending on your approach.
<b>ffmpegcreator</b>	Node.js library to build/compose short videos with pictures/videos/audio etc. <a href="#">npm</a>	Good for simple edits, combining clips, adding music, transitions. But less about automating browser interactions; more about processing & composing assets after capture.

## How to build a flow using these

Here's a sketch of how you might combine the tools above into a workable "Script → Storyboard → Video" pipeline:

1. **Define a script / DSL** — e.g., JSON or YAML that describes steps like:

```
yaml
steps:
- goto: "/login"
- click: "#username-field"
- type: "#username-field", "demoUser"
- type: "#password-field", "password123"
- click: "#submit-button"
- wait: 2000
- screenshot: "#dashboard"
```

2. **Generate an automation runner** — Use Puppeteer to read that script and execute it. You can even auto-generate the Puppeteer code from the script, or directly interpret.

3. **Record visuals:**

- Use **puppeteer-screen-recorder** (or similar) to capture video of the browser during the run.

- Optionally, also take screenshots at key steps (for “storyboard style” visuals), or capture browser logs.

#### 4. Post-process / edit:

- If you want overlays (labels like “*User clicks login*”) or highlight effects, you can use FFmpeg (scripted) or libraries like **ffcreator**.
- For more polished editing (transitions, smooth cuts, title screens), you might send the video + assets to a service like **Shotstack** via their SDK.

#### 5. Optional narration / captions:

- Either generate captions based on the script automatically.
- Use a TTS library or service (Google Cloud TTS, AWS Polly etc) to generate voice-over.
- Merge audio + video using FFmpeg or via video editor API.

#### 6. Final rendering & output:

- Export MP4 (or whatever format).
- Possibly compress or process for different resolutions.

---

## Caveats / Things to watch out for

- Synchronisation / timing: script steps vs what shows up on screen (network delays, animations etc). You’ll probably need waits or steps to assert content loaded before moving on.
- Performance & video quality: screen recording can consume CPU and produce large files. Need to pick frame rate, quality etc.
- Audio capture: many browser automation tools do NOT capture audio by default. If you want sound (narration, UI sounds), you’ll need extra setup.
- Cost / limits: services like Shotstack have quotas or costs. If many videos or high resolution, cost adds up.

---

If you want, I can put together a **starter npm package / boilerplate** (or pick a combination of above) so you can drop in your text script + receive the demo video automatically. Do you want me to prepare that?