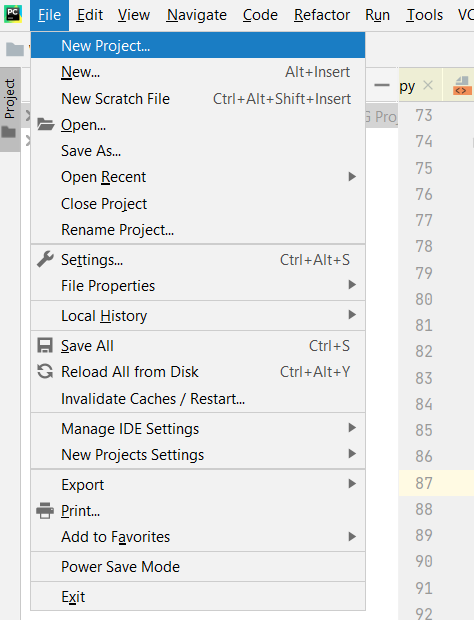
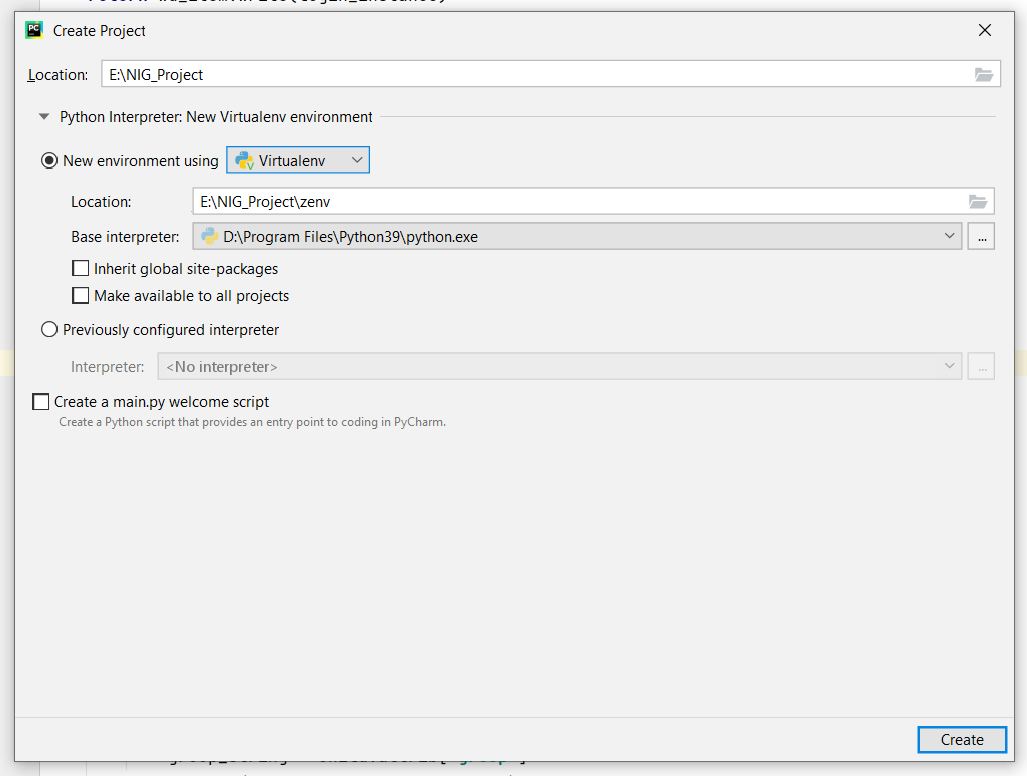
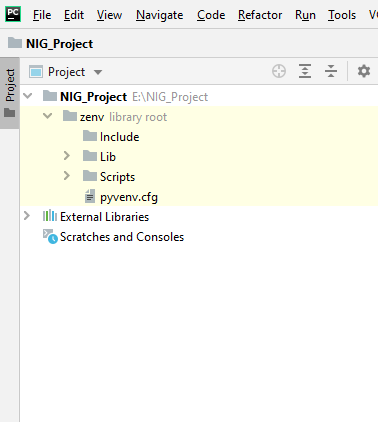
**Instruction to run doid python script**

**Prerequisites**

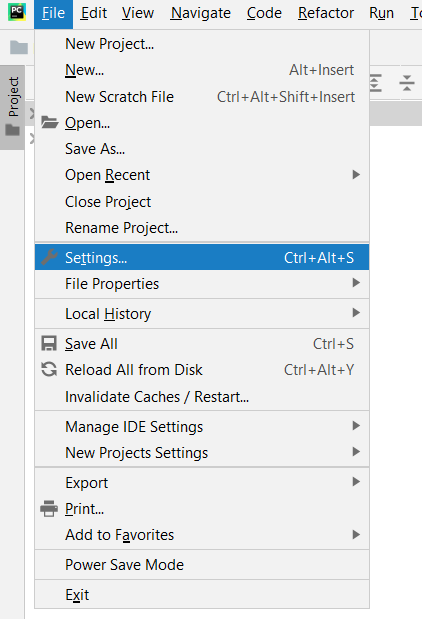
* PyCharm IDE <https://www.jetbrains.com/pycharm/>
* Python (version 3.9) <https://www.python.org/downloads/>

