Audion DevOps Tools: Архитектура И Правила Развития

Короткая архитектурная карта проекта для людей и AI-агентов, которые меняют поведение, GUI или service layer.

Пользовательский обзор находится в `USER_GUIDE_RU.md`. Практические runbook-и по опасным подсистемам лежат рядом в `docs\*_RU.md`.

Позиционирование

Audion DevOps Tools - не generic Windows tuner и не debloater. Это project-local cockpit для точных Windows/WSL/virtualization/hardware/storage/network/policy/secrets операций.

Главные свойства:

- явные параметры;
- risk labels;
- backup перед изменением;
- confirmation для dangerous-команд;
- live terminal log;
- отсутствие зависимости от старых внешних папок.

Основные Файлы

```
Start.exe                          starts the window App\AudionDevOpsTools.exe
App\                               the window and AudionPs.exe, the PowerShell of the
program
Source\                            the sources of the window (C#, Avalonia)
devops_cli.cmd                     the console command behind every button
config\tool_manifest.yaml           command tree, fields, risk metadata
config\window_layout.yaml           sections, pages and cards of the window
config\terminal_commands.json       terminal command cache
system_core\cli.py                  the console command
system_core\core\jobs.py            operation runner, terminal decoding
system_core\core\paths.py           first-class project folders, including backup\
system_core\services\devops_tools.py
system_core\cli_operation.py        manifest operation CLI runner
tools\                             project-local utilities
backup\                            snapshots and rollback data
logs\                              operation logs
```

Service Boundary

GUI не должен содержать системную логику. Он собирает параметры, показывает risk/confirmation и вызывает operation через manifest/service layer.

Системные действия живут в:

- `system_core\services\devops_tools.py`;
- project-local modules under `system_core\...`;
- project-owned scripts under `tools\...`;
- manifest operations in `config\tool_manifest.yaml`.

Не подключай старые соседние папки как runtime dependency. Исторические папки можно читать только для сравнения или восстановления по прямой просьбе.

Граница окна

Окно на Avalonia (`Source\`) — лицо консольной команды `devops_cli.cmd`. Где стоит операция, говорит `config\window_layout.yaml`: раздел → страница → карточка (`op`, `switch`, `pick`, `guests`), рамка `block`, тон `tone`; что она делает — `operation_groups` манифеста.

Правила:

- разделы, строки ИСТОЧНИК и НАЗНАЧЕНИЕ и терминал остаются на месте, меняется только страница;
- поля берутся из YAML `fields`;
- `fields.id` уникален по смыслу;
- GUI не переписывает YAML runtime-значениями;
- длинные/опасные описания идут в hint/tooltip, а не в название кнопки;
- видимое описание команды компактное и может быть автоматически сжато до 1-2 строк, но tooltip и dangerous-confirmation сохраняют полный смысл;
- `action_group` показывает логическую рамку вокруг пар/pipeline (`backup / restore`, `export / import`, `block / unblock`).
- `tone: run` — главное действие рамки (залитая кнопка), `tone: burn` — удаляет; это приборная подсказка, не замена `kind`, `risk_level` и подтверждений.

Подробно о раскладке — шапка `config\window_layout.yaml`.

Risk Model

`kind` и `risk_level` в manifest не декоративны.

Типовые уровни:

- `safe` - read-only или слабое действие без системной записи;
- `dangerous` - требует явного понимания и часто admin context;
- `system_change` - меняет Windows policy/service/device/network state;
- `user_write` - меняет профиль/current-user state;
- `destructive` - удаляет, unregister, перезаписывает или чистит state;
- `secret_export` - создаёт artifacts с private keys/password-equivalent data.

Dangerous CLI-запуск должен требовать `--yes-i-understand`.

Документационная Модель

`USER_GUIDE_RU.md` - пользовательский guide; `COMMANDS_RU.md` - каждая операция с её полями.

Runbook-и в `docs` нужны для быстрого поиска и безопасного исполнения:

- `BITRIX_HOSTS_RU.md`;
- `NETWORK_CONNECTIVITY_RU.md`;
- `WSL_TOOLKIT_RU.md`;
- `VIRTUALIZATION_SWITCHER_RU.md`;
- `DEFAULT_APPS_GUARD_RU.md`;
- `ASSOCIATION_DEFENSE_RU.md`;
- `HARDWARE_DRIVER_GUARD_RU.md`;
- `STORAGE_DISK_PROCEDURES_RU.md`;
- `OPENSSH_KEYKIT_RU.md`;
- `CERTIFICATE_KEYKIT_RU.md`;
- `MAINTENANCE_CLEANUP_RU.md`.

Internal/dev docs:

- `docs\SMOKE_TEST_RU.md`;
- `SKILL.md` в корне;
- `MEMORY.md`.

Проверка После Правок

Минимум:

```
runtime\python.exe system_core\doctor.py
runtime\python.exe -m pytest -q
```

Для изменений окна — сборка `install\Build-App.cmd` и список проверок `docs\SMOKE_TEST_RU.md`.