AI BASED SOFTWARE DEVELOPMENT APPROACH OVER TRADITIONAL SOFTWARE DEVELOPMENT APPROACH IN THE DESIGN PHASE OF SOFTWARE DEVELOPMENT LIFE CYCLE

Presented by:

Jediael Meshua Sumer

Marlom Bey

Mriganka Shekhar Sarmah

OBJECTIVES

- Develop a class diagram generator that utilises artificial intelligence in the form of natural language processing to generate class diagrams
- Using the above as the basis, create a comparative study between using AI in software development and the traditional method of software development

PROBLEM STATEMENT

- To measure the effects of Artificial Intelligence in Software Development in comparison to Traditional Software Development.

MOTIVATION & SCOPE

- The motivation for taking this project comes from the growing rise in demand for artificial intelligence to make our lives easier.
- The scope of this research is to introduce artificial intelligence in software development so that we can increase the efficiency of software development with reduced cost and effort.

LITERATURE REVIEW

Artificial intelligence and software engineering: Status and future trends

The disciplines of Artificial Intelligence and Software Engineering have many commonalities. Both deal with modelling real world objects from the real world like business processes, expert knowledge, or process models. This article gives a short overview about these disciplines and describes some current research topics against the background of common points of contact.

Artificial intelligence and systems engineering. Prospects for Artificial Intelligence

This paper discusses the problems of applying artificial intelligence technology in the domain of systems engineering. The different process models used for systems engineering and AI are discussed and it is suggested that these differences are largely responsible for some of the mutual hostility which exists between the systems engineering and AI communities.

Software engineering using artificial intelligence techniques: Current state and open problems

This paper surveys the application of artificial intelligence approaches to the software engineering processes. These approaches can have a major impact on reducing the time to market and improving the quality of software systems in general. Existing survey papers are driven by the AI techniques used, or are focused on specific software engineering processes. This paper relates AI techniques to software engineering processes specified by the IEEE 12207 standard of software engineering. Some of the tools discussed are:

- SPECIFIER, a CASE based system that takes as input an informal specification of an operation
- ATGen, a software test data generator
- Class-Model Builder (CM Builder), a natural language based CASE tool that builds class diagrams specified in UML from natural language requirements documents

Ways of applying artificial intelligence in software engineering

As Artificial Intelligence (AI) techniques have become more powerful and easier to use they are increasingly deployed as key components of modern software systems. While this enables new functionality and often allows better adaptation to user needs it also creates additional problems for software engineers and exposes companies to new risks. Some work has been done to better understand the interaction between Software Engineering and AI but we lack methods to classify ways of applying AI in software systems and to analyse and understand the risks this poses. Only by doing so can we devise tools and solutions to help mitigate them.

Artificial intelligence and techniques in software engineering

This paper highlights a comparative study between the software development and expert system development. This paper also highlights absence of risk management strategies or risk management phase in AI based systems.

From user requirements to UML class diagram

The transition from user requirements to UML diagrams is a difficult task for the designer especially when he handles large texts expressing these needs. Modelling class Diagram must be performed frequently, even during the development of a simple application. This paper proposes an approach to facilitate class diagram extraction from textual requirements using NLP techniques and domain ontology.

FEASIBILITY STUDY & REQUIREMENT ANALYSIS

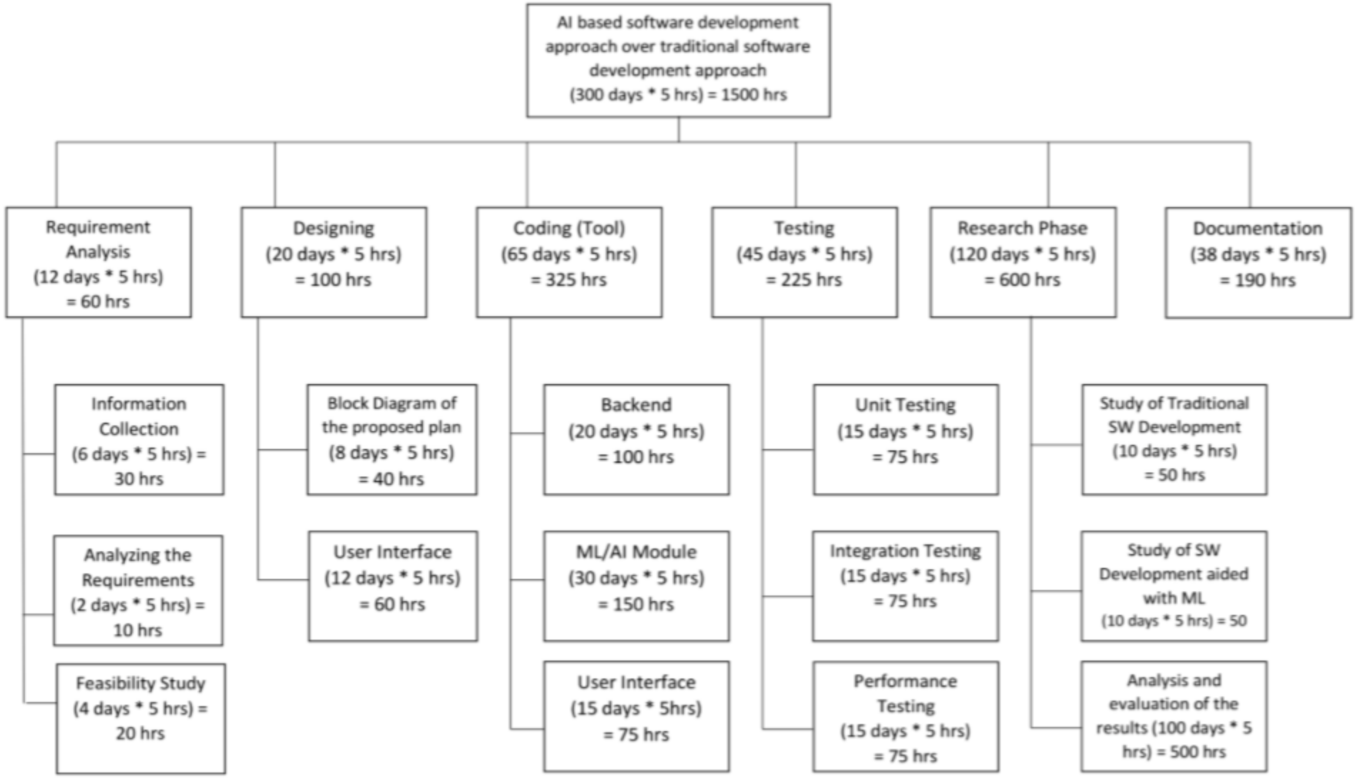
HARDWARE REQUIREMENTS

- Processor: Intel Core i5
- RAM: 3 GB or more
- Disk Space: [1.5] GB or more

SOFTWARE REQUIREMENTS

- Windows 7 or later or MacOS 10.10 or higher
- Programming languages: Python
- Anaconda Navigator 1.9.7

