1η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ

Εφαρμογή ιστού για κοινοχρησία φωτογραφιών στοRuby-on Rails

Ονοματεπώνυμο και ΑΜ:

Σπύρος Τσότζολας 3099

➢ Σύνοψη:

Στην παρούσα εργαστηριακή άσκηση μας ζητήθηκε να επεκτείνουμε τον κώδικα της εφαρμογής treegram. Η treegram ήταν μια πολύ απλή εφαρμογή κοινοχρησίας φωτογραφιών που επέτερεπε σε έναν χρήστη να ανοίξει λογαριασμό, να εισέλθει και να ανεβάσει φωτογραφίες. Κάτω από μια φωτογραφία, υπάρχει ένα μενού με όλους τους εγγεγραμμένους χρήστες από τους οποίους ο χρήστης μπορεί να επιλέξει έναν για να τον προσθέσει ως ετικέτα(tag) στην φωτογραφία. Μας ζητήθηκε λοιπόν να επεκτείνουμε την εφαρμογή treegram με λειτουργίες που επιτρέπουν σε έναν χρήστη να προσθέτει τίτλο σε μια φωτογραφία, να ακολουθεί άλλους χρήστες, και να βλεπει τις φωτογραφίες που ανεβάστηκαν από τους ακολουθούμενους χρήστες, να μπορεί να σχολιάσει δικιές του φωτογραφίες ή φωτογραφιές χρηστών που ακολούθει και τέλος, ένας χρήστης να μπορεί να διαγράφει μια φωτογραφία που προηγουμένως δημοσίευσε με όλα τα σχόλιά και της ετικέτες της.Η υλοποίηση έγινε με χρήστη της γλώσσας Ruby, των βοηθητικών αρχείων και μεθόδων του Rails, και της γλώσσας Haml.

◆ Ερώτημα 1: Πρόσθεση τίτλου σε φωτογραφία

➢ Βήματα Υλοποίησης:

1. Αρχικά με την εντολή rails generate migration AddTitleToPhotos title:string δημιούργησα ένα καινούργιο migration για την εισαγωγή του πεδίου title στον πίνακα photo στην βάση μας.
2. Με την εντολή rake db:migrate αποθήκευσα την παραπάνω αλλαγή πηγαίνωντας στην συνέχεια στο αρχειο schema.rb για να δω ότι όντως είχε προσθεθεί το πεδίο title στον πίνακα photo.

3. Έπειτα πήγα στον κατάλογο app/models και στο αρχείο photo.rb πρόσθεσα την εντολη validate για τον τίτλο για να εξασφαλίσω την εγκυρότητα της βάσης μας
4. Στην συνέχεια πήγα στον κατάλογο app/controllers και πρόσθεσα στο υπάρχον αρχείο photos_controller.rb στην private method photo_params το πεδίο :title ώστε να είναι προσβάσιμο στην συνέχεια απο την create method που εχει την εντολή: Photo.create(photo_params) για την δημιουργία μιας νεας εγγραφης στον πίνακα photo.
5. Έπειτα πήγα στον κατάλογο app/views/photos και πρόσθεσα στο αρχείο new.html.haml τις εξής 2 εντολές: =form.label : title, =form.text_field : title ωστε κάτα την προσθήκη νέας φωτογραφίας απο τον χρήστη, αυτός να έχει την δυνατότητα να δώσει και εναν δικό του τίτλο στην φωτογραφία.
6. Τέλος στον κατάλογο app/views/users/ και στο αρχειο show.html.haml πρόσθεσα κάποιες γραμμές κώδικα ώστε να εμφανίζω τον τιτλό της φωτογραφίας στην κατάλληλη θεση(πάνω από την φωτογραφία).

- **Ερώτημα 2: Δυνατότητα στον χρήστη να ακολουθεί άλους χρήστες**

  Σημείωση: επέκτεινα την ιδέα και πρόσθεσα την δυνατότητα στον χρήστη να έχει και την επιλογή unfollow για άτομα που ήδη ακολουθεί.

