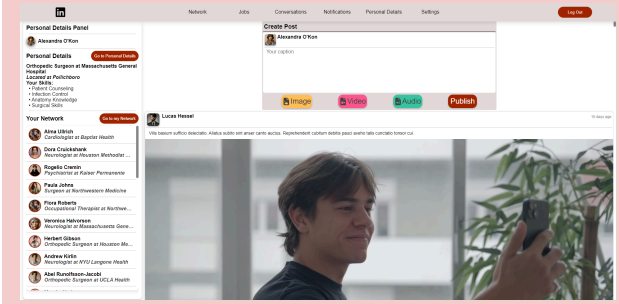
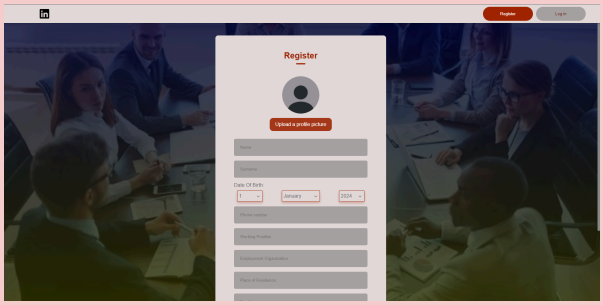
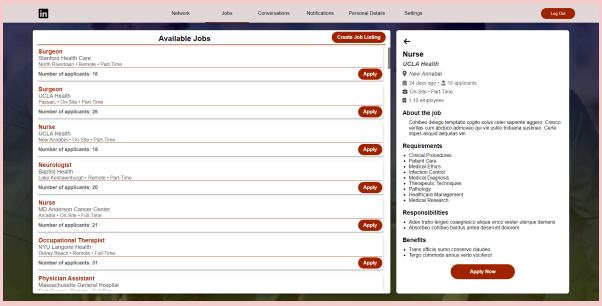
Τεχνολογίες Εφαρμογών Διαδικτύου
Εαρινό Εξάμηνο 2024 Εργασία (Linkedin)

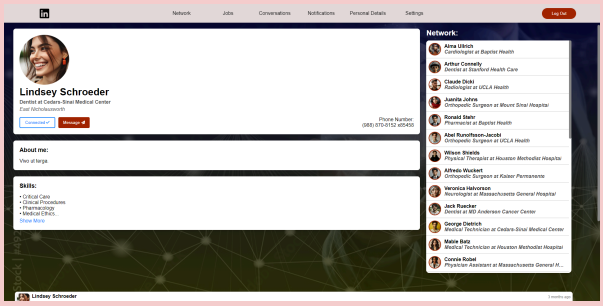
Ομάδα:

- Αλέξανδρος Θεοφυλάκτου - 1115202100220
- Νικόλαος Μιχαλούτσος - 1115202000133









Περιεχόμενα:

1. **<u>Στόχος:</u>**
   Ο στόχος της εργασίας είναι η ανάπτυξη εφαρμογής επαγγελματικής δικτύωσης. Οι χρήστες θα έχουν πρόσβαση στην εφαρμογή μέσω σύγχρονου φυλλομετρητή παγκόσμιου ιστού (web browser).

2. **<u>Τεχνολογίες:</u>**
   Οι τεχνολογίες που χρησιμοποιήθηκαν στην υλοποίηση της εργασίας είναι οι παρακάτω:

- **MongoDB** (Μη-σχεσιακή Βάση Δεδομένων) σε συνδυασμό με Google Cloud Storage (Αποθήκευση Αρχείων)
- **Express.js** (Node.js Web Server)
- **React.js** (JavaScript Frontend Framework)
- **Node.js** (JavaScript Web Server)

3. **<u>Περιεχόμενο που ακολουθεί:</u>**
   Στα παρακάτω κεφάλαια θα ακολουθήσει περιγραφή για κάθε ξεχωριστό τομέα της εφαρμογής μαζί με μερικές λεπτομέρειες.

