6 december 2024

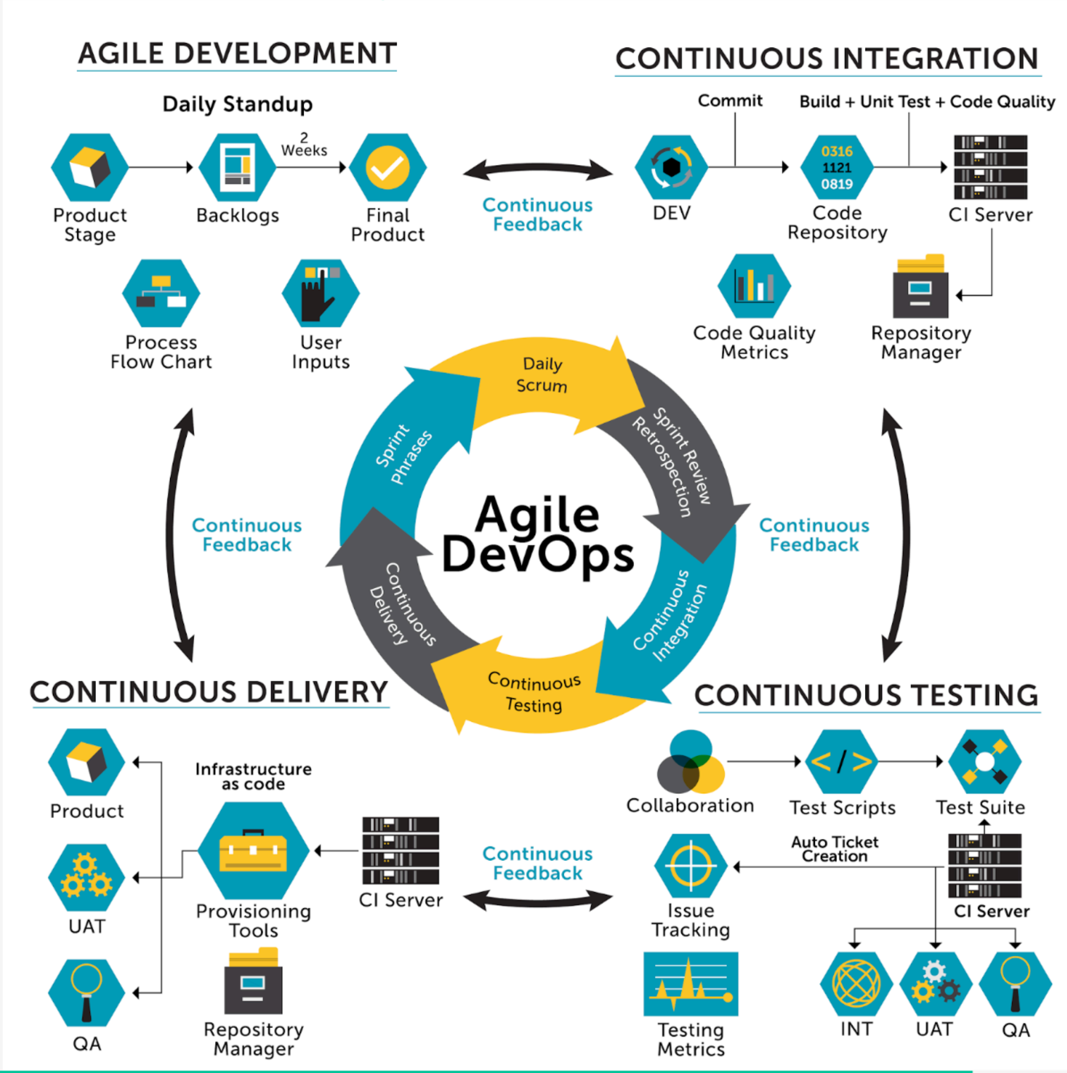
Specialisterne Akademi - Ballerup

CYBER SECURITY TEAM

JEAN M. BABONNEAU

Secure CI/CD Pipeline for PDF Downloader and Metadata Updater

A Case Study in Automation, Security, and GitHub Actions Integration



Introduction

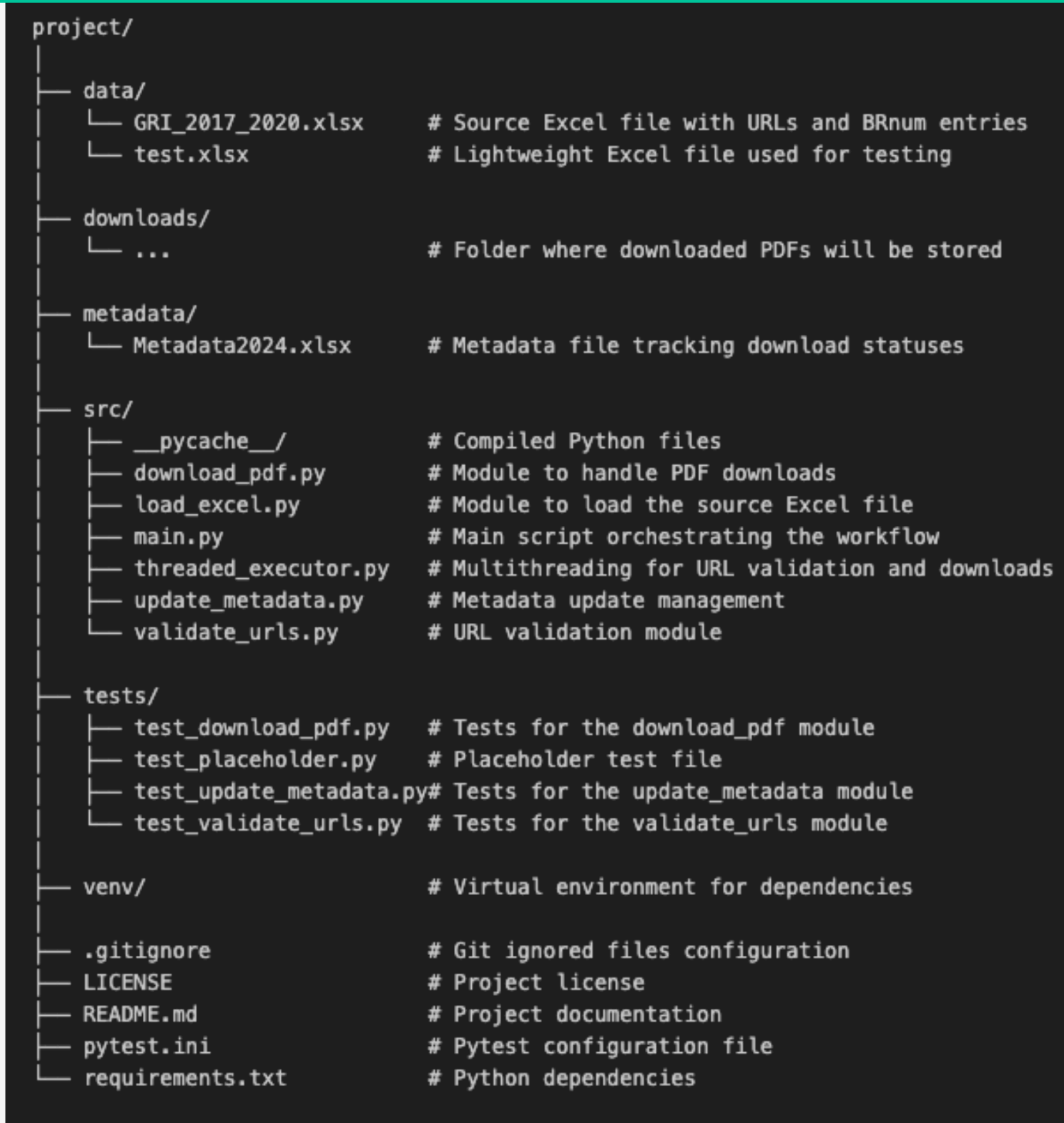
Project Overview

- Automates PDF downloads from Excel-sourced URLs.
- Validates URLs, updates metadata, and manages threading for efficiency.
- Logs results and addresses broken links.

Key Objective

- Automate the process securely while ensuring maintainability (through component modularity)

Modular directory structure & separation of concerns



```
project/
│
├── data/
│   └── GRI_2017_2020.xlsx     # Source Excel file with URLs and BRnum entries
│   └── test.xlsx              # Lightweight Excel file used for testing
│
├── downloads/
│   └── ...                    # Folder where downloaded PDFs will be stored
│
├── metadata/
│   └── Metadata2024.xlsx      # Metadata file tracking download statuses
│
├── src/
│   ├── __pycache__/           # Compiled Python files
│   ├── download_pdf.py        # Module to handle PDF downloads
│   ├── load_excel.py          # Module to load the source Excel file
│   ├── main.py                # Main script orchestrating the workflow
│   ├── threaded_executor.py   # Multithreading for URL validation and downloads
