**Εφαρμογή Υποστήριξης Αρχαιολογικού Μουσείου**

Πρότζεκτ μαθήματος ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΔΙΑΔΙΚΤΥΟΥ , Ομάδα

ΠΑΠΑΔΟΠΟΥΛΟΣ ΣΤΑΜΑΤΙΟΣ – 1046962

Προπτυχιακός Φοιτητής Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών του Πανεπιστημίου Πατρών – Κατεύθυνση Υπολογιστών, [1046962@upnet.gr](mailto:1046962@upnet.gr)

ΠΑΛΗΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ – 1059501

Προπτυχιακός Φοιτητής Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών του Πανεπιστημίου Πατρών – Κατεύθυνση Υπολογιστών, [1059501@upnet.gr](mailto:1059501@upnet.gr)

H κεντρική ιδέα της εργασίας είναι η υλοποίηση μιας εφαρμογής υποστήριξης αρχαιολογικού μουσείου. Χρησιμοποιώντας τις τεχνολογίες των βάσεων δεδομένων και της γλώσσας MongoDB για τo Backend, και τεχνολογίες διαδικτύου (HTML, CSS, Javascript, Bootstrap) για το Frontend, χτίστηκε μια εφαρμογή υψηλών προδιαγραφών με καλλιτεχνικό τρόπο, ο οποία έχει δύο πλευρές: Η πρώτη απευθύνεται στον επισκέπτη (user) και αφορά την ηλεκτρονική έκθεση των συλλογών/εκθεμάτων και εκδηλώσεων του μουσείου, καθώς και τρόπους έκδοσης εισιτηρίων. Η δεύτερη απευθύνεται στον διαχειριστή, η οποία αφορά την διαχείριση των δεδομένων που φιλοξενεί το μουσείο. Η υλοποίηση έγινε με γνώμονα πρώτα απ’όλα την λειτουργικότητα, προσβασιμότητα και το responsiveness (και για μικρές οθόνες λειτουργικότητα), και έπειτα το στυλ και την εμφάνιση.

**Λέξεις κλειδιά •** Βάσεις Δεδομένων •MongoDB • Javascript • Web App • HTML • CSS • Μουσείο • Bootstrap • NodeJs • ExpressJs

**1. Εισαγωγή**

Η εφαρμογή σχεδιάστηκε σε 6 στάδια ανάπτυξης, για την παροχή υπηρεσιών ενός «Εθνικού Αρχαιολογικού Μουσείου», το οποίο φιλοξενεί εκθέματα κυρίως από την Αρχαία Ελλάδα.

* 1. **Στάδια Ανάπτυξης**

Α. Σχεδίαση εννοιολογικού και λογικού-σχεσιακού μοντέλου του μικρόκοσμου του μουσείου.

Β. Κατασκευή της Βάσης Δεδομένων χρησιμοποιώντας τα εργαλεία της MongoDB (ExpressJS, Mongoose κλπ.)

C. Εισαγωγή στοιχείων χρησιμοποιώντας τα εργαλεία της Javascript και της MongoDB.

D. Υλοποίηση των βασικών σεναρίων και συναρτήσεων που αφορούν την αναζήτηση κι επεξεργασία καταγραφών της βάσης.

Ε. Κατασκευή διεπαφής σε μορφή ιστοσελίδας χρησιμοποιώντας τα εργαλεία των HTML,CSS,JS.

F. Υλοποίηση πλαισίου λειτουργίας (framework) σύνδεσης της βάσης με τη διεπαφή χρησιμοποιώντας τα εργαλεία των Εxpress, NodeJS και της Javascript.

G. Aξιολόγηση και διόρθωση λαθών.

**1.2 Καινοτομία**

Η βασική καινοτομία της εφαρμογής είναι η αυτονομία της από τους προγραμματιστές. Ο Διαχειριστής μπορεί να εισάγει, να επεξεργαστεί ή να διαγράψει δεδομένα και η ιστοσελίδα θα ανανεωθεί αυτόματα στην επόμενη συνεδρία. Οι χρήστες μπορούν να αλληλεπιδράσουν με την εφαρμογή και οι ενέργειες τους καταγράφονται στη βάση όπου και είναι προσβάσιμες από το διαχειριστή.

**2. Μεθοδολογία**

Πρωταρχικό και απαραίτητο χαρακτηριστικό μιας υλοποίησης είναι η σχεδίαση ενός δομικού διαγράμματος που απεικονίζει τους στόχους και την αρχή λειτουργίας της εφαρμογής. Αντλήθηκε λοιπόν έμπνευση από υπάρχουσες πλατφόρμες συναφείς με αυτή που ανατέθηκε στην ομάδα, ενώ ταυτόχρονα εξετάστηκαν οι ανάγκες της σημερινής αγοράς υπηρεσιών υποστήριξης μουσείων για την σύλληψη καινοτόμων λύσεων. Ιδιαίτερα χρήσιμες πλατφόρμες σε αυτό ήταν οι ιστοσελίδες του Εθνικού Αρχαιολογικού Μουσείου [1], του Μουσείου Κυκλαδικής Τέχνης [2], του Αρχαιολογικού Μουσείου Θεσσαλονίκης [3] και του Μουσείου Μπενάκη [4].