4. **<u>Μετωπιαίο Άκρο (Frontend):</u>**
   Στο frontend, οπως προαναφέρθηκε χρησιμοποιήθηκε το **JavaScript React Framework**. Για τις σχεδιαστικές μας επιλογές ακολουθήσαμε την εκφώνηση καθώς και πήραμε ιδέες από ήδη υπάρχον, διάσημες εφαρμογές κοινωνικής δικτύωσης όπως *Facebook, Linked In, Instagram* κλπ.
   Εστιάσαμε τη σχεδίαση της κάθε σελίδας στο να είναι **user-friendly** αλλά και να παρέχει μία **ευχάριστη/ξεκούραστη εμπειρία**, με την χρήση **smooth transitions**, **pop ups**, όπως και **reusable components** τα οποία είτε δημιουργήσαμε εμείς (ManyInputFields, InteractiveProfile, NetworkUsersList κλπ.), είτε τα πήραμε έτοιμα από βιβλιοθήκες της React (Select για dropdown menus, DragAndDrop πεδίο για ανέβασμα αρχείων κλπ.).
   Κρατήσαμε ένα ενιαίο θέμα σε όλη την εφαρμογή, το οποίο είναι κοινό σε όλες τις σελίδες.
   Θεωρήσαμε πως μέσα στους στοχους της εφαρμογής συμπεριλαμβάνεται και η **προώθηση της κοινωνικής / επαγγελματικής δικτύωσης** όπως και καταφέρνουμε εντός των σελίδων, προσφέροντας συνεχώς επιλογές στον χρήστη για

πλοήγηση σε προφίλ άλλων χρηστών, αποδοχή/αποστολή αιτημάτων και μηνυμάτων και προβολή περισσότερων δεδομένων με ένα κλίκ. Ένας άλλος τρόπος που επιτυγχάνεται αυτός ο στοχος είναι μέσω των συνιστώμενων άρθρων (μπόνους ερώτημα) προωθώντας στον χρήστη άρθρα χρηστών με τους οποίους πιθανόν να έχει κοινά ενδιαφέροντα ή και να γνωρίζει, αφού το δίκτυο του αλληλεπιδρά με αυτά. Επίσης, ο χρήστης στην εφαρμογή μπαίνει τόσο για να δικτυωθεί κοινωνικά/επαγγελματικά όσο και για να βρει νέες επαγγελματικές ευκαιρίες. Για αυτο και επιδιώκουμε να παραμείνει/επιστρέψει στην εφαρμογή συνιστώντας του δουλειές που ταιριάζουν με τις δεξιότητες του, όσο και δουλειές που προτείνονται βάσει των αλληλεπιδράσεων των συνδεδεμένων με αυτόν χρηστών (μπόνους ερώτημα).

5. **<u>Νωτιαίο Άκρο (Backend):</u>**
   Στο backend**,** ακολουθήσαμε τη δομή **RESTful API**, διαχωρίζοντας τις ασχολίες του frontend και του backend. Το backend αναλαμβάνει την επεξεργασία και μεταφορά των δεδομένων από τη βάση στο frontend. Για την υλοποίηση του **server**, χρησιμοποιούμε τον συνδυασμό **Node.js - Express.js.**
   Για την υλοποίηση της **βάσης δεδομένων**, χρησιμοποιούμε το **mongoose**, το οποίο είναι ένα εργαλείο που επιτρέπει την αλληλεπίδραση JavaScript με MongoDB.

   <u>Φάκελος models:</u> Περιέχει τα model.js αρχεία, τα οποία αναπαριστούν την αντίστοιχη συλλογή στην βάση δεδομένων.

   <u>Φάκελος controllers:</u> Περιέχει τα controller.js αρχεία, τα οποία περιέχουν τις ασύγχρονες συναρτήσεις που θα κληθούν ανάλογα με κάθε αίτημα (request) από τον πελάτη (client), και επιστρέφουν την αντίστοιχη απάντηση (response).

