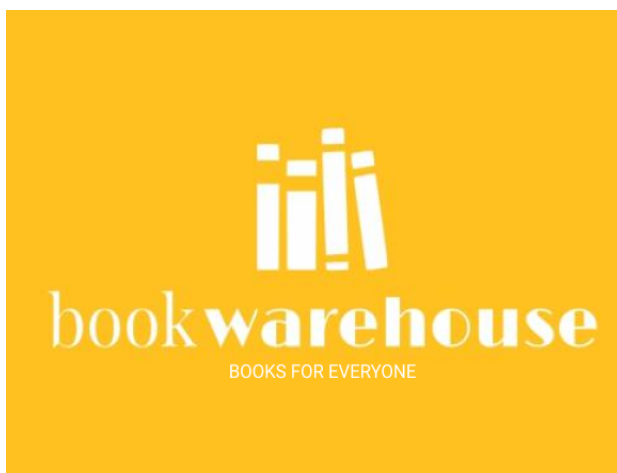




Book Warehouse

Books for everyone



Προδιαγραφές Λογισμικού και Υλοποίηση συστήματος

Del.2.1

Version 0.6 (draft)

Μωράτης Κωνσταντίνος, kmoratis@ece.auth.gr
Διαμάντης Γεώργιος-Μιχαήλ, gdiamanti@ece.auth.gr
Παρτσάνης Παναγιώτης, partsanis@ece.auth.gr
Αβραμίδης Παναγιώτης, pavramid@ece.auth.gr

31/5/2021



Ιστορικό Αλλαγών

Όνομα	Ημερομηνία	Αλλαγή	Έκδοση
Α. Συμεωνίδης	17/05/2007	Δημιουργία εγγράφου. Προσαρμογή των προτύπων του K. E. Wiegers ¹ και του M. Smialek's.	0.1
Α. Συμεωνίδης	29/3/2014	Μικρή αναθεώρηση – τροποποίηση ενότητων	0.1.3
Χ. Ζολώτας	10/4/2020	Μεγάλη αναθεώρηση – αφαίρεση ενότητων	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
Μωράτης Κωνσταντίνος	26	kmoratis@ece.auth.gr
Διαμάντης Γεώργιος-Μιχαήλ	26	gdiamanti@ece.auth.gr
Παρτσάνης Παναγιώτης	26	partsanis@ece.auth.gr
Αβραμίδης Παναγιώτης	26	pavramid@ece.auth.gr

¹ Copyright © 2002 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document. Original template is available at: <http://www.processimpact.com/>



Πίνακας Περιεχομένων

Πίνακας Περιεχομένων	3
Λίστα Σχημάτων	4
2. Αρχιτεκτονική Συστήματος	8
2.1 Αναγνώριση Πόρων (Resources) Συστήματος	9
2.2 Τεκμηρίωση REST διεπαφής	10
2.2.1 Πόρος User	10
2.2.1.1 Μοντέλο δεδομένων user	10
2.2.2 Πόρος Book	12
2.2.2.1 Μοντέλο δεδομένων Book	12
2.2.3 Πόρος Rating	14
2.2.3.1 Μοντέλο δεδομένων Rating	14
2.2.3.2 Μοντέλο δεδομένων AllRatings	14
2.2.4 Πόρος Recommended	18
2.2.4.1 Μοντέλο δεδομένων Recommended	18
2.2.5 Πόρος Wishlist	19
2.2.5.1 Μοντέλο δεδομένων Wishlist	19
2.2.6 Πόρος PersonalInfo	21
2.2.6.1 Μοντέλο δεδομένων PersonalInfo	21
2.2.7 Αλγοριθμικός πόρος SearchByTitle	23
2.2.7.1 Μοντέλο δεδομένων SearchResults	23
2.2.8 Αλγοριθμικός πόρος SearchByCategory	25
3. Υλοποίηση Συστήματος με Node-RED	26
3.1 Αντιστοίχιση των REST Υπηρεσιών σε Ροές NodeRed	26
3.1.1 Ροές πόρου User	26
3.1.2 Ροή πόρου Book	27
3.1.3 Ροές πόρου Rating	27
3.1.4 Ροή πόρου Recommended	28
3.1.5 Ροή πόρου Wishlist	29
3.1.6 Ροές πόρου PersonalInfo	29
3.1.7 Ροή αλγοριθμικού πόρου SearchByTitle	30
3.1.8 Ροή αλγοριθμικού πόρου SearchByCategory	31
3.2 Υλοποίηση Ιστοριών χρήστη	31
3.2.1 Ιστορία Χρήστη: User create a new account	31
3.2.2 Ιστορία Χρήστη: User login to account	32
3.2.3 Ιστορία Χρήστη: User select a book	32
3.2.4 Ιστορία Χρήστη: User rate a book	33



3.2.5 Ιστορία Χρήστη: User see book's ratings	33
3.2.6 Ιστορία Χρήστη: User see a book's rating	34
3.2.7 Ιστορία Χρήστη: User get recommended	34
3.2.8 Ιστορία Χρήστη: User edit wishlist	35
3.2.9 Ιστορία Χρήστη: User add personal information	35
3.2.10 Ιστορία Χρήστη: User edit personal information	36
3.2.11 Ιστορία Χρήστη: User search book by title	36
3.2.12 Ιστορία Χρήστη: User search book by category	37
Παράρτημα Ι – Ανοιχτά Θέματα	38

1. Πρότυπα Σχεδιασμού που υιοθετήθηκαν

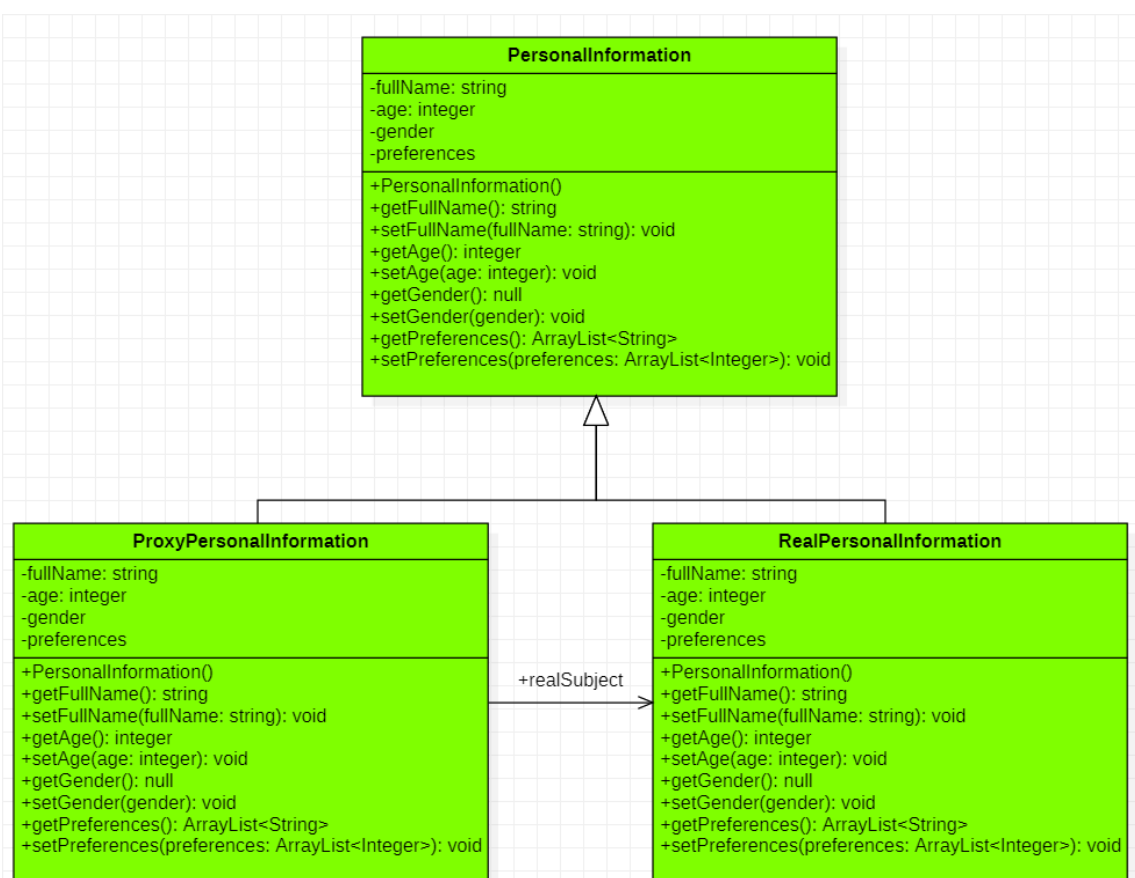
Στο σημείο αυτό ορίζουμε μία επιπλέον Μη Λειτουργική Απαίτηση την οποία πρέπει να ικανοποιεί η εφαρμογή μας.

<ΜΛΑ- 6> (Απαιτήσεις διασύνδεσης)

Το σύστημα πρέπει να μπορεί να συνδέεται σε βάσεις δεδομένων MySQL και MsSQL για να εγγράφει, να αντλεί και να αλλάζει δεδομένα για τα βιβλία και τους χρήστες του.

1.1 Πρότυπο Proxy

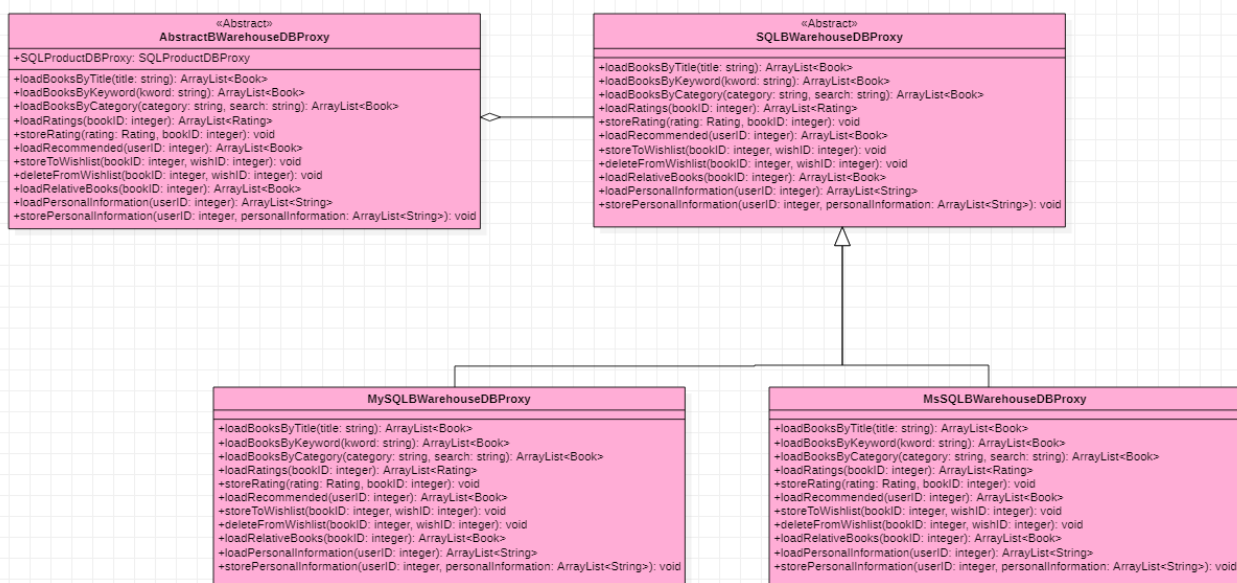
Το πρότυπο Proxy αποτελεί ένα δομικό πρότυπο. Στα πλαίσια της εφαρμογής μας το χρησιμοποιούμε ως Protection Proxy για να επιτύχουμε έλεγχο πρόσβασης στα αντικείμενα της κλάσης Personal Information. Με τον τρόπο αυτόν μπορούμε να προστατεύσουμε τα προσωπικά δεδομένα των χρηστών του συστήματος μας και επομένως ικανοποιείται η ΜΛΑ-2 που έχει οριστεί στο έγγραφο απαιτήσεων χρηστών. Ειδικότερα, θέλουμε τα προσωπικά δεδομένα των χρηστών να μπορούν να προβληθούν μόνο από τους διαχειριστές του συστήματος καθώς και από τους ίδιους τους χρήστες.



Σχήμα 1: Εφαρμογή προτύπου Proxy.

1.2 Πρότυπο Bridge

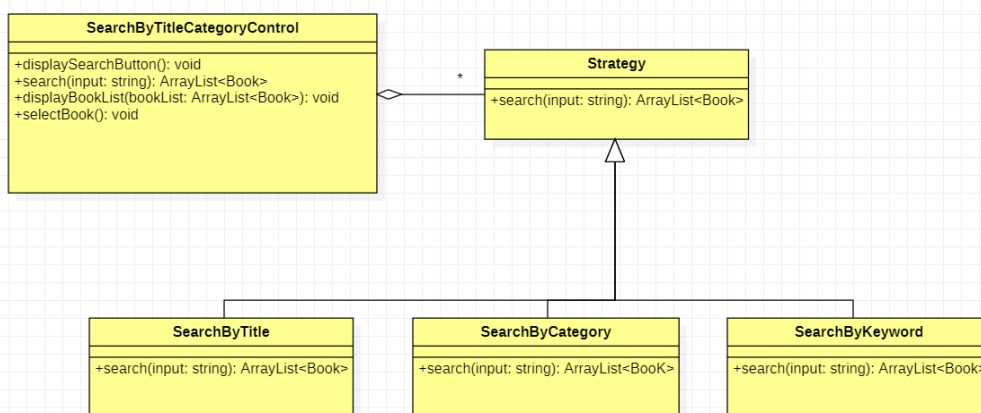
Το πρότυπο Bridge είναι ένα δομικό πρότυπο. Στο πλαίσιο της εφαρμογής μας το χρησιμοποιούμε για να επιτύχουμε ανεξαρτησία μιας συγκεκριμένης αφαίρεσης από την υλοποίησή της. Πιο συγκεκριμένα, χρησιμοποιώντας το συγκεκριμένο πρότυπο καταφέραμε να διαχωρίσουμε την αρχική διεπαφή BWarehouseDBProxy από την υλοποίησή της ώστε να μπορούν να υπάρχουν και οι δύο ως ανεξάρτητες κλάσεις. Με τον τρόπο αυτόν και χρησιμοποιώντας κληρονομικότητα μπορούμε να έχουμε δύο διαφορετικές υλοποιήσεις της διεπαφής, μία για MySQL και μία για MsSQL βάσεις. Επομένως, ικανοποιείται η ΜΛΑ-6 του συστήματος μας. Ένα ακόμη πλεονέκτημα που προκύπτει από τη χρήση του συγκεκριμένου προτύπου, είναι ότι η τροποποίηση του συστήματος ώστε να μπορεί να συνδέεται και σε άλλου τύπου βάσεις δεδομένων είναι εύκολη και δεν απαιτεί επανασχεδιασμό του υπάρχοντος συστήματος.



Σχήμα 2: Εφαρμογή προτύπου Bridge.

1.3 Πρότυπο Strategy

Το πρότυπο Strategy είναι ένα πρότυπο Συμπεριφοράς. Με την χρήση του συγκεκριμένου προτύπου, καλύπτεται η σχεδιαστική ανάγκη της διαχείρισης των τριών διαφορετικών αλγορίθμων που υπάρχουν στην εφαρμογή για αναζήτηση βιβλίων. Έτσι έχουμε υλοποιήσει τρεις κλάσεις που κληρονομούν την κλάση Strategy, μία για αναζήτηση βιβλίων με τον τίτλο, μία για αναζήτηση ανά κατηγορία και μία για αναζήτηση με μία λέξη- κλειδί. Επιπλέον, με τον τρόπο αυτόν η σχεδίαση του συστήματός μας γίνεται τροποποιήσιμη, καθώς σε περίπτωση που χρειαστεί να προσθέσουμε έναν ακόμη αλγόριθμο για εξειδικευμένη αναζήτηση, αυτό μπορεί να γίνει εύκολα και χωρίς να επηρεάζει το ήδη υπάρχον σύστημα.



Σχήμα 3: Εφαρμογή προτύπου Strategy.



2. Αρχιτεκτονική Συστήματος

Για λόγους πληρότητας της εργασίας μας, ορίζουμε εδώ δύο ακόμη Λειτουργικές Απαιτήσεις που σχετίζονται με τη δημιουργία λογαριασμού χρήστη και τη σύνδεση σε αυτόν. Πιο συγκεκριμένα, ορίζουμε τις Λειτουργικές Απαιτήσεις:

<ΛΑ- 11>

Ο χρήστης πρέπει να μπορεί να δημιουργεί ένα λογαριασμό, εισάγοντας ένα όνομα χρήστη και έναν κωδικό πρόσβασης.

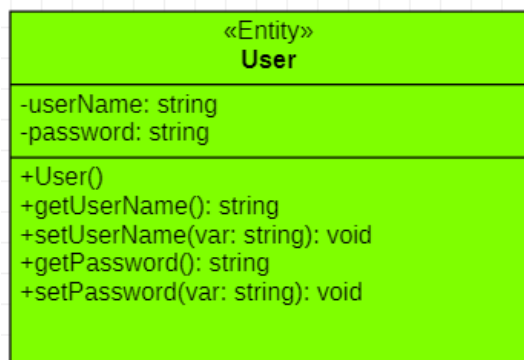
<ΛΑ- 12>

Ο χρήστης πρέπει να μπορεί να συνδέεται στο λογαριασμό του, χρησιμοποιώντας το όνομα χρήστη και τον κωδικό πρόσβασής του.

Επομένως, οι Λειτουργικές Απαιτήσεις 1-10 που έχουμε ορίσει στο Έγγραφο Απαιτήσεων Χρηστών, θα πρέπει να τροποποιηθούν και όπου αναφέρεται ο όρος <<χρήστης>>, να αντικατασταθεί από τον όρο <<συνδεδεμένος χρήστης>>.

Επιπλέον, για να προδιαγράψουμε το REST API της εφαρμογής μας, πρέπει να προσθέσουμε μία επιπλέον κλάση σε αυτές που έχουμε ορίσει στο πρώτο παραδοτέο, για την υλοποίηση της οντότητας του χρήστη (User). Η κλάση αυτή περιέχει τα χαρακτηριστικά username και password που σχετίζονται με την σύνδεση του χρήστη στην εφαρμογή.

Entity Book



Σχήμα 4: Κλάση user του συστήματος.

2.1 Αναγνώριση Πόρων (Resources) Συστήματος

Κλάση BEC	Πόρος REST	Endpoints (HTTP Verbs)
User	/user	POST
User	/user/login	PUT
Rating, (Book, User)	/user/{username}/book/{bookID}/rating	POST, GET
Book, (User)	/user/{username}/book/{bookID}	GET



Recommended, (User)	/user/{username}/recommended	GET
Rating, (Book, User)	/user/{username}/book/{bookID}/rating/{ratingID}	GET
Wishlist, (Book, User)	/user/{username}/wishlist	PUT
PersonalInfo, (User)	/user/{username}/personalInfo	POST
PersonalInfo, (User)	/user/{username}/personalInfo/{infoID}	PUT
SearchByTitleCategoryContol, (User)	/user/{username}/searchByTitle	GET
SearchByTitleCategoryControl, (User)	/user/{username}/searchByCategory	GET

2.2 Τεκμηρίωση REST διεπαφής

Για το specification του API της εφαρμογής μας χρησιμοποιήθηκε το εργαλείο Swaggerhub, καθώς και ο Swagger Editor. Παραθέτουμε στη συνέχεια τους ενεργούς υπερσυνδέσμους για τα ζητούμενα αρχεία:

- Swagger API
- Dropbox json file
- Dropbox server

2.2.1 Πόρος User

2.2.1.1 Μοντέλο δεδομένων user

```
User {  
  username      string  
  password      string  
}
```

Σχήμα 5: Μοντέλο δεδομένων User.

2.2.1.2 Endpoint POST πόρου User

POST /user Create a new user account

FR11 - a user must be able to create a new account by providing a username and a password

Parameters Cancel

Name	Description
body * required	User model
object	
(body)	<div>Edit Value Model</div> <pre>{ "username": "exampleUser", "password": "examplePass" }</pre>



Responses	
Code	Description
200	successful operation.
400	bad input parameter

Σχήμα 6: Επεξήγηση και παράθεση των παραμέτρων του πόρου.
Σχήμα 7: Δυνατές αποκρίσεις του server.

Curl

```
curl -X 'POST' \
  'https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "username": "string",
    "password": "string"
  }'
```

Request URL

https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user

Server response

Code	Details
200	<p>Response headers</p> <pre>access-control-allow-credentials: true access-control-allow-headers: X-Requested-With,Content-Type,Accept,Origin access-control-allow-methods: *</pre>

Σχήμα 8: Δοκιμή του endpoint.

2.2.1.3 Endpoint PUT πόρου User

PUT /user/login Logs user into the system

FR12- a user must be able to log to his/ her account, by providing his/ her username and password

Parameters Cancel

Name	Description
username * required	the username for login
string (query)	<input type="text" value="exampleUser"/>
password * required	the password for login in clear text
string (query)	<input type="text" value="examplePass"/>

Execute Clear



Responses

Code	Description	
200	successful operation	
Example Value Model		
<pre>{ "token": "string" }</pre>		
Headers:		
Name	Description	Type
X-Rate-Limit	calls per hour allowed by the user	integer
X-Expires-After	date on UTC then token expires	string
400	invalid username/ password supplied	

Σχήμα 9: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 10: Δυνατές αποκρίσεις του server.

Responses Response content type application/json

Curl

```
curl -X 'PUT' \
'https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/login?username=exampleUser&password=examplePass' \
-H 'accept: application/json'
```

Request URL

```
https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/login?username=exampleUser&password=examplePass
```

Server response

Code	Details
200	<div>Response body</div> <pre>{ "token": "string" }</pre> <div>Response headers</div> <pre>access-control-allow-credentials: true access-control-allow-headers: X-Requested-With, Content-Type, Accept, Origin access-control-allow-methods: * access-control-allow-origin: *</pre>

Σχήμα 11: Δοκιμή του endpoint.

2.2.2 Πόρος Book

2.2.2.1 Μοντέλο δεδομένων Book



```
Book {  
  bookID      integer  
  title       string  
  type        string  
  author      string  
  genre       string  
  publisher   string  
  publicationDate string  
  relativeBookIds [integer]  
}
```

Σχήμα 12: Μοντέλο δεδομένων Book.

2.2.2.2 Endpoint GET πόρου Book, για συγκεκριμένο username και book id.

GET /user/{username}/book/{bookID} Returns a book's attributes

FR3- a user must be able to select a book, FR10- a user must be able to see a book's relative books

Parameters

Name	Description
username * required	username for user identification
string (path)	myUser
bookID * required	book id for book identification
integer (path)	1245

Execute

Responses

Code	Description
200	successful operation
400	bad input parameters
404	not found

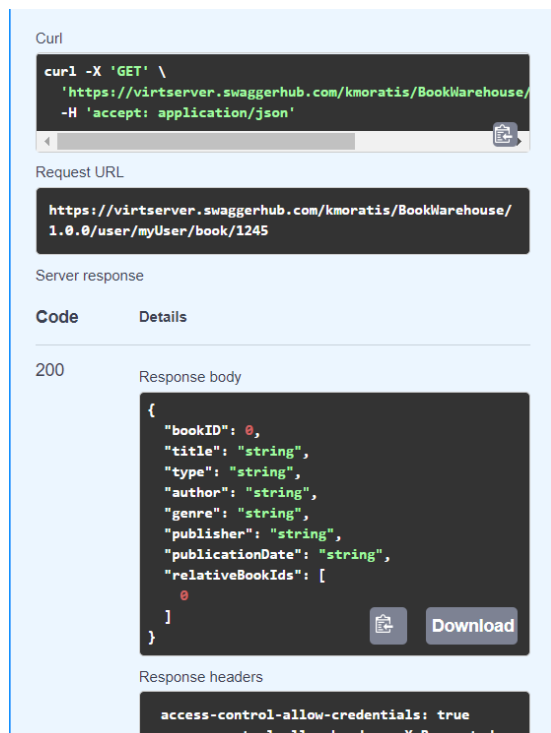
Example Value

```
{  
  "bookID": 0,  
  "title": "string",  
  "type": "string",  
  "author": "string",  
  "genre": "string",  
  "publisher": "string",  
  "publicationDate": "string",  
  "relativeBookIds": [  
    0  
  ]  
}
```

Σχήμα 13: Επεξήγηση και παράθεση των παραμέτρων του πόρου.



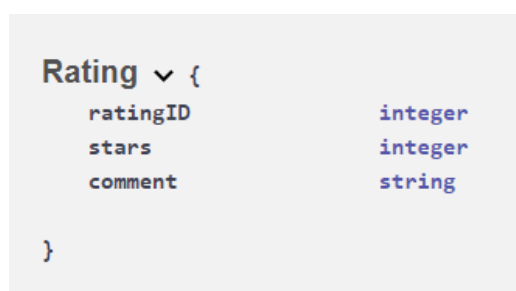
Σχήμα 14: Δυνατές αποκρίσεις του server.



Σχήμα 15: Δοκιμή του endpoint.

2.2.3 Πόρος Rating

2.2.3.1 Μοντέλο δεδομένων Rating



Σχήμα 16: Μοντέλο δεδομένων Rating.

2.2.3.2 Μοντέλο δεδομένων AllRatings



```
AllRatings ▾ [AllRatings ▾ {  
  ratingID      integer  
  stars         integer  
  comment       string  
}]
```

Σχήμα 17: Μοντέλο δεδομένων AllRatings.

2.2.3.3 Endpoint POST πόρου Rating, για συγκεκριμένο username και book id.



POST /user/{username}/book
/{bookID}/rating

Creates a rating
for a book

FR2- a logged in user must be able to rate a book

Parameters

Name	Description
username * required	username for user identification
string (path)	<input type="text" value="myUser"/>
bookID * required	book id for book identification
integer (path)	<input type="text" value="1256"/>
rating * required	Rating model
object (body)	<div>Edit Value Model</div> <pre>{ "ratingID": 23, "stars": 4, "comment": "really good book" }</pre>

Responses

Code	Description
400	bad input parameter
default	successful operation

Σχήμα 18: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 19: Δυνατές αποκρίσεις του server.



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

Curl

```
curl -X 'POST' \
  'https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/%22myUser%22/book/1256/rating' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "ratingID": 23,
    "stars": 4,
    "comment": "really good book"
  }'
```

Request URL

```
https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/%22myUser%22/book/1256/rating
```

Server response

Code	Details
400	Error: Response headers <pre>access-control-allow-credentials: true access-control-allow-headers: X-Requested-With,Content-Type,Accept,Origin access-control-allow-methods: * access-control-allow-origin: * cache-control: no-cache content-encoding: gzip</pre>

Σχήμα 20: Δοκιμή του endpoint.

2.2.3.4 Endpoint GET πόρου Rating, για συγκεκριμένο username και book id.

GET /user/{username}/book/{bookID}/rating Retrieve all ratings for a specific book

FR7- a user must be able to view a book's ratings

Parameters **Cancel**

Name	Description
username * required	username for user identification
string (path)	<input type="text" value="myUser"/>
bookID * required	book id for identification
integer (path)	<input type="text" value="1256"/>

Execute



Responses	
Response content type application/json	
Code	Description
200	successful operation Example Value Model <pre>[{ "ratingID": 0, "stars": 0, "comment": "string" }]</pre>
400	bad input parameters
404	not found

Σχήμα 21: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 22: Δυνατές αποκρίσεις του server.

Curl	
<pre>curl -X 'GET' \ 'https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myU -H 'accept: application/json'</pre>	
Request URL	
<pre>https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myU ser/book/1256/rating</pre>	
Server response	
Code	Details
200	Response body <pre>[{ "ratingID": 0, "stars": 0, "comment": "string" }]</pre> Response headers <pre>access-control-allow-credentials: true access-control-allow-headers: X-Requested-With,Content- Type,Accept,Origin access-control-allow-methods: *</pre>

Σχήμα 23: Δοκιμή του endpoint.

