

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής



Εργασία Μαθήματος «Τεχνολογίες Blockchain και  
εφαρμογές»

Προγραμματιστική Εργασία Μαθήματος	Εφαρμογή για τη διαχείριση των βαθμολογιών των προπτυχιακών φοιτητών ενός πανεπιστημίου.
Εκπαιδευτές	<b>Κάραλης Α.</b>
Όνομα φοιτητή – Αρ. Μητρώου	Αργυροπούλου Μαρία : Π18011
	Στεργίου Χρήστος: Π18147
Ημερομηνία παράδοσης	07-09-2023

## Περιεχόμενα

Ανάλυση του κώδικα .....	3
Contract Deployment .....	6
Παράδειγμα εκτέλεσης του κώδικα .....	8
Λειτουργίες διαχειριστή .....	8
Εισαγωγή και διαγραφή διαχειριστών .....	8
Εισαγωγή και διαγραφή γραμματείας .....	10
Εκτέλεση άλλων λειτουργιών που υποστηρίζονται από την εφαρμογή .....	11
Λειτουργίες γραμματείας .....	11
Καθορισμός μαθημάτων .....	11
Καθορισμό διδασκόντων .....	12
Καθορισμός φοιτητών .....	13
Εισαγωγή ή τροποποίηση βαθμολογιών .....	14
Καθορισμός προθεσμίας υποβολής των βαθμολογιών .....	15
Λειτουργίες καθηγητών .....	15
Λήψη της λίστας φοιτητών .....	15
Ανάρτηση των βαθμολογιών .....	16
Λειτουργίες φοιτητών .....	17
Λήψη βαθμού για ένα συγκεκριμένο μάθημα .....	17
Λήψη της λίστας με τις βαθμολογίες για όλα τα μαθήματα .....	17

## Ανάλυση του κώδικα

Αρχικά δηλώνονται η άδεια χρήσης του συμβολαίου και οι εκδόσεις της γλώσσας Solidity που υποστηρίζει ο κώδικας.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.13;
```

Δηλώνονται οι μεταβλητές που θα αποθηκεύουν τα δεδομένα. Γίνεται χρήση mappings για την αντιστοίχιση διευθύνσεων σε χρήστες και ενός struct που περιγράφει τη δομή του μαθήματος με τα αντίστοιχα πεδία. Επιπλέον χρησιμοποιούνται nested mappings για να αποθηκευτούν οι βαθμοί, η λίστα μαθημάτων ανά ακαδημαϊκό έτος και ένα "flag" για να ελέγχει το αν κάποιος καθηγητής έχει ήδη αναρτήσει βαθμολογία για κάποιο μάθημα.

```
contract UniversityGrades {
    address public admin;
    mapping(address => bool) public admins;
    mapping(address => bool) public secretaries;
    mapping(address => bool) public instructors;
    mapping(address => bool) public students;
```

```
// Course details
struct Course {
    bytes32 name;
    address[] instructors;
    address[] students;
}
```

```
mapping(uint16 => mapping(bytes32 => Course)) public courses;
mapping (uint16 => mapping(bytes32 => mapping (address => uint256))) public grades;
mapping (uint16 => mapping(bytes32 => bool)) public flag;
uint256 public gradesDeadline;
```

Κατά την κλήση του συμβολαίου καλείται ο constructor και ορίζει σαν αρχικό administrator την διεύθυνση που έκανε deploy το contract, αρχικοποιεί το ακαδημαϊκό έτος και θέτει την προθεσμία υποβολής βαθμολογιών ως τέσσερις εβδομάδες από την ημερομηνία που δημιουργήθηκε το συμβόλαιο. Στην υλοποίηση του κώδικα η ημερομηνία λαμβάνει τη μορφή unix timestamp.

```
constructor() { // @ infinite gas infinite gas
    admin = msg.sender;
    admins[msg.sender] = true;
    currentAcademicYear = 2023;
    // Set a default deadline (unix timestamp) for grade submissions
    gradesDeadline = block.timestamp + 4 weeks;
}
```

Προκειμένου να εξασφαλιστεί πως ο χρήστης επιτρέπεται να καλέσει τις συναρτήσεις που αντιστοιχούν στην αρμοδιότητά του, χρησιμοποιούνται τα παρακάτω modifiers κατά την δήλωση των λειτουργιών. Σε περίπτωση που ο χρήστης προσπαθήσει να καλέσει μία λειτουργία εκτός της δικαιοδοσίας του, εμφανίζεται μήνυμα σφάλματος.

```

modifier onlyAdmin() {require(admins[msg.sender], "Only administrators allowed");}
modifier onlySecretary() {require(secretaries[msg.sender] || admins[msg.sender], "Only secretaries and admins allowed");}
modifier onlyInstructor() {require(instructors[msg.sender], "Only instructors allowed");}
modifier onlyStudent() {require(students[msg.sender], "Only students allowed");}

```

Οι διαχειριστές της εφαρμογής αναλαμβάνουν να εισάγουν και να διαγράψουν χρήστες καθώς και να εκτελέσουν οποιαδήποτε λειτουργία έχουν δικαίωμα και οι γραμματείς. Η εισαγωγή ενός χρήστη γίνεται αντιστοιχίζοντας τη διεύθυνσή του σε true στο αντίστοιχο mapping ενώ σε false κατά τη διαγραφή του. Οι συναρτήσεις **'addAdmin'**, **'deleteAdmin'**, **'addSecretary'**, και **'deleteSecretary'** επιτρέπουν στους διαχειριστές να προσθέσουν ή να αφαιρέσουν άλλους διαχειριστές και γραμματείς. Ενώ οι συναρτήσεις **'addInstructor'**, **'addStudent'**, **'deleteInstructor'**, και **'deleteStudent'** επιτρέπουν στους διαχειριστές και τις γραμματείς να προσθέσουν ή να αφαιρέσουν διδάσκοντες και φοιτητές.

```

//Admin's part
function addAdmin(address _newAdmin) external onlyAdmin {admins[_newAdmin] = true;}
function deleteAdmin(address _admin) external onlyAdmin {admins[_admin] = false;}
function addSecretary(address _newSecretary) external onlyAdmin {secretaries[_newSecretary] = true;}
function deleteSecretary(address _secretary) external onlyAdmin {secretaries[_secretary] = false;}

function addInstructor(address _newInstructor) external onlySecretary {instructors[_newInstructor] = true;}
function addStudent(address _newStudent) external onlySecretary {students[_newStudent] = true;}
function deleteInstructor(address _instructor) external onlySecretary {instructors[_instructor] = false;}
function deleteStudent(address _student) external onlySecretary {students[_student] = false;}

```

Οι λειτουργίες των γραμματέων αποτελούνται από την εισαγωγή μαθημάτων στο πρόγραμμα σπουδών ανά ακαδημαϊκό έτος, η ανάθεση διδασκόντων στα μαθήματα, η ανάθεση φοιτητών στα μαθήματα, η εισαγωγή ή τροποποίηση βαθμολογίας για τα μαθήματα καθώς και ο ορισμός ημερομηνίας υποβολής βαθμολογιών.

Για την δημιουργία ενός μαθήματος απαιτούνται το όνομα και το ακαδημαϊκό έτος, γίνεται έλεγχος ως προς το όνομα να μην είναι κενό και ύστερα εισάγεται στο mapping με κενούς πίνακες τα ονόματα των διδασκόντων και των φοιτητών.

```

// Secretary's functions
// function to add a course for a specific year
function createCourse(uint16 _courseName, uint16 _academicYear) external onlySecretary{
    require(_courseName.length > 0, "Course name can't be empty");
    courses[_academicYear][_courseName] = Course({
        name: _courseName,
        instructors: new address[](0),
        students: new address[](0));
}

```

Για την ανάθεση διδασκόντων στο μάθημα, απαιτείται το ακαδημαϊκό έτος του μαθήματος και η διεύθυνση του καθηγητή που εισάγεται στο πεδίο-πίνακα του εκάστοτε μαθήματος. Παρόμοια είναι και η διαδικασία για τους φοιτητές με επιπλέον βήμα την αρχικοποίηση της βαθμολογίας.

