Large-Scale and Multi-Structured Databases

BioConnect

Network Analysis of Protein Interactions and Drugs Connections

Ivan Brillo
Daniel Pipitone
Andrea Bochicchio

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

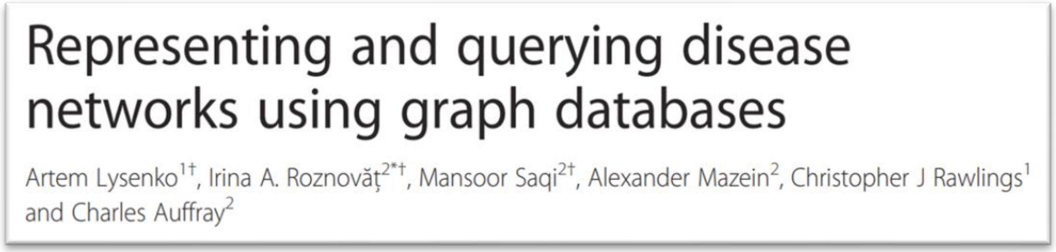
CROSSLAB Innovation for industry 4.0

Application Highlights

- **Similarity and Functional Search** (pathways and subcellular location) in Proteins
- **Analysis** and Research of **Biological Interactions** between Proteins, Diseases, and Drugs
    - **Hypothesis generation** that helps researchers to identify possible causes of a biological events
- **Trend Analysis** of Publications and Patents on Proteins or Drugs.
- Registered users (such as researchers) can provide comments on a biological molecule, for example, to suggest a correction in case of errors

*(All the biological information's are **human-related**, we don't take in consideration other spices)*

This work draws inspiration from various scientific articles. The most important:



Representing and querying disease networks using graph databases

Artem Lysenko[1†], Irina A. Roznovăţ[2*†], Mansoor Saqi[2†], Alexander Mazein[2], Christopher J Rawlings[1] and Charles Auffray[2]

https://biodatamining.biomedcentral.com/articles/10.1186/s13040-016-0102-8

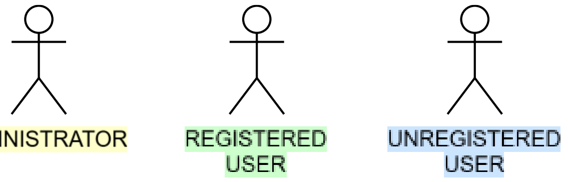
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

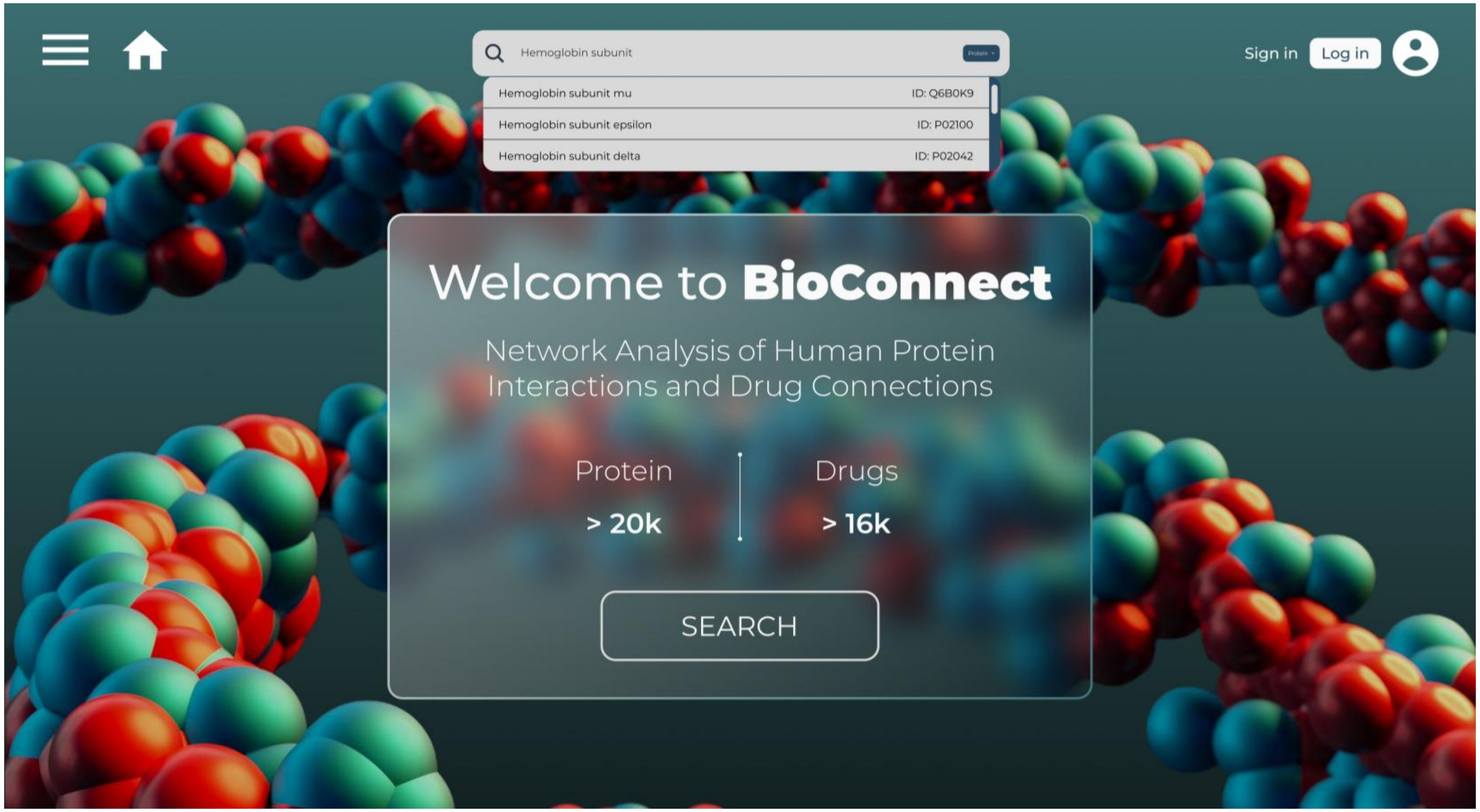
IN SUPREMÆ DIGNITATIS 1343

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Actors and main mock-ups (1)





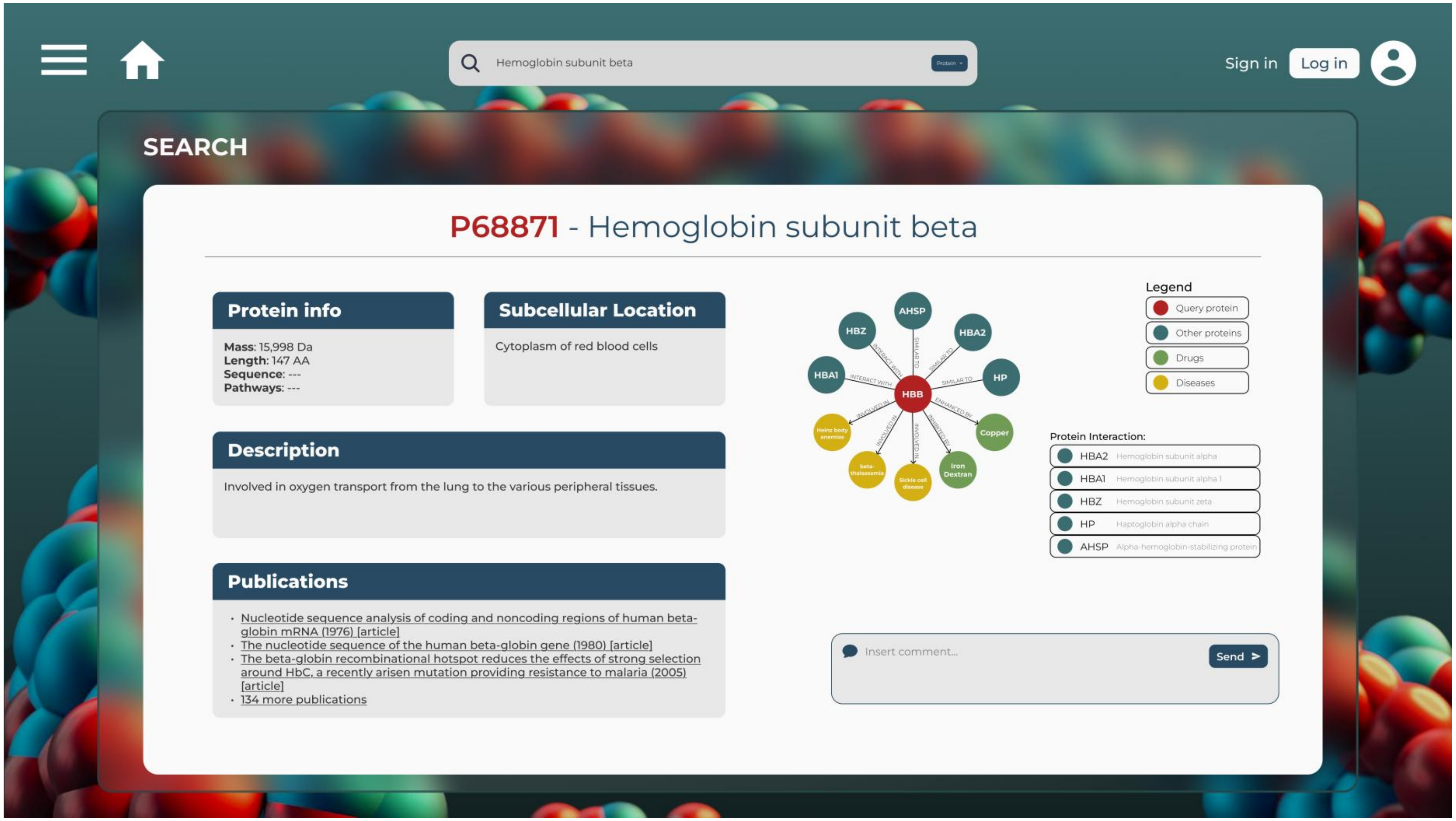
Hemoglobin subunit mu ID: Q6B0K9
Hemoglobin subunit epsilon ID: P02100
Hemoglobin subunit delta ID: P02042

Welcome to **BioConnect**

Network Analysis of Human Protein Interactions and Drug Connections

Protein	Drugs
> 20k	> 16k

Actors and main mock-ups (2)



P68871 - Hemoglobin subunit beta

Protein info

Mass: 15,998 Da
Length: 147 AA
Sequence: ---
Pathways: ---

Subcellular Location

Cytoplasm of red blood cells

Description

Involved in oxygen transport from the lung to the various peripheral tissues.

Publications

- Nucleotide sequence analysis of coding and noncoding regions of human beta-globin mRNA (1976) [article]
- The nucleotide sequence of the human beta-globin gene (1980) [article]
- The beta-globin recombinational hotspot reduces the effects of strong selection around HbC, a recently arisen mutation providing resistance to malaria (2005) [article]
- 134 more publications

Legend

🟥	Query protein
●	Other proteins
🟩	Drugs
🟨	Diseases

Protein Interaction:

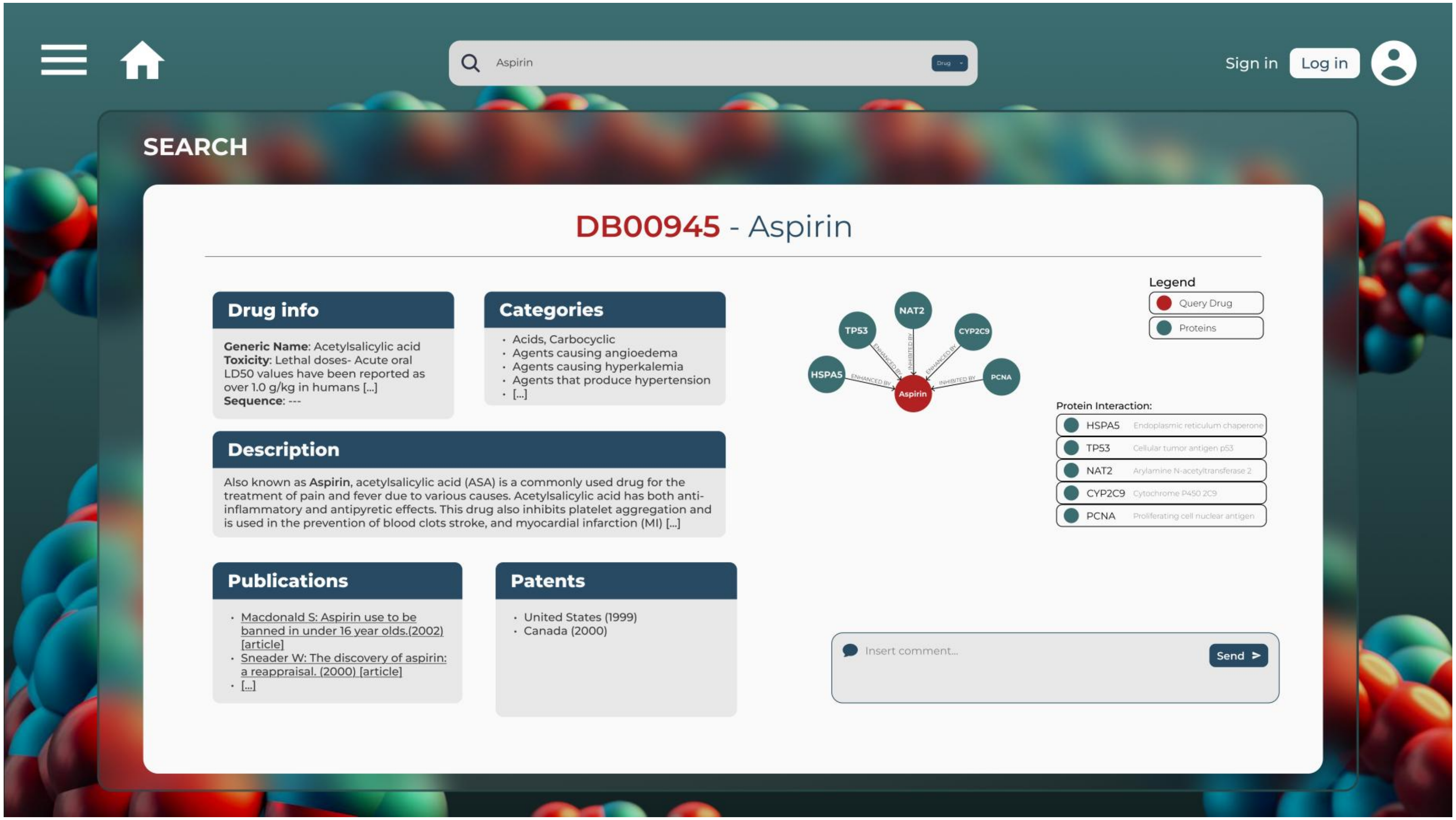
●	HBA2	Hemoglobin subunit alpha
●	HBA1	Hemoglobin subunit alpha 1
●	HBZ	Hemoglobin subunit zeta
●	HP	Haptoglobin alpha chain
●	AHSP	Alpha-hemoglobin-stabilizing protein

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

IN SUPREMÆ DIGNITATIS 1343
UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Actors and main mock-ups (3)



DB00945 - Aspirin

Drug info

Generic Name: Acetylsalicylic acid
Toxicity: Lethal doses- Acute oral LD50 values have been reported as over 1.0 g/kg in humans [...]
Sequence: ---

Categories

- Acids, Carbocyclic
- Agents causing angioedema
- Agents causing hyperkalemia
- Agents that produce hypertension
- [...]

