



# Summer Fellowship Report

On

**Python Code for Connection Design and 3D  
Drawing , Development of GUI, Python 2 to 3  
Conversion and Creating Python3 Windows  
Installer**

Submitted by

**Pragya**

Under the guidance of

**Prof. Siddhartha Ghosh**  
Civil Engineering Department  
IIT Bombay

July 11, 2019

## Acknowledgment

I would like to thank the FOSSEE Summer Fellowship, IIT Bombay for giving me an opportunity to do the internship in the floss PYTHON and with the Osdag. It helped me to enhance my knowledge in python, making of installer for the software and testing of software. Therefore, I consider myself as a very lucky individual as I was provided with an opportunity to be a part of it. I am also grateful for having a chance to meet so many wonderful people and professionals who led me through this internship period.

I would like to specially acknowledge Prof. Siddhartha Ghosh with my deepest gratitude who in spite of being busy with his duties, took time out to hear, guide and keep me on the correct path and allowing me to carry out my project at their esteemed research lab (SSRR lab).

I would also like to acknowledge the members of the Osdag team; Danish Ansari (Project Research Assistant), Ajmal Babu MS (Project Research Engineer) for their careful and precious guidance which was extremely valuable for my both theoretical and practical study.

I perceive as this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperation with all of you in the future.

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	What is Osdag Software? . . . . .	4
1.2	Who can use ? . . . . .	8
<b>2</b>	<b>Software Modification</b>	<b>9</b>
2.1	GUI Alteration . . . . .	9
2.1.1	Disabling of Combo-box . . . . .	10
2.1.2	Adding Title . . . . .	12
2.1.3	Adding Connectivity Images . . . . .	12
2.1.4	UI of Pitch Details . . . . .	13
2.1.5	Hide Quick Access Bar . . . . .	14
2.1.6	UI of Stiffener Details . . . . .	14
2.2	Beam-Beam Log Messages . . . . .	15
2.3	Fixing Load Input . . . . .	16
2.4	Debugging of Beam-Beam Code . . . . .	16
<b>3</b>	<b>Writing Python Code for 3D Drawing</b>	<b>17</b>
<b>4</b>	<b>Python 2.7 to Python 3.6</b>	<b>19</b>
<b>5</b>	<b>Windows Installer</b>	<b>20</b>
5.1	GitHub Link . . . . .	20
<b>6</b>	<b>Testing and Installation</b>	<b>21</b>

<b>7</b>	<b>Restructuring of Osdag</b>	<b>23</b>
<b>8</b>	<b>Workshop Volunteering</b>	<b>24</b>
<b>9</b>	<b>Conclusion</b>	<b>25</b>

# Chapter 1

## Introduction

Python internship is provided under the FOSSEE project. FOSSEE project promotes the use of FOSS (Free and Open Source Software) tools to improve quality of education in our country. FOSSEE encourages the use of FOSS tools through various activities to ensure commercial (paid) softwares are replaced by equivalent FOSS tools.

The [FOSSEE](#) project is a part of the National Mission on Education through Infrastructure and Communication Technology(ICT), Ministry of Human Resources and Development, Government of India.

Osdag is one such open source software which comes under the FOSSEE project. Osdag internship is provided through FOSSEE project. Any UG/PG/PhD holder can apply for this internship. And the selection will be based on a screening task.

### 1.1 What is Osdag Software?

Osdag is a cross-platform free and open-source software for the design of steel structures, following the Indian standard IS

800:2007. It allows the users to design steel connections, members and systems using a graphical user interface. The interactive GUI provides a 3D visualisation of the designed component and creates images for construction/fabrication drawings.

It is used for solving steel structures problems and to see how the connection will look after practical implementation. There are different modules available in Osdag with various connectivities.

Osdag provides various features such as:

- An interactive window displaying a 3D CAD model, which provides a clear visualisation of the designed component.