```

function enrollInstructor(uint16 _academicYear, address _instructor, string memory _courseName) external onlySecretary{
    courses[_academicYear][_courseName].instructors.push(_instructor);
}

function enrollStudent(uint16 _academicYear, address _student, string memory _courseName) external onlySecretary{
    courses[_academicYear][_courseName].students.push(_student);
    grades[_academicYear][_courseName][_student]=1; //0 is solidity to set 1 to 0
}

```

Για τον ορισμό της βαθμολογίας χρειάζονται ως παράμετροι το ακαδημαϊκό έτος του μαθήματος, το όνομα του μαθήματος, η διεύθυνση του μαθητή και ο βαθμός. Γίνονται έλεγχοι ως προς το όνομα να είναι μη κενό και να έχει εγγραφεί ο φοιτητής στο μάθημα.

```
function setGrade(uint16 _academicYear, uint8 _grade, address _student, string memory _courseName) external onlySecretary{
    require(bytes(courses[_academicYear][_courseName].name).length > 0, "Course entry does not exist.");
    require(grades[_academicYear][_courseName][_student] != 0, "Student isn't enrolled yet");
    require(_grade != 0, "Grade can not be 0");
    grades[_academicYear][_courseName][_student] = _grade;
}
```

Για τον καθορισμό της καταλυτικής ημερομηνίας υποβολής των βαθμολογιών χρειάζεται μόνο η ημερομηνία σε μορφή ακεραίου (π.χ. 1694120399).

```
function setGradesDeadline(uint256 _deadline) external onlySecretary {gradesDeadline = _deadline;} 27877 gas
```

Οι λειτουργίες των διδασκόντων περιλαμβάνουν περιλαμβάνουν τη λήψη της λίστας των μαθητών για το μάθημα που δίδαξε ένα ακαδημαϊκό έτος και την ανάρτηση των βαθμολογιών του μαθήματος που δίδαξε ένα συγκεκριμένο ακαδημαϊκό έτος.

Για την λήψη της λίστας των φοιτητών χρειάζονται ως παράμετροι το όνομα του μαθήματος και το ακαδημαϊκό έτος. Γίνεται έλεγχος ώστε το πεδίο του ονόματος να υπάρχει και έπειτα επιστρέφονται οι εγγεγραμμένοι φοιτητές.

```
function getStudents(string memory _courseName, uint16 _academicYear) external onlyInstructor view returns (address[] memory){
    require(bytes(courses[_academicYear][_courseName].name).length > 0, "Course does not exist.");
    return courses[_academicYear][_courseName].students;
}
```

Για την ανάρτηση των βαθμολογιών χρειάζονται ως παράμετροι το όνομα του μαθήματος και το ακαδημαϊκό έτος. Έπειτα ελέγχεται το αν έχει παρέλθει η προθεσμία ανάρτησης βαθμολογιών που έχει οριστεί από τη γραμματεία, εάν έχει ολοκληρωθεί η διαδικασία από άλλον διδάσκοντα και εάν ο χρήστης ανήκει στους διδάσκοντες του μαθήματος. Για τον τελευταίο έλεγχο υλοποιήθηκε η βοηθητική συνάρτηση findPerson.

```
function postGrades(string memory _courseName, uint16 _academicYear, uint8[] memory _grades) external onlyInstructor{
    require(block.timestamp <= gradesDeadline, "Grade submission deadline passed");
    require(!flag[_academicYear][_courseName], "Grades already posted");
    require(findPerson(courses[_academicYear][_courseName].instructors, msg.sender), "Only the instructor of the course");
    address _student;
    for (uint8 i=0; i<courses[_academicYear][_courseName].students.length; i++){
        _student=courses[_academicYear][_courseName].students[i];
        grades[_academicYear][_courseName][_student] = _grades[i];
    }
    flag[_academicYear][_courseName]=true;
}

function findPerson(address[] memory _array, address _person) internal pure returns (bool){
    for (uint i = 0; i < _array.length; i++) {
        if (_array[i] == _person) {
            return true;
        }
    }
    return false;
}
```

Οι φοιτητές μπορούν μέσω της εφαρμογής να δουν τον βαθμό που έχουν λάβει σε ένα μάθημα με δεδομένο το όνομα του μαθήματος και το ακαδημαϊκό έτος που διδάχθηκε.

