**Test cases**

| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Status** |
| --- | --- | --- | --- | --- |
| TC\_SFS\_001 | Upload a valid file | 1. Login as a registered user  2. Select a valid file  3. Upload the file | File should be uploaded successfully and stored securely with encryption | Passed |
| TC\_SFS\_002 | Upload an invalid file type | 1. Login as a user  2. Attempt to upload a disallowed file type (e.g. .exe) | System should reject upload and show a proper error message | Passed |
| TC\_SFS\_003 | Download a file with valid secure link | 1. Login as a user  2. Generate a secure link for a file3. Use the link within allowed time | File should download successfully without unauthorized access | Passed |
| TC\_SFS\_004 | Attempt to download a file with an expired secure link | 1. Generate a secure download link  2. Wait for expiration time3. Attempt download | System should reject the request and show a 'link expired' message | Pending |
| TC\_SFS\_005 | Attempt to download a file with a tampered or invalid link | 1. Generate a secure link  2. Alter the token in URL manually  3. Attempt download | System should throw an 'Invalid or unauthorized request' error |  |
| TC\_SFS\_006 | Unauthorized access to download page | 1. Without login or valid link, access the download URL | System should redirect to login or display unauthorized access error | Passed |
| TC\_SFS\_007 | Test file deletion by authorized user | 1. Login as file uploader or admin  2. Delete uploaded file | System should securely delete file, remove record from database, and prevent further access via old download links | Passed |
| TC\_SFS\_008 | Attempt to delete a file by an unauthorized user | 1. Login as a different user  2. Attempt to delete a file uploaded by another user | System should reject the deletion attempt and display an error | Passed |

**How do you plan on deploying this to the production environment?**

For my SecureFileShare project, I’d focus on testing all the critical areas a real user might interact with. I’d test uploading valid and invalid files, check how the system handles large files, and confirm that the secure download links work as intended — including cases where the link is expired or tampered with. I’d also verify that encryption and decryption maintain file integrity by comparing the original and downloaded files. Access control is important too, so I’d make sure unauthorized users can’t download or delete files they shouldn’t have access to. Activity logs should correctly track every upload and download action for security auditing. Lastly, I’d perform basic performance testing by uploading multiple files in a row to see how the system handles load. I’d write these tests using Django’s TestCase or pytest for backend functions and use Postman for API checks. Overall, I aim to cover both expected and edge cases to make sure the system is reliable and secure before it’s deployed to production.