HERMES

A SIMPLE OPEN-SOURCE APPOINTMENT SCHEDULER

CONTRIBUTORS

1η ΦΑΣΗ ΑΝΑΠΤΥΞΗΣ (11/20 – 02/21) – Σχεδιασμός Και Κύρια Υλοποίηση:

- Νίκος Δημητρακόπουλος
- Παρασκευή-Θεοφανία Γουργιώτη
- Ιωάννης Χρήστου

2η ΦΑΣΗ ΑΝΑΠΤΥΞΗΣ (05/21 – 07/21) – Αυτοματοποίηση και DevOps:

- Νίκος Δημητρακόπουλος
- Αθανάσιος Αποστολίδης

ΛΕΙΤΟΥΡΓΙΚΕΣ ΠΡΟΔΙΑΓΡΑΦΕΣ

Εσωτερικό Σύστημα:

- Οι Διαχειριστές/στριες Οργανισμών μπορούν:
  - Να διαχειρίζονται τον κατάλογο οργανισμών του συστήματος.
  - Για κάθε οργανισμό να διαχειρίζονται τον κατάλογο των προϊσταμένων του.
  - Να διαχειρίζονται τον κατάλογο χρηστών των πολιτών.
- Οι Προϊστάμενοι/ες Οργανισμών μπορούν:
  - Να διαχειρίζονται τον κατάλογο χρηστών των υπαλλήλων στον οργανισμό τον οποίο εργάζονται.
- Οι Υπάλληλοι Οργανισμών μπορούν:
  - Να διαχειρίζονται τις αιτήσεις ραντεβού του οργανισμού στον οποίο εργάζεται.

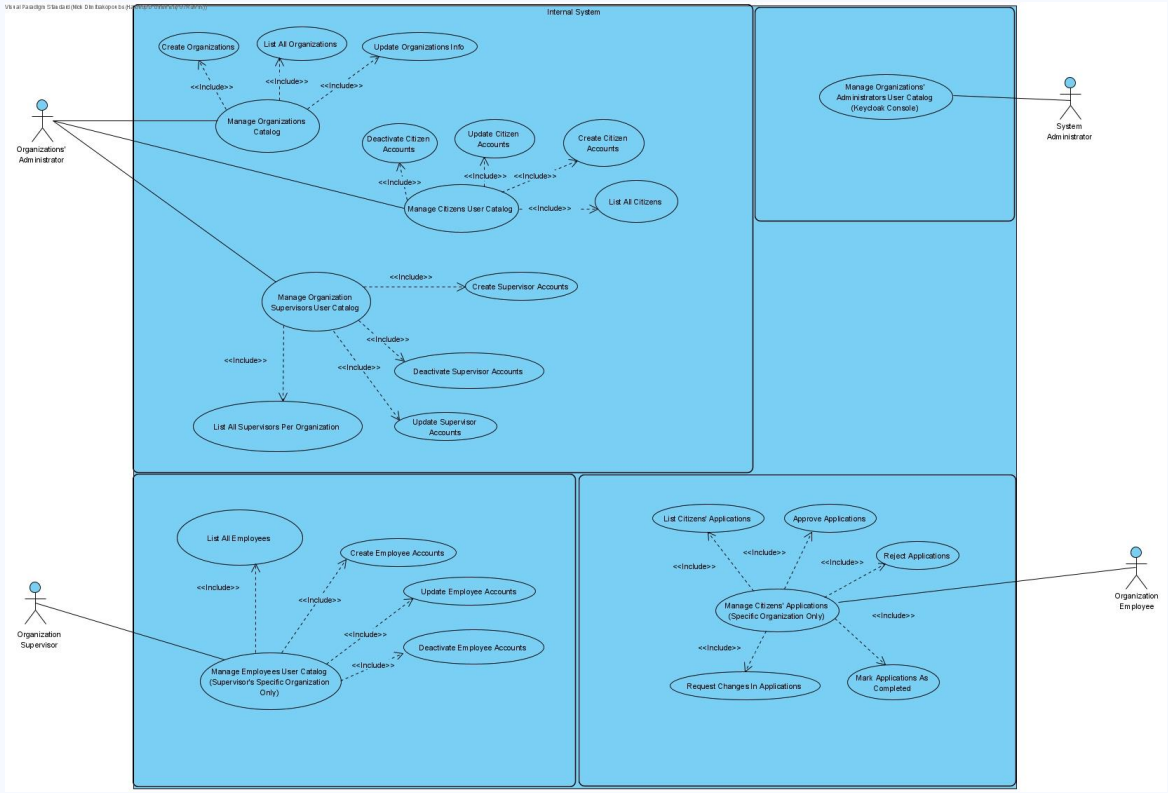
ΛΕΙΤΟΥΡΓΙΚΕΣ ΠΡΟΔΙΑΓΡΑΦΕΣ

Εξωτερικό Σύστημα:

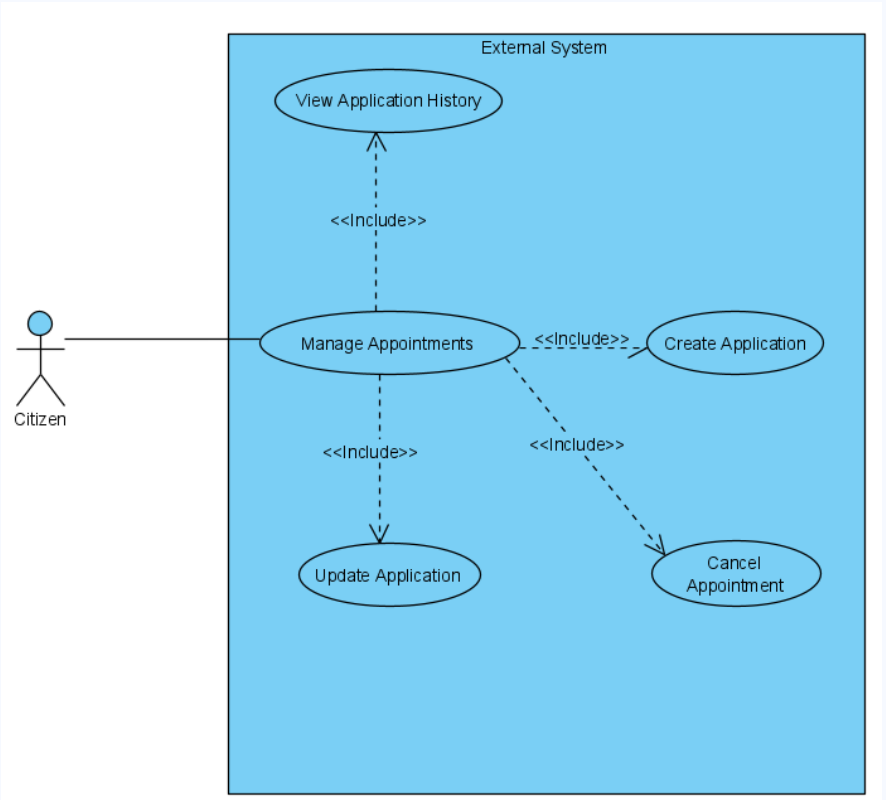
- Οι Πολίτες μπορούν:
  - Να δημιουργούν και να διαχειρίζονται τις αιτήσεις που πραγματοποιούν σε οποιοδήποτε από τους οργανισμούς του συστήματος

