Πανεπιστήμιο Πειραιά
ΠΜΣ "Πληροφορική"

Ανάπτυξη Εφαρμογής για παραμετροποίηση δικτύου με το Django Framework

Διπλωματικη Εργασια

των

Ιάσονας Σιμώτας, Μαρία Μπαμπανέλου

Επιβλέπων: Πάνος Γκοτσιόπουλος
Καθηγητής ΠΑΠΕΙ
Επιβλέπων: Δουληγέρης Χρήστος
Καθηγητής ΠΑΠΕΙ

Αθήνα, Μήνας Έτος

Πανεπιστήμιο Πειραιά
ΠΜΣ "Πληροφορική"
Τομέας

Ανάπτυξη Εφαρμογής για παραμετροποίηση δικτύου με το Django Framework

Διπλωματικη

των

Ιάσονας Σιμωτας, Μαρία Μπαμπανέλου

Επιβλέπων: Πάνος Γκοτσιόπουλος
Καθηγητής ΠΑΠΕΙ
Επιβλέπων: Δουληγέρης Χρήστος
Καθηγητής ΠΑΠΕΙ

Εγκρίθηκε από την κάτωθι τριμελή επιτροπή την 1η Ιανουαρίου 2019.

Όνομα Επώνυμο Καθηγητής	Όνομα Επώνυμο Καθηγητής	Όνομα Επώνυμο Αναπληρωτής Καθηγητής

Ιάσονας Σιμώτας, Μαρία Μπαμπανέλου
Πτυχιούχοι Μεταπτυχιακού ΠΜΣ Πληροφορικής

Περίληψη

Περίληψη διπλωματικής εργασίας.

Η παρουσα διπλωματική εργασία έχει σκοπό τη μελέτη και την ανάπτυξη λογισμικού για την παραμετροποίηση δικτυακών συσκευών με βασικό Framework το Django της Python. Η ιδέα για την υλοποίηση αυτής της εφαρμογής στάθηκε το υπόβαθρό μας στο software development ,στο χώρο του networking και με αφορμή μια εφαρμογή ενός μηχανικού της Cisco αποφασίσαμε να πάρουμε σαν βάση αυτήν και κάποιες ακόμα δημοσιεύσεις στο χώρο και να τις πάμε ένα βήμα παρακάτω . Σκοπός μας είναι να εισάγουμε τις ένοιες του microservices kubernetes και να μπορέσουμε να φτιάξουμε μια εφαρμογή η οποία περιλαμβάνει γνώσεις γύρω από διάφορους τομείς της Πληροφορικής. Εκτενής βιβλιογραφία θα παρουσιαστεί στο τέλος της διπλωματικής εργασίας.

Abstract

Diploma thesis abstract.

Keywords: Keyword 1, Keyword 2

Ευχαριστίες

Ευχαριστούμε πολύ όσους μας στήριξαν που ενώ παράλληλα με τη δουλειά μας μπορέσαμε να τελειώσουμε και τη διπλωματική εργασία μας. Παρόλο που η δουλειά μας μας βοήθησε σε πολλά επίπεδα στην κατανόηση πολλών εννοιών η φύση της ήταν τέτοια που δε μας έδινε χρόνο στο να κάτσουμε και να τελειώσουμε την εργασία αυτή.

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1

Εισαγωγή

Φαίνεται ότι ο χρόνος που σπαταλάνε οι μηχανικοί δικτύωσης όταν εισέρχονται σε εξοπλισμό δικτύωσης για να εισάγουν χειροκίνητα εντολές οπως για τη διαμόρφωση των συσκευών ή να εισέρχονται σε διακομιστές για τη χειροκίνητη ρύθμιση μία προς μία μια λίστα συσκευών/δικτύων(access lists) είναι πολύ μεγάλος, συνεπώς η εποχή που όλα αυτά γινόντουσταν χειροκίνητα φτάνει στο τέλος της. Όλο και περισσότεροι/περισσότερες εταιρείες προωθούν την αυτοματοποίηση καθώς βλέπουν ότι κάθε ώρα που επενδύεται στην αυτοματοποίηση μεταφράζεται σε πολλές ώρες εργασίας που εξοικονομούνται.

Η αυτοματοποίηση αυτών των εργασιών με κάποια καλοφτιαγμένη λογική προγραμματισμού επιτρέπει τη διαμόρφωση εκατοντάδων συσκευών μέσα σε λίγα λεπτά, απομακρύνει τη δυνατότητα των λανθασμένων ρυθμίσεων που προέρχονται από ανθρώπινο λάθος, επιτρέπει την καταγραφή των αλλαγών διαμόρφωσης και έχει το πλεονέκτημα ότι καθιστά τη διαμόρφωση τεχνικές λεπτομέρειες διαφανείς στον χρήστη που πρόκειται να ξεκινήσει τη διαδικασία αυτοματοποίησης. Για παράδειγμα, μια εταιρεία θα μπορούσε να αναθέσει υπεργολαβικά σε μια ομάδα λειτουργίας που δεν έχει τεχνικές γνώσεις δικτύωσης και απλά παρέχοντάς τους μια συγκεκριμένο σύνολο εισόδων θα μπορούσαν να διαμορφώσουν για X σημεία πρόσβασης σε Y δίκτυα ένα συγκεκριμένο ΣΣΙΔ με τις επιθυμητές παραμέτρους. Οι δεδομένες είσοδοι θα μπορούσαν να εισαχθούν από αυτούς σε μια εφαρμογή ιστού και ο υποκείμενος προγραμματισμός κώδικας θα έκανε τα υπόλοιπα. Τελικά αυτό μεταφράζεται σε ένα πολύ γρήγορο και αξιόπιστο πλάνο κατά το οποίο η παραμετροποίηση και η διαμόρφωση ενός δικτύου δεν θα χρειάζεται να γίνει από τους μηχανικούς δικτύωσης χειροκίνητα.

Μπορούν να επενδύσουν συνεπώς αυτόν τον επιπλέον χρόνο σε άλλες εργασίες όπως ο σχεδιασμός και έτσι οι πιο χρονοβόρες διαδικασίες να αυτοματοποιηθούν. Αλλά η αυτοματοποίηση δεν είναι μόνο κάνει θαύματα όσον αφορά τη διαμόρφωση, είναι επίσης εξαιρετική για την παρακολούθηση της κατάστασης των δικτύων/συσκευών/θυρών, την απόκτηση πληροφοριών για την υγεία των ασύρματων δικτύων και κάθε άλλες πληροφορίες που μπορούν να λάβουν από της δυκτυακές συσκευές.

Είναι σημαντικό να ληφθεί υπόψη ότι οι επαναλαμβανόμενες καθημερινές/εβδομαδιαίες εργασίες που απαιτούν τη συλλογή πληροφοριών είναι εξαιρετικοί υποψήφιοι για αυτοματοποίηση. Με μια αυτοματοποίηση που αναζητά τα απαιτούμενα δεδομένα και κάνει κάποια επεξεργασία οι απαιτούμενες πληροφορίες μπορούν να ληφθούν γρήγορα και να παρουσιαστούν στους μηχανικό και τον/την απαλλάσσει από το να συνδέεται χειροκίνητα σε πολλές συσκευές, να ελέγχει ορισμένων γραμμών διαμόρφωσης, κ.λπ.

Η αυτοματοποίηση συσκευών χρησιμοποιείται εδώ και πολλά χρόνια για τη διαχείριση βλαβών ή την παρακολούθηση του επιπέδου υπηρεσιών, αλλά με τις αυξανόμενες επιχειρηματικές ανάγκες προκύπτουν νέες προκλήσεις και νέες ευκαιρίες. Μία από αυτές τις ευκαιρίες είναι η επένδυση των εταιρειών σε αυτοματισμούς δικτύων. Η ραγδαία ανάπτυξη των σύγχρονων δικτύων στις επιχειρήσεις μαζί με τις νέες τεχνολογίες, όπως το Διαδίκτυο των πραγμάτων (IOT) και το υπολογιστικό νέφος που βασίζονται επίσης στο δίκτυο, οδήγησαν στην ανάγκη ανάπτυξης της δικτυακής υποδομής με αποτέλεσμα την αύξηση του φόρτου εργασίας. απαιτήσεις για την παροχή, τη συντήρηση, την παρακολούθηση και τη διαχείριση από το δίκτυο.

Οι μέθοδοι που χρησιμοποιούσαν μέχρι σήμερα οι μηχανικοί δικτύων δεν ήταν μόνο χρονοβόρες αλλά και απαιτούνταν και γνώσεις σχετικά με ιδιόκτητα πρωτόκολλα και τεχνολογίες. Σε μια προσπάθεια να μειώσουν το κόστος και να δημιουργήσουν αποτελεσματικότητα οι μηχανικοί δικτύου ανέπτυξαν το Network automation ως τεχνικές αυτοματοποίησης για την αυτοματοποίηση επαναλαμβανόμενων καθημερινών εργασιών. Με την υποστήριξη σχεδόν όλων των μεγάλων εταιρειών δικτύωσης (όπως η Cisco) δημιουργήθηκε μια κοινότητα ανοιχτού κώδικα που είχε ως στόχο την υλοποίηση εφαρμογών αυτοματοποίησης κυρίως με τη χρήση τυποποιημένων διεπαφών (SSH, REST) και γενικών γλωσσών προγραμματισμού όπως η python. Με τη χρήση της Python και μιας συλλογής ενοτήτων και συναρτήσεων θα προσπαθήσουμε να φτιάξουμε μία εφαρμογή που συνδέει όλα τα παραπάνω.

Παράλληλα η επανάσταση που έφερε η εισαγώγή της λογικής των microservices στον κλαδο της Πληροφορικής μπορεί να καθιστήσει την εφαρμογή αυτή ακόμα πιο αξιόπιστη γιατί μπορεί να συμβάλει στο σχεδίασμό ενός συστήματος λογισμικού με μεγαλύτερη αξιοπιστία καθώς και να προσφέρει όλες εκείνες της θετικές προεκτάσεις χρήσης αυτών. Θα γίνει λοιπόν μια προσπάθεια εισαγώγής τεχνολογιών διαχείρισης και ανάπτυξης microservices όπως kubernetes και containers. Τα πλεονεκτήματα της χρήσης της αρχιτεκτονική Microservices είναι ότι προσφέρουν μεγαλύτερη ευελιξία μέσω της ανεξαρτησίας των υπηρεσιών, επιτρέποντας στους οργανισμούς να γίνουν πιο ευέλικτοι όσον αφορά τον τρόπο με τον οποίο προσφέρουν νέες επιχειρηματικές δυνατότητες ή ανταποκρίνονται στις μεταβαλλόμενες συνθήκες της αγοράς. Αναλυτική παρουσίαση αυτών θα γίνει σε επόμενο κεφάλαιο.

Σας συσκευές θα χρησιμοποιήσουμε αυτές της Cisco καθώς υπάρχουν ήδη βιβλιοθήκες οι οποίες υλοποιούν τα πρωτόκολλα επικοινωνίας και τις λειτουργίες που εμείς θέλουμε να υλοποιήσουμε. Η ανάπτυξη λογισμικού τέτοιων βιβλιοθηκών είναι

αντικείμενο μελέτης διπλωματικής εργασίας καθώς ξεφεύγει από τα πλάισια μια μεταπτυχιακής διατριβής.

1.1 Απαιτήσεις και προδιαγραφές

- GNS3 VM ,Cisco Images και GNS3 περιβάλλον
- Vs Code development environment
- Virtual box ή οποιονδήποτε type B hypervisor

Κεφάλαιο 2

State of the art

2.1 Η επανάσταση στο web development

Στην αρχή, οι εφαρμογές ιστού δεν ήταν τίποτα περισσότερο από ένα μάτσο HTML, CSS και javascript που ήταν τοποθετημένα μαζί, συνδεδεμένα μεταξύ τους. Ένας καλός προγραμματιστής ήταν σε θέση να φτιάξει σπουδαίες εφαρμογές ιστού αν αυτός/αυτή είχε αρκετές δεξιότητες/γνώσεις.

Στην εποχή μας, εμφανίστηκαν τα frameworks και λαμβάνοντας υπόψη ότι δεν βελτιώνουν αυτό που τελικά βλέπει ο χρήστης και ο/η αλληλεπιδράσεις του με το frontend που τελικά είναι ο τελικός στόχος, τότε θα μπορούσε κανείς να αναρωτηθεί γιατί χρησιμοποιούνται ευρέως στις μέρες μας. Παρόμοιες δουλειές υπάρχουν και σε άλλες διπλωματικές εργασίες καθώς και σε μη διπλωματικές εργασίες. Μηχανικοί από όλο τον κόσμο ασχολούνται με την αυτοματοποίηση συστημάτων και τη δημιουργία κώδικα που να αυτοματοποιεί συσκευές/συστήματα.

Με βάση άλλες τέτοιες προσπάθειες που έχουν γίνει στο παρελθόν εμείς συλλέξαμε την εως τώρα βιβλιογραφία και θα προσπαθήσουμε να φτιάξουμε μία τέτοια εφαρμογή η οποία όμως να βασίζεται στα τωρινά δεδομένα και να ενσωματσώσουμε τις τελευταίες τεχνολογίες αιχμής όπως το Cloud Native. Θα γίνει προσπάθεια να δωθεί Εκτενής ανάλυση στο πως λειτουργεί η εφαρμογή καθώς και η αλληλλεπίδραση της με τα συστήματα.

2.2 Frameworks και γιατί χρησιμοποιούνται

Στην τωρινή επόχή η ανάπτυξη λογισμικού είναι στενά συνδεδεμένη με τα frameworks. Η σύνδεση αυτή δεν είναι τυχαία καθώς η χρήση αυτών έχει κάνει τη ζωή των μηχανικών ανάπτυξης λογισμικού ευκολότερη. Θα εξηγήσουμε παρακάτω τους λόγους που συμβαίνει αυτό στο πλάισιο κυρίως της δικιάς μας διπλωματικής εργασίας.

