# Testing

**Hadoop/HDFS Infrastructure**: Tested our Hadoop infrastructure installation by moving a sample file into HDFS, performing basic file system operations on the HDFS directory and the uploaded file, and downloading the file from HDFS.

**Spark Infrastructure**: Tested our Spark infrastructure installation by running several provided spark queries and inspecting the spark web UI to verify the sample jobs are being run on our spark cluster. In particular, we used the sample query SparkPI, which computes a Monte Carlo approximation of PI across multiple spark nodes.

**Kafka Infrastructure**: Tested our Kafka infrastructure installation by creating a test Kafka topic, watching it with the kafka-console-consumer terminal utility, inserting sample data into the topic with the kafka-console-producer terminal utility, and verifying the messages showed up in the watching kafka-console-consumer terminal window.

**OSRM Infrastructure**: Tested our OSRM installation by running the web application configured against the OSRM instance and requesting a route within the City of Toronto.

**Streaming Job**: Tested our spark streaming job with the following:

1. Wrote integration tests for the database output component. This verifies the logic of the MongoDB update module when applied to a live database on the testing or development machine; these tests require a MongoDB installation on the testing computer.
2. Ran the implemented Spark streaming job, inserted a small amount of fake data into the source Kafka topic, and checked that the target database contained the expected ‘live’ road speeds. This confirmed that the job as a whole, including the sensor to OSM segment mapping component, functioned as specified.

**Routetagger**: Tested the Routetagger application by running it and:

1. Opening a handcrafted input file,
2. Tagged a route on a sensor in the file,
3. Verified the sensors were rendered in the appropriate colours,
4. Saved output file, and
5. Inspected the saved file in an editor and verified it had the expected columns and data.

**Web Application**: Tested the public-facing web application by navigating to mxpress.ca and using it. Tasked others to also use the website in order to complete some user testing. Some basic unit tests were also creating during development. Load testing was also done by generating as many routes as possible per second.

**OSRM Loader**: Tested the OSRM loader scripts module by running it and, with the web application, dragged endpoints on the map around to reload current route repeatedly even while OSRM was restarting. In this process, no errors occurred, indicating that OSRM restarts do not adversely affect pending requests from the app.

**System**: Tested the system as a whole by overriding the Kafka driver script to only report congestion on the highways for which we had data; on the web application we ran queries along this highway and verified that the planned routes avoided the ‘congested’ highways. In this test, a nearby highway for which we did not have data was also avoided and reported long commute times, indicating that the map of sensors to OSM segments we generated for testing took longer routes than desired on roads we did not intend to cover. This is acceptable as these sensor mappings were made from temporary testing data provided to us by the University of Toronto and do not represent the actual mappings for the ITSOS sensors.

# Limitations Encountered in Testing

The only functioning spark installation was on our server, so we could not develop and run tests locally. Our testing process for spark jobs was to code, deploy, and then test. This prevented automated tests relating to spark, which is core of our system. This is mainly caused by the team’s lack of experience with spark. With our recent work on containerization, will be able to do automated tests in future against a local deployment of a containerized spark cluster. The only automated tests were on the MongoDB component of the spark stream job because that component did not depend on any local spark infrastructure.

This project is an integration challenge, not a programming challenge, so most unit tests have limited usefulness and were not prioritized. However, lack of experience in writing integration tests and the need to gain experience in other areas made writing integration tests less feasible. This was the team’s first time using most of the frameworks used in this project.