Шаблон отчёта по лабораторной работе

Дисциплина: Архитектура компьютеров. Операционные системы.

Романова Елизавета Романовна

Содержание

Список иллюстраций

No table of figures entries found.

Список таблиц

No table of figures entries found.

1 Цель работы

Изучить идеологию и применение средств контроля версий. Освоить умения по работе с git.

2 Задание

1. Настройка GitHub.
2. Базовая настройка Git.
3. Создание SSH-ключа.
4. Создание рабочего пространства и репозитория курса на основе шаблона.
5. Верификация коммитов с помощью PGP.
6. Настройка каталога курса.

3 Теоретическое введение

Системы контроля версий. Общие понятия

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется.

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных.

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом.

Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд.

4 Выполнение лабораторной работы

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. **¿fig:001?**). Первым делом я устанавливаю git:

```
[erromanova@erromanova ~]$ sudo -i
[sudo] пароль для erromanova:
[root@erromanova ~]# dnf install git
Обновление и загрузка репозиториев:
 Fedora 41 - x86_64 - Updates              100% |  30.2 KiB/s |  25.6 KiB |  00m01s
Репозитории загружены.
Пакет "git-2.47.0-1.fc41.x86_64" уже установлен.

Нечего делать.
```

Далее устанавливаю gh:

```
[root@erromanova ~]# dnf install gh
Обновление и загрузка репозиториев:
Репозитории загружены.
Пакет "gh-2.65.0-1.fc41.x86_64" уже установлен.

Нечего делать.
```

Базовая настройка git. Завожу имя и email владельца репозитория, настраиваю utf-8 в выводе сообщений git, завожу имя начальной ветки (будем называть её master), параметр autocrlf и safecrlf. Создаю ключ ssh.

```
[root@erromanova ~]# git config --global user.name "erromanova"
[root@erromanova ~]# git config --global user.email "lizaveta021037@mail.ru"
[root@erromanova ~]# git config --global core.quotepath false
[root@erromanova ~]# git config --global init.defaultBranch master
[root@erromanova ~]# git config --global core.autocrlf input
[root@erromanova ~]# git config --global core.safecrlf warn
[root@erromanova ~]# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /root/.ssh/id_rsa
Enter passphrase for "/root/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:y9/rSzekT2cOU7YH9BKMYZWYPFucEZVfhifHg83Fn6w root@erromanova
The key's randomart image is:
+---[RSA 4096]----+
|           .o=BX=|
|           .==B=B|
|            .+==*|
|            .. =o|
|        S     = +|
|       . .   E =.|
|        o   o *.+|
|         . o + B.|
|          ..=o. .|
+----[SHA256]-----+
[root@erromanova ~]#

ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519): /root/.ssh/id_ed25519
Enter passphrase for "/root/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_ed25519
```

Далее я начинаю создание gpg.

```
[root@erromanova ~]# git config --global user.name "erromanova"
[root@erromanova ~]# git config --global user.email "lizaveta021037@mail.ru"
[root@erromanova ~]# git config --global core.quotepath false
[root@erromanova ~]# git config --global init.defaultBranch master
[root@erromanova ~]# git config --global core.autocrlf input
[root@erromanova ~]# git config --global core.safecrlf warn
[root@erromanova ~]# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /root/.ssh/id_rsa
Enter passphrase for "/root/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:y9/rSzekT2cOU7YH9BKMYZWYPFucEZVfhifHg83Fn6w root@erromanova
The key's randomart image is:
+---[RSA 4096]----+
|           .o=BX=|
|           .==B=B|
|            .+==*|
|            .. =o|
|        S     = +|
|       . .   E =.|
|        o   o *.+|
|         . o + B.|
|          ..=o. .|
+----[SHA256]-----+
[root@erromanova ~]#

ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519): /root/.ssh/id_ed25519
Enter passphrase for "/root/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_ed25519
```

```
[root@erromanova ~]# gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: глубина: 0  достоверных:   1  подписанных:   0  доверие: 0-, 0q, 0n, 0m, 0f, 1u
[keyboxd]
---------
sec   rsa4096/E6131BF331C27BB6 2025-03-06 [SC]
      5141779902B59A28C9D46423E6131BF331C27BB6
uid                 [  абсолютно ] erromanova <lizaveta021037@mail.ru>
ssb   rsa4096/047F9C86AB2FFEAE 2025-03-06 [E]
```

Добавляю ключ gpg на github.

```
[root@erromanova ~]# gpg --armor --export E6131BF331C27BB6
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGfJ8kwBEADIUb3unbHjwQhSuJ9mFRykRUy31v8pWDYrPacl6CmoLn/b/4Rj
S4/jxOV19VCtVZ9lRYeWnAIGIB+DliKBoB3g4NJ03qMjALRBL7dBH7BYShqN1ys6
+NLR21VXvCi3zlm07PBQtZw2EGcfjqxqKmAgGYsW+vVeOHFNLysiGpM5OEzdGGRz
UYOmDccS9k47XBSwqrAfWDMdIqZcjfZatZdAybyL+2imGduwGLvMEGp/AsMvOFtf
XErP+vaviQVXPBGZLeJ3DFOXvX4+A73QUgWsKWC6UtuVzxxt1M5IjBd4CCivBNT1
DLScG+hgT32+463MLY7I4K6wMKcSGFAu8OQ/af9TT4LEucL34nYhcI7VWNtAJ4a1
Fl4BELV1FVGGl6QAKGhIAzhnf2soj6zXtm5+kYLjX0r/XnzVb0+TF0VfqZy7npj1
O/oDiWjLh//+p7eE3VyVmiIccYMtUJqVVhaMZ9AELnYJwC2sCbdEfcHaWXiakgA7
ep1RMcOEPrgkBXT6J0tCJa0AlGFiuKGs/s6fXmfC22OI3GxDd8E6+axQNcY0Xu2E
rsSU7v+SNrwhUygl1XuxrKBh/EFSnw5K18MDESZS6bEif3T91CaNN3W8un4H4/7f
```

Настройка автоматических подписей коммитов git. Используя введёный email, указываю Git применять его при подписи коммитов:

```
[root@erromanova ~]# git config --global user.signingkey E6131BF331C27BB6
[root@erromanova ~]# git config --global commit.gpgsign true
[root@erromanova ~]# git config --global gpg.program $(which gpg2)
[root@erromanova ~]# gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 5A76-A4FD
Press Enter to open https://github.com/login/device in your browser...
Authorization required, but no authorization protocol specified

Error: cannot open display: :0
failed to authenticate via web browser: This 'device_code' has expired. (expired_token)
[root@erromanova ~]# gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser
```

Создание репозитория курса на основе шаблона. Создаю шаблон рабочего пространства. Например, для 2024–2025 учебного года и предмета «Операционные системы» (код предмета os-intro) создание репозитория примет следующий вид:

```
[root@erromanova ~]# mkdir -p ~/work/study/2024-2025/"Операционные системы"
```

```
[root@erromanova ~]# cd ~/work/study/2024-2025/"Операционные системы"
[root@erromanova Операционные системы]# gh repo create study_2024-2025_os-intro --template=ya
dharma/course-directory-student-template --public
GraphQL: Could not clone: Name already exists on this account (cloneTemplateRepository)
[root@erromanova Операционные системы]# git clone --recursive git@github.com:erromanova/study
024-2025_os-intro.git os-intro
Клонирование в «os-intro»...
Enter passphrase for key '/root/.ssh/id_ed25519':
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 36 (delta 1), reused 21 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (36/36), 19.37 КиБ | 4.84 МиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdo
-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-templat
git) зарегистрирован по пути «template/report»
Клонирование в «/root/work/study/2024-2025/Операционные системы/os-intro/template/presentatio
...
```

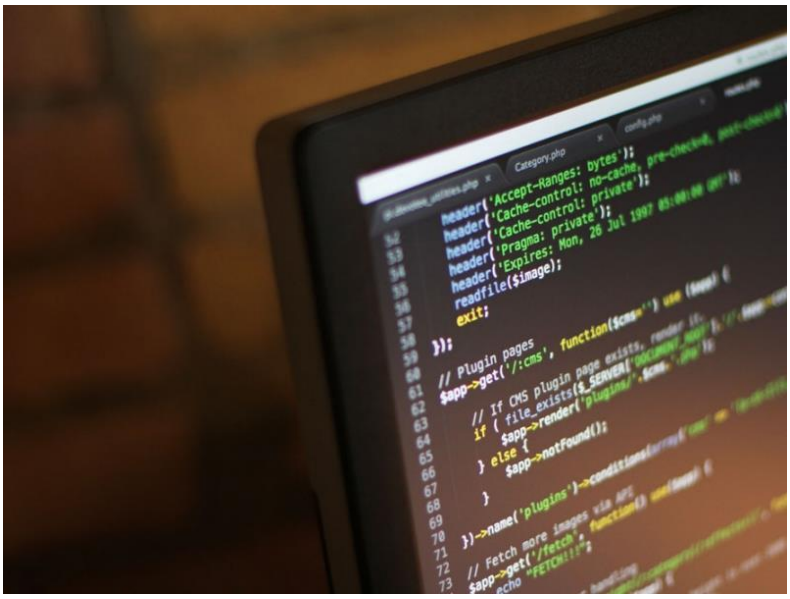
Перехожу в каталог курса. Удаляю лищние файлы и создаю необходимые каталоги

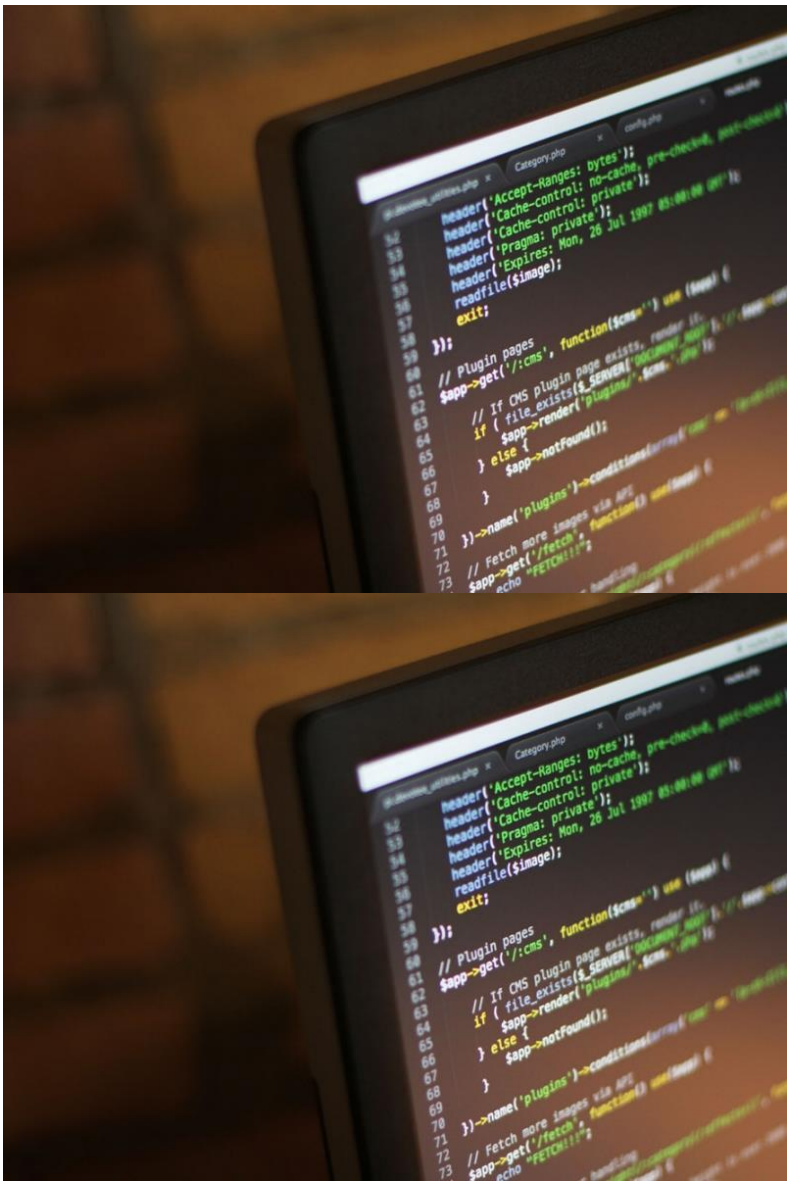
```
[root@erromanova Операционные системы]# cd ~/work/study/2024-2025/"Операционные системы"/os-int
ro
[root@erromanova os-intro]# rm package.json
rm: удалить обычный файл 'package.json'? y
[root@erromanova os-intro]# echo os-intro > COURSE
[root@erromanova os-intro]# make
Usage:
  make <target>

Targets:
  list                          List of courses
  prepare                       Generate directories structure
  submodule                     Update submules
```

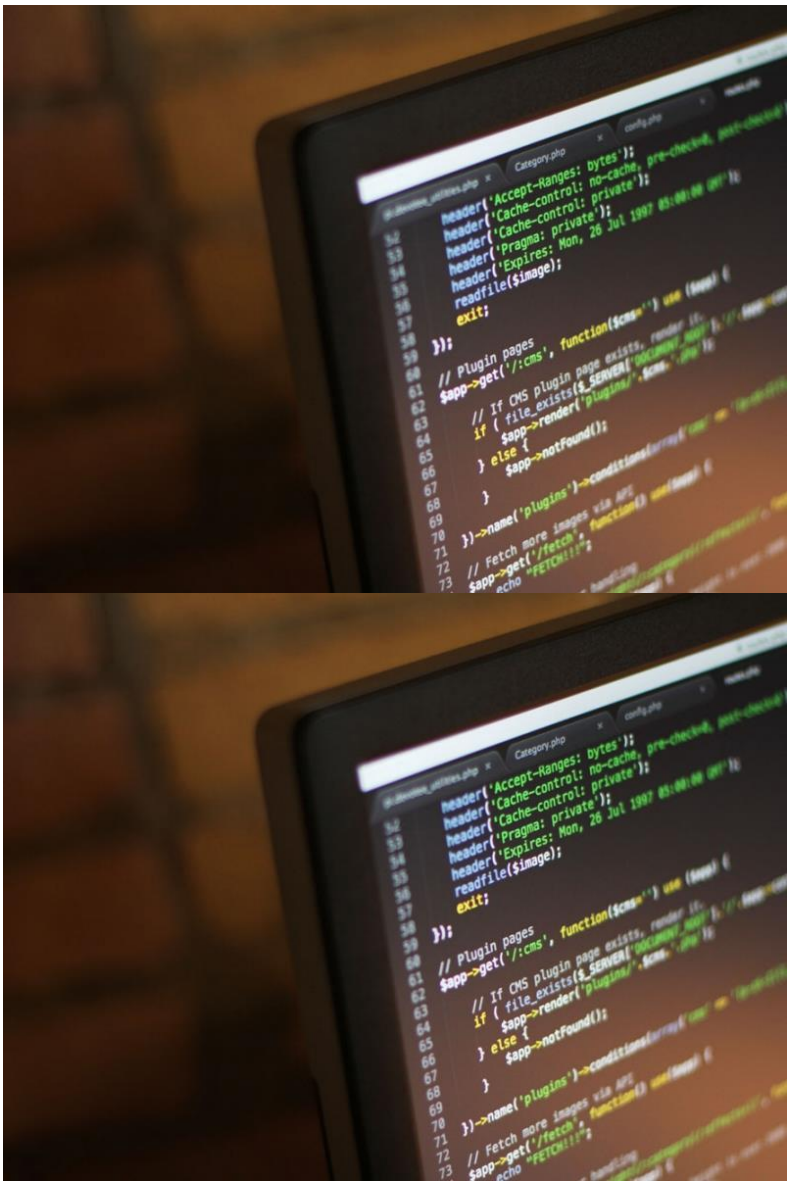
И отправляю файлы на сервер.

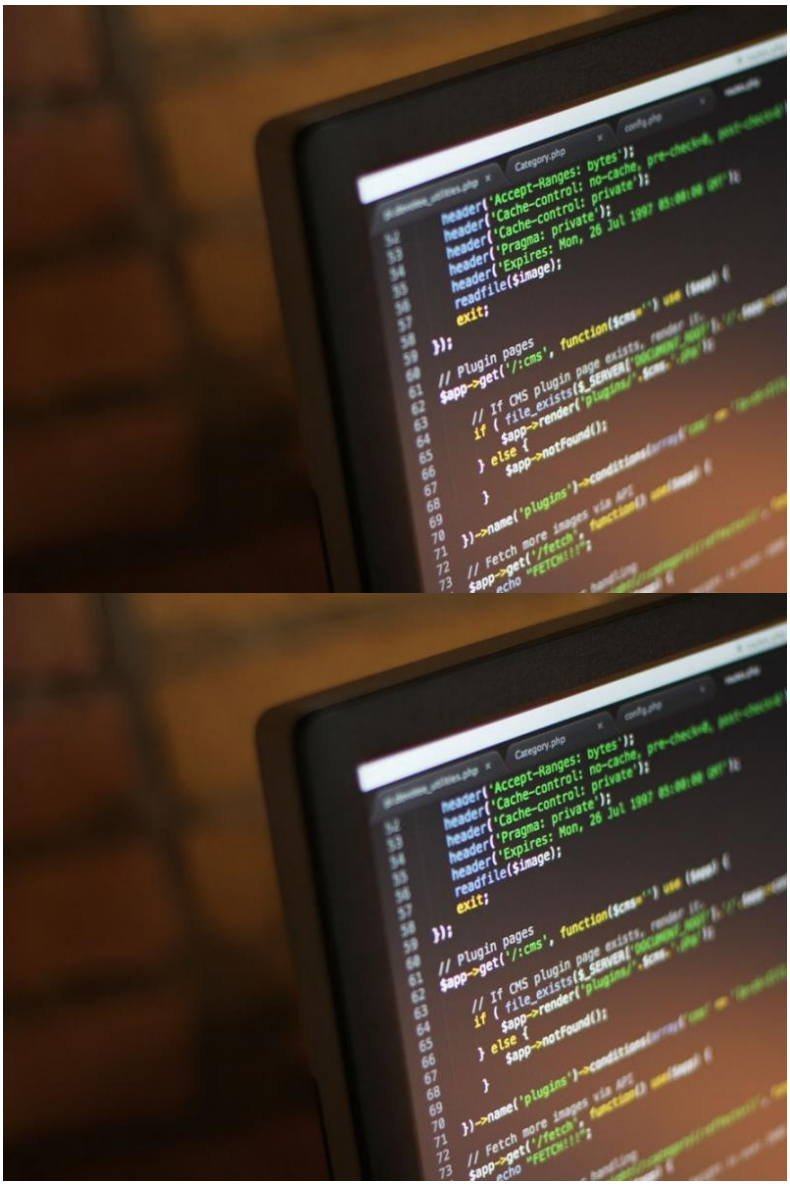
```
[root@erromanova os-intro]# git add .
[root@erromanova os-intro]# git commit -am 'feat(main): make course structure'
[root@erromanova os-intro]# git push
```

