**PDF TOOLS SUITE**

Semester VI Academic Year 2024-25 A Project Submitted to

University of Mumbai for Partial Completion of the Degree of Bachelor of Science (Computer Science)

Under the Faculty of Science By

**Shravani Gajanan Sawant**

**243104**

**Under the Guidance of**

**Asst. Prof. Sampurna Mishra**



**Saket College of Arts, Science and Commerce, Kalyan East UNIVERSITY OF MUMBAI**



SAKET GYANPEETH’S

**SAKET COLLEGE OF ARTS, SCIENCE & COMMERCE**

**(Affiliated to University of Mumbai) NAAC Accredited B Grade**

**Department of Computer Science**

CERTIFICATE

**This is to certify that**

**Shravani Gajanan Sawant**

Has Completed the Project Work Entitled

**PDF TOOLS SUITE**

Submitted the same in the Partial Fulfillment of Bachelor of Computer Science

Degreeof Mumbai University.

**Project Guide Head of the Department**

**Principal Internal Examiner External Examiner**

# Declaration

I, **Shravani Gajanan Sawant** hereby declare that the project entitled **“PDF TOOLS SUITE”** done at **“Saket College of Arts, Science & Commerce”** submitted in the partial fulfilment for the award of **BACHELOR OF SCIENCE IN COMPUTER SCIENCE.** During the academic year 2024-25 is our original work and the project has not formed the basic for the award of any degree, associateship, fellowship or any other similar titles.

**Signature of Students,**

**Place:**

**Date:**

**Abstract**

The **PDF Tools Suite** is a comprehensive, web-based platform designed to simplify the manipulation and editing of PDF documents. Built with modern web technologies such as HTML, CSS, and JavaScript, this suite offers a wide range of powerful tools that address common PDF-related tasks.

These include **PDF compression**, which reduces file size without compromising quality; **JPG to PDF,PDF to Word** and **Word to PDF** converters, enabling seamless format transformation; **PDF splitting and merging**, allowing users to extract and combine pages or entire documents; **adding page numbers** for enhanced document organization; **password protection** for securing sensitive content; which extracts text from scanned or image-based PDFs for easier editing and searching. Accessible directly through a web browser, the PDF Tools Suite provides an intuitive, no-installation-required solution for users, from casual users to professionals, needing to manage their PDFs quickly and efficiently.

By combining ease of use with powerful functionality, the platform makes it easier than ever to handle a variety of PDF tasks in a single, user-friendly interface. Whether you're reducing file sizes for email sharing, converting documents for compatibility, or protecting sensitive files, the PDF Tools Suite is a versatile toolset that meets the needs of today's digital document workflows.

# Acknowledgement

"I would like to express my sincere gratitude to those who have contributed to the development and success of the Taxi booking system. While the project was primarily conceived and executed by an individual, the support and assistance from various sources were instrumental in its completion.

I am especially grateful to my mentor, **Asst. prof. Sampurna Mishra**, whose expertise, guidance, and encouragement were invaluable throughout the development process.I would also like to acknowledge the contributions of our respected Principal**, Prof. (Dr.) Vasant D. Barhate**, whose visionary leadership and commitment to innovation have fostered an environment conducive to academic excellence. Their unwavering support and encouragement have been a source of motivation throughout the project.

Furthermore, I extend my gratitude to our esteemed Vice-Principal, **Dr. Praseena Biju**, whose guidance and wisdom have been instrumental in shaping the project.

To my friends and family, thank you for your unwavering support and encouragement. Your belief in my abilities has been a driving force throughout the project journey.

Lastly, I express my deepest appreciation to the users of the Taxi booking system for their engagement, feedback, and willingness to participate in the testing and development phases. Your valuable input has contributed significantly to the refinement and improvement of the application.

With sincere Thanks,

Shravani G. Sawant

# Approval for Project Proposal

PRN NO.:- 2022016401788361

Roll No.:- 243104

1. Name of the Students: - Shravani G. Sawant
2. Title of the Project: - PDF Tools Suite
3. Name of the Guide: - Asst. Prof Sampurna Mishra
4. Teaching experience of the Guide: -
5. Is this your first submission? Yes  No 

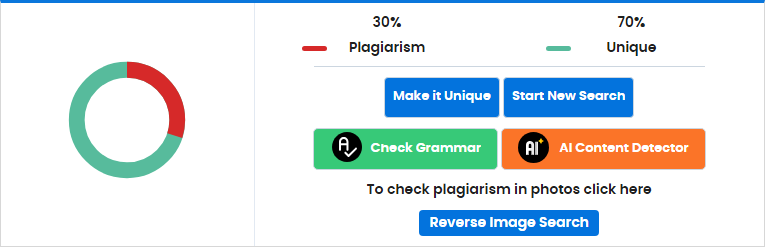
Signature of the Student: Signature of the Guide:

Date: Date:

Signature of the Coordinator:

Date:

# Plagiarism Report



**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Content** | **Page No.** |
| 1 | Introduction | 1-2 |
| 2 | Feasibility study | 3-4 |
| 3 | Scope of Project | 5-6 |
| 4 | Software & Hardware Requirements | 7 |
| 5 | System Design | 8 |
| 6 | UML | 9-16 |
| 7 | Database Design | 17-18 |
| 8 | System Testing | 19-20 |
| 9 | Output Of the Screen | 21-27 |
| 10 | Conclusion | 29 |
| 11 | Bibliography | 31 |
| 12 | Appendices | 33-35 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **UML Diagrams** | **Page no.** |
| 1 | Use Case Diagram | 19 |
| 2 | Class Diagram | 20 |
| 3 | Entity-Relationship Diagram | 21-24 |
| 4 | 0 level DFD | 25 |
| 5 | 1 level DFD | 26 |
| 6 | 2 level DFD | 27 |
| 7 | Sequence Diagram | 28 |
| 8 | Activity Diagram | 29 |

# Introduction

PDF Tools Suite is a comprehensive web-based platform designed to provide a range of PDF-related functionalities. This suite offers users an intuitive and efficient way to perform various operations on PDF files, including conversion, merging, splitting, encryption, and more. The project aims to deliver a seamless experience with a user-friendly interface and a consistent design theme.

Managing and editing PDFs can be challenging without the right tools. PDF Tools Suite is developed to simplify these tasks, allowing users to efficiently handle their documents online. The suite is particularly beneficial for students, professionals, and businesses that regularly work with PDFs and require an all-in-one solution for their document processing needs.

As the project evolves, additional features such as e-signatures, OCR, PDF annotations, and cloud storage integration may be incorporated. The aim is to create a highly versatile and scalable PDF processing suite that caters to a wide range of user needs.

