



VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Nhi Le

AUTOMATED JIRA BACKLOG PROCESS USING PYTHON

School of Technology

2022

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Information Technology

ABSTRACT

| | |
|--------------------|---|
| Author | Nhi Le |
| Title | Automated Jira Backlog Process Using Python |
| Year | 2022 |
| Language | English |
| Pages | 37 |
| Name of Supervisor | Mikael Jakas |

The topic of the thesis is creating a background application to automate a backlog (tasks management) process in the Jira server which is hosted by Nokia. The application was almost completed and is now run on the server, then in the future will continue to be maintained and updated with a more comprehensive version. The main purpose of the project was to minimize the workload of the coordinator, who must divide or categorize tasks manually.

The Jira software which was a task management software provides a Python API. Therefore, to take advantage of this, the main programming language was Python 3, it comprises some necessary packages and libraries. Jira Query and a small percentage of Shell script were also supported this project.

While developing this application, TDD (Test Driven Development) was used, it helped developers significantly reduce unworthy bugs, and in the process afterward, the maintainer by reading the test cases will have a comprehensive understanding of the source code.

The application is now running successfully and is still in process of developing more interesting functions. This automation is targeted to remove manual manipulation, therefore there is no scope for this application, as long as there was still an issue that can be solved by coding, the application will be updated and released in the latest version. We aim to bring the application to become a full-functioned bot.

Keywords Python, software development, testing methods

CONTENTS

| | |
|---|----|
| 1. INTRODUCTION | 7 |
| 2. TECHNOLOGY OVERVIEW | 9 |
| 2.1 Python..... | 9 |
| 2.2 Jira | 10 |
| 2.3 Jira Issue | 11 |
| 2.4 Jira Issue Type..... | 12 |
| 2.5 Jira Workflow..... | 13 |
| 2.6 JQL | 14 |
| 3. PROJECT DETAILS | 15 |
| 3.1 Project Overview..... | 15 |
| 4. STRUCTURE | 18 |
| 4.1 Abstract Base Classes in Python | 18 |
| 4.2 Classes Work Details | 20 |
| 5. USE CASES | 23 |
| 5.1 Assigning Pre-check Tasks | 23 |
| 5.2 Overdue Issue Mail Inform | 25 |
| 6. TESTING PROCESS | 27 |
| 6.1 Unit Test:..... | 27 |
| 6.2 Test Driven Development | 28 |
| 6.2.1 Pytest and Unittest..... | 29 |
| 7. DEPLOYMENT | 34 |

7.1 Virtual environment 34

7.2 Crontab 34

8. CONCLUSIONS 35

9. REFERENCES 36

LIST OF FIGURES

| | |
|--|----|
| Figure 1. Simple Jira Issue. | 11 |
| Figure 2. Basic of Jira Issue hierarchy. | 13 |
| Figure 3. Jira Workflow sample..... | 13 |
| Figure 4. Example some queries in Jira from basic to advanced..... | 14 |
| Figure 5. Overview structure of POBot. | 16 |
| Figure 6. POBot Python Class Diagram. | 18 |
| Figure 7. Logfile sample. | 22 |
| Figure 8. Sequence diagram displays the structure of assigning tasks in POBot. | 23 |
| Figure 9. The expected output of the POBot in the log file..... | 24 |
| Figure 10. Unassigned Issue..... | 24 |
| Figure 11. Issue before and after updating the assignee field. | 24 |
| Figure 12. Sequence diagram displays structure of email sending task in POBot. | 25 |
| Figure 13. Reminded email sample. | 26 |
| Figure 14. Log file message in Overdue Mail task. | 26 |
| Figure 15. Unit Test structure. | 27 |
| Figure 16. TDD cycle..... | 29 |
| Figure 17. Simple calculate function in Python. | 30 |
| Figure 18. Test file of main.py. | 31 |
| Figure 19. Run single test..... | 32 |
| Figure 20. Result after running command..... | 32 |

LIST OF CODE SNIPPETS

| | |
|---|----|
| Snippet 1. ‘AuthorizationBase’ class which is an Abstract Base Class..... | 19 |
| Snippet 2. ‘Authorization’ class which is the child class of class ‘AuthorizationBase’. | 19 |

| | |
|--|----|
| Snippet 3. The function that handles the operation of tasks. | 20 |
| Snippet 4. Basic Jira class format. | 21 |
| Snippet 5. Command for installing testing library in Python. | 30 |

1. INTRODUCTION

Nowadays, Task Management software is an essential tool. It helps each developer, their team and further their organizations to manage the work more efficiently. By providing users a regulated working environment where all the work can be visual as a form of task, those kinds of task reduce time wasted using for organizing works manually. Hence, developers can be utilized wisely and meet the deadline.

With the outburst of modern technology leading to a high demand for software/technical job, many kinds of Task Management Software are released annually with more and more comprehensive version. Most of them offer for service with a certain price, companies consider which one will be the good option for them. However, many tools include some free-of-charged service, if the scale of organization is not too big, the user can be satisfied with some of free service.

However, there is not a perfect Software that can manage every need of a company, especially big and old companies who have such a complicated structure and heavy working traffic. In this case, many of Task Management Software provide the user with their REST API which can be used to interact remotely with the application, allow an organization to access their service by programming languages and easily custom their own function or at least automate some process that only exist within internal organizations.

In the Nokia sites, one of the main Task Management Software they are currently using is Jira Software. In some specific departments, employees are dealing with a large number of related tasks, there is an employee called the Coordinator, who is responsible for viewing, deciding then creating tasks (which is known as issues in Jira) and assigning to other Nokia employees. Primarily this is a backlog management process. However, it is considered impractical in Nokia as there are up to a hundred issues and it will require plenty of manual work, by taking advantage of REST API from

Jira and software development, the best solution would be building an application to automate this process.

2. TECHNOLOGY OVERVIEW

The technology used in PObot is considered quite simple, as it was built from scratch so that to make it is easier for future maintenance, the technology options should be independent from framework. Strongly focus on creating a firm base for interaction with the Jira server and utilize available offers from JIRA library.

2.1 Python

Python is acknowledged as a high-level language which means it is much simpler to use, more user-friendly context and independent from hardware components such as addressing or memory. Therefore, this programming language rather focuses on the logic part than defines hardware architecture for program. Python is used widely because of its following superior features.

- Ease: Python provides users with a clear and simple syntax allowing them to catch up with the code quickly from that user can find it easy to maintain the code as well.
- Diversity of library: Python contains many libraries in many fields such as automation, machine learning, AI and Web development. It well-supported in many ranges of careers.
- Strongly supports OOP.
- Easily integrated with other languages for example C, C++, Java, etc

Object-oriented-programming is a programming model where an 'object' is used instead of just simple functions or scripts. An object will have its own attributes and behaviours and define to run based on users need.

OOP has its own advantages which will support in a big project which requires complicated structures:

- Encapsulation: allows to hide information and the other internal process of object. This feature enhances security of an object and avoid data breaking.
- Inheritance: This feature allows to create child classes.

The reason for choosing Python as main technology in this project firstly because Jira's API options. Jira provides developers its API in Java and Python. Then in terms of technical Python will be the better option with its easy-to-work feature, moreover Python also provide a library which is JIRA to support methods and calls from Jira.

