ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Διαδικτυακή εφαρμογή για την διαχείριση και ενημέρωση των πελατών ενός λογιστικού γραφείου

ΓΕΩΡΓΙΑ ΤΣΙΑΜΑΝΤΑ

Α.Μ. 11201

Επιβλέπων καθηγητής:

Δημήτριος Λιαροκάπης

ΑΡΤΑ 2014

ΠΕΡΙΛΗΨΗ

Στην παρούσα διπλωματική εργασία παρουσιάζεται μια διαδικτυακή εφαρμογή, η οποία απευθύνεται σε λογιστικά γραφεία. Ο λογιστής είναι και ο διαχειριστής του προγράμματος και έχει πρόσβαση στην σελίδα διαχείρισης, όπου μπορεί να εισάγει δεδομένα που αφορούν τα στοιχεία και τις οφειλές των πελατών. Ο πελάτης είναι ο χρήστης της ιστοσελίδας στην οποία εισάγεται για να μπορέσει να δει τις οφειλές που τον αφορούν, σε μορφή πινάκων ή διαγραμμάτων και φιλτράροντας τις επιλογές του με βάση κάποια κριτήρια που θέτει. Η εφαρμογή έχει αναπτυχθεί με βάση το πλαίσιο ανάπτυξης διαδικτυακών εφαρμογών Django, το οποίο βασίζεται στην γλώσσα προγραμματισμού Python. Επιπλέον για την ανάπτυξη της ιστοσελίδας χρησιμοποιήθηκαν οι τεχνολογιες HTML, CSS και Javascript.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ : Ανάπτυξη διαδικτυακών εφαρμογών

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ : διαδικτυακή εφαρμογή, γλώσσα προγραμματισμού Python, πλαίσιο ανάπτυξης διαδικτυακών εφαρμογών Django

ABSTRACT

In this thesis is presented a web application for an accounting office. The accountant is the application's administrator and has access to the administration page, where he can import data concerning the information about the debts of the clients. The client is the user of the web page, where he logs in, in order to see all the debts that he is concerned, in a table or a plot, by filtering the outcome, according to several criteria. The application has been developed using the web framework Django, which is based on the programming language Python. Moreover for the fond-end development were used technologies, such as HTML, CSS and Javascript.

SUBJECT AREA : Web development

KEYWORDS : web application, programming language Python, web framework Django

Πίνακας Περιεχομένων

Πίνακας Εικόνων

1. Εισαγωγή

Είναι πλέον γεγονός ότι η τεχνολογία έχει παρεισφρήσει σε όλους τους τομείς της ζωής του σύγχρονου ανθρώπου, στην εργασία, την εκπαίδευση, την κοινωνική ζωή, την συναλλαγή του με δημόσιους φορείς. Βλέπουμε ότι υπάρχει μεγάλη ανάπτυξη του ηλεκτρονικού εμπορίου, των διαδικτυακών πληρωμών, των τραπεζικών συναλλαγών μέσω διαδικτύου.

Πολύ βασικό βέβαια για τους επαγγελματίες στην σύγχρονη εποχή της οικονομικής κρίσης είναι να έχουν έλεγχο των οικονομικών τους, ώστε να μπορούν να κάνουν καλύτερη διαχείριση έχοντας σαν πληροφορία το πώς ήταν οι συναλλαγές τους και οι οφειλές τους μέχρι τώρα. Με αυτό τον τρόπο θα μπορέσουν να κάνουν καλύτερο προγραμματισμό της μελλοντικής τους δραστηριότητας και θα μπορέσουν να να αναπτυχθούν επιχειρηματικά.

1.1. Περιγραφή της εφαρμογής

Σε αυτή την εφαρμογή που αναπτύχθηκε εδώ ικανοποιούνται οι ανάγκες ενημέρωσης του επαγγελματία από τον λογιστή του όσον αφορά τις οφειλές του σε ΔΕΚΟ, στην εφορία, σε πάγια έξοδα της εταιρίας και άλλα. Οι ρόλοι στο σύστημα είναι αυτοί του λογιστή/διαχειριστή και του πελάτη/χρήστη.

Ο λογιστής/ εισέρχεται στο περιβάλλον του διαχειριστή της εφαρμογής και εκεί μπορεί να προβεί στις παρακάτω ενέργειες:

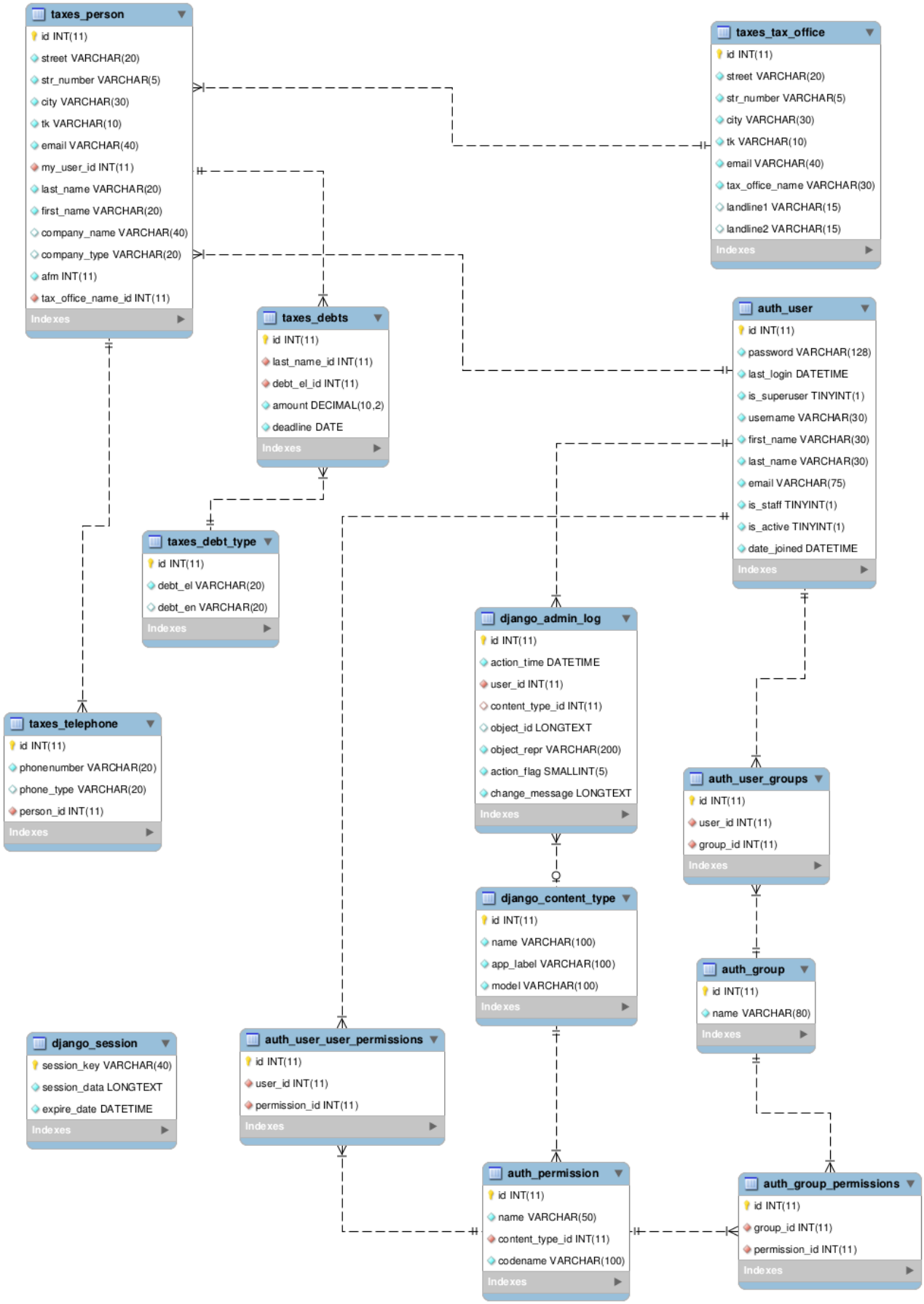
- να δημιουργήσει νέους χρήστες
- να δημιουργεί ομάδες χρηστών
- να εισάγει, να τροποποιεί και να διαγράφει δεδομένα που αφορούν:
    - νέους πελάτες
    - στοιχεία επικοινωνίας του πελάτη
    - στοιχεία επικοινωνίας με τις υπηρεσίες με τις οποίες συναλλάσσεται ο πελάτης
    - στοιχεία που αφορούν τις οφειλές του πελάτη
- να παίρνει σε πινακοποιημένη μορφή τα στοιχεία που τον ενδιαφέρουν και να μπορεί να εφαρμόζει φίλτρα για την οπτικοποίηση και ομαδοποίηση των δεδομένων
- να παρακολουθεί τις πρόσφατες ενέργειες που έχουν πραγματοποιηθεί

Ο πελάτης/χρήστης μπορεί να επισκέπτεται την ιστοσελίδα της εφαρμογής και να εισάγεται σε αυτήν με τα συνθηματικά που έχει λάβει από τον λογιστή του. Εκεί ο πελάτης μπορεί να μεταβεί στις σελίδες όπου μπορεί:

- να δει σε πινακοποιημένη μορφή τα στοιχεία των υπηρεσιών που τον αφορούν
- να δει σε πινακοποιημένη μορφή τα στοιχεία του
- να φιλτράρει ως προς το είδος και την ημερομηνία τις οφειλές του και να τις δει σε πινακοποιημένη μορφή
- να φιλτράρει ανά μήνα τις οφειλές του και να δει το αντίστοιχο διάγραμμα, όπου οι γραμμές αντιστοιχούν στις οφειλές και οι στήλες στα ποσά των οφειλών σε ευρώ.
- να φιλτράρει ανά έτος τις οφειλές του και να δει το αντίστοιχο διάγραμμα, όπου οι γραμμές αντιστοιχούν στις οφειλές και οι στήλες στο ετήσιο άθροισμα των ποσών οφειλών σε ευρώ.

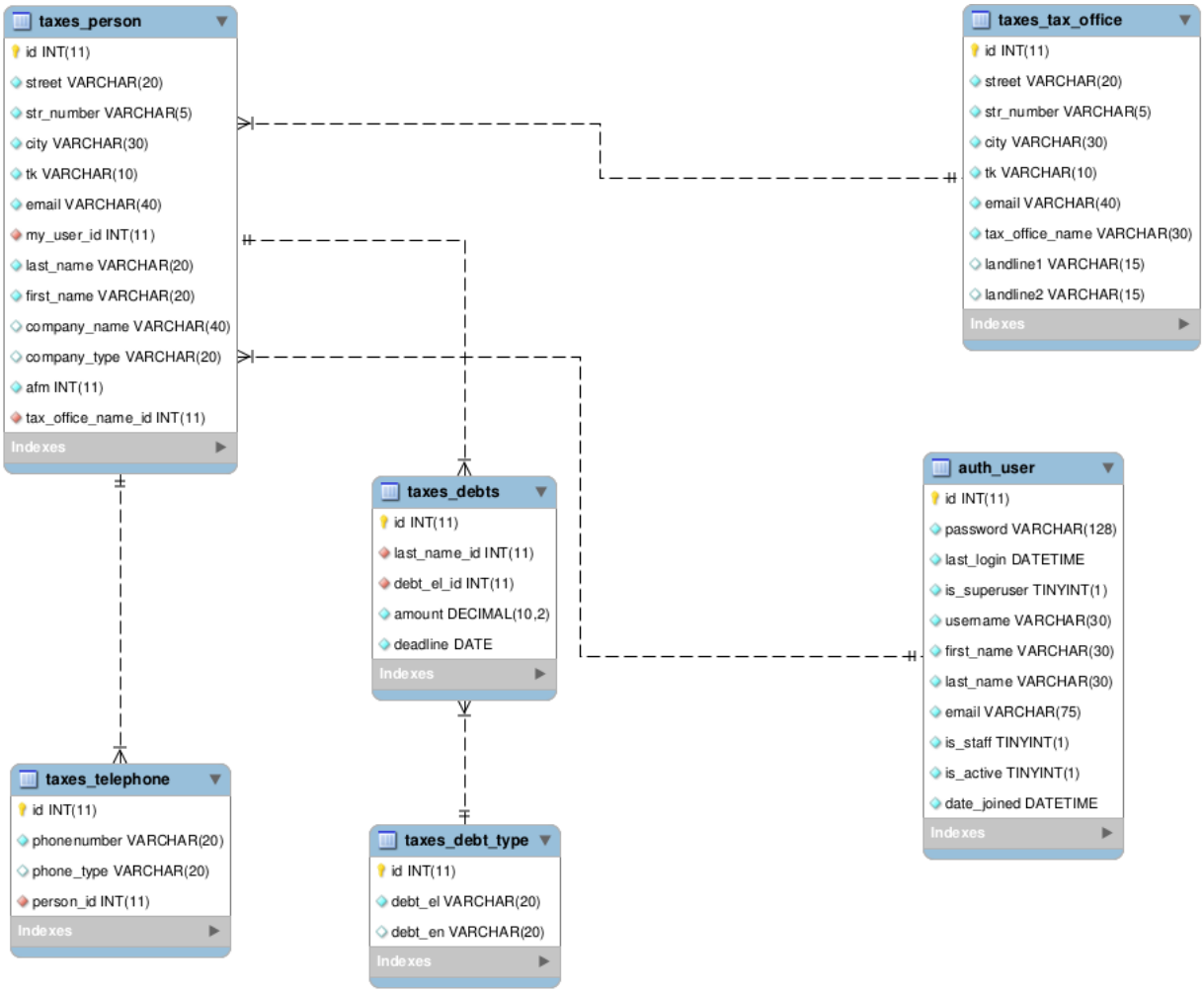
1.2. Περιγραφή της βάσης δεδομένων

Η βάση δεδομένων αποτελείται τόσο από πίνακες για την σελίδα διαχείρισης, για την αυθεντικοποίηση των χρηστών, για τα στοιχεία που κρατούνται για τις οφειλές και προβάλλονται στην εφαρμογή. Στην Εικόνα 1 παρατίθεται η συνολική μορφή της βάσης και οι συσχετίσεις των πινάκων.



Εικόνα 1 - Βάση Δεδομένων της εφαρμογής

Η εφαρμογή αποτελείται από του παρακάτω πίνακες (Εικόνα 2):



Εικόνα 2 - Συσχετίσεις πινάκων της εφαρμογής

Η σχέση των πινάκων person και tax_office είναι 1-προς-πολλά, ένα άτομα μπορεί να έχει μία μόνο εφορία στην οποία ανήκει σε σε μία εφορία μπορούν να ανήκουν πολλά άτομα.

Η σχέση των πινάκων person και telephone είναι 1-προς-πολλά, ένα άτομα μπορεί να έχει πολλά τηλέφωνα, αλλά ένα τηλέφωνο ανήκει αποκλειστικά σε ένα άτομο.

Η σχέση των πινάκων person και debts είναι 1-προς-πολλά, ένα άτομα μπορεί να έχει πολλές οφειλές, αλλά μία οφειλή ανήκει αποκλειστικά σε ένα άτομο.

Η σχέση των πινάκων debts και debt_type είναι 1-προς-πολλά, μία οφειλή μπορεί να ανήκει αποκλειστικά σε μια κατηγορία οφειλών, αλλά πολλές οφειλές μπορούν να έχουν την ίδια κατηγορία.

Η σχέση των πινάκων person και auth_user είναι 1-προς-ένα, ένα άτομα μπορεί να έχει έχει μόνο ένα κωδικό πρόσβασης, και ένας κωδικός πρόσβασης μπορεί να αντιστοιχεί σε ένα μόνο άτομο.

2. Εργαλεία Ανάπτυξης της Εφαρμογής

Για την ανάπτυξη αυτής της εφαρμογής χρησιμοποιήθηκαν κάποια εργαλεία για την ανάπτυξη του κώδικα τόσο στο επίπεδο του server όσο και στο επίπεδο του γραφικού περιβάλλοντος της ιστοσελίδας. Για το στήσιμο της εφαρμογής χρησιμοποιήθηκαν τα παρακάτω:

- Ubuntu 14.04 LTS (Trusty Tahr)
- Sublime Text2
- Python 2.7.5+
- pip, package management system
- Django 1.6.5
- MySql Datebase
- Django-Chartit 0.1
- jQuery 1.11.1
- Highcharts 4.0.4
- HTML
- CSS
- Bootstrap 3
- Bootstrap-datepicker widget
- Git 1.8

2.1. Ubuntu 14.04 LTS (Trusty Tahr)

Το Ubuntu είναι ένα ανοικτού κώδικα, ελεύθερο και δωρεάν λειτουργικό σύστημα βασισμένο στον πυρήνα Linux. Ο στόχος του Ubuntu είναι η παροχή ενός διαρκώς ενημερωμένου, σταθερού λειτουργικού συστήματος για τον μέσο χρήστη, με ενισχυμένη έμφαση στην ευκολία χρήσης και εγκατάστασης. Αποτελεί μια από τις πιο διαδεδομένες διανομές Linux και υποστηρίζεται από μια πολυπληθή παγκόσμια κοινότητα. Αποτελεί μια ασφαλή λύση, διότι δεν πλήττεται από ιούς και προσφέρει αυξημένη ασφάλεια ως προς εξωτερικές επιθέσεις. Το κυριότερο χαρακτηριστικό του είναι ότι διατίθεται δωρεάν το ίδιο καθώς και όλες οι εφαρμογές και τα προγράμματα που είτε είναι εγκατεστημένα σε αυτό, είτε μπορούν να κατέβουν τα επίσημα αποθετήρια με την βοήθεια του Κέντρου Λογισμικού. Διατίθεται τόσο σε έκδοση Server, όσο και σε έκδοση Desktop. [1][2][3]

2.2. Sublime Text

Το 1.2. Sublime Text είναι ένας επεξεργαστής κειμένου που παρέχει στον προγραμματιστή τα απαραίτητα εργαλεία για την ανάπτυξη του κώδικά του. Με την προσθήκη κατάλληλων plugins η επεξεργασία του κώδικα γίνεται ακόμα πιο αποδοτική. Κάποιες από τις λειτουργίες του είναι η ταυτόχρονη επεξεργασία πολλαπλών γραμμών, η ομαδοποίηση κώδικα σε blocks, η αυτόματη συμπλήρωση κώδικα και ο χρωματισμός του σύμφωνα με την γλώσσα προγραμματισμού που έχει προεπιλεγεί, η εύκολη πλοήγηση και εντός των αρχείων και εντός όλου του πρότζεκτ.[4]

2.3. Python

Η Python είναι μια ευρέως χρησιμοποιούμενη, υψηλού επιπέδου, αντικειμενοστραφής γλώσσα προγραμματισμού που δημιουργήθηκε τον Ολλανδό Γκουίντο βαν Ρόσσουμ (Guido van Rossum) το 1990. Δόθηκε ιδιαίτερη έμφαση στην αναγνωσιμότητα του κώδικα από τον προγραμματιστή και στην ευχρηστία της. Το σύστημα τύπων της είναι πλήρως δυναμικό και η διαχείριση μνήμης αυτόματη, η standard υλοποίησή της (CPython) είναι ένας bytecode compiler και interpreter και η βιβλιοθήκη της πολύ πλούσια. Αναπτύσσεται ως ανοιχτό λογισμικό (open source) και η διαχείρισή της γίνεται από τον μη κερδοσκοπικό οργανισμό Python Software Foundation. Ο κώδικας διανέμεται με την άδεια Python Software Foundation License η οποία είναι συμβατή με την GPL. [5][6][7][8]

2.4. Pip

Το pip είναι ένα σύστημα διαχείρισης πακέτων που χρησιμοποιείται για την εγκατάσταση και τη διαχείριση των πακέτων λογισμικού γραμμένα σε Python. Πολλά πακέτα βρίσκονται στο Python Package Index (PyPI). [9]
Ένα σημαντικό πλεονέκτημα του pip είναι η ευκολία διασύνδεσης της γραμμής εντολών του, το οποίο καθιστά την εγκατάσταση και απεγκατάσταση πακέτων λογισμικού Python εφικτή σε μια γραμμή:

```
$ pip install some-package-name
$ pip uninstall some-package-name
```

