**Django Framework Integration with Angular**

**Django**is an open-source framework for back-end web applications based on **Python**— one of the top web development languages. **Django**is a collection of Python libraries allowing you to quickly and efficiently create a quality Web application, and is suitable for both front-end and back-end. Its main goals is simplicity, flexibility, reliability, and scalability.

**Django**has its own naming system for all functions and components (e.g., HTTP responses are called “views”). The key features include:

1. Simple syntax;
2. Its own web server;
3. MVC (Model-View-Controller) core architecture;
4. “Batteries included” (comes with all the essentials needed to solve solving common cases);
5. An ORM (Object Relational Mapper);
6. HTTP libraries;
7. Middle-ware support; and
8. A Python unit test framework.

Additionally, **Django**provides a dynamic CRUD (create, read, update and delete) interface, configured with admin models and generated via introspection. CRUD is used to describe the basic database commands, which means the interface facilitates viewing, changing, and searching for information.

## **Why Django ?**

1) It is fast and simple

One of **Django**’s main goals is to simplify work for developers. To do that, the Django framework uses:

* The principles of rapid development, which means developers can do more than one iteration at a time without starting the whole schedule from scratch;
* DRY philosophy — don’t repeat yourself — which means developers can reuse existing code and focus on the unique one.

As a result, it takes a lot less time to get the project to market.

### **2) It is secure**

Security is also a high priority for Django. It has one of the best out-of-the-box security systems out there, and it helps developers avoid common security issues, including

* clickjacking,
* cross-site scripting, and
* SQL injection.

**Django**promptly releases new security patches. It’s usually the first one to respond to vulnerabilities and alert other frameworks.

### **3) It suits any web application project**

With **Django**, you can tackle projects of any size and capacity, whether it’s a simple website or a high-load web application. **Why use Django for your project?** Because:

* It’s fully loaded with extras and scalable, so you can make applications that handle heavy traffic and large volumes of information;
* It is cross-platform, meaning that your project can be based on Mac, Linux or PC;
* It works with most major databases and allows using a database that is more suitable in a particular project, or even multiple databases at the same time.

### **4) It is well-established**

**Django**is time- and crowd-tested. It has a big, supportive community accessed through numerous forums, channels, and dedicated websites. It’s easy to find help when there’s a problematic function in the code, and to find developers if your company is looking to base the next project on Django.

**Django**started off with great documentation, the best of any other open-source framework. And it’s still maintained on a high level, updated along with the new functions and fixes, so you can easily adapt to changes.

You can trust that any issues with the framework will be solved as soon as they arise. The software is constantly updated and new packages are released to make working with Django more convenient than it already is.

## **What you can do with Django?**

Here’s an interesting fact: Django was first created to power a web application for a newspaper publisher, the [**Lawrence Journal-World**](https://www2.ljworld.com/). You can expect it to be amazing at handling projects with volumes of text content, media files, and heavy traffic — or anything else that works like a web-based periodical.

But the publishing industry is not the framework’s only area of application. Django is also used to build e-commerce websites and health care and financial applications for transportation and booking, social media sites, and more. Here are some of the many project types you can develop using the framework:

* Financial platforms with features for analyzing and calculating approximate results based on personal data, risk tolerance, the probability of achieving goals;
* Built-in custom CRM systems for internal data;
* B2B CRM systems for handling communication between businesses;
* Platforms that facilitate communication between two parties, such as a business and a consumer;
* High-load booking engines or shopping platforms;
* Android and iOS mobile apps that support web applications;
* Real estate property evaluation systems;
* Document management systems;
* Platforms for handling legal issues like verifying mortgage conditions or lease status.

Some companies choose to base their projects on more than one framework. Django can also be used to create separate features, such as:

* An emailing system for sending notifications to users;
* A filtering system with advanced logic and dynamically changing rules;
* Algorithm-based generators;
* Data-analysis tools;
* Interfaces for managing investment funds;
* Admin dashboards;
* Photo-based verification systems;
* And other features that facilitate the development of CRM and B2B platforms, online marketplaces, booking systems, and more.

