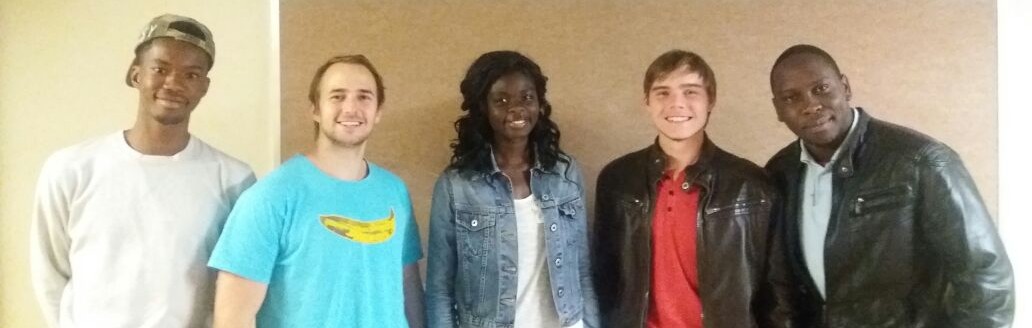
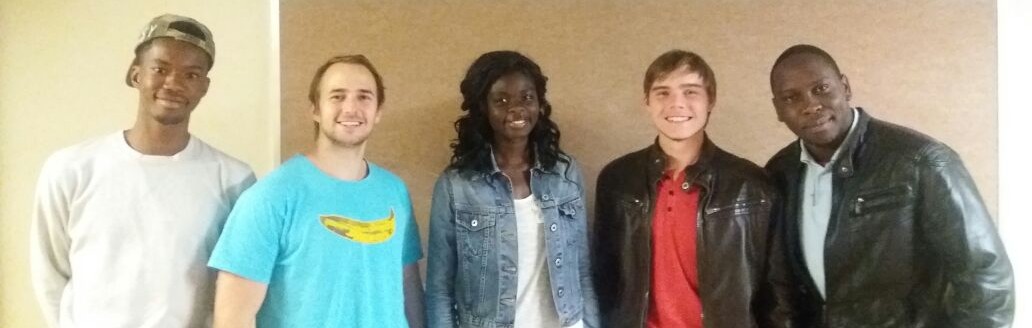


The Team:



**From left to right:**

* **Daniel Malangu**

****

Daniel Malangu. I'm a final year Bsc IT at University of Pretoria and a Software Engineer intern at advance. I can code In various languages such as C#, C++, C, Java, Angular, TypeScript, Javascript and php. I'm also a huge sports fan and have a interest in music

* **Tim Kirker:**



I am currently finishing off my BSc IT and Enterprises at the University of Pretoria. This has allowed me to develop not only my logic, analysis and programming skills but has also allowed me to strengthen my business management skills as well. I have done part time front end web development for a few different businesses as well as have competed in the discovery hackathon. I would consider some of my best traits as being charismatic and friendly. Outside of IT music is my passion.\

* **Eunice Hammond:**

****

A final Year BSc CS Student at the University of Pretoria with efficient skills in various programming languages (C++, Java) , web development( HTML, Bootstrap) and others(SQL, Android, MEAN stack). I am also drawn towards creative writing, enjoy photography and am keen to learning new things in these fields.

* **Heinrich Burgers:**

****

I am currently a final year Computer Science student at the University of Pretoria. I find AI fascinating as well as Computer Security, Networks and Software Engineering. I am currently working part time at Burgers Management Consulting as an intern and as a contract programmer where I work in a variety of fields and platforms. These include Java, SQL, Zizo, ASP.NET, MEAN Stack and LAMP Stack. I also have a lot of experience working in multicultural and international environments.

* **Jocelyn Bondjobo:**

Experienced Junior Software Developer with a demonstrated history of working in the computer software industry. Skilled in Microsoft applications and several programming languages such as Python, PHP, HTML5, JS, Java, Ruby and C++.

Goal:

The goal of our system is to allow people and companies in the pharmaceutical industry to easily find the products they would like to order. These products are all stored by different wholesalers under different names. The different clients also search for the same products using different search terms.

Solution:

Our system allows the large amounts of data to be processed at a very high speed. Using our system, the user can link related terms and word together. By doing so the user can search for a term using custom, user specified terms.

Technologies

Zizo:

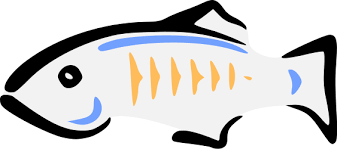
For our system, we used zizo as a database system. Zizo is a pattern database that was mainly developed for Big Data analysis, data warehousing and reporting. Zizo automatically transforms large volumes of data into a lossless set of patterns. It allows querying large data sets at fast speeds and low-cost server infrastructure. Zizo identifies patterns in the data set and stores it once for every pattern.

