TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Tarkvarateaduse instituut

Enrico Vompa 185787IAIB

Õppetöö läbiviimiseks kasutatav TalTechi õppesüsteemi automaattestimissüsteem tudengi koodi esituste jaoks

Bakalaureusetöö

Juhendaja: Ago Luberg

PhD

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed, on töös viidatud.

Autor: Enrico Vompa

17.05.2021

Annotatsioon

Parim viis programmeerimiskeelte, programmeerimise paradigmade, algortimide ning andmestruktuuride õppimiseks on neid läbi lahendades. Seda lähenemist rakendatakse TalTechi IT teaduskonnas mitmetes õppeaines.

Antud lähenemise eeldus on kiire tehniline tagasiside, ühene standard ning integratsioon eksisteerivate süsteemidega, kus tagasiside on saadaval. Peamised on TalTech Moodle keskkond ja mail.

Töö eesmärk on pakkuda koodi testimise teenust, mis saab sisendiks tudengi koodi, testid ning valikulise konfiguratsiooni ja annab väljundiks testitulemused standardsel kujul, mida eksisteerivad süsteemid toetavad ning seda skaleeruval viisil.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 41 leheküljel, 7 peatükki, 19 joonist, 1 tabel.

Abstract

AUTOMATED TESTING SYSTEM FOR STUDENT CODE USED IN TALTECH LEARNING SYSTEM

The best way of learning programming languages, programming paradigms, algorithms and data structures is by implementing them. This approach is used in many subjects in the TalTech faculty of IT.

The prerequisite for this method is fast technical feedback and integration with existing systems. Main integrations being with TalTech Moodle and mail.

The goal of this thesis is to provide an automated testing service, which takes the student's code and the tests as an input, and outputs the test results in a standardised format which is supported by existing systems in a scalable fashion.

The thesis is in Estonian language and contains 41 pages of text, 7 chapters, 19 figures, 1 table.

Lühendite ja mõistete sõnastik

Alamtester	Tõmmise peale ehitatud konteiner, kus toimub tudengi koodi testimine
API	*Application programming interface.* Rakendustarkvara liides
CPU	*Central processing unit.* Keskprotsessor
CRUD	*Create, read, update and delete.* Operatsioonid
Git	Versioonihaldustarkvara
HTML	*HyperText Markup Language.* Hüperteksti märgistuskeel
IDE	*Integrated Development Environment*. Tarkvaraarenduse keskkond
IEEE	*Institute of Electrical and Electronics Engineers*. Organisatsioon
JDBC	*Java Database Connectivity.* API andmebaasiga suhtlemiseks
JSON	*JavaScript Object Notation.* Formaat andmete esitamiseks
JPA	*Java Persistence API.* Andmebaasi ning Java programmi vaheline liides
ORM	*Object-relational Mapping.* Objekt-relatsiooniline kaardistamine
POM	*Project Object Model.* Projekti objektimudel
POJO	*Plain Old Java Object.* Java objekt
REST	*Representational state transfer.* Tarkvaraarhitektuuri standard
SaaS	*Software as a Service.* Tarkvara teenusena
SQL	*Structured Query Language.* Relatsiooniliste andmebaasidega suhtlemiseks kasutatav päringukeel
XML	*Extensible Markup Language.* Formaat andmete esitamiseks
YAML	*Human-readable data-serialization language.* Formaat andmete esitamiseks

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Programmeerimine on probleemide lahendamine läbi koodi. Üks viis probleemide lahendamise oskuse arendamiseks on ülesandeid läbi lahendada, mida praktiseeritakse TalTechis näiteks järgnevates õppeainetes (sulgudes ainekood ja kursusel kasutatavad testrid) esimese jaanuari seisuga 2021:

- Programmeerimise täiendusõpe (ITI0002; python_tester)
- Programmeerimise algkursus (ITI0102; python_tester)
- Programmeerimise põhikursus (ITI0202; java15_tester)
- Loogiline programmeerimine (ITI0211; prolog_tester)
- Algoritmid ja andmestruktuurid (ITI0204; java11_tester)
- Programmeerimise erikursus (ITI0214; hackerrank-, kattis- ja uva_tester)
- Programmeerimise erikursus (ITT8060; fsharp_tester)
- Programmeerimine C++ keeles (YFX0505; c_tester)

Selle meetodi teeb edukaks esitatud koodile põhjaliku õigeaegse tagasiside saamine, mis võimaldab näha kirjutatud koodi funktsionaalsuse vastavust ülesandes soovitule. Aine läbiviimist lihtsustab testri tagasiside koodile, sest siis tudengid teavad täpsemalt, kust viga otsida. Lisaks ei pea õppejõud igat tudengit individuaalselt aitama, mis hoiab kokku ressursse ja võimaldab õpetada rohkematele tudengitele oluliselt rohkem.

Töö eesmärgiks on luua tester (edaspidi Arete), mis vastab kooli nõuetele ja vajadustele ning mida saab tudengi koodi testimiseks kasutada läbi API.

Antud töö koosneb järgnevatest osadest: esmalt viiakse läbi taustauuring, kus tutvutakse eelmise testriga ja teiste koodi kirjutamise platvormidega. Seejärel tutvustatakse nõudeid projektile ning tehakse ülevaade projekti arhitektuurist ning kasutatavatest tehnoloogiatest. Lõpetuseks räägitakse publitseerimiseks kasutatud tehnoloogiatest.

2 Taustauuring

Selles peatükis võrreldakse erinevate testrite positiivseid ja negatiivseid külgi.

2.1 Eelmine tester

Enne käesoleva töö tegemist oli kasutusel eelmine tester[1], millel oli puudujääke. Üheks puudujäägiks oli halb ressursside jaotus, põhjustades ebaoptimaalse töövoo, kus ühe töö testimine võttis märgatavalt kauem aega, kui peaks. Tihtipeale lõpetas tester funktsioneerimise kontrolltöö või eksami ajal. Eelmine tester polnud ka jälgitav, sest andmebaasi midagi olulist ei salvestatud. Lisaks oli palju ebavajalikku funktsionaalsust, mis tegi arenduse keeruliseks.

Sellest tulenevalt ka nõuded, mida uus tester täitma peab:

- Õppejõud saab lisada enda ülesandeid, mis jõuavad kohe testrisse.
- Õppejõud ning arendajad saavad uuendada alamtestreid, mis jõuavad kohe testrisse.
- Tudengi koodi saab üksikasjalikult testida üksustestidega.
- Tester peab olema jälgitav ning uuendatav, et võimaldada uudsemaid ja huvitavamaid ülesandeid, soodustades alamtestrite arendust.
- Testril peab olema hea ressursside jaotus, et see kriitilisel hetkel jääks töötama.

Eelmine tester andis hea ettekujutuse potentsiaalsetest kitsaskohtadest, mida vältida ning tehnoloogiatest ja raamistikest, mida kasutada.

[1] "api", 2019, GitLab repository, https://gitlab.cs.ttu.ee/testing/api.

2.2 Eksisteerivad platvormid

ICPC[2] ja IEEEXtreme[3] võistlusel oli testri tööpõhimõte hästi läbi mõeldud, sest võistluse jooksul, kus osales üle 50 000 üliõpilase, ei tekkinud ühtegi probleemi seoses koodi testimisega. Sealsetel testritel oli võimalus testida koodi paljudes keeltes aga puuduseks oli see, et testida sai ainult läbi standardsisendi ja väljundi, mis tähendab, et üksustestide kirjutamine on tudengi koodile võimatu. Sellise testimise tagajärjel toimib tudengi kood musta kastina, sest seal sees tehtavad operatsioonid jäävad selgusetuks. Koodi kirjutamise platvorme on tänapäeval mitmeid. Igal ühel neist on omad aspektid:

Exercism[4]

- \+ Avatud lähtekoodiga
- \+ Enam kui sada programmeerimiskeelt
- − Testimine toimub puhtalt läbi standardsisendi ja -väljundi.

Codecademy[5]

- \+ Saab ka koodi jooksmist jälgida reaalajas.
- − Integreerimine Moodlega pole võimalik (Kinnine lähtekood)
- − Õppejõud ei saa lisada uusi ülesandeid.

Hackerrank[6]

- \+ Palju eksisteerivaid ülesandeid
- \+ Läbi partnerluse saab õppejõud ülesandeid lisada.
- − Integreerimine otse Moodlega pole võimalik
- − Testimine toimub läbi standardsisendi ja -väljundi.

CMS[7]

- \+ Avatud lähtekoodiga
- − Testimine ainult läbi standardsisendi ja -väljundi

[2] "The ICPC International Collegiate Programming Contest." https://icpc.baylor.edu/.
[3] "IEEEXtreme – 24-Programming Competition." https://ieeextreme.org/.
[4] "Exercism." https://exercism.io/.
[5] "Codecademy." https://www.codecademy.com/.
[6] "HackerRank." https://www.hackerrank.com/.
[7] "cms-dev/cms: Contest Management System - GitHub." https://github.com/cms-dev/cms. Accessed 28 Feb. 2021.

Moodle CodeRunner[8]

+ Eksisteeriv integratsioon Moodlega
− Puudub integratsioon GitLab'iga
− Ebamugav, Ei saa IDEt kasutada

Kõik eelnevad koodi kirjutamise platvormid on hästi läbimõeldud ja täidavad oma funktsiooni, aga ükski ei täida kõiki nõudeid ning nõude lisamisel oli ületamatu takistus, kus nõueteks olid:

+ Integratsioon Moodle ja GitLab'iga
+ Testimine üksustestidega
+ Ülesannete manuaalne lisamine

2.3 Kokkuvõte

Eeltöö andis hea ettekujutuse vigadest, mida vältida ja headest omadustest, mida implementeerida. Taustauuringust sai järeldada järgmist:

- Olemasolev tester on vaja välja vahetada.
- Teiste kirjutatud testreid kasutada ei saa, sest testrid pole kas avatud lähtekoodiga või ei täida kooli nõudeid.
- Tester tuleks liidestada väliste süsteemidega, et suurendada võimalike ülesannete arvu, mida kasutada erinevates ainetes. Valituks said HackerRank[9], Kattis[10] ning OnlineJudge[11].

Peale taustauuringu tegemist algas analüüs, kus vormistati kooli nõuded ümber funktsionaalseteks ja mittefunktsionaalseteks nõueteks.

[8] "Moodle plugins directory: CodeRunner." https://moodle.org/plugins/qtype_coderunner. Accessed 28 Feb. 2021.
[9] "HackerRank." https://www.hackerrank.com/. Accessed 1 Jan. 2021.
[10] "Kattis." https://open.kattis.com/. Accessed 6 Jan. 2021.
[11] "Online Judge." https://onlinejudge.org/. Accessed 1 Jan. 2021.

