Review Board: Paper Trail

Team 12 Handover Package

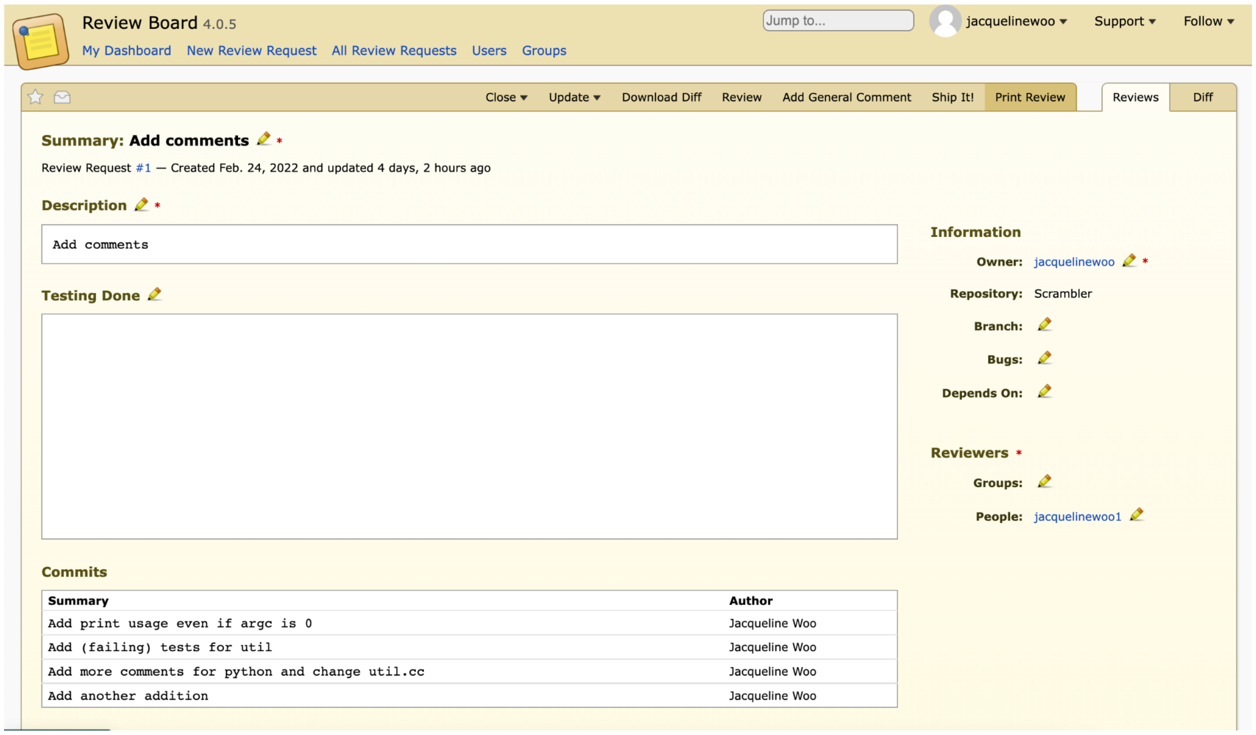
Table of Contents

User Manual

Since our project revolves around making any review accessible to all user roles, the appropriate documentation provided is applicable to all Review Board user roles. Here we document how to interact with each of our new capabilities. Specifically, we describe how to interact with the new changes as a whole, and then go into detail regarding each of the print configurations.

Accessing Print Review Page and Printing Review

The print preview page in Figure 1.2 is easily accessed through the “Print Review” button on a review page (Figure 1.1). From there, the user can configure their print preview page by selecting various options on the left side print configuration panel. By selecting the “Update Preview” button below the print configurations, the right side print preview panel updates based on the configurations selected. Finally, the user can press the “Print” button at the bottom of the page to directly download the review, formatted almost identically to how it is displayed on the right side print preview panel. Clicking the “Print” button displays a popup dialog box stating the number of file diffs that are included in the review (Figure 1.3). This box prompts the user to confirm that they would like to proceed in downloading the file. The user has the opportunity to cancel the download at this point. Otherwise, the review is automatically downloaded as an HTML file.



Summary: Add comments *

Review Request #1 — Created Feb. 24, 2022 and updated 4 days, 2 hours ago

Description *

```
Add comments
```

Testing Done

Information

Owner:	jacquelinewoo *
Repository:	Scrambler
Branch:	
Bugs:	
Depends On:	

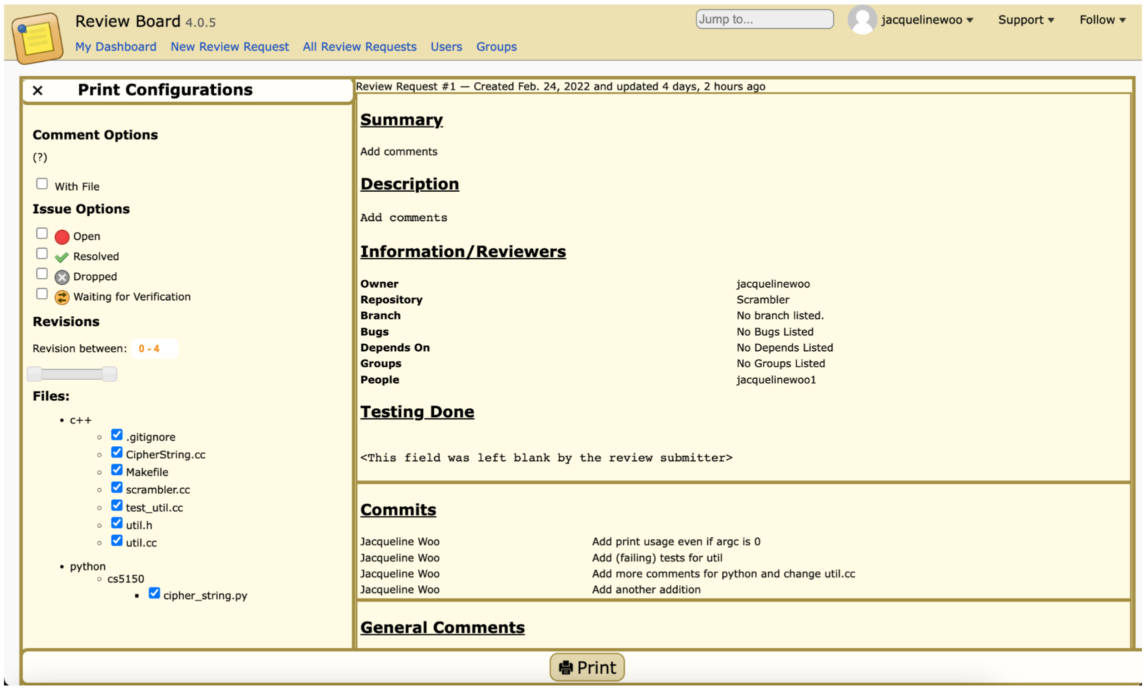
Reviewers *

Groups:	
People:	jacquelinewoo1

Commits

Summary	Author
Add print usage even if argc is 0	Jacqueline Woo
Add (failing) tests for util	Jacqueline Woo
Add more comments for python and change util.cc	Jacqueline Woo
Add another addition	Jacqueline Woo

Figure 1.1: Print Review Button on the existing Review Board Review Page.



Print Configurations

Comment Options

(?)