Description

Also known as **Aspirin**, acetylsalicylic acid (ASA) is a commonly used drug for the treatment of pain and fever due to various causes. Acetylsalicylic acid has both anti-inflammatory and antipyretic effects. This drug also inhibits platelet aggregation and is used in the prevention of blood clots stroke, and myocardial infarction (MI) [...]

Publications

- Macdonald S: Aspirin use to be banned in under 16 year olds.(2002) [article]
- Sneader W: The discovery of aspirin: a reappraisal. (2000) [article]
- [...]

Patents

- United States (1999)
- Canada (2000)

Legend

🟥	Query Drug
●	Proteins

Protein Interaction:

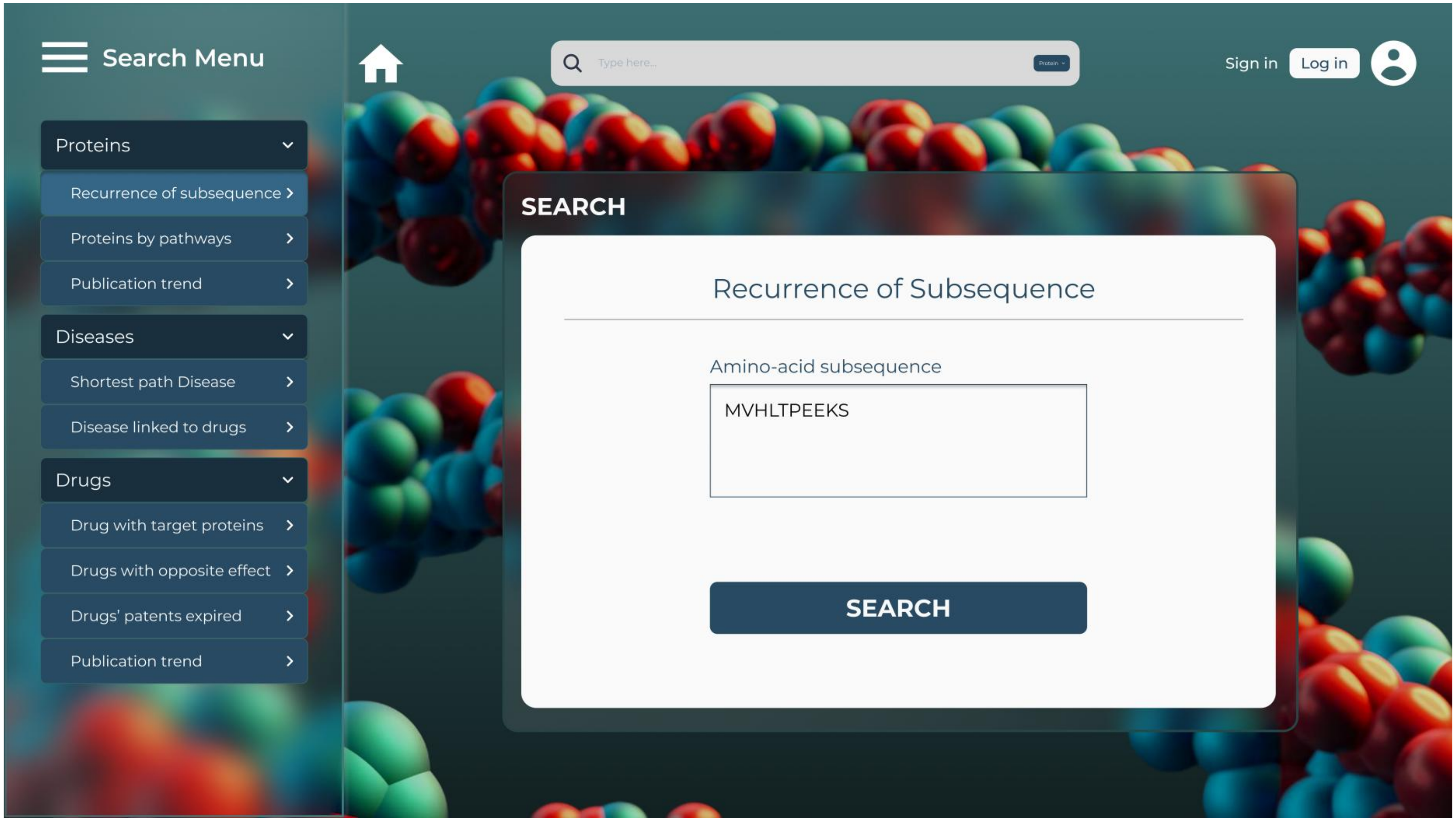
●	HSPA5	Endoplasmic reticulum chaperone
●	TP53	Cellular tumor antigen p53
●	NAT2	Arylamine N-acetyltransferase 2
●	CYP2C9	Cytochrome P450 2C9
●	PCNA	Proliferating cell nuclear antigen

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

IN SUPREMÆ DIGNITATIS 1343
UNIVERSITÀ DI PISA

CROSSLAB
Innovation for industry 4.0

Actors and main mock-ups (4)

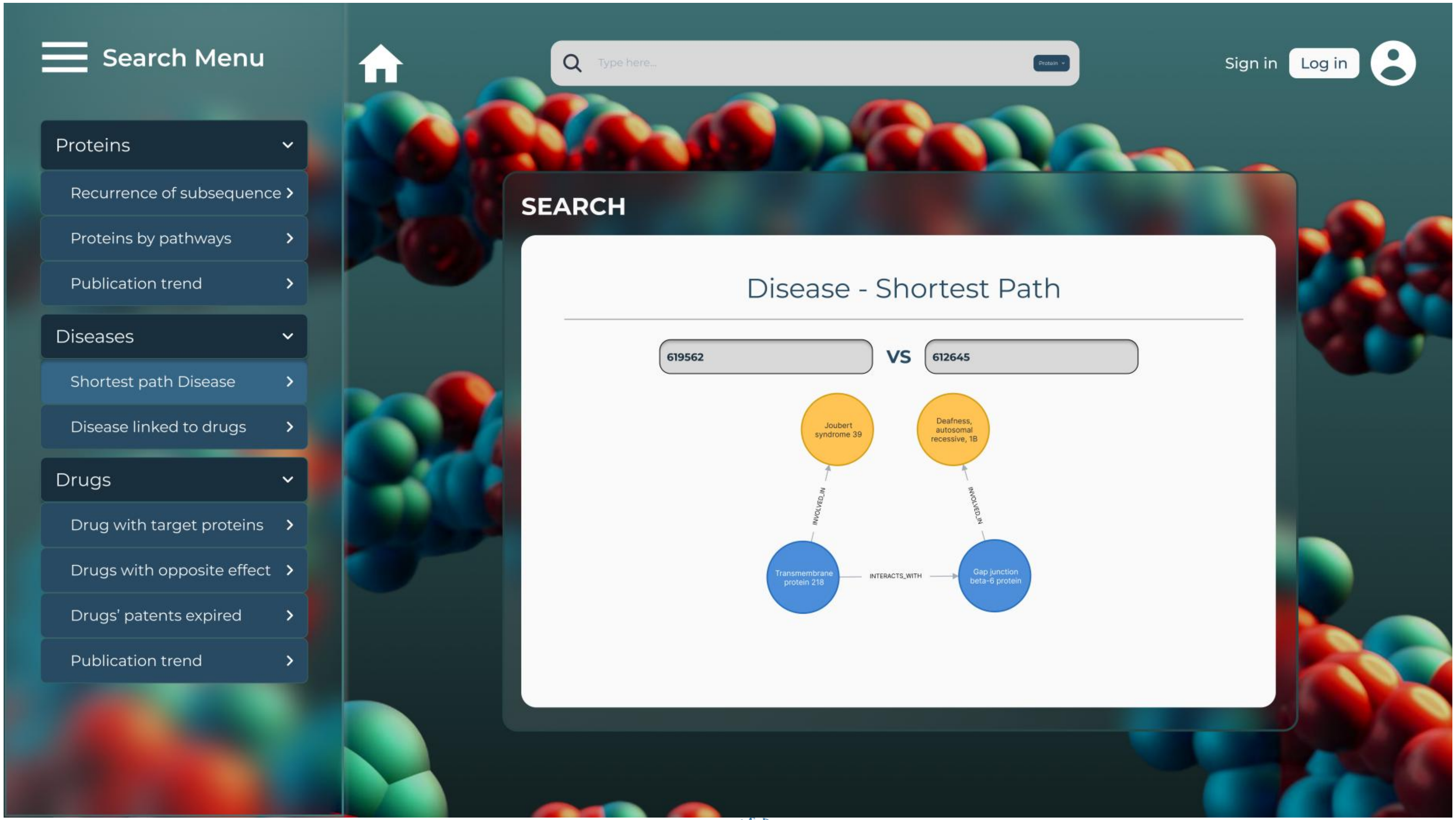


DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

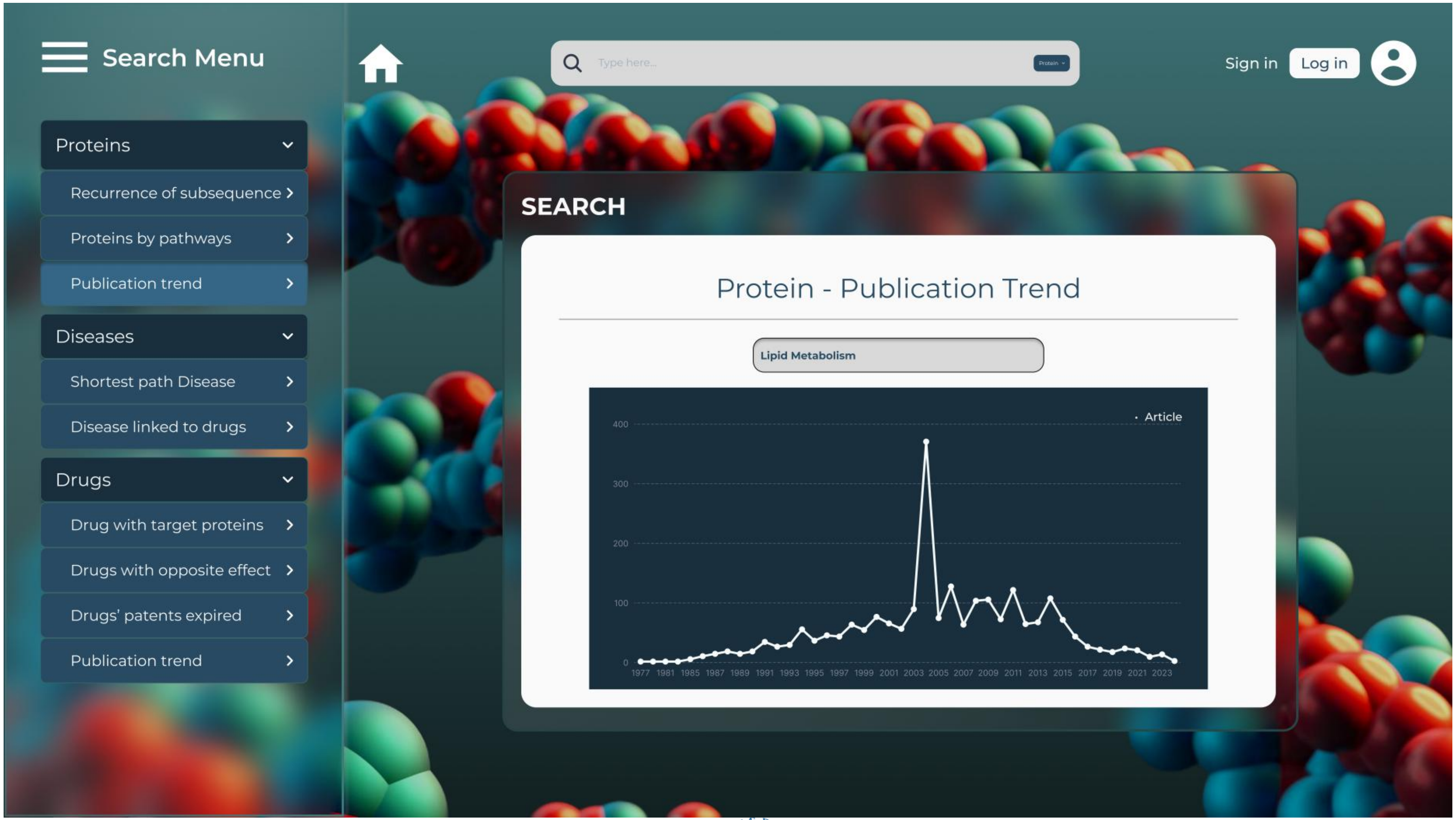
Actors and main mock-ups (6)



619562

612645

Actors and main mock-ups (5)