2.2.3.5 Endpoint GET πόρου Rating, για συγκεκριμένο username, συγκεκριμένο book id και rating id.



GET

/user/{username}/book
/{bookID}/rating/{ratingID}

Retrieve a rating
for a specific
book

🔒 ↩

FR7- a user must be able to view a book's ratings

Parameters

Cancel

Name	Description
username * required	username for user identification
string (path)	<input type="text" value="myUser"/>
bookID * required	book id for identification
integer (path)	<input type="text" value="1256"/>
ratingID * required	rating id for identification
integer (path)	<input type="text" value="2"/>

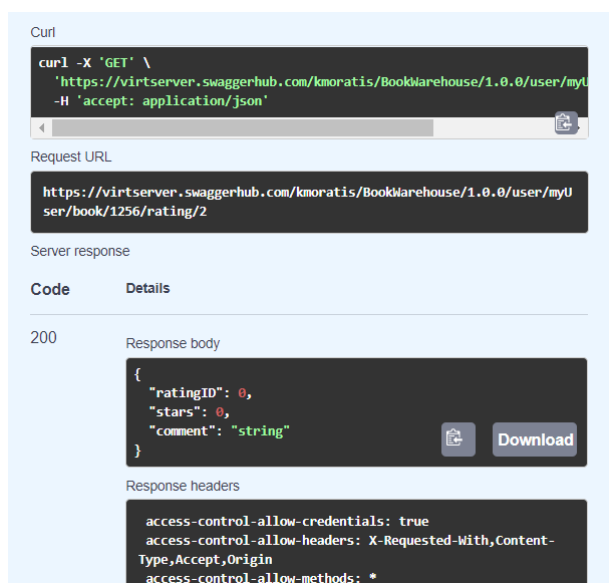
Execute

Responses

Response content type application/json

Code	Description
200	successful operation <div>Example Value Model</div> <pre>{ "ratingID": 0, "stars": 0, "comment": "string" }</pre>
400	bad input parameters
404	not found

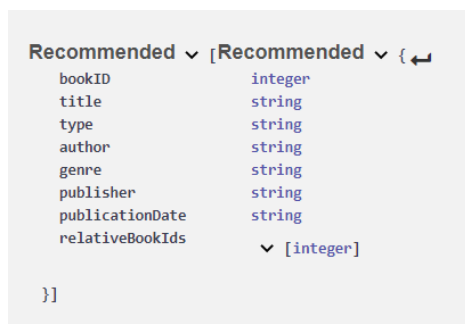
Σχήμα 24: Επεξήγηση και παράθεση των παραμέτρων του πόρου.
Σχήμα 25: Δυνατές αποκρίσεις του server.



Σχήμα 26: Δοκιμή του endpoint.

2.2.4 Πόρος Recommended

2.2.4.1 Μοντέλο δεδομένων Recommended



Σχήμα 27: Μοντέλο δεδομένων Recommended.

2.2.4.2 Endpoint GET πόρου Recommended, για συγκεκριμένο username.



GET

/user/{username}/recommended

Retrieve user's recommended books

FR4- a user must be able to see his/ her recommended books

Parameters

Cancel

Name	Description
username <small>* required</small>	username for user identification
string (path)	<input type="text" value="myUser"/>

Execute

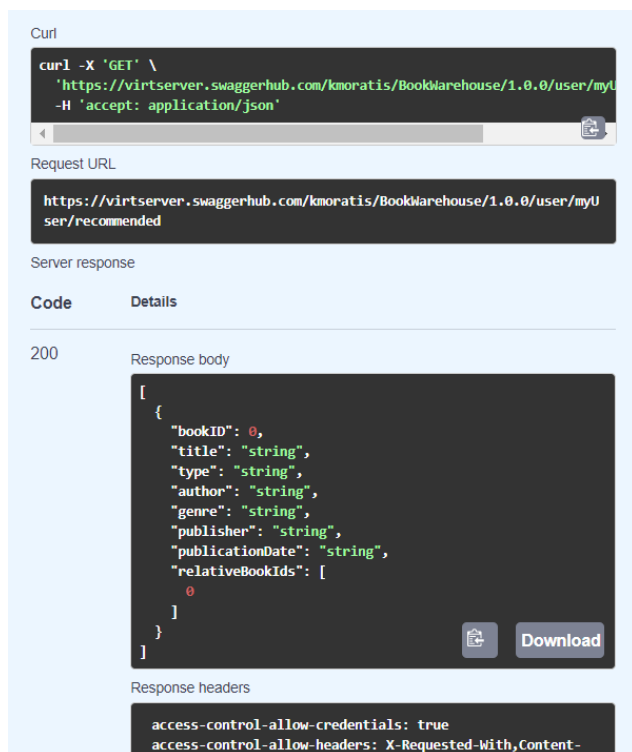
Responses

Response content type application/json

Code	Description
200	successful operation <div>Example Value Model</div> <pre>[{ "bookID": 0, "title": "string", "type": "string", "author": "string", "genre": "string", "publisher": "string", "publicationDate": "string", "relativeBookIds": [0] }]</pre>
400	bad input parameters
404	not found

Σχήμα 28: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 29: Δυνατές αποκρίσεις του server.



Σχήμα 30: Δοκιμή του Endpoint.

2.2.5 Πόρος Wishlist

2.2.5.1 Μοντέλο δεδομένων Wishlist



Σχήμα 31: Μοντέλο δεδομένων Wishlist.



2.2.5.2 Endpoint PUT πόρου Wishlist, για συγκεκριμένο username.

PUT

/user/{username}/wishlist

Edit wishlist contents

🔒 ↩

FR9- a user must be able to edit his/ her wishlist

Parameters

Cancel

Name	Description
username <small>* required</small>	username for user identification
string (path)	<input type="text" value="myUser"/>
wishlist <small>* required</small>	Wishlist model
object (body)	<div>Edit Value Model</div> <pre>{ "wishlistID": 0, "wishlistBooks": [{ "bookID": 0, "title": "string", "type": "string", "author": "string", "genre": "string", "publisher": "string", "publicationDate": "string", "relativeBookIds": [0] }]}</pre>

Responses

Code	Description
200	successful operation
400	bad input parameters

Σχήμα 32: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 33: Δυνατές αποκρίσεις του server.

Curl

```
curl -X 'PUT' \
  'https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myUser/wishlist' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "wishlistID": 0,
    "wishlistBooks": [
      {
        "bookID": 0,
        "title": "string",
        "type": "string",
        "author": "string",
        "genre": "string",
        "publisher": "string",
        "publicationDate": "string",
        "relativeBookIds": [
          0
        ]
      }
    ]
  }'
```

Request URL

https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myUser/wishlist

Server response

Code	Details
200	<div>Response headers</div> <div>access-control-allow-credentials: true access-control-allow-headers: X-Requested-With, Content-Type, Accept, Origin</div>

Σχήμα 34: Δοκιμή του endpoint.



2.2.6 Πόρος PersonalInfo

2.2.6.1 Μοντέλο δεδομένων PersonalInfo

```
PersonalInfo {  
  fullName: string  
  age: integer  
  gender: string  
  preferences: > [...]  
}
```

Σχήμα 35: Μοντέλο δεδομένων PersonalInfo.

2.2.6.2 Endpoint POST πόρου PersonalInfo, για συγκεκριμένο username.

POST

/user/{username}/personalInfo

Create personal information entity

🔒 ↩

FR5- a user must be able to add his/ her personal information

Parameters

Cancel

Name	Description
username * required	username for user identification
string (path)	<input type="text" value="myUser"/>
personalInfo * required	Personal info model
object (body)	<div>Edit Value Model</div> <pre>{ "fullName": "string", "age": 0, "gender": "string", "preferences": ["string"] }</pre>

Responses

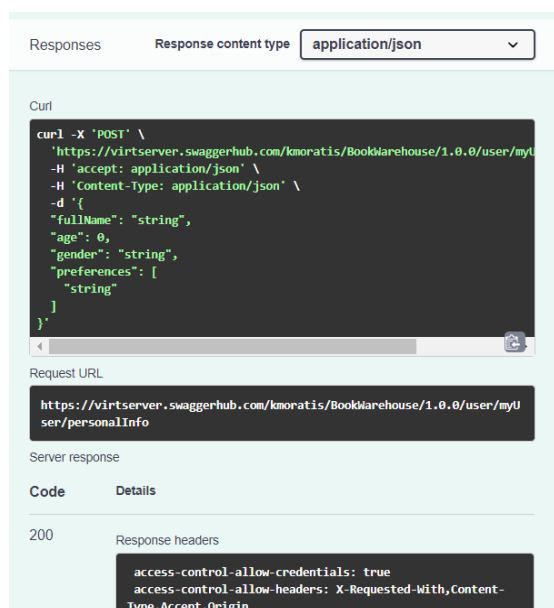
Response content type

application/json

Code	Description
200	successful operation
400	bad input parameters

Σχήμα 36: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 37: Δυνατές αποκρίσεις του server.



Σχήμα 38: Δοκιμή του endpoint.

2.2.6.3 Endpoint PUT πόρου PersonalInfo, για συγκεκριμένο username και info id.

PUT /user/{username}/personalInfo Edit personal information
/{infoID}

FR6- a user must be able to edit his/ her personal information

Parameters Cancel

Name	Description
username * required string (path)	username for user identification
infoID * required integer (path)	personal info id for identification
personalInfo * required object (body)	Personal info model Edit Value Model

```
{
  "fullName": "string",
  "age": 0,
  "gender": "string",
  "preferences": [
    "string"
  ]
}
```



Responses		Response content type	application/json
Code	Description		
200	successful operation		
400	bad input parameters		

Σχήμα 39: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 40: Δυνατές αποκρίσεις του server.

Responses

Response content type

application/json

Curl

```
curl -X 'PUT' \
  'https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myU
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "fullName": "string",
    "age": 0,
    "gender": "string",
    "preferences": [
      "string"
    ]
  }'
```

Request URL

```
https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myU
ser/personalInfo/1289
```

Server response

Code	Details
200	<div>Response headers</div> <pre>access-control-allow-credentials: true access-control-allow-headers: X-Requested-With,Content- Type,Accept,Origin access-control-allow-methods: * access-control-allow-origin: * cache-control: no-cache content-encoding: gzip</pre>

Σχήμα 41: Δοκιμή του endpoint.



2.2.7 Αλγοριθμικός πόρος SearchByTitle

2.2.7.1 Μοντέλο δεδομένων SearchResults

```
SearchResults ▾ [SearchResults ▾ {  
  bookID      integer  
  title       string  
  type        string  
  author      string  
  genre       string  
  publisher   string  
  publicationDate string  
  relativeBookIds > [...]  
}]
```

Σχήμα 42: Μοντέλο δεδομένων SearchResults.

