Πρότυπο Δοκιμών Αποδοχής Χρήστη (UAT)

Ολοκληρωμένο Πλαίσιο για τη Διεξαγωγή Δοκιμών Αποδοχής Χρήστη

|  |  |
| --- | --- |
| **Όνομα Έργου** | Φοιτητολόγιο++ |
| **Μέλη Ομάδας Έργου** | Ιερωνυμίδης Μιχαήλ (22390065)  Γαλάνης Ιωάννης (22390288)  Ζούμπος Γεώργιος (22390058) |
| **Υπεύθυνος UAT** | [Εισαγάγετε το Όνομα του Υπεύθυνου UAT] |
| **Ημερομηνία Έναρξης UAT** | 16/06/2025 |
| **Ημερομηνία Λήξης UAT** | [Εισαγάγετε την Ημερομηνία Λήξης] |

# Στόχοι UAT

* Να διασφαλιστεί ότι το σύστημα εκπληρώνει όλες τις καθορισμένες επιχειρηματικές απαιτήσεις.
* Να επιβεβαιωθεί η σωστή λειτουργία των ροών εργασίας των χρηστών όπως προβλέπεται.
* Να εντοπιστούν και να καταγραφούν τυχόν ζητήματα ή ελλείψεις πριν από την επίσημη ανάπτυξη.

# Περιβάλλον Δοκιμών

* Τύπος Περιβάλλοντος: [Ανάπτυξη/Δοκιμή/Προετοιμασία]
* Στοιχεία Πρόσβασης: [Εισαγάγετε τα Διαπιστευτήρια Πρόσβασης]
* Απαιτήσεις Προγράμματος Περιήγησης/Συσκευής: [Εισαγάγετε τις Λεπτομέρειες]

# Προσέγγιση Δοκιμών

* Οι περιπτώσεις δοκιμών θα καθοριστούν βάσει λεπτομερών επιχειρηματικών σεναρίων.
* Τα βήματα και τα αναμενόμενα αποτελέσματα πρέπει να τεκμηριώνονται σχολαστικά.
* Τα πραγματικά αποτελέσματα και τυχόν αποκλίσεις θα καταγράφονται και θα αναλύονται λεπτομερώς.

