Τεχνολογίες και Προγραμματισμός Εφαρμογών στον Ιστό

1η Σειρά Ασκήσεων (ομαδική – 2 ατόμων)

Επίκουρος Καθηγητής Γιώργος Στεργιόπουλος

Χειμερινό Εξάμηνο 2025 – 2026

Περιεχόμενα

1 Εισαγωγή

Ομαδική εργασία: ομάδες των δύο (2) φοιτητών
Χειμερινό Εξάμηνο 2025–2026

Η εργασία έχει ως αντικείμενο τον σχεδιασμό και την υλοποίηση μιας υπηρεσίας ηλεκτρονικής μάθησης για μαθήματα Πληροφορικής, η οποία θα παρουσιάζει εκπαιδευτικό υλικό (βιβλία, βίντεο, θεματικές ενότητες), θα υποστηρίζει εγγραφή χρηστών και θα οργανώνεται με τρόπο που να μπορεί να επεκταθεί σε πλήρη web εφαρμογή με backend. Η εργασία αποτελείται από δύο μέρη.

Το Μέρος Α είναι υποχρεωτικό και αντιστοιχεί σε 20% της τελικής βαθμολογίας του μαθήματος. Το Μέρος Β είναι προαιρετικό, επεκτείνει το Μέρος Α σε πλήρη αρχιτεκτονική MEAN, αντιστοιχεί σε 40% της τελικής βαθμολογίας, και βαθμολογείται μόνο εφόσον υλοποιηθεί και παρουσιαστεί με επίδειξη.

Και τα δύο μέρη εργασίας θα εξεταστούν/παρουσιαστούν εργαστηριακά.

Η βαθμολογία καταχωρείται ως εξής:

- Αν καταθέσει μια ομάδα μόνο την πρώτη έργασία, η βαθμολογία υπολογίζεται ως εξής:
  (Βαθμός γραπτού) * 0,8 + (βαθμός εργασίας Ι) *0,2
- Αν καταθέσει μια ομάδα καταθέσει και τις δύο εργασίες, η βαθμολογία υπολογίζεται ως εξής:
  (Βαθμός γραπτού) * 0,4 + (βαθμός εργασίας ΙΙ) * 0,4 + (βαθμός εργασίας Ι) *0,2

2 ΕΡΓΑΣΙΑ Ι – ΥΠΟΧΡΕΩΤΙΚΗ (20%)

Στόχος είναι να υλοποιήσετε έναν πλήρως λειτουργικό, στατικό αλλά δυναμικά συμπεριφερόμενο ιστότοπο, ο οποίος θα παρουσιάζει την υπηρεσία e-learning και θα επιτρέπει τη συλλογή και βασική επεξεργασία δεδομένων χρηστών αποκλειστικά στον browser. Θα χρησιμοποιήσετε HTML5 για την δομή, CSS3 για την εμφάνιση και layout, και JavaScript σε εξωτερικά αρχεία για τη λογική και την αλληλεπίδραση. Δεν επιτρέπεται η χρήση server-side κώδικα ή frameworks στο Μέρος Α. Ο στόχος είναι να σχεδιάσετε προσεκτικά τη δομή του περιεχομένου, να εφαρμόσετε σύγχρονες τεχνικές responsive design και να οργανώσετε JavaScript κώδικα που να καλύπτει ρεαλιστικές ανάγκες μιας υπηρεσίας e-learning.

Ξεκινήστε από τον σχεδιασμό της δομής του ιστοτόπου σας. Ορίστε ως βασική οντότητα την «Υπηρεσία Ηλεκτρονικής Μάθησης», η οποία προσφέρει κατηγορίες μαθημάτων Πληροφορικής (π.χ. προγραμματισμός, δίκτυα, ασφάλεια, βάσεις δεδομένων), σελίδες με λίστες βιβλίων και βίντεο, σελίδα αναλυτικής παρουσίασης ενός μαθήματος ή ενός βιβλίου, σελίδα παρουσίασης της ομάδας ανάπτυξης και σελίδα εγγραφής χρήστη. Η αρχική σελίδα θα πρέπει να περιγράφει καθαρά

τι προσφέρει η πλατφόρμα, να περιλαμβάνει βασικό κείμενο εισαγωγής, ενότητες με σύντομες περιγραφές κατηγοριών, τμήμα με προτεινόμενα μαθήματα ή βιβλία και συνδέσμους προς τις υπόλοιπες σελίδες. Χρησιμοποιήστε σημασιολογικά στοιχεία HTML5 (header, nav, main, section, article, footer) ώστε να είναι σαφές πού βρίσκεται κάθε μέρος του περιεχομένου και οργανώστε τη δομή εξαρχής με τρόπο που θα υποστηρίζει την επόμενη επέκταση σε δυναμική εφαρμογή.