```
function getGrade(uint16 _academicYear, string memory _courseName) external onlyStudent view returns (uint8){
    return grades[_academicYear][_courseName][msg.sender];
}
```

## Contract Deployment

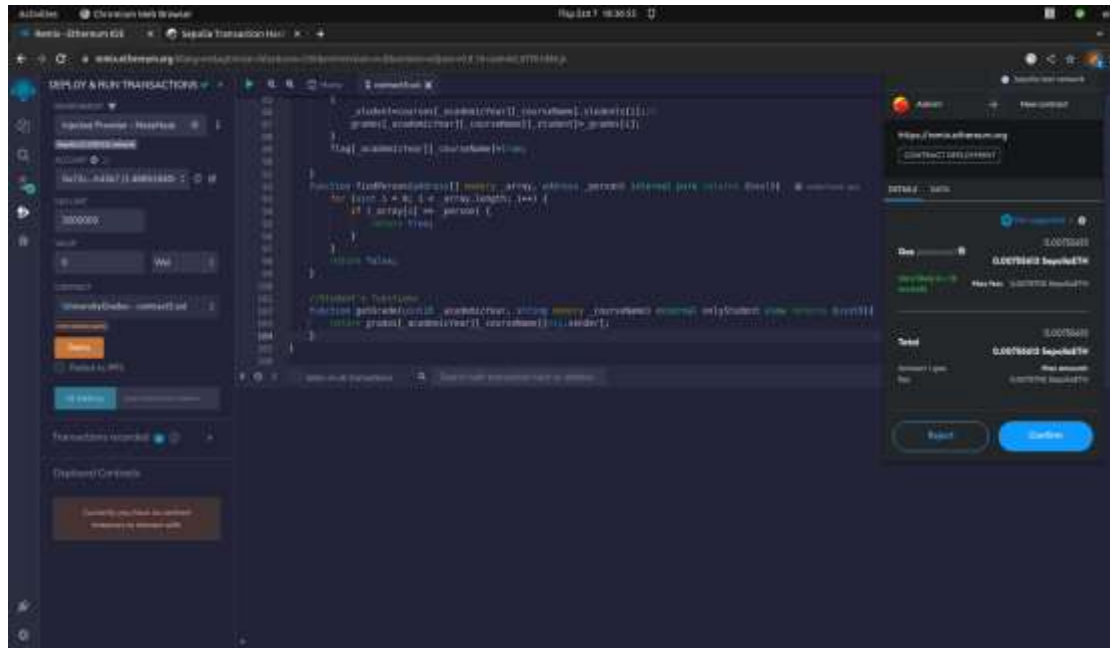


Figure 1: Επιβεβαίωση δημιουργίας συμβολαίου.

Το συμβόλαιο δημιουργήθηκε στο testnet sepolia με contract address:

0xa0C3dD703F54C7389c9C222EA25e0A256E5E8050 .

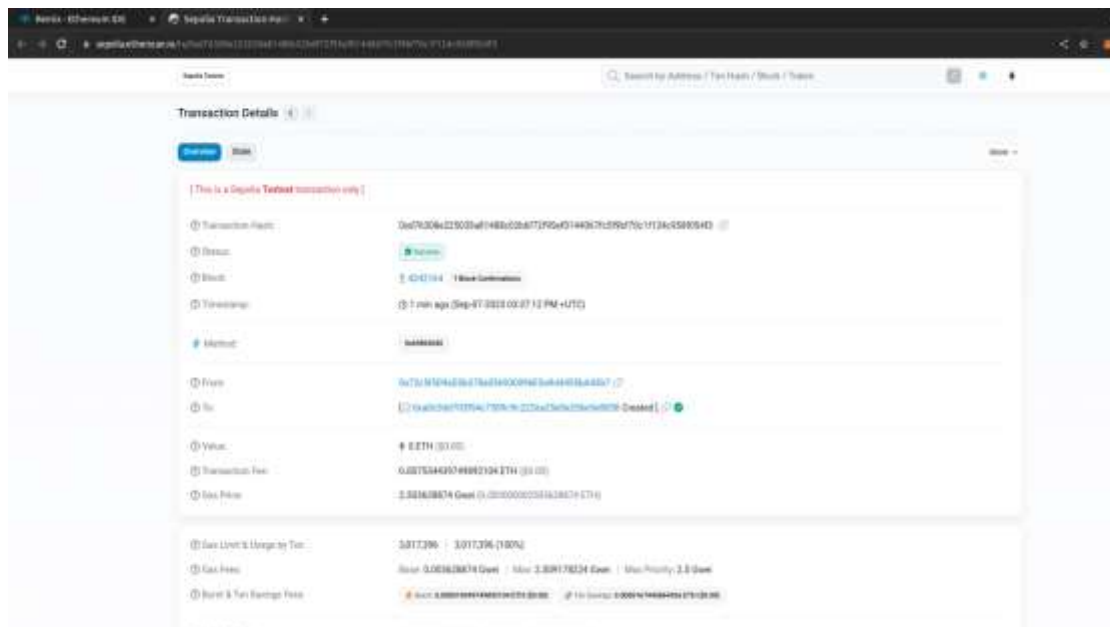


Figure 2: Το συμβόλαιο στο Etherscan.

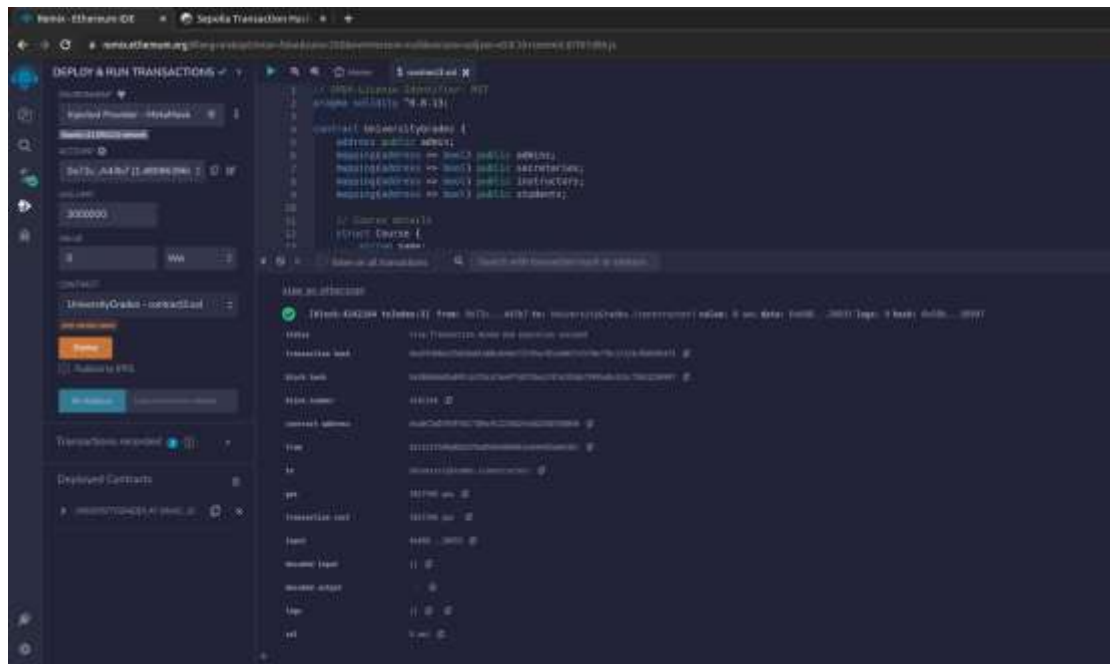


Figure 3: Δημιουργία συμβολαίου στο remixide.

## Παράδειγμα εκτέλεσης του κώδικα

Κατά τη δημιουργία του συμβολαίου ορίζεται ως αρχικός admin ο χρήστης που έκανε το deployment.

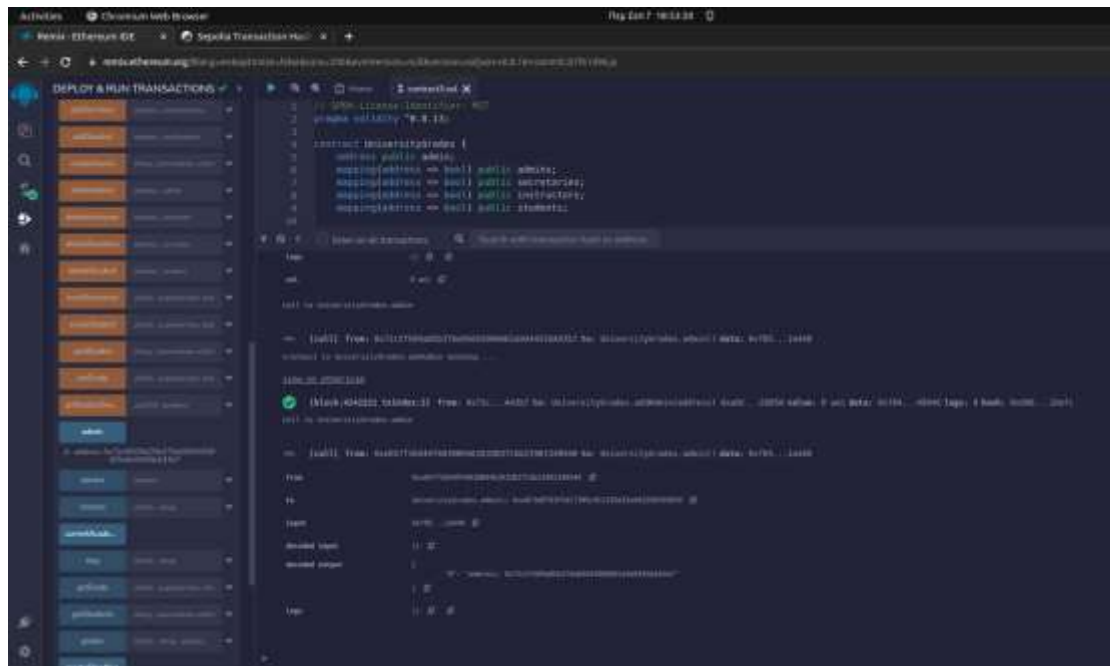



Figure 4: Αρχικός διαχειριστής

## Λειτουργίες διαχειριστή

### Εισαγωγή και διαγραφή διαχειριστών

Στην παρακάτω εικόνα μπορούμε να παρατηρήσουμε την ορθή λειτουργία της συνάρτησης `addAdmin`. Ο αρχικός admin με address `0xa0C3dD .... 8050` μπορεί να προσθέσει κάποιον διαχειριστή, πληκτρολογώντας την διεύθυνση του και πατώντας το κουμπί `addAdmin`. Στο παράδειγμα ο δεύτερος διαχειριστής που προστίθεται έχει την διεύθυνση `0x72c...3b7`.

