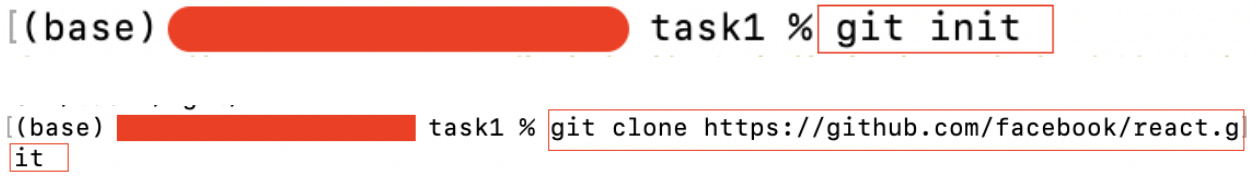
SE1_TASK1

1.List all components in React codebase

(1)Use git to pull the source code of React v18.3.1

code:

```
git init
git clone https://github.com/facebook/react.git
```



(2)Follow the rules introduced in the task instructions, we need to search each .js file in the source code and use regrex to find the patterns that identify the components.

The regrex expression:

(a)class-based components:

```
class\s+(\w+)\s+extends\s+React\.(Component|PureComponent)
```

(b)functional components:

```
return\s+<.*?>
```

(3)Use python with the logic in(2) to automatically search all components in the react source code

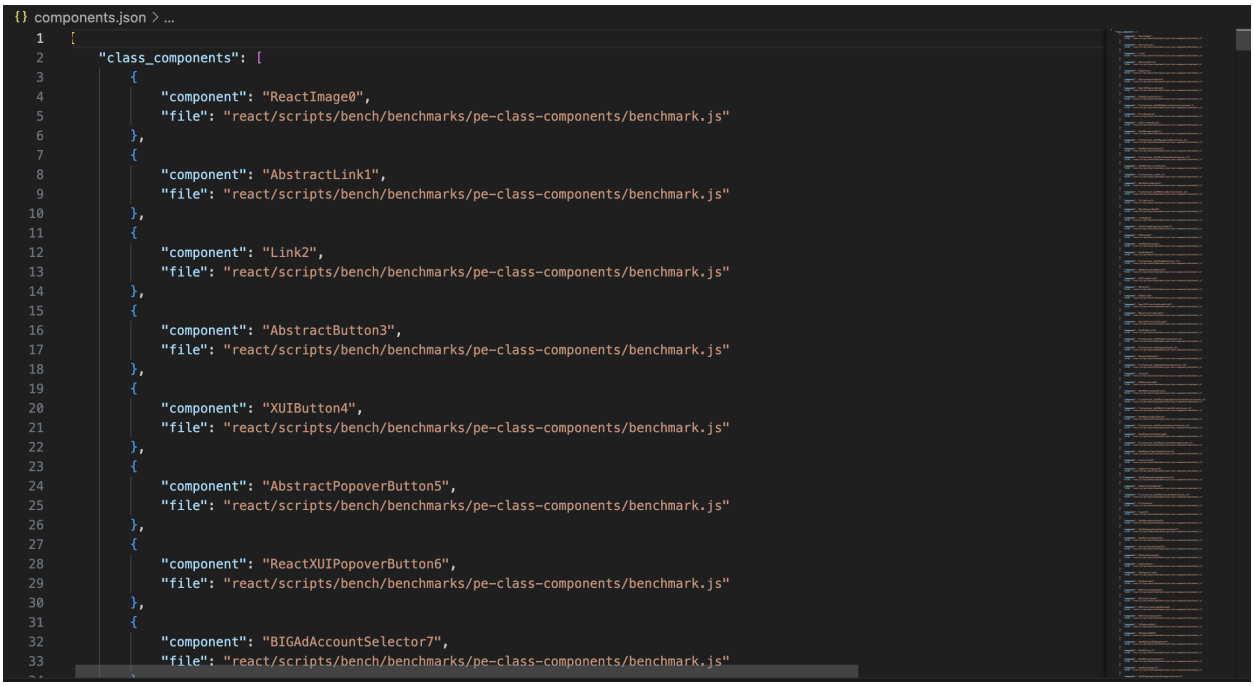
the code:

search_components.py

```
import os
import re
import json

def list_components_in_react_codebase(directory):
    # Define regex patterns for components
    class_based_pattern = r'class\s+(\w+)\s+extends\s+React\.(Co
    function_based_pattern = r'return\s+<.*?>'

    # Initialize results
    components = {
        "class_components": [],
        "function_components": []
    }

    # Walk through all files in the directory
    for root, _, files in os.walk(directory):
        for file in files:
            if file.endswith('.js'):
                file_path = os.path.join(root, file)
                with open(file_path, 'r', encoding='utf-8') as f
                    content = f.read()

                    # Match class-based components
                    class_matches = re.findall(class_based_patte
                    for match in class_matches:
                        components["class_components"].append({
                            "component": match[0],
                            "file": file_path
                        })

                    # Match function-based components
                    if re.search(function_based_pattern, content
                        components["function_components"].append
                            "file": file_path
```

```
                            })

    # Return components found
    return components

# Define the directory to scan
react_codebase_directory = "react"

# List components
components_found = list_components_in_react_codebase(react_codeb

# Save results to a JSON file
output_file = "components.json"
with open(output_file, 'w', encoding='utf-8') as json_file:
    json.dump(components_found, json_file, indent=4)

print(f"Components listed successfully! Results saved in {output
```

the output is the **components.json** which lists all the components:

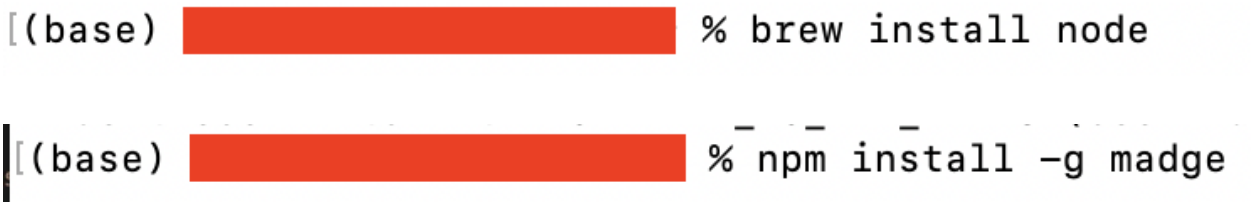


2.Detect dependencies between React files

(1)Download Madge

```
brew install node
```

```
npm install -g madge
```



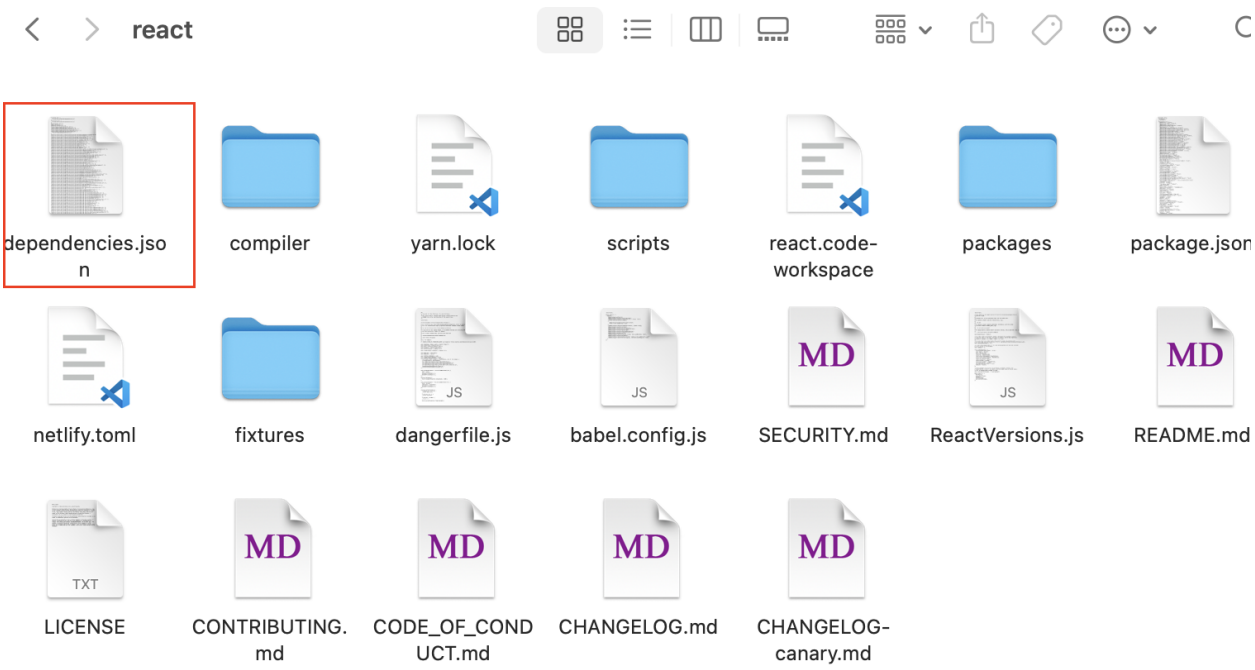
(2)Use Madge to analyze the dependencies between react files

In the react file, use the command below to generate the dependency json file.

