

Ολοκληρωμένο Σύστημα Διαχείρισης Ακαδημαϊκών Έργων

Θεόδωρος Φλωρέντζης

Επιβλέπων: Βασίλειος Δημακόπουλος

Ιωάννινα, Φεβρουάριος, 2025



**ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA**

Περίληψη

Η παρούσα διπλωματική εργασία επικεντρώνεται στην ανάπτυξη ενός ολοκληρωμένου συστήματος διαχείρισης ακαδημαϊκών έργων (Integrated Project Management System For Academic Environments) με χρήση ενός low-code περιβάλλοντος ανάπτυξης εφαρμογών ιστού.

Σκοπός ήταν η ανάπτυξη ενός εσωτερικού εργαλείου για την διαχείριση ακαδημαϊκών έργων που καλύπτει την πλειοψηφία των αναγκών των ακαδημαϊκών μελών και περιλαμβάνει βασικές λειτουργίες όπως: δημιουργία έργων και εργασιών, ανάθεση ρόλων, διαχείριση συναντήσεων (meeting), κοινοποίηση αρχείων κ.α.

Παρακάτω θα παρουσιάσουμε τι είναι τα εσωτερικά εργαλεία (internal tools), μια γενικότερη έρευνα σχετικά με τα low-code και no-code framework για την κατασκευή εσωτερικών εργαλείων, γιατί επιλέξαμε την low-code πλατφόρμα Appsmith για την ανάπτυξη της εφαρμογής, καθώς και τους λόγους που οδήγησαν στην ανάγκη ανάπτυξης ενός τέτοιου εργαλείου.

Λέξεις Κλειδιά: Διαχείριση έργων-πρότζεκτ, Low-code framework, No-code framework, Appsmith

Abstract

This thesis is focused on developing an Integrated Project Management System for Academic Environments using a low-code web application development platform. The aim was to create a custom project management tool that addresses most academic staff needs, offering core functionalities such as project and task creation, role assignment, meeting management, file sharing, and more.

The thesis presents internal tools, a comprehensive study of what are low-code and no-code frameworks and why we chose the low-code platform Appsmith for the application's development. It also explores the reasons that necessitated the creation of such a tool and discusses why existing solutions were inadequate.

Keywords: Project Management, Low-code Framework, No-code Framework, Appsmith

Περιεχόμενα

Κεφάλαιο 1. Εισαγωγή.....	6
1.1 Διαχείριση ακαδημαϊκών ομάδων	6
1.2 Υπάρχον λογισμικό διαχείρισης project.....	7
1.3 Αντικείμενο της διπλωματικής.....	7
1.4 Δομή της διπλωματικής.....	8
Κεφάλαιο 2. Εσωτερικά εργαλεία (Internal Tools)	10
2.1 Εισαγωγή στα internal tools	10
2.2 Τύποι internal tools	10
2.3 Project management tools	11
2.4 Custom internal tools.....	12
2.5 Προσεγγίσεις για την ανάπτυξη internal tools: Custom κώδικας και frameworks	13
2.6 Internal tools σε πανεπιστημιακά ιδρύματα.....	14
2.7 Κριτήρια επιλογής ανάπτυξης custom εργαλείου	15
Κεφάλαιο 3. Πλατφόρμες ανάπτυξης εσωτερικών εργαλείων no/low-code	17
3.1 Εισαγωγή στις No/Low-Code πλατφόρμες	17
3.2 Πλατφόρμες No/Low-code.....	18
3.2.1 Nocobase.....	18
3.2.2 Budibase.....	19
3.2.3 Refine.....	21
3.2.4 ToolJet.....	23
3.2.5 Appsmith.....	24
3.3 Σύγκριση των δωρεάν πακέτων των framework.....	26
3.4 Τελική επιλογή του framework	26

Κεφάλαιο 4. Academic Project Management (APM): Ένα νέο εργαλείο για διαχείριση ακαδημαϊκών ομάδων	28
4.1 Ακαδημαϊκές ομάδες και διαχείριση project.....	28
4.2 Απαιτήσεις του APM	29
4.3 Λειτουργίες του APM	42
4.3.1 Dashboard.....	43
4.3.2 Project Dashboard.....	44
4.3.3 Task.....	47
4.3.4 Meeting.....	48
4.3.5 My Preferences.....	49
4.3.6 Manage Users	50
4.3.7 Ειδοποιήσεις.....	51
4.4 Αρχιτεκτονική του APM	53
4.4.1 Βάση δεδομένων.....	54
4.4.2 Appsmith.....	57
4.4.3 Nginx Server.....	58
Κεφάλαιο 5. Υλοποίηση στο Appsmith.....	60
5.1 Διαχείριση και ταυτοποίηση χρηστών	60
5.2 Σύνδεση βάσης δεδομένων και υλοποίηση queries	64
5.3 Γραφικό περιβάλλον με χρήση έτοιμων widget.....	68
5.4 Υλοποίηση των user role.....	73
5.5 Υλοποίηση ημερολογίου με χρήση custom widget	76
Κεφάλαιο 6. Συμπεράσματα.....	79
6.1 Αντικείμενο της εργασίας	79
6.2 Περιγραφή υλοποίησης	79
6.3 Προβλήματα και προκλήσεις	80
6.4 Μελλοντικές επεκτάσεις	81

Βιβλιογραφία - Πηγές	83
Παράρτημα κώδικα.....	85
Κώδικας Εικόνα 24 – Μέθοδοι Log-in σελίδας	85
Κώδικας Εικόνα 34 – Queries <i>getCurrentUserProjects</i> και <i>getCurrentUserActiveProjects</i>	86
Κώδικας Εικόνα 35 – Μέθοδος <i>navigateToProject</i>	87
Κώδικας Εικόνα 38 – Μέθοδος <i>createOrSubscribeToProject</i>	88
Κώδικας Εικόνα 39 – Μέθοδος <i>createNewProject</i>	88
Κώδικας Εικόνα 40 – Query <i>addProjectToCurrentUser</i>	89
Κώδικας Εικόνα 42 – Μέθοδοι <i>shouldWidgetsBeDisabled</i> και <i>shouldWidgetBeVisible</i>	90

Κεφάλαιο 1 - Εισαγωγή

Η διαχείριση έργων (Project Management) αποτελεί έναν κρίσιμο και πολυδιάστατο τομέα, που επηρεάζει την αποτελεσματική λειτουργία οργανισμών σε πολλούς κλάδους, μεταξύ αυτών και των ακαδημαϊκών ιδρυμάτων. Στην σύγχρονη τεχνολογική εποχή, τα ακαδημαϊκά ιδρύματα, καλούνται να διαχειριστούν ένα μεγάλο όγκο δεδομένων, δραστηριοτήτων και διαδικασιών που δεν αφορούν μόνο την διδασκαλία αλλά και την έρευνα, την συνεργασία των ακαδημαϊκών μελών, την παρακολούθηση της προόδου των έργων, τη διαχείριση πόρων και την αξιολόγηση των αποτελεσμάτων.

Η αυξανόμενη πολυπλοκότητα των έργων, οι αυστηρές προθεσμίες και οι περιορισμοί στους διαθέσιμους πόρους εντείνουν την ανάγκη για καινοτόμες και αποδοτικές προσεγγίσεις στη διαχείριση. Σε αυτό το πλαίσιο, η χρήση εξειδικευμένων εργαλείων διαχείρισης έργων, προσαρμοσμένων στις ανάγκες της ακαδημαϊκής κοινότητας, καθίσταται ζωτικής σημασίας. Η χρήση ενός εργαλείου project management, μπορεί να προσφέρει λύσεις που όχι μόνο διευκολύνουν την κατανομή των πόρων και την τήρηση χρονοδιαγραμμάτων, αλλά παράλληλα ενισχύουν τη διαφάνεια, την παραγωγικότητα και τη συνολική ποιότητα των παρεχόμενων εκπαιδευτικών και ερευνητικών υπηρεσιών.

1.1 Διαχείριση ακαδημαϊκών ομάδων

Τα ακαδημαϊκά ιδρύματα αποτελούνται από ποικίλες ομάδες, κάθε μία από τις οποίες έχει ξεχωριστούς στόχους και ανάγκες. Η διαχείριση τους αποτελεί μια πολυδιάστατη διαδικασία που περιλαμβάνει τον συντονισμό καθηγητών, φοιτητών, και ερευνητικών ομάδων για την επίτευξη κοινών στόχων, όπως η εκπόνηση διπλωματικών εργασιών, η ανάπτυξη ερευνητικών έργων και η οργάνωση μαθημάτων. Κάποιες από τις ανάγκες που προκύπτουν από τις παραπάνω δραστηριότητες περιλαμβάνουν τη διαχείριση εργασιών (tasks), την ανάθεση ρόλων, την παρακολούθηση προόδου, την τήρηση προθεσμιών (deadlines) και τη διασφάλιση αποτελεσματικής επικοινωνίας μεταξύ των μελών της ομάδας.

Τα εργαλεία διαχείρισης έργων παρέχουν μια ολοκληρωμένη και συστηματική προσέγγιση για την αντιμετώπιση αυτών των προκλήσεων. Ειδικότερα, προσφέρουν τη δυνατότητα καταγραφής και οργάνωσης όλων των διαδικασιών, διευκολύνουν την κατανομή των αρμοδιοτήτων και επιτρέπουν τη συνεχή παρακολούθηση της προόδου. Επίσης επιτρέπουν στα ακαδημαϊκά μέλη να διαχειρίζονται με μεγαλύτερη ευελιξία και αποδοτικότητα τις πολύπλοκες απαιτήσεις των ομάδων τους, ενισχύοντας τη συνεργασία και διασφαλίζοντας υψηλά επίπεδα παραγωγικότητας και ποιότητας [1]. Με αυτόν τον τρόπο, τέτοιες λύσεις συμβάλλουν στη βελτίωση της συνολικής εμπειρίας μάθησης και έρευνας εντός των ακαδημαϊκών ιδρυμάτων.

1.2 Υπάρχον λογισμικό διαχείρισης project

Τα διαθέσιμα λογισμικά διαχείρισης έργων έχουν αναπτυχθεί για να καλύψουν ένα ευρύ φάσμα αναγκών, από μικρές ομάδες των 10-20 ατόμων μέχρι ολόκληρους οργανισμούς και επιχειρήσεις χιλιάδων υπαλλήλων. Παρόλο που παρέχουν πολλά χρήσιμα χαρακτηριστικά, η πολυπλοκότητά τους και πολλές φορές η υπερεξειδίκευση τους στον τομέα των επιχειρήσεων τα καθιστά ακατάλληλα για χρήση σε ακαδημαϊκά περιβάλλοντα.

Εργαλεία όπως το Microsoft Project, το Jira και το Asana, είναι σχεδιασμένα για εταιρική χρήση, με έμφαση σε εταιρικές ροές εργασίας και προσεγγίσεις διαχείρισης έργων όπως το Agile [2] ή το Scrum [3]. Αυτές ενώ είναι απαραίτητες για την εύρυθμη λειτουργία των εταιρειών λογισμικού και την οργάνωση των ομάδων τους, συχνά δεν ευθυγραμμίζονται με τις ανάγκες των ακαδημαϊκών έργων, όπου η φύση των εργασιών είναι περισσότερο ερευνητική, πολυεπιστημονική και λιγότερο επαναληπτική. Επιπρόσθετα, η χρήση τους απαιτεί συχνά εκτενή εκπαίδευση για να μπορέσουν οι χρήστες να αξιοποιήσουν αποτελεσματικά τις δυνατότητες που προσφέρουν. Τέλος, να σημειωθεί πως τα συγκεκριμένα εργαλεία έχουν υψηλό χρηματικό κόστος, το οποίο τις περισσότερες φορές δεν μπορεί να καλυφθεί από πανεπιστημιακά ιδρύματα και οργανισμούς εκτός του επιχειρηματικού κλάδου.

Καταλαβαίνουμε λοιπόν πως οι ακαδημαϊκές ομάδες έχουν ιδιαίτερες απαιτήσεις που δεν καλύπτονται πλήρως από τις υπάρχουσες λύσεις που προσφέρονται στην αγορά. Η ανάπτυξη ενός ειδικά προσαρμοσμένου λογισμικού για τη διαχείριση ακαδημαϊκών έργων μπορεί να καλύψει αυτές τις ανάγκες με έναν πιο αποδοτικό και αποτελεσματικό τρόπο, προσαρμόζοντάς το πάνω στις ανάγκες της ακαδημαϊκής κοινότητας.

1.3 Αντικείμενο της διπλωματικής

Η παρούσα εργασία εστιάζει στη σχεδίαση και ανάπτυξη ενός εξειδικευμένου εργαλείου διαχείρισης έργων, ειδικά προσαρμοσμένου στις ανάγκες των ακαδημαϊκών ιδρυμάτων. Το εργαλείο αυτό έχει στόχο να αντιμετωπίσει τα προβλήματα και τις προκλήσεις που περιεγράφηκαν προηγουμένως, προσφέροντας μια λύση που θα επιτρέπει την ομαλή και αποδοτική διαχείριση ακαδημαϊκών έργων, όπως διπλωματικές εργασίες, ερευνητικά project και άλλες εκπαιδευτικές δραστηριότητες.

Μεταξύ άλλων η εφαρμογή που αναπτύξαμε στοχεύει να προσφέρει λειτουργίες όπως:

- **Διαχείριση έργων:** Δημιουργία και διαχείριση έργων με δυνατότητες προσθήκης εργασιών, συναντήσεων και κοινοποίηση αρχείων με απλό και κατανοητό τρόπο.
- **Χρονοπρογραμματισμός:** Δυνατότητα διαχείρισης χρόνου με λειτουργίες όπως διαδραστικό ημερολόγιο και πίνακες επικείμενων προθεσμιών των διαφόρων οντοτήτων.

- **Αυτόματες Ειδοποιήσεις:** Ειδοποιήσεις μέσω email για διάφορες αλλαγές που αφορούν τους χρήστες της εφαρμογής όπως την δημιουργία του λογαριασμού τους στην εφαρμογή, ανάθεσης κάποιας εργασίας κα.
- **Συνεργασία:** Ενίσχυση της αποδοτικής επικοινωνίας μεταξύ μελών μιας ομάδας μέσω της χρήσης ενός κοινού εργαλείου, που καλύπτει όλες τις ανάγκες τους.
- **Παραμετροποίηση της εφαρμογής:** Στους χρήστες προσφέρονται δυνατότητες παραμετροποίησης όπως αλλαγές χρωμάτων και εμφάνιση ή όχι των διάφορων widget της εφαρμογής.

Για την υλοποίηση του εργαλείου επιλέχθηκε η χρήση της πλατφόρμας Appsmith, ενός low-code framework ανοιχτού κώδικα που εξειδικεύεται στην ανάπτυξη εσωτερικών εργαλείων. Σε επόμενα κεφάλαια θα εξηγήσουμε αναλυτικότερα την έννοια των εργαλείων no/low-code και γιατί επιλέξαμε συγκεκριμένα το Appsmith για την υλοποίηση της εφαρμογής μας.

Η παρούσα εργασία στοχεύει να προσφέρει ένα πρακτικό παράδειγμα χρήσης τεχνολογιών no/low-code για την επίλυση εξειδικευμένων προβλημάτων στο χώρο της ακαδημαϊκής διαχείρισης. Επιπλέον, ευελπιστεί να αποτελέσει μια βάση για περαιτέρω επεκτάσεις και προσαρμογές, προσφέροντας την δυνατότητα ανάπτυξης νέων λειτουργιών καλύπτοντας μελλοντικές ανάγκες των ακαδημαϊκών ιδρυμάτων.

1.4 Δομή της διπλωματικής

Η διπλωματική εργασία αποτελείται από έξι κύρια κεφάλαια, με στόχο να προσφέρει στον αναγνώστη μια ολοκληρωμένη κατανόηση τόσο του προβλήματος που αντιμετωπίζουμε όσο και της λύσης που προτείνουμε, αναφορικά με τη διαχείριση έργων σε ακαδημαϊκά ιδρύματα.

Στο 2^ο κεφάλαιο ξεκινάμε με μια γενική επισκόπηση των εσωτερικών εργαλείων (Internal tools). Παρουσιάζουμε τους διάφορους τύπους τους, δίνοντας έμφαση στα εργαλεία διαχείρισης έργων (project management tools). Επιπλέον, αναλύουμε εκτενώς την έννοια των προσαρμοσμένων εσωτερικών εργαλείων (custom internal tools) και τα συγκρίνουμε με τις έτοιμες λύσεις του εμπορίου. Ιδιαίτερη έμφαση δίνεται στα πλεονεκτήματα που προσφέρουν τα custom εργαλεία και τα εξειδικευμένα no/low-code framework για την κατασκευή τους. Τέλος, διερευνούμε τα γενικότερα οφέλη που μπορεί να αποκομίσουν τα ακαδημαϊκά ιδρύματα από τη χρήση τέτοιων εργαλείων, όπως η βελτίωση της παραγωγικότητας, η ενίσχυση της συνεργασίας και η αποδοτικότερη διαχείριση των πόρων.

Στο 3^ο κεφάλαιο, παρουσιάζουμε την έρευνά μας σχετικά με πέντε εργαλεία no/low-code που εντοπίσαμε και αξιολογήσαμε. Αναλύουμε συνοπτικά τα πλεονεκτήματα και τα μειονεκτήματα του κάθε εργαλείου, ενώ παράλληλα παραθέτουμε μια συγκριτική ανάλυση των λειτουργιών που προσφέρουν στο πλαίσιο του δωρεάν πακέτου τους. Επιπλέον, εξηγούμε διεξοδικά τους λόγους για τους οποίους επιλέξαμε το Appsmith ως

την ιδανική λύση για την ανάπτυξη του εσωτερικού εργαλείου μας, λαμβάνοντας υπόψη τις απαιτήσεις του έργου και τα χαρακτηριστικά που προσφέρει το εργαλείο.

Συνεχίζοντας στο 4^ο κεφάλαιο, εστιάζουμε στη δημιουργία της νέας μας εφαρμογής για τη διαχείριση ακαδημαϊκών έργων. Αρχικά, παρέχουμε μια γενική ανάλυση σχετικά με τις ακαδημαϊκές ομάδες και τα έργα που καλούνται να διαχειριστούν, τονίζοντας τις ανάγκες και τις προκλήσεις που αντιμετωπίζουν. Στη συνέχεια, παρουσιάζουμε το εργαλείο που υλοποιήσαμε, αναλύοντας τις λειτουργίες του μέσα από τις σελίδες της εφαρμογής. Τέλος, ολοκληρώνουμε το κεφάλαιο με μια παρουσίαση της αρχιτεκτονικής του συστήματος, εξηγώντας πώς οι επιμέρους συνιστώσες συνεργάζονται για την επίτευξη ενός αποδοτικού και ολοκληρωμένου αποτελέσματος.

Στο 5^ο κεφάλαιο, παρουσιάζουμε αναλυτικά την υλοποίηση της εφαρμογής μας στο Appsmith, περιγράφοντας τη διαδικασία σχεδιασμού και ανάπτυξης. Εστιάζουμε σε χαρακτηριστικές περιπτώσεις χρήσης και συγκεκριμένες ανάγκες που αναδείχθηκαν κατά την ανάπτυξη, εξετάζοντας πώς διαχειριστήκαμε αυτές τις προκλήσεις αξιοποιώντας τις δυνατότητες που παρέχει το framework αναλύοντας τις στρατηγικές και τις τεχνικές που χρησιμοποιήσαμε για την επίτευξη των απαιτούμενων λειτουργιών.

Τέλος, στο 6^ο και τελευταίο κεφάλαιο, παρουσιάζουμε συνοπτικά το αντικείμενο της εργασίας καθώς και τις προκλήσεις και τα προβλήματα που αντιμετωπίσαμε καθ' όλη την διάρκεια ανάπτυξης του εργαλείου. Επίσης αναφέρουμε κάποιες μελλοντικές προεκτάσεις και βελτιώσεις που μπορούν να υλοποιηθούν στην εφαρμογή μας.

Κεφάλαιο 2 - Εσωτερικά εργαλεία (Internal Tools)

2.1 Εισαγωγή στα internal tools

Με τον όρο «Εσωτερικά Εργαλεία» (Internal tools) αναφερόμαστε σε εξειδικευμένες εφαρμογές λογισμικού που αναπτύσσονται για να υποστηρίξουν τις εσωτερικές διαδικασίες ενός οργανισμού όπως ενός ακαδημαϊκού ιδρύματος ή μιας επιχείρησης και συνήθως είναι μη ελεύθερα προσβάσιμες σε εξωτερικούς χρήστες (π.χ. είναι προσβάσιμες μόνο μέσω VPN του εκάστοτε οργανισμού). Τα internal tools μπορούν να αφορούν διαφορετικούς τομείς του οργανισμού και συνήθως αποσκοπούν στην συγκέντρωση, διαχείριση, ανάλυση και επεξεργασία δεδομένων και πληροφορίας με σκοπό την διευκόλυνση των χρηστών στην εξαγωγή συμπερασμάτων, την λήψη αποφάσεων και την γενικότερη διαχείριση διαφορετικών τομέων του οργανισμού.

Η χρήση και ανάπτυξη τέτοιων εργαλείων, δεν είναι κάτι καινούργιο καθώς παρατηρείται από τα πρώτα χρόνια της ανάπτυξης λογισμικού (software development) [4]. Σήμερα η πλειοψηφία των εταιρειών και των οργανισμών χρησιμοποιούν εσωτερικά εργαλεία για αύξηση της αποδοτικότητας τους.

2.2 Τύποι internal tools

Τα internal tools καλύπτουν διάφορες κατηγορίες εφαρμογών που εξυπηρετούν τις εσωτερικές ανάγκες ενός οργανισμού και μπορεί να περιλαμβάνουν από μία απλή εφαρμογή για διαχείριση TODOs, ένα γραφικό περιβάλλον για μία βάση δεδομένων μέχρι wiki που καλύπτουν τις ανάγκες όλου του οργανισμού και προσφέρουν πιο πολύπλοκες λειτουργίες [4]. Αυτό που πρέπει να γίνει κατανοητό σε αυτό το σημείο είναι πως η πολυπλοκότητα της εφαρμογής δεν παίζει κανέναν ρόλο στην κατηγοριοποίηση της ως 'Internal Tool'. Σημασία έχει ο τρόπος χρήσης της εφαρμογής και όχι η πολυπλοκότητα της.

Κάποιοι από τους βασικούς τύπους εσωτερικών εργαλείων μαζί με κάποια παραδείγματα εφαρμογών του εμπορίου παρατίθενται παρακάτω [5]:

- 1. Εργαλεία Συνεργασίας Και Επικοινωνίας:** Εσωτερικές πλατφόρμες επικοινωνίας. Προσφέρουν δυνατότητες ανταλλαγής μηνυμάτων, βιντεοκλήσεων, κοινή χρήση αρχείων κα. Παραδείγματα: Slack, Microsoft Teams.
- 2. Εργαλεία Διαχείρισης Έργων (Project Management Tools):** Διαχείριση, οργάνωση και παρακολούθηση έργων. Στα πλαίσια αυτού του τύπου των internal tools διαχειρίζονται αναθέσεις εργασιών σε χρήστες,

χρονοπρογραμματισμός των έργων, παρακολούθηση εξέλιξης των έργων κα. Παραδείγματα: JIRA, Asana.

3. **Εργαλεία Διαχείρισης Αρχείων (DMS):** Εργαλεία κοινής χρήσης και επεξεργασίας αρχείων. Ουσιαστικά ένα εσωτερικό «wiki». Περιέχει την απαραίτητη «γνώση» που πρέπει να είναι κοινή στους εσωτερικούς χρήστες. Κάποια παραδείγματα ενοτήτων σε ένα τέτοιο εργαλείο είναι συχνές ερωτήσεις, οδηγοί, κοινή επεξεργασία αρχείου, προτεινόμενο υλικό-πηγές κα. Παραδείγματα: Confluence.
4. **Συστήματα Ενδοεπιχειρησιακού Σχεδιασμού (ERP - Enterprise Resource Planning):** Λογισμικό για την διαχείριση βασικών διαδικασιών μια επιχείρησης όπως την παραγωγή, τη διαχείριση αποθεμάτων, τις χρηματοοικονομικές υπηρεσίες, τις πωλήσεις, τους ανθρώπινους πόρους κα. Παραδείγματα: SAP, Entersoft.
5. **Συστήματα Διαχείρισης Ανθρώπινου Δυναμικού (HRMS):** Χρησιμοποιούνται για την διαχείριση του ανθρωπίνου δυναμικού (HR) ενός οργανισμού-επιχείρησης. Μέσω αυτών των εφαρμογών ο εργαζόμενος μπορεί να έχει πρόσβαση σε οργανογράμματα, πληροφορίες αδείας, γενικές πληροφορίες επικοινωνίας, έγγραφα (όπως σύμβαση εργασίας, πολιτικές της εταιρείας) κα. Παραδείγματα: Bamboo.

Η παραπάνω λίστα σε καμία περίπτωση δεν είναι εξαντλητική αλλά μας δίνει μια ιδέα για το ευρύ φάσμα των εφαρμογών και των αναγκών που μπορούν να καλύψουν τα internal tools.

2.3 Project management tools

Τα Project Management Tools είναι μια υποκατηγορία εσωτερικών εργαλείων που ειδικεύεται στην οργάνωση, διαχείριση και παρακολούθηση των έργων/ομάδων ενός οργανισμού. Εξυπηρετούν την ανάγκη για τον συντονισμό των εργασιών, την κατανομή των πόρων, την παρακολούθηση των χρονοδιαγραμμάτων και την αποτελεσματική επικοινωνία μεταξύ των μελών των ομάδων. Αναλόγως την πολυπλοκότητα του εργαλείου, μπορεί να ενσωματώνει λειτουργίες, όπως η δημιουργία και η ανάθεση εργασιών (tasks), ο χρονικός προγραμματισμός, η παρακολούθηση της προόδου και η εκτίμηση του κόστους και των πόρων που απαιτούνται για την ολοκλήρωση των έργων κα. [6] [7].

Στόχος τους είναι η βελτίωση της αποδοτικότητας των ομάδων, καθώς επιτρέπουν την καλύτερη οργάνωση του έργου και τη διασφάλιση της τήρησης των προθεσμιών. Στην αγορά υπάρχουν πολλές λύσεις εργαλείων διαχείρισης έργων που καλύπτουν διαφορετικές ανάγκες, ανάλογα με την πολυπλοκότητα των έργων και τον τύπο του οργανισμού.

Με τη σωστή επιλογή και χρήση ενός τέτοιου εργαλείου, οι οργανισμοί μπορούν να ενισχύσουν την παραγωγικότητα τους, να βελτιώσουν τη συνεργασία και να διασφαλίσουν την επιτυχή ολοκλήρωση των έργων τους [7].

2.4 Custom internal tools

Τα εργαλεία διαχείρισης έργων, όπως και πολλές κλασικές εφαρμογές, διατίθενται στην αγορά και προσφέρονται, στην πλειοψηφία τους, με την μορφή συνδρομής στους ενδιαφερόμενους οργανισμούς με περιορισμένες δυνατότητες προσαρμογής (customization). Δηλαδή ο ενδιαφερόμενος πελάτης πρέπει να διαλέξει από μια πληθώρα έτοιμων εφαρμογών με συγκεκριμένες δυνατότητες και χαρακτηριστικά (features) και να επιδιώξει να βρει το λογισμικό που καλύπτει σε μεγαλύτερο ποσοστό τις ανάγκες του.

Από την άλλη πλευρά εκτός από την έλλειψη λειτουργιών, πολλές φορές το πρόβλημα που αντιμετωπίζουν οργανισμοί με το έτοιμο εσωτερικό λογισμικό που προσφέρεται είναι η μεγαλύτερη πολυπλοκότητα του [7]. Στην πλειοψηφία τους, τέτοιου είδους εφαρμογές προσφέρουν πολύ περισσότερες λειτουργίες, για να καλύψουν όσο το δυνατόν περισσότερες ανάγκες που μπορεί να προκύψουν από τον εκάστοτε οργανισμό με αποτέλεσμα την αύξηση της πολυπλοκότητας, την δυσκολία παραμετροποίησής τους και εν τέλει την δυσκολία της χρήσης τους.

Φανταστείτε να θέλει ένα πανεπιστήμιο ένα απλό εργαλείο διαχείρισης των φοιτητών του αλλά λόγω έλλειψης ενός τέτοιου εξειδικευμένου εργαλείου να χρησιμοποιεί ένα internal tool προορισμένο για εταιρική χρήση διαχείρισης πελατών (CRM). Καταλαβαίνουμε πως οι ανάγκες ενός πανεπιστημίου είναι εντελώς διαφορετικές για την διαχείριση των φοιτητών του και στην συγκεκριμένη περίπτωση μπορεί να υπάρχει περιττή πληροφορία όπως αριθμός παραγγελιών, τζίρος από πελάτη κτλ. που ουδεμία σχέση έχουν με τον ακαδημαϊκό τομέα.

Η ανάγκη λοιπόν των οργανισμών, για την χρήση εξειδικευμένων εργαλείων τους οδήγησε στην επιλογή της υλοποίησης custom εσωτερικών εργαλείων. Η επιλογή της υλοποίησης custom internal tools επιτρέπει στους οργανισμούς να διαμορφώσουν ένα εργαλείο απόλυτα προσαρμοσμένο στις ιδιαίτερες απαιτήσεις και προκλήσεις που αντιμετωπίζουν. Ένα τέτοιο εργαλείο δεν περιορίζεται από τις γενικές δυνατότητες των έτοιμων εφαρμογών της αγοράς, αλλά επικεντρώνεται αποκλειστικά στις ανάγκες του εκάστοτε οργανισμού. Αυτό σημαίνει ότι κάθε λειτουργία, από τη διαχείριση δεδομένων μέχρι τη διασύνδεση με υπάρχουσες υποδομές, σχεδιάζεται εξ αρχής με βάση τις συγκεκριμένες απαιτήσεις του οργανισμού.

Μερικά από τα πλεονεκτήματα της ανάπτυξης προσαρμοσμένων εργαλείων είναι [8]:

- **Ακρίβεια στην κάλυψη των αναγκών:** Τα custom internal tools σχεδιάζονται για να καλύψουν ακριβώς τις ανάγκες του οργανισμού, αποφεύγοντας περιττές λειτουργίες που μειώνουν την αποδοτικότητα και την ευχρηστία του εργαλείου.

- **Παραμετροποίηση:** Η υλοποίηση custom internal tools, δίνει τον μεγαλύτερο βαθμό ευελιξίας και παραμετροποίησης στους οργανισμούς, καλύπτοντας και πιθανόν μελλοντικές τους ανάγκες.
- **Απλότητα στην χρήση:** Συνήθως η υλοποίηση custom εσωτερικών εργαλείων, λόγω της εξειδίκευσης τους πάνω στον τρόπο με τον οποίο λειτουργεί μια ομάδα/οργανισμός, εξαλείφει την ανάγκη για πολύωρη εκπαίδευση μειώνοντας την σπατάλη χρόνου και πόρων.
- **Μειωμένο οικονομικό κόστος:** Η ανάπτυξη ενός custom εργαλείου μπορεί να είναι μια πιο φθηνή λύση, ειδικά εάν ο οργανισμός διαθέτει το εργατικό δυναμικό για την ανεξάρτητη ανάπτυξη του εργαλείου, καθώς η μακροχρόνια χρήση του χωρίς συνδρομές και οι μειωμένες ανάγκες για εξωτερική υποστήριξη καθιστούν την επιλογή αυτή πιο βιώσιμη.

Φυσικά η ανάπτυξη ενός custom internal tool, έρχεται και με αρκετές προκλήσεις και μειονεκτήματα που πρέπει να ληφθούν υπόψιν [8]:

- **Κόστος Προσωπικού:** Η διαδικασία ανάπτυξης μπορεί να έχει υψηλό κόστος σε επίπεδο προσωπικού καθώς θα στερήσει από τον οργανισμό εργατικό δυναμικό κατά την διάρκεια ανάπτυξης του εργαλείου.
- **Ανεπαρκής τεχνογνωσία:** Σε περίπτωση που οργανισμός δεν έχει την απαραίτητη τεχνογνωσία για την υλοποίηση του επιθυμητού εργαλείου, το οικονομικό κόστος αυξάνεται καθώς είτε θα πρέπει να προσληφθεί το κατάλληλο προσωπικό είτε να χρησιμοποιηθεί μια εξωτερική ομάδα προγραμματιστών για την ανάπτυξη του εργαλείου.
- **Τεχνική υποστήριξη:** Η συντήρηση και η τυχόν αναβάθμιση του εργαλείου απαιτεί μια σταθερή ομάδα υποστήριξης με επαρκή τεχνογνωσία.

2.5 Προσεγγίσεις για την ανάπτυξη internal tools: Custom κώδικας και frameworks

Η παραδοσιακή ανάπτυξη εσωτερικών εργαλείων, μέσω custom κώδικα, αποτελεί μια διαδικασία που, αν και ιδιαίτερα ευέλικτη, συνοδεύεται από πολλές προκλήσεις και περιορισμούς. Η δημιουργία ενός εργαλείου από την αρχή απαιτεί ενδελεχή σχεδιασμό, προηγμένες δεξιότητες προγραμματισμού και σημαντικούς χρονικούς και οικονομικούς πόρους. Το επίπεδο της απαιτούμενης τεχνογνωσίας επηρεάζεται σε μεγάλο βαθμό από την πολυπλοκότητα του επιθυμητού εργαλείου και τις λειτουργίες που καλείται να καλύψει.

Για την ανάπτυξη ενός internal tool, και οποιασδήποτε εφαρμογής, απαιτείται αρχικά η προσεκτική σχεδίαση και ανάλυση των απαιτήσεων, προκειμένου να διασφαλιστεί ότι το τελικό αποτέλεσμα θα καλύπτει πλήρως τις ανάγκες του οργανισμού. Επιπλέον, είναι απαραίτητο να συμφωνηθεί το κατάλληλο περιβάλλον ανάπτυξης της εφαρμογής, το οποίο θα επηρεάσει τη συνολική αποτελεσματικότητα και ευελιξία του εργαλείου. Η διαδικασία περιλαμβάνει την επιλογή της γλώσσας προγραμματισμού που θα

χρησιμοποιηθεί, καθώς και τη σχεδίαση και υλοποίηση του back-end, το οποίο είναι υπεύθυνο για τη διαχείριση της custom λογικής της εφαρμογής, τη διασύνδεση με το front-end και την επικοινωνία με βάσεις δεδομένων ή άλλες εξωτερικές υπηρεσίες. Παράλληλα, απαιτείται η ανάπτυξη του front-end, με την υλοποίηση του γραφικού περιβάλλοντος (User Interface), αξιοποιώντας τεχνολογίες όπως HTML και CSS για τη δημιουργία λειτουργικών και αισθητικά ευχάριστων σελίδων.

Τα παραπάνω είναι μερικές από τις διαδικασίες και αποφάσεις που μια ομάδα προγραμματιστών θα κληθεί να πάρει για τον σχεδιασμό ενός custom εσωτερικού εργαλείου. Στην εποχή μας, με την αύξηση της πληροφορίας και της τεχνολογικής ανάπτυξης η ζήτηση εξειδικευμένων εσωτερικών εργαλείων από τους διάφορους οργανισμούς και επιχειρήσεις αυξάνεται συνεχώς με αποτέλεσμα να καθιστά την παραδοσιακή ανάπτυξη εφαρμογών ανεπαρκή και αναποτελεσματική από πλευράς χρόνου και κόστους.

