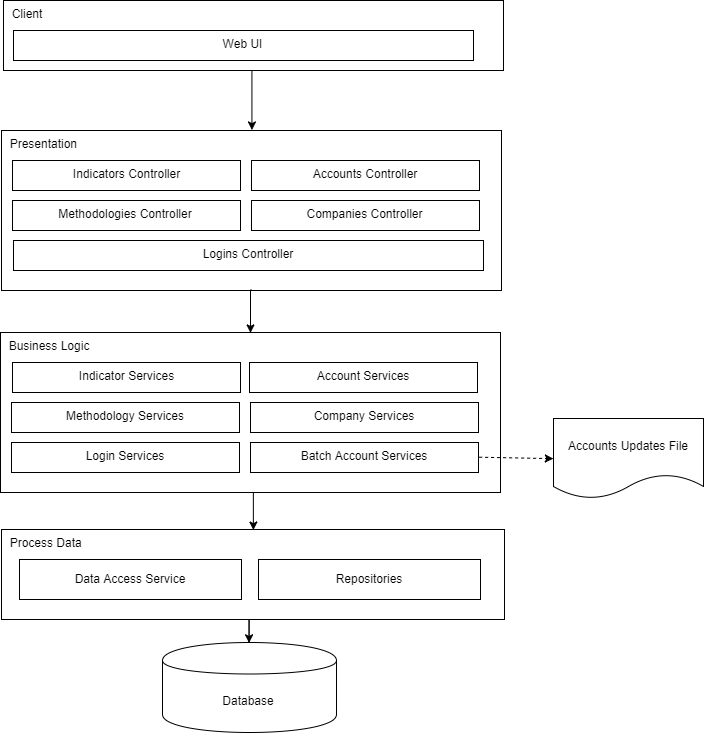
**Investment-Management: Arquitectura final del sistema**

L**a aplicación puede ser accedida desde cualquier dispositivo mediante el uso de un** navegador web. La misma se encuentra alojada en un servidor, provisto por *DigitalOcean*, el cual cuenta con las siguientes características:

* **SO:** Ubuntu 16.04.3 x64
* **RAM:** 1 GB
* **HARD DISK:** 20 GB Disk
* **Web Server:** Apache Tomcat 7.0.68
* **MySQL:** 5.7.20
* **JVM 1.8.0\_151-8u151-b12-0ubuntu0.16.04.2-b12**

Respecto al aspecto lógico, se encuentra estructurada de manera tal que se obtenga una división de responsabilidades y se puedan aislar correctamente lo componentes.Por esta razón implementa una arquitectura en capas (incorporando el patrón arquitectónico MVC) que se divide en:

* La obtención y persistencia de información desde y hacia la base de datos.
* El manejo de la lógica estrictamente relacionada con el negocio.
* La presentación de los datos al usuario.



Como se puede observar, contamos con los siguientes componentes lógicos:

Controladores:

En esta capa se definen los componentes cuyo propósito es brindar respuesta antes las distintas solicitudes realizadas por los clientes. Los controladores se encargan de obtener la información necesaria para responder a una solicitud y de definir cuál será la vista encargada de mostrar la misma. A su vez funcionan como un punto de entrada para las solicitudes que persisten información en la aplicación.

Servicios:

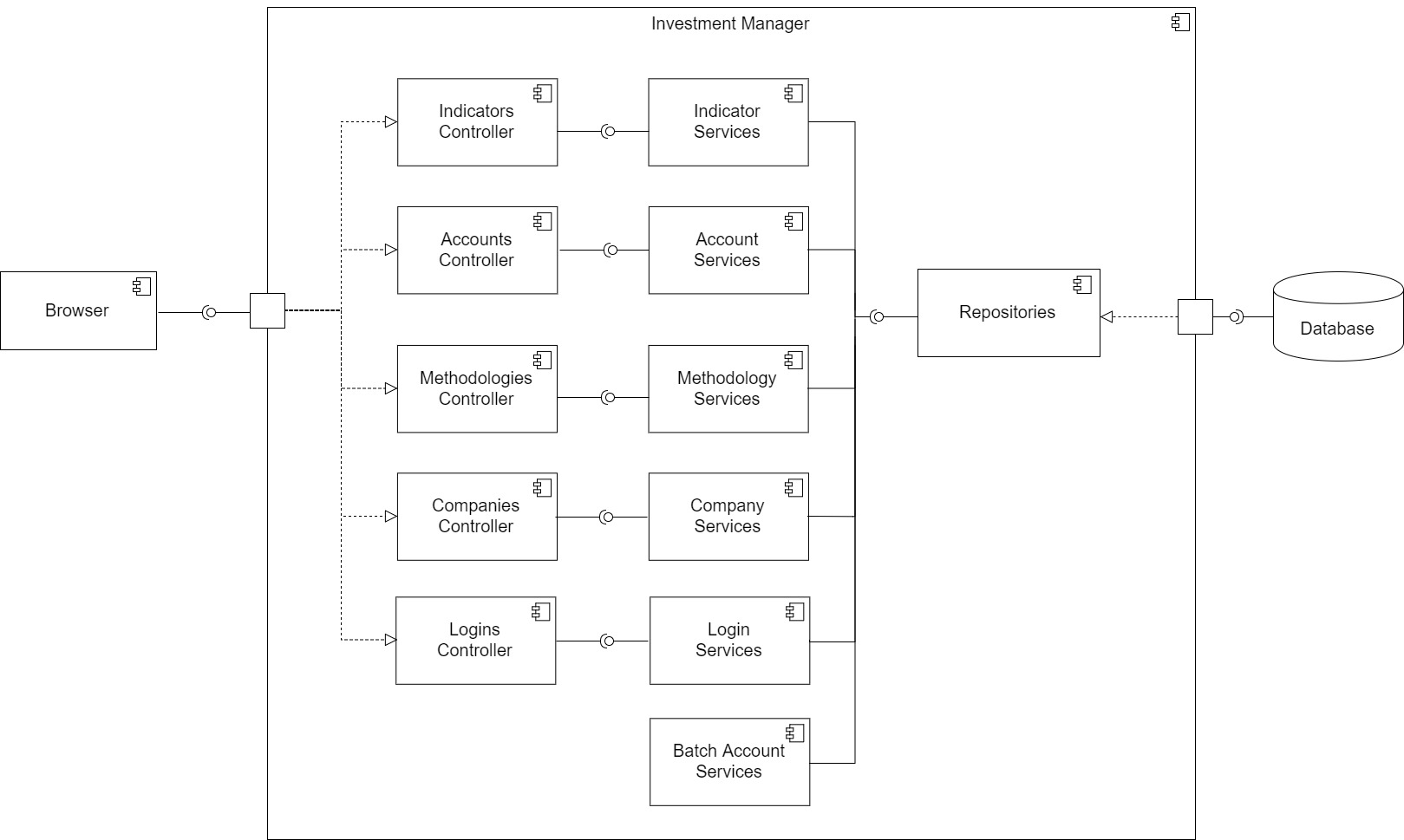
Los servicios abstraen cualquier lógica de negocio que pudiera llegar a existir en la aplicación.

Actúan como un intermediario entre los controladores y los repositorios

Datos:

Esta capa es la responsable de gestionar la persistencia y consulta de datos en la base.

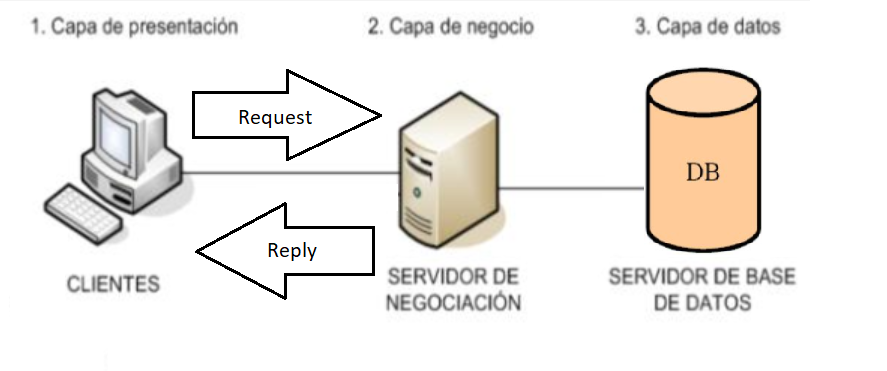
Además de una base de datos relacional en MySQL. El mismo provee funciones para el almacenamiento y protección de los datos de la aplicación. A su vez permite que varios usuarios puedan acceder a la base de datos de forma concurrente.



Por último, podemos observar una comunicación sincrónica del tipo request-reply, conformada por 2 partes:

* Un cliente que consume servicios como parte de sus operaciones (request)
* Un servidor que provee uno o más servicios (reply)

Contamos con componentes que solicitan servicios y componentes que los prestan. Sumado a lo explicado con anterioridad esto resulta en lo que se denomina arquitectura Cliente-Servidor de 3 capas:



Tecnologías utilizadas:

Spark, Java 8, Quartz Job Scheduler, ANTLR, MySQL, Hibernate, Maven, HTML, CSS, Bootstrap