<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Project Generation Options</td>
<td>3</td>
</tr>
<tr>
<td>2 Getting Up and Running Locally</td>
<td>5</td>
</tr>
<tr>
<td>3 Settings</td>
<td>7</td>
</tr>
<tr>
<td>4 Included gulp tasks</td>
<td>9</td>
</tr>
<tr>
<td>5 FAQ</td>
<td>11</td>
</tr>
<tr>
<td>5.1 Why is there a django.contrib.sites directory in Cookiecutter Django Gulp?</td>
<td>11</td>
</tr>
<tr>
<td>6 Troubleshooting</td>
<td>13</td>
</tr>
<tr>
<td>7 Indices and tables</td>
<td>15</td>
</tr>
</tbody>
</table>
A Cookiecutter template for Django.

Contents:
Project Generation Options

**project_name [Project Name]:** Your Human Readable project name, including any capitalization or spaces.

**project_slug [project_slug]:** The slug of your project, without dashes or spaces. Used to name your repo and in other places where a Python-importable version of your project name is needed.

**author_name [Your Name]:** You! This goes into places like the LICENSE file.

**email [Your email]:** Your email address.

**description [A short description of the project.]:** Used in the generated README.rst and other places.

**domain_name [example.com]:** Whatever domain name you plan to use for your project when it goes live.

**version [0.1.0]:** The starting version number for your project.

**timezone [UTC]:** Used in the common settings file for the `TIME_ZONE` value.

**windows [n]:** Whether you’ll be developing on Windows.

**use_python3 [y]:** By default, the Python code generated will be for Python 3.x. But if you answer `n` here, it will be legacy Python 2.7 code.

**open_source_license [1]:** Select a software license for the project. The choices are:

1. MIT
2. BSD
3. GPLv3
4. Apache Software License 2.0
5. Not open source
CHAPTER 2

Getting Up and Running Locally

The steps below will get you up and running with a local development environment. We assume you have the following installed:

- pip
- virtualenv
- PostgreSQL

First make sure to create and activate a virtualenv, then open a terminal at the project root.

Then install the requirements for your local development:

```bash
$ pip install -r requirements/local.txt
```

Then, create a PostgreSQL database with the following command, where `[project_slug]` is what value you entered for your project’s `project_slug`:

```bash
$ createdb [project_slug]
```

_cookiecutter Django_ uses the excellent _django-environ_ package, which includes a DATABASE_URL environment variable to simplify database configuration in your Django settings. Rename env.example to .env to begin updating the file with your own environment variables. To add your database, define DATABASE_URL and add it to the .env file, as shown below:

```ini
DATABASE_URL="postgres://<pg_user_name>:<pg_user_password>@127.0.0.1:<pg_port>.../<pg_database_name>"
```

You can now run the usual Django `migrate` and `runserver` commands:

```bash
$ python manage.py migrate
$ python manage.py runserver
```

**Setup your email backend**

To send email you need to configure your email backend

In development emails are printed to the console.
Integrate Gulp to your project

If you’d like to take advantage of common frontend development tools, you can do so with the included Gulpfile. Make sure that nodejs is installed. Then in the project root run:

```
$ npm install
$ gulp
```

The base app will now run as it would with the usual `manage.py runserver` but with:

- Live reloading
- Sass compilation, CSS concatenation and compression
- JavaScript validation, concatenation and compression
- Images optimization

all enabled.

Optimized static files are generated in a `dist` folder in your static files folder. To serve them in your project, you can add something like this in your HTML: `<link href="{% static 'css/project.min.css' %}" rel="stylesheet">`

To read about all included gulp tasks see `Included gulp tasks`.

It’s time to write the code!!!
This project relies extensively on environment settings which **will not work with Apache/mod_wsgi setups.** It has been deployed successfully with both Gunicorn/Nginx and even uWSGI/Nginx.

