Development

For my development notes

Backup Bookstack Using Docker

Don't drive as root, create a BookStack administrator account by following the adduser instructions .

If you are running Bookstack in a docker container, run the following command to backup the database from **outside** the docker container. Navigate to the root of the running docker container, and run the following command. Don't forget to replace the `<INSERTS>` with actual file names, users, or other information regarding your Bookstack instance. `sudo docker exec <DOCKER-CONTAINER-NAME> /usr/bin/mysqldump -u <USER> -p <DATABASE> > <DATABASE>.backup.sql` so an example of this command would be the following -

```
sudo docker exec bookstack /usr/bin/mysqldump -u bookstackadmin -p bookstack_db > bookstack.backup.sql
```

This will output the file `bookstack.backup.sql` into your working directory, move this file to a safe place so you can restore the database should something go wrong in the future. Alternatively, you could manually enter the container with `sudo docker exec -it CONTAINER bash` and then just run `mysqldump -u USER -p DATABASE > DATABASE.backup.sql && exit` followed by `sudo docker cp CONTAINER:/container/path/DATABASE.backup.sql local/path` to copy the SQL backup onto our container's host.

This is all that needs to be done to backup the base content of BookStack, but there are some important configurations and upload directories you'll want to zip up, too. To zip these directories, enter your docker container and run the following commands

```
sudo docker exec -it BOOKSTACK_CONTAINER bash
tar -czvf bookstack-files-backup.tar.gz /var/www/html/.env /var/www/html/public/uploads
/var/www/html/storage/uploads
exit
sudo docker cp BOOKSTACK_CONTAINER:/var/www/html/bookstack/bookstack-files-backup.tar.gz
/home/USER/ftp/
```

If you used the method above, the database can be easily restored using the following commands

```
# Move our backup files into the containers that need them
sudo docker cp /home/USER/ftp/backup/bookstack.backup.sql BOOKSTACK_MYSQL_CONTAINER:/
sudo docker cp /home/USER/ftp/backup/bookstack-files-backup.tar.gz
```

```
BOOKSTACK_CONTAINER:/var/www/html/bookstack/

# Enter MySQL container and restore the DB
sudo docker exec -it BOOKSTACK_MYQL_CONTAINER bash
mysql -u {mysql_user} -p {database_name} < {backup_file_name}
exit

# Enter Bookstack container and restore local data
sudo docker exec -it BOOKSTACK_CONTAINER bash
tar -xvzf bookstack-files-backup.tar.gz
exit
```

If you are restoring to a new version of BookStack you will have to run `php artisan migrate` after restore to perform any required updates to the database. For safe keeping, toss this file somewhere so you can quickly peek at it whenever you need it. But once you run these commands a few times, you won't forget them.

```
#!/bin/bash
##Reference for backing up the BookStack database within docker container
################

# Backup Bookstack Database
sudo docker exec DOCKER_MYSQL_CONTAINER /usr/bin/mysqldump -u USER --password=PASSWORD DATABASE
> DATABASE.backup.sql

# Backup Bookstack Files
sudo docker exec -it DOCKER_CONTAINER bash
tar -czvf bookstack-files-backup.tar.gz .env public/uploads storage/uploads
exit
sudo docker cp CONTAINER:/var/www/html/bookstack/bookstack-files-backup.tar.gz /home/USER/ftp/

# Or manually copy them...
 # sudo docker cp BOOKSTACK_CONTAINER:/var/www/html/bookstack/.env /home/USER/ftp/
 # sudo docker cp BOOKSTACK_CONTAINER:/var/www/html/bookstack/storage/uploads /home/USER/ftp/
 # sudo docker cp BOOKSTACK_CONTAINER:/var/www/html/bookstack/public/uploads /home/USER/ftp/

# Restore
# Move our backup files into the containers that need them
 # sudo docker cp /home/USER/ftp/backup/bookstack.backup.sql BOOKSTACK_MYSQL_CONTAINER:/
 # sudo docker cp /home/USER/ftp/backup/bookstack-files-backup.tar.gz
BOOKSTACK_CONTAINER:/var/www/html/bookstack/
```

```

# Enter MySQL container and restore the DB
 # sudo docker exec -it BOOKSTACK_MYQL_CONTAINER bash
 # mysql -u USER -p DATABASE < DATABASE.backup.sql
 # exit

# Enter Bookstack container and restore local data
 # sudo docker exec -it BOOKSTACK_CONTAINER bash
 # tar -xvzf bookstack-files-backup.tar.gz
 # exit
```

