

## Setting up a website to Access your Classifier

After confirming that your model works and correctly infers your species with reasonable accuracy, it is time to deploy it to a server or mobile app where you can be able to carry out inference from your model. Here i will illustrate how to make a web page where you can upload and classify images - more complicated user design and interface might need the services of a front end software engineer who are now abundant in the software engineering domain.

Depending on the complexity of the project there are many options to deploy including - Google App Engine, AWS Lambda, Amazon Sage Maker, AWS Elastic Beanstalk and Microsoft Azure Functions. There are also options to host it on your own servers for grand projects that might require dedicated devops engineers.

For our purpose we will deploy it to a pay as you go platform that can easily scale and is easy for anyone to setup. The link here leads to a page of how to do this by deploying your trained models to such a platform by the name - Render [https://course.fast.ai/deployment\\_render.html](https://course.fast.ai/deployment_render.html). Below i explore the process step by step with an assumption of basic github skills.

### 1. Github configurations

The process involves forking a starter app from <https://github.com/render-examples/fastai-v3>. Signing up for a render account here <https://dashboard.render.com/>. Then uploading the trained model file created when you run `#CODE BLOCKS 18#` (it is located on your path and might have a name like `export.pkl`) to a cloud service like Google Drive or Dropbox. Then you can copy the download link for the file from these cloud services.

You can then go to the forked github folder above and edit the file `server.py` inside the `app` directory and update the `export_file_url` variable with the URL copied above from the cloud services. In the same file, you should update the line:

```
classes = ['Pisaster ochraceus', 'Pycnopodia helianthiodes', 'Solaster dawsonii']
```

with the classes, you expect from your model - in our case the three sea star species. You might be going back and forth by committing to your GitHub as you improve your model - Render integrates with your GitHub repo and automatically builds and deploys changes every time you push a change making the process seamless.

### 2. Configurations on Render

You can now go to Render and create a new **Web Service** and use the repo you created above. You will need to grant Render permission to access your Github repo in this step. On the deployment screen, pick a name for your service and use `Docker` for the Environment. The URL will be created using this service name. The service name can be changed if necessary, but the URL initially created can't be edited. Then click **Save Web Service**.

That's it! Your service will begin building and should be live in a few minutes at the URL displayed in your Render dashboard. You can follow its progress in the deploy logs.

## **TESTING**

Your app's URL will look like <https://seastar-classifier.onrender.com> depending on the name of your service. You can also monitor the service logs as you test your app. You are now ready to go and start classifying species from real life photos that you might have.