**Τεχνολογία Λογισμικού**
Τομέας Ηλεκτρονικής και Υπολογιστών
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Α.Π.Θ.

8ο Εξάμηνο
Άνοιξη 2021

# Σύστημα *WalkMe*

**Did somebody say “walk”?**

# Προδιαγραφές Λογισμικού και Υλοποίηση συστήματος

**Version 1.5**

**Κοσέογλου Σ. , Οικονόμου Α., Πετρίδης Α., Χατζής Κ.**

**31/05/2021**

# Ιστορικό Αλλαγών

| Όνομα | Ημερομηνία | Αλλαγή | Έκδοση |
|---|---|---|---|
| Α. Συμεωνίδης | 17/05/2007 | Δημιουργία εγγράφου.<br>Προσαρμογή των προτύπων του K. E. Wiegers[1] και του M. Smialek's. | 0.1 |
| Α. Συμεωνίδης | 29/3/2014 | Μικρή αναθεώρηση – τροποποίηση ενοτήτων | 0.1.3 |
| Χ. Ζολώτας | 10/4/2020 | Μεγάλη αναθεώρηση – αφαίρεση ενοτήτων | 0.4 |
| Χ. Ζολώτας | 15/4/2020 | Μεγάλη αναθεώρηση – προσθήκη ενότητας REST προδιαγραφών | 0.5.3 |
| Κ. Παναγιώτου | 25/4/2020 | Μεγάλη αναθεώρηση – προσθήκη ενότητας Nodered περιγραφής | 0.5.7 |
| Α. Συμεωνίδης | 30/4/2020 | Αναθεώρηση και τελική δομή προτύπου | 0.6 |

# Μέλη της Ομάδας Ανάπτυξης

| Όνομα | ΟΑ | Email |
|---|---|---|
| Κοσέογλου Σωκράτης | 1 | sokrkose@ece.auth.gr |
| Οικονόμου Αλέξανδρος | 1 | alexanco@ece.auth.gr |
| Πετρίδης Αλέξανδρος | 1 | alepetpan@ece.auth.gr |
| Χατζής Κωνσταντίνος | 1 | kachatzis@ece.auth.gr |

# Πίνακας Περιεχομένων

# Λίστα Σχημάτων

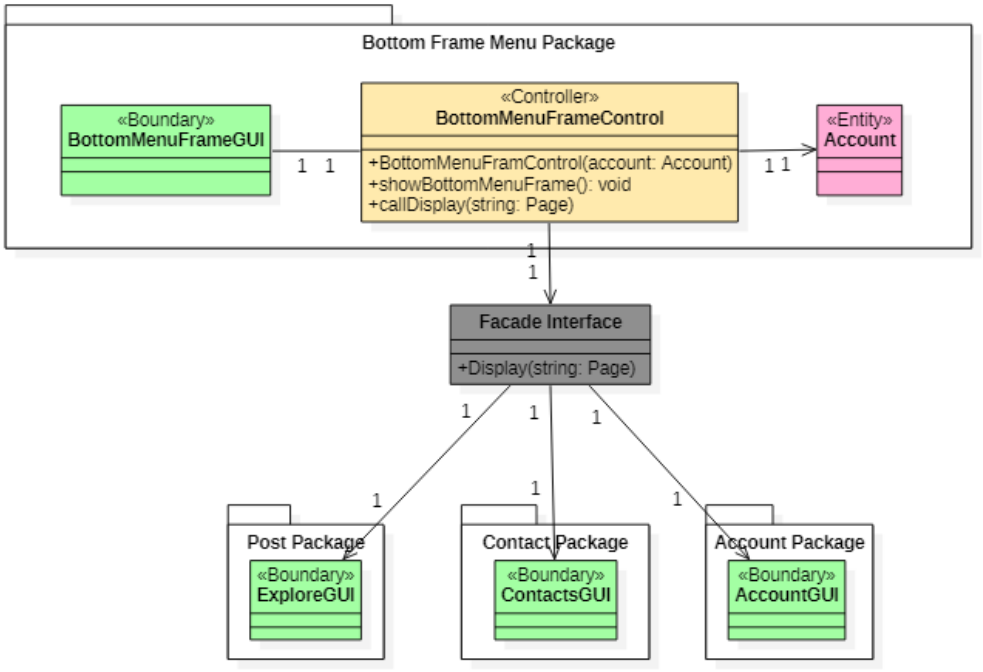
# 1. Πρότυπα Σχεδιασμού που υιοθετήθηκαν

## 1.1 Πρότυπο σχεδίασης Facade

- Πρότυπο Facade
- Η υλοποίηση του συγκεκριμένου σχεδιαστικού προτύπου παρέχει μια πιο ενοποιημένη διεπαφή στο συνολικό διάγραμμα κλάσεων.

Το πρότυπο Facade χρησιμοποιείται έτσι ώστε να υλοποιήσει μια διεπαφή υψηλού επιπέδου η οποία κάνει το σύστημα πιο εύκολο στην χρήση του, ενώ ταυτόχρονα ικανοποιεί την μη-λειτουργική απαίτηση (ΜΛΑ) συντηρησιμότητας του συστήματος. Πιο συγκεκριμένα, όπως φαίνεται στο διάγραμμα παρακάτω, το πρότυπο εφαρμόζεται με την χρήση της κλάσης "Facade Interface" η οποία χρησιμοποιείται για την εναλλαγή της οθόνης διεπαφής.

Παρακάτω παρουσιάζεται η υλοποίηση του προτύπου για το μενού επιλογής οθόνης.

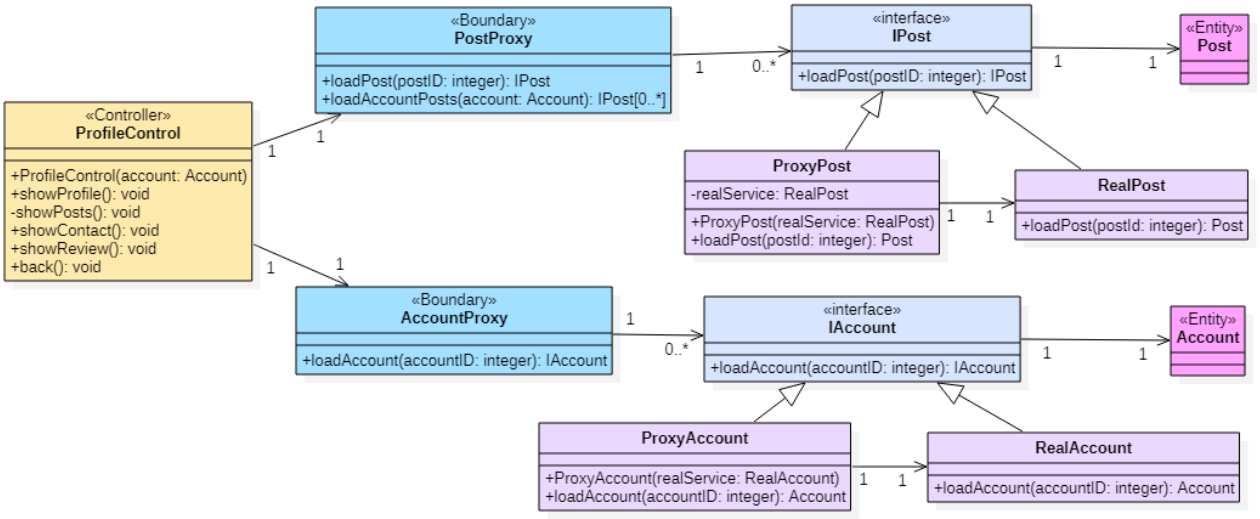



Σχήμα Ι: Πρότυπο Σχεδίασης Facade

## 1.2 Πρότυπο σχεδίασης Proxy

- Πρότυπο Proxy (Διαμεσολαβητής)
- Η υλοποίηση του συγκεκριμένου σχεδιαστικού προτύπου ικανοποιεί την ανάγκη της γρήγορης φόρτωσης και παρουσίασης της διεπαφής χρήστη.

Το πρότυπο αυτό αποτελεί μια υλοποίηση του "Virtual Proxy". Ικανοποιεί την σχεδιαστική ανάγκη της **ΜΛΑ-1** "Το σύστημα πρέπει να έχει χρόνο απόκρισης μικρότερο των 200 ms". Η παραλλαγή "Virtual Proxy" εφαρμόζεται για τη γρήγορη φόρτωση ενός προσωρινού αντικειμένου χωρίς ολοκληρωμένο περιεχόμενο, έτσι ώστε να δίνεται η εντύπωση της γρήγορης απόκρισης του συστήματος προς το χρήστη. Παρουσιάζεται η υλοποίηση του συστήματος στη λειτουργία της φόρτωσης των δημοσιεύσεων ενός χρήστη. Αντίστοιχα θα υλοποιηθεί το πρότυπο σε άλλες δυναμικές λίστες, όπως αυτή των αξιολογήσεων και των μηνυμάτων μεταξύ χρηστών.



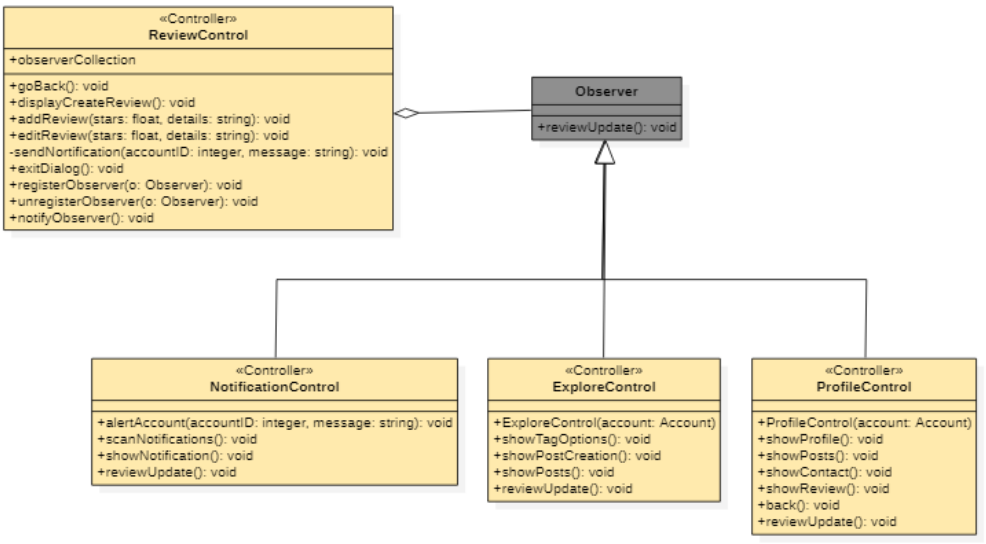

Σχήμα II: Πρότυπο Σχεδίασης Proxy

# 1.3 Πρότυπο σχεδίασης Παρατηρητής

- Πρότυπο Παρατηρητής (Observer)
- Η υλοποίηση του συγκεκριμένου σχεδιαστικού προτύπου ικανοποιεί την ανάγκη συνοχής ανάμεσα στις όψεις όταν αλλάζει η κατάσταση κάποιου αντικειμένου.

## 1.3.1 Παρατηρητής για ReviewControl

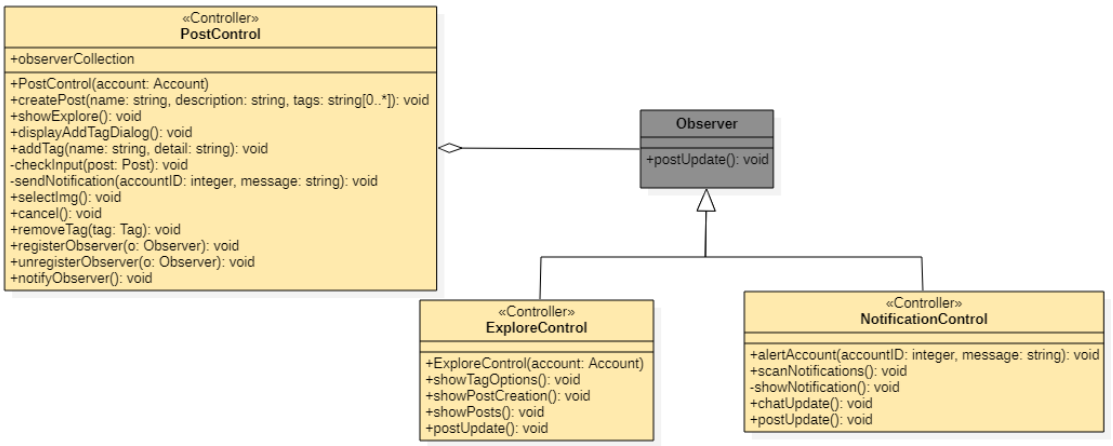
Σκοπός της εφαρμογής του συγκεκριμένου προτύπου στην κλάση ReviewControl είναι η αυτόματη ενημέρωση όλων των οθονών που απεικονίζουν τις αξιολογήσεις και συναφή αυτής (τα αστεράκια μέσης αξιολόγησης κάθε χρήστη), οι οποίες είναι αρχικά στην σελίδα Αξιολογήσεων του χρήστη, στην σελίδα του Προφίλ του χρήστη,πάνω σε κάθε δημοσίευση του χρήστη που βρίσκεται στην κεντρική σελίδα και στις ειδοποιήσεις. Παρακάτω παρουσιάζεται το συνοπτικό διάγραμμα κλάσεων που προκύπτει μετά την εφαρμογή του προτύπου.




Σχήμα III: Πρότυπο Σχεδίασης Observer -Αξιολογήσεις

### 1.3.2 Παρατηρητής για PostControl

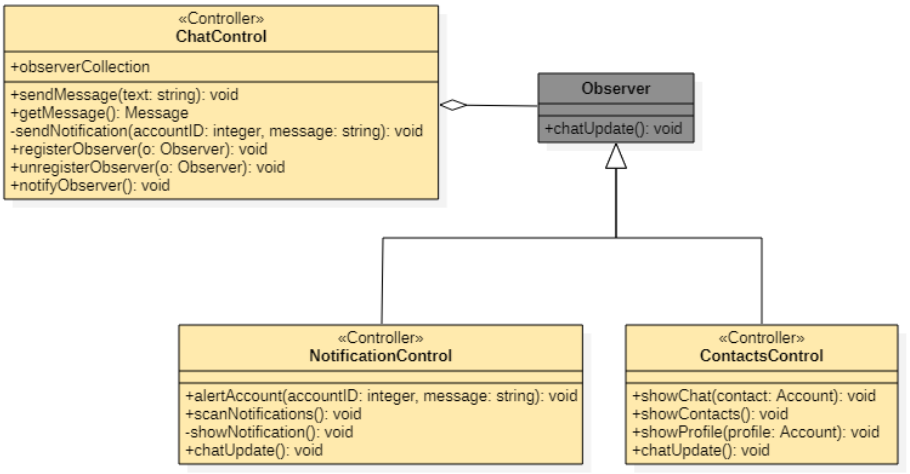
Σκοπός της εφαρμογής του συγκεκριμένου προτύπου στην κλάση PostControl είναι η αυτόματη ενημέρωση όλων των οθονών που απεικονίζουν τις δημοσιεύσεις των χρηστών, οι οποίες είναι η κεντρική σελίδα δημοσιεύσεων, το προφίλ του χρήστη και στις ειδοποιήσεις. Παρακάτω παρουσιάζεται το συνοπτικό διάγραμμα κλάσεων που προκύπτει μετά την εφαρμογή του προτύπου.




Σχήμα IV: Πρότυπο Σχεδίασης Observer - Δημοσιεύσεις

### 1.3.3 Παρατηρητής για ChatControl

Σκοπός της εφαρμογής του συγκεκριμένου προτύπου στην κλάση ChatControl είναι η αυτόματη ενημέρωση όλων των οθονών που απεικονίζουν τα μηνύματα μεταξύ χρηστών, οι οποίες είναι η σελίδα μηνυμάτων και οι ειδοποιήσεις. Παρακάτω παρουσιάζεται το συνοπτικό διάγραμμα κλάσεων που προκύπτει μετά την εφαρμογή του προτύπου.




Σχήμα V: Πρότυπο Σχεδίασης Observer - Συνομιλία

# 2. Αρχιτεκτονική Συστήματος

Στο συγκεκριμένο κεφάλαιο αναλύεται η REST αρχιτεκτονική του συστήματος. Η ολοκληρωμένη υλοποίηση σε OpenAPI μπορεί να βρεθεί στο σύνδεσμο:

**OpenAPI v2 Specification (json)**

**OpenAPI v2 Specification (yml)**

Το πακέτο εξυπηρετητή για Node.js βρίσκεται αντίστοιχα στο σύνδεσμο:

**Node.js Server Package**

## 2.1 Αναγνώριση Πόρων Συστήματος

| Κλάση BEC | Πόρος REST | Endpoints (HTTP Verbs) |
|---|---|---|
| **Account** | **/account**<br>**/account/{accountID}** | **POST**<br>**GET, PUT** |
| **Token** | **/account/login**<br>**/account/logout** | **PUT**<br>**GET** |
| **Post** | **/post**<br>**/account/{accountID}/post/findByTag/{tagID}**<br>**/account/{accountID}/post/{postID}**<br>**/account/{accountID}/post**<br>**/account/{accountID}/findPostsByAccount/{accountID}** | **GET**<br>**GET**<br>**GET**<br>**POST**<br>**GET** |
| **Chat** | **/account/{accountID}/chat**<br>**/account/{accountID}/chat/{chatID}**<br>**/account/{accountID}/findChatByAccount/{accountID}** | **GET, POST**<br>**GET**<br>**GET** |
| **Message** | **/account/{accountID}/chat/{chatID}/message**<br>**/account/{accountID}/chat/{chatID}/message/{messageID}** | **GET, POST**<br>**GET** |
| **Review** | **/account/{accountID}/review**<br>**/account/{accountID}/review/{reviewID}**<br>**/account/{accountID}/findReviewsByAccount/{refAccountID}** | **POST**<br>**GET, POST, PUT, DELETE**<br>**GET** |
| **Tag** | **/account/{accountID}/tag**<br>**/account/{accountID}/tag/{tagID}** | **POST, GET**<br>**PUT, DELETE** |
| **Notification** | **/account/{accountID}/notification**<br>**/account/{accountID}/notification/{notificationID}** | **POST, GET**<br>**GET, PUT** |

