ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής

Εικόνα που περιέχει ορθογώνιο παραλληλόγραμμο, σχεδίαση

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εργασία Μαθήματος ***Σχεδίαση Αρχιτεκτονικών Ασφαλείας***

|  |  |
| --- | --- |
| Αρ. Άσκησης - Τίτλος Άσκησης | ***Application Threat Modelling*** |
| Όνομα φοιτητή - Αρ. Μητρώου | Ραυτόπουλος Μάριος – ΜΠΚΕΔ24034 |
| Ημερομηνία παράδοσης | 05/03/25 |

Εκφώνηση Εργασίας:

Θα πρέπει να επιλέξετε μια web εφαρμογή την οποία έχετε αναπτύξει σε κάποιο άλλο μάθημα (ή το eclass) ώστε να διενεργήσετε Application Threat Modelling λαμβάνοντας υπόψη σας:

* τις περαιτέρω οδηγίες και παραδείγματα από το site του OWASP (<https://www.owasp.org/index.php/Application_Threat_Modeling>)
* Για τα μέτρα προστασίας θα επιλέξετε τις βασικές κατηγορίες του OWASP ASVS (<https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project>, <https://owasp.org/www-chapter-frankfurt/assets/slides/43_OWASP_Frankfurt_Stammtisch_1.pdf>
* Αξιολογήστε τους κινδύνους με βάση τη μεθοδολογία DREAD Threat Modeling (https://threat-modeling.com/dread-threat-modeling/)

Το παραδοτέο θα είναι σε μορφή doc ή docx και θα περιλαμβάνει τουλάχιστον τα υπόλοιπα:

* τα βήματα που προτείνει η μεθοδολογία του OWASP (<https://www.owasp.org/index.php/Application_Threat_Modeling>)
* τα αποτελέσματα της αξιολόγησης.

Περιεχόμενα

[1. Εισαγωγή: 4](#_Toc191566515)

[1.1 Σκοπός Εγγράφου 4](#_Toc191566516)

[1.2 Πεδίο Εφαρμογής 4](#_Toc191566517)

[1.3 Μεθοδολογία 5](#_Toc191566518)

[2. Βήματα OWASP Application Threat Modeling 5](#_Toc191566519)

[2.1 Step1: Scope your Work 5](#_Toc191566520)

[2.1.1 Threat Model Information: 5](#_Toc191566521)

[2.1.2 External Dependencies: 6](#_Toc191566522)

[2.1.3 Entry Points: 6](#_Toc191566523)

[2.1.4 Assets: 7](#_Toc191566524)

[2.1.5 Trust Levels: 7](#_Toc191566525)

[2.1.6 Exit Points: 8](#_Toc191566526)

[2.1.7 Entities: 9](#_Toc191566527)

[2.1.8 Data Flow Diagrams: 11](#_Toc191566528)

[2.1.9 Data Flows Analysis 12](#_Toc191566529)

[2.2 Step 2: Determine Threats 14](#_Toc191566530)

[2.2.1 Threat Categorization 14](#_Toc191566531)

[2.2.2 Threat Identification 15](#_Toc191566532)

[2.2.3 Threat Ranking 19](#_Toc191566533)

[2.2.4 Ενδεικτικά Παραδείγματα Βαθμολόγησης DREAD 19](#_Toc191566534)

[2.2.5 Threat Ranking with DREAD 21](#_Toc191566535)

[2.3 Step 3: Determine Countermeasures and Mitigation 25](#_Toc191566536)

[2.3.1 Countermeasures by OWASP ASVS 25](#_Toc191566537)

[2.3.2 Mitigation Strategy 31](#_Toc191566538)

[2.4 Step 4: Assess your work 33](#_Toc191566539)

[3. Συμπεράσματα: 34](#_Toc191566540)

[4. Πηγές: 34](#_Toc191566541)

# 1. Εισαγωγή:

Σε αυτό το έγγραφο πραγματοποιείται Application Threat Modeling για μια e-learning Web εφαρμογή (π.χ. “Open eClass”). Ακολουθούνται κυρίως τις κατευθύνσεις του OWASP Application Threat Modeling σε συνδυασμό με το μοντέλο STRIDE για την κατηγοριοποίηση των απειλών και το μοντέλο DREAD για την αξιολόγηση και την ποσοτικοποίησή τους. Τέλος, υιοθετούνται βέλτιστες πρακτικές από το OWASP ASVS για τα απαραίτητα μέτρα προστασίας. Για την οπτική αναπαράσταση των στοιχείων και των αλληλεπιδράσεων της εφαρμογής χρησιμοποιείται το Microsoft Threat Modeling Tool (MTMT) 2016.

## 1.1 Σκοπός Εγγράφου

* Να παρουσιάσει τα βήματα του Threat Modeling (OWASP-based).
* Να αναδείξει τις κύριες απειλές με STRIDE.
* Να βαθμολογήσει (DREAD) τους κινδύνους.
* Να προτείνει αντίμετρα ασφάλειας βάσει OWASP ASVS.

## 1.2 Πεδίο Εφαρμογής

Εξετάζεται η e-learning πλατφόρμα (Open eClass) με δύο είδη χρηστών:

* Student (πρόσβαση σε υλικό, βαθμολογίες, ανέβασμα εργασιών).
* Teacher (ανέβασμα εκπαιδευτικού υλικού, βαθμολογίες, διαχείριση περιεχομένου).

## 1.3 Μεθοδολογία

Η μεθοδολογία που ακολουθείται βασίζεται στο OWASP Threat Modeling Process και περιλαμβάνει τα εξής βήματα:

* Βήμα 1: Scope your work – Αναγνώριση περιουσιακών στοιχείων, ροών δεδομένων, ορίων εμπιστοσύνης και επιφανειών επίθεσης.
* Βήμα 2: Determine Threats – Χρήση μιας δομημένης προσέγγισης για την ανακάλυψη ευπαθειών και την κατηγοριοποίησή τους με το μοντέλο STRIDE.
* Βήμα 3: Determine Countermeasures and Mitigation

Ορισμός μέτρων ασφαλείας και βέλτιστων πρακτικών για τη μείωση των κινδύνων.

* Βήμα 4: Assess your work– Εκτίμηση της αποτελεσματικότητας του Threat Model και βελτιστοποίηση των μετριασμών βάσει της ανάλυσης κινδύνου.

# 2. Βήματα OWASP Application Threat Modeling

## 2.1 Step1: Scope your Work

### 2.1.1 Threat Model Information:

Όνομα Εφαρμογής: Open eClass

Έκδοση: 1.0

Σκοπός: Παροχή διαδικτυακών εκπαιδευτικών υπηρεσιών σε φοιτητές και εκπαιδευτικούς. Οι φοιτητές μπορούν να εγγραφούν σε μαθήματα, να κατεβάζουν εκπαιδευτικό υλικό και να ανεβάζουν εργασίες. Οι εκπαιδευτικοί μπορούν να δημιουργούν/επεξεργάζονται μαθήματα, να ανεβάζουν σημειώσεις/διαφάνειες, να διαχειρίζονται βαθμολογίες και να βλέπουν αναφορές προόδου φοιτητών.