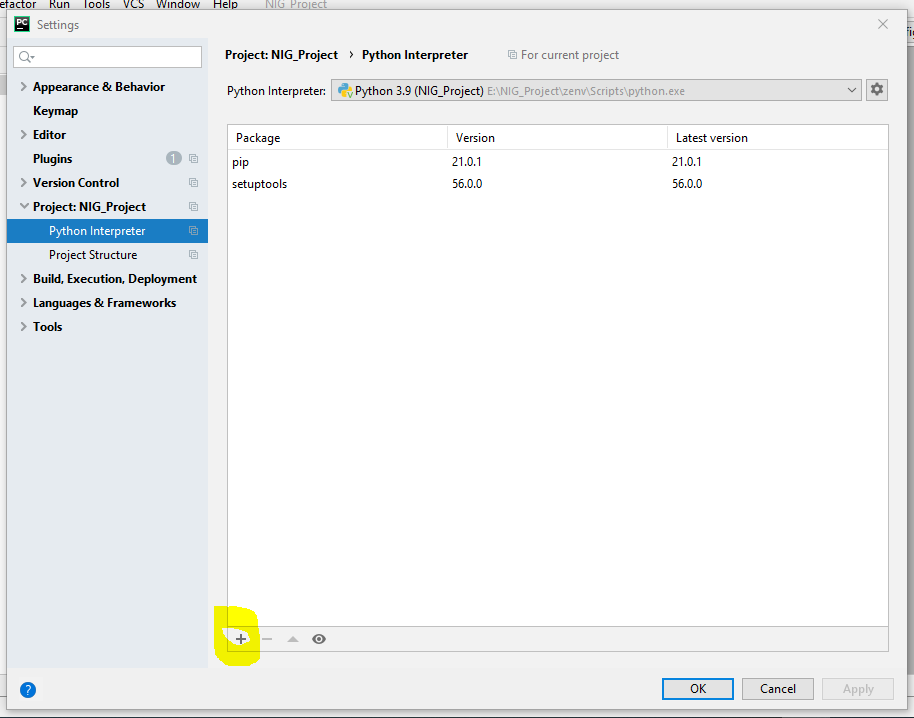
1. Open PyCharm IDE.
2. Create a new project. **MainMenu -> File -> NewProject**
3. Fill the required fields in the Create Project window as follows and click on the create button.

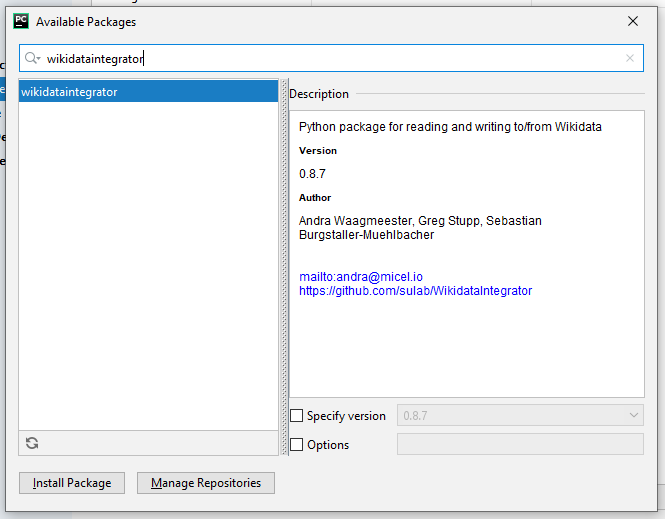
**Note**: - “NIG\_Project” is given as Project Name.

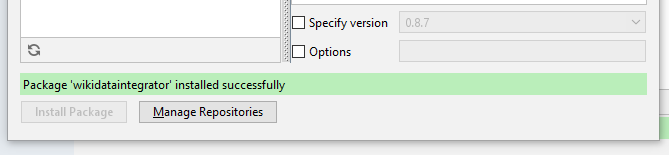
1. NIG\_Project structure can be seen in the Project window as the following image.

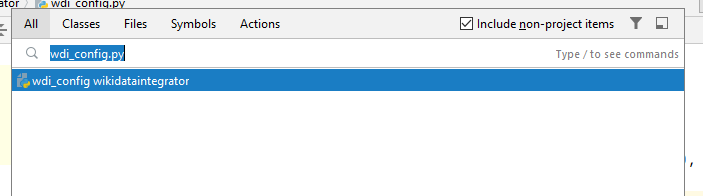
1. Install “wikidataintegrator” to the project environment. **MainMenu -> File -> Settings**



1. In resulted window go to **Project: NIG\_Project -> Python Interpreter -> Click on the Plus sign** (Highlighted in the following image).
2. In the next window (Available Packages) search for “wikidataintegrator” as follows. It will give the following package as results. Then click on Install Packages.