Η αυξανόμενη αυτή ανάγκη και ζήτηση των οργανισμών για γρήγορη και αποδοτική δημιουργία προσαρμοσμένων εσωτερικών εργαλείων, οδήγησε στην εμφάνιση διαφόρων framework που δίνουν την δυνατότητα ανάπτυξης custom εργαλείων, μειώνοντας το απαιτούμενο επίπεδο τεχνογνωσίας και τον χρόνο υλοποίησης παρέχοντας ευκολίες στις προγραμματιστικές ομάδες που καλούνται να υλοποιήσουν το οποιοδήποτε εργαλείο. Framework όπως το Appsmith, που θα δούμε αναλυτικότερα παρακάτω, παρέχουν ευκολίες για την σχεδίαση του γραφικού περιβάλλοντος, την διασύνδεση των δεδομένων, την σχεδίαση και χρήση API, την μετέπειτα διαχείριση της εφαρμογής κ.ά. που επιταχύνουν κατά πολύ τον απαιτούμενο χρόνο ανάπτυξης ενός ολοκληρωμένου εργαλείου.

Με αυτόν τον τρόπο, η χρήση εργαλείων no/low-code όπως το Appsmith προσφέρει μια σημαντική εναλλακτική λύση σε σχέση με τον παραδοσιακό τρόπο υλοποίησης εφαρμογών. Ενώ η ανάπτυξη με custom κώδικα παρέχει απόλυτη ευχέρεια και πλήρη έλεγχο στις δυνατότητες της εφαρμογής, απαιτεί όμως υψηλό κόστος σε χρόνο και πόρους, καθώς και εξειδικευμένη τεχνογνωσία. Αντίθετα, τα no/low-code frameworks επιτρέπουν την ταχεία ανάπτυξη εξειδικευμένων εργαλείων με λιγότερες απαιτήσεις σε τεχνικές γνώσεις και σε μικρότερο χρονικό διάστημα εστιάζοντας στην ποιότητα του παραγόμενου εργαλείου. Η προσέγγιση αυτή συνδυάζει την ευελιξία του custom κώδικα με την απλότητα και την ταχύτητα των σύγχρονων εργαλείων, κάνοντάς τα μια ιδανική λύση για οργανισμούς που χρειάζονται προσωποποιημένες εφαρμογές χωρίς τις προκλήσεις της παραδοσιακής ανάπτυξης. Αυτοί είναι και οι λόγοι για τους οποίους επιλέξαμε την προσέγγιση των no/low-code frameworks για την υλοποίηση του εργαλείου διαχείρισης έργων.

2.6 Internal tools σε πανεπιστημιακά ιδρύματα

Όπως έχει προαναφερθεί, η χρήση των internal tools δεν περιορίζεται μόνο σε επιχειρήσεις αλλά μπορούν να επωφεληθούν από αυτά, σε μεγάλο βαθμό, τα πανεπιστημιακά ιδρύματα και άλλοι τύποι οργανισμών. Λαμβάνοντας υπόψιν την φύση

των ακαδημαϊκών ιδρυμάτων, που χαρακτηρίζεται από σύνθετες δομές, μεγάλο όγκο δεδομένων και ανάγκες για συνεργασία μεταξύ πολλών διαφορετικών ομάδων, καθηγητών και φοιτητών καθώς και τις ανάγκες της σύγχρονης ψηφιακής εποχής, θα μπορούσαμε να πούμε πως τα εσωτερικά εργαλεία είναι πλέον απαραίτητα για την ομαλή και αποδοτική λειτουργία κάθε πανεπιστημιακού ιδρύματος.

Η πανδημία του covid, έδωσε ίσως το καλύτερο παράδειγμα του πόσο αναγκαίος είναι ο ψηφιακός μετασχηματισμός και η χρήση τέτοιων εργαλείων στα ακαδημαϊκά ιδρύματα. Εργαλεία επικοινωνίας, διαχείρισης διοικητικών διαδικασιών, διαχείρισης ακαδημαϊκών έργων, ασύγχρονης εκπαίδευσης είναι μόνο μερικά από τα παραδείγματα internal tools, που φάνηκαν απαραίτητα ώστε τα πανεπιστήμια να καταφέρουν να συνεχίσουν να παρέχουν υψηλού επιπέδου εκπαίδευση χωρίς συμβιβασμούς.

Εκτός όμως από ακραία φαινόμενα, όπως μια πανδημία, τα εσωτερικά εργαλεία μπορούν και υπό κανονικές συνθήκες λειτουργίας να κάνουν ευκολότερη και πιο αποδοτική την καθημερινότητα του ακαδημαϊκού προσωπικού και των φοιτητών. Για παράδειγμα, η εφαρμογή ενός εργαλείου διαχείρισης έργων θα μπορούσε να προσφέρει σημαντικά οφέλη τόσο στους καθηγητές όσο και στους φοιτητές. Από την μία οι καθηγητές καλούνται να διαχειριστούν μια πληθώρα μαθημάτων, διδακτορικών και ερευνητικών προγραμμάτων που το καθένα έρχεται με πλήθος συναντήσεων και εργασιών, και από την άλλη, οι φοιτητές επωφελούνται από την αμεσότητα και τη διαφάνεια στην ανάθεση και στον έλεγχο των εργασιών τους.

Ένα ακόμα σημαντικό παράδειγμα αποτελεί η χρήση εργαλείων επικοινωνίας. Πολλά πανεπιστήμια υιοθετούν σύγχρονες πλατφόρμες όπως το Microsoft 365 για χρήση τους ως internal tool για την βελτίωση της εσωτερικής τους επικοινωνίας [9]. Η δυνατότητα για ομαδικές συνομιλίες, βιντεοκλήσεις και κοινή χρήση αρχείων, καθιστά την καθημερινή λειτουργία του πανεπιστημίου πιο αποδοτική και οργανωμένη.

Τέλος, τα εργαλεία ασύγχρονης εκπαίδευσης, όπως παραδείγματος χάριν στην περίπτωση του Πανεπιστημίου Ιωαννίνων η πλατφόρμα ecourse, δίνουν τη δυνατότητα στους φοιτητές να παρακολουθούν διαλέξεις, να συμμετέχουν σε forum συζητήσεων και να υποβάλλουν εργασίες από οπουδήποτε και οποτεδήποτε. Αυτό όχι μόνο διευκολύνει την πρόσβαση στην εκπαίδευση αλλά και προάγει την αυτονομία και την υπευθυνότητα των φοιτητών στη διαχείριση του χρόνου και των υποχρεώσεών τους.

2.7 Κριτήρια επιλογής ανάπτυξης custom εργαλείου

Τα οφέλη μιας custom υλοποίησης είναι πολλά. Στην δική μας περίπτωση χρειαζόμασταν ένα εργαλείο διαχείρισης έργων στοχευμένο κυρίως στην διαχείριση διπλωματικών, διδακτορικών αλλά και ερευνητικών προγραμμάτων. Οι λόγοι που μας οδήγησαν στην ανάπτυξη ενός custom εργαλείου είναι πολλοί και ποικίλοι.

Καταρχάς, το κόστος απόκτησης και συντήρησης ενός έτοιμου εμπορικού εργαλείου θα ήταν αρκετά υψηλό και ενδεχομένως υπερβολικό για ένα εργαλείο που προορίζεται για χρήση από λίγους καθηγητές ενός τμήματος του πανεπιστημίου. Για να καταλάβουμε καλύτερα το εύρος τιμών, εργαλεία όπως το Asana και το Jira προσφέρουν βασικές εκδόσεις με κόστος περίπου 9-12 ευρώ ανά χρήστη μηνιαίως για την standard έκδοση και 17-30 ευρώ για την premium έκδοση [10] [11] . Με μια ομάδα των 20 ατόμων το ετήσιο κόστος ανέρχεται σε 2160-2880 ευρώ για την απλή έκδοση και 4080-7200 ευρώ για την πλήρη έκδοση. Καταλαβαίνουμε λοιπόν πως το κόστος είναι αρκετά υψηλό για ένα εργαλείο που θα χρησιμοποιείται από περιορισμένο αριθμό χρηστών, όπως είναι οι καθηγητές ενός συγκεκριμένου τμήματος, καθιστώντας την ανάπτυξη ενός custom εργαλείου με χρήση no/low-code framework πιο οικονομικά βιώσιμη λύση.

Επιπλέον, πέρα από το κόστος, διαπιστώσαμε ότι δεν υπήρχε κάποιο διαθέσιμο εργαλείο διαχείρισης έργων στην αγορά που να καλύπτει πλήρως τις ειδικές και διαφοροποιημένες ανάγκες μας. Τα υπάρχοντα εργαλεία ήταν είτε υπερβολικά στις λειτουργίες που προσφέρουν αυξάνοντας κατά πολύ την πολυπλοκότητά τους και την περίοδο εκμάθησής τους, είτε εστίαζαν σε χαρακτηριστικά που δεν ήταν ουσιώδη για την αποτελεσματική διαχείριση διπλωματικών, διδακτορικών και ερευνητικών έργων, περιορίζοντας τη χρησιμότητά τους για το ακαδημαϊκό μας περιβάλλον.

Επίσης με την ανάπτυξη ενός custom εργαλείου διαχείρισης έργων έχουμε σημαντικά πλεονεκτήματα όσον αφορά την κάλυψη μελλοντικών αναγκών και παραμετροποιήσεων που μπορεί να προκύψουν. Καθώς οι ανάγκες του πανεπιστημίου και των χρηστών του εξελίσσονται με το χρόνο, ένα έτοιμο εμπορικό εργαλείο ενδέχεται να μην είναι αρκετά ευέλικτο για να ανταποκριθεί σε νέες απαιτήσεις ή αλλαγές στην ακαδημαϊκή διαδικασία. Αντίθετα, ένα custom σύστημα μπορεί να προσαρμοστεί και να εξελιχθεί σύμφωνα με τις μελλοντικές μας ανάγκες, προσφέροντας τη δυνατότητα ενσωμάτωσης νέων λειτουργιών, επεκτάσεων ή τροποποιήσεων χωρίς περιορισμούς ή εξωτερικές εξαρτήσεις. Με αυτόν τον τρόπο η επιλογή να υλοποιήσουμε ένα custom εργαλείο δεν είναι μόνο μια λύση για το παρόν, αλλά και μια ευέλικτη επένδυση για το μέλλον.

Τέλος, σημαντικό ρόλο έπαιξε και η δυνατότητα για πλήρη έλεγχο της ασφάλειας και της ιδιωτικότητας των δεδομένων μας. Επιλέγοντας ένα custom εργαλείο που θα φιλοξενείται σε δικούς μας διακομιστές, έχουμε τον πλήρη έλεγχο της υποδομής και των δεδομένων, διασφαλίζοντας την σωστή χρήση τους χωρίς συμβιβασμούς με εξωτερικές πολιτικές άλλων οργανισμών.

Κεφάλαιο 3 - Πλατφόρμες ανάπτυξης εσωτερικών εργαλείων no/low-code

3.1 Εισαγωγή στις No/Low-Code πλατφόρμες

Οι πλατφόρμες no/low-code είναι εργαλεία ανάπτυξης λογισμικού που επιτρέπουν τη δημιουργία εφαρμογών με ελάχιστη ή καθόλου χρήση κώδικα. Συνήθως προσφέρουν ένα γραφικό περιβάλλον εργασίας (GUI), το οποίο επιτρέπει στους χρήστες να σχεδιάζουν, να παραμετροποιούν και να υλοποιούν εφαρμογές μέσα από οπτικά εργαλεία, drag-and-drop λειτουργίες και προδιαμορφωμένα στοιχεία (widgets).

Πιο συγκεκριμένα τα no-code frameworks, στοχεύουν στην εξάλειψη της ανάγκης για γραφή κώδικα χρησιμοποιώντας σχεδόν αποκλειστικά μόνο κάποιο γραφικό περιβάλλον σε όλη την διάρκεια ανάπτυξης της εφαρμογής. Αντίθετα τα low-code frameworks, επιλέγουν μια πιο μετριασμένη προσέγγιση προσφέροντας τις ευκολίες των no-code frameworks (UI, drag-and-drop λειτουργίες κτλ.) χωρίς όμως να εμποδίζουν και τις παραμετροποιήσεις μέσω custom κώδικα (πχ με χρήση JavaScript σε προδιαμορφωμένα widgets για έλεγχο κάποιων πεδίων τους). Παρά τις διαφορές τους σε επίπεδο προσέγγισης, ο στόχος τους είναι κοινός. Να μειώσουν τον χρόνο ανάπτυξης, να απλοποιήσουν τη διαδικασία για μη προγραμματιστές ή για άτομα με λιγότερη τεχνογνωσία και να ενισχύσουν την παραγωγικότητα των ομάδων, επιτρέποντας την εύκολη και γρήγορη υλοποίηση εφαρμογών.

Για την επιλογή του no/low-code framework που θα χρησιμοποιούσαμε για την υλοποίηση της εφαρμογής μας, πραγματοποιήσαμε εκτενή έρευνα σχετικά με τις διαθέσιμες επιλογές. Βασικό μας κριτήριο ήταν η πλατφόρμα να είναι ανοιχτού κώδικα (open-source), ώστε να διασφαλίσουμε τη διαφάνεια, την προσαρμοστικότητα και το μειωμένο κόστος. Επιπλέον, δεδομένου ότι διαθέτουμε την απαραίτητη τεχνογνωσία, επιδιώξαμε η πλατφόρμα να υποστηρίζει παραμετροποίηση μέσω κώδικα, ώστε να μπορούμε να προσαρμόζουμε την εφαρμογή στις ανάγκες μας με μεγαλύτερη ευελιξία κατά τη διάρκεια της ανάπτυξης.

Στις επόμενες υποενότητες, θα παρουσιάσουμε τα αποτελέσματα της έρευνάς μας σχετικά με τις πλατφόρμες που πληρούν τα προαναφερθέντα κριτήρια, καθώς και τους λόγους που μας οδήγησαν στην επιλογή του Appsmith για την τελική μας υλοποίηση.

Αξίζει να σημειωθεί ότι, παρόλο που οι πλατφόρμες που εξετάζονται είναι ανοιχτού κώδικα, όλες διαθέτουν και εμπορικά πακέτα που προσφέρουν πρόσθετες δυνατότητες. Αυτό συμβαίνει διότι πολλά από αυτά τα εργαλεία απευθύνονται, μεταξύ άλλων, και σε επιχειρηματικούς οργανισμούς, ενισχύοντας έτσι την ευελιξία και τη λειτουργικότητά τους σε επαγγελματικά περιβάλλοντα.

3.2 Πλατφόρμες No/Low-code

3.2.1 Nocobase

Το NoCobase είναι μια πλατφόρμα για ανάπτυξη εσωτερικών εργαλείων και ανήκει στην κατηγορία των no-code frameworks [12]. Η πλατφόρμα είναι σχεδιασμένη να δίνει τη δυνατότητα σε χρήστες χωρίς προγραμματιστικές γνώσεις να δημιουργούν προσαρμοσμένες λύσεις μέσα από ένα απλό και κατανοητό περιβάλλον εργασίας. Επίσης προσφέρει ευέλικτες λύσεις στο κομμάτι του hosting της εφαρμογής με self-hosted άλλα και cloud-based επιλογές.

Ένα από τα βασικά χαρακτηριστικά του Nocobase είναι η modular αρχιτεκτονική του βασισμένη σε microservices [5], η οποία επιτρέπει την εύκολη επέκταση και παραμετροποίηση. Αυτός είναι και ο λόγος που αρχικά το συμπεριλάβαμε στις επιλογές μας.

Ουσιαστικά στο Nocobase όλες οι λειτουργίες είναι μοντελοποιημένες με την μορφή plugin. Από βασικές λειτουργίες όπως η σύνδεση με εξωτερικές βάσεις δεδομένων, έως πιο προχωρημένες όπως σύνδεση με email (Gmail, Outlook κτλ.), για διαχείριση ειδοποιήσεων (notification manager) κ.α. Επιπρόσθετα, για πιο εξειδικευμένες απαιτήσεις στην υπό ανάπτυξη εφαρμογή, δίνεται η δυνατότητα ανάπτυξης custom plugins, καθιστώντας την ιδανική για ομάδες που επιθυμούν έναν συνδυασμό απλότητας αλλά και ευελιξίας.

Κάποια παραδείγματα plugin που παρέχονται από το Nocobase είναι:

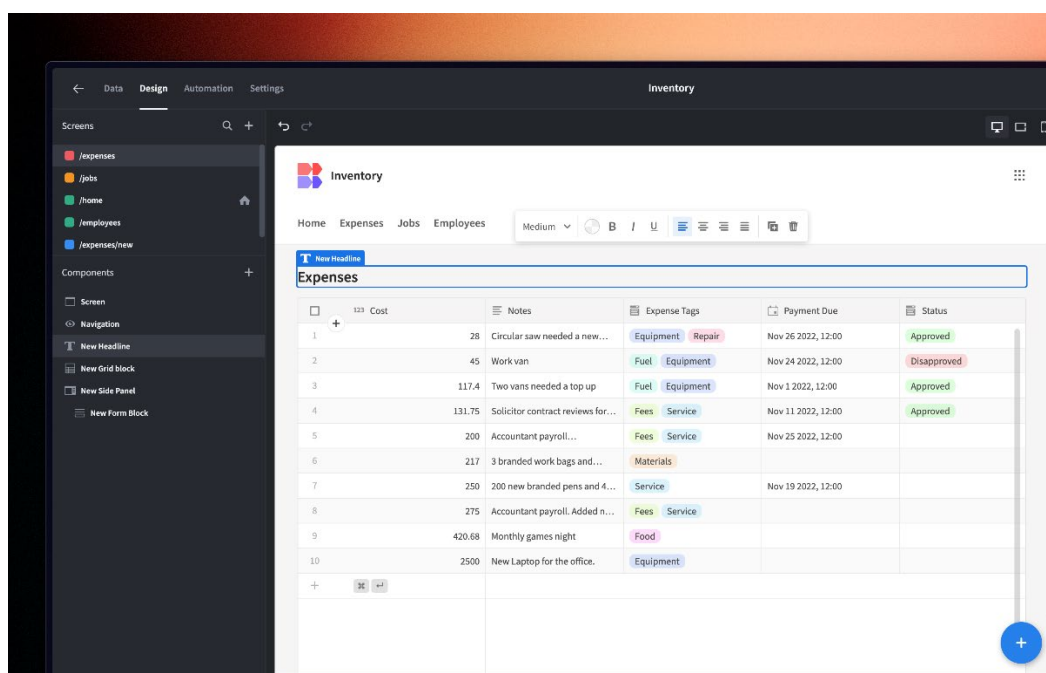
- **Data sources**
 - Main Database: Η κύρια βάση δεδομένων του Nocobase
 - External MySQL: Σύνδεση με εξωτερικές βάσεις MySQL
 - External PostgreSQL: Σύνδεση με εξωτερικές βάσεις PostgreSQL
 - HTTP API: Σύνδεση με εξωτερικά HTTP API
- **Blocks**
 - Kanban
 - Map
 - Iframe
 - Calendar
- **Action**
 - Scan QR code: Σκανάρισμα QR Code με μεταφορά στο αντίστοιχο URL
 - Custom Request: Αποστολή HTTP κλήσεων

Παρόλο που το Nocobase αποτελεί μια ευέλικτη πλατφόρμα, για την κατηγορία no-code, παρουσιάζει ορισμένα σημαντικά μειονεκτήματα. Αρχικά, παρόλο που το πλήθος των διαθέσιμων plugins για επέκταση των λειτουργιών του framework είναι μεγάλο, η πλειοψηφία τους παρέχεται μόνο επί πληρωμή [13]. Για παράδειγμα, η σύνδεση με εξωτερικές βάσεις δεδομένων όπως η MySQL, δεν παρέχεται σε δωρεάν μορφή. Επιπρόσθετα οι δυνατότητες που προσφέρει το Nocobase χωρίς πρόσθετες επεκτάσεις δεν παρέχουν την παραμετροποίηση που επιθυμούσαμε. Για παράδειγμα, δεν υπάρχει τρόπος να ελέγξεις την εμφάνιση και την λειτουργικότητα των διαφόρων widget μέσω CSS και JavaScript καθιστώντας τα αρκετά περιορισμένα και «στατικά».

Έτσι, καθώς και οι ενσωματωμένες δυνατότητες του εργαλείου δεν ήταν επαρκείς για τους σκοπούς μας και την φιλοσοφία ανάπτυξης που είχαμε, αλλά και το γεγονός ότι πολλές και κύριες λειτουργίες που έλειπαν δεν μπορούσαν να προστεθούν ως plugin μας απέτρεψε από την επιλογή του NoCobase ως το εργαλείο ανάπτυξης της εφαρμογής μας.

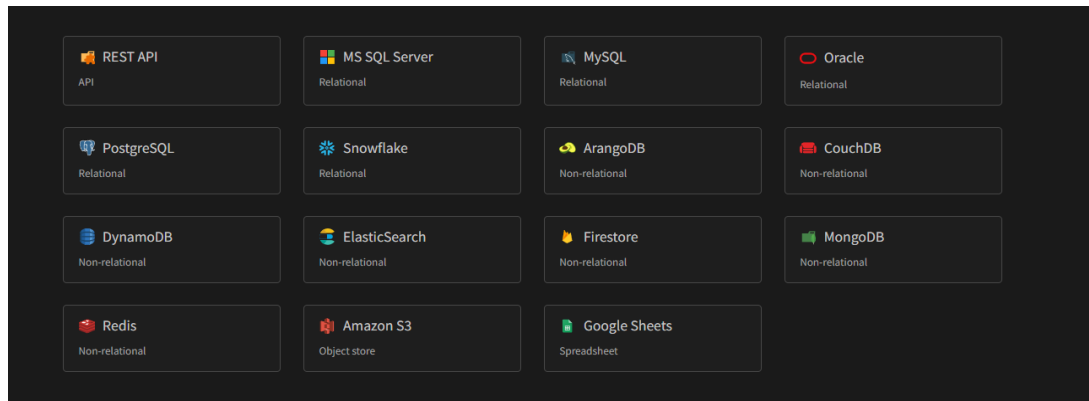
3.2.2 Budibase

Το Budibase είναι ένα low-code framework σχεδιασμένο για την ανάπτυξη εσωτερικών εργαλείων και επιχειρησιακών εφαρμογών με ταχύτητα και ευελιξία [14]. Το framework επιτρέπει στους χρήστες να αναπτύσσουν εφαρμογές μέσω γραφικού περιβάλλοντος, το οποίο φαίνεται στην Εικόνα 1, και αυτοματοποιημένων διαδικασιών, ενώ παράλληλα διατηρεί τη δυνατότητα προσαρμογής και επέκτασης για πιο εξειδικευμένες ανάγκες.



Εικόνα 1. Περιβάλλον ανάπτυξης του Budibase

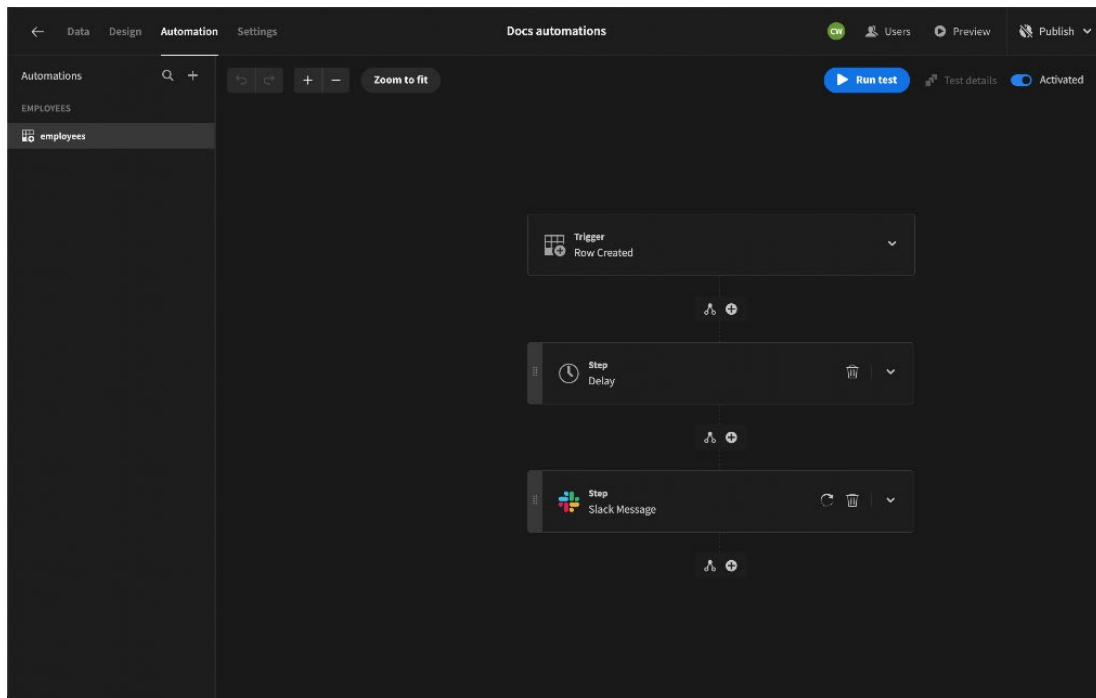
Αρχικά το εργαλείο παρέχει μια πληθώρα επιλογών για σύνδεση με εξωτερικές βάσεις δεδομένων όπως φαίνονται στην Εικόνα 2, επεκτείνοντας κατά πολύ τις δυνατότητες του και την προσαρμοστικότητα του σε διαφορετικά περιβάλλοντα ανάπτυξης.



Εικόνα 2. Βάσεις δεδομένων που υποστηρίζει το Budibase.

Επιπλέον, ενσωματώνει μια σειρά από προκαθορισμένα στοιχεία διεπαφής (components) όπως φόρμες, πίνακες, γραφήματα, κουμπιά, text editors κ.ά. τα οποία μπορούν να χρησιμοποιηθούν για την κατασκευή της εφαρμογής χωρίς να απαιτείται εξειδικευμένος προγραμματισμός από τον χρήστη. Αντίθετα όμως με μια no-code πλατφόρμα όπως το NocoBase, το Budibase δίνει την δυνατότητα παραμετροποίησης των components με διάφορους τρόπους. Παραδείγματος χάριν, εάν επιλέξουμε το «Headline» component που διατίθεται, ένα απλό component για επικεφαλίδες κειμένου, μπορούμε εκτός από τις προκαθορισμένες επιλογές που δίνονται για την εμφάνιση του κειμένου, να εισάγουμε custom CSS ή να καθορίσουμε εναλλαγές της εμφάνισης του component μέσω custom λογικής [15]. Επίσης παρέχεται και η δυνατότητα περιορισμένης χρήσης JavaScript σε συγκεκριμένους τομείς της εφαρμογής.

Μία από τις σημαντικότερες δυνατότητες του Budibase είναι η υποστήριξη αυτοματοποιημένων ροών εργασίας (automation). Οι χρήστες μπορούν να δημιουργούν ροές εργασίας για να αυτοματοποιούν τη διαδικασία διαχείρισης δεδομένων, τις ειδοποιήσεις και τις αλληλεπιδράσεις με τους χρήστες. Αυτές οι ροές μπορούν να ενσωματωθούν στο περιβάλλον ανάπτυξης και να εκτελούνται χωρίς την ανάγκη επιπλέον παρέμβασης. Το περιβάλλον ανάπτυξης αυτοματοποιημένων ροών εργασίας φαίνεται στην Εικόνα 3.



Εικόνα 3. Σελίδα ανάπτυξης αυτοματισμών στο Budibase.

Τέλος όπως και το Nocobase, το Budibase παρέχει δυνατότητες για hosting είτε σε τοπικούς διακομιστές με χρήση του Docker, είτε στο cloud εξασφαλίζοντας ευελιξία και συμβατότητα με τις ανάγκες πολλών οργανισμών.

Το Budibase, μαζί με το Appsmith που θα αναλύσουμε παρακάτω, αποτέλεσε μία από τις δύο βασικές επιλογές μας για την ανάπτυξη του εσωτερικού εργαλείου διαχείρισης έργων καθώς οι δυνατότητες παραμετροποίησης με κώδικα στα διάφορα component φάνηκε αρκετά δελεαστική συνδυάζοντας τις ευκολίες που επιθυμούσαμε με τις δυνατότητες customization.

3.2.3 Refine

Από τα εργαλεία που ανακαλύψαμε στην έρευνά μας, το Refine ξεχωρίζει ως ένα από τα πιο παραμετροποιήσιμα framework ανάπτυξης εσωτερικών εργαλείων ανάμεσα στις διαθέσιμες επιλογές [16]. Πρόκειται για μια πλατφόρμα που κατηγοριοποιείται ως low-code και βασίζεται σε React. Είναι κατάλληλο για ανάπτυξη αρκετών τύπων εφαρμογών όπως εσωτερικά εργαλεία (internal tools), πίνακες διαχείρισης (admin panels) και dashboards. Ο κύριος στόχος του Refine είναι να προσφέρει την καλύτερη δυνατή ισορροπία μεταξύ low-code και παραδοσιακού προγραμματισμού, συνδυάζοντας τα πλεονεκτήματα των low-code frameworks με τη δυνατότητα για εκτεταμένη παραμετροποίηση και εξατομίκευση του μέσω κώδικα. Για αυτόν τον λόγο, απευθύνεται περισσότερο σε προγραμματιστές με τεχνογνωσία που θέλουν να επιταχύνουν τον χρόνο ανάπτυξης της επιθυμητής εφαρμογής.

Η βασική φιλοσοφία του Refine είναι ότι αντί να περιορίσει τους προγραμματιστές σε κάποιες καθολικές επιλογές τους δίνει την ευχέρεια να επιλέξουν και να χτίσουν μόνοι τους το ιδανικό περιβάλλον για την εφαρμογή τους. Επιλογές όπως το UI που θα

χρησιμοποιηθεί, η πλατφόρμα ανάπτυξης κ.α. είναι μερικές από τις επιλογές που δίνονται. Με αυτόν τον τρόπο, το Refine συνδυάζει την ταχύτητα ανάπτυξης με την ευχέρεια για εξατομίκευση σε βάθος, ιδανικό για απαιτητικά έργα που ενδέχεται να απαιτούν εξατομικευμένες λειτουργίες. Στην Εικόνα 4 φαίνονται αναλυτικότερα οι διαθέσιμες επιλογές που προσφέρει η πλατφόρμα.



Εικόνα 4. Αρχιτεκτονική του Refine framework.

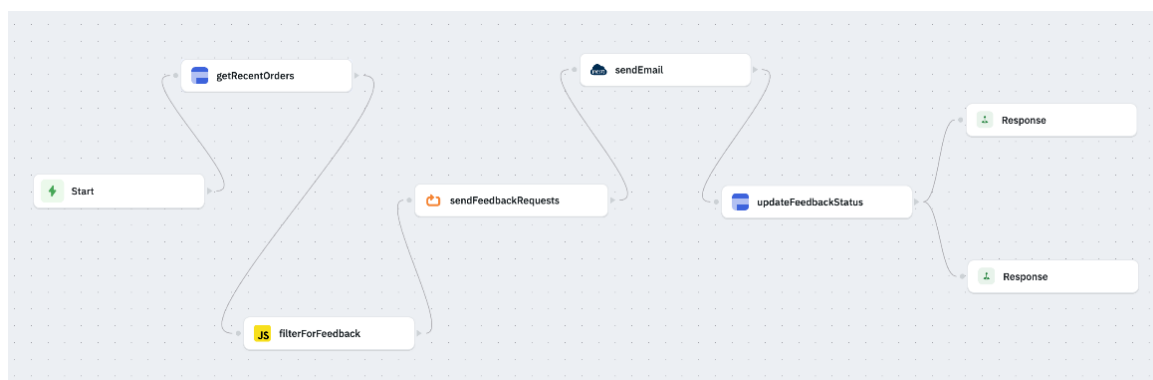
Συνολικά, το Refine ξεχωρίζει ως ένα framework που προσφέρει ταχύτητα ανάπτυξης και ευχρηστία μέσω του low-code μοντέλου, αλλά δεν περιορίζει τους χρήστες του όταν πρόκειται για παραμετροποίηση και εξατομίκευση των λειτουργιών της εφαρμογής. Αυτή η ισχυρή συνδυαστική προσέγγιση το καθιστά ιδανικό για ομάδες ανάπτυξης που επιθυμούν να διατηρήσουν την ευελιξία και την ισχύ του παραδοσιακού προγραμματισμού, ενώ ταυτόχρονα επωφελούνται από τα πλεονεκτήματα του low-code.

Από την δική μας πλευρά, οι δυνατότητες που προσφέρει το Refine δεν ήταν αρκετές για να το επιλέξουμε για την ανάπτυξη της τελικής μας εφαρμογής. Η χρονική περίοδος εκμάθησης του εργαλείου φάνηκε αρκετά υψηλή, ενώ οι δυνατότητες που προσφέρει ως low-code framework ήταν περιορισμένες σε σχέση με άλλα εργαλεία της κατηγορίας. Αν και προσφέρει μεγάλη παραμετροποίηση, αυτό καθιστά τη χρήση του λιγότερο φιλική προς τον χρήστη και πιο απαιτητική σε επίπεδο τεχνικής γνώσης. Στην ουσία, το Refine φάνηκε περισσότερο σαν μια βιβλιοθήκη για την ανάπτυξη εσωτερικών εργαλείων παρά ως πλήρες low-code εργαλείο.

3.2.4 ToolJet

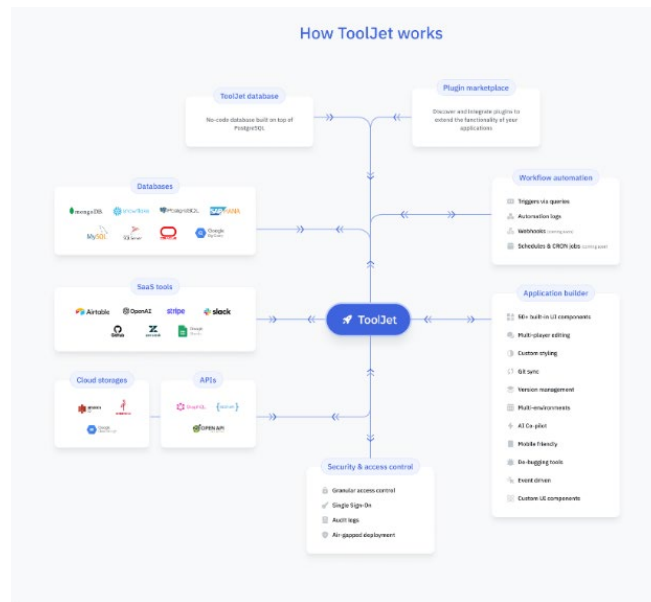
Το ToolJet, όπως και όλα τα framework που έχουμε παρουσιάσει, είναι ένα ανοιχτού κώδικα εργαλείο που ανήκει στην κατηγορία low-code για την ανάπτυξη εσωτερικών εργαλείων [17]. Αποτελείται από τρεις κύριες περιοχές ανάπτυξης:

- App-builder: Σχετίζεται με όλη την κατασκευή του UI της εφαρμογής μας με λειτουργίες όπως drag-and-drop και έτοιμα widgets όπως πίνακες, καρτέλες και γραφήματα.
- ToolJet Database: Αποτελεί την ενσωματωμένη PostgreSQL βάση δεδομένων του framework χωρίς ανάγκη για οποιοδήποτε set-up. Δίνει δυνατότητα ανάπτυξης και διαχείρισης της βάσης δεδομένων της εφαρμογής μας απευθείας μέσα από το περιβάλλον του ToolJet.
- Workflows: Παρόμοια λειτουργικότητα με το «Automation» που παρέχει το Budibase. Δίνει δυνατότητα με οπτικό τρόπο δημιουργίας ροών εργασιών που πρέπει να γίνουν έπειτα από κάποια συγκεκριμένη ενέργεια στην εφαρμογή. Παράδειγμα ενός «Workflow» φαίνεται στην Εικόνα 5.



