**Geeko – A platform for geeks to share new ideas**

Overview:

Geeko is a web-based community platform designed for developers, engineers, and tech enthusiasts to collaborate, share projects, and engage in discussions. The platform provides a space where users can showcase their work, interact with like-minded individuals, and grow their technical skills.

Q1: Django Models / Database Tables (with foreign key)

Database Tables

The project includes the following Django models (SQL tables):

1. User Model (Custom user model extending AbstractBaseUser)
   * Fields: username, first\_name, last\_name, is\_active, is\_staff
   * Serves as the authentication model for the platform
2. Upload Model
   * Fields: user (ForeignKey to User), title, description, file, uploaded\_at
   * Represents projects uploaded by users
   * ForeignKey relation: Links each upload to a specific user
3. ContactSubmission Model
   * Fields: first\_name, surname, email, address, country, message, submitted\_at
   * Stores contact form submissions from users
4. Discussion Model
   * Fields: id (primary key), title, author (ForeignKey to User), created\_at
   * Represents discussion threads in the community forum
5. Reply Model
   * Fields: discussion (ForeignKey to Discussion), author (ForeignKey to User), text, created\_at
   * Stores replies to discussion threads
   * ForeignKey relations: Links to both Discussion and User models

Migration Files

The project includes all necessary migration files generated by Django's manage.py makemigrations command, ensuring the database schema matches the defined models when the project is run.

Q2 : Forms for typical CRUD (Create, Read, Update, Delete) operations

POST Request Forms

The project includes the following forms for CRUD operations:

1. User Registration Form (Create)
   * Path: /register
   * Creates new user accounts
   * Server-side validation for all fields
   * Client-side validation (via required HTML attributes)
2. Project Upload Form (Create)
   * Path: /upload
   * Creates new project entries
   * Server-side validation (via ModelForm)
   * Client-side validation in upload.js (file type, title length, description length)
3. Profile Update Form (Update)
   * Path: /profile (via modal)
   * Updates user profile information
   * Server-side validation in update\_profile\_ajax view
   * Client-side validation for username length
4. Profile Deletion Form (Delete)
   * Path: /profile (via modal)
   * Deletes user accounts
   * Server-side validation in delete\_profile\_ajax view
5. Contact Form (Create)
   * Path: /contact
   * Creates new contact submissions
   * Server-side validation in submit\_contact\_ajax view
   * Comprehensive client-side validation in contact.js
6. Discussion Creation Form (Create)
   * Path: /community
   * Creates new discussion threads
   * Server-side validation in create\_discussion view
   * Client-side validation for title length
7. Reply Form (Create)
   * Path: /community
   * Creates replies to discussions
   * Server-side validation in add\_reply view
   * Client-side validation for reply text

All forms show the results of their operations:

* Uploads appear on the projects page
* Profile changes are immediately visible
* Discussions and replies appear in the community forum
* Contact submissions can be viewed in the admin interface

Q3 : Client-side AJAX Functionality

**AJAX GET Request**

* **Discussion Loading**:
  + Path: /get\_discussions/
  + Loads all discussions and their replies on page load
  + Updates the discussion feed section of the page
  + Error handling shows a message if loading fails

**AJAX POST Requests**

1. **Contact Form Submission**
   * Path: /submit\_contact\_ajax
   * Creates new contact submissions
   * Updates the UI with success/error messages
   * Stores data in both database and local storage
2. **Profile Update**
   * Path: /update\_profile\_ajax/
   * Updates user profile information
   * Reloads the page on success to show changes
3. **Profile Deletion**
   * Path: /delete\_profile\_ajax/
   * Deletes user accounts
   * Redirects to homepage on success
4. **Discussion Creation**
   * Path: /create\_discussion/
   * Creates new discussion threads
   * Reloads discussions on success
5. **Reply Submission**
   * Path: /add\_reply/
   * Adds replies to discussions
   * Reloads discussions on success

All AJAX requests include proper error handling and CSRF token protection.

Q4 : User registration & login functionality

**User Registration**

* Path: /register
* Allows new users to create accounts
* Requires username, first name, last name, and password
* Includes validation for:
  + All fields required
  + Username minimum length (3 chars)
  + Password minimum length (6 chars)
  + Unique username

**User Login**

* Path: /login
* Authenticates users with username and password
* Redirects to homepage on success
* Shows error messages for invalid credentials

**User Logout**

* Path: /logout
* Logs out the current user
* Redirects to homepage with success message

**Staff Functionality**

* Custom User model includes is\_staff field
* Staff users can be created via Django admin
* Navigation bar shows different options for authenticated vs. anonymous users

Q5: Docker container, SQL Database connectivity & migrations

The project is configured to work with Docker and supports MySQL database connectivity through the DATABASE\_URL environment variable. The Docker setup includes:

* Proper Django configuration for database connections
* Volume for persistent storage
* Port mapping for web access

To run the project with Docker:

docker build -t myapp

docker volume create myapp-storage

docker run -ti -e DATABASE\_URL="mysql://myappdbuser:myappdbpass@host.docker.internal:3306/myappdb" -v myapp-storage:/app/storage -p 8000:8000 myapp

Q6: Overall Summary

Video ihas been already shared on the loop.

Additional Features

1. Dynamic Navigation Bar: JavaScript-powered navbar that responds to scrolling
2. Form Validation: Comprehensive client-side validation for forms
3. Local Storage: Contact form data stored in local storage
4. Hover Effects: Interactive hover effects on homepage cards
5. Responsive Design: Bootstrap-powered responsive layout

**Conclusion**

**The Geeko platform successfully implements all required functionality for Assignment 3, including database models, CRUD forms, AJAX functionality, user authentication, and Docker support.**