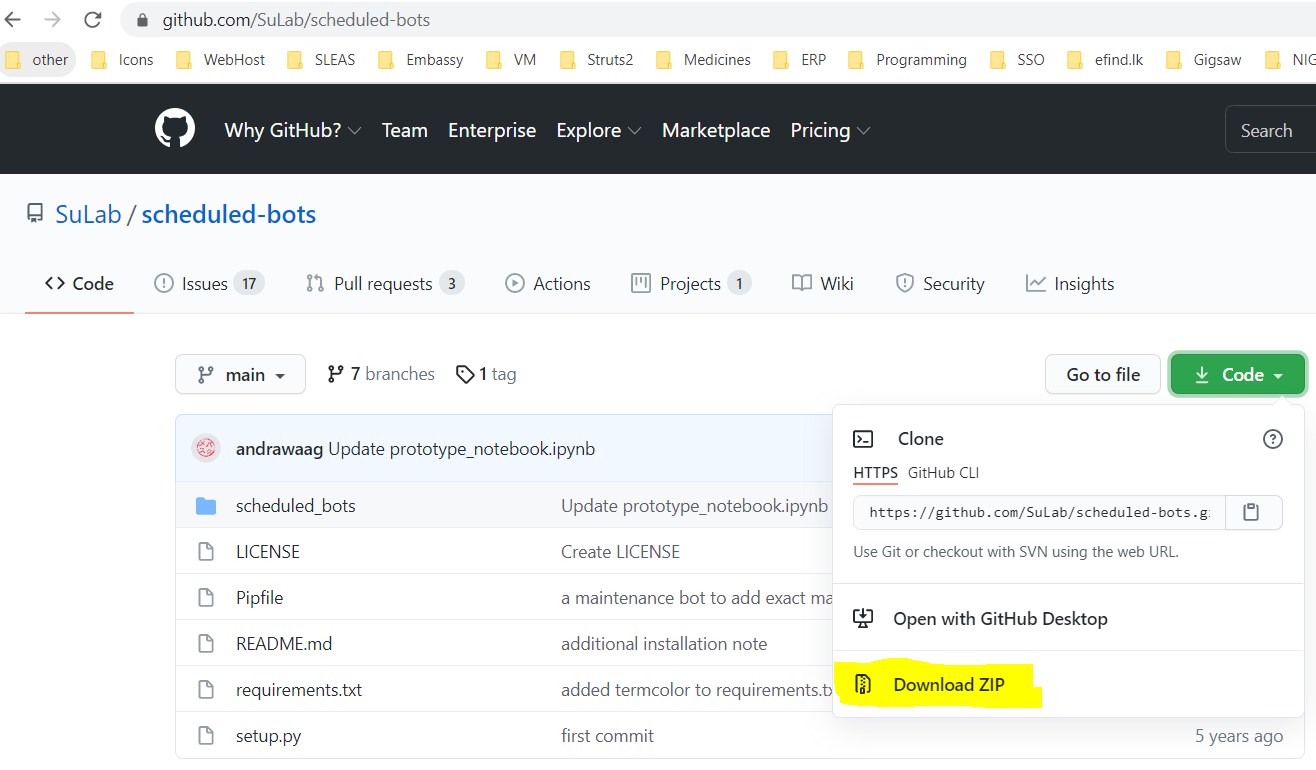
1. After successful installation it will give the following success message. Close the windows (Available Packages and Settings).
2. To find wdi\_config.py file go to **MainMenu -> Navigate -> Search Everywhere**

In the search window type “wdi\_config.py” as follows and open the file.

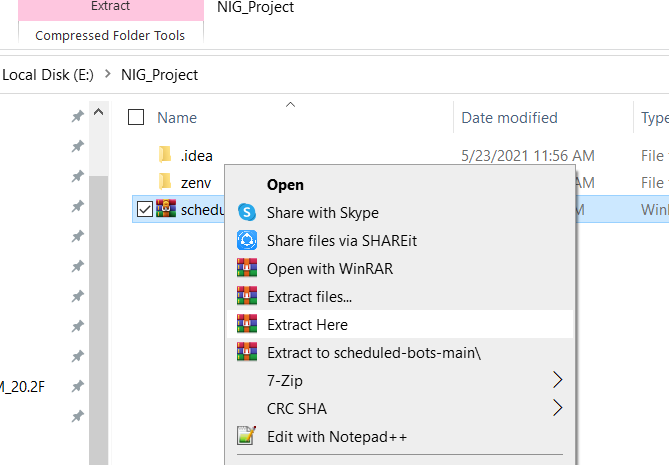
1. In “wdi\_config.py” change the **config** array (in line 21) as follows.

config = {  
 **'BACKOFF\_MAX\_TRIES'**: **None**,  
 **'BACKOFF\_MAX\_VALUE'**: 3600,  
 **'USER\_AGENT\_DEFAULT'**: **'wikidataintegrator/{}'**.format(\_\_version\_\_),  
 **'MAXLAG'**: 5,  
 **'PROPERTY\_CONSTRAINT\_PID'**: **'P2302'**,  
 **'DISTINCT\_VALUES\_CONSTRAINT\_QID'**: **'Q21502410'**,  
 **'COORDINATE\_GLOBE\_QID'**: **'http://www.wikidata.org/entity/Q2'**,  
 **'CALENDAR\_MODEL\_QID'**: **'http://www.wikidata.org/entity/Q1985727'**,  
 **'MEDIAWIKI\_API\_URL'**: **'http://localhost:8181/w/api.php'**,  
 **'MEDIAWIKI\_INDEX\_URL'**: **'http:// localhost:8181/w/index.php'**,  
 **'SPARQL\_ENDPOINT\_URL'**: **'http://localhost:8282/proxy/wdqs/bigdata/namespace/wdq/sparql'**,  
 **'WIKIBASE\_URL'**: **'http://wikibase.org'**,  
 **'CONCEPT\_BASE\_URI'**: **'http:// localhost:8181/entity/'**,  
 **'WIKIDATA\_SPARQL\_ENDPOINT\_URL'**: **'https://query.wikidata.org/bigdata/namespace/wdq/sparql/'**,  
}

1. Go to <https://github.com/SuLab/scheduled-bots> and download the scheduled-bots source click on the green “Code” button on the resulted page. The dropdown menu, chooses the “Download ZIP” option and save it in the project folder (E:\NIG\_Project).