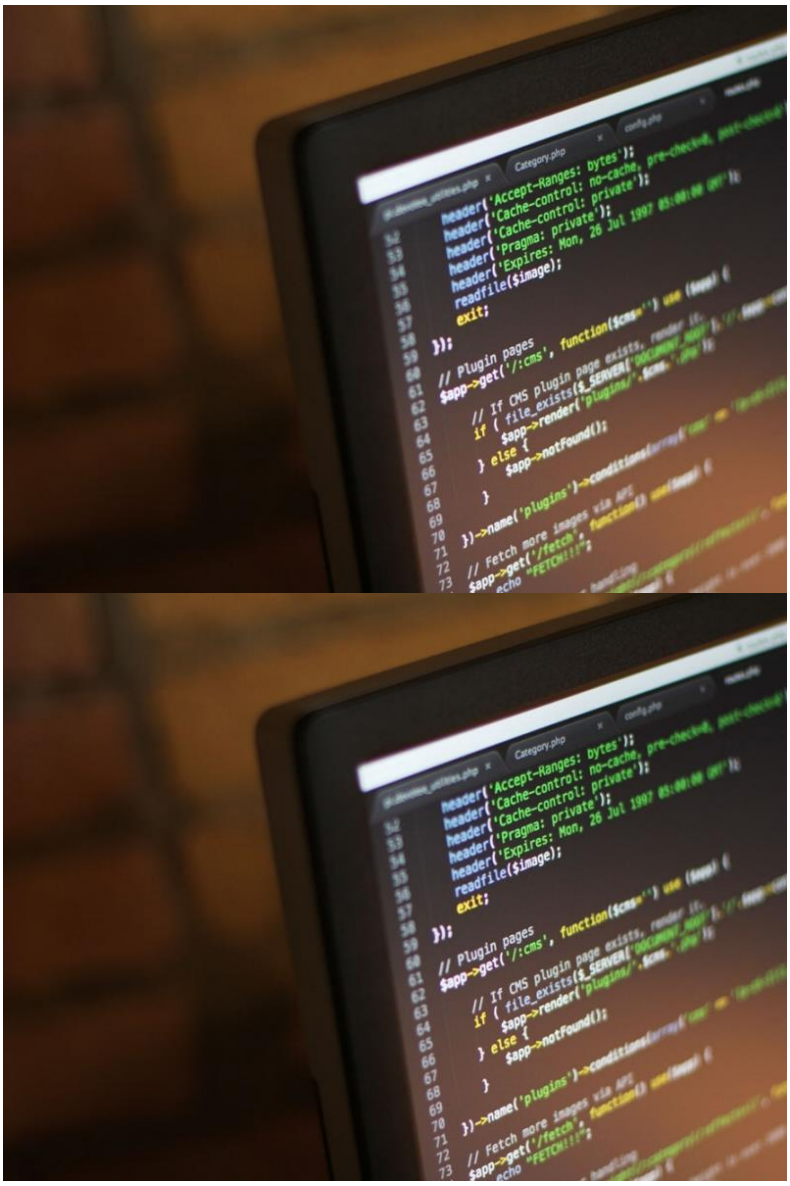
fig:001 width=70%}

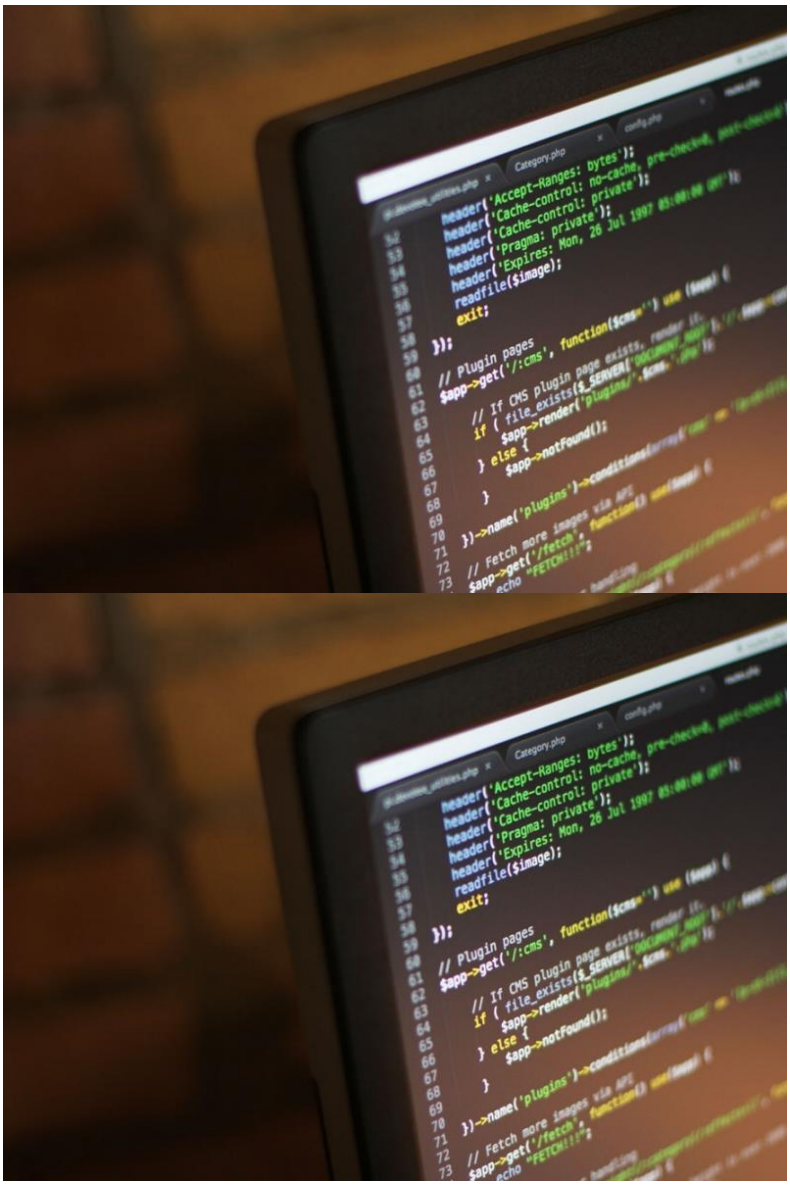


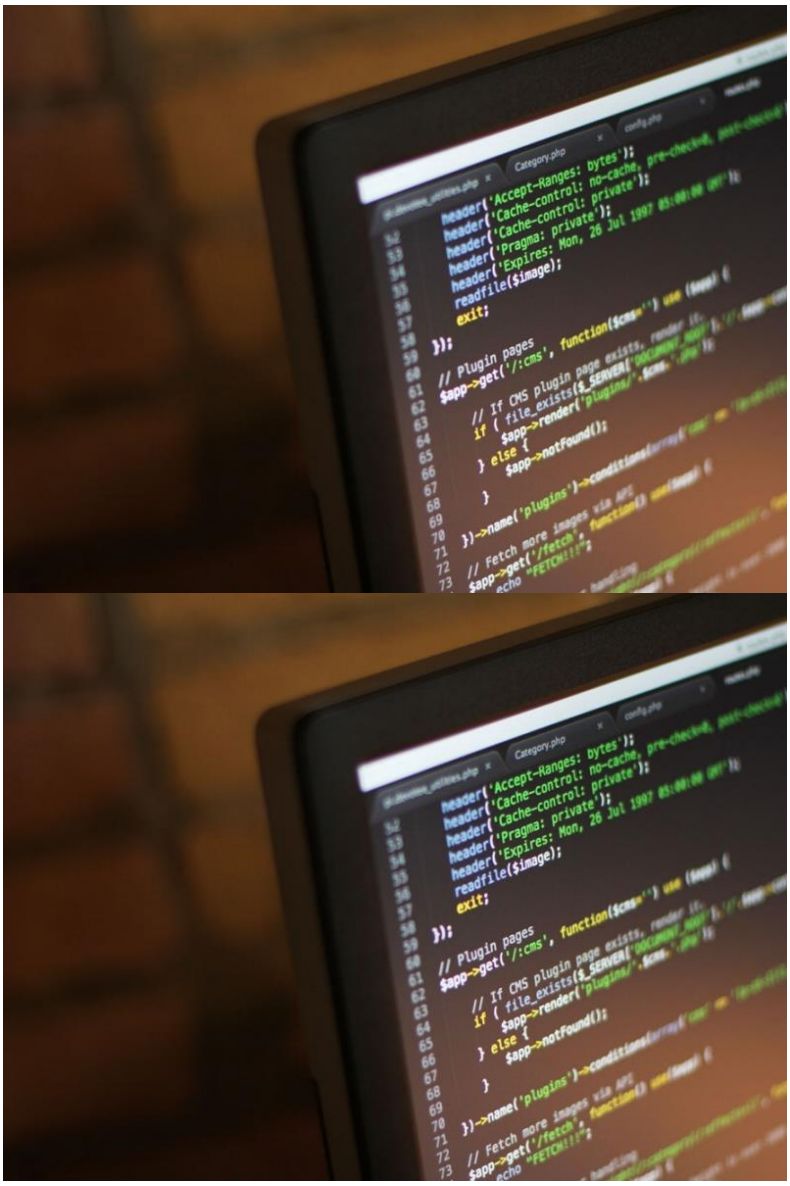


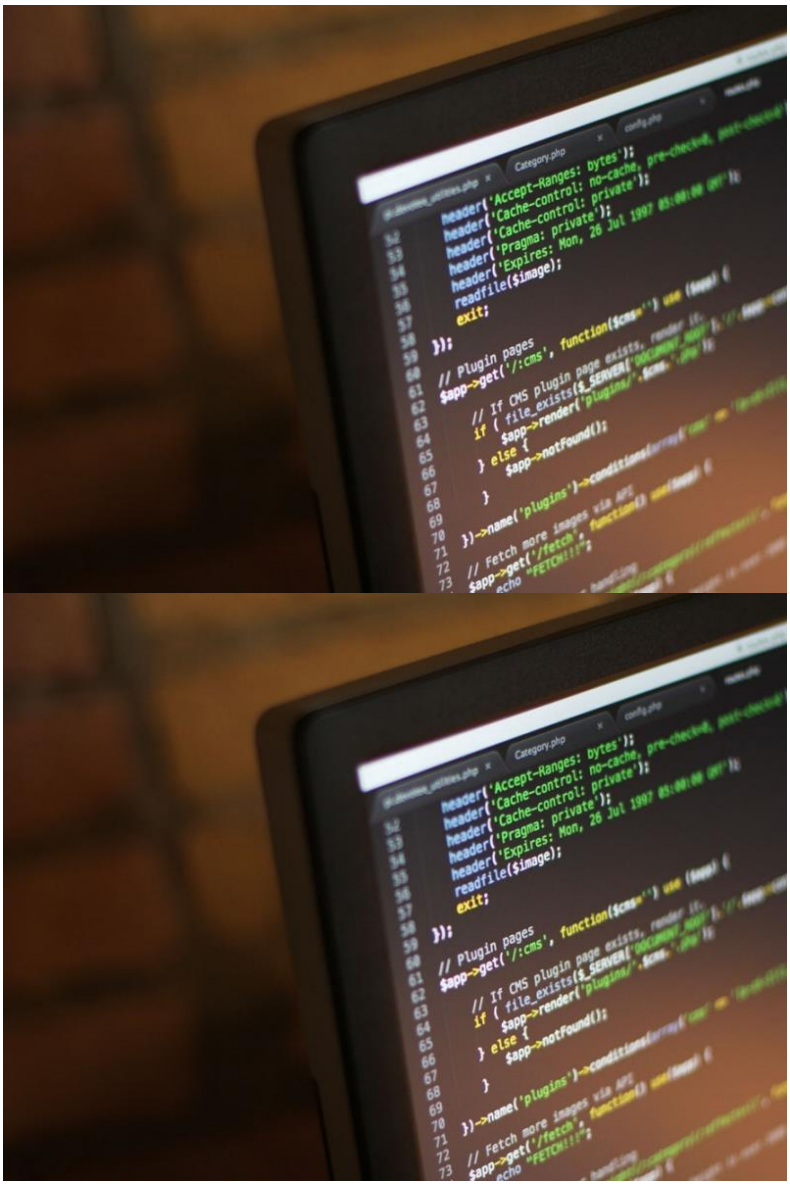


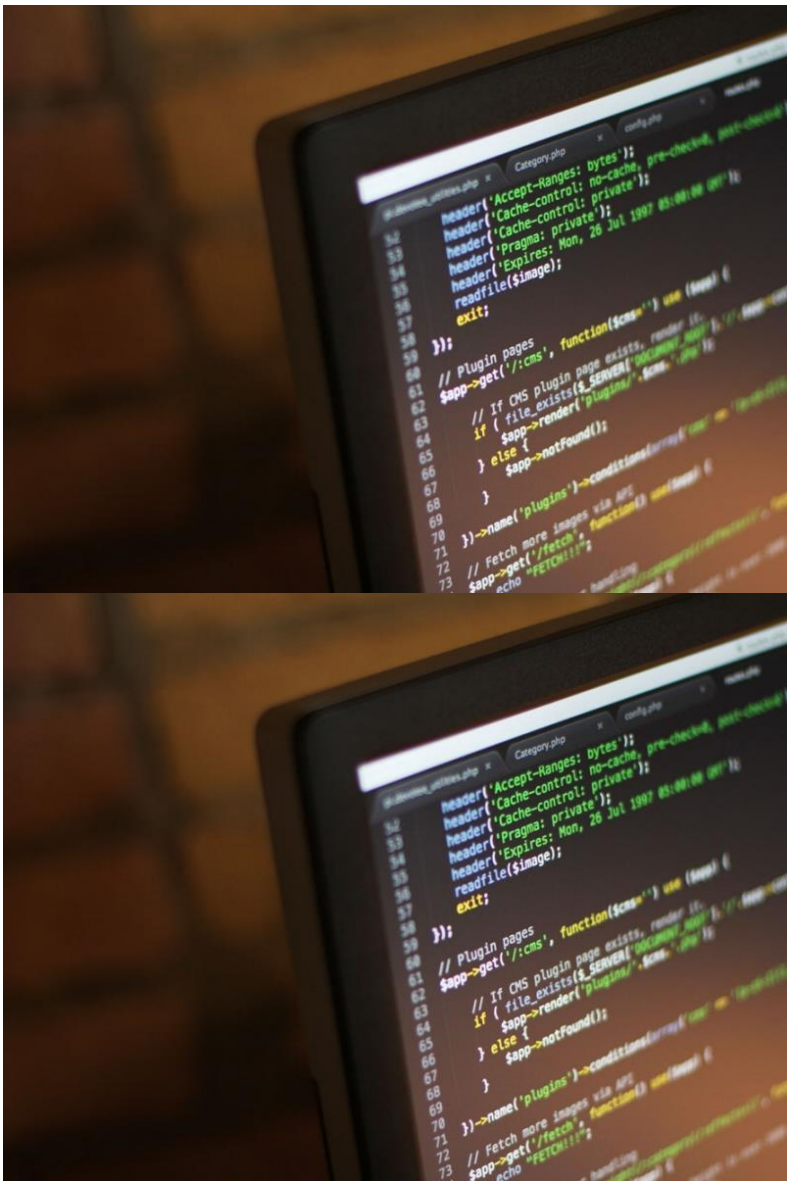


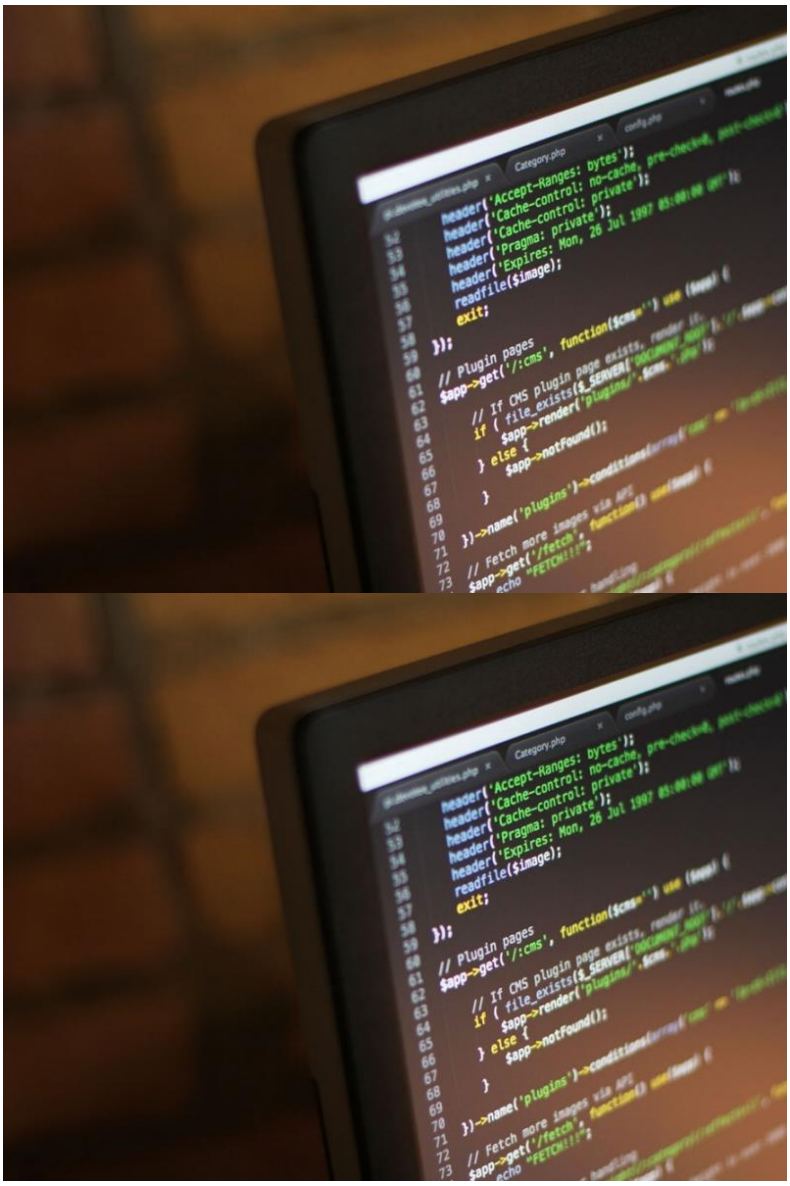


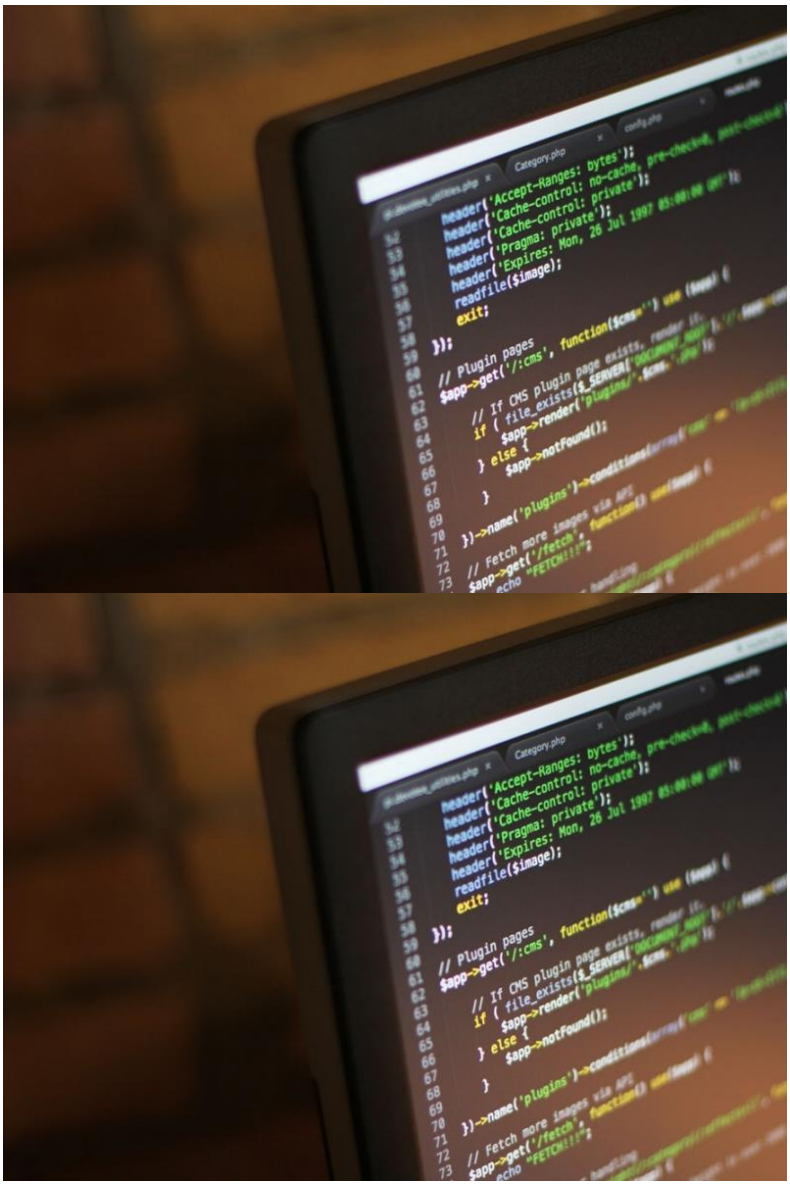


