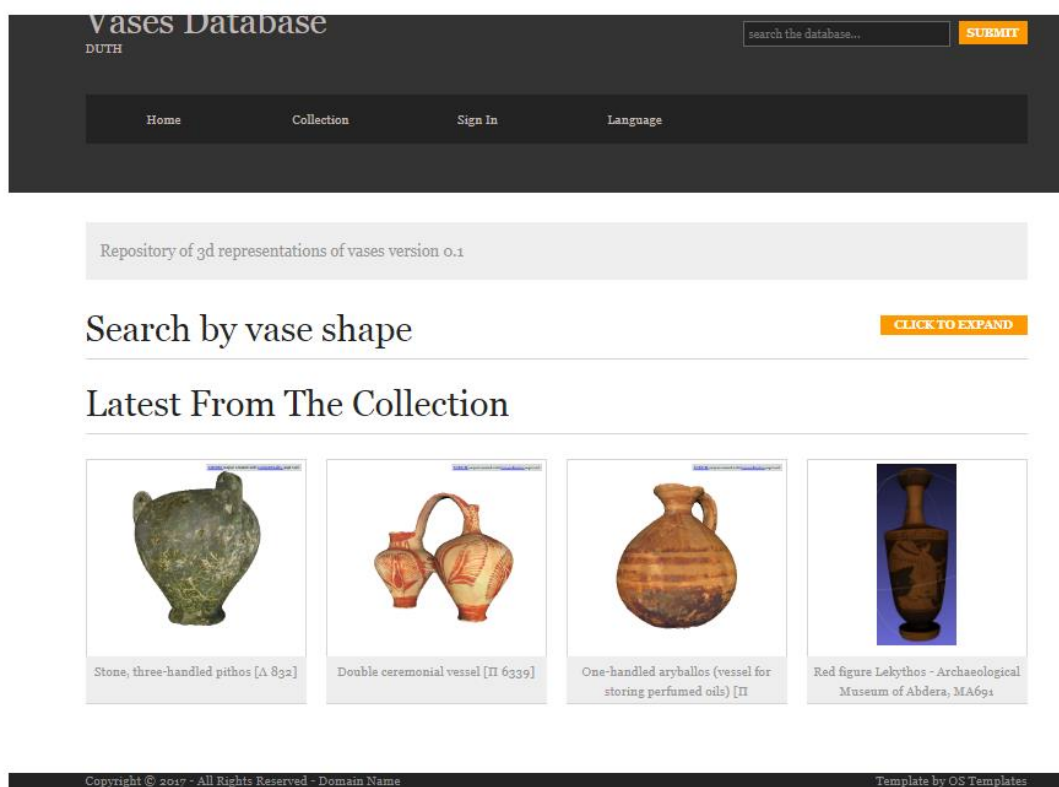




Δημοκρίτειο Πανεπιστήμιο Θράκης
Πολυτεχνική Σχολή Ξάνθης
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Ανάπτυξη και υλοποίηση βάσης δεδομένων 3D αντικειμένων

Διπλωματική εργασία
Στεργιούλας Ανδρέας
Α.Ε.Μ.:56017



Επιβλέπων: Καθηγητής Χριστόδουλος Χαμζάς

Ξάνθη, 2018

Περίληψη.....	iii
Abstract	iv
Ευχαριστίες.....	v
Κεφάλαιο 1ο: Διαχείριση και αποθήκευση δεδομένων	1
1.1 Εισαγωγή	1
1.2 Δεδομένα και πληροφορία	1
1.3 Βάσεις Δεδομένων	2
1.3.1 Ιστορική εξέλιξη βάσεων δεδομένων	2
1.3.2 Σύστημα βάσης δεδομένων	5
1.3.3 Αρχιτεκτονική των βάσεων δεδομένων	6
1.4 Μοντέλα βάσεων δεδομένων	7
1.5 Σύνοψη	7
Κεφάλαιο 2ο: Native Βάσεις δεδομένων και η 'eXistdb'	8
2.1 Η XML γλώσσα	8
2.2 Βάσεις δεδομένων XML και μοντέλο δεδομένων XML	9
2.2.1 Δομή δέντρου.....	10
2.2.2 Elements (οντότητες)	11
2.2.3 Attributes (γνωρίσματα).....	12
2.2.4 Namespace (χώρος ονομάτων)	12
2.2.5 DTD και XML Schema	13
2.2.6 Η XQuery.....	14
2.3 Η 'eXistdb'	15
2.4 Σύνοψη	17
Κεφάλαιο 3ο: Εργαλεία κατασκευής διεπαφής χρήστη(frontend)	18
3.1 Εισαγωγή	18
3.2 Η γλώσσα HTML.....	18
3.2.1 Ετικέτες (Tags)	19
3.2.2 Μία απλή σελίδα σε HTML.....	20
3.2.3 Γνωρίσματα (Attributes).....	20
3.2.4 Επικεφαλίδες (Headings).....	21
3.2.5 Παράγραφοι HTML	21
3.2.6 Μορφοποίηση με την χρήση CSS αρχείων	22
3.2.7 Έντυπα (forms) και λίστες (lists).....	23
3.3 Η γλώσσα PHP	26
3.3.1 Σύνταξη της γλώσσας.....	27
3.3.2 Συναρτήσεις και Κλάσεις.....	28
Κεφάλαιο 4ο: Οργάνωση και ταξινόμηση των καταχωρίσεων.....	31
4.1 Εισαγωγή	31

4.2 Το εργαλείο <i>Retrieval</i> και η συμβολή του στην κατηγοριοποίηση των αποθηκευμένων αντικειμένων.....	31
4.3 Απόδοση σε αρχεία XML των κατηγοριοποιήσεων.....	36
4.4 Αποθήκευση μεταδεδομένων των αγγείων.....	38
4.4.1 Σχήμα μεταδεδομένων CARARE.....	38
4.4.2 Αποθήκευση των μεταδεδομένων	43
Κεφάλαιο 5ο: Υλοποίηση.....	46
5.1 Στήσιμο της βάσης δεδομένων.....	46
5.2 Κατασκευή της ιστοσελίδας.....	46
5.2.1 Αρχική σελίδα.....	47
5.2.2 Περιήγηση στη συλλογή	48
5.2.3 Σύνδεση χρήστη/δημιουργία νέου λογαριασμού	51
Κεφάλαιο 6ο: Συμπεράσματα.....	55
Παραρτήματα	56
Παράρτημα Α' κώδικες.....	56
Παράρτημα Β' ταξινόμηση αγγείων βάση σχήματος	57
Ανοιχτά αγγεία.....	57
Κλειστά αγγεία.....	61
Μικροαντικείμενα	70
Βιβλιογραφία	71

Περίληψη

Στην παρούσα εργασία παρουσιάζεται η κατασκευή ενός αποθετηρίου φιλοξενίας τρισδιάστατων αρχαίων ελληνικών αγγείων, με σκοπό την αξιοποίησή τους από αλγόριθμους ανάκτησης πληροφοριών.

Στο πρώτο κεφάλαιο γίνεται μία εισαγωγή στις έννοιες των δεδομένων και πληροφοριών και των βάσεων δεδομένων.

Στο δεύτερο κεφάλαιο παρουσιάζεται η γλώσσα σήμανσης XML που χρησιμοποιείται για την περιγραφή των μεταπληροφοριών των αντικειμένων καθώς και των εργαλείων που αναπτύχθηκαν γύρω από αυτήν.

Κατόπιν, στο τρίτο κεφάλαιο παρουσιάζονται οι γλώσσες που χρησιμοποιήθηκαν επί το πλείστον για την ανάπτυξη της ιστοσελίδας που παρέχει πρόσβαση στα δεδομένα της βάσης.

Στη συνέχεια, το τέταρτο κεφάλαιο περιέχει την ταξινόμηση στην οποία υποβλήθηκαν τα τρισδιάστατα αντικείμενα ώστε να μπορέσουν να χρησιμοποιηθούν από αλγόριθμους ανάπτυξης καθώς και τον τρόπο απόδοσης της ταξινόμησης στη βάση. Επίσης καταγράφεται η οργάνωση των πληροφοριών των αντικειμένων σε έγγραφα XML της βάσης δεδομένων.

Τέλος, στο πέμπτο κεφάλαιο αναπτύσσεται η κωδικοποίηση της ιστοσελίδας και των δυνατοτήτων που προσφέρεται στους χρήστες της.

Abstract

This thesis paper presents the construction of a three-dimensional ancient Greek vases hosting repository for use in information retrieval algorithms.

The first chapter describes an introduction to the concepts of data, information and databases.

The second chapter introduces the XML markup language used to describe the metadata of the objects and the tools developed around it.

Then, in the third chapter, the languages most used to develop the web page that provides access to database data are presented.

Subsequently, the fourth chapter contains the classification of the three-dimensional objects to be used by retrieval algorithms as well as the way in which the classification is rendered in the database. It also records the organization of the information that is used to describe each entry of the database in XML documents.

Finally, chapter five develops the coding of the website and the possibilities offered to its users.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένειά μου για όση υποστήριξη μου έχει προσφέρει μέχρι στιγμής καθώς και τον καθηγητή Χριστόδουλο Χαμζά για την επίβλεψη της διπλωματικής εργασίας. Τέλος ευχαριστώ τον υποψήφιο διδακτορικό και ερευνητή Γεώργιο-Αλέξη Ιωαννάκη για τη βοήθειά του κατά την εκπόνηση της εργασίας.

Κεφάλαιο 1ο: Διαχείριση και αποθήκευση δεδομένων

1.1 Εισαγωγή

Η αλματώδης ανάπτυξη της επιστήμης της πληροφορικής και των επικοινωνιών τα τελευταία χρόνια έχει καταστήσει την πληροφορία ως ένα από τα βασικότερα και πολυτιμότερα αγαθά. Είναι πλέον κοινώς αποδεκτό πως το αγαθό της πληροφορίας είναι επιθυμητό απ' όλους τους εργαζόμενους αλλά και τους εκπαιδευόμενους, ώστε να είναι πιο αποδοτικοί, ανταγωνιστικοί αλλά και παραγωγικοί στην εργασία τους. Το κοινό χαρακτηριστικό των σύγχρονων εφαρμογών είναι πως απαιτούν αποτελεσματική και αποδοτική διαχείριση της αποθηκευμένης πληροφορίας[1].

Οι βάσεις δεδομένων και η τεχνολογία βάσεων δεδομένων εξασκούν σημαντική επίδραση στην αυξανόμενη χρήση των υπολογιστών. Είναι εύλογο να ειπωθεί ότι οι βάσεις δεδομένων διαδραματίζουν ένα σημαντικό ρόλο σε όλες τις περιοχές όπου χρησιμοποιούνται υπολογιστές, συμπεριλαμβανομένων των επιχειρήσεων, του ηλεκτρονικού εμπορίου, της μηχανικής, της ιατρικής, της νομικής, της εκπαίδευσης και της βιβλιοθηκονομίας, για να αναφέρουμε μερικές μόνο από αυτές. Επομένως είναι απαραίτητη η μελέτη των βάσεων δεδομένων, όταν πρόκειται να αναφερθούμε στην διαχείριση δεδομένων.

1.2 Δεδομένα και πληροφορία

Με τον όρο πληροφορία αναφερόμαστε συνήθως σε ειδήσεις, γεγονότα και έννοιες που αποκτάμε από την καθημερινή μας επικοινωνία και τα θεωρούμε ως αποκτηθείσα γνώση, ενώ τα δεδομένα μπορούν να είναι μη κατάλληλα επεξεργασμένα και μη ταξινομημένα σύνολα πληροφοριών. Ένας αυστηρός ορισμός για το τι είναι δεδομένα και τι είναι πληροφορία, σύμφωνα με την επιτροπή ANSI των ΗΠΑ, είναι ο εξής : Δεδομένα (*data*) είναι μια παράσταση, όπως γράμματα, αριθμοί, σύμβολα κ.ά. στα οποία μπορούμε να δώσουμε κάποια σημασία (έννοια). Πληροφορία (*information*) είναι η σημασία που δίνουμε σ' ένα σύνολο από δεδομένα, τα οποία μπορούμε να επεξεργαστούμε βάσει προκαθορισμένων κανόνων και να βγάλουμε έτσι κάποια χρήσιμα συμπεράσματα. Με τις πληροφορίες περιορίζεται η αβεβαιότητα που έχουμε για διάφορα πράγματα και βοηθιόμαστε, έτσι, στο να λάβουμε σωστές αποφάσεις[1].

Τα δεδομένα μπορούν να θεωρηθούν ως τρόποι αναπαράστασης εννοιών και γεγονότων που μπορούν να υποστούν διαχείριση και επεξεργασία. Η συλλογή και αποθήκευση ενός τεράστιου όγκου δεδομένων όπως απαιτούν οι κοινωνικές συνθήκες σήμερα, δεν λύνει τελείως το πρόβλημα της σωστής οργάνωσης και ταξινόμησης των δεδομένων. Τα δεδομένα θα πρέπει να οργανωθούν με τέτοιο τρόπο έτσι ώστε να μπορούμε να τα εντοπίζουμε και να τα αξιοποιούμε εύκολα και γρήγορα και τη στιγμή που τα χρειαζόμαστε[1].

Ένα παράδειγμα κακής οργάνωσης δεδομένων θα ήταν ένα έγγραφο με τα ονοματεπώνυμα όλων των φοιτητών του τμήματος των Ηλεκτρολόγων Μηχανικών και Μηχανικών Ξάνθης στο οποίο δεν θα υπήρχε καμία οργάνωση αλλά θα ήταν εντελώς τυχαία. Μία τέτοια λίστα, παρόλο που θα περιείχε έναν μεγάλο όγκο πληροφορίας, θα ήταν πρακτικά άχρηστος, μιας και η οποιαδήποτε αναζήτηση σε αυτήν θα ήταν πολύ χρονοβόρα.

Χαρακτηριστικά παραδείγματα δεδομένων που απαιτούν σωστή και αποδοτική οργάνωση είναι τα εξής :

- Τα στοιχεία υπαλλήλων, πελατών, προμηθευτών και παραγγελιών
- μιας εμπορικής επιχείρησης.
- Τα στοιχεία υλικών μιας αποθήκης.
- Τα στοιχεία ταινιών, πελατών και δανεισμών μιας βιντεολέσχης.

- Τα στοιχεία υπαλλήλων, γιατρών, ασθενών αλλά και υλικών ενός νοσοκομείου.
- Τα στοιχεία βιβλίων, χρηστών (δανειστών) και δανεισμών μιας βιβλιοθήκης.

1.3 Βάσεις Δεδομένων

Όπως αναφέρθηκε και στην εισαγωγή, εξαιτίας του μεγάλου όγκου πληροφοριών που απαιτείται για την λειτουργία των σύγχρονων εφαρμογών, η αρχική προσέγγιση διαχείρισης της πληροφορίας δεν είναι αποδοτική. Επομένως είναι απαραίτητη η χρήση μίας βάσης δεδομένων. Τα συστήματα βάσεων δεδομένων τα χρησιμοποιούμε για να μπορούμε να αποθηκεύσουμε, να επεξεργαστούμε αλλά και να εκμεταλλευτούμε αποδοτικά αυτόν τον τεράστιο όγκο των πληροφοριών που αυξάνονται με αλματώδεις ρυθμούς καθημερινά. Μια Βάση Δεδομένων (DB) είναι ένα σύνολο αρχείων με υψηλό βαθμό οργάνωσης τα οποία είναι συνδεδεμένα μεταξύ τους με λογικές σχέσεις, έτσι ώστε να μπορούν να χρησιμοποιούνται από πολλές εφαρμογές και από πολλούς χρήστες ταυτόχρονα. Με τον όρο σύστημα διαχείρισης βάσης δεδομένων (DBMS) καλούμε κάποιο ειδικό λογισμικό το οποίο μεσολαβεί μεταξύ των αρχείων δεδομένων και των διάφορων εφαρμογών και καταγράφει, κατατάσσει, εντοπίζει, αποθηκεύει και ανακτά δεδομένα, διατηρώντας την ακεραιότητά τους και τα προβάλλει στην επιθυμητή μορφή από τον χρήστη[1]. Στην ουσία ένα DBMS είναι ένα σύνολο από προγράμματα και υπορουτίνες που έχουν να κάνουν με τον χειρισμό της βάσης δεδομένων, όσον αφορά τη δημιουργία, τροποποίηση, διαγραφή στοιχείων, με ελέγχους ασφαλείας κ.ά.

Σε μία βάση δεδομένων, τα διάφορα δεδομένα που υπάρχουν αποθηκευμένα θα πρέπει να πληρούν τις παρακάτω ιδιότητες[2]:

- Να είναι ολοκληρωμένα (*Integrated*), δηλ. τα δεδομένα πρέπει να είναι αποθηκευμένα σε ομοιόμορφα οργανωμένα σύνολα αρχείων όπου δεν πρέπει να υπάρχει επανάληψη ή πλεονασμός (*redundancy*) των ίδιων στοιχείων, και
- Να είναι καταμεριζόμενα (*Shared*), δηλ. να μπορούν περισσότεροι του ενός χρήστες να βλέπουν και να μοιράζονται τα ίδια δεδομένα την ίδια χρονική στιγμή.

Οι στόχοι μιας βάσης δεδομένων είναι οι εξής :

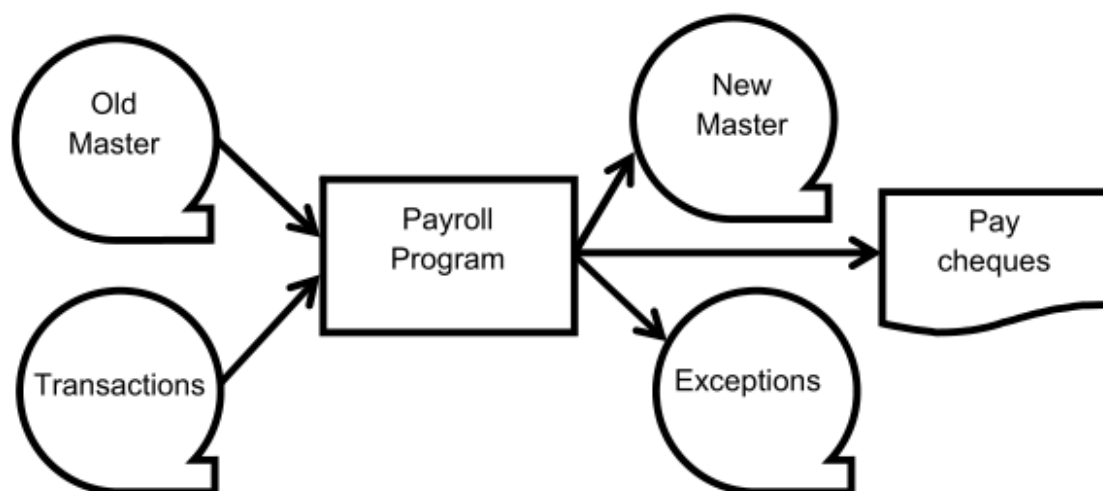
- Ο περιορισμός της πολλαπλής αποθήκευσης των ίδιων στοιχείων (*redundancy*).
- Ο καταμερισμός (*sharing*) των ίδιων στοιχείων σ' όλους τους χρήστες.
- Η ομοιομορφία (*uniformity*) στον χειρισμό και την αναπαράσταση των δεδομένων.
- Η επιβολή κανόνων ασφαλείας (*security*).
- Η διατήρηση της ακεραιότητας (*integrity*) και της αξιοπιστίας (*reliability*) των δεδομένων.
- Η ανεξαρτησία των δεδομένων (*data independence*) και των προγραμμάτων από τον φυσικό τρόπο αποθήκευσης των δεδομένων.

1.3.1 Ιστορική εξέλιξη βάσεων δεδομένων

Τα αρχικά δεδομένα επεξεργασίας συστημάτων υπολογιστών βασίστηκαν σε ήδη υπάρχοντα συστήματα. Αυτά ήταν διαδοχικά συστήματα, από τα οποία αποτελούνταν από μεμονωμένα αρχεία, τα οποία οργανώνονται με κάποια προκαθορισμένη σειρά. Αν και αυτό δεν αποτελεί μια βάση δεδομένων επειδή δεν υπήρχε μία ολοκληρωμένη πηγή δεδομένων, η ηλεκτρονική επεξεργασία αρχείων ήταν το πρώτο βήμα προς ένα ηλεκτρονικό σύστημα πληροφοριών, βασισμένο σε δεδομένα. Η επεξεργασία απαιτούσε να γίνετε η προσπέλαση από την πρώτη εγγραφή και να συνεχίζετε μέχρι τέλους. Αυτό ήταν πολύ καλό για την παραγωγή μισθοδοσίας και μηνιαίων λογαριασμών, αλλά λιγότερο χρήσιμο στην κλήση μεμονωμένων αρχείων. Ως αποτέλεσμα, δεν ήταν ασυνήθιστο να υπάρχει εκτυπωμένη η

τελευταία έκδοση ενός αρχείου το οποίο θα μπορούσε να αναζητηθεί με το χέρι, αν χρειαζόταν, το οποίο διατηρούταν έως όταν να εκτυπωνόταν μία καινούρια έκδοση του αρχείου. Αυτή η διαδοχική επεξεργασία αρχείων επίσης σήμαινε ότι έπρεπε να περικοπούν οι χρόνοι όταν δεν γίνονταν δεκτές νέες ενημερώσεις πριν από την εκτέλεση των προγραμμάτων[3].

Παρόλο που υπήρχαν οι δίσκοι αποθήκευσης, ήταν αρκετά κοστοβόροι και τα πρώτα συστήματα έτειναν να βασίζονται σε διάτρητες κάρτες 80 στηλών, διάτρητες χαρτοταινίες ή μαγνητικές ταινίες αποθήκευσης, πράγμα που σήμαινε ότι η εισαγωγή, η ενημέρωση και η παραγωγή έπρεπε να γίνουν με πολύ συγκεκριμένους τρόπους.



Εικόνα 1.1: ένα σύστημα βασισμένο σε ταινίες

Η IBM για παράδειγμα, χρησιμοποιούσε μία κάρτα εισόδου η οποία αντιγραφόταν σε μία μαγνητική ταινία. Αυτά τα δεδομένα, αφού θα γινόταν μία ταξινόμηση τους, θα συγχωνεύονταν και θα χρησιμοποιούνταν για την ενημέρωση ήδη υπαρχόντων αρχείων. Αυτά τα ήταν συνήθως αποθηκευμένα στην κύρια ταινία, η οποία διατηρούσε τα επεξεργασμένα κύρια αρχεία από την προηγούμενη εκτέλεση του συστήματος. Μόλις γινόταν η ενημέρωση, παραγόταν μια νέα κύρια ταινία, που θα γινόταν η παλιά κύρια στην επόμενη εκτέλεση. Η οποιαδήποτε επεξεργασία έπρεπε να είναι προγραμματισμένη το πότε θα γινόταν. Δεδομένου ότι μόνο ένα πρόγραμμα θα μπορούσε να φορτωθεί και να τρέξει σε κάθε χρονική στιγμή, η απαιτήσεις για τον υπολογιστικό χρόνο αυξήθηκε γρήγορα. Στις περισσότερες περιπτώσεις, πολλά συστήματα ηλεκτρονικών υπολογιστών είχαν προγραμματιστεί να τρέχουν για 24 ώρες την ημέρα, 7 ημέρες εβδομάδα.

Υπήρχαν επίσης πολύ αυστηροί κανόνες για τις ταινίες αποθήκευσης, ώστε στην περίπτωση όπου μία κύρια ταινία καταστρεφόταν, θα μπορούσε να αναδημιουργηθεί χρησιμοποιώντας αρχειοθετημένες κύριες ταινίες. Αν υπήρχε οποιοδήποτε πρόβλημα κατά την διαδικασία, αυτό δεν θα επεξεργαζόταν αλλά θα να αντιγραφόταν σε ένα αρχείο εξαιρέσεων σε κάποια άλλη ταινία για να εξεταστεί αργότερα προκειμένου να βρεθεί τι είχε πάει στραβά. Κανονικά το πρόβλημα θα διορθωνόταν προσθέτοντας μία νέα ενέργεια προς εκτέλεση, που θα εκτελούνταν κατά την επόμενη εκτέλεση του προγράμματος[3].

Ήταν αναπόφευκτο να οργανωθούν τα δεδομένα με κάποιο τρόπο ώστε να γίνει η αποθήκευση και η επαναχρησιμοποίηση δεδομένων πιο αποτελεσματική. Ωστόσο, ενώ αναπτυσσόταν αυτά τα συστήματα, υπήρχε επίσης μια κίνηση από τους κατασκευαστές να κλειδώσουν τους πελάτες στα ιδιόκτητα προϊόντα τους.

Η UNIVAC και η IBM ανταγωνίζονταν για την ανάπτυξη της πρώτης βάσης δεδομένων όπου τα αρχεία θα συνδέονταν με έναν τρόπο που δεν ήταν διαδοχικός. Η UNIVAC αποφάσισε να υιοθετήσει την γλώσσα προγραμματισμού *COBOL (Common Business Oriented Language)* και συνεπώς επίσης υιοθέτησε τους κανόνες *CODASYL (Conference on Data Systems Languages)* για την ανάπτυξη της βάσης δεδομένων τους (μοντέλο δικτύου). Το CODASYL ήταν μια κοινοπραξία που δημιουργήθηκε το 1959 για να αναπτυχθεί μια κοινή γλώσσα προγραμματισμού που έγινε η βάση για τη COBOL. Μολονότι η Honeywell ήταν μέλος της ομάδας CODASYL, προσπάθησαν προτείνουν τη γλώσσα προγραμματισμού *FACT* ως δοκιμασμένη και λειτουργική εναλλακτική λύση σε σχέση με τη δοκιμασμένη COBOL. Όπως αποδείχθηκε, αυτή η στρατηγική δεν ήταν επιτυχής. Το 1967 η CODASYL μετονομάστηκε σε *Database Task Group (DBTG)* και ανέπτυξε μια σειρά επεκτάσεων σε COBOL για τη διαχείριση βάσεων δεδομένων. Η Honeywell, η Univac και η εταιρεία ψηφιακών εξοπλισμών (*DEC*) ήταν μεταξύ εκείνων που υιοθέτησαν αυτό το πρότυπο για τις εφαρμογές βάσεων δεδομένων τους[3].

Η IBM, από την άλλη πλευρά, είχε αναπτύξει τη δική της γλώσσα προγραμματισμού *PL/1 (Programming Language 1)* και μια εφαρμογή βάσης δεδομένων γνωστή ως *IMS (Information Management System)* για την εξαιρετικά δημοφιλής οικογένεια υπολογιστών IBM Series 360. Το σύστημα UNIVAC ήταν μια βάση δεδομένων δικτύου και η IBM ακολουθούσε μία αυστηρά ιεραρχική δομή. Το μοντέλο πλοήγησης ήταν ως εξής: ιεραρχικό, όπου το σύστημα προσπελάσεται σε ένα σημείο και μετά γινόταν ιεραρχική πλοήγηση προς τα κάτω, πριν γίνει η οποιαδήποτε σειριακή αναζήτηση για την εύρεση του επιθυμητού αρχείου. Και τα δύο συστήματα βασίζονταν σε pointers.

Το Σχεσιακό Μοντέλο, προτάθηκε για πρώτη φορά το 1970 από τον *Edgar F. Codd*, ο οποίος επιμένει ότι οι αιτήσεις ενός χρήστη θα πρέπει να αναζητούν δεδομένα με βάση το περιεχόμενο και όχι ακολουθώντας συνδέσμους. Αυτό θεωρήθηκε αναγκαίο, για να επιτραπεί το περιεχόμενο μιας βάσης δεδομένων να εξελιχθεί χωρίς την συνεχή αναμόρφωση των εφαρμογών. Τα σχεσιακά συστήματα έθεσαν σοβαρές απαιτήσεις όσον αφορά τους πόρους επεξεργασίας και δεν ήταν μέχρι τα μέσα της δεκαετίας του 1980 όπου το υλικό των υπολογιστών έγινε αρκετά έτσι ώστε να μπορούν να χρησιμοποιηθούν ευρέως. Από τις αρχές του 1990, ωστόσο, τα σχεσιακά συστήματα ήταν κυρίαρχα σε όλες τις εφαρμογές μεγάλης κλίμακας επεξεργασίας στοιχείων και παραμένουν κυρίαρχα μέχρι και σήμερα, εκτός από εξειδικευμένες περιπτώσεις. Η κυρίαρχη η γλώσσα των βάσεων δεδομένων είναι το πρότυπο *SQL (Structured Query Language)* ή Δομημένη Γλώσσα Ερωτημάτων για το σχεσιακό μοντέλο, το που έχει επηρεάσει τις γλώσσες των βάσεων δεδομένων και άλλων μοντέλων δεδομένων.

