# DB with RDKit

**Mishima.syk #16**
**@iwatobipen**

# Who am I?


- Runner
- Chemoinformatician(?)
- Medicinal chemist

# Today's topic

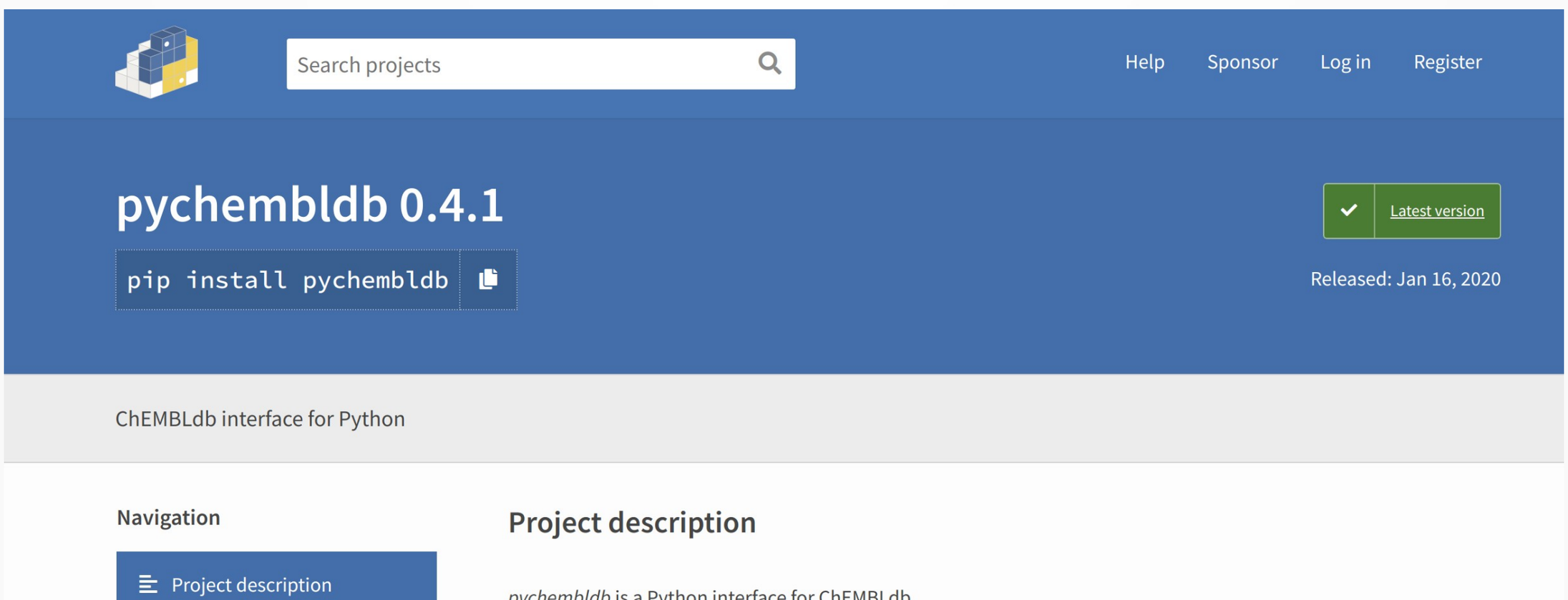
- DataBase handling with RDKit
- from RDKit UGM

# Data Base handling with RDKit

- RDKit can integrate many kinds of Databases
  - ☆Postgresql
  - SQLite3
  - Neo4j
  - MongoDB
  - .....

# Useful package for handing ChEMBL1

https://pypi.org/project/pychembldb/



# By using pychembldb, SQL statement will be pythonic! Power of sql alchemy;)

```
SELECT target_dictionary.tid AS target_dictionary_tid,
target_dictionary.target_type AS target_dictionary_target_type,
target_dictionary.pref_name AS target_dictionary_pref_name,
target_dictionary.tax_id AS target_dictionary_tax_id, target_dictionary.organism
AS target_dictionary_organism, target_dictionary.chembl_id AS
target_dictionary_chembl_id, target_dictionary.species_group_flag AS
target_dictionary_species_group_flag
FROM target_dictionary
WHERE target_dictionary.pref_name = "Tyrosine-protein kinase ABL”;
```

```
from pychembldb import *
for target in chembldb.query(Target).filter_by(pref_name="Tyrosine-protein
kinase ABL"):
    for assay in target.assays:
        for activity in assay.activities:
            print(activity.value,
activity.compound.molecule.structure.standard_inchi_key)
```