ΛΕΙΤΟΥΡΓΙΚΕΣ ΠΡΟΔΙΑΓΡΑΦΕΣ

Εσωτερικό Σύστημα

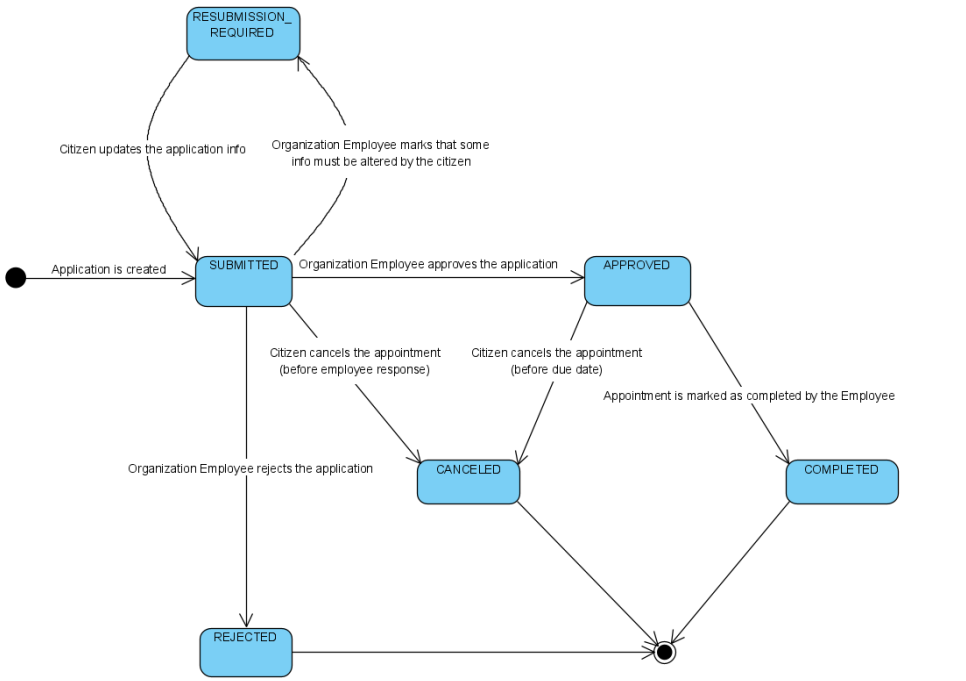


Εξωτερικό Σύστημα



ΛΕΙΤΟΥΡΓΙΚΕΣ ΠΡΟΔΙΑΓΡΑΦΕΣ

Πιθανές Καταστάσεις Μιας Αίτησης



Απόκομμα από User Guide εξωτερικού συστήματος

Στο πεδίο Details μπορείτε να αφήσετε οποιοδήποτε σχόλιο επιθυμείτε σχετικά με την αίτηση σας. Επιλέξτε την ημερομηνία και την ώρα που επιθυμείτε να κλείσετε το ραντεβού σας και πατήστε Save(4). Ένας υπάλληλος του οργανισμού στον οποίο καταθέσατε την αίτηση, θα ελέγξει το αίτημά σας και θα ενημερώσει καταλλήλως την κατάσταση της αίτησης σας. Βεβαίως, αν μετανιώσετε για κάτι μπορείτε πάντα να ακυρώσετε την καταχώριση της αίτησης, πατώντας το Cancel(5).

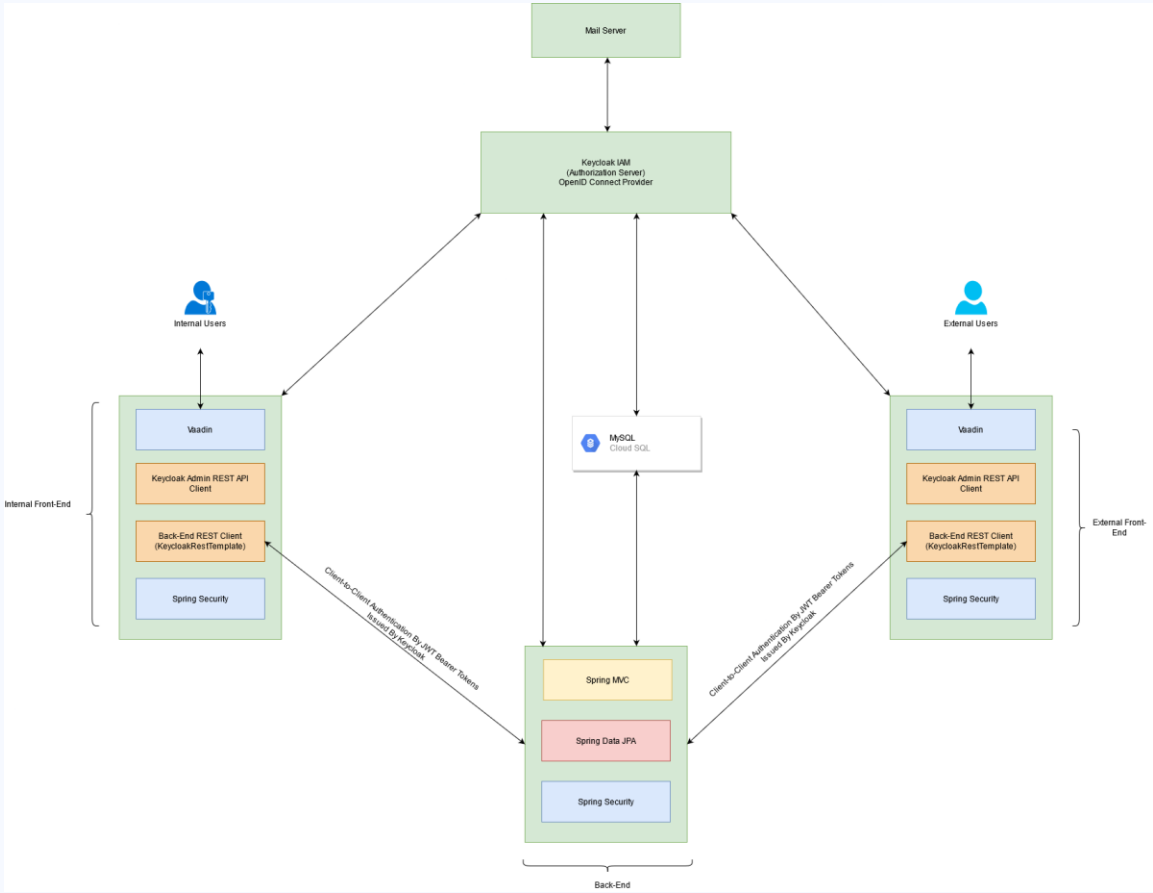
Μπορείτε να **τροποποιήσετε** μία αίτηση πατώντας το Edit(2) **αλλά μόνο** αν ο υπάλληλος του οργανισμού έχει ζητήσει να γίνουν αλλαγές στην αίτηση (Resubmission Required κατάσταση). Η εμπειρία χρήσης είναι ακριβώς η ίδια με την δημιουργία μίας νέας αίτησης.

Μπορείτε οποιοδήποτε στιγμή να **ακυρώσετε** τις αιτήσεις σας πατώντας το Cancel(3), **εκτός** αν το ραντεβού **έχει ήδη πραγματοποιηθεί** (Completed κατάσταση), **έχει απορριφθεί** (Rejected κατάσταση) ή **έχει ήδη ακυρωθεί** (Canceled κατάσταση).

HIGH LEVEL DESIGN

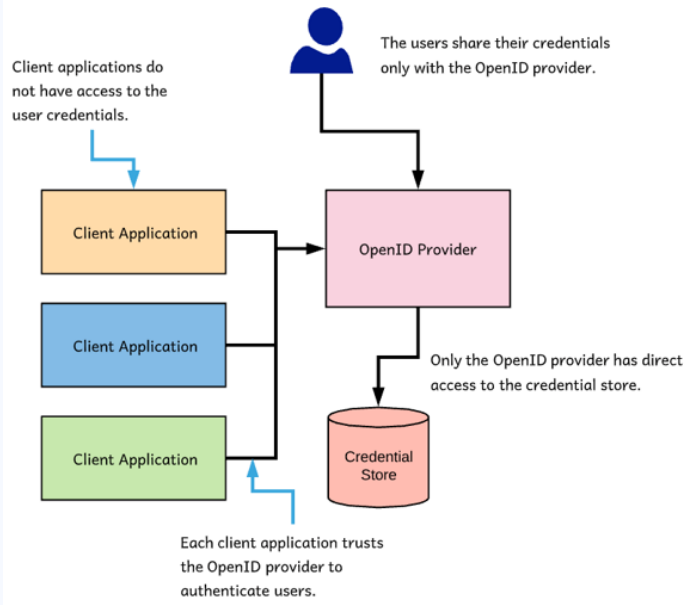
6 Services:

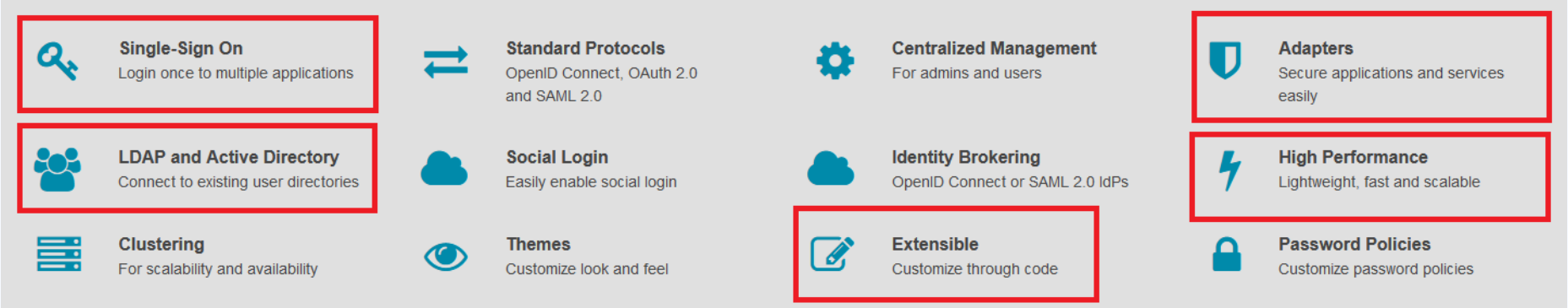
- Authorization Server (Keycloak)
- Applications Service (Back-End)
- Internal Front-End
- External Front-End
- MySQL RDBMS
- Mailhog (Mock Mail Server)