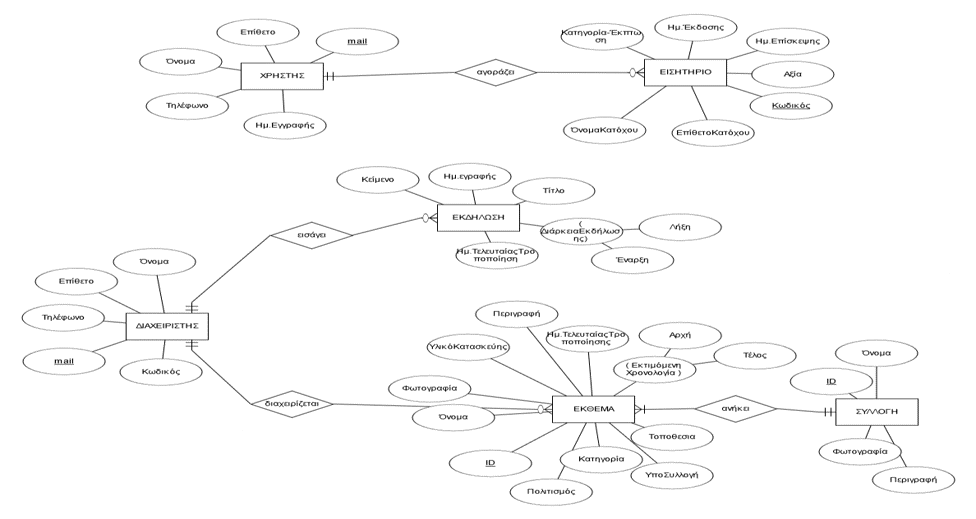
Το διάγραμμα που σχεδιάστηκε απεικονίζεται στην εικόνα 1. Πιο συγκεκριμένα η βάση δεδομένων υλοποιείται με χρήση της MongoDB και επικοινωνεί με έναν εξυπηρετητή (server), υλοποιημένος σε NodeJs, που εκτελεί εντολές τόσο από και προς τη βάση όσο και με τον έξω κόσμο. Στη συνέχεια το σύστημα επικοινωνεί μέσω ξεχωριστών διαύλων προς τους επισκέπτες και προς τον διαχειριστή. Οι επισκέπτες αλληλεπιδρούν με μια ιστοσελίδα υλοποιημένη σε HTML,CSS,JS. Το ίδιο σενάριο ισχύει και για τον διαχειριστή.

Στην πράξη, ο διαχειριστής εποπτεύει, ενημερώνει και επεξεργάζεται τη βάση μέσω της δικής του διεπαφής και η βάση με τη σειρά της ενημερώνει για τις αλλαγές τον server, ο οποίος διαμορφώνει ανάλογα την ιστοσελίδα παρέχοντας τις υπηρεσίες του στους επισκέπτες. Αντίστροφα, οι χρήστες δημιουργώντας λογαριασμούς και κλείνοντας εισιτήρια ηλεκτρονικά, ενημερώνουν την βάση, η οποία στη συνέχεια μέσω του server, ενημερώνει στην επισκόπηση το διαχειριστή του μουσείου.

**2.1**

**Στάδιο Α: Σχεδίαση εννοιολογικού και λογικού-σχεσιακού μοντέλου του μικρόκοσμου του μουσείου**

Η σύλληψη των εννοιολογικών και σχεσιακών μοντέλων βασίστηκε κατά πολύ πάνω σε έρευνα διαφόρων δημοσιεύσεων που μελετούν τα σχετικά διαγράμματα μέσω της πλατφόρμας researchgate.net. Βάσει των στοιχείων που αντλήθηκαν από την εργασίες και βάσει την επισκόπηση που έγινε στις ιστοσελίδες [1],[2],[3],[4] σχεδιάστηκε το ακόλουθο διάγραμμα ERD το οποίο εγκρίθηκε από τον κ. Σιντόρη Χρήστο στην ενδιάμεση παρουσίαση που διεξήχθη.



Εικόνα 1: **Διάγραμμα Οντοτήτων-Συσχετίσεων** (Entity–Relationship Diagram) δημιουργήθηκε με χρήση του εργαλείου ERDplus [5].

Ο μικρόκοσμος του μουσείου αποτελείται από **6 βασικές οντότητες**:

1. Έκθεμα

2. Συλλογή

3. Διαχεριστής

4. Εισιτήριο

5. Χρήστης

6. Εκδήλωση

Το **Έκθεμα** αντιστοιχεί σε οποιοδήποτε αντικείμενο φιλοξενεί το μουσείο και έχει τα εξής γνωρίσματα: ID, Όνομα, Φωτογραφία, Υλικό Κατασκευής, Περιγραφή, Ημερομηνία Τελευταίας Τροποποίησης, (Εκτιμώμενη Χρονολογία), Αρχή, Τέλος, Τοποθεσία, Υποσυλλογή, Κατηγορία, Πολιτισμός.

