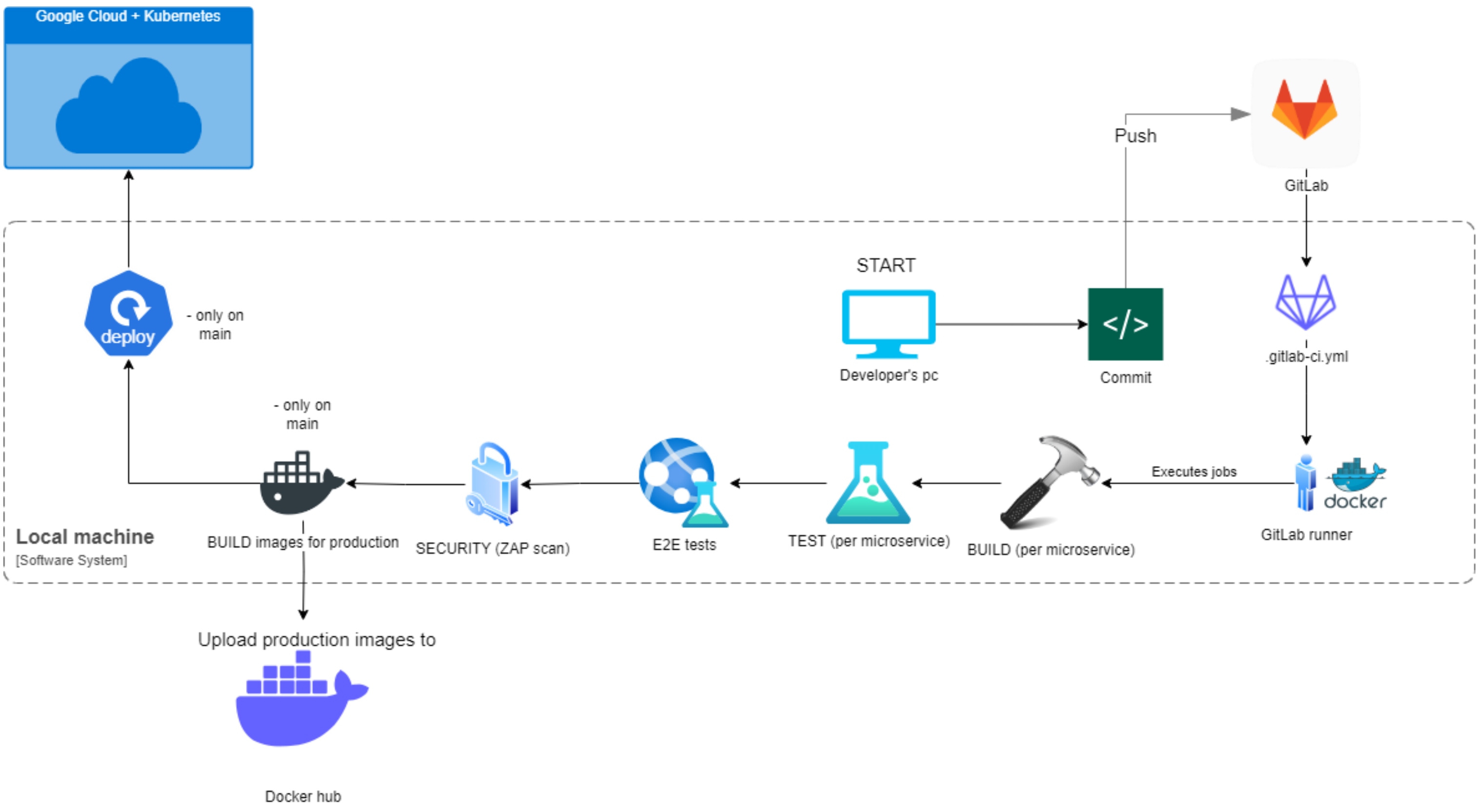
CI/CD Setup in individual project advanced software

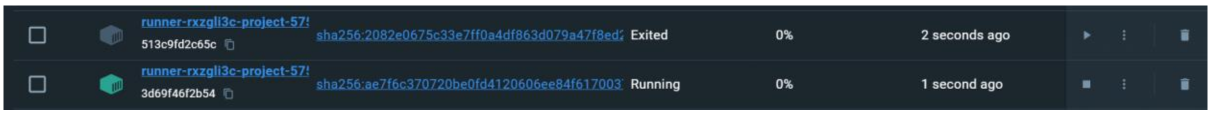
The image below illustrates how the CI pipeline of HouseHunters is structured in Sem.6, along with explanations of each stage.



Runner setup

This semester a GitLab runner is used that runs on the developer's machine. In order to make the pipeline more transferable the runner uses a Docker executor, which means that each pipeline job runs in an isolated docker container. This also gives developers the ability to specify what docker image they want to use for their job, which is a great abstraction from the PC that the runner is on.

Below is an image of what a CI/CD job looks in docker while it is being executed:



```
runner-rxzgli3c-project-57!  sha256:2082e0675c33e7ff0a4df863d079a47f8ed2  Exited   0%  2 seconds ago
513c9fd2c65c
runner-rxzgli3c-project-57!  sha256:ae7f6c370720be0fd4120606ee84f617003   Running  0%  1 second ago
3d69f46f2b54
```

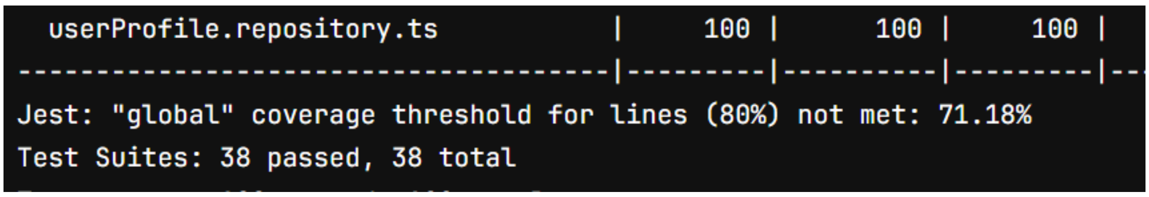
Pipeline stages

Build

The build stage's main purpose is to try to build each respective microservice's code and check for any compilation errors. This would catch early problems with the code itself and prevent the pipeline from going any further.

Test

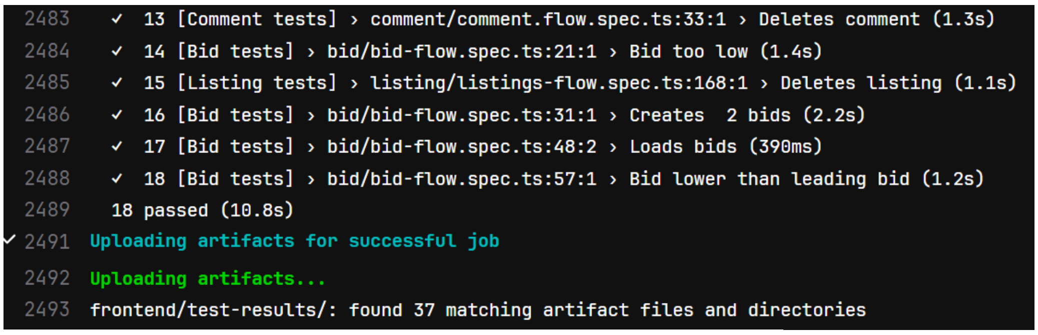
The test stage runs controller, unit and integration tests on each microservice. The acceptance criteria is that for each microservice the code coverage is at least 80%. If the tests result in an insufficient coverage, the job fails.



