MySQL Backup and Restore System

Overview

This documentation provides guidance for administrators on how to manage backups and restores of the Book Management system's MySQL database. It covers manual and automated backup processes, restoration, log monitoring, and basic maintenance.

1. Backup Process

1.1 Manual Backup

An admin can trigger a manual backup using the following API:

- **Endpoint:** GET /api/backup
- **Authentication:** Requires a valid JWT token with admin privileges
- **Response:** Initiates a file download of the .sql backup file

Use Postman or a frontend interface with the token included in the Authorization header as:

```
Authorization: Bearer <admin_token>
```

1.2 Automated Backups

Backups are automatically created every 15 minutes using a cron job configured in the backend.

- Backup files are saved in the /backups directory.
- File naming format: auto_backup_<timestamp>.sql
- Only the last 7 days of backups are retained. Older files are automatically deleted.

No action is required from the admin to trigger this process.

2. Restore Process

2.1 Manual Restore

To restore the database from a .sql file:

- **Endpoint:** POST /api/restore
- **Authentication:** Requires a valid JWT token with admin privileges
- **Request Body:** multipart/form-data
    - Key: backupFile

  - Value: Upload a valid .sql file (created via the backup system)

Successful restores return a 200 OK response with the message “Database restored.” If the file is invalid or an error occurs, a proper error message is returned.

Note: Only .sql files generated by the backup system are guaranteed to be compatible with restore.

3. Logs

All automatic backup activities are logged to a file located at:

/logs/backup.log

This file includes timestamps for:

- Successful backup creations
- Deleted backup files older than 7 days
- Errors encountered during automated processes

4. Security and Access

- All backup and restore operations are protected by JWT authentication.
- Only users with an admin role can access these endpoints.
- Uploaded files are restricted to the /backups folder and are not accessible publicly.
- You can create a JWT token by running the login.js script ie. by using node login.js

5. Maintenance

- Ensure the server is running continuously for cron jobs to operate.
- Periodically verify that backups are created and the logs reflect accurate activity.
- Consider integrating cloud storage (such as AWS S3) for long-term backup retention if using a host with ephemeral storage.
- Monitor deployment logs or configure alerts to capture any unexpected failures.

6. Deployment

- The backup scheduler runs automatically when the server starts.
- If using Render or similar hosting, remember that the local file system is temporary. All backup files will be cleared on redeploy or restart unless saved externally.
- All backup and restore logic is defined in the scheduler/backupScheduler.js file, which is loaded from app.js.
- The backend server is deployed at https://bookmanagement-sc3n.onrender.com

7. Additional API End Points

- The server contains these additional CRUD api end points:
    - GET /api/books — Get list of all books
    - POST /api/books — Create a new book
    - PUT /api/books/:id — Update an existing book
    - DELETE /api/books/:id — Delete a book

7. Github Repository

- Github Repository link : https://github.com/gunagi-anish/Backup-Restore.git