

Two stacks, one difference

The design of the containers in `compose.harness.yaml`

Two identical stacks

```

shared fixtures (identical for both sides)
course server      identity stub      frozen clock (HARNESS_NOW)
      |              |              |
+-----+-----+-----+-----+   +-----+-----+-----+-----+
| side a: PRODUCTION images          |   | side b: CANDIDATE images          |
| reader catalogue live time         |   | reader catalogue live time         |
| reader-auth (signed in)            |   | reader-auth (signed in)            |
| persistence stub (records all)     |   | persistence stub (records all)     |
+-----+-----+-----+-----+   +-----+-----+-----+-----+
```

The two sides differ in **one thing only: the image reference.**

Everything else is fixed

Anything that differs except the image would show up as the harness's own noise, so:

- **one frozen instant:** `HARNESS_NOW` in every container, and the browser clock frozen to it;
- the same viewport (1280 by 800), locale `en-IE`, time zone `Europe/Dublin`, light theme, reduced motion;
- a pinned fixture course served with no `Date`, `ETag` or `Last-Modified`;
- an identity stub that issues GitHub-shaped sign-ins for fixed users;
- third-party fonts and icons **recorded once** and replayed to both sides.

Run them as production runs them

Every app container carries the same posture:

```
read_only: true           # read-only root filesystem
tmpfs: [/tmp]             # the only writable place
cap_drop: [ALL]           # no Linux capabilities
security_opt: [no-new-privileges:true]
healthcheck: GET /healthz/live
```

JSON logs, anonymous mode, one reader per side with sign-in on. The `runtime` artefact then measures that posture from *inside* each container, so a candidate that runs as root or writes outside `/tmp` is called out.

A persistence stub per side

- Each side has its own Supabase-shaped stub that **records every write** the reader tries.
- Writes are attributable to a side and a journey: `<journey>/<table>`.
- Any write on b during an *anonymous* journey is a failure on its own.
- The same idea covers a message bus (`<journey>/<topic>`) when one is configured.

Where the images come from

- A bare tag expands per app: `--a 17.0.0` means `quay.io/tutors-sdk/tutors-reader:17.0.0`, and the same for catalogue, live and time.
- Pinned by digest where it can be, so what was judged is what deploys.
- **Signature or nothing**: unsigned, or signed by the wrong identity, exits 2 with no verdict.
- SBOM and vulnerability scans are read from the images before any stack starts.

Profiles and substrates

Profile / substrate	Adds
<code>upgrade</code>	an edge proxy in front of both readers: b rolls in under k6 load and no request may fail
<code>a2</code>	a second copy of side a, to measure noise inside one run
<code>--substrate kind</code>	the two sides as namespaces under restricted Pod Security Admission

Same captures, same engines, whichever substrate runs them.

Takeaways

- Two stacks that differ **only** in the image: every other difference is noise by construction.
- Production posture is the default, not a special case.
- Writes are recorded per side, so data changes are visible, not inferred.