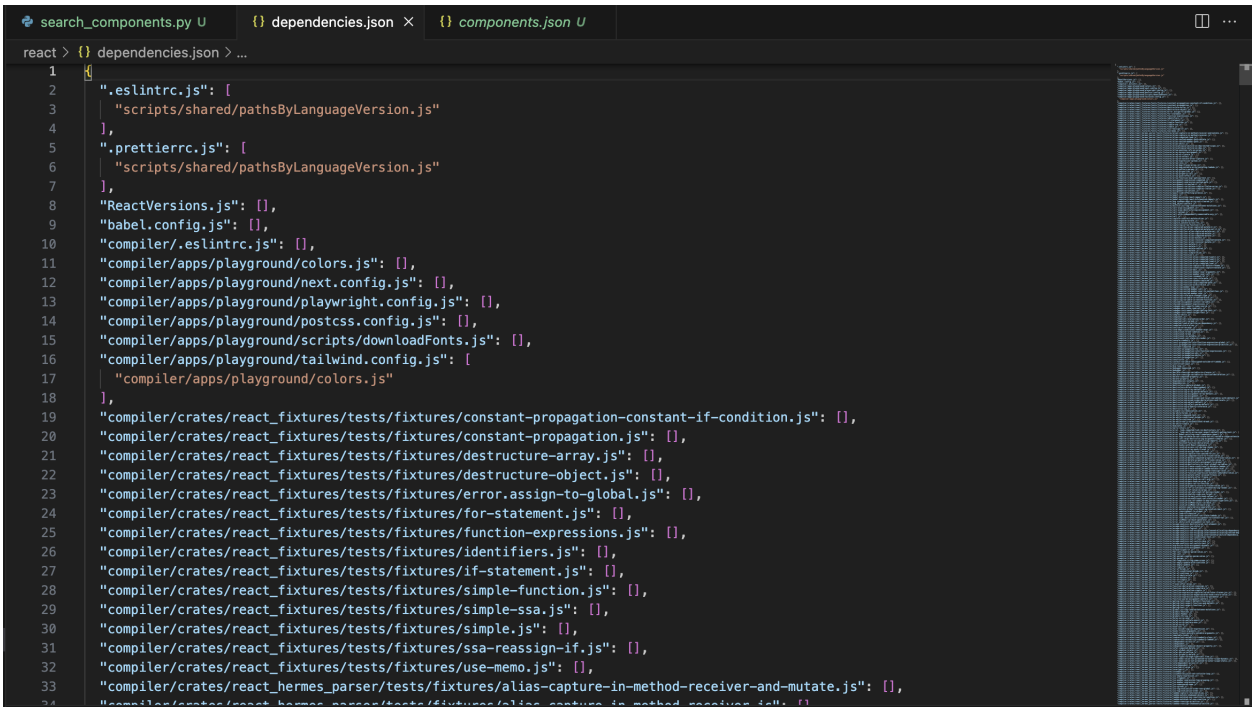
```
madge . --json > dependencies.json
```

```
(base) [redacted] ~ % cd [redacted]
[redacted]/react
(base) adriasong@songyuyun react % madge . --json > dependencies.json
```

The file dependencies.json is generated:



The content in **dependencies.json**:



(3)Identify the Top 3 Files with the Most Dependencies

With the dependencies.json generated in (2), we use python to do the dependency file calculation and document the top 3 files with the highest number of dependencies.

find_top3_highdep_files.py

```
import json

# Load the JSON file
with open('react/dependencies.json', 'r') as file:
    data = json.load(file)

# Calculate the number of dependencies for each file
dependencies_count = {file: len(dependencies) for file, depender

# Sort files by the number of dependencies in descending order
sorted_dependencies = sorted(dependencies_count.items(), key=lar

# Get the top 3 files with the highest number of dependencies
```

```
top_files = sorted_dependencies[:3]

# Print the results
print("Top 3 files with the most dependencies:")
for file, count in top_files:
    print(f"{file}: {count} dependencies")

# Save the top 3 files to a new JSON file
top_files_dict = {file: data[file] for file, _ in top_files}

with open('top_dependencies.json', 'w') as output_file:
    json.dump(top_files_dict, output_file, indent=4)

print("Top dependencies saved to top_dependencies.json")
```

the output:

```
Top 3 files with the most dependencies:
packages/react-reconciler/src/ReactFiberWorkLoop.js: 45 dependencies
packages/react-reconciler/src/ReactFiberBeginWork.js: 36 dependencies
packages/react-reconciler/src/ReactFiberHooks.js: 27 dependencies
```