Δώστε ιδιαίτερη προσοχή στην πλοήγηση. Υλοποιήστε κοινό μενού πλοήγησης που εμφανίζεται σε όλες τις σελίδες με τα ίδια στοιχεία, ώστε ο χρήστης να μπορεί ανά πάσα στιγμή να μετακινηθεί μεταξύ αρχικής σελίδας, σελίδας κατηγοριών, σελίδων λίστας μαθημάτων ή βιβλίων, σελίδας λεπτομερειών, σελίδας «Ποιοι είμαστε» και σελίδας εγγραφής. Φροντίστε το μενού να είναι σαφές, απλό και σταθερό στην εμφάνισή του, με ομοιόμορφο στυλ. Για κινητές συσκευές, το μενού πρέπει να προσαρμόζεται ώστε να καταλαμβάνει ελάχιστο χώρο, με χρήση JavaScript για εμφάνιση ή απόκρυψη (π.χ. menu icon που ανοίγει και κλείνει την πλοήγηση).

Υλοποιήστε το responsive design με προσέγγιση mobile-first. Αρχικά διαμορφώστε τη διάταξη σε μονή στήλη κατάλληλη για μικρές οθόνες. Βεβαιωθείτε ότι τα στοιχεία (κεφαλίδα, λογότυπο, μενού, τίτλοι μαθημάτων, λίστες βιβλίων, κουμπιά) εμφανίζονται σε λογική σειρά από πάνω προς τα κάτω. Στη συνέχεια χρησιμοποιήστε media queries για να προσαρμόσετε τη διάταξη σε δύο στήλες για tablet και τρεις στήλες για οθόνες επιτραπέζιων υπολογιστών, με τρόπο που να γίνεται σαφώς αντιληπτή η αλλαγή στη δομή του περιεχομένου. Χωρίστε το περιεχόμενο σε διακριτές ενότητες (π.χ. περιοχή κατηγοριών, περιοχή προτεινόμενων μαθημάτων, πλαϊνό πλαίσιο για αναζήτηση ή φίλτρα) και, στις μεγαλύτερες αναλύσεις, τοποθετήστε τις ενότητες αυτές σε ξεχωριστές στήλες χρησιμοποιώντας Flexbox ή CSS Grid. Διασφαλίστε ότι τα στοιχεία δεν αλληλεπικαλύπτονται, ότι τα κείμενα παραμένουν ευανάγνωστα και ότι δεν δημιουργούνται οριζόντιες μπάρες κύλισης εκεί όπου δεν χρειάζεται.

Βελτιστοποιήστε τις εικόνες έτσι ώστε να συμπεριφέρονται σωστά σε όλες τις αναλύσεις. Ορίστε κανόνες ώστε οι εικόνες να προσαρμόζονται στο πλάτος του container που τις περιέχει και, όπου κρίνεται απαραίτητο, χρησιμοποιήστε χαρακτηριστικά όπως srcset για να παρέχετε διαφορετικές εκδοχές της ίδιας εικόνας για διαφορετικά μεγέθη οθόνης. Αποφύγετε την παραμόρφωση των εικόνων και βεβαιωθείτε ότι η φόρτωσή τους είναι λογική για ένα περιβάλλον e-learning (όχι υπερβολικά μεγάλα αρχεία, όχι άσχετες εικόνες).

