

Drawing FreeCAD

Submitted for partial fulfilment of the Degree
of
Bachelor of Technology
(Information Technology)



Submitted By:
Amritpal Singh
1411234
146006

Submitted To:

Department of Information Technology
Guru Nanak Dev Engineering College
Ludhiana 141006

Acknowledgement

I, student of Guru Nanak Dev Engineering College, Ludhiana, have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

The author is highly grateful to Dr. M.S. Saini Director, Guru Nanak Dev Engineering College, Ludhiana for providing him with the opportunity to carry out his Six Weeks Training at Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana.

The author would like to whole heartedly thank Dr. H.S. Rai Dean, Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana who is a vast sea of knowledge and without whose constant and never ending support and motivation, it would never have been possible to complete the project and other assignments so efficiently and effectively.

Amritpal Singh

Abstract

CAD development project discuss the work done in computer-aided-design. Computer-aided design (CAD) is the use of computer systems to assist in the creation, modification, analysis, or optimization of a design. I explored FreeCAD's source code. FreeCAD is Free and Open Source CAD Software. FreeCAD is a fully comprehensive 3D CAD application that you can download and install for free. There is a large base of satisfied FreeCAD users worldwide, and it is available in more than 20 languages and for all major operating systems, including Microsoft Windows, Mac OS X and Linux (Debian, Ubuntu, Fedora, Mandriva, Suse ...). FreeCAD is an application for Computer Aided Design (CAD) in three dimensions (3d). With FreeCAD you can create technical drawings such as plans for buildings, interiors, mechanical parts or schematics and diagrams.

This project makes the drawings of different views of the building on the drawing sheets. In this project, the user will be able to enter the specifications of the building through the web browser using Django (Python Web Framework) and on the back-end, the FreeCAD macros will use those input values to draw the drawings of different views of the building on different drawing sheets. The output of the same can then be taken by a user in SVG, PDF and fstd formats.

Also, this project is completely open source and the entire code is available to the user as and when required. There is also Complete developer's Documentation as well as User manual alongwith it that helps using it a lot easier.

1	Introduction To Organisation	1
1.1	Testing and Consutancy Cell	2
2	Introduction To Project	4
2.1	Introduction	4
2.2	Introduction to CAD	4
2.2.1	Introduction to FreeCAD	5
2.2.2	About FreeCAD	6
2.2.3	FreeCAD's features	6
2.2.3.1	Key features	6
2.2.3.2	Genral features	7
2.2.4	Application and User Interface	9
2.2.5	How it started?	9
2.2.6	Downloading Source Code	10
3	Project Work	11
3.1	Introduction	11
3.1.1	Installation Guide	11
3.2	FreeCAD workbenches used in this project	11
3.2.1	Part module	11
3.2.2	Draft module	13
3.2.3	Drawing module	13
3.3	Design implementation using Python scripting language	13
3.3.1	Writing python code	14
3.4	Macros	14
4	Technologies Used	17
4.1	Inkscape	17
4.1.1	About SVG	17
4.1.2	Objectives of the SVG Format	18
4.1.3	Export PDF macro	18
4.1.3.1	What does it do?	19
4.1.3.2	How to use it	19

4.2	Introduction to L ^A T _E X	20
4.2.1	Typesetting	20
4.2.2	Installing L ^A T _E X on System	21
4.2.3	Graphical Editors for L ^A T _E X	22
4.2.4	Pdftscreen L ^A T _E X	23
4.2.5	Web based graphic generation using L ^A T _E X	23
4.3	Introduction to Github	24
4.3.1	What is Git?	25
4.3.2	Installation of Git	26
4.3.3	Various Git Commands	26
4.3.3.1	Create Repositories	26
4.3.3.2	Make Changes	27
4.3.3.3	Group Changes	27
4.3.3.4	Save Fragments	28
4.3.3.5	Synchronize Changes	28
5	Project Legacy	29
5.1	Technical and Managerial Lesson Learnt	29
5.1.1	Ubuntu: An open source OS	29
5.1.2	Internet Relay Chat	30
5.1.3	Shell Scripting	30

1.1	Guru Nanak Dev Engineering College	1
1.2	Testing and Consultancy Cell	2
3.1	Front page	12
3.2	Example of Part shapes in FreeCAD	12
3.3	Example of Drawing module in FreeCAD	13
3.4	Example of python scripting	14
3.5	Example of building macro	15
3.6	Example of drawing macro	15
4.1	Donald Knuth, Inventor Of T _E X typesetting system	20
4.2	L ^A T _E X output of above program.	21
4.3	Texmaker, A Graphical L ^A T _E X Editor	22
4.4	LEd, A Graphical L ^A T _E X Editor	23
4.5	Web based graphic generation using L ^A T _E X(input page)	24
4.6	Github Logo	24
4.7	Git Logo	26

CHAPTER 1

INTRODUCTION TO ORGANISATION



Figure 1.1: Guru Nanak Dev Engineering College

I had my Six Weeks Industrial Training at TCC-Testing And Consultancy Cell, GNDEC Ludhiana. Guru Nanak Dev Engineering College was established by the Nankana Sahib Education Trust Ludhiana. The Nankana Sahib Education Trust i.e NSET was founded in memory of the most sacred temple of Sri Nankana Sahib, birth place of Sri Guru Nanak Dev Ji. With the mission of Removal of Economic Backwardness through Technology Shiromani Gurudwara Parbandhak Committee i.e SGPC started a Poly technical was started in 1953 and Guru Nanak Dev Engineering College was established in 1956.

NSET resolved to uplift Rural areas by admitting 70% of students from these rural areas every year. This commitment was made to nation on 8th April, 1956, the day foundation stone of the college building was laid by Dr. Rajendra Prasad Ji, the First President of India. The College is now ISO 9001:2000 certified.

Guru Nanak Dev Engineering College campus is spread over 88 acres of prime land about 5 Km s from Bus Stand and 8 Km s from Ludhiana Railway Station on Ludhiana-Malerkotla Road. The college campus is well planned with beautifully laid out tree plantation, pathways, flowerbeds besides the well maintained sprawling lawns all around. It has beautiful building for College,Hostels,Swimming Pool,Sports and Gymnasium Hall Complex, Gurudwara Sahib, Bank, Dispensary, Post Office etc. There are two hostels for boys and one for girls with total accommodation of about 550 students. The main goal of this institute is:

- To build and promote teams of experts in the upcoming specialisations.
- To promote quality research and undertake research projects keeping in view their relevance to needs and requirements of technology in local industry.
- To achieve total financial independence.
- To start online transfer of knowledge in appropriate technology by means of establishing multipurpose resource centres.

1.1 Testing and Consutancy Cell

My Six Weeks Institutional Training was done by me at TCC i.e Testing And Consultancy Cell, GNDEC Ludhiana under the guidance of Dr. H.S.Rai Dean Testing and Consultancy Cell. Testing and Consultancy Cell was established in the year 1979 with a basic aim to produce quality service for technical problems at reasonable and affordable rates as a service to society in general and Engineering fraternity in particular.



Figure 1.2: Testing and Consultancy Cell

Consultancy Services are being rendered by various Departments of the College to the industry, Sate Government Departments and Entrepreneurs and are extended in the form of expert advice in design, testing of materials & equipment, technical surveys, technical audit, calibration of instruments, preparation of technical feasibility reports etc. This consultancy cell of the college has given a new dimension to the development programmers of the College. Consultancy projects of

over Rs. one crore are completed by the Consultancy cell during financial year 2009-10.

