Dynamics of Structure (DoS)

Submitted for partial fulfilment of the Degree
of
Bachelor of Technology
(Computer Science and Engineering)

Submitted By:
Amarjeet Singh Kapoor
1311017

Submitted To:
Sukhjit Singh Sehra
Training Coordinator
CSE Department

Department of Computer Science & Engineering
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.

Finally, I would thanks to all whoever have contributed in this report work with Mandeep Singh (D4 CSE) and all other trainees. Without their encouragement, it would not have been possible to complete this project in such an efficient manner.

Amarjeet Singh Kapoor

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION OF ORGANIZATION

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 ever 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.

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

CHAPTER 2

INTRODUCTION TO PROJECT

2.1 Overview

Dynamics of Structures (DoS) is an open-source, free web-based software, developed by students of Testing and Consultancy Cell (TCC), under the guidance of Dr. H.S. Rai. This software is used to compute the Modes of vibration in which the structure can move and also force applied on each floor due to the vibration caused by the earthquake. So, that Civil engineers can analysis the stability of structure consisting of many stories.

The main task of this application is to get data as input from user and then it can compute the result at back-end and when the result is ready it send output as email. This software is structured by keeping in view that user of this software can be both a Civil Engineer or a simple man whose job is just to enter data to software in order that an engineer can analyze later from the result stability of the structure. This software also provides the intermediate values for engineering student to deeply analyze the process of computation to be done in order to get output values.

The core part of DoS is implemented using Sagemath for processing, LaTeX for output file (PDF) generation, django for web interface. Bash Shell Scripting has been used to join all these. To provide the User Experience to the users, Bootstrap has been used.

My training being not based on particular language or technology, different type of open-source softwares and technologies are used in this project and many during my training which are not used in this project like OpenCV (for image processing and computer vision), CGI (for web interface through c++).

2.2 The Existing System

There are few existing systems for solving this particular problem like STAAD.Pro, SAP2000 but they don't have following features required by our mentor. These system were not open source and free web based software that were need.

All exiting system suffers from at least one of the following system.

Limitations of previous system

- No batch mode
- Don't give output as PDF

- They are costly (STAAD.Pro costs nearly 80,000 rupees)
- They don't mail output
- They need installation and a lot of system resources
- They don't accept values in form of *.csv file

2.3 User Requirement Analysis

For User Requirement Analysis, users of this system have been asked about possible requirements that this software should have and we got following resultant list of outputs-:

1. Generates the final output in the form of pdf
2. Provide on-line way to analysis so that individual does not have to install anything.
3. Send PDF to the user in the form of email
4. Make it work like batch mode. so, that user can give inputs together and relax.
5. Accept inputs from the user in *.csv file format
6. Help M.Tech and Civil Engineer to analysis structure.
7. Automate calculation of modal force and modes.
8. Reduce the time for analysis.

2.4 Feasibility Analysis

Feasibility analysis aims to uncover the strengths and weaknesses of a project. In its simplest term, the two criteria to judge feasibility are cost required and value to be attained. As such, a well-designed feasibility analysis should provide a historical background of the project, description of the project or service, details of the operations and management and legal requirements. Generally, feasibility analysis precedes technical development and project implementation. There is some feasibility factors by which we can determine that project is feasible or not:

- **Technical feasibility**: Technological feasibility is carried out to determine whether the project has the capability, in terms of software, hardware, personnel to handle and fulfill the user requirements. This whole project is based on solving Mathematics equations for which we have used SAGEMATH and to provide output we have used LaTeX for providing the output and Django for user interface. Technical feasibility of this project revolves around the technical boundaries and limitations of the SAGEMATH, LaTeX and Django. But as LaTeX is much powerful Typesetting tool and Django is secure and structured server side framework, so these languages and technologies are perfect to design the software under this project. Dynamics of sturcture is technically feasible as it is built up in Open Source Environment and thus it can be run on any Open Source platform.

- **Economic feasibility**: Economic analysis is the most frequently used method to determine the cost/benefit factor for evaluating the effectiveness of a new system. In this analysis we determine whether the benefit is gain according to the cost invested to develop the project or not. If benefits outweigh costs, only then the decision is made to design and implement the system. It is important to identify cost and benefit factors, which can be categorized as follows:

  1. Development costs.
  2. Operating costs.

  Dynamics of structure Software is also Economically feasible with 0 Development and Operating Charges as it is developed in Django framework, SAGEMATH and LaTeX which is FOSS technology and the software is operated on Open Source platform.

- **Operational feasibility**: Operational feasibility is a measure of how well a project solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. All the Operations performed in the software are very quick and satisfies all the reuirements. This project is also operational feasible as it automates the work of solving the problem of analysising the structures which not only saves time but also saves money as most of the work is done by Employees and M.Tech students is done by this software.

2.5 Objective of Project

Dynamic of structure is a web based software and the main objectives of this project is to -:

1. To inspire M.Tech students to automate their work and do programming
2. Perform most of difficult Calculation work.
3. Make it work like batch mode. so, that user can give inputs together and relax.
4. Accept inputs from the user in *.csv file format
5. Help M.Tech and Civil Engineer to analysis structure.
6. Automatic calculation of modal force and modes.
7. Reduce the time for analysis.
8. Generates the final output in the form of pdf
9. Provide on-line way to analysis so that individual does not have to install anything.
10. Send PDF to the user in the form of email.

CHAPTER 3

PROJECT DESIGN

3.1 Software Requirement Analysis

A Software Requirements Analysis for a software system is a complete description of the behavior of a system to be developed. It include functional Requirements and Software Requirements. In addition to these, the SRS also contains non-functional requirements. Non-functional requirements are requirements which impose constraints on the design or implementation.

- **Purpose**: Dynamic of structure is a web based software and the main purpose of this project is to:
  1. Perform most of difficult Calculation work.
  2. Make it work like batch mode. so, that user can give inputs together and relax.
  3. Help M.Tech and Civil Engineer to analysis structure.
  4. Automatic calculation of modal force and modes.
  5. Reduce the time for analysis.
  6. Provide on-line way to analysis so that individual does not have to install anything.
- **Users of the System**
  1. Client : Clients are the end users that benefit from this software. They just provide input and gets output in form of PDF.Client of this WEB Application can be of two types -:
     (a) Civil Engineer -: They have knowledge of working of procedure and what input is being provided.
     (b) Layman -: They don't know anything about what's going on, their just work is to give input to system.

3.1.1 Functional Requirements

- **Specific Requirements**: This phase covers the whole requirements for the system. After understanding the system we need the input data to the system then we watch the output

and determine whether the output from the system is according to our requirements or not. So what we have to input and then what well get as output is given in this phase. This phase also describe the software and non-function requirements of the system.

- **Input Requirements of the System**

  1. Type of soil
  2. Number of storeys
  3. Importance Factor
  4. Response Reduction Factor
  5. Zone Factor
  6. Input method (CSV or manual)
  7. Output method (Email or direct PDF)
  8. Mass of each storey
  9. Height of each storey
  10. Stiffness of each storey
  11. Input in form of csv file

- **Output Requirements of the System**

  1. Calculation of modal force and modes.
  2. Generation of output in form of PDF.
  3. Mailing output PDF

- **Special User Requirements**

  1. Automatic Email Generation of Output and Sending to the concerned person.
  2. Taking bulk input values in form of CSV file

- **Software Requirements**

  1. Programming language: Python 2.7
  2. software: SAGEMATH, LaTeX
  3. Framework: Django 1.7, Bootstrap
  4. Web Languages: Html, Java Script, CSS
  5. Database: Sqlite
  6. Documentation: Doxygen 1.8.3
  7. Text Editor: Vim
  8. Operating System: Ubuntu 12.04 or up
  9. Revision System: Git

3.1.2 Non functional requirements

1. Scalability: System should be able to handle a number of users. For e.g., handling around thousand users at the same time.
2. Usability: Simple user interfaces that a layman can understand.
3. Speed: Processing input should be done in reasonable time i.e. we can say maximum 24 hrs.

3.2 DFDs

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs of DoS is as following-:

1. Data flow LEVEL 0 figure 3.1
2. Data flow LEVEL 1 figure 3.2

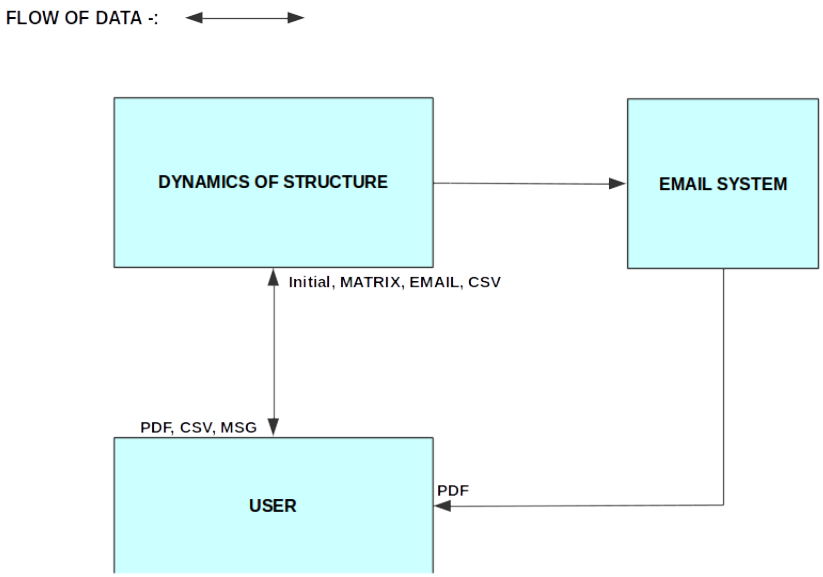


Figure 3.1: Data flow LEVEL 0

Here, In figure 3.1 and figure 3.2

1. MSG means Message
2. initial represent all initial input value
3. matrix represent all Mass, Height, Stiffness matrix

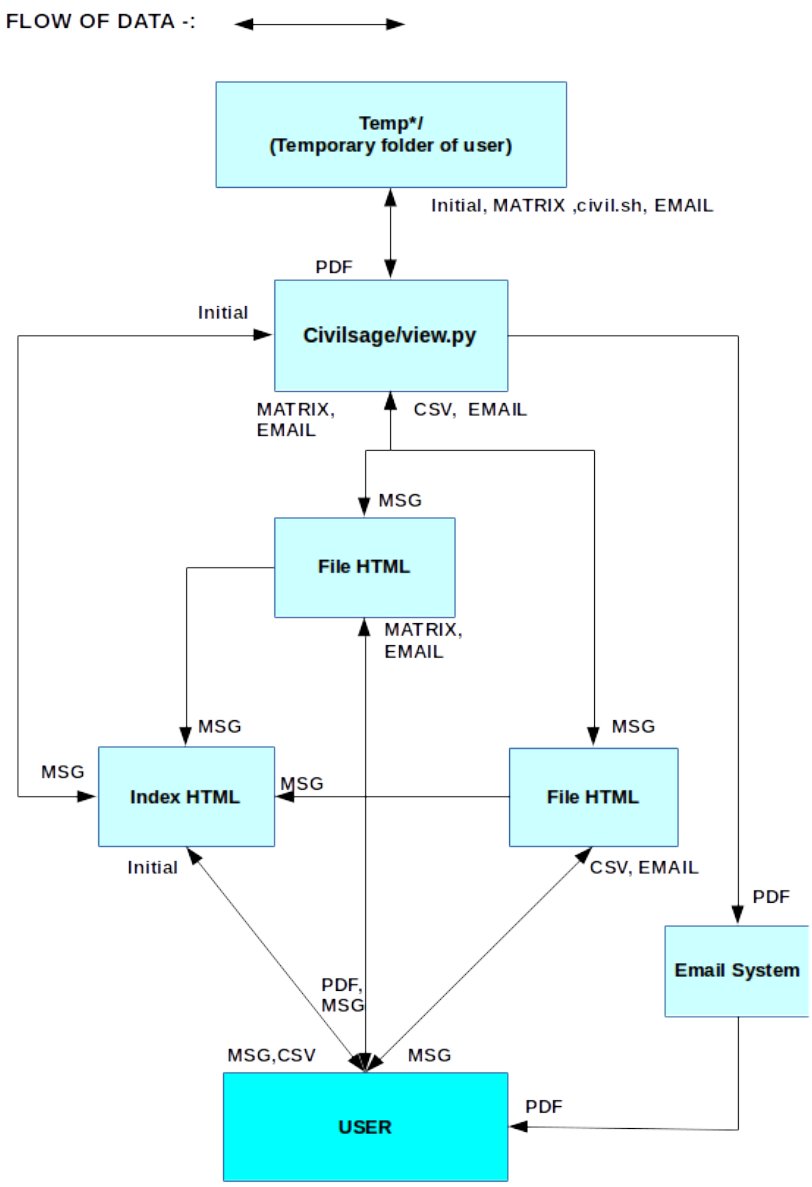


Figure 3.2: Data Flow LEVEL 1

3.3 UI Flow Diagram

UI Flow diagram tells how user will perceive different interface on click of different buttons or trigger. The rectangular blocks represent entities and Arrow represents change of view from one to another on the bases of button clicked mentioned near arrow.UI flow Diagram of DoS is given below figure 3.3 -:

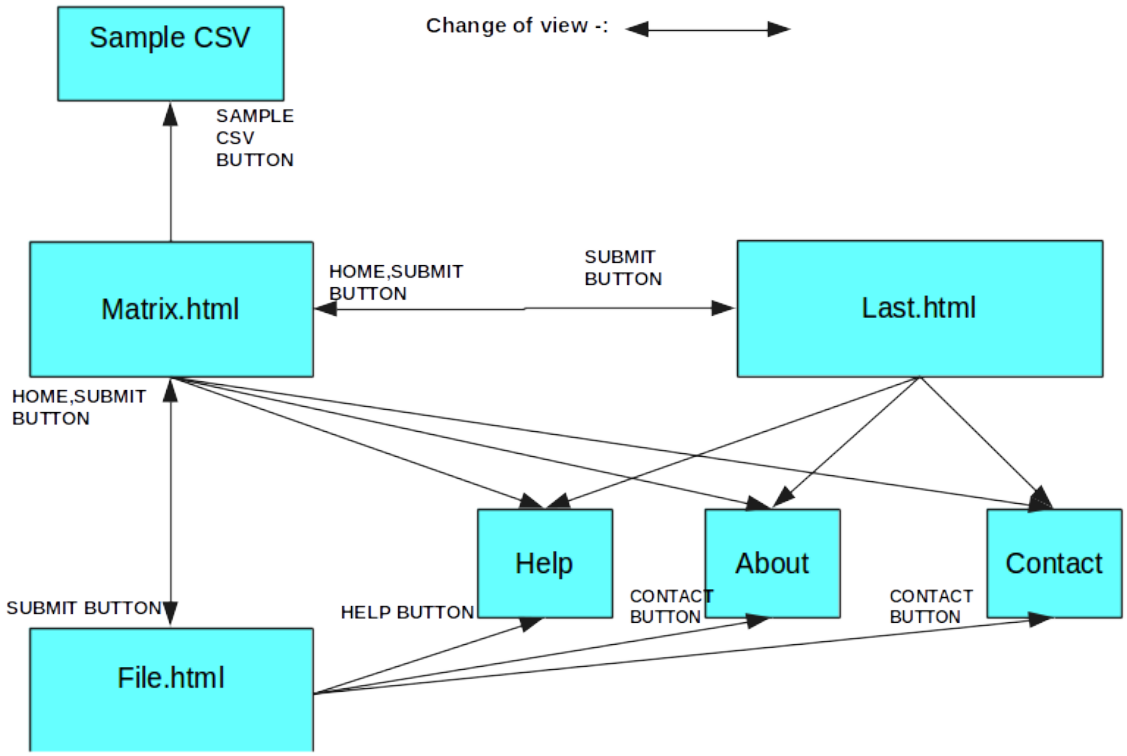


Figure 3.3: UI Flow diagram

3.4 Flowchart

A flowchart is a type of diagram that represents an algorithm, work flow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows and following are flowchart of DoS showing flow of control and Data in the software-:

Figure 3.4 give flow of whole system with flow of mentioned modules explained at -:

1. view.matrix() 3.4
2. initialfile.emailpdf() 3.5
3. view.last() at figure 3.6
4. view.file() at figure 3.7
5. view.pdfemail() at figure 3.8

Here, In figure 3.5

1. Initial input represent all initial input value given in home page
2. MLS represent all Mass, Height(level), Stiffness matrix

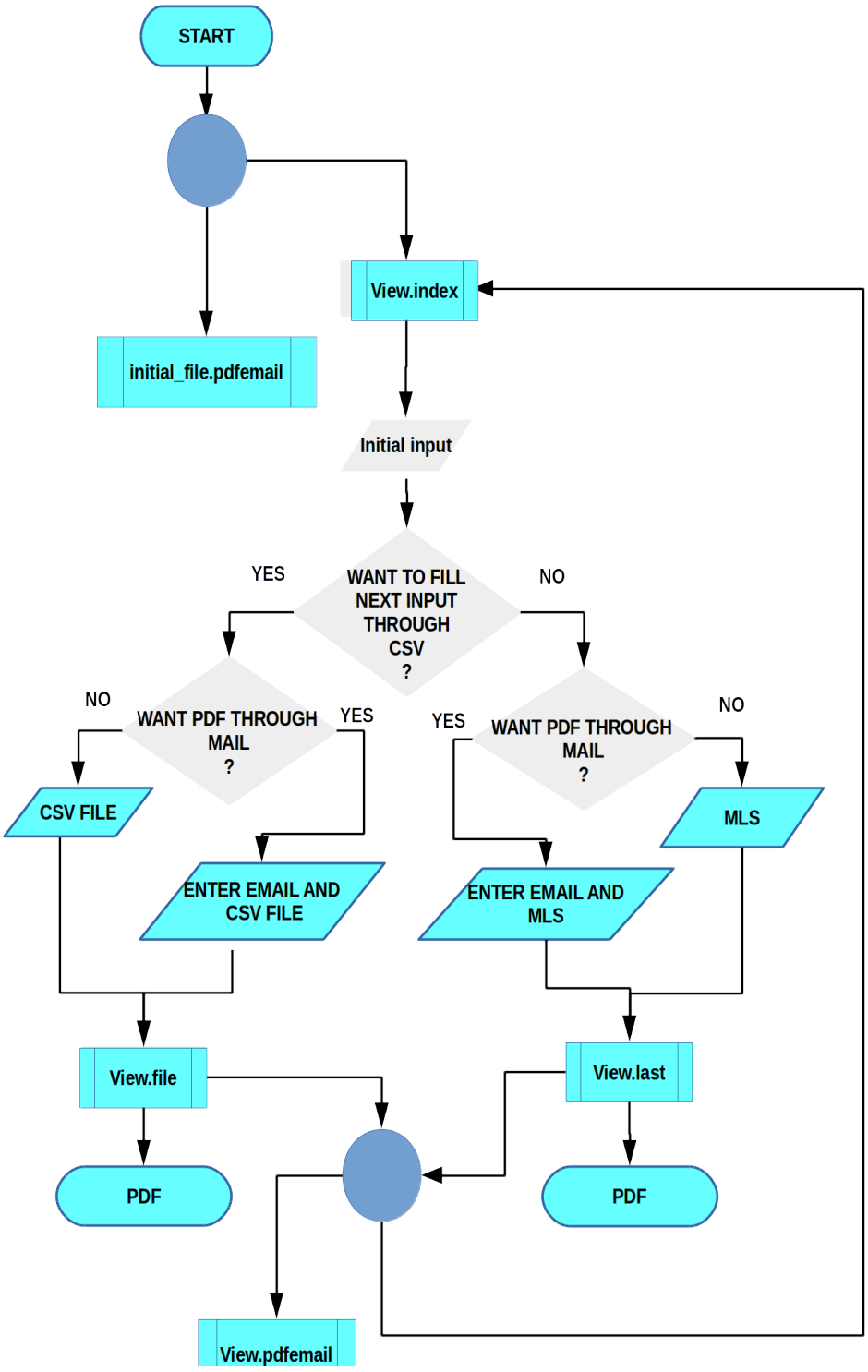


Figure 3.4: Flowchart of Whole System

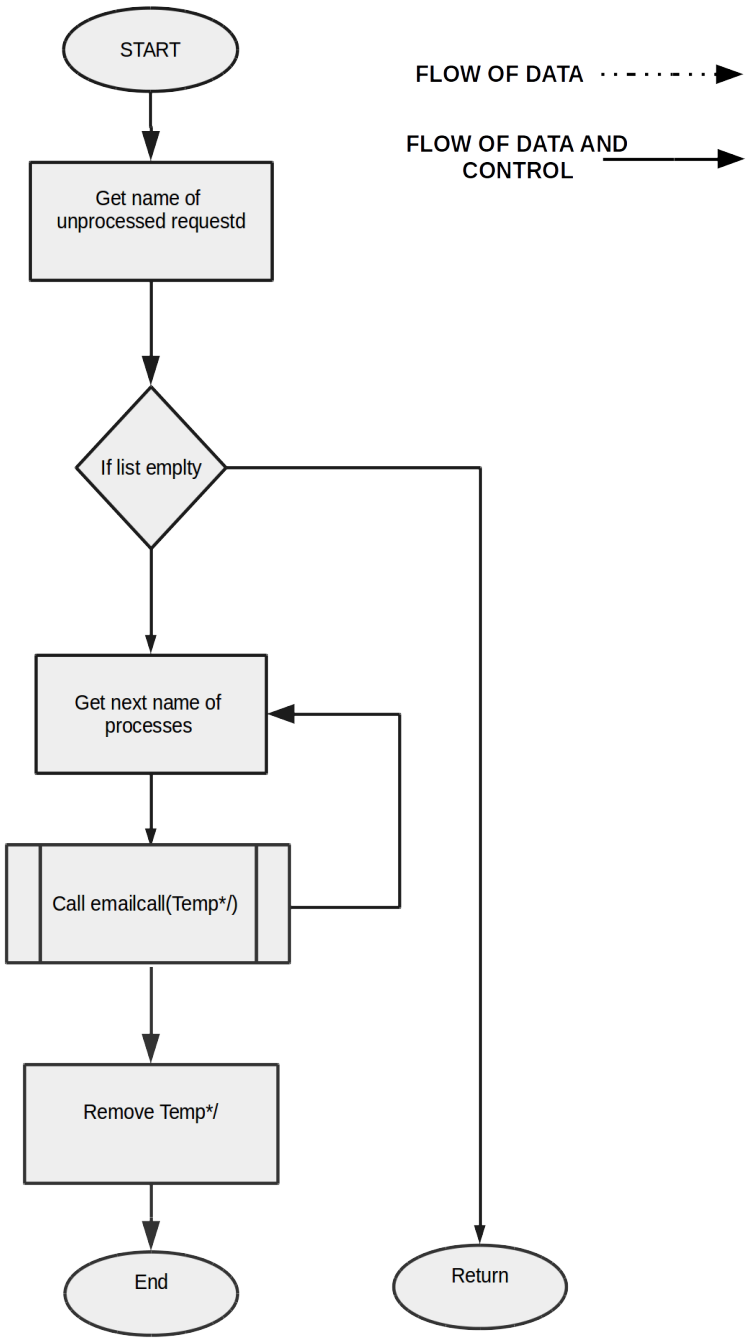


Figure 3.5: Flowchart of initialfile.pdfemail()