Κεντρικές Λειτουργίες:

Διαχείριση Μαθημάτων: Ο εκπαιδευτικός δημιουργεί/διαχειρίζεται το μάθημα, ανεβάζει εκπαιδευτικό υλικό.

Υποβολή Εργασιών: Ο φοιτητής ανεβάζει την εργασία του, και ο εκπαιδευτικός τη βαθμολογεί.

Ενημέρωση & Συνεργασία: Ανακοινώσεις, δημοσίευση βαθμών.

Ιδιοκτήτης εγγράφου: -

Συμμετέχοντες: -

Κριτής: -

### 2.1.2 External Dependencies:

HTTPS / TLS: Απαιτείται κρυπτογραφημένη επικοινωνία μεταξύ browser και server.

Web Server: Apache

Firewall: Προστατεύει το εσωτερικό δίκτυο, φιλτράρει απροσδόκητα αιτήματα.

Database (MySQL): Αποθήκευση δεδομένων χρηστών, μαθημάτων, βαθμολογιών.

File Storage: Χώρος αποθήκευσης των αρχείων (σημειώσεις, διαφάνειες, εργασίες).

### 2.1.3 Entry Points:

Τα Entry Points είναι τα «κανάλια» μέσω των οποίων εισέρχονται αιτήματα / δεδομένα στην εφαρμογή.

1. Login/Authentication Page

Δέχεται credentials (username/password) από τον χρήστη (Student/Teacher) για είσοδο στο σύστημα.

2. File Upload Endpoint

Ο φοιτητής ανεβάζει εργασία ή ο εκπαιδευτικός ανεβάζει εκπαιδευτικό υλικό. Αποτελεί κρίσιμο σημείο, γιατί εισέρχονται αρχεία στο σύστημα.

3. Course Management / Content Creation

Ο εκπαιδευτικός εισάγει δεδομένα (τίτλος μαθήματος, περιγραφή) στο σύστημα.

4. Password Reset (Forgot Password) Endpoint

Δέχεται email ή username για την εκκίνηση της διαδικασίας επαναφοράς κωδικού.

### 2.1.4 Assets:

1. User Credentials (Όνομα χρήστη / Κωδικός, διακριτά για φοιτητή ή εκπαιδευτικό).

2. Βαθμολογίες / Αξιολογήσεις (επικίνδυνο να τροποποιηθούν παράνομα).

3. Εκπαιδευτικό Υλικό (σημειώσεις, διαφάνειες, διαλέξεις) με πνευματικά δικαιώματα

4. Υποβληθείσες Εργασίες (Assignments) των φοιτητών, με πνευματικά δικαιώματα.

5. Φήμη / Αξιοπιστία Οργανισμού (αν η εφαρμογή παραβιαστεί ή σταματήσει να λειτουργεί, πλήττεται το κύρος του ιδρύματος).

### 2.1.5 Trust Levels:

1. Anonymous (Untrusted)
   * Ο χρήστης που δεν έχει καν λογαριασμό ή δεν έχει κάνει login. Έχει πρόσβαση μόνο σε ελάχιστες/δημόσιες σελίδες (π.χ. homepage, public info).
2. User with Invalid Credentials (Unsuccessful Login)
   * Κάποιος προσπαθεί να συνδεθεί (login) αλλά απέτυχε ή δεν έχει δικαιώματα. Θεωρείται αναξιόπιστος (επίπεδο “low trust”).
3. Student (Valid Credentials)
   * Ρόλος μεσαίου επιπέδου εμπιστοσύνης. Έχει πρόσβαση μόνο στις δικές του εργασίες, μπορεί να δει υλικό/βαθμολογία του, αλλά όχι να διαχειριστεί τα δεδομένα άλλων φοιτητών.
4. Teacher (Higher Trust)
   * Εκπαιδευτικός με δικαιώματα δημιουργίας/επεξεργασίας περιεχομένου (upload/download αρχείων, αλλαγή βαθμολογιών, προσθήκη ανακοινώσεων). Πιθανόν ευαίσθητος στόχος, καθώς χειρίζεται πολλά δεδομένα.
5. Admin / System Administrator (Highest Trust)
   * (Προαιρετικός ρόλος) Μπορεί να προσθέσει/αφαιρέσει χρήστες, να ρυθμίσει την πλατφόρμα, να δει logs. Έχει τα μέγιστα δικαιώματα, άρα αποτελεί “high value target”.

### 2.1.6 Exit Points:

Τα Exit Points είναι τα «κανάλια» μέσω των οποίων φεύγουν δεδομένα από το σύστημα προς τον χρήστη ή προς εξωτερικά συστήματα.

1. File Download Endpoint

Ο φοιτητής ή ο εκπαιδευτικός κατεβάζει αρχεία (PDF, διαφάνειες, κ.λπ.) ή βαθμολογημένες εργασίες.

Αποτελεί σημείο «εξόδου» δεδομένων από το σύστημα προς τον πελάτη (browser).

2. Logout Endpoint

Παρότι δεν «επιστρέφει» μεγάλο όγκο δεδομένων, θεωρείται τεχνικά σημείο «εξόδου» από την authenticated session. Μπορεί να τερματίζει ή να καθαρίζει session tokens.

### 2.1.7 Entities:

1. Student (Χρήστης Φοιτητής)
   * Μπορεί να κάνει login, να εγγραφεί σε μαθήματα, να βλέπει ανακοινώσεις, να ανεβάζει εργασίες, να βλέπει μόνο τα δικά του στοιχεία.
2. Teacher (Χρήστης Εκπαιδευτικός)
   * Μπορεί να δημιουργήσει/επεξεργαστεί μαθήματα, να ανεβάζει εκπαιδευτικό υλικό, να βαθμολογεί εργασίες, να προβάλει στατιστικά επίδοσης.
3. Web Client / Browser
   * Το λογισμικό (frontend) που τρέχει στον υπολογιστή/κινητό του χρήστη. Στέλνει αιτήματα στο server (HTTPS), λαμβάνει HTML/JSON/CSS.
4. Web Server
   * Η βασική εφαρμογή (business logic, authorization) που διαχειρίζεται τα αιτήματα χρηστών, εφαρμόζει κανόνες ασφάλειας (session management κ.ο.κ.) και συνδέεται στη βάση δεδομένων.
5. Database
   * Αποθηκεύει δεδομένα χρηστών, στοιχεία μαθημάτων, βαθμολογίες.
6. File Storage / File System
   * Χώρος αποθήκευσης των αρχείων (pdfs, docs, zip) που ανεβάζουν εκπαιδευτικοί ή φοιτητές (σημειώσεις, εργασίες κ.λπ.).
7. Optional: Admin (Διαχειριστής Συστήματος)
   * Πρόσωπο ή ομάδα με δικαιώματα ρύθμισης υποδομών (εγκατάσταση ενημερώσεων, ρυθμίσεις server, σύνδεση με SMTP). Άμεσα υπεύθυνος για ζητήματα ασφάλειας, εφεδρικών αντιγράφων (backups), logs κ.λπ.
8. Όρια ασφαλείας (Security Boundaries):