│   ├── update_metadata.py     # Metadata update management
│   └── validate_urls.py       # URL validation module
│
├── tests/
│   ├── test_download_pdf.py   # Tests for the download_pdf module
│   ├── test_placeholder.py    # Placeholder test file
│   ├── test_update_metadata.py# Tests for the update_metadata module
│   └── test_validate_urls.py  # Tests for the validate_urls module
│
├── venv/                      # Virtual environment for dependencies
│
├── .gitignore                 # Git ignored files configuration
├── LICENSE                    # Project license
├── README.md                  # Project documentation
├── pytest.ini                 # Pytest configuration file
└── requirements.txt           # Python dependencies
```

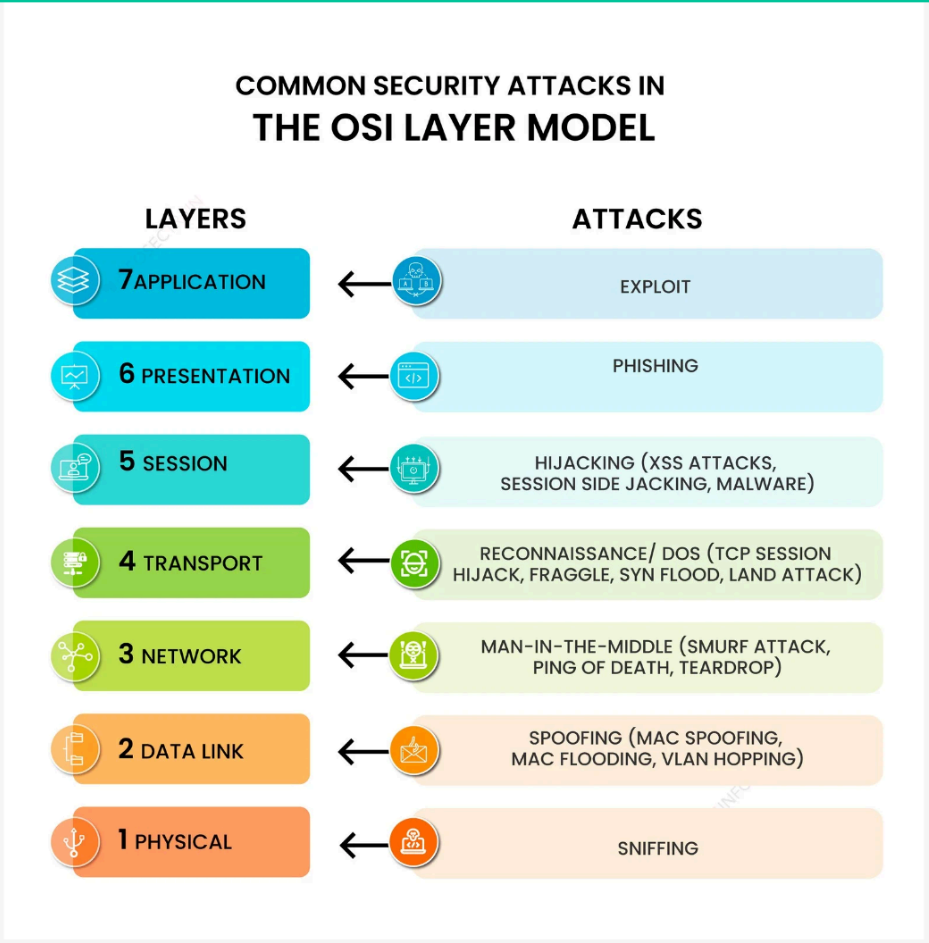
Key Challenges & Vulnerability Analysis

Key Security Risks Identified

- Library Injection: Third-party libraries like pandas and requests could introduce vulnerabilities.
- Secrets Exposure: Hardcoding file paths or API keys could risk exposure.
- Unvalidated Inputs: URLs from the Excel file were not always sanitized, risking injection attacks.
- Lack of initial automated validation for dependencies and file paths.

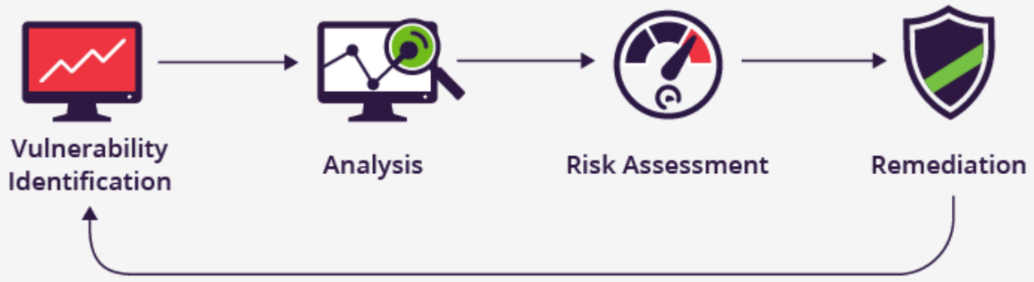
Mitigation Focus

- Automated dependency checks with pip-audit during CI.
- Sanitizing all URLs before processing.
- Using GitHub Secrets to secure sensitive data like file paths.