WORK BREAKDOWN STRUCTURE



AI based software development approach over traditional software development approach (300 days * 5 hrs) = 1500 hrs

Requirement Analysis (12 days * 5 hrs) = 60 hrs

Information Collection (6 days * 5 hrs) = 30 hrs

Analyzing the Requirements (2 days * 5 hrs) = 10 hrs

Feasibility Study (4 days * 5 hrs) = 20 hrs

Designing (20 days * 5 hrs) = 100 hrs

Block Diagram of the proposed plan (8 days * 5 hrs) = 40 hrs

User Interface (12 days * 5 hrs) = 60 hrs

Coding (Tool) (65 days * 5 hrs) = 325 hrs

Backend (20 days * 5 hrs) = 100 hrs

ML/AI Module (30 days * 5 hrs) = 150 hrs

User Interface (15 days * 5hrs) = 75 hrs

Testing (45 days * 5 hrs) = 225 hrs

Unit Testing (15 days * 5 hrs) = 75 hrs

Integration Testing (15 days * 5 hrs) = 75 hrs

Performance Testing (15 days * 5 hrs) = 75 hrs

Research Phase (120 days * 5 hrs) = 600 hrs

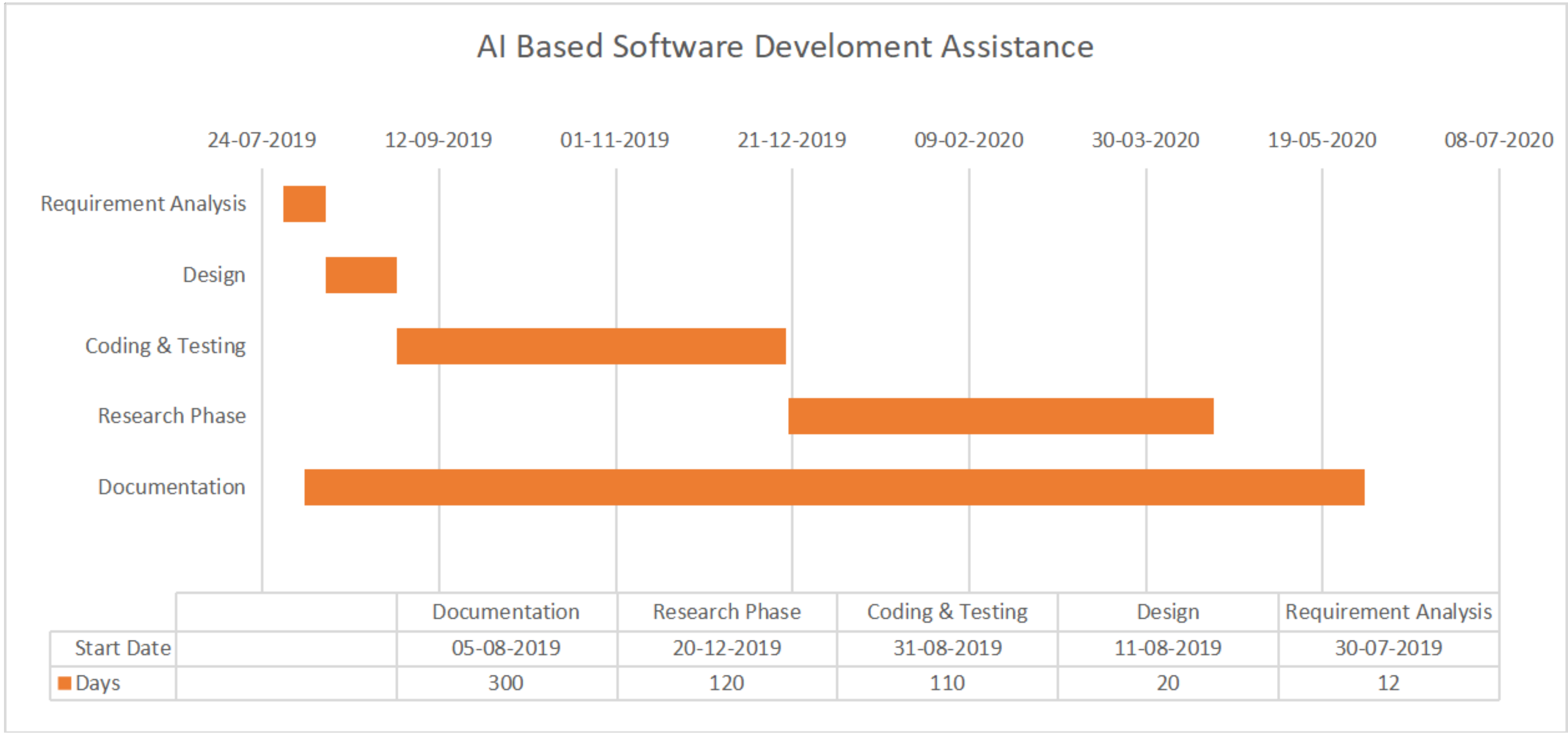
Study of Traditional SW Development (10 days * 5 hrs) = 50 hrs