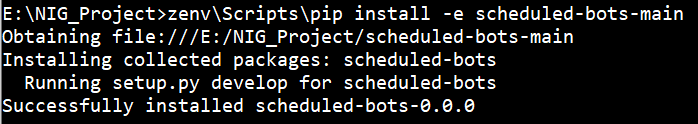
1. Extract scheduled-bot-main.zip into the project folder by choosing the “Extract Here” option from the right-click menu.



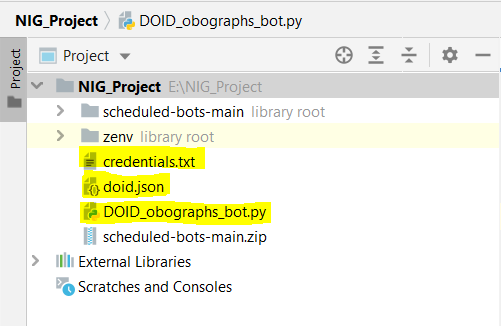
1. Open command prompt, navigate to project folder, and issue following command to install scheduled-bots package.

* **zenv\Scripts\pip install -e scheduled-bots-main**

If installation successful it shows a success message as shown in the above figure.



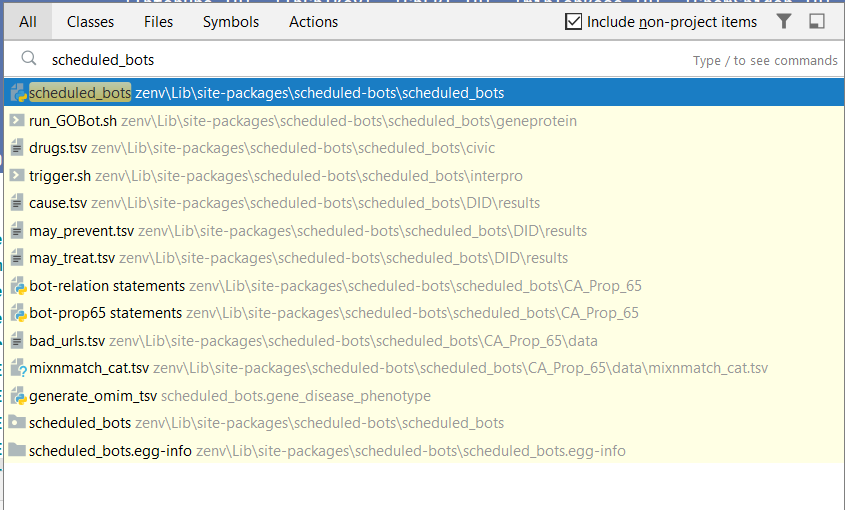
1. Copy and paste the following project files into the NIG\_Project.



1. To find “scheduled\_bots\\_\_init.py” file go to **MainMenu -> Navigate -> Search Everywhere**

In the search window type “scheduled\_bots” as follows and open it.

**Note:** Please choose the selected option as shown in the following figure.



1. Add the following lines to it (“scheduled\_bots\\_\_init.py”).

**Note: Replace the new arrays with existing ones.**

DEFAULT\_PROPS = [**'reference URL'**,**'equivalent property'**]  
  
DEFAULT\_ITEMS = [**'version, edition, or translation'**,**'Disease Ontology'**,**'Identifiers.org Registry'**,**'disease'**,**'distinct-values constraint'**]  
  