COMMON SECURITY ATTACKS IN
THE OSI LAYER MODEL

LAYERS	ATTACKS
7 APPLICATION	EXPLOIT
6 PRESENTATION	PHISHING
5 SESSION	HIJACKING (XSS ATTACKS, SESSION SIDE JACKING, MALWARE)
4 TRANSPORT	RECONNAISSANCE/ DOS (TCP SESSION HIJACK, FRAGGLE, SYN FLOOD, LAND ATTACK)
3 NETWORK	MAN-IN-THE-MIDDLE (SMURF ATTACK, PING OF DEATH, TEARDROP)
2 DATA LINK	SPOOFING (MAC SPOOFING, MAC FLOODING, VLAN HOPPING)
1 PHYSICAL	SNIFFING



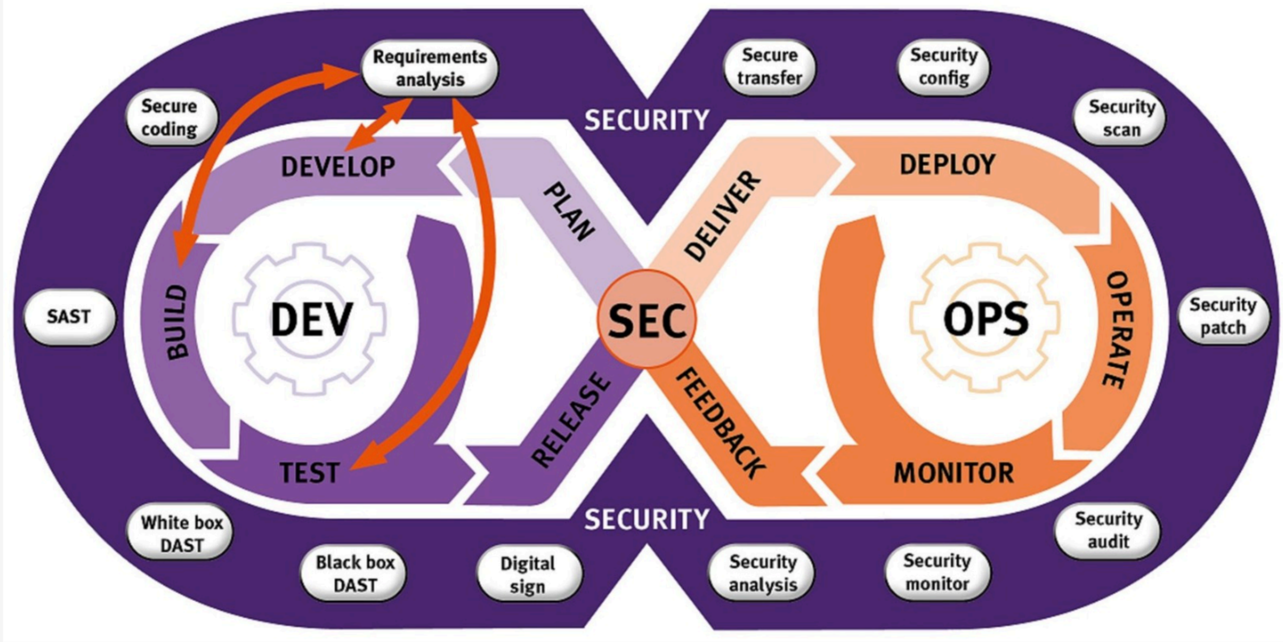
CI Pipeline Design

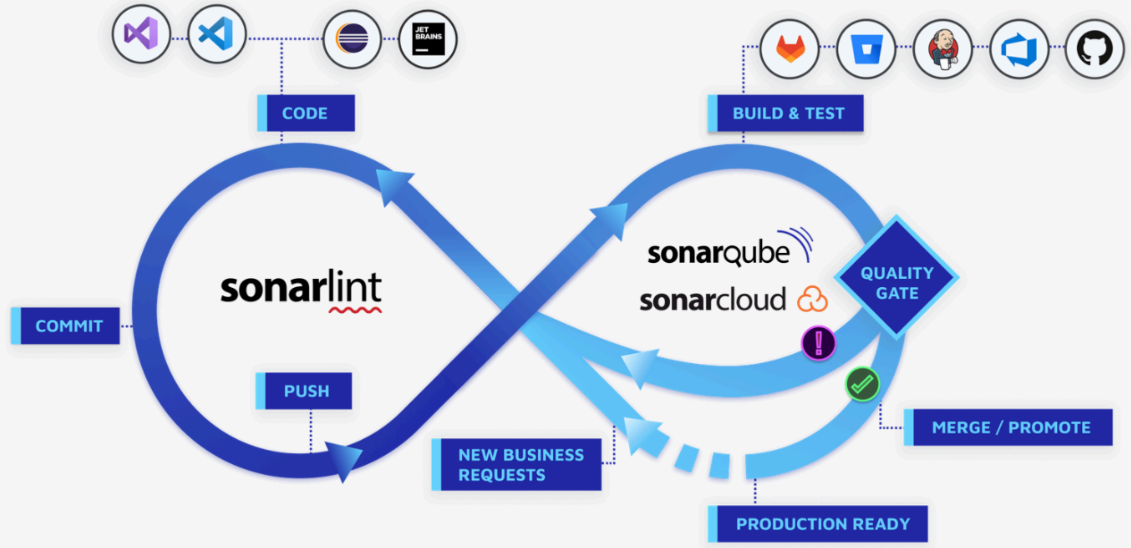
Steps Implemented

- **Code Validation:** Automated linting with flake8.
- **Static Code Analysis / Dependency Management:** Using pip-audit. In a future version, tools like: SonarQube or Snyk could be used.
- **Automated Unit Testing:** unit testing with pytest for key modules: URL validation, metadata updates, PDF downloads.

Why It's Secure

- It automates checks to catch issues early in the development process.

