




20 ΦΕΒΡΟΥΑΡΙΟΥ 2023

ΣΥΓΧΡΟΝΑ ΘΕΜΑΤΑ  
ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ  
SMART ALERT APP

Συντελεστές εργασίας

Σεϊμένης Γεώργιος - Π19204  
Μαυρέλης Κωνσταντίνος - Π19101  
Καρκάνης Ευστράτιος - Π19064



## Πίνακας περιεχομένων

1.	Εισαγωγή.....	2
2.	Σύντομη παρουσίαση της RUP (Rational Unified Process).....	2
3.	Βασικές λεπτομέρειες του λογισμικού .....	3
3.1	Σύλληψη απαιτήσεων .....	3
3.2	Διαγράμματα UML .....	4
3.2.1	Διάγραμμα περιπτώσεων χρήσης .....	4
3.2.2	Διάγραμμα Τάξεων .....	5
4	Το σύστημα Back-end.....	6

## 1. Εισαγωγή

Η συγκεκριμένη εφαρμογή αποτελεί ένα λογισμικό ειδοποίησης χρηστών για ακραία καιρικά φαινόμενα που συμβαίνουν γύρω τους. Πιο συγκεκριμένα, η εφαρμογή αυτή διατίθεται για κινητές συσκευές Android και υποθέτουμε ότι είναι εγκατεστημένη από έναν μεγάλο αριθμό ατόμων σε όλη την Ελλάδα.

Η εφαρμογή καλύπτει τις βασικές λειτουργίες αυθεντικοποίησης ενός χρήστη (σύστημα login/register). Επιπλέον, στην εφαρμογή μπορούν να συνδεθούν δύο κατηγορίες χρηστών, οι **απλοί πολίτες** και οι **υπάλληλοι πολιτικής προστασίας**. Κάθε κατηγορία χρήστη φέρει διαφορετικά δικαιώματα. Συγκεκριμένα,

- Ο υπάλληλος πολιτικής προστασίας είναι σε θέση να βλέπει όλους τους συναγερμούς που έχουν καταγράψει άλλοι χρήστες και να αξιολογεί εάν κάθε συναγερμός είναι έγκυρος ή όχι.
- Ο υπάλληλος πολιτικής προστασίας μπορεί να δημιουργήσει ένα νέο περιστατικό συναγερμού στο σύστημα, προκειμένου να ενημερώσει τους χρήστες που βρίσκονται κοντά στην τοποθεσία που γίνεται η καταστροφή.
- Ο απλός χρήστης μπορεί να υποβάλλει ένα νέο περιστατικό για ακραία καιρικά φαινόμενα στην εφαρμογή, ειδοποιώντας έτσι τους χρήστες που βρίσκονται κοντά στο σημείο της καταστροφής.
- Ο απλός χρήστης μπορεί να δει όλα τα συμβάντα που έχουν υποβληθεί από άλλους χρήστες και να ενημερωθεί για τις καταστροφές που συμβαίνουν, όπως και την τοποθεσία και ώρα που αυτές συμβαίνουν.

Όλα τα δεδομένα που καταγράφονται από τους χρήστες και επεξεργάζεται η εφαρμογή, αποθηκεύονται σε ένα απομακρυσμένο σύστημα βάσης δεδομένων. Συγκεκριμένα, αυτό γίνεται στη firebase.

## 2. Σύνοψη παρουσίαση της RUP (Rational Unified Process)

Η RUP είναι μια διαδικασία τεχνολογίας λογισμικού (Software Engineering Process). Ο στόχος της είναι να διασφαλίσει την παραγωγή λογισμικού υψηλής ποιότητας που ικανοποιεί τις ανάγκες των τελικών χρηστών μέσα σε ένα συγκεκριμένο χρονοδιάγραμμα και κόστος. Ο κύκλος ζωής του λογισμικού (Software life-cycle) σκιαγραφεί τη ζωή του προγράμματος λογισμικού από τη στιγμή της γέννησής του μέχρι τη στιγμή της αντικατάστασης ή της εγκατάλειψής του.

Ο κύκλος ζωής του λογισμικού στην RUP υποδιαιρείται σε τέσσερις συνεχόμενες φάσεις.