Η **Συλλογή** εκφράζει το σύνολο των εκθεμάτων που έχουν κοινά χαρακτηριστικά και κρίνεται από το μουσείο ότι ανήκουν στην ίδια ομάδα αντικειμένων. Χαρακτηρίζεται από τα εξής γνωρίσματα: ID, Όνομα, Περιγραφή και Φωτογραφία.

O **Διαχειριστής** έχει τα εξής γνωρίσματα: Mail, Τηλέφωνο, Επίθετο, Όνομα, Ημερομηνία Εγγραφής.

Η **εκδήλωση** έχει τα εξής γνωρίσματα: Κείμενο, Ημερομηνία Εγγραφής, Τίτλος, Διάρκεια Εκδήλωσης, Λήξη, Έναρξη, Ημερομηνία Τελευταίας Τροποποίησης.

Αναφορικά με τη σχέση των τεσσάρων παραπάνω οντοτήτων οφείλουν να αναφερθούν οι εξής παραδοχές:

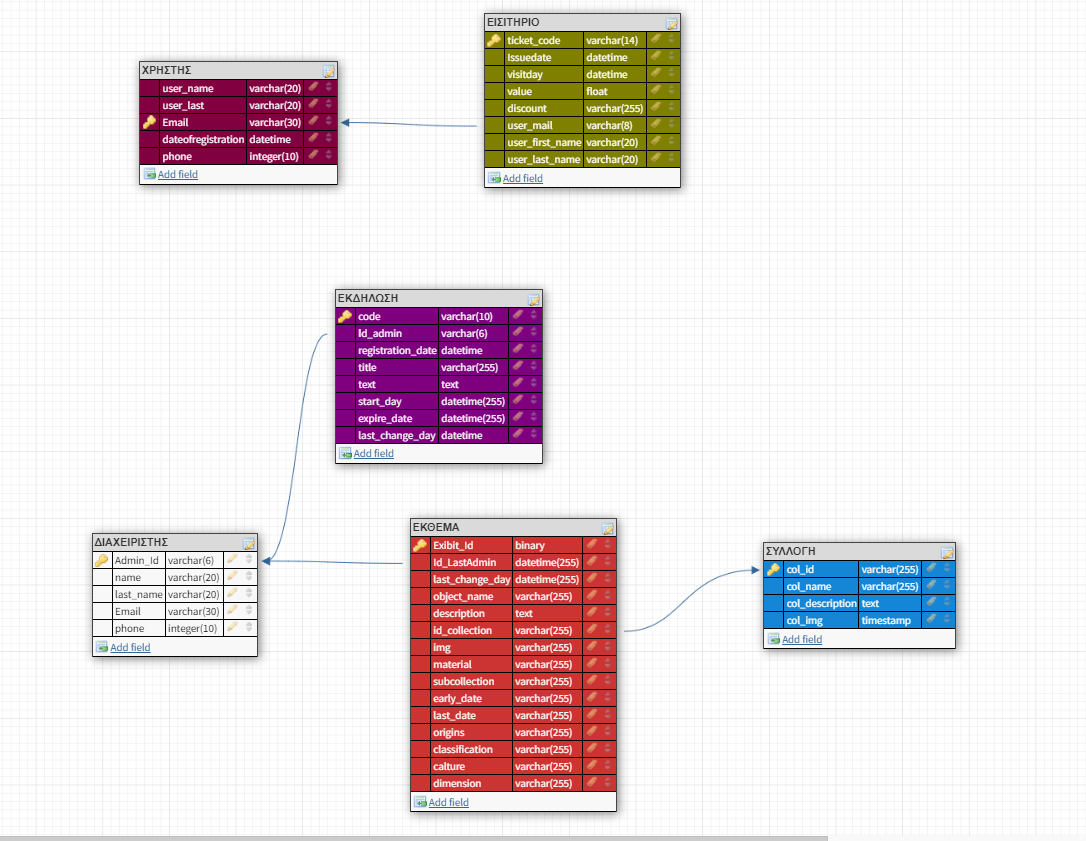
Ένα έκθεμα ανήκει σε μία και μόνο μία υποσυλλογή. Μια υποσυλλογή μπορεί να αποτελείται από κανένα ή πολλά εκθέματα. Ο Διαχειριστής διαχειρίζεται ένα ή περισσότερα εκθέματα. Ο Διαχειριστής εισάγει μία ή περισσότερες εκδηλώσεις.

Ο **Χρήστης** εκφράζει το άτομο που ενεργεί εντός του μικρόκοσμου του μουσείου, ως επισκέπτης. Τα γνωρίσματα του είναι: όνομα, username, email, κωδικός πρόσβασης και ρόλος.

Το **Εισιτήριο** είναι το έγγραφο πρόσβασης ενός επισκέπτη στο χώρο του μουσείου, τα γνωρίσματα του είναι όνομα και επώνυμο κατόχου, barcode, τηλέφωνο κατόχου, email κατόχου, χρηματική αξία, ημερομηνία έκδοσης και επικύρωση.