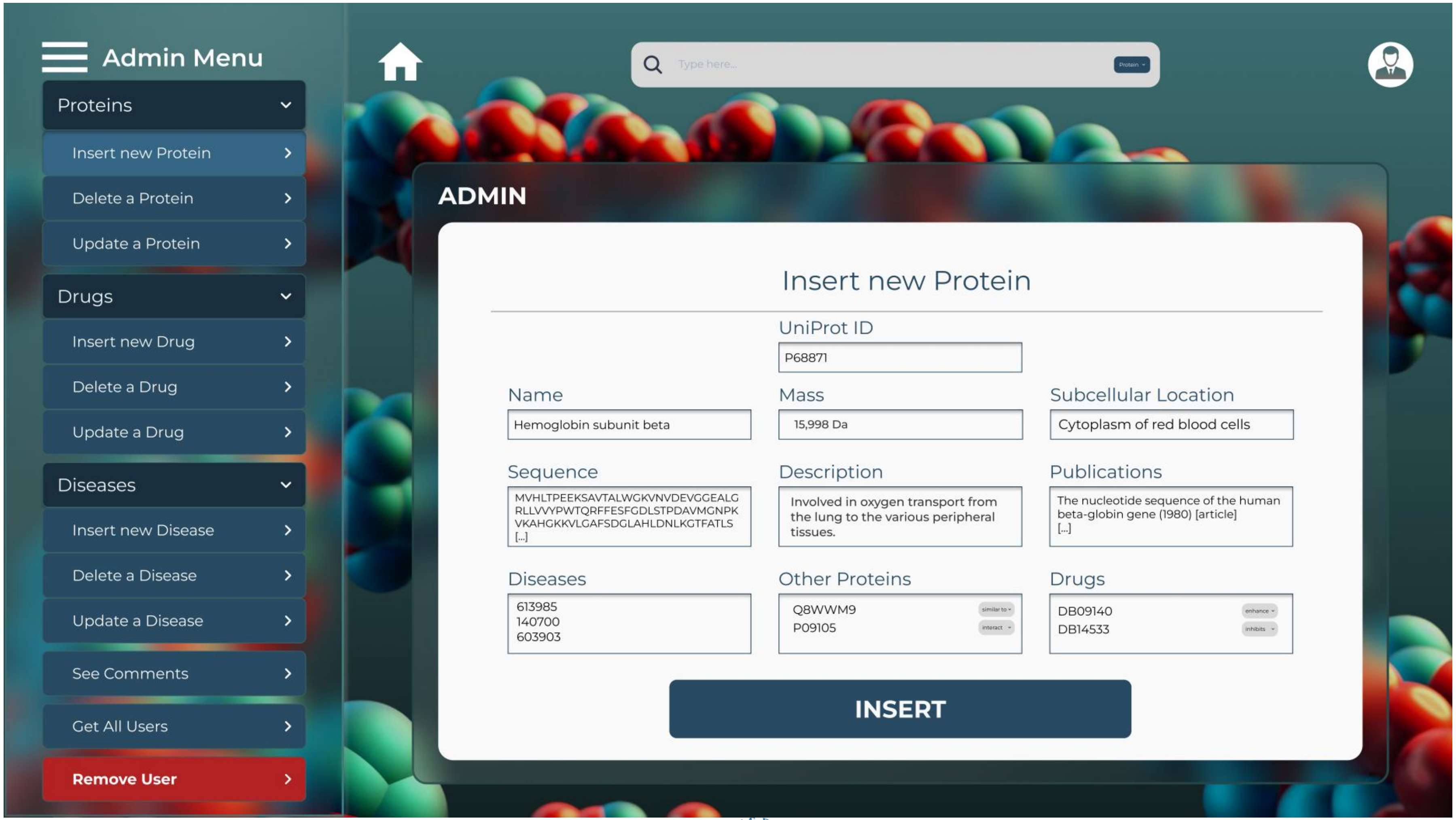


DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

IN SUPREMÆ DIGNITATIS 1343
UNIVERSITÀ DI PISA

CROSSLAB
Innovation for industry 4.0

Actors and main mock-ups (7)



Insert new Protein

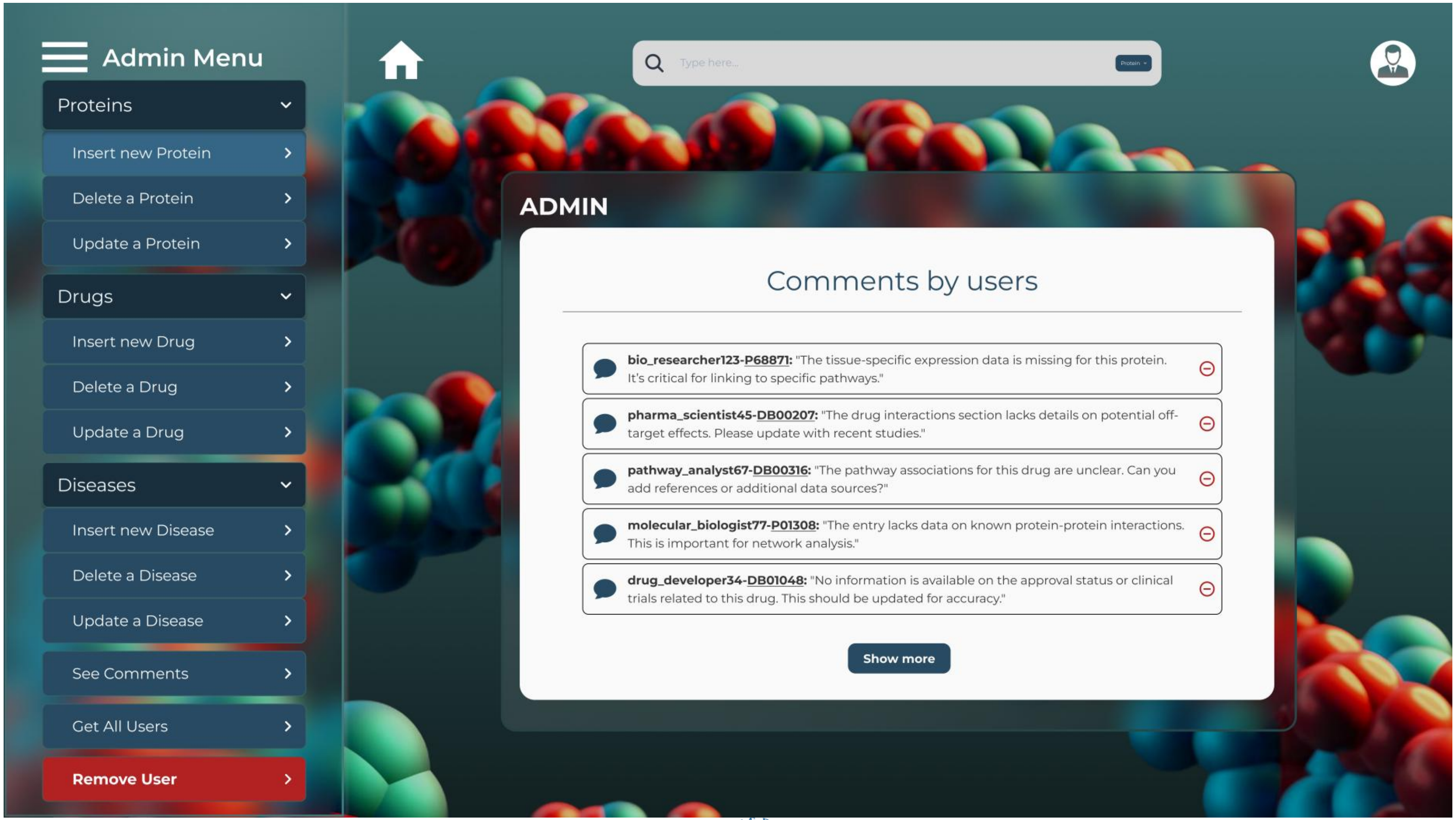
	UniProt ID: P68871	
Name: Hemoglobin subunit beta	Mass: 15,998 Da	Subcellular Location: Cytoplasm of red blood cells
Sequence: MVHLTPEEKSAVTALWGKVNVDEVGGEALG RLLVVYPWTQRFFESFGDLSTPDAVMGNPK VKAHGKKVLGAFSDGLAHLDNLKGTFATLS [...]	Description: Involved in oxygen transport from the lung to the various peripheral tissues.	Publications: The nucleotide sequence of the human beta-globin gene (1980) [article] [...]
Diseases: 613985 140700 603903	Other Proteins: Q8WWM9 similar to; P09105 interact	Drugs: DB09140 enhance; DB14533 inhibits

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

IN SUPREMÆ DIGNITATIS 1343 UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Actors and main mock-ups (8)



Comments by users

bio_researcher123-P68871: "The tissue-specific expression data is missing for this protein. It's critical for linking to specific pathways."

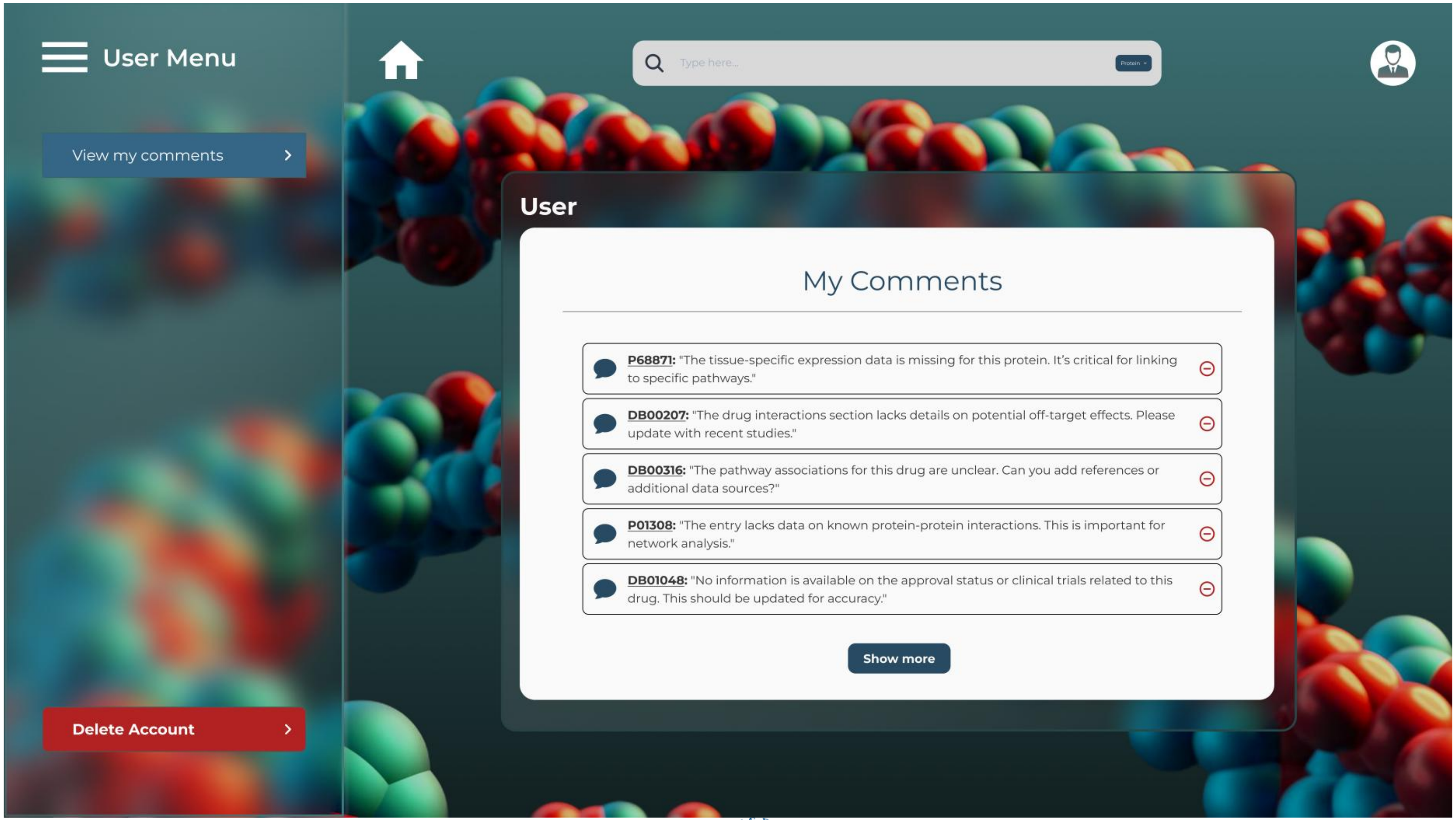
pharma_scientist45-DB00207: "The drug interactions section lacks details on potential off-target effects. Please update with recent studies."

pathway_analyst67-DB00316: "The pathway associations for this drug are unclear. Can you add references or additional data sources?"

molecular_biologist77-P01308: "The entry lacks data on known protein-protein interactions. This is important for network analysis."

drug_developer34-DB01048: "No information is available on the approval status or clinical trials related to this drug. This should be updated for accuracy."

Actors and main mock-ups (9)



My Comments

P68871: "The tissue-specific expression data is missing for this protein. It's critical for linking to specific pathways."

DB00207: "The drug interactions section lacks details on potential off-target effects. Please update with recent studies."

DB00316: "The pathway associations for this drug are unclear. Can you add references or additional data sources?"

P01308: "The entry lacks data on known protein-protein interactions. This is important for network analysis."

DB01048: "No information is available on the approval status or clinical trials related to this drug. This should be updated for accuracy."