Ours is a pioneer institute providing Consultancy Services in the States of Punjab, Haryana, Himachal, J&K and Rajasthan. Various Major Clients of the Consultancy Cell are as under:

- Northern Railway, Govt. of India
- Indian Oil Corporation Ltd.
- Larson & Turbo.
- Multi National Companies like AFCON & PAULINGS.
- Punjab Water Supply & Sewage Board
- Power Grid Corporation of India.
- National Building Construction Co.
- Punjab State Electricity Board.
- Punjab Mandi Board.
- Punjab Police Housing Corporation.
- National Fertilizers Ltd.
- GLADA, Ludhiana

2.1 Introduction

The term computer graphics includes almost everything on computers that is not text or sound. Today almost every computer can do some graphics, and people have even come to expect to control their computer through icons and pictures rather than just by typing. Here in our lab at the Program of Computer Graphics, we think of computer graphics as drawing pictures on computers, also called rendering. The pictures can be photographs, drawings, movies, or simulations pictures of things which do not yet exist and maybe could never exist. Or they may be pictures from places we cannot see directly, such as medical images from inside your body. Computer graphics is now used in various fields; for industrial, educational, medical and entertainment purposes. The aim of computer graphics is to visualize real objects and imaginary or other abstract items. In order to visualize various things, many technologies are necessary and they are mainly divided into two types in computer graphics: modeling and rendering technologies.

2.2 Introduction to CAD

Computer-aided design (CAD) is the use of computer systems to assist in the creation, modification, analysis, or optimization of a design. CAD software is used to increase the productivity of the designer, improve the quality of design, improve communications through documentation, and to create a database for manufacturing. CAD is an important industrial art extensively used in many applications, including automotive, shipbuilding, and aerospace industries, industrial and architectural design, prosthetics, and many more. CAD is also widely used to produce computer animation for special effects in movies, advertising and technical manuals. CAD output is often in the form of electronic files for print, machining, or other manufacturing operations. CAD is also used for the accurate creation of photo simulations that are often required in the preparation of Environmental Impact Reports, in which computer-aided designs of intended buildings are superimposed into photographs of existing environments to represent what that locale will be like were the proposed facilities allowed to be built. Computer Aided Drafting describes the process of creating a technical drawings with the use of computer software. CAD software is used to increase the productivity of the designer, improve the quality of design, improve communications

through documentation, and to create a database for manufacturing. CAD output is often in the form of electronic files for print or machining operations. CAD software uses either vector based graphics to depict the objects of traditional drafting, or may also produce raster graphics showing the overall appearance of designed objects.

Today there are very few aspects of our lives not affected by computers. Practically every cash or monetary transaction that takes place daily involves a computer. In many cases, the same is true of computer graphics. Whether you see them on television, in newspapers, in weather reports or while at the doctors surgery, computer images are all around you. A picture is worth a thousand words is a well known saying and highlights the advantages and benefits of the visual ation of our data. We are able to obtain a comprehensive overall view of our data and also study features and areas of particular interest. A range of tools and facilities are available to enable users to visualize their data, and this document provides a brief summary and overview. Computer graphics can be used in many disciplines. Charting, ations, Drawing, Painting and Design, Image Processing and Scientific Visualization are some among them. Computer graphics is concerned with all aspects of producing images using a computer. It concerns with the pictorial synthesis of real or imaginary objects from their computerbased models.

CAD often involves more than just shapes. As in the manual drafting of Technical and Engineering Drawings, the output of CAD must convey information, such as material, processes, dimensions and tolerances, according to applicationspecific conventions. CAD may be used to design curves and figures in twoDimensional(2D) space; or curves, surfaces, and solids in three dimensional(3D) space. CAD is an important industrial art extensively used in many applications, including automotive, shipbuilding, and aerospace industries, industrial and architectural design, prosthetic, and many more. CAD is also widely used to produce Computer animation for special Effects in movies, advertising and technical manuals. The modern ubiquity and power of computers means that even perfume bottles and shampoo dispensers are designed using techniques unheard of by engineers of the 1960s. Because of its enormous economic importance, CAD has been a major driving force for research in computational geometry, computer graphics (both hardware and software), and discrete differential geometry. The design of geometric model for object shapes, in particular, is occasionally called Computer Aided Geometric Design (CAGD). While the goal of automated CAD systems is to increase efficiency, they are not necessarily the best way to allow newcomers to understand the geometrical principles of Solid Modeling.

I explored FreeCAD's source code. FreeCAD is Free and Open Source CAD Software. FreeCAD is a fully comprehensive 3D CAD application that you can download and install for free. There is a large base of satisfied FreeCAD users worldwide, and it is available in more than 20 languages and for all major operating systems, including Microsoft Windows, Mac OS X and Linux (Debian, Ubuntu, Fedora, Mandriva, Suse ...). FreeCAD is an application for Computer Aided Design (CAD) in three dimensions (2d). With FreeCAD you can create technical drawings such as plans for buildings, interiors, mechanical parts or schematics and diagrams.

2.2.1 Introduction to FreeCAD

FreeCAD is a general purpose parametric 3D CAD modeler. The development is completely Open Source (LGPL License). FreeCAD is aimed directly at mechanical engineering and product design

but also fits in a wider range of uses around engineering, such as architecture or other engineering specialties.

FreeCAD features tools similar to Catia, SolidWorks or Solid Edge, and therefore also falls into the category of MCAD, PLM, CAx and CAE. It is a feature based parametric modeler with a modular software architecture which makes it easy to provide additional functionality without modifying the core system.

As with many modern 3D CAD modelers it has many 2D components in order to sketch 2D shapes or extract design details from the 3D model to create 2D production drawings, but direct 2D drawing (like AutoCAD LT) is not the focus, neither are animation or organic shapes (like Maya, 3ds Max, Blender or Cinema 4D), although, thanks to its wide adaptability, FreeCAD might become useful in a much broader area than its current focus.

FreeCAD makes heavy use of all the great open-source libraries that exist out there in the field of Scientific Computing. Among them are OpenCascade, a powerful CAD kernel, Coin3D, an incarnation of Open Inventor, Qt, the world-famous UI framework, and Python, one of the best scripting languages available. FreeCAD itself can also be used as a library by other programs.

FreeCAD is also fully multi-platform, and currently runs flawlessly on Windows and Linux/Unix and Mac OSX systems, with the exact same look and functionality on all platforms.

2.2.2 About FreeCAD

FreeCAD is maintained and developed by a community of enthusiastic developers and users (see the contributors page). They work on FreeCAD voluntarily, in their free time. They cannot guarantee that FreeCAD contains or will contain everything you might wish, but they will usually do their best! The community gathers on the FreeCAD forum, where most of the ideas and decisions are discussed.

2.2.3 FreeCAD's features

2.2.3.1 Key features

- A complete Open CASCADE Technology-based geometry kernel allowing complex 3D operations on complex shape types, with native support for concepts like brep, nurbs curves and surfaces, a wide range of geometric entities, boolean operations and fillets, and built-in support of STEP and IGES formats.
- A full parametric model. All FreeCAD objects are natively parametric, which means their shape can be based on properties or even depend on other objects, all changes being recalculated on demand, and recorded by the undo/redo stack. New object types can be added easily, that can even be fully programmed in Python.
- A modular architecture that allow plugins (modules) to add functionality to the core application. Those extensions can be as complex as whole new applications programmed in C++ or as simple as Python scripts or self-recorded macros. You have complete access from the Python built-in interpreter, macros or external scripts to almost any part of FreeCAD, being geometry creation and transformation, the 2D or 3D representation of that geometry (scenegraph) or even the FreeCAD interface.

