| http://www.paso.gr/wp-content/uploads/2013/01/%CE%A0%CE%B1%CE%BD%CE%B5%CF%80%CE%B9%CF%83%CF%84%CE%AE%CE%BC%CE%B9%CE%BF-%CE%9A%CF%81%CE%AE%CF%84%CE%B7%CF%82.jpg | Πανεπιστήμιο Κρήτης  Τμήμα Επιστήμης Υπολογιστών |
| --- | --- |

**TASK MANAGER**

Organize your tasks Better

**<>**

Επιβλέπων/Επόπτης: Πιτικάκης Μάριος

Μέλος της Επιτροπής Παρακολούθησης/Αξιολόγησης:Τζίτζικας Ιωάννης

Χρονικό διάστημα: 10 /11/2023 - 20/09/2024

1. **Contents**

[2](#_heading=h.gjdgxs) Εισαγωγή 3

[2.1](#_heading=h.30j0zll) Γενική περιγραφή 4

[2.2](#_heading=h.1fob9te) Σκοπός συστήματος 5

[3](#_heading=h.3znysh7) Σενάρια Χρήσης και Τύποι Χρηστών 5

[4](#_heading=h.2et92p0) Απαιτήσεις 5

[5](#_heading=h.tyjcwt) Οθόνες της Εφαρμογής 6

[5.1](#_heading=h.3dy6vkm) Διαχειριστής 6

[5.1.1 Αρχική οθόνη – Σύνδεση χρήστη – Δημιουργία νέου χρήστη 6](#_heading=h.1t3h5sf)

[5.1.2 Λίστα εργασιών – tasks 7](#_heading=h.4d34og8)

[5.1.3 Δημιουργία εργασίας 8](#_heading=h.2s8eyo1)

[5.1.4 Δημιουργία κατηγορίας 15](#_heading=h.17dp8vu)

[5.1.5 Δημιουργία project 17](#_heading=h.3rdcrjn)

[5.1.6 Δημιουργία χρήστη 21](#_heading=h.26in1rg)

[5.1.7 Αρίθμηση λίστας 23](#_heading=h.lnxbz9)

[5.2](#_heading=h.35nkun2) Απλός χρήστης 24

[5.2.1 Αποδοχή ανάθεσης εργασίας 24](#_heading=h.1ksv4uv)

[5.2.2 Χρήση σχολίων 26](#_heading=h.44sinio)

[6](#_heading=h.2jxsxqh) Σχεδίαση και Μοντελοποίηση 26

[6.1](#_heading=h.z337ya) Σχεδιασμός Βάσης Δεδομένων 26

[6.1.1 Μοντέλα Οντοτήτων 27](#_heading=h.3j2qqm3)

[6.2](#_heading=h.1y810tw) Μοντέλα Οντοτήτων – Σχέσεις 31

[6.3](#_heading=h.4i7ojhp) Schemas 31

[7](#_heading=h.2xcytpi) Αρχιτεκτονική του Συστήματος και Εργαλεία 37

[7.1](#_heading=h.1ci93xb) Environment setup and deployment 37

[7.2](#_heading=h.3whwml4) Τεχνολογίες/frameworks used 37

[8](#_heading=h.2bn6wsx) Δοκιμή Συστήματος 39

[8.1](#_heading=h.qsh70q) Functionality Testing 39

[8.2](#_heading=h.3as4poj) User Testing 40

[8.3](#_heading=h.3o7alnk) Performance Testing 41

[9](#_heading=h.23ckvvd) Οδηγίες Χρήσης του Συστήματος 41

[10](#_heading=h.1pxezwc) Επίλογος 41

[10.1](#_heading=h.49x2ik5) Συμπεράσματα 41

[10.2](#_heading=h.2p2csry) Μελλοντικές επεκτάσεις 42

[Πηγές 43](#_heading=h.147n2zr)

# Εισαγωγή

Σε έναν ταχύτατα αναπτυσσόμενο κόσμο, αδιάσπαστα συνδεδεμένο με την τεχνολογία, η αποτελεσματική και επαρκής οργάνωση και διαχείριση χρόνου αποτελεί ζωτικής σημασίας. Η σημασία αυτή έγκειται τόσο στην ατομική παραγωγικότητα όσο στην επιτυχία συνεργασίας ομάδων, από δύο άτομα όπως σε μια οικογένεια, έως και εκατοντάδες διαφορετικα άτομα με εντελώς ετερογενή προγράμματα, όπως σε μια εταιρεία.

Αναπόσπαστο κομμάτι της ζωής όλων σχεδόν των ανθρώπων αποτελεί το διαδίκτυο, καθώς προσφέρει τα μέσα για την ολοκλήρωση αμέτρητων εργασιών αλλά και παροχή πληροφοριών σε οποιοδήποτε πρόβλημα χρήζει περαιτέρω διερεύνησης. Η ταχύτητα και ο όγκος πληροφοριών που παρέχεται μειώνει το χρόνο αλληλεπίδρασης και αναζήτησης σε οποιαδήποτε ανάγκη παρουσιάζεται. Αυτό έχει ως αποτέλεσμα στον εργασιακό χώρο οι ευθύνες των εργαζομένων να γίνονται όλο και πιο σύνθετες, λόγω του ατελείωτου όγκου πληροφοριών που έχει στη διάθεσή του ώστε να τις ολοκληρώσει και να τις βελτιστοποιήσει.

Στο σημερινό απαιτητικό περιβάλλον εργασίας, οι επαγγελματίες συχνά κατακλύζονται από τον τεράστιο όγκο πληροφοριών και εργασιών που πρέπει να διαχειρίζονται καθημερινά. Η πολυπλοκότητα της οργάνωσης, της ιεράρχησης και της παρακολούθησης αυτών των ευθυνών μπορεί να εμποδίσει την παραγωγικότητα και να οδηγήσει σε αναποτελεσματικότητα. Αυτή η αυξανόμενη πρόκληση υπογραμμίζει την ανάγκη για ένα δομημένο σύστημα διαχείρισης εργασιών, το οποίο επιδιώκει να αντιμετωπίσει αυτή η διατριβή αναπτύσσοντας ένα εργαλείο που εξορθολογίζει αυτές τις διαδικασίες και ενισχύει την καθημερινή επαγγελματική ζωή. Μια τέτοια εφαρμογή ήταν ο στόχος της ακόλουθης διπλωματικής εργασίας, στην οποία παρουσιάζεται η δομή και οι δυνατότητές της στο Τμήμα Επιστήμης Υπολογιστών του Πανεπιστημίου Κρήτης κατά το διάστημα 10/11/2023 έως 20/09/2024.

## Γενική περιγραφή

Το σύστημα θα απεικονίζεται σε μια διαδικτυακή εφαρμογή. Σε αυτήν ανάλογα με τον τύπο χρήστη που θα κάνει είσοδο στο σύστημα θα του παρουσιάζεται η αντίστοιχη διεπαφή, με τις ανάλογες λειτουργίες και δυνατότητες.

Το σύστημα θα έχει ως χρήστες:

* Τον administrator ή διαχειριστή, που χειρίζεται την εφαρμογή και έχει απεριόριστες δυνατότητες ως προς τη διαχείρισή της
* Τον απλό χρήστη, του οποίου οι δυνατότητες περιορίζονται στις βασικές, αρκετές ώστε να μπορέσει να εκτελέσει τις υποχρεώσεις/καθήκοντά του

Η παρούσα πτυχιακή εργασία έχει ως τίτλο «Εφαρμογή Διαχείρισης Εργασιών» η οποία υλοποιήθηκε κα τα το χρονικό διάστημα 10/11/2023 – 20/09/2024.

Το Σύστημα Εφαρμογής Διαχείρισης Εργασιών αποτελεί μια διαδικτυακή εφαρμογή που μπορεί να χρησιμοποιηθεί από τους χρήστες:

* Administrator (admin): έχει πρόσβαση σε όλες τις σελίδες της εφαρμογής και μπορεί να εκτελέσει όλες τις εντολές της εφαρμογής.
* Simple user: έχει πρόσβαση σε περιορισμένο αριθμό σελίδων της εφαρμογής και δεν μπορεί να εκτελέσει όλες τις εντολές της.

Ανάλογα με τον τύπο του χρήστη που συνδέεται εμφανίζονται οι αντίστοιχες διεπαφές και στη συνέχεια ο χρήστης έχει πρόσβαση στις διαφορετικές λειτουργίες.