- [ ] With File

Issue Options

- [ ] Open
- [ ] Resolved
- [ ] Dropped
- [ ] Waiting for Verification

Revisions

Revision between: 0 - 4

Files:

- c++
  - [✓] .gitignore
  - [✓] CipherString.cc
  - [✓] Makefile
  - [✓] scrambler.cc
  - [✓] test_util.cc
  - [✓] util.h
  - [✓] util.cc
- python
  - cs5150
    - [✓] cipher_string.py

Review Request #1 — Created Feb. 24, 2022 and updated 4 days, 2 hours ago

Summary

Add comments

Description

```
Add comments
```

Information/Reviewers

Owner	jacquelinewoo
Repository	Scrambler
Branch	No branch listed.
Bugs	No Bugs Listed
Depends On	No Depends Listed
Groups	No Groups Listed
People	jacquelinewoo1

Testing Done

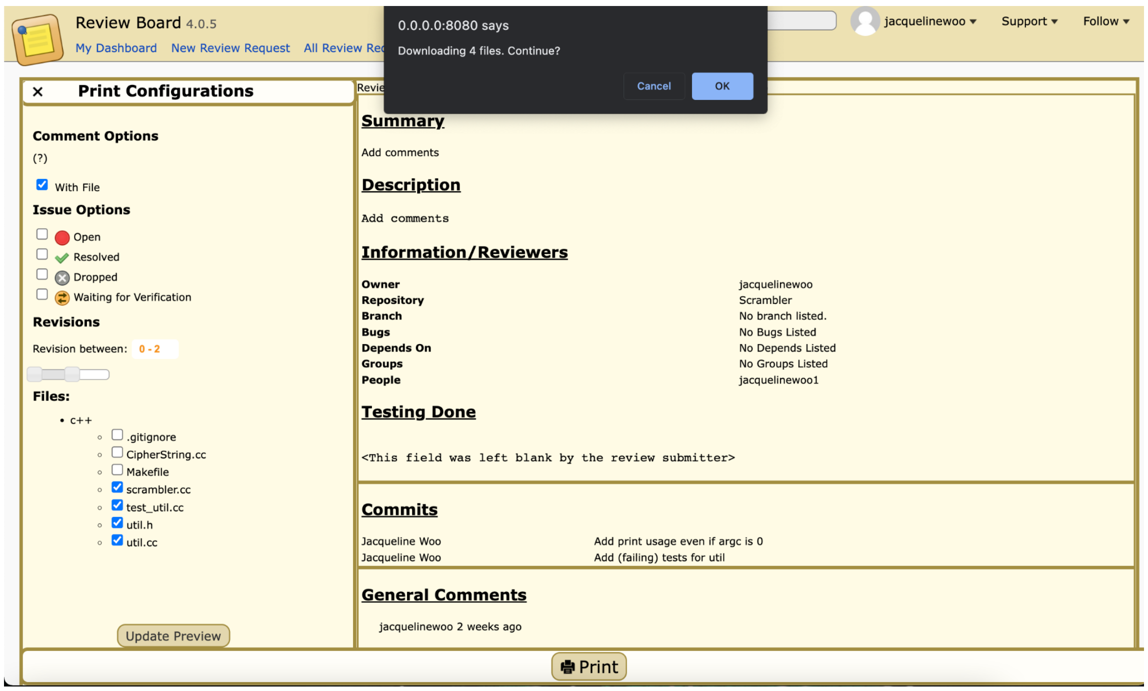
```
<This field was left blank by the review submitter>
```

Commits

Jacqueline Woo	Add print usage even if argc is 0
Jacqueline Woo	Add (failing) tests for util
Jacqueline Woo	Add more comments for python and change util.cc
Jacqueline Woo	Add another addition

General Comments

Figure 1.2: The Print Review Page. It has a print configuration panel on the left side and a print preview panel on the right side. The "Print" button at the bottom.



0.0.0.0:8080 says

Downloading 4 files. Continue?

Cancel OK

× **Print Configurations**

Comment Options

(?)

[✓] With File

Issue Options

[ ] Open
[ ] Resolved
[ ] Dropped
[ ] Waiting for Verification

Revisions

Revision between: 0 - 2

Files:

- c++
  - [ ] .gitignore
  - [ ] CipherString.cc
  - [ ] Makefile
  - [✓] scramblec.cc
  - [✓] test_util.cc
  - [✓] util.h
  - [✓] util.cc

Update Preview

Summary

Add comments

Description

Add comments

Information/Reviewers

Owner	jacquelinewoo
Repository	Scrambler
Branch	No branch listed.
Bugs	No Bugs Listed
Depends On	No Depends Listed
Groups	No Groups Listed
People	jacquelinewoo1

Testing Done

<This field was left blank by the review submitter>

Commits

Jacqueline Woo	Add print usage even if argc is 0
Jacqueline Woo	Add (failing) tests for util

General Comments

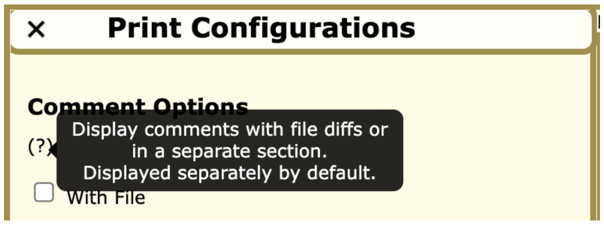
jacquelinewoo 2 weeks ago

Print

Figure 1.3: Popup Dialog.

Code Comments

The first configuration is the ability to choose having code comments attached to file diffs or in a separate comments section. A tooltip hover shown in Figure 2.1 provides additional information on how comments are shown and explains the “With File” checkbox in the section. By default, the code comments are in a separate comments section, as shown by Figure 2.2. If there are no code comments and the comments are to be displayed in the comments section, the HTML will state there are no code comments in the review. Currently, inline comments are not supported. Instead, they are in-file comments, where code comments that are related to a file are displayed directly below the file diff, as shown by Figure 2.3. Only code comments that are relevant to the revisions shown on the revision slider are displayed; this internally means that a code comment is shown only if its diffset is the same as the right slider diffset.



× Print Configurations

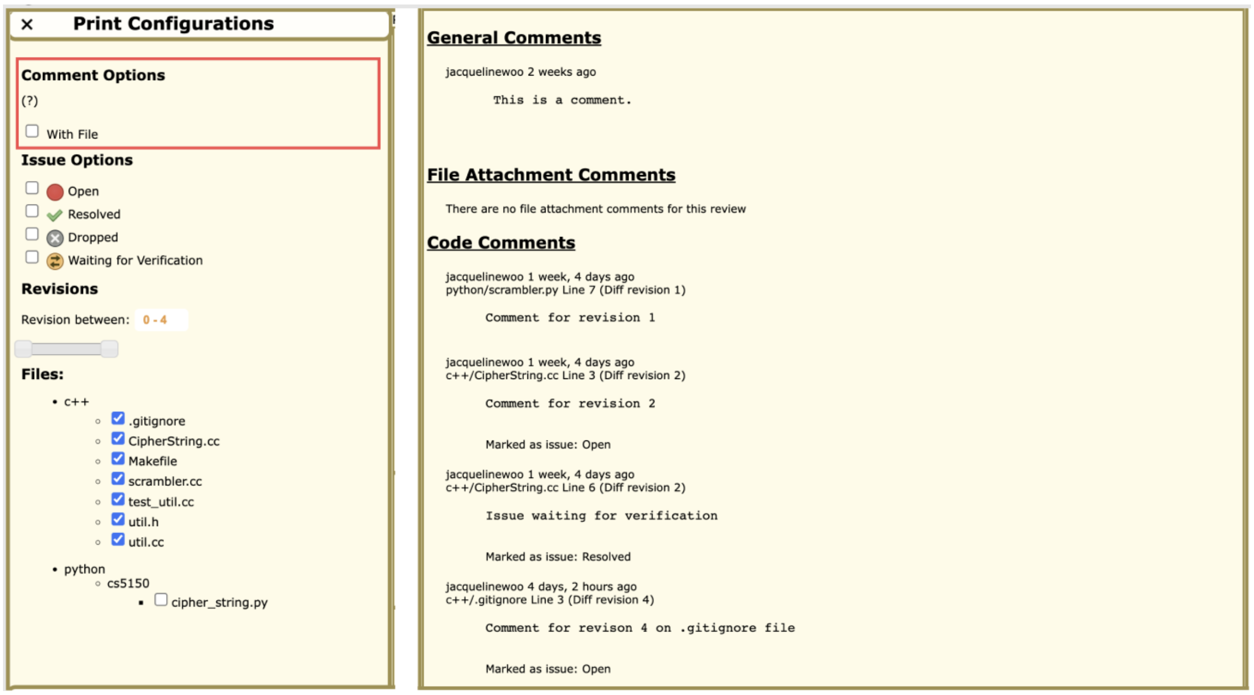
Comment Options

(?)