  - **Βήματα Υλοποίσης:**

    1. Αρχικά με την εντολή rails g model Follow follower_id:integer followed_id:integer δημιούργησα το αντίστοιχο model και migration.
    2. Στην συνέχεια πήγα στο migration που δημουργήθηκε με την παραπάνω εντολή και πρόσθεσα indexes για τα πεδία follower_id και followed_id για να έχω πρόσβαση στον πίνακα follow με τα πεδία αυτα.
    3. Με την εντολή rake db:migrate πρόσθεσα τον παραπάνω πίνακα follow στην βάση μας πηγαίνωντας στην συνέχεια στο αρχειο schema.rb για να δω οτι όντως προσθέθηκε ο πίνακας.
    4. Έπειτα πήγα στον κατάλογο app/models και πρόσθεσα στο αρχείο user.rb τις εξής εντολές:

       has_many :active_follows, class_name: “Follow”, foreign_key: “follower_id”
       has_many :following, through: active_follows, source: :followed
       has_many :passive_follows, class_name: “Follow”, foreign_key: “followed_id”
       has_many :followers, through: passive_follows, source: :follower

Για να δηλώσω οτι ένας χρήστης μπορεί να έχει πολλά άτομα που ακολουθεί, όπως και επίσης και πολλά άτομα που τον ακολουθούν.

Επίσης στο μοντέλο αυτό πρόσθεσα και 3 συναρτήσης. Μια για να να κάνω μια νέα εισαγωγή στον πίνακα follow(follow επιλογή από τον χρήστη), μία άλλη για να κάνω μια διαγραφή στον πίνακα follow(unfollow επιλογή από τον χρήστη) και μια τελευταία για να ελεγχώ αν ένας χρήστης ακολουθεί εναν άλλο χρήστη.

5. Εν συνεχεία πήγα στον καταλογο app/models και στο αρχείο follow.rb πρόσθεσα τις εξής εντολές:

   belongs_to :follower, class_name: “User”

   belongs_to :followed, class_name: “User”

   validate :follower_id, presence: true

   validate :followed_id, presence: true

   Για να δηλώσω οτι ενας follower ή followed ανήκει σε έναν user και να συγουρέψω οτι στην βάση μας αποθηκεύονται μόνο έγκυρα δεδομένα.

6. Με την εντολη rails generate controller Follows δημιούργησα τον controller και μεσα σε αυτον προσθεσα δυο μεθοδους. Την create η οποια ειναι υπευθυνει και καλειται καθε φορα που ο χρηστης κανει καποιον αλλο χρηστη follow και την delete που καλειται οταν ο χρηστης επιλεξει την επιλογη unfollow για εναν χρηστη που ηδη ακολουθει. Οι μεθοδοι αυτοι καλουν τις αντιστοιχες μεθοδους για εισαγωγη και διαγραφει που περιεγραψα στο βημα4.

7. Στην συνεχεια στον καταλογο app/views/users και στο αρχειο show.html.hml προσθεσα ενα κουμπι για να με πεταει σε μια σελιδα με ολους τους χρηστες. Χρησιμοποιησα την συναρτηση index που ειναι υλοποιημενη στον users_controller και δημηουργοντας το αντιστοιχο αρχειο index.html.haml στον καταλογο app/views/users για να κανω την παραπανω ενεργεια.

8. Στο αρχειο αυτο views/users/index.html.haml αφου εμφανισα ολους τους εγκεγραμενους χρηστες προσθεσα κουμπι follow διπλα σε αυτους που ο τρεχον χρηστης δεν ακολουθει και unfollow κουμπι για αυτους που ηδη ακολουθει. Για να ελεγξω αν ο συγκεκριμενος χρηστης ακολουθει καποιον αλλο χρηστη χρησημοποιησα την μεθοδο που περιεγραψα στο βημα 4.

9. Τελος μεταφερθηκα στον καταλογο app/views/users και στο αρχειο show.html.haml για να εμφανισω ολες τις φωτογραφιες(αυτες του συνδεδεμενου χρηστη και αυτες των χρηστων που ακολουθει). Αρχικα θεωρησα σωστο να εμφανισω πανω πανω τις φωτογραφιες του συνδεδεμενου χρηστη για να εχει επιλογη να παει συντομα σε αυτες και να διαγραψει οποια θελει(ερωτημα μπονους) και στην συνεχεια εμφανιζονται ολες οι

φωτογραφιες(και δικιες του και των χρηστων που ακολουθει) σε σειρα αντιστροφη απο την σειρα εμφανισης των φωτογραφιων στην εφαρμογη.