Οι τέσσερις φάσεις είναι:

- η φάση σύλληψης (Inception Phase),
- η φάση επεξεργασίας (Elaboration Phase),
- η φάση κατασκευής (Construction Phase) και
- η φάση μετάβασης (Transition Phase)

Ορισμοί σχετικοί με την RUP:

- **Κύκλος ανάπτυξης:** Ένα πέρασμα από τις τέσσερις φάσεις. Κάθε τέτοιο πέρασμα παράγει μια νέα γενιά (generation) λογισμικού.
- **Κύκλος εξέλιξης:** Οι διαδοχικοί κύκλοι στην περίπτωση που το προϊόν συνεχίσει να εξελίσσεται στην επόμενη γενιά του, ονομάζονται κύκλοι εξέλιξης (evolution cycles).

### 3. Βασικές λεπτομέρειες του λογισμικού

#### 3.1 Σύλληψη απαιτήσεων

Παρακάτω καταγράφονται οι λειτουργικές απαιτήσεις της εφαρμογής, δηλαδή οι υπηρεσίες που πρέπει να παρέχει το σύστημα και πως εκείνο πρέπει να αντιδρά στις διάφορες καταστάσεις και εισόδους:

- να μπορεί να γίνει δημιουργία λογαριασμού από ένα νέο χρήστη.
- να συνδέεται ένας υπάρχων χρήστης στην εφαρμογή όταν υποβάλει τα στοιχεία του (email και κωδικός) και αυτά είναι σωστά.
- δυνατότητα σύνδεσης και αποσύνδεσης στο σύστημα με εντολή του χρήστη.
- σε περίπτωση που ένας χρήστης δεν ταυτοποιείται ή υπάρχει πρόβλημα επικοινωνίας της εφαρμογής με τη βάση δεδομένων, το πρόγραμμα στέλνει αντίστοιχα μηνύματα στον χρήστη.
- οι κατηγορίες χρηστών είναι δύο, οι απλοί πολίτες και οι υπάλληλοι πολιτικής προστασίας.
- προσθήκη ενός νέου συμβάντος υψηλού κινδύνου από οποιονδήποτε χρήστη. Χαρακτηριστικά που μπορούν να υποβληθούν για ένα περιστατικό είναι ο τύπος του περιστατικού και η περιγραφή του περιστατικού. Αυτόματα, κατά την υποβολή, προσαρτώνται η τοποθεσία και η ώρα υποβολής ως δεδομένα που συνοδεύουν το συμβάν. Όλα αυτά τα δεδομένα καταγράφονται στην firebase.

- όλα τα συμβάντα που δημιουργούν οι χρήστες θα πρέπει να αποθηκεύονται στην firebase.
- να ειδοποιεί η πλατφόρμα τους χρήστες σε περίπτωση ενός νέου συμβάντος υψηλού κινδύνου κοντά τους.
- μόνο οι υπάλληλοι πολιτικής προστασίας είναι σε θέση να απορρίπτουν ή να εγκρίνουν ένα περιστατικό (ως προς την εγκυρότητά τους). Κατά την έγκριση ενός περιστατικού, οι χρήστες που βρίσκονται σε κοντινή απόσταση από το συμβάν ειδοποιούνται για τον κίνδυνο.

### 3.2 Διαγράμματα UML

Σε αυτό το κεφάλαιο παραθέτουμε δύο διαγράμματα της UML, το διάγραμμα περιπτώσεων χρήσης και το διάγραμμα τάξεων.

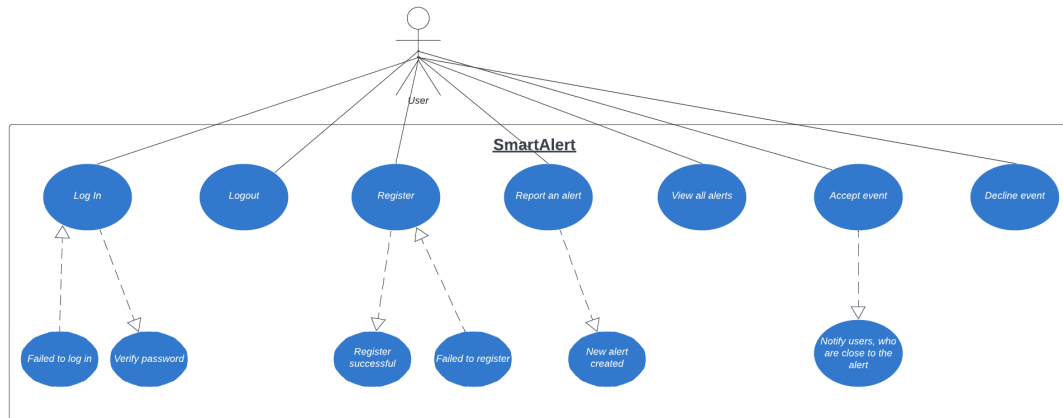
**Σημαντική παρατήρηση:** κατά το πρώτο στάδιο της διαδικασίας RUP, οι απαιτήσεις που υλοποιήσαμε και παραθέσαμε παραπάνω, παρέμειναν οι ίδιες μέχρι την τελική μορφή της εφαρμογής. Επομένως, αυτό σημαίνει ότι τα δύο διαγράμματα που ακολουθούν είναι τα ίδια και στις επόμενες τρεις φάσεις του μοντέλου ζωής RUP. Για αυτόν τον λόγο, δεν τα ξανααναφέρουμε παρακάτω.