Ο **κώδικας JavaScript** θα τοποθετηθεί εξολοκλήρου σε ένα ή περισσότερα εξωτερικά αρχεία, τα οποία θα συνδέονται με τις HTML σελίδες. Ξεκινήστε ορίζοντας μια βασική δομή, π.χ. ορισμό ενός πίνακα αντικειμένων για τα μαθήματα, με πεδία όπως τίτλος, κατηγορία, επίπεδο, σύντομη περιγραφή και ενδεχομένως ενδείξεις δυσκολίας ή διαθεσιμότητας. Χρησιμοποιήστε JavaScript για να γεμίσετε δυναμικά τις λίστες μαθημάτων και βιβλίων σε συγκεκριμένα τμήματα της σελίδας, αντί να γράφετε όλες τις εγγραφές στατικά σε HTML. Με αυτόν τον τρόπο θα εξασκηθείτε στην επεξεργασία δεδομένων και στην τροποποίηση του DOM.

Υλοποιήστε βασική λειτουργικότητα φιλτραρίσματος και αναζήτησης από την πλευρά του client. Για παράδειγμα, επιτρέψτε στον χρήστη να επιλέγει κατηγορία μαθημάτων από dropdown και ενημερώστε δυναμικά τη λίστα εμφανιζόμενων μαθημάτων χωρίς ανανέωση σελίδας. Μπορείτε να προσθέσετε απλά κριτήρια, όπως φιλτράρισμα ανά επίπεδο δυσκολίας ή εμφάνιση μόνο διαθέσιμων βιβλίων. Φροντίστε ο κώδικας να είναι οργανωμένος σε συναρτήσεις που διαβάζουν τις επιλογές του χρήστη, επεξεργάζονται τα δεδομένα και ενημερώνουν τα στοιχεία HTML.

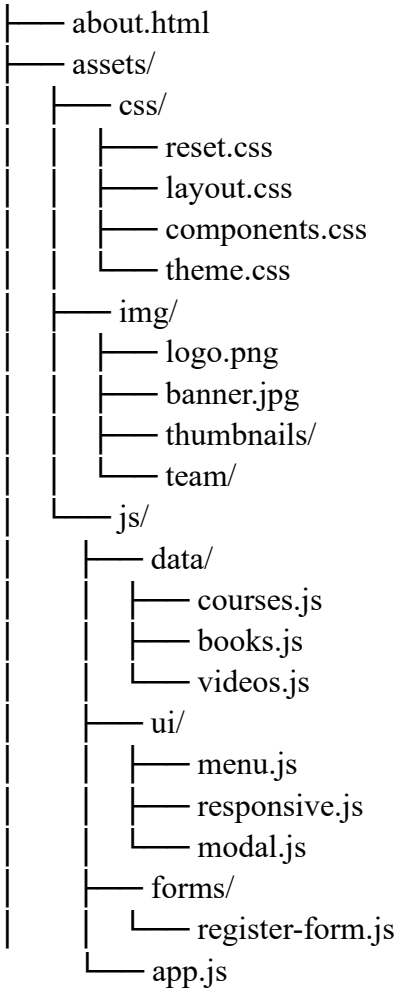
Δώστε ιδιαίτερη σημασία στη σελίδα εγγραφής χρήστη. Ο σχεδιασμός της φόρμας πρέπει να είναι πλήρης και ρεαλιστικός. Περιλάβετε πεδία για βασικά στοιχεία χρήστη, στοιχεία λογαριασμού, επιλογές ενδιαφερόντων και μια σύντομη ενότητα με ερωτήσεις που θα βοηθούσαν την πλατφόρμα να προτείνει μαθήματα (π.χ. προτιμώμενες θεματικές, εμπειρία, στόχοι). **Χρησιμοποιήστε κατάλληλους τύπους input** (email, password, date, radio, checkbox) και ιδιότητες επικύρωσης HTML5 (required, pattern) για να ορίσετε βασικούς ελέγχους. Προσθέστε JavaScript επικύρωση η οποία θα ελέγχει αν τα πεδία είναι σωστά συμπληρωμένα, αν οι κωδικοί ταυτίζονται, αν η διεύθυνση email έχει αποδεκτή μορφή και αν πληρούνται πιθανές προϋποθέσεις ηλικίας ή πολυπλοκότητας κωδικού. Πριν την τελική "υποβολή", χρησιμοποιήστε JavaScript για να συλλέξετε τα δεδομένα από τη φόρμα, να τα επεξεργαστείτε σε αντικείμενο και να εμφανίσετε μια σύνοψη στον χρήστη (π.χ. σε ειδική ενότητα ή αναδυόμενο πλαίσιο) ώστε να επιβεβαιώσει τα στοιχεία του. Δεν θα αποστείλετε τα δεδομένα σε server, αλλά πρέπει να φαίνεται καθαρά ότι τα επεξεργάζεστε στην πλευρά του client. Μπορείτε να χρησιμοποιήσετε localStorage για να αποθηκεύετε προσωρινά την πρόοδο συμπλήρωσης.