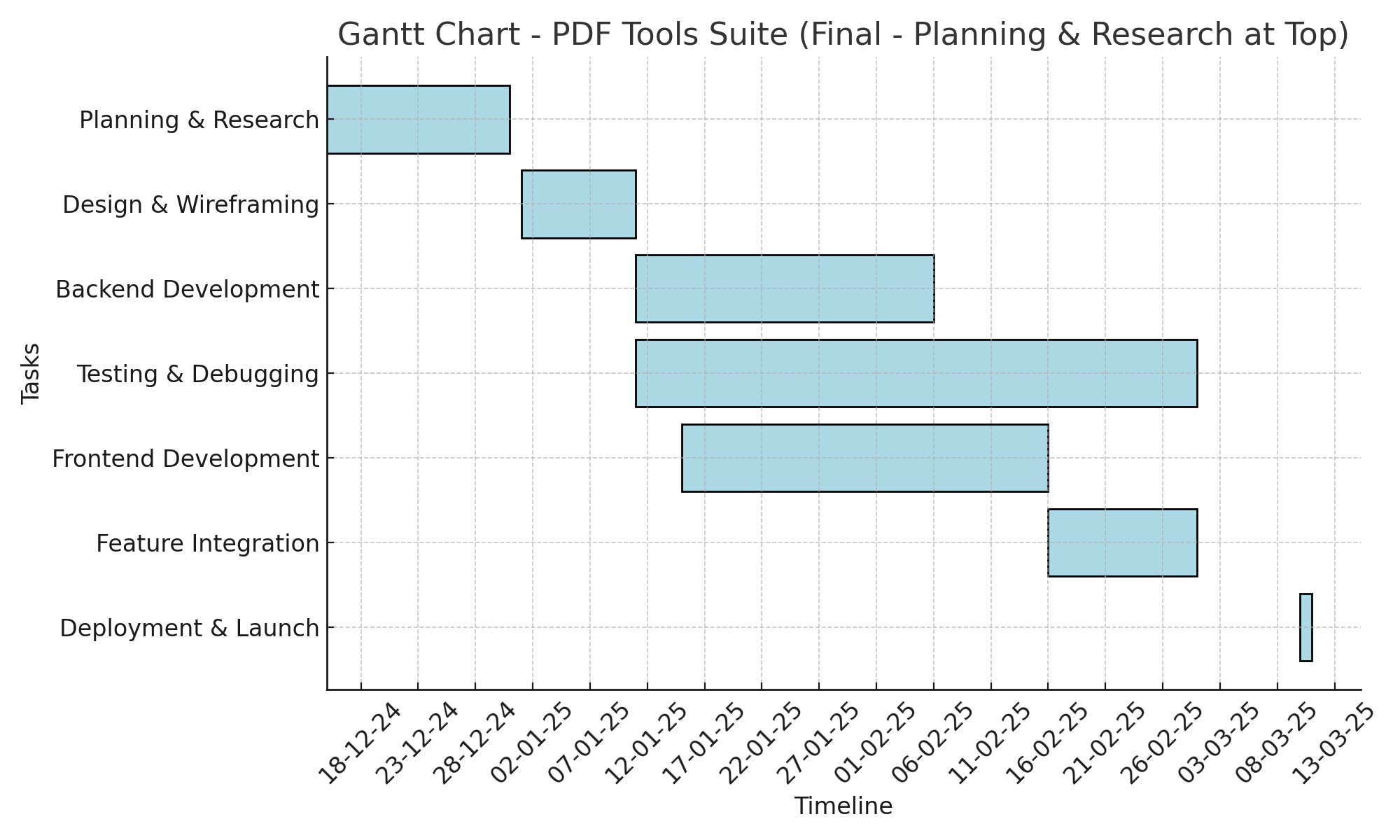
### Advantages:

* **Comprehensive Feature Set** The suite offers a wide range of tools, including merging, splitting, compressing, extracting text, converting PDFs to images, and adding watermarks. This eliminates the need for multiple software applications, saving time and effort.
* **User-Friendly Interface** The suite is designed to cater to both technical and non-technical users. A graphical user interface (GUI) makes it easy for beginners, while a command-line interface (CLI) and API provide flexibility for advanced users.
* **Cross-Platform Compatibility** Built using modern programming practices, the suite is compatible with major operating systems (Windows, macOS, Linux), ensuring accessibility for a wide range of users.
* **Open-Source and Customizable** Being open-source, the suite allows developers to modify and extend its functionality to suit specific needs. This also fosters a community-driven approach to improvement and innovation.
* **Efficient Performance** The suite is lightweight and optimized for performance, capable of handling large PDF files efficiently without significant delays or crashes.
* **Cost-Effective** As an open-source tool, the PDF Tools Suite is free to use, making it an affordable alternative to expensive proprietary PDF software.
* **Preservation of Document Quality** Tools like compression and watermarking are designed to maintain the quality of the original document while performing the required operations.
* **Community Support** Being open-source, the suite benefits from community contributions, regular updates, and a wealth of user-generated resources like tutorials and forums.

### Disadvantages:

* **Learning Curve for Advanced Features** While the GUI is user-friendly, advanced features like CLI and API usage may require technical expertise, which could be a barrier for non-technical users.
* **Dependency on External Libraries** The suite relies on third-party libraries for certain functionalities, which may introduce compatibility issues or require additional setup steps.
* **Performance with Extremely Large Files** While the suite is optimized for performance, processing extremely large PDF files (e.g., hundreds of pages or high-resolution images) may still result in slower performance or higher memory usage.
* **Limited Customer Support** As an open-source project, the suite may not offer dedicated customer support. Users may need to rely on community forums or documentation for troubleshooting.
* **Potential Security Concerns** Since the suite is open-source, users need to ensure they download it from trusted sources to avoid malicious versions. Additionally, handling sensitive documents may require extra precautions.
* **Feature Limitations Compared to Premium Tools** While the suite covers most common PDF tasks, it may lack some niche features offered by premium PDF software, such as advanced redaction, digital signatures, or cloud integration.
* **Requires Technical Setup for Customization** Customizing the suite or adding new features may require programming knowledge, which could be a limitation for users without a technical background.

**Project Gantt chart:**



# Feasibility study

A **feasibility study** is a critical evaluation of the proposed project to determine whether it is viable. It examines the potential for success in terms of financial, operational, technical, and market aspects. Here's a breakdown of the **feasibility study** for the **PDF Tools Suite**

**A. Technical Feasibility**

**1. Technology Requirements:**

* Development using **Flask** (Python) and **Node.js** (JavaScript) for backend functionality.
* Frontend technologies: **HTML, CSS, JavaScript, Bootstrap** for user interface.
* PDF processing libraries: **PyMuPDF (fitz), pdf2image, pdfkit, pdfminer, pdf-lib** (for Node.js).
* Database (if required): **SQLite, PostgreSQL, or MongoDB** for storing user preferences

**2. Deployment Requirements:**

* Hosting: **Cloud-based solutions (AWS, DigitalOcean, or Heroku) or self-hosted**.
* Platform: **Web-based, cross-platform compatibility** (Windows, macOS, Linux).
* API support: **For automation and third-party integration**.

**3. Risks and Challenges:**

* Handling **large PDFs** efficiently without performance issues.
* Ensuring **data security and privacy** for user documents.
* Managing **dependencies on third-party libraries**.

#### **B. Market Overview**

PDF files are widely used in **business, education, government, and personal document management**. The demand for **PDF manipulation tools** is increasing due to the rise in **remote work, digital document storage, and paperless transactions**.

##### **Key Market Insights:**

* The global **PDF software market** is growing due to the increasing need for **secure, editable, and shareable digital documents**.
* Many users look for **free or affordable alternatives** to expensive software like **Adobe Acrobat**.
* Businesses require **automation tools** to process large volumes of PDFs efficiently.

#### **C. Economic Feasibility**

**1. Development Costs:**

* **Open-source libraries** reduce software licensing costs.
* Hosting and server costs (if cloud-based).
* Developer costs (if hiring a team for enhancements).

**2. Revenue Model (if monetized):**

* **Free with premium features** (e.g., advanced PDF editing, cloud storage).
* **Subscription-based or pay-per-use** for businesses.
* **Advertisements (if applicable)** to generate revenue.

**3. Cost vs. Benefit Analysis:**

* Reduces dependency on **expensive proprietary software** like Adobe Acrobat.
* Increases productivity by providing **all PDF tools in one suite**

#### **D. Operational Feasibility**

**1. Usability & Accessibility:**

* **User-friendly UI** for non-technical users.
* **CLI & API support** for advanced users and automation.
* **Multi-language support** for a broader audience.

**2. Maintenance & Scalability:**

* Regular updates through **GitHub or other version control systems**.
* Can be **scaled by adding cloud-based processing** for large files.

**3. Stakeholders & End Users:**

* **Businesses**, **students**, **researchers**, and **general users** who need PDF tools.
* **Developers** looking for API-based PDF processing.

#### **E. Legal Feasibility**

**1. Compliance & Licensing:**

* Uses **open-source libraries**, ensuring no legal restrictions.
* Needs to comply with **data privacy laws (GDPR, CCPA, etc.)** if storing user files.

**2. Security & Data Protection:**

* Implement **end-to-end encryption** for file processing.
* Temporary storage of user files with **auto-deletion after processing**.

# Scope of the Project

The **scope of a project** refers to the detailed description of what the project will accomplish, what deliverables will be produced, and the boundaries within which the project will operate.

It outlines the specific objectives, tasks, and outcomes of the project, providing a clear understanding of the project’s goals and expectations for all stakeholders involved.

It also defines the limitations, constraints, and exclusions to ensure that the project remains focused and manageable.