PROPS\_NAMES = [**'property constraint'**,**'edition number'**,**'edition or translation of'**,**'publication date'**,**'ATC code'**,**'archive URL'**,**'authority'**,**'approved by'**,**'CAS registry number'**,**'CIViC Variant ID'**,**'ChEBI ID'**,**'ChEMBL ID'**,**'ChemSpider ID'**,**'DOI'**,**'Disease Ontology ID'**,**'Drugbank ID'**,**'drug used for treatment'**,**'EC Number'**,**'Encoded By'**,**'Ensembl Gene ID'**,**'Ensembl Protein ID'**,**'Ensembl Transcript ID'**,**'Entrez Gene ID'**,**'Entrez Gene ID'**,**'GARD rare disease ID'**,**'Gene Ontology ID'**,**'genomic end'**,**'genomic start'**,**'HGNC gene symbol'**,**'HGNC ID'**,**'HGVS nomenclature'**,**'HGNC gene symbol'**,**'Human Phenotype Ontology ID'**,**'ICD-10'**,**'ICD-10-CM'**,**'ICD-9'**,**'ICD-9-CM'**,**'Guide to Pharmacology Ligand ID'**,**'InChI'**,**'InChIKey'**,**'incidence'**,**'InterPro ID'**, **'canonical SMILES'**,**'KEGG ID'**,**'Mouse Genome Informatics ID'**,**'MeSH tree code'**,**'MeSH descriptor ID'**,**'Mondo ID'**,**'NCBI Locus tag'**,**'NCBI taxonomy ID'**,**'NCI Thesaurus ID'**,**'NDF-RT ID'**,**'National Cancer Institute ID'**,**'OMIM ID'**,**'Orphanet ID'**,**'PDB structure ID'**,**'PMCID'**,**'image'**,**'PubChem CID'**,**'PubMed ID'**,**'RTECS Number'**,**'Rat Genome Database ID'**,**'RefSeq RNA ID'**,**'Refseq Genome ID'**,**'Refseq Protein ID'**,**'RxNorm ID'**,**'canonical SMILES'**,**'Saccharomyces Genome Database ID'**,**'Sequence Ontology ID'**,**'start time'**,**'UBERON ID'**,**'UMLS CUI'**,**'UNII'**,**'UniProt protein ID'**,**'WikiPathways ID'**,**'anatomical location'**,**'biological process'**,**'biological variant of'**,**'cell component'**,**'chemical formula'**,  
**'chromosome'**,**'curator'**,**'cytogenetic location'**,**'determination method'**,**'develops from'**,**'encodes'**,**'equivalent property'**,**'equivalent class'**,**'exact match'**,**'found in taxon'**,**'genetic association'**, **'genomic assembly'**,**'genomic end'**,**'genomic start'**,**'has cause'**,**'has part'**,**'has phenotype'**,**'homologene id'**,**'instance of'**,**'location'**,**'medical condition treated'**,**'miRBase mature miRNA ID'**,**'miRBase pre-miRNA ID'**,**'mirTarBase ID'**,**'molecular function'**,**'negative diagnostic predictor'**,**'negative prognostic predictor'**,**'negative therapeutic predictor'**,**'observed in'**,**'official website'**,**'ortholog'**,**'pathogen transmission process'**,**'parent taxon'**,**'part of'**,**'positive diagnostic predictor'**,**'positive prognostic predictor'**,**'positive therapeutic predictor'**,**'rating'**,**'reference URL'**,**'regulates (molecular biology)'**,**'retrieved'**,**'route of administration'**,**'stated in'**,**'statement disputed by'**,**'subclass of'**,**'subject has role'**,**'symptoms'**,**'taxon name'**,  
**'uberon id'**,**'IEDB Epitope ID'**,**'ChEBI ID'**,**'encoded by'**,**'RefSeq Protein ID'**,**'UniProt protein ID'**,**'strand orientation'**,**'HGNC ID'**,**'HGNC Gene Symbol'**,**'HomoloGene ID'**,**'Mouse Genome Informatics ID'**,**'Mouse Genome Informatics ID'**,**'Wormbase Gene ID'**,**'FlyBase Gene ID'**,**'ZFIN Gene ID'**]  
  
PROPS = {}  
  
DEFAULT\_CORE\_PROPS = [**'NCBI taxonomy ID'**, **'National Cancer Institute ID'**, **'UNII'**, **'MeSH tree code'**, **'Guide to Pharmacology Ligand ID'**, **'archive URL'**, **'UniProt protein ID'**, **'RefSeq RNA ID'**, **'Entrez Gene ID'**, **'DOI'**, **'Refseq Genome ID'**,**'image'**, **'NCBI Locus tag'**, **'PMCID'**, **'PubMed ID'**, **'NCI Thesaurus ID'**,**'InterPro ID'**, **'InChIKey'**, **'ChEBI ID'**, **'mirTarBase ID'**, **'ChemSpider ID'**, **'Mouse Genome Informatics ID'**, **'miRBase mature miRNA ID'**, **'Disease Ontology ID'**, **'PubChem CID'**, **'HGNC ID'**, **'RTECS Number'**, **'NDF-RT ID'**, **'Gene Ontology ID'**, **'Drugbank ID'**, **'miRBase pre-miRNA ID'**, **'KEGG ID'**, **'InChI'**, **'MeSH descriptor ID'**, **'HGNC gene symbol'**, **'ChEMBL ID'**, **'Orphanet ID'**, **'CIViC Variant ID'**, **'WikiPathways ID'**]  
  
DEFAULT\_CORE\_PROPS\_PIDS = {}

1. To install the following required packages repeat above steps (5, 6, 7 and 8) recursively.

* cachetools
* shexer
* chardet

1. To find wikibase\_helper.py file go to **MainMenu -> Navigate -> Search Everywhere**

In the search window type “wikibase\_helper.py”. Find and open it.

Add following line to the import

**from** wikidataintegrator **import** wdi\_config

In line 34 add the following codes.

**if** uri\_qid == **None**:  
 uri\_qid = dict()

In line 56 replace existing if condition by following codes.

**if** (self.sparql\_endpoint\_url == wdi\_config.config[**'SPARQL\_ENDPOINT\_URL'**] **and** (uri.startswith(**"P"**) **or** uri.startswith(wdi\_config.config[**'CONCEPT\_BASE\_URI'**]))):

In line 66 replace existing if condition by following codes.