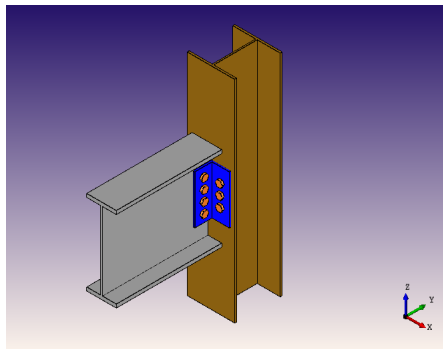


Figure 1.1: CAD Image

- Creation of 3D CAD models that can be imported to generic CAD softwares.
- User-friendly input and output docs, with text-validated fields grouped according to the design flow.

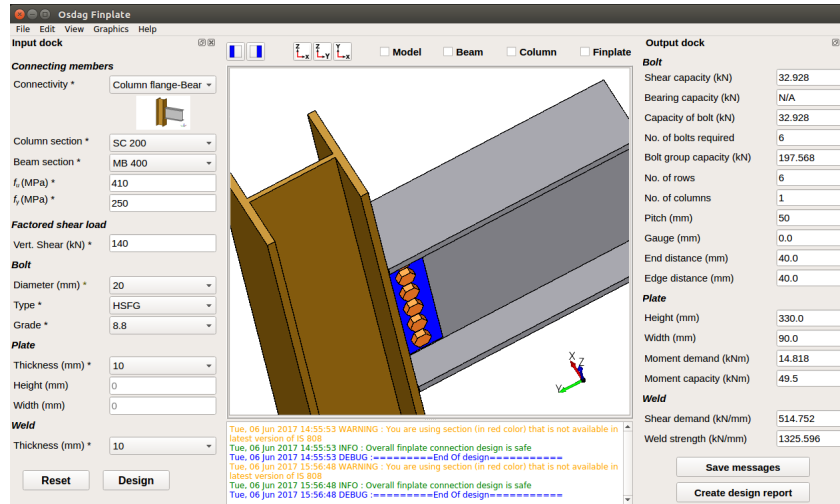


Figure 1.2: Module Page

- A text window for message display, that also suggests necessary changes if a trial design is found unsafe.

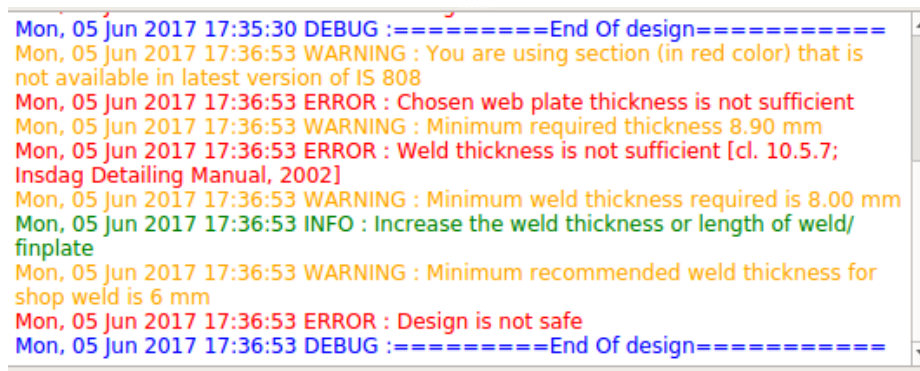


Figure 1.3: Log Messages

- Creation of a professional design report showing all necessary checks, design calculations as per IS 800:2007, and standard views of the designed component.

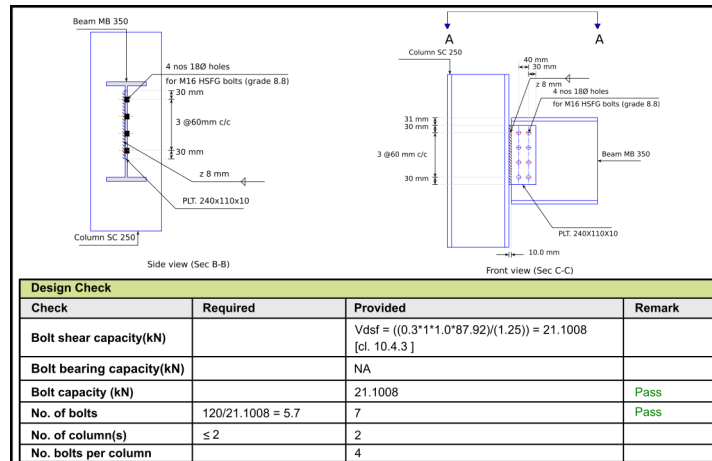


Figure 1.4: Design Report

- Creation of 2D vector (and raster) images that can be used in a design report or class assignment.