Boundary #1: Ανάμεσα στο Public Internet και τον Web Client / Web Server

* + Ο χρήστης βρίσκεται σε ένα απρόβλεπτο/μη ελεγχόμενο περιβάλλον (διαδίκτυο).
  + Τα αιτήματα φτάνουν στον Web Server μέσω HTTPS.
  + Εφόσον βρισκόμαστε στο Διαδίκτυο, τα δεδομένα πρέπει να κρυπτογραφούνται (TLS) για να αποφευχθεί υποκλοπή (network sniffing).
* Κίνδυνοι:
  + Κακόβουλοι χρήστες μπορούν να στέλνουν «πειραγμένα» αιτήματα (SQLi, XSS payloads, τεράστια αρχεία για DoS, κ.λπ.).
  + Επιθέσεις credential stuffing, brute force attack.

Boundary #2: Ανάμεσα στον Web Server και τα Εσωτερικά Συστήματα (Database, File System)

* + Ο Web Server επικοινωνεί μέσω ιδιωτικού δικτύου με τη Database και το File System.
  + Ο ίδιος ο τελικός χρήστης (Student/Teacher) *δεν* έχει απευθείας πρόσβαση στη DB ή στο filesystem.
  + Ο Web Server λειτουργεί ως «μεσολαβητής» (mediator): παίρνει τα αιτήματα από τον browser, κάνει έλεγχο πρόσβασης (authorization), επικοινωνεί με τη βάση για να διαβάσει/γράψει δεδομένα ή με το
  + filesystem για ανέβασμα/κατέβασμα αρχείων.
* Κίνδυνοι:
  + Αν ο Web Server παραβιαστεί (π.χ. μέσω RCE ή credentials leak), μπορεί να αποκτήσει κάποιος πρόσβαση στη βάση ή στα αρχεία.
  + Αν δεν υπάρχουν κατάλληλοι έλεγχοι (privileges, firewalls σε επίπεδο εσωτερικού δικτύου), ένας επιτιθέμενος μπορεί να μιμηθεί τον Web Server και να κάνει κακόβουλα queries στη DB.

### 2.1.8 Data Flow Diagrams:

A diagram of a computer system

AI-generated content may be incorrect.

### 2.1.9 Data Flows Analysis

Login / Authentication:

User (Student/Teacher) ανοίγει το login page.

Πληκτρολογεί username/password → γίνεται HTTPS POST αίτημα στον Web Server.

Web Server (Application Logic) ελέγχει τα credentials στη Database (π.χ. SELECT \* FROM users WHERE username=?).

Αν η αυθεντικοποίηση πετύχει, ο Web Server δημιουργεί session/token και το επιστρέφει στον browser. Αυτό το token θα χρησιμοποιείται σε επόμενα αιτήματα.

Σημείο Ελέγχου: Επιβεβαίωση user role (Student/Teacher), ρύθμιση session cookies (httpOnly, secure flag), rate limiting για brute force.

Ανέβασμα Αρχείου (Upload):

Ο Teacher (ή Student, π.χ. για εργασία) επιλέγει αρχείο και στέλνει POST αίτημα (multipart form data) στο endpoint POST /upload.

Ο Web Server ελέγχει αν ο χρήστης έχει δικαιώματα να ανεβάσει (π.χ. Teacher role) και εφόσον είναι έγκυρος, αποθηκεύει το αρχείο στο File System (συνήθως σε φάκελο /var/www/courses/ κ.λπ.).

Καταχωρεί στη Database (π.χ. πίνακας files) μεταδεδομένα (όνομα μαθήματος, ποιος χρήστης το ανέβασε, όνομα αρχείου).

Επιστρέφει επιτυχία στο browser (JSON ή redirect) και εμφανίζει μήνυμα «Το αρχείο ανέβηκε επιτυχώς».

Σημείο Ελέγχου: Input validation (π.χ. περιορισμός μεγέθους αρχείου), έλεγχος τύπου αρχείου, antivirus scan (πιθανώς), proper permissions στο filesystem.

Κατέβασμα Αρχείου (Download):

Ο Student/Teacher κάνει κλικ σε σύνδεσμο (π.χ. /files/12345) για να κατεβάσει ένα συγκεκριμένο αρχείο.

Ο Web Server λαμβάνει το αίτημα και ελέγχει ποιος ζητάει το αρχείο (session token, role).

Αν ο χρήστης έχει δικαίωμα, ο server διαβάζει το αντίστοιχο αρχείο από το File System.

Ο Server επιστρέφει το αρχείο μέσω HTTPS στον browser.

Σημείο Ελέγχου: Authorization checks (μήπως ένας Student προσπαθεί να κατεβάσει αρχείο άλλου μαθήματος/φοιτητή;), σωστό content-type κ.λπ.

Password Reset:

Χρήστης που ξέχασε κωδικό μπαίνει στο /forgot\_password. Στέλνει email/username.

Ο Web Server βρίσκει τον λογαριασμό στη DB, δημιουργεί ένα reset token και το στέλνει με Email (Exit Point) στον χρήστη.

Ο χρήστης κάνει κλικ στον σύνδεσμο (παράμετρο token=), ο server ελέγχει αν είναι έγκυρο + δεν έχει λήξει.

Ο χρήστης επιλέγει νέο κωδικό, ο server κάνει UPDATE στον πίνακα users (πεδίο hashed\_password).

Σημείο Ελέγχου: Προστασία του token (cryptographically secure), λήξη token μετά από Χ ώρα, hashing νέου password, μη διαρροή email σε logs κ.ο.κ.

## 2.2 Step 2: Determine Threats

### 2.2.1 Threat Categorization

Για τον εντοπισμό και την κατηγοριοποίηση των απειλών χρησιμοποιείται το μοντέλο STRIDE ( Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privileges)

STRIDE Threat List:

Spoofing: Δράση απειλής με στόχο την πρόσβαση και τη χρήση των διαπιστευτηρίων ενός άλλου χρήστη, όπως το όνομα χρήστη και ο κωδικός πρόσβασης. (Authentication Threats)

Tampering: Δράση απειλής που αποσκοπεί στην κακόβουλη αλλαγή ή τροποποίηση μόνιμων δεδομένων, όπως εγγραφές σε μια βάση δεδομένων, και η αλλοίωση δεδομένων κατά τη διαμετακόμιση μεταξύ δύο υπολογιστών μέσω ανοικτού δικτύου, όπως το Διαδίκτυο. (Integrity Threats)

Repudiation: Δράση απειλής που αποσκοπεί στην εκτέλεση απαγορευμένων λειτουργιών σε ένα σύστημα που δεν έχει τη δυνατότητα να τις εντοπίσει. (Non-Repudiation Threats)

Information Disclosure: ενέργεια απειλής που αποσκοπεί στην ανάγνωση αρχείου στο οποίο δεν έχει χορηγηθεί πρόσβαση ή στην ανάγνωση δεδομένων κατά τη μεταφορά. (Confidentiality Threats)

Denial of service: Δράση απειλής που επιχειρεί να αρνηθεί την πρόσβαση σε έγκυρους χρήστες, π.χ. καθιστώντας προσωρινά μη διαθέσιμο ή άχρηστο έναν διακομιστή ιστού. (Availability Threats)

Elevation of privilege: Δράση απειλής που αποσκοπεί στην απόκτηση προνομιακής πρόσβασης σε πόρους για την απόκτηση μη εξουσιοδοτημένης πρόσβασης σε πληροφορίες ή για την παραβίαση ενός συστήματος. (Authorization Threats)

### 2.2.2 Threat Identification

Μέσω του analysis view του Microsoft Threat Modelling Tool παρατηρούνται τα παρακάτω:

A screenshot of a computer

AI-generated content may be incorrect.

