**Django E-commerce website Project**

**Abstract**

Online Shopping play a great importance in the modern business environment. La BELLE Fashion Store has opened the door of opportunity and advantage to the firms. This paper analysed the different issue of online shopping. The research aims to provide theoretical contribution in understanding the present status of online shopping. The Study Discuss the consumers’ online shopping behaviours. Paper also identify the problems face by the consumers when they want to accept internet shopping. Present paper is an expressive study based on the detailed review of earlier pertinent studies related to the various concepts of online shopping to discover the concept of online shopping. Solitude and safety risk emerge regularly as a reason for being cautious about internet shopping. Shopping convenience, information seeking, social contact, and diversity affects the consumer attitude towards online shopping. The impossibility of product testing, problems with complaints, product return and missus of personal data are the main doubts regarding on-line shopping.

**INTRODUCTION**

An E-commerce website requires appropriate strategy of successful design and implementation. Everything is required to plan from scratch to end of website. The e-commerce sector is seen the exponential growth thus a new option will easily part of this regatta of commercial website. The e-commerce website will feature the online shopping facility of various fashion products under a single web space. The proposed web application will allow business personnel to make their total business using it and increase their reachability thousands of times more than today they have, over the internet. It will allow multiple shopping vendors to sale their products online. The product management in the system will be done in the form of categories. The safety of information is the main requirement of the system and will be handling according to that. To formulate this project first task is to do is cost estimation. For probabilistic assessment of the project cost estimation is required. Cost estimation covers the accurate; estimations of cost and effort required for the project. As a project manager and developer as well, it’s is estimates are defined to early stage in the project. Cost estimation in application development project includes the set of procedures and techniques that will be utilized, required to produce by organisation for development (Alex,2013). The available resources of a company are also affecting the cost estimation. It will be very complex project. To demonstrate knowledge learnt in class, tech communities and online materials, I will undertake the entire project alone even though it requires a team of 6 or more. It will take time of 3months to get the shape or get the basic structure. The environment variants depend on the further requirements of the ecommerce web application.

**AIMS AND OBJECTIVES**

The main objective of the study is to develop an online fashion brochure system. The system aims to achieve the following objectives:

• To design an online fashion system.

• To provides a solution to reduce and optimize the expenses of customer order management .

• To create an avenue where people can shop for fashion products online.

• To develop a database to store information on fashion products and services.

**SCOPE AND LIMITATION**

Every project is done to achieve a set of goals with some conditions keeping in mind that it should be easy to use, feasible and user friendly. As the goal of this project is to develop an online fashion brochure system, this system will be designed keeping in mind the conditions (easy to use, feasibility and user friendly) stated above. It may help in effective and efficient order management. In every shot time, the collection will be obvious, simple and sensible. It is very possible to observe the customer potentials and purchase patterns because all the ordering history is store in the database. It is efficient managing all the operations of an online store within a single platform. The project aims to automate the business process of La BELLE Fashion store. The proposed project would cover:

**Customer Side**

• Customer can view/search products without login.

• Customer can also add/remove product to cart without login (if customer try to add same product in cart. It will add only one)

• When customer try to purchase product, then he/she must login to system.

• After creating account and login to system, he/she can place order.

• If customer click on pay button, then their payment will be successful and their order will be placed.

• Customer can check their ordered details by clicking on orders button.

• Customer can see the order status (Pending, Confirmed, Delivered) for each order

• Customer can Download their order invoice for each order

• Customer can send feedback to admin (without login).

**Administrator Side**

• Admin can provide username, email, password and your admin account will be created.

• After login, there is a dashboard where admin can see how many customers is registered, how many products are there for sale, how many orders placed.

• Admin can add/delete/view/edit the products.

• Admin can view/edit/delete customer details.

• Admin can view/delete orders.

• Admin can change status of order (order is pending, confirmed, out for delivery, delivered)

• Admin can view the feedbacks sent by customers.

**LITERATURE REVIEW**

My first source (“The Myth of Secure E-Shopping”) has proposed, even though online merchants have tried their best to beef up the security, threats and attacks still prevail. For this reason, consumer should act fast to protect their privacy when shopping online.