## **Angular and Django integration:**

**Requirements:** Django > 1.8, Angular 8, 9 or above. I've done this on Angular 9.

Let's jump directly into a pool, I'm guessing, you already have a basic Django setup.

- Vritualenv setup

- Django project

- Static files and template files setup

Start by installing angular cli into your local.

npm install -g @angular/cli

If you are on windows', please set the path of the ng command. Otherwise, you can use **npm run ng <your ng command>** for the rest of the article.

1. In **Django**, create a Landing view and URL pointing to that view. And render a template which you're going to use for Angular. Let's say **angular\_index.html** which is residing into **Django**templates dir.
2. Now, from command-line, create a new angular project names front-end into a **Django**static directory. ng new front-end
3. Now your Angular app's basic structure is done, for testing, run **ng build**. This will compile your code and produce some files into dist directory which will be created into a front-end (your angular app)` directory.
4. Now go into **angular\_index.html**and include these compiled files runtime-es2015.js, polyfills.js, styles-es2015.js, vendor-es2015.js, main-es2015.js into Django template js block.

[**https://gist.github.com/PranamBhat/4bc2d27032a06916cf92229234c0260a**](https://gist.github.com/PranamBhat/4bc2d27032a06916cf92229234c0260a)

And now you're almost done, just add

<body>

<app-root></app-root>

</body>

into **angular\_index.html** and run your python server by

**python manage.py runserver**

Go to your defined URL for your angular app and you'll be seeing the app.

Up until now, everything is working fine, but the problem is every time you do some code change, you have run ng build to compile the angular code and then see the changes reflect in the browser.

So you might be thinking, why not use ng serve, yes ng serve is a great option. But the sad part is, it does not provide us with the compiled files. it saves those files in the memory. So we need to use watch option in ng build for development purpose.

Run **ng build --watch** into a terminal and it will hot reload your project with every file change. But now, your browser will not show anything. Because at watch state, ng build --watch produces different filenames without post-fix of es2015. So you need to change those files names in the angular\_index.html and include the following files.

[**https://gist.github.com/PranamBhat/c5b84294775a00b5fbf647c7687fc35b**](https://gist.github.com/PranamBhat/c5b84294775a00b5fbf647c7687fc35b)

For better configuration, you can get the environment from **settings.py** and with the help of if condition, you can set the files according to the environment.

Just in case, if you add support for IE, there will extra file which will generate. You need to add that file to render everything perfectly.

## **Famous companies that use the Django framework:**

* [**YouTube**](https://www.youtube.com/)
* [**Instagram**](https://www.instagram.com/)
* [**Quora**](https://www.quora.com/)
* [**Spotify**](https://www.spotify.com/)
* [**National Geographic**](https://www.nationalgeographic.com/)
* [**The Washington Post**](https://www.washingtonpost.com/)
* [**Dropbox**](https://www.dropbox.com/)
* [**Mozilla**](https://www.mozilla.org/en-US/)
* [**NASA**](https://www.nasa.gov/)
* [**Pinterest**](https://www.pinterest.com/)
* [**Reddit**](https://www.reddit.com/)
* [**Bitbucket**](https://bitbucket.org/)

**Conclusion:**

**Django**is a great addition to projects that need to handle large volumes of content (e.g., media files), user interactions or heavy traffic, or deal with complex functions or technology (e.g., machine learning). Yet it is simple enough for smaller-scale projects, or if you intend to scale your project to a much higher level. That’s why **Django**is used by so many companies that vary in size and goals.

**Examples:**

1) [**https://shriniket.home.blog/2019/09/22/integrate-angular-js-with-django-filter-application/**](https://shriniket.home.blog/2019/09/22/integrate-angular-js-with-django-filter-application/)

2) [**https://medium.com/swlh/django-angular-4-a-powerful-web-application-60b6fb39ef34**](https://medium.com/swlh/django-angular-4-a-powerful-web-application-60b6fb39ef34)