Η πράσινη ένδειξη  στην κονσόλα δείχνει την επιτυχημένη ολοκλήρωση της συναλλαγής και πατώντας σε αυτήν βλέπουμε και τις λεπτομέρειές της. Μπορούμε επίσης να επιβεβαιώσουμε ότι ένα χρήστης είναι διαχειριστής πληκτρολογώντας την διεύθυνση του στο κουμπί `admins`. Όπου και η κονσόλα μας επιβεβαιώνει ότι είναι, εμφανίζοντας `true`.



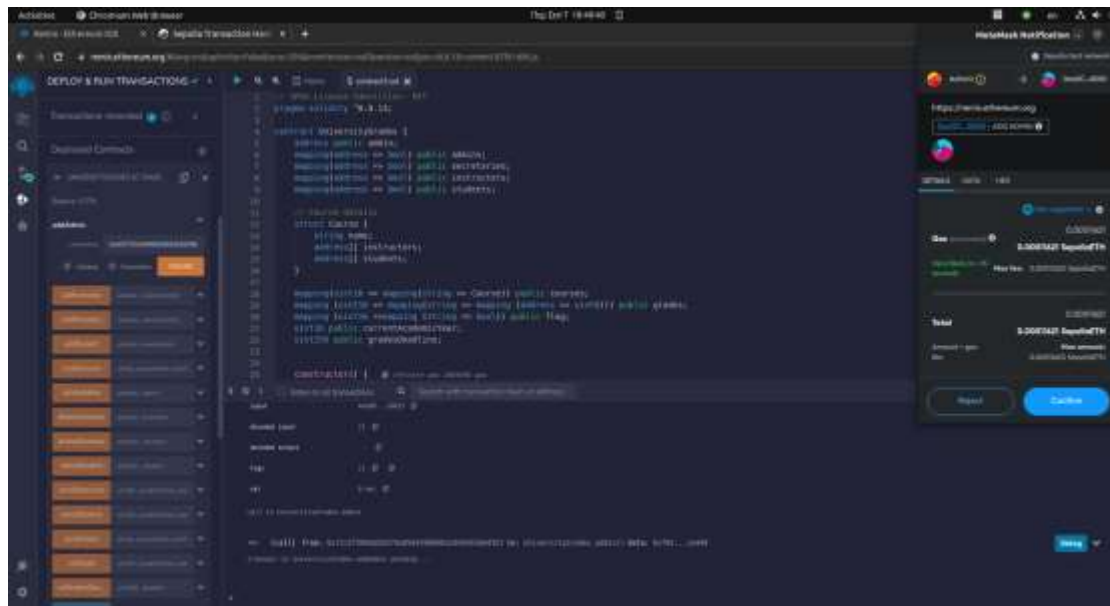


Figure 5: Επιβεβαίωση εισαγωγής admin.

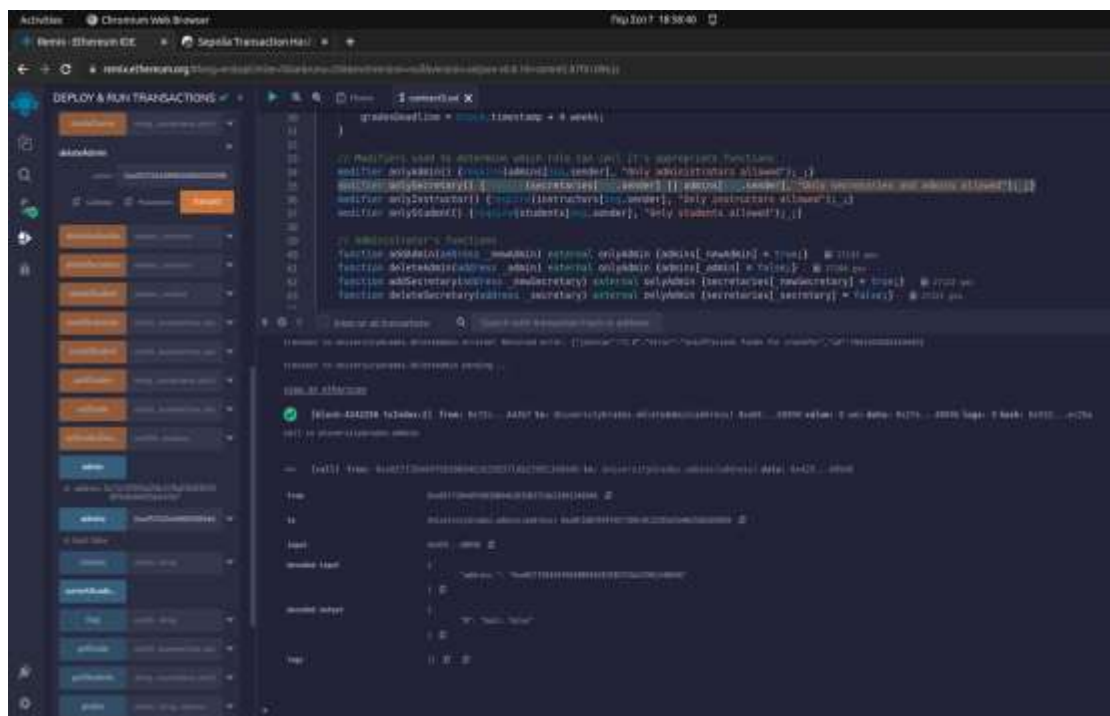


Figure 6: Διαγραφή διαχειριστή.

Παραπάνω διαγράφηκε ο προηγούμενος λογαριασμός και πλέον στην αναζήτηση για το εάν είναι admin, επιστρέφεται η μεταβλητή false.

## Εισαγωγή και διαγραφή γραμματείας

Αφού εισάγουμε το νέο διαχειριστή, μπορεί μετά αυτός να κάνει την εγγραφή γραμματείας όπως στο παράδειγμα που ακολουθεί με την διεύθυνση 0xD8f....D065

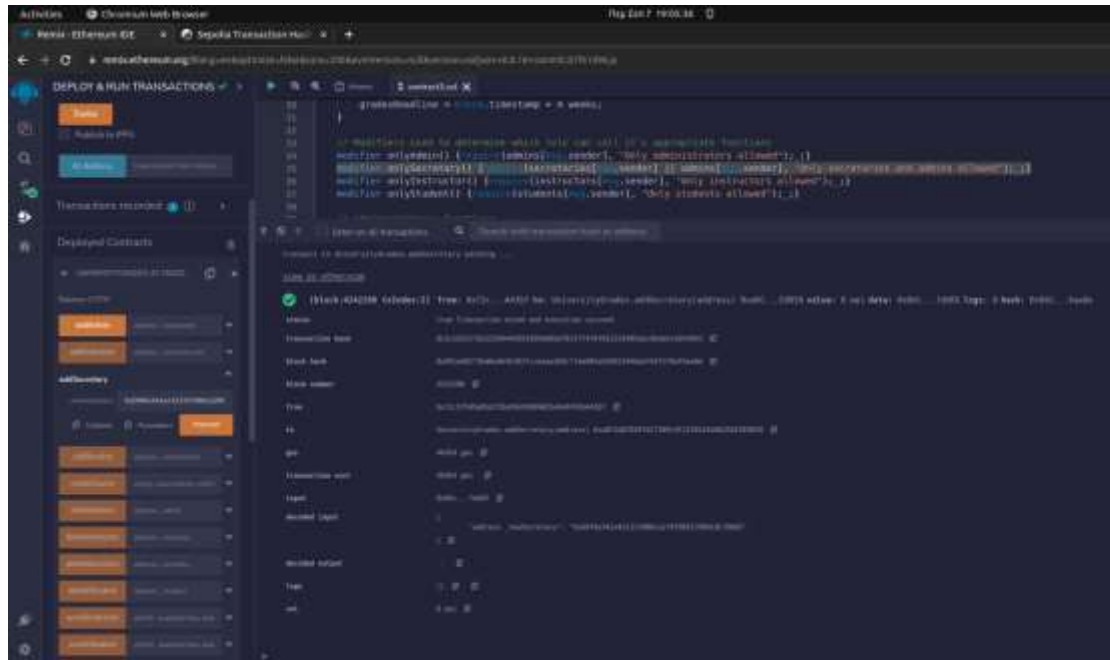


Figure 7: Εισαγωγή γραμματείας.

