Appendix C — Installation & Running

This appendix explains how to set up the project from a zipped archive, verify the installation, and run both the command-line tools and the Streamlit UI.

C.1. System prerequisites

- **OS:** macOS (Intel/Apple Silicon), Linux. (Windows works via WSL2 or native Python; commands below include Windows notes where relevant.)
- **Python:** 3.11.x (recommended).
  Rationale: several libraries used here publish well-tested wheels for 3.11; newer versions sometimes lag.
- **FFmpeg:** required for audio decoding (used by faster-whisper).
  - macOS: `brew install ffmpeg`
  - Ubuntu/Debian: `sudo apt-get update && sudo apt-get install -y ffmpeg`
- **Build tools (Linux only):** `build-essential` and `python3-dev` are often needed:
  `sudo apt-get install -y build-essential python3-dev`

Note (Apple Silicon): faster-whisper/ctranslate2 provide ARM64 wheels. Make sure Homebrew is ARM64 (`/opt/homebrew`) and Python is not running under Rosetta.

C.2. Unpack the code and create a virtual environment

1 **Unzip the archive** you received (example path shown):

```
unzip s2c_project.zip -d ~/projects
```

```
2  cd ~/projects/s2c
3
```

4 **Create and activate a virtual environment** (Python 3.11):
  - macOS/Linux:

```
python3.11 -m venv .venv
```

-
  ```
  source .venv/bin/activate
  ```
-

- Windows (PowerShell):

```
py -3.11 -m venv .venv
```

-
  ```
  .\.venv\Scripts\Activate.ps1
  ```
-

C.3. Install Python dependencies

The project ships with a `requirements.txt` that covers the core stack (Streamlit app, LibCST, pytest, etc.). Install it first:

```
pip install --upgrade pip
pip install -r requirements.txt
```

Optional: enable on-device ASR (faster-whisper)

If you plan to use live speech decoding instead of the mock backend:

```
pip install faster-whisper
```

This pulls in **ctranslate2** under the hood. If you hit wheel issues on Linux, try:

```
pip install "ctranslate2<4"  # some environments prefer a
pinned major version
```

If installation complains about FFmpeg, confirm it's on your PATH (`ffmpeg -version`).

C.4. Environment configuration (optional but useful)

You can influence ASR behavior with the following environment variables (set them in your shell or a `.env` you load in your shell profile):

```
# Model/device knobs for faster-whisper
export S2C_WHISPER_MODEL=base.en        # e.g., tiny.en, 
base.en, small.en
export S2C_WHISPER_DEVICE=cpu           # or cuda (if you 
have an NVIDIA GPU)
export S2C_WHISPER_COMPUTE_TYPE=int8     # cpu: int8; cuda: 
float16

# N-best synthesis via temperature sweeps (comma-separated 
floats)
export S2C_WHISPER_TEMPS="0.0,0.3,0.6,0.8"

# Bias decoding with a short prompt including domain terms/
hotwords
export S2C_WHISPER_USE_PROMPT=1
```

You can also pass a **VAD** (voice activity detection) window from the UI or CLI; default is 400 ms.

C.5. Verify the installation

Run the quick sanity suite (isolated sandboxed tests):

```
python -m s2c.verify
```

Expected output resembles:

```
{
  "pass_frac": 1.0,
  "passed": 3,
  "failed": 0,
  "total": 3,
  "returncode": 0
}
```

If you see `ModuleNotFoundError: No module named 's2c'`, ensure you are running inside the virtual environment and from the project root (or use `pip install -e .` if a `pyproject.toml/setup.cfg` is provided).

C.6. Running the Streamlit UI

The Streamlit app provides push-to-talk (browser mic), mock/Whisper decoding, step-through processing, and test-gated edits. Open Terminal, change to the s2c directory, and run the following commands:

```
pip install -e .
streamlit run src/s2c/app.py
```

Then open the local URL shown in the terminal (typically http://localhost:8501). In the **Settings** sidebar:

- **Target file:** e.g., `src/s2c/user_solution.py`
- **ASR backend:** choose **mock** (typed text) or **whisper** (speech).
- For Whisper:
  - Toggle **Use decoding prompt** if you want a domain prompt.
  - Adjust **VAD (ms)**, **n-best**, and canonicalization/disfluency settings as desired.

Use the **Command** panel to record audio or type a mock command (e.g., `add parameter min_count default 2 to word_count`).
Use **Run Step** to walk each stage or **Run Edit** to execute the full pipeline. The **Last result** panel shows diagnostics, diffs, and test outcomes.

C.7. Troubleshooting

- **`ffmpeg: command not found`**
  Install FFmpeg and re-open your shell: `brew install ffmpeg` (macOS) or `sudo apt-get install -y ffmpeg` (Linux).
- **Whisper decoding is blank or missing words**
  Increase **VAD (ms)** (e.g., 400→600), enable **Use decoding prompt**, and raise **n-best** to 5. Consider enabling **Canonicalization** and **Strip disfluencies**.
- **Tests fail; UI shows "after" code**
  By design, the file on disk is always restored if tests fail; the UI only previews "after" in memory. To confirm, open the target file in an editor—the on-disk content remains unchanged on failure.
- **`ModuleNotFoundError: s2c` when running CLIs**
  Ensure you are in the project root and the virtual environment is active. If your environment expects an install step, run `pip install -e .` (if a project config is provided), or run from the root with `python -m s2c.<module>`.
- **Port 8501 already in use**
  Run the app on a different port:
  ```
  streamlit run src/s2c/app.py --server.port 8502
  ```