Επειδή το Σχέδιο μοντέλο δίνει έμφαση στην αναζήτηση παρά στη πλοήγηση, δεν παρουσιάζει τις σχέσεις μεταξύ διαφόρων φορέων αποκλειστικά με την μορφή των δείκτη, αλλά τις αντιπροσωπεύει με το να χρησιμοποιεί *πρωτεύοντα* και *ξένα κλειδιά*. Ενώ αυτό αποτελεί μία καλή βάση για μια *query*(ερωταπαντήσεις) γλώσσα, είναι λιγότερο κατάλληλο ως γλώσσα μοντελοποίησης. Για το λόγο αυτό, ένα διαφορετικό μοντέλο, το μοντέλο Οντοτήτων-Συσχετίσεων (*Entity-Relationship model*) που αναπτύχθηκε λίγο αργότερα (1976), κέρδισε την δημοτικότητα στον σχεδιασμό βάσεων δεδομένων[3].

Από το 1970 και μετά, η τεχνολογία βάσεων δεδομένων έχει συμβαδίζει με τους αυξανόμενους πόρους που θα καταστούν διαθέσιμοι μέσω της τεχνολογίας των υπολογιστών και συγκεκριμένα μέσω της ταχείας αύξησης της χωρητικότητας και της ταχύτητας (με ταυτόχρονη μείωση των τιμών) των την αποθήκευση και την αύξηση της χωρητικότητας της κύριας μνήμης. Η δυσκολία του σχεσιακού μοντέλου, στην οποία όλα τα δεδομένα αποθηκεύονται σε πίνακες με συγκεκριμένη δομή γραμμών και στηλών, όλο και περισσότερο θεωρείται ως περιορισμός κατά τον χειρισμό πληροφοριών, η οποία είναι πιο πλούσια ή ποικίλει στη δομή από την παραδοσιακή μορφή των δεδομένων των εταιρικών πληροφοριακών συστημάτων, για παράδειγμα βάσεις δεδομένων εγγράφου, βάσεις

δεδομένων μηχανικής, Βάσεις δεδομένων πολυμέσων ή βάσεις δεδομένων που χρησιμοποιούνται στις μοριακές επιστήμες. Διαφορετικές προσπάθειες έχουν γίνει για να αντιμετωπίζονται αυτό το πρόβλημα, πολλά από αυτά οδηγούν σε λύσεις όπως οι μετασχεσιακές ή NoSQL. Δύο εξελίξεις που αξίζει να σημειώθηκαν είναι η Βάση Δεδομένων Αντικειμένων (*Object Database*) και η Βάση Δεδομένων XML(*eXtensible Markup Language*), οι οποία είναι αυτή που χρησιμοποιείται στην παρούσα διπλωματική εργασία. Οι προμηθευτές των σχεσιακών βάσεων δεδομένων έχουν καταπολεμήσει τον ανταγωνισμό με τα νεότερα μοντέλα, επεκτείνοντας τις δυνατότητες των προϊόντων τους ώστε να υποστηρίξουν μια μεγαλύτερη ποικιλία δεδομένων.

1.3.2 Σύστημα βάσης δεδομένων

Η σχεδίαση, η ανάπτυξη καθώς και η συντήρηση μίας βάσης δεδομένων απαιτεί την ύπαρξη εξειδικευμένου προσωπικού, το οποίο θα εκτελεί τις παραπάνω ενέργειες. Το προσωπικό αυτό, εκτελεί τις παραπάνω ενέργειες με την χρήση εξειδικευμένων εργαλείων τα οποία παρέχονται είτε ως μέρος του συστήματος διαχείρισης βάσης δεδομένων(DBMS) ή είναι αυτόνομα εργαλεία. Στα εργαλεία αυτά περιλαμβάνονται εξειδικευμένες γλώσσες βάσεων δεδομένων που περιλαμβάνουν γλώσσες ορισμού δεδομένων (*data definition languages- DDL*), γλώσσες χειρισμού δεδομένων (*data manipulation languages – DML*), και γλώσσες ερωταπαντήσεων (*query languages*). Αυτές είναι στην ουσία γλώσσες προγραμματισμού ειδικού σκοπού, κατασκευασμένες για να διαχειρίζονται τις βάσεις δεδομένων. Τις περισσότερες φορές δίνονται ως επεκτάσεις των ήδη υπαρχόντων γλωσσών προγραμματισμού. Οι γλώσσες των βάσεων δεδομένων είναι συγκεκριμένες για ένα μοντέλο δεδομένων, και σε πολλές περιπτώσεις για έναν συγκεκριμένο τύπο DBMS.

Οι χρήστες των εφαρμογών, ανακτούν πληροφορίες από μία βάση δεδομένων χωρίς να χρειάζεται να γνωρίζουν την δομή της. Το DBMS παίζει τον ρόλο του μεσάζοντα μεταξύ του χρήστη και της βάσης. Μόνο μέσω του DBMS μπορεί ο χρήστης να αντλήσει πληροφορίες από τη βάση δεδομένων. Ένα DBMS μπορεί να είναι εγκατεστημένο σ' έναν μόνο υπολογιστή ή σε ένα δίκτυο(cluster) υπολογιστών και μπορεί να χρησιμοποιείται από έναν χρήστη ή και από πολλούς χρήστες.

Ένα Σύστημα Βάσης Δεδομένων (ΣΒΔ) ή DBS (*Data Base System*) αποτελείται από το υλικό, το λογισμικό, τη βάση δεδομένων και τους χρήστες[4]. Είναι στην ουσία ένα σύστημα που δίνει την δυνατότητα αποθήκευσης και να αξιοποίησης δεδομένων με τη βοήθεια ηλεκτρονικού υπολογιστή. Παρακάτω αναπτύσσονται πιο αναλυτικά.

Το υλικό (*hardware*) αποτελείται από τους ηλεκτρονικούς υπολογιστές, τα περιφερειακά, τους σκληρούς δίσκους, τις μαγνητικές ταινίες κ.ά., όπου είναι αποθηκευμένα τα αρχεία της βάσης δεδομένων αλλά και τα προγράμματα που χρησιμοποιούνται για την επεξεργασία τους. Το λογισμικό (*software*) είναι το σύνολο των προγραμμάτων που χρησιμοποιούνται για την επεξεργασία των δεδομένων της βάσης δεδομένων. Η βάση δεδομένων (*database*) αποτελείται από το σύνολο των αρχείων όπου είναι αποθηκευμένα τα δεδομένα του συστήματος. Τα στοιχεία αυτά μπορεί να βρίσκονται αποθηκευμένα σ' έναν φυσικό υπολογιστή αλλά και σε περισσότερους. Όμως, στον χρήστη δίνεται η εντύπωση ότι βρίσκονται συγκεντρωμένα στον ίδιο υπολογιστή. Τα δεδομένα των αρχείων αυτών είναι ενοποιημένα (*data integration*), δηλαδή, δεν υπάρχει πλεονασμός δεδομένων και κατακερματισμός τους (*data sharing*), δίνοντας την δυνατότητα ύπαρξης ταυτόχρονης προσπέλασης των δεδομένων από πολλούς χρήστες. Ο κάθε χρήστης έχει διαφορετικά δικαιώματα και βλέπει διαφορετικό κομμάτι της βάσης δεδομένων, ανάλογα με τον σκοπό για τον οποίο συνδέεται και τα δικαιώματα που του έχουν δοθεί από τον διαχειριστή του συστήματος.

Σε αυτό το σημείο θα γίνει μία ανάλυση των κατηγοριών χρηστών μίας βάσης δεδομένων. Οι κατηγορίες αυτές είναι α)οι προγραμματιστές του DBMS (*developers*). Αυτοί είναι οι

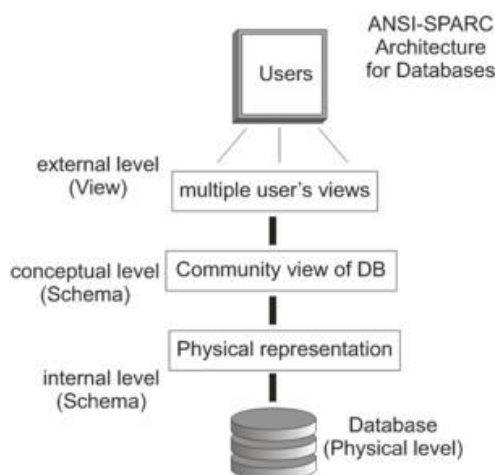
άνθρωποι, οι οποίοι σχεδιάζουν και αναπτύσσουν το DBMS και οι μόνοι που θα ασχοληθούν με τον κώδικα που υπάρχει από πίσω. Είναι συνήθως οι εργαζόμενοι ενός προμηθευτή DBMS π.χ. η Oracle, η IBM, η Microsoft, η Sybase, ή στην περίπτωση των open-source π.χ. η MySQL, εθελοντές ή άτομα που υποστηρίζονται από τις ενδιαφερόμενες επιχειρήσεις και οργανισμούς. Τα άτομα αυτά έχουν τυπικά προσόντα προγραμματιστών συστημάτων. Η ανάπτυξη ενός DBMS είναι πολύπλοκη και δύσκολη εργασία και μερικά από τα πιο δημοφιλή DBMS είναι υπό συνεχή ανάπτυξη και βελτίωση για δεκαετίες. Β) οι διαχειριστές Δεδομένων (*Data Administrator*). Έχουν τη διοικητική αρμοδιότητα και ευθύνη για την οργάνωση της βάσης δεδομένων και την απόδοση δικαιωμάτων πρόσβασης στους χρήστες. γ) οι προγραμματιστές εφαρμογών (*Applications developers*) και δ) οι διαχειριστές Βάσεων Δεδομένων (*DataBase Administrators*). Οι πρώτοι αναπτύσσουν τις εφαρμογές του ΣΒΔ σε κάποια από τις γνωστές γλώσσες προγραμματισμού ενώ οι δεύτεροι σχεδιάζουν την απαιτούμενη βάση δεδομένων, λαμβάνουν οδηγίες από τον διαχειριστή δεδομένων και είναι αυτοί που διαθέτουν τις τεχνικές γνώσεις και αρμοδιότητες για τη σωστή και αποδοτική λειτουργία του DBMS. Και οι δύο ωστόσο είναι εξοικειωμένοι με το προϊόν. Η τελική κατηγορία είναι ε) οι τελικοί χρήστες (*end users*). Χρησιμοποιούν κάποια εφαρμογή για να παίρνουν στοιχεία από μια βάση δεδομένων, έχουν τις λιγότερες δυνατότητες επέμβασης στα στοιχεία της βάσης δεδομένων, χρησιμοποιούν ειδικούς κωδικούς πρόσβασης και το σύστημα τους επιτρέπει ανάλογη πρόσβαση σε συγκεκριμένο κομμάτι της βάσης δεδομένων.

1.3.3 Αρχιτεκτονική των βάσεων δεδομένων

Τα δεδομένα αποθηκεύονται ουσιαστικά ως δυαδικά ψηφία ή ως αριθμούς και σειρές, αλλά είναι δύσκολο να γίνει η οποιαδήποτε αλληλεπίδραση με αυτά σε τόσο χαμηλό επίπεδο. Είναι απαραίτητο να η οργάνωσή τους σε διαφορετικά επίπεδα.

Μία αρχική πρόταση για την ύπαρξη πρότυπης ορολογίας καθώς και γενικής αρχιτεκτονικής στα συστήματα βάσεων δεδομένων δημιουργήθηκε το 1971 από τη *DBTG (Data Base Task Group)*, που διορίστηκε από την CODACIL. Η DBTG αναγνώρισε αμέσως πως υπήρχε η ανάγκη για μία προσέγγιση δύο επιπέδων, το ένα από τη σκοπιά του συστήματος η οποία καλείτε σχήμα και το δεύτερο επίπεδο από τη σκοπιά του χρήστη, το υπο-σχήμα[5].

Παρακάτω είναι μία εικόνα που δείχνει την *ANSI_SPARC* αρχιτεκτονική του συστήματος βάσεων δεδομένων.



Εικόνα 1.2: η *ANSI_SPARC* αρχιτεκτονική

Τα επίπεδα αποτελούν μια αρχιτεκτονική τριών επιπέδων που περιλαμβάνει το εξωτερικό, το εννοιολογικό και το εσωτερικό επίπεδο. Ο τρόπος με τον οποίο οι χρήστες αναγνωρίζουν τα δεδομένα ονομάζεται εξωτερικό επίπεδο. Ο τρόπος με τον οποίο το DBMS και το λειτουργικό σύστημα διακρίνουν τα δεδομένα είναι το εσωτερικό επίπεδο, όπου γίνεται η πραγματική αποθήκευση των δεδομένων, χρησιμοποιώντας τις δομές δεδομένων. Το εννοιολογικό επίπεδο προσφέρει τόσο τη χαρτογράφηση όσο και την επιθυμητή ανεξαρτησία μεταξύ των εξωτερικών και των εσωτερικών επιπέδων. Αναλυτικότερα, το εσωτερικό επίπεδο έχει να κάνει με την αποθήκευση των αρχείων στον σκληρό δίσκο, δηλαδή την πραγματική ή φυσική κατάστασή τους. Ασχολείται με το κόστος, την απόδοση, την επεκτασιμότητα, με τη διάταξη αποθήκευσης του εννοιολογικού επιπέδου και παρέχει υποστήριξη αποθήκευσης. Το εξωτερικό επίπεδο έχει να κάνει με τους χρήστες είτε αυτοί είναι απλοί χειριστές, είτε προγραμματιστές ή και διαχειριστές της βάσης δεδομένων και καθορίζει το πως κάθε τελικός χρήστης κατανοεί την οργάνωση των αντίστοιχων σχετικών στοιχείων του στη βάση δεδομένων και τέλος το εννοιολογικό επίπεδο είναι ένα ενδιάμεσο επίπεδο που διασυνδέει τα δύο άλλα επίπεδα και έχει να κάνει με τη λογική σχεδίαση των αρχείων της βάσης δεδομένων. Παρέχεται με τον απλούστερο δυνατό τρόπο και αποτελεί τη σπονδυλική στήλη της βάσης δεδομένων. Είναι έξω από το πεδίο εφαρμογής των διάφορων τελικών χρηστών, εξυπηρετεί τους προγραμματιστές εφαρμογών και ορίζεται από τους διαχειριστές της βάσης δεδομένων.

1.4 Μοντέλα βάσεων δεδομένων

Ένα μοντέλο βάσης δεδομένων είναι ένας τύπος μοντέλου δεδομένων που καθορίζει τη λογική δομή μιας βάσης δεδομένων και καθορίζει κατά βάση τον τρόπο με τον οποίο τα δεδομένα μπορούν να αποθηκευτούν, να οργανωθούν και να υποβληθούν σε επεξεργασία. Για να επιτύχουμε τη μέγιστη αποτελεσματικότητα σε ένα μοντέλο, αυτό θα πρέπει να είναι σχετικά απλό. Το πιο δημοφιλές παράδειγμα ενός μοντέλου βάσης δεδομένων είναι το σχεσιακό μοντέλο, στο οποίο τα δεδομένα είναι οργανωμένα σε πίνακες. Γενικά, τα επικρατέστερα μοντέλα είναι τρία, το ιεραρχικό, το δικτυωτό και το σχεσιακό. Επειδή όμως στην παρούσα διπλωματική δεν έχει χρησιμοποιηθεί κάποιο από τα παραπάνω αλλά το μοντέλο βάσης δεδομένων XML, δεν θα γίνει περαιτέρω ανάλυση αυτών των μοντέλων, ωστόσο θα γίνει μία παρουσίαση του XML μοντέλου παρακάτω.

1.5 Σύνοψη

Σε αυτό το κεφάλαιο έγινε μια ανάλυση στο τι καλούμε δεδομένα, και πληροφορία, στη σημαντικότητα αυτών των όρων και στην ανάγκη ύπαρξης βάσεων δεδομένων για την διαχείριση του μεγάλου όγκου πληροφοριών με τον βέλτιστο δυνατό τρόπο. Παράλληλα, παρουσιάστηκε η ιστορική εξέλιξη των βάσεων δεδομένων, εισάχθηκε η έννοια των συστημάτων διαχείρισης βάσης δεδομένων (DBMS) και των συστημάτων βάσης δεδομένων (ΣΒΔ) καθώς και της γενικής αρχιτεκτονικής που διέπει μία βάση δεδομένων.

Κεφάλαιο 2ο: Native Βάσεις δεδομένων και η 'eXistdb'

Στόχος της ενότητας αυτής είναι η παρουσίαση της XML βάσεως δεδομένων 'eXistdb' καθώς και την μορφή των ερωταπαντήσεων (queries) που διαβάζουν ή γράφουν πληροφορίες στη βάση αυτή. Είναι επομένως αναπόφευκτο να γίνει και μία επεξήγηση του τρόπου λειτουργίας του XML μοντέλου βάσεων δεδομένων.

2.1 Η XML γλώσσα

Η XML ή αλλιώς επεκτάσιμη γλώσσα σήμανσης(*Extensible Markup Language*), είναι μία γλώσσα σήμανσης που καθορίζει ένα σύνολο κανόνων για την κωδικοποίηση εγγράφων σε μορφή που είναι αναγνώσιμη τόσο από τον άνθρωπο όσο και αναγνώσιμη από υπολογιστές. Περιγράφει μια κατηγορία αντικειμένων δεδομένων που καλούνται έγγραφα XML και περιγράφει εν μέρει τη συμπεριφορά των προγραμμάτων υπολογιστών που τα επεξεργάζονται. Η XML είναι ένα προφίλ εφαρμογής (application profile), δηλαδή ένα σύνολο μεταδεδομένων (metadata), πολιτικών και κατευθυντήριων γραμμών που ορίζονται για μια συγκεκριμένη εφαρμογή ή μια περιορισμένη μορφή του *SGML (Standard Generalized Markup Language)* προτύπου [ISO 8879]. Κατά την κατασκευή, τα έγγραφα XML συμμορφώνονται με τα έγγραφα SGML[6].

Τα έγγραφα XML αποτελούνται από μονάδες αποθήκευσης που ονομάζονται οντότητες (entities), οι οποίες περιέχουν αναλυμένα (parsed) ή μη αναλυμένα(unparsed) δεδομένα. Τα ταξινομημένα δεδομένα αποτελούνται από χαρακτήρες, μερικοί από τους οποίους σχηματίζουν δεδομένα. Ορισμένοι όμως από τους χαρακτήρες χρησιμοποιούνται για σήμανση, η οποία στην ουσία κωδικοποιεί μια περιγραφή της διάταξης αποθήκευσης του εγγράφου και της λογικής δομής[6]. Η XML παρέχει ένα μηχανισμό για την επιβολή περιορισμών στη διάταξη αποθήκευσης και στη λογική δομή. Για την ανάγνωση των XML εγγράφων και την παροχή πρόσβασης στο περιεχόμενο καθώς και στη δομή τους απαιτείται η χρήση μίας μονάδας λογισμικού που καλείται επεξεργαστής XML. Υποτίθεται ότι ένας επεξεργαστής XML κάνει το έργο του για λογαριασμό ενός άλλου στοιχείου, το στοιχείο εφαρμογής. Αυτή η προδιαγραφή περιγράφει την απαιτούμενη συμπεριφορά ενός επεξεργαστή XML ως προς τον τρόπο με τον οποίο πρέπει να διαβάζει δεδομένα XML και τις πληροφορίες που πρέπει να παρέχει στην εφαρμογή.

Η XML αναπτύχθηκε αρχικά από μια ομάδα εργασίας πάνω στην XML, αρχικά γνωστή ως *SGML Editorial Review Board*, που δημιουργήθηκε υπό την αιγίδα της *World Wide Web Consortium (W3C)* το 1996. Οι σχεδιαστικοί στόχοι της XML είναι οι εξής:

- Να μπορεί να χρησιμοποιηθεί μέσω του διαδικτύου.
- Να υποστηρίζει μια μεγάλη ποικιλία εφαρμογών.
- Η XML πρέπει να είναι συμβατή με το SGML.
- Η σύνταξη προγραμμάτων που επεξεργάζονται XML έγγραφα πρέπει να είναι εύκολη.
- Ο αριθμός των προαιρετικών χαρακτηριστικών σε XML πρέπει να είναι όσο το δυνατόν μικρότερος, ιδανικά μηδέν.
- Τα έγγραφα XML θα πρέπει να είναι ευανάγνωστα από τον άνθρωπο και λογικά σαφή.
- Ο σχεδιασμός XML θα πρέπει να προετοιμάζεται γρήγορα.

- Ο σχεδιασμός της XML να είναι τυπικός και συνοπτικός.
- Τα έγγραφα XML να είναι εύκολο να δημιουργηθούν.
- Η ακεραιότητα στη σήμανση XML να είναι ελάχιστης σημασίας.

Οι παραπάνω προσδιορισμοί, μαζί με τα συναφή πρότυπα, το *Unicode [Unicode]* και *ISO/IEC 10646 [ISO/IEC 10646]* για χαρακτήρες, το *Internet BCP 47 [IETF BCP 47]* και το μητρώο γλώσσας *[IANA-LANGCODES]* για ετικέτες αναγνώρισης γλώσσας, παρέχουν όλες τις απαραίτητες πληροφορίες για την κατανόηση της τρέχουσας εκδόσεως της XML (έκδοση 1.0) και την κατασκευή προγραμμάτων υπολογιστή για την επεξεργασία της[6].

2.2 Βάσεις δεδομένων XML και μοντέλο δεδομένων XML

Μια βάση δεδομένων XML είναι ένα *data persistence* σύστημα λογισμικού επιτήρησης δεδομένων που επιτρέπει τον καθορισμό και μερικές φορές και την αποθήκευση των δεδομένων σε μορφή XML. Σε αυτά τα δεδομένα μπορούν να γίνουν διεργασίες ανάκλησης, μετατροπής, εξαγωγής και επιστροφής από ένα σύστημα κλήσης. Οι βάσεις δεδομένων XML είναι μια κατηγορία της βάσης δεδομένων *NoSQL*. Υπάρχουν διάφοροι λόγοι για τους οποίους ο προσδιορισμός δεδομένων σε μορφή XML ή σε άλλες μορφές εγγράφων, όπως το *JSON* είναι καταλληλότερος και επομένως οι XML βάσεις είναι αποδοτικότερες[7]. Ειδικότερα για την XML, αυτοί οι λόγοι μπορεί να είναι:

- Μια επιχείρηση μπορεί να έχει πολλές εγγραφές XML σε μια υπάρχουσα τυποποιημένη μορφή.
- Τα δεδομένα μπορεί να χρειαστεί να ληφθούν ως XML, οπότε χρησιμοποιώντας μια άλλη μορφή, παραδείγματος χάριν σχεσιακή, να απαιτεί διπλή μοντελοποίηση των δεδομένων.
- Η XML είναι καταλληλότερη για σποραδικά δεδομένα, βαθιά εφωλιασμένα δεδομένα και μικτό περιεχόμενο (όπως κείμενο με ενσωματωμένες ετικέτες σήμανσης).
- Η XML είναι αναγνώσιμη από τον άνθρωπο, ενώ οι σχεσιακοί πίνακες απαιτούν εμπειρία για πρόσβαση σε αυτούς.
- Τα μεταδεδομένα είναι συχνά διαθέσιμα σε XML.
- Τα δεδομένα του Σημασιολογικού Ιστού είναι διαθέσιμα ως RDF/XML.

Ένας άλλος λόγος για τη χρήση της XML σε βάσεις δεδομένων, είναι η όλο και συχνότερη χρήση της XML για τη μεταφορά δεδομένων (*data migration*)[8], δηλαδή τα διάφορα δεδομένα που βρίσκονται αποθηκευμένα σε κάποια βάση δεδομένων εξάγονται από αυτήν σε αρχεία XML ή πληροφορίες σε XML αρχεία μεταφέρονται από αυτά και αποθηκεύονται σε βάσεις δεδομένων. Συνεπώς η XML μπορεί να αποδειχθεί αποδοτικότερη, από άποψη κόστους μετατροπής. Στις εφαρμογές που βασίζονται σε περιεχόμενο, μία βάση δεδομένων XML ελαχιστοποιεί επίσης την ανάγκη για εξαγωγή ή εισαγωγή μεταδεδομένων για υποστήριξη της αναζήτησης και της πλοήγησης.

Οι πλέον καταλληλότερες βάσεις για την διαχείριση XML αρχείων είναι οι “εγγενείς” (*native*) βάσεις δεδομένων. Βάσεις τέτοιου τύπου είναι ειδικά κατασκευασμένες για XML αρχεία. Κατά την αποθήκευση των δεδομένων XML σε μια native βάση δεδομένων XML, η εσωτερική αναπαράσταση των δεδομένων διατηρεί τη δομή του αρχικού XML, σε αντίθεση με μία σχεσιακή βάση, η οποία θα κατακερμάτιζε το εν λόγω έγγραφο για την αποθήκευσή του σε σχεσιακούς πίνακες, με αποτέλεσμα αλλαγές στο XML σχήμα να απαιτούν αλλαγές και στην

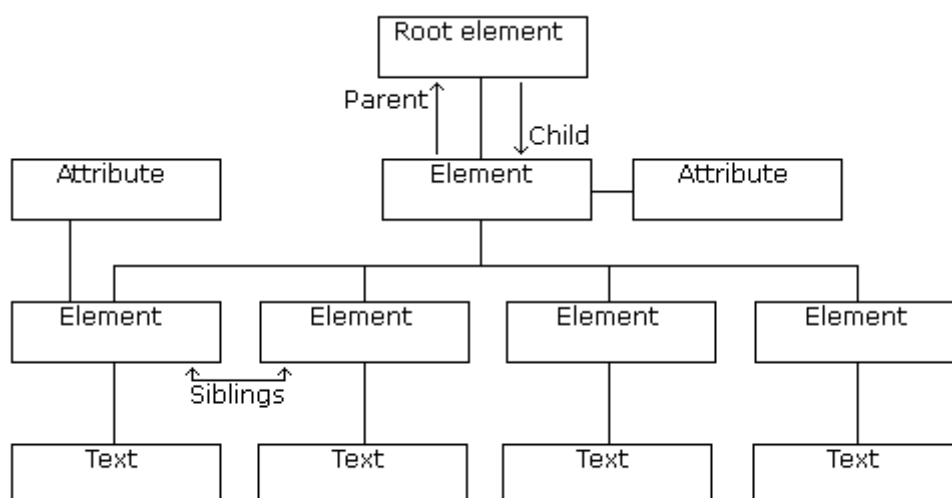
χαρτογράφηση που έγινε προηγουμένως για την αρχική αποθήκευση των κομματιών του XML αρχείου στους πίνακες. Ένα ακόμα παράδειγμα που δείχνει την πως οι native βάσεις υπερτερούν σε σχέση με σχεσιακές για την διαχείριση XML δεδομένων, είναι πως σε μία σχεσιακή βάση υπάρχει μεγάλη πιθανότητα κατά την εισαγωγή ενός μεγάλου XML εγγράφου ή εγγράφου που περιέχει πολλούς διαφορετικούς τύπους οντοτήτων, να δημιουργηθεί μεγάλη επιβάρυνση εξαιτίας της ανάγκης ανάλυσής (*parsing*) του. Αυτή η δαπάνη προέρχεται από το γεγονός ότι πολλές διαφορετικές οντότητες απαιτούν την εισαγωγή τους σε μεγάλο αριθμό διαφορετικών πινάκων. Επίσης, μπορεί να χαθεί η δομή του εγγράφου όταν δεν αποθηκεύονται επιπλέον πληροφορίες σχετικά με τη δομή κατά την ανάλυση του εγγράφου[8].

Σύμφωνα με την W3C, το πρότυπο για τη δημιουργία ερωταπαντήσεων είναι η γλώσσα *XQuery*. Η τελευταία έκδοση είναι το *Xquery 3.1*. Το *Xquery* περιλαμβάνει το *Xpath* ως υπογλώσσα και το ίδιο το XML είναι μια έγκυρη υποσύνταξη του *Xquery*. Σε αντίθεση με τις βάσεις δεδομένων με δυνατότητα XML, οι native βάσεις δεδομένων παρέχουν πλήρη υποστήριξη για το *Xquery*. Εκτός από το *Xpath*, ορισμένες βάσεις δεδομένων XML υποστηρίζουν το *XSLT* ως μέθοδο μετατροπής εγγράφων ή αποτελεσμάτων ερωταπαντήσεων που ανακτώνται από τη βάση δεδομένων.

Σε αυτό το σημείο θα πρέπει να αναλυθεί η το μοντέλο το οποίο ακολουθούν τα XML δεδομένα. Σύμφωνα με την W3C, *“Το μοντέλο δεδομένων για την XML είναι πολύ απλό – ή πολύ αφηρημένο, ανάλογα με την άποψη κάποιου. Η XML παρέχει μόνο μια βασική γραμμή πάνω στην οποία μπορούν να κατασκευαστούν πιο σύνθετα μοντέλα. Ωστόσο, όλες αυτές οι πιο περιορισμένες εφαρμογές θα μοιράζονται μερικές κοινές μεταβλητές”*[6]. Οι κοινές αυτές μεταβλητές θα αναφερθούν στη συνέχεια αυτού του κεφαλαίου.

2.2.1 Δομή δέντρου

Η κύρια δομή ενός εγγράφου XML έχει τη μορφή δέντρου, όπου σε κάθε κόμβο του δέντρου υπάρχουν πολλές ακολουθίες χαρακτήρων. Η δομή του δέντρου και οι συμβολοσειρές χαρακτήρων αποτελούν μαζί το περιεχόμενο ενός εγγράφου XML. Σε γενικές γραμμές η μορφή και το περιεχόμενο των XML αρχείων ακολουθούν την παραπάνω πρόταση. Μερικοί από τους χαρακτήρες του εγγράφου υπάρχουν μόνο για την υποστήριξη της γραμμικοποίησης, άλλοι αποτελούν μέρος του περιεχομένου των πληροφοριών[9]. Η παρακάτω εικόνα κάνει μία αναπαράσταση αυτής της δομής δέντρου.



Εικόνα 2.1: δομή δέντρου της XML

Ένα δέντρο XML ξεκινά από μία “γονεϊκή” οντότητα (*root element*) και διακλαδώνεται από αυτό σε υποοντότητες (*child elements*). Όλες οι υποοντότητες μπορούν να περιέχουν άλλες οντότητες, όπως φαίνεται και στο παρακάτω παράδειγμα:

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

Παράδειγμα 2.1: μορφή ενός δέντρου XML

Οι όροι *γονέας*, *παιδί* και *αδελφός* χρησιμοποιούνται για να περιγράψουν τις σχέσεις μεταξύ των οντοτήτων. Ο γονέας έχει παιδιά και κάθε παιδί έχει έναν γονιό. Επίσης κάθε παιδί μπορεί να έχει “αδέρφια”, δηλαδή οντότητες τα οποία βρίσκονται στο ίδιο επίπεδο με αυτά.

2.2.2 Elements (οντότητες)

Σε κάθε έγγραφο XML επομένως υπάρχει μόνο μία γονεϊκή οντότητα από την οποία προκύπτουν όλα τα υπόλοιπα παιδιά του εγγράφου. Η XML δίνει τη δυνατότητα εισαγωγής χαρακτήρων από διάφορες γλώσσες, αρκεί να καθορίζεται η κωδικοποίηση που χρησιμοποιείται ή να γίνει αποθήκευση του αρχείου ως UTF-8. Αν οριστεί η κωδικοποίηση στο κείμενο, πρέπει να γίνει στην αρχή του, στο *XML prolog*. Το XML prolog είναι προαιρετικό, στην περίπτωση όμως που υπάρχει βρίσκεται πάντα στην αρχή του κειμένου. Συνήθως είναι της μορφής:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Όπως παρατηρείται στο παραπάνω παράδειγμα της δομής ενός XML, κάθε οντότητα πρέπει να έχει ετικέτες έναρξης και κλεισίματος, οι οποίες είναι *case sensitive*, για παράδειγμα η ετικέτα *<Αρχή>* είναι διαφορετική από την ετικέτα *<αρχή>*. Μία οντότητα XML μπορεί να περιέχει απλό κείμενο, attributes (γνωρίσματα), άλλες οντότητες, μία μίξη των τριών ή να είναι κενή. Οι κανόνες για την ονοματοδοσία είναι οι εξής:

- Τα ονόματα των οντοτήτων είναι case sensitive.
- Τα ονόματα πρέπει να ξεκινούν από γράμμα ή με κάτω παύλα (_).
- Τα ονόματα δεν μπορούν να ξεκινήσουν με τα γράμματα xml ή XML ή Xml, κλπ.
- Τα ονόματα μπορούν να περιέχουν γράμματα, ψηφία, παύλες, κάτω παύλες και τελείες.
- Τα ονόματα δεν μπορούν να περιέχουν κενά.
- Εκτός από το όνομα “xml”, μπορεί χρησιμοποιηθεί οποιοδήποτε άλλο όνομα.

2.2.3 Attributes (γνωρίσματα)

Παραπάνω, αναφέρθηκε πως οι οντότητες μπορούν να περιέχουν attributes (γνωρίσματα). Τα γνωρίσματα περιέχουν πληροφορίες σχετικές μόνο για την οντότητα στην οποία περιέχονται. Σε αντίθεση όμως με τις οντότητες δεν γίνεται να περιέχουν πάνω από μία τιμή, δομή δέντρου και δεν είναι εύκολα επεκτάσιμα. Για παράδειγμα:

```
<Αρχή τιμή="ένα">...</Αρχή>
```

Παράδειγμα 2.2: XML attribute

Το τμήμα «τιμή="ένα"» είναι μία έγκυρη δήλωση γνωρίσματος. Τα γνωρίσματα προσφέρουν μεγάλη χρηστικότητα στην αποθήκευση μεταδεδομένων, επομένως η πιο κοινή πρακτική που ακολουθείται είναι η αποθήκευση δεδομένων σε οντότητες, οι οποίες θα περιέχουν γνωρίσματα για την αποθήκευση των μεταδεδομένων τους.

2.2.4 Namespace (χώρος ονομάτων)

Τα namespaces σε ένα XML χρησιμοποιούνται για την παροχή μοναδικά ονομασμένων οντοτήτων και γνωρισμάτων σε ένα έγγραφο XML. Ένα έγγραφο XML μπορεί να περιέχει ονόματα οντοτήτων ή γνωρισμάτων από περισσότερα από ένα «λεξιλόγιο» XML. Αν σε κάθε λεξιλόγιο δίνεται από ένα namespace, μπορεί να επιλυθεί η ασάφεια μεταξύ πανομοιότυπων ονομάτων οντοτήτων ή γνωρισμάτων[10]. Ένα απλό παράδειγμα θα ήταν να εξετάσουμε ένα έγγραφο XML που περιέχει αναφορές σε έναν πελάτη και ένα παραγγελθέν προϊόν. Τόσο η οντότητα του πελάτη όσο και η οντότητα του προϊόντος θα μπορούσαν να έχουν ένα child element με όνομα *id*, κάτι που θα καθιστούσε οποιαδήποτε αναφορά σε αυτή διφορούμενη. Η τοποθέτηση τους σε διαφορετικά namespaces λύνει αυτήν την ασάφεια.

Ένα namespace έχει πάντα τη μορφή ενός *URI*, το οποίο περιγράφει έναν πόρο υπό τον έλεγχο του δημιουργού ή του οργανισμού που καθορίζει το λεξιλόγιο XML, όπως μια διεύθυνση URL για τον διακομιστή Web του δημιουργού. Παρά ταύτα, οι προδιαγραφές του χώρου ονομάτων δεν απαιτούν ούτε προτείνουν να χρησιμοποιηθεί το URI ονομάτων για την ανάκτηση πληροφοριών. Ένας XML parser θα το αντιμετωπίζε ως μία ακολουθία χαρακτήρων. Μία namespace δήλωση είναι της μορφής:

```
xmlns:prefix="URI"
```

όπου το *xmlns* είναι ένα ήδη κατοχυρωμένο XML γνώρισμα και η τιμή *prefix* ένα πρόθεμα το οποίο θα χρησιμοποιηθεί για να γίνει η οποιαδήποτε αναφορά σε οντότητες που ακολουθούν ένα συγκεκριμένο namespace.

Συγκεντρώνοντας όλα τα παραπάνω μπορούμε να δημιουργήσουμε ένα παράδειγμα ενός έγκυρου XML εγγράφου:

```
<root xmlns:h="http://www.w3.org/TR/html4/"  
xmlns:f="https://www.w3schools.com/furniture">
```

```
<h:table>  
  <h:tr>  
    <h:td>Apples</h:td>  
    <h:td>Bananas</h:td>  
  </h:tr>  
</h:table>
```

```

<f:table>
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>

</root>

```

Παράδειγμα 2.3: ένα απλό έγγραφο XML

2.2.5 DTD και XML Schema

Συνήθως λέμε πως ένα XML αρχείο είναι “καλά διαμορφωμένο” όταν έχει σωστή σύνταξη. Τις περισσότερες φορές τα XML έγγραφα ακολουθούν κάποιους περιορισμούς που δίνονται από ένα σύνολο δηλώσεων σήμανσης που δίνουν κάποιους περιορισμούς στη δομή των οντοτήτων και γνωρισμάτων που μπορεί να έχει ένα XML έγγραφο. Όταν λοιπόν ένα έγγραφο ακολουθεί αυτούς τους περιορισμούς χωρίς καμία παράβαση, λέμε πως το έγγραφο είναι “έγκυρο”. Οι δύο πιο διαδεδομένοι ορισμοί τύπων εγγράφων που μπορούν να χρησιμοποιηθούν με την XML είναι το DTD (Document Type Definition) και το XML Schema, με το δεύτερο να έχει ξεπεράσει κατά πολύ το DTD. Επομένως δεν θα γίνει εκτενέστερη ανάλυσή του. Αντιθέτως θα γίνει πάνω στο XML Schema, το οποίο χρησιμοποιείται στην διπλωματική εργασία για την επικύρωση των XML εγγράφων.

Το XML Schema ή XSD (XML Schema Definition), είναι ο προτεινόμενος ορισμός της W3C και καθορίζει, όπως και κάθε ορισμός τύπων εγγράφων, τον τρόπο επίσημης περιγραφής των οντοτήτων και γνωρισμάτων σε ένα έγγραφο και μπορεί να χρησιμοποιηθεί από προγραμματιστές για να επαληθεύσει κάθε κομμάτι περιεχομένου ενός εγγράφου αν τηρεί την περιγραφή της οντότητας στο οποίο είναι τοποθετημένο. Όπως όλες οι γλώσσες σχήματος XML, το XSD μπορεί να χρησιμοποιηθεί για να εκφράσει ένα σύνολο κανόνων στους οποίους ένα έγγραφο XML πρέπει να συμμορφώνεται, προκειμένου να θεωρηθεί «έγκυρο» σύμφωνα με το σχήμα[11]. Ωστόσο, σε αντίθεση με τις περισσότερες άλλες γλώσσες σχημάτων, το XSD σχεδιάστηκε επίσης με την πρόθεση ότι ο προσδιορισμός της εγκυρότητας ενός εγγράφου θα παρήγαγε μια συλλογή πληροφοριών που θα τηρούν συγκεκριμένους τύπους δεδομένων. Τέτοιες πληροφορίες μπορεί να είναι χρήσιμες για την ανάπτυξη λογισμικού επεξεργασίας εγγράφων XML.

Ορισμένα από τα πλεονεκτήματα του XSD έναντι των υπολοίπων σχημάτων είναι:

- Το XSD είναι βασισμένη στην XML και προσφέρει περισσότερες δυνατότητες από το DTD.
- Μία από τις μεγαλύτερες δυνατότητες της XSD είναι πως υποστηρίζει τους τύπους δεδομένων, επομένως είναι ευκολότερο να περιγράψουμε το επιτρεπόμενο περιεχόμενο των εγγράφων, να ελέγξουμε την ορθότητα των δεδομένων, να γίνει μετατροπή δεδομένων μεταξύ διαφορετικών τύπων και να οριστούν περιορισμοί καθώς και πρότυπα δεδομένων.
- Τα XSD αρχεία είναι γραμμένα σε XML, επομένως δεν χρειάζεται η εκμάθηση κάποιας νέας γλώσσας, μπορεί να γίνει parsing από κάποιον XML parser και υπάρχει η δυνατότητα επεκτασιμότητας.
- Προσφέρει ασφαλή επικοινωνία δεδομένων χωρίς να υπάρχει κίνδυνος σύγχυσης μεταξύ αποστολέα και αποδέκτη.

Σε κάθε XML Schema η γονεϊκή οντότητα είναι η *<schema>* και μέσα σε αυτό βρίσκονται όλες οι υπόλοιπες. Η οντότητα *<schema>* μπορεί να περιέχει και γνωρίσματα. Ένα παράδειγμα σχήματος παρουσιάζεται στη συνέχεια:

```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="https://www.w3schools.com"
  xmlns="https://www.w3schools.com"
  elementFormDefault="qualified">
  ...
  ...
</xs:schema>
```

Παράδειγμα 2.4: ένα XML Schema έγγραφο

Το κομμάτι: *xmlns:xs="http://www.w3.org/2001/XMLSchema"*, υποδηλώνει ότι οι οντότητες και οι τύποι δεδομένων που χρησιμοποιούνται στο σχήμα προέρχονται από το namespace «*http://www.w3.org/2001/XMLSchema*» και πως πρέπει να φέρουν πρόθεμα *xs*. Το τμήμα *TargetNamespace = "https://www.w3schools.com"* υποδηλώνει πως όλες οι οντότητες που καθορίζονται από το σχήμα προέρχονται από αυτό το namespace, με τη δήλωση *xmlns="https://www.w3schools.com"* να καθορίζει πως αυτό είναι το προκαθορισμένο namespace. Τέλος, το *elementFormDefault="qualified"* καθορίζει πως όποιες οντότητες που χρησιμοποιούνται από ένα XML έγγραφο που έχουν δηλωθεί στο σχήμα θα πρέπει να ακολουθούν το namespace.

Η XML Schema μας δίνει τη δυνατότητα να καθορίσουμε το πλήθος και είδος των οντοτήτων και γνωρισμάτων που θα έχουν τα XML έγγραφα που ακολουθούν τους κανόνες ενός XSD, για παράδειγμα μπορούμε να καθορίσουμε πως η οντότητα *<date>...</date>* θα εμφανίζεται το πολύ μία φορά στο έγγραφο και οι μόνες πληροφορίες που θα μπορούν να αποθηκευτούν σε αυτή θα είναι ημερομηνίες και δομημένες ως τέτοιες, δηλαδή θα έχουν τη μορφή *Ημέρα-Μήνας-Έτος*. Κάτι τέτοιο είναι πολύ χρήσιμο, μιας και γνωρίζοντας πως τα έγγραφα XML μίας βάσης θα ακολουθούν κάποια συγκεκριμένα πρότυπα μπορούμε γράψουμε τις διάφορες ερωταπαντήσεις προς τη βάση ώστε να έχουν τη μεγαλύτερη δυνατή αποτελεσματικότητα.

2.2.6 Η XQuery

Η Xquery είναι η γλώσσα στην οποία γράφονται οι ερωταπαντήσεις προς μία native βάση δεδομένων. Βασίζεται στο μοντέλο δεδομένων XPath, το οποίο χρησιμοποιεί μία δομή δέντρου για το περιεχόμενο των πληροφοριών ενός εγγράφου XML και περιέχει επτά είδη κόμβων: κόμβους εγγράφων, οντότητες, γνωρίσματα, κόμβους κειμένου, σχόλια, οδηγίες επεξεργασίας και namespaces. Διερευνά και μετατρέπει βάσεις δεδομένων και αδόμητων δεδομένων συνήθως σε XML, κείμενο και με εξειδικευμένες επεκτάσεις σε άλλους τύπους δεδομένων (JSON, δυαδικά δεδομένα κλπ.). Η γλώσσα αναπτύσσεται από την ομάδα εργασίας XML Query της W3C. Το έργο συντονίζεται στενά με την ανάπτυξη του XSLT από την Ομάδα Εργασίας XSL. οι δύο ομάδες μοιράζονται την ευθύνη για το XPath, το οποίο είναι ένα υποσύνολο του XQuery. Το βασικό δομικό κομμάτι της XQuery είναι η *έκφραση*, που είναι μία συμβολοσειρά χαρακτήρων Unicode. Η έκδοση του Unicode που πρόκειται να χρησιμοποιηθεί είναι καθορισμένη από την εφαρμογή. Η γλώσσα παρέχει διάφορες μορφές εκφράσεων οι οποίες μπορούν να κατασκευαστούν από λέξεις-κλειδιά, σύμβολα και τελεστές. Γενικά, οι τελεστές μιας έκφρασης είναι άλλες εκφράσεις. Η XQuery επιτρέπει σε εκφράσεις να είναι φωλιασμένες με πλήρη γενικότητα. Ωστόσο, σε αντίθεση με μια πλήρως

λειτουργική γλώσσα δεν επιτρέπει την αντικατάσταση μεταβλητών εάν μία μεταβλητή περιέχει κατασκευή νέων κόμβων. Όπως και στην XML, έτσι και στην *XQuery* παίζει ρόλο αν κάποια μεταβλητή είναι γραμμένη με κεφαλαία γράμματα ή μικρά. Περιέχει μια σύνθετη σύνταξη XPath για τη διεύθυνση συγκεκριμένων τμημάτων ενός εγγράφου XML. Αυτό συμπληρώνεται από μία έκφραση *FLWOR* για την εκτέλεση συνδέσεων. Μια έκφραση *FLWOR* κατασκευάζεται από τις πέντε όρους, από τους οποίους κιάλας έχει πάρει το όνομά της: *FOR*, *LET*, *WHERE*, *ORDER BY*, *RETURN*.

Επίσης παρέχεται από τη γλώσσα σύνταξη που επιτρέπει την κατασκευή νέων εγγράφων XML. Όταν τα ονόματα των οντοτήτων και των γνωρισμάτων είναι γνωστά εκ των προτέρων, μπορεί να χρησιμοποιηθεί σύνταξη τύπου XML. Σε άλλες περιπτώσεις, είναι διαθέσιμες εκφράσεις που αναφέρονται ως *δυναμικοί κατασκευαστές κόμβων*. Αυτή η δυνατότητα, για την κατασκευή, την ενημέρωση των εγγράφων ή των βάσεων δεδομένων XML και η δυνατότητα αναζήτησης πλήρους κειμένου δεν αποτελούν μέρος της βασικής γλώσσας αλλά καθορίζονται στα πρόσθετες επεκτάσεις. Μέχρι αυτή την στιγμή η τελευταία έκδοση της *XQuery* είναι η *XQuery 3.1*[12].

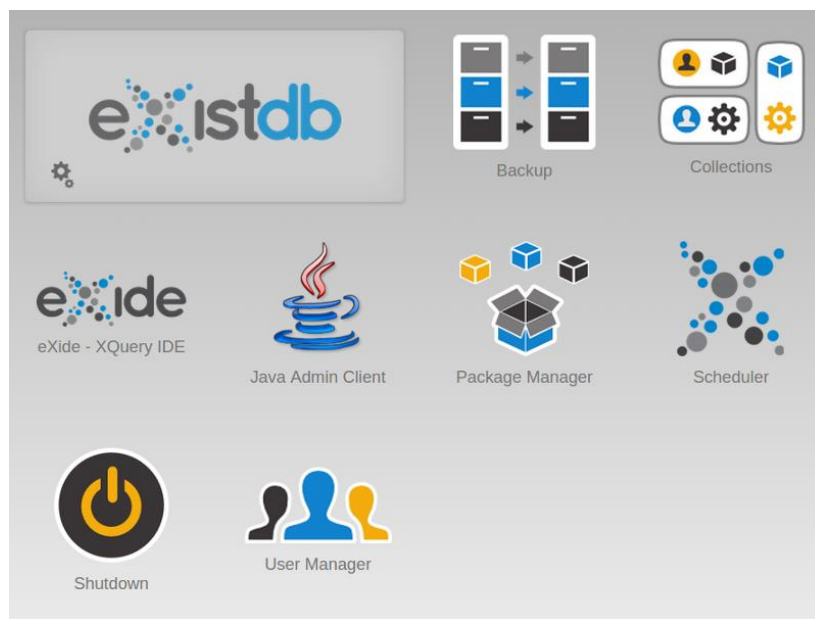
2.3 Η 'eXistdb'

Αφού έγινε η απαραίτητη ανάλυση μελέτη γύρω από τη γλώσσα XML, των δυνατοτήτων που αυτή προσφέρει και της γλώσσας στην οποία είναι κατασκευασμένες οι ερωταπαντήσεις, είναι πλέον ικανή η κατανόηση της native βάσης δεδομένων *eXistdb* η οποία χρησιμοποιείται στην παρούσα διπλωματική εργασία.

Η *eXistdb* είναι όπως προαναφέρθηκε μία native βάση δεδομένων ανοιχτού λογισμικού και σε αντίθεση με τα περισσότερα σχεσιακά συστήματα διαχείρισης βάσεων δεδομένων καθώς και άλλων NoSQL βάσεων παρέχει τις *XQuery* και *XSLT* ως γλώσσα ερωταπαντήσεων και εφαρμογών αντίστοιχα. Κατασκευάστηκε το 2000 από τον *Wolfgang Meier*, με συχνές ενημερώσεις νέων εκδόσεων, με την πιο πρόσφατη μέχρι αυτή την στιγμή να είναι η έκδοση *eXist-db v4.0.0* και να βρίσκετε μέσα στις πέντε πρώτες πιο χρησιμοποιημένες native βάσεις δεδομένων[13].

Η εγκατάστασή της σε ένα σύστημα υπολογιστών είναι αρκετά απλή και υπάρχουν αναλυτικές οδηγίες στην ιστοσελίδα της *eXist*. Είναι συμβατή με όλες τις πρόσφατες εκδόσεις των λειτουργικών Linux, MacOS και Windows, με ελάχιστες απαιτήσεις έχει 512 MB μνήμης RAM και περίπου 200 MB χώρου στο δίσκο.

Μετά την εγκατάστασή της, η επεξεργασία και οι οποιοσδήποτε διεργασίες στην βάση μπορούν να γίνουν μέσω της εφαρμογής *Dashboard*, η οποία ανοίγει με μία εφαρμογή φυλλομετρητή (*web browser*). Αυτή η εφαρμογή είναι το κέντρο εκκίνησης και εκτέλεσης εργασιών διαχείρισης της βάσης. Όπως στην αρχική οθόνη ενός tablet ή smartphone, το *Dashboard* εμφανίζει μια λίστα εφαρμογών ή "εφαρμογές".



Εικόνα 2.2: To Dashboard της eXistdb

Το Dashboard υποστηρίζει και τις "εφαρμογές" (apps) όπως και τις "προσθήκες" (plugins). Οι εφαρμογές είναι αυτοτελείς εφαρμογές που παρέχουν το δικό τους διαδικτυακό γραφικό περιβάλλον χρήστη ενώ τα "plugins" τρέχουν μέσα στο Dashboard. Παραδείγματα εφαρμογών είναι οι οδηγίες χρήσης της eXistdb (eXistdb documentation), το eXide ή η το demo app. Παραδείγματα για plugins είναι ο διαχειριστής πακέτων (packet manager) ή ο περιηγητής συλλογών (collection browser). Οι προσθήκες είναι οι πλέον κατάλληλες για διαχειριστικές λειτουργίες.

Εκτός από τις εφαρμογές και τα πρόσθετα, το αποθετήριο πακέτων παρέχει επίσης πακέτα βιβλιοθηκών και πόρων. Επειδή όμως δεν παρέχουν διαδικτυακό γραφικό περιβάλλον δεν είναι ορατά στην αρχική οθόνη του Dashboard. Ωστόσο εμφανίζονται στη λίστα εγκατεστημένων πακέτων μέσα στον διαχειριστή πακέτων. Για να χρησιμοποιήσετε πλήρως τα πρόσθετα διαχείρισης, απαιτείται σύνδεση στη βάση ως χρήστης *dba*. Οι *dba* χρήστες έχουν δικαιώματα διαχειριστή (Administrator privileges). Αν και υπάρχει η δυνατότητα απόκτησης πρόσβασης στις προσθήκες ως χρήστης εκτός του *dba group*, οποιεσδήποτε αλλαγές για να εφαρμοστούν θα ζητήσουν την εκ νέου σύνδεση με λογαριασμό χρήστη που ανήκει στην προαναφερθείσα ομάδα χρηστών.

Η βάση δεδομένων eXistdb προσφέρει μία πληθώρα υποστηριζόμενων τεχνολογιών και προτύπων τα οποία δεν υπάρχει κάποια χρησιμότητα στο να γίνει κάποια αναφορά σε αυτά, εκτός του *XML-RPC*, του πρωτοκόλλου απομακρυσμένης επικοινωνίας. Ο λόγος για αυτή την ενέργεια, είναι πως το προαναφερθέν πρωτόκολλο θα αξιοποιηθεί για την επικοινωνία μεταξύ της βάσης δεδομένων και του *frondend*, το οποίο θα χρησιμοποιεί *HTML*, *PHP* και ορισμένες συναρτήσεις σε *javascript*.

Το πρωτόκολλο *XML-RPC*, χρησιμοποιεί XML για την κωδικοποίηση των κλήσεων και το πρωτόκολλο HTTP ως μηχανισμού μεταφοράς των κλήσεων. Σύμφωνα με τις οδηγίες στην ιστοσελίδα της eXist, "η διεπαφή προγραμματισμού εφαρμογών *XML-RPC* διευκολύνει την πρόσβαση στην eXistdb από άλλες εφαρμογές, CGI κωδίκων, PHP, JSP και πολλές άλλες".

Ο τρόπος λειτουργίας του πρωτοκόλλου είναι ως εξής: το *XML-RPC* λειτουργεί στέλνοντας ένα αίτημα HTTP σε ένα διακομιστή που υλοποιεί το πρωτόκολλο. Ο πελάτης σε αυτή την περίπτωση είναι συνήθως λογισμικό που θέλει να καλέσει μία μέθοδο ενός

απομακρυσμένου συστήματος. Πολλές παράμετροι εισόδου μπορούν να μεταβιβαστούν στην απομακρυσμένη μέθοδο με μόνο μία επιστρέφεται ως έξοδος. Οι τύποι παραμέτρων επιτρέπουν τον φωλιασμό των παραμέτρων σε χάρτες και λίστες, παρέχοντας έτσι την δυνατότητα μεταφοράς μεγαλύτερων δομών. Επομένως, το XML-RPC μπορεί να χρησιμοποιηθεί για τη μεταφορά αντικειμένων ή δομών τόσο ως παραμέτρους εισόδου όσο και εξόδου. Η αναγνώριση των πελατών για σκοπούς εξουσιοδότησης μπορεί να επιτευχθεί χρησιμοποιώντας δημοφιλείς μεθόδους ασφαλείας του HTTP. Για την ταυτοποίηση γίνεται ένας βασικός έλεγχος πρόσβασης, με το HTTPS να χρησιμοποιείται όταν απαιτούνται κρυπτογραφημένα μηνύματα και αναγνώριση (μέσω πιστοποιητικών). Υπάρχει ωστόσο η δυνατότητα συνδυασμού και των δύο μεθόδων.

2.4 Σύνοψη

Σε αυτό το κεφάλαιο έγινε μια ανάλυση της γλώσσας σήμανσης XML, των δυνατοτήτων της καθώς και κάποιοι βασικοί κανόνες σύνταξης που ακολουθεί. Στη συνέχεια παρουσιάστηκαν οι βάσεις δεδομένων ειδικά σχεδιασμένες για την διαχείριση εγγράφων γραμμένα σε XML, οι native βάσεις δεδομένων και της γλώσσας ερωταπαντήσεων για την εκτέλεση διεργασιών ανάκλησης ή μετατροπής των εν λόγω αρχείων. Τέλος, παρουσιάστηκε η eXistdb, η native βάση που επιλέχθηκε για την αποθήκευση των πληροφοριών των τρισδιάστατων αγγείων.

Στο επόμενο κεφάλαιο, θα παρουσιαστεί η HTML, η κύρια γλώσσα σήμανσης του διαδικτύου και ο τρόπος σύνδεσης, η PHP, μια προγραμματιστική γλώσσα για τη δημιουργία δυναμικών διαδικτυακών σελίδων και ο τρόπος με τον οποίο χρησιμοποιεί το XML-RPC πρωτόκολλο για την κλήση των πληροφοριών από τα XML έγγραφα και την αναπαράστασή τους στην HTML σελίδα.

Κεφάλαιο 3ο: Εργαλεία κατασκευής διεπαφής χρήστη(frontend)

3.1 Εισαγωγή

Ένα σύστημα *frontend* είναι μέρος ενός συστήματος πληροφοριών που είναι άμεσα προσβάσιμο και αλληλοεπιδρά με τον χρήστη για να λαμβάνει ή να χρησιμοποιεί τις δυνατότητες του *backend* του κεντρικού συστήματος. Επιτρέπει στους χρήστες να έχουν πρόσβαση και να αιτούνται λειτουργίες και υπηρεσίες από το σύστημα πληροφοριών. Το σύστημα *frontend* μπορεί να είναι μια εφαρμογή λογισμικού ή ο συνδυασμός υλικού, λογισμικού και πόρων δικτύου. Συνήθως ένα σύστημα *frontend* χρησιμοποιείται κυρίως για την αποστολή ερωταπαντήσεων και αιτημάτων και τη λήψη δεδομένων από το σύστημα *backend* ή το σύστημα πληροφοριών κεντρικού υπολογιστή. Εξυπηρετεί ή παρέχει στους χρήστες τη δυνατότητα αλληλεπίδρασης και χρήσης ενός συστήματος πληροφοριών. Τυπικά, τα συστήματα *frontend* έχουν πολύ περιορισμένες δυνατότητες επεξεργασίας υπολογιστικής ή επιχειρησιακής λογικής και βασίζονται στα δεδομένα και τις λειτουργίες από το σύστημα κεντρικού υπολογιστή. Ωστόσο, ορισμένα προηγμένα συστήματα *frontend* διατηρούν αντίγραφα των δεδομένων, όπως ένα αντίγραφο κάθε συναλλαγής που αποστέλλεται στο σύστημα *backend*.

Στην παρούσα διπλωματική για την υλοποίηση του frontend χρησιμοποιήθηκε η γλώσσα HTML, η PHP καθώς και ορισμένες συναρτήσεις γραμμένες σε javascript και επομένως θα ακολουθήσει μία παρουσίαση αυτών των γλωσσών ώστε να είναι πιο κατανοητή η λειτουργία των διαφόρων κωδίκων που αποτελούν την βάση δεδομένων καθώς και του frontend.

3.2 Η γλώσσα HTML

Η HTML (*HyperText Markup Language*) είναι η παγκόσμια γλώσσα που χρησιμοποιείται για έκδοση υπερκειμένου στον Παγκόσμιο Ιστό (*World Wide Web*). Με τον όρο υπερκείμενο (*hypertext*) καλούμαστε την πληροφορία που, κατευθείαν από την οπτική της αναπαράσταση, μπορούμε να κατευθυνθούμε σε άλλη πληροφορία σχετική με αυτήν. Η HTML είναι βασισμένη στην SGML (*Standard Generalized Markup Language*), μία γλώσσα για τον ορισμό της δομής ηλεκτρονικών κειμένων σε ηλεκτρονική μορφή και τη διαχείρισή τους. Η δημιουργία και επεξεργασία της γίνεται με ποικιλία εργαλείων, από απλούς συντάκτες (editors) μέχρι εξειδικευμένα προγράμματα συγγραφής[14]. Το 1980, ο φυσικός *Tim Berners-Lee*, ένας ανάδοχος έργου που δούλευε στο CERN (*Conseil Européen pour la Recherche Nucléaire*), πρότεινε και πρωτοτύπησε το INQUIRE, ένα σύστημα για τους ερευνητές του CERN έτσι ώστε να χρησιμοποιούν και να μοιράζονται έγγραφα. Το 1989, ο *Berners-Lee* έγραψε ένα σημείωμα που πρότεινε ένα σύστημα υπερκειμένου με βάση το Διαδίκτυο. Έτσι, προσδιόρισε την HTML και έγραψε το πρόγραμμα περιήγησης και το λογισμικό διακομιστή στα τέλη του 1990. Το έτος αυτό, ο προαναφερθείς μαζί με τον *Robert Cailliau*, μηχανικός συστημάτων δεδομένων του CERN, συνεργάστηκαν ώστε να τους δοθεί κοινή χρηματοδότηση από το CERN, κάτι που δεν εγκρίθηκε ποτέ επισήμως.

Η πρώτη δημόσια διαθέσιμη περιγραφή της HTML ήταν ένα έγγραφο που ονομαζόταν “HTML Tags “. Περιγράφει 18 στοιχεία που περιέγραφαν τον αρχικό, σχετικά απλό σχεδιασμό του HTML. Εκτός από την ετικέτα υπερσυνδέσμου, αυτές επηρεάστηκαν έντονα από το SGMLguid, μια εσωτερική μορφή βασισμένη στη γλώσσα SGML (*Standard Generalized Markup Language*) που φτιάχτηκε στο CERN. Έντεκα από αυτά τα στοιχεία εξακολουθούν να υπάρχουν στην HTML 4.

Αρχικά, ο δημιουργός της HTML Berners-Lee την θεώρησε ως μια εφαρμογή της SGML, κάτι που ορίστηκε και επίσημα από την *Internet Engineering Task Force (IETF)* το 1993 με τη δημοσίευση της πρώτης πρότασης για μια προδιαγραφή της HTML, με τίτλο "*Hypertext Markup Language (HTML)*". Αυτή περιλάμβανε και έναν Ορισμό τύπου εγγράφου (*DTD*) της SGML, ο οποίος όριζε την γραμματική. Το σχέδιο οστόσο έληξε μετά την πάροδο έξι μηνών, αλλά ήταν αξιοσημείωτο το γεγονός ότι αναγνώρισε την προσαρμοσμένη ετικέτα του NCSA Mosaic φυλλομετρητή για την ενσωμάτωση εικόνων στο κείμενο, αντικατοπτρίζοντας τη φιλοσοφία του *IETF* να στηρίζει πρότυπα σε επιτυχημένα πρωτότυπα. Κάτι παρόμοιο πρότεινε και ο *Dave Raggett* στο έγγραφό του "HTML+ (Hypertext Markup Format)" στα τέλη του 1993. Πρότεινε την προτυποποίηση μερικών ήδη υλοποιημένων δυνατοτήτων, όπως οι πίνακες και οι φόρμες. Μετά την λήξη των σχεδίων *HTML* και *HTML+* στις αρχές του 1994, η *IETF* δημιούργησε μια ομάδα εργασίας πάνω στην HTML, η οποία ολοκλήρωσε την πρώτη προδιαγραφή HTML το 1995, την "HTML 2.0" και προοριζόταν να αντιμετωπιστεί ως πρότυπο πάνω στο οποίο θα βασίζονταν οποιεσδήποτε μελλοντικές εφαρμογές. Μέχρι αυτή την στιγμή, η *HTML5* αποτελεί την πιο πρόσφατη έκδοση της HTML.

3.2.1 Ετικέτες (*Tags*)

Ετικέτα ονομάζουμε μία οδηγία γραμμένη σε HTML, την οποία πρέπει να αναγνωρίσει και ερμηνεύσει ο φυλλομετρητής. Οι ετικέτες βρίσκονται πάντοτε μεταξύ των συμβόλων "<" και ">". Πρέπει να γίνει αντιληπτό πως τα αρχεία της HTML είναι απλά αρχεία κειμένου στα οποία υπάρχουν ειδικοί κώδικες, οι ετικέτες (*tags*), που δηλώνουν στον φυλλομετρητή τον τρόπο με τον οποίο θα εμφανιστούν τα αρχεία αυτά. Οι ετικέτες έχουν την παρακάτω γενική δομή:

<όνομα ετικέτας>κείμενο</όνομα ετικέτας>

Παράδειγμα 3.1: HTML Tags

Με το *όνομα ετικέτας* δηλώνεται η ενέργεια που θα εκτελεστεί στο κείμενο. Για να δηλώσουμε το σημείο που θα σταματήσει να εκτελείτε η ενέργεια αυτή χρησιμοποιούμε την αντίστοιχη ετικέτα κλεισίματος </ *όνομα ετικέτας* >. Υπάρχουν περιπτώσεις ετικετών HTML που δεν έχουν ετικέτα κλεισίματος, π.χ. όταν θέλουμε να εισάγουμε μία εικόνα απλά δηλώνουμε την ενέργεια αυτή με την αντίστοιχη HTML ετικέτα. Η ετικέτα κλεισίματος στην περίπτωση αυτή δεν έχει κανένα νόημα. Αντίθετα, όταν θέλουμε να κάνουμε ένα κείμενο να έχει έντονη γραφή τότε πρέπει να συμπεριλάβουμε το κείμενο εντός της αντίστοιχης HTML ετικέτας. Στο *όνομα ετικέτας* δε γίνεται διάκριση ανάμεσα σε πεζούς και κεφαλαίους χαρακτήρες. Οποιοσδήποτε συνδυασμός είναι αποδεκτός Π.χ. οι ετικέτες <html>, <HTML> ή <HtMl> είναι ισοδύναμες. Αυτός ο συνδυασμός ετικετών και περιεχομένου αποτελούν μία οντότητα HTML (*HTML element*). Δεν πρέπει να γίνεται σύγχυση με τις οντότητες της XML, αν και έχουν κάποια κοινά γνωρίσματα.

Όπως και στην XML, έτσι και στην HTML οι οντότητες μπορούν να είναι εφωλιασμένες.

3.2.2 Μία απλή σελίδα σε HTML

Ένα αρχείο HTML περικλείεται πάντα μέσα μεταξύ των ετικετών αρχής και τέλους `<HTML>...</HTML>`. Δηλαδή, η οντότητα `<HTML>` καθορίζει πως όλο το έγγραφο είναι ένα HTML κείμενο. Αποτελείται από δύο ξεχωριστά μέρη, την οντότητα `<HEAD>` στην οποία περιέχονται πληροφορίες που αφορούν εσωτερικές λειτουργίες του αρχείου (μεταδεδομένα) και την οντότητα `<BODY>`, στο οποίο περιέχεται το περιεχόμενο της σελίδας. Πριν δείξουμε τα παραπάνω με ένα παράδειγμα, θα γίνει αναφορά σε μία άλλη οντότητα, την `<TITLE>`, που εμφανίζει τον τίτλο στην μπάρα τίτλου του φυλλομετρητή. Η παραπάνω οντότητα δεν εμφανίζει κάτι στην κεντρική σελίδα και βρίσκεται μέσα στην `<HEAD>`. Ένα αρχείο HTML είναι ένα αρχείο κειμένου (ASCII), επομένως για τη δημιουργία ενός HTML εγγράφου αρκεί να ανοίξουμε έναν επεξεργαστή κειμένου, να πληκτρολογήσουμε το κείμενό, να το αποθηκεύσουμε σε ένα αρχείο με κατάληξη `.htm` ή `.html` και στη συνέχεια να το εμφανίσουμε κάνοντας χρήση ενός φυλλομετρητή. Παρακάτω δίνεται ο κώδικας σε HTML του παραδείγματος.

```
<html>
  <head>
    <title>
      Τίτλος
    </title>
  </head>
  <body>
    Κείμενο προς εμφάνιση
  </body>
</html>
```

Παράδειγμα 3.2: συνήθης μορφή ενός HTML εγγράφου

3.2.3 Γνωρίσματα (Attributes)

Όπως και με τις XML οντότητες, έτσι και οι HTML οντότητες μπορούν να περιέχουν γνωρίσματα τα οποία προσφέρουν περαιτέρω πληροφορίες για αυτές. Ορίζονται πάντα στην ετικέτα έναρξης και είναι συνήθως της μορφής `όνομα="τιμή"`. Παρακάτω αναφέρονται ορισμένα τέτοια γνωρίσματα.

- Το *href* γνώρισμα: στην HTML, οι διάφοροι σύνδεσμοι (links) ορίζονται με την ετικέτα `<a>`, με τη διεύθυνση του συνδέσμου να αναγράφεται στο γνώρισμα *href*. Για παράδειγμα η οντότητα `αυτός είναι ένας σύνδεσμος` θα εμφανίσει στη σελίδα την πρόταση «αυτός είναι ένας σύνδεσμος», ο οποίος όταν επιλεγεί από ένα χρήστη θα τον ανακατευθύνει στην κεντρική σελίδα της Google.
- Το *src* γνώρισμα: χρησιμοποιείται για τον καθορισμό του ονόματος ενός αρχείου εικόνας, π.χ. ``
- Τα *width* και *height*: όπως φαίνεται και από τα ονόματά τους, καθορίζουν το πλάτος και ύψος αντίστοιχα μίας εικόνας σε *pixels*. Π.χ. ``
- Γνώρισμα *style*: καθορίζει τα διάφορα χαρακτηριστικά της οντότητας όπως το χρώμα της, το μέγεθός της και ούτω καθεξής. Π.χ. η παρακάτω οντότητα

`<p style=»color:red»>...</p>` όταν εμφανιστεί σε έναν φυλλομετρητή θα έχει κόκκινο χρώμα.

- Το *lang* γνώρισμα: χρησιμοποιείται για την δήλωση της γλώσσας στην οποία είναι γραμμένο το κείμενο που περικλείεται στο HTML έγγραφο. Η οποιαδήποτε δήλωση γλώσσας γίνεται στην ετικέτα `<HTML>` και χρήζει ιδιαίτερης σημασίας για μηχανές αναζήτησης και εφαρμογές προσβασιμότητας, όπως είναι εφαρμογές ανάγνωσης της οθόνης για άτομα με προβλήματα όρασης. Π.χ. `<html lang=»en-US»>`, με τα δύο πρώτα γράμματα να καθορίζουν τη γλώσσα (*en*) και στην περίπτωση ύπαρξης διαλέκτου, χρησιμοποιούνται δύο επιπλέον γράμματα (*US*).

3.2.4 Επικεφαλίδες (Headings)

Οι επικεφαλίδες, όπως φαίνεται και από το όνομά τους, χρησιμοποιούνται για τον καθορισμό των επικεφαλίδων στο κείμενο και ανάλογα με το πια ετικέτα επικεφαλίδας επιλέγεται αλλάζει και η μορφοποίησή τους. Οι ετικέτες επικεφαλίδας είναι οι `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` και `<h6>` με την `<h1>` να ορίζει τον πιο σημαντικό τίτλο και την `<h6>` τον λιγότερο σημαντικό.

Η χρήση επικεφαλίδων είναι σημαντική για την εμφάνιση της δομής του εγγράφου. Οι μηχανές αναζήτησης χρησιμοποιούν τις επικεφαλίδες για την κατάταξη της δομής και του περιεχομένου των ιστοσελίδων. Οι επικεφαλίδες `<h1>` θα πρέπει να χρησιμοποιούνται για τις κύριες επικεφαλίδες, ακολουθούμενες από επικεφαλίδες `<h2>`, τις λιγότερο σημαντικές `<h3>` και ούτω καθεξής.

Εκτός από τις παραπάνω επικεφαλίδες υπάρχουν και αυτές με την ετικέτα `<hr>` οι οποίες συνήθως ορίζουν μία θεματική αλλαγή στην HTML σελίδα και αναπαρίστανται από τον φυλλομετρητή ως μία οριζόντια γραμμή.

Τέλος, αξίζει να σημειωθεί πως η οντότητα `<HEAD>` ανήκει και αυτό στις επικεφαλίδες αν και διαφέρει ουσιαστικά από τις άλλες επικεφαλίδες, μιας και περιέχει όπως αναφέρθηκε μεταδεδομένα για το HTML αρχείο.

3.2.5 Παράγραφοι HTML

Η HTML αγνοεί τα κενά διαστήματα ή τις αλλαγές γραμμής. Για τη δημιουργία αυτόνομων τμημάτων κειμένου γίνεται χρήση διαχωριστικών. Οι παράγραφοι διαχωρίζουν το κείμενο μέσω της εμφάνισης μίας κενής γραμμής και δηλώνονται μέσω της ετικέτας `<p>`. Στις επικεφαλίδες δεν είναι απαραίτητη η χρήση της ετικέτας παραγράφου, αφού η επικεφαλίδα είναι από μόνη της ένα ξεχωριστό κομμάτι κειμένου. Τέλος, αν και η ετικέτα τερματισμού `</p>` είναι προαιρετική, καλό είναι να χρησιμοποιείται γιατί σε περίπτωση που χρησιμοποιείται κάποια μέθοδος στοίχισης κειμένου η παρουσία της είναι υποχρεωτική. Παρόμοια λειτουργία με την `<p>` έχει και η ετικέτα `
`, η οποία είναι ετικέτα αλλαγής γραμμής.

3.2.6 Μορφοποίηση με την χρήση CSS αρχείων

Τα αρχεία CSS (*Cascading Style Sheets*), περιγράφουν το πως θα εμφανίζονται οι οντότητες HTML σε μία οθόνη ή/και διαφορετική αναπαράσταση ανάλογα με το μέγεθος της οθόνης ή της συσκευής στην οποία θα εμφανιστεί η ιστοσελίδα. Υπάρχουν τρεις τρόποι για την εισαγωγή ενός CSS στο HTML κείμενο, α) μέσα στις διάφορες οντοτητες με την χρήση του γνωρίσματος *style*, β) με την χρήση ενός πεδίου `<style>` μέσα στο `<head>` και γ) χρησιμοποιώντας ένα εξωτερικό αρχείο CSS.

Ο επικρατέστερος και βέλτιστος τρόπος εισαγωγής μορφοποίησης είναι ο τρίτος, η χρήση ενός εξωτερικού αρχείου, μιας και μόνο με ένα τέτοιο αρχείο παρέχεται η δυνατότητα μορφοποίησης πολλών αρχείων HTML χωρίς να υπάρχει η ανάγκη γραφής του σε κάθε ένα από αυτά. Η χρήση του σε ένα έγγραφο είναι απλή, το μόνο που χρειάζεται είναι η προσθήκη ενός συνδέσμου στην οντότητα `<head>` του HTML, που θα δείχνει σε πια τοποθεσία είναι το CSS καθώς και το όνομά του. Ένα παράδειγμα εισαγωγής αρχείου CSS ακολουθεί στη συνέχεια:

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="styles.css">
  </head>
  <body>
    .....
    .....
  </body>
</html>
```

Παράδειγμα 3.3: αναφορά CSS σε HTML

Στο παράδειγμα αυτό το αρχείο CSS βρίσκεται στον ίδιο φάκελο με το αρχείο HTML, γι' αυτό δεν χρειάστηκε να γραφεί ολόκληρο το μονοπάτι στο οποίο βρίσκεται.

Οποιοδήποτε αρχείο CSS μπορεί να γραφτεί σε έναν απλό επεξεργαστή κειμένου αρκεί να αποθηκευτεί με την επέκταση `".css"`. Η μορφή ενός CSS είναι ως εξής:

```
body {
  background-color: powderblue;
}
h1 {
  color: blue;
}
p {
  color: red;
}
```

Παράδειγμα 3.4: μορφή ενός CSS

Μία ιστοσελίδα γραμμένη σε HTML χωρίς κάποια μορφοποίηση από ένα CSS θα εμφανιζόταν ως μία απλή σελίδα χαρτιού με κάποιο κείμενο γραμμένη σε αυτήν. Χάρη των ικανοτήτων που προσφέρει το CSS, μπορούμε να διαλέξουμε τη γραμματοσειρά του κειμένου, το μέγεθός του, το χώρο που θα καταλαμβάνουν στην οθόνη οι οντότητες, το χρώμα που θα εμφανίζονται στο HTML έγγραφο, η απόσταση που θα έχουν μεταξύ τους στην ιστοσελίδα, να καθορίσουμε διαφορετικές εμφανίσεις του ίδιου ιστότοπου σε διαφορετικές συσκευές, και πολλές άλλες δυνατότητες.

3.2.7 Έντυπα (*forms*) και λίστες (*lists*)

Δύο τελευταίες ικανότητες της HTML που αξίζει να αναφερθεί μιας και χρησιμοποιούνται σε μεγάλο βαθμό στη διπλωματική εργασία είναι η δημιουργία εντύπων συμπλήρωσης και η κατασκευή αναπτυσσόμενων καταλόγων (*drop-down lists*).

Μια φόρμα είναι ένα πεδίο μιας ιστοσελίδας που έχει στοιχεία ελέγχου φόρμας, όπως πεδία κειμένου, κουμπιά, πλαίσια ελέγχου, στοιχεία ελέγχου εμβέλειας ή επιλογείς χρωμάτων. Ένας χρήστης μπορεί να αλληλοεπιδράσει με μια τέτοια φόρμα, παρέχοντας δεδομένα τα οποία στη συνέχεια μπορούν να σταλούν στον διακομιστή για περαιτέρω επεξεργασία (π.χ. επιστρέφοντας τα αποτελέσματα μιας αναζήτησης ή υπολογισμού). Σε πολλές περιπτώσεις δεν απαιτείται κάποιο πρόγραμμα από την πλευρά του πελάτη, αν και είναι διαθέσιμη μία διεπαφή προγραμματισμού εφαρμογών (API) ώστε τέτοια προγράμματα να αυξήσουν την εμπειρία του χρήστη ή να χρησιμοποιήσουν φόρμες για σκοπούς άλλους από την υποβολή δεδομένων σε ένα διακομιστή. Η σύνταξη μιας φόρμας αποτελείται από διάφορα βήματα, τα οποία μπορούν να εκτελεστούν με οποιαδήποτε σειρά: γράφοντας το περιβάλλον χρήστη, υλοποιώντας την επεξεργασία από την πλευρά του διακομιστή και διαμορφώνοντας το περιβάλλον χρήστη για την πραγματοποίηση της επικοινωνίας με τον διακομιστή.

Οποιαδήποτε έντυπο συμπλήρωσης ξεκινά με την ετικέτα `<form>`, μέσα στην οποία τοποθετούνται τα στοιχεία ελέγχου. Τα περισσότερα στοιχεία ελέγχου βρίσκονται εντός της οντότητας `<input>`, που από προεπιλογή παρέχει ένα πεδίο κειμένου μιας γραμμής. Το πεδίο `<input>` μπορεί να εμφανιστεί με διάφορους τρόπους, ανάλογα με την τιμή που έχει το γνώρισμα (*attribute*) `type`. Αν για παράδειγμα το παραπάνω γνώρισμα είχε την τιμή `"submit"`, τότε θα είχαμε ένα κουμπί το οποίο όταν θα επιλεγόταν θα έκανε υποβολή της φόρμας ενώ αν είχε την τιμή `"radio"` θα εμφανιζόταν ένα πεδίο κυκλικής μορφής που θα επέτρεπε την επιλογή μίας ή πολλών αντικειμένων.

Ένα παράδειγμα εντύπου συμπλήρωσης ακολουθεί στη συνέχεια:

```
<!DOCTYPE html>
<html>
  <body>

    <form action="/action_page.php" method="get">
      <fieldset>
        <legend> λίστα συμπλήρωσης ονοματεπώνυμου </legend>
        First name:<br>
        <input type="text" name="firstname" >
        <br>
        Last name:<br>
        <input type="text" name="lastname" >
```

```

<br><br>
<input type="submit" value="Submit">
</fieldset>
</form>

```

<p> κατά την επιλογή του κουμπιού "submit" θα αποσταλθούν τα δεδομένα σε μία σελίδα που ονομάζεται "/action_page.php".

```

</body>
</html>

```

Παράδειγμα 3.5: φόρμα συμπλήρωσης

Το εμφανιζόμενο αποτέλεσμα στην οθόνη του θα είναι η παρακάτω εικόνα. Αμέσως παρατηρείται πως εξαιτίας της έλλειψης κάποιου αρχείου CSS, το έντυπο συμπλήρωσης εμφανίζεται ως μία λευκή σελίδα.

λίστα συμπλήρωσης ονοματεπώνυμου

First name:

Last name:

Submit

κατά την επιλογή του κουμπιού "submit" θα αποσταλθούν τα δεδομένα σε μία σελίδα που ονομάζεται "/action_page.php".

Εικόνα 3.1: μία απλή φόρμα συμπλήρωσης

Στον κώδικα που υλοποιεί την φόρμα, παρατηρείται πως υπάρχει ένα γνώρισμα με το όνομα *method*, και τιμή "get", για το οποίο δεν έχει γίνει κάποια αναφορά μέχρι τώρα καθώς και η ετικέτα *<fieldset>*. Το γνώρισμα *method* καθορίζει τη μέθοδο HTTP (*GET* ή *POST*) που πρέπει να χρησιμοποιείται κατά την υποβολή των δεδομένων φόρμας. Η προεπιλεγμένη μέθοδος κατά την υποβολή δεδομένων φόρμας είναι η *GET*. Ωστόσο, όταν χρησιμοποιείται η *GET* τα δεδομένα της φόρμας υποβολής θα είναι ορατά στο πεδίο διεύθυνσης σελίδας. Ορισμένα χαρακτηριστικά της μεθόδου είναι πως προσάπτει τα δεδομένα του εντύπου στη διεύθυνση URL σε ζεύγη ονόματος/τιμής και υστερεί σε θέματα ασφάλειας, για παράδειγμα αιτήσεις με την μέθοδο αυτή παραμένουν το ιστορικό του φυλλομετρητή. Η μέθοδος *POST* από την άλλη προσφέρει μεγαλύτερη ασφάλεια από την *GET*. Χρησιμοποιείτε πάντα εάν τα δεδομένα της φόρμας περιέχουν ευαίσθητες ή προσωπικές πληροφορίες. Η μέθοδος *POST* δεν εμφανίζει τα δεδομένα φόρμας που υποβλήθηκαν στο πεδίο διεύθυνσης σελίδας. Οι αιτήσεις αυτές δεν αποθηκεύονται ποτέ και επομένως δεν μένουν στο ιστορικό του φυλλομετρητή.

Η οντότητα *<fieldset>*, χρησιμοποιείται για την ομαδοποίηση μίας δέσμης οντοτήτων *<input>* μαζί, όπως στην περίπτωση αυτή τα στοιχεία εισόδου ονόματος και επωνύμου. Η

ετικέτα `<fieldset>` σχεδιάζει ένα πλαίσιο γύρω από τα τις οντότητες που περικλείει, με την ετικέτα `<legend>` να ορίζει μια λεζάντα για αυτήν.

Οι αναπτυσσόμενες λίστες είναι ένας άλλος τρόπος να δοθεί στον χρήστη η ευκαιρία να επιλέξει μόνο μία από μια σειρά επιλογών. Η αναπτυσσόμενη λίστα πρέπει να χρησιμοποιείται μόνο όταν ο χρήστης έχει να επιλέξει μεταξύ των επιλογών, καθώς δεν δίνει στον χρήστη τη δυνατότητα να μην επιλέξει τίποτα (όπως τα κουμπιά ραδιοφώνου). Οι αναπτυσσόμενες λίστες ορίζονται από την ετικέτα `<select>` και οι τιμές που αποστέλλονται καθορίζονται από το γνώρισμα `value`.

```
<!DOCTYPE html>
<html>
<body>

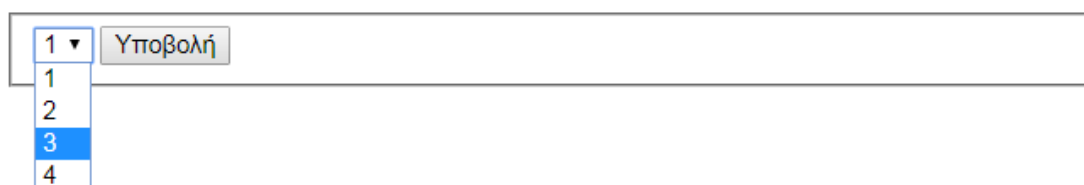
<form action="/action_page.php">
  <select name="numbers">
    <option value="1">1</option>
    <option value="2">2</option>
    <option value="3">3</option>
    <option value="4">4</option>
  </select>

  <input type="submit">
</form>

</body>
</html>
```

Παράδειγμα 3.6: μία drop-down λίστα

Για το απλό παράδειγμα *drop-down* λίστας, η αντίστοιχη αναπαράσταση του σε έναν φυλλομετρητή παρουσιάζεται παρακάτω.



Εικόνα 3.2: μία απλή αναπτυσσόμενη λίστα

Η οντότητα `<select>` αντιπροσωπεύει την δυνατότητα για την επιλογή μεταξύ ενός συνόλου επιλογών. Αν είναι παρών το γνώρισμα `multiple`, τότε η `<select>` οντότητα αντιπροσωπεύει τη δυνατότητα για την επιλογή μηδέν ή περισσότερων επιλογών από τη λίστα επιλογών. Αν το γνώρισμα απουσιάζει, τότε δίνεται η δυνατότητα επιλογής μόνο μίας τιμής από τη λίστα. Το σύνολο των επιλογών που δίνονται από μία αναπτυσσόμενη λίστα αποτελείται από όλες τις οντότητες `<option>` που περιέχονται στη `<select>`. Η ετικέτα `<option>` μπορεί να χρησιμοποιηθεί χωρίς κάποια γνωρίσματα, αλλά συνήθως χρειάζεται το γνώρισμα `value`, το οποίο υποδεικνύει τι αποστέλλεται στον διακομιστή.

3.3 Η γλώσσα PHP

Η PHP είναι μια γλώσσα προγραμματισμού για διακομιστές σχεδιασμένη για ανάπτυξη εφαρμογών ιστού αλλά χρησιμοποιείται επίσης και ως γλώσσα προγραμματισμού γενικής χρήσης. Αρχικά το ακρωνύμιο *PHP* σήμαινε *Personal Home Page*, αλλά τώρα αντιπροσωπεύει το αναδρομικό ακρωνύμιο *PHP: Hypertext Preprocessor*. Ένας κώδικας PHP μπορεί να ενσωματωθεί σε κώδικα HTML ή μπορεί να χρησιμοποιηθεί σε συνδυασμό με διάφορα συστήματα πρότυπων ιστού, συστήματα διαχείρισης περιεχομένου ιστού και πλαίσια ιστού. Συνήθως η επεξεργασία ενός κώδικα γραμμένου σε *PHP* γίνεται από έναν *μεταγλωττιστή PHP*, που παρέχεται ως ξεχωριστό κομμάτι στον εξυπηρετητή ιστού ή ως κοινή πυλαία διεπαφή (*Common Gateway Interface-CGI*). Ο διακομιστής ιστού συνδυάζει τα αποτελέσματα του μεταγλωττισμένου και εκτελεσμένου κώδικα, τα οποία μπορεί να είναι οποιοσδήποτε τύπος δεδομένων, συμπεριλαμβανομένων και εικόνων, με την παραγόμενη ιστοσελίδα[15].

Η *PHP* παρέχει μία πληθώρα δυνατοτήτων, όπως:

- Μπορεί να δημιουργήσει σελίδα δυναμικού περιεχομένου.
- Μπορεί να δημιουργήσει, να ανοίξει, να διαβάσει, να γράψει, να διαγράψει και να κλείσει αρχεία στην πλευρά του διακομιστή.
- Μπορεί να συλλέξει δεδομένα φόρμας.
- Μπορεί να στείλει και να λάβει *cookies*.
- Μπορεί να προσθέσει, να διαγράψει, να τροποποιήσει δεδομένα σε μία βάση δεδομένων.
- Μπορεί να χρησιμοποιηθεί για τον έλεγχο πρόσβασης των χρηστών.
- Μπορεί να κρυπτογραφεί δεδομένα.

Ορισμένα από τους λόγους για τους οποίους επιλέχτηκε η *PHP* είναι η συμβατότητά της με σχεδόν όλους τους διακομιστές που χρησιμοποιούνται αυτή την στιγμή (*Apache*, *IIS*, κ.τ.λ.), η συμβατότητά της με μεγάλο αριθμό λειτουργικών και πως παρόλο που είναι εύκολη στην εκμάθηση λειτουργεί πολύ αποτελεσματικά στην πλευρά του διακομιστή. Η ανάπτυξη της *PHP* ξεκίνησε το 1995 από τον *Rasmus Lerdorf*, όταν έγραψε διάφορα CGI προγράμματα σε C, που τα χρησιμοποιούσε για να διατηρήσει την προσωπική του σελίδα. Στη συνέχεια τα επέκτεινε έτσι ώστε να μπορούν να συνεργάζονται με φόρμες ιστού και για να επικοινωνούν με βάσεις δεδομένων. Ονόμασε αυτή την εφαρμογή "*Personal Home Page/Forms Interpreter*" ή *PHP/FI*.

Οι αρχικές εκδόσεις της *PHP* δεν προοριζόταν να είναι μια νέα γλώσσα προγραμματισμού, η γλώσσα άρχισε να αποκτά μορφή σταδιακά. Μια ομάδα ανάπτυξης ξεκίνησε να διαμορφώνεται και μετά από μήνες εργασίας και δοκιμές κυκλοφόρησε επισήμως το Νοέμβριο του 1997 η *PHP/FI 2*. Αυτή ακριβώς η ιδιομορφία της *PHP*, το ότι δεν προοριζόταν να είναι μια νέα γλώσσα προγραμματισμού, έχει οδηγήσει σε ασυνεπή ονομασία συναρτήσεων και ασυνεπή ταξινόμηση των παραμέτρων τους. Σε ορισμένες περιπτώσεις, τα ονόματα των λειτουργιών επιλέχθηκαν για να ταιριάζουν με τις βιβλιοθήκες χαμηλότερου επιπέδου που περιλαμβάνει η *PHP* ενώ σε μερικές πολύ πρώτες εκδόσεις της *PHP* το μήκος των ονομάτων λειτουργιών χρησιμοποιήθηκε εσωτερικά ως συνάρτηση *hash*, με αποτέλεσμα τα ονόματα να επιλέγονται με σκοπό τη βελτίωση των κατανομών των τιμών *hash*. Η πιο πρόσφατη έκδοση της *PHP* είναι η *PHP 7*, που εκδόθηκε το 2015.

3.3.1 Σύνταξη της γλώσσας

Ένα πρόγραμμα PHP μπορεί να τοποθετηθεί σε οποιοδήποτε σημείο ενός *HTML* εγγράφου, αρκεί να περικλείεται μεταξύ των αναγνωριστικών εκκίνησης και τερματισμού του *PHP* κώδικα “<?” και “?>”. Η σύνταξη της είναι πολύ κοντά στις εξής γλώσσες : *C*, *C++*, *Java*, *JavaScript* και *Perl*. Ο κώδικας αποτελείται από μια σειρά εντολών καθεμία από τις οποίες είναι μια οδηγία (*instruction*) που πρέπει να ακολουθήσει ο διακομιστής δικτύου πριν προχωρήσει στην επόμενη. Πάντα κάθε εντολή τερματίζεται με τη χρήση του χαρακτήρα *semicolon* (;).

Οι μεταβλητές πάντα τάσσονται μετά από το πρόθεμα “\$” χωρίς ωστόσο να υπάρχει ανάγκη να χαρακτηριστεί ο τύπος των δεδομένων που θα ανατεθούν στη μεταβλητή. Αν και η *PHP* δεν είναι *case sensitive*, τα ονόματα των μεταβλητών είναι.

Στην *PHP* υπάρχουν δύο βασικοί τρόποι για την εκτύπωση μίας εξόδου, οι συναρτήσεις *echo()* και *print()*. Οι διαφορές τους είναι ελάχιστες, με την πρώτη να μην επιστρέφει κάποια τιμή ενώ η *print()* επιστρέφει 1, με αποτέλεσμα να μπορεί να χρησιμοποιηθεί σε εκφράσεις. Επίσης η *echo()* μπορεί να πάρει πολλές παραμέτρους, παρόλο που τέτοια χρήση είναι σπάνια ενώ η *print()* μόνο ένα και τέλος η *echo()* είναι οριακά ταχύτερη από την *print()*. Ακολουθεί παρακάτω ένα παράδειγμα κώδικα σε *PHP*:

```
<!DOCTYPE html>
<html>
<body>
<?php
echo "Hello World!";
?>
</body>
</html>
```

Παράδειγμα 3.7: ένα *PHP* script

Το παραπάνω κομμάτι κώδικα θα εκτυπώσει στην οθόνη του χρήστη το μήνυμα *Hello World!*. Όπως και οι περισσότερες γλώσσες προγραμματισμού, έτσι και η *PHP* προσφέρει δυνατότητες κατασκευής δηλώσεων υπό συνθήκη (*Conditional Statements*), βρόχων επανάληψης και πινάκων. Οι δηλώσεις υπό συνθήκη έχουν την παρακάτω σύνταξη:

```
if (συνθήκη) {
    εκτελέσιμος κώδικας;
} elseif (συνθήκη) {
    εκτελέσιμος κώδικας;
} else{
    εκτελέσιμος κώδικας;
}
```

Παράδειγμα 3.8: *if* statement

Οι βρόχοι επανάληψης είναι δύο ειδών, οι *while{} βρόχοι* και οι *for (){} βρόχοι*. Η σύνταξή τους είναι η εξής, αντίστοιχα για τον καθένα βρόχο:

```
while (η συνθήκη ισχύει) {
    εκτελέσιμος κώδικας;
}
```

```
for (αρχική τιμή μετρητή; Τελική τιμή μετρητή; αύξηση τιμή μετρητή) {
    εκτελέσιμος κώδικας;
}
```

Παράδειγμα 3.8: while & for loops

Στην *PHP* οι πίνακες είναι τριών ειδών, πίνακες ευρετηρίου, συνειρμικοί πίνακες και πολυδιάστατοι πίνακες. Οι πίνακες ευρετηρίου χρησιμοποιούν έναν αριθμό για την κατάταξη κάποιου στοιχείου σε αυτόν ενώ οι συνειρμικοί χρησιμοποιούν κάποια λέξη-κλειδί για την συσχέτιση του στοιχείου του πίνακα. Τέλος, οι πολυδιάστατοι πίνακες είναι πίνακες οι οποίο περιέχουν πάνω από ένα πίνακα. Η κατασκευή ενός πίνακα γίνεται με την κλήση της συνάρτησης *array()*.

3.3.2 Συναρτήσεις και Κλάσεις

Όπως και οι περισσότερες γλώσσες προγραμματισμού έτσι και η *PHP* προσφέρει την δυνατότητα κατασκευής συναρτήσεων και κλάσεων για την διευκόλυνση των προγραμματιστών *PHP*. Εξαρχής, η γλώσσα προσφέρει μια μεγάλη γκάμα συναρτήσεων και πολλές είναι διαθέσιμες και σε διάφορες επεκτάσεις. Αυτές οι συναρτήσεις είναι καταγεγραμμένες στην επίσημη ιστοσελίδα της *PHP*. Η σύνταξη μίας συνάρτησης παρουσιάζεται παρακάτω.

```
<?php
function myFunction() {
    .....
    .....
    ....
}

myFunction (); // call the function
?>
```

Παράδειγμα 3.9: μορφή μίας συνάρτησης

Οι συναρτήσεις στην *PHP* έχουν τις ίδιες δυνατότητες με της συναρτήσεις άλλων γλωσσών προγραμματισμού. Μπορούν να δεχτούν πληροφορίες μέσω των *arguments*, τα οποία δηλώνονται εντός των παρενθέσεως που ακολουθεί το όνομα της συνάρτησης και να επιστρέψουν αποτελέσματα, με την χρήση της δήλωσης *return*. Αντί για δείκτες, οι συναρτήσεις στην *PHP* μπορούν να αναφέρονται από μια σειρά που περιέχει το όνομά τους. Με αυτό τον τρόπο, μπορούν να χρησιμοποιηθούν κανονικές συναρτήσεις *PHP*, για παράδειγμα, ως *callbacks* ή μέσα σε πίνακες συναρτήσεων. Οι συναρτήσεις που ορίζονται από τον χρήστη μπορούν να δημιουργηθούν ανά πάσα στιγμή. Μπορούν να οριστούν μέσα σε τμήματα κώδικα, επιτρέποντας τη λήψη απόφασης κατά την εκτέλεση του κώδικα ως το

αν πρέπει να οριστεί κάποια συνάρτηση. Υπάρχει μια συνάρτηση, η *function_exists*, που βλέπει αν έχει ήδη καθοριστεί μια συνάρτηση. Οι κλήσεις των συναρτήσεων από το κύριο πρόγραμμα πρέπει να χρησιμοποιούν παρενθέσεις, με εξαίρεση τις λειτουργίες κατασκευαστών τάξεων μηδενικής εισόδου που καλούνται με την χρήση του τελεστή *new*, οπότε οι παρενθέσεις είναι προαιρετικές.

Βασικές λειτουργίες αντικειμενοστραφή προγραμματισμού προστέθηκαν στην *PHP 3* και βελτιώθηκε στην *PHP 4*. Αυτό κατέστησε την *PHP* πιο αφαιρετική γλώσσα, με αποτέλεσμα να είναι ευκολότερη η εκτέλεση εργασιών για τους προγραμματιστές που χρησιμοποιούν τη γλώσσα. Ο χειρισμός αντικειμένων ξαναγράφηκε πλήρως για την *PHP 5*, επεκτείνοντας το σύνολο χαρακτηριστικών και βελτιώνοντας την απόδοση. Η *PHP 5* εισήγαγε ιδιωτικές και προστατευμένες μεταβλητές και μεθόδους, μαζί με αφηρημένες κλάσεις, τελικές κλάσεις, αφηρημένες μεθόδους και τελικές μεθόδους. Εισήγαγε επίσης ένα συγκεκριμένο τρόπο δήλωσης κατασκευαστών και καταστροφών, παρόμοιο με αυτό άλλων αντικειμενοστραφών γλωσσών, όπως στη *C++*, και ένα πρότυπο μοντέλο χειρισμού εξαιρέσεων. Επιπλέον, η *PHP 5* πρόσθεσε διεπαφές και επέτρεψε την εφαρμογή πολλαπλών διεπαφών. Υπάρχουν ειδικές διεπαφές που επιτρέπουν στα αντικείμενα να αλληλοεπιδρούν με το σύστημα χρόνου εκτέλεσης. Για παράδειγμα, με την κλάση *ArrayAccess* δίνεται η επιλογή απεικόνισης των κλάσεων με τη μορφή πινάκων.

```
<?php
class Books {
    /* Member variables */
    var $price;
    var $title;

    /* Member functions */
    function setPrice($par){
        $this->price = $par;
    }

    function getPrice(){
        echo $this->price . "<br/>";
    }

    function setTitle($par){
        $this->title = $par;
    }

    function getTitle(){
        echo $this->title . " <br/>";
    }
}
?>
```

Παράδειγμα 3.10: μία κλάση σε *PHP*

Η ορατότητα των ιδιοτήτων και των μεθόδων της *PHP* καθορίζεται χρησιμοποιώντας τις λέξεις-κλειδιά *public*, *private* και *protected*, οι οποίες καθορίζουν δημόσια, ιδιωτική ή προστατευμένη προβολή αντίστοιχα. Τα αντικείμενα που δηλώνονται δημόσια μπορούν να προσπελαθούν από παντού, τα ιδιωτικά μόνο από την κλάση στην οποία βρίσκονται και τα προστατευμένα μπορούν να προσπελαθούν από την κλάση που τα δημιουργεί και τις κλάσεις που την κληρονομούν. Η προεπιλογή είναι η δημόσια, αν χρησιμοποιείται μόνο το *var*. Η μεταβλητή *\$this* είναι μια ειδική μεταβλητή και αναφέρεται στο ίδιο αντικείμενο, δηλαδή στον εαυτό του.

Μετά από τον ορισμό της κλάσης ο προγραμματιστής μπορεί να ορίσει όσα αντικείμενα από αυτόν τον τύπο κλάσης θέλει. Στο παρακάτω τμήμα κώδικα γίνεται η δημιουργία τριών αντικειμένων, χρησιμοποιώντας την κλάση του παραδείγματος,

```
$physics = new Books;  
$maths = new Books;  
$chemistry = new Books;
```

και στη συνέχεια υπάρχει η δυνατότητα κλήσης συναρτήσεων που σχετίζονται με την κλάση.

```
$physics->setTitle( "Physics for High School" );  
$chemistry->setTitle( "Advanced Chemistry" );  
$maths->setTitle( "Algebra" );  
  
$physics->setPrice( 10 );  
$chemistry->setPrice( 15 );  
$maths->setPrice( 7 );
```

Κεφάλαιο 4ο: Οργάνωση και ταξινόμηση των καταχωρίσεων

4.1 Εισαγωγή

Η βάση δεδομένων που αναπτύχθηκε, έχει ως σκοπό την αποθήκευση τρισδιάστατων αντικειμένων και πιο συγκεκριμένα αρχαίων αγγείων, των οποίων τα μεταδεδομένα αποθηκεύονται σε μορφή XML σε μία υλοποίηση της βάσης δεδομένων *eXist-db*. Οι πληροφορίες για τα αγγεία που φιλοξενούνται καθώς και οι τρισδιάστατες απεικονίσεις τους προήλθαν από το αποθετήριο του αρχαιολογικού μουσείου των Χανίων, στο οποίο η συλλογή αγγείων και σκευών έχει συνολικά 147 καταχωρίσεις αντικειμένων. Ως διακομιστής χρησιμοποιείται η έκδοση του 2.4.18 του *apache server* ο οποίος εξυπηρετεί την ιστοσελίδα που επιτρέπει την δυνατότητα προβολής των τρισδιάστατων αγγείων, δημιουργίας λογαριασμού νέου χρήστη με τον δικό του αποθηκευτικό χώρο ώστε να ανεβάσει τα οποιαδήποτε τρισδιάστατα αντικείμενα που επιθυμεί καθώς και εξαγωγής των αρχείων που αναπαριστούν την τρισδιάστατη υλοποίηση των αντικειμένων και μεταπληροφοριών για κάθε αντικείμενο, με σκοπό να χρησιμοποιηθούν από το εργαλείο αξιολόγησης απόδοσης μεθόδων ανάκτησης πληροφοριών *Retrieval*. Για την κατασκευή του αποθετηρίου χρησιμοποιήθηκαν επίσης οι τεχνολογίες *PHP*, *HTML*, *eXist-db* που παρουσιάστηκαν προηγουμένως και ορισμένες συναρτήσεις γραμμένες σε *Javascript* που αξιοποιούν το σύνολο τεχνικών ανάπτυξης ιστού AJAX(Asynchronous JavaScript And XML) που επιτρέπουν στην ιστοσελίδα να ενημερώνετε ασύγχρονα, ανταλλάσσοντας δεδομένα με τον διακομιστή.

4.2 Το εργαλείο Retrieval και η συμβολή του στην κατηγοριοποίηση των αποθηκευμένων αντικειμένων

Το *Retrieval*, είναι μία πλατφόρμα αξιολόγησης της απόδοσης αλγορίθμων ανάκτησης πληροφοριών που δουλεύει στο διαδίκτυο[16]. Η αξιολόγηση της απόδοσης ενός τέτοιου αλγορίθμου από την σκοπιά του χρήστη, εκτελείται χρησιμοποιώντας τις αποστάσεις ανομοιότητας μεταξύ αντικειμένων που ανήκουν σε ένα σύνολο δεδομένων, που παράγονται από έναν αλγόριθμο ανάκτησης πληροφορίας. Οι ερευνητές βασίζουν τον υπολογισμό των αποστάσεων ανομοιότητας σε μια συνάρτηση που ακολουθεί έναν αριθμό ιδιοτήτων και υποδεικνύει τον βαθμό ομοιότητας μεταξύ του στοιχείου υπό αξιολόγηση και των στοιχείων στο σύνολο δεδομένων. Το *Retrieval* δέχεται ως δεδομένα εισόδου τιμές ανομοιότητας στοιχείων, συνοδευόμενες από δείκτες ταξινόμησης στοιχείων, χωρίς να απαιτείται πρόσβαση στον αλγόριθμο της μεθόδου ανάκτησης υπό αξιολόγηση, έχοντας ως αποτέλεσμα το εργαλείο να απαιτεί πληροφορίες σχετικές με την ταξινόμηση των στοιχείων, οι οποίες θα είναι αντιπροσωπεύουν και τα ιδανικά αποτελέσματα μίας μεθόδου ανάκτησης πληροφοριών. Αυτό συνετέλεσε στην ανάγκη κατηγοριοποίησης των τρισδιάστατων αγγείων που βρίσκονται στη βάση ώστε να εξαχθούν ως μεταδεδομένα για την δημιουργία των αρχείων αληθείας(*groudtruth files*).

Εξαιτίας του είδους των τρισδιάστατων αντικειμένων που φιλοξενεί η βάση, η οποιαδήποτε κατηγοριοποίησή τους απαιτεί την συνεργασία με την επιστήμη της Αρχαιολογίας. Ένας από τους κυριότερους τρόπους ταξινόμησης των αρχαίων ελληνικών αγγείων είναι με βάση την χρήση τους, κάτι τέτοιο όμως δεν θα εξυπηρετούσε τον σκοπό κατασκευής του αποθετηρίου. Αν και η μορφολογία και τα τεχνικά χαρακτηριστικά των αγγείων σε ένα βαθμό δε και η διακόσμηση τους, καθορίζονται από το προβλεπόμενο περιεχόμενο τους και τις συνθήκες χρήσης και αντιστρόφως η απόδοση των αγγείων ως προς την εκτέλεση συγκεκριμένων

καθηκόντων, καθορίζεται από τα γνωρίσματα της κατασκευής[17], μία τέτοια ταξινόμηση είναι αρκετά γενικού χαρακτήρα και δεν θα αποτελούσε ιδιαίτερη πρόκληση σε μία μέθοδο ανάκτησης πληροφορίας. Ένας ακόμα τρόπος χαρακτηρισμού τους θα ήταν επίσης με βάση της τεχνικής εικονογράφησής τους, κάτι που απορρίφθηκε για τον ίδιο λόγο που δεν επιλέχθηκε και η ταξινόμηση με βάση τη χρήση.

Αντ' αυτού επιλέχθηκε ως τρόπος ταξινόμησης των αγγείων το όνομά τους και κατ' επέκταση το σχήμα των αγγείων. Αν και τα σχήματα των αγγείων είχαν ως πρωταρχικό σκοπό την εξυπηρέτηση των αναγκών των ιδιοκτητών τους, δημιουργήθηκαν εκατοντάδες διαφορετικές υποκατηγορίες αγγείων κατά το πέρασμα των αιώνων. Πολλές από αυτές τις ονομασίες των διαφορετικών κατηγοριών σώζονται σε γραπτά κείμενα σύγχρονα με την περίοδο παραγωγής τους, όπως για παράδειγμα στις κωμωδίες του Αριστοφάνη, ο οποίος μας παραδίδει τον μεγαλύτερο αριθμό ονομασιών αρχαίων σκευών και στις τραγωδίες των τριών μεγάλων τραγικών ποιητών. Στις κωμωδίες του Αριστοφάνη συγκεκριμένα, αναφέρονται συνολικά 126 ονομασίες σκευών που έχουν άμεση σχέση με το φαγητό[18].

Ως στόχος της ταξινόμησης, ήταν η δημιουργία ενός “θησαυρού”, όπως είναι για παράδειγμα το *Getty Research Institute* και το *European Heritage Network (HEREIN)*. Η χρήση ελεγχόμενου λεξιλογίου για την περιγραφή πολιτιστικών κληρονομιών προσφέρει μεγάλη βοήθεια στις διάφορες κοινότητες χρηστών που ασχολούνται με την ανάκτηση δεδομένων.

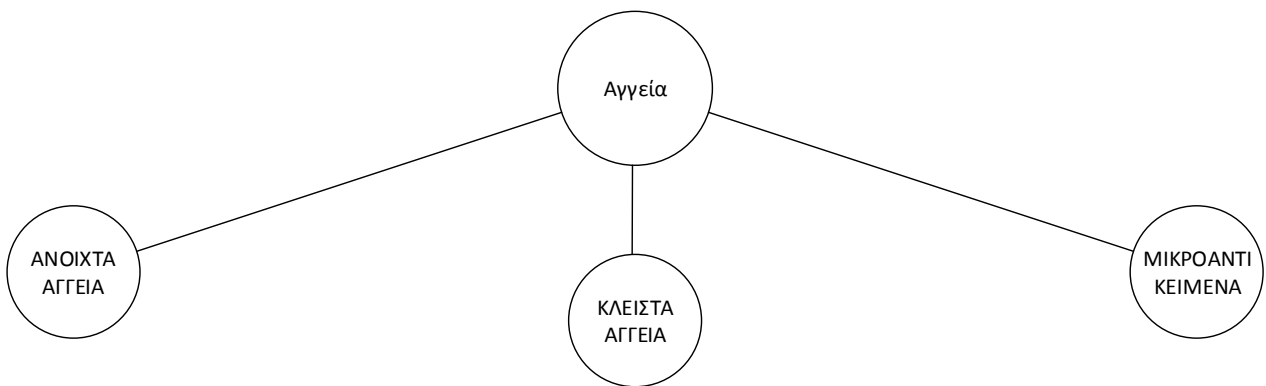
Σε συνεργασία με τον αρχαιολόγο *Ιωάννη Μούρθο*, προέκυψαν 146 κατηγορίες στις οποίες υπάρχει η δυνατότητα να ανήκει οπουδήποτε από τα φιλοξενούμενα αγγεία. Να σημειωθεί πως κάθε αγγείο μπορεί να ανήκει σε μία μόνο από αυτές τις κατηγορίες.

Σύμφωνα με την ταξινόμηση του αρχαιολόγου *Ι. Μούρθου* υπάρχουν τρεις βασικές κατηγορίες αγγείων, α) τα ανοιχτά αγγεία, β) τα κλειστά αγγεία και γ) τα μικροαντικείμενα. Οι τρεις βασικές αυτές κατηγορίες διακρίνονται σε ένα σύνολο υποκατηγοριών με βάση το σχήμα τους και εν συνεχεία αυτές οι υποκατηγορίες να έχουν ακόμα δύο επίπεδα ταξινόμησης με βάση τον τύπο τους. Οι κατηγορίες ως προς το σχήμα περιλαμβάνουν τα εξής σχήματα:

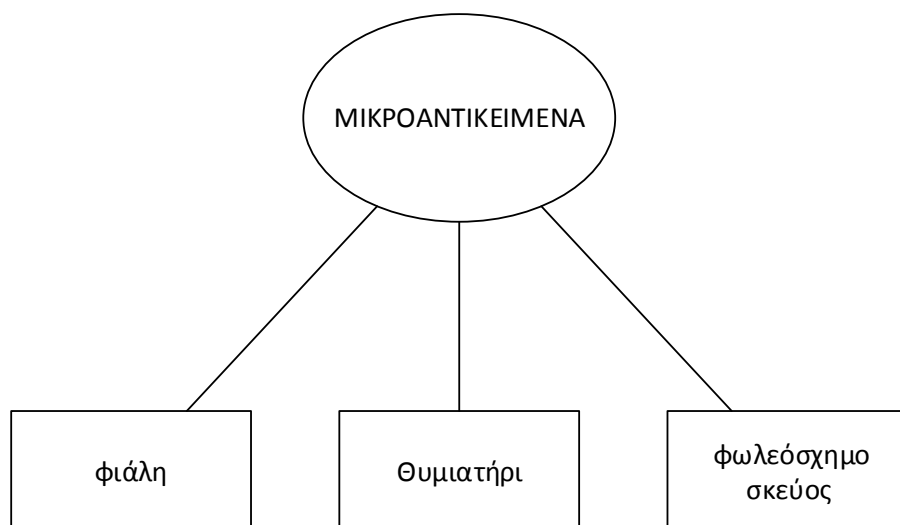
- γαβάθα
- εξάλειπτρο
- κάνθαρος
- κλεψύδρα
- κρατήρας
- κύαθος
- κύλικα
- λάκαινα
- λέβης/δίνος
- λουτήριο
- πινάκιο
- σκύφος
- στάμνος
- φιάλη
- αλάβαστρο
- αμφορέας
- αρύβαλλος
- ασκός
- θήλαστρο

- λήκυθος
- λύδιο
- οينوχόη
- πίθος
- πιθαμφορέας
- πυριατήρι
- πυξίδα
- ρυτό
- υδρία
- φορμίσκος
- ψυκτήρας
- λύχνος

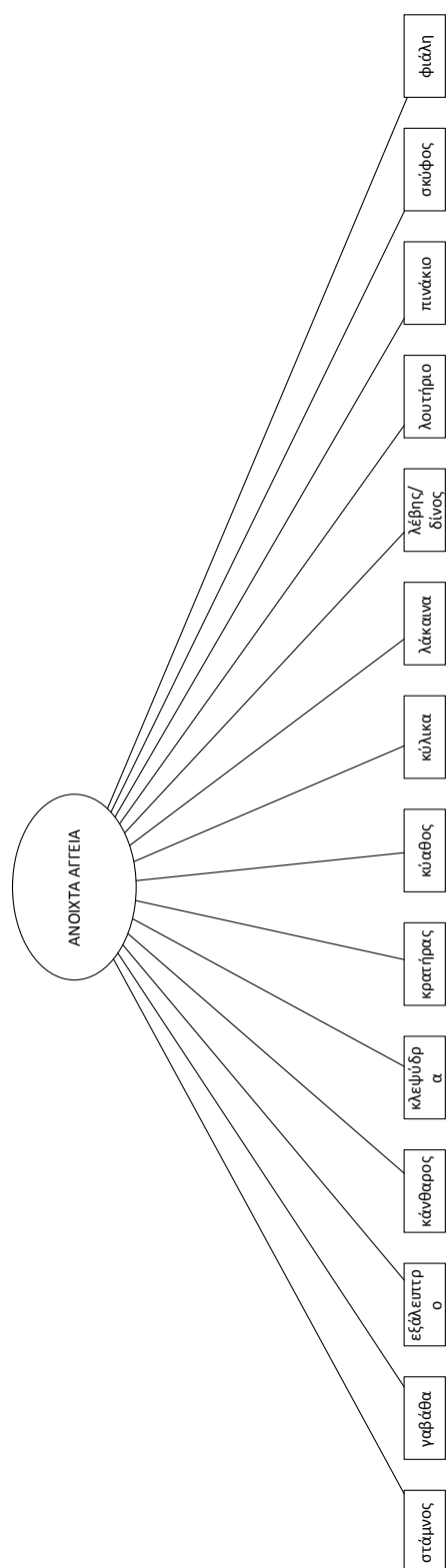
Στην συνέχεια ακολουθούν τα σχήματα που περιέχουν την συνολική έκταση της ταξινόμησης στο πρώτο και δεύτερο επίπεδο.



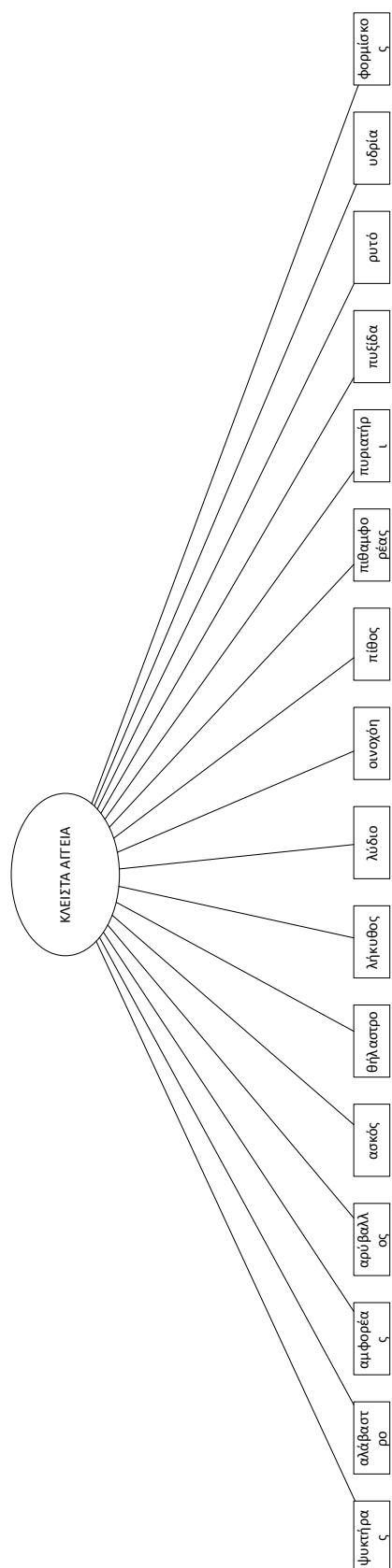
Σχήμα 4.1: πρώτο επίπεδο



Σχήμα 4.2: δεύτερο επίπεδο-μικροαντικείμενα



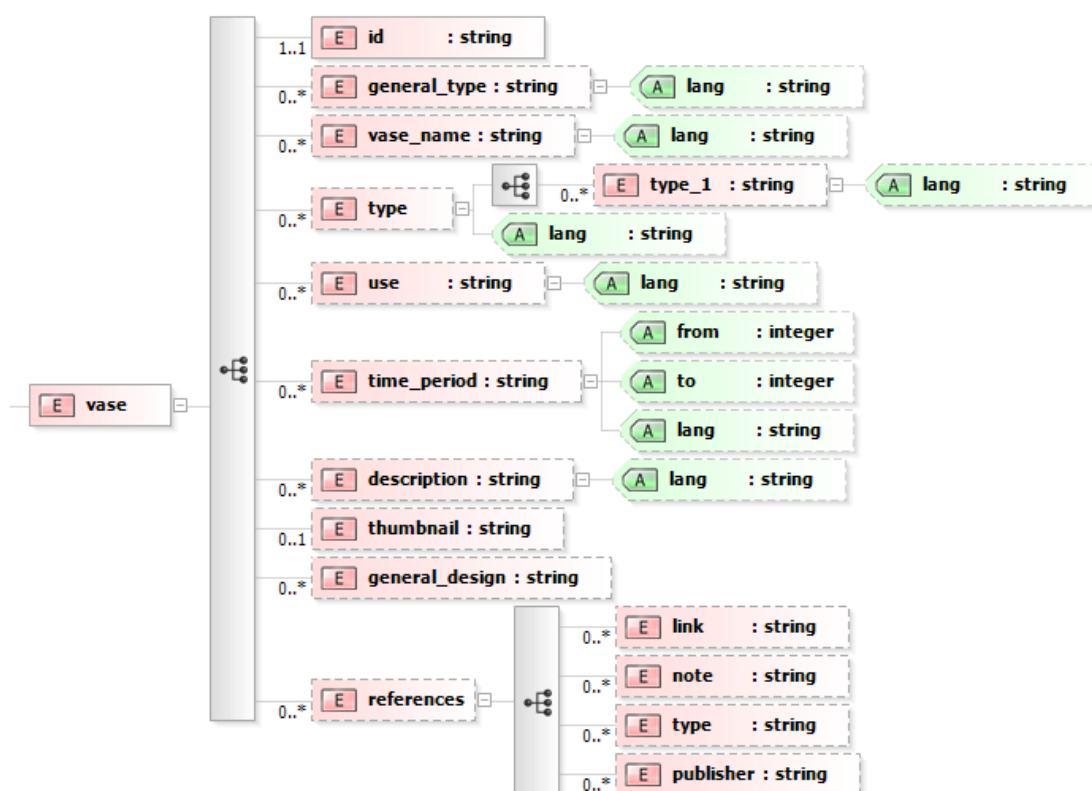
Σχήμα 4.3: δεύτερο επίπεδο-ανοιχτά αγγεία



Σχήμα 4.4: δεύτερο επίπεδο-κλειστά αγγεία

4.3 Απόδοση σε αρχεία XML των κατηγοριοποιήσεων

Για την αξιοποίηση της ταξινόμησης με βάση το σχήμα, δημιουργήθηκε ένα σύνολο αρχείων XML όπου στο καθένα υπάρχει από και από μία κατηγορία αγγείων. Το κάθε αρχείο έχει από ένα μοναδικό αναγνωριστικό(*id*), καθώς και τέσσερα ακόμα πεδία που αναφέρουν την χρήση του αγγείου αυτής της κατηγορίας, την χρονολόγησή του, μία παράγραφος στην οποία γίνεται μία μικρή περιγραφή και τέλος ένα πεδίο που περιέχει την τοποθεσία μίας εικόνας που εμφανίζεται ένα παράδειγμα αγγείου της κατηγορίας. Υπάρχουν συνολικά 146 τέτοια αρχεία με όλα συμμορφώνονται σύμφωνα με το αρχείο XSD που κατασκευάστηκε για τον έλεγχο εγκυρότητας των συγκεκριμένων XML, το αρχείο *dictionary.xsd*.



Σχήμα 4.5: διάγραμμα του αρχείου XSD *dictionary.xsd*

Η γονεϊκή οντότητα του κάθε XML αρχείου είναι το `<vase>`, που περιλαμβάνει την οντότητα *id*, το αναγνωριστικό της κατηγορίας αγγείου, στο *general_type* βρίσκετε η πληροφορία για την βασική κατηγορία(κλειστά αγγεία, ανοιχτά αγγεία, μικροαντικείμενα). Στο *vase_name* καταγράφεται το σχήμα και στις οντότητες *type* και *type_1* βρίσκεται αντίστοιχα το τρίτο και τέταρτο επίπεδο ταξινόμησης. Το πεδίο *description* υπάρχει περιγραφή αγγείου αυτής της κατηγορίας, στο *thumbnail* είναι το URL της φωτογραφίας και δίνεται επίσης η επιλογή εισαγωγής βιβλιογραφίας για την περιγραφή. Τέλος, εκτός από το *id* όλες οι οντότητες μπορούν να επαναληφθούν μέσα σε ένα XML και να καταγραφούν πληροφορίες σε διάφορες γλώσσες. Το γνώρισμα *lang* είναι υπεύθυνο για την καταγραφή της γλώσσας στην οποία είναι γραμμένες οι πληροφορίες εντός του πεδίου.

Η δημιουργία των XML έγινε με την χρήση του εργαλείου *matlab*, αφού πρώτα μέσω του *Microsoft excel* εξάχθηκαν οι πληροφορίες σε ένα αρχείο *txt* που διάβασε ο κώδικας *matlab* και στην συνέχεια αποθήκευσε την πληροφορία για κάθε κατηγορία σε ένα XML έγγραφο. Ο *matlab* κώδικας για αυτή την διεργασία βρίσκεται στο *dictionary_filler.m*.

```
<?xml version="1.0" encoding="utf-8"?>
<vase xmlns="https://www.w3schools.com"
  xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
  xs:schemaLocation="dictionary.xsd">
  <id>dic_7</id>
  <general_type lang="gr">ΑΝΟΙΧΤΑ ΑΓΓΕΙΑ</general_type>
  <general_type lang="en">OPEN VESSELS</general_type>
  <vase_name lang="gr">κάνθαρος</vase_name>
  <vase_name lang="en">kantharos</vase_name>
  <type lang="gr">γεωμετρικός<type_1 lang="gr">N/A</type_1>
  </type>
  <type lang="en">geometric<type_1 lang="en">N/A</type_1>
  </type>
  <use lang="gr">τελετουργικό αγγείο πόσεως</use>
  <use lang="en">ritual drinking vessel</use>
  <time_period lang="gr">N/A</time_period>
  <time_period lang="en">N/A</time_period>
  <description lang="gr">Κύπελλο με δύο ψηλές, καμπύλες, κάθετες λαβές. Κατά τη
  διάρκεια της γεωμετρικής περιόδου αντί για πόδι έχει ένα ψηλό
  στέλεχος.</description>
  <description lang="en">A footed cup with two high-curving vertical handles. During the
  Geometric period instead of a foot it has a tall stem.</description>
  <thumbnail>dic_screenshots/kantharos_geometrikos_bretaniko.jpg</thumbnail>
  <general_design>N/A</general_design>
  <references>
    <link/>
    <note/>
    <type/>
    <publisher/>
  </references>
</vase>
```

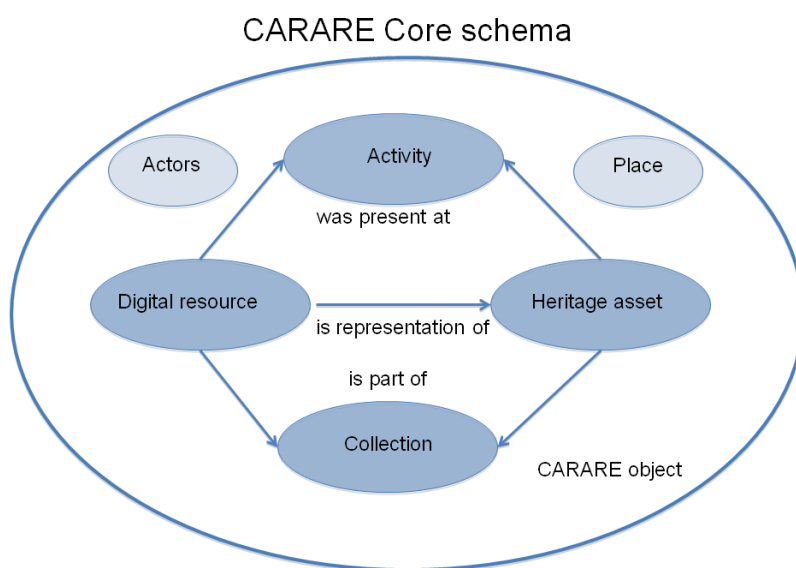
Παράδειγμα 4.1:κατηγορία αγγείου σε μορφή XML αρχείου. Είναι έγκυρο σύμφωνα με το σχήμα *dictionary.xsd*.

4.4 Αποθήκευση μεταδεδομένων των αγγείων

Εκτός από την ανάγκη καταγραφής της ταξινόμησης των αγγείων, ήταν απαραίτητη και η καταγραφή ενός συνόλου πληροφοριών που χαρακτήριζαν το κάθε ένα αγγείο της βάσης. Η βέλτιστη επιλογή για την οργάνωσή τους μέσα στα αρχεία *XML* κρίθηκε πως είναι η δεύτερη έκδοση του σχήματος μεταδεδομένων *CARARE*.

4.4.1 Σχήμα μεταδεδομένων *CARARE*

Το *CARARE* είναι ένα έργο που απευθύνεται στους χρήστες της ευρωπαϊκής ψηφιακής βιβλιοθήκης *Europeana*, με σκοπό να κάνει προσιτό ένα σύνολο μοναδικών αρχαιολογικών μνημείων, αρχιτεκτονικά σημαντικών κτιρίων, ιστορικών κέντρων των πόλεων, βιομηχανικών μνημείων και τοπίων. Ένας από τους κύριους στόχους του *CARARE* ήταν να εξασφαλίσει τη διαλειτουργικότητα μεταξύ των τοπικών μεταδεδομένων που διατηρούν οι διάφοροι τοπικοί οργανισμοί πολιτιστικών κληρονομιών και μεταδεδομένων που χρησιμοποιούνται από την *Europeana*. Για την ενσωμάτωση των ψηφιακών πόρων των διάφορων οργανισμών, ένα από τα σημαντικότερα καθήκοντα του *CARARE* ήταν να δημιουργήσει ένα σχήμα μεταδεδομένων ικανό να χαρτογραφήσει τα υπάρχοντα μεταδεδομένα σε ένα κοινό σχήμα[19]. Το έργο έχει δημιουργήσει ένα σχήμα μεταδεδομένων που χρησιμοποιείται για τη διαμεσολάβηση μεταξύ των αρχικών δεδομένων και των προτύπων και σχημάτων που ορίζονται από την *Europeana*. Καθώς όλες οι οντότητες του *CARARE* έπρεπε να συγκεντρωθούν και να παραδοθούν σε κοινή μορφή, το *CARARE* έχει χαρτογραφηθεί στο σχήμα το οποίο χρησιμοποιεί η *Europeana* για να περιγράψει το περιεχόμενό της, το *μοντέλο δεδομένων Europeana (EDM)*. Το ψηφιακό περιεχόμενο που υποβλήθηκε από το έργο έχει περιγραφεί χρησιμοποιώντας το *EDM 5.2*. Δεδομένου ότι το *EDM* είναι ένα εννοιολογικό μοντέλο, οι οντότητες *CARARE* έχουν χαρτογραφηθεί απευθείας σε κλάσεις του *EDM* ή σε συγκεκριμένες διαδρομές του[19].

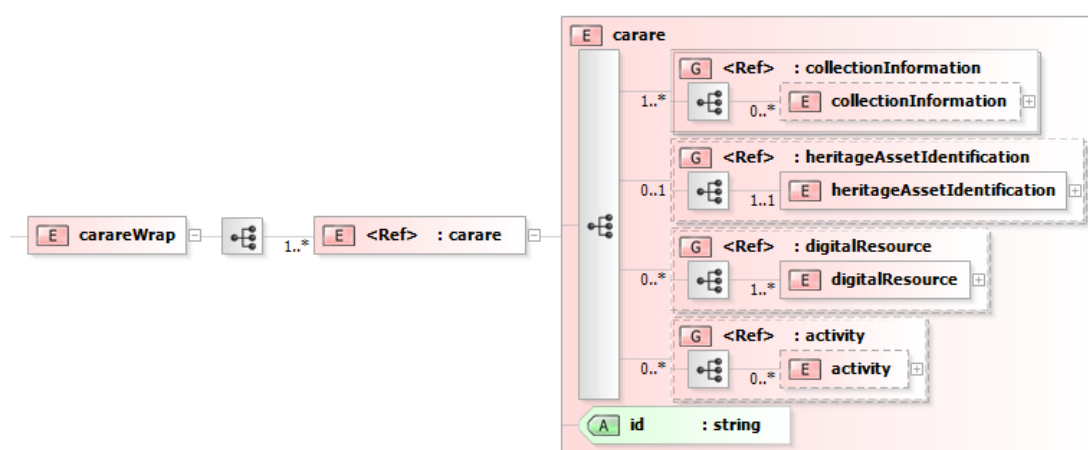


Σχήμα 4.6: βασικό σχήμα περιγραφής του σχήματος *CARARE*. Πάρθηκε από το έγγραφο D6.1 *Report on Metadata and Thesaurii*.

Όπως και στο μοντέλο δεδομένων *Europeana*, έτσι και στο σχήμα CARARE υπάρχουν οντότητες που επιτρέπουν το χαρακτηρισμό μεγάλου όγκου πληροφοριών. Τέτοιες οντότητες είναι για παράδειγμα η *contacts*, που μπορεί να περιλαμβάνει ένα πλήθος οντοτήτων. Επί της ουσίας, το CARARE σχήμα ακολουθεί το *οντοκεντρικό μοντέλο(object centric model)* προσέγγισης της πληροφορίας, δηλαδή επικεντρώνεται στο αντικείμενο που περιγράφεται. Οι πληροφορίες έρχονται με τη μορφή δηλώσεων που παρέχουν μια άμεση σύνδεση μεταξύ του περιγραφόμενου αντικειμένου και των χαρακτηριστικών του[19].

Η δεύτερη έκδοση του σχήματος η οποία υπαγορεύει την δομή που έχουν τα αρχεία XML της βάσης, έχει τις εξής κορυφαίες οντότητες[20]:

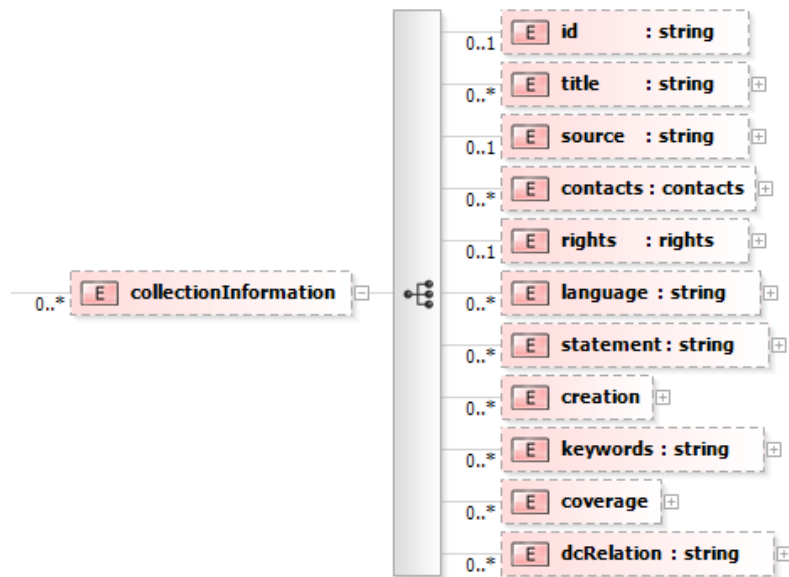
- *CARARE Wrap* – περικλείει το σχήμα CARARE. Μέσα σε αυτό μπορούν να περιέχονται παραπάνω από μία εγγραφή που ακολουθεί το πρότυπο του CARARE.
- *CARARE* – Η οντότητα έναρξης CARARE. Περιλαμβάνει τις οντότητες *Heritage Asset*, *Collection information*, *Digital Resource* and *Activity*:
 - *Heritage Asset* - κατέχει τα μεταδεδομένα για ένα μνημείο, κτίριο ή πολιτιστικό αντικείμενο συμπεριλαμβανομένων των τυπωμένων υλικών και ψηφιακών αντικειμένων, συμπεριλαμβανομένων περιγραφικών και μεταδεδομένα διαχείρισης.
 - *Digital resource* - κατέχει μεταδεδομένα σχετικά με έναν ψηφιακό πόρο, συμπεριλαμβανομένης της τοποθεσίας του στο διαδίκτυο.
 - *Activity* - περιέχει τα μεταδεδομένα για ένα συμβάν ή δραστηριότητα.
 - *Collection information* - περιέχει την περιγραφή της συλλογής.



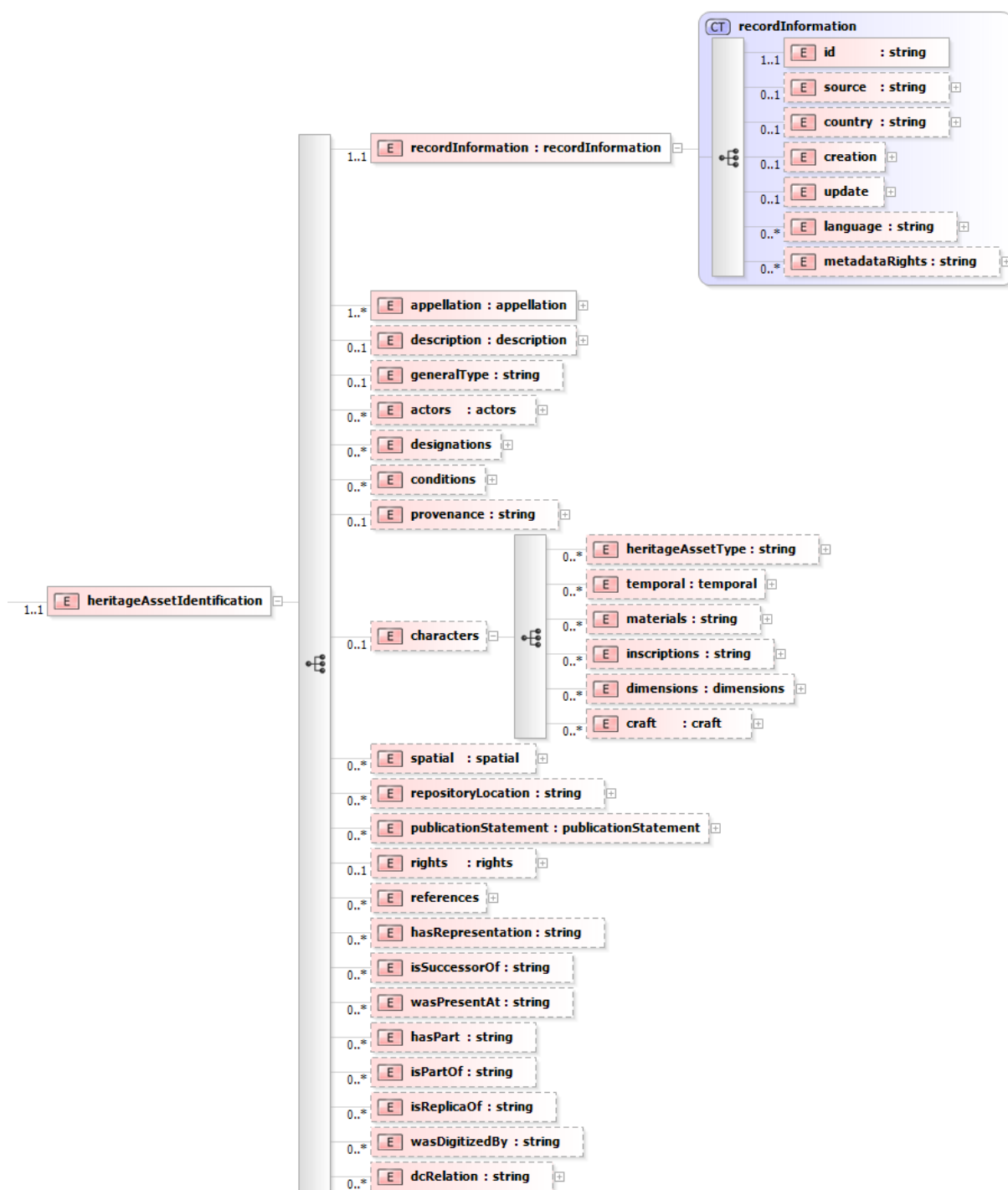
Σχήμα 4.7: οι κορυφαίες οντότητες του σχήματος CARARE

Στο πεδίο *Collection information* παρέχει μία περιγραφή σε επίπεδο συλλογής των πόρων πολιτιστικής κληρονομιάς που περιγράφονται στην οντότητα *Heritage Asset Identification* και χρησιμοποιείται για την περιγραφή του συνόλου των πληροφοριών. Μπορεί να χρησιμοποιηθεί για να περιγράψει αρχαιολογικά μνημεία, ιστορικά κτίρια, βιομηχανικά

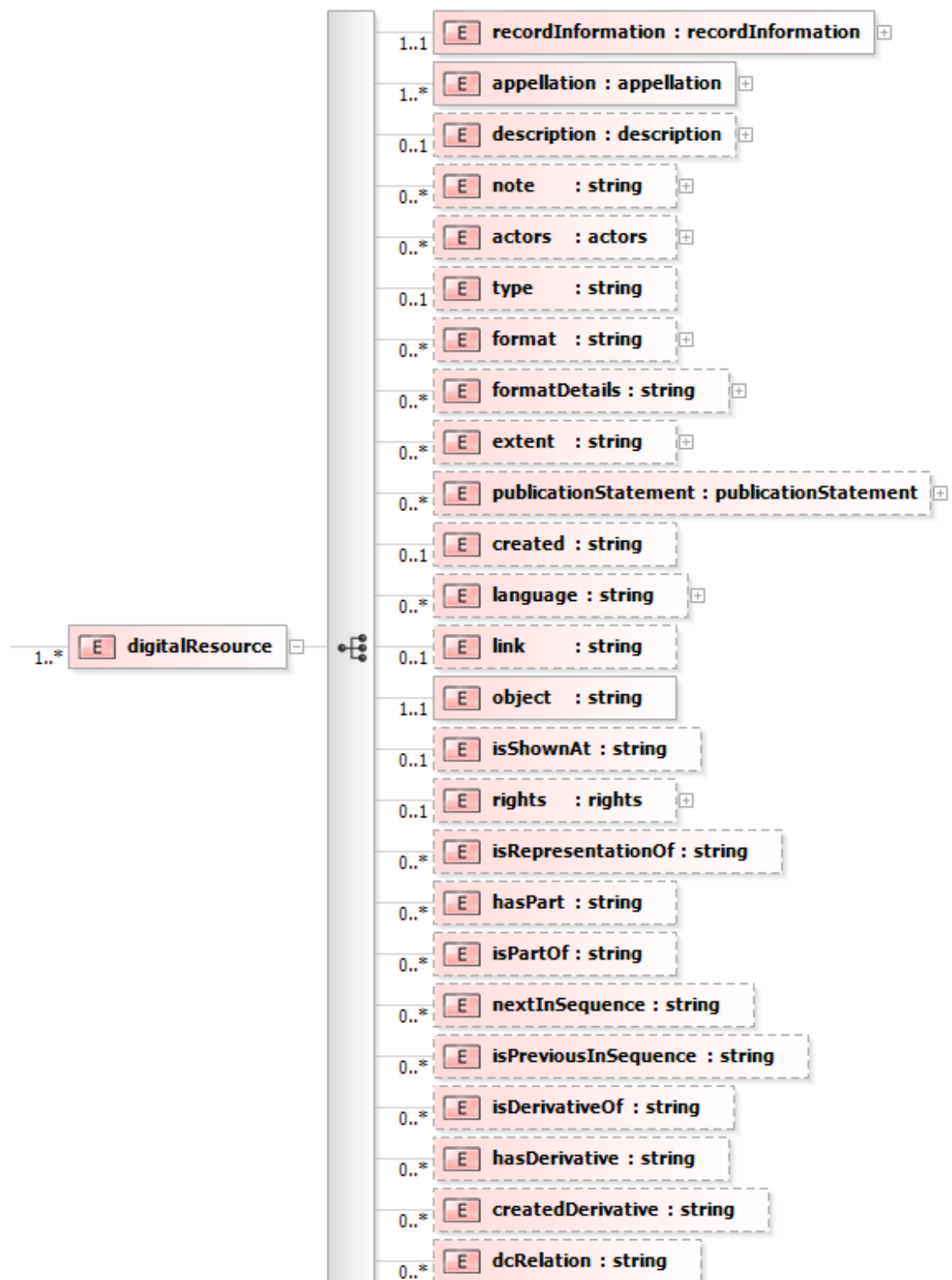
μνημεία, περιοχές αρχαιολογικού τοπίου, ναυάγια, τεχνουργήματα και οικολογικά σκεύη, καθώς και βιβλία, έντυπα, τυπωμένους χάρτες, σχέδια, φωτογραφίες, ταινίες, τρισδιάστατα ψηφιακά μοντέλα που σχετίζονται με την αρχαιολογική και αρχιτεκτονική κληρονομιά. Η δυνατότητα δημιουργίας σχέσεων μεταξύ αρχείων πολιτιστικής κληρονομιάς επιτρέπει την σύνδεση μνημείων που αποτελούν τμήματα ενός μεγαλύτερου συγκροτήματος, μεταξύ τους. Για παράδειγμα ο Παρθενώνας, Τα Προπύλαια και το Ερέχθειο είναι μέρος της Ακρόπολης της Αθήνας. Το *Digital Resource* περιλαμβάνει τους ψηφιακούς πόρους (εικόνα, κείμενο, βίντεο, ήχο, μοντέλο 3D) που προσφέρονται από τον οργανισμό προς τους χρήστες και τέλος η οντότητα *Activity* δίνει τη δυνατότητα καταγραφής πληροφοριών σχετικά με δραστηριότητες που συμμετείχε το μνημείο[20].



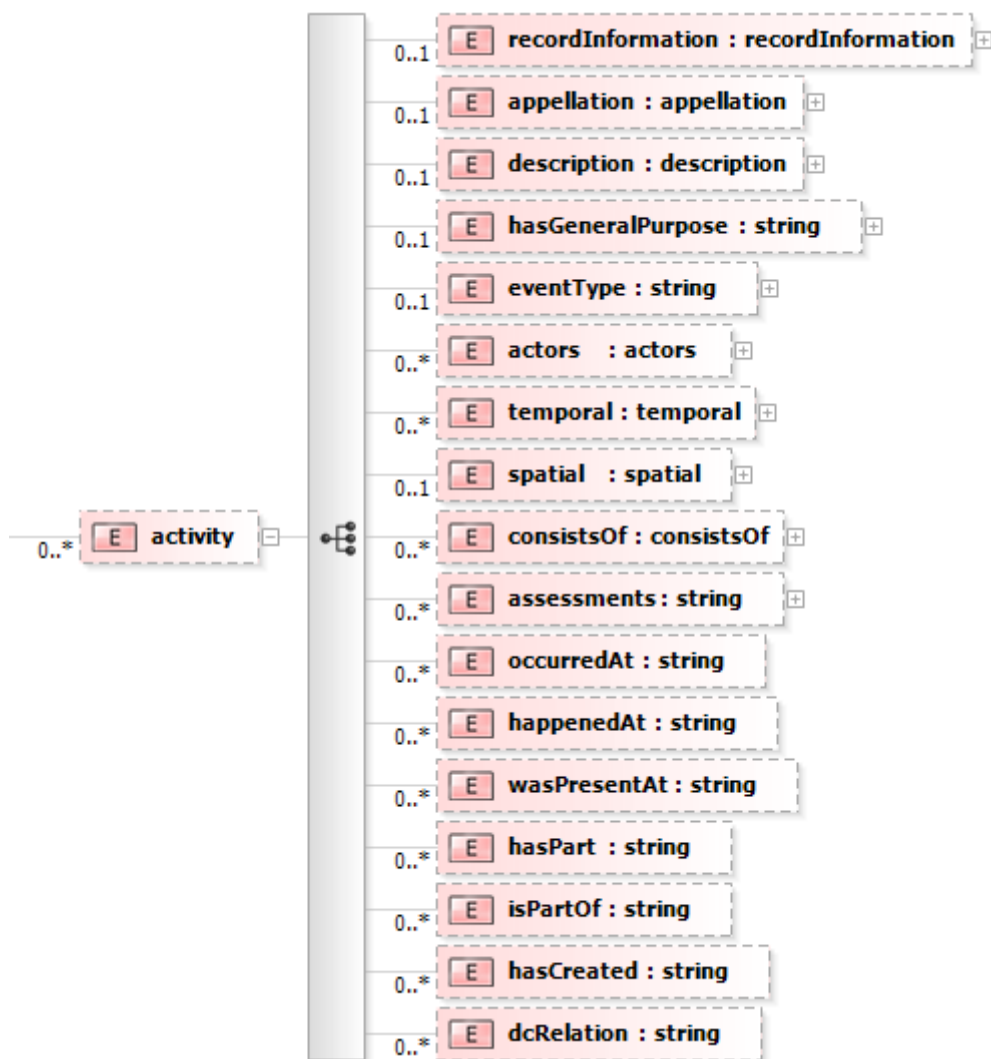
Σχήμα 4.8: Οι οντότητες που περιλαμβάνονται στο *collection Information*



Σχήμα 4.9: Οι οντότητες που περιλαμβάνονται στο *Heritage asset identification*



Σχήμα 4.10: Οι οντότητες που περιλαμβάνονται στο *digital resource*



Σχήμα 4.11: Οι οντότητες που περιλαμβάνονται στο activity

4.4.2 Αποθήκευση των μεταδεδομένων

Τα μεταδεδομένα των αγγείων στη βάση όπως προαναφέρθηκε είναι έγκυρα σύμφωνα με το σχήμα CARARE. Όμως δεν χρειάστηκε να γίνει χρήση όλων των οντοτήτων που περιεγράφηκαν παραπάνω. Μία από τις τέσσερις κύριες οντότητες του σχήματος, το activity, δεν χρειάστηκε σε κανένα από τα αγγεία επομένως δεν συμπληρώθηκε. Ακολουθεί στη συνέχεια μία αναφορά των οντοτήτων που χρησιμοποιήθηκαν καθώς και του είδους των πληροφοριών που περιέχουν.

Collection information:

- *Id*: μοναδικό αναγνωριστικό της συλλογής. Μιας και όλα τα αγγεία ανήκουν στην ίδια συλλογή, για κάθε ένα από αυτά είναι ίδιο.
- *Title*: ο τίτλος της συλλογής. Είναι γραμμένος στα ελληνικά και στα αγγλικά.
- *Source*: αναφέρετε η πηγή προέλευσης της συλλογής.

- *Rights*: δικαιώματα διαχείρισης της συλλογής.
- *Language*: η γλώσσα στην οποία είναι γραμμένα τα μεταδεδομένα. Η οντότητα υπάρχει δύο φορές.
- *Keywords*: λέξεις-κλειδιά που χαρακτηρίζουν τη συλλογή.
- *Coverage*: καταγράφει πληροφορίες για τη χρονική προέλευση των καταχωρίσεων της συλλογής καθώς και πληροφορίες σε σχέση με την τοποθεσία προέλευσής της.

Heritage Asset Identification:

- *RecordInformation*: περιλαμβάνει το μοναδικό χαρακτηριστικό του αγγείου, την πηγή προέλευσής του, τα δικαιώματα διαχείρισης των μεταδεδομένων και την χώρα προέλευσης.
- *Appellation*: το όνομα του αγγείου. Διατίθεται στα ελληνικά και στα αγγλικά.
- *Description*: περιγραφή των χαρακτηριστικών του. Διατίθεται στα ελληνικά και στα αγγλικά.
- *GeneralType*: καταγράφει το είδος των καταχωρίσεων, στην προκειμένη περίπτωση.
- *Provenance*
- *Characters*: τα κύρια χαρακτηριστικά του αγγείου. Περιλαμβάνει εκτός από πληροφορίες σχετικά με την χρονολόγηση, τις διαστάσεις και το υλικό κατασκευής του, καταγράφεται επίσης και η ταξινόμηση του με βάση το σχήμα του. Η πληροφορία βρίσκεται στην οντότητα *heritageAssetType*, που διαθέτει τα γνωρίσματα «*lang*», «*namespace*» και «*term*». Στο *namespace* είναι το μοναδικό αναγνωριστικό της ταξινόμησης(*id*), στο *term* το επίπεδο της ταξινόμησης που αναφέρεται στην οντότητα *heritageAssetType* και τέλος το *lang* καταγράφει την γλώσσα που είναι γραμμένη η πληροφορία. Το *heritageAssetType* επαναλαμβάνεται τρεις φορές, για κάθε ένα από τα επίπεδα της ταξινόμησης. Ο βασικός διαχωρισμός σε ανοιχτά αγγεία, κλειστά αγγεία και μικροαντικείμενα δεν θεωρήθηκε αναγκαίος για τον προσδιορισμό τους.
- *Spatial*: πληροφορίες για την τοποθεσία προέλευσης του αγγείου.
- *Rights*: δικαιώματα διαχείρισης του αγγείου.
- *References*: βιβλιογραφική αναφορά των πληροφοριών του πεδίου *Description*.

Digital Resource:

- *RecordInformation*: περιλαμβάνει τις ίδιες πληροφορίες με την οντότητα *Record Information* της *Heritage Asset Identification* οντότητας αλλά για τον χαρακτηρισμό της τρισδιάστατης απεικόνισής του.
- *Appellation*
- *Type*: το είδος του ψηφιακού αρχείου. Σε κάθε αγγείο είναι το ίδιο, δηλώνεται πως πρόκειται για τρισδιάστατο ψηφιακό αρχείο.
- *Object*: δηλώνεται η διαδρομή στην οποία βρίσκεται το thumbnail για το συγκεκριμένο αγγείο.
- *isShownAt*: δηλώνεται η διαδρομή του τρισδιάστατου αρχείου του αγγείου.

Η δημιουργία των XML έγινε με τον ίδιο τρόπο κατασκευής των αρχείων της ταξινόμησης. Πρώτα μέσω του *Microsoft excel* εξάχθηκαν οι πληροφορίες σε ένα αρχείο *txt* που διάβασε ο κώδικας *matlab* και στην συνέχεια αποθήκευσε την πληροφορία για κάθε αγγείο σε ένα

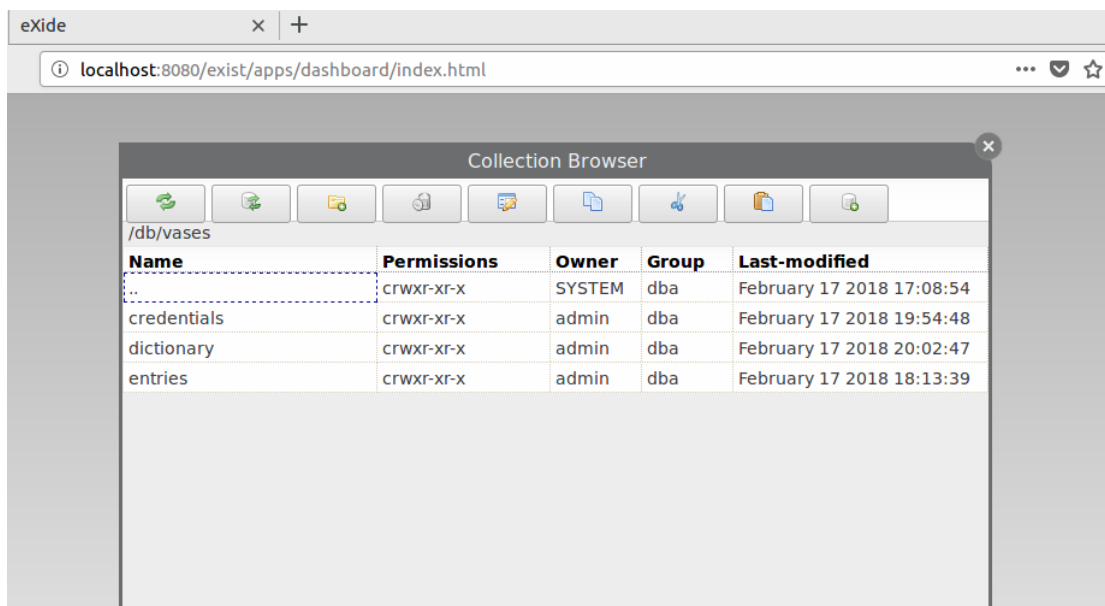
XML έγγραφο. Ο κώδικας που εκτελεί αυτήν τη διαδικασία βρίσκεται γραμμένος στο αρχείο `xml_filler_carrare.m`.

Κεφάλαιο 5ο: Υλοποίηση

5.1 Στήσιμο της βάσης δεδομένων

Αρχικά, στήθηκε ένα εικονικό μηχάνημα(*virtual machine*) στο οποίο εγκαταστάθηκε το λειτουργικό *Ubuntu server 16.4.2*. το μηχάνημα διαθέτει 2 gb μνήμη RAM και 30 gb αποθηκευτικό χώρο. Στη συνέχεια έγινε εγκατάσταση της έκδοσης 5.6 της PHP και του *Pear*, ενός πλαισίου και συστήματος διανομής συναρτήσεων και κλάσεων PHP[21]. Το *Pear* χρειάστηκε για την εγκατάσταση του πακέτου *XML_RPC2*, που επιτρέπει την επικοινωνία της βάσης *eXistdb* με τον κώδικα PHP, που αξιοποιεί το πρωτόκολλο *xml-rpc*[22].

Στη συνέχεια εγκαταστάθηκε η βάση δεδομένων *eXist-db*. Επιλέχθηκε η έκδοση 3.6.0. Μετά από την εγκατάσταση της βάσης δημιουργήθηκε η συλλογή *vases*, στην οποία βρίσκονται αποθηκευμένα τα XML αρχεία. Σε αυτή περιέχονται υποσυλλογές για κάθε χρήστη της βάσης, και το σύνολο των αρχείων XML που προέκυψαν από την διαδικασία ταξινόμησης και μία συλλογή με τα ονόματα των εγγεγραμμένων χρηστών.



Εικόνα 5.1: συλλογές στην βάση

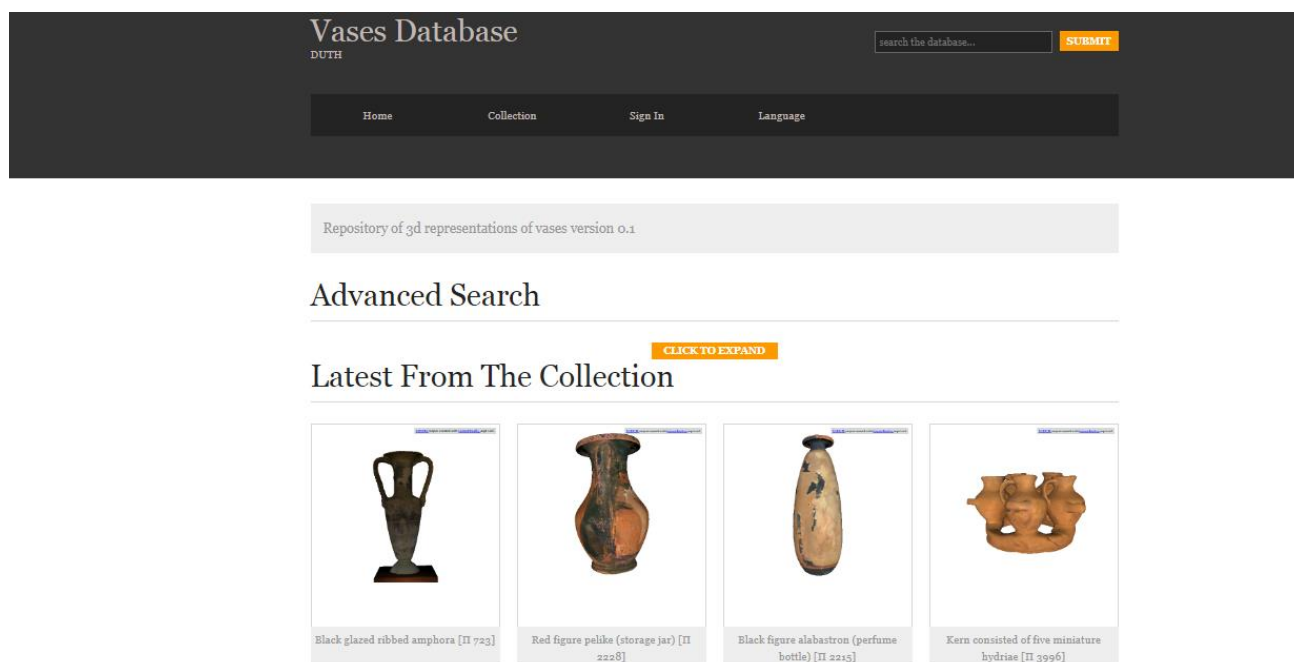
Αφού δημιουργήθηκε η κεντρική συλλογή, στη συνέχεια ανέβηκαν στην βάση τα αρχεία των δεδομένων κάθε αγγείου. Τέλος, ανέβηκαν τα αρχεία των τρισδιάστατων αντικειμένων στον φάκελο αρχείων του *apache server*.

5.2 Κατασκευή της ιστοσελίδας

Μετά την εγκατάσταση της βάσης, σειρά είχε η υλοποίηση του κώδικα της ιστοσελίδας που επιτρέπει την περιήγηση στη συλλογή των αγγείων. Για την δημιουργία όμως των διάφορων ερωταπαντήσεων χρειάστηκε η βιβλιοθήκη συναρτήσεων και κλάσεων PHP *php-eXist-db-Client*, που παρέχεται από το documentation της *eXist-db*.

5.2.1 Αρχική σελίδα

Η ιστοσελίδα βρίσκεται στην τοποθεσία <http://orion.ee.duth.gr>, με το php αρχείο *index.php* περιγράφει την αρχική σελίδα της. Ο html κώδικας που χρησιμοποιήθηκε ως βάση για την μορφοποίηση της σελίδας πάρθηκε από τον ιστότοπο <http://www.os-templates.com>, που περιέχει ένα πλήθος δωρεάν προτύπων ιστοσελίδων. Το περιεχόμενο της σελίδας παρέχεται μέχρι στιγμής σε δύο γλώσσες, ελληνικά και αγγλικά.



Εικόνα 5.2: αρχική σελίδα

Στην αρχική σελίδα παρουσιάζονται τέσσερα από τα πιο πρόσφατα αντικείμενα που έχουν καταχωρηθεί στη βάση καθώς και αναζήτηση αντικειμένων σύμφωνα με την ταξινόμηση βάσης σχήματος. Για την φόρτωση των πιο πρόσφατων αντικειμένων γίνεται ένα ερώτημα *query* στη βάση που βρίσκεται ενσωματωμένο στον κώδικα php της σελίδας ενώ για την αναζήτηση βάσης σχήματος στέλνεται ένα δεύτερο ερώτημα που ελέγχει το πλήθος των διαφορετικών κατηγοριοποιήσεων δευτέρου επιπέδου που υπάρχουν στα αγγεία της βάσης και επιστρέφει αυτό το πλήθος. Με το πάτημα του κουμπιού "click to expand", εμφανίζεται μία λίστα κατηγοριών από τις οποίες ο χρήστης μπορεί να διαλέξει παραπάνω από μία για να προβάλει στην οθόνη του. Για την κατασκευή του πεδίου της προχωρημένης αναζήτησης αξιοποιήθηκε η βιβλιοθήκη ανοιχτού λογισμικού *bootstrap*.

Προχωρημένη Αναζήτηση

CLICK TO EXPAND

☐ εξάλευτρο ☐ λύχνος ☐ πίθος ☐ οينوχόη ☐ πυξίδα ☐ ρυτό ☐ κάνθαρος ☐ θυμιατήρι ☐ σκύφος ☐ φιάλη ☐ ασκός ☐ κύλικα ☐ κρατήρας ☐ αμφορέας ☐ αλάβαστρο ☐ αρύβαλλος ☐ υδρία ☐ ατλό ☐ θήλαστρο ☐ πιθαμφορέας ☐ Κύλικα ☐ N/A ☐ φωλεόσχημο σκεύος ☐ "σκύφος" ☐ λουτήριο ☐ λέβης/δίνος ☐ Κέρνος

Submit

Πιό πρόσφατα από την συλλογή

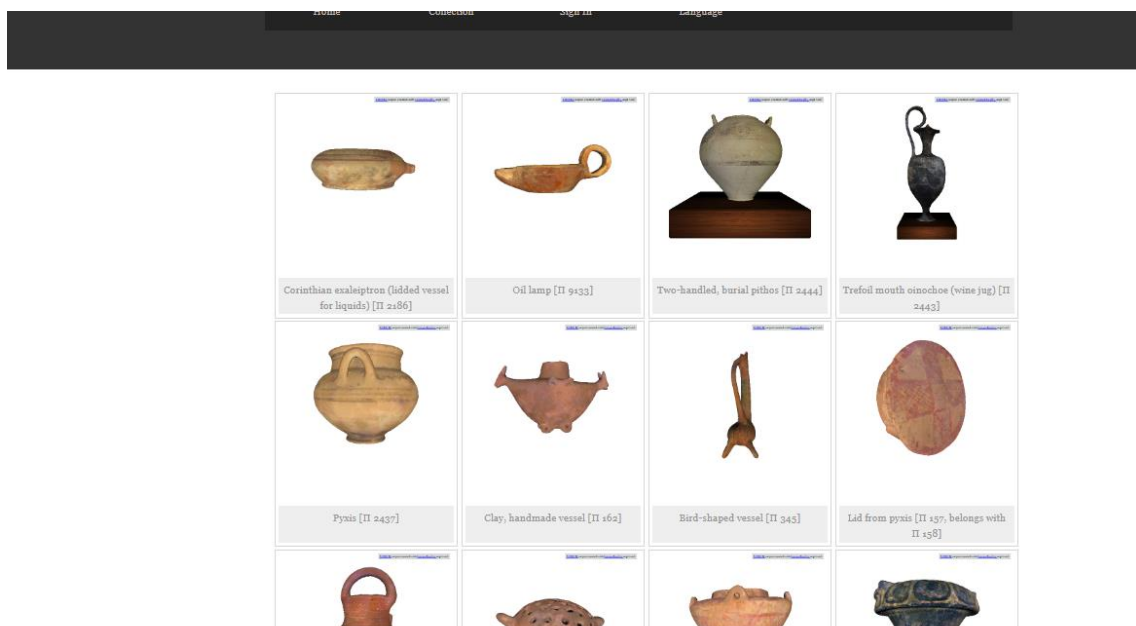


Εικόνα 5.3: αναζήτηση βάση σχήματος

Η μπάρα πλοήγησης που υπάρχει στην κορυφή της σελίδας έχει τοποθετηθεί σε κάθε μία από τις σελίδες του ιστότοπου. Από εκεί παρέχεται η δυνατότητα πλοήγησης στο σύνολο των αντικειμένων της βάσης στο πεδίο "Συλλογή", αλλαγή γλώσσας εμφάνισης της σελίδας, αναζήτηση πλήρους κειμένου στη βάση και είσοδος ή δημιουργία νέου χρήστη από το πεδίο "Είσοδος".

5.2.2 Περιήγηση στη συλλογή

Κατά την επιλογή "Συλλογή" ο χρήστης ανακατευθύνεται στη σελίδα *collections.php* όπου και μπορεί να δει το σύνολο όλων των αντικειμένων που βρίσκονται στη βάση. Με την χρήση ερωτήματος *query* επιστρέφει το αναγνωριστικό του κάθε αντικειμένου, το περιεχόμενο της οντότητας *object* που περιέχει την τοποθεσία του thumbnail, το όνομα του αγγείου από την οντότητα *appellation* ανάλογα με την γλώσσα που έχει επιλέξει ο χρήστης από την μπάρα πλοήγησης και το αναγνωριστικό της κατηγορίας στην οποία ταξινομήθηκε.



Εικόνα 5.4: περιήγηση στη συλλογή

Αν επιλεγθεί κάποιο αντικείμενο για προβολή, γίνεται ένας έλεγχος για το αν πρόκειται για κάποιον εγγεγραμμένο χρήστη της βάσης ή αν είναι απλά επισκέπτης και αναλόγως θα προβληθούν μαζί με το τρισδιάστατο αντικείμενο είτε η πληροφορία της καταχώρισης από τον οργανισμό απ' όπου προήλθαν(στην προκειμένη περίπτωση οι πληροφορίες από το μουσείο των Χανίων) είτε οι πληροφορίες σχετικά με την ταξινόμηση βάση σχήματος. Για την προβολή του τρισδιάστατου αντικείμενου στη σελίδα γίνεται χρήση του *X3DOM*. Πρόκειται για ένα πλαίσιο JavaScript ανοιχτού λογισμικού που επιτρέπει την προβολή τρισδιάστατων σκηνών σε ιστοσελίδες χωρίς την ανάγκη εγγραφής κώδικα απλά μία δομημένη αναπαράσταση κειμένου. Στην περίπτωση του *X3DOM*, αυτή η αναπαράσταση κειμένου είναι μέρος ενός εγγράφου HTML που αντιπροσωπεύει μια ιστοσελίδα. Το μόνο που χρειάζεται ένας φυλλομετρητής για την προβολή του τρισδιάστατου είναι τυπικές τεχνολογίες περιήγησης, όπως *HTML5* και το *WebGL*[23].

DUTH

Home
Collection
Sign In
Language

General Type: OPEN VESSELS

Shape: exaleipton

Type of Ware: plain

Sub-type of Ware :N/A

Use: container of liquids

Chronology: N/A

Description: Vessel the typical shape of which has cylindrical body and an incurving lip that would prevent spillage of the contents whilst carrying. In the sixth century, some may have three feet. Later examples tend to have a high splaying foot.

Example vase of this category

Εικόνα 5.5: πληροφορίες που προβάλλονται σε μη εγγεγραμμένο χρήστη

Ο επισκέπτης μπορεί να δει την ταξινόμηση σχήματος του αγγείου καθώς και πληροφορίες που χαρακτηρίζουν αυτή την κατηγορία αγγείων, μαζί με ένα παράδειγμα τέτοιου αγγείου.

Κορινθιακό εξάλειπτρο [Π 2186]

Περιγραφή: Κορινθιακό εξάλειπτρο με χαμηλή, δακτυλιόσχημη βάση, πεπλατυσμένο σώμα και χείλος που στρέφεται προς το εσωτερικό του αγγείου. Μία οριζόντια, ταινιωτή λαβή στο μέσο του σώματος πλαισιώνεται από ακανθοειδείς αποφύσεις. Εξωτερικά το αγγείο φέρει γραπτές, διακοσμητικές, ανισομεγέθεις, μαύρου και ερυθρού χρώματος ταινίες, ενώ τον ώμο κοσμεύ ζώνη με δύο σειρές γραπτών στιγμών. Το αγγείο προέρχεται από τη Φαλάσαρνα και χρονολογείται στον α μισό του 5ου αι. π.Χ.

Τύπος στοιχείου: **εξάλειπτρο, ατλό, N/A**

Εποχή: 5ος αι. π.Χ. (Α μισό) Πρώιμη Κλασική εποχή

Από: N/A

Μέχρι: N/A

Κλειδιά: Αγγείο

Υλικό: Πηλός

Πηγή: Αρχείο της Εφορείας Αρχαιοτήτων Χανίων

Copyrights: Αρχείο της Εφορείας Αρχαιοτήτων Χανίων

Δικαιώματα πρόσβασης: Αρχαιολογικό Μουσείο Χανίων, AVMap GIS A.E.

Δικαιώματα αναπαραγωγής: <https://creativecommons.org/licenses/by-nc-nd/3.0/gr/deed.en>

Δικαιώματα Europeana: Creative Commons - Attribution, Non-Commercial, No Derivatives (BY-NC-ND)

Δικαιώματα μεταδεδομένων: Attribution-NonCommercial-NoDerivs 3.0 Greece (CC BY-NC-ND 3.0 GR)

Χώρα: Ελλάδα

Περιοχή: Φαλάσαρνα

Αναφορές: N/A

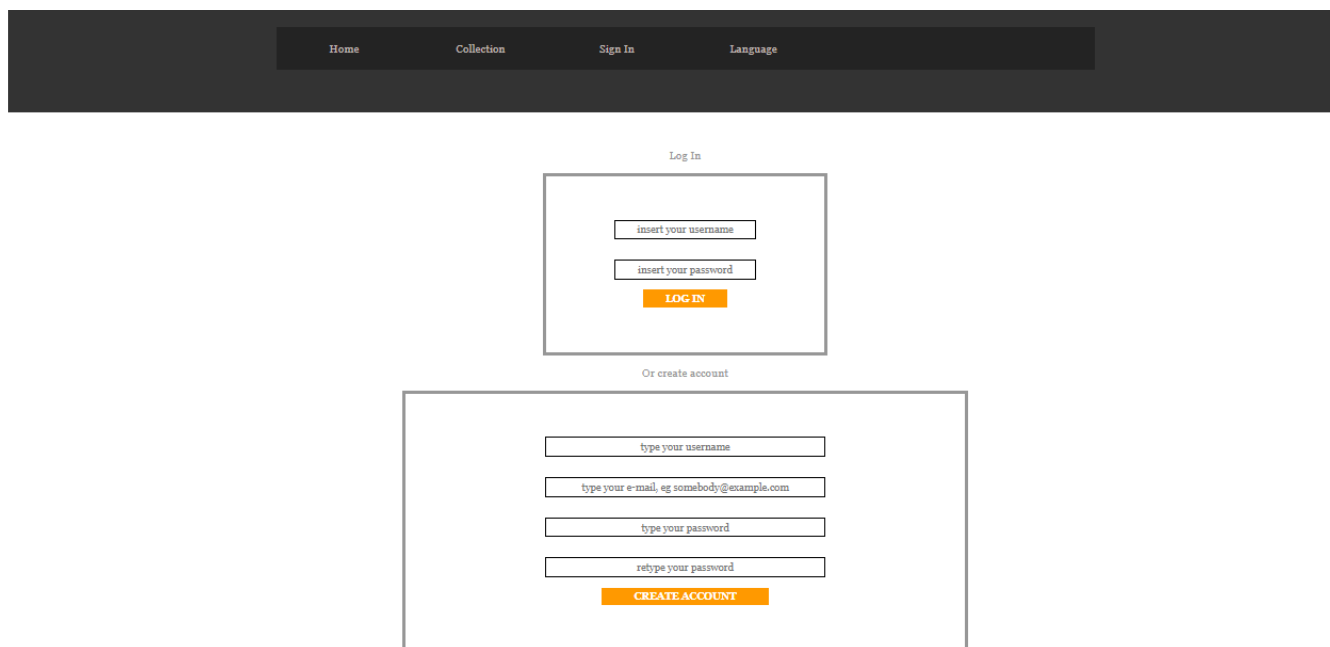


Εικόνα 5.6: πληροφορίες που προβάλλονται σε εγγεγραμμένο χρήστη

Ο εγγεγραμμένος χρήστης βλέπει την περιγραφή του συγκεκριμένου αγγείου, την εποχή στην οποία ανήκει, το υλικό κατασκευής του, τα δικαιώματα που το χαρακτηρίζουν και μπορεί επίσης να δει και τις πληροφορίες της ταξινόμησης πατώντας στο πεδίο “Τύπος στοιχείου”.

5.2.3 Σύνδεση χρήστη/δημιουργία νέου λογαριασμού

Αν επιλεγθεί από την μπάρα πλοήγησης το πεδίο “Είσοδος” θα εμφανιστεί στην οθόνη η σελίδα εισόδου/δημιουργία λογαριασμού. Ο κώδικας για τη συγκεκριμένη ενέργεια βρίσκεται στο *session.php*.

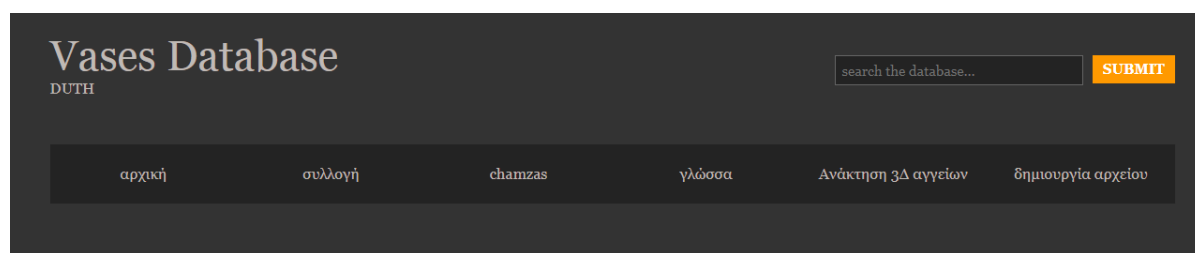


Εικόνα 5.7: είσοδος χρήστη/δημιουργία λογαριασμού

Για την είσοδο του χρήστη, γίνεται έλεγχος στη βάση για ύπαρξη του και στην περίπτωση που το όνομα χρήστη αντιστοιχίσετε ελέγχετε αν ο κωδικός είναι σωστός. Αν δεν προκύψει κάποιο πρόβλημα ο χρήστης ανακατευθύνεται στην αρχική σελίδα. Για τη δημιουργία νέου χρήστη, τα δεδομένα εισόδου ελέγχονται από τον κώδικα *register.php*. Εκεί γίνεται έλεγχος ήδη ύπαρξης λογαριασμού με το ίδιο όνομα χρήστη και για τυχών κενά πεδία στη φόρμα συμπλήρωσης. Αν η εγγραφή γίνει με επιτυχία δημιουργείται στη βάση ο χρήστης και μία ομάδα χρηστών με το ίδιο όνομα στην οποία βρίσκεται ο λογαριασμός που μόλις δημιουργήθηκε και ο λογαριασμός του διαχειριστή της βάσης καθώς και ένας φάκελος στο χώρο αποθήκευσης δεδομένων του διακομιστή, όπου και θα αποθηκεύονται οι διάφορες καταχωρίσεις τρισδιάστατων αντικειμένων που θα γίνουν από αυτόν τον λογαριασμό. Προκειμένου να ολοκληρωθεί η εγγραφή, απαιτείται έγκριση από τον διαχειριστή της βάσης ή κάποιο άλλο εξουσιοδοτημένο άτομο.

Μετά τη σύνδεση στη βάση με τα κατάλληλα διαπιστευτήριά, εκτός από τη δυνατότητα προβολής των πληροφοριών που χαρακτηρίζουν το κάθε αντικείμενο, ο χρήστης μπορεί να κάνει και τα εξής:

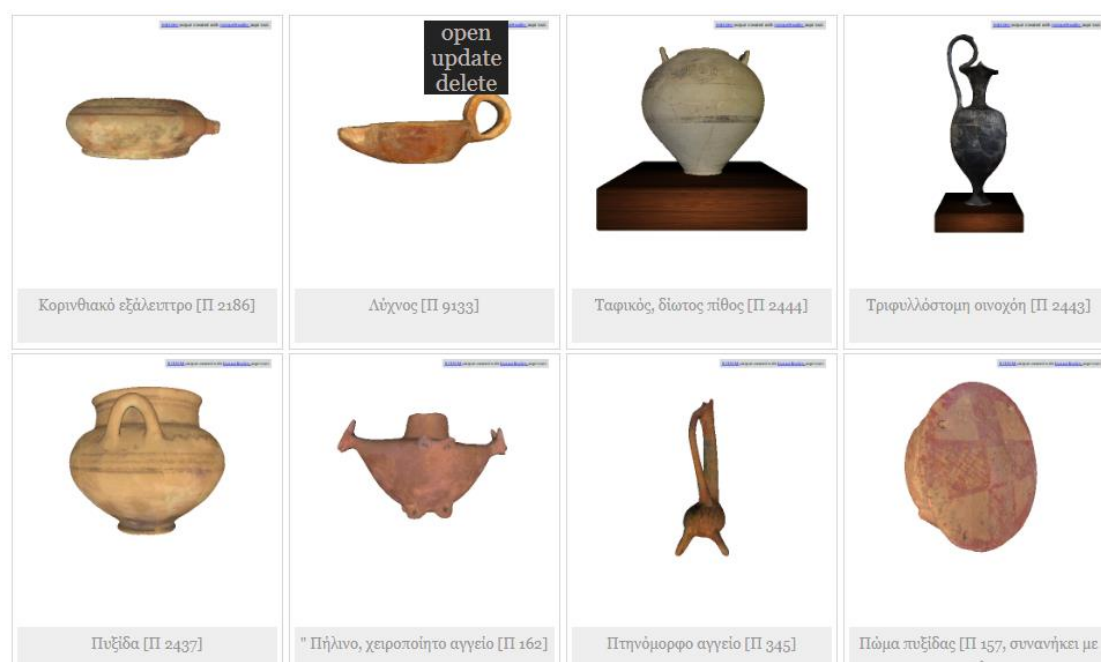
- Προβολή της συλλογής αντικειμένων που πρόσθεσε ο ίδιος ο χρήστης,
- Εισαγωγή νέων αρχείων,
- Αντιγραφή των τρισδιάστατων αρχείων της βάσης στον υπολογιστή του μαζί με ένα αρχείο αληθείας με την ταξινόμηση των αντικειμένων, για σκοπούς ανάκτησης πληροφοριών.



Αποθετήριο τρισδιάστατων απεικονίσεων αγγείων έκδοση 0.1

Εικόνα 5.8: μπάρα πλοήγησης μετά τη σύνδεση χρήστη

Στη προσωπική συλλογή του χρήστη δίνονται τρεις επιλογές για κάθε αντικείμενο, η προβολή του, η ενημέρωση των πληροφοριών στο αρχείο XML και η διαγραφή του.



Εικόνα 5.9: επιλογή προβολής, ενημέρωσης ή διαγραφής του αντικειμένου

Αν επιλεγθεί να διαγραφεί το αρχείο, ο χρήστης θα ερωτηθεί αν όντως επιθυμεί να το διαγράψει και σε θετική απάντηση του χρήστη το έγγραφο XML θα διαγραφεί από τη βάση. Η επιλογή “open” θα εμφανίσει το τρισδιάστατο αντικείμενο μαζί με τις πληροφορίες του και η επιλογή “update” θα μεταφέρει τον χρήστη σε μία άλλη σελίδα όπου τα μεταδεδομένα θα φορτωθούν σε μία φόρμα HTML, οργανωμένα ανάλογα με την XML οντότητα στην οποία ανήκουν. Για παράδειγμα οι πληροφορίες του *Collection Information* είναι βρίσκονται στην ίδια κατηγορία και είναι ξεχωριστά από τις πληροφορίες του *Heritage Asset Identification*. Σε κάθε οντότητα επισυνάπτεται ένα κείμενο σχολιασμού, που εμφανίζει τις πληροφορίες από την οντότητα *xs:annotation* του σχήματος CARARE για την περιγραφή των πληροφοριών που περιγράφει η αντίστοιχη οντότητα. Για την επανάληψη οντοτήτων που μπορούν να υπάρχουν

στο έγγραφο XML παραπάνω από μία φορές, όπως για παράδειγμα το πεδίο *appellation*, έχει τοποθετηθεί το κουμπί “repeat element”. Όταν πατηθεί, καλείται μία συνάρτηση Javascript που τοποθετεί ένα πανομοιότυπο πεδίο συμπλήρωσης κάτω από το πεδίο προς επανάληψη.

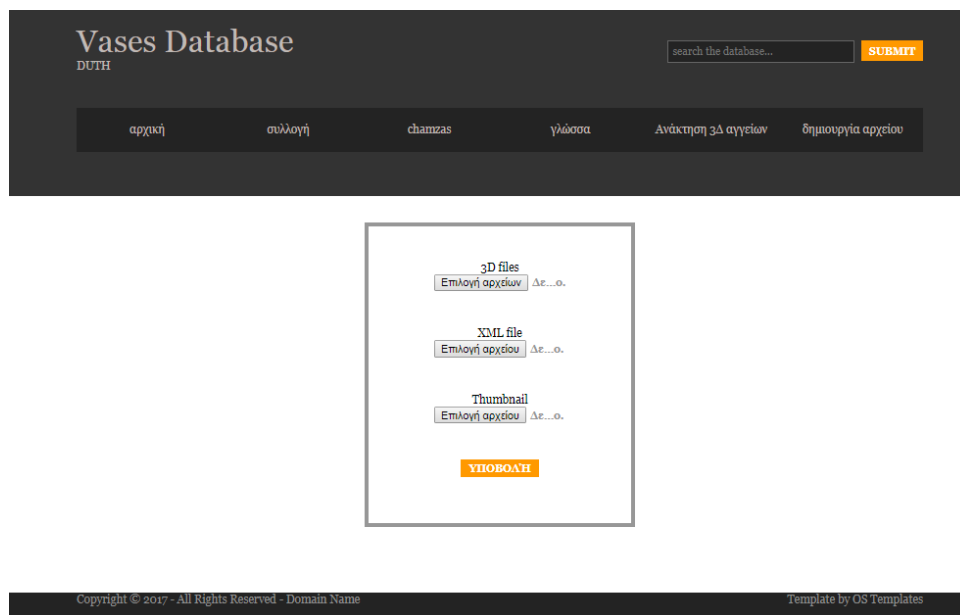
Για τη ταξινόμηση βάσει σχήματος του αγγείου υπάρχουν τρία πεδία *Heritage Asset Type*, ένα για κάθε επίπεδο ταξινόμησης εκτός του πρώτου επιπέδου. Για την επιλογή του δεύτερου επιπέδου υπάρχει μία λίστα με όλες τις κατηγορίες του επιπέδου που μπορεί να ανήκει ένα αγγείο. Αφού επιλεγεί μία από αυτές, στέλνεται ένα αίτημα *Ajax* στο διακομιστή και ανάλογα με την επιλογή που έγινε εμφανίζεται μία λίστα με τις πιθανές υποκατηγορίες που μπορεί να περιέχει το δεύτερο επίπεδο ταξινόμησης, στο δεύτερο πεδίο *Heritage Asset Type*. Με τον ίδιο τρόπο επιλέγετε και το τέταρτο επίπεδο ταξινόμησης. Μετά την αποστολή της φόρμας ελέγχεται αν τα πεδία έχουν συμπληρωθεί και αν οι επιλογές που επιλογές που έγιναν αντιστοιχούν σε κάποια από της υπάρχουσες κατηγορίες αγγείων. Αν κάτι από τα παραπάνω δεν ισχύει τότε η φόρμα δεν αποστέλλεται και ο χρήστης θα πρέπει να τη συμπληρώσει ξανά.

Εικόνα 5.10: φόρμα ενημέρωσης

Στη φόρμα ενημέρωσης εκτός από τα πεδία *Heritage Asset Type* υπάρχουν και άλλα πεδία που είναι απαραίτητο να συμπληρωθούν. Τα πεδία αυτά περιέχουν έναν αστερίσκο. Σε περίπτωση μη συμπλήρωσής τους το έγγραφο XML δεν θα θεωρείται έγκυρο.

Η εισαγωγή νέων καταχωρήσεων μπορεί να γίνει με δύο τρόπους, με απευθείας ανέβασμα στη βάση του XML εγγράφου μαζί με ένα thumbnail και τα απαραίτητα αρχεία για την αναπαράσταση του τρισδιάστατου (ένα αρχείο x3d μαζί με μία εικόνα για την εμφάνιση της υφής του) ή χειροκίνητα. Κατά την αντιγραφή του XML στη βάση γίνεται έλεγχος εγκυρότητάς του σε σχέση με το σχήμα *CARARE*.

Αν επιλεγεί ο δεύτερος τρόπος, εμφανίζετε μία φόρμα συμπλήρωσης της ίδιας μορφής με αυτής για την ενημέρωση των πληροφοριών με τη διαφορά πως απαιτείται ανέβασμα ενός thumbnail και ενός αρχείο x3d.



Εικόνα 5.11: δημιουργία καταχώρισης με απευθείας ανέβασμα αρχείου

Για την αντιγραφή των τρισδιάστατων αρχείων της βάσης, ο χρήστης πρέπει να επιλέξει την επιλογή “Ανάκτηση 3Δ αγγείων” από τη μπάρα πλοήγησης. Από εκεί θα μεταφερθεί σε μία άλλη σελίδα όπου μπορεί, πατώντας σε ένα σύνδεσμο, να κατεβάσει ένα αρχείο μορφής *zip* που περιέχει όλα τα τρισδιάστατα αρχεία μαζί με ένα αρχείο αληθείας της ταξινόμησης. Η ανάπτυξη του κώδικα για την κατασκευή των αρχείων αληθείας και *zip* έγινε με τη συμβολή του Αλέξη Ιωαννάκη. Τα αρχεία αυτά, δημιουργούνται από τον διαχειριστή της βάσης ανά τακτά χρονικά διαστήματα για κάθε νέα έκδοση της βάσης, ώστε να περιέχονται τα ίδια τρισδιάστατα αγγεία και στο αρχείο αληθείας όσο και στο *zip*. Ο λόγος που επιλέχθηκε αυτός ο τρόπος υλοποίησης, είναι η πιο δίκαιη αξιολόγηση αλγορίθμων ανάκτησης πληροφορίας. Μέχρι αυτή την στιγμή η βάση βρίσκεται στην έκδοση μηδέν(0).

Εδώ μπορείτε να κατεβάσετε τη βάση για σκοπούς ανάκτησης.

-Η βάση διατίθεται μόνο για ερευνητικούς σκοπούς.

-Περιέχει 100 μοντέλα και μέχρι στιγμής προσφέρετε η έκδοση 0.

-Συμπεριλαμβάνεται ένα αρχείο αληθείας με βάθος ταξινόμησης 3 επιπέδων.

[Κατεβάστε](#)

Εικόνα 5.12: σελίδα κατεβάσματος βάσης

Κεφάλαιο 6ο: Συμπεράσματα

Στην παρούσα διπλωματική εργασία δημιουργήθηκε μία βάση δεδομένων αποθήκευσης τρισδιάστατων αντικειμένων που εκπληρώνει δύο σκοπούς: α) να εξυπηρετεί επισκέπτες που τους ενδιαφέρει να δουν αντικείμενα πολιτιστικής κληρονομιάς όχι μόνο σε δύο διαστάσεις αλλά και τρισδιάστατά, β) να επιτρέπει σε χρήστες να κατεβάζουν καθώς και να ανεβάζουν τα δικά τους τρισδιάστατα αρχεία για με σκοπό την αξιολόγηση αλγορίθμων ανάκτησης πληροφορίας.

Περαιτέρω βελτιώσεις που μπορούν να γίνουν, θα ήταν η αύξηση των καταχωρίσεων της βάσης, η οργάνωση νέων ταξινομήσεων, καθώς και ενέργειες που θα επέτρεπαν την αποθήκευση και άλλου είδους αντικειμένων εκτός από αγγεία. Επίσης βελτιώσεις μπορούν να γίνουν και ως προς τη φιλικότητα του περιβάλλοντος της σελίδας προς τον χρήστη.

Παραρτήματα

Παράρτημα Α' κώδικες

Οι κώδικες βρίσκονται στη διεύθυνση: https://github.com/OrionVases/orion_db











Παράρτημα Β' ταξινόμηση αγγείων βάση σχήματος

Η ταξινόμηση έγινε σε συνεργασία με τον Αρχαιολόγο Ιωάννη Μούρθο.




Ανοιχτά αγγεία

ΣΧΗΜΑ	ΤΥΠΟΙ ΑΓΓΕΙΩΝ	ΧΡΗΣΗ	ΕΙΚΟΝΑ
γαβάθα	λεκάνη	ποδόλουτρο	
γαβάθα	λεκανίδα	μαγειρικό σκεύος, πυξίδα	
εξάλειπτρο	τριποδικό εξάλειπτρο-κώθων	αρωματοδοχείο	
εξάλειπτρο	πλημοχή	αρωματοδοχείο (οικιακό και τελετουργικό)	
κάνθαρος	βοιωτικός	τελετουργικό αγγείο πόσεως	
	γεωμετρικός	τελετουργικό αγγείο πόσεως	
	με πλαστική διακόσμηση	τελετουργικό αγγείο πόσεως	
	μόνωτος	τελετουργικό αγγείο πόσεως	N/A
	τύπου Α	τελετουργικό αγγείο πόσεως	N/A
	τύπου Β	τελετουργικό αγγείο πόσεως	N/A
	τύπου Γ	τελετουργικό αγγείο πόσεως	N/A

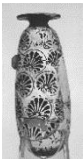



κλεψύδρα			μέτρηση χρόνου	
κρατήρας	γεωμετρικός		σήμα τάφου	
κρατήρας	ελικωτός		μίξη οίνου	
κρατήρας	καλυκωτός		μίξη οίνου	
κρατήρας	κιονωτός	αττικός	μίξη οίνου	
κρατήρας		κορινθιακός	μίξη οίνου	
κρατήρας		λακωνικός	μίξη οίνου	
κρατήρας		χαλκιδικός	μίξη οίνου	N/A
κρατήρας	κωδωνόσχημος		μίξη οίνου	
κρατήρας	κάλαθος		μίξη οίνου	N/A
κύαθος			σερβίρισμα οίνου	
κύλικα	με διχαλωτές λαβές		αγγείο πόσεως	
κύλικα	άποδη		αγγείο πόσεως	










κύλικα	Βρουλιάς	αγγείο πόσεως	
κύλικα	Γορδίου	αγγείο πόσεως	N/A
κύλικα	Κάσσελ	αγγείο πόσεως	N/A
κύλικα	Κωμαστών	αγγείο πόσεως	N/A
κύλικα	λακωνική		N/A
κύλικα	Ντρόουπ	αγγείο πόσεως	
κύλικα	Σιάννων	αγγείο πόσεως	
κύλικα	ταινιωτή	αγγείο πόσεως	N/A
κύλικα	τύπου Α	αγγείο πόσεως	
κύλικα	τύπου Α-οφθαλμωτή	αγγείο πόσεως	
κύλικα	τύπου Β	αγγείο πόσεως	
κύλικα	τύπου Γ	αγγείο πόσεως	
κύλικα	χειλεωτή	αγγείο πόσεως	
λάκαινα		αγγείο πόσεως	
λέβης/δίνος	με υπόστατο	μαγειρική και μίξη οίνου	
λέβης/δίνος	χωρίς υπόστατο	μαγειρική και μίξη οίνου	N/A

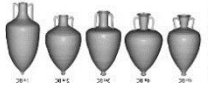

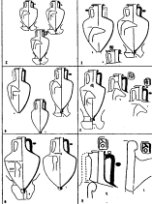





λέβης/δίνος	γαμικός	τύπου Α	νυφικό λουτρό	
λέβης/δίνος		τύπου Β	νυφικό λουτρό	
λουτήριο			νεκρικό λουτρό- σήμα τάφου	
πινάκιο	απλό		συμποσίου	
πινάκιο	με πόδι		συμποσίου	
πινάκιο	ιχθυοπινάκιο		συμποσίου	
σκύφος	καβειρικός		αγγείο πόσεως- λατρευτική χρήση	
σκύφος	κοτύλη		αγγείο πόσεως	
σκύφος	μεγαρικός		αγγείο πόσεως	
σκύφος	τύπου Α		αγγείο πόσεως	




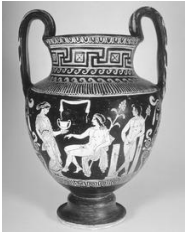


σκύφος	τύπου Β	αγγείο πόσεως	
σκύφος	μαστός	αγγείο πόσεως	
στάμνος			
φιάλη		τελετουργικό (για σπονδές)	N/A









Κλειστά αγγεία







ΣΧΗΜΑ	ΤΥΠΟΙ ΑΓΓΕΙΩΝ		ΧΡΗΣΗ	ΕΙΚΟΝΑ
αλάβαστρο	αττικό		αρωματοδοχείο	
αλάβαστρο	κορινθιακό		αρωματοδοχείο	
αμφορέας	ενιαίος	νικοσθένειος	αποθήκευση και μεταφορά	N/A
αμφορέας	ενιαίος	τύπου Α	αποθήκευση και μεταφορά	
αμφορέας	ενιαίος	τύπου Β	κυρίως ταφική (τεφροδόχος, σήμα τάφου και κτέρισμα)	N/A
αμφορέας	ενιαίος	τύπου Γ		

αμφορέας	ενιαίος	πελίκη			
αμφορέας	με λαιμό	απουλικός (ειδικού σχήματος)		αποθήκευση και μεταφορά οίνου	
αμφορέας	με λαιμό	γεωμετρικός		σήμα τάφου	
αμφορέας	κλαζομενικός	κατηγορία Knirionitch		αποθήκευση και μεταφορά	
αμφορέας	με λαιμό	λουτροφόρος		ταφική (6ος αι. π.Χ.) και τελετουργική (5ος και 4ος αι. π.Χ.)	
αμφορέας	με λαιμό	παναθηναϊκός		έπαθλο	
αμφορέας	με λαιμό	τύπου SOS		αποθήκευση και μεταφορά	
αμφορέας	με λαιμό	οξυπύθμενος	θασιακός	αποθήκευση και μεταφορά	
αμφορέας	με λαιμό	οξυπύθμενος	ροδιακός	αποθήκευση και μεταφορά	


αμφορέας	με λαιμό	οξυπύθμεν ος	κωακός	αποθήκευση και μεταφορά	
αμφορέας	με λαιμό	οξυπύθμεν ος	χιακός	αποθήκευση και μεταφορά	
αμφορέας	με λαιμό	οξυπύθμεν ος	κνιδιακ ός	αποθήκευση και μεταφορά	
αμφορέας	με λαιμό	οξυπύθμεν ος	ρωμαϊκ ός	αποθήκευση και μεταφορά	
αμφορέας	με λαιμό	τύπου Nola	τύπου Nola	αποθήκευση και μεταφορά	
αμφορέας	πρωτοαττικός			σήμα τάφου	
αμφορέας	τυρρηνικός			αποθήκευση και μεταφορά	
αμφορέας	νεστορίδα	la		αποθήκευση και μεταφορά	








αμφορέας	νεστορίδα	Ib	αποθήκευση και μεταφορά	
αμφορέας	νεστορίδα	IIa	αποθήκευση και μεταφορά	
αμφορέας	νεστορίδα	IIb	αποθήκευση και μεταφορά	
αμφορέας	νεστορίδα	III	αποθήκευση και μεταφορά	
αμφορέας	αμφορίσκος		οικιακή (φύλαξη ψιμμυθίων, αρωμάτων, αλοιφών), αναθηματική, ταφική (κτέρισμα)	
αμφορέας	κάδος (situla)		αποθήκευση και μεταφορά	



αμφορέας	ψευδόστομος		αρωματοδοχείο-ελαιοδοχείο	
αρύβαλλος	αττικός		ελαιοδοχείο-αρωματοδοχείο	
αρύβαλλος	κορινθιακός	απιόσχημος	ελαιοδοχείο-αρωματοδοχείο	
αρύβαλλος	κορινθιακός	σφαιρικός	ελαιοδοχείο-αρωματοδοχείο	
αρύβαλλος	κορινθιακός	ωοειδής	ελαιοδοχείο-αρωματοδοχείο	
αρύβαλλος	λακωνικός		ελαιοδοχείο-αρωματοδοχείο	
αρύβαλλος	με πλαστική διακόσμηση		ελαιοδοχείο-αρωματοδοχείο	
ασκός			ελαιοδοχείο	
θήλαστρο			τροφοδότηση βρέφους	N/A
λήκυθος	κοινή		οικιακή (για έλαια αρώματα ή	N/A

			καρυκεύματα) και τελετουργική (ταφική)	
λήκυθος	σφαιρική ή αρυβαλλοειδής		οικιακή (για έλαια αρώματα ή καρυκεύματα) και τελετουργική (ταφική)	
λήκυθος	με ώμο		οικιακή (για έλαια αρώματα ή καρυκεύματα) και τελετουργική (ταφική)	
λήκυθος	τύπου Διηάνειρας		οικιακή (για έλαια αρώματα ή καρυκεύματα) και τελετουργική (ταφική)	
λύδιο	ιωνικό		αρωματοδοχείο	
λύδιο	λυδικό		αρωματοδοχείο	
οινοχόη	πτολεμαϊκή		σερβίρισμα οίνου	N/A
οινοχόη	τριφυλλόστομη	απλή	σερβίρισμα οίνου	

οινοχόη		δίσωμη	σερβίρισμα οίνου	
οινοχόη	τύπου I		σερβίρισμα οίνου	
οινοχόη	τύπου II		σερβίρισμα οίνου	
οινοχόη	τύπου III/χους		σερβίρισμα οίνου	
οινοχόη	τύπου V- όλπη	Va	σερβίρισμα οίνου	
οινοχόη	τύπου V- όλπη	Vb	σερβίρισμα οίνου	
οινοχόη	τύπου VI		σερβίρισμα οίνου	

οινοχόη	τύπου VII	σερβίρισμα οίνου	
οινοχόη	τύπου VIII	σερβίρισμα οίνου	
οινοχόη	τύπου X	σερβίρισμα οίνου	
πίθος		αποθήκευση- ταφή	
πιθαμφορέ ας		αποθήκευση	N/A
πυριατήρι		κτέρισμα τάφων	N/A
πυξίδα	γεωμετρική	φύλαξη μικρών αντικειμένων	
πυξίδα	νικοσθένεια	φύλαξη μικρών αντικειμένων	
πυξίδα	τύπου A	φύλαξη μικρών αντικειμένων	

πυξίδα	τύπου Β	φύλαξη μικρών αντικειμένων	
πυξίδα	τύπου Γ	φύλαξη μικρών αντικειμένων	
πυξίδα	τύπου Δ	φύλαξη μικρών αντικειμένων	
ρυτό		σπονδικό αγγείο (τελετουργικό) ή αγγείο πόσεως	N/A
υδρία	τύπου Hadra	υδροδοχεία, τεφροδόχοι	
υδρία	κάλπες	υδροδοχεία, τεφροδόχοι, κάλπες	
υδρία	λουτροφόρος	τελετουργική	
υδρία	με λαιμό	υδροδοχεία, τεφροδόχοι, κάλπες	

φορμίσκος		άθυρμα, κτέρισμα τάφου	
ψυκτήρας		ψύξη οίνου	
λάγυνος			N/A

Μικροαντικείμενα

ΣΧΗΜΑ	ΤΥΠΟΙ ΑΓΓΕΙΩΝ	ΧΡΗΣΗ	ΕΙΚΟΝΑ
λύχνος		φωτισμός	N/A
Θυμιατήρι		κάψιμο αρωματικών ουσιών	N/A
φωλεόσχημο σκεύος			N/A

Βιβλιογραφία

- [1] ΚΕΝΤΡΟ ΠΛΗ.ΝΕ.Τ. Ν. ΦΛΩΡΙΝΑΣ, Η Θεωρία των Βάσεων Δεδομένων
- [2] Διπλωματική εργασία «Σχεδιασμός και Ανάπτυξη Προσαρμοζόμενου Συστήματος για την Διδασκαλία του Μαθήματος Βάσεις Δεδομένων», Ξηροτύρης Νικόλαος, 2012
- [3] Concise Guide to Databases, Lake Peter. Crowther Paul
- [4] Διπλωματική εργασία «PhytoKaryon: Μία κυτταρολογική βάση δεδομένων των φυτών της Ευρώπης και της Μεσογείου – Αξιοποίηση και παρουσίαση δεδομένων. Ι», Πέτρος Σταυρόπουλος, 2009
- [5] Three-Level ANSI-SPARC Architecture, w3schools
- [6] Extensible Markup Language (XML) 1.0 (Fifth Edition), W3C Recommendation, 2008
- [7] MarkLogic Data Modeling Guidelines, Imran Chaudhri and Damon Feldman, 2015
- [8] Native XML databases, JH Gerritsen - 5th Twente Student Conference on IT, 2006
- [9] w3schools.com, XML Tree
- [10] Namespaces in XML 1.0 (Third Edition), W3C Recommendation, 2009
- [11] W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures, W3C Recommendation, 2012
- [12] XQuery and XPath Data Model 3.1, W3C Recommendation, 2017
- [13] eXist: An Open Source Native XML Database, Wolfgang Meier
- [14] HTML 5.2, W3C Recommendation, 2017
- [15] php.net, PHP manual
- [16] RETRIEVAL - An Online Performance Evaluation Tool for Information Retrieval Methods, George Ioannakis, Member, IEEE, Anestis Koutsoudis, Senior Member, IEEE, Ioannis Pratikakis, Senior Member, IEEE, Christodoulos Chamzas, Senior Member, IEEE
- [17] Τα Αγγεία ως Χρηστικά Αντικείμενα-Σχεδιασμός Συστήματος Τεκμηρίωσης των Χρήσεων της Αρχαίας Κεραμικής, Μεταπτυχιακή εργασία, Ανθή Ν. Κουκουράκη
- [18] Χρήσεις αττικών αγγείων κατά τους αρχαϊκούς και κλασικούς χρόνους με βάση την εικονογραφική και γραπτή παράδοση, διδακτορική διατριβή, Κωνσταντίνα Τσονάκα
- [19] D6.1 Report on Metadata and Thesaurii, Andrea D'Andrea (CISA), Kate Fernie (MDR)

[20] The CARARE metadata schema, v.2.0, Kate Fernie, Dimitris Gavriliis and Stavros Angelis

[21] <https://pear.php.net/>

[22] https://pear.php.net/package/XML_RPC2

[23] <https://doc.x3dom.org/gettingStarted/background/index.html>