   <u>Φάκελος routes:</u> Περιέχει τα routes.js αρχεία, τα οποία είναι υπεύθυνα για να ανακατευθύνουν τα αιτήματα (requests) στην αντίστοιχη τους συνάρτηση στα controller αρχεία, αφού πρώτα περάσουν τα απαραίτητα ενδιάμεσα λογισμικά (middleware).

Φάκελος middleware: Περιέχει τα αρχεία για τα ενδιάμεσα λογισμικά (middleware), τα οποία επιτρέπουν να μεταφερθεί κάποιο αρχείο μαζί με το request, στο fileUpload.js, και προστατεύουν τα routes από το να μπορούν να χρησιμοποιηθούν από κάποιο χρήστη ο οποίος δεν είναι συνδεδεμένος, δηλαδή, δεν κατέχει ένα έγκυρο **Json Web Token** μέσω του requireAuth.js.

Google Cloud Storage: Με το setup ενός **storage bucket**, και την εισαγωγή του στην εφαρμογή, έχουμε τη δυνατότητα να αποθηκεύουμε αρχεία στο συγκεκριμένο bucket, και να αποθηκεύουμε στη βάση δεδομένων το **link** στα αρχεία αυτά αντί να τα αποθηκεύουμε απευθείας στο MongoDB ως BLOBs (Binary Large OBjects)

Κρυπτογράφηση HTTP αιτημάτων: Η ασφάλεια των HTTP αιτημάτων επιτυγχάνεται με τη χρήση ενός self-signed SSL certificate από openssl (φάκελος SSL).

6. **Βάση Δεδομένων (Database):**
   Η βάση δεδομένων μας αποτελείται από λίγα και συνεκτικά/περιεκτικά collections (συλλογές). Οι συλλογές μας είναι ονομαστικά οι:

   - Admins
   - Users
   - Posts
   - Jobs
   - Comments
   - Post Notifications
   - Conversations

   Οι συλλογές αυτές περιγράφονται καταλλήλως από το όνομα του αλλά θα αναλυθούν περισσότερο παρακάτω:

   - **Admins**
     Η συλλογή αυτή αποτελείται από αντικείμενα τα οποία αναπαριστούν τους λογαριασμούς διαχειριστών. Τα αντικείμενα αυτά αποτελούνται από:

- Email (Ηλεκτρονική διεύθυνση)
- Password (Κωδικός)

Ο διαχειριστής μπορεί μέσω του Welcome Page στην επιλογή Login να συνδεθεί στον λογαριασμό που αντιστοιχούν στο email και password που πληκτρολόγησε στην φόρμα.

- **Users**
  Η συλλογή αυτή αποτελείται από αντικείμενα που αναπαριστούν τους λογαριασμούς χρηστών. Τα αντικείμενα αυτά αποτελούνται από:

  - Email (Ηλεκτρονική διεύθυνση)
  - Password (Κωδικός)
  - Name (Όνομα)
  - Surname (Επώνυμο)
  - Date of Birth (Ημερομηνία γέννησης)
  - Profile picture (Φωτογραφία προφίλ)
  - Phone number (Αριθμός τηλεφώνου)
  - Working position (Θέση εργασίας)
  - Employment organization (Εταιρία όπου εργάζεται)
  - Place of residence (Μέρος κατοικίας)

  Και άλλα δεδομένα από τα οποία αποτελείται και βοηθούν στην λειτουργικότητα της εφαρμογής. Ο χρήστης έχει την δυνατότητα να εγγραφεί/συνδεθεί στην εφαρμογή, να επεξεργαστεί τα προσωπικά του στοιχεία, να μοιραστεί τις σκέψεις του μέσω άρθρων κλπ.

