

# **Προγραμματιστική Εργασία**

## **Ανάπτυξη Εφαρμογής υποστηριζόμενης από NoSQL**

### **Βάση Δεδομένων**

Λευκοπούλου Ελένη Μαρία 2557

**Προχωρημένη Διαχείριση Δεδομένων 2021**  
Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών  
Πανεπιστήμιο Θεσσαλίας, Βόλος  
elefkopoulou@e-ce.uth.gr

## **1 Επιλογή θέματος**

Στα πλαίσια της προγραμματιστικής εργασίας του μαθήματος επέλεξα να υλοποιήσω μια εφαρμογή για εύρεση και καταχώρηση ακινήτων. Εμπνευσμένη από παρόμοιες εφαρμογές σχεδίασα μια ιστοσελίδα που λειτουργεί σαν ένα ευρύτερο μεσητικό γραφείο και δέχεται αγγελίες ακινήτων από πολλά μεσητικά και ιδιώτες. Οι χρήστες μπορούν να αναζητούν στη σελίδα διάφορα ακίνητα .

## **2 Περιγραφή και προδιαγραφές του θέματος.**

Πιο συγκεκριμένα το e-spriti είναι μια ιστοσελίδα όπου οποιοδήποτε μεσητικό γραφείο ή ιδιώτης μπορεί να κάνει login / sign up ώστε να καταχωρήσει τα ακίνητα τα οποία θέλει. Σε όσους έχουν λογαριασμό δίνεται η δυνατότητα να κάνουν update ή να διαγράψουν κάποιο ακίνητο από αυτά που οι ίδιοι έχουν προσθέσει. Οι απλοί επισκέπτες έχουν τη δυνατότητα χωρίς να έχουν λογαριασμό να βλέπουν τα τρέχοντα ακίνητα. Τους δίνεται η δυνατότητα να κάνουν αναζήτηση με βάση το αν θέλουν να νοικιάσουν ή να αγοράσουν κάποιο ακίνητο. Μπορούν επίσης να επιλέξουν αν αυτό το ακίνητο είναι κατοικία, γή , ή εργασιακός χώρος. Επιπλέον τους δίνεται δυνατότητα να εισάγουν πιο συγκεκριμένα χαρακτηριστικά όπως είναι τα τετραγωνικά απο που έως που να κειμούνται , η τιμή και η πόλη. Πατώντας πάνω σε κάποιο ακίνητο μπορούν να δούνε πιο αναλυτικές πληροφορίες σχετικά με αυτό.

## **3 Επιλογή Συστήματος NoSQL ΒΔ**

Για την υλοποίηση του μεσητικού γραφείου επιλέχτηκε μια Document-based βάση δεδομένων στην οποία ένα έγγραφο μπορεί να περιέχει πολλά ζεύγη κλειδιού-τιμής ή ακόμα και εμφωλευμένα έγγραφα. Το σύστημα βάσης που επιλέχτηκε είναι η mongoDB . Πιο συγκεκριμένα το σκεπτικό ήταν να μπορούν να υποστηριχθούν το ότι κάποια ακίνητα έχουν κάποια επιπλέον χαρακτηριστικά συγκριτικά με άλλα και με τις nosql βάσεις εγγράφων αυτό δεν αποτελεί πρόβλημα. Επίσης υπάρχουν πεδία στη μορφή πίνακα που δίνουν τη δυνατότητα να έχει το κάθε ακίνητο διαφορετικό αριθμό ακινήτων.

### 3.1 mongoDB

Με την ανάπτυξη εφαρμογών μεγάλης κλίμακας όπου χρειάζεται μεγάλος όγκος αποθήκευσης δεδομένων εμφανίστηκε ένας εναλλακτικός τύπος οι NoSQL ή αλλιώς μη-σχεσιακές βάσεις όπου ξεπερνούν τα limitations των relational dbs. Η οριζόντια κλιμάκωση (scaling) ήταν ένα από τα βασικά σημεία όπου υπήρχε πρόβλημα στις σχεσιακές βάσεις και παρόλο τη χρήση τεχνικών όπως το sharding η πολυπλοκότητα διαχείρισης αυξανόταν χωρίς πραγματικές λύσεις. Η MongoDB είναι μία document-oriented database, αυτό σημαίνει ότι η αποθήκευση μιας εγγραφής έχει τη μορφή ενός document δηλαδή ενός αρχείου όπου περιέχει τα δεδομένα της εγγραφής σε μορφή value-pair που είναι JSON. Αυτό έχει ως αποτέλεσμα ότι δεν χρειάζεται εξαρχής να καθορίσουμε τη μορφή του πίνακα και το αντίστοιχο αριθμό πεδίων όπως στις σχεσιακές βάσεις δεδομένων. Η αντίστοιχη έννοια του πίνακα εδώ έχει την έννοια της συλλογής (Collection) και οι εγγραφές-έγγραφα που ανήκουν σε κάθε Collection όπου μπορούν να έχουν διαφορετικά πεδία. Γενικά αυτού του τύπου οι βάσεις είναι χρήσιμες για να λύσουν συγκεκριμένα προβλήματα και είναι ένα απαραίτητο εργαλείο για του κατασκευαστές εφαρμογών web όταν γνωρίζουν να κάνουν τη σωστή χρήση. Τα διάφορα ερωτήματα πάνω στη βάση αυτή μπορούν να γίνουν είτε μέσω κάποιου shell είτε με τη σύνδεση με κάποια εφαρμογή.

## 4 Εγκατάσταση του Συστήματος NoSQL ΒΔ

Για την εγκατάσταση της βάσης σε Windows θα χρειαστεί να πάμε στο <https://www.mongodb.com/> . Πατάμε try free. Επιλέγουμε on-premises mongoDB locally και μετά mongoDB enterprise Server. Αφού κατέβει και τρέξουμε την εγκατάσταση. Πηγαίνουμε στο C: και δημιουργούμε ένα φάκελο db και ένα φάκελο db μέσα στο data. Για τη συνέχεια από τη σελίδα <https://medium.com/@LondonAppBrewery/how-to-download-install-mongodb-on-windows-4ee4b3493514> βλέπουμε πως να δημιουργήσουμε ένα αρχείο και να είναι έτοιμο το σύστημα μας. Τώρα ανοίγοντας οπουδήποτε ένα terminal Hyper και γράφουμε mongod και σε ένα άλλο mongo και μπορούμε να τρέξουμε τα διάφορα ερωτήματα και να δούμε τις collections.

## 5 Προσδιορισμός χρήσιμων ερωτημάτων για την εφαρμογή.

Για την εφαρμογή χρειάστηκα ερωτήματα αναζήτησης, προσθήκης, διαγραφής και ενημέρωσης. Πιο συγκεκριμένα υπάρχουν και ερωτήματα αναζήτησης που δημιουργούνται δυναμικά με τις επιλογές του χρήστη.

## 6 Δεδομένα

Τα δεδομένα για να είναι αληθοφανή επειδή δεν βρήκα κάτι πολύ αξιόπιστο τα πέ- ρασα με το χέρι.

## 7 Σχεδιασμός του μοντέλου της ΒΔ και υλοποίησή του στο Σύστημα NoSQL ΒΔ

Παρακάτω φαίνεται το μοντέλο που χρησιμοποίησα για τη βάση μου για τα ακίνητα και για τους χρήστες.

Η συλλογή members για τα μέλη :

```
const memberSchema = new mongoose.Schema({
  username:{
    type:String,
    required:[true,"Please enter username"]
  },
  email:{
    type:String,
  },
  password:String,
  phone:String
});
const Member = mongoose.model("member",memberSchema);
```

Η συλλογή real\_estates για τα ακίνητα:

```
//for real estate agency properties
const real_estateSchema = new mongoose.Schema({
  title:{
    type:String,
    required:[true,"Please enter title"]
  },
  case : String,
  real_estate_type : String,
  building : String ,
  town : String ,
  region : String ,
  address : String ,
  area : Number,
  price : Number,
  floor : Number,
  parking : String,
  heating_system : String,
  rooms : Number,
  inner_char : Array,
  outer_char : Array,
  extra_char : Array,
  agent :String,
  year_of_construct : Number,
  last_update : String,
  details : String
});
const Real_estate = mongoose.model("real_estate",real_estateSchema);
```

## 8 Φόρτωση των δεδομένων στη ΒΔ

Για τη φόρτωση των δεδομένων τρέχουμε το insertion.js όπου χρησιμοποιείται η InsertMany.

## 9 Create, read, update and delete ερωτήματα

Παρακάτω φαίνονται φωτογραφίες από τα διάφορα ερωτήματα στον κώδικα.

### 9.1 Read

Όσον αφορά τη read στην ουσία είναι η find όπου στην εφαρμογή πραγματοποιείται και απλό find για όλα τα δεδομένα και πιο εξηζητημένα ανάλογα με το τι θέλει ο χρήστης. Παρακάτω φαίνονται 3 από τα ερωτήματα αυτά. Τα υπόλοιπα θα εξηγηθούν στην παρουσίαση.

```

app.post("/studio_sales",function(req, res){
    Real_estate.find( {case:'Παράγωγη', real_estate_type:'Σπίτι',building:'Στούντιο'}, function (err, result) {
        if (err) {
            console.log(err);
        }else{
            console.log(result.length);
            res.render('e-split',{data:result , username : req.body.username,page_title:'STUDIOS SALES', page_case:'sale',real_estate_type:'home',building:'studio'});
        }
    });
});

app.post("/all_sales",function(req, res){
    Real_estate.find( {case:'Παράγωγη'}, function (err, result) {
        if (err) {
            console.log(err);
        }else{
            console.log(result.length);
            res.render('e-split',{data:result , username : req.body.username,page_title:'ALL THE PROPERTIES FOR SALE', page_case:'sale',real_estate_type:'notype',building:'nobuilding'});
        }
    });
});

app.post("/one_property",function(req, res){
    console.log(req.body.property_id);
    Real_estate.findById(req.body.property_id, function (err, property) {
        if(err){
            console.log(err);
        }else{
            console.log( req.body.username);
            console.log( property.agent);
            res.render('property',{prop_data:property , username : req.body.username});
        }
    });
});

```

## 9.2 Create

Η create είναι η αντίστοιχη add που παίρνοντας στοιχεία από το χρήστη η εφαρμογή δημιουργεί τον κώδικα η insert. Επίσης insert γίνεται και στη singup. Παρακάτω φαίνονται φωτογραφίες από κώδικα.

Add new:

```
app.post("/addit",function(req, res){
  Real_estate.findOne( { title: req.body.title }, function (err, result) {
    if (err) {
      console.log(err);
    }
    if (result) {
      console.log("already exist");
      var errortitle = req.body.title;
      res.render('add',{username : req.body.username, error : errortitle});
    }
    if (!result) {
      query ={};
      if( req.body.title != "" ){
        query["title"] = req.body.title;
      }
      if( req.body.case != "" ){
        query["case"] = req.body.case;
      }
      if( req.body.real_estate_type != "" ){
        query["real_estate_type"] = req.body.real_estate_type;
      }
      if( req.body.town!= "" ){
        query["town"] = req.body.town;
      }
      if( req.body.region != "" ){
        query["region"] = req.body.region;
      }
      if(req.body.address != "" ){
        query["address"] = req.body.address;
      }
      if( req.body.area != "" ){
        query["area"] = req.body.area;
      }
    }
  })
})
```

```

if(req.body.price != ""){
  query["price"] = req.body.price;
}
if( req.body.area != ""){
  query["area"] = req.body.area;
}
if( req.body.floor != "" ){
  query["floor"] = req.body.floor;
}
if( req.body.parking != "" ){
  query["parking"] = req.body.parking;
}
if( req.body.heating_system != "" ){
  query["heating_system"] = req.body.heating_system;
}
if( req.body.details != "" ){
  query["details"] = req.body.details;
}

if( req.body.year_of_construct != "" ){
  query["year_of_construct"] = req.body.year_of_construct;
}

if( req.body.iner_char != "" ){
  const iner = req.body.iner_char.split(',');
  query["iner_char"] = iner;
}
if( req.body.outer_char != "" ){
  const outer = req.body.outer_char.split(',');
  query["outer_char"] = outer;
}
if( req.body.extra_char != "" ){
  const extra = req.body.extra_char.split(',');

  query["extra_char"] = extra;
}

```

```

  query["extra_char"] = extra;
}
query["agent"] = req.body.username;
console.log(query);
Real_estate.insertMany([query]).then(function(){
  console.log("data inserted") // Success
}).catch(function(error){
  console.log(error) // Failure
});
Real_estate.find( {}, function (err, dataresult) {
  if (err) {
    console.log(err);
  }else{
    console.log(dataresult.length);
    res.render('e-split',{data:dataresult , username : req.body.username,page_title:'ALL THE PROPERTIES', page_case:'nocase',real_estate_type:'notype',building:'nobuilding'});
  }
});
});

```

Signup:

```

app.post("/signup", function(req, res){
  Member.findOne( { username: req.body.username }, function (err, result) {
    if (err) {
      console.log(err);
    }
    if (!result) {
      const newmember = new Member({
        username : req.body.username,
        email : req.body.email,
        password : req.body.password,
        phone : req.body.phone
      });
      newmember.save();
      Real_estate.find( {}, function (err, result) {
        if (err) {
          console.log(err);
        }else{
          console.log(result.length);
          res.render("e-split1",{data:result , username : req.body.username,page_title:'ALL THE PROPERTIES', page_case:'nocase',real_estate_type:'notype',building:'nobuilding'});
        }
      });
    }
  }
  if (result) {
    console.log("user exist");
    res.render("login",{user_error_login : parseInt(0) , password_error_login : parseInt(0) , user_error_signup : parseInt(1)});
  }
});
});

```

### 9.3 Update

```

var MongoClient = require("mongodb").MongoClient;
var url = "mongodb://127.0.0.1:27017/";

MongoClient.connect(url,{ useUnifiedTopology: true }, function(err, db) {
  if (err) throw err;
  var dbo = db.db("real_estate_db");
  var myquery = dbo.collection("real_estates").findOne({ _id:req.body.id });
  var newvalues = { $set: query };
  dbo.collection("real_estates").updateOne(myquery, newvalues, function(err, res) {
    if (err) throw err;
    console.log("1 document updated");
    db.close();
    Real_estate.find( {}, function (err, dataresult) {
      if (err) {
        console.log(err);
      }else{
        console.log(dataresult.length);
        res.render("e-split1",{data:dataresult , username : req.body.username,page_title:'ALL THE PROPERTIES', page_case:'nocase',real_estate_type:'notype',building:'nobuilding'});
      }
    }
  });
});

```

### 9.4 Delete

```

app.post("/delete",function(req, res){
  Real_estate.findByIdAndRemove({_id: req.body.id}, function(err,data){
    if (err) {
      console.log(err);
    }else{
      console.log("Deleted");
      Real_estate.find( {}, function (err, dataresult) {
        if (err) {
          console.log(err);
        }else{
          console.log(dataresult.length);
          res.render("e-split1",{data:dataresult , username : req.body.username,page_title:'ALL THE PROPERTIES', page_case:'nocase',real_estate_type:'notype',building:'nobuilding'});
        }
      });
    }
  });
});

```

## 10 Διεπαφής χρήστη για λειτουργίες CRUD

Παρακάτω φαίνονται οι επιλογές του χρήστη για τα ερωτήματα αυτά στην εφαρμογή.



[LOGO](#) [SALES](#) [RENTALS](#) [LOGIN](#)

# Easily find your next property

Search whatever you want

[Land](#) [Business premises](#) [Home](#)

€  €  m<sup>2</sup>  m<sup>2</sup>

[Sale](#) [Rent](#)

LOGIN

LOGIN

SIGN UP

If you are new join us and  
sign up to e-spiti

SIGN UP

The image displays a user interface design for a login and sign-up system. It features two main components: a 'LOGIN' modal and a 'SIGN UP' form, set against a dark blue-grey background.

**LOGIN Modal:**

- Header: **LOGIN**
- Text: "If you are already user" and "login to e-spiti"
- Button: **LOGIN** (dark blue)

**SIGN UP Form:**

- Header: **SIGN UP**
- Form Fields:
  - Username
  - Phone
  - Email
  - Password
  - Confirm Password
- Button: **SIGN UP** (orange)

LOGO SALES RENTALS mariatena's ADDS ADD NEW

Price : 220 m² : 4000



Ενοικίαση. Οικόπεδο 1600 τ.μ. Κέντρο, Κοζάνη. € 950

Οικόπεδο 1600 τ.μ. στην είσοδο της πόλης της Κοζάνης. Κατάλληλο για εμπορική χρήση. Δυνατότητα κατασκευής κτιρίου για μακροχρόνια μίσθωση.....

Price : 950 m² : 1600



Ενοικίαση. Οικόπεδο 3165 τ.μ. Αβέρωφ, Λάρισα. € 1800

Διαμορφωμένος χώρος 3165τμ. μεγάλης προβολής, στον κυκλικό κόμβο της Λεωφ. Καραμανλή που συνδέει την Φαρσάλων.

Price : 1800 m² : 3165



Ενοικίαση. Studio/Γκαρσονιέρα 22 τ.μ. Άγιος Αντώνιος, Λάρισα. € 160

Κωδικός ακινήτου: 90-737 - Λάρισα Άγιος Αντώνιος ΕΝΟΙΚΙΑΖΕΤΑΙ γκαρσονιέρα συνολικής επιφάνειας 22 τ.μ. στον 5 ο όροφο . Αποτελείται από 1 υ.....


Price : 160 m² : 22

Delete

Update

### Basic info

Case :	ενοικίαση
Real estate type :	Σπίτι
Bulding :	Διαμέρισμα
Code :	60a9432ef73e4a2694aace56
Area:	44
Price:	430



## ADD A NEW PROPERTY

Title

Select case

Select real estate type

Select building

Town

LOGO


SALES

RENTALS

marialena 's ADDS


ADD NEW

### PROPERTIES THAT marialena HAS ADD




**Ενοίκηση, Διαμέρισμα 75 τ.μ., Κέντρο, Βόλος, € 450**

Διατίθεται προς ενοίκηση κατά αποκλειστικότητα, διαμέρισμα πλήρως ανακατασκευασμένο συνολικής επιφάνειας 75τμ. Το ακίνητο αποτελείται από 2 υπ.



**Ενοίκηση, Διαμέρισμα 44 τ.μ., Κέντρο, Βόλος, € 430**

Διατίθεται προς ενοίκηση και αποκλειστικότητα νεόδμητο, πολυτελές, γυναικά διαμέρισμα 1ου ορόφου στο κέντρο του Βόλου ευθαδού 44.94 τ.μ. με.....



**Ενοίκηση, Διαμέρισμα 50 τ.μ., Επτά Πλάτνια, Βόλος, € 150**

ΒΟΛΟΣ-ΕΠΤΑ ΠΛΑΤΑΝΙΑ Διατίθεται προς ενοίκηση 2άρι διαμέρισμα 50 τ.μ. συμπαγές, πλήρως ανακατασκευασμένο με.....

## 11 Υλοποίηση των ερωτημάτων στη ΒΔ

## 12 αρχεία που παραδίδονται και τι περιλαμβάνει το καθένα + οδηγίες εγκατάστασης και χρήσης της εφαρμογής σας

Παραδίδονται τα αρχεία για την ιστοσελίδα το powerpoint και το τρέχον αρχείο .Κάποιος πρέπει να έχει Atom για να ανοίξει τα αρχεία και να τα δει και το Hyper από όπου πηγαίνουμε στον κατάλογο του project και τρέχουμε node insertion.js για να αρχικοποιηθεί η βάση μας που στην ουσία είναι το αρχείο για την αρχικοποίηση. Μετα τρέχουμε node server.js και μπαίνουμε στο <http://localhost:3000/main> για να τρέξουμε

την εφαρμογή. Στο server.js είναι ο κώδικας του server και υπάρχουν υλοποιημένα όλα τα ερωτήματα. Όλες οι άλλες είναι τα ejs για τις διάφορες σελίδες .

### **13 μελλοντικές επεκτάσεις και προσθήκες**

Μελλοντικές επεκτάσεις της εφαρμογής είναι να γίνει deploy στο διαδίκτυο , να προστεθούν εικόνες στη βάση και στο κάθε ακίνητο ξεχωριστά και ίσως αναζήτηση με τα πιο εξειδικευμένα χαρακτηριστικά.