My second source (“10 Things Your Mother Never Told You About Online Shopping”) explains many ways that consumers could do to enhance the privacy and security aspect apart from what online merchants have done for the same reason. Taking all these contents as a whole, I would say that in any situation, people can still shop online safely provided they understand the reality and take some precautions above all.

**IMPORTANCE OF ONLINE**

Consumers can purchase any goods and services anytime at everywhere. Online shopping is user friendly compare to in store shopping because consumers can just complete his requirements just with a click of mouse without leaving their home. Online shopping has some advantages like below

• Save the Time of the consumers.

• They can purchase any time anywhere

• They can compare the price with the others retailers very easily.

• Compare the advertising price and actual price

• They can easily track their product

• They can use cash back policy

• They can purchase the product from the foreign marketers.

PROBLEMS OF ONLINE SHOPPING

Online shopping problems are great barrier to the online purchase aim of customers. General problems include prospect of having credit card. The obscurity to confirm the reliability of the provide goods and the risk to buy a product that it would not value as much as customer pay for it. Aftersales problems, involved difficulty to change not working product with a new one and products warranty are not assured. Online shopping has various disadvantages:

• The customers can not touch and fell of the products when they want to Purchase.

• Some time delivery time is so much late

• Some time they will pay the shipping charges so why the cost of the product may increase.

• Lack of personal attention by the sellers. More chance to fraud.

• Security of internet banking password and credit card password

• Lack of quality.

**Methodology**

This Section describes the methodology applied during the development of la BELLE Fashions store. A methodology is a model, which project managers employ for the design, planning, implementation and achievement of their project objectives. Effective project management is essential in absolutely any organization, regardless of the nature of the business and the scale of the organization. From choosing a project to right through to the end, it is important that the project is carefully and closely managed. Based on the nature of my project solution, it was essential to use incremental Software development life cycle (SDLC). The project typically has a number of Phases and the level of control required over each phase are primarily defined by the nature of the Project, the complexity of the same and the industry to which the Project has to cater to. An Incremental (SDLC) model consists of a number of dependent increments that are completed in a prescribed sequence. Each increment includes a Launching, Monitoring and Controlling, and Closing Process Group for the functions and features in that increment only. Each increment integrates additional parts of the solution until the final increment, where the remaining parts of the solution are integrated.

**functional Requirements**

The following is the desired functionality of the new system. The proposed project would cover:

**Customer Module**

• Customer can view/search products without login.

• Customer can also add/remove product to cart without login (if customer try to add same product in cart. It will add only one)

• When customer try to purchase product, then he/she must login to system.

• After creating account and login to system, he/she can place order.

• If customer click on pay button, then their payment will be successful and their order will be placed.

• Customer can check their ordered details by clicking on orders button.

• Customer can see the order status (Pending, Confirmed, Delivered) for each order .

**Admin Module**

• Admin can provide username, email, password and your admin account will be created.

• After login, there is a dashboard where admin can see how many customers is registered, how many products are there for sale, how many orders placed.

• Admin can add/delete/view/edit the products.

• Admin can view/edit/delete customer details.

• Admin can view/delete orders.

• Admin can change status of order (order is pending, confirmed, out for delivery, delivered).

**Non-functional Requirements**

It specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system.

**• Availability:** The system should remain operational in any day and any place.

• **Accuracy:** There is a need to optimize the system to ensure more accurate results and calculations.

• **Usability:** The system should provide a User-friendly user interface and tooltips to enhance itself and be effectively responsive.

**• Secure:** The system must be able to provide security against any external injections by using a layered security system. Implementation of user login functionalities also ensures the system is secure from unauthorized persons.

• **Performance of the system:** Response time is very good for given piece of work. The system will support multi user environment.

• **Reliability of the system:** The system will be highly reliable and it generates all the updates information in correct order. Data validation and verification is done at every stage of activity. System recovery will also be speed.

