Πανεπιστήμιο Πειραιώς
Τμήμα Πληροφορικής

# Εγχειρίδιο Ανάλυσης και Σχεδιασμού

Εκπαιδευτικού Λογισμικού Εκμάθησης JavaScript

**Συγγραφείς:**
Γιαγιας Δημητριος Π21018
Χρτσικοπουλος Γεωργιος Π21191
Κοργιανιτης Μαρκος Π21067

**Υπεύθυνη Διδάσκουσα:**
Α. Κοντογιαννη

Ιούνιος 2025

# Περιεχόμενα

# 1 Εισαγωγή

## 1.1 Σκοπός και Στόχοι του Λογισμικού

Ο κύριος σκοπός της εφαρμογής (ScriptWise) είναι η δημιουργία ενός σύγχρονου και διαδραστικού εκπαιδευτικού περιβάλλοντος για την εκμάθηση της γλώσσας προγραμματισμού JavaScript. Η εφαρμογή σχεδιάστηκε με επίκεντρο τον χρήστη, υιοθετώντας αρχές της ενεργητικής μάθησης (active learning), όπου ο εκπαιδευόμενος συμμετέχει ενεργά μέσω πρακτικών ασκήσεων και άμεσης ανατροφοδότησης, καθώς και της εξατομικευμένης μάθησης (personalized learning), όπου το περιεχόμενο και ο ρυθμός προσαρμόζονται στις ατομικές του ανάγκες. Στόχος είναι η ενίσχυση της κατανόησης, η καλλιέργεια του ενδιαφέροντος για τον προγραμματισμό και η αποτελεσματική ανάπτυξη δεξιοτήτων.

Οι βασικοί στόχοι που τέθηκαν κατά την ανάπτυξη του λογισμικού, αντικατοπτρίζοντας τις παραπάνω μαθησιακές φιλοσοφίες, είναι οι εξής:

- **Παροχή Δομημένου Εκπαιδευτικού Υλικού:** Προσφορά σαφών και κατανοητών ενοτήτων διδασκαλίας που καλύπτουν από τις θεμελιώδεις αρχές έως και πιο προχωρημένες έννοιες της JavaScript, υποστηρίζοντας την οικοδόμηση της γνώσης (constructivism) βήμα προς βήμα.
- **Ενεργητική Μάθηση και Πρακτική:** Ενσωμάτωση διαδραστικού επεξεργαστή κώδικα (code editor) εντός των μαθημάτων, επιτρέποντας στους χρήστες να πειραματιστούν και να ολοκληρώσουν ασκήσεις κώδικα σε πραγματικό χρόνο, προωθώντας τη μάθηση μέσω της πράξης (learning by doing).
- **Συνεχής Αυτοαξιολόγηση και Ανατροφοδότηση:** Δυνατότητα αξιολόγησης της γνώσης μέσω ποικίλων quiz (ανά ενότητα, επαναληπτικά, αξιολόγησης αρχικού επιπέδου και τελικής επανάληψης) με διαφορετικούς τύπους ερωτήσεων, παρέχοντας άμεση ανατροφοδότηση για την ενίσχυση της κατανόησης.
- **Εξατομίκευση της Μαθησιακής Εμπειρίας:**
  - Προσαρμογή της μαθησιακής διαδρομής βάσει των επιδόσεων του χρήστη (π.χ., πρόταση παράκαμψης μαθημάτων μετά από επιτυχή αξιολόγηση αρχικού επιπέδου).
  - Προσαρμογή της παρουσίασης του περιεχομένου ανάλογα με τις δηλωμένες προτιμήσεις του χρήστη (π.χ., έμφαση σε οπτικό ή κειμενικό υλικό).
  - Πρόταση επιπλέον υλικού και στοχευμένης επανάληψης σε περιοχές όπου ο χρήστης αντιμετωπίζει δυσκολίες, υποστηρίζοντας την καθοδηγούμενη ανακάλυψη (scaffolding).
  - Δυνατότητα επιλογής συγκεκριμένων ¨μαθησιακών μονοπατιών' (learning paths) που ταιριάζουν στους στόχους του χρήστη.
- **Παρακολούθηση και Οπτικοποίηση της Προόδου:** Καταγραφή και εμφάνιση των στατιστικών προόδου του χρήστη (π.χ., ολοκληρωμένα μαθήματα, επιδόσεις στα quiz) μέσω γραφημάτων για την παροχή ανατροφοδότησης και την ενίσχυση του κινήτρου, σύμφωνα με τις αρχές της ορατής μάθησης (visible learning).

## 1.2 Βασικές Λειτουργίες της Εφαρμογής

Η εφαρμογή αυτή δημιουργήθηκε για να κάνει την εκμάθηση της γλώσσας προγραμματισμού JavaScript πιο εύκολη και προσαρμοσμένη στις ανάγκες του καθενός. Συγκεκριμένα, η πλατφόρμα μας προσφέρει:

- **Μαθήματα για όλα τα Επίπεδα:** Βρίσκεις υλικό και παραδείγματα κώδικα για να μάθεις JavaScript, από τα πρώτα σου βήματα μέχρι πιο σύνθετα θέματα. Όλα είναι

οργανωμένα σε σειρές μαθημάτων (courses) και μικρότερα κεφάλαια (lessons).

- **Γράψε Κώδικα Άμεσα:** Μέσα σε κάθε μάθημα, υπάρχει ένας επεξεργαστής κώδικα. Εκεί μπορείς να γράψεις τη δική σου JavaScript, να τη δεις να ¨τρέχει' απευθείας στον browser σου και να κάνεις τις ασκήσεις.
- **Τεστ για να Δεις τι Έμαθες:** Υπάρχουν διάφορα quiz (πολλαπλής επιλογής, σωστό/λάθος, συμπλήρωσης κενού) για να τεστάρεις τις γνώσεις σου μετά από κάθε ενότητα, ή για να κάνεις μια γενική επανάληψη. Θα βρεις επίσης τεστ για να δεις το αρχικό σου επίπεδο ή για μια τελική αξιολόγηση.
- **Δες την Πρόοδό σου:** Η εφαρμογή κρατάει αρχείο για τα μαθήματα που ολοκληρώνεις και πώς τα πήγες στα quiz.
- **Μάθηση στα Μέτρα σου:** Η εφαρμογή προσπαθεί να σε βοηθήσει προσωπικά:
  - Αν ξέρεις ήδη κάποια πράγματα, ένα αρχικό τεστ μπορεί να σου προτείνει να προσπεράσεις κάποια από τα πρώτα μαθήματα.
  - Αν δυσκολευτείς σε κάποιο τελικό quiz, θα σου προτείνει μαθήματα για επανάληψη και επιπλέον υλικό από άλλες πηγές.
  - Μπορείς να πεις αν προτιμάς να μαθαίνεις διαβάζοντας ή βλέποντας (π.χ. βίντεο), και η εφαρμογή θα προσπαθεί να σου δείχνει πρώτα αυτό που σου ταιριάζει.
  - Μπορείς να διαλέξεις ένα ¨μονοπάτι μάθησης' (learning path) που σε οδηγεί βήμα-βήμα ανάλογα με τους στόχους σου (π.χ. 'Frontend Developer¨).
- **Τα Στατιστικά σου με μια Ματιά:** Βλέπεις πόσες μέρες στη σειρά μαθαίνεις (learning streak), πόσα quiz έχεις κάνει, και πώς τα πας γενικά στα quiz μέσα από απλά γραφήματα.
- **Ο Λογαριασμός σου:** Μπορείς να κάνεις εγγραφή, να συνδέεσαι, να αποσυνδέεσαι και να αλλάζεις τα στοιχεία του προφίλ σου (όνομα, email, password) και τις προτιμήσεις μάθησης.

## 1.3 Αναμενόμενοι Χρήστες

Η παρούσα εκπαιδευτική εφαρμογή σχεδιάστηκε με σκοπό να εξυπηρετήσει ένα ευρύ φάσμα εκπαιδευομένων που επιθυμούν να μάθουν ή να βελτιώσουν τις γνώσεις τους στη γλώσσα προγραμματισμού JavaScript. Οι κύριες ομάδες χρηστών στις οποίες απευθυνόμαστε είναι:

- **Αρχάριοι στον Προγραμματισμό:** Άτομα που δεν έχουν προηγούμενη εμπειρία στον προγραμματισμό και επιθυμούν να ξεκινήσουν το ταξίδι τους με τη JavaScript ως πρώτη γλώσσα.
- **Φοιτητές και Σπουδαστές:** Εκπαιδευόμενοι σε τμήματα πληροφορικής, web development ή συναφείς τομείς που διδάσκονται JavaScript και αναζητούν ένα συμπληρωματικό, διαδραστικό εργαλείο.
- **Προγραμματιστές Άλλων Γλωσσών:** Προγραμματιστές με εμπειρία σε άλλες γλώσσες που θέλουν να προσθέσουν τη JavaScript στις δεξιότητές τους.
- **Επαγγελματίες για Upskilling/Reskilling:** Επαγγελματίες που επιθυμούν να αποκτήσουν γνώσεις JavaScript για να ενισχύσουν το επαγγελματικό τους προφίλ.
- **Άτομα που Μαθαίνουν Αυτοδίδακτα:** Όσοι προτιμούν να μαθαίνουν με τον δικό τους ρυθμό.

