

## **ABSTRACT:**

Safeguarding personally identifiable information (PII) is crucial because such information is increasingly used to engineer privacy attacks, identity thefts and security breaches. But is it likely that individuals may choose to just share this information with strangers? We find that individuals not only reciprocate and share PII when the disclosure of such information is private and directed towards them by a stranger, but also when the stranger shares this information through a public channel that is not directed towards anyone in particular. The project focus to Identify personally identifiable information on such public or open web sites and notifying user about the availability.

Information about an individual including but not limited to, Education, Employment, Financial Transactions, Medical History, and Criminal Background information which can be used to distinguish or trace an individual's identity, such as their name, social security number, date and place of birth, mother's maiden name, biometric records etc. , including any other personal information that can be linked to an individual.

<b>CONTENTS</b>	<b>PAGE NO</b>
<b>1. INTRODUCTION</b>	<b>1</b>
1.1 Introduction to PII	2
1.2 History of Data Breach	2
1.3 Python	3
1.4 MongoDB	3
1.5 Examples of sensitive PII	4
<b>2.MOTIVATION FOR THE PROJECT</b>	<b>5</b>
2.1 Uber Incident	7
2.2 PizzaHut Incident	7
<b>3. LITERATURE SURVEY</b>	<b>8</b>
3.1Python	9
3.1MongoDB	18
<b>4. SYSTEM ANALYSIS</b>	<b>23</b>
4.1 Existing System	24
4.2 Proposed System	24
4.3 Feasibility Study	25
4.3.1 Economical Feasibility	25
4.3.2 Technical Feasibility	25
4.3.3 Social Feasibility	26
<b>5. SYSTEM REQUIREMENTS SPECIFICATIONS</b>	<b>27</b>
5.1 Introduction	28
5.2 Non-Functional Requirements	28
5.3 System Requirements	29
<b>6. SYSTEM DESIGN</b>	<b>30</b>
6.1 Introduction	31
6.2 High-level design	31
6.3 Low-level design	31
6.3.1 UML Diagrams	31

<b>7. WORKING DESCRIPTION</b>	39
<b>8. CODING</b>	42
<b>9. TESTING</b>	47
9.1 Types Of Testing	48
9.2 Test Strategy and Approach	49
9.3 Test Cases	50
<b>10. RESULTS &amp;SCREENSHOTS</b>	51
<b>11. CONCLUSION</b>	60
<b>12. BIBILIOGRAPHY</b>	62

# **INTRODUCTION**

# **1. INTRODUCTION**

## **1.1 Introduction to PII:**

Personally identifiable information (PII) is any data that could potentially identify a specific individual. Any information that can be used to distinguish one person from another and can be used for de-anonymizing anonymous data can be considered PII. PII can be sensitive or non-sensitive. Non-sensitive PII is information that can be transmitted in an unencrypted form without resulting in harm to the individual. Non-sensitive PII can be easily gathered from public records, phone books, corporate directories and websites. Sensitive PII is information which, when disclosed, could result in harm to the individual whose privacy has been breached. Sensitive PII should therefore be encrypted in transit and when data is at rest. Such information includes biometric information, medical information, personally identifiable financial information (PIFI) and unique identifiers such as passport or Social Security numbers. We often talk about PII in the context of data breaches and identity theft. If a company or organization suffers a data breach, a significant concern is what PII might be exposed—the personal data of the customers that do business or otherwise interact with the entity. Exposed PII can be sold on the dark web and used to commit identity theft, putting breach victims at risk.

## **1.2 History of Data Breach**

Don't think this is some minor issue. Identity theft is a big consumer concern. A 2017 FICO survey revealed some surprising attitudes toward identity theft compared with other concerns "44 percent of American consumers say identity theft and banking fraud, combined, is their top concern".

Identity theft is the deliberate use of someone else's identity, usually as a method to gain a financial advantage or obtain credit and other benefits in the other person's name, and perhaps to the other person's disadvantage or loss. The person whose identity has been assumed may suffer adverse consequences, especially if they are held responsible for the perpetrator's actions. Identity theft occurs when someone uses another's personally identifying information, like their name, identifying number, or credit card number, without their permission, to commit fraud or other crimes. The term identity theft was coined in 1964. Since that time, the definition of identity theft has been statutorily prescribed throughout both the U.K. and the United States as the theft of personally identifying information, generally including a person's name, date of birth, social security number, driver's license number, bank account or credit card numbers, PIN numbers, electronic signatures, fingerprints, passwords, or any other information that can be used to access a person's financial resources. The acquisition of personal identifiers is made possible through serious breaches of privacy. For consumers, this is usually a result of them naively providing their personal information or login credentials to the identity thieves as

a result of being duped but identity-related documents such as credit cards, bank statements, utility bills, check books etc. may also be physically stolen from vehicles, homes, offices, and not the least letter boxes, or directly from victims by pickpockets and bag snatchers. Guardianship of personal identifiers by consumers is the most common intervention strategy recommended by the US Federal Trade Commission, Canadian Phone Busters and most sites that address identity theft. Such organizations offer recommendations on how individuals can prevent their information falling into the wrong hands. With more consumers concerned about identity theft than their mortality, it makes sense to remember the value of PII.

## **1.3 PYTHON**

Python is an Interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

## **1.4 MongoDB**

MongoDB is a cross-platform and open-source document-oriented database, a kind of NoSQL database. As a NoSQL database, MongoDB shuns the relational database's table-based structure to adapt JSON-like documents that have dynamic schemas which it calls BSON.

This makes data integration for certain types of applications faster and easier. MongoDB is built for scalability, high availability and performance from a single server deployment to large and complex multi-site infrastructures. MongoDB was first developed by MongoDB Inc., known then as 10gen, in October 2007 originally as a major part in a PaaS (Platform as a Service) product similar to Windows Azure and Google App Engine. The development was shifted to open source in 2009.

## **1.5 Examples of sensitive PII elements include, but are not limited to:**

- Name and other names used;
- Social Security number, full and truncated;
- Driver's license and other government identification numbers;
- Citizenship, legal status, gender, race/ethnicity;
- Birth date, place of birth;
- Home and personal cell telephone numbers;
- Personal email address, mailing and home address;
- Religious preference;
- Security clearance;
- Mother's middle and maiden names;
- Biometrics;
- Financial information, medical information, disability information;
- Law enforcement information, employment information, educational information;
- Military records.

## **Examples of non-sensitive PII elements include, but are not limited to:**

- Office location;
- Business telephone number;
- Business email address;
- Badge number; and
- Other information that is releasable to the public.

Information about an individual that identifies, links, relates, or is unique to, or describes him or her, e.g., a social security number; age; military rank; civilian grade; marital status; race; salary; home phone numbers; other demographic, biometric, personnel, medical, and financial information, etc. Such information is also known as personally identifiable information (i.e., information which can be used to distinguish or trace an individual's identity, such as their name, social security number, date and place of birth, mother's maiden name, biometric records, including any other personal information which is linked or linkable to a specified individual).

## **MOTIVATION FOR THE PROJECT**



## 2. MOTIVATION FOR THE PROJECT

Not so long ago, the most common way people protected their personally identifiable information (PII) was to pay for an unlisted telephone number. Today, there are many types of PII—and it's not just businesses that use and must protect PII. Schools, universities, healthcare facilities, retailers, government offices and many other organizations also acquire, process and store highly sensitive records. Use of technology has resulted in much greater flexibility and speed when it comes to making purchases, processing payments and managing data records. However, it has also led to a growing data loss prevention (DLP) problem that puts people's PII at risk. There are two types of data loss: accidental and malicious. Human error or carelessness as well as a lack of data security processes in an organization can lead to accidental loss, including something as simple as sending an email attachment containing PII to the wrong recipient. Malicious data breaches, on the other hand, are deliberate internal or external attacks on an organization's data systems. Consequences of not protecting PII Regardless of how the data is lost, the cost of a data breach can be huge. Fines are one of the most widely-known consequences of losing personal data, and they can be very expensive (e.g., up to \$1.5 million per year in the case of a breach of healthcare records in violation of the Health Insurance Portability and Accountability Act [HIPAA] regulation or up to £500,000 from the UK Information Commissioner). However, the consequences extend much further and include reputation damage, loss of customer trust, employee dissatisfaction and attrition, and clean-up costs following the breach

Despite of all great security products available today, the number of data breaches continues to rise. More than a half-billion personal records of Indian residents have been exposed since 2005, and in 2018 alone 10 million Indian were victims of identity fraud, an increase from 2007 by over 20%, its time for a fresh approach.

Reminders:

- Be familiar with security, privacy and confidentiality practices.
- Use beneficiary personal data only for purposes for which you have authorization.
- Lock or logoff computer workstation/terminal prior to leaving it unattended. Act in an ethical, informed and trustworthy manner.
- Protect sensitive electronic records.
- Be alert to threats and vulnerabilities to your systems.
- Ensure that employee screening for sensitive positions within your organization has occurred prior to any individual being authorized access to sensitive or critical applications.
- Avoid leaving paper documents containing personal data lying unprotected on desktops.

## **2.1 Uber Incident**

Uber concealed a hack that affected 57 million customers and drivers worldwide and 2.7 million users in the UK, the company has confirmed. The breach - which took place in 2016 - was kept under wraps by the taxi-hailing firm, which paid hackers \$100,000 (£75,000) to delete the data. Uber confirmed that names, email addresses and mobile phone numbers of customers were exposed and of the 57 million impacted, 600,000 drivers had their names and licence details compromised.

And while the drivers have been offered free credit monitoring protection, the firm is yet to offer anything to affected customers. According to Bloomberg, Uber's former chief executive Travis Kalanick knew about the breach over a year ago. The firm's chief security officer Joe Sullivan has left the company. In a written statement, Uber CEO Dara Khosrowshahi said: "While we have not seen evidence of fraud or misuse tied to the incident, we are monitoring the affected accounts and have flagged them for additional fraud protection.

"None of this should have happened, and I will not make excuses for it." Uber confirmed to the Information Commissioner that 2.7 million - over half - of Uber's UK users had been affected. The National Cyber Security Centre suggested "vigilance" against email phishing or scam phone calls in light of the hack.

## **2.2 Pizza Hut Incident**

Pizza Hut has revealed that its website and app were hacked on 1 October, with personal information for an undisclosed amount of customers being jeopardised. The hack is thought to have compromised billing information including delivery addresses, email addresses and payment card information containing account numbers, expiration dates and CVV numbers. Pizza Hut has sent out emails to customers informing them of the breach, which reveal Pizza Hut knew of the breach two weeks before disclosing it.

In the email, the company said: "Pizza Hut has recently identified a temporary security intrusion that occurred on our website. We have learned that the information of some customers who visited our website or mobile application during an approximately 28-hour period (from the morning of October 1, 2017, through midday on October 2, 2017) and subsequently placed an order may have been compromised. "Pizza hut identified the security intrusion quickly and took immediate action to halt it." It's unclear how many customers have been affected by the hack, but a figure of 60,000 US customers has been reported by Slashdot.

## **LITERATURE SURVEY**

## 3. LITERATURE SURVEY

### 3.1 Python

We heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to “big name” languages. Hopefully I can explain it for you.

