Textify

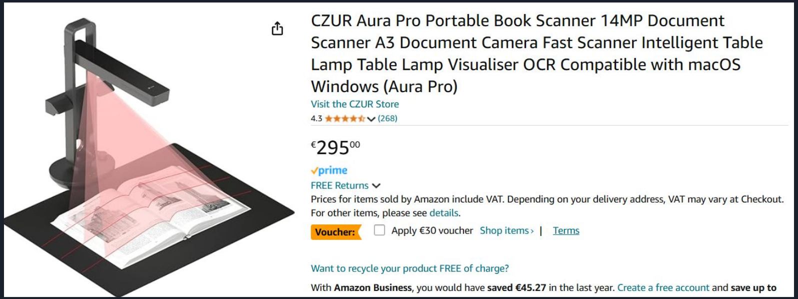
Aaron Sonntag, Christian Jaide, Joel Koch, Deepseek, Claude and ChatGPT

Systems and Software Engineering 1

Starting Point

Status Quo

- Book scanning is expensive
- Book scanning is not accessible
- Mostly used in commercial settings
- Mostly need more than one application
- Not really privacy oriented
- Dependent on the provider, when using an app


CZUR Aura Pro Portable Book Scanner 14MP Document Scanner A3 Document Camera Fast Scanner Intelligent Table Lamp Table Lamp Visualiser OCR Compatible with macOS Windows (Aura Pro)

Visit the CZUR Store

4.3 ★★★★☆ (268)

€295.00

✓prime

FREE Returns

Prices for items sold by Amazon include VAT. Depending on your delivery address, VAT may vary at Checkout. For other items, please see details.

Voucher: ☐ Apply €30 voucher Shop items › | Terms

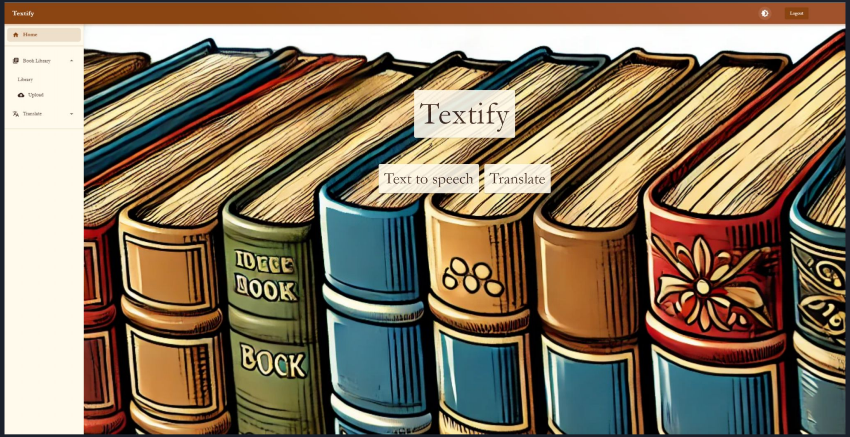
Want to recycle your product FREE of charge?

With **Amazon Business**, you would have **saved €45.27** in the last year. Create a free account and **save up to**

14MP for 295€ a phone for ≈270–300€ has between 50-64MP

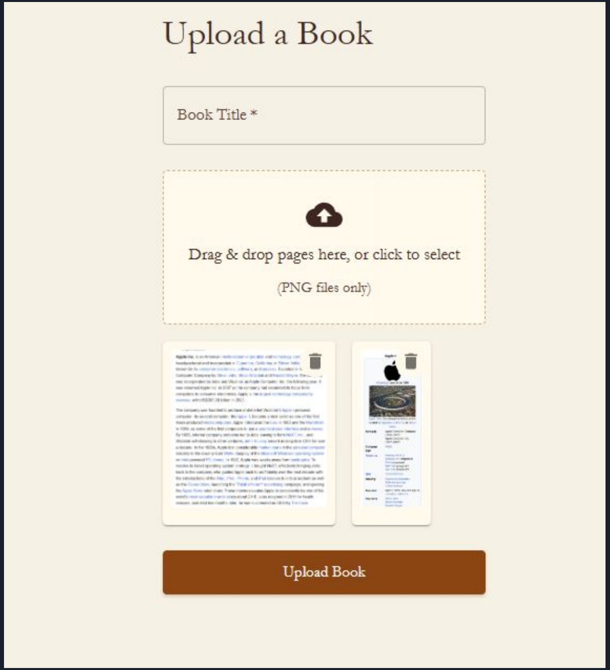
Idea

- Web-based application
- Can be used on most devices
- Multiple input options
- Scans images taken in advance
- Multiple export options
- Extra features like translation, TTS,
- Multiple output options (PDF, DOCX, HTML)