Ο κώδικας σας πρέπει να είναι καθαρός και οργανωμένος. Διαχωρίστε τη λογική διαχείρισης DOM, τα δεδομένα και τους ελέγχους. Αποφύγετε την επανάληψη κώδικα, χρησιμοποιήστε σαφή ονόματα μεταβλητών και συναρτήσεων και φροντίστε η υλοποίηση να είναι επαρκώς πολύπλοκη ώστε να απαιτεί ουσιαστικό προγραμματισμό, δοκιμή και βελτίωση από την ομάδα. Η εργασία πρέπει να παραδίδεται σε δομημένο φάκελο με κεντρικό index.html, ξεχωριστό φάκελο για CSS και JavaScript, σωστές σχετικές διαδρομές και δυνατότητα εκτέλεσης τοπικά χωρίς πρόσθετη ρύθμιση.

Ενδεικτική δομή αρχείων για Μέρος Α

Ο φάκελος του πρώτου μέρους πρέπει να μπορεί να ανοιχτεί απευθείας σε browser (χωρίς build, χωρίς npm). Η δομή μπορεί να είναι η εξής:

```
project/
└── part-a-frontend/
    ├── index.html
    ├── courses.html
    ├── course-details.html
    ├── books.html
    ├── register.html
```



Καθοδήγηση για τους φοιτητές με βάση τη δομή:

Το index.html αποτελεί το σημείο εισόδου και πρέπει να περιλαμβάνει το κύριο μενού, βασική παρουσίαση της πλατφόρμας, ενότητες με προτεινόμενα μαθήματα και συνδέσμους προς τις υπόλοιπες σελίδες. Τα courses.html, books.html παρουσιάζουν λίστες από περιεχόμενο οργανωμένο σε κατηγορίες. Το course-details.html παρουσιάζει πλήρη πληροφορία για ένα μάθημα ή βιβλίο, χρησιμοποιώντας τα ίδια δομικά στοιχεία που θα χρησιμοποιηθούν αργότερα και στο Μέρος Β. Το register.html περιέχει την αναλυτική φόρμα εγγραφής χρήστη με όλα τα απαιτούμενα πεδία. Το about.html παρουσιάζει την ομάδα, τα στοιχεία επικοινωνίας και μία σύντομη περιγραφή της «υπηρεσίας».

Στον φάκελο assets/css ορίζονται διακριτά αρχεία. Το reset.css φροντίζει να εξουδετερώσει τις προεπιλεγμένες μορφοποιήσεις των browsers. Το layout.css ορίζει το grid / flexbox layout, τη συμπεριφορά των στηλών και το responsive σπάσιμο για mobile, tablet και desktop. Το components.css περιγράφει επαναχρησιμοποιήσιμα στοιχεία (καρτέλες μαθημάτων, κουμπιά, κάρτες βιβλίων, banner). Το theme.css συγκεντρώνει χρώματα, γραμματοσειρές και βασικές μεταβλητές, ώστε η αισθητική να είναι συνεπής και να μπορεί να τροποποιηθεί κεντρικά.

Οι φοιτητές πρέπει να χρησιμοποιήσουν media queries στο layout.css για: μονή στήλη σε μικρές οθόνες, δύο στήλες σε tablet, τρεις στήλες σε desktop, με ξεκάθαρη αναδιάταξη των ενοτήτων. Η HTML πρέπει να είναι δομημένη σε λογικές ενότητες (sections, articles) έτσι ώστε η CSS να μπορεί να μετακινεί blocks μεταξύ στηλών χωρίς να αλλάζει η σειρά στο HTML.