Study of SW Development aided with ML (10 days * 5 hrs) = 50

Analysis and evaluation of the results (100 days * 5 hrs) = 500 hrs

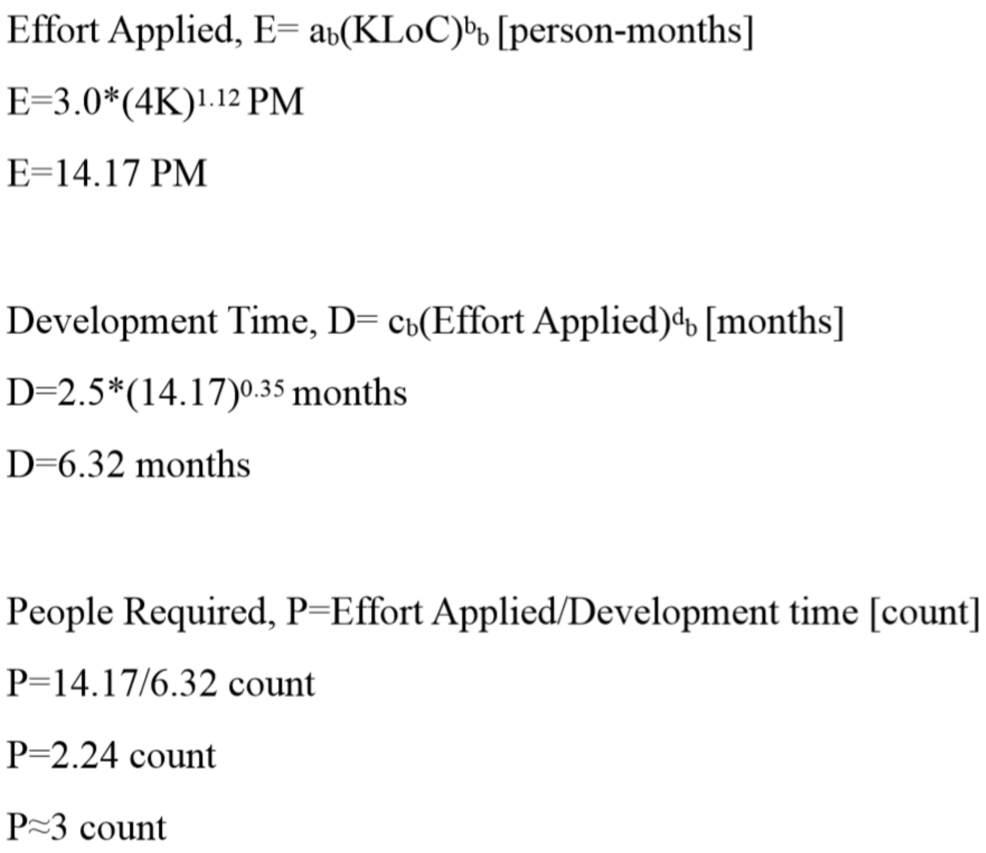
Documentation (38 days * 5 hrs) = 190 hrs

GANTT CHART



		Documentation	Research Phase	Coding & Testing	Design	Requirement Analysis
Start Date		05-08-2019	20-12-2019	31-08-2019	11-08-2019	30-07-2019
🟧 Days		300	120	110	20	12

COCOMO MODEL



Effort Applied, E= $a_b(\text{KLoC})^{b_b}$ [person-months]

E=$3.0*(4K)^{1.12}$ PM

E=14.17 PM

Development Time, D= $c_b(\text{Effort Applied})^{d_b}$ [months]

D=$2.5*(14.17)^{0.35}$ months

D=6.32 months

People Required, P=Effort Applied/Development time [count]

P=14.17/6.32 count

P=2.24 count

P≈3 count

PROPOSED PLAN

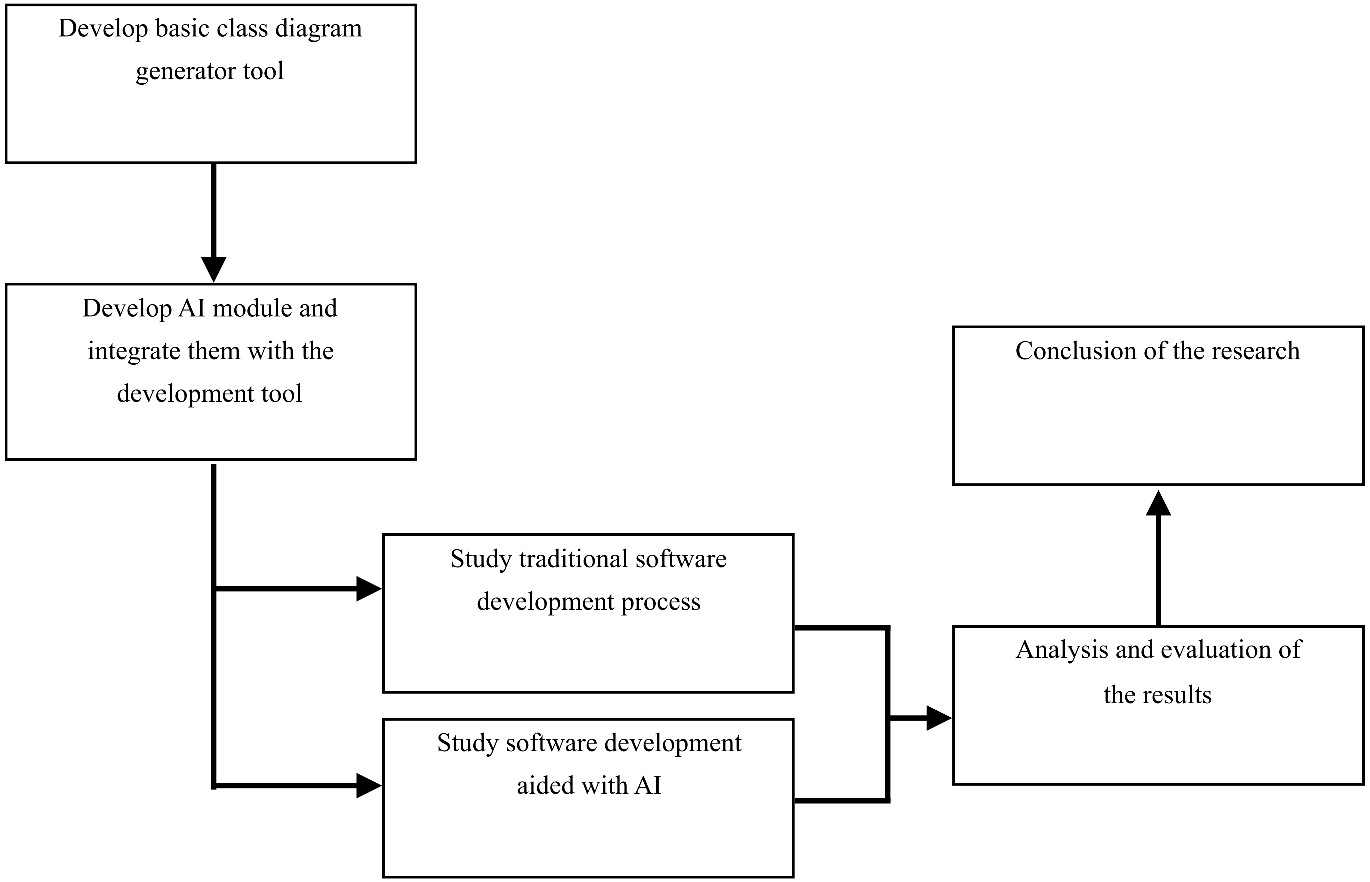
METHODOLOGY

1. Develop class diagram generator which utilises artificial intelligence, in the form of natural language processing
2. Define the traditional method of software development.
3. Define the sample space for the research.
4. Carry out the research and draw a comparison between the use of artificial intelligence, and the traditional method in software development.