2.2.7.2 Endpoint GET πόρου SearchByTitle, για συγκεκριμένο username.

GET

/user/{username}
/searchByTitle

Search by title
operation

FR1- a user must be able to search for a book by title

Parameters

Cancel

Name	Description
username * required	username for user identification
string (path)	<input type="text" value="myUser"/>
title * required	book's title for search algorithm
string (query)	<input type="text" value="Origin"/>

Execute

Clear



Responses	
Code	Description
200	successful operation Example Value Model <pre>[{ "bookID": 0, "title": "string", "type": "string", "author": "string", "genre": "string", "publisher": "string", "publicationDate": "string", "relativeBookIds": [0] }]</pre>
400	bad input parameters
404	not found

Σχήμα 43: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 44: Δυνατές αποκρίσεις του server.

Curl

```
curl -X 'GET' \
  'https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myUser/searchByTitle?title=origin' \
  -H 'accept: application/json'
```

Request URL

```
https://virtserver.swaggerhub.com/kmoratis/BookWarehouse/1.0.0/user/myUser/searchByTitle?title=origin
```

Server response

Code	Details
200	Response body <pre>[{ "bookID": 0, "title": "string", "type": "string", "author": "string", "genre": "string", "publisher": "string", "publicationDate": "string", "relativeBookIds": [0] }]</pre> Response headers <pre>access-control-allow-credentials: true</pre>

Σχήμα 45: Δοκιμή του endpoint.



2.2.8 Αλγοριθμικός πόρος SearchByCategory

2.2.8.1 Endpoint GET πόρου SearchByCategory, για συγκεκριμένο username.

GET

/user/{username}
/searchByCategory

Search by
category
operation

🔒 ↩

FR8- a user must be able to search for books by category

Parameters

Cancel

Name	Description
username * required	username for user identification
string (path)	<input type="text" value="myUser"/>
category * required	book category for search algorithm
string (query)	<input type="text" value="Science"/>

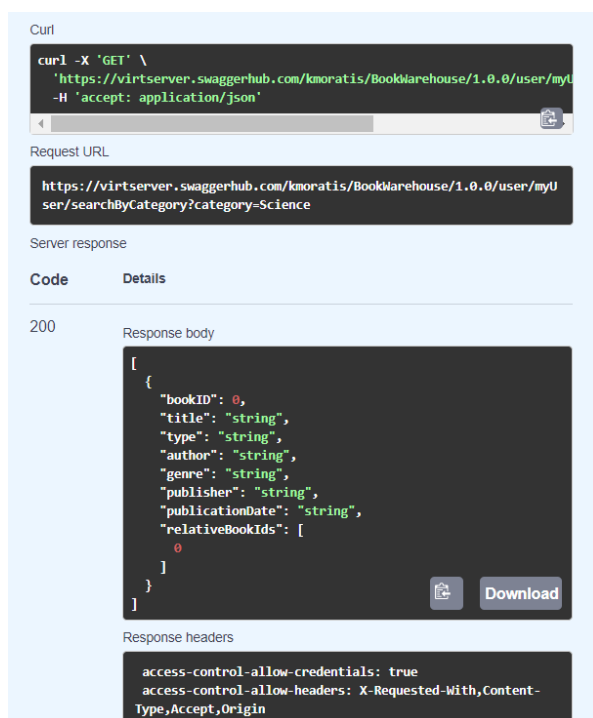
ExecuteClear

Responses

Code	Description
200	successful operation <div>Example Value Model</div> <pre>[{ "bookID": 0, "title": "string", "type": "string", "author": "string", "genre": "string", "publisher": "string", "publicationDate": "string", "relativeBookIds": [0] }]</pre>
400	bad input parameters
404	not found

Σχήμα 46: Επεξήγηση και παράθεση των παραμέτρων του πόρου.

Σχήμα 47: Δυνατές αποκρίσεις του server.



Σχήμα 48: Δοκιμή του endpoint.

3. Υλοποίηση Συστήματος με Node-RED

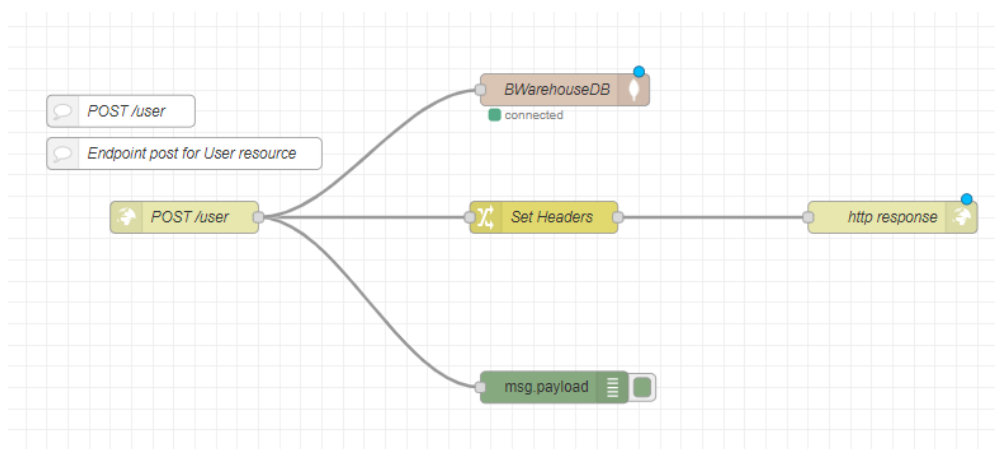
- [Σύνδεσμος](#) για το αρχείο τύπου zip που περιέχει τα flows που υλοποιήθηκαν με node-red.
- [Σύνδεσμος](#) για το αρχείο τύπου zip που περιέχει την MongoDB βάση δεδομένων που χρησιμοποιήθηκε στις δοκιμές τις εφαρμογής.

3.1 Αντιστοίχιση των REST Υπηρεσιών σε Ροές NodeRed

3.1.1 Ροές πόρου User

Ροή του endpoint POST /user:

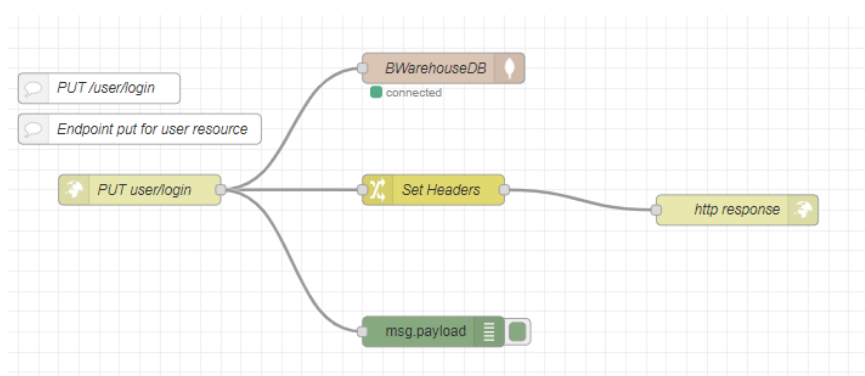
Η συγκεκριμένη ροή υλοποιεί την υπηρεσία μέσω της οποίας ένας χρήστης μπορεί να δημιουργεί έναν λογαριασμό στην βάση δεδομένων.



Σχήμα 49: Ροή endpoint POST /user.

Ροή του endpoint PUT /user/login:

Με τη συγκεκριμένη ροή υλοποιείται η υπηρεσία με την οποία ένας χρήστης μπορεί να συνδεθεί στο λογαριασμό του, αλλάζοντας την κατάσταση του λογαριασμού.

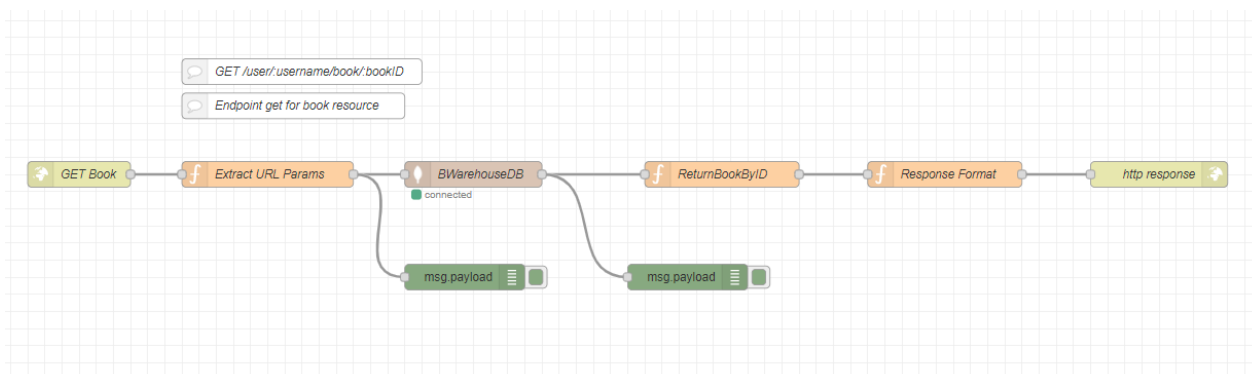


Σχήμα 50: Ροή endpoint PUT /user/login.

3.1.2 Ροή πόρου Book

Ροή του endpoint GET /user/{username}/book/{bookID}:

Η υπηρεσία αυτή είναι υπεύθυνη για την επιστροφή των χαρακτηριστικών ενός συγκεκριμένου βιβλίου, δεδομένου το id του βιβλίου, το οποίο δίνεται σαν URL (path) parameter. Η συνάρτηση Response Format μετασχηματίζει την απάντηση του server στο επιθυμητό από το χρήστη format.