- Import/export to standard formats such as STEP, IGES, OBJ, STL, DXF, SVG, STL, DAE, IFC or OFF, NASTRAN, VRML in addition to FreeCAD's native Fcstd file format. The level of compatibility between FreeCAD and a given file format can vary, since it depends on the module that implements it.
- A Sketcher with constraint-solver, allowing to sketch geometry-constrained 2D shapes. The sketcher currently allows you to build several types of constrained geometry, and use them as a base to build other objects throughout FreeCAD.
- A Robot simulation module that allows to study robot movements. The robot module already has an extended graphical interface allowing GUI-only workflow.
- A Drawing sheets module that permit to put 2D views of your 3D models on a sheet. This module then produces ready-to-export SVG or PDF sheets. The module is still sparse but already features a powerful Python functionality.
- A Rendering module that can export 3D objects for rendering with external renderers. Currently only supports povray and LuxRender, but is expected to be extended to other renderers in the future.
- An Architecture module that allows BIM-like workflow, with IFC compatibility. The making of the Arch module is heavily discussed by the community [here](#).

2.2.3.2 Genral features

- **FreeCAD is multi-platform.** It runs and behaves exactly the same way on Windows Linux and Mac OSX platforms.
- **FreeCAD is a full GUI application.** FreeCAD has a complete Graphical User Interface based on the famous Qt framework, with a 3D viewer based on Open Inventor, allowing fast rendering of 3D scenes and a very accessible scene graph representation.
- **FreeCAD also runs as a command line application,** with low memory footprint. In command line mode, FreeCAD runs without its interface, but with all its geometry tools. It can be, for example, used as server to produce content for other applications.
- **FreeCAD can be imported as a Python module,** inside other applications that can run python scripts, or in a python console. Like in console mode, the interface part of FreeCAD is unavailable, but all geometry tools are accessible.
- **Workbench concept:** In the FreeCAD interface, tools are grouped by workbenches. This allows to display only the tools used to accomplish a certain task, keeping the workspace uncluttered and responsive, and the application fast to load.
- **Plugin/Module framework for late loading of features/data-types.** FreeCAD is divided into a core application and modules, that are loaded only when needed. Almost all the tools and geometry types are stored in modules. Modules behave like plugins, and can be added or removed to an existing installation of FreeCAD.

- **Parametric associative document objects:** All objects in a FreeCAD document can be defined by parameters. Those parameters can be modified on the fly, and recomputed anytime. The relationship between objects is also stored, so modifying one object also modifies its dependent objects.
- **Parametric primitive creation** (box, sphere, cylinder, etc).
- Graphical **modification operations** like translation, rotation, scaling, mirroring, offset (trivial or after Jung/Shin/Choi) or shape conversion, in any plane of the 3D space.
- **Boolean operations** (union, difference, intersect)
- Graphical creation of **simple planar geometry** like lines, wires, rectangles, arcs or circles in any plane of the 3D space.
- Modeling with straight or revolution **extrusions, sections** and **fillets**.
- Topological components like **vertices, edges, wires** and **planes** (via python scripting).
- **Testing and repairing** tools for meshes: solid test, non-two-manifolds test, self-intersection test, hole filling and uniform orientation.
- **Annotations** like texts or dimensions
- **Undo/Redo framework:** Everything is undo/redoeable, with access to the undo stack, so multiple steps can be undone at a time.
- **Transaction management:** The undo/redo stack stores document transactions and not single actions, allowing each tool to define exactly what must be undone or redone.
- **Built-in scripting framework:** FreeCAD features a built-in Python interpreter, and an API that covers almost any part of the application, the interface, the geometry and the representation of this geometry in the 3D viewer. The interpreter can run single commands up to complex scripts, in fact entire modules can even be programmed completely in Python.
- **Built-in Python console** with syntax highlighting, autocomplete and class browser: Python commands can be issued directly in FreeCAD and immediately return results, permitting scriptwriters to test functionality on the fly, explore the contents of the modules and easily learn about FreeCAD internals.
- **User interaction mirroring on the console:** Everything the user does in the FreeCAD interface executes python code, which can be printed on the console and recorded in macros.
- **Full macro recording and editing:** The python commands issued when the user manipulates the interface can then be recorded, edited if needed, and saved to be reproduced later.
- **Compound (ZIP based) document save format:** FreeCAD documents saved with .fcstd extension can contain many different types of information, such as geometry, scripts or thumbnail icons. The .fcstd file is itself a zip container, so a saved FreeCAD file has already been compressed.

- **Fully customizable/scriptable Graphical User Interface.** The Qt-based interface of FreeCAD is entirely accessible via the python interpreter. Aside from the simple functions that FreeCAD itself provides to workbenches, the whole Qt framework is accessible too, allowing any operation on the GUI, such as creating, adding, docking, modifying or removing widgets and toolbars.
- **Thumbnailer** (Linux systems only at the moment): The FreeCAD document icons show the contents of the file in most file manager applications such as gnome's nautilus.
- **A modular MSI installer** allows flexible installations on Windows systems. Packages for Ubuntu systems are also maintained.

2.2.4 Application and User Interface

Like almost everything else in FreeCAD, the user interface part (Gui) is separated from the base application part (App). This is also valid for documents. The documents are also made of two parts: the Application document, which contains our objects, and the View document, which contains the representation on screen of our objects.

Think of it as two spaces, where the objects are defined. Their constructive parameters (is it a cube? a cone? which size?) are stored in the Application document, while their graphical representation (is it drawn with black lines? with blue faces?) are stored in the View document. Why is that? Because FreeCAD can also be used WITHOUT graphical interface, for example inside other programs, and we must still be able to manipulate our objects, even if nothing is drawn on the screen.

Another thing that is contained inside the View document are 3D views. One document can have several views opened, so you can inspect your document from several points of view at the same time. Maybe you would want to see a top view and a front view of your work at the same time? Then, you will have two views of the same document, both stored in the View document. Creating new views or closing views can be done from the View menu or by right-clicking on a view tab.

2.2.5 How it started?

FreeCAD's focus is to allow you to make high-precision 3D models, to keep tight control over those models (being able to go back into modelling history and change parameters), and eventually to build those models (via 3D printing, CNC machining or even construction worksite). It is therefore very different from some other 3D applications made for other purposes, such as animation film or gaming. Its learning curve can be steep, specially if this is your first contact with 3D modeling. If you are struck at some point, don't forget that the friendly community of users on the FreeCAD forum might be able to get you out in no time.

The workbench you will start using in FreeCAD depends on the type of job you need to do: If you are going to work on mechanical models, or more generally any small-scale objects, you'll probably want to try the PartDesign Workbench. If you will work in 2D, then switch to the Draft Workbench, or the Sketcher Workbench if you need constraints. If you want to do BIM, launch

the Arch Workbench. If you are working with ship design, there is a special Ship Workbench for you. And if you come from the OpenSCAD world, try the OpenSCAD Workbench.

You can switch workbenches at any time, and also customize your favorite workbench to add tools from other workbenches.

2.2.6 Downloading Source Code

Fired up the terminal because you need to install the qt4 development libraries, tools, compiler and git.

```
git clone https://github.com/FreeCAD/FreeCAD free-cad-code
```

Getting the dependencies:

```
sudo apt install build-essential cmake python python-matplotlib
libtool libcoin80-dev libsoqt4-dev libxerces-c-dev libboost-dev
libboost-filesystem-dev libboost-regex-dev libboost-program-options-dev
libboost-signals-dev libboost-thread-dev libboost-python-dev
libqt4-dev libqt4-opengl-dev qt4-dev-tools python-dev python-pyside
pyside-tools oce-draw libeigen3-dev libqtwebkit-dev libshiboken-dev
libpyside-dev libode-dev swig libzipios++-dev libfreetype6
libfreetype6-dev liboce-foundation-dev liboce-modeling-dev
liboce-ocaf-dev liboce-visualization-dev liboce-ocaf-lite-dev
libsimage-dev checkinstall python-pivy python-qt4 doxygen libspnav-d
```

Compiling: Now first well create a separate directory for storing our build files. This will be out-of-source build. Which means the source code directory will not get modified and there wont be any issue with git.

```
mkdir build
cd build
cmake ../free-cad-code
make
```

Execution: Execute using the following command:

```
./bin/FreeCAD
```


3.1 Introduction

This project makes the drawings of different views of the building on the drawing sheets.