**if** (self.sparql\_endpoint\_url == wdi\_config.config[**'SPARQL\_ENDPOINT\_URL'**] **and** (uri.startswith(**"Q"**) **or** uri.startswith(wdi\_config.config[**'CONCEPT\_BASE\_URI'**]))):

1. To find release.py file go to **MainMenu -> Navigate -> Search Everywhere**

In the search window type “release.py” Find and open it.

Add the following line to the import

**from** scheduled\_bots **import** PROPS,ITEMS

Codes from line 73 to line 83 replace with following lines of codes.

s.append(wdi\_core.WDItemID(helper.get\_qid(ITEMS[**'version, edition, or translation'**]),  
 helper.get\_pid(PROPS[**'instance of'**])))  
*# edition or translation of*s.append(wdi\_core.WDItemID(self.edition\_of\_qid, helper.get\_pid(PROPS[**'edition or translation of'**])))  
*# edition number*s.append(wdi\_core.WDString(self.edition, helper.get\_pid(PROPS[**'edition number'**])))  
  
**if** self.archive\_url:  
 s.append(wdi\_core.WDUrl(self.archive\_url, helper.get\_pid(PROPS[**'archive URL'**])))  
  
**if** self.pub\_date:  
 s.append(wdi\_core.WDTime(self.pub\_date, helper.get\_pid(PROPS[**'publication date'**]),  
 precision=self.date\_precision))

Around line 99 find id\_mapper method call and replace it by adding following codes.

edition\_dict = id\_mapper(helper.get\_pid(PROPS[**'edition number'**]),  
 ((helper.get\_pid(PROPS[**'edition or translation of'**]), self.edition\_of\_qid),  
 (helper.get\_pid(PROPS[**'instance of'**]),

helper.get\_qid(ITEMS[**'version, edition, or translation'**]))), endpoint=self.sparql\_endpoint\_url)

1. To find update\_retrieved\_if\_new.py file go to **MainMenu -> Navigate -> Search Everywhere**

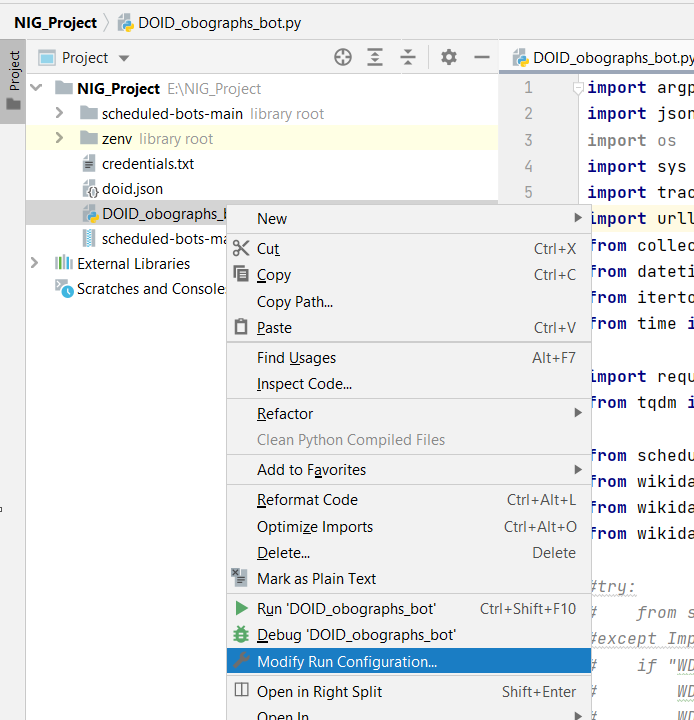
In the search window type “update\_retrieved\_if\_new.py” Find and open it.

Comment print statements in lines 46 and 49 by adding # in front of it as follows.

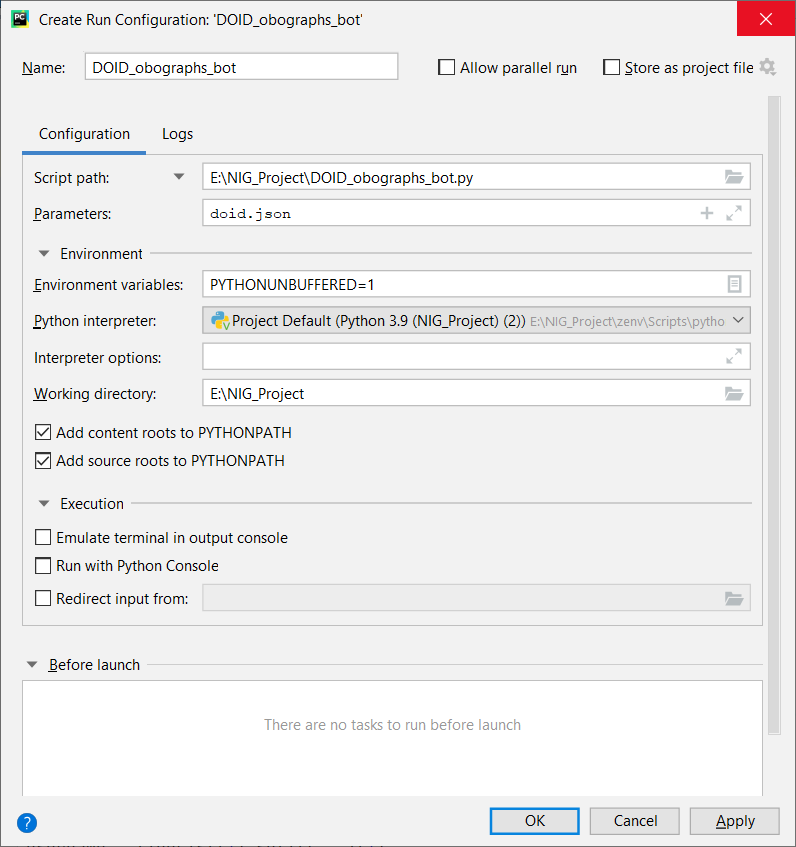
#print("updating ref")