Expected Benefits

- Cheap to use
- Easy to setup and easy to use
- No knowledge needed
- Easy to access via Browser
- Opensource
  - You can set it up in your own network
  - Data is private
  - No cloud connection required
  - Customizable by using different ai models



Outline

1. Requirements
2. Working in Teams
3. Use Case
4. System Design and Architecture
5. Documentation and Tools
6. Pain Points
7. Live Demo

Requirements

Functional Requirements

Text Translation

- Requirement:
  - The system shall accept input text in a source language and translate it into a target language.
- Acceptance Criteria:
  - Users shall enter text and select source/target languages.
  - The system shall provide a translation within 2 seconds for text under 500 words.

Text-to-Speech (TTS)

- Requirement:
  - The system shall convert text into speech and allow users to download audio files in standard formats.
- Acceptance Criteria:
  - Users shall adjust speed (slow, normal, fast) and volume via a slider.

Speech-to-Text (STT)

- Requirement:
  - The system shall transcribe spoken input into text with 90% accuracy in moderate noisy environments.
- Acceptance Criteria:
  - Users shall upload audio files or provide live input.
  - Transcriptions shall include proper punctuation and capitalization.

Functional Requirements

Optical Character Recognition (OCR)

- Requirement:
  - The system shall extract text from uploaded images with 95% accuracy for standard fonts.
- Acceptance Criteria:
  - Users shall upload images in formats such as PNG and JPEG.

User Interface

- Requirement:
  - The application shall have a responsive design for desktops and mobile devices.
- Acceptance Criteria:
  - The interface shall allow users to easily access translation, OCR, TTS, and STT features.
  - Key features shall load within 1 second on standard devices.

Exports

- Requirement:
  - The application shall provide an export function for the converted text.
- Acceptance Criteria:
  - The interface shall allow users to easily export into various formats.

Non-Functional Requirements

Scalability

- Requirement:
    - The system shall support multiple concurrent users with a response time under 3 seconds.

Security

- Requirement:
    - All data transmissions shall be encrypted using SSL/TLS.
    - User data shall not be stored in unencrypted formats.

Privacy

- Requirement:
    - User data shall be encrypted.
    - User data shall only be used for training models with explicit consent.

Non-Functional Requirements

Maintainability

- Requirement:
  - The system shall have a modular architecture with clear interfaces, minimal dependencies, and centralized management for code, dependencies, and configuration to ensure maintainability and ease of updates.
- Acceptance Criteria:
  - Each module shall have clear interfaces and minimal dependencies.
  - Code shall include inline comments and detailed documentation.
  - Code shall be managed via a version-control-system.
  - Dependencies and third party integrations shall be managed via central Dependency management.
  - Deployment and Configuration of the application shall be managed via a central file.

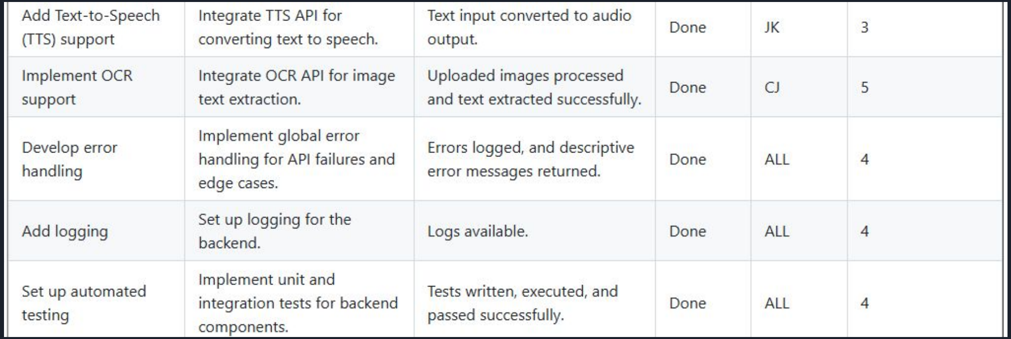
Reliability

- Requirement:
  - The system shall provide error detection, logging with timestamps and error codes, and clear error messages, supporting debugging and automated testing where feasible.
- Acceptance Criteria:
  - Logs shall include timestamps and error codes.
  - Users shall receive clear, actionable error messages.
  - The application shall have automated tests if possible.