# 2 Ανάλυση Απαιτήσεων

Το παρόν κεφάλαιο περιγράφει τις απαιτήσεις που τέθηκαν για την ανάπτυξη του εκπαιδευτικού λογισμικού εκμάθησης JavaScript. Οι απαιτήσεις διακρίνονται σε λειτουργικές, οι οποίες καθορίζουν τις συγκεκριμένες ενέργειες και λειτουργίες που εκτελεί το σύστημα, και σε μη λειτουργικές, οι οποίες περιγράφουν τα ποιοτικά χαρακτηριστικά και τους περιορισμούς του συστήματος.

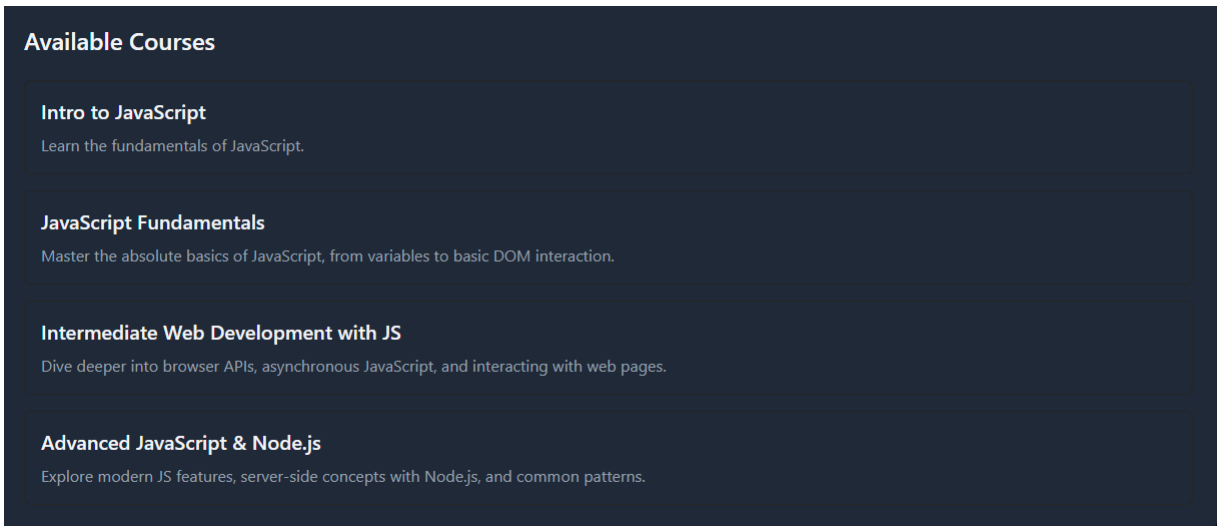
## 2.1 Λειτουργικές Απαιτήσεις

Οι λειτουργικές απαιτήσεις καθορίζουν τις συγκεκριμένες υπηρεσίες και λειτουργίες που παρέχει η εφαρμογή στους χρήστες της.

### 2.1.1 Ενότητες Διδασκαλίας

Η εφαρμογή πρέπει να παρέχει δομημένο εκπαιδευτικό περιεχόμενο για τη γλώσσα JavaScript, οργανωμένο σε διακριτές εκπαιδευτικές ενότητες (Courses) και επιμέρους μαθήματα (Lessons). Συγκεκριμένα:

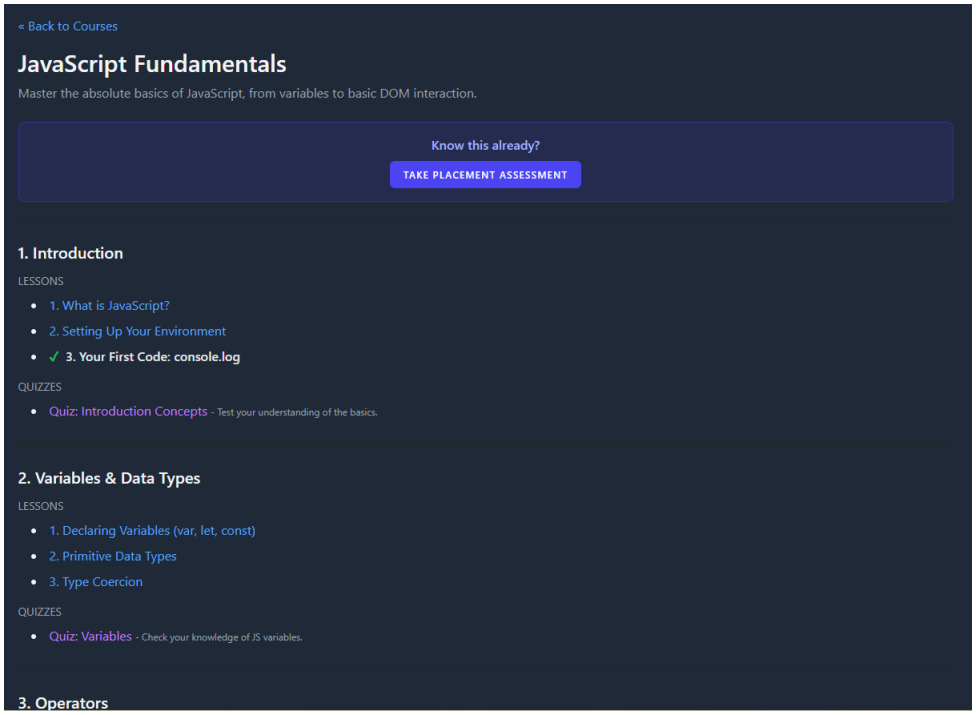
- **Παροχή Πολλαπλών Ενοτήτων (Courses):** Το σύστημα προσφέρει τις ενότητες: 'JavaScript Fundamentals¨, 'Intermediate Web Development with JS¨, και 'Advanced JavaScript & Node.js¨.




Σχήμα 1: Λίστα διαθέσιμων κύκλων μαθημάτων.

- **Δομή Ενοτήτων:** Κάθε Course αποτελείται από υποενότητες (Modules), οι οποίες περιέχουν τα μαθήματα (Lessons).
- **Περιεχόμενο Μαθημάτων (Lessons):** Κάθε Lesson περιλαμβάνει θεωρητικό υλικό, παραδείγματα κώδικα, και προαιρετικά οπτικοακουστικό υλικό (video_embed_html).

- **Διαδραστικός Επεξεργαστής Κώδικα:** Κάθε Lesson με άσκηση διαθέτει ενσωματωμένο code editor (Monaco Editor) για γραφή, τροποποίηση και client-side εκτέλεση κώδικα.
- **Ασκήσεις Μαθημάτων:** Τα Lessons μπορούν να περιλαμβάνουν assignment. Η επιτυχής ολοκλήρωση (βάσει expected_output) μαρκάρει το μάθημα ως ολοκληρωμένο.



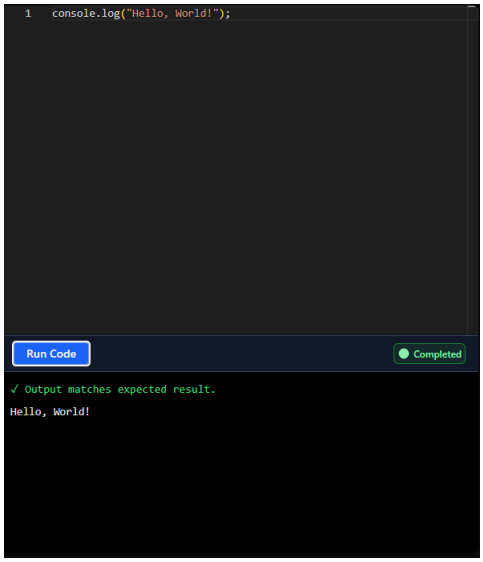

Σχήμα 2: Επισκόπηση ενός κύκλου μαθημάτων και των ενοτήτων του.


Σχήμα 3: Περιεχόμενο ενός μαθήματος.

### 2.1.2 Ερωτήσεις / Τεστ Αυτοαξιολόγησης

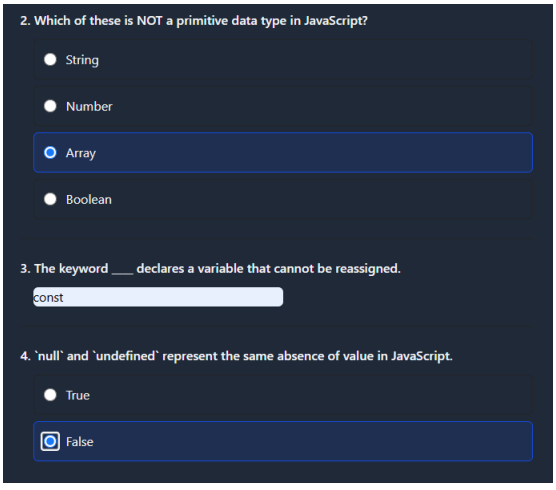
Η εφαρμογή παρέχει σύστημα για quizzes αυτοαξιολόγησης:

- **Τεστ ανά Ενότητα (Module Quizzes):** Κάθε διδακτική υποενότητα (Module) ενός Course πρέπει να μπορεί να συνδέεται με ένα ή περισσότερα quizzes που καλύπτουν το υλικό της συγκεκριμένης υποενότητας. Αυτά τα quizzes βοηθούν τον χρήστη να ελέγξει την κατανόηση των εννοιών που μόλις διδάχθηκε.
- **Επαναληπτικά Τεστ:** Η εφαρμογή πρέπει να υποστηρίζει επαναληπτικά τεστ που συνδυάζουν ερωτήσεις από προηγούμενες ενότητες ή καλύπτουν το σύνολο ενός Course. Συγκεκριμένα, υλοποιήθηκαν:




Σχήμα 4: Διαδραστικός επεξεργαστής κώδικα και άσκηση μαθήματος.