#### 3.2.1 Διάγραμμα περιπτώσεων χρήσης

Σε αυτό το διάγραμμα περιγράφουμε τις λειτουργίες που είναι διαθέσιμες στον χρήστη όπως ορίζονται από την εκφώνηση, καθώς και οι μετέπειτα καταλήξεις αυτών. Συγκεκριμένα ο χρήστης μπορεί να:

1. κάνει εγγραφή
2. κάνει έξοδο από την εφαρμογή
3. κάνει είσοδο στην εφαρμογή
4. δημιουργήσει ένα νέο περιστατικό υψηλού κινδύνου
5. εάν είναι υπάλληλος πολιτικής προστασίας, να απορρίψει ή να εγκρίνει έναν συναγερμό, και
6. να δει όλες τις αναφορές που έχουν γίνει από άλλους χρήστες της εφαρμογής.

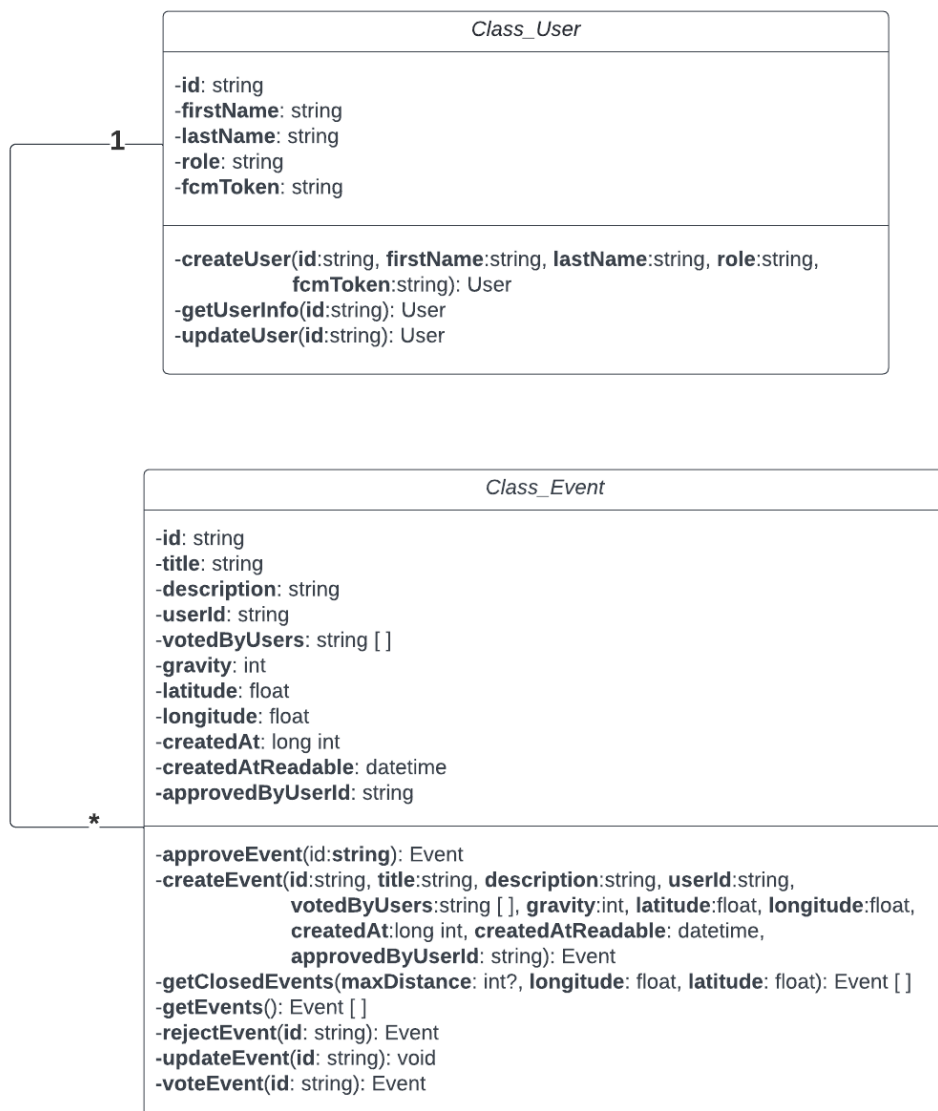
Η φορά από τα βέλη δείχνει την χρονική συνέχεια εκτέλεσης. Για παράδειγμα ο χρήστης δεν μπορεί να κάνει «Verify Password» αν δεν πετύχει η λειτουργία «Log in». Παρ' όλα αυτά, το «Failed to Login» μπορεί να πραγματοποιηθεί ανεξάρτητα από την επιτυχία του Login.