# 2.2 Τεκμηρίωση REST διεπαφής

## 2.2.1 Πόρος Account

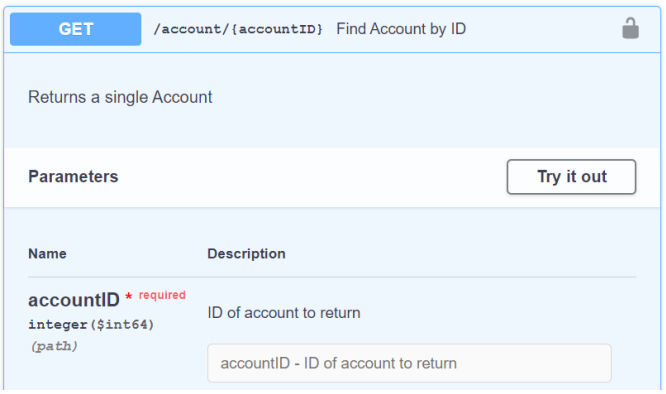
### 2.2.1.1 Μοντέλο δεδομένων Account

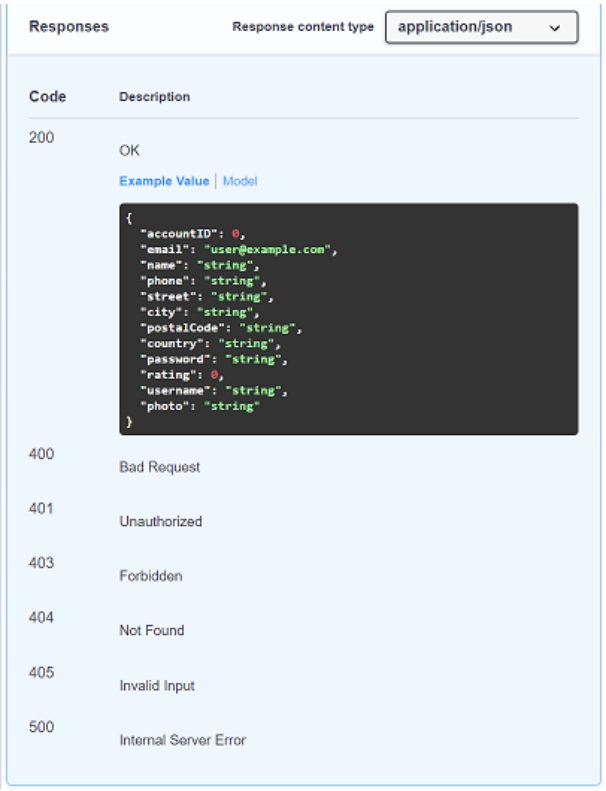
**Account** {

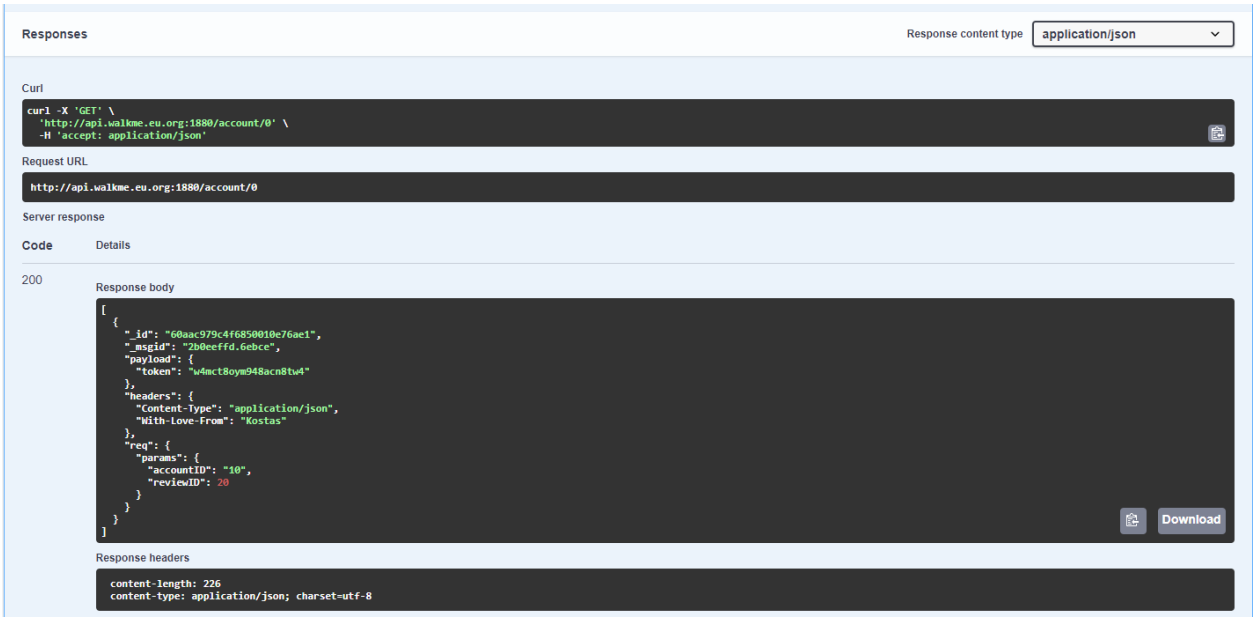
| | |
|---|---|
| accountID | integer |
| email | string($email) |
| name | string |
| phone | string |
| street | string |
| city | string |
| postalCode | string |
| country | string |
| password | string($password) |
| rating | number($float) |
| username | string |
| photo | string($byte) |
| | Picture encoded in Base64. |

}

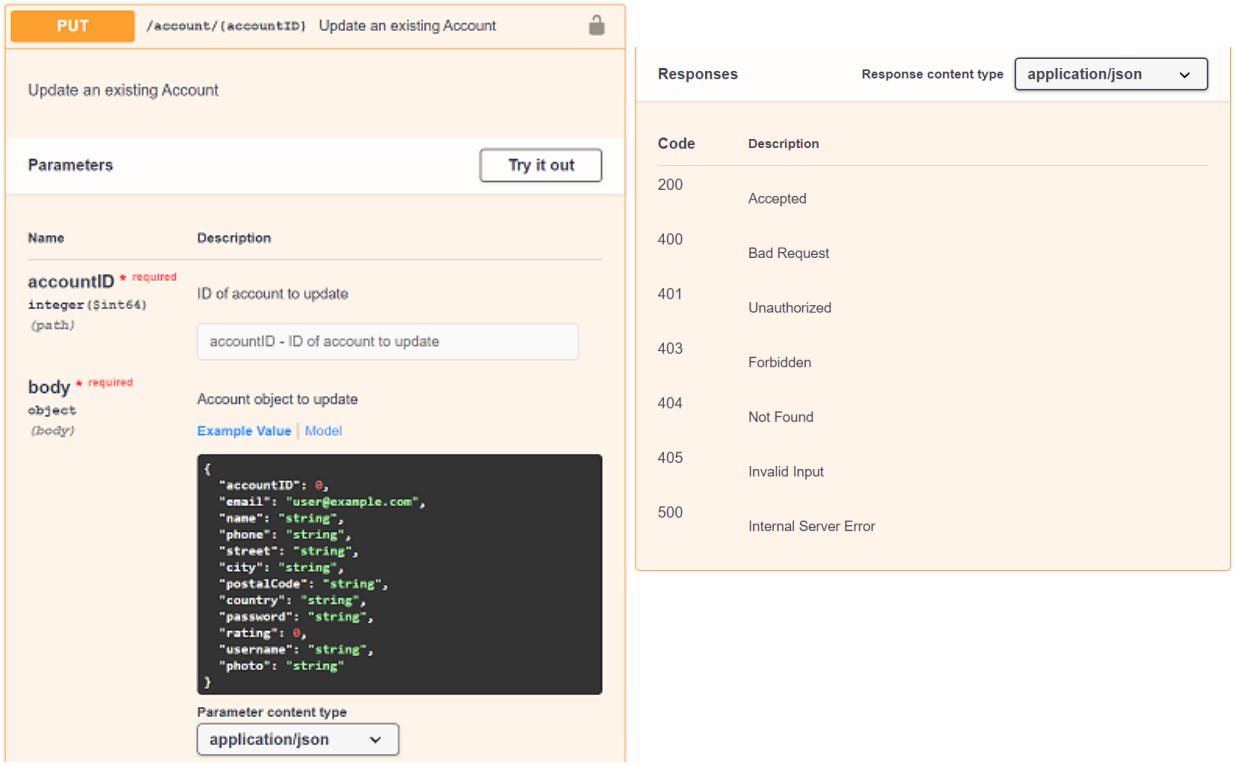
### 2.2.1.2 Endpoint GET πόρου Account






### 2.2.1.3 Endpoint PUT πόρου Account



**2.2.1.4 Endpoint POST πόρου Account**

POST /account Sign Up as a new user account

Creates a new user account

**Parameters** Try it out

| Name | Description |
|---|---|
| **body** * required<br>object<br>*(body)* | The Account to be created<br>Example Value \| Model |

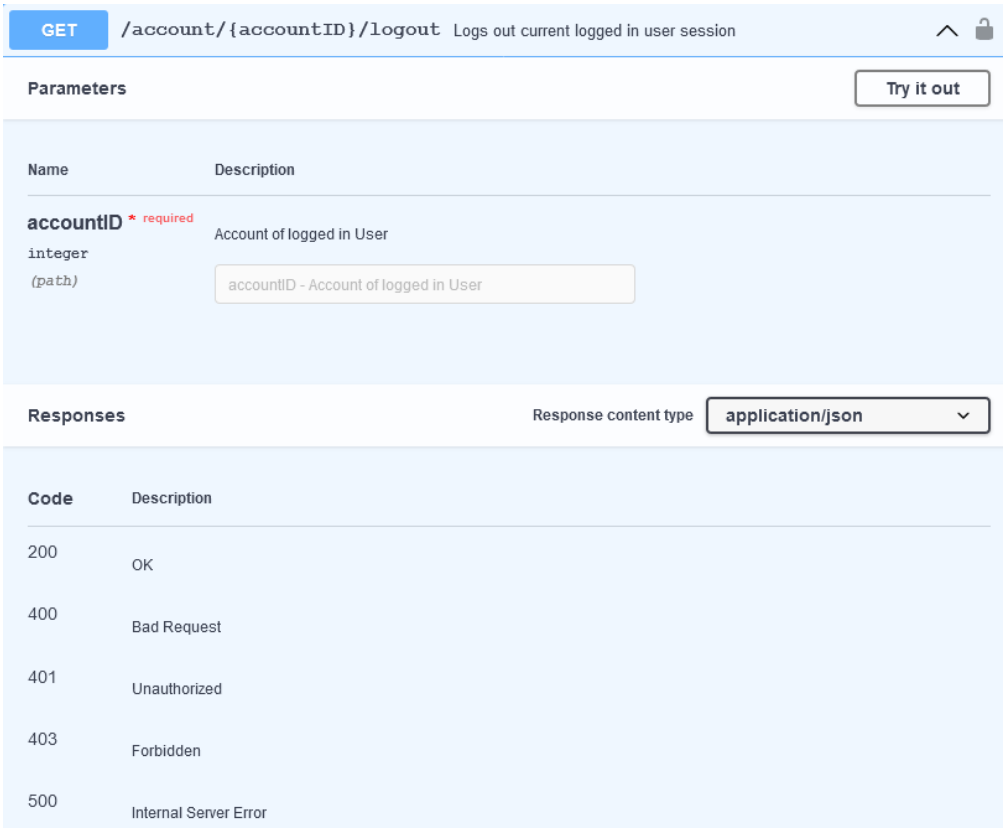
```
{
  "accountID": 0,
  "email": "user@example.com",
  "name": "string",
  "phone": "string",
  "street": "string",
  "city": "string",
  "postalCode": "string",
  "country": "string",
  "password": "string",
  "rating": 0,
  "username": "string",
  "photo": "string"
}
```

Parameter content type: application/json

**Responses** Response content type: application/json

| Code | Description |
|---|---|
| 200 | OK |
| 400 | Bad Request |
| 403 | Forbidden |
| 405 | Invalid Input |
| 500 | Internal Server Error |

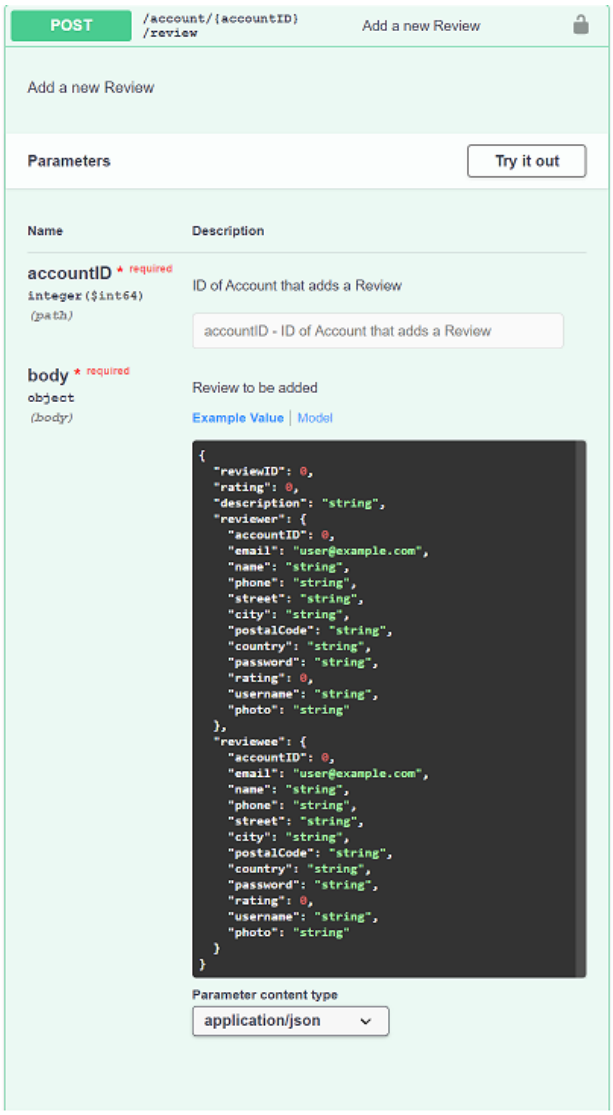
### 2.2.1.5 Endpoint GET πόρου Account (Αποσύνδεση χρήστη)