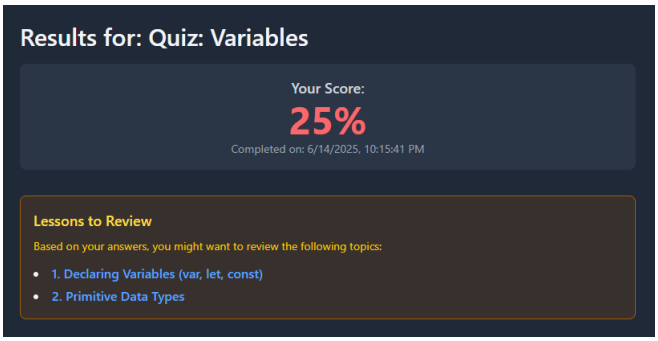
  - **Τελικό Επαναληπτικό Quiz Κύκλου Μαθημάτων (Post-course Review Quiz):** Στο τέλος κάθε Course, ο χρήστης μπορεί να κάνει ένα επαναληπτικό quiz. Αυτό το quiz περιλαμβάνει δυναμικά ερωτήσεις στις οποίες ο χρήστης είχε απαντήσει λανθασμένα σε προηγούμενα module quizzes του συγκεκριμένου course, καθώς και τυχαίες νέες ερωτήσεις από το σύνολο του course, για την ενίσχυση της μνήμης και της συνολικής κατανόησης.
  - **Τυχαίο Quiz Επανάληψης (Random Review Quiz):** Τυχαίες ερωτήσεις από ολοκληρωμένα μαθήματα.
- **Αξιολόγηση Αρχικού Επιπέδου (Pre-course Assessment Quiz):** Για πρόταση σημείου εκκίνησης.
- **Ποικιλία Μορφών Ερωτήσεων:** Πολλαπλής Επιλογής, Σωστού/Λάθους, Συμπλήρωσης Κενού.




Σχήμα 5: Διεπαφή διεξαγωγής Quiz.

- **Διεπαφή Διεξαγωγής Quiz:** Φιλικό περιβάλλον, άμεση ανατροφοδότηση (σκορ, σωστές/λανθασμένες απαντήσεις, εξηγήσεις).

- **Συστάσεις Βάσει Αποτελεσμάτων:** Πρόταση επανάληψης μαθημάτων και εξωτερικών πηγών.




Σχήμα 6: Αποτελέσματα Quiz και προτάσεις επανάληψης.

### 2.1.3 Αποθήκευση Στατιστικών Προόδου

Η εφαρμογή καταγράφει και αποθηκεύει δεδομένα προόδου:

- **Ολοκλήρωση Μαθημάτων (Lessons):** Μέσω του πίνακα user_progress (user_id, lesson_id, completed_at).
- **Προσπάθειες και Αποτελέσματα Quiz:** Στους πίνακες quiz_attempts (user_id, quiz_id (nullable), type, score, timestamps) και quiz_answers (attempt_id, question_id, user_answer, is_correct).
- **Δεδομένα Προφίλ και Προτιμήσεων Χρήστη:** Στον πίνακα users.

### 2.1.4 Προσαρμοσμένη Μάθηση (Adaptive Learning)

Η εφαρμογή στοχεύει στην παροχή μιας εμπειρίας προσαρμοσμένης μάθησης, όπου το εκπαιδευτικό περιεχόμενο και οι δραστηριότητες προσαρμόζονται δυναμικά στις επιδόσεις, τις προτιμήσεις και τις ανάγκες του κάθε μαθητή. Αυτό επιτυγχάνεται μέσω των παρακάτω μηχανισμών:

- **Αξιολόγηση Αρχικού Επιπέδου και Πρόταση Διαδρομής:** Μέσω του Pre-course Assessment Quiz και του CourseAssessmentController. Πριν την έναρξη ενός κύκλου μαθημάτων (Course), ο χρήστης έχει τη δυνατότητα να συμμετάσχει σε ένα χυιζ αξιολόγησης αρχικού επιπέδου (Pre-course Assessment Quiz). Βάσει των απαντήσεών του σε αυτό το quiz, το σύστημα αξιολογεί την κατανόησή του σε θέματα που καλύπτονται στα αρχικά μαθήματα του course. Εάν ο χρήστης επιδείξει επαρκή γνώση σε συγκεκριμένες ενότητες, η εφαρμογή του προτείνει να παρακάμψει τα αντίστοιχα αρχικά μαθήματα και να ξεκινήσει από ένα πιο προχωρημένο σημείο, αποφεύγοντας την επανάληψη υλικού που ήδη κατέχει.
- **Στοχευμένη Επανάληψη και Ενισχυτικό Υλικό:** Μέσω του Post-course Review Quiz, του CourseReviewController και του πίνακα external_resources.Μετά την ολοκλήρωση ενός τελικού επαναληπτικού quiz ενός course (Post-course Review Quiz), εάν ο χρήστης εξακολουθεί να κάνει λάθη σε ερωτήσεις που αντιστοιχούν σε συγκεκριμένα μαθήματα, το σύστημα: Τον ανακατευθύνει ή του προτείνει να επισκεφθεί ξανά τα μαθήματα (lessons) στα οποία παρουσίασε αδυναμίες. Παρέχει συνδέσμους προς επιλεγμένες εξωτερικές πηγές υλικού (π.χ. άρθρα, βίντεο, τεκμηρίωση) σχετικές με τα συγκεκριμένα μαθήματα, για περαιτέρω μελέτη και ενίσχυση της κατανόησης.

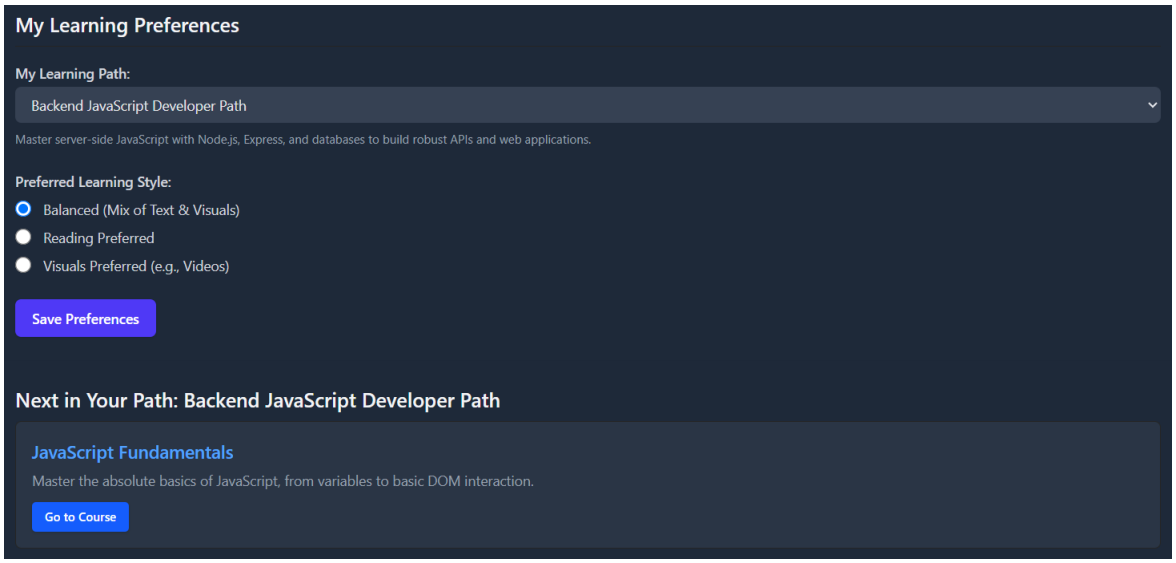

Σχήμα 7: Πρόταση μαθήματος μετά από αξιολόγηση αρχικού επιπέδου.

- **Προσαρμογή Παρουσίασης Περιεχομένου βάσει Μαθησιακού Στυλ:** Ο χρήστης μπορεί να δηλώσει στο dashboard τον προτιμώμενο τρόπο μάθησης (π.χ. 'visual' για έμφαση σε οπτικό υλικό, 'reading' για έμφαση σε κείμενο). Στα μαθήματα που διαθέτουν τόσο κειμενικό όσο και οπτικοακουστικό υλικό (π.χ. ενσωματωμένα βίντεο), η εφαρμογή προσαρμόζει τη σειρά εμφάνισης αυτών των στοιχείων. Για παράδειγμα, ένας χρήστης που προτιμά οπτικό υλικό θα δει πρώτα το βίντεο και μετά το σχετικό κείμενο.


Σχήμα 8: Προσαρμογή εμφάνισης περιεχομένου βάσει μαθησιακού στυλ.

- **Καθοδηγούμενα Μονοπάτια Μάθησης (Learning Paths):** Ο χρήστης μπορεί να επιλέξει μια προκαθορισμένη μαθησιακή διαδρομή (Learning Path) που αντιστοιχεί σε συγκεκριμένους εκπαιδευτικούς ή επαγγελματικούς στόχους (π.χ. "Frontend Developer Path", "Backend JavaScript Developer Path"). Η εφαρμογή, βάσει της επιλεγμένης διαδρομής και της προόδου του χρήστη στα γουρσες που την αποτελούν, προτείνει το

επόμενο λογικό course ή μάθημα για παρακολούθηση, παρέχοντας μια πιο δομημένη και καθοδηγούμενη πορεία μάθησης.




