A collaborative peer review process for grading coding assignments in coursework

Pratik Nayak, Fritz Göbel, Hartwig Anzt

Karlsruhe Institute of Technology

June 18th
Workshop on Teaching Computational Science, ICCS 2021

Outline

- Why ?
- What ?
- Where ?
- How ?

Outline

- Why ?
- What ?
- Where ?
- How ?
- Feedback
- Conclusions and Perspectives

Why ?

- Almost all research employs some form of software.
- Software lifecycle often exceeds hardware lifecycles.
- Good sustainable software is **THE** key component of computational science.
- Ingraining good software practices in students is important to their careers in industry and in academia.

What ?

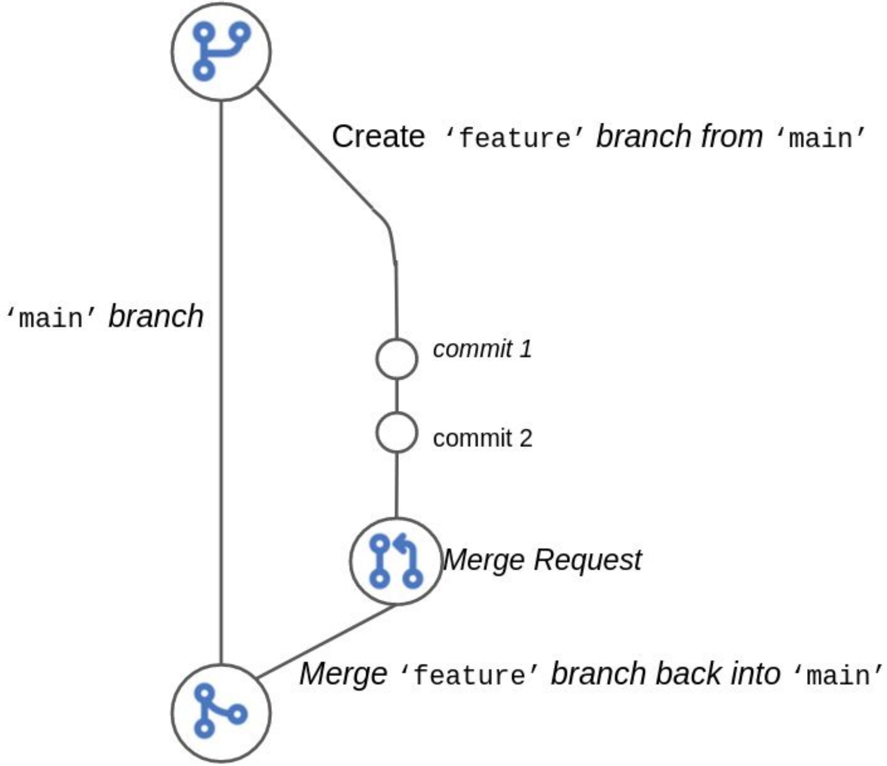
- Version control.
- Continuous Integration.
- Automated testing
- Collaborative peer review.

Version Control

- Offload the code “version” management.
- Eases collaboration and parallel work on a single code base.
- Examples: SVN, Mercurial, git, Perforce etc.

Version Control

- “branch” your work from the main codebase.
- “commit” your distinct changes.
- Request your changes to be merged back into the main codebase.

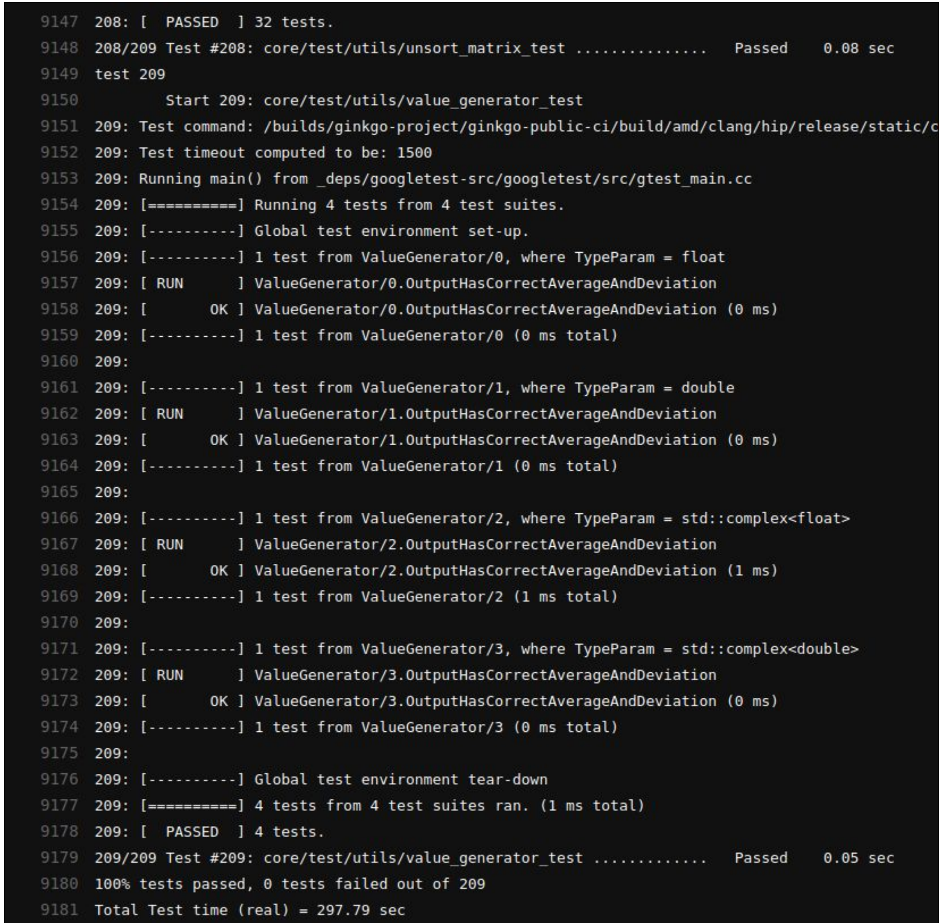


Create ‘feature’ branch from ‘main’
‘main’ branch
commit 1
commit 2
Merge Request
Merge ‘feature’ branch back into ‘main’