Διάγραμμα περιπτώσεων χρήσης

### 3.2.2 Διάγραμμα Τάξεων

Σε αυτό το διάγραμμα αναλύουμε τις κλάσεις που χρησιμοποιούμε στην εφαρμογή μας. Έχουμε αποφασίσει να φτιάξουμε δύο κλάσεις με τα παρακάτω χαρακτηριστικά και συναρτήσεις. Οι κλάσεις αυτές συνδέονται καθώς τα αντικείμενα «Event» φτιάχνονται από τα αντικείμενα «User». Ένα αντικείμενο «User» μπορεί να φτιάξει όσα «Event» επιθυμεί.



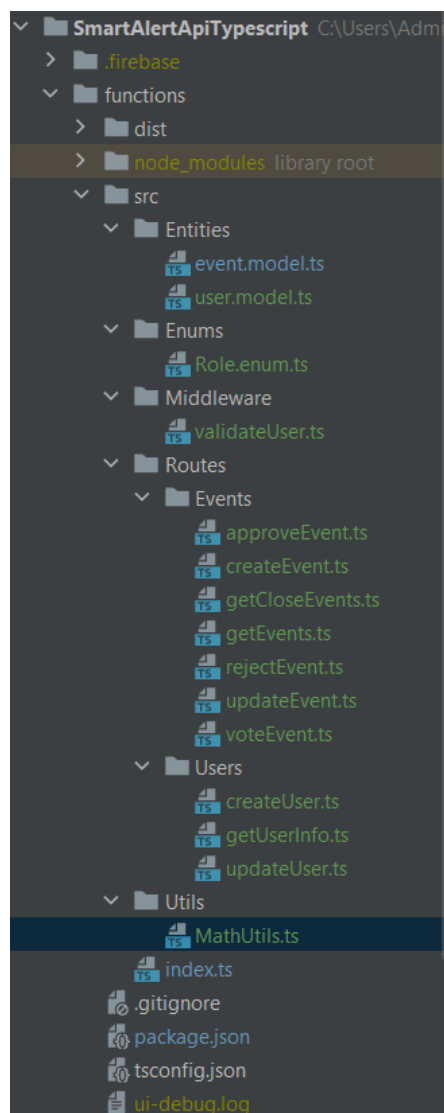
Διάγραμμα Τάξεων

## 4 Το σύστημα Back-end

Το σύστημα Back End της εφαρμογής είναι γραμμένο σε γλώσσα JavaScript με την βοήθεια του framework **node typescript**. Το API μας χρησιμοποιεί servlet αρχιτεκτονική, πράγμα που επιτυγχάνεται με την χρήση firebase μεθόδων.