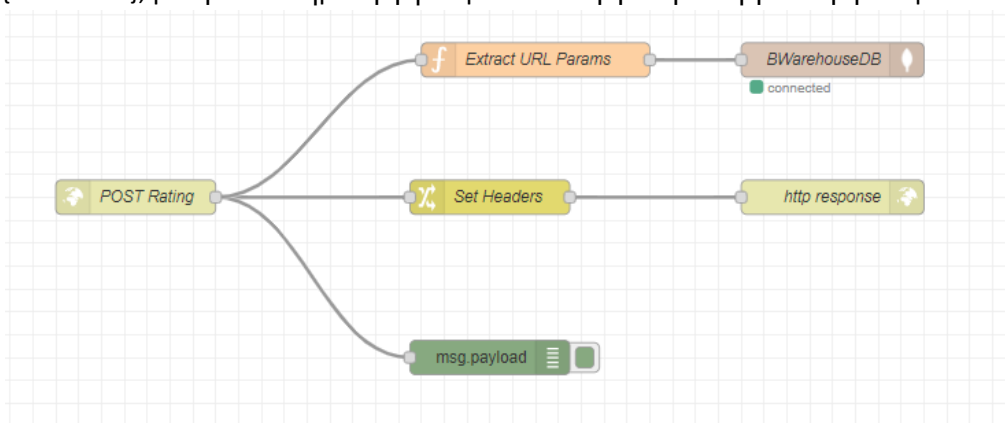


Σχήμα 51: Ροή endpoint GET `/user/{username}/book/{bookID}/`.

3.1.3 Ροές πόρου Rating

Ροή του endpoint POST `/user/{username}/book/{bookID}/rating`:

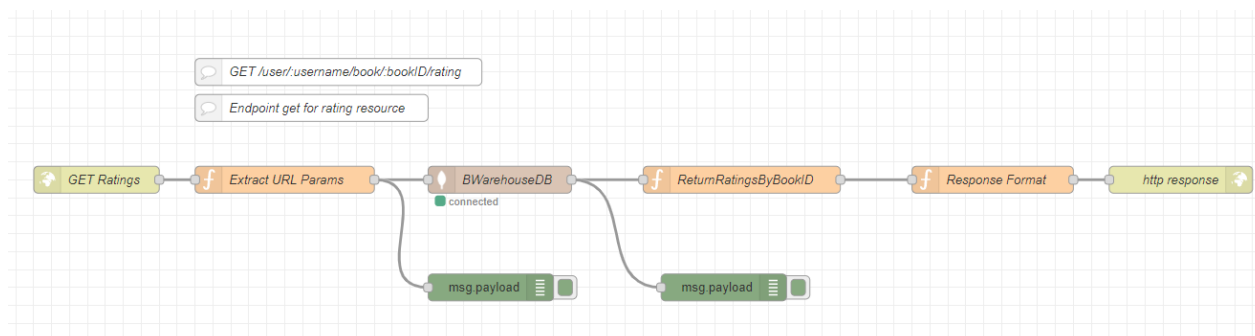
Η συγκεκριμένη ροή υλοποιεί την υπηρεσία με την οποία ο χρήστης με όνομα χρήστη `{username}`, μπορεί να δημιουργήσει μία καινούργια κριτική για το βιβλίο με book id το `{bookID}`.



Σχήμα 52: Ροή endpoint POST `/user/{username}/book/{bookID}/rating`.

Ροή του endpoint GET `/user/{username}/book/{bookID}/rating`:

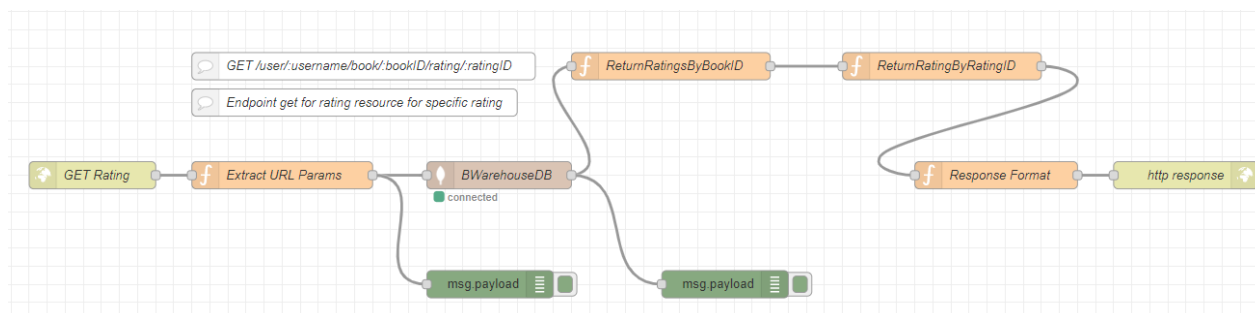
Η ροή που υλοποιεί την υπηρεσία με την οποία ο χρήστης μπορεί να παίρνει όλες τις κριτικές για ένα συγκεκριμένο βιβλίο.



Σχήμα 53: Ποή endpoint GET /user/{username}/book/{bookID}/rating.

Ποή του endpoint GET /user/{username}/book/{bookID}/rating/{ratingID}:

Ποή της υπηρεσίας που επιστρέφει στο χρήστη μία συγκεκριμένη κριτική του βιβλίου με id το {bookID}. Τα rating και Book ids δίνονται από τον χρήστη και αποστέλλονται στην υπηρεσία μέσω URL parameters.

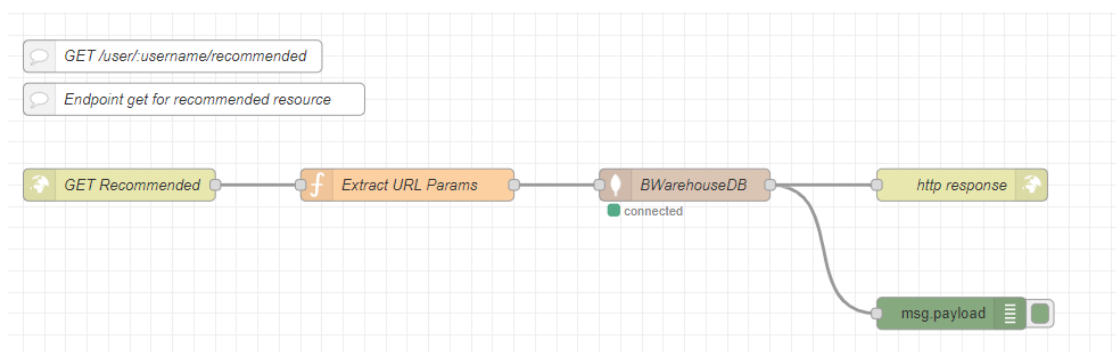


Σχήμα 54: Ποή endpoint GET /user/{username}/book/{bookID}/rating/:ratingID.

3.1.4 Ποή πόρου Recommended

Ποή του endpoint GET /user/{username}/recommended:

Η υπηρεσία αυτή είναι υπεύθυνη για την επιστροφή των προτεινόμενων βιβλίων για ένα συγκεκριμένο χρήστη, δεδομένο το username του, το οποίο δίνεται σαν URL (path) parameter.



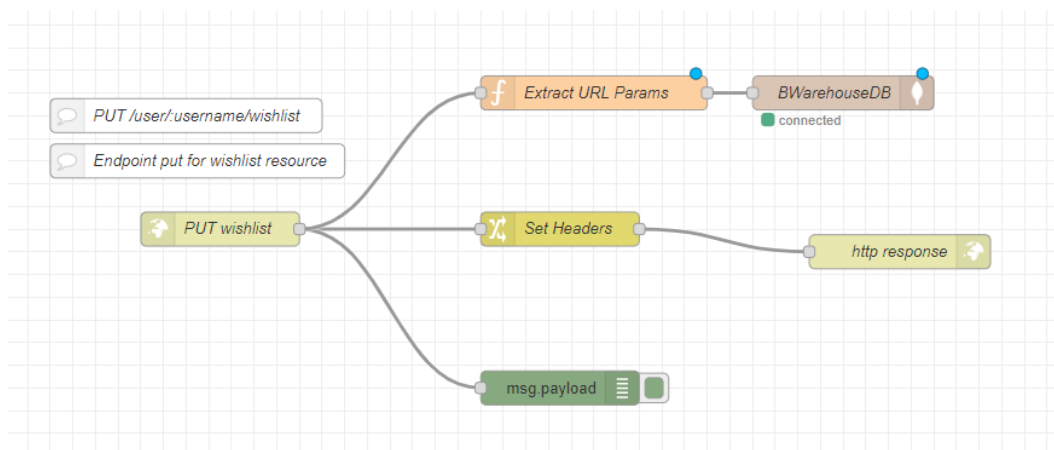
Σχήμα 55: Ποή endpoint GET /user/{username}/recommended.



3.1.5 Ροή πόρου Wishlist

Ροή του endpoint PUT /user/{username}/wishlist:

Η συγκεκριμένη ροή υλοποιεί την υπηρεσία μέσω της οποίας ο χρήστης μπορεί να αλλάξει τα περιεχόμενα απο το wishlist του.

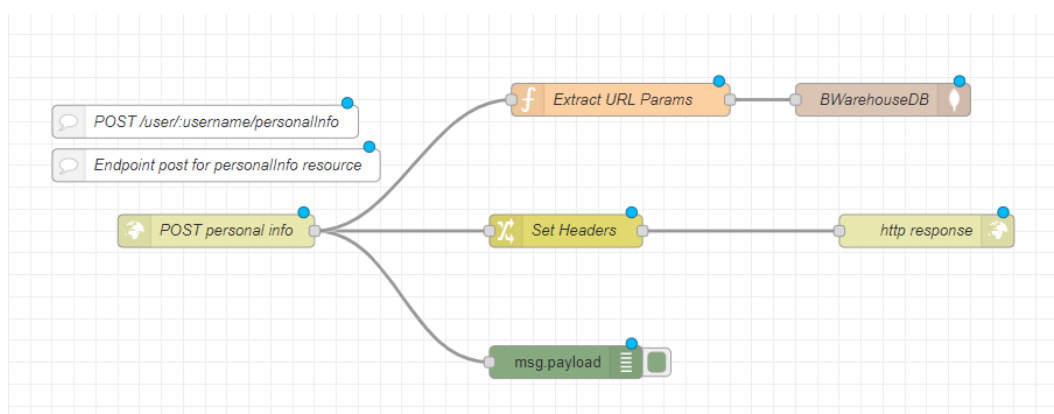


Σχήμα 56: Ροή endpoint PUT /user/{username}/wishlist.