Το πιο σημαντικό στο pip είναι ότι έχει το χαρακτηριστικό να διαχειρίζεται τους πλήρεις καταλόγους των πακέτων και των αντίστοιχων αριθμών εκδόσεων, και είναι δυνατόν μέσω ενός αρχείου με τα “requirements”. Αυτό επιτρέπει την αποτελεσματική αναδημιουργία μιας ολόκληρης ομάδας των πακέτων σε ένα ξεχωριστό περιβάλλον (π.χ. ένα άλλο υπολογιστή) ή σε ένα εικονικό περιβάλλον. Αυτό μπορεί να επιτευχθεί με ένα κατάλληλα διαμορφωμένο αρχείο requirements.txt και την ακόλουθη εντολή:

```
$ pip install -r requirements.txt
```

2.5. Django

Το Django είναι ένα ελεύθερο, ανοιχτού κώδικα πλαίσιο διαδικτυακών εφαρμογών (web application framework), γραμμένο σε Python. Το framework ουσιαστικά έχει σαν στόχο να μειώσει το φόρτο που σχετίζεται με συνήθεις εργασίες κατά την ανάπτυξη μια ιστοσελίδας και παρέχει βιβλιοθήκες για πρόσβαση στην βάση δεδομένων, πρότυπα πλαισίων και διαχείριση συνεδριών. Ακολουθεί το αρχιτεκτονικό πρότυπο model-view-controller. Συντηρείται από το Django Software Foundation (DSF), έναν ανεξάρτητο, με κερδοσκοπικό οργανισμό. Βασικός του στόχος είναι η ευκολία δημιουργίας σύνθετων ιστοσελίδων, βασισμένες σε κάποια βάση δεδομένων. Το Django δίνει έμφαση στην επαναχρησιμοποίηση των κομματιών του κώδικα, στην γρήγορη ανάπτυξη και στην αρχή να μην επαναλαμβάνεται ο κώδικας. Η γλώσσα προγραμματισμού Python χρησιμοποιείται σε όλο το εύρος της συγγραφής του κώδικα, ακόμα και στον καθορισμό ρυθμίσεων, αρχείων και δημιουργία μοντέλων βάσεων δεδομένων. Επιπλέον το Django διαθέτει ένα προαιρετικό διαχειριστικό περιβάλλον, όπου είναι δυνατή η δημιουργία, ανάγνωση, ανανέωση και διαγραφή δεδομένων, το οποίο δημιουργείται δυναμικά και μπορεί να παραμετροποιηθεί μέσα από τα μοντέλα του διαχειριστή. [10][11]

2.6. MySql

Η MySql είναι ένα από τα πιο διαδεδομένα, ανοιχτού κώδικα συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων. Η mysql είναι πολυνηματική και πολυχρηστική και υποστηρίζει τα τελευταία standards της SQL. Ο πηγαίος κώδικας είναι διαθέσιμος υπό την άδεια GNU General Public License, και διαθέτει και εμπορικές διανομές. Η MySql είναι ιδιαίτερα δημοφιλής για την χρήση της σε διαδικτυακές εφαρμογές και είναι δομικό συστατικό του ευρέως χρησιμοποιούμενου,

ανοιχτού κώδικα πακέτου λογισμικού διαδικτύακών εφαρμογών LAMP. Το LAMP χρησιμοποιείται ως ακρωνύμιο για τα "Linux, Apache, MySQL, Perl/PHP/Python". Γενικά οι εφαρμογές ανοιχτού κώδικα που χρειάζονται μια εξελιγμένη πλατφόρμα διαχείρισης βάσεων δεδομένων, χρησιμοποιούν συχνά την MySql. [13]

Ένας MySQL server αποτελείται από ένα σύνολο βάσεων δεδομένων (databases). Κάθε βάση δεδομένων είναι ένα σύνολο πινάκων (tables) παρόμοιων με αυτών που ενός προγράμματος spreadsheet όπως το Excel. Οι γραμμές του πίνακα ονομάζονται εγγραφές (records) και οι στήλες του πίνακα ονομάζονται πεδία (fields). Δεν υπάρχει ουσιαστικό όριο στην ποσότητα βάσεων που περιέχονται στον server, στην ποσότητα πινάκων που περιέχονται στις βάσεις, ούτε και στην ποσότητα πεδίων και εγγραφών που περιέχονται στους πίνακες. Τα πεδία των πινάκων μπορούν να περιέχουν μόνο ένα συγκεκριμένο τύπο πληροφορίας ο οποίος ορίζεται κατά την δημιουργία του πίνακα. Για παράδειγμα ένα πεδίο που είναι ορισμένο ως ακέραιος αριθμός δεν είναι κατάλληλο για αποθήκευση δεκαδικού αριθμού ή κειμένου. Η δομή αυτή χρησιμοποιείται από όλα τα σύγχρονα συστήματα βάσεων δεδομένων και έχει αποδειχτεί πολύ πρακτική για την αποθήκευση τεράστιων όγκων δομημένης πληροφορίας.[12]

2.7. Django-Chartit

Το Chartit είναι μια εφαρμογή του Django που μπορεί να χρησιμοποιηθεί για την δημιουργία γραφημάτων, χρησιμοποιώντας τα δεδομένα της βάση δεδομένων της διαδικτυακής εφαρμογής. Τα διαγράμματα δημιουργούνται με τη χρήση Highcharts και jQuery JavaScript βιβλιοθήκες. Τα δεδομένα της βάσης δεδομένων μπορούν να παρασταθούν ως απλή γραφήματα γραμμής, γραφήματα στήλης, γραφήματα περιοχής, διαγράμματα διασποράς, και πολλά άλλα. [14]

2.8. jQuery

Το jQuery είναι μια γρήγορη, μικρή, ελεύθερη και ανοιχτού λογισμικού βιβλιοθήκη JavaScript με πολλά πρόσθετα και έχει σχεδιαστεί για να απλοποιεί τις λειτουργίες του client. Διαχειρίζεται εύκολα και απλά διαδικασίες, όπως ο χειρισμός και η διαχείριση εγγράφων HTML, η διαχείριση συμβάντων, η κίνηση εικόνων, η ανάπτυξη κώδικα Ajax. Λειτουργεί σε ένα πλήθος από προγράμματα περιήγησης και είναι ευέλικτο και επεκτάσιμο. [15]

2.9. Highcharts

Το Highcharts είναι μια ελεύθερη και μη εμπορική βιβλιοθήκη γραφημάτων. Είναι γραμμένο σε καθαρή JavaScript και προσφέρει έναν εύκολο τρόπο προσθήκης διαδραστικών γραφήματα για να μια ιστοσελίδα ή μια διαδικτυακή εφαρμογή. Το Highcharts υποστηρίζει γραφήματα σε διάφορες μορφές, όπως ιστογράμματα, κυκλικά διαγράμματα, μπάρες, στήλες κ.λ.π. Δεν χρειάζεται εγκατάσταση, αλλά για να τρέξει αρκεί να συμπεριληφθούν στην διαδικτυακή εφαρμογή τα αρχεία highcharts.js του πυρήνα του Highcharts και το αρχείο Javascript του jQuery. Είναι απλό στην χρήση, φορτώνεται δυναμικά, υποστηρίζει πολλαπλούς άξονες και ετικέτες και δίνει την δυνατότητα εκτύπωσης και εξαγωγής των γραφημάτων.[16]

2.10. HTML

Η HTML είναι το ακρωνύμιο των λέξεων HyperText Markup Language (γλώσσα μορφοποίηση υπερκειμένου) και είναι η βασική γλώσσα δόμηση σελίδων του World Wide Web. Είναι μία γλώσσα προγραμματισμού, που χρησιμοποιείται για να προσδιορίσει ένα τμήμα κειμένου και να το κάνει να εμφανίζεται καλύτερα. Επιτρέπει την ενσωμάτωση ήχου και εικόνων στις web σελίδες. Αρχικά είχε κατασκευασθεί με σκοπό μόνο την μορφοποίηση κειμένου, αλλά κατά την εξέληξή της ενσωμάτωσε σχεδιαστικές τεχνικές κ.α. Η γλώσσα χρησιμοποιεί ένα αριθμό από tags για την μορφοποίηση κειμένου, για την δημιουργία συνδέσμων (links) μετάβασης ανάμεσα των σελίδα, για την εισαγωγή εικόνων, ήχου κ.α. Οι ετικέτες (tags), οι οποίες περικλείονται μέσα σε σύμβολα «μεγαλύτερο από» και «μικρότερο από» (για παράδειγμα <html>), μέσα στο περιεχόμενο της ιστοσελίδας. Οι ετικέτες HTML συνήθως λειτουργούν ανά ζεύγη (για παράδειγμα <h1> και </h1>), με την πρώτη να ονομάζεται ετικέτα έναρξης και τη δεύτερη ετικέτα λήξης (ή σε άλλες περιπτώσεις ετικέτα ανοίγματος και ετικέτα κλεισίματος αντίστοιχα).

Τα στοιχεία της HTML χρησιμοποιούνται για να κτίσουν όλους του ιστότοπους. Η HTML επιτρέπει την ενσωμάτωση εικόνων και άλλων αντικειμένων μέσα στη σελίδα, και μπορεί να χρησιμοποιηθεί για να εμφανίσει διαδραστικές φόρμες. Παρέχει τις μεθόδους δημιουργίας δομημένων εγγράφων (δηλαδή εγγράφων που αποτελούνται από το περιεχόμενο που μεταφέρουν και από τον κώδικα μορφοποίησης του περιεχομένου) καθορίζοντας δομικά σημαντικά στοιχεία για το κείμενο, όπως κεφαλίδες, παραγράφους, λίστες, συνδέσμους, παραθέσεις και άλλα. Μπορούν επίσης να ενσωματώνονται σενάρια εντολών σε γλώσσες όπως η JavaScript, τα οποία επηρεάζουν τη συμπεριφορά των ιστοσελίδων HTML. [17][18]

2.11. CSS

Η CSS (Cascading Style Sheets-Διαδοχικά Φύλλα Στυλ) ή (αλληλουχία φύλλων στύλ) είναι μια γλώσσα υπολογιστή που ανήκει στην κατηγορία των γλωσσών φύλλων στυλ που χρησιμοποιείται για τον έλεγχο της εμφάνισης ενός εγγράφου που έχει γραφτεί με μια γλώσσα σήμανσης.

Χρησιμοποιείται δηλαδή για τον έλεγχο της εμφάνισης ενός εγγράφου που γράφτηκε στις γλώσσες HTML και XHTML, δηλαδή για τον έλεγχο της εμφάνισης μιας ιστοσελίδας και γενικότερα ενός ιστοτόπου. Η CSS είναι μια γλώσσα υπολογιστή προορισμένη να αναπτύσσει στυλιστικά μια ιστοσελίδα δηλαδή να διαμορφώνει περισσότερα χαρακτηριστικά, χρώμματα, στοίχιση και δίνει περισσότερες δυνατότητες σε σχέση με την html. Παρέχει πολύ μεγάλη ευελιξία και καθιστά εφικτές μορφοποιήσεις που θα ήταν αδύνατες ή πολύ δύσκολες αλλιώς. Επιτρέπει την ευκολότερη συντήρηση των ιστοσελίδων και καθορίζει την τελική εμφάνιση της ιστοσελίδας μέσα από μόνο ένα αρχείο. [19][20]

2.12. Bootstrap

Το Bootstrap είναι μια συλλογή εργαλείων ανοιχτού κώδικα (Ελεύθερο λογισμικό) για τη δημιουργία ιστοσελίδων και διαδικτυακών εφαρμογών. Περιέχει HTML και CSS για τις μορφές τυπογραφίας, κουμπιά πλοήγησης και άλλων στοιχείων του περιβάλλοντος, καθώς και προαιρετικές επεκτάσεις JavaScript. Είναι συμβατό με όλους τους φυλλομετρητές και υποστηρίζει ανταποκρίσιμο σχεδιασμό (responsive design), μπορεί δηλαδή να προσαρμόσει την διάταξη των ιστοσελίδων δυναμικά, λαμβάνοντας υπόψη τα χαρακτηριστικά της συσκευής που χρησιμοποιείται (PC, tablet, κινητό τηλέφωνο). Περιέχει CSS αρχεία, που καθορίζουν τους βασικούς ορισμούς στυλ για όλα τα βασικά στοιχεία HTML. Αυτά καθορίζουν την ομοιογενή εμφάνιση για τους πίνακες, την μορφοποίηση του κειμένου, καθώς και τα στοιχεία μιας φόρμας. Εκτός από τα βασικά HTML στοιχεία, το Bootstrap περιέχει και άλλα στοιχεία που χρησιμοποιούνται συχνά, όπως κουμπιά με προηγμένα χαρακτηριστικά (πχ. ομαδοποίηση κουμπιών ή dropdown επιλογή, οριζόντιες και κάθετες καρτέλες, πλοήγηση, σελιδοποίηση, κλπ.), ετικέτες, προηγμένες τυπογραφικές δυνατότητες, εικονίδια, προειδοποιητικά μηνύματα και μια γραμμή προόδου. Επιπλέον περιέχει αρχεία Javascript με μορφή jQuery. Αυτά περιέχουν επιπλέον στοιχεία, όπως αυτόματη συμπλήρωση φορμών εισαγωγής, αναδυόμενα παράθυρα και άλλα. [22]

2.13. Bootstrap-datepicker widget

Το Bootstrap-datepicker παρέχει ένα ευέλικτο widget επιλογής ημερομηνίας. Δεν γίνεται κάποια εγκατάστασή του, αλλά προστίθεται το Javascript αρχείο στον φάκελο με τα αντίστοιχα αρχεία του Bootstrap.

2.14. Git

Το git είναι ένα ανοιχτό και ελεύθερου κώδικα κατανεμημένο σύστημα ελέγχου έκδοσης (distributed revision control) και διαχείρισης πηγαίου κώδικα (source code management – SCM), που δίνει έμφαση στην ταχύτητα, την ακεραιότητα των δεδομένων και στις κατανεμημένες, μη γραμμικές ροές εργασίας. Το Git σχεδιάστηκε και αναπτύχθηκε αρχικά από τον Λίνους Τόρβαλντς για τη ανάπτυξη του πυρήνα Linux το 2005 και έχει γίνει από τότε το πιο πλατιά διαδεδομένο σύστημα ελέγχου εκδόσεων για ανάπτυξη λογισμικού. Ουσιαστικά το git επιτρέπει σε μια πολυπληθή ομάδα προγραμματιστών να αναπτύσσουν παράλληλα τον κώδικά τους. Κατά την διαδικασία υποβολής του κώδικα και συγχώνευσής του με τον κεντρικό κλάδο του git ανιχνεύονται οι συγκρούσεις σε κομμάτια του κώδικα που έχουν τροποποιηθεί από περισσότερα του ενός άτομα και με εποπτικά εργαλεία υποβοηθάται η επίλυση των συγκρούσεων στον κώδικα. Φυσικά το git καθιστά και έναν αποδοτικό τρόπο, ώστε να καταγράφονται οι αλλαγές στον κώδικα και να μπορεί ο προγραμματιστής να ανατρέξει εύκολα σε προηγούμενες εκδόσεις του κώδικά του, και να παρατηρήσει με τα εποπτικά εργαλεία που παρέχονται τις τροποποιήσεις που έχουν γίνει σε καθένα αρχείο. [24]

3. Django framework

3.1. Βασικές αρχές Django

Μία θεμελιώδης αρχή του του Django είναι η χαλαρή σύζευξη και σφιχτή συνοχή. Τα διάφορα στρώματα του πλαισίου δεν θα πρέπει να "γνωρίζουν" το καθένα για το άλλο, εκτός αν είναι απολύτως απαραίτητο. Για παράδειγμα, το σύστημα προτύπων (templates) δεν γνωρίζει τίποτα για τα αιτήματα Web, το επίπεδο της βάσης δεδομένων δεν γνωρίζει τίποτα σχετικά με την προβολή των δεδομένων και το σύστημα προβολής (view) δεν ενδιαφέρεται για το ποια πρότυπα χρησιμοποιεί ο προγραμματιστής στην εφαρμογή. Επιπλέον στις εφαρμογές Django θα πρέπει να χρησιμοποιείται το δυνατόν λιγότερος κώδικας και να γίνεται πλήρης χρήση των πλεονεκτημάτων της Python και όλων των δυναμικών δυνατοτήτων της. Στόχος είναι οι επίπονες διαδικασίες του διαδικτυακού προγραμματισμού να υλοποιούνται εξαιρετικά γρήγορα.

Μια ακόμα από τις βασικότερες αρχές του Django είναι το DRY(Don't repat yourself), δηλαδή “Μην επαναλαμβάνεις τον εαυτό σου”. Η αλληλοεπικάλυψη κώδικα μπορεί να οδηγήσει σε προβλήματα συντήρησης, κακή προσαρμογή και σε λογικές αντιθέσεις. Αυτό μπορεί να έχει επιπτώσεις παντού στο σύστημα, στην αρχιτεκτονική, τα προαπαιτούμενα, τον κώδικα, την υλοποίηση ή στην τεκμηρίωση. Οι επιπτώσεις αυτές μπορούν να οδηγήσουν σε σύγχυση και ολόκληρο το σύστημα να καταρρεύσει. Οπότε σύμφωνα με την συγκεκριμένη αρχή, κάθε κομμάτι της γνώσης πρέπει να έχει μια ενιαία, σαφή, έγκυρη παράσταση μέσα σε ένα σύστημα. Είναι δυνατόν να υπάρχει μια μηχανική ή λεκτική επανάληψη, εφόσον είναι γνωστό κατά την οργάνωση του συστήματος. Όταν υπάρχουν διαφορετικά επίπεδα αφαίρεσης που εμπλέκονται, θα πρέπει να χρησιμοποιείται ένα συνεπές σύστημα επίλυσης των συγκρούσεων. [10]

3.2. Πλεονεκτήματα του Django

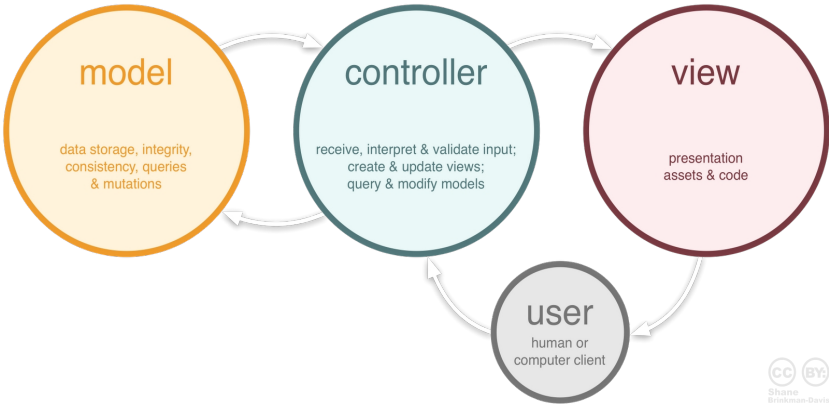
Το Django είναι ένα υψηλού επιπέδου διαδικτυακό πλαίσιο Python που ενθαρρύνει την ταχεία ανάπτυξη και τον καθαρό, ρεαλιστικό σχεδιασμό. Αναπτύχθηκε για τις ανάγκες των ειδήσεων στο διαδίκτυο και σχεδιάστηκε για να χειριστεί δύο προκλήσεις: τις εντατικές προθεσμίες ενός κέντρου ειδήσεων και τις αυστηρές απαιτήσεις των έμπειρων προγραμματιστών Web που το έγραψαν. Το Django επικεντρώνεται στην αυτοματοποίηση και προσπαθεί να τηρεί την αρχή DRY το δυνατόν περισσότερο. Επιτρέπει να καθοριστούν τα μοντέλα δεδομένων εξολοκλήρου σε Python, και οι σχέσεις των δεδομένων καταγράφονται ως αντικείμενα. Επιπλέον δημιουργεί αυτόματα την

γραφική διεπαφή διαχειριστή και έτσι μπορεί αυτόματα να είναι διαθέσιμο ένα εύχρηστο περιβάλλον προσθήκης και ανανέωσης περιεχομένου. Τα URL έχουν πρακτικό και κομψό σχεδιασμό και δεν έχουν περιορισμούς από το πλαίσιο,μπορούν να είναι όσο ευέλικτα χρειάζεται. Το σύστημα προτύπων (templates) επιτρέπει την δυναμική,επεκτάσιμη και φιλική προς την σχεδίαση ανάπτυξη του κώδικα και διαχωρίζει τον σχεδιασμό από το περιεχόμενο και από το κώδικα της εφαρμογής. Τέλος παρέχει πλήρη υποστήριξη πολυγλωσσικών εφαρμογών.