Εικόνα 5. Παράδειγμα ενός Workflow στο ToolJet [10].

Η πλατφόρμα υποστηρίζει παραμετροποίηση μέσω JavaScript, επιτρέποντας στους χρήστες να προσθέτουν custom λογική και να ελέγχουν τη λειτουργικότητα των στοιχείων με μεγαλύτερη ακρίβεια. Φυσικά προσφέρει και δυνατότητα ενσωμάτωσης με εξωτερικές βάσεις δεδομένων (όπως MySQL, PostgreSQL) και υπηρεσίες όπως REST API, τα οποία φαίνονται και στην Εικόνα 6.



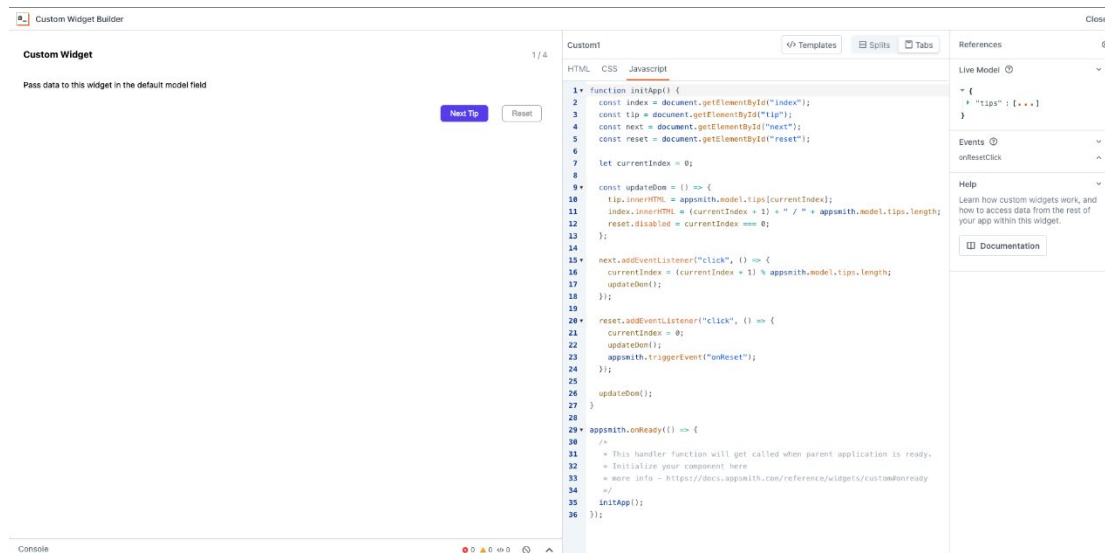
Εικόνα 6. Υψηλού επιπέδου αρχιτεκτονική του ToolJet

Ένας σημαντικός περιορισμός του ToolJet, τουλάχιστον για την δική μας περίπτωση, είναι η υποστήριξη του μόνο σε Linux λειτουργικά συστήματα. Όπως αναφέρει και το επίσημο documentation, η χρήση σε λειτουργικά συστήματα Windows είναι δυνατή είτε μέσω του WSL (Windows Subsystem For Linux) είτε με χρήση κάποιου virtual machine [18]. Αυτός ήταν και ο κύριος λόγος που δεν συμπεριλάβαμε το συγκεκριμένο framework στις τελικές μας επιλογές.

3.2.5 Appsmith

Το Appsmith είναι ένα ανοιχτού κώδικα low-code framework [19] για την ανάπτυξη εσωτερικών εργαλείων και εφαρμογών, που διακρίνεται για την ευκολία χρήσης και τις προηγμένες δυνατότητες παραμετροποίησης που προσφέρει. Όπως και το ToolJet, υποστηρίζει διασύνδεση με εξωτερικές βάσεις δεδομένων (όπως MySQL, PostgreSQL, MongoDB) και REST APIs, καθώς και ένα φιλικό προς τον χρήστη περιβάλλον ανάπτυξης με χαρακτηριστικά drag-and-drop.

Ένα από τα πιο αξιοσημείωτα χαρακτηριστικά του Appsmith είναι η εκτενής βιβλιοθήκη UI στοιχείων που προσφέρει. Αυτή περιλαμβάνει βασικά στοιχεία όπως πίνακες, φόρμες, γραφήματα, κουμπιά, αλλά και πιο εξειδικευμένα όπως το "Custom Widget" του οποίου το περιβάλλον ανάπτυξης φαίνεται στην Εικόνα 7. Μέσω αυτού, οι προγραμματιστές μπορούν να δημιουργήσουν πλήρως προσαρμοσμένα στοιχεία χρησιμοποιώντας HTML, CSS και JavaScript, ικανοποιώντας ακόμα και τις πιο σύνθετες σχεδιαστικές απαιτήσεις, παραμένοντας ταυτόχρονα στο περιβάλλον ενός εργαλείου low-code και των ευκολιών που προσφέρει.



Εικόνα 7. Περιβάλλον ανάπτυξης custom widget στο Appsmith

Όλα τα widgets του Appsmith είναι πλήρως παραμετροποιήσιμα, με δυνατότητα ενσωμάτωσης custom λογικής μέσω JavaScript. Οι χρήστες μπορούν να αλλάξουν τα χρώματα, τις λειτουργίες, ή ακόμα και να ορίσουν conditional λογική για τις τιμές που εμφανίζονται στα widgets, δημιουργώντας δυναμικές και διαδραστικές εφαρμογές. Από τις πιο απλές παραμετροποιήσεις έως σύνθετες λειτουργίες, το Appsmith προσφέρει ένα εύρος εργαλείων που απευθύνεται τόσο σε αρχάριους όσο και σε έμπειρους χρήστες, ίσως με τον πιο ισορροπημένο τρόπο που έχουμε συναντήσει.

Κάτι ακόμα που κάνει το Appsmith να ξεχωρίζει σε σύγκριση με τα υπόλοιπα frameworks που παρουσιάζονται, είναι το εξαιρετικά πλούσιο και καλά δομημένο documentation που προσφέρει. Το υλικό του περιλαμβάνει αναλυτικούς οδηγούς χρήσης, βίντεο εκμάθησης καθώς και παραδείγματα κώδικα διευκολύνοντας σημαντικά την εκμάθηση και την αποτελεσματική χρήση της πλατφόρμας.

Η ευελιξία και οι δυνατότητες προσαρμογής που προσφέρει το Appsmith το καθιστούν ιδανικό για οργανισμούς που αναζητούν ένα ισχυρό και προσαρμόσιμο εργαλείο ανάπτυξης εσωτερικών εφαρμογών.

3.3 Σύγκριση των δωρεάν πακέτων των framework

Όπως έχουμε αναφέρει, όλα τα προαναφερθέντα frameworks παρέχουν και επί πληρωμή αλλά και δωρεάν πακέτα χρήσης. Για την καλύτερη κατανόηση των λειτουργιών που προσφέρουν τα εργαλεία στην δωρεάν έκδοσή τους, παραθέτουμε παρακάτω τον Πίνακα 1 με κάποια από τα σημαντικότερα για μας χαρακτηριστικά, και το πώς διαμορφώνονται στην δωρεάν έκδοση των εργαλείων.

	NoCobase	Budibase	Refine	ToolJet	Appsmith
Επίπεδο Κώδικα (Code Level)	No-code	No-code/Low-code	Low-code React-based framework	Low-code	Low-code
Μέγιστος Αριθμός Χρηστών	Απεριόριστοι	Budibase Cloud: 5 χρήστες Self-host:20 χρήστες	Απεριόριστοι	Απεριόριστοι	Απεριόριστοι
Μέγιστος Αριθμός Εφαρμογών	Απεριόριστες	Απεριόριστες	Απεριόριστες	Απεριόριστες	Απεριόριστες
Εξωτερικές Βάσεις δεδομένων	Καμία	MySQL, PostgreSQL, SQLite, MongoDB, MSSQL, Google Sheets, Airtable κ.α.	Καμία	MySQL, PostgreSQL, MongoDB, Microsoft SQL Server (MSSQL), Oracle Database κ.α.	MySQL, PostgreSQL, Microsoft SQL Server (MSSQL), Oracle Database, MongoDB, Redis, Amazon Redshift, Snowflake, Google Sheets, Airtable κ.α.
Παραμετροποίηση JavaScript	Όχι	Ναι	Ναι	Ναι	Ναι
Custom domain	Όχι	Όχι	Ναι	Ναι	Ναι

Πίνακας 1. Σύγκριση κύριων χαρακτηριστικών των framework στο δωρεάν πακέτο.

Η χρήση του παραπάνω πίνακα μας βοήθησε σημαντικά, καθώς μας επέτρεψε να δούμε με πιο συνοπτικό τρόπο τα θετικά και τα αρνητικά χαρακτηριστικά κάθε framework, σε σχέση με τις ανάγκες μας και να περιορίσουμε τις επιλογές μας.

3.4 Τελική επιλογή του framework

Το τελευταίο βήμα πριν την τελική μας επιλογή ήταν να «στήσουμε», χρησιμοποιώντας το Docker, ορισμένα από τα frameworks, προκειμένου να αποκτήσουμε μια πιο ολοκληρωμένη εικόνα του περιβάλλοντος ανάπτυξης, της ευκολίας χρήσης και της φιλοσοφίας κάθε εργαλείου. Δημιουργήσαμε μια βασική εφαρμογή, συνδέοντας μια βάση δεδομένων, με σκοπό να απεικονίσουμε τα δεδομένα και να εξετάσουμε τις δυνατότητες που προσφέρονται για τη διαχείρισή τους. Επιπλέον, παρατηρήσαμε στην πράξη τις δυνατότητες παραμετροποίησης μέσω κώδικα και αναλύσαμε τον τρόπο με τον οποίο επιτυγχάνεται η ενσωμάτωση των διαφόρων στοιχείων στην εφαρμογή.

Από τις διάφορες πλατφόρμες που εξετάσαμε, αποφασίσαμε να δοκιμάσουμε το Appsmith και το Budibase, καθώς τα υπόλοιπα framework είχαν κάποια χαρακτηριστικά που εξαρχής μας απομάκρυναν από το να τα επιλέξουμε και τα οποία αναφέρθηκαν αναλυτικά στις αντίστοιχες υποενότητες της εργασίας. Στη συνέχεια, και μετά από τη διαδικασία δοκιμής των δύο αυτών frameworks, καταλήξαμε τελικά στην επιλογή του Appsmith.

Παρόλο που το Budibase προσφέρει πλήθος λειτουργιών και μοιράζεται κάποια κοινά χαρακτηριστικά με το Appsmith, όπως οι πολλές δυνατότητες ενσωμάτωσης με εξωτερικές βάσεις δεδομένων, η έμπρακτη χρήση του μας έδειξε ότι η φιλοσοφία του δεν ταίριαζε πλήρως με τις ανάγκες του έργου μας. Από τον γενικότερο σχεδιασμό του εργαλείου, διαπιστώσαμε ότι είναι περισσότερο προσανατολισμένο σε ομάδες με λιγότερη τεχνογνωσία. Αν και προσφέρει δυνατότητες παραμετροποίησης, αυτές είναι πιο περιορισμένες σε σύγκριση με το Appsmith, κάτι που για εμάς αποτέλεσε έναν καθοριστικό παράγοντα στην απόφασή μας. Επιπλέον, η δωρεάν έκδοση του Budibase επιτρέπει μόνο έως 20 χρήστες σε self-hosted περιβάλλοντα, γεγονός που αποτελούσε περιορισμό για την ανάπτυξη του εργαλείου διαχείρισης έργων.

Αντίθετα, το Appsmith είχε ήδη κερδίσει την εκτίμησή μας από τη φάση μελέτης των διάφορων επιλογών. Με τις δυνατότητες του για παραμετροποίηση μέσω κώδικα αλλά και την ευχρηστία του με το γραφικό του περιβάλλον ήταν μία από τις πρώτες μας επιλογές. Κατά την πρακτική δοκιμή, το Appsmith απέδειξε ότι ανταποκρίνεται πλήρως στις προσδοκίες μας, προσφέροντας μια εξαιρετική ισορροπία μεταξύ ευκολίας χρήσης και ευχέρειας παραμετροποίησης. Ένα χαρακτηριστικό που ενισχύει τον προγραμματιστικό προσανατολισμό του είναι το ενσωματωμένο IDE για την JavaScript. Αυτό επιτρέπει στους χρήστες να δημιουργούν και να ενσωματώνουν αρχεία JavaScript απευθείας σε κάθε σελίδα της εφαρμογής, κάτι που είναι ιδιαίτερα οικείο και βολικό για προγραμματιστές. Από την έρευνα μας κανένα άλλο εργαλείο δεν προσέφερε τέτοιο επίπεδο παραμετροποίησης σε ισορροπία με τις ευκολίες του low-code περιβάλλοντος.

Κεφάλαιο 4 - Academic Project Management (APM): Ένα νέο εργαλείο για διαχείριση ακαδημαϊκών ομάδων

4.1 Ακαδημαϊκές ομάδες και διαχείριση project

Με τον όρο ακαδημαϊκές ομάδες αναφερόμαστε συνήθως σε ομάδες φοιτητών, ερευνητών, καθηγητών και άλλων ατόμων που συνεργάζονται για την επίτευξη ενός κοινού ακαδημαϊκού στόχου, ο οποίος μπορεί να περιλαμβάνει την εκπόνηση ερευνητικών εργασιών, την ανάπτυξη νέων θεωρητικών προσεγγίσεων, την ολοκλήρωση διπλωματικών και διδακτορικών εργασιών ή την υλοποίηση άλλων ακαδημαϊκών project. Οι ακαδημαϊκές ομάδες συχνά διαχειρίζονται πολύπλοκα έργα που απαιτούν συντονισμό, συνεργασία και την ορθή παρακολούθηση της προόδου των δραστηριοτήτων τους.

Η φύση αυτών των έργων διαφοροποιείται σημαντικά από άλλες επιχειρηματικές ή οργανωτικές δραστηριότητες καθώς οι ανάγκες και οι προτεραιότητες των μελών της ομάδας σχετίζονται άμεσα με την ακαδημαϊκή έρευνα, τη μάθηση και την προσωπική ανάπτυξη. Συχνά, οι ακαδημαϊκές ομάδες καλούνται να αντιμετωπίσουν προβλήματα διαχείρισης χρόνου, προθεσμιών, συνεργασίας μεταξύ πολλών μελών, και συντονισμού μεταξύ διαφορετικών επιστημονικών πεδίων. Επιπλέον ο τρόπος εργασίας τους έχει μία πιο ελεύθερη μορφή σε σύγκριση με εταιρικούς οργανισμούς καθιστώντας επιτακτική την ανάγκη για χρήση ενός εργαλείου που οργανώνει με απλότητα τις διάφορες ομάδες και τα έργα τους.

Κάποια από τα έργα που καλούνται να διαχειριστούν οι ακαδημαϊκές ομάδες είναι:

- **Διπλωματικές και Διδακτορικές Εργασίες:** Ερευνητικά έργα που αναλαμβάνουν φοιτητές ή υποψήφιοι διδάκτορες με την καθοδήγηση καθηγητών ή ερευνητικών ομάδων.
- **Ερευνητικά Projects:** Συνεχιζόμενη έρευνα σε ένα συγκεκριμένο επιστημονικό ή τεχνολογικό πεδίο, όπου η ομάδα συνήθως συνεργάζεται για την ανάπτυξη νέων θεωρητικών μοντέλων, λύσεων ή πρωτότυπων εφαρμογών.
- **Προπτυχιακές Εργασίες:** Εργασίες που αφορούν προπτυχιακά μαθήματα του εκάστοτε τμήματος όπως σειρές ασκήσεων, προγραμματιστικές ασκήσεις κα.

Για τη σωστή διαχείριση των έργων αυτών, απαιτείται:

1. **Οργάνωση και Προγραμματισμός:** Η δημιουργία χρονοδιαγραμμάτων και η σωστή κατανομή των καθηκόντων σε όλα τα μέλη της ομάδας είναι καίριας σημασίας για την αποτελεσματική εκτέλεση των διαφόρων έργων.

2. **Επικοινωνία και Συνεργασία:** Η συνεχής και ξεκάθαρη επικοινωνία μεταξύ των μελών της ομάδας είναι κρίσιμη για την αποφυγή λαθών και για την προώθηση της συνεργασίας και της ανταλλαγής ιδεών.
3. **Παρακολούθηση της Προόδου:** Η συνεχής αξιολόγηση της προόδου των εργασιών και του έργου συνολικά βοηθά στην έγκαιρη αναγνώριση προβλημάτων και καθυστερήσεων, ενώ παράλληλα επιτρέπει τη λήψη διορθωτικών μέτρων όταν χρειαστεί.
4. **Ανάθεση Ρόλων και Ευθυνών:** Κάθε μέλος της ομάδας θα πρέπει να έχει σαφείς ρόλους και ευθύνες, οι οποίοι να ανταποκρίνονται στις ικανότητες και τις γνώσεις του, ώστε το έργο να προχωρά με τη μεγαλύτερη δυνατή αποδοτικότητα.

Η σωστή διαχείριση αυτών των στοιχείων απαιτεί τη χρήση ενός κατάλληλου εργαλείου διαχείρισης έργων, το οποίο θα βοηθήσει στην οργάνωση, την παρακολούθηση της προόδου και την επικοινωνία μεταξύ των μελών της ομάδας. Εδώ εντοπίζεται η ανάγκη για την ανάπτυξη ειδικών εργαλείων που να ανταποκρίνονται στις ανάγκες των ακαδημαϊκών ομάδων, διευκολύνοντας την αποτελεσματική διαχείριση των έργων και την ολοκλήρωσή τους με επιτυχία.

Το ιδανικό εργαλείο για την υποστήριξη αυτών των απαιτήσεων θα πρέπει να παρέχει μια ολοκληρωμένη λύση που θα επιτρέπει με απλό τρόπο την ενσωμάτωση πολλαπλών λειτουργιών, όπως η διαχείριση χρονοδιαγραμμάτων, η παρακολούθηση προόδου, η κατανομή εργασιών και η γενικότερη οργάνωση πληροφοριών.

4.2 Απαιτήσεις του APM

Στην παρούσα υποενότητα θα δούμε αναλυτικά τις απαιτούμενες βασικές λειτουργίες που υλοποιήσαμε στο εργαλείο μας με την μορφή use cases.

Use Case ID		APM-01
Title	Είσοδος Χρήστη	
Actors	Χρήστες με ρόλο <i>Admin, User, Guest</i>	
Preconditions	Ο χρήστης δεν είναι συνδεδεμένος στην εφαρμογή	
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης μεταφερθεί στην σελίδα της εφαρμογής 2. Ο χρήστης συμπληρώνει το email του και τον κωδικό του 3. Ο χρήστης πατάει το κουμπί <i>Sign In</i> 4. Εάν ένα από τα στοιχεία του χρήστη είναι λανθασμένα <ol style="list-style-type: none"> 4.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 4.2 Το use case τερματίζει 5. Το σύστημα τον μεταφέρει στην αρχική σελίδα <i>Dashboard</i> της εφαρμογής 	

Use Case ID		APM-02
Title	Αποσύνδεση Χρήστη	
Actors	Χρήστες με ρόλο <i>Admin, User, Guest</i>	
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή	
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Log-out</i> 2. Το jwt token του χρήστη διαγράφεται 3. Ο χρήστης μεταφέρεται στην σελίδα <i>Log-in</i> της εφαρμογής 	

Use Case ID		APM-03
Title	Εγγραφή Χρήστη	
Actors	Χρήστης με ρόλο <i>Admin</i>	
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και έχει μεταφερθεί στην σελίδα <i>Manage Users</i>	
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Add User</i> 2. Ο χρήστης συμπληρώνει τα απαραίτητα πεδία στην εμφανιζόμενη φόρμα (<i>user name, email, password και user category</i>) 3. Ο χρήστης πατάει το κουμπί <i>Save</i> 4. Η φόρμα κλείνει και το σύστημα ενημερώνει για την επιτυχημένη προσθήκη του νέου χρήστη. 5. Το σύστημα στέλνει ενημερωτικό email στον νέο χρήστη. 	
Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Πατώντας το κουμπί <i>cancel</i> 1.2 Κάνοντας κλικ εξωτερικά της φόρμας 1.3 Πατώντας το κουμπί 'X' 	
Postconditions	Ένας νέος χρήστης έχει προστεθεί στην βάση δεδομένων και εμφανίζεται στον πίνακα της σελίδας	

Use Case ID		APM-04
Title	Διαγραφή Χρήστη	
Actors	Χρήστης με ρόλο <i>Admin</i>	
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και έχει μεταφερθεί στην σελίδα <i>Manage Users</i>	
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Delete</i> για κάποια γραμμή του πίνακα 2. Ο χρήστης πατάει το κουμπί <i>Delete</i> στο εμφανιζόμενο popup. 3. Εάν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 3.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 3.2 Το use case τερματίζει 	

	<ol style="list-style-type: none"> 4. Το σύστημα κλείνει το popup και ενημερώνει για την επιτυχημένη διαγραφή ενός χρήστη 5. Το σύστημα στέλνει ενημερωτικό email στον χρήστη που διαγράφηκε
Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Πατώντας το κουμπί <i>cancel</i> 1.2 Κάνοντας κλικ εξωτερικά της φόρμας 1.3 Πατώντας το κουμπί 'X'
Postconditions	Ο χρήστης έχει διαγραφεί από την βάση δεδομένων και έχει αφαιρεθεί από τον πίνακα των χρηστών της σελίδας

Use Case ID	APM-05
Title	Ανανέωση Στοιχείων Χρήστη Από Admin
Actors	Χρήστης με ρόλο <i>Admin</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και έχει μεταφερθεί στην σελίδα <i>Manage Users</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Edit</i> για κάποιο πεδίο ενός χρήστη στον πίνακα 2. Ο χρήστης αλλάζει τιμή στο επιθυμητό πεδίο και κάνει κλικ έξω από τον πίνακα 3. Εάν προύψει κάποιο σφάλμα <ol style="list-style-type: none"> 3.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 3.2 Το use case τερματίζει 4. Αλλιώς το σύστημα ενημερώνει για την επιτυχημένη ανανέωση του πεδίου
Postconditions	Τα στοιχεία του χρήστη έχουν ανανεωθεί στην βάση δεδομένων καθώς και στον πίνακα της σελίδας

Use Case ID	APM-06
Title	Ανανέωση Στοιχείων Χρήστη
Actors	Χρήστες με ρόλο <i>Admin, User, Guest</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και έχει μεταφερθεί στην σελίδα <i>My Preferences</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης αλλάξει κάποιο από τα στοιχεία του στο <i>My Info</i> section της σελίδας 2. Ο χρήστης αλλάζει τα επιθυμητά πεδία (<i>user name, email, password</i>) 3. Ο χρήστης πατάει το κουμπί <i>Update</i> 4. Εάν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 4.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 4.2 Το use case τερματίζει 5. Αλλιώς το σύστημα εμφανίζει ενημερωτικό μήνυμα επιτυχίας

Postconditions	Τα στοιχεία του χρήστη έχουν ανανεωθεί στην βάση δεδομένων
-----------------------	--

Use Case ID	APM-07
Title	Προσθήκη Project
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί '+' στον πίνακα <i>Your Projects</i> 2. Ο χρήστης συμπληρώνει την απαραίτητη πληροφορία στην φόρμα που εμφανίζεται (<i>Project title, Description, Start date, End date</i> etc.) 3. Ο χρήστης πατάει το κουμπί <i>Save</i> 4. Εάν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 1.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 1.2 Το use case τερματίζει 5. Αλλιώς το σύστημα κλείνει το Popup και ενημερώνει τον χρήστη για την επιτυχημένη δημιουργία του project
Alternative flow	<ol style="list-style-type: none"> 1. Ο χρήστης αφήνει κενό το όνομα του project 2. Ο χρήστης πατάει το κουμπί <i>Save</i> 3. Το σύστημα εμφανίζει μήνυμα σφάλματος
Postconditions	Το νέο project προστίθεται στην βάση δεδομένων και ανανεώνονται τα περιεχόμενα του πίνακα <i>Your Projects</i>

Use Case ID	APM-08
Title	Εγγραφή Σε Project
Actors	Χρήστες με ρόλο <i>Admin, User, Guest</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί '+' στον πίνακα <i>Your Projects</i> 2. Ο χρήστης κάνει κλικ στο dropdown μενού και επιλέγει το project που θέλει να εγγραφεί 3. Ο χρήστης πατάει το κουμπί <i>Save</i> 4. Εάν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 4.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 4.2 Το use case τερματίζει 5. Αλλιώς το σύστημα κλείνει το popup και ενημερώνει τον χρήστη για την επιτυχημένη εγγραφή του στο project

Postconditions	Το νέο project προστίθεται στην βάση δεδομένων στα project του χρήστη και ανανεώνονται τα περιεχόμενα των σχετικών πίνακων
-----------------------	--

Use Case ID		APM-09
Title	Προσθήκη Task	
Actors	Χρήστες με ρόλο <i>Admin, User</i>	
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Dashboard</i> ή <i>Project Dashboard</i>	
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί '+' στον σχετικό πίνακα tasks της σελίδας 2. Ο χρήστης συμπληρώνει τα απαραίτητα πεδία στην εμφανιζόμενη φόρμα <ol style="list-style-type: none"> 2.1 Εάν ο χρήστης βρίσκεται στην σελίδα <i>Project Dashboard</i> τότε το πεδίο που αφορά το σε ποιό project θα ανήκει το καινούγιο task είναι συμπληρωμένο με το αντίστοιχο project της σελίδας 3. Ο χρήστης πατάει το κουμπί <i>Save</i> 4. Το σύστημα κλείνει το popup και ενημερώνει τον χρήστη για την επιτυχημένη δημιουργία του task 	
Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Πατώντας το κουμπί <i>cancel</i> 1.2 Κάνοντας κλικ εξωτερικά της φόρμας 1.3 Πατώντας το κουμπί 'X' 	
Alternative flow 2	<ol style="list-style-type: none"> 1. Ο χρήστης αφήνει το όνομα του task ή το πεδίο του σχετιζόμενου project κενό 2. Πατάει το κουμπί <i>Save</i> 3. Το σύστημα εμφανίζει σχετικό μήνυμα σφάλματος στην κάθε περίπτωση 	
Postconditions	Το νέο task προστίθεται στην βάση δεδομένων και ανανεώνονται τα περιεχόμενα των σχετικών στοιχείων της σελίδας (Πίνακες, ημερολόγιο κτλ.)	

Use Case ID		APM-10
Title	Προσθήκη Task Απο Kanban	
Actors	Χρήστες με ρόλο <i>Admin, User</i>	
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard</i>	
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί '+' στο <i>Kanban widget</i> 2. Το σύστημα εμφανίζει μία φόρμα με συμπληρωμένα τα πεδία που αφορούν το project και την κατάσταση του task 	

	<ol style="list-style-type: none"> 3. Ο χρήστης συμπληρώνει τα υπόλοιπα απαραίτητα πεδία στην φόρμα 4. Ο χρήστης πατάει το κουμπί <i>Save</i> 5. Το σύστημα κλείνει το popup και ενημερώνει τον χρήστη για την επιτυχημένη δημιουργία του task
Alternative flow 1	Ίδιο με το use case APM-09
Alternative flow 2	<ol style="list-style-type: none"> 1. Ο χρήστης αφήνει το όνομα του task κενό 2. Πατάει το κουμπί <i>Save</i> 3. Το σύστημα εμφανίζει μήνυμα σφάλματος πως το όνομα δεν μπορεί να είναι κενό
Postconditions	Το νέο task προστίθεται στην βάση δεδομένων και ανανεώνονται τα περιεχόμενα των σχετικών στοιχείων της σελίδας (Πίνακες, ημερολόγιο, Kanban)

Use Case ID	APM-11
Title	Προσθήκη Meeting
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Dashboard</i> ή <i>Project Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί '+' στον σχετικό πίνακα meeting της σελίδας 2. Ο χρήστης συμπληρώνει τα απαραίτητα πεδία στην εμφανιζόμενη φόρμα <ol style="list-style-type: none"> 2.1 Εάν ο χρήστης βρίσκεται στην σελίδα <i>Project Dashboard</i> τότε το πεδίο που αφορά το σε ποιο project θα ανήκει το καινούγιο meeting είναι συμπληρωμένο με το αντίστοιχο project της σελίδας 3. Ο χρήστης πατάει το κουμπί <i>Save</i> 4. Το σύστημα κλείνει το popup και ενημερώνει τον χρήστη για την επιτυχημένη δημιουργία του meeting
Alternative flow 1	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Πατώντας το κουμπί <i>cancel</i> 1.2 Κάνοντας κλικ εξωτερικά της φόρμας 1.3 Πατώντας το κουμπί 'X'
Alternative flow 2	<ol style="list-style-type: none"> 1. Ο χρήστης αφήνει το όνομα του meeting κενό 2. Πατάει το κουμπί <i>Save</i> 3. Το σύστημα εμφανίζει μήνυμα σφάλματος πως το όνομα του <i>meeting</i> δεν μπορεί να είναι κενό
Postconditions	<ol style="list-style-type: none"> 1. Το νέο meeting προστίθεται στην βάση δεδομένων 2. Ο χρήστης έχει προστεθεί ως participant του meeting με ανανέωση του σχετικού πίνακα στην βάση 3. Ανανεώνονται τα περιεχόμενα των σχετικών στοιχείων της σελίδας (Πίνακες, ημερολόγιο)

Use Case ID	APM-11
Title	Προσθήκη Meeting Απο Ημερολόγιο
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή, βρίσκεται στην σελίδα <i>Dashboard</i> ή <i>Project Dashboard</i> και έχει πατήσει πάνω στον κενό χώρο του ημερολογίου
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει την επιλογή <i>Create new meeting</i> απο το εμφανιζόμενο μενού 2. Το σύστημα εμφανίζει μία φόρμα με συμπληρωμένη την ημερομηνία διεξαγωγής του meeting 3. Ο χρήστης συμπληρώνει τα υπόλοιπα απαραίτητα πεδία στην φόρμα 4. Ο χρήστης πατάει το κουμπί <i>Save</i> 5. Το σύστημα κλείνει το popup και ενημερώνει τον χρήστη για την επιτυχημένη δημιουργία του meeting
Alternative flow	Ίδια με το use case APM-11
Postconditions	Ίδια με το use case APM-11

Use Case ID	APM-12
Title	Ανανέωση Πεδίων Οντότητας Απο Ημερολόγιο
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Dashboard</i> ή <i>Project Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει κάποιο απο τα events του ημερολογίου <ol style="list-style-type: none"> 1.1 Εαν πατήσει κάποιο meeting ανοίγει η φόρμα για την ανανέωση των στοιχείων του meeting 1.2 Εαν πατήσει κάποιο task ανοίγει η φόρμα για ανανέωση των στοιχείων του task 2. Ο χρήστης αλλάζει τα επιθυμητά πεδία 3. Ο χρήστης πατάει το κουμπί <i>Update</i> 4. Εάν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 4.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 4.2 Το use case τερματίζει 5. Αλλιώς το σύστημα κλείνει την φόρμα και ενημερώνει τον χρήστη για την επιτυχημένη ανανέωση της οντότητας
Alternative flow 1	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Πατώντας το κουμπί <i>cancel</i> 1.2 Κάνοντας κλικ εξωτερικά της φόρμας

	1.3 Πατώντας το κουμπί 'X'
Postconditions	Τα στοιχεία της οντότητας έχουν ανανεωθεί στην βάση δεδομένων και έχουν ανανεωθεί τα σχετικά στοιχεία της σελίδας (Πίνακες, ημερολόγιο, κτλ.)