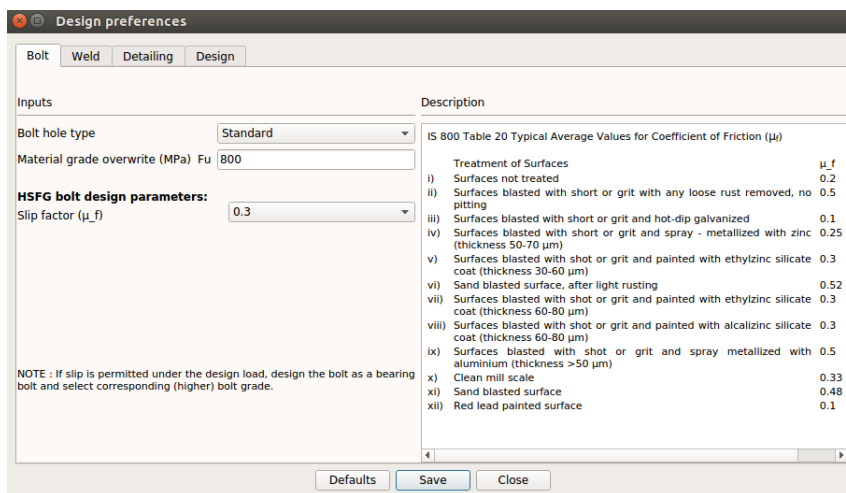


Figure 1.5: Design Preferences

- Selection of design preferences, considering different construction and detailing aspects, using a design preference toolbox.



## 1.2 Who can use ?

Osdag is generally created for industry professionals but it also keep students in mind. As Osdag is funded by MHRD, Osdag team tries to manipulate software in such a way that it can be used by the students during their academics and to give them a better insight look in the subject.

Basically Osdag can be used by anyone starting from novice to professionals. It's simple and sober user interface makes it flexible and attractive than the other softwares. Also there are video tutorials to get started. The video tutorials of Osdag can be accessed [here](#).

# Chapter 2

## Software Modification

### 2.1 GUI Alteration

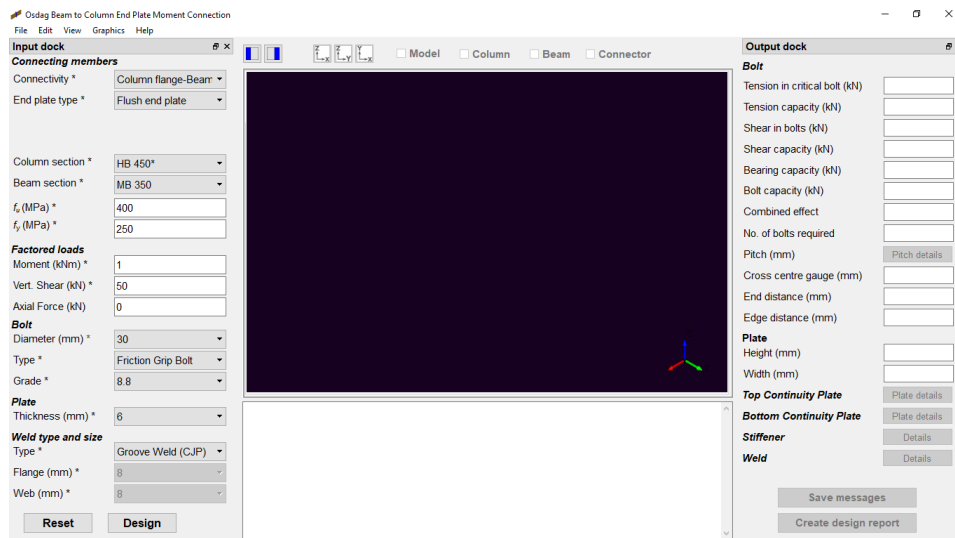


Figure 2.1: Beam to Column GUI

Osdag software basically uses Python 2.7 version. The front-end is developed on PyQt5. PyQt5 is a comprehensive set of Python bindings for Qt v5. It is implemented as more than 35 extension modules and enables Python to be used as an alternative application development language to C++ on all supported platforms including iOS and Android.