So, top 3 files with the most dependencies are:

packages/react-reconciler/src/ReactFiberWorkLoop.js: 45 dependencies

packages/react-reconciler/src/ReactFiberBeginWork.js: 36 dependencies
packages/react-reconciler/src/ReactFiberHooks.js: 27 dependencies

The detailed dependency relationship is saved in file **top_dependency.json**

3.Identify commit with most substantial changes between React v17.0.1 and v17.0.2

check out the commits between v17.0.1 and v17.0.2

```
git log --oneline v17.0.1..v17.0.2
```

check out the difference between v17.0.1 and v17.0.2

```
git diff v17.0.1..v17.0.2 --stat
```

The output:

```
[(base) [redacted] react % git log --oneline v17.0.1..v17.0.2                ]

12adaffef7 (tag: v17.0.2, origin/17.0.2) Remove scheduler sampling profiler shar
ed array buffer (#20840)
b2bbee7ba3 Disable (unstable) scheduler sampling profiler for OSS builds (#20832
)
8cc6ff2488 fix: use SharedArrayBuffer only when cross-origin isolation is enable
d (#20831)
[(base) [redacted] react % git diff v17.0.1..v17.0.2 --stat                   ]

 packages/scheduler/src/Scheduler.js                | 10 +--
 packages/scheduler/src/SchedulerFeatureFlags.js    |  2 +-
 packages/scheduler/src/SchedulerProfiling.js       | 51 ---------------
 .../src/__tests__/SchedulerProfiling-test.js       | 81 +++++++----------------
 4 files changed, 23 insertions(+), 121 deletions(-)
```

There are three commits between v17.0.1 and v17.0.2: 12adaffef7 (**tag: v17.0.2**, **origin/17.0.2**) ,b2bbee7ba3, and 8cc6ff2488

The stat info shows there are 4 files changed with 23 insertions and 121 deletions.

By analyzing the commit message and the difference between React 17.0.1 and 17.0.2, we consider the most substantial commit to be 12adaffef7 (**tag: v17.0.2**, **origin/17.0.2**), as its commit message is 'Remove scheduler sampling profiler

shared array buffer (#20840)'. It introduces changes to the scheduler sampling profiler and matches with the difference of 121 deletions.

The commit hash and all the change information are documented in file **commit_change.md**

commit_change.md

```
**Commit Hash**: 12adaffe7
**Commit Message**: Remove scheduler sampling profiler shared array buffer (#20840)
**Number of Files Changed**: 4
**Number of Insertions**: 23
**Number of Deletions**: 121
**Files Affected**:
- `packages/scheduler/src/Scheduler.js` (+10, -10)
- `packages/scheduler/src/SchedulerFeatureFlags.js` (+2, -0)
- `packages/scheduler/src/SchedulerProfiling.js` (+0, -51)
- `src/__tests__/SchedulerProfiling-test.js` (+81, -60)
```

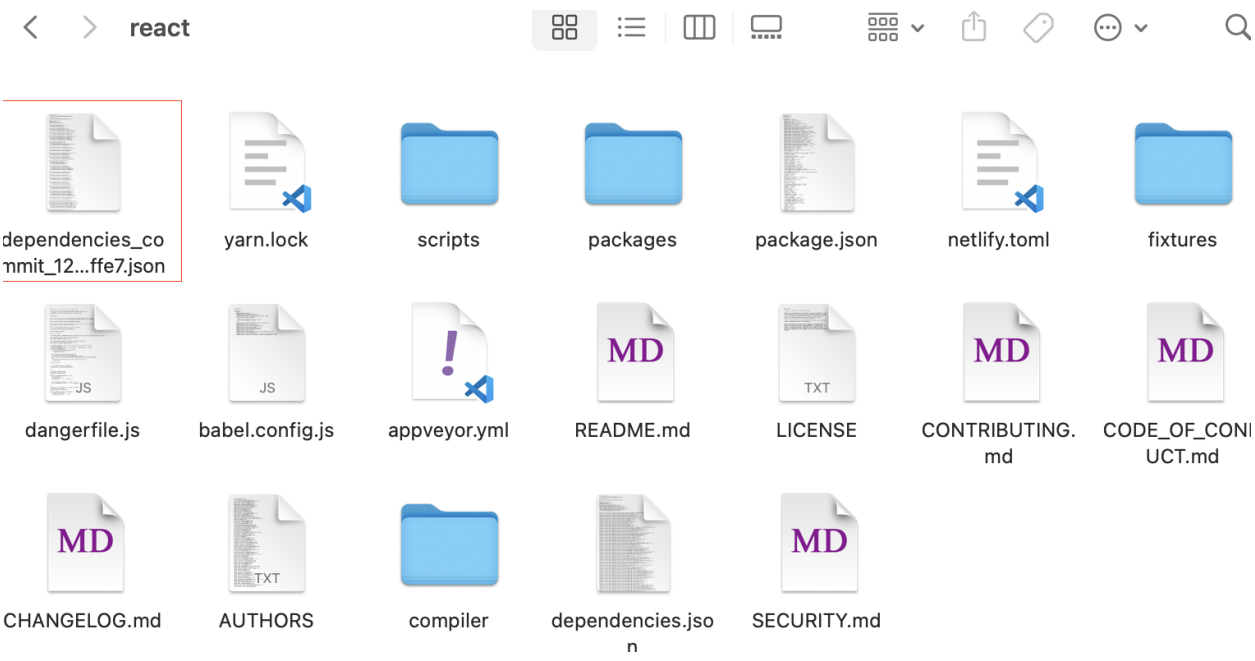
4.Analyze the dependencies at the commit mentioned above and compare it with 18.3.1 version