Use Case ID	APM-13
Title	Ανανέωση Πεδίων Project
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης αλλάξει κάποιο πεδίο του project 2. Ο χρήστης αλλάζει όλα τα επιθυμητά πεδία 3. Ο χρήστης πατάει το κουμπί <i>Save</i> 4. Το σύστημα ενημερώνει τον χρήστη για την επιτυχημένη ανανέωση των πεδίων
Postconditions	Τα στοιχεία του Project έχουν ανανεωθεί στην βάση δεδομένων

Use Case ID	APM-14
Title	Αλλαγή Ονόματος Και Χρώματος Project
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Edit project name or colour</i> 2. Ο χρήστης κάνει τις επιθυμητές αλλαγές στο όνομα του project στο εμφανιζόμενο popup 3. Ο χρήστης αλλάζει το χρώμα του project εάν το επιθυμεί 4. Ο χρήστης πατάει το κουμπί <i>Save</i> 5. Εάν υπάρξει σφάλμα <ol style="list-style-type: none"> 5.1 Το σύστημα ενημερώνει τον χρήστη με σχετικό μήνυμα 5.2 Το use case τερματίζει 6. Αλλιώς το σύστημα κλείνει το popup και ενημερώνει τον χρήστη για την επιτυχημένη ανανέωση του ονόματος και του χρώματος του project
Alternative flow 1	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Πατώντας το κουμπί <i>cancel</i> 1.2 Κάνοντας κλικ εξωτερικά της φόρμας 1.3 Πατώντας το κουμπί 'X'

Postconditions	Το όνομα και το χρώμα του Project έχουν ανανεωθεί στην βάση δεδομένων
-----------------------	---

Use Case ID	APM-15
Title	Προσθήκη Χρηστών Σε Project
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Manage Users</i> 2. Ο χρήστης επιλέγει τους χρήστες που θέλει να προσθέσει στο project από το εμφανιζόμενο popup 3. Ο χρήστης πατάει το κουμπί <i>Add Users</i> 4. Το σύστημα κλείνει το popup και εμφανίζει ενημερωτικό μήνυμα επιτυχίας
Alternative flow 1	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Κάνοντας κλικ εξωτερικά της φόρμας 1.2 Πατώντας το κουμπί 'X'
Alternative flow 2	<ol style="list-style-type: none"> 1. Ο χρήστης δεν επιλέγει κανένα χρήστη για προσθήκη 2. Ο χρήστης πατάει το κουμπί <i>Add Users</i> 3. Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 4. Το use case τερματίζει
Postconditions	Οι επιλεγμένοι χρήστες έχουν προστεθεί στον πίνακα συσχέτισης των user με τα project της βάσης δεδομένων

Use Case ID	APM-16
Title	Διαγραφή Χρήστη Από Project
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Manage Users</i> 2. Ο χρήστης πατάει το κουμπί <i>Delete</i> στον χρήστη που επιθυμεί να διαγράψει 3. Εάν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 3.1 Το σύστημα εμφανίζει σχετικό μήνυμα σφάλματος 3.2 Το use case τερματίζει 4. Αλλιώς το σύστημα εμφανίζει ενημερωτικό μήνυμα επιτυχίας
Alternative flow 1	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Κάνοντας κλικ εξωτερικά της φόρμας

	1.2 Πατώντας το κουμπί 'X'
Postconditions	Ο χρήστης έχει αφαιρεθεί από τον πίνακα συσχέτισης των user με τα project της βάσης δεδομένων

Use Case ID	APM-17
Title	Απεγραφή Χρήστη Από Project
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Unsubscribe</i> 2. Ο χρήστης πατάει το κουμπί <i>Unsubscribe</i> στο εμφανιζόμενο popup 3. Το σύστημα μεταφέρει τον χρήστη στην σελίδα <i>Dashboard</i>
Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Κάνοντας κλικ εξωτερικά της φόρμας 1.2 Πατώντας το κουμπί <i>Cancel</i> 1.3 Πατώντας το κουμπί 'X'
Postconditions	Ο χρήστης έχει αφαιρεθεί από τον πίνακα συσχέτισης των user με τα project της βάσης δεδομένων

Use Case ID	APM-18
Title	Διαγραφή Οντότητας
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard, Meeting ή Task</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Delete</i> 2. Ο χρήστης πατάει το κουμπί <i>Delete</i> στο εμφανιζόμενο popup 3. Σε περίπτωση σφάλματος <ol style="list-style-type: none"> 3.1 Το σύστημα εμφανίζει σχετικό ενημερωτικό μήνυμα 3.2 Το use case τερματίζει 4. Αλλιώς το σύστημα μεταφέρει τον χρήστη <ol style="list-style-type: none"> 4.1 Στην σελίδα <i>Project dashboard</i> του σχετιζόμενου project αν η διαγραφή αφορά την οντότητα task 4.2 Στην σελίδα <i>Dashboard</i> αν η διαγραφή αφορά τις οντότητες project ή meeting
Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Κάνοντας κλικ εξωτερικά της φόρμας 1.2 Πατώντας το κουμπί <i>Cancel</i> 1.3 Πατώντας το κουμπί 'X'

Postconditions	Η οντότητα έχει διαγραφεί από τον σχετικό πίνακα της βάσης δεδομένων
-----------------------	--

Use Case ID	APM-19
Title	Ανέβασμα Αρχείου Σε Οντότητα
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard, Meeting</i> ή <i>Task</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Select files to upload</i> 2. Το σύστημα ανοίγει το popup επιλογής αρχείων 3. Ο χρήστης ανεβάζει τα επιθυμητά αρχεία με drag-and-drop ή επιλέγοντας τα από τον υπολογιστή του έπειτα από αναζήτηση τους 4. Ο χρήστης πατάει το κουμπί <i>Upload files</i> 5. Σε περίπτωση σφάλματος <ol style="list-style-type: none"> 5.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 5.2 Το use case συνεχίζει από το βήμα 2 6. Αλλιώς το σύστημα κλείνει το popup και ενημερώνει τον χρήστη για την επιτυχία ανεβάσματος των αρχείων με σχετικό μήνυμα
Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Κάνοντας κλικ εξωτερικά της φόρμας 1.2 Πατώντας το κουμπί <i>Cancel</i> 1.3 Πατώντας το κουμπί 'X'
Postconditions	<ol style="list-style-type: none"> 1. Τα αρχεία προστίθενται στον σχετικό πίνακα της βάσης δεδομένων 2. Τα αρχεία προτίθενται στον file server

Use Case ID	APM-20
Title	Ανέβασμα Google Doc Link Σε Οντότητα
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Project Dashboard</i> ή <i>Task</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Upload Doc Link</i> 2. Το σύστημα εμφανίζει σε μορφή popup μια φόρμα με σχετικά στοιχεία 3. Ο χρήστης συμπληρώνει τα επιθυμητά πεδία <ol style="list-style-type: none"> 3.1 Εάν ο χρήστης αφήσει κενό το πεδίο του ονόματος το κουμπί <i>Save</i> είναι απενεργοποιημένο 4. Ο χρήστης πατάει το κουμπί <i>Save</i> 5. Το σύστημα κλείνει το popup και ενημερώνει για το επιτυχημένο ανέβασμα του link

Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Κάνοντας κλικ εξωτερικά της φόρμας 1.2 Πατώντας το κουμπί <i>Cancel</i> 1.3 Πατώντας το κουμπί 'X'
Postconditions	Τα λινκ προστίθενται στον σχετικό πίνακα της βάσης δεδομένων

Use Case ID	APM-21
Title	Αλλαγή Ονόματος Task
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Task</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Edit task name</i> 2. Ο χρήστης κάνει την επιθυμητή αλλαγή στο όνομα του task στο εμφανιζόμενο popup 3. Ο χρήστης πατάει το κουμπί <i>Update</i> 4. Αν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 4.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος 4.2 Το use case συνεχίζει από το βήμα 2 5. Αλλιώς το σύστημα κλείνει το Popup και εμφανίζει ενημερωτικό μήνυμα επιτυχίας
Alternative flow	<ol style="list-style-type: none"> 1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία: <ol style="list-style-type: none"> 1.1 Κάνοντας κλικ εξωτερικά της φόρμας 1.2 Πατώντας το κουμπί <i>Cancel</i> 1.3 Πατώντας το κουμπί 'X'
Postconditions	Το όνομα του task έχει ανανεωθεί στον σχετικό πίνακα της βάσης δεδομένων.

Use Case ID	APM-22
Title	Αλλαγή Ονόματος Meeting
Actors	Χρήστες με ρόλο <i>Admin, User</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Meeting</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης πατήσει το κουμπί <i>Edit meeting name</i> 2. Ο χρήστης κάνει την επιθυμητή αλλαγή στο όνομα του meeting στο εμφανιζόμενο popup 3. Ο χρήστης πατάει το κουμπί <i>Update</i> 4. Αν προκύψει κάποιο σφάλμα <ol style="list-style-type: none"> 4.1 Το σύστημα εμφανίζει ενημερωτικό μήνυμα σφάλματος

	<p>4.2 Το use case συνεχίζει από το βήμα 2</p> <p>5. Αλλιώς το σύστημα κλείνει το Popur και εμφανίζει ενημερωτικό μήνυμα επιτυχίας</p>
Alternative flow	<p>1. Οποιαδήποτε στιγμή ο χρήστης μπορεί να ακυρώσει την διαδικασία:</p> <p>1.1 Κάνοντας κλικ εξωτερικά της φόρμας</p> <p>1.2 Πατώντας το κουμπί <i>Cancel</i></p> <p>1.3 Πατώντας το κουμπί 'X'</p>
Postconditions	Το όνομα του meeting έχει ανανεωθεί στον σχετικό πίνακα της βάσης δεδομένων

Use Case ID	APM-23
Title	Διαμοιρασμός Google Document
Actors	Χρήστες με ρόλο <i>Admin, User, Guest</i>
Preconditions	Ο χρήστης είναι συνδεδεμένος στην εφαρμογή και βρίσκεται στην σελίδα <i>Meeting</i>
Main flow	<ol style="list-style-type: none"> 1. Το use case ξεκινάει όταν ο χρήστης προσθέσει κάποιο Google Document link στο πεδίο <i>Google Doc Link</i> 2. Το σύστημα ενημερώνει τον χρήστη για την επιτυχημένη ανανέωση του google document link 3. Το σύστημα εμφανίζει το google document με το περιεχόμενο του και δυνατότητες επεξεργασίας στο σχετικό widget της σελίδας
Postconditions	<ol style="list-style-type: none"> 1. Το σχετικό πεδίο στην βάση δεδομένων έχει ανανεωθεί με το λίνκ του Google document 2. Το στοιχείο iframe της σελίδας εμφανίζει πλέον το Google document

Τα παραπάνω use cases αποτελούν μόνο μια ενδεικτική παρουσίαση των βασικών λειτουργιών που επιδιώξαμε να ενσωματώσουμε στην εφαρμογή μας. Παρόλο που καλύπτουν τις πιο κρίσιμες και καθοριστικές απαιτήσεις που καθορίσαμε για την ανάπτυξη της, δεν είναι σε καμία περίπτωση εξαντλητικά. Η εφαρμογή μας περιλαμβάνει πλήθος άλλων λειτουργιών και χαρακτηριστικών, τα οποία επιλέξαμε να υλοποιήσουμε και προέκυψαν κυρίως κατά την ανάπτυξη της για να εξασφαλίσουμε μια πιο ολοκληρωμένη και ευέλικτη εμπειρία για τον τελικό χρήστη.

4.3 Λειτουργίες του APM

Με την ανάπτυξη του εργαλείου διαχείρισης έργων σκοπός μας ήταν η κάλυψη των αναγκών των ακαδημαϊκών ομάδων με τρόπο απλό, εύχρηστο και πρακτικό χωρίς υπερβολές σε χαρακτηριστικά που θα οδηγούσαν σε μεγάλα χρονικά διαστήματα εκπαίδευσης πάνω στο εργαλείο παρεκκλίνοντας τις ομάδες από τον κύριο στόχο τους που είναι η επιτυχημένη ολοκλήρωση των εργασιών τους.

Θεωρούμε πως στην πρώτη του έκδοση, το εργαλείο μας παρέχει όλες τις βασικές λειτουργίες που μπορεί να χρειαστεί μια ακαδημαϊκή ομάδα για να ξεκινήσει να διαχειρίζεται τα έργα της με τρόπο οργανωμένο και αποτελεσματικό.

Η εφαρμογή έχει σχεδιαστεί για να παρέχει ένα αποδοτικό περιβάλλον διαχείρισης ακαδημαϊκών έργων, επιτρέποντας στους χρήστες να συμμετέχουν σε έργα και να διαχειρίζονται τις σχετικές εργασίες, τις συναντήσεις και τα αρχεία τους.

Κάθε χρήστης της εφαρμογής μπορεί να συμμετέχει σε ένα ή περισσότερα έργα (projects), τα οποία αποτελούν τον βασικό οργανωτικό άξονα της εφαρμογής. Στο κάθε έργο υπάρχει η δυνατότητα δημιουργίας και ανάθεσης εργασιών, οι οποίες μπορεί να έχουν προθεσμίες και χρονοδιαγράμματα για την καλύτερη παρακολούθηση της εξέλιξής τους. Παράλληλα, οι χρήστες μπορούν να προγραμματίζουν συναντήσεις, όπου θα συζητούν την πρόοδο του έργου και θα οργανώνουν τις επόμενες ενέργειες. Η δυνατότητα διαμοιρασμού αρχείων τόσο στις εργασίες όσο και στις συναντήσεις διευκολύνει τη συνεργασία, παρέχοντας ένα ενιαίο σημείο αναφοράς για όλα τα απαραίτητα έγγραφα και πληροφορίες που χρειάζονται για την διεκπεραίωση του έργου.

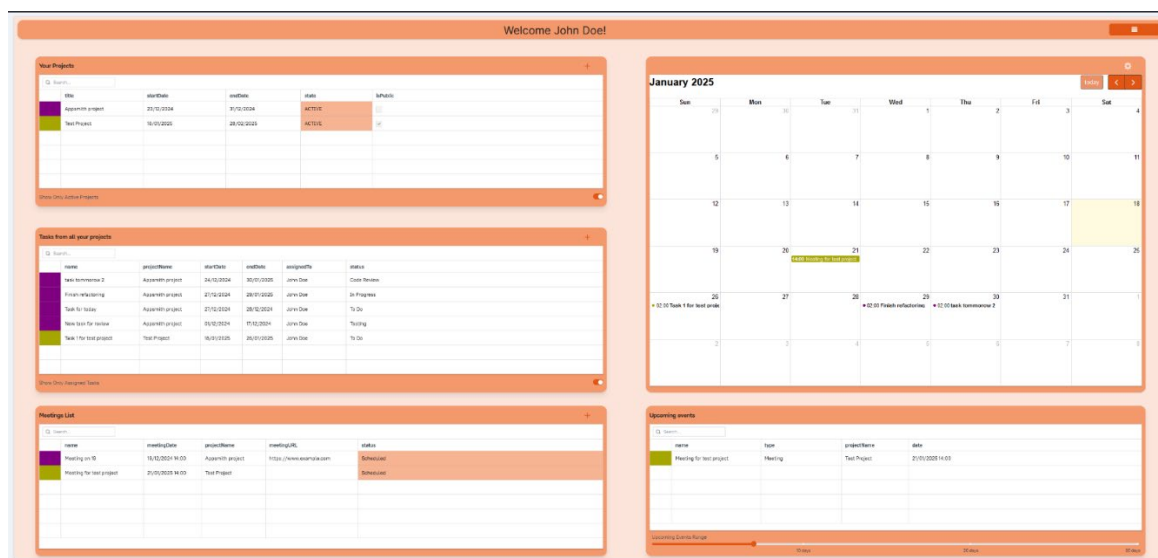
Έτσι για την σωστή οργάνωση των παραπάνω δεδομένων, η εφαρμογή παρέχει στους χρήστες ένα οργανωμένο περιβάλλον όπου μπορούν να βλέπουν μια σύνοψη όλων των εργασιών και των προθεσμιών ενός συγκεκριμένου έργου, αλλά και να έχουν πρόσβαση σε λεπτομερείς πληροφορίες για κάθε εργασία ή συνάντηση ξεχωριστά. Επιπλέον, μέσω του κεντρικού dashboard της εφαρμογής, προσφέρεται μια συνολική εικόνα των έργων στα οποία συμμετέχει ο χρήστης, επιτρέποντάς του να εκτελεί γρήγορα ενέργειες όπως δημιουργία εργασιών και συναντήσεων καθώς και να έχει άμεση πρόσβαση στις πιο σημαντικές πληροφορίες που αφορούν επικείμενα γεγονότα και καταληκτικές ημερομηνίες εργασιών και συναντήσεων.

Τέλος ένα ακόμα κρίσιμο στοιχείο της εφαρμογής είναι η υποστήριξη ασύγχρονων ειδοποιήσεων, οι οποίες επιτρέπουν στους χρήστες να ενημερώνονται άμεσα για γεγονότα που τους αφορούν όπως για παράδειγμα την ανάθεση τους σε μία εργασία ή την προσθήκη τους σε ένα νέο έργο.

Στις επόμενες υποενότητες, θα εξετάσουμε αναλυτικά τις δυνατότητες και τις λειτουργίες που προσφέρει το εργαλείο, παρουσιάζοντας κάθε σελίδα της εφαρμογής ξεχωριστά. Η ανάλυση αυτή στοχεύει να αναδείξει τόσο τις επιμέρους λειτουργίες όσο και τον τρόπο με τον οποίο αυτές συμβάλλουν στη συνολική εμπειρία χρήσης, προσφέροντας μια ολοκληρωμένη εικόνα για τις παρεχόμενες δυνατότητες.

4.3.1 Dashboard

Το Dashboard αποτελεί την αρχική σελίδα που αντικρίζει ο χρήστης με την είσοδό του στην εφαρμογή. Η σελίδα παρέχει μια συνοπτική και οργανωμένη απεικόνιση των έργων, των εργασιών και των συναντήσεων στα οποία συμμετέχει ο χρήστης όπως φαίνεται στην Εικόνα 8. Παράλληλα, δίνει τη δυνατότητα δημιουργίας νέων αντικειμένων ή μεταφοράς στις αντίστοιχες σελίδες για πιο αναλυτική διαχείριση και εκτέλεση λειτουργιών.



Εικόνα 8. Dashboard σελίδα του APM

Ακολουθεί αναλυτική περιγραφή των λειτουργιών που προσφέρει η σελίδα Dashboard:

- Πίνακας Project:** Μέσω του συγκεκριμένου πίνακα ο χρήστης μπορεί να δει όλα τα έργα που είναι εγγεγραμμένα, να μεταφερθεί στην σελίδα του κάνοντας κλικ στην αντίστοιχη γραμμή αλλά και να δημιουργήσει ή να εγγραφεί σε ένα καινούργιο έργο πατώντας το «+» εικονίδιο. Επίσης προσφέρονται δυνατότητες παραμετροποίησης της προβολής όπως ταξινόμηση με βάση οποιαδήποτε στήλη του πίνακα, αλλά και επιλογή μέσω του toggle στο κάτω μέρος για την προβολή ή μη των INACTIVE και COMPLETED project που μπορεί να είναι εγγεγραμμένος ο χρήστης. Τέλος το κάθε Project έχει το δικό του χρώμα, το οποίο κληρονομούν και όλες οι εργασίες (task) και συναντήσεις (meeting), με αποτέλεσμα την αυξημένη ευκολία στην παρατήρηση και εύρεση της απαραίτητης πληροφορίας στους διάφορους πίνακες.
- Πίνακας Task:** Οι λειτουργίες του πίνακα «Task» είναι παρόμοιες προσφέροντας δυνατότητες δημιουργίας μίας εργασίας, προβολή των εργασιών που ο χρήστης είναι εγγεγραμμένος ή όλων των εργασιών που είναι συνδεδεμένα σε κάποιο έργο που είναι εγγεγραμμένος, μεταφορά στην σελίδα της εργασίας και ταξινόμηση με βάση τις στήλες του πίνακα.

3. **Πίνακας Meeting:** μέσω του πίνακα «Meeting» ο χρήστης μπορεί να δει τα meeting στα οποία συμμετέχει, να μεταφερθεί στην σελίδα τους αλλά και να δημιουργήσει μία καινούργια συνάντηση. Να σημειωθεί πως στην περίπτωση των συναντήσεων υπάρχει η δυνατότητα να μην είναι συνδεδεμένες με κάποιο έργο, προσφέροντας ευελιξία στους χρήστες σε σχέση με τον γενικότερη επικοινωνία με την ομάδα τους.
4. **Ημερολόγιο:** Το ημερολόγιο παρέχει πληροφορίες για τις ημερομηνίες και την ώρα των συναντήσεων αλλά και τις καταληκτικές ημερομηνίες των διαφόρων εργασιών που έχουν ανατεθεί στον χρήστη. Τα χρωματισμένα κουτιά αναφέρονται στις συναντήσεις ενώ τα κουτιά με την χρωματιστή βούλα αναφέρονται στις εργασίες. Πατώντας οποιοδήποτε event ο χρήστης μπορεί να ανανεώσει τα διάφορα πεδία του αντικειμένου ενώ με απλό mouseover εμφανίζεται ένα popup για ευκολότερη ανάγνωση των λεπτομερειών του εκάστοτε event. Επιπλέον με απλό κλικ σε ένα κενό μέρος του ημερολογίου ο χρήστης μπορεί να προσθέσει μία καινούργια συνάντηση στην συγκεκριμένη ημερομηνία ή να αλλάξει την προβολή του ημερολογίου από μηνιαίο σε ημερήσιο. Τέλος προσφέρεται η δυνατότητα επιλογής, μέσω το εικονιδίου πάνω δεξιά, για το αν θα εμφανίζονται οι εργασίες και οι συναντήσεις στο ημερολόγιο προσφέροντας παραμετροποίηση ανάλογα τις επιθυμίες του χρήστη.
5. **Πίνακας Upcoming:** Στον πίνακα περιέχονται όλες οι συναντήσεις και οι εργασίες οι οποίες βρίσκονται κοντά στην πραγματοποίησή τους ή στην καταληκτική τους ημερομηνία. Το διάστημα αυτό μπορεί να καθοριστεί δυναμικά από τον slider που βρίσκεται στο κάτω μέρος του widget και κυμαίνεται από 1 έως 30 μέρες.

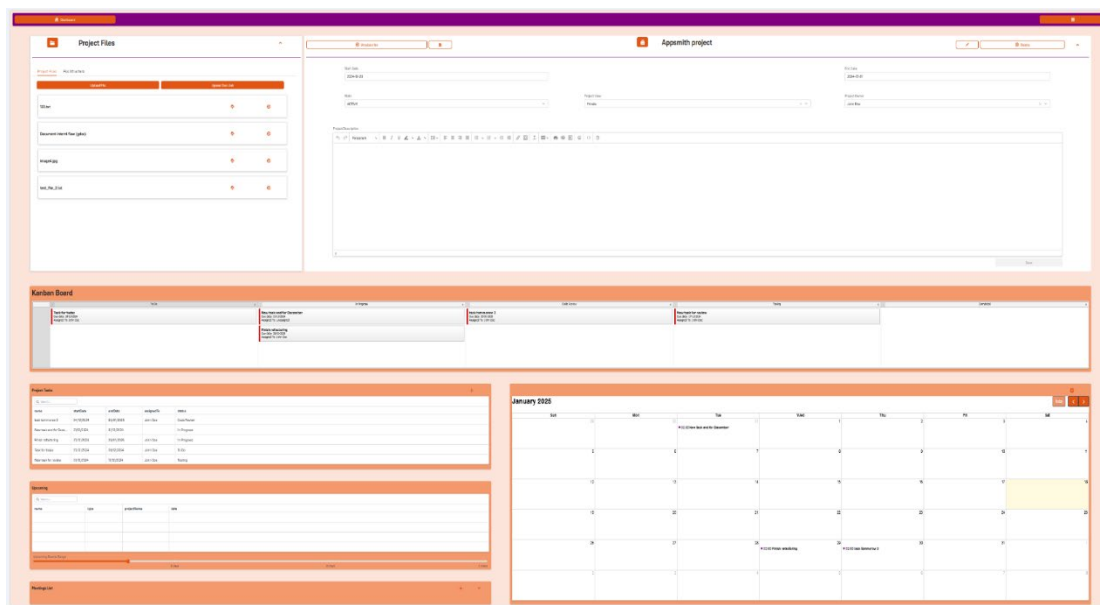
Όσον αφορά τα δικαιώματα των χρηστών, έχουμε υλοποιήσει τρεις διαφορετικούς τύπους χρηστών: Admin, User και Guest καθένας από τους οποίους διαθέτει ξεχωριστά δικαιώματα και επίπεδα πρόσβασης. Στην συγκεκριμένη σελίδα οι Admin και User έχουν ακριβώς τα ίδια δικαιώματα με δυνατότητες δημιουργίας και εγγραφής σε έργα, δημιουργία εργασιών και συναντήσεων.

Από την άλλη οι χρήστες Guest είναι οι πιο περιορισμένοι χρήστες τις εφαρμογής ουσιαστικά μόνο με δικαιώματα ανάγνωσης και καμία δυνατότητα επεξεργασίας ή προσθήκης οντοτήτων. Έτσι στην «Dashboard» σελίδα τα κουμπιά δημιουργίας είναι μη ορατά και στους διάφορους πίνακες αλλά και στο ημερολόγιο με μόνη δυνατότητα του χρήστη να εγγραφεί σε κάποιο έργο.

4.3.2 Project Dashboard

Κάνοντας κλικ σε ένα οποιοδήποτε project από τον πίνακα «Project» του Dashboard μπορούμε να μεταφερθούμε στην σελίδα του συγκεκριμένου έργου. Στην «Project Dashboard» μπορούμε να δούμε αναλυτικότερα τα στοιχεία ενός έργου και να διαχειριστούμε τις διάφορες εργασίες και συναντήσεις του. Όπως μπορούμε να δούμε και από την Εικόνα 9, κάποια widget και οι λειτουργίες τους παραμένουν κοινά με την αρχική

μας σελίδα. Οι πίνακες «Task», «Meeting», «Upcoming» καθώς και το ημερολόγιο διατηρούν την ίδια λειτουργικότητα προσαρμοσμένη όμως στο συγκεκριμένο έργο. Για παράδειγμα, στον πίνακα των εργασιών βλέπουμε πλέον μόνο τις εργασίες που είναι συνδεδεμένα με το συγκεκριμένο έργο και όχι όλες τις εργασίες του χρήστη όπως προηγουμένως. Το ίδιο ισχύει και στους υπόλοιπους πίνακες και widget.



Εικόνα 9. Project Dashboard σελίδα του APM

Λόγω της πληθώρας πληροφοριών που έπρεπε να παρουσιαστούν στη συγκεκριμένη σελίδα, εφαρμόσαμε σε ορισμένα widgets τη δυνατότητα ελαχιστοποίησης, εξασφαλίζοντας ένα πιο καθαρό και ευέλικτο περιβάλλον εργασίας. Συγκεκριμένα, η λειτουργία αυτή υλοποιήθηκε στα widgets "Project Files", "Project Update" και "Meeting". Επιπλέον, η navigation bar στο πάνω μέρος της σελίδας προσαρμόζεται δυναμικά, υιοθετώντας το χρώμα του έργου.

Ας δούμε αναλυτικότερα κάποιες από τις νέες λειτουργίες και widget που προσφέρονται στην σελίδα:

1. **Διαχείριση του Project:** Μέσω της σελίδας ο χρήστης με τα απαραίτητα δικαιώματα τα οποία θα αναλύσουμε παρακάτω, έχει την δυνατότητα να ανανεώσει τα διάφορα πεδία του έργου όπως τον τίτλο του, το χρώμα του, την ημερομηνία έναρξης, την κατάσταση του, την περιγραφή, τον ιδιοκτήτη του κ.α. Επίσης μπορεί να απεγραφεί, να διαγράψει και να προσθέσει χρήστες στο έργο μέσω των αντίστοιχων κουμπιών που βρίσκονται αριστερά και δεξιά από τον τίτλο του.
2. **Project Files:** Το custom widget "Project Files" αποτελείται από δύο διαφορετικές καρτέλες. Τις «Project Files» και «File Structure». Η καρτέλα «Project Files» προσφέρει στους χρήστες τη δυνατότητα να ανεβάζουν αρχεία όπως εικόνες και PDF αλλά και να τα κατεβάσουν με χρήση του αντίστοιχου

κουμπιού. Παράλληλα, παρέχεται η επιλογή διαμοιρασμού συνδέσμων (π.χ. Google Documents), όπου με ένα κλικ ο χρήστης μπορεί να μεταφερθεί άμεσα στον ενδιαφερόμενο διαμοιρασμένο σύνδεσμο, εξασφαλίζοντας ευελιξία στις διαφορετικές ανάγκες των ομάδων. Επιλέγοντας την καρτέλα «File Structure», ο χρήστης μπορεί να δει μια δενδροειδής προβολή (tree-view) των αρχείων με δυνατότητα κατεβάσματος με κλικ στο αντίστοιχο αρχείο. Αυτό το χαρακτηριστικό επιτρέπει στον χρήστη να δει τα αρχεία όχι μόνο του συγκεκριμένου έργου, αλλά και οποιουδήποτε σχετιζόμενου αντικειμένου, όπως των εργασιών και των συναντήσεων. Με αυτόν τον τρόπο, επιτυγχάνεται η συγκεντρωμένη προβολή όλων των σχετικών δεδομένων, διευκολύνοντας τη διαχείριση.

3. **Kanban Board:** Το «Kanban Board» αποτελεί ένα από τα βασικότερα και πιο ευέλικτα εργαλεία οπτικής διαχείρισης έργων, παρέχοντας στους χρήστες τη δυνατότητα να παρακολουθούν την κατάσταση και την πρόοδο των εργασιών με έναν απλό και κατανοητό τρόπο. Στο πλαίσιο του APM, το «Kanban Board» υλοποιήθηκε και αυτό μέσω ενός custom widget και παρέχει δυνατότητες όπως:
 - Αλλαγή της κατάστασης μίας εργασίας με drag-and-drop.
 - Προσθήκη εργασίας σε συγκεκριμένο status.
 - Μετακίνηση με χρήση drag-and-drop των στηλών του που αντιστοιχούν στις καταστάσεις των εργασιών.
 - Μετακίνηση στην σελίδα μίας συγκεκριμένης εργασίας κάνοντας κλικ πάνω στο στοιχείο του.

4. **Απενεργοποίηση λειτουργιών σε μη ACTIVE project:** Μια λειτουργία που προσθέσαμε είναι η απενεργοποίηση των λειτουργιών της σελίδας όταν το έργο δεν βρίσκεται σε κατάσταση ACTIVE. Με αυτόν τον τρόπο θέλαμε να προφυλάξουμε την πληροφορία του εκάστοτε έργου μετά την ολοκλήρωση του. Οι μόνες λειτουργίες που παραμένουν ενεργοποιημένες είναι η αλλαγή της κατάστασης του καθώς και η διαγραφή και απεγραφή από το έργο. Όλες οι υπόλοιπες λειτουργίες της σελίδας σε όλα τα widget της όπως η προσθήκη εργασιών και συναντήσεων, ανέβασμα και διαγραφή αρχείων κα. απενεργοποιούνται.

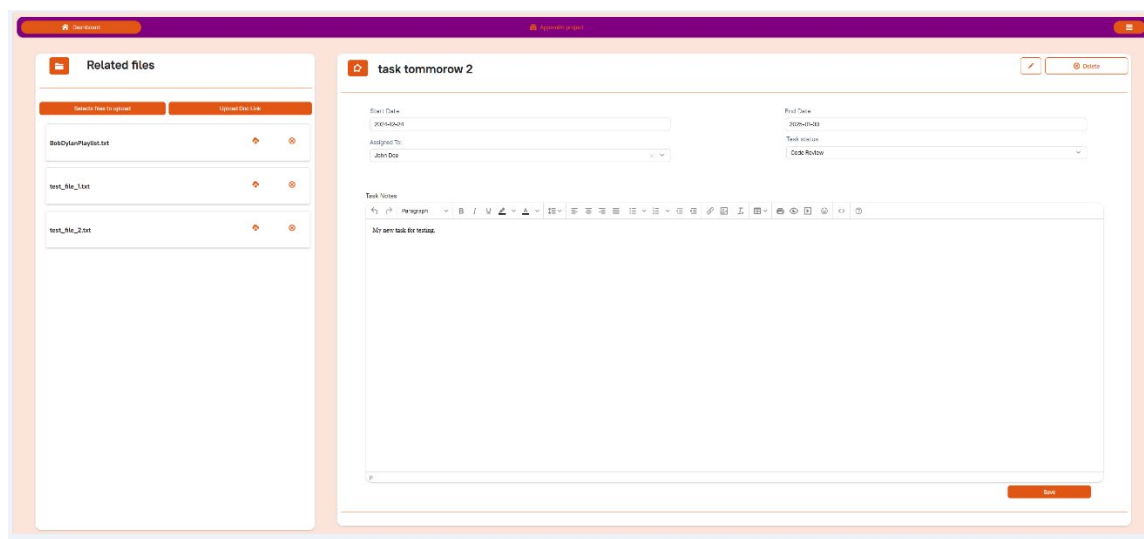
Σχετικά με τα δικαιώματα των χρηστών υπάρχει πλέον σαφής διαφοροποίηση των δικαιωμάτων των Admin και User χρηστών στην σελίδα «Project Dashboard». Οι χρήστες Admin μπορούν να διαχειρίζονται το έργο εκτελώντας ενέργειες όπως ανανέωση των πεδίων, διαγραφή του έργου, διαχείριση χρηστών κα ακόμα και αν δεν είναι ιδιοκτήτες του. Από την άλλη οι User χρήστες δεν μπορούν να εκτελέσουν τέτοιου είδους ενέργειες παρά μόνο εάν είναι οι ιδιοκτήτες του συγκεκριμένου έργου (project owner). Έτσι ένας User χρήστης που δεν είναι ο ιδιοκτήτης του έργου δεν μπορεί να εκτελέσει τις παραπάνω ενέργειες καθώς είτε τα κουμπιά είναι απενεργοποιημένα ή δεν εμφανίζονται καθόλου στον χρήστη. Παρόλα αυτά, στην συγκεκριμένη περίπτωση ο χρήστης μπορεί κανονικά να προσθέσει εργασίες και συναντήσεις όπως και να μεταβεί στις σελίδες τους.

Τέλος οι Guest, όπως και στο «Dashboard» δεν μπορούν να κάνουν καμία ενέργεια πάρα μόνο να μεταβούν σε σελίδες των αντικειμένων με δικαίωμα μόνο προβολής της σχετικής πληροφορίας, χωρίς καμία δυνατότητα επεξεργασίας ή προσθήκης.

4.3.3 Task

Στην σελίδα «Task» μπορούμε να μεταφερθούμε από τις σελίδες «Dashboard» και «Project Dashboard» με τρόπους που ήδη έχουν παρουσιαστεί στις προηγούμενες υποενότητες.

Παρατηρώντας την σελίδα μίας εργασίας, όπως φαίνεται και στην Εικόνα 10, τα διάφορα μέρη που την απαρτίζουν μας είναι πλέον γνωστά. Μπορούμε από το «Task Update» widget να ανανεώσουμε όλα τα πεδία μίας εργασίας καθώς και να την διαγράψουμε και από το «Related Files» widget να ανεβάσουμε αρχεία όπως ακριβώς κάναμε και στην «Project Dashboard» σελίδα, χωρίς όμως την δυνατότητα της δενδροειδούς απεικόνισης.



Εικόνα 10. Task σελίδα του εργαλείου APM.

Μια διαφοροποίηση που μπορούμε να παρατηρήσουμε είναι στο navigation bar, όπου βλέπουμε πως έχει προστεθεί ένα κουμπί με το όνομα του έργου για ευκολία μεταφοράς στην σελίδα του χωρίς την ανάγκη επιστροφής στην «Dashboard» σελίδα. Επίσης να παρατηρήσουμε πως το χρώμα του έργου που συσχετίζεται η εκάστοτε εργασία εμφανίζεται ως το χρώμα του navigation bar, ακριβώς όπως και στην σελίδα «Project Dashboard».

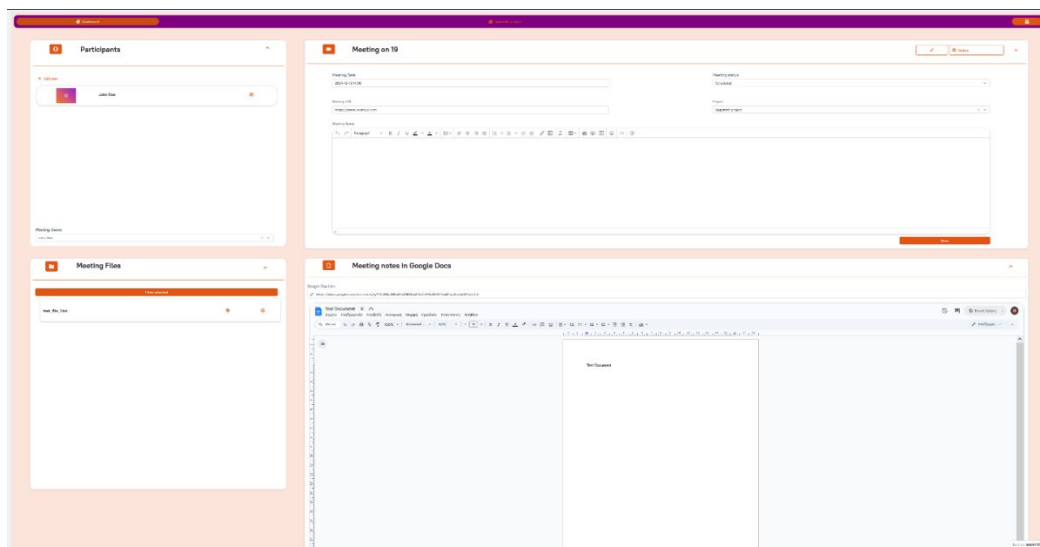
Σε παρόμοιο μοτίβο με την σελίδα «Project Dashboard» βρίσκονται και τα δικαιώματα των διαφόρων τύπων χρηστών. Οι Admin μπορούν, ασχέτως με το αν τους έχει ανατεθεί η συγκεκριμένη εργασία, να διαχειριστούν και να επεξεργαστούν την πληροφορία της ενώ αντίθετα οι User χρήστες, μπορούν να ανανεώσουν την πληροφορία της task μόνο εάν τους έχει ανατεθεί (assigned).

Οι Guest συνεχίζουν να έχουν δικαιώματα μόνο προβολής και καθόλου επεξεργασίας.

4.3.4 Meeting

Όπως και στην περίπτωση των εργασιών, στην «Meeting» σελίδα μπορούμε να μεταφερθούμε είτε από το «Dashboard» είτε από το «Project Dashboard». Παρατηρώντας την Εικόνα 11, ενώ βλέπουμε κάποια κοινά στοιχεία με την σελίδα «Task», υπάρχουν κάποιες προσαρμοσμένες λειτουργίες πάνω στις συναντήσεις που διευκολύνουν την διαχείριση τους και την διεξαγωγή τους που αξίζει να αναλύσουμε:

1. **Meeting Participants:** Για την ευκολότερη διαχείριση των συμμετεχόντων μίας συνάντησης κατασκευάσαμε το συγκεκριμένο widget με δυνατότητες προσθήκης, διαγραφής και παρακολούθησης των συμμετεχόντων της.
2. **Meeting notes in Google Docs:** Το συγκεκριμένο widget εμφανίζεται μόνο στην συγκεκριμένη σελίδα και δίνει την δυνατότητα διαμοιρασμού ενός «Google Document» αρχείου και επεξεργασίας του μέσω της εφαρμογής μας. Ο σύνδεσμος αποθηκεύεται στην βάση δεδομένων μας και συσχετίζεται με την συγκεκριμένη συνάντηση, για ευκολότερη διαχείριση και traceability.