# Useful package for chemoinfo DB2

A cheminformatics extension for the SQLAlchemy database toolkit.

https://github.com/rvianello/razi

**rvianello** stop using a virtualenv inside a docker-based dev environment `3131fe2` on Apr 22 **129** commits

| | | |
|---|---|---|
| .devcontainer | stop using a virtualenv inside a docker-based dev environment | 7 months ago |
| .vscode | stop using a virtualenv inside a docker-based dev environment | 7 months ago |
| docs | update the function parsing the chembl data | 3 years ago |
| razi | relocate tests | 7 months ago |
| test_data | fix connection handling and add some tests using a reaction column type | 3 years ago |
| tests | fix path to tests data | 7 months ago |
| .gitignore | small updates | 7 months ago |
| LICENSE | update the copyright notice | 3 years ago |
| README.md | small updates | 7 months ago |
| setup.py | small updates | 7 months ago |

**README.md**

## Razi

A cheminformatics extension for the SQLAlchemy database toolkit.

SQLAlchemy database toolkit.

razi.readthedocs.org/

Readme

View license

**Releases**

**1** tags

**Packages**

No packages published

**Contributors** 4

**rvianello** Riccardo Vianello

**yamasakih** yamasakih

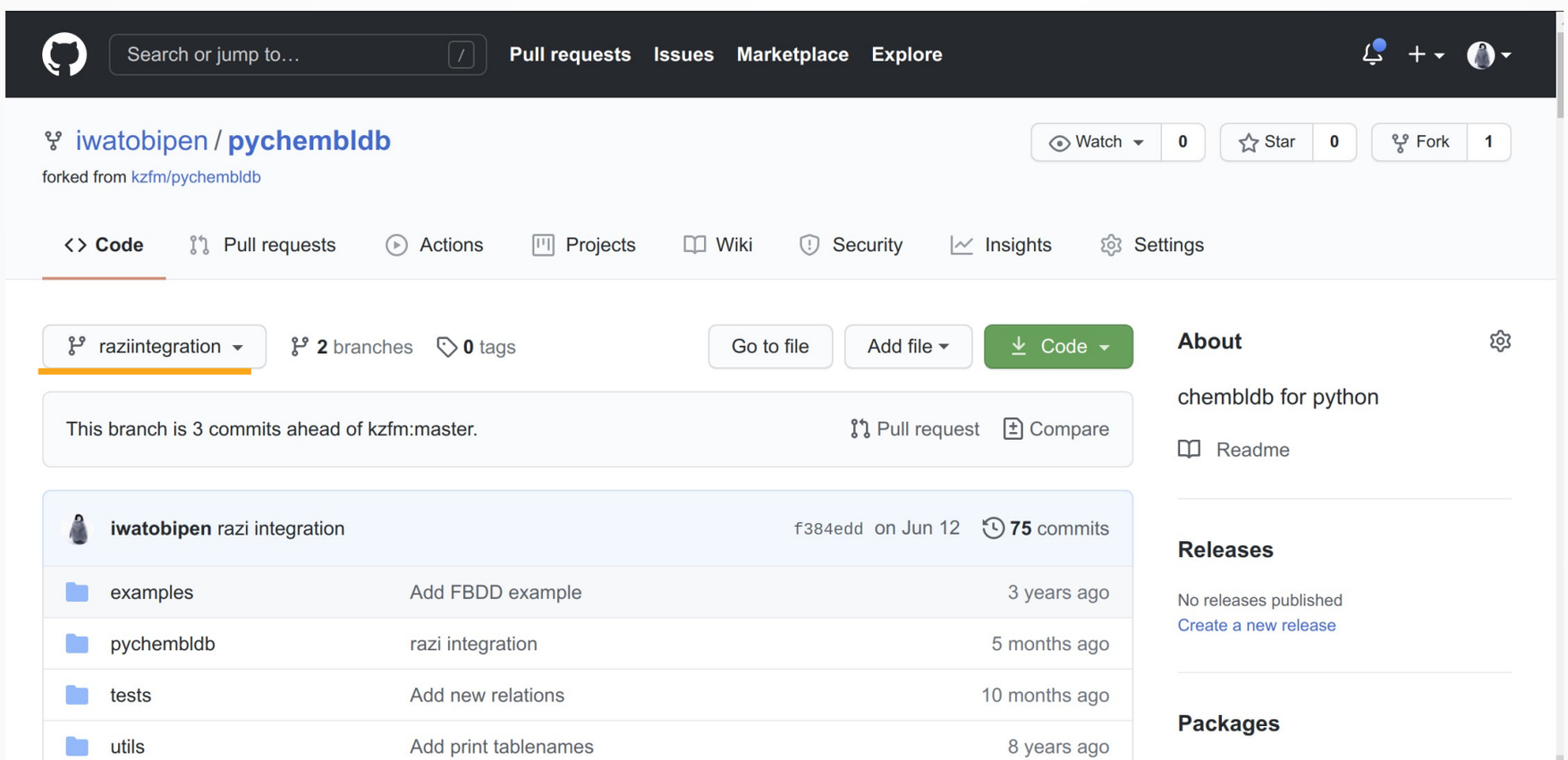
**llazzaro** Leonardo Lazzaro

**admed** Adrian

# pychembldb + razi = useful isn't it?

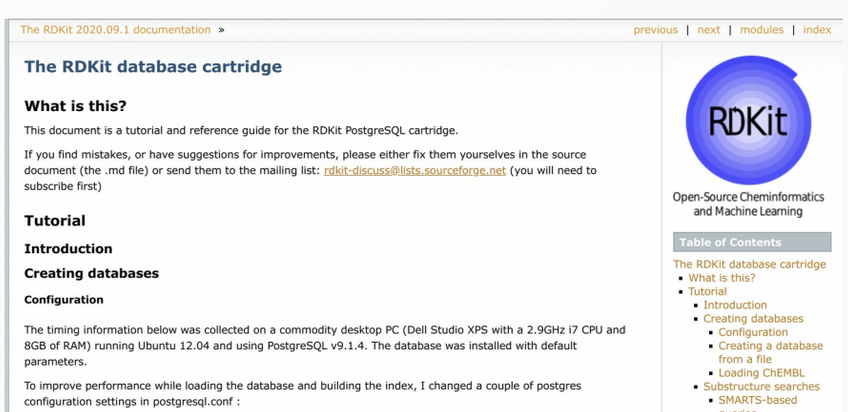
Forked pychembldb and integrate the package with razi ;)

https://github.com/iwatobipen/pychembldb/tree/raziintegration



# Mapping rdk.mols table

```
userdk = False
try:
    from razi import rdkit_postgresql