### ****Objectives****

The primary goal of the **PDF Tools Suite** is to simplify PDF management by offering an intuitive **GUI and CLI** for both regular users and developers. It focuses on **automation and integration** with an API, ensuring high performance and security. The project also aims to be a **free alternative** to expensive proprietary software while maintaining **data privacy and encryption**.

### ****Services Offered****

The suite provides multiple **PDF-related features**, including merging and splitting PDFs, compressing files for size reduction, converting PDFs to Word and vice versa, extracting text and images, adding watermarks, and numbering pages. Additionally, it offers **API and CLI support**, enabling seamless automation and integration into larger workflows.

### ****Project Deliverables****

Key deliverables include a **fully functional web application**, a **CLI-based standalone tool**, and an **API with documentation** for integration. The suite will also provide **detailed user guides and video tutorials** for better accessibility. The entire project will be **open-source**, allowing community contributions for future enhancements.

### ****Target Audience****

The tool is designed for **students, educators, businesses, freelancers, legal professionals, and developers** who frequently work with PDF documents. It caters to a broad audience, helping users **organize study materials, manage contracts, automate tasks, and streamline document workflows** efficiently.

### ****Timeline****

The project is estimated to be completed in **eight months**, divided into different phases: **planning and research (1 month), UI/UX design (1 month), core development (3 months), testing (2 months), deployment (1 month), and ongoing updates**. Future enhancements will be based on community feedback and evolving user needs.

### ****Exclusions****

The first version of the **PDF Tools Suite** will **not** include **advanced PDF editing, cloud storage, digital signatures, or AI-based OCR**. However, these features may be added in later updates based on demand and feasibility.

**Software & Hardware requirements**

Here are the software requirements and hardware requirements for the project.

* **Software Requirements :**
  + Server: localhost
  + Frontend: HTML, CSS
  + Scripting Language: JavaScript
  + IDE: Visual Studio Code
  + Backend Technology: Node.js,python
  + Backend Framework : Express.js,python
* **Hardware Requirements :**
  + Processor: Any processor after Pentium 4 or equivalent (e.g., Intel Core i3, i5)
  + Operating System: Windows 10 or later
  + Processor Speed: 2.0 GHz or higher
  + RAM: 4GB or more
  + Hard Disk: 80GB to 256GB SSD or HDD

**System Design**

The system design for the tuition service project includes the following key components:

* **Frontend (Client Side):**
* User interface for students and tutors (login, course listings, chat).
* Technologies: HTML, CSS, JavaScript
* **Backend (Server Side):**
* Web server handling requests and generating dynamic content.
* Technologies: Express.js
* Database: Stores user info, courses, schedules.
* Technologies: Firebase
* **Security**:
* SSL/TLS encryption, secure authentication, and data protection.
* **System Flow:**
* Users (students/tutors) register and log in.
* Students enroll in courses, and tutors manage them.

This design ensures smooth operation of the platform with secure and efficient communication and course management.

# Unified Modelling Language Diagrams (UML)

# For the PDF Tools Suite, several UML (Unified Modeling Language) diagrams can help represent different aspects of the system design. Below are the key UML diagrams that would be useful for this project:

# Use Case Diagram :

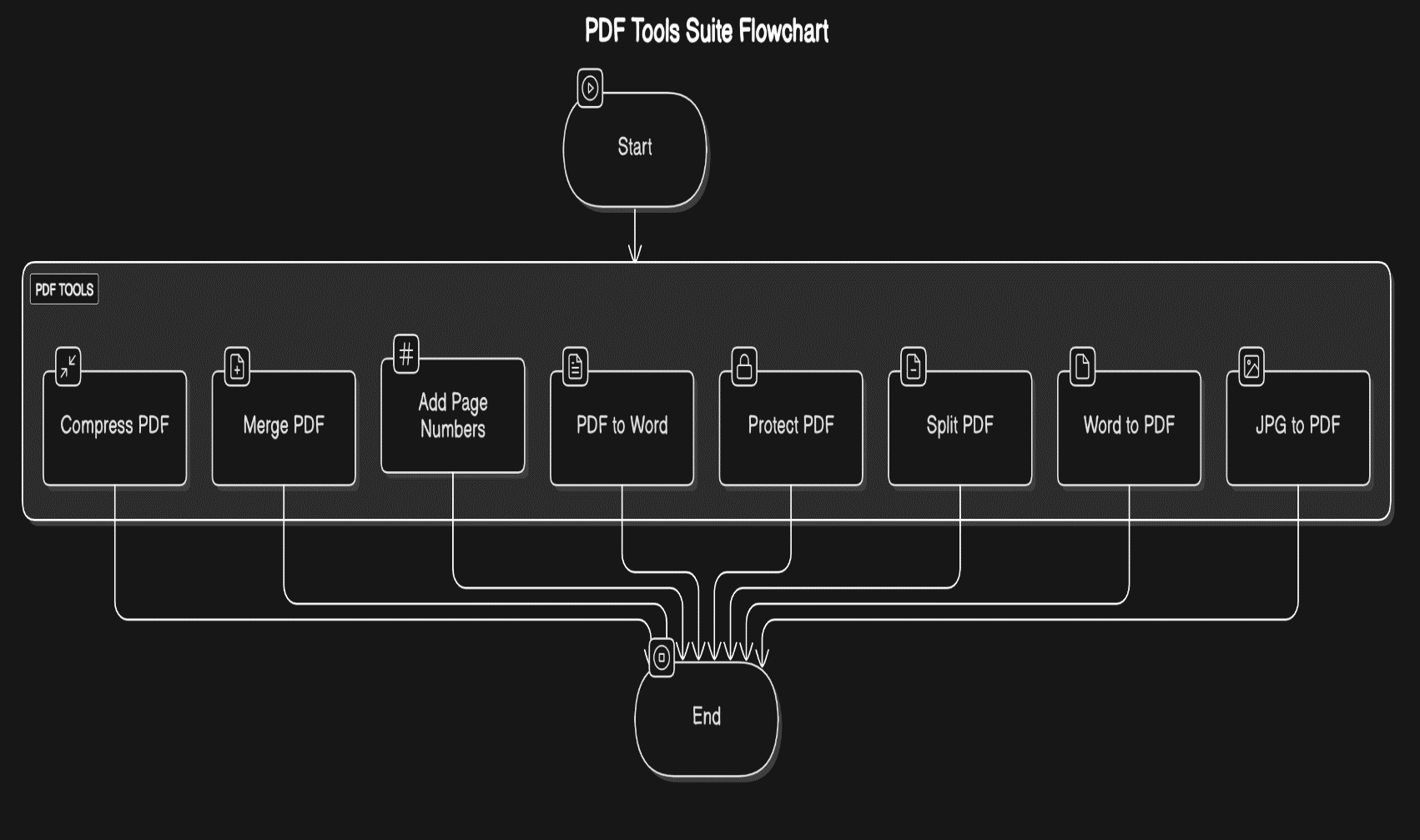
# Purpose: This diagram represents the high-level functionality of the system and the interactions between users and the system.