3.3. Πρότυπο σχεδίασης Model–view–controller

Το Model–view–controller (MVC) αποτελεί ένα αρχιτεκτονικό πρότυπο σχεδίασης λογισμικού το οποίο χρησιμοποιείται κυρίως για την δημιουργία γραφικών διεπαφών χρήστη (Graphical User Interfafce-GUI). Ένα πρότυπο σχεδιασμού είναι μια δομή κώδικα που επιτρέπει σε συνήθη πλαίσια (frameworks) να επαναχρησιμοποίηθούν εύκολα. Το πρότυπο σχέδιο αποτελεί ένα σκελετό ή το πλαίσιο στο οποίο κατασκευάζεται η εφαρμογή. Η εφαρμογή αποτελείται από τρία μέρη, το μοντέλο (model), την προβολή (view) και τον ελεγκτή (controller).
Το μοντέλο (model) καθορίζει τον τύπο των δεδομένων της εφαρμογής. Είναι αυτό που κατασκευάζεται πρώτα, καθορίζει πώς θα αποθηκευθούν τα δεδομένα και δεν πρέπει να εξαρτάται από την προβολή (view) ή τον ελεγκτή (controller).
Η προβολή (view) ζητάει τα δεδομένα από το μοντέλο και παράγει αποτελέσματα από αυτό. Επίσης διαχειρίζεται την γραφική και την αλφαριθμητική έξοδο στο μοντέλο. Μπορεί να δημιουργήσει html έξοδο ή να παράγει αρχεία της μορφής pdf,csv και άλλα.
Ο ελεγκτής (controller) παρέχει τον έλεγχο των αλλαγών στην παρουσίαση της προβολής ή στην ανανέωση των δεδομένων των μοντέλων. Καθιστά εφικτή την πλοήγηση στα δεδομένα και πολλές φορές ακόμα και την τροποποίησή τους.

Μια γραφική απεικόνιση του προτύπου παρουσιάζεται στην Εικόνα 3.



Εικόνα 3 - Αρχιτεκτονικό πρότυπο σχεδίασης Model–view–controller (MVC)

Παρόλο που το Django ακολουθεί το πρότυπο σχεδίασης MVC, το εφαρμόζει λίγο διαφορετικά από τον κλασσικό του ορισμό. Στην ορολογία του Django αυτό περιγράφεται ως Models-Templates-Views (MTV), όπου τα templates καθορίζουν το πώς παρουσιάζονται στον χρήστη τα δεδομένα. Ένα μοντέλο περιγράφει τι είδους δεδομένα αποθηκεύονται στο διακομιστή και ο ελεγκτής καθορίζει τον μηχανισμό που παρέχεται από το πλαίσιο και ο κώδικας μεταφέρει την εισερχόμενη κλήση σε μια κατάλληλη προβολή, όπως και το μοντέλο του MVC. Αντιθέτως η προβολή περιγράφει ποια δεδομένα θα επιστραφούν στους χρήστες, ενώ στο πρότυπο MVC η προβολή καθορίζει τον τρόπο παρουσίασης των δεδομένων. [25][26][27]

4. Εγκαταστάσεις και ρυθμίσεις

4.1. Εγκατάσταση του Django

Για να γίνει η εγκατάσταση του Dajngo στο Λειτουργικό Σύστημα Ubuntu 14.04 θα πρέπει να εγκατασταθούν πιο πριν η γλώσσα Python και το σύστημα διαχείρισης πακέτων Python, pip. Με την βοήθεια του pip εγκαθίσταται στην συνέχεια και το Django, όπως αυτό περιγράφεται παρακάτω:

```
$ apt-get install python
$ apt-get install pip
$ pip install django
```

4.2. Apache webserver σε mod_wsgi

Η εγκατάσταση του Apache είναι απαραίτητη όταν η εφαρμογή ανέβει σε development mode. Για σκοπούς ανάπτυξης του κώδικα το Django διαθέτει ένα ελαφρύ web server, ο οποίος μπορεί να χρησιμοποιηθεί για να τεσταριστεί της εφαρμογής.
Για το production site, δηλαδή για την εφαρμογή που θα σερβίρεται μέσω ίντερνετ σε όλους, είναι απαραίτητη η εγκατάσταση του Apache με mod_wsgi. Το mod_wsgi μπορεί να έχει δύο καταστάσεις, την ενσωματωμένη και την deamon. Συνήθως χρησιμοποιείται το daemon mode, το οποίο γεννά μια ανεξάρτητη διαδικασία δαίμονα, η οποία διαχειρίζεται τα αιτήματα. Η διαδικασία δαίμονα μπορεί να τρέχει ως διαφορετικός χρήστης απ' ότι ο Web server, πιθανών αυξάνοντας την ασφάλεια και ο δαίμονας αυτός μπορεί να επανεκκινήσει, χωρίς να επανεκκινήσει ολόκληρος ο Apache Web server, αφήνοντας να ανανεωθεί η βάση του κώδικα πιο ομαλά.

5. Ανάπτυξη της Εφαρμογής

5.1. Σκελετός της εφαρμογής

Το πρώτο βήμα για να στηθεί ο σκελετός της εφαρμογής είναι να τρέξει η παρακάτω εντολή, με την οποία παράγει αυτοματοποιημένα τον αρχικό κώδικα πάνω στον οποίο θα βασιστεί η ανάπτυξη.

```
$ mkdir tax_office && cd tax_office
$ django-admin.py startproject tax_office
```

Με αυτό τον τρόπο στον φάκελο tax_office που δημιουργήσαμε με την εντολή mkdir, δημιουργείται αυτόματα ένας υποφάκελος tax_office me την εξής δομή:

```
tax_office/
    |__ manage.py
    |__ tax_office/
        |__ __init__.py
        |__ settings.py
        |__ urls.py
        |__ wsgi.py
```

Ο εξωτερικός φάκελος tax_office είναι ο φάκελος που περιέχει την εφαρμογή. Το όνομά του δεν είναι σημαντικό για το Django και μπορεί να αλλάξει ανά πάσα στιγμή.
Το αρχείο manage.py είναι ένα αρχείο που μπορεί να τρέξει στην γραμμή εντολών και επιτρέπει στον προγραμματιστή να αλληλεπιδράσει με την εφαρμογή με διάφορους τρόπους.
Ο εσωτερικός φάκελος tax_office είναι ο φάκελος που βρίσκεται ο πραγματικός κώδικας Python της εφαρμογής. Αυτό το όνομα είναι το όνομα του πακέτου Python που θα χρησιμοποιηθεί σε ολόκληρο το εύρος της εφαρμογής.
Το αρχείο tax_office/__init.py__ είναι το αρχείο που δηλώνει στην Python ποιον φάκελο θα θεωρήσει ως αυτόν που περιέχει τα πακέτα της.
Το αρχείο tax_office/setting.py είναι το αρχείο που δηλώνει τις ρυθμίσεις της εφαρμογής. Εδώ ορίζεται ο web server που θα χρησιμοποιηθεί, η βάση δεδομένων, η γλώσσα του συστήματος, η ζώνη της ώρας, οι εξωτερικές εφαρμογές που θα χρησιμοποιηθούν από το σύστημα και άλλα.
Το αρχείο tax_office/url.py είναι το αρχείο που καθορίζονται τα url της εφαρμογής και λειτουργεί

σαν πίνακα περιεχομένων για τις ιστοσελίδες που αναπτύσσονται σε Django.
Το αρχείο tax_office/wsgi.py είναι ένα αρχείο το οποίο λειτουργεί σαν είσοδος για τους web servers που είναι συμβατοί με το WSGI.

5.2. Δημιουργία των βάσεων εφαρμογών

Αρχικά συνδεόμαστε σαν υπερχρήστης user και δημιουργούμε έναν νέο χρήστη accountant, όπως φαίνεται παρακάτω. Δημιουργούμε την βάση δεδομένων που θα αποθηκευτούν τα δεδομένα μας, και παραχωρούμε όλα δικαιώματα, εγγραφής, τροποποίησης, διαγραφής αυτής της βάσης στο νέο χρήστη. Τα στοιχεία του χρήστη, το όνομα και τον κωδικό σύνδεσης τα καταγράφουμε στο αρχείο settings.py, ώστε να μπορεί η εφαρμογή να συνδεθεί με την βάση.

```
$ mysql -uroot -p

> CREATE USER 'accountant'@'localhost' IDENTIFIED BY 'mypass';
> CREATE DATABASE tax_office;
> GRANT ALL PRIVILEGES ON *.* TO 'accountant'@'localhost'  WITH
GRANT OPTION;
> FLUSH PRIVILEGES;
```

Στο αρχείο των ρυθμίσεων του συστήματος settings.py παρατηρούμε ότι στην κατηγορία των εγκατεστημένων εφαρμογών INSTALLED_APPS, υπάρχουν εφαρμογές που ενεργοποιούνται από το Django. Αυτές οι εφαρμογές μπορούν να χρησιμοποιηθούν σε πολλαπλές εφαρμογές και τα πακέτα του Django μπορούν να τα παρέχουν σε πολλές εφαρμογές. Οι προεπιλεγμένες εφαρμογές που περιέχονται στο INSTALLED_APPS είναι οι παρακάτω:

- django.contrib.admin – Η σελίδα του διαχειριστή
- django.contrib.auth – Το σύστημα αυθεντικοποίησης της εφαρμογής
- django.contrib.contenttypes – Το πλαίσιο τύπων περιεχομένου
- django.contrib.sessions – Το πλαίσιο
- django.contrib.messages – Το πλαίσιο μηνυμάτων
- django.contrib.staticfiles – Το πλαίσιο διαχείρισης στατικών αρχείων

Κάποιες από αυτές τις εφαρμογές χρειάζονται τουλάχιστον έναν πίνακα στην βάση δεδομένων για να αποθηκεύσουν τα δεδομένα τους. Για να δημιουργηθούν οι απαραίτητες βάσεις τρέχουμε την παρακάτω εντολή και η παραγωγή των πινάκων γίνεται αυτόματα.

```
$ python manage.py syncdb
```

```
Output:
Creating tables ...
Creating table django_admin_log
Creating table auth_permission
Creating table auth_group_permissions
Creating table auth_group
Creating table auth_user_groups
Creating table auth_user_user_permissions
Creating table auth_user
Creating table django_content_type
Creating table django_session

You just installed Django's auth system, which means you don't have any
superusers defined.
Would you like to create one now? (yes/no): yes
Username (leave blank to use 'georgia'): accountant
Email address: accountant@mtax.gr
Password:
Password (again):
Superuser created successfully.
Installing custom SQL ...
Installing indexes ...
Installed 0 object(s) from 0 fixture(s)
```

Η εντολή syncdb κοιτάει στις ρυθμίσεις τα INSTALLED_APPS και δημιουργεί τους απαραίτητους πίνακες σύμφωνα με τις ρυθμίσεις στο αρχείο tax_office/settings.py. Πιο πάνω βλέπουμε τα μηνύματα που παράγονται για κάθε πίνακα που παράγεται. Μόλις παραχθούν οι πίνακες στην γραμμή εντολών εμφανίζεται το ερώτημα αν θέλουμε να δημιουργήσουμε υπερχρήστη (superuser) για το σύστημα αυθεντικοποίσης. Ζητείται όνομα χρήστη και συνθηματικό για να δημιουργηθεί ο υπερχρήστης.

5.3. Δημιουργία μοντέλων

Μέτα την ολοκλήρωση της εγκατάστασης του βασικού κορμού της εφαρμογής μπορούμε να προχωρήσουμε στην δημιουργία της ίδιας της εφαρμογής.

Κάθε εφαρμογή στο Django αποτελείται από ένα πακέτο Python που ακολουθεί κάποιες συγκεκριμένες συμβάσεις. Το Django έρχεται με ένα βοηθητικό πρόγραμμα που δημιουργεί αυτόματα τη δομή του βασικού καταλόγου της εφαρμογής, οπότε ο προγραμματιστής επικεντρώνεται στην συγγραφή κώδικα και όχι στην δημιουργία φακέλων.

Για την δημιουργία της εφαρμογής πρέπει να σιγουρευτούμε ότι βρισκόμαστε στον φάκελο όπου βρίσκεται το αρχείο manage.py και να δώσουμε την συγκεκριμένη εντολή:

```
$ python manage.py startapp taxes
```

Παρατηρούμε ότι παράγεται αυτόματα ο φάκελος taxes ο οποίος έχει την παρακάτω δομή:

```
taxes/
  |__ __init__.py
  |__ admin.py
  |__ models.py
  |__ tests.py
  |__ views.py
  |__ urls.py
```

Ο κατάλογος αυτός θα φιλοξενήσει την εφαρμογή taxes. Το πρώτο βήμα για το στήσιμο της βάσης είναι να καθοριστούν τα μοντέλα, δηλαδή η δομή την βάσης και τα αντίστοιχα μεταδεδομένα.
Ο κώδικας είναι πολύ καθαρός, και απλός. Αναλυτικά ο κώδικας που περιέχει τις κλάσεις, taxes/models.py, παρατίθεται στο Παράρτημα Α. Κάθε μοντέλο αντιπροσωπεύεται από μία κλάση, που αποτελεί υποκλάση της django.db.models.Model. Κάθε μοντέλο έχει έναν αριθμό από μεταβλητές κλάσης, κάθε μία από τις οποίες αντιπροσωπεύει ένα πεδίο της βάσης στο μοντέλο. Κάθε πεδίο παριστάνεται από ένα στιγμιότυπο της κλάσης Fields, πχ CharField για πεδία από χαρακτήρες και DateTimeField για ημερομηνίες. Αυτό λέει στο Django το είδος του κάθε πεδίου που μπορεί να αποθηκευθεί. Το όνομα κάθε στιγμιότυπου Field (πχ last_name ή address), είναι το όνομα του πεδίου σε μια μορφή φιλική προς την μηχανή. Αυτή είναι η τιμή που θα χρησιμοποιήσουμε στον κώδικα Python και η βάση θα το χρησιμοποιήσει σαν το όνομα της στήλης του πίνακα που θα δημιουργηθεί. Κάποια πεδία έχουν και κάποια προαπαιτούμενα ορίσματα, όπως το CharField που χρειάζεται ένα μέγιστο αριθμό χαρακτήρων (max_length).
Επιπλέον μπορούν να οριστούν ξένα κλειδιά (ForeignKey). Αυτά ορίζουν ότι κάθε μια κλάση σχετίζεται με ακριβώς μία άλλη. Το Django υποστηρίζει όλες τις συνήθεις σχέσεις των βάσεων δεδομένων: πολλά-προς-ένα, πολλά-προς-πολλά και ένα-προς-ένα.

5.4. Ενεργοποίηση μοντέλων

Αυτό το μικρό κομμάτι κώδικα που περιέχεται στο αρχείο taxes/models.py, παράγει μεγάλη ποσότητα πληροφορίας. Δημιουργεί το σχήμα της βάσης και επιπλέον δημιουγεί ένα API προσβάσιμο από την βάση για την πρόσβαση των αντικειμένων των κλάσεων.

Η εφαρμογή 'taxes' θα πρέπει σε αυτό το σημείο να προστεθεί στο αρχείο settings.py, στα INSTALLED_APPS, ώστε να γνωρίζει το Django ότι πρέχει να συμπεριλάβει και την εφαρμογή taxes.

Για να δημιουργηθούν οι πίνακες της εφαρμογής μας στην βάση δεδομένων τρέχουμε την παρακάτω εντολή:

```
$ python manage.py sql polls
```

Το εξαγόμενο που παίρνουμε στην γραμμή εντολών είναι το παρακάτω:

```
BEGIN;
CREATE TABLE `taxes_tax_office` (
    `id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
    `street` varchar(20) NOT NULL,
    `str_number` varchar(5) NOT NULL,
    `city` varchar(30) NOT NULL,
    `tk` varchar(10) NOT NULL,
    `email` varchar(40) NOT NULL,
    `tax_office_name` varchar(30) NOT NULL,
    `landline1` varchar(15),
    `landline2` varchar(15)
);
CREATE TABLE `taxes_person` (
    `id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
    `street` varchar(20) NOT NULL,
    `str_number` varchar(5) NOT NULL,
    `city` varchar(30) NOT NULL,
    `tk` varchar(10) NOT NULL,
    `email` varchar(40) NOT NULL,
    `my_user_id` integer NOT NULL UNIQUE,
    `last_name` varchar(20) NOT NULL,
    `first_name` varchar(20) NOT NULL,
    `company_name` varchar(40),
    `company_type` varchar(20),
    `afm` integer NOT NULL,
    `tax_office_name_id` integer NOT NULL
);
ALTER TABLE `taxes_person` ADD CONSTRAINT
`tax_office_name_id_refs_id_c15fc79a` FOREIGN KEY (`tax_office_name_id`)
REFERENCES `taxes_tax_office` (`id`);
ALTER TABLE `taxes_person` ADD CONSTRAINT `my_user_id_refs_id_34c6da73`
FOREIGN KEY (`my_user_id`) REFERENCES `auth_user` (`id`);
CREATE TABLE `taxes_telephone` (
    `id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
    `phonenumber` varchar(20) NOT NULL,
    `phone_type` varchar(20),
    `person_id` integer NOT NULL
);
ALTER TABLE `taxes_telephone` ADD CONSTRAINT `person_id_refs_id_e8b4768d`
FOREIGN KEY (`person_id`) REFERENCES `taxes_person` (`id`);
CREATE TABLE `taxes_debt_type` (
    `id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
    `debt_el` varchar(20) NOT NULL,
    `debt_en` varchar(20)
);
CREATE TABLE `taxes_debts` (
    `id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
    `last_name_id` integer NOT NULL,
    `debt_el_id` integer NOT NULL,
    `amount` numeric(10, 2) NOT NULL,
    `deadline` date NOT NULL
);
ALTER TABLE `taxes_debts` ADD CONSTRAINT `last_name_id_refs_id_e6ca8532`
FOREIGN KEY (`last_name_id`) REFERENCES `taxes_person` (`id`);
ALTER TABLE `taxes_debts` ADD CONSTRAINT `debt_el_id_refs_id_7636311f` FOREIGN
KEY (`debt_el_id`) REFERENCES `taxes_debt_type` (`id`);


COMMIT;
```

Παρατηρούμε τα εξής:
Επειδή χρησιμοποιούμε mysql το εξαγόμενο έχει την μορφή εντολών mysql. Τα ονόματα των πινάκων που δημιουργούνται, σχηματίζονται από τον συνδυασμό του ονόματος της εφαρμογής taxes και το όνομα του μοντέλου με μικρά γράμματα. Τα πρωτεύοντα κλειδιά (IDs) προστίθενται αυτόματα. Η σύμβαση που κάνει το Django είναι να προσθέτει το "_id" στο όνομα του πεδίου του ξένου κλειδιού. Φυσικά οι συμβάσεις αυτές μπορούν να τροποποιηθούν. Το ξένο κλειδί καθορίζεται εκ των υστέρων με την εντολή ALTER TABLE και η σχέση καθορίζεται από μια δήλωση τύπου REFERENCES.
Αυτές οι εντολές SQL δεν τρέχουν πραγματικά στην βάση δεδομένων. Απλά τυπώνονται στην οθόνη για να μπορούμε να δούμε τι SQL πιστεύει το Django ότι είναι αναγκαία. Το Django προσφέρει έναν ευκολότερο τρόπο να καταχωρεί SQL στην βάση, όπως θα δούμε παρακάτω.

Για να δημιουργήσουμε τους πίνακες αυτών των μοντέλων στην βάση τρέχουμε την παρακάτω εντολή:

```
$ python manage.py syncdb

Creating tables ...
Creating table taxes_tax_office
Creating table taxes_person
Creating table taxes_telephone
Creating table taxes_debt_type
Creating table taxes_debts

Installing custom SQL ...
Installing indexes ...
Installed 0 object(s) from 0 fixture(s)
```

Η εντολή syncdb τρέχει τον κώδικα SQL στην βάση μας για όλες τις εφαρμογές, που βρίσκονται στις INSTALLED_APPS και που δεν υπάρχουν ήδη στην βάση. Έτσι δημιουργούνται οι πίνακες, τα αρχικά δεδομένα και οι δείκτες για όσες εφαρμογές έχουν προστεθεί στην εφαρμογή από την τελευταία φορά που είχε τρέξει η εντολή. Μπορούμε να την τρέξουμε ανά πάσα στιγμή και αυτή απλά δημιουργεί τους πίνακες που δεν υπάρχουν ήδη.