#### Python concepts

If you are not interested in the the hows and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it's a great one to start programming with.

- Open source general-purpose language.
- Object Oriented, Procedural, Functional
- Easy to interface with C/ObjC/Java/Fortran
- Easy-ish to interface with C++ (via SWIG)
- Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

**Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

**Python is Interactive** – We can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

**Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner's Language – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

#### History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public

License (GPL).Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

## Python Features

Python's features include –

**Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

**Easy-to-read** – Python code is more clearly defined and visible to the eyes.

**Easy-to-maintain** – Python's source code is fairly easy-to-maintain.A broad standard library – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

**Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

**Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

**Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

**Databases** – Python provides interfaces to all major commercial databases.

**GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

**Scalable** – Python provides a better structure and support for large programs than shell scripting.Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

It supports functional and structured programming methods as well as OOP. It can be used as a scripting language or can be compiled to byte-code for building large applications. It provides very high-level dynamic data types and supports dynamic type checking. IT supports automatic garbage collection. It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

## Dynamic vs Static

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of “thing” each data value is. For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a “float” type. This tells the compiler that the only data that can be used for that variable must be a floating point number, i.e. a number with a decimal point. If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn't require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using. This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating point number) you need in your program. With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment). If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double.

With Python, it doesn't matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values. For example, say you are dividing two numbers. One is a floating point number and one is an integer. Python realizes that it's more accurate to keep track of decimals so it automatically calculates the result as a floating point number

## **Variables**

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory. Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

## **Standard Data Types**

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types –

Numbers

String

List

Tuple

Dictionary

Python Numbers

Number data types store numeric values. Number objects are created when you assign a value to them

## **Python Strings**

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of

strings can be taken using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

## Python Lists

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (\*) is the repetition operator.

## Python Tuples

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses. The main differences between lists and tuples are: Lists are enclosed in brackets ([ ]) and their elements and size can be changed, while tuples are enclosed in parentheses (( )) and cannot be updated. Tuples can be thought of as read-only lists.

## Python Dictionary

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object. Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

Different modes in python

Python has two basic modes: normal and interactive.

The **normal mode** is the mode where the scripted and finished .py files are run in the Python interpreter.

**Interactive mode** is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

## Twenty Python libraries

1. **Requests.** The most famous http library written by kenneth reitz. It's a must have for every python developer.
2. **Scrappy.** If you are involved in webscraping then this is a must have library for you. After using this library you won't use any other.

3. **wxPython.** A gui toolkit for python. I have primarily used it in place of tkinter. You will really love it.
4. **Pillow.** A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.
5. **SQLAlchemy.** A database library. Many love it and many hate it. The choice is yours.
6. **BeautifulSoup.** I know it's slow but this xml and html parsing library is very useful for beginners.
7. **Twisted.** The most important tool for any network application developer. It has a very beautiful api and is used by a lot of famous python developers.
8. **NumPy.** How can we leave this very important library ? It provides some advance math functionalities to python.
9. **SciPy.** When we talk about NumPy then we have to talk about scipy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.
10. **matplotlib.** A numerical plotting library. It is very useful for any data scientist or any data analyzer.
11. **Pygame.** Which developer does not like to play games and develop them ? This library will help you achieve your goal of 2d game development.
12. **Pyglet.** A 3d animation and game creation engine. This is the engine in which the famous python port of minecraft was made
13. **pyQT.** A GUI toolkit for python. It is my second choice after wxpython for developing GUI's for my python scripts.
14. **pyGtk.** Another python GUI library. It is the same library in which the famous Bittorrent client is created.
15. **Scapy.** A packet sniffer and analyzer for python made in python.
16. **pywin32.** A python library which provides some useful methods and classes for interacting with windows.
17. **nltk.** Natural Language Toolkit – I realize most people won't be using this one, but it's generic enough. It is a very useful library if you want to manipulate strings. But it's capacity is beyond that. Do check it out.
18. **nose.** A testing framework for python. It is used by millions of python developers. It is a must have if you do test driven development.
19. **SymPy.** SymPy can do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.



**20. IPython.** I just can't stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

## **Numpy**

NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called axes. The number of axes is rank.

- Offers Matlab-ish capabilities within Python
- Fast array operations
- 2D arrays, multi-D arrays, linear algebra etc.

## **Python class and objects**

These are the building blocks of OOP. class creates a new object. This object can be anything, whether an abstract data concept or a model of a physical object, e.g. a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently "the big thing" in most programming languages. The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something. Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest, it took me several months to get a handle on objects. However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered. Once you learn OOP, you'll realize that it's actually a pretty powerful tool. Plus many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing.

One thing to note about Python and OOP: it's not mandatory to use objects in your code in a way that works best; maybe you don't need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want. As you've already seen, Python can do just fine with functions. Unlike languages such as Java, you aren't tied down to a single way of doing things; you can mix functions and classes as necessary in the same program. This lets you build the code Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

### **Here's a brief list of Python OOP ideas:**

- The class statement creates a class object and gives it a name. This creates a new namespace.

- Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: `ClassName.Attribute`.
- Class attributes export the state of an object and its associated behavior. These attributes are shared by all instances of a class.
- Calling a class (just like a function) creates a new instance of the class.

This is where the multiple copies part comes in.

- Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.
- Using the term `self` identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

## **Inheritance**

First off, classes allow you to modify a program without really making changes to it. To elaborate, by subclassing a class, you can change the behavior of the program by simply adding new components to it rather than rewriting the existing components. As we've seen, an instance of a class inherits the attributes of that class. However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses. The subclasses can override the logic in a superclass, allowing you to change the behavior of your classes without changing the superclass at all.

## **Operator Overloads**

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc. Even though these actions can be implemented via class methods, using overloading ties the behavior closer to Python's object model and the object interfaces are more consistent to Python's built-in objects, hence overloading is easier to learn and use. User-made classes can override nearly all of Python's built-in operation methods

## **Exceptions**

I've talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program's flow, either intentionally or due to errors. They are special events that can occur due to an error, e.g. trying to open a file that doesn't exist, or when the program reaches a marker, such as the completion of a loop. Exceptions, by definition, don't occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python. Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples:

- Accessing a non-existent dictionary key will raise a `KeyError` exception.
- Searching a list for a non-existent value will raise a `ValueError` exception.
- Calling a non-existent method will raise an `AttributeError` exception.
- Referencing a non-existent variable will raise a `NameError` exception.
- Mixing datatypes without coercion will raise a `TypeError` exception.

One use of exceptions is to catch a fault and allow the program to continue working; we have seen this before when we talked about files. This is the most common way to use exceptions. When programming with the Python command line interpreter, you don't need to worry about catching exceptions. Your program is usually short enough to not be hurt too much if an exception occurs.

Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem. However, if the same error occurred in your real program, it will fail and stop working. Exceptions can be created manually in the code by raising an exception. It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don't put any overhead on the code processing. Because exceptions aren't supposed to happen very often, they aren't processed until they occur. Exceptions can be thought of as a special form of the `if/elif` statements. You can realistically do the same thing with `if` blocks as you can with exceptions. However, as already mentioned, exceptions aren't processed until they occur; `if` blocks are processed all the time. Proper use of exceptions can help the performance of your program.

The more infrequent the error might occur, the better off you are to use exceptions; using `if` blocks requires Python to always test extra conditions before continuing. Exceptions also make code management easier: if your programming logic is mixed in with error-handling `if` statements, it can be difficult to read, modify, and debug your program.

## **User-Defined Exceptions**

I won't spend too much time talking about this, but Python does allow for a programmer to create his own exceptions. You probably won't have to do this very often but it's nice to have the option when necessary. However, before making your own exceptions, make sure there isn't one of the built-in exceptions that will work for you. They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.

## **Python modules**

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library. To support this, Python has a way to put definitions in a file and use them in a script or in an

interactive instance of the interpreter. Such a file is called a module; definitions from a module can be imported into other modules or into the main module.

## Testing code

As indicated above, code is usually developed in a file using an editor. To test the code, import it into a Python session and try to run it. Usually there is an error, so you go back to the file, make a correction, and test again. This process is repeated until you are satisfied that the code works.

The entire process is known as the development cycle. There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid. This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

## Functions in Python

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function. You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called when needed. That means that a function is a piece of code written to carry out a specified task. To carry out that specific task, the function might or might not need multiple inputs. When the task is carried out, the function can or cannot return one or more values. There are three types of functions in python: `help()`, `min()`, `print()`.

## Python Namespace

Generally speaking, a namespace (sometimes also called a context) is a naming system for making names unique to avoid ambiguity. Everybody knows a namespacing system from daily life, i.e. the naming of people in firstname and family name (surname). An example is a network: each network device (workstation, server, printer) .The same file name can be used in different directories, the files can be uniquely accessed via the pathnames. Many programming languages use namespaces or contexts for identifiers. An identifier defined in a namespace is associated with that namespace. This way, the same identifier can be independently defined in multiple namespaces. (Like the same file names in different directories) Programming languages, which support namespaces, may have different rules that determine to which namespace an identifier belongs.

Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

Some namespaces in Python:

**global** names of a module

**local** names in a function or method invocation

**built-in** names: this namespace contains built-in functions (e.g. `abs()`, `cmp()`, ...) and built-in exception names

## Garbage Collection

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector. The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

## Python XML Parser

XML is a portable, open source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language. What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML. This is recommended by the World Wide Web Consortium and available as an open standard. XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone. XML Parser Architectures and APIs The Python standard library provides a minimal but useful set of interfaces to work with XML. The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces. Simple API for XML SAX : Here, you register callbacks for events of interest and then let the parser proceed through the document. This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory.

**Document Object Model DOM API :** This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree – based form to represent all the features of an XML document. SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files. SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

## 3.2 MongoDB

MongoDB is the database for today's applications, enabling you to:

- Leverage data and technology to maximize competitive advantage
- Reduce risk for mission-critical deployments
- Accelerate time-to-value
- Dramatically lower total cost of ownership

With MongoDB, you can build applications that were never possible with traditional relational databases. Here's how.

## **Fast, Iterative Development**

Scope creep and changing business requirements no longer stand between you and successful project delivery. A flexible data model coupled with dynamic schema, and idiomatic drivers, with powerful GUI and command line tools make it fast for developers to build and evolve applications. Automated provisioning and management enable continuous integration and delivery for highly productive operations. Contrast this against static relational schemas and complex operations that have hindered you in the past.

## **Flexible Data Model**

MongoDB stores data in flexible, JSON-like documents, making it easy for you to persist and combine data of any structure. The document model maps to the objects in your application code, making data easy to work with, without giving up schema governance controls, data access, complex aggregations, and rich indexing functionality. You can dynamically modify the schema without downtime. You spend less time prepping your data for the database, and more time putting your data to work.

## **Distributed Data Platform**

MongoDB can be run within and across geographically distributed data centers and cloud regions, providing new levels of availability and scalability. As your deployments grow in terms of data volume and throughput, MongoDB scales elastically with no downtime, and without changing your application. And as your performance and availability goals evolve, MongoDB lets you adapt flexibly, across data centers, with tunable consistency.