The back-end is developed in SQLite. SQLite is a C-language library that implements a small, fast, self-contained,

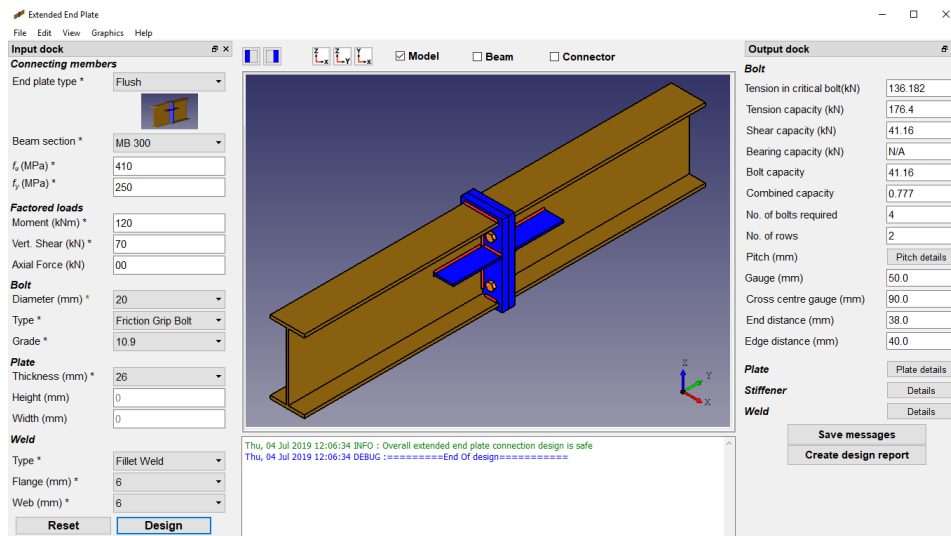


Figure 2.2: Beam to Beam Extended End Plate GUI

high-reliability, full-featured, SQL database engine. SQLite is the most used database engine in the world.

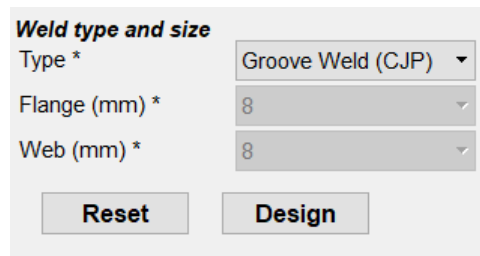
So, I have done alterations in GUI and added features required for the software and so that it can be made more handy to users.

### 2.1.1 Disabling of Combo-box

#### Beam-Column

Once you go through the software, you will find different modules for different type of connection so I basically modified the input dock of beam to column end plate connection. There are number of combo-box present to take the input from user to built that connection. Clearly in the below figure you can see there are three combo-boxes in the end of input dock, first is to select type of weld, I did that when "Groove Weld" is selected the other two remain disabled so that user cannot fill it.

There were some bugs that it was coded in such a way if any field is empty it will pop-up and ask to fill first that field so I fixed the same.



**Weld type and size**

Type \* Groove Weld (CJP) ▾

Flange (mm) \* 8 ▾

Web (mm) \* 8 ▾

Reset Design

Figure 2.3: Disabled Combo-box in Beam - Column

[https://github.com/Pragya007/Osdag/blob/bc\\_endplate/Connections/Moment/BCEndPlate/bc\\_endplate\\_main.py](https://github.com/Pragya007/Osdag/blob/bc_endplate/Connections/Moment/BCEndPlate/bc_endplate_main.py)

## Beam-Beam

This is different module, here also I debug the code and did the same alterations as above.



Figure 2.4: Disabled Combo-box in Beam - Beam

[https://github.com/Pragya007/Osdag/blob/bc\\_endplate/Connections/Moment/ExtendedEndPlate/extended\\_main.py](https://github.com/Pragya007/Osdag/blob/bc_endplate/Connections/Moment/ExtendedEndPlate/extended_main.py)

### 2.1.2 Adding Title

I changed the title in beam-column.