- **Posts**
  Η συλλογή αυτή αποτελείται από αντικείμενα που αναπαριστούν τα δημοσιευμένα άρθρα. Τα αντικείμενα αυτά αποτελούνται από:

  - Author (Συγγραφέας)
  - Caption (Λεζάντα)
  - Multimedia Type (Τύπος πολυμέσου)

- Multimedia URL (Ηλεκτρονική διεύθυνση πολυμέσου)
- Comments List (Λίστα σχολίων)
- Likes List (Λίστα ενδιαφερόντων χρηστών)
- createdAt (Ημερομηνία και ώρα ανάρτησης)

Οι χρήστες μπορούν να αναρτήσουν ένα άρθρο ή να δηλώσουν ενδιαφέρον/σχολιάσουν σε άλλα άρθρα.

- **Jobs**
  Η συλλογή αυτή αποτελείται από αντικείμενα που αναπαριστούν τις δημοσιευμένες αγγελίες. Τα αντικείμενα αυτά αποτελούνται από:

  - Author (Συγγραφέας)
  - Title (Τίτλος)
  - Employer (Εταιρία/Εργοδότης)
  - Description (Περιγραφή)
  - Location (Περιοχή)
  - Requirements (Απαιτήσεις)
  - Benefits (Οφέλη)
  - Responsibilities (Απαιτούμενες δεξιότητες)

  Οι χρήστες μπορούν να αναρτήσουν μία αγγελία ή να κάνουν αίτηση σε άλλες αγγελίες.

- **Comments**
  Η συλλογή αυτή αποτελείται από αντικείμενα που αναπαριστούν τα δημοσιευμένα σχόλια σε άρθρα. Τα αντικείμενα αυτά αποτελούνται από:

  - Author (Συγγραφέας)
  - Post (Άρθρο στο οποίο υποβλήθηκε το σχόλιο)
  - Content (Περιεχόμενο)
  - createdAt (Ημερομηνία και ώρα ανάρτησης)

  Οι χρήστες μπορούν να υποβάλουν ένα ή περισσότερα σχόλια σε δημοσιευμένα άρθρα.

- **Post Notifications**
  Η συλλογή αυτή αποτελείται από αντικείμενα που αναπαριστούν τις ειδοποιήσεις του χρήστη από δηλώσεις ενδιαφέροντος ή υποβολή σχολίων στα άρθρα του. Τα αντικείμενα αυτά αποτελούνται από:

  - Author (Συγγραφέας σχολίου ή ενδιαφέροντας χρήστης)
  - Post_id (Άρθρο στο οποίο αναφέρεται η ειδοποίηση)
  - Comment Content (Περιεχόμενο σχολίου αν αυτό υπάρχει)
  - isRead (Αν έχει διαβαστεί η ειδοποίηση ή οχι)
  - isLike (Αν η ειδοποίηση αναφέρεται σε Like ή Comment)
  - createdAt (Ημερομηνία και ώρα δημιουργίας)

  Οι χρήστες θα ειδοποιηθούν καταλλήλως όταν κάποιος χρήστης δηλώσει ενδιαφέρον ή σχολιάσει σε κάποιο δημοσιευμένο άρθρο του.

- **Conversations**
  Η συλλογή αυτή αποτελείται από αντικείμενα που αναπαριστούν μια συνομιλία μεταξύ δύο χρηστών. Τα αντικείμενα αυτά αποτελούνται από:

  - Participant_1 (Συμμετέχοντας 1)
  - Participant_2 (Συμμετέχοντας 2)
  - Message Log (Λίστα από μηνύματα)
  - createdAt (Ημερομηνία και ώρα δημιουργίας)

  Όταν οι χρήστες ξεκινήσουν μια νεα συνομιλία, ένα τέτοιο αντικείμενο θα δημιουργηθεί.