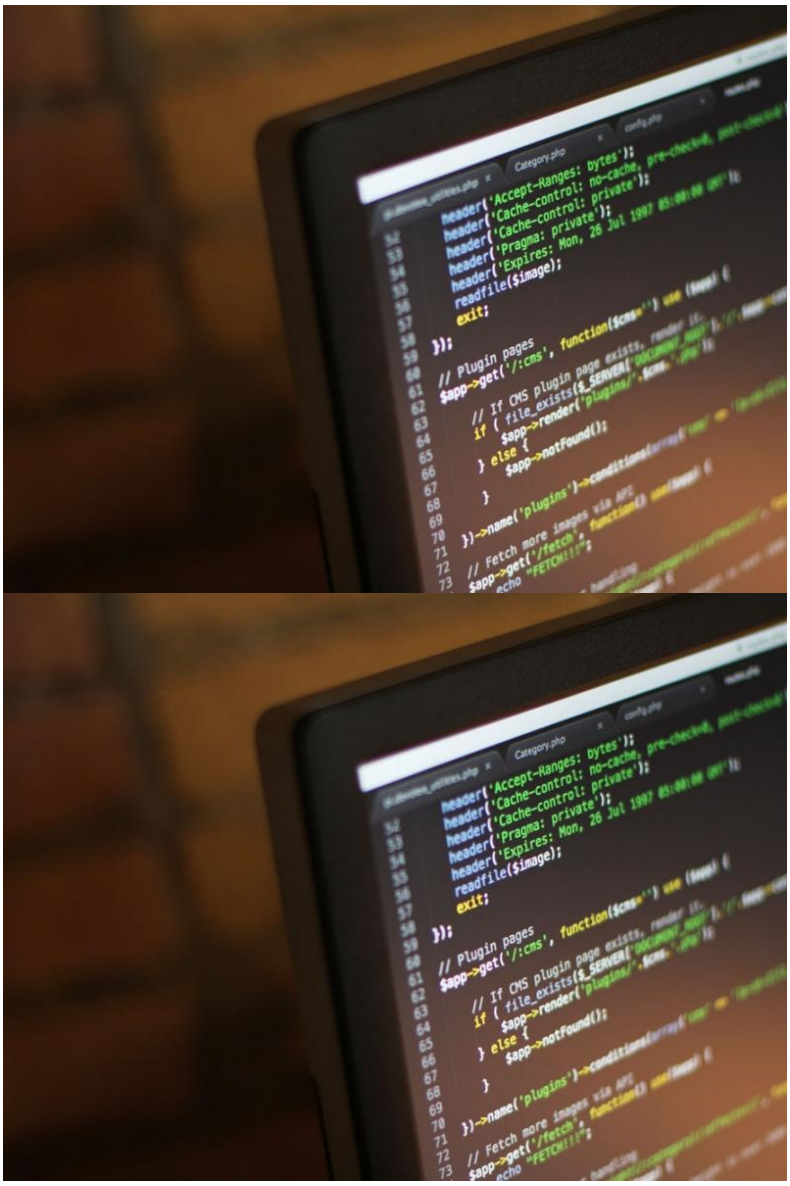


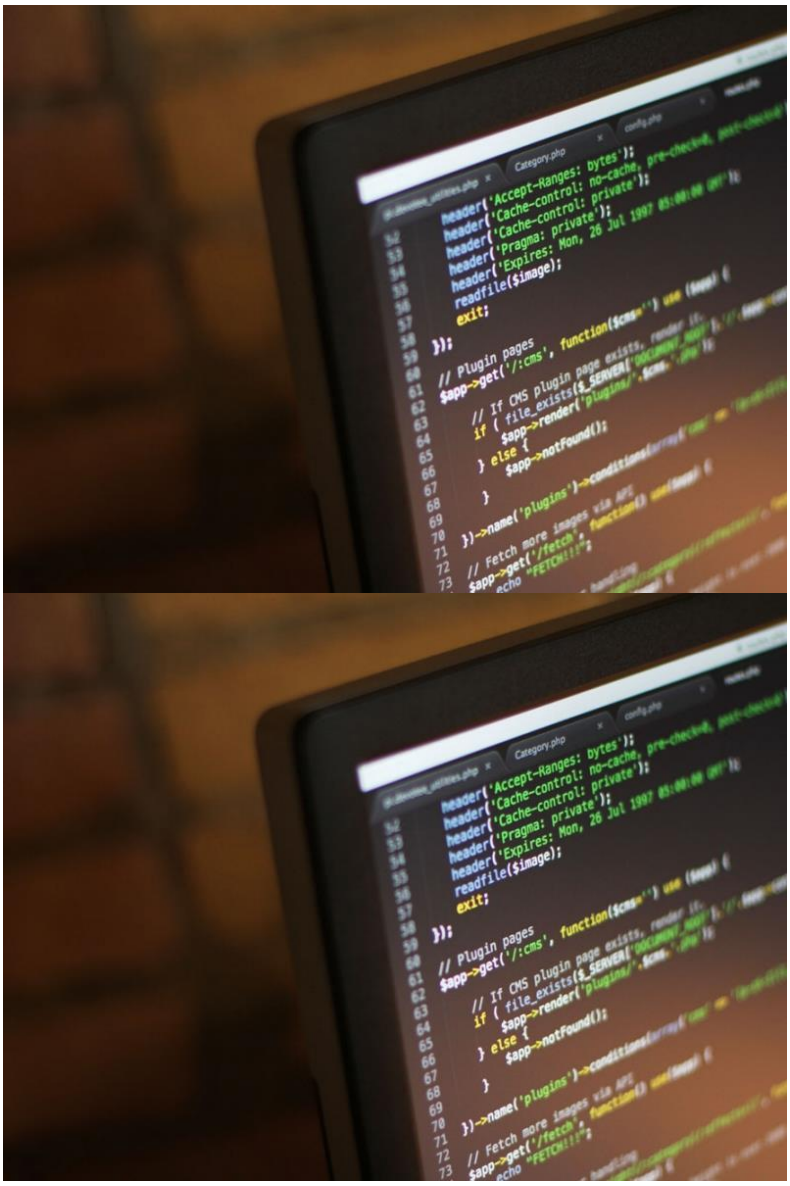


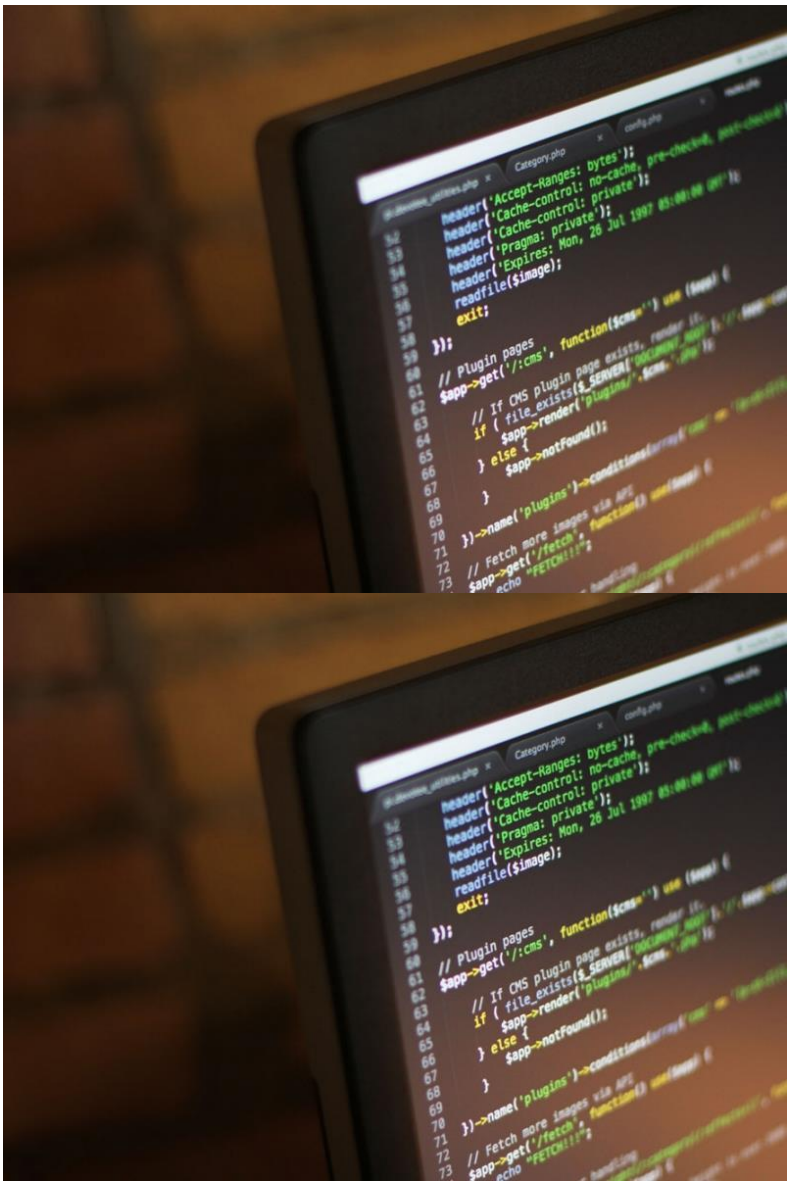


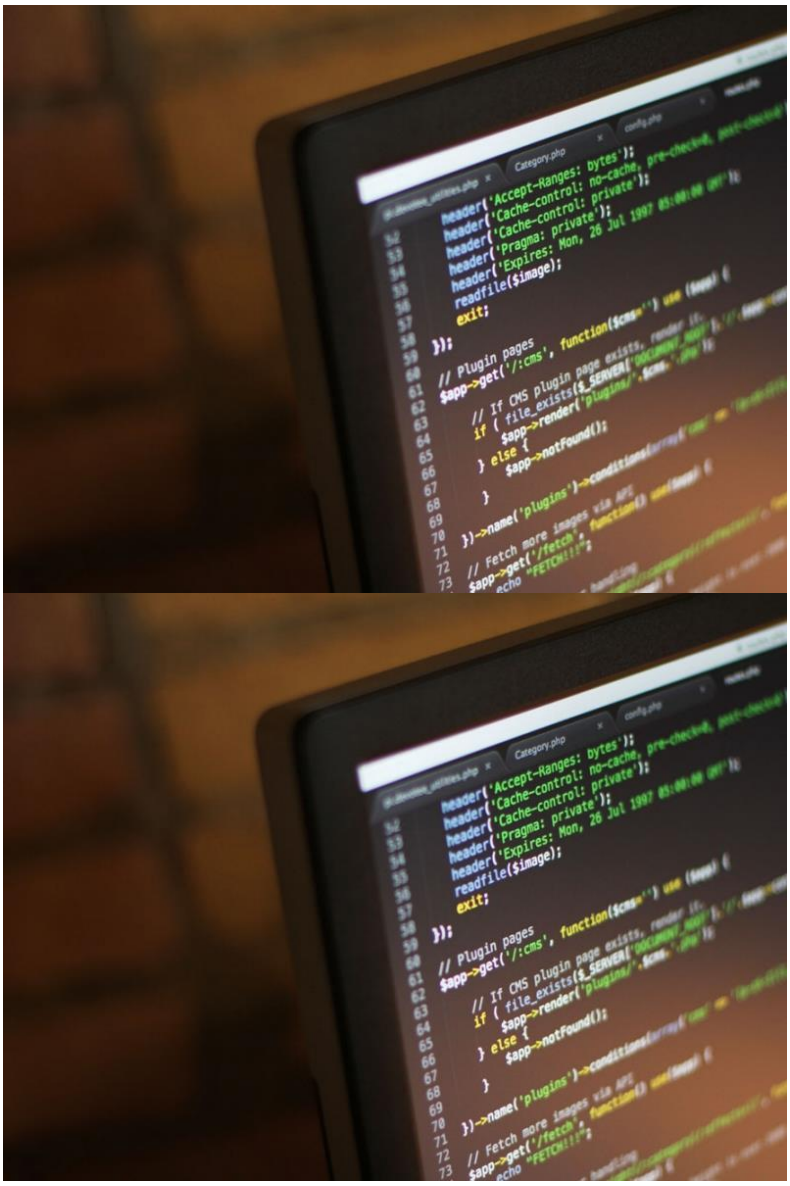


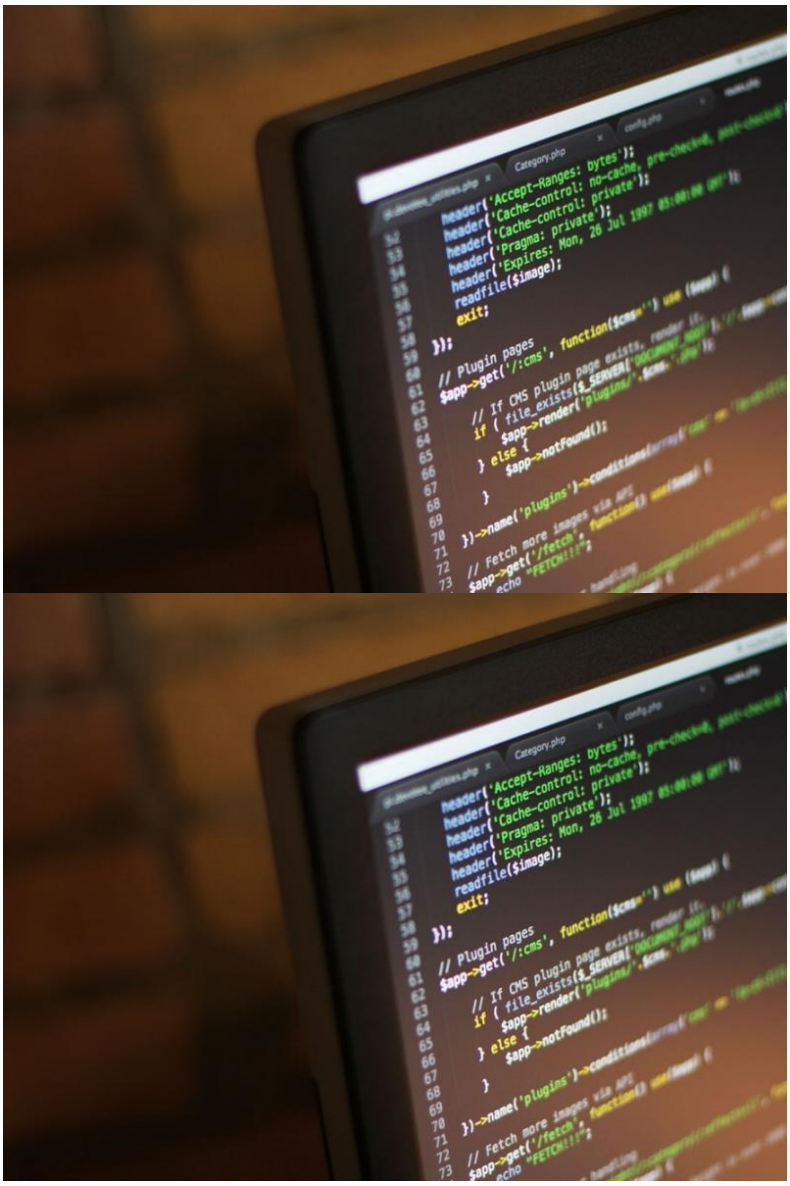


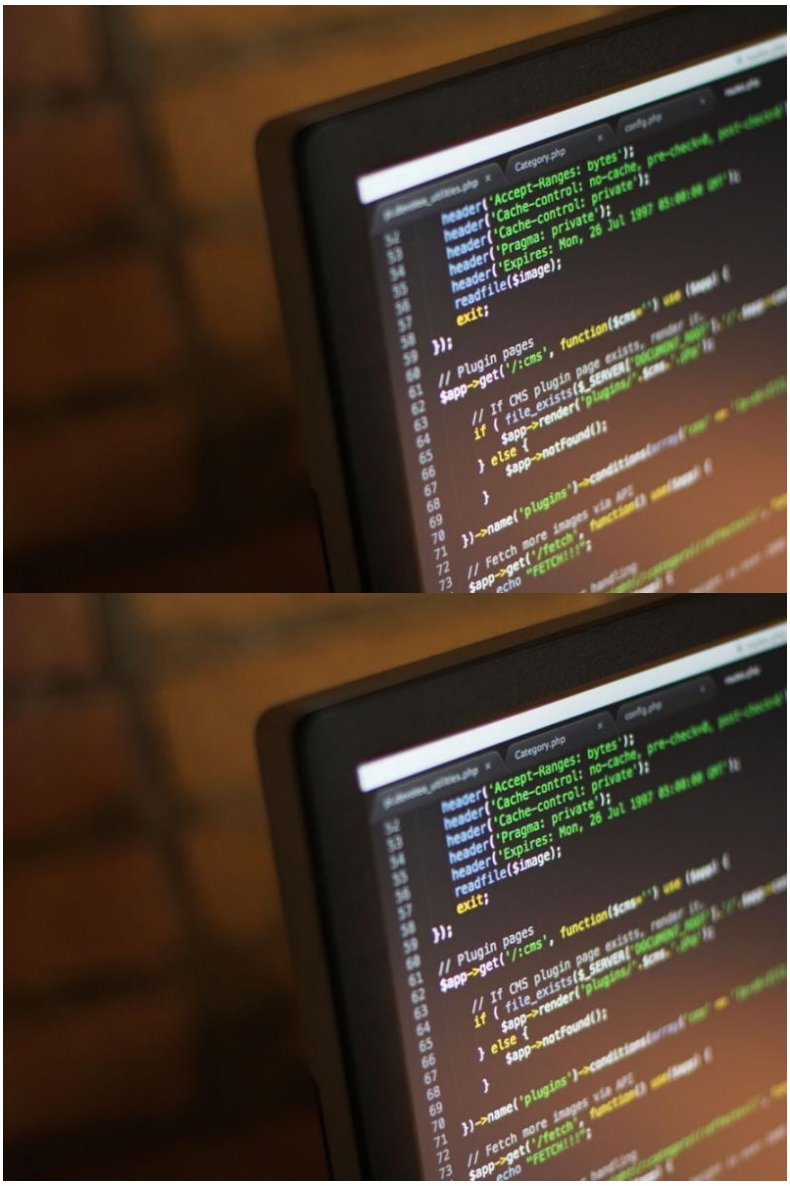


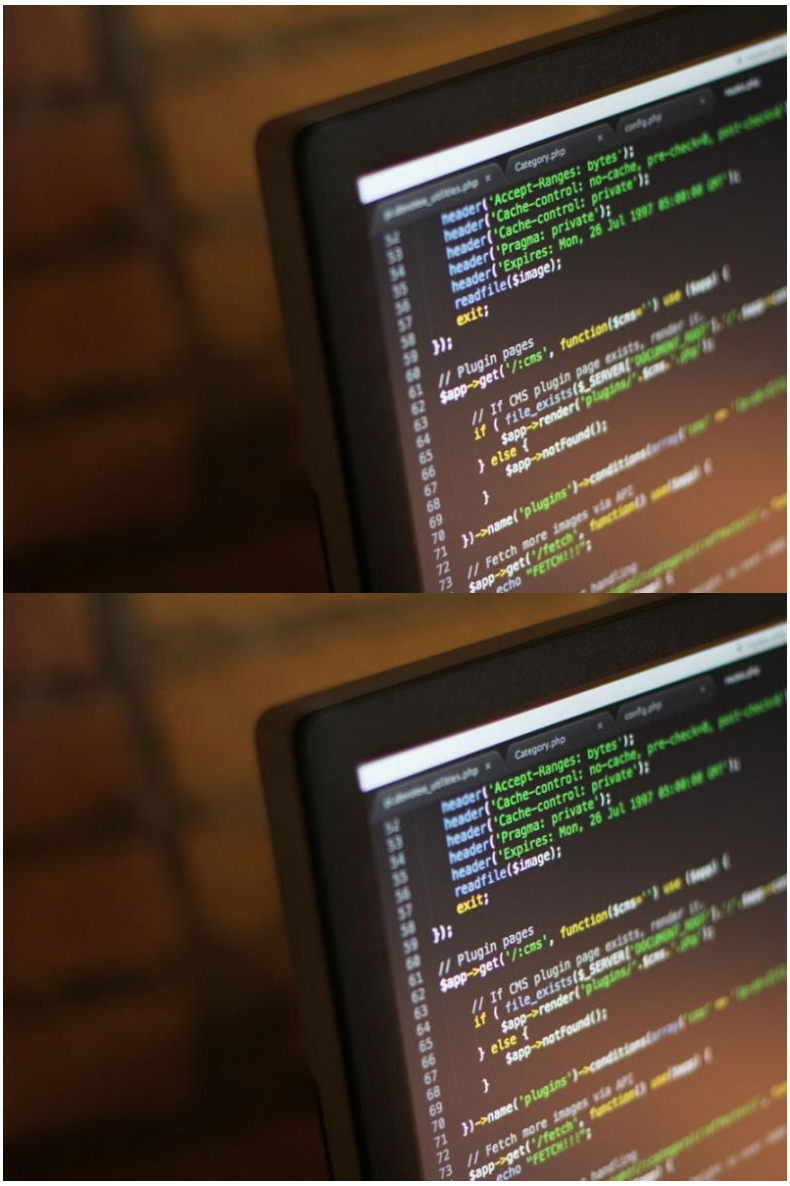


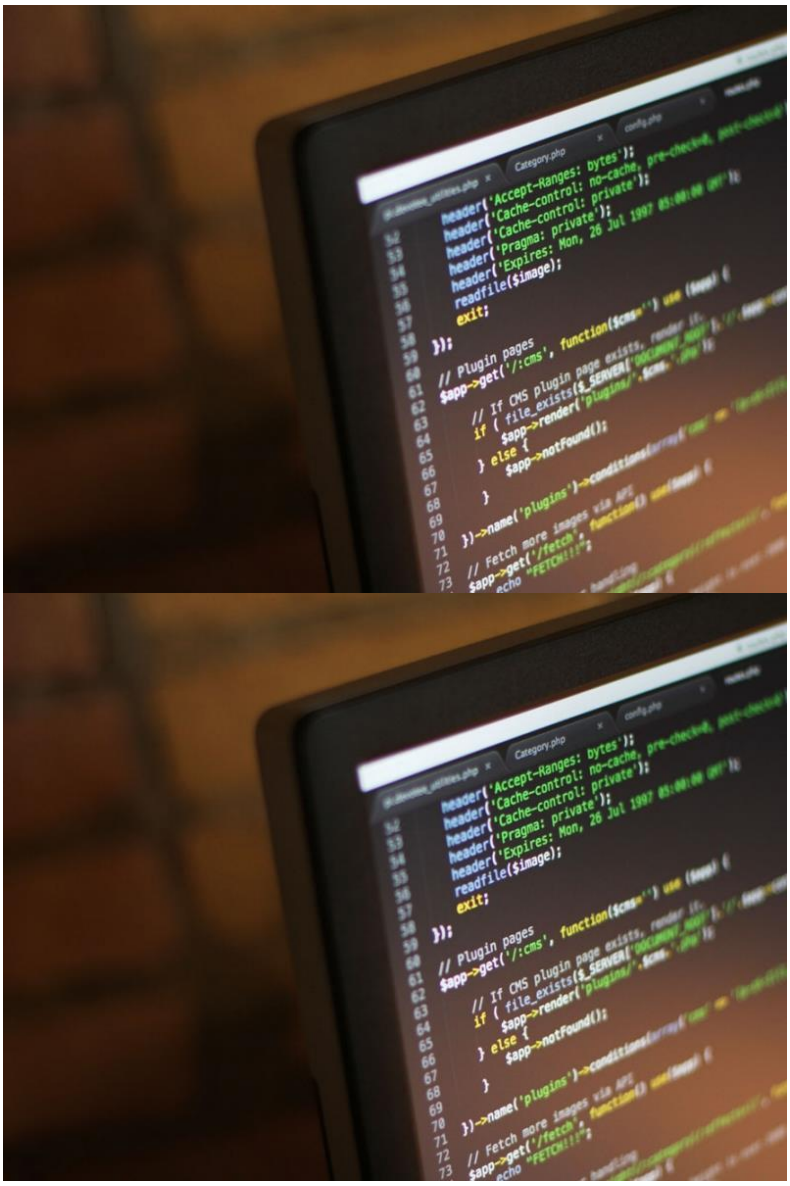