5.5. Δημιουργία προβολών (Views)

Η προβολή είναι ένας τύπος ιστοσελίδας στην Django εφαρμογή που ακολουθεί ένα συγκεκριμένο πρότυπο και ικανοποιεί μια συγκεκριμένη συνάρτηση. Ο κώδικας για τις προβολές βρίσκεται στο αρχείο taxes/views.py, που παραθέτουμε αναλυτικά στο Παράρτημα Α.

5.5.1. HTTP κλήσεις και αποκρίσεις

Το Django χρησιμοποιεί αντικείμενα request (κλήσης) και response (απόκρισης) για να περάσει αυτές τις καταστάσεις σε όλο το σύστημα. Όταν ζητείται μια σελίδα το Django δημιουργεί ένα αντικείμενο HttpRequest, το οποίο περιέχει μεταδεδομένα για αυτή την κλήση. Τότε το Django φορτώνει την κατάλληλη προβολή, περνώντας το HttpRequest ως πρώτο όρισμα στην προβολή της συνάρτησης. Κάθε προβολή είναι υπεύθυνη να επιστρέψει ένα HttpResponse αντικείμενο. Το HttpRequest περιέχει επιπλέον όλα τα http get και post χαρακτηριστικά και παραμέτρους.

```
from django.http import HttpResponse

def home_page(request):
    template = loader.get_template('taxes/first_page.html')
    context=RequestContext(request)
    return HttpResponse(template.render(context))

def debt_office_list(request):
    a=request.user.id
    latest_debt_office_list = debts.objects.filter(last_name=a)
    template = loader.get_template('taxes/debt_office.html')
    context = RequestContext(request, {
        'latest_debt_office_list': latest_debt_office_list,
    })
    return HttpResponse(template.render(context))
```

Όπως βλέπουμε στο παραπάνω παράδειγμα δημιουργούμε μια συνάρτηση που στέλνει ένα αντικείμενο HttpResponse. Εισάγουμε σε αυτό περιεχόμενο της σελίδας, όπως ένα αλφαριθμητικό και επιστρέφει μια HttpResponse απόκριση ή μια εξαίρεση όπως την Http404. Η προβολή μπορεί να διαβάσει εγγραφές από την βάση, και χρησιμοποιώντας το σύστημα προτύπων του Django και να παράγει εκτός από ένα αποτέλεσμα στο φυλλομέτρητη και αρχεία τύπου pdf, xml ή zip.

Φυσικά για να καλέσουμε την προβολή αυτή θα πρέπει να δημιουργήσουμε μια σύνδεση με ένα URL, οπότε χρειαζόμαστε ένα URLconfig στον κατάλογο taxes με το όνομα urls.py. Τώρα ο κατάλογος της εφαρμογής μας θα πρέπει να έχει την εξής δομή:

```
taxes/
|__ __init__.py
|__ admin.py
|__ models.py
|__ tests.py
|__ urls.py
|__ views.py
```

Το taxes/urls.py θα έχει την παρακάτω μορφή:

```
from django.conf.urls import patterns, url

from taxes import views

urlpatterns = patterns('',
    url(r'^$', views.home_page, name='home_page'),
    url(r'^debt_list', views.doi_list, name='debt_list'),

)
```

Το επόμενο βήμα είναι να μεταβούμε και στο αρχικό, εξωτερικό αρχείο urls.py και να συμπεριλάβουμε εκεί όλα τα url που ορίσαμε στην εφαρμογή μας taxes. Αυτό γίνεται προσθέτοντας ένα include() στα urlpaterns, προσθέτοντας δηλαδή το παρακάτω στο αρχείο urls.py, που παραθέτουμε αναλυτικά στο Παράρτημα Α:

```
urlpatterns = patterns('',
    url(r'^taxes/', include('taxes.urls')),
    url(r'^admin/', include(admin.site.urls)),
)
```

Έτσι λοιπόν συνδέουμε τα urls της εφαρμογής μας, με τα γενικότερα urls όλου του πρότζεκτ. Επομένως το url http://127.0.0.1:8000/taxes/tax_office θα μας οδηγήσει στην σελίδα που καθορίζεται από το view tax_office_list και με τρόπο που περιγράφεται μέσα στο html αρχείο tax_office.html και θα δούμε αναλυτικά παρακάτω.

5.5.2. Κλήση δεδομένων από την βάση

Τα δεδομένα της βάσης δεν μπορούν κληθούν απευθείας με SQL ερωτήματα από το Django, οπότε το Django χρησιμοποιεί έναν εσωτερικό μηχανισμό για να μπορέσει να τα ανακτήσει. Η κάθε κλάση του μοντέλου αντιστοιχεί σε έναν πίνακα της βάσης και το κάθε στιγμιότυπο της βάσης σε μια εγγραφή του πίνακα.

Για να γίνει η κλήση όλων των αντικειμένων από έναν πίνακα χρησιμοποιείται η συνάρτηση all(). Το επιστρεφόμενο αποτέλεσμα είναι τύπου QuerySet και περιλαμβάνει όλα τα αντικείμενα του πίνακα που πληρούν πιθανόν κάποιες συγκεκριμένες συνθήκες.

Για να ανακτήσουμε για παράδειγμα δεδομένα με τρόπο αντίστοιχο του WHERE της SQL χρησιμοποιούμε τις συναρτήσεις filter(), exclude() και get().

Οι εντολές του Django

```
 debts.objects.filter(deadline__lte='2014-01-01').order_by('-
deadline', 'debt_el')

 debts.objects.exclude(deadline__gt=datetime.date(2014, 1, 3),
debt_el='ΔΕΗ')

 debts.objects.get(id=5)
```

αντιστοιχούν στις εντολές της SQL:

```
SELECT * FROM debts WHERE deadline <= '2014-01-01'
ORDER BY deadline, debt_el;

SELECT ...
WHERE NOT (deadline > '2014-1-3' AND debt_el = 'ΔΕΗ')

SELECT *
WHERE (id = 5)
```

5.6. Templates

Εντός του φακέλου όπου βρίσκεται η εφαρμογή taxes θα δημιουργήσουμε έναν νέο φάκελο taxes/templates, για τα πρότυπα μας. Μέσα σε αυτόν θα δημιουργήσουμε άλλον έναν φάκελο με το όνομα της εφαρμογής taxes όπου εκεί θα μπούν όλα τα html αρχεία μας. Η διαδρομή τους θα είναι του τύπου taxes/templates/taxes/tax_office.html. Εξαιτίας του τρόπου που έχει στηθεί η κλήση των προτύπων από το Django μπορούμε κατά την ανάπτυξη να αναφερόμαστε σε αυτά τα πρότυπα χρησιμοποιώντας απλά το τελευταίο κομμάτι της διαδρομής, δηλαδή taxes/tax_office.html. Εάν δεν είχαμε δημιουργήσει τον εσωτερικό υποφάκελο taxes το Django θα επέλεγε το πρώτο αρχείο που θα θεωρούσε ότι το όνομα μου ταιριάζει, ακόμα και αν ανήκε σε άλλη εφαρμογή που θα αναπτύσσαμε παράλληλα. Μέσα στον φάκελο templates μπορούμε να προσθέσουμε και άλλα στατικά αρχεία, όπως css,xml,csv, js ή φωτογραφίες.

Κάθε πρότυπο περιέχει μεταβλητές, οι οποίες αντικαθίστανται από τιμές που υπολογίζονται από το πρότυπο και από ετικέτες (tags), που ελέγχουν την λογική του προτύπου.

Οι μεταβλητές έχουν την μορφή {{ variable }}. Όταν ο μηχανισμός του προτύπου βρίσκει μια μεταβλητή, υπολογίζει την μεταβλητή και την αντικαθιστά με το αποτέλεσμά της. Τα ονόματα των μεταβλητών αποτελούνται από οποιοδήποτε συνδιασμό αλφαριθμητικών χαρακτήρων, την κάτω

παύλα (_), αλλά δεν προτείνεται να χρησιμοποιείται η τελεία γιατί έχει ειδική σημασία. Σημαντικό είναι ότι δεν μπορούμε να χρησιμοποιήσουμε κενά στα ονόματα των μεταβλητών.

Οι ετικέτες έχουν την μορφή {% tag %} και είναι πιο σύνθετες από τις μεταβλητές. Κάποιες παράγουν κείμενο κατά την έξοδό τους, κάποιες ελέγχουν την ροή εφαρμόζοντας βρόγχους, πχ for, while,if κλπ και κάποιες μεταφορτώνουν εξωτερική πληροφορία που θα χρησιμοποιηθεί στην συνέχεια ως μεταβλητές του προτύπου.

Εξετάζοντας το αρχείο taxes/tax_office.html που παρατίθεται στο Παράρτημα Α, έχουμε το παρακάτω κομμάτι κώδικα:

```
{% extends 'taxes/home.html' %}

{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static
'taxes/old_style.css' %}" />


{% block content%}

{% if latest_tax_office_list %}
<div class="container">


<div class="row">
<div class="span4">
    <table border="1" style="width:300px">
      <tr class="head">
          <th>Όνομα εφορίας</th>
          <th>Οδός</th>
          <th>Αριθμός</th>
          <th>Πόλη</th>
          <th>Ταχυδρομικός Κώδικας</th>
          <th>E-mail</th>
          <th>Τηλέφωνο 1</th>
          <th>Τηλέφωνο 2</th>
      </tr>
    {% for sth in latest_tax_office_list.all %}
      <tr>
        <td>{{sth.tax_office_name}}</td>
        <td>{{sth.street|default:"-" }}</td>
        <td>{{sth.str_number|default:"-" }}</td>
        <td>{{sth.city|default:"-" }}</td>
        <td>{{sth.tk|default:"-" }}</td>
        <td>{{sth.email}}</td>
        <td>{{sth.landline1}}</td>
        <td>{{sth.landline2|default:"-" }}</td>
      </tr>
    {% endfor %}
    </table>
{% else %}
    <h2>Δεν υπάρχουν καταχωρημένες εφορίες</h2>
{% endif %}

</div>
</div>
</div>
{% endblock %}
```

Αρχικά παρατηρούμε την ετικέτα {% if .. %} που ακολουθείται από τις {% else %} και {% endif %}. Εάν η αρχική συνθήκη ικανοποιείται και η λίστα επιστρέφει αποτελέσματα, τότε το πιο κάτω κομμάτι κώδικα εκτελείται, ειδάλλως εκτελείται ο κώδικας που βρίσκεται κάτω από το else, και επιστρέφεται το μήνυμα ότι δεν είχαμε

Επιπλέον χρησιμοποιούμε την ετικέτα {% for %} για να διατρέξουμε τα αντικείμενα που βρίσκονται στην λίστα latest_tax_office_list, όπως την ορίσαμε στο αρχείο taxes/views.py. Η λίστα

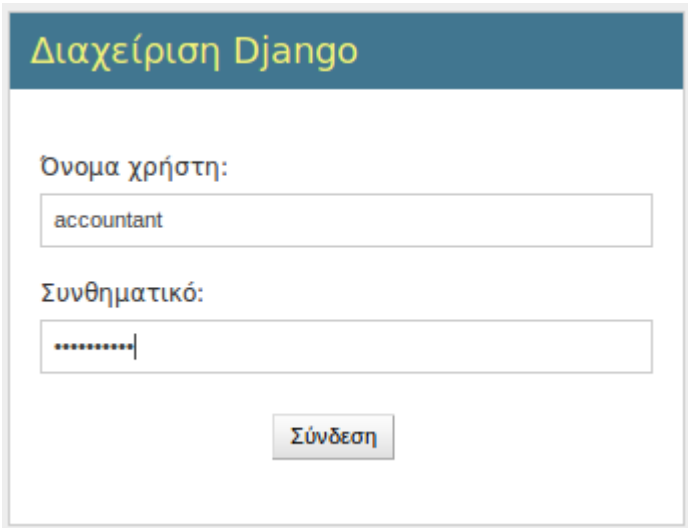
αυτή επιστρέφει αποτελέσματα και με την χρήση της τελείας ανακτούμε καθένα από τα πεδία των αντικειμένων της λίστας. Ο κώδικας εκτελείται και κλείνει στην ετικέτα {% endfor%}.

Με τις ετικέτες είναι επιπλέον δυνατόν να υλοποιηθεί και η κληρονομικότητα στα πρότυπά μας. Η ετικέτα {% extends "base.html" %} δηλώνει ότι το πρότυπό μας επεκτείνει ένα αρχικό πρότυπο σκελετό, που αποτελεί την βάση της ιστοσελίδας μας για όλα τα πρότυπα. Οπότε όταν καλούμε ένα τέτοιο πρότυπο αυτό πρώτα φορτώνει το πρότυπο γονιό και το επεκτίνει. Επιπλέον χρησιμοποιούμε την ετικέτα block {% block content %} ... {% endblock %} που οριοθετεί τον κώδικα του πρότυπου παιδιού, και καλείται στην συνέχεια από το πρότυπο γονιό για να τον συμπληρώσει.

Τέλος οι ετικέτες είναι πολύ χρήσιμες σε περιπτώσεις που θέλουμε να κάνουμε κάποια απλή πράξη ή να εκτελέσουμε καθαρό κώδικα Python, και αυτό προφανώς δεν μπορούμε να το κάνουμε μέσα στο πρότυπο. Οι πράξεις εκτελούνται μέσα σε συναρτήσεις των προβολών, δηλαδή στο αρχείο taxes/views.py και επιστρέφονται στο πρότυπο μέσα από την κλήση του.

6. Σελίδα διαχείρισης

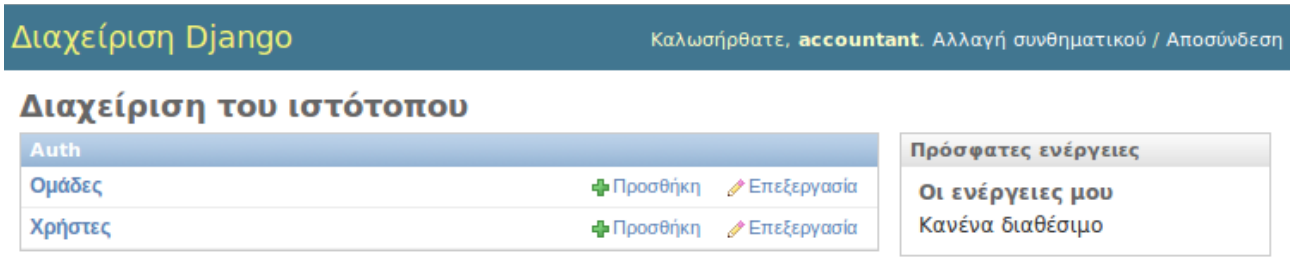
Μεταβαίνοντας στην αρχική σελίδα του διαχειριστή, προβάλλεται η φόρμα που φαίνεται στην Εικόνα 4. Εκεί συμπληρώνουμε τα συνθηματικά που ορίσαμε κατά την δημιουργία των βάσεων του συστήματος και συνδεόμαστε στην σελίδα. Παρατηρούμε ότι η ελληνική γλώσσα υποστηρίζεται αυτόματα από το Django, μιας που στις ρυθμίσεις ορίσαμε σαν γλώσσα της εφαρμογής τα ελληνικά.



Εικόνα 4 - Φόρμα εισαγωγής στην διαχείριση Django

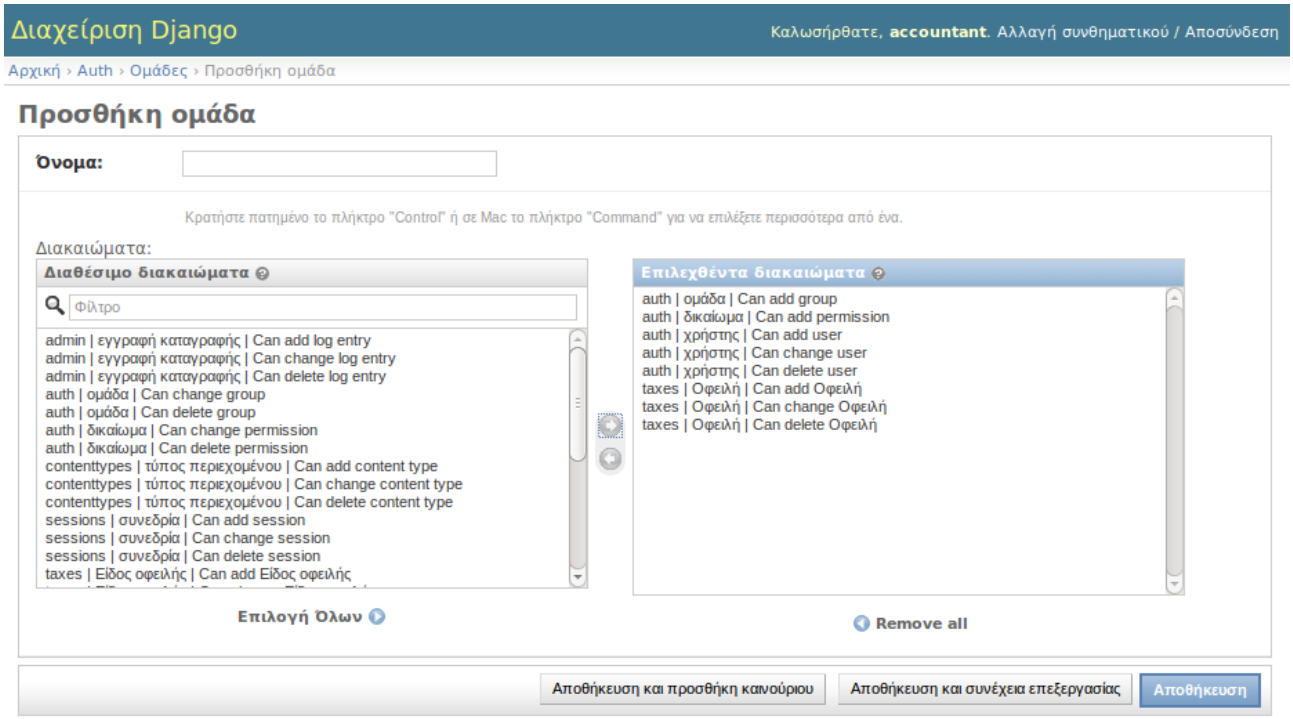
6.1. Ομάδες και χρήστες

Αφού πατήσουμε το κουμπί “Σύνδεση” στην συνέχεια εισαγόμαστε στην αρχική σελίδα της Διαχείρισης ως υπερχρήστης (superuser), όπου μπορούμε να ορίσουμε ομάδες και χρήστες και να αλλάξουμε το συνθηματικό του χρήστη με τον οποίο έχουμε συνδεθεί, όπως φαίνεται στην Εικόνα 5. Επιπλέον μπορούμε να αλλάξουμε το συνθηματικό του υπερχρήστη και να αποσυνδεθούμε από τις επιλογές στην πάνω δεξιά πλευρά της σελίδας.