**System Specifications**

This section describes the hardware components and software requirements needed for effective and efficient running of the system

**Platform Requirements**

|  |  |  |
| --- | --- | --- |
| Hardware/Software | Hardware / Software element | Specification /version |
| Hardware | RAM | 2GB |
| Hard Disk | 500GB |
| Software | OS | Windows,Linux.  Jupyter NoteBook.  Python 3. |
| Python IDE  Microsoft Azure |

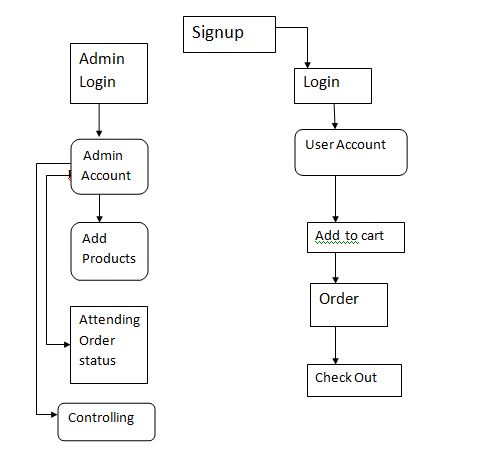
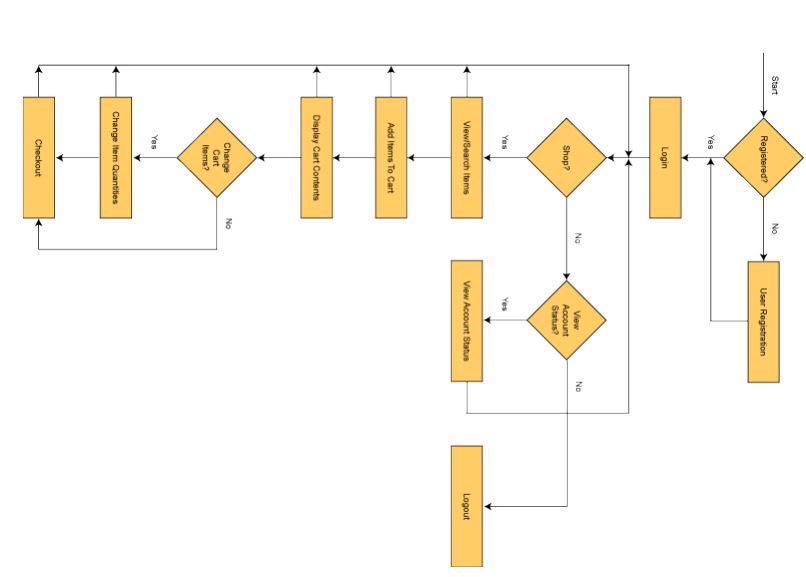
**Sytem Design:  
**

Fig show Architecture

**Data flow**



**Flow Chart**

****

**Result screenshots**

Add Result screenshots yourself.

**Structure of the Project Files**

Django is a Model View Template (MVT) framework

• Mange.py -This file is used basically as a command-line utility and for deploying, debugging, or running our web application. It contains code for run-server, or make migrations or migrations, etc. that we use in the shell. Anyway, we do not need to make any changes to the file.

• Sql is the database file.

• Static folder contains all the static files like CSS and images.

• Env folder is the project specific development environment. Its created through a command ‘virtualenev Env’

• Ecomm folder is the Django projects consisting of init, admin, models, views, apps and forms.

•Models are basically the blueprints of the database we are using and hence contain the information regarding attributes and the fields etc of the database.

•Views is a crucial one, it contains all the Views (usually as classes). Views.py can be considered as a file that interacts with the client. Views are a user interface for what we see when we render a Django Web application.

• URLs Just like the project urls.py file, this file handles all the URLs of our web application.

• Ecommerce folder is the Django application directory consisting of the following essential files.

•Init.py This file remains empty and is present them only to tell that this particular directory is a package.

•Settings.py This file is present for adding all the applications and the middleware application present. Also, it has information about templates and databases. Overall, this is the main file of our Django web application.

•Urls.py This file handles all the URLs of our web application. This file has the lists of all the endpoints that we will have for our website.

• Wsgi.py This file mainly concerns with the WSGI server and is used for deploying our applications on to servers like Apache etc.

•Asgi.py In the newer versions of Django, you will also find a file named as asgi.py apart from wsgi.py. ASGI can be considered as a succeeded interface to the WSGI. ASGI, short for Asynchronous Server Gateway interface.