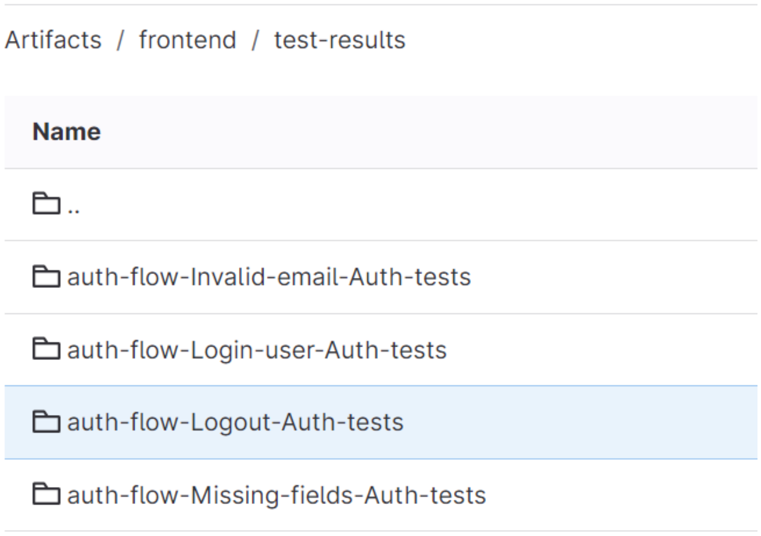
```
  userProfile.repository.ts            |     100 |       100 |     100 |
--------------------------------------|---------|----------|---------|--
Jest: "global" coverage threshold for lines (80%) not met: 71.18%
Test Suites: 38 passed, 38 total
```

E2E

This stage uses docker compose and playwright to start the HouseHunters frontend and all microservices, along with a local MongoDB database and a RabbitMQ container.
After that, Playwright is used to execute end-to-end tests. An unsuccessful test fails the pipeline job and an artifact of the test is saved that can be inspected in trace.playwright.dev



```
  ✓  13 [Comment tests] › comment/comment.flow.spec.ts:33:1 › Deletes comment (1.3s)
  ✓  14 [Bid tests] › bid/bid-flow.spec.ts:21:1 › Bid too low (1.4s)
  ✓  15 [Listing tests] › listing/listings-flow.spec.ts:168:1 › Deletes listing (1.1s)
  ✓  16 [Bid tests] › bid/bid-flow.spec.ts:31:1 › Creates  2 bids (2.2s)
  ✓  17 [Bid tests] › bid/bid-flow.spec.ts:48:2 › Loads bids (390ms)
  ✓  18 [Bid tests] › bid/bid-flow.spec.ts:57:1 › Bid lower than leading bid (1.2s)
  18 passed (10.8s)
Uploading artifacts for successful job
Uploading artifacts...
frontend/test-results/: found 37 matching artifact files and directories
```

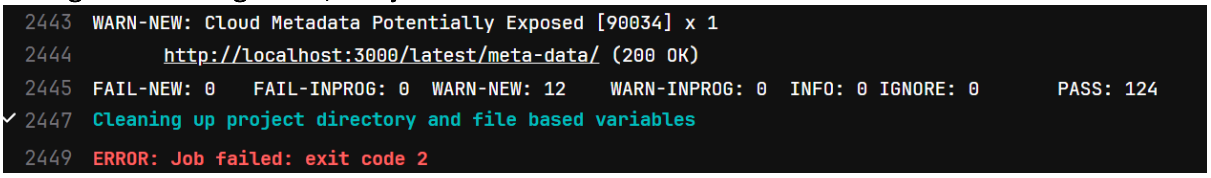


Artifacts / frontend / test-results

Name
..
auth-flow-Invalid-email-Auth-tests
auth-flow-Login-user-Auth-tests
auth-flow-Logout-Auth-tests
auth-flow-Missing-fields-Auth-tests

Security

This stage uses the same docker compose setup from the E2E stage to execute a full scan from OWASP’s ZAP tool. Over 130 security tests are run and if a single on fails and is not configured to be ignored, the job fails.



```
WARN-NEW: Cloud Metadata Potentially Exposed [90034] x 1
        http://localhost:3000/latest/meta-data/ (200 OK)
FAIL-NEW: 0    FAIL-INPROG: 0  WARN-NEW: 12    WARN-INPROG: 0  INFO: 0 IGNORE: 0       PASS: 124
Cleaning up project directory and file based variables
ERROR: Job failed: exit code 2
```