KEYCLOAK

- An Open-Source Identity And Access Management (IAM) Software Product
- Maintained By Red Hat
- Enables Single Sign-On (SSO)
- Can Be Easily Integrated In Software Projects
- Extremely Customizable
- High Performance

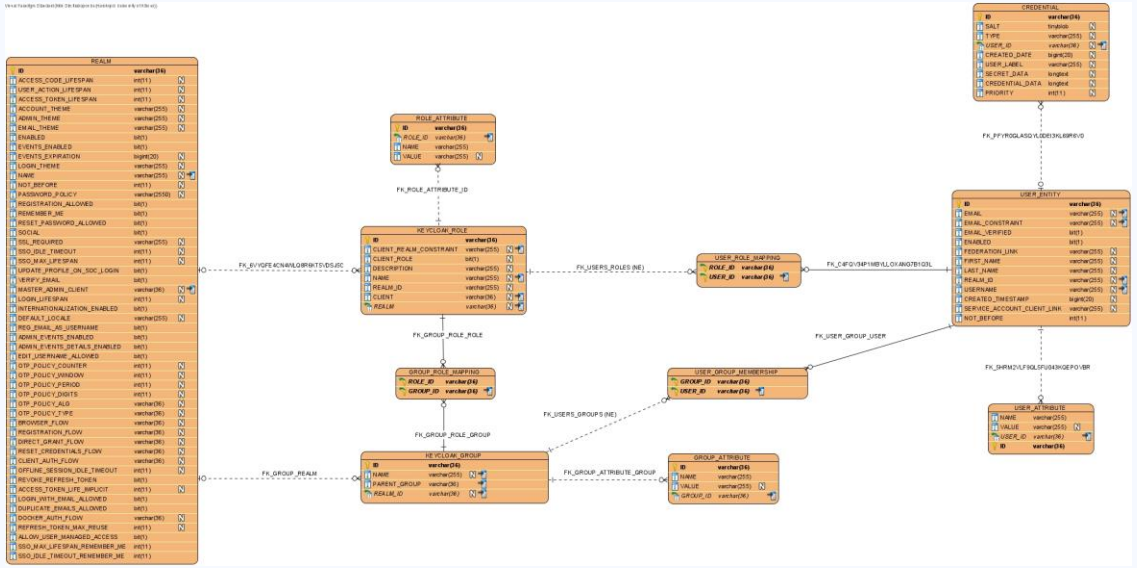


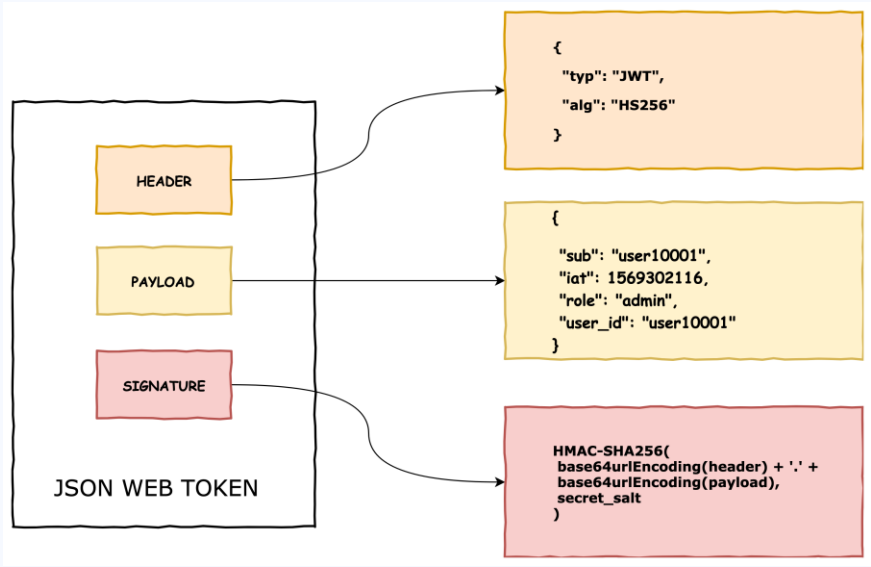


KEYCLOAK

STATE OF THE ART SECURITY OUT-OF-THE-BOX

- We don't need to create a DB Schema For Users & Organizations – It's already there
- We don't need to write custom code to handle core security – Keycloak does it for us

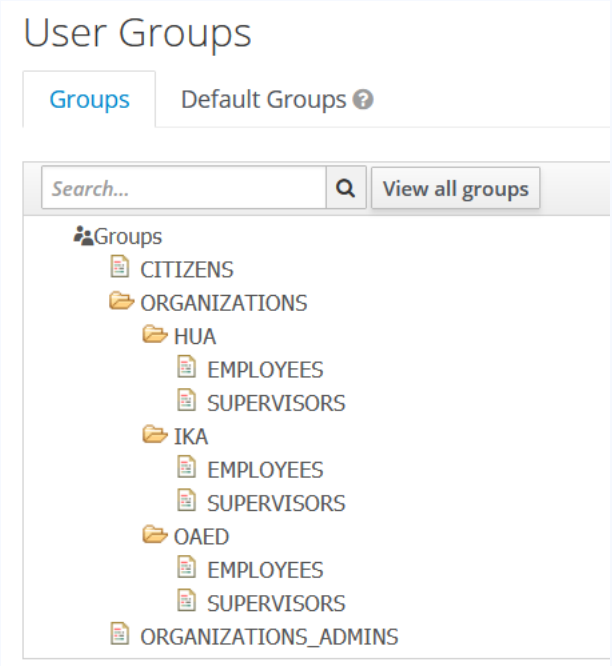




KEYCLOAK

USER + ORGANIZATION CATALOGS HASSLE FREE

- Hierarchical Catalog Management Out-Of-The-Box



KEYCLOAK
CUSTOMIZATION

Custom REST Endpoints (JAX-RS – RESTEasy)

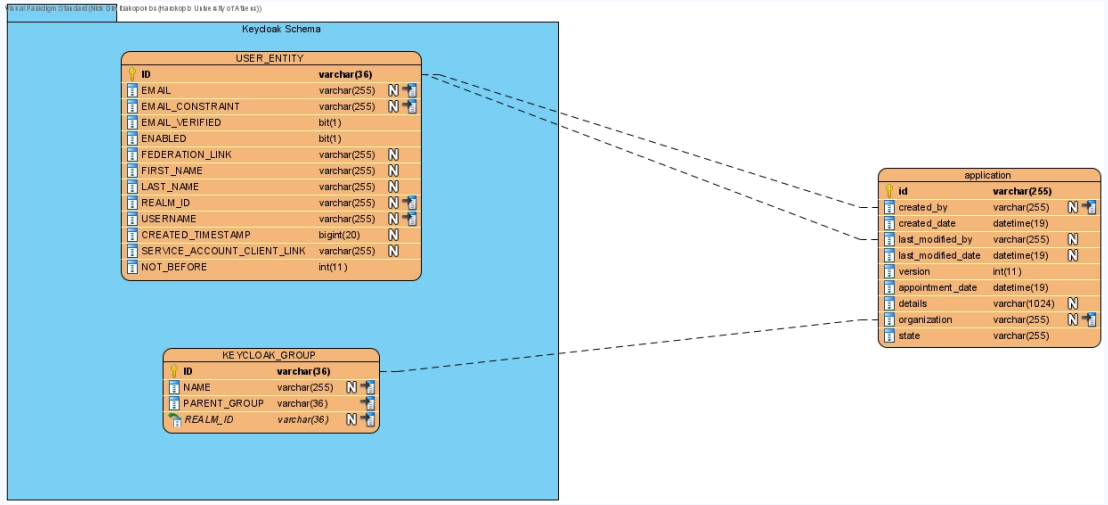
```
public OrganizationsResource(KeycloakSession session) {
    this.session = session;
    this.realm = session.getContext().getRealm();
    this.em = session.getProvider(JpaConnectionProvider.class).getEntityManager();
    this.clientConnection = session.getContext().getConnection();

    AdminAuth adminAuth = authenticateRealmAdminRequest(session.getContext().getRequestHeaders());
    this.auth = AdminPermissions.evaluator(session, realm, adminAuth);
    this.adminEvent = new AdminEventBuilder(realm, adminAuth, session, clientConnection);
    this.organizationsGroup = findOrganizationsGroup();
}

@GET
@Produces({MediaType.APPLICATION_JSON})
public List<GroupRepresentation> list(@QueryParam("offset") Integer offset,
                                      @QueryParam("limit")  Integer limit)
{
    auth.groups().requireView();

    //As far as I know, there isn't any ready implementation to get subgroups with paginated queries
    //So I made one myself
    return em.createQuery("SELECT g FROM GroupEntity g WHERE g.parentId = :parentId", GroupEntity.class)
            .setParameter("parentId",organizationsGroup.getId())
            .setFirstResult(offset)
            .setMaxResults(limit)
            .getResultStream()
            .map(entity -> (GroupModel)new GroupAdapter(realm,em,entity))
            .map(model -> ModelToRepresentation.toRepresentation(model,true))
            .collect(Collectors.toList());

}
```