**Add some output Screenshot**

Homepage

Admin Login

Organization Login

Student Login

And others

**DETAILED PROJECT DESIGN**

The html pages are used for the front end of the Django framework. CSS is also used for the purpose of designing.

**DJANGO FRAMEWORK:**

Django is a free and open source web application framework, written in Python. A web framework is a set of components that helps you to develop websites faster and easier.

When you're building a website, you always need a similar set of components: a way to handle user authentication (signing up, signing in, signing out), a management panel for your website, forms, a way to upload files, etc.

Luckily for you, other people long ago noticed that web developers face similar problems when building a new site, so they teamed up and created frameworks (Django being one of them) that give you ready-made components to use.

Frameworks exist to save you from having to reinvent the wheel and to help alleviate some of the overhead when you’re building a new site.

Django helps you write software that is:

**Complete**

Django follows the "Batteries included" philosophy and provides almost everything developers might want to do "out of the box". Because everything you need is part of the one "product", it all works seamlessly together, follows consistent design principles, and has extensive and [up-to-date documentation](https://docs.djangoproject.com/en/stable/).

**Versatile**

Django can be (and has been) used to build almost any type of website — from content management systems and wikis, through to social networks and news sites. It can work with any client-side framework, and can deliver content in almost any format (including HTML, RSS feeds, JSON, XML, etc). The site you are currently reading is built with Django!  
  
Internally, while it provides choices for almost any functionality you might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed.

**Secure**

Django helps developers avoid many common security mistakes by providing a framework that has been engineered to "do the right things" to protect the website automatically. For example, Django provides a secure way to manage user accounts and passwords, avoiding common mistakes like putting session information in cookies where it is vulnerable (instead cookies just contain a key, and the actual data is stored in the database) or directly storing passwords rather than a password hash.  
  
*A password hash is a fixed-length value created by sending the password through a*[*cryptographic hash function*](https://en.wikipedia.org/wiki/Cryptographic_hash_function)*. Django can check if an entered password is correct by running it through the hash function and comparing the output to the stored hash value. However due to the "one-way" nature of the function, even if a stored hash value is compromised it is hard for an attacker to work out the original password.*  
  
Django enables protection against many vulnerabilities by default, including SQL injection, cross-site scripting, cross-site request forgery and clickjacking.

**Scalable**

Django uses a component-based “[shared-nothing](https://en.wikipedia.org/wiki/Shared_nothing_architecture)” architecture (each part of the architecture is independent of the others, and can hence be replaced or changed if needed). Having a clear separation between the different parts means that it can scale for increased traffic by adding hardware at any level: caching servers, database servers, or application servers. Some of the busiest sites have successfully scaled Django to meet their demands.

**Maintainable**

Django code is written using design principles and patterns that encourage the creation of maintainable and reusable code. In particular, it makes use of the Don't Repeat Yourself (DRY) principle so there is no unnecessary duplication, reducing the amount of code. Django also promotes the grouping of related functionality into reusable "applications" and, at a lower level, groups related code into modules (along the lines of the [Model View Controller (MVC)](https://developer.mozilla.org/en-US/docs/Glossary/MVC) pattern).

**Portable**

Django is written in Python, which runs on many platforms. That means that you are not tied to any particular server platform, and can run your applications on many flavors of Linux, Windows, and Mac OS X. Furthermore, Django is well-supported by many web hosting providers, who often provide specific infrastructure and documentation for hosting Django sites.