PERFORMANCE EVALUATION METRICS

- Level of Automation
- Time
- Cost of development
- Accuracy of the generated diagrams

BLOCK DIAGRAM



Develop basic class diagram generator tool

Develop AI module and integrate them with the development tool

Study traditional software development process

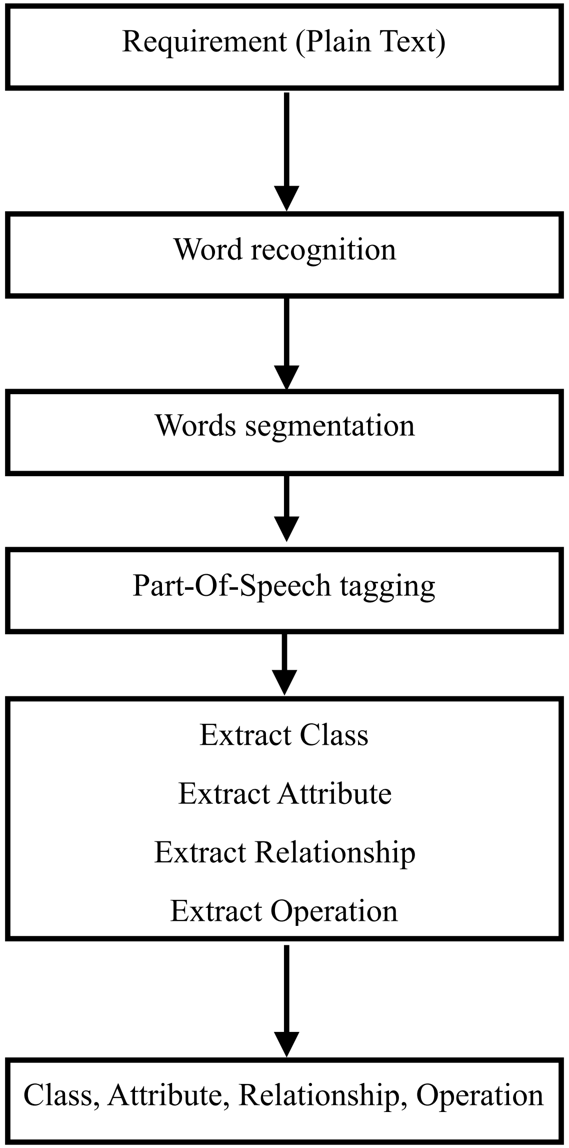
Study software development aided with AI

