

**Dr. B R AMBEDKAR NATIONAL INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
JUNE-DECEMBER 2016**

LOCAL COMPILER OF NITJ

**Mentor :
Dr. Rajneesh Rani**

**Co-mentors :
Ms Kirti
Ms Amandeep Kaur**

**Saurbhi Agarwal
13103026**

**Aman Garg
13103050**

**Jyoti Kumar
13103064**

**Shivanshi
13103065**

Outline

Objective

Technology – FrontEnd

Technology – BackEnd

Current Status

Flow Chart of Project

Working of Project

References

Objective

To make a Local Compiler for languages like C,C++,Java and Python hosted on the [Cyberoam Server](#) so that we can access, compile and debug the code anywhere in the college.

Technology - FrontEnd

- ❖ Using twitter Bootstrap, jQuery, jQueryUI and custom CSS wherever necessary.
- ❖ Editor support is mirrored by CodeMirror UI library.
- ❖ Shell UI support is provided by GateOne
- ❖ Tabbed editor support using custom CSS

Technology - BackEnd

- ❖ Chroot Jail environment running on a Linux Server System with support for Django and encrypted users .
- ❖ Paramiko library for providing SSH and SFTP support for the underlying network.
- ❖ Python scripts that compile and test the execution.

Server Side Implementation

```
new1@aman: ~  
aman@aman ~ $ echo && tail /etc/ssh/sshd_config &&echo
```

```
Match group sshusers  
    ChrootDirectory /var/jail/  
    X11Forwarding no  
    AllowTcpForwarding no
```

```
Match User new1  
    ChrootDirectory /chroot/Ubuntu
```

```
Match User new2  
    ChrootDirectory /chroot/Ubuntu
```

```
aman@aman ~ $ sudo chroot /chroot/Ubuntu/
```

```
root@aman:/# cat /etc/issue  
Ubuntu 12.04 LTS \n \l
```

```
root@aman:/# echo && ls /home/
```