For configuration purposes, the following table maps environment variables to their Django setting:

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Django Setting</th>
<th>Development Default</th>
<th>Production Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DJANGO_ADMIN_URL</td>
<td>n/a</td>
<td>r'^admin/'</td>
<td>raises error</td>
</tr>
<tr>
<td>DJANGO_CACHES</td>
<td>CACHES (default)</td>
<td>locmem</td>
<td>redis</td>
</tr>
<tr>
<td>DJANGO_DATABASES</td>
<td>DATABASES (default)</td>
<td>See code</td>
<td>See code</td>
</tr>
<tr>
<td>DJANGO_DEBUG</td>
<td>DEBUG</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>DJANGO_SECRET_KEY</td>
<td>SECRET_KEY</td>
<td>CHANGEME!!</td>
<td>raises error</td>
</tr>
<tr>
<td>DJANGO_SECURE_BROWSER_XSS_FILTER</td>
<td>n/a</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>DJANGO_SECURE_SSL_REDIRECT</td>
<td>n/a</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>DJANGO_SECURE_CONTENT_TYPE_NOSNIFF</td>
<td>n/a</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>DJANGO_SECURE_FRAME_DENY</td>
<td>n/a</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>DJANGO_SECURE_HSTS_INCLUDE_SUBDOMAINS</td>
<td>n/a</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>DJANGO_SESSION_COOKIE_HTTPONLY</td>
<td>n/a</td>
<td></td>
<td>True</td>
</tr>
<tr>
<td>DJANGO_SESSION_COOKIE_SECURE</td>
<td>n/a</td>
<td></td>
<td>False</td>
</tr>
<tr>
<td>DJANGO_DEFAULT_FROM_EMAIL</td>
<td>n/a</td>
<td>&quot;your_project_name  &lt;noreply@your_domain_name&gt;&quot;</td>
<td></td>
</tr>
<tr>
<td>DJANGO_SERVER_EMAIL</td>
<td>n/a</td>
<td>&quot;your_project_name  &lt;noreply@your_domain_name&gt;&quot;</td>
<td></td>
</tr>
<tr>
<td>DJANGO_EMAIL_SUBJECT_PREFIX</td>
<td>n/a</td>
<td>&quot;[your_project_name]  &quot;</td>
<td></td>
</tr>
<tr>
<td>DJANGO_ALLOWED_HOSTS</td>
<td>ALLOWED_HOSTS</td>
<td>['*']</td>
<td>['your_domain_name']</td>
</tr>
</tbody>
</table>
The following gulp tasks are available from the default gulpfile:

**Styles autoprefixing and minification:**

```bash
$ gulp styles
```

**Javascript validation minification:**

```bash
$ gulp scripts
```

**Image compression:**

```bash
$ gulp imgCompression
```

**Run Django server:**

```bash
$ gulp runServer
```

**Browser sync server for live reload:**

```bash
$ gulp browserSync
```

**Default task:**

```bash
$ gulp
```

**Watch files for changes and reload the browser:**

```bash
$ gulp watch
```
5.1 Why is there a django.contrib.sites directory in Cookiecutter Django Gulp?

It is there to add a migration so you don’t have to manually change the sites.Site record from example.com to whatever your domain is. Instead, your {{cookiecutter.domain_name}} and {{cookiecutter.project_name}} value is placed by Cookiecutter in the domain and name fields respectively.

See 0002_set_site_domain_and_name.py.
This page contains some advice about errors and problems commonly encountered during the development of Cookiecutter Django Gulp applications.

1. If you get the error `jinja2.exceptions.TemplateSyntaxError: Encountered unknown tag 'now'.. please upgrade your cookiecutter version to >= 1.4 (see issue #528)

2. `project_slug` must be a valid Python module name or you will have issues on imports.
CHAPTER 7

Indices and tables

• genindex
• search
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>F</strong></td>
</tr>
<tr>
<td><strong>P</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>V</strong></td>
</tr>
</tbody>
</table>