在Flask中，为了避免硬编码并进行配置管理，可以使用以下方法：  
  
1. \*\*Environment Variables\*\*：使用环境变量来存储配置信息。 Flask 具有一个模块 `flask.config`，可以用来读取和写入环境变量。  
  
```python  
import os  
  
app = Flask(\_\_name\_\_)

# **读取 environment variable**

config = {  
 'DEBUG': bool(os.environ.get('FLASK\_DEBUG', False)),  
 'SQLALCHEMY\_DATABASE\_URI': os.environ.get('DATABASE\_URL')  
}  
  
app.config.update(config)  
```  
  
2. \*\*Config File\*\*：使用一个配置文件（`.ini` 或 `.env`) 来存储配置信息。 Flask 具有一个模块 `flask config`，可以用来读取和写入配置文件。  
  
```python  
from flask import Flask, Config  
  
app = Flask(\_\_name\_\_)

# **读取 configuration file**

app.config.from\_pyfile('config.ini')  
```  
  
3. \*\*Environment Variables in .env File\*\*：使用一个`.env`文件来存储配置信息，且不在 Git 中被托管。  
  
```python  
import os  
from flask import Flask  
  
app = Flask(\_\_name\_\_)

# **读取 environment variable from .env file**

app.config['DEBUG'] = bool(os.environ.get('FLASK\_DEBUG', False))  
app.config['SQLALCHEMY\_DATABASE\_URI'] = os.environ.get('DATABASE\_URL')  
```  
  
4. \*\*Database Configuration\*\*：使用 SQLAlchemy 来存储数据库配置信息。  
  
```python  
from flask import Flask, g  
from flask\_sqlalchemy import SQLAlchemy  
  
app = Flask(\_\_name\_\_)  
app.config.from\_object('config.Config')  
  
db = SQLAlchemy(app)

# **读取 database configuration from config.py**

class Config(object):  
 SQLALCHEMY\_DATABASE\_URI = os.environ.get('DATABASE\_URL')  
```  
  
5. \*\*动态配置\*\*：使用一个类来存储和读取配置信息，且可以根据不同的环境执行不同的配置。  
  
```python  
from flask import Flask  
  
app = Flask(\_\_name\_\_)

# **读取 configuration from config.py**

class Config(object):  
 DEBUG = os.environ.get('FLASK\_DEBUG', False)  
 SQLALCHEMY\_DATABASE\_URI = os.environ.get('DATABASE\_URL')  
  
config = Config()

# **根据环境执行不同的配置**

if config.DEBUG:  
 # DEBUG模式的配置  
 pass  
else:  
 # 非DEBUG模式的配置  
 pass  
```  
  
总之，避免硬编码并进行配置管理在Flask中，可以使用各种方法来存储和读取配置信息，并根据环境执行不同的配置。