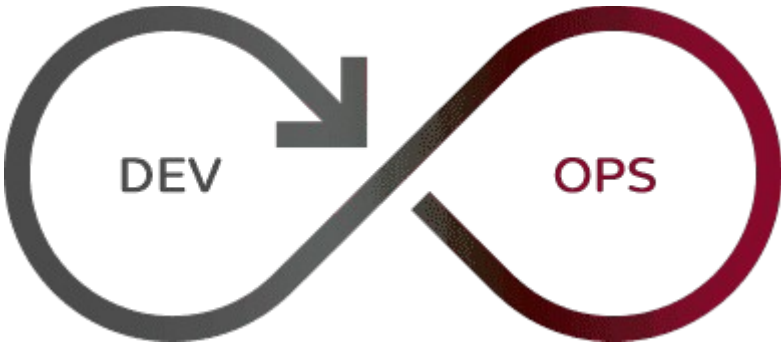
ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΜΑΤΙΚΗΣ



Ομάδα 15

Αίας Κωνσταντίνος Καρκάνης - it2022031

Ιωάννης Καχραμάνος - it2022034

Θέμα Εργασίας

Εργασία στο Μάθημα «**Βασικές έννοιες και εργαλεία DevOps**»

Περιεχόμενα

Ανάλυση και Σχεδίαση

Η υλοποίηση της εργασίας βασίστηκε στις αρχές του **Infrastructure as Code** και της πλήρους αυτοματοποίησης. Όλες οι διαδικασίες εγκατάστασης και ρύθμισης του συστήματος πραγματοποιούνται χωρίς την ανάγκη χειροκίνητων παρεμβάσεων.

Σε σύγκριση με την αρχική εφαρμογή που είχε παραδοθεί στο πλαίσιο του μαθήματος των Κατανεμημένων Συστημάτων (DS-2025), προστέθηκαν επιπλέον λειτουργικότητες. Συγκεκριμένα, επεκτάθηκε η λειτουργικότητα της εφαρμογής με:

- Σύστημα αποστολής email και επιβεβαίωσης λογαριασμών μέσω του MailHog, το οποίο χρησιμοποιείται για την προσομοίωση SMTP server και τη δοκιμή της διαδικασίας ενεργοποίησης χρηστών.
- Σύστημα αποθήκευσης αρχείων PDF μέσω του MinIO, το οποίο χρησιμοποιείται ως αντικειμενοστραφές σύστημα αποθήκευσης (object storage) για την αποθήκευση εγγράφων της εφαρμογής.

Για όλα τα υπόλοιπα στοιχεία της εφαρμογής, όπως οι παραδοχές, η ανάλυση και τα σενάρια χρήσης, ισχύουν όσα περιγράφονται στο report της εφαρμογής DS-2025.

Deployment

Αρχικά, πραγματοποιήθηκε ανάπτυξη σε τοπικό περιβάλλον χρησιμοποιώντας Vagrant και VirtualBox. Δημιουργήθηκαν τρία βασικά virtual machines:

- VM για το frontend
- VM για το backend
- VM για τη βάση δεδομένων PostgreSQL

Στο τοπικό περιβάλλον χρησιμοποιήθηκε ο Nginx ως reverse proxy για τη διασύνδεση των υπηρεσιών και την έκθεση της εφαρμογής μέσω ενιαίου URL.

Στη συνέχεια, υλοποιήθηκε deployment σε περιβάλλον Docker, όπου όλα τα components της εφαρμογής εκτελούνται ως containers μέσα σε ένα VM. Στο συγκεκριμένο σενάριο δεν χρησιμοποιείται ξεχωριστός Nginx server, καθώς η δρομολόγηση των αιτημάτων πραγματοποιείται εσωτερικά μέσω του Docker networking.

Τέλος, πραγματοποιήθηκε deployment σε Kubernetes cluster. Στο Kubernetes κάθε υπηρεσία εκτελείται σε ξεχωριστό pod, ενώ η πρόσβαση στην εφαρμογή πραγματοποιείται μέσω Ingress controller. Όπως και στο περιβάλλον Docker, δεν χρησιμοποιείται ξεχωριστός Nginx server εκτός cluster, καθώς η δρομολόγηση των αιτημάτων πραγματοποιείται μέσω του μηχανισμού ingress.