# 

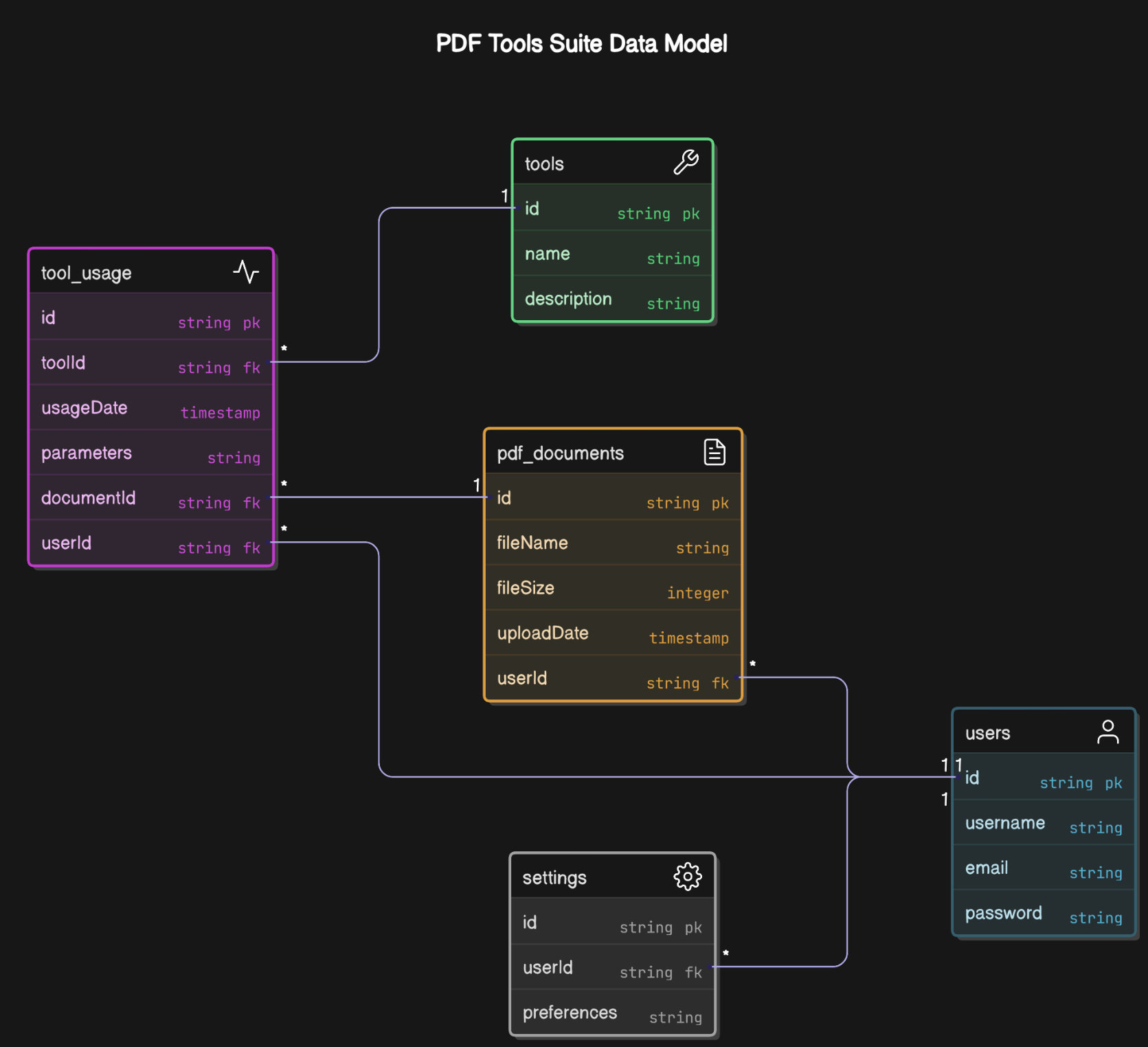
# 

* **Class Diagram :**

Purpose: This diagram shows the structure of the system by representing classes, attributes, methods, and relationships between classes.

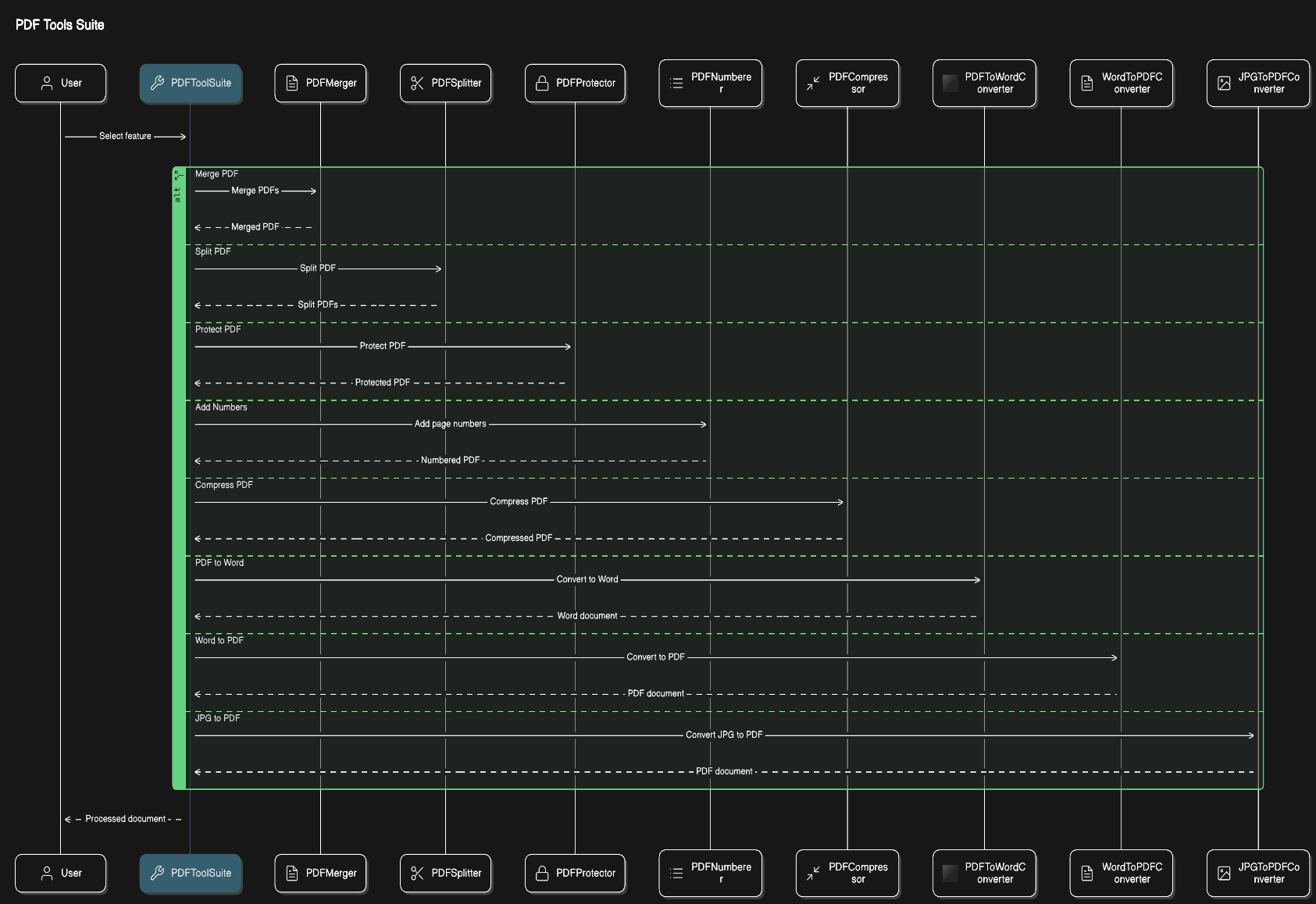
****

* **ER Diagram :**

****

* **Sequence Diagram :**

Purpose: This diagram shows how objects interact in a sequence of events for a particular use case .



### Activity Diagram :

### Purpose : An Activity Diagram visually represents the workflow of processes within the tuition service system. It helps in understanding the sequence of actions, decision points, and interactions between users and the system.

### 

**System Testing**

System testing is a crucial phase in software development where the entire tuition service system is tested as a whole. It ensures that all modules—such as user registration, course management, session scheduling, payments, and communication—work together seamlessly.

This testing evaluates both functional and non-functional aspects of the system. Functional testing ensures that features like course enrollment, payment processing, and chat functions operate correctly. Non-functional testing assesses performance, security, usability, and scalability. Since the tuition service involves multiple actors (students, tutors, and admins), system testing verifies whether role-based access and permissions are correctly implemented..

