**Documentation for FastAPI Application with Prediction Endpoints**

**FastAPI:**

FastAPI is a modern, fast (hence the name), web framework for building APIs with Python. It is built on top of the Starlette framework and provides a simple, declarative syntax for defining API endpoints, request and response models, and route parameters. Some advantages of FastAPI are:

1. **High Performance**: FastAPI is designed to be highly performant, thanks to its use of asynchronous I/O and code generation techniques. It can handle a large number of requests per second with low latency.

2. **Easy to Use**: FastAPI is designed to be easy to use, with a simple, intuitive syntax that is similar to popular Python libraries like Flask and Django. It provides automatic validation of request and response data, making it easier to catch errors early in the development process.

3. **Automatic Documentation**: FastAPI automatically generates interactive API documentation based on the code you write, making it easy for developers to understand and use your API.

4. **Type Annotations**: FastAPI uses Python's type annotation system to provide automatic data validation and documentation. This makes it easier to catch errors early in the development process and to understand what data is expected and returned by your API.

5**. Easy Integration with Other Libraries**: FastAPI is designed to work well with other popular Python libraries like SQLAlchemy, Pydantic, and OAuth2. This makes it easy to build complex APIs with minimal effort.

6. **Fast Development Time**: FastAPI's easy-to-use syntax and automatic data validation and documentation can help reduce development time significantly.

Our FastAPI application provides three endpoints for predicting different types of data: words, numbers, and alphabets. The code processes uploaded files, writes them to a local directory called "data", and passes the file path to a corresponding prediction script. The predictions are then returned as the response to the client.

**API Endpoints:**

***1. /prediction/word POST***

* Endpoint for predicting words contained in an uploaded file.
* Uploads a file using the "file" parameter.
* Returns the prediction result as the response.

***2. /prediction/number POST***

* Endpoint for predicting numbers contained in an uploaded file.
* Uploads a file using the "file" parameter.
* Returns the prediction result as the response.

***3. /prediction/alphabet POST***

* Endpoint for predicting alphabets contained in an uploaded file.
* Uploads a file using the "file" parameter.
* Returns the prediction result as the response.

**Custom Prediction Scripts:**

* `*prediction\_script\_words.py*`: contains a function that takes a file path as input, processes the file to predict words, and returns the prediction result.
* `*prediction\_script\_numbers.py*`: contains a function that takes a file path as input, processes the file to predict numbers, and returns the prediction result.
* `*prediction\_script\_alphabet.py*`: contains a function that takes a file path as input, processes the file to predict alphabets, and returns the prediction result.

**Error Handling:**

* If there is an error uploading the file, the API returns a JSON response with the message "There was an error uploading the file".

**Security**:

* This code snippet does not include any authentication or authorization mechanisms. Additional security measures may need to be deployed in production environments.

**Note**:

* This code snippet assumes that the `data` directory exists and is writable.
* The `file` parameter is required for each endpoint and must be of type "multipart/form-data".

**Implementations:**

from fastapi import FastAPI, File, UploadFile

from prediction\_script\_words import predict\_video

from prediction\_script\_numbers import predict\_number

from prediction\_script\_alphabet import predict\_alphabet

import os

app = FastAPI()

@app.post("/prediction/word")

async def create\_upload\_file(file: UploadFile = File(...)):

    try:

        contents = file.file.read()

        with open(os.path.join('data', file.filename), 'wb') as f:

            f.write(contents)

    except Exception:

        return {"message": "There was an error uploading the file"}

    finally:

        file.file.close()

    result = predict\_video(os.path.join('data', file.filename))

    return result

@app.post("/prediction/number")

async def create\_upload\_file(file: UploadFile = File(...)):

    try:

        contents = file.file.read()

        with open(os.path.join('data', file.filename), 'wb') as f:

            f.write(contents)

    except Exception:

        return {"message": "There was an error uploading the file"}

    finally:

        file.file.close()

    result = predict\_number(os.path.join('data', file.filename))

    return result

@app.post("/prediction/alphabet")

async def create\_upload\_file(file: UploadFile = File(...)):

    try:

        contents = file.file.read()

        with open(os.path.join('data', file.filename), 'wb') as f:

            f.write(contents)

    except Exception:

        return {"message": "There was an error uploading the file"}

    finally:

        file.file.close()

    result = predict\_alphabet(os.path.join('data', file.filename))

    return result

**Conclusion**:

Overall, FastAPI is a powerful, easy-to-use web framework for building high-performance APIs with Python, and its many advantages make it a popular choice among developers.

**References:**

1. Official documentation: [https://fastapi.tiangolo.com/ ↗](https://fastapi.tiangolo.com/)
2. FastAPI GitHub repository: [https://github.com/tiangolo/fastapi ↗](https://github.com/tiangolo/fastapi)
3. "FastAPI: Building High-Performance APIs with Python 3.7+" by Sebastián Ramírez: [https://www.packtpub.com/product/fastapi-building-high-performance-apis-with-python-3-7/9781800205457 ↗](https://www.packtpub.com/product/fastapi-building-high-performance-apis-with-python-3-7/9781800205457)
4. "Building APIs with FastAPI" by Michael Herman: [https://testdriven.io/courses/fastapi-building-apis-with-python-and-fastapi/ ↗](https://testdriven.io/courses/fastapi-building-apis-with-python-and-fastapi/)
5. "FastAPI Web Development Cookbook" by Yasoob Khalid: [https://www.packtpub.com/product/fastapi-web-development-cookbook/9781800563762 ↗](https://www.packtpub.com/product/fastapi-web-development-cookbook/9781800563762)