Custom Mappers(Put Info In Token)

```
protected void setClaim(IDToken token, ProtocolMapperModel mappingModel, UserSessionModel userSession) {

    UserModel user = userSession.getUser();
    Set<RoleModel> roles = new HashSet<>();
    roles.add(KeycloakModelUtils.getRoleFromString(userSession.getRealm(),"ROLE_ORG_SUPERVISOR"));
    roles.add(KeycloakModelUtils.getRoleFromString(userSession.getRealm(),"ROLE_ORG_EMPLOYEE"));

    boolean worksInOrganization = user.getGroupsStream()
            .flatMap(RoleMapperModel::getRoleMappingsStream)
            .anyMatch(roles::contains);

    if(!worksInOrganization)
        return;

    //We assume that a correct setup is made here.
    //If the role is not assigned to a group or
    //that group does not have a parent, this will fail miserably.
    GroupModel group = userSession.getUser()
            .getGroupsStream()
            .findFirst()
            .get()
            .getParent();


    String protocolClaim = mappingModel.getConfig().get(OIDCAttributeMapperHelper.TOKEN_CLAIM_NAME);
    token.getOtherClaims().put(protocolClaim, new SummarizedGroup(group.getId(),group.getName()));
```

APPLICATIONS SERVICE (SPRING BOOT)

- Simple Back-End
- Handles Applications Business Logic For Employees And Citizens
- REST API With Pagination Support (Integrates Smoothly With Vaadin)
- Simple DB Schema (Just 1 Table – Thank You Keycloak !)
- Not Exposed To The Outside World (But That Doesn't Mean That Security Is Unnecessary)

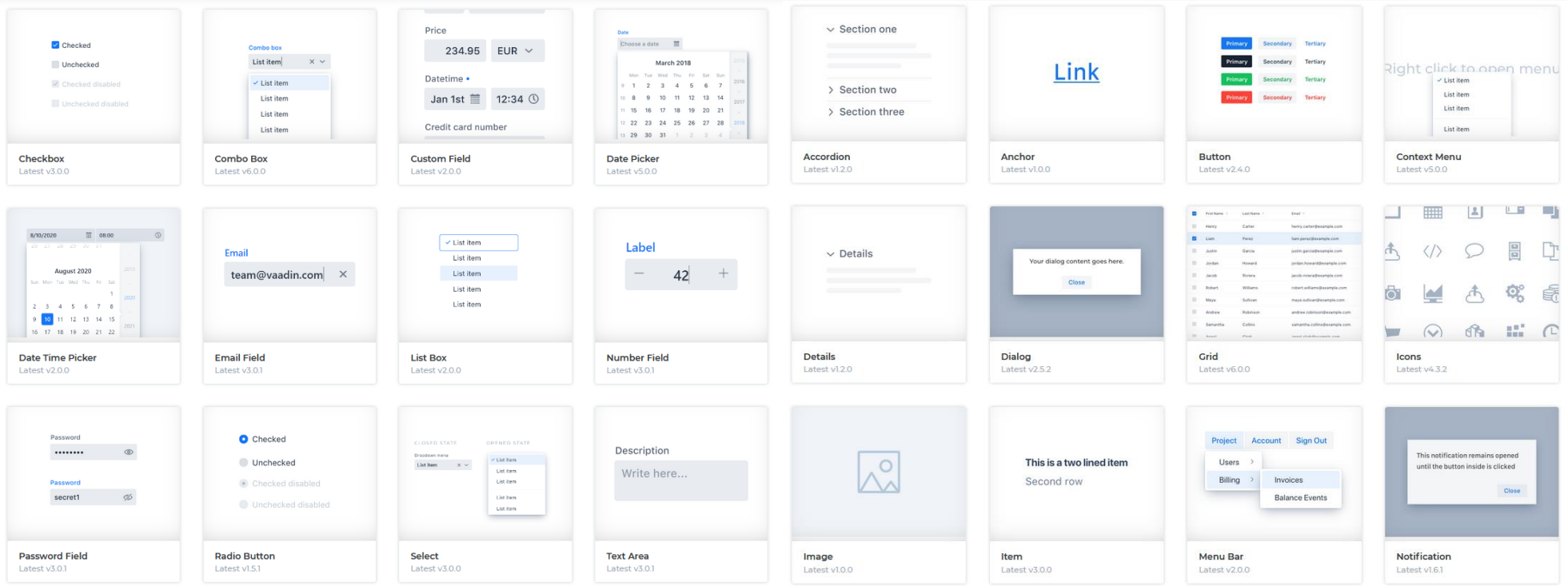


VAADIN

- An Open-Source Platform For Building Web Apps In Java
- Scalable UIs Without Extensive Knowledge of HTML, CSS, Javascript Or Any Other Modern Front-End Library / Framework (React,Vue.js,Angular)
- With This Framework We Handicaped Our Lack In Front-End Development Experience
- Very Powerful If Combined With Spring Boot
- Great Community (They Even Have A Discord Server Where They Help Devs In Their Vaadin Projects)

VAADIN

- Ready UI Components – 100% Java Compatible



TEST DRIVE 

Ώρα για δοκιμή !

Όλοι **οι φοιτητές (πολίτες)** μπορούν να συνδεθούν στο **external.hermesapp.xyz** και να κάνουν αιτήσεις στο πανεπιστήμιό μας (τον οργανισμό HUA).

Είναι καταχωρημένοι όλοι οι it217XXX και it218XXX χρήστες του τμήματος μας. Αν είστε από παλιότερο έτος, απλά επιλέξτε τυχαία έναν μητρώο εντός αυτού του εύρους.

Το Εσωτερικό Σύστημα βρίσκεται στην διεύθυνση **internal.hermesapp.xyz**

Ο κ. Τσαδήμας(tsadimas) είναι **διαχειριστής οργανισμών**. Αν θελήσει, μπορεί να δημιουργήσει έναν νέο οργανισμό και στην συνέχεια να διαχειριστεί τους/τις προϊσταμένους/ες του.