* **Functional Testing Techniques :**

* **PDF Security Tool** :-

**Test Case FT-01:- PDF Encryption**  
Objective: Verify password protection functionality.

Steps:

1. Upload PDF
2. Set password (e.g., "Secure@123")
3. Download encrypted file

Verification: Encrypted PDF requires password to open

**Test Case FT-02: PDF Decryption**  
Objective: Test password removal

Steps:

1. Upload password-protected PDF
2. Enter correct password
3. Download decrypted file

Verification: Decrypted PDF opens without password

* **PDF Split Tool**

**Test Case FT-03: Page Range Extraction**  
Objective: Verify splitting by page range

Steps:

1. Upload multi-page PDF
2. Select pages 3-5
3. Download extracted pages

Verification: Output contains only specified pages

**Test Case FT-04**: Single Page Extraction  
Objective: Test single page extraction

Steps:

1. Upload PDF
2. Select page 7
3. Download single page

Verification: Output is single-page PDF

* **PDF Merge Tool**

**Test Case FT-05**: Multiple PDF Merging  
Objective: Test combining 3+ PDFs

Steps:

1. Upload 3 PDF files
2. Set merge order
3. Download combined PDF

Verification: Output contains all pages in correct order

* **PDF Conversion Tools**

**Test Case FT-06: PDF to Word**  
Objective: Verify DOCX conversion

Steps:

1. Upload PDF
2. Convert to Word
3. Download DOCX file

Verification**:** Text content preserved in Word

**Test Case FT-07: JPG to PDF**  
Objective: Test image conversion

Steps:

1. Upload JPG image
2. Convert to PDF
3. Download PDF

Verification: PDF contains image at original quality

* **Non-Functional Testing Techniques :**
* **Performance Testing**

**Test Case NFT-01:** Large File Handling  
Objective: Verify 50MB PDF processing

Steps:

1. Upload 50MB PDF
2. Perform split operation

Metric: Completion within 30 seconds

**Test Case NFT-02: Concurrent Users**  
Objective: Test with 50 simultaneous users

Steps: Simulate 50 conversion requests  
Metric: All complete within 2 minutes

* **Security Testing**

**Test Case NFT-03: Malicious File Upload**  
Objective: Test virus scanning

Steps: Upload PDF with embedded script  
Verification: File rejected with warning

* **Usability Testing**

**Test Case NFT-04: Mobile Responsiveness**  
Objective: Verify mobile interface

Steps: Test on 5" smartphone  
Verification: All controls accessible

* **Test Results Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Category** | **Test Cases** | **Passed** | **Failed** |
| Security Tools | 6 | 6 | 0 |
| Split/Merge | 5 | 5 | 0 |
| Conversion Tools | 8 | 8 | 0 |
| Performance | 4 | 4 | 0 |
| Security | 3 | 3 | 0 |
| Usability | 3 | 3 | 0 |
| **Total** | **29** | **29** | **0** |

It also checks how the system handles high user loads, ensuring a smooth experience even during peak times. Security is another critical aspect, as the system deals with sensitive user data and financial transactions.

Ultimately, system testing ensures that the tuition service platform is reliable, secure, and user-friendly, providing a seamless learning experience for students and tutors.