### 2.1.3 Adding Connectivity Images

#### Beam-Column

I have added six connectivity images for respective different connections. So that when user select different options in the combo-boxes with respect to that the image is shown there. ‘

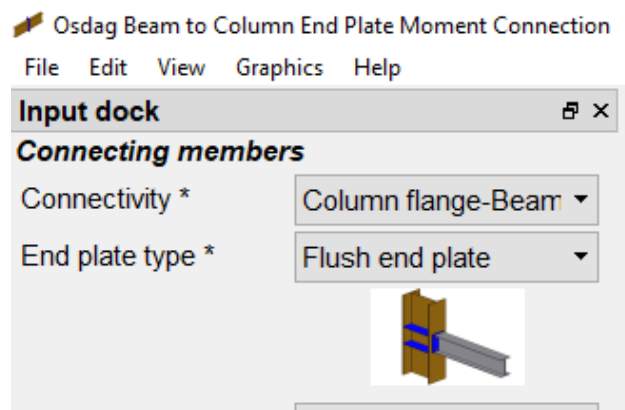


Figure 2.5: Connectivity Images in Beam-Column

## Beam-Beam

In this module I added three connectivity images.

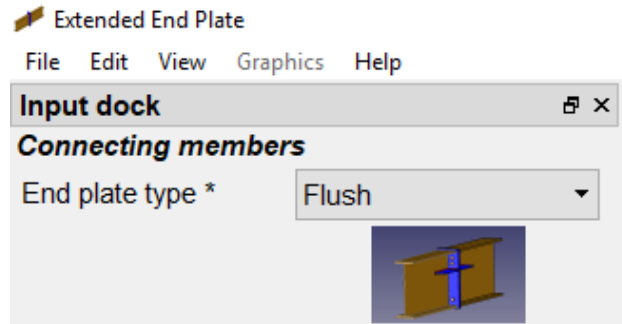


Figure 2.6: Connectivity Images in Beam-Beam

### 2.1.4 UI of Pitch Details

Each design will be having different number of pitch depending upon how much bolts are present in the model, so I added the images of different pitch accordingly, this option is present in output dock which will tell user how many pitch are there in model.

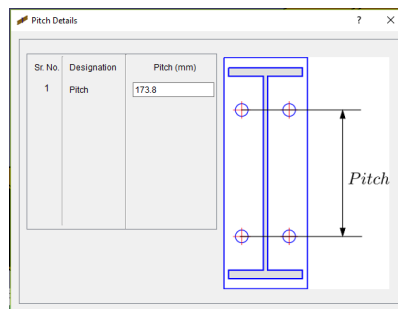


Figure 2.7: Pitch Details 1

[https://github.com/Pragya007/Osdag/blob/bc\\_endplate/Connections/Moment/ExtendedEndPlate/ui\\_pitch.ui](https://github.com/Pragya007/Osdag/blob/bc_endplate/Connections/Moment/ExtendedEndPlate/ui_pitch.ui)

[https://github.com/Pragya007/Osdag/blob/bc\\_endplate/Connections/Moment/ExtendedEndPlate/ui\\_plate.py](https://github.com/Pragya007/Osdag/blob/bc_endplate/Connections/Moment/ExtendedEndPlate/ui_plate.py)

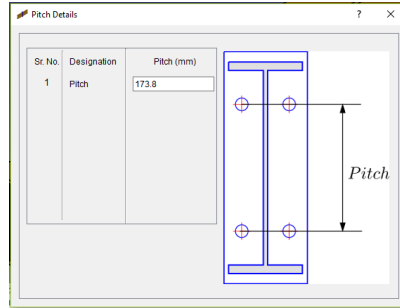


Figure 2.8: Pitch Details 2

### 2.1.5 Hide Quick Access Bar

There is bar where you can quickly change the setting of views. You can change the output window and can set it to full or half screen by hiding input and output dock and can again reset it by clicking one option, but it will be enabled when once user get the safe design model and it's CAD images is generated, so I fixed it so that it will get disable/enable accordingly.