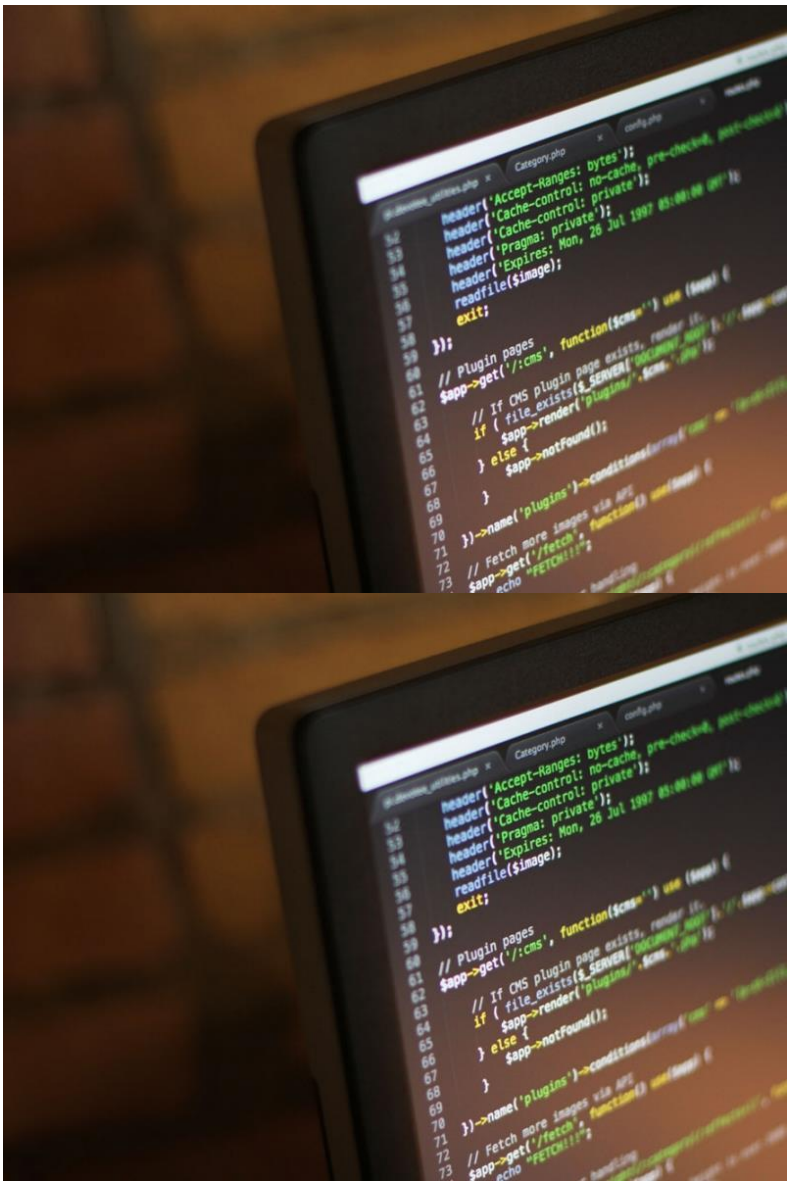


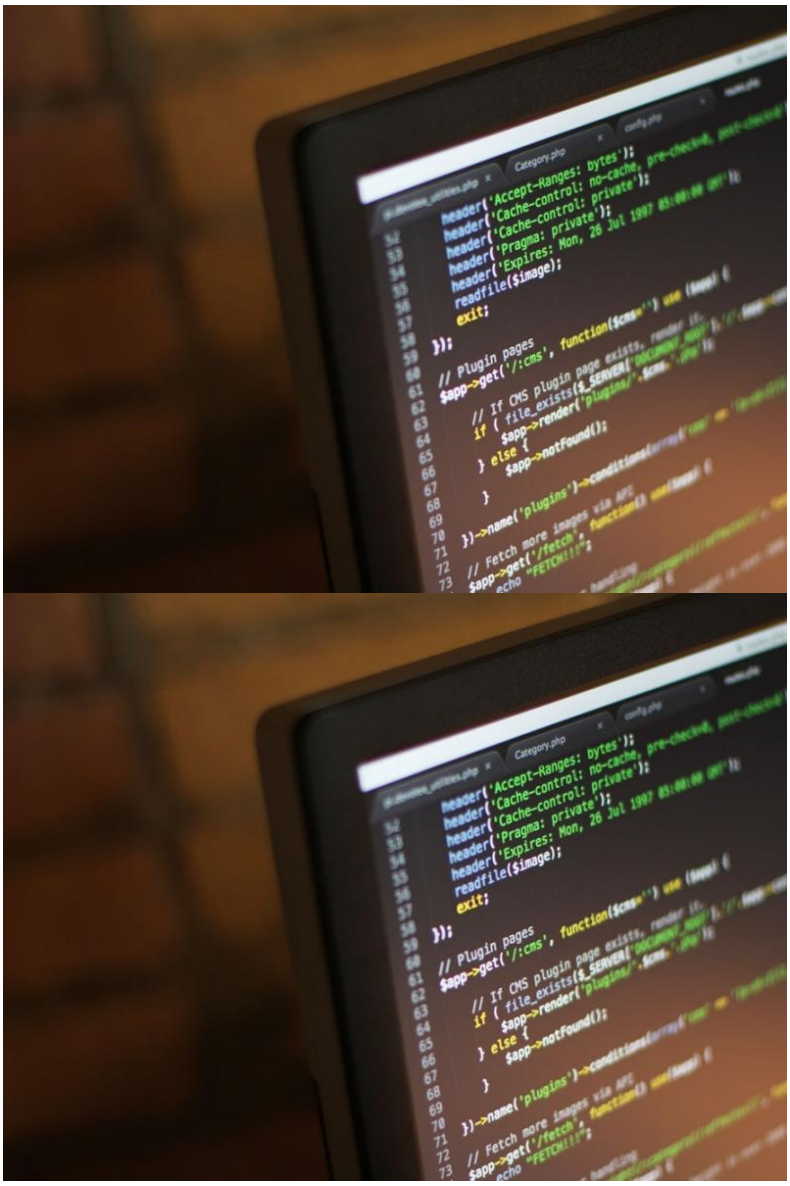


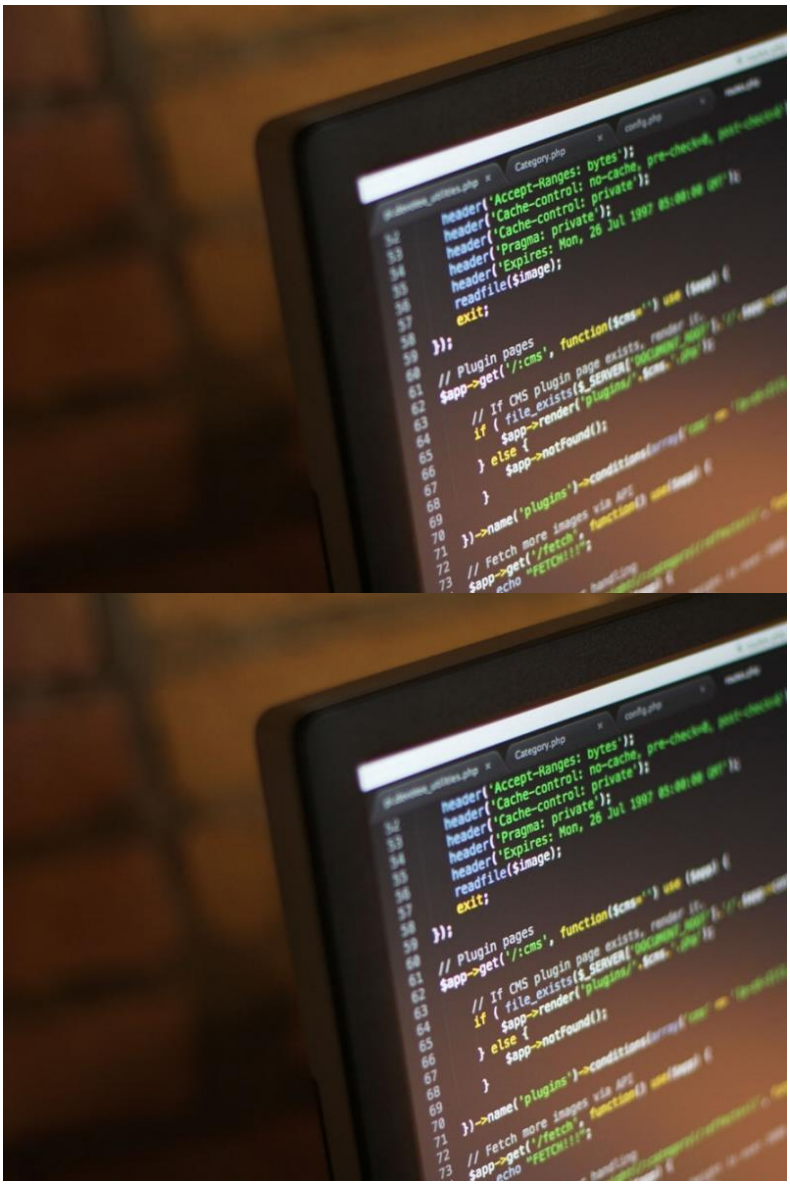


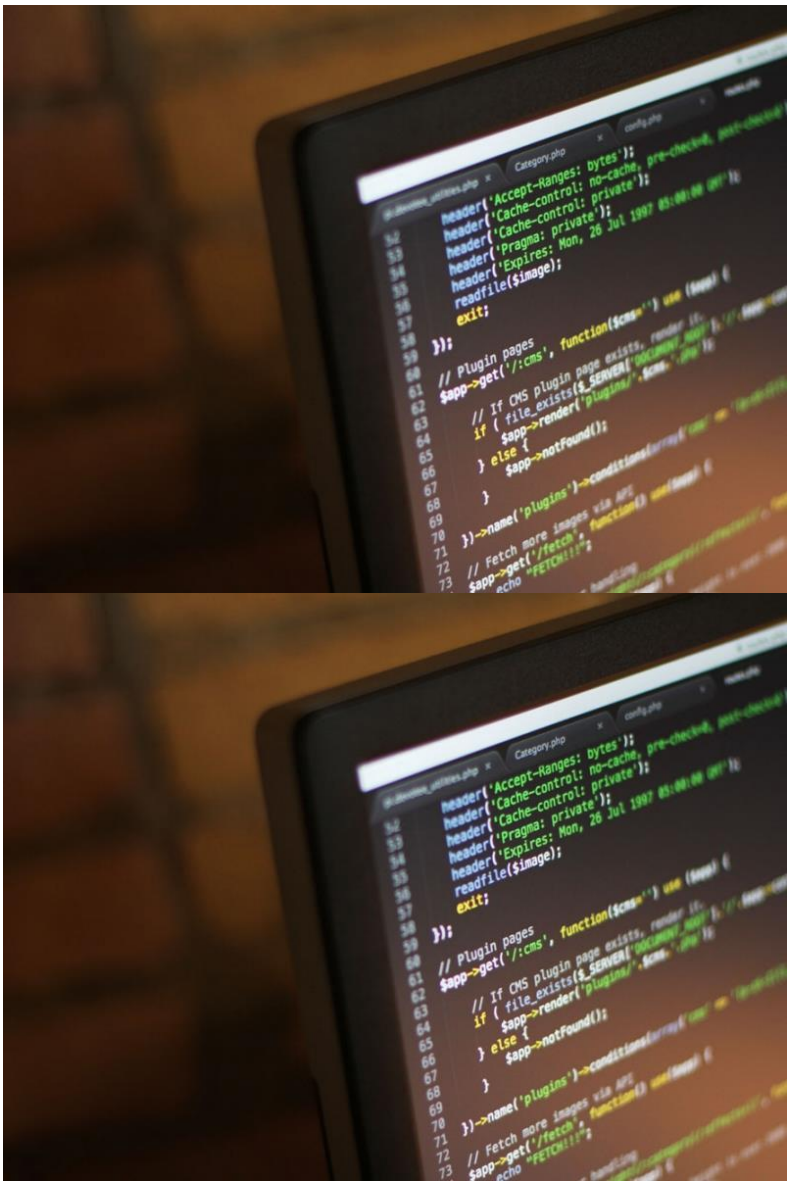


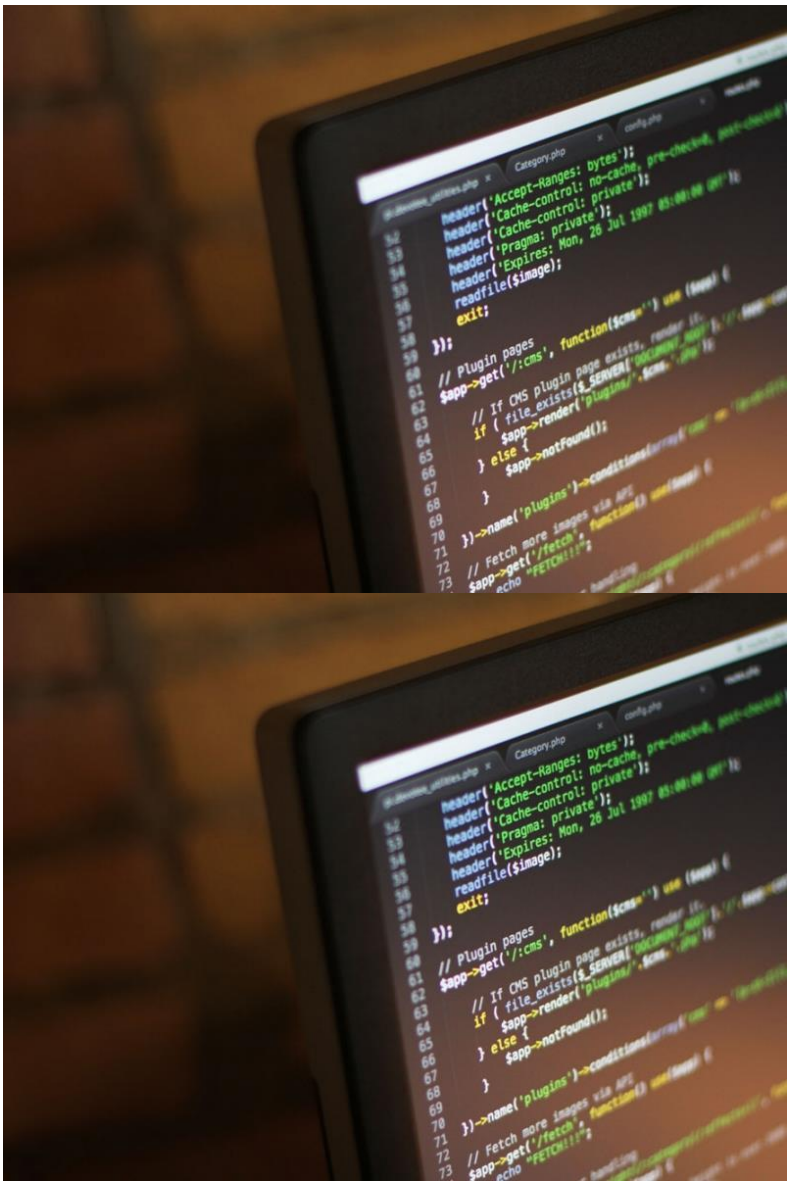


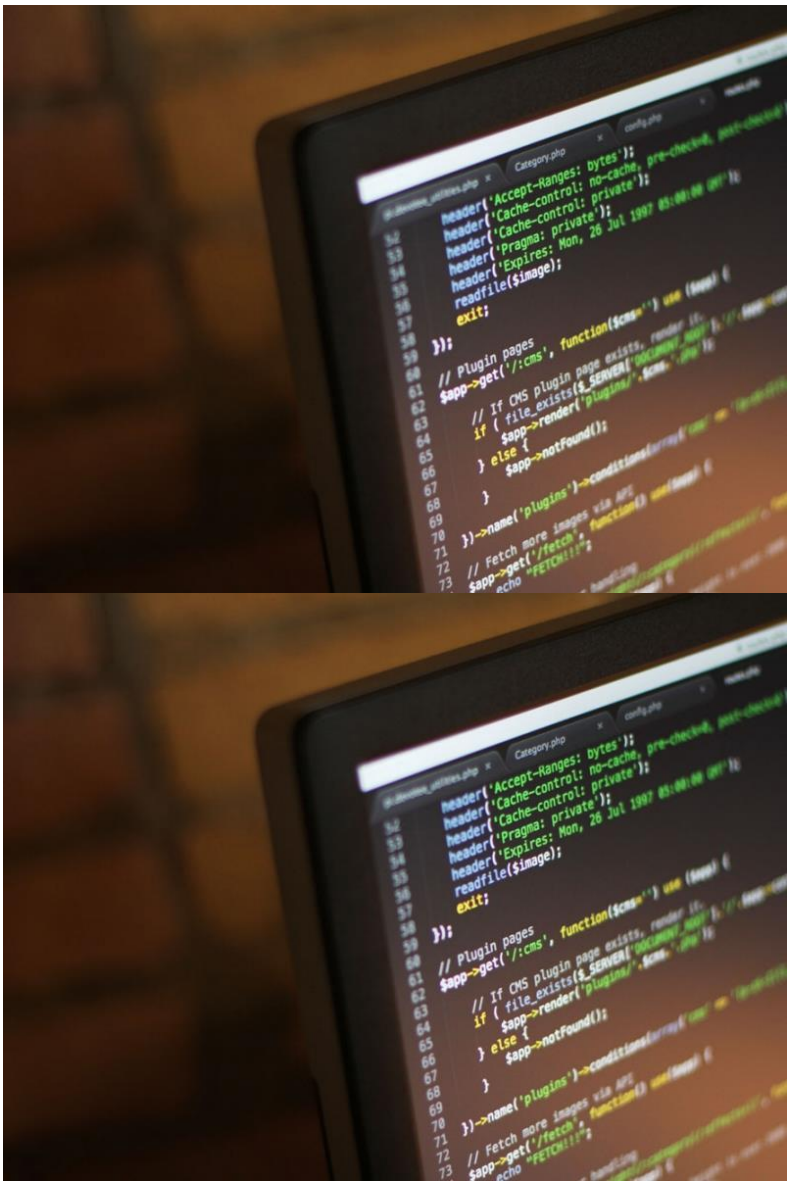


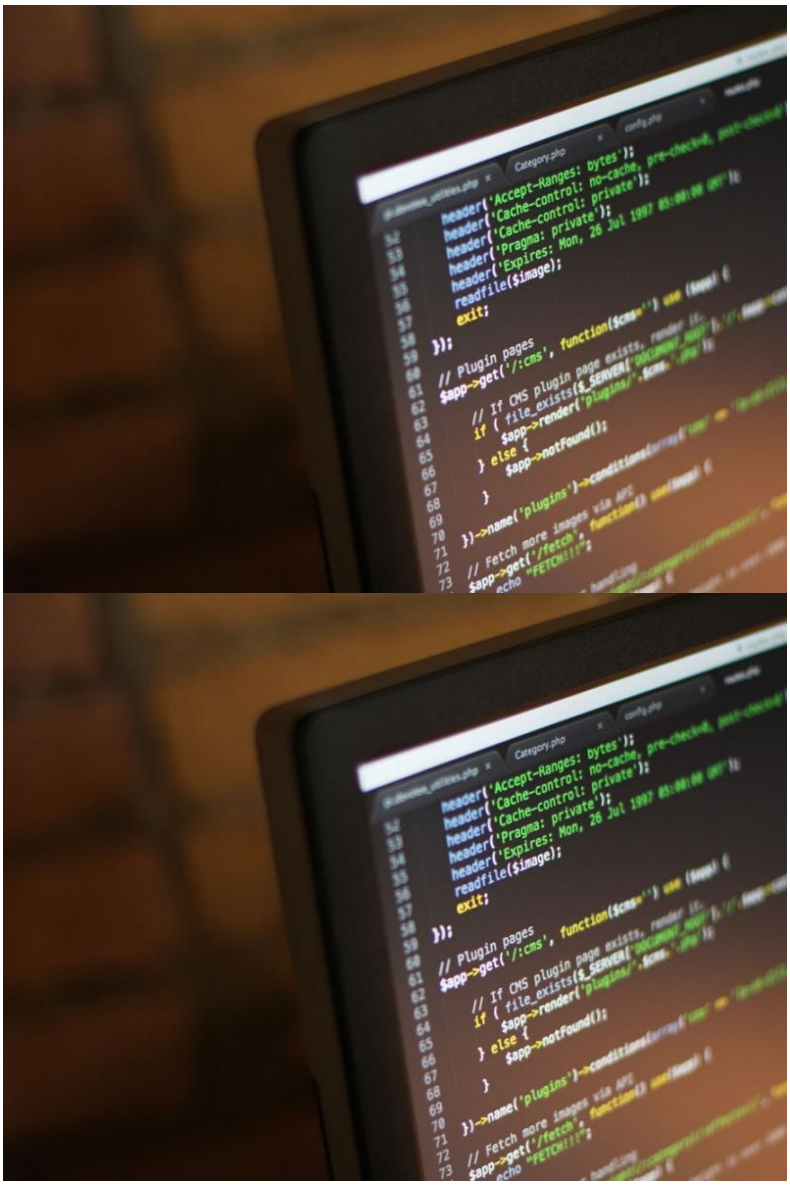


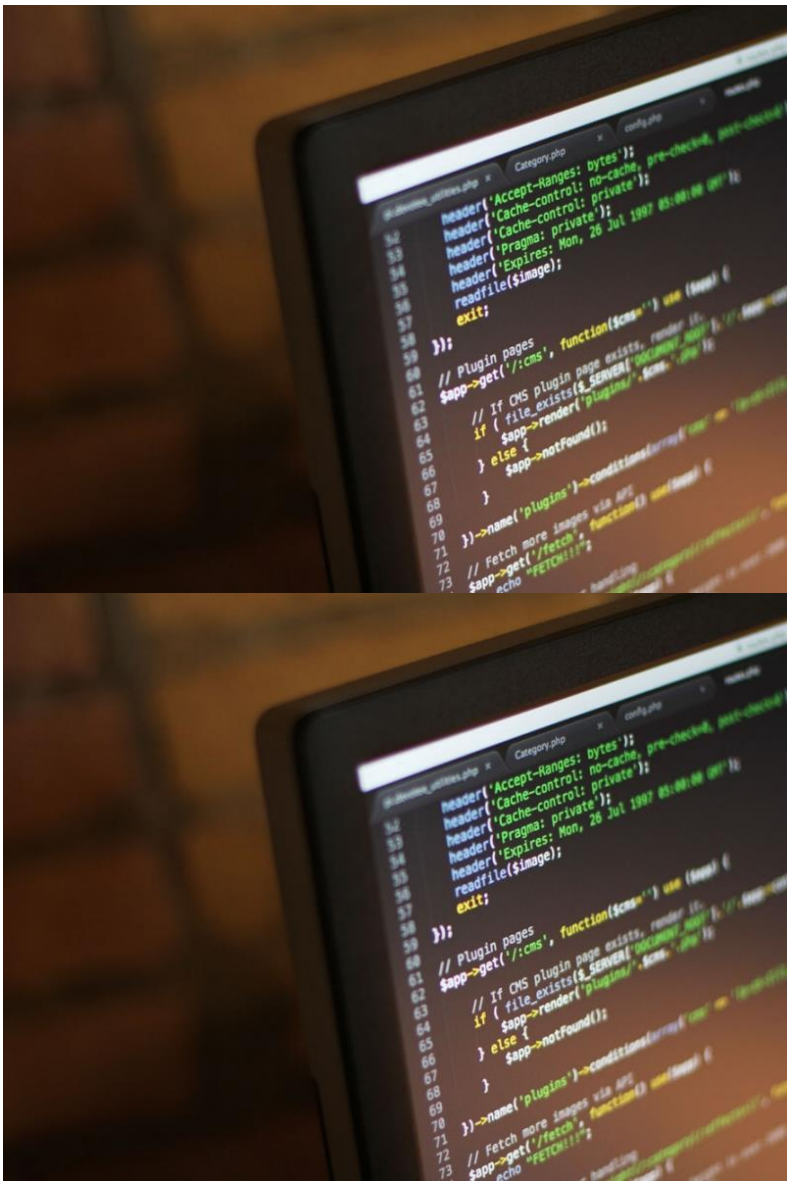


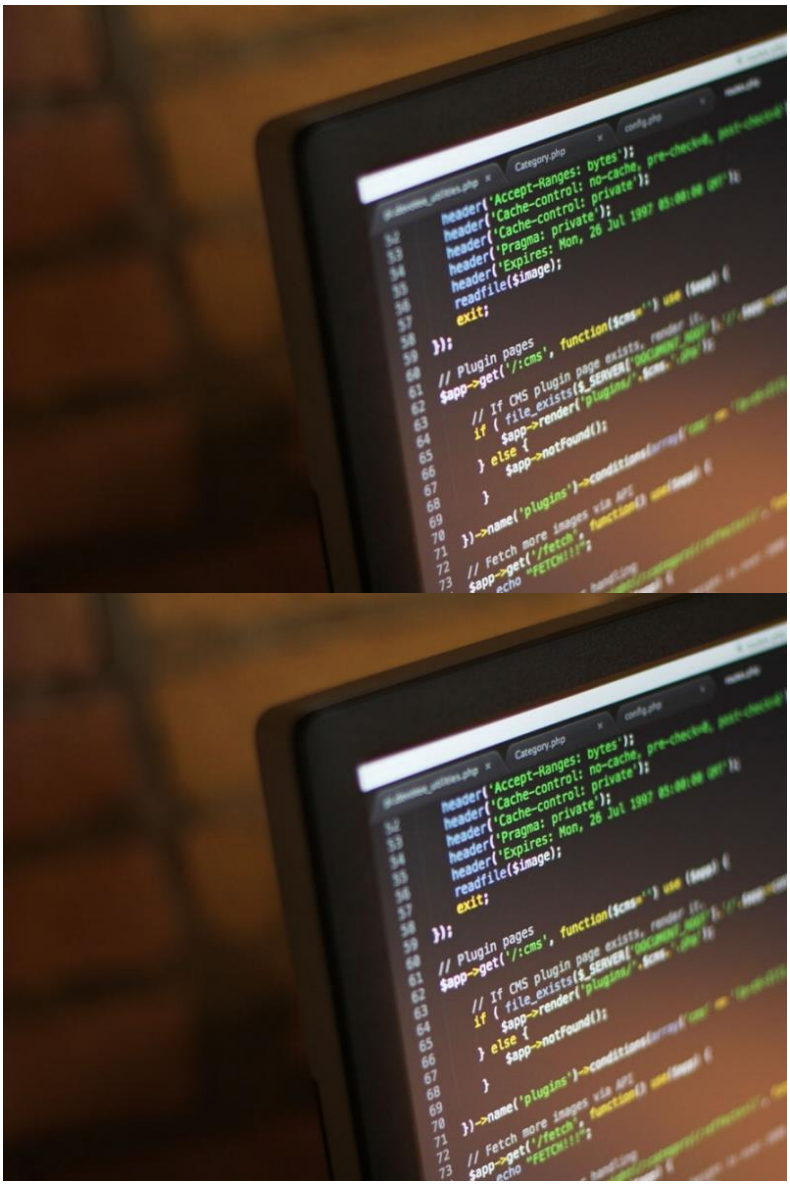


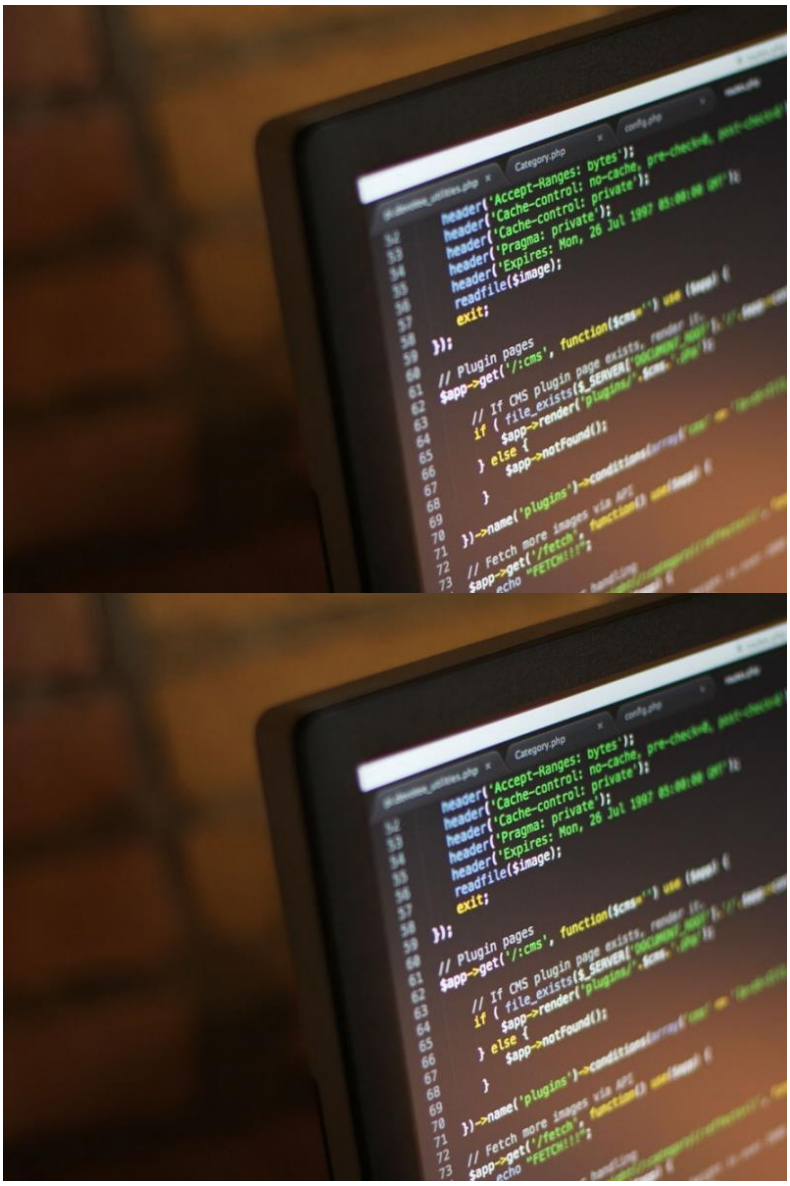


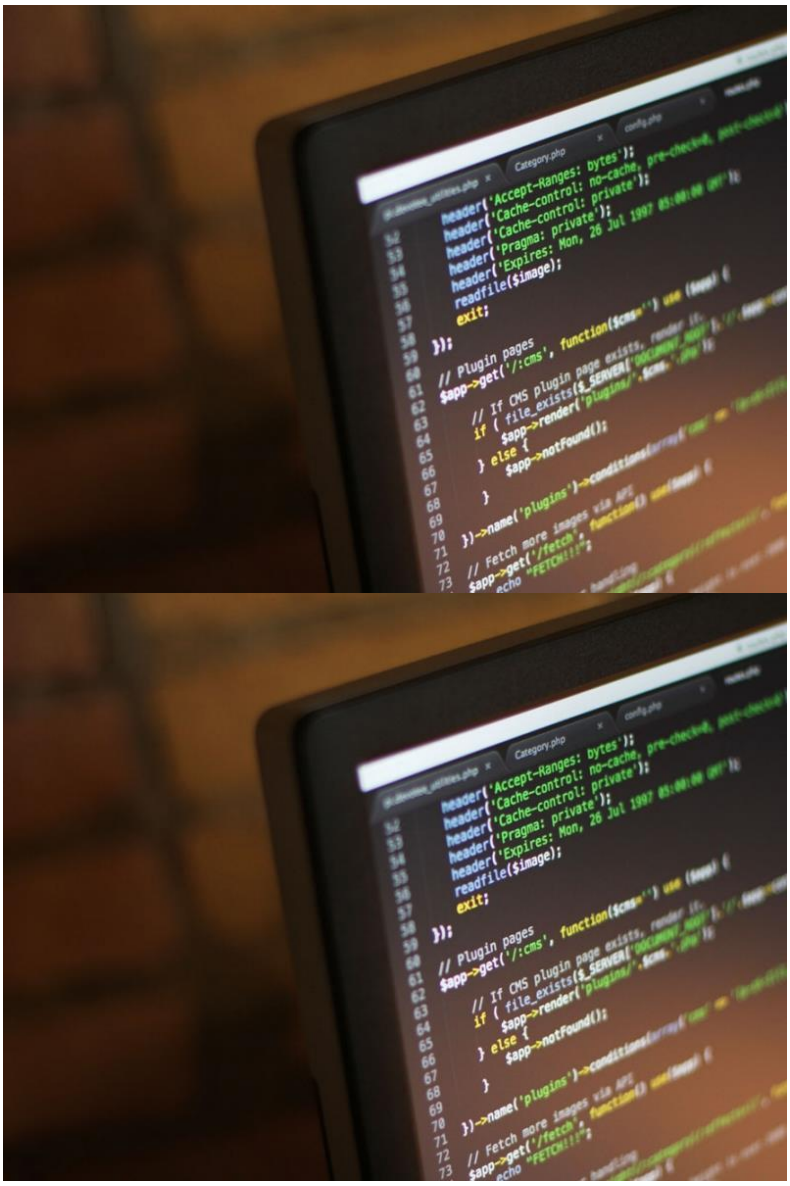


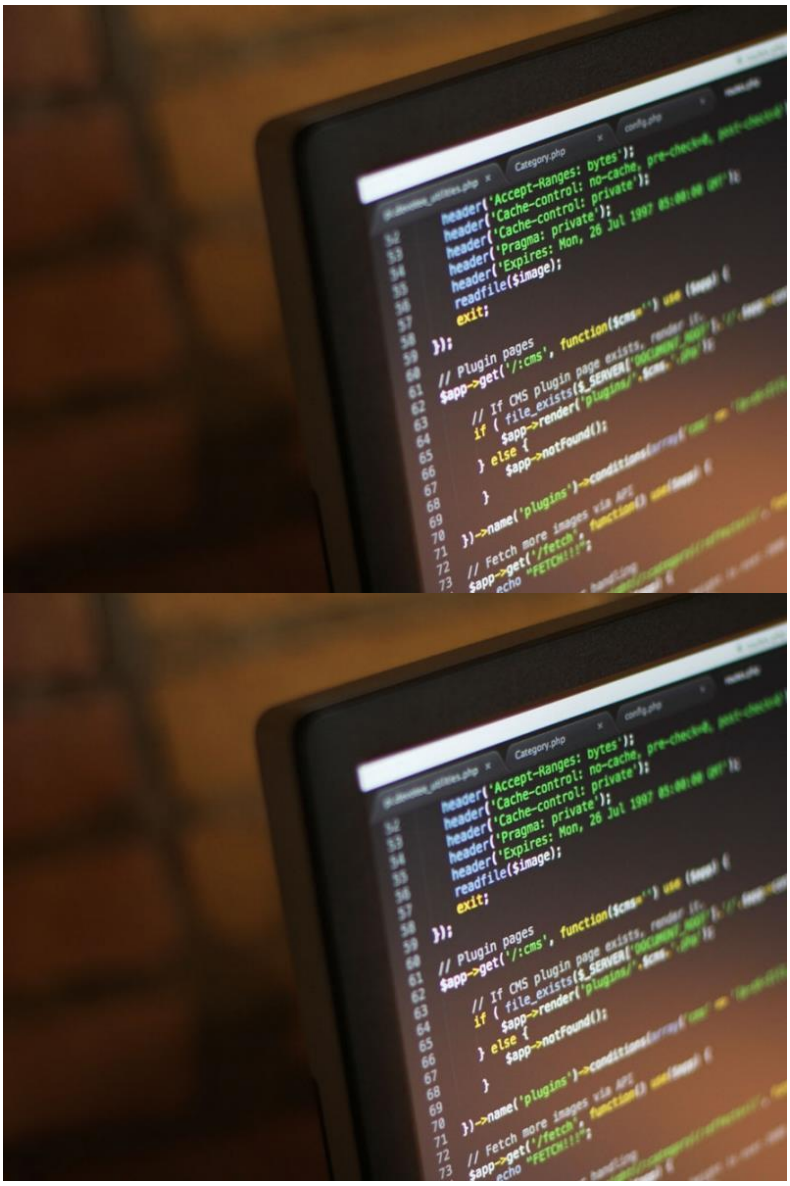


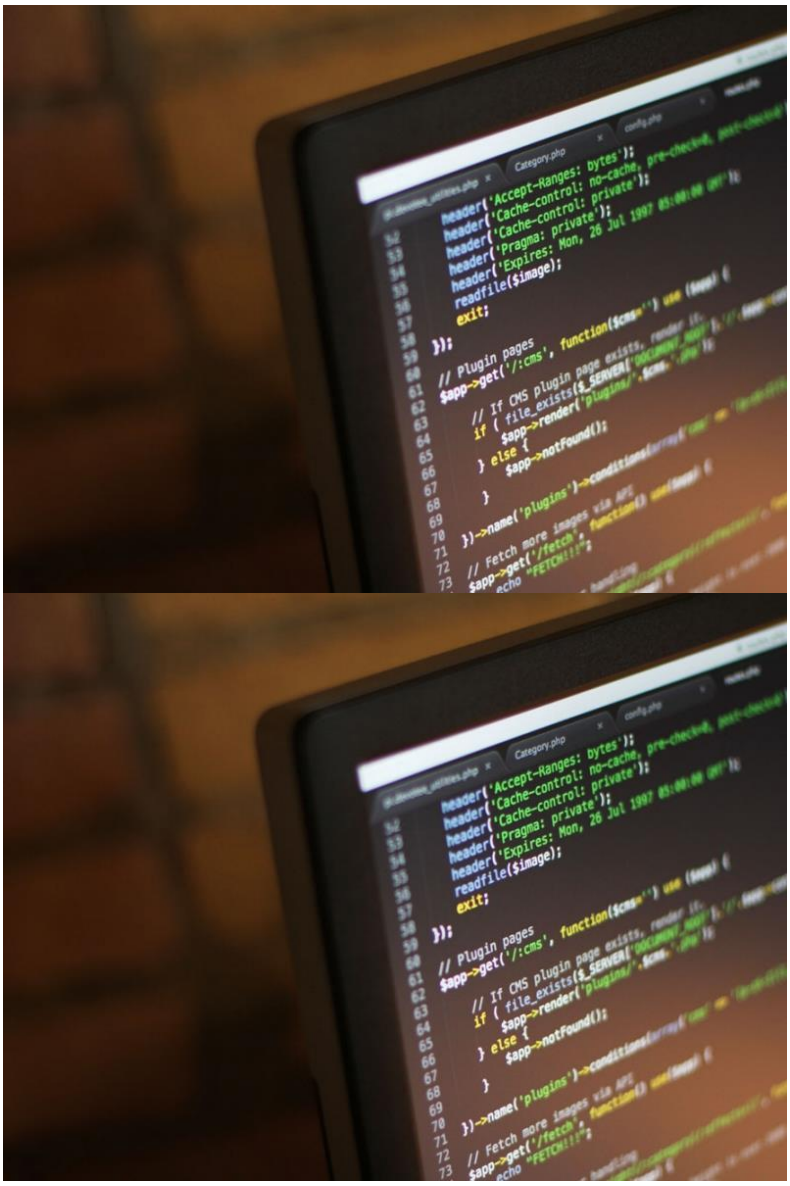


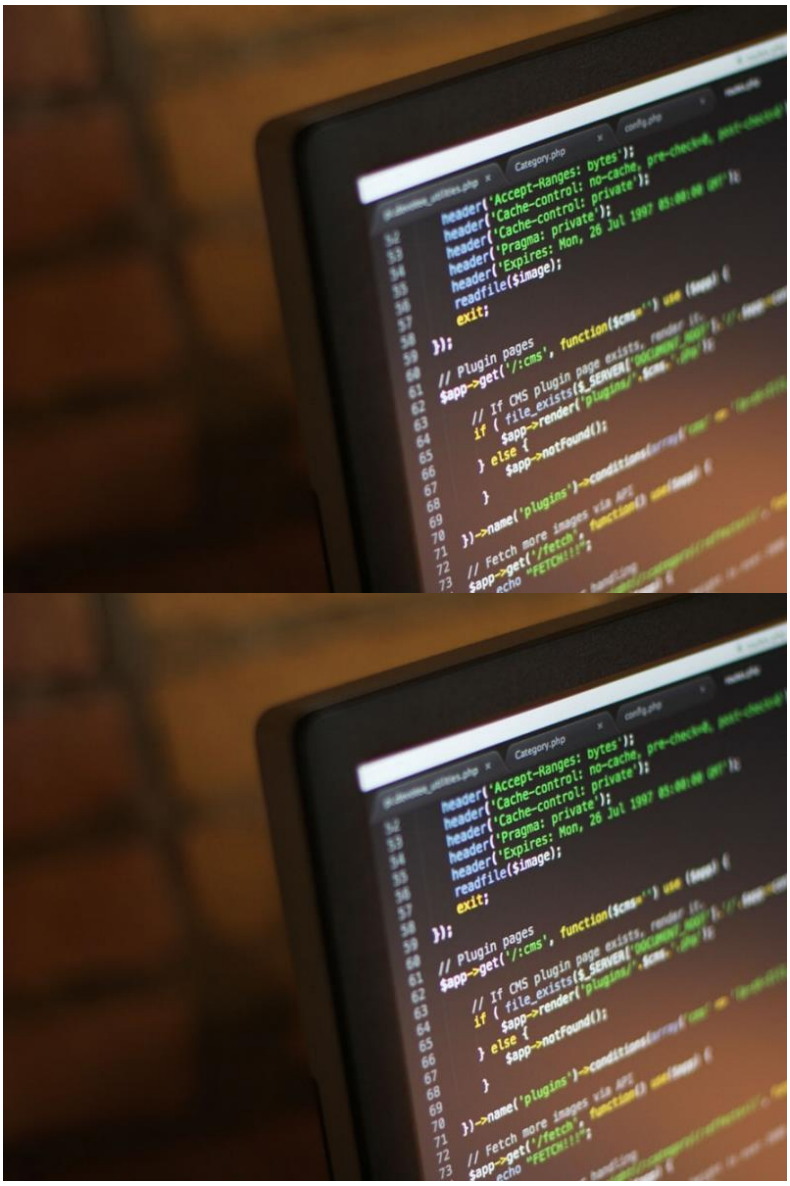


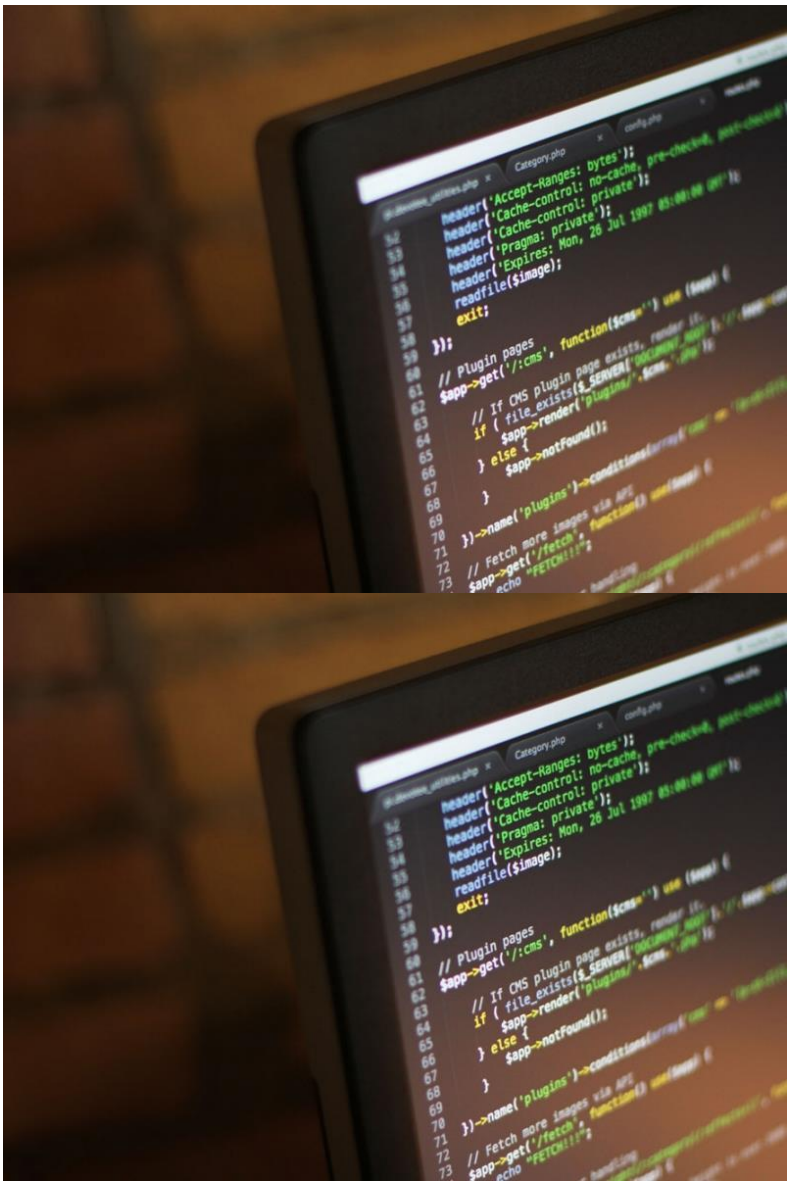


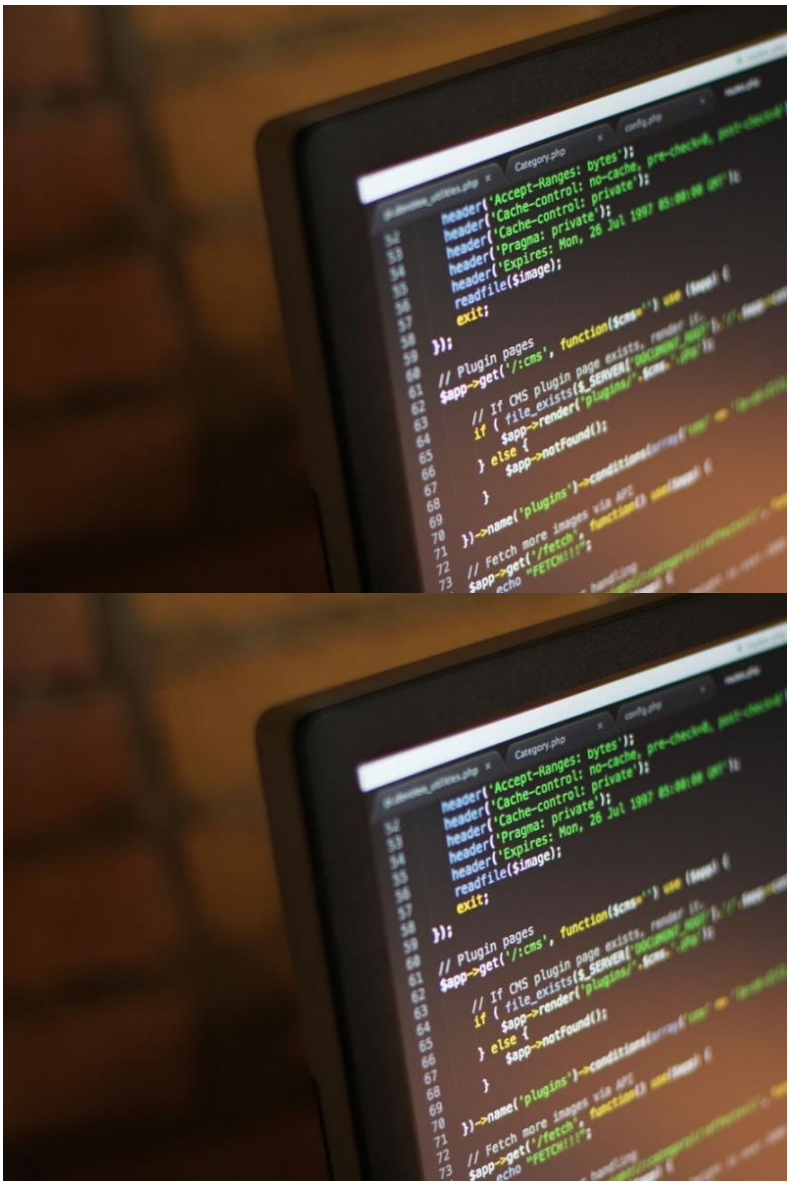


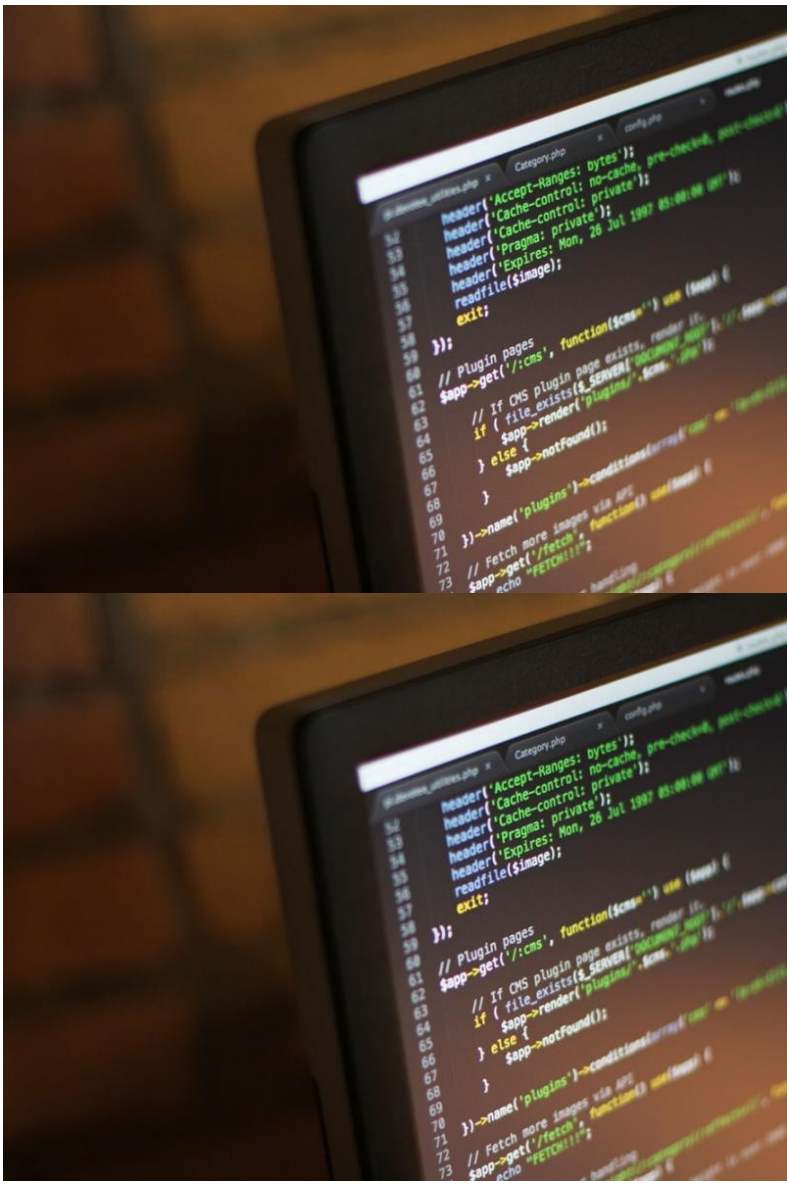


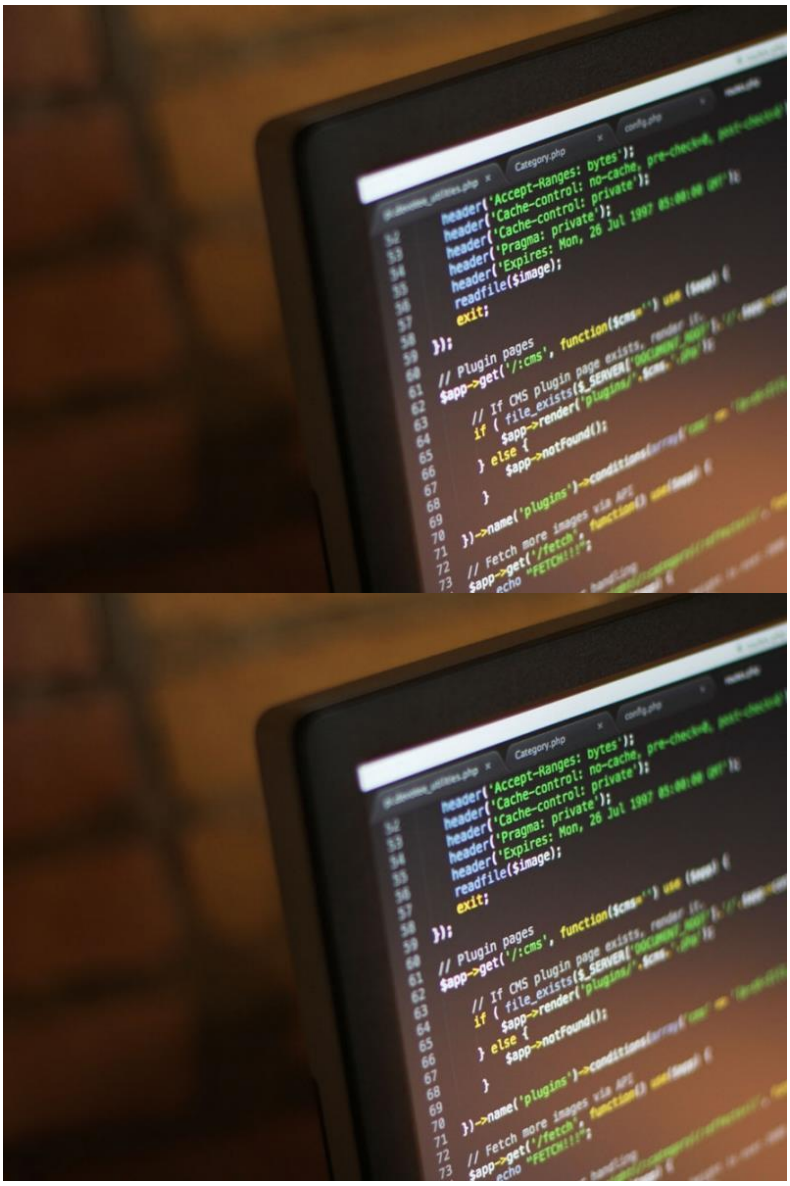


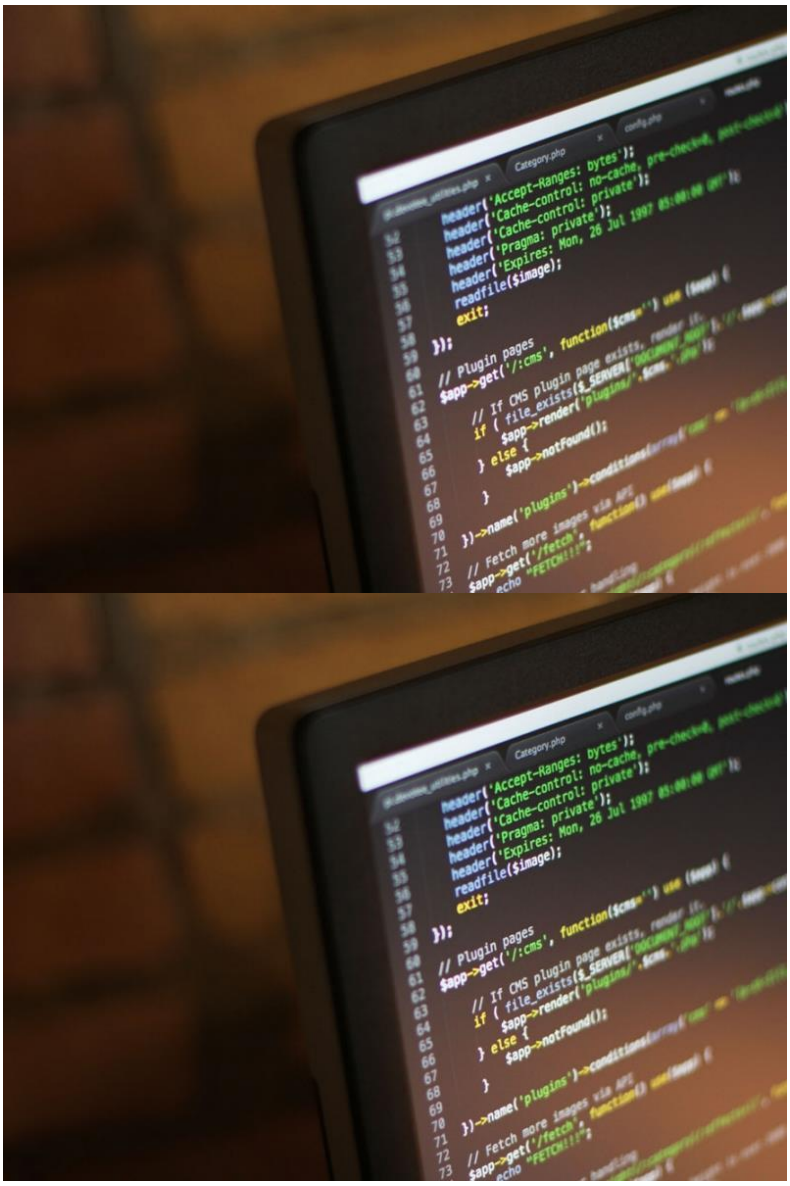


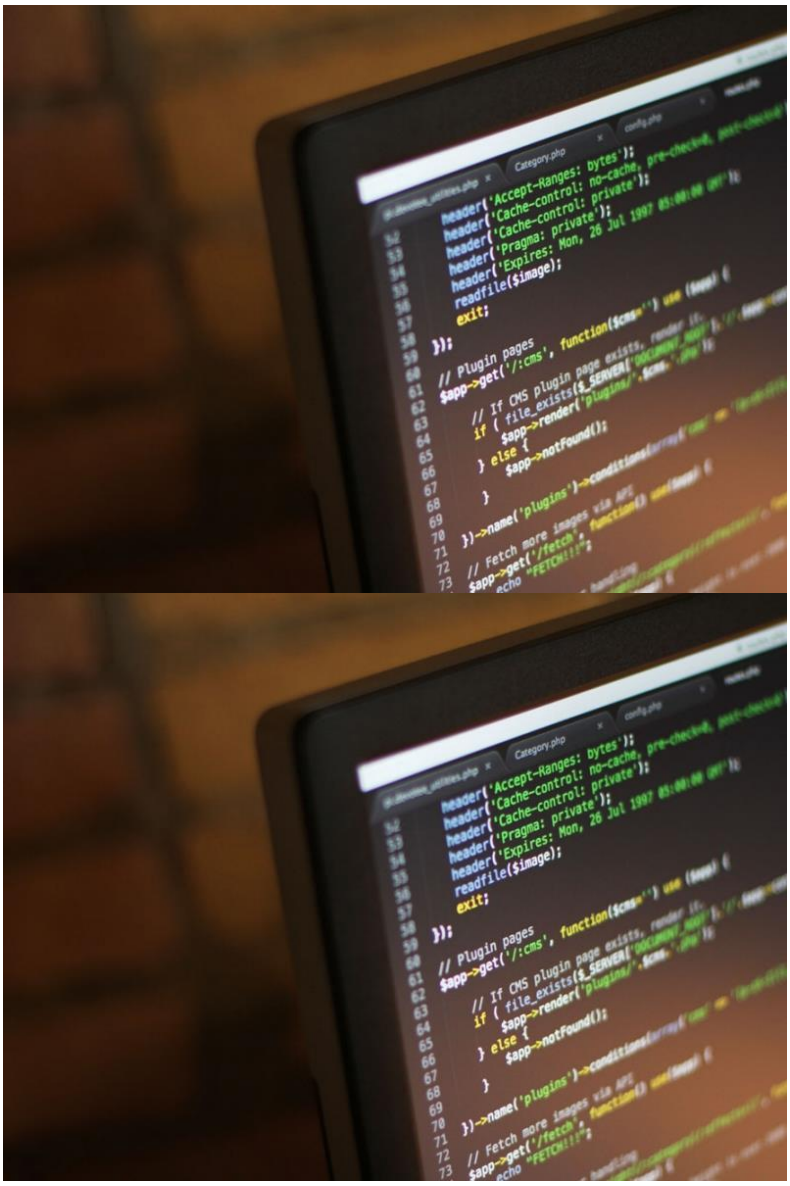


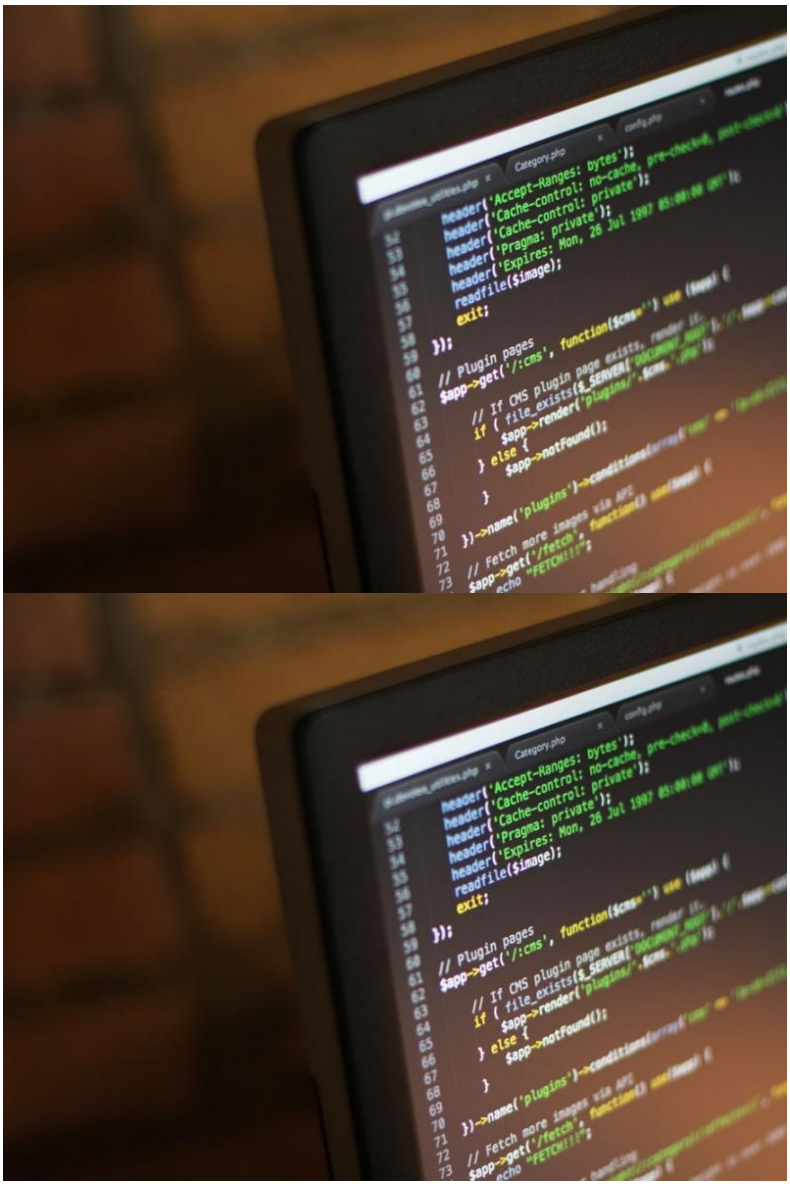


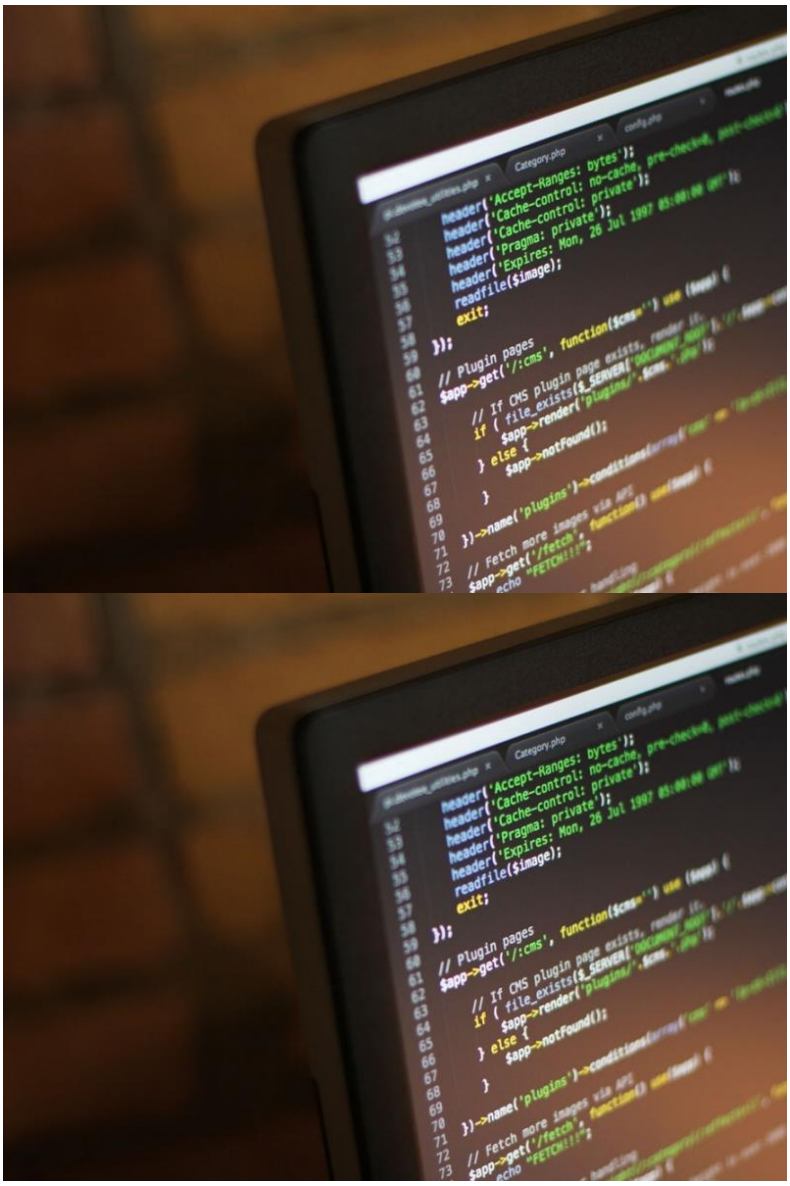


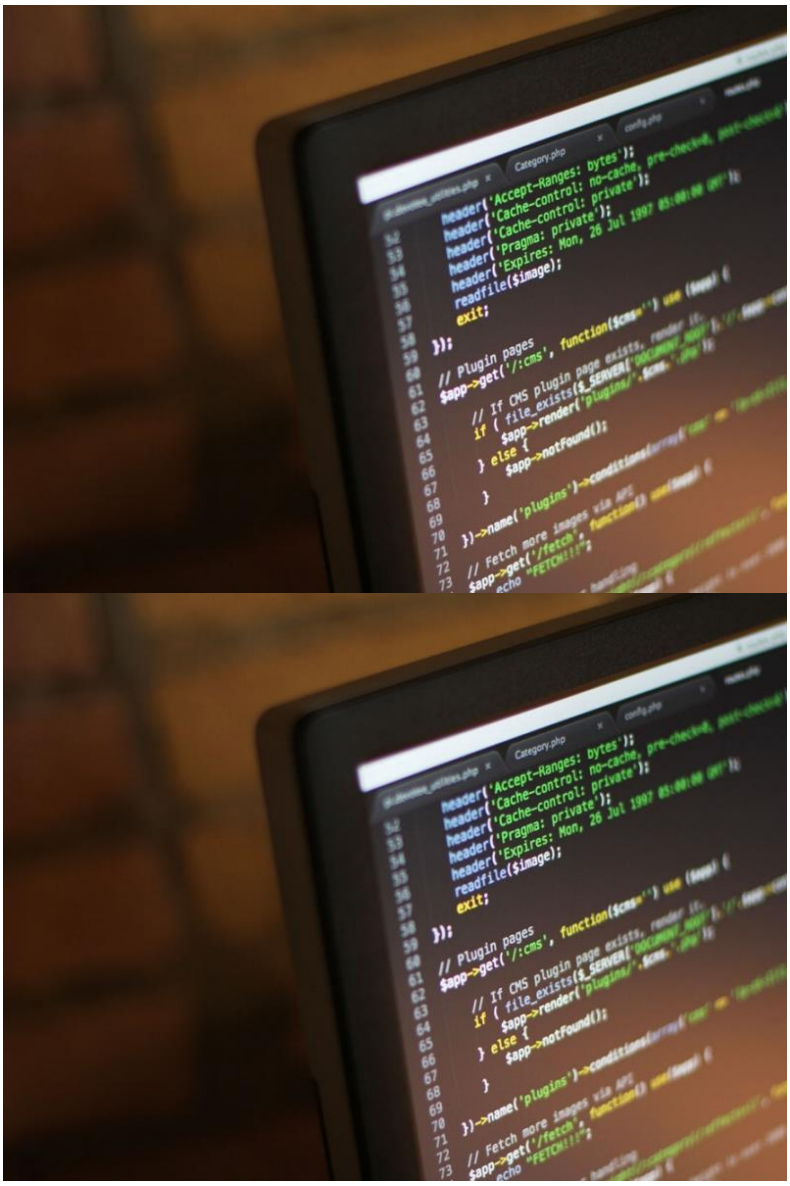


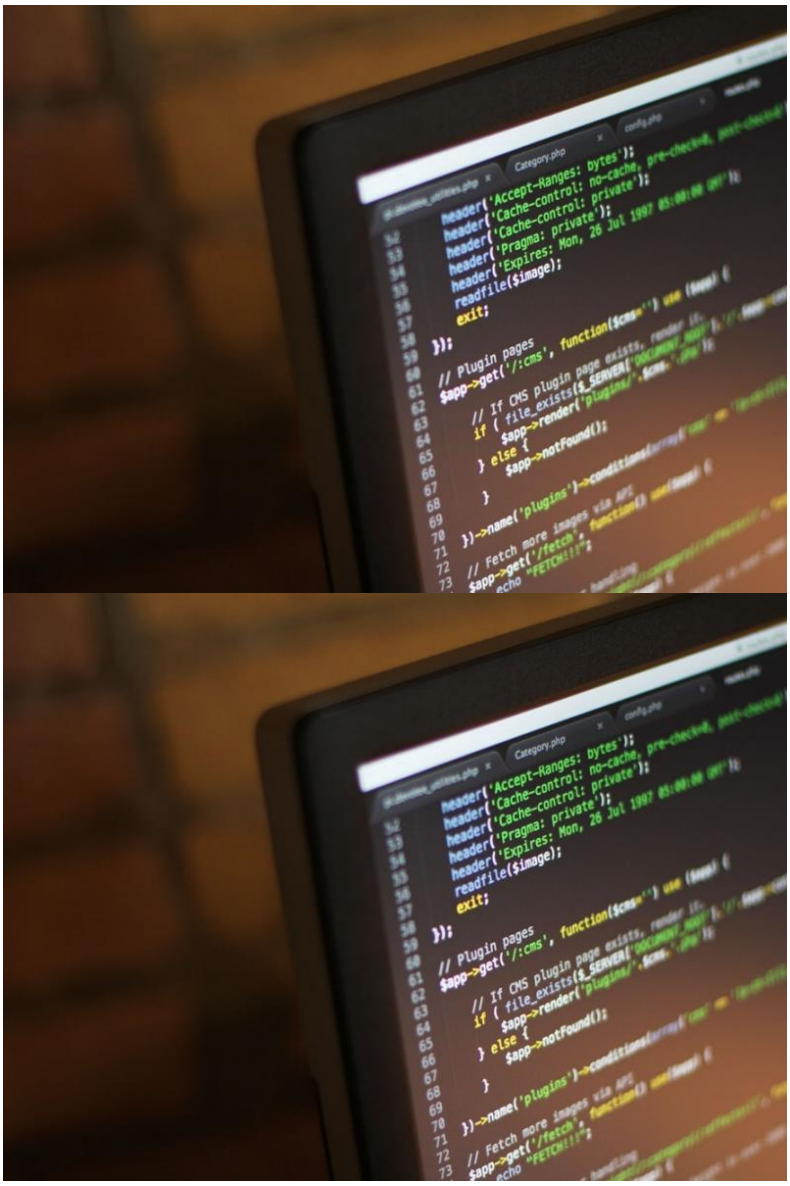


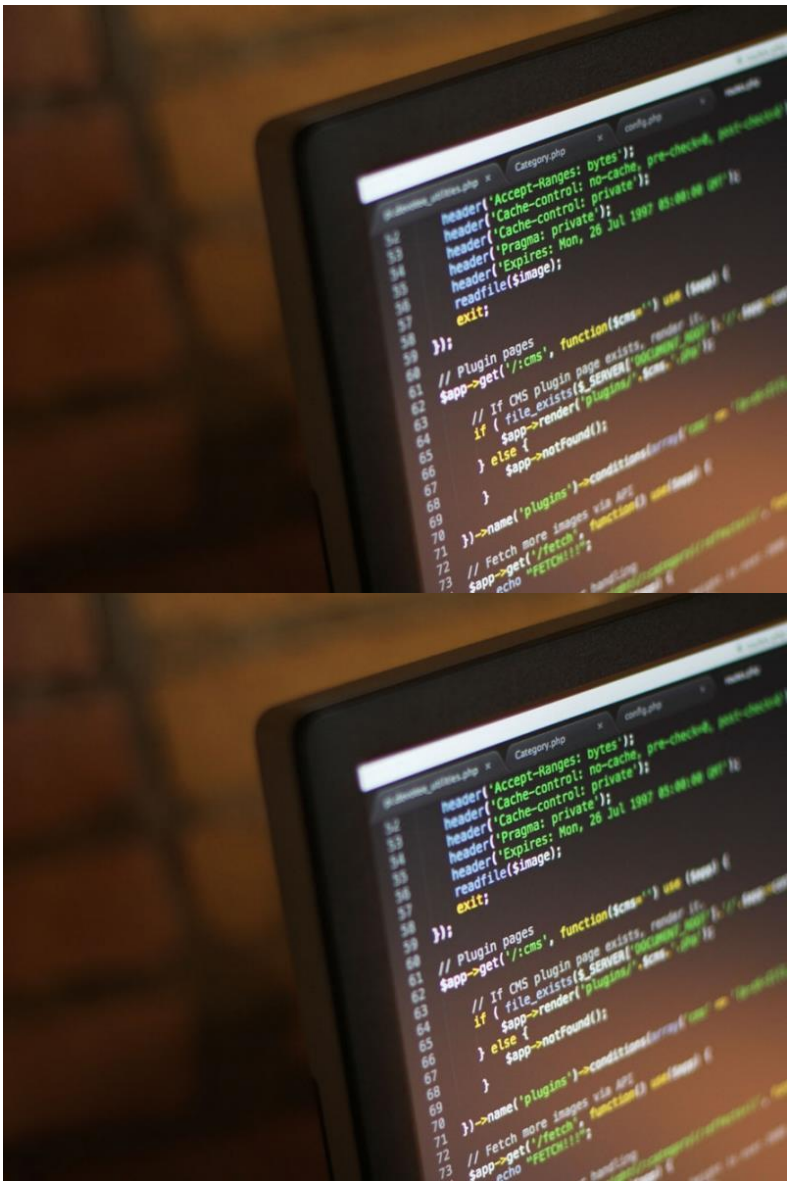


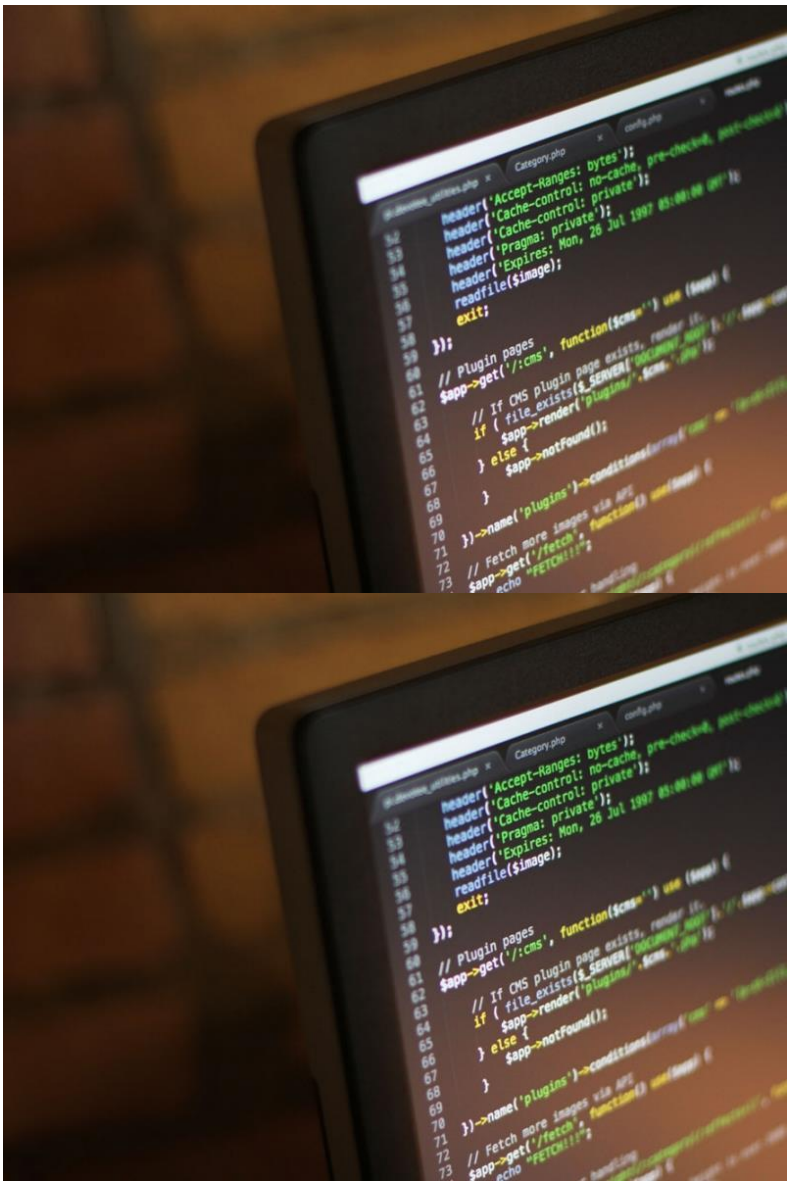


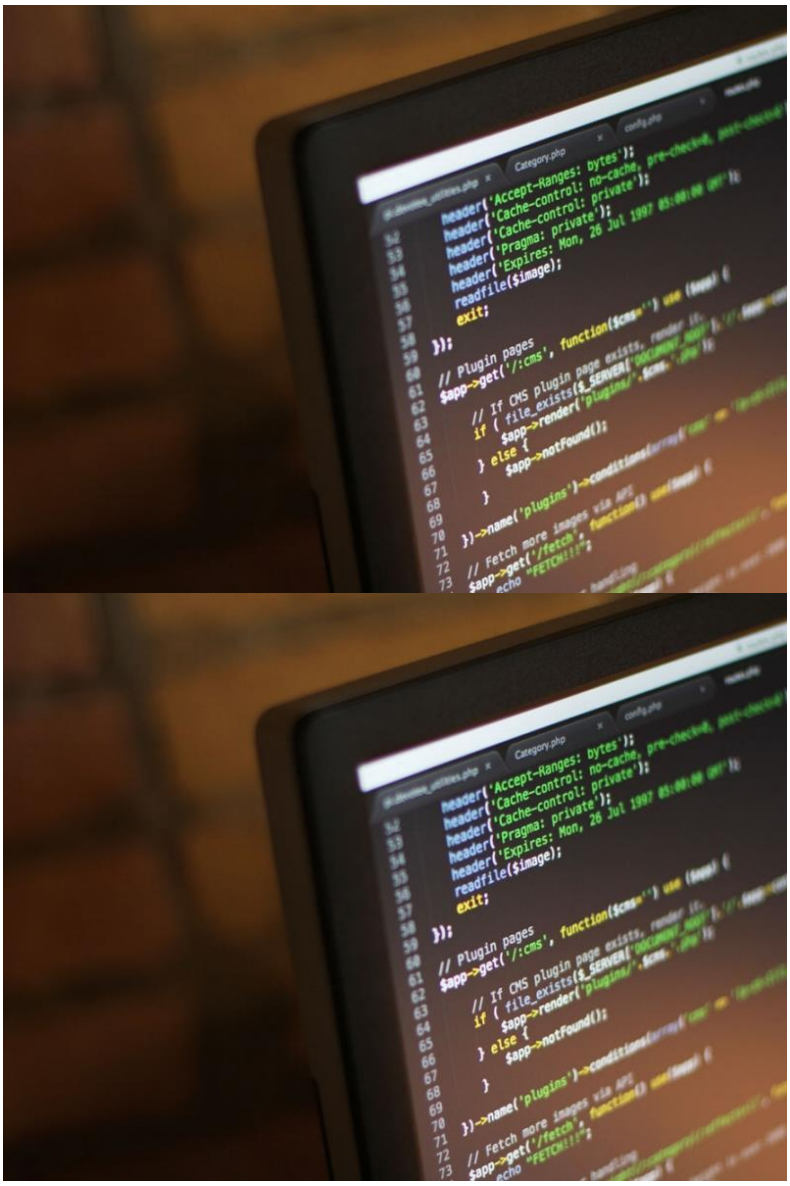


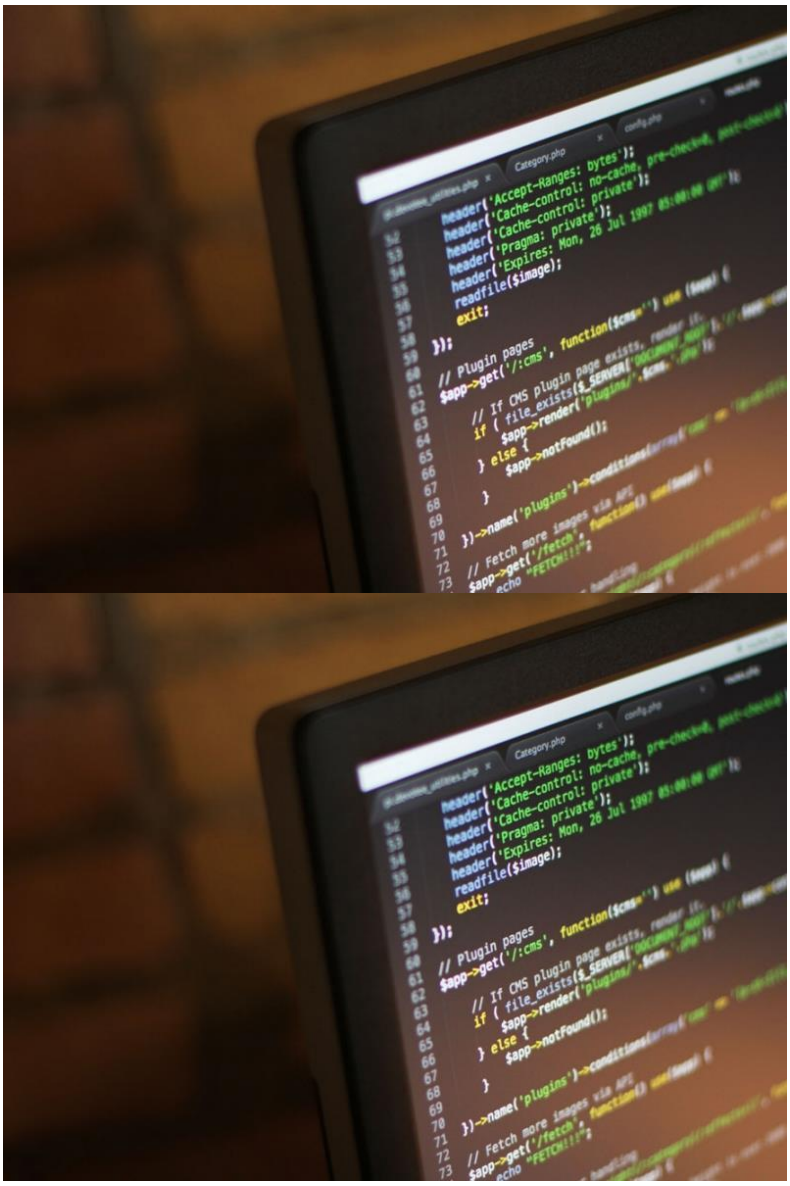


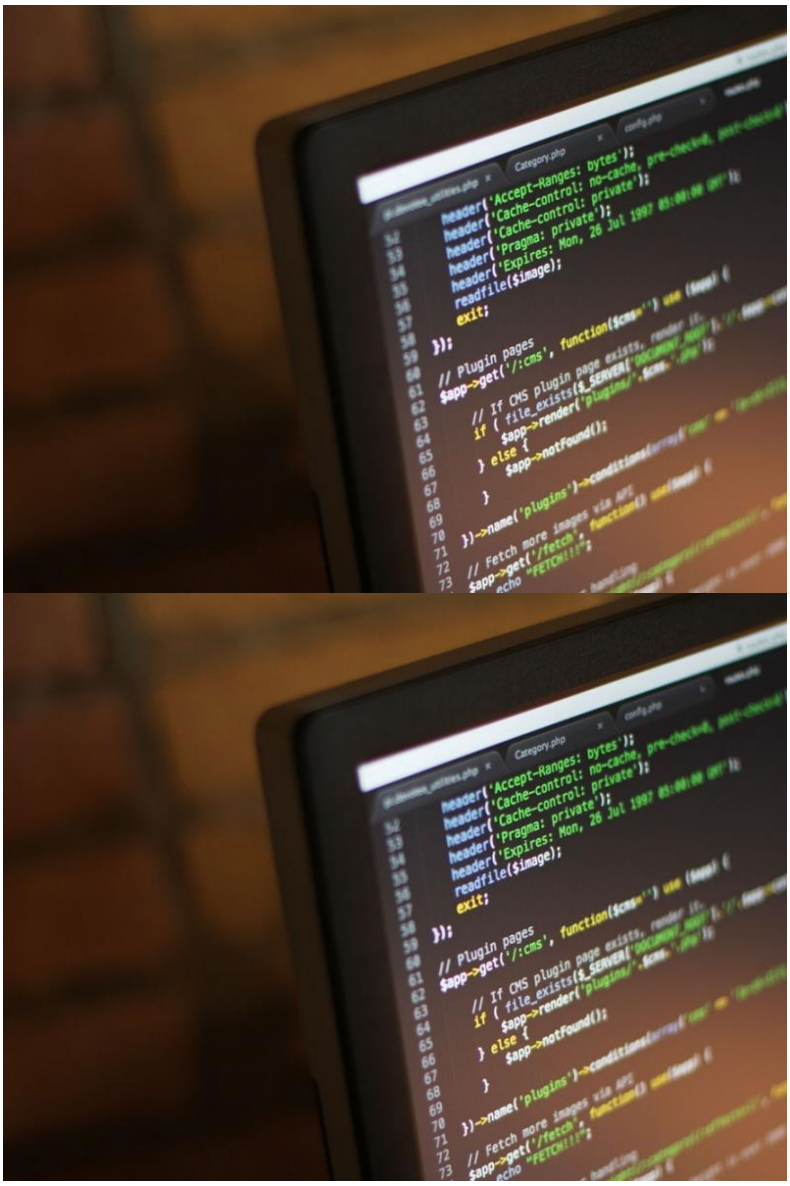


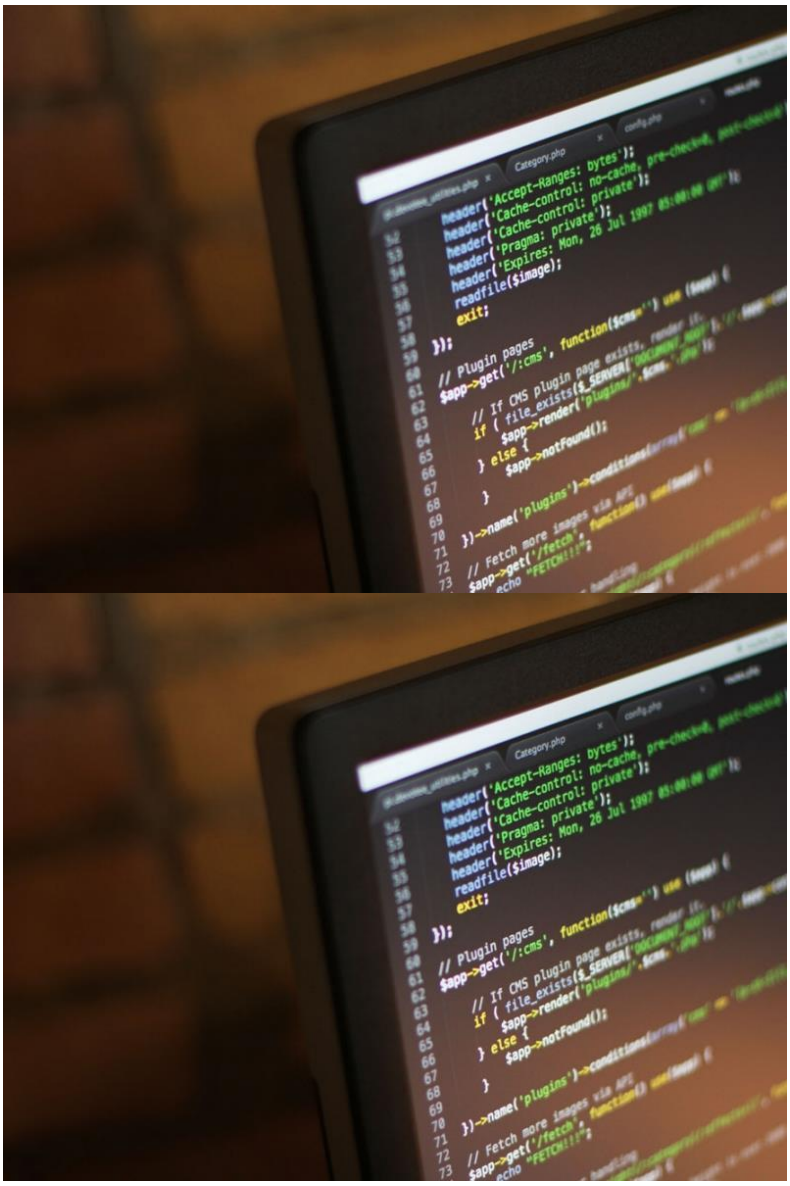


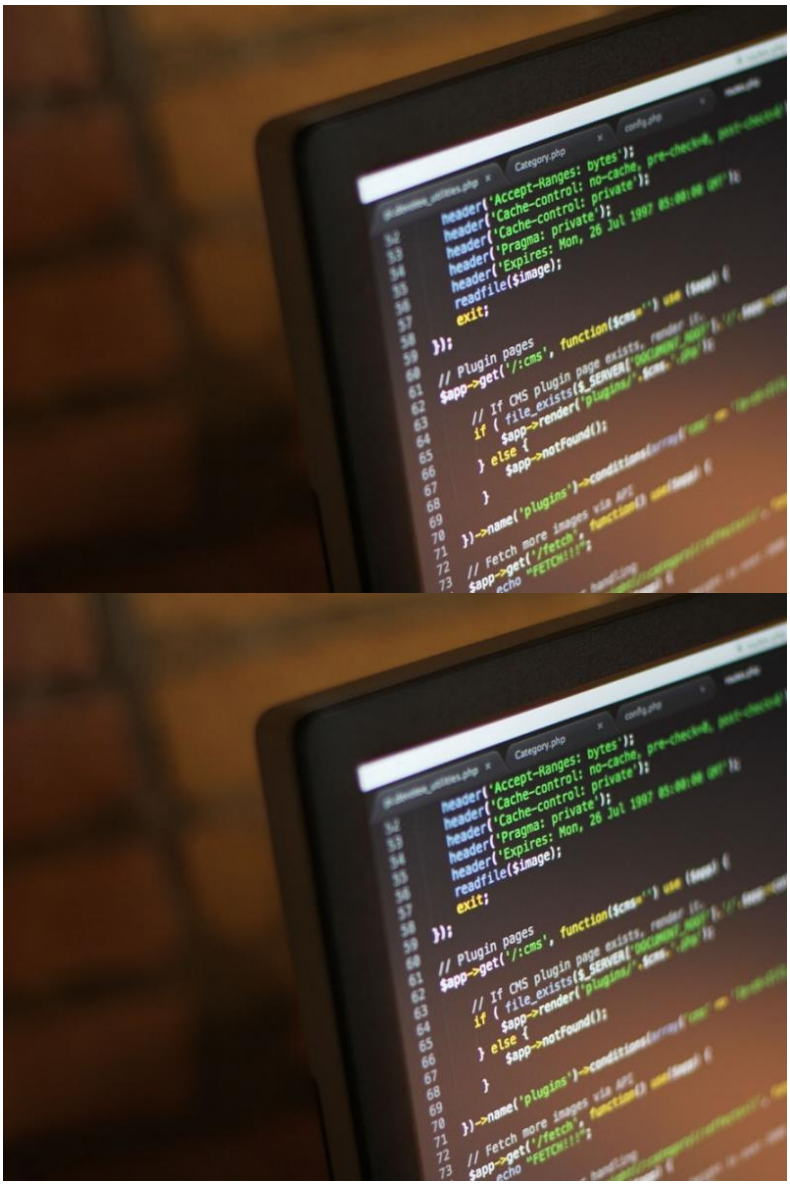


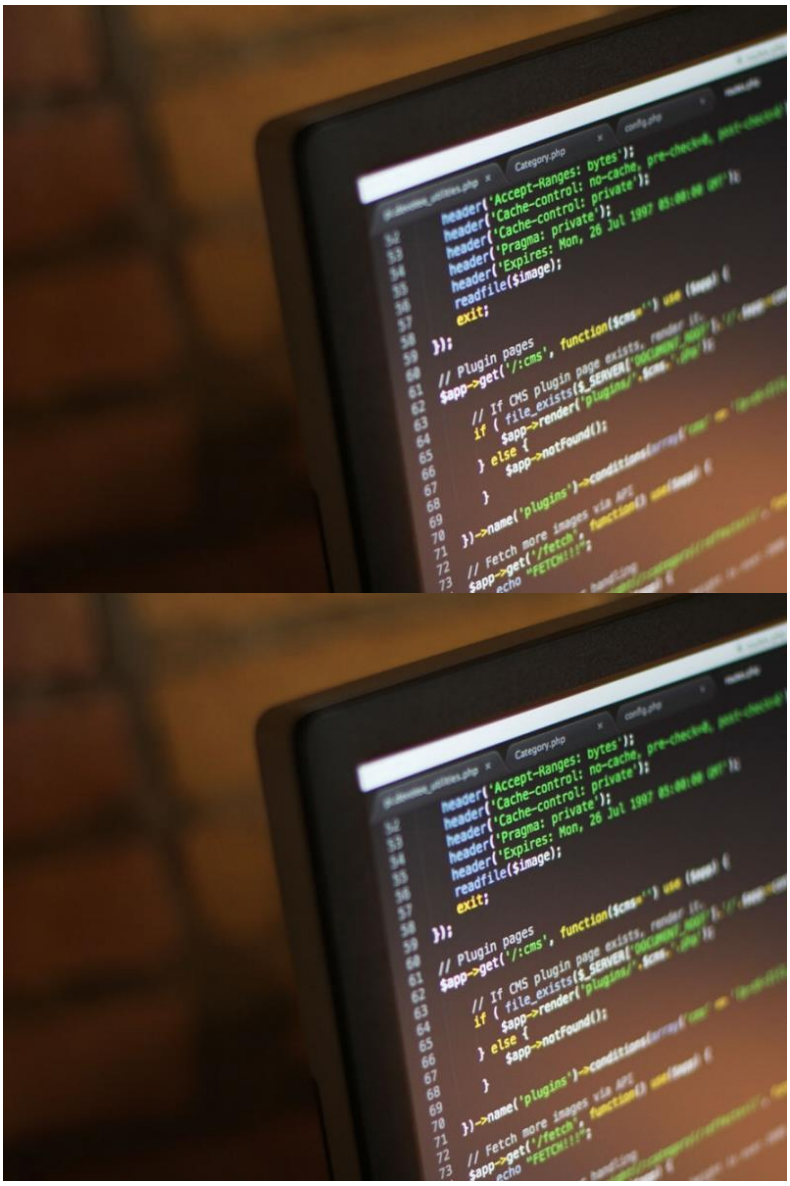


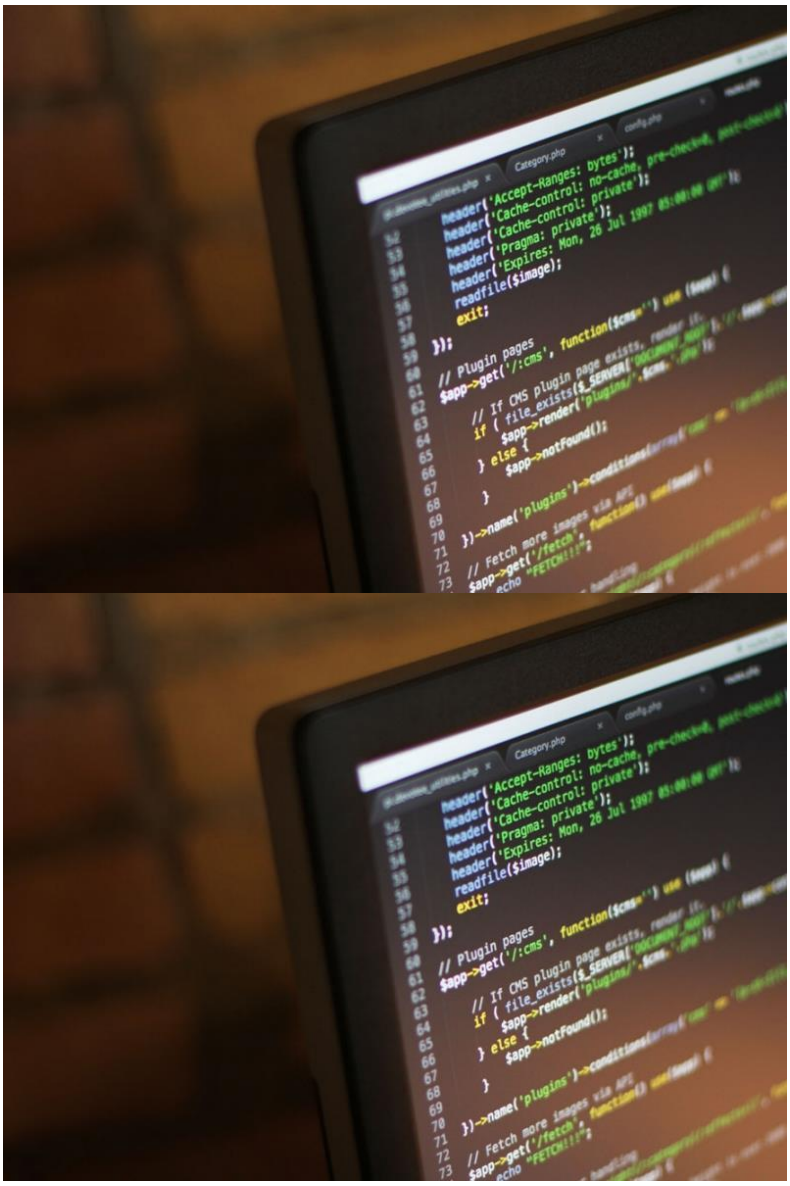


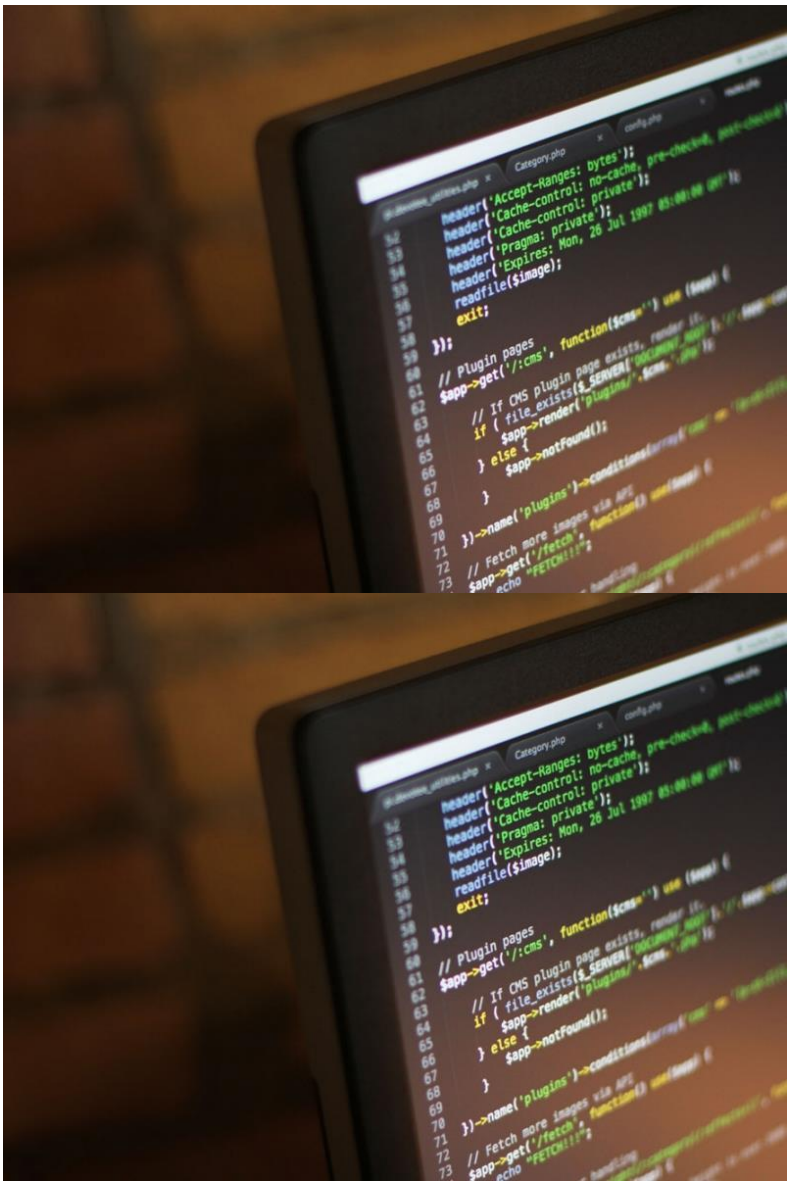


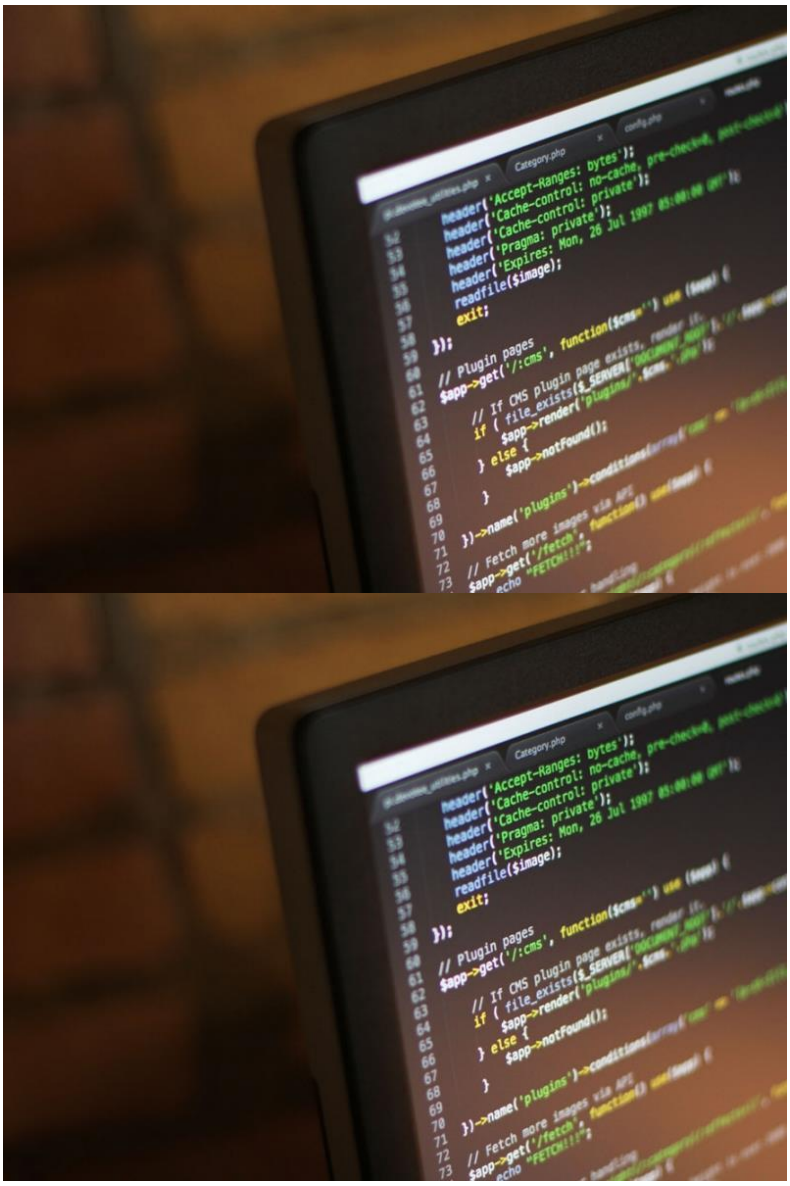


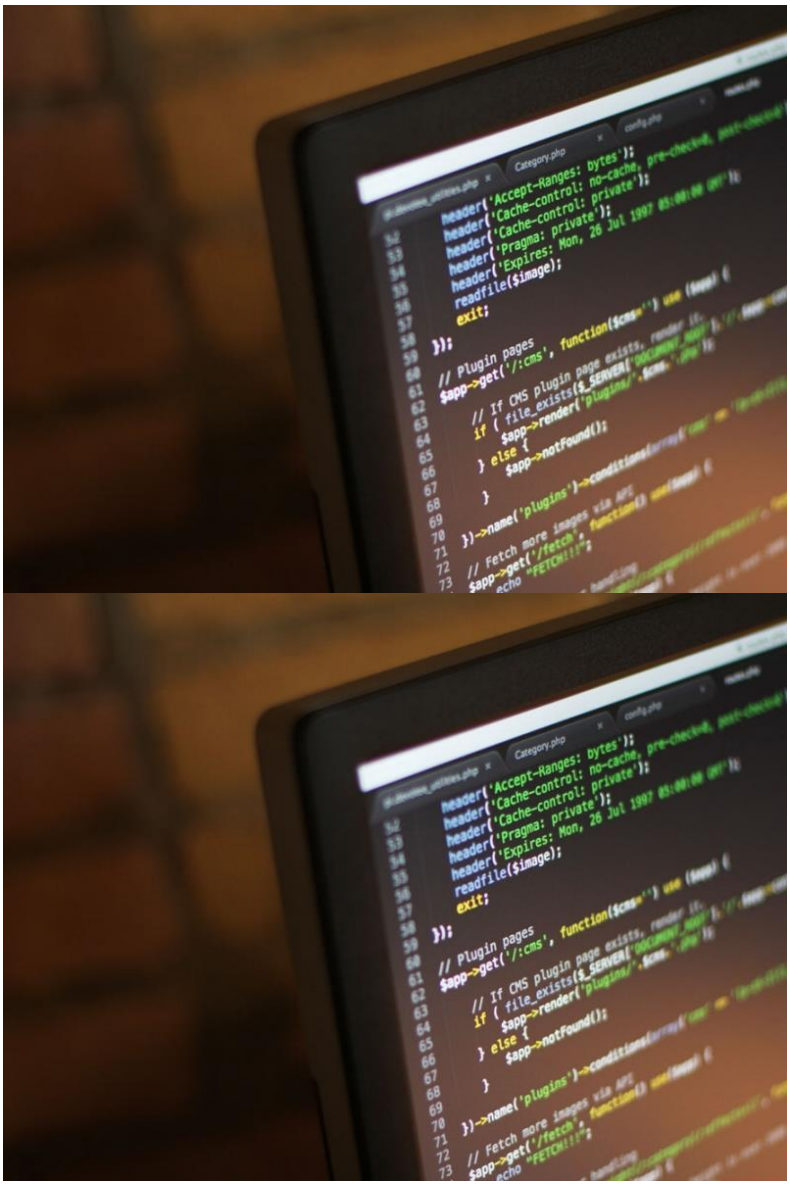


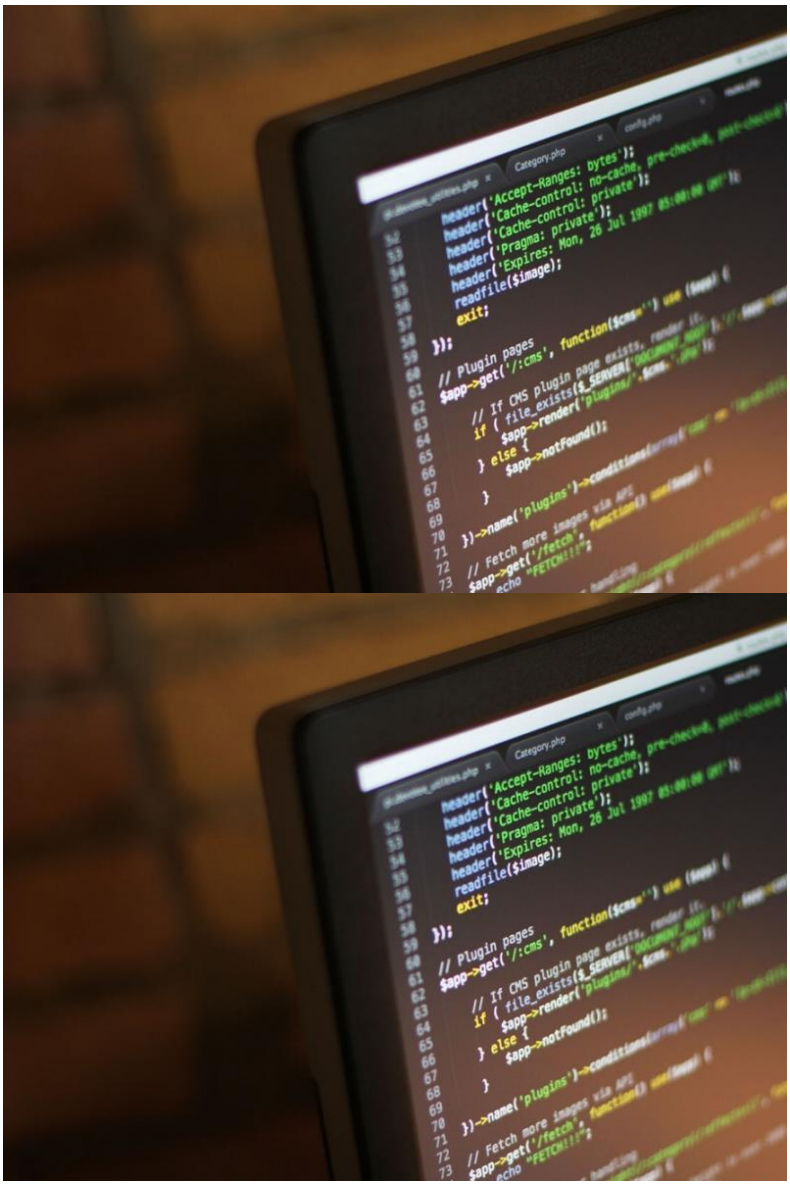


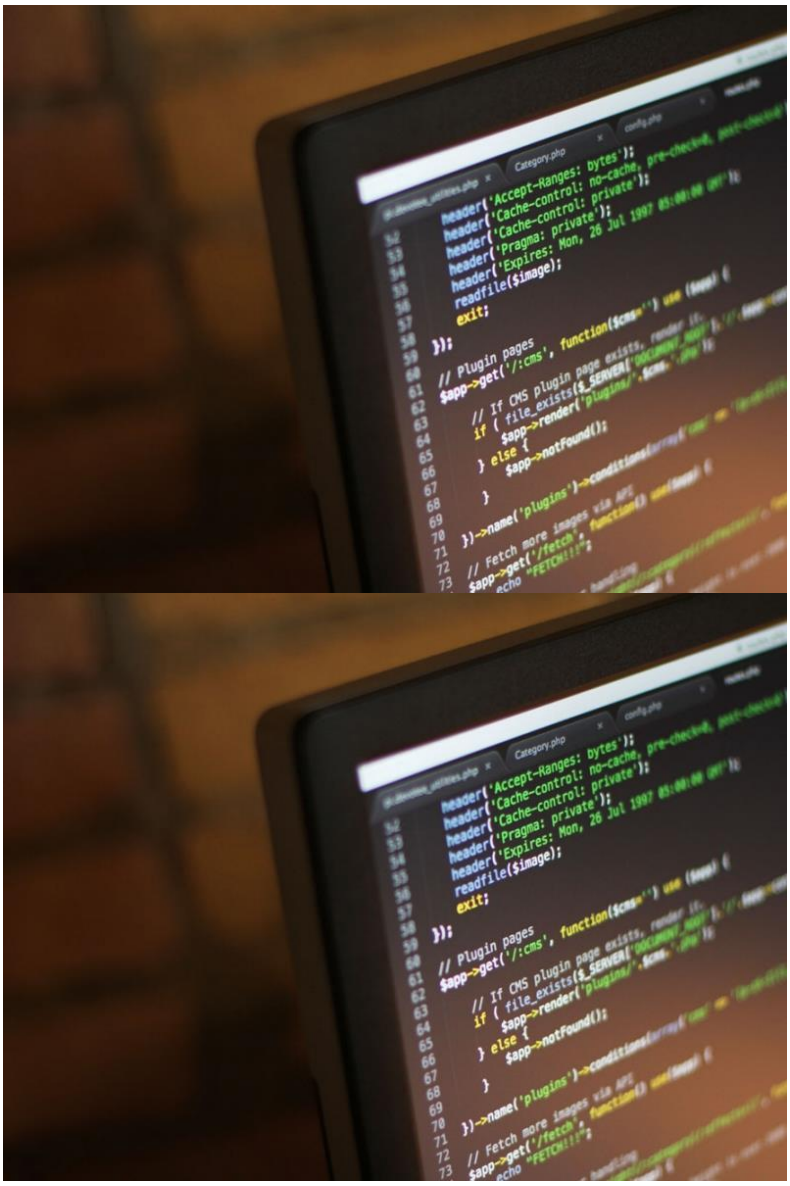