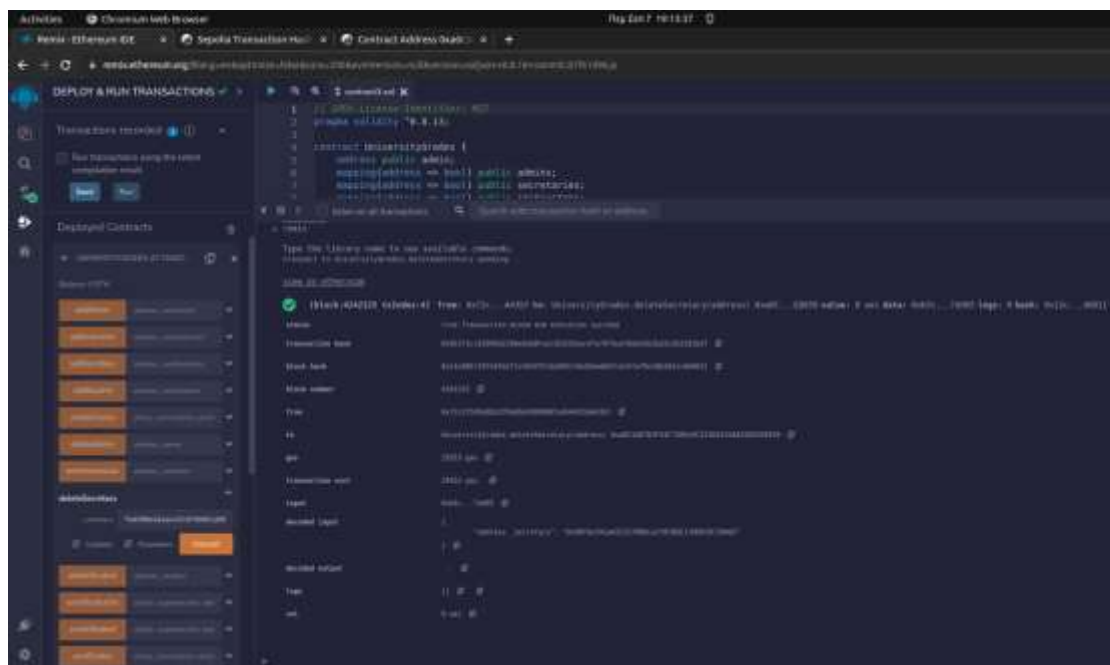


Figure 8: Διαγραφή γραμματείας.

Εκτέλεση άλλων λειτουργιών που υποστηρίζονται από την εφαρμογή

Ένας διαχειριστής έχει πλήρη δικαιώματα που του επιτρέπουν να εκτελέσει και οποιαδήποτε λειτουργία έχει δικαίωμα να εκτελέσει και μια γραμματεία. Για παράδειγμα την εισαγωγή ενός καθηγητή 0xf77E....41fD.

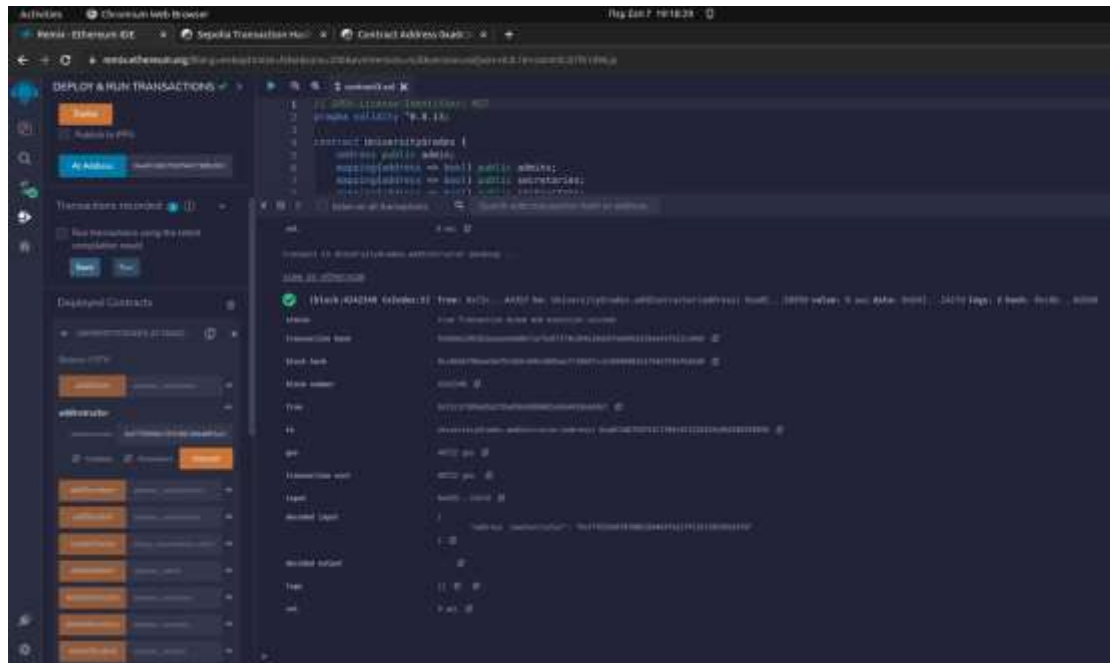


Figure 9: Εισαγωγή καθηγητή.

## Λειτουργίες γραμματείας

### Καθορισμός μαθημάτων

Η γραμματεία ενός τμήματος ορίζει ποια είναι τα μαθήματα του (προπτυχιακού) προγράμματος σπουδών του τμήματος για κάθε ακαδημαϊκό έτος.

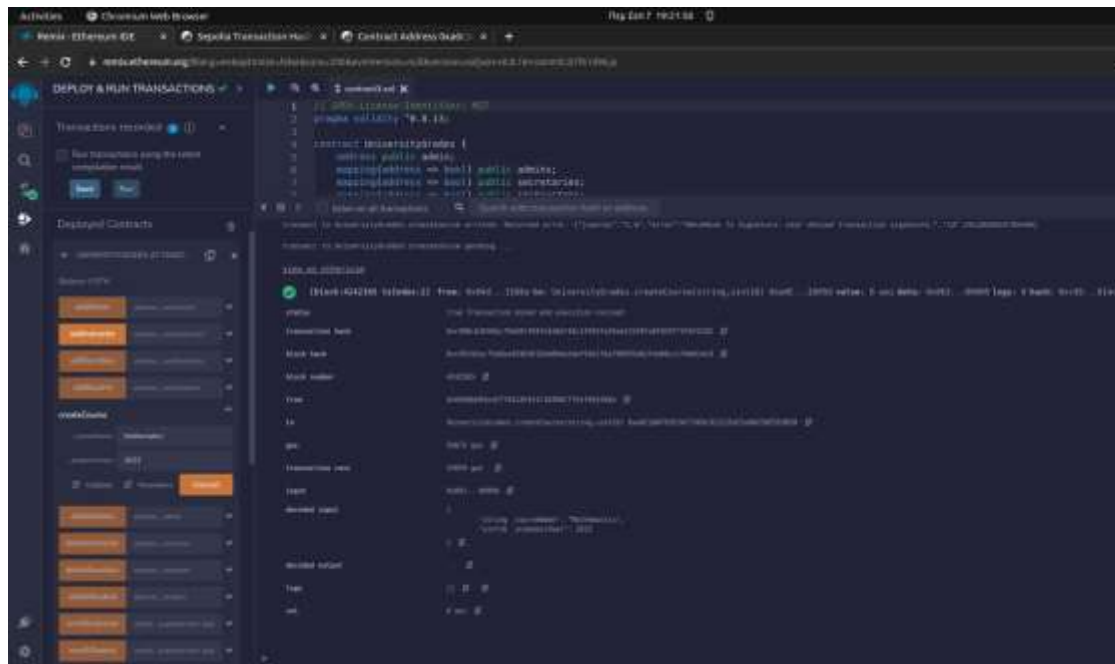
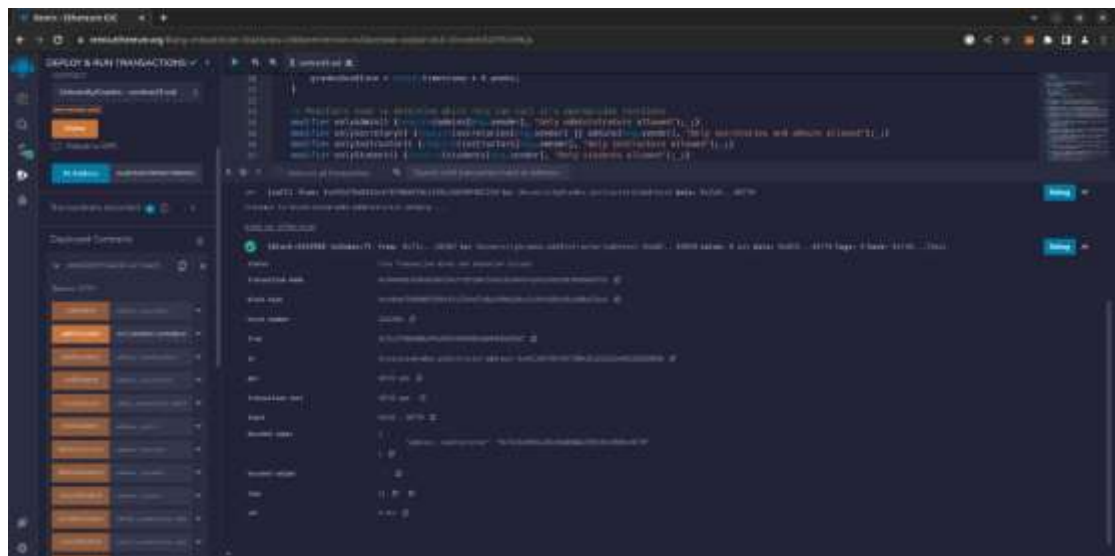


Figure 10: Εισαγωγή μαθήματος.

## Καθορισμό διδασκόντων



Εισαγωγή διδάσκοντα.

Παρακάτω καθορίζουμε τον καθηγητή 0xf77E....41fD για το μάθημα “Mathematics” του έτους 2023.

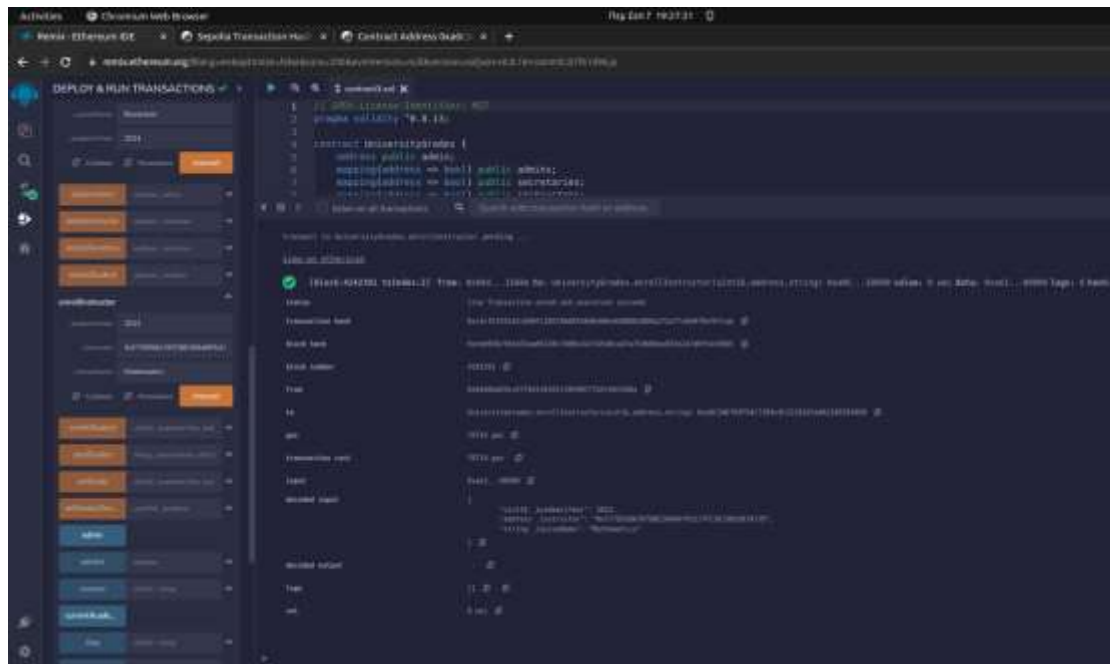


Figure 11: Καθορισμός διδάσκοντα.

Καθορισμός φοιτητών



Εισαγωγή φοιτητή

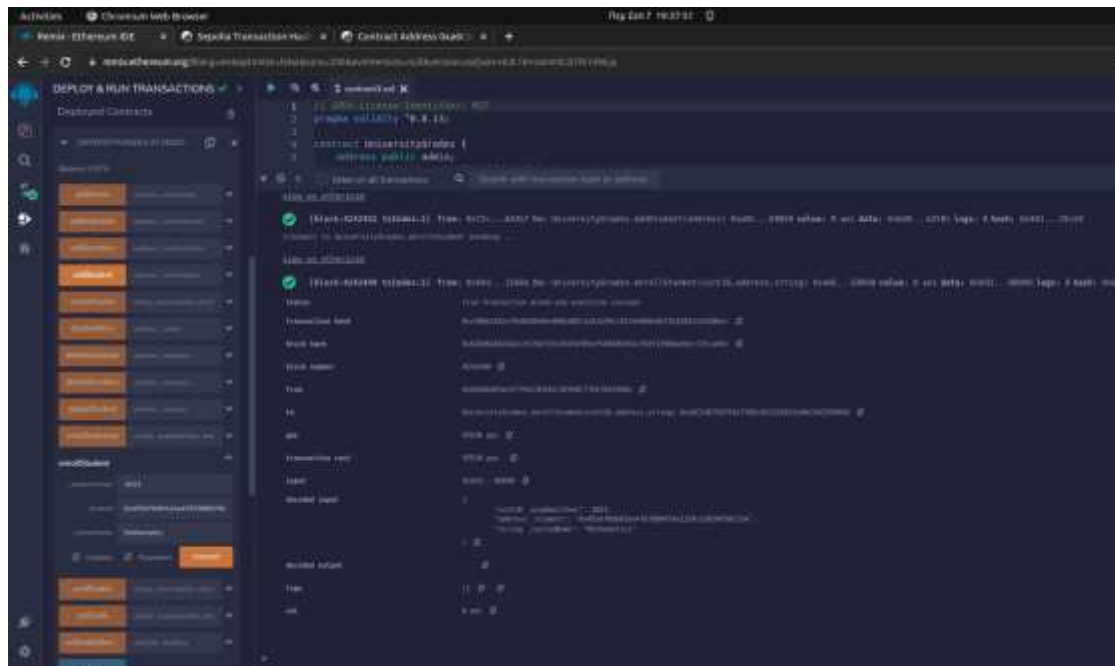