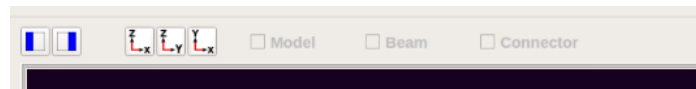


Figure 2.9: Quick Access Bar

### 2.1.6 UI of Stiffener Details

I added images of stiffener and it will change accordingly in selection of different type of weld.

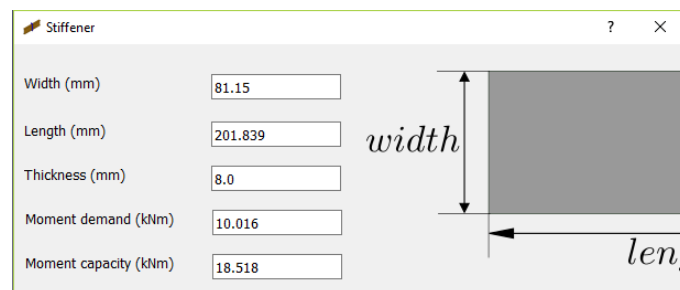


Figure 2.10: Stiffener Detail 1

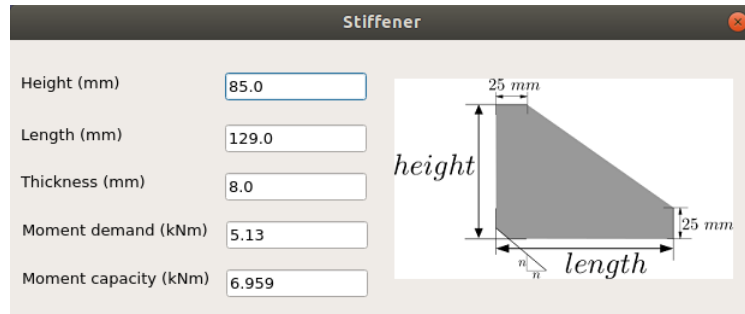


Figure 2.11: Stiffener Detail 2

[https://github.com/Pragya007/Osdag/blob/bc\\_endplate/Connections/Moment/ExtendedEndPlate/ui\\_stiffener.py](https://github.com/Pragya007/Osdag/blob/bc_endplate/Connections/Moment/ExtendedEndPlate/ui_stiffener.py)

[https://github.com/Pragya007/Osdag/blob/bc\\_endplate/Connections/Moment/ExtendedEndPlate/ui\\_stiffener.ui](https://github.com/Pragya007/Osdag/blob/bc_endplate/Connections/Moment/ExtendedEndPlate/ui_stiffener.ui)

## 2.2 Beam-Beam Log Messages

When you hit the start design button you will be getting log messages which are displayed to tell user when the design is not safe: the error, what changes you could possibly make: the warning, and if you have created a safe design then: safe messages in different colors, so I fixed so that it will come in colored and will not confuse user.

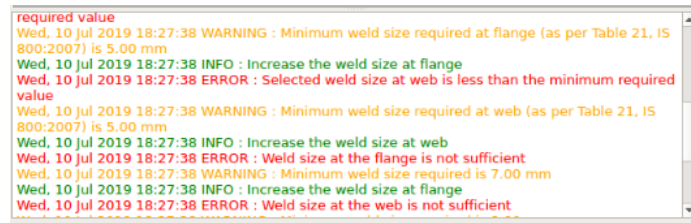


Figure 2.12: Log Messages



## 2.3 Fixing Load Input

In the title bar there is one option where you can load file(\*.osi) that is osi file which you can save earlier from the input dock by the option itself after filling it once, so it was not loaded properly so I fixed that bug.

## 2.4 Debugging of Beam-Beam Code

Some bugs were there and it was not working properly so fixed those things.

[https://github.com/Pragya007/Osdag/blob/test/Connections/Moment/ExtendedEndPlate/extended\\_main.py](https://github.com/Pragya007/Osdag/blob/test/Connections/Moment/ExtendedEndPlate/extended_main.py)

## Chapter 3

# Writing Python Code for 3D Drawing

I got task to create fillet weld between continuity plates and column in beam - column module. So, there were four continuity plates for which I need to add 24 fillet weld, 6 for each plate by writing code.