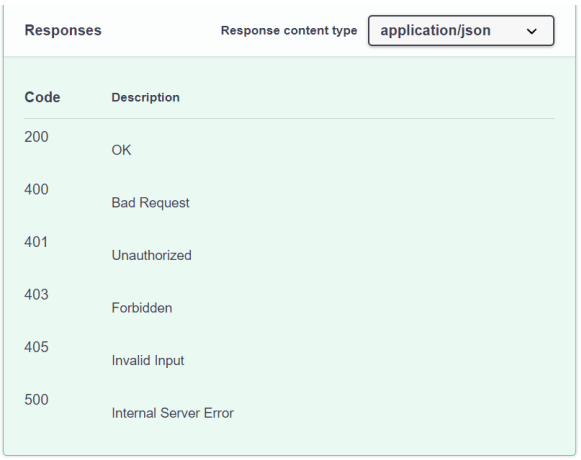


## 2.2.2 Πόρος Review

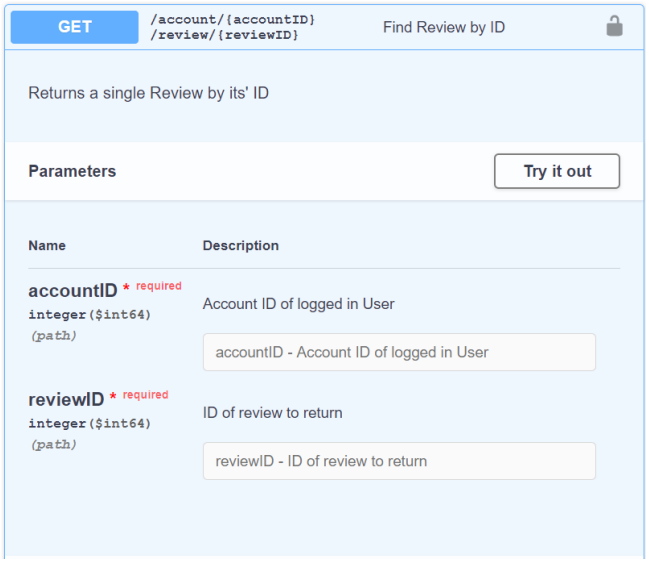
### 2.2.2.1 Μοντέλο δεδομένων Review

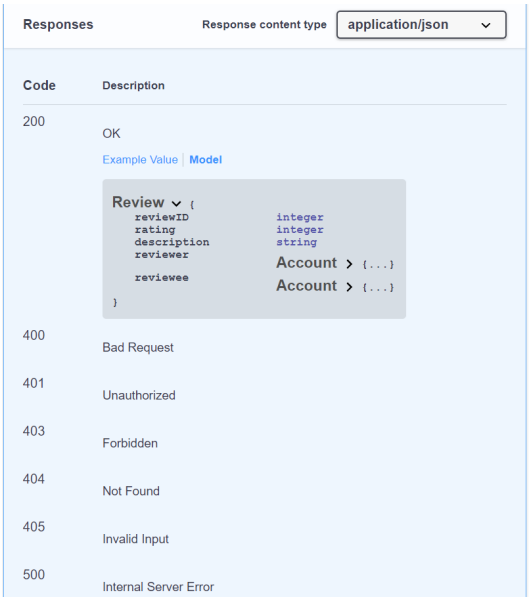

### 2.2.2.2 Endpoint POST πόρου Review



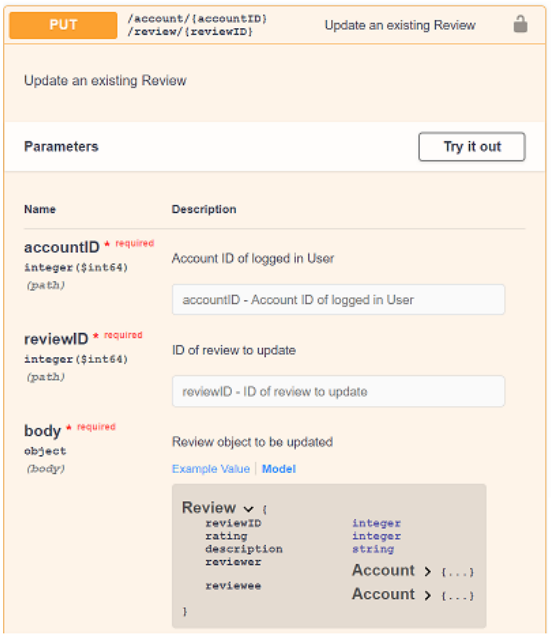



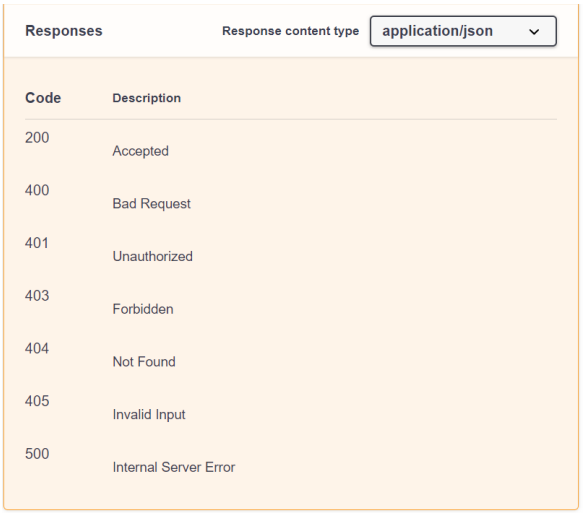
### 2.2.2.3 Endpoint GET πόρου Review



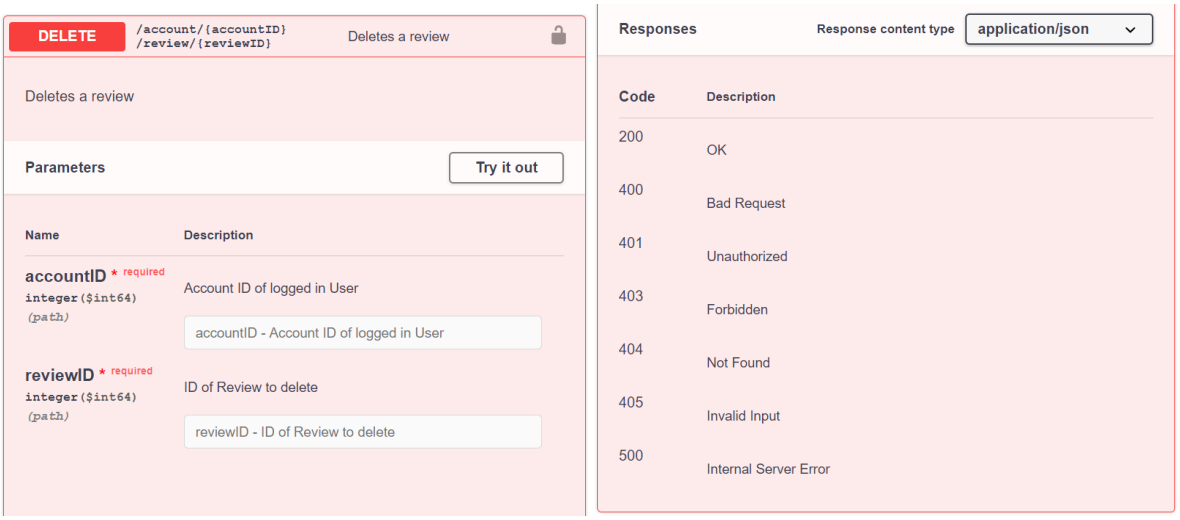




### 2.2.2.4 Endpoint PUT πόρου Review



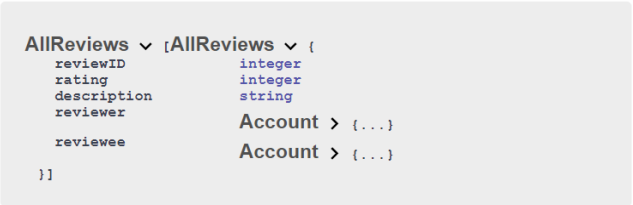



#### 2.2.2.5 Endpoint DELETE πόρου Review

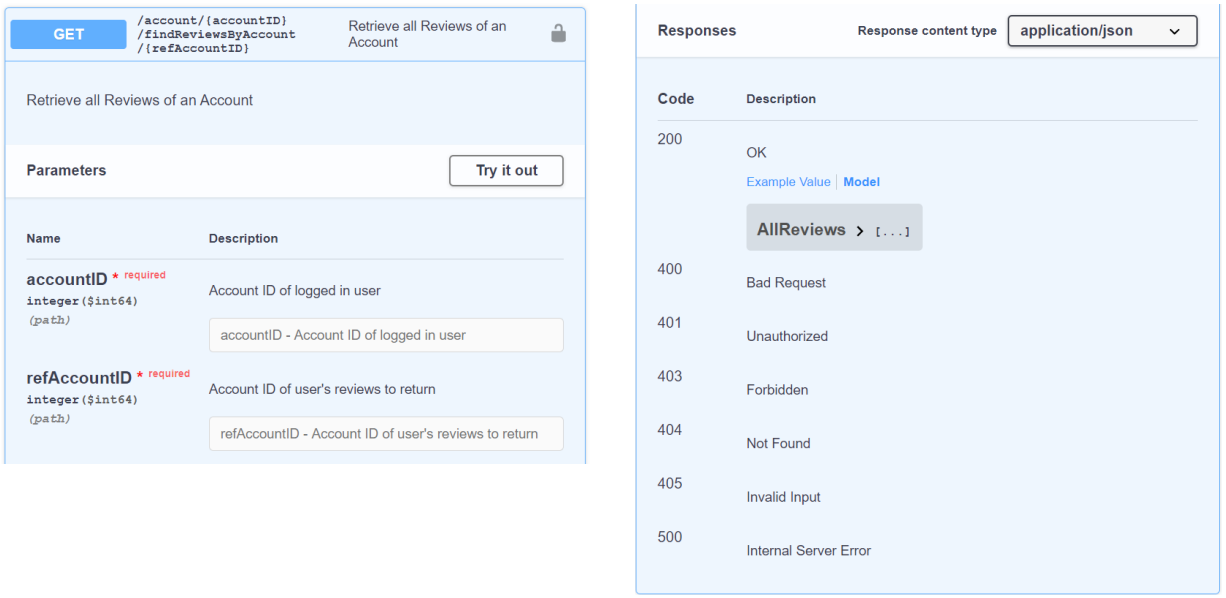



## 2.2.3 Πόρος AllReviews

### 2.2.3.1 Μοντέλο δεδομένων AllReviews



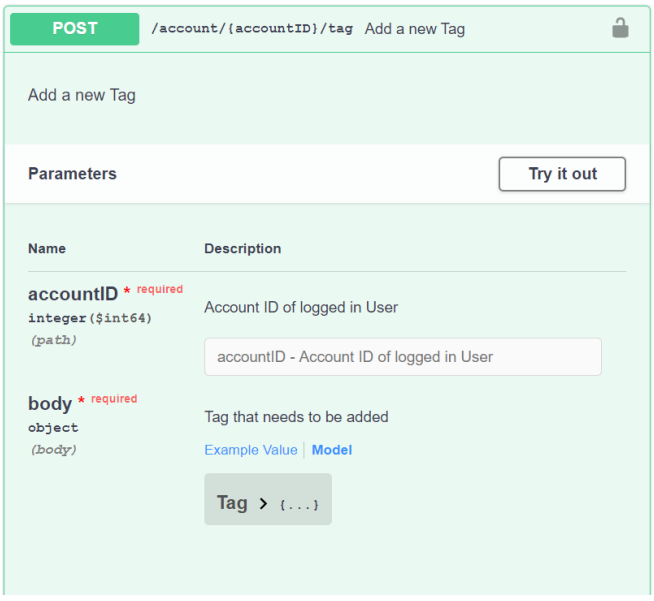

#### 2.2.3.2 Endpoint GET πόρου AllReviews

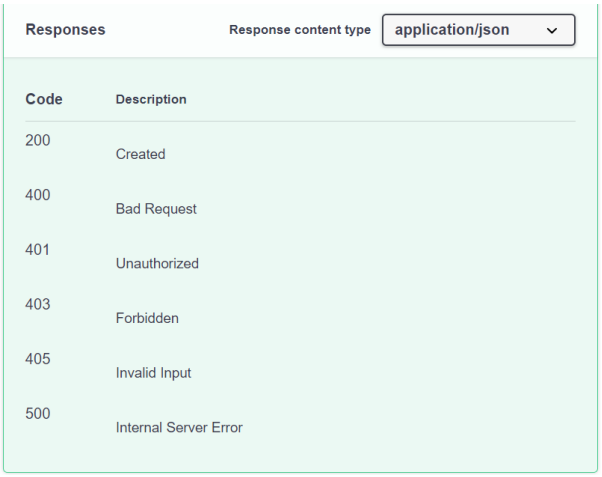


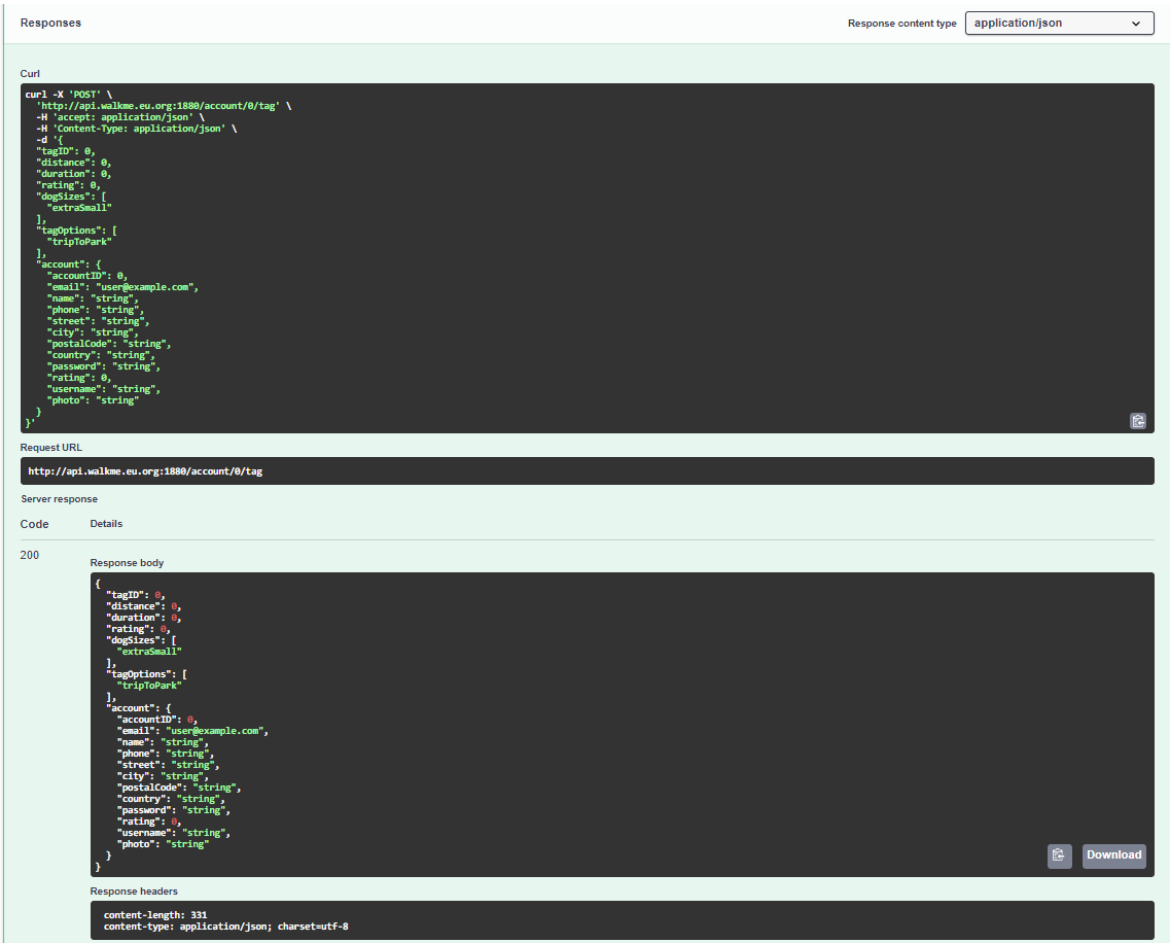
## 2.2.4 Πόρος Tag

### 2.2.4.1 Μοντέλο δεδομένων Tag

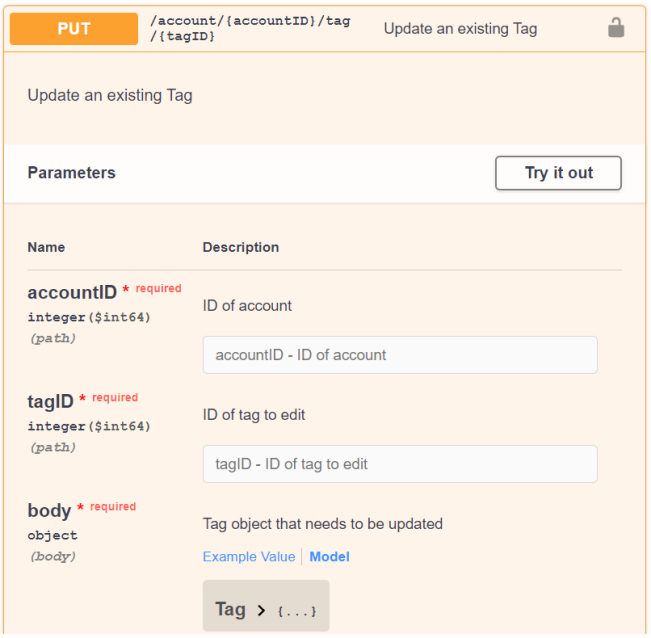

### 2.2.4.2 Endpoint POST πόρου Tag



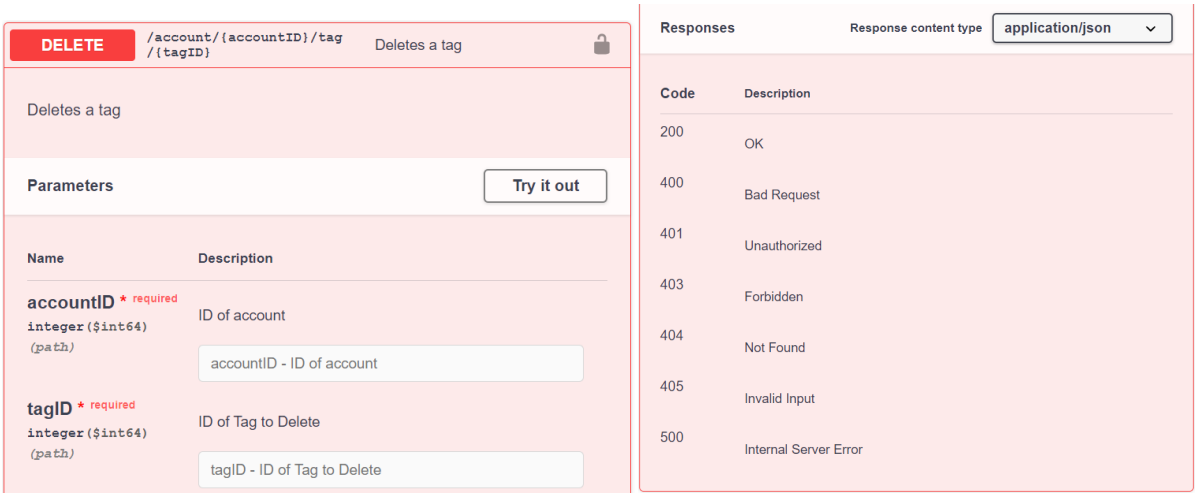






### 2.2.4.3 Endpoint PUT πόρου Tag

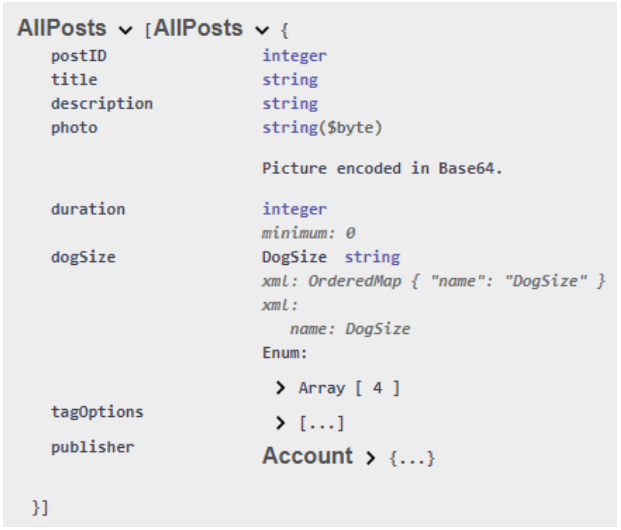


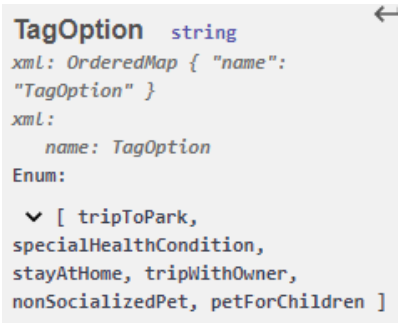

#### 2.2.4.4 Endpoint DELETE πόρου Tag

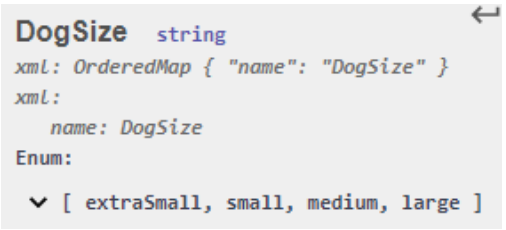



## 2.2.5 Πόρος AllPosts

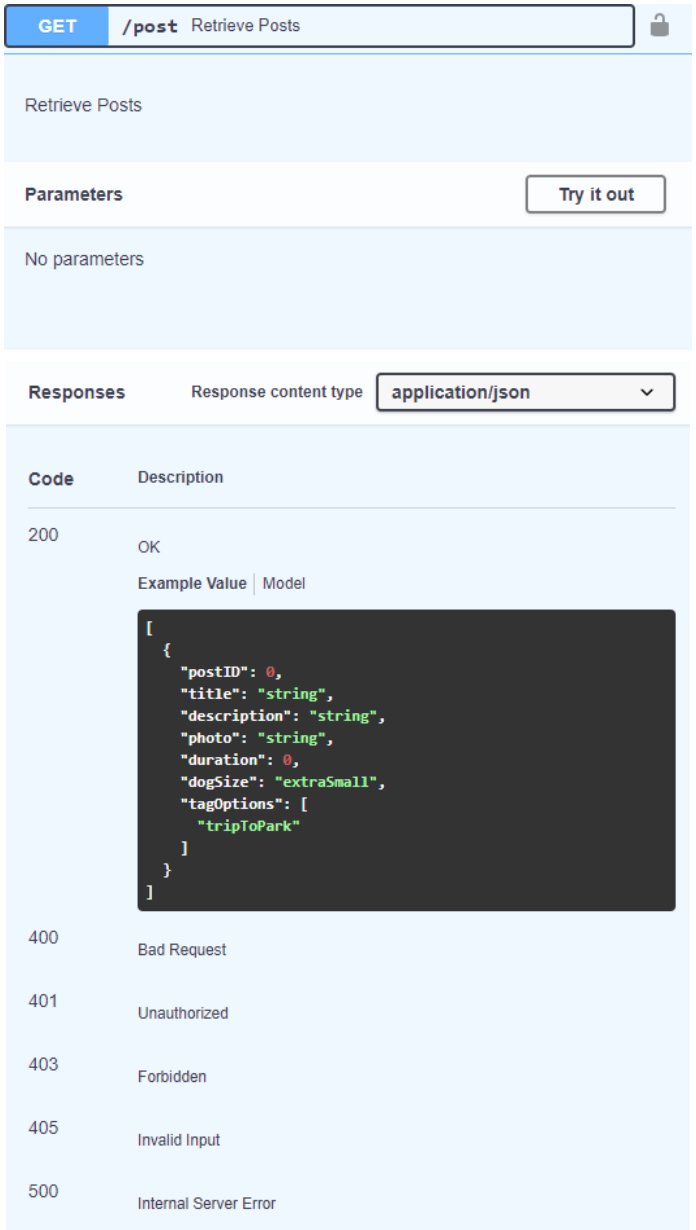
### 2.2.5.1 Μοντέλο δεδομένων AllPosts



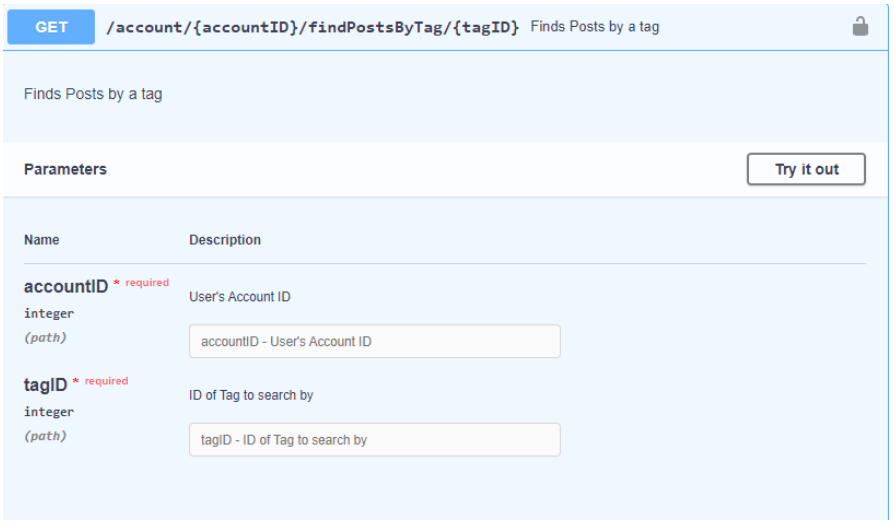



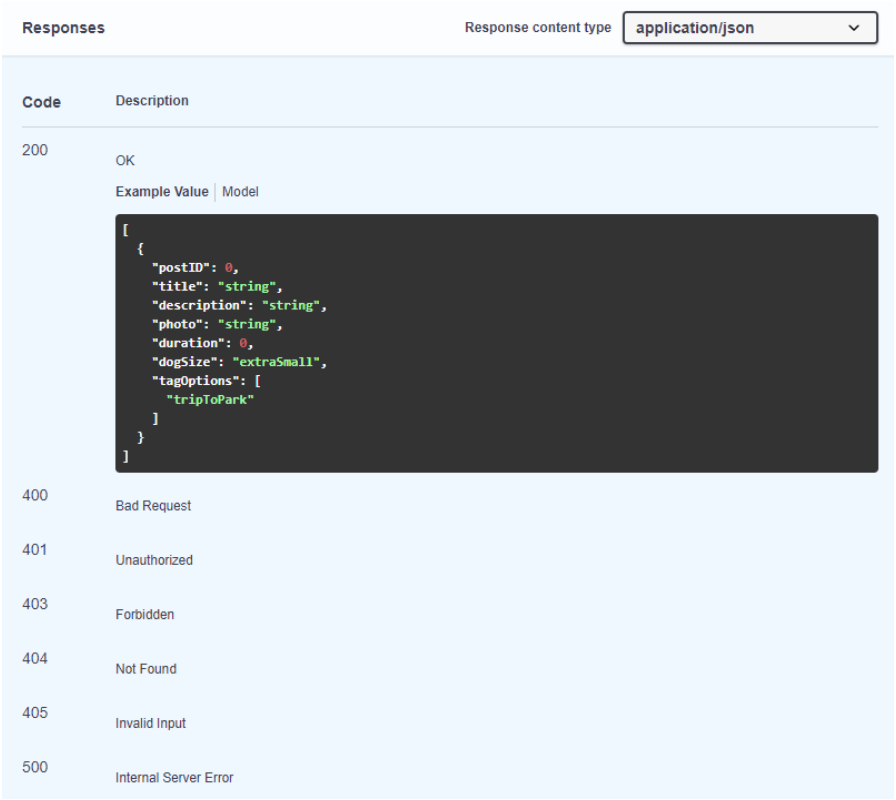



**2.2.5.2 Endpoint GET πόρου AllPosts (εμφάνιση όλων των δημοσιεύσεων)**

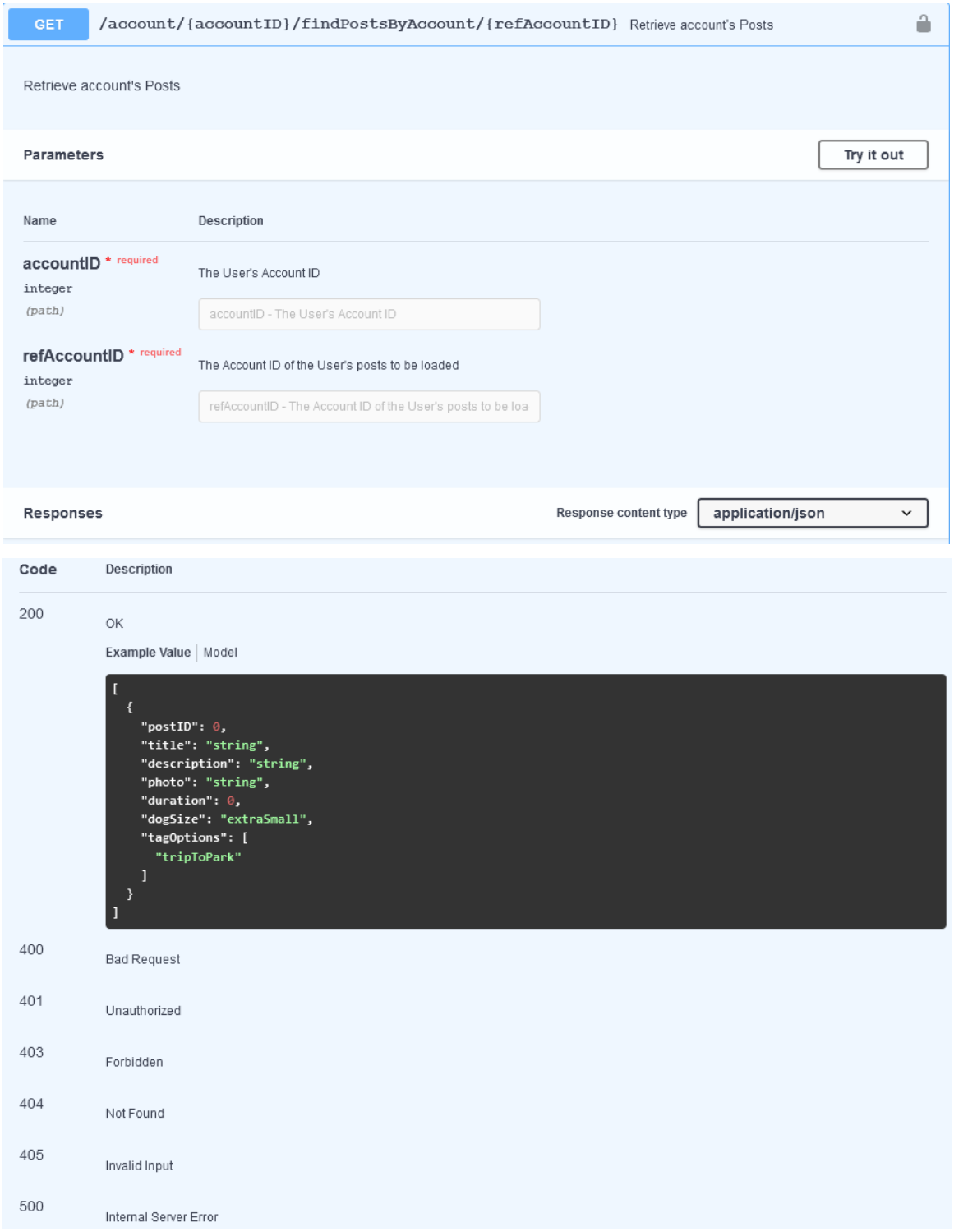


### 2.2.5.3 Endpoint GET πόρου AllPosts (εύρεση βάσει φίλτρου)



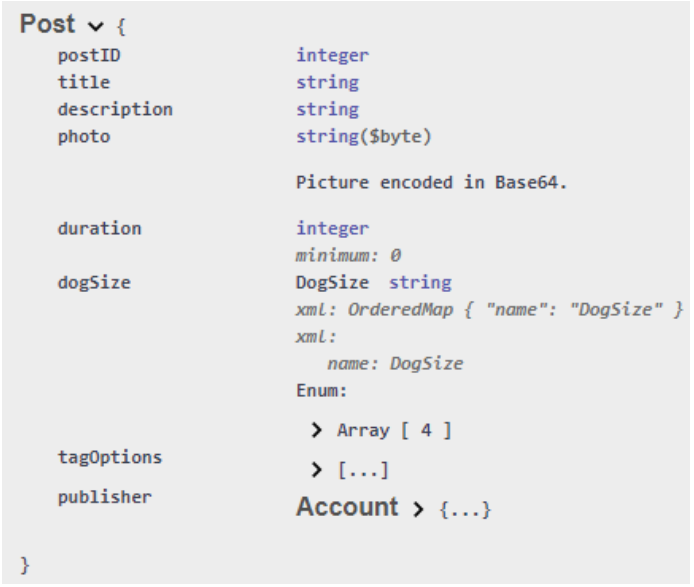



### 2.2.5.4 Endpoint GET πόρου AllPosts (εύρεση βάσει χρήστη)

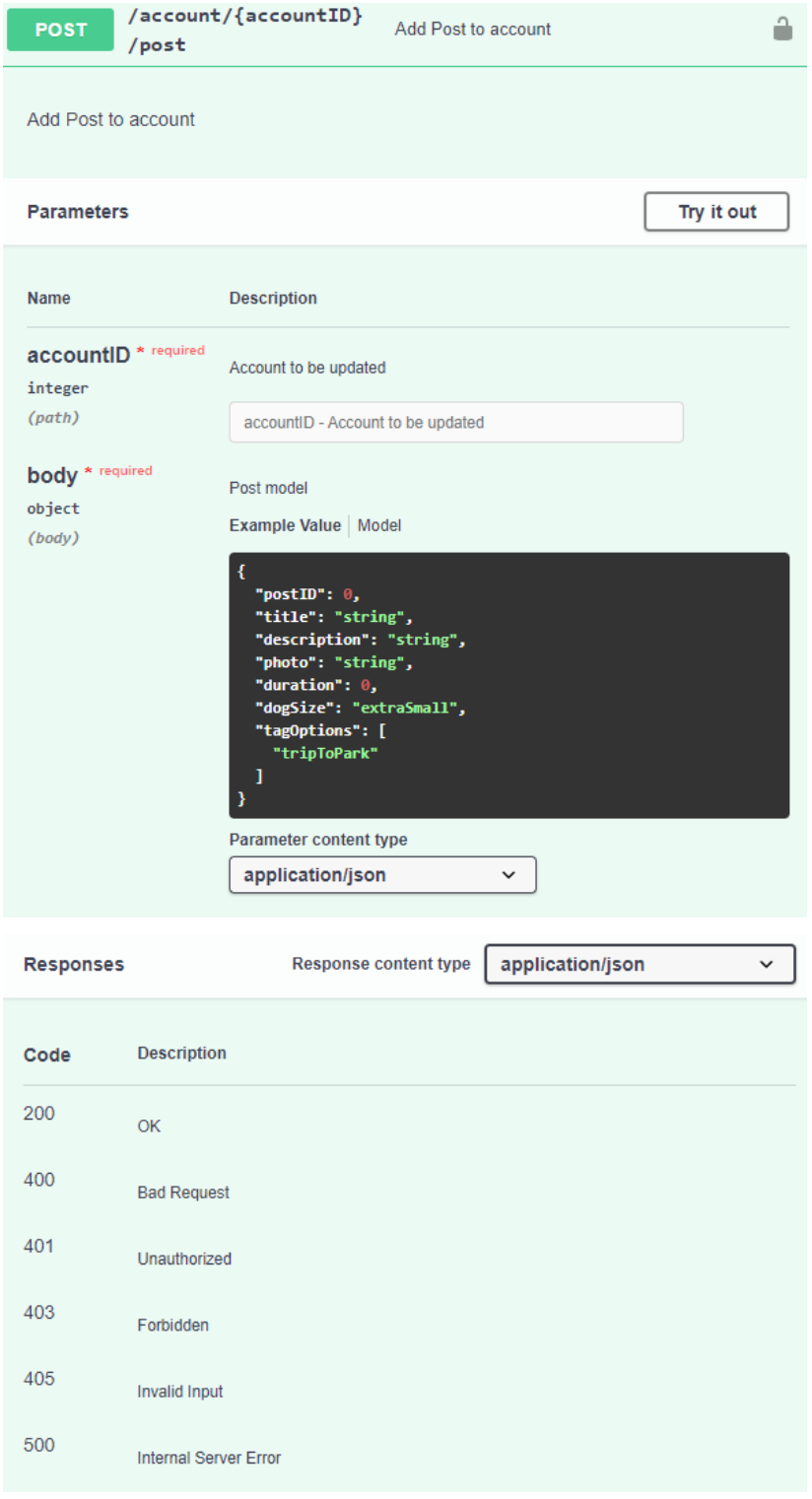


## 2.2.6 Πόρος Post

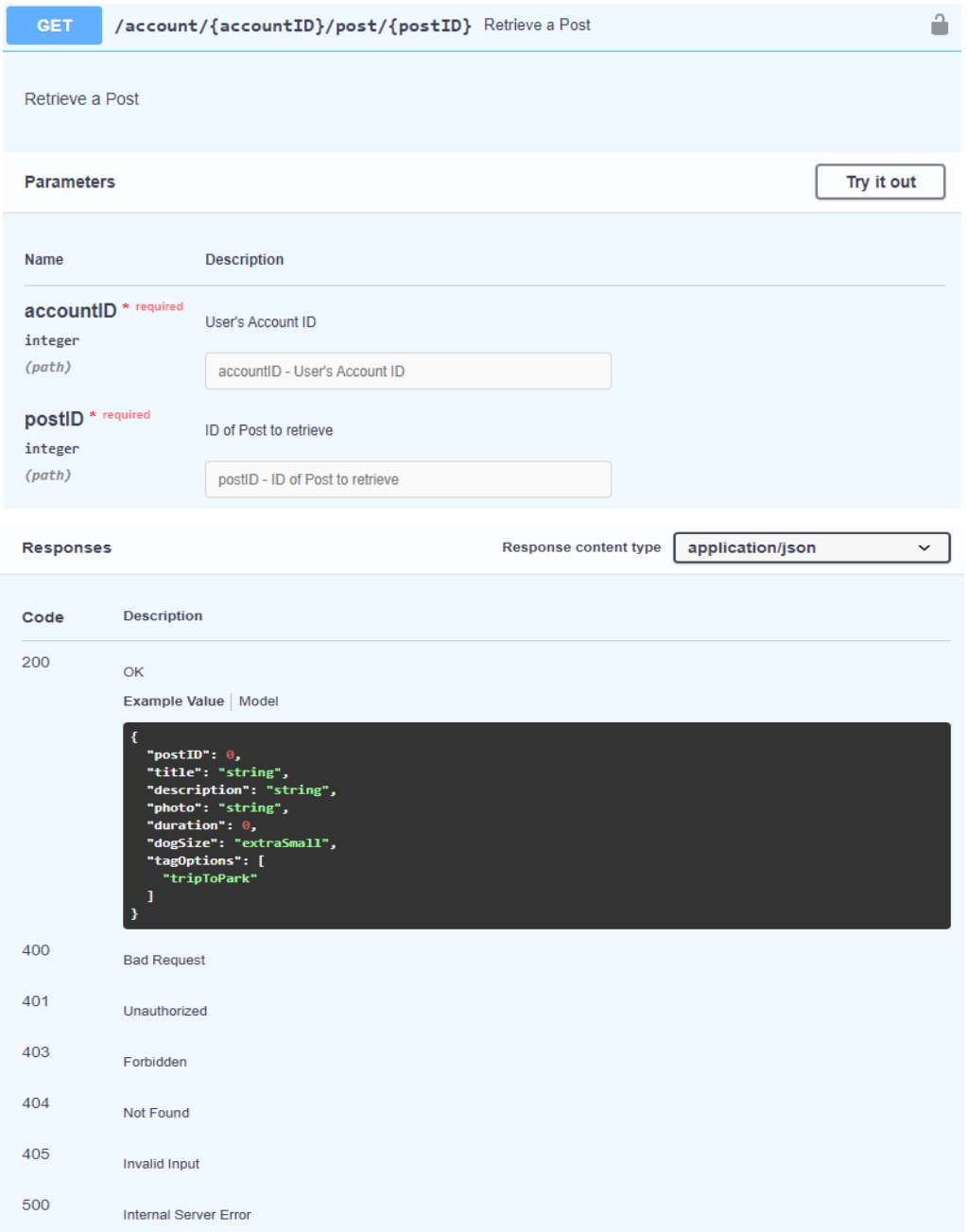
### 2.2.6.1 Μοντέλο δεδομένων Post



**2.2.6.2 Endpoint POST πόρου Post**

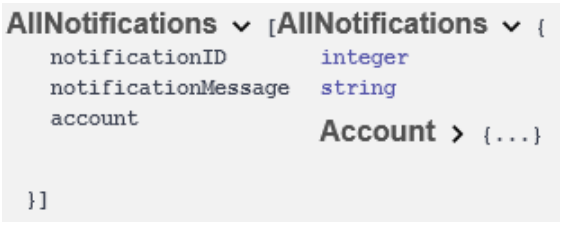


### 2.2.6.3 Endpoint GET πόρου Post

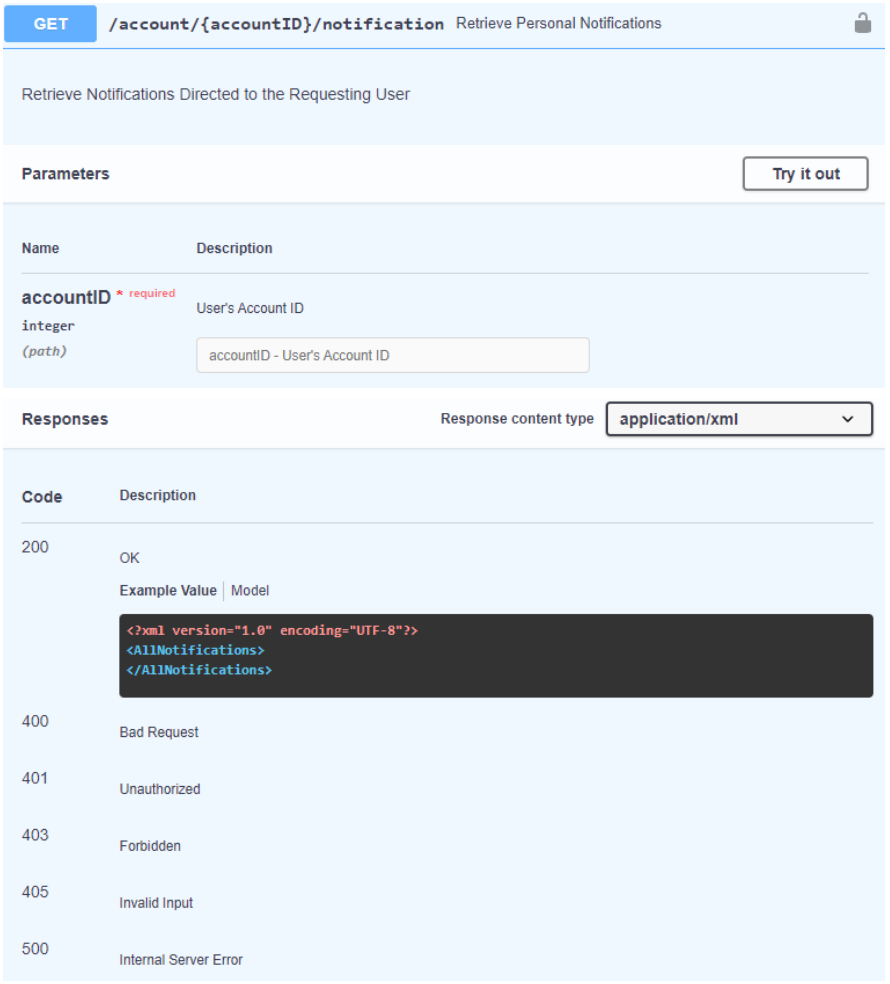


## 2.2.7 Πόρος AllNotifications

### 2.2.7.1 Μοντέλο δεδομένων AllNotifications

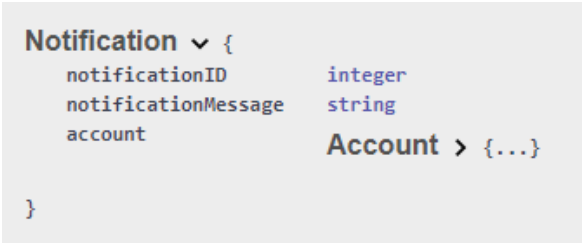