2.2 Jira

Jira is considered as one of most effective Software Management tools nowadays, it helps people in a software team to manage, track, and report their working process, workload and even working status, and share them with every team member. Jira Software was developed by Atlassian first in 2002, since then, the software has been changed and improved a lot to offer the best fit adapting user expectation (Atlassian, 2020). Because of that, in many big or long-term companies, so far Jira is still their best choice for main server, though recently many new Task Management Software has been released. The Jira Software package is Agile teams, Bug tracking, Project, Product, Process, Task and management.

Atlassian also owned such a large community where the users of any of its products (Jira, Jira Align or Confluence) can connect with each other, and even raise questions or issues they are currently facing to get help from the community. The support may be received from a master Jira user or even Atlassian IT help desk can give users a hand. The user can also leave feedback there, and the Atlassian team can know their user trend and try to improve the software day by day.

The Jira Software provides users with its Jira REST API in Python and allows users to interact with the Jira Server remotely. Therefore, by running simple scripts together with an appropriate permission account we can modify a field of an issue without really touching them in terms of GUI. From that, developers can take advantage of this and develop or automate the process as they would like to.

2.3 Jira Issue

Jira Issue is a working item that contains all the relevant working information including dates and people; Jira Issue also contains links to another system of Jira Product such as Confluence or Bamboo. Teams use these issues to keep a track of individual process and working load. An issue in Jira can be called as a task, ticket, request.

The user can carry on a very basic action in Jira Issue such as create, delete, edit and modify. A completed issue as normal looks as shown in Figure 1:

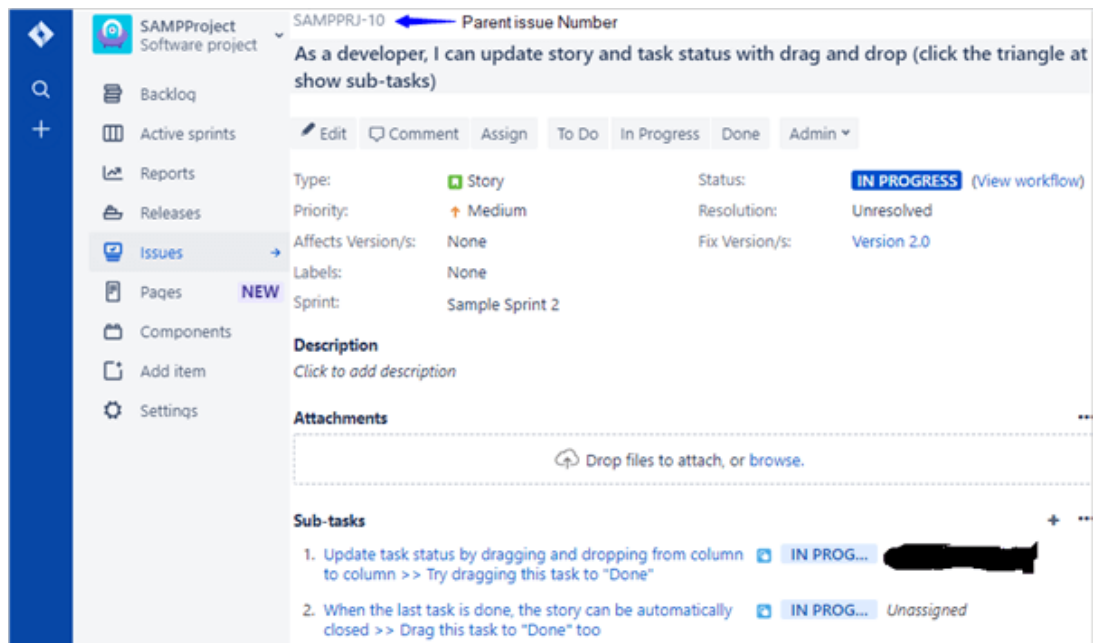


Figure 1. Simple Jira Issue.

The information displayed in a Jira Issue such as 'Type', 'Priority', 'Description', or even the link below 'Sub-tasks' area are called 'Fields' in terms of technical. These fields can be either filled, created manually by the user or handle by code. On the other hand, a Jira issue is supposed to made up of many fields and they cannot all appear at the same in a specific issue, it will depend on user to choose any number of fields to display in their own issue while creating and editing.

2.4 Jira Issue Type

In this project, we mainly use Jira Software, which is one of Jira Product, therefore the Issue Type in each product will be a bit different. Jira Software provides user 5 types of Issue: Epic, Bug, Story, Task, Sub Task.

- Epic: in terms of agile, epic is a big user story or project which will be divided into small bugs, stories and then to tasks. Epic can be considered as a parent issue.
- Bug: As describe as its name, a bug is an problem which blocks process of a task/ticket/issue.
- Story: A story can be included many of tasks.
- Task: Task is used for plan specific work which is not exceed few days of working. Related task will be connected to a story. A task can be transfer to person to person for review and mark 'Done' if it is completed.
- Sub task: Task can be broken down to smaller task which is called sub-task

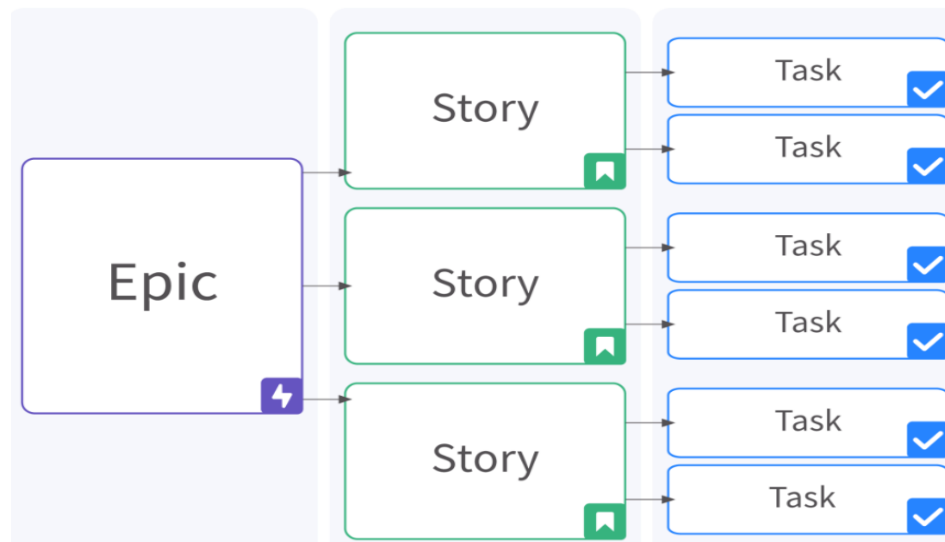


Figure 2. Basic of Jira Issue hierarchy.

2.5 Jira Workflow

Workflow is a sequential path of task indicating the status of an issue from its beginning till completion. A basic workflow format is illustrated in Figure 3.