(1)Check out to the commit and use Madge to analyze the file dependencies at that time

```
git checkout v17.0.2
```

```
(base) [redacted] react % git checkout v17.0.2
注意：正在切换到 'v17.0.2'。

您正处于分离头指针状态。您可以查看、做试验性的修改及提交，并且您可以在切换
回一个分支时，丢弃在此状态下所做的提交而不对分支造成影响。

如果您想要通过创建分支来保留在此状态下所做的提交，您可以通过在 switch 命令
中添加参数 -c 来实现（现在或稍后）。例如：

  git switch -c <新分支名>

或者撤销此操作：

  git switch -

通过将配置变量 advice.detachedHead 设置为 false 来关闭此建议

HEAD 目前位于 12adaffef7 Remove scheduler sampling profiler shared array buffer
(#20840)
```

```
madge . --json > dependencies_commit_12adaffe7.json
```



(2)Compare these dependencies with the current ones and document the changes(introduced and removed).

Use python to compare these two dependency files

compare_dependencies.py

```
import json

# Load dependencies from JSON files
with open('dependencies.json', 'r') as current_file:
    current_dependencies = json.load(current_file)

with open('dependencies_commit_12adaffe7.json', 'r') as commit_f
    commit_dependencies = json.load(commit_file)

# Compare dependencies
added_dependencies = {}
removed_dependencies = {}

for file in current_dependencies:
    current_set = set(current_dependencies.get(file, []))
    commit_set = set(commit_dependencies.get(file, []))
```

```
    added = current_set - commit_set
    removed = commit_set - current_set

    if added:
        added_dependencies[file] = list(added)
    if removed:
        removed_dependencies[file] = list(removed)

# Save the results
with open('dependencies_added.json', 'w') as added_file:
    json.dump(added_dependencies, added_file, indent=4)

with open('dependencies_removed.json', 'w') as removed_file:
    json.dump(removed_dependencies, removed_file, indent=4)

print("Comparison complete! Results saved to 'dependencies_adde
```

The dependencies added are documented in dependencies_added.json and those removed are in dependencies_removed.json.


{} dependencies_removed.json > ...
{
"dangerfile.js": [
"scripts/rollup/stats.js"
],
"fixtures/flight/src/index.js": [
"fixtures/flight/src/App.js"
],
"packages/react-client/flight.js": [
"packages/react-client/src/ReactFlightClientStream.js"
],
"packages/react-client/src/ReactFlightClient.js": [
"packages/react-client/src/ReactFlightClientHostConfig.js"
],
"packages/react-devtools-inline/src/backend.js": [
"packages/react-devtools-inline/src/constants.js"
],
"packages/react-devtools-inline/src/frontend.js": [
"packages/react-devtools-inline/src/constants.js"
],
"packages/react-devtools-shared/src/__tests__/console-test.js": [
"packages/react-devtools-shared/src/__tests__/utils.js"
],
"packages/react-devtools-shared/src/__tests__/storeStressSync-test.js": [
"packages/react-devtools-shared/src/__tests__/storeSerializer.js"
],
"packages/react-devtools-shared/src/__tests__/storeStressTestConcurrent-test.js": [
"packages/react-devtools-shared/src/__tests__/storeSerializer.js"
],