Να σημειωθεί ότι η έκδοση του εισιτηρίου δεν απαιτεί από τον πελάτη να είναι εγγεγραμμένος στη βάση δεδομένων, απαιτεί όμως τα στοιχεία του. Συνεπώς ένας χρήστης μπορεί να αγοράσει κανένα ή πολλά εισιτήρια και ένα εισιτήριο μπορεί να ανήκει σε έναν ή κανένα χρήστη.

Συμπερασματικά, το λογικό-σχεσιακό μοντέλο που προκύπτει είναι το ακόλουθο:



Εικόνα 3: **Λογικό-σχεσιακό μοντέλο** (δημιουργήθηκε για τις ανάγκες της έκθεσης με χρήση του DBdesigner [6])

Άλλες παρατηρήσεις που μπορούν να γίνουν είναι ότι, για τα εισιτήρια κύριο κλειδί αποτελεί το ticket\_code, ενώ για τον χρήστη αναγνωριστικά αποτελούν το email, το οποίο με τη σειρά του είναι ξένο κλειδί στο εισιτήριο. Ακόμα, για το Διαχειριστή, κύριο κλειδί αποτελεί το Admin\_Id, για τα Εκθέματα το Exhibit\_Id, για τις Συλλογές το col\_id­, ενώ για τις Εκδηλώσεις το code.

Τέλος, θα πρέπει να σημειωθεί πως όλες οι ενέργειες αυτού του σταδίου πραγματοποιήθηκαν από κοινού και από τα δύο μέλη της ομάδας μέσω των εργαλείων που παρέχει η υπηρεσία Discord [7].

**2.2 Στάδιο Β: Κατασκευή της Βάσης Δεδομένων χρησιμοποιώντας τα εργαλεία της MongoDB (ExpressJS, Mongoose κλπ.).**

Λαμβάνοντας υπόψη τις αποφάσεις για τα παραπάνω μοντέλα και τις κατευθυντήριες οδηγίες των διδασκόντων του μαθήματος, υλοποιήθηκε από κοινού ο κώδικας για τη δημιουργία των πινάκων της βάσης. Αξιοποιήθηκε η πλατφόρμα Atlas, ούτως ώστε και τα δύο μέλη της ομάδας να έχουν πρόσβαση παρ’όλο που εργάζονται σε ξεχωριστούς σταθμούς εργασίας (workstations).

**ExhibitSchema:**

const mongoose = require('mongoose');

const ExhibitSchema = new mongoose.Schema({

    Exhibit\_Id:  {

        type:String ,

        required:true,

        unique:true

      },

    Id\_LastAdmin: {

      type:String ,

      required:true

    },

    last\_change\_day: {

      type:Date ,

      required:true,

      default : Date.now

    },

    object\_name: {

      type:String,

      required:true} ,

    coll : {

        type:String,

        required:true

    },

    ex\_description:{

      type:String,

      required:true} ,

    img:{

      type:String,

      required:true},

    period:{

      type:String

    },

    made\_of:{

      type:String,

      required:true},

    sub\_collection:{

      type:String,

      required:true},

    early\_date:{

      type:String,

      required:true},

    late\_date:{

      type:String,

      required:true},

    origins:{

      type:String

    },

    object\_type:{

      type:String,

      required:true},

    culture:{

      type:String

    },

    dimensions:{

      type:String},

    path:{

        type:String

    }

});

  const exhibit=mongoose.model("exhibits",ExhibitSchema);

  module.exports=exhibit;

**EventsSchema**

const mongoose = require("mongoose");

const { Schema } = mongoose;

const eventsSchema = new Schema({

  code:  {

      type:String ,

      required:true,

      unique:true

    }, // String is shorthand for {type: String}

  Id\_admin: {

    type:String ,

    required:true

  },

  registration\_date: {

    type:Date ,

    required:true,

  },

  title: {

    type:String,

    required:true,

    unique:true} ,

  text:{

    type:String,

    required:true} ,

  start\_day:{

    type:String,

    required:true

  },

  expire\_day:{

    type:String,

    required:true

  },

  last\_change\_day:{

    type:Date,

    required:true,

    default: Date.new

  },

  img:{

    type:String,

    required:true

  }

});

const events=mongoose.model("events",eventsSchema);

module.exports=events;

Mε παρόμοιο τρόπο δημιουργούνται και τα υπόλοιπα μοντέλα.

**2.3 Στάδιο C: Εισαγωγή στοιχείων χρησιμοποιώντας τα εργαλεία της Javascript και της MongoDB.**

Τόσο για τη δημιουργία όσο και για την αξιολόγηση του προγράμματος, ήταν αναγκαία η συγκομιδή δεδομένων που θα γέμιζαν τη βάση, ούτως ώστε να είναι εφικτή η προσομοίωση της εφαρμογής σε πραγματικές συνθήκες. Για αυτό το λόγο, συλλέχθηκαν πολυάριθμα αρχεία (datasheets και φωτογραφίες) για διάφορα εκθέματα και εκδηλώσεις και για διάφορες χρονολογικές περιόδους. Συγκεκριμένα, τα εκθέματα που χρησιμοποιήθηκαν ανήκουν στις συλλογές μεταλλοκατασκευών, γλυπτών και αμφορέων του Εθνικού Αρχαιολογικού Μουσείου [8]. Στη συνέχεια, οι πληροφορίες κάθε εκθέματος αντιγράφηκαν σε ξεχωριστά αρχεία της μορφής txt, και με τη βοήθεια της javascript και της βιβλιοθήκης mοngoose [9], εισήχθησαν τα δεδομένα στη βάση.