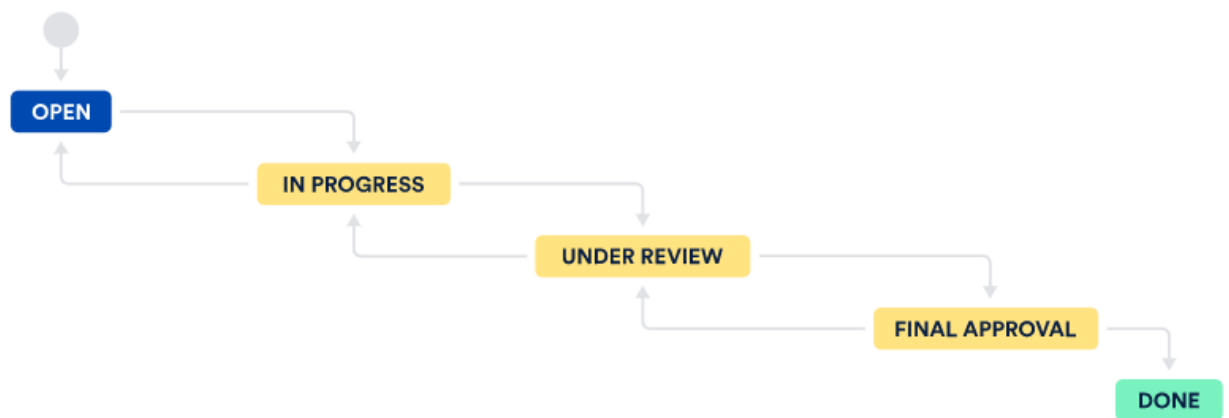


Figure 3. Jira Workflow sample.

The workflow of each issue will be different, depending on what type of issue is and how the user or team would like it to be. However, some basic statuses always performed such as 'Open', 'Done', or 'In Progress'. In some complicated and big

projects, the workflow may contain some statuses such as 'Re-open', 'Re-check', 'Closed', or 'Obsoleted'.

2.6 JQL

JQL is shorted for Jira Query Language, and it has the same function as the other query languages, which is used for searching and extracting information. Each query language will have their own syntax and specific use. Jira Query Language is a term to indicate that the query is only used in Jira, developed to provide the user best suitable syntax for searching issues (Dan, 2020). JQL is for everyone even developers, testers, business users that are inside the Jira service.

A basic query in JQL contains a **field**, an **operator** and **values** or **function** which specifically point to current working data that a team owns.

✓ **project = Test**

field operator value



To perform a more complex query, you can link clauses together with **keywords**.

✓ **project = TEST AND assignee in (currentuser())**

field operator value keyword field operator function

Figure 4. Example some queries in Jira from basic to advanced.

3. PROJECT DETAILS

In Nokia, the scale of workflow is extremely large, they have many products/projects on the line at the same time. Each department will handle a part of the projects and they need to manage it separately before merging these parts together and releasing a completed product and launching it. However, in every release version, there is always feedback from the manager or customers, they would like to make some change as the product may not meet their requirements yet. This will make the Task Management become difficult to control, the coordinator must again deal with the re-modification request and decide what further action will be with those tasks that have been completed before.

For example, in a basic issue there will be fields named 'Status', which indicates the current process of an issue. When the issue is first completed, the user will set the Status into 'Done' or 'Close' (depending on which has been set as a default). After the first feedback come, customers would like to add a new function in the product. Therefore, in Jira, the status of this issue needs to be set to 'Open' or 'Re-open', the description of the issue will also need to change, more requests will be added, and a new date of deadline will be set.

3.1 Project Overview

Figure 5 is a diagram expressing the basic idea about application. Following the figure are detailed explanations for each of components.

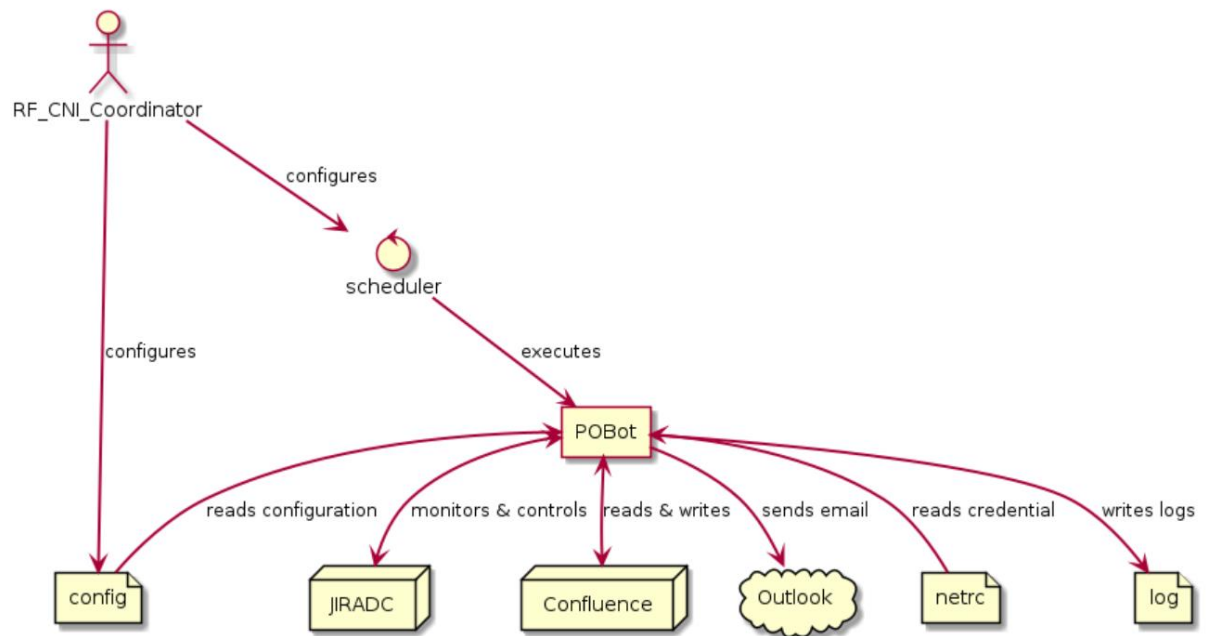


Figure 5. Overview structure of POBot.

Coordinator is a person who designs the task and an assignee to related employees
'PObot' is the name of this application, it can be considered as a bot that simulating the work of the coordinator.

Configure file generally is a json file used for storing necessary data or internal rules given by the coordinator. One of the main data is the mapping between assignees (Nokia employees) and the name of Product line they are in charge of or provide Nokia's server option.

JiraDC and **Confluence** are both products of Atlassian, they are main servers using for Task/Work Management. The results of POBot will be display under those servers.

Outlook is an email server, PObot will send mail to an assignee via Outlook to remind that Issues/Tasks they took responsibility for before having been Re-Opened now and ask for their modification, or simply just inform them that their issues overdue soon.

Netrc file is simply a text file containing the username and password under a specific format. The netrc file is the current option for this application. Even though the netrc file is considered insecure, however at this version of the bot, we have not got any token for making them more effective. Therefore, by taking advantage of the fact that the working scope of this application is inside the Nokia network, to some extent this method could be acceptable.