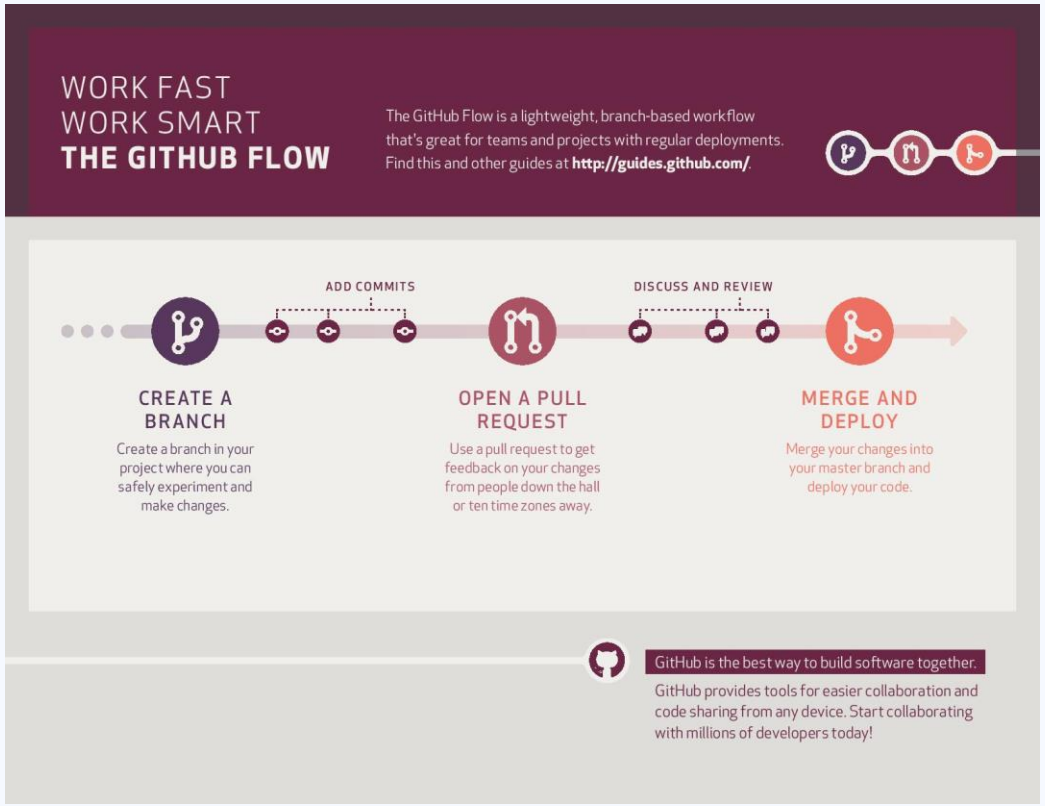
Ο κ. Μιχαήλ(michail) είναι **προϊστάμενος του HUA.** Μπορεί να διαχειριστεί τους υπαλλήλους του HUA.

Η κα. Μάρα(mara) είναι **υπάλληλος του HUA**. Μπορεί να δει τις αιτήσεις που καταθέτουν οι φοιτητές και να ενημερώνει την κατάστασή τους.

Κωδικός Πρόσβασης σε κάθε λογαριασμό είναι το **username** του account (πχ για το it21821 κωδικός είναι το it21821).

VCS

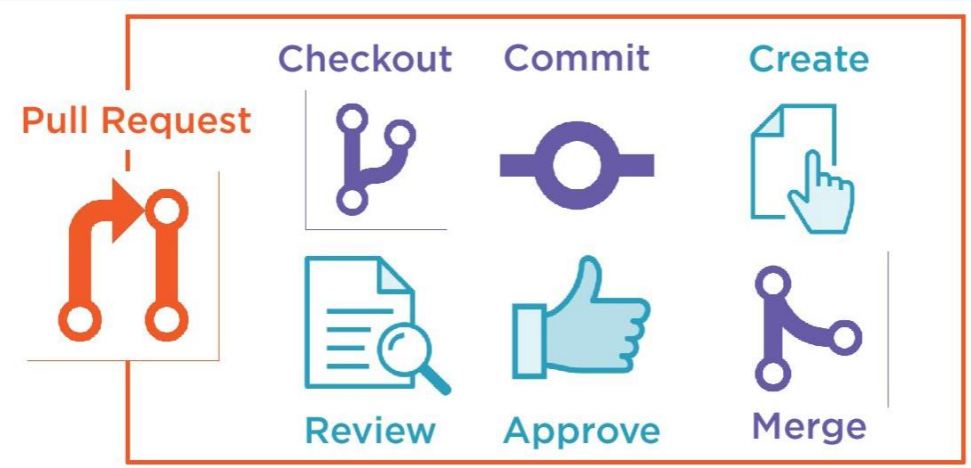
VERSION CONTROL (GIT & GITHUB)



- Version Control Systems (such as Git) play a significant role in DevOps.
- They enable developers to collaborate in a distributed manner.
- Git hosting providers such as GitHub offer extra tools for even more powerful collaboration & integration with other services.
- Hermes used the GitHub Flow as a workflow.

COLLABORATION

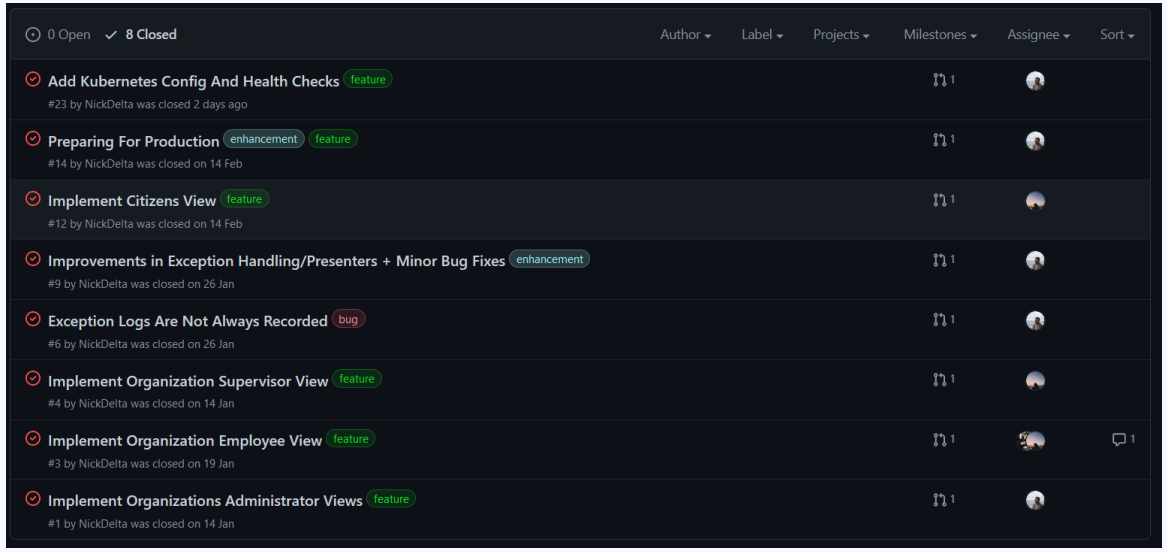
PULL REQUESTS



- New code must not be deployed to production before it is tested, reviewed and approved by the dev team.
- CI and Peer Reviews eliminate most chances of a fatal mistake.
- But how does each member of a team know what to do in a project ?

COLLABORATION

ISSUE TRACKING



Internal Frontend Issue List

- Issues are shared with each team member and are implemented as parallel as possible.
- An issue can be a new feature, an improvement, a bug fix or whatever you want !
- Issue tracking helps keeping a project history. For example, you may clearly see the decisions that were made throughout the development of this project.

DEVOPS




CI/CD PIPELINE OVERVIEW

```yaml
steps:
  # Use docker image because mvn cloud builder doesn't support jdk11
  - name: maven:3.6.3-jdk-11-slim
    entrypoint: 'mvn'
    # -Dhttp.keepAlive=false fixes a bug that causes cloud build to fail downloading dependencies
    args: ['vaadin:prepare-frontend', 'package', '-Pproduction', '-DskipTests','-Dhttp.keepAlive=false']
  - name: 'gcr.io/cloud-builders/docker'
    args: [ 'build', '-t', 'europe-west2-docker.pkg.dev/$PROJECT_ID/internal-front-end/prod:latest', '.']
  - name: 'gcr.io/cloud-builders/docker'
    args: [ 'build', '-t', 'europe-west2-docker.pkg.dev/$PROJECT_ID/internal-front-end/prod:$SHORT_SHA', '.']
  - name: 'gcr.io/cloud-builders/docker'
    args: [ 'push', 'europe-west2-docker.pkg.dev/$PROJECT_ID/internal-front-end/prod:latest' ]
  - name: 'gcr.io/cloud-builders/docker'
    args: [ 'push', 'europe-west2-docker.pkg.dev/$PROJECT_ID/internal-front-end/prod:$SHORT_SHA' ]
  - name: "gcr.io/cloud-builders/gke-deploy"
    args:
      - run
      - --filename=k8s
      - --image=europe-west2-docker.pkg.dev/$PROJECT_ID/internal-front-end/prod:$SHORT_SHA
      - --location=europe-west2-c
      - --cluster=hermes-cluster
```

