Django Application Security

Django Security Features

Cross Site Scripting Protection

- Django templates escape specific dangerous characters to stop users storing malicious scripts into the database which could be displayed to other users.
- Not foolproof however, and extra care should be taken if HTML is stored in a database.

Cross Site Request Forgery Protection

- Django has protections against a malicious actor emulating the credentials of a trusted user, by using a secret key with each POST, which is user specific.

SQL Injection Protection

- Django protects from a user injecting SQL code into the database to alter the data, through a field like a username or password, by escaping the SQL code in the database driver.

Security Features to Handle

The Use of HTTPS

- While Django allows the use of HTTPS, using an SSL certificate, it has to be set up by the user, and does not work out-the-box.
- A user must create and implement an SSL certificate, get it validated, and get it verified to be working.
- This is easily done using something like Let's Encrypt.

User Uploaded Files

- If there is any position where a user uploads a file to the server, Django does not contain any way of checking this is not malicious.
- The steps to stopping this would be mainly making sure the server cannot be tricked into executing code, this can be done by making sure the file type is not something that can execute code.
- The file should also be uploaded to a directory with as few permissions as possible, e.g. not a directory that has something like PHP enabled.

Denial Of Service Protection

- Django does not provide any protection to DoS Attacks. To protect against this, primarily the site should not allow repeated requests from a single host.
- To protect against a Distributed DoS, the site would most likely need some form of 3rd party help, using a service like Cloudfare.