## **Integrated Feature Set**

Analytics and data visualization, text and geospatial search, graph processing, event-driven streaming data pipelines, in-memory performance and global replication allow you to deliver a wide variety of real-time applications on one technology, reliably and securely. RDBMS systems require additional, complex technologies demanding separate integration overhead and expense to do this well.

## **Lower TCO**

Application development teams can be 5x more productive when they use MongoDB. The fully managed Atlas cloud service means operations team are as well. MongoDB runs on commodity hardware, dramatically lowering costs. MongoDB offers on-demand, pay-as-you-go pricing and affordable annual subscriptions, including 24x7x365 global support. Your applications can be one tenth the cost to deliver compared to using a relational database.

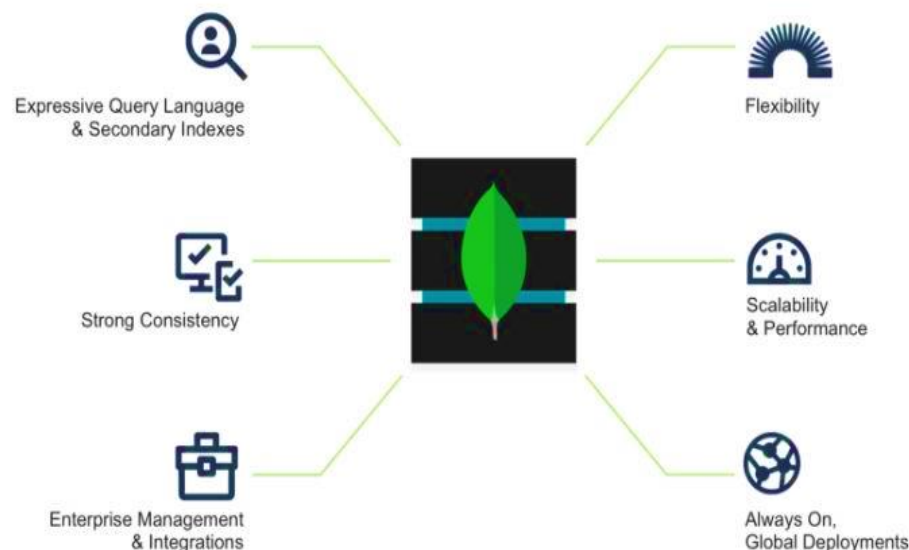
## **Long-Term Commitment**

MongoDB Inc. and the MongoDB ecosystem stand behind the world's fastest-growing database: 30M+ downloads, 4,900+ customers and over 1,000 partners. You can be sure your investment is protected.

## **Multi-Document ACID Transactions**

MongoDB 4.0, scheduled for Summer 2018\*, will add support for multi-document transactions, making it the only database to combine the ACID guarantees of traditional relational databases, the speed, flexibility, and power of the document model, with the intelligent distributed systems design to scale-out and place data where you need it. Sign up for the beta program.

MongoDB's design philosophy is focused on combining the critical capabilities of relational databases with the innovations of NoSQL technologies. Our vision is to leverage the work that Oracle and others have done over the last 40 years to make relational databases what they are today. Rather than discard decades of proven database maturity, MongoDB is picking up where they left off by combining key relational database capabilities with the work that Internet pioneers have done to address the requirements of modern applications.



Relational databases have reliably served applications for many years, and offer features that remain critical today as developers build the next generation of applications:

### **Expressive query language & secondary Indexes**

Users should be able to access and manipulate their data in sophisticated ways to support both operational and analytical applications. Indexes play a critical role in providing efficient access to data, supported natively by the database rather than maintained in application code.

### **Strong consistency**

Applications should be able to immediately read what has been written to the database. It is much more complex to build applications around an eventually consistent model, imposing significant work on the developer, even for the most sophisticated engineering teams.

### **Enterprise Management and Integrations**

Databases are just one piece of application infrastructure, and need to fit seamlessly into the enterprise IT stack. Organizations need a database that can be secured, monitored, automated, and integrated with their existing technology infrastructure, processes, and staff, including operations teams, DBAs, and data analysts. However, modern applications impose requirements not addressed by relational databases, and this has driven the development of NoSQL databases which offer:

### **Flexible Data Model**

NoSQL databases emerged to address the requirements for the data we see

dominating modern applications. Whether document, graph, key-value, or wide-column, all of them offer a flexible data model, making it easy to store and combine data of any structure and allow dynamic modification of the schema without downtime or performance impact.

### **Scalability and Performance**

NoSQL databases were all built with a focus on scalability, so they all include some form of sharding or partitioning. This allows the database to scale out on commodity hardware deployed on-premises or in the cloud, enabling almost unlimited growth with higher throughput and lower latency than relational databases.

### **Always-On Global Deployments**

NoSQL databases are designed for highly available systems that provide a consistent, high quality experience for users all over the world. They are designed to run across many nodes, including replication to automatically synchronize data across servers, racks, and data centers. While offering these innovations, NoSQL systems have sacrificed the critical capabilities that people have come to expect and rely upon from relational databases. MongoDB offers a different approach. With its Nexus Architecture, MongoDB is the only database that harnesses the innovations of NoSQL while maintaining the foundation of relational databases.

### **Data as Documents**

MongoDB stores data in a binary representation called BSON (Binary JSON). The BSON encoding extends the popular JSON (JavaScript Object Notation) representation to include additional types such as int, long, date, floating point, and decimal128. BSON documents contain one or more fields, and each field contains a value of a specific data type, including arrays, binary data and sub-documents. MongoDB BSON documents are closely aligned to the structure of objects in the programming language. This makes it simpler and faster for developers to model how data in the application will map to data stored in the database. MongoDB documents tend to have all data for a given record in a single document, whereas in a relational database information for a given record is usually spread across many tables.

For example, consider the data model for a blogging application. In a relational database, the data model would comprise multiple tables such as Categories, Tags, Users, Comments and Articles. In MongoDB the data could be modeled as two collections, one for users, and the other for articles. In each blog document there might be multiple comments, multiple tags, and multiple categories, each expressed as an embedded array.



	<b>MongoDB</b>	<b>Relational</b>	<b>Key-value</b>
<b>Rich Data Model</b>	Yes	No	No
<b>Dynamic Schema</b>	Yes	No	Yes
<b>Schema Validation</b>	Yes	Yes	No
<b>Typed Data</b>	Yes	Yes	No
<b>Data Locality</b>	Yes	No	Yes
<b>Field Updates</b>	Yes	Yes	No
<b>Easy for Programmers</b>	Yes	No	Not when modeling complex data structures

## **SYSTEM ANALYSIS**

## 4. SYSTEM ANALYSIS:

### 4.1 Existing System

There are some web interface tools which provide security for the personally identifiable information but before providing security we need to dump all our personal data on their servers where all the security mechanisms are applied in order to enforce the security.

Many of these web interface tools provide security for our data after signing up into their database and only provide security for few days as free trail and after the completion of the trail period we need to pay the money in order to get security for our data.

Most of these web interface tools are entrepreneurship oriented mainly utilized by entrepreneurs to safeguard their organizations private data

#### **SPIRION:**

Spirion introduces a new philosophy to data leakage prevention. Traditional Data-in-Motion data loss prevention approaches defend the border of an organization, trying to keep the bad guys out. If history has taught us anything, it is that hackers, viruses, malware, and others will always find a way onto our systems. Spirion goes to the source of the data loss problem and secures Data-at-Rest. Read more about the strength of Data-at-Rest versus Data-in-Motion.

Spirion offers Businesses the strongest Data-at-Rest data loss prevention solution on the market. Our software quickly and effectively finds sensitive data – data that can be anywhere and that most organizations do not even know still exists. With a number of configurations options, Spirion can be deployed to each machine, used to search agentlessly, or a hybrid approach. Beyond identification, we offer remediation capabilities to clean this data. This way you reduce your risk exposure. If hackers or viruses get onto your system, you dramatically reduce the chances of them finding sensitive data and a data breach from ever occurring again.

#### **GROUNDLABS:**

This have helped more than 2,500 organisations in 80 countries discover sensitive PII and cardholder data in documents, databases, emails, log files and many other locations.

### 4.2 Proposed system

Despite all of the great security products available today, the number of data breaches continues to rise. Even government organizations publicly display some of user content knowingly or unknowingly, it's our duty to inform them not to do so before the information goes into the wrong hands. This tool makes it possible by finding the appropriate urls from the google search, thus enabling the government of India (NIC) to do the job with an ease. The main of the project is to stop the misuse of personally identifiable information in any means. Our solution analyses the personally identifiable information

which is available open in the websites and also categorize the websites based on the security provided to a particular website. As the data leakage is the main problem now-a-days we mainly tend to target the owners of website who publish the personal data in their websites without the prior information of any particular person. It is a Web Enabled tool which is of free of cost, Robust which can be executed on any system and the tool from hackers.

Here the innocent people need not directly use this tool. National Information Centre Hyderabad will initiate the search and find the vulnerable sites and inform the owners of the sites accordingly which makes tool more.

### **4.3 Feasibility Study**

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. For feasibility analysis, some understanding of the major requirements for the system is essential. This is to ensure that the proposed system is not a burden to the company.

Three key considerations involved in the feasibility analysis are

#### **4.3.1. ECONOMICAL FEASIBILITY**

#### **4.3.2. TECHNICAL FEASIBILITY**

#### **4.3.3. SOCIAL FEASIBILITY**

##### **4.3.1. Economic Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

##### **4.3.2. Technical Feasibility**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

#### **4.3.3. Social Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

# **SYSTEM REQUIREMENT SPECIFICATION**

## 5. SYSTEM REQUIREMENT SPECIFICATION

### 5.1 Introduction

Software Requirements Specification plays an important role in creating quality software solutions. Specification is basically a representation process. Requirements are represented in a manner that ultimately leads to successful software implementation. Requirements may be specified in a variety of ways. However there are some guidelines worth following: - Representation format and content should be relevant to the problem. Information contained within the specification should be nested Diagrams and other notational forms should be restricted in number and consistent in use. Representations should be revisable

### 5.2 Non-functional requirements:

#### ➤ Usability

Usability is the ease of use and learns ability of a human-made object. The object of use can be a software application, website, book, tool, machine, process, or anything a human interacts with. A usability study may be conducted as a primary job function by a usability analyst or as a secondary job function by designers, technical writers, marketing personnel, and others.

#### ➤ Reliability

The probability that a component part, equipment, or system will satisfactorily perform its intended function under given circumstances, such as environmental conditions, limitations as to operating time, and frequently and thoroughness of maintenance for a specified period of time.

#### ➤ Performance:

Accomplishment of a given task measured against present standards of accuracy, completeness, cost, and speed.

#### ➤ Supportability:

To which the design characteristics of a stand by or support system meet the operational requirements of an organization.

#### ➤ Implementation:

Implementation is the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy.

#### ➤ Interface:

An interface refers to a point of interaction between components, and is applicable at the level of both hardware and software. This allows a component whether a piece of hardware such as a graphics card or a piece of software such as an internet browser to function independently while using interfaces to communicate with other components via an input/output system and an associated protocol.