With the use of OCC dependency in python we create fillet weld in the shape as shown below having three dimensions(b,h,l) which I passed in the [filletweld class](#) then I set the local origin with the global origin so as to place it at suitable position.[Github Link to Class\(Origin & Orientation](#)

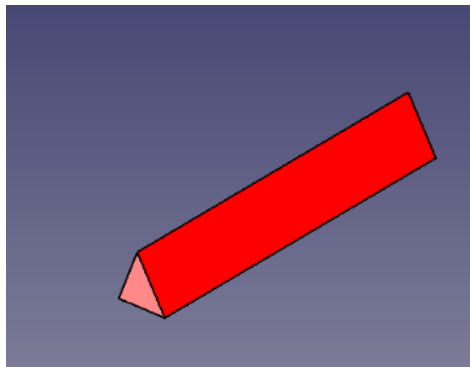


Figure 3.1: Fillet Weld

After setting the origin the I set the orientation, as direction of weld also plays an important role because of it's shape it should be oriented in correct direction. Then I write function to display those welds in the CAD model. [Github Link to Main File\(Display\)](#)

Also with the use of an OOP concept known as inheritance I added groove welds in beam - column module. The difference between groove and fillet weld is that groove is placed between two surfaces and fillet is applied after joining the two surfaces.

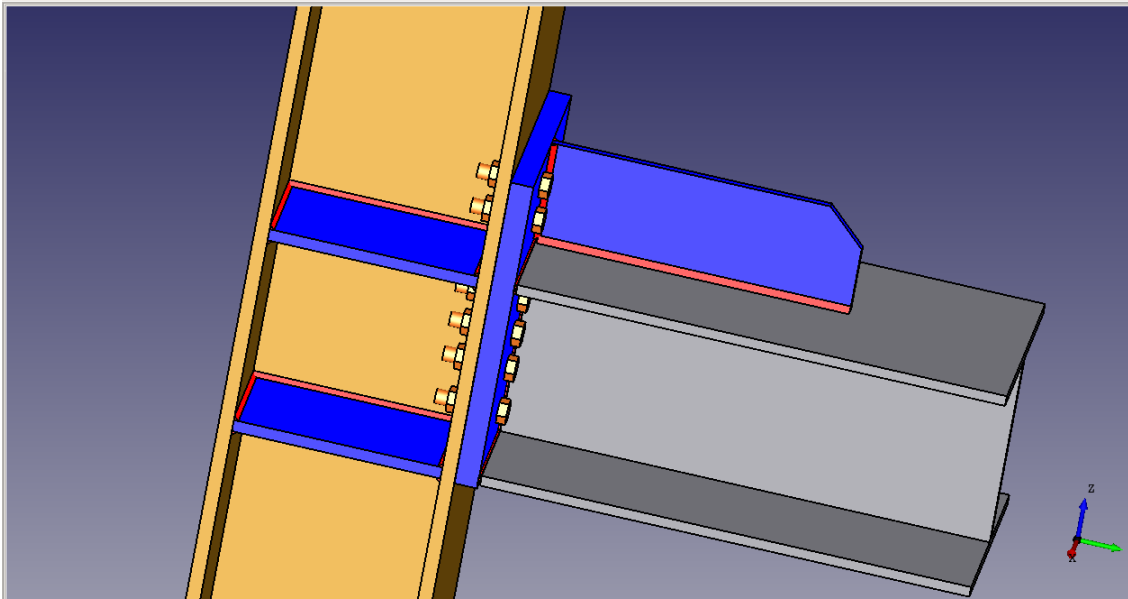


Figure 3.2: BC Fillet Welds

## Chapter 4

### Python 2.7 to Python 3.6

As in the growing world we also need to be updated so when this software started python 2.7 was the stable version and team opted that but now we are having python 3.6 as a stable one in our hand so we decided it to shift to this stable and updated version.

So what I did I used "lib2to3" library to convert all the codes to python 3 by changing respective Syntax. But there are some dependencies which are different in their names so we need to find all these, this took time but I find all those and changed respectively.

# Chapter 5

## Windows Installer

I have created the NSIS script for the installer and when you compile this NSIS script with the files which is having all the codes then it will create a installer which can be used to install the software on different computers.