3.1.6 Ροές πόρου PersonalInfo

Ροή του endpoint POST /user/{username}/personalInfo:

Η συγκεκριμένη ροή υλοποιεί την υπηρεσία μέσω της οποίας ο χρήστης μπορεί να εισάγει τα προσωπικά του στοιχεία στο σύστημα.

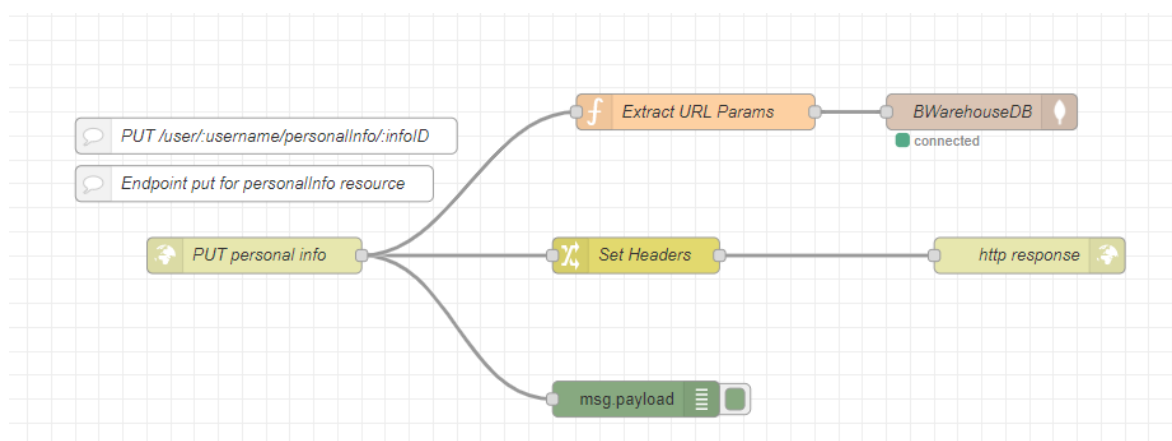


Σχήμα 57: Ροή endpoint POST /user/{username}/personalInfo.



Ροή του endpoint PUT /user/{username}/personalInfo/{infoID}:

Η συγκεκριμένη ροή υλοποιεί την υπηρεσία μέσω της οποίας ο χρήστης μπορεί να τροποποιεί τα προσωπικά του στοιχεία στο σύστημα.

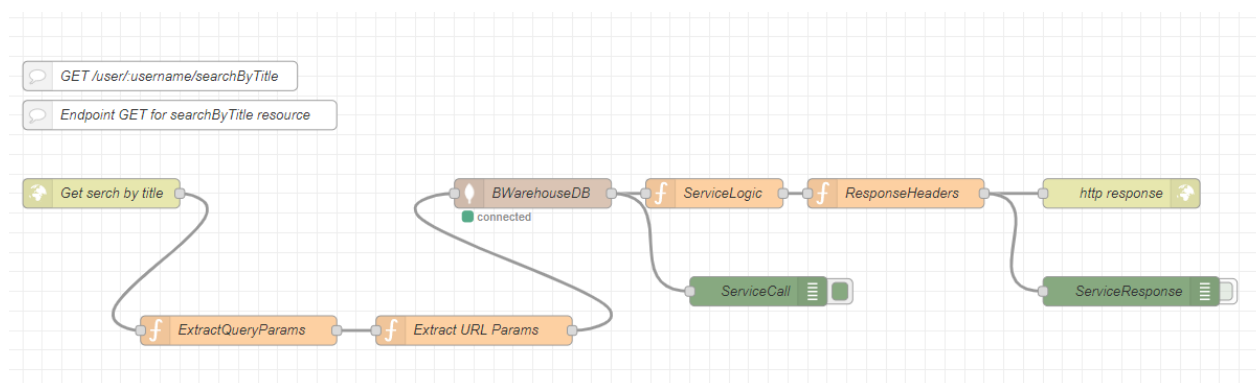


Σχήμα 58: Ροή endpoint PUT /user/{username}/personalInfo/{infoID}.

3.1.7 Ροή αλγοριθμικού πόρου SearchByTitle

Ροή του endpoint GET /user/{username}/searchByTitle:

Με την συγκεκριμένη υπηρεσία ο χρήστης μπορεί να κάνει αναζήτηση ενός βιβλίου χρησιμοποιώντας τον τίτλο του.



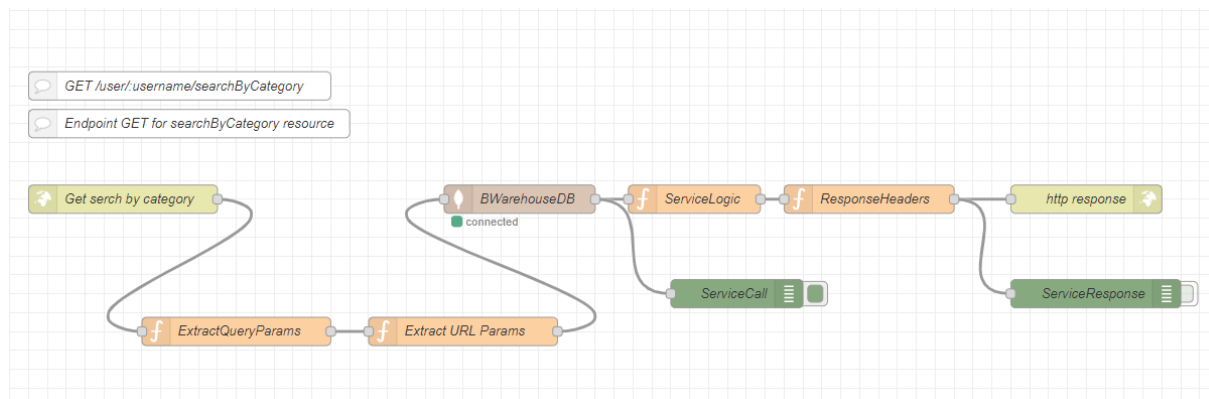
Σχήμα 59: Ροή endpoint GET /user/{username}/searchByTitle.



3.1.8 Ροή αλγοριθμικού πόρου SearchByCategory

Ροή του endpoint GET /user/{username}/searchByCategory:

Με την συγκεκριμένη υπηρεσία ο χρήστης μπορεί να κάνει αναζήτηση βιβλίων χρησιμοποιώντας την κατηγορία στην οποία ανήκουν.

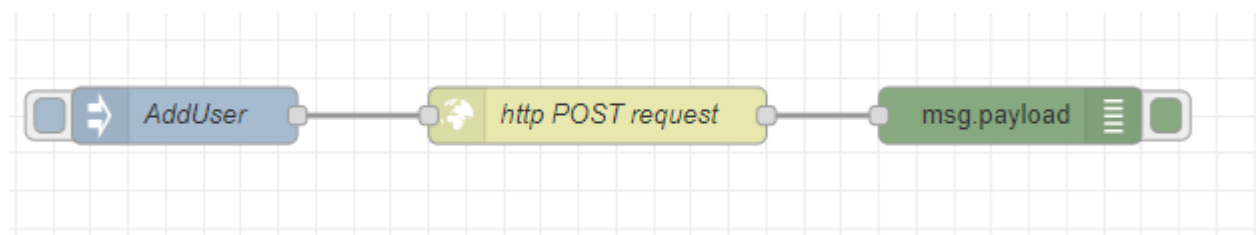


Σχήμα 60: Ροή endpoint GET /user/{username}/searchByCategory.

3.2 Υλοποίηση Ιστοριών χρήστη

3.2.1 Ιστορία Χρήστη: User create a new account

Ροή μέσω της οποίας ο χρήστης μπορεί να δημιουργήσει έναν καινούργιο λογαριασμό στο σύστημα, παρέχοντας ένα όνομα χρήστη και έναν κωδικό πρόσβασης.



Σχήμα 61: Ροή ιστορίας user create account.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
AddUser	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http POST request	http-request	Κάνει κλήση προς την υπηρεσία POST /user η οποία δίνει στο χρήστη τη δυνατότητα να δημιουργήσει έναν λογαριασμό.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη.



3.2.2 Ιστορία Χρήστη: User login to account

Ροή μέσω της οποίας ο χρήστης μπορεί να συνδεθεί στο λογαριασμό του, παρέχοντας το όνομα χρήστη και τον κωδικό πρόσβασης.



Σχήμα 62: Ροή ιστορίας user login to account.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
UserLogin	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http PUT request	http-request	Κάνει κλήση προς την υπηρεσία PUT /user/login η οποία δίνει στο χρήστη τη δυνατότητα να συνδεθεί σε ένα λογαριασμό.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη.

3.2.3 Ιστορία Χρήστη: User select a book

Ροή μέσω της οποίας ο χρήστης μπορεί να επιλέξει ένα βιβλίο, δηλαδή να του επιστραφούν από τον server τα χαρακτηριστικά του βιβλίου με id το {bookID}.



Σχήμα 63: Ροή ιστορίας user select book.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
GetBook	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http GET request	http-request	Κάνει κλήση προς την υπηρεσία GET /user/{username}/book/{bookID} η οποία δίνει στο χρήστη τη δυνατότητα να επιλέξει ένα βιβλίο για να δει τα χαρακτηριστικά του.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη, δηλαδή τον κατάλληλο κωδικό και το σώμα του επιλεγμένου βιβλίου.



3.2.4 Ιστορία Χρήστη: User rate a book

Ροή μέσω της οποίας ο χρήστης μπορεί να αξιολογήσει ένα βιβλίο.