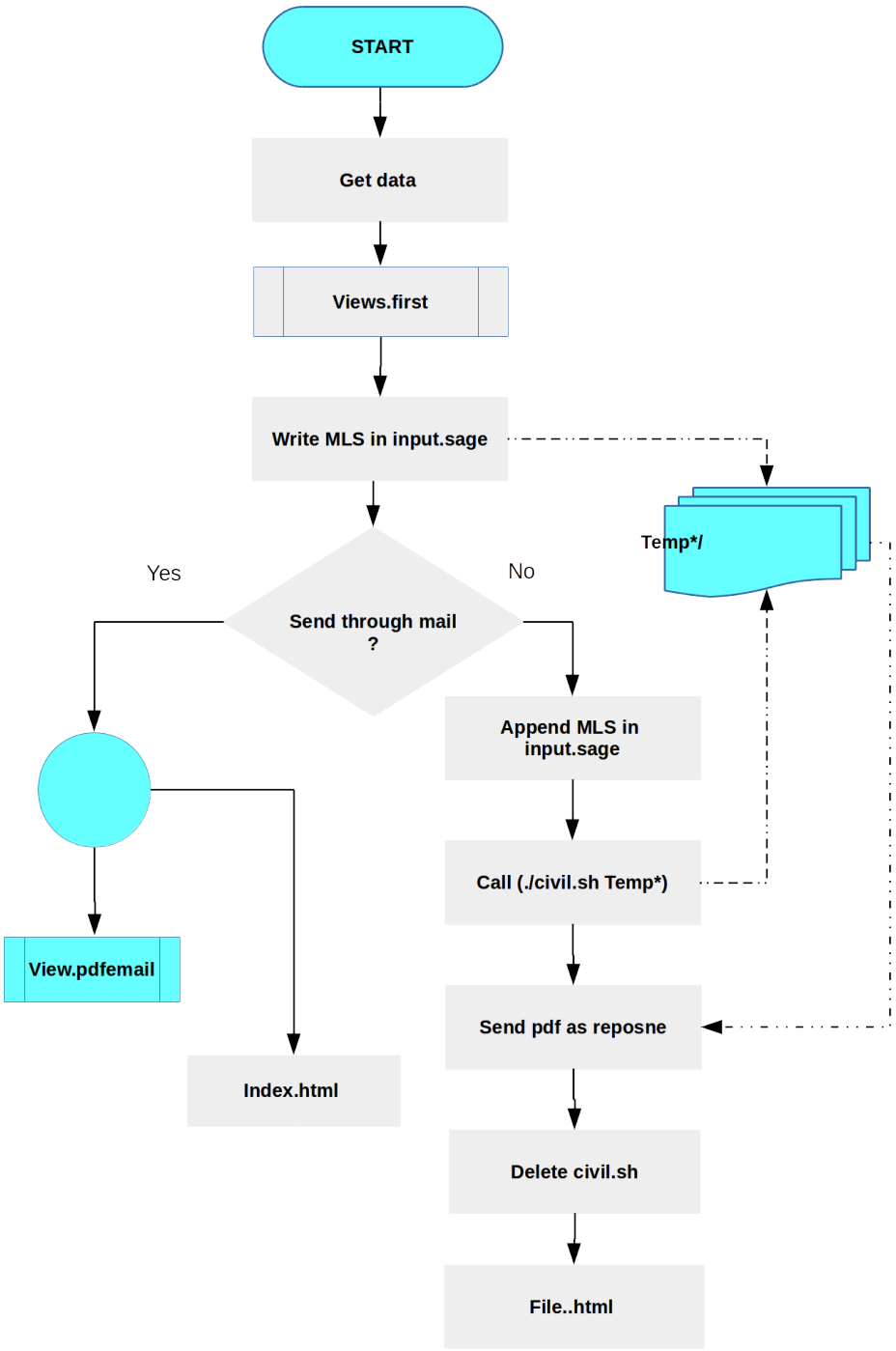


Figure 3.6: Flowchart of veiw.last()

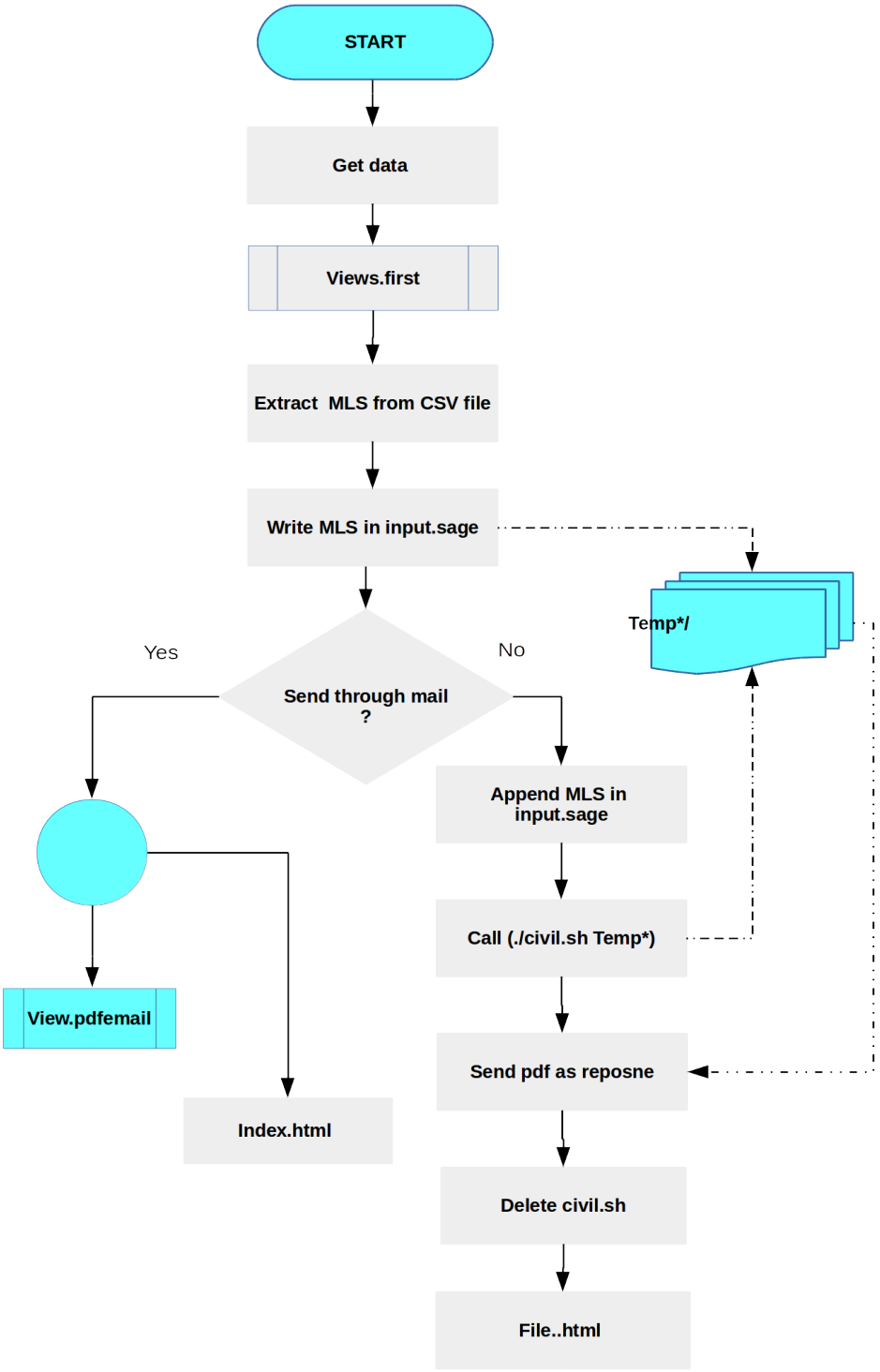


Figure 3.7: Flowchart of veiw.file()

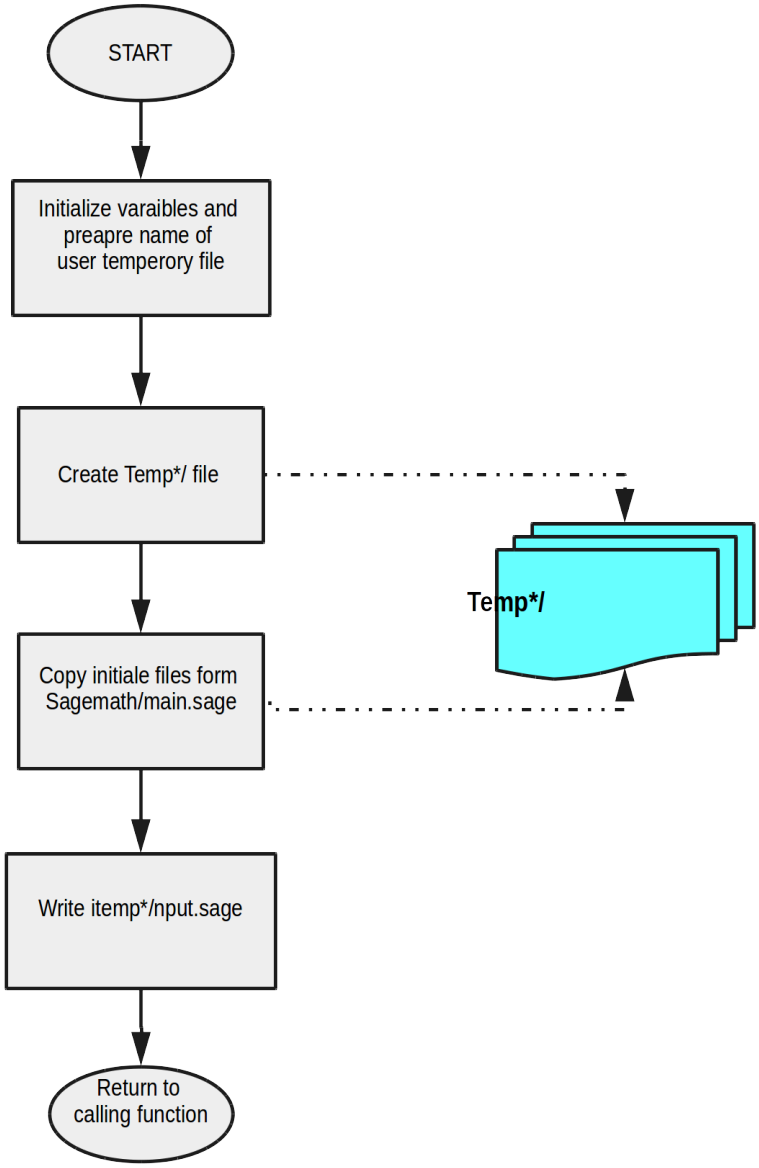


Figure 3.8: Flowchart of veiw.pdfemail()

3.5 Dependencies

Dependencies include softwares or framework that need to be installed for proper working of this software.