Πιο συγκεκριμένα παρατηρούνται 43 απειλές και πιο συγκεκριμένα ανά κατηγορία:

**S**poofing: 8

**T**ampering: 9

**R**epudiation: 4

**I**nformation Disclosure: 2

**D**enial Of Service: 8

**E**levation Of Privilege: 12

**STRIDE: Αναλυτική Παρουσίαση Απειλών**

**1. Spoofing (Πλαστοπροσωπία)**

1. **Teacher Spoofing**
   * **Περιγραφή**: Ένας κακόβουλος χρήστης (π.χ. φοιτητής) προσπαθεί να αποκτήσει τα διαπιστευτήρια (username/password) ή το session token κάποιου καθηγητή. Με αυτόν τον τρόπο αποκτά δικαιώματα “Teacher” και μπορεί να τροποποιήσει βαθμολογίες, να ανεβάσει/διαγράψει υλικό κ.λπ.
   * **Πιθανά Σενάρια**:
     + Κλοπή session cookie (XSS σε forum, μη ασφαλή δίκτυα Wi-Fi).
     + Brute force ή password guessing σε καθηγητές που επιλέγουν αδύναμους κωδικούς.
   * **Συνέπειες**: Παραβίαση εμπιστευτικότητας (δείτε δεδομένα μαθητών), ακεραιότητας (αλλοίωση βαθμολογιών/υλικού).
2. **Student Spoofing**
   * **Περιγραφή**: Παρόμοια κατάσταση, αλλά ο στόχος είναι ένας φοιτητής με σκοπό να κλέψει τα στοιχεία κάποιου άλλου φοιτητή για να δει βαθμούς ή εργασίες που δεν του ανήκουν, ή να υποβάλλει εργασία εκ μέρους άλλου.
   * **Πιθανά Σενάρια**:
     + “Credential stuffing” (δοκιμή κωδικών που έχουν διαρρεύσει).
     + Απλή εκτίμηση/σπάσιμο κωδικού (αν ο χρήστης επέλεξε απλό κωδικό).

**2. Tampering (Αλλοίωση/Παραποίηση Δεδομένων)**

1. **SQL Injection**
   * **Περιγραφή**: Μέσω φόρμας (π.χ. αναζήτηση μαθήματος ή login), ο επιτιθέμενος στέλνει ειδικά crafted SQL queries. Αν δεν γίνεται proper parameterization, μπορεί να τροποποιήσει/διαγράψει εγγραφές (π.χ. βαθμολογίες).
   * **Συνέπειες**: Καταστροφή ή αλλοίωση βαθμολογιών, παραβίαση εμπιστευτικότητας (π.χ. άντληση λιστών χρηστών).
2. **Malicious File Upload**
   * **Περιγραφή**: Ο επιτιθέμενος ανεβάζει κακόβουλο script (π.χ. PHP backdoor αν το σύστημα επιτρέπει .php αρχεία) ή “Trojan” PDF. Εφόσον εκτελεστεί στον server ή σε σταθμούς εργασίας άλλων, μπορεί να προκαλέσει αλλοίωση συστήματος ή μόλυνση.
   * **Συνέπειες**: Παραβίαση ακεραιότητας αρχείων συστήματος, διάδοση malware, πιθανή απομακρυσμένη εκτέλεση κώδικα.

**3. Repudiation (Άρνηση Πράξεων)**

1. **Insufficient Logging & Auditing**
   * **Περιγραφή**: Το σύστημα δεν κρατά επαρκή logs για αλλαγές βαθμολογιών, ανεβάσματα αρχείων κ.λπ. Κάποιος μπορεί να αλλάξει δεδομένα ή να αφαιρέσει υλικό και μετά να ισχυριστεί ότι “δεν το έκανε”, χωρίς δυνατότητα εντοπισμού.
   * **Συνέπειες**: Αδυναμία απόδοσης ευθύνης, δεν μπορούμε να αποδείξουμε πότε και από ποιον έγιναν οι κρίσιμες ενέργειες.

**4. Information Disclosure (Διαρροή Πληροφοριών)**

1. **Unauthorized File Access**
   * **Περιγραφή**: Ένας φοιτητής αποκτά πρόσβαση σε αρχεία που ανήκουν σε άλλο φοιτητή ή σε υλικό που δεν θα έπρεπε να δει (π.χ. λύσεις ασκήσεων, βαθμοί).
   * **Συνέπειες**: Παραβίαση εμπιστευτικότητας, πιθανή διαρροή προσωπικών δεδομένων (GDPR issues).

**5. Denial of Service (Άρνηση Εξυπηρέτησης)**

1. **Large File Upload Flood**
   * **Περιγραφή**: Ο επιτιθέμενος ανεβάζει πολλαπλά τεράστια αρχεία (π.χ. αρχεία-«σκουπίδια» gigabytes) γονατίζοντας το χώρο αποθήκευσης ή τον ίδιο τον server.
   * **Συνέπειες**: Το σύστημα γίνεται μη διαθέσιμο (γεμάτος δίσκος, κρεμάει ο server).
2. **Login Brute Force / High-Rate Requests**
   * **Περιγραφή**: Μαζικές αιτήσεις στο endpoint /login ή σε άλλα κρίσιμα endpoints, εξαντλώντας τους πόρους ή καθιστώντας τον server πολύ αργό.
   * **Συνέπειες**: Νόμιμοι χρήστες δεν μπορούν να κάνουν login, επίθεση άρνησης υπηρεσίας.

**6. Elevation of Privilege (Αναβάθμιση Προνομίων)**

1. **Student → Teacher Escalation**
   * **Περιγραφή**: Μέσω κάποιας ευπάθειας (Insecure Direct Object References, ελλιπής έλεγχος ρόλου), ένας απλός φοιτητής αποκτά πρόσβαση σε λειτουργίες του καθηγητή (π.χ. μεταβολή βαθμολογιών).
   * **Συνέπειες**: Παραβίαση ακεραιότητας δεδομένων, διάλυση αξιοπιστίας του συστήματος.