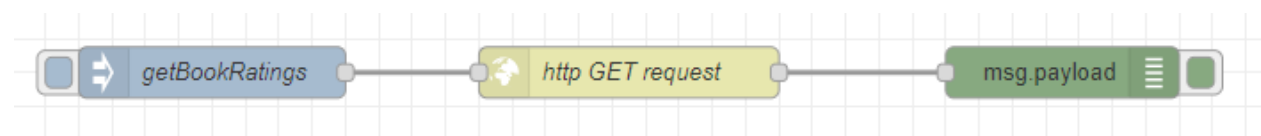


Σχήμα 64: Ροή ιστορίας user rate book.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
AddRating	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http POST request	http-request	Κάνει κλήση προς την υπηρεσία POST /user/{username}/book/{bookID}/rating η οποία δίνει στο χρήστη τη δυνατότητα να αξιολογήσει το συγκεκριμένο βιβλίο.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη.

3.2.5 Ιστορία Χρήστη: User see book's ratings

Ροή μέσω της οποίας ο χρήστης μπορεί να δει τις αξιολογήσεις ενός βιβλίου.



Σχήμα 65: Ροή ιστορίας user see book's ratings.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
getBookRatings	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http GET request	http-request	Κάνει κλήση προς την υπηρεσία GET /user/{username}/book/{bookID}/rating η οποία δίνει στο χρήστη τη δυνατότητα να δει τις αξιολογήσεις για το συγκεκριμένο βιβλίο.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη, δηλαδή τον κατάλληλο κωδικό και το σώμα των Rating του βιβλίου.



3.2.6 Ιστορία Χρήστη: User see a book's rating

Ροή μέσω της οποίας ο χρήστης μπορεί να δει μία συγκεκριμένη αξιολόγηση ενός βιβλίου.

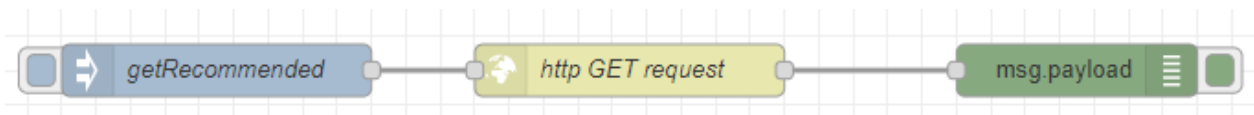


Σχήμα 66: Ροή ιστορίας user see a book's rating.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
getBookRating	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http GET request	http-request	Κάνει κλήση προς την υπηρεσία GET /user/{username}/book/{bookID}/rating/{ratingID} η οποία δίνει στο χρήστη τη δυνατότητα να δει την αξιολόγηση με id το {ratingID} του συγκεκριμένου βιβλίου.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη, δηλαδή τον κατάλληλο κωδικό και το σώμα του Rating του βιβλίου.

3.2.7 Ιστορία Χρήστη: User get recommended

Ροή μέσω της οποίας ο χρήστης μπορεί να δει τα προτεινόμενα για αυτόν βιβλία.



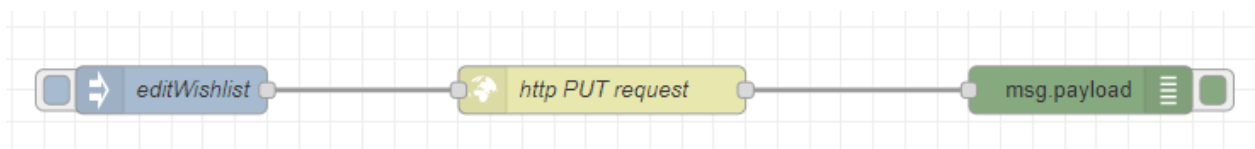
Σχήμα 67: Ροή ιστορίας user get recommended.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
getRecommended	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http GET request	http-request	Κάνει κλήση προς την υπηρεσία GET /user/{username}/recommended η οποία δίνει στο χρήστη τη δυνατότητα να δει τα προτεινόμενα για αυτόν βιβλία.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη, δηλαδή τον κατάλληλο κωδικό και τα προτεινόμενα βιβλία.



3.2.8 Ιστορία Χρήστη: User edit wishlist

Ροή μέσω της οποίας ο χρήστης μπορεί να τροποποιήσει τα περιεχόμενα του wishlist του.



Σχήμα 68: Ροή ιστορίας user edit wishlist.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
editWishlist	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http PUT request	http-request	Κάνει κλήση προς την υπηρεσία PUT /user/{username}/wishlist η οποία δίνει στο χρήστη τη δυνατότητα να τροποποιήσει το wishlist του.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη.

3.2.9 Ιστορία Χρήστη: User add personal information

Ροή μέσω της οποίας ο χρήστης μπορεί να εισάγει τα προσωπικά του στοιχεία στο σύστημα.



Σχήμα 69: Ροή ιστορίας user add personal information.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
addPersonalInfo	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http POST request	http-request	Κάνει κλήση προς την υπηρεσία POST /user/{username}/personalInfo η οποία δίνει στο χρήστη τη δυνατότητα να εισάγει τα προσωπικά του στοιχεία.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη.



3.2.10 Ιστορία Χρήστη: User edit personal information

Ροή μέσω της οποίας ο χρήστης μπορεί να τροποποιεί τα προσωπικά του στοιχεία.

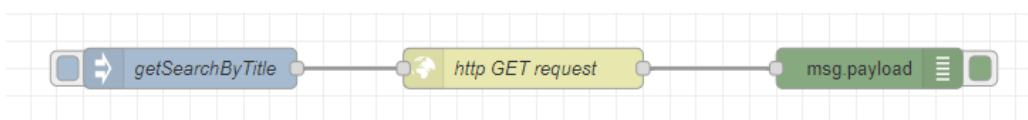


Σχήμα 70: Ροή ιστορίας user edit personal information.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
editPersonalInfo	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http PUT request	http-request	Κάνει κλήση προς την υπηρεσία PUT /user/{username}/personalInfo/{personalInfo} η οποία δίνει στο χρήστη τη δυνατότητα να τροποποιήσει τα προσωπικά του στοιχεία.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη.

3.2.11 Ιστορία Χρήστη: User search book by title

Ροή μέσω της οποίας ο χρήστης έχει την δυνατότητα να αναζητήσει ένα βιβλίο με βάση τον τίτλο του.



Σχήμα 71: Ροή ιστορίας user search book by title.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
getSearchByTitle	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http GET request	http-request	Κάνει κλήση προς την υπηρεσία GET /user/{username}/searchByTitle η οποία δίνει στο χρήστη τη δυνατότητα να κάνει αναζήτηση ενός βιβλίου με βάση τον τίτλο του.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη, δηλαδή τον κατάλληλο κωδικό και τα επιστρεφόμενα βιβλία της αναζήτησης.



3.2.12 Ιστορία Χρήστη: User search book by category

Ροή μέσω της οποίας ο χρήστης μπορεί να κάνει αναζήτηση βιβλίων με βάση την κατηγορία στην οποία ανήκουν.



Σχήμα 72: Ροή ιστορίας user search book by category.

Όνομα κόμβου	Τύπος κόμβου	Περιγραφή
getSearchByCategory	Inject	Χρησιμοποιείται για την ενεργοποίηση της εκτέλεσης της ροής και για τον καθορισμό των παραμέτρων.
http GET request	http-request	Κάνει κλήση προς την υπηρεσία GET /user/{username}/searchByCategory η οποία δίνει στο χρήστη τη δυνατότητα να κάνει αναζήτηση βιβλίων με βάση την κατηγορία τους.
Msg.payload	debug	Τυπώνει στην κονσόλα το response του συστήματος για τη συγκεκριμένη κλήση της υπηρεσίας από το χρήστη, δηλαδή τον κατάλληλο κωδικό και τα επιστρεφόμενα βιβλία της αναζήτησης.



Παράρτημα Ι – Ανοιχτά Θέματα

1. Ορισμένες από τις ροές, και επομένως και από τις Λειτουργικές Απαιτήσεις, που έχουμε ορίσει και προδιαγράψει απαιτούν την υλοποίηση περαιτέρω ροών για να προσφέρουν λειτουργικότητα σε πραγματικές συνθήκες. Για παράδειγμα, στην μέχρι τώρα ανάπτυξη της εφαρμογής μας, ο χρήστης μπορεί να προσθαφαιρέσει βιβλία στο wishlist του, αλλά χωρίς την υλοποίηση μιας Λειτουργικής Απαιτήσης για να μπορεί να δει τα βιβλία αυτά, καθώς και της αντίστοιχης ροής, η πρώτη δεν προσφέρει κάποιο πρακτικό όφελος. Παρόμοια θέματα υπάρχουν στο κομμάτι της αποσύνδεσης των χρηστών από την εφαρμογή, καθώς και των ροών που θα πραγματοποιούνται σε οριακές συνθήκες, δηλαδή κατά τις πρώτες στιγμές της λειτουργίας της εφαρμογής, τις διαφορές περιπτώσεις σφαλμάτων που μπορεί να προκύψουν, καθώς και την επαναλειτουργία της εφαρμογής μετά από αυτά. Επομένως, σε μία πραγματική εφαρμογή θα απαιτούνταν στη συνέχεια της διαδικασίας ανάπτυξής της, η προσθήκη περαιτέρω λειτουργικότητας και των αντίστοιχων ροών για την κάλυψη των συγκεκριμένων αναγκών.
2. Απαιτείται επίσης ο σχεδιασμός των ροών που υλοποιούν το front-end μέρος της εφαρμογής, δηλαδή το κομμάτι που ασχολείται με τη μετατροπή και την παρουσίαση των δεδομένων που παρέχονται από τον server σε κατάλληλη μορφή, έτσι ώστε οι χρήστες να μπορούν να δουν και να αλληλεπιδράσουν με τα δεδομένα αυτά.