1. Programming language: Python 2.7
2. Software: SAGEMATH, LaTeX
3. Framework: Django 1.7
4. Operating System: Any on which above dependencies can be installed

CHAPTER 4

DEVELOPMENT AND IMPLEMENTATION

4.1 Python

Figure 4.1: Python logo

Python is a dynamic language, as in python coding is very easy and also it require less coding and about its interpreted nature it is just exellent. Python is a high level programming language and Django which is a web development framework is written in python language.

Python is an easy to learn, powerful programming language.Python runs on Windows, Linux/Unix, Mac OS X. Python is free to use, even for commercial products. Python can also be used as an extension language for existing modules and applications that need a programmable interface. Python is free to use, even for commercial products, because of its OSI-approved open source license.

4.1.1 Features of Python

- Very clear, readable syntax.
- Strong introspection capabilities.
- Intuitive object orientation.
- Natural expression of procedural code.
- Full modularity, supporting hierarchical packages.
- Exception-based error handling.
- Very high level dynamic data types.

- Extensive standard libraries and third party modules for virtually every task.
- Extensions and modules easily written in C, C++ (or Java for Jython, or .NET languages for IronPython).
- Embeddable within applications as a scripting interface.

4.1.2 Installation of Python

Installation of python is a very easy proccess. The current python versions are: Python 2.7.1 and Python 3.2. Type the commands in the terminal:

$ wget http://www.python.org/ftp/python/2.7/Python-2.7.tgz

$ tar xzf Python-2.7.tgz

This will install the python on your pc/laptop.

4.2 Front End Languages and Framework [1]

Front End languages are language that are used to give better user experince and user interface. These mainly include HTML, CSS, Javascript. Some Frameforks like Bootstrap are also used with these basic languages.

4.2.1 HTML

Figure 4.2: HTML5 logo

HyperText Markup Language, commonly referred to as HTML, is the standard markup language used to create web pages. Along with CSS, and JavaScript, HTML is a cornerstone technology, used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications. Web browsers can read HTML files and render them into visible or audible web pages. HTML describes the structure of a website semantically along with cues for presentation, making it a markup language, rather than a programming language.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured

[1] Used in project but not by me accept basic HTML

documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

4.2.2 CSS

Figure 4.3: CSS logo

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language.Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers. CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions

```
p {
    color: red;
```

```
    text-align: center;
}
```

4.2.3 Javascript

Figure 4.4: Javascript logo

JavaScript (/dvskrpt/) is a high-level, dynamic, untyped, and interpreted programming language. It has been standardized in the ECMAScript language specification. Alongside HTML and CSS, it is one of the three essential technologies of World Wide Web content production; the majority of websites employ it and it is supported by all modern web browsers without plug-ins. JavaScript is prototype-based with first-class functions, making it a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles. It has an API for working with text, arrays, dates and regular expressions, but does not include any I/O, such as networking, storage or graphics facilities, relying for these upon the host environment in which it is embedded.

4.2.4 BootStarp

Figure 4.5: BootStrap logo

Bootstrap is a free and open-source collection of tools for creating websites and web applications. It contains HTML and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. It aims to ease the development of dynamic websites and web applications.

Bootstrap is a front end framework, that is, an interface for the user, unlike the server-side code which resides on the "back end" or server.

4.3 Shell Scripting

Normally shells are interactive. It means shell accept command from you (via keyboard) and execute them. But if you use command one by one (sequence of 'n' number of commands) , the you can store this sequence of command to text file and tell the shell to execute this text file instead of entering the commands. This is know as shell script. Shell script defined as series of command written in plain text file. Shell script is just like batch file is MS-DOS but have more power than the MS-DOS batch file. why to Write Shell Script ?

1. Shell script can take input from user, file and output them on screen.
2. Useful to create our own commands.
3. Save lots of time.
4. To automate some task of day today life.
5. System Administration part can be also automated.

Execute your script as syntax:

```
chmod 755 your-script-name
sh your-script-name
./your-script-name
```

4.4 Introduction to LaTeX

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

Figure 4.6: Donald Knuth, Inventor Of TeX typesetting system

Within the typesetting system, its name is styled as LaTeX. The term LaTeX 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 LaTeX, a .tex file must be created using some form of text editor. While most text editors can be used to create a LaTeX document, a number of editors have been created specifically for working with LaTeX.

LaTeX 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, LaTeX is used because of the high quality of typesetting achievable by TeX. 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.

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

4.4.1 Typesetting

LaTeX 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 LaTeX document, the author specifies the logical structure using familiar concepts such as chapter, section, table, figure, etc., and lets the LaTeX system worry about the presentation of these structures. It therefore encourages 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}
```

4.5 Introduction to Django

Figure 4.7: Django logo

Django is an open source web application framework written in python. It lets you build high-performing, elegant Web applications quickly. Django focuses on automating as much as possible. Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "pluggability" of components, rapid development, and the DRY principal. Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Django takes it's name from the early jazz guitarist Django Reinhardt, a gypsy savant who managed to play dazzling and electrifying runs on his instrument even though two of the fingers on his left hand were paralyzed in an accident when he was young.

Thus its a fitting name for the framework. Django can do some very complex things with less code and a simpler execution than youd expect. It doesn't take a heavy hand to build with Django. The framework does the repetitive work for you, allowing you to get a working website up quickly and easily.

4.5.1 Features of Django

- Clean URLs
- Object- Relational Mapping
- Loosely coupled components
- Designer-friendly templates
- Cache framework
- MVC architecture
- Jython support
- DRY (Don't Repeat Yourself)

4.5.2 Installation of Django

Installation of Django is very easy. To install Django version 1.4, type the following commands:

```
$ wget http://www.djangoproject.com/download/1.4/tarball

$ tar xzvf Django-1.4.tar.gz

$ cd Django-1.4

$ sudo python setup.py install
```

This will install the django on your system.

4.5.3 MTV

Django adopts the standard MVC (Model-View-Controller) design pattern. But instead, their naming convention is the MTV (Model-Template-View).

- **Model** is an object relational mapping to your database schema. So each model is a class which represents a table in your database. Django models provide easy access to an underlying data storage mechanism, and can also encapsulate any core business logic, which must always remain in effect, regardless of which application is using it. Models exist independent of the rest of the system, and are designed to be used by any application that has access to them. In fact, the database manipulation methods that are available on model instances can be utilized even from the interactive interpreter, without loading a Web server or any application-specific logic.
- **Template** is simply HTML for your views. It also allows you to display different messages depending on whether or not a user is logged in. Templates are Django's provided way of generating text-based output, such as HTML or emails, where the people editing those documents may not have any experience with Python. Therefore, templates are designed to avoid using Python directly, instead favoring an extensible, easy-to-use custom language built just for Django.
- **View** could be a homepage or a page to display a user's information, for instance. A view accepts user input, including simple requests for information; behaves according to the application's interaction logic; and returns a display that is suitable for user's to access the data represented by models.

4.5.4 Creating Prject in Django

If this is your first time using Django, you'll have to take care of some initial setup. Namely, you'll need to auto-generate some code that establishes a Django project- a collection of settings for an instance of Django, including database configuration, Django-specific options and application-specific settings. From the command line, cd into a directory where you'd like to store your code, then run the command

```
$ django-admin.py startproject mysite
```

This will create a mysite directory in your current directory.

4.5.5 Development Server in Django

Change into the outer mysite directory, if you haven't already, and run the command

```
$ python manage.py runserver
```

You'll see the following output on the command line:

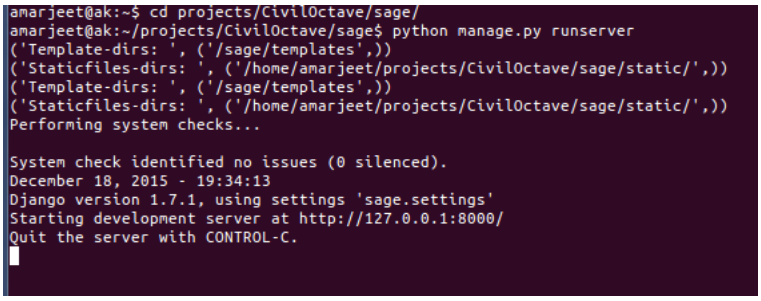


Figure 4.8: Output of runserver

4.5.6 Database setup

In this, we need to edit the settings.py file of the Project, that is the configuration file. It's a normal Python module with module-level variables representing Django settings. Change the following keys in the DATABASES 'default' item to match your database connection settings.

- ENGINE – Either 'django.db.backends.postgresql_psycopg2', 'django.db.backends.mysql', 'django.db.backends.sqlite3' or 'django.db.backends.oracle'. Other backends are also available.
- NAME – The name of your database. If you're using SQLite, the database will be a file on your computer; in that case, NAME should be the full absolute path, including filename, of that file. If the file doesn't exist, it will automatically be created when you synchronize the database for the first time (see below). When specifying the path, always use forward slashes, even on Windows (e.g. C:/homes/user/mysite/sqlite3.db).
- USER – Your database username (not used for SQLite).
- PASSWORD – Your database password (not used for SQLite).
- HOST – The host your database is on. Leave this as an empty string if your database server is on the same physical machine (not used for SQLite).

If you're new to databases, we recommend simply using SQLite by setting ENGINE to 'django.db.backends.sqlite3' and NAME to the place where you'd like to store the database. SQLite is included as part of Python 2.5 and later, so you won't need to install anything else to support your database.

While you're editing settings.py, set TIME_ZONE to your time zone. The default value is the Central time zone in the U.S. (Chicago).

Also, note the INSTALLED_APPS setting toward the bottom of the file. That holds the names of all Django applications that are activated in this Django instance. Apps can be used in multiple projects, and you can package and distribute them for use by others in their projects.

By default, INSTALLED_APPS contain the following apps, all of which come with Django:

- django.contrib.auth – An authentication system.
- django.contrib.contenttypes – A framework for content types.
- django.contrib.sessions – A session framework.

- django.contrib.sites – A framework for managing multiple sites with one Django installation.
- django.contrib.messages – A messaging framework.
- django.contrib.staticfiles – A framework for managing static files.

These applications are included by default as a convenience for the common case.

Each of these applications makes use of at least one database table, though, so we need to create the tables in the database before we can use them. To do that, run the following command:

```
$ python manage.py syncdb
```

The syncdb command looks at the INSTALLED_APPS setting and creates any necessary database tables according to the database settings in your settings.py file. You'll see a message for each database table it creates, and you'll get a prompt asking you if you'd like to create a superuser account for the authentication system. Go ahead and do that.

4.6 Introduction to Doxygen

Figure 4.9: Doxygen logo

Doxygen is a documentation generator, a tool for writing software reference documentation. The documentation is written within code, and is thus relatively easy to keep up to date. Doxygen can cross reference documentation and code, so that the reader of a document can easily refer to the actual code.

Doxygen supports multiple programming languages, especially C++, C, C#, Objective-C, Java, Python, IDL, VHDL, Fortran and PHP.[2] Doxygen is free software, released under the terms of the GNU General Public License.

4.6.1 Features of Doxygen

- Requires very little overhead from the writer of the documentation. Plain text will do, Markdown is support, and for more fancy or structured output HTML tags and/or some of doxygen's special commands can be used.
- Cross platform: Works on Windows and many Unix flavors (including Linux and Mac OS X).
- Comes with a GUI frontend (Doxywizard) to ease editing the options and run doxygen. The GUI is available on Windows, Linux, and Mac OS X.
- Automatically generates class and collaboration diagrams in HTML (as clickable image maps) and LaTeX (as Encapsulated PostScript images).

- Allows grouping of entities in modules and creating a hierarchy of modules.
- Doxygen can generate a layout which you can use and edit to change the layout of each page.
- Can cope with large projects easily.

4.6.2 Installation of Doxygen

Doxygen can be installed using following commands:

```
$ git clone https://github.com/doxygen/doxygen.git
$ cd doxygen
$ ./configure
$ make
```

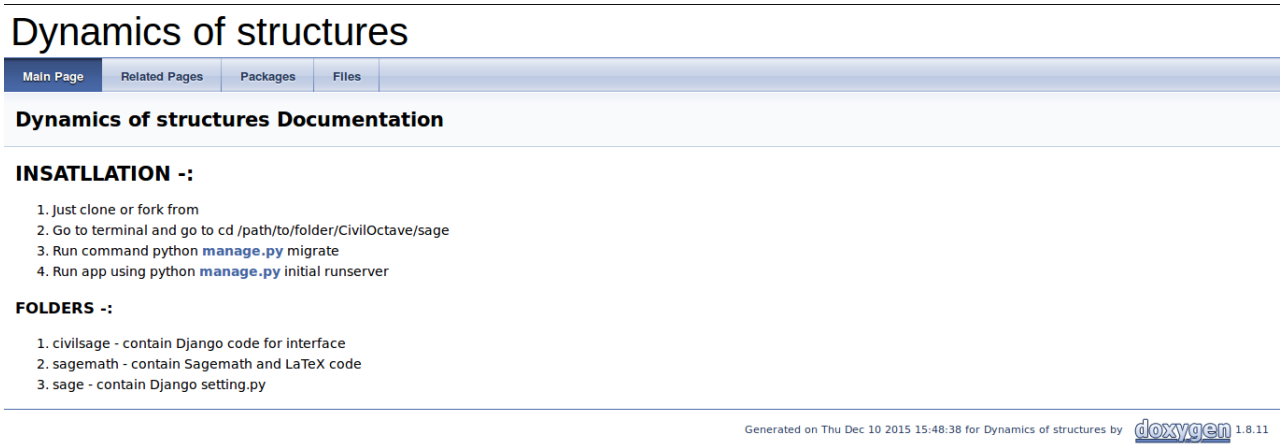


Figure 4.10: Documentation using Doxygen (main page)

Figure 4.11: Doxygen documentation of a function

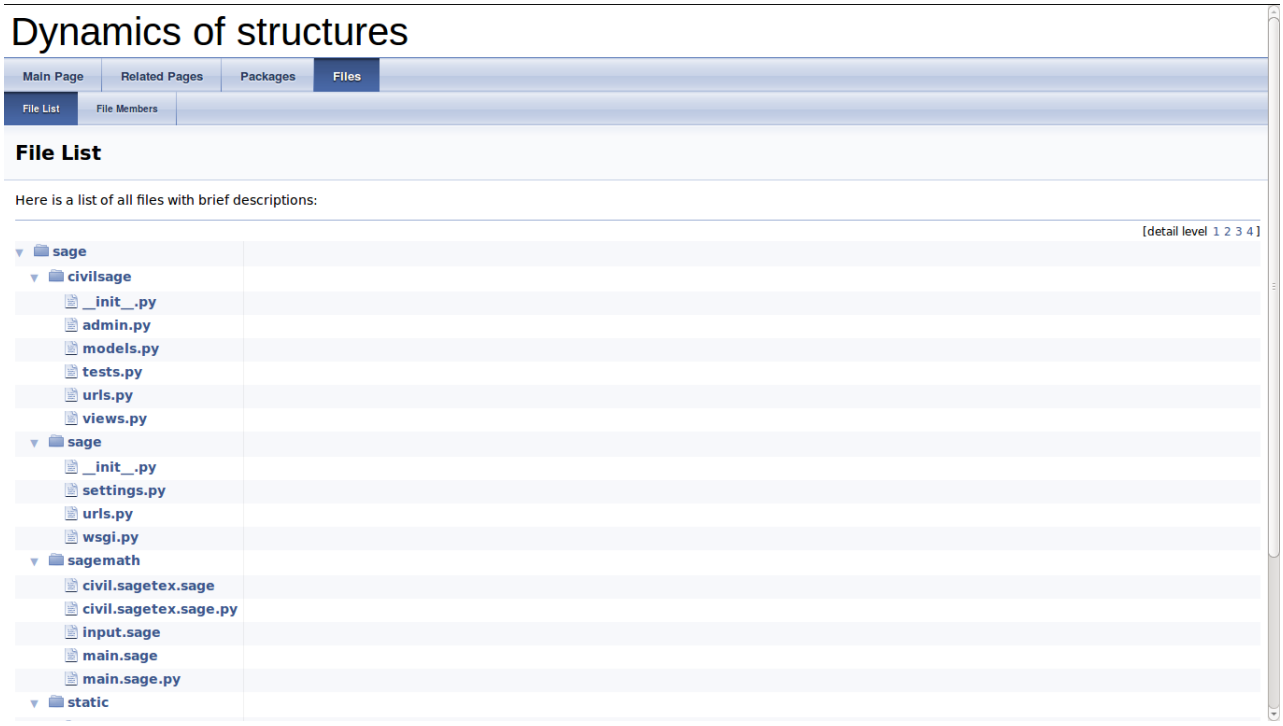


Figure 4.12: Documentation using Doxygen(list of files)

Figure 4.13: Github Logo

4.7 Introduction to Github

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 Code lines.
  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 Work flow.
  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 workwith 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.

4.7.1 What is Git?

Figure 4.14: Git Logo

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 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.

4.7.2 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.

4.7.3 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:-

4.7.3.1 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

4.7.3.2 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

4.7.3.3 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 branchs history into the current branch

$ git branch -d [**branch-name**]
Deletes the specified branch

4.7.3.4 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

4.7.3.5 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

4.8 SageMath

Figure 4.15: SageMath Logo

SageMath (previously Sage or SAGE, System for Algebra and Geometry Experimentation) is mathematical software with features covering many aspects of mathematics, including algebra, combinatorics, numerical mathematics, number theory, and calculus.

The first version of SageMath was released on 24 February 2005 as free and open source software under the terms of the GNU General Public License, with the initial goals of creating an "open source alternative to Magma, Maple, Mathematica, and MATLAB". The originator and leader of the SageMath project, William Stein, is a mathematician at the University of Washington.

SageMath uses the Python programming language, supporting procedural, functional and object-oriented constructs.

4.8.1 Features

The Sage notebook document interface in a web browser. Equation solving and typesetting using the SageMath notebook web interface

Features of SageMath include:

- A browser-based notebook for review and re-use of previous inputs and outputs, including graphics and text annotations. Compatible with Firefox, Opera, Konqueror, Google Chrome and Safari. Notebooks can be accessed locally or remotely and the connection can be secured with HTTPS.
- A text-based command-line interface using IPython
- Support for parallel processing using multi-core processors, multiple processors, or distributed computing
- Calculus using Maxima and SymPy
- Numerical linear algebra using the GSL, SciPy and NumPy
- 2D and 3D graphs of symbolic functions and numerical data
- Matrix manipulation, including sparse arrays
- Multivariate statistics libraries, using R and SciPy
- A toolkit for adding user interfaces to calculations and applications

- Graph theory visualization and analysis tools
- Libraries of number theory functions
- Support for complex numbers, arbitrary precision and symbolic computation
- Technical word processing including formula editing and embedding SageMath within LaTeX documents
- The Python standard library, including tools for connecting to SQL, HTTP, HTTPS, NNTP, IMAP, SSH, IRC, FTP and others
- Interfaces to some third-party applications like Mathematica, Magma, R, and Maple
- Documentation using Sphinx
- An automated test-suite
- Execution of Fortran, C, C++, and Cython code
- Although not provided by SageMath directly, SageMath can be called from within Mathematica as is done in this Mathematica notebook example

4.8.2 Installation From Source Code

Steps for install SageMath -:

- sudo apt-get install g++ lrzip
- lrunzip "Downloaded-Zip-file".tar.lrz
- sudo apt-get dpkg-dev
- cd sageL
- sudo apt-get install binutils gcc make m4 perl tar
- sudo apt-get install build-essential m4
- export MAKE='make -j8'
- sudo apt-get install tk8.5-dev
- cd sage-directory
- ./configure
- make

4.9 Implementation

Development of DoS started with development in phases which focus on particular need of project. Various phases and their detail are given below -:

- Phase I (Sagemath) -:
  During Phase I, we wrote code in Sagemath to compute all the required output variable. We used two files for this.Input variable are written in input.sage and main.sage loads this variable and compute all desired inputs.

- Phase II (LaTeX) -:
  During phase II, we embedded sage modules into LaTeX by loading them into civil.tex and then displayed these variables in output PDF and all this work is done by sagetex.sty. To execute all this commands on one go we wrote script civil.sh.

- Phase III (Django) -:
  During phase III, we provided web interface to this software using Django. Djanog was used to get input from user and write input.sage file for particular user then civil.sh is called by passing name of user directory to it and then get output PDF.

- Phase IV (Improvise) -:
  During phase IV, we improved the code structure and added additional functionality like sending PDF as email and accepting input as CSV file. Finally, the UI was improved and made responsive.

- Phase V (Testing) -:
  During phase V, we tested the software for various conditions and then applied required error control and messaging mechanism. initialfile.py file was created to save software from problem of server restart which can causes processing user request to stop. so, that the interrupted request of user can be restart and send PDF.

- Phase VI (Documentation) -:
  During final phase, we documented the project(developers documentation and README.md) using doxygen and wrote the report for this software.

This is the first interface that is shown to the user. This is the Home page. It brings a nice UI to get input from user about the Number of storeys and some factor affecting the structure. It also provides the user some additional features as an aid like uploading data as CSV files and Email after completion. There is an additional help icon near every field to get an idea of what it is. There is a static navigation bar that provides an Auto fill values option. By clicking that option, default values are automatically filled figure 4.16 and figure 4.17.

After clicking the 'Submit' button on the homepage, user is shown another form based on his input "Number of Storeys" in the homepage form, for filling out the matrix values manually Figure 4.18.

The top navigation bar is static to all pages. There is a link to the Help. After clicking which, the help regarding the different inputs fields provided Figure 4.19.

There are small icons near each of the input field on the homepage. They are the help icons. After hovering them, they display a little help related to that correspond field Figure 4.20.

After running a set of commands of LaTeXand SageMath, it produces the output in PDF form (with pdflatex) Figure 4.21.

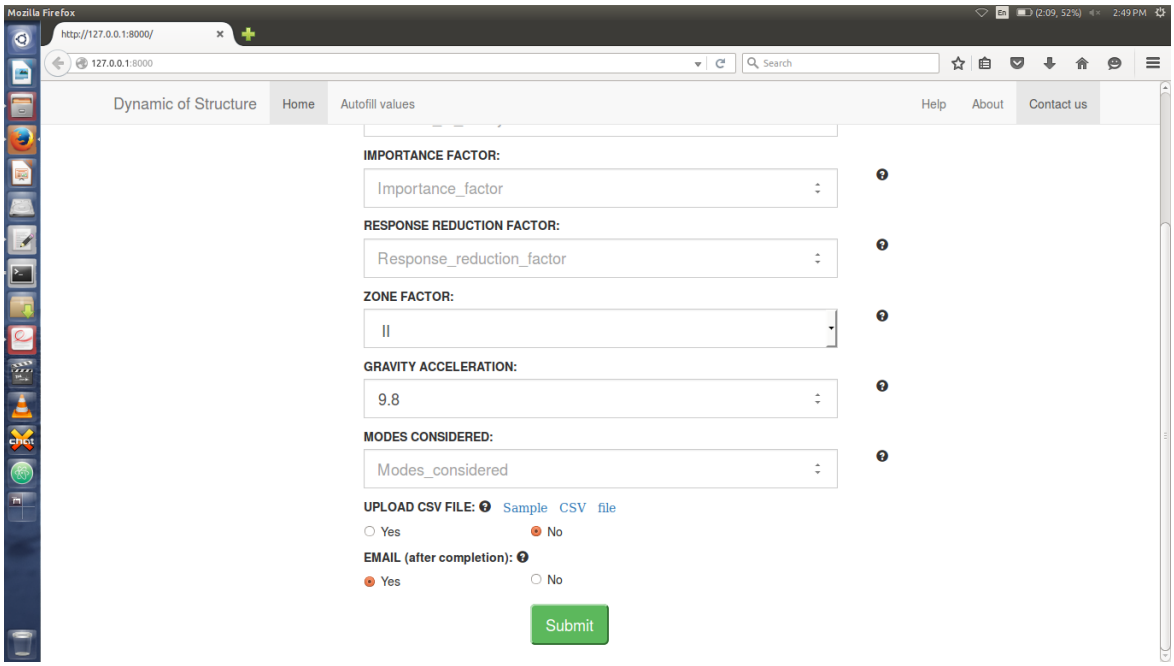


Figure 4.16: Home page of DoS

After the titlepage, table of content and some other stuff, the PDF contains the values (matrices) entered in the form at the homepage Figure 4.22.

There is graph embedded in the PDF which is being created using Sagemath Figure 4.23.

After all calculations, the computed final result is printed on the PDF Figure 4.24.

4.10 Testing

Testing a program consists of providing the program with a set of test inputs (or test cases) and observing if the program behaves as expected. If the program fails to behave as expected, then the conditions under which failure occurs are noted for later debugging and correction.

This software had been taken through rigorous test to fully found potential causes of error and system failure and full focus have been given to cover all possible exceptions that can occur and cause failure of the software. As this software is based on intensive background process it have been taken care that if correct input and email address are given then processing of user job can even continue or a least automatically restart even after server shuts down or even crash.

Following is data collected Table no. 4.5 and corresponding Graph figure 4.25 for analyzing the computation time taken by software to its maximum supported value at this time-:

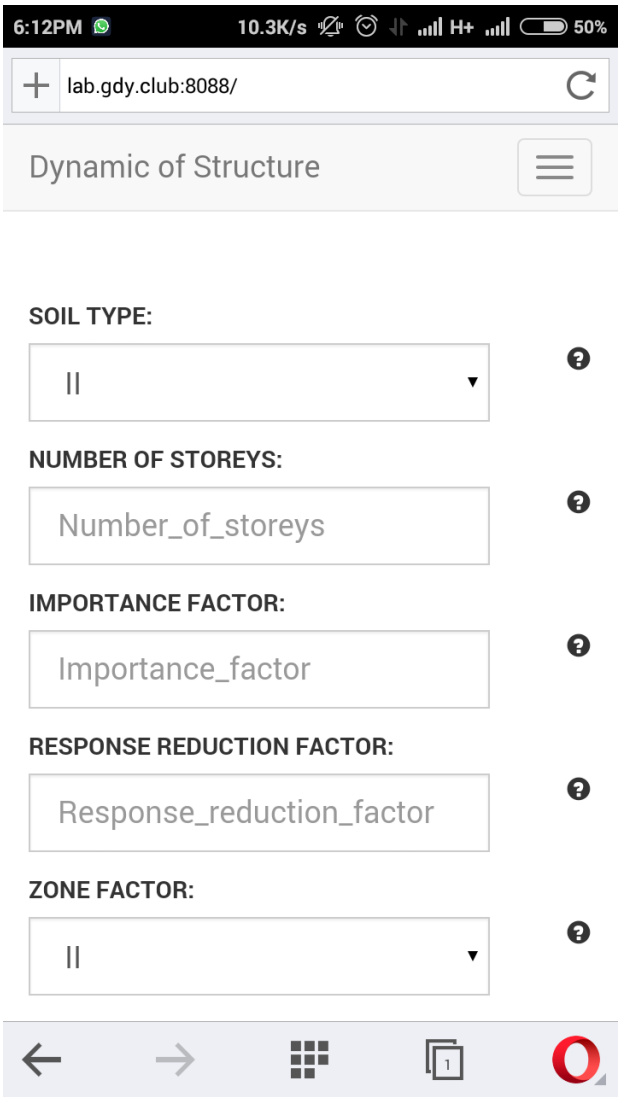


Figure 4.17: Home page of DoS(mobile view)

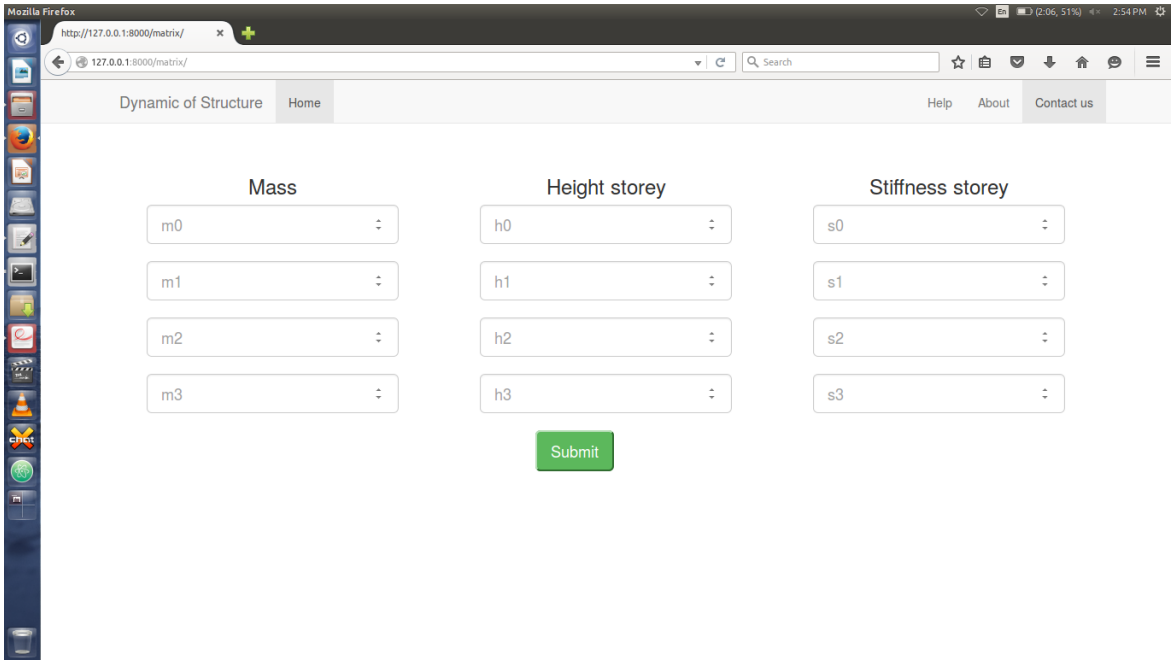


Figure 4.18: Matrix.html for manually filling values

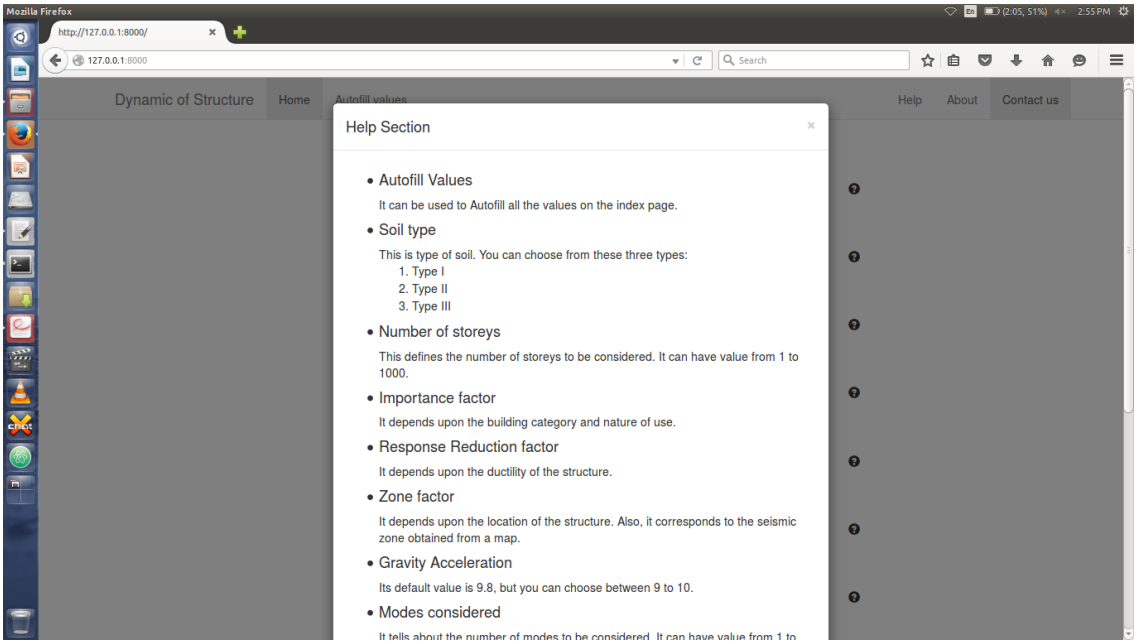


Figure 4.19: Help section in Home page

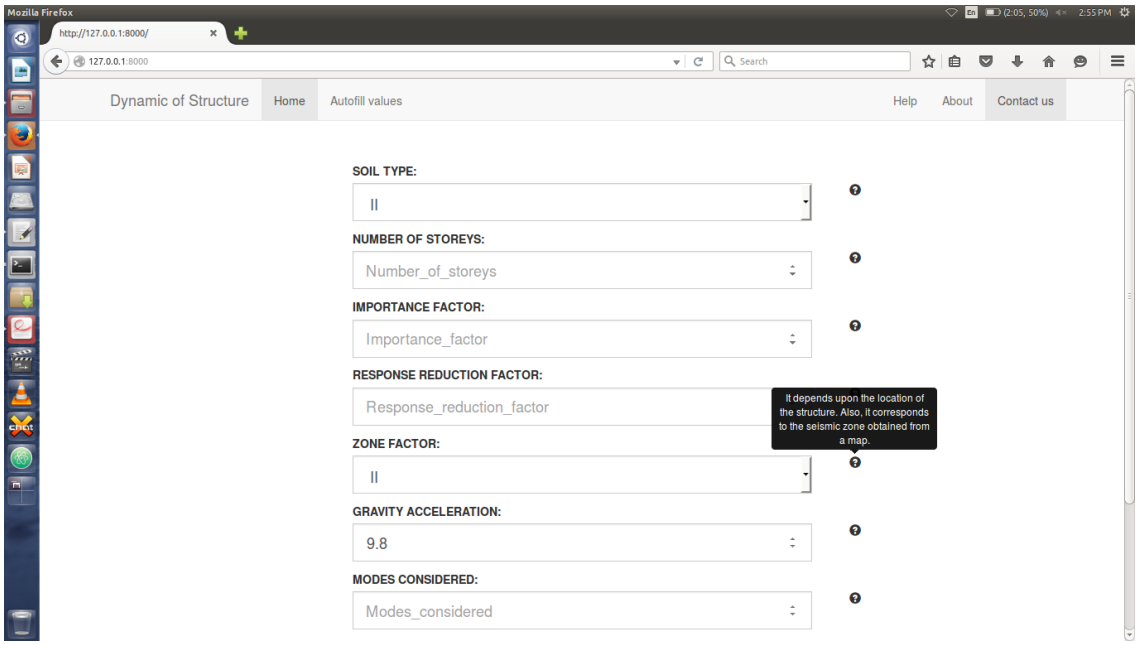


Figure 4.20: Local help option

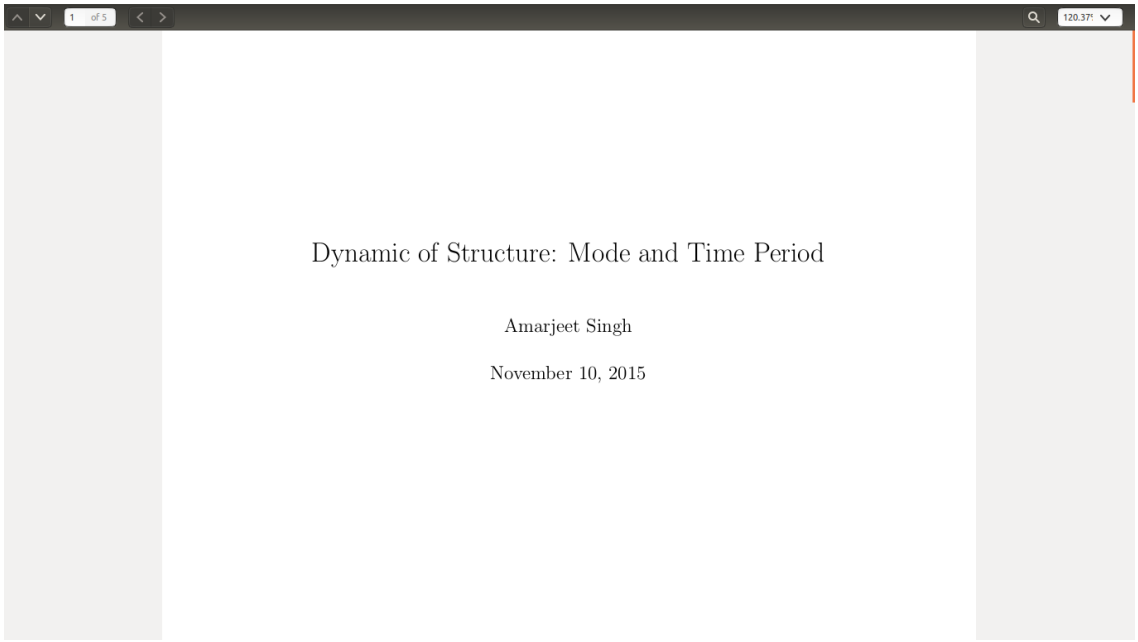
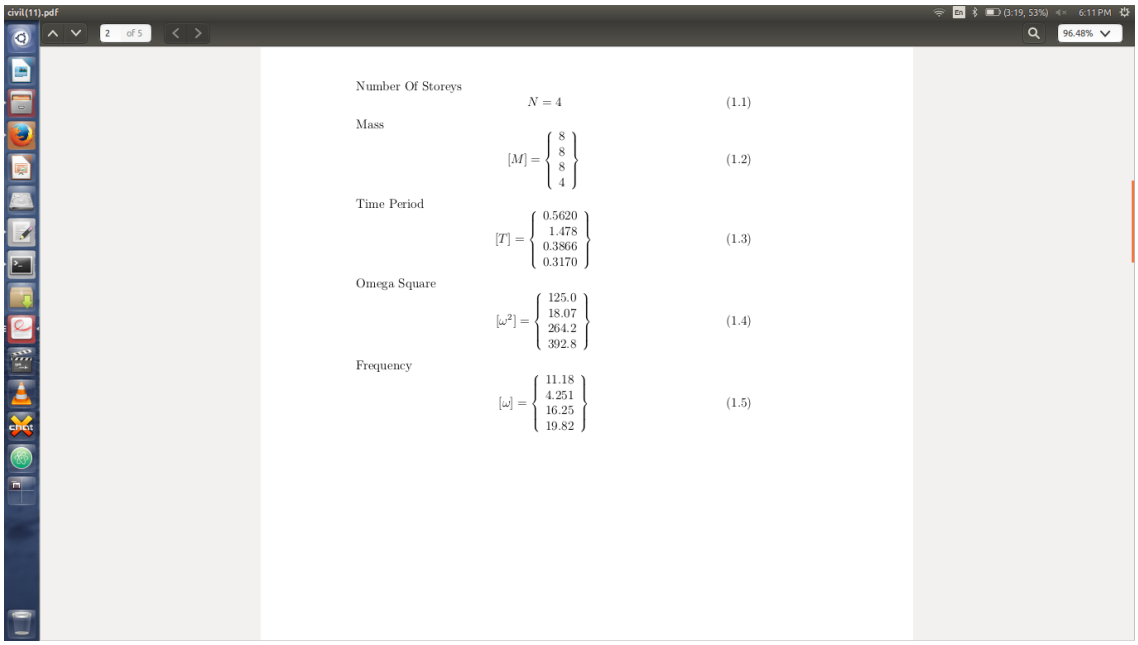


Figure 4.21: First page of PDF generated by DoS



Number Of Storeys

$$N = 4 \tag{1.1}$$

Mass

$$[M] = \begin{Bmatrix} 8 \\ 8 \\ 8 \\ 4 \end{Bmatrix} \tag{1.2}$$

Time Period

$$[T] = \begin{Bmatrix} 0.5620 \\ 1.478 \\ 0.3866 \\ 0.3170 \end{Bmatrix} \tag{1.3}$$

Omega Square

$$[\omega^2] = \begin{Bmatrix} 125.0 \\ 18.07 \\ 264.2 \\ 392.8 \end{Bmatrix} \tag{1.4}$$

Frequency

$$[\omega] = \begin{Bmatrix} 11.18 \\ 4.251 \\ 16.25 \\ 19.82 \end{Bmatrix} \tag{1.5}$$

Figure 4.22: Initial values given for checking in PDF

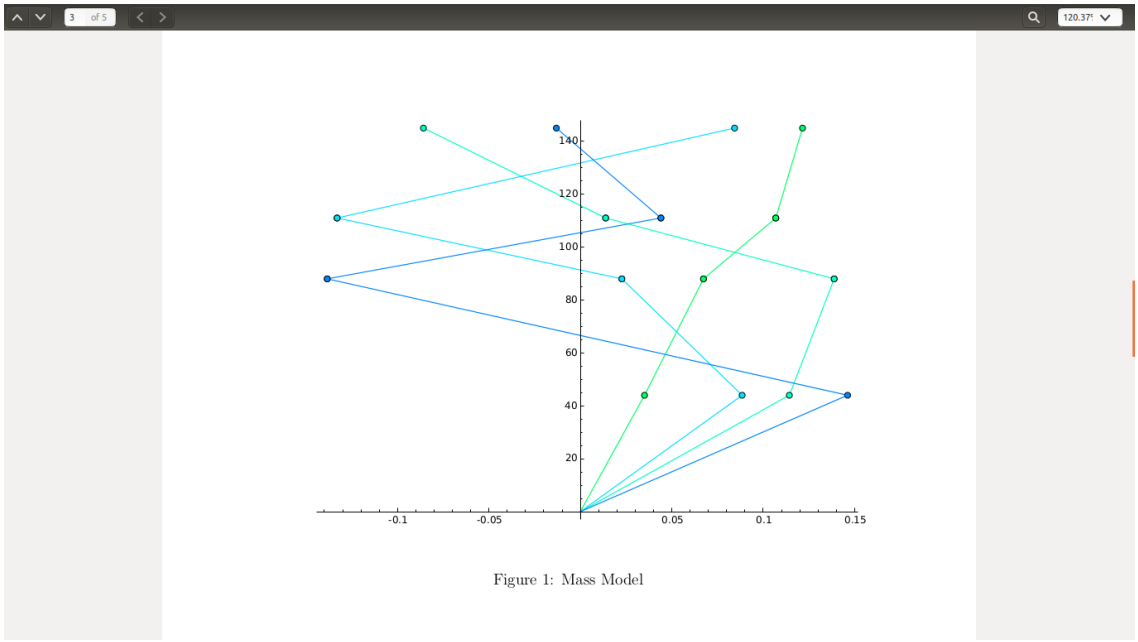
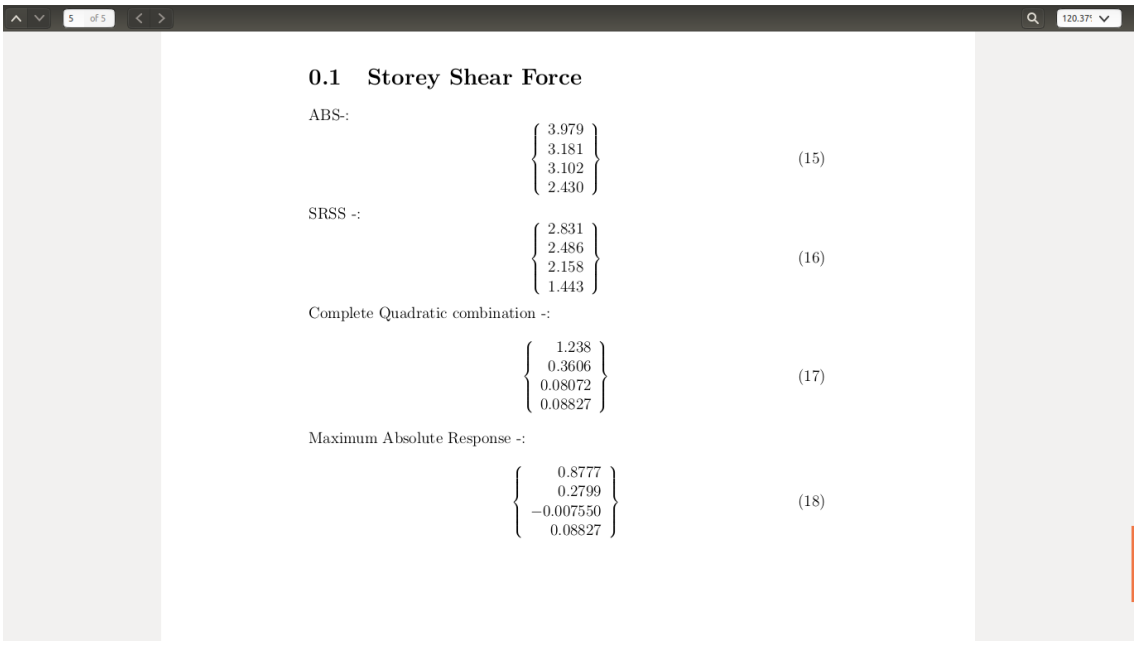


Figure 1: Mass Model

Figure 4.23: Graph Generated in PDF



0.1 Storey Shear Force

ABS-:

$$\begin{Bmatrix} 3.979 \\ 3.181 \\ 3.102 \\ 2.430 \end{Bmatrix} \quad (15)$$

SRSS -:

$$\begin{Bmatrix} 2.831 \\ 2.486 \\ 2.158 \\ 1.443 \end{Bmatrix} \quad (16)$$

Complete Quadratic combination -:

$$\begin{Bmatrix} 1.238 \\ 0.3606 \\ 0.08072 \\ 0.08827 \end{Bmatrix} \quad (17)$$