A common git workflow

Automated Testing

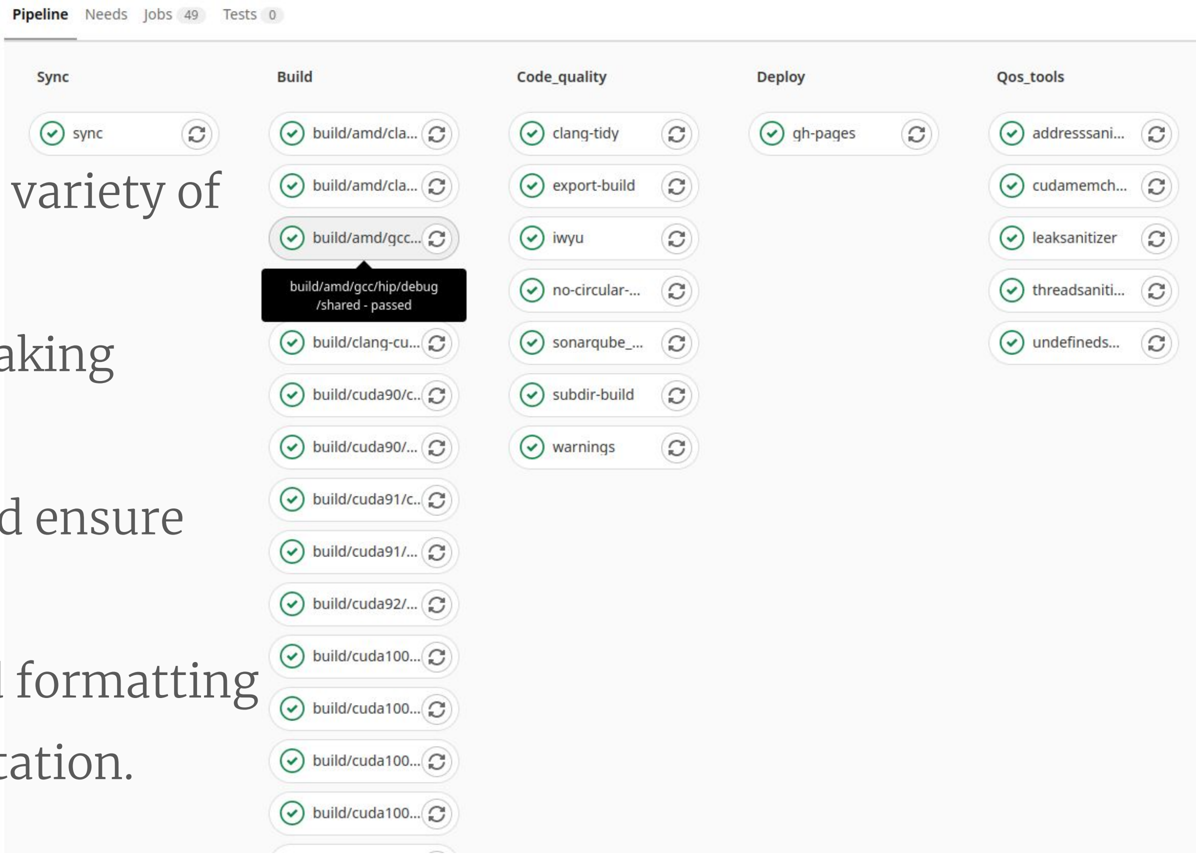
- Verify code correctness and robustness.
- Allow easy identification of bugs in large code-bases.
- Streamline development and integrate verification into development.



```
208: [  PASSED  ] 32 tests.
208/209 Test #208: core/test/utils/unsort_matrix_test ..............   Passed    0.08 sec
test 209
        Start 209: core/test/utils/value_generator_test
209: Test command: /builds/ginkgo-project/ginkgo-public-ci/build/amd/clang/hip/release/static/c
209: Test timeout computed to be: 1500
209: Running main() from _deps/googletest-src/googletest/src/gtest_main.cc
209: [==========] Running 4 tests from 4 test suites.
209: [----------] Global test environment set-up.
209: [----------] 1 test from ValueGenerator/0, where TypeParam = float
209: [ RUN      ] ValueGenerator/0.OutputHasCorrectAverageAndDeviation
209: [       OK ] ValueGenerator/0.OutputHasCorrectAverageAndDeviation (0 ms)
209: [----------] 1 test from ValueGenerator/0 (0 ms total)
209:
209: [----------] 1 test from ValueGenerator/1, where TypeParam = double
209: [ RUN      ] ValueGenerator/1.OutputHasCorrectAverageAndDeviation
209: [       OK ] ValueGenerator/1.OutputHasCorrectAverageAndDeviation (0 ms)
209: [----------] 1 test from ValueGenerator/1 (0 ms total)
209:
209: [----------] 1 test from ValueGenerator/2, where TypeParam = std::complex<float>
209: [ RUN      ] ValueGenerator/2.OutputHasCorrectAverageAndDeviation
209: [       OK ] ValueGenerator/2.OutputHasCorrectAverageAndDeviation (1 ms)
209: [----------] 1 test from ValueGenerator/2 (1 ms total)
209:
209: [----------] 1 test from ValueGenerator/3, where TypeParam = std::complex<double>
209: [ RUN      ] ValueGenerator/3.OutputHasCorrectAverageAndDeviation
209: [       OK ] ValueGenerator/3.OutputHasCorrectAverageAndDeviation (0 ms)
209: [----------] 1 test from ValueGenerator/3 (0 ms total)
209:
209: [----------] Global test environment tear-down
209: [==========] 4 tests from 4 test suites ran. (1 ms total)
209: [  PASSED  ] 4 tests.
209/209 Test #209: core/test/utils/value_generator_test ............   Passed    0.05 sec
100% tests passed, 0 tests failed out of 209
Total Test time (real) = 297.79 sec
```

Continuous integration

- Compile and test code on a variety of platforms and machines.
- Automatically identify breaking changes.
- Perform static analyses and ensure high code quality.
- Verify code uniformity and formatting
- Deploy code and documentation.