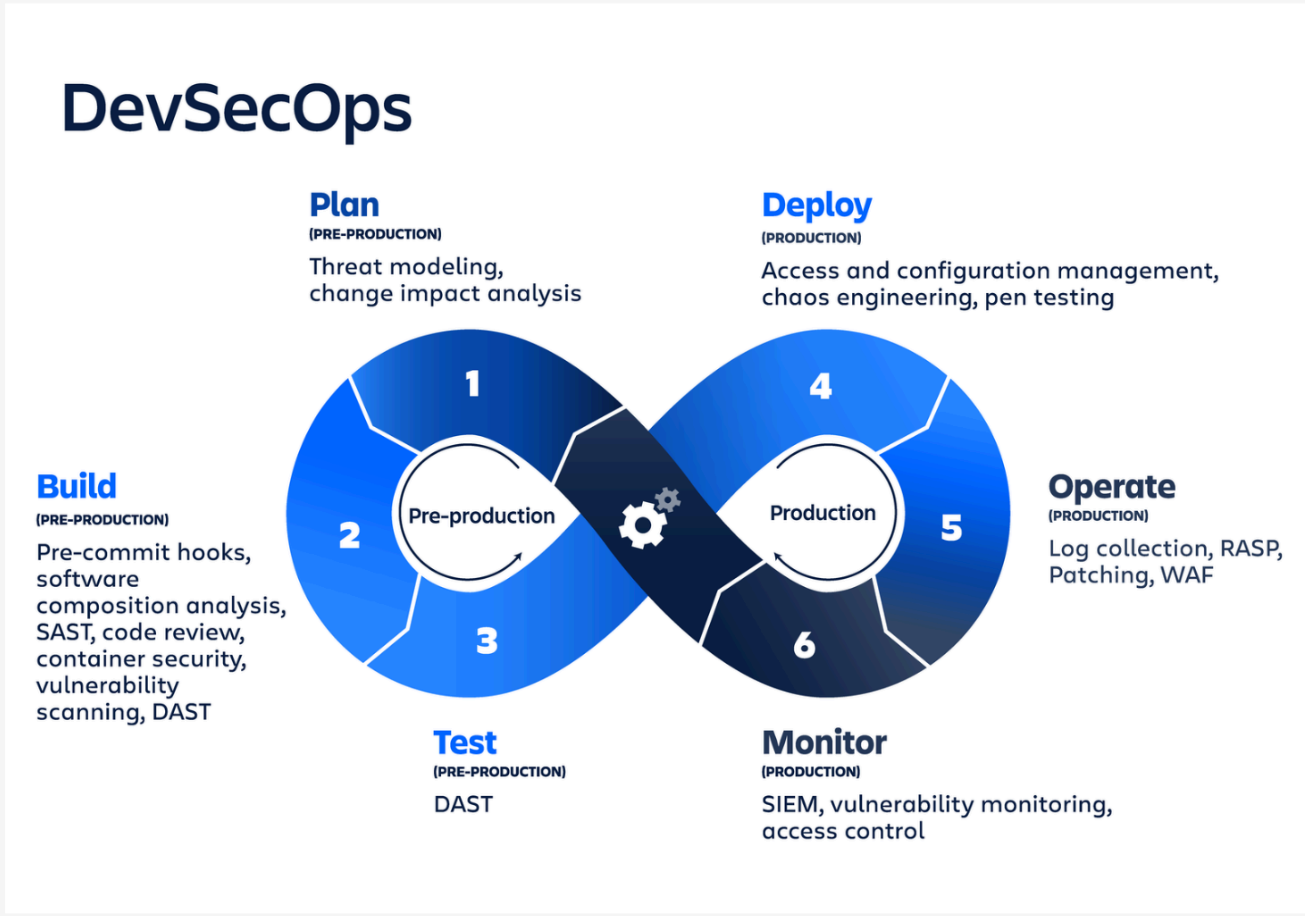




CD Pipeline Vision

Future Steps for CD Pipeline

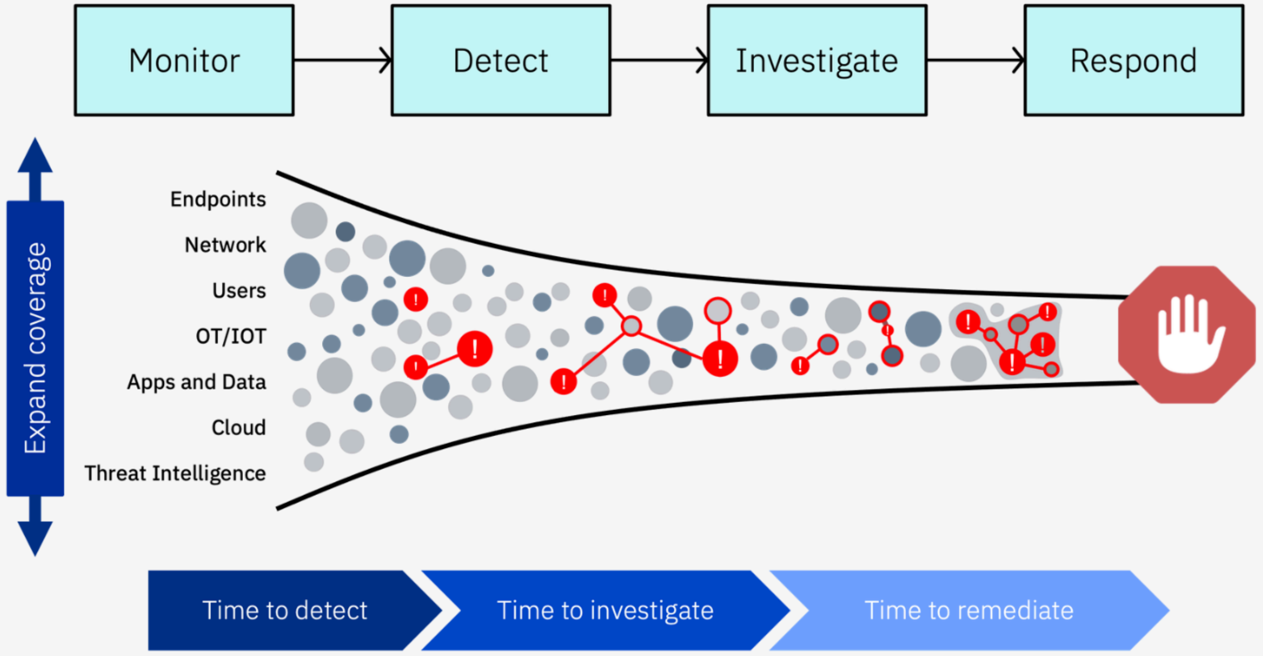
- **Environment Separation:** Separate staging, testing, and production environments.
- **Deployment Automation:** GitHub Actions to deploy updates automatically.
- **Approval Gates:** Manual approval for production releases.
- **Secrets Management:** Use GitHub Secrets for API keys and sensitive data.
- Rollback mechanisms for failure recovery if deployment fails.