Log file is also a text file for storing the output. Because this application does not carry any Front-End work, Jira will display for us the output as an Issue. However, in case errors happen there is nothing to show or inform the developer where the error come from. To handle this, a simple text file is applied. The text file will log every result of POBot whether it failed or passed, in case it fails, this file has been added a function to notify to developers by their email, so that they are able to fix it. The Log file also have a mechanism to automatically clean up whenever the number of messages inside reach 100 messages. It helps the owner to keep track in what PObot is currently doing conveniently.

To conclude, the coordinator will set rules or 'needed knowledge' of the bot in a json file called a Config file, the bot will use Python to execute Tasks given by Coordinator based on data inside the config file, the result is the Jira Issues after being changed by the bot. The bot may send the email to the user for informing if it needed. Every time the bot runs all the output will be stored in the log file for certain period.

4. STRUCTURE

The Figure 6 below is a class diagram that shows the initial structure of the PObot, which includes six components, defines how the looks like under Python classes and what is the function of each of them:

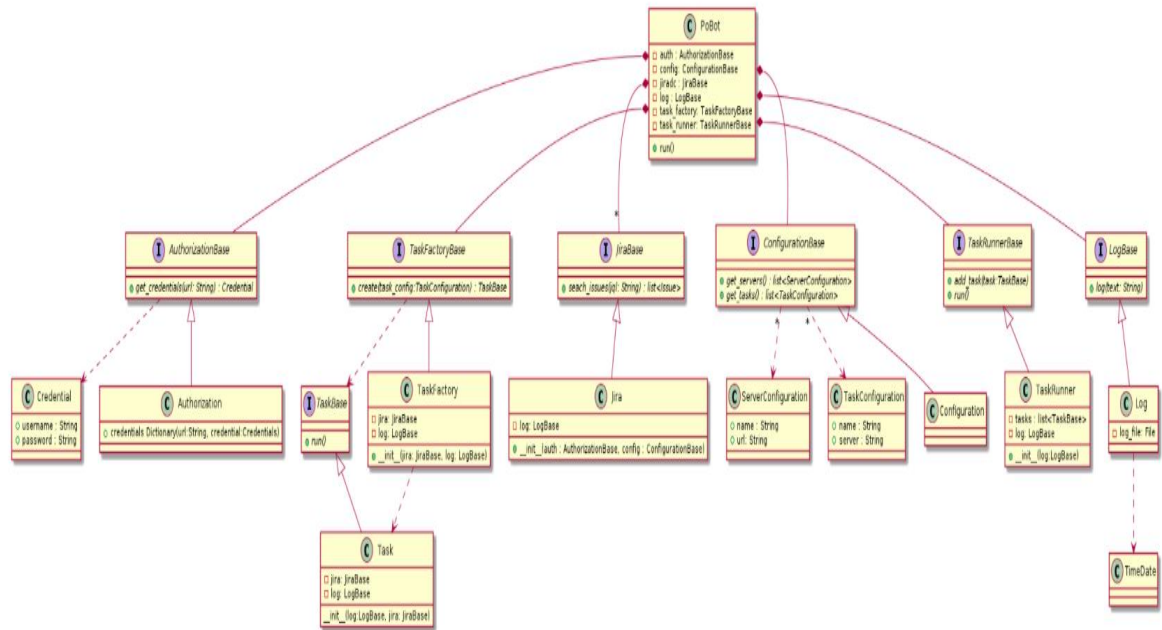


Figure 6. POBot Python Class Diagram.

4.1 Abstract Base Classes in Python

This project used pure Python, utilized its Object-oriented programming and some simple Python library. The image is the main structure in code, the bot contains five main bases, each of base has their own classes and dependencies. The base here means Abstract Class which is a class where we define defaults function without any declaration. They can be defined and customized in their child classes. The Abstract Class will support well for projects have a large scope, it makes the implementation become simpler and easy to create test cases.

Python does not provide Abstract Class, however, there is a module called Abstract Base Classes (ABC) that can support Python users with this. This module belongs to package 'ABC', therefore, to use it, the user needs to import the package.

```

1  import abc
2
3
4  class AuthorizationBase(metaclass=abc.ABCMeta):
5      @classmethod
6      def __subclasshook__(cls, subclass):
7          return (hasattr(subclass, 'get_credentials') and
8                  callable(subclass.get_credentials) or
9                      NotImplemented)
10
11     @abc.abstractmethod
12     def get_credentials(self, text):
13         pass

```

Snippet 1. 'AuthorizationBase' class which is an Abstract Base Class.

In this abstract class, an abstract method/function is defined, and passed with no further implementation.

```

1  import netrc
2  from AuthorizationBase import AuthorizationBase
3
4
5  class Authorization(AuthorizationBase):
6      def __init__(self, argument=None):
7          self.netrcInstall = netrc.netrc(argument)
8
9      def get_credentials(self, serverName):
10         return self.netrcInstall.authenticators(serverName)
11

```

Snippet 2. 'Authorization' class which is the child class of class 'AuthorizationBase'.

The function `get_credentials()` is now implemented in this child class, the symbol next to line number 9 indicates that this function is already overridden from class 'AuthorizationBase'.

4.2 Classes Work Details

Authorization class points to the netrc file and provides a function that can access to the 'netrcInstall' module to get back the username and the password. The account contains a username, and the password is provided by the coordinator; it is a fully accessed account with the necessary permission to run inside main servers.

Tasks is a separated folder consisting of multiple Python files, these python files are the use cases that will be designed by a coordinator, each of the Python files in the Task folder is in charge of one specific modification in Jira; however, in general there are 2 types of tasks, one will be trigger hourly, another is daily. The nature of this bot always finds a specific group of issues and applying needed modifications on it. Therefore, they need to be triggered in a continuous-time, sometimes issues may not exist, however in the next search they can appear again.

One common specification of these Python files is they all have the same class name as 'Task' and inherited 'TaskBase' class, and a function name 'run'. Their module name will display the real name of that task.

```
12 def run(self):
13     for task in self.tasks:
14         try:
15             task.run()
16         except Exception as e:
17             if self.log:
18                 self.log.log(e)
19
20 def add_task(self, tasks):
21     for task in tasks:
22         try:
23             module = importlib.import_module('Tasks.' + task)
24             class_ = getattr(module, 'Task')
25             self.tasks.append(class_(self.objectList))
26         except Exception as e:
27             if self.log:
28                 self.log.log(e)
```

Snippet 3. The function that handles the operation of tasks.

TaskRunner is the place where the bot defines the way to trigger all the tasks. Here, there is an empty list, every task will be added and then operated one by one. The reason why in **the Tasks** folder, all classes share the same class name, and the function name is that it will make it easier to trigger all tasks via the list.

Each task in 'self.tasks' as a list, is the module of task where they have the same class name and function name. Then in line 15, by sharing the same function name, now all the tasks can be called to run at one easily.

