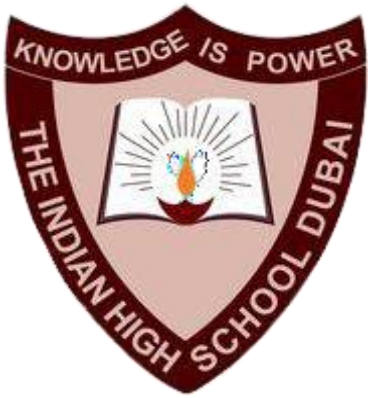
THE INDIAN HIGH SCHOOL - DUBAI



COMPUTER SCIENCE

Project

Documentation

2024-25

Team Members: Abyaz, Abbas, Kenz, Sahil

THE INDIAN HIGH SCHOOL

DUBAI

CERTIFICATE

This is to certify that the work in this journal is the bonafide work of

Masters Abyaz Javid, Mohammed Kenz Valappil, Sahil Ahamed, Abbas Hasan Rangoonwalla

Class 11 SCI *Div* H

recorded in the school lab during the academic year

2024-2025

Date: _____ _____

Teacher in – Charge

INDEX

ACKNOWLEDGEMENT

We would like to take this opportunity to thank the Central Board of Secondary Education (CBSE) and our school The Indian High School-Dubai, for granting me the opportunity to deepen our knowledge in our favorite subject, Computer Science.

We would also like to thank our teacher Ms. Teena Raphel for guiding us and sharing her wide variety of knowledge.

INTRODUCTION

This PDF tool streamlines the management of PDF files by enabling users to efficiently split and merge documents, tasks that are frequently required. Designed with simplicity and ease of use in mind, it allows users to handle their PDFs quickly and without hassle.

Key Advantages:

- Fast and Lightweight: This tool can operate seamlessly on any machine. It can split up to 1,400 PDF pages per second and merge 650 pages per second on mediocre hardware, ensuring more-than-necessary performance.

- Enhanced Security and Privacy: Since the program runs locally, users can manage their PDFs without the risk of sensitive documents being intercepted over the cloud, which can be a threat with online PDF services.

- Reliable and Trustworthy: Unlike many online services that may store and sell uploaded documents without the user's approval, this tool does not retain any processed PDFs, making it a secure choice for handling confidential files.

MODULES & BUILT-IN FUNCTIONS USED

Modules

- `flask` -> For GUI purposes, integrating HTML and CSS
- `os` -> For IO, to move and create PDF files, etc.
- `random` -> For choosing the background color of the app
- `threading` -> To open the app via `webbrowser` whilst also running the app
- `webbrowser` -> To open the web browser
- `json` -> Used for communicating between HTML and Python to understand file operation, metadata, etc.

Built-in Functions used:

- `print()`
- `len()`
- `int()`
- `str()`
- `range()`
- `open()`

PROGRAM SCREENSHOTS

Code to handle “Split PDF” and “Merge PDF”:

```python
def split(file_name: str, page_number: int):
    reader = PdfReader(f"uploads/{file_name}")
    total_pages = len(reader.pages)

    writer1 = PdfWriter()
    writer2 = PdfWriter()

    for i in range(total_pages):
        if i < page_number:
            writer1.add_page(reader.pages[i])
        else:
            writer2.add_page(reader.pages[i])

    output_filename1 = path.join(DOWNLOADS_PATH, f"{file_name}_part_1.pdf")

    with open(output_filename1, "wb") as output_file1:
        writer1.write(output_file1)

    if total_pages > page_number:
        output_filename2 = path.join(DOWNLOADS_PATH, f"{file_name}_part_2.pdf")
        with open(output_filename2, "wb") as output_file2:
            writer2.write(output_file2)

    PDFHandler.cleanup()


def merge(files: list[str]):
    merger = PdfWriter()

    for file in files:
        merger.append(path.join("uploads", file))

    merger.write(path.join(DOWNLOADS_PATH, "merged.pdf"))
    merger.close()
    PDFHandler.cleanup()
```

Code to handle when a file is uploaded:

```python
def upload_file():

    files = request.files.getlist("file")
    metadata = request.form.get("metadata")

    allowedExts = {".pdf"}
    for i in files:
        if i.filename == "":
            return jsonify({"error": "No selected file"}), 400

        if i.filename is None:
            return jsonify({"error": "File name is None"}), 400
        ext = path.splitext(i.filename)[1].lower()
        if ext not in allowedExts:
            return jsonify({"error": f"Invalid file type for {i.filename}"}), 400

    uploaded_files = []
    for i in files:
        name = i.filename
        if name is not None:
            file_path = path.join(app.config["UPLOAD_FOLDER"], name)

            try:
                i.save(file_path)
                uploaded_files.append(file_path)
            except OSError:
                return jsonify({"error": "Failed to save file"}), 500

    with open(path.join(app.config["UPLOAD_FOLDER"], "metadata.json"), "w") as f:
        dump(metadata, f, indent=4)

    Handler.handle()
    return jsonify({"message": "Operation completed successfully."}), 200
```

User interface: