

```
/**  
 * perit.hu  
 */
```

# MicroDMS

USER MANUAL

1.1

```
/**  
 * perit.hu  
 */
```

## Document history

Version	Date	Modified by	Modification
1.0	2025-02-02	Peter Nagy	First version
1.1	2025-03-16	Peter Nagy	Storage subsystem description, Autocomplete keywords

```
/**  
 * perit.hu  
 */
```

## Table of Content

<b>1.</b>	<b>What is this</b>	<b>5</b>
1.1.	<i>Main characteristics</i>	5
1.2.	<i>Technologies</i>	6
<b>2.</b>	<b>System configuration</b>	<b>7</b>
2.1.	<i>The storage subsystem</i>	7
2.2.	<i>Keywords</i>	9
2.3.	<i>Document types</i>	12
2.4.	<i>Document type groups</i>	13
2.5.	<i>User groups</i>	14
<b>3.</b>	<b>MicroDMS API</b>	<b>16</b>
3.1.	<i>Authentication</i>	16
3.1.1.	<i>authenticate()</i>	16
3.2.	<i>Config</i>	16
3.2.1.	<i>getConfig()</i>	16
3.2.2.	<i>applyConfig()</i>	17
3.3.	<i>System</i>	17
3.3.1.	<i>getDocumentTypes()</i>	17
3.3.2.	<i>getDocumentTypeGroups()</i>	18
3.3.3.	<i>getKeywordTypes()</i>	18
3.4.	<i>Files</i>	19
3.4.1.	<i>upload()</i>	19
3.5.	<i>Documents</i>	19
3.5.1.	<i>createDocument()</i>	19
3.5.2.	<i>readDocument()</i>	21
3.5.3.	<i>updateDocument()</i>	21
3.5.4.	<i>deleteDocument</i>	22
3.5.5.	<i>getDocumentKeywords()</i>	22
3.5.6.	<i>getDocumentInfo()</i>	23
3.5.7.	<i>queryDocument()</i>	23
<b>4.</b>	<b>MicroDMS Client</b>	<b>29</b>
4.1.	<i>The Retrieval page</i>	29
4.2.	<i>The Import page</i>	35
<b>5.</b>	<b>Performance and volume measures</b>	<b>36</b>

```
/**  
 * perit.hu  
 */
```

<b>6.</b>	<b>Appendix</b>	<b>40</b>
6.1.	<i>Referenced Documents</i>	40
6.2.	<i>List of Figures</i>	40
6.3.	<i>Open Issues</i>	40
6.4.	<i>Requirement Keys</i>	40

## 1. WHAT IS THIS

The MicroDMS system is a document archive for storing and retrieving documents. It was inspired by Saperion and OnBase but does not want to compete with any of them. Both archives have many more features. The MicroDMS system focuses on the basic functionality of a document archive, which is storing, reading, updateing and deleting of documents.

However the system will be prepared for GDPR removal and recertification of digitally signed documents through external tools to provide a complete solution for various businesses.

### 1.1. MAIN CHARACTERISTICS

- Windows and Linux systems are supported.
- Various deployments are possible, starting from a Window service through Docker until Kubernetes or Openshift.
- Can be hosted in an on-premise infrastructure or in the cloud.
- The meta-data is stored in a database. Currently supported Microsoft SQL-Server and PostgreSQL.
- Multivalue keywords are supported.
- The following data types are supported:
  - INTEGER
  - LONG
  - ALPHANUMERIC
  - CURRENCY
  - SPECIFIC\_CURRENCY
  - DATE
  - DATETIME
  - FLOATING\_POINT
  - BOOLEAN
- Files can be stored in local drives or on SMB shares.
- The configuration of the archive system is stored in the database. The current configuration can be downloaded into a file in yaml format, or can be applied from a file.

```
/**  
 * perit.hu  
 */
```

## 1.2. TECHNOLOGIES

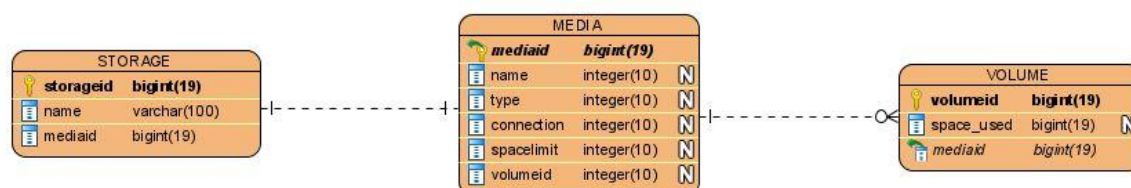
- Spring Boot
- Java17
- Sql-Server, PostgreSQL
- Ngface
- Spvitamin

```
/**
 * perit.hu
 */
```

## 2. SYSTEM CONFIGURATION

### 2.1. THE STORAGE SUBSYSTEM

The following model is used:



**Figure 1: Database model of the storage subsystem**

**Media:** a particular physical storage medium. It can be e.g. a local drive, an SMB share or later a cloud storage. In this entity we store the path of the physical medium and the data needed to access it, e.g. username and password. Also the maximum size of the volume will be stored here.

**Volume:** files are stored in volumes on the media. The primary goal of having volumes is the ability of making backups easily. A volume is filled only until the maximum size is reached. In case the volume reached its maximum size, a new volume will be created.

**Storage:** this is a logical entity, which references a particular media. The referenced media within the storage may change with the time. For example, if the currently used media gets full, we can assign another media to the storage, and archiving of documents can be continued using the same storage.

This is a sample configuration, which configures two media and a storage for each.

```
#-----
media:
#-----
# LOCAL_DRIVE
- kind: Media
  apiVersion: v1
  name: Y-drive
  mediaType: LOCAL_DRIVE
  connection:
    type: LOCAL_DRIVE
    path: y:\filestorage\microdms
  spaceLimit: 1 GB
# SMB
- kind: Media
  apiVersion: v1
  name: NETAPP
  mediaType: SMB
```

```
/**
 * perit.hu
 */
```

# MicroDMS

```
connection:
  type: SMB
  host: smb://perit
  share: smb-test
  username: smb_test
  # alma
  password: ntJ9hrf4awc=
  domain: perit
  spaceLimit: 1 GB

#-----
storages:
#-----

# PERF_TEST_STORAGE
- kind: Storage
  apiVersion: v1
  name: PERF_TEST_STORAGE
  media: Z-drive

# LONG_TERM_STORAGE
- kind: Storage
  apiVersion: v1
  name: LONG_TERM_STORAGE
  media: NETAPP
```

Files are stored on the media as depicted on the following picture.

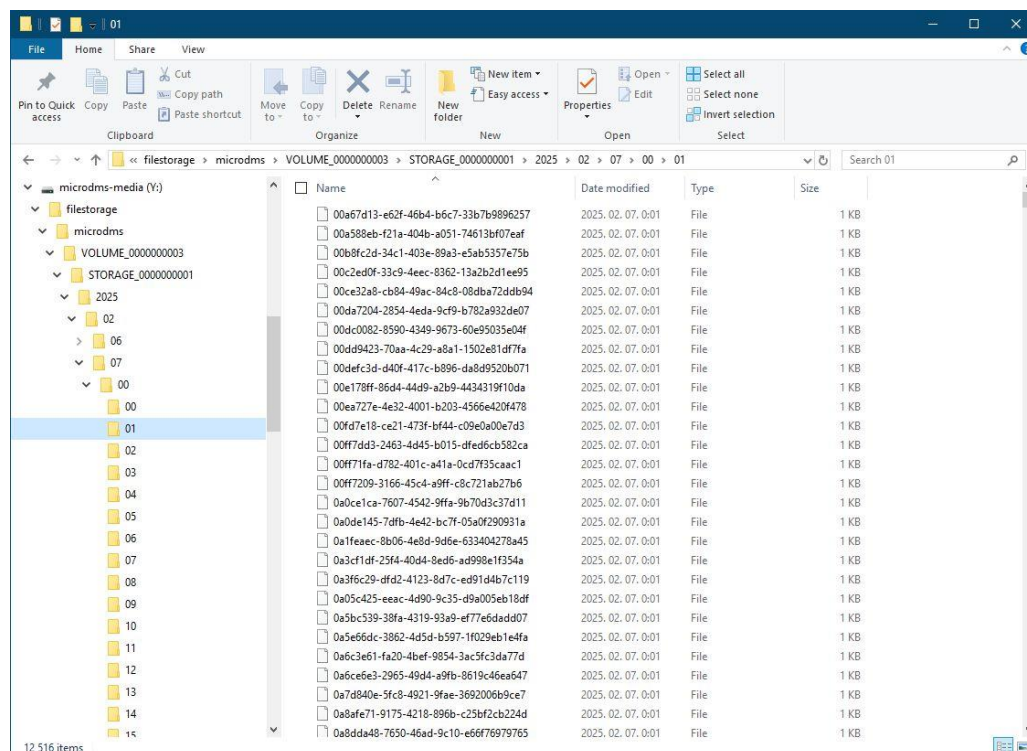


Figure 2: Files on the media



```
/**
 * perit.hu
 */
```

## 2.2. KEYWORDS

Sample configuration:

```
#-----
keywords:
#-----
# Alphanumeric
- kind: Keyword
  apiVersion: v1
  name: Alphanumeric
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_UPPERCASE
  length: 50
# Numeric9
- kind: Keyword
  apiVersion: v1
  name: Integer
  dataType: INTEGER
# Numeric20
- kind: Keyword
  apiVersion: v1
  name: Long
  dataType: LONG
# Date
- kind: Keyword
  apiVersion: v1
  name: Date
  dataType: DATE
# DateTime
- kind: Keyword
  apiVersion: v1
  name: DateTime
  dataType: DATETIME
# FloatingPoint
- kind: Keyword
  apiVersion: v1
  name: FloatingPoint
  dataType: FLOATING_POINT
# Boolean
- kind: Keyword
  apiVersion: v1
  name: Boolean
  dataType: BOOLEAN
# FirstName
- kind: Keyword
  apiVersion: v1
  name: FirstName
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_MIXEDCASE
  length: 30
# LastName
- kind: Keyword
  apiVersion: v1
  name: LastName
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_MIXEDCASE
  length: 30
```

```
/**
 * perit.hu
 */
```

```
# Birthdate
- kind: Keyword
  apiVersion: v1
  name: Birthdate
  dataType: DATE
# Address
- kind: Keyword
  apiVersion: v1
  name: Address
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_MIXEDCASE
  length: 100
# Salary
- kind: Keyword
  apiVersion: v1
  name: Salary
  dataType: INTEGER
# InvoiceNumber
- kind: Keyword
  apiVersion: v1
  name: InvoiceNumber
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_MIXEDCASE
  length: 20
# InvoiceDate
- kind: Keyword
  apiVersion: v1
  name: InvoiceDate
  dataType: DATE
# DueDate
- kind: Keyword
  apiVersion: v1
  name: DueDate
  dataType: DATE
# Currency
- kind: Keyword
  apiVersion: v1
  name: Currency
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_UPPERCASE
  length: 3
# IssuerName
- kind: Keyword
  apiVersion: v1
  name: IssuerName
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_MIXEDCASE
  length: 100
# IssuerAddress
- kind: Keyword
  apiVersion: v1
  name: IssuerAddress
  dataType: ALPHANUMERIC
  dataTypeOption: ALPHANUMERIC_MIXEDCASE
  length: 100
# IssuerContactPhone
- kind: Keyword
  apiVersion: v1
  name: IssuerContactPhone
```

```
/**  
 * perit.hu  
 */
```

```
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_UPPERCASE  
    length: 20  
  # IssuerVATNumber  
  - kind: Keyword  
    apiVersion: v1  
    name: IssuerVATNumber  
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_UPPERCASE  
    length: 20  
  # IssuerAccountNumber  
  - kind: Keyword  
    apiVersion: v1  
    name: IssuerAccountNumber  
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_UPPERCASE  
    length: 40  
  # CustomerName  
  - kind: Keyword  
    apiVersion: v1  
    name: CustomerName  
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_MIXEDCASE  
    length: 100  
  # CustomerAddress  
  - kind: Keyword  
    apiVersion: v1  
    name: CustomerAddress  
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_MIXEDCASE  
    length: 100  
  # CustomerContactPhone  
  - kind: Keyword  
    apiVersion: v1  
    name: CustomerContactPhone  
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_UPPERCASE  
    length: 20  
  # CustomerVATNumber  
  - kind: Keyword  
    apiVersion: v1  
    name: CustomerVATNumber  
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_UPPERCASE  
    length: 20  
  # InvoiceDescription  
  - kind: Keyword  
    apiVersion: v1  
    name: InvoiceDescription  
    dataType: ALPHANUMERIC  
    dataTypeOption: ALPHANUMERIC_MIXEDCASE  
    length: 100  
  # NetPrice  
  - kind: Keyword  
    apiVersion: v1  
    name: NetPrice  
    dataType: FLOATING_POINT  
  # Tax  
  - kind: Keyword
```

```
/**
 * perit.hu
 */
```

```
    apiVersion: v1
    name: Tax
    dataType: FLOATING_POINT
  # TotalAmount
  - kind: Keyword
    apiVersion: v1
    name: TotalAmount
    dataType: FLOATING_POINT
```

## 2.3. DOCUMENTTYPES

Here we can configure document types and assign keywords to them.

```
#-----
documentTypes:
#-----
  # KeywordTest
  - kind: DocumentType
    apiVersion: v1
    name: KeywordTest
    storage: LONG_TERM_STORAGE
    keywords:
      - name: Alphanumeric
      - name: Integer
      - name: Long
      - name: Date
      - name: DateTime
      - name: FloatingPoint
      - name: Boolean
      - name: AGE
      flags:
        - READONLY
        - HIDDEN
  # PerformanceTest
  - kind: DocumentType
    apiVersion: v1
    name: PerformanceTest
    storage: PERF_TEST_STORAGE
    keywords:
      - name: Long
      - name: Alphanumeric
      - name: FloatingPoint
      - name: DateTime
      - name: Date
  # Employee
  - kind: DocumentType
    apiVersion: v1
    name: Employee
    storage: LONG_TERM_STORAGE
    keywords:
      - name: FirstName
      flags:
        - AUTOCOMPLETE
      - name: LastName
      flags:
        - AUTOCOMPLETE
```

```
/**
 * perit.hu
 */
```

```
- name: Birthdate
- name: Address
  flags:
    - AUTOCOMPLETE
- name: Salary
# Invoice
- kind: DocumentType
  apiVersion: v1
  name: Invoice
  storage: LONG_TERM_STORAGE
  keywords:
    - name: InvoiceNumber
    - name: InvoiceDescription
    - name: InvoiceDate
    - name: DueDate
    - name: IssuerName
    - name: IssuerAddress
    - name: IssuerContactPhone
    - name: IssuerVATNumber
    - name: IssuerAccountNumber
    - name: CustomerName
    - name: CustomerAddress
    - name: CustomerContactPhone
    - name: CustomerVATNumber
    - name: Currency
    - name: NetPrice
    - name: Tax
    - name: TotalAmount
```

Keywords may have the following flags: **READONLY**, **HIDDEN**, **NOTNULL**, **AUTOCOMPLETE**.

Flag	Description
<b>READONLY</b>	Not yet implemented
<b>HIDDEN</b>	Not yet implemented
<b>NOTNULL</b>	Not yet implemented
<b>AUTOCOMPLETE</b>	On the Keywords pane the existing values of the keyword will be displayed in an autocomplete field. This may have a performance impact. Use this feature only with smaller keyword tables which consist of less than 1 million records.

## 2.4. DOCUMENTTYPE GROUPS

```
#-----
documentTypeGroups:
#-----
# Test
- kind: DocumentTypeGroup
  apiVersion: v1
  name: Test
  documentTypes:
    - KeywordTest
    - PerformanceTest
# HR Documents
```

```
/**
 * perit.hu
 */
```

```
- kind: DocumentTypeGroup
  apiVersion: v1
  name: HR Documents
  documentTypes:
    - Employee
# FIN Documents
- kind: DocumentTypeGroup
  apiVersion: v1
  name: FIN Documents
  documentTypes:
    - Invoice
```

## 2.5. USER GROUPS

User groups control access to document types.

```
#-----
usergroups:
#-----
- kind: UserGroup
  apiVersion: v1
  name: MANAGER
  documentTypes:
    - KeywordTest
    - PerformanceTest
    - Employee
    - Invoice
- kind: UserGroup
  apiVersion: v1
  name: HR
  documentTypes:
    - Employee
- kind: UserGroup
  apiVersion: v1
  name: FIN
  documentTypes:
    - Invoice
```

Configuration of local users or AD group assignments can be made in the usual way on the application.yaml.

```
#-----
# Local users for test reasons
#-----
# Please set either password or encrypted-password property
localuser:
  admin.encrypted-password: ij3HfoF+WYo=
  user.password: user
  manager.password: manager
  hr.password: hr
  fin.password: fin

#-----
# AD group -> role mapping (all = '*')
```

```
/**
 * perit.hu
 */
```

```
#-----
roles:
  # Site admin
  ROLE_ADMIN:
    #groups: any AD group
    users: nagy_peter,admin
    includes: ROLE_USER

  # User
  ROLE_USER:
    #groups: any AD group
    users: nagy_peter,user

  # MANAGER
  ROLE_MANAGER:
    #groups: any AD group
    users: manager
    includes: ROLE_USER

  # HR
  ROLE_HR:
    #groups: any AD group
    users: hr
    includes: ROLE_USER

  # FIN
  ROLE_FIN:
    #groups: any AD group
    users: fin
    includes: ROLE_USER
```

## MicroDMS



```
/**
 * perit.hu
 */
```

### 3.2.1.2. Response

The endpoint returns the current configuration as a yaml file.

## 3.2.2. APPLYCONFIG()

### 3.2.2.1. Request

```
curl --location 'http://localhost:8488/api/config' \
--form 'file=@"/C:/np/github/microdms-project/test-data/microdms-config.yaml"'
```

### 3.2.2.2. Response

Only the HTTP response code is returned.

## 3.3. SYSTEM

### 3.3.1. GETDOCUMENTTYPES()

#### 3.3.1.1. Request

```
curl --location 'http://localhost:8488/api/system/documenttypes'

curl --location
'http://localhost:8488/api/system/documenttypes?documentTypeGroupName=Test'
```

#### 3.3.1.2. Response

The endpoint returns the accessible document types for the authenticated user. If the documentTypeGroupName query parameter is provided, the result contains only document types of the given document type group.

```
[
  {
    "id": "1",
    "name": "KeywordTest"
  },
  {
    "id": "2",
    "name": "PerformanceTest"
  },
  {
    "id": "3",
    "name": "Employee"
  },
  {
    "id": "4",
```

```
/**
 * perit.hu
 */
```

```
        "name": "Invoice"
    }
]
```

## 3.3.2. GETDOCUMENTTYPEGROUPS()

### 3.3.2.1. Request

```
curl --location 'http://localhost:8488/api/system/documenttypegroups'
```

### 3.3.2.2. Response

```
[
  "Test",
  "HR Documents",
  "FIN Documents"
]
```

## 3.3.3. GETKEYWORDTYPES()

### 3.3.3.1. Request

```
curl --location
'http://localhost:8488/api/system/keywordtypes?documentTypeName=Employee'
```

### 3.3.3.2. Response

```
[
  {
    "name": "FirstName",
    "id": 8,
    "dataType": "ALPHANUMERIC",
    "length": 30,
    "dataTypeOption": "ALPHANUMERIC_MIXEDCASE"
  },
  {
    "name": "LastName",
    "id": 9,
    "dataType": "ALPHANUMERIC",
    "length": 30,
    "dataTypeOption": "ALPHANUMERIC_MIXEDCASE"
  },
  {
    "name": "Birthdate",
    "id": 10,
    "dataType": "DATE"
  },
  {
    "name": "Address",
    "id": 11,
```

```
/**
 * perit.hu
 */
```

```
        "dataType": "ALPHANUMERIC",
        "length": 100,
        "dataTypeOption": "ALPHANUMERIC_MIXEDCASE"
    },
    {
        "name": "Salary",
        "id": 12,
        "dataType": "INTEGER"
    }
]
```

## 3.4. FILES

### 3.4.1. UPLOAD()

Files can be uploaded in binary form with the following endpoint.

#### 3.4.1.1. Request

```
curl --location 'http://localhost:8488/api/files' \
--form 'file=@"/C:/np/github/microdms-project/test-data/SZEKK.pdf"'
```

#### 3.4.1.2. Response

The response is the reference of the file within the temporary store. The file will be removed from the temp store after successful archiving or after 1 hour if not used.

```
{
  "location": "2024/12/31/12/51/6915b3ba-8167-48a9-9645-025db3f6e5a0"
}
```

## 3.5. DOCUMENTS

### 3.5.1. CREATEDOCUMENT()

#### 3.5.1.1. Request

```
curl --location 'http://localhost:8488/api/documents' \
--header 'Content-Type: application/json' \
--data '{
  "documentTypeName": "Employee",
  "docRef": "2025/02/02/08/42/3fba3113-c933-47da-8be4-1312b2ee5f85",
  "fileName": "excelfile.pdf",
  "keywords": [
    {
      "name": "FirstName",
```

```
/**
 * perit.hu
 */
```

```
        "value": "John"
      },
      {
        "name": "LastName",
        "value": "Doe"
      },
      {
        "name": "Birthdate",
        "value": "1999-12-01"
      }
    ]
  },
}
```

Multivalue keywords are supported in the following form.

```
{
  "documentTypeName": "KeywordTest",
  "docRef": "{{fileRef}}",
  "fileName": "excelfile.pdf",
  "keywords": [
    {
      "name": "Alphanumeric",
      "values": [
        "alma",
        "körte"
      ]
    },
    {
      "name": "Integer",
      "values": [
        123,
        124
      ]
    },
    {
      "name": "Long",
      "values": [
        123456789,
        1234567890
      ]
    },
    {
      "name": "Date",
      "values": [
        "2024-11-30",
        "2024-12-01"
      ]
    }
  ]
}
```

Please note the syntax: instead of "value", the plural form is used: "values" and the value is supposed to be a list of objects.

### 3.5.1.2. Response

```
{
```

```
/**
 * perit.hu
 */
```

```
"location": "http://localhost:8488/api/documents/12409602"
}
```

## 3.5.2. READDOCUMENT()

### 3.5.2.1. Request

```
curl --location 'http://localhost:8488/api/documents/12409602'
```

### 3.5.2.2. Response

The file content as application/octet-stream.

## 3.5.3. UPDATEDOCUMENT()

### 3.5.3.1. Request

```
curl --location --request PUT 'http://localhost:8488/api/documents' \
--header 'Content-Type: application/json' \
--data '{
  "docId": "10001",
  "keywords": [
    {
      "name": "Alphanumeric",
      "value": "TEST"
    }
  ],
  "docRef": "2024/12/31/12/51/6915b3ba-8167-48a9-9645-025db3f6e5a0",
  "fileName": "updated.pdf"
}'
```

The keyword will be deleted if the value is 'null'. Both keywords and docRef are optional, only the provided data will be updated.

### 3.5.3.1.1. Response

```
{
  "id": "10001",
  "rev": "1",
  "name": "updated.pdf",
  "documentDate": "2024-12-31T13:13:50.335+0100",
  "dateStored": "2024-12-31T13:13:50.362+0100",
  "documentTypeName": "PerformanceTest",
  "documentTypeId": "3",
  "createdBy": "manager",
  "latestAllowedRevisionID": 1,
  "status": 0,
  "fullFileName": "updated.pdf",
  "fileSize": 80728
}
```

```
/**
 * perit.hu
 */
```

```
}
```

## 3.5.4. DELETEDOCUMENT

### 3.5.4.1. Request

The body part shall contain the type of the removal. There are two types:

- LOGICAL
- PHYSICAL

```
curl --location --request DELETE 'http://localhost:8488/api/documents/12407517' \
--header 'Content-Type: application/json' \
--data '"PHYSICAL"'
```

### 3.5.4.2. Response

Only the HTTP status will be returned.

## 3.5.5. GETDOCUMENTKEYWORDS()

### 3.5.5.1. Request

```
curl --location --request POST 'http://localhost:8488/api/documents/8365730/keywords'
```

### 3.5.5.2. Response

```
{
  "keywords": {
    "LASTNAME": {
      "name": "LastName",
      "value": "Doe",
      "dataType": "ALPHANUMERIC",
      "dataTypeOption": "ALPHANUMERIC_MIXEDCASE"
    },
    "FIRSTNAME": {
      "name": "FirstName",
      "value": "John",
      "dataType": "ALPHANUMERIC",
      "dataTypeOption": "ALPHANUMERIC_MIXEDCASE"
    },
    "BIRTHDATE": {
      "name": "Birthdate",
      "value": "1999-12-01",
      "dataType": "DATE"
    }
  }
}
```

```
/**
 * perit.hu
 */
```

An example of a multivalue keyword:

```
{
  "keywords": {
    "ALPHANUMERIC": {
      "name": "Alphanumeric",
      "values": [
        "ALMAKÖRTE",
        "BARACK",
        "SZILVA"
      ],
      "dataType": "ALPHANUMERIC",
      "dataTypeOption": "ALPHANUMERIC_UPPERCASE",
      "multiValue": true
    }
  }
}
```

### 3.5.6. GETDOCUMENTINFO()

#### 3.5.6.1. Request

```
curl --location 'http://localhost:8488/api/documents/102/info'
```

#### 3.5.6.2. Response

```
{
  "id": 8365730,
  "rev": 0,
  "name": "excelfile",
  "documentDate": "2025-02-02T08:43:10.386+0100",
  "dateStored": "2025-02-02T08:43:10.392+0100",
  "documentTypeName": "Employee",
  "documentTypeId": 3,
  "createdBy": "manager",
  "latestAllowedRevisionID": 0,
  "status": 0,
  "fileName": "excelfile.pdf",
  "fileSize": 80728,
  "media": "smb://perit/smb-test",
  "location": "2025/02/02/08/43/2ee618cc-d5ba-4284-bd9a-04196ec1fee6"
}
```

### 3.5.7. QUERYDOCUMENT()

The queryDocument() endpoint is a powerful tool for searching documents by document attributes or by keywords. The result set will be a distinct list of documents, even if the result set contains keywords with multiple values.

```
/**
 * perit.hu
 */
```

## 3.5.7.1. Request

```
curl --location 'http://localhost:8488/api/query?size=20&page=0' \
--data '{
  "where": [
    {
      "property": "Document.ID",
      "relation": "=",
      "value": 45209
    }
  ],
  "orderBy": {
    "property": "Alphanumeric",
    "direction": "ASC"
  },
  "include": ["Alphanumeric", "DateTime"],
  "countOnly": false
}'
```

The request has 4 body parts: **where**, **orderBy**, **include** and **countOnly**, and it has 2 query parameters for pagination: **size** and **page**.

### 3.5.7.1.1. The where clause

The where clause consists of a list of criteria, which will be handled using the AND relation. With other words the more criteria is provided here, the result set will be smaller. The property may be a keyword name or the following document attributes:

- Document.ID
- Document.Name
- Document.Type
- Document.Date
- Document.DateStored
- Document.CreatedBy

The relation can be one of the following:

- "="
- "<>"
- ">"
- ">="
- "<"
- "<="
- "IN"
- "BETWEEN"
- "LIKE"

### 3.5.7.1.2. Narrowing the result using Document.Type

By default all document types are queried which contain the given keywords. You can narrow down the result like this:



```
/**
 * perit.hu
 */
```

```
curl --location 'http://localhost:8488/api/query' \  
--data '{  
  "where": [  
    {  
      ...  
    },  
    {  
      "property": "Document.Type",  
      "relation": "=",  
      "value": "Employee"  
    }  
  ]  
'
```

### 3.5.7.1.3. Using the IN relation

When querying for documents using the IN relation, the values must be a list of objects.

```
curl --location 'http://localhost:8488/api/query?size=3' \  
--header 'Content-Type: application/json' \  
--data '{  
  "where": [  
    {  
      "property": "Alphanumeric",  
      "relation": "IN",  
      "values": [  
        "ID_163",  
        "ID_164",  
        "ID_165"  
      ]  
    }  
  ]  
'
```

### 3.5.7.1.4. Using the BETWEEN relation

In this case the values list must contain exactly 2 items.

```
curl --location 'http://localhost:8488/api/query' \  
--data '{  
  "where": [  
    {  
      "property": "Alphanumeric",  
      "relation": "BETWEEN",  
      "values": [  
        "ID_1",  
        "ID_10"  
      ]  
    }  
  ]  
'
```

```
/**
 * perit.hu
 */
```

## 3.5.7.1.5. Using the LIKE relation

The LIKE relation may only be used with alphanumeric keywords. Since those keywords are stored as fixed length texts (they are padded with spaces), please use % at the end of the filter text.

```
curl --location 'http://localhost:8488/api/query' \
--data '{
  "where": [
    {
      "property": "Alphanumeric",
      "relation": "LIKE",
      "value": "%ALMA%"
    }
  ]
}'
```

## 3.5.7.1.6. Sorting the result

You can use the **orderBy** part of the request for sorting the result. Please consider that sorting will be applied on the result set specified by the where clause, that is why it may be slow if the set to be sorted is large.

For saving server resources, the orderBy clause will be ignored if the result set is larger than 100.000 documents. In that case a warning appears in the result.

```
Request:
{
  "where": [
    {
      "property": "Document.ID",
      "relation": "<>",
      "value": null
    }
  ],
  "orderBy": {
    "property": "Alphanumeric",
    "direction": "ASC"
  },
  "include": ["Alphanumeric"]
}

Response:
{
  "size": 100,
  "page": 1000,
  "totalPages": 124068,
  "totalItems": 12406736,
  "warning": "OrderBy clause ignored due to too many results!",
  "keywords": [
    "Alphanumeric"
  ],
  "list": [
    {
      "documentInfo": {
        "id": "100001",
```

```
/**
 * perit.hu
 */
```

```
{
  "rev": "0",
  "name": "tiny.txt",
  "documentDate": "2024-12-22T14:05:04.258+0100",
  "dateStored": "2024-12-22T14:05:04.258+0100",
  "documentTypeName": "PerformanceTest",
  "documentTypeId": "3",
  "createdBy": "manager",
  "status": 0
},
"keywords": [
  {
    "name": "Alphanumeric",
    "value": "ID_486",
    "dataType": "ALPHANUMERIC",
    "dataTypeOption": "ALPHANUMERIC_UPPERCASE"
  }
]
},
...
]
```

The socket-timeout is set to 20 seconds, which will terminate each query after this amount of time.

### 3.5.7.1.7. Including keywords in the result

You can get not only document attributes with the result, but also keywords. This is especially good if you want to show the result in a data table, because the whole table is retrieved in one step. Please use the "include" field in the request.

### 3.5.7.1.8. Pagination of the result

The response contains information about the size of the result set. You can iterate through the result set using the usual size and page parameters in the query like this:

```
curl --location 'http://localhost:8488/api/query?size=3&page=10'
```

If not provided, 100 items will be returned with offset = 0.

### 3.5.7.2. Response

```
{
  "size": 1,
  "page": 0,
  "totalPages": 1,
  "totalItems": 1,
  "keywords": [
    "Alphanumeric",
    "DateTime"
  ],
  "list": [
```

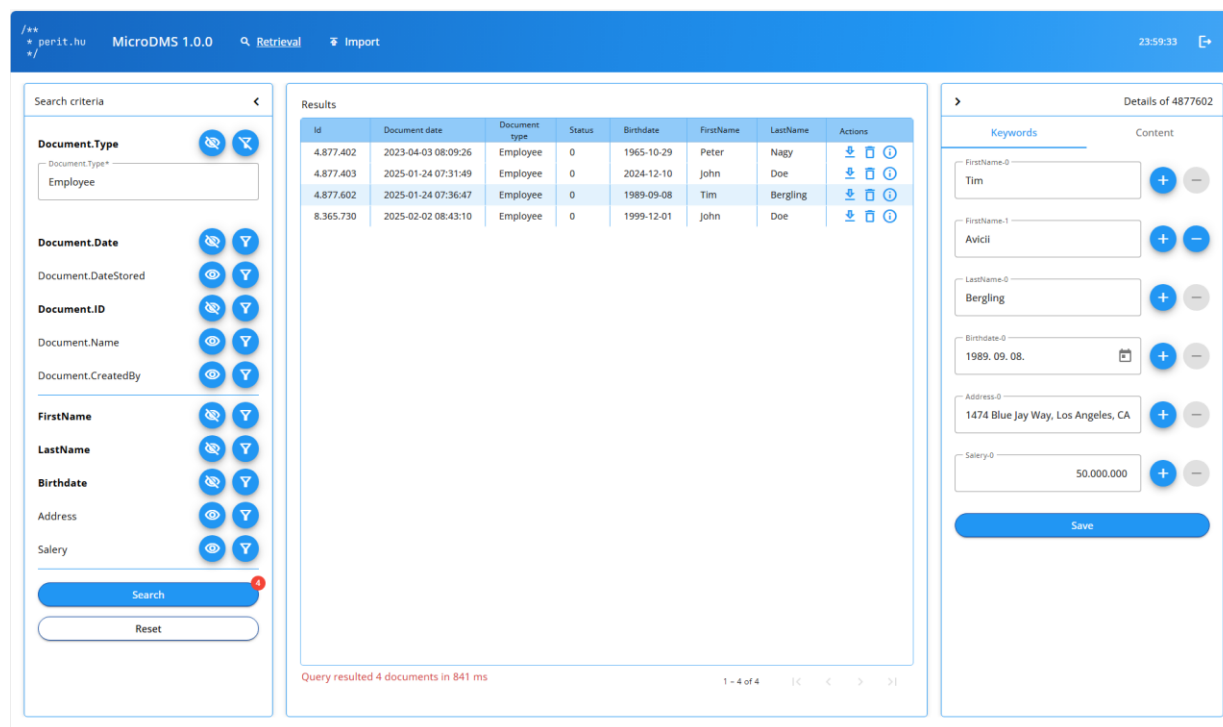
```
/**
 * perit.hu
 */
```

```
{
  "documentInfo": {
    "id": "45209",
    "rev": "0",
    "name": "tiny.txt",
    "documentDate": "2024-12-22T13:48:56.273+0100",
    "dateStored": "2024-12-22T13:48:56.273+0100",
    "documentTypeName": "PerformanceTest",
    "documentTypeId": "3",
    "createdBy": "manager",
    "status": 0
  },
  "keywords": [
    {
      "name": "Alphanumeric",
      "value": "ID_748",
      "dataType": "ALPHANUMERIC",
      "dataTypeOption": "ALPHANUMERIC_UPPERCASE"
    },
    {
      "name": "DateTime",
      "value": "2024-12-22T12:48:56.236+0100",
      "dataType": "DATETIME"
    }
  ]
}
```

## 4. MICRODMS CLIENT

### 4.1. THE RETRIEVAL PAGE

The main purpose of the client is searching for documents. The result set is displayed on the central panel.



**Figure 3: The MicroDMS web client**

The search result can be narrowed down by specifying some search criteria on the left panel. Here we can switch on/off document attributes and keywords.

```
/**
 * perit.hu
 */
```

The screenshot shows the 'Search criteria' panel with the following fields and controls:

- Document.Type**: A dropdown menu with 'Employee' selected. It has a 'Display on/off' toggle (eye icon) and a filter icon (funnel).
- Document.Date**: A text input field with 'Document.DateStored' as the placeholder. It has a 'Display on/off' toggle and a filter icon.
- Document.ID**: A text input field. It has a 'Display on/off' toggle and a filter icon.
- Document.Name**: A text input field with 'Document.CreatedBy' as the placeholder. It has a 'Display on/off' toggle and a filter icon.
- FirstName**: A text input field with 'Peter' entered. It has a 'Filter on/off' toggle (eye icon) and a filter icon.
- LastName**: A text input field. It has a 'Display on/off' toggle and a filter icon.
- Birthdate**: A text input field. It has a 'Display on/off' toggle and a filter icon.
- Address**: A text input field. It has a 'Display on/off' toggle and a filter icon.
- Salary**: A text input field. It has a 'Display on/off' toggle and a filter icon.
- Search**: A blue button with a red '1' badge indicating one match.
- Reset**: A white button with a blue border.

Annotations on the right side of the panel:

- Document attributes**: A bracket grouping the Document.Type, Document.Date, Document.ID, and Document.Name fields.
- Keywords**: A bracket grouping the FirstName, LastName, Birthdate, Address, and Salary fields.
- Count of matching documents**: A bracket pointing to the red '1' badge on the Search button.

**Figure 4: The Search criteria panel**

Document.ID, Document.Name, Document.Date and Document.Type are displayed on the result panel, others are hidden.

```
/**
 * perit.hu
 */
```

Search criteria

Document.Type  
Document.Type\*  
PerformanceTest

Document.Date  
Document.DateStored  
Document.ID  
Document.Name  
Document.CreatedBy

Long  
Alphanumeric  
= Alphanumeric\*  
ID\_555

FloatingPoint  
DateTime  
Date

Search 4 847

Reset

Results

Id	Name	Document date	Status	Alphanumeric	Long	Actions
791	tiny	2025-01-12 12:41:44	0	ID_555	664	Download Delete Document info
3.777	tiny	2025-01-12 12:42:42	0	ID_555	138	Download Delete Document info
4.505	tiny	2025-01-12 12:42:56	0	ID_555	873	Download Delete Document info
4.700	tiny	2025-01-12 12:43:00	0	ID_555	641	Download Delete Document info
5.293	tiny	2025-01-12 12:43:12	0	ID_555	25	Download Delete Document info
5.774	tiny	2025-01-12 12:43:22	0	ID_555	744	Download Delete Document info
7.038	tiny	2025-01-12 12:43:48	0	ID_555	235	Download Delete Document info
7.707	tiny	2025-01-12 12:44:01	0	ID_555	441	Download Delete Document info
8.191	tiny	2025-01-12 12:44:10	0	ID_555	460	Download Delete Document info
9.895	tiny	2025-01-12 12:44:44	0	ID_555	149	Download Delete Document info
12.461				ID_555	344	Download Delete Document info
13.482				ID_555	572	Download Delete Document info
14.399				ID_555	338	Download Delete Document info
14.443				ID_555	86	Download Delete Document info
15.295	tiny	2025-01-12 12:46:30	0	ID_555	727	Download Delete Document info
17.115	tiny	2025-01-12 12:47:06	0	ID_555	786	Download Delete Document info
19.752	tiny	2025-01-12 12:47:59	0	ID_555		Download Delete Document info
21.762	tiny	2025-01-12 12:48:39	0	ID_555		Download Delete Document info
22.429	tiny	2025-01-12 12:48:52	0	ID_555		Download Delete Document info
23.439	tiny	2025-01-12 12:49:12	0	ID_555		Download Delete Document info
25.245	tiny	2025-01-12 12:49:48	0	ID_555	609	Download Delete Document info
26.312	tiny	2025-01-12 12:50:09	0	ID_555		Download Delete Document info
28.155	tiny	2025-01-12 12:50:46	0	ID_555		Download Delete Document info
28.495	tiny	2025-01-12 12:50:53	0	ID_555	951	Download Delete Document info
28.633	tiny	2025-01-12 12:50:55	0	ID_555	336	Download Delete Document info

Query resulted 4 847 documents in 91 ms

Items per page: 25 1 - 25 of 4847

**Figure 5: Customized result set**

The table columns can be sorted if the result set is smaller than 10.000 documents.

Document info 4 877 602

Properties

Revisions

Property	Value
Id	4 877 602
Revision	0
Name	kutya1
Document date	2021-10-15 15:11:24
Date stored	2025-01-24 07:36:06
Document type	Employee
Document type id	3
Created by	manager
Latest allowed revision id	0
File name	kutya1.jpg
File size	6 KB
Comment	NULL
Media	smb://innodex-32/smb-test
Location	2025/01/24/07/36/c402e729-243a-49c3-962d-fac0d56e394d

OK

Figure 6: Document info - properties

Document info 4 877 602

Properties

Revisions

Rev	Date stored	Created by	File name	File size	Comment	Actions
0	2025-01-24 07:36:06	manager	kutya1.jpg	6 KB	NULL	<a href="#">↓</a>

OK

Figure 7: Document info - revisions



```
/**  
 * perit.hu  
 */
```

> Details of 4877602

Keywords Content

FirstName-0  
Tim + -

FirstName-1  
Avicii + -

LastName-0  
Bergling + -

Birthdate-0  
1989. 09. 08. + -

Address-0  
1474 Blue Jay Way, Los Angeles, CA + -

Salary-0  
50.000.000 + -

Save

Add new value

Delete value

Figure 8: Changing keywords of a document

```
/**  
 * perit.hu  
 */
```

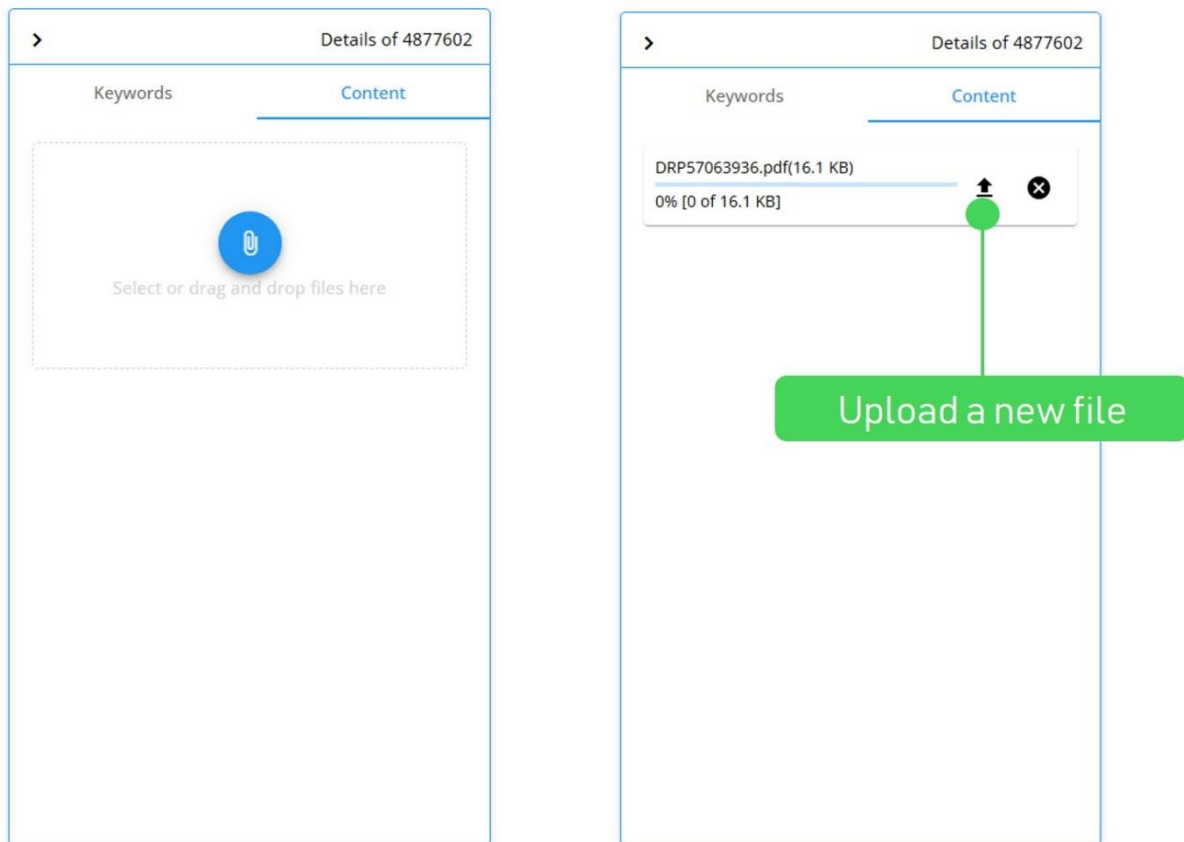


Figure 9: Updateing the content

## 4.2. THE IMPORT PAGE

```
/**
 * perit.hu
 */
```

MicroDMS 1.0.0 🔍 Retrieval 📄 Import

Document type  
Employee

Brad\_Pitt.jpg(114.2 KB)

Document date - Date  
2025. 01. 24. 📅

Time  
07:46:05

Comment

FirstName-0  
Brad + -

LastName-0  
Pitt + -

Birthdate-0  
1963. 12. 18. 📅 + -

Address-0  
9150 Wilshire Blvd. Suite 350 + -

Address-1  
Beverly Hills, CA 90212 USA + -

Salary-0 + -

Save

Cancel

Choose document type

Upload document

Specify metadata

Figure 10: Import a new file into the archive

```
/**  
 * perit.hu  
 */
```

## 5. PERFORMANCE AND VOLUME MEASURES

The performance was measured on a Windows Server 2019 computer with 6 cores and 20 GB RAM. The MicroDMS service was running in one single instance. The PostgreSQL database was running in Docker on an Ubuntu system with 4 cores and 30 GB RAM.

The load was generated by 3 JMeter scripts, each running in 10 threads for an hour. The 3 scripts are:

- createDocument() using 5 keywords and a very small text document.
- readDocument() with random docIds between 0 and 10.000.000.
- queryDocument() using 2 keywords for the query which results approximately 10-20 results.

The database contained ~12.500.000 documents at the time of testing. The web client worked fine during testing, it was responsive, one could work with it without any problems.