### 2.2.7.2 Endpoint GET πόρου Notification

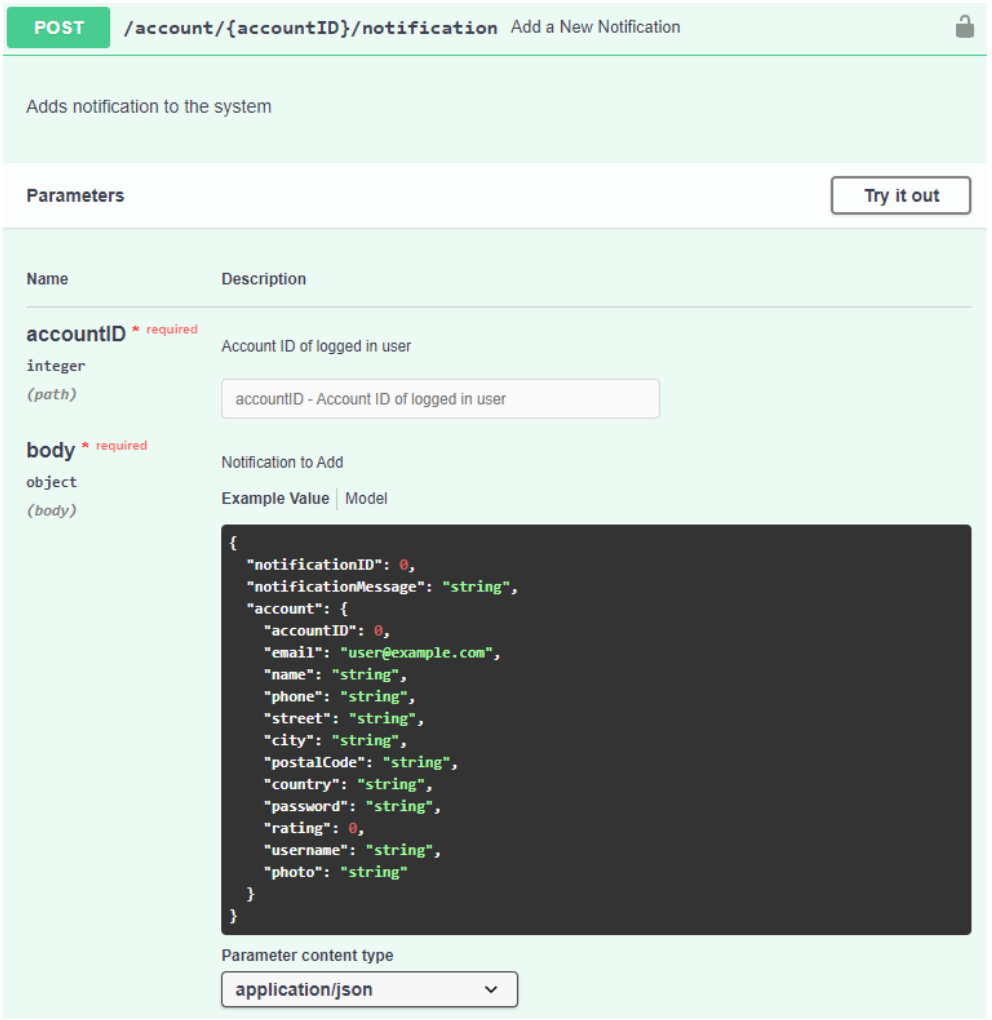


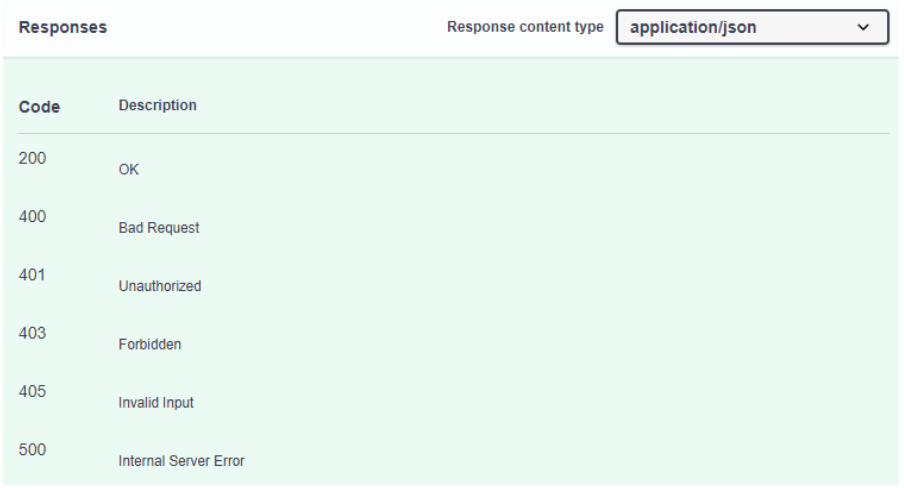
## 2.2.8 Πόρος Notification

### 2.2.8.1 Μοντέλο δεδομένων Notification

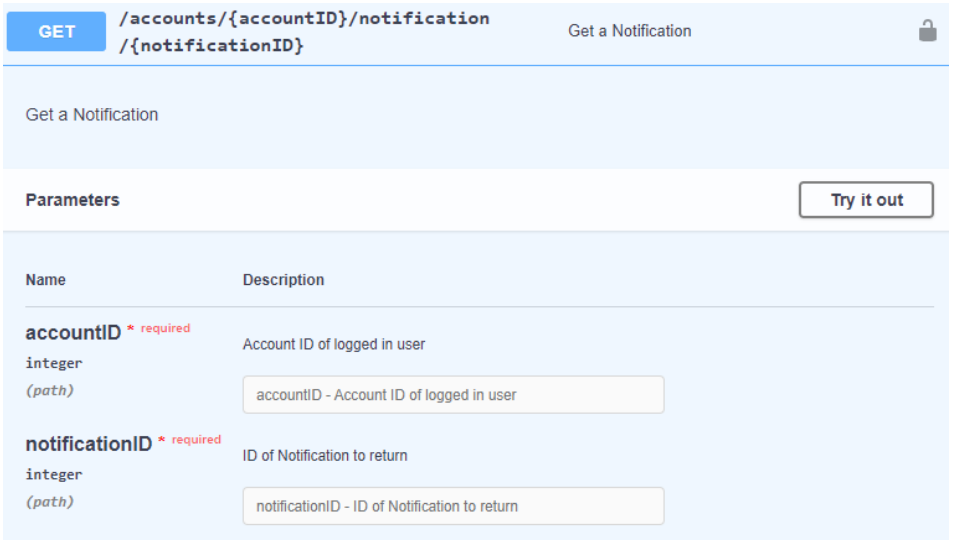



### 2.2.8.2 Endpoint POST πόρου Notification






### 2.2.8.3 Endpoint GET πόρου Notification



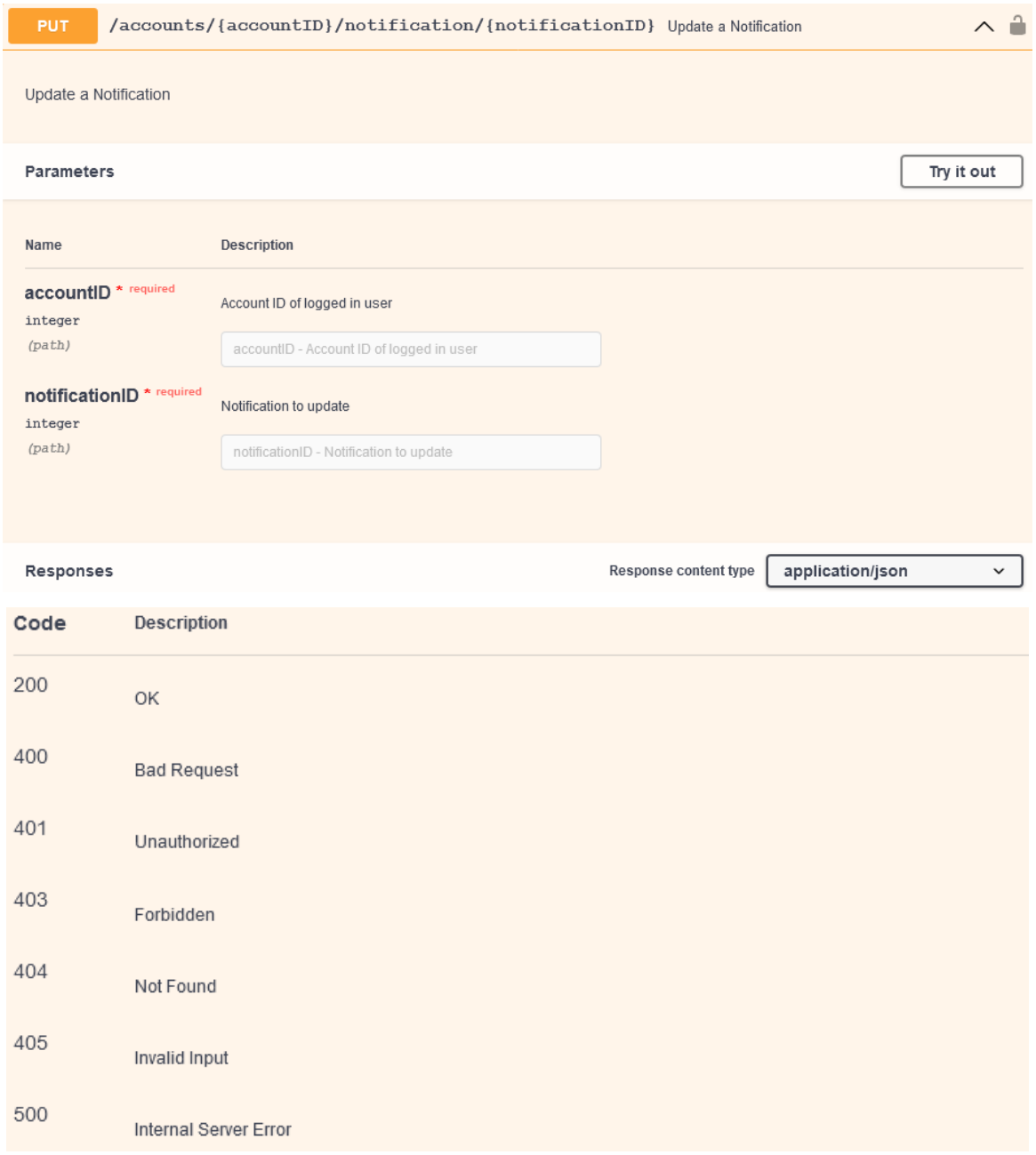
**Responses** Response content type application/json

| Code | Description |
|---|---|
| 200 | OK<br>Example Value \| Model |

```
{
  "notificationID": 0,
  "notificationMessage": "string",
  "account": {
    "accountID": 0,
    "email": "user@example.com",
    "name": "string",
    "phone": "string",
    "street": "string",
    "city": "string",
    "postalCode": "string",
    "country": "string",
    "password": "string",
    "rating": 0,
    "username": "string",
    "photo": "string"
  }
}
```

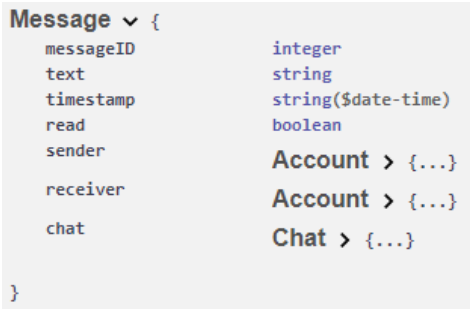
| | |
|---|---|
| 400 | Bad Request |
| 401 | Unauthorized |
| 403 | Forbidden |
| 404 | Not Found |
| 405 | Invalid Input |
| 500 | Internal Server Error |