The GitHub link is given to that NSIS script.

### 5.1 GitHub Link

[NSIS Script for windows installer](#)

## Chapter 6

### Testing and Installation

After addition of two new modules we came up with the newer version of software having new features and in python 3 application.

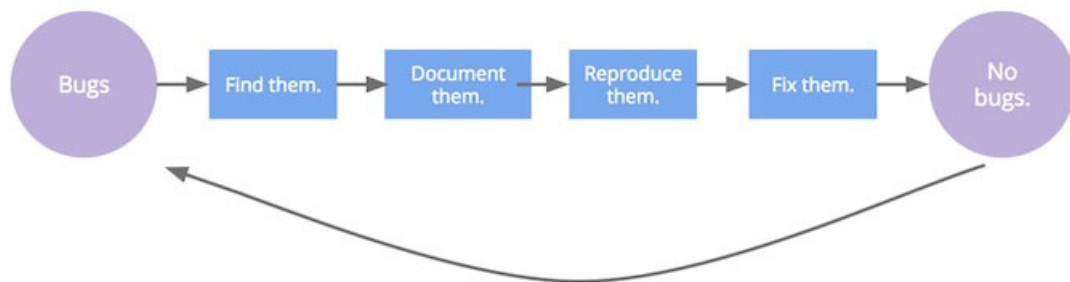
The task was to test each and every module with its sub-connectivities currently available in Osdag. There was a need to test the software for edge and corner cases and make sure that the software gave appropriate results/suggestions. Testing of 2-D drawings and design report was equally important. Last but not the least testing of the GUI and other small features was required to ensure smooth functioning of the software. The found bugs were documented and reported to the Osdag team where the members worked on fixing the bugs/issues simultaneously.

A bug in a software is an error, flaw, or fault in a computer program or system that causes it to produce an incorrect or unexpected result or to behave in unintended ways. Every software or application either old or new might be having some bugs, sometimes noticeable while sometimes non-noticeable. And Osdag being in its initial stages, it is natural that it might have bugs.

Finding new bugs was really a challenging task, to report any bug statement it was necessary to cross check that

statement for every section, connectivity and for various input values. Generally my task follows the following flow chart.

Test the software → Find bugs (If any) → Report to osdag team → Bug fixed by osdag team → Test the software after bug fixed.



## Chapter 7

### Restructuring of Osdag

Task was given to review all the code of module and restructure to avoid repetition of code as same methods and classes are written number of times hence to improve efficiency neatness of code we need to make different classes and methods their using oop's concept we can recreate every classes in a structured way.

Also we need to implement MVC (Model View Controller), which will ensure the efficiency of software, we tried doing this and had added some of our ideas to it but due to lack of time we were unable to complete it.



## Chapter 8

### Workshop Volunteering

I also volunteered in Osdag Workshop held on June 16,2019. There we installed the software in all computers and tested it.



Figure 8.1: Osdag Workshop

## Chapter 9

### Conclusion

After completing my summer fellowship, I had been exposed to software developer and programmer's working life. Throughout my fellowship, I could understand more about the definition of an software developer and programmer and prepare myself to become a responsible and innovative developer and programmer in future. Throughout my fellowship period, I realize that observation is a main element to find out the root cause of a problem. Not only for my fellowship, but daily activities too.

During my project, I cooperate with my colleagues and operators to determine the problems. Moreover, the project indirectly helped me to learn independently, discipline myself, be considerate/patient, self-trust, take initiative and the ability to solve problems. Besides, my communication skills has strengthened as well when communicating with others. During my internship period, I have received criticism and advice from engineers and mentors whenever mistakes were made. However, those advises are useful guidance for me to change myself and avoid myself making the same mistakes again.

Apart from that, I had also developed my programming skills through various programs that I had done. This also helps sharpen my skills in python, PyQt, SQLite since most

of the programs were done with the aid of PyCharm. To sum up, the activities that I had learned during summer fellowship really are useful for me in future to face challenges in a working environment.

Here during the internship period I developed my skills in following softwares/tools :

1. Python
2. PyQt
3. Latex
4. Git and Git hub

I would like to once again appreciate everyone who has made my summer fellowship a superb experience.