7. **Bonus (Matrix Factorization Collaborative Filtering):**

Γενική ιδέα: Δημιουργία πίνακα συσχετίσεων χρηστών - αντικειμένων βάσει της αλληλεπίδρασης τους με αυτά. Στη συνέχεια, στις αλληλεπιδράσεις του κάθε χρήστη προστίθεται ένα ποσοστό των αλληλεπιδράσεων των συνδεδεμένων με αυτόν χρηστών.
Ο αλγόριθμος γίνεται train πάνω στις μη-μηδενικές τιμές αλληλεπίδρασης, με αποτέλεσμα να “μαντέψει” τιμές αλληλεπίδρασης στα αντικείμενα που δεν έχει αλληλεπιδράσει ο χρήστης ακόμη.

Επικοινωνία με Node.js server: Από το Node.js server, καλείται η συνάρτηση matrixFactorization() με όρισμα ένα path στο αντίστοιχο python script.
Ανάλογα με το όρισμα, η συνάρτηση αναλαμβάνει να μαζέψει τα δεδομένα των χρηστών και των αντίστοιχων αντικειμένων, και να τα τοποθετήσει σε συμφωνημένα αρχεία, για να μπορούν να διαβαστούν από τα python scripts.
Υπάρχουν δύο python scripts, ένα υπεύθυνο για τις αγγελίες και ένα υπεύθυνο για τα άρθρα. Το κάθε script διαβάζει από τα συμφωνημένα αρχεία, και δημιουργεί τον πίνακα συσχετίσεων βάσει αυτών.
Στη συνέχεια δημιουργεί ένα αντικείμενο κλάσης MF, με τα δεδομένα αυτά και τις αντίστοιχες παραμέτρους και το κάνει train έτσι ώστε να "μάθει" τις κρυμμένες ιδιότητες στον πίνακα αυτό και να μπορεί να προτείνει με ακρίβεια αντικείμενα τα οποία είναι ταιριαστά στα δεδομένα.
Αφού το μοντέλο εκπαιδευτεί, για κάθε χρήστη, υπολογίζονται τα προτεινόμενα αντικείμενα, καταγράφονται σε άλλο, επίσης συμφωνημένο αρχείο, το οποίο θα διαβάσει το Node.js server μολις λάβει το μήνυμα *"MF DONE"* από το stdout του python script.
Αφού λάβει το μήνυμα, διαβάζει το περιεχόμενο των αρχείων και αποθηκεύει στη βάση δεδομένων τα προτεινόμενα αντικείμενα σε κάθε χρήστη.
Αυτή η διαδικασία, τρέχει περιοδικά, κάθε 3 ώρες που είναι ενεργός ο server.

Δημιουργία Τεχνητού Dataset: Για τη δημιουργία τεχνητού dataset, χρησιμοποιήσαμε μερικά scripts τα οποία γράψαμε για να δημιουργούν χρήστες, άρθρα, σχόλια, αγγελίες και σημειώσεις ενδιαφέροντος. Τα scripts αυτά βρίσκονται στον φάκελο dataGeneration στο backend. Μερικά στοιχεία των δεδομένων που θα δημιουργηθούν παράγονται από τη βιβλιοθήκη faker, και τα υπόλοιπα, έχουμε αναλάβει να δημιουργούνται βάσει μερικών παραδοχών και επιλογών που αποφασίσαμε.
Παρακάτω, θα αναπτυχθεί σε συντομία ο τρόπος δημιουργίας κάθε αντικειμένου, για περισσότερη λεπτομέρεια και καλύτερη κατανόηση, προτείνεται να δείτε τον κώδικα, στον αντίστοιχο φάκελο.

Θεωρήσαμε πως υπάρχουν 10 επαγγελματικές κλάσεις, με συγκεκριμένα πεδία η κάθε μία, τα οποία χρησιμοποιούνται για να δημιουργηθούν τα δεδομένα. Οι κλάσεις σε ανάπτυξη, βρίσκονται στο αρχείο classes.json στον φάκελο dataGeneration στο backend.