Εικόνα 11. Meeting σελίδα του εργαλείου APM.

Συνεχίζοντας με τα δικαιώματα ανά τύπο χρήστη, τα πράγματα διαφοροποιούνται λίγο σε σχέση με τις υπόλοιπες σελίδες της εφαρμογής καθώς εισέρχεται και ο παράγοντας του εάν ο χρήστης συμμετέχει στην συνάντηση. Έτσι οι Admin έχουν πλήρη δικαιώματα επεξεργασίας των δεδομένων (meeting date, meeting state κτ) ασχέτως του αν είναι οι ιδιοκτήτες της συνάντησης (meeting owners). Όμως σε περίπτωση που ο Admin δεν συμμετέχει στην συγκεκριμένη συνάντηση δεν μπορεί να αλλάξει πληροφορία που αφορά την διεξαγωγή της, να ανεβάσει και να διαγράψει αρχεία, να προσθέσει google doc σύνδεσμο κτλ. Έτσι διαχωρίζεται η στατική πληροφορία όπως η ημερομηνία και η ώρα διεξαγωγής, από την ίδια την διεξαγωγή της συνάντησης με δεδομένα που θα

αλλάζουν δυναμικά (όπως η προσθήκη σημειώσεων ή η παράλληλη επεξεργασία ενός «Google Document»).

Από την άλλη, οι User χρήστες μπορούν να αλλάξουν τα στατικά δεδομένα της συνάντησης μόνο εφόσον είναι ιδιοκτήτες της, ενώ ως συμμετέχοντες μπορούν μόνο να ανεβάσουν και να κατεβάσουν αρχεία, να μοιραστούν google document σύνδεσμο καθώς και να αλλάξουν την περιγραφή. Σε περίπτωση που δεν είναι συμμετέχοντες, τότε δεν μπορούν να επεξεργαστούν καμία πληροφορία.

Τέλος οι Guest διαφοροποιούνται στην συγκεκριμένη σελίδα σε σύγκριση με τις υπόλοιπες, παίρνοντας δικαιώματα επεξεργασίας εφόσον συμμετέχουν στην συνάντηση με δυνατότητες να ανεβάσουν και να κατεβάσουν αρχεία, να μοιραστούν google document σύνδεσμο καθώς και να αλλάξουν την περιγραφή δίνοντας τους έναν πιο ενεργητικό ρόλο στην σελίδα των «Meeting».

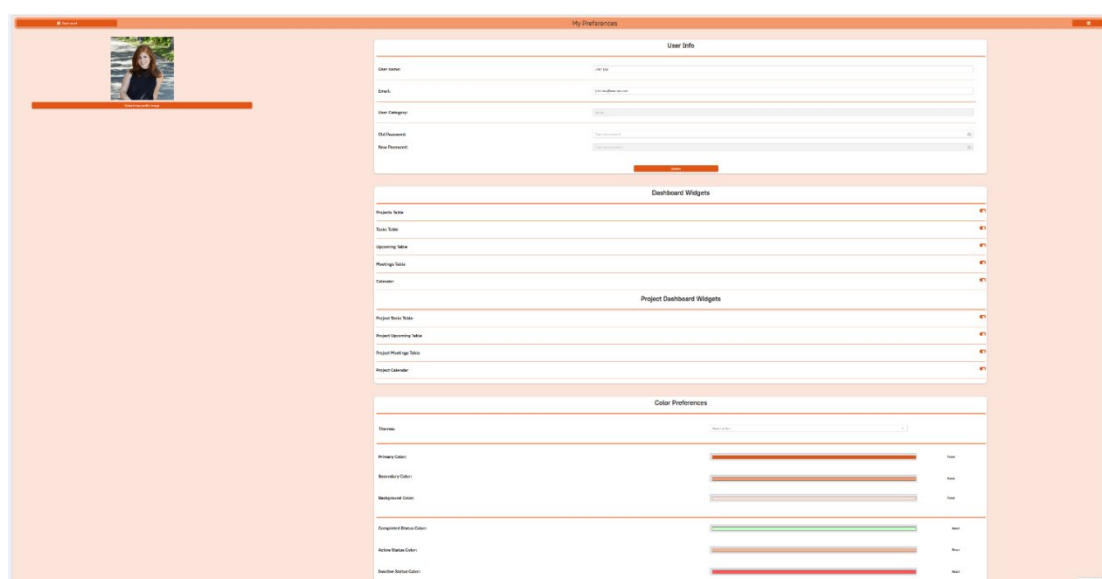
4.3.5 My Preferences

Η σελίδα «My Preferences», είναι προσβάσιμη μέσω του navigation bar πατώντας το κουμπί που εμφανίζεται πάνω δεξιά όπως φαίνεται και στην Εικόνα 12.



Εικόνα 12. Επιλογές του Menu κουμπιού του navigation bar

Η σελίδα «My Preferences» ουσιαστικά καλύπτει τις ανάγκες του χρήστη για διαχείριση των πληροφοριών του καθώς και λειτουργίες παραμετροποίησης σε σχέση με την εμφάνιση της εφαρμογής.



Εικόνα 13. Σελίδα My Preferences του APM.

Όπως μπορούμε να δούμε και στην Εικόνα 13, η σελίδα αποτελείται από διάφορες καρτέλες που αφορούν τα δεδομένα του χρήστη, επιλογές εμφάνισης των widget στην σελίδα «Dashboard» και «Project Dashboard» καθώς και επιλογές χρωμάτων γενικότερα για την εφαρμογή αλλά και ειδικότερα για τον χρωματισμό της στήλης των καταστάσεων των διαφόρων πινάκων.

Αρχικά ο χρήστης μπορεί να ανανεώσει την πληροφορία του λογαριασμού του αλλάζοντας το όνομα χρήστη του, ανεβάζοντας φωτογραφία προφίλ, και ανανεώνοντας το email του και τον κωδικό του.

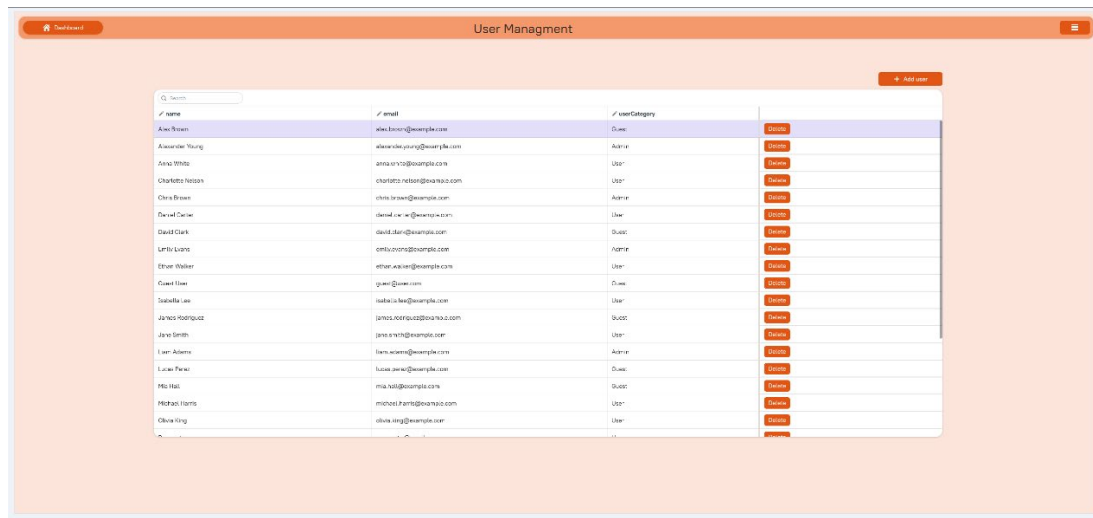
Επιπλέον η δυνατότητα απόκρυψης των ανεπιθύμητων widget στις σελίδες «Dashboard» και «Project Dashboard» δίνει στον χρήστη ελευθερίες παραμετροποίησης της εμφάνισης της εφαρμογής, συμβάλλοντας έτσι στη δημιουργία μιας πιο εξατομικευμένης και λειτουργικής εμπειρίας χρήσης.

Τέλος, οι επιλογές χρωμάτων παρέχουν στον χρήστη τη δυνατότητα να προσαρμόσει την εμφάνιση της εφαρμογής στις προσωπικές του προτιμήσεις. Συγκεκριμένα, μπορεί να επιλέξει τρία βασικά χρώματα: «Primary Color», «Secondary Color» και «Background Color», τα οποία αποτελούν προτιμήσεις σε επίπεδο χρήστη (user-level preferences) και επηρεάζουν τα κύρια χρωματικά στοιχεία της εφαρμογής. Επιπλέον με την δυνατότητα επιλογής των χρωμάτων που αντιστοιχούν στις καταστάσεις των διάφορων οντοτήτων της εφαρμογής ο χρήστης μπορεί να θέσει τα επιθυμητά χρώματα που τον βολεύουν βοηθώντας με αυτόν τον τρόπο στην παρακολούθηση της προόδου μέσω των διάφορων πινάκων και της ευκολίας χρήσης της εφαρμογής.

Να σημειώσουμε πως η λειτουργικότητα της συγκεκριμένης σελίδας παραμένει ίδια και για τους τρεις τύπους χρηστών.

4.3.6 Manage Users

Στην Εικόνα 14 βλέπουμε την σελίδα «Manage Users» η οποία είναι διαθέσιμη αποκλειστικά σε Admin χρήστες και έχει ως κύριο σκοπό τη διαχείριση των χρηστών της εφαρμογής. Μέσω αυτής, οι Admins έχουν τη δυνατότητα να τροποποιήσουν πληροφορίες χρηστών, όπως το όνομα, το email και την κατηγορία τους (user role). Επιπλέον, παρέχεται η δυνατότητα διαγραφής ή προσθήκης νέων χρηστών.



Εικόνα 14. Σελίδα Manage Users του APM.

Η προσθήκη χρηστών από Admins αποτελεί τον μοναδικό τρόπο εγγραφής στην εφαρμογή. Δεδομένου ότι πρόκειται για εσωτερικό εργαλείο, δόθηκε ιδιαίτερη έμφαση στην ασφάλεια και στον έλεγχο της πρόσβασης, αποκλείοντας την εγγραφή χρηστών από τους ίδιους. Όταν ένας Admin προσθέτει έναν νέο χρήστη, καταχωρεί όλες τις απαραίτητες πληροφορίες, όπως το email, το όνομα χρήστη, την κατηγορία του και τον αρχικό κωδικό πρόσβασης.

Ο νέος χρήστης ενημερώνεται μέσω email, όπου λαμβάνει τα στοιχεία του λογαριασμού του και οδηγίες για την πρώτη του σύνδεση. Στο email, προτείνεται έντονα να αλλάξει άμεσα τον αρχικό του κωδικό, προκειμένου να διασφαλιστεί η μέγιστη δυνατή ασφάλεια του λογαριασμού του.

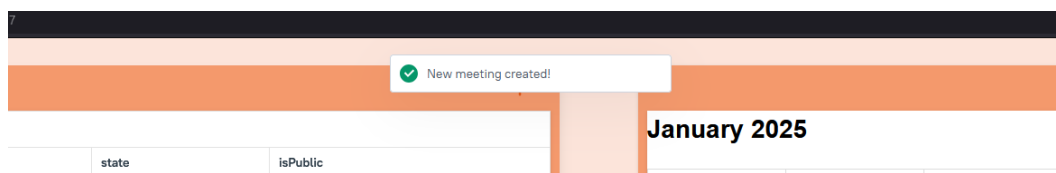
Αυτή η διαδικασία εγγραφής και διαχείρισης χρηστών διασφαλίζει ένα ασφαλές και ελεγχόμενο περιβάλλον, κατάλληλο για τις απαιτήσεις ενός εσωτερικού εργαλείου.

4.3.7 Ειδοποιήσεις

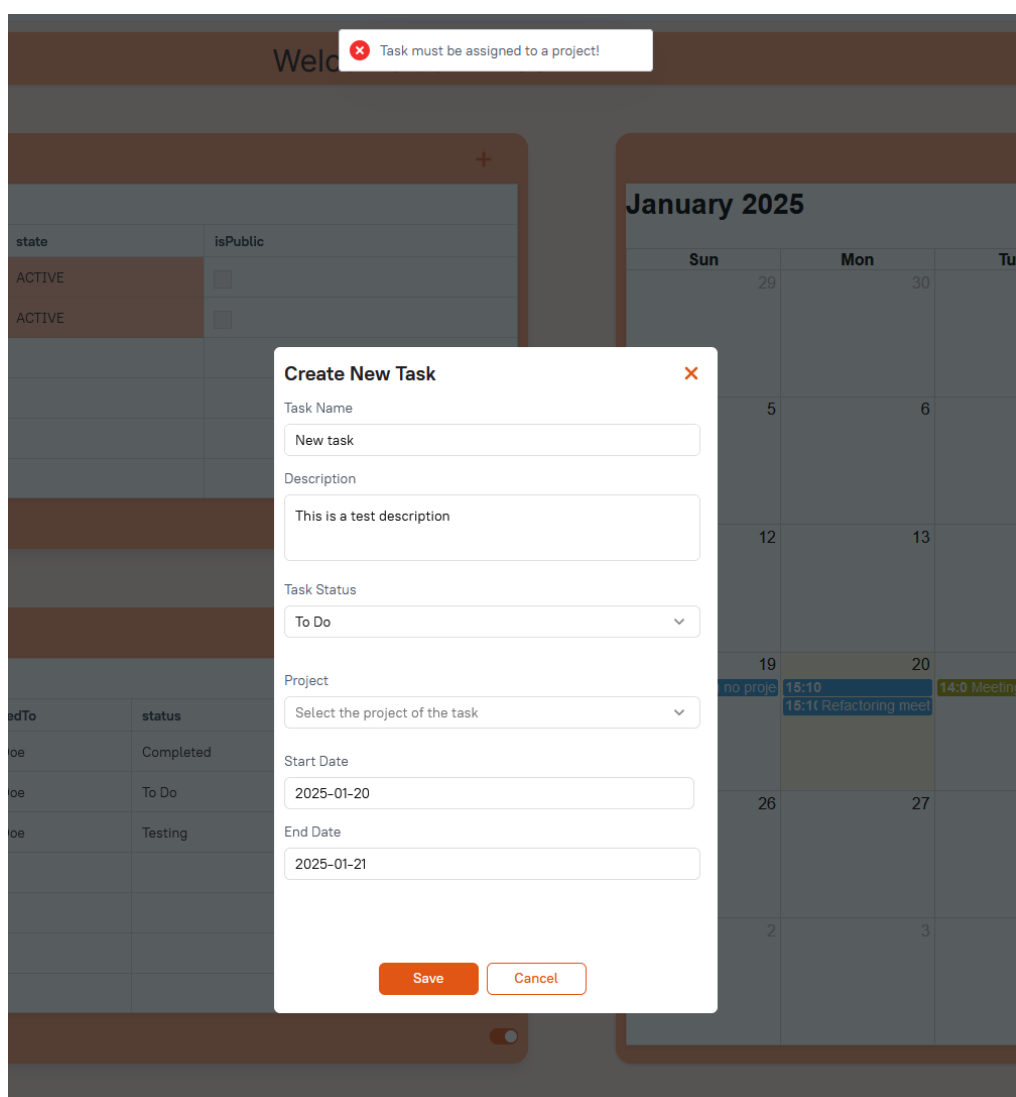
Αφού αναλύσαμε τις λειτουργίες της εφαρμογής μέσα από τις σελίδες της, αξίζει να αναφερθούμε στη λειτουργία των ειδοποιήσεων, οι οποίες διαδραματίζουν σημαντικό ρόλο στην εμπειρία του χρήστη.

Ο πρώτος τύπος ειδοποιήσεων αφορά τα διάφορα alerts που εμφανίζονται κατά τη διάρκεια χρήσης της εφαρμογής. Ο στόχος αυτών των ειδοποιήσεων είναι να ενημερώνουν τον χρήστη για την επιτυχή εκτέλεση μιας ενέργειας, όπως φαίνεται στην Εικόνα 15, ή να προειδοποιήσουν για την αποτυχία της όπως φαίνεται στην Εικόνα 16. Σε κάθε περίπτωση, τα μηνύματα που προβάλλονται προσπαθήσαμε να είναι σαφή, κατανοητά και κατάλληλα προσαρμοσμένα για την εκάστοτε κατάσταση. Στις περιπτώσεις επιτυχίας, τα alerts διαβεβαιώνουν τον χρήστη ότι η ενέργειά του ολοκληρώθηκε με επιτυχία ενώ σε περιπτώσεις σφάλματος τα μηνύματα είναι συγκεκριμένα και παρέχουν χρήσιμες πληροφορίες για τη φύση του προβλήματος. Με

αυτόν τον τρόπο επιτρέπουμε στον χρήστη να κατανοήσει την αιτία του σφάλματος και να προβεί στις απαραίτητες διορθωτικές ενέργειες για την επίλυσή του.



Εικόνα 15. Παράδειγμα επιτυχημένης δημιουργίας ενός Meeting με εμφάνιση του κατάλληλου alert.

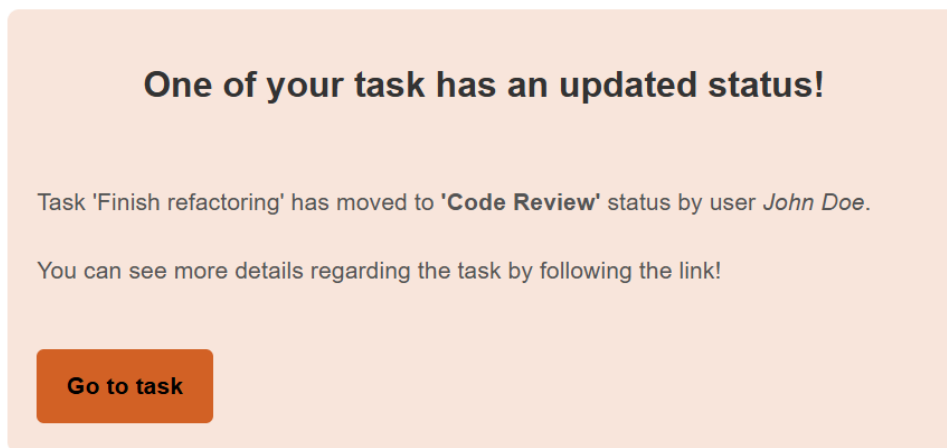


Εικόνα 16. Ανεπιτυχής δημιουργία ενός Task με εμφάνιση του κατάλληλου alert.

Ο δεύτερος τύπος ειδοποιήσεων αφορά τα email. Οι ειδοποιήσεις email προσφέρουν επιπλέον επίπεδο επικοινωνίας, διασφαλίζοντας ότι οι χρήστες ενημερώνονται για σημαντικά γεγονότα ή αλλαγές, ακόμη και όταν δεν βρίσκονται μέσα στην εφαρμογή. Οι ενέργειες που ενεργοποιούν την αποστολή ενός email στους σχετικούς χρήστες περιλαμβάνουν:

- Εγγραφή νέου χρήστη
- Διαγραφή χρήστη
- Ανάθεση task σε χρήστη
- Εγγραφή χρήστη σε project
- Πρόσκληση σε meeting
- Αλλαγή του status ενός task

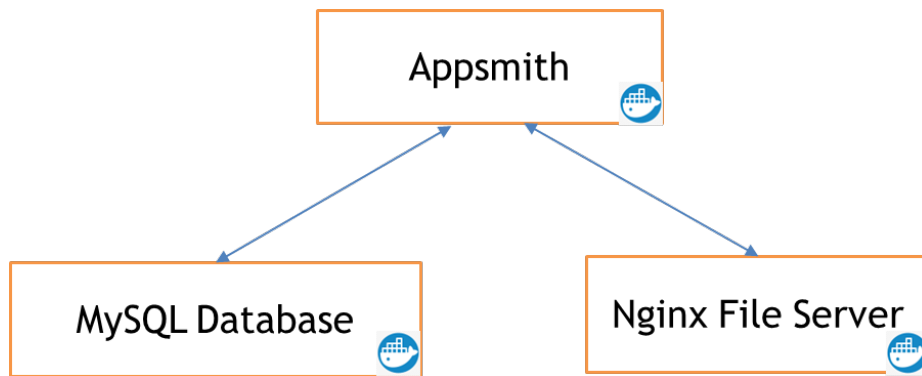
Καθεμία από τις παραπάνω ενέργειες αποστέλλει ένα διαφορετικό email που περιλαμβάνει σχετική πληροφορία για την κάθε περίπτωση. Για παράδειγμα στην αλλαγή της κατάστασης μίας εργασίας ο χρήστης που του έχει ανατεθεί καθώς και ο ιδιοκτήτης του αντίστοιχου έργου του οποίου ανήκει λαμβάνουν ενημερωτικό email σχετικά με την νέα κατάσταση της εργασίας αλλά και έναν σύνδεσμο για άμεση μεταφορά τους στην σελίδα της εργασίας. Η μορφή του συγκεκριμένου email φαίνεται στην Εικόνα 17.



Εικόνα 17. Ενημερωτικό email στην περίπτωση αλλαγής του status ενός task.

4.4 Αρχιτεκτονική του APM

Η αρχιτεκτονική του εργαλείου APM, η οποία φαίνεται στην Εικόνα 18, Εικόνα 17 βασίζεται σε τρία κύρια στοιχεία: τη βάση δεδομένων, το Appsmith και τον file server που διαχειρίζεται τα αρχεία χρηστών της εφαρμογής. Αυτά τα στοιχεία συνεργάζονται για να διασφαλίσουν την ομαλή λειτουργία και απόδοση του συστήματος.



Εικόνα 18. Διάγραμμα αρχιτεκτονικής APM

Κάθε ένα από αυτά τρέχει ως ένα docker container για να εξασφαλιστεί η απομόνωση, η φορητότητα και η ευκολία ανάπτυξης της εφαρμογής. Η χρήση docker containers επιτρέπει την ταχύτερη διαχείριση ενημερώσεων, την εξάλειψη προβλημάτων συμβατότητας μεταξύ των στοιχείων και την απλοποίηση της διαδικασίας ανάπτυξης σε διαφορετικά περιβάλλοντα.

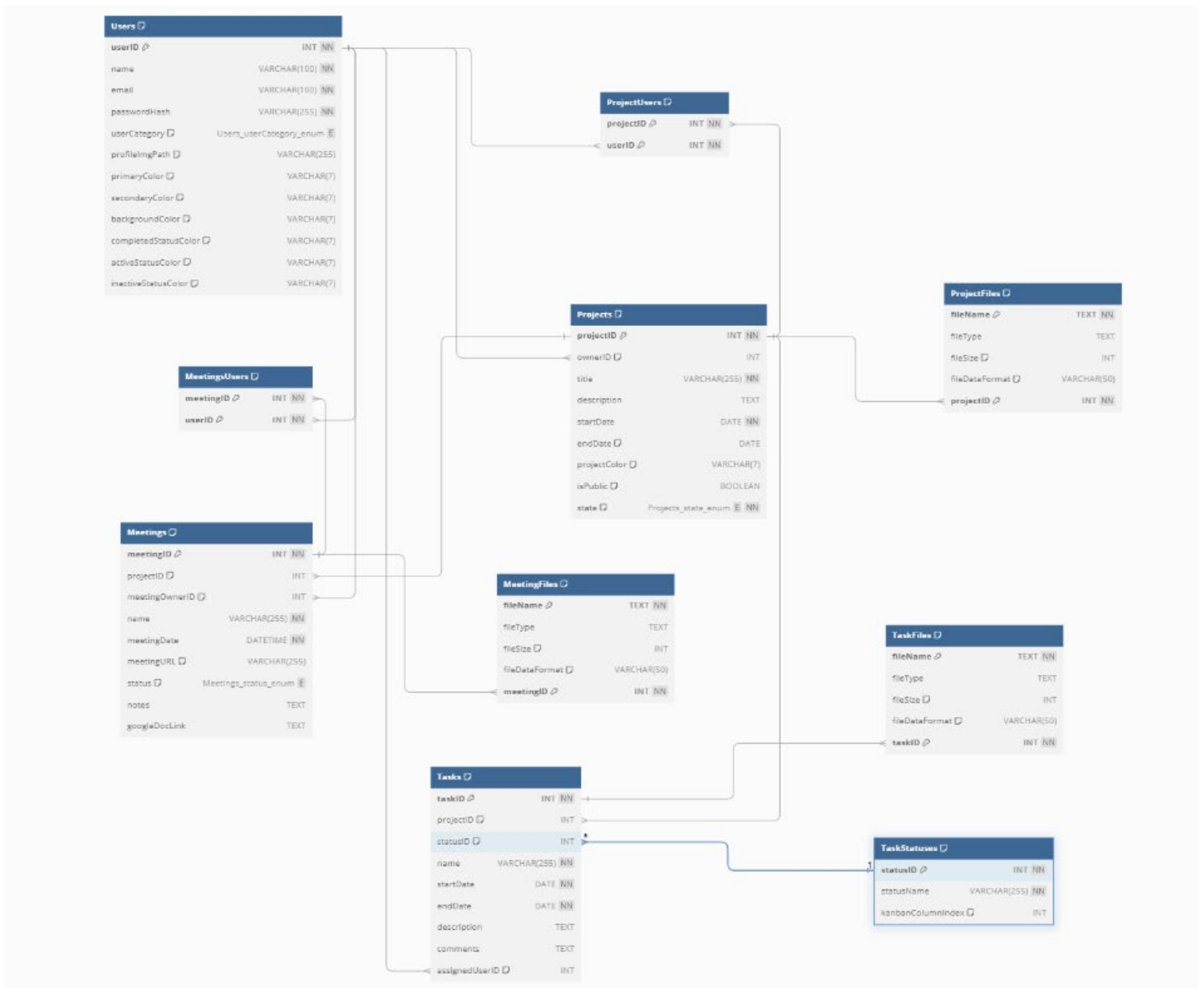
Στις παρακάτω υποενότητες, θα αναλύσουμε κάθε ένα από αυτά τα στοιχεία ξεχωριστά, εστιάζοντας στη λειτουργία, τη σημασία και τον ρόλο τους στο σύνολο της εφαρμογής.

4.4.1 Βάση δεδομένων

Η βάση δεδομένων αποτελεί το θεμέλιο της εφαρμογής μας και είναι υπεύθυνη για την αποθήκευση, την ανάκτηση και τη διαχείριση όλων των δεδομένων που απαιτούνται για τη λειτουργία του εργαλείου. Η ανάπτυξη της βάσης δεδομένων πραγματοποιήθηκε εξ ολοκλήρου από το μηδέν, ώστε να διασφαλιστεί ότι θα είναι πλήρως προσαρμοσμένη στις ιδιαίτερες απαιτήσεις και ανάγκες της εφαρμογής μας.

Για την ανάπτυξη της βάσης δεδομένων επιλέξαμε τη χρήση της MySQL, μιας ευρέως χρησιμοποιούμενης και αξιόπιστης σχεσιακής βάσης δεδομένων. Η επιλογή αυτή έγινε λόγω της σταθερότητας, της ευελιξίας και της δυνατότητάς της να διαχειρίζεται μεγάλα σύνολα δεδομένων, ενώ ταυτόχρονα παρέχει εξαιρετική απόδοση και υποστήριξη για πολύπλοκα ερωτήματα.

Η δομή της βάσης δεδομένων φαίνεται αναλυτικά στην Εικόνα 19. Περιλαμβάνει τόσο τους κύριους πίνακες όσο και υποστηρικτικούς πίνακες, οι οποίοι υλοποιούν διαφορετικά σχεσιακά μοντέλα και διευκολύνουν την αποτελεσματική αποθήκευση δεδομένων.



Εικόνα 19. Το σχήμα της βάσης δεδομένων του APM.

Οι βασικές οντότητες που διαχειρίζεται η εφαρμογή μας μοντελοποιούνται μέσω των κύριων πινάκων Users, Projects, Tasks και Meetings. Αναλυτικότερα:

- **Users**: Περιέχει πληροφορίες για τους χρήστες, όπως το όνομά τους (*name*), το email (*email*), το hash του κωδικού πρόσβασης (*passwordHash*), καθώς και άλλες λεπτομέρειες που αφορούν την εμφάνιση της εφαρμογής (π.χ., *primaryColor*, *secondaryColor*).
- **Projects**: Καταγράφει πληροφορίες για τα έργα, όπως ο τίτλος του έργου (*title*), ο ιδιοκτήτης του έργου (*ownerID*), το χρώμα του έργου (*projectColor*), την

κατάσταση (*state*), το αν είναι δημόσιο ή ιδιωτικό (*isPublic*) και άλλες λεπτομέρειες που διευκολύνουν τη διαχείριση των project.

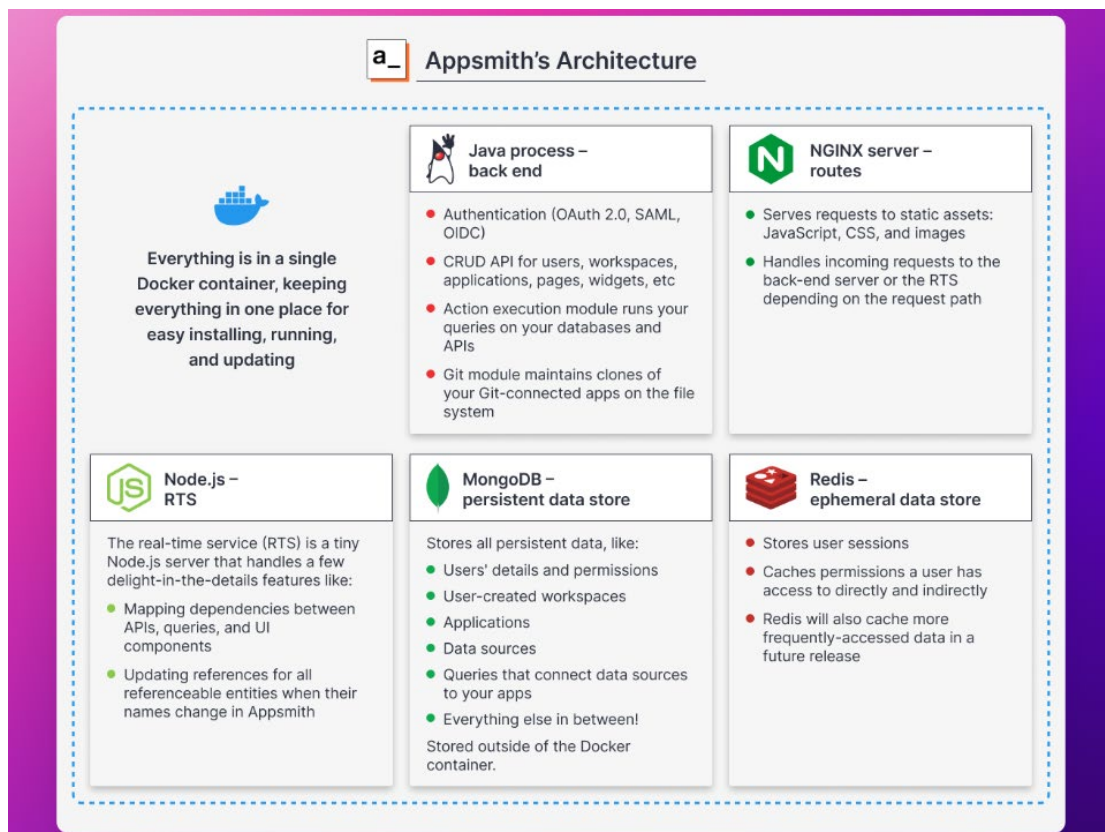
- **Tasks:** Αποθηκεύει δεδομένα σχετικά με τις εργασίες, όπως το όνομα της εργασίας (*name*), τον χρήστη που έχει ανατεθεί η εργασία (*assignedUserID*), την ημερομηνία έναρξης και λήξης (*startDate, endDate*) και το έργο στο οποίο ανήκει (*projectID*).
- **Meetings:** Περιλαμβάνει στοιχεία για τις συναντήσεις, όπως το όνομα της συνάντησης (*name*), την ημερομηνία και ώρα (*meetingDate*), τον ιδιοκτήτη (*meetingOwnerID*) και τον σχετικό σύνδεσμο για την βιντεοκλήση (*meetingURL*).

Εκτός από τους παραπάνω κύριους πίνακες, σχεδιάστηκαν και οι παρακάτω υποστηρικτικοί πίνακες:

- **ProjectUsers:** Υλοποιεί τη σχέση πολλά-προς-πολλά (*many-to-many*) μεταξύ χρηστών και έργων, επιτρέποντας τη συμμετοχή πολλαπλών χρηστών σε ένα έργο.
- **MeetingUsers:** Διαχειρίζεται τη συμμετοχή πολλών χρηστών σε συναντήσεις, επιτρέποντας πολλαπλούς χρήστες να συμμετέχουν σε αυτήν.
- **TaskStatuses:** Παρέχει δυνατότητα δυναμικής διαχείρισης των καταστάσεων των εργασιών (*status*). Ο συγκεκριμένος πίνακας έχει δημιουργηθεί κυρίως με στόχο την δυνατότητα προέκτασης της εφαρμογής σε δυναμική διαχείριση καταστάσεων ανά project μέσω του «Kanban Board».
- **ProjectFiles, MeetingFiles, TaskFiles:** Οι συγκεκριμένοι πίνακες μοντελοποιούν την οντότητα των αρχείων που μπορούμε να ανεβάσουμε στην εφαρμογή στα διάφορα έργα, εργασίες και συναντήσεις. Επιπλέον υλοποιούν μία σχέση πολλά-προς-πολλά (*many-to-many*) μεταξύ των αρχείων και των οντοτήτων. Μέσω αυτών δίνεται η δυνατότητα να μπορούν οι διαφορετικές οντότητες να συσχετίζονται με πολλαπλά αρχεία. Να σημειώσουμε πως τα δεδομένα του κάθε αρχείου δεν κρατούνται άμεσα στην βάση μας αλλά διαχειρίζονται σε συνεργασία με τον file server, τον οποίο θα δούμε αναλυτικότερα παρακάτω.

4.4.2 Appsmith

Σε προηγούμενες ενότητες έχουμε ήδη αναλύσει την χρήση του Appsmith ως ένα low-code εργαλείο ανάπτυξης εσωτερικών εφαρμογών ιστού. Εκτός όμως από την δυνατότητα ανάπτυξης εφαρμογών στα πλαίσια του framework υλοποιείται και το deployment και η διαχείριση της ίδιας της εφαρμογής ή των εφαρμογών που επιθυμούμε να υλοποιήσουμε. Έτσι πρόκειται για μια ολοκληρωμένη λύση που ενσωματώνει λειτουργικότητες ανάπτυξης, διασύνδεσης και διαχείρισης.



Εικόνα 20. Εσωτερική αρχιτεκτονική του Appsmith.

Όπως και η βάση δεδομένων μας, το Appsmith εκτελείται στην μορφή ενός docker container. Η πρόσβαση σε αυτό γίνεται μέσω μιας συγκεκριμένης διεύθυνσης IP και ενός port που καθορίζονται από εμάς. Έτσι, η εφαρμογή μας λειτουργεί εντός αυτού του container, επικοινωνώντας με το αντίστοιχο container της βάσης δεδομένων μας. Η εσωτερική αρχιτεκτονική του Appsmith φαίνεται αναλυτικότερα στην Εικόνα 20.