Στον φάκελο assets/img τοποθετούνται μόνο οι εικόνες που πράγματι χρησιμοποιούνται. Προτείνεται κάποια βασική ιεραρχία (π.χ. thumbnails για μαθήματα, team για φωτογραφίες ομάδας), ώστε οι φοιτητές να εξασκηθούν σε καθαρή οργάνωση πόρων. Οι εικόνες πρέπει να δηλώνονται με responsive συμπεριφορά (max-width: 100% κ.λπ.) και όπου είναι εφικτό, να χρησιμοποιείται srcset για διαφορετικά μεγέθη.

Στον φάκελο assets/js γίνεται ο πλήρης διαχωρισμός λογικής. Τα αρχεία στον υποφάκελο data περιέχουν στατικά δεδομένα σε μορφή πινάκων/αντικειμένων (π.χ. courses, books, videos), τα οποία θα φορτώνονται δυναμικά στις σελίδες αντί για σκληροκωδικομένη HTML. Τα αρχεία στον υποφάκελο ui διαχειρίζονται καθαρά την εμφάνιση: το menu.js υλοποιεί το mobile menu (άνοιγμα/κλείσιμο, ARIA attributes), το responsive.js μπορεί να χειριστεί απλές προσαρμογές στο DOM με βάση το μέγεθος οθόνης αν χρειαστεί, το modal.js αναλαμβάνει διαλόγους/αναδυόμενα (π.χ. λεπτομέρειες μαθήματος). Ο φάκελος forms περιλαμβάνει τη λογική επικύρωσης για τη φόρμα εγγραφής, όπως έλεγχο μορφής email, πολυπλοκότητας κωδικού, ταύτισης πεδίων, επιλογής υποχρεωτικών πεδίων. Το app.js λειτουργεί ως σημείο εκκίνησης: εντοπίζει σε ποια σελίδα βρισκόμαστε, αρχικοποιεί το μενού, φορτώνει τα δεδομένα, ρυθμίζει event listeners και συνδέει όλα τα παραπάνω.

3 ΕΡΓΑΣΙΑ ΙΙ – Client-Server Mean Stack - ΠΡΟΑΙΡΕΤΙΚΗ (40%)

Στο Μέρος Β θα επεκτείνετε την υλοποίηση του Μέρος Α σε πλήρη client–server εφαρμογή βασισμένη στην αρχιτεκτονική MEAN. Διατηρείτε τη λογική της πλατφόρμας e-learning, αλλά πλέον τα δεδομένα χρηστών και μαθημάτων δεν είναι στατικά μέσα στον κώδικα. Θα υλοποιήσετε backend με **Node.js και Express**, θα αποθηκεύετε δεδομένα σε **MongoDB** και θα συνδέετε τον client με τον server μέσω REST API. Ο στόχος είναι να αποκτήσετε πρακτική εμπειρία στον σχεδιασμό και την κατανόηση μιας σύγχρονης web εφαρμογής.

Ξεκινήστε δημιουργώντας ένα έργο server με **Node.js**. Χρησιμοποιήστε npm για την αρχικοποίηση του έργου και οργανώστε τον κώδικα σε αρχεία για την εκκίνηση του server, τον ορισμό των διαδρομών (routes) και τη διαχείριση των δεδομένων. Εγκαταστήστε και χρησιμοποιήστε Express για να υλοποιήσετε **REST endpoints**. Ορίστε σαφείς διαδρομές για βασικές οντότητες: για παράδειγμα endpoints για εγγραφή χρήστη, ανάκτηση λίστας μαθημάτων, ανάκτηση λεπτομερειών μαθήματος, καταχώριση εγγραφής σε μάθημα, καταχώριση απλής αξιολόγησης. Για κάθε endpoint, αποφασίστε και εφαρμόστε κατάλληλες HTTP μεθόδους και

κωδικούς κατάστασης και χειριστείτε περιπτώσεις σφαλμάτων, όπως μη έγκυρα δεδομένα ή απόπειρα πρόσβασης σε ανύπαρκτο πόρο.