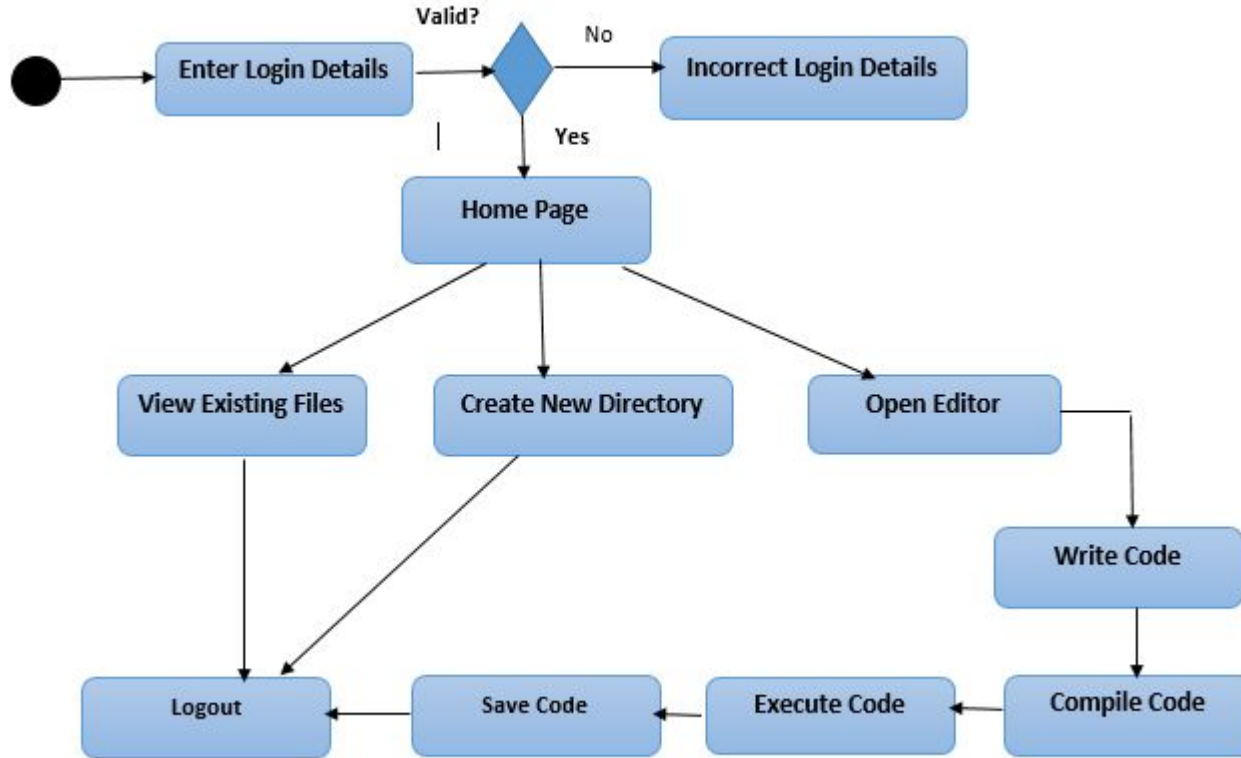
```
new1 new2
```

```
root@aman:/#
```

Current Status

- ❖ Successfully created a Chroot jail server running Ubuntu 12.04 inside the host machine running Linux Mint 17.3
- ❖ Created users inside jail with SSH support to the outers.
- ❖ Fully working editor with a tree view updated with the latest directory through asynchronous (AJAX) files
- ❖ Error testing of various user scenarios is ongoing.
- ❖ Timeouts are set and access to memory, network sockets is restricted

Flow Chart of Project



User Portal Registration

Register Page for New User

User Registration

Username:

Email address:

Password:

Password (again):

User Portal Login

Login Page for the existing User

Login

Username

Password

Login

Register

Workspace For Existing User

Aman Garg

Workspaces

Workspaces

Create a new workspace

nitj_cs_5th_13103050

cpp workspace

Aman Garg 13103050 NITJ

Updated a few seconds ago.

Clone Open

1 CPU 512MB RAM 1GB HDD

Editor

NITJ COMPILER

Home

Editor

new1 ▾

✓

test

aaa.py

aprinte.py

fileListe.py

hello.cpp

Hello.java

hii.c

pp.pyc

File name

C ▾

✖

3024-night ▾

Browse File

New Tab

1 /*

2 Your C code goes here

3 Main method should return 0

4 */

EDITOR PAGE

Code Output

Your code output appears here.

Click the resize icon in the top right to make this fullscreen.

Execute

Enter custom input

Working of Editor

The screenshot shows the NITJ Compiler Editor interface. At the top, there's a navigation bar with 'NITJ COMPILER', 'Home', and 'Editor' tabs, and a 'new1' button. On the left, a file explorer shows a 'test' folder containing 'aaa.py', 'aprinte.py', 'fileListe.py', and 'pp.pyc'. The main editor area has a 'File name' input field, a language dropdown menu (circled in red), a '3024-night' theme selector, and a 'Browse File' button. The language dropdown is open, showing options: C (selected), C++, Java, Python, and Text. A red arrow points from a text box to the dropdown. The text box says: 'Choose the language in which you want to compile and execute the code.' The editor area contains a 'New Tab' button and a code editor with the following text:

```
1 /*  
2 Your C code goes here  
3 Main method should return 0  
4 */
```

 At the bottom, there are 'Execute' and 'Enter custom input' buttons. On the right, a 'Code Output' panel shows the message: 'Your code output appears here.' and instructions: 'Click the resize icon in the top right to make this fullscreen.'

NITJ COMPILER Home Editor new1

File name 3024-night Browse File

New Tab

test
aaa.py
aprinte.py
fileListe.py
pp.pyc

C
C
C++
Java
Python
Text

Choose the language in which you want to compile and execute the code.

1 /*
2 Your C code goes here
3 Main method should return 0
4 */

Code Output

Your code output appears here.

Click the resize icon in the top right to make this fullscreen.