Dataset Description

Sources: *Uniprot, DrugBank and DisGeNET*

Description: Uniprot *contains information about proteins and their relations with other proteins. We used* ***DrugBank*** *to point out connection's drug-protein and drug-drug. Finally,* ***DisGeNET*** *is used to understand the connections between protein and diseases. To get the* publication *date of scientific materials we used the* ***National Library of Medicine*** *API*

Pre-processing: Raw data were in various formats : Uniprot (CSV), DrugBank (XML), and DisGeNET (JSON). Python scripts were developed to clean and filter the data, converting all files to JSON for MongoDB collections. Further scripts integrated the data, generating CSV files for populating the Neo4j Graph database

Dataset Description (2)

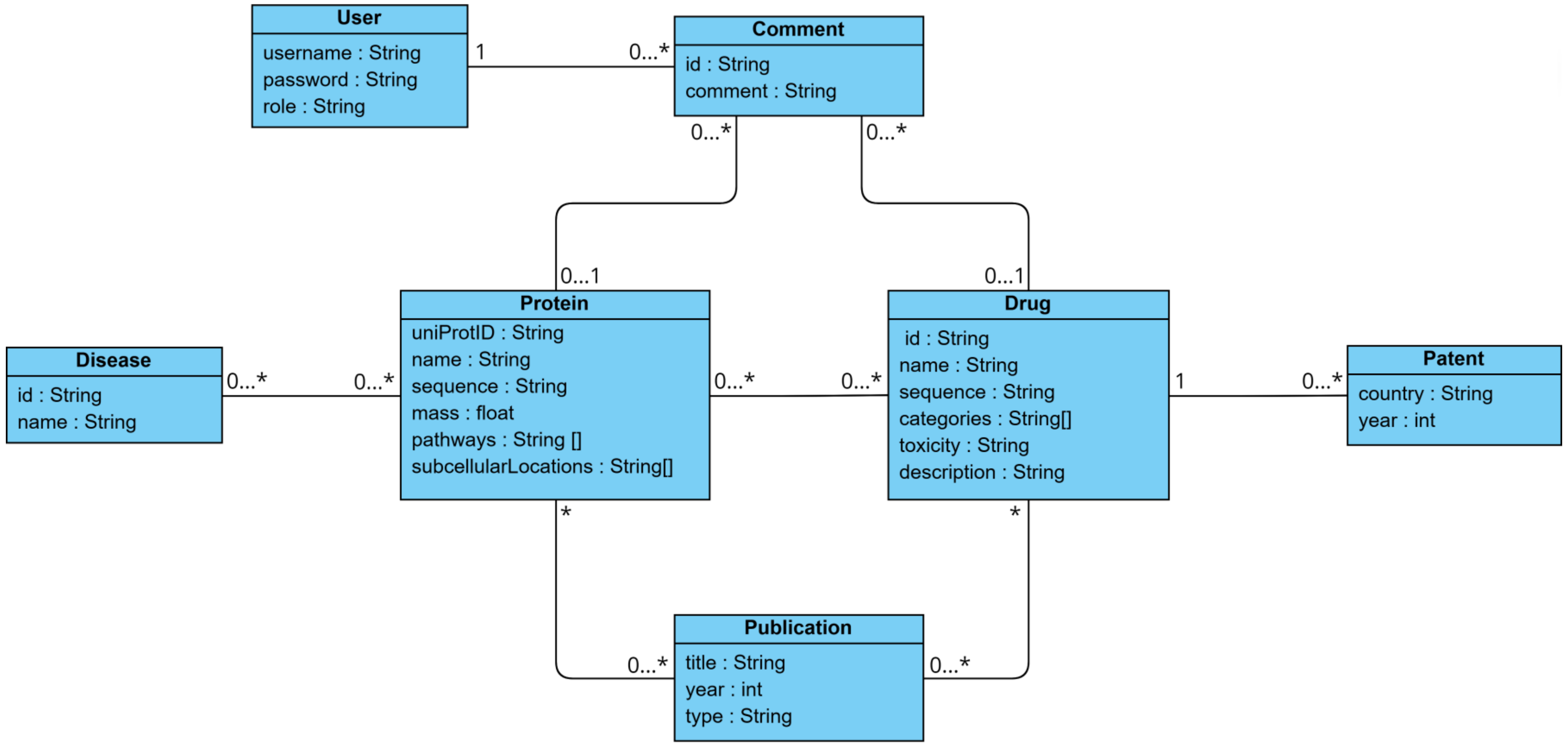
Volume:

MongoDB (42 MB)	Neo4j CSV File (15MB)
• 7MB for Drug collection • 34MB for Protein collection • 1MB for User collection	• 1MB for nodes data • 14MB for relationships data

Variety: High data variety coming from different biological databases

Velocity/Variability: Low data velocity and variability

UML Class Diagram



DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Application non-functional requirements

Performance

- The system must process complex queries on large biological datasets with high efficiency
- The system must support concurrent user queries without significant performance degradation
- The system must be highly available and fault tolerant

Reliability

- The system must prevent permanent data loss

Usability

- The user interface must be intuitive and easy to navigate

Security

- Critical system features must ensure efficient and reliable access by requiring role authentication for users
- The system must encrypt the user password

Extensibility

- The system architecture must support easy addition of new biological entities and the extension of the current ones

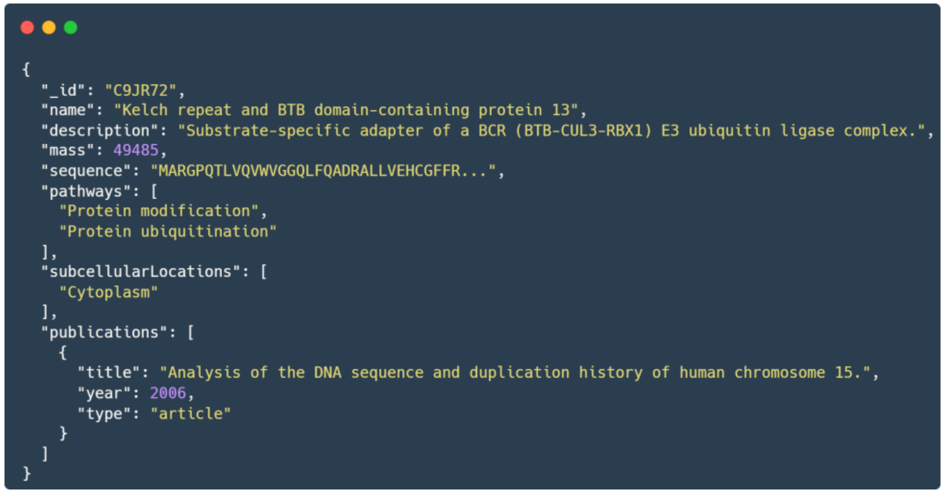
Error Handling

- Error messages must be clear, descriptive, and user-friendly

Document DB Design (1)

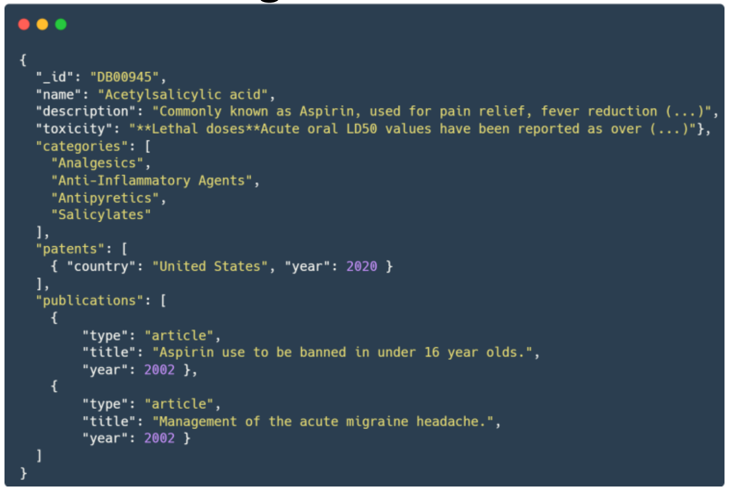
Collections

Protein's collection



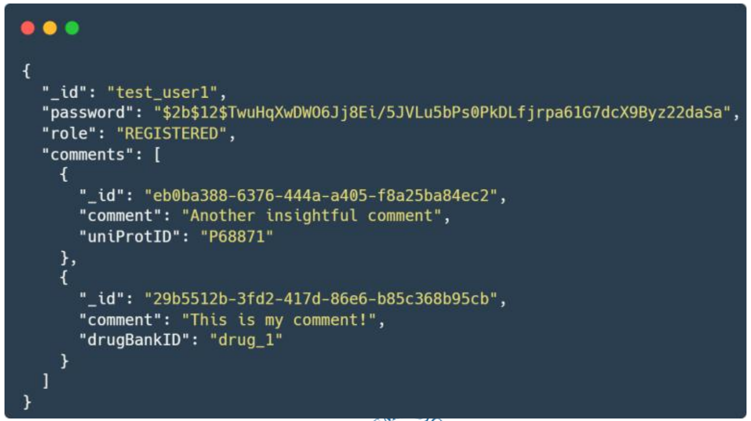
```
{
  "_id": "C9JR72",
  "name": "Kelch repeat and BTB domain-containing protein 13",
  "description": "Substrate-specific adapter of a BCR (BTB-CUL3-RBX1) E3 ubiquitin ligase complex.",
  "mass": 49485,
  "sequence": "MARGPQTLVQVWVGGQLFQADRALLVEHCGFFR...",
  "pathways": [
    "Protein modification",
    "Protein ubiquitination"
  ],
  "subcellularLocations": [
    "Cytoplasm"
  ],
  "publications": [
    {
      "title": "Analysis of the DNA sequence and duplication history of human chromosome 15.",
      "year": 2006,
      "type": "article"
    }
  ]
}
```

Drug's collection



```
{
  "_id": "DB00945",
  "name": "Acetylsalicylic acid",
  "description": "Commonly known as Aspirin, used for pain relief, fever reduction (...)",
  "toxicity": "**Lethal doses**Acute oral LD50 values have been reported as over (...)"},
  "categories": [
    "Analgesics",
    "Anti-Inflammatory Agents",
    "Antipyretics",
    "Salicylates"
  ],
  "patents": [
    { "country": "United States", "year": 2020 }
  ],
  "publications": [
    {
        "type": "article",
        "title": "Aspirin use to be banned in under 16 year olds.",
        "year": 2002 },
    {
        "type": "article",
        "title": "Management of the acute migraine headache.",
        "year": 2002 }
  ]
}
```

User's collection



```
{
  "_id": "test_user1",
  "password": "$2b$12$TwuHqXwDWO6Jj8Ei/5JVLu5bPs0PkDLfjrpa61G7dcX9Byz22daSa",
  "role": "REGISTERED",
  "comments": [
    {
      "_id": "eb0ba388-6376-444a-a405-f8a25ba84ec2",
      "comment": "Another insightful comment",
      "uniProtID": "P68871"
    },
    {
      "_id": "29b5512b-3fd2-417d-86e6-b85c368b95cb",
      "comment": "This is my comment!",
      "drugBankID": "drug_1"
    }
  ]
}
```

Document DB Design (2)

Relevant Queries

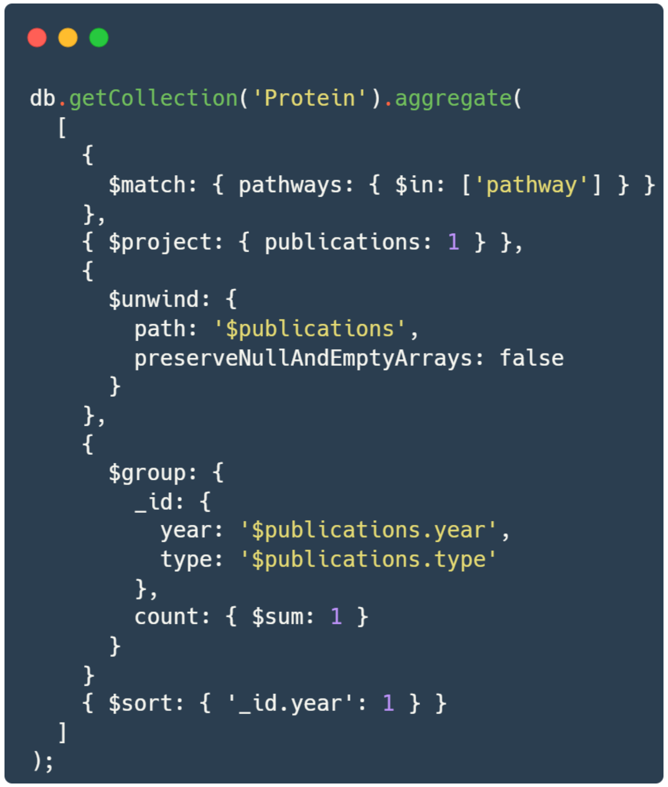
Aggregations:

- **Trend Analysis of Publications**

Identifies trends in the number of publications related to a specific pathway

Given a pathway (**Glycolysis**) found for each year the distribution of publications per type (articles/books/URL)

- Helps identify which pathways are gaining or losing research interest over time



```
db.getCollection('Protein').aggregate(
  [
    {
      $match: { pathways: { $in: ['pathway'] } }
    },
    { $project: { publications: 1 } },
    {
      $unwind: {
        path: '$publications',
        preserveNullAndEmptyArrays: false
      }
    },
    {
      $group: {
        _id: {
          year: '$publications.year',
          type: '$publications.type'
        },
        count: { $sum: 1 }
      }
    }
    { $sort: { '_id.year': 1 } }
  ]
);
```

Document DB Design (3)

Relevant Queries

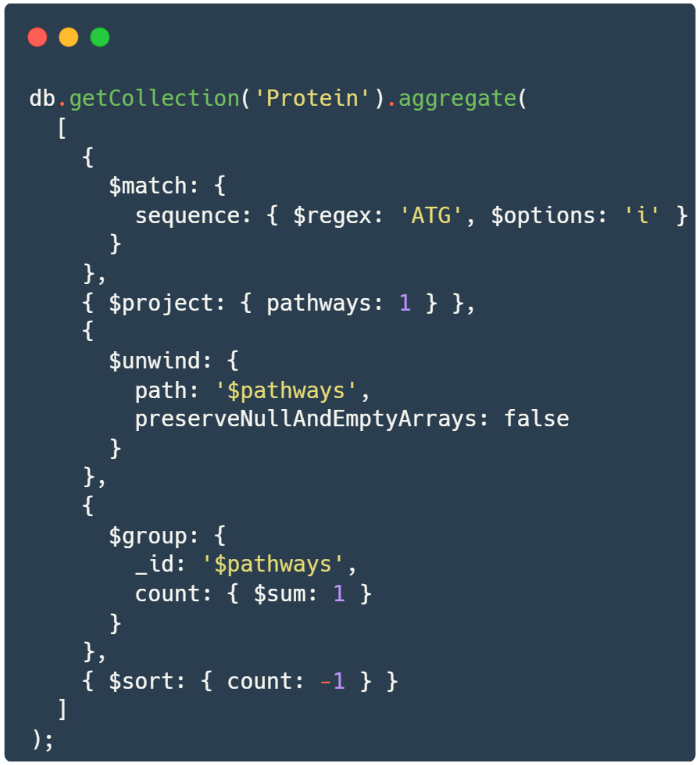
Aggregations:

- **Pathway Recurrence Analysis**

Analyzes the recurrence of specific pathways in protein sequences

Given Ubiquitin-Interacting Motif (functional subunit **ATG**) found the number of its recurrence in different pathways

- Identifying their recurrence across pathways can highlight cross-pathway interactions during proteins creations (drugs)



```
db.getCollection('Protein').aggregate(
  [
    {
      $match: {
        sequence: { $regex: 'ATG', $options: 'i' }
      }
    },
    { $project: { pathways: 1 } },
    {
      $unwind: {
        path: '$pathways',
        preserveNullAndEmptyArrays: false
      }
    },
    {
      $group: {
        _id: '$pathways',
        count: { $sum: 1 }
      }
    },
    { $sort: { count: -1 } }
  ]
);
```

Document DB Design (4)