    from razi.rdkit_postgresql.types import Mol
    from razi.rdkit_postgresql.types import Bfp
    from rdkit import Chem
    metadata_rdk = MetaData(schema='rdk', bind=engine)
    class Mols(Base):
        __table__ = Table('mols',
                          metadata_rdk,
                          Column('molregno', BIGINT, primary_key=True),
                          Column('m', Mol),
                          extend_existing=True,
                          )
        __table_args__ = (
            Index('molidx', 'structure',
                        postgresql_using='gist'),

            )


        def __repr__(self):
            if isinstance(self.m, Chem.Mol):
                return '(%s) < %s >' % (self.molregno, Chem.MolToSmiles(self.m))
            return '(%s) < %s >' % (self.molregno, self.m)
```



The RDKit 2020.09.1 documentation » previous | next | modules | index

## The RDKit database cartridge

### What is this?

This document is a tutorial and reference guide for the RDKit PostgreSQL cartridge.

If you find mistakes, or have suggestions for improvements, please either fix them yourselves in the source document (the .md file) or send them to the mailing list: rdkit-discuss@lists.sourceforge.net (you will need to subscribe first)

### Tutorial

#### Introduction

#### Creating databases

##### Configuration

The timing information below was collected on a commodity desktop PC (Dell Studio XPS with a 2.9GHz i7 CPU and 8GB of RAM) running Ubuntu 12.04 and using PostgreSQL v9.1.4. The database was installed with default parameters.

To improve performance while loading the database and building the index, I changed a couple of postgres configuration settings in postgresql.conf :

RDKit

Open-Source Cheminformatics and Machine Learning

Table of Contents



https://www.rdkit.org/docs/Cartridge.html

# Example Usage1

- Substructure Search

# Without razi....

```
In [2]:  query = chembldb.query(Mols, Assay, Activity, TargetDictionary)

In [3]:  res = query.join(Activity).join(
                                TargetDictionary).filter(
                                Mols.molregno==Activity.molregno).filter(
                                TargetDictionary.chembl_id=='CHEMBL2362975').filter(
                                Activity.standard_type=='LogD')

In [4]:  res.count()
Out[4]:  17850

In [5]:  mols = []
         for row in res:
             if row[0].m != None:
                 mols.append(row[0].m)

In [6]:  len(mols)
Out[6]:  17850

In [7]:  matchmols = []
         querymol = Chem.MolFromSmiles('c1cncnc1')

In [8]:  for mol in mols:
             if mol.HasSubstructMatch(querymol):
                 matchmols.append(mol)

In [9]:  len(matchmols)
Out[9]:  2835
```

RDKit Mols 取得

該当構造だけに絞る

# With razi....

In [11]:
```
query = chembldb.query(Mols, Assay, Activity, TargetDictionary)
res2 = query.join(Activity).join(
                                TargetDictionary).filter(
                                Mols.molregno==Activity.molregno).filter(
                                TargetDictionary.chembl_id=='CHEMBL2362975').filter(
                                Activity.standard_type=='LogD').filter(
                                Mols.m.hassubstruct('c1cncnc1')
)
```

SSFilter SQL

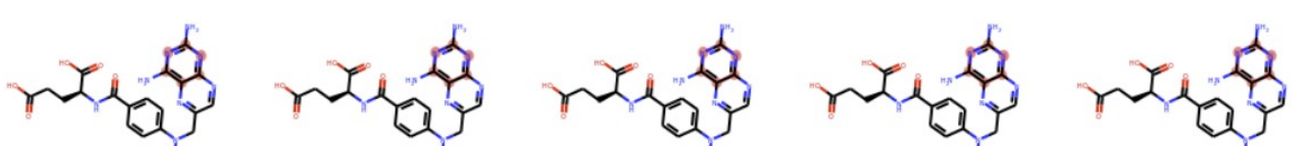
In [12]:
```
mols2 = []
for row in res2:
    if row[0].m != None:
        mols2.append(row[0].m)
```

In [13]:
```
print(len(mols2), len(matchmols))
```

```
2835 2835
```

In [14]:
```
Draw.MolsToGridImage(mols2[:10], molsPerRow=5, highlightAtomLists=[mol.GetSubstructMatch(
```

Out[14]:



# Example Usage2

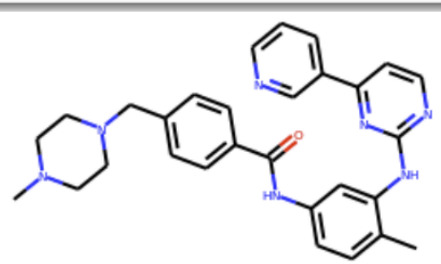
- Similarity Search

# Without razi....

```
chembl_27=# select rdk.mols.m from rdk.fps, rdk.mols where mfp2%morganbv_fp('Cc1ccc2nc(-
c3ccc(NC(C4N(C(c5cccs5)=O)CCC4)=O)cc3)sc2c1') and rdk.fps.molregno=rdk.fps.molregno limit
10;
                                    m
--------------------------------------------------------------------------------
 Cc1cc(-n2ncc(=O)[nH]c2=O)ccc1C(=O)c1ccccc1Cl
 Cc1cc(-n2ncc(=O)[nH]c2=O)ccc1C(=O)c1ccc(C#N)cc1
 Cc1cc(-n2ncc(=O)[nH]c2=O)cc(C)c1C(O)c1ccc(Cl)cc1
 Cc1ccc(C(=O)c2ccc(-n3ncc(=O)[nH]c3=O)cc2)cc1
 Cc1cc(-n2ncc(=O)[nH]c2=O)ccc1C(=O)c1ccc(Cl)cc1
 Cc1cc(-n2ncc(=O)[nH]c2=O)ccc1C(=O)c1ccccc1
 Cc1cc(Br)ccc1C(=O)c1ccc(-n2ncc(=O)[nH]c2=O)cc1Cl
 O=C(c1ccc(Cl)cc1Cl)c1ccc(-n2ncc(=O)[nH]c2=O)cc1Cl
 CS(=O)(=O)c1ccc(C(=O)c2ccc(-n3ncc(=O)[nH]c3=O)cc2Cl)cc1
 c1cc2cc(c1)-c1cccc(c1)C[n+]1ccc(c3ccccc31)NCCCCCCCCCCNc1cc[n+](c3ccccc13)C2
(10 rows)
```

# With razi

```
In [45]:  imatinib = Chem.MolFromSmiles('CC1=C(C=C(C=C1)NC(=O)C2=CC=C(C=C2)CN3CCN(CC3)C)NC4=NC=CC(=N4)C5=CN=CC=C5')
          imatinib = 'CC1=C(C=C(C=C1)NC(=O)C2=CC=C(C=C2)CN3CCN(CC3)C)NC4=NC=CC(=N4)C5=CN=CC=C5'

In [53]:  from razi.rdkit_postgresql.functions import morganbv_fp

In [74]:  query = chembldb.query(Assay, Activity, Mols, Fps).join(Activity).filter(
              Activity.molregno==Mols.molregno).filter(
              Mols.molregno==Fps.molregno)
          query_bv = morganbv_fp(imatinib,2)

In [75]:  res = query.filter(Fps.mfp2.tanimoto_sml(query_bv))

In [106]: from IPython.display import display
          for row in res.limit(10):
              display(row[2].m)
              print(row[2].molregno, row[3].molregno, row[4])
```



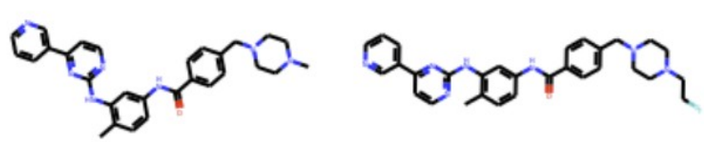
```
88797 88797 1.0
```

# With razi

```
In [15]:  rdk_imatinib = Chem.MolFromSmiles('CC1=C(C=C(C=C1)NC(=O)C2=CC=C(C=C2)CN3CCN(CC3)C)NC4=NC=CC(=N4)C5=CN=CC=C5')
          imatinib = 'CC1=C(C=C(C=C1)NC(=O)C2=CC=C(C=C2)CN3CCN(CC3)C)NC4=NC=CC(=N4)C5=CN=CC=C5'

In [16]:  from razi.rdkit_postgresql.functions import morganbv_fp
          from razi.rdkit_postgresql.functions import tanimoto_sml

In [17]:  query_bv = morganbv_fp(imatinib,2)
          tanimotosim = tanimoto_sml(Fps.mfp2, query_bv).label('similarity')

          query = chembldb.query(Assay, Activity, Mols, Fps, tanimotosim).join(Activity).filter(
              Activity.molregno==Mols.molregno).filter(
              Mols.molregno==Fps.molregno)

In [22]:  res = query.filter(Fps.mfp2.tanimoto_sml(query_bv)).order_by(desc('similarity'))
```

# With razi

```
In [22]:  res = query.filter(Fps.mfp2.tanimoto_sml(query_bv)).order_by(desc('similarity'))

In [28]:  from IPython.display import display
          for idx, row in enumerate(res):
              if row[4] < 0.9 and row[4] > 0.8:
                  display(Draw.MolsToGridImage([rdk_imatinib, row[2].m]))
                  print(row[2].molregno, row[3].molregno, row[4])
```

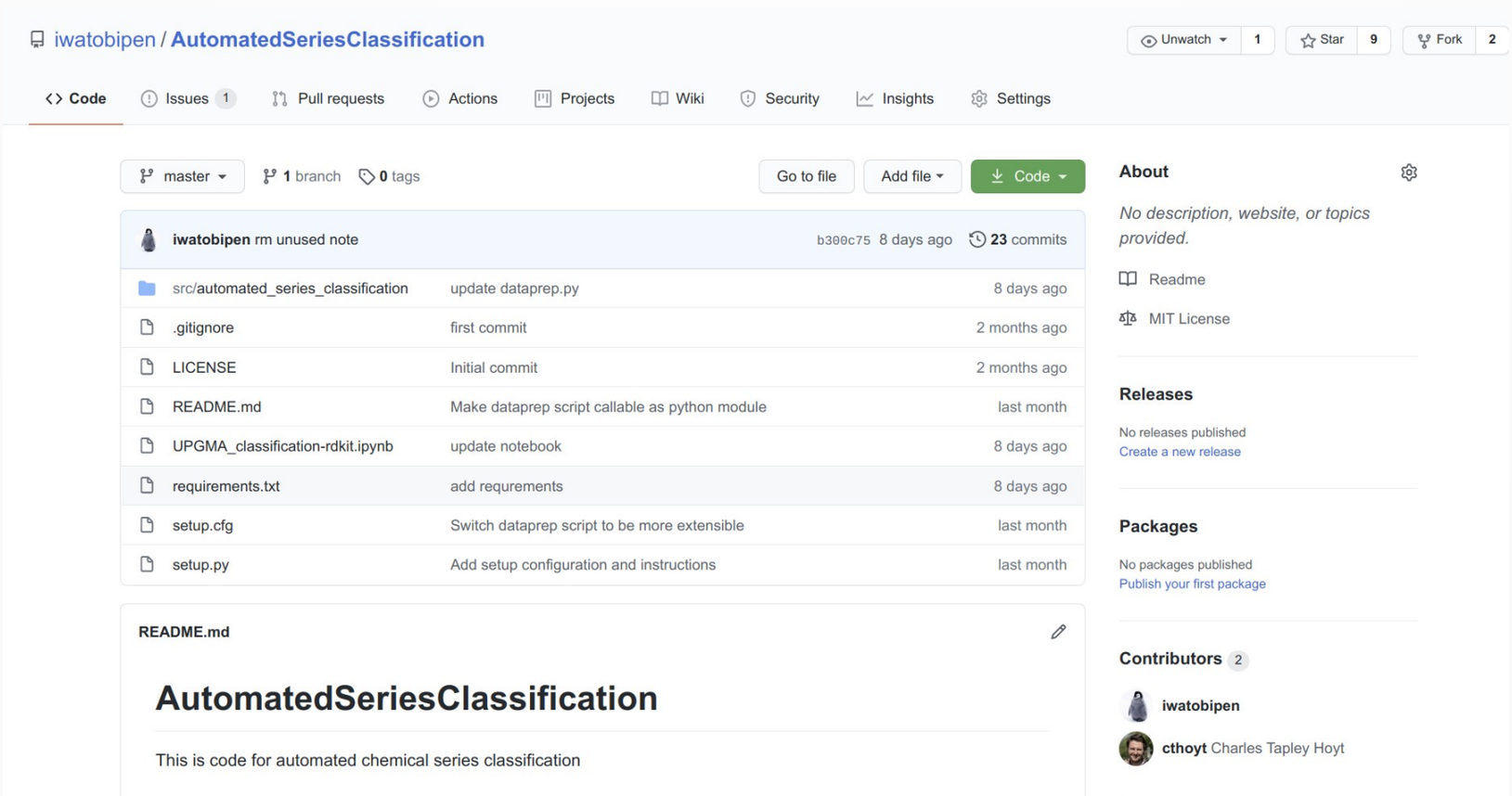


```
1609383 1609383 0.898550724637681
```

# RDKit UGM topic

- I had an opportunity to present at RDKit UGM ;)

  After the UGM, I got nice PR!



# Acknowledgment


- @fmkz__
- @yamasaKit_
- @cthoyt