Pipeline Needs Jobs 49 Tests 0

Sync
sync

Build
build/amd/cla...
build/amd/cla...
build/amd/gcc...
build/amd/gcc/hip/debug/shared - passed
build/clang-cu...
build/cuda90/c...
build/cuda90/...
build/cuda91/c...
build/cuda91/...
build/cuda92/...
build/cuda100...
build/cuda100...
build/cuda100...
build/cuda100...

Code_quality
clang-tidy
export-build
iwyu
no-circular-...
sonarqube_...
subdir-build
warnings

Deploy
gh-pages

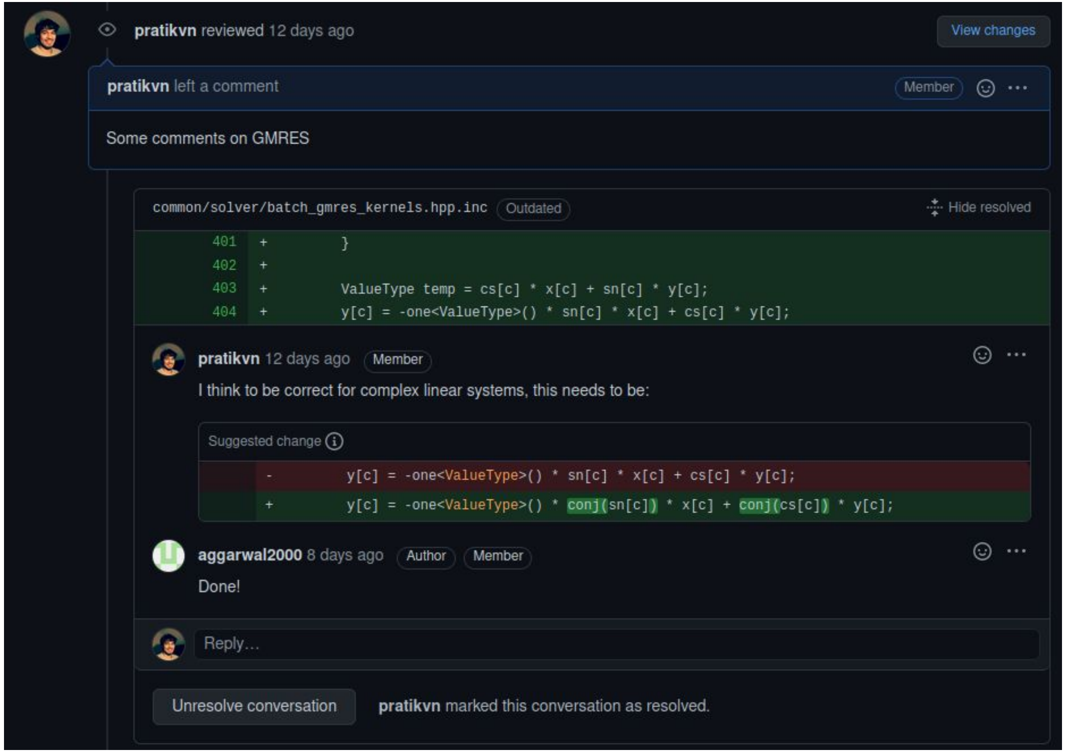
Qos_tools
addresssani...
cudamemch...
leaksanitizer
threadsaniti...
undefineds...

Peer review - Platforms

- Many platforms available for collaborative coding.
- Gitlab, Github, Bitbucket are examples of such platforms.
- They provide web-based interface
  - to interact with code,
  - give feedback on specific lines of code,
  - show pass/failure of the CI pipelines

Peer review – code review

- Peer review is essential in collaborative software development.
- View code critically in context of the entire project.
- Making sure the code follows the code guidelines.
- Looking for performance and correctness gotchas.



pratikvn reviewed 12 days ago

pratikvn left a comment

Some comments on GMRES

common/solver/batch_gmres_kernels.hpp.inc

```
+        }
+
+        ValueType temp = cs[c] * x[c] + sn[c] * y[c];
+        y[c] = -one<ValueType>() * sn[c] * x[c] + cs[c] * y[c];
```

pratikvn 12 days ago

I think to be correct for complex linear systems, this needs to be:

Suggested change

```
-        y[c] = -one<ValueType>() * sn[c] * x[c] + cs[c] * y[c];
+        y[c] = -one<ValueType>() * conj(sn[c]) * x[c] + conj(cs[c]) * y[c];
```

aggarwal2000 8 days ago

Done!

pratikvn marked this conversation as resolved.

Where ?

- Numerical Linear Algebra for High Performance Computing course at Karlsruhe Institute of Technology as a pilot programme.
- Course content:
  - Parallel programming basics.
  - OpenMP, CUDA and MPI programming models.
  - BLAS kernels and their implementations.
  - Iterative solvers.
  - Preconditioners.

Target audience: Master students in the Math and Computer Science departments.

How ?

