

# Wartung und Bedienung

- Terminal Cli Befehle
- Backup and Restore

# Terminal Cli Befehle

## Beschreibung:

Einige Dinge kann man in Bookstack nur über php programme einrichten im Terminal. Das php programm nennt sich artisan und ist in der BookStack installation zu finden.

Der standard Pfad lautet:

```
/var/www/Bookstack
```

Also würde ein aufruf

```
cd /var/www/Boackstack  
php artisan parameter
```

## Die Befehle:

### Create an Admin User

Create a new admin user via the command line. Can offer a good last resort if you ever get locked out the system. Will use the details provided as options otherwise will request them interactively.

```
# Interactive usage  
php artisan bookstack:create-admin  
  
# Non-interactive usage example  
php artisan bookstack:create-admin --email="barry@example.com" --name="Bazza" --password="hunter2"  
  
# Defining "External Authentication ID" instead of password for LDAP/SAML2/OIDC environments  
php artisan bookstack:create-admin --email="barry.booker@example.com" --name="Bazza" --external-auth-  
id="bbooker"
```

## Copy Shelf Permission

By default shelf permissions will not auto-cascade since a book can be in many shelves. This command will copy the permissions of a shelf to all child books. This can be done for a single shelf or for all shelves in the system:

```
# Run for all shelves
php artisan bookstack:copy-shelf-permissions --all

# Run for a single shelf
php artisan bookstack:copy-shelf-permissions --slug=my_shelf_slug
```

## Update System URL

BookStack will store absolute URL paths for some content, such as images, in the database. If you change your base URL for BookStack this can be problematic. This command will essentially run a find & replace operation on all relevant tables in the database. Be sure to take a database backup for running this command.

```
# Searches for <oldUrl> and replaces it with <newUrl>
php artisan bookstack:update-url <oldUrl> <newUrl>

# Example:
php artisan bookstack:update-url http://docs.example.com https://demo.bookstackapp.com
```

## Reset User MFA Methods

This will reset/clear any existing multi-factor-authentication methods for the given user. If MFA is enforced for one of their roles they'll be prompted to reconfigure MFA upon next login.

```
# Via email address:
```

```
php artisan bookstack:reset-mfa --email=john@example.com
```

```
# Via system ID:
```

```
php artisan bookstack:reset-mfa --id=5
```

## Refresh User Avatars

This will re-fetch avatar images for users in the system. By default avatars are fetched from Gravatar unless an alternative avatar system has been configured.

*Note: This will **not** load avatars from any connected authentication system.*

```
# Refresh avatar for just the user of the given email
```

```
php artisan bookstack:refresh-avatar --email=user@example.com
```

```
# Or the user of the given id
```

```
php artisan bookstack:refresh-avatar --id=101
```

```
# Refresh avatars for all users without avatars already assigned
```

```
php artisan bookstack:refresh-avatar --users-without-avatars
```

```
# Refresh avatars for all users in the system
```

```
php artisan bookstack:refresh-avatar --all
```

## Delete All Activity History

This will clear all tracked activities in the system. Note: Some areas of the interface rely on this data, including the Audit Log and any “Recent Activity” lists.

```
php artisan bookstack:clear-activity
```

## Delete Page Revisions

By default this deletes all page revisions apart from page update drafts which share the same system. A `-a` flag can be used to also delete these update drafts.

```
# Delete just the page revisions
php artisan bookstack:clear-revisions

# Delete all page revisions from the system including update drafts
php artisan bookstack:clear-revisions -a
```

## Delete Page Views

Delete tracked content views from the system. These are primarily used to populate any “Recently Used” lists.

```
php artisan bookstack:clear-views
```

## Cleanup Unused Images

Searches and removes images that are not used in page content. While this can be done in the “Maintenance” section of the interface, running this via the command-line is often safer to avoid timeouts.

```
php artisan bookstack:cleanup-images
```

## Regenerate the Search Index

BookStack uses a custom database-based search index system for efficient querying within the system. The command below re-builds this search index. This does not usually need to be manually ran, but it can be useful if manually inserting pages into the system or to pick-up BookStack indexing changes.

```
php artisan bookstack:regenerate-search
```

## Regenerate Access Permissions

BookStack pre-calculates and stores certain access permissions in the database so that access can be calculated in a performant manner. The below command will regenerate these permissions. This is primarily used in development but can be useful if manually adding content via the database.

```
php artisan bookstack:regenerate-permissions
```

## Regenerate Reference Index

BookStack stores references between content within the system to track how content is interlinked. Such references are generally managed by BookStack upon certain actions, such as when saving a page, but in some cases the below command may help to re-index these references. This can be useful upon upgrade of old content, or when manually adding content via the database.

```
php artisan bookstack:regenerate-references
```

## Delete Users

Delete all users from the system that are not original “admin” or system-level users.

```
php artisan bookstack:delete-users
```

## Generate UTF8mb4 SQL Upgrade Commands

Generates out the SQL which can be used to upgrade the database to UTF8mb4. See [UTF8mb4/Emoji Support](#) for further details.

php artisan bookstack:db-utf8mb4

# Backup and Restore

## Beschreibung:

Bookstack besteht aus Dateien und Der Datenbank

Hier liegen die Dateien bzw Verzeichnisse

- `.env` - File, contains important configuration information.
- `public/uploads` - Folder, contains any uploaded images.
- `storage/uploads` - Folder, contains uploaded page attachments.
- `themes` - Folder, contains any configured visual/logical themes.

## Backup Manuell:

Die Dateien sichern.

In das Verzeichnis gehen bei ner direkt installation:

```
cd /var/www/Bookstack
```

Bei ner docker installtion reicht das ganze Projektverzeichnis, das der Vorteil von der Docker.  
nun ein archiv erstellen

```
tar -czvf bookstack-files-backup-$(date +"%Y-%m-%d_%H-%M-%S").tar.gz .env public/uploads storage/uploads themes
```

Nun die Datenbank sichern

```
mysqldump -u root bookstackdb > bookstack.backup-$(date +"%Y-%m-%d_%H-%M-%S").sql
```

Diese beiden Dateien ins root Verzeichnisi vom neuen Server kopieren via scp zum Beispiel

## Restore/Migration:

Manuell:

Im Docker container, weil direkt installation machen wir in Zukunft nicht mehr:  
Falls der Container schon unglücklicherweise gestartet wurde den container app stoppen

```
docker-compose stop app
```

Das Datenverzeichnis löschen

```
rm -r /root/bookstack/bookstack_data
```

Nun in phpmyadmin alle Tabellen löschen, falls der Container schon gestartet wurde.

Nun die Datenbank restaurieren via phpmyadmin

Nun den app key aus der alten instance in die .env Datei eintragen.  
in einer nicht Docker installation liegt diese dann unter

```
cat /var/www/Bookstack/.env
```

Ein bisschen hochscrollen da steht der APP\_KEY diesen kopieren und in die neue .env einfügen /  
ersetzen

Nun die tar Datei hochladen die wir im Punkt backup erstellt haben

```
scp bookstack-files-backup-2025-02-02_14-23-51.tar.gz root@<ip>:/root/bookstack
```

Nun das archiv die Bilder entpacken nach bookstack\_data

```
tar -xf bookstack-files-backup-2025-02-02_14-26-12.tar.gz --strip-components=1 -C  
/root/bookstack/bookstack_data public/images
```

Nun die Files/attachments entpacken

```
tar -xzf bookstack-files-backup-2025-02-02_14-23-51.tar.gz --strip-components=2 -C  
/root/bookstack/bookstack_data/www/ storage/uploads
```

Nun die Pfade eventuell anpassen wenn die URL sich geändert haben sollte, sonst diesen Schritt  
überspringen

Dazu in die instance einloggen

```
docker-compose exec app bash
```

Nun ins Verzeichnis von bookstack gehen

```
cd /app/www
```

Nun mittels php artisan die Urls aktualisieren, als erstes die alte URL als zweiten Parameter die neue.

Falls die alte URL einen port mit dran ahtte zum docs.example.com:3000 dann diesen auch so angeben

```
# Searches for <oldUrl> and replaces it with <newUrl>
php artisan bookstack:update-url <oldUrl> <newUrl>

# Example:
php artisan bookstack:update-url http://docs.example.com https://demo.bookstackapp.com
```

Nun noch den cache löschen

```
php artisan cache:clear
```

Fertig

## Nachbearbeitung bei Manuell wie auch mit Backup Tool wenn das Ziel KEIN Docker ist:

Sollte nicht auf die gleiche Version migriert werden nachdem der Container gestartet ist, die migration ausführen das die Datenbank angepasst wird.

Dzu einmal in den container einloggen

```
docker-compose exec app bash
```

Dann diese Befehle ausführen

```
cd /app/www
php artisan migrate
```

Ausgabe:

```
php artisan migrate
```

APPLICATION IN

PRODUCTION.

└ Are you sure you want to run this command? ───────────┘

| ● Yes / ○ No |

└──┘

bestätigen