* **URLs:**While it is possible to process requests from every single URL via a single function, it is much more maintainable to write a separate view function to handle each resource. A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also match particular patterns of strings or digits that appear in a URL and pass these to a view function as data.
* **View:** A view is a request handler function, which receives HTTP requests and returns HTTP responses. Views access the data needed to satisfy requests via *models*, and delegate the formatting of the response to *templates*.
* **Models:** Models are Python objects that define the structure of an application's data, and provide mechanisms to manage (add, modify, delete) and query records in the database.
* **Templates:** A template is a text file defining the structure or layout of a file (such as an HTML page), with placeholders used to represent actual content. A *view* can dynamically create an HTML page using an HTML template, populating it with data from a *model*. A template can be used to define the structure of any type of file; it doesn't have to be HTML!
* **Forms**: HTML Forms are used to collect user data for processing on the server. Django simplifies form creation, validation, and processing.
* **User authentication and permissions**: Django includes a robust user authentication and permission system that has been built with security in mind.
* **Caching**: Creating content dynamically is much more computationally intensive (and slow) than serving static content. Django provides flexible caching so that you can store all or part of a rendered page so that it doesn't get re-rendered except when necessary.
* **Administration site**: The Django administration site is included by default when you create an app using the basic skeleton. It makes it trivially easy to provide an admin page for site administrators to create, edit, and view any data models in your site.
* **Serialising data**: Django makes it easy to serialise and serve your data as XML or JSON. This can be useful when creating a web service (a website that purely serves data to be consumed by other applications or sites, and doesn't display anything itself), or when creating a website in which the client-side code handles all the rendering of data.

## Why do you need a framework?

To understand what Django is actually for, we need to take a closer look at the servers. The first thing is that the server needs to know that you want it to serve you a web page.

Imagine a mailbox (port) which is monitored for incoming letters (requests). This is done by a web server. The web server reads the letter and then sends a response with a webpage. But when you want to send something, you need to have some content. And Django is something that helps you create the content.

## What happens when someone requests a website from your server?

When a request comes to a web server, it's passed to Django which tries to figure out what is actually requested. It takes a web page address first and tries to figure out what to do. This part is done by Django's **urlresolver** (note that a website address is called a URL – Uniform Resource Locator – so the name urlresolver makes sense). It is not very smart – it takes a list of patterns and tries to match the URL. Django checks patterns from top to bottom and if something is matched, then Django passes the request to the associated function (which is called view).

Imagine a mail carrier with a letter. She is walking down the street and checks each house number against the one on the letter. If it matches, she puts the letter there. This is how the urlresolver works!

In the view function, all the interesting things are done: we can look at a database to look for some information. Maybe the user asked to change something in the data? Like a letter saying, "Please change the description of my job." The view can check if you are allowed to do that, then update the job description for you and send back a message: "Done!" Then the view generates a response and Django can send it to the user's web browser.

**HTML:**

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively **easy to learn**, with the basics being accessible to most people in one sitting; and quite **powerful** in what it allows you to create. It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience under the direction of the [**» W3C**](http://www.w3.org/), the organization charged with designing and maintaining the language.

The definition of HTML is **HyperText Markup Language**.

* *HyperText* is the method by which you move around on the web — by clicking on special text called **hyperlinks** which bring you to the next page. The fact that it is *hyper* just means it is not linear — i.e. you can go to any place on the Internet whenever you want by clicking on links — there is no set order to do things in.
* *Markup* is what **HTML tags** do to the text inside them. They mark it as a certain type of text (*italicised* text, for example).
* HTML is a *Language*, as it has code-words and syntax like any other language.

HTML consists of a series of short **codes** typed into a text-file by the site author — these are the tags. The text is then **saved as a html file**, and **viewed through a**[browser](https://www.yourhtmlsource.com/starthere/glossary.html#browser), like Internet Explorer or Netscape Navigator. This browser reads the file and translates the text into a visible form, hopefully rendering the page as the author had intended. Writing your own HTML entails using tags correctly to create your vision. You can use anything from a rudimentary text-editor to a powerful graphical editor to create HTML pages.

**CSS:**

**C**ascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

Advantages of CSS

* **CSS saves time** − You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
* **Pages load faster** − If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
* **Easy maintenance** − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
* **Superior styles to HTML** − CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
* **Multiple Device Compatibility** − Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
* **Global web standards** − Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

**SYSTEM REQUIREMENT SPECIFICATION**

**Functional Requirements**

The particular necessities are user interfaces. The outside clients are the customers. Every one of the customers can utilize this product for ordering and looking.

* Hardware Interfaces: The outside equipment interface utilized for ordering and looking is PCs of the customers. The PC's might be portable PCs with remote LAN as the web association gave will be remote.
* Software Interfaces: The working Frameworks can be any rendition of windows.
* Performance Prerequisites: The PC's utilized must be at least Pentium 4 machine with the goal that they can give ideal execution of the item.