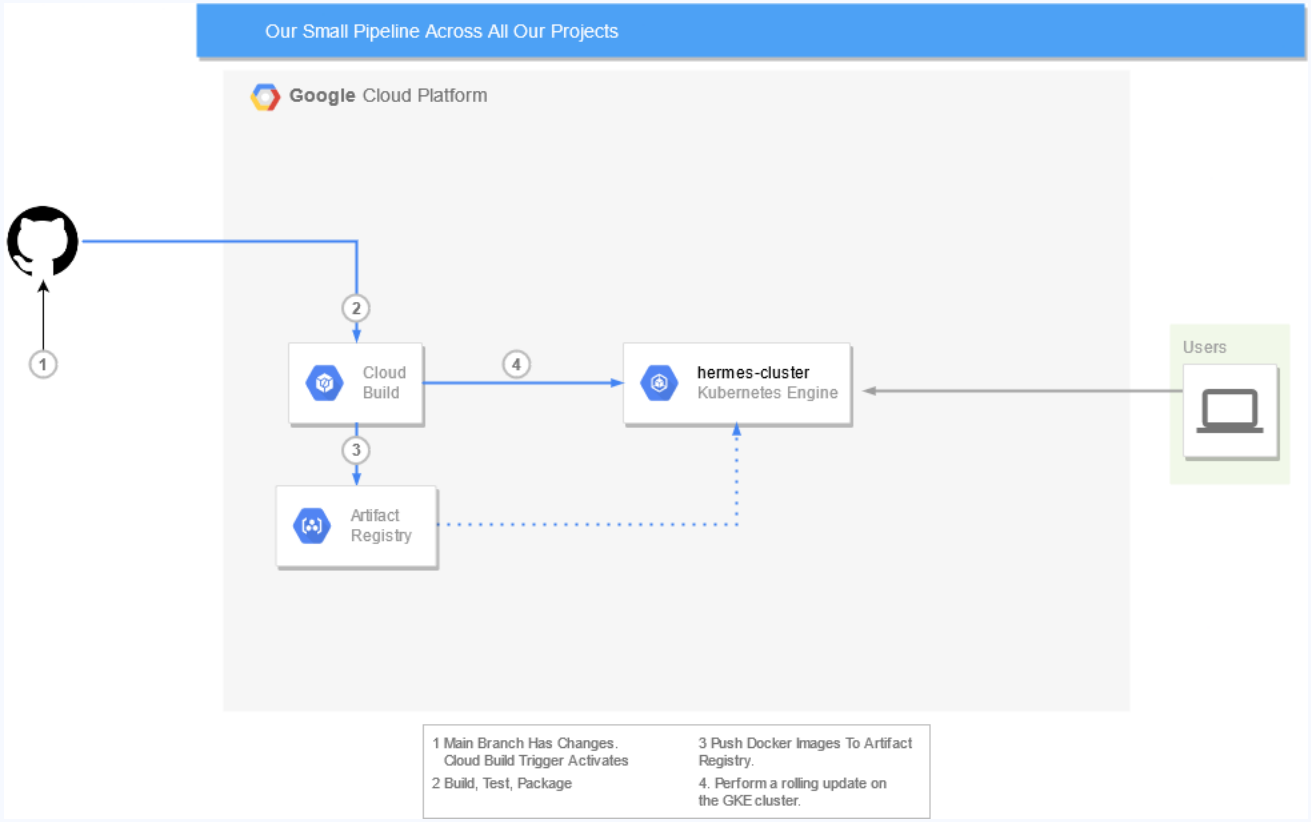
cloudbuild.yaml of the Internal Frontend

CI/CD Pipeline steps across all our projects:

1. Through Maven, the jar file containing the app is built.
2. A docker image is made using the jar file built in step 1 which has 2 tags : latest and the first 7 letters of the commit's SHA1 ($SHORT_SHA)
3. The docker image is pushed to Google Artifact Registry (equivalent of DockerHub)
4. A Rolling Update of the app is issued on the Google Kubernetes Engine (GKE) cluster.

DEVOPS

CI/CD PIPELINE OVERVIEW



DEVOPS

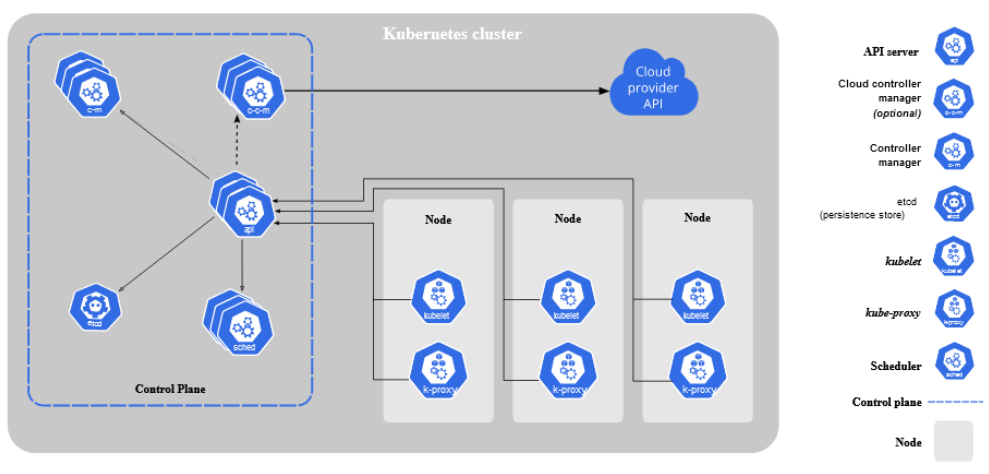
DOCKERFILE

```
1  FROM openjdk:11-jre-slim
2
3  # Install curl
4  RUN apt-get update && apt-get install -y curl
5
6  # Copy the jar file containing the app
7  COPY target/hermes-internal-front-end.jar app.jar
8
9  # Container must run in non-root mode
10 RUN groupadd hermes && useradd -g users -G hermes hermes
11 USER hermes
12
13 EXPOSE 8082
14 EXPOSE 9001
15
16 # Add healthcheck
17 HEALTHCHECK --interval=5s --timeout=20s \
18   CMD curl -f http://localhost:9001/actuator/health || exit 1
19
20 CMD java -jar app.jar
```

Dockerfile of the Internal Frontend

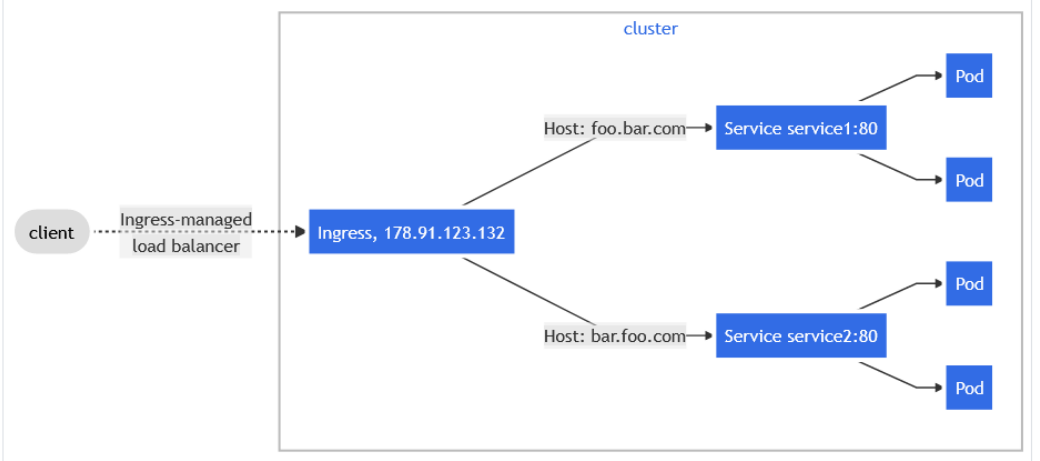
DEVOPS

KUBERNETES



High Level Kubernetes Architecture

On GCP, you get a Control Pane / month for free, you only pay for nodes at GCE pricing.

