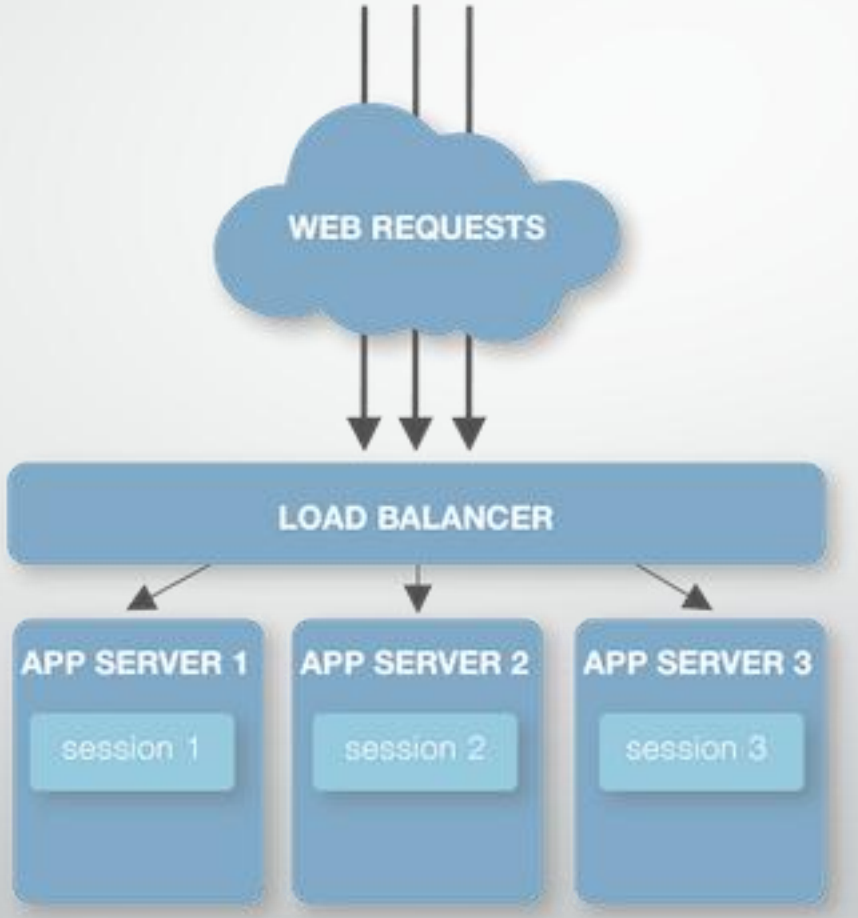
Programowanie w chmurze na platformie Java EE