Τα χαρακτηριστικά της εφαρμογής είναι:

* δημιουργία εργασίας (task)
* διαγραφή εργασίας
* ανάθεση εργασίας
* αλλαγή και παρακολούθηση κατάστασης εργασίας (open, in progress, completed, rejected, accepted)
* ανάθεση βαρύτητας/προτεραιότητας σε ένα task (low, medium, high)
* ορισμός προθεσμίας ολοκλήρωσης (due date)
* παράθυρο διαλόγου (task history/chat box)
* δημιουργία σχολίων
* δημιουργία project που περιέχει συλλογή εργασιών
* επεξεργασία και διαγραφή project
* δημιουργία κατηγορίας εργασιών
* επεξεργασία και διαγραφή κατηγορίας εργασιών
* δημιουργία χρηστών στους οποίους ανατίθενται εργασίες
* επεξεργασία και διαγραφή χρηστών

Στη συνέχεια παρουσιάζονται τα στάδια ανάπτυξης της εφαρμογής και οι διαφορετικές λειτουργίες και δυνατότητές της.

## Σκοπός συστήματος

Το λογισμικό του συστήματος που αναπτύχθηκε εξυπηρετεί τα άτομα μιας ομάδας προς το διαμοιρασμό φόρτου εργασίας και ανάθεσης καθηκόντων σχετικά με αυτή, μιας και όλοι οι χρήστες επεμβαίνουν στο σύνολο των εργασιών. Μέσω του συστήματος εκτελούνται διαφορετικές διεργασίες ανάλογα με το χρήστη, όπως δημιουργία task, ανάθεσή του σε κάποιον εργαζόμενο, σχολιασμός και επικοινωνία ανάμεσα στους χρήστες, δημιουργία προθεσμίας ολοκλήρωσης, παρακολούθηση, εξέλιξης του task και ολοκλήρωσής του.

Κεντρικό στοιχείο ανάπτυξης της εφαρμογής είναι η ανθρώπινη συνεργασία με στόχο την ολοκλήρωση καθηκόντων, χωρίς την οποία το έργο αυτό θα ήταν χρονοβόρο και δυσνόητο. Αντίθετα, μέσω της επικοινωνίας των ατόμων της ομάδας, ο κοινός τους στόχος φτάνει σε πέρας γρηγορότερα και πιο ολοκληρωμένα, διαθέτοντας γνώση συνδυασμένη από όλους.

# Σενάρια Χρήσης και Τύποι Χρηστών

Οι διαφορετικοί ρόλοι της εφαρμογής είναι δύο:

* Administrator - Διαχειριστής
* Simple user - Απλός χρήστης

Ο διαχειριστής έχει πρόσβαση σε όλες τις διεπαφές της εφαρμογής. Μπορεί να δημιουργεί, να επεξεργάζεται και να διαγράφει χρήστες, να δημιουργεί εργασίες, να τις επεξεργάζεται, να τις αναθέτει σε άλλους χρήστες, να αλλάζει την κατάσταση προόδου τους και τη σημαντικότητά τους καθώς και να τις διαγράφει. Έχει, επίσης, τη δυνατότητα να δημιουργεί projects στα οποία ανήκουν οι εργασίες, να τα επεξεργάζεται και να τα διαγράφει αλλά και να αναθέτει τις εργασίες σε κατηγορίες της δικής του αρεσκείας.

Ο απλός χρήστης έχει πρόσβαση μόνο στη διεπαφή η οποία αφορά τις εργασίες και μπορεί να δει και να αλλάξει την κατάσταση προόδου των εργασιών μόνο εφόσον έχουν ανατεθεί στον ίδιο. Η πρόσβασή του σε τόσο περιορισμένο ποσοστό της εφαρμογής εξυπηρετεί τη λειτουργία του συστήματος διαχείρισης εργασιών έτσι ώστε να μη δημιουργείται σύγχυση κατά τη δημιουργία και κατανομή του φόρτου και των εργασιών.

Σε αυτή την εργασία οι διαφορετικοί τύποι χρηστών ταυτίζονται και με τους ρόλους του συστήματος.

# Απαιτήσεις

Οι προδιαγραφές του υπολογιστή στον οποίο δημιουργήθηκε και εκτελέστηκε η εφαρμογή:

| Component | Model |
| --- | --- |
| Επεξεργαστής-processor | AMD Ryzen 5 3600 6-Core Processor 3.59 GHz |
| Μνήμη-memory | 16.0 GB |
| Κάρτα γραφικών | NVIDIA GeForce GTX 1660 |

Επιπλέον χρησιμοποιήθηκαν:

* VS Code version **1.93.1** (IDE)
* Express.js version 4.17.21 (server)
* React version 18.3.3 (Javascript library)
* Mongo DB 5.11.97 & (βάση δεδομένων)
* Google Chrome Version 95.0.4638.54 (browser)

Τα εργαλεία αυτά και η εγκατάστασή τους (Express.js, React, Mongo DB) είναι προαπαιτούμενα για την ανάπτυξη και λειτουργία της εφαρμογής, ενώ το IDE & o browser μπορούν να αλλαχθούν ανάλογα με τις ανάγκες του προγραμματιστή.

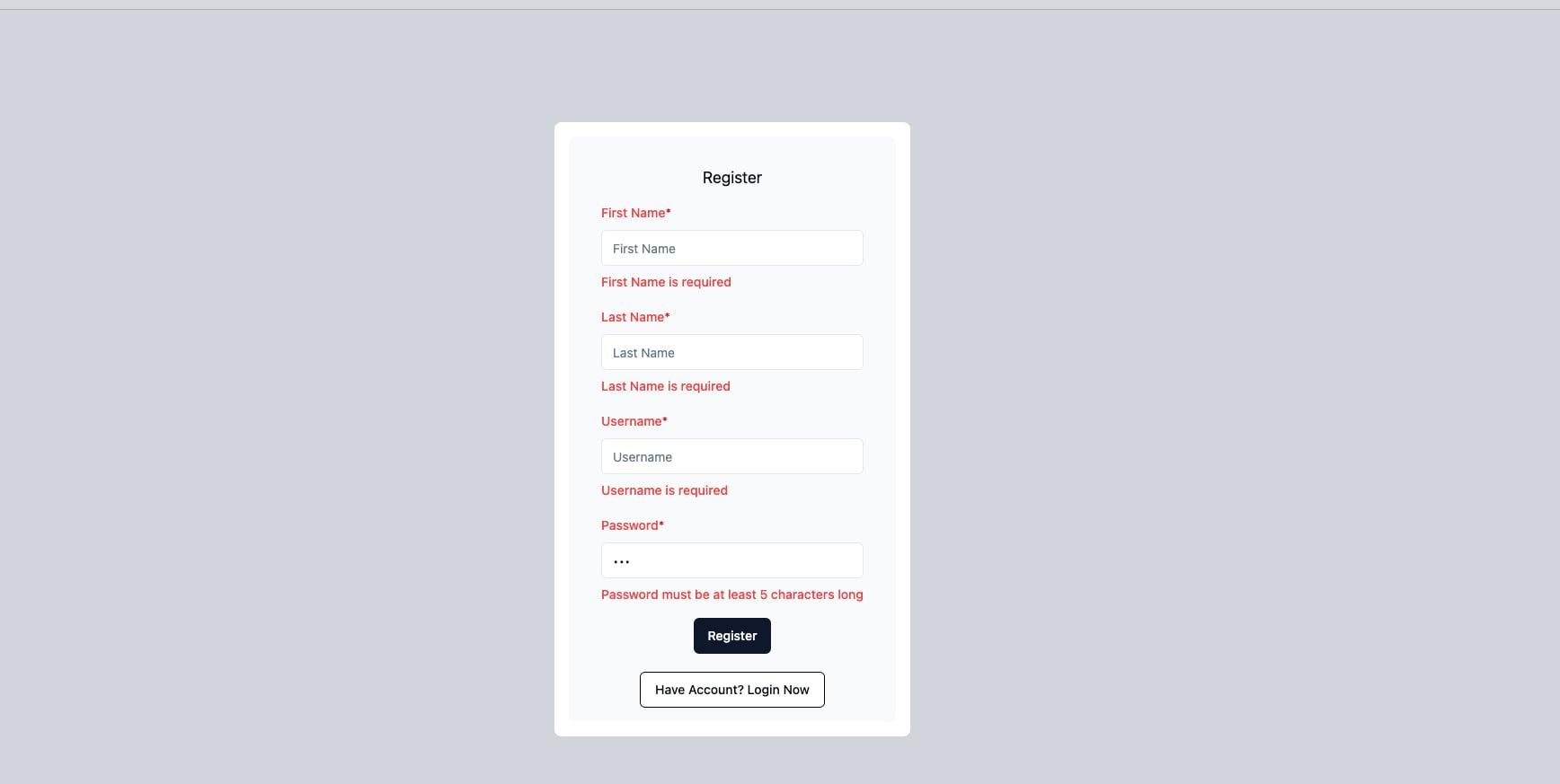
# Οθόνες της Εφαρμογής

Σε πρώτο στάδιο παρουσιάζονται οι οθόνες στις οποίες έχει πρόσβαση ο διαχειριστής της σελίδας και οι νέοι χρήστες με τα ίδια δικαιώματα/ρόλο.

## Διαχειριστής

### Αρχική οθόνη – Σύνδεση χρήστη – Δημιουργία νέου χρήστη

Η αρχική οθόνη σύνδεσης χρηστών περιέχει τα πεδία που είναι απαραίτητα για τη δημιουργία τους. Τα πεδία αυτά είναι το όνομα και το επώνυμό του, το ψευδώνυμό (username) του και ένας κωδικός εισόδου. Αυτά τα τέσσερα πεδία είναι υποχρεωτικά, κάτι που γίνεται εμφανές τόσο από τον αστερίσκο μετά από κάθε πεδία όσο και από τα μηνύματα λάθους που εμφανίζονται στην περίπτωση που ο χρήστης προσπαθήσει να συνδεθεί προσπερνώντας τη δημιουργία τους. Ένας επιπλέον μηχανισμός ελέγχου αποτελεί η προϋπόθεση ύπαρξης κωδικού σύνδεσης με ελάχιστο μήκος τους 6 χαρακτήρες. Η αρχική οθόνη περιέχει, τέλος, ένα κουμπί που αντιστοιχεί στην αποθήκευση ενός νέου χρήστη και ένα κουμπί που οδηγεί στη σύνδεση υπάρχοντος χρήστη με εισαγωγή του ψευδωνύμου του και του κωδικού πρόσβασης.



A screenshot of a login form

Description automatically generated

### Λίστα εργασιών – tasks

Αρχικά η λίστα εργασιών είναι κενή, καθώς δεν έχουν δημιουργηθεί ακόμα νέες εργασίες. Υπάρχει δυνατότητα δημιουργίας νέας εργασίας (“Create Task” κουμπί - 1), αναζήτηση συγκεκριμένης εργασίας με βάση κείμενο (φίλτρο “Search” - 2) και πρόσβαση σε διαφορετικές λίστες (εργασίες, κατηγορίες, projects και χρήστες – εικονίδια 3). Τέλος υπάρχει δυνατότητα πρόσβασης στον προσωπικό λογαριασμό του τρέχοντος χρήστη και η αποσύνδεση από την εφαρμογή (εικονίδιο 4).

A screenshot of a computer

Description automatically generated

A screenshot of a computer

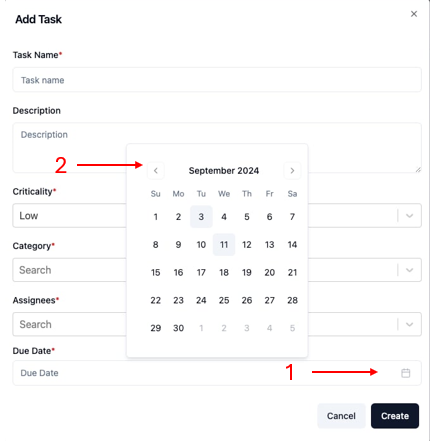
Description automatically generated

### Δημιουργία εργασίας

Τα υποχρεωτικά πεδία κατά τη δημιουργία νέας εργασίας είναι το όνομα – αναγνωριστικό (υποχρεωτικό πεδίο), μια προαιρετική περιγραφή σχετικά με την εργασία, η σημαντικότητα της εργασίας (υποχρεωτικό πεδίο), η κατηγορία της εργασίας (υποχρεωτικό πεδίο), ο χρήστης στον οποίο ανατίθεται η εργασία (υποχρεωτικό πεδίο) και η προθεσμία ολοκλήρωσής της (υποχρεωτικό πεδίο). Τέλος, υπάρχουν δύο κουμπιά, η δημιουργία και αποθήκευση της εργασίας και η ακύρωσή της.

A screenshot of a computer

Description automatically generated



1. Δημιουργία εργασίας – μηνύματα λάθους

A screenshot of a computer

Description automatically generated

1. Δημιουργία εργασίας – ανάθεση τιμών

A screenshot of a computer

Description automatically generated

Μετά τη δημιουργία δύο διαφορετικών εργασιών η λίστα εμφανίζεται ως εξής:

A screenshot of a computer

Description automatically generated

Σημειώνεται πως μετά τη δημιουργία κάθε εργασίας και την καταχώρησή της στη λίστα (μαύρο βέλος) ή μετά την επεξεργασία του περιεχομένου της εμφανίζεται ένα μήνυμα επιβεβαίωσης (κόκκινο βέλος).

Στα επόμενα στιγμιότυπα εμφανίζονται διαφορετικές καταστάσεις κατά τη λειτουργία της εφαρμογής με τα απαραίτητα σχόλια/επεξηγήσεις.

Σε αυτή την περίπτωση τα δύο βέλη υποδεικνύουν την εμφάνιση προειδοποίησης για μια εργασία η οποία έχει ξεπεράσει την προθεσμία ολοκλήρωσης της χωρίς να έχει ολοκληρωθεί αλλά και τη δυνατότητα αποδοχής μια εργασίας από το χρήστη (γεγονός που θα αλλάξει την κατάσταση της εργασίας από open σε in progress.

A screenshot of a computer

Description automatically generated

Η επόμενη κατάσταση στην οποία μπορεί να μεταβεί η εργασία είναι από in progress σε completed.

A screenshot of a computer

Description automatically generated

Αφού η εργασία έχει ολοκληρωθεί, η επόμενη κατάσταση στην οποία μεταβαίνει είναι η completed ή rejected, ανάλογα με την τρέχουσα κατάσταση και απόφαση του χρήστη στον οποίο έχει ανατεθεί.

A screenshot of a computer

Description automatically generated

Οι καταστάσεις completed και rejected ταυτοποιούνται ως εξής στο ακόλουθο στιγμιότυπο. Κατά τον κύκλο ζωής μιας εργασίας είναι επίσης πιθανό ο χρήστης να επιλέξει την επεξεργασία της ή και τη διαγραφή της όπως υπογραμμίζεται στο στιγμιότυπο.

A screenshot of a computer

Description automatically generated

Αν ο χρήστης επιλέξει να επεξεργαστεί την εργασία (επιλογή “Edit”) εμφανίζεται το ιστορικό της εργασίας, που περιέχει όλα τα στιγμιότυπα κατά τον κύκλο ζωής της. Μπορεί να περιέχει πληροφορίες για τις αλλαγές κατάστασης της εργασίας, την ανάθεση σε κάποιο χρήστη, την αλλαγή προτεραιότητας και την επεξεργασία οποιουδήποτε από τα πεδία της. Τέλος, δίνεται η δυνατότητα στο χρήστη στον οποίο έχει ανατεθεί η εργασία (εκτός του διαχειριστή) να αφήσει κάποιο σχόλιο, είτε προς τον ίδιο ώστε να λειτουργήσει ως σημείωση, είτε σε κάποιον από τους άλλους χρήστες υπό τη μορφή ενός απλού chat box. Παραδείγματα αυτών των πληροφοριών ακολουθούν στα παρακάτω στιγμιότυπα:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Εάν ο χρήστης επιλέξει να διαγράψει τη συγκεκριμένη εργασίας, εμφανίζεται ένα μήνυμα επιβεβαίωσης το οποίο μπορεί να επιβεβαιώσει ή να ακυρώσει, ακυρώνοντας κατά συνέπεια και τη διαγραφή.

A screenshot of a computer

Description automatically generated

### Δημιουργία κατηγορίας

Αρχικά, η λίστα περιέχει μόνο μια κλειδωμένη κατηγορία. Σε λάθη περίπτωση, η κατηγορία “Default Category” έχει δημιουργηθεί έτσι ώστε να μην μπορεί να διαγραφεί (εικονίδιο σε κόκκινο πλαίσιο). Με αυτό τον τρόπο δρα ως προεπιλογή για όλες τις εργασίες ακόμα και αν δεν έχει δημιουργηθεί νέα κατηγορία.

A screenshot of a computer

Description automatically generated

Μια κατηγορία περιέχει μόνο ένα πεδίο (το όνομα), το οποίο είναι και υποχρεωτικό.

A screenshot of a computer

Description automatically generated

Σε περίπτωση που ο χρήστης επιχειρήσει να αποθηκεύσει μια νέα κατηγορία χωρίς να καταχωρήσει το όνομά της, εμφανίζεται το αντίστοιχο μήνυμα λάθους. Ως επιπλέον έλεγχος ασφαλείας, γίνεται βέβαιο πως το όνομα μιας νέας κατηγορίας δεν μπορεί να είναι λιδιο με αυτό της προϋπάρχουσας.

A screenshot of a computer

Description automatically generated

### Δημιουργία project

Αρχικά η λίστα παραμένει κενή πριν τη δημιουργία νέου project. Κατά τη δημιουργία ενός νέου, τα υποχρεωτικά πεδία του είναι μόνο ο τίτλος του. Προαιρετικά, μπορεί να περιέχει μια λίστα από νέες εργασίες (η λίστα μπορεί να είναι κενή, να περιέχει μία ή και περισσότερες εργασίες (κουμπί “Add Task”). Το μέγεθος της λίστας αφήνεται στην κρίση του χρήστη). Στη λίστα αυτή ο χρήστης μπορεί να προσθέσει αλλά και να αφαιρέσει εργασίες (κουμπί “Remove”). Αν γίνει επιλογή προσθήκης τουλάχιστον μιας εργασίας αυτή δεν μπορεί να παραμείνει κενή (εμφανίζεται μήνυμα λάθους).

A screenshot of a computer

Description automatically generated

A screenshot of a project

Description automatically generated

Αφού προστεθούν οι επιθυμητές εργασίες και ανατεθούν σε κάποιο χρήστη, έστω ότι οι καταστάσεις τους έχουν αλλάξει και το project περιέχει 5 εργασίες: 1 open (γκρι), 1 in progress (γαλάζιο), 1 accepted (σκούρο πράσινο), 1 completed(ανοιχτό πράσινο) & 1 rejected (κόκκινο). Σημειώνεται, επιπλέον, πως αμέσως μετά τη δημιουργία ενός νέου project εμφανίζεται ένα μήνυμα που το επιβεβαιώνει.

A screenshot of a project list

Description automatically generated

Σε επόμενο βήμα, έστω δημιουργείται μια ακόμα εργασία με κατάσταση in progress και μια εργασία με κατάσταση open. Η μπάρα προόδου διαμορφώνεται ακολούθως και το αντίστοιχο μήνυμα εμφανίζεται στην οθόνη:

A screenshot of a project

Description automatically generated

Εάν ο χρήστης επιλέξει το όνομα του project εμφανίζεται η λίστα με τις εργασίες, τα ονόματά τους και την κατάσταση στην οποία βρίσκονται.

1. Επιλογή ονόματος

A screenshot of a project list

Description automatically generated

1. Εμφάνιση λίστας

A screenshot of a project

Description automatically generated

Εάν ο χρήστης επιλέξει να διαγράψει το project (ο τρόπος διαγραφής είναι ίδιος με αυτό των εργασιών) εμφανίζεται ένα μήνυμα επιβεβαίωσης με δύο επιλογες: επιβεβαίωση και ακύρωση.

A screenshot of a computer

Description automatically generated

### Δημιουργία χρήστη

Αρχικά, όπως και στις προηγούμενες περιπτώσεις, η λίστα εμφανίζεται κενή εάν δεν έχει δημιουργηθεί κάποιος νέος χρήστης. Μετά τη δημιουργία ενός χρήστη η λίστα εμφανίζεται ως εξής:

A screenshot of a computer

Description automatically generated

Στην περίπτωση που ο χρήστης επιλέξει το μενού (3 τελείες) εμφανίζεται η επιλογή να μετατρέψει τον τρέχον (απλό) χρήστη σε διαχειριστή, η οποία συνοδεύεται από ένα μήνυμα επιβεβαίωσης το οποίο ο χρήστης μπορεί να επιβεβαιώσει ή να ακυρώσει. Μια τέτοια ενέργεια μπορεί να πραγματοποιηθεί μόνο από ένα διαχειριστή. Μια επιπλέον επιλογή που δίνεται στο διαχειριστή είναι να διαγράψει το χρήστη και τότε εμφανίζεται ένα μήνυμα επιβεβαίωσης, ανάλογο με τις προηγούμενες περιπτώσεις διαγραφής.

A screenshot of a delete

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Η ίδια διαδικασία επαναλαμβάνεται στην περίπτωση που ένας διαχειριστής μετατραπεί σε απλό χρήστη από κάποιον άλλο διαχειριστή.

A screenshot of a computer

Description automatically generated

### Αρίθμηση λίστας

Ένα επιπλέον στοιχείο που υλοποιήθηκε είναι η δυνατότητα προβολής διαφορετικού πλήθους στοιχείων της λίστας.

A screenshot of a computer

Description automatically generated

Την ίδια στιγμή μπορούν να προβάλλονται στη λίστα στην ίδια σελίδα 15, 25, 50 ή 100 στοιχεία/σειρές.

A screenshot of a computer

Description automatically generated

## Απλός χρήστης

### Αποδοχή ανάθεσης εργασίας

Όπως έχει προαναφερθεί, ένας απλός χρήστης έχει πρόσβαση μόνο στη λίστα με εργασίες οι οποίες του έχουν ανατεθεί. Αυτό σημαίνει πως όταν υπάρχουν επιπλέον εργασίες οι οποίες έχουν ανατεθεί σε διαφορετικούς χρήστες, αυτές δεν είναι ορατές από το χρήστη που είναι συνδεδεμένος στην εφαρμογή την τρέχουσα στιγμή.

Επιλέγοντας τη λίστα επιλογών μιας εργασίας (τρεις τελείες), ο χρήστης μπορεί να επιλέξει να αποδεχτεί τη συγκεκριμένη εργασία (αν δεν το έχει κάνει ήδη). Επαναλαμβάνεται πως για να πραγματοποιηθεί αυτή η ενέργεια ο συνδεδεμένος χρήστης πρέπει να είναι ένας από αυτούς στους οποίους η εργασία έχει ανατεθεί μέσω του διαχειριστή.

A screenshot of a computer

Description automatically generated

Όταν μια εργασία έχει ανατεθεί και επομένως έχει αρχίσει να αλλάζει καταστάσεις (όλες οι καταστάσεις μετά την open) το όνομα του συνδεδεμένου χρήστη εμφανίζεται ως πράσινο στη λίστα εργασιών οι οποίες του έχουν ανατεθεί.

A screenshot of a computer

Description automatically generated

### Χρήση σχολίων

Μια ακόμα ενέργεια που αντιστοιχεί στον απλό χρήστη είναι η χρήση σχολίων στο ιστορικό μιας εργασίας με τον ίδιο τρόπο όπως ένας διαχειριστής.

A screenshot of a computer

Description automatically generated

**Υποσημειώσεις:**

* Ο μόνος χρήστης που μπορεί να επεξεργαστεί οποιαδήποτε από τις οντότητες της βάσης δεδομένων είναι ο διαχειριστής και όποιος άλλος χρήστης αποκτά τα ίδια δικαιώματα.
* Σε περίπτωση που ένας χρήστης έχει αποδεχτεί την ανάθεση μιας εργασίας από το διαχειριστή, αυτός μπορεί να τον αφαιρέσει από τη λίστα χρηστών στους οποίους ανατίθεται η εργασία και για αυτό το λόγο δε θα είναι πλέον ορατός στη λίστα με τους assignees (χρήστες στους οποίους μπορεί να ανατεθεί μια εργασία)
* Το ιστορικό μιας εργασίας καταγράφει όλες τις αλλαγές σχετικά με την κατάσταση της ίδιας της εργασίας, των χρηστών που την αναλαμβάνουν, τις κατηγορίες στις οποίες ανήκει καθώς και τα project στα οποία ανήκει (ενέργειες: δημιουργία, επεξεργασία, διαγραφή)
* Εκτός του δεσμευμένου ονόματος της προεπιλεγμένης κατηγορίας (Default Category) οι κατηγορίες μπορούν να έχουν διπλότυπα ονόματα
* Κάθε ενέργεια διαγραφής συνοδεύεται από ένα μήνυμα επιβεβαίωσης και ενημέρωσης του χρήστη
* Η προσθήκη και η επεξεργασία έχουν την ίδια διεπιφάνεια (προσθήκη και επεξεργασία project)
* Ένας απλός χρήστης δεν μπορεί να εγκρίνει (approve) εργασίες (μόνο να ολοκληρώσει, complete) μιας και δεν έχει τα απαραίτητα δικαιώματα
* Η εφαρμογή διαθέτει εξ αρχής ένα χρήστη διαχειριστή, ώστε να μπορέσει να δημιουργήσει οποιεσδήποτε οντότητες χρειάζεται χωρίς κωλύματα

# Σχεδίαση και Μοντελοποίηση

Παρακάτω αναφέρεται η διαμόρφωση της βάσης δεδομένων και τα μοντέλα της.

Να σημειωθεί ότι κάθε μοντέλο περιέχει ένα μοναδικό αναγνωριστικό (\_id) το οποίο προστίθεται

αυτόματα από το Mongo DB.

## Σχεδιασμός Βάσης Δεδομένων

### Μοντέλα Οντοτήτων

* Project

1. *Title*: ο τίτλος
2. *Tasks*: η λίστα με tasks/εργασίες που περιέχονται στο project
3. *CreatedAt*: ημερομηνία δημιουργίας project
4. *UpdatedAt*: ημερομηνία επεξεργασίας project

A black background with white rectangles and text

Description automatically generated

* Category

1. *Name*: όνομα κατηγορίας
2. *CreatedAt*: ημερομηνία δημιουργίας
3. *UpdatedAt*: ημερομηνία επεξεργασίας

A black background with white text

Description automatically generated

* Task History

1. *Title*: τίτλος
2. *Action*: παίρνει 7 διαφορετικές τιμές: create, update, accept, complete, approve, reject, comment και είναι οι ενέργειες οι οποίες μπορεί να αναλάβει ο χρήστης ως προς τα tasks.
3. *User*: ο χρήστης που ανέλαβε το task
4. *Property*: αντιπροσωπεύει το task και τις τιμές των πεδίων του (criticality, users, …), οι πληροφορίες αποθηκεύονται σαν σύνολο σε ένα string
5. *OldValue*: προηγούμενη τιμή κατάστασης εργασίας
6. *NewValue*: νέα τιμή κατάστασης εργασίας
7. *CreatedAt*: ημερομηνία δημιουργίας
8. *UpdatedAt*: ημερομηνία επεξεργασίας

A black background with white rectangles and white text

Description automatically generated

* Task
  1. *Title*: τίτλος
  2. *Description*: περιγραφή task
  3. *Category* id: αναγνωριστικό κατηγορίας
  4. *Date*: ημερομηνία
  5. *Status*: κατάσταση task, η οποία μπορεί να λάβει μια από τις 5 τιμές: open, in progress, completed, approved, rejected
  6. *Criticality*: σημαντικότητα (παίρνει 3 διαφορετικές τιμές: low, medium, high)
  7. *AcceptedBy*: ο χρήστης στον οποίο ανατέθηκε (αποδέχτηκε)
  8. *Users*: οι χρήστες στους οποίους έχει ανατεθεί η εργασία
  9. *CreatedAt*: ημερομηνία δημιουργίας
  10. *UpdatedAt*: ημερομηνία επεξεργασίας

A black and white screen with white text

Description automatically generated

* User
  1. *Username*: μοναδικό όνομα χρήστη
  2. *FirstName*: όνομα
  3. *LastName*: επίθετο
  4. *Password*: κωδικός χρήστη
  5. *Role*: ρόλος (διαχειριστής, απλός χρήστης)
  6. *CreatedAt*: ημερομηνία δημιουργίας
  7. *UpdatedAt*: ημερομηνία επεξεργασίας

A screenshot of a computer screen

Description automatically generated

Μια ακόμα σημαντική οντότητα της βάσης είναι η *user DTO*. Η συγκεκριμένη οντότητα συμβάλλει στην καταγραφή του τρέχοντος συνδεδεμένου χρήστη. Ο logged user ελέγχεται από το middleware για επαλήθευση (authentication) και στη συνέχεια αντιγράφεται στο user DTO που αρχικοποιείται με τις κατάλληλες τιμές (αυτή η διαδικασία αποτελεί common practice στην ανάπτυξη εφαρμογών). Το σύστημα επιτρέπει να υπάρχουν συνδεδεμένοι ταυτόχρονα μέχρι και δύο χρήστες.

## Μοντέλα Οντοτήτων – Σχέσεις

* Ένα project έχει μια λίστα από tasks

A white diamond with black text

Description automatically generated

* Ένα task έχει ένα μοναδικό αναγνωριστικό (category id) category



* Ένα task ανατίθεται σε έναν τουλάχιστον user

A white diamond with black text

Description automatically generated

* Ένα task history αντιστοιχεί σε ένα task και σε έναν user





## Schemas

* Project

| **Attribute** | **Type** | **Required** |
| --- | --- | --- |
| Title | String | True |
| Tasks | Task[] | True |
| CreatedAt | Date | - |
| UpdatedAt | Date | - |

Το πεδίο tasks είναι ξένο κλειδί στο Task, καθώς ένα Project διαθέτει μια λίστα από Tasks.

Υλοποίηση κώδικα:

A computer screen shot of a program code

Description automatically generated

* Category

| **Attribute** | **Type** | **Required** |
| --- | --- | --- |
| Name | String | True |
| CreatedAt | Date | - |
| UpdatedAt | Date | - |

Υλοποίηση κώδικα:

A screen shot of a computer program

Description automatically generated

* Task History

| **Attribute** | **Type** | **Required** |
| --- | --- | --- |
| Task | Task | - |
| Action | TaskHistoryAction (enum) | - |
| User | User | - |
| Property | String | - |
| OldValue | String | - |
| NewValue | String | - |
| CreatedAt | Date | - |
| UpdatedAt | Date | - |

Το πεδίο Task είναι ξένο κλειδί στην οντότητα Task και το πεδίο User στην οντότητα User.

Υλοποίηση κώδικα:

A screen shot of a computer program

Description automatically generated A black screen with colorful text

Description automatically generated

* Task

| **Attribute** | **Type** | **Required** |
| --- | --- | --- |
| Title | String | True |
| Description | String | - |
| Category\_id | Category | True |
| Date | Date | True |
| Status | TaskStatus | - |
| Criticality | TaskCriticality (enum string) | - |
| AcceptedBy | User | - |
| Users | User[] | True |
| CreatedBy | Date | - |
| UpdatedBy | Date | - |

To πεδίο category\_id είναι ξένο κλειδί στο Category, το πεδίο acceptedBy είναι ξένο κλειδί στην οντότητα User, αφού κάποιος User αποδέχεται ένα Task και η λίστα users είναι ξένο κλειδί στην οντότητα User.

Υλοποίηση κώδικα:

A screen shot of a computer program

Description automatically generated A computer screen with many colorful text

Description automatically generated

* User

| **Attribute** | **Type** | **Required** |
| --- | --- | --- |
| Username | String | True |
| FirstName | String | True |
| LastName | String | True |
| Password | String | True |
| Role | UserRole (enum) | - |
| CreatedAt | Date | - |
| UpdatedAt | Date | - |

Υλοποίηση κώδικα:

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

Υποσημείωση: το πεδίο approved στη μέχρι τώρα υλοποίηση του κώδικα είναι περιττό. Δημιουργήθηκε, όμως με σκοπό την επαλήθευση περαιτέρω χρηστών.

## Enums

Πολύ σημαντικά για την υλοποίηση του κώδικα και της βάσης δεδομένων. Καταγράφουν μια λίστα πιθανών τιμών που μπορεί να λάβει ένα πεδίο. Τα enums που χρησιμοποιήθηκαν σε αυτή την εφαρμογή είναι:

* *HttpStatusCode*: καταγράφει τα πιθανά σφάλματα που μπορεί να συμβούν κατά την εκτέλεση και τα κωδικοποιεί ανάλογα

A screen shot of a computer program

Description automatically generated

* *TaskCriticality*: καθορίζει την προτεραιότητα με την οποία πρέπει να εκτελεστούν οι διαφορετικές εργασίες, αλλιώς η σημαντικότητά τους

A screenshot of a computer program

Description automatically generated

* *TaskHistoryActions*: οι ενέργειες στις οποίες μπορεί να προβεί ο χρήστης για την εκτέλεση μιας εργασίας, από τη στιγμή της δημιουργίας της έως και την ολοκλήρωσή της

A screen shot of a computer program

Description automatically generated

* *TaskStatus*: τα στάδια τα οποία διανύει μια εργασία από τη στιγμή της δημιουργίας της έως και την ολοκλήρωσή της

A screen shot of a computer program

Description automatically generated

* *UserRole*: οι πιθανοί ρόλοι που μπορεί να ανατεθούν σε ένα χρήστη και επηρεάζουν τα δικαιώματα που έχει στην εφαρμογή

A black screen with text and symbols

Description automatically generated with medium confidence

## Ασφάλεια

Ένα JWK token (JSON Web Key) είναι μια δομή δεδομένων που βασίζεται σε JSON το οποίο αντιπροσωπεύει ένα κρυπτογραφικό κλειδί. Αποτελεί μέρος του πλαισίου JSON Web Token (JWT) και χρησιμοποιείται για ασφαλή έλεγχο ταυτότητας που βασίζεται σε διακριτικά και ανταλλαγή δεδομένων σε εφαρμογές web. Η μορφή JWK επιτρέπει την ασφαλή ανταλλαγή κλειδιών που χρησιμοποιούνται για την υπογραφή ή την κρυπτογράφηση διακριτικών.

Ένα JWK περιέχει συνήθως πεδία όπως kty (τύπος κλειδιού), χρήση (προβλεπόμενη χρήση του κλειδιού), alg (αλγόριθμος) και το πραγματικό υλικό κλειδιού (π.χ. n και e για δημόσια κλειδιά RSA).

Τα JWK χρησιμοποιούνται κυρίως για την αναπαράσταση κλειδιών με τρόπο που είναι εύκολο να σειριοποιηθεί και να μεταδοθεί μέσω του ιστού. Χρησιμοποιούνται σε καταστάσεις όπου τα κλειδιά πρέπει να μοιράζονται μεταξύ συστημάτων, όπως κατά τη διαδικασία επικύρωσης JWT.

Χρησιμοποιείται για την επαλήθευση του χρήστη (authmiddleware).

# Αρχιτεκτονική του Συστήματος και Εργαλεία

Frontend: React.js

* Διεπαφή χρήστη: Η React είναι υπεύθυνη για τη διεπαφή χρήστη. Ο διαχειριστής εργασιών θα έχει στοιχεία όπως λίστες εργασιών, λεπτομέρειες εργασιών, φόρμες για προσθήκη/επεξεργασία εργασιών και διεπαφές ελέγχου ταυτότητας χρήστη.
* Διαχείριση κατάστασης: Η διαχείριση κατάστασης του React (χρησιμοποιώντας αγκίστρια όπως useState και useReducer ή χρησιμοποιώντας βιβλιοθήκη όπως το Redux) θα χειριστεί την κατάσταση της εφαρμογής, όπως εργασίες παρακολούθησης και δεδομένα χρήστη.
* Κλήσεις API: Το React θα κάνει αιτήματα HTTP στο back-end χρησιμοποιώντας το axios ή το Fetch API για αλληλεπίδραση με το RESTful API που παρέχεται από το back-end. Αυτό περιλαμβάνει λειτουργίες CRUD (Δημιουργία, Ανάγνωση, Ενημέρωση, Διαγραφή) για εργασίες και έλεγχο ταυτότητας χρήστη.

Back-end: Node.js & Express.js

* Διακομιστής και API: Το Node.js με το Express.js θα χειριστεί τη λογική από την πλευρά του διακομιστή. Θα παρέχει ένα RESTful API που το front-end μπορεί να καλέσει για να διαχειριστεί εργασίες. Τα τελικά σημεία API μπορεί να περιλαμβάνουν διαδρομές όπως /tasks, /tasks/:id, /users και /auth.
* Έλεγχος ταυτότητας: Εφαρμογή ελέγχου ταυτότητας χρησιμοποιώντας JWT (JSON Web Tokens). Οι χρήστες θα συνδεθούν και θα λάβουν ένα διακριτικό, το οποίο θα απαιτείται για την πρόσβαση σε προστατευμένες διαδρομές (όπως η δημιουργία ή η επεξεργασία εργασιών).
* Business Logic: Το Express.js θα διαχειρίζεται την επιχειρηματική λογική, συμπεριλαμβανομένων των κανόνων επικύρωσης, εξουσιοδότησης και διαχείρισης εργασιών (π.χ. μια εργασία δεν μπορεί να διαγραφεί από κανέναν άλλον εκτός από τον δημιουργό της).

Βάση δεδομένων: MongoDB

* Αποθήκευση δεδομένων: Η MongoDB, μια βάση δεδομένων NoSQL, θα αποθηκεύσει όλα τα δεδομένα της εφαρμογής, συμπεριλαμβανομένων των στοιχείων χρήστη, των εργασιών και οποιασδήποτε άλλης σχετικής πληροφορίας.
* Σχήματα και μοντέλα: Ορίστε σχήματα χρησιμοποιώντας το Mongoose (μια βιβλιοθήκη ODM για MongoDB) για εργασίες και χρήστες. Το σχήμα εργασιών μπορεί να περιλαμβάνει πεδία όπως τίτλος, περιγραφή, κατάσταση, προτεραιότητα, ημερομηνία λήξης και κάτοχος.
* Ερωτήματα: Το MongoDB θα επιτρέψει την αποτελεσματική αναζήτηση εργασιών με βάση διάφορες παραμέτρους (π.χ. ημερομηνία λήξης, προτεραιότητα, χρήστης).

Ροή Επικοινωνίας

* Front-end to Back-end: Το React επικοινωνεί με τον διακομιστή Node.js μέσω αιτημάτων API. Για παράδειγμα, όταν ένας χρήστης προσθέτει μια εργασία, το React στέλνει ένα αίτημα POST στο τελικό σημείο /tasks του διακομιστή.
* Back-end στη βάση δεδομένων: Ο διακομιστής Node.js χρησιμοποιεί το Mongoose για αλληλεπίδραση με το MongoDB, εκτελώντας λειτουργίες CRUD με βάση τα αιτήματα που λαμβάνονται από το front-end.
* Χειρισμός απόκρισης: Το back-end ανταποκρίνεται στο front-end με το αποτέλεσμα της λειτουργίας (π.χ. τη νέα εργασία ή ένα μήνυμα σφάλματος). Το React στη συνέχεια ενημερώνει τη διεπαφή χρήστη με βάση αυτήν την απάντηση.

Ροή ελέγχου ταυτότητας

* Είσοδος χρήστη: Ένας χρήστης συνδέεται μέσω της διεπαφής του React. Τα διαπιστευτήρια αποστέλλονται στο back-end, όπου και επαληθεύονται. Εάν είναι έγκυρο, το back-end επιστρέφει ένα JWT.
* Αποθήκευση Token: Το JWT αποθηκεύεται στο localStorage ή σε ένα cookie στην πλευρά του πελάτη.
* Προστατευμένες διαδρομές: Όταν ο χρήστης προσπαθεί να αποκτήσει πρόσβαση σε μια προστατευμένη διαδρομή (π.χ. δημιουργία μιας εργασίας), το JWT αποστέλλεται μαζί με το αίτημα API. Το back-end επαληθεύει το διακριτικό πριν επεξεργαστεί το αίτημα.

Θέματα ασφαλείας

* Επικύρωση και εξυγίανση δεδομένων: Βεβαιωθείτε ότι όλα τα δεδομένα που εισέρχονται στο API έχουν επικυρωθεί και απολυμανθεί για την αποφυγή τρωτών σημείων ασφαλείας όπως η έγχυση SQL ή οι επιθέσεις XSS.
* HTTPS: Χρησιμοποιήστε HTTPS για κρυπτογράφηση δεδομένων κατά τη μεταφορά.
* Λήξη JWT: Ορίστε έναν εύλογο χρόνο λήξης για τα διακριτικά JWT για να μειώσετε τον κίνδυνο κακής χρήσης εάν ένα διακριτικό παραβιαστεί.

## Environment setup and deployment

Η πρώτη εντολή που εκτελείται είναι η **docker compose up** η οποία δημιουργεί τα 3 containers που χρειάζεται η εφαρμογή για να εκτελεστεί σωστά. Τα containers δημιουργούνται πάντα με την ίδια σειρά και έχουν άμεση εξάρτηση μεταξύ τους:

* Το πρώτο είναι το MongoDB
* Το δεύτερο είναι ο server
* Το τρίτο είναι το container της React, δηλαδή ο client

Η επόμενη εντολή που εκτελείται είναι η **yarn && yarn dev --host.**

Υποσημείωση: για λόγους οι οποίοι δεν είναι εντελώς ξεκάθαροι μερικές φορές αυτή η εντολή δημιουργεί προβλήματα. Η λύση στο πρόβλημα αυτό είναι η επανεκτέλεσή της ως **yarn dev --host** και στη συνέχεια η επανεκτέλεση της **docker compose up.** Η διερεύνηση και βελτίωση αυτού του προβλήματος μπορεί να προστεθεί ως βελτίωση για τη μελλοντική επανέκδοση της εφαρμογής. Συγκεκριμένα μετά την εκτέλεση της **docker compose up** και την **ολοκλήρωση** των dependencies και των packages από το frontend και την εμφάνιση του μηνύματος **frontend-1 exited with code 0**(διαδικασία που μπορεί να χρειαστεί λίγα λεπτά), είναι απαραίτητη η διακοπή της εργασίας και η αλλαγή στο αρχείο **docker-compose.yaml ,**στο πεδίο **command,** από **yarn && yarn dev --host** σε **yarn dev --host** και η επανεκτέλεση της **docker compose up**. Έπειτα μέσα από το **Docker** στο παράθυρο **Containers,** κάνοντας κλικ στο **3000:3000** ανοίγει το frontend. Ο default διαχειριστής έχει **username**: vasilis και **password**: vasilispass .

Το βασικό αρχείο της εφαρμογής το οποίο οδηγεί στην εκτέλεσή του ονομάζεται **docker-compose.yaml**. Τόσο το front-end όσο και το back-end έχουν τα δικά τους Dockerfile (καθορίζουν το περιβάλλον στο οποίο θα τρέξει η εφαρμογή).Αφού έχει ολοκληρωθεί το compilation, η εκκίνηση του frontend γίνεται από το **Docker**, κάνοντας κλικ στο Ακολουθούν στιγμιότυπα του **docker-compose.yaml**  και των δύο Dockerfiles.

A screen shot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Τέλος αξίζει να δοθεί ένα στιγμιότυπο των καταλόγων της εφαρμογής (logs) τα οποία καταγράφουν όλες τις χρήσιμες πληροφορίες της εφαρμογής, καθώς και τη χρονική σειρά με την οποία συμβαίνουν διαφορετικά γεγονότα.

Τα logs των containers αφού δημιουργηθούν θα πρέπει να είναι παρόμοια με το ακόλουθο στιγμιότυπο:

A screenshot of a computer

Description automatically generated

## Τεχνολογίες/frameworks used

Η υλοποίηση της εφαρμογής βασίστηκε στο MERM stack, το οποίο χρησιμοποιεί:

* MongoDB: Μία NoSQL database γνωστή για την προσαρμοστικότητά της. Χρησιμοποιείται για αποθήκευση δεδομένων σε JSON-like αρχεία.
* Express.js: ‘Ενα web application framework για Node.js που χρησιμοποιείται για τη δημιουργία web εφαρμογών και APIs (back-end).
* React: μια δημοφιλής JavaScript βιβλιοθήκη που χρησιμοποιείται για τη δημιουργία user interfaces και ειδικότερα εφαρμογές single page (front-end. Η εφαρμογή είναι χωρισμένη σε δύο βασικά τμήματα: το front-end & το back-end τα οποία συνδέονται και επικοινωνούν μεταξύ τους σύμφωνα με το παρακάτω σχεδιάγραμμα:

A screenshot of a phone

Description automatically generated

Για το front-end, η React επιτρέπει τη δυναμική δημιουργία της διεπιφάνειας χρήστη (user interface) η οποία βασίζεται σε ένα σύνολο από components, η λογική των οποίων γράφεται σε Javascript και η τελική τους μορφή είναι η HTML.

Για το back-end, η Express JS βοηθά στη δρομολόγηση και την οργάνωση των αιτημάτων του client εκτελώντας ένα σύνολο HTTP αιτημάτων που αντιστοιχούν στις ενέργειες που πραγματοποιούνται κατά την εκτέλεση της εφαρμογής, όπως κουμπιά, επεξεργασία και διαγραφή οντοτήτων, φόρτωση συνόλου περιεχομένου χρηστών από τη βάση.

## Modules και βιβλιοθήκες

Κυριότερα modules τα οποία χρησιμοποιήθηκαν στο back-end είναι:

1. *Typescript*: Ένα υπερσύνολο της JavaScript που προσθέτει στατική πληκτρολόγηση και διεπαφές, επιτρέποντας στους προγραμματιστές να γράφουν πιο ισχυρό και χωρίς σφάλματα κώδικα.
2. *Mongoose*: Μια βιβλιοθήκη Object Data Modeling (ODM) για MongoDB και Node.js, η οποία απλοποιεί τις αλληλεπιδράσεις με το MongoDB παρέχοντας μοντελοποίηση δεδομένων βάσει σχήματος
3. *Winston*: Μια ευέλικτη βιβλιοθήκη καταγραφής για εφαρμογές Node.js, η οποία επιτρέπει προσαρμόσιμα επίπεδα, μορφές και μεταφορές καταγραφής.
4. *JWT* (JSON Web Token): Ένα συμπαγές, ασφαλές για URL μέσο αναπαράστασης αξιώσεων προς μεταφορά μεταξύ δύο μερών, που χρησιμοποιείται συνήθως για ασφαλή έλεγχο ταυτότητας και ανταλλαγή πληροφοριών.
5. *Nodemon*: επιτρέπει την αυτόματη φόρτωση της σελίδας μετά από κάθε αλλαγή που μπορεί να συμβεί στον server
6. *Node* (MERN Stack): Ένα περιβάλλον χρόνου εκτέλεσης JavaScript που βασίζεται στον κινητήρα V8 του Chrome, το οποίο επιτρέπει στους προγραμματιστές να εκτελούν κώδικα JavaScript από την πλευρά του διακομιστή, επιτρέποντας τη δημιουργία κλιμακούμενων εφαρμογών δικτύου υψηλής απόδοσης

Οι κυριότερες βιβλιοθήκες που χρησιμοποιήθηκαν στο front-end είναι:

1. *Shadcn*: Μια βιβλιοθήκη για τη δημιουργία συνεπών και προσβάσιμων στοιχείων διεπαφής χρήστη σε εφαρμογές React χρησιμοποιώντας Tailwind CSS και άλλα σύγχρονα εργαλεία.
2. *React* *router*: Μια βιβλιοθήκη για δρομολόγηση σε εφαρμογές React που επιτρέπει την πλοήγηση μεταξύ διαφορετικών προβολών ή σελίδων σε μια εφαρμογή μιας σελίδας.
3. *Tanstack query*: Μια ισχυρή βιβλιοθήκη ανάκτησης δεδομένων για το React που απλοποιεί τη διαχείριση κατάστασης διακομιστή, την προσωρινή αποθήκευση και το συγχρονισμό, επιτρέποντας την αποτελεσματική φόρτωση και ενημέρωση δεδομένων.
4. *React redux:* Μια βιβλιοθήκη που παρέχει συνδέσμους μεταξύ React και Redux, επιτρέποντας στα στοιχεία του React να αλληλεπιδρούν με το κατάστημα Redux και να διαχειρίζονται την κατάσταση κεντρικά.
5. *React hook format*: Μια αποδοτική, ευέλικτη και εύχρηστη βιβλιοθήκη επικύρωσης φόρμας για το React που αξιοποιεί τα hook για τη διαχείριση της κατάστασης και της επικύρωσης φόρμας με ελάχιστες εκ νέου απόδοση.
6. *Lucide*: Μια βιβλιοθήκη στοιχείων React για την ενσωμάτωση προσαρμόσιμων εικονιδίων και εικονιδίων ανοιχτού κώδικα σε εφαρμογές React, με βάση το δημοφιλές σύνολο εικονιδίων Lucide.
7. *Zod*: Μια βιβλιοθήκη δήλωσης και επικύρωσης σχήματος πρώτου τύπου TypeScript που επιτρέπει στους προγραμματιστές να ορίζουν και να επικυρώνουν δομές δεδομένων με ισχυρή ασφάλεια τύπου και ελάχιστο boilerplate.
8. *React Select*: Μια ευέλικτη και προσαρμόσιμη αναπτυσσόμενη/επιλογή βιβλιοθήκη εισόδου για το React που παρέχει υποστήριξη για επιλογές μίας και πολλαπλής επιλογής, ασύγχρονη φόρτωση και προσαρμοσμένο στυλ.
9. *Tailwind CSS*: Ένα πλαίσιο CSS για πρώτη φορά το βοηθητικό πρόγραμμα που επιτρέπει στους προγραμματιστές να δημιουργήσουν γρήγορα προσαρμοσμένες διεπαφές χρήστη εφαρμόζοντας προκαθορισμένες κλάσεις απευθείας στη δομή HTML, προωθώντας τη συνέπεια σχεδιασμού και την επαναχρησιμοποίηση.

# Δοκιμή Συστήματος

## Functionality Testing

Το functionality testing της εφαρμογής πραγματοποιήθηκε παράλληλα με τη συγγραφη του κώδικα και την εκτέλεση της εφαρμογής με σκοπό να αποφευχθεί η σπατάλη χρόνου στην περίπτωση που έπρεπε να συμβούν αλλαγές ή διορθώσεις. Ο έλεγχος αυτού του σταδίου αφορά:

* Την αναμενόμενη ανταπόκριση των λιστών, των εικονιδίων και των κουμπιών
* Την εμφάνιση των μηνυμάτων ελέγχου και παράλειψης υποχρεωτικών πεδίων ή λανθασμένου μήκους κάποιου συγκεκριμένου πεδίου
* Την εμφάνιση μηνυμάτων επιβεβαίωσης πραγματοποίησης ενεργειών (εμφάνιση μηνύματος κατά την προσθήκη κάποιου χρήστη ή μιας νέας ενέργειας)
* Την ανανέωση της βάσης δεδομένων μετά από ενέργειες που τη μεταβάλλουν
* Τη μετάβαση σε διαφορετικές σελίδες της εφαρμογής
* Τη σύνδεση και αποσύνδεση του εκάστοτε χρήστη
* Την εμφάνιση νέων παραθύρων ή στοιχείων της σελίδας (εμφάνιση ιστορικού εργασίας και εισαγωγή σχολίων)

Πιο αναλυτικές περιπτώσεις ελέγχου αποτελούν οι ακόλουθες:

**Δημιουργία εργασίας**

* Επαλήθευση δυνατότητας χρήστη να δημιουργήσει μια νέα εργασία
* Επαλήθευση σωστής αποθήκευσης των πεδίων της εργασίας

**Επεξεργασία εργασίας**

* Επαλήθευση δυνατότητας διαχειριστή χρήστη να επεξεργαστεί τα πεδία μιας εργασίας
* Επαλήθευση μη δυνατότητας απλού χρήστη να προβεί στην ίδια ενέργεια
* Επαλήθευση αλλαγής και επεξεργασίας των στοιχείων στη βάση
* Επαλήθευση αλλαγής κατάστασης εργασίας μετά από αντίστοιχη ενέργεια
* Επαλήθευση λειτουργίας φίλτρου αναζήτησης στη λίστα εργασιών (και όχι μόνο)

**Διαγραφή εργασίας**

* Επαλήθευση διαγραφής από το διαχειριστή
* Επαλήθευση διαγραφής από τη βάση μετά τη

**Βάση δεδομένων**

* Επαλήθευση σωστής αποθήκευσης και αλλαγής δεδομένων μετά από αντίστοιχες ενέργειες στην εφαρμογή

## User Testing

Το User Testing εστιάζει στην αξιολόγηση της εφαρμογής διαχείρισης εργασιών από την οπτική γωνία του τελικού χρήστη για να διασφαλίσει ότι είναι διαισθητική, φιλική προς το χρήστη και ανταποκρίνεται στις ανάγκες του. Αυτός ο τύπος δοκιμών είναι ζωτικής σημασίας για την κατανόηση του τρόπου με τον οποίο οι πραγματικοί χρήστες αλληλεπιδρούν με την εφαρμογή και τον εντοπισμό τυχόν προβλημάτων χρηστικότητας

Σημεία που οφείλουν να ελεγχθούν:

**Ευχρηστία διεπαφής χρήστη (UI).**

* Πλοήγηση: έλεγχος ευκολίας πλοήγησης χρηστών στην εφαρμογή (μετακίνηση μεταξύ διαφορετικών ενοτήτων όπως εργασίες, χρήστες και ρυθμίσεις)
* Διάταξη: αξιολόγηση διάταξης της εφαρμογής (εάν είναι διαισθητική και εάν σημαντικές δυνατότητες είναι εύκολα προσβάσιμες)
* Απόκριση: έλεγχος απόκρισης διεπαφής χρήστη στις ενέργειες του χρήστη, χωρίς καθυστερήσεις.
* Συνέπεια: έλεγχος πως τα στοιχεία σχεδίασης (κουμπιά, μενού, εικονίδια) είναι συνεπή σε όλη την εφαρμογή

**Εμπειρία χρήστη (UX)**

* Ευκολία χρήσης: ευκολία κατανόησης και χρήσης της εφαρμογής από νέους χρήστες χωρίς ύπαρξη οδηγιών
* Ροή: Αποτελεσματικότητα ολοκλήρωσης διεργασιών
* Διαχείριση σφαλμάτων: έλεγχος για σαφή παροχή μηνυμάτων σχετικών με σφάλματα όταν κάτι πάει στραβά (εισαγωγή μη έγκυρων δεδομένων)

**Προσβασιμότητα**

* Δυνατότητες προσβασιμότητας της εφαρμογής με προγράμματα ανάγνωσης οθόνης και άλλες βοηθητικές τεχνολογίες
* Μέγεθος γραμματοσειράς και αντίθεση χρώματος για χρήστες με προβλήματα όρασης
* Πλοήγηση πληκτρολογίου στην εφαρμογή χρησιμοποιώντας συντομεύσεις πληκτρολογίου, εάν υπάρχουν

**Απόδοση σε σενάρια πραγματικού κόσμου**

* Χρόνοι φόρτωσης εφαρμογής ειδικά όταν πρόκειται για μεγάλο αριθμό εργασιών
* Χρήση μπαταρίας και πόρων της εκάστοτε συσκευής

**Ικανοποίηση χρήστη**

* Σχόλια χρήστη (ή χρηστών, ανάλογα με το πόσα άτομα μπορούν να πραγματοποιήσουν αυτό τον έλεγχο) σχετικά με τη συνολική ικανοποίησή τους από την εφαρμογή, τι τους αρέσει και τι βρίσκουν απογοητευτικό
* Αιτήματα δυνατοτήτων για πρόσθετες λειτουργίες

**Αναμενόμενα αποτελέσματα:**

• Θετικά σχόλια χρήστη όσον αφορά τη χρήση, με λογική ροή που καθιστά τη διαχείριση εργασιών απλή

• Λίγα ζητήματα χρηστικότητας. Εάν προκύψουν σημαντικά ζητήματα, θα πρέπει να αντιμετωπιστούν όσο το δυνατόν νωρίτερα

• Η εφαρμογή θα πρέπει να μπορεί να χρησιμοποιηθεί από άτομα με διάφορες αναπηρίες, πληρώντας κοινά πρότυπα προσβασιμότητας.

• Αποτελεσματική ολοκλήρωση εργασιών (όπως η δημιουργία, η επεξεργασία και η διαγραφή εργασιών) χωρίς σύγχυση ή περιττά βήματα.

# Επίλογος

## Συμπεράσματα

Μέσα από την ανάπτυξη και τον έλεγχο της εφαρμογής, επιβεβαιώθηκε η χρησιμότητά της και η ευκολία χρήσης της είτε όταν πρόκειται για τη συνεργασία μιας ομάδας σε μια εταιρεία είτε και στην περίπτωση συνεργασίας φοιτητών με σκοπό την ολοκλήρωση κάποιας ομαδικής εργασίας.

Τα συμπεράσματα αυτά ενισχύουν τη σημαντικότητα ενός τέτοιου εργαλείου, αφού βοηθά στην κατανομή φόρτου και στην έγκαιρη και εύκολη επικοινωνία ανάμεσα στους χρήστες που συμμετέχουν στην εκτέλεσή του.

Τα βασικά χαρακτηριστικά του συστήματος διαχείρισης εργασιών τα οποία είναι απαραίτητα για την ομαλή λειτουργία του είναι:

* Επικοινωνία βάσης δεδομένων με την εφαρμογή και επικοινωνία της εφαρμογής με τους χρήστες
* Ταχύτητα αλληλεπίδρασης και ανταπόκρισης της εφαρμογής
* Ευκολία χρήσης χωρίς χρονοβόρα επεξήγηση η οποία καθυστερεί τόσο τους χρήστες όσο και την ολοκλήρωση του στόχου
* Ευκολία χρήσης διεπιφάνειας χρήστη

## Μελλοντικές επεκτάσεις - Βελτιώσεις

* Real-time φόρτωσης και αναβάθμισης μέσω web sockets, κάτι το οποίο μέχρι στιγμής δε συμβαίνει
* Groups και group manager: δημιουργία νέας οντότητας (database table) η οποία θα διαχειρίζεται ομάδες χρηστών με σκοπό την ευκολότερη πρόσβαση στις εργασίες που ανήκουν σε αυτή την ομάδα και πιο εκτεταμένη ορατότητα αυτών των εργασιών
* Περισσότερες κατηγορίες χρηστών εκτός του διαχειριστή και του απλού χρήστη με διαφορετικά δικαιώματα
* Ενσωμάτωση διαφορετικών γλωσσών
* Dark mode
* Προσθήκη φίλτρων για αναλυτικότερη αναζήτηση σε όλες τις σελίδες της εφαρμογής
* Προσθήκη στήλης στη λίστα εργασιών για την ημερομηνία/ώρα έναρξης και λήξης κάθε εργασίας
* Διερεύνηση και διόρθωση προβλήματος εντολής yarn
* Επέκταση της χρήσης του πεδίου approved (schema user) για περαιτέρω επαλήθευση και έλεγχο πολλαπλών χρηστών

# Πηγές

* **Τεχνολογίες**

MERN Stack

REACT: https://react.dev/

Express JS: https://en.wikipedia.org/wiki/Express.js

* **Σχεδίαση Διαγραμμάτων της Βάσης Δεδομένων**

Draw.io: <https://app.diagrams.net/>

* **Front-end modules**

<https://ui.shadcn.com/>

<https://reactrouter.com/en/main>

<https://tanstack.com/query/latest/docs/framework/react/overview>

<https://redux.js.org/>

<https://react-hook-form.com/>

<https://lucide.dev/>

<https://zod.dev/>

<https://react-select.com/home>

https://tailwindcss.com/

* **User Testing**

ΗΥ-364 Διάλεξη σχετικά με την αξιολόγηση: https://www.csd.uoc.gr/~hy364/gr/lectures.php