3 Analüüs

Käesoleva töö analüüsi etapis koostati funktsionaalsed ja mittefunktsionaalsed nõuded.[12]

3.1 Funktsionaalsed nõuded

Funktsionaalsed nõuded määravad infosüsteemi omaniku vajadused sisendinfo muutmiseks väljundinfoks. [1]

Järgmisena on välja toodud funktsionaalsed nõuded:

- Administraator saab näha tööde ajalugu ja detaile iga töö kohta.
- Administraator ning arendaja saab uusi alamtestereid lisada ja uuendada.
- Administraator ning õpetaja saab uusi ülesandeid lisada ja uuendada.
- Kasutajad saavad anda sisendparameetreid testimiseks.
- Kasutajad saavad testida koodi sünkroonselt, kus päringu (*request*) vastuseks on testi tulemused, ning asünkroonselt, kus saadetakse tagasiside tema valitud URL-ile (näiteks moodle.taltech.ee[13]) ja mailile.
- Kasutajad saavad testida oma koodi igas keeles, millele on alamtester olemas.
- Õpetajad saavad kasutada ülesandeid välistest süsteemidest (Online Judge, Kattis ning HackerRank).

[12] "Functional Requirements vs Non Functional Requirements ...." https://www.guru99.com/functional-vs-non-functional-requirements.html.
[13] "TalTech Moodle." https://moodle.taltech.ee/?lang=en. Accessed 28 Feb. 2021.

3.2 Mittefunktsionaalsed nõuded

Mittefunktsionaalsed nõuded täpsustavad tarkvara ja nõuete projekteerimises kriteeriume, mille alusel hinnatakse süsteemi töökäiku. [2]

Järgmisena on välja toodud ja lahti seletatud mittefunktsionaalsed nõuded:

- Kättesaadavus: Teenuse funktsionaalsed nõuded peavad olema kättesaadavad.
- Konfidentsiaalsus: Kasutaja saab näha ainult seda, milleks tal ka õigus on.
- Terviklikkus: Peale süsteemi uuenduse läbiviimise ei tohi midagi salvestatut kaduma minna ning kõiki alamtestreid peab uuendama vastavalt.
- Turvalisus: Testitaval koodil ei tohi olla ligipääsu testrile endale.
- Hallatavus: Süsteemi on lihtne üles panna, seadistada ning hallata.

Testri koormus on suur läbi semestri, mida illustreerib järgmine SQL lause joonisel 1:

```
SELECT to_timestamp(cast(timestamp/1000 as bigint))::date as date, count(*) as submissions
FROM "submission"
GROUP BY to_timestamp(cast(timestamp/1000 as bigint))::date
ORDER BY submissions desc
LIMIT 10
```

date	submissions
2020-09-21	5372
2020-11-03	4506
2020-10-01	3852
2020-11-05	3829
2020-09-22	3537
2020-10-19	3443
2020-09-28	3363
2020-09-29	3223
2020-11-16	3159
2020-10-26	3148

Joonis 1: 10 suurima testimise koormusega päeva 1. jaanuari 2021 seisuga.

Nagu tulemusest näha, siis ükski eksami päev ei ole isegi esi kümnes tööde arvu poolest, mis tähendab, et tester peab pidama vastu suurtele koormustele iga päev.

4 Arendus

Selles peatükis kirjeldatakse lähemalt antud teenuse arhitektuuri ning seal kasutatud tehnoloogiaid.

4.1 Arhitektuur

Süsteemi arhitektuur on loogikakomponentide ja nende vaheliste seoste ülesehitus. [3] Arete on toote poolest *SaaS*, kus klientideks on peamiselt õppejõud, kes annavad programmeerimise aineid ning tudengid, kes osalevad nende ainete õppetöös. *SaaS* on standardne arhitektuurimuster veebirakenduste jaoks. Server on jaotatud alamkomponentideks, mis täidavad spetsiifilisi ülesandeid.

Arete[14] koosneb kahest peamisest alamkomponendist. Autentimisteenus ning koormuse tasakaalustamise teenus.

4.1.1 Autentimisteenus

Autentimisteenus kontrollib, et testri funktsionaalsust kasutaks ainult selleks autoriseeritud kasutaja. Hetkel on ära defineeritud kasutajad: Admin (süsteemi administreeriv isik), Developer (arendaja), Tester (testimist kasutav klient nagu Moodle või Charon) ja Hook (ülesandeid, teste või alamtestreid uuendav klient nagu GitLab).

Teenus loob ka ühenduse andmebaasiga, et hoida kõiki testimiselt tulnud tulemusi püsimälus ja kõige uuemaid tulemusi ka vahemälus, mida kuvatakse kasutajaliideses. Lisaks talletatakse vahemälus ka agregeeritud tulemusi kõikide testimistulemuste kohta mida samuti kuvatakse kasutajaliideses. Pikemalt räägitakse sellest alampeatükis “Vue”.

Teenus vahendab ka päringuid avaliku API ja koormuse tasakaalustamise teenuse vahel.

[14] "Arete - automated testing service — IT doc 1.0 ... - TalTech." https://ained.pages.taltech.ee/it-doc/arete/index.html. Accessed 1 Jan. 2021.

4.1.2 Koormuse tasakaalustamise teenus

Koormuse tasakaalustamise teenus hoiab töid järjekorras ning jooksutab neid paralleelselt alamtestrites. Jooksutamiseks on vaja:

- Laadida alla kõige uuem tudengi kood Git repositooriumist või kasutada päringu kehas olevaid faile.
- Piirata ühe tudengi poolt kasutatavate ressursside kogust. Iga tudeng saab jooksutada ainult ühte tööd korraga.
- Lugeda päringu kehast ning testide failidest süsteemi ning testimise konfiguratsioon. Antud konfiguratsioon võib olla jagatud mitme faili vahel. Detailsemalt on räägitud konfiguratsiooni parameetritest teenuse kasutusjuhendis[15].
- alamtester seadistada koos konteinerile rakendatud piirangutega ning testimiseks vajalike ressurssidega. Rohkem sellest alampeatükis “docker-java”.
- alamtesteri seest tulemused lugeda ning taastada seis, mis oli enne testimist.
- Koostada HTML ning JSON formaadis tulemused.
- Saata tudengile mail HTML formaadis tulemusega, vajadusel hoopis veasõnum ning ka artefaktid, mis tekkisid testimise käigus. Näiteks *.png* faililaiendusega pilt roboti teekonnast.
- Saata autentimisteenusele ning päringu koostajale eelnevalt mainitud tulemused.

Eduka töö tagamiseks peavad tööd olema üksteisest täielikult eraldatud ning üks kasutaja ei või kõike süsteemi ressursse ära kasutada.

[15] "Arete - automated testing service — IT doc 1.0 ... - TalTech." https://ained.pages.taltech.ee/it-doc/arete/test-runner/index.html#usage. Access date: 7 may. 2021.

4.2 Serverirakendus

Serverirakendus pakub teenust teistele serverirakendustele, oodates päringuid ja vastates neile. Andmevahetuseks kasutab Arete REST API liidest. [4]

4.2.1 REST API

Representational State Transfer (edaspidi REST[16]) on tarkvaraarhitektuuri standard, mis seab veebirakenduse loomisele kindlad piirid ja neid veebirakendusi kutsutakse RESTful veebirakendusteks. REST arhitektuuri põhiidee seisneb selles, et on olemas lõpp-punktid erinevatele CRUD[17]-toimingutele. Arete kasutab neist kolme: GET, mis tagastab ressursse, POST, mis lisab ressursse ja PUT, mis uuendab ressursse. [5]

Aretes on kasutusel 2021 aasta alguse seisuga 24 lõpp-punkti[18] millega saab teha järgmiseid tegevusi:

- Autentida end ning saada token, et teisi lõpp-punkte kasutada
- Lisada, muuta ning kustutada kasutajaid ning nende õiguseid
- Saada ülevaade Arete poolt kasutatavatest ressurssidest
- Saata testimisele uus töö
- Näha viimaseid testimisi ning detaile nende kohta
- Näha koondülevaadet tudengitest, ülesannetest ning kursustest
- Uuendada ülesannet
- Uuendada testimiseks kasutatavat tõmmist

Vastuseks saab klient HTTP koodi, mis annab tagasisidet toimingu õnnestumise kohta ja sõltuvalt operatsioonist ka JSON objekti vastuse kehas. [6]

---

[16] "What is REST – Learn to create timeless REST APIs." https://restfulapi.net/.
[17] "What are CRUD Operations? Examples, Tutorials & More." https://stackify.com/what-are-crud-operations/.
[18] "Swagger UI - TalTech." https://cs.ttu.ee/services/arete/api/v2/swagger-ui/index.html?configUrl=/services/arete/api/v2/docs/swagger-config. Accessed 1 Jan. 2021.

4.3 Serverirakenduse tehnoloogiad

Antud peatükis mainitud serverirakenduse tehnoloogiad said valitud tänu autori varasemale kogemusele.

4.3.1 Vue

Vue.js[19] raamistiku abil sai realiseeritud graafiline kasutajaliides, mille kaudu saab ligi Arete lõpp-punktidele, mis on jaotatud viieks avaleheks. Avalehtede kuvatõmmised on nähtavad töö lisades 4 ja 5.

Arete ülevaade, mis on mõeldud kiire ülevaate saamiseks, veendumaks, et kõik töötab:

- Viimase 10000 töö ajaline jaotus.
- Viimase 10000 töö jaotus kursuste raames.
- Viimase 10000 töö jaotus ülesannete raames.
- Testri poolt kasutatud CPU.
- Testri poolt kasutatud kettaruum.

Tööde vaheleht, mis on eelkõige arendajale mõeldud:

- Uue alamtestri tõmmise allalaadimine/uuendamine.
- Git repositooriumi allalaadimine/uuendamine.
- Manuaalne testimine erinevate konfiguratsioonidega.
- Viimase 10000 töö detailne ülevaade.

Tudengite, ülesannete ning kursuste vahelehed, kus on kogu testimise jooksul tekkinud informatsioon agregeeritud kujul, et õpetajad ja administraatorid saaksid ülevaate kursustel toimuvast.

Leht ise on saadaval https://cs.ttu.ee/services/arete/.

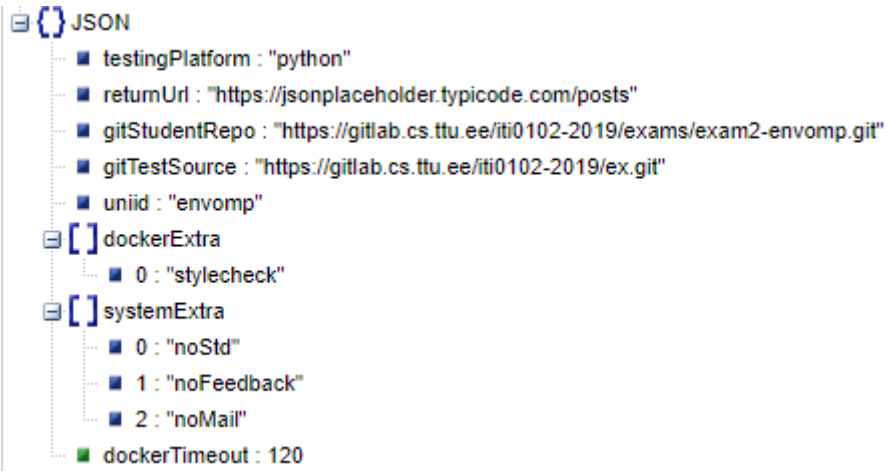
[19] "Vue.js." https://vuejs.org/. Accessed 1 Jan. 2021.

4.3.2 Java

Java on platvormist sõltumatu objektorienteeritud programmeerimiskeel. Java programmide käivitamiseks on vajalik JRE[20]. Lähtekood kirjutatakse. *java*laiendiga failidesse, kompileeritakse baitkoodi, mis asetsevad. *class*laiendiga failides, mida oskab käivitada iga JVM[21]. Arete kasutab Java versiooni 11, millele on tugi aastani 2023. [7]

4.3.3 Arete-java

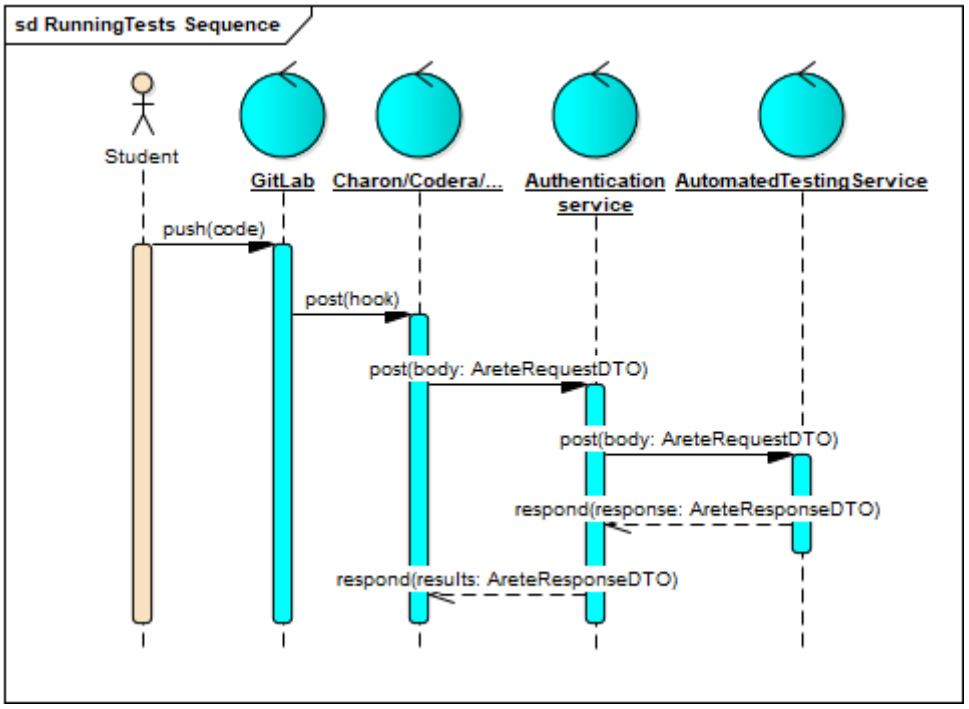
Arete-java on tarkvara teek Javas. Arete klient (*Arete Client*) suhtleb Aretega programmiliselt läbi REST[22] kihi, mis teeb Arete teenuse implementeerimise enda rakendusega oluliselt lihtsamaks. Näidis testimisele saadetavast päringust on nähtav joonisel 2.



Joonis 2: Päring Arete pihta.

[20] "Java SE Runtime Environment 8 - Downloads - Oracle." https://www.oracle.com/java/technologies/javase-jre8-downloads.html. Accessed 28 Feb. 2021.
[21] "Java virtual machine - Wikipedia." https://en.wikipedia.org/wiki/Java_virtual_machine. Accessed 28 Feb. 2021.
[22] "Representational state transfer - Wikipedia." https://en.wikipedia.org/wiki/Representational_state_transfer. Accessed 28 Feb. 2021.

Joonisel 2 oleva päringu peale läheb töö Aretesse, kus täiendatakse sisendit ja lisatakse puuduv info. Seejärel läheb töö järjekorda, kus selle jooksutamise esimene samm on testide ja testitava koodi valitud versiooni hankimine. Antud põhivoogusid illustreerib järgmine jadadiagramm joonisel 3.



Joonis 3: Arete põhivoog.

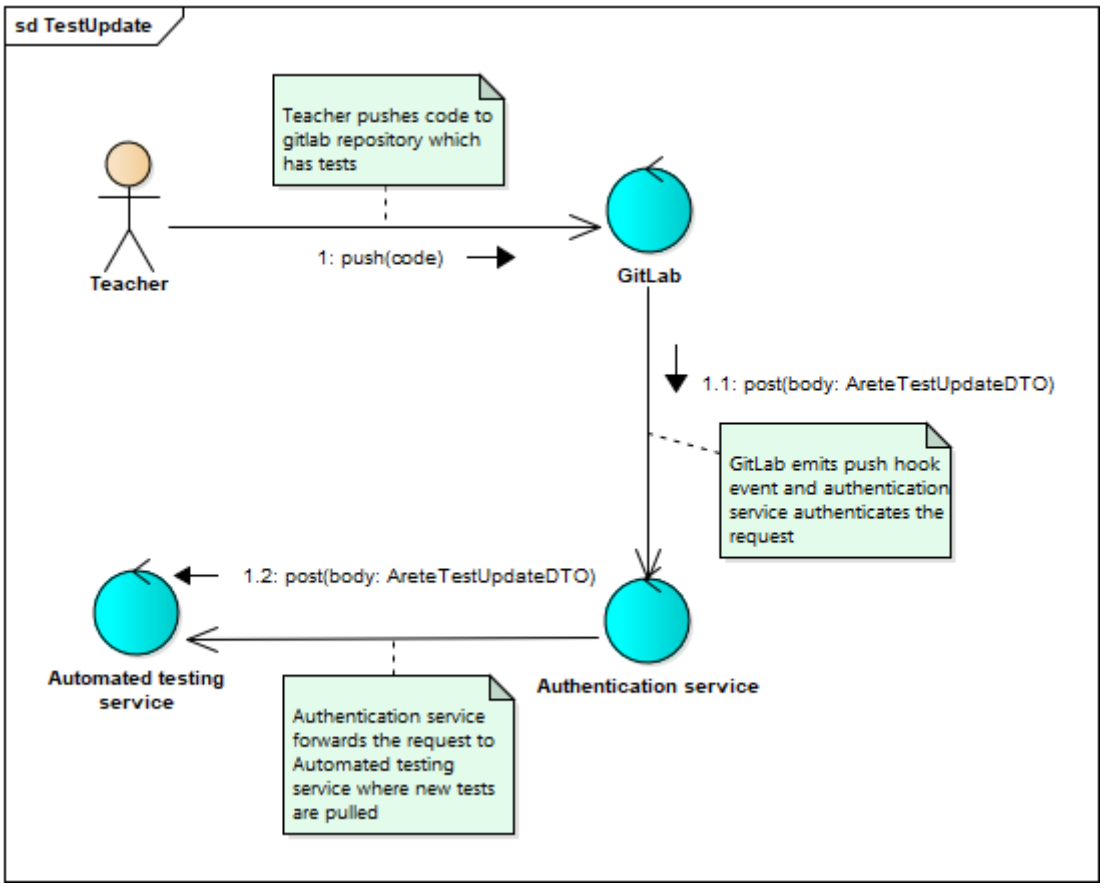
Arete poolt on toetatud sünkroonne ja asünkroonne testimine, mida kasutavad Codera[23] ja Moodle[24]. Asünkroonne testimine tähendab endast seda, et testi tulemused saadetakse Moodle poolt määratud lõpp-punktile. Sünkroonne testimine aga tähendab endast seda, et testi tulemused saadakse POST päringu tulemuseks samale päringule, mida Codera kasutab. Async lahendus on optimaalsem, kuna sellisel juhul ei hoita masina ressursse päringu taga kinni, mis võib võtta kümneid minuteid kui sedasi seadistatud läbi konfiguratsiooniparameetrite.

[23] "Codera." https://codera.cs.ttu.ee/.
[24] "Programmeerimisülesannete ... - Digikogu." https://digikogu.taltech.ee/et/Item/b628d504-57e3-4d90-9c60-4bcd6bea3d61.

4.3.4 Jgit

JGit[25] on tarkvara teek Javas. JGit võimaldab hallata Git[26] salvesid programmiliselt. Edukaks testimiseks on vajalikud testid ja testitav kood. Teste ja testitavat koodi saab hoida GitLab'is ning JGit abil ka valitud versiooni Arete'sse tõmmata.

Joonisel 4 on näha kuidas uuemaid teste alla laadida läbi lõpp-punkti.



Joonis 4: Testide uuendamine.

Git versioonihaldur põhineb hajusmudelil, seepärast on igal kasutajal Git projektist oma lokaalne koopia. Sellest tulenevalt on toimingud (näiteks kasutaja koodi või testide uuendamine) kiired. Alamtestri tõmmiste uuendamine käib analoogselt, kuid Git asemel kasutatakse docker repository't[27].

[25] "JGit | The Eclipse Foundation." https://www.eclipse.org/jgit/.
[26] "Git." https://git-scm.com/. Accessed 1 Jan. 2021.
[27] "Docker Hub." https://hub.docker.com/. Accessed 1 Jan. 2021.

4.3.5 Docker

Docker[28] on tehnoloogia, mis võimaldab rakenduste käivitamist konteinerina isoleeritud keskonnas. Konteiner pole sama, mis virtuaalmasin, sest konteiner sisaldab ainult operatsioonisüsteemi tuuma, kuhu Dockeri tarkvara on paigaldatud. Konteineri sisse pannakse lisaks rakendusele ka kõik vajalikud teegid ja failid, mida rakendus töö jaoks vajab. [8]

Dockeri üheks suurimaks eeliseks on masinast sõltumatu ühene ja kiire evituse (*deploy*) protsess, võimaldades sama rakendust jooksutada erinevates masinates. Eeliseks on veel turvalisus, kuna konteiner on algselt täielikult eraldatud serverist (*host*) ja teistest konteineritest. [9] Aretes on antud ligipääs teatud failidele serveri masinas (tudengi koodile ning testidele), aga turvalisuse kaalutlustel on need koopiad tegelikest failidest, millest vabanetakse töö lõppedes.

Dockeril on veel eeliseid: [10]

- Konteinerite skaleeruvus (*scaling*) ja nende lihtne haldus.
- Nüüdse seisuga on Docker serveri operatsioonisüsteemist sõltumatu.
- Virtuaalmasinaga võrreldes on Docker konteineritel parem ressursside kasutus.
- Arendajal pole tehnilise valiku tegemisel Arete poolseid piiranguid.
- Docker on laialt kasutatud ning arendajatel on eelnev kogemus olemas.

[28] "Docker." https://www.docker.com/.

4.3.6 Docker-java

Docker-java[29] on tarkvara teek Javas. Docker-java võimaldab suhelda *Docker Client*'iga läbi API ja selle abil jooksutada koodi isoleeritult konteinerites programmaatiliselt, kus tudengi koodi testimiseks kasutatakse valitud testri tõmmist. Joonisel 5 on näha tudengi koodi testimise jaoks loodud konteineri struktuuri.

```
container = dockerClient.createContainerCmd(imageId)
    .withName(containerName)
    .withEnv(uniidEnv, hashEnv, systemExtraEnv, dockerExtraEnv, dockerTimeoutEnv,
        dockerContentRootEnv, dockerTestRootEnv, commitMessageEnv)
    .withVolumes(volumeStudent, volumeTester, volumeOutput)
    .withAttachStdout(true)
    .withAttachStderr(true)
    .withHostConfig(newHostConfig()
        .withBinds(
            new Bind(new java.io.File(output).getAbsolutePath(), volumeOutput, rw),
            new Bind(new java.io.File(studentHost).getAbsolutePath(), volumeStudent, rw),
            new Bind(new java.io.File(testerHost).getAbsolutePath(), volumeTester, ro))
        .withCpuCount((isPriorityJob(submission) ? HIGH_PRIORITY_CPUS : LOW_PRIORITY_CPUS))
        .withMemory((isPriorityJob(submission) ? HIGH_PRIORITY_MEMORY : LOW_PRIORITY_MEMORY))
        .withMemorySwap((isPriorityJob(submission) ? HIGH_PRIORITY_MEMORY : LOW_PRIORITY_MEMORY))
        .withAutoRemove(true)
        .withPidsLimit(8192L)
        .withRestartPolicy(RestartPolicy.noRestart())
    ).exec();
```

Joonis 5. Tudengi koodi testimise jaoks loodud konteiner.

Antud joonisel on näha, et konteiner luuakse kindla testimise tõmmise peale ning kaasa antakse testimiseks vajalikud parameetrid ning failid. Lisaks on ära määratud konteineri kasutatavatele ressurssidele ülimad piirangud, mida konteiner kasutada võib.

[29] "Docker Java - GitHub." https://github.com/docker-java/docker-java. Accessed 8 Mar 2021.

Dockeri avalikus hoidlas[30] on järgmised olulisemad tõmmised:

- python-tester - alamtester, mille peal saab jooksutada Pythoni keeles kirjutatud koodi.
- java-tester - alamtester Java jaoks (lisaks on ka veel java11-tester, java13-tester ning java15-tester).
- prolog-tester - alamtester Prologi jaoks.
- fsharp-tester - alamtester Fsharp-i jaoks.
- uva-tester - alamtester, mis teeb päringuid Online Judge keskkonna pihta ning võtab sealt tudengi tulemused.
- hackerrank-tester - alamtester, mis teeb päringuid HackerRank keskkonna pihta.
- kattis-tester - alamtester, mis teeb päringuid Kattis keskkonna pihta.
- authentication_service - arhitektuuri peatükis kirjeldatud autentimisteenus.
- automated_testing_service - arhitektuuri peatükis kirjeldatud koormuse tasakaalustamise teenus.
- monitoring_service - Vue peatükis kirjeldatud teenus, kus on realiseeritud graafiline liides Arete lõpp-punktidega suhtlemine.
- arete_integration_tests - integratsioonitestid, mida saab jooksutada, et tagada kõikide süsteemiosade funktsionaalsed nõuded. Lisaks lihtsustab alamtestrite arendamist.

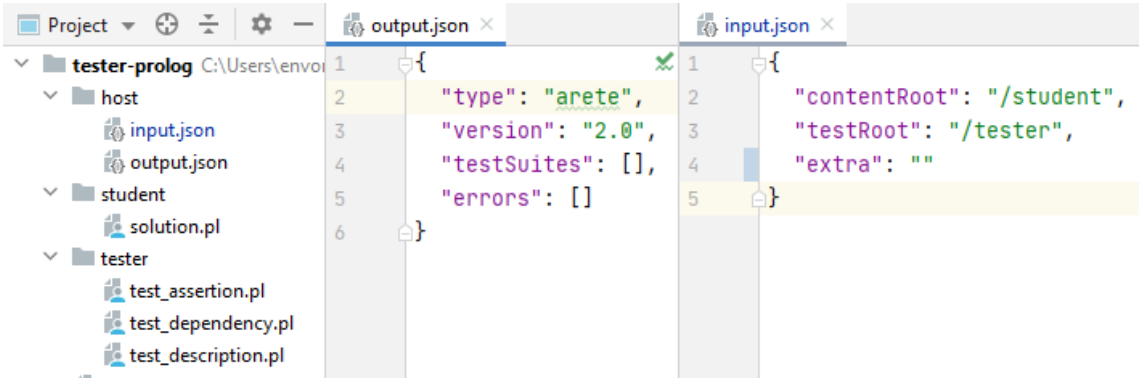
Testimine konteineris tagab:

- Konfidentsiaalsuse, kus testitav kood pääseb ligi ainult lubatud resurssidele.
- Käideldavuse, et testimiseks oleksid vajalikud ressursid kättesaadavad.
- Terviklikkuse, et testimine viidaks läbi alati samasuguses keskkonnas, kus testi tulemused ei sõltu kõrvalistest mõjutavatest teguritest.

[30] "Docker: Empowering App Development for Developers." https://hub.docker.com/u/automatedtestingservice.

4.3.7 Alamtester

Alamtester käivitatakse Joonisel 5 olevate piirangutega ning sisenditega, mille struktuur on nähtav joonisel 6:



Joonis 6. Alamtestri struktuur.

Joonisel on märkimist väärt järgmised kaustad:

- */student* kaust, kus asetsevad kõik tudengi failid, mis tudengil Git salves antud ülesande alamkaustas olid.
- */tester* kaust, kus asetsevad kõik testimise failid, mis testide Git repositooriumis antud ülesande alamkaustas olid.
- */host* kaust, mis hoiustab endas järgmisi faile:
    - *input.json* fail, kus väärtused on täielikult konfigureeritavad, mis võimaldavad alamtestris kõike, mida vaja. Kasvõi liidestada teise süsteemi pihta. Hetkel on nähtavad vaikimisi väärtused, mis viitavad tudengi ja testi koodi kaustadele.
    - *output.json* fail, kuhu alamtester paneb koodi testimisel tekkivad tulemused. Täielik klassidiagramm on saadav antud lõputöö lisades.

Kõik muud failid, mis */host* kaustas on, saadetakse tudengile mailile.

4.3.8 Arete-response

Arete-response on POJO — formaat, mida kõik Arete kliendid toetavad. Antud töö tegi keerulisemaks eksisteeriva formaadi puudumine. Alamtestrid (python_tester, java_tester, prolog_tester) tagastasid erineval kujul tulemusi. Töö käigus pandi paika struktuur, mida kõik alamtestrid tagastama peavad ning migreeriti testrid üle uuele struktuurile. Täielik klassidiagramm on saadaval töö lisades ning näide on joonisel 7:


```
{
  "testingPlatform": "python",
  "gitStudentRepo": "git@gitlab.cs.ttu.ee:envomp/iti0201-2021.git",
  "gitTestRepo": "git@gitlab.cs.ttu.ee:iti0201-2021/ex.git",
  "hash": "1717d3748cba0d3de0fbc1a517c84707ab5cc9ea",
  "uniid": "envomp",
  "root": "iti0201-2021/ex",
  "slug": "EX12",
  "errors": [{
    "columnNo": 3,
      "fileName": "file.py",
      "hint": null,
      "kind": "Style Error",
      "lineNo": 5,
      "message": "Unused import"
  }],
  "testSuites": [{
      "unitTests": [
        {
          "status": "PASSED",
          "weight": 1,
          "timeElapsed": 10,
          "name": "test_dummy",
          "exceptionClass": null,
          "exceptionMessage": null
        }],
      "passedCount": 1,
      "grade": 100
    }
  ],
  "dockerExtra": ["stylecheck"],
  "systemExtra": ["noFiles"],
  "dockerTimeout": 120,
  "timestamp": 1618780300578,
  "priority": 5,
  "failed": false,
  "output": "HTML output for student",
  "consoleOutput": "Docker produced console output"
}
```

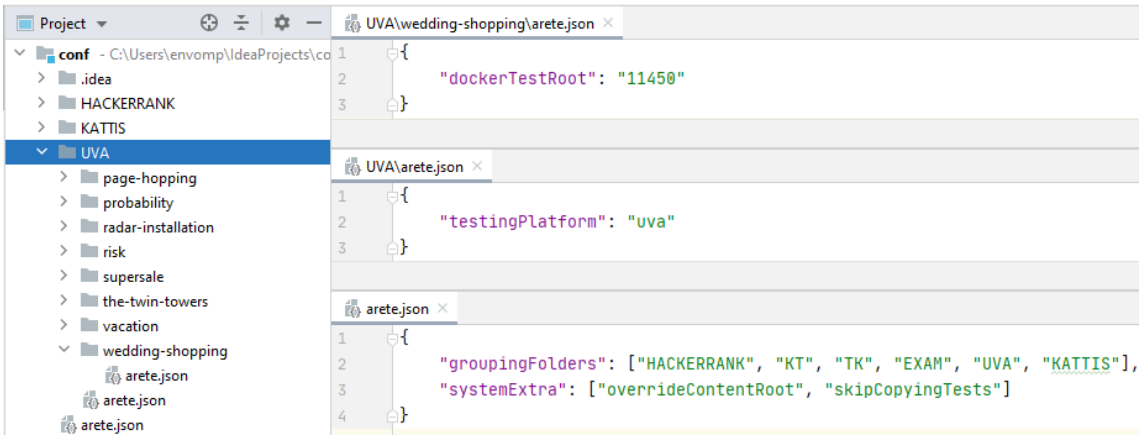

Joonis 7. Arete-response.

Arete-response sees on veel ka failid, mida kasutati testimisel aga jäeti välja “noFiles” systemExtra parameetri tõttu. Arete response kujutab endast detailset testi tulemust, mille järgi tudengitele tagasisidet anda — milline oli iga üksustesti tulemus (FAILED, SKIPPED, PASSED) ning mis vead koodi jooksutamisel tekkisid (stiili viga, kompileerimisel tekkinud viga, vms.).

Arete-response sees on lisaks ka veel lisainformatsioon, et viia testimine kokku kindla tudengi, ülesande ning kursusega.

4.3.9 Arete.json

arete.json on konfiguratsioonifail, mille saab asetada Git repositooriumisse, et edastada testrile ülesande spetsiifilist konfiguratsiooni, mille illustratiivne näide on joonisel 8:



Joonis 8. arete.json konfiguratsioon.

Duplikaatide eemaldamiseks pannakse lõplik konfiguratsioon kokku mestides alamkaust gruppeerimiskausta ning ülesande kaustaga. Grupeerimiskaustasid saab defineerida alamkausta *arete.json* failis. Täielik juhend on Arete kasutusjuhendis.[31]

[31] "Arete - automated testing service — IT doc 1.0 ... - TalTech." https://ained.pages.taltech.ee/it-doc/arete/index.html#usage. Accessed 8 May. 2021.

4.3.10 Maven

Apache Maven[32] on POM[33] kontseptsioonil põhinev projektihalduse tööriist. Maveni peamine ülesanne on anda arendajale võimalus hallata Java projekti kõiki arenduskäigu olekuid ja muuta arendusprotsess võimalikult kiireks ja lihtsaks. [12]

pom.xml on fail, mis sisaldab endas informatsiooni ja sõltuvusi, mida arendaja saab ise lisada ja muuta. Nähtav joonisel 9.

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.2.1.RELEASE</version>
</parent>

<groupId>ee.taltech</groupId>
<artifactId>arete</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>Arete</name>
<description>Automated testing service</description>

<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter</artifactId>
    </dependency>
</dependencies>
```

Joonis 9: Spring booti sõltuvuse lisamine Maven-i.

Põhjus, miks valituks sai Maven üle muude projektihaldustööriistade (näiteks Gradle[34]) oli see, et Maven on algajale lihtsam ja rohkem kasutatud, mistõttu antud rakendust oskaksid ka teised arendajad tulevikus edasi arendada.

[32] "Maven – Welcome to Apache Maven." https://maven.apache.org/. Accessed 6 Mar. 2021.
[33] "Maven – POM Reference." https://maven.apache.org/pom.html. Accessed 28 Feb. 2021.
[34] "Gradle." 23 Feb. 2021, https://gradle.org/. Accessed 6 Mar. 2021.

4.3.11 Spring

Spring on tarkvara rakenduse raamistik, mille põhifunktsioone saab kasutada ükskõik milline Java-rakendus. Spring Boot[35] on osa Springi raamistikust, millega saab Springi rakenduse luua võimalikult lihtsalt ja kiirelt. Spring Boot valib teegid ja seadistused ise, kuid neid on võimalik muuta. [13] Springi raamistikul loodud rakendused koosnevad Spring Bean[36] objektidest, mis on Springi poolt seadistatud ja kokku ühendatud. [14]

Põhjus, miks muude raamistike (nagu näiteks DropWizard[37]) seast osutus valituks Spring Boot oli autori varasem kokkupuude ja kasutuskogemus sellega. Tänu sellele saavad tulevased arendajad koodi lihtsamini edasi arendada.

Lisaks on Spring Bootil annotatsioon @Async, mis võimaldab tööde järjekorras olevaid töid paralleelselt jooksutada.

Tööde järjekorras hoidmine on oluline järgmistel põhjustel:

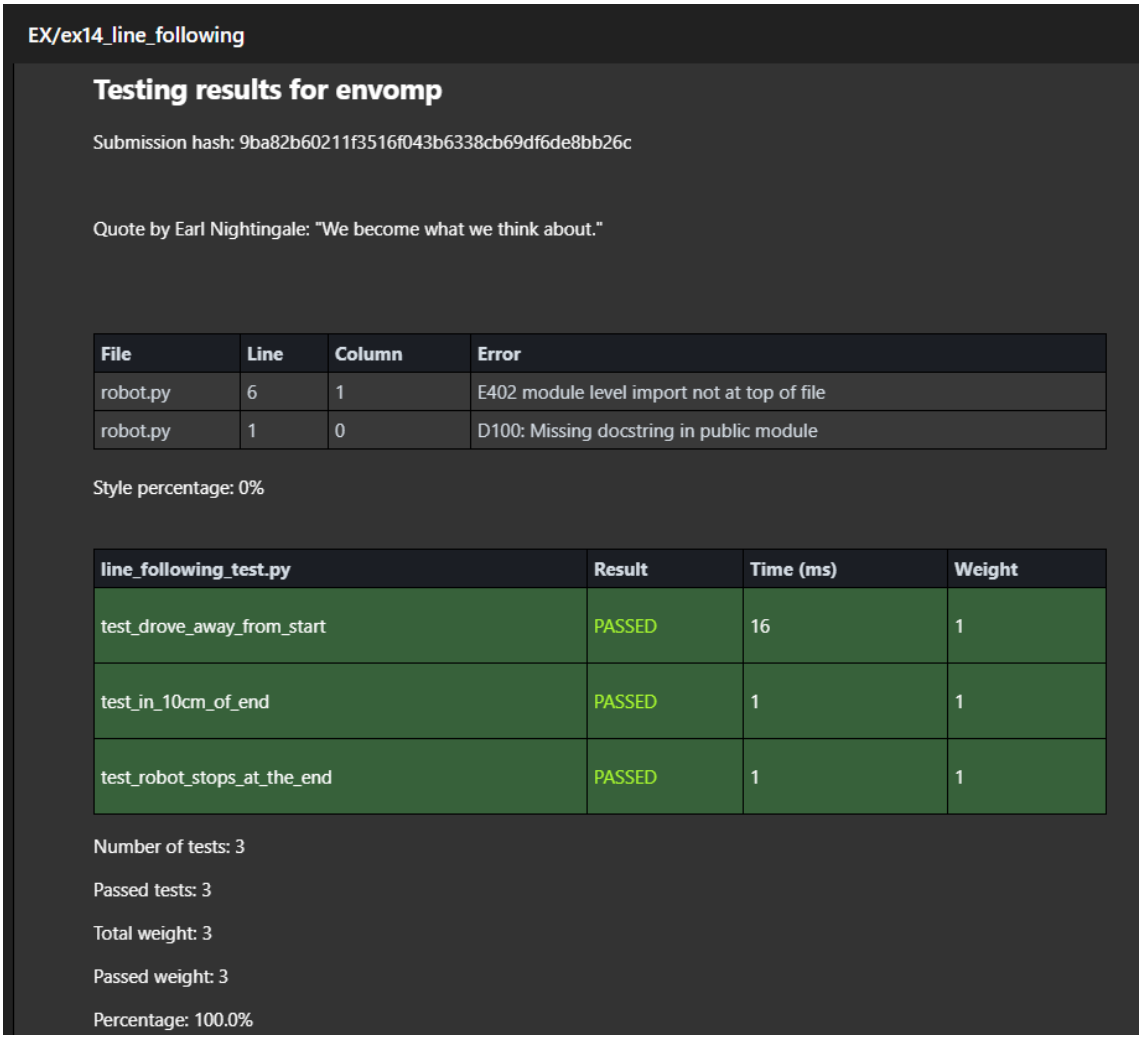
- Üks tudeng saab korraga ühte tööd jooksutada.
- Tagab teenuse töökindluse, et tööde samaaegse kuhjumise korral masinat üle ei koormata.

4.3.12 Postfix

Postfix[38] on postiülekande agent (*mail transfer agent*), mis on mõeldud mailide saatmiseks ja vastuvõtmiseks. Arete kasutab mailiteenust, et anda tudengitele tagasisidet nende esitatud koodile.

[35] "Spring Boot." https://spring.io/projects/spring-boot. Accessed 6 Mar. 2021.
[36] "What is a Spring Bean? | Baeldung." 7 Dec. 2019, https://www.baeldung.com/spring-bean.
[37] "Dropwizard." https://www.dropwizard.io/. Accessed 6 Mar. 2021.
[38] "The Postfix Home Page." http://www.postfix.org/.

Joonisel 10 on näide ühest tagastatavast mailist, millel nähtav HTML kujunduseks kasutatud informatsioon tuli alamtesterist. Tagastatav mail võib olla ka lihtsalt sõne (*String*), kui testimine lõppes veateatega.



EX/ex14_line_following

Testing results for envomp

Submission hash: 9ba82b60211f3516f043b6338cb69df6de8bb26c

Quote by Earl Nightingale: "We become what we think about."

File	Line	Column	Error
robot.py	6	1	E402 module level import not at top of file
robot.py	1	0	D100: Missing docstring in public module

Style percentage: 0%

line_following_test.py	Result	Time (ms)	Weight
test_drove_away_from_start	PASSED	16	1
test_in_10cm_of_end	PASSED	1	1
test_robot_stops_at_the_end	PASSED	1	1

Number of tests: 3

Passed tests: 3

Total weight: 3

Passed weight: 3

Percentage: 100.0%

Joonis 10: Tudengile saadetav mail.

Mailis on manustatud failid, mis on mõeldud tudengile nägemiseks.

Koodis on postfix implementeeritud Spring Boot *bean*'ina, kus maile saadetakse paralleelselt. Postfixi põhilisteks seadistusfailideks on *master.cf* ja *main.cf*, kus *master.cf* seadistab deemonprotsesse ning *main.cf* sisaldab postfixi seadistusparameetreid, mis on vajalikud mailide liigutamiseks. [15]

4.3.13 Andmebaasid

Spring Boot toetab SQL andmebaase. Andmebaasiga suhtlemiseks kasutab Spring Boot JDBC[39] või ORM[40] tehnoloogiaid. JPA võimaldab kaardistada java objekte relatsioonilise andmebaasiga. [16] *Spring-boot-starter-data-jpa starter* sisaldab Spring ORM, Spring Data JPA[41] ja Hibernate[42] sõltuvusi. [14] Spring Data JPA genereerib andmebaasi päringud otse hoidla meetodist, mida illustreerib joonis 11.

```
@Repository
public interface SubmissionRepository extends JpaRepository<Submission, Long> {

    ArrayList<Submission> findByHash(@Param("hash") String hash);

}
```

Joonis 11: Spring Data JPA hoidla meetod.

Java javax.sql.DataSource liides loob ühenduse andmebaasiga. DataSource'i saab seadistada application.properties failiga, mida illustreerib joonis 12.

```
spring.datasource.url=jdbc:postgresql://db/arete
spring.datasource.username=arete
spring.datasource.password=arete
spring.datasource.platform=postgres
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
spring.datasource.driverClassName=org.postgresql.Driver
```

Joonis 12: Spring Data Source parameetrid.

Aretes kasutatakse kahte andmebaasi andmete hoiustamiseks. Toodangus kasutab Arete PostgreSQL[43] andmebaasi. Arendamiseks ja testimiseks kasutab Arete H2[44] andmebaasi, kus andmeid hoitakse arvuti muutmälus.

[39] "JDBC - Oracle." https://www.oracle.com/technetwork/java/javase/jdbc/index.html.
[40] "19. Object Relational Mapping (ORM) Data Access - Spring." https://docs.spring.io/spring/docs/4.2.x/spring-framework-reference/html/orm.html.
[41] "Spring Data JPA." https://spring.io/projects/spring-data-jpa.
[42] "Hibernate. Everything data. - Hibernate." https://hibernate.org/.
[43] "PostgreSQL." https://www.postgresql.org/.
[44] "H2 Database Engine." https://www.h2database.com/.

4.3.14 Testimisraamistikud

Spring boot pakub annotatsioone ja tööriistu rakenduse testimiseks. Arete kasutab selleks *spring-boot-starter-test* sõltuvust nii integratsiooni- kui ka üksustestide kirjutamiseks. Aretes jälgitakse ühest malli testide kirjutamisel: [17]

- Testi nimi kirjeldab testis kontrollitavat.
- Testis on eristatavad kolm osa:
    - Given, kus defineeritakse sisend.
    - When, kus kutsutakse välja testitav meetod.
    - Then, kus kontrollitakse tulemust.

Üksustestides jäljendatakse Mockito[45] raamistiku abil neid objekte, millest antud klass sõltub. Joonisel 13 on näha näidet üksustestist.

```
@Test
void populateDefaultValues() {
    Submission submission = Submission.empty();
    submission.setUniid("envomp");

    submissionService.populateDefaultValues(submission);

    assertTrue( condition: submission.getHash().length() > 0);
    assertEquals( expected: 5, submission.getPriority());
    assertTrue( condition: submission.getTimestamp() > 100000000);
    assertTrue( condition: submission.getReceivedTimestamp() > 100000000);
    assertEquals( expected: 120, submission.getDockerTimeout());
    assertEquals( expected: "envomp@ttu.ee", submission.getEmail());
}
```

Joonis 13: Üksustesti näide.

Üksustestimise kirjutamine toimub arendamise kõrvalt, sest siis on kõige lihtsam vigu avastada ja parandada.

[45] "Mockito framework site." https://site.mockito.org/.

Integratsioonitestides testitakse reaalseid objekte. Integratsioonitestide kirjutamiseks kasutati REST Assured[46] teeki, mis teeb testid loetavamaks ja oskab REST päringuid teha. Joonis 14 illustreerib integratsiooni testi REST Assured teegi kasutusega.

```
@Test
public void testNoFiles() {
    AreteRequestDTO payload = getSubmissionPythonFiles();
    payload.getSystemExtra().add("noFiles");

    AreteResponseDTO response = areteService.makeRequestSync(payload);

    assertEquals( expected: 0, response.getTestFiles().size());
    assertEquals( expected: 0, response.getFiles().size());
}
```

Joonis 14: Integratsioonitesti näide.

Testide jooksutamiseks kasutatakse JUnit 5[47] raamistiku.

[46] "REST Assured." http://rest-assured.io/.
[47] "JUnit 5." https://junit.org/junit5/.

5 Rakenduse publitseerimine

Arete jaoks eraldati TalTechis server. Selles peatükis kirjeldatakse, kuidas koodimuudatused jõuavad serverisse.

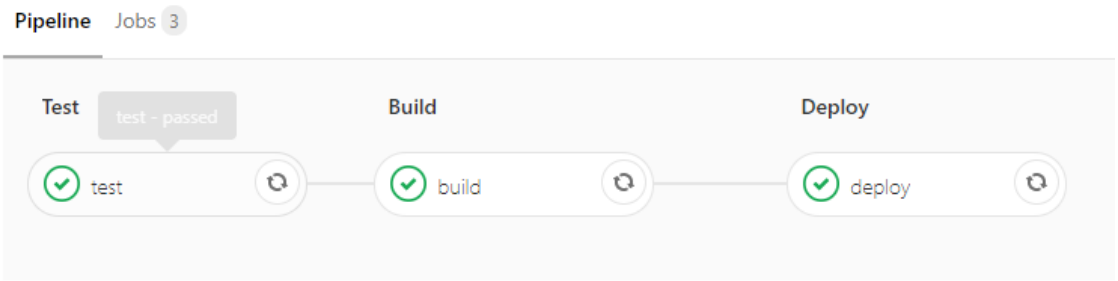
5.1 Gitlab

Projektihaldus toimub läbi TalTechi Gitlabi[48]. Gitlab on veebipõhine Git salvede haldamise tööriist.

Koodi hoidmine Gitlab'i serveris võimaldab autoriseeritud isikutel ja rakendustel koodile lihtsalt ligi pääseda ning peale muudatuste tegemist testitud toode publitseerida läbi CI.

5.2 Gitlabi pidevkooste

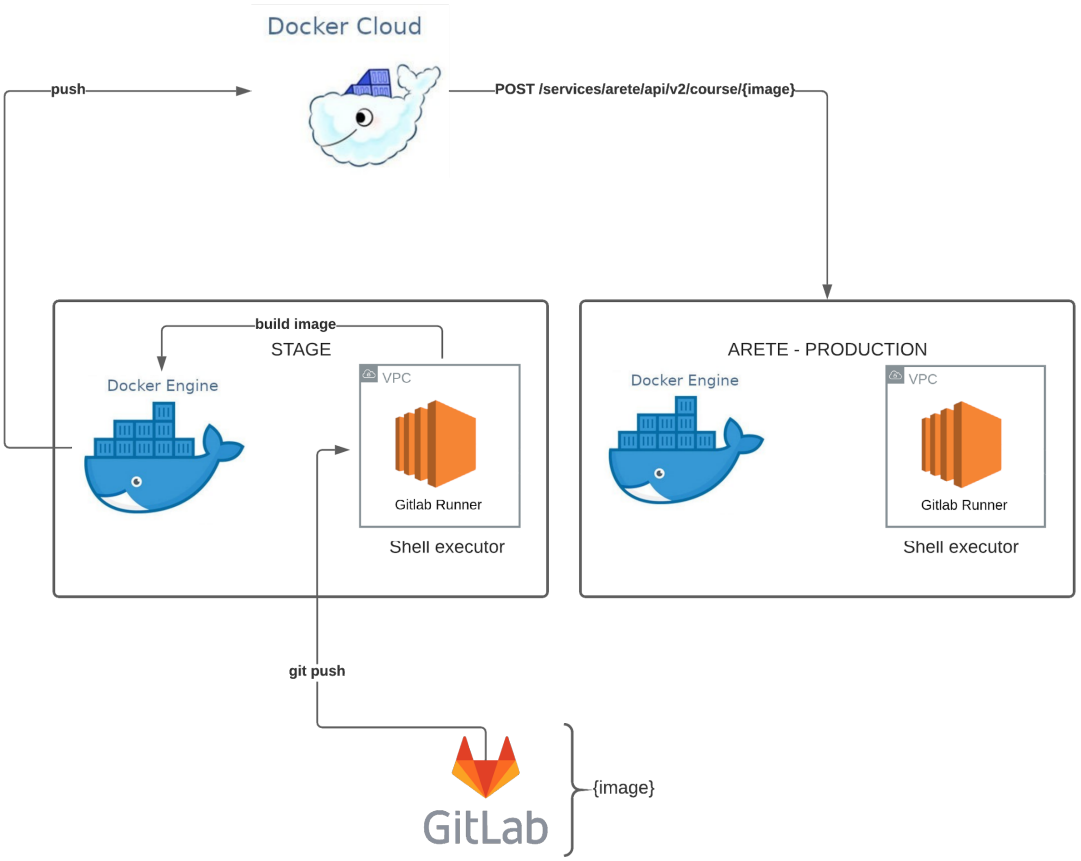
Muudatuste tegemisel jooksutatakse teste veendumaks, et uuendused ei teinud midagi katki. Konveier (*pipeline*) loetakse õnnestunuks, kui kõik tööd tagastasid koodi 0. Joonis 15 illustreerib õnnestunud töö konveierit.



Joonis 15: pipeline.

[48] "Sign in · GitLab - TTÜ." https://gitlab.cs.ttu.ee/.

Kõigile Arete alamkomponentidele on seadistatud pidevkooste, ehk CI ning alamtestrite CI on nähtav joonisel 16.



Joonis 16: Alamtestri CI.

See soodustab agiilset arendust ning hoiab Aretet ning alamtestreid arendades aega kokku evitusperioodi pealt. [18]

Gitlab CI seadistus toimub *.gitlab-ci.yml* failis. YAML formaadis konfiguratsiooni fail koosneb etappidest (*stages*), etapid koosnevad töödest (*jobs*), ning tööd omakorda käskudest.

Alamtestrite konfiguratsioonifaili illustreerib joonis 17.

```
stages:
  - stage

deploy:
  tags:
    - codetest
  stage: stage
  script:
    - docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
    - docker build -t automatedtestingservice/python-tester:latest .
    - docker push automatedtestingservice/python-tester:latest
    - |
      curl -X POST "https://cs.ttu.ee/services/arete/api/v2/course/python-tester"
        -H "accept: */*" -H "X-Docker-Token: python-tester $ARETE_TOKEN"
  only:
    - master
```

Joonis 17: CI konfiguratsioon.

Konfiguratsioonifailis on näha käsklused, mis suhtlevad *Docker Client*'iga *docker-compose* vahendusel.

5.3 Docker compose

Docker compose[49] lihtsustab konteinerite haldamist. Docker compose tarkvara taandab *Docker Client*'iga suhtluse paarile käsule:

- docker-compose build, mis ehitab uued konteinerid *Dockerfile*'de järgi.
- docker-compose down, mis paneb konteinerid kinni.
- docker-compose up, mis jooksutab uusi konteinerid.

Ka *docker-compose.yml* fail on YAML formaadis ning seal on ära määratud konteinerid ning neile vajalikud parameetrid. Joonisel 18 on näide testimise teenuses kasutatavast *docker-compose.yaml* failist.

```
version: '3.4'

services:

    testing_service:
        container_name: automated_testing_service
        image: ${DOCKER_USER}/automated_testing_service:latest
        restart: unless-stopped
        privileged: true
        working_dir: ${ARETE_HOME}
        ports:
            - "8098:8098"
        network_mode: "host" #Communication to authentication service
        environment:
            BACKEND_URL: ${BACKEND_URL}
            ARETE_HOME: ${ARETE_HOME}
            SHARED_SECRET: ${SHARED_SECRET}
        volumes:
            - /var/run/docker.sock:/var/run/docker.sock
            - ${ARETE_HOME}/input_and_output:${ARETE_HOME}/input_and_output
            - ${ARETE_HOME}/students:${ARETE_HOME}/students
            - ${ARETE_HOME}/tests:${ARETE_HOME}/tests
            - ${ARETE_HOME}/ssh:/root/.ssh
```

Joonis 18: docker-compose.yaml.

[49] "Overview of Docker Compose | Docker ...." https://docs.docker.com/compose/.

Arete kasutab Dockerit rakenduste serveerimiseks. Toodangus jooksutatakse nelja konteinerit:

- Adminer[50], mille kaudu saab reaalajas andmebaase hallata.
- Authentication service, rakendus, mis autentib kasutajaid.
- Automated testing service, rakendus ise, mis testib tudengite koode.
- ProstgeSQL[51] andmebaas, kus hoitakse tulemused ja testimiseks vajalik.

Dockeri tõmmised (*image)* luuakse projektis olevate *Dockerfile*'de järgi, mille näide on illustreeritud joonisel 19.

```
FROM openjdk:11

MAINTAINER enrico.vompa@gmail.com

ADD . /integration-tests
WORKDIR /integration-tests

RUN apt-get update && apt-get install dos2unix
RUN dos2unix mvnw
RUN chmod +x mvnw

RUN ./mvnw install -DskipTests

ENTRYPOINT [ "sh", "-c", "./mvnw test" ]
```

Joonis 19: Dockerfile.

Peale docker-compose up tegemist alustavad konteinerid töötamist ja testimise teenus on kättesaadav.

[50] "adminer - Docker Hub." https://hub.docker.com/_/adminer/.
[51] "postgres - Docker Hub." https://hub.docker.com/_/postgres.

6 Töö tulemuste analüüs

Antud töö kirjutamise ajal on Aretet kasutatud pea 2 aastat ning 9. mai 2021 seisuga on TalTechi kursuste raames järgmine seis:

Tabel 1: Tööde arv kursuste raames.

Kursuse ainekood	Jooksutatud tööde arv	Jooksutatud üksustestide arv
ITI0102-2020	146 295	3 136 270
ITI0201-2021	27 313	61 158
ITI0200-2020	25 535	666 114
ITI0202-2021	18 092	499 483
ITI0204-2020	14 645	148 845
ITT8060-2020	9 888	507 327
YFX0505-2021	2 389	13 112
ITI0214-2021	1 008	3 789

Arete on ajale ja koormusele hästi vastu pidanud ning leiab juba kasutust ka väljaspool IT teaduskonda.

Alamtestrite edasiarenduse ning uute alamtestrite loomiseks on olemas DEVELOPER kasutaja roll Arete kasutajaliideses, kust on vajalikud lõpp-punktid arendajale kättesaadavad. Kasutajaid saab registreerida ainult administraator. On olemas ka integratsioonitestide repositoorium kuhu saab enda testid lisada, kontrollimaks kas tester annab oodatuid tulemusi: https://gitlab.cs.ttu.ee/testing/arete-integration-tests.

7 Kokkuvõte

Töö esmaseks eesmärgiks oli luua testimisplatvorm, millel oleks eelmise testri funktsionaalsus, ning mis rahuldaks uusi kooli poolt kehtestatud vajadusi. Teiseks eesmärgiks oli teha teenus klientidele kättesaadavamaks.

Esmane eesmärk saavutati läbi mitme sammu. Esimeseks sammuks oli analüüs, mille käigus uuriti eelmist testrit ja mujal kasutatavaid testreid ning toodi välja mittefunktsionaalsed ja funktsionaalsed nõuded uuele testrile. Järgmise etapina valiti tehnoloogiad ja platvormid, milleks osutusid Docker, Vue raamistik, Spring Boot raamistik ning PostgreSQL andmebaasisüsteem.

Teine eesmärk saavutati platvormi kättesaadavaks tegemisel TalTechi serveris *docker-compose* abil. Lisaks sai Gitlab CI tarkvaraga välja töötatud protsess, mis lihtsustab muudatuste lisamist.

Kokkuvõtvalt, saavutatud tulemus täidab etteantud nõudeid ja testri kasutuse jooksul pole suuremaid probleeme tekkinud.

Kasutatud kirjandus

1. Kaastöölised WP. Funktsionaalsed nõuded – Vikipeedia. In: Wikimedia Foundation, Inc. [Internet]. 14 Mar 2010 [cited 21 Jan 2020]. Available: https://et.wikipedia.org/wiki/Funktsionaalsed_n%C3%B5uded

2. Kaastöölised WP. Mittefunktsionaalsed nõuded – Vikipeedia. In: Wikimedia Foundation, Inc. [Internet]. 18 Nov 2016 [cited 21 Jan 2020]. Available: https://et.wikipedia.org/wiki/Mittefunktsionaalsed_n%C3%B5uded

3. Mallawaarachchi V. 10 Common Software Architectural Patterns in a nutshell. In: Medium [Internet]. Towards Data Science; 4 Sep 2017 [cited 21 Jan 2020]. Available: https://towardsdatascience.com/10-common-software-architectural-patterns-in-a-nutshell-a0b47a1e9013

4. What is a “server application”? In: Stack Overflow [Internet]. [cited 21 Jan 2020]. Available: https://stackoverflow.com/questions/20579840/what-is-a-server-application

5. Custom Methods | Cloud APIs | Google Cloud. In: Google Cloud [Internet]. [cited 18 Jan 2020]. Available: https://cloud.google.com/apis/design/custom_methods

6. Doglio F. Architecting a REST API. Pro REST API Development with Node.js. 2015. pp. 65–78. doi:10.1007/978-1-4842-0917-2_4

7. Ashok KP. An Introduction to Java Programming Language. LAP Lambert Academic Publishing; 2014.

8. What is a Container? | Docker. In: Docker [Internet]. [cited 21 Jan 2020]. Available: https://www.docker.com/resources/what-container

9. DataFlair Team. Advantages and Disadvantages of Docker - Learn Docker - DataFlair. In: DataFlair [Internet]. 22 Nov 2018 [cited 21 Jan 2020]. Available: https://data-flair.training/blogs/advantages-and-disadvantages-of-docker/

10. Sergey C, Sviatoslav A. Advantages of Using Docker for Microservices. In: RubyGarage [Internet]. 9 Jan 2020 [cited 21 Jan 2020]. Available: https://rubygarage.org/blog/advantages-of-using-docker-for-microservices

11. Docker Consulting Services and Solutions Company: TX, USA. In: Veritis Group Inc [Internet]. [cited 21 Jan 2020]. Available: https://www.veritis.com/services/docker/

12. Varanasi B. Maven Dependency Management. Introducing Maven. 2019. pp. 23–35. doi:10.1007/978-1-4842-5410-3_3

13. Gutierrez F. Web Development with Spring Boot. Pro Spring Boot. 2016. pp. 149–175. doi:10.1007/978-1-4842-1431-2_8

14. Website. [cited 18 Jan 2020]. Available: https://www.cs.tlu.ee/teemaderegister/get_file.php?id=671&name=Karl%20Oha_Spring%20Boot%20raamistik.pdf

15. Postfix'i arhitektuuri kirjeldus ja kasutamise keerulisemad võimalused – Kuutõrvaja. [cited 18 Jan 2020]. Available: https://kuutorvaja.eenet.ee/wiki/Postfix%27i_arhitektuuri_kirjeldus_ja_kasutamise_keerulisemad_v%C3%B5imalused

16. [No title]. [cited 1 Jan 2021]. Available: https://www.cs.tlu.ee/teemaderegister/get_file.php?id=671&name=Karl%20Oha_Spring%20Boot%20raamistik.pdf15.

17. Gutierrez F. Testing with Spring Boot. Pro Spring Boot. 2016. pp. 107–120. doi:10.1007/978-1-4842-1431-2_6

18. Wilson J, Watkins J. The Power of Continuous Integration Builds and Agile Development. Agile Testing. pp. 93–102. doi:10.1017/cbo9780511596797.017

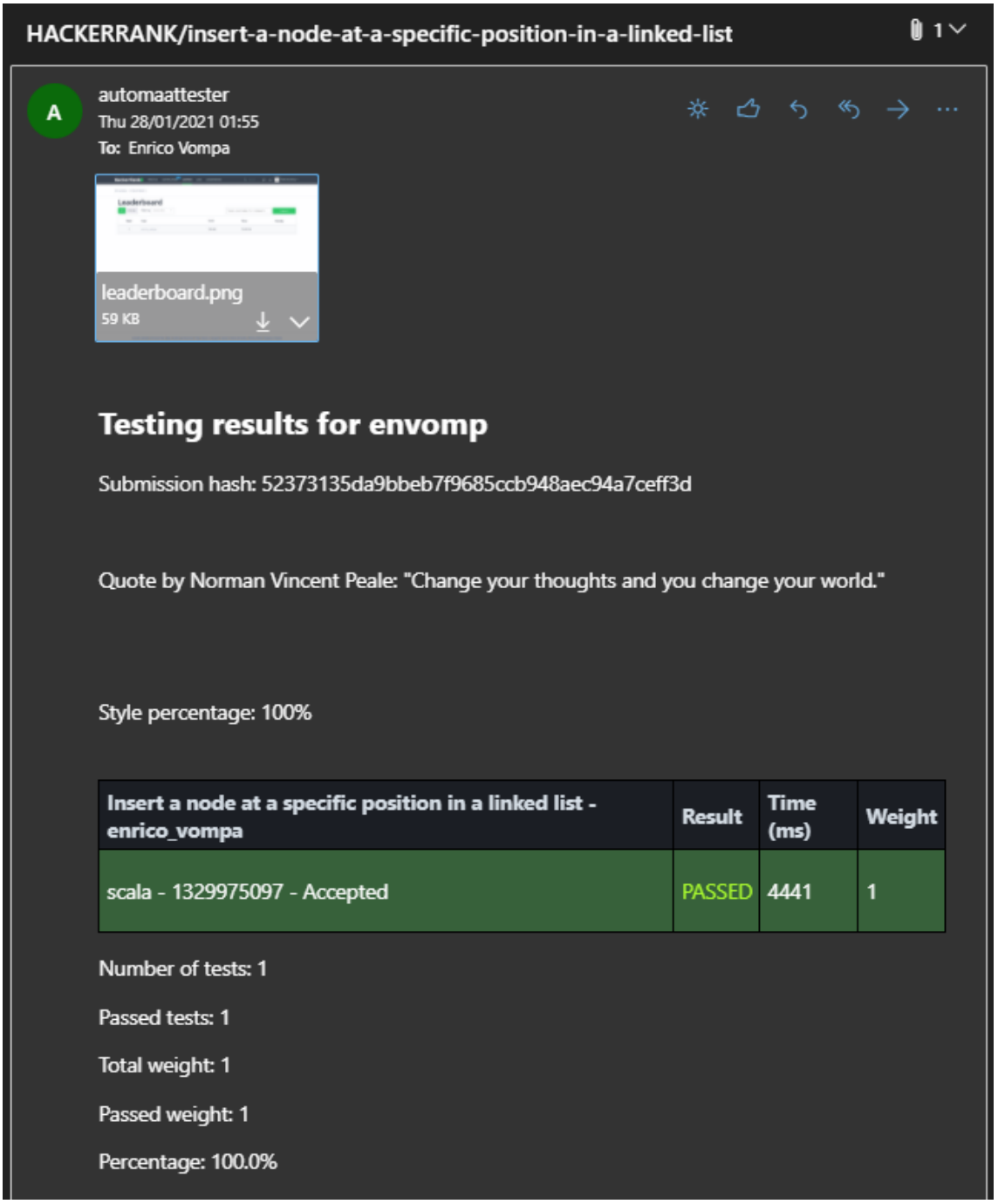
Lisa 1. Lihtlitsents töö üldsusele kättesaadavaks tegemiseks ja reprodutseerimiseks

Mina, Enrico Vompa

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose ”Õppetöö läbiviimiseks kasutatav TalTechi õppesüsteemi automaattestimissüsteem tudengi koodi esituste jaoks”, mille juhendaja on Ago Luberg
    1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonilise avaldamise eesmärgil, sh TTÜ raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
    1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas TTÜ raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta kolmandate isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

17.05.2021

Lisa 2. HackerRank'i integratsioon ja manus mailis



HACKERRANK/insert-a-node-at-a-specific-position-in-a-linked-list

1

A

automaattester
Thu 28/01/2021 01:55
To: Enrico Vompa

leaderboard.png
59 KB

Testing results for envomp

Submission hash: 52373135da9bbeb7f9685ccb948aec94a7ceff3d

Quote by Norman Vincent Peale: "Change your thoughts and you change your world."

Style percentage: 100%

Insert a node at a specific position in a linked list - enrico_vompa	Result	Time (ms)	Weight
scala - 1329975097 - Accepted	PASSED	4441	1

Number of tests: 1

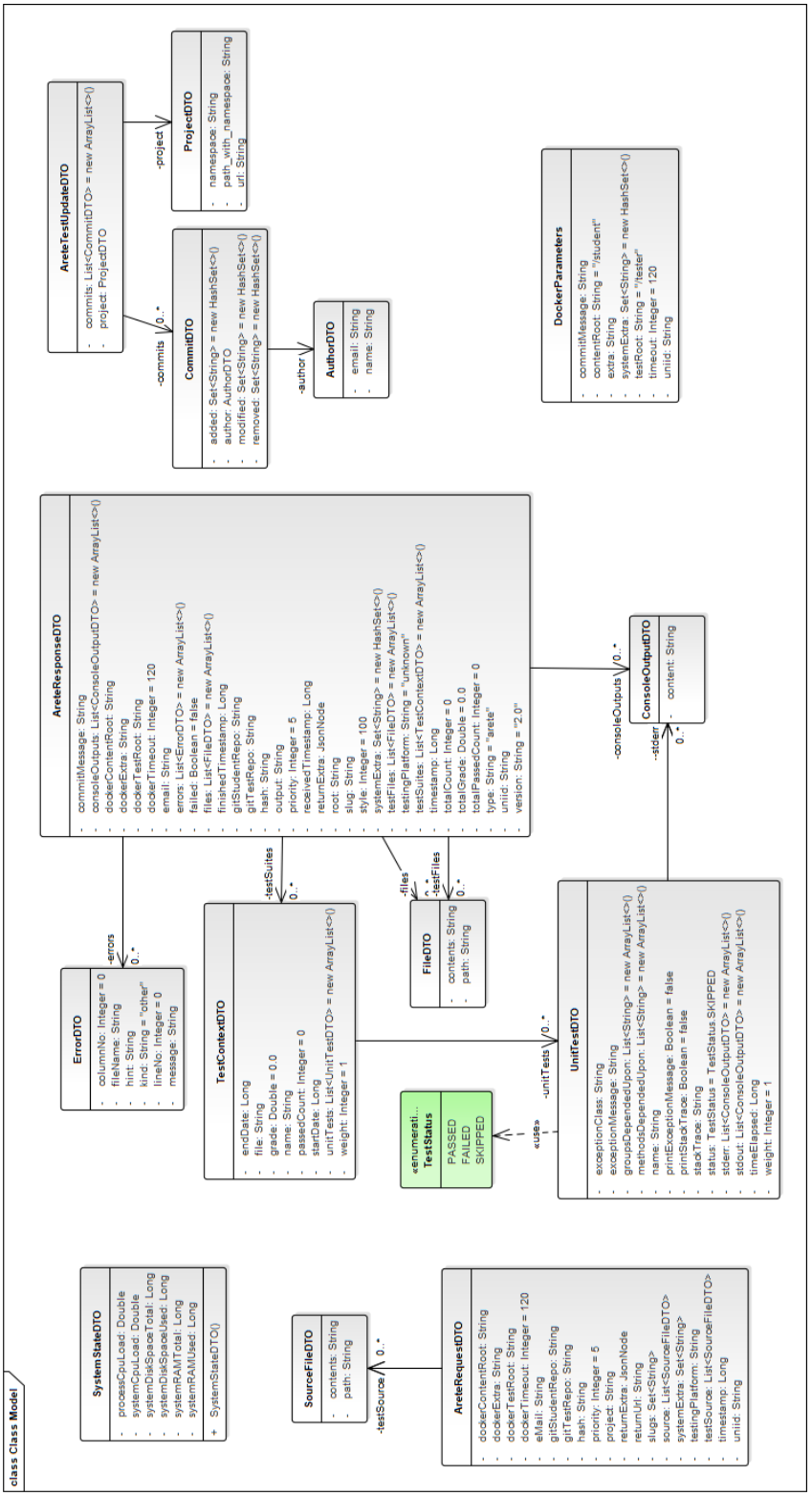
Passed tests: 1

Total weight: 1

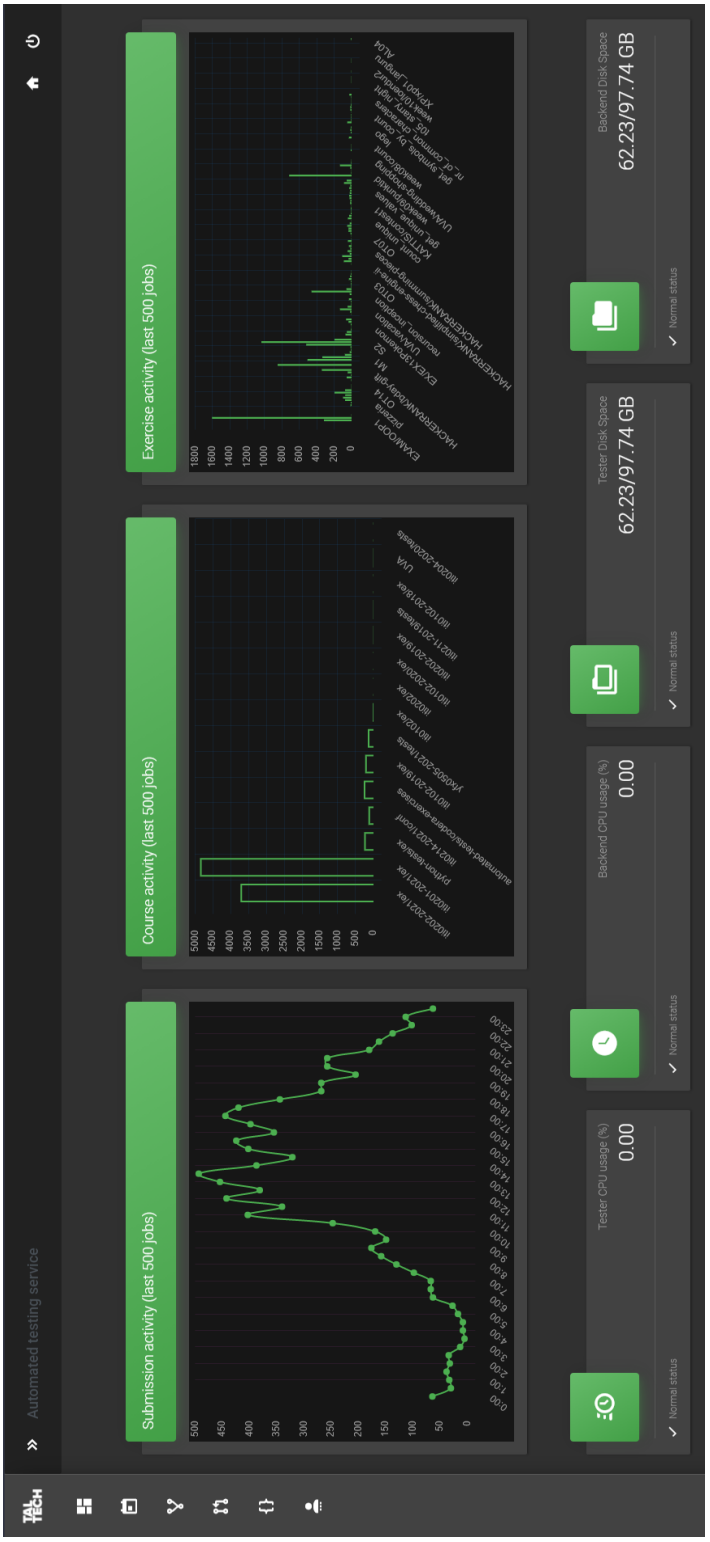
Passed weight: 1

Percentage: 100.0%

Lisa 3. Arete Response klassi diagramm



Lisa 4. Arete UI Dashboard



Lisa 5. Arete UI ülesannete vaade