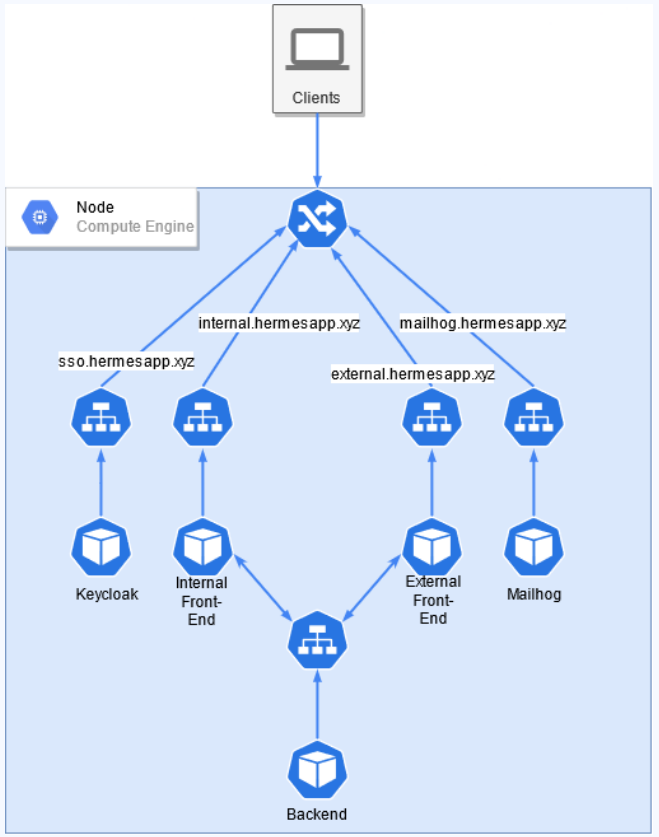


Name based Virtual Hosting

Name-based virtual hosts support routing HTTP traffic to multiple host names at the same IP address.

DEVOPS

HIGH LEVEL OVERVIEW OF OUR KUBERNETES ARCHITECTURE

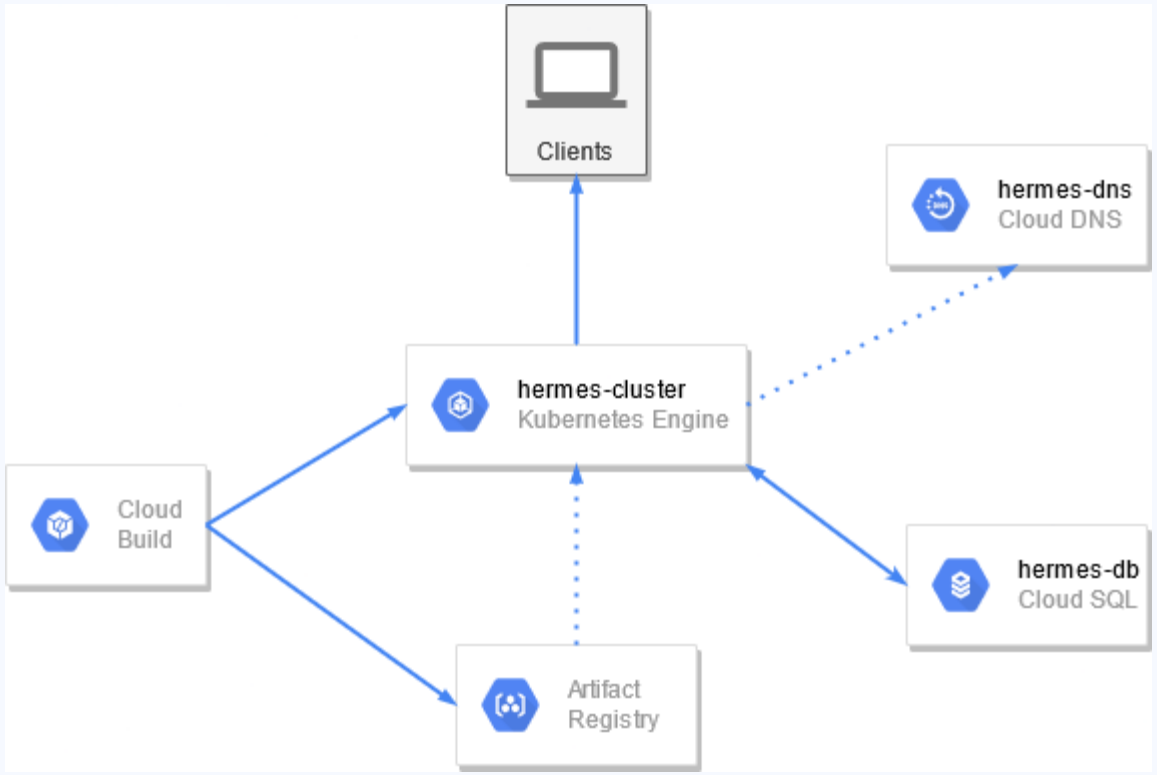



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: internal-frontend
  name: internal-frontend
  namespace: hermes
spec:
  replicas: 1
  selector:
    matchLabels:
      app: internal-frontend
  strategy: {}
  template:
    metadata:
      labels:
        app: internal-frontend
    spec:
      containers:
        - env:
            - name: HERMES_BACKEND_URL ···
            - name: KEYCLOAK_ADMIN_CLIENTID ···
            - name: KEYCLOAK_ADMIN_CLIENTSECRET ···
            - name: KEYCLOAK_AUTH_SERVER_URL ···
            - name: KEYCLOAK_CREDENTIALS_SECRET ···
          image: europe-west2-docker.pkg.dev/hua-hermes/internal-front-end/prod
          name: hermes-internal-frontend
          ports:
            - containerPort: 8082
            - containerPort: 9001
          resources: {}
          livenessProbe:
            httpGet:
              path: /actuator/health/liveness
              port: 9001
            initialDelaySeconds: 20
            periodSeconds: 5
          readinessProbe:
            httpGet:
              path: /actuator/health/readiness
              port: 9001
            initialDelaySeconds: 20
            periodSeconds: 5
      restartPolicy: Always
```

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: internal-frontend
  name: internal-frontend
  namespace: hermes
spec:
  ports:
    - name: "8082"
      port: 8082
      targetPort: 8082
  selector:
    app: internal-frontend
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: internal-frontend-ingress
  namespace: hermes
  annotations:
    kubernetes.io/ingress.class: nginx
spec:
  tls:
    - secretName: secret-tls
  rules:
    - host: "internal.hermesapp.xyz"
      http:
        paths:
          - pathType: ImplementationSpecific
            path: "/"
            backend:
              service:
                name: internal-frontend
                port:
                  number: 8082
```

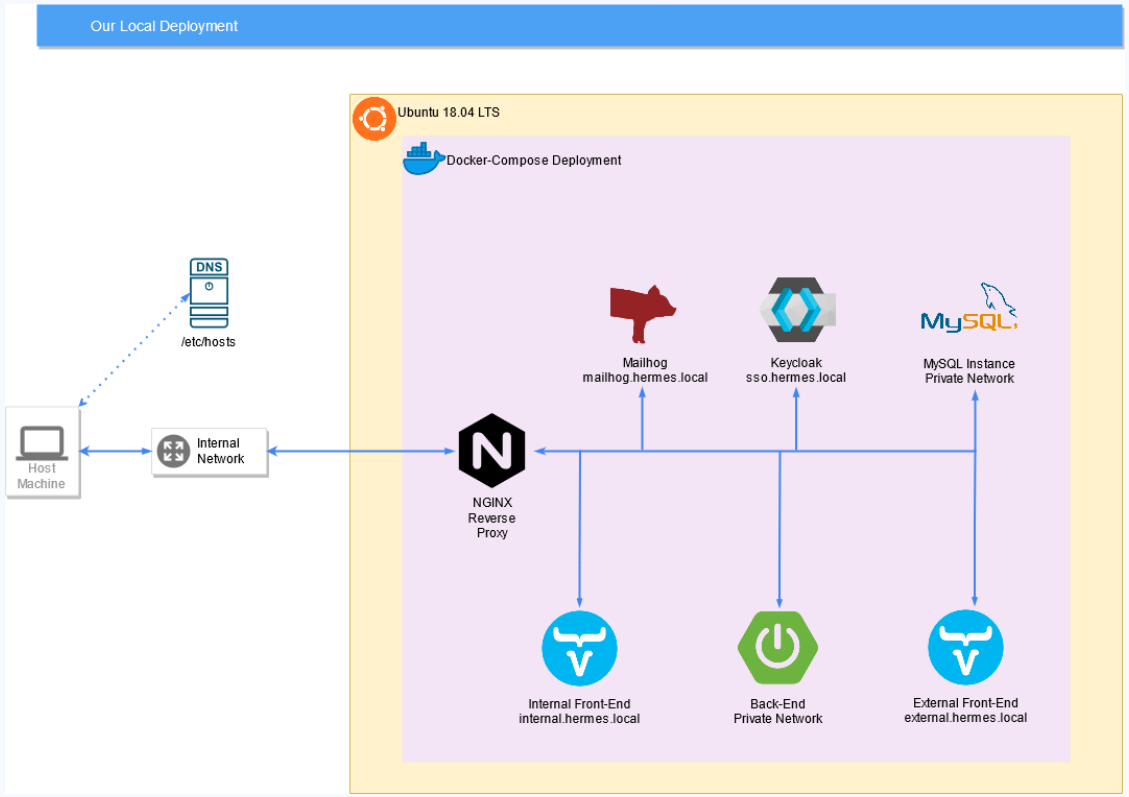
DEPLOYMENT: GCP



Basic Idea:

- Google Kubernetes Engine (GKE) for a fully-managed Kubernetes experience:
  - NGINX Ingress Controller
  - Kubernetes Dashboard
- Google Cloud SQL using MySQL 5.7
  - Keep DBs out of Kubernetes
- Google Artifact Registry for storing container images
- Google Cloud Build for CI/CD
- Google Cloud DNS for DNS Service

DEPLOYMENT: LOCAL



Basic Idea:

- Vagrant for building a VM on Virtualbox
  - Ubuntu 18.04 LTS
- Ansible for automating VM configuration
  - Installs Docker & Docker-Compose
  - Copies some essential files
  - Executes a docker-compose.yaml file which deploys the app
- Container images are still pulled from Google Artifact Registry
- NGINX for Reverse Proxy
- /etc/hosts for local DNS with some fake FQDNs.