Working in Teams

Organizational Requirements/Working in team

- Weekly meetups
- Making decisions together
- Pair Programming
- Independent working
- Being highly flexible
- Kanban board for tasks



Add Text-to-Speech (TTS) support	Integrate TTS API for converting text to speech.	Text input converted to audio output.	Done	JK	3
Implement OCR support	Integrate OCR API for image text extraction.	Uploaded images processed and text extracted successfully.	Done	CJ	5
Develop error handling	Implement global error handling for API failures and edge cases.	Errors logged, and descriptive error messages returned.	Done	ALL	4
Add logging	Set up logging for the backend.	Logs available.	Done	ALL	4
Set up automated testing	Implement unit and integration tests for backend components.	Tests written, executed, and passed successfully.	Done	ALL	4

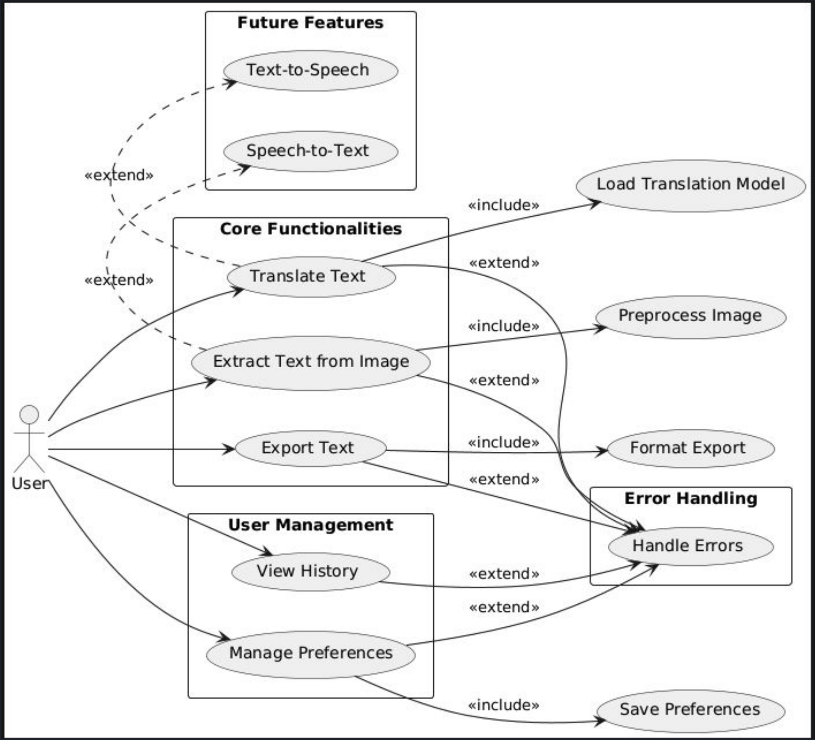
Person in Charge	Responsibilities
Christian	OCR, Frontend, REST-API, DB, Security, Bugs, Testing with GPU, Presentation, Documents and a bit Text processing
Joel	Translation, TTS, STT, REST-API, DB, Docker and Deployment, Security, Bugs, Presentation, Documents, Organisational Matters
Aaron	Presentation and Documents

Tools for Teamwork

- Git and Github for collaboration
- Discord and emails for communication
- Google Docs for working together on documents

Use Case

Use Case


Future Features
Text-to-Speech
Speech-to-Text
«extend»
«extend»
Core Functionalities
Translate Text
Extract Text from Image
Export Text
User
User Management
View History
Manage Preferences
«include»
Load Translation Model
«extend»
«include»
Preprocess Image
«extend»
«include»
Format Export
«extend»
Error Handling
Handle Errors
«extend»
«extend»
«include»
Save Preferences

Use Case

Use Case: Text Translation

- Actors: User
- Description: The user shall input text and select source/target languages for translation.
- Preconditions: The user is on the translation page.
- Postconditions: The translated text is displayed to the user.

Use Case: OCR (Image-to-Text)

- Actors: User
- Description: The user shall upload an image for text extraction using OCR.
- Preconditions: The user is on the OCR page and has an image to upload.
- Postconditions: The extracted text is displayed.