### 2.2.3 Threat Ranking

Η αξιολόγηση των απειλών πραγματοποιείται με το μοντέλο DREAD(Damage Potential, Reproducibility, Exploitability, Affected Users, Discoverability)

Για κάθε κατηγορία έχει βαθμολογία 0 έως 10, οπότε μέγιστη βαθμολογία τα 50.

Πιο συγκεκριμένα αναφέρεται στις παρακάτω κατηγορίες:

* Damage: The total damage (or impact) that a threat can cause.
* Reproducibility: The ease at which an attack can occur (or be replicated).
* Exploitability: How likely or easily the weakness or threat can be exploited.
* Affected Users: The number of (end) users that could be affected by a threat being exploited.
* Discoverability: How likely a threat will be discovered by an attacker.

### 2.2.4 Ενδεικτικά Παραδείγματα Βαθμολόγησης DREAD

**1. Damage Potential (D)**

* **0**: Καμία ζημιά (π.χ. ανούσια ή εικονική επίθεση).
* **5**: Απώλεια/confidential data με *μέτριο* αντίκτυπο (π.χ. μη ευαίσθητα δεδομένα).
* **8**: Παραβίαση προσωπικών δεδομένων (non-critical personal data).
* **9**: Παραβίαση ευαίσθητων ή διοικητικών δεδομένων.
* **10**: Ολοσχερής καταστροφή του συστήματος ή απώλεια *πολύ* ευαίσθητων δεδομένων (π.χ. κρυφές πληροφορίες, οικονομικά, GDPR κ.λπ.).

**2. Reproducibility (R)**

* **0**: Σχεδόν αδύνατο να αναπαραχθεί (χρειάζεται πολύ εξεζητημένο περιβάλλον).
* **5**: Απαιτείται *κάποια* πολυπλοκότητα (συνδυασμός εργαλείων, συνθήκες).
* **7.5**: Σχετικά εύκολο, υπάρχουν διαθέσιμα εργαλεία (script, scanners).
* **10**: Πολύ εύκολο/απλό, οποιοσδήποτε μπορεί να το κάνει με ελάχιστες γνώσεις (π.χ. μια απλή φόρμα επιτρέπει injection).

**3. Exploitability (E)**

* **2.5**: Χρειάζεται *εξειδικευμένες* γνώσεις και τεχνικές ικανότητες (π.χ. reverse engineering).
* **5**: Υπάρχουν public εργαλεία/exploits, αλλά απαιτούνται κάποιοι βασικοί χειρισμοί.
* **9**: Υπάρχουν ευρέως διαθέσιμα scripts ή tutorials, εύκολα στη χρήση.
* **10**: Μπορεί να γίνει *ακόμα και από browser* χωρίς επιπλέον εργαλεία (π.χ. copy-paste σε μια URL).

**4. Affected Users (A)**

* **0**: Κανένας χρήστης δεν επηρεάζεται (θεωρητικό exploit).
* **2.5**: Μόνο ένας χρήστης (στοχευμένα).
* **6**: Μια μικρή ομάδα χρηστών (subset).
* **8**: Αφορά *πιο κρίσιμους* χρήστες (admin, teacher), ή αρκετούς «υψηλής αξίας» λογαριασμούς.
* **10**: Όλοι οι χρήστες ή όλη η υποδομή μπορούν να επηρεαστούν.

**5. Discoverability (D’)**

* **0**: Εξαιρετικά δύσκολη ή άγνωστη ευπάθεια (πρέπει να είσαι “insider”).
* **5**: Χρειάζεται *ψάξιμο*, αλλά όχι αδύνατον· με δοκιμές ή ειδικά εργαλεία.
* **8**: Η ευπάθεια είναι γνωστή (public) ή υπάρχει POC/αναφορά ήδη στο internet.
* **10**: Προφανής ευπάθεια, πολύ εμφανής (π.χ. ένα πεδίο που δεν έχει καν input validation).

**Overall DREAD Rating**

Low for overall ratings between 1-10

Medium for overall ratings between 11-24

High for overall ratings between 25-39

Critical for overall ratings between 40-50

**OVERALL DREAD SCORE == Damage + Reproducibility + Exploitability + Affected Users + Discoverability**

### 2.2.5 Threat Ranking with DREAD

Spoofing: Teacher Spoofing

* Damage (D): 9  
  (μπορεί να τροποποιήσει βαθμολογίες/υλικό)
* Reproducibility (R): 7.5  
  (αρκετά εφικτό με brute force, κλοπή session κ.λπ.)
* Exploitability (E): 5  
  (χρειάζεται κάποια εργαλεία/γνώσεις)
* Affected Users (A): 10  
  (επηρεάζει *όλα* τα μαθήματα/φοιτητές, γιατί Teacher έχει ευρύτερη πρόσβαση)
* Discoverability (D’): 10  
  (καθόλου κρυφό, εύκολο να δοκιμάσει κάποιος brute force / session hijack)

Σύνολο: 9 + 7.5 + 5 + 10 + 10 = 41.5  
Σοβαρότητα: Critical

Tampering: SQL Injection

* Damage (D): 10  
  (μπορεί να καταστρέψει/διαρρεύσει ολόκληρη τη βάση)
* Reproducibility (R): 7.5  
  (ευρέως γνωστές τεχνικές, αρκετά “standard” exploit)
* Exploitability (E): 5  
  (χρειάζεται κάποια γνώση, αλλά υπάρχουν πολλά tools)
* Affected Users (A): 10  
  (πρακτικά *όλοι* οι χρήστες επηρεάζονται, καθώς η DB περιέχει τα πάντα)
* Discoverability (D’): 8  
  (συνήθως υπάρχουν scanners και δημόσιες λίστες injection payloads)

Σύνολο: 10 + 7.5 + 5 + 10 + 8 = 40.5  
Σοβαρότητα: Critical

Repudiation: Insufficient Logging

* Damage (D): 10  
  (η επίθεση μπορεί να καταστρέψει/αλλοιώσει δεδομένα χωρίς ίχνος)
* Reproducibility (R): 5  
  (χρειάζεται κάποια “αλλοίωση” ή πράξη που περνάει κάτω από τα ραντάρ)
* Exploitability (E): 5  
  (όποιος έχει πρόσβαση στο σύστημα με κάποια μορφή εξουσίας, π.χ. Teacher με κακόβουλη πρόθεση)
* Affected Users (A): 6  
  (μπορεί να επηρεάσει έναν αριθμό χρηστών — όχι πάντα όλους, αλλά και αρκετούς)
* Discoverability (D’): 8  
  (δεν είναι “κρυφό” το ότι δεν υπάρχουν logs· ο επιτιθέμενος το ανακαλύπτει εύκολα)