Analysis and evaluation of the results

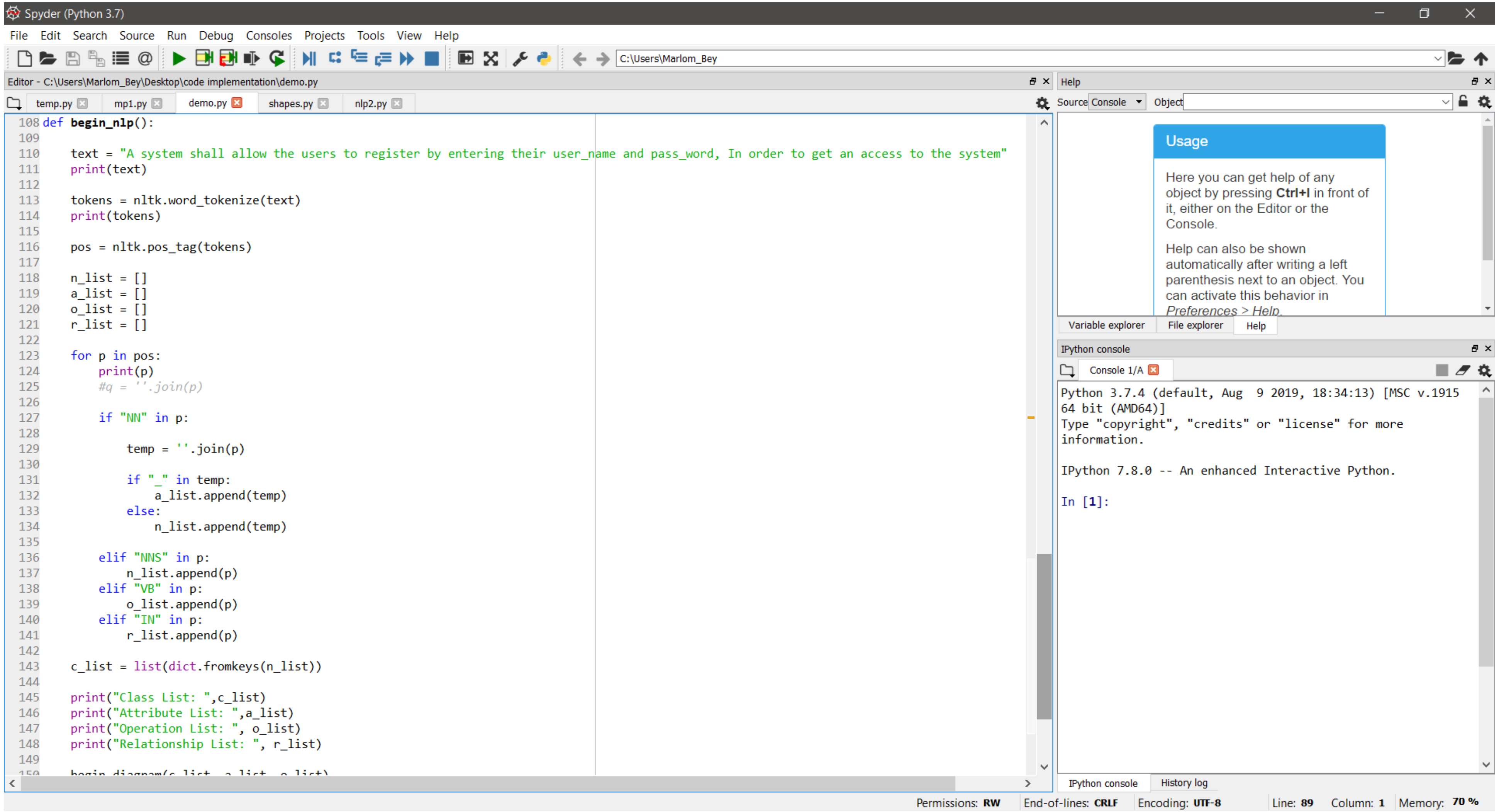
Conclusion of the research

IMPLEMENTATION

IMPLEMENTATION PROCEDURE



NATURAL LANGUAGE PROCESSING



Editor - C:\Users\Marlom_Bey\Desktop\code implementation\demo.py

```python
def begin_nlp():

    text = "A system shall allow the users to register by entering their user_name and pass_word, In order to get an access to the system"
    print(text)

    tokens = nltk.word_tokenize(text)
    print(tokens)

    pos = nltk.pos_tag(tokens)

    n_list = []
    a_list = []
    o_list = []
    r_list = []

    for p in pos:
        print(p)
        #q = ''.join(p)

        if "NN" in p:

            temp = ''.join(p)

            if "_" in temp:
                a_list.append(temp)
            else:
                n_list.append(temp)

        elif "NNS" in p:
            n_list.append(p)
        elif "VB" in p:
            o_list.append(p)
        elif "IN" in p:
            r_list.append(p)

    c_list = list(dict.fromkeys(n_list))

    print("Class List: ",c_list)
    print("Attribute List: ",a_list)
    print("Operation List: ", o_list)
    print("Relationship List: ", r_list)

    begin_diagram(c_list, a_list, o_list)
```

Usage

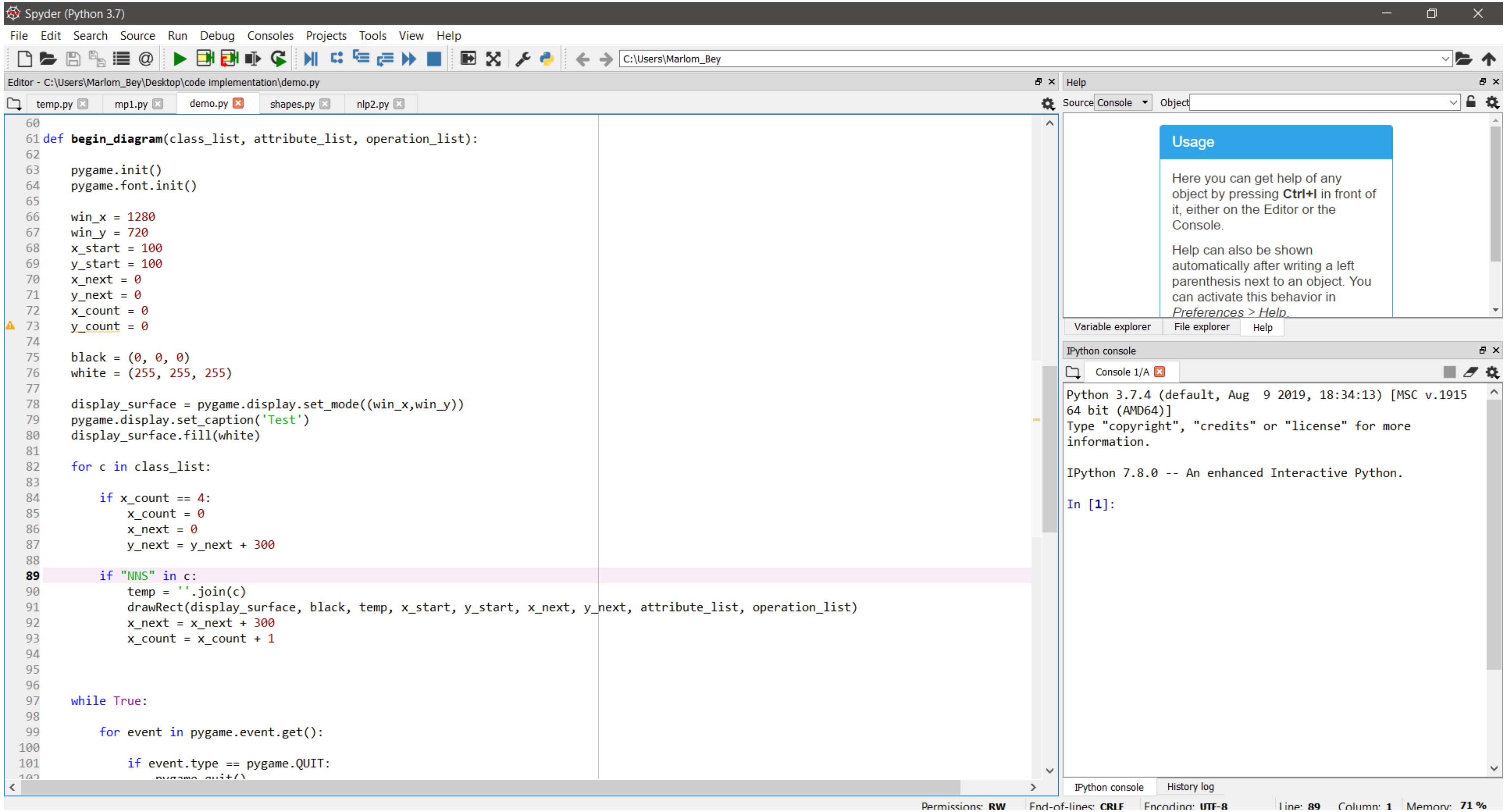
Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help.*

```
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915
64 bit (AMD64)]
Type "copyright", "credits" or "license" for more
information.

IPython 7.8.0 -- An enhanced Interactive Python.

In [1]:
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 89 Column: 1 Memory: 70 %

INITIALISING THE CLASS DIAGRAM GENERATING METHOD



```python
def begin_diagram(class_list, attribute_list, operation_list):

    pygame.init()
    pygame.font.init()

    win_x = 1280
    win_y = 720
    x_start = 100
    y_start = 100
    x_next = 0
    y_next = 0
    x_count = 0
    y_count = 0

    black = (0, 0, 0)
    white = (255, 255, 255)

    display_surface = pygame.display.set_mode((win_x,win_y))
    pygame.display.set_caption('Test')
    display_surface.fill(white)

    for c in class_list:

        if x_count == 4:
            x_count = 0
            x_next = 0
            y_next = y_next + 300

        if "NNS" in c:
            temp = ''.join(c)
            drawRect(display_surface, black, temp, x_start, y_start, x_next, y_next, attribute_list, operation_list)
            x_next = x_next + 300
            x_count = x_count + 1



    while True:

        for event in pygame.event.get():