Ο πλήρης αλγόριθμος μπορεί να βρεθεί στο αρχείο insertX, όπου Χ: Coll, Admin, Events, Exhibits. Ο αλγόριθμος ανοίγει τα εν λόγω αρχεία και τις φωτογραφίες ένα προς μέσω ενός βρόγχου for, στη συνέχεια καταχωρεί τις πληροφορίες μέσω της εντολής insertMany στα κατάλληλα γνωρίσματα. Επίσης, θα πρέπει να επισημανθεί ότι όλες οι ενέργειες τόσο για τον προγραμματισμό σε javascript όσο και για την ανάπτυξη της διεπαφής, που θα αναλυθεί παρακάτω, χρησιμοποιήθηκε τo ολοκληρωμένo περιβάλλον ανάπτυξης (IDE) Visual Studio Code [10]. Τέλος, την εκπόνηση αυτού του σταδίου, ανέλαβε ο κ. Παληός λόγω της καλής εξοικείωσης που διαθέτει με τη γλώσσα Javascript και τις βάσεις δεδομένων

**2.4 Στάδιο D: Υλοποίηση των βασικών σεναρίων και συναρτήσεων που αφορούν την αναζήτηση κι επεξεργασία καταγραφών της βάσης.**

Για να καταστεί η εφαρμογή πλήρης και λειτουργική, οφείλει να παρέχει ένα ευρύ φάσμα υπηρεσιών με αξιόπιστο τρόπο. Επομένως ο σχεδιασμός των σεναρίων και των βασικών συναρτήσεων επικοινωνίας του εξυπηρετητή (server) με τη βάση δεδομένων, αποτέλεσε μεγάλο μέρος της εργασίας. Οι υπηρεσίες που παρέχονται μέσω της ιστοσελίδας είναι: η προβολή των συλλογών και των εκθεμάτων του μουσείου, η φιλτραρισμένη αναζήτηση αυτών, η ηλεκτρονική έκδοση εισιτηρίου, η σύνδεση χρήστη και η επισκόπηση όλων των παραπάνω, η προσθήκη νέων καταγραφών, και επεξεργασία αυτών από τον διαχειριστή.

Στο παρακάτω πλαίσιο επισυνάπτονται μερικά βασικά ερωτήματα (queries) που τίθενται στη βάση για την εκτέλεση των σεναρίων. Να σημειωθεί πως ο πλήρης κώδικας όλων των queries βρίσκεται στο αρχείο …???...

Τέλος, η MongoDB μέσω της βοήθειας της βιβλιοθήκης mongoose.connect επιστρέφει στην javascript όλα τα αποτελέσματα υπό τη μορφή object. Η διαδρομή των δεδομένων από την εν λόγω δομή προς την διεπαφή αναλύεται στο στάδιο E.

**2.5 Στάδιο E Κατασκευή διεπαφής σε μορφή ιστοσελίδας χρησιμοποιώντας τα εργαλεία των HTML,CSS,JS.**

Όπως αναφέρθηκε και στην εισαγωγή, ένας κύριος σκοπός ης εργασίας ήταν να συνδέσει την αρχαία ελληνική ομορφιά με το σύγχρονο τεχνολογικό στυλ. Έπειτα από εκτενή έρευνα στις πλατφόρμες W3schools [11] και MDN Web Docs [12], και Bootstrap [13] αναπτύχθηκε το Frontend. Στη συνέχεια, προστέθηκαν πολλά διαφορετικά στυλ, μέσω αρχείων CSS, κάποια μέσω της βιβλιοθήκης Bootstrap και τα υπόλοιπα συντάχθηκαν από τα μέλη της ομάδας.

Η ιστοσελίδα περιλαμβάνει τις εξής βασικές σελίδες, που συγγράφηκαν με χρήση της HTML:

1. **Αρχική Σελίδα (Home Page).** Καλωσορίζει τον επισκέπτη, μέσω ενός carousel από αφίσες.

2. **Εκθέματα (Exhibitions).** Προβάλλει στον χρήστη όλα τα εκθέματα του μουσείου. Σε αυτή τη σελίδα παρουσιάζονται αναλυτικά όλες οι πληροφορίες που αφορούν τις εκθέσεις (φωτογραφία προφίλ, περιγραφή, αίθουσα, υποσυλλογές κλπ.).

3. **Εισιτήρια (Tickets).** Εμφανίζει στο χρήστη την φόρμα που πρέπει να συμπληρώσει για να αγοράσει ηλεκτρονικά το εισιτήριο του, το οποίο του εμφανίζεται μόλις εισάγει τα στοιχεία του, με τα δικά του στοιχεία που έχει εισάγει.

4. **Σύνδεση - Aποσύνδεση (Sign In/Out).** Εμφανίζει στον χρήστη τις φόρμες για σύνδεση και αποσύνδεση από το λογαριασμό του.