Wykład 5 - dr inż. Piotr Zając

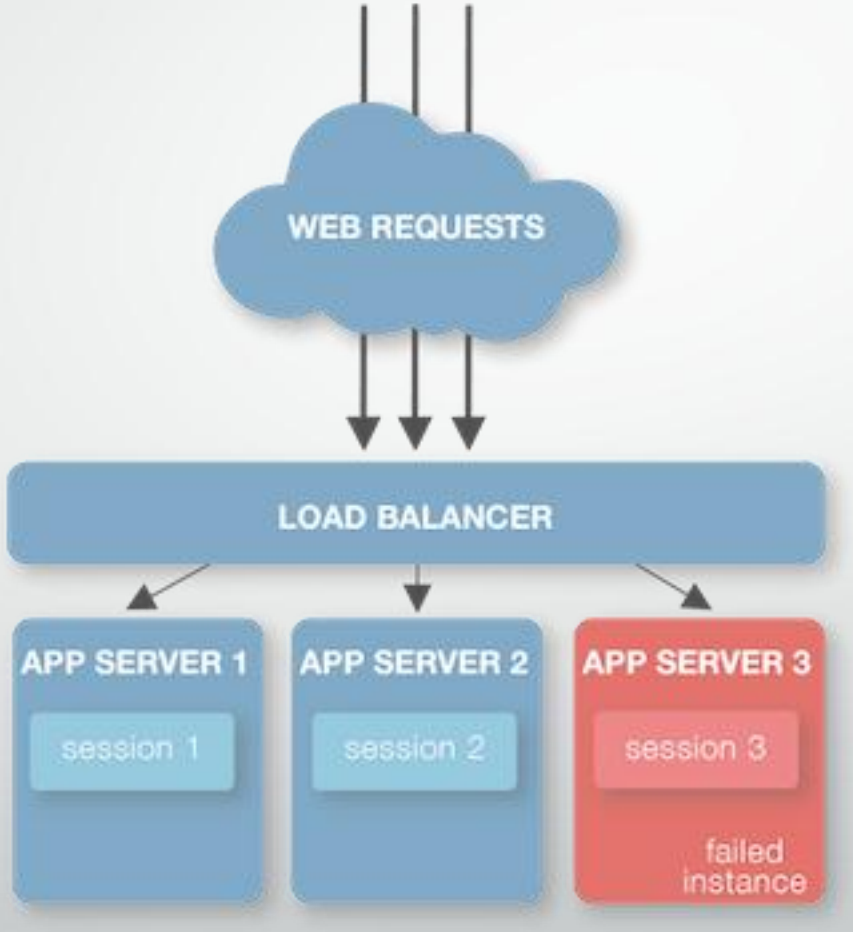
Web session clustering

- Ignite In-Memory Data Fabric is capable of caching web sessions of all Java Servlet containers that follow Java Servlet 3.0 Specification, including Apache Tomcat, Eclipse Jetty, Oracle WebLogic, and others.
- Web sessions caching becomes useful when running a cluster of app servers. When running a web application in a servlet container, you may face performance and scalability problems. A single app server is usually not able to handle large volumes of traffic by itself. A common solution is to scale your web application across multiple clustered instances

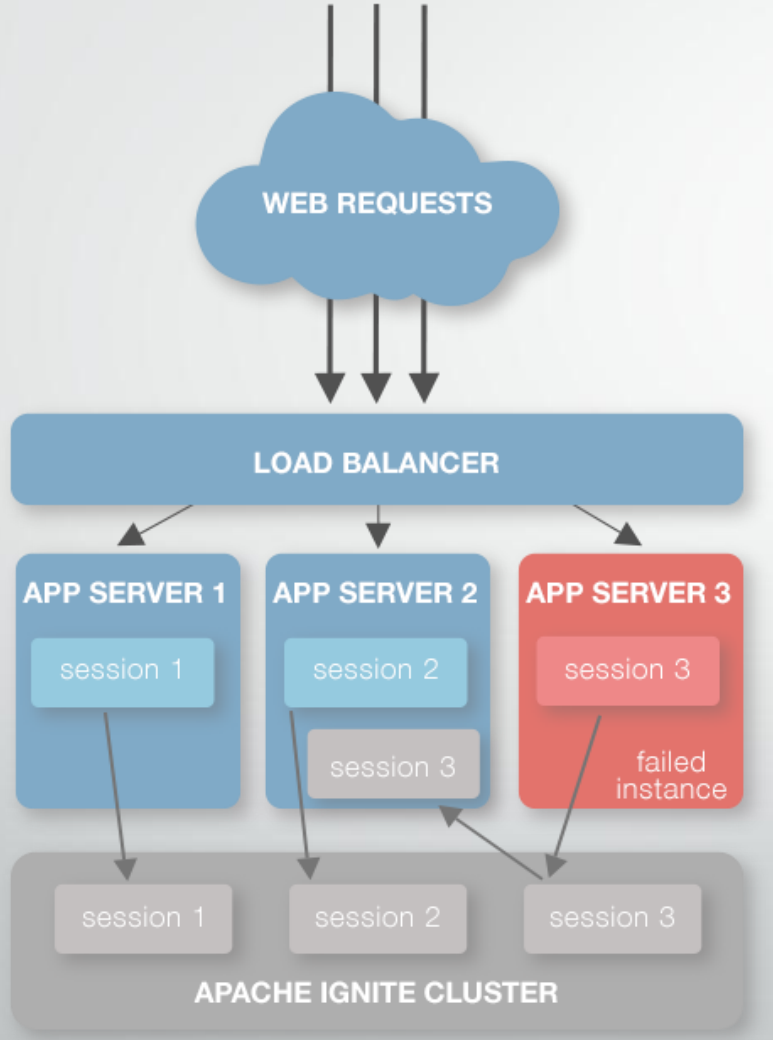
Web session clustering



Web session clustering



Web session clustering



- A solution here is to use Ignite In-Memory Data Fabric web sessions cache - a distributed cache that maintains a copy of each created session, sharing them between all instances. If any of your application instances fails, Ignite will automatically restore the sessions

Web session clustering

- Add Apache Ignite to your project
- Configure cache in XML

```
<dependency>
      <groupId>org.apache.ignite</groupId>
      <artifactId>ignite-core</artifactId>
      <version> ${ignite.version}</version>
</dependency>

<dependency>
    <groupId>org.apache.ignite</groupId>
    <artifactId>ignite-web</artifactId>
    <version> ${ignite.version}</version>
</dependency>

<dependency>
    <groupId>org.apache.ignite</groupId>
    <artifactId>ignite-log4j</artifactId>
    <version>${ignite.version}</version>
</dependency>

<dependency>
    <groupId>org.apache.ignite</groupId>
    <artifactId>ignite-spring</artifactId>
    <version>${ignite.version}</version>
</dependency>
```

```
<bean class="org.apache.ignite.configuration.CacheConfiguration">
    <!-- Cache mode. -->
    <property name="cacheMode" value="REPLICATED"/>
    ...
</bean>
```

```
<bean class="org.apache.ignite.configuration.CacheConfiguration">
    <!-- Cache mode. -->
    <property name="cacheMode" value="PARTITIONED"/>
    <property name="backups" value="1"/>
</bean>
```

Web session clustering

- Declare context listener
- Declare web session filter

```
<listener>
    <listener-class>org.apache.ignite.startup.servlet.ServletContextListenerStartup</listener-class>
</listener>

<filter>
    <filter-name>IgniteWebSessionsFilter</filter-name>
    <filter-class>org.apache.ignite.cache.websession.WebSessionFilter</filter-class>
</filter>

<!-- You can also specify a custom URL pattern. -->
<filter-mapping>
    <filter-name>IgniteWebSessionsFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>

<!-- Specify Ignite configuration (relative to META-INF folder or Ignite_HOME). -->
<context-param>
    <param-name>IgniteConfigurationFilePath</param-name>
    <param-value>config/default-config.xml </param-value>
</context-param>

<!-- Specify the name of Ignite cache for web sessions. -->
<context-param>
    <param-name>IgniteWebSessionsCacheName</param-name>
    <param-value>partitioned</param-value>
</context-param>
```

Web session clustering

- https://apacheignite-mix.readme.io/docs/web-session-clustering

Exercise 2: Cache

- Web Session Clustering
- Start at least 2 Ignite nodes (in a VM or on the host)
- Implement a java app using session data according to recommendation in the documentation
- Implement following requests:
  - /login - two states
    - logged in (display a webpage to show that the user is logged in)
    - logged out
  - /logout
- Create a simple web page with a form for the login app

Exercise 2: Cache

- Use the same repo as in Exercise 1
- Setup Apache Ignite Web Session Clustering for this service
- Setup cached data to be spread on whole available cluster memory in copies connected to started cluster, use the cacheMode that is suitable for performance tasks

Exercise 2: Cache

Example:

- start two console apps from ignite download directory
  - bin/ignite.sh <your-configuration.xml>
  - bin/ignite.sh <your-configuration.xml>
- run the login project on app server
- login in the login app
- display webpage (it should show that user is logged in)
- kill login app, do not kill console apps
- start login app again
- display the webpage: your user session should be kept in cache on Ignite nodes run from console, so the user should be still logged in