**Fortune Teller**

**Fortune Teller** is a very basic application composed of two services:

1. [Fortune Service](https://github.com/spring-cloud-services-samples/fortune-teller/blob/master/fortune-teller-fortune-service) - serves up random Chinese fortune cookie fortunes
2. [Fortune UI](https://github.com/spring-cloud-services-samples/fortune-teller/blob/master/fortune-teller-ui) - presents a UI that consumes the fortune service

It leverages libraries and services from Spring Cloud and Netflix OSS to compose the system.

Fortune Teller is deployable to any Cloud Foundry environment utilizing the service components that have been packaged with the project. However, it is most easily deployed to Pivotal Cloud Foundry environments that have installed the [Spring Cloud Services](https://network.pivotal.io/products/p-spring-cloud-services) package.

**Build**

1. Using Maven, build and package the application:

$ mvn clean package

Maven will automatically download all of *Fortune Teller*'s dependencies. This may take a few moments.

**Deploy to Pivotal Cloud Foundry with Spring Cloud Services**

1. Run scripts/pcf-create-services.sh to create the services that you need:
2. $ ./scripts/pcf-create-services.sh
3. Creating service fortunes-db in org microservices / space fortune-teller as admin...
4. OK
5. Creating service config-server in org microservices / space fortune-teller as admin...
6. OK
7. Creating service service-registry in org microservices / space fortune-teller as admin...
8. OK
9. Creating service circuit-breaker-dashboard in org microservices / space fortune-teller as admin...

OK

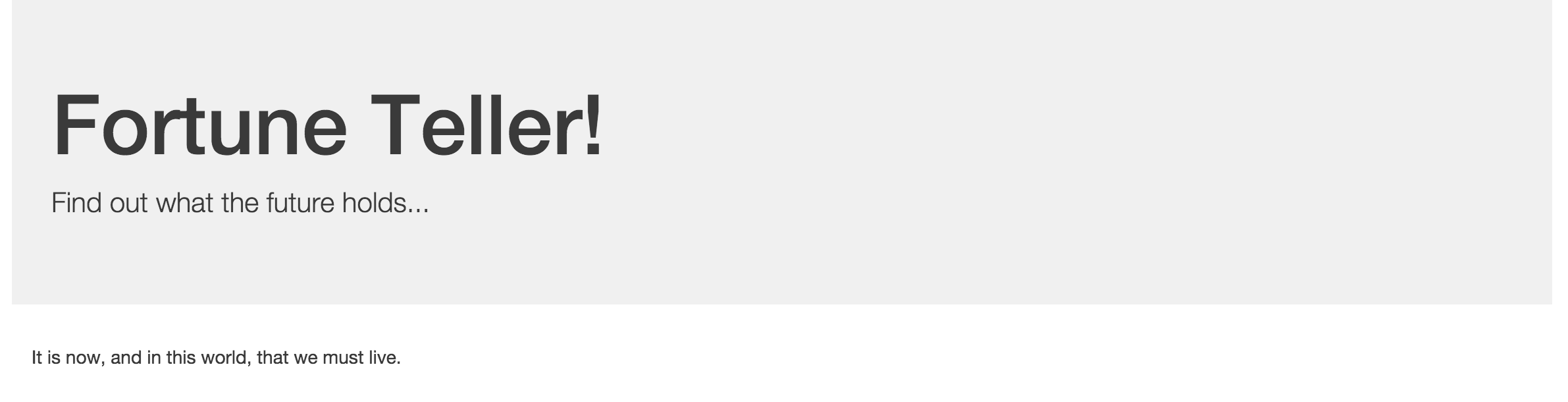
1. Click on the **Manage** links for the **Config Service**, **Service Registry**, and **Circuit Breaker Dashboard**. Make sure the services are finished initializing before you proceed.
2. Edit the manifest-pws.yml file to specify the Cloud Foundry target the apps are being pushed to, replacing the URL in CF\_TARGET: <https://api.yourpcfenvironment.local> with the API endpoint for your Cloud Foundry deployment.
3. Push the microservices:

$ cf push -f manifest-pcf.yml

This will push the fortunes service and the ui application and bind all of the services.

**Test the Application**

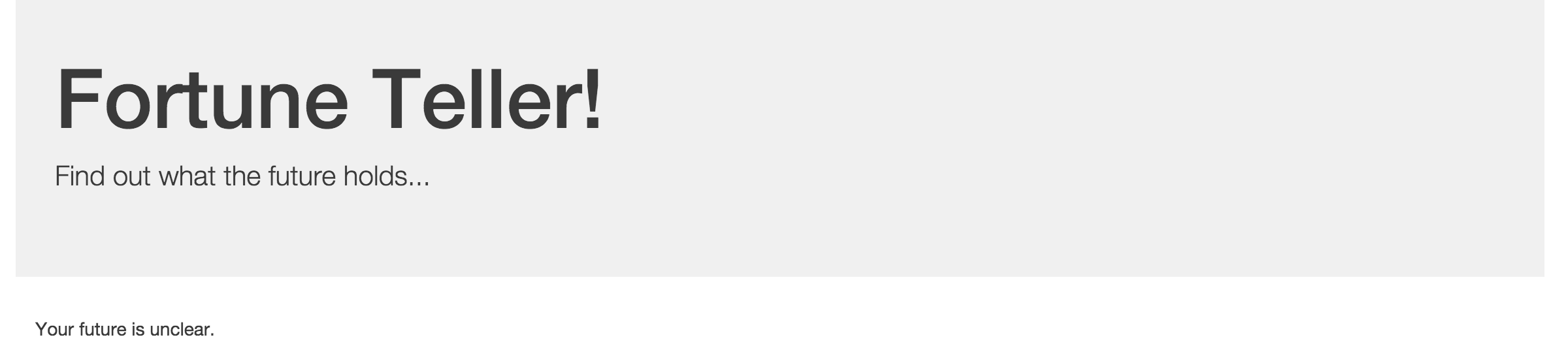
1. In a browser, access the fortunes-ui application at the route that was created for you:

[](https://github.com/spring-cloud-services-samples/fortune-teller/blob/master/docs/images/fortunes_1.png)

1. Stop the fortunes application:

$ cf stop fortune-service

1. Access the fortunes-ui and see that the ``fallback fortune'' is being returned.

[](https://github.com/spring-cloud-services-samples/fortune-teller/blob/master/docs/images/fortunes_4.png)

1. Start the fortunes application:

$ cf start fortune-service

1. Continue to access the fortunes-ui. After the fortunes service has re-registered with Eureka and the fortunes-ui load balancer caches are refreshed, you should then start getting random fortunes again!