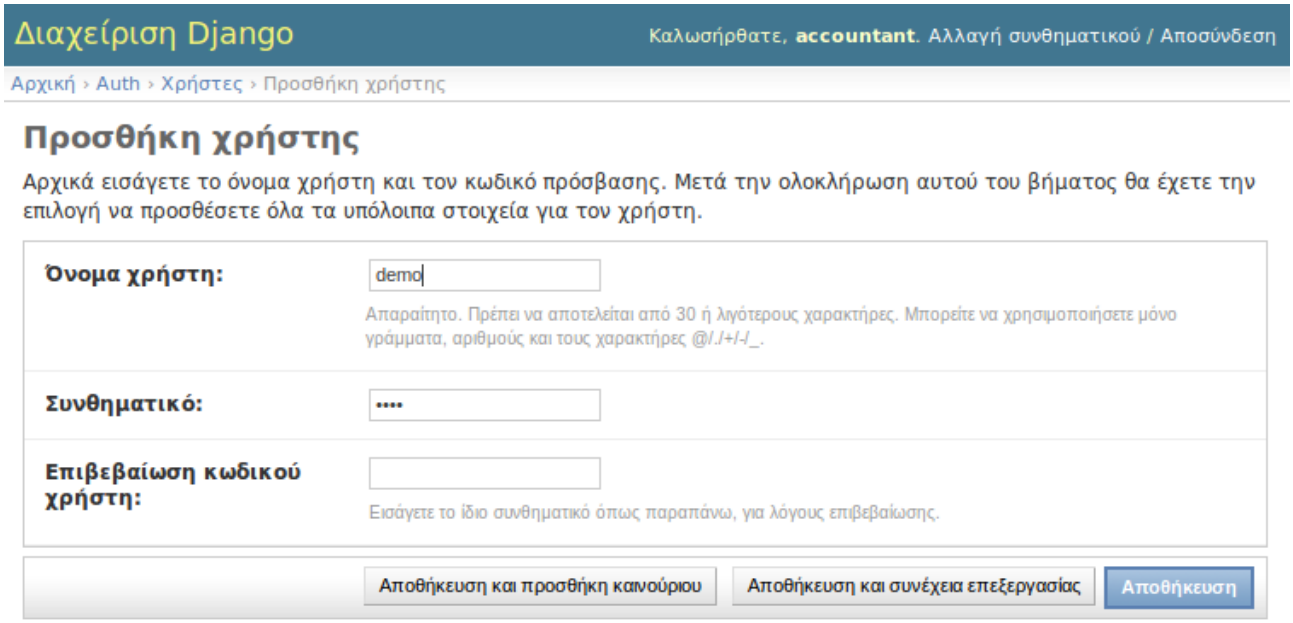
Εικόνα 5 -Διαχείριση ομάδων και χρηστών

Μπορούμε να προσθέσουμε και να επεξεργαστούμε ομάδες χρηστών και χρήστες παραχωρώντας σε κάθε ομάδα δικαιώματα που αφορούν την προσθήκη/ επεξεργασία/ διαγραφή δεδομένων στις βάσεις που έχουν δημιουργηθεί από την εφαρμογή, όπως αυτό φαίνεται στην Εικόνα 6



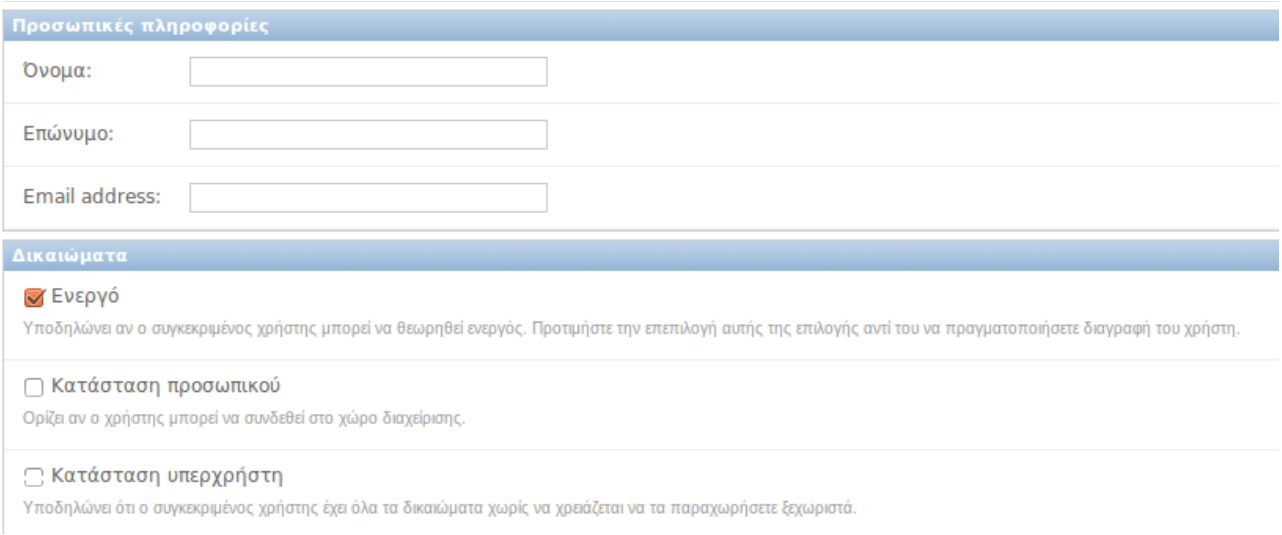
Εικόνα 6 - Προσθήκη ομάδας

Για την προσθήκη νέου χρήστη αρχικά ζητείται το όνομα χρήστη, το συνθηματικό και η επιβεβαίωση συνθηματικού και αφού καταχωρηθούν αυτά αποθηκεύουμε τις επιλογές μας(Εικόνα 7).



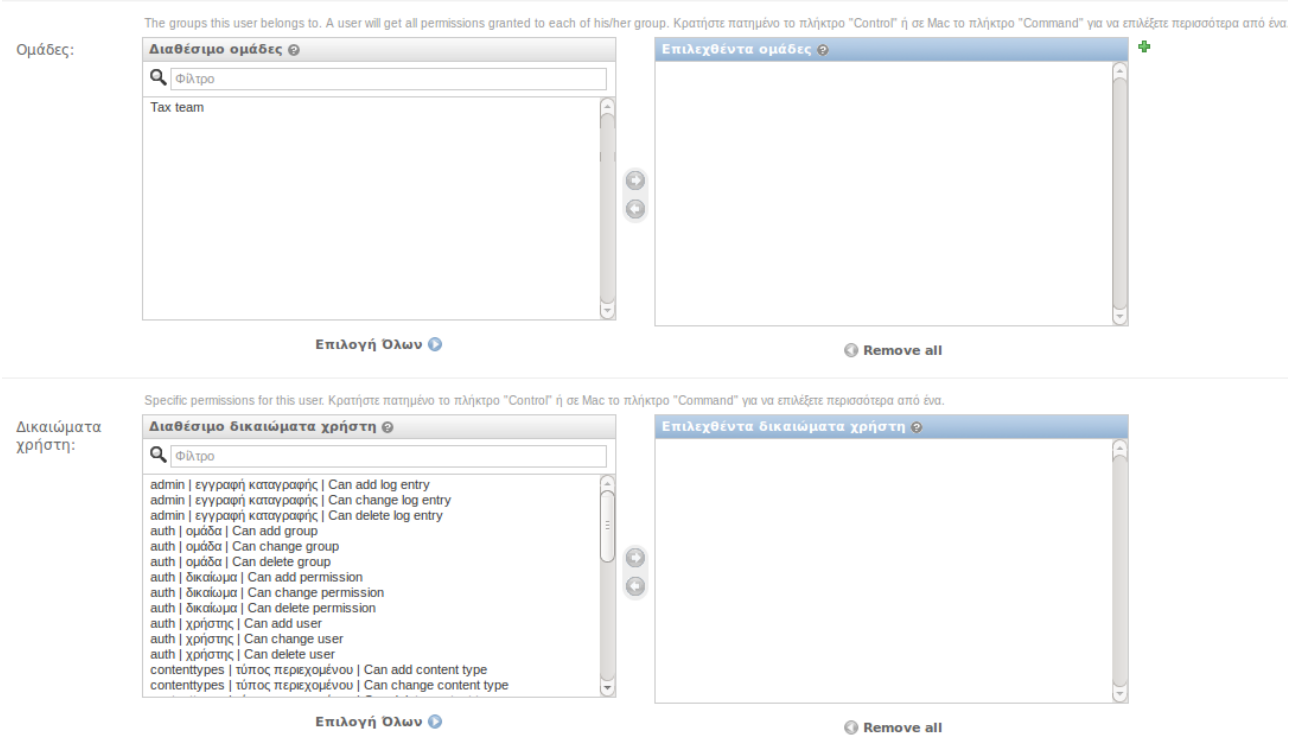
Εικόνα 7 - Προσθήκη χρήστη

Στην συνέχεια επιλέγοντας τον χρήστη αυτό μεταφερόμαστε στην σελίδα επεξεργασίας του χρήστη, όπου μπορούν να συμπληρωθούν προσωπικές πληροφορίες που αφορούν το όνομα, το επώνυμο και το email του χρήστη. Επιπλέον ορίζεται αν ο χρήστης είναι ενεργός. Προτείνεται αντί να διαγράφονται οι χρήστες, να αποεπιλέγεται αυτό το πεδίο, ώστε να χάνονται και οι ιστορικές πληροφορίες διαγράφοντας τελείως ένα χρήστη. Εάν επιλέγει η κατάσταση προσωπικού, τότε ο χρήστης μπορεί να συνδεθεί στον χώρο διαχείρισης και εδώ μπορεί να καθοριστεί αν αυτός ο χρήστης θα έχει δικαιώματα υπερχρήστη (Εικόνα 8)



Εικόνα 8 - Πληροφορίες και δικαιώματα χρήστη

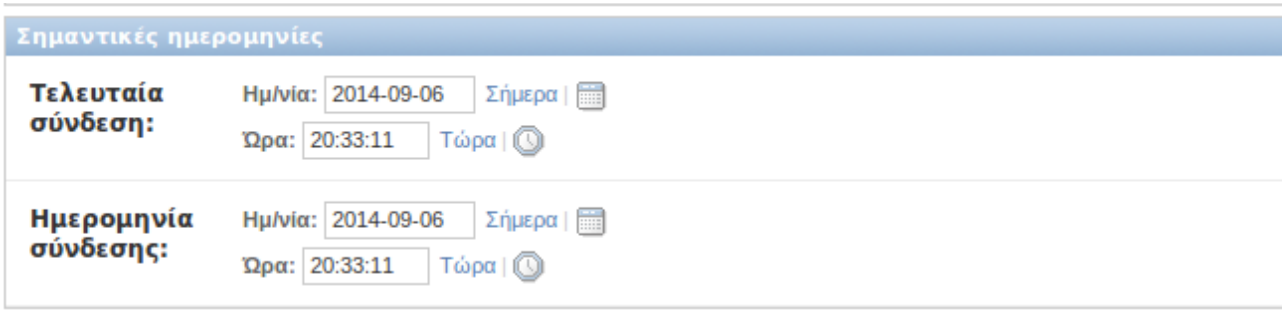
Επιπλέον ορίζονται οι ομάδες στις οποίες ανήκει ο χρήστης και τα δικαιώματά του, εάν επιθυμούμε αυτά να είναι μοναδικά για τον συγκεκριμένο χρήστη και διαφορετικά από τα δικαιώματα των ορισμένων ομάδων (Εικόνα 9).



Εικόνα 9 - Ομάδες και δικαιώματα χρήστη

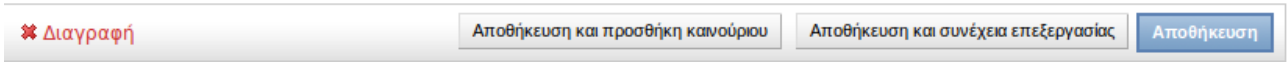
Εδώ βρίσκονται και τα στοιχεία της τελευταίας σύνδεσης του χρήστη στο σύστημα και κρατιέται

ιστορικό των μεταβολών που έχουν γίνει για τον συγκεκριμένο χρήστη (Εικόνα 10).



Εικόνα 10 - Σημαντικές ημερομηνίες σύνδεσης του χρήστη

Τέλος μπορούμε να είτε να επιβεβαιώσουμε και να αποθηκεύσουμε τις αλλαγές μας, είτε να διαγράψουμε τελείως τον χρήστη, επιλέγοντας “Αποθήκευση” ή “Διαγραφή” αντίστοιχα (Εικόνα 11).



Εικόνα 11 - Αποθήκευση ή διαγραφή χρήστη

6.2. Πίνακες της εφαρμογής taxes

Για να μπορούμε να δούμε και να διαχειριστούμε τους πίνακες, τα πεδία τους και τον τρόπο με τον οποίο εμφανίζονται οι πίνακες της εφαρμογής στην σελίδα διαχείρισης θα πρέπει να τροποποιήσουμε το αρχείο taxes/admin.py. Αναλυτικά ο κώδικας αυτός παρατίθεται στο Παράρτημα Α.

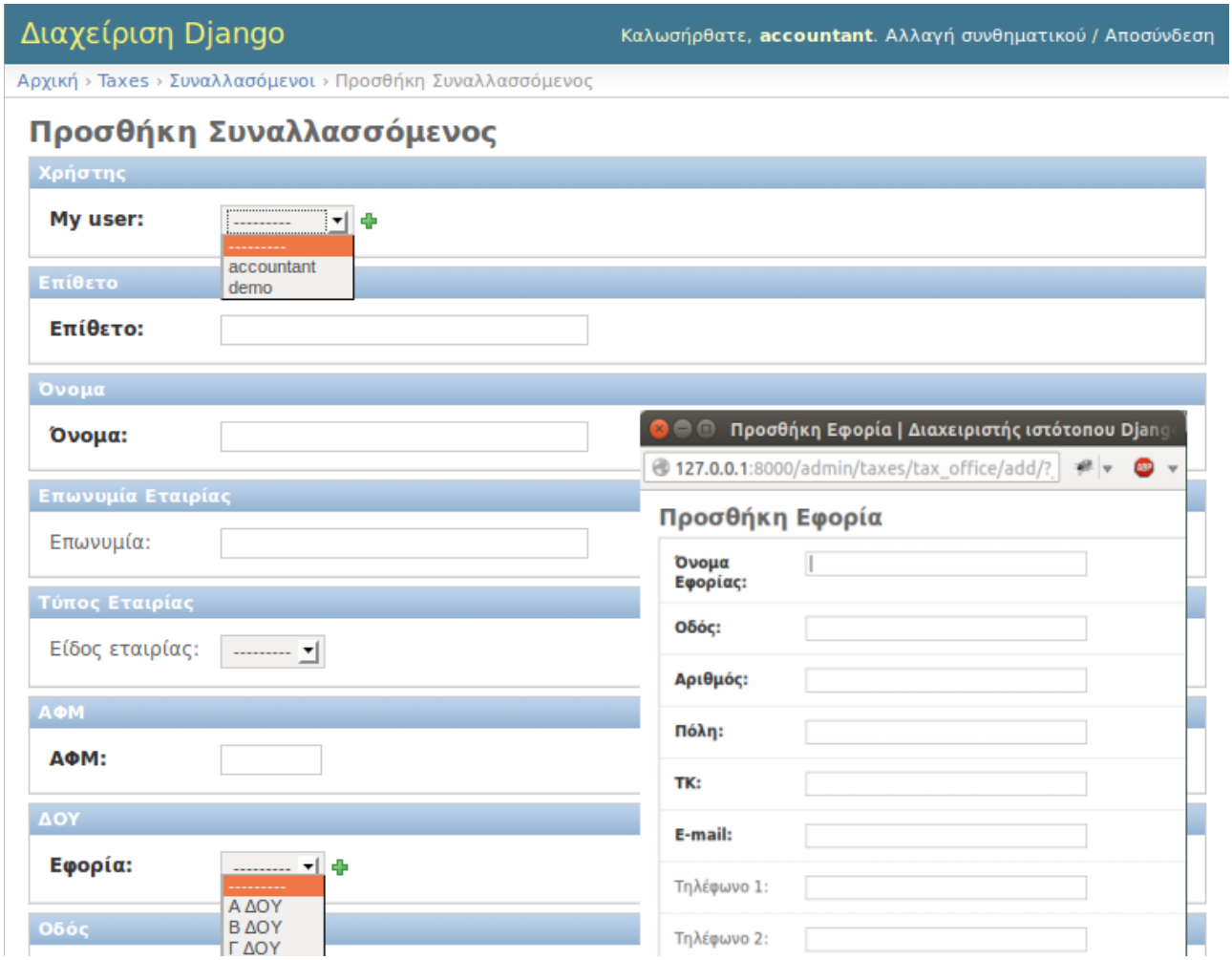
Μόλις προσθέσουμε το συγκεκριμένο αρχείο και επανεκκινήσουμε τον web server παρατηρούμε ότι στην κεντρική σελίδα εμφανίζεται και ένας ακόμα πίνακας με τον τίτλο Taxes (Εικόνα 12), όπως και η εφαρμογή μας και εκεί παρατίθενται οι πίνακες που επιλέξαμε να εμφανίζονται μέσω του αρχείου taxes/admin.py. Παρατηρούμε ότι τα λεκτικά των ονομάτων είναι αυτά που ορίσαμε σαν verbose_name στον ορισμό των κλάσεων μέσα στο αρχείο taxes/models.py.

Εικόνα 12 - Διαχείριση των πινάκων της εφαρμογής

6.2.1. Εισαγωγή δεδομένων

Επιλέγοντας κάθε μια από τις παραπάνω κατηγορίες μας επιτρέπει τόσο να προσθέσουμε όσο και να τροποποιήσουμε τους πίνακες της εφαρμογής μας.

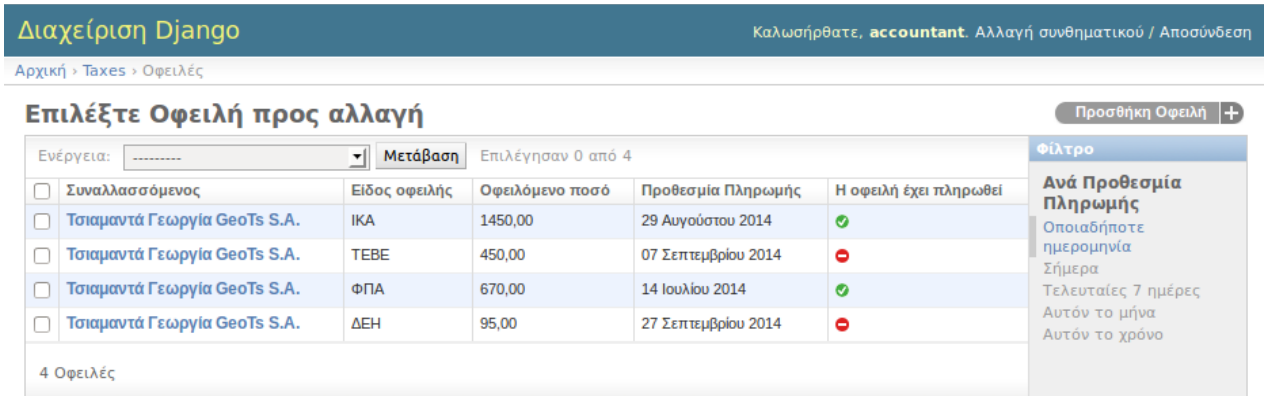
Επιλέγοντας προσθήκη Συναλλασσόμενου εμφανίζεται η σελίδα εισαγωγής νέου πελάτη, όπου συμπληρώνουμε όλα τα στοιχεία που θα καταχωρηθούν στην βάση. Μέσω του αρχείου taxes/models.py καθορίζονται τα ονόματα των ετικετών δίπλα σε κάθε πεδίο που εισάγεται στην βάση. Επιπλέον μπορούμε να χωρίσουμε τα πεδία μας σε fieldsets και να τα χωρίσουμε με αυτόν τον τρόπο οπτικά στην οθόνη μας. Παρατηρούμε ότι όταν ένα πεδίο αποτελεί ξένο κλειδί ενός άλλου πίνακα, τότε δεν μας επιτρέπεται η απευθείας συμπλήρωση του πεδίου στον πίνακα, αλλά εμφανίζεται ένα μενού από επιλογές που μπορούμε να επιλέξουμε. Σε περίπτωση που θέλουμε να προσθέσουμε και να χρησιμοποιήσουμε μια εγγραφή που δεν υπάρχει στην βάση, τότε μπορούμε να πατήσουμε στο σύμβολο συν (+) που βρίσκεται δεξιά του πεδίου. Τότε εμφανίζεται ένα νέο παράθυρο εισαγωγής για τον πίνακα που αφορά αυτό το ξένο κλειδί (Εικόνα 13)



Εικόνα 13 - Προσθήκη εγγραφής με ξένα κλειδιά

6.2.2. Προβολή δεδομένων

Επιλέγοντας έναν από τους πίνακες τότε προβάλλονται εκείνα τα πεδία και με τον τρόπο που έχουμε ορίσει στο αρχείο taxes/admin.py. Ένα τέτοιο παράδειγμα βλέπουμε στην Εικόνα 14.



