1. HTTP operates without retaining memory of previous interactions, treating each client-server exchange as a standalone transaction with no stored user information. To maintain application continuity across multiple interactions—particularly for authentication and session tracking—web applications rely on tools like cookies, sessions, and tokens. Upon login, the server creates a unique session ID and transmits it to the client, usually as a cookie. With each following request, the client returns this ID to the server, enabling the server to link the request to the appropriate user session and preserve authentication status. As an alternative, contemporary applications often employ stateless tokens such as JWT (JSON Web Tokens), which contain user data and accompany each request, permitting the server to authenticate users without maintaining session information. These approaches allow web applications to deliver a seamless, customized user experience while working within HTTP's inherently stateless framework.

2.To execute Django database migrations on a server-based relational system such as MariaDB, begin by installing a compatible database driver like mysqlclient or PyMySQL within your Django setup. Then, modify the DATABASES configuration in your project's settings.py file to establish the MariaDB server connection, defining the ENGINE as 'django.db.backends.mysql' and providing details including database name, username, password, host address, and port number. Once the configuration is complete, execute python manage.py makemigrations to generate migration files reflecting your model modifications, followed by python manage.py migrate to implement these migrations on the MariaDB database. This workflow establishes the required tables and database structure in MariaDB, guaranteeing your Django models are accurately represented in the database. Adequate user permissions and network connectivity to the MariaDB server are essential for migration success.