Display comments with file diffs or in a separate section. Displayed separately by default.

☐ With File

Figure 2.1: Comment Options Tooltip.



× Print Configurations

Comment Options

(?)

☐ With File

Issue Options

☐ Open

☐ Resolved

☐ Dropped

☐ Waiting for Verification

Revisions

Revision between: 0 - 4

Files:

- c++
  - ☑ .gitignore
  - ☑ CipherString.cc
  - ☑ Makefile
  - ☑ scrambler.cc
  - ☑ test_util.cc
  - ☑ util.h
  - ☑ util.cc
- python
  - cs5150
    - ☐ cipher_string.py

General Comments

jacquelinewoo 2 weeks ago

```
This is a comment.
```

File Attachment Comments

There are no file attachment comments for this review

Code Comments

jacquelinewoo 1 week, 4 days ago
python/scrambler.py Line 7 (Diff revision 1)

```
Comment for revision 1
```

jacquelinewoo 1 week, 4 days ago
c++/CipherString.cc Line 3 (Diff revision 2)

```
Comment for revision 2
```

Marked as issue: Open

jacquelinewoo 1 week, 4 days ago
c++/CipherString.cc Line 6 (Diff revision 2)

```
Issue waiting for verification
```

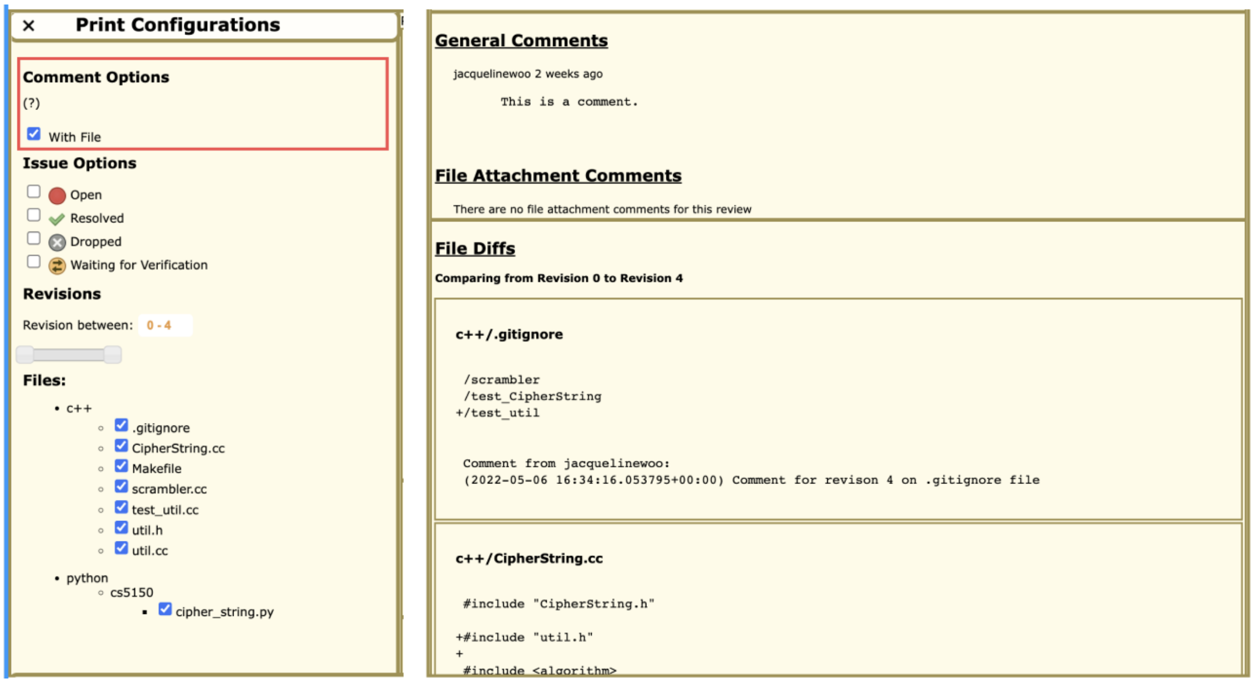
Marked as issue: Resolved

jacquelinewoo 4 days, 2 hours ago
c++/.gitignore Line 3 (Diff revision 4)

```
Comment for revison 4 on .gitignore file
```

Marked as issue: Open

Figure 2.2: Comment Options Section (highlighted). On the right side, code comments are in the comments section.


× Print Configurations

Comment Options
(?)
[✓] With File

Issue Options
[ ] Open
[ ] Resolved
[ ] Dropped
[ ] Waiting for Verification

Revisions
Revision between: 0 - 4

Files:
- c++
  - [✓] .gitignore
  - [✓] CipherString.cc
  - [✓] Makefile
  - [✓] scrambler.cc
  - [✓] test_util.cc
  - [✓] util.h
  - [✓] util.cc
- python
  - cs5150
    - [✓] cipher_string.py

General Comments

jacquelinewoo 2 weeks ago

```
This is a comment.
```

File Attachment Comments

There are no file attachment comments for this review

File Diffs

Comparing from Revision 0 to Revision 4

c++/.gitignore

```
 /scrambler
 /test_CipherString
+/test_util


 Comment from jacquelinewoo:
 (2022-05-06 16:34:16.053795+00:00) Comment for revison 4 on .gitignore file
```

c++/CipherString.cc

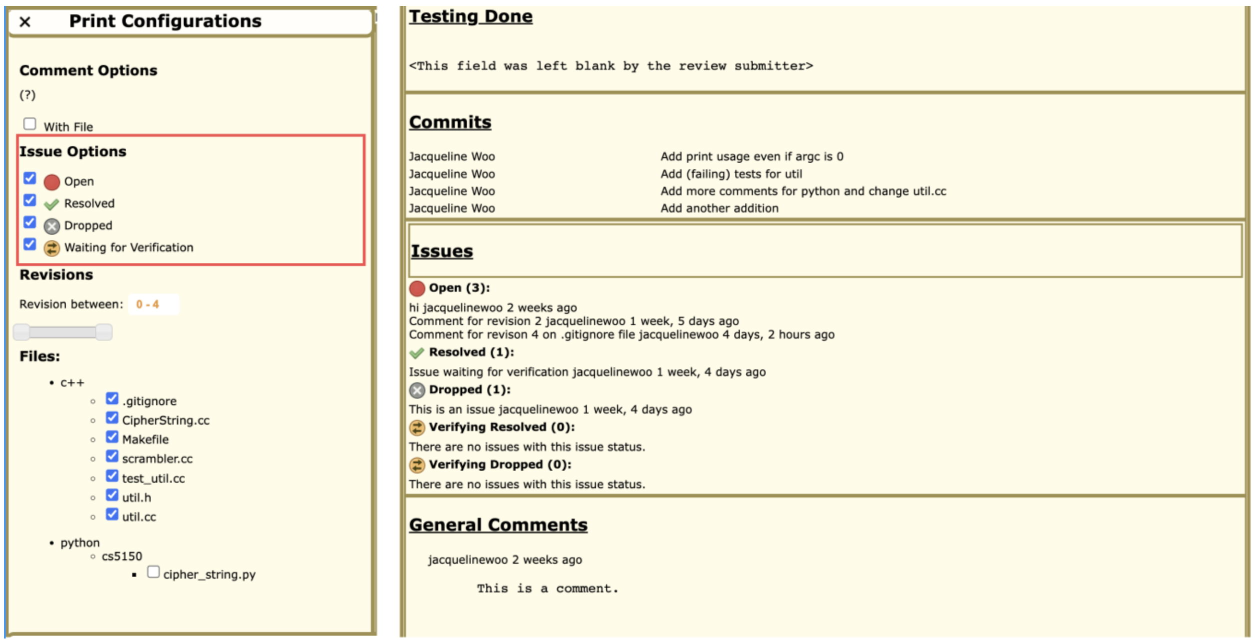
```
 #include "CipherString.h"

+#include "util.h"
+
 #include <algorithm>
```