- Incorporate these practices into the course.
- Encourage students to experiment with algorithms and implementations and not be bogged down in build and platform issues.
- Homework schedule:
    - HW1: Basic Linear Algebra operations.
    - HW2: Dense Matrix-Vector multiplication with OpenMP and CUDA.
    - HW3: LU factorization with OpenMP tasking framework.
    - HW4: Sparse Matrix-Vector multiplication with OpenMP and CUDA.
    - HW5: An Iterative linear system solver with CUDA.

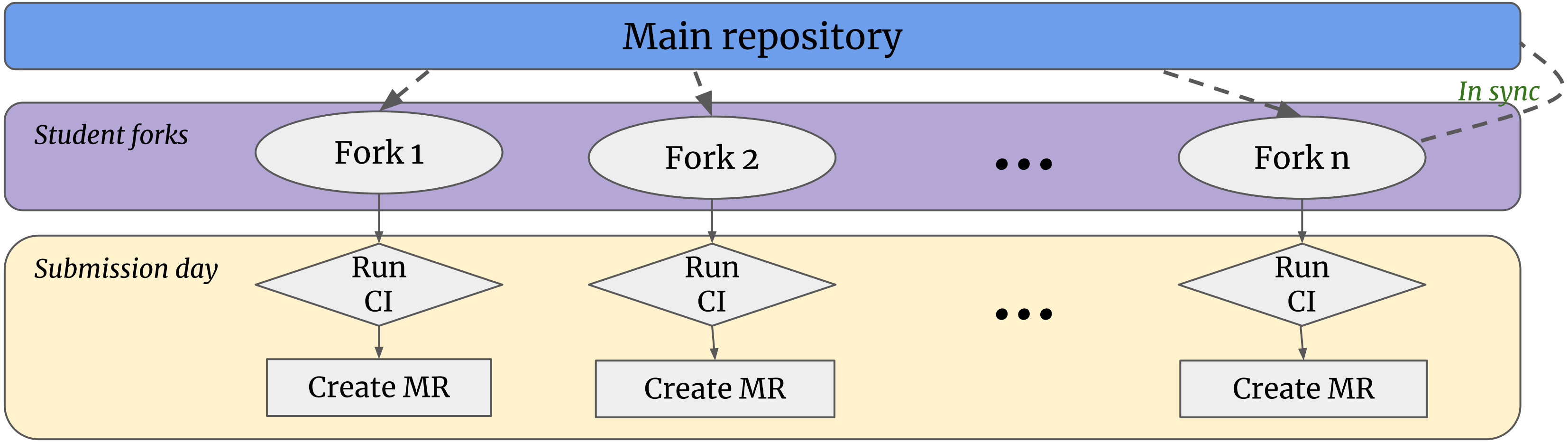
How ?

Main repository

- Create a common Exercise framework for all to work on.
- Provide the building blocks: Compilation, testing and benchmarking frameworks and setup a Continuous Integration setup to automatically test the code on push.

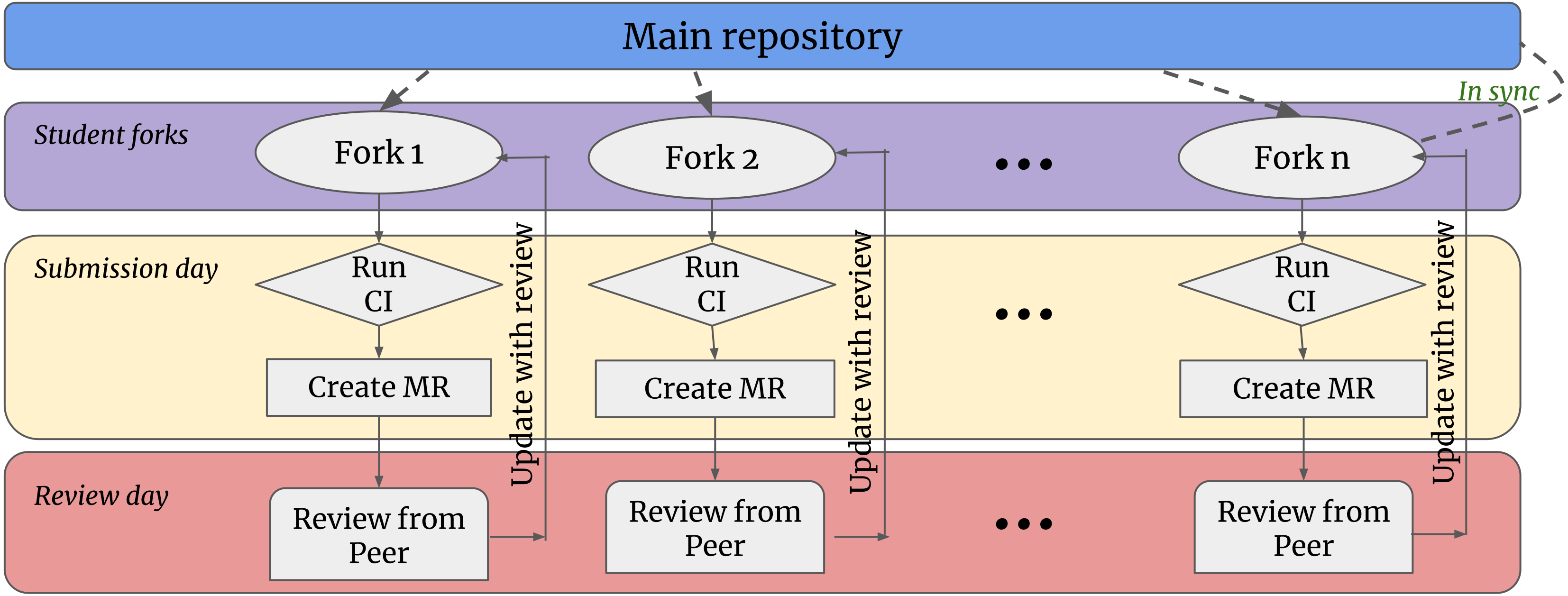
https://github.com/pratikvn/nla4hpc-exercises-framework

How ?



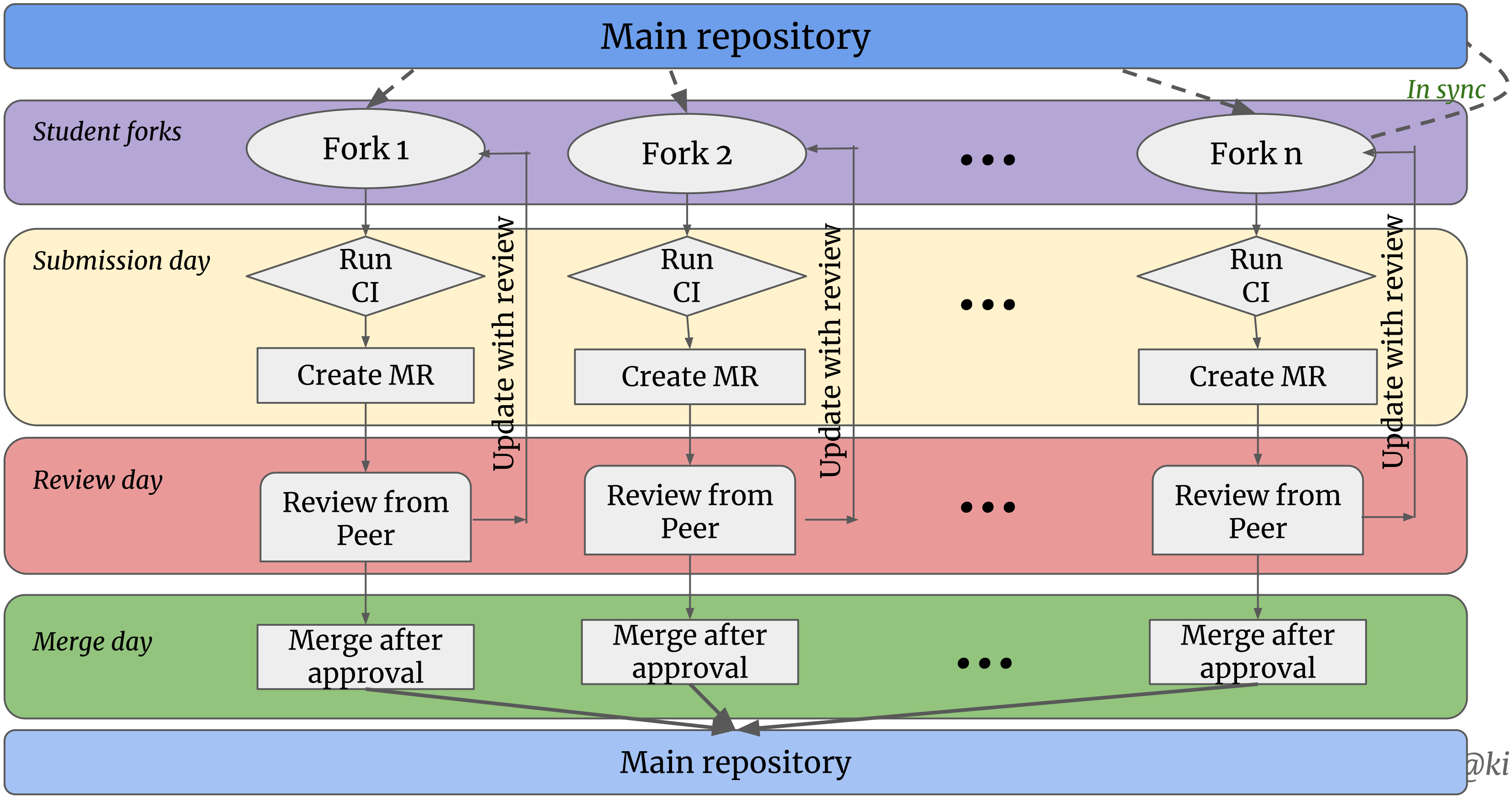
- Students create forks from the main repository, create separate branches for each HW and on submission date submit a Merge Request to merge the changes back to the main repository.

How ?

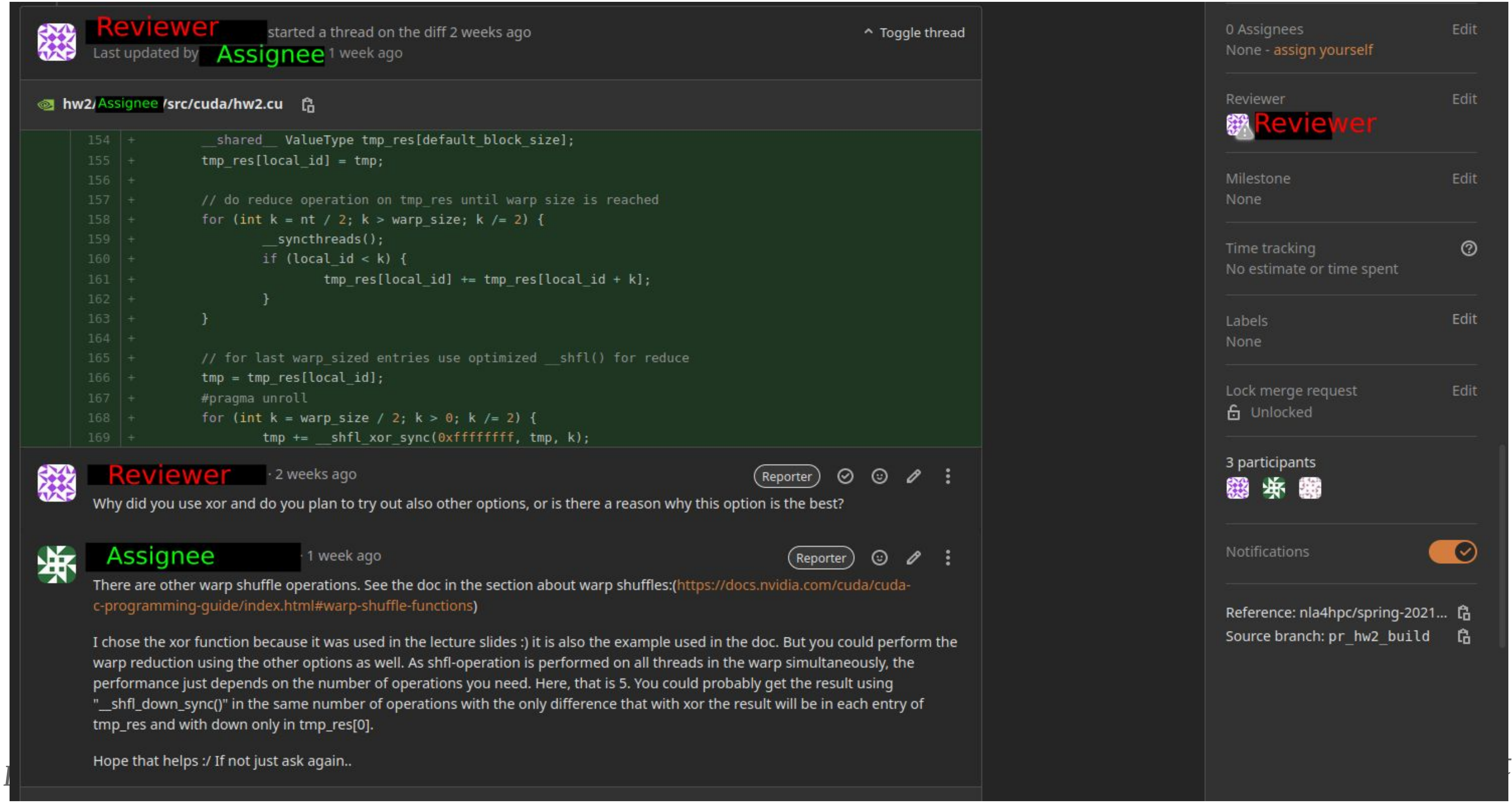


- They are assigned a Merge Request to review and have certain guidelines to follow.
- At the deadline, the Merge Requests are merged back into the main branch and graded.

How ?


Main repository
In sync
Student forks
Fork 1
Fork 2
Fork n
Submission day
Run CI
Run CI
Run CI
Create MR
Create MR
Create MR
Update with review
Update with review
Update with review
Review day
Review from Peer
Review from Peer
Review from Peer
Merge day
Merge after approval
Merge after approval
Merge after approval
Main repository

Peer review in action



Reviewer [redacted] started a thread on the diff 2 weeks ago
Last updated by Assignee [redacted] 1 week ago

hw2/Assignee [redacted]/src/cuda/hw2.cu

```
+           __shared__ ValueType tmp_res[default_block_size];
+           tmp_res[local_id] = tmp;
+
+           // do reduce operation on tmp_res until warp size is reached
+           for (int k = nt / 2; k > warp_size; k /= 2) {
+                   __syncthreads();
+                   if (local_id < k) {
+                           tmp_res[local_id] += tmp_res[local_id + k];
+                   }
+           }
+
+           // for last warp_sized entries use optimized __shfl() for reduce
+           tmp = tmp_res[local_id];
+           #pragma unroll
+           for (int k = warp_size / 2; k > 0; k /= 2) {
+                   tmp += __shfl_xor_sync(0xffffffff, tmp, k);
```

Reviewer [redacted] · 2 weeks ago

Why did you use xor and do you plan to try out also other options, or is there a reason why this option is the best?

Assignee [redacted] · 1 week ago

There are other warp shuffle operations. See the doc in the section about warp shuffles:(https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#warp-shuffle-functions)

I chose the xor function because it was used in the lecture slides :) it is also the example used in the doc. But you could perform the warp reduction using the other options as well. As shfl-operation is performed on all threads in the warp simultaneously, the performance just depends on the number of operations you need. Here, that is 5. You could probably get the result using "__shfl_down_sync()" in the same number of operations with the only difference that with xor the result will be in each entry of tmp_res and with down only in tmp_res[0].

Hope that helps :/ If not just ask again..

0 Assignees — Edit
None - assign yourself

Reviewer — Edit
Reviewer [redacted]

Milestone — Edit
None

Time tracking
No estimate or time spent

Labels — Edit
None

Lock merge request — Edit
Unlocked

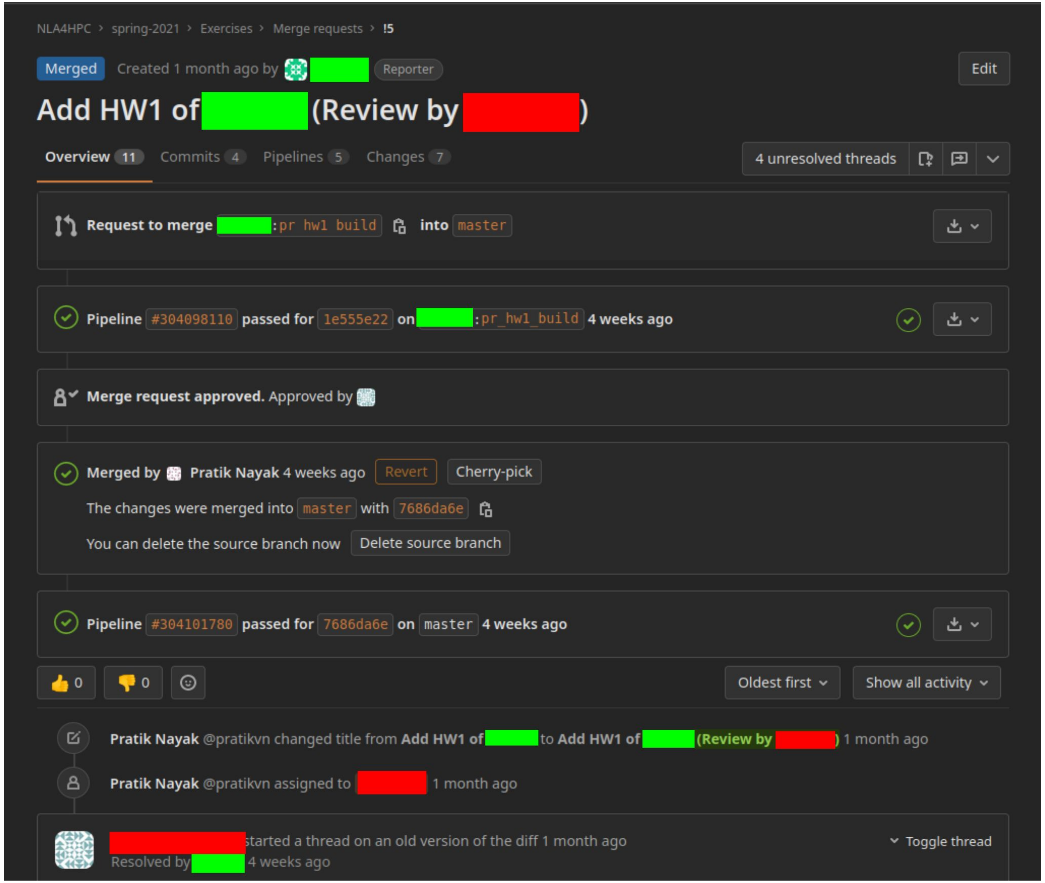
3 participants

Notifications

Reference: nla4hpc/spring-2021...
Source branch: pr_hw2_build

Peer review in action

- CI is run on the forks.
- Discussion on the gitlab web interface.
- Approval from reviewer after addressing the comments.
- Merge after approval.
- Green is assignee
- Red is reviewer.



NLA4HPC › spring-2021 › Exercises › Merge requests › !5

Merged Created 1 month ago by [redacted] Reporter

Edit

Add HW1 of [redacted] (Review by [redacted])

Overview 11 Commits 4 Pipelines 5 Changes 7

4 unresolved threads

Request to merge [redacted]:pr hw1 build into master

Pipeline #304098110 passed for 1e555e22 on [redacted]:pr_hw1_build 4 weeks ago

Merge request approved. Approved by

Merged by Pratik Nayak 4 weeks ago Revert Cherry-pick

The changes were merged into master with 7686da6e

You can delete the source branch now Delete source branch

Pipeline #304101780 passed for 7686da6e on master 4 weeks ago

👍 0 👎 0

Oldest first Show all activity

Pratik Nayak @pratikvn changed title from Add HW1 of [redacted] to Add HW1 of [redacted] (Review by [redacted]) 1 month ago

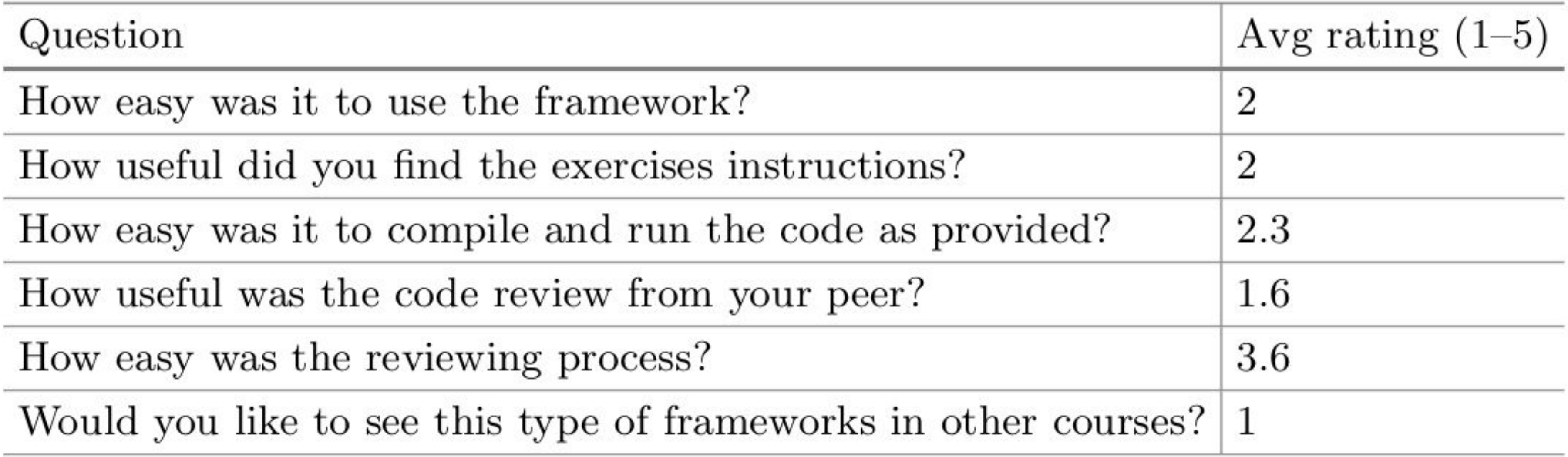
Pratik Nayak @pratikvn assigned to [redacted] 1 month ago

[redacted] started a thread on an old version of the diff 1 month ago

Toggle thread

Resolved by [redacted] 4 weeks ago

Feedback.



