**1. Create new app using docker:**

**Syntax: $ docker-compose run [service\_name] rails new . --force --no-deps**

**–database=[database\_name]**

Let service name is **web** and database is **MySQL:**

**Command: $ docker-compose run web rails new . --force --no-deps**

**–database=mysql**

**2. Change the ownership of file (Only for linux platform)**

Using Docker on Linux, **rails new** create files owned by root user. This happens because the container runs as the root user. If this is the case, change the ownership of the new files.

**Command:** **$ sudo chown -R $USER:$USER .**

**3. Add sidekiq gem in gemfile**

Write in ./Gemfile

**...**

**gem “sidekiq”**

**...**

**4. Change database.yml file**

Remove every thing and all the below line in ./config/database.yml

**For PostgreSQL Database:**

**default: &default**

**adapter: postgresql**

**encoding: unicode**

**pool: <%= ENV.fetch("RAILS\_MAX\_THREADS") { 5 } %>**

**username: <%= ENV['DB\_USERNAME'] %>**

**password: <%= ENV['DB\_PASSWORD'] %>**

**host: <%= ENV['DB\_HOST'] %>**

**For MySQL Database:**

**default: &default**

**adapter: mysql2**

**encoding: utf8mb4**

**pool: <%= ENV.fetch("RAILS\_MAX\_THREADS") { 5 } %>**

**username: <%= ENV['DB\_USERNAME'] %>**

**password: <%= ENV['DB\_PASSWORD'] %>**

**host: <%= ENV['DB\_HOST'] %>**

**development:**

**<<: \*default**

**database: test\_development**

**test:**

**<<: \*default**

**database: app\_test**

**production:**

**<<: \*default**

**database: app\_production**

**username: app**

**password: <%= ENV['APP\_DATABASE\_PASSWORD'] %>**

**4. Run build command again**

As we change the gemfile, We need to run build again

**Command:** **$ docker-compose build**

**5. Run Docker-compose up**

**Command:** **$ docker-compose up**