In this project, the user will be able to enter the specifications of the building through the web browser using Django (Python Web Framework) and on the back-end, the FreeCAD macros will use those input values to draw the drawings of different views of the building on different drawing sheets. The output of the same can then be taken by a user in SVG, PDF and fcsd formats.

3.1.1 Installation Guide

To install Drawing FreeCAD project, you need to clone it from github.

- Go to terminal and type

```
git clone https://github.com/amrit3701/Drawing-FreeCAD.git
```
- Now go to the project directory by using

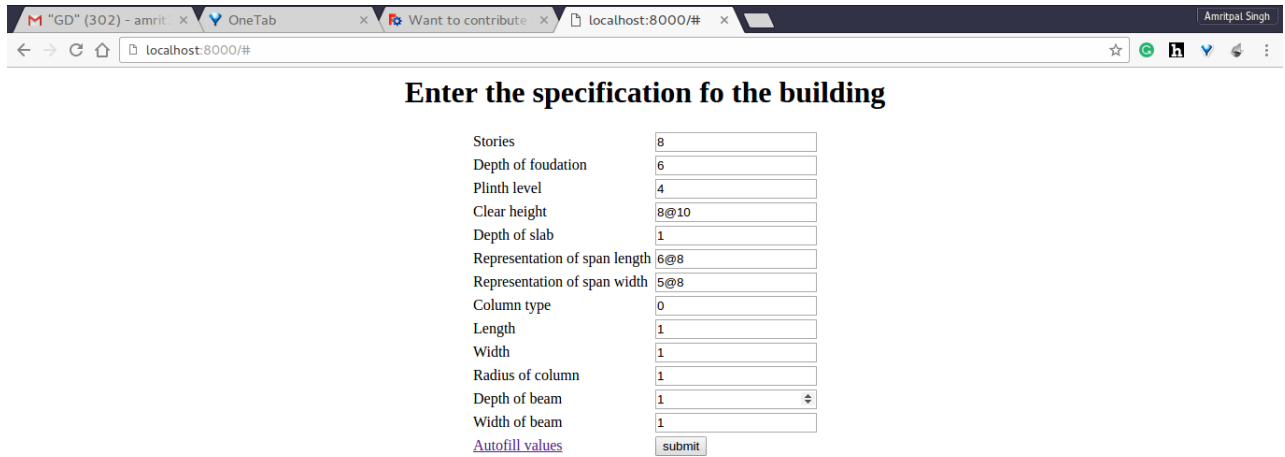
```
cd Drawing-FreeCAD
```
- Now running the below command.

```
python manage.py runserver
```
- Now open *localhost* : 8000 on the web-browser and you will see the below web page.

3.2 FreeCAD workbenches used in this project

3.2.1 Part module

The CAD capabilities of FreeCAD are based on the OpenCasCade kernel. The Part module allows FreeCAD to access and use the OpenCasCade objects and functions. OpenCascade is a professional-level CAD kernel, that features advanced 3D geometry manipulation and objects.



Enter the specification for the building

Stories	8
Depth of foundation	6
Plinth level	4
Clear height	8@10
Depth of slab	1
Representation of span length	6@8
Representation of span width	5@8
Column type	0
Length	1
Width	1
Radius of column	1
Depth of beam	1
Width of beam	1

[Autofill values](#)

Figure 3.1: Front page

The Part objects, unlike Mesh Module objects, are much more complex, and therefore permit much more advanced operations, like coherent boolean operations, modifications history and parametric behaviour.

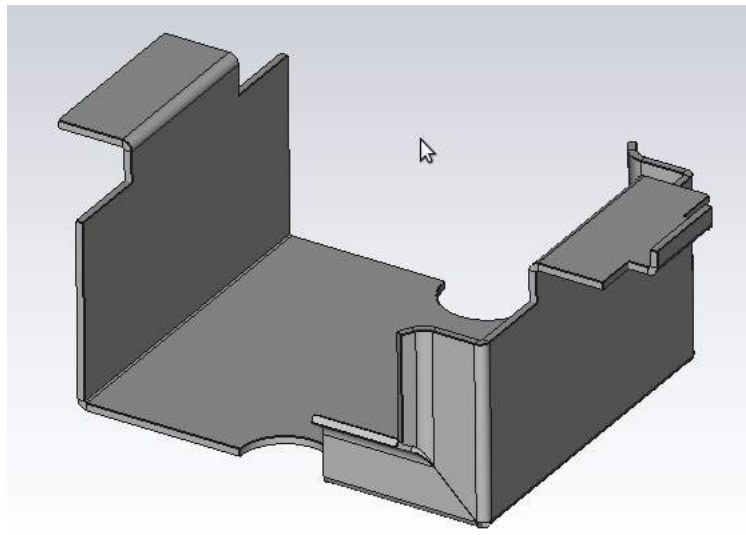


Figure 3.2: Example of Part shapes in FreeCAD

3.2.2 Draft module

The Draft workbench allows to quickly draw simple 2D objects in the current document, and offers several tools to modify them afterwards. Some of these tools also work on all other FreeCAD objects, not only those created with the Draft workbench. It also provides a complete snapping system, and several utilities to manage objects and settings.

3.2.3 Drawing module

The Drawing module allows you to put your 3D work on paper. That is, to put views of your models in a 2D window and to insert that window in a drawing, for example a sheet with a border, a title and your logo and finally print that sheet. The Drawing module is currently under construction and more or less a technology preview!

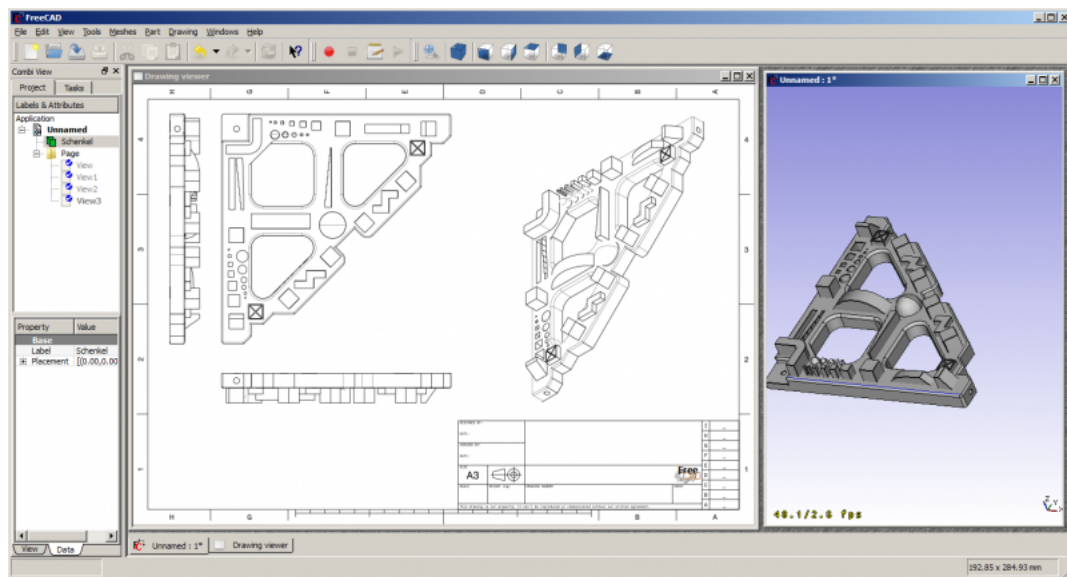


Figure 3.3: Example of Drawing module in FreeCAD

3.3 Design implementation using Python scripting language

Python is a programming language, very simple to use and very fast to learn. It is open-source, multi-platform, and can be used alone for a wide array of things, from programming simple shell scripts to very complex programs. But one of its most widespread uses is as a scripting language, since it is easy to embed in other applications. That's exactly how it is used inside FreeCAD. From the python console, or from your custom scripts, you can pilot FreeCAD, and make it perform very complex actions for which there is still no graphical user interface tool.