Επιπλέον, χρησιμοποιήθηκε ξεχωριστό VM για το Jenkins, το οποίο στο τοπικό περιβάλλον περιλαμβάνει το σύνολο των λειτουργιών CI/CD. Στο online περιβάλλον το Jenkins VM διατηρεί τον βασικό ρόλο αυτοματοποίησης και εκτέλεσης pipelines, με εξαίρεση τα jobs που αφορούν τη φόρτωση demo δεδομένων σε Docker και Kubernetes, τα οποία δεν υλοποιήθηκαν στο cloud περιβάλλον.

Επιπλέον, υλοποιήθηκε deployment και σε περιβάλλον cloud (Google Cloud), όπου αναπτύχθηκαν αντίστοιχα VMs. Εξαίρεση αποτέλεσε η αρχική υλοποίηση, όπου το frontend, το backend, καθώς και οι υποστηρικτικές υπηρεσίες (PostgreSQL, MinIO και MailHog) εκτελούνταν στο ίδιο VM . Στην τελική μορφή του cloud deployment χρησιμοποιήθηκε πλήρης διασύνδεση μέσω Nginx, με κανονικά URLs και reverse proxy αρχιτεκτονική.

Ansible

Η Ansible χρησιμοποιήθηκε ως βασικό εργαλείο αυτοματοποίησης της εγκατάστασης και παραμετροποίησης όλων των υποδομών.

Μέσω Ansible playbooks υλοποιήθηκαν:

- Εγκατάσταση PostgreSQL και δημιουργία βάσης δεδομένων.
- Εγκατάσταση και παραμετροποίηση backend εφαρμογής.
- Εγκατάσταση και build frontend εφαρμογής.
- Δημιουργία systemd services για την αυτόματη εκκίνηση και διαχείριση των υπηρεσιών στο επίπεδο του λειτουργικού συστήματος.
- Παραμετροποίηση Nginx.
- Εγκατάσταση MailHog για αποστολή και έλεγχο email.
- Εγκατάσταση MinIO για αποθήκευση αρχείων.

Επιπλέον, υλοποιήθηκαν playbooks τα οποία εκτελούν SQL scripts για την εισαγωγή δοκιμαστικών δεδομένων και φωτογραφιών στη βάση δεδομένων. Η διαδικασία αυτή εφαρμόζεται κυρίως στα deployments που βασίζονται σε VMs, ενώ στα περιβάλλοντα Docker και Kubernetes η αρχικοποίηση δεδομένων υλοποιείται μέσω containers και Kubernetes jobs.

Η δομή των playbooks είναι modular, καθώς η εγκατάσταση χωρίζεται σε επιμέρους playbooks για κάθε υπηρεσία.

Ansible - Docker

Στο σενάριο Docker, η Ansible χρησιμοποιήθηκε για την αυτοματοποίηση της διαδικασίας containerization.

Συγκεκριμένα υλοποιήθηκαν:

- Δημιουργία Docker images για backend και frontend.
- Push των images σε GitHub Container Registry.
- Εκκίνηση containers μέσω Docker Compose.
- Παραμετροποίηση μεταβλητών περιβάλλοντος.
- Εκτέλεση health checks για την επαλήθευση λειτουργίας.

Η υλοποίηση πραγματοποιείται σε ξεχωριστό VM το οποίο λειτουργεί ως docker host. Το VM αυτό χρησιμοποιείται τόσο σε τοπικό περιβάλλον όσο και στο cloud deployment.

Kubernetes

Στο Kubernetes δημιουργήθηκαν οι ακόλουθες βασικές οντότητες:

Namespace

Δημιουργήθηκε ξεχωριστό namespace για την απομόνωση της εφαρμογής.

Deployments

Υλοποιήθηκαν deployments για:

- Frontend
- Backend
- MinIO
- MailHog

StatefulSet

Χρησιμοποιήθηκε StatefulSet για την PostgreSQL βάση δεδομένων, ώστε να εξασφαλιστεί μόνιμη αποθήκευση δεδομένων.