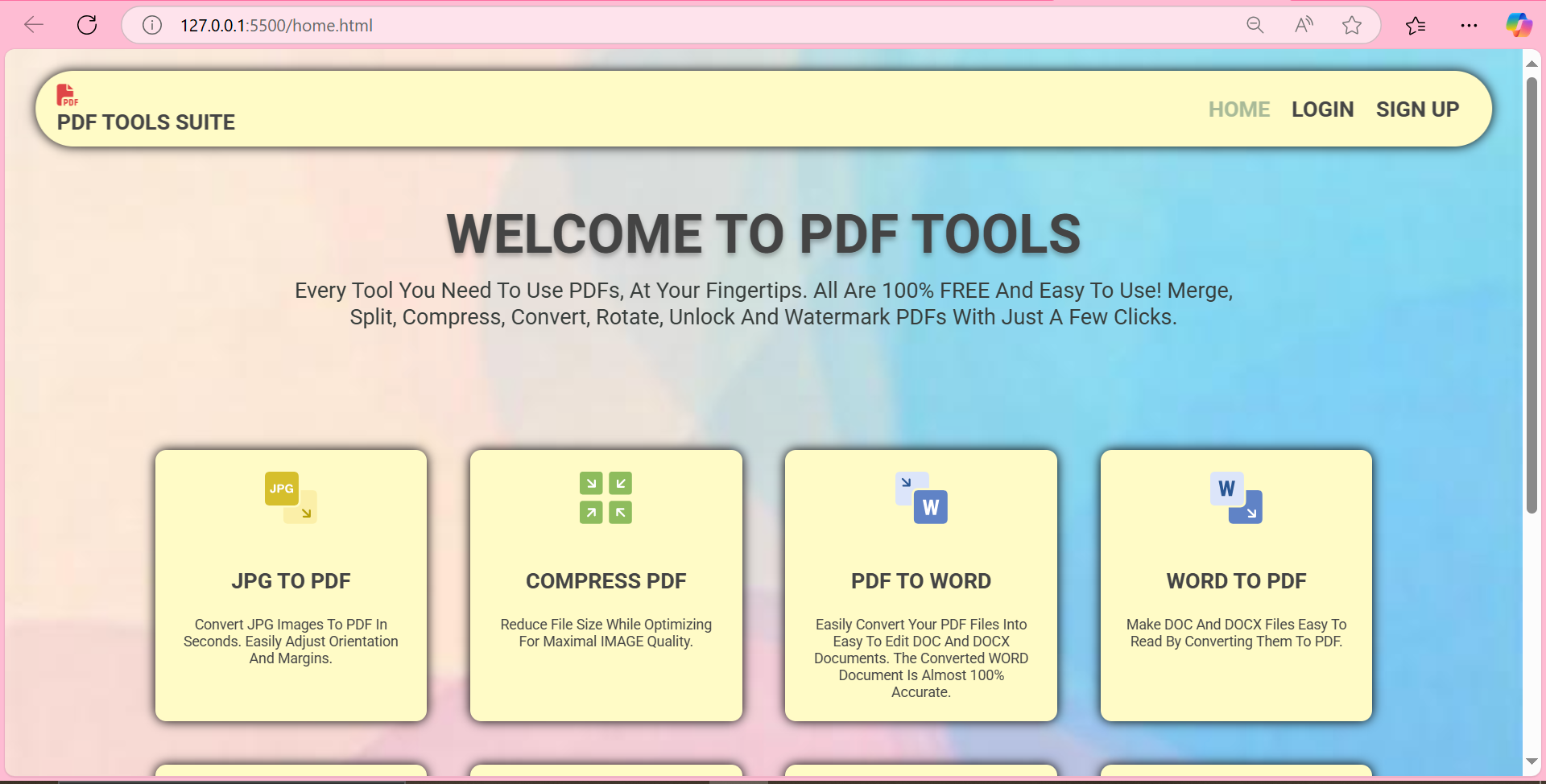
**Database Design**

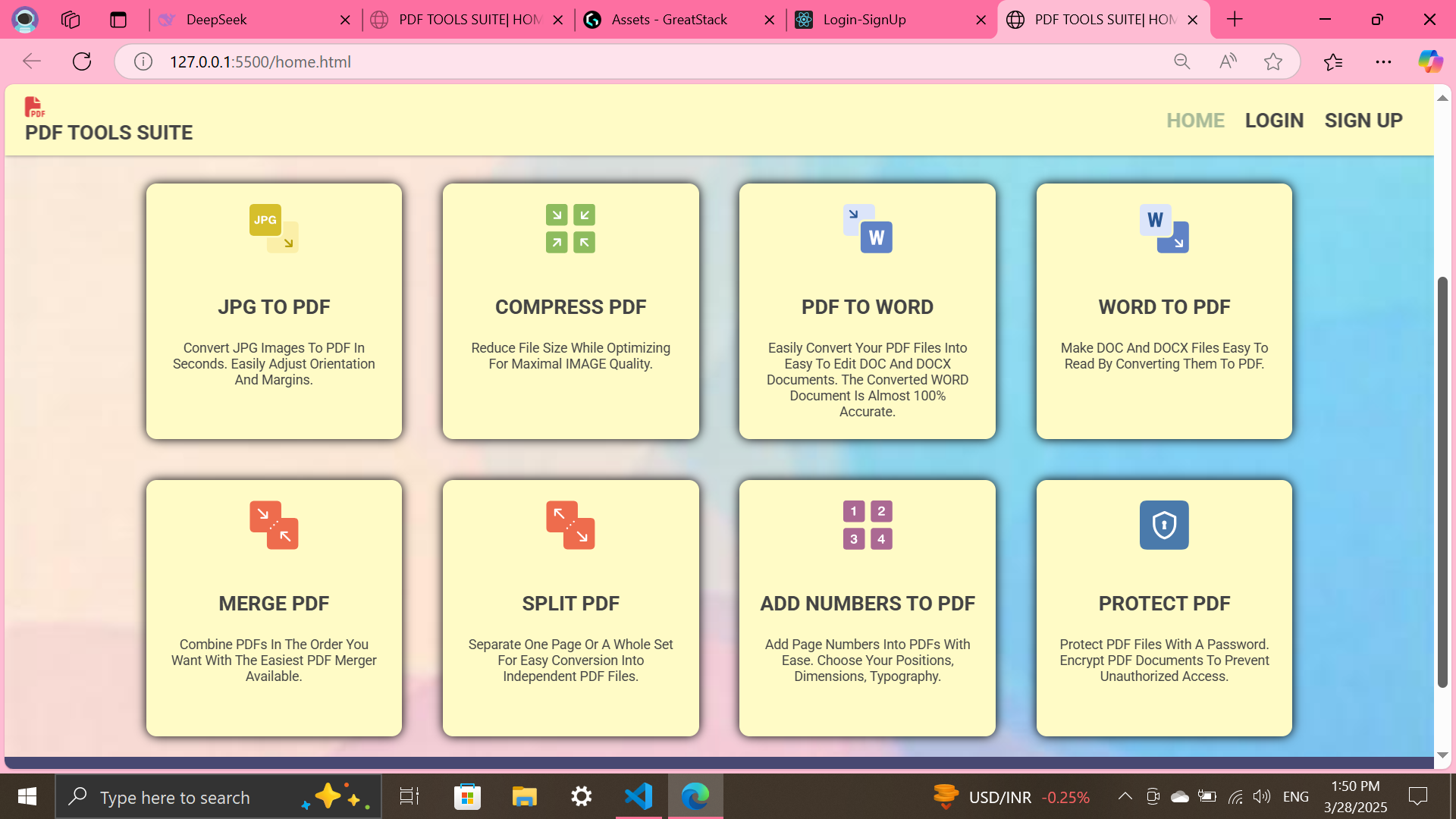
The database design of a project ensures efficient data management, seamless interactions between students and tutors, and secure transactions. It involves structuring data entities, relationships, and constraints to optimize performance and maintain data integrity.

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make database access easy, quick, inexpensive and flexible for the user. Relationships are established between the data items and unnecessary data items are removed. Normalization is done to get an internal consistency of data and to have minimum redundancy and maximum stability. This ensures minimizing data storage required, minimizing chances of data inconsistencies and optimizing for updates. The MS Access database has been chosen for developing the relevant database.