Σχήμα 9: Επιλεγμένη μαθησιακή διαδρομή και προτεινόμενο επόμενο μάθημα.

- **Ρύθμιση Δυσκολίας Ασκήσεων/Quiz:** Αν και δεν έχει υλοποιηθεί πλήρως δυναμική ρύθμιση δυσκολίας εντός ενός quiz, η επιλογή των ερωτήσεων στα επαναληπτικά quizzes (Post-course Review και Random Review) λαμβάνει υπόψη τις προηγούμενες επιδόσεις (εστιάζοντας σε λανθασμένες απαντήσεις), προσφέροντας έμμεσα μια μορφή προσαρμογής της δυσκολίας στην επανάληψη.

### 2.1.5 Διαχείριση Χρήστη

Παρέχονται βασικές λειτουργίες διαχείρισης λογαριασμού:

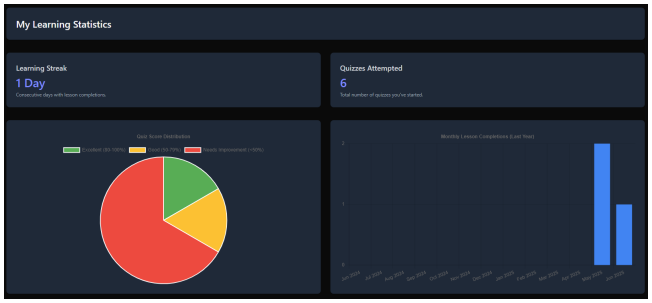
- Εγγραφή, Σύνδεση, Αποσύνδεση.
- Επεξεργασία Προφίλ (όνομα, email, password) και Μαθησιακών Προτιμήσεων (στυλ, path) μέσω του Dashboard και του UserPreferenceController.


Σχήμα 10: Διαχείριση προτιμήσεων χρήστη στο Dashboard.

### 2.1.6 Προβολή Αναφορών Προόδου

Οπτικοποιημένες αναφορές προόδου και στατιστικά στοιχεία:

- Μαθησιακό Σερί (Learning Streak).
- Σύνολο Προσπαθειών σε Quiz.
- Κατανομή Αποτελεσμάτων Quiz (Πίτα).
- Γράφημα Ολοκληρωμένων Μαθημάτων ανά Ημέρα ('Contribution Graph¨).




Σχήμα 11: Σελίδα στατιστικών προόδου χρήστη.

## 2.2 Μη Λειτουργικές Απαιτήσεις

Πέραν των λειτουργιών, τέθηκαν και ποιοτικές απαιτήσεις:

- **Χρηστικότητα (Usability):** Διαισθητικό και εύχρηστο περιβάλλον.
- **Απόδοση (Performance):** Λογικοί χρόνοι φόρτωσης, άμεση απόκριση του editor.
- **Ασφάλεια (Security):** Προστασία δεδομένων χρήστη, client-side εκτέλεση κώδικα από ελεγχόμενες πηγές περιεχομένου.
- **Συντηρησιμότητα (Maintainability):** Οργανωμένος κώδικας.
- **Συμβατότητα (Compatibility):** Λειτουργία σε σύγχρονους web browsers.

# 3 Σχεδιασμός Συστήματος

Το παρόν κεφάλαιο περιγράφει τον σχεδιασμό του εκπαιδευτικού λογισμικού.

## 3.1 Αρχιτεκτονική

Η αρχιτεκτονική της εφαρμογής βασίζεται σε ένα σύγχρονο web stack, συνδυάζοντας την ισχύ του Laravel framework για το backend και την ευελιξία του Vue.js με το Inertia.js για ένα δυναμικό frontend. **Βασικά Συστατικά Αρχιτεκτονικής:**

1. **Client (Πελάτης - Φυλλομετρητής Ιστού):** Ο χρήστης αλληλεπιδρά με την εφαρμογή μέσω ενός φυλλομετρητή ιστού (web browser). Το frontend, χτισμένο με Vue.js, είναι υπεύθυνο για την παρουσίαση της διεπαφής χρήστη (UI) και την αρχική διαχείριση των αλληλεπιδράσεων.
2. **Web Server & Backend Εφαρμογή (Laravel):** Ένας web serverφιλοξενεί την backend εφαρμογή που έχει αναπτυχθεί με το PHP framework Laravel.

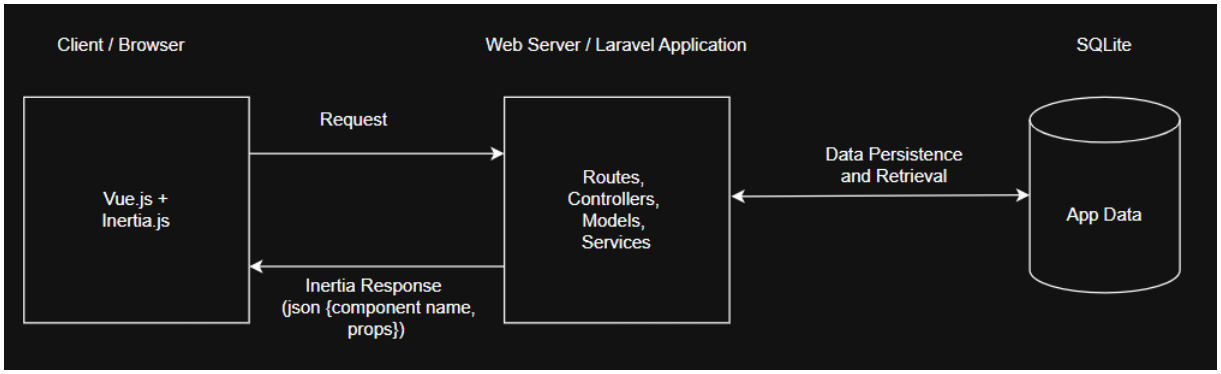
- Το Laravel ακολουθεί την αρχιτεκτονική Model-View-Controller (MVC):
  - **Models:** Αντιπροσωπεύουν τη δομή των δεδομένων και την αλληλεπίδραση με τη βάση δεδομένων (π.χ. User, Course, Lesson, Quiz).
  - **Views:** Σε αυτή την εφαρμογή, οι 'Views' του Laravel δεν είναι παραδοσιακά Blade templates που παράγουν HTML. Αντ΄ αυτού, οι Controllers επιστρέφουν απαντήσεις Inertia.js, οι οποίες ουσιαστικά ενημερώνουν το frontend ποιο Vue component να φορτώσει και τι δεδομένα (props) να του περάσει.
  - **Controllers:** Διαχειρίζονται τα αιτήματα του χρήστη, αλληλεπιδρούν με τα Models για την ανάκτηση ή αποθήκευση δεδομένων, και επιστρέφουν την κατάλληλη απόκριση (Inertia response) στο frontend.
- Το backend είναι επίσης υπεύθυνο για την αυθεντικοποίηση των χρηστών, την εξουσιοδότηση, την επικύρωση των δεδομένων και τη συνολική επιχειρησιακή λογική της εφαρμογής.

3. **Βάση Δεδομένων:** Μια σχεσιακή βάση δεδομένων (SQLite) χρησιμοποιείται για την αποθήκευση όλων των μόνιμων δεδομένων της εφαρμογής, όπως πληροφορίες χρηστών, περιεχόμενο μαθημάτων, πρόοδο χρηστών, και αποτελέσματα quiz. Το Laravel Eloquent ORM χρησιμοποιείται για την εύκολη αλληλεπίδραση με τη βάση δεδομένων.
4. **Inertia.js:** Το Inertia.js λειτουργεί ως ¨κόλλα' μεταξύ του Laravel backend και του Vue.js frontend. Επιτρέπει την ανάπτυξη σύγχρονων, single-page application (SPA) εμπειριών χωρίς την πολυπλοκότητα της δημιουργίας ενός ξεχωριστού API. Όταν ο χρήστης πλοηγείται, το Inertia κάνει αιτήματα XHR στο backend, και το backend επιστρέφει JSON που περιλαμβάνει το όνομα του Vue component που πρέπει να φορτωθεί και τα δεδομένα (props) που χρειάζεται. Το routing γίνεται κυρίως από το Laravel.

**Ροή Δεδομένων (Υψηλού Επιπέδου):**

1. Ο χρήστης αλληλεπιδρά με το Vue.js component στον browser του (π.χ. κλικάρει ένα link για ένα μάθημα).
2. Το Inertia.js υποβάλλει ένα αίτημα (request) στον Laravel server.
3. Το αντίστοιχο Route στο Laravel ενεργοποιεί τον κατάλληλο Controller.
4. Ο Controller επεξεργάζεται το αίτημα, αλληλεπιδρά με τα Models για να πάρει δεδομένα από τη Βάση Δεδομένων.

5. Ο Controller επιστρέφει ένα Inertia Response, το οποίο περιέχει το όνομα του Vue page component και τα δεδομένα (props) που χρειάζεται.
6. Το Inertia.js στο frontend λαμβάνει την απάντηση, φορτώνει δυναμικά το νέο Vue component (ή ενημερώνει το υπάρχον) με τα νέα props, ενημερώνοντας το UI χωρίς πλήρη επαναφόρτωση της σελίδας.