Εικόνα 14 - Προβολή πίνακα οφειλών

Μπορούμε να ορίσουμε και λογικές συναρτήσεις για τα στοιχεία μας. Στο αρχείο taxes/models.py

κατά τον ορισμό της κλάσης των οφειλών ορίζουμε την παρακάτω συνάρτηση.

```
def has_been_paid(self):
        return self.deadline<=date.today()
    has_been_paid.boolean = True
    has_been_paid.short_description='Η οφειλή έχει πληρωθεί'
```

Σύμφωνα με αυτή την συνάρτηση αν η ημερομηνία πληρωμής έχει παρέλθει θεωρούμε ότι η οφειλή έχει πληρωθεί, αλλιώς θεωρούμε ότι η οφειλή εκκρεμεί.

Το Django αντιλαμβάνεται την συνάρτηση αυτή σαν boolean και την παρουσιάζει με το αντίστοιχο εικονίδιο στην σελίδα του διαχειριστή (Εικόνα 14), δηλαδή με ένα πράσινο τικ αν η οφειλή έχει πληρωθεί, επιστρέφεται δηλαδή η τιμή True από την συνάρτηση και με ένα κόκκινο απαγορευτικό αν η οφειλή δεν έχει πληρωθεί, επιστρέφεται δηλαδή η τιμή False από την συνάρτηση.

Παρατηρούμε ακόμα ότι δεξιά του πίνακα (Εικόνα 14) παρουσιάζεται και η επιλογή φιλτραρίσματος με βάση χρονικά κριτήρια. Αυτό καθορίζεται στον κώδικα από την ιδιότητα list_filter. Παρατηρούμε ότι για τις οφειλές στον αρχείο taxes/admin.py έχει προστεθεί η γραμμή :

```
list_filter = ['deadline']
```

Επιπλέον μπορούμε να θέσουμε την επιλογή της αναζήτησης μέσα σε επιλεγμένους πίνακες με βάση κάποια ή όλα τα πεδία τους. Επιπλέον για κάθε πίνακα μπορούν να επιλεγούν μία ή περισσότερες εγγραφές και να διαγραφούν από το αναδυόμενο μενού “Ενέργεια” και την επιλογή “Διαγραφή επιλεγμένων ”.

Σε γενικές γραμμές οι δυνατότητες προβολής των δεδομένων, φιλτραρίσματος τους και

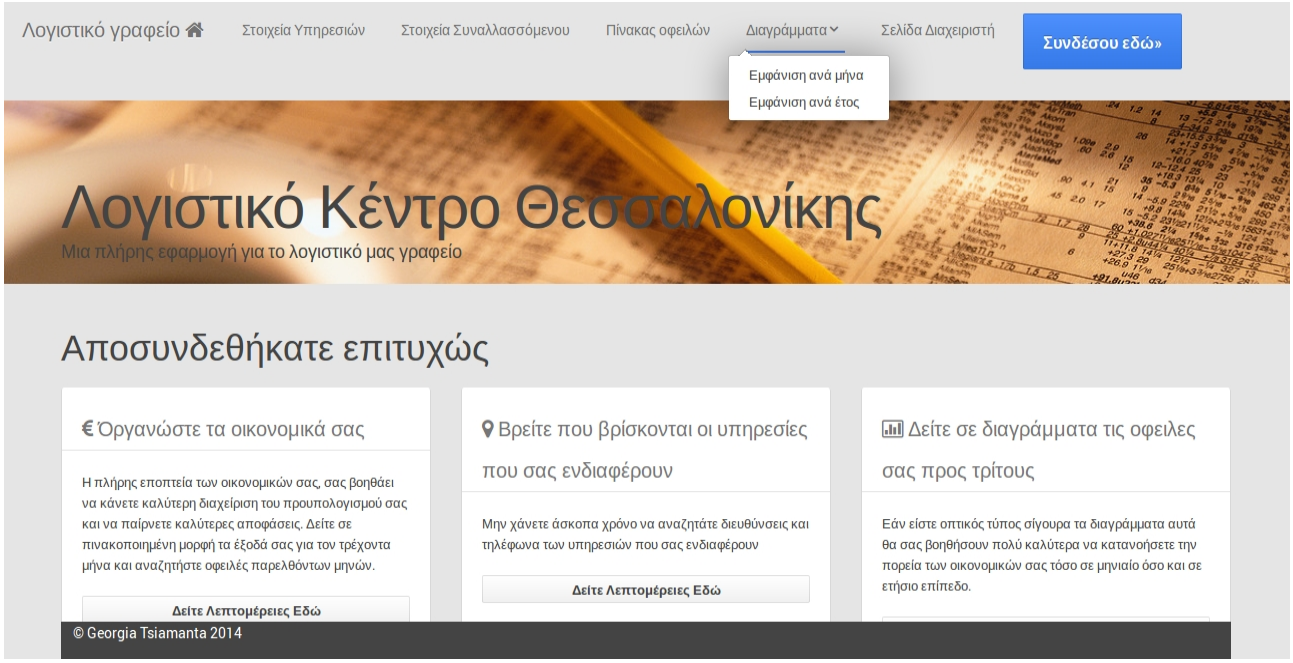
ομαδοποίησης τους που μας δίνει η σελίδα διαχείρισης είναι πολλές και αφήνουν στον προγραμματιστή να τις προσαρμόσει με εύκολο και διάφανο τρόπο, ανάλογα με τις ανάγκες του διαχειριστή και της εφαρμογής.

7. Παρουσίαση της ιστοσελίδας

7.1. Κεντρική σελίδα

Ο χρήστης της ιστοσελίδας είναι ο πελάτης του λογιστικού γραφείου ο οποίος χρησιμοποιεί την πλατφόρμα για να ενημερωθεί για τις οφειλές του, όπως αυτές έχουν καταχωρηθεί από τον λογιστή του.

Ο χρήστης εισάγεται στην αρχική σελίδα (Εικόνα 15) και μπορεί να πλοηγηθεί από την μπάρα πλοήγησης σε όλες τις διαθέσιμες σελίδες.



Εικόνα 15 - Κεντρική σελίδα εφαρμογής

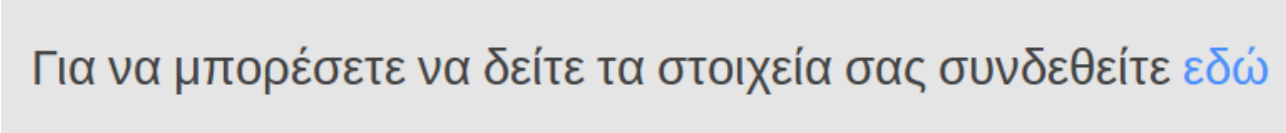
Οι επιλογές πλοήγησης είναι οι παρακάτω:

- Λογιστικό γραφείο, που είναι η αρχική σελίδα
- Στοιχεία Υπηρεσιών
- Στοιχεία Συναλλασσόμενου
- Πίνακας Οφειλών
- Διαγράμματα
    - Εμφάνιση ανά μήνα
    - Εμφάνιση ανά έτος
- Σελίδα διαχειριστή

7.2. Σύνδεση χρήστη

Ο χρήστης δεν μπορεί να έχει πρόσβαση σε όλες τις σελίδες της εφαρμογής αν δεν συνδεθεί.

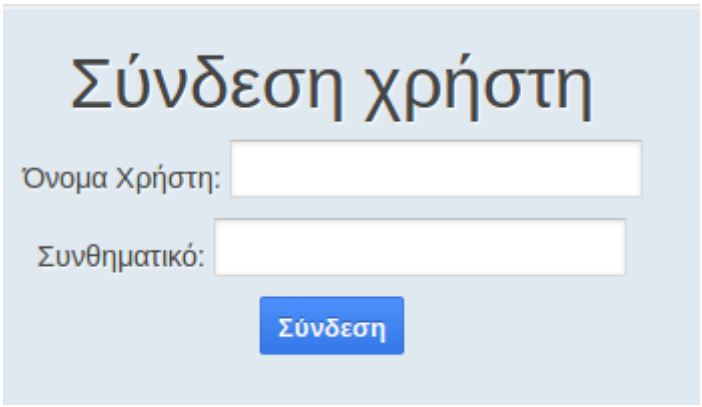
Εκτός από την αρχική σελίδα και την σελίδα “Στοιχεία υπηρεσιών”, επιλέγοντας μία από τις παρακάτω: “Στοιχεία Συναλλασσόμενου”, “Πίνακας οφειλών”, “Διαγράμματα/Εμφάνιση ανά μήνα” και “Διαγράμματα/Εμφάνιση ανά έτος” οδηγείται σε μια σελίδα όπου εμφανίζεται το μήνυμα ότι πρέπει να συνδεθεί στο σύστημα (Εικόνα 16):



Εικόνα 16 - Μήνυμα σύνδεσης στο σύστημα

Ο χρήστης θα χρειαστεί σε αυτό το σημείο να συνδεθεί στο σύστημα. Αυτό μπορεί να το κάνει επιλέγοντας το κουμπί “Συνδέσου εδώ” που βρίσκεται στο πάνω δεξιά μέρος της σελίδας, δίπλα στη μπάρα πλοήγησης.

Πατώντας το κουμπί “Συνδέσου εδώ” ο χρήστης μεταβαίνει στην σελίδα σύνδεσης στο σύστημα (Εικόνα 17)

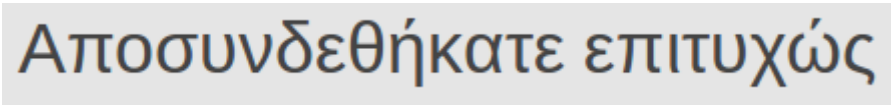


Εικόνα 17 - Σύνδεση χρήστη

Μόλις ο χρήστης συμπληρώσει το όνομα χρήστη και το συνθηματικό και πατήσει το κουμπί σύνδεση μεταβαίνει στην σελίδα επιβεβαίωσης (Εικόνα 18).

Εικόνα 18 - Επιβεβαίωση σύνδεσης

Παρατηρούμε ότι εκτός από το μήνυμα “Συνδεθήκατε επιτυχώς”, αυτό που έχει αλλάξει είναι το κουμπί πάνω δεξιά το οποίο έχει πλέον μεταβληθεί σε “Αποσύνδεση”, γιατί το σύστημα αναγνωρίζει ότι ο χρήστης μας είναι συνδεδεμένος. Σε περίπτωση που ο χρήστης θέλει να αποσυνδεθεί τότε πατώντας το κουμπί “Αποσύνδεση”, μεταβαίνει στην σελίδα αποσύνδεσης και λαμβάνει το μήνυμα της επιτυχούς αποσύνδεσης.



Εικόνα 19 - Αποσύνδεση χρήστη

7.3. Στοιχεία Υπηρεσιών

Ο χρήστης επιλέγει από την μπάρα πλοήγησης τα “Στοιχεία Υπηρεσιών” και μεταφέρεται στην αντίστοιχή σελίδα. Εκεί προβάλλονται σε πινακοποιημένη μορφή τα στοιχεία των υπηρεσιών που τον ενδιαφέρουν, όπως την διεύθυνση, το email και το τηλέφωνό τους (Εικόνα 20)

Όνομα εφορίας	Οδός	Αριθμός	Πόλη	Ταχυδρομικός Κώδικας	E-mail	Τηλέφωνο 1	Τηλέφωνο 2
Α ΔΟΥ	Παπαναστασίου	120	Θεσσαλονίκη	54249	aeforia@mtax.gr	2310365419	-
Β ΔΟΥ	Μαρτίου	59	Θεσσαλονίκη	54245	beforia@mtax.gr	2310598769	2310598768
Γ ΔΟΥ	Τσιμισκή	54	Θεσσαλονίκη	54215	ceforia@mtax.gr	2310250174	2310250175

Εικόνα 20 - Στοιχεία Υπηρεσιών

7.4. Στοιχεία Συναλλασσόμενου

Ο χρήστης επιλέγει από την μπάρα πλοήγησης τα “Στοιχεία Συναλλασσόμενου” και μεταφέρεται στην αντίστοιχή σελίδα. Εκεί προβάλλονται σε πινακοποιημένη μορφή τα στοιχεία του αναλυτικά. (Εικόνα 21)

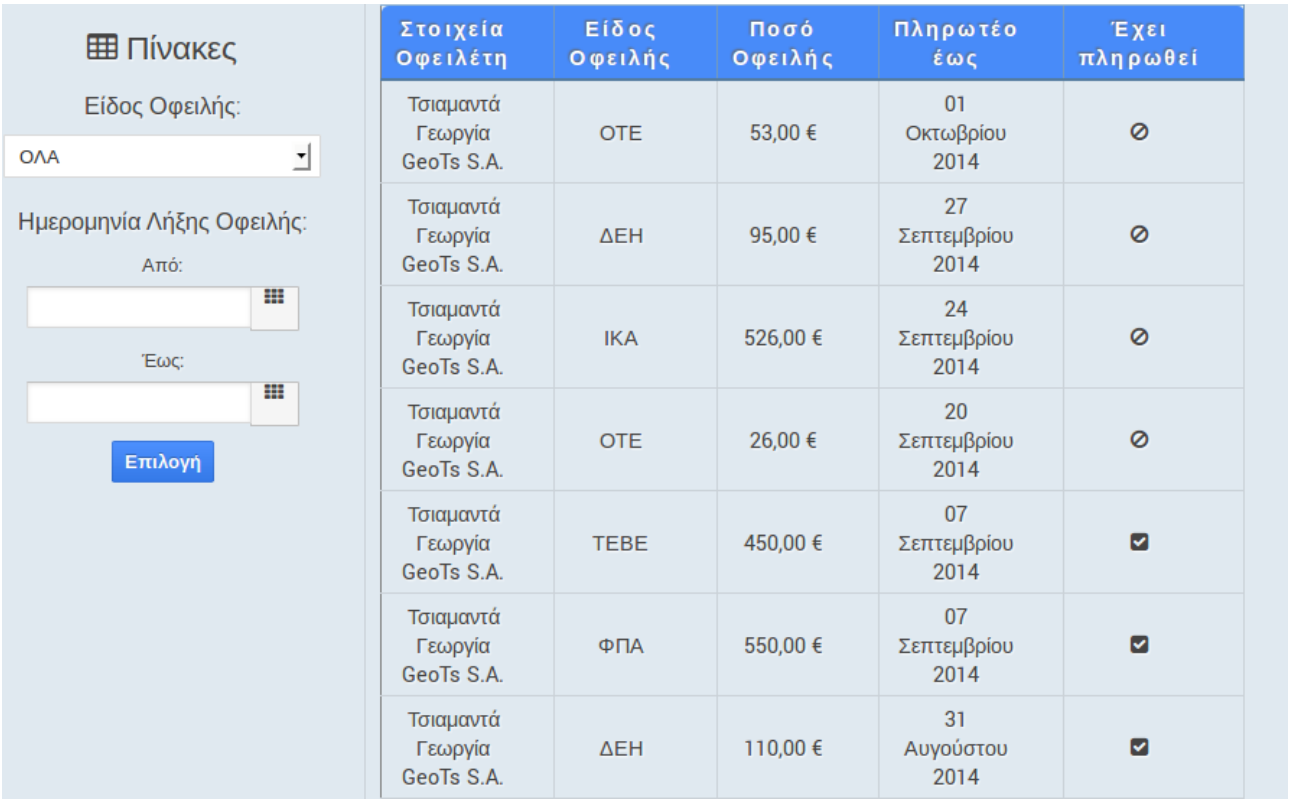
Επώνυμο	Όνομα	Όνομα Επιχείρησης	Είδος Επιχείρησης	ΑΦΜ	Εφορία	Συνθηματικό	email
Τσιαμαντά	Γεωργία	GeoTs S.A.	ΑΕ	245863110	Α ΔΟΥ	georgia	georgia@mytax.gr

Εικόνα 21 - Στοιχεία Συναλλασσόμενου

7.5. Πίνακας οφειλών

Ο χρήστης επιλέγει από την μπάρα πλοήγησης τον “Πίνακα Οφειλών” και μεταφέρεται στην αντίστοιχή σελίδα. Εκεί προβάλλονται σε πινακοποιημένη μορφή τα στοιχεία των οφειλών του.

Η προβολή των στοιχείων μπορεί να φιλτραριστεί με βάση το είδος τους και με βάση την ημερομηνία λήξης πληρωμής τους. Από το πεδίο “Είδος οφειλής” μπορεί να επιλεγεί από το αναδυόμενο μενού η οφειλή που τον ενδιαφέρει. Επιπλέον μπορεί στο πεδίο “Ημερομηνία Λήξης οφειλής” να θέσει το διάστημα “Από”, “Έως” και να καθορίσει από το αναδυόμενο παράθυρο ημερολογίου που θα εμφανιστεί τις ημερομηνίες που θέλει. Επιπλέον στο είδος οφειλής υπάρχει η επιλογή “ΟΛΑ” ώστε να μπορεί ο χρήστης να ενημερωθεί για όλες τις οφειλές του. (Εικόνα 22).



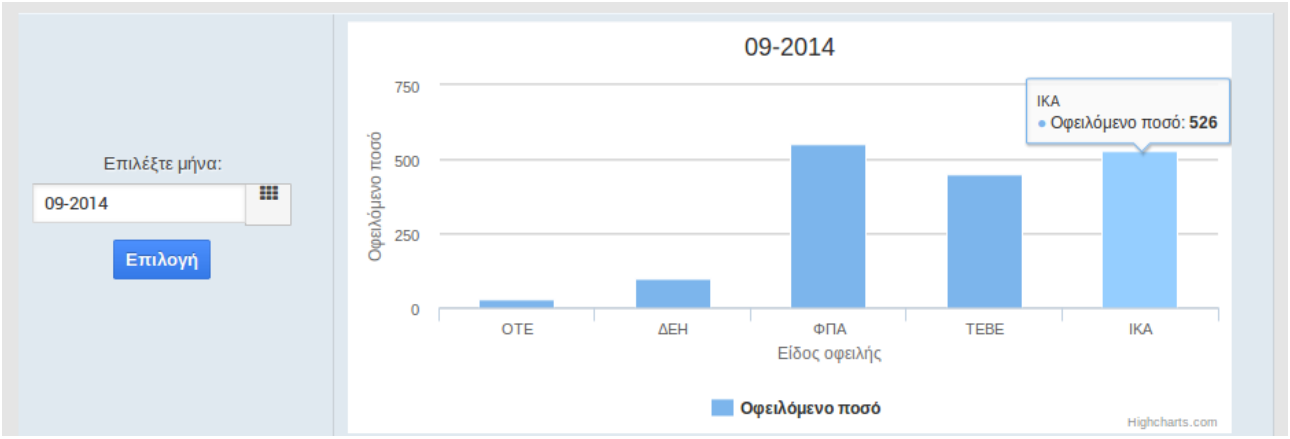
Στοιχεία Οφειλέτη	Είδος Οφειλής	Ποσό Οφειλής	Πληρωτέο έως	Έχει πληρωθεί
Τσιαμαντά Γεωργία GeoTs S.A.	ΟΤΕ	53,00 €	01 Οκτωβρίου 2014	⊘
Τσιαμαντά Γεωργία GeoTs S.A.	ΔΕΗ	95,00 €	27 Σεπτεμβρίου 2014	⊘
Τσιαμαντά Γεωργία GeoTs S.A.	ΙΚΑ	526,00 €	24 Σεπτεμβρίου 2014	⊘
Τσιαμαντά Γεωργία GeoTs S.A.	ΟΤΕ	26,00 €	20 Σεπτεμβρίου 2014	⊘
Τσιαμαντά Γεωργία GeoTs S.A.	ΤΕΒΕ	450,00 €	07 Σεπτεμβρίου 2014	☑
Τσιαμαντά Γεωργία GeoTs S.A.	ΦΠΑ	550,00 €	07 Σεπτεμβρίου 2014	☑
Τσιαμαντά Γεωργία GeoTs S.A.	ΔΕΗ	110,00 €	31 Αυγούστου 2014	☑

Εικόνα 22 - Πίνακας οφειλών

7.6. Διαγράμματα

7.6.1. Εμφάνιση ανά μήνα

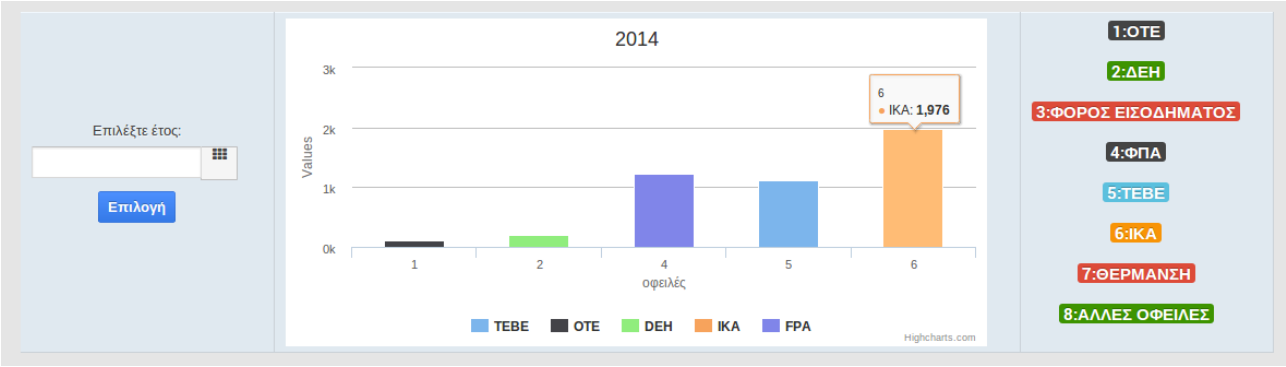
Ο χρήστης επιλέγει από την μπάρα πλοήγησης “Διαγράμματα” → “Εμφάνιση ανά μήνα” και μεταφέρεται στην αντίστοιχή σελίδα. Εκεί προβάλλονται σε διαγραμματική μορφή τα στοιχεία των οφειλών του ανά μήνα, και έτσι μπορεί να έχει καλύτερη εποπτεία της συνολικής εικόνας των οφειλών. Η προβολή των διαγραμμάτων φιλτράρεται με βάση το μήνα και ο χρήστης μπορεί να επιλέξει τον μήνα που τον ενδιαφέρει (Εικόνα 23).



Εικόνα 23 - Διάγραμμα προβολής ανά μήνα

7.6.2. Εμφάνιση ανά έτος

Ο χρήστης επιλέγει από την μπάρα πλοήγησης “Διαγράμματα” → “Εμφάνιση ανά έτος” και μεταφέρεται στην αντίστοιχή σελίδα. Εκεί προβάλλονται σε διαγραμματική μορφή τα στοιχεία των οφειλών του ανά έτος αθροιστικά. Δηλαδή αθροίζεται το ποσό που δαπάνησε ο συναλλασσόμενος ανά κατηγορία το συγκεκριμένο έτος. Η προβολή των διαγραμμάτων φιλτράρεται με βάση το μήνα και ο χρήστης μπορεί να επιλέξει το έτος που τον ενδιαφέρει (Εικόνα 24).



Εικόνα 24 - Διάγραμμα προβολής ανά έτος

7.7. Σελίδα Διαχειριστή

Ο χρήστης επιλέγοντας από την μπάρα πλοήγησης “Σελίδα Διαχειριστή”, ανακατευθύνεται στην σελίδα διαχειριστή όπου μπορεί να συνδεθεί και να να προβεί στις ενέργειες διαχείρισης ανάλογα με το επίπεδο πρόσβασης που του έχει παραχωρηθεί. Αναλυτικά αυτές οι λειτουργίες έχουν περιγραφεί στο Κεφάλαιο 6 .

8. Συμπεράσματα

Έχοντας μεταβεί σε μια πλήρη ηλεκτρονική διαχείριση όλων των συναλλαγών και των αναγκών των πελατών, ένα λογιστικό γραφείο καταφέρνει να παρέχει μεγαλύτερη και πληρέστερη ενημέρωση στους πελάτες του, όπως επίσης και να προσφέρει υπηρεσίες προστιθέμενης αξίας για το ίδιο. Μελλοντικές επεκτάσεις του συστήματος μπορεί να είναι η αυτόματη αποστολή μηνυμάτων όταν μια νέα οφειλή έχει εισαχθεί στο σύστημα και η δυνατότητα στον συναλλασσόμενο να μεταβάλλει ο ίδιος τα στοιχεία των οφειλών του. Παρέχοντας όλες αυτές τις δυνατότητες ο συναλλασσόμενος μπορεί και έχει καλύτερη διαχείριση των οικονομικών του και να προβαίνει σε πληρέστερο έλεγχο τους καθώς και να μπορεί να κάνει πιο πλήρη οικονομικό απολογισμό και προγραμματισμό.

9. Παράρτημα Α – Κώδικας

settings.py

```
"""
Django settings for tax_office project.

For more information on this file, see
https://docs.djangoproject.com/en/1.6/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/1.6/ref/settings/
"""

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
import os
BASE_DIR = os.path.dirname(os.path.dirname(__file__))


# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/1.6/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'w@tffd!p_b@_$631d6b*yk9llf=6yehhco0nz!su1irhyu@s3('

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

TEMPLATE_DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
```

```
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'taxes',
    'chartit',
)

MIDDLEWARE_CLASSES = (
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
    'django.contrib.sessions.serializers.JSONSerializer',
    #'django.middleware.locale.LocaleMiddlewar',
)

ROOT_URLCONF = 'tax_office.urls'

WSGI_APPLICATION = 'tax_office.wsgi.application'


# Database
# https://docs.djangoproject.com/en/1.6/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE':'django.db.backends.mysql',
        'NAME': 'tax_office',
        'USER': 'accountant',
        'PASSWORD': 'acc!@#$',
        'HOST': '127.0.0.1',
        'PORT': '3306',
    }
}

# Internationalization
# https://docs.djangoproject.com/en/1.6/topics/i18n/

LANGUAGE_CODE = 'el-GR'
```

```
TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True

DEFAULT_CHARSET= 'utf-8'


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/1.6/howto/static-files/

STATIC_URL = '/static/'

LOGIN_URL = '/admin/'
```

urls.py

```
from django.conf.urls import patterns, include, url
from django.contrib import admin
admin.autodiscover()

urlpatterns = patterns('',
    # Examples:
    # url(r'^$', 'tax_office.views.home', name='home'),
    # url(r'^blog/', include('blog.urls')),

    url(r'^admin/', include(admin.site.urls)),
    url(r'^taxes/', include('taxes.urls')),
)
```

taxes/models.py

```
# -*- coding: utf-8 -*-

from django.db import models
from django.contrib.auth.models import User
from datetime import date
```

```
class address(models.Model):
    street=models.CharField("Οδός",max_length=20)
    str_number=models.CharField("Αριθμός",max_length=5)
    city=models.CharField("Πόλη",max_length=30)
    tk=models.CharField("ΤΚ",max_length=10)
    email=models.EmailField("e-mail",max_length=40)
    class Meta:
        abstract = True

class tax_office(address):
    tax_office_name = models.CharField("Όνομα Εφορίας",max_length=30)
    landline1=models.CharField("Τηλέφωνο 1",max_length=15,null=True,blank=True)
    landline2=models.CharField("Τηλέφωνο 2",max_length=15,null=True,blank=True)
    def __unicode__(self):
        return self.tax_office_name
    class Meta:
        verbose_name="Εφορία"
        verbose_name_plural="Εφορίες"

class person(address):
    my_user = models.ForeignKey(User,unique=True)
    comp_type=(('Idiotis','Ιδιώτης'),('OE','OE'),('AE','AE'),('EE','EE'),)
    last_name=models.CharField("Επίθετο",max_length=20)
    first_name=models.CharField("Όνομα",max_length=20)
    company_name=models.CharField("Επωνυμία",max_length=40,null=True,blank=True)
    company_type=models.CharField("Είδος
εταιρίας",max_length=20,choices=comp_type,null=True,blank=True)
    afm=models.IntegerField("ΑΦΜ",max_length=9)
    tax_office_name= models.ForeignKey(tax_office,verbose_name='Εφορία')
    def __unicode__(self):
        return u"%s %s %s" %(self.last_name,self.first_name,self.company_name)
    class Meta:
        verbose_name="Συναλλασσόμενος"
        verbose_name_plural="Συναλλασόμενοι"

class telephone(models.Model):
    typos=(('landline','Σταθερό'), ('mobile','Κινητό'),)
    phonenumber=models.CharField("Τηλέφωνο",max_length=20)
    phone_type=models.CharField("Τύπος
τηλεφώνου",max_length=20,choices=typos,null=True,blank=True)
```

```
    person=models.ForeignKey(person)
    def __unicode__(self):
        return self.phonenumber

class debt_type(models.Model):
    debt_el=models.CharField("Είδος Οφειλής",max_length=20)
    debt_en=models.CharField("Είδος
Οφειλής",max_length=20,null=True,blank=True,editable=False)

    def __unicode__(self):
        return self.debt_el
    class Meta:
        verbose_name="Είδος οφειλής"
        verbose_name_plural="Είδη οφειλών"

class debts(models.Model):
    last_name=models.ForeignKey('person',verbose_name='Συναλλασσόμενος')
    debt_el=models.ForeignKey(debt_type,verbose_name='Είδος οφειλής')
    amount=models.DecimalField("Οφειλόμενο ποσό",max_digits=10,
decimal_places=2)
    deadline=models.DateField("Προθεσμία Πληρωμής",auto_now=False,
auto_now_add=False)

    def __unicode__(self):
        return u"%s %s %s" %(self.last_name,self.debt_el,self.amount)
    def has_been_paid(self):
        return self.deadline<=date.today()
    has_been_paid.boolean = True
    has_been_paid.short_description='Η οφειλή έχει πληρωθεί'
    class Meta:
        verbose_name="Οφειλή"
        verbose_name_plural="Οφειλές"
```

taxes/admin.py

```
import os, sys
from django.contrib import admin
from taxes.models import *
```

```
class ChoicePhone(admin.StackedInline):
    model = telephone
    extra = 2

class PersonAdmin(admin.ModelAdmin):
    fieldsets = [
        ('Χρήστης', {'fields': ['my_user']}),
        ('Επίθετο', {'fields': ['last_name']}),
        ('Όνομα', {'fields': ['first_name']}),
        ('Επωνυμία Εταιρίας', {'fields': ['company_name']}),
        ('Τύπος Εταιρίας', {'fields': ['company_type']}),
        ('ΑΦΜ', {'fields': ['afm']}),
        ('ΔΟΥ', {'fields': ['tax_office_name']}),
        ('Οδός', {'fields': ['street']}),
        ('Αριθμός', {'fields': ['str_number']}),
        ('Πόλη', {'fields': ['city']}),
        ('ΤΚ', {'fields': ['tk']}),
        ('email', {'fields': ['email']}),
    ]
    inlines= [ChoicePhone,]
    search_fields = ['last_name','company_name']
    list_display =
['last_name','first_name','company_name','company_type','afm','tax_office_name',
]

class OfeilesAdmin(admin.ModelAdmin):
    def save_prothesmia(self,request, obj, form, change):
        obj.dhf=request.deadline
        obj.save()

    list_display = ('last_name','debt_el','amount','deadline','has_been_paid')
    list_filter = ['deadline']

class DoiAdmin(admin.ModelAdmin):
    fields = ['tax_office_name','street','str_number','city','tk','email',
'landline1','landline2']
    list_filter = ['tax_office_name']
    verbose_name='Εφορία'


admin.site.register(tax_office,DoiAdmin)
```

```
admin.site.register(person,PersonAdmin)
admin.site.register(debts,OfeilesAdmin)
admin.site.register(debt_type)
```

taxes/views.py

```
# -*- coding: utf-8 -*-

from django.shortcuts import render_to_response
from django.shortcuts import render
from django.template import RequestContext, loader
from django.http import HttpResponse,HttpResponseRedirect

from django.contrib import auth
from django.contrib.auth import authenticate, login
from django.contrib.auth.models import User
from django.core.context_processors import csrf

from taxes.models import *
from django.db.models import Sum
from chartit import DataPool, Chart ,PivotChart, PivotDataPool
import datetime

def home_page(request):
    template = loader.get_template('taxes/first_page.html')
    context=RequestContext(request)
    return HttpResponse(template.render(context))

def login(request):
    c={}
    c.update(csrf(request))
    return render_to_response('taxes/login.html',c)

def auth_view(request):
    username=request.POST.get('username','')
    password=request.POST.get('password','')
    user =auth.authenticate(username=username, password=password)

    if user is not None :
```

```
        auth.login(request,user)
        return HttpResponseRedirect('/taxes/loggedin')
    else :
        return HttpResponseRedirect('/taxes/invalid')

def my_view(request):
    username = request.POST['username']
    password = request.POST['password']
    user = authenticate(username=username, password=password)
    if user is not None:
        if user.is_active:
            login(request, user)

def loggedin(request):
  template = loader.get_template('taxes/loggedin.html')
  context = RequestContext(request)
  return HttpResponse(template.render(context))

def logout(request):
    auth.logout(request)
    template = loader.get_template('taxes/logout.html')
    context = RequestContext(request)

    return HttpResponse(template.render(context))

def person_list(request):
    a=request.user.id
    latest_tax_list = person.objects.filter(my_user=a)
    template = loader.get_template('taxes/person_list.html')
    context = RequestContext(request, {
            'latest_tax_list': latest_tax_list,
    })
    return HttpResponse(template.render(context))

def tax_office_list(request):
    latest_tax_office_list =
tax_office.objects.all().order_by('tax_office_name')
    template = loader.get_template('taxes/tax_office_list.html')
    context = RequestContext(request, {
        'latest_tax_office_list': latest_tax_office_list,
    })
```

```
    return HttpResponse(template.render(context))

def tables(request):
  if request.user.is_authenticated():
    a=request.user.id
    this_pelatis=person.objects.get(my_user=a).id
    debt_list=debt_type.objects.all()

    eidos1 = request.POST.get("eidos1", "")
    deadline1= request.POST.get('deadline1',"")
    deadline2= request.POST.get('deadline2',"")

    my_eid=debt_type.objects.filter(debt_el=eidos1)
    try:
      d1=int(deadline1[0:2])
      m1=int(deadline1[3:5])
      y1=int(deadline1[6:10])
      d2=int(deadline2[0:2])
      m2=int(deadline2[3:5])
      y2=int(deadline2[6:10])

      if my_eid :
        latest_debt_list = debts.objects.order_by('-
deadline').filter(last_name=this_pelatis,debt_el=my_eid).filter(deadline__gte=da
te(y1,m1,d1)).filter(deadline__lte=date(y2,m2,d2))
      else:
        latest_debt_list = debts.objects.order_by('-
deadline').filter(last_name=this_pelatis).filter(deadline__gte=date(y1,m1,d1)).f
ilter(deadline__lte=date(y2,m2,d2))
    except Exception:
      if my_eid:
        latest_debt_list = debts.objects.order_by('-
deadline').filter(last_name=this_pelatis,debt_el=my_eid)
      else:
        latest_debt_list = debts.objects.order_by('-
deadline').filter(last_name=this_pelatis)

    template = loader.get_template('taxes/tables.html')
    context = RequestContext(request, {
        'latest_eidos_list': latest_debt_list,
        'debt_list':debt_list,
```

```
        'eidos1':eidos1,
    })
    return HttpResponse(template.render(context))
 else:
    template = loader.get_template('taxes/tables.html')
    context = RequestContext(request)

    return HttpResponse(template.render(context))



def debts_list(request):
    a=request.user.id
    latest_debts_list = debts.objects.filter(last_name=a)
    template = loader.get_template('taxes/charts.html')
    context = RequestContext(request, {
        'latest_debts_list': latest_debt_list,
    })
    return HttpResponse(template.render(context))


def year_charts(request):

  if request.user.is_authenticated():
    a=request.user.id
    imerominia2= request.POST.get("imerominia2", "2014")
    this_pelatis2=person.objects.get(my_user=a).id

    try:
      y1=int(imerominia2[0:4])

this_chart2=debts.objects.filter(last_name=this_pelatis2,deadline__year=y1)
      chart_title2=y1
    except Exception:

this_chart2=debts.objects.filter(last_name=this_pelatis2).filter(deadline__year=
datetime.datetime.now().strftime('%Y'))
      chart_title2=datetime.datetime.now().strftime('%Y')

    def change(debt_el):
         deko={"1":"ΟΤΕ","2":"ΔΕΗ","3":"ΦΟΡΟΣ
ΕΙΣΟΔΗΜΑΤΟΣ","4":"ΦΠΑ","5":"ΤΕΒΕ","6":"ΙΚΑ","7":"ΘΕΡΜΑΝΣΗ","8":"ΑΛΛΕΣ ΟΦΕΙΛΕΣ"}
```

```
        return (deko[debt_el])

data2 = \
    PivotDataPool(
       series =
        [{'options': {
           'source': this_chart2,
           'categories': ['debt_el'],
           'legend_by':'debt_el__debt_en'
           },
          'terms': {
            'Etisio_athroisma': Sum('amount')
            }}],
          #sortf_mapf_mts=(None,change,False)
         )

#Step 2: Create the PivotChart object
cht2 = \
    PivotChart(
        datasource = data2,
        series_options =
          [{'options':{
              'type': 'column',
              'stacking': True},
            'terms':[
              'Etisio_athroisma']}],
        chart_options =
          {'title': {
               'text': chart_title2},
           'xAxis': {
                'title': {
                   'text': 'οφειλές'}}},
        )


template = loader.get_template('taxes/year_charts.html')
context = RequestContext(request, {
          'my_chart2': cht2}
  )
return HttpResponse(template.render(context))
```

```
  else:
    template = loader.get_template('taxes/year_charts.html')
    context = RequestContext(request)

    return HttpResponse(template.render(context))


def month_charts(request):
  if request.user.is_authenticated():
    a=request.user.id
    imerominia = request.POST.get("imerominia", "09-2014")
    this_pelatis=person.objects.get(my_user=a).id

    try:
      d1=int(imerominia[0:2])
      m1=int(imerominia[3:7])

this_chart=debts.objects.filter(last_name=this_pelatis,deadline__month=d1,deadli
ne__year=m1)
      chart_title=imerominia
    except Exception:

this_chart=debts.objects.filter(last_name=this_pelatis,deadline__month=int(datet
ime.datetime.now().strftime('%m')),deadline__year=int(datetime.datetime.now().st
rftime('%y')))
      chart_title=datetime.datetime.now().strftime('%m')+"-"+
datetime.datetime.now().strftime('%Y')

    data = \
          DataPool(
            series=
              [{'options': {
                'source': this_chart,
                'categories': ['debt_el'],
                'legend_by':'debt_el__debt_en'
                },
                'terms':['amount','debt_el']

                }
              ])
    def change(x):
```

```
          deko={1:'ΟΤΕ',2:'ΔΕΗ',3:'ΦΟΡΟΣ
ΕΙΣΟΔΗΜΑΤΟΣ',4:'ΦΠΑ',5:'ΤΕΒΕ',6:'ΙΚΑ',7:'ΓΕΡΜΑΝΣΗ',8:'ΑΛΛΕΣ ΟΦΕΙΛΕΣ'}
          return deko[x]

      #Step 2: Create the Chart object
    cht = Chart(
          datasource = data,
          series_options =
                [{'options':{
                    'type': 'column'},
                  'terms' :{'debt_el':
                            ['amount']
                            },
                   }],
          chart_options =
                {'title': {
                  'text': chart_title},
                'xAxis': {
                  'text': 'debt_type',
                   }},
          x_sortf_mapf_mts=(None,change,False))
      #Step 3: Send the chart object to the template.

    template = loader.get_template('taxes/month_charts.html')
    context = RequestContext(request, {
              'my_chart': cht}
      )
    return HttpResponse(template.render(context))

  else:
    template = loader.get_template('taxes/month_charts.html')
    context = RequestContext(request)

    return HttpResponse(template.render(context))
```

taxes/urls.py

```
from django.conf.urls import patterns, url

from taxes import views
```

```
urlpatterns = patterns('',
    url(r'^$', views.home_page, name='home_page'),

    url(r'^person_list', views.person_list, name='person_list'),
    url(r'^tax_office_list', views.tax_office_list, name='tax_office_list'),
    url(r'^month_charts', views.month_charts, name='month_charts'),
    url(r'^year_charts', views.year_charts, name='year_charts'),
    url(r'^tables', views.tables, name='tables'),

    url(r'^login', views.login, name='login'),
    url(r'^auth', views.auth_view, name='auth'),
    url(r'^logout', views.logout, name='logout'),
    url(r'^loggedin', views.loggedin, name='loggedin'),
)
```

taxes/templates/taxes/home.html

```
{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />

<!DOCTYPE html>
<html lang="en">
   <head>

      <meta charset="utf-8">
      <title>Λογιστικό Κέντρο</title>

      <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
      <meta name="description" content="">
      <meta name="author" content="">

      <!-- Le styles -->
      <link href='http://fonts.googleapis.com/css?family=Roboto:400,300,700'
rel='stylesheet' type='text/css'>
      <link href={% static  "css/bootplus.css" %} rel="stylesheet">
      <link href="//maxcdn.bootstrapcdn.com/font-awesome/4.1.0/css/font-
awesome.min.css" rel="stylesheet">
```

```
      <link href={% static "css/bootplus-responsive.css" %} rel="stylesheet">
      <link href={% static "css/datepicker.css" %} rel="stylesheet">
      <script src={% static "js/jquery-1.11.1.js" %}></script>
      <script src={% static "js/dropdown.js" %}></script>
      <script src={% static "js/bootstrap.js" %}></script>
      <script src={% static "js/bootstrap.min.js" %}></script>
      <script src={% static "js/bootstrap-datepicker.js" %}></script>
      <script src={% static "js/locales/bootstrap-datepicker.el.js" %}></script>

      <!-- Fav and touch icons -->
      <link rel="shortcut icon" href={% static "images/background.png" %} >
  </head>

  <body>
  <nav class="navbar navbar-default" role="navigation">
        <div class="container-fluid">
    <!-- Brand and toggle get grouped for better mobile display -->
    <div class="navbar-header">
       <a class="brand" href="/taxes/">Λογιστικό γραφείο <i class="fa fa-
home"></i></a>
       <div class="nav-collapse collapse">
         <ul class="nav">
           <li><a href="/taxes/tax_office_list.html">Στοιχεία Υπηρεσιών</a></li>
           <li><a href="/taxes/person_list.html">Στοιχεία
Συναλλασσόμενου</a></li>
           <li><a href="/taxes/tables.html">Πίνακας οφειλών</a></li>
           <li class="dropdown"><a href="#" class="dropdown-toggle" data-
toggle="dropdown">Διαγράμματα<span class="caret"></span></a>
              <ul class="dropdown-menu" role="menu" aria-
labelledby="dropdownMenu">
               <li><a href="/taxes/month_charts.html">Εμφάνιση ανά μήνα</a></li>
               <li><a href="/taxes/year_charts.html">Εμφάνιση ανά έτος</a></li>
             </ul>
           </li>
         </ul>
       </div>

{% if user.is_authenticated %}
<p><a href="/taxes/logout.html" class="btn btn-primary btn-large"
class="right">Αποσύνδεση&raquo;</a></p>
{% else %}
```

```
<p><a href="/taxes/login.html" class="btn btn-primary btn-large"
class="right">Συνδέσου εδώ&raquo;</a></p>

{% endif %}
  </div>
  </div>
  </nav>

  <div class="hero-unit">
     <div class="container">
        <h1>Λογιστικό Κέντρο Θεσσαλονίκης</h1>
        <p>Μια πλήρης εφαρμογή για το λογιστικό μας γραφείο </p>
     </div>
  </div>

  {% block content%}

  {% endblock %}

  </div>

  <footer>
    <div class="foot-fixed-bottom">
    <div class="container">
            <p>&copy; Georgia Tsiamanta 2014</p>
    </div>
    </div>
  </footer>


  </body>

</html>
```

taxes/templates/taxes/first_page.html

```
{% extends 'taxes/home.html' %}

{% load static from staticfiles %}
{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />
```

```
{% block content%}

<div class="container">
{% block connected%}
{% endblock %}

<!-- Example row of columns -->
<div class="row">
<div class="span4">

    <div class="card">
       <h3 class="card-heading simple"><i class="fa fa-eur"></i> Όργανώστε τα
οικονομικά σας </h3>
       <div class="card-body">
          <p>Η πλήρης εποπτεία των οικονομικών σας, σας βοηθάει να κάνετε
καλύτερη διαχείριση του προυπολογισμού σας και
          να παίρνετε καλύτερες αποφάσεις. Δείτε σε πινακοποιημένη μορφή τα
έξοδά σας για τον τρέχοντα μήνα και αναζητήστε
          οφειλές παρελθόντων μηνών.
          <p>
       </div>
       <div class="card-actions">
          <a href="/taxes/month_charts.html"  class="btn btn-block">Δείτε
Λεπτομέρειες Εδώ</a>
       </div>
    </div>

</div>
<div class="span4">

    <div class="card">
       <h3 class="card-heading simple"><i class="fa fa-map-marker"></i> Βρείτε
που βρίσκονται οι υπηρεσίες που σας ενδιαφέρουν</h3>
       <div class="card-body">
          <p>Μην χάνετε άσκοπα χρόνο να αναζητάτε διευθύνσεις και τηλέφωνα των
υπηρεσιών που σας ενδιαφέρουν
          <p>
       </div>
       <div class="card-actions">
          <a href="/taxes/tax_office_list.html"  class="btn btn-block">Δείτε
```

```
Λεπτομέρειες Εδώ</a>
       </div>
    </div>
</div>
<div class="span4">
    <div class="card">
       <h3 class="card-heading simple"><i class="fa fa-bar-chart-o"></i> Δείτε
σε διαγράμματα τις οφειλες σας προς τρίτους</h3>
       <div class="card-body">
          <p>Εάν είστε οπτικός τύπος σίγουρα τα διαγράμματα αυτά θα σας
βοηθήσουν πολύ καλύτερα να κατανοήσετε την πορεία των οικονομικών σας
            τόσο σε μηνιαίο όσο και σε ετήσιο επίπεδο.
          <p>
       </div>
       <div class="card-actions">
          <a href="/taxes/month_charts.html"  class="btn btn-block">Δείτε
Λεπτομέρειες Εδώ</a>
       </div>
    </div>

</div>
</div>
{% endblock %}
```

taxes/templates/taxes/tax_office_list.html

```
{% extends 'taxes/home.html' %}

{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/old_style.css' %}"
/>

{% block content%}

{% if latest_tax_office_list %}
<div class="container">
```

```
<div class="row">
<div class="span4">

    <table border="1" style="width:300px">
      <tr class="head">
                <th>Όνομα εφορίας</th>
                  <th>Οδός</th>
                <th>Αριθμός</th>
                <th>Πόλη</th>
            <th>Ταχυδρομικός Κώδικας</th>
            <th>E-mail</th>
            <th>Τηλέφωνο 1</th>
            <th>Τηλέφωνο 2</th>

                </tr>
    {% for sbd in latest_tax_office_list.all %}
        <tr>
                        <td>{{sbd.tax_office_name}}</td>
                        <td>{{sbd.street|default:"-" }}</td>
        <td>{{sbd.str_number|default:"-" }}</td>
        <td>{{sbd.city|default:"-" }}</td>
        <td>{{sbd.tk|default:"-" }}</td>
        <td>{{sbd.email}}</td>
        <td>{{sbd.landline1}}</td>
        <td>{{sbd.landline2|default:"-" }}</td>

                </tr>
    {% endfor %}
    </table>
{% else %}
    <h2>Δεν υπάρχουν καταχωρημένες εφορίες</h2>
{% endif %}
</div>
</div>
</div>
{% endblock %}
```

taxes/templates/taxes/person_list.html

```
{% extends 'taxes/home.html' %}

{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'css/style.css' %}" />


{% block content%}

{% if user.is_authenticated %}
  {% if latest_tax_list %}
    <div class="container">


<!-- Example row of columns -->
    <div class="row">
    <div class="span4">
    <table border="1" style="width:300px">
      <tr class="head">
                <th>Επώνυμο</th>
                  <th>Όνομα</th>
                  <th>Όνομα Επιχείρησης</th>
                <th>Είδος Επιχείρησης</th>
                <th>ΑΦΜ</th>
          <th>Εφορία</th>
          <th>Συνθηματικό</th>
          <th>email</th>
                </tr>
          {% for sbd in latest_tax_list.all %}

        <tr>
                        <td>{{sbd.last_name}}</td>
                        <td>{{sbd.first_name}}</td>
                        <td>{{sbd.company_name|default:"-" }}</td>
                        <td>{{sbd.company_type}}</td>
                        <td>{{sbd.afm}}</td>
                        <td>{{sbd.tax_office_name}}</td>
        <td>{{sbd.my_user}}</td>
```

```
        <td>{{sbd.email}}</td>

                </tr>
      {% endfor %}

    </table>

  {% endif %}
{% else %}
  <div class="container">
    <h2>Για να μπορέσετε να δείτε τα στοιχεία σας συνδεθείτε <a
href="/taxes/login.html">εδώ</a></h2>
  </div>

{% endif %}

</div>
</div>
</div>
{% endblock %}
```

taxes/templates/taxes/views.html

```
{% extends 'taxes/home.html' %}

{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />
<meta name="viewport" content="width=device-width, initial-scale=1">

{% block content%}
<div class="container" >
{% if user.is_authenticated %}
<table>
    <tr>
    <td valign="top">
                    <h3>
                        <span class="fa fa-table"></span> Πίνακες
```

```
                    </h3>
                    <form method="POST" action="/taxes/tables.html/">{%
csrf_token %}
                        <label><h4>Είδος Οφειλής:</h4></label>
                        <select name="eidos1" id="eidos1">
                            <option value="" selected="selected">ΟΛΑ</option>
                        {% for f in debt_list %}
                            <option >{{f.debt_el}}</option>
                        {% endfor%}

                        </select>
                        <br />

                        <label><h4>Ημερομηνία Λήξης Οφειλής:</h4></label>
                        <label for="ofeiles">Από:</label>
                        <div class="input-append date" name="deadline1"
id="deadline1" data-date="01-01-2014" data-date-format="dd-mm-yyyy">
                            <input id="deadline1"  name="deadline1"
class="span2" size="16" type="text" value="" />
                            <span class="add-on"><i class="fa fa-th"></i></span>
                        </div>

                        <script type="text/javascript">

                        $('#deadline1').datepicker({
                            weekStart: 1,
                            todayBtn: "linked",
                            language: "el",
                            autoclose: true,
                            todayHighlight: true,
                            orientation: "top"
                        });
                        </script>

                        <label>Έως:</label>
                        <div class="input-append date" name="deadline2"
id="deadline2" data-date="15-09-2014" data-date-format="dd-mm-yyyy">
                            <input id="deadline2"  name="deadline2"
class="span2" size="16" type="text" value="" />
                            <span class="add-on"><i class="fa fa-th"></i></span>
                        </div>
```

```
                        <script type="text/javascript">

                        $('#deadline2').datepicker({
                            weekStart: 1,
                            todayBtn: "linked",
                            language: "el",
                            autoclose: true,
                            todayHighlight: true,
                            orientation: "top"
                        });
                        </script>

                        <br />
                        <input type="submit" value="Επιλογή" class="btn btn-
primary" />
                    </form>
                    </div>

    </td>

{% if latest_eidos_list %}
<td>
    <table border="1" style="width:400px">
    <tr class="head">
        <th>Στοιχεία Οφειλέτη</th>
        <th>Είδος Οφειλής</th>
        <th>Ποσό Οφειλής</th>
        <th>Πληρωτέο έως</th>
        </tr>

      {% for sbd in latest_eidos_list.all %}
        <tr>
        <td>{{sbd.last_name}}</td>
        <td>{{sbd.debt_el}}</td>
                <td>{{sbd.amount|default:"-" }} €</td>
                <td>{{sbd.deadline}}</td>
                </tr>
      {% endfor %}

    </table>
```

```
</td>
{% else %}
    <td>
        <h3>Δεν βρέθηκαν αποτελέσματα για τα επιλεγμένα κριτήρια</h3>
    </td>
</td>
</tr>
</table>
{% endif %}

{% else %}
  <div class="container">
    <h2>Για να μπορέσετε να δείτε τα στοιχεία σας συνδεθείτε <a
href="/taxes/login.html">εδώ</a></h2>
  </div>
{% endif %}
</div>
{% endblock %}
```

taxes/templates/taxes/month_charts.html

```
{% extends 'taxes/home.html' %}

{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />
<link href="//netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap-
glyphicons.css" rel="stylesheet">

{% block content%}
<div class="container">

{% if user.is_authenticated %}
<script src="http://code.highcharts.com/highcharts.js"></script>
<table>
    <tr>
    <td>
     <form method="POST" action="/taxes/month_charts.html/">{% csrf_token %}
```

```
        <label>Επιλέξτε μήνα:</label>
            <div class="input-append date" name="imerominia" id="imerominia"
data-date-minviewmode="months" data-date-viewmode="years" data-date-format="mm-
yyyy" data-date="09-2014">
                <input id="imerominia"  name="imerominia" class="span2"
size="16" type="text" value="" />
                <span class="add-on"><i class="fa fa-th"></i></span>
            </div>

            <script type="text/javascript">
                $('#imerominia').datepicker({
                    language: "el",
                    startView: 1,
                    minViewMode: 1,
                    autoclose: true
                    });
            </script>

                        <br />
                        <input type="submit" value="Επιλογή" class="btn btn-
primary" />
                    </form>
        </td>
<td>
    <div id="container" style="width:50%; height:300px;"></div>
<head>
    <!-- code to include the highcharts and jQuery libraries goes here -->
    <!-- load_charts filter takes a comma-separated list of id's where -->
    <!-- the charts need to be rendered to                             -->
    {% load chartit %}
    {{ my_chart|load_charts:"container" }}
</head>
<body>
    <div id='container'></div>
</body>
</td>
</tr>
</table>


{% else %}
```

```
 <div class="container">
    <h2>Για να μπορέσετε να δείτε το διάγραμμα των οφειλών σας συνδεθείτε <a
href="/taxes/login.html">εδώ</a></h2>
 </div>

{% endif %}
{% endblock %}
```

taxes/templates/taxes/year_charts.html

```
{% extends 'taxes/home.html' %}

{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />
<link href="//netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap-
glyphicons.css" rel="stylesheet">

{% block content%}
<div class="container">



{% if user.is_authenticated %}
<script src="http://code.highcharts.com/highcharts.js"></script>
<table >
    <tr>
    <td>
    <form method="POST" action="/taxes/year_charts.html/">{% csrf_token %}
        <label>Επιλέξτε έτος:</label>
            <div class="input-append date" name="imerominia2" id="imerominia2"
data-date-minviewmode="years" data-date-viewmode="years" data-date-
format="yyyy">
                <input id="imerominia2"  name="imerominia2" class="span2"
size="16" type="text" value="" />
                <span class="add-on"><i class="fa fa-th"></i></span>
            </div>

            <script type="text/javascript">
                $('#imerominia2').datepicker({
```

```
                    language: "el",
                    startView: 2,
                    minViewMode: 2,
                    autoclose: true
                    });
            </script>

                        <br />
                        <input type="submit" value="Επιλογή" class="btn btn-
primary" />
                    </form>
        </td>
<td>
    <div id="container" style="width:50%; height:300px;"></div>
<head>
    <!-- code to include the highcharts and jQuery libraries goes here -->
    <!-- load_charts filter takes a comma-separated list of id's where -->
    <!-- the charts need to be rendered to                             -->
    {% load chartit %}
    {{ my_chart2|load_charts:"container" }}
</head>
<body>
    <div id='container'></div>
</body>
</td>

<td align="left">
<span class="label label-inverse" style="font-size:15px">1:ΟΤΕ </span><br/><br/>
<span class="label label-success" style="font-size:15px">2:ΔΕΗ</span><br/><br/>
<span class="label label-important" style="font-size:15px">3:ΦΟΡΟΣ
ΕΙΣΟΔΗΜΑΤΟΣ</span><br/><br/>
<span class="label label-inverse" style="font-size:15px">4:ΦΠΑ</span><br/><br/>
<span class="label label-info" style="font-size:15px">5:ΤΕΒΕ</span><br/><br/>
<span class="label label-warning" style="font-size:15px">6:ΙΚΑ</span><br/><br/>
<span class="label label-important" style="font-
size:15px">7:ΕΡΜΑΝΣΗ</span><br/><br/>
<span class="label label-success" style="font-size:15px">8:ΑΛΛΕΣ
ΟΦΕΙΛΕΣ</span><br/><br/>

</td>
</tr>
```

```
</table>

{% else %}

<table>
    <tr>
    <td>
 <div class="container">
    <h2>Για να μπορέσετε να δείτε το διάγραμμα των οφειλών σας συνδεθείτε <a
href="/taxes/login.html">εδώ</a></h2>
 </div>
</table>

{% endif %}

{% endblock %}
```

taxes/templates/taxes/login.html

```
{% extends 'taxes/home.html' %}
{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />

{% block content%}

{% if user.is_authenticated %}
<div class="container">
    <body>
        <h1>apodindesi</h1>
    </body>

</div>

{% else %}

<div class="container">
<div class="row">
<div class="span4">
    <table>
```

```
    <tr>
    <td>
    <body>
        <h1>Σύνδεση χρήστη</h1>

        <form id="login_form" method="post" action="/taxes/auth/">{% csrf_token
%}
            Όνομα Χρήστη: <input type="text" name="username" value="" />
            <br />
            Συνθηματικό: <input type="password" name="password" value="" />
            <br />
            <input type="submit" value="Σύνδεση" class="btn btn-primary" />
        </form>

    </body>
    </td>
    </tr>
    </table>
</div>
</div>
</div>

{% endif %}
{% endblock %}
```

taxes/templates/taxes/loggedin.html

```
{% extends 'taxes/first_page.html' %}
{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />

{% block connected%}
        <h1>Συνδεθήκατε επιτυχώς</h1>
{% endblock %}
```

taxes/templates/taxes/logout.html

```
{% extends 'taxes/first_page.html' %}

{% load static from staticfiles %}

{% load staticfiles %}
<link rel="stylesheet" type="text/css" href="{% static 'taxes/style.css' %}" />

{% block connected%}
        <h1>Αποσυνδεθήκατε επιτυχώς</h1>
{% endblock %}
```

Βιβλιογραφία

[1] http://www.ubuntu.com/

[2] http://el.wikipedia.org/wiki/Ubuntu

[3] http://ubuntu-gr.org/content/%CF%84%CE%B9-%CE%B5%CE%AF%CE%BD%CE%B1%CE%B9-%CF%84%CE%BF-ubuntu

[4] http://sublime-text-unofficial-documentation.readthedocs.org/en/latest/index.html

[5] http://el.wikipedia.org/wiki/Python

[6] http://en.wikipedia.org/wiki/Python_%28programming_language%29

[7] https://foss.ntua.gr/wiki/index.php/%CE%95%CE%B9%CF%83%CE%B1%CE%B3%CF%89%CE%B3%CE%AE_%CF%83%CF%84%CE%B7%CE%BD_Python

[8] https://docs.python.org/2.7/

[9] http://en.wikipedia.org/wiki/Pip_%28package_manager%29

[10] https://www.djangoproject.com/

[11] http://en.wikipedia.org/wiki/Django_%28web_framework%29

[12] http://deltahacker.gr/php-mysql-tutorial-the-fellowship-of-the-ring/

[13] http://www.mysql.com/

[14] http://chartit.shutupandship.com/

[15] http://jquery.com/

[16] http://www.highcharts.com/

[17] http://pacific.jour.auth.gr/html/

[18] http://el.wikipedia.org/wiki/HTML

[19] http://pages.cs.aueb.gr/courses/epl131/files/CSS_notes.pdf

[20] http://el.wikipedia.org/wiki/CSS

[21] http://en.wikipedia.org/wiki/Bootstrap_%28front-end_framework%29

[22] http://getbootstrap.com/

[23] http://el.wikipedia.org/wiki/Git_%28%CE%BB%CE%BF%CE%B3%CE%B9%CF%83%CE%BC%CE%B9%CE%BA%CF%8C%29

[24] http://git-scm.com/

[25] http://yalantis.com/blog/lightweight-ios-view-controllers-separate-data-sources-guided-mvc/

[26] http://en.wikibooks.org/wiki/Computer_Science_Design_Patterns/Model%E2%80%93view%E2%80%93controller

[27] http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller