In this activity, you will apply the knowledge from [Mini-Lesson 17.4](https://classroom.emeritus.org/courses/10605/pages/mini-lesson-17-dot-4-using-nifi-to-create-a-pipeline-in-mongodb-30-00) to create a NiFi workflow to read data from a MongoDB database and produce output files.

Before you begin this activity, ensure you have a NiFi *server* running inside a Docker *container*. You will also need a MongoDB *server* running inside a Docker *container*. Both the NiFi *server* and the MongoDB *server* need to be on the same Docker network.

Before starting this activity, review the submission instructions below to ensure that you collect the required screenshots as you progress through the activity.

Note that this activity has been tested using a Windows OS and the Catalina version of a Mac OS. If you use the Big Sur OS, you are recommended to use the myPhpAdmin *container* as demonstrated in this article: [Run MySQL & phpMyAdmin Locally Using Docker](https://migueldoctor.medium.com/run-mysql-phpmyadmin-locally-in-3-steps-using-docker-74eb735fa1fc)

[Links to an external site.](https://migueldoctor.medium.com/run-mysql-phpmyadmin-locally-in-3-steps-using-docker-74eb735fa1fc)

[Links to an external site.](https://migueldoctor.medium.com/run-mysql-phpmyadmin-locally-in-3-steps-using-docker-74eb735fa1fc)

.

**Reference**

[Yuste, Miguel. “Run MySQL & PhpMyAdmin Locally in 3 Steps Using Docker.” *Medium*. 2019. https://migueldoctor.medium.com/run-mysql-phpmyadmin-locally-in-3-steps-using-docker-74eb735fa1fc.](https://migueldoctor.medium.com/run-mysql-phpmyadmin-locally-in-3-steps-using-docker-74eb735fa1fc)

[Links to an external site.](https://migueldoctor.medium.com/run-mysql-phpmyadmin-locally-in-3-steps-using-docker-74eb735fa1fc)

**To complete this activity, follow these steps:**

1. Start the MongoDB and NiFi Docker *containers* running on the same network. Provide a screenshot of your Docker desktop to show the MongoDB and NiFi Docker *containers* running.
2. From the Docker desktop, select the CLI bash for the NiFi *container*. Create the output folder if it has not already been created. Provide a screenshot to show that the output *directory* exists in the NiFi *container*.
3. From the Docker desktop, select the CLI bash for the MongoDB *container*. Populate the MongoDB database by following Steps 3 and 4 in [Mini-Lesson 17.4](https://classroom.emeritus.org/courses/10605/pages/mini-lesson-17-dot-4-using-nifi-to-create-a-pipeline-in-mongodb-30-00). Provide a screenshot to show that you successfully populated the MongoDB database.
4. From the NiFi canvas, located at http://localhost/8080/nifi, create a newprocess group and name it Activity17.3. Provide a screenshot to show that you successfully created the new Activity17.3 process group.
5. In the newly created process group, add a new *processor* and select the GetMongo *processor* type. Provide a screenshot to show that you successfully added the GetMongo process group.
6. Configure the PROPERTIES tab in the GetMongo *processor* to connect to the MongoDB database. Provide a screenshot to show that you successfully configured the GetMongo *processor* to connect to the MongoDB database.
7. Add a PutFile *processor* to the NiFi canvas. Provide a screenshot to show that you successfully added the PutFile *processor*.
8. Configure the SETTINGS and PROPERTIES tabs in the PutFile *processor* to create files in the output *directory* as shown in [Mini-Lesson 17.4](https://classroom.emeritus.org/courses/10605/pages/mini-lesson-17-dot-4-using-nifi-to-create-a-pipeline-in-mongodb-30-00). Provide two screenshots. The first screenshot should show that you set the values in the SETTINGS tab correctly. The second screenshot should show that you updated the scheduling time in the PROPERTIES tab correctly.
9. Create a *connector* between the GetMongo and PutFile *processors*. Configure the *connector* as shown in [Mini-Lesson 17.4](https://classroom.emeritus.org/courses/10605/pages/mini-lesson-17-dot-4-using-nifi-to-create-a-pipeline-in-mongodb-30-00). Provide a screenshot to show that you successfully created and configured the *connector* between the GetMongo and PutFile *processors*.
10. In the NiFi UI, run the GetMongo and PutFile *processors*. Provide a screenshot of the NiFi canvas to show that the GetMongo and PutFile *processors* are running.
11. From the Docker desktop, select the CLI to open a bash window for the NiFi *server*. From the bash window, navigate to the /opt/nifi/nifi-current/output folder and list the files in the folder. Verify that the *process* is creating files. Provide a screenshot to show that the /opt/nifi/nifi-current/output *directory* has files that are being populated.
12. Use the cat command to display the contents of one of the files. Provide a screenshot to show that you successfully issued the cat command to display the contents of one file.

**Submission Instructions:**

Your submission for this activity should be a Word document that includes the following screenshots, each labeled for the step that the screenshot represents:

1. Provide a screenshot of your Docker desktop to show the MongoDB and NiFi Docker *containers* running.
2. Provide a screenshot to show that the output *directory* exists in the NiFi *container*.
3. Provide a screenshot to show that you successfully populated the MongoDB database.
4. Provide a screenshot to show that you successfully created the new Activity17.3 process group.
5. Provide a screenshot to show that you successfully added the GetMongo process group.
6. Provide a screenshot to show that you successfully configured the GetMongo *processor* to connect to the MongoDB database.
7. Provide a screenshot to show that you successfully added the PutFile *processor*.
8. Provide two screenshots. The first screenshot should show that you set the values in the SETTINGS tab correctly. The second screenshot should show that you updated the scheduling time in the PROPERTIES tab correctly.
9. Provide a screenshot to show that you successfully created and configured the *connector* between the GetMongo and PutFile *processors*.
10. Provide a screenshot of the NiFi canvas to show the GetMongo and PutFile *processors* running.
11. Provide a screenshot to show that the /opt/nifi/nifi-current/output *directory* has files being populated.
12. Provide a screenshot to show that you successfully issued the cat command to display the contents of one file.