Jira is the class that is inherited from the JIRA class which is a Python library for connecting and fetching data from the real application. To import and use, this library requires at least Python version 3.5 to install Jira used followed command:

```
pip install jira
```

When the library is ready to be used, we need to import, create a JIRA client instance and then pass the Authentication parameters which will be taken from the 'Authentication' class. The Jira class will have the basic format shown in Snippet 4 below.

```
# import the installed Jira library
from jira import JIRA

# Specify a server key. It should be your
# domain name link. yourdomainname.atlassian.net
jiraOptions = {'server': "https://txxxxxpython.atlassian.net"}

# Get a JIRA client instance, pass,
# Authentication parameters
# and the Server name.
# emailID = your emailID
# token = token you receive after registration
jira = JIRA(options=jiraOptions, basic_auth=(
    "prxxxxxh@gmail.com", "bj9xxxxxxxxxxxxxxxxxxxx5A"))
```

Snippet 4. Basic Jira class format.

Inheriting JIRA class, we now can override and custom functions from real Jira based on project needs. **Log** class owns functions defining the format of the Log file.

```
2022-04-07 19.23.54 Assign CDVL Task Initialize:
2022-04-07 19.23.54 CVDL-771241: assigned to David Marca
2022-04-07 19.23.54 CVDL-771111: assigned to Emily Ganrit
2022-04-07 19.23.55 CVDL-124151: assigned to Jaguv Sean
2022-04-07 19.23.56 Overdue issues email sent :
2022-04-07 29.23.57 VHH-671331: email sent to David Marca
2022-04-07 19.23.57 CDD-771211: email sent to Marek Bulzo
2022-04-07 19.23.57 CDD-131222: email sent to Pekka Lummark|
-----
2022-04-07 20.23.54 Initialization Done:
2022-04-07 20.23.54 assign_un_pre-check_task: No new issues found
2022-04-07 20.23.55 Closed Approved Issues:
2022-04-07 20.23.55 CVDL-551241: issue closed.
2022-04-07 20.23.55 CVDL-124151: Not Close. No matching status found
2022-04-07 20.23.56 Overdue issues email sent :
2022-04-07 20.23.57 CDD-771211: Error: Email cannot be sent. No assignee found
-----
```

Figure 7. Logfile sample.

By checking the log file, developers can catch up with the process of the bot, which task has been operating, or any error has been raised, what type of error is, where does it come from the server, or it may come from logic part or even conflicts between environments.

5. USE CASES

In terms of technology, use case is a term indicates the way of the application work in the real life via some specific functions. There are many tasks that the bot executed however I am allowed to show two of them which is considered informative and easy to follow.

5.1 Assigning Pre-check Tasks

- a. Story: The coordinator wants to assign all unassigned Pre-check issues to the right manager.
- b. Description:

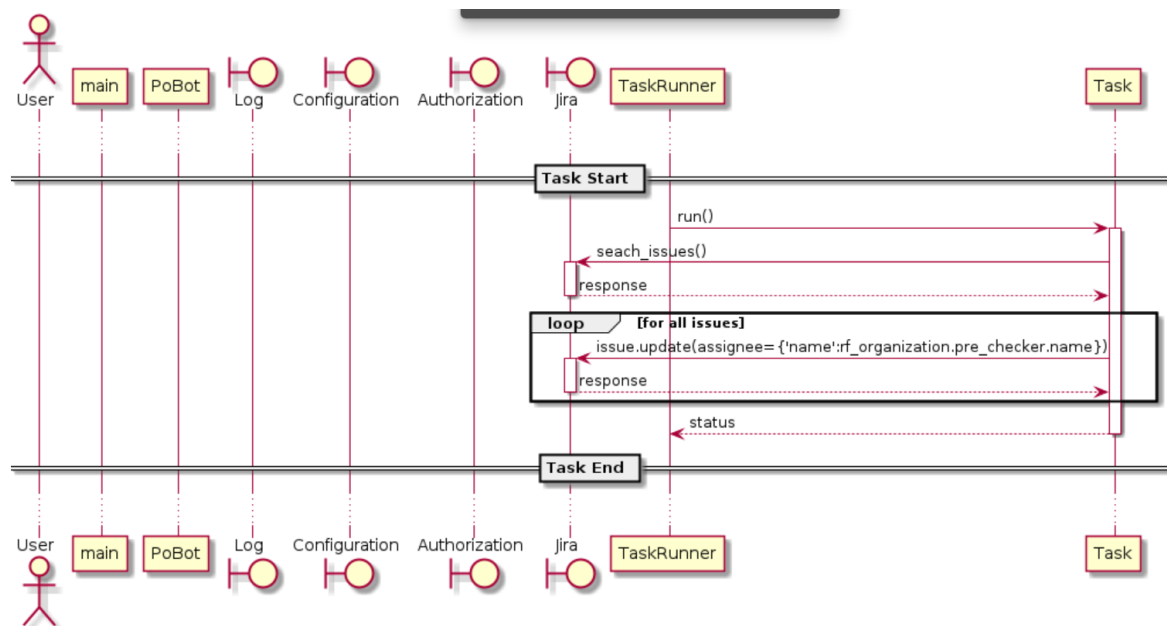


Figure 8. Sequence diagram displays the structure of assigning tasks in POBot.

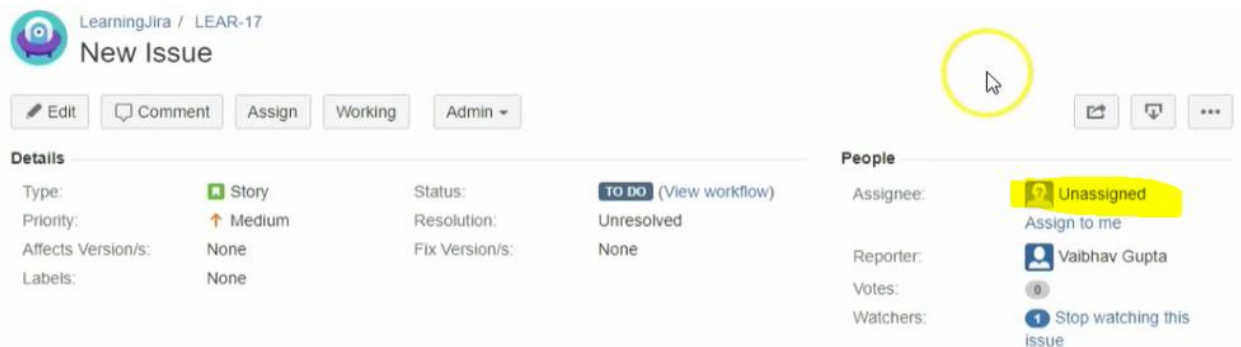
The Figure 8 above shows process after triggering the task in TaskRunner, POBot will get a JQL returning a group of unassigned Jira issues as a list. Then iterates every single issue in the list and use JIRA library to make the assignment by the command in Python. The result of the script will be saved into the log file.

The log file should contain the execution timestamp and a list of found items. Figure 9 below shows information about each item, they should contains both keys and summaries of the issue.

```
2022-04-07 20.23.54 Initialization Done:
2022-04-07 20.23.54 Unassigned pre-check assigned:
2022-04-07 20.23.55 VMO-443124: Improve code robustenss by writing good quality code
2022-04-07 20.23.55 VMO-551241: Close XYZ-1234 bug and pretend that is a new feature
|
```

Figure 9. The expected output of the POBot in the log file.

The UI result will be displayed in the JIRA server, this task first search for issues that have blank in the 'Assignee' field and fill it out based on the coordinator's requirement.