- **Ερωτημα 3: Δυνατοτητα στον χρηστη να σχολιασει οποιαδηποτε φωτογραφια**

  - **Βηματα Υλοποιησης:**

    1. Αρχικα με την εντολη rails generate model Comment user:reference photo:reference body:text δημιουργησα το αντιστοιχο model και migration.
    2. Με την εντολη rake db:migrate αποθηκευσα τον παραπανω πινακα στην βαση μας.
    3. Με την εντολη rails generate controller Comments εφτιαξα τον controller μου και προσθεσα σε αυτον μια μεθοδο create η οποια ειναι υπευθυνει να εισαγει στην βαση μας στον πινακα comments την καταλληλη εγγραφη, και μια μεθοδο new η οποια απλα κανει redirect_to "/"
    4. Στην συνεχεα πηγα στον καταλογο app/model και στο αρχειο comment.rb και εβαλα τις εξης εντολες: belongs_to :user, belongs_to :photo για να δηλωσω οτι ενα σχολιο απαυθειναι σε μια συγκκριμενη φωτογραφια συγκεκριμενου χρηστη.
    5. Έπειτα στον καταλογο app/models και στα αρχεια user.rb και photo.rb προσθεσα την εντολη has_many_comments για να τονισω οτι ενας χρηστης μπορει να κανει πολλα σχολια και μια φωτογραφια μπορει να εχει πολλα σχολια.
    6. Ενσυνεχεία μεταφερθηκα στον καταλογο app/views/users και στο αρχειο show.html.haml και προσθεσα κατω απο καθε φωτογραφια ενα κουμπι "Add a comment.."
    7. Το path που εδωσα στο κουμπι με πηγαινε στον καταλογο app/controllers/photo_controller και συγκεκριμενα στην μεδοδο show που το αντιστοιχο view στον καταλογο app/views/photo/show.html.haml ηταν υπευθυνο να παρει το σχολιο απο τον χρηστη και αφου αποθηκευτει στον πινακα comment στην βαση μας να μας επιστρεψει στην αρχικη σελιδα με ολες τις φωτογραφιες και με την εμφανισει του συγκεκριμενου σχολιου που εδωσε ο χρηστης.

- **Ερωτημα Bonus: Ο χρηστης να εχει δυνατοτητα να διαγραφει δικιες του φωτογραφιες**

  - **Βηματα Υλοποιησης:**

    1. Αρχικα προσθεσα στον καταλογο app/controllers και στο αρχειο photo_controller μια επιπλεον μεθοδο την delete η οποια ειναι υπευθυνη για την διαγραφη μιας εγγραφης στον πινακα photo στην βαση μας.
    2. Εν συνεχεια μεταφερθηκα στο καταλογο model και στο αρχειο photo.rb και προσθεσα στις γραμμες has_many tags και has_many comments την εντολη dependent destroy ωστε καθε φορα που διαγραφεται μια φωτογραφια να διαγραφονται και τα αντιστοιχα comments και tags απο την βαση μας.
    3. Τελος μεταφερθηκα στον καταλογο app/views/users και στο αρχειο show.html.haml και προσθεσα ενα κουμπι διπλα απο τις φωτογραφιες του χρηστη που ειναι συνδεδεμενος, ωστε να του δωσω την δυνατοτητα να διαγραψει καποια που δεν θελει πλεον. Το path που εδωσα στο κουμπι με πηγαινε στον καταλογο app/controllers/photo_controller και στην συγκεκριμενα στην μεθοδο που περιεγραψα στο βημα 1.

Σχετικα με τα routes:

Στο αρχειο routes.rb προσθεσα τις εξης εντολες:

1.μεσα στο resourse :users do εβαλα: resources :follows, only: [:create]

2. resources :follows, only: [:destroy]

3. recources :comments, only: [:create, :destroy]

Διευκρινηση:

Στην παραπανω αναφορα οπου αναφερω οτι προσθεσα κουμπι σε καποια σελιδα, αυτο το εκανα με την βοηθητικη εντολη:

link_to ‘name_btn’, path, :class[...]