Figure 12: Εγγραφή φοιτητή σε μάθημα.

Εισαγωγή ή τροποποίηση βαθμολογιών.

Εισαγωγή βαθμού 6 στον φοιτητή 0x455d78dDA1be47670B8076e12E0c2d830F88C254 στο μάθημα Mathematics του έτους 2023.

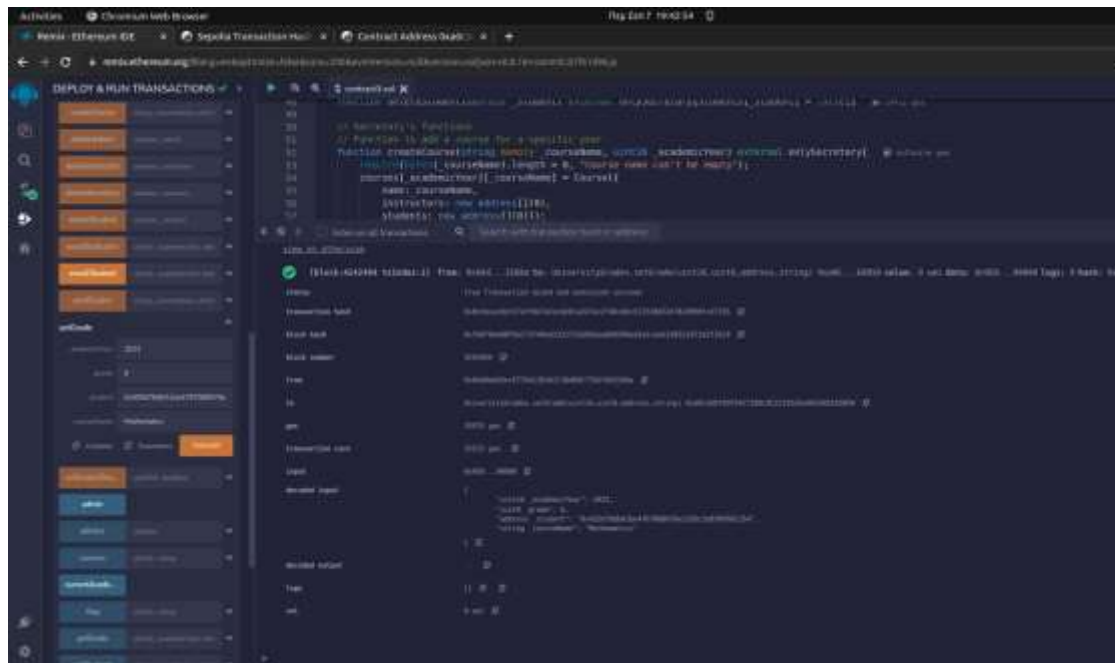


Figure 13: Εισαγωγή βαθμολογίας φοιτητή σε μάθημα.

## Καθορισμός προθεσμίας υποβολής των βαθμολογιών

Η γραμματεία επίσης ορίζει για κάθε εξεταστική περίοδο μια προθεσμία για την υποβολή των βαθμολογιών από τους διδάσκοντες του τμήματος. Η προθεσμία αυτή είναι κοινή για όλα τα μαθήματα και όπως προαναφέρθηκε η ημερομηνία λαμβάνει τη μορφή unix timestamp, μπορεί κάποιος να βρει την ημερομηνία που θέλει σε ένα εργαλείο μετατροπέα όπως το [unixtimestamp.com](https://unixtimestamp.com) ή το [epochconverter.com](https://epochconverter.com)

Yr: 2023 Mon: 10 Day: 18 Hr: 00 Min: 00 Sec: 00 Local time: Human date to Timestamp

Epoch timestamp: 1697576400  
Timestamp in milliseconds: 1697576400000  
Date and time (GMT): Tuesday, 17 October 2023 9:00:00 PM  
Date and time (your time zone): Τετάρτη, 18 Οκτωβρίου 2023 12:00:00 PM GMT+03:00