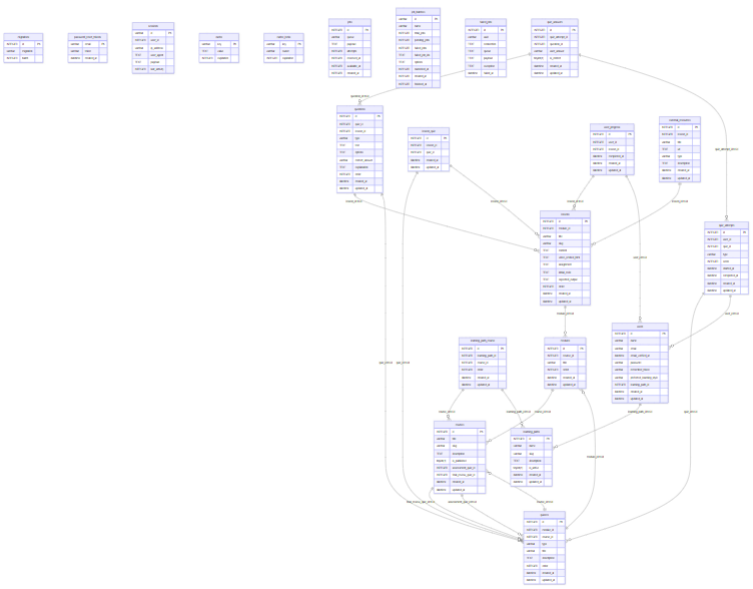



Σχήμα 12: Υψηλού Επιπέδου Αρχιτεκτονική Συστήματος.

## 3.2 Σχεδιασμός Βάσης Δεδομένων

Ο σχεδιασμός της βάσης δεδομένων αποτελεί θεμελιώδες τμήμα της εφαρμογής, καθώς είναι υπεύθυνος για την αποθήκευση και οργάνωση όλων των πληροφοριών που απαιτούνται για τη λειτουργία του εκπαιδευτικού λογισμικού. Χρησιμοποιήθηκε μια σχεσιακή βάση δεδομένων (SQLite για την ανάπτυξη), και η αλληλεπίδραση με αυτήν γίνεται μέσω του Laravel Eloquent ORM, το οποίο αντιστοιχίζει τους πίνακες της βάσης σε αντικείμενα (Models).

Παρακάτω περιγράφονται οι κύριοι πίνακες (μοντέλα) της βάσης δεδομένων και οι μεταξύ τους σχέσεις.



Σχήμα 13: Διάγραμμα Σχήματος Βάσης Δεδομένων (ERD).

**Περιγραφή Πινάκων (Μοντέλων):**

1. `users` **(Χρήστες):**

- *Περιγραφή:* Αποθηκεύει πληροφορίες για τους εγγεγραμμένους χρήστες της πλατφόρμας.
- *Βασικά Πεδία:* `id (PK)`, `name`, `email (unique)`, `password (hashed)`, `email_verified_at`, `preferred_learning_style (enum: 'reading', 'visual')`, `learning_path_id (FK to learning_paths, nullable)`, `remember_token`, `timestamp`.
- *Σχέσεις:*
  - Ένας χρήστης έχει πολλές εγγραφές προόδου (one-to-many with user_progress).
  - Ένας χρήστης έχει πολλές προσπάθειες quiz (one-to-many with quiz_attempts).
  - Ένας χρήστης ανήκει σε μία μαθησιακή διαδρομή (many-to-one with learning_paths).

2. `learning_paths` **(Μαθησιακές Διαδρομές):**

- *Περιγραφή:* Καθορίζει προκαθορισμένες ακολουθίες μαθημάτων για συγκεκριμένους μαθησιακούς στόχους.
- *Βασικά Πεδία:* `id (PK)`, `name`, `slug (unique)`, `description`, `is_active`, `timestamp`.
- *Σχέσεις:*
  - Μία μαθησιακή διαδρομή έχει πολλούς χρήστες (one-to-many with users).
  - Μία μαθησιακή διαδρομή έχει πολλούς κύκλους μαθημάτων (many-to-many with courses μέσω του πίνακα `learning_path_course`).

3. `courses` **(Κύκλοι Μαθημάτων):**

- *Περιγραφή:* Αντιπροσωπεύει ένα ολοκληρωμένο σύνολο διδακτικού υλικού για ένα ευρύτερο θέμα (π.χ., 'JavaScript Fundamentals¨).
- *Βασικά Πεδία:* `id (PK)`, `title`, `slug (unique)`, `description`, `is_published`, `assessment_quiz_i (FK to quizzes, nullable)`, `final_review_quiz_id (FK to quizzes, nullable)`, `timestamp`.
- *Σχέσεις:*
  - Ένας κύκλος μαθημάτων έχει πολλές ενότητες (one-to-many with modules).
  - Ένας κύκλος μαθημάτων ανήκει σε ένα (προαιρετικό) quiz αξιολόγησης (many-to-one with quizzes).
  - Ένας κύκλος μαθημάτων ανήκει σε ένα (προαιρετικό) quiz τελικής επανάληψης (many-to-one with quizzes).
  - Ένας κύκλος μαθημάτων ανήκει σε πολλές μαθησιακές διαδρομές (many-to-many with learning_paths μέσω του πίνακα `learning_path_course`).

4. `modules` **(Ενότητες):**

- *Περιγραφή:* Υποενότητες ενός Course, ομαδοποιούν σχετικά μαθήματα.
- *Βασικά Πεδία:* `id (PK)`, `course_id (FK)`, `title`, `order`, `timestamp`.
- *Σχέσεις:*
  - Μία ενότητα ανήκει σε έναν κύκλο μαθημάτων (many-to-one with courses).
  - Μία ενότητα έχει πολλά μαθήματα (one-to-many with lessons).
  - Μία ενότητα έχει πολλά quizzes (one-to-many with quizzes - για τα module quizzes).

5. `lessons` **(Μαθήματα):**

- *Περιγραφή:* Η μικρότερη διδακτική μονάδα, περιέχει το εκπαιδευτικό υλικό και τις ασκήσεις.
- *Βασικά Πεδία:* `id (PK)`, `module_id (FK)`, `title`, `slug (unique)`, `content (HTML/Markdown)`, `assignment`, `initial_code`, `expected_output`, `video_embed_html (nullable)`, `order`, `timestamp`.
- *Σχέσεις:*
  - Ένα μάθημα ανήκει σε μία ενότητα (many-to-one with modules).
  - Ένα μάθημα έχει πολλές εγγραφές προόδου χρηστών (one-to-many with user_progress).
  - Ένα μάθημα έχει πολλές εξωτερικές πηγές (one-to-many with external_resources).
  - Ένα μάθημα μπορεί να σχετίζεται με πολλές ερωτήσεις quiz (many-to-one from questions).

6. `quizzes`:

- *Περιγραφή:* Αντιπροσωπεύει ένα σύνολο ερωτήσεων για αξιολόγηση.
- *Βασικά Πεδία:* `id (PK)`, `type (enum: 'assessment', 'module', 'final_review')`, `module_id (FK, nullable)`, `title`, `description`, `order`, `timestamp`.
- *Σχέσεις:*
  - Ένα quiz έχει πολλές ερωτήσεις (one-to-many with questions).
  - Ένα quiz έχει πολλές προσπάθειες από χρήστες (one-to-many with quiz_attempts).
  - Ένα quiz (τύπου 'module') ανήκει σε μία ενότητα (many-to-one with modules).
  - Ένα quiz (τύπου 'assessment' ή 'final_review') μπορεί να συνδέεται με ένα course (one-to-one relationships from courses).

7. `questions` **(Ερωτήσεις):**

- *Περιγραφή:* Μια μεμονωμένη ερώτηση ενός quiz.
- *Βασικά Πεδία:* `id (PK)`, `quiz_id (FK)`, `lesson_id (FK, nullable)`, `type (enum: 'multiple_choice', 'true_false', 'fill_blank')`, `text`, `options (JSON, nullable)`, `correct_answer`, `explanation`, `order`, `timestamp`.
- *Σχέσεις:*
  - Μία ερώτηση ανήκει σε ένα quiz (many-to-one with quizzes).
  - Μία ερώτηση μπορεί να σχετίζεται με ένα μάθημα (many-to-one with lessons).

8. `user_progress` **(Πρόοδος Χρήστη):**

- *Περιγραφή:* Καταγράφει την ολοκλήρωση μαθημάτων από τους χρήστες.
- *Βασικά Πεδία:* `id (PK)`, `user_id (FK)`, `lesson_id (FK)`, `completed_at`, `timestamp`.
- *Σχέσεις:*
  - Ανήκει σε έναν χρήστη (many-to-one with users).
  - Ανήκει σε ένα μάθημα (many-to-one with lessons).
- *Περιορισμός:* Μοναδικό ζεύγος (`user_id, lesson_id`).

9. `quiz_attempts` **(Προσπάθειες Quiz):**

- *Περιγραφή:* Καταγράφει κάθε προσπάθεια ενός χρήστη σε ένα quiz.
- *Βασικά Πεδία:* `id (PK)`, `user_id (FK)`, `quiz_id (FK, nullable)`, `type (string: 'standard', 'random', 'final_review')`, `score`, `started_at`, `completed_at`, `timestamp`.
- *Σχέσεις:*
  - Ανήκει σε έναν χρήστη (many-to-one with users).

- Ανήκει σε ένα quiz (many-to-one with quizzes, αν `χυιζ_ιδ νοτ νυλλ`).
- Μία προσπάθεια έχει πολλές απαντήσεις (one-to-many with quiz_answers).

10. `quiz_answers` **(Απαντήσεις Quiz):**

- *Περιγραφή:* Αποθηκεύει τη συγκεκριμένη απάντηση ενός χρήστη σε μια ερώτηση μιας προσπάθειας quiz.
- *Βασικά Πεδία:* `id (PK)`, `quiz_attempt_id (FK)`, `question_id (FK)`, `user_answer`, `is_correct`, `timestamp`.
- *Σχέσεις:*
  - Ανήκει σε μία προσπάθεια quiz (many-to-one with quiz_attempts).
  - Ανήκει σε μία ερώτηση (many-to-one with questions).
- *Περιορισμός:* Μοναδικό ζεύγος (`quiz_attempt_id, question_id`).

