DOCUMENTACIÓN ESTRUCTURADA DE ABSTRACCIONES

A. IUserService

Endpoint:
GET /api/users - GetAllUsers
GET /api/users?id={id} - GetUserById
POST /api/users - CreateUser
PUT /api/users - UpdateUser
DELETE /api/users?id={id} - DeleteUser

Método HTTP:
GET, POST, PUT, DELETE

Formato de serialización de intercambio:
JSON

Cabeceras de entrada:
Content-Type: application/json

Cabeceras de salida:
Content-Type: application/json

Estructura de datos de entrada del endpoint:

GetAllUsers:
No requiere parámetros

GetUserById:
Parámetro de consulta: id (integer)

CreateUser:
```
{
  "id": 0,
  "name": "string",
  "email": "string"
}
```

UpdateUser:
```
{
  "id": 0,
  "name": "string",
  "email": "string"
}
```

DeleteUser:
Parámetro de consulta: id (integer)

Estructura de datos de salida del endpoint (Caso satisfactorio):

GetAllUsers:

```
{
  "users": [
    {
      "id": 1,
      "name": "string",
      "email": "string"
    }
  ]
}
```

GetUserById:

```
{
  "id": 1,
  "name": "string",
  "email": "string"
}
```

CreateUser:

```
{
  "id": 1
}
```

UpdateUser:

```
{
  "success": true
}
```

DeleteUser:

```
{
  "success": true
}
```

Estructura de datos de salida del endpoint (caso no satisfactorio):

GetUserById:

```
{
  "error": "Usuario no encontrado"
}
```

CreateUser:

```
{
  "id": 0
}
```

UpdateUser:

```
{
  "success": false
}
```

DeleteUser:

```
{
  "success": false
}
```

B. IProductCatalog

Endpoint:
GET /api/products/search?keyword={keyword} - SearchProducts
GET /api/products?id={id} - GetProductById
PUT /api/products/stock - UpdateStock

Método HTTP:
GET, PUT

Formato de serialización de intercambio:
JSON

Cabeceras de entrada:
Content-Type: application/json

Cabeceras de salida:
Content-Type: application/json

Estructura de datos de entrada del endpoint:

SearchProducts:
Parámetro de consulta: keyword (string)

GetProductById:
Parámetro de consulta: id (integer)

UpdateStock:

```
{
  "productId": 0,
  "newStock": 0
}
```

Estructura de datos de salida del endpoint (caso satisfactorio):

SearchProducts:

```
{
  "products": [
    {
```

```
      "id": 1,
      "name": "string",
      "price": 0.0
    }
  ]
}
```


GetProductById:


```
{
  "id": 1,
  "name": "string",
  "price": 0.0
}
```


UpdateStock:


```
{
  "success": true
}
```


Estructura de datos de salida del endpoint (caso no satisfactorio):

GetProductById:


```
{
  "error": "Producto no encontrado"
}
```


UpdateStock:


```
{
  "success": false
}
```


C. IAnalyticsService

Endpoint:
POST /api/analytics/track - TrackEvent
GET /api/analytics/count?eventName={eventName} - GetEventCount
GET /api/analytics/top?limit={limit} - GetTopEvents

Método HTTP:
POST, GET

Formato de serialización de intercambio:
JSON

Cabeceras de entrada:
Content-Type: application/json

Cabeceras de salida:

Content-Type: application/json

Estructura de datos de entrada del endpoint:

TrackEvent:
```
{
  "eventName": "string",
  "properties": {
    "key1": "value1",
    "key2": "value2"
  }
}
```

GetEventCount:
Parámetro de consulta: eventName (string)

GetTopEvents:
Parámetro de consulta: limit (integer)

Estructura de datos de salida del endpoint (caso satisfactorio):

TrackEvent:
```
{
  "status": "tracked"
}
```

GetEventCount:
```
{
  "eventName": "string",
  "count": 0
}
```

GetTopEvents:
```
{
  "events": {
    "event1": 150,
    "event2": 120,
    "event3": 95
  }
}
```

Estructura de datos de salida del endpoint (caso no satisfactorio):

TrackEvent:
```
{
  "error": "Error al registrar evento"
}
```

GetEventCount:
```
{
  "eventName": "string",
  "count": 0
}
```

GetTopEvents:
```
{
  "events": {}
}
```

D. IFileStorage

Endpoint:
GET /api/storage/files?directory={directory} - ListFiles
GET /api/storage/file?path={path} - ReadFile
POST /api/storage/file - WriteFile
DELETE /api/storage/file?path={path} - DeleteFile

Método HTTP:
GET, POST, DELETE

Formato de serialización de intercambio:
JSON

Cabeceras de entrada:
Content-Type: application/json

Cabeceras de salida:
Content-Type: application/json
Content-Type: text/plain (para ReadFile)

Estructura de datos de entrada del endpoint:

ListFiles:
Parámetro de consulta: directory (string)

ReadFile:
Parámetro de consulta: path (string)

WriteFile:
```
{
  "path": "string",
  "content": "string"
}
```

DeleteFile:
Parámetro de consulta: path (string)

Estructura de datos de salida del endpoint (caso satisfactorio):

ListFiles:

```
{
  "directory": "string",
  "files": [
    "file1.txt",
    "file2.txt",
    "file3.txt"
  ]
}
```

ReadFile:

```
{
  "path": "string",
  "content": "contenido del archivo"
}
```

WriteFile:

```
{
  "status": "written"
}
```

DeleteFile:

```
{
  "status": "deleted"
}
```

Estructura de datos de salida del endpoint (caso no satisfactorio):

FileNotFoundException:

```
{
  "error": "FileNotFoundException",
  "message": "Archivo no encontrado: {path}"
}
```

PermissionDeniedException:

```
{
  "error": "PermissionDeniedException",
  "message": "Permiso denegado para: {path}"
}
```

StorageException:

```
{
  "error": "StorageException",
  "message": "Error en operación de almacenamiento"
}
```

E. IAuthService

Endpoint:
POST /api/auth/login - Login
POST /api/auth/logout - Logout
GET /api/auth/validate?token={token} - ValidateToken

Método HTTP:
POST, GET

Formato de serialización de intercambio:
JSON

Cabeceras de entrada:
Content-Type: application/json

Cabeceras de salida:
Content-Type: application/json

Estructura de datos de entrada del endpoint:

Login:
```
{
  "user": "string",
  "password": "string"
}
```

Logout:
```
{
  "token": "string"
}
```

ValidateToken:
Parámetro de consulta: token (string)

Estructura de datos de salida del endpoint (caso satisfactorio):

Login:
```
{
  "token": "string"
}
```

Logout:
```
{
  "status": "logged out"
}
```

ValidateToken:

```
{
  "valid": true
}
```

Estructura de datos de salida del endpoint (caso no satisfactorio):

Login:

```
{
  "error": "AuthException",
  "message": "Auth error: {msg}"
}
```

ValidateToken:

```
{
  "valid": false
}
```