#print("don't change")

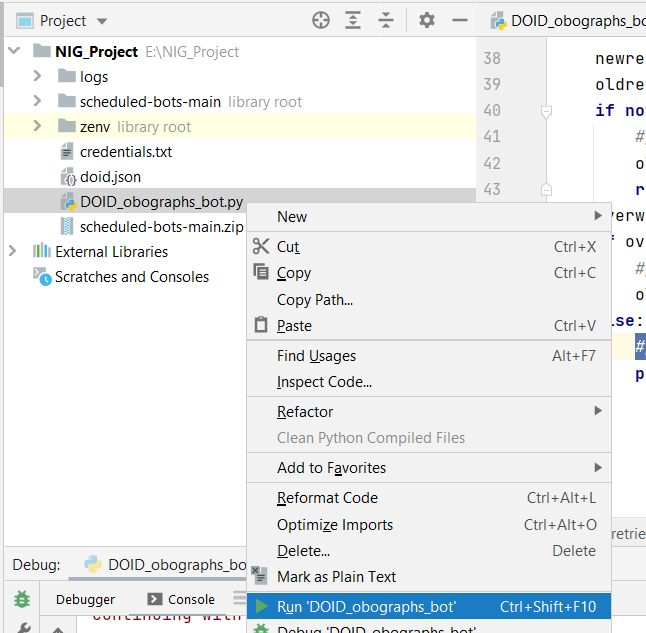
1. To set the run parameters to the project right click on the “DOID\_obographs\_bot.py” in the left side project explorer menu and choose Modify Run Configuration option from the resulted menu as shown below.



1. In resulted window set “doid.json” as Parameters and click Ok.



1. After setting run parameters, to run the script right click on “DOID\_obographs\_bot.py” file in left side menu and click on Run “DOID\_obographs\_bot”.



1. To view the changes in the wikibase use the following link.

<https://wb.ddbj.nig.ac.jp/wiki/Special:RecentChanges>

For fresh wikidata start

1. SSH to the server where wikidata docker instance hosted.
2. Move to the folder where docker-composer file exists.

**cd /usr/bin**

1. Remove the existing wikidata

**docker-compose down --rmi all --volumes --remove-orphans**

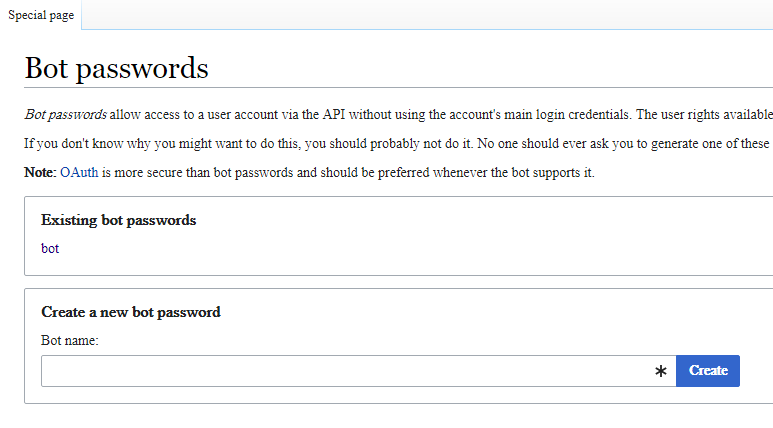
1. Restart the containers.