Σύνολο: 10 + 5 + 5 + 6 + 8 = 34  
Σοβαρότητα: High

Information Disclosure: Unauthorized File Access

* Damage (D): 5  
  (συνήθως διαρροή υλικού/βαθμών· όχι ολική καταστροφή)
* Reproducibility (R): 7.5  
  (εφικτό με δοκιμές path, IDOR, λανθασμένα permission checks)
* Exploitability (E): 5  
  (υπάρχουν πιθανές μέθοδοι, αλλά δεν είναι “κάνε κλικ και έγινε”)
* Affected Users (A): 10  
  (στη χειρότερη, μπορούν να διαρρεύσουν δεδομένα όλων των φοιτητών)
* Discoverability (D’): 8  
  (εάν δεν υπάρχει server-side authorization, εύκολα εντοπίζεται με βασικές δοκιμές)

Σύνολο: 5 + 7.5 + 5 + 10 + 8 = 35.5  
Σοβαρότητα: High

Denial of Service: Large File Upload Flood

* Damage (D): 10  
  (το σύστημα “κρεμάει” ή γεμίζει ο δίσκος· το e-class γίνεται μη διαθέσιμο)
* Reproducibility (R): 7.5  
  (αρκετά απλό με scripts/automation)
* Exploitability (E): 5  
  (χρειάζεται κάποια βασική ικανότητα, αλλά όχι κάτι τρομερό)
* Affected Users (A): 10  
  (κανείς δεν έχει πρόσβαση στο σύστημα όταν πέφτει)
* Discoverability (D’): 8  
  (η δυνατότητα ανεβάσματος μεγάλων αρχείων είναι εμφανής, μάλλον εύκολο να το δοκιμάσει κάποιος)

Σύνολο: 10 + 7.5 + 5 + 10 + 8 = 40.5  
Σοβαρότητα: Critical

Elevation of Privilege: Student → Teacher Escalation

* Damage (D): 9  
  (ο φοιτητής γίνεται δυνάμει “καθηγητής”, τροποποιεί μαθήματα/βαθμολογίες)
* Reproducibility (R): 7.5  
  (συνήθως αν υπάρχει κάποιο λογικό κενό π.χ. IDOR ή λανθασμένος role check)
* Exploitability (E): 5  
  (απαιτείται γνώση ή κάποιο exploit λογικής)
* Affected Users (A): 10  
  (όλοι οι φοιτητές μπορεί να επηρεαστούν· π.χ. αλλοιώσεις βαθμών σε πολλά μαθήματα)
* Discoverability (D’): 7  
  (δεν είναι όσο εμφανές όσο π.χ. μία κλασική SQLi· αλλά με προσπάθεια εντοπίζεται)

Σύνολο: 9 + 7.5 + 5 + 10 + 7 = 38.5  
Σοβαρότητα: High

Συνολική Εικόνα:

* Critical (≥ 40): Teacher Spoofing, SQL Injection, Large File Upload DoS
* High (≈ 30–39): Insufficient Logging, Unauthorized File Access, Student → Teacher Escalation

## 2.3 Step 3: Determine Countermeasures and Mitigation

### 2.3.1 Countermeasures by OWASP ASVS

Για την επιλογή των μέτρων προστασίας ακολουθείται το πρότυπο OWASP ASVS

(Application Security Verification Standard).

To ASVS έχει 3 επίπεδα εφαρμογών:

Επίπεδο 1(Opportunistic): αποτελεί το ελάχιστο security που απαιτείται για μια εφαρμογή. Αφορά μικρές εφαρμογές/επιχειρήσεις. Mostly automated reviews

Επίπεδο 2(Standard): Καλό για εφαρμογές που περιέχουν ευαίσθητά δεδομένα. Έχει πιο αυστηρό authentication και manual code reviews.

Επίπεδο 3(Advanced): Έχει πολύ αυστηρό security που απαιτεί κρυπτογραφία, εκτενείς μηχανισμούς ασφαλείας και διαρκείς ελέγχους. Αφορά εφαρμογές μεγάλου βεληνεκούς σε τομείς όπως η τράπεζες και η υγεία.

Η προσέγγιση της εφαρμογής συνδυάζει τα πρώτα 2 επίπεδα.

**Εφαρμογή μέτρων προστασίας με βάση το ASVS του OWASP:**

**Teacher Spoofing (Spoofing)**

OWASP ASVS Κατηγορίες

* V2: Authentication Verification Requirements
  + *V2.1 Password Security*
  + *V2.2 General Authenticator Security*
* V3: Session Management Requirements
  + (Αν κλαπεί το session token, χρειάζεται ασφαλής διαχείριση session)

Προτεινόμενα Αντίμετρα

1. Strong Password Policies & Storage
   * *ASVS V2.1.2*: Χρήση αλγορίθμων hashing (bcrypt, Argon2).
   * Επιβολή ελάχιστου μήκους, πολυπλοκότητας, επανάληψης check για κοινά passwords (password blacklist).
2. Brute Force Protection
   * *ASVS V2.2.1*: Rate limiting ή CAPTCHA μετά από συγκεκριμένο αριθμό αποτυχημένων προσπαθειών.
   * Lockout / progressive delay για επαναλαμβανόμενες αποτυχημένες συνδέσεις.
3. Secure Session Tokens
   * *ASVS V3.1.2*: Μοναδικά και τυχαία session IDs, μόνο μέσω HTTPS, HttpOnly και Secure flags.
   * Προαιρετικά, υλοποίηση session rotation μετά το login (ανανεώνεται το session ID).
4. Δυνατότητα 2FA (προαιρετικά)
   * Για “Teacher” ή “Admin” logins, μπορεί να απαιτείται επιπλέον επίπεδο ασφαλείας (π.χ. OTP).

**SQL Injection (Tampering)**

OWASP ASVS Κατηγορίες

* V5: Validation, Sanitization and Encoding Requirements
  + *V5.3* (Output Encoding and Injection Prevention)

Προτεινόμενα Αντίμετρα (Mitigation Strategies)

1. Parametrized Queries / Prepared Statements
   * Για κάθε εντολή SQL, να χρησιμοποιούνται *prepared statements* (π.χ. SELECT \* FROM users WHERE email=?) με binding παραμέτρων.
   * Αποφεύγουμε concatenation (π.χ. "...WHERE email='" + userInput + "'").
2. Input Validation & Escaping
   * Επιβολή ορίων (length checks), whitelist χαρακτήρων όπου εφικτό.
   * Ειδικά σε πεδία επικίνδυνα (π.χ. login, search), κάνουμε αυστηρό filtering πριν περάσει στην DB.
3. Least Privilege DB Accounts
   * Ο λογαριασμός της εφαρμογής στη βάση δεν πρέπει να έχει δικαιώματα DROP ή ALTER αν δεν είναι απολύτως απαραίτητο.
4. Security Testing
   * Χρήση εργαλείων SAST/DAST για εντοπισμό πιθανών injection points.
   * Ενημέρωση προγραμματιστών για secure coding practices.