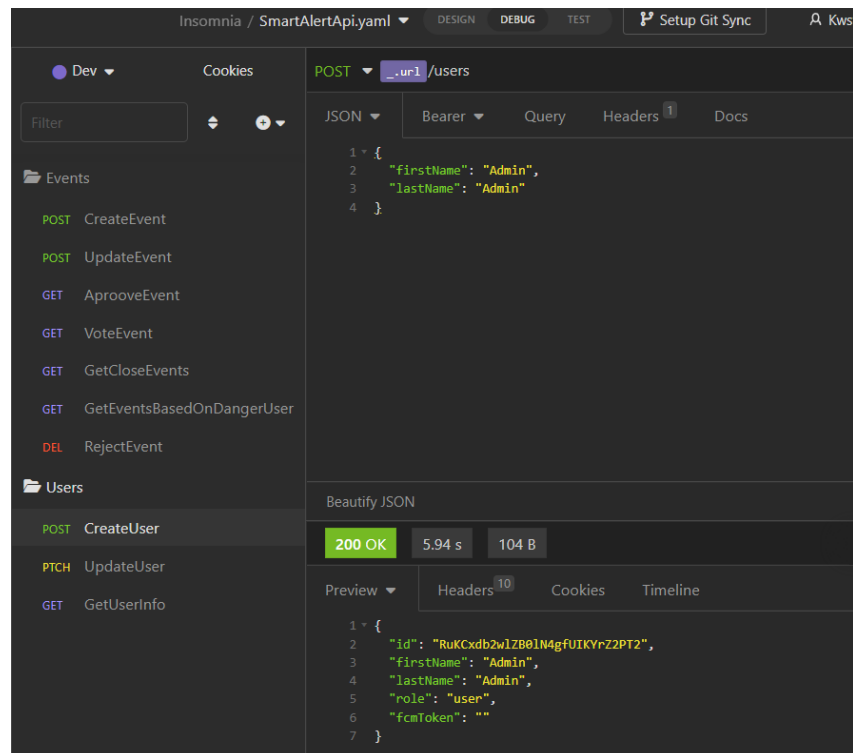
Η βάση δεδομένων που χρησιμοποιούμε, βρίσκεται και αυτή στην firebase και ονομάζεται «firestore». Παρόλο που η firestore δεν είναι σχεσιακή βάση, χρησιμοποιήθηκε το «fireorm» στον προγραμματισμό του API, έτσι ώστε να είμαστε όσο πιο κοντά γίνεται στην ιδέα του αντικειμενοστραφή προγραμματισμού.

Η δομή του API είναι η ακόλουθη:





Τα endpoints του backend συστήματος είναι τα ακόλουθα:



Τα endpoints του συστήματος σε open API μορφή.

```
//Validation Auth
app.use(validateUserMiddle)

//User
app.post( path: "/users",createUser)
app.patch( path: "/users",updateUser)
app.get("/users",getUserInfo)

//Events
app.get("/events",getEvents)
app.get("/events/:id/vote",voteEvent)
app.post( path: "/events",createEvent)
app.post( path: "/events/:id",updateEvent)
app.get("/events/:id/approve",approveEvent)
app.delete( path: "/events/:id",rejectOrDeleteEvent)
app.get("/events/close",getCloseEvent)
```

Τα endpoints του συστήματος σε text based μορφή.

Επιπλέον, το authentication γίνεται με βάση το firebase authentication και ελέγχεται μέσω του API μέσω ενός access token.

Για τον υπολογισμό της επικινδυνότητας κάθε περιστατικού χρησιμοποιήθηκε η παρακάτω γνησίως αύξουσα συνάρτηση δύο μεταβλητών:  $f(\text{users}, \text{gravity}) = 100 * (1 - e^{-(\text{users} * \text{gravity} * 0.1)})$ , όπου το users δηλώνει το πόσοι χρήστες έχουν ψηφίσει το

συγκεκριμένο περιστατικό και το gravity δηλώνει την επικινδυνότητα του περιστατικού. Για παράδειγμα, η φωτιά έχει βαθμό επικινδυνότητας 5, ο σεισμός έχει βαρύτητα 2 κτλ. .

Η εφαρμογή, για να αποφύγει όσο το δυνατόν περισσότερα διπλότυπα περιστατικά, εμφανίζει στον χρήστη τα περιστατικά, τα οποία βρίσκονται σε κοντινή απόσταση από εκείνον πριν γίνει κάποια υποβολή περιστατικού από αυτόν. Στην περίπτωση που κάποιο περιστατικό είναι ίδιο με αυτό που εμφανίζει η εφαρμογή, ο χρήστης το επιλέγει και το ψηφίζει. Με αυτόν τον τρόπο, η δημοτικότητα του περιστατικού ανεβαίνει στη λίστα. Εάν αυτό περάσει τον βαθμό επικινδυνότητας 80, τότε το περιστατικό δημοσιεύεται αυτόματα, χωρίς να είναι απαραίτητη η έγκριση από κάποιον υπάλληλο πολιτικής προστασίας.

Αξίζει να σημειωθεί ότι κατά την είσοδο του χρήστη στην εφαρμογή, το λογισμικό μας δημιουργεί ένα «firebase notification token», το οποίο αποθηκεύεται στην βάση δεδομένων μας. Με αυτό το token, το backend σύστημά μας εντοπίζει και ειδοποιεί τους χρήστες που βρίσκονται έως και 2 χιλιόμετρα από το εκάστοτε περιστατικό.

Τέλος, όσον αφορά τον κώδικα του backend συστήματος, αυτός βρίσκεται αποθηκευμένος σε απομακρυσμένο repository, ο σύνδεσμος προς το οποίο είναι ο ακόλουθος: <https://github.com/CyberGl1tch/SmartAlertApiTypescript>