### 2.2.8.4 Endpoint PUT πόρου Notification

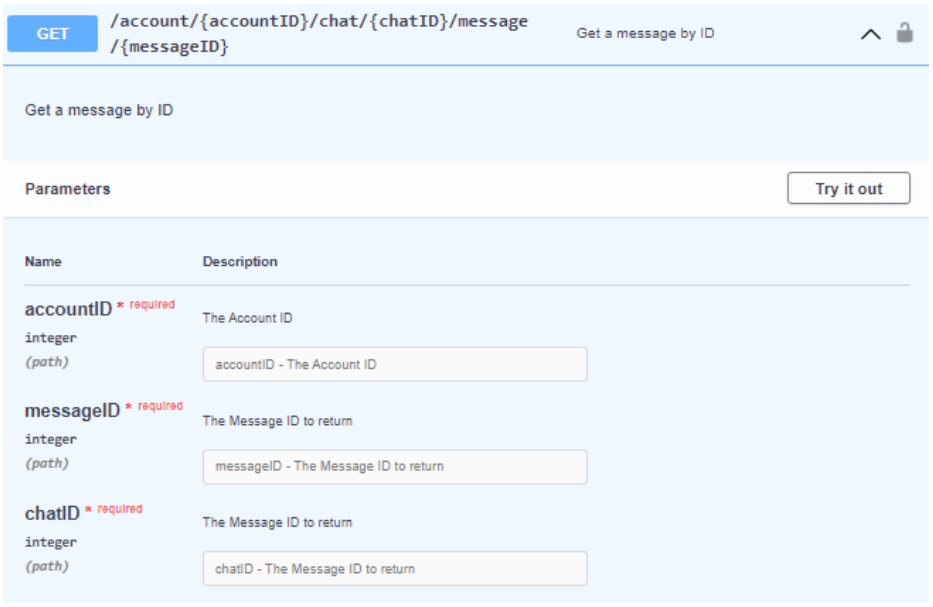


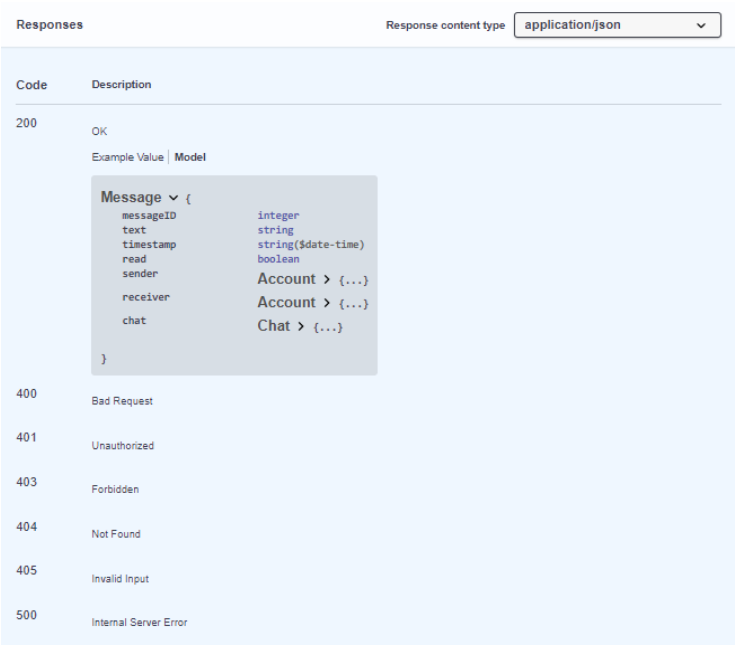
## 2.2.9 Πόρος Message

### 2.2.9.1 Μοντέλο δεδομένων Message

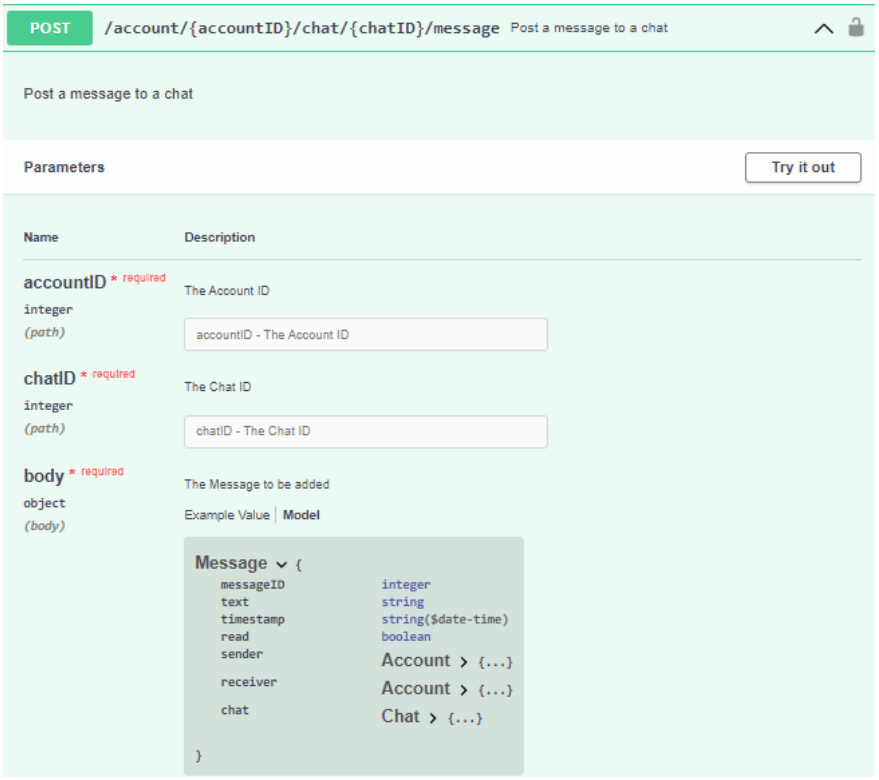



### 2.2.9.2 Endpoint GET πόρου Message





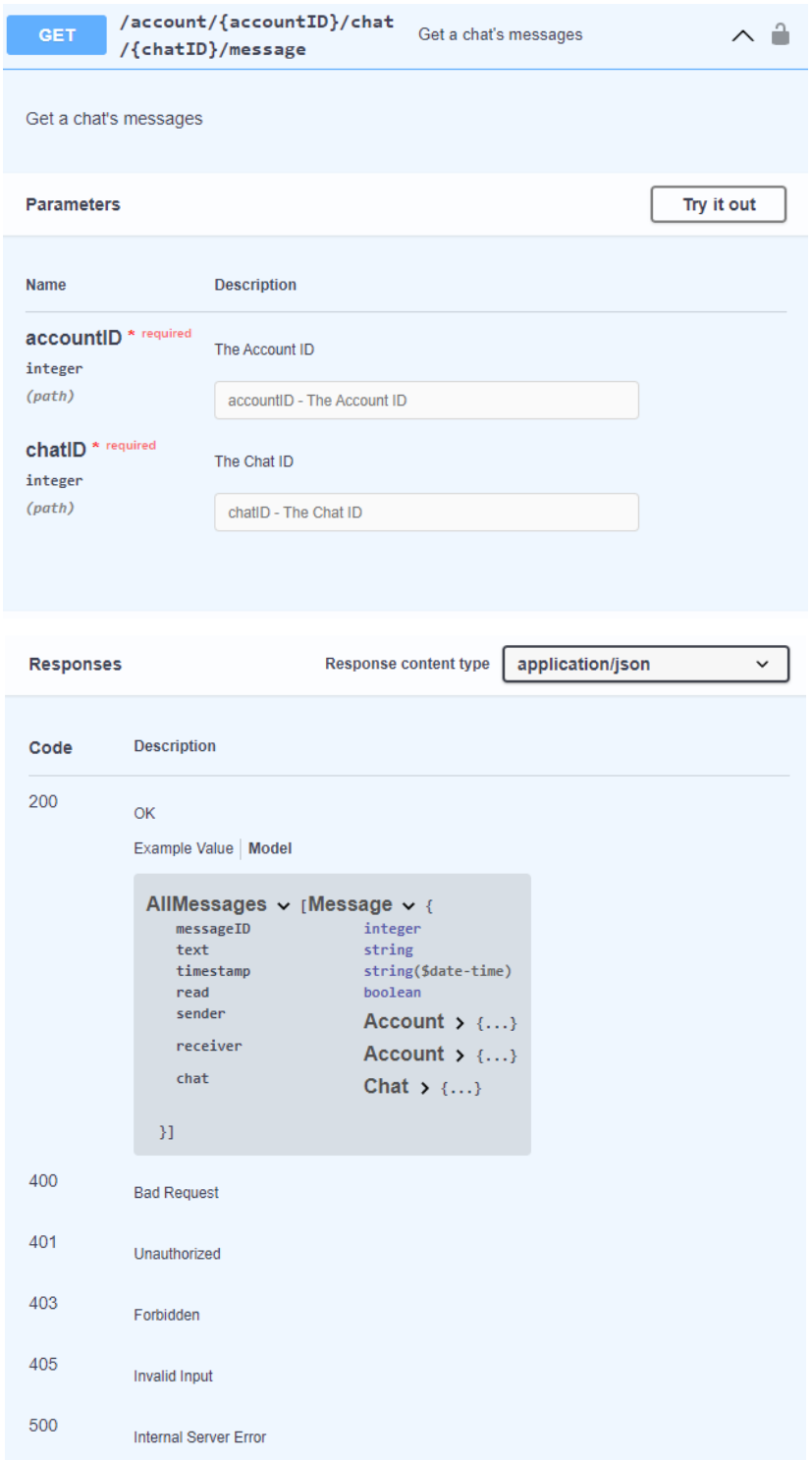

### 2.2.9.2 Endpoint POST πόρου Message




## 2.2.10 Πόρος AllMessages

### 2.2.10.1 Μοντέλο πόρου AllMessages

**2.2.10.2 Endpoint GET πόρου AllMessages**

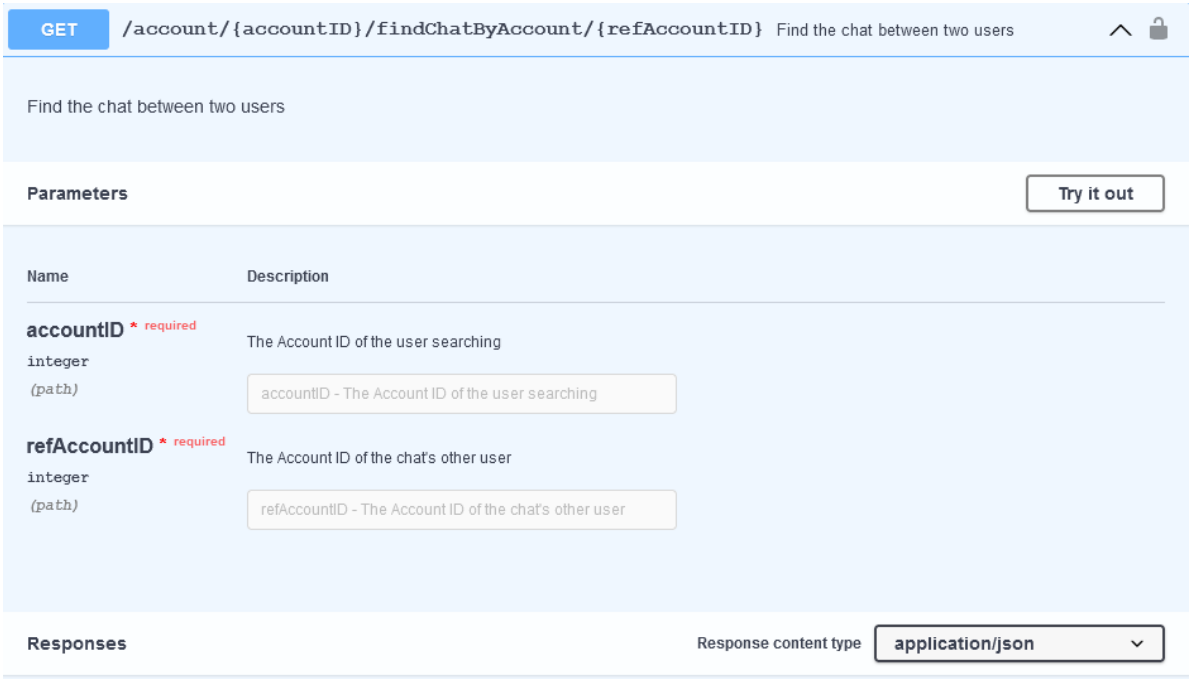


## 2.2.11 Πόρος Chat

### 2.2.11.1 Μοντέλο δεδομένων Chat

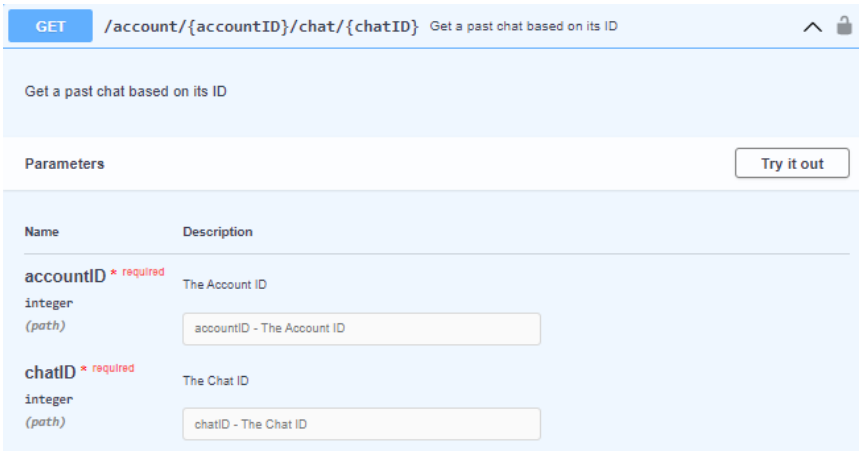
```
Chat ⌄ {
    chatID              integer
    unreadMessages      ⌄ [
                        minItems: 2
                        maxItems: 2
                        integer]
    lastMessage         ⌄ [
                        minItems: 2
                        maxItems: 2

                        Last Message for each one of the two accounts.

                        string]
    accounts            ⌄ [
                        minItems: 2
                        maxItems: 2
                        uniqueItems: true
                        Account › {...}]

}
```

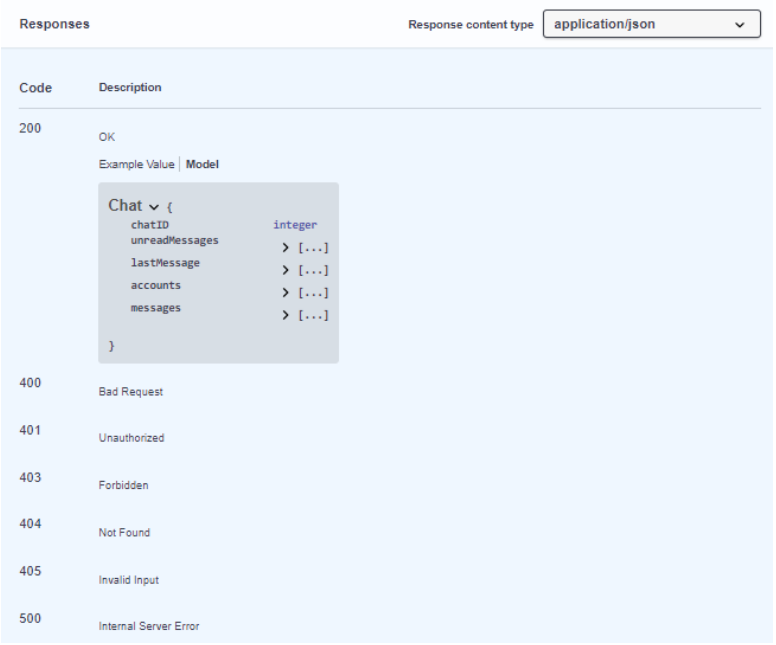
### 2.2.11.2 Endpoint GET πόρου Chat (εύρεση βάσει χρήστη)



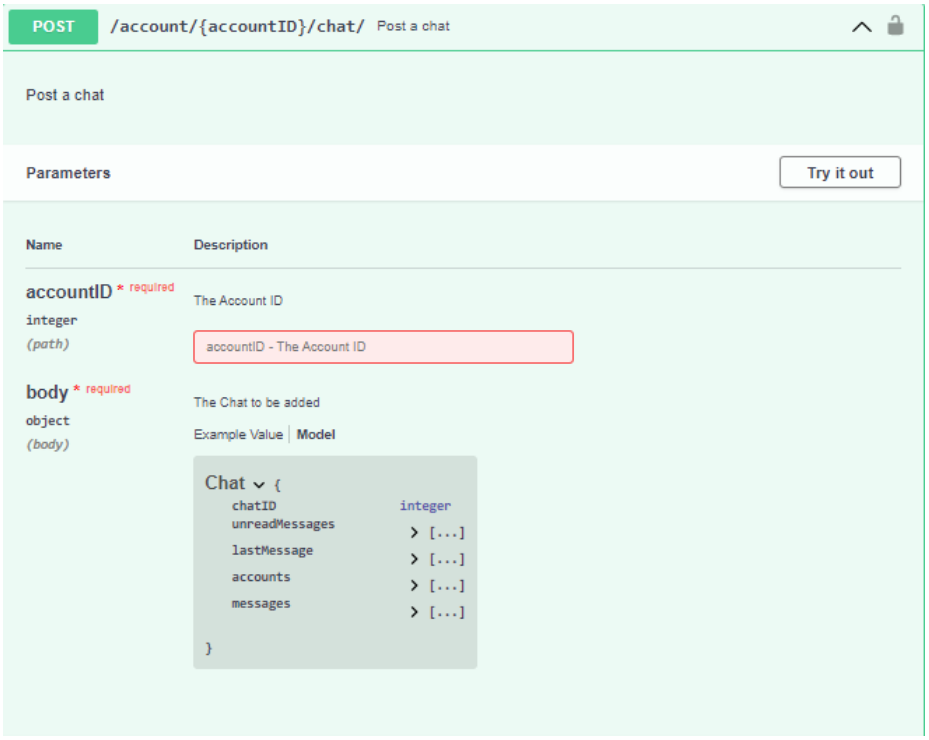
| Code | Description |
|---|---|
| 200 | OK<br>Example Value \| **Model**<br>**Chat** { chatID integer; unreadMessages [...]; lastMessage [...]; accounts [...]; messages [...] } |
| 400 | Bad Request |
| 401 | Unauthorized |
| 403 | Forbidden |
| 404 | Not Found |
| 405 | Invalid Input |
| 500 | Internal Server Error |

### 2.2.11.3 Endpoint GET πόρου Chat






### 2.2.11.4 Endpoint POST πόρου Chat



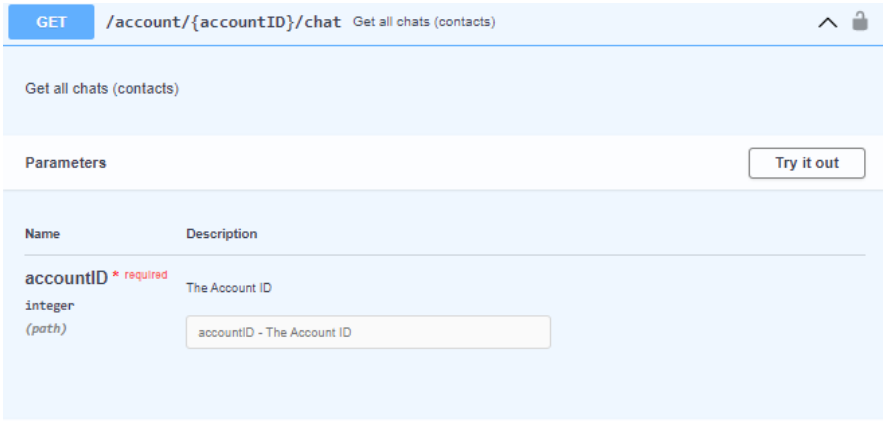
| Code | Description |
|---|---|
| 200 | OK |
| 400 | Bad Request |
| 401 | Unauthorized |
| 403 | Forbidden |
| 405 | Invalid Input |
| 500 | Internal Server Error |

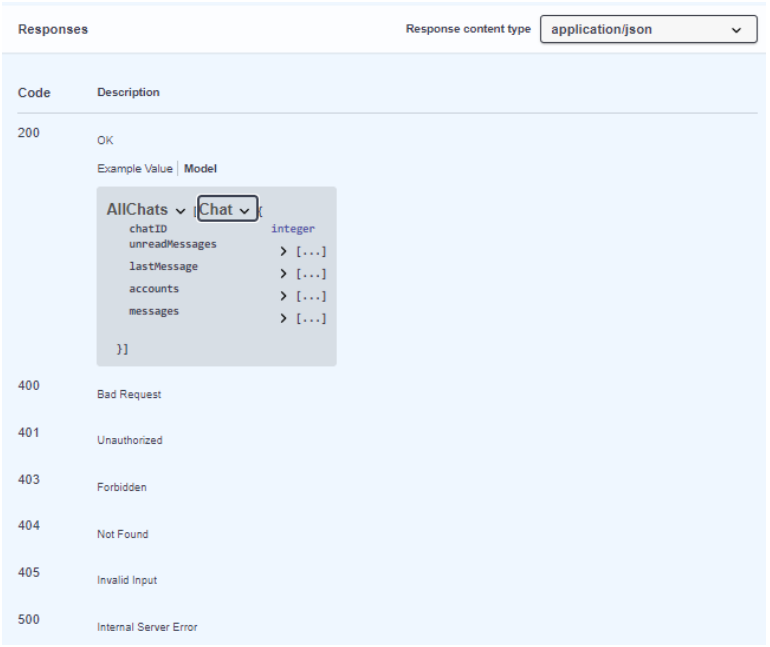
## 2.2.12 Πόρος AllChats

### 2.2.12.1 Μοντέλο πόρου AllChats

```
AllChats [AllChats {
    chatID              integer
    unreadMessages      [
                        minItems: 2
                        maxItems: 2
                        integer]
    lastMessage         [
                        minItems: 2
                        maxItems: 2

                        Last Message for each one of the two accounts.

                        string]
    accounts            [
                        minItems: 2
                        maxItems: 2
                        uniqueItems: true
                        Account > {...}]

}]
```

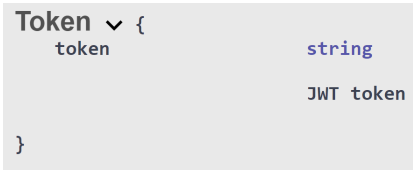
### 2.2.11.2 Endpoint GET πόρου AllChats



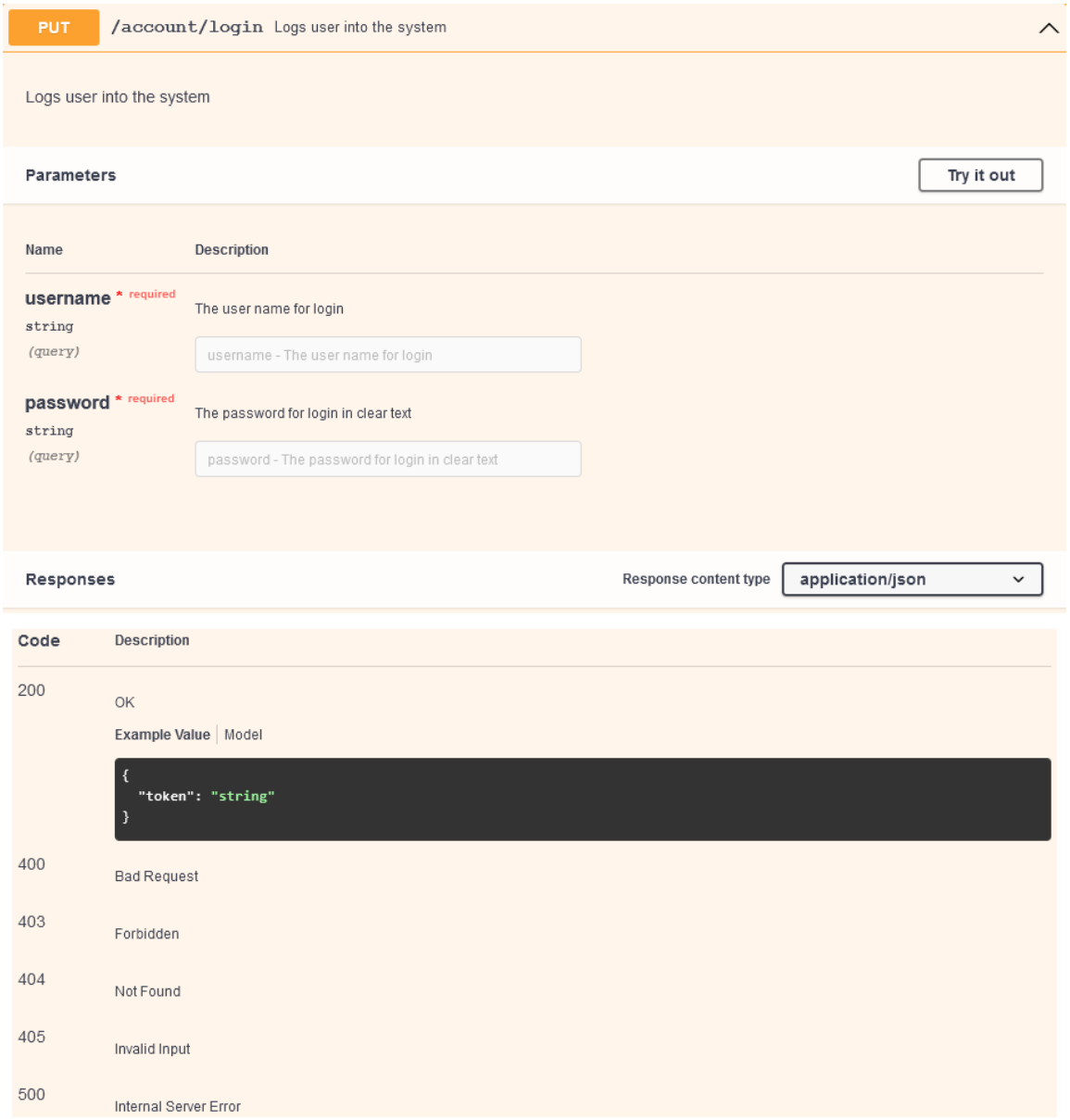



## 2.2.13 Πόρος Token

### 2.2.13.1 Μοντέλο Πόρου Token



### 2.2.13.2 Endpoint PUT πόρου Token



# 3. Υλοποίηση Συστήματος με Node-RED

Στο τρίτο κεφάλαιο γίνεται ανάλυση της υλοποίηση του συστήματος σε Node-RED. Παρακάτω θα γίνει αρχικά αντιστοίχιση των REST υπηρεσιών του κεφαλαίου 2, και στη συνέχεια θα υλοποιηθούν οι ιστορίες χρήστη, όπως αυτές περιγράφονται στο πρώτο παραδοτέο. Η ολοκληρωμένη υλοποίηση Node-RED μπορεί να βρεθεί στο σύνδεσμο:

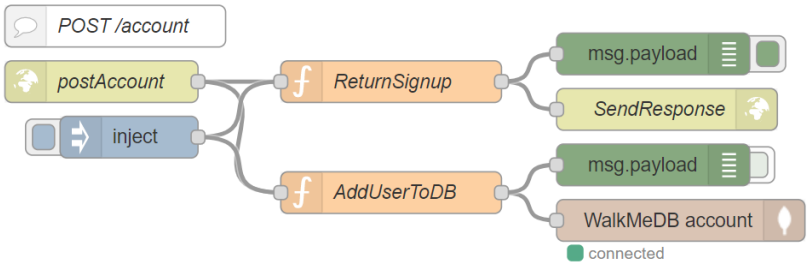
**Node-RED Implementation (zip)**

**Node-RED Implementation (json)**

## 3.1 Αντιστοίχιση REST υπηρεσιών σε Ροές Node-RED

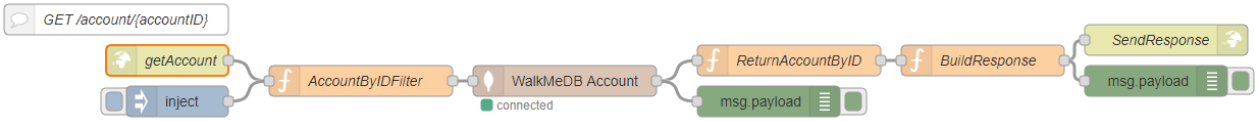
### 3.1.1 Ροές πόρου Account

#### 3.1.1.1 Ροή endpoint POST /account



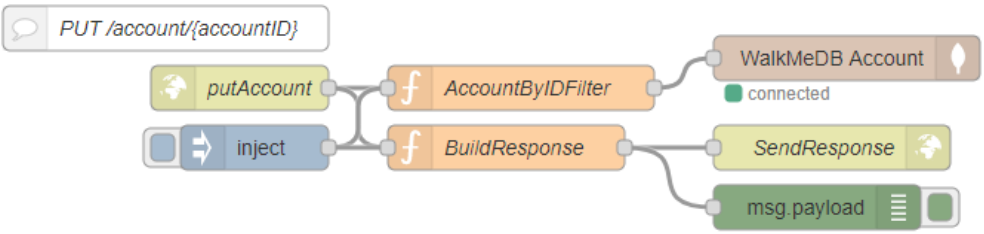

Η ροή αυτή είναι υπεύθυνη για την εγγραφή ενός χρήστη στο σύστημα, δηλαδή για τη δημιουργία λογαριασμού χρήστη. Δέχεται ως είσοδο στο σώμα (body) του αιτήματος ένα αντικείμενο τύπου Account, το εισάγει στη βάση δεδομένων μέσω της συνάρτησης AddUserToDB, και επιστρέφει την επιτυχία της εγγραφής στο χρήστη μέσω της συνάρτησης ReturnSignup.

#### 3.1.1.2 Ροή endpoint GET /account/{accountID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή του λογαριασμού ενός χρήστη. Μέσω της συνάρτησης AccountByIDFilter, στέλνεται στην βάση το ID του λογαριασμού που ζητείται και μέσω των ReturnAccountByID και BuildResponse χτίζεται η απάντηση που επιστρέφεται στο SendResponse.

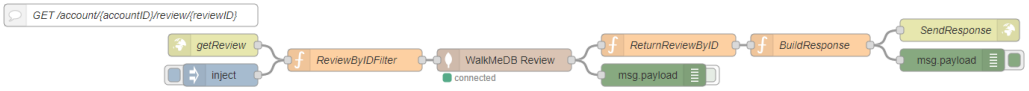
#### 3.1.1.3 Ροή endpoint PUT /account/{accountID}



Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την ενημέρωση του λογαριασμού ενός χρήστη. Μέσω της συνάρτησης AccountByIDFilter στέλνεται στην βάση ερώτηση για την ύπαρξη του λογαριασμού και στη συνέχεια μέσω αποθηκεύεται στην βάση ο ενημερωμένος λογαριασμός .Τέλος μέσω του BuildResponse χτίζεται η απάντηση που επιστρέφεται.

## 3.1.2 Ροές πόρου Review

### 3.1.2.1 Ροή endpoint GET /account/{accountID}/review/{reviewID}

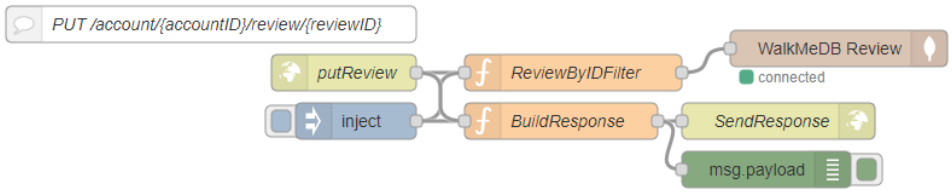



Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή κάποιας αξιολόγησης του χρήστη. Μέσω της συνάρτησης ReviewByIDFilter, στέλνεται στην βάση το ID της αξιολόγησης που ζητείται και μέσω των ReturnReviewByID και BuildResponse χτίζεται η απάντηση που επιστρέφεται στο SendResponse.

### 3.1.2.2 Ροή endpoint DELETE /account/{accountID}/review/{reviewID}

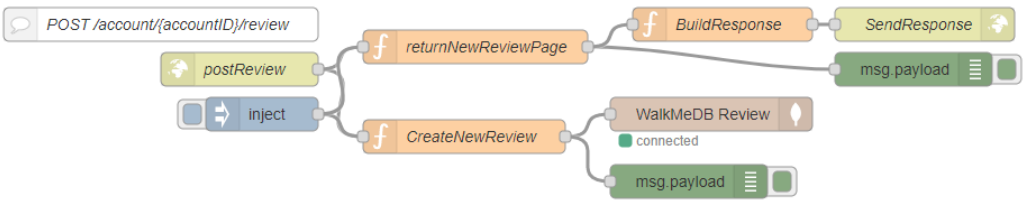

Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την διαγραφή μιας αξιολόγησης ενός χρήστη. Μέσω της συνάρτησης ReviewByIDFilter στέλνεται στην βάση ερώτηση για την ύπαρξη της αξιολόγησης και στη συνέχεια διαγράφεται από την βάση. Τέλος μέσω του BuildResponse χτίζεται η απάντηση που επιστρέφεται.

### 3.1.2.3 Ροή endpoint PUT /account/{accountID}/review/{reviewID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την ενημέρωση μιας αξιολόγησης ενός χρήστη. Μέσω της συνάρτησης ReviewtByIDFilter στέλνεται στην βάση ερώτηση για την ύπαρξη της αξιολόγησης και στη συνέχεια μέσω αποθηκεύεται στην βάση η ενημερωμένη αξιολόγηση. Τέλος μέσω του BuildResponse χτίζεται η απάντηση που επιστρέφεται.

### 3.1.2.4 Ροή endpoint POST /account/{accountID}/review

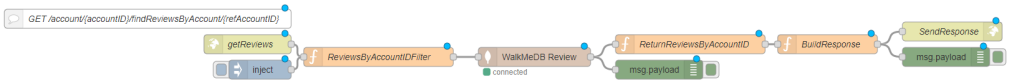



Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την καταχώρηση μιας αξιολόγησης σε έναν

χρήστη. Μέσω της συνάρτησης returnNewReviewPage και της BuildResponse χτίζεται η σελίδα που θα επιστρέψει η εφαρμογή, ενώ μέσω της CreateNewReview στέλνεται η πληροφορία για την καταχώρηση της αξιολόγησης στην βάση δεδομένων.

### 3.1.3 Ροές πόρου AllReviews

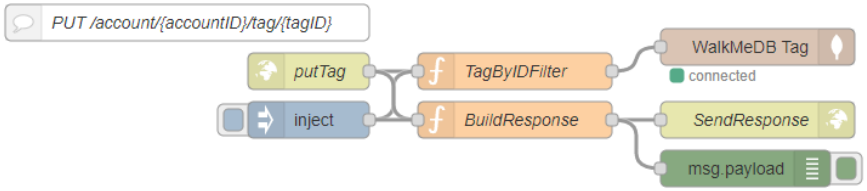
#### 3.1.3.1 Ροή endpoint GET /account/{accountID}/findReviewsByAccount/{refAccountID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή των αξιολογήσεων ενός χρήστη. Μέσω της συνάρτησης ReviewsByAccountFilter, στέλνεται στην βάση το ID του λογαριασμού και μέσω των ReturnReviewsByAccountID και BuildResponse χτίζεται η απάντηση που επιστρέφεται στο SendResponse.

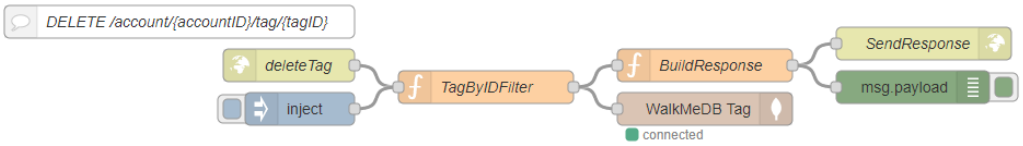
### 3.1.4 Ροές πόρου Tag

#### 3.1.4.1 Ροή endpoint PUT /account/{accountID}/tag/{tagID}



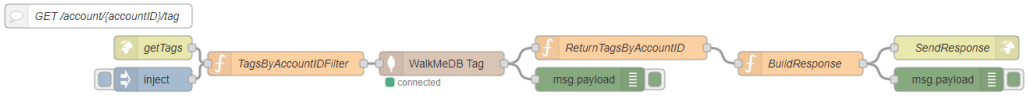

Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την ενημέρωση μιας ετικέτας ενός χρήστη. Μέσω της συνάρτησης TagByIDFilter στέλνεται στην βάση ερώτηση για την ύπαρξη της ετικέτας και στη συνέχεια αποθηκεύεται στην βάση ο ενημερωμένος λογαριασμός. Τέλος μέσω του BuildResponse χτίζεται η απάντηση που επιστρέφεται.

#### 3.1.4.2 Ροή endpoint DELETE /account/{accountID}/tag/{tagID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την διαγραφή μιας ετικέτας ενός χρήστη. Μέσω της συνάρτησης TagByIDFilter στέλνεται στην βάση ερώτηση για την ύπαρξη της αξιολόγησης και στη συνέχεια διαγράφεται από την βάση. Τέλος μέσω του BuildResponse χτίζεται η απάντηση που επιστρέφεται.

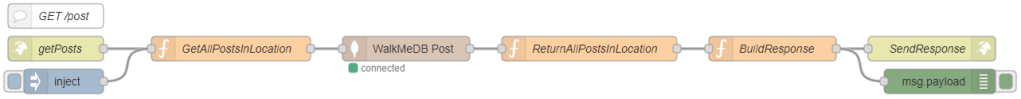
#### 3.1.4.3 Ροή endpoint GET /account/{accountID}/tag




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή των ταμπελών ενός χρήστη. Μέσω της συνάρτησης TagsByAccountIDFilter, στέλνεται στην βάση το ID του λογαριασμού που ζητείται και μέσω των ReturnTagsByAccountID και BuildResponse χτίζεται η απάντηση που επιστρέφεται στο SendResponse.

## 3.1.5 Ροές πόρου AllPosts

### 3.1.5.1 Ροή endpoint GET /post

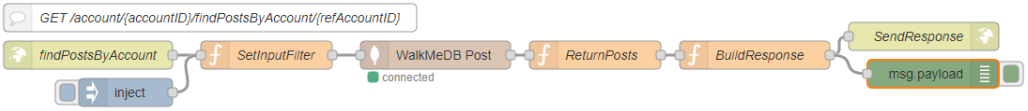



Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή όλων των δημοσιεύσεων εντός εμβέλειας του χρήστη. Μέσω της συνάρτησης GetAllPostsInLocation στέλνεται στη βάση η εντολή και στη συνέχεια μέσω των ReturnAllPostsInLocation και BuildResponse χτίζεται η απάντηση που επιστρέφεται.

### 3.1.5.2 Ροή endpoint GET /account/{accountID}/findPostsByTag/{tagID}

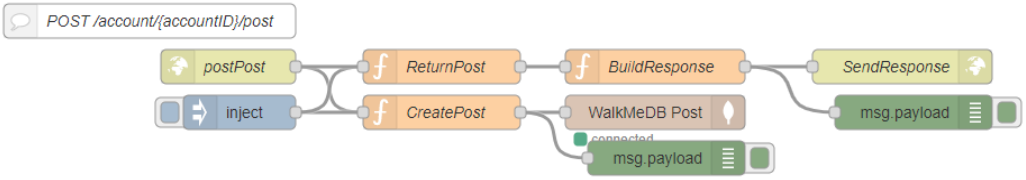

Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή των δημοσιεύσεων με συγκεκριμένη ετικέτα. Μέσω της συνάρτησης GetPostsByTagID στέλνεται στην βάση το ID της ετικέτας και στη συνέχεια μέσω των ReturnPostsByTagID και BuildResponse χτίζεται η απάντηση που επιστρέφεται.

### 3.1.5.3 Ροή endpoint GET /account/{accountID}/findPostsByAccount/{refAccountID}




## 3.1.6 Ροές πόρου Post

### 3.1.6.1 Ροή endpoint POST /account/{accountID}/post



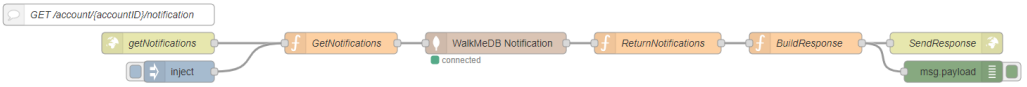

Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για τη δημιουργία μιας δημοσίευσης ενός χρήστη. Μέσω της συνάρτησης CreatePost στέλνεται στην βάση το ID του χρήστη και στη συνέχεια μέσω των ReturnPost και BuildResponse χτίζεται η απάντηση που επιστρέφεται.

### 3.1.6.2 Ροή endpoint GET /account/{accountID}/post/{postID}


Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή μιας δημοσίευσης ενός χρήστη. Μέσω της συνάρτησης GetPostsByPostID στέλνεται στην βάση το ID της δημοσίευσης και στη συνέχεια μέσω των ReturnPostsByPostID και BuildResponse χτίζεται η απάντηση που επιστρέφεται.

## 3.1.7 Ροές πόρου AllNotifications

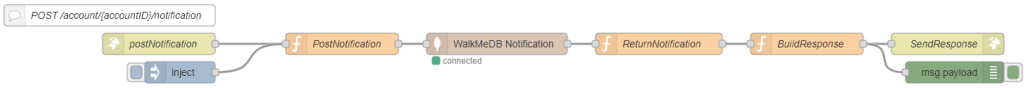
### 3.1.7.1 Ροή endpoint GET /account/{accountID}/notification




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή των ειδοποιήσεων ενός χρήστη. Μέσω της συνάρτησης GetNotification στέλνεται στην βάση το ID του χρήστη και στη συνέχεια μέσω των ReturnNotification και BuildResponse χτίζεται η απάντηση που επιστρέφεται.

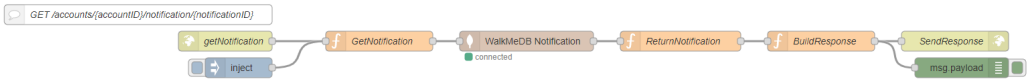
## 3.1.8 Ροές πόρου Notification

### 3.1.8.1 Ροή endpoint POST /account/{accountID}/notification



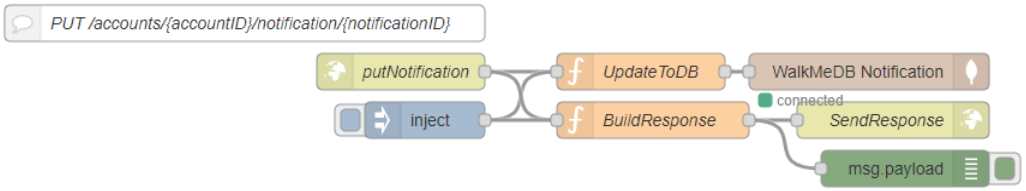

Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για τη δημιουργία μιας ειδοποίησης. Μέσω της συνάρτησης PostNotification στέλνεται στην βάση το ID του χρήστη και στη συνέχεια μέσω των ReturnNotification και BuildResponse χτίζεται η απάντηση που επιστρέφεται.

### 3.1.8.2 Ροή endpoint GET /account/{accountID}/notification/{notificationID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή μιας ειδοποίησης ενός χρήστη. Μέσω της συνάρτησης GetNotification στέλνεται στην βάση το ID της ειδοποίησης και στη συνέχεια μέσω των ReturnNotification και BuildResponse χτίζεται η απάντηση που επιστρέφεται.

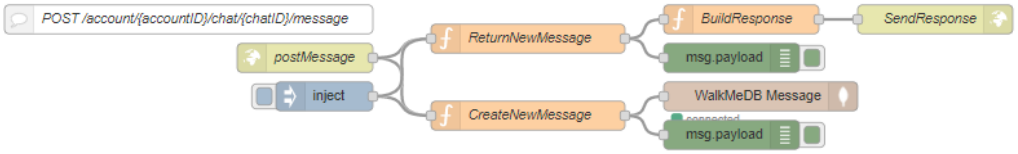
### 3.1.8.3 Ροή endpoint PUT /account/{accountID}/notification/{notificationID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την ενημέρωση μιας ειδοποίησης ενός χρήστη. Μέσω της συνάρτησης PutNotification στέλνεται στην βάση ερώτηση για την ύπαρξη της ειδοποίησης και στη συνέχεια μέσω της ReturnNotification αποθηκεύεται στην βάση ο ενημερωμένη ειδοποίηση. Τέλος μέσω του BuildResponse χτίζεται η απάντηση που επιστρέφεται.

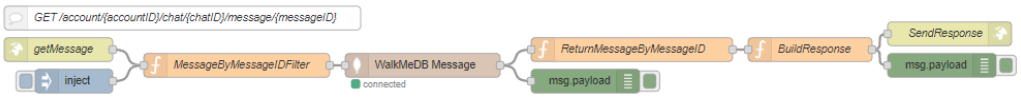
## 3.1.9 Ροές πόρου Message

### 3.1.9.1 Ροή endpoint POST /account/{accountID}/chat/{chatID}/message



Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την αποστολή ενός μηνύματος από έναν χρήστη σε έναν άλλο χρήστη. Μέσω της συνάρτησης returnNewMessage και της BuildResponse χτίζεται η σελίδα που θα επιστρέψει η εφαρμογή, ενώ μέσω της CreateNewMessage στέλνεται η πληροφορία για την καταχώρηση του μηνύματος στην βάση δεδομένων.