            if event.type == pygame.QUIT:
                pygame.quit()
```

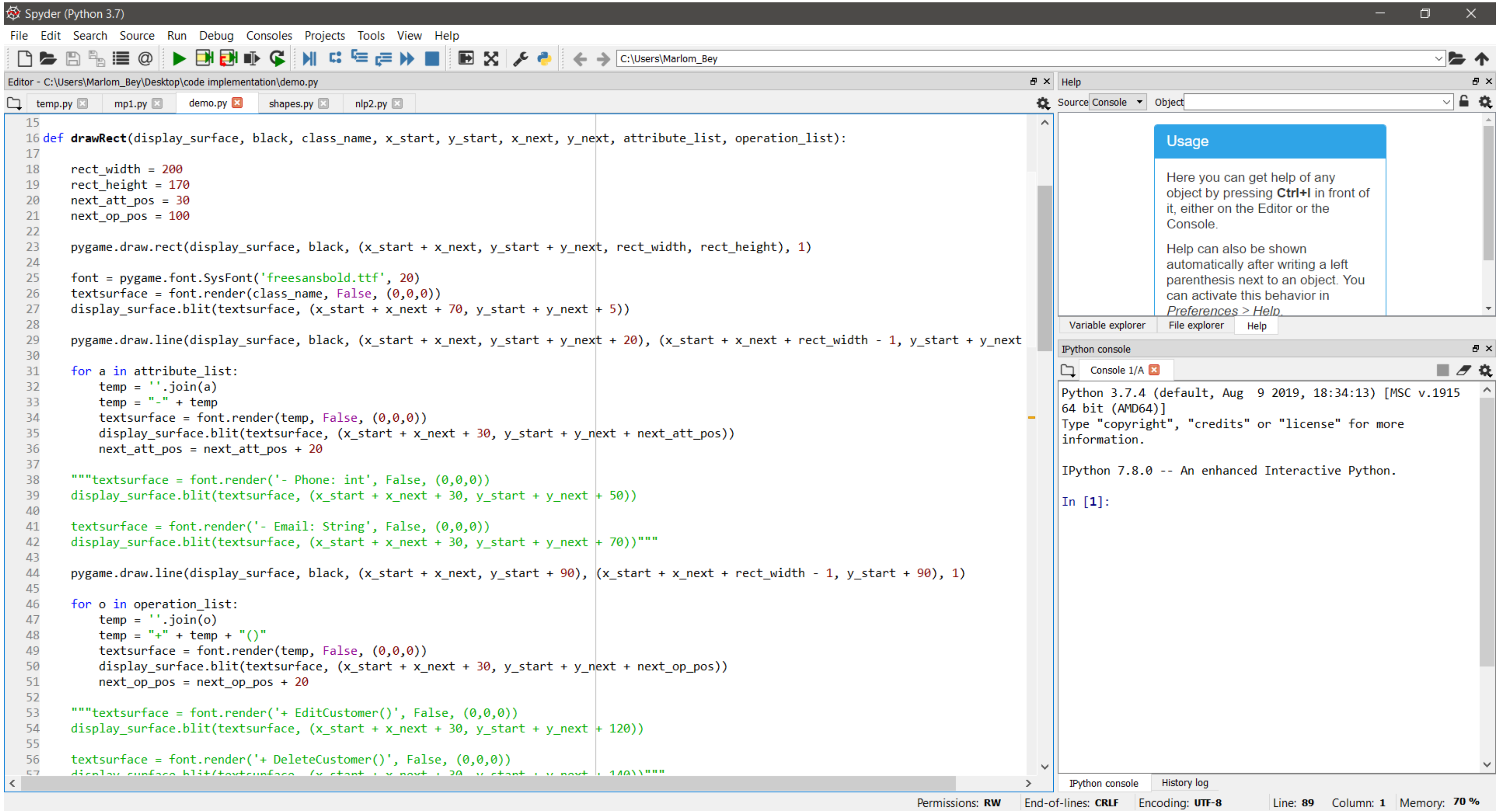
Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help.*

```
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915
64 bit (AMD64)]
Type "copyright", "credits" or "license" for more
information.

IPython 7.8.0 -- An enhanced Interactive Python.

In [1]:
```

Permissions: RW   End-of-lines: CRLF   Encoding: UTF-8   Line: 89   Column: 1   Memory: 71 %

GENERATING CLASS DIAGRAM USING PYGAME



```python

def drawRect(display_surface, black, class_name, x_start, y_start, x_next, y_next, attribute_list, operation_list):

    rect_width = 200
    rect_height = 170
    next_att_pos = 30
    next_op_pos = 100

    pygame.draw.rect(display_surface, black, (x_start + x_next, y_start + y_next, rect_width, rect_height), 1)

    font = pygame.font.SysFont('freesansbold.ttf', 20)
    textsurface = font.render(class_name, False, (0,0,0))
    display_surface.blit(textsurface, (x_start + x_next + 70, y_start + y_next + 5))

    pygame.draw.line(display_surface, black, (x_start + x_next, y_start + y_next + 20), (x_start + x_next + rect_width - 1, y_start + y_next

    for a in attribute_list:
        temp = ''.join(a)
        temp = "-" + temp
        textsurface = font.render(temp, False, (0,0,0))
        display_surface.blit(textsurface, (x_start + x_next + 30, y_start + y_next + next_att_pos))
        next_att_pos = next_att_pos + 20

    """textsurface = font.render('- Phone: int', False, (0,0,0))
    display_surface.blit(textsurface, (x_start + x_next + 30, y_start + y_next + 50))

    textsurface = font.render('- Email: String', False, (0,0,0))
    display_surface.blit(textsurface, (x_start + x_next + 30, y_start + y_next + 70))"""

    pygame.draw.line(display_surface, black, (x_start + x_next, y_start + 90), (x_start + x_next + rect_width - 1, y_start + 90), 1)

    for o in operation_list:
        temp = ''.join(o)
        temp = "+" + temp + "()"
        textsurface = font.render(temp, False, (0,0,0))
        display_surface.blit(textsurface, (x_start + x_next + 30, y_start + y_next + next_op_pos))
        next_op_pos = next_op_pos + 20