**docker-compose up -d**

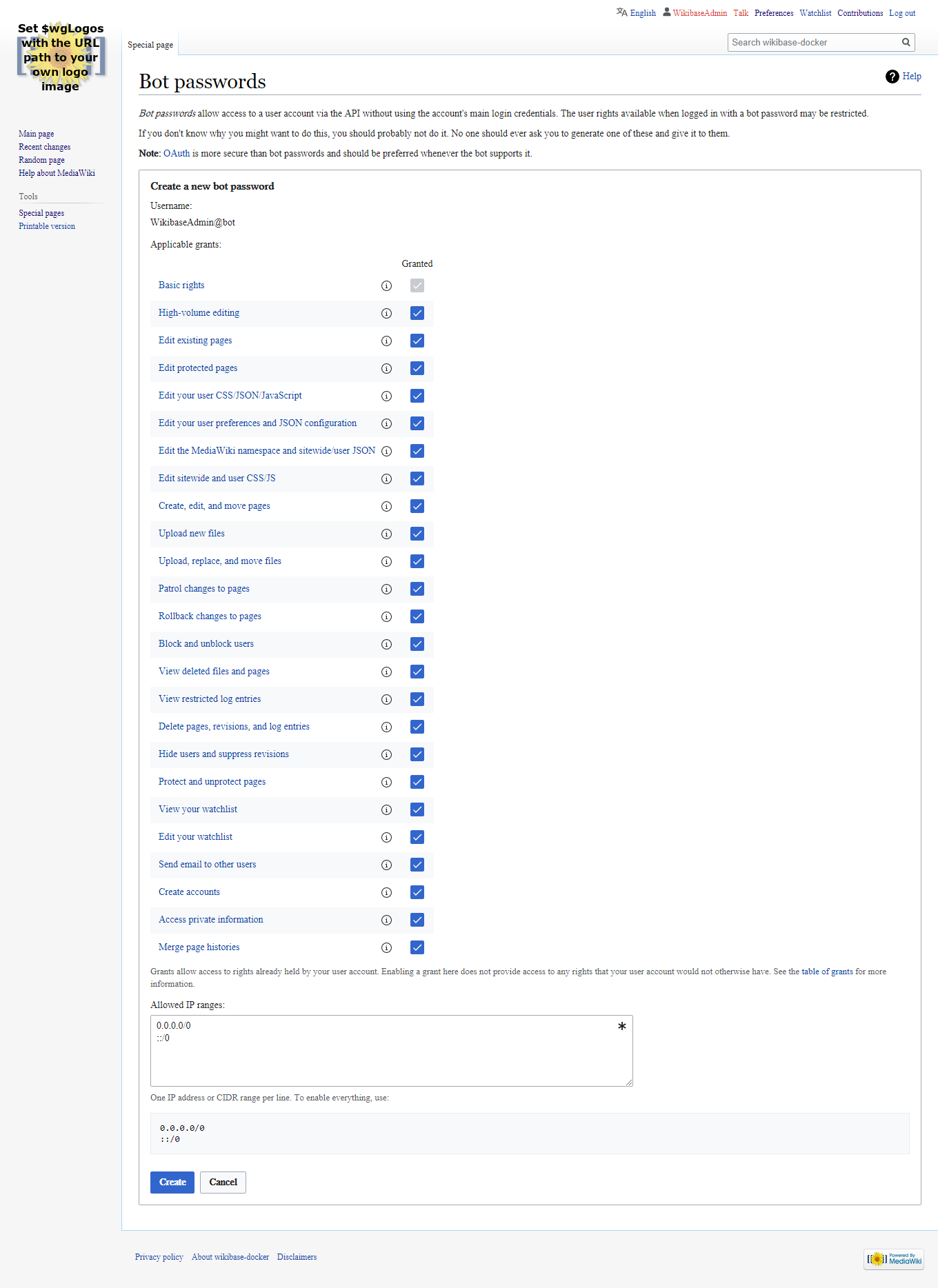
1. To get a new **wikidata bot** password login into the wikidata instance at <http://124.43.16.118:8181>

**Username:** WikibaseAdmin **Password:** WikibaseDockerAdminPass

1. Go to the Bot password page. <http://124.43.16.118:8181/wiki/Special:BotPasswords>



1. Give a bot name and click on create. As resulted windows check all the permissions and click create button.



1. Replace the new bot password with the existing one in “credentials.txt” file in the project.

OWL to OBO Graphs JSON

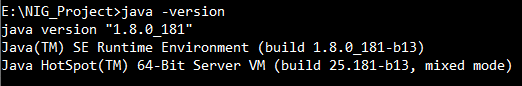
Reference to OBO Graphs JSON : <https://github.com/geneontology/obographs>

**Prerequisites**

* Java 8

1. To check java is correctly installed open the command prompt and issue the following command.

**Java** –version



1. Then run obographs.jar with the following command.

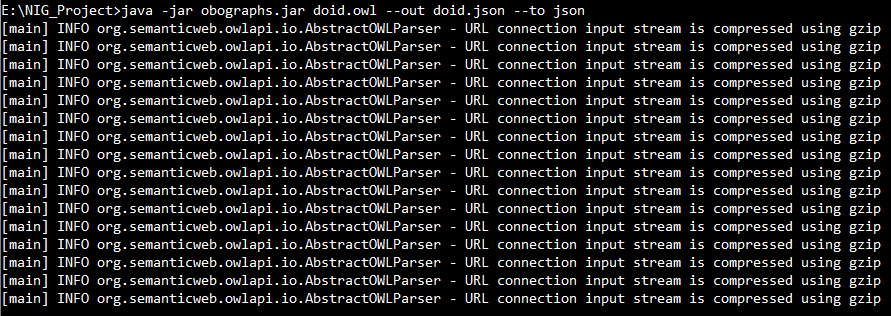
**java -jar** obographs.jar doid.owl --out doid.json --to json

Command line argument details:

**First argument** – file path of the OWL file

**--out** – file path of the output JSON/YAML

**--to** – file type of the generated file | options json or yaml



The above command creates an OBO Graph JSON file with the given name.

**Note:** This generated OBO Graph JSON can be used as an input to DOID\_obographs\_bot.py