- Modularity Καθώς η εφαρμογή μεγαλώνει, ο κώδικας πρέπει να είναι καλά δομημένος σε φακέλους και αρχεία ανάλογα με το τι κάνει ο κώδικας. Στο πα-

Σχήμα 2.1: Γενική αρχιτεκτονική του Django

ρελθόν, οι μεγάλες εφαρμογές υπέφεραν όταν η εφαρμογή μεγάλωνε, υπήρχαν προβλήματα επεκτασιμότητας καθώς ο αριθμός των αρχείων javascript και CSS αυξανόταν ραγδαία και υπήρχαν πολύς επαναλαμβανόμενος κώδικας μεταξύ των αρχείων. Με την παροχή μιας καθορισμένης δομής, ένα συγκεκριμένο κομμάτι κώδικα μπορεί να αναζητηθεί εύκολα. Αν πάρουμε ως παράδειγμα παράδειγμα το πλαίσιο Django, το Django δομεί τον κώδικα σε ένα πολύ συγκεκριμένο τρόπο. Μέσα στο αρχείο μοδελς.πψ ορίζονται τα μοντέλα της βάσης δεδομένων. Αυτό συμβαίνει προκειμένου να μπορούν να γίνουν ερωτήματα στη βάση δεδομένων που δεν σχετίζονται με τη συγκεκριμένη βάση δεδομένων που χρησιμοποιείται στο έργο. Μέσα στο αρχείο views.py γίνεται η λογική για την ανάκτηση και την επεξεργασία των δεδομένων όταν το ζητάει ο χρήστης. Μέσα στο αρχείο urls.py υλοποιείται η δρομολόγηση της εφαρμογής . Μέσα στο φάκελο templates υπάρχουν όλα τα .html αρχεία στα οποία το αρχείο views.py στέλνει τα δεδομένα που λαμβάνει για να τα απεικονίσει, κ.λπ.

- Ταχύτερη ανάπτυξη Τα πλαίσια παρέχουν έτοιμες λειτουργίες, καλώντας απλώς τις ήδη ενσωματωμένες συναρτήσεις/μεθόδους. Διαβάζοντας απλώς την τεκμηρίωση του πλαισίου και μαθαίνοντας πώς να τις χρησιμοποιεί, ο προγραμματιστής μπορεί να να ενσωματώσει λειτουργικότητες που διαφορετικά θα ήταν δύσκολο να υλοποιήσει και επίσης πολύ χρονοβόρες. Παραδείγματα περιλαμβάνουν λειτουργίες ελέγχου ταυτότητας, λειτουργίες διαχείρισης συνεδριών, λειτουργίες λειτουργίας βάσεων δεδομένων, λειτουργίες επικύρωσης φορμών και λειτουργίες για την παροχή ασφάλειας έναντι κακόβουλων επιτιθέμενων.

- Επέκταση κώδικα Τα περισσότερα frameworks επιτρέπουν την επέκταση κάποιου κομματιού κώδικα που θα χρησιμοποιηθεί σε πολλά άλλα αρχεία. Αυτό εξασφαλίζει ότι δεν υπάρχει επαναλαμβανόμενος κώδικας και οποιαδήποτε αλλαγή σε αυτόν τον κώδικα μεταφράζεται σε όλες τις περιπτώσεις που χρησιμοποιούν αυτόν τον κώδικα.

- Ευκολότερη αναγνωσιμότητα του κώδικα Δεδομένου ότι ο κώδικας χρησιμοποιεί καλά καθορισμένες τυποποιημένες συναρτήσεις και μια συγκεκριμένη δομή, είναι ευκολότερο να τον καταλάβει κάποιος που είναι νέος στο κώδικα αλλά γνωρίζει πώς λειτουργεί το πλαίσιο

2.3 CI/CD pipeline

Μόλις η εφαρμογή ιστού συνδεθεί με το απομακρυσμένο αποθετήριο, η τελευταία τάση στο στον κόσμο του DevOps είναι η υλοποίηση ενός αγωγού CI/CD, ο οποίος ουσιαστικά είναι μια αυτοματοποιημένη διαδικασία που ενεργοποιείται όταν νέος κώδικας δημοσιεύεται στο απομακρυσμένο αποθετήριο. Αυτή η διαδικασία ξεκινάει τη δημιουργία κώδικα, εκτελεί κάποιες δοκιμές και τέλος, αν όλα είναι εντάξει, αναπτύσσει αυτόματα τον κώδικα στην παραγωγή περιβάλλον. Με αυτόν τον τρόπο, οι προγραμματιστές μπορούν να διασφαλίσουν ότι τίποτα δεν θα χαλάσει στην παραγωγή και οι νέες λειτουργικότητες εξυπηρετούνται το συντομότερο δυνατό στους πελάτη. Στην περίπτωσής μας τόσο η διπλωματική εργασία(latex) όσο και η εφαρμογή υλοποιήθηκαν με αυτή τη λογική.

2.4 Τι είναι ο kubernetes

Ο κυβερνήτης είναι ο διαχειριστής των με απλά λόγια ο διαχειριστής των containers. Είναι μια πλατφόρμα ανοικτού κώδικα για τη διαχείριση φορτίων εργασίας και υπηρεσιών που περιέχουν containers , η οποία διευκολύνει τόσο τη δηλωτική διαμόρφωση όσο και την αυτοματοποίηση. Διαθέτει ένα μεγάλο, ταχέως αναπτυσσόμενο οικοσύστημα. Οι υπηρεσίες, η υποστήριξη και τα εργαλεία του Kubernetes είναι ευρέως διαθέσιμα.

Κεφάλαιο 3

Τεχνολογιές

3.1 Εισαγωγή

Αρχικά δημιουργήθηκε η βασική δομή και η δομή της διαδικτυακής πύλης καθορίστηκε. Αυτή η βασική δομή φαίνεται στο Παρακάτω σχήμα

Είναι σημαντικό να σημειωθεί ότι ολόκληρη η εφαρμογή Δθανγο βρίσκεται στον ίδιο φυσικό διακομιστή. Όταν ο χρήστης στέλνει ένα αίτημα για την εκτέλεση ενός σεναρίου με συγκεκριμένες εισόδους αυτό στέλνεται στις συσκευές στο τοπικό δίκτυο.

Το πρώτο βήμα στη διαδικασία ήταν να καθοριστεί τι θα αυτοματοποιηθεί με βάση τις διάφορες εκτιμήσεις. Για να καθοριστεί αυτό, πραγματοποιήθηκαν πολλές συναντήσεις καταιγισμού ιδεών. με την ομάδα. Προτού γίνει αυτό όμως η αρχική ιδεά που τέθηκε στο τραπέζι βγήκε με βάση μια παρόμοια δουλειά ενός μηχανικού της Cisco. Το έργο του θα αναφερθεί αναλυτικά στην εκτενή βιβλιογραφία στο τέλος. Με βάση λοιπόν αυτο το έργο ξεκινήσαν συζητήσεις για το πως θα μπορέσουμε να αναπτύξουμε κάτι παρόμοιο καθώς και να το εμπλουτίσουμε στο τέλος έτσι ώστε να αντοπρίνεται όσο γίνεται στις τεχνολογίες του σήμερα.

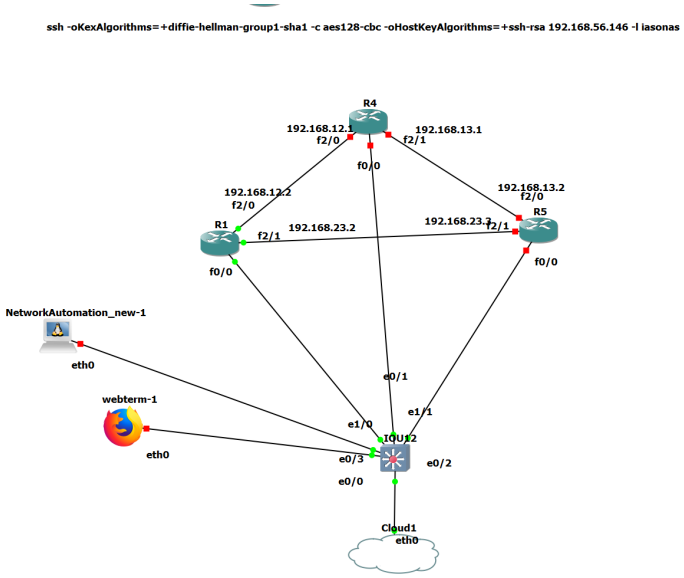
Σε αυτές τις συναντήσεις που γίνανε μεταξύ μας τέθηκαν πολλές ιδέες τέθηκαν στο τραπέζι και η ομάδα καθόρισε μια σειρά προτεραιότητας για την ανάπτυξη. Ο στόχος σε πολλούς αυτοματισμούς είναι να μειωθεί ο χρόνος που καταναλώνεται για την εκτέλεση αυτές τις επαναλαμβανόμενες εργασίες. Πολλές τεχνολογίες χρησιμοποιήθηκαν για την υλοποίηση του συγκεκριμένου έργου οι οποίες θα παρουσιαστούν εκτενώς σε άλλες ενότητες.

Η υλοποίησή μιας τέτοιας εφαρμογής είχε κάποιες δυσκολίες. Κυρίως ποιο θα είναι το περιβάλλον στο οποίο η εφαρμογή θα μπορούσε να τεσταριστεί και υλοποιηθεί. Για αυτό πάρθηκε η απόφαση οι συσκευές με τις οποίες θα τεσταριστεί και συνάμα θα λειτουργήσει η εφαρμογή θα είναι virtual συσκευές της Cisco οι οποίες θα τρέχουν στο GNS3 και το GNS3 θα μπορεί να επικοινωνεί δικτυακά με τον Django Server στο τοπικό δίκτυο. Το στήσιμο όλου του περιβάλλοντος και της εφαρμογής θα αναλυθεί εκτενώς περαιτέρω σε άλλο κεφάλαιο.

3.2 GNS3

Το GNS3 Είναι ένα εργαλείο προσομοίωσης δικτύων ανοικτού κώδικα που επιτρέπει στους χρήστες να προσομοιώσουν σύνθετες τοπολογίες δικτύων στους υπολογιστές τους. Μηχανικοί δικτύων και φοιτητές το χρησιμοποιούν ευρέως για να μάθουν και να εξασκηθούν σε έννοιες δικτύωσης, να δοκιμάσουν διαμορφώσεις δικτύου και να δημιουργήσουν εικονικά περιβάλλοντα δικτύου.

Το GNS3 υποστηρίζει διάφορες συσκευές δικτύου, όπως δρομολογητές, μεταγωγείς και τείχη προστασίας από διάφορους προμηθευτές, συμπεριλαμβανομένων των Cisco, Juniper, Nokia και άλλων. Επιτρέπει στους χρήστες να προσομοιώσουν διάφορα σενάρια και διαμορφώσεις δικτύου και να δοκιμάσουν τη συμπεριφορά των συσκευών δικτύου σε ένα ελεγχόμενο περιβάλλον.



Σχήμα 3.1: General Network Topology

3.3 Cisco IOS

Το IOU σημαίνει IOS on Unix είναι μια εικονική έκδοση του λογισμικού IOS της Cisco που μπορεί να χρησιμοποιηθεί για σκοπούς προσομοίωσης και δοκιμής δικτύου. Επιτρέπει στους μηχανικούς δικτύου να δημιουργούν εικονικές τοπολογίες δικτύου και να εξασκούνται σε διάφορες εργασίες δικτύου, όπως η διαμόρφωση δρομολογητών και μεταγωγέων, χωρίς να απαιτείται φυσικό υλικό. Το πλεονέκτημα του GNS3 σε σχέση με εφαρμογές άλλες όπως το Packet tracer είναι ότι το GNS3 μπορεί να σηκώσει πραγματικά images άρα πραγματικό λογισμικό συνεπώς οι λειτουργίες που μπορείς να κάνεις είναι πολύ περισσότερες.

Το Cisco IOU χρησιμοποιείται συχνά σε συνδυασμό με λογισμικό προσομοίωσης δικτύου όπως GNS3 ή EVE-NG, τα οποία αποτελούν την πλατφόρμα εικονικοποίησης δικτύου που σας επιτρέπει να να δημιουργείτε και να διαχειρίζεστε εικονικά

περιβάλλοντα δικτύου για σκοπούς δοκιμής και εκμάθησης, τα οποία παρέχουν ένα γραφικό περιβάλλον χρήστη για τη δημιουργία και τη διαχείριση εικονικών τοπολογιών δικτύου. Οι εικόνες IOU μπορούν να φορτωθούν σε αυτά τα εργαλεία προσομοίωσης για τη δημιουργία εικονικών συσκευών Cisco που μπορούν να διαμορφωθούν και να δοκιμαστούν όπως το φυσικό δίκτυο συσκευές.

3.4 Εικονικοποίηση

Στην επιστήμη της πληροφορικής, η εικονικοποίηση virtualization είναι ένας ευρύς όρος των υπολογιστικών συστημάτων που αναφέρεται σε έναν μηχανισμό αφαίρεσης, στοχευμένο στην απόκρυψη λεπτομερειών της υλοποίησης και της κατάστασης ορισμένων υπολογιστικών πόρων από πελάτες των πόρων αυτών (π.χ. εφαρμογές, άλλα συστήματα, χρήστες κλπ). Η εν λόγω αφαίρεση μπορεί είτε να αναγκάζει έναν πόρο να συμπεριφέρεται ως πλειάδα πόρων (π.χ. μία συσκευή αποθήκευσης σε διακομιστή τοπικού δικτύου), είτε πολλαπλούς πόρους να συμπεριφέρονται ως ένας (π.χ. συσκευές αποθήκευσης σε κατανεμημένα συστήματα).

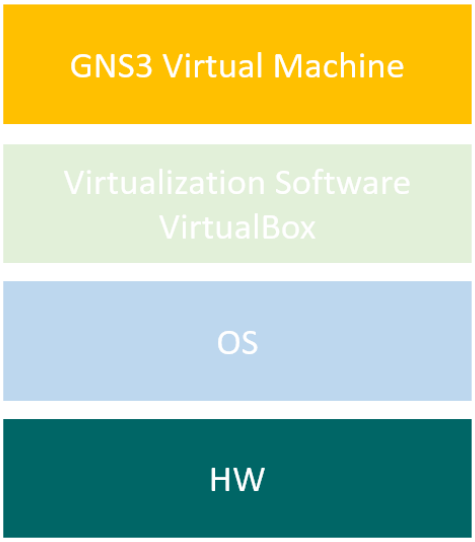
Η εικονικοποίηση δημιουργεί μία εξωτερική διασύνδεση η οποία αποκρύπτει την υποκείμενη υλοποίηση (π.χ. πολυπλέκοντας την πρόσβαση από διαφορετικούς χρήστες). Αυτή η προσέγγιση στην εικονικοποίηση αναφέρεται ως εικονικοποίηση πόρων. Μία άλλη προσέγγιση, ίδιας όμως νοοτροπίας, είναι η εικονικοποίηση πλατφόρμας, όπου η αφαίρεση που επιτελείται προσομοιώνει ολόκληρους υπολογιστές. Το αντίθετο της εικονικοποίησης είναι η διαφάνεια: ένας εικονικός πόρος είναι ορατός, αντιληπτός, αλλά στην πραγματικότητα ανύπαρκτος, ενώ ένας διαφανής πόρος είναι υπαρκτός αλλά αόρατος.

Θα εξηγήσουμε την εικονικοποίηση στην δικιά μας περίπτωση. Το πρώτο επίπεδο είναι αυτό του υλικού. Η εικονικοποίηση σα τεχνολογία εικονοποιεί το υλικό για να μπορέσει να δώσε πόρους στις εικονικές μηχανές. Η υλοιποίηση της εικονικοποίησης γίνεται με λογισμικό hypervisor. Στη δικιά μας περίπτωση ο hypervisor είναι το Virtual Box ο οποίος είναι ένας τύπου Β hypervisor. Ο hypervisor τύπου 2 είναι μια εφαρμογή εγκατεστημένη στο λειτουργικό σύστημα του κεντρικού υπολογιστή το οποίο μας δίνει τη δυνατότητα να σηκώσουμε εικονικές μηχανές άλλων λειτουργικών συστημάτων πάνω στο ήδη υπάρχον σύστημα.

Οι παρακάτω εικόνες μπορούν να εξηγήσουν σχηματικά τη γενική καθώς και την ειδική αρχιτεκτονική.

3.5 Πρόγραμμα εικονοποίσης για το GNS3 VM-VirtualBox

Το Oracle VM VirtualBox ή VirtualBox (πρώην ενΣυν ἴρτυαλΒοξ, Sun xVM VirtualBox και Innotek VirtualBox) είναι υπερεπόπτης ανοιχτού κώδικα για υπολογιστές x86 που αναπτύσσεται από την Oracle Corporation. Αναπτύχθηκε αρχικά από την Innotek GmbH και αποκτήθηκε από τη Sun Microsystems το 2008, η οποία



Σχήμα 3.2: Virtualization Γενική αρχιτεκτονική

εξαγοράστηκε από την Oracle το 2010.

Το VirtualBox μπορεί να εγκατασταθεί σε διάφορα λειτουργικά συστήματα, συμπεριλαμβανόμενων των Linux, macOS, Windows, Solaris και OpenSolaris. Υπάρχουν επίσης μεταφορές για το FreeBSD και το Genode. Υποστηρίζει τη δημιουργία και τη διαχείριση εικονικών μηχανών που εκτελούν εκδόσεις και παραλλαγές των Microsoft Windows, Linux, BSD, Solaris, Haiku, OSx86 και άλλα, καθώς και περιορισμένη εικονικοποίηση macOS. Για ορισμένα λειτουργικά συστήματα είναι διαθέσιμο ένα πακέτο "Guest Additions" από μηχανές συσκευών και εφαρμογές συστήματος που συνήθως βελτιώνει την απόδοση, ειδικά των γραφικών, επίσης δίνει την δηνατότητα στον χρήστη να μεταφέρει αρχεία ή κείμενο από μία εικονική μηχανή στον υπολογιστή του χρήστη και να αυξήσει την ανάλυση του παράθυρου της μηχανή. Στην εικόνα 3.4 μπορούμε να δούμε το VirtualBox και την εικονική μηχανή GNS3 VM

3.6 Django Web Framework

Το Django είναι ένα backedn framework το οποίο βασίζεται στη γλώσσα προγραμματισμού Python. Με το Django, μπορείτε να μεταφέρετε τις εφαρμογές Ιστού από την ιδέα στην κυκλοφορία μέσα σε λίγες ώρες. Το Django φροντίζει για μεγάλο μέρος της ταλαιπωρίας της ανάπτυξης ιστού, ώστε να μπορείτε να εστιάσετε στη σύνταξη της εφαρμογής σας χωρίς να χρειάζεται να ανακαλύψετε ξανά τον τροχό. Είναι δωρεάν και ανοιχτού κώδικα. Ορισμένες από τις πιο μεγάλες εταιρίες στον πλανήτη χρησιμοποιούν την ικανότητα του να κλιμακώνεται γρήγορα και με ευελιξία για να ανταποκρίνεται στις μεγαλύτερες απαιτήσεις κίνησης. Στη δικιά μας περίπτωση χρησιμοποιήθηκε το συγκεκριμένου Φραμεωορκ γιατί θα μας έδινε τη δυνατότητα να φτιάξουμε μία εφαρμογή με μεγάλη επεκτασιμότητα και παράλληλα να μπορέσουμε

Σχήμα 3.3: Virtualization Γενική αρχιτεκτονική

να ενσωματώσουμε μεσα διαφορετικές τεχνολογίες.

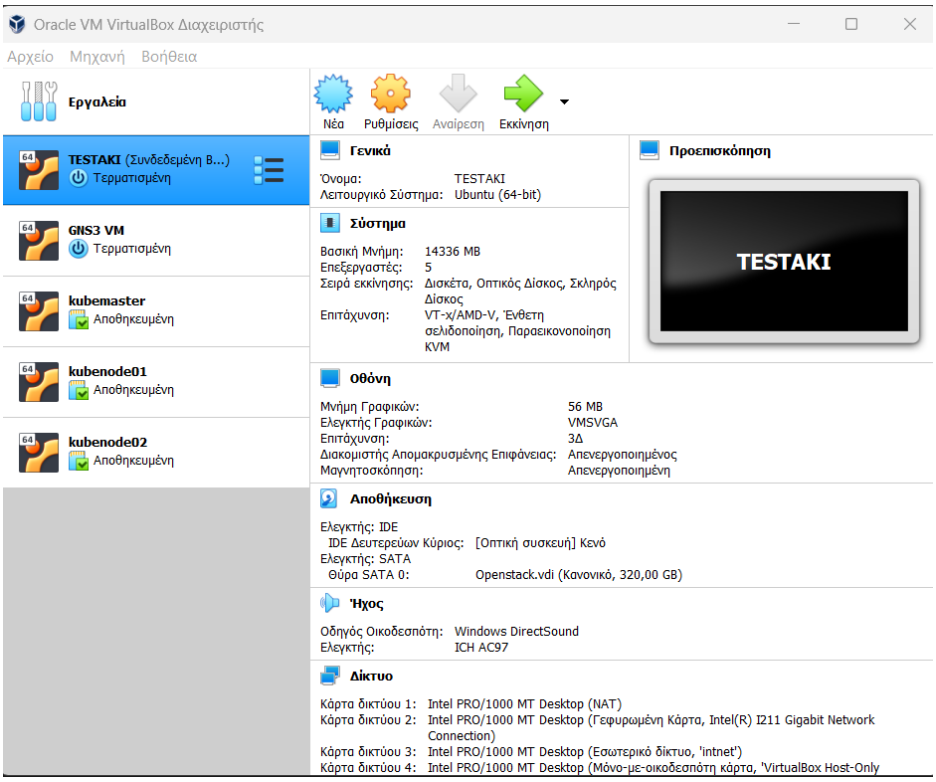
Παράλληλα με το Django χρησιμοποιήθηκαν έτοιμες βιβλιοθήκες της Python προκειμένου να μπορέσουν να εκτελεστούν βασικές λειτουργίες της εφαρμογής όπως τα πρωτοκολλα επικοινωνίας. Θεωρούμε ότι η δημιουργία τέτοιων βιβλιοθηκών ξεφεύγει από τα όρια μιας διπλωματικής εργασίας καθώς απαιτεί πολύ χρόνο και ερευνητική ενασχόληση που μόνο στα πλαίσια ενός διδακτορικού θα μπορούσε να υλοποιηθεί μιά τέτοια ιδέα. Παρακάτων παρουσιάζονται οι βιβλιοθήκες που χρησιμοποιήθηκαν και κάποια βασικά χαρακτηριστικά τους.

3.6.1 Paramiko

Το Paramiko είναι μια διασύνδεση καθαρά Python που υλοποιεί το πρωτόκολλο SSH έκδοσης 2 σε Python, παρέχοντας λειτουργικότητα τόσο πελάτη όσο και διακομιστή. Το Paramiko μπορεί να επιτύχει υψηλές επιδόσεις σε χαμηλού επιπέδου κρυπτογραφικές έννοιες. Οποιαδήποτε συσκευή που μπορεί να ρυθμιστεί μέσω SSH μπορεί επίσης να ρυθμιστεί από την Python με σενάρια με τη χρήση αυτής της μονάδας.

3.6.2 Netmiko

Το Netmiko είναι μια βιβλιοθήκη ανοικτού κώδικα για πολλούς προμηθευτές, που σημαίνει ότι πολλές συσκευές μπορούν να ρυθμιστούν από την python χρησιμοποιώντας το Netmiko. Ορισμένες από τις συσκευές που υποστηρίζει το Netmiko είναι οι εξής: Cisco IOS, Juniper, Arista, HP και Linux. Μπορεί επίσης να υποστηρίζει και άλλους προμηθευτές όπως η Alcatel, η Huawei και η Ubiquity αλλά περιορισμένα



Σχήμα 3.4: Virtualbox

δοκιμές έχουν γίνει με αυτούς τους προμηθευτές. Το Netmiko τρέχει πάνω από το Paramiko για να κάνει τη σύνδεση SSH σε συσκευές δικτύου λιγότερο περίπλοκη, πιο ευέλικτη και πιο εύκολη στη χρήση. Παρόλο που το Netmiko είναι ευκολότερο στη χρήση, όπως αναφέρθηκε παραπάνω, υποστηρίζει συγκεκριμένους προμηθευτές και μόνο έναν αριθμό συσκευών τους. Από την άλλη πλευρά, το Paramiko μπορεί να χρησιμοποιηθεί για την επικοινωνία με οποιαδήποτε συσκευή που υποστηρίζει SSH. Τόσο το Paramikoόσο και το Netmiko αποτελούν εναλλακτικές επιλογές για συσκευές που δεν υποστηρίζουν APIs.

3.6.3 Napalm

Το NAPALM (Network Automation and Programmability Abstraction Layer with Multivendor support) είναι μια βιβλιοθήκη Python που υλοποιεί ένα σύνολο λειτουργιών για την αλληλεπίδραση με διαφορετικά λειτουργικά συστήματα συσκευών δικτύου χρησιμοποιώντας ένα ενοποιημένο API. Το NAPALM υποστηρίζει διάφορες μεθόδους σύνδεσης με τις συσκευές, χειρισμού των ρυθμίσεων ή ανάκτησης δεδομένων.

Κεφάλαιο 4

Virtual Environment Set up

4.1 GNS3 Installation

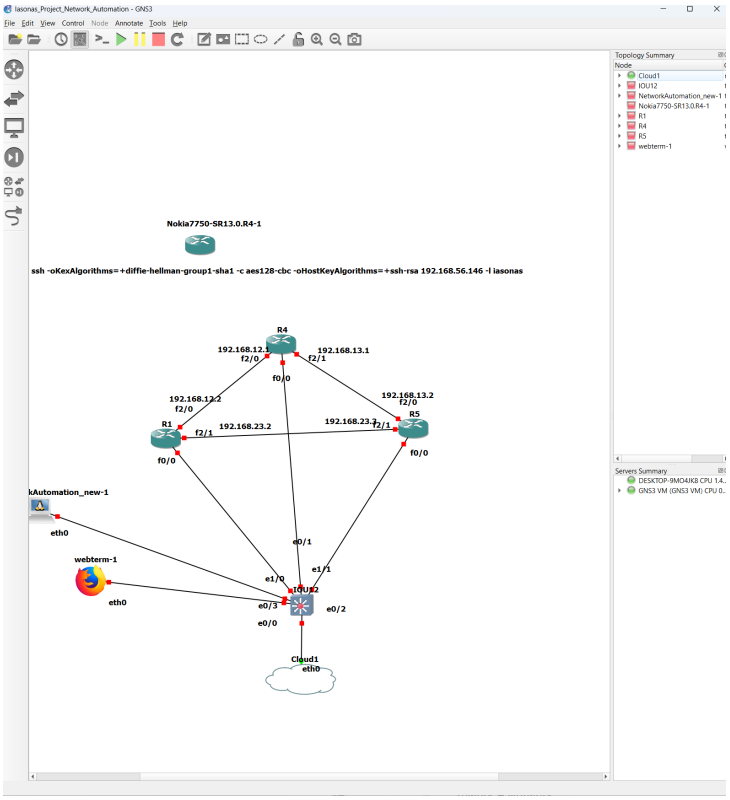
Το GNS3 είναι ένα λογισμικό που χρησιμοποιείται για την εξομοίωση, τη διαμόρφωση και τη δοκιμή ενός περιβάλλοντος δικτύου. Είναι είναι ένα ελεύθερο λογισμικό ανοικτού κώδικα και μπορείτε να το κατεβάσετε από τον επίσημο δικτυακό τόπο https://www.gns3.com/ .Το GNS3 αποτελείται από δύο στοιχεία. Το ολοκληρωμένο λογισμικό (GUI) το οποίο είναι ένα γραφικό διεπαφή χρήστη και την εικονική μηχανή (VM), η οποία είναι ένας διακομιστής που εκτελείται σε εικονικό περιβάλλον και παρέχει καλύτερο μέγεθος τοπολογίας και υποστήριξη συσκευών. Η εγκατάσταση είναι απλή και θα πρέπει να χρησιμοποιούνται οι προεπιλεγμένες επιλογές.

Για να γίνει σωστά η εγκατάσταση θα πρέπει το software version του GNS3 να είναι το ίδιο με το software version του GNS3 VM. Όταν λοιπόν γίνει η εγκατάσταση και ανοίγουμε το GNS3 GUI αυτή η ενέργεια θα κάνει trigger το booting του GNS3 VM Μόλις γίνει η εγκατάσταση μπορεί να ανοίξει η εφαρμογή και να κάνουμε import cisco IOS images. Στην παρακάτω εικόνα μπορούμε να δούμε τι γίνεται όταν ανοίγουμε το GNS3.

Προκειμένου να μπορέσει να επικοινωνήσει το PC μας στο τοπικό δίκτυο με το GNS3 VM στο τοπικό δίκτυο θα πρέπει να γίνουν κάποιες ρυθμίσεις τόσο στο GNS3 VM όσο και στις συσκευές της Cisco

Στις συσκευές της Cisco θα πρέπει να γίνει η παρακάτω παραμετροποίηση όπως εμφανίζεται στις εικόνες 4.1,4.2,4.3.

Μέχρι αυτή τη στιγμή έχουμε παραμετροποιήσει τις συσκευές με τέτοιο τρόπο ώστε να δέχονται απομακρυσμένη σύνδεση. Τώρα θα εξηγήσουμε πως μπορούμε να φτιάξουμε την εποικοινωνία μεταξύ εικονικών μηχανών της Cisco και του τοπικού μας υπολογιστή. Η λογική είναι ότι η συσκευή Cloud θα μας επιτρέψει να φτιάξουμε τη σύνδεση αυτή. Η εικόνα 4.4 μας παρουσιάζει σε ανώτερο επίπεδο τη λογική αυτή σύνδεση.