    """textsurface = font.render('+ EditCustomer()', False, (0,0,0))
    display_surface.blit(textsurface, (x_start + x_next + 30, y_start + y_next + 120))

    textsurface = font.render('+ DeleteCustomer()', False, (0,0,0))
    display_surface.blit(textsurface, (x_start + x_next + 30, y_start + y_next + 140))"""
```

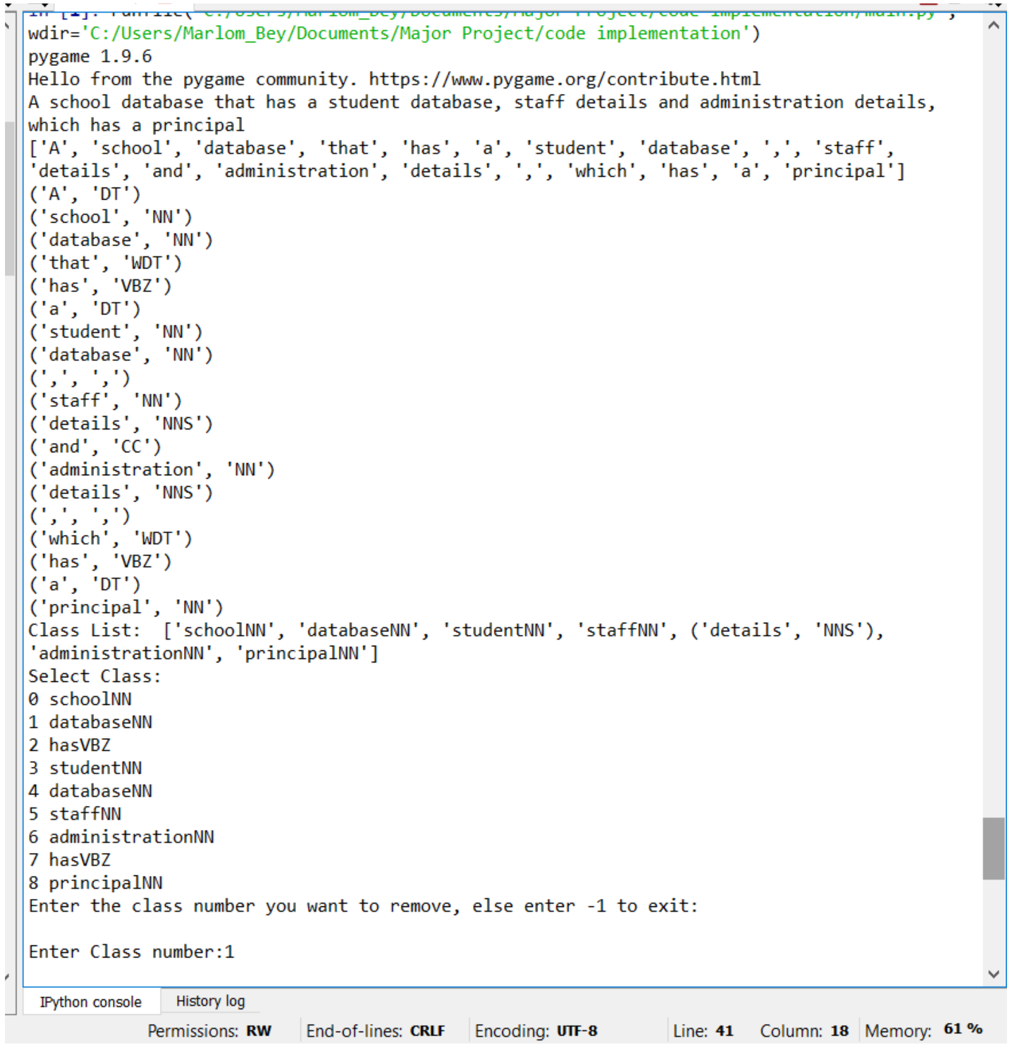
Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help.*

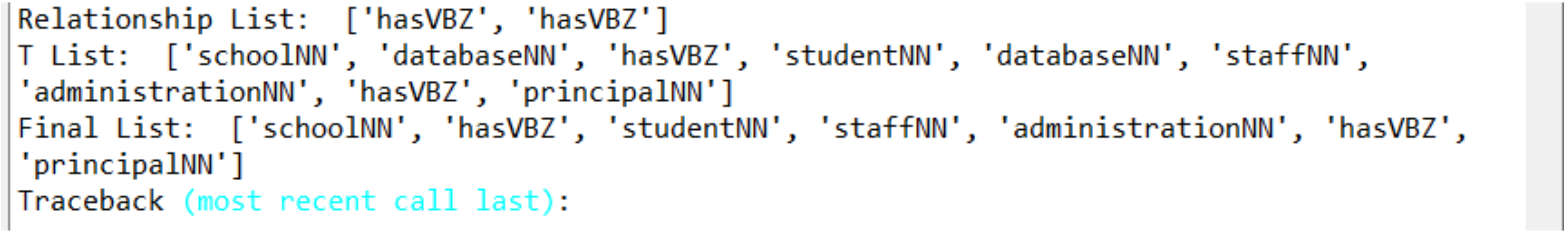
```
Python 3.7.4 (default, Aug  9 2019, 18:34:13) [MSC v.1915
64 bit (AMD64)]
Type "copyright", "credits" or "license" for more
information.

IPython 7.8.0 -- An enhanced Interactive Python.