**Insufficient Logging (Repudiation)**

OWASP ASVS Κατηγορίες

* V7: Error Handling and Logging Requirements
  + *V7.1* (Log Content Requirements)
  + *V7.2* (Log Processing)
  + *V7.3* (Log Protection)

Προτεινόμενα Αντίμετρα

1. Audit Logging
   * Καταγραφή κρίσιμων γεγονότων (login, logout, upload, αλλαγή βαθμολογιών).
   * Include userID, timestamp, IP, τι δράση έγινε.
2. Log Integrity
   * *ASVS V7.3.3*: Προστασία logs από τροποποίηση (access μόνο από admin, hashing logs, κ.λπ.).
3. Centralized Logging & Monitoring
   * Χρήση SIEM ή κεντρικού logger για καλύτερη ανάλυση.
   * Alerts όταν συμβαίνουν ύποπτες ενέργειες (π.χ. μαζικά uploads, πολλά logins).
4. Consistent Log Format
   * Έτσι, διευκολύνεις το incident investigation και αποτρέπεις ο επιτιθέμενος να διαφύγει αφήνοντας μηδενικά ίχνη.

**Unauthorized File Access (Information Disclosure)**

OWASP ASVS Κατηγορίες

* V4: Access Control Verification Requirements
  + Π.χ. *V4.1.1* (Verify that the principle of least privilege exists)
  + *V4.2.1* (Verify that all user and data attributes and policy information used by access controls cannot be manipulated…)
* V5.3 (Injection Prevention) – αν η πρόσβαση γίνεται μέσω URLs/IDs

Προτεινόμενα Αντίμετρα

1. Server-Side Authorization Checks
   * Σε κάθε αίτημα (π.χ. /files/123.pdf), ο server ελέγχει αν ο τρέχων χρήστης δικαιούται πρόσβαση (role, owner κ.λπ.).
   * Αποφυγή “direct link” που παρακάμπτει δικαιώματα.
2. Randomized File Paths / Filenames
   * Μην αποθηκεύεις τα αρχεία σε φάκελο /uploads/username/… που ο καθένας μπορεί να μαντέψει. Χρησιμοποιούμε UUID ή hash-based naming.
3. Least Privilege File System Permissions
   * Ο web server να μην έχει read/write δικαιώματα εκτός του απαραίτητου φακέλου.
   * Αν οι φοιτητές δεν δικαιούνται πρόσβαση σε κάποια αρχεία, αυτά παραμένουν εκτός “public” περιοχής.
4. Insecure Direct Object References
   * Χρήση *indirect* identifiers (π.χ. tokens) αντί για εμφανή IDs στις URLs, ώστε να μη μαντεύονται εύκολα.

**Large File Upload DoS (Denial of Service)**

OWASP ASVS Κατηγορίες

* V12: File and Resources Verification Requirements
  + *V12.1* (File Upload requirements)
* V11.1 Business Logic Security Requirements
  + (Έλεγχος για υπερβολικά / ύποπτα αιτήματα)

Προτεινόμενα Αντίμετρα

1. Μέγιστο Επιτρεπόμενο Μέγεθος Αρχείου
   * *ASVS V12.1.1*: Θέτουμε όριο μεγέθους (π.χ. 10MB). Απορρίπτουμε ή τερματίζουμε σύνδεση αν ξεπεραστεί.
   * Εφαρμογή quota *ανά χρήστη* ή *ανά μάθημα*.
2. Rate Limiting / Throttling
   * Περιορισμός ταχύτητας ανεβάσματος (π.χ. 1 request το δευτερόλεπτο) ή αριθμού uploads ανά λεπτό για τον ίδιο χρήστη/IP.
   * Αν παρατηρείται *burst* traffic, μπορεί να χρειαστεί CAPTCHA ή προσωρινό block.
3. Adequate Disk Monitoring
   * Ειδοποιήσεις (alerts) αν ο ελεύθερος χώρος στο File System πέφτει κάτω από ένα όριο (π.χ. 80%).
   * Αυτόματος καθαρισμός προσωρινών αρχείων (temp files).
4. Validation Τύπου Αρχείου
   * Αν η εφαρμογή επιτρέπει μόνο PDF, DOC κ.λπ., επιβάλλουμε MIME/extension checks (μειώνει έμμεσα κάποιους τύπους κακόβουλων επιθέσεων).

**Student → Teacher Escalation (Elevation of Privilege)**

OWASP ASVS Κατηγορίες

* V4: Access Control
  + *V4.1.3* (Verify the principle of least privilege…)
  + *V4.2* (Enforce role-based access control properly)

Προτεινόμενα Αντίμετρα

1. Strict Role-Based Access Control (RBAC)
   * Ο ρόλος “Student” έχει μόνο *readonly* πρόσβαση σε βαθμούς του ίδιου.
   * Ο ρόλος “Teacher” διαχειρίζεται μόνο συγκεκριμένα μαθήματα.
   * Να μην υπάρχουν endpoints που επιτρέπουν αλλαγή ρόλου (εκτός αν το κάνει ο Admin).
2. Server-Side Role Validation
   * Σε κάθε αίτημα που καλεί μια ευαίσθητη λειτουργία (update grade, upload υλικού), ελέγχουμε if (role == TEACHER) στο server. Δεν βασιζόμαστε σε client-side “role=teacher”.
3. Session Management
   * (Σε περίπτωση προσπάθειας bypass session) Απαραίτητο session token rotation, secure flags κ.λπ.

### 2.3.2 Mitigation Strategy

Προχωρώντας σε ένα ολοκληρωμένο σχέδιο προστασίας, εξετάζουμε πρώτα τα ευρήματα της αξιολόγησής μας μέσω DREAD, ώστε να αναγνωρίσουμε ποιες απειλές επηρεάζουν περισσότερο την ακεραιότητα και τη διαθεσιμότητα της πλατφόρμας.

Με γνώμονα τη σοβαρότητα που προκύπτει από την ανάλυση (Critical ή High), δίνουμε προτεραιότητα στις πιο επικίνδυνες ευπάθειες, όπως η πιθανότητα αλλοίωσης δεδομένων από SQL Injection ή η πλαστοπροσωπία χρηστών με αυξημένα δικαιώματα.

Αρχικά, υλοποιούμε άμεσα διορθωτικές δράσεις για να περιορίσουμε τον κίνδυνο στις κρισιμότερες περιοχές (Phase 1), ενώ παράλληλα προβλέπουμε επόμενο στάδιο (Phase 2) όπου θα ενσωματωθούν βελτιωμένες πρακτικές θωράκισης και επιπρόσθετα εργαλεία παρακολούθησης. Τέλος, σε ένα τρίτο επίπεδο, στοχεύουμε στη διαρκή παρατήρηση και επαναξιολόγηση του συστήματος (Phase 3), εφαρμόζοντας περιοδικούς ελέγχους και αναβαθμίσεις, ώστε να διασφαλίζεται ότι η πλατφόρμα ανταποκρίνεται στις νέες απαιτήσεις και τις εξελισσόμενες τεχνικές επιθέσεις.

Με αυτήν τη στρατηγική, εξασφαλίζουμε ότι η αντιμετώπιση των απειλών δεν εξαντλείται σε μια μονοδιάστατη υλοποίηση, αλλά αποτελεί μια δυναμική διαδικασία που προσαρμόζεται στη συνεχή εξέλιξη του περιβάλλοντος ασφαλείας.