➤ **Legal:**

It is established by or founded upon law or official or accepted rules of or relating to jurisprudence; “legal loop hole”. Having legal efficacy or force’, “a sound title to the property” Relating to or characteristic of the profession of law, “the legal profession”. Allowed by official rules; “a legal pass receiver”.

## **5.3 System Requirements**

The following are the system requirements

### **5.3.1 Hardware Requirements:**

<b>Processor Type</b>	<b>: Intel (any version)</b>
<b>Speed</b>	<b>: 1.1 GHZ</b>
<b>RAM</b>	<b>: 4GB</b>
<b>Hard disk</b>	<b>: 20 GB</b>
<b>Keyboard</b>	<b>: 101/102 Standard Keys</b>

### **5.3.2 Software Requirements:**

- Operating System : Windows/Linux**
- Coding Language : Python**
- Database : MongoDB**
- Tools : Robo3T**



## **SYSTEM DESIGN**

## **6. SYSTEM DESIGN**

### **6.1 Introduction**

The most creative and challenging phase of the life cycle is system design. The term design describes a final system and the process by which it is developed. It refers to the technical specifications that will be applied in implementations of the candidate system. The design may be defined as “the process of applying various techniques and principles for the purpose of defining a device, a process or a system with sufficient details to permit its physical realization”.

The designer’s goal is how the output is to be produced and in what format. Samples of the output and input are also presented. Second input data and database files have to be designed to meet the requirements of the proposed output.

The processing phases are handled through the program Construction and Testing. Finally, details related to justification of the system and an estimate of the impact of the candidate system on the user and the organization are documented and evaluated by management as a step toward implementation.

The importance of software design can be stated in a single word “Quality”. Design provides us with representations of software that can be assessed for quality. Design is the only way where we can accurately translate a customer’s requirements into a complete software product or system. Without design we risk building an unstable system that might fail if small changes are made. It may as well be difficult to test, or could be one who’s quality can’t be tested. So it is an essential phase in the development of a software product.

### **6.2 High-level design**

High Level Design defines complete scale architecture of the developing system required. In short it is an overall representation of a design required for our target developing system/application. It is usually done by higher level professionals/software architects

### **6.3 Low-level design**

#### **6.3.1 UML Diagrams**

The UML is a language for

- Visualizing
- Specifying
- Constructing
- Documenting

These are the artefacts of a software-intensive system.

## A conceptual model of UML:

The three major elements of UML are

1. The UML's basic building blocks
2. The rules that dictate how those building blocks may be put together.
3. Some common mechanisms that apply throughout the UML.

## Basic building blocks of the UML

The vocabulary of UML encompasses three kinds of building blocks:

1. Things
2. Relationships
3. Diagrams

Things are the abstractions that are first-class citizens in a model;

Relationships tie these things together;

Diagrams group the interesting collection of things.

Things in UML: There are four kinds of things in the UML

1. Structural things
2. Behavioural things.
3. Grouping things
4. Annotational things

These things are the basic object oriented building blocks of the UML. They are used to write well-formed models.

### 6.3.1.1 STRUCTURAL THINGS:

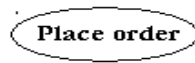
**Structural things** are the nouns of UML models. The structural things used in the project design are:

First, a **class** is a description of a set of objects that share the same attributes, operations, relationships and semantics.

Window
Origin Size
Open () Close () Move () Display ()

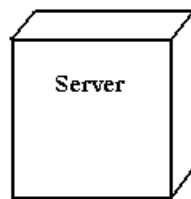
**Fig: Class**

Second, a **use case** is a description of set of sequence of actions that a system performs that yields an observable result of value to particular actor.



**Fig: Use Cases**

Third, a node is a physical element that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability.

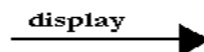


**Fig: Nodes**

**6.3.1.2 Behavioral things** are the dynamic parts of UML models. The behavioral thing used is:

### **Interaction:**

An interaction is a behavior that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. An interaction involves a number of other elements, including messages, action sequences (the behavior invoked by a message, and links (the connection between objects).



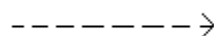
**Fig: Messages**

### **6.3.1.3 Relationships in UML**

There are four kinds of relationships in the UML:

- Dependency
- Association
- Generalization
- Realization

A **dependency** is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing).



**Fig: Dependencies**

An **association** is a structural relationship that describes a set links, a link being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts.

---

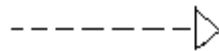
**Fig: Association**

A **generalization** is a specialization/ generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).



**Fig: Generalization**

A **realization** is a semantic relationship between classifiers, where in one classifier specifies a contract that another classifier guarantees to carry out.



**Fig: Realization**

#### 6.3.1.4 USECASE DIAGRAM:

**User:** They have accessibility upon the overall system with specific to the data insertion, deletion,

**Updation and queries.** They are the highest authorities within the system, which have maximum. Control upon the entire database.

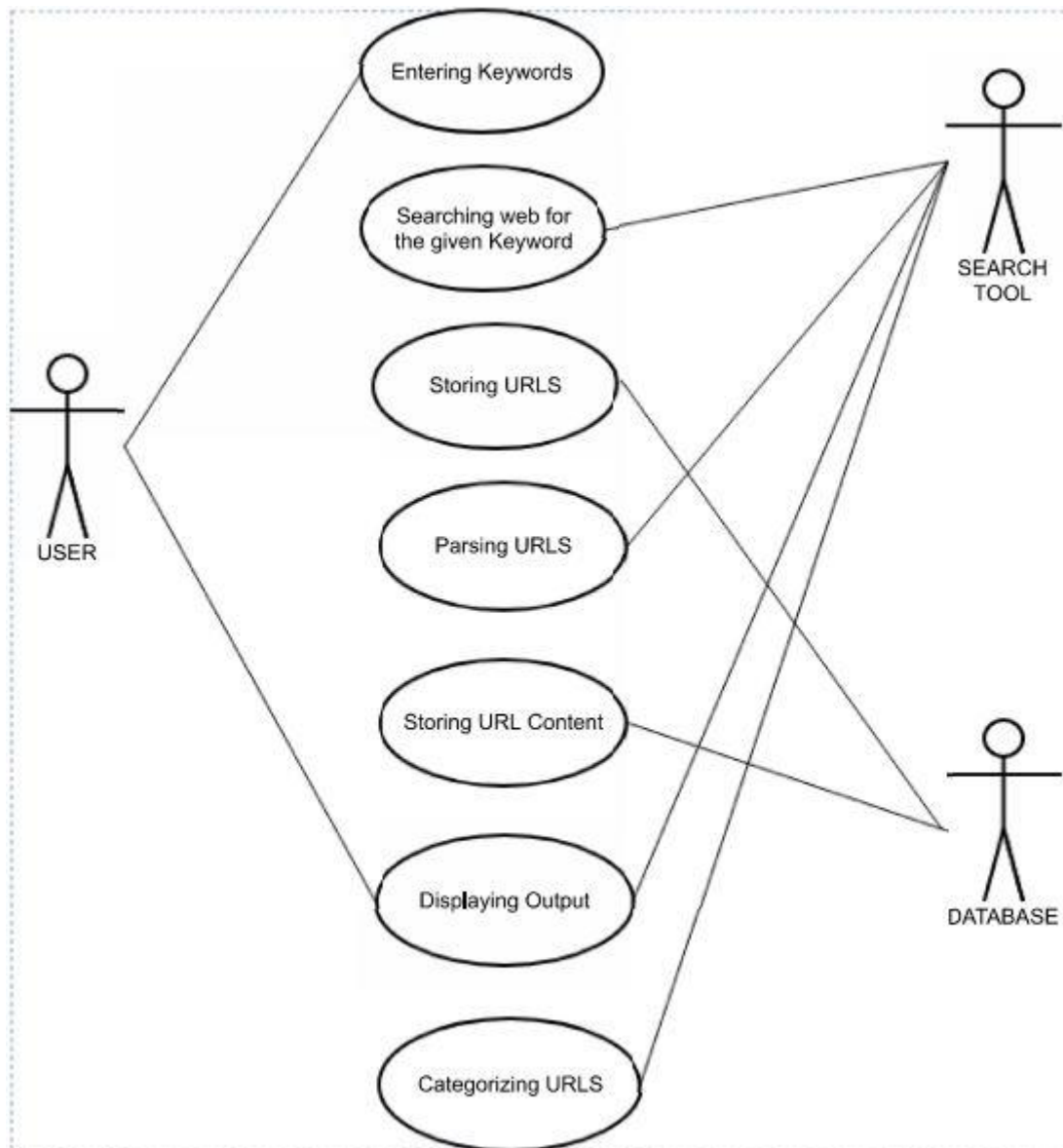


Fig Use Case Diagram

#### 6.3.1.5 Class Diagram

An object is any person, place, thing, concept, event, screen, or report applicable to your system. Objects both know things (they have attributes) and they do things (they have methods). A class is a representation of an object and, in many ways, it is simply a template from which objects are created. Classes form the main building blocks of an object-oriented application.

Responsibilities:

Classes are typically modelled as rectangles with three sections: the top section for the name of the class, the middle section for the attributes of the class, and the bottom section for the methods of the class. Attributes are the information stored about an object, while methods are the things an object or class do. You should think of methods as the object-oriented equivalent of functions and procedures.

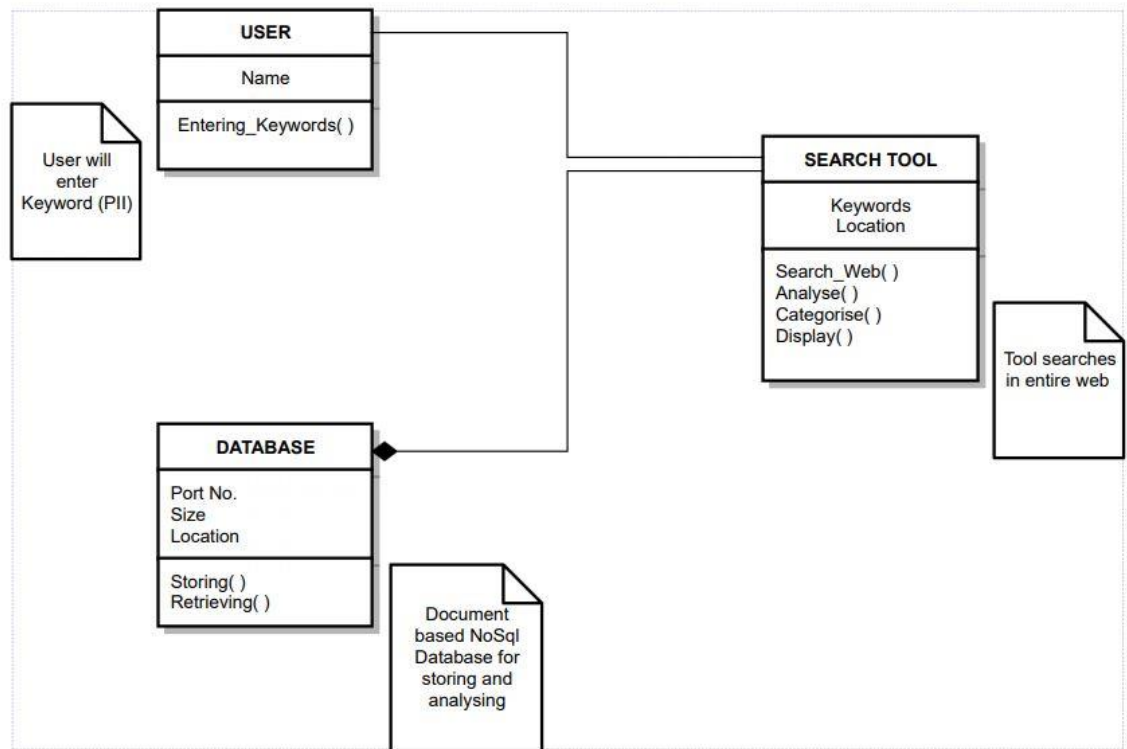


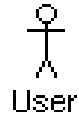
Fig 5.2 Class diagram

### 6.3.1.6 Sequence Diagram

UML sequence diagrams are used to represent the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram. Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behaviour which can be difficult to extract from static diagrams or specifications.