Execute Enter custom input

Working of Editor

The screenshot displays the NITJ Compiler Editor interface. At the top, there is a navigation bar with 'NITJ COMPILER', 'Home', and 'Editor' tabs, and a 'new1' button on the right. On the left, a file explorer shows a 'test' directory containing 'aaa.py', 'aprinte.py', 'fileListe.py', and 'pp.pyc'. The main editor area has a 'File name' input field, a language dropdown set to 'C', and a 'Browse File' button. A dropdown menu for themes is open, showing options like '3024-night', 'default-theme', '3024-day', 'abcdef', 'ambiance', 'base16-dark', 'base16-light', 'bespin', 'blackboard', 'cobalt', 'colorforth', 'dracula', 'duotone-dark', 'duotone-light', 'eclipse', 'elegant', 'erlang-dark', 'hopscotch', 'icecoder', and 'isotope'. The '3024-night' theme is highlighted. A red box with the text 'select the theme of the editor page' and a red arrow points to this dropdown menu. The editor area contains a code snippet:

```
1 /*
2 Your C code goes here
3 Main method should return 0
4 */
```

. At the bottom, there are buttons for 'Execute' and 'Enter custom input'. On the right, a 'Code Output' panel shows the text 'Your code output appears here.' and a note: 'Click the resize icon in the top right to make this fullscreen.'

NITJ COMPILER Home Editor new1

test
aaa.py
aprinte.py
fileListe.py
pp.pyc

File name C Browse File

New Tab

1 /*
2 Your C code goes here
3 Main method should return 0
4 */

select the theme of the editor page

3024-night
default-theme
3024-day
3024-night
abcdef
ambiance
base16-dark
base16-light
bespin
blackboard
cobalt
colorforth
dracula
duotone-dark
duotone-light
eclipse
elegant
erlang-dark
hopscotch
icecoder
isotope