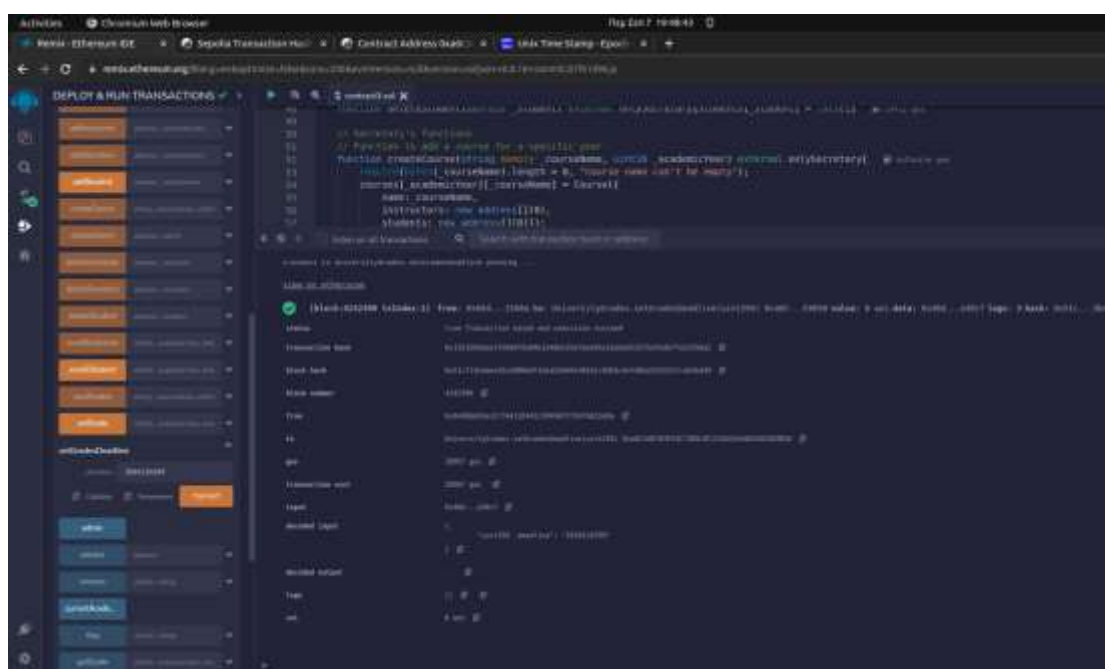


Figure 14: Ορισμός ημερομηνίας υποβολής βαθμολογιών από τους διδάσκοντες.

## Λειτουργίες καθηγητών

### Λήψη της λίστας φοιτητών

Ένα καθηγητή μπορεί να λάβει τη λίστα των φοιτητών, ενός μαθήματος, που δίδασκε ένα συγκεκριμένο ακαδημαϊκό έτος. Παρακάτω φαίνεται η λίστα των φοιτητών για το μάθημα Mathematics 2023.



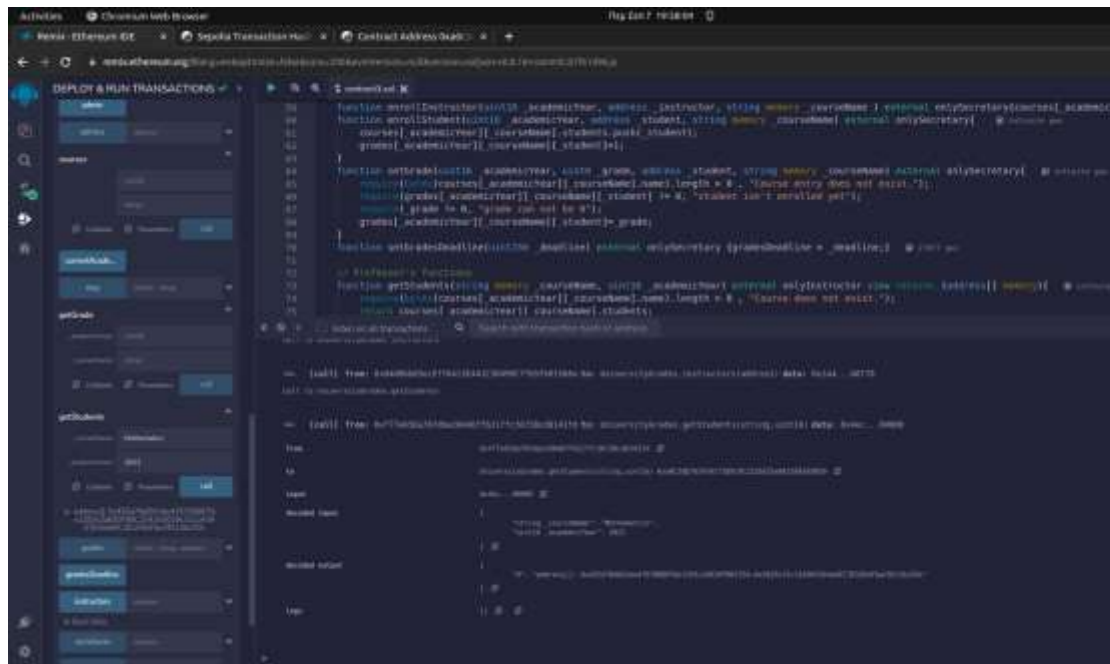


Figure 15: Λήψη λίστας φοιτητών μαθήματος.

### Ανάρτηση των βαθμολογιών

Ο καθηγητής αφού λάβει την λίστα των φοιτητών μπορεί με την αντίστοιχη κατάταξη να αναρτήσει τις βαθμολογίες για τους φοιτητές του μαθήματος του. Η βαθμολογία εισάγεται με μορφή πίνακα (π.χ. [6,8]) και η πρώτη βαθμολογία αντιστοιχεί στον πρώτο μαθητή της λίστας, η δεύτερη στον δεύτερο μαθητή κ.ο.κ.

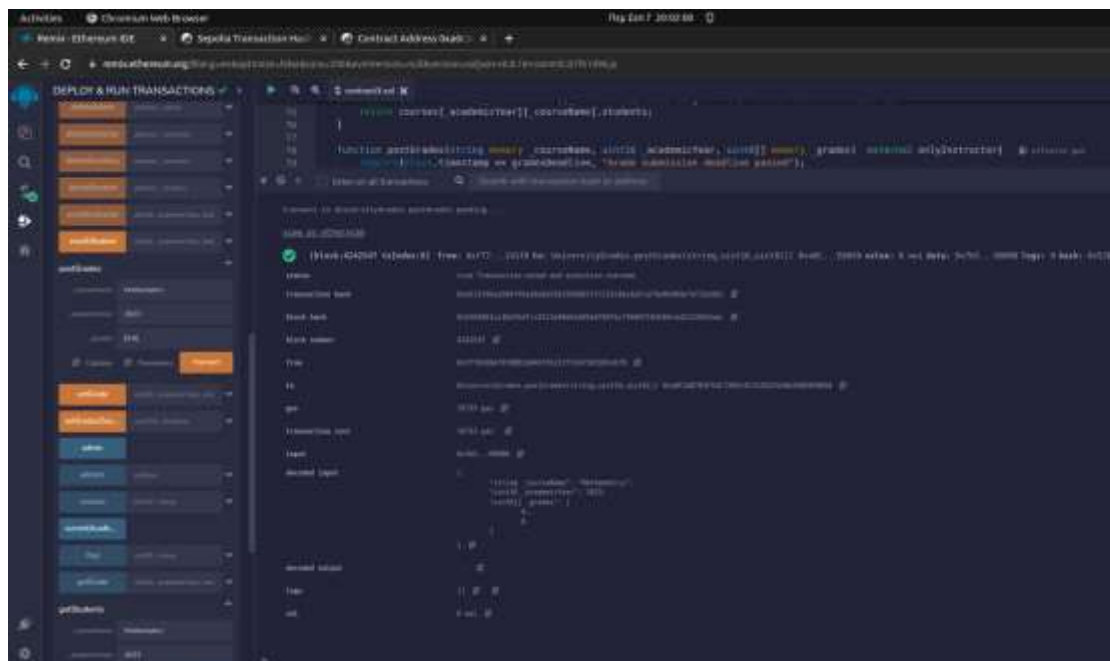
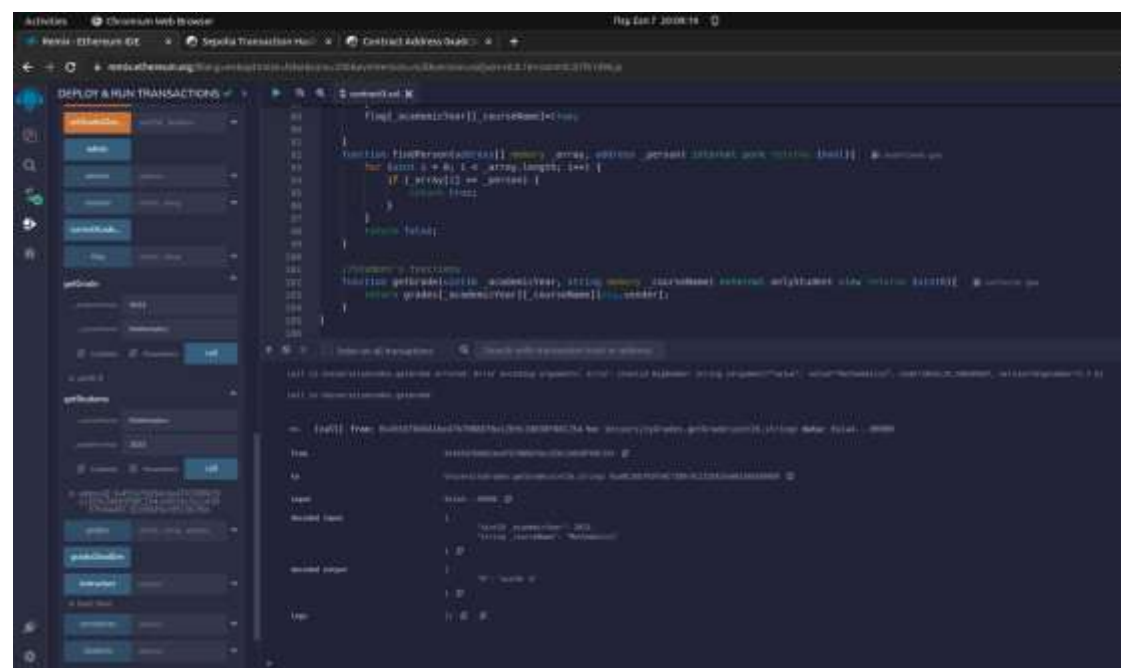


Figure 16: Ανάρτηση βαθμολογίας φοιτητών για μάθημα.