11. `external_resources` **(Εξωτερικές Πηγές):**

- *Περιγραφή:* Σύνδεσμοι και πληροφορίες για επιπλέον υλικό μελέτης.
- *Βασικά Πεδία:* `id (PK)`, `lesson_id (FK)`, `title`, `url`, `type (enum)`, `description`, `timestamps`.
- *Σχέσεις:*
  - Ανήκει σε ένα μάθημα (many-to-one with lessons).

12. `learning_path_course` **(Πίνακας Συσχέτισης):**

- *Περιγραφή:* Συνδέει τις μαθησιακές διαδρομές με τους κύκλους μαθημάτων, καθορίζοντας τη σειρά τους.
- *Βασικά Πεδία:* `id (PK)`, `learning_path_id (FK)`, `course_id (FK)`, `order`.
- *Περιορισμός:* Μοναδικό ζεύγος (`learning_path_id, course_id`).

## 3.3 Σχεδιασμός Λειτουργιών Εξατομίκευσης

Η εφαρμογή ενσωματώνει διάφορους μηχανισμούς για την παροχή μιας εξατομικευμένης μαθησιακής εμπειρίας. Ο σχεδιασμός αυτών των λειτουργιών βασίζεται στην ανάλυση των δεδομένων προόδου και των προτιμήσεων του χρήστη. Παρακάτω περιγράφεται η βασική λογική πίσω από κάθε χαρακτηριστικό εξατομίκευσης:

1. **Πρόταση Παράκαμψης Μαθημάτων (Pre-course Assessment):**

- *Ενεργοποίηση:* Όταν ο χρήστης επιλέγει να ξεκινήσει ένα Course που διαθέτει Quiz Αξιολόγησης Αρχικού Επιπέδου (`assessment_quiz`).
- *Διαδικασία:*

  α′) Ο χρήστης ολοκληρώνει το Assessment Quiz.

  β′) Ο `CourseAssessmentController` λαμβάνει τις απαντήσεις.

  γ′) Για κάθε ερώτηση του quiz (οι οποίες είναι συνδεδεμένες με συγκεκριμένα αρχικά lessons του Course μέσω του πεδίου `lesson_id` στον πίνακα `questions`), ελέγχεται η ορθότητα της απάντησης.

  δ′) Συγκεντρώνονται τα αποτελέσματα ανά `lesson_id`. Υπολογίζεται ένα ποσοστό επιτυχίας για τις ερωτήσεις που αντιστοιχούν σε κάθε αρχικό lesson.

  ε′) Το σύστημα διατρέχει τα lessons του Course με τη σειρά.

ς′) Εάν ένα lesson δεν είχε ερωτήσεις στο assessment quiz, ή εάν ο χρήστης δεν πέτυχε ένα προκαθορισμένο ποσοστό επιτυχίας (π.χ., 80%) στις ερωτήσεις που αντιστοιχούσαν σε αυτό, τότε αυτό το lesson προτείνεται ως το σημείο εκκίνησης.

ζ′) Εάν ο χρήστης περάσει επιτυχώς τις ερωτήσεις για όλα τα αρχικά lessons που καλύπτονταν από το assessment, του προτείνεται να ξεκινήσει από το πρώτο lesson (ως ασφαλής επιλογή) ή ενημερώνεται ότι φαίνεται να κατέχει το υλικό και μπορεί να προχωρήσει σε πιο προχωρημένες ενότητες.

- *Αποτέλεσμα:* Ο χρήστης λαμβάνει μια προσωποποιημένη πρόταση για το από ποιο μάθημα θα μπορούσε να ξεκινήσει, μαζί με την επιλογή να αγνοήσει την πρόταση και να ξεκινήσει από την αρχή.


Σχήμα 14: Πρόταση μαθήματος βάσει αξιολόγησης αρχικού επιπέδου.

2. **Πρόταση Ενισχυτικού Υλικού (Post-course Review Quiz):**

- *Ενεργοποίηση:* Μετά την ολοκλήρωση του Τελικού Επαναληπτικού Quiz ενός Course.
- *Διαδικασία:*

α′) Ο χρήστης ολοκληρώνει το Post-course Review Quiz.

β′) Ο `CourseReviewController` λαμβάνει και βαθμολογεί τις απαντήσεις.

γ′) Για κάθε λανθασμένη απάντηση, εντοπίζεται το `lesson_id` με το οποίο σχετίζεται η ερώτηση.

δ′) Για κάθε τέτοιο lesson, ανακτώνται από τη βάση δεδομένων οι σχετικές εξωτερικές πηγές υλικού (`external_resources`).

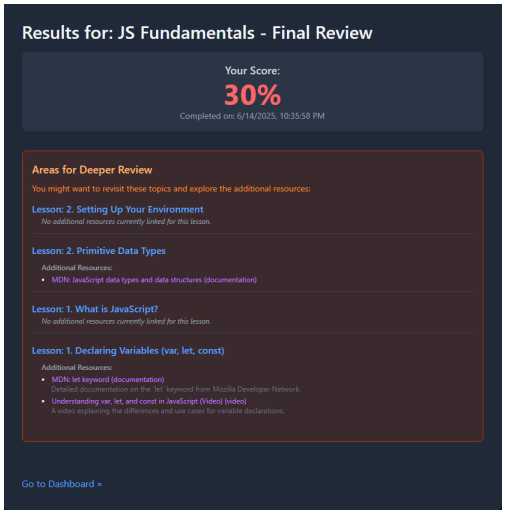
- *Αποτέλεσμα:* Στη σελίδα αποτελεσμάτων του quiz, μαζί με τις σωστές/λανθασμένες απαντήσεις, ο χρήστης βλέπει μια λίστα με τα μαθήματα που χρειάζονται επανάληψη και, για καθένα από αυτά, προτεινόμενους συνδέσμους προς εξωτερικές πηγές (άρθρα, βίντεο, τεκμηρίωση) για περαιτέρω μελέτη.

3. **Προσαρμογή Εμφάνισης Περιεχομένου (Visual vs. Reading Style):**

- *Ενεργοποίηση:* Ο χρήστης ορίζει την προτίμησή του (`preferred_learning_style`) στο Dashboard.
- *Διαδικασία:*

α′) Η προτίμηση του χρήστη είναι διαθέσιμη στο frontend (Vue component lessons/Show.vue) μέσω των global props του Inertia.

β′) Το component ελέγχει την τιμή του `user.preferred_learning_style` και την ύπαρξη ενσωματωμένου βίντεο (`lesson.video_embed_html`).




Σχήμα 15: Προτάσεις επανάληψης και ενισχυτικού υλικού.

γ′) Εάν η προτίμηση είναι ′visual′ και υπάρχει βίντεο, το μπλοκ του βίντεο εμφανίζεται πριν από το κειμενικό περιεχόμενο του μαθήματος.

δ′) Σε αντίθετη περίπτωση (προτίμηση ′reading′ ή δεν υπάρχει βίντεο), το κειμενικό περιεχόμενο εμφανίζεται πρώτο, και το βίντεο (αν υπάρχει) εμφανίζεται μετά, ως προαιρετικό.

- *Αποτέλεσμα:* Η σειρά παρουσίασης των διαφορετικών τύπων περιεχομένου προσαρμόζεται στην προτίμηση του χρήστη, με στόχο την καλύτερη δυνατή εμπλοκή.


Σχήμα 16: Προσαρμογή εμφάνισης περιεχομένου.

4. **Καθοδήγηση βάσει Μαθησιακού Στόχου (Learning Paths):**

- *Ενεργοποίηση:* Ο χρήστης επιλέγει μια Μαθησιακή Διαδρομή (`learning_path`) στο Dashboard.
- *Διαδικασία:*

α′) Η επιλεγμένη διαδρομή του χρήστη είναι γνωστή στο backend (DashboardController).

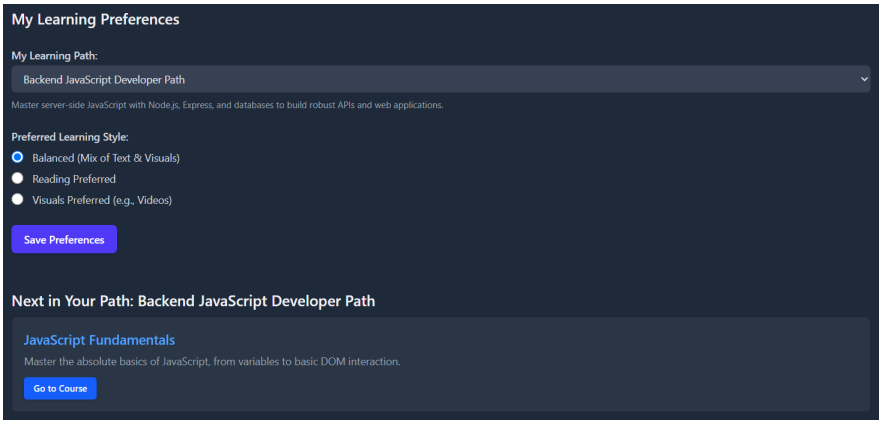
β′) Ο controller ανακτά τα Courses που ανήκουν σε αυτή τη διαδρομή, στη σωστή σειρά (βάσει του πεδίου `ορδερ` στον πίνακα `learning_path_course`).