LearningJira / LEAR-17
New Issue

Edit Comment Assign Working Admin

Details

| | | | |
|--------------------|--------|----------------|-----------------------|
| Type: | Story | Status: | TO DO (View workflow) |
| Priority: | Medium | Resolution: | Unresolved |
| Affects Version/s: | None | Fix Version/s: | None |
| Labels: | None | | |

People

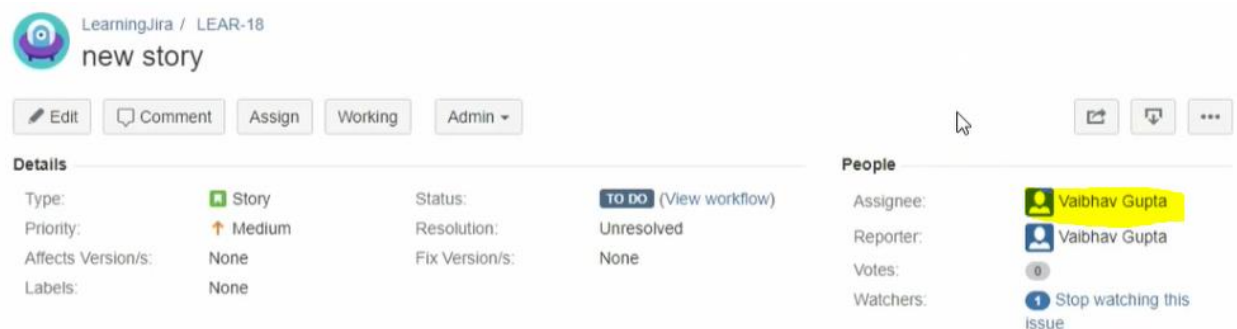
Assignee: Unassigned
Assign to me

Reporter: Vaibhav Gupta

Votes: 0

Watchers: Stop watching this issue

Figure 10. Unassigned Issue.



LearningJira / LEAR-18
new story

Edit Comment Assign Working Admin

Details

| | | | |
|--------------------|--------|----------------|-----------------------|
| Type: | Story | Status: | TO DO (View workflow) |
| Priority: | Medium | Resolution: | Unresolved |
| Affects Version/s: | None | Fix Version/s: | None |
| Labels: | None | | |

People

Assignee: Vaibhav Gupta

Reporter: Vaibhav Gupta

Votes: 0

Watchers: Stop watching this issue

Figure 11. Issue before and after updating the assignee field.

5.2 Overdue Issue Mail Inform

- a. Story: The coordinator wants to send to assignee notification mail for items has not been closed in the target timeline so the assignee will be aware that action is soon overdue.
- b. Description:

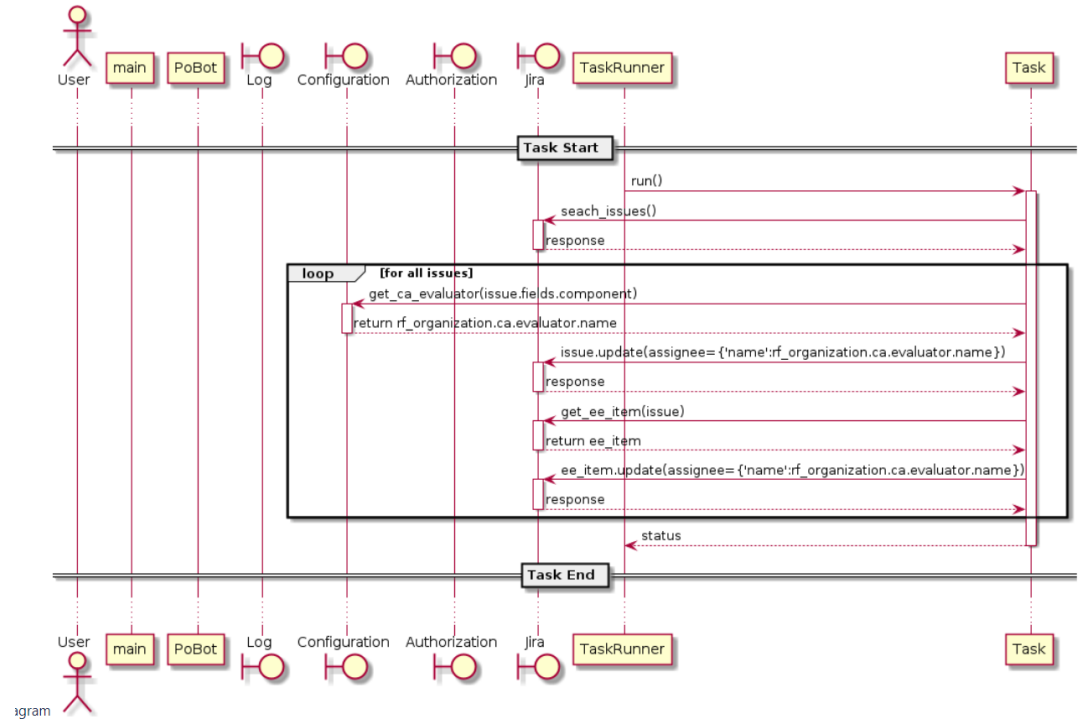


Figure 12. Sequence diagram displays structure of email sending task in POBot.

The Figure 12 above is a sequence diagram, it includes 2 basic components, first is the JQL to filter issues that the bot would like to proceed with. Iterating a single issue with the loop, with the support from JIRA, in each issue the bot is allowed to take the email address and related data that is needed for email content.

The second component is the run function which contains all logic parts of the task in the Jira server, in this case, they would be collecting data for email, preparing email content, Outlook email send function, and log the results.

Figure 13 below shows the sample of an email sent to the assignee via PObot. The static text such as subject or regards will remain the same in every email, however the information of each assignee for example name, issues or all values inside the table are filled into depending on their own data.

| Email | | | |
|---|------------------------------------|--------|------------|
| Subject: Overdue CNIs | | | |
| Dear Wick, John 1 (Nokia - PL/Wroclaw) | | | |
| Following CNI items assigned to you has not been finalized in target timeframe: | | | |
| Item | Summary | Status | Due date |
| CNI-1234 | Sharpen the pencil - RF evaluation | Open | 2021-09-17 |
| CNI-1354 | Sharpen the pencil - RF pre-check | Reopen | 2021-09-17 |
| Please finalize required activities without any additional delay. | | | |
| Best regards | | | |
| RF PO Team | | | |

Figure 13. Reminded email sample.

The log file should log information about each notification and its item.

```
2021-08-04 15:07:04:22 Overdue VMO notification send:
2021-08-04 15:07:04:23 Wick, John (Nokia - PL/Wroclaw): VMO-123, VMO-1333
2021-08-04 15:07:04:24 Wick, John 2 (Nokia - FI/Oulu): VMO-2213
```

Figure 14. Log file message in Overdue Mail task.

6. TESTING PROCESS

Testing is one of the final phases of software development, in which developers will create test cases for evaluating the whole project before the deployment. The process is basically a pre-check work by implementing a system to detect errors or bugs that may cause conflicts with the requirements or the real server.