Question	Avg rating (1–5)
How easy was it to use the framework?	2
How useful did you find the exercises instructions?	2
How easy was it to compile and run the code as provided?	2.3
How useful was the code review from your peer?	1.6
How easy was the reviewing process?	3.6
Would you like to see this type of frameworks in other courses?	1

Some feedback from students. (N=4)

Very useful/ Very easy 1 ← → 5 Not useful/ Not easy

Conclusions and Perspectives.

- We saw a marked improvement in code quality as the course progressed, which was not the case in our previous course offerings.
- This approach seems more scalable even with a lot more students.
    - It can be almost completely automated.
- The students were able to focus on algorithms and optimizations rather than on build system and other orthogonal issues.
- It encourages students to showcase their code and makes them comfortable with contributing to open-source projects.

Thank you!

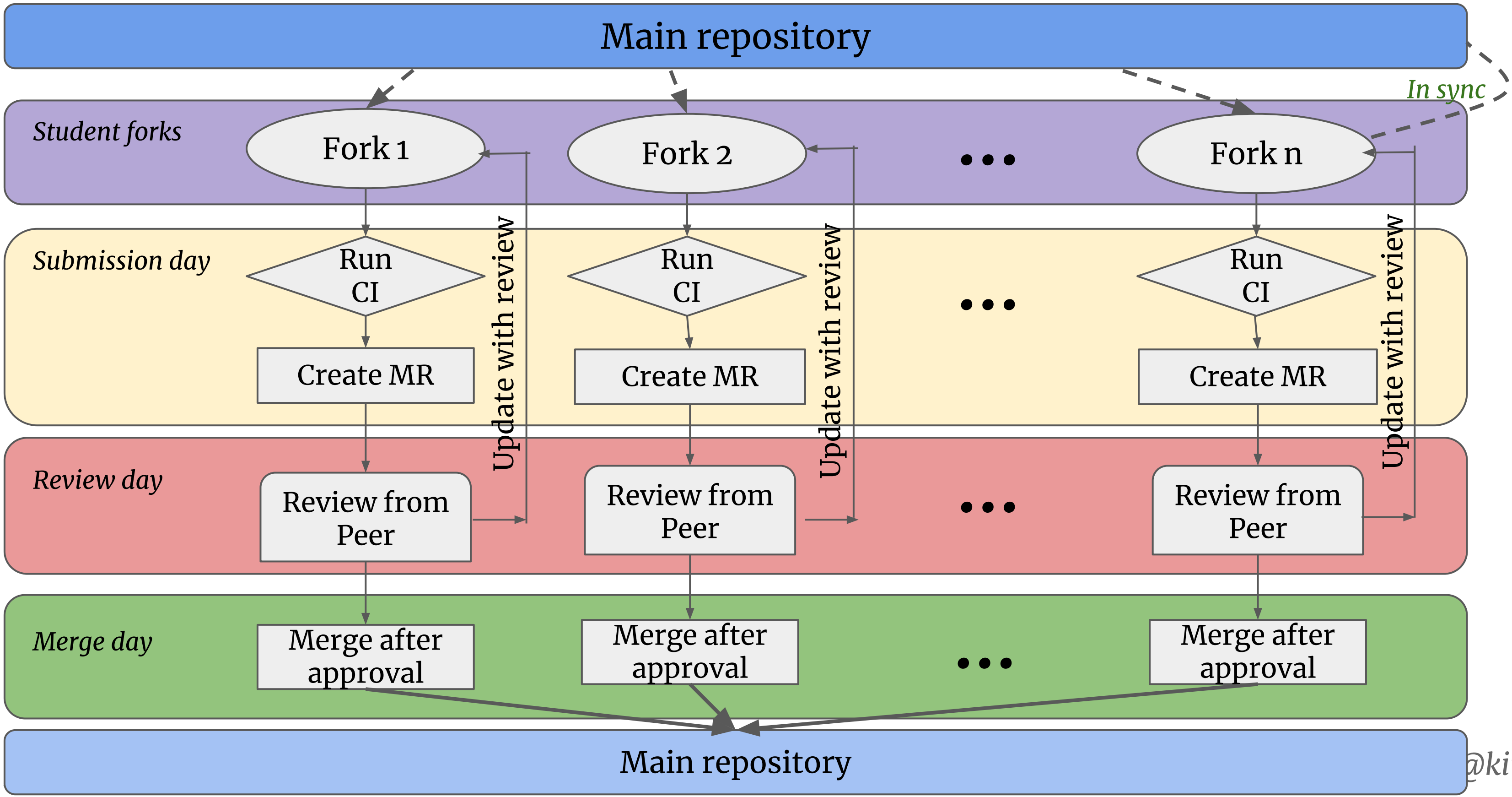
nayak@kit.edu

Backup

How ?

- Create a common Exercise framework for all to work on.
- Provide the building blocks: Compilation, testing and benchmarking frameworks and setup a Continuous Integration setup to automatically test the code on push.
- Students create forks from the main repository, create separate branches for each HW and on submission date submit a Merge Request to merge the changes back to the main repository.
- They are assigned a Merge Request to review and have certain guidelines to follow.
- At the deadline, the Merge Requests are merged back into the main branch and graded.
- Students submit an additional report analyzing their code and performance.

How ?



Main repository

In sync

Student forks

Fork 1

Fork 2

Fork n

Submission day

Run CI

Create MR

Update with review

Review day

Review from Peer

Merge day

Merge after approval

Main repository