5. **Δρώμενα - Περιοδικές εκθέσεις (Events**). Προβάλλει στο χρήστη, το σύνολο των περιοδικών εκθέσεων που, είτε τρέχουν αυτή τη στιγμή, είτε πρόκειται να έρθουν στο μέλλον. Στις καρτέλες αυτές φαίνονται όλα τα στοιχεία των εκθέσεων που αφορούν το χρονικό πλαίσιο φιλοξενίας τους στους χώρους του μουσείου.

6. **Αναζήτηση εκθέματος** **(Search Bar).** Περιλαμβάνει επίσης ένα κουμπί φίλτρων (filters) το οποίο ενεργοποιεί ένα παράθυρο διαλόγου (modal) με τον χρήστη. Μέσω αυτού, ο χρήστης μπορεί να επιλέξει ποια εκθέματα επιθυμεί να δει, είτε επιλέγοντας γνωρίσματα από τις ανάλογες λίστες, είτε εισάγοντας χρονολογικό εύρος στα αντίστοιχα κελιά, είτε ακόμα εισάγοντας λέξεις κλειδιά.

Επιπρόσθετα η πλατφόρμα, στις υπηρεσίες που παρέχει στο διαχειριστή, διαθέτει τις εξής σελίδες:

**Σελίδα επισκόπησης**. Προβάλλονται υπό μορφή πινάκων όλες οι πληροφορίες που επιθυμεί ο διαχειριστής, αναφορικά με οποιαδήποτε καταγραφή της βάσης.

**Σελίδα προσθήκης**. Αποτελεί φόρμα με την οποία ο διαχειριστής δύναται να προσθέσει νέα εκθέματα, συλλογές και εκθέσεις.

**Σελίδα Επεξεργασίας.** Αποτελεί φόρμα με την οποία ο διαχειριστής δύναται να επεξεργαστεί υπάρχοντας εκθέματα και συλλογές

**Σελίδα Επιλογής Διαγραφής**. Αποτελεί φόρμα με την οποία ο διαχειριστής επιλέγει ποια καταγραφή θέλει να σβήσει.

Ο διαχειριστής μπορεί να φιλτράρει την Πληροφορία που επιθυμεί να επισκοπήσει είτε μέσω φίλτρων (όπως επισκέπτης στη γκαλερί), είτε μέσω της λειτουργίας για προχωρημένους όπου μπορεί να εκτελέσει απευθείας query σε mongodb. (???????)

Η αδυναμία της HTML είναι ότι αποτελεί μια στατική γλώσσα προγραμματισμού, όμως απαίτηση της εφαρμογής είναι η δυναμικότητα. Έτσι λοιπόν, αναζητήθηκε λύση στη Ηandlebars, η οποία είναι μια δυναμική βιβλιοθήκη.

Όλοι οι κώδικες σε HTML είναι διαθέσιμες στον αναγνώστη και βρίσκονται στο φάκελο static/html της εφαρμογής.

Τέλος, αναφορικά με την εκτέλεση αυτού του σταδίου ανάπτυξης, ο κ. Παληός ασχολήθηκε με την επιμέλεια των σελίδων που αφορούν τις εκδηλώσεις, την αρχική σελίδα και την επισκόπηση του διαχειριστή. Ο κ. Παπαδόπουλος υλοποίησε τις φόρμες σύνδεσης, έκδοσης εισιτηρίου καθώς και το σύνολο των φορμών επεξεργασίας και προσθήκης στην ενότητα του διαχειριστή.

**2.6 Στάδιο F:** **Υλοποίηση πλαισίου λειτουργίας (framework) σύνδεσης της βάσης με τη διεπαφή χρησιμοποιώντας τα εργαλεία των Express, NodeJS και της Javascript.**

Οι δυνατότητες που προσφέρει το Εxpress framework [14] και το NodeJS [15] είναι πολυάριθμες και πολύ χρήσιμες. Με το Express ήταν δυνατό να στηθεί ένα framework με δυναμική δρομολόγηση σελίδων (dynamic routing). Αυτό σημαίνει, ότι στο προηγούμενο στάδιο, ήταν απαραίτητη η σύνταξη μόνο κάποιων συγκεκριμένων σελίδων-προτύπων (static) και όχι ξεχωριστές για κάθε προφίλ εκθέματος, συλλογής ή έκθεσης. Αυτή η δυνατότητα άλλωστε αποτελεί τον πυρήνα της καινοτομίας της εφαρμογής: ότι η πλατφόρμα ανανεώνεται αυτόματα, όταν ο διαχειριστής κάνει τροποποιήσεις στη βάση δεδομένων.

Πριν συνεχιστεί η ανάλυση θα πρέπει να επισημανθεί το εξής. Όπως αναφέρθηκε στην ενότητα 2.4, η mongoose.connect επιστρέφει στην Javascript τις καταγραφές σε δομή **object**. Όμως η εν λόγω δομή είναι αμετάβλητη (immutable), συνεπώς η επεξεργασία των καταγραφών και η μεταφορά τους προς την ιστοσελίδα θα ήταν μια επίπονη διαδικασία. Για αυτό λοιπόν, όλες οι δομές object, μετατράπηκαν σε **λίστες** που είναι μεταβλητές (mutable). ……????.........