# Log In (Correct Input)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC1-U-1 |
| **Περιγραφή** | Έλεγχος περίπτωσης χρήσης Log In με σωστά δεδομένα. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει χρήστης (φοιτητής, καθηγητής) στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | {  “userID”:1,  “userPWD”:“password”  “userType”:“student”  } |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε τα δεδομένα μας στο POST /login route της nodejs εφαρμογής. |
| Bήμα 2 :Χρησιμοποιούμε το JSON web token που μας αποστέλλεται για να επαληθεύσουμε την εγκυρότητα του. |
| **Αναμενόμενο αποτέλεσμα** | Έγκυρο JSON web token που μπορεί να χρησιμοποιηθεί για να πραγματοποιήσει ενέργειες χρήστη τύπου φοιτητή. |
| **Πραγματικό**  **αποτέλεσμα** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (Έγκυρο JSON web token) |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Log In (Missing Input)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC2-U-1 |
| **Περιγραφή** | Έλεγχος περίπτωσης χρήσης Log In με ελλείποντα δεδομένα. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει χρήστης (φοιτητής, καθηγητής) στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | {  “userID”:1,  “userType”:“student”  } |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε τα δεδομένα μας στο POST /login route της nodejs εφαρμογής. |
| **Αναμενόμενο αποτέλεσμα** | Warning της εφαρμογής για ελλιπή δεδομένα. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Invalid Inputs"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Log In (Invalid Password)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC3-U-1 |
| **Περιγραφή** | Έλεγχος περίπτωσης χρήσης Log In με λάθος κωδικό. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει χρήστης (φοιτητής, καθηγητής) στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | {  “userID”:1,  “userPWD”:“wrong”,  “userType”:“student”  } |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε τα δεδομένα μας στο POST /login route της nodejs εφαρμογής. |
| **Αναμενόμενο αποτέλεσμα** | Warning της εφαρμογής για λάθος credentials. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Invalid Credentials "  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# View Managed Courses (Authorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC1-ΤΕ-100 |
| **Περιγραφή** | Έλεγχος περίπτωσης εμφάνισης διαχειριζόμενων μαθημάτων από εξουσιοδοτημένο χρήστη (Καθηγητή). |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον ένας χρήστης καθηγητή, και μια εγγραφή μαθήματος στην βάση δεδομένων. Τέλος, πρέπει ο υπάρχων καθηγητής να έχει τεθεί ως διαχειριστής του μαθήματος. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο GET /managedCourses route της nodejs εφαρμογής, πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Σύνολο διαχειριζόμενων μαθημάτων καθηγητή. |
| **Πραγματικό**  **αποτέλεσμα** | [  {  "CID": 1,  "NAME": "Microelectronics",  "SEMESTER": 6,  "TID": 1  }  ] |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# View Managed Courses (Unauthorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC2-ΤΕ-100 |
| **Περιγραφή** | Έλεγχος περίπτωσης εμφάνισης διαχειριζόμενων μαθημάτων από μη εξουσιοδοτημένο χρήστη (φοιτητή). |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον ένας χρήστης φοιτητή, και μια εγγραφή μαθήματος στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο GET /managedCourses route της nodejs εφαρμογής, πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα απουσίας εξουσιοδότησης. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Invalid token"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# View Students of Managed Course (Authorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC1-ΤΕ-110 |
| **Περιγραφή** | Έλεγχος περίπτωσης εμφάνισης φοιτητών διαχειριζόμενων μαθημάτων από εξουσιοδοτημένο χρήστη (καθηγητή). |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον ένας χρήστης καθηγητή, μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. Τέλος, πρέπει ο υπάρχων καθηγητής να έχει τεθεί ως διαχειριστής του μαθήματος και ο υπάρχων μαθητής να έχει εγγραφεί στο μάθημα. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο GET /managedCourses/:courseID/students route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό μαθήματος του οποίου τους μαθητές θέλουμε να λάβουμε) (στην δική μας περίπτωση το μάθημα αυτό είναι η Μικροηλεκτρονική με αναγνωριστικό 1), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Σύνολο μαθητών που παρακολουθεί το μάθημα της μικροηλεκτρονικής (που δεν έχουν βαθμολογηθεί ακόμα). |
| **Πραγματικό**  **αποτέλεσμα** | [  {  "SID":1,  "NAME":"Michael",  "SURNAME":"Ieronymidis"  }  ] |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# View Students of Not Managed Course (Authorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC2-ΤΕ-110 |
| **Περιγραφή** | Έλεγχος περίπτωσης εμφάνισης φοιτητών μη διαχειριζόμενων μαθημάτων από εξουσιοδοτημένο χρήστη (καθηγητή). |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον ένας χρήστης καθηγητή, μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. Τέλος, ο υπάρχων καθηγητής δεν πρέπει να έχει τεθεί ως διαχειριστής του μαθήματος και ο υπάρχων μαθητής να έχει εγγραφεί στο μάθημα. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο GET /managedCourses/:courseID/students route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό μαθήματος του οποίου τους μαθητές θέλουμε να λάβουμε) (στην δική μας περίπτωση το μάθημα αυτό είναι η Ηλεκτρονική με αναγνωριστικό 2 καθώς ο καθηγητής με αναγνωριστικό TID=1 δεν έχει πρόσβαση σε αυτο), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα μη εξουσιοδότησης πρόσβασης μαθήματος. |
| **Πραγματικό**  **αποτέλεσμα** | [  ] |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | Στην παρούσα κατάσταση ο καθηγητής που δεν έχει πρόσβαση σε ένα μάθημα απλά λαμβάνει ως αποτέλεσμα το κενό και όχι μήνυμα. Αυτό μπορεί να λυθεί στο front end του συστήματος καθώς ένας καθηγητής που δεν είναι υπεύθυνος ενός μαθήματος δεν θα μπορεί να κάνει api call για το μάθημα αυτό. |

