Table of Contents

1. Overview
2. System Requirements
3. Installation & Setup
4. Core Features
5. User Interface Guide
6. API Reference
7. File Management
8. Data Analysis Features
9. Visualization Tools
10. Technical Implementation
11. Troubleshooting
12. Future Enhancements

Overview

The Data Analytics Dashboard **“PROJECT DA”** is a web-based application built with Flask that provides comprehensive data analysis and visualization capabilities. This version focuses on core functionality for data upload, analysis, and visualization without user authentication systems.

Key Features

* File Upload: Support for CSV and Excel files
* Data Viewing: Interactive data tables with pagination
* Statistical Analysis: Comprehensive descriptive statistics
* Data Visualization: Multiple chart types (histograms, bar charts, pie charts, line plots, scatter plots, box plots)
* Notes Management: Upload and manage analysis notes and documents
* -Responsive Design: Mobile-friendly interface

Technology Stack

* Backend: Flask (Python)
* Frontend: Bootstrap 5, HTML5, CSS3, JavaScript
* Data Processing: Pandas, NumPy
* Visualization: Matplotlib, Seaborn
* File Handling: Werkzeug

System Requirements

Minimum Requirements

* Python 3.8 or higher
* 4GB RAM
* 1GB available disk space
* Modern web browser (Chrome, Firefox, Safari, Edge)

Recommended Requirements

* Python 3.9+
* 8GB RAM
* 5GB available disk space
* High-speed internet connection

Installation & Setup

1. Environment Setup

# Create virtual environment

python -m venv analytics\_dashboard

source analytics\_dashboard/bin/activate

# On Windows: analytics\_dashboard\Scripts\activate

# Install dependencies

pip install -r requirements.txt

1. Dependencies

Flask==2.3.3

pandas==2.1.1

numpy==1.24.3

matplotlib==3.7.2

seaborn==0.12.2

openpyxl==3.1.2

xlrd==2.0.1

Werkzeug==2.3.7

1. Application Configuration

# Configuration settings in app.py

UPLOAD\_FOLDER = 'uploads'

NOTES\_FOLDER = 'notes'

ALLOWED\_EXTENSIONS = {'csv', 'xlsx', 'xls', 'txt', 'pdf', 'docx'}

MAX\_CONTENT\_LENGTH = 16 \* 1024 \* 1024 # 16MB max file size

1. Running the Application

python app.py

Core Features

File Upload System

The application supports secure file upload with validation and processing capabilities.

Supported File Types

* Data Files: CSV (.csv), Excel (.xlsx, .xls)
* Notes Files: TXT, PDF, DOCX, CSV, Excel

Upload Process

def upload\_file():

if 'file' not in request.files:

flash('No file selected')

return redirect(request.url)

file = request.files['file']

if file and allowed\_data\_file(file.filename):

filename = secure\_filename(file.filename)

filepath = os.path.join(app.config['UPLOAD\_FOLDER'], filename)

file.save(filepath)

flash(f'File {filename} uploaded successfully!')

return redirect(url\_for('view\_data', filename=filename))

Security Measures

* File type validation using `allowed\_file()` function
* Filename sanitization with `secure\_filename()`
* File size limits (16MB maximum)
* Directory traversal protection

User Interface Guide

Navigation Structure

The application features a responsive navigation bar with the following sections:

<nav class="navbar navbar-expand-lg navbar-dark bg-primary">

<div class="container">

<a class="navbar-brand" href="/">

<i class="fas fa-chart-line me-2"></i>Analytics Dashboard

</a>

<ul class="navbar-nav ms-auto">

<li class="nav-item"><a href="/">Home</a></li>

<li class="nav-item"><a href="/upload">Upload Data</a></li>

<li class="nav-item"><a href="/datasets">Datasets</a></li>

<li class="nav-item"><a href="/notes">Notes</a></li>

</ul>

</div>

</nav>

Home Page Features

The landing page provides:

* Hero section with call-to-action buttons
* Feature overview cards
* Quick access to main functionalities

Upload Interface

Drag & Drop Functionality

uploadArea.addEventListener("drop", (e) => {

e.preventDefault()

const files = e.dataTransfer.files

if (files.length > 0) {

fileInput.files = files

updateFileLabel(files[0].name)

}

})