γ′) Για κάθε Course στη διαδρομή, ελέγχεται η πρόοδος του χρήστη (ολοκληρωμένα lessons μέσω του `user_progress`).

δ′) Το πρώτο Course στη σειρά της διαδρομής για το οποίο ο χρήστης δεν έχει ολοκληρώσει όλα τα μαθήματα (ή ένα σημαντικό ποσοστό αυτών) προτείνεται ως το ¨Επόμενο Προτεινόμενο Course¨.

ε′) Αν όλα τα Courses της διαδρομής έχουν ολοκληρωθεί, εμφανίζεται ένα μήνυμα επιτυχίας.

- *Αποτέλεσμα:* Στο Dashboard, ο χρήστης βλέπει την τρέχουσα μαθησιακή του διαδρομή και μια σαφή πρόταση για το ποιο course να παρακολουθήσει στη συνέχεια, παρέχοντας δομή και κατεύθυνση.




Σχήμα 17: Καθοδήγηση μαθησιακής διαδρομής.

## 3.4 Σημαντικά Components

Η εφαρμογή δομείται γύρω από ένα σύνολο διακριτών components στο backend (κυρίως Laravel Controllers) και στο frontend (Vue Pages και επαναχρησιμοποιήσιμα Vue Components). Αυτή η προσέγγιση προάγει την οργάνωση, τη συντηρησιμότητα και την επεκτασιμότητα του κώδικα.

### 3.4.1 Backend Components (Laravel Controllers)

Οι παρακάτω είναι οι κύριοι controllers που διαχειρίζονται την επιχειρησιακή λογική της εφαρμογής στην πλευρά του server:

- `DashboardController`:

– *Ρόλος:* Υπεύθυνος για τη συγκέντρωση και παροχή των δεδομένων που απαιτούνται για την αρχική σελίδα (dashboard) του συνδεδεμένου χρήστη. Αυτό περιλαμβάνει τις προτιμήσεις του χρήστη (learning path, learning style), τις διαθέσιμες μαθησιακές διαδρομές, το επόμενο προτεινόμενο course βάσει της επιλεγμένης διαδρομής, και το ιστορικό πρόσφατων προσπαθειών σε quizzes.

- `CourseController`:

– *Ρόλος:* Διαχειρίζεται τις λειτουργίες που σχετίζονται με τους κύκλους μαθημάτων (Courses).

– *Βασικές Μέθοδοι:* `index()` για την εμφάνιση λίστας όλων των δημοσιευμένων courses, `show()` για την εμφάνιση των λεπτομερειών ενός συγκεκριμένου course (modules, lessons, quizzes ενότητας, link για assessment/review).

• `LessonController:`

– *Ρόλος:* Διαχειρίζεται την εμφάνιση των μεμονωμένων μαθημάτων (Lessons).
– *Βασικές Μέθοδοι:* `show()` την παρουσίαση του περιεχομένου ενός μαθήματος, της άσκησης, του αρχικού κώδικα και την προετοιμασία της διεπαφής με τον code editor.

• `QuizController:`

– *Ρόλος:* Διαχειρίζεται τη διεξαγωγή των τυπικών quizzes (module quizzes).
– *Βασικές Μέθοδοι:* `show()` την εμφάνιση των ερωτήσεων ενός quiz, `submit()` την παραλαβή των απαντήσεων, τη βαθμολόγηση, την αποθήκευση της προσπάθειας (QuizAttempt και QuizAnswer) και την εμφάνιση των αποτελεσμάτων.

• `UserProgressController:`

– *Ρόλος:* Διαχειρίζεται την καταγραφή της προόδου του χρήστη.
– *Βασικές Μέθοδοι:* `store()` την επισήμανση ενός μαθήματος (Lesson) ως ολοκληρωμένου από τον χρήστη, συνήθως μετά την επιτυχή ολοκλήρωση της άσκησης κώδικα.

• `CourseAssessmentController:`

– *Ρόλος:* Διαχειρίζεται τα quizzes αξιολόγησης αρχικού επιπέδου (Pre-course Assessment).
– *Βασικές Μέθοδοι:* `show()` για την εμφάνιση του assessment quiz, `submit()` για τη βαθμολόγηση και την παροχή πρότασης σχετικά με το από ποιο μάθημα να ξεκινήσει ο χρήστης.

• `CourseReviewController:`

– *Ρόλος:* Διαχειρίζεται τα τελικά επαναληπτικά quizzes ενός course (Post-course Review).
– *Βασικές Μέθοδοι:* `generate()` για τη δυναμική δημιουργία του quiz (βάσει προηγούμενων λανθασμένων απαντήσεων και νέων ερωτήσεων), `submit()` για τη βαθμολόγηση, την αποθήκευση της προσπάθειας και την παροχή προτάσεων για επανάληψη μαθημάτων και εξωτερικών πηγών.

• `RandomQuizController:`

– *Ρόλος:* Διαχειρίζεται τη δημιουργία και διεξαγωγή τυχαίων quizzes επανάληψης.
– *Βασικές Μέθοδοι:* `generate()` για την επιλογή τυχαίων ερωτήσεων από τα ολοκληρωμένα μαθήματα του χρήστη, `submit()` για τη βαθμολόγηση και την αποθήκευση της προσπάθειας.

• `UserPreferenceController:`

– *Ρόλος:* Διαχειρίζεται την ενημέρωση των μαθησιακών προτιμήσεων του χρήστη (learning style, learning path) από το Dashboard.
– *Βασικές Μέθοδοι:* `update()` για την αποθήκευση των νέων προτιμήσεων.

• `UserStatsController:`

– *Ρόλος:* Συγκεντρώνει και παρέχει τα δεδομένα για τη σελίδα στατιστικών του χρήστη.
– *Βασικές Μέθοδοι:* `show()` για την προετοιμασία των δεδομένων (learning streak, κατανομή σκορ quiz, δεδομένα για contribution graph).

• `ProfileController:`

– *Ρόλος:* Διαχειρίζεται τις βασικές λειτουργίες προφίλ χρήστη (επεξεργασία ονόματος, email, password). Η λειτουργικότητα για τις μαθησιακές προτιμήσεις μεταφέρθηκε στο

DashboardController και UserPreferenceController για ευκολότερη πρόσβαση από τον χρήστη.

### 3.4.2 Frontend Components (Vue Pages & Κύρια Components)

Η διεπαφή χρήστη (UI) υλοποιείται με Vue.js και Inertia.js. Οι παρακάτω είναι οι κύριες σελίδες (Vue Page components που βρίσκονται στο `resources/js/Pages/`):

- `Dashboard.vue`:
  - *Ρόλος:* Η αρχική σελίδα μετά τη σύνδεση του χρήστη. Εμφανίζει τις μαθησιακές προτιμήσεις, το επόμενο προτεινόμενο course, συνδέσμους προς λειτουργίες, και το ιστορικό quiz. Επιτρέπει την επεξεργασία των προτιμήσεων.
- `courses/Index.vue`:
  - *Ρόλος:* Εμφανίζει τη λίστα όλων των διαθέσιμων Courses.
- `courses/Show.vue`:
  - *Ρόλος:* Εμφανίζει τις λεπτομέρειες ενός Course, συμπεριλαμβανομένων των Modules, Lessons, και Quizzes του. Περιλαμβάνει συνδέσμους για Pre-course Assessment και Post-course Review Quiz.
- `lessons/Show.vue`:
  - *Ρόλος:* Η καρδιά της μαθησιακής εμπειρίας. Εμφανίζει το περιεχόμενο ενός Lesson, την άσκηση, τον Monaco code editor, και την περιοχή εξόδου. Διαχειρίζεται την client-side εκτέλεση κώδικα.
- `quizzes/Show.vue`:
  - *Ρόλος:* Παρέχει τη διεπαφή για τη διεξαγωγή ενός quiz. Εμφανίζει τις ερωτήσεις και διαχειρίζεται την υποβολή των απαντήσεων.
- `quizzes/Result.vue`:
  - *Ρόλος:* Εμφανίζει τα αποτελέσματα μετά την υποβολή ενός quiz. Περιλαμβάνει σκορ, ανατροφοδότηση, εξηγήσεις και προτάσεις.
- `stats/Show.vue`:
  - *Ρόλος:* Εμφανίζει τα οπτικοποιημένα στατιστικά προόδου του χρήστη (learning streak, σύνολο quizzes, πίτα αποτελεσμάτων, contribution graph). Χρησιμοποιεί vue-chartjs.

# 4 Υλοποίηση και Τεχνολογίες

Το παρόν κεφάλαιο παρέχει μια επισκόπηση των τεχνολογιών και των εργαλείων που αξιοποιήθηκαν για την ανάπτυξη του εκπαιδευτικού λογισμικού εκμάθησης JavaScript, καθώς και μια συνοπτική περιγραφή των βασικών βημάτων που ακολουθήθηκαν κατά τη διαδικασία της υλοποίησης.