# View Students of Managed Course (Unauthorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC3-ΤΕ-110 |
| **Περιγραφή** | Έλεγχος περίπτωσης εμφάνισης φοιτητών μη διαχειριζόμενων μαθημάτων από μη εξουσιοδοτημένο χρήστη (φοιτητή). |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (JSON Web Token Φοιτητή με SID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο GET /managedCourses/:courseID/students route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό μαθήματος του οποίου τους μαθητές θέλουμε να λάβουμε) (στην δική μας περίπτωση το μάθημα αυτό είναι η Μικροηλεκτρονική με αναγνωριστικό 1), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα μη εξουσιοδότησης. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Access denied"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Grade Student (Authorized User)(Correct Grade)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC1-ΤΕ-111 |
| **Περιγραφή** | Έλεγχος περίπτωσης βαθμολογίας φοιτητή από εξουσιοδοτημένο χρήστη (καθηγητή), διαχειριστή μαθήματος. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον ένας χρήστης καθηγητή, μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. Τέλος, πρέπει ο υπάρχων καθηγητής να έχει τεθεί ως διαχειριστής του μαθήματος και ο υπάρχων μαθητής να έχει εγγραφεί στο μάθημα και να μην έχει βαθμολογηθεί. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1)  {  “grade”:”8”  }  (Request Body) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Ελέγχουμε τους φοιτητές του επιλεγμένου μαθήματος για να σιγουρευτούμε ότι τουλάχιστον ένας δεν έχει βαθμολογηθεί και υπάρχει εγγεγραμμένος στο μάθημα. |
| Βήμα 2: Αποστέλλουμε request στο PATCH /managedCourses/:courseID/students/:studentID route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό μαθήματος του οποίου τον φοιτητή βαθμολογούμε και το :studentID είναι το αναγνωριστικό φοιτητή που θέλουμε να βαθμολογήσουμε) (στην δική μας περίπτωση το μάθημα αυτό είναι η Μικροηλεκτρονική με αναγνωριστικό 1 και φοιτητής με αναγνωριστικό 1), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας και την βαθμολογίας μας. |
| Βήμα 3: Ελέγχουμε αν ο φοιτητής που βαθμολογήθηκε έχει αφαιρεθεί από τους φοιτητές του μαθήματος. |
| **Αναμενόμενο αποτέλεσμα** | Επιτυχημένη βαθμολόγηση φοιτητή |
| **Πραγματικό**  **αποτέλεσμα** | [  {  "SID": 1,  "NAME": "Michael",  "SURNAME": "Ieronymidis"  }  ] |
| Success |
| [  ] |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Grade Student (Authorized User) (Incorrect Grade)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC2-ΤΕ-111 |
| **Περιγραφή** | Έλεγχος περίπτωσης λάθος βαθμολογίας φοιτητή από εξουσιοδοτημένο χρήστη (καθηγητή), διαχειριστή μαθήματος. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον ένας χρήστης καθηγητή, μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. Τέλος, πρέπει ο υπάρχων καθηγητής να έχει τεθεί ως διαχειριστής του μαθήματος και ο υπάρχων μαθητής να έχει εγγραφεί στο μάθημα και να μην έχει βαθμολογηθεί. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1)  {  “grade”:”-5”  }  (Request Body) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο PATCH /managedCourses/:courseID/students/:studentID route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό μαθήματος του οποίου τον φοιτητή βαθμολογούμε και το :studentID είναι το αναγνωριστικό φοιτητή που θέλουμε να βαθμολογήσουμε) (στην δική μας περίπτωση το μάθημα αυτό είναι η Μικροηλεκτρονική με αναγνωριστικό 1 και φοιτητής με αναγνωριστικό 1), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας και την λανθασμένη βαθμολογία. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα Σφάλματος |
| **Πραγματικό**  **αποτέλεσμα** | {  "error": "Incorrect Grade"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Grade Student (Unauthorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC3-ΤΕ-111 |
| **Περιγραφή** | Έλεγχος περίπτωσης λάθος βαθμολογίας φοιτητή από μη εξουσιοδοτημένο χρήστη (φοιτητή). |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (JSON Web Token Φοιτητή με SID=1)  {  “grade”:”5”  }  (Request Body) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο PATCH /managedCourses/:courseID/students/:studentID route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό μαθήματος του οποίου τον φοιτητή βαθμολογούμε και το :studentID είναι το αναγνωριστικό φοιτητή που θέλουμε να βαθμολογήσουμε) (στην δική μας περίπτωση το μάθημα αυτό είναι η Μικροηλεκτρονική με αναγνωριστικό 1 και φοιτητής με αναγνωριστικό 1), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας και την λανθασμένη βαθμολογία. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα Σφάλματος |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Invalid token"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# View All Registered Courses (Authorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC1-ST-100 |
| **Περιγραφή** | Έλεγχος περίπτωσης εμφάνισης μαθημάτων στα οποία έχει εγγραφεί ένας εξουσιοδοτημένος χρήστης (φοιτητής). |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. Πρέπει επίσης ο φοιτητής να έχει εγγραφεί σε τουλάχιστον ένα μάθημα. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (JSON Web Token Φοιτητή Με SID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο GET /registeredCourses route της nodejs εφαρμογής, πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μαθήματα στα οποία είναι εγγεγραμμένος ο φοιτητής. |