* **Phase 1 (Immediate Mitigation):** Σε αυτό το πρώτο στάδιο, επικεντρωνόμαστε στις πιο σοβαρές ευπάθειες που εντοπίστηκαν από την ανάλυση DREAD. Εφαρμόζονται ταχύτατα διορθωτικά μέτρα που μειώνουν δραστικά την πιθανότητα κρίσιμων επιθέσεων—για παράδειγμα, περιορίζονται αμέσως τα κενά που θα μπορούσαν να επιτρέψουν αλλοίωση ζωτικών δεδομένων ή πλαστοπροσωπία χρηστών με αυξημένα δικαιώματα.
* **Phase 2 (Hardening & Enhancements):** Αφού αντιμετωπιστούν οι επείγοντες κίνδυνοι, ακολουθεί ένα στάδιο ενίσχυσης της εφαρμογής με πιο εξελιγμένες πρακτικές και ολοκληρωμένες λύσεις. Σε αυτό το στάδιο, πέρα από την απλή «διόρθωση» επιμέρους σημείων, επανασχεδιάζονται ή αναβαθμίζονται κρίσιμα υποσυστήματα, προκειμένου να διασφαλιστεί ότι δεν θα υπάρξουν κρυφά παραθυράκια που θα επιτρέψουν μελλοντικές επιθέσεις. Ακόμα, αξιοποιείται η ευκαιρία για καλύτερη διαχείριση δικαιωμάτων και για εφαρμογή βέλτιστων πρακτικών στην αρχιτεκτονική και τον κώδικα.
* **Phase 3 (Ongoing Monitoring & Continuous Improvement):** Αφού η πλατφόρμα διαθέτει πλέον τα βασικά και τα ενισχυμένα μέτρα προστασίας, δίνεται έμφαση στη συνεχή παρακολούθηση και αξιολόγηση. Εφαρμόζονται διεργασίες και εργαλεία που καταγράφουν συστηματικά τα συμβάντα, αναλύουν ύποπτες δραστηριότητες και παράγουν ειδοποιήσεις σε πραγματικό χρόνο. Παράλληλα, διενεργούνται περιοδικές δοκιμές ασφαλείας (penetration tests, κ.λπ.) και επικαιροποιείται το μοντέλο απειλών (Threat Model) εφόσον αλλάξει η αρχιτεκτονική ή προστεθούν νέες λειτουργίες. Με αυτήν την προσέγγιση, η ασφάλεια γίνεται δυναμική και προσαρμοστική στις νεότερες απαιτήσεις και τεχνικές επιθέσεων.

## 2.4 Step 4: Assess your work

**Σύντομη Ανακεφαλαίωση:**

Μέχρι στιγμής, έχουμε προχωρήσει σε μια ολοκληρωμένη διαδικασία Threat Modeling για την e-learning πλατφόρμα, ακολουθώντας τα κλασικά βήματα της ανάλυσης του OWASP και αξιοποιώντας τη μεθοδολογία STRIDE για τον εντοπισμό απειλών και το μοντέλο DREAD για την αξιολόγηση κινδύνων. Συγκεκριμένα, καταγράψαμε το High-Level Data Flow Diagram, οριοθετήσαμε τα βασικά security boundaries και τα entry/exit points, και κατηγοριοποιήσαμε τις απειλές (SQL Injection, Spoofing, DoS, κ.λπ.) σε σχέση με το ποια στοιχεία της εφαρμογής επηρεάζουν (Web Server, Database, File System κ.λπ.). Στη συνέχεια, υπολογίσαμε τη σοβαρότητα (Critical/High) των πιο σημαντικών απειλών με βάση το DREAD, ώστε να έχουμε μια ξεκάθαρη προτεραιοποίηση. Γι’ αυτές τις κρίσιμες ευπάθειες, διαμορφώσαμε ένα σχέδιο αντιμετώπισης, καθώς και ένα θεωρητικό πλαίσιο για τη σταδιακή υλοποίηση και τη διαρκή βελτίωση της ασφάλειας της εφαρμογής.

# 3. Συμπεράσματα:

Σκοπός της παραπάνω εργασίας ήταν η υλοποίηση ενός Application Threat Modelling για μια web εφαρμογή ακολουθώντας τα βήματα τις μεθοδολογίας του OWASP Threat Modelling Process. Χρησιμοποιήθηκε το εργαλείο Microsoft Threat Modelling Tool 2016 με το οποίο αναλύσαμε την εφαρμογή σε περαιτέρω οντότητες και δημιουργήσαμε Data Flow Diagrams. Εφόσον είδαμε την εφαρμογή από σχεδιαστική άποψη βάζοντας περιορισμούς/ παραδοχές σε κάθε οντότητα, με βάση τον τρόπο υλοποίησής μας, προχωρήσαμε στην αναλυτική πλευρά της εφαρμογής μας, καθώς το εργαλείο μας αναλύει τις απειλές με βάση το σχεδιασμό που έχουμε πραγματοποιήσει. Σε κάθε φάση του σχεδιασμού είχαμε τη δυνατότητα του να εξάγουμε ένα report από το εργαλείο Microsoft Threat Modelling Tool 2016 για να βλέπουμε πιο αναλυτικά την κατάσταση της εφαρμογής μας.

Οι απειλές κατηγοριοποιούνται από το εργαλείο με βάση το μοντέλο STRIDE, οπότε γνωρίσαμε και ένα τρόπο κατηγοριοποίησής των απειλών.

Στη συνέχεια αξιολογήσαμε τις απειλές με το DREAD μοντέλο το οποίο αποδίδει αριθμητική τιμή σε κάθε απειλή οπότε είναι μια χρήσιμη μεθοδολογία για βαθμολόγηση και σύγκριση των απειλών.

Τέλος προτείναμε αντίμετρα με βάση το OWASP ASVS το οποίο προτείνει security controls και secure development requirements, και κάναμε mitigate τις κυριότερες απειλές.

Όλες αυτές οι διαδικασίες γίνονται χωρίς κώδικα και είναι εξαιρετικά χρήσιμες σε πολλές περιπτώσεις, για παράδειγμα ως προεργασία penetration testing ή και ως μέτρο επικοινωνίας με το developer team σε περίπτωση δημιουργίας εφαρμογής.

# 4. Πηγές:

[1] <https://owasp.org/www-community/Threat_Modeling>

[2] <https://owasp.org/www-community/Threat_Modeling_Process>

[3] <https://en.wikipedia.org/wiki/STRIDE_model>

[4] <https://threat-modeling.com/dread-threat-modeling/>

[5] <https://en.wikipedia.org/wiki/DREAD_(risk_assessment_model)>

[6] <https://owasp.org/www-project-application-security-verification-standard/>

[7] <https://cheatsheetseries.owasp.org/index.html>

[8] https://github.com/OWASP/ASVS/tree/v4.0.3