Persistent Volume Claims

Χρησιμοποιήθηκαν PVCs για:

- Αποθήκευση δεδομένων PostgreSQL
- Αποθήκευση δεδομένων MinIO

Services

Δημιουργήθηκαν services για την εσωτερική επικοινωνία των pods.

Ingress

Υλοποιήθηκε ingress controller για τη δρομολόγηση αιτημάτων προς frontend και backend.

Jobs

Δημιουργήθηκαν Kubernetes jobs για:

- Εισαγωγή δοκιμαστικών δεδομένων.
- Εισαγωγή φωτογραφιών στη βάση δεδομένων.

Σημείωση – MailHog

Στο τοπικό περιβάλλον Kubernetes (microk8s στο k8shost), η πρόσβαση στο MailHog UI δεν είναι μόνιμα εκτεθειμένη. Η θύρα παραμένει κλειστή και ανοίγει μόνο όταν χρειάζεται, μέσω port-forward.

Το script εκτελεί port-forward προς το service mailhog στο namespace ds2025 και επιτρέπει πρόσβαση από τον host στη διεύθυνση: http://127.0.0.1:18025

CI/CD

Για τη συνεχή ολοκλήρωση και παράδοση χρησιμοποιήθηκε Jenkins.

Jenkins – Διαφοροποίηση Local και Online

- Στο Local (Vagrant) περιβάλλον, η εγκατάσταση και ρύθμιση του Jenkins (SSH user, keypair, authorization στα target VMs) πραγματοποιείται αυτοματοποιημένα μέσω Ansible.

- Στο Online περιβάλλον, η αρχική εγκατάσταση και η διαχείριση των SSH keys/credentials πραγματοποιήθηκαν χειροκίνητα.

Η δημιουργία ή ρύθμιση του admin λογαριασμού και η αρχική πρόσβαση στο UI παραμένουν εκτός του αυτοματοποιημένου provisioning.

Στο Jenkins δημιουργήθηκαν pipelines τα οποία υλοποιούν:

Pipelines Docker

Για το περιβάλλον Docker υλοποιήθηκαν δύο ξεχωριστά pipelines:

Pipeline Build & Push Images

- Δημιουργία Docker images για το backend και το frontend.
- Push των images στο GitHub Container Registry.

Σημείωση:

Στο πλαίσιο του CI/CD, τα Docker images του frontend και του backend δημιουργούνται μέσω Jenkins pipelines και αποθηκεύονται στο GitHub Container Registry. Στη συνέχεια, στα deployments με Docker και Kubernetes δεν γίνεται νέο build της εφαρμογής. Αντίθετα, τα περιβάλλοντα αυτά κατεβάζουν (pull) τα έτοιμα images από το registry.

Pipeline Docker Deployment

- Deployment της εφαρμογής σε docker host μέσω Ansible.

Pipelines Kubernetes

Υλοποιήθηκε pipeline το οποίο περιλαμβάνει:

- Εγκατάσταση και παραμετροποίηση MicroK8s.
- Deployment των Kubernetes manifests.
- Έλεγχο της κατάστασης του cluster και των pods.

Pipelines VM Deployment

Υλοποιήθηκε pipeline το οποίο περιλαμβάνει:

- Αυτόματη εκκίνηση των VM.
- Εκτέλεση Ansible playbooks για εγκατάσταση και παραμετροποίηση των υπηρεσιών.
- Εκτέλεση health checks για επαλήθευση της λειτουργίας των υπηρεσιών.

Seed Job

Επιπλέον, υλοποιήθηκε seed job, το οποίο δημιουργεί αυτόματα τα υπόλοιπα Jenkins pipelines. Με τον τρόπο αυτό απαιτείται η δημιουργία και εκτέλεση μόνο ενός pipeline, το οποίο στη συνέχεια κατασκευάζει αυτόματα όλα τα υπόλοιπα pipelines και τα αντίστοιχα jobs.

Jenkins στο Cloud

Στο cloud περιβάλλον υλοποιήθηκε ξεχωριστό Jenkins VM, το οποίο εκτελεί τα pipelines deployment.

Στο online περιβάλλον υλοποιούνται αντίστοιχα pipelines, με τη διαφορά ότι δεν περιλαμβάνονται τα jobs που χρησιμοποιούνται για την εισαγωγή δοκιμαστικών δεδομένων στα περιβάλλοντα Docker και Kubernetes.

Γενικά Σχόλια/Παρατηρήσεις

Επιπλέον, υλοποιήθηκαν scripts τα οποία εκτελούν σειριακά όλα τα απαιτούμενα Ansible playbooks για κάθε deployment σενάριο. Συγκεκριμένα, στο τοπικό περιβάλλον υπάρχουν scripts για deployment σε Docker, σε VMs, σε Kubernetes καθώς και για την εγκατάσταση και παραμετροποίηση του Jenkins.

Τα scripts αυτά αναλαμβάνουν την αυτόματη εκκίνηση των αντίστοιχων VM μέσω Vagrant και VirtualBox, καθώς και τη δημιουργία και διαχείριση των απαραίτητων SSH κλειδιών για την ασφαλή επικοινωνία μεταξύ των συστημάτων.

Με δυσκόλεψε / δεν υλοποίησα

Η μεγαλύτερη δυσκολία παρουσιάστηκε στην αρχική κατανόηση της αρχιτεκτονικής του συστήματος και στον συνολικό σχεδιασμό της υποδομής.

Σημαντική δυσκολία προέκυψε επίσης στο frontend της εφαρμογής, όπου υπήρχαν πολλοί σταθεροί σύνδεσμοι προς τοπικές διευθύνσεις (localhost). Για την επίλυση του προβλήματος απαιτήθηκε εκτεταμένη τροποποίηση του κώδικα, καθώς και η χρήση αυτοματοποιημένων scripts για την αντικατάσταση των συγκεκριμένων URLs.

Επιπλέον, λόγω χρονικών περιορισμών, ορισμένα Ansible playbooks παρουσιάζουν παρόμοια λειτουργικότητα σε διαφορετικά deployment σενάρια. Η δομή τους θα μπορούσε να ενοποιηθεί περαιτέρω ώστε να μειωθεί ο επαναλαμβανόμενος κώδικας και να αυξηθεί η επεκτασιμότητα του συστήματος.

Στην παρουσίαση της εργασίας θα δοθεί έμφαση κυρίως στην υλοποίηση του online περιβάλλοντος deployment.

Κώδικας

Αποθετήρια κώδικα

- https://github.com/it2022031/DS-2025.git
- https://github.com/it2022031/DevOps-2025

Δοκιμαστικά accounts και urls

Ρόλος των Scripts (Local & Cloud)

Τα scripts λειτουργούν ως wrapper εκτέλεσης των αντίστοιχων Ansible playbooks. Καθορίζουν:

- το κατάλληλο inventory (-i),
- το αντίστοιχο playbook (.yml)
- τις απαραίτητες μεταβλητές περιβάλλοντος (π.χ. ANSIBLE_CONFIG, ANSIBLE_HOST_KEY_CHECKING=False)

Κάθε script αντιστοιχεί ουσιαστικά σε μία ή δύο εντολές ansible/ansible-playbook που μπορούν να εκτελεστούν και χειροκίνητα.

Ρόλος των Scripts (Local Only)

- Εκκινούν αυτόματα τα απαιτούμενα Virtual Machines (vagrant up).]
- Αναμένουν την πλήρη διαθεσιμότητα του SSH.
- Δημιουργούν δυναμικά το αρχείο infra/ssh/ssh.config μέσω vagrant ssh-config.
- Χρησιμοποιούν αυτόματα τα τοπικά SSH κλειδιά που παρέχει το Vagrant.
- Ρυθμίζουν τις απαραίτητες παραμέτρους Ansible (ANSIBLE_CONFIG, remote_tmp, κ.λπ.).

Url δοκιμαστικού περιβάλλοντος

Online Περιβάλλον – Μεταβαλλόμενες IP Διευθύνσεις

Στο online/cloud περιβάλλον οι IP διευθύνσεις δεν θεωρούνται σταθερές.
Κατά συνέπεια, πριν από κάθε νέο deployment ενδέχεται να απαιτείται επικαιροποίηση των αρχείων:

infra/inventories/cloud_*.ini

Οδηγίες Χρήσης / Εγκατάστασης

Για οδηγίες μπορείτε να συμβουλευτείτε το αρχείο README.md στο repo του project.