Η ανάρτηση των βαθμολογιών επιτρέπεται μόνο αν δεν έχει παρέλθει η προθεσμία υποβολής που έχει ορίσει η γραμματεία ή δεν έχει υποβληθεί ήδη από άλλον διδάσκοντα του μαθήματος, διαφορετικά δεν γίνεται η συναλλαγή όπως φαίνεται και στην παρακάτω εικόνα.



Λήψη βαθμού για ένα συγκεκριμένο μάθημα.



Λήψη της λίστας με τις βαθμολογίες για όλα τα μαθήματα.

## Σχολιασμός χρήσης gas

Αξίζει να αναφερθεί ότι η ανάπτυξη της εφαρμογής δεν ήταν μονοδιάστατη, αναπτύχθηκαν διάφορες εκδόσεις του κώδικα και καταλήξαμε στον παραπάνω με γνώμονα την ελαχιστοποίηση των υπολογιστικών πόρων, ώστε να περιοριστεί το απαιτούμενο gas. Οι διάφορες τεχνικές που χρησιμοποιήσαμε για την βελτιστοποίηση του κώδικα φαίνονται συνοπτικά παρακάτω:

- Χρήση κατάλληλων δομών για παράδειγμα mapping αντί για arrays για την αποθήκευση των διευθύνσεων.
- Χρήση view και pure συναρτήσεων (συναρτήσεις που δεν τροποποιούν την κατάσταση του συμβολαίου) που υποδεικνύουν στον Ethereum client ότι δεν απαιτούν εξόρυξη και μπορούν να εκτελεστούν τοπικά.
- Χρήση 'bytes' αντί για 'strings' , αφού γενικά οι μεταβλητές bytes είναι γενικά «φθηνότερες» από τις μεταβλητές συμβολοσειράς.