**1.0 Introduction:**

The basic idea of this system is to understand the CRUD(Create, Retrieve, Update, Delete) operations with Django that uses Python framework to work at it’s core.

**1.1 About Django:**

Django is a Web Application Framework which is used to develop web applications. Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. The Django is very demanding due to its rapid development feature. It takes less time to build application after collecting client requirement. By using Django, we can build web applications in very less time. Django is designed in such a manner that it handles much of configure things automatically, so we can focus on application development only.  
  
**1.1.1 System that uses Django:**

Django is widely accepted and used by various well-known sites such as:

* Instagram
* Mozilla
* Disqus
* Pinterest
* Bitbucket
* The Washington Times

**1.1.2 Features of Django:**

* Rapid Development
* Secure
* Scalable
* Fully loaded
* Versatile
* Open Source
* Vast and Supported Community

**2.0 CRUD Operations:**

Within computer programming, the acronym CRUD stands for create, read, update and delete. These are the four basic functions of persistent storage. Also, each letter in the acronym can refer to all functions executed in relational database applications and mapped to a standard HTTP method, SQL statement or DDS operation.

It can [also describe](https://softwareengineering.stackexchange.com/questions/120716/difference-between-rest-and-crud) user-interface conventions that allow viewing, searching and modifying information through computer-based forms and reports. In essence, entities are read, created, updated and deleted. Those same entities can be modified by taking the data from a service and changing the setting properties before sending the data back to the service for an update. Plus, CRUD is data-oriented and the standardized use of HTTP action verbs.

**3.0 About this project:**

This is a small Django project to demonstrate Django CRUD functionality, it consists of 3 small applications:  
  
- books\\_cbv: Implement CRUD using CBV (Class Based Views).  
- books\\_fbv: Implement CRUD using FBV (Function Based Views).  
- books\\_fbv\\_user: add user interaction to books\\_fbv example.

The Django CRUD project only need a single Python package "Django", it was built and tested with Django 2.x version. To install it use the following command:

pip install -r requirements.txt

This will install all the Python modules required for this project to run. Post installation move to directory and run:

manage.py migrate

This will move all the SQLITE3 files to system to have it start the table and database migration for Django to use. After that the development server can begin to run with simple command:

manage.py runserver

And can be accessed via a basic browser with the link:

<http://localhost:8000/>

To perform operations, one requires a Superuser (Or to create a non-superuser) which can be achieved by the following set of steps:

manage.py createsuperuser

To create non-superuser a superuser can visit a the following page and make the accounts. create it at: http://localhost:8000/admin/

**4.0 How Project Works:**

All the files are inter-related and to have a flow we will have to make them into Objects with Connections, looking for the files here’s how it is works:

**4.0.1 Manage.py**

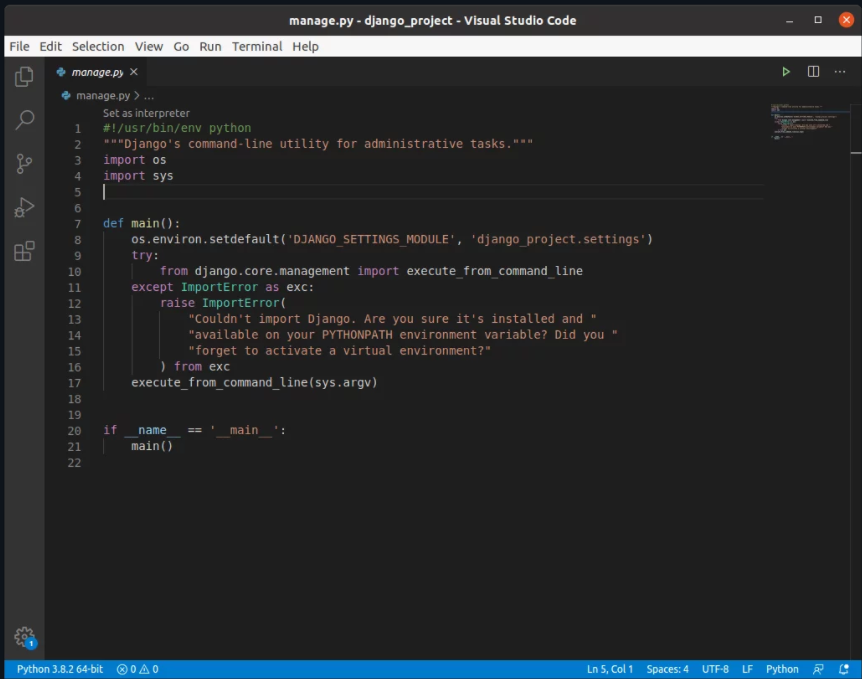
This file is used basically as a command-line utility and for deploying, debugging, or running our web application.

This file contains code for runserver, or makemigrations or migrations, etc. that we use in the shell. Anyway, we do not need to make any changes to the file.

• runserver: This command is used to run the server for our web application.

• Migration: This is used for applying the changes done to our models into the database. That is if we make any changes to our database then we use migrate command. This is used the first time we create a database.

• Makemigration: this is done to apply new migrations that have been carried out due to the changes in the database.



**4.0.2 \_init\_.py**

This file remains empty and is present their only to tell that this particular directory (in this case django\_project) is a package. We won’t be doing any changes to this file as well.

**4.0.3 settings.py**

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

**4.0.4 Urls.py**

Just like the project urls.py file, this file handles all the URLs of our web application. This file is just to link the Views in the app with the host web URL. The settings urls.py has the endpoints corresponding to the Views

**4.0.5 Models.py**

This file contains the models of our web applications (usually as classes).Models are basically the blueprints of the database we are using and hence contain the information regarding attributes and the fields etc of the database.

**4.0.6 Views.py**

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

**4.0.7 tests.py**

This file contains the code that contains different test cases for the application. It is used to test the working of the application.