For example, from a python script, you can:

- create new objects
- modify existing objects
- modify the 3D representation of those objects

- modify the FreeCAD interface

There are also several different ways to use python in FreeCAD:

- From the FreeCAD python interpreter, where you can issue simple commands like in a "command line"-style interface
- From macros, which are a convenient way to quickly add a missing tool to the FreeCAD interface
- From external scripts, which can be used to program much more complex things. like entire Workbenches.

3.3.1 Writing python code

There are two easy ways to write python code in FreeCAD: From the python console (available from the View -> Panels -> Python console menu) or from the Macro editor (Tools -> Macros). In the console, you write python commands one by one, which are executed when you press return, while the macros can contain a more complex script made of several lines, which is executed only when the macro is executed.

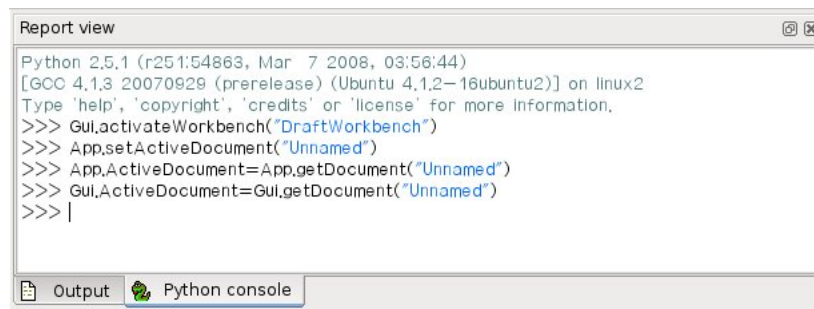


Figure 3.4: Example of python scripting

3.4 Macros

Macros are a convenient way to create complex actions in FreeCAD. You simply record actions as you do them, then save that under a name, and replay them whenever you want. Since macros are in reality a list of python commands, you can also edit them, and create very complex scripts.

In this project I have made many macros which are using for different purposes.

- **Builing macro:** This macro will create the model of building by using the user input in 3D space of FreeCAD. For running this macro in the project I have used the below command.

```
freecadcmd builing_macro.py
```

- **Drawing macro:** This macro will draw the drawing of different views (top, front and side view) of the builing along with sectional view of each building storey on the drawing sheet. (3D model).