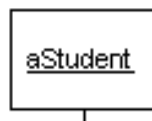
## Actor

Represents an external person or entity that interacts with the system



## Object

Represents an object in the system or one of its components



## *Sequence Diagram Body Elements*

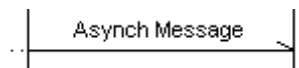
### Action

Represents an action taken by an actor, object or unit



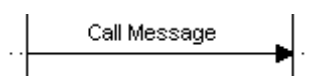
## Asynchronous Message

An asynchronous message between header elements



## Call Message

A call (procedure) message between header elements





## Create Message

A "create" message that creates a header element (represented by lifeline going from dashed to solid pattern)



## Message

A simple message between header elements

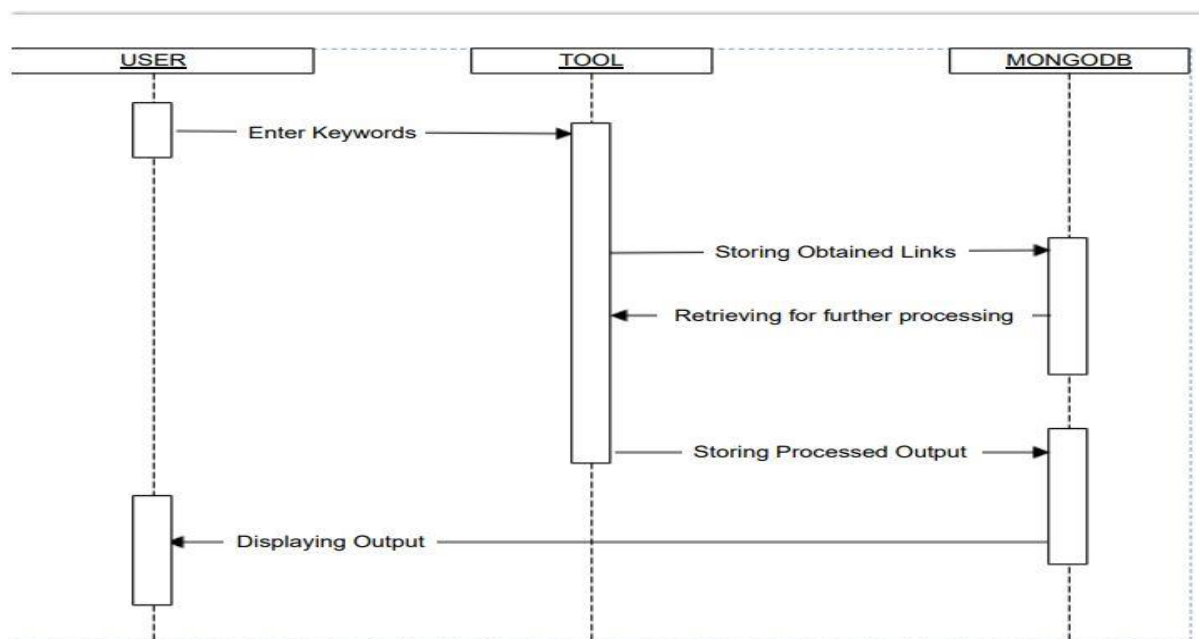


Fig 5.3 Sequence Diagram

## **WORKING DESCRIPTION**

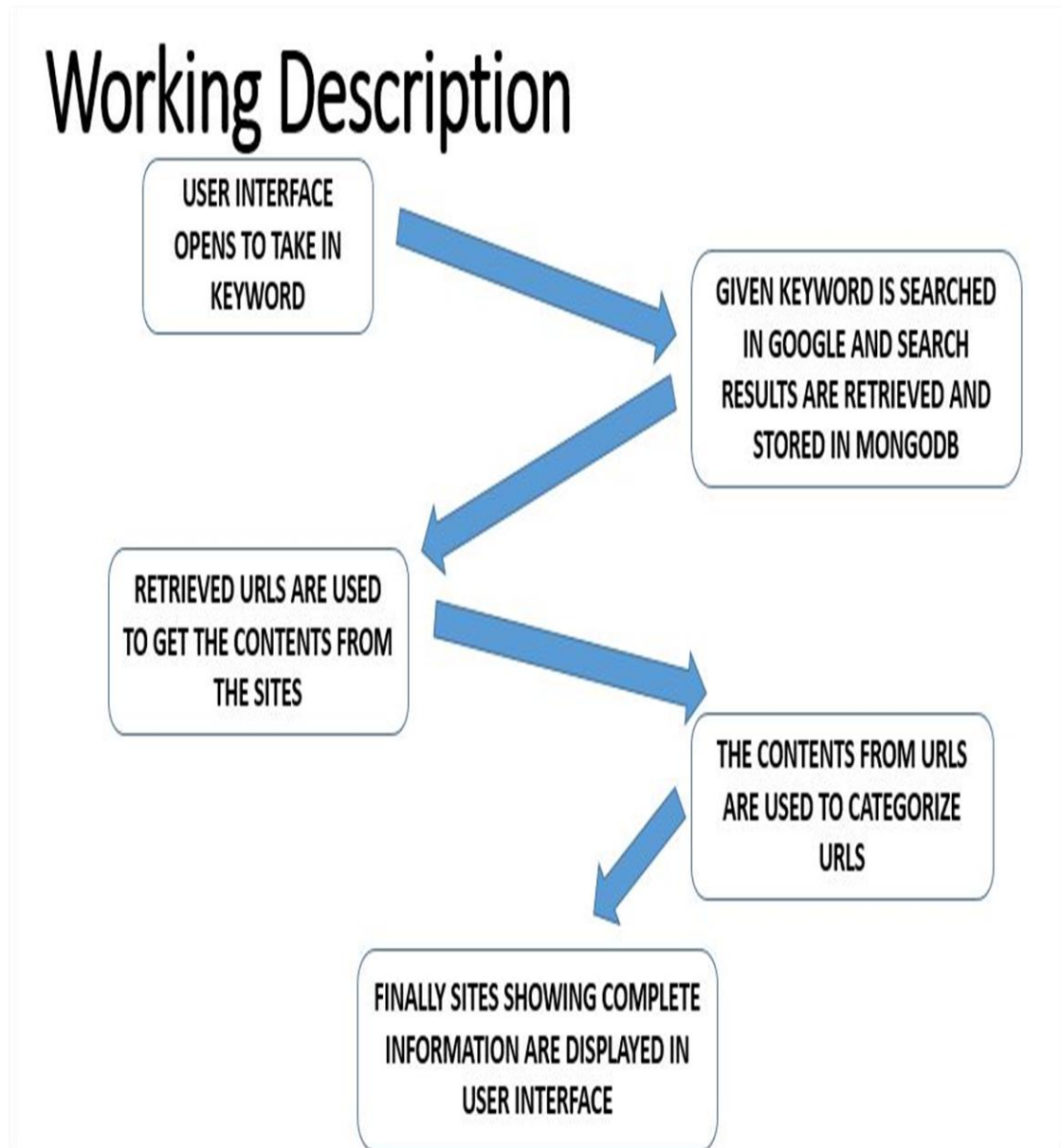
## **7. WORKING DESCRIPTION**

### **7.1 Principle of working:**

- When we execute the program a User Interface opens up showing an empty Entry Box for taking input, a GO Button for starting the process, a Text Box for showing the program status & showing the output and a drop down menu for displaying categories for displaying the final result.
- We have to give some PII as keyword in the Entry Box for the process to start getting the results and categorizing.
- So, when we enter a keyword in the Entry box the keyword is searched for in Google Search Engine and the top search result links are retrieved.
- Connection to Mongo Database Server is established for storing URLs in a collection for further processing. Later these Links (URLs) are used to retrieve contents from the respective sites and this data is utilized for categorization of the URL on the basis of different criteria.
- The URLs are categorized on the basis of whether the keyword (PII) was found completely in the contents or partially. Also, URLs are categorized on the basis of whether the data was publicly (open to everyone) available or the data requires LOGIN or if the data requires OTP or at least a CAPTCHA for accessing the data. Connection to Mongo Database is terminated.
- A message saying “Insertion Completed and MongoDB connection terminated” is shown on the screen after which the results can be seen. After the Categorization is finished and the results are ready the User can use Buttons on the User Interface to get the result on the basis of Categories.
- By selecting which category he/she wants from a Drop down menu and clicking ok. Connection to Mongo Database is again established to execute the query and display the results. The results are displayed in the Text box only erasing the previous status results on the basis of latest results first. Mongo Database connection is terminated.

## 7.2 FLOW CHART

The following is the diagram showing the working description:



## **CODING**

## 8. CODING

The following are the packages used in code

**tkinter** – Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.

**pymongo** - The PyMongo distribution contains tools for interacting with MongoDB database from Python. The bson package is an implementation of the BSON format for Python. The pymongo package is a native Python driver for MongoDB.