DevSecOps
Plan
(PRE-PRODUCTION)
Threat modeling, change impact analysis
Deploy
(PRODUCTION)
Access and configuration management, chaos engineering, pen testing
Build
(PRE-PRODUCTION)
Pre-commit hooks, software composition analysis, SAST, code review, container security, vulnerability scanning, DAST
1
2
3
Pre-production
Production
4
5
6
Operate
(PRODUCTION)
Log collection, RASP, Patching, WAF
Test
(PRE-PRODUCTION)
DAST
Monitor
(PRODUCTION)
SIEM, vulnerability monitoring, access control

Automating Security Checks

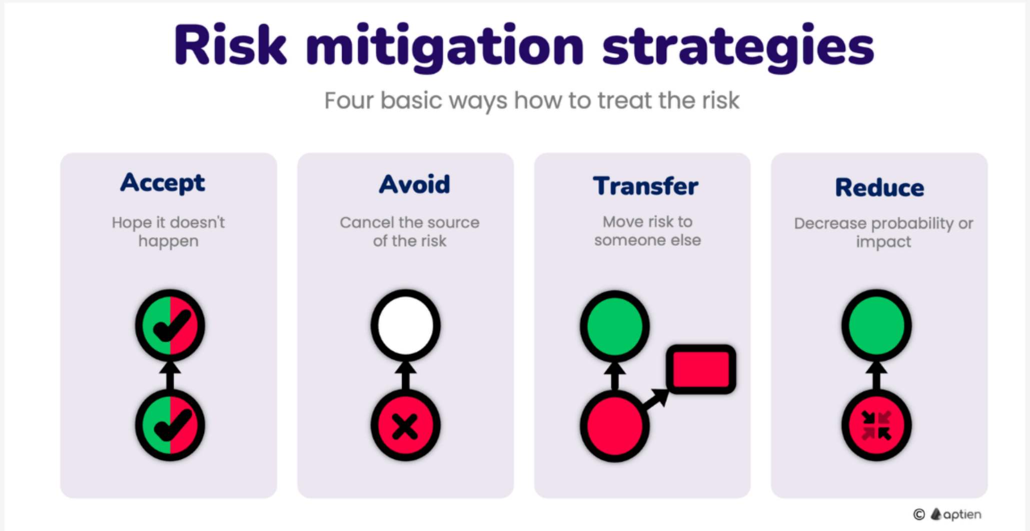
Automated Safeguards

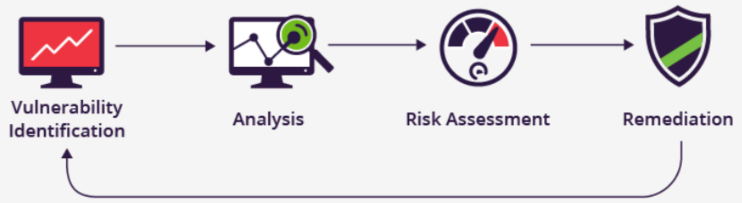
- **Static code analysis tools** catch vulnerabilities (e.g., in Excel parsing logic).
- **Automated rollbacks** if the pipeline detects critical failures.
- **Logging and monitoring for suspicious activities** (e.g., tampering with metadata).



Security Enhancements Examples

- Dependencies audited for vulnerabilities (pip-audit).
- Secrets securely managed using GitHub Secrets.
- Invalid or malicious Excel inputs flagged early in CI.

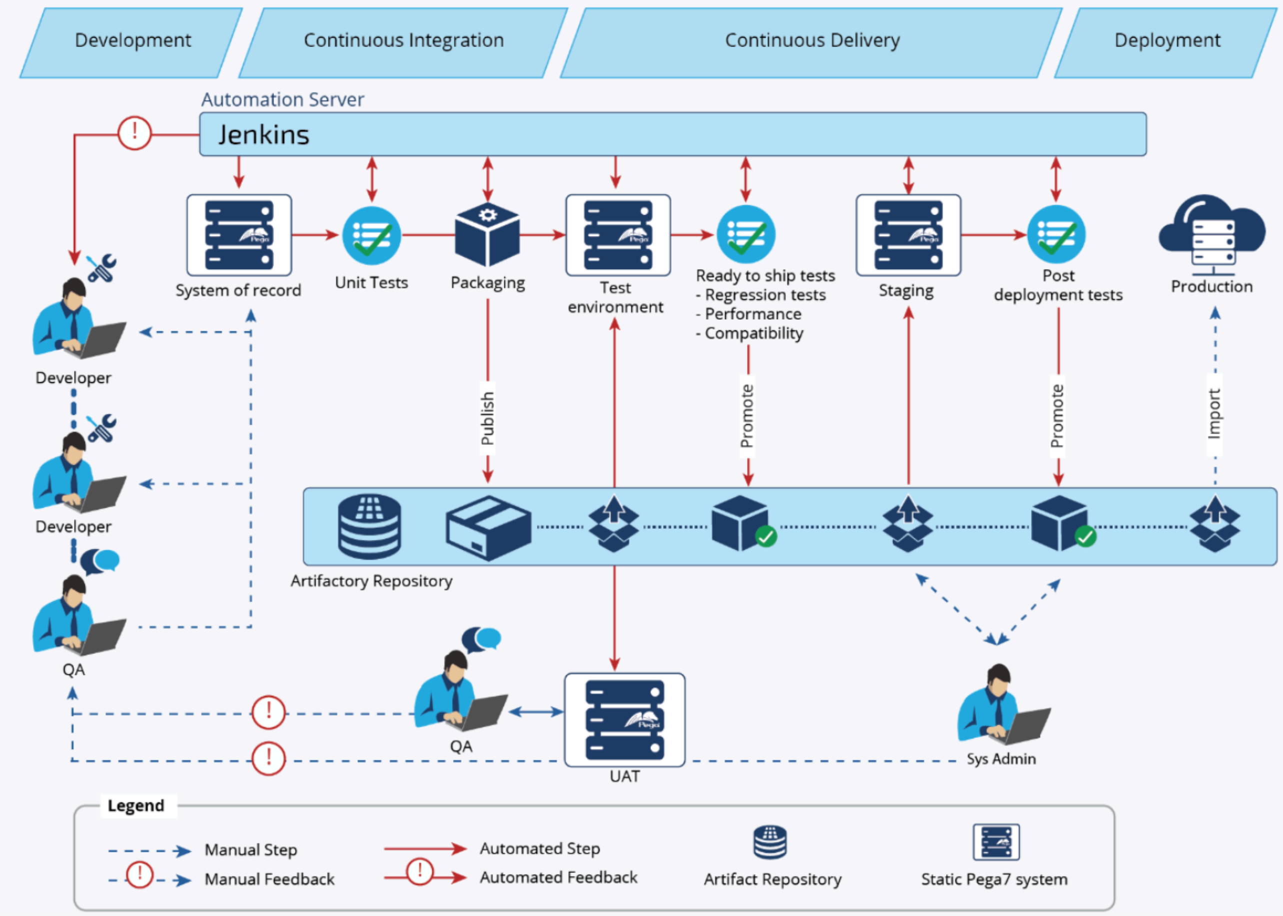

Risk mitigation strategies
Four basic ways how to treat the risk
Accept
Hope it doesn't happen
Avoid
Cancel the source of the risk
Transfer
Move risk to someone else
Reduce
Decrease probability or impact
© aptien