Σχήμα 4.1: GNS3 homepage

4.2 Connection Establishment

Όταν λοιπόν γίνει αυτή η παραμετροποίηση και τοπολογία θα πρέπει όλα αυτά τα διαφορετικά ϲομπονεντς να ανήκουν στο ίδιο τοπικό δίκτυο. Η εικονική διεπαφή από την οποία θα περνάει όλη η κίνηση είτε μιλάμε για REST ειτε για SSH θα είναι η eth0 στο GNS3 VM. Παρακάτω ένα trace στην εικόνα 4.5 που συλλέχτηκε αποδεικνύει ότι η σύνδεση πραγματοποιείται χωρίς προβλήματα.

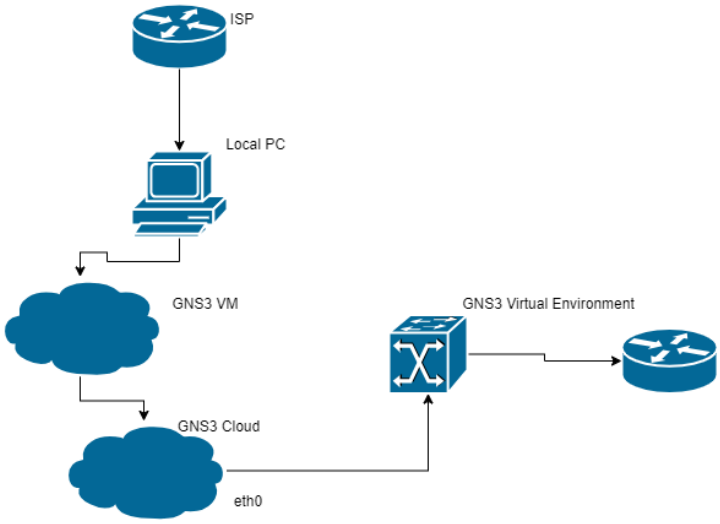
Προκειμένου να γίνει η συλλογή του συγκεκριμένου trace χρησιμοποιήθηκε η παρακάτω εντολή: tcpdump -i eth0 -v -w /home/gns3/test.pcap .Η συλλογή του trace έγινε με το πρωτόκολλο SFTP.

```
line vty 0 4
 login local
 transport input ssh
line vty 5 15
 login local
 transport input ssh
!
```

Σχήμα 4.2: Cisco ssh config

```
interface FastEthernet0/0
 ip address dhcp
 duplex half
!
```

Σχήμα 4.3: Cisco dhcp config



Σχήμα 4.4: Local PC-GNS3VM-CISCO IOS Connection Architecture

226	16.412596	192.168.56.1	192.168.56.146	SSHv2	106	Client: Encrypted packet (len=52)
227	16.419894	192.168.56.146	192.168.56.1	SSHv2	106	Server: Encrypted packet (len=52)
228	16.420665	192.168.56.1	192.168.56.146	TCP	54	50708 → 22 [ACK] Seq=2758 Ack=1764 Win=63877 Len=0
229	16.532536	192.168.56.1	192.168.56.146	SSHv2	106	Client: Encrypted packet (len=52)
230	16.538112	192.168.56.146	192.168.56.1	SSHv2	106	Server: Encrypted packet (len=52)
231	16.538564	192.168.56.1	192.168.56.146	TCP	54	50708 → 22 [ACK] Seq=2810 Ack=1816 Win=63877 Len=0
232	16.657343	192.168.56.146	192.168.56.1	SSHv2	122	Server: Encrypted packet (len=68)
233	16.657392	192.168.56.146	192.168.56.1	SSHv2	90	Server: Encrypted packet (len=36)
234	16.657394	192.168.56.146	192.168.56.1	SSHv2	90	Server: Encrypted packet (len=36)

Σχήμα 4.5: SSH traffic

Σχήμα 4.6: SSH traffic

4.3 Σύνδεση με Django Server

Ο Django Server τρέχει στον τοπικό υπολογιστή. Μπορεί να τρέξει σε οποιοδήποτε μηχάνημα είναι Linux είτε Windows αρκεί να είναι στο τοπικό δίκτυο είτε να υπάρχει κάποια συσκευή layer2 η οποία να αναλάβει τη σύνδεση στο λεγόμενο data link layer.

4.4 Δομή της διαδικτυακής εφαρμογής Django

4.4.1 Τα αρχεία urls.py

Αυτά τα αρχεία καθορίζουν τη δρομολόγηση URL της εφαρμογής. Οι διευθύνσεις URL που ταιριάζουν με με τα μοτίβα URL που περιγράφονται στο αρχείο urls.py, αποστέλλονται στην αντίστοιχη συνάρτηση στο αρχείο views.py. Η αντιστοίχιση γίνεται από πάνω προς τα κάτω στο αρχείο urls.py. Ενώ το ακριβές ταίριασμα URL υλοποιήθηκε σε αυτό το έργο, το Django επιτρέπει επίσης τη χρήση ταυτοποίησης κανονικών εκφράσεων. Στην εικόνα παρακάτω, φαίνονται οι διευθύνσεις URL του API όπου κάθε διεύθυνση URL αντιστοιχεί στη συνάρτηση στο αρχείο api1/views.py που καταλήγει στην εκτέλεση του σεναρίου.

```
urlpatterns = [
    path('', views.firstPage),
    path('manage/',views.index,name="manage"),
    path('manage2/',views.index2,name="manage2"),
    path('manage3/',views.index3,name="manage3"),
    path('device_statistics/<int:device_id>', views.get_interface_statistics,name="device_statistics"),
    path('interface_statistics/<int:device_id>', views.get_interfaces_counters, name="interface_statistics"),
    path('device/<int:device_id>', views.get_device_stats, name="device"),
    path('execute_script/', views.execute_script_on_remote, name='execute_script'),
    path('manage4/',views.index4,name="manage4"),
    path('manage5/',views.index5,name="manage5"),
    path('running_config/<int:device_id>', views.get_running_config, name="running_config"),
    path('show_running_config/<int:device_id>/', views.show_running_config, name='show_running_config'),
```

Σχήμα 4.7: url.py

4.4.2 Τα αρχεία views.py

Οι συναρτήσεις σε ένα αρχείο views.py καλούνται όταν η δεδομένη διεύθυνση URL που αποστέλλεται από το χρήστη ταιριάζει με το αντίστοιχο μοτίβο URL στο αρχείο urls.py. Παράμετροι που αποστέλλονται μέσω κλήσης HTTP εισέρχονται στην αντίστοιχη συνάρτηση μέσω παραμέτρων ή σώματος αίτησης. Στο αρχείο views.py του API, η συνάρτηση εκτελεί το σενάριο κώδικα με τις δεδομένες παραμέτρους εισόδου. Όταν τελειώσει η εκτέλεση του κώδικα δέσμης ενεργειών, το αποτέλεσμα επιστρέφεται στη συνάρτηση views και μεταβιβάζεται ως πλαίσιο στο αντίστοιχο αρχείο .html για την εμφάνιση των αποτελεσμάτων στον χρήστη που εκτέλεσε το σενάριο. Ένα παράδειγμα μιας συνάρτησης σε ένα αρχείο ενιεως.πψ μπορείτε να δείτε στο σχήμα παρακάτω

```python
def index3(request: HttpRequest) -> HttpResponse:
    devices = Device.objects.all()
    context = {
        'title': 'Interface Statistics',

        'devices': devices
    }
    return render(request, 'index3.html', context)

def index4(request: HttpRequest) -> HttpResponse:
    devices = Device.objects.all()
    context = {
        'title': 'Backup running config',

        'devices': devices
    }
    return render(request, 'index4.html', context)

def index5(request: HttpRequest) -> HttpResponse:
    devices = Device.objects.all()
    context = {
        'title': 'Backup running config',

        'devices': devices
    }
    return render(request, 'index5.html', context)
```

Σχήμα 4.8: views.py

4.4.3 Τα αρχεία html template

Στη συνάρτηση views.py, το Django αποδίδει το αντίστοιχο πρότυπο .html αρχείο με ένα συγκεκριμένο πλαίσιο. Το πλαίσιο είναι σε μορφή JavaScript Object Notation (JSON) και αποστέλλεται στο αρχείο .html. Τα δεδομένα στο πλαίσιο εμφανίζονται στο .html, εάν το .html έχει παραμετροποιηθεί κατάλληλα. Ένα παράδειγμα .html με τον συντακτικό κώδικα για τον τρόπο πρόσβασης στα δεδομένα του πλαισίου παρουσιάζεται στην εικόνα παρακάτω

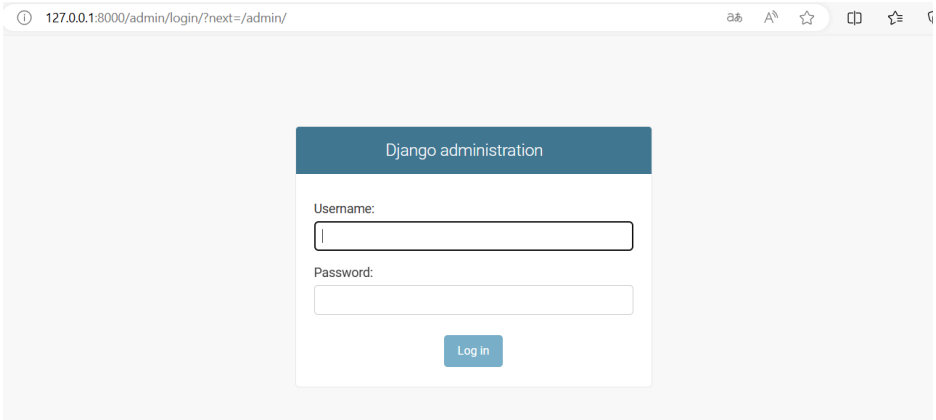
4.4.4 Η βάση δεδομένων

Η διαδικτυακή πύλη χρησιμοποιεί μια βάση δεδομένων SQLite. Αυτή η βάση δεδομένων περιέχει τρία μοντέλα τα οποία ορίζονται στο αρχείο models.py. Το αρχείο αυτό περιέχει μία κλαση Δειγε η οποία δέχεται σαν ορίσματα το όνομα, την IP, το όνομα χρήστη, τον κωδικό, τον κρυφό κωδικό και το μοντέλο της συσκευής. Υπάρχει μία εγγραφή στη βάση δεδομένων ένα από αυτά τα αντικείμενα τα οποία εμείς τα δημιουργούμε. Το μοντέλο διαπιστευτηρίων αποθηκεύει τα διαπιστευτήρια τα οποία έχουν προηγουμένως κρυπτογραφημένα. Με αυτό, ορισμένες δέσμες ενεργειών μπορούν να λάβουν τα διαπιστευτήρια που απαιτούνται για να λειτουργήσουν χωρίς την είσοδο του χρήστη και χωρίς να γίνεται άμεση αναφορά στα διαπιστευτήρια

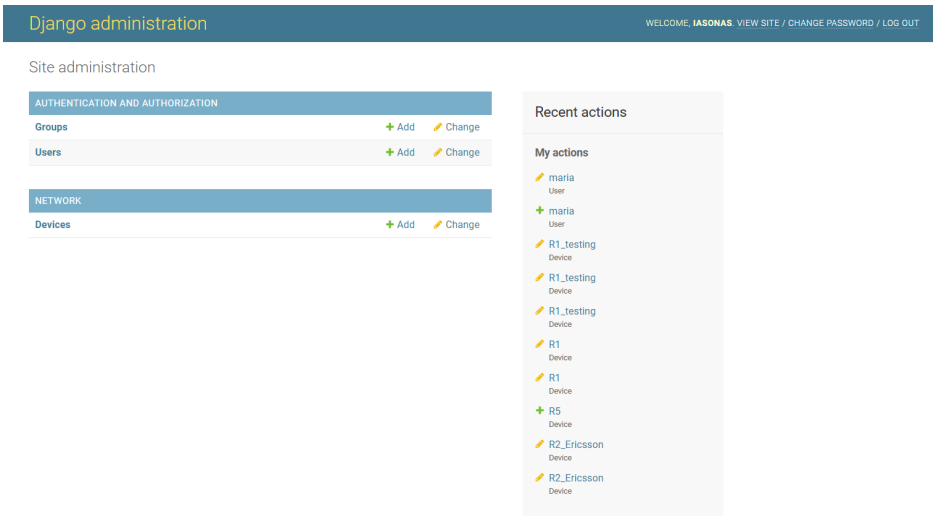
```
</head>
<body>
  <div class="container">
    <h1>{{ title }}</h1>
    <h2>Devices</h2>
    <table>
      <tr>
        <th>Id</th>
        <th>Name</th>
        <th>Host</th>
      </tr>
      <tr>
        <td>result</td>
        <td><a href="{% url 'device' device.id %}">{{ device.name }}</a></td>
        <td>{{ device.host }}</td>
      </tr>
    </table>
  </div>
</body>
</html>
```

Σχήμα 4.9: Παράδειγμα html αρχείου

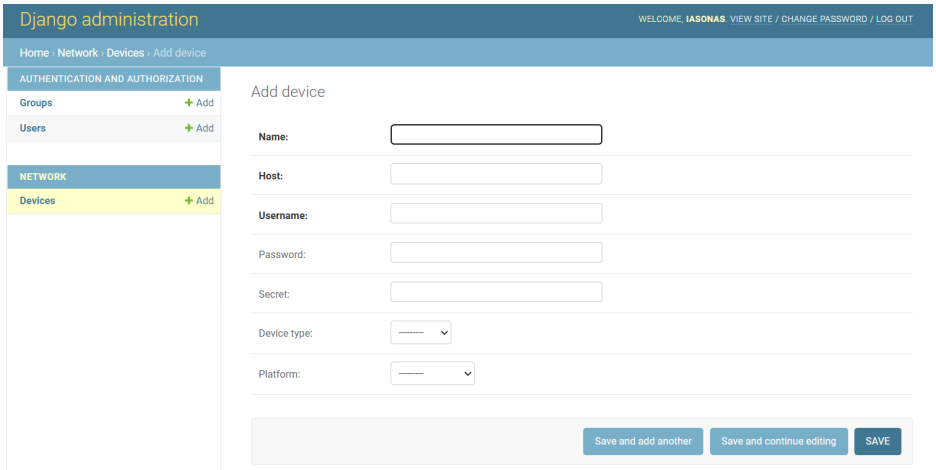
στο κώδικα. Η διαχείριση της βάσης δεδομένων μπορεί να γίνει απευθείας μέσω μιας γραφικής διεπαφής χρήστη (GUI) στο Django. Σε αυτό το GUI μπορούν να έχουν πρόσβαση μόνο οι χρήστες διαχειριστές. Παραδείγματα των εγγραφών του μοντέλου της εφαρμογής και του μοντέλου του εργάτη που εμφανίζονται σε αυτό το GUI παρουσιάζονται παρακάτω στην εικόνα.



Σχήμα 4.10: Είσοδος στο GUI



Σχήμα 4.11: Κεντρική σελίδα του Django Administration GUI



Σχήμα 4.12: Προσθήκη συσκευής