**beautifulsoup** - Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

**re** - re module provides excellent support for regular expressions, with a modern and complete regex flavor. The only significant features missing from Python's regex syntax are atomic grouping, possessive quantifiers, and Unicode properties.

```
import urllib.request
from tkinter import *
from pymongo import MongoClient
from urllib.request import urlopen
from bs4 import BeautifulSoup
import re

#Keyword search in contents stored in database
def keysrchincontnt(cont):
    key = e1.get()
    adr = key.replace("'", "'].?[")
    adr = adr[3:-3]
    adr = adr.replace("[ ].?", "")
    ada = "\\b(" + adr + ")\\b"
    ad = re.findall(ada, cont, re.I)
    if ad:
        return "Full"
    else:
        adr = key.replace("'", "X"].?[")
        adr = adr[4:-3]
        adr = adr.replace("[ X].?", "")
        ada = "\\b(" + adr + ")\\b"
        ad = re.findall(ada, cont, re.I)
        if ad:
            return "Partial,masked"
        else:
```

```

        return "Substring,Notfound"

#Categorizing into CAPTCHA, LOGIN, OTP
def categorize(str):
    category = " "
    ad = re.search(r'captcha', str, re.I)
    if(ad):
        category = "CAPTCHA"
    ad = re.search(r'log ?in', str, re.I)
    if(ad):
        category = category + "LOGIN"
    ad = re.search(r'otp', str, re.I)
    if(ad):
        category = category + "OTP"
    return category

#Extracting Contents from a url
def contentsfmpage(url):
    try:
        op.insert(INSERT, url + "\n")
        print(url)
        headers = { }
        headers['User-Agent'] = "Mozilla/5.0 (X11; Linux x86_64; rv:47.0)
Gecko/20100101 Firefox/47.0"
        req = urllib.request.Request(url, headers=headers)
        resp = urllib.request.urlopen(req)
        pagehtml = resp.read()
        pagesoup = BeautifulSoup(pagehtml,"html.parser")
        pagesoup = str(pagesoup)
        print("\n\n")
        return pagesoup
    except Exception as e:
        op.insert(INSERT, e)
        op.insert(INSERT, "\n")
        print(e)
        return str(e)
    window.update_idletasks()

#Database Connection and Insertion of link, content, categorization, Presence into the
database
def lnkstodatabase():
    window.update_idletasks()
    # database connection and storage
    client = MongoClient('127.0.0.1', 27017)
    db = client.test

```

```

coltn = db.adarsh
op.insert(INSERT, "\nDatabase Connection formed successfully \n Inserting into
Mongodb:\n")

try:
for lnk in linklist:
    cont = contentsfmpage(lnk)
    coltn.insert({"SearchUrl": lnk, "Category": categorize(cont), "Presence":
keysrchincontnt(cont)})
    op.insert(INSERT, "Successfully Inserted in DB\n")
    window.update_idletasks()
    print("Successfully Inserted in DB\n")
coltn = db.key
coltn.insert({"Key": e1.get()})
op.insert(INSERT, "Keyword inserted in Keys collection in test\n")
client.close()
op.insert(INSERT, "\n Insertion Complete...\n Database Connection Terminated")
except Exception as e:
    op.insert(INSERT, e)
    op.insert(INSERT, "\n")
    window.update_idletasks()
    print(e)

def keywordsearch():
    op.insert(INSERT, "Processing...\n")
    window.update_idletasks()
    keys = e1.get()
    keys = keys.replace(" ", "+")
    url = "https://www.google.com/search?q=" + keys
    headers = { }
    headers['User-Agent'] = "Mozilla/5.0 (X11; Linux x86_64; rv:47.0) Gecko/20100101
Firefox/47.0"
    req = urllib.request.Request(url, headers=headers)
    try:
        resp = urllib.request.urlopen(req)
        respData = resp.read()
        soup = BeautifulSoup(respData, "html.parser")
        link = soup.findAll("h3", {"class": "r"})
        print("\n\n")
        op.insert(INSERT, "Links retrieved from Google:\n")
        for lin in link:
            match = re.search(r'href=[\"]?([^\"] >)]+', str(lin))
            if match:

```



```

        ser = match.group(0)
        po = ser[6:]
        print(po)
        op.insert(INSERT, po + "\n")
        linkslst.append(po)
    lnkstodatabase()

    except Exception as e:
        print(e)
        op.insert(INSERT, e)
        op.insert(INSERT, "\n")

#####
linkslst = []

#UI code

window = Tk()
window.title("NIC PROJECT-2018")
window.geometry('800x600')
l = Label(window, text="")
l.grid(row=0, column=0)
Label(window, text="Enter Keys :", font=(14)).place(x=370, y=10)
e1 = Entry(window)
print(e1.get())
e1.place(x=350, y=40)
op = Text(window, height=28, width=95)
op.place(x=20, y=120)
#Buttons
Button(window, text='Quit', command=window.quit).place(x=365, y=80)
Button(window, text='Search', command=keywordsearch).place(x=415, y=80)
#
window.mainloop()

```

# TESTING

## **9. TESTING:**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### **9.1 TYPES OF TESTS:**

#### **9.1.1 Unit testing:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

#### **9.1.2 Integration testing:**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were

Individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components

#### **9.1.3 Functional testing:**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

#### **9.1.4 System Test:**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

#### **9.1.5 White Box Testing:**

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

#### **9.1.6 Black Box Testing:**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

#### **9.1.7 Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

### **9.2 TEST STRATEGY AND APPROACH:**

Field testing will be performed manually and functional tests will be written in detail.

#### **9.2.1 Test objectives:**

All field entries must work properly. Pages must be activated from the identified link. The entry screen, messages and responses must not be delayed.

#### **9.2.2 Features to be tested:**

Verify that the entries are of the correct format. No duplicate entries should be allowed

#### **9.2.3 Integration Testing:**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Table 9.2.4 Test Cases**

S.no	Testcase name	Testcase Description	Expected Output	Actual Output	Result
1	7799626311	Phone Number	facebook.com/7799626311 superfriends.blogspot.com	facebook.com/7799626311 superfriends.blogspot.com	success
2	916091740509	Aadhar Card Number	Did not match any document	Did not match any document	success
3	Blank	Test	Please Enter Keyword	Please Enter Keyword	success
4	Person Name	Syed Sajid	Wikipedia.org	Facebook.com Wikipedia.org	success
5	E53B69	Random Numbers	Nothing	colorhexa.com	success
6	06061997	DOB	Bhanu Bangarugadda	br.op.gg/summoner/userName=06061997	fail
7	bp.bhanu1997@gmail.com	Email	Freshersworld.com/bhanubangarugadda	Gmail.com, Freshersworld.com/bhanubangarugadda	success
8	agtagtag	Random	nothing	https://twitter.com/agtagtagt	success

**Test Results:**

All the test cases mentioned above passed successfully and one of them is a fail which is intentional. No defects encountered.

**Acceptance Testing:**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

## **Results and Screenshots**

## 10.Results and Screenshots

It gives in brief general idea of possible outcomes, experimental analysis result and advantages of the system. A prototype of car has been developed and our system has been implemented in it.

### 10.1 User Interface

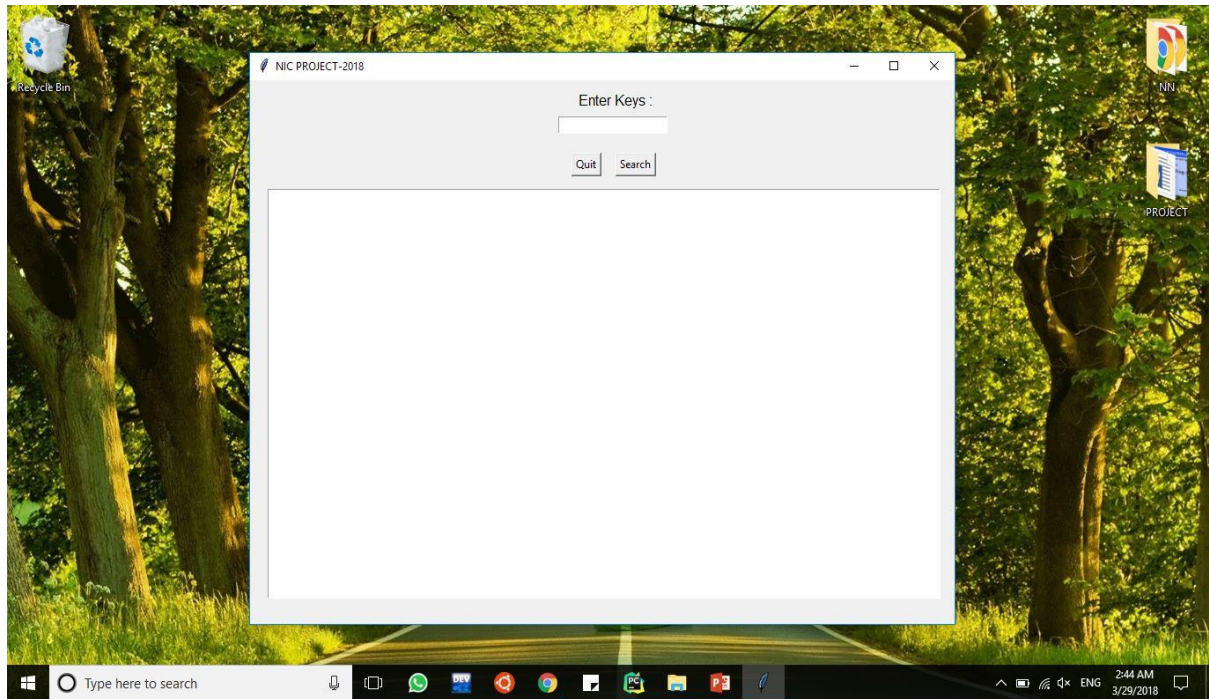
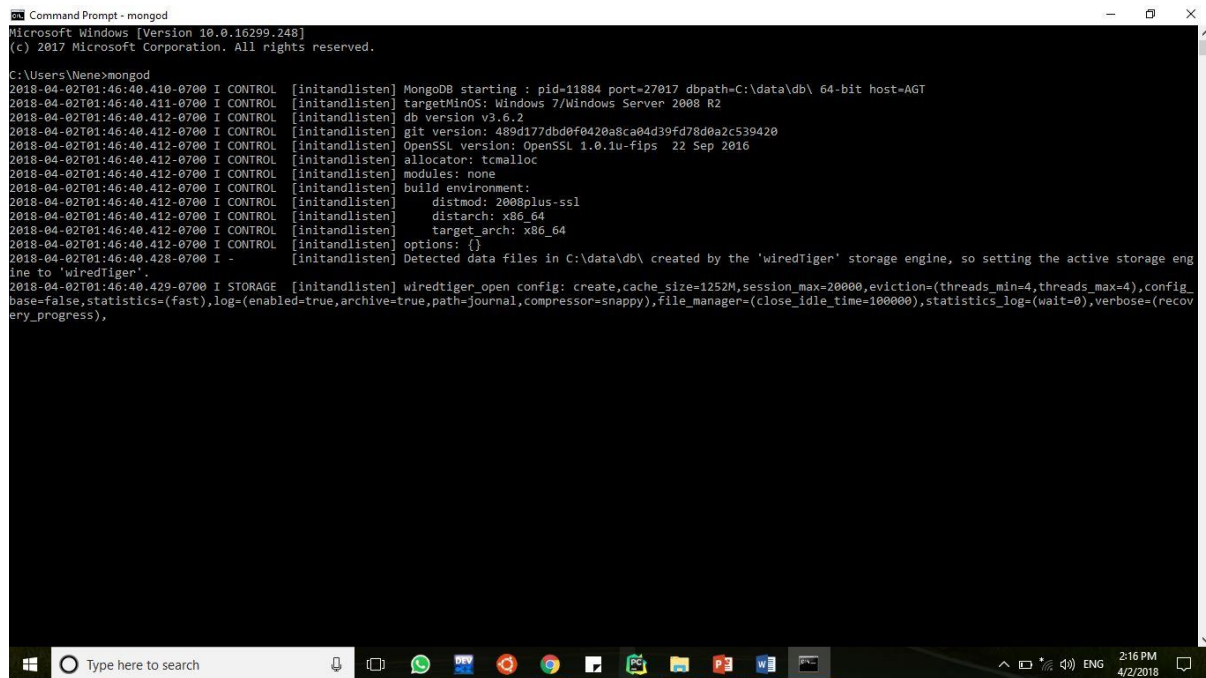