Use Case

Use Case: Export Translation

- Actors: User
- Description: The user shall export translations into formats such as PDF or HTML.
- Preconditions: A translation is completed.
- Postconditions: The translated text is downloaded in the selected format

Use Case: STT

- Actors: User
- Description: The audio file is transcribed so that a text file containing the content of the audio file is output.
- Precondition: The user uploads an audio file.
- Postcondition: The user exports his text file or can translate it.

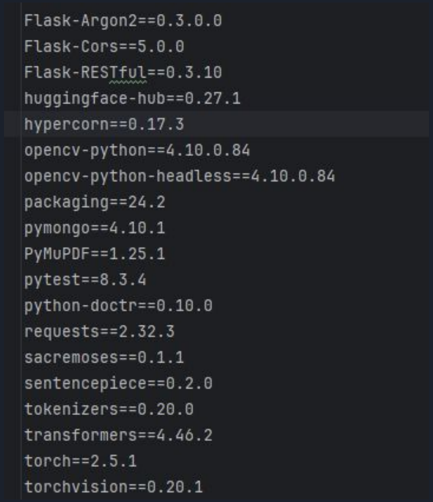
5.5 Use Case: TTS

- Actors: User
- Description: The system synthesises the speech in an audio file. This can be played on the website and downloaded as a file.
- Precondition: The user copies or writes a text into an input field.
- Postcondition: The user has an mp4 file.

System Design and Architecture

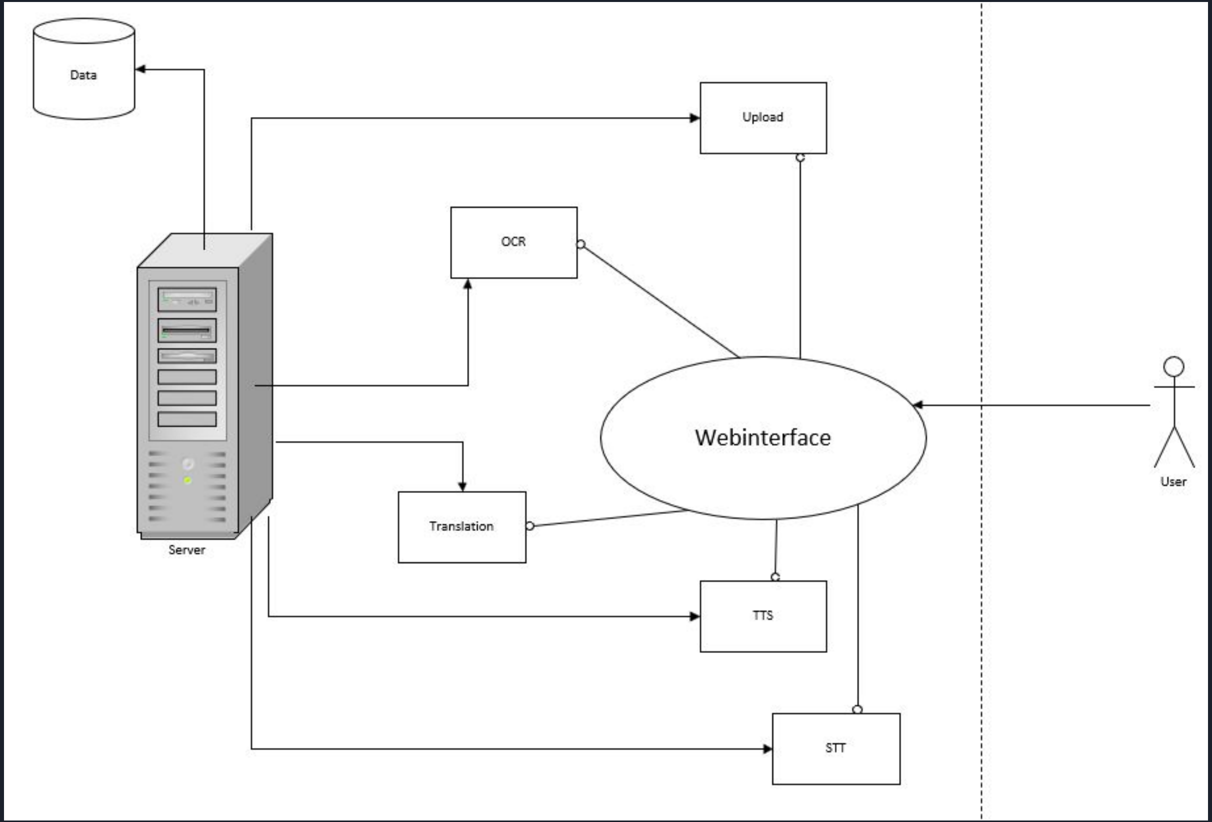
Architecture Overview

- Frontend: A web-based user interface for interaction.
  - Vite, React TS, MUI, Craft.js and Quill
- Backend: A server hosting the AI translation model and managing API communication (REST).
  - Python3.12, Flask and Hypercorn, Cryptodome, CUDA, Torch
  - OpusMT, XTTSv2, Whisper, docTR and some more small stuff
- Database: Storage of user data and translation history.
  - MongoDB and GridFS
- Deployment:
  - Docker with CUDA-Linux-Image
  - Docker-Compose



```
Flask-Argon2==0.3.0.0
Flask-Cors==5.0.0
Flask-RESTful==0.3.10
huggingface-hub==0.27.1
hypercorn==0.17.3
opencv-python==4.10.0.84
opencv-python-headless==4.10.0.84
packaging==24.2
pymongo==4.10.1
PyMuPDF==1.25.1
pytest==8.3.4
python-doctr==0.10.0
requests==2.32.3
sacremoses==0.1.1
sentencepiece==0.2.0
tokenizers==0.20.0
transformers==4.46.2
torch==2.5.1
torchvision==0.20.1
```

Sketch System Design



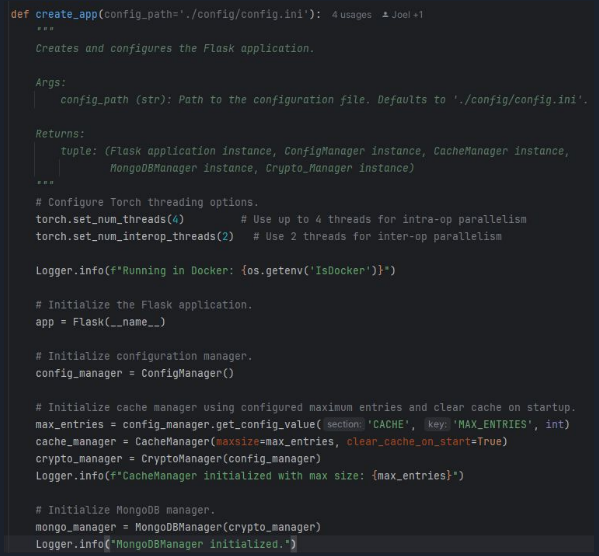
Documentation

Tool for Development

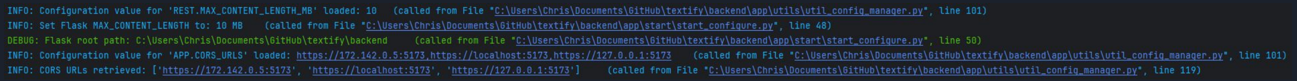
- Docker-Desktop
- Pycharm/VS-Code
- Postman
- Browser

Documenation

- ReadME.md
- Inline Comments and comments on import code lines
- Python Reference Documentation
- OpenAPI.yml for the REST-API
- System requirement documentation
- ADR for technological choices
- Kanban board for tasks



```
def create_app(config_path='./config/config.ini'):  4 usages  Joel +1
    """
    Creates and configures the Flask application.

    Args:
        config_path (str): Path to the configuration file. Defaults to './config/config.ini'.

    Returns:
        tuple: (Flask application instance, ConfigManager instance, CacheManager instance,
                MongoDBManager instance, Crypto_Manager instance)
    """
    # Configure Torch threading options.
    torch.set_num_threads(4)         # Use up to 4 threads for intra-op parallelism
    torch.set_num_interop_threads(2)   # Use 2 threads for inter-op parallelism

    Logger.info(f"Running in Docker: {os.getenv('IsDocker')}")

    # Initialize the Flask application.
    app = Flask(__name__)

    # Initialize configuration manager.
    config_manager = ConfigManager()

    # Initialize cache manager using configured maximum entries and clear cache on startup.
    max_entries = config_manager.get_config_value( section: 'CACHE',  key: 'MAX_ENTRIES', int)
    cache_manager = CacheManager(maxsize=max_entries, clear_cache_on_start=True)
    crypto_manager = CryptoManager(config_manager)
    Logger.info(f"CacheManager initialized with max size: {max_entries}")

    # Initialize MongoDB manager.
    mongo_manager = MongoDBManager(crypto_manager)
    Logger.info("MongoDBManager initialized.")
```



