perplexity

Quick Troubleshooting Checklist

Component	What to Verify	Fast Commands / Checks
PostgreSQL	- Service is running - Correct host/port in `.env.local` - Credentials actually work	**Linux systemd**: `systemctl status postgresql` or `sudo service postgresql status` **macOS (Homebrew)**: `brew services list` → look for `postgresql` **Container**: `docker ps | grep postgres` **Ping DB**: `pg_isready -h <host> -p <port> -d <db>`
MinIO / S3	- Service/container is up - Access- & secret-key match `.env.local` - Bucket exists	**Local MinIO**: `minio server /data` (foreground) or `docker ps | grep minio` Browse `http://localhost:9001` to confirm buckets and creds
ADCF binary	- Starts without dependency errors - Path to config and `.env.local` correct	Run it manually: `./adcf serve --log-level=debug` (or equivalent). Watch stdout/stderr for lines such as **"cannot connect to PostgreSQL"** or **"S3 client init failed"**.

Step-by-Step Guide

1. **Check PostgreSQL is running**
    1. Run the status command for your OS (table above).
    2. If **inactive**, start it:

```
# Linux
sudo systemctl start postgresql

# macOS (Homebrew)
brew services start postgresql
```

    3. Confirm the ADCF credentials work:

```
psql -h <host> -U <user> -d <db>
```

    A successful psql prompt means the credentials in `.env.local` are valid.

2. **Verify MinIO / S3 (if your deployment uses object storage)**
    1. For a local MinIO container:

```
docker run -p 9000:9000 -p 9001:9001 \
  -e "MINIO_ROOT_USER=$MINIO_ACCESS_KEY" \
  -e "MINIO_ROOT_PASSWORD=$MINIO_SECRET_KEY" \
  -v minio_data:/data minio/minio server /data
```

2. Open the browser UI at `http://localhost:9001`, log in with the keys from `.env.local`, and make sure the bucket defined in your ADCF config exists.

3. **Run ADCF in the foreground to inspect logs**

```
# from the project root
chmod +x ./adcf        # if necessary
./adcf serve --log-level=debug
```

   - Look for the **first ERROR** line; it normally spells out the missing or wrong dependency (e.g., "dial TCP 127.0.0.1:5432: connection refused").
   - Fix the reported issue, then restart.

4. **Still stuck?**
   - Paste the first 20–30 log lines around the error into your next message.
   - Mention whether PostgreSQL and MinIO are on the host or in Docker—networking (ports, `localhost` vs container names) is a common cause.

Need hands-on help?

- "Set up PostgreSQL for me" – I'll walk you through installation and creating the database/user expected by ADCF.
- "Show me the full ADCF error logs" – Copy/paste or attach the log snippet and I'll interpret it.

Let me know which you'd like to do next!

Citations

PostgreSQL systemd service docs – postgresql.org
Homebrew services manual – brew.sh
`pg_isready` utility reference – postgresql.org
MinIO quick-start guide – docs.min.io

What the error means

`brew services start postgresql` can only start a service that Home-brew has **already installed**. Because Homebrew cannot find the `postgresql@14` formula on your machine, it shows:

```
Error: Formula `postgresql@14` is not installed
```

So PostgreSQL 14 was never installed (or was uninstalled), and there is no macOS launch agent for Homebrew to start.

Fix in two steps

Step	Command	What it does
1. Install PostgreSQL 14	`bash brew install postgresql@14 `	
2. Start it at login	`bash brew services start postgresql@14 `	

> If you only need a one-off foreground server, replace step 2 with `pg_ctl -D /usr/local/var/postgresql@14 start`.

Verify the service

```
brew services list           # should show postgresql@14 as started
pg_isready -h localhost      # returns "accepting connections"
psql -U postgres -d postgres # opens a psql prompt
```

Common follow-ups

- **Add psql to your PATH (Intel Macs):**

  ```
  echo 'export PATH="/usr/local/opt/postgresql@14/bin:$PATH"' >> ~/.zshrc
  exec $SHELL
  ```

  (for Apple Silicon use `/opt/homebrew/opt/...`).
- **Initialize a fresh database:** Homebrew usually does this automatically, but if needed: `initdb -D /usr/local/var/postgresql@14`.
- **Different major version?**
  Run `brew search postgresql@` to see 12 – 17; substitute the version you actually want in both commands.

After PostgreSQL is running, re-run the ADCF backend. If ADCF still fails, capture the first 20-30 log lines and share them so we can check the next dependency.

[1] Homebrew formula page for `postgresql@14` (install instructions).
[2] Homebrew `brew services` documentation (starting launch agents).

⁂

1 — Fix the directory typo first

Your backend folder is named `uars7-backed`, not `uars7-backend`.

```
cd ~/uars7/uars7-backed
```

If you need the more readable name, just rename the folder:

```
mv uars7-backed uars7-backend
```

2 — Install PostgreSQL 14 with Homebrew

The error shows the formula is missing, so install it:

```
brew update                      # good habit before any install
brew install postgresql@14       # grabs the server and client tools[^3_1]
```

3 — Start the server as a macOS service

```
brew services start postgresql@14 # launches PostgreSQL at login[^3_2]
```

Verify it's running:

```
brew services list               # should now list postgresql@14 as "started"
```

4 — Put `psql`, `pg_isready`, … on your PATH

Homebrew installs the binaries under its **opt** directory.
Add them once to your shell profile (`~/.zshrc` for zsh):

```
# Intel Macs
echo 'export PATH="/usr/local/opt/postgresql@14/bin:$PATH"' >> ~/.zshrc

# Apple-silicon Macs
echo 'export PATH="/opt/homebrew/opt/postgresql@14/bin:$PATH"' >> ~/.zprofile
```

Reload the profile, then check:

```
exec $SHELL            # or: source ~/.zshrc
psql --version         # should print "psql (PostgreSQL) 14.x"[^3_3][^3_7]
pg_isready -h localhost
```

5 — Initial database & superuser (first-time only)

```
# create the default cluster if Homebrew didn't do it
initdb -D /usr/local/var/postgresql@14

# create the postgres superuser so ADCF can log in
createuser -s postgres
```

6 — Quick sanity checklist before rerunning ADCF

Check	Expected result	Command
Service status	`started`	`brew services list`
Port 5432 open	`accepting connections`	`pg_isready -h localhost`
psql prompt	connects without error	`psql -U postgres -d postgres`
`.env.local`	matches host = localhost, port = 5432, user = postgres	*open file*

Once each row is green, restart the ADCF backend from its project root; it should now connect to PostgreSQL without the earlier "backend did not start" message.

7 — Common follow-ups

- Need a one-off foreground server?
  `pg_ctl -D /usr/local/var/postgresql@14 start`
- Switching versions? Replace `@14` with `@15`, `@16`, etc., in every command.
- Still stuck? Run `./adcf serve --log-level=debug` and share the first error block for the next round of troubleshooting.

Good luck—once Postgres is running, ADCF should spring to life.

⁂

1. https://formulae.brew.sh/formula/postgresql@14
2. https://www.dataquest.io/blog/install-postgresql-14-7-for-macos/