DEPLOYMENT: LOCAL


```ruby
Vagrant.configure("2") do |config|
    config.vm.box = "hashicorp/bionic64"
    config.vm.hostname = "vagrantstation"
    config.vm.network :public_network, ip: "192.168.1.69" # Change the IP if you are on a different subnet

    config.vm.provider "virtualbox" do |v|
      v.memory = 4096
      v.cpus = 4
    end

    config.vm.provision "shell" do |s|
        ssh_pub_key = File.readlines(File.join(Dir.home, ".ssh/id_rsa.pub")).first.strip
        s.inline = <<-SHELL
          echo #{ssh_pub_key} >> /home/vagrant/.ssh/authorized_keys
        SHELL
      end
end
```

Vagrantfile

DEPLOYMENT: LOCAL


```yaml
- name: Setup hermes on Vargrant VM
  hosts: Servers
  become: yes
  pre_tasks:
  - name: Update apt packages
    apt:
      update_cache: yes
  tasks:
  - name: Install docker && docker-compose
    vars:
      docker_service_state: started
      docker_service_enabled: true
      docker_install_compose: true
      docker_compose_path: /usr/local/bin/docker-compose
    # See https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html#including-roles-dynamic-reuse
    include_role:
      name: geerlingguy.docker
  - name: Deploy app
    block:
    - name: Create hermes directory
      file:
        path: /usr/src/hermes
        state: directory
        mode: '0755'
    - name: Create nginx conf directory
      file:
        path: /usr/src/hermes/nginx
        state: directory
        mode: '0755'
    - name: Copy docker-compose file from host to vm
      copy:
        src: files/docker-compose.yml
        dest: /usr/src/hermes
        mode: '0644'
    - name: Copy nginx file from host to vm
      copy:
        src: files/nginx/default.conf
        dest: /usr/src/hermes/nginx/default.conf
        mode: '0644'
    - name: Run docker-compose
      command: docker-compose up -d
      args:
        chdir: /usr/src/hermes
      changed_when: true
```

Ansible Playbook

DEPLOYMENT: LOCAL

```yaml
version: "3.3"

services:

  db:
    container_name: "hermes-mysql"
    image: "mysql:5.7"
    volumes:
      - "hermes-mysql-data:/var/lib/mysql"
    environment:
      MYSQL_ROOT_USER: root
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: keycloak # Create an empty keycloak schema
    restart: unless-stopped

  keycloak:
    container_name: "hermes-keycloak"
    depends_on:
      - db
    image: "europe-west2-docker.pkg.dev/hua-hermes/keycloak/prod"
    # Both internal and external requests are being made to keycloak. Sc
    # Change the HTTP port to 80 to make the alias truly work
    command: ["-b 0.0.0.0", "-Dkeycloak.migration.action=import", "-Dkey
    networks:
      default:
        aliases:
          - sso.hermes.local
    environment:
      KEYCLOAK_USER: admin
      KEYCLOAK_PASSWORD: admin
      DB_ADDR: hermes-mysql
      DB_VENDOR: mysql
      DB_USER: root
      DB_PASSWORD: root
    restart: unless-stopped

  backend:
    container_name: "hermes-backend"
    depends_on:
      - db
      - keycloak
    image: "europe-west2-docker.pkg.dev/hua-hermes/back-end/prod"
    environment:
      DB_URL: hermes-mysql
      DB_SCHEMA: hermes
      DB_USER: root
      DB_PASS: root
      KEYCLOAK_AUTH_SERVER_URL: http://sso.hermes.local/auth
      KEYCLOAK_CREDENTIALS_SECRET: 81f7d18a-b8b6-4ff9-86b1-ce6e18cb097b
    restart: unless-stopped

  intenal-frontend:
    container_name: "hermes-internal-frontend"
    depends_on:
      - db
      - keycloak
      - backend
    image: "europe-west2-docker.pkg.dev/hua-hermes/internal-front-end/prod"
    environment:
      KEYCLOAK_AUTH_SERVER_URL: http://sso.hermes.local/auth
      KEYCLOAK_CREDENTIALS_SECRET: 9855054d-647a-426b-902c-9d9f539027bf
      KEYCLOAK_ADMIN_CLIENTID: keycloak-admin
      KEYCLOAK_ADMIN_CLIENTSECRET: 71ff7d46-ad8f-4b52-8107-00415c07f0d4
      HERMES_BACKEND_URL: http://hermes-backend:8080
    restart: unless-stopped

  external-frontend:
    container_name: "hermes-external-frontend"
    depends_on:
      - db
      - keycloak
      - backend
    image: "europe-west2-docker.pkg.dev/hua-hermes/external-front-end/prod"
    environment:
      KEYCLOAK_AUTH_SERVER_URL: http://sso.hermes.local/auth
      KEYCLOAK_CREDENTIALS_SECRET: 81aac51a-17cf-4bff-8983-97f90403ee0a
      KEYCLOAK_ADMIN_CLIENTID: keycloak-admin
      KEYCLOAK_ADMIN_CLIENTSECRET: 71ff7d46-ad8f-4b52-8107-00415c07f0d4
      HERMES_BACKEND_URL: http://hermes-backend:8080
    restart: unless-stopped

  nginx:
    container_name: hermes-nginx
    image: nginx:stable-alpine
    volumes:
      - ./nginx/default.conf:/etc/nginx/conf.d/default.conf
    ports:
      - 80:80
    restart: unless-stopped

  mailhog:
    container_name: "hermes-mailhog"
    image: "mailhog/mailhog:v1.0.1"
    restart: unless-stopped

volumes:
  hermes-mysql-data:
```

docker-compose.yaml

DEPLOYMENT: LOCAL

```
server {

        location / {
                proxy_set_header X-Forwarded-Host $host;
                proxy_set_header X-Forwarded-Server $host;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header Host $host;
                proxy_pass http://hermes-internal-frontend:8082;
        }

        server_name internal.hermes.local;
        listen 80;
        listen [::]:80;

}

server {

        location / {
                proxy_set_header X-Forwarded-Host $host;
                proxy_set_header X-Forwarded-Server $host;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header Host $host;
                proxy_pass http://hermes-external-frontend:8083;
        }

        server_name external.hermes.local;
        listen 80;
        listen [::]:80;

}
```

```
server {

        location / {
                proxy_pass http://hermes-keycloak;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-Proto $scheme;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Scheme $scheme;
        }

    server_tokens off;
    server_name sso.hermes.local;
    listen 80;
    listen [::]:80;

}

# See https://gist.github.com/blakethepatton/7928abb131593fd192606b681ee47b2b
server {

        location / {
                proxy_pass      http://hermes-mailhog:8025;
                proxy_set_header    Host            $host;
                proxy_set_header    X-Real-IP       $remote_addr;
                proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header    X-Client-Verify SUCCESS;
                proxy_set_header    X-Client-DN     $ssl_client_s_dn;
                proxy_set_header    X-SSL-Subject   $ssl_client_s_dn;
                proxy_set_header    X-SSL-Issuer    $ssl_client_i_dn;
                proxy_read_timeout 1800;
                proxy_connect_timeout 1800;
                chunked_transfer_encoding on;
                proxy_set_header X-NginX-Proxy true;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection "upgrade";
                proxy_http_version 1.1;
                proxy_redirect off;
                proxy_buffering off;
        }

        server_name mailhog.hermes.local;
        listen 80;
        listen [::]:80;
}
```

default.conf (NGINX)