Relevant Queries

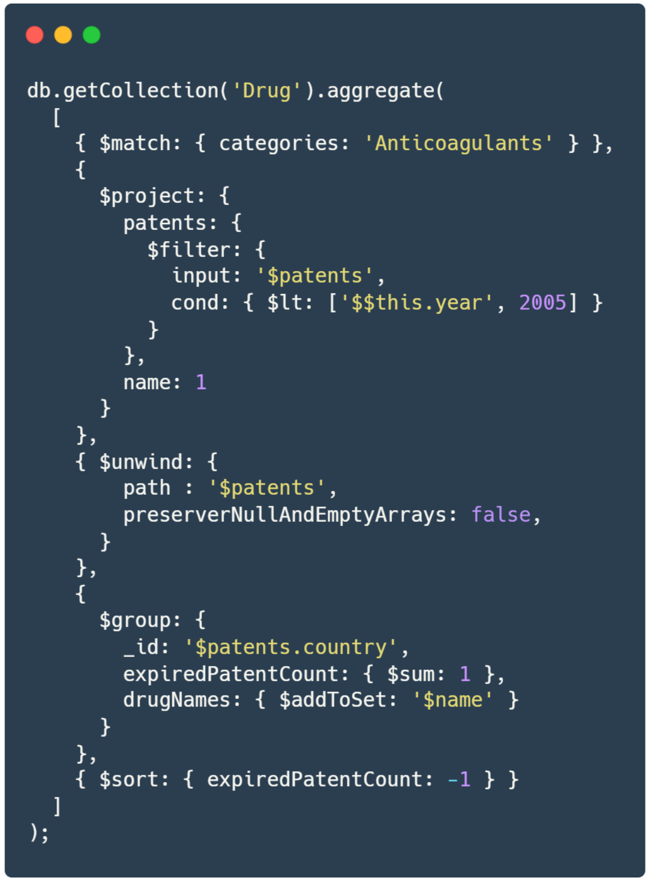
Aggregations:

- **Expired Patents Analysis**

Analyzes expired patents by country for a specific drug category

Given the **anticoagulant** category, found the expired patents per Country

- Companies can identify opportunities to bring new drugs to market



```
db.getCollection('Drug').aggregate(
  [
    { $match: { categories: 'Anticoagulants' } },
    {
      $project: {
        patents: {
          $filter: {
            input: '$patents',
            cond: { $lt: ['$$this.year', 2005] }
          }
        },
        name: 1
      }
    },
    { $unwind: {
        path : '$patents',
        preserverNullAndEmptyArrays: false,
      }
    },
    {
      $group: {
        _id: '$patents.country',
        expiredPatentCount: { $sum: 1 },
        drugNames: { $addToSet: '$name' }
      }
    },
    { $sort: { expiredPatentCount: -1 } }
  ]
);
```

MongoDB Replica Set Configuration

`w=1`	Writes are acknowledged by one node
`journal=true`	Writes are persisted to disk-based
`readPreference=nearest`	Reads from the nearest replica

How these configurations support non-functional requirements?

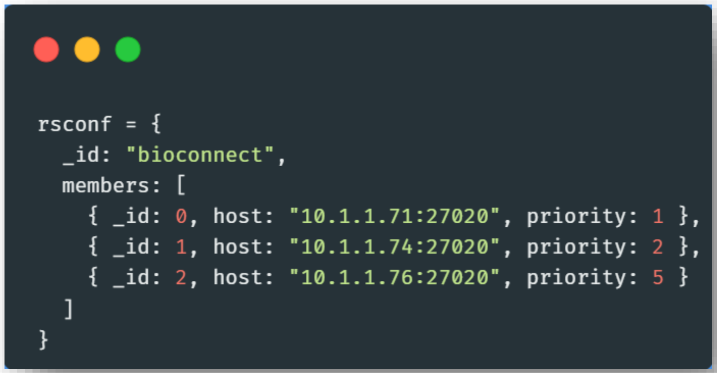
`w=1` and **`readPreference=nearest`** are essential to ensure high availability. Consistency issues are minimal and do not have a significant impact on the system, especially given the low frequency of writes

`journal=true` prevent permanent data loss

IP: 10.1.1.76
mongoDB: PRIMARY

IP: 10.1.1.74
mongoDB: SECONDARY

IP: 10.1.1.71
mongoDB: SECONDARY



```
rsconf = {
  _id: "bioconnect",
  members: [
    { _id: 0, host: "10.1.1.71:27020", priority: 1 },
    { _id: 1, host: "10.1.1.74:27020", priority: 2 },
    { _id: 2, host: "10.1.1.76:27020", priority: 5 }
  ]
}
```

MongoDB Indexes

COLLECTION	FIELD	TYPE
Protein	_id	Regular (default)
Protein	name	Text
Protein	pathways	Regular
Drug	_id	Regular (default)
Drug	name	Text
Drug	categories	Regular

PROTEIN COLLECTION

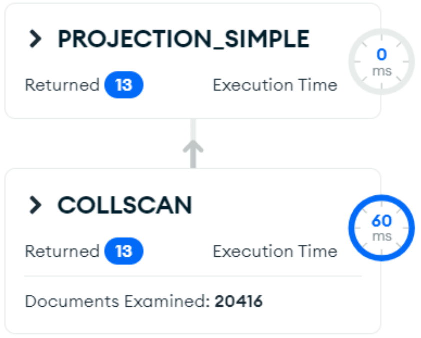
- **Search Protein**
  before: 20416 document examinated, 19ms
  after: 2 document examinated, 0ms
- **Trend Analysis of Publications for Pathway**
  before: 20416 document examinated, 64ms
  after: 13 document examinated, 2ms

DRUG COLLECTION

- **Search Drug**
  before: 16581 document examinated, 17ms
  after: 2 document examinated, 0ms
- **Expired Patents Analysis For Category**
  before: 16581 document examinated, 37ms
  after: 455 document examinated, 3ms

MongoDB Indexes(2)

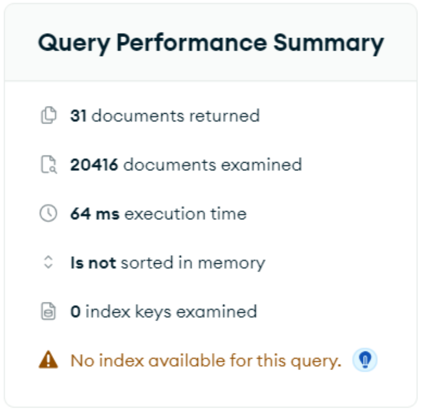
Trend Analysis of Publications for Pathway



> PROJECTION_SIMPLE
Returned 13 Execution Time 0 ms

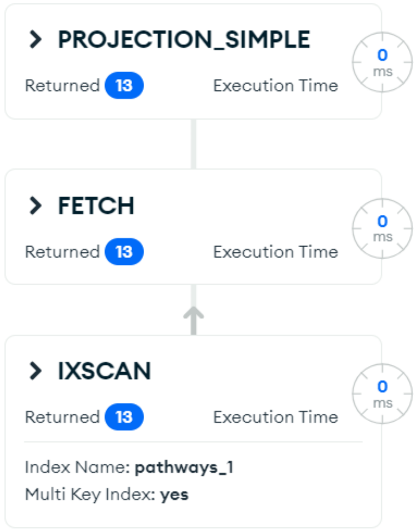
> COLLSCAN
Returned 13 Execution Time 60 ms
Documents Examined: **20416**

BEFORE
20416 document examinated, 64ms



Query Performance Summary

- **31** documents returned
- **20416** documents examined
- **64 ms** execution time
- **Is not** sorted in memory
- **0** index keys examined
- No index available for this query.

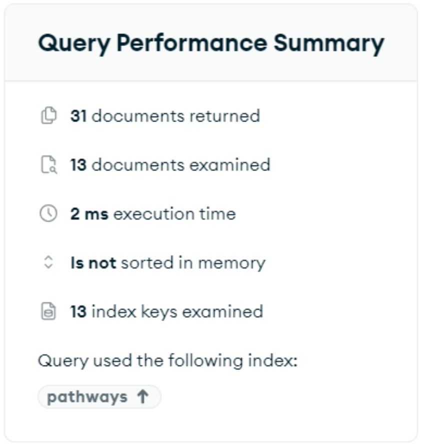


> PROJECTION_SIMPLE
Returned 13 Execution Time 0 ms

> FETCH
Returned 13 Execution Time 0 ms

> IXSCAN
Returned 13 Execution Time 0 ms
Index Name: **pathways_1**
Multi Key Index: **yes**

AFTER
13 document examinated, 2ms



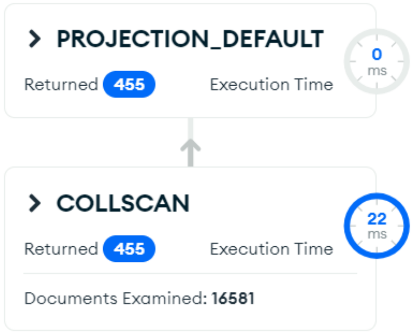
Query Performance Summary

- **31** documents returned
- **13** documents examined
- **2 ms** execution time
- **Is not** sorted in memory
- **13** index keys examined

Query used the following index:
pathways ↑

MongoDB Indexes(3)

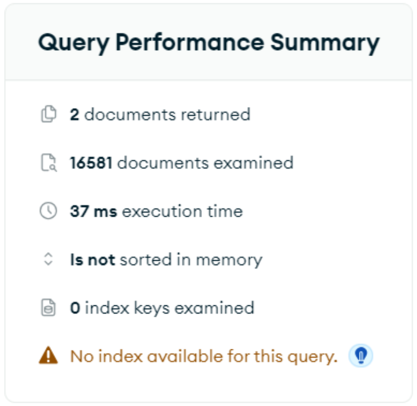
Expired Patents Analysis For Category



> PROJECTION_DEFAULT
Returned 455 Execution Time 0 ms

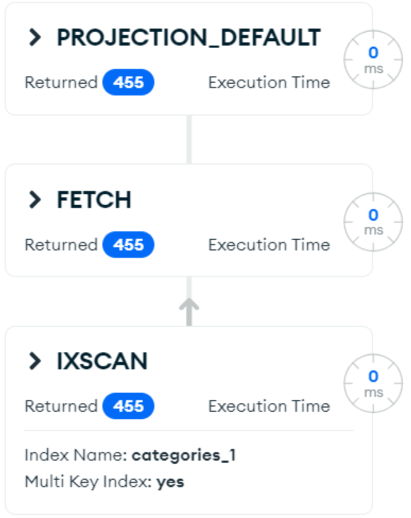
> COLLSCAN
Returned 455 Execution Time 22 ms
Documents Examined: **16581**

BEFORE
16581 document examinated, 37ms



Query Performance Summary

- **2** documents returned
- **16581** documents examined
- **37 ms** execution time
- **Is not** sorted in memory
- **0** index keys examined
- No index available for this query.

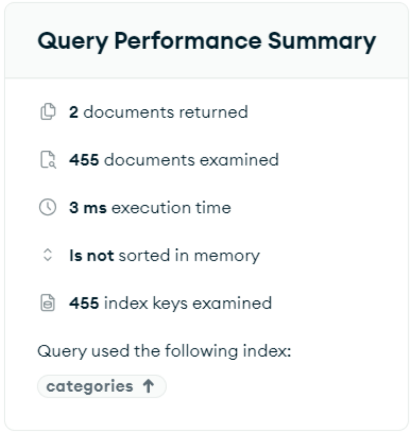


> PROJECTION_DEFAULT
Returned 455 Execution Time 0 ms

> FETCH
Returned 455 Execution Time 0 ms

> IXSCAN
Returned 455 Execution Time 0 ms
Index Name: **categories_1**
Multi Key Index: **yes**

AFTER
455 document examinated, 3ms



Query Performance Summary

- **2** documents returned
- **455** documents examined
- **3 ms** execution time
- **Is not** sorted in memory
- **455** index keys examined

Query used the following index:
categories ↑

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Discussion on MongoDB Data Sharding

How could sharding have been implemented in MongoDB?

To implement sharding, database designers must select a shard key, which determines how the data is distributed across the various shards.

In this case, the shard keys that could be chosen for each collection are:

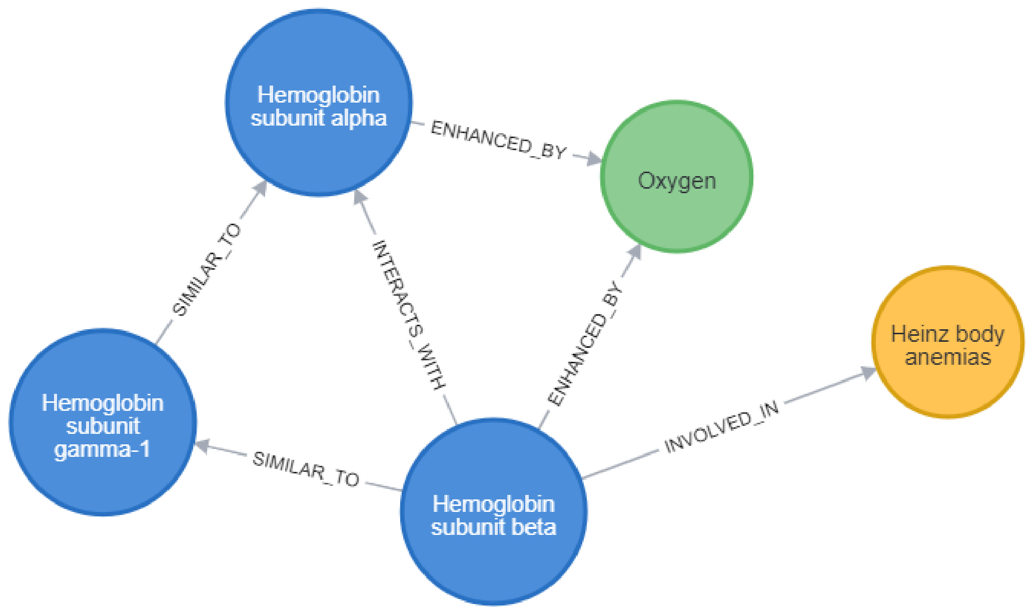
`Drug: _id, Protein: _id and User: _id`.

Regarding the partitioning algorithm, the **hashed algorithm** could be chosen in all cases since all keys are alphanumeric strings for identification.

Why sharding on our application wouldn't be beneficial?