Δημιουργία Χρηστών: Αριθμός χρηστών: 1000.
Όλοι οι χρήστες έχουν τον ίδιο κωδικό, Password123! Για το λόγο ότι ο κωδικός που αποθηκεύεται στη βάση δεδομένων είναι hashed, οπότε θα ήταν αδύνατο να τον βρούμε έτσι ώστε να μπορούμε να έχουμε πρόσβαση στους χρήστες.
Για τον κάθε χρήστη, αποφασίζεται το φύλο, και επιλέγεται όνομα, επίθετο και φωτογραφία προφίλ δεδομένου αυτού.
Αποφασίζεται επίσης μία κλάση από τις 10, στην οποία θα ανήκει ο επαγγελματικός προσανατολισμός του. Σύμφωνα με αυτό, δημιουργούνται οι δεξιότητες του, και το δίκτυο του (αφού πρώτα δημιουργηθούν όλοι οι χρήστες).

Δημιουργία Αγγελιών: Αριθμός αγγελιών: 800
Οι αγγελίες δημιουργούνται επιλέγοντας μία από τις 10 επαγγελματικές κλάσεις, και τους ορίζονται οι αντίστοιχες απαιτούμενες δεξιότητες. Επιλέγεται ένας τυχαίος χρήστης σαν author της αγγελίας.

Δημιουργία Άρθρων: Αριθμός άρθρων: 1000
Τα άρθρα δημιουργούνται αυθαίρετα, και ορίζεται ένας τυχαίος χρήστης ως author, εφόσον δεν περιέχουν κάποιο επαγγελματικό περιεχόμενο. Το αν το άρθρο θα περιέχει κάποιου είδους πολυμέσα, αποφασίζεται τυχαία, και στη συνέχεια αποφασίζεται τυχαία από ένα σύνολο προκαθορισμένων αρχείων ποιο από αυτά θα ανεβεί.

Δημιουργία Σχολίων: Αριθμός σχολίων: 2000
Τα σχόλια στα άρθρα δημιουργούνται από κάποιο χρήστη, σε ένα άρθρο το οποίο θα ήταν πιθανό να εμφανιστεί στο χρονολόγιο του βάσει της εκφώνησης (συνδεδεμένος χρήστης ή liked από κάποιο συνδεδεμένο χρήστη). Το περιεχόμενο δημιουργείται από τη βιβλιοθήκη faker.

Δημιουργία Σημειώσεων ενδιαφέροντος: Αριθμός σημειώσεων ενδιαφέροντος: 2500

Οι σημειώσεις ενδιαφέροντος στα άρθρα δημιουργούνται από κάποιο χρήστη, σε ένα άρθρο το οποίο θα ήταν πιθανό να εμφανιστεί στο χρονολόγιο του βάσει της εκφώνησης (συνδεδεμένος χρήστης ή liked από κάποιο συνδεδεμένο χρήστη).

8. **<u>Οδηγίες εγκατάστασης και εκτέλεσης:</u>**
   Η εφαρμογή είναι χωρισμένη σε 3 κύριους φακέλους.

   <u>frontend:</u> Απαιτείται η εντολή ***npm install*** σε ένα terminal στο συγκεκριμένο φάκελο, για να εγκατασταθούν οι απαραίτητες βιβλιοθήκες.
   Για εκκίνηση του frontend, χρησιμοποιείται η εντολή ***npm start***.

   <u>backend:</u> Απαιτείται η εντολή ***npm install*** σε ένα terminal στο συγκεκριμένο φάκελο, για να εγκατασταθούν οι απαραίτητες βιβλιοθήκες.
   Για την εκκίνηση του backend server, χρησιμοποιείται η εντολή ***npm run dev*** (για να κάνει reload σε κάθε save) ή απλά η εντολή ***npm start***.

   <u>matrixFactorization:</u> Απαιτείται η εγκατάσταση της python με τις βιβλιοθήκες numpy και dotenv. Υπάρχει εσωτερικά ένα virtual python environment στο οποίο και πρέπει να εγκατασταθούν οι παραπάνω βιβλιοθήκες.