Figure 2.3: "With File" checkbox is selected. On the right side, code comments are underneath the file diff.

Issue Types

The next configuration is the ability to select different issue types. By default, there are no issues selected, and, as such, there is no Issues section in the print preview panel. If the review doesn't have an issue, the checkboxes related to issues are disabled and the user is unable to interact with them. If the review has an issue, the user may select any issue type. Otherwise, the user can select any combination of open, dropped, resolved, or waiting-for-verification issues. Figure 3.1 displays a review with all the issue types selected. If there are no issues of a specific type, the HTML file will state there are no issues of that type.


× **Print Configurations**

Comment Options
(?)
☐ With File

Issue Options
☑ ● Open
☑ ✓ Resolved
☑ ⊗ Dropped
☑ Waiting for Verification

Revisions
Revision between: 0 - 4

Files:
- c++
  - ☑ .gitignore
  - ☑ CipherString.cc
  - ☑ Makefile
  - ☑ scrambler.cc
  - ☑ test_util.cc
  - ☑ util.h
  - ☑ util.cc
- python
  - cs5150
    - ☐ cipher_string.py

Testing Done

```
<This field was left blank by the review submitter>
```

Commits

Jacqueline Woo	Add print usage even if argc is 0
Jacqueline Woo	Add (failing) tests for util
Jacqueline Woo	Add more comments for python and change util.cc
Jacqueline Woo	Add another addition

Issues

● **Open (3):**
hi jacquelinewoo 2 weeks ago
Comment for revision 2 jacquelinewoo 1 week, 5 days ago
Comment for revison 4 on .gitignore file jacquelinewoo 4 days, 2 hours ago
✓ **Resolved (1):**
Issue waiting for verification jacquelinewoo 1 week, 4 days ago
⊗ **Dropped (1):**
This is an issue jacquelinewoo 1 week, 4 days ago
Verifying Resolved (0):
There are no issues with this issue status.
Verifying Dropped (0):
There are no issues with this issue status.

General Comments

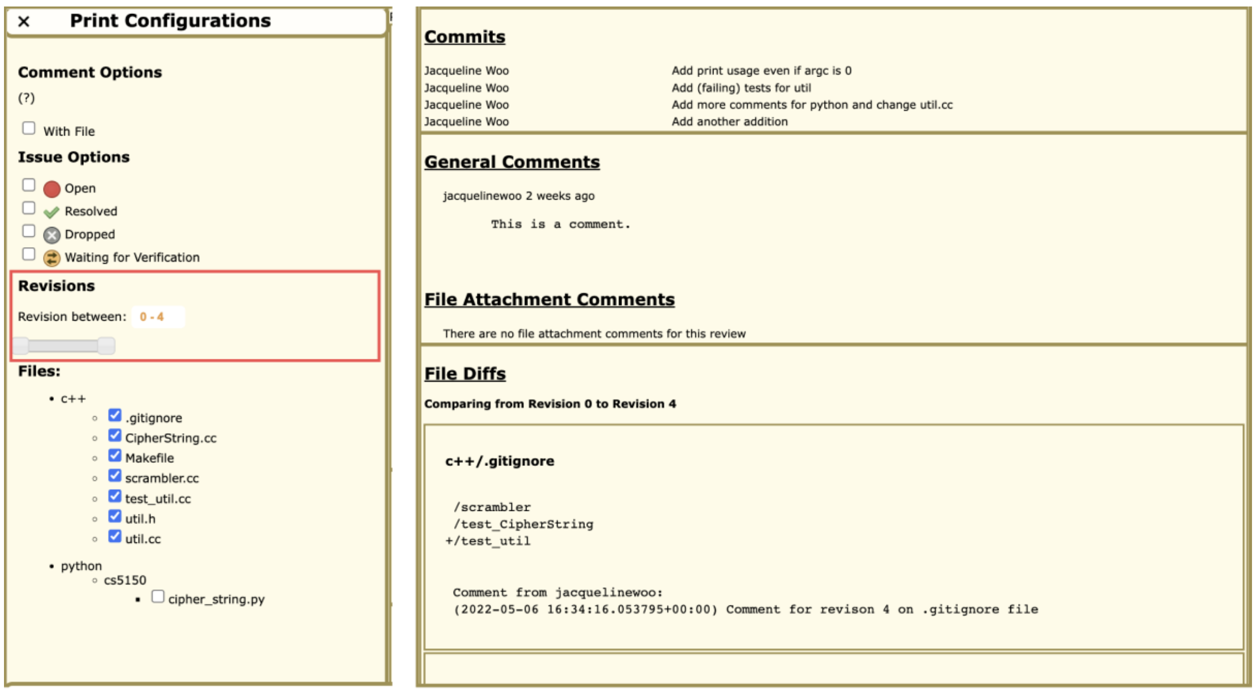
jacquelinewoo 2 weeks ago

```
This is a comment.
```

Figure 3.1: Issue Options Section. On the right side, the issues are displayed between the Commits and General Comments Section.

Revision Slider

The revision slider is the next configurable option and allows the user to choose the start and end revisions for the print review to include. By default, the 0th (original) and most recent revision are at the ends of the slider, as shown by Figure 4.1. The revision slider changes the file selection tree on the left side print configurations panel as well as the commit messages and diffs shown on the right side print preview panel, as demonstrated in Figure 4.2. Currently, the revision slider, signaling the interdiff value, works as expected only for the files and commit messages. This means that the commit messages and the files (structure and diffs shown) change correctly with the left slider, but does not work for the code content within the file diffs. Instead, file diffs are always compared to the original commit (revision 0) regardless of the value of the left slider.



× **Print Configurations**

Comment Options

(?)