- **No Heavy Write Load**: Our application primarily involves read operations, with no significant write workload, making sharding for load distribution unnecessary, since we can use replicas to spread the reads on different nodes.
- **Limited Data Growth**: The number of discoverable biological data in the coming decades is finite, so we don't expect the needing of sharding for scalability.
- **Stable Traffic Patterns**: We do not expect sudden spikes or high variability in user requests, reducing the need for sharding to handle application flexibility.
- **Efficient Single-Node Performance**: A single-node write setup is sufficient to manage the current and expected data volume, avoiding the added complexity of sharding and its overhead in managing multi-document requests.

Graph DB Design



Entities:

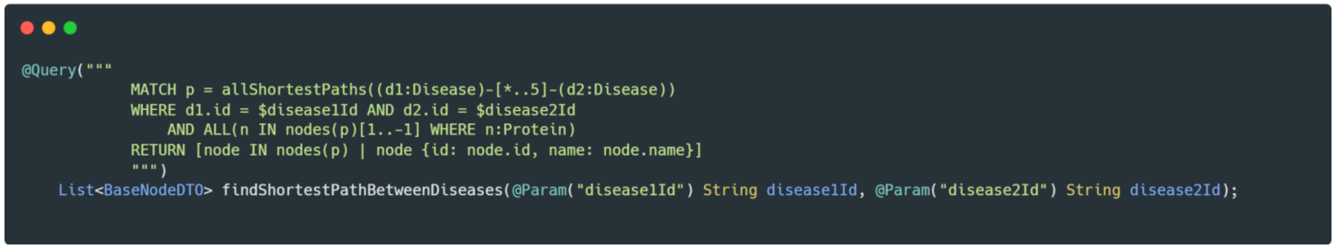
- Protein (id, name)
- Drug (id, name)
- Disease (id, name)

Relationships

- `INTERACTS_WITH` (Protein-Protein)
- `SIMILAR_TO` (Protein-Protein)
- `INVOLVED_IN` (Protein-Disease)
- `ENHANCED_BY` (Protein-Drug)
- `INHIBITED_BY` (Protein-Drug)

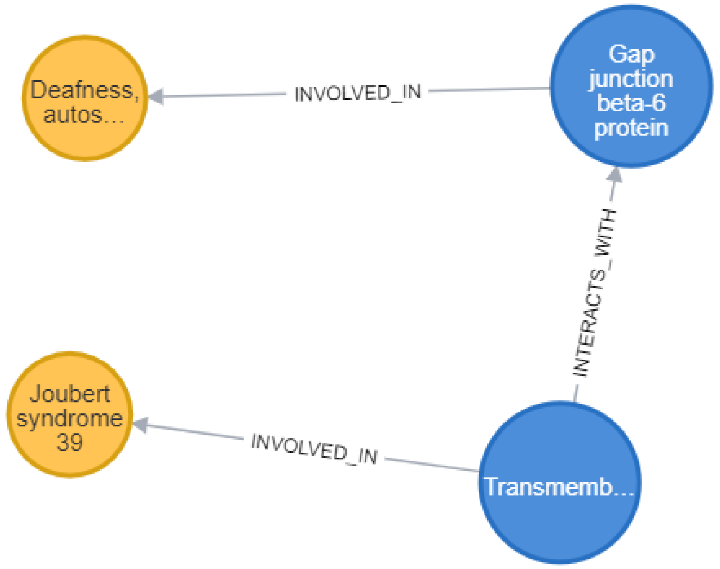
Graph-based queries (1)

Find Shortest Path Between Diseases



```
@Query("""
            MATCH p = allShortestPaths((d1:Disease)-[*..5]-(d2:Disease))
            WHERE d1.id = $disease1Id AND d2.id = $disease2Id
                AND ALL(n IN nodes(p)[1..-1] WHERE n:Protein)
            RETURN [node IN nodes(p) | node {id: node.id, name: node.name}]
            """)
    List<BaseNodeDTO> findShortestPathBetweenDiseases(@Param("disease1Id") String disease1Id, @Param("disease2Id") String disease2Id);
```

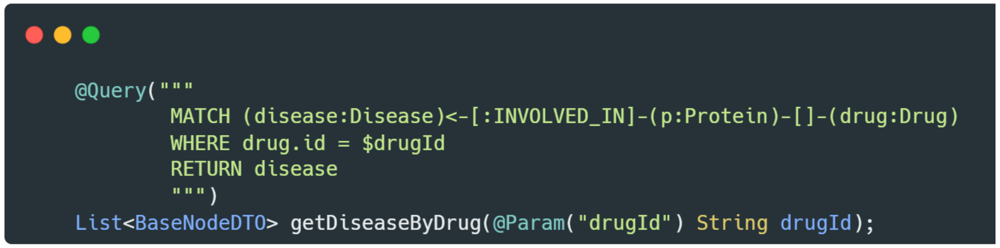
- Enables personalized treatment plans for patients with co-occurring diseases by understanding shared molecular pathways.
- Highlights possible common causes (most probable one) between two diseases



Legend:
- ● **Protein**
- ● **Disease**

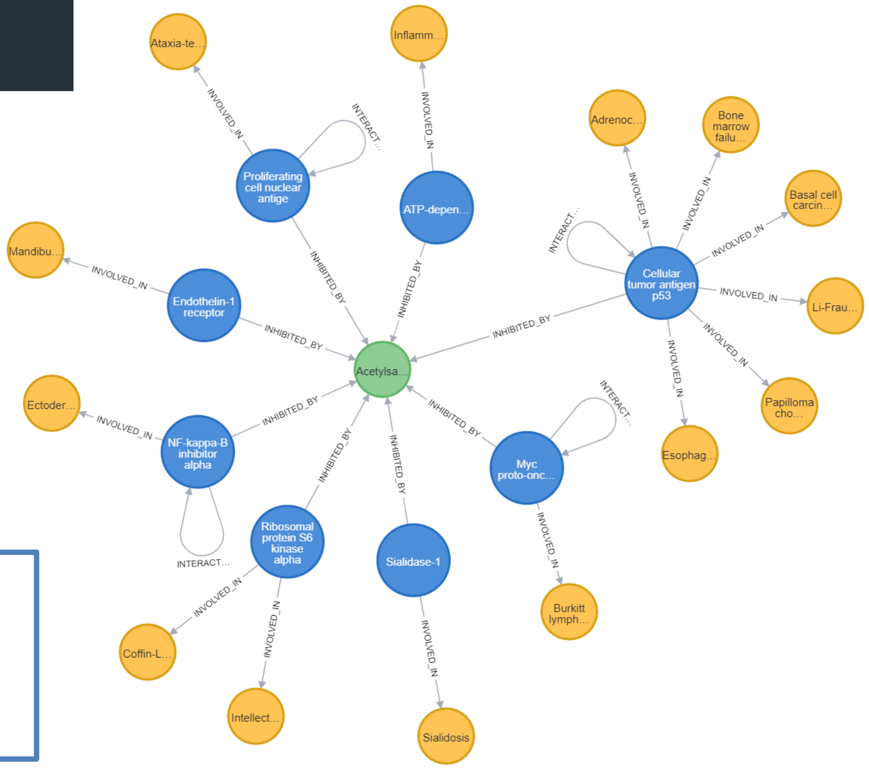
Graph-based queries (2)

Get Diseases Linked to Drug



```
    @Query("""
            MATCH (disease:Disease)<-[:INVOLVED_IN]-(p:Protein)-[]-(drug:Drug)
            WHERE drug.id = $drugId
            RETURN disease
            """)
    List<BaseNodeDTO> getDiseaseByDrug(@Param("drugId") String drugId);
```

Useful for drug companies or researches since a drug, originally developed for one disease, may be applied to multiple conditions



Legend:

- ● Protein
- ● Drug
- ● Disease

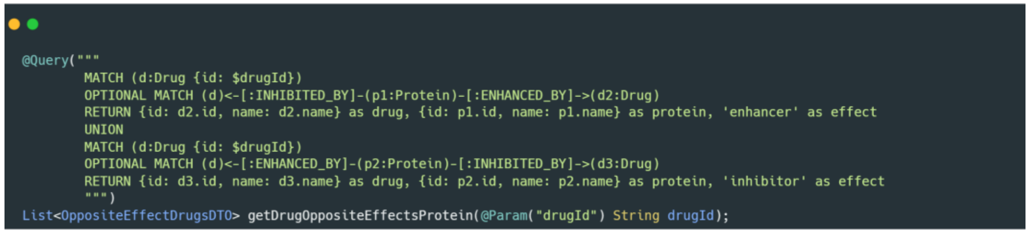
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

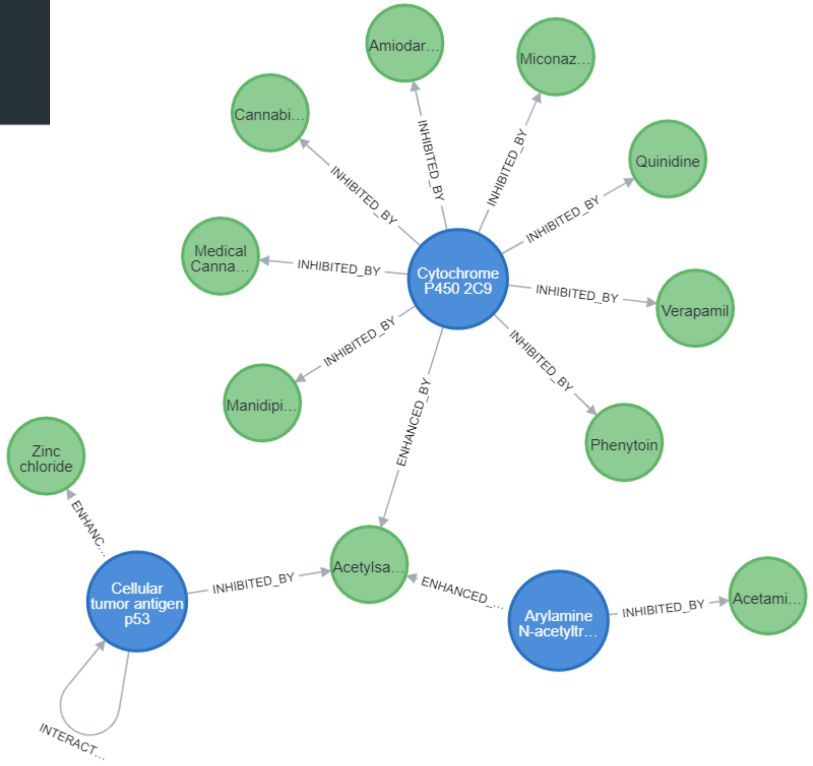
Graph-based queries (3)

Get Drugs with Opposite Effects on Protein



```
@Query("""
        MATCH (d:Drug {id: $drugId})
        OPTIONAL MATCH (d)<-[:INHIBITED_BY]-(p1:Protein)-[:ENHANCED_BY]->(d2:Drug)
        RETURN {id: d2.id, name: d2.name} as drug, {id: p1.id, name: p1.name} as protein, 'enhancer' as effect
        UNION
        MATCH (d:Drug {id: $drugId})
        OPTIONAL MATCH (d)<-[:ENHANCED_BY]-(p2:Protein)-[:INHIBITED_BY]->(d3:Drug)
        RETURN {id: d3.id, name: d3.name} as drug, {id: p2.id, name: p2.name} as protein, 'inhibitor' as effect
        """)
List<OppositeEffectDrugsDTO> getDrugOppositeEffectsProtein(@Param("drugId") String drugId);
```

- Useful to understand Drugs with opposite effects on a specific protein for adverse reaction management or overdoses
- Useful for understanding the problematics of using multi-drugs combinations on a patient

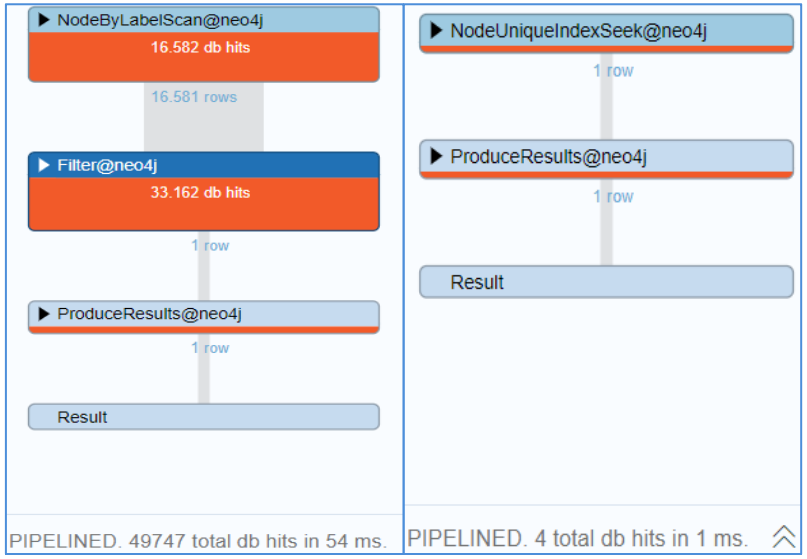


Legend:
- ● **Protein**
- ● **Drug**

Neo4j Indexes

The main role of **indexes** is to efficiently locate the starting node for a query, so we chose to use indexes on the IDs

COLLECTION	FIELD
Protein	id
Drug	id
Disease	id



PIPELINED. 49747 total db hits in 54 ms.

PIPELINED. 4 total db hits in 1 ms.

The images on the right illustrate what happens when the `Drug search` is performed with or without using the index

Below each image, the number of database accesses and the time are displayed

The same type of execution occurs when using the other indexes

Handling Intra-DB Consistency

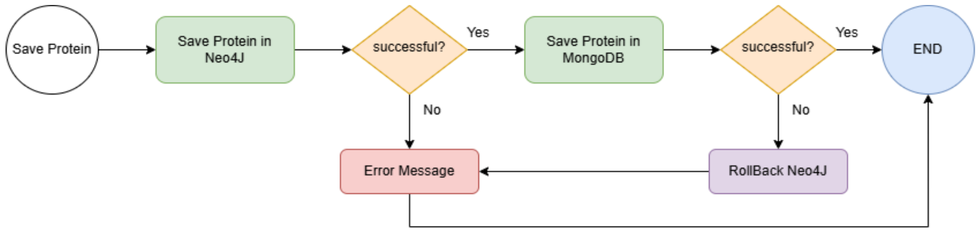
In **Neo4J** the consistency is guaranteed by the atomicity of the CRUD operations inside the single database replica. The only method in our application that perform two sequential modification is handled by a transactional manager.