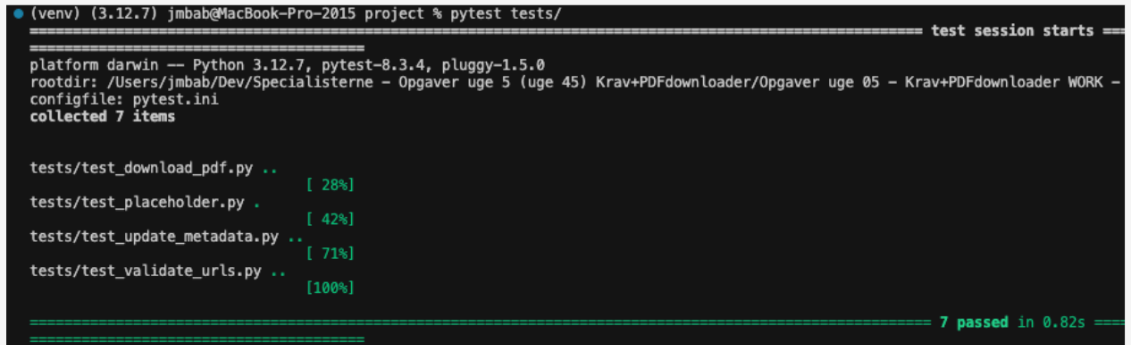


Developer Workflow

Workflow Summary

- Commit and push code to GitHub → Trigger GitHub Actions CI.
- Run automated linting, testing, and dependency scans.
- Log results for debugging and flag potential failures.
- CI pipeline validates, scans, and tests code.
- CD pipeline deploys updates to staging or production.
- Logs and alerts help monitor the process for anomalies.

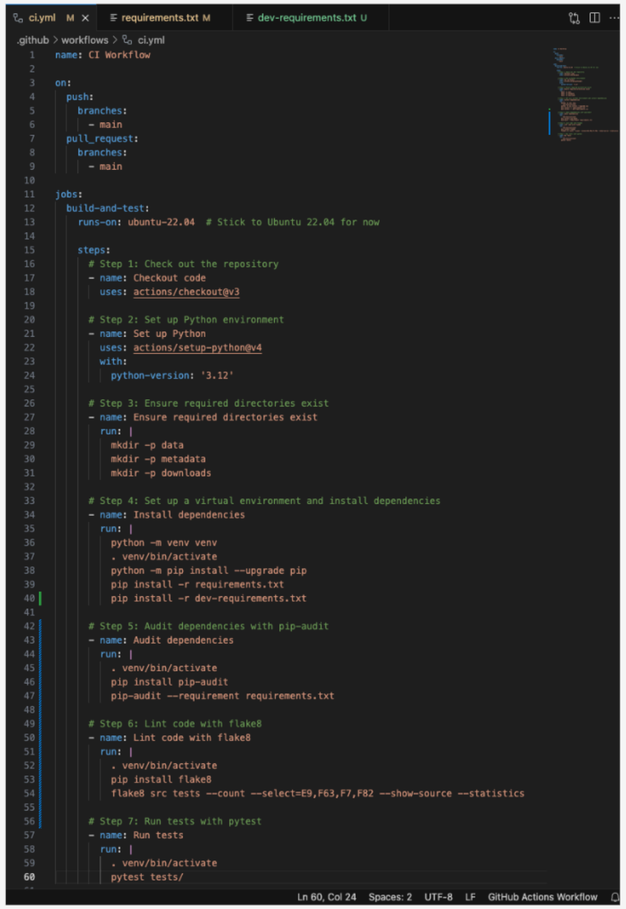



```
● (venv) (3.12.7) jmbab@MacBook-Pro-2015 project % pytest tests/
======================================================================================================= test session starts ===
=========================================
platform darwin -- Python 3.12.7, pytest-8.3.4, pluggy-1.5.0
rootdir: /Users/jmbab/Dev/Specialisterne - Opgaver uge 5 (uge 45) Krav+PDFdownloader/Opgaver uge 05 - Krav+PDFdownloader WORK -
configfile: pytest.ini
collected 7 items


tests/test_download_pdf.py ..
                                 [ 28%]
tests/test_placeholder.py .
                                 [ 42%]
tests/test_update_metadata.py ..
                                 [ 71%]
tests/test_validate_urls.py ..
                                 [100%]

======================================================================================================= 7 passed in 0.82s ===
=========================================
```

Visual Pipeline Design

Pipeline Overview

- **Stages:** Development → → Continuous Integration → → Continuous Delivery.
- **Key automation points:** Dependency management, security checks, testing.

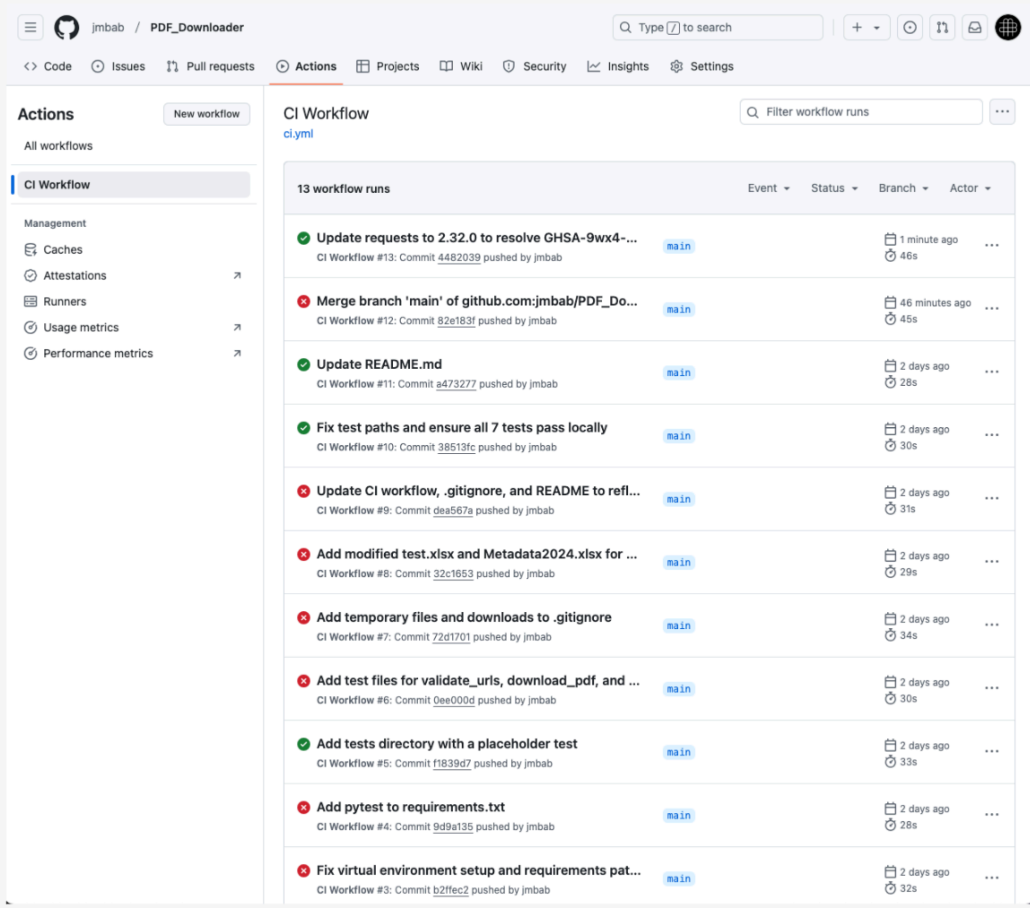


ci.yml M × | requirements.txt M | dev-requirements.txt U

.github > workflows > ci.yml

```yaml
name: CI Workflow

on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main

jobs:
  build-and-test:
    runs-on: ubuntu-22.04  # Stick to Ubuntu 22.04 for now

    steps:
      # Step 1: Check out the repository
      - name: Checkout code
        uses: actions/checkout@v3

      # Step 2: Set up Python environment
      - name: Set up Python
        uses: actions/setup-python@v4
        with:
          python-version: '3.12'

      # Step 3: Ensure required directories exist
      - name: Ensure required directories exist
        run: |
          mkdir -p data
          mkdir -p metadata
          mkdir -p downloads

      # Step 4: Set up a virtual environment and install dependencies
      - name: Install dependencies
        run: |
          python -m venv venv
          . venv/bin/activate
          python -m pip install --upgrade pip
          pip install -r requirements.txt
          pip install -r dev-requirements.txt

      # Step 5: Audit dependencies with pip-audit
      - name: Audit dependencies
        run: |
          . venv/bin/activate
          pip install pip-audit
          pip-audit --requirement requirements.txt

      # Step 6: Lint code with flake8
      - name: Lint code with flake8
        run: |
          . venv/bin/activate
          pip install flake8
          flake8 src tests --count --select=E9,F63,F7,F82 --show-source --statistics

      # Step 7: Run tests with pytest
      - name: Run tests
        run: |
          . venv/bin/activate
          pytest tests/
```

Ln 60, Col 24 | Spaces: 2 | UTF-8 | LF | GitHub Actions Workflow



CI Workflow
ci.yml

13 workflow runs

✓	Update requests to 2.32.0 to resolve GHSA-9wx4-... CI Workflow #13: Commit 4482039 pushed by jmbab	main	1 minute ago	46s
✗	Merge branch 'main' of github.com:jmbab/PDF_Do... CI Workflow #12: Commit 82e183f pushed by jmbab	main	46 minutes ago	45s
✓	Update README.md CI Workflow #11: Commit a473277 pushed by jmbab	main	2 days ago	28s
✓	Fix test paths and ensure all 7 tests pass locally CI Workflow #10: Commit 38513fc pushed by jmbab	main	2 days ago	30s
✗	Update CI workflow, .gitignore, and README to refl... CI Workflow #9: Commit dea567a pushed by jmbab	main	2 days ago	31s
✗	Add modified test.xlsx and Metadata2024.xlsx for ... CI Workflow #8: Commit 32c1653 pushed by jmbab	main	2 days ago	29s
✗	Add temporary files and downloads to .gitignore CI Workflow #7: Commit 72d1701 pushed by jmbab	main	2 days ago	34s
✗	Add test files for validate_urls, download_pdf, and ... CI Workflow #6: Commit 0ee000d pushed by jmbab	main	2 days ago	30s
✓	Add tests directory with a placeholder test CI Workflow #5: Commit f1839d7 pushed by jmbab	main	2 days ago	33s
✗	Add pytest to requirements.txt CI Workflow #4: Commit 9d9a135 pushed by jmbab	main	2 days ago	28s
✗	Fix virtual environment setup and requirements pat... CI Workflow #3: Commit b2ffec2 pushed by jmbab	main	2 days ago	32s

ci.yml file that will trigger automated tests in GitHub Actions

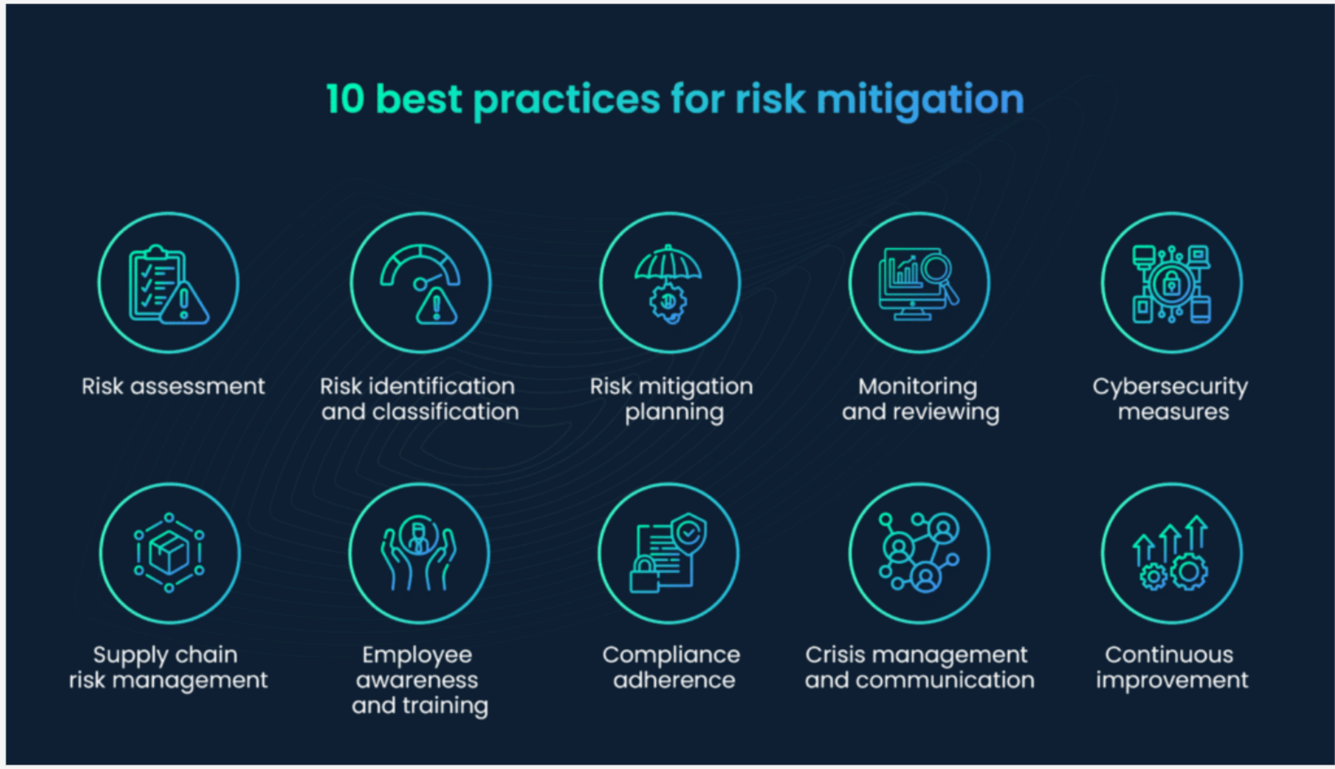
Lessons Learned & Next Steps

Key Takeaways

- CI/CD pipelines improve efficiency, security, and reliability.
- Automation reduces errors and manual overhead.
- Tailored for small projects but scalable for larger ones.
- Focused on mitigating vulnerabilities and automating risk management.

Future Work

- Containerize processes with Docker.
- Expand the pipeline to Integrate advanced security tools like OWASP ZAP or DAST.


10 best practices for risk mitigation
Risk assessment
Risk identification and classification
Risk mitigation planning
Monitoring and reviewing
Cybersecurity measures
Supply chain risk management
Employee awareness and training
Compliance adherence
Crisis management and communication
Continuous improvement

Future User-friendly Features - Nice To Have

Features for the Web Interface

View Downloaded PDF Files

- Display a grid of thumbnails of the first pages of the available PDF files.
- Show file names under the thumbnails, each linking to the full PDF file (opening in a new tab or window).

Pagination

Results should be paginated with:

- A maximum of 20 PDF thumbnails per page.
- A navigation bar (sticky for mobile devices) with:
- Previous and Next buttons (inactive on first/last pages).
- A display of the current page number and total pages.

Responsive Design

- The layout should adapt to different screen sizes:
- On smaller devices, show 1-2 columns.
- The navigation bar should remain visible while scrolling.

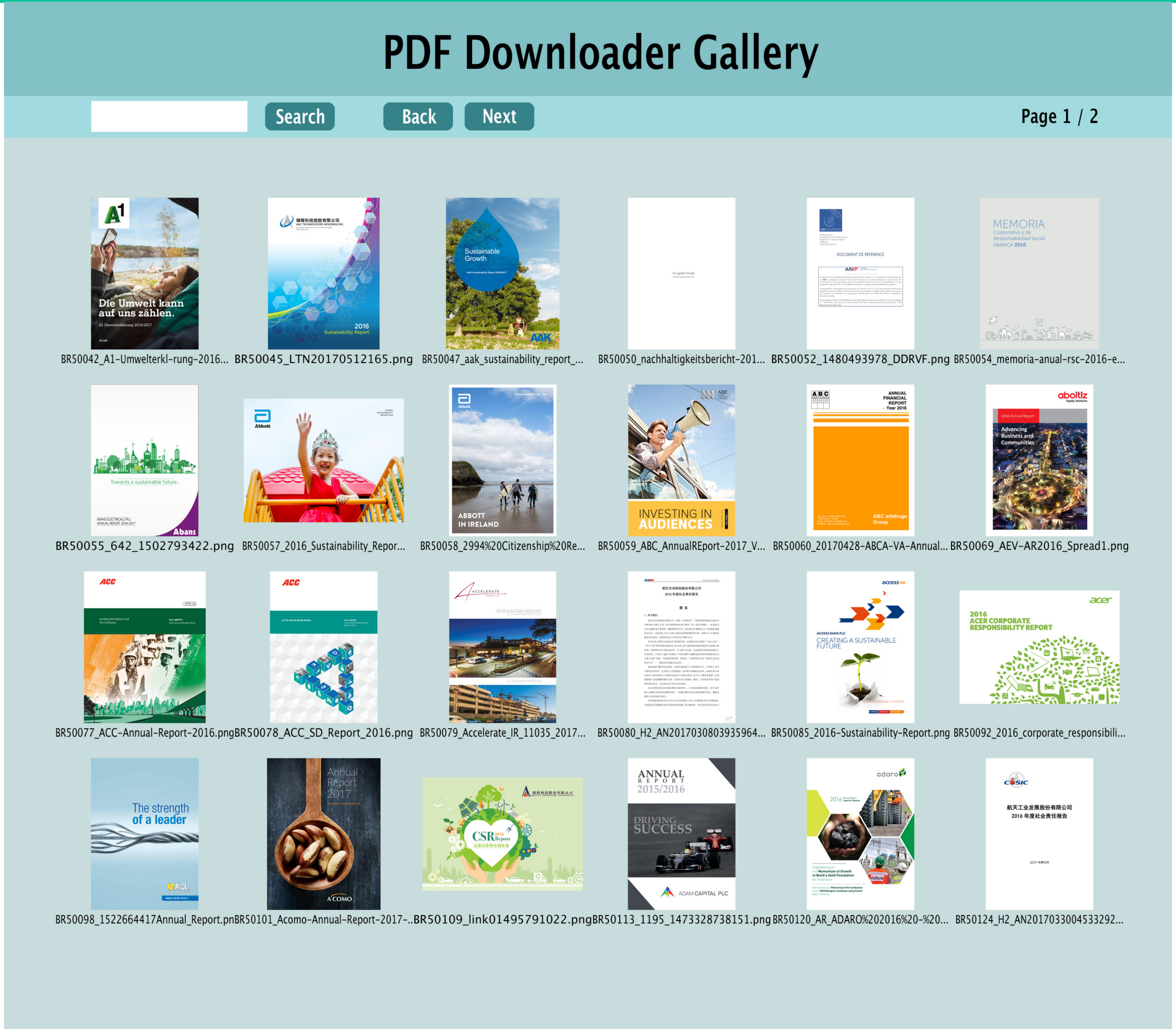
Edit Excel Data File

- Allow users to manually:
- Start, stop, and resume the PDF scanning and downloading process.
- CRUD: Edit the Excel file directly through a web-based editor to add, update, or delete rows.

File Cleanup

- Automate the deletion of invalid PDF files (under 300KB) and update their statuses in the Metadata and Source Excel files.

Future User-friendly Features - Nice To Have



PDF Downloader Gallery

BR50042_A1-Umwelterkl-rung-2016... BR50045_LTN20170512165.png BR50047_aak_sustainability_report_... BR50050_nachhaltigkeitsbericht-201... BR50052_1480493978_DDRVF.png BR50054_memoria-anual-rsc-2016-e...

BR50055_642_1502793422.png BR50057_2016_Sustainability_Repor... BR50058_2994%20Citizenship%20Re... BR50059_ABC_AnnualREport-2017_V... BR50060_20170428-ABCA-VA-Annual... BR50069_AEV-AR2016_Spread1.png

BR50077_ACC-Annual-Report-2016.pngBR50078_ACC_SD_Report_2016.png BR50079_Accelerate_IR_11035_2017... BR50080_H2_AN201703080393964... BR50085_2016-Sustainability-Report.png BR50092_2016_corporate_responsibili...

BR50098_1522664417Annual_Report.pnBR50101_Acomo-Annual-Report-2017-..BR50109_link01495791022.pngBR50113_1195_1473328738151.png BR50120_AR_ADARO%202016%20-%20... BR50124_H2_AN2017033004533292...

Thank you
for your time
and attention!