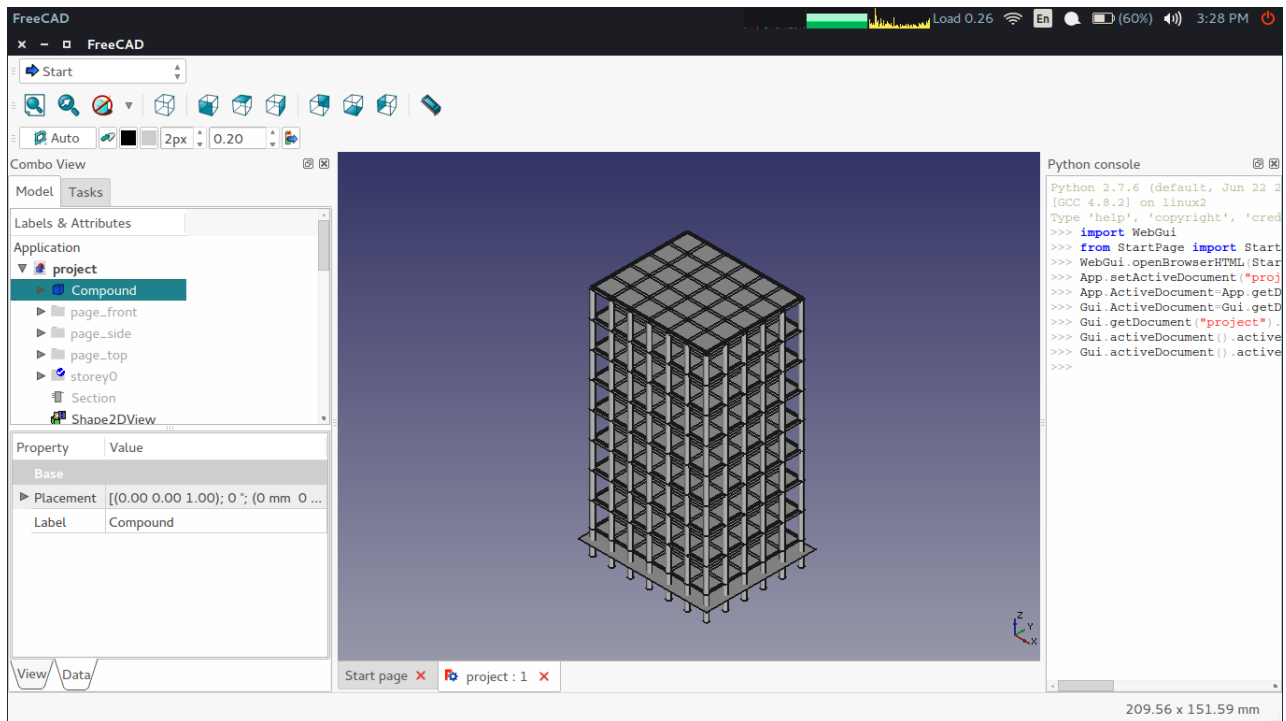


Figure 3.5: Example of building macro

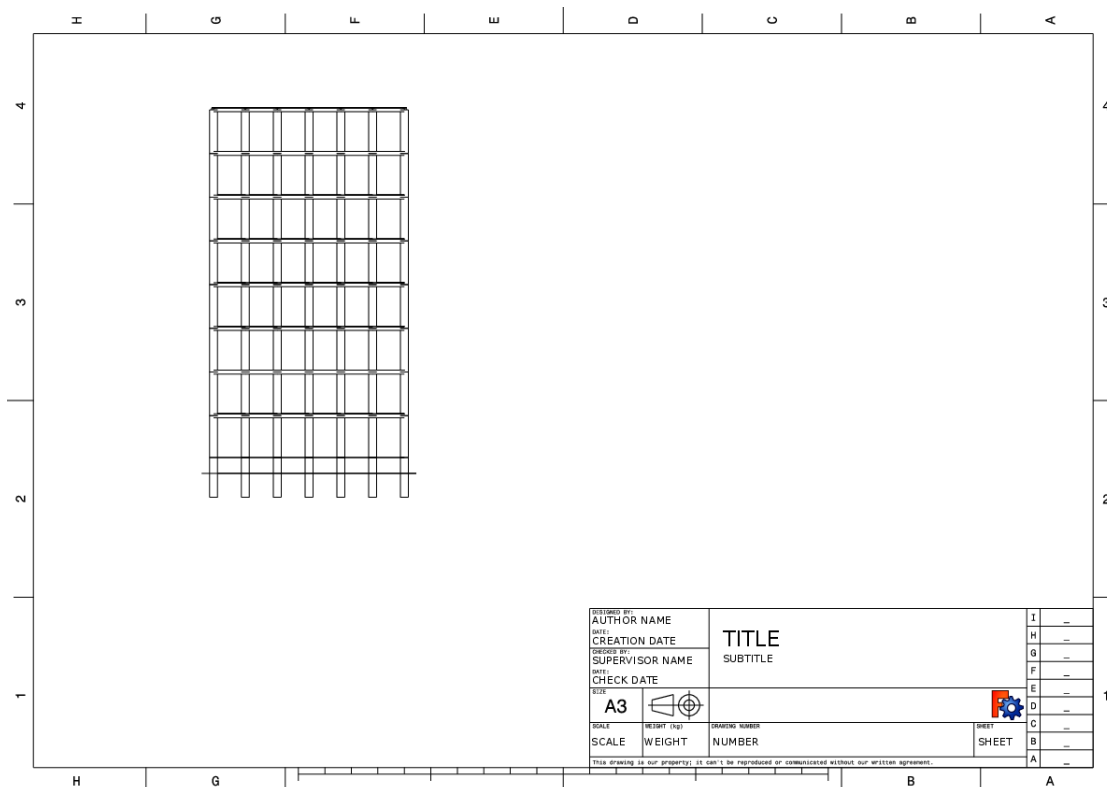


Figure 3.6: Example of drawing macro

```
freecadcmd drawing.py
```

- **Saving drawing macro:** This macro will save the drawing of the building in SVG (Scalable Vector Graphics) and then convert SVG file into PDF (Portable Document Format).

```
freecadcmd save_drawing.py
```

4.1 Inkscape

Inkscape is an open source drawing tool for creating and editing SVG graphics. More than just a text vector editor, Inkscape provides a WYSIWYG interface for manipulation of vector images, allowing the artist to express himself freely. While other free and proprietary software exists with similar capabilities, Inkscape provides an interface to directly manipulate the underlying SVG code, which allows one to be certain that the code is in compliance with W3C standards. Since the beginning of its development, the Inkscape project has been a very active, providing stability for the current software and growth of capacities in the future. Like other drawing programs, Inkscape offers creation of basic shapes (such as ellipses, rectangles, stars, polygons and spirals) as well as the ability to transform and manipulate these basic shapes by rotation, stretching and skewing.

Inkscape also offers functionality to manipulate objects more precisely by adjusting node points and curves. These functions are indispensable to useful drawing software, and allow the advanced artist to freely create what he imagines.

The properties of objects can either be manipulated individually and precisely through the XML editor, or more generally in an intuitive fashion by input devices such as mice, pen tablets or even touch screen. In addition, Inkscape allows one to insert text and bitmaps (such as PNG, another W3C recommended bitmap image format) into an image as well as perform some basic editing functions on them. If further bitmap editing is required, other tools may be used (such as the GIMP) on images before importing them or after. In fact, if a linked bitmap is edited in another program, Inkscape will reflect these changes once the SVG is reloaded.

All of these characteristics make Inkscape a model drawing application, especially considering its flexibility and many other capabilities. Its strict compliance with the W3C SVG standards allow excellent portability of images to many applications and platforms on which these applications are used.

4.1.1 About SVG

Those who work with graphics for internet use are familiar with the problems tied to publication of images on the web. Traditionally, bitmap images (such as JPG or GIF) have been the only

option for use in such documents, with the disadvantage that these images are either too large for quick transfer or, if they are small or highly compressed to reduce file-size, of poor quality.

As a solution to this problem, Macromedia created the Flash image format. While Flash satisfactorily solved the main problems inherent to bitmap images, there has been discontent for some users that the common vector format for the web is dependent solely on Macromedia for development of the file format and software. In order to address this discontent and provide an open option for vector graphics, the W3C created the SVG file format, making a freely usable vector format available to everyone.

Most image files are only able to be read by specific software that renders the image. SVG, however, is described in XML and CSS, and its files can be opened and edited in any ASCII text editor. While it is possible to create SVG images in this manner, it is highly unproductive and unintuitive. SVG editors and renderers have the ability to easily open and manipulate SVG files without a special interpreter.

4.1.2 Objectives of the SVG Format

The advantages of SVG are the same as for any vector image: high-quality images that are smooth and crisp ability to resize the image to any dimensions without diminishing quality, which is impossible with bitmap images. The SVG standard also defines animation, and with a little use of Javascript, one can make SVG interactive. Finally, since SVG is written in XML, it is possible to create graphics based on data that is stored in other XML-based formats, such as graphs, charts and maps. Despite its benefits, there is a lack of usable software to create and edit SVG files and take full advantage of its capacities; for this reason, SVG is not as usable at the moment as Flash.

4.1.3 Export PDF macro

It uses Inkscape to do the SVG to PDF conversion. Customize ToolsBar instructions can be used to add tool button in Drawing WB and therefore to be able to use Export PDF macro with ease when needed.

```
import os
import subprocess
from PySide import QtGui

obj = FreeCADGui.Selection.getSelection()[0]

# Path to Inkscape executable
inkscape_path = "/usr/bin/inkscape"
# Exported PDF file location
file_location = os.path.expanduser("~" + os.sep + obj.Label + ".pdf")

# -----
try:
    page = getattr(obj, 'PageResult')
except AttributeError:
    QtGui.QMessageBox.warning(None, "Export PDF", "Page object has \
        not been selected.")
else:
```



```
file_exists = os.path.isfile(file_location)
if file_exists == True:
    QtGui.QMessageBox.warning(None, "Export PDF", "A \
        file with the name " + obj.Label + ".pdf already \
        exists:\n\n" + file_location)
else:
    call_inkscape = [inkscape_path, "-f", \
        obj.PageResult, "-A", file_location]
    subprocess.call(call_inkscape)
    QtGui.QMessageBox.information(None, "Export \
        PDF", "Successfully exported:\n\n" + \
        file_location)
```

4.1.3.1 What does it do?

It exports drawing page to PDF file using Inkscape. Instead of bitmap image inserted in PDF file elements like text are searchable and exported PDF file size can be smaller when the drawings do not have big number of elements in it.

4.1.3.2 How to use it

- Select Page object in Tree View.
- Run Export PDF macro.

4.2 Introduction to L^AT_EX

L^AT_EX, I had never heard about this term before doing this project, but when I came to know about its features, found it excellent. L^AT_EX (pronounced /letk/, /letx/, /ltx/, or /ltk/) is a document markup language and document preparation system for the T_EX typesetting program. Within the typesetting system, its name is styled as L^AT_EX.



Figure 4.1: Donald Knuth, Inventor Of T_EX typesetting system

Within the typesetting system, its name is styled as L^AT_EX. The term L^AT_EX refers only to the language in which documents are written, not to the editor used to write those documents. In order to create a document in L^AT_EX, a .tex file must be created using some form of text editor. While most text editors can be used to create a L^AT_EX document, a number of editors have been created specifically for working with L^AT_EX.

L^AT_EX is most widely used by mathematicians, scientists, engineers, philosophers, linguists, economists and other scholars in academia. As a primary or intermediate format, e.g., translating DocBook and other XML-based formats to PDF, L^AT_EX is used because of the high quality of typesetting achievable by T_EX. The typesetting system offers programmable desktop publishing features and extensive facilities for automating most aspects of typesetting and desktop publishing, including numbering and cross-referencing, tables and figures, page layout and bibliographies.