**Non-Functional Requirements**

Non utilitarian necessities are the capacities offered by the framework. It incorporates time imperative and requirement on the advancement procedure and models. The non-useful prerequisites are as per the following:

* Speed: The framework ought to prepare the given contribution to yield inside fitting time.
* Ease of utilization: The product thought to be easy to understand. At that point the clients can utilize effortlessly, so it doesn't require much preparing time.
* Reliability: The rate of disappointments ought to be less than just the framework is more solid.
* Portability: It thought to be anything but difficult to actualize in any framework

**Python**

1. Python is a general-purpose high-level programming Language (human understandable languages are High level programming languages)
2. Python Developed by Guido Van Rossum
3. 1989 National Research Institute (NRI) At Netherland
4. Officially Python available to the public in 1991: FEB 20th 1991

Python was imagined in the late 1980s,[29] and its usage started in December 1989[30] by Guido van Rossum at Centrum Wickenden and Informatica (CWI) in the Netherlands as a successor to the ABC dialect (itself roused by SETL)[31]capable of exemption dealing with and interfacing with the Amoeba working system.[6] Van Rossum remains Python's chief creator. His proceeding with focal part in Python's advancement is reflected in the title given him by the Python people group:

**Python Feature**

1. Simple and easy to learn

Python as only 33 keywords but JAVA as (53) keywords

1. Free ware (There is no license we cannot pay anything)

& Open source (we can able to see source code if source is not good I can able to customize our requirements)

1. High level programming language (human understandable language)

Python Is Platform Independent (It means I can write a program once and run anywhere (WORA)

1. Portability

Moving python program from one platform to another platform without changing any thing

1. Dynamically Typed Programming Language

In python we are not required to declare type in Python

1. Both Object Oriented and Procedure Oriented Language
2. Interpreted Language

It means we are not going to compile

1. Extensible

We can use Other Programming Language in Python

**Limitations of python**

1. Performance wise it is not up to the mark Because it’s an interpreted language

Interpreter able to see only one line

(JAVA is better performance compare to python in java JIT (just intime compiler) concept is there

1. Mobile applications it is not up to the mark

Myth: -python is not suitable large-scale enterprise applications

**Flavors of python**

1. Cpython: - It can be standard, it can be used to c language python
2. Jpython or jpython :- it is for JAVA application
3. Iron python: -to work with Microsoft .net platform
4. Pypy :-Internally JIT (just intime compiler) compiler is there so performance wise too good
5. Ruby python: - used for ruby application
6. Anaconda python: - To handle Big-data happily go for Anaconda python
7. Stackless (python for concurrency): -
8. parallelly you execute (like multithread) so go for stackless

**Installation**

You can install software for machine learning in any of the two methods as discussed here:

Download and install Python separately from **python.org** on various operating systems as explained below:

To install Python after downloading, double click the **.exe** (for Windows) or .**pkg** (for Mac) file and follow the instructions on the screen.

For Linux OS, check if Python is already installed by using the following command at the prompt:

$ python --version. ...

If Python 2.7 or later is not installed, install Python with the distribution's package manager. Note that the command and package name vary.

On Debian derivatives such as Ubuntu, you can use **apt**:

$ sudo apt-get install python3

Now, open the command prompt and run the following command to verify that Python is installed correctly:

$ python3 --version

Python 3.8

**Applications of Python**

**1. GUI-Based Desktop Applications:**

Python has simple syntax, modular architecture, rich text processing tools and the ability to work on multiple operating systems which make it a desirable choice for developing desktop-based applications. There are various GUI toolkits like wxPython, PyQt or PyGtk available which help developers create highly functional Graphical User Interface (GUI). The various applications developed using Python includes:

1. **Image Processing and Graphic Design Applications:**

Python has been used to make 2D imaging software such as Inkscape, GIMP, Paint Shop Pro and Scribus. Further, 3D animation packages, like Blender, 3ds Max, Cinema 4D, Houdini, Lightwave and Maya, also use Python in variable proportions.