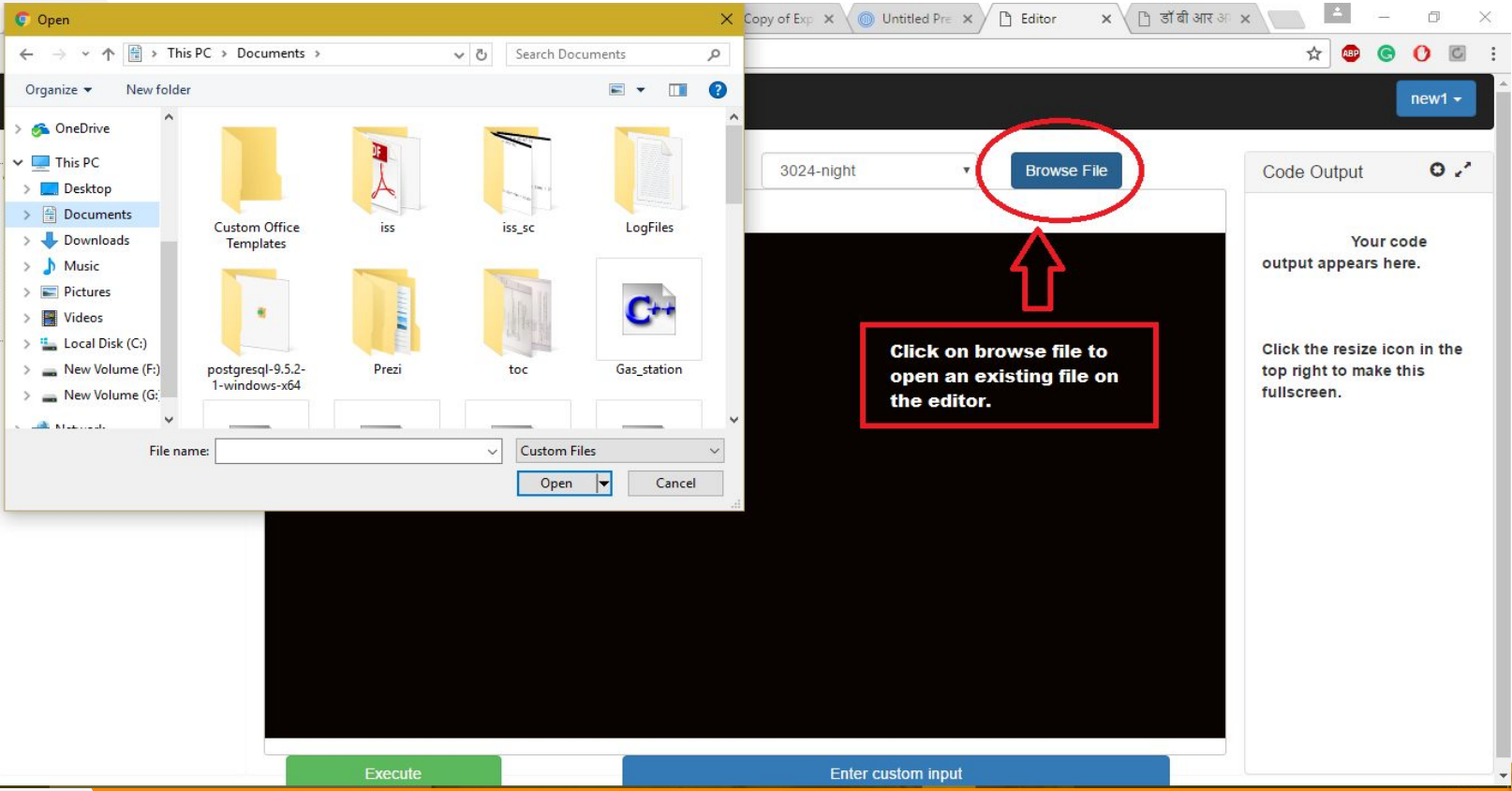
Code Output

Your code output appears here.

Click the resize icon in the top right to make this fullscreen.

Execute Enter custom input

Working of Editor



Working of Editor

NITJ COMPILER

Home

Editor

new1 ▾

File name

C ▾

3024-night ▾

Browse File

New Tab

1 /*

2 Your C code goes here

3 Main method should return 0

4 */

Code Output

Your code output appears here.

Click the resize icon in the top right to make this fullscreen.

Specify stdin input here

Enter custom input

test

test

hjh

jkj

aa.cpp

sub

breco.cpp

print.py

xvvv.c

aaa.py

aprinte.py

fileListe.py

pp.pyc

Tree like structure of the workspace of the user.

Server Side Implementation

```
new1@aman:~$ pwd
/home/new1
new1@aman:~$
new1@aman:~$
new1@aman:~$ ls -R
.:
test
./test:
print.py test xvvv.c
./test/test:
hjh jkj sub
./test/test/hjh:
./test/test/jkj:
aa.cpp
./test/test/sub:
breco.cpp
new1@aman:~$
```

**User Workspace at
Server.**

Working of Editor

NITJ COMPILERHomeEditornew1

+ Add folder

+ Add file

Remove

Rename

1 /*

2 Your C code goes here

3 Main method should return 0

4

File name

C

3024-night

Browse File

New Tab

To add a directory,
add a file,remove or
rename a file/directory to
your workspace.

Code Output

Your code
output appears here.

Click the resize icon in the
top right to make this
fullscreen.

Execute

Enter custom input

Working of Editor

The screenshot displays the NITJ Compiler web interface. At the top, a dark navigation bar contains the text "NITJ COMPILER" and "Home Editor", along with a "new1" button. Below this, a toolbar includes a file icon, a "File name" input field, a language dropdown set to "C", a red "X" icon, a "3024-night" dropdown, and a "Browse File" button. On the left, a file explorer shows a directory structure with folders "test" and "sub", and various files including "aa.cpp", "breco.cpp", "print.py", "xvvv.c", "aaa.py", "aprinte.py", "fileListe.py", and "pp.pyc". The "sub" folder is selected. The main editor area has two tabs, "New Tab" and "New tab", with the first tab active. It contains a C program with the following code:

```
1 /*
2  Your C code goes here
3  Main method should return 0
4  */
5 #include<stdio.h>
6
7 int main()
8 {
9     printf("hello");
10     return 0;
11 }
```

A red-bordered box highlights the text: "Program compiled and executed successfully in C language." Below the editor are two buttons: "Execute" (green) and "Enter custom input" (blue). On the right, a "Code Output" panel shows the results: "Compilation : success", "Runtime : success", and the output "hello".