6.1 Unit Test:

In this project, method testing was used as the unit test, which indicates the small level of testing (function, class, method, module). The unit test was conducted by exploiting libraries provided by different programming languages, technically it is also programmatically written by developers and automatically run. The unit test basically has two steps: it first takes the random given data, passes it to the test, and uses the 'assert' method from the library to make a comparison with the expected result.

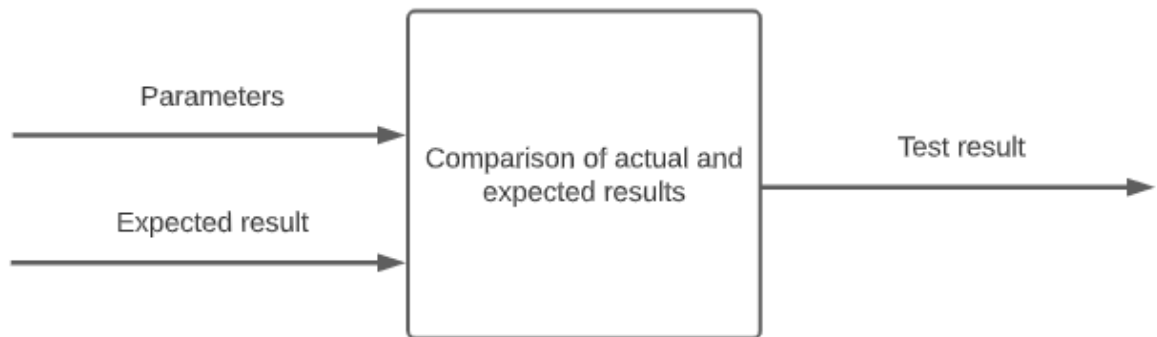


Figure 15. Unit Test structure.

A test code may be simple or complicated depending on the code we would like to make the assertion on. However, there are some standards to determine a well-written test.

- Fast and straight forward: in big projects, they may contain up to hundreds or thousands of tests. Therefore, unnecessary tested function may put a halt during CI/CD pipeline.
- Isolated: A unit test should be separated in order not to affect the other tests in case one of them fail.
- Deterministic: Always returns the same result no matter how many times, tests are run.
- Simple and Readable: Both the test and source code need to be maintained; therefore, a clear test could make this process a lot easier.

The unit test is considered an essential step in developing, a well-knowledge developer in unit testing is a big advantage. The usage of unit test in a software project will help in early bug detection, fixing them before the project grows will save a lot of effort and time. Moreover, the way writing a code also a reflect to the source code, in case any things happen with it, the unit test could become a great way to restore the original code.

6.2 Test Driven Development

Test Driven Development (TDD) is a software development practise in which the unit test will be implemented before logic code writing. TDD is created by three phases illustrated in Figure 16:

- Red Phase indicates the process of creating a precise test for functionality that is not implemented. Therefore, tests at this phase are usually failed test.
- Green Phase refers to correcting the code by writing basic code enough for the test to pass.
- Refactor phase means refactoring the Code. This is the final stage, here we focus on improving the code quality.

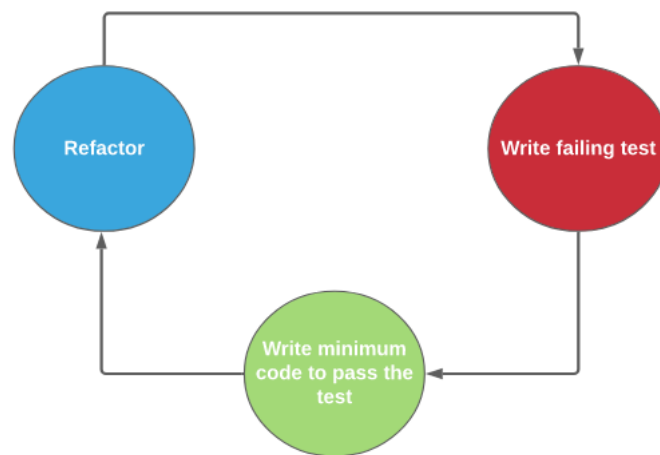


Figure 16. TDD cycle.

In general, TDD describes a process to make some modification in the code to pass the test that already implemented previously, therefore in terms of Software Engineering TDD also called 'Test First Development'. TDD is a significant example that proves the unit test has ability to clarify the source code, based on TDD, the developer could have a better understanding of the structure of the actual code.

6.2.1 Pytest and Unittest

Pytest are most popular library of Python using for unit test. In this project, I mostly using "pytest" for testing purpose however in some case "unittest" provide better function, it can also be used as substitution, all the method or installation of both are

remain the same. To work with those libraries, the working environment should have them installed.

```
C:\Users\MSI_\PycharmProjects\Demo_Testing>pip install pytest
```

Snippet 5. Command for installing testing library in Python.

For example, we have a file name 'main.py' including a very simple calculation function. The function will take 2 initial values and return the sum of 2 values.

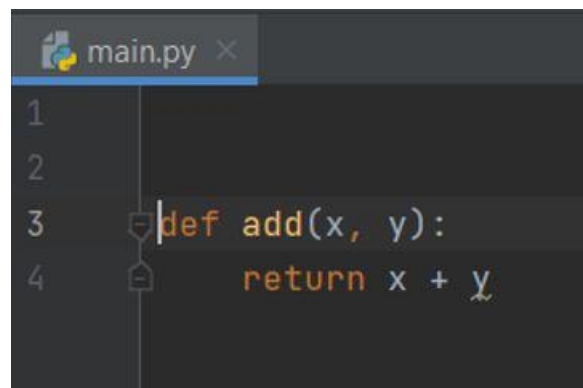
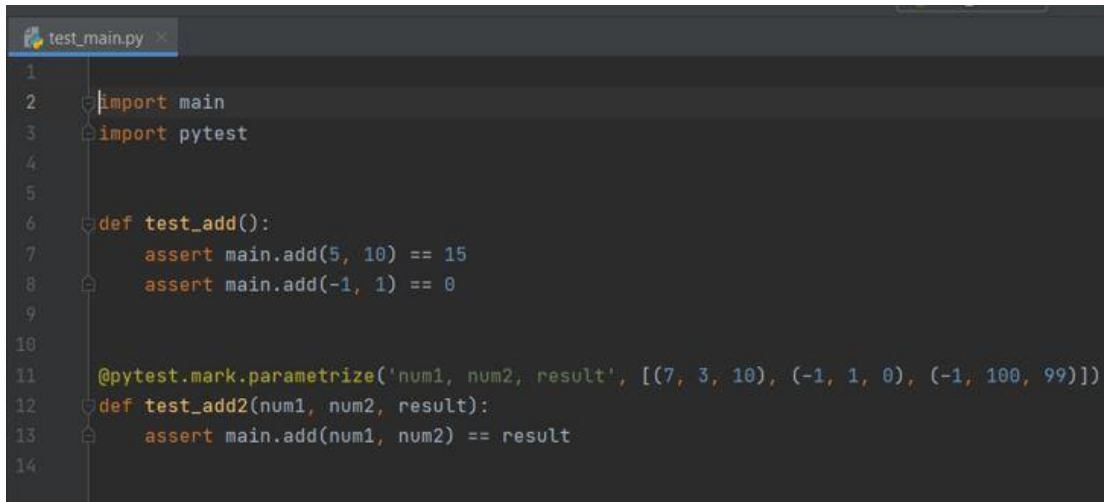
A screenshot of a code editor window titled 'main.py'. The editor shows a simple Python function definition. Line 1 is blank. Line 2 is blank. Line 3 contains the function signature 'def add(x, y):'. Line 4 contains the return statement 'return x + y'. The code is written in a light-colored font on a dark background. The cursor is positioned at the end of line 4.