## 4.1 Τεχνολογίες που Χρησιμοποιήθηκαν

Η ανάπτυξη της εφαρμογής βασίστηκε σε ένα σύνολο σύγχρονων τεχνολογιών και εργαλείων, τα οποία επιλέχθηκαν για την αποτελεσματικότητα, την ευελιξία και την υποστήριξη που προσφέρουν στην ανάπτυξη διαδικτυακών εφαρμογών. Οι κύριες τεχνολογίες που χρησιμοποιήθηκαν είναι:

**Backend:**

- **PHP (Έκδοση 8.4):** Η γλώσσα προγραμματισμού στην οποία βασίζεται το backend της εφαρμογής.
- **Laravel Framework (Έκδοση 12):** Ένα δημοφιλές PHP framework που ακολουθεί το αρχιτεκτονικό πρότυπο Model-View-Controller (MVC). Παρέχει πληθώρα ενσωματωμένων λειτουργιών για routing, ORM (Eloquent), αυθεντικοποίηση, διαχείριση session, και πολλά άλλα, επιταχύνοντας σημαντικά την ανάπτυξη.
- **Composer:** Διαχειριστής πακέτων (dependency manager) για τη PHP, χρησιμοποιήθηκε για την εγκατάσταση και διαχείριση των βιβλιοθηκών του Laravel και άλλων PHP πακέτων.

**Frontend:**

- **JavaScript (ES6+):** Η γλώσσα προγραμματισμού για τη δημιουργία της διαδραστικότητας στην πλευρά του client.
- **Vue.js 3:** Ένα προοδευτικό JavaScript framework για τη δημιουργία διεπαφών χρήστη. Χρησιμοποιήθηκε για την ανάπτυξη των frontend components της εφαρμογής.
- **Inertia.js:** Λειτουργεί ως ¨γέφυρα' μεταξύ του Laravel backend και του Vue.js frontend, επιτρέποντας την ανάπτυξη σύγχρονων μονοσέλιδων εφαρμογών (SPA-like experience) χωρίς την ανάγκη δημιουργίας ξεχωριστού API.
- **Vite:** Ένα σύγχρονο εργαλείο frontend build που παρέχει εξαιρετικά γρήγορο Hot Module Replacement (HMR) κατά την ανάπτυξη και βελτιστοποιημένα builds για παραγωγή.
- **Tailwind CSS:** Ένα utility-first CSS framework που χρησιμοποιήθηκε για τη γρήγορη και ευέλικτη διαμόρφωση της εμφάνισης της εφαρμογής.
- **Monaco Editor (μέσω @monaco-editor/loader):** Ο επεξεργαστής κώδικα που τροφοδοτεί το VS Code, ενσωματώθηκε για την παροχή μιας πλούσιας εμπειρίας επεξεργασίας κώδικα JavaScript εντός της πλατφόρμας.
- **vue-chartjs:** Βιβλιοθήκη JavaScript για τη δημιουργία των διαδραστικών γραφημάτων για την οπτικοποίηση των στατιστικών προόδου.
- **npm (Node Package Manager):** Διαχειριστής πακέτων για τη JavaScript, χρησιμοποιήθηκε για την εγκατάσταση και διαχείριση των frontend βιβλιοθηκών και εργαλείων.

**Βάση Δεδομένων:**

- **SQLite:** Μια ελαφριά, file-based βάση δεδομένων που είναι εύκολη στη χρήση για τοπική ανάπτυξη και δοκιμές.

**Εργαλεία Ανάπτυξης:** PHP Storm, Git, Browser Developer Tools, Tinker.

## 4.2 Συνοπτικά τα Βασικά Βήματα Υλοποίησης

Η υλοποίηση της εκπαιδευτικής πλατφόρμας ακολούθησε μια επαυξητική προσέγγιση, όπου οι βασικές λειτουργίες αναπτύχθηκαν πρώτα και στη συνέχεια προστέθηκαν σταδιακά πιο σύνθετα χαρακτηριστικά και οι μηχανισμοί εξατομίκευσης. Τα κύρια βήματα της διαδικασίας υλοποίησης μπορούν να συνοψιστούν ως εξής:

1. Αρχική Ρύθμιση του Project (Project Setup): Δημιουργία νέου Laravel project. Εγκατάσταση και παραμετροποίηση του Laravel Breeze (ή Jetstream) με το Vue.js και Inertia.js stack για την παροχή βασικής αυθεντικοποίησης και αρχικής δομής frontend. Ρύθμιση της βάσης δεδομένων (αρχικά SQLite). Εγκατάσταση του Tailwind CSS για το styling.
2. Σχεδιασμός και Υλοποίηση Βασικών Μοντέλων Βάσης Δεδομένων: Δημιουργία των αρχικών migrations και Eloquent models για τις κύριες οντότητες: User, Course, Module, Lesson. Καθορισμός των αρχικών σχέσεων μεταξύ αυτών των μοντέλων.
3. Υλοποίηση Βασικής Λειτουργικότητας Μαθημάτων: Δημιουργία των CourseController και LessonController. Ανάπτυξη των Vue page components (Courses/Index.vue, Courses/Show.vue, Lessons/Show.vue) για την εμφάνιση της λίστας των courses, των λεπτομερειών ενός course (modules  lessons), και του περιεχομένου ενός lesson. Δημιουργία αρχικών seeders για την εισαγωγή δοκιμαστικού περιεχομένου courses, modules, lessons.
4. Ενσωμάτωση Επεξεργαστή Κώδικα και Εκτέλεσης Ασκήσεων: Προσθήκη των απαραίτητων πεδίων (assignment, initial_code, expected_output) στο Lesson model και migration. Ενσωμάτωση του Monaco Editor στο Lessons/Show.vue. Υλοποίηση της client-side εκτέλεσης JavaScript κώδικα και της εμφάνισης του output/errors. Υλοποίηση της αυτόματης ολοκλήρωσης μαθήματος (UserProgressController) βάσει της επιτυχούς εκτέλεσης της άσκησης.
5. Ανάπτυξη Συστήματος Quiz: Σχεδιασμός και υλοποίηση των models Quiz, Question, QuizAttempt, QuizAnswer. Επέκταση των seeders για τη δημιουργία module quizzes και ερωτήσεων διαφόρων τύπων (multiple choice, true/false, fill_blank). Δημιουργία του QuizController για τη διαχείριση της εμφάνισης και υποβολής των module quizzes. Ανάπτυξη των Vue components quizzes/Show.vue και quizzes/Result.vue.
6. Υλοποίηση Μηχανισμών Εξατομικευμένης Μάθησης:
   - **Pre-course Assessment:** Τροποποίηση models και seeders για υποστήριξη assessment quizzes. Δημιουργία CourseAssessmentController και των αντίστοιχων Vue pages.
   - **Learning Style & Path Preferences:** Τροποποίηση User model και migration, δημιουργία LearningPath model και pivot table learning_path_course. Δημιουργία seeders για learning paths. Ενημέρωση DashboardController και UserPreferenceController για διαχείριση προτιμήσεων. Ενημέρωση Dashboard.vue για επιλογή προτιμήσεων και lessons/Show.vue για προσαρμογή περιεχομένου.
   - **Post-course Review Quiz & External Resources:** Τροποποίηση models για final review quizzes, δημιουργία ExternalResource model. Δημιουργία CourseReviewController και ενημέρωση seeders/Vue components.
   - **Random Review Quiz:** Δημιουργία RandomQuizController και σχετικών Vue components. Τροποποίηση QuizAttempt model για υποστήριξη random quiz attempts.
7. Ανάπτυξη Σελίδας Στατιστικών Χρήστη: Υλοποίηση μεθόδων στο User model για υπολογισμό learning streak, κατανομής σκορ quiz, και δεδομένων για contribution graph.

Δημιουργία UserStatsController. Εγκατάσταση vue-chartjs και ανάπτυξη του profile/Statistics.vue για την οπτικοποίηση των δεδομένων.

8. Συνεχής Δοκιμή και Βελτίωση: Καθ΄ όλη τη διάρκεια της ανάπτυξης, πραγματοποιούνταν λειτουργικές δοκιμές για την επαλήθευση της ορθής λειτουργίας των νέων χαρακτηριστικών και τη διόρθωση τυχόν σφαλμάτων.

# 5 Συμπεράσματα και Μελλοντικές Επεκτάσεις

Το παρόν κεφάλαιο συνοψίζει τα βασικά συμπεράσματα και παραθέτει προτάσεις για μελλοντικές βελτιώσεις.

## 5.1 Επίτευξη Στόχων

Η ανάπτυξη της παρούσας εκπαιδευτικής πλατφόρμας στόχευε στη δημιουργία ενός ολοκληρωμένου και εξατομικευμένου περιβάλλοντος για την εκμάθηση της JavaScript. Οι κύριοι στόχοι, όπως η παροχή δομημένου υλικού, η διαδραστική εξάσκηση, η αυτοαξιολόγηση, η εξατομίκευση και η παρακολούθηση προόδου, έχουν επιτευχθεί σε σημαντικό βαθμό, προσφέροντας ένα λειτουργικό και πλούσιο σε χαρακτηριστικά περιβάλλον.

## 5.2 Προτάσεις για Μελλοντικές Επεκτάσεις

- **Πιο Προηγμένη Αξιολόγηση Κώδικα:** Server-side εκτέλεση κώδικα, αυτόματη βαθμολόγηση με πολλαπλά test cases.
- **Εμπλουτισμός Περιεχομένου:** Περισσότερα Courses, νέοι τύποι ερωτήσεων, mini-projects.
- **Ενίσχυση Εξατομίκευσης:** Πιο εξελιγμένοι αλγόριθμοι, δυναμική ρύθμιση δυσκολίας, 'Targeted Practice¨.
- **Gamification και Κοινωνική Μάθηση:** Πόντοι, badges, leaderboards, forums, σχόλια.
- **Βελτιώσεις UI/UX:** Αναζήτηση περιεχομένου, καλύτερη προσβασιμότητα.
- **Τεχνικές Βελτιώσεις:** Unit/Integration Tests, βελτιστοποιήσεις απόδοσης.