Το Appsmith, στην συγκεκριμένη περίπτωση λειτουργεί ως το front-end, με επιλεκτικές αρμοδιότητες back-end λόγω της JavaScript παραμετροποίησης, ενώ η βάση δεδομένων μας και ο file server ως το back-end, με παραδοσιακούς ορισμούς της ανάπτυξης εφαρμογών ιστού. Όλα τα component διασυνδέονται μέσω του Appsmith καθιστώντας το κέντρο της υλοποίησής μας.

Μέσω του γραφικού περιβάλλοντος (UI) της εφαρμογής μπορούμε να εκτελέσουμε όλες τις ενέργειες στην εφαρμογή μας που έχουμε δει και προηγουμένως όπως δημιουργία ενός έργου, ανάθεση εργασίας σε χρήστη κτλ. επικοινωνώντας με την βάση και

εκτελώντας τις απαραίτητες εντολές αλλαγής των δεδομένων σε κάθε περίπτωση. Επίσης με χρήση API κλήσεων εκτελούμε τις απαραίτητες ενέργειες στον file server μας διαγράφοντας, προσθέτοντας και ανακτώντας αρχεία.

Το Appsmith λοιπόν λειτουργεί ως η ραχοκοκαλιά της διεπαφής και της διαχείρισης των στοιχείων της εφαρμογής μας, επιτρέποντας την ευκολότερη ανάπτυξη, την ομαλή επικοινωνία με τη βάση δεδομένων και τη διαχείριση αρχείων μέσω του Nginx server. Περισσότερες λεπτομέρειες σχετικά με την υλοποίηση διάφορων λειτουργιών της εφαρμογής θα δούμε στο επόμενο κεφάλαιο.

4.4.3 Nginx Server

Η ανάγκη για την δημιουργία ενός Nginx server προέκυψε από την επιθυμία μας να παρέχουμε στους χρήστες τη δυνατότητα να ανεβάζουν και να διαχειρίζονται αρχεία μέσα από την εφαρμογή. Αν και το Appsmith παρέχει ένα έτοιμο widget για το ανέβασμα αρχείων στο περιβάλλον της εφαρμογής, το βασικό μας πρόβλημα ήταν ο τρόπος αποθήκευσης αυτών των αρχείων, ώστε να είναι δυνατή η ασφαλής και εύκολη ανάκτησή τους.

Μία ιδέα ήταν η αποθήκευση των δεδομένων απευθείας στην βάση μας και συγκεκριμένα στο κατάλληλο πεδίο που θα δημιουργούσαμε στους αντίστοιχους πίνακες. Αποφύγαμε την συγκεκριμένη υλοποίηση καθώς η αποθήκευση αρχείων στη βάση δεδομένων μπορεί να οδηγήσει σε σημαντική αύξηση του μεγέθους της, γεγονός που θα επηρέαζε αρνητικά την απόδοσή της, ειδικά όταν οι όγκοι δεδομένων αυξάνονται με την πάροδο του χρόνου. Επιπλέον, η ανάκτηση μεγάλων αρχείων από τη βάση δεδομένων είναι συχνά πιο χρονοβόρα σε σύγκριση με την αποθήκευσή και ανάκτηση τους μέσω ενός file server.

Έτσι λόγω του ότι θέλαμε να κρατήσουμε ολόκληρη την υλοποίησή μας και τα δεδομένα μας σε τοπικό επίπεδο χωρίς να εμπλέξουμε έτοιμες cloud-hosted λύσεις τις οποίες προτείνει και το Appsmith στο documentation του [20], ήταν απαραίτητη η υλοποίηση ενός server που θα τρέχει σε δικά μας μηχανήματα και θα διαχειριζόμαστε εξ ολοκλήρου εμείς.

Όπως και τα υπόλοιπα στοιχεία της εφαρμογής μας, ο Nginx server τρέχει ως ένα docker container. Αυτό που κάναμε είναι ουσιαστικά να ενεργοποιήσουμε τις λειτουργίες *autoindex* που επιτρέπει την πρόσβαση στα τοπικά αρχεία μας μέσω URL συνδέσμων και της λειτουργίας *dav_methods* που μας επιτρέπει μέσω API κλήσεων να εκτελούμε προσθήκες και διαγραφές αρχείων (*PUT* και *DELETE*).

Με την αξιοποίηση αυτών των λειτουργιών, μέσω του Appsmith υλοποιήσαμε, στις απαραίτητες σελίδες της εφαρμογής μας, τα κατάλληλα API, επιτρέποντας τη διαχείριση αρχείων όπως η αποστολή, η διαγραφή και η ανάκτηση των δεδομένων ενός αρχείου. Στην βάση δεδομένων μας αποθηκεύουμε μόνο κάποια μετα-πληροφορία (metadata) των αρχείων όπως το όνομα, τον τύπο του, το μέγεθος του κα. τα οποία

χρησιμοποιούνται για την εύρεση του κατάλληλου URL path για την εκτέλεση των διαφόρων λειτουργιών στον file server.

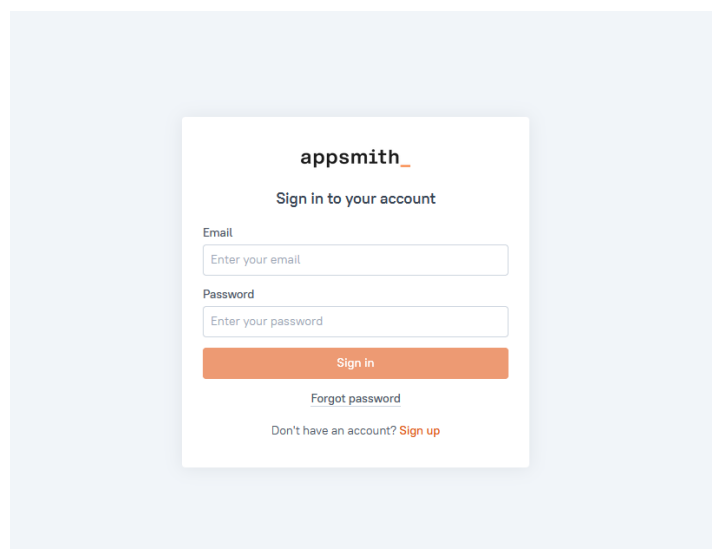
Η παραπάνω λύση εξασφαλίζει ευελιξία, ασφάλεια και πλήρη έλεγχο των δεδομένων μας, χωρίς την εξάρτηση από τρίτους παρόχους.

Κεφάλαιο 5 - Υλοποίηση στο Appsmith

Αφού έχουμε αναλύσει τις δυνατότητες του Appsmith και τις λειτουργίες που προσφέρει η εφαρμογή που αναπτύξαμε, στο παρόν κεφάλαιο θα εστιάσουμε πιο αναλυτικά στα ζητήματα που αφορούν την υλοποίηση της εφαρμογής μας με τη χρήση του συγκεκριμένου low-code framework. Συγκεκριμένα, θα εξετάσουμε την υλοποίηση της ελεγχόμενης πρόσβασης χρηστών και την γενικότερη διαχείριση τους, τη δημιουργία και χρήση προσαρμοσμένων (custom) widget που κατασκευάσαμε, καθώς και τον τρόπο με τον οποίο αξιοποιήσαμε τα έτοιμα widget και τις δυνατότητές τους. Επιπλέον, θα αναφερθούμε στις τεχνικές προσαρμογής και ενσωμάτωσης αυτών των λειτουργιών, ώστε να εξυπηρετήσουν τις ανάγκες της εφαρμογής μας. Ο κώδικας των διάφορων μεθόδων που παρουσιάζονται στο κεφάλαιο, δίνονται στην πλήρη μορφή τους στο Παράρτημα κώδικα για λόγους αναγνωσιμότητας.

5.1 Διαχείριση και ταυτοποίηση χρηστών

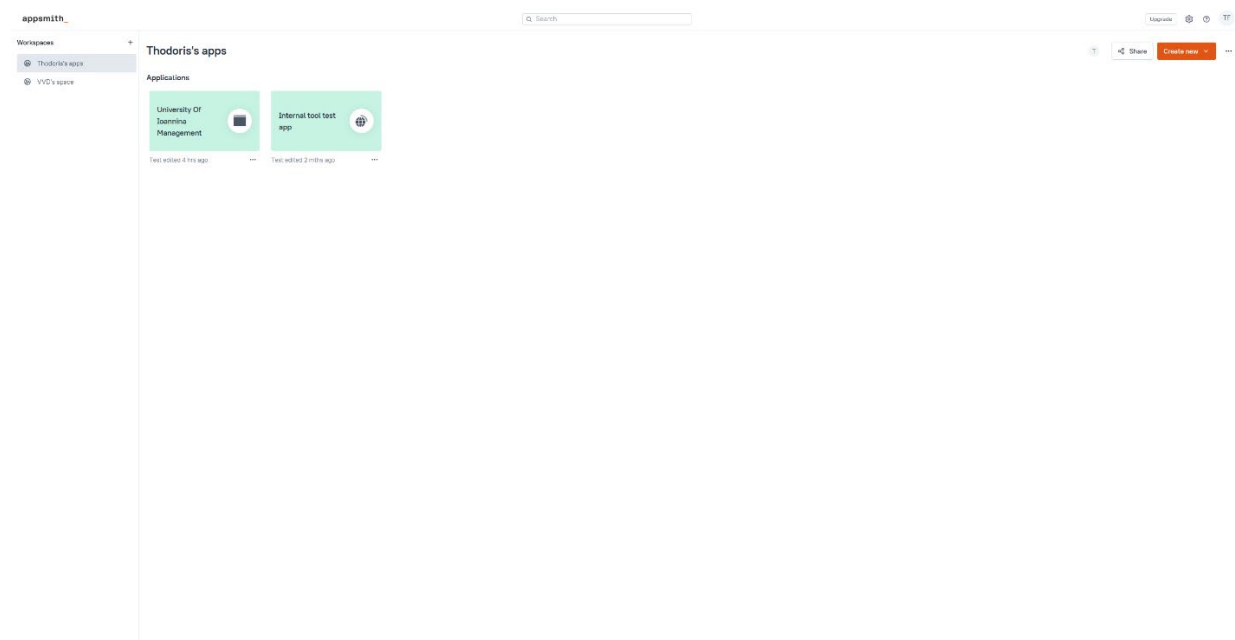
Το Appsmith διαθέτει ένα ενσωματωμένο σύστημα πιστοποίησης (authentication) για την είσοδο στο περιβάλλον ανάπτυξης, το οποίο περιλαμβάνει συγκεκριμένα χαρακτηριστικά και δυνατότητες. Στη δωρεάν έκδοση του framework προσφέρονται τρία επίπεδα χρηστών: Admin, Developer και App Viewer, καθένα από τα οποία συνοδεύεται από συγκεκριμένα δικαιώματα εντός του εργαλείου.



Εικόνα 21. Log-in σελίδα του Appsmith framework.

Η αρχική μας ιδέα ήταν να χρησιμοποιήσουμε τους χρήστες του Appsmith ως χρήστες της εφαρμογής μας χρησιμοποιώντας την log-in σελίδα που φαίνεται στην Εικόνα 21. Δηλαδή, για να αποκτήσει πρόσβαση ένας απλός χρήστης στην εφαρμογή μας, θα έπρεπε

να έχει λογαριασμό στο Appsmith, πιθανώς με δικαιώματα App Viewer, ώστε να μπορεί να βλέπει τις διαθέσιμες εφαρμογές και να εισέρχεται στο εργαλείο. Αξίζει να σημειωθεί ότι οι χρήστες του Appsmith, σε αυτή την περίπτωση, αφορούν το τοπικό περιβάλλον του framework που έχουμε στήσει και δεν σχετίζονται με κάποια εγγραφή στην επίσημη σελίδα του Appsmith.

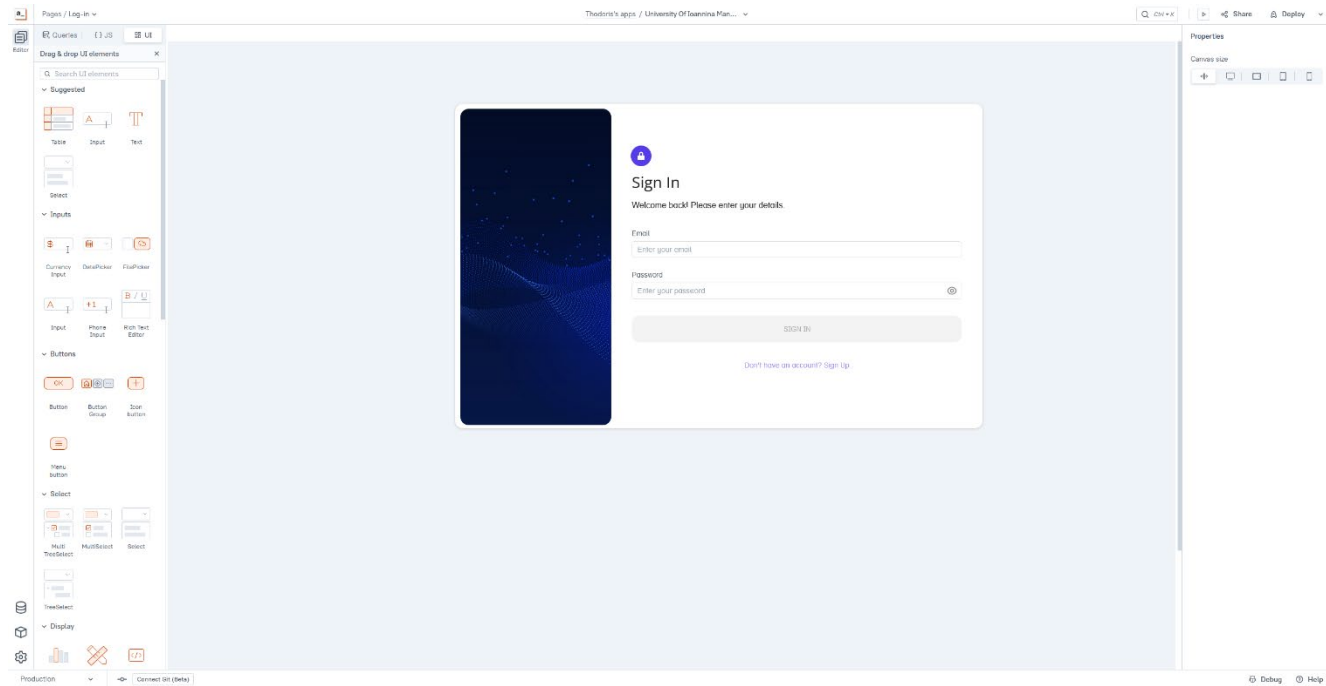


Εικόνα 22. Περιβάλλον διαχείρισης και δημιουργίας εφαρμογών του Appsmith.

Με βάση αυτή την προσέγγιση, κάθε χρήστης θα εισερχόταν στο περιβάλλον που φαίνεται στην Εικόνα 22 και θα επέλεγε την επιλογή "Launch" για την επιθυμητή εφαρμογή.

Ωστόσο, αυτή η υλοποίηση παρουσίαζε σημαντικούς περιορισμούς, κυρίως στο ζήτημα του ελέγχου πρόσβασης των χρηστών. Στη δωρεάν έκδοση του Appsmith, δεν παρέχεται η δυνατότητα περιορισμού πρόσβασης σε μία μόνο εφαρμογή, όπως απαιτούσαμε στην παρούσα υλοποίηση. Έτσι, όλοι οι χρήστες θα μπορούσαν να αποκτήσουν πρόσβαση σε όλες τις διαθέσιμες εφαρμογές που θα μπορούσε να αναπτύξει κάποιος προγραμματιστής, σε οποιοδήποτε διαθέσιμο workspace.

Λόγω αυτών των περιορισμών και της αδυναμίας παραμετροποίησης του ελέγχου πρόσβασης στο δωρεάν πακέτο, αποφασίσαμε να υλοποιήσουμε ένα προσαρμοσμένο σύστημα εισόδου για την πιστοποίηση των χρηστών στην εφαρμογή μας. Με αυτόν τον τρόπο, καταφέραμε όχι μόνο να ξεπεράσουμε τους περιορισμούς της ενσωματωμένης λύσης του Appsmith, αλλά και να δημιουργήσουμε ένα δικό μας σύστημα τριών επιπέδων ελέγχου πρόσβασης μέσα στην εφαρμογή. Αυτή η προσέγγιση μας παρείχε την ευελιξία και τη λειτουργικότητα που απαιτούνταν για να προσαρμόσουμε την εφαρμογή στις συγκεκριμένες ανάγκες μας, κάτι που δεν θα ήταν δυνατό να επιτευχθεί με τη χρήση του ενσωματωμένου συστήματος του Appsmith.



Εικόνα 23. Η σελίδα log-in της εφαρμογής μας στο περιβάλλον ανάπτυξης του Appsmith.

Έτσι μόλις ο χρήστης συνδεθεί στην εφαρμογή μας απευθείας μέσω του URL της, του εμφανίζεται η custom log-in σελίδα που υλοποιήσαμε, η οποία φαίνεται στην Εικόνα 23.

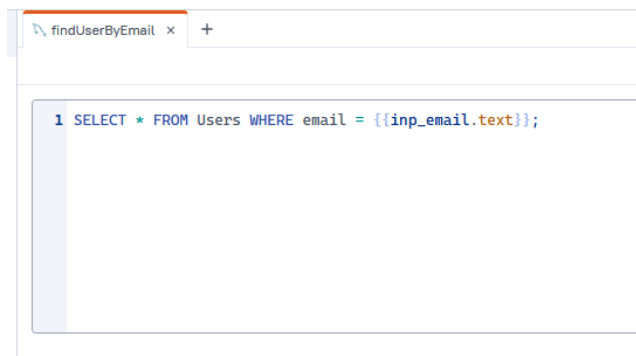
Ας δούμε αναλυτικότερα κάποιες μεθόδους και SQL queries, που απαιτήθηκαν για την υλοποίηση του authentication των οποίων ο κώδικας φαίνεται στην Εικόνα 24 και Εικόνα 25 αντίστοιχα:

```

12 deleteStore: () => {
13   //Reset fields
14   inp_email.setValue('');
15   inp_password.setValue('');
16
17   //Remove everything from appsmith store
18   Object.keys(appsmith.store).forEach(key => {
19     storeValue(key, undefined);
20   });
21 },
22
23 verifyHash: (password, hash) => {
24   return dcodeIO.bcrypt.compareSync(password, hash);
25 },
26
27 createToken: async (user) => {
28   return jsonwebtoken.sign(user, 'secret', {expiresIn: 60*60});
29 },
30
31 generateSecretKey() {
32   const array = new Uint8Array(32); // 32 bytes = 256 bits (AES-256 strength)
33   crypto.getRandomValues(array);
34   return btoa(String.fromCharCode.apply(null, array));
35 },
36
37 signIn: async () => {
38   const password = inp_password.text;
39
40   const [user] = await findUserByEmail.run(inp_email.text);
41
42   if (user && this.verifyHash(password, user.passwordHash))
43   {
44     storeValue('user', user);
45     storeValue('userRights', this.userRights);
46     storeValue('fileServerURL', this.fileServerURL);
47     storeValue('smtpEmail', this.smtpEmail);
48     storeValue('secretKey', this.generateSecretKey());
49     storeValue('token', await this.createToken(user))
50     .then(navigateTo('Dashboard'))
51   }
52   else
53   {
54     return showAlert('Invalid email/password combination', 'error');
55   }
56 },

```

Εικόνα 24. Μέθοδοι στην σελίδα Log-in.



```
1 SELECT * FROM Users WHERE email = {{inp_email.text}};
```

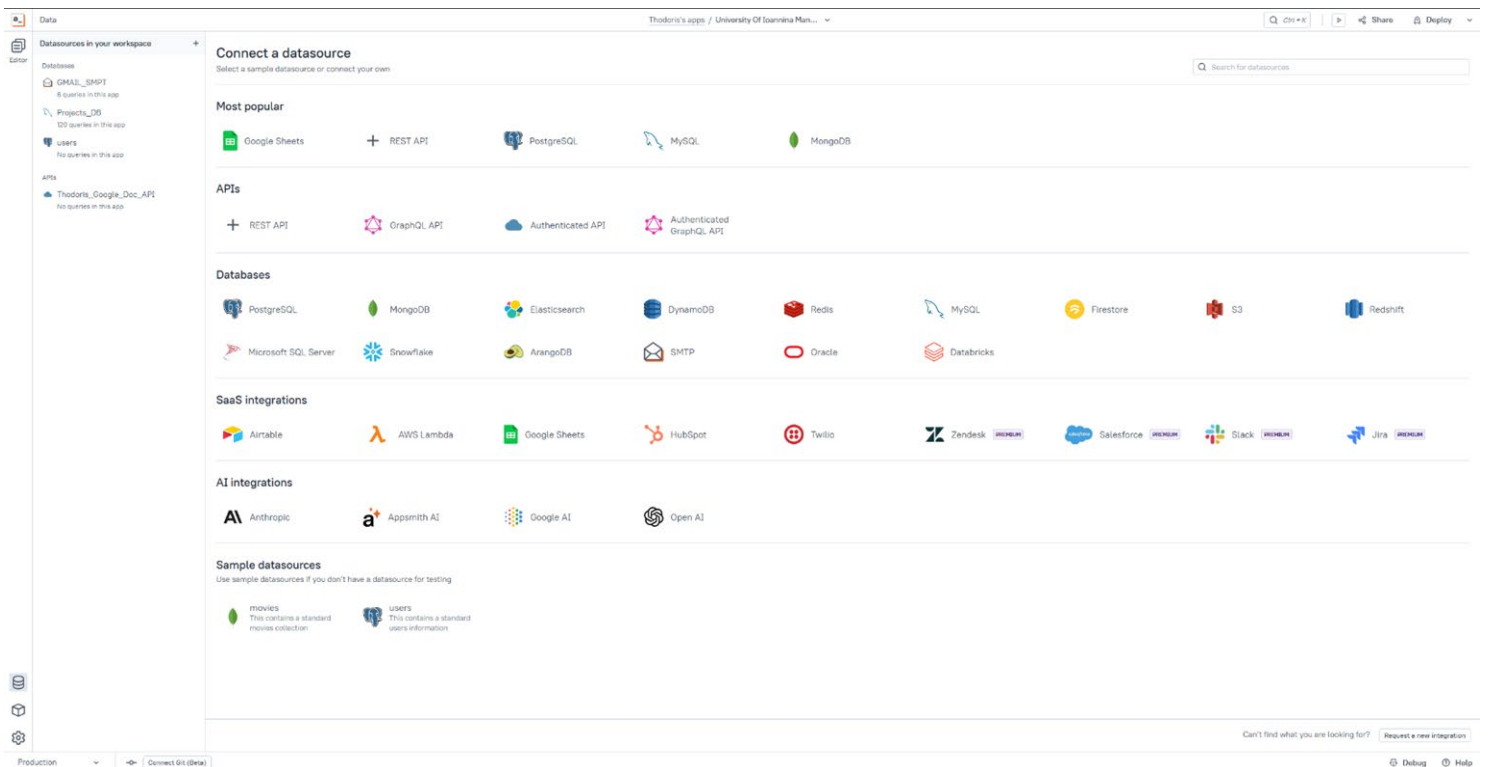
Εικόνα 25. Το μοναδικό query που χρειαζόμαστε στην Log-in σελίδα.

- **signIn():** Η μέθοδος υλοποιεί τη διαδικασία εισόδου (sign in) του χρήστη στην εφαρμογή. Αρχικά, ελέγχει αν ο χρήστης με το συγκεκριμένο email υπάρχει στη βάση δεδομένων. Εφόσον ο χρήστης εντοπιστεί, πραγματοποιείται επιβεβαίωση της εγκυρότητας του αποκρυπτογραφημένου *passwordHash* που υπάρχει στη βάση, συγκρίνοντάς το με τον κωδικό που εισήγαγε ο χρήστης.
 - Σε περίπτωση επιτυχούς ταυτοποίησης, αποθηκεύονται στο *appsmith.store* (ένας χώρος αποθήκευσης δεδομένων που παραμένει ενεργός καθ' όλη τη διάρκεια της συνεδρίας του χρήστη) τα στοιχεία του χρήστη, το παραγόμενο *token*, καθώς και πρόσθετες τιμές, όπως η διεύθυνση του file server (*fileServerURL*) και η διεύθυνση email του SMTP server (*smtpEmail*).
 - Σε περίπτωση αποτυχίας, εμφανίζεται στον χρήστη σχετικό μήνυμα σφάλματος, ενημερώνοντάς τον για την αποτυχημένη προσπάθεια εισόδου.
- **verifyHash():** Χρησιμοποιείται για την επεξεργασία του *passwordHash* που είναι αποθηκευμένο στη βάση δεδομένων, αποκρυπτογραφεί την τιμή του και τη συγκρίνει με τον κωδικό που εισήγαγε ο χρήστης, προκειμένου να τον ταυτοποιήσει.
- **createToken():** Δημιουργεί ένα *JSON Web Token (JWT)*, το οποίο χρησιμοποιείται για την επιβεβαίωση της επιτυχούς ταυτοποίησης του χρήστη και για τη διατήρηση της σύνδεσής του στην εφαρμογή.
- **deleteStore():** Υπεύθυνη για τη διαγραφή όλων των πληροφοριών που είναι αποθηκευμένες στο *appsmith.store*. Η μέθοδος αυτή εκτελείται κάθε φορά που φορτώνει η σελίδα *Log-in*, εξασφαλίζοντας ότι δεν παραμένουν δεδομένα από προηγούμενες συνεδρίες.
- **generateSecretKey():** Παράγει ένα μυστικό κλειδί (*secret key*), το οποίο χρησιμοποιείται για σκοπούς κρυπτογράφησης μέσα στην εφαρμογή.

5.2 Σύνδεση βάσης δεδομένων και υλοποίηση queries

Αφού εξετάσαμε την υλοποίηση της λειτουργίας log-in της εφαρμογής, πριν προχωρήσουμε στην ανάλυση επιπλέον λειτουργιών και των τρόπων υλοποίησής τους, είναι χρήσιμο να δούμε αναλυτικότερα τη διαδικασία σύνδεσης και διαχείρισης της βάσης δεδομένων στο περιβάλλον του Appsmith.

Ο προγραμματιστής έχει στη διάθεσή του μια πληθώρα επιλογών για την πηγή δεδομένων που θα χρησιμοποιήσει, όπως φαίνεται στην Εικόνα 27.



Εικόνα 26. Οι διαθέσιμες πηγές δεδομένων που παρέχει το Appsmith.

Στη δική μας περίπτωση, όπως έχει ήδη αναφερθεί, επιλέξαμε να αναπτύξουμε τη βάση δεδομένων μας χρησιμοποιώντας την MySQL. Πατώντας την αντίστοιχη επιλογή, μεταφερόμαστε σε μια σελίδα όπου απαιτείται η συμπλήρωση των απαραίτητων στοιχείων για τη διασύνδεση της βάσης, όπως η διεύθυνση της (host address), η θύρα σύνδεσης (port), το όνομα χρήστη και ο κωδικός πρόσβασης. Το περιβάλλον εισαγωγής αυτών των στοιχείων φαίνεται στην Εικόνα 29.

Αφού συμπληρώσουμε την απαραίτητη πληροφορία και αποθηκεύσουμε την συγκεκριμένη πηγή δεδομένων, μπορούμε να την αξιοποιήσουμε μέσα στην εφαρμογή, δημιουργώντας και εκτελώντας queries για την ανάκτηση, εισαγωγή ή επεξεργασία δεδομένων, διευκολύνοντας έτσι τη δυναμική διαχείριση της πληροφορίας.

Production

Staging Business

MySQL host address MySQL port

myapp.abcd.mysql.net 3306

⊕ Add more

Database name

admin

Authentication

MySQL username

Username

MySQL password Encrypted

Password

SSL (optional)

SSL mode

Default

MySQL Specific Parameters

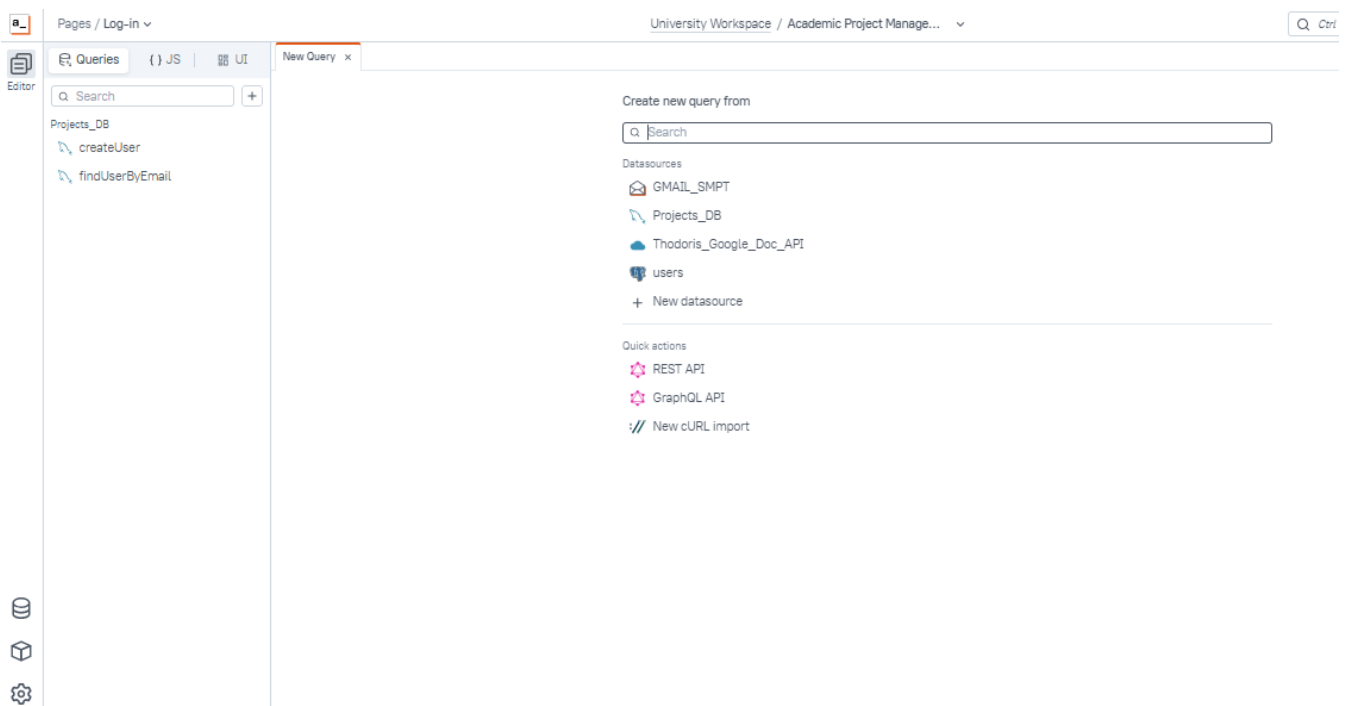
Server Timezone Override

UTC or any valid timezone

Test configuration Cancel Save

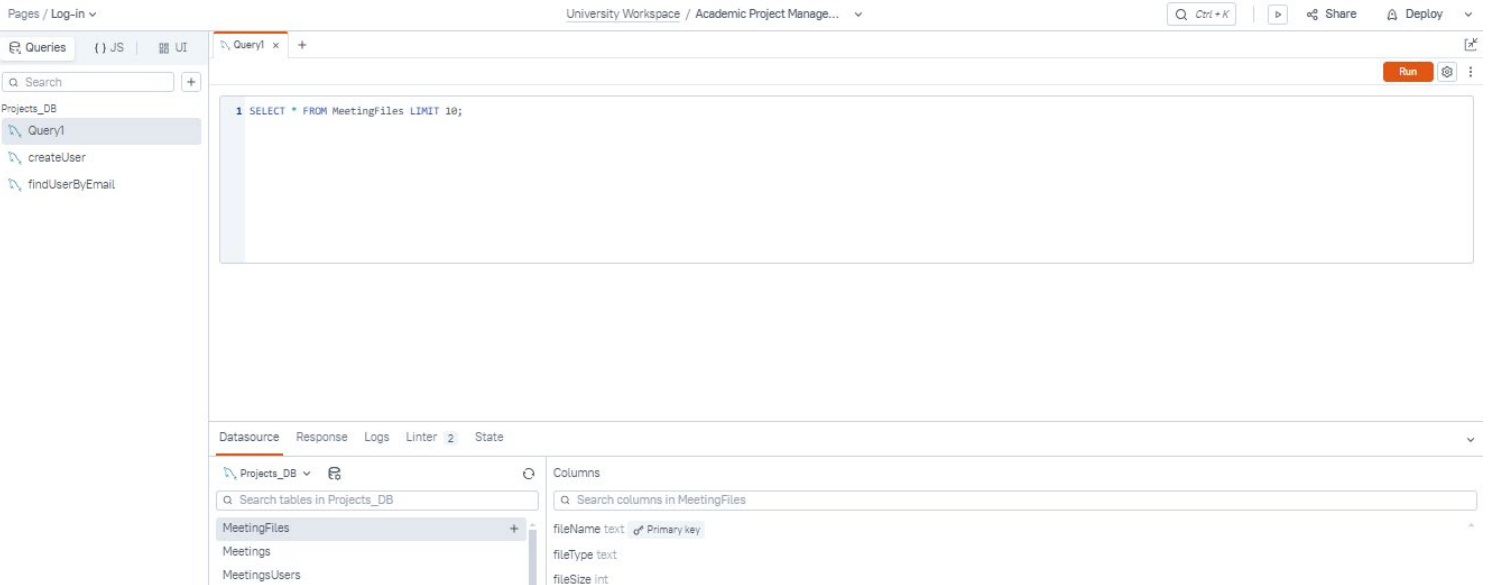
Εικόνα 27. Φόρμα συμπλήρωσης της απαραίτητης πληροφορία για την σύνδεση μίας MySQL βάσης δεδομένων στο Appsmith.

Η δυνατότητα δημιουργίας queries δίνεται από την καρτέλα 'Queries' που υπάρχει στο περιβάλλον ανάπτυξης πάνω αριστερά, όπως φαίνεται στην Εικόνα 23. Με την επιλογή της συγκεκριμένης καρτέλας μεταφερόμαστε στο περιβάλλον που φαίνεται στην Εικόνα 28.