Στη συνέχεια συνδέστε τον server με MongoDB. Ορίστε σχήματα δεδομένων που να ταιριάζουν στη λογική της εφαρμογής σας, π.χ. σχήμα για χρήστες με βασικά στοιχεία και προτιμήσεις, σχήμα για μαθήματα ή βιβλία με κατηγορία, τίτλο, περιγραφή και διαθεσιμότητα, σχήμα για εγγραφές ή αξιολογήσεις. Επιλέξτε αν θα χρησιμοποιήσετε επίσημο driver ή ORM όπως Mongoose, αλλά σε κάθε περίπτωση τεκμηριώστε στον κώδικα σας τον τρόπο με τον οποίο δημιουργείται η σύνδεση με τη βάση, πώς γίνονται τα ερωτήματα και πώς γίνεται ο χειρισμός σφαλμάτων. Βεβαιωθείτε ότι μπορείτε να εκκινήσετε τον server και να εκτελέσετε βασικά αιτήματα προς τα endpoints σας.

Για τον client, ξεκινήστε από το αποτέλεσμα του Μέρος Α και προσαρμόστε το ώστε να επικοινωνεί με το **REST API**. Χρησιμοποιήστε **npm** και **Vite** για να ρυθμίσετε το περιβάλλον ανάπτυξης του client, έτσι ώστε τα αρχεία σας να εξυπηρετούνται και να γίνεται εύκολα η δόμηση του έργου. Μπορείτε να χρησιμοποιήσετε Angular ώστε να προσεγγίσετε την πλήρη λογική MEAN ή να οργανώσετε modular JavaScript με σαφή διαχωρισμό αρχείων. Το frontend πρέπει πλέον να αντλεί τα δεδομένα μαθημάτων από το backend με αιτήματα fetch ή αντίστοιχη βιβλιοθήκη, να εμφανίζει δυναμικά τις λίστες μαθημάτων και τις λεπτομέρειες, και η φόρμα εγγραφής χρήστη να αποστέλλει τα δεδομένα της ως JSON στο κατάλληλο endpoint του server. Φροντίστε να εμφανίζονται μηνύματα επιτυχίας ή αποτυχίας ανάλογα με την απόκριση του server και να διαχειρίζεστε βασικά σενάρια σφάλματος (π.χ. αδυναμία επικοινωνίας με τον server, μη έγκυρα δεδομένα).

Καθοδηγήστε τη σχεδίασή σας με τρόπο ώστε ο κώδικας να είναι χωρισμένος σε λογικές ενότητες. Η πλευρά του client δεν πρέπει να περιέχει hardcoded δεδομένα εκεί όπου αυτά προέρχονται από την βάση. Η πλευρά του server δεν πρέπει να περιέχει HTML, αλλά μόνο λογική API. Τεκμηριώστε σε σύντομο αρχείο οδηγιών πώς εγκαθίσταται το έργο (π.χ. εντολές npm install, ρυθμίσεις περιβάλλοντος για τη βάση, εντολές εκκίνησης server και client), ώστε ο αξιολογητής να μπορεί να αναπαράγει την εκτέλεση χωρίς πρόσθετη διερεύνηση.

Ενδεικτική δομή αρχείων για Μέρος Β

Το δεύτερο μέρος πρέπει να κρατήσει σαφή διάκριση ανάμεσα σε server και client. Η προτεινόμενη δομή:

```
project/
└── part-b-mean/
    ├── server/
    │   ├── package.json
    │   ├── .env.example
    │   ├── src/
    │   │   ├── app.js
```

```
│   │   ├── config/
│   │   │   └── db.js
│   │   ├── models/
│   │   │   ├── User.js
│   │   │   ├── Course.js
│   │   │   ├── Enrollment.js
│   │   │   └── Review.js
│   │   ├── routes/
│   │   │   ├── auth.routes.js
│   │   │   ├── courses.routes.js
│   │   │   ├── users.routes.js
│   │   │   └── enrollments.routes.js
│   │   ├── controllers/
│   │   │   ├── auth.controller.js
│   │   │   ├── courses.controller.js
│   │   │   ├── users.controller.js
│   │   │   └── enrollments.controller.js
│   │   └── middleware/
│   │       ├── validateRequest.js
│   │       └── errorHandler.js
│   └── README-server.md
├── client/
│   ├── package.json
│   ├── vite.config.js
│   ├── index.html
│   ├── src/
│   │   ├── main.ts
│   │   ├── app/
│   │   │   ├── app.component.ts
│   │   │   ├── app.component.html
│   │   │   └── app.routes.ts
│   │   ├── pages/
│   │   │   ├── home/
│   │   │   ├── courses/
│   │   │   ├── course-details/
│   │   │   ├── register/
│   │   │   └── about/
│   │   ├── components/
│   │   │   ├── course-card/
│   │   │   └── navbar/
│   │   └── services/
```

```
│   │       └── api.service.ts
│   └── README-client.md
└── README-part-b.md
```