Για να υλοποιηθεί λοιπόν το framework, ορίστηκαν όλες οι διαδρομές (routes) που αντιστοιχούν στις σελίδες της ενότητας 2.5. Η διαδρομή συντελείται από μια συνάρτηση στην οποία εμπεριέχεται ένα μπλοκ εντολών και στο τέλος πάντα επιστρέφεται ένα πρότυπο το οποίο αντιστοιχεί σε μια από τις ιστοσελίδες σε html και δέχεται όσα ορίσματα απαιτείται από τον σχεδιαστή.

Kαλείται η τρέχουσα ημερομηνία όπως και το σύνολο των καταγραφών των εκθέσεων-τρέχοντων,καινούριων- υπό τη μορφή λίστας όπως έχουν διαμορφωθεί κατάλληλα από τη συνάρτηση που επικοινωνεί με τη βάση χρησιμοποιώντας την mongoose.connect. Οι λίστες αυτές επιστρέφονται στο πρότυπο ‘events.html’. Τη σκυτάλη στην απεικόνιση των δεδομένων παίρνει τώρα η HMTL, σε συνεργασία με το Express. Δημιουργούνται, αυτόματα μέσω ενός βρόγχου, όλα τα στοιχεία (elements) που πρέπει να απεικονιστούν σε κώδικα html.

Γενικά, όλες οι λειτουργίες της διεπαφής δομούνται με την ίδια λογική.

**Λεπτομέρειες Framework**

Το **appstart.js (server)** ακολουθεί τη λογική MVC. Το πρόγραμμα μοιράζεται σε διάφορα τμήματα:

- **/models** παρέχει την πρόσβαση στη βάση δεδομένων

- **/controllers** περιέχει συναρτήσεις για να έχουμε πρόσβαση στα δεδομένα που μας δίνουν τα models.

- **/static** περιέχει αρχεία HTML, CSS, Javascript στους φακέλους /html, /css, /js, αντίστοιχα.

- **/insertX** περιέχει τα αρχεία αρχικοποίησης της βάσης, όπου Χ: Coll, Events, Exhibits, Admin, Ticketzero, userzero.

- Οι φάκελοι **views** και **routers**:

- Ο **/views** περιέχει template γραμμένα σε handlebars.

- Ο **/routers** περιέχει τις διαδρομές που αναγνωρίζει η εφαρμογή.

**Τα μοντέλα - The models**

Ο φάκελος `/models` περιέχει **έξι (6) μοντέλα:**

- col\_model: Μοντέλο που περιέχει δεδομένα για τις συλλογές (collections),

- events\_model: Μοντέλο που περιέχει δεδομένα για τις εκδηλώσεις (events),

- model\_admin: Μοντέλο που περιέχει δεδομένα για τoυς administrators (admins),

- model\_exhibitions: Μοντέλο που περιέχει δεδομένα για τα εκθέματα (exhibits),

- ticket\_model: Μοντέλο που περιέχει δεδομένα για τα εισιτήρια (tickets),

- user\_model: Μοντέλο που περιέχει δεδομένα για τον χρήστη (user).

**Οι controllers**

Στο αρχείο /controllers/eventsControllers περιέχονται οι συναρτήσεις χειρισμού του μοντέλου των εκδηλώσεων.

Στο αρχείο /controllers/exhibitsControllers περιέχονται οι συναρτήσεις χειρισμού του μοντέλου των εκθεμάτων.

**2.7 Στάδιο G: Αξιολόγηση και διόρθωση λαθών.**

Κατά τη διάρκεια της υλοποίησης εκτελέστηκαν πολυάριθμες δοκιμές, στην αρχή στο κάθε τμήμα ξεχωριστά και στη συνέχεια στο ολοκληρωμένο σύστημα, για να εξασφαλιστεί η ομαλή λειτουργία της εφαρμογής για όλα τα πιθανά σενάρια χρήσης της από τον χρήστη. Σε αυτό το στάδιο προστέθηκαν και τα κατάλληλα μηνύματα προς τον χρήστη κάθε φορά που αυτός εκτελεί μια ενέργεια. Οι δοκιμές αυτές πραγματοποιήθηκαν σε πρώτη φάση από τα μέλη της ομάδας και σε δεύτερη από τρίτους.

Η αξιολόγηση για τα ποιοτικά χαρακτηριστικά έγινε βάσει συζήτησης με τους ανθρώπους που εθελοντικά ανέλαβαν να δοκιμάσουν την εφαρμογή. Η συντριπτική πλειοψηφία τους εξέφρασε πως της φάνηκε ιδιαίτερα πρωτότυπη η καινοτομία της απεξάρτησης από τους προγραμματιστές, σχολίασε θετικά τη λειτουργικότητα και ενέδειξε σημάδια ενθουσιασμού αναφορικά με την αισθητική της ιστοσελίδας.

**2.8 Κατάθεση επισημάνσεων, σχετικά με κάποια τεχνικά χαρακτηριστικά**

Η σύνταξη του τμήματος/προγράμματος υπεύθυνο για την έκδοση ηλεκτρονικού εισιτηρίου ήταν μια ιδιαίτερα ενδιαφέρουσα διαδικασία και μια σημαντική πρόκληση.