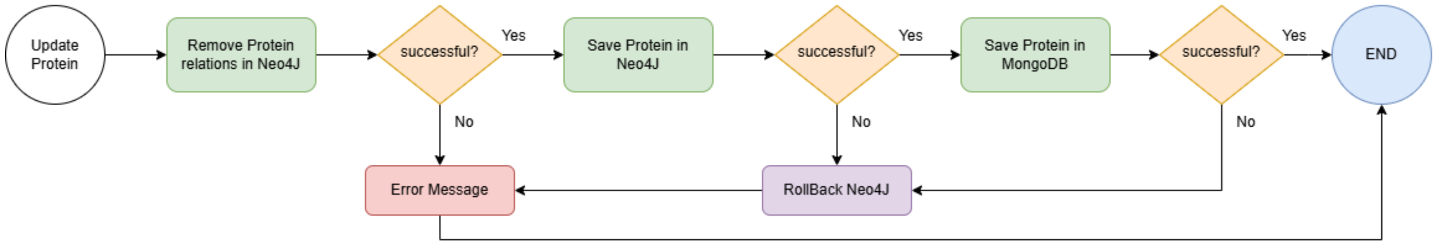
In **MongoDB**, consistency for the primary node is ensured by the atomicity of CRUD operations within a single document. The only actions in our application that modify multiple documents are managed by a transactional manager.

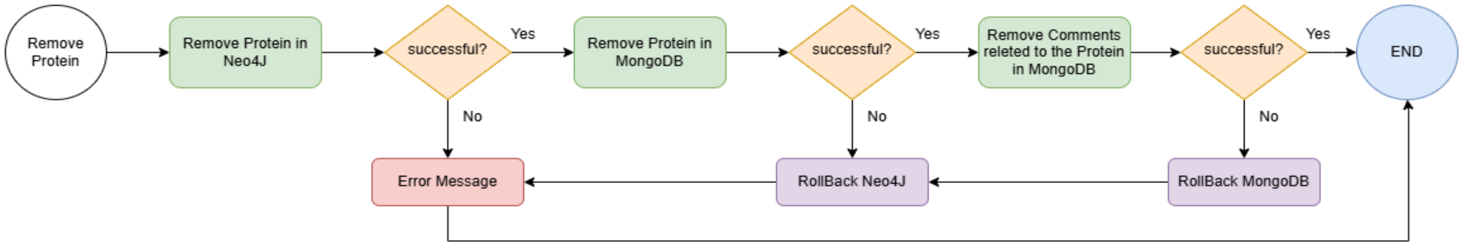
- For consistency between replicas, MongoDB itself handles eventual consistency across the database nodes.

Handling Inter-DB Consistency

The two databases perform joint-operations that complete only if both succeed. Consequently, the data remains consistent between the MongoDB primary and Neo4j, while MongoDB replicas achieve eventual consistency



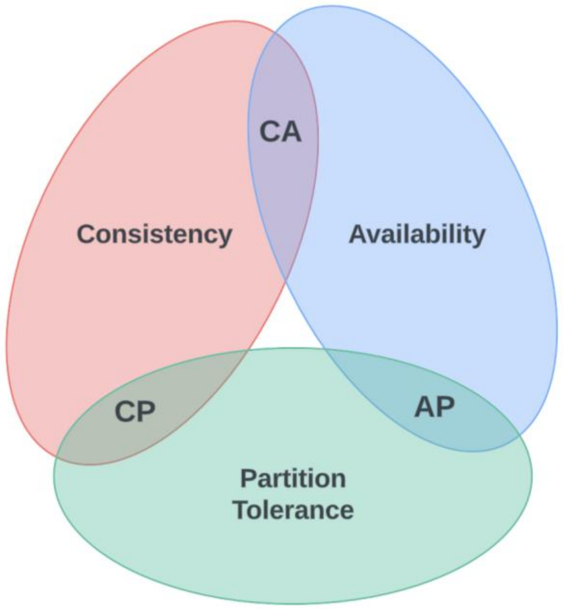




CAP Theorem

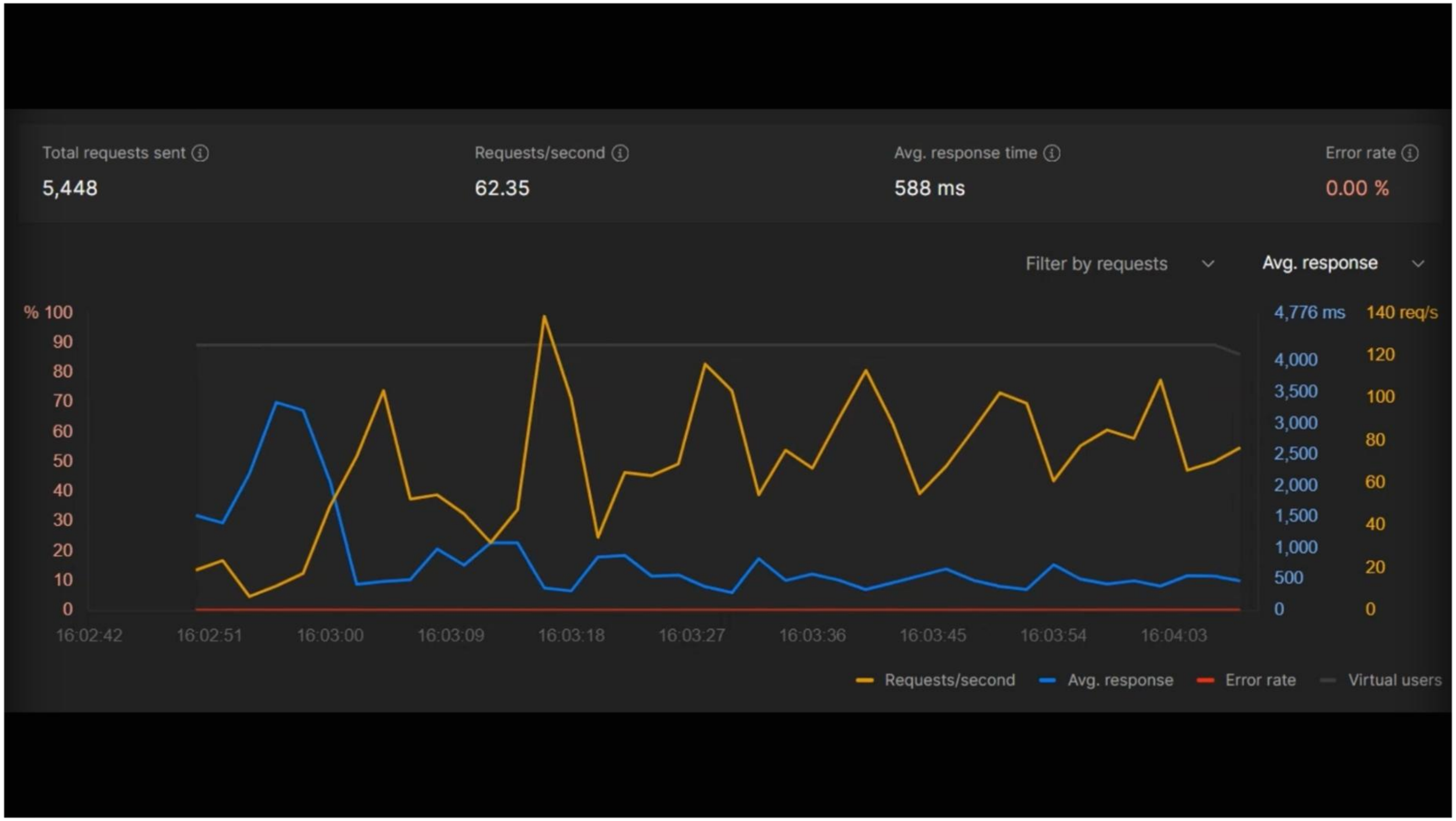
Based on the project requirements and how the entire architecture has been implemented, it is in the **AP (Availability and Partition Tolerance) area** of the CAP theorem.

- Avoid a single point of failure in the system
- Availability in case of a network partition
- Possible retrieval of stale data



IP: 10.1.1.76	**IP: 10.1.1.74**	**IP: 10.1.1.71**
mongoDB: PRIMARY	**mongoDB:** SECONDARY	**mongoDB:** SECONDARY
	API Server	**Neo4j**

System Performance test


Total requests sent ⓘ 5,448

Requests/second ⓘ 62.35

Avg. response time ⓘ 588 ms

Error rate ⓘ 0.00 %

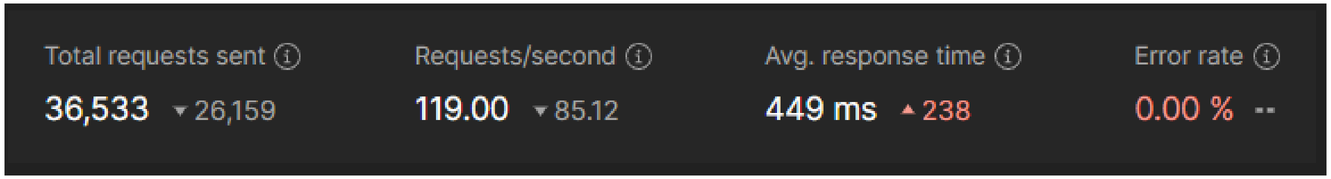
Filter by requests ⌄ Avg. response ⌄

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

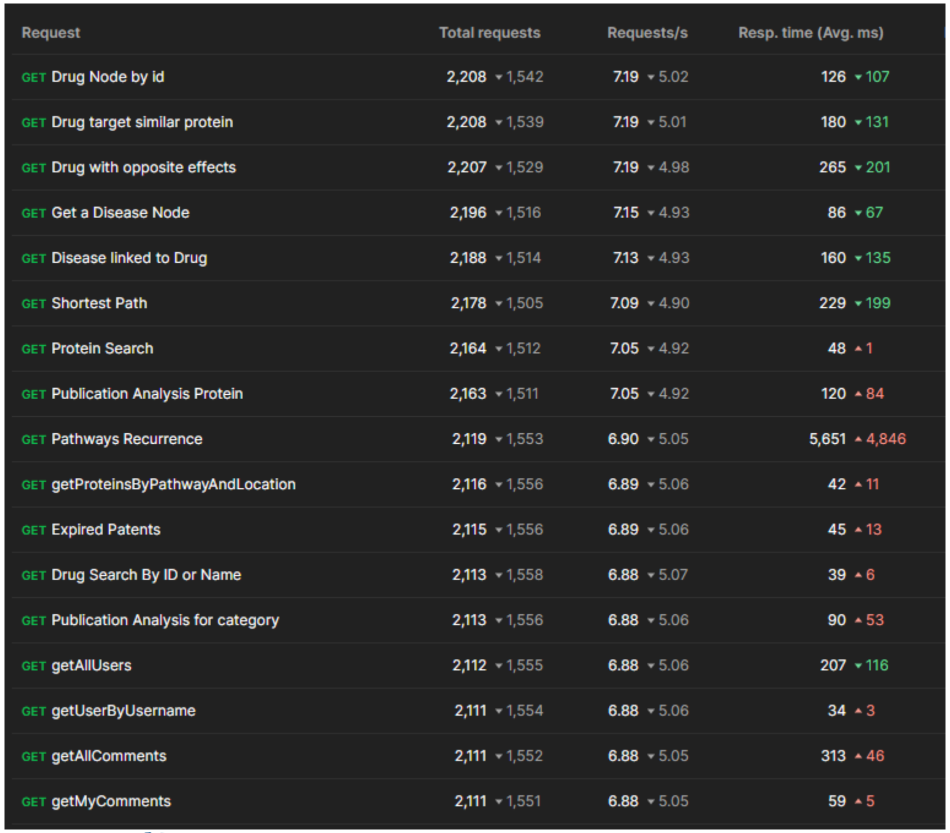
UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

readPreference analysis



Total requests sent	Requests/second	Avg. response time	Error rate
36,533 ▾ 26,159	119.00 ▾ 85.12	449 ms ▴ 238	0.00 % --

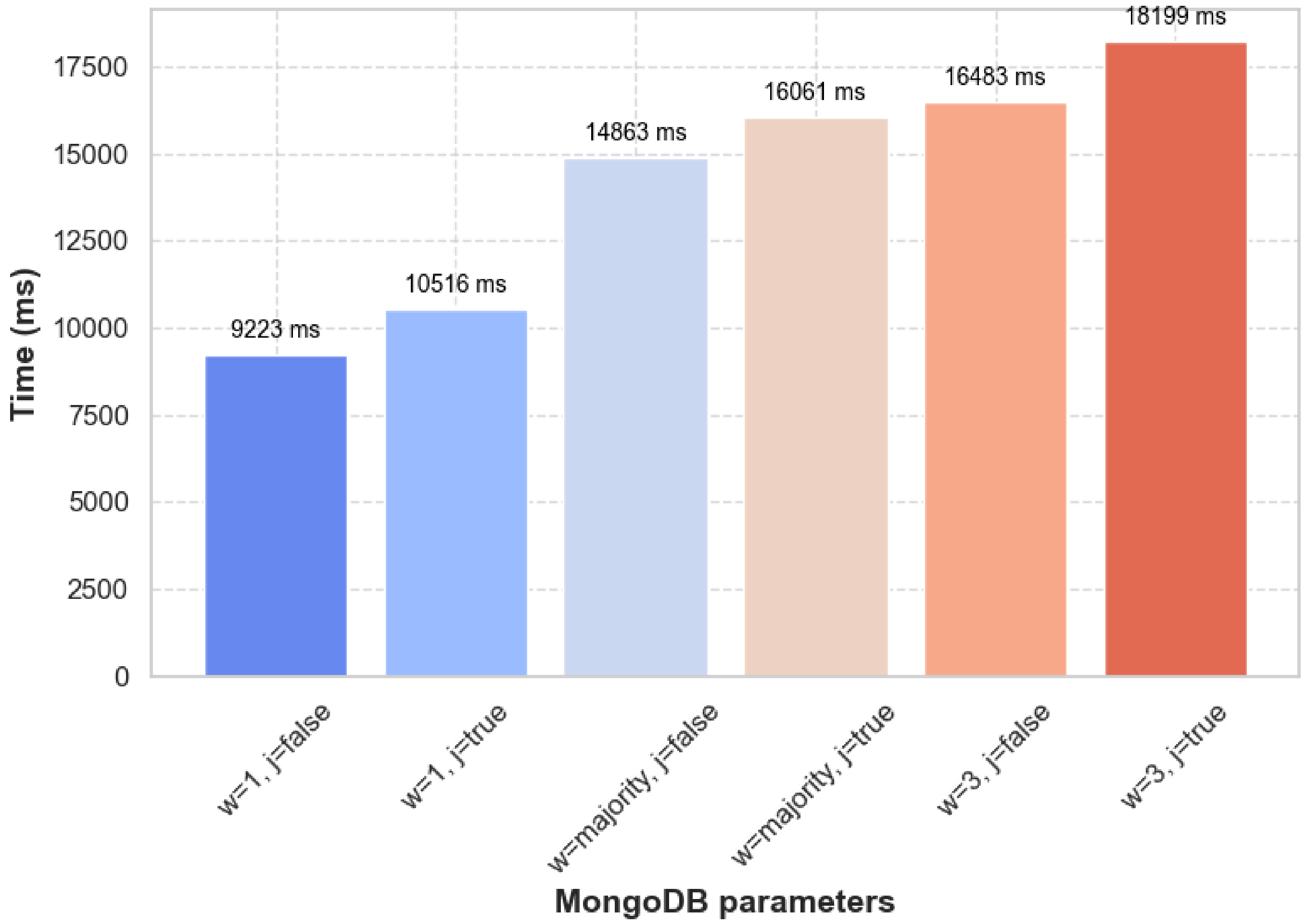


Request	Total requests	Requests/s	Resp. time (Avg. ms)
GET Drug Node by id	2,208 ▾ 1,542	7.19 ▾ 5.02	126 ▾ 107
GET Drug target similar protein	2,208 ▾ 1,539	7.19 ▾ 5.01	180 ▾ 131
GET Drug with opposite effects	2,207 ▾ 1,529	7.19 ▾ 4.98	265 ▾ 201
GET Get a Disease Node	2,196 ▾ 1,516	7.15 ▾ 4.93	86 ▾ 67
GET Disease linked to Drug	2,188 ▾ 1,514	7.13 ▾ 4.93	160 ▾ 135
GET Shortest Path	2,178 ▾ 1,505	7.09 ▾ 4.90	229 ▾ 199
GET Protein Search	2,164 ▾ 1,512	7.05 ▾ 4.92	48 ▴ 1
GET Publication Analysis Protein	2,163 ▾ 1,511	7.05 ▾ 4.92	120 ▴ 84
GET Pathways Recurrence	2,119 ▾ 1,553	6.90 ▾ 5.05	5,651 ▴ 4,846
GET getProteinsByPathwayAndLocation	2,116 ▾ 1,556	6.89 ▾ 5.06	42 ▴ 11
GET Expired Patents	2,115 ▾ 1,556	6.89 ▾ 5.06	45 ▴ 13
GET Drug Search By ID or Name	2,113 ▾ 1,558	6.88 ▾ 5.07	39 ▴ 6
GET Publication Analysis for category	2,113 ▾ 1,556	6.88 ▾ 5.06	90 ▴ 53
GET getAllUsers	2,112 ▾ 1,555	6.88 ▾ 5.06	207 ▾ 116
GET getUserByUsername	2,111 ▾ 1,554	6.88 ▾ 5.06	34 ▴ 3
GET getAllComments	2,111 ▾ 1,552	6.88 ▾ 5.05	313 ▴ 46
GET getMyComments	2,111 ▾ 1,551	6.88 ▾ 5.05	59 ▴ 5

Analysis of the response time for ***readPreference=primary*** in contrapposition with ***readPreference=nearest***

Writes Response Time Comparison

During the insertion of 1000 proteins, sequentially



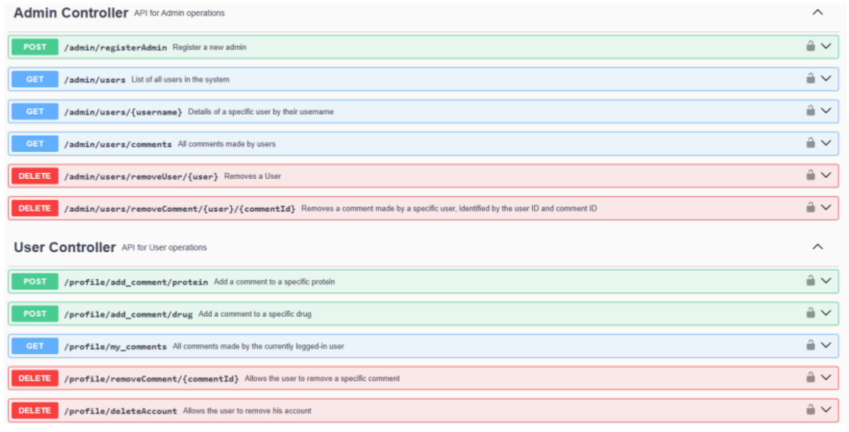
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Swagger UI REST APIs documentation

Here are some of the end-points using Swagger UI

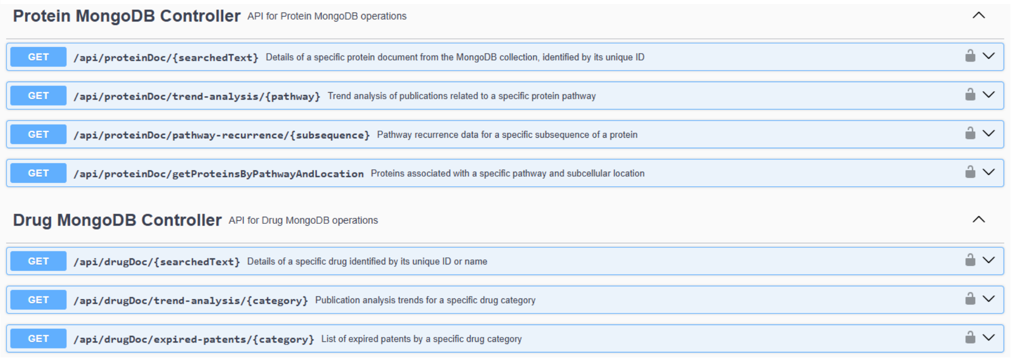


Admin Controller API for Admin operations

POST /admin/registerAdmin Register a new admin
GET /admin/users List of all users in the system
GET /admin/users/{username} Details of a specific user by their username
GET /admin/users/comments All comments made by users
DELETE /admin/users/removeUser/{user} Removes a User
DELETE /admin/users/removeComment/{user}/{commentId} Removes a comment made by a specific user, identified by the user ID and comment ID

User Controller API for User operations

POST /profile/add_comment/protein Add a comment to a specific protein
POST /profile/add_comment/drug Add a comment to a specific drug
GET /profile/my_comments All comments made by the currently logged-in user
DELETE /profile/removeComment/{commentId} Allows the user to remove a specific comment
DELETE /profile/deleteAccount Allows the user to remove his account

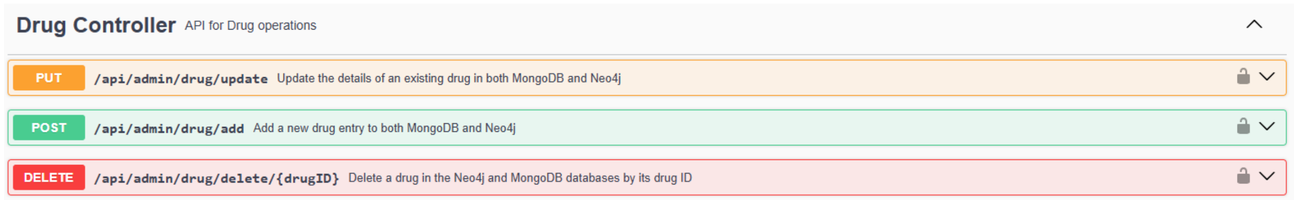


Protein MongoDB Controller API for Protein MongoDB operations

GET /api/proteinDoc/{searchedText} Details of a specific protein document from the MongoDB collection, identified by its unique ID
GET /api/proteinDoc/trend-analysis/{pathway} Trend analysis of publications related to a specific protein pathway
GET /api/proteinDoc/pathway-recurrence/{subsequence} Pathway recurrence data for a specific subsequence of a protein
GET /api/proteinDoc/getProteinsByPathwayAndLocation Proteins associated with a specific pathway and subcellular location

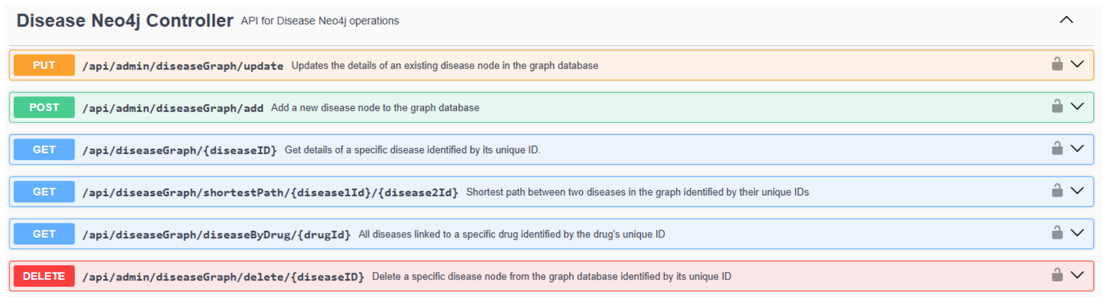
Drug MongoDB Controller API for Drug MongoDB operations

GET /api/drugDoc/{searchedText} Details of a specific drug identified by its unique ID or name
GET /api/drugDoc/trend-analysis/{category} Publication analysis trends for a specific drug category
GET /api/drugDoc/expired-patents/{category} List of expired patents by a specific drug category



Drug Controller API for Drug operations

PUT /api/admin/drug/update Update the details of an existing drug in both MongoDB and Neo4j
POST /api/admin/drug/add Add a new drug entry to both MongoDB and Neo4j
DELETE /api/admin/drug/delete/{drugID} Delete a drug in the Neo4j and MongoDB databases by its drug ID



Disease Neo4j Controller API for Disease Neo4j operations

PUT /api/admin/diseaseGraph/update Updates the details of an existing disease node in the graph database
POST /api/admin/diseaseGraph/add Add a new disease node to the graph database
GET /api/diseaseGraph/{diseaseID} Get details of a specific disease identified by its unique ID.
GET /api/diseaseGraph/shortestPath/{disease1Id}/{disease2Id} Shortest path between two diseases in the graph identified by their unique IDs
GET /api/diseaseGraph/diseaseByDrug/{drugId} All diseases linked to a specific drug identified by the drug's unique ID
DELETE /api/admin/diseaseGraph/delete/{diseaseID} Delete a specific disease node from the graph database identified by its unique ID

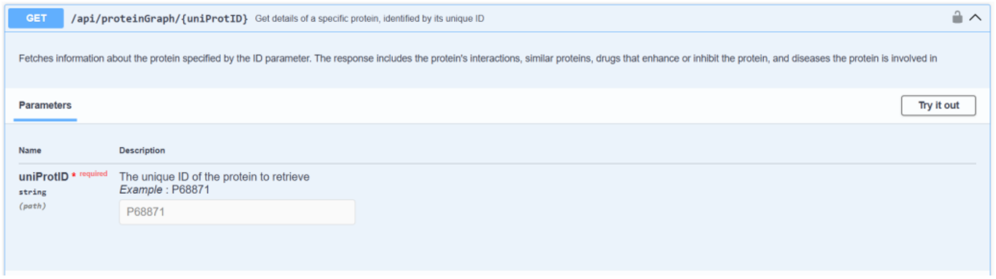
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

IN SUPREMÆ DIGNITATIS 1343
UNIVERSITÀ DI PISA

CROSSLAB Innovation for industry 4.0

Swagger UI REST APIs documentation

For each endpoint, we provided examples of **input parameters** and documented the possible **HTTP responses**, including returned payloads and error messages

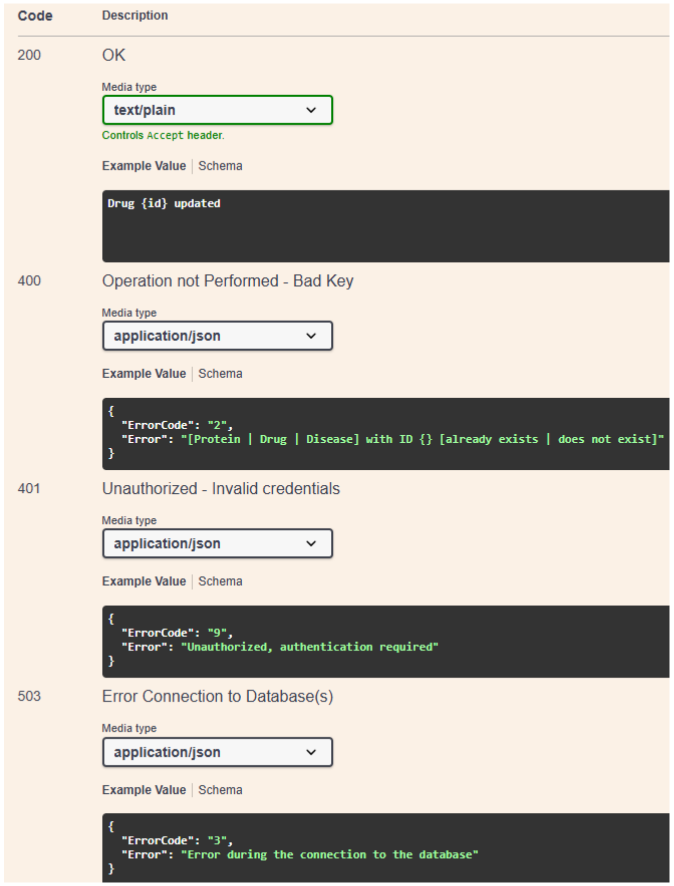


GET `/api/proteinGraph/{uniProtID}` Get details of a specific protein, identified by its unique ID

Fetches information about the protein specified by the ID parameter. The response includes the protein's interactions, similar proteins, drugs that enhance or inhibit the protein, and diseases the protein is involved in

Parameters — Try it out

Name	Description
uniProtID * required `string` *(path)*	The unique ID of the protein to retrieve *Example* : P68871 P68871



Code	Description
200	OK Media type: text/plain Controls `Accept` header. Example Value | Schema `Drug {id} updated`
400	Operation not Performed - Bad Key Media type: application/json Example Value | Schema
401	Unauthorized - Invalid credentials Media type: application/json Example Value | Schema
503	Error Connection to Database(s) Media type: application/json Example Value | Schema

400 example value:

```
{
  "ErrorCode": "2",
  "Error": "[Protein | Drug | Disease] with ID {} [already exists | does not exist]"
}
```

401 example value:

```
{
  "ErrorCode": "9",
  "Error": "Unauthorized, authentication required"
}
```

503 example value:

```
{
  "ErrorCode": "3",
  "Error": "Error during the connection to the database"
}
```