# LearnToCodeApp Documentation

**Will Mason (DjangoWill)**

**Jan 09, 2019**

# Table of Contents:

CHAPTER 1

# About - a summary of the project

The LearnToCodeApp project has been created to provide a focus for learning programming and working collaboratively. Everyone is encouraged to join in, whatever the level of their experience.

The overall aim is to produce office administration software. We intend to produce the software in a modular form, allowing developers to work on and duplicate individual parts of the project. Users should be able to pick and choose functionality by including or omitting modules.

The Core App is a generic module intended to help a business organize itself. In particular, the Core App will:

- Provide a contact database (including contact details)
- Provide a database of projects carried out for clients (which might otherwise be known as “files” or “accounts”).

The Core App will be browser based.

CHAPTER 2

# Contributors to the project

You can find a list of the contributors to the project in the “readme” file. An html version of this file is shown on the Github project home page.

# CHAPTER 3

## Dependencies, tools and technologies we use

h2 – The community

You can join the conversation and ask questions using Discord (discordapp.com) using the following invite: https://discord.gg/BFNsDh2 –

h2 – The technologies and tools

The project uses the following technologies and tools:

h2 – Python 3.

Python is a higher level programming language, meaning it is written in a way humans can read fairly easily. The the code is automatically turned into instructions the computer understands.

Full details about Python can be found here: https://www.python.org/

With a beginners guide here: https://wiki.python.org/moin/BeginnersGuide

Before starting out with a project, please consider setting up a virtual environment. An explanation of how to do this can be found here: https://tutorial.djangogirls.org/en/django_installation/

h2 – Django 2.1.4

Django is a web development framework which uses Python. It has been created to automate some of the processes involved in a dynamic web application. Overall, it makes developing a website quicker and easier.

A web development framework is a way of organising the files that make up a website. A general description can be found here: https://en.wikipedia.org/wiki/Web_framework - a more specific description can be found here: https://jeffknupp.com/blog/2014/03/03/what-is-a-web-framework/

The official Django site is here: https://www.djangoproject.com/

A great tutorial on how to get started is here: https://tutorial.djangogirls.org/en/

h2 – Postgresql

Postgres is an open source database programme. Its official site is here: https://www.postgresql.org/

More information about how to install it and get started can be found here: http://postgresguide.com/

You may also wish to install pgAdmin (https://www.pgadmin.org/) or another database administration programme.

h2 – HMTL5 and CSS3

Short tutorials and information on HTML5, CSS as well as javascript and Python can be found here: https://www.w3schools.com/html/default.asp

h2 – Git

https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

This link shows how to set your username and email in Git: https://help.github.com/articles/setting-your-username-in-git/ You will probably want to use the same username and email as your Github account (at least for this repository) so if you have not yet set up a Github account, you may wish to do so and return to this step.

h2 – Github

Github is a hosting service for Git repositories (https://github.com/). It allows users to host their repositories (groups of files being organised by the Git programme) on the site. Repositories can be private or public (ours is public). As well as uploading their own repository, users can download, comment on, make changes to or "fork" an existing one. A fork is a copy of the original, which can then be changed without changing the original or "master" repository.

Information about how to use Github with Git can be found here: http://rogerdudler.github.io/git-guide/ and here: https://guides.github.com/activities/hello-world/#commit

h2 – Sphinx

We use Python's Sphinx module to help create documentation. The module allows users to save a series of restructured text files explaining the project to users and contributors. Restructured text is similar to HTML markup but is a little bit quicker to write. The module converts the files to full HTML - the html version is hosted on ReadTheDocs.

The official site is here:

http://www.sphinx-doc.org/en/master/index.html

A great tutorial with hints and tips is here: https://media.readthedocs.org/pdf/brandons-sphinx-tutorial/latest/brandons-sphinx-tutorial.pdf

CHAPTER 4

# How to Contribute

## 4.1 How the work will be divided

**The project overall**

The project will comprise of one core application - a primary module - which can be supplemented by additional apps.

The Core App should work independently of any additional modules. The additional modules will add functionality and should work independently of all other apps except the Core App.

The additional modules may duplicate functionality or achieve the same functionality in a different way. For example:

- A module which retrieves information from an outside source may be duplicated by another module which obtains the same information from a different source. Users should be able to choose one or other of the modules, depending on their preference and needs.
- A module may be written to achieve a function in Windows. The module may be 'forked' to do the same thing on a Gnu/Linux distribution.

Initially, we will focus on the Core App. Eventually, we hope contributors will be able to work on separate, compatible apps. When we are at that stage, we will decide whether those apps need their separate repository or can be integrated into the main one.

**The Core App**

Development of the Core App will be divided into the following "work streams":

**Markup**: This will focus on creating the HTML and CSS for the user interface.

**Javascript**: This will focus on adding any functionality to the user interface which is strictly not possible using HTML and CSS.

**Django / Database**: This will focus on two things:

- The Django models required to create the database.
- The structural and administrative elements of the Django framework.

**Python3**: This will focus on server-side scripts and Python functionality.

**Documentation**: This will focus on keeping the documentation up-to-date and easy to understand.

**Security**: This will look at every area of the application to try to improve its security.

The Core App will have two main branches: *Master* and *Development*. Pull Requests should be made to the Development branch. Only once the admins are sure the development branch is in a stable state will the Development branch be merged into the Master branch - this will happen periodically.

## 4.2 How to claim tasks and contribute code

**How to claim tasks**

The people leading the work streams will agree with the admins the specification for their stream. They will then divide up the work and submit an 'Issue' on Github.

Issues will be added to the Projects page for that work stream. Each work stream will have the following columns:

- "Beginner's Tasks"
- "Intermediate Tasks"
- "Claimed Tasks"
- "Completed Tasks/Please Test".
- "Approved/closed tasks".

If you would like to contribute to the project, please check the work streams for tasks you wish to tackle. Once you have chosen, please move the task to the "claimed" column. Please only claim tasks if you believe you have a reasonable chance of completing them within a week. It does not matter if it takes a little bit longer but, in fairness to other collaborators, you should not "reserve" a task that you like the look of.

If there are no tasks you think are suitable, please consider doing one of the following:

- Check the "Beginner's Tasks" or "Intermediate Tasks" pages of the documentation. We will aim to give examples of things you can do to get a feel for the project.
- Review and comment on pull requests or Issues. Pull requests should relate to an Issue that has been raised. You can check the specification for the Issue and compare it to the Pull Request. Pull Requests relating to your preferred work stream will be listed in the "Please Test" column. Please refer to the "How to Review a Pull Request" page.
- Review/test existing code. This is an educational project so it is unlikely that all code will be perfect. In addition, the comments in the code may not be complete and/or may be unclear. You can suggest changes to existing code or the comments by submitting an Issue.
- Suggest functionality. If you have experience of working in a firm and/or using similar software, you might be able to suggest functionality, either for the Core App or for a new module. This should be done by submitting an Issue.

**How to contribute code**

Once you have claimed a task, please follow this workflow:

1. *Create a branch of the development branch*. If submitted by an admin, the Issue should have a brief title. Please use this title as the name of the branch, so others can easily find it.
2. *Clone the branch*. You will need to clone the branch to your own machine in order to work on it.
3. *Write the code*.

4. *Test*. Please do your best to test that the code works. We will try to prepare guides about how to do this but in the meantime, you should be able to find information online. You will, for example, want to test markup using more than one brower and may need to test it using older versions of those browers.
5. *Commit*. Commit the code to your branch.
6. *Create a pull request*. Create a request to merge the changes into the *development* branch. Please then remember to move the
7. *Deal with any comments on the pull request*. Once you have submitted the Pull Request, you should check to see whether it receives any comments or change requests. Pull Requests will not be merged into the code until these are responded to or dealt with.
8. *Close the branch*. Your task-specific branch should only last as long as it takes to deal with the task.