# Output Screen of Project

**HOME SCREEN:-**

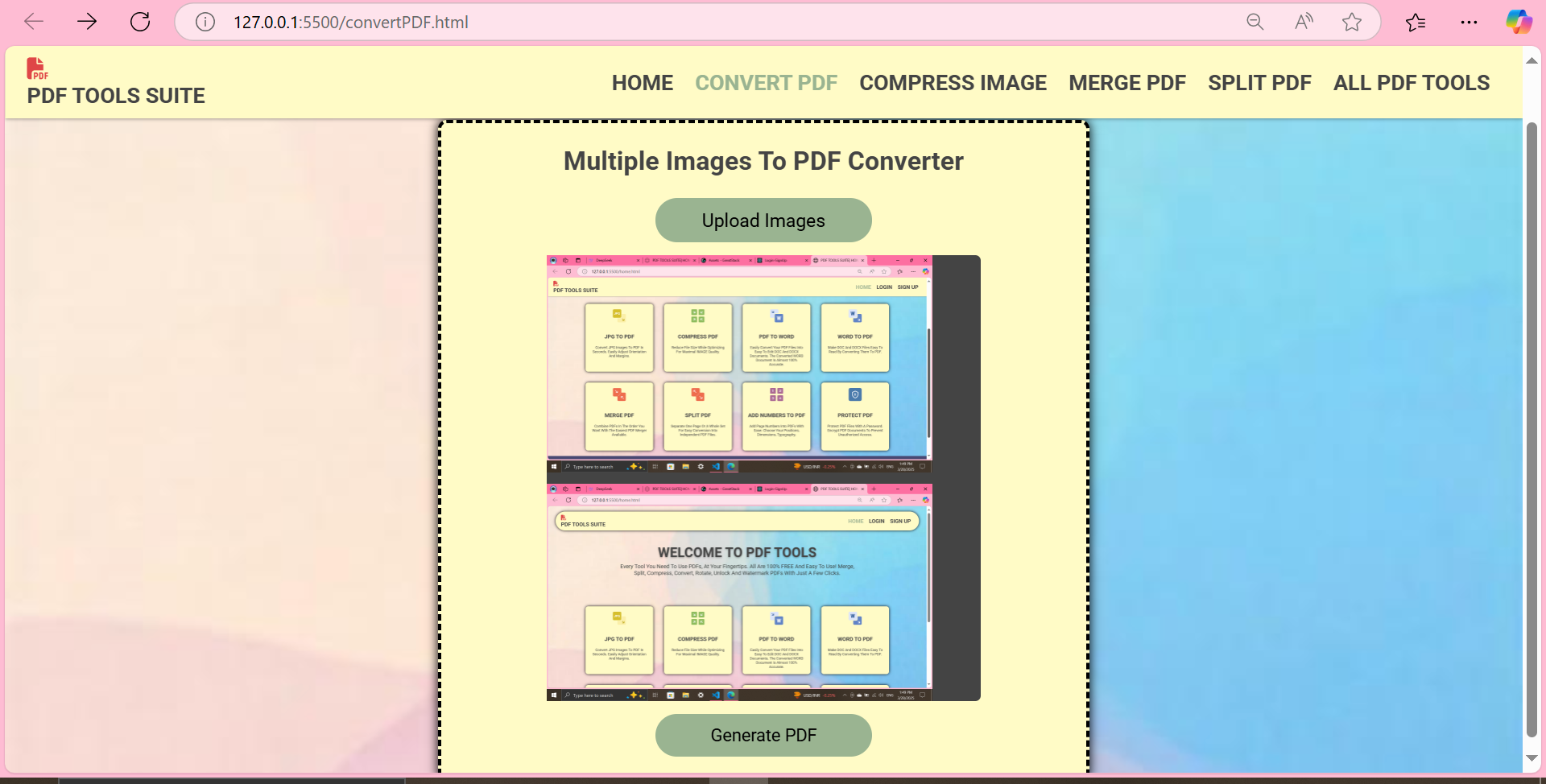
****

****

**2) JPG TO PDF CONVERTER:-**

****

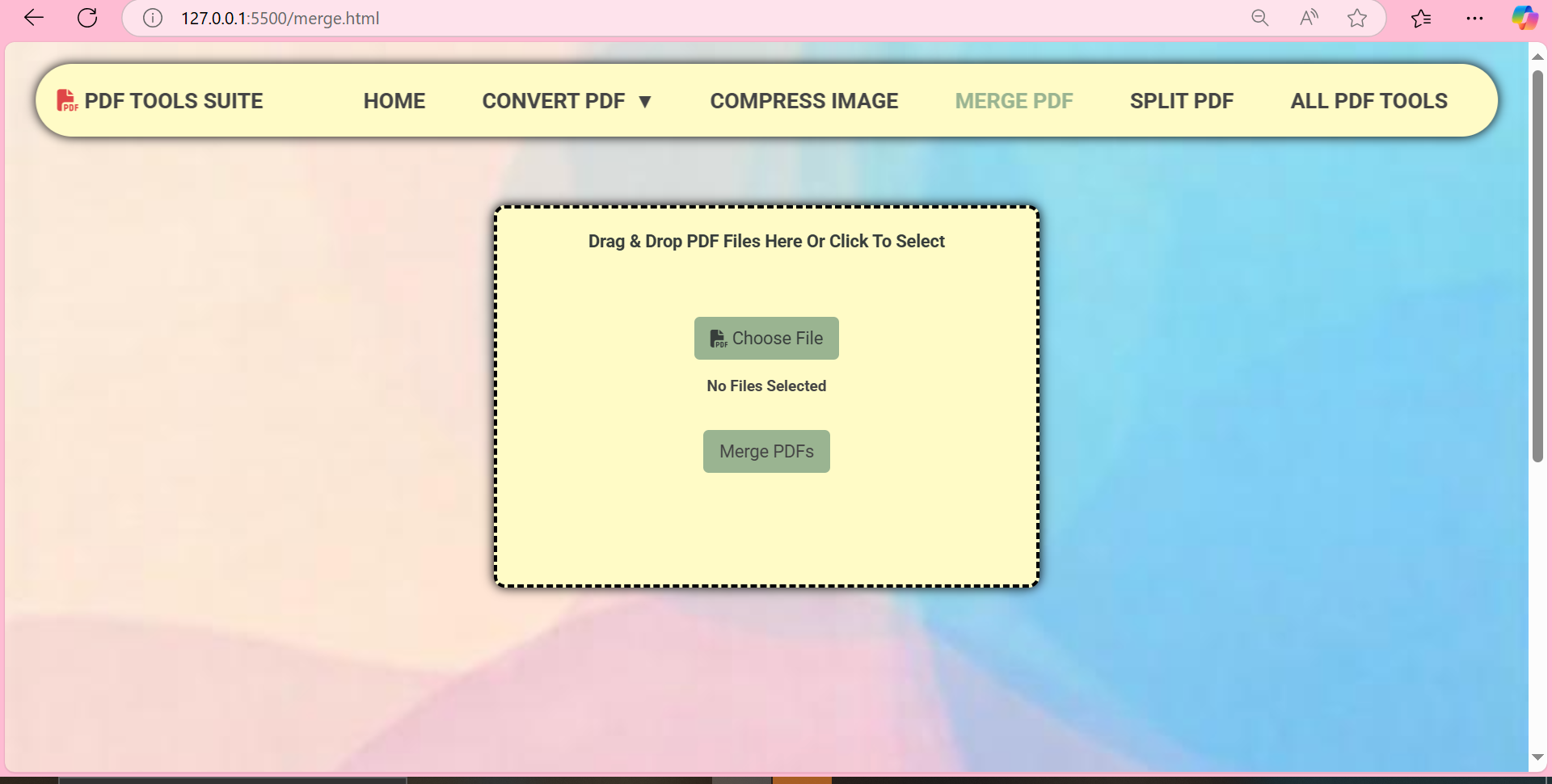
AFTER SELECTING IMAGES:-



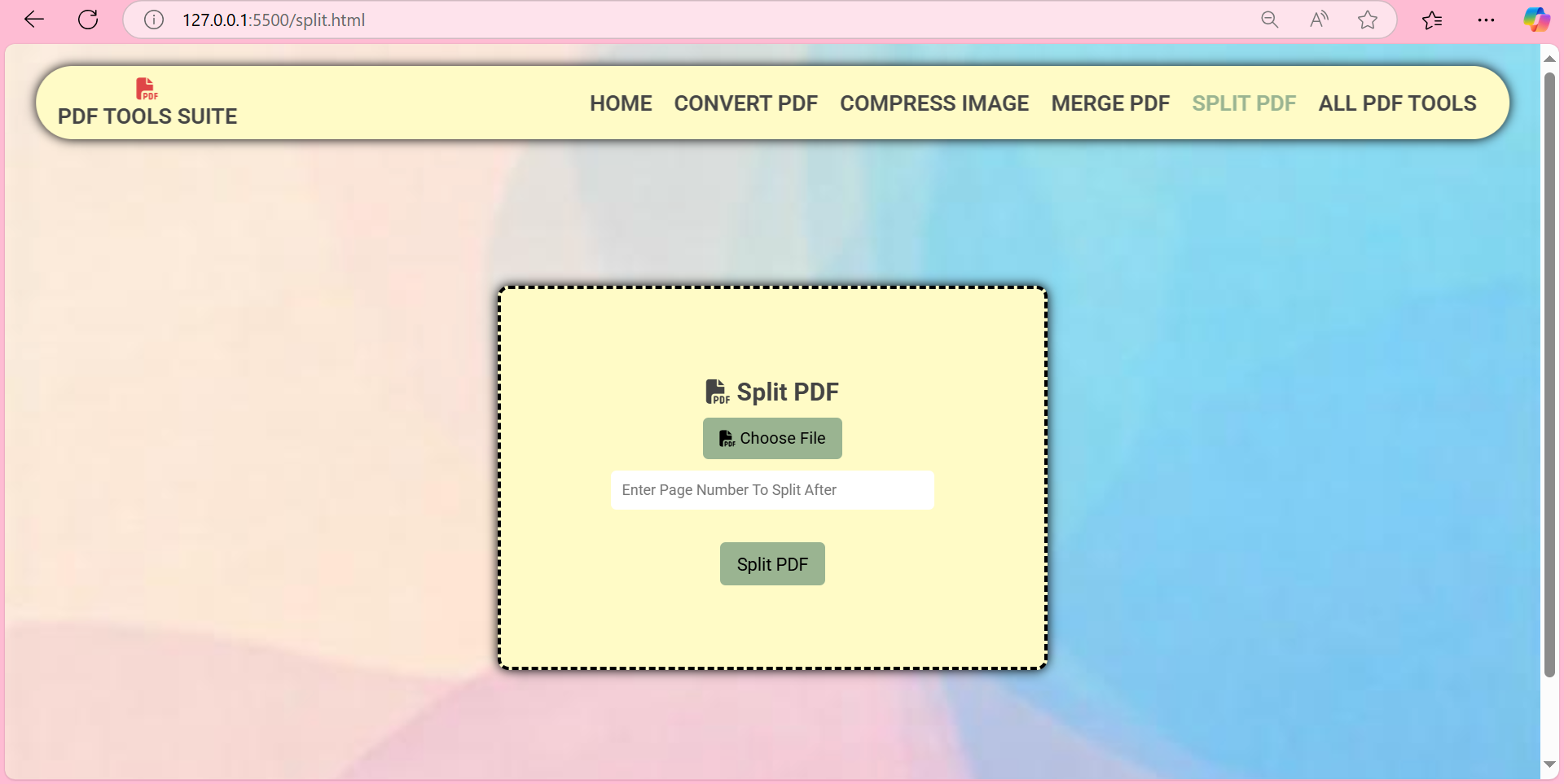
**WORD TO PDF:-**



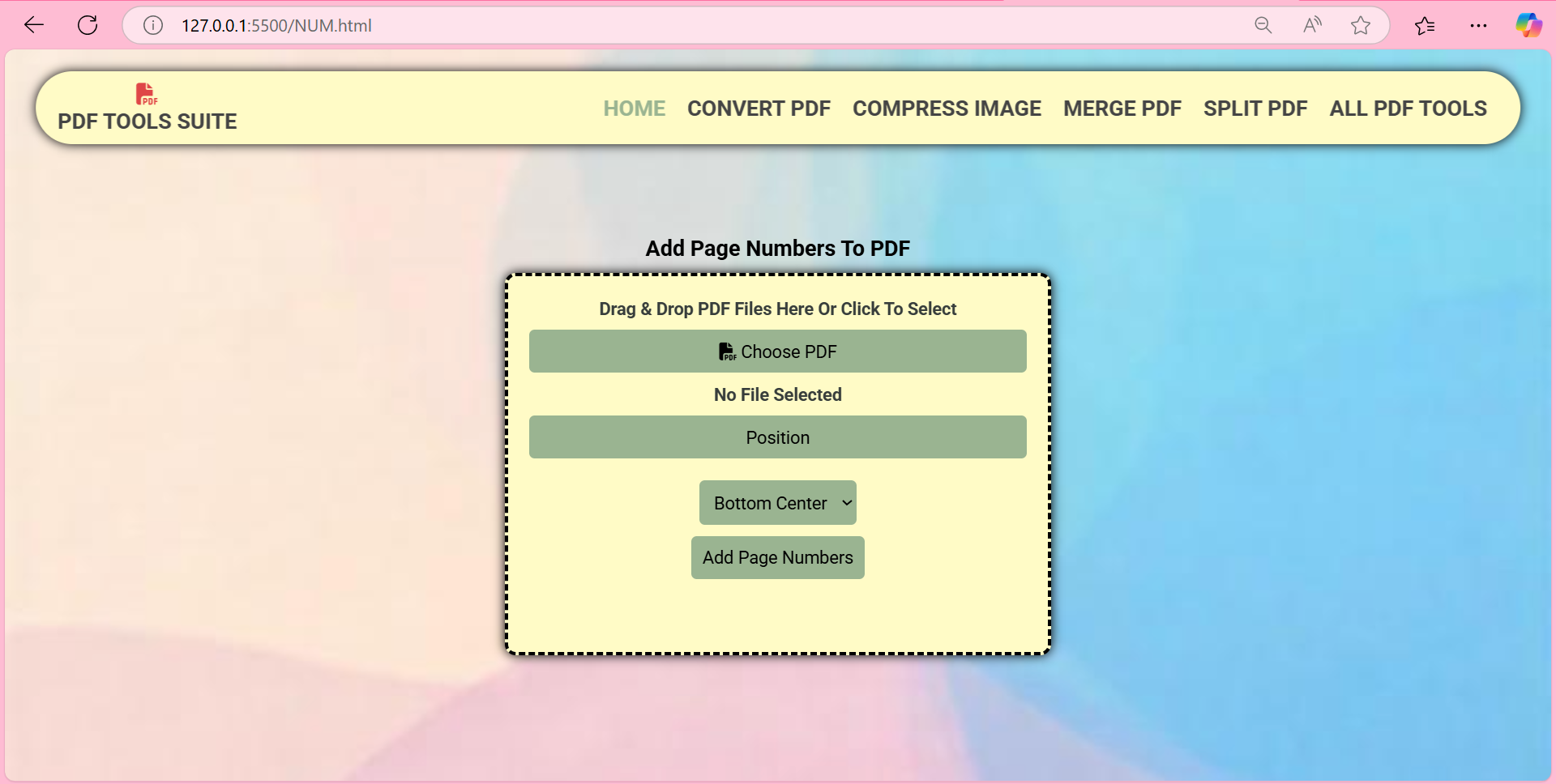
**MERGE PDF:-**

****

**SPLIT PDF:-**



**ADD NUMBERS TO THE PDF:-**

****

# Conclusion

The **PDF Tools Suite** is a **powerful, user-friendly web application** designed to simplify the management and manipulation of PDF documents. Built with **Flask (Python) for the backend** and **HTML/CSS/JavaScript for the frontend**, this tool offers a seamless experience for handling PDF files efficiently.

With a **wide range of features**, users can **encrypt, decrypt, split, merge, compress, and convert** PDFs (to/from Word and JPG) while also adding page numbers with ease.

## ****Key Achievements****

### ****Feature-Rich & Fully Functional****

* Effortlessly **encrypt, split, merge, and convert** PDFs.
* **AES-256 encryption** ensures strong document security.
* Smart **error handling** prevents processing failures due to invalid inputs.

### ****Optimized for Performance & Scalability****

* Efficiently processes **large PDFs (50MB+)** without slowdowns.
* Handles **multiple users simultaneously** without compromising speed.

### ****Designed for an Exceptional User Experience****

* **Intuitive and responsive** interface (works smoothly on mobile, tablet, and desktop).
* **Clear error messages & real-time feedback** for seamless processing.

### ****Security & Reliability at its Core****

* Protects against **malicious file uploads** and unauthorized access.
* **Server-side validation** ensures data integrity and reliability.

## ****Future Enhancements****

To further enhance the **PDF Tools Suite**, the following upgrades are planned:

* **OCR Integration** – Extract text from scanned PDFs.
* **Batch Processing** – Perform multiple file operations at once.
* **Cloud Storage Support** – Seamless integration with Google Drive & Dropbox.

## ****Final Thoughts****

The **PDF Tools Suite** successfully delivers on its goals, providing a **secure, efficient, and easy-to-use** PDF management solution. Whether you’re a **student, professional, or business user**, this tool is designed to make PDF handling effortless.

### ****Developed By:**** SHRAVANI G. SAWANT

### ****Date:****

### ****Version:**** 1.0.0

# Bibliography

**1. Technical References**  
Flask Documentation (2023). Flask Web Framework. Available at: <https://flask.palletsprojects.com/>

PyPDF2 Documentation (2023). Python PDF Library. Available at: <https://pypdf2.readthedocs.io/>

ReportLab Documentation (2023). PDF Generation in Python. Available at: <https://www.reportlab.com/docs/reportlab-userguide.pdf>

PikePDF Documentation (2023). PDF Encryption/Decryption. Available at: <https://pikepdf.readthedocs.io/>

**2. Security & Cryptography**  
NIST (2023). Advanced Encryption Standard (AES). Available at: <https://csrc.nist.gov/publications/detail/fips/197/final>