Figure 17. Simple calculate function in Python.

In terms of testing, we would need to create a python file with the file name always start with either *test_filename.py* or *filename_test.py*, and the name of every function in a test file should also start with test_. For using "pytest" in code, we need to import it as the way we import other packages, we also need to import the module/file we would like to carry the test on. One function in real code can have as many test cases as the developers want as long as it covers all the function.



```
1
2 import main
3 import pytest
4
5
6 def test_add():
7     assert main.add(5, 10) == 15
8     assert main.add(-1, 1) == 0
9
10
11 @pytest.mark.parametrize('num1, num2, result', [(7, 3, 10), (-1, 1, 0), (-1, 100, 99)])
12 def test_add2(num1, num2, result):
13     assert main.add(num1, num2) == result
14
```

Figure 18. Test file of main.py.

In the test file, `test_add()` is a test case for function `add` in `main.py`. 'main' is called as an object which is imported at the top of the code. We can use this object to call exactly corresponding function to `main.py`. 'assert' is the statement from 'pytest' used to execute the real function and check if the given condition is match with expectation or not. The `add` function here is called with 2 variables are 5 and 10. We would like to check if the result 15 is corrected with these 2 values.

To run a single test, we can click right mouse anywhere inside that test function and choose 'Run'. By running a single test, we can refactor and complete it before moving to the next test case.

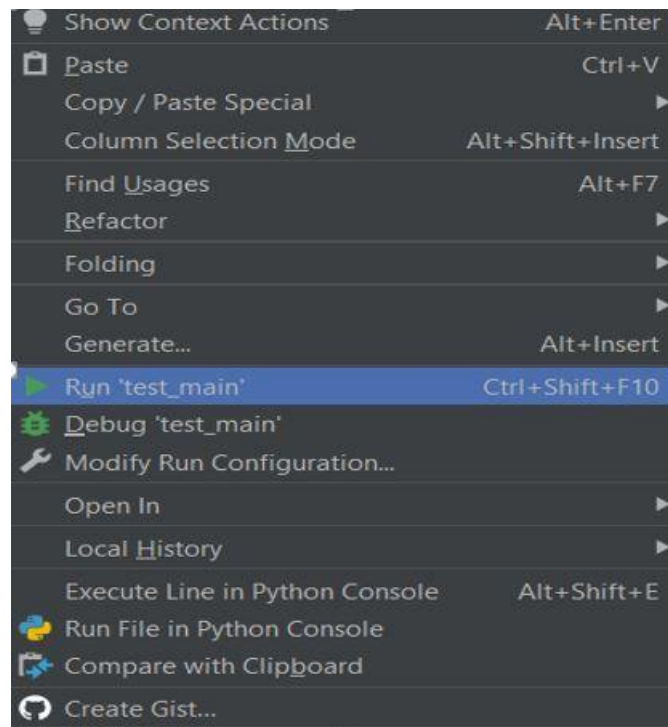


Figure 19. Run single test.

To run all the test during either working or CI/CD pipeline, testing Python library provides user command: **'pytest -s-v'**, the result will have the format below.

```
(venv) C:\Users\MSI_\PycharmProjects\Demo_Testing>pytest -s -v
===== test session starts =====
platform win32 -- Python 3.9.10, pytest-7.0.1, pluggy-1.0.0 -- c:\users\msi_\pycharmprojects\demo_testing\venv\scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\MSI_\PycharmProjects\Demo_Testing
collected 4 items

test_main.py::test_add PASSED
test_main.py::test_add2[7-3-10] PASSED
test_main.py::test_add2[-1-1-0] PASSED
test_main.py::test_add2[-1-100-99] PASSED
```

Figure 20. Result after running command.

As mentioned above in unit test section, the test usually contains 2 main steps, the initial output and assertion method to check result with expected output. Therefore, assert is the statement that have been used often throughout project testing phase;

in fact, the assert method supports a lot in checking and detecting errors in Log file by operating assert with expected string and actual output.

Although this project implanted both 'pytest' and 'unittest', each of them have their own advantage; however, from my experience 'pyest' is easier to work with, simple and efficient. For 'unittest', user need to import modules, define a function within a class scope before using, it makes the usage a bit more complicated than 'pytest' which just need to define only the testing function (Python Pool, 2021).

7. DEPLOYMENT

Deployment is the final stage of software development in which the whole source code will be put into the working environment in the server, the application will execute its function in the reality.

7.1 Virtual environment

POBot is deployed and hosted in internal Nokia Linux server, the account with fully accessed was provided by the coordinator. To operate the bot in server, not the local host we prepared them a virtual environment. This is simply an environment only worked for Python, it will include Python interpreter, all libraries, packages that the POBot used such as pytest, unites, Jira.

7.2 Crontab

In the Task section mentioned that the bot will have 2 types of tasks, one should be run hourly, and another is daily. To make it execute automatically in the server, we are now using cron job (cron schedule). This is a time-based job scheduler in Unix, by providing (crontab) the specific of time, the current task will be triggered automatically (TechTarget, 2005). Cron job is a huge support for emailing task, it helps people keep up date with the task soon will overdue.

8. CONCLUSIONS

To conclude, this thesis used the software development to customize an already existed Software (Jira Software) which is a Task Management Tools, in order to make it fit perfectly with the needs of internal working. The fact that every software is unique, they are built it for common use; therefore, the initial version is never the fully completed as it is not designed for any unique company. However, they will provide their REST API after the purchase, so that the customer will have to access to their API to personalize their organization needs. In general, PObot is considered as a bot which is a near replacement for the coordinator in managing the backlog inside a specific department. The fact that the bot is currently running and receives many of positive feedback. Because of automation work, the issues now can be updated quickly and punctually in a widely scope, the keep updated email have a clear format and fully informative. In future, more function will be updated so that the bot will have enough ability to control the workflow inside the department.

9. REFERENCES

TechTarget. 2005. Crontab. Accessed 2.5.2022.

<https://www.techtarget.com/searchdatacenter/definition/crontab>

Atlassian. 2020. What is Jira Used For. Accessed 22.4.2022.

<https://www.atlassian.com/software/jira/guides/use-cases/what-is-jira-used-for#Jira-for-requirements-&-test-case-management>

Dan, R. 2020. JQL: The most Flexible way to Search Jira. Accessed 25.05.2022.

<https://www.atlassian.com/blog/jira-software/jql-the-most-flexible-way-to-search-jira-14>

Python Pool. 2021. Python Unittest Vs Pytest Choose the Best. Accessed 1.5.2022.

<https://www.pythonpool.com/python-unittest-vs-pytest/>