Fig 10.1 User Interface

## 10.2 Starting Mongo Database Server



```
Command Prompt - mongod
Microsoft Windows [Version 10.0.16299.248]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Nene>mongod
2018-04-02T01:46:40.410-0700 I CONTROL [initandlisten] MongoDB starting : pid=11884 port=27017 dbpath=C:\data\db\ 64-bit host=AGT
2018-04-02T01:46:40.411-0700 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] db version v3.6.2
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] git version: 489d177dbd0f0420a8ca04d39fd78d0a2c539420
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.1u-fips 22 Sep 2016
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] allocator: tcmalloc
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] modules: none
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] build environment:
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] distmod: 2008plus-ssl
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] distarch: x86_64
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] target arch: x86_64
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] options: {}
2018-04-02T01:46:40.428-0700 I - [initandlisten] Detected data files in C:\data\db\ created by the 'wiredtiger' storage engine, so setting the active storage eng
ine to 'wiredtiger'.
2018-04-02T01:46:40.429-0700 I STORAGE [initandlisten] wiredtiger_open config: create,cache_size=1252M,session_max=20000,eviction=(threads_min=4,threads_max=4),config_
base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000),statistics_log=(wait=0),verbose=(recov
ery_progress),
```

Fig 10.2 Starting Mongo Database Server



## 10.3 Mongo Database Server running on Port 27017

```
Command Prompt - mongod
Microsoft Windows [Version 10.0.16299.248]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Nene>mongod
2018-04-02T01:46:40.410-0700 I CONTROL [initandlisten] MongoDB starting : pid=11884 port=27017 dbpath=C:\data\db\ 64-bit host=AGT
2018-04-02T01:46:40.411-0700 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] db version v3.6.2
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] git version: 489d177dbd0f0420a8ca04d39fd78d0a2c539420
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.1u-fips 22 Sep 2016
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] allocator: tcmalloc
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] modules: none
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] build environment:
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten]   distmod: 2008plus-ssl
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten]   distarch: x86_64
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten]   target_arch: x86_64
2018-04-02T01:46:40.412-0700 I CONTROL [initandlisten] options: {}
2018-04-02T01:46:40.428-0700 I - [initandlisten] Detected data files in C:\data\db\ created by the 'wiredTiger' storage engine, so setting the active storage eng
ine to 'wiredTiger'.
2018-04-02T01:46:40.429-0700 I STORAGE [initandlisten] wiredtiger_open config: create,cache_size=1252M,session_max=20000,eviction=(threads_min=4,threads_max=4),config_
base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000),statistics_log=(wait=0),verbose=(recov
ery_progress),
2018-04-02T01:46:41.714-0700 I STORAGE [initandlisten] WiredTiger message [1522658801:713373][11884:140723328868688], txn-recover: Main recovery loop: starting at 31/6
528
2018-04-02T01:46:42.271-0700 I STORAGE [initandlisten] WiredTiger message [1522658802:271425][11884:140723328868688], txn-recover: Recovering log 31 through 32
2018-04-02T01:46:42.732-0700 I STORAGE [initandlisten] WiredTiger message [1522658802:731877][11884:140723328868688], txn-recover: Recovering log 32 through 32
2018-04-02T01:46:45.225-0700 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] ** addresses it should serve responses from, or with --bind_ip_all to
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] ** bind to all interfaces. If this behavior is desired, start the
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] ** server with --bind_ip 127.0.0.1 to disable this warning.
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] ** WARNING: The file system cache of this machine is configured to be greater than 40% of the total memory. This
can lead to increased memory pressure and poor performance.
2018-04-02T01:46:45.229-0700 I CONTROL [initandlisten] See http://dochub.mongodb.org/core/wt-windows-system-file-cache
2018-04-02T01:46:45.229-0700 I CONTROL [initandlisten]
2018-04-02T01:46:48.848+0530 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory 'C:\data\db\diagnostic.data'
2018-04-02T14:16:48.923+0530 I NETWORK [initandlisten] waiting for connections on port 27017
```