* Compresses data
* Fast queries
* Works well with large amounts of data.



GlassFish:

Zizo communicates to the rest of the system using SOAP calls and REST calls. These services can be hosted from the using GlassFish. GlassFish allows us to use multiple clients and does memory management.



JMeter:

We used JMeter for benchmark testing. In this we compared our system to the old system as well as set standards.



Junit:

We used Junit for unit testing of our system. We tested every function of our code as well as the system as a whole.



Java:

Java acts as our user interface and connects the user with database. Java receives the zizo service and presents the response to the user. Java performs some of the business logic and memory management.

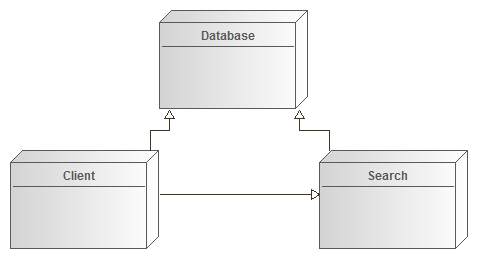


Architecture

**Service Oriented:**

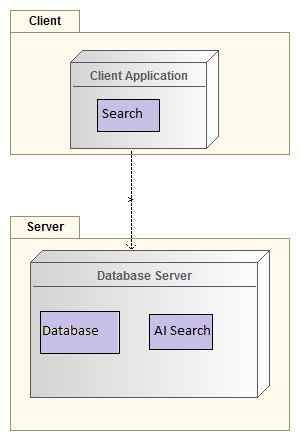
This architecture would work well for our project as our project fits within to the four properties of Service Oriented Architecture.

Namely:

* It logically represents a business activity with a specified outcome.
* It is self-contained.
* It is a black box for its consumers.
* It may consist of other underlying services.

**Client Server:**

This architecture complements our project quite well as it is a network architecture where each computer or process is either a client or a server. In our case our database will act as our server and the requests being made to the server will come from service requesters called clients. The server will share its resources with the clients.



Design Patterns

**Composite:**

The Composite pattern fits in with our project as the intent of a composite is to "compose" objects into tree structures to represent part-whole hierarchies. Implementing the composite pattern lets clients treat individual objects and compositions uniformly

**Strategy:**

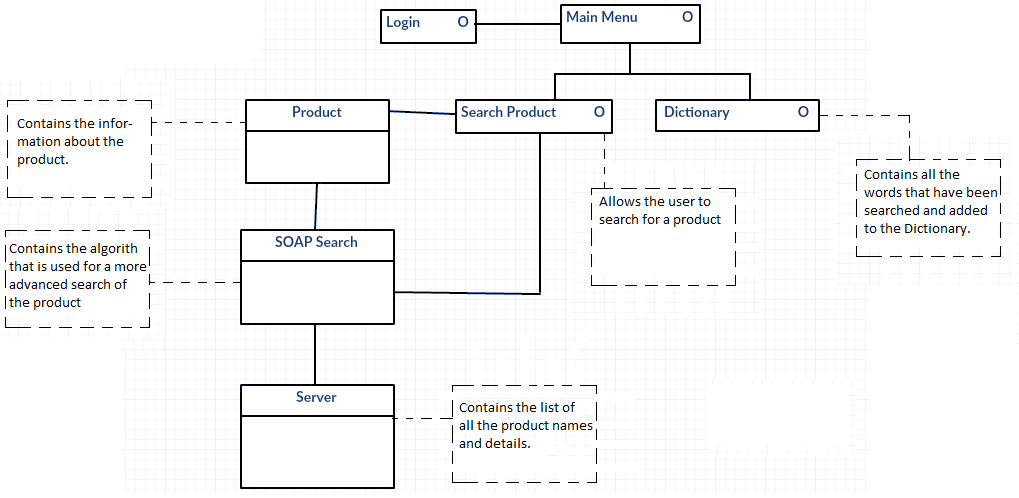
This pattern could be extremely beneficial to our project as it is a behavioural software design pattern that enables to select an algorithm at runtime. The properties of the strategy pattern can be applied to our search algorithms as well as our machine learning algorithms.

Its properties are:

* Defines a family of algorithms.
* Encapsulates each algorithm.
* Makes the algorithms interchangeable within that family.

The essential requirement to implement the Strategy pattern is the ability to store a reference to some code in a data structure and retrieve it.

**Façade:**

This pattern could be a potential solution to our project as it is an object that provides a simplified interface to a larger body of code, such as a class library. This pattern hides the complexities of the larger system and provides a simpler interface to the client.