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

```
config\tool_manifest.yaml          GUI command tree, fields, risk metadata
config\gui_settings.yaml           startup GUI settings
config\ui_colors.yaml              theme catalog
config\terminal_commands.json      terminal command cache
system_core\ui_nicegui\app.py      NiceGUI UI
system_core\ui_nicegui\window.py   pywebview wrapper
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.

GUI Boundary

Текущий GUI построен на `operation_groups`.

Правила:

- root/child navigation остаётся внутри command area;
- status, workspace controls и terminal остаются стабильными;

- leaf command показывает параметры и финальные `Запустить` / `Назад`;
- 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`).
- `action_tone` мягко кодирует силу конкретной кнопки цветовым акцентом: `friendly`, `medium`, `danger`. Акцент состоит из слабой заливки и читаемой обводки; это приборная подсказка, не замена `kind`, `risk_level` и подтверждений.

Подробно про дерево: `docs\GUI_TREE_REFACTOR_RU.md`.

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` - полный пользовательский guide и parameter reference.

PDF-копии пользовательских гайдов и инструкций генерируются через `cli\launcher_docs_pdf.cmd` / `system_core\docs_pdf.py` в `docs\PDF`. Dry-run автономен. Для реального рендера путь к внешнему `dev_markdown_pdf_engine.py` задаётся через `--`

`engine`, `AUDION_MARKDOWN_PDF_ENGINE` или находится рядом с семейством проектов Audion; layout/theme общего engine остаются source of truth для PDF-оформления.

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:

- `MANIFEST_REFERENCE_RU.md`;
- `GUI_TREE_REFACTOR_RU.md`;
- `SMOKE_TEST_CHECKLIST_RU.md`;
- `KNOWN_PITFALLS_RU.md`;
- `MEMORY.md`.

Проверка После Правок

Минимум:

```
runtime\python.exe system_core\doctor.py
runtime\python.exe -m py_compile system_core\ui_nicegui\app.py
```

Для GUI/layout изменений дополнительно использовать встроенный браузер и чеклист `docs\SMOKE_TEST_CHECKLIST_RU.md`.