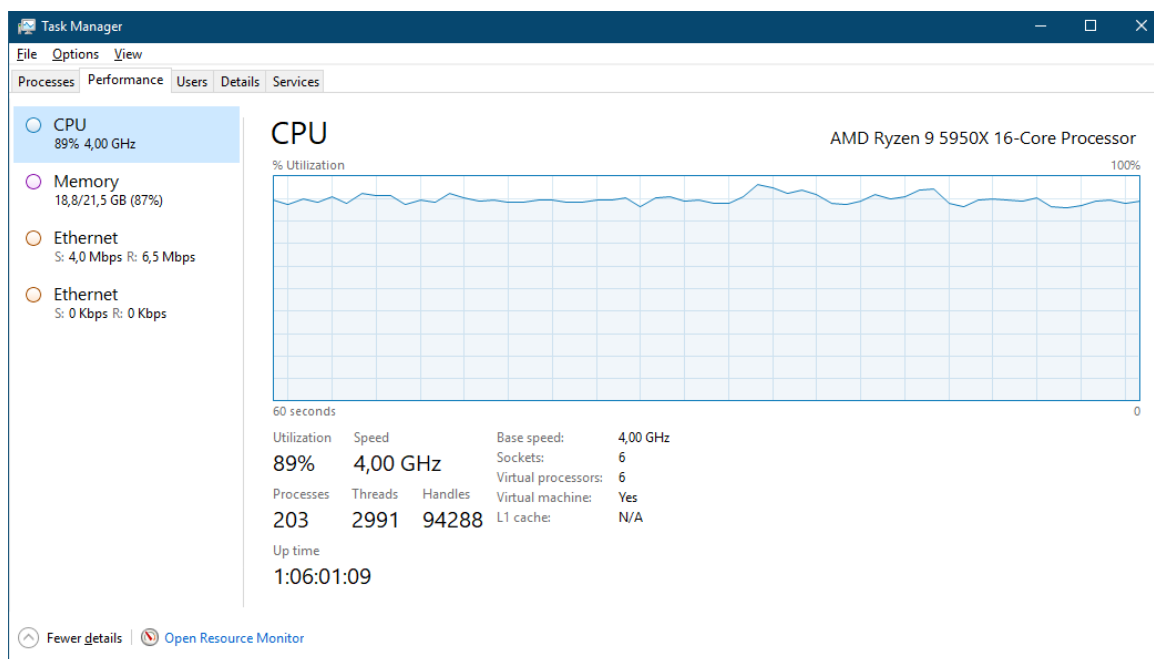
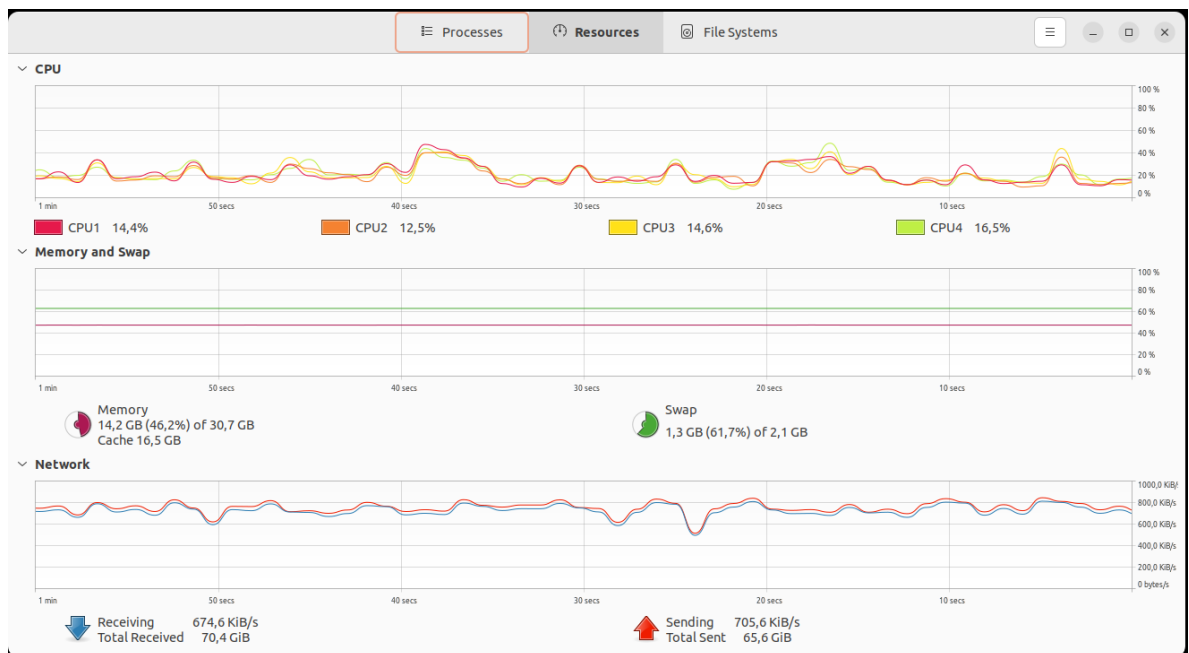
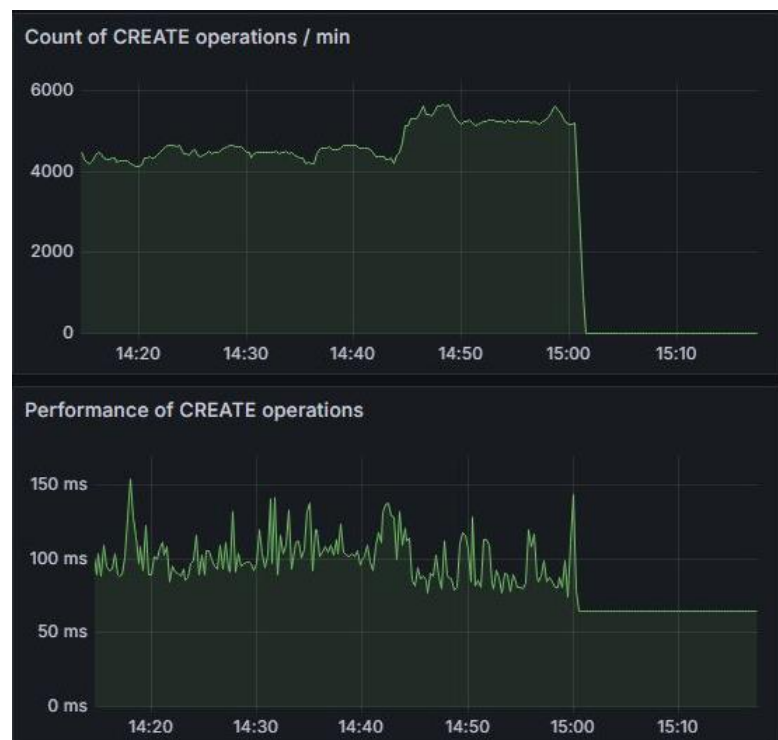


Figure 11: CPU load of the Windows system during testing



**Figure 12: Resource consumption of the Ubuntu computer hosting the database**



**Figure 13: Performance of createDocument()**

```
/**  
 * perit.hu  
 */
```



Figure 14: Performance of readDocument()

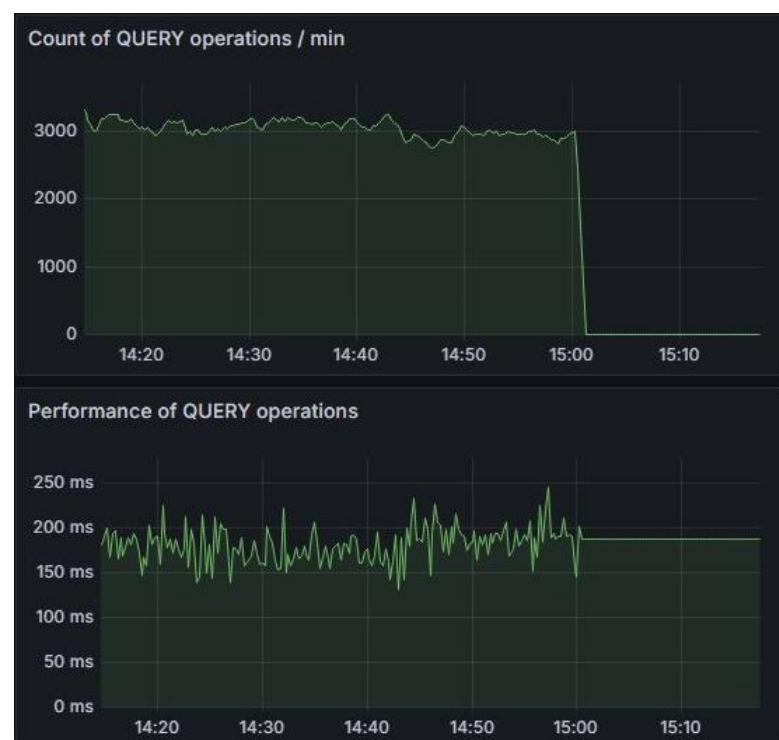


Figure 15: Performance of queryDocument()

```
/**  
 * perit.hu  
 */
```

Of course the system is horizontally scalable by running more than one instances of the service.

```
/**
 * perit.hu
 */
```

## 6. APPENDIX

### 6.1. REFERENCED DOCUMENTS

Ref	Név
1	
2	
3	

### 6.2. LIST OF FIGURES

Figure 1: Database model of the storage subsystem .....	7
Figure 2: Files on the media .....	8
Figure 3: The MicroDMS web client .....	29
Figure 4: The Search criteria panel .....	30
Figure 5: Customized result set .....	31
Figure 6: Document info - properties .....	32
Figure 7: Document info - revisions .....	32
Figure 8: Changing keywords of a document .....	33
Figure 9: Updating the content .....	34
Figure 10: Import a new file into the archive .....	35
Figure 11: CPU load of the Windows system during testing .....	36
Figure 12: Resource consumption of the Ubuntu computer hosting the database .....	37
Figure 13: Performance of createDocument() .....	37
Figure 14: Performance of readDocument() .....	38
Figure 15: Performance of queryDocument() .....	38

### 6.3. OPEN ISSUES

### 6.4. REQUIREMENT KEYS

No index entries found.