Updating BookStack Using Docker

You can use volumes within docker-compose to store your configurations for your services, including other files and databases, on the host and pass them to the container to be used - this enables you to easily remove containers and purge images without worrying about backing up your data. As personal preference, I still run manual backups using the instructions shared on the Backup BookStack Page, but you shouldn't need to worry as the data is never touched when you are updating these images - the only concern would be if the image changes the way it uses the information, in which case you may need to change some paths for the volumes within your `docker-compose.yml`.

Run the below commands to update all services defined within your `docker-compose.yml`.

```
docker-compose down
docker container prune
docker image prune -a

#Verify you have no images left with your service, if you do, remove them.
docker images -a
docker-compose ps -a

#Once you verify the above, you are ready to spin up your services and pull the latest images.
docker-compose up -d
```

The above commands are more general and less specific to BookStack, and the same approach could be used to update containers for any service you are using, just be sure to persist and / or backup your data.

For this example, we run the commands to update BookStack -

```
user@bookstack:~/linuxserver-bookstack$ docker-compose down
Stopping bookstack    ... done
Stopping bookstack_db ... done
Removing bookstack    ... done
Removing bookstack_db ... done
Removing network linuxserver-bookstack_default
```

```
user@bookstack:~/linuxserver-bookstack$ docker image prune
WARNING! This will remove all dangling images.
Are you sure you want to continue? [y/N] y
Deleted Images:
untagged:
linuxserver/bookstack@sha256:d94bdeaea3eb9d2935e38s1dcca85450cdbd144706ccb6x78a2e75f0bde07
deleted: sha256:c32f40ccc7s51e508d2x2b241a00d529a35123a18d661b1edde15aac9bfee
deleted: sha256:5f2a803x93bb9fa8ab547c31d5a4f3e8520402d6425s85b53df80ebe515a9
deleted: sha256:84c9530d404015fc887c22xeecb1aebd99d2303aade5d48a2sfda044427f8
deleted: sha256:ff6dcf5097f8cfa05ff357d8axaf334c3a06259d97a6f2082cs8e7f280608

Total reclaimed space: 150.9MB
user@bookstack:~/linuxserver-bookstack$ docker-compose images
Container   Repository   Tag   Image Id   Size
----------------------------------------------
user@bookstack:~/linuxserver-bookstack$ docker-compose up -d
Creating network "linuxserver-bookstack_default" with the default driver
Creating bookstack_db ... done
Creating bookstack    ... done
user@bookstack:~/linuxserver-bookstack$
```

Exploring the Database

I'm running my Bookstack in a docker container, so for me if I want to explore my database the first step is to get a bash terminal within the container. Lets say you've named your bookstack database container `db_bookstack` - the command to enter a bash terminal within the container would be `docker exec -it db_bookstack bash`

Once you have a terminal, login to the database by running `mysql -u username -p` and enter the password to login when prompted. If you setup your bookstack with a docker container this should have been configured within your `docker-compose.yml` that you use to spin up your services.

Once your logged in, you'll see a mysql prompt like the below -

```
MariaDB [(none)]>
```

The prompt shows `(none)` because we haven't selected a database. Run `show databases;` to show the databases available to you and select one with `use database;`, see below for an example.

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| bookstack          |
| information_schema |
+--------------------+
2 rows in set (0.000 sec)

MariaDB [(none)]> use bookstack;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [bookstack]>
```

NOTE: The names of these databases do not reflect any actual databases

Now that we've selected the database we want to explore, let's see what it contains by running `show tables;` -

```
MariaDB [bookstack]> show tables;
+---------------------+
| Tables_in_bookstack |
+---------------------+
| activities          |
| api_tokens          |
| attachments         |
| books               |
| bookshelves         |
| bookshelves_books   |
| cache               |
| chapters            |
| comments            |
| email_confirmations |
| entity_permissions  |
| images              |
| joint_permissions   |
| migrations          |
| page_revisions      |
| pages               |
| password_resets     |
| permission_role     |
| role_permissions    |
| role_user           |
| roles               |
| search_terms        |
| sessions            |
| settings            |
| social_accounts     |
| tags                |
| user_invites        |
| users               |
| views               |
+---------------------+
29 rows in set (0.000 sec)
```

See something you want to explore further? Describe it by ruinng `describe table;`, like below where I describe the `books` table.

```
MariaDB [bookstack]> describe books;
+-------------+------------------+------+-----+---------+----------------+
| Field       | Type             | Null | Key | Default | Extra          |
+-------------+------------------+------+-----+---------+----------------+
| id          | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| name        | varchar(191)     | NO   |     | NULL    |                |
| slug        | varchar(191)     | NO   | MUL | NULL    |                |
| description | text             | NO   |     | NULL    |                |
| created_at  | timestamp        | YES  |     | NULL    |                |
| updated_at  | timestamp        | YES  |     | NULL    |                |
| created_by  | int(11)          | NO   | MUL | NULL    |                |
| updated_by  | int(11)          | NO   | MUL | NULL    |                |
| restricted  | tinyint(1)       | NO   | MUL | 0       |                |
| image_id    | int(11)          | YES  |     | NULL    |                |
+-------------+------------------+------+-----+---------+----------------+
10 rows in set (0.001 sec)
```

Neat! We have all the tools we need to step through any database that we have access to explore, and now we just need to be able to list the actual content within the tables of the database. Below, we run `select` and `from` commands that refer to specific tables and elements within our database. After the commands are input, the tables and their values we selected are displayed. See below for an example.

```
MariaDB [bookstack]> select id, name, created_at, updated_at, created_by from books;
+----+-------------------------+---------------------+---------------------+------------+
| id | name                    | created_at          | updated_at          | created_by |
+----+-------------------------+---------------------+---------------------+------------+
|  1 | Pi                      | 2019-04-06 00:00:35 | 2019-05-17 15:41:22 |          1 |
|  4 | Docker                  | 2019-04-06 00:02:12 | 2020-02-24 03:26:34 |          1 |
|  7 | BookStack               | 2019-04-06 04:41:38 | 2019-05-17 20:42:20 |          4 |
|  9 | Self-Hosted Applications | 2019-04-06 12:12:51 | 2019-07-06 14:17:31 |          4 |
| 13 | Networking              | 2019-04-19 06:48:57 | 2019-07-06 12:54:22 |          4 |
| 16 | Security                | 2019-05-04 15:02:19 | 2019-05-12 13:53:57 |          4 |
| 20 | Linux Admin             | 2019-07-06 12:57:22 | 2019-07-06 13:20:10 |          4 |
| 21 | Git                     | 2019-07-19 03:11:03 | 2019-07-26 21:54:29 |          4 |
| 23 | Vim                     | 2019-07-21 05:15:16 | 2019-08-30 08:22:47 |          4 |
| 24 | Ansible                 | 2019-07-28 08:17:04 | 2019-08-30 07:58:16 |          4 |
| 28 | C++                     | 2019-10-13 20:24:23 | 2020-03-01 14:31:16 |          4 |
+----+-------------------------+---------------------+---------------------+------------+
```