In [1]:
```

SELECTING THE CLASSES FROM THE CANDIDATE CLASSES



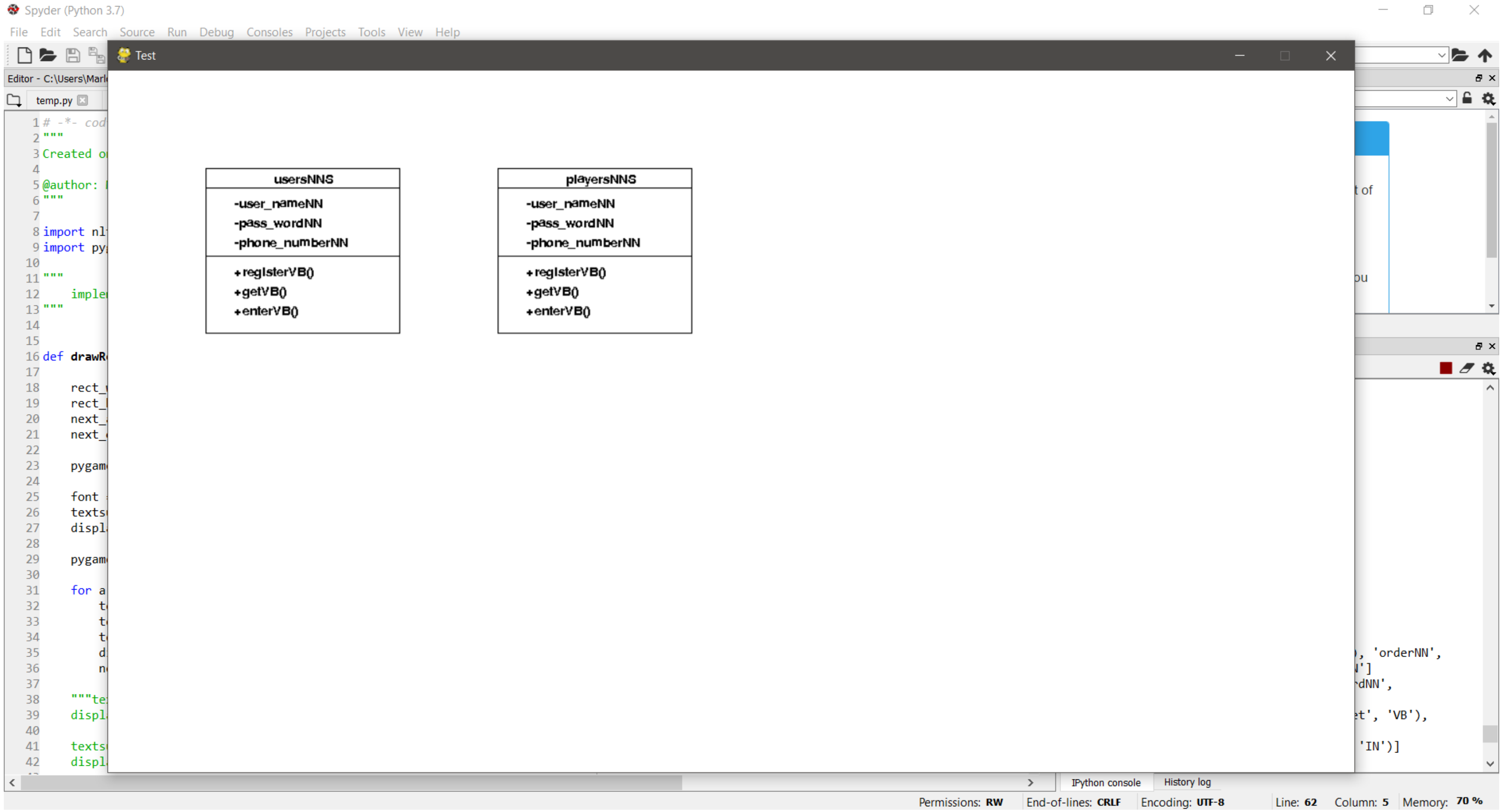
```
In [1]: runfile('C:/Users/Marlom_Bey/Documents/Major Project/code implementation/main.py',
wdir='C:/Users/Marlom_Bey/Documents/Major Project/code implementation')
pygame 1.9.6
Hello from the pygame community. https://www.pygame.org/contribute.html
A school database that has a student database, staff details and administration details,
which has a principal
['A', 'school', 'database', 'that', 'has', 'a', 'student', 'database', ',', 'staff',
'details', 'and', 'administration', 'details', ',', 'which', 'has', 'a', 'principal']
('A', 'DT')
('school', 'NN')
('database', 'NN')
('that', 'WDT')
('has', 'VBZ')
('a', 'DT')
('student', 'NN')
('database', 'NN')
(',', ',')
('staff', 'NN')
('details', 'NNS')
('and', 'CC')
('administration', 'NN')
('details', 'NNS')
(',', ',')
('which', 'WDT')
('has', 'VBZ')
('a', 'DT')
('principal', 'NN')
Class List:  ['schoolNN', 'databaseNN', 'studentNN', 'staffNN', ('details', 'NNS'),
'administrationNN', 'principalNN']
Select Class:
0 schoolNN
1 databaseNN
2 hasVBZ
3 studentNN
4 databaseNN
5 staffNN
6 administrationNN
7 hasVBZ
8 principalNN
Enter the class number you want to remove, else enter -1 to exit:

Enter Class number:1
```

GENERATING THE FINAL LIST



```
Relationship List:  ['hasVBZ', 'hasVBZ']
T List:  ['schoolNN', 'databaseNN', 'hasVBZ', 'studentNN', 'databaseNN', 'staffNN',
'administrationNN', 'hasVBZ', 'principalNN']
Final List:  ['schoolNN', 'hasVBZ', 'studentNN', 'staffNN', 'administrationNN', 'hasVBZ',
'principalNN']
Traceback (most recent call last):
```

GENERATED CLASS DIAGRAM



```
# -*- cod
"""
Created o

@author: 
"""

import nl
import py

"""
    imple
"""


def drawR

    rect_
    rect_
    next_
    next_

    pygam

    font 
    texts
    displ

    pygam

    for a
        t
        t
        t
        d
        n

    """te
    displ

    texts
    displ
```

```
t of
ou
), 'orderNN',
N']
rdNN',
et', 'VB'),
'IN')]
```

usersNNS
- -user_nameNN
- -pass_wordNN
- -phone_numberNN
- +registerVB()
- +getVB()
- +enterVB()

playersNNS
- -user_nameNN
- -pass_wordNN
- -phone_numberNN
- +registerVB()
- +getVB()
- +enterVB()

CONCLUSION

Thus with this project, an exact line can be drawn to distinguish the effects that artificial intelligence have on software development.

References: Research Papers

- Rech, J., & Althoff, K. D. (2004). Artificial intelligence and software engineering: Status and future trends. KI, 18(3), 5-11.
- Sommerville, I. (1993). Artificial intelligence and systems engineering. Prospects for Artificial Intelligence: Proceedings of AISB'93, 29 March-2 April 1993, Birmingham, UK, 17, 48.Computing Department, Lancaster University, LANCASTER LA1 4YR, UK.
- Ammar, H. H., Abdelmoez, W., & Hamdi, M. S. (2012, February). Software engineering using artificial intelligence techniques: Current state and open problems. In Proceedings of the First Taibah University International Conference on Computing and Information Technology (ICCIT 2012), Al-Madinah Al-Munawwarah, Saudi Arabia (p. 52).
- Feldt, R., de Oliveira Neto, F. G., & Torkar, R. (2018, May). Ways of applying artificial intelligence in software engineering. In Proceedings of the 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (pp. 35-41). ACM.
- Raza, F. N. (2009, March). Artificial intelligence techniques in software engineering (AITSE). In International MultiConference of Engineers and Computer Scientists (IMECS 2009) (Vol. 1).
- Herchi, H., & Abdessalem, W. B. (2012). From user requirements to UML class diagram. *arXiv preprint arXiv:1211.0713*.

References: Links

- Web Links:
  - COCOMO Model: https://www.geeksforgeeks.org/softwareengineering-cocomo-model (02/08/2019)
  - Anaconda Navigator: https://docs.anaconda.com/anaconda/navigator (18/10/2019)
- Project Idea:
  - https://www.smartdatacollective.com/traditional-vs-machine-learning-for-software-developmentparadigms/ (25/07/2019) https://www.analyticsindiamag.com/how-machine-learning-is-changing-the-software-developmentparadigm/ (25/07/2019)

THANK YOU