Η λογική του προγράμματος αυτού είναι να λαμβάνει τα στοιχεία που δηλώνει ο χρήστης στη σχετική φόρμα και ύστερα δημιουργεί έναν μοναδικό κωδικό, στον οποίο αναγράφονται τα αρχικά του ονόματος του επισκέπτη, η ημερομηνία και η ακριβής χρονική στιγμή έκδοσης. Το πρόγραμμα χρησιμοποιεί ένα πρότυπο σε HTML που απεικονίζει το εισιτήριο, το οποίο συμπληρώνεται κατάλληλα με τα στοιχεία του επισκέπτη με χρήση απλών τεχνικών χειρισμού αρχείων (file handling).

Τέλος, όλοι οι κώδικες αναφορικά με την έκδοση του εισιτηρίου βρίσκονται στο φάκελο static, στον υποφάκελο html.

**3. Χρονοδιάγραμμα**

|  |  |
| --- | --- |
| **Ημερομηνία** | **Ενέργεια** |
| 8-12/3/21 | Ανάθεση θέματος πρότζεκτ |
| 2-3/4/21 | Τελικός Σχεδιασμός ERD και Schema  Σύσκεψη μέσω Discord για επιλογή προτύπων ιστοσελίδας |
| 9/4/21 | Έναρξη εξόρυξης δεδομένων |
| 10/4/21 | Λήξη εξόρυξης δεδομένων |

|  |  |
| --- | --- |
| 16/4/21 | Υλοποίηση Αρχικής Σελίδας (homepage) |
| 17/4/21 | Υλοποίηση Σελίδας Εκθεμάτων/Συλλογών |
| 23/4/21 | Yλοποίηση Eισιτηρίου |
| 24/4/21 | Υλοποίηση Αναζήτησης |
| 30/4/21 | Έναρξη υλοποίησης εντολών για τα διάφορα σενάρια |
| 1/5/21 | Υλοποίηση Events |
| 7/5/21 | Διακοπές Πάσχα – Περίοδος Ξεκούρασης |
| 8/5/21 | Τελική Κατασκευή Πινάκων της Βάσης Δεδομένων |
| 14/5/21 | Υλοποίηση κώδικα μεταφοράς δεδομένων στη βάση |
| 15/5/21 | Υλοποίηση βάσης και μεταφορά δεδομένων |
| 22/5/21 | Υλοποίηση σελίδας Διαχειριστή |
| 23/5/21 | Yλοποίηση Sign In/Out |
| 28/5/21 | Έναρξη δοκιμών και επιδιόρθωση προβλημάτων |
| 29/5/21 | Συγγραφή οδηγιών χρήσης (readme), έκθεσης |
| 30/5/21 | Τελική έκδοση και δημιουργία εκτελέσιμου |
| 6/6/21 | Ανέβασμα αρχείων σε cloud |
| 6/6/21 | Tελική παράδοση έκθεσης |

**4. Αναφορές**

1: Εθνικό Αρχαιολογικό Μουσείο. (n.d). Αρχική. Ανάκτηση Απρίλιος, 2021, από https://www.namuseum.gr/

2: Μουσείο Κυκλαδικής Τέχνης. (n.d). Αρχική. Ανάκτηση Απρίλιος, 2021, από https://cycladic.gr/

3: Αρχαιολογικό Μουσείο Θεσσαλονίκης (n.d). Αρχική. Ανάκτηση Απρίλιος, 2021, από https://www.amth.gr/

4: Μουσείο Μπενάκη (n,d). Αρχική. Ανάκτηση Απρίλιος, 2021, από https://www.benaki.org

5: ERDplus. (n.d). Home. Ανάκτηση Απρίλιος, 2021, από https://erdplus.com/

6: DBdesigner. (n.d). Home. Ανάκτηση Απρίλιος, 2021, από <https://www.dbdesigner.net/>

7: Discord. (n.d). Home. Ανάκτηση Απρίλιος, 2021, από <https://discord.com/>

8: Εθνικό Αρχαιολογικό Μουσείο. Συλλογές. Ανάκτηση Απρίλιος, 2021, <https://www.namuseum.gr/collections/>

9: Βιβλιοθήκη Mongoose. Ανάκτηση Μάιος, 2021, <https://mongoosejs.com/>

10: Visual Studio Code. Ανάκτηση Απρίλιος 2021, <https://code.visualstudio.com/>

11:W3Schools, Aνάκτηση Μάρτιος 2021, https://www.w3schools.com/.

12: MDN, Ανάκτηση Μάρτιος 2021, https://developer.mozilla.org/en-US/.

13: Bootstrap, Ανάκτηση Μάρτιος 2021, https://getbootstrap.com/.

14: Express, Ανάκτηση Μάιος 2021,. https://expressjs.com/.

15: NodeJS, Ανάκτηση Μάιος 2021, <https://nodejs.org/en/>.

16: Handlebars, Ανάκτηση Μάιος 2021, https://handlebarsjs.com/.