AI Text Summarization Platform

This is a comprehensive text summarization platform built with Python Flask. It allows users to summarize content from various sources including PDF, DOCX, TXT files, URLs, YouTube videos, and manual text input.

Features

- Multi-format document summarization (PDF, DOCX, TXT)
- URL and YouTube video summarization
- Customizable compression ratios
- Multiple output formats (bullet points, paragraphs, plain text)
- Export options (TXT, DOCX, PDF)
- User authentication system
- Performance metrics (ROUGE score, compression ratio)
- AI chatbot assistant

Project Workflow

1. User Authentication Flow
- User registration with email verification
- Secure login with password hashing
- Session management
- Password reset functionality
- User profile management

2. Document Processing Flow
- File upload handling (PDF, DOCX, TXT)
- Text extraction from documents
- URL content scraping
- YouTube transcript extraction
- Manual text input processing

3. Summarization Process

- Text preprocessing
- Language detection
- Sentence tokenization
- Key sentence extraction
- Summary generation
- Compression ratio calculation
- ROUGE score computation

4. Output Generation

- Multiple format support
  - Plain text
  - Bullet points
  - Paragraphs
- Export options
  - TXT file generation
  - DOCX document creation
  - PDF conversion
- Font and styling customization

5. AI Assistant Integration

- Chatbot interface
- Context-aware responses
- Feature explanations
- Usage guidance
- Error handling

6. Data Management

- Summary history tracking
- User data storage

- File management
- Database operations
- Cache management

7. Security Measures
- Input validation
- XSS protection
- CSRF protection
- Rate limiting
- Secure file handling

8. Performance Optimization
- Asynchronous processing
- Caching mechanisms
- Resource optimization
- Load balancing
- Error logging

How to Run the Project

Prerequisites
- Python 3.6 or higher
- pip (Python package installer)
- Git
- Virtual environment (recommended)
- Modern web browser
- Internet connection

Step-by-Step Setup

1. **Clone the Repository**

```bash
git clone <repository-url>
cd text-summarization-platform
```

2. **Set Up Virtual Environment**

```bash
# Windows
python -m venv venv
venv\Scripts\activate

# Linux/MacOS
python3 -m venv venv
source venv/bin/activate
```

3. **Install Dependencies**

```bash
pip install -r requirements.txt
```

4. **Configure Environment Variables**

- Create a `.env` file in the root directory
- Add the following variables:

```
SECRET_KEY=your_secret_key_here
FLASK_APP=app.py
FLASK_ENV=development
```

5. **Initialize Database**

```bash
python init_db.py
```

6. **Download NLTK Resources**

```bash
python -c "import nltk; nltk.download('punkt'); nltk.download('stopwords')"
```

7. **Run the Application**

```bash
# Development mode
flask run

# Production mode
python app.py
```

8. **Access the Application**

- Open your web browser
- Navigate to `http://127.0.0.1:5000`

Running in Production

1. **Set Production Environment**

```bash
export FLASK_ENV=production  # Linux/MacOS
set FLASK_ENV=production     # Windows
```

2. **Configure Gunicorn (Linux/MacOS)**

```bash
pip install gunicorn
gunicorn -w 4 -b 127.0.0.1:5000 app:app
```

3. **Set Up Nginx (Optional)**
   - Install Nginx
   - Configure reverse proxy
   - Set up SSL certificates

Troubleshooting

1. **Database Issues**
   - Delete `instance/summarization.db`
   - Run `python init_db.py` again

2. **Module Not Found Errors**
   - Ensure virtual environment is activated
   - Run `pip install -r requirements.txt` again

3. **NLTK Resource Errors**
   - Run NLTK download commands manually
   - Check internet connection

4. **Port Already in Use**
   - Change port in `app.py`
   - Kill process using the port

Development Guidelines

1. **Code Style**

- Follow PEP 8 guidelines
- Use meaningful variable names
- Add comments for complex logic

2. **Testing**

```bash
# Run tests
python -m pytest

# Run with coverage
python -m pytest --cov=.
```

3. **Debugging**

- Use Flask debug mode
- Check logs in `logs/` directory
- Monitor error messages