L^AT_EX is intended to provide a high-level language that accesses the power of T_EX. L^AT_EX essentially comprises a collection of T_EX macros and a program to process L^AT_EX documents. Because the T_EX formatting commands are very low-level, it is usually much simpler for end-users to use L^AT_EX.

4.2.1 Typesetting

L^AT_EX is based on the idea that authors should be able to focus on the content of what they are writing without being distracted by its visual presentation. In preparing a L^AT_EX document, the author specifies the logical structure using familiar concepts such as chapter, section, table, figure, etc., and lets the L^AT_EX system worry about the presentation of these structures. It therefore en-

courages the separation of layout from content while still allowing manual typesetting adjustments where needed.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\title{\LaTeX}
\date{}
\begin{document}
  \maketitle
  \LaTeX{} is a document preparation system
  for the \TeX{} typesetting program.
  \par
  $E=mc^2$
\end{document}
```

LaTeX

August 10, 2013

LaTeX is a document preparation system for the TeX typesetting program.
 $E = mc^2$

Figure 4.2: LaTeX output of above program.

4.2.2 Installing LaTeX on System

Installation of LaTeX on personal system is quite easy. As i have used LaTeX on Ubuntu 13.04 so i am discussing the installation steps for Ubuntu 13.04 here:

- Go to terminal and type

```
$ sudo apt-get install texlive-full
```

- Your Latex will be installed on your system and you can check for manual page by typing.

```
$ man latex
```

in terminal which gives manual for latex command.

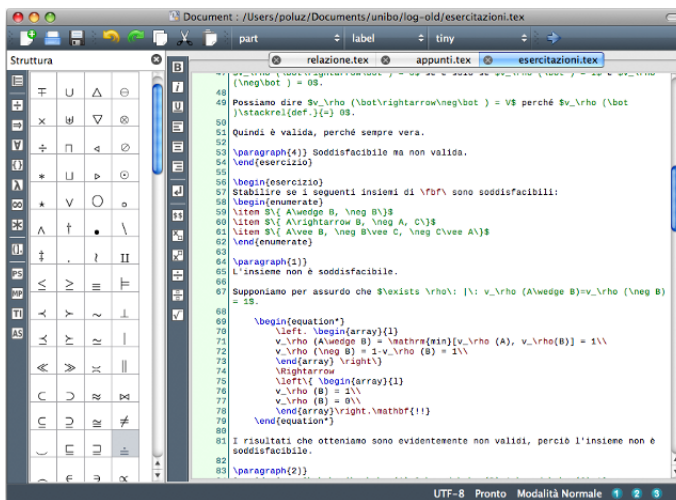
- To do very next step now one should stick this to mind that the document which one is going to produce is written in any type of editor whether it may be your most common usable editor Gedit or you can use vim by installing first vim into your system using command.

```
$ sudo apt-get install vim
```

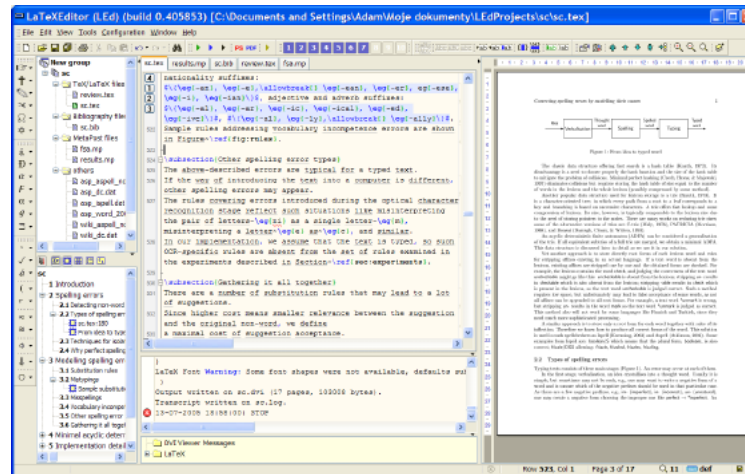
- latex path of the file test.tex Or pdflatex path of the file test.tex*
eg: *pdflatex test.tex*
for producing pdf file simultaneously.
After compiling it type command

4.2.3 Graphical Editors for L^AT_EX

- Texmaker

Figure 4.3: Texmaker, A Graphical L^AT_EX Editor

- And many more but the preferred method to produce L^AT_EX document is through console mode only.

Figure 4.4: LEd, A Graphical \LaTeX Editor

4.2.4 Pdfscreen \LaTeX

There are some packages that can help to have unified document using \LaTeX . Example of such a package is pdfscreen that let the user view its document in two forms-print and screen. Print for hard copy and screen for viewing your document on screen. Download this package from www.ctan.org/tex-archive/macros/latex/contrib/pdfscreen/. Then install it using above mention method.

Just changing print to screen gives an entirely different view. But for working of pdfscreen another package required are comment and fancybox.

The fancybox package provides several different styles of boxes for framing and rotating content in your document. Fancybox provides commands that produce square-cornered boxes with single or double lines, boxes with shadows, and round-cornered boxes with normal or bold lines. You can box mathematics, floats, center, flushleft, and flushright, lists, and pages.

Whereas comments package selectively include/excludes portions of text. The comment package allows you to declare areas of a document to be included or excluded. One need to make these declarations in the preamble of your file. The package uses a method for exclusion that is pretty robust, and can cope with ill-formed bunches of text.

So these extra packages needed to be installed on system for the proper working of pdfscreen package.

4.2.5 Web based graphic generation using \LaTeX

\LaTeX is also useful when there is need of generating the graphics from browser. For example to draw a circle by just entering its radius in html input box. So this kind A of project can be conveniently handled using \LaTeX . Basic idea behind this generation process is that when user clicks on submit button after entering radius a script will run that enter the radius in already made .tex file and recompiles it on server and makes its pdf and postscript file. After that user can view those files by clicking on link provided to view the files. See some screen shots of such a graphic

generation project made by Dr. H.S. Rai:

So here in the above input page which is also the index page user can enter input for length of rectangle, breadth of rectangle and for radius of circle after that user can submit the values. After the values get submitted a script get runs by php code at server side. This script first enters the dimensions of rectangle and circle that were selected by user in to an already existing .tex file and replace with the older dimensions there. After that script recompiles the the tex file and make it available for user.

In above figure it gets clear that .tex file has been compiled and pdf and postscript files are available to user and user can download the graphics so produced. Hence graphics can be generated in L^AT_EX through web interface.

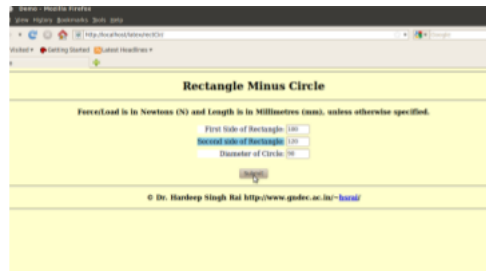


Figure 4.5: Web based graphic generation using L^AT_EX(input page)

4.3 Introduction to Github



Figure 4.6: Github Logo

GitHub is a Git repository web-based hosting service which offers all of the functionality of Git as well as adding many of its own features. Unlike Git which is strictly a command-line tool, Github provides a web-based graphical interface and desktop as well as mobile integration. It also provides access control and several collaboration features such as wikis, task management, and bug tracking and feature requests for every project.

GitHub offers both paid plans for private repto handle everything from small to very large projects with speed and efficiency. ositories, and free accounts, which are usually used to host

open source software projects. As of 2014, Github reports having over 3.4 million users, making it the largest code host in the world.

GitHub has become such a staple amongst the open-source development community that many developers have begun considering it a replacement for a conventional resume and some employers require applications to provide a link to and have an active contributing GitHub account in order to qualify for a job.

The Git feature that really makes it stand apart from nearly every other Source Code Management (SCM) out there is its branching model.

Git allows and encourages you to have multiple local branches that can be entirely independent of each other. The creation, merging, and deletion of those lines of development takes seconds.

This means that you can do things like:

- **Frictionless Context Switching.**
Create a branch to try out an idea, commit a few times, switch back to where you branched from, apply a patch, switch back to where you are experimenting, and merge it in.
- **Role-Based Codelines.**
Have a branch that always contains only what goes to production, another that you merge work into for testing, and several smaller ones for day to day work.
- **Feature Based Workflow.**
Create new branches for each new feature you're working on so you can seamlessly switch back and forth between them, then delete each branch when that feature gets merged into your main line.
- **Disposable Experimentation.**
Create a branch to experiment in, realize it's not going to work, and just delete it - abandoning the work with nobody else ever seeing it (even if you've pushed other branches in the meantime).

Notably, when you push to a remote repository, you do not have to push all of your branches. You can choose to share just one of your branches, a few of them, or all of them. This tends to free people to try new ideas without worrying about having to plan how and when they are going to merge it in or share it with others.

There are ways to accomplish some of this with other systems, but the work involved is much more difficult and error-prone. Git makes this process incredibly easy and it changes the way most developers work when they learn it.

4.3.1 What is Git?

Git is a distributed revision control and source code management (SCM) system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows. Git was initially designed and developed by Linus Torvalds for Linux kernel development in 2005, and has since



Figure 4.7: Git Logo

become the most widely adopted version control system for software development.

As with most other distributed revision control systems, and unlike most clientserver systems, every Git working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server. Like the Linux kernel, Git is free and open source software distributed under the terms of the GNU General Public License version 2 to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

4.3.2 Installation of Git

Installation of git is a very easy process. The current git version is: 2.0.4. Type the commands in the terminal:

```
$ sudo apt-get update
```

```
$ sudo apt-get install git
```

This will install the git on your pc or laptop.

4.3.3 Various Git Commands

Git is the open source distributed version control system that facilitates GitHub activities on your laptop or desktop. The commonly used Git command line instructions are:-

4.3.3.1 Create Repositories

Start a new repository or obtain from an exiting URL

\$ git init [project-name]

Creates a new local repository with the specified name

\$ git clone [url]

Downloads a project and its entire version history

4.3.3.2 Make Changes

Review edits and craft a commit transaction

\$ git status

Lists all new or modified files to be committed

\$ git diff

Shows file differences not yet staged

\$ git add [file]

Snapshots the file in preparation for versioning

\$ git reset [file]

Unstages the file, but preserve its contents

\$ git commit -m "[descriptive message]"

Records file snapshots permanently in version history

4.3.3.3 Group Changes

Name a series of commits and combine completed efforts

\$ git branch

Lists all local branches in the current repository

\$ git branch [branch-name]

Creates a new branch

\$ git checkout [branch-name]

Switches to the specified branch and updates the working directory

\$ git merge [branch]

Combines the specified branches history into the current branch

\$ git branch -d [branch-name]

Deletes the specified branch

4.3.3.4 Save Fragments

Shelve and restore incomplete changes

\$ git stash

Temporarily stores all modified tracked files

\$ git stash pop

Restores the most recently stashed files

\$ git stash list

Lists all stashed changesets

\$ git stash drop

Discards the most recently stashed changeset

4.3.3.5 Synchronize Changes

Register a repository bookmark and exchange version history

\$ git fetch [bookmark]

Downloads all history from the repository bookmark

\$ git merge [bookmark /[branch]]

Combines bookmarks branch into current local branch

\$ git push [alias [branch]]

Uploads all local branch commits to GitHub

\$ git pull

Downloads bookmark history and incorporates changes

5.1 Technical and Managerial Lesson Learnt

I learned a lot by doing this project . During this period I got to learn a vast number of technologies. These are listed below :

- **Operating system:** Ubuntu
- **Languages used:** Python, Bash scripting
- **Framework:** Django
- **Typesetting:** LaTeX
- **Other Learnings:** Internet Relay Chat(IRC), Wordpress

So during this project I learned all the above things. Above all I got to know how Softwares are developed from the scratch. Planning, designing, developing code, working in a team, testing etc. These are all very precious things I got to learn during this period.

5.1.1 Ubuntu: An open source OS

During my training, I also got familiar with a great and open source Operating System, Ubuntu. Firstly, it was quite difficult for a regular MS Windows user to port to Ubuntu. I did all of my project work using this vast operating system. Ubuntu (/ubuntu/ oo-BOON-too) is a Debian-based Linux operating system, with Unity as its default desktop environment. It is based on free software and named after the Southern African philosophy of ubuntu (literally, "human-ness"), which often is translated as "humanity towards others" or "the belief in a universal bond of sharing that connects all humanity".

Ubuntu's goal is to be secure "out-of-the box". By default user's programs run with low privileges and cannot corrupt the operating system or other user's files. For increased security, the sudo tool is used to assign temporary privileges for performing administrative tasks, which allows the root account to remain locked and helps prevent inexperienced users from inadvertently making catastrophic system changes or opening security holes.

5.1.2 Internet Relay Chat

Internet Relay Chat (IRC) is an application layer protocol that facilitates transfer of messages in the form of text. The chat process works on a client/server model of networking. IRC clients are computer programs that a user can install on their system. These clients are able to communicate with chat servers to transfer messages to other clients. It is mainly designed for group communication in discussion forums, called channels, but also allows one-to-one communication via private message as well as chat and data transfer, including file sharing. Client software is available for every major operating system that supports Internet access. IRC is an open protocol that uses TCP and, optionally, TLS. An IRC server can connect to other IRC servers to expand the IRC network. Users access IRC networks by connecting a client to a server. There are many client implementations, such as mIRC, HexChat and irssi, and server implementations, e.g. the original IRCd. Most IRC servers do not require users to register an account but a user will have to set a nickname before being connected.

IRC has a line-based structure with the client sending single-line messages to the server, receiving replies to those messages and receiving copies of some messages sent by other clients. In most clients, users can enter commands by prefixing them with a '/'. Depending on the command, these may either be handled entirely by the client, or passed directly to the server, possibly with some modification.

The basic means of communicating to a group of users in an established IRC session is through a channel. Channels on a network can be displayed using the IRC command LIST, which lists all currently available channels that do not have the modes +s or +p set, on that particular network. Users can join a channel using the JOIN command, in most clients available as /join #channel-name. Messages sent to the joined channels are then relayed to all other users.

I used "Xchat IRC" client for IRC during my training to communicate with the Open Source communities whenever I found some error during my work or for some extra knowledge. XChat is a graphical IRC Client with a GTK+ GUI. It's special features include the multiple server/channel windows, dialog windows, plugin API.

5.1.3 Shell Scripting

Because all my project work was being done in Ubuntu (i.e. a Linux distribution), I came to know the power of Bash(Terminal). We can simply call it a "Shell". It is a great tool to work faster. With terminal, the efficiency of work increases. Ubuntu uses the Bash(Bourne Again Shell) as default shell. During training, I came to know about the new commands daily. Working with CLI(Command Line Interface) is a fun.

- [1] FreeCAD, <https://github.com/FreeCAD/FreeCAD>
- [2] \LaTeX Beginner's Guide By Stefan Kottwitz [Pact Publishing]
- [3] My Blog, <http://amritpals.com>
- [4] My Github Profile, <http://github.com/amrit3701>