Καθοδήγηση για τους φοιτητές με βάση τη δομή:
Στον φάκελο server η package.json περιέχει τα dependencies (Express, Mongoose ή επίσημο MongoDB driver, dotenv, cors κ.λπ.) και scripts όπως npm run dev και npm start. Το app.js ρυθμίζει το Express, φορτώνει τα routes, ενεργοποιεί το JSON parsing, το CORS και τον χειρισμό σφαλμάτων. Το db.js στον φάκελο config υλοποιεί τη σύνδεση με τη MongoDB χρησιμοποιώντας μεταβλητές περιβάλλοντος από .env. Τα models ορίζουν τα σχήματα και τη χαρτογράφηση των δεδομένων (χρήστες, μαθήματα, εγγραφές, αξιολογήσεις). Τα routes περιγράφουν καθαρά τα REST endpoints (π.χ. GET /api/courses, GET /api/courses/:id, POST /api/users, POST /api/enrollments) ενώ τα controllers περιέχουν τη λογική κάθε endpoint, με καθαρό διαχωρισμό από τη διαδρομή. Ο φάκελος middleware χρησιμοποιείται για επικύρωση εισόδου (validateRequest) και κεντρικό χειρισμό σφαλμάτων (errorHandler).

Το .env.example δείχνει ξεκάθαρα ποια στοιχεία πρέπει να ρυθμίσει ο αξιολογητής (π.χ. MONGODB_URI, PORT). Το README-server.md εξηγεί τα βήματα: εγκατάσταση dependencies, ρύθμιση .env, εντολές εκκίνησης, βασικά endpoints.

Στον φάκελο client, η χρήση Vite και, προαιρετικά, Angular (ή modular JS) επιβάλλει οργανωμένη δομή. Η package.json περιέχει τα dependencies (Angular + Vite plugin ή απλό Vite setup) και scripts (npm run dev, npm run build). Το vite.config.js ρυθμίζει τυχόν proxy προς το backend (/api προς http://localhost:PORT). Η δομή στον φάκελο src πρέπει να αντικατοπτρίζει σαφώς τις σελίδες της εφαρμογής (home, courses, course-details, register, about) και να υλοποιεί components (π.χ. course-card, navbar) που επαναχρησιμοποιούν το layout του Μέρος Α. Ο api.service.ts (ή αντίστοιχο JS module) κεντρικοποιεί όλες τις κλήσεις προς το REST API (GET, POST, PUT, DELETE), ώστε η υπόλοιπη εφαρμογή να μη γνωρίζει λεπτομέρειες URL.

Το README-client.md πρέπει να εξηγεί με απλά βήματα πώς ο αξιολογητής εγκαθιστά τα dependencies, πώς εκκινεί την εφαρμογή, σε ποιο URL τρέχει ο client και πώς αυτός συνδέεται με το backend. Το README-part-b.md δίνει συνολική εικόνα: ποια endpoints υπάρχουν, πώς ρέουν τα δεδομένα, ποια είναι η ελάχιστη λειτουργικότητα που υλοποιήθηκε.

4 Ενδεικτικά εργαλεία ανάπτυξης

Για επεξεργασία κώδικα και frontend ανάπτυξη μπορούν να χρησιμοποιήσουν έναν σύγχρονο text tool όπως το Notepad++ ή έναν code editor όπως το Visual Studio Code, διαθέσιμο δωρεάν για όλα τα λειτουργικά συστήματα από την επίσημη σελίδα του. Visual Studio Code

Για backend και MEAN χρειάζονται Node.js και npm, τα οποία εγκαθίστανται από τον επίσημο ιστότοπο του Node.js. Node.js

Για το στήσιμο του περιβάλλοντος frontend στο Μέρος Β μπορούν να αξιοποιήσουν το Vite ως εργαλείο ανάπτυξης και build. Ο επίσημος ιστότοπος και η τεκμηρίωση εξηγούν τα βήματα δημιουργίας νέου έργου και ρύθμισης dev server. vitejs

Για τη βάση δεδομένων απαιτείται εγκατάσταση MongoDB Community Server από την επίσημη σελίδα της MongoDB, ενώ για ευκολότερη επιθεώρηση των δεδομένων είναι χρήσιμο το MongoDB Compass που προσφέρεται επίσης από την MongoDB. MongoDB

Για έλεγχο και δοκιμή των REST APIs συνιστάται η χρήση του Postman, το οποίο παρέχει γραφικό περιβάλλον για κλήσεις HTTP, αποστολή JSON και αποθήκευση συλλογών αιτημάτων. Postman

Τέλος, μπορούν να χρησιμοποιήσουν τα Developer Tools σύγχρονων browsers (Chrome, Firefox, Edge) για επιθεώρηση DOM, CSS και JavaScript, έλεγχο responsive σχεδίασης και παρακολούθηση των HTTP αιτημάτων προς το API στο Μέρος Β.

5 ΟΔΗΓΙΕΣ ΥΠΟΒΟΛΗΣ

Η εξέταση/παρουσίαση των εργασιών είναι υποχρεωτική και πρέπει να περιλαμβάνει συνοπτική περιγραφή της αρχιτεκτονικής από τους φοιτητές, επεξήγηση της ροής δεδομένων και ζωντανή επίδειξη της λειτουργίας ή προβολή βιντεοσκοπημένης επίδειξης του κώδικα. Αν επιλέξετε να δημιουργήσετε βίντεο της εκτέλεσης του project, θα πρέπει στην αρχή να εμφανίζεται ξεκάθαρα ότι η εκτέλεση γίνεται σε δικό σας υπολογιστή (π.χ. προβολή ονόματος υπολογιστή, ημερομηνίας και σχετικών στοιχείων του περιβάλλοντος), πριν τρέξει ο server και ο client. Κατά την παρουσίαση μπορεί να σας ζητηθεί να εντοπίσετε συγκεκριμένα σημεία στον κώδικα, να εξηγήσετε τις επιλογές σχεδίασης και να απαντήσετε σε ερωτήσεις για τον τρόπο με τον οποίο επικοινωνούν client, server και βάση. Μόνο ομάδες που παρουσιάζουν με επαρκή γνώση του κώδικά τους λαμβάνουν βαθμό για το Μέρος Β.

Παραδώστε την εργασία σε έναν ενιαίο συμπιεσμένο φάκελο. Σε αυτόν θα υπάρχει σαφής διάκριση ανάμεσα στο Μέρος Α και στο Μέρος Β, με φακέλους που φέρουν κατανοητές ονομασίες. Για το Μέρος Α, η εκτέλεση θα πρέπει να γίνεται τοπικά ανοίγοντας το index.html. Για το Μέρος Β, θα πρέπει να περιγράφετε με σαφήνεια σε αρχείο οδηγιών τις εντολές εγκατάστασης και εκκίνησης server και client. Η καθαρή οργάνωση των αρχείων, η συνέπεια στις ονομασίες, η τεκμηρίωση και η ποιότητα του κώδικα λαμβάνονται σοβαρά υπόψη στη βαθμολόγηση.

Σε κάθε άσκηση θα **πρέπει** να αντιγράφετε σε νέο κατάλογο και να επεκτείνετε τα αρχεία της προηγούμενης άσκησης.

- Η αρχική σελίδα κάθε άσκησης θα **πρέπει** να ονομάζεται “index.html”.

- Βεβαιωθείτε ότι κατά την παράδοση της εργασίας έχετε προσθέσει στον κατάλογο το παραδοτέου αρχείου “index.html” με κατάλληλους συνδέσμους προς τις αρχικές σελίδες των ασκήσεων του παραδοτέου .

Δομή τελικού παραδοτέου

⇨ p3xxxxxx- p3yyyyyy.zip

- index.html
- /Exercise1/
- /Exercise2/
- /Exercise3/
- /Exercise4/
- /Exercise5/