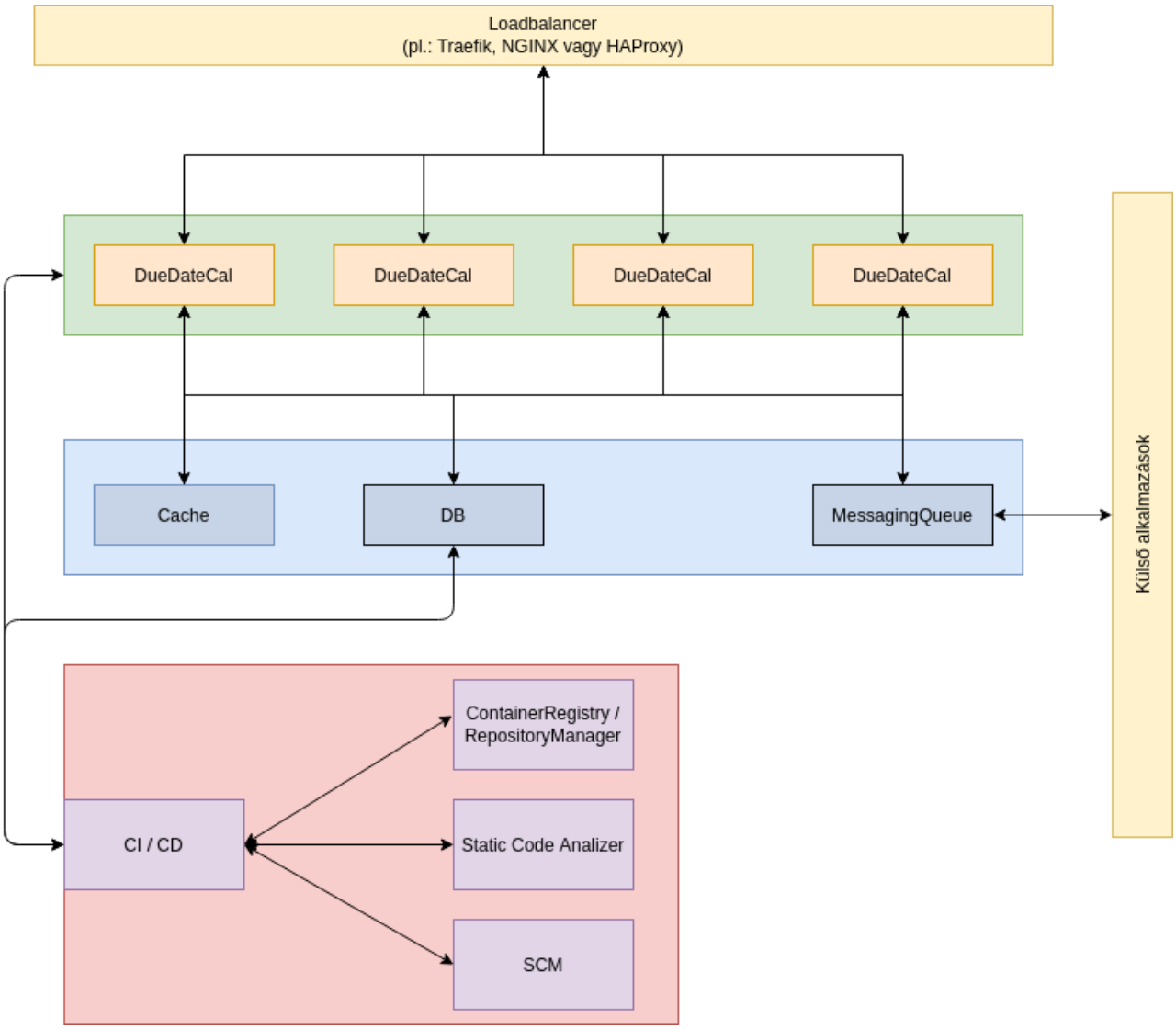
DueDateCalendar Service

Ahogy a feladatkiírásban kértétek igyekeztem megtervezni egy magas rendelkezésre állású rendszert. A felépítést az alábbi ábrán láthatjátok:



Elképzelésem szerint konténerizáltan kerülnének futtatásra az alkalmazás példányok így jobb hardveres kihasználtságot érhetünk el. Az alkalmazások skálázását valamilyen orchestration rendszer végezné (pl.: Kubernetes) amelyben az igényeknek megfelelően lehet a végrehajtási csúcsokat és a skálázási küszöböket beállítani.

Akár konténeres akár hagyományos (pl. systemd) szolgáltatásként futtatjuk az alkalmazásokat egy terheléselosztó mindenképpen szükséges az alkalmazások elé. Ez egyrészt elrejti az alkalmazások felületét az Internetről érkező forgalom elől ami biztonságosabbá teszi a működést, másrészt pedig az akár az alkalmazások terheltségéhez mérten tereli a forgalmat a futó példányok között.

Az alkalmazások egymás közötti adatok megosztásához ha egy hagyományos adatbázis lassúnak bizonyulna egy cache alkalmazást használhatnának, az adatok perzisztens tárolásához pedig egy NoSQL adatbázist választanék. Amennyiben szükséges, hogy a DueDateCalendar példányok más, a

céges hálózaton belül futó alkalmazásokkal is kommunikálhassanak erre egy MessagingQueue implementációt választanék. Így egy másik, pl. adminisztratív adatokhoz a DueDateCalendar adatait használó alkalmazásunk anélkül kérhetne adatokat / küldhetne üzeneteket, hogy az tüskeszerű terhelésként jelentkezne.

A rendszer telepítéséhez valamilyen Infrastructure as a Code megoldást választanék (mondjuk Ansible) amely segítségével könnyen és gyorsan húzhatunk fel új virtuális gépeket ha a szükség úgy kívánja. Public cloud esetében (vagy akár privát rendszer esetében VMWare mellett) Terraformmal kombinálva teljes mértékben automatizálható az infrastruktúránk építése és üzemeltetése.

Az alkalmazásokat persze oda is kell juttatni valahogy a fejlesztői műhelyből. A fejlesztők számára mindenképpen szükséges egy központi source controll management system – lehetőleg Git alapú – amelyre valamilyen CI / CD rendszert ültetnék. A futtatható állományok / konténer imagek elkészítése mellett ez a rendszer végezné az alkalmazások tesztelését. A tesztelésbe a megírt unit / integrációs / e2e tesztek futtatását és statikus kód analízését, valamint e2e tesztek megléte esetén biztonsági ellenőrzését értem. Ez utóbbit mondjuk az OWASP ZAP proxy elláthatná amelyet Selenium tesztek és az alkalmazásunk közé mint proxy-t köthetünk be és míg a tesztünk békésen kattintgat addig a ZAP az általa ismert módokon próbálja megtörni a felületeket.

A tesztfutás adatait felhasználva quality gateket lehetne bevezetni amelyek még időben megfoghatják a hibás implementációkat.