Working of Editor


The screenshot displays the NITJ Compiler Editor interface. On the left, a file explorer shows a 'test' directory containing files like 'aaa.py', 'aprint.py', 'fileList.py', 'hello.c', and 'pp.pyc'. The main editor area shows a C program with the following code:

```
1 /*
2  Your C code goes here
3  Main method should return 0
4  */
5 #include<stdio.h>
6
7 int main()
8 {
9     printf("hello");
10     return 0;
11 }
```

At the top of the editor, there are tabs for 'hello', 'C', and '3024-night', along with a 'Browse' button. A red arrow points from a red-bordered box containing the text 'Program saved succesfully in workspace.' to a green checkmark icon in a toast notification that says 'Saved file successfully a...'. Below the editor, there are buttons for 'Execute' and 'Enter custom input'. On the right side, a panel shows the output of the program, which is 'hello', and status messages: 'Compilation : success' and 'Runtime : success'.

Server Side Implementation

```
new1@aman: ~  
new1@aman:~$ for i in `ls`;do printf $i"\n\n";done;  
aaa.py  
  
aprinte.py  
  
fileLister.py  
hello.c  
pp.py  
  
test  
  
new1@aman:~$
```



File saved at Server.

Working of Editor

The screenshot displays the NITJ Compiler web interface. At the top, a dark navigation bar contains the text "NITJ COMPILER", "Home", "Editor", and a "new1" dropdown menu. Below this, the interface is divided into several sections:

- File Explorer (Left):** Shows a directory structure with a "test" folder containing files "aaa.py", "aprinte.py", "fileListe.py", "hii.c" (highlighted in blue), and "pp.pyc".
- Editor (Center):** Features a "File name" input field, a "C++" language selector, a "3024-night" theme dropdown, and a "Browse File" button. Below these are two tabs labeled "New Tab" and "New tab". The main editor area has a black background with white text showing a C++ program:

```
1 /*
2  Your C++ code goes here
3  Main method should return 0
4  */
5 #include<bits/stdc++.h>
6 using namespace std;
7
8 int main()
9 {
10     cout<<"hello";
11     return 0;
12 }
```

A red rectangular box is overlaid on the editor with the text: **Program compiled and executed successfully in C++ language**.
- Code Output (Right):** Displays the results of the compilation and execution:

```
Compilation : success
Runtime : success

hello
```
- Execution Controls (Bottom):** Includes a green "Execute" button and a blue "Enter custom input" button.

Working of Editor

The screenshot displays the NITJ Compiler web application. The top navigation bar includes 'NITJ COMPILER', 'Home', 'Editor', and a 'new1' dropdown. The left sidebar shows a file explorer with a 'test' directory containing files like 'aaa.py', 'aprinte.py', 'fileLister.py', 'hello.cpp', 'Hello.java', 'hii.c', and 'pp.pyc'. The main editor area has a tab labeled 'Hello' and a language dropdown set to 'Java'. The code editor contains the following Java code:

```
1 /*
2  Your Java code goes here
3  Name of class should be kept as main and public
4  If using a custom name to save the file, change the name of a class to the name of the program
5  */
6  public class Hello
7  {
8      public static void main(String[] args)
9      {
10         System.out.println("hello");
11     }
12 }
```

A red-bordered box highlights the text: **Program compiled and executed successfully in Java language.**

The right sidebar, titled 'Code Output', shows the results: 'Compilation : success' and 'Runtime : success', followed by the output 'hello'. At the bottom, there are buttons for 'Execute' and 'Enter custom input'.

Working of Editor

The screenshot displays the NITJ COMPILER web application interface. At the top, the header includes 'NITJ COMPILER', 'Home', 'Editor', and a 'new1' dropdown menu. Below the header, the interface is divided into several sections:

- File Explorer (Left):** A sidebar showing a directory structure with a 'test' folder containing files: 'aaa.py', 'aprinte.py', 'fileLister.py', 'hello.cpp', 'Hello.java', 'hii.c', and 'pp.pyc'.
- Editor (Center):** A code editor with two tabs labeled 'New Tab' and 'New tab'. The active tab shows Python code:

```
1 #Your Python code goes here
2 print "hello"
```

A red-bordered box in the center of the editor contains the text: **Program executed successfully in Python.**
- Code Output (Right):** A panel titled 'Code Output' showing the results of the execution:

```
Compilation : success
Runtime : success

hello
```
- Buttons (Bottom):** A green 'Execute' button and a blue 'Enter custom input' button are located at the bottom of the interface.