☐ With File

Issue Options

☐ ● Open
☐ ✓ Resolved
☐ ⊗ Dropped
☐ Waiting for Verification

Revisions

Revision between: 0 - 4

Files:

- c++
  - ☑ .gitignore
  - ☑ CipherString.cc
  - ☑ Makefile
  - ☑ scrambler.cc
  - ☑ test_util.cc
  - ☑ util.h
  - ☑ util.cc
- python
  - cs5150
    - ☐ cipher_string.py

Commits

Jacqueline Woo	Add print usage even if argc is 0
Jacqueline Woo	Add (failing) tests for util
Jacqueline Woo	Add more comments for python and change util.cc
Jacqueline Woo	Add another addition

General Comments

jacquelinewoo 2 weeks ago

```
This is a comment.
```

File Attachment Comments

There are no file attachment comments for this review

File Diffs

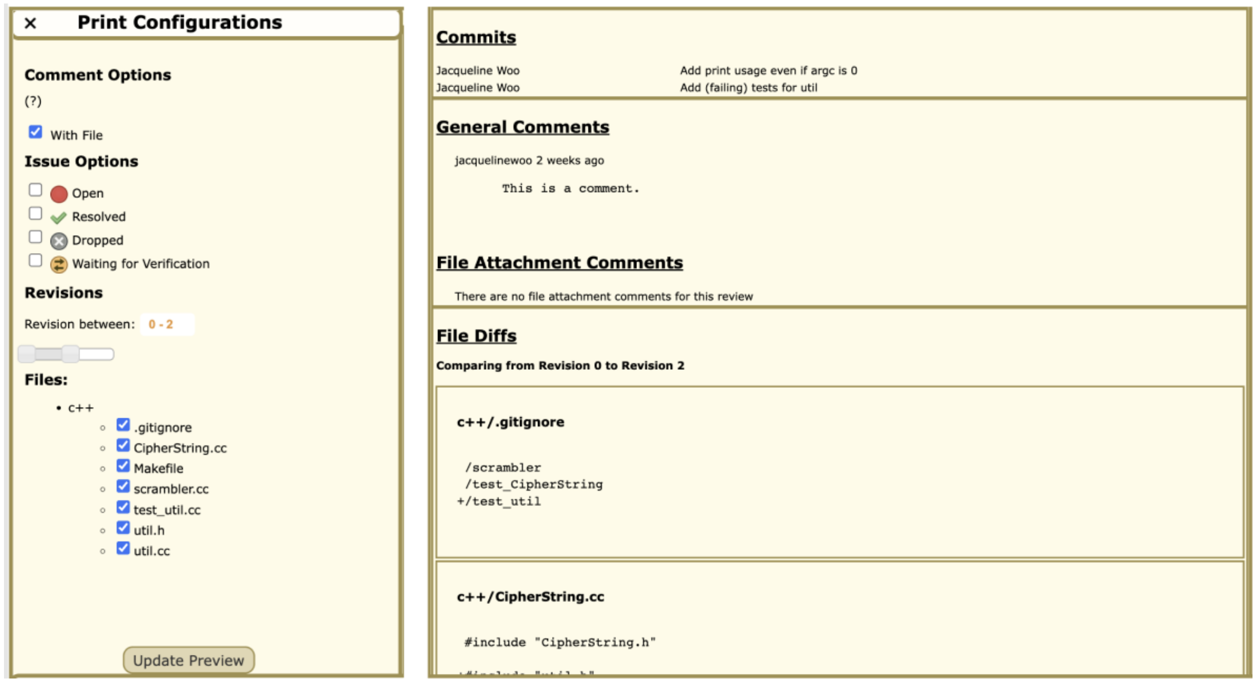
Comparing from Revision 0 to Revision 4

c++/.gitignore

```
 /scrambler
 /test_CipherString
+/test_util


 Comment from jacquelinewoo:
 (2022-05-06 16:34:16.053795+00:00) Comment for revison 4 on .gitignore file
```

Figure 4.1: Revision Slider. On the right side, the review has the two revisions it is comparing.



× **Print Configurations**

Comment Options

(?)

☑ With File

Issue Options

☐ ● Open
☐ ✓ Resolved
☐ ⊗ Dropped
☐ Waiting for Verification

Revisions

Revision between: 0 - 2

Files:

- c++
  - ☑ .gitignore
  - ☑ CipherString.cc
  - ☑ Makefile
  - ☑ scrambler.cc
  - ☑ test_util.cc
  - ☑ util.h
  - ☑ util.cc

Commits

Jacqueline Woo	Add print usage even if argc is 0
Jacqueline Woo	Add (failing) tests for util

General Comments

jacquelinewoo 2 weeks ago

```
This is a comment.
```

File Attachment Comments

There are no file attachment comments for this review

File Diffs

Comparing from Revision 0 to Revision 2

c++/.gitignore

```
 /scrambler
 /test_CipherString
+/test_util
```

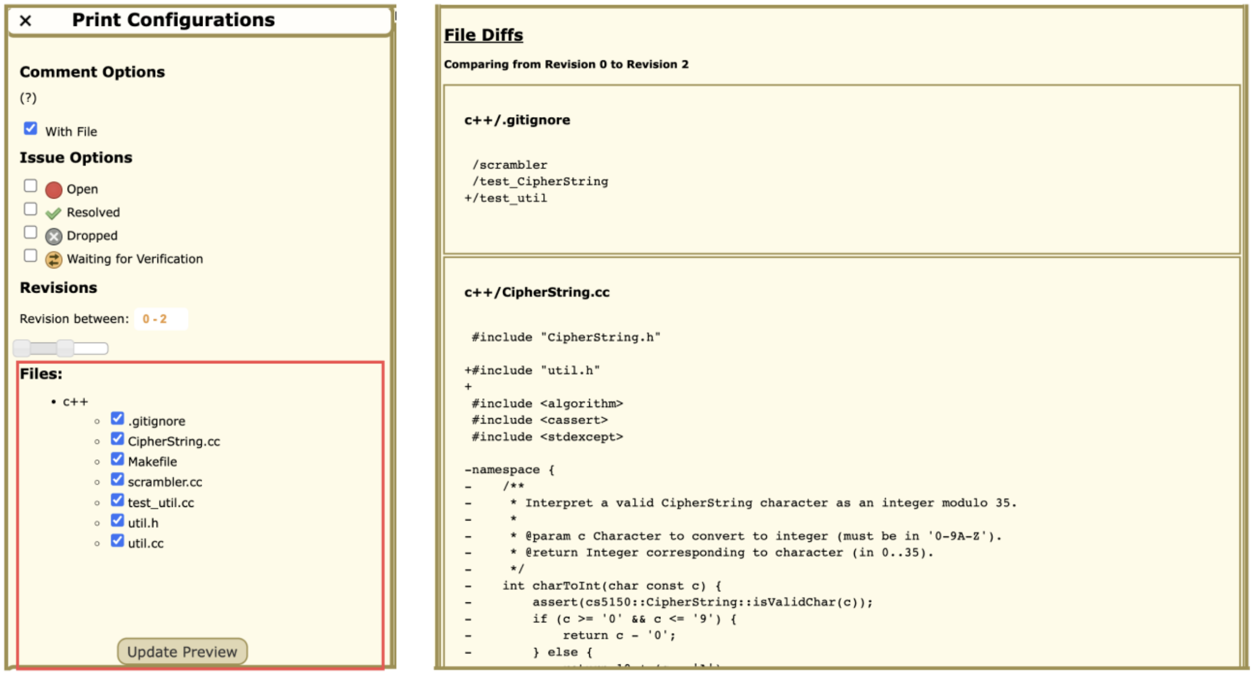
c++/CipherString.cc

```
 #include "CipherString.h"
```

Figure 4.2: When changing the revision slider, the file tree, commits, and diffs change.

File Tree

Finally, the user can select the files they want to include in the print review in the file selection tree, as shown in Figure 5.1. Files are organized in a tree structure based on the file path. By default, all of the files are selected. If files are deselected from the print configurations panel, the files are removed from the print preview, as shown in Figure 5.2. The state of being checked/unchecked persists after clicking the "Update Preview" button.



Comment Options

(?)

