# Documentación Arquitectura de Software

A continuación se analizan distintos aspectos de dos stacks elegidos para el desarrollo de una aplicación web.

1- Descripción
2- URLS Repos (Back , Front)
3- Puntos a Favor / Justificación de elección
4- Puntos en contra.
[- Conclusión / elección
 - Contraposición de 3 y 4 ]

## Primer Stack

1 - Como primer opción se eligió Angular 4, Java, Hibernate y Jetty
2 - Front : https://github.com/arq-unq-Ladavaz-Toledo-Zagarella/arq-inscription-frontend
Back : https://github.com/arq-unq-Ladavaz-Toledo-Zagarella/arq-inscription-backend
3 -

- La gran cantidad de documentación disponible.
- La experiencia de los integrantes del equipo desarrollando en estas tecnologías.
- Java corre en múltiples núcleos de modo transparente.
- Hibernate permite gran escabilidad.
- En Jetty los tiempos de carga suelen ser reducidos, además consumen poca memoria y puede atender gran cantidad de solicitudes.
- Por último Jetty aumenta la cantidad de conexiones de manera dinámica.

4 -

- Lentitud a la hora de ejecutar aplicaciones.

## Segundo Stack

1 - Como primer opción se eligió también Angular 4 pero con NodeJs, Mongodb y Express

2 - Front : https://github.com/arq-unq-Ladavaz-Toledo-Zagarella/arq-inscription-frontend
Back:
https://github.com/arq-unq-Ladavaz-Toledo-Zagarella/arq-inscription-backend-nodejs

3 -

- Con NodeJs se desarrollan aplicaciones web de manera rápida.
- Es altamente escalable debido a que es capaz de manejar un gran número de conexiones simultáneas con alto rendimiento.
- Express fue elegido por su sencillez y robustez.

4 -

- Cuando se trata de cómputo pesado, Node.js no es la mejor opción.
- Cualquier operación de uso intensivo de CPU anula todas las ventajas de rendimiento.

# Conclusiones - Análisis

Teniendo pros y contras de cada stack por separado. vamos a proceder a compararlos por igual:
La cantidad de documentación disponible sobre java,hibernate,etc consideramos que es mayor a la del segundo stack ya que se trata de tecnologías que no tienen la misma antigüedad.
La experiencia del equipo trabajando con el primer stack es en parte también debido al primer punto, la antigüedad, aunque los últimos proyectos realizados se hicieron con las tecnologías del segundo stack.
Una de las cosas que se ve en ambos stack es la escalabilidad, esto es importante ya que nuestro sistema será utilizado por los alumnos de TPI, donde la cantidad crece año a año, por lo tanto nuestro sistema debe estar preparado para soportar este crecimiento continuo.
También queremos que la aplicación sea rápida, donde se puedan inscribir fácilmente, en este sentido vemos una pequeña ventaja en el segundo stack, donde las aplicaciones desarrolladas con node suelen ser más veloces.
En cuanto a la seguridad de la aplicación no vemos que tenga un rol importante dentro de esta, solo la mínima, se puede realizar con ambos stacks.
Otro de los puntos que se menciona es la capacidad de Java para realizar cálculos pesados mientras que Node falla en este punto, sin embargo consideramos que los requerimientos pedidos no van a ser un problema para Node.

Analizando los pros y contras de ambas opciones por separado y luego uniendolas y analizando también requerimientos no funcionales, se decidió por desarrollar la aplicación con el stack de la segunda opción (NodeJs, Mongodb, ..) basándonos en gran medida por la escalabilidad de la tecnología.

Para la integración continua se utilizará Travis CI.
Como parte de la primer entrega se realizó la siguiente funcionalidad:

> *Permitir que un estudiante pueda completar su encuesta de pre-inscripción, indicando a qué materias desea inscribirse y registrando esa información en el sistema.*

Modo de ejecución:

```
$ npm install
$ node node_modules/webpack/bin/webpack.js --progress
$ npm start
```