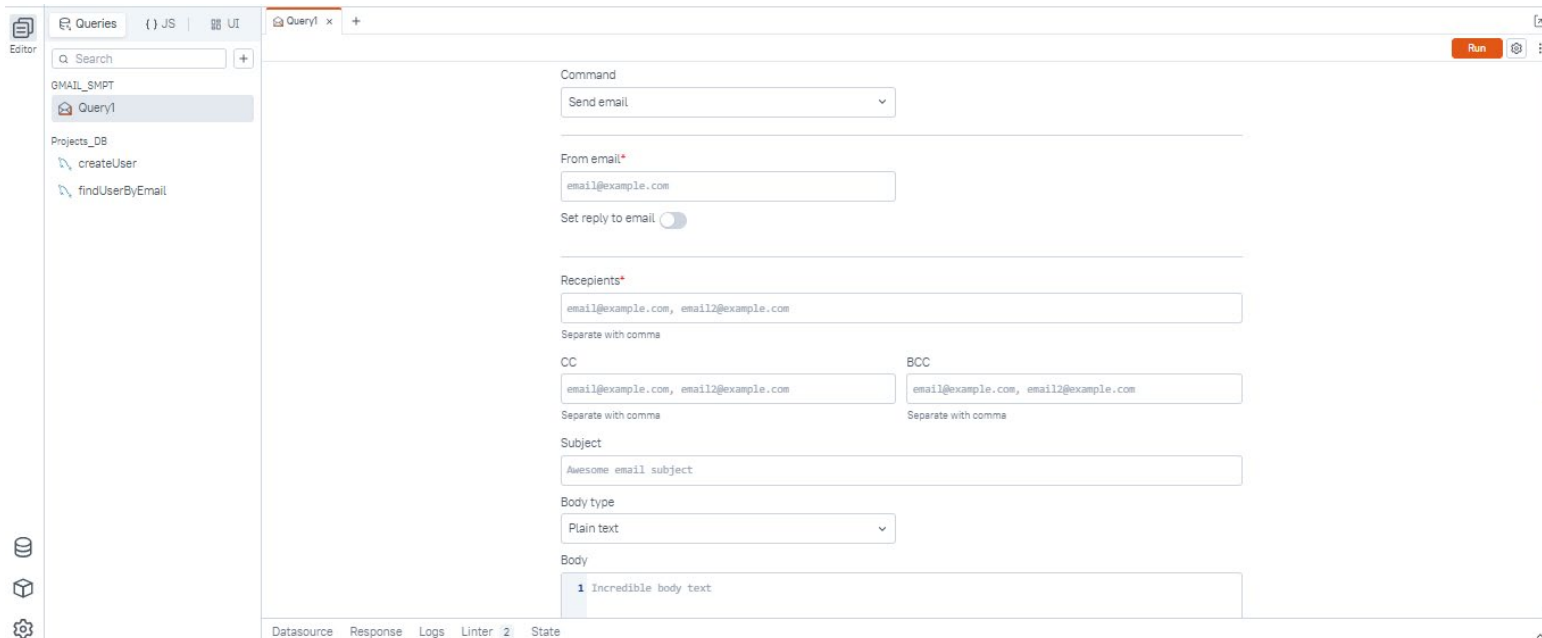


Εικόνα 28. Περιβάλλον δημιουργίας queries στην σελίδα Log-in της εφαρμογής.

Από την συγκεκριμένη σελίδα μπορούμε να επιλέξουμε την πηγή δεδομένων στην οποία θέλουμε να δημιουργήσουμε ένα query για ανάκτηση ή επεξεργασία των δεδομένων της. Για παράδειγμα στην περίπτωση των βάσεων δεδομένων μας δίνεται η δυνατότητα να γράψουμε SQL queries προς την βάση ενώ στην περίπτωση μίας πηγής δεδομένων SMTP server μεταφερόμαστε σε ένα ειδικό περιβάλλον για συμπλήρωση στοιχείων όπως το email παραλήπτη και το περιεχόμενο του email που θέλουμε να στείλουμε. Οι δύο διαφορετικές σελίδες φαίνονται στην Εικόνα 30 και την Εικόνα 31 αντίστοιχα.



Εικόνα 30. Η σελίδα για γραφή των SQL ερωτήσεων στην βάση δεδομένων μας.



Εικόνα 29. Η σελίδα για δημιουργία ενός email query στο Appsmith.

Αξίζει να σημειωθεί ότι η δημιουργία μιας πηγής δεδομένων, όπως φαίνεται στην Εικόνα 29, πραγματοποιείται σε επίπεδο εφαρμογής. Αυτό σημαίνει ότι η συγκεκριμένη πηγή είναι διαθέσιμη σε όλες τις σελίδες της εφαρμογής, επιτρέποντας την επιλογή της και τη δημιουργία των απαραίτητων queries.

Αντίθετα, η δημιουργία των queries, όπως φαίνεται στην Εικόνα 30, γίνεται σε επίπεδο σελίδας. Αυτό σημαίνει ότι ένα query που υλοποιείται, για παράδειγμα, στη σελίδα “Dashboard” δεν είναι προσβάσιμο από τη σελίδα “Project Dashboard”. Σε περιπτώσεις όπου το ίδιο query απαιτείται σε πολλαπλές σελίδες, το Appsmith παρέχει τη λειτουργία “Copy to page”, διευκολύνοντας την αντιγραφή και μεταφορά του σε άλλες σελίδες χωρίς να χρειάζεται εκ νέου υλοποίηση.

Τα queries που δημιουργούμε μπορούμε να τα καλέσουμε μετέπειτα μέσα στην εφαρμογή μας είτε απευθείας πάνω σε κάποιο widget είτε σε κάποια JavaScript μέθοδο χρησιμοποιώντας την ενσωματωμένη συνάρτηση `run()` που παρέχει το Appsmith. Έτσι για παράδειγμα αν θα θέλαμε να εκτελέσουμε και να ανακτήσουμε τα δεδομένα ενός query με όνομα ‘Query1’ θα γράφαμε τον κώδικα που φαίνεται στην Εικόνα 31.

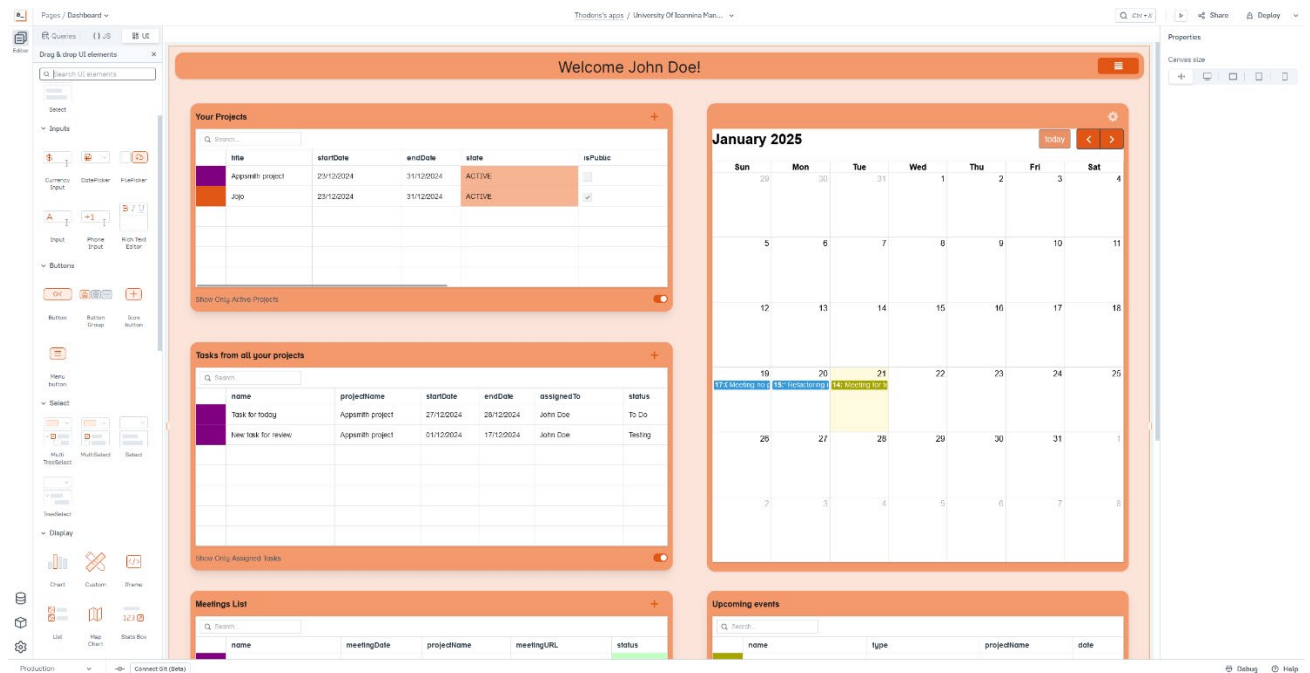
```
1  
2 Query1.run();  
3 const queryData = Query1.data;
```

Εικόνα 31. Τρόπος εκτέλεσης και ανάκτησης των δεδομένων ενός query στο Appsmith.

Έτσι με την παραπάνω διαδικασία δημιουργήσαμε τις απαραίτητες πηγές δεδομένων και υλοποιήσαμε τα απαραίτητα queries ανά σελίδα της εφαρμογής για την υλοποίηση των αναγκαίων λειτουργιών που αφορά τα δεδομένα της εφαρμογής μας.

5.3 Γραφικό περιβάλλον με χρήση έτοιμων widget

Από το περιβάλλον που απεικονίζεται στην Εικόνα 22, οι προγραμματιστές έχουν τη δυνατότητα να δημιουργούν και να διαχειρίζονται τις εφαρμογές τους. Πατώντας το κουμπί "Edit" σε μία εφαρμογή, μεταφερόμαστε στο περιβάλλον ανάπτυξης της. Ας εξετάσουμε αναλυτικότερα το περιβάλλον αυτό, χρησιμοποιώντας ως παράδειγμα τη σελίδα "Dashboard" της εφαρμογής μας.



Εικόνα 32. Η σελίδα "Dashboard" σε περιβάλλον ανάπτυξης.

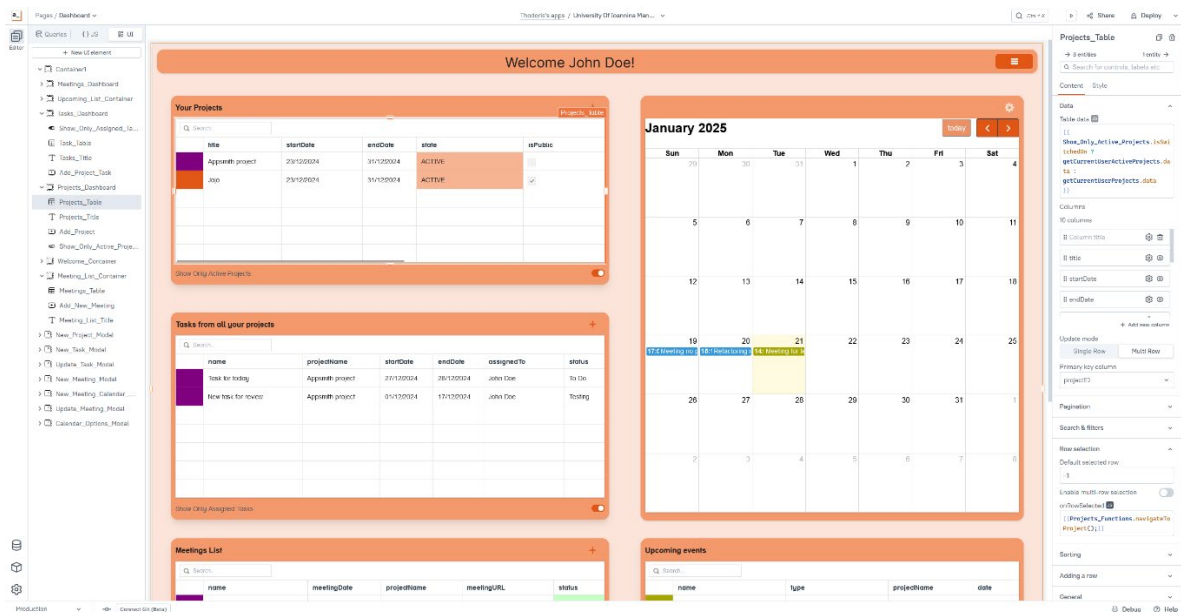
Στην Εικόνα 32 παρατηρούμε, στην αριστερή πλευρά της εικόνας, τις διάφορες επιλογές διαθέσιμων widget. Κάποια από τα έτοιμα widget που χρησιμοποιήθηκαν στην εφαρμογή μας περιλαμβάνουν: "Button", "Icon Button", "Select", "List", "Table", "Container", και "Modal". Χρησιμοποιώντας τη λειτουργία *drag-and-drop*, μπορούμε να προσθέσουμε οποιοδήποτε widget στο παράθυρο της εφαρμογής μας. Με την επιλογή ενός widget, εμφανίζονται στα δεξιά οι διαθέσιμες επιλογές διαμόρφωσης του. Αυτές κατηγοριοποιούνται σε δύο ομάδες:

- **Content:** Περιλαμβάνει επιλογές σχετικές με τη λειτουργικότητα και τη λογική του widget.
- **Style:** Περιλαμβάνει επιλογές που αφορούν την εμφάνιση και τον σχεδιασμό.

Ας πάρουμε ως παράδειγμα την υλοποίηση που αφορά τον πίνακα των έργων. Η δομή του περιλαμβάνει ένα εξωτερικό 'Container' widget, μέσα στο οποίο έχουν προστεθεί τα εξής:

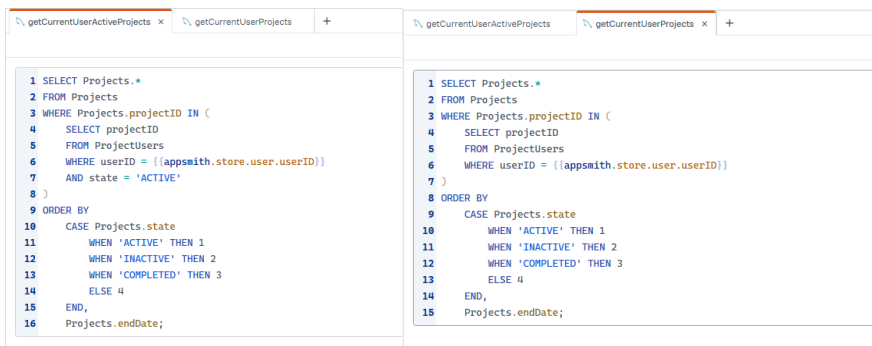
- Ένα 'Text' widget που περιλαμβάνει τον τίτλο "Your Projects"

- Ένα 'Switch' widget που βλέπουμε στο κάτω μέρος με περιγραφή 'Show Only Active Projects' για εμφάνιση μόνο των ACTIVE project του χρήστη.
- Ένα 'Icon Button' widget για την δημιουργία ενός καινούργιου project.
- Ένα 'Modal' widget που υλοποιεί την φόρμα συμπλήρωσης των νέων στοιχείων ενός καινούργιου project.
- Και ένα 'Table' widget, το οποίο περιλαμβάνει όλη την πληροφορία των project που είναι εγγεγραμμένος ο συγκεκριμένος χρήστης.



Εικόνα 33. Διαθέσιμες επιλογές στην καρτέλα 'Content' επιλέγοντας ένα 'Table' widget.

Για τη σύνδεση των δεδομένων από τη βάση με τον πίνακα των έργων, ακολουθείται η εξής διαδικασία: Όπως φαίνεται και στην Εικόνα 33, επιλέγοντας τον πίνακα από το γραφικό περιβάλλον, εμφανίζονται τα διάφορα πεδία του και οι ενέργειες που μπορούμε να εκτελέσουμε. Η διασύνδεση των δεδομένων γίνεται μέσω του πεδίου 'Table Data'. Στην υλοποίησή μας, χρησιμοποιούμε δύο διαφορετικά queries: *getCurrentUserActiveProjects* και *getCurrentUserProjects*. Η εναλλαγή των δεδομένων που εμφανίζονται στον πίνακα γίνεται δυναμικά, ανάλογα με την κατάσταση (ενεργοποιημένο ή απενεργοποιημένο) του 'Switch' widget. Ο κώδικας των queries φαίνεται στην Εικόνα 34.



Εικόνα 34. Ο κώδικας των queries *getCurrentUserProjects* και *getCurrentUserActiveProjects*. Αξίζει να παρατηρήσουμε την χρήση του *appsmith.store* για την ανάκτηση του *userID* του συνδεδεμένου χρήστη.

Η μεταφορά μας στην σελίδα ενός συγκεκριμένου project υλοποιείται με την χρήση της ενέργειας *onRowSelected* που προσφέρει το 'Table' widget. Με την επιλογή λοιπόν μίας γραμμής του πίνακα, εκτελούμε την μέθοδο *navigateToProject* η οποία φαίνεται στην Εικόνα 35.

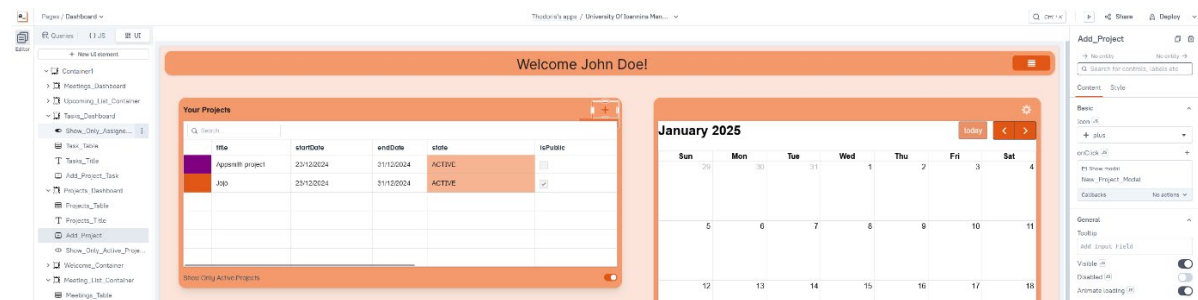
```

72 //Get new projectID and check its not empty
73 await getProjectIDByName.run({newProjectName: New_Project_title.text});
74 if (getProjectIDByName.data.length === 0) {
75   throw new Error('Project ID not found after creation. ');
76 }
77
78 //Insert the relation of Project-Current User in ProjectUsers table
79 await addProjectToCurrentUser.run({newProjectID: getProjectIDByName.data[0].projectID});
80
81 //Refresh the relevant tables data
82 await getCurrentUserProjects.run();
83 await getCurrentUserActiveProjects.run();
84 await getCurrentUserProjectTasks.run();
85
86 //On success of both insert queries
87 this.closeModalAfterProjectCreation();
88 showAlert('New project created!', 'success');
89 }
90 catch(error){
91   console.error('Error during project creation: ', error.message);
92   showAlert(error.message + " : " + createNewProject.data, 'error');
93 }
94
95 //If the createNewProject run, delete the new entry from the database
96 if (createNewProject.data[0].affectedRows == 1) {
97   await deleteProjectById.run({projectID: getProjectIDByName.data[0].projectID});
98 }
99
100 return error.message + " : " + createNewProject.data;
101 }
102 },
103
104 closeModalAfterProjectCreation: async () => {
105   closeModal(New_Project_Modal.name);
106
107 //Reset all fields of new project modal
108 New_Project_title.setValue('');
109 New_Project_Description.setValue('');
110 New_Project_Start_Date.setValue(moment().toString());
111 New_Project_End_Date.setValue('');
112 },
113
114 navigateToProject: async () => {
115   await navigateTo('Project', {projectID: Projects_Table.selectedRow.projectID,
116     ownerID: Projects_Table.selectedRow.ownerID,
117     title: Projects_Table.selectedRow.title,
118     description: Projects_Table.selectedRow.description,
119     startDate: Projects_Table.selectedRow.startDate,
120     endDate: Projects_Table.selectedRow.endDate,
121     projectColor: Projects_Table.selectedRow.projectColor,
122     state: Projects_Table.selectedRow.state,
123     isPublic: Projects_Table.selectedRow.isPublic});
124 }
125 }

```

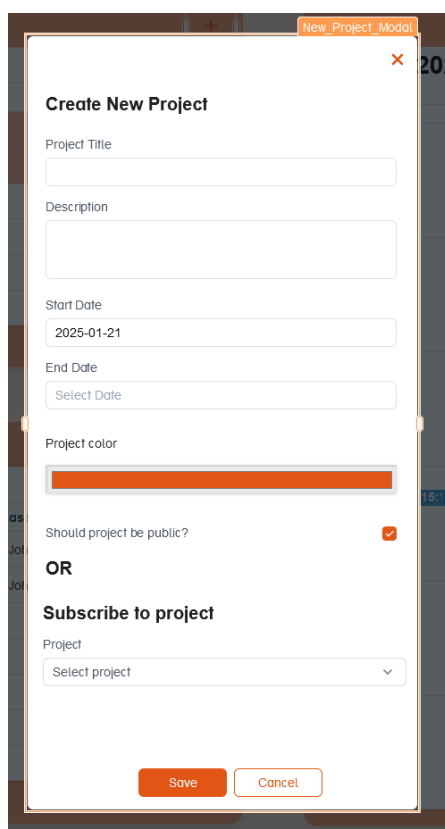
Εικόνα 35. Η μέθοδος *navigateToProject* που υλοποιεί την μετάβαση μας στην σελίδα του επιλεγμένου έργου. Η μέθοδος βρίσκεται στο αρχείο *Project_functions.js* που έχουμε δημιουργήσει για διαχείριση των λειτουργιών που αφορούν τα *project* στην σελίδα 'Dashboard'.

Τέλος ας δούμε πως υλοποιείται η διαδικασία δημιουργίας ενός νέου project από τη σελίδα «Dashboard».



Εικόνα 36. Διαθέσιμες επιλογές για το widget 'Icon Button'.

Χρησιμοποιώντας τη διαθέσιμη ενέργεια *onClick* που παρέχεται από το widget *Icon Button* όπως φαίνεται και στην Εικόνα 36, μπορούμε να καθορίσουμε την εκτέλεση διαφόρων ενεργειών, όπως: εκτέλεση μεθόδου JavaScript, εκτέλεση συγκεκριμένου query, άνοιγμα ενός *Modal* και άλλες λειτουργίες. Στην προκειμένη περίπτωση, έχουμε επιλέξει την ενέργεια ανοίγματος ενός *Modal*. Το *Modal* που εμφανίζεται, το οποίο φαίνεται στην Εικόνα 37 λειτουργεί ως μια φόρμα, περιλαμβάνοντας όλα τα απαραίτητα πεδία για τον χρήστη ώστε είτε να δημιουργήσει ένα νέο έργο είτε να εγγραφεί σε κάποιο υπάρχον που έχει ήδη δημιουργηθεί στην εφαρμογή.



Εικόνα 37. Το *Modal* που ανοίγει για την δημιουργία ή εγγραφή σε ένα *project*.

Το συγκεκριμένο *Modal* widget περιέχει έτοιμα στοιχεία όπως *Input*, *Select* και *DatePicker* αλλά και ένα *Custom* widget που έχουμε υλοποιήσει για την επιλογή χρώματος του νέου έργου. Με την συμπλήρωση των απαραίτητων πεδίων στην φόρμα και το πάτημα του *Save* εκτελείτε η JavaScript μέθοδος που φαίνεται στην Εικόνα 38.


```

15 createOrSubscribeToProject: async () => {
16   if(Select_New_Project.selectedOptionLabel.length > 0){
17     this.subscribeToProject();
18   }
19   else if(New_Project_title.isValid) {
20     this.createNewProject();
21   }
22   else {
23     console.error('Project title is empty or not valid based on field conditions. ');
24     showAlert('Project title is empty or not valid.', 'error');
25   }
26 },
27

```

Εικόνα 38. Μέθοδος επιλογής μεταξύ δημιουργίας ενός καινούργιου project η εγγραφής σε ένα υπάρχον.

Η επιλογή μεταξύ εγγραφής ή δημιουργίας γίνεται μέσω ελέγχου της τιμής που έχει επιλεγεί στο 'Select' widget που βρίσκεται στην φόρμα. Σε περίπτωση που δεν έχει επιλεγεί κάποια τιμή, εκτελείται η μέθοδος *createNewProject* της οποίας ο κώδικας φαίνεται στην Εικόνα 39:

```

55 createNewProject: async () => {
56   try{
57     //Check what status the project must be
58     const currentDate = new Date();
59     const startDate = new Date(New_Project_Start_Date.formattedDate);
60     const endDate = new Date(New_Project_End_Date.formattedDate);
61
62     if(startDate >= currentDate){
63       await createNewProject.run({projectState: 'INACTIVE'});
64     }
65     else if((startDate <= currentDate && endDate >= currentDate) || New_Project_End_Date.formattedDate == '') {
66       await createNewProject.run({projectState: 'ACTIVE'});
67     }
68     else if(startDate <= currentDate && endDate < currentDate){
69       await createNewProject.run({projectState: 'COMPLETED'});
70     }
71
72     //Get new projectID and check its not empty
73     await getProjectIdByName.run({newProjectName: New_Project_title.text});
74     if (getProjectIdByName.data.length == 0) {
75       throw new Error('Project ID not found after creation. ');
76     }
77
78     //Insert the relation of Project-Current User in ProjectUsers table
79     await addProjectToCurrentUser.run({newProjectID: getProjectIdByName.data[0].projectID});
80
81     //Refresh the relevant tables data
82     await getCurrentUserProjects.run();
83     await getCurrentUserActiveProjects.run();
84     await getCurrentUserProjectTasks.run();
85
86     //On success of both insert queries
87     this.closeModalAfterProjectCreation();
88     showAlert('New project created!', 'success');
89   }
90   catch(error){
91     console.error('Error during project creation: ', error.message);
92     showAlert(error.message + " : " + createNewProject.data, 'error');
93
94     //If the createNewProject run, delete the new entry from the database
95     if (createNewProject.data[0].affectedRows == 1) {
96       await deleteProjectById.run({projectID: getProjectIdByName.data[0].projectID});
97     }
98
99     return error.message + " : " + createNewProject.data;
100   }
101 },

```

Εικόνα 39. Μέθοδος *createNewProject* που χρησιμοποιείται για την δημιουργία ενός καινούργιου project.

Η μέθοδος καλεί το query *createNewProject* περνώντας του την κατάλληλη παράμετρο σε σχέση με την τιμή του state πεδίου ανάλογα με τις ημερομηνίες έναρξης και λήξης. Στην συνέχεια ανακτούμε το *projectID* του νέου έργου και προσθέτουμε τον χρήστη που το δημιούργησε στον πίνακα *ProjectUsers* με χρήση του query *addProjectToCurrentUser* που φαίνεται στην Εικόνα 40. Τέλος ανανεώνουμε τα δεδομένα των διάφορων σχετικών πινάκων τρέχοντας τα κατάλληλα queries και κλείνουμε το 'Modal' ολοκληρώνοντας τη διαδικασία δημιουργίας του νέου έργου.



```
1 INSERT INTO ProjectUsers (projectID, userID)
2 VALUES ('{{this.params.newProjectID}}', '{{appsmith.store.user.userID}}');
3
```

Εικόνα 40. Ο κώδικας του `addProjectToCurrentUser` query που προσθέτει τον δημιουργό του νέου `project` ως συμμετέχοντα σε αυτό.

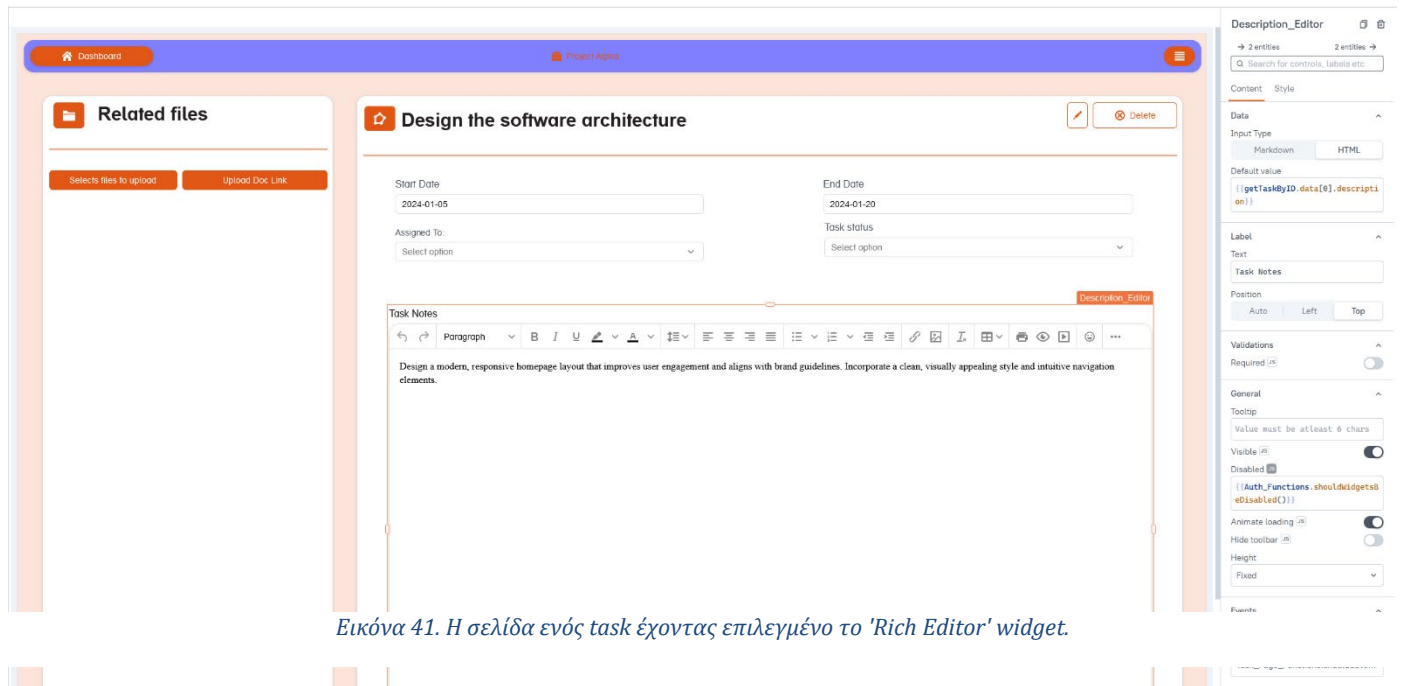
Παρουσιάσαμε, λοιπόν, αναλυτικά τον τρόπο υλοποίησης των λειτουργιών που σχετίζονται με τα έργα στη σελίδα «Dashboard», αναδεικνύοντας το σχετικό περιβάλλον ανάπτυξης και τον αντίστοιχο κώδικα JavaScript και SQL. Με παρόμοιες μεθόδους και την ίδια φιλοσοφία, υλοποιήθηκαν και οι λειτουργίες που αφορούν τα υπόλοιπα αντικείμενα της εφαρμογής, όπως η δημιουργία εργασιών και συναντήσεων, καθώς και η πλοήγηση στις αντίστοιχες σελίδες τους. Η προσέγγιση αυτή εξασφαλίζει συνέπεια στην ανάπτυξη, ευκολία στη διαχείριση του κώδικα και αποτυπώνει τον τρόπο χρήσης της πλατφόρμας για την επίτευξη των επιθυμητών λειτουργιών της εφαρμογής.

5.4 Υλοποίηση των user role

Συνεχίζοντας ως δούμε με ποιον τρόπο υλοποιήσαμε τον έλεγχο πρόσβασης των χρηστών. Όπως ήδη έχουμε αναφέρει έχουμε υλοποιήσει τρία επίπεδα χρηστών: Admin, User και Guest. Για την υλοποίηση της συγκεκριμένης λειτουργίας, εκμεταλλευτήκαμε τις δυνατότητες που μας δίνει το Appsmith για τον χειρισμό των widget με χρήση JavaScript. Συγκεκριμένα, χρησιμοποιήσαμε τα πεδία 'Disabled' και 'Visible' τα οποία επιτρέπουν την απόκρυψη ή απενεργοποίηση των widget.

Μέσα από τη χρήση αυτών των πεδίων, μπορούμε να αποκρύπτουμε ή να απενεργοποιούμε συγκεκριμένα *widgets* για διαφορετικούς τύπους χρηστών, εξασφαλίζοντας έτσι έναν ευέλικτο και ασφαλή έλεγχο πρόσβασης. Αυτή η προσέγγιση εγγυάται ότι κάθε χρήστης έχει πρόσβαση μόνο στις λειτουργίες που του αντιστοιχούν, προσφέροντας παράλληλα μια πιο εξατομικευμένη εμπειρία χρήσης.

Ας εξετάσουμε πιο αναλυτικά την υλοποίηση μέσα από τη σελίδα μίας εργασίας, επιλέγοντας συγκεκριμένα *widgets* της σελίδας και παρουσιάζοντας τον τρόπο με τον οποίο διαχειριστήκαμε τα πεδία για την υλοποίηση της λειτουργία των δικαιωμάτων πρόσβασης των χρηστών.



Εικόνα 41. Η σελίδα ενός task έχοντας επιλεγμένο το 'Rich Editor' widget.

Για την ανάπτυξη αυτής της λειτουργικότητας, υλοποιήσαμε δύο βασικές μεθόδους: μία για το πεδίο 'Disabled', με την ονομασία *shouldWidgetsBeDisabled*, και μία για το πεδίο 'Visible', με την ονομασία *shouldWidgetBeVisible*.

Στην Εικόνα 42, παρατηρούμε ότι στο πεδίο 'Disabled' του 'Rich editor' widget, χρησιμοποιούμε την μέθοδο *shouldWidgetsBeDisabled*. Η συγκεκριμένη μέθοδος, της οποίας ο κώδικας φαίνεται στην Εικόνα 42, επιστρέφει false, δηλαδή το widget μας πρέπει να είναι ενεργοποιημένο, στην περίπτωση που ο συνδεδεμένος χρήστης είναι Admin ή είναι User και του έχει ανατεθεί η συγκεκριμένη εργασία. Σε όλες τις υπόλοιπες περιπτώσεις η μέθοδος επιστρέφει true με αποτέλεσμα την απενεργοποίηση του 'Rich Editor' widget.

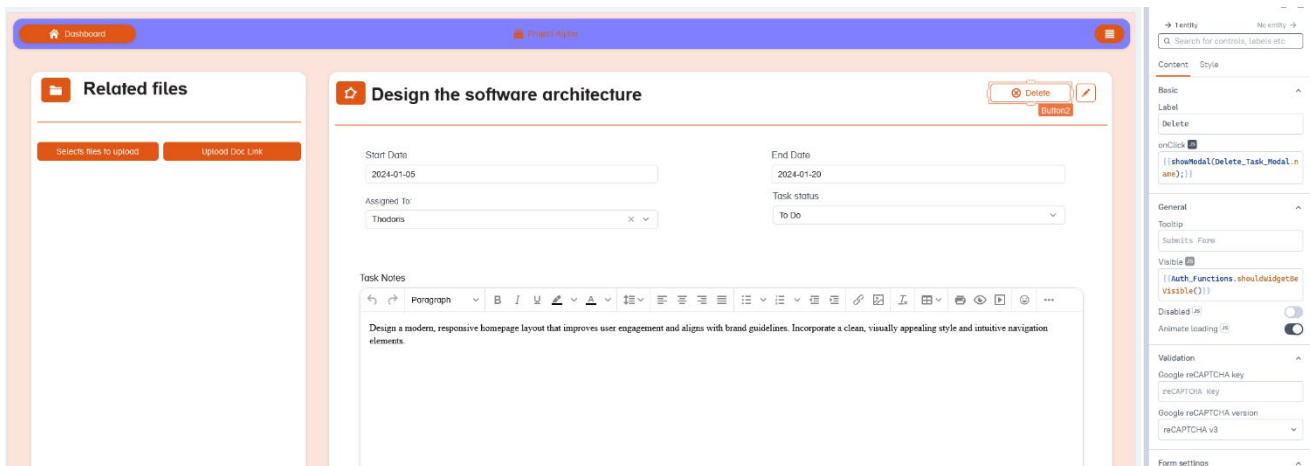
```

20
21 ▾ shouldWidgetsBeDisabled() {
22   if(appsmith.store.user.userCategory == appsmith.store.userRights['admin'] ||
23     (appsmith.store.user.userCategory == appsmith.store.userRights['user'] && getTaskByID.data[0].assignedUserID == appsmith.store.user.userID)){
24
25     return false;
26   }
27 ▾   else{
28     return true;
29   }
30 },
31
32 ▾ shouldWidgetBeVisible() {
33 ▾   if(appsmith.store.user.userCategory == appsmith.store.userRights['admin']) {
34     return true;
35   }
36 ▾   else{
37     return false;
38   }
39 },
40

```

Εικόνα 42. Ο κώδικας των μεθόδων *shouldWidgetBeDisabled* και *shouldWidgetBeVisible* απο την σελίδα 'Task'.

Με παρόμοιο τρόπο χρησιμοποιείται και η μέθοδος *shouldWidgetBeVisible* η οποία χρησιμοποιείται στα widget που αφορούν την διαγραφή και την αλλαγή του ονόματος της εργασίας.



Εικόνα 43. Τα πεδία του Delete κουμπιού στην Task σελίδα.

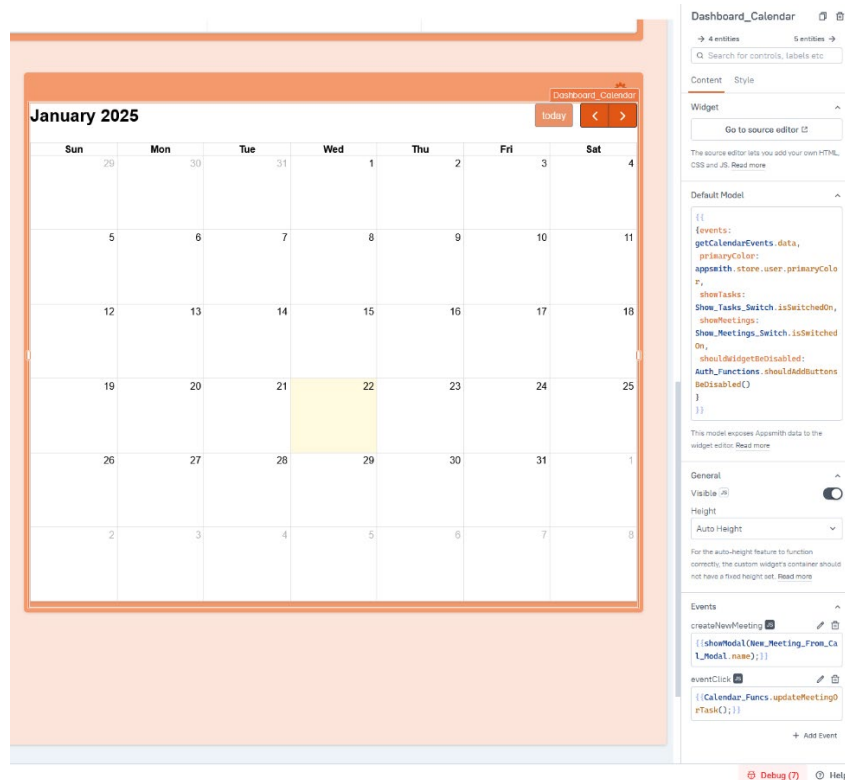
Όπως φαίνεται στην Εικόνα 44, πλέον χρησιμοποιούμε το πεδίο 'Visible' και την μέθοδο *shouldWidgetBeVisible* επιτυγχάνοντας την απόκρυψη του Delete κουμπιού σε περίπτωση που ο συνδεδεμένος χρήστης δεν είναι Admin.

Η παραπάνω μεθοδολογία εφαρμόστηκε σε όλα τα διαφορετικά widget τα οποία θέλαμε να συμπεριλάβουμε στην ελεγχόμενη πρόσβαση με βάση την κατηγορία του συνδεδεμένου χρήστη. Σε κάθε σελίδα της εφαρμογής υλοποιήθηκαν παρόμοιες μέθοδοι, όπως οι *shouldWidgetsBeDisabled* και *shouldWidgetBeVisible*, οι οποίες προσαρμόστηκαν στις ιδιαίτερες ανάγκες και απαιτήσεις της εκάστοτε σελίδας. Με αυτόν τον τρόπο εξασφαλίστηκε η σωστή και δυναμική διαχείριση δικαιωμάτων πρόσβασης, προσφέροντας μία ευέλικτη και ασφαλή εμπειρία χρήσης.

5.5 Υλοποίηση ημερολογίου με χρήση custom widget

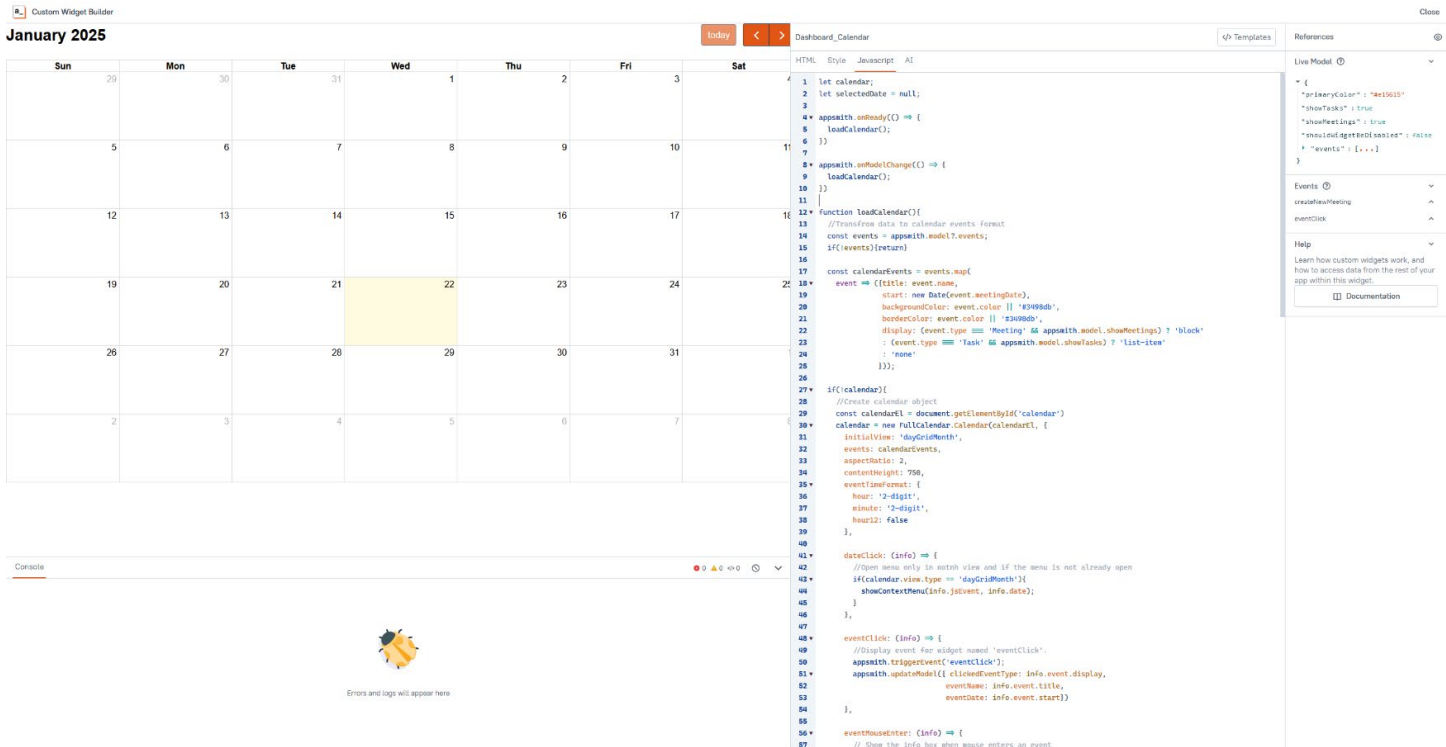
Το Appsmith, με τη δυνατότητα που προσφέρει για υλοποίηση custom widgets, επεκτείνει σημαντικά τις δυνατότητες του εργαλείου. Για να κατανοήσουμε πώς λειτουργούν αυτά τα widgets, θα αναλύσουμε τη δική μας υλοποίηση σχετικά με το ημερολόγιο. Δεδομένου ότι το Appsmith δεν περιλάμβανε κάποιο έτοιμο widget ημερολογίου, προχωρήσαμε στη δημιουργία ενός προσαρμοσμένου widget.

Όπως ισχύει και για όλα τα έτοιμα στοιχεία του Appsmith, για να ξεκινήσουμε την υλοποίηση μας πρέπει να προσθέσουμε στο γραφικό περιβάλλον μέσω drag-and-drop το στοιχείο με όνομα 'Custom'. Επιλέγοντας το μπορούμε να δούμε τα διαθέσιμα πεδία του όπως 'Visible' και 'Height'. Μια πρόσθετη επιλογή που διατίθεται στα custom widgets είναι το κουμπί 'Go to source editor' η οποία μας μεταφέρει στο περιβάλλον ανάπτυξης ενός custom widget.



Εικόνα 44. Διαθέσιμες επιλογές ενός 'Custom' widget. Στην συγκεκριμένη περίπτωση έχουμε επιλεγμένο το ημερολόγιο που έχουμε υλοποιήσει.

Η υλοποίηση ενός custom widget απαιτεί από μέρους μας την ανάπτυξη κώδικα σε HTML, CSS και JavaScript για τη δημιουργία της διεπαφής και της λειτουργικότητάς του. Αυτή η προσέγγιση προσομοιάζει περισσότερο την ανάπτυξη παραδοσιακών εφαρμογών παρά τη χρήση ενός low-code framework.



Εικόνα 45. Το περιβάλλον ανάπτυξης του ημερολογίου 'Custom' widget που υλοποιήσαμε.

Αρχικά, η διασύνδεση των δεδομένων μεταξύ του custom widget και του Appsmith γίνεται εφικτή μέσω του πεδίου 'Default Model' που φαίνεται στην Εικόνα 44, όπου μας επιτρέπει να περάσουμε τα απαραίτητα δεδομένα όπως αποτελέσματα από queries, στο περιβάλλον του widget. Η προσπέλαση τους γίνεται εφικτή μέσω του αντικειμένου *appsmith.model* που προσφέρεται από το framework.

Για την ανάπτυξη του ημερολογίου, επιλέξαμε τη βιβλιοθήκη *FullCalendar.js*, η οποία παρέχει πλούσιες δυνατότητες διαχείρισης ημερολογιακών δεδομένων. Ακολουθώντας το επίσημο documentation της βιβλιοθήκης, δημιουργούμε ένα αντικείμενο FullCalendar και παραμετροποιούμε τις διαθέσιμες ρυθμίσεις του σύμφωνα με τις ανάγκες μας.

Μια ιδιαιτερότητα στα custom widgets του Appsmith είναι η χρήση των *onReady* και *onModelChange* που παρέχει. Η μέθοδος *onReady* αναμένει την ετοιμότητα του περιβάλλοντος του Appsmith πριν εκτελεστεί οποιοσδήποτε κώδικας του custom widget. Έτσι η χρήση της είναι απαραίτητη για την εκτέλεση του κώδικα του widget μας. Θα μπορούσαμε να την παρομοιάσουμε με την μέθοδο *main* της γλώσσας προγραμματισμού Java. Από την άλλη η *onModelChange* εκτελείται όποτε έχουμε κάποια αλλαγή στα δεδομένα που περνάμε στο widget, μέσω του πεδίου 'Default Model'. Για παράδειγμα εάν

η τιμή του attribute `'showMeetings'` που βλέπουμε στην Εικόνα 44 αλλάξει, τότε θα εκτελεστεί η συγκεκριμένη μέθοδος.

Τέλος αξίζει να αναφέρουμε και τον τρόπο υλοποίησης `trigger events` και πως γίνεται η επικοινωνία με το περιβάλλον του Appsmith. Παίρνοντας ως παράδειγμα την λειτουργία `update` πατώντας ένα event του ημερολογίου, ας παρατηρήσουμε την μέθοδο `eventClick` στην Εικόνα 46.

Με την χρήση του `appsmith.triggerEvent("eventClick")` ουσιαστικά δημιουργούμε το συγκεκριμένο event στο περιβάλλον του Appsmith και μπορούμε να εκτελέσουμε ενέργειες όπως και στο event `onClick` που είδαμε σε προηγούμενα έτοιμα widget του Appsmith. Δηλαδή μπορούμε να ανοίξουμε ένα 'Modal', να εκτελέσουμε μία JavaScript μέθοδο κτλ. Στην συγκεκριμένη περίπτωση με το κλικ σε ένα event του ημερολογίου ανοίγουμε ένα 'Modal' που λειτουργεί ως φόρμα για την ανανέωση των πεδίων του event.

Το πέρασμα των επιθυμητών δεδομένων στο περιβάλλον του Appsmith γίνεται με χρήση της `appsmith.updateModel()` μεθόδου. Στην συγκεκριμένη περίπτωση βλέπουμε πως περνάμε δεδομένα που αφορούν το event που κάναμε κλικ όπως το όνομα του, την ημερομηνία του και τον τύπο εμφάνισης του (με αυτόν τον τρόπο διαφοροποιούμε τα events που αφορούν τις εργασίες από των συναντήσεων).

Με παρόμοια μεθοδολογία και προσέγγιση, υλοποιήθηκαν όλα τα custom widgets της εφαρμογής μας. Χρησιμοποιώντας κυρίως έτοιμες βιβλιοθήκες που προσφέρουν τη βασική λειτουργικότητα που θέλουμε να επιτύχουμε και με ενσωμάτωσης τους με το Appsmith καταφέραμε να καλύψουμε ανάγκες που το framework δεν υποστήριζε εγγενώς. Άλλα παραδείγματα custom widgets στην εφαρμογή μας περιλαμβάνουν το «Kanban board», τον επιλογέα χρωμάτων (colour picker) καθώς και τη δενδροειδή απεικόνιση των αρχείων που συναντάμε στην σελίδα «Project Dashboard».

Συμπεραίνουμε, λοιπόν, ότι η δυνατότητα δημιουργίας custom widgets καθιστά το Appsmith εξαιρετικά ευέλικτο και αποτέλεσε καθοριστικό παράγοντα για την υλοποίηση των λειτουργιών που απαιτούσαμε από το εργαλείο.

Κεφάλαιο 6 - Συμπεράσματα

6.1 Αντικείμενο της εργασίας

Η παρούσα διπλωματική εργασία επικεντρώνεται στην ανάπτυξη ενός ολοκληρωμένου συστήματος διαχείρισης ακαδημαϊκών έργων, το οποίο προορίζεται να καλύψει τις εξειδικευμένες ανάγκες των ακαδημαϊκών ομάδων. Ο βασικός στόχος ήταν η δημιουργία ενός εργαλείου που να διευκολύνει την οργάνωση, τη διαχείριση και την παρακολούθηση της προόδου διαφόρων έργων, όπως διπλωματικές εργασίες, ερευνητικά προγράμματα και ομαδικά έργα.

Η εργασία εστιάζει στη χρήση ενός low-code περιβάλλοντος ανάπτυξης, συγκεκριμένα της πλατφόρμας Appsmith, ώστε να επιτευχθεί ένας συνδυασμός ευελιξίας, ευχρηστίας και ταχύτητας στην υλοποίηση της επιθυμητής εφαρμογής. Μέσα από την χρήση ενός τέτοιου εργαλείου, επιχειρήθηκε να παραχθεί μια λύση που όχι μόνο ανταποκρίνεται στις άμεσες ανάγκες της ακαδημαϊκής κοινότητας αλλά και διατηρεί τη δυνατότητα μελλοντικής προσαρμογής και επέκτασης.

Αφετηρία της εργασίας ήταν η διαπίστωση ότι τα υπάρχοντα λογισμικά διαχείρισης έργων συχνά δεν ανταποκρίνονται στις ιδιαιτερότητες των ακαδημαϊκών περιβαλλόντων, είτε λόγω υπερβολικής πολυπλοκότητας είτε λόγω έλλειψης των απαιτούμενων λειτουργιών. Με βάση αυτή την ανάγκη, η εργασία επιδιώκει να καλύψει το κενό με την ανάπτυξη ενός ειδικά προσαρμοσμένου εργαλείου, το οποίο ενσωματώνει λειτουργίες όπως η δημιουργία και διαχείριση έργων, η ανάθεση ρόλων, η παρακολούθηση χρονοδιαγραμμάτων και η ενίσχυση της συνεργασίας μεταξύ των μελών των ακαδημαϊκών ομάδων.

Με αυτόν τον τρόπο, η εργασία συμβάλλει ουσιαστικά στην υποστήριξη της ακαδημαϊκής διαχείρισης έργων, παρέχοντας μια λύση που συνδυάζει την καινοτομία της τεχνολογίας low-code μέσω μιας πρακτικής εφαρμογής που καλύπτει πραγματικές ανάγκες.

6.2 Περιγραφή υλοποίησης

Για την υλοποίηση της παρούσας εργασίας, χρησιμοποιήθηκε η πλατφόρμα Appsmith, ένα low-code περιβάλλον ανάπτυξης εφαρμογών που επιτρέπει τη δημιουργία εσωτερικών εργαλείων με ευκολία και ταχύτητα. Η επιλογή του Appsmith βασίστηκε στη δυνατότητά του να συνδυάζει την απλότητα χρήσης με την ευελιξία της παραμετροποίησης μέσω κώδικα, καλύπτοντας πλήρως τις ανάγκες που επιθυμούσαμε να καλύψουμε με την εφαρμογής μας.

Η διαδικασία ανάπτυξης ξεκίνησε με τον καθορισμό των απαιτήσεων της εφαρμογής, οι οποίες περιλάμβαναν τη διαχείριση έργων, την ανάθεση ρόλων, τη διαχείριση συναντήσεων, τη δημιουργία αυτόματων ειδοποιήσεων και τη διαχείριση αρχείων.

Για την επίτευξη των λειτουργιών της εφαρμογής, ξεκινήσαμε με τη δημιουργία της βάσης δεδομένων, όπου σχεδιάσαμε και υλοποιήσαμε τους βασικούς πίνακες για τις κύριες οντότητες που έπρεπε να διαχειριστούμε (Projects, Tasks, Meetings). Η ανάπτυξη της βάσης δεδομένων πραγματοποιήθηκε σταδιακά, με την προσθήκη επιπλέον πινάκων ανάλογα με τις ανάγκες που προέκυπταν κατά την ανάπτυξη συγκεκριμένων λειτουργιών, εξασφαλίζοντας τη δομική ευελιξία και την επεκτασιμότητα.

Η υλοποίηση του γραφικού περιβάλλοντος και της λογικής των λειτουργιών της εφαρμογής πραγματοποιήθηκε εξολοκλήρου μέσω της πλατφόρμας Appsmith. Με την αξιοποίηση των ενσωματωμένων widget, των custom widget και των προηγμένων δυνατοτήτων παραμετροποίησης που προσφέρει η πλατφόρμα, καταφέραμε να ικανοποιήσουμε τις απαιτήσεις που θέσαμε, παρέχοντας ένα λειτουργικό και ευχάριστο περιβάλλον χρήστη.

Για τη διαχείριση των αρχείων, επιλέξαμε να αναπτύξουμε έναν custom file server, με χρήση του nginx, δεδομένης της έλλειψης πλήρως ενσωματωμένης λύσης στο Appsmith για τη συγκεκριμένη λειτουργία. Ο file server, που φιλοξενείται σε δικούς μας διακομιστές, εξυπηρετεί τις λειτουργίες αποθήκευσης και διαγραφής αρχείων, σε συνεργασία με την βάση δεδομένων μας, προσφέροντας μεγαλύτερο έλεγχο και ευελιξία στις διαδικασίες διαχείρισης αρχείων.

Το εσωτερικό μας εργαλείο αναπτύχθηκε με γνώμονα τη φιλικότητα προς τον χρήστη, διατηρώντας μια καθαρή και οργανωμένη διάταξη. Η ευχρηστία της εφαρμογής ενισχύθηκε περαιτέρω μέσω επιλογών παραμετροποίησης, όπως η αλλαγή χρωμάτων και η ενεργοποίηση/απενεργοποίηση widget ανάλογα με τις προτιμήσεις του χρήστη. Η συνολική προσέγγιση εστίασε στην επίτευξη ενός ισορροπημένου αποτελέσματος, συνδυάζοντας την τεχνολογική καινοτομία με την πρακτική χρηστικότητα, ώστε να παραχθεί ένα εργαλείο που να ανταποκρίνεται πλήρως στις ανάγκες των ακαδημαϊκών ομάδων.

6.3 Προβλήματα και προκλήσεις

Κατά την υλοποίηση της παρούσας εργασίας, αντιμετωπίστηκαν διάφορα προβλήματα και προκλήσεις που σχετίζονται τόσο με τεχνικές πτυχές όσο και με ζητήματα σχεδιασμού και προσαρμογής. Οι κυριότερες από αυτές ήταν:

- **Εκμάθηση και Εξοικείωση με το Appsmith:** Παρά το φιλικό περιβάλλον του Appsmith, χρειάστηκε χρόνος για να κατανοηθούν πλήρως οι δυνατότητες και οι περιορισμοί του. Η εκμάθηση της πλατφόρμας και η ανακάλυψη των βέλτιστων πρακτικών για την υλοποίηση συγκεκριμένων απαιτήσεων ήταν μια διαρκής διαδικασία καθ' όλη την διάρκεια ανάπτυξης της εφαρμογής.
- **Διαχείριση Πολυπλοκότητας:** Η σταδιακή ανάπτυξη της εφαρμογής, με την προσθήκη νέων λειτουργιών και την κάλυψη αυξανόμενων απαιτήσεων, οδήγησε σε σημαντική αύξηση της πολυπλοκότητας. Οι συνεχείς προσθήκες queries και JavaScript μεθόδων κατέστησαν απαραίτητα τα συχνά refactoring, ώστε να επιτευχθεί ενοποίηση λειτουργιών, μείωση της πλεονάζουσας λογικής

και διαγραφή περιττού κώδικα. Αυτή η διαδικασία συνέβαλε στη διατήρηση της απόδοσης και της ευελιξίας της εφαρμογής.

- **Περιορισμοί της πλατφόρμας Appsmith:** Παρότι το Appsmith προσφέρει εκτεταμένες δυνατότητες παραμετροποίησης, ορισμένες ειδικές ανάγκες της εφαρμογής δεν ήταν δυνατόν να καλυφθούν με τις υπάρχουσες έτοιμες λύσεις. Η χρήση custom widget αποτέλεσε τη λύση για την υλοποίηση αυτών των λειτουργιών, αυξάνοντας όμως την πολυπλοκότητα της ανάπτυξης. Για την υλοποίηση των custom widget, απαιτήθηκε έρευνα και μελέτη των κατάλληλων βιβλιοθηκών, γεγονός που προσέθεσε επιπλέον χρόνο και προσπάθεια στο συνολικό έργο.

Παρά τις δυσκολίες και της προκλήσεις που αντιμετωπίσαμε η εμπειρία που αποκτήθηκε ήταν εξαιρετικά πολύτιμη, ενώ η αντιμετώπιση αυτών των προβλημάτων συνέβαλε στη βελτίωση της τελικής εφαρμογής και στην απόκτηση μιας βαθύτερης κατανόησης των διάφορων τεχνολογιών που χρησιμοποιήθηκαν.

6.4 Μελλοντικές επεκτάσεις

Η παρούσα έκδοση της εφαρμογής που παρουσιάζουμε θεωρούμε πως καλύπτει στο έπακρο τις βασικές ανάγκες που θα έχει η εκάστοτε ομάδα για την αποδοτική διαχείριση των έργων της. Ωστόσο υπάρχουν αρκετές δυνατότητες για μελλοντικές επεκτάσεις και βελτιώσεις που μπορούν να ενισχύσουν περαιτέρω τη λειτουργικότητα και τη χρηστικότητα της εφαρμογής. Κάποιες από αυτές είναι:

- **User-level παραμετροποίηση των Dashboard:** Η υλοποίηση μας σχετικά με την εμφάνιση ή όχι κάποιων widget στις σελίδες «Dashboard» και «Project Dashboard» είναι προς το παρόν περιορισμένη στην συνεδρία (session) του χρήστη. Δηλαδή με την έξοδο του από την εφαρμογή οι προτιμήσεις του χάνονται. Μια σημαντική βελτίωση θα ήταν η αποθήκευση των προτιμήσεων αυτών στη βάση δεδομένων, επιτρέποντας την αυτόματη ανάκτησή τους κάθε φορά που ο χρήστης συνδέεται. Αυτό θα προσέφερε μια πιο εξατομικευμένη και διαχρονική εμπειρία χρήστη.
- **Project file manager widget:** Παρότι έχουμε υλοποιήσει μία δενδροειδή απεικόνιση των αρχείων στην σελίδα «Project Dashboard» η λειτουργικότητα του περιορίζεται μόνο στην προβολή και την λήψη των αρχείων. Στόχος είναι η βελτίωση αυτής της λειτουργίας, ώστε να προσφέρει μια πιο διαδραστική εμπειρία. Οι βελτιώσεις περιλαμβάνουν την οργάνωση των αρχείων ανά αντικείμενο στο οποίο ανήκουν και την προσθήκη δυνατότητας ανεβάσματος αρχείων απευθείας μέσω του widget.
- **Δυναμική διαχείριση καταστάσεων εργασιών ανά project:** Όπως έχει ήδη αναφερθεί, οι καταστάσεις που αφορούν τις εργασίες αποθηκεύονται σε έναν ξεχωριστό πίνακα στη βάση δεδομένων, διευκολύνοντας την επεκτασιμότητα και τη διαχείρισή τους. Ωστόσο, αυτή τη στιγμή δεν υπάρχει δυνατότητα δημιουργίας ή διαγραφής καταστάσεων μέσα από την εφαρμογή. Οι σχετικές αλλαγές μπορούν να πραγματοποιηθούν μόνο μέσω queries που εκτελούνται απευθείας στη βάση δεδομένων. Στόχος για το μέλλον είναι η ενσωμάτωση

λειτουργικότητας στο «Kanban board» της σελίδας «Project Dashboard», ώστε οι χρήστες να μπορούν να προσθέτουν ή να αφαιρούν καταστάσεις. Αυτή η δυνατότητα θα επιτρέπει την ευέλικτη προσαρμογή των καταστάσεων στις ανάγκες του κάθε έργου, ενισχύοντας την αποδοτικότητα και την ευχρηστία της εφαρμογής.

Οι παραπάνω βελτιώσεις θεωρούμε πως μπορούν να συμβάλουν σημαντικά στην συνολική εμπειρία του χρήστη, καθιστώντας την εφαρμογή πιο ευέλικτη και φιλική προς τις ανάγκες του. Η προσθήκη τέτοιων λειτουργιών όχι μόνο ενισχύει την αποτελεσματικότητα αλλά δημιουργεί και τις βάσεις για περαιτέρω εξέλιξη της εφαρμογής.

Βιβλιογραφία - Πηγές

- [1] T. a. M. D. Sedej, «The optimal selection of internal communication tools,» *Int. J. Globalisation and Small Business*, Vol. 7, No. 1,, p. 6–25, 2015.
- [2] J. Highsmith, *Agile Project Management: Creating Innovative Products*, 2004.
- [3] M. Cohn, *Agile Estimating and Planning*, 2005.
- [4] «What are internal tools?,» Budibase, 2025. [Ηλεκτρονικό]. Available: <https://budibase.com/internal-tools/>.
- [5] «Top 5 Open Source Projects for Building Internal Tools,» Nocobase, 2024. [Ηλεκτρονικό]. Available: <https://www.nocobase.com/en/blog/top-5-open-source-projects-for-building-internal-tools-in-2024>.
- [6] «Project management software,» Wikipedia, [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Project_management_software.
- [7] A. S. A. P. S. R. K. V. M. Yagyamitra Chouhan, «Project Management Tool: A Review,» *IJS DR*, αρ. Volume 7 Issue 1, January 2022.
- [8] «Internal Software Development: Pros/Cons of Building Internal Tools,» Basedash, [Ηλεκτρονικό]. Available: <https://www.basedash.com/blog/internal-software-development-pros-cons-of-building-internal-tools>.
- [9] «Successful Internal Communications in Higher Education,» [Ηλεκτρονικό]. Available: <https://siliconreef.co.uk/guides/successful-internal-communications-in-higher-education/>.
- [10] «Asana Pricing,» 2025. [Ηλεκτρονικό]. Available: <https://asana.com/pricing>.
- [11] «Jira Pricing,» 2025. [Ηλεκτρονικό]. Available: <https://www.atlassian.com/software/jira/pricing>.
- [12] «Nocobase,» 2025. [Ηλεκτρονικό]. Available: <https://www.nocobase.com/>.
- [13] «Nocobase Plugins,» Nocobase, [Ηλεκτρονικό]. Available: <https://www.nocobase.com/en/plugins>.
- [14] «Budibase,» 2025. [Ηλεκτρονικό]. Available: <https://budibase.com/>.

- [15] «Conditional UI,» Budibase, 2025. [Ηλεκτρονικό]. Available: <https://docs.budibase.com/docs/conditions>.
- [16] «Refine,» 2025. [Ηλεκτρονικό]. Available: <https://refine.dev/>.
- [17] «Tooljet,» 2025. [Ηλεκτρονικό]. Available: <https://www.tooljet.com/>.
- [18] «ToolJet System Requirements,» ToolJet , 2025. [Ηλεκτρονικό]. Available: <https://docs.tooljet.com/docs/setup/system-requirements>.
- [19] «Appsmith,» 2025. [Ηλεκτρονικό]. Available: <https://www.appsmith.com/>.
- [20] «Upload Files to S3,» Appsmith, 2025. [Ηλεκτρονικό]. Available: <https://docs.appsmith.com/connect-data/how-to-guides/how-to-upload-to-s3>.
- [21] «ToolJet Workflows Overview,» ToolJet , 2025. [Ηλεκτρονικό]. Available: <https://docs.tooljet.com/docs/workflows/overview>.

Παράρτημα κώδικα

Κώδικας Εικόνα 24 – Μέθοδοι Log-in σελίδας

```
export default {
  smtpEmail: '',
  fileServerURL: '10.7.3.137:2081',
  defaultTab: 'Sign In',
  userRights: {
    admin: 'Admin',
    user: 'User',
    guest: 'Guest'
  },

  deleteStore: () => {
    //Reset fields
    inp_email.setValue('');
    inp_password.setValue('');

    //Remove everything from appsmith store
    Object.keys(appsmith.store).forEach(key => {
      storeValue(key, undefined);
    });
  },

  verifyHash: (password, hash) => {
    return dcodeIO.bcrypt.compareSync(password, hash);
  },

  createToken: async (user) => {
    return jsonwebtoken.sign(user, 'secret', {expiresIn: 60*60});
  },

  generateSecretKey() {
    const array = new Uint8Array(32); // 32 bytes = 256 bits (AES-256 strength)
    crypto.getRandomValues(array);
    return btoa(String.fromCharCode.apply(null, array));
  },

  signIn: async () => {
    const password = inp_password.text;
    const [user] = await findUserByEmail.run(inp_email.text);
```

```

if (user && this.verifyHash(password, user.passwordHash))
{
  storeValue('user', user);
  storeValue('userRights', this.userRights);
  storeValue('fileServerURL', this.fileServerURL);
  storeValue('smtpEmail', this.smtpEmail);
  storeValue('secretKey', this.generateSecretKey());
  storeValue('token', await this.createToken(user))
    .then(navigateTo('Dashboard'))
}
else
{
  return showAlert('Invalid email/password combination', 'error');
}
},
}

```

Κώδικας Εικόνα 34 – Queries *getCurrentUserProjects* και *getCurrentUserActiveProjects*

getCurrentUserProjects

```

SELECT Projects.*
FROM Projects
WHERE Projects.projectID IN (
  SELECT projectID
  FROM ProjectUsers
  WHERE userID = {{appsmith.store.user.userID}}
)
ORDER BY
CASE Projects.state
  WHEN 'ACTIVE' THEN 1
  WHEN 'INACTIVE' THEN 2
  WHEN 'COMPLETED' THEN 3
  ELSE 4
END,
Projects.endDate;

```

getCurrentUserActiveProjects

```
SELECT Projects.*
FROM Projects
WHERE Projects.projectID IN (
  SELECT projectID
  FROM ProjectUsers
  WHERE userID = {{appsmith.store.user.userID}}
  AND state = 'ACTIVE'
)
ORDER BY
CASE Projects.state
  WHEN 'ACTIVE' THEN 1
  WHEN 'INACTIVE' THEN 2
  WHEN 'COMPLETED' THEN 3
  ELSE 4
END,
Projects.endDate;
```

Κώδικας Εικόνα 35 – Μέθοδος *navigateToProject*

```
navigateToProject: async () => {
  await navigateTo('Project', {projectID: Projects_Table.selectedRow.projectID,
    ownerID: Projects_Table.selectedRow.ownerID,
    title: Projects_Table.selectedRow.title,
    description: Projects_Table.selectedRow.description,
    startDate: Projects_Table.selectedRow.startDate,
    endDate: Projects_Table.selectedRow.endDate,
    projectColor: Projects_Table.selectedRow.projectColor,
    state: Projects_Table.selectedRow.state,
    isPublic: Projects_Table.selectedRow.isPublic});
}
```


Κώδικας Εικόνα 38 – Μέθοδος *createOrSubscribeToProject*

```
createOrSubscribeToProject: async () => {
  if(Select_New_Project.selectedOptionLabel.length > 0){
    this.subscribeToProject();
  }
  else if(New_Project_title.isValid) {
    this.createNewProject();
  }
  else {
    console.error('Project title is empty or not valid based on field conditions.');
```

```
    showAlert('Project title is empty or not valid.', 'error');
```

```
  }
},
```

Κώδικας Εικόνα 39 – Μέθοδος *createNewProject*

```
createNewProject: async () => {
  try{
    //Check what status the project must be
    const currentDate = new Date();
    const startDate = new Date(New_Project_Start_Date.formattedDate);
    const endDate = new Date(New_Project_End_Date.formattedDate);

    if(startDate >= currentDate){
      await createNewProject.run({projectState: 'INACTIVE'});
    }
    else if((startDate <= currentDate && endDate >= currentDate) ||
New_Project_End_Date.formattedDate == '') {
      await createNewProject.run({projectState: 'ACTIVE'});
    }
    else if(startDate <= currentDate && endDate < currentDate){
      await createNewProject.run({projectState: 'COMPLETED'});
    }

    //Get new projectID and check its not empty
    await getProjectIdByName.run({newProjectName: New_Project_title.text});
    if (getProjectIdByName.data.length === 0) {
      throw new Error('Project ID not found after creation.');
```

```
    }
  }
}
```

```

//Insert the relation of Project-Current User in ProjectUsers table
await addProjectToCurrentUser.run({newProjectID:
getProjectIdByName.data[0].projectID});

//Refresh the relevant tables data
await getCurrentUserProjects.run();
await getCurrentUserActiveProjects.run();
await getCurrentUserProjectTasks.run();

//On success of both insert queries
this.closeModalAfterProjectCreation();
showAlert('New project created!', 'success');
}
catch(error){
console.error('Error during project creation: ', error.message);
showAlert(error.message + " : " + createNewProject.data, 'error');

//If the createNewProject run, delete the new entry from the database
if (createNewProject.data[0].affectedRows == 1) {
await deleteProjectById.run({projectID: getProjectIdByName.data[0].projectID});
}

return error.message + " : " + createNewProject.data;
}
},

```

Κώδικας Εικόνα 40 – Query *addProjectToCurrentUser*

```

INSERT INTO ProjectUsers (projectID, userID)
VALUES ('{{this.params.newProjectID}}', '{{appsmith.store.user.userID}}');

```

Κώδικας Εικόνα 42 – Μέθοδοι *shouldWidgetsBeDisabled* και *shouldWidgetBeVisible*

```
shouldWidgetsBeDisabled() {
    if(appsmith.store.user.userCategory == appsmith.store.userRights['admin'] ||
        (appsmith.store.user.userCategory == appsmith.store.userRights['user'] &&
        getTaskByID.data[0].assignedUserID == appsmith.store.user.userID)){

        return false;
    }
    else{
        return true;
    }
},

shouldWidgetBeVisible() {
    if(appsmith.store.user.userCategory == appsmith.store.userRights['admin']) {
        return true;
    }
    else{
        return false;
    }
},
```