OWASP (2023). Secure File Upload Guidelines. Available at: <https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload>

**3. Web Development & UI/UX**  
MDN Web Docs (2023). HTML, CSS, and JavaScript References. Available at: <https://developer.mozilla.org/>

W3Schools (2023). Responsive Web Design. Available at: <https://www.w3schools.com/css/css_rwd_intro.asp>

**4. Performance Optimization**  
Python Software Foundation (2023). Python Performance Tips. Available at: <https://wiki.python.org/moin/PythonSpeed/PerformanceTips>

Google Developers (2023). Web Performance Best Practices. Available at: <https://web.dev/learn/#performance>

**5. Testing & Quality Assurance**  
Python Testing (2023). unittest Framework. Available at: <https://docs.python.org/3/library/unittest.html>

Selenium (2023). Automated Web Testing. Available at: <https://www.selenium.dev/>

**Additional Resources**  
Stack Overflow (2023). Community Support for Coding Issues. Available at: <https://stackoverflow.com/>

GitHub Repositories (2023). Open-Source PDF Tools. Available at: <https://github.com/topics/pdf-tools>

**Appendix**

**Appendix A: Sample Test Files**

**B1. Test PDFs Used for Validation**

* sample\_encrypted.pdf (Password: Secure123)
* sample\_multipage.pdf (10 pages)
* sample\_image.jpg (For JPG-to-PDF testing)

**B2. Sample Output Files**

* encrypted.pdf
* splitpdf\_3-5.pdf
* merged.pdf

**Appendix B: Error Logs & Debugging**

**C1. Common Error Messages**

| **Error Code** | **Description** | **Solution** |
| --- | --- | --- |
| ERR-401 | Invalid file format | Upload only PDF/JPG files |
| ERR-402 | Password too weak | Use ≥8 characters with symbols |
| ERR-403 | File size exceeded | Limit: 50MB per file |

**C2. Debugging Steps**

* Check server logs for detailed error info.
* Verify file integrity (corrupted files may fail).
* Test with smaller files to isolate performance issues.

**Appendix C: User Feedback & Improvements**

**D1. Beta Tester Comments**

* *"The merge tool works perfectly!"* – User A
* *"Add a progress bar for large files."* – User B

**D2. Planned Feature Updates**

* Dark mode for better readability
* Drag-and-drop file upload support
* Bulk processing for multiple files

**Appendix D: Code Snippets (Key Functions)**

**E1. PDF Encryption (Python - Flask)**

python

CopyEdit

from pikepdf import Pdf, Encryption

def encrypt\_pdf(input\_path, password):

pdf = Pdf.open(input\_path)

pdf.save("encrypted.pdf", encryption=Encryption(user=password, owner=password, aes=True))

**E2. PDF Splitting (Python – PyPDF2)**

python

CopyEdit

from PyPDF2 import PdfReader, PdfWriter

def split\_pdf(input\_path, start\_page, end\_page):

reader = PdfReader(input\_path)

writer = PdfWriter()

for page\_num in range(start\_page-1, end\_page):

writer.add\_page(reader.pages[page\_num])

with open("split.pdf", "wb") as f:

writer.write(f)

**Appendix E: System Requirements**

**F1. Minimum Requirements**

* **OS:** Windows 10/macOS 10.15+/Linux
* **RAM:** 4GB
* **Storage:** 200MB free space

**F2. Supported Browsers**

* Chrome (v90+)
* Firefox (v88+)
* Edge (v91+)