   <u>Γενικά:</u>
   Και στους 3 φακέλους, υπάρχει ένα αντίστοιχο αρχείο .env, το οποίο πρέπει να είναι σωστά διαμορφωμένο, έτσι ώστε να οδηγεί στους απαραίτητους πόρους. Στη δική μας περίπτωση, μερικά από τα μονοπάτια είναι local στους υπολογιστές μας, προτείνεται πριν την εκτέλεση να σιγουρευτείτε ότι τα μονοπάτια στα .env αρχεία δείχνουν στα σωστά paths στον υπολογιστή σας.

9. **Επίλογος:**

Δυσκολίες και τρόποι αντιμετώπισης:

- **Αποθήκευση/φόρτωση πολυμέσων αρχείων**
  Μια από τις δυσκολίες που αντιμετωπίσαμε ήταν η **αποθήκευση/φόρτωση πολυμέσων αρχείων**. Μετά από κάποιες μέρες έρευνας, καταλάβαμε ότι το MongoDB, δεν είναι κατάλληλο/αποδοτικό στο να αποθηκεύει images/videos/audios (πολυμέσα αρχεία). Μάλιστα, αποθαρρύνεται και απο το ίδιο το MongoDB να αποθηκεύει κανείς γενικότερα αρχεία στη βάση δεδομένων. Για αυτόν τον λόγο χρησιμοποιούμε το Google Cloud Storage για την αποθήκευση τους. Έτσι, όλα τα αρχεία αποθηκεύονται στο Google Cloud Storage και στο MongoDB αποθηκευεται το path (μονοπάτι) σε αυτα τα αρχεία, λύνοντας το πρόβλημα που αντιμετωπίζαμε. Όλα αυτά υλοποιούνται μέσω middlewares που προαναφέρθηκαν παραπάνω.

- **Matrix Factorization Collaborative Filtering**
  Μια άλλη δυσκολία που αντιμετωπίσαμε της οποίας η λύση αποδείχτηκε πολύ χρονοβόρα, ήταν η υλοποίηση του **Matrix Factorization Collaborative Filtering**. Κάνοντας έρευνα στο διαδίκτυο η κατανόηση του αλγορίθμου δεν ήταν δύσκολη, αλλά το να κατανοήσει κανείς τις παραλλαγές που απαιτούνταν από την εργασία, λαμβάνοντας υπόψην το δίκτυο των χρηστών του χρήστη. Εκείνο που όμως αποδείχτηκε πραγματικά χρονοβόρο, είναι η προσαρμογη στα δεδομένα μας των οποίων η φύση είναι διαφορετική των περισσότερων αλγορίθμων που εξηγούνται στο διαδίκτυο. Η κλασική μορφή και το κλασικό παράδειγμα του Matrix Factorization Collaborative Filtering περιέχει δεδομένα με ratings όπως αυτά του *Netflix*. Όμως, τα δικά μας δεδομένα είχαν δυαδική μορφή (υπάρχει αλληλεπίδραση ή δεν υπάρχει αλληλεπίδραση). Δοκιμάστηκαν υλοποιήσεις όπως BMF (*Binary Matrix Factorization*) υλοποιημένο με sigmoid και binary cross-entropy αλλά δεν είχαμε τα επιθυμητά αποτελέσματα. Δοκιμάστηκε επίσης υλοποίηση με SVD MF (*Singular Value Decomposition Matrix Factorization*) όμως τα αποτελέσματα πάλι δεν ήταν επιθυμητά. Το πρόβλημα τελικά λύθηκε δίνοντας διαφορετική βαρύτητα σε κάθε αλληλεπίδραση αναλόγως το είδος της, και έτσι ανάγοντας

το σε μη δυαδική μορφή. Ύστερα από αυτό, υλοποιώντας τον κλασικό αλγόριθμο είχαμε τα αποτελέσματα που επιθυμούσαμε.

Μερικές Σημειώσεις:

ADMIN:

- **Email: adminlinkedin@gmail.com**
- **Password: AdminPassword123!**

Users: user{x}@gmail.com (x = {1,2,3,4})