- [✓] With File

Issue Options

- [ ] ● Open
- [ ] ✓ Resolved
- [ ] Dropped
- [ ] Waiting for Verification

Revisions

Revision between: 0 - 2

Files:

- c++
  - [✓] .gitignore
  - [✓] CipherString.cc
  - [✓] Makefile
  - [✓] scrambler.cc
  - [✓] test_util.cc
  - [✓] util.h
  - [✓] util.cc

File Diffs

Comparing from Revision 0 to Revision 2

c++/.gitignore

```
 /scrambler
 /test_CipherString
+/test_util
```

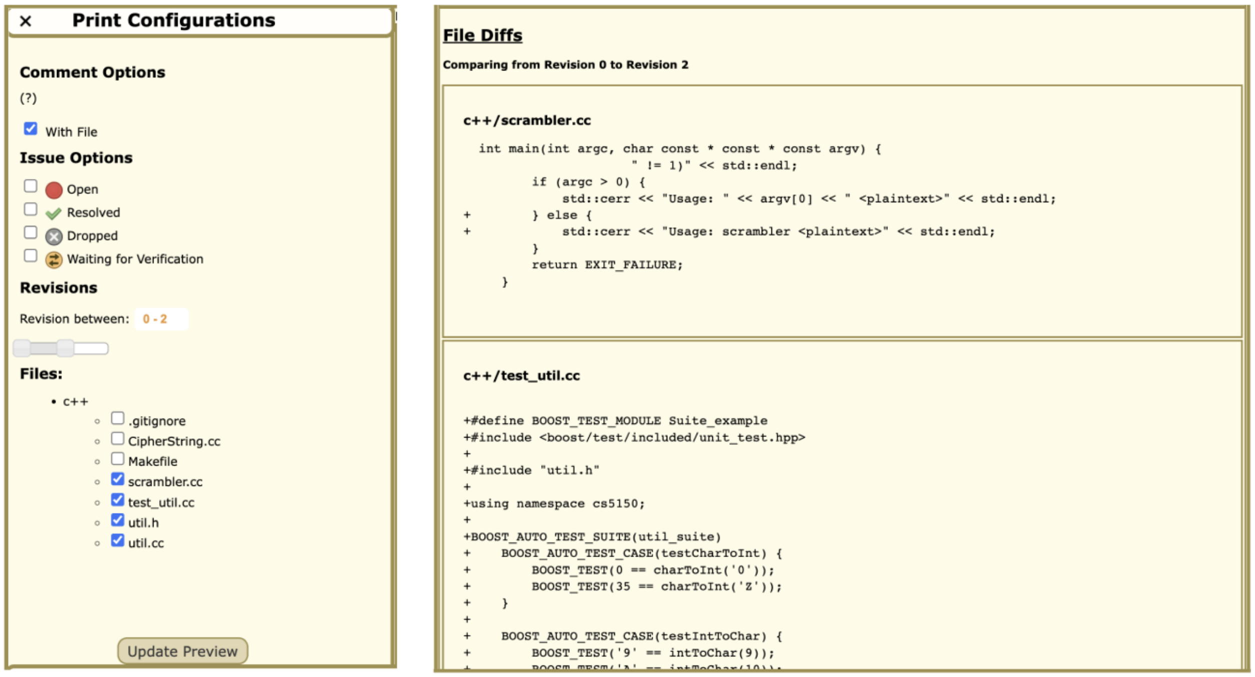
c++/CipherString.cc

```
 #include "CipherString.h"

+#include "util.h"
+
 #include <algorithm>
 #include <cassert>
 #include <stdexcept>

-namespace {
-    /**
-     * Interpret a valid CipherString character as an integer modulo 35.
-     *
-     * @param c Character to convert to integer (must be in '0-9A-Z').
-     * @return Integer corresponding to character (in 0..35).
-     */
-    int charToInt(char const c) {
-        assert(cs5150::CipherString::isValidChar(c));
-        if (c >= '0' && c <= '9') {
-            return c - '0';
-        } else {
```

Figure 5.1: File Tree and "Update Preview" Button. On the right side, the files selected are displayed.



Comment Options

(?)

- [✓] With File

Issue Options

- [ ] ● Open
- [ ] ✓ Resolved
- [ ] ⊗ Dropped
- [ ] Waiting for Verification

Revisions

Revision between: 0 - 2

Files:

- c++
  - [ ] .gitignore
  - [ ] CipherString.cc
  - [ ] Makefile
  - [✓] scrambler.cc
  - [✓] test_util.cc
  - [✓] util.h
  - [✓] util.cc

File Diffs

Comparing from Revision 0 to Revision 2

c++/scrambler.cc

```
  int main(int argc, char const * const * const argv) {
                      " != 1)" << std::endl;
         if (argc > 0) {
             std::cerr << "Usage: " << argv[0] << " <plaintext>" << std::endl;
+        } else {
+            std::cerr << "Usage: scrambler <plaintext>" << std::endl;
         }
         return EXIT_FAILURE;
     }
```

c++/test_util.cc

```
+#define BOOST_TEST_MODULE Suite_example
+#include <boost/test/included/unit_test.hpp>
+
+#include "util.h"
+
+using namespace cs5150;
+
+BOOST_AUTO_TEST_SUITE(util_suite)
+    BOOST_AUTO_TEST_CASE(testCharToInt) {
+        BOOST_TEST(0 == charToInt('0'));
+        BOOST_TEST(35 == charToInt('Z'));
+    }
+
+    BOOST_AUTO_TEST_CASE(testIntToChar) {
+        BOOST_TEST('9' == intToChar(9));
```

Figure 5.2: Deselecting Files. On the right side, the files are no longer displayed.

Maintainer Manual

The maintainer manual documents system design, test facilities, and deployment procedure.

Deployment Procedure

Pull from the main branch in Reviewboard B. In the virtual environment outlined in the project setup, run the script: `./contrib/internal/devserver.py`.

There are no external dependencies.

Requirements Analysis

The main objective of this project was to provide users of Review Board with the ability to export a printable copy of any code review. This additional feature to Review Board will provide printed code review to auditors who otherwise have systems that cannot link to the online code review tool. The printable copy will first display summary information relating to the project being reviewed, including descriptions, a list of files updated, and metadata. Each printable copy shall also contain the entire review history, including code snippets for each revision, reviewer comments and approvals, and, tentatively, the status of automated checks. This additional feature to Review Board shall allow users to configure several aspects of the output format, including which snippets, revisions, and files are relative. Furthermore, we'll allow users to configure how comments/reviews are printed. Users can either have comments/reviews attached to lines of code shown inline or displayed in a separate section of the printout. Given that some users may request very large snippets and/or revisions/files that would be expensive to render, our feature shall work to warn the user in such situations (before hanging the server or their browser). We detail each of our preliminary requirements below.

Preliminary Requirements

- **Summary Information**
    - Prints summary information displayed at the top of the review board project page
    - Prints description of project
    - Prints list of files updated by owner of project
    - Prints metadata
- **Code Snippets**
    - Prints code snippets that were changed/reviewed
- **Code Revisions**
    - Prints several revisions
    - By default prints latest revisions
    - Code snippets are printed for each revision
- **Reviewer Comments and Approvals**
    - Prints comments/reviews of code by reviewers
    - Prints comment/review approval status
- **Output Format Configuration**
    - Users choose which code snippets to print from the files that have changed
    - Users choose the format in which comments/reviews are displayed in printable output (inline vs. separate section)
    - Users choose which revisions to print
    - Users choose which modified files are printed out
        - Warn users if their requested print out is too large

Requirements Summary

In this section we discuss the status of the originally requested features, with sections for features that were successfully delivered, partially delivered, and missed.

Requirements Delivered

One of the main features originally requested by our client was to extend Review Board to provide a printable HTML copy of a review. We successfully created this copy, complete with summary information relating to the project being reviewed, including descriptions, commits (the full commit message, not just the first line), general comments and metadata. The printable copy contains the entire review history, including file diffs for each revision, reviewer comments and approvals. We also successfully incorporated a popup dialog to confirm to the user if they would like to download the preview. The dialog confirms the number of files in the preview page they are downloading and verifies if they would like to proceed.

Requirements Partially Complete or Revised

In our implementation, we were partially successful in implementing the user configurations and having the print preview be updated based on the configurations. On one hand, we were successful with allowing the user to select different types of issues for a review. If the review didn't have any issues, the selections are disabled. On the other hand, we were partially successful in the inline versus separate section comments. Code comments being displayed in a separate section was implemented, but not inline comments. In-file comments, where comments were placed underneath the diff of a file, were used in lieu of inline comments. We were also only partially successful in implementing the revision slider, so users could choose the revisions they want to print. Additionally, while we got the file tree and commits to dynamically update based on the revision, we could not do the same for file diffs. Currently, only file diffs compared to the original revision are displayed.

Requirements Missed

In our original user interface designs, we hoped to use the existing Review Board CSS styling in our own page. However, as mentioned earlier, we were unable to download the CSS along with the main HTML page. Therefore, we explained the situation to our client, and worked to formulate our own minimalistic styling that could meet his expectations. While our styling isn't as detailed as Review Board's styling, it is sufficient for reading and parsing information on the review.

Deployment Diagram

The figure below (Figure 6.1) shows our deployment diagram. From their personal computer, the user sends a HTTP connection request to access the web server of the review board. The web server of Review Board uses the Django web framework and employs the ORM protocol to query the SQLite database.

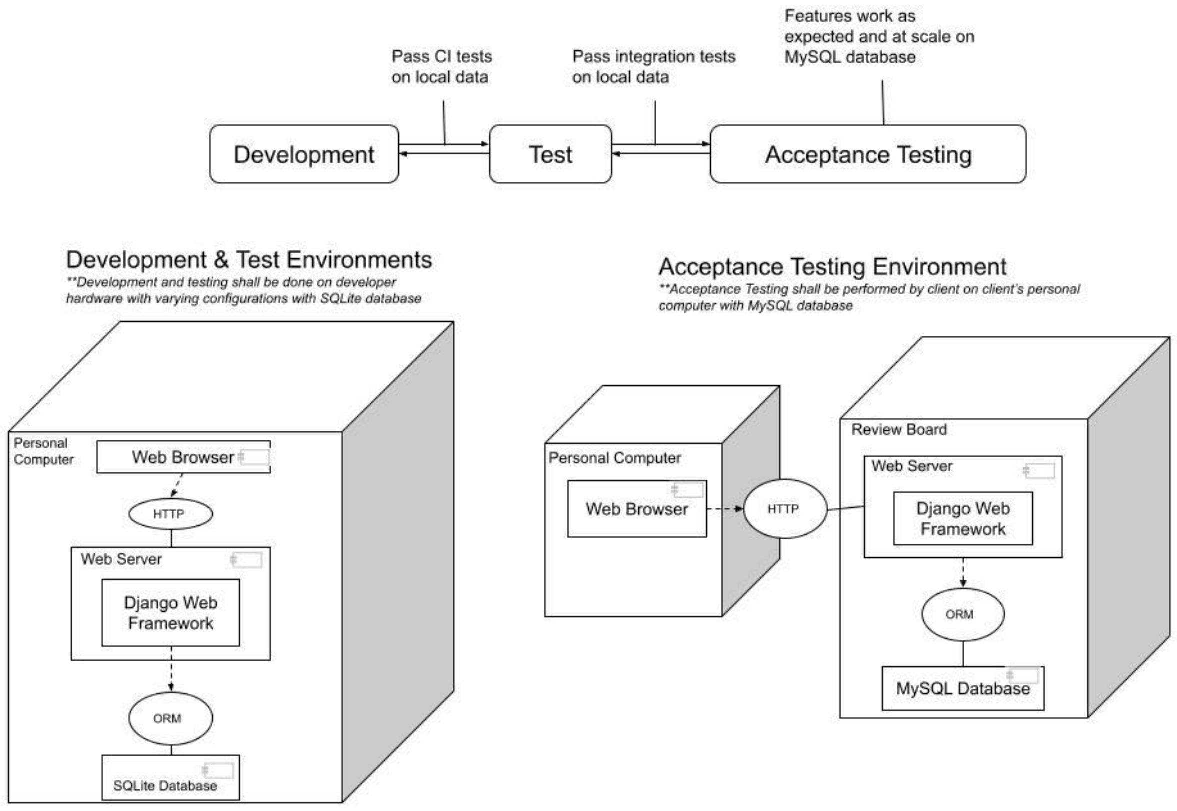

Features work as expected and at scale on MySQL database
Pass CI tests on local data
Pass integration tests on local data
Development
Test
Acceptance Testing
Development & Test Environments
**Development and testing shall be done on developer hardware with varying configurations with SQLite database
Acceptance Testing Environment
**Acceptance Testing shall be performed by client on client's personal computer with MySQL database
Personal Computer
Web Browser
HTTP
Web Server
Django Web Framework
ORM
SQLite Database
Review Board
MySQL Database

Figure 6.1: Deployment Diagram

Component Diagram

The component diagram in Figure 6.2 shows the components that we added (highlighted in red) and how it interfaces with the existing Review Board code. We created a new HTML template for our page and added a new view that inherits from Review Diff Viewer View and Review Request Detail View.

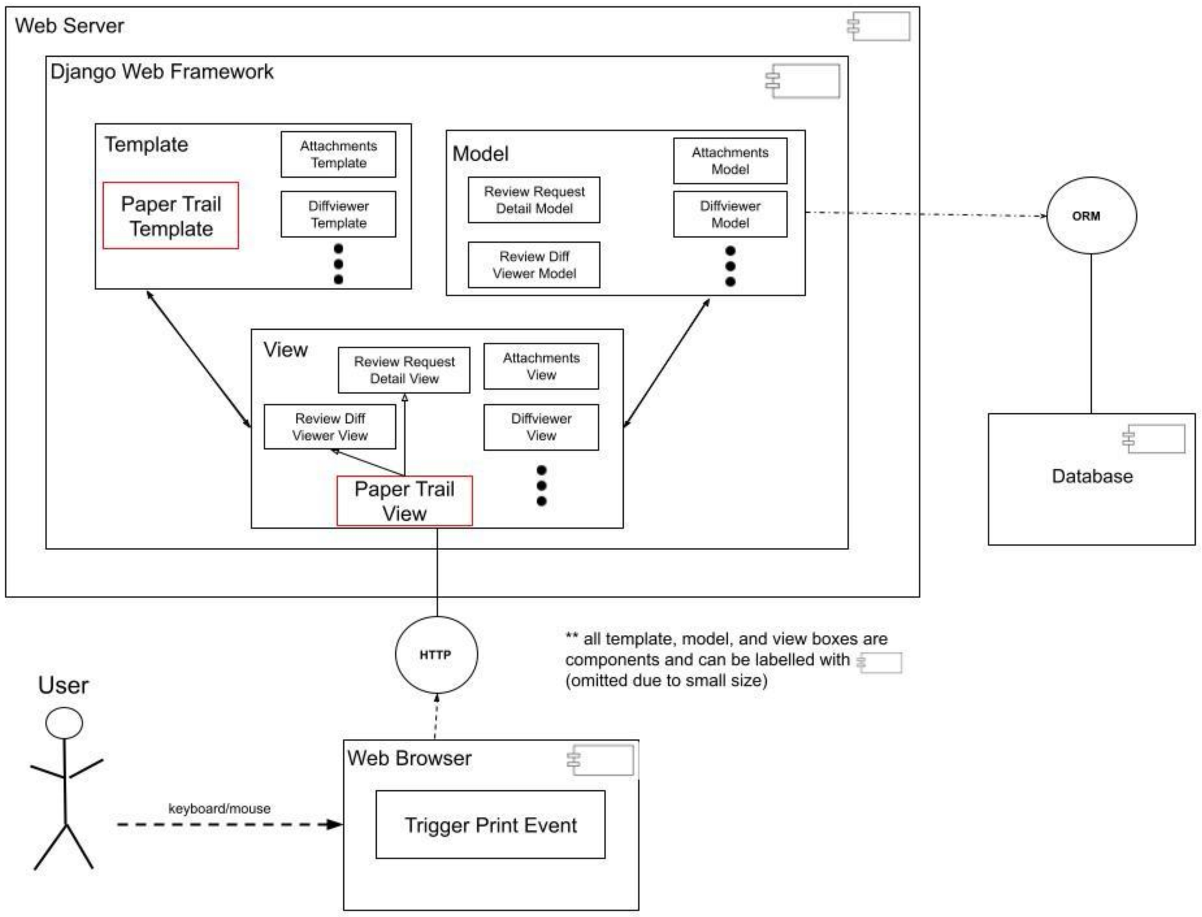
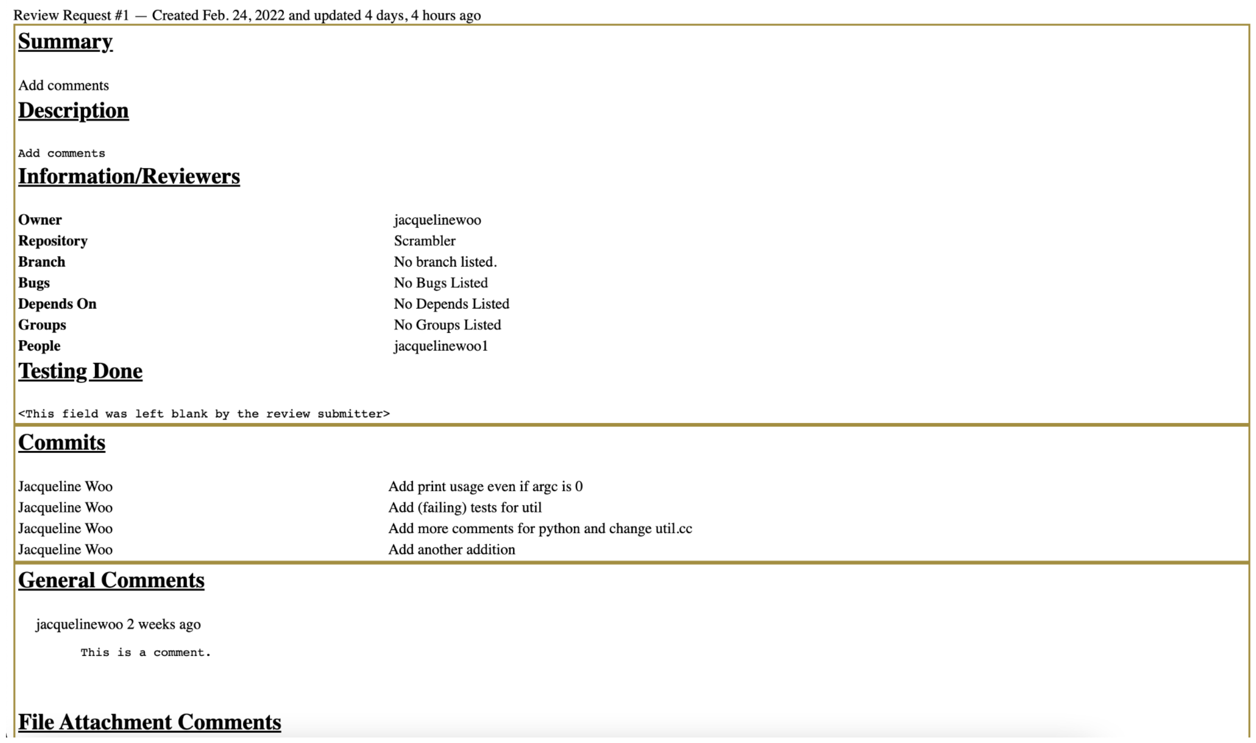


Figure 6.2: Component Diagram

Final User Interface Designs

The final user interface design mockup is split into two panels, a left side panel that has the print configurations and the right side panel that has the print preview (Figure 1.2). The comments and issues section of the print configuration

The panel on the right includes everything that will be printed: Summary, Commits, General Comments, and Diffs. The left panel contains sliders that allow users to select different versions, a selection bar to check the questions users want, and a file selection display. The “Update Review” button at the bottom can update the content of the right panel according to the user's selection. There is a prominent print button at the very bottom of the page that prints the right panel (Figure 1.2). Once the print button is pressed, a confirmation dialog prompts the user to verify if they would like to continue to print the HTML file (Figure 1.3). The HTML file printout will look similar to how it was displayed in the right print preview panel (Figure 6.3).



Review Request #1 — Created Feb. 24, 2022 and updated 4 days, 4 hours ago

Summary

Add comments

Description

```
Add comments
```

Information/Reviewers

Owner	jacquelinewoo
Repository	Scrambler
Branch	No branch listed.
Bugs	No Bugs Listed
Depends On	No Depends Listed
Groups	No Groups Listed
People	jacquelinewoo1

Testing Done

```
<This field was left blank by the review submitter>
```

Commits

Jacqueline Woo	Add print usage even if argc is 0
Jacqueline Woo	Add (failing) tests for util
Jacqueline Woo	Add more comments for python and change util.cc
Jacqueline Woo	Add another addition

General Comments

jacquelinewoo 2 weeks ago

```
This is a comment.
```

File Attachment Comments

Figure 6.3: A sample of a printable HTML page.

Test Plan and Results

We wrote unit tests for the tree parsing structure on the left panel (in test_treenode.py) and unit tests for the new print view (in test_review_request_print_view.py). These unit tests can be run using the existing test command for reviewboard: `./tests/runtests.py` in the /rb/reviewboard folder. Alternatively you can run just the tests we wrote by running: `./tests/runtests.py reviewboard.print`. In terms of what is tested in each of the two, the treenode tests validate the n-ary tree structure is constructed correctly given the file names and confirms that the generated html output is what is expected; the test for the print view verifies that the get requests for the print page return a 200 (success) status code, the flags for the HTML are as expected, and the issue content is as expected. Much of the functionality for the print view (template in MVC) comes from inheriting the views (ReviewRequestDetailView and ReviewsDiffViewerView), these two views have the corresponding unit tests in (`test_review_request_detail_view.py` and `test_reviews_diff_viewer_view.py`).

On manual testing:

We have done extensive manual testing on the print configurations page. That is, on the print configurations page, there are checkboxes for issues, in-code comments, and individual files; we have tested all combinations of the checkboxes such that we have 100% conditional coverage for the Jinja template logic. In addition, we have tested edge cases such as what happens when there are no issues/comments in a review and what happens when no files on the left panel are selected. (It is not possible to make a review without changing a file so we do not need to test for what happens when there is no diffset). We also manually tested the buttons on the print configuration page: the print button, the back button, and the update button. The buttons were manually tested by clicking the buttons and then verifying the result was as expected (e.g. the print popup shows up, the review page is loaded, and the print preview updates respectively).

Signed License Agreement

We agree that our joint work can be published under the host application's open source license.

Robel Ayalew

[illegible]

Yunjiao Chai

[illegible]

Berk Erdal

Berk Erdal

Gary Ho

Gary Ho

Leo Leyva

L. Leyva

Banpreet Singh

[illegible]

Jackie Woo

[illegible]