Fig 10.3 Mongo Database Server running on Port 27017

## 10.4 Giving input to the User Interface

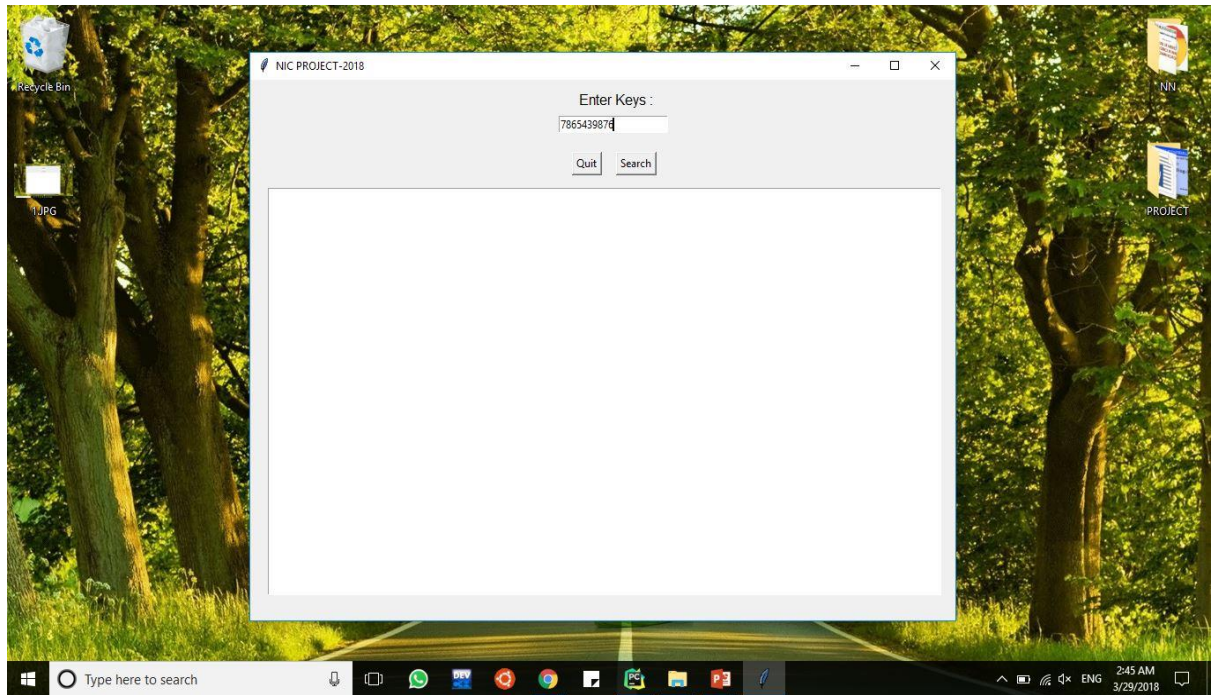


Fig 10.4 Giving input in the User Interface

## 10.5 Showing Process of Code

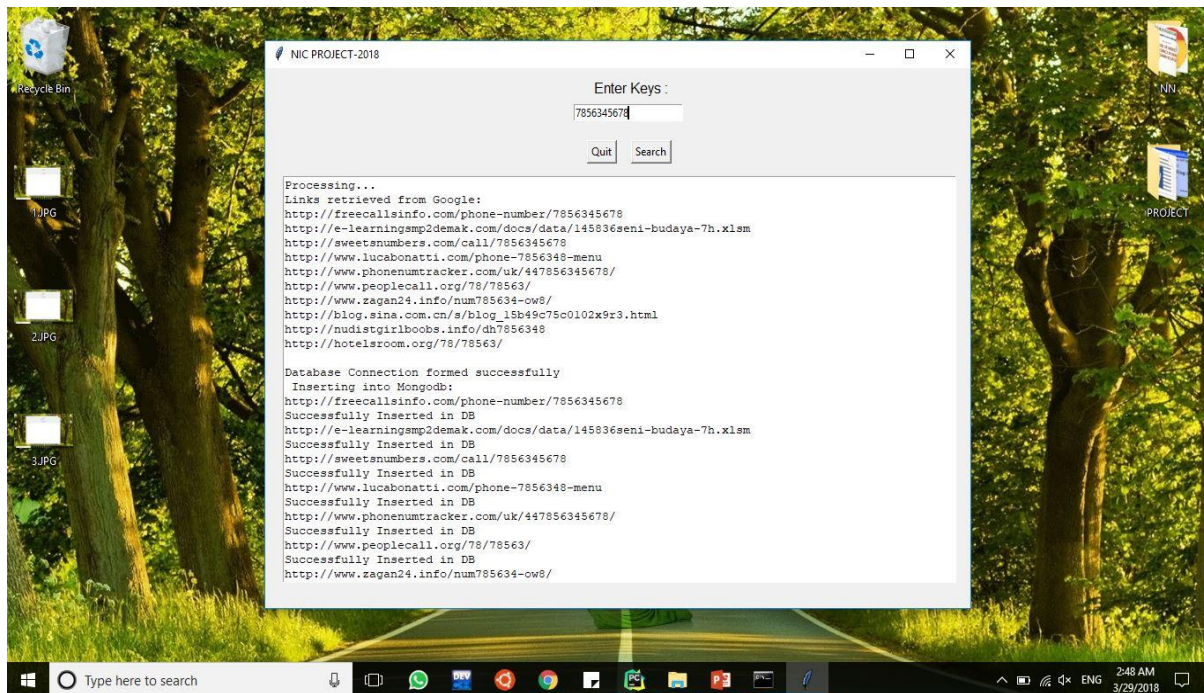


Fig 10.5 Showing Process of Code



## 10.6 Giving Final Output

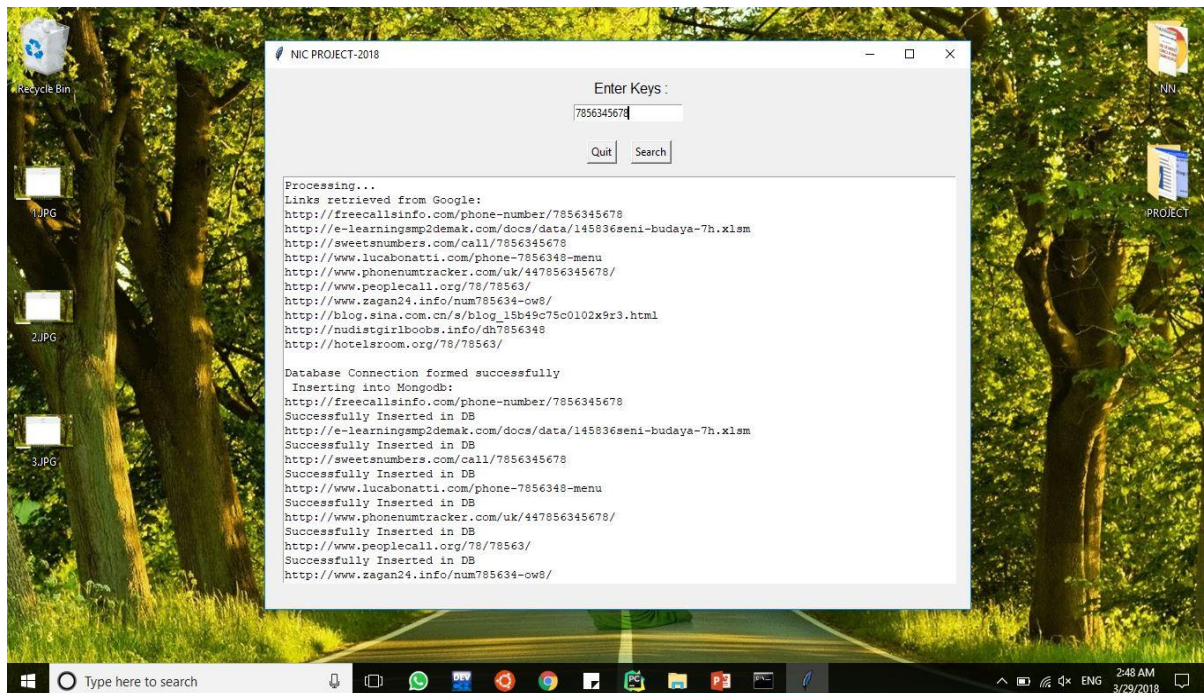


Fig 10.6 Giving Final Output

## 10.7 Giving Final Output

```
Command Prompt - mongo
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Wene>mongo
MongoDB shell version v3.6.2
connecting to: mongod://127.0.0.1:27017
MongoDB server version: 3.6.2
Server has startup warnings:
2018-04-02T01:46:45.225-0700 I CONTROL [initandlisten]
2018-04-02T01:46:45.225-0700 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten] **          Read and write access to data and configuration is unrestricted.
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten]
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] **          Remote systems will be unable to connect to this server.
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] **          Start the server with --bind_ip <address> to specify which IP
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] **          addresses it should serve responses from, or with --bind_ip_all to
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] **          bind to all interfaces. If this behavior is desired, start the
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] **          server with --bind_ip 127.0.0.1 to disable this warning.
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten]
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] ** WARNING: The file system cache of this machine is configured to be greater than 40% of the total memory. This
2018-04-02T01:46:45.229-0700 I CONTROL [initandlisten] **          can lead to increased memory pressure and poor performance.
2018-04-02T01:46:45.229-0700 I CONTROL [initandlisten] See http://dochub.mongodb.org/core/ut-windows-system-file-cache
2018-04-02T01:46:45.229-0700 I CONTROL [initandlisten]
>
```

Fig 10.7 Giving Final Output

## 10.8 Giving Final Output

```

Command Prompt - mongo
MongoDB server version: 3.6.2
Server has startup warnings:
2018-04-02T01:46:45.225-0700 I CONTROL [initandlisten]
2018-04-02T01:46:45.225-0700 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten] Read and write access to data and configuration is unrestricted.
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten]
2018-04-02T01:46:45.226-0700 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] Remote systems will be unable to connect to this server.
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] ** Start the server with --bind ip address to specify which ip
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] addresses it should serve responses from, or with --bind ip all to
2018-04-02T01:46:45.227-0700 I CONTROL [initandlisten] bind to all interfaces. If this behavior is desired, start the
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] server with --bind ip 127.0.0.1 to disable this warning.
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten]
2018-04-02T01:46:45.228-0700 I CONTROL [initandlisten] ** WARNING: The file system cache of this machine is configured to be greater than 40% of the total memory. This
can lead to increased memory pressure and poor performance.
2018-04-02T01:46:45.229-0700 I CONTROL [initandlisten] See http://dochub.mongodb.org/core/nt-windows-system-file-cache
2018-04-02T01:46:45.229-0700 I CONTROL [initandlisten]
> use test
switched to db test
> db.adarsh.find()
{ "_id" : "5ab01648f9612e233ca2ea7e5", "SearchUrl" : "https://in.linkedin.com/in/syed-thowfeeq-ahmed-268385110", "Category" : " " }
>
{ "_id" : "5ab01649f9612e233ca2ea7e6", "SearchUrl" : "https://in.linkedin.com/in/syed-thowfeeq-98b98b50", "Category" : " " }
>
{ "_id" : "5ab0164cf9612e233ca2ea7e7", "SearchUrl" : "https://www.facebook.com/thowfeeq178", "Category" : " LOGIN" }
>
{ "_id" : "5ab01651f9612e233ca2ea7e8", "SearchUrl" : "https://www.facebook.com/public/Syed-Ahmed-Tho", "Category" : " LOGIN" }
>
{ "_id" : "5ab01655f9612e233ca2ea7e9", "SearchUrl" : "https://sayat.me/thfq", "Category" : " LOGIN" }
>
{ "_id" : "5ab01652f9612e233ca2ea7ea", "SearchUrl" : "http://www.hackathon.io/e14-01-sbmind", "Category" : " LOGIN" }
>
{ "_id" : "5ab01653f9612e233ca2ea7eb", "SearchUrl" : "https://en.wikipedia.org/wiki/Syed_Ahmed", "Category" : " LOGINNOTP" }
>
{ "_id" : "5ab01654f9612e233ca2ea7ec", "SearchUrl" : "https://en.wikipedia.org/wiki/Syed_Ahmed_(businessman)", "Category" : " LOGINNOTP" }
>
{ "_id" : "5ab01656f9612e233ca2ea7ed", "SearchUrl" : "https://books.google.co.in/books?id=d33mP8FvW8C&pg=PA901&dq=syed+thowfeeq+ahmed&am
p;source=bl&amp;ots=CuRjIMDwX&camp;sig=SCy7V3j3qj-XwLl_8JESOPU2oV8&camp;len&amp;sa=X&ved=0ahUKEwI4H5X-1fnZAJM0QKHYGQAEUQ6AEitAI", "Category" : " " }
>
{ "_id" : "5ab01a1ef9612e2399ef36275", "SearchUrl" : "https://colnmarketcap.com/currencies/binance-com", "Category" : " " }
>
{ "_id" : "5ab01a3ef9612e2399ef36276", "SearchUrl" : "https://www.viv8r.com", "Category" : " LOGIN" }
>
{ "_id" : "5ab0272f9612e345c980849", "SearchUrl" : "https://www.youtube.com/watch?v=Av7rnIkElyU", "Category" : " LOGIN" }
>
{ "_id" : "5ab0278f9612e345c980849", "SearchUrl" : "https://www.youtube.com/watch?v=xv68sJmkZ4", "Category" : " LOGINNOTP" }
>
{ "_id" : "5ab0280f9612e345c980849", "SearchUrl" : "http://pubs.acs.org/doi/abs/10.1021/jp9812684", "Category" : " " }
>
{ "_id" : "5ab0286f9612e345c980849", "SearchUrl" : "https://pubs.acs.org/doi/pdf/10.1021/jp9812684", "Category" : " " }
>
{ "_id" : "5ab028af9612e345c980849", "SearchUrl" : "https://pubs.acs.org/doi/full/10.1021/jp9812684", "Category" : " " }
>
{ "_id" : "5ab03aaf9612e334e84873", "SearchUrl" : "http://www.hgf.com/", "Category" : " LOGIN", "Presence" : "Full" }
>
{ "_id" : "5ab03abf9612e334e84874", "SearchUrl" : "http://www.hgf.com/careers/", "Category" : " LOGIN", "Presence" : "Full" }
>
{ "_id" : "5ab03bbf9612e334e84875", "SearchUrl" : "http://www.hgf.com/about-us/our-people/", "Category" : " LOGIN", "Presence" : "Full" }
>
{ "_id" : "5ab03bbf9612e334e84876", "SearchUrl" : "http://www.ncbi.nlm.nih.gov/gene/3982", "Category" : " " , "Presence" : "Full" }
>
Type "it" for more

```

### Fig 10.8 Giving Final Output

## **CONCLUSION**

## **11. CONCLUSION:**

Our tool is able to detect publicly available Personally Identifiable Information (PII) of people which can be misused. It gives a list of URLs which are publicly showing PII and further this list maybe used for reporting to the concerned authorities regarding the same. The tool can be used for continuous searching of web for PII and reporting it to the concerned authorities.

The tool also categorizes the URLs on the basis of the protection criteria (Login, OTP, CAPTCHA) they are using and whether the data is completely or partially available in the concerned websites.

We also request the people not to share their personally identifiable information (PII). Once it goes into the wrong hands there can be a lot of problems. Our government is so concerned with its public privacy and made many efforts to make many laws to stop such wrong happenings. One of such effort is Information Technology Amendment Act 2008.

### **Future Scope**

This type of system can be used by concerned Government organization for continuously searching web for our Personally Identifiable Information based on the database they have so that any leakage can be detected and removed from the sites as soon as possible and necessary actions can be taken against the offenders of the Law.

The above all process can be automated and artificial intelligence can also be implemented so that the machine learn itself from the previous search results.



## **BIBLIOGRAPHY**

## 12. BIBLIOGRAPHY:

- Inukollu, V. N., Keshamoni, D. D., Kang, T., & Inukollu, M., Factors Influencing Quality of Mobile Apps: Role of Mobile App Development. International Journal of Software Engineering & Applications (IJSEA). 5(5): 15-34.
- Lianne Caetano, Apps Tracking Your Location: Friendly or Creepy? In McAfee Blog, 2013
- Statista. Forecast of mobile phone users worldwide.
- Statista. Number of apps available in leading app stores as of July 2015.
- Nielsen. So Many Apps, So Much More Time for Entertainment, <http://www.nielsen.com>
- Survey of Canadians on Privacy-Related Issues In Office of the privacy commissioner of Canada, 2013
- Marble security. Mobile apps expose enterprises to loss of sensitive data.
- Zhang, Y., Yang, M., Xu, B., Yang, Z., Gu, G., Ning, P., & Zang, B "Vetting undesirable behaviors in android apps with permission use analysis." Proceedings of the ACM SIGSAC conference on Computer & communications security. ACM, 2013.
- Vera code. Static Code Analysis, by Neil DuPaul
- TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones
- Zang J, Dummit K, Graves J, Lisker P, & Sweeney L. Who Knows What About Me? A Survey of Behind the Scenes Personal Data Sharing to Third Parties by Mobile Apps. Technology Science. October 30, 2015.
- Erika McCallister, Guide To Protecting Personally Identifiable Information, ITL Bulletin For April 2010
- Lyndon Cerejo, Lessons Learned From An App Graveyard In Smashing magazine, 2013
- Thurm S, Kane Y. Your Apps Are Watching You. The Wall Street Journal. December 18, 2010
- <https://www.ics.uci.edu/~pattis/common/handouts/pythoneclipsejava/python.html>
- [Stackoverflow.io](http://Stackoverflow.io)
- [Wikipedia.com/python](http://Wikipedia.com/python)
- [www.mongodb.com](http://www.mongodb.com)
- [Youtube.com](http://Youtube.com)