CHAPTER 5

# How to Review a Pull Request

h2 – For admins

Move the task to the “approved/closed tasks” column in the Projects tab.

# CHAPTER 6

## Jargon list

::glossary:

```
Branch
   A copy of the project's files. Before working on the project's files, users should␣
↪create a "branch" of the project and "clone" that to their own computer. If the␣
↪contributor is successful, the branch will eventually be merged into the master␣
↪branch.

Core App
   the primary module of the project, with which all other apps must be compatible.

DB
   "database". The database we use is Postgresql

Projects tab
   The fourth tab along under the headings on this page: https://github.com/
↪LearnToCodeApp-Group/LearnToCodeApp

Pull Request
    A request on Github (using the Pull Request tab) for the contributor's code to be␣
↪added to the master branch.

UI
   User Interface. This is the user facing part of the software - the "screens" or␣
↪pages that the user interacts with. In a website it would be the webpage, for␣
↪Firefox it would be the Toolbar and settings pages etc.

Views
   A web page generated by the Django templating system. In other words, it is an␣
↪HTML webpage seen by either a user of the programme or a client (if client login in␣
↪enabled).

Work stream
   One of the main workflows for the project. To see how these are divided, please␣
↪see the "How to Contribute" page.
```

CHAPTER 7

# Licence for use

The current licence can be found in the “license.md” file in the file tree.

CHAPTER 8

# Submit a change to the repository

Most contributions you make to the project will involve making a change to the code. The changes will not be automatically incorporated within the shared version of the project (the "Master" branch).

Instead, you make a change to a copy of the repository (a "branch") and suggest the change is included in the shared version (a "pull request").

Since this is central to the task, you may as well get started straight away. If you have no other changes to suggest, you can add your name or username to the list of collaborators in the "readme" file.

You will need to do the following:

Make sure you have a virtual environment set up and are using it.

Make sure you have Python3 installed.

Make sure you have Git installed in the virtual environment.

"branch" the main repository. You will need to click on the button which currently says "main" and give the branch a name.

Clone the branch to your computer.

Make a change to the files on your computer.

Add then commit the changes to your local copy of the repository.

Push the changes to the remote copy of the repository (your branch).

Submit a pull request.

CHAPTER 9

# Suggest an Issue

Github allows collaborators and third parties to suggest improvements to a project. Improvements might be:

- Something the software does that it should not or, conversely, that it does not do but it should (a bug)
- Something the software does not do but could be included (an improvement or "feature request")
- A piece of code which is not explained well enough in the documentation.

Suggesting an improvement is quite a simple way of getting involved.

First, you will need to check the Issues page on Github to check whether the issue has already been raised.

Then, you will need to choose the most appropriate template for your issue: A 'bug report' or 'feature request'.

Use the template to describe the issue.

Lastly, submit the issue.

CHAPTER 10

# Make a change to the documentation

We use Python's Sphinx module to help create documentation.

The official site is here:

http://www.sphinx-doc.org/en/master/index.html

A great tutorial with hints and tips is here: https://media.readthedocs.org/pdf/brandons-sphinx-tutorial/latest/brandons-sphinx-tutorial.pdf

h2 – The task

Keeping the documentation up-to-date is key to helping us work together. It also helps beginners join in, which is a significant part of this project.

You can help by updating the documentation. If you are not sure what to change, try checking the "Issues" page on Github or the Discord threads to see if there are any tasks or jargon words which have caused people problems. Please check the existing structure of the documentaion to see where your explanation best fits (eg, the "jargon" page).

CHAPTER 11

# Javascript Aims and Standards

CHAPTER 12

# Markup Aims and Standards

h2 – HTML

h2 – CSS

CHAPTER 13

# Python Aims and Standards

CHAPTER 14

# Core App Specifications and List of Functionality

The core app will provide the primary user interface. Allow other apps to fill space on that interface.

Welcome to the LearnToCodeApp documentation. Please read the “about” section or our Github “readme” for more information.

CHAPTER 15

# Indices and tables

- genindex
- modindex
- search