Working of Editor

The screenshot shows a web browser window at the address `10.10.129.226:8000/editor/`. The interface includes a file explorer on the left with files `aaa.py`, `aprinte.py`, `fileListe.py`, and `pp.pyc`. The main editor area has a dark background with the following code:

```
1 /*  
2 Your C code goes here  
3 Main method should return 0  
4 */
```

Annotations on the image include:

- A red box with the text "Enter stdin input here." and a red arrow pointing to the "Specify stdin input here" text box.
- A red circle around the "Enter custom input" button.

On the right side, there is a panel for output with the text "Your code output appears here." and a note: "Click the resize icon in the top right to make this fullscreen."

Working of Editor

The screenshot displays the NITJ Compiler Editor interface. At the top, there is a dark header bar with the text "NITJ COMPILER" on the left, "Home" and "Editor" in the center, and a "new1" button on the right. Below the header, the interface is divided into several sections. On the left, a file explorer shows a directory structure with a folder named "test" containing files "aaa.py", "apn.py", "file", and "pp.py". The file "apn.py" is selected, and a context menu is open over it, showing options: "Open", "Remove", and "Rename". A red circle highlights this menu, and a red arrow points from a text box to it. The text box, which has a red border, contains the text: "To open , remove or rename a file." In the center, there is a code editor with a dark background. It has a tab labeled "New Tab" and a "File name" input field. The code editor contains the following text: "1 /*", "2 Your C code goes here", "3 Main method should return 0", and "4 /*". To the right of the code editor is a "Code Output" panel with a plus icon and a fullscreen icon. It contains the text: "Your code output appears here." and "Click the resize icon in the top right to make this fullscreen." At the bottom, there is a light blue bar with the text "Specify stdin input here" and a button labeled "Enter custom input".

NITJ COMPILER Home Editor new1

File name C 3024-night Browse File

New Tab

1 /*
2 Your C code goes here
3 Main method should return 0
4 /*

test
aaa.py
apn.py
file
pp.py

Open
Remove
Rename

To open , remove or rename a file.

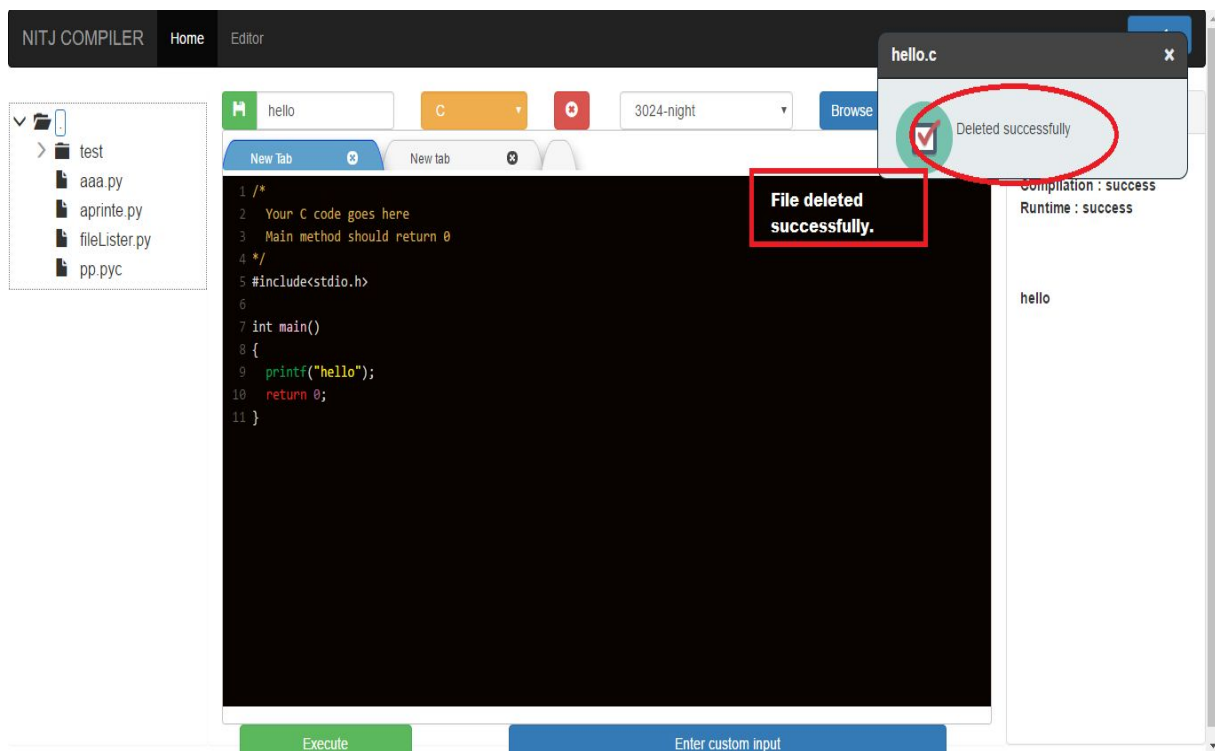
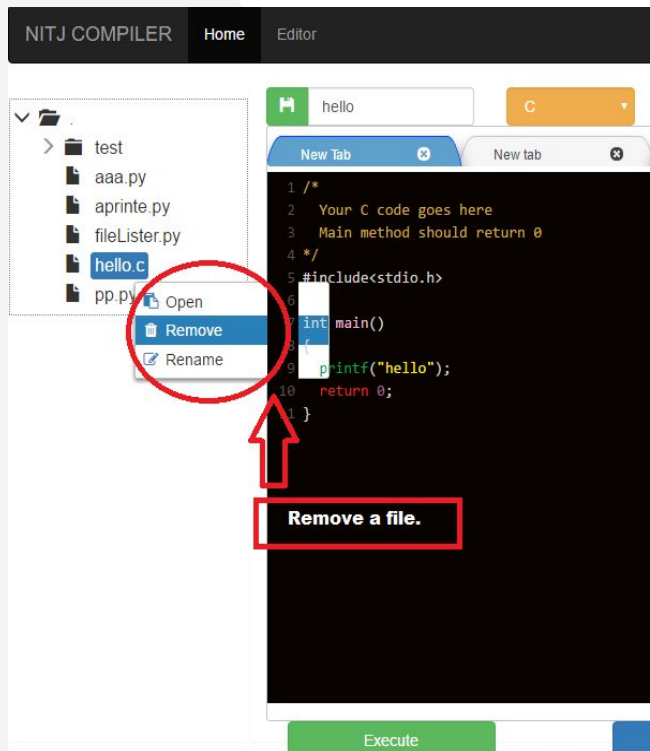
Code Output

Your code output appears here.

Click the resize icon in the top right to make this fullscreen.

Specify stdin input here Enter custom input

Working of Editor



Working of Editor

NITJ COMPILER Home Editor

test
aaa.py
aprinte.py
fileListe.py
hello.c
pp.pyc

Open
Remove
Rename

Rename a file.

hello

C

New Tab New tab

```
1 /*
2  Your C code goes here
3  Main method should return 0
4  */
5 #include<stdio.h>
6
7 int main()
8 {
9     printf("hello");
10     return 0;
11 }
```

Execute

NITJ COMPILER Home Editor

test
aaa.py
aprinte.py
fileListe.py
hii.c
pp.pyc

File renamed.

hello

C

New Tab New tab

```
1 /*
2  Your C code goes here
3  Main method should return 0
4  */
5 #include<stdio.h>
6
7 int main()
8 {
9     printf("hello");
10     return 0;
11 }
```

Execute

Server Side Implementation

```
new1@aman: ~  
new1@aman:~$ for i in `ls`;do printf $i"\n\n";done;  
aaa.py  
aprinte.py  
fileLister.py  
hi.c  
pp.pyc  
test  
new1@aman:~$
```

File renamed at Server.

References

- ❖ binarytides.com
- ❖ coldattic.info/blogs
- ❖ github.com/paramiko
- ❖ jessenoller.com/blog
- ❖ djangoproject.com



THANK YOU !