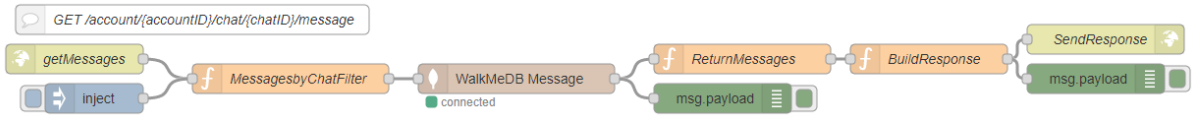
#### 3.1.9.2 Ροή endpoint GET /account/{accountID}/chat/{chatID}/message/{messageID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή ενός μηνύματος που έλαβε ο χρήστης από κάποιον άλλο χρήστη. Μέσω της συνάρτησης AccountByIDFilter, στέλνεται στην βάση το ID του λογαριασμού που ζητείται και μέσω των ReturnAccountByID και BuildResponse χτίζεται η απάντηση που επιστρέφεται στο SendResponse.

### 3.1.10 Ροές πόρου AllMessages

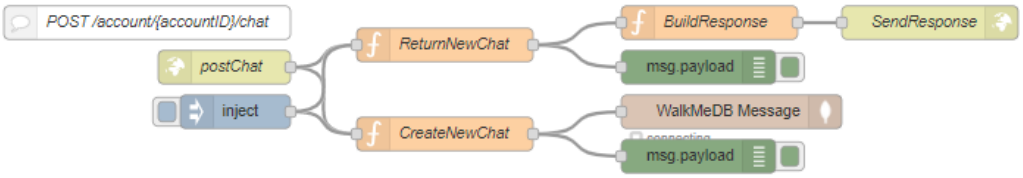
#### 3.1.10.1 Ροή endpoint GET /account/{accountID}/chat/{chatID}/message




Η ροή αυτή υλοποιεί την επιστροφή μηνυμάτων που ανήκουν σε μια συνομιλία. Η συνάρτηση MessagesbyChatFilter αναζητά στη βάση δεδομένων μηνύματα που ανήκουν στη συνομιλία, ενώ η ReturnMessages επιστρέφει αυτά τα μηνύματα. Η BuildResponse δομεί την απάντηση προς το χρήστη.

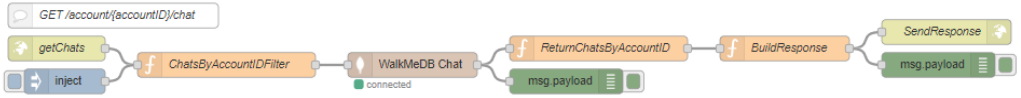
### 3.1.11 Ροές πόρου Chat

#### 3.1.11.1 Ροή endpoint POST /account/{accountID}/chat



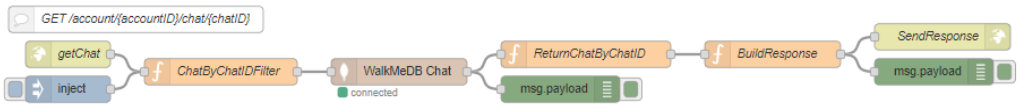

Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την δημιουργία μιας συνομιλίας μεταξύ δυο χρηστών. Μέσω της συνάρτησης returnNewChat και της BuildResponse χτίζεται η σελίδα που θα επιστρέψει η εφαρμογή, ενώ μέσω της CreateNewChat στέλνεται η πληροφορία για την δημιουργία της συνομιλίας στην βάση δεδομένων.

#### 3.1.11.2 Ροή endpoint GET /account/{accountID}/chat




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή όλων των επαφών του χρήστη. Μέσω της συνάρτησης AccountByIDFilter, στέλνεται στην βάση το ID του λογαριασμού που ζητείται και μέσω των ReturnAccountByID και BuildResponse χτίζεται η απάντηση που επιστρέφεται στο SendResponse.

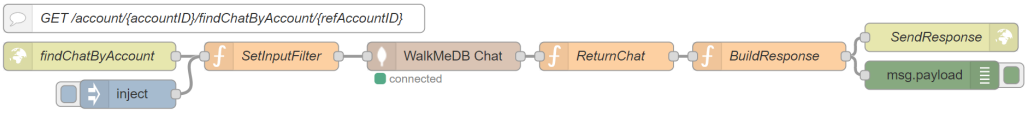
#### 3.1.11.3 Ροή endpoint GET /account/{accountID}/chat/{chatID}




Η ροή που υλοποιεί την υπηρεσία, είναι υπεύθυνη για την επιστροφή της συνομιλίας του χρήστη με κάποιον άλλο χρήστη. Μέσω της συνάρτησης AccountByIDFilter, στέλνεται στην βάση το ID του λογαριασμού που ζητείται και μέσω των ReturnAccountByID και BuildResponse χτίζεται η απάντηση που επιστρέφεται στο SendResponse.

### 3.1.12 Ροές πόρου AllChats

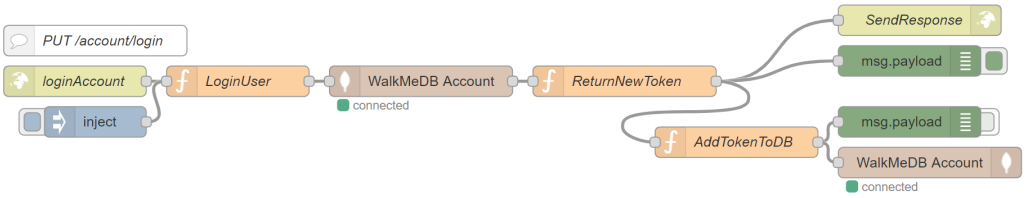
#### 3.1.12.1 Ροή endpoint GET /account/{accountID}/findChatByAccount/{refAccountID}




Η ροή αυτή είναι υπεύθυνη για την επιστροφή μιας συνομιλίας μεταξύ δύο χρηστών. Μέσω της συνάρτησης SetInputFilter γίνεται αναζήτηση στη βάση δεδομένων, ενός αντικειμένου Chat που αντιστοιχεί στους δυο χρήστες. Στη συνέχεια μέσω των ReturnChat και BuildResponse χτίζεται η απάντηση που θα επιστραφεί από το σύστημα.

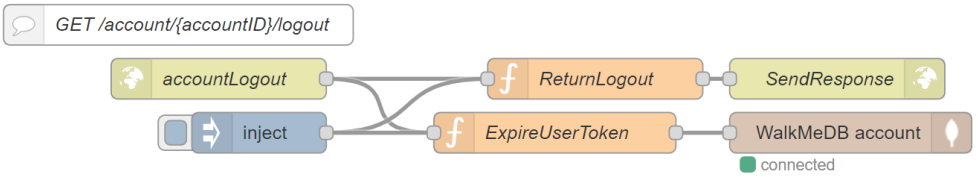
### 3.1.13 Ροές πόρου Token

#### 3.1.13.1 Ροή endpoint PUT /account/login




Η ροή αυτή δίνει τη δυνατότητα αυθεντικοποίησης χρηστών με το σύστημα. Δέχεται ως είσοδο τα στοιχεία σύνδεσης του χρήστη, η συνάρτηση LoginUser αναζητά στη βάση δεδομένων το χρήστη, η ReturnNewToken κατασκευάζει ένα νέο Token και μια απάντηση προς το αίτημα του χρήστη, και η AddTokenToDB προσθέτει το Token αυτό στη βάση δεδομένων.

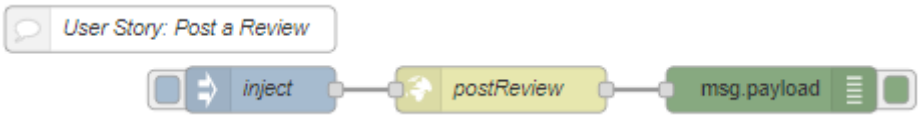
#### 3.1.13.2 Ροή endpoint GET /account/{accountID}/logout




Η ροή αυτή αποσυνδέει έναν χρήστη από το σύστημα. Αποδέχεται, μέσω των επικεφαλίδων του αιτήματος τους χρήστη, το ενεργό Token του χρήστη. Στη συνέχεια η ExpireUserToken διαγράφει το τελευταίο από τη βάση δεδομένων. Η ReturnLogout κατασκευάζει το αντικείμενο απάντησης του συστήματος προς το χρήστη.

# 3.2 Υλοποίηση Ιστοριών χρήστη

## 3.2.1 Ιστορία Χρήστη Post Review




Η ροή αυτή αποτελεί μια αναπαράσταση της διαδικασίας υποβολής αξιολόγησης ενός χρήστη. Η ροή εκκινείται με το πάτημα του κουμπιού υποβολής αξιολόγησης, μέσω του οποίου αποστέλλονται τα στοιχεία αξιολόγησης στο endpoint “postReview” του συστήματος.

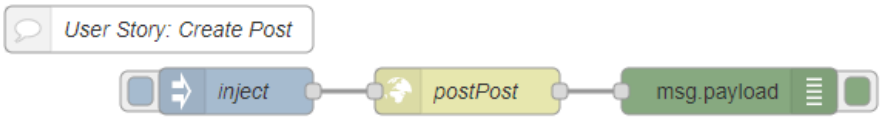
| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας αξιολογήσεων του χρήστη. |
| postReview | http-request | Κάνει κλήση προς την υπηρεσία postReview, η οποία δημοσιεύει την αξιολόγηση ενός χρήστη. |
| msg.payload | debug | Τυπώνει στη κονσόλα την απάντηση του συστήματος στο συγκεκριμένο αίτημα. |

## 3.2.2 Ιστορία Χρήστη Manage Review


Η ροή αυτή αποτελεί μια αναπαράσταση της διαδικασίας επεξεργασίας μιας αξιολόγησης ενός χρήστη. Η ροή εκκινείται με το πάτημα του κουμπιού αξιολόγησης, μέσω του οποίου που αποστέλλονται τα στοιχεία σύνδεσης του χρήστη στο endpoint “putReview” του συστήματος.

| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας αξιολόγησης του χρήστη. |
| putReview | http-request | Κάνει κλήση προς την υπηρεσία putReview, η οποία τροποποιεί την αξιολόγηση ενός χρήστη. |
| msg.payload | debug | Τυπώνει στη κονσόλα την απάντηση του συστήματος στο συγκεκριμένο αίτημα. |

## 3.2.3 Ιστορία Χρήστη Create Post

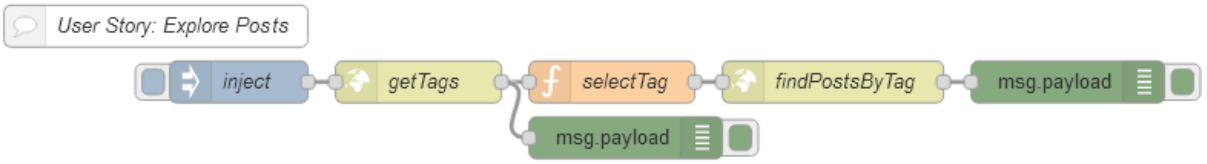



Ροή μέσω της οποίας ο χρήστης μπορεί να δημιουργήσει μια νέα δημοσίευση. Η ροή εκκινείται με το

πάτημα του κουμπιού δημιουργίας δημοσίευσης, μέσω του οποίου αποστέλλονται τα στοιχεία της δημοσίευσης χρήστη στο endpoint "postPost" του συστήματος.

| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της αρχικής σελίδας. |
| postPost | http-request | Κάνει κλήση προς την υπηρεσία postPost, η οποία επιστρέφει τη δημοσίευση του χρήστη. |
| mg.payload | debug | Τυπώνει στην κονσόλα το response του συστήματος στην κλήση του χρήστη, δηλαδή τη δημοσίευσή του. |

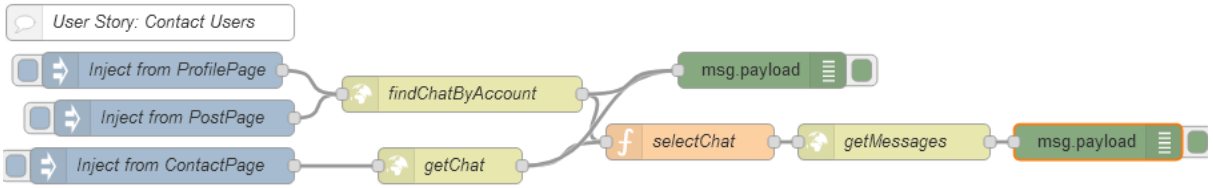
### 3.2.4 Ιστορία Χρήστη Explore Posts




Ροή μέσω της οποίας ο χρήστης μπορεί να προβάλλει τις δημοσιεύσεις άλλων χρηστών. Η ροή εκκινείται με το άνοιγμα της οθόνης εξερεύνησης δημοσιεύσεων. Τότε η συνάρτηση getTags λαμβάνει τις ετικέτες (φίλτρα) του χρήστη, και αφού η selectTag επιλέξει τις ετικέτες, γίνεται κλήση της findPostsByTag για λήψη των αντίστοιχων δημοσιεύσεων που πληρούν τα φίλτρα αυτά.

| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας εξερεύνησης δημοσιεύσεων. |
| getTags | http-request | Κάνει κλήση προς την υπηρεσία getTags, η οποία επιστρέφει όλες τις ετικέτες του χρήστη |
| selectTag | function | Συνάρτηση μέσω της οποίας επιλέγεται η εκάστοτε ετικέτα με βάση την είσοδο της συνάρτησης. |
| findPostsByTag | http-request | Κάνει κλήση προς την υπηρεσία findPostByTag, η οποία επιστρέφει όλες τις δημοσιεύσεις με τις συγκεκριμένες ετικέτες. |
| msg.payload | debug | Τυπώνει στην κονσόλα τις απαντήσεις του συστήματος στα αιτήματα του χρήστη. |

### 3.2.5 Ιστορία Χρήστη Contact Users

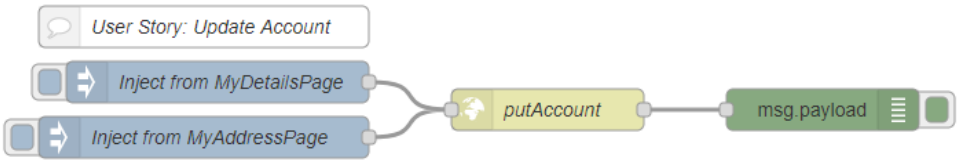



Ροή μέσω της οποίας ο χρήστης μπορεί να επικοινωνήσει με άλλους χρήστες. Η ροή εκκινείται με το πάτημα του κουμπιού επικοινωνίας της σελίδας προφίλ ή της σελίδας δημοσίευσης, μέσω του οποίου αποστέλλονται τα ήδη υπάρχοντα μηνύματα με τον εκάστοτε χρήστη στο endpoint "findChatByAccount" και στην συνέχεια στο endpoint "getMessages" και του συστήματος. Επίσης, αν η ροή εκκινείται με το πάτημα του κουμπιού επικοινωνίας, μέσω του οποίου αποστέλλονται τα ήδη υπάρχοντα μηνύματα με τον εκάστοτε χρήστη στο endpoint "getChat" και στην συνέχεια στο endpoint "getMessages" και του

συστήματος.

| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject from ProfilePage | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας προφίλ του χρήστη. |
| Inject from PostPage | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της αρχικής σελίδας και του post του χρήστη. |
| Inject from ContactPage | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας επικοινωνίας με άλλους χρήστες. |
| findChatByAccount | http-request | Κάνει κλήση προς την υπηρεσία findChatByAccount η οποία επιστρέφει την συνομιλία με τον χρήστη με το συγκεκριμένο ID. |
| getChat | http-request | Κάνει κλήση προς την υπηρεσία getChat η οποία επιστρέφει την συνομιλία με τον χρήστη μέσω της υπηρεσίας getChats στην οποία βρίσκονται όλες οι συνομιλίες. |
| selectChat | function | Συνάρτηση μέσω της οποίας επιλέγεται η εκάστοτε συνομιλία με βάση την είσοδο της συνάρτησης |
| msg.payload | debug | Τυπώνει στην κονσόλα το response που στέλνει η εκάστοτε http-request υπηρεσία, δηλαδή τα μηνύματα με τον εκάστοτε χρήστη. |

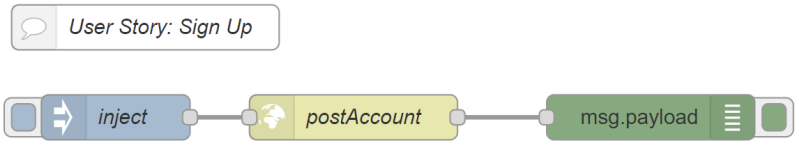
### 3.2.6 Ιστορία Χρήστη Update Account




Ροή μέσω της οποίας ο χρήστης μπορεί να επικαιροποιήσει τα προσωπικά του στοιχεία. Η ροή εκκινείται με το πάτημα του κουμπιού λογαριασμού ή τοποθεσίας, μέσω του οποίου αποστέλλονται τα προσωπικά στοιχεία του χρήστη στο endpoint “putAccount” του συστήματος.

| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject from MyDetailsPage | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας MyDetailsPage. |
| Inject from MyAddressPage | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας MyAddressPage. |
| putAccount | http-request | Κάνει κλήση προς την υπηρεσία putAccount η οποία δίνει στον χρήστη την δυνατότητα να ενημερώσει τις πληροφορίες του. |
| msg.payload | debug | Τυπώνει στην κονσόλα το response του συστήματος στην κλήση του χρήστη, δηλαδή τις ενημερωμένες πληροφορίες του λογαριασμού. |

### 3.2.7 Ιστορία Χρήστη Signup




Η ροή αυτή αποτελεί μια αναπαράσταση της διαδικασίας εγγραφής ενός νέου χρήστη. Η ροη εκκινείται με το πάτημα του κουμπιού εγγραφής, που αποστέλλονται τα στοιχεία του νέου χρήστη στο endpoint"postAccount" του συστήματος. Αυτός εξάγει σαν αποτέλεσμα την επιτυχία εγγραφής του χρήστη.

| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας εγγραφής χρήστη. |
| postAccount | http-request | Κάνει κλήση προς την υπηρεσία postAccount, η οποία δημιουργεί ένα νέο λογαριασμό χρήση (Account) στη βάση δεδομένων. |
| msg.payload | debug | Τυπώνει στη κονσόλα την απάντηση του συστήματος στο συγκεκριμένο αίτημα: δηλαδή την επιτυχία της δημιουργίας λογαριασμού. |

### 3.2.8 Ιστορία Χρήστη Sign In


Η ροή αυτή αποτελεί μια αναπαράσταση της διαδικασίας σύνδεσης ενός εγγεγραμμένου χρήστη. Η ροη εκκινείται με το πάτημα του κουμπιού σύνδεσης, που αποστέλλονται τα στοιχεία σύνδεσης του χρήστη στο endpoint "loginAccount" του συστήματος. Αυτός επιστρέφει σαν αποτέλεσμα ένα μοναδικό Token σύνδεσης χρήστη για το σύστημα (JWT Token).

| Όνομα κόμβου | Τύπος κόμβου | Περιγραφή |
|---|---|---|
| Inject | Inject | Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής, μέσω της σελίδας σύνδεσης χρήστη. |
| loginAccount | http-request | Κάνει κλήση προς την υπηρεσία loginAccount, η οποία επιστρέφει ένα JWT Token σύνδεσης χρήστη στην εφαρμογή. |
| msg.payload | debug | Τυπώνει στη κονσόλα την απάντηση του συστήματος στο συγκεκριμένο αίτημα: δηλαδή το JWT Token σύνδεσης του χρήστη. |