

## Deploy the online forum application on a platform of your choice.

Deploying a full-stack web application with a Django backend and a React frontend can be done on various platforms. For this guide, we will use **Heroku** to deploy the Django backend and **Netlify** to deploy the React frontend.

### Part 1: Deploy the Django Backend on Heroku

#### Step 1: Prepare the Django App for Deployment

##### 1. Install Required Packages:

```
pip install gunicorn psycopg2-binary dj-database-url django-heroku
```

##### 2. Update `settings.py` for Deployment:

```
pytho
import django_heroku
import dj_database_url
from decouple import config

# Configure Django App for Heroku.
django_heroku.settings(locals())

# Database settings for PostgreSQL
DATABASES = {
    'default': dj_database_url.config(
        default=config('DATABASE_URL')
    )
}

# Security settings
DEBUG = False
ALLOWED_HOSTS = ['your-heroku-app-name.herokuapp.com']

# Static files settings
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')
STATIC_URL = '/static/'
```

##### 3. Create a `Procfile` in the root directory:

```
web: gunicorn myproject.wsgi --log-file -
```

##### 4. Add `requirements.txt` file:

```
pip freeze > requirements.txt
```

##### 5. Create a `.gitignore` file:

```
gitignore
*.pyc
__pycache__/*
db.sqlite3
env/
.env
```

### Step 2: Set Up Heroku

1. **Install the Heroku CLI:**
  - o Follow the instructions [here](#).
2. **Log In to Heroku:**

```
heroku login
```

3. **Create a New Heroku App:**

```
heroku create your-heroku-app-name
```

4. **Add Heroku Postgres Add-on:**

```
heroku addons:create heroku-postgresql:hobby-dev
```

5. **Set Environment Variables:**

```
heroku config:set DEBUG=False
```

6. **Push the Code to Heroku:**

```
bash
git init
git add .
git commit -m "Initial commit"
git push heroku master
```

### Step 3: Migrate the Database

1. **Run Migrations on Heroku:**

```
heroku run python manage.py migrate
```

2. **Create a Superuser on Heroku:**

```
heroku run python manage.py createsuperuser
```

## Part 2: Deploy the React Frontend on Netlify

### Step 1: Prepare the React App for Deployment

1. **Build the React App:**

```
npm run build
```

## Step 2: Set Up Netlify

1. **Create a Netlify Account:**
  - Sign up for a free account at [Netlify](#).
2. **Deploy the React App:**
  - Drag and drop the `build` folder into the Netlify dashboard.
  - Alternatively, you can use the Netlify CLI:

```
npm install -g netlify-cli  
netlify deploy
```

- Follow the prompts to link your site to your Netlify account and choose the `build` folder.
3. **Set Up Continuous Deployment:**
    - Link your GitHub repository to Netlify for continuous deployment.
    - Every time you push changes to your GitHub repository, Netlify will automatically build and deploy your React app.

## Summary

By following these steps, you will have deployed your Django backend on Heroku and your React frontend on Netlify. This setup ensures that your full-stack application is live and accessible on the internet.

- **Backend URL:** `https://your-heroku-app-name.herokuapp.com/`
- **Frontend URL:** `https://your-netlify-app-name.netlify.app/`

Make sure to update your React frontend to use the correct API endpoints provided by your Heroku backend. For example, if your API endpoint on Heroku is `https://your-heroku-app-name.herokuapp.com/api/posts/`, update your React app's Axios requests to point to this URL.

This deployment strategy leverages the strengths of Heroku for backend services and Netlify for static site hosting, providing a scalable and efficient solution for full-stack web applications.