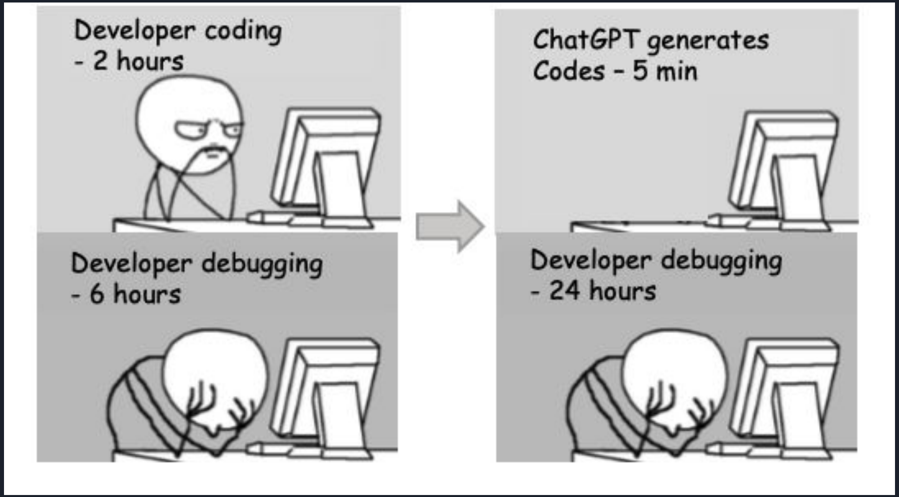
```
INFO: Configuration value for 'REST.MAX_CONTENT_LENGTH_MB' loaded: 10   (called from File "C:\Users\Chris\Documents\GitHub\textify\backend\app\utils\util_config_manager.py", line 101)
INFO: Set Flask MAX_CONTENT_LENGTH to: 10 MB    (called from File "C:\Users\Chris\Documents\GitHub\textify\backend\app\start\start_configure.py", line 48)
DEBUG: Flask root path: C:\Users\Chris\Documents\GitHub\textify\backend     (called from File "C:\Users\Chris\Documents\GitHub\textify\backend\app\start\start_configure.py", line 50)
INFO: Configuration value for 'APP.CORS_URLS' loaded: https://172.142.0.5:5173,https://localhost:5173,https://127.0.0.1:5173    (called from File "C:\Users\Chris\Documents\GitHub\textify\backend\app\utils\util_config_manager.py", line 101)
INFO: CORS URLs retrieved: ['https://172.142.0.5:5173', 'https://localhost:5173', 'https://127.0.0.1:5173']     (called from File "C:\Users\Chris\Documents\GitHub\textify\backend\app\utils\util_config_manager.py", line 119)
```

Test-Plan

- Manual Tests and trying different models
- Unit-Tests for each feature
- Integration-Tests together after a feature is finished
- Have siblings/family testing the functions
- In the future Selenium-Tests for the UI

Pain Points

- Nvidia Cuda
- Merge Conflicts (broken merges)
- Opposite dependencies
- Hardware limitations
- Time pressure
- Being hung up on details
- Testing in Unittest
- Testing Ai
- Bugs introduced by using LLM for Coding (including but not limited to Claude 3.7)


Developer coding - 2 hours
Developer debugging - 6 hours
ChatGPT generates Codes - 5 min
Developer debugging - 24 hours

Lessons learned

- Importance of a good Requirement documentation
- Importance of proper diagrams for the vision and visualisation
- Importance of reading a documentation
- Importance of well structured code basis and modularity
- Communication and Coordination due to dependencies
- Value of pair programming
- Never trust LLMs and AI completely

Live Demo

Future Prospects

- What is left to do:
  - Proper connection between STT and frontend
  - Thread, Threads-Safety and multiple users at once for the application/ai-models
  - More and better Unit-Tests or even Selenium-Tests for the UI
  - Documentation and code clean up in frontend-module
  - More Export-Options for the user
  - Updating texts from editor in the database
  - Improved test extraction
- Future Prospects:
  - Admin-Panel for the user
  - Editing book editing on website
  - Image support and detection
  - Switching to HTTP-Streams or Websockets for real time speed
  - Improved test extraction
  - Test-Driven-Development