API Description Tool – Solution Design

1. Overview

The API Description Tool converts **OpenAPI 3.0.x YAML specs** (JSON only) into **tabular API descriptions** for both technical and non-technical users.

- **Standalone CLI app** (Windows 11+).
- **Configurable** via text file.
- **Outputs**:
  - Single Excel with 3 sheets (Params, Req Body, Res Body), OR
  - 3 CSVs (one per section).

---

2. Architecture

🔷 High-Level Components

- **CLI Layer** → run app, show progress.
- **Config Manager** → parse INI config.
- **Parser & Validator** → read YAML, validate OpenAPI 3.0.x.
- **Flattener & Normalizer** → expand schemas, handle arrays, constraints.
- **Filter Module** → restrict by path/method.
- **Table Generator** → build tabular structures.
- **Writers** → Excel (3 sheets) or CSV (3 files).
- **Error & Logger** → user messages + optional technical log.

🔷 Data Flow

YAML → Parser → Flattener → Filter → Table Generator → Writer → Output

3. Modules

Module	Purpose
`cli.py`	Arguments, progress
`config.py`	Config parsing
`parser.py`	Load YAML, validate spec
`flattener.py`	Flatten schemas, arrays, combinators
`filter.py`	Apply path/method
`tables.py`	Generate rows for 3 table types
`writer_excel.py`	Excel writer (1 file, 3 sheets)
`writer_csv.py`	CSV writer (3 files)
`logger.py`	Error handling & logs
`tests/`	Unit, functional, regression

4. Outputs

Request Parameters Table

| Name | Mandatory | Expected Value(s) | In | Description* | Examples* |

Request Body Table

| Path | Property | Mandatory | Expected Value(s) | Description* | Examples* |

Response Body Table

| Status | Path | Property | Mandatory | Expected Value(s) | Description* | Examples* |

- Only if enabled in config.

5. Config Example

[input]
spec=openapi:3.0.1
input_format=YAML

[output]
format=xlsx

file_name=api_tab_desc
include_provided_description=True
include_examples=True
include_read_only=True
include_write_only=False
create_log=True

[filtering]
path=/search/package/v1/alternative/flights
method=post

6. Edge Case Handling

- **Combinators**: summarize unless expand=True.
- **Binary bodies**: type=binary, no flatten.
- **Read/Write only**: include, prefix [RO]/[WO].
- **Vendor extensions**: insert into Description.
- **Examples precedence**: schema.examples > content.examples.
- **External $ref**: left as reference string.
- **Response scope**: 2xx + default.

7. Non-Functional Requirements

- Performance: <1 min for 100k lines (Intel i5, 16GB).
- Portability: standalone exe.
- Maintainability: modular + documented.
- Testability: unit + functional + regression.

8. Testing

- **Unit tests** → config parsing, schema flattening.
- **Validation tests** → malformed YAML, unresolved $ref.

- **CLI tests** → invalid args, missing filters.
- **Output tests** → Excel/CSV structure.
- **Performance benchmarks** → 10k–100k lines.
- **Regression tests** → future changes.

9. Future Enhancements

- GUI for config/file selection.
- Confluence integration.
- Web service + natural language query.
- Excel aesthetics (freeze headers, filters).