El siguiente experimento planea poner a prueba las tecnologías escogidas para verificar si se cumplen los requerimientos no funcionales de latencia y rendimiento. Se espera crear un prototipo básico en el cual se puedan probar tales requerimientos con la tecnología escogida.

Los atributos de calidad que serán evaluados en este experimento son:

- Desempeño: Que el sistema sea ágil y eficiente para cumplir sus respectivas funcionalidades (Entre menos latencia haya, mejor el desempeño).
- Escalabilidad: Que el sistema pueda crecer sin sacrificar considerablemente el desempeño (Entre mejor desempeño con concurrencia, mejor escalabilidad).

Pre-Experimentación:

1. Problemática: El problema principal expuesto en el enunciado es la dificultad del control oportuno de los vehículos en su flota. Para esto se propone que utilizando el dispositivo GPS integrado en cada uno de los vehículos (excepto los vcubs), se reporte la localización actual de los vehículos al sistema central, quien se encargará de tomar decisiones en base a las necesidades actuales. Además de esto se plantea que las estaciones de vcub den su porcentaje de ocupación. Todo esto se plantea hacer en real time.
2. Objetivo: Se plantea utilizar PhP lumen para implementar el sistema de recepción de información en real time tanto de las estaciones como de los vehículos. Y en el aviso de emergencia de los tranvías. Una vez recibida la información se utiliza la arquitectura dicha para tomar decisiones de las rutas de los mobibuses, llenado nuevo de las estaciones de vcubs y supervisión del sistema de tranvías.
3. Descripción del experimento: Las actividades a realizar son:

➔ Desarrollar la capa de lógica de negocio que debe ofrecer el servidor del centro de control.

➔ Desarrollar las pruebas de carga con JMeter.

➔ Realizar una primera versión del documento de experimentación.

➔ Desarrollar la capa de persistencia.

➔ Volver a ejecutar las pruebas de carga.

➔ Actualizar el documento de experimentación.

➔ Analizar y comparar los resultados obtenidos en la entrega parcial con los de la entrega final del experimento.

➔ Desarrollar la capa de presentación de la aplicación Web requerida por el operador de Tbc.

➔ Desarrollar la capa de presentación de las aplicaciones Standalone que simulan Mobibus, VCub, Tranvía.

Y los datos a reconectar son:

➔ Datos de latencia: tiempo de respuesta de la aplicación.

➔ Datos de escalabilidad: tiempo de soporte de x cantidad de vehículos.

4. Los artefactos a construir inicialmente son: los controladores de las entidades básicas (Mobibus, tranvía, usuario y estación de vcub), el ruteo de dichos elementos y lo mínimo para que los requerimientos CRUD se cumplan adecuadamente.

5. Los recursos que se van a utilizar para el experimento son:

- El IDE PhpStorm: para el desarrollo del código.
- Php lumen: para la escritura del código.
- Github: para un repositorio común de todos los miembros.
- JMeter y loader.io para la realización de pruebas del codigo y ruteo.

6. Los resultados esperados son:
   - Que el tiempo de respuesta de la aplicación con 4500 elementos sea de un segundo o menos.
   - Que el tiempo de soporte sea de 5 segundos máximo con 4500 vehículos en el sistema.

7. El tiempo esperado para la implementación de la capa lógica es de aproximadamente 4 horas de trabajo en lenguaje PhP por integrante, encima de esto 2 horas de todo el grupo para identificación de los casos de usos y requerimientos del experimento y 1 hora para la realización de este documento.

Post-Experimentación

1. Resultados Obtenidos

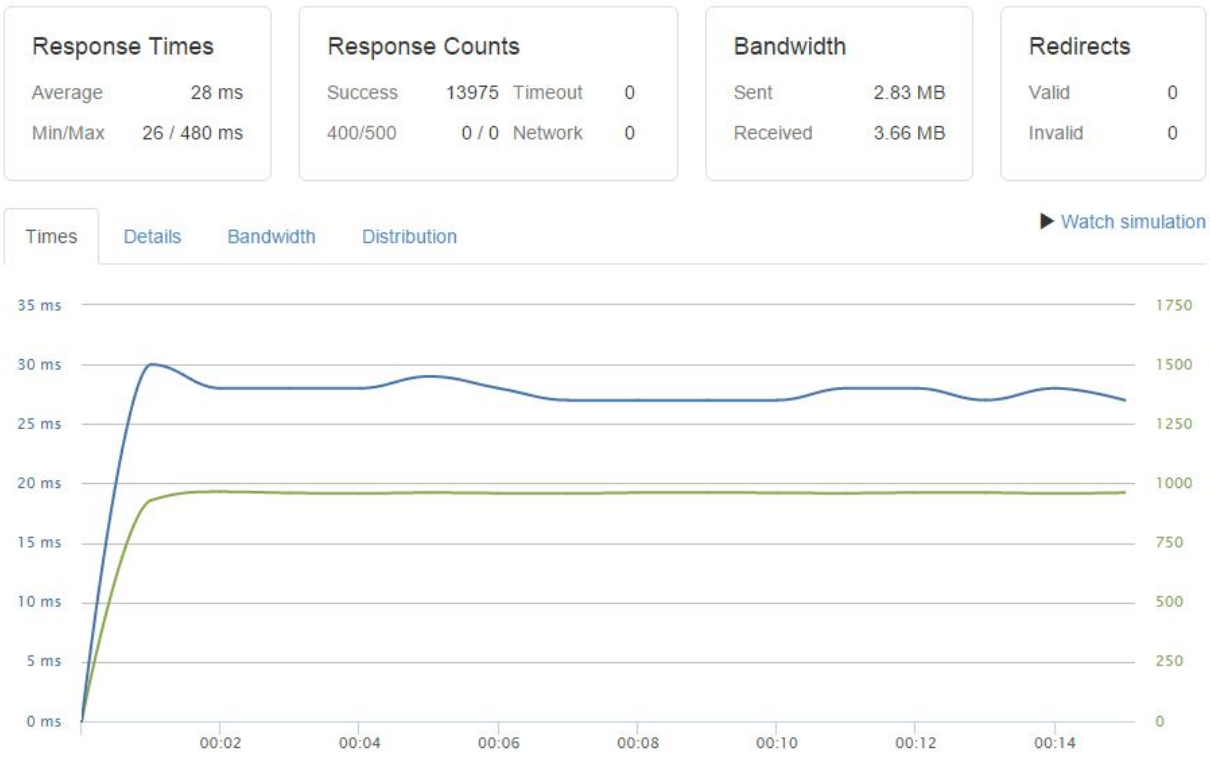
Los resultados obtenidos fue la construcción de la lógica de la aplicacion en lenguaje php en un tiempo estimado de 5 horas por integrante. Identificando 12 casos de uso.

Pruebas de latencia - Casos de uso:

CASOS PARA EL MOBIBUS:

1. Ver un mobibus (GET):

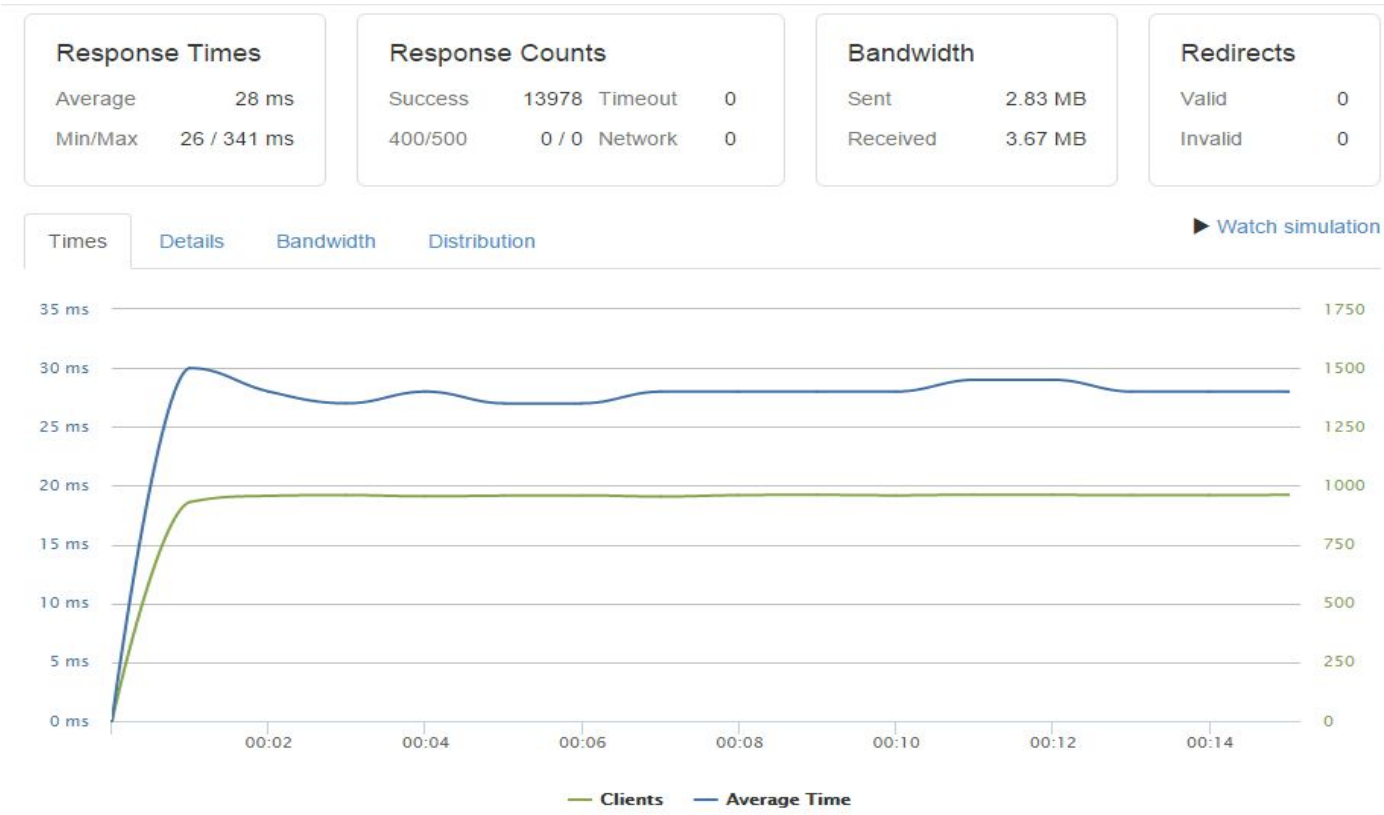
2. Reportar la posición de un mobibus (POST):



3. Ocupar un Mobibus (POST)

CASOS PARA TRANVÍA:

1. Obtener un tranvía (GET):



2. Reportar La posición (POST)

3. Reportar Emergencia

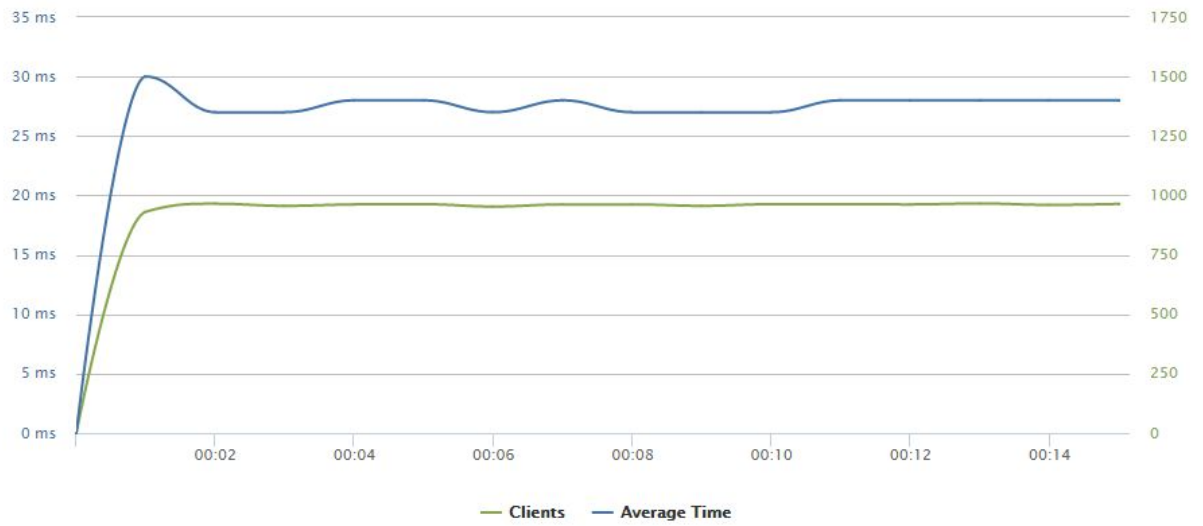
CASOS PARA VCUBS:

1. Registrar un Vcub en una terminal (POST):

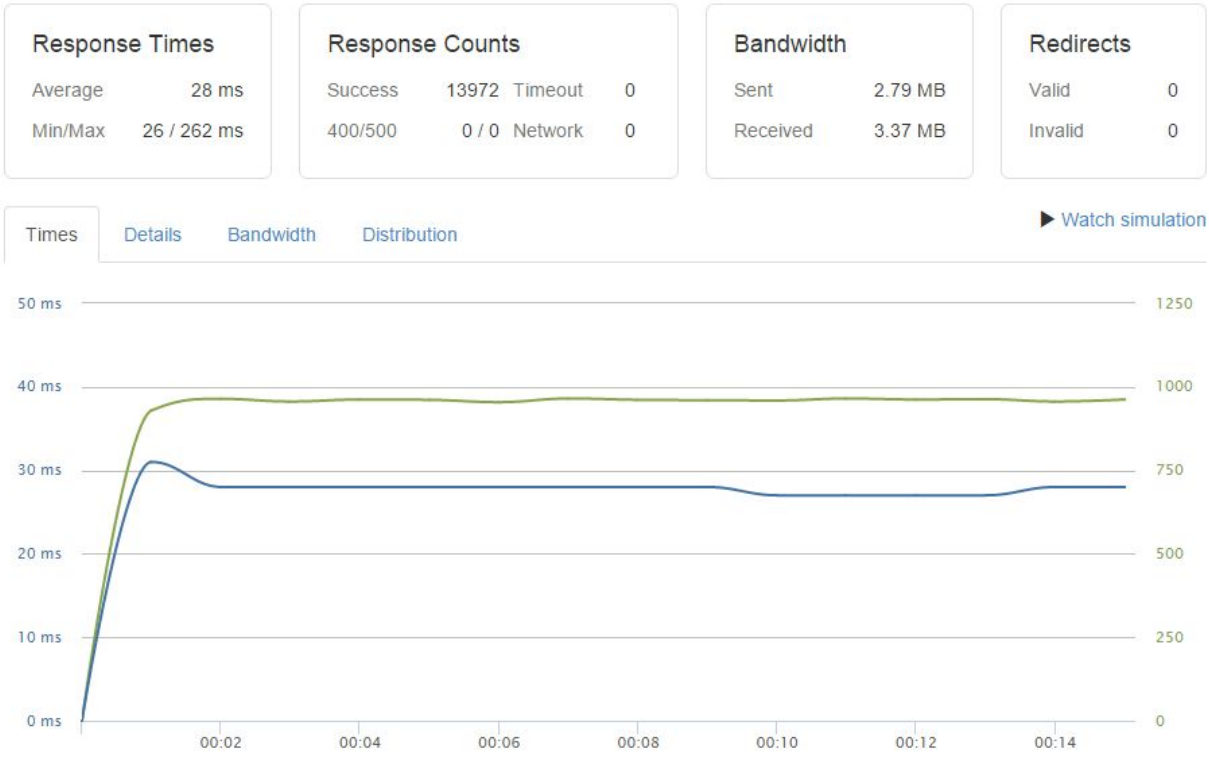
2. Prestar un Vcub (POST):

Response Times		Response Counts			Bandwidth		Redirects	
Average	28 ms	Success	13973	Timeout 0	Sent	2.80 MB	Valid	0
Min/Max	26 / 260 ms	400/500	0 / 0	Network 0	Received	3.34 MB	Invalid	0

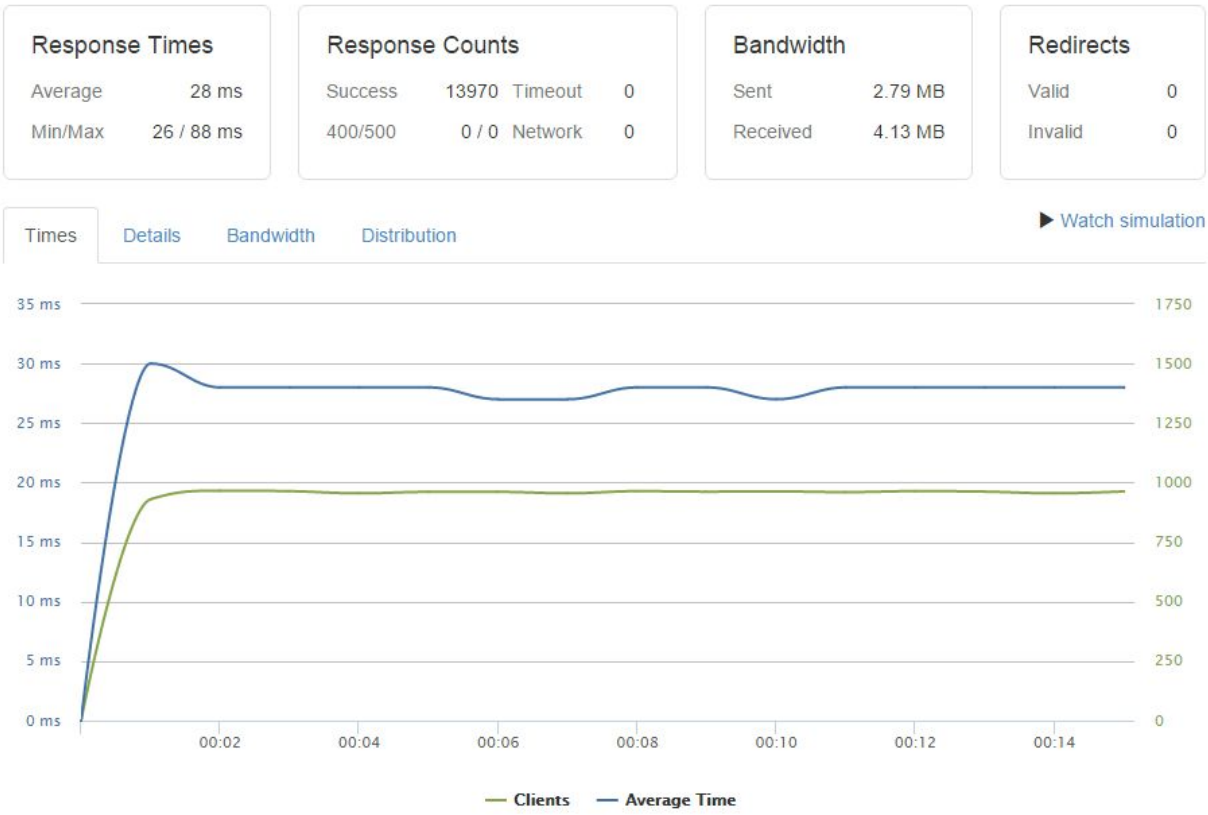
Times Details Bandwidth Distribution ▶ Watch simulation



3. Recibir un Vcub (PUT):



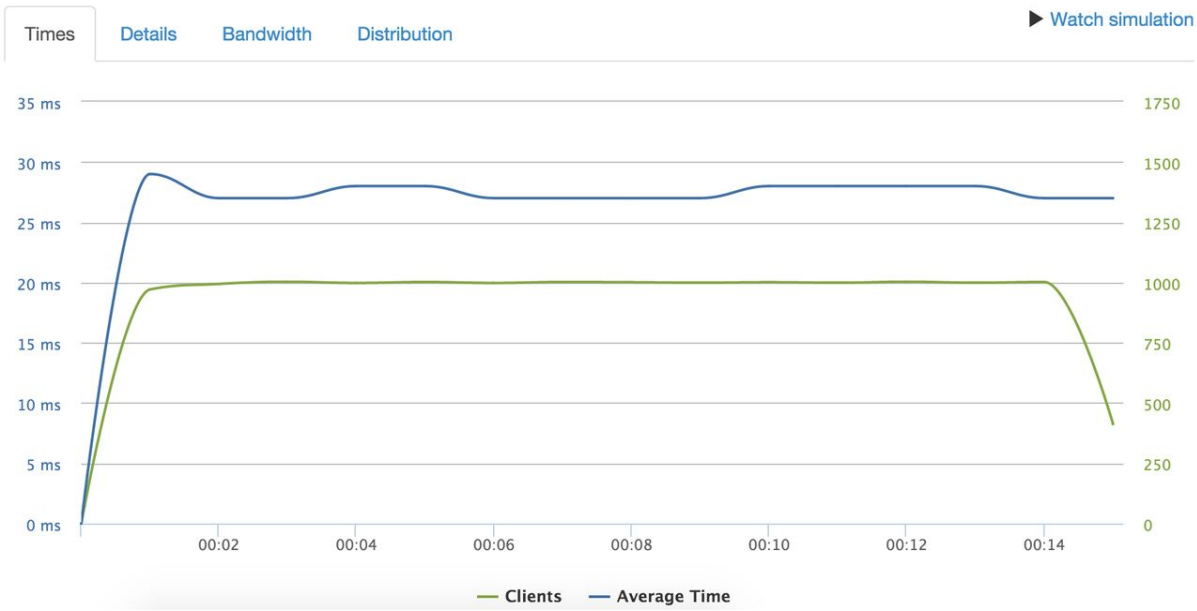
4. Pedir un llenado de Vcubs:



CASOS PARA Usuario:

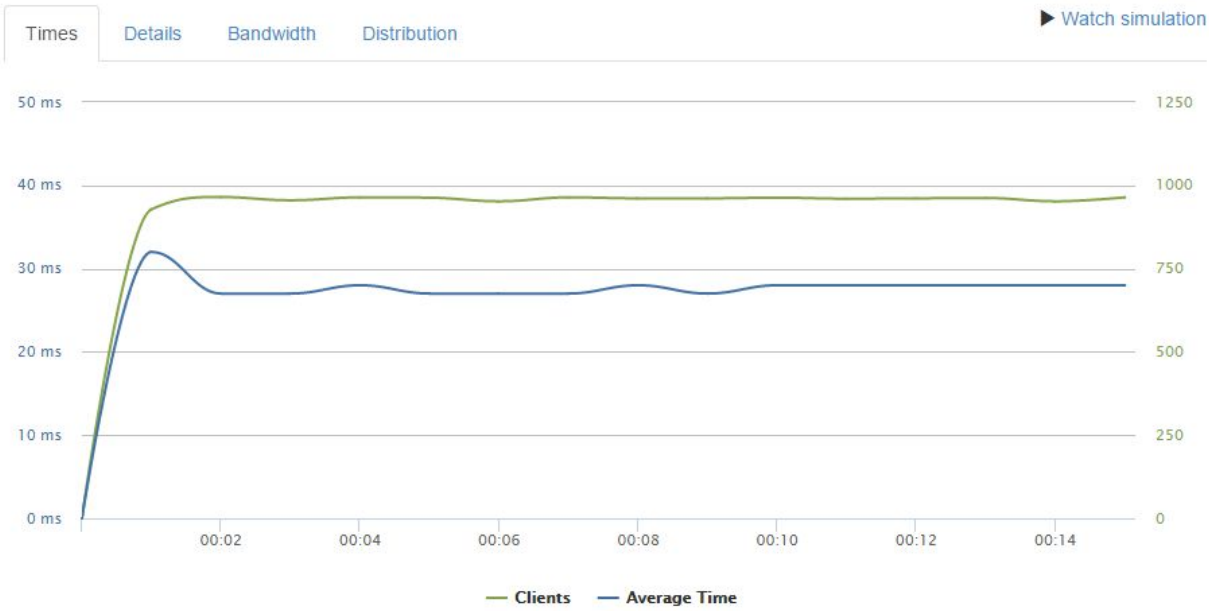
1. **Solicitar un mobibus (GET):**

Response Times		Response Counts			Bandwidth		Redirects	
Average	28 ms	Success	13994	Timeout 0	Sent	2.88 MB	Valid	0
Min/Max	26 / 255 ms	400/500	0 / 0	Network 0	Received	4.52 MB	Invalid	0



2. Ver estado de la solicitud del mobibus (GET):

Response Times		Response Counts			Bandwidth		Redirects	
Average	28 ms	Success	13975	Timeout 0	Sent	1.84 MB	Valid	0
Min/Max	26 / 100 ms	400/500	0 / 0	Network 0	Received	4.40 MB	Invalid	0



2. Duración real: Realizar una comparación con los tiempos teóricos y los reales del desarrollo para cada etapa del proceso de experimentación

   Se puede ver la comparación implícita en las gráficas arriba.

3. Artefactos construidos: Cuáles artefactos fueron construidos y cuáles no, explicando las causas por las que no fueron desarrollados.

   Principalmente se quería desarrollar la capa lógica de negocio con todas las entidades pero sin mucha profundidad. Por lo cual solo se desarrollaron los controladores, el ruteo y lo mínimo posible requerido para que todos los requerimientos CRUD se pudieran comprobar.

   La causa principal de solo desarrollar las entidades básicas es que para este experimento solo se quería comprobar que los requerimientos no funcionales se cumplieran a cabalidad. Por esta razón no se desarrolló un componente de persistencia, ni una capa de presentación.

4. Análisis: Interpreta, explica y justifica los resultados obtenidos basado en patrones y tácticas arquitecturales implementadas en el experimento.

   - Análisis de resultados caso -Mobibus:

     Se cumple con el criterio de calidad ya que las solicitudes son atendidas a una gran velocidad, con un máximo de 480 milisegundos por solicitud. eso es gracias a que el proyecto está en la nube y la cantidad de threads para este tipo de elemento no eran tantas.

   - Análisis de resultados caso Tranvía:

     Se cumple con el criterio de calidad ya que las solicitudes son atendidas a una gran velocidad, con un máximo de 341 milisegundos por solicitud. eso es gracias a que el proyecto está en la nube y la cantidad de threads para este tipo de elemento no eran tantas.

   - Análisis de resultados caso Vcubs:

     Se cumple con el criterio de calidad ya que las solicitudes son atendidas a una gran velocidad, con un máximo de 262 milisegundos por solicitud. eso es gracias a que el proyecto está en la nube y a pesar de la gran cantidad de elementos que poseía esta clase, las tareas a realizar eran muy sencillas; lo cual hizo que el procesamiento de información no tome mucho tiempo

   - Análisis de resultados caso Usuario:

No cumple con el criterio de calidad para todos los casos ya que a pesar de que las solicitudes son atendidas a gran velocidad, el caso en el que se solicita un mobibus tiene un máximo de 1057 milisegundos, 57 milisegundos más lento de lo que se esperaba, esto se debe a que hay un análisis de todos los mobibuses cerca del área, esta información se compara con la latitud y la longitud en la que se encuentra el usuario y por lo tanto hay que revisar la posición de todos los mobibuses en el sistema.

5. Conclusiones: Teniendo en cuenta la hipótesis, plantea recomendaciones basadas en los resultados obtenidos en la etapa de experimentación.

   Para todos los casos de uso en el sistema planteado, se cumple de manera óptima los criterios de calidad estipulados por nuestro cliente, por lo tanto podemos concluir que gracias a nuestra arquitectura, lógica y recursos el producto final será eficiente cumplira con satisfacer al usuario.