Maximum Absolute Response -:

$$\begin{Bmatrix} 0.8777 \\ 0.2799 \\ -0.007550 \\ 0.08827 \end{Bmatrix} \quad (18)$$

Figure 4.24: Final output in PDF

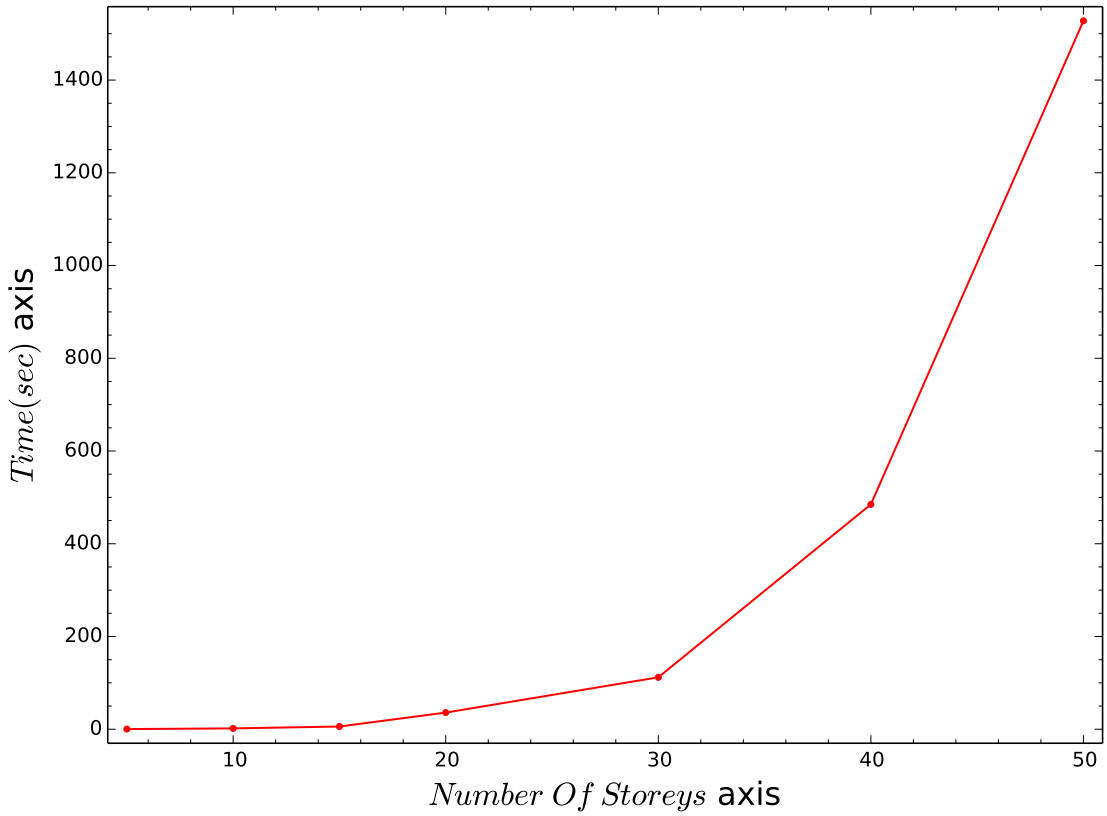


Figure 4.25: Time complexity graph of Dynamics of Structure

Time complexity analysis table	
Number of Storeys	Time(sec)
5	0.341143
10	1.768803
15	5.827224
20	35.870348
30	111.939167
40	484.906122
50	1527.764697

Table 4.1: Computational analysis of DoS

Test cases for main page(index.html)			
Input	Desired Output	Actual Output	Status
Inputs range exceeds	Alert user,Don't proceed	Alert user,Don't proceed.	Pass
Any one or more field left empty	Alert user about range. Don't proceed	Alert user about range. Don't proceed	Pass
CSV selected: No	Direct to manual fill of matrices	Direct to manual fill of matrices	Pass
CSV selected: Yes	Direct to uploading of file	Direct to uploading of file	Pass
Email selected: Yes	Show email field after Submit	Show email field after Submit	Pass
Email selected: No	Proceed without email	Proceed without email	Pass
Help pressed	Show Detailed user help	Show Detailed user help	Pass
Autofill values	Automatically fill default values	Automatically fill default values	Pass

Table 4.2: Computaional anaysis of DoS

Test cases for manual fill(matrix.hmtl)			
Input	Desired Output	Actual Output	Status
Email Address not filled	Give PDF after all processing	Give PDF after all processing	Pass
Email Address filled and Submit	Show message, redirect to homepage	Show message, redirect to homepage	Pass
Input range exceeds or not filled	Show error message	Show error message	Pass

Table 4.3: Tests for manual fill

Test cases for file fill(file.html)			
Input	Desired Output	Actual Output	Status
Email Address not filled	Give PDF after all processing	Give PDF after all processing	Pass
Email Address filled and Submit	Show message, redirect to homepage	Show message, redirect to homepage	Pass
Input range exceeds or not filled	Show error message	Show error message	Pass
Wrongly formatted CSV file	Give error message with Possible errors	Give error message with Possible errors	Pass

Table 4.4: Tests for csv upload page

Test cases for possible source of problems			
Input	Desired Output	Actual Output	Status
URL refreshed	Send to homepage	Send to homepage	Pass
server stops or rebooted	Start processing interrupted requests	Start processing interrupted requests	Pass

Table 4.5: Test case (general)

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

DoS is a very efficient application which help in generating result for analysis of structures. It can be used by Civil Engineers and M.Tech. students and even layman. It's less time consuming and user-friendly which let the User work in batch mode and free him from all the installation process. It is a web application that can be accessed from a number of devices. The responsive User Interface makes it easy for the users to operate it. Many efforts were made to ease the usage for the users. Hence, it is expected to be work properly in different conditions. But any future bug reports or improvements are always welcomed and will be processed happily.

I learn a lot by working on this project. During this period I got to learn a vast number of technologies. These are listed below: Operating system: Ubuntu Language used: Python, HTML, CSS, shellscript Framework: Django Technogloy: LaTeX, Djanog, Doxygen, Sagemath, Git So during this project I learn all the above things. Above all I got to know how software is developed and how much work and attention to details is required in building even the most basic of components of any project. Planning, designing, developing code, working in a team, testing, etc. These are all very precious lessons in themselves. Aside from all above I got go know about various methods like -:

1. Threading the programs
2. Embedding and using different tech in one software.
3. How to work like in group for development of software.
4. How to apply juggaar(innovated) in softwares to get problem solved.

Beside these technology used in project I also get to know some other tech also like -:

1. OpneCV (image processing)
2. opensshserver
3. reveal.md, impress.js (for making presentations)

5.2 Future Scope

This software being a open source have allot of scope for future improvements and additions as other individuals can also contibute in it and add additional functionality like-:

1. output real time graph of modes
2. Give user other anaylises option
3. Add abiltiy to model structor online
4. automating other anaylises techniques

BIBLIOGRAPHY

[1] Dynamics of Structure(DoS), https://github.com/amarjeetkapoor1/CivilOctave

[2] LaTeX https://www.sharelatex.com

[3] Sagemath www.sagemath.org/

[4] Django https://docs.djangoproject.com/

[5] Doxygen www.doxygen.org

[6] My Blog, https://amarjeetkapoor1.wordpress.com

[7] My Github Profile, https://github.com/amarjeetkapoor1