1. **Scientific and Computational Applications:**

The higher speeds, productivity and availability of tools, such as Scientific Python and Numeric Python, have resulted in Python becoming an integral part of applications involved in computation and processing of scientific data. 3D modeling software, such as FreeCAD, and finite element method software, such as Abaqus, are coded in Python.

1. **Games:**

Python has various modules, libraries and platforms that support development of games. For example, PySoy is a 3D game engine supporting Python 3, and PyGame provides functionality and a library for game development. There have been numerous games built using Python including Civilization-IV, Disney’s Toontown Online, Vega Strike etc.

**2. Web Frameworks and Web Applications:**

Python has been used to create a variety of web-frameworks including CherryPy, Django, TurboGears, Bottle, Flask etc. These frameworks provide standard libraries and modules which simplify tasks related to content management, interaction with database and interfacing with different internet protocols such as HTTP, SMTP, XML-RPC, FTP and POP. Plone, a content management system; ERP5, an open source ERP which is used in aerospace, apparel and banking; Odoo – a consolidated suite of business applications; and Google App engine are a few of the popular web applications based on Python.

**3. Enterprise and Business Applications:**

With features that include special libraries, extensibility, scalability and easily readable syntax, Python is a suitable coding language for customizing larger applications. Reddit, which was originally written in Common Lips, was rewritten in Python in 2005. Python also contributed in a large part to functionality in YouTube.

**4. Operating Systems:**

Python is often an integral part of Linux distributions. For instance, Ubuntu’s Ubiquity Installer, and Fedora’s and Red Hat Enterprise Linux’s Anaconda Installer are written in Python. Gentoo Linux makes use of Python for Portage, its package management system.

**5. Language Development:**

Python’s design and module architecture has influenced development of numerous languages. Boo language uses an object model, syntax and indentation, similar to Python. Further, syntax of languages like Apple’s Swift, CoffeeScript, Cobra, and OCaml all share similarity with Python.

**6. Prototyping:**

Besides being quick and easy to learn, Python also has the open source advantage of being free with the support of a large community. This makes it the preferred choice for prototype development. Further, the agility, extensibility and scalability and ease of refactoring code associated with Python allow faster development from initial prototype. Since its origin in 1989, Python has grown to become part of a plethora of web-based, desktop-based, graphic design, scientific, and computational applications. With Python available for Windows, Mac OS X and Linux / UNIX, it offers ease of development for enterprises. Additionally, the latest release Python 3.4.3 builds on the existing strengths of the language, with drastic improvement in Unicode support, among other new features.

**Versions of python**

1. Python 1.0 Introduced in Jan 1994
2. Python 2.0 Introduced in oct 2000
3. Python 3.0 introduced in Dec 2008

latest version

python 3.8

Any new version should provide support for old version programs

1. There is no- backward compatibility support

Python 3 is not support to python 2 program

**CONCLUSION**

The project entitled La BELLE Fashion Store system was completed successfully. The system has been developed with much care and free of errors and at the same time it is efficient and less time consuming. The purpose of this project was to develop a web application for purchasing items from a fashion shop. This project enabled me gain valuable information and practical knowledge on several topics like designing web pages using html & CSS, usage of responsive templates, designing of full stack Django application, and management of database using SQL. The entire system is secured. Also, the project helped me understanding about the development phases of a project and software development life cycle. I learned how to test different features of a project. This project has given me great satisfaction in having designed an application which can be implemented to any nearby shops or branded shops selling various kinds of products by simple modifications. However, it was very challenging learning and developing an application using a new technology.

**REFERENCES**

Fernandez, A.& Anthony D. Miyazaki. Consumer Perceptions of Privacy and Security Risks for Online Shopping. The Journal of Consumer Affairs 35.1: 27-44.

Kandra, A. (2001). The Myth of Secure E-shopping. PC World July: 29-32.

Halpin, M. (2011). 10 Things Your Mother Never Told You About Online Shopping. Yahoo! Internet Life: 61-63.

Wu, J., & Guo, X. (2011). Online booking system design and experimental realization. 2011 International Conference on Electrical and Control Engineering. doi:10.1109/iceceng.2011.6057629

Kumar, V., & Singh, R. (2014). Women Online Shopping: A Critical Review of Literature. SSRN Electronic Journal. doi:10.2139/ssrn.2466824