| **Πραγματικό**  **αποτέλεσμα** | [  {  "CID": 1,  "NAME": "Microelectronics",  "GRADE": null,   "AVAILCOURSES": 7  },  {  "CID": 2,  "NAME": "Electronics",  "GRADE": null,  "AVAILCOURSES": 7  }  ] |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# View All Registered Courses (Unauthorized User)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC2-ST-100 |
| **Περιγραφή** | Έλεγχος περίπτωσης εμφάνισης μαθημάτων από χρήστη καθηγητή. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή καθηγητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο GET /registeredCourses route της nodejs εφαρμογής, πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα σφάλματος. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Invalid token"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Register Course (Authorized User)(Existing Course)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC1-ST-110 |
| **Περιγραφή** | Έλεγχος περίπτωσης εγγραφής σε μάθημα από χρήστη φοιτητή. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (JSON Web Token Φοιτητή Με SID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο PUT /registeredCourses/:courseID route της nodejs εφαρμογής (όπου :courseID είναι το αναγνωριστικό του μαθήματος στο οποίο θέλει να εγγραφεί ο φοιτητής) (στην περίπτωση μας θα χρησιμοποιήσουμε το μάθημα της ηλεκτρονικής με αναγνωριστικό CID=2), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα επιτυχίας εγγραφής. |
| **Πραγματικό**  **αποτέλεσμα** | Registered To Course |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Register Course (Authorized User)(Non Existing Course)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC2-ST-110 |
| **Περιγραφή** | Έλεγχος περίπτωσης εγγραφής σε μη υπάρχον μάθημα από χρήστη φοιτητή. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή φοιτητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (JSON Web Token Φοιτητή Με SID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο PUT /registeredCourses/:courseID route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό του μαθήματος στο οποίο θέλει να εγγραφεί ο καθηγητής) (στην περίπτωση μας θα χρησιμοποιήσουμε ένα μη υπάρχον μάθημα με αναγνωριστικό CID=3), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα σφάλματος. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error": "Course does not exist."  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Register Course (Unauthorized User)(Existing Course)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC3-ST-110 |
| **Περιγραφή** | Έλεγχος περίπτωσης εγγραφής σε μάθημα από χρήστη καθηγητή. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή καθηγητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο PUT /registeredCourses/:courseID route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό του μαθήματος στο οποίο θέλει να εγγραφεί ο καθηγητής) (στην περίπτωση μας θα χρησιμοποιήσουμε το μάθημα της ηλεκτρονικής με αναγνωριστικό CID=2), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα σφάλματος. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Invalid token"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Remove Course (Authorized User)(Existing Course)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC1-ST-120 |
| **Περιγραφή** | Έλεγχος περίπτωσης απεγγραφής από μάθημα από χρήστη φοιτητή. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή φοιτητή στην βάση δεδομένων. Πρέπει επίσης ο φοιτητής να είναι εγγεγραμμένος στο μάθημα αυτό. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (JSON Web Token Φοιτητή Με SID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο DELETE /registeredCourses/:courseID route της nodejs εφαρμογής (όπου :courseID είναι το αναγνωριστικό του μαθήματος από το οποίο θέλει να απεγγραφεί ο φοιτητής) (στην περίπτωση μας θα χρησιμοποιήσουμε το μάθημα της Μικροηλεκτρονικής με αναγνωριστικό CID=1), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα επιτυχίας απεγγραφής. |
| **Πραγματικό**  **αποτέλεσμα** | Removed From Course |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Remove Course (Authorized User)(Non Existing Course)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC2-ST-120 |
| **Περιγραφή** | Έλεγχος περίπτωσης απεγγραφής από μάθημα το οποίο δεν υπάρχει. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή φοιτητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMTAwOTQ0LCJleHAiOjE3NTAxMDgxNDR9.  sXjj3kAkBtW7wU71E6K4Zsvu9\_oyViWp-GdF6qk2lA0  (JSON Web Token Φοιτητή Με SID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο DELETE /registeredCourses/:courseID route της nodejs εφαρμογής (όπου :courseID είναι το αναγνωριστικό του μαθήματος από το οποίο θέλει να απεγγραφεί ο φοιτητής) (στην περίπτωση μας θα χρησιμοποιήσουμε ένα μη υπάρχον μάθημα με αναγνωριστικό CID=3), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα σφάλματος εύρεσης μαθήματος. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Course not found"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |

# Remove Course (Unauthorized User)(Existing Course)

|  |  |
| --- | --- |
| **ID Περίπτωσης Δοκιμής** | TC3-ST-120 |
| **Περιγραφή** | Έλεγχος περίπτωσης απεγγραφής από μάθημα από χρήστη καθηγητή. |
| **Προϋποθέσεις** | Πρέπει να υπάρχει τουλάχιστον μια εγγραφή μαθήματος καθώς και μια εγγραφή καθηγητή στην βάση δεδομένων. |
| **Δεδομένα εισόδου** | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  eyJ1c2VySUQiOiIxIiwidXNlclR5cGUiOiJzdHVkZW50IiwiaWF0IjoxNzUwMDc2NzA0LCJleHAiOjE3NTAwODM5MDR9.  igqlyj\_SIytZakA4t18ONEpMxKcZsGuVi7W79vKPY3A  (JSON Web Token Καθηγητή με TID=1) |
| **Βήματα Εκτέλεσης** | Βήμα 1: Αποστέλλουμε request στο DELETE /registeredCourses/:courseID route της nodejs εφαρμογής (οπου :courseID είναι το αναγνωριστικό του μαθήματος από το οποίο θέλει να απεγγραφεί ο καθηγητής) (στην περίπτωση μας θα χρησιμοποιήσουμε το μάθημα της ηλεκτρονικής με αναγνωριστικό CID=2), πρέπει επίσης να επισημάνουμε το token μας στο authorization header του request μας. |
| **Αναμενόμενο αποτέλεσμα** | Μήνυμα σφάλματος. |
| **Πραγματικό**  **αποτέλεσμα** | {  "error":"Invalid token"  } |
| **Κατάσταση:** | Επιτυχία |
| **Σχόλια** | - |