File Processing Tool

Group Members

Ahmet Sevinç (2001833)

Project Description

The File Processing Tool is a desktop application designed to efficiently handle file compression and decompression operations. It provides a user-friendly graphical interface that allows users to compress individual files or entire folders, as well as decompress previously compressed files. The tool supports various file formats and implements robust error handling to ensure reliable operation.

Implemented Features

- Single file compression using gzip algorithm
- Batch compression of folders
- File decompression capability
- Support for multiple file formats (.txt, .csv, .json, .xml, .log)
- Modern graphical user interface using PyQt5
- Comprehensive error handling and user feedback
- Cross-platform compatibility

Technologies and Libraries

Technology	Purpose
Python 3.8+	Core programming language
PyQt5	Graphical user interface framework
gzip	File compression algorithm
pytest	Testing framework
pathlib	File system operations
typing	Type hints and annotations

Challenges and Solutions

Cross-platform Compatibility

Challenge: Ensuring consistent file path handling across different operating systems. Solution: Implemented pathlib for platform-independent path manipulation.

Error Handling

Challenge: Graceful handling of various error scenarios (invalid files, permissions, etc.). Solution: Implemented comprehensive try-except blocks and user-friendly error messages.

User Interface Design

Challenge: Creating an intuitive and responsive interface. Solution: Utilized PyQt5's layout management system and implemented progress feedback.

Code Snippets

```
def pack_single(self, source_path: Union[str, Path]) -> str:
    """Pack a single file using gzip algorithm."""
    input_path = Path(source_path)
    if not input_path.exists():
        raise FileNotFoundError(f"Unable to locate file: {input_path}")

    if input_path.suffix not in self.allowed_formats:
        raise ValueError(f"Format not supported: {input_path.suffix}")

    result_path = str(input_path) + '.gz'

    with open(input_path, 'rb') as source, gzip.open(result_path, 'wb') as target:
        shutil.copyfileobj(source, target)

    return result_path
```

References

• Python Documentation - https://docs.python.org/
• PyQt5 Documentation - https://www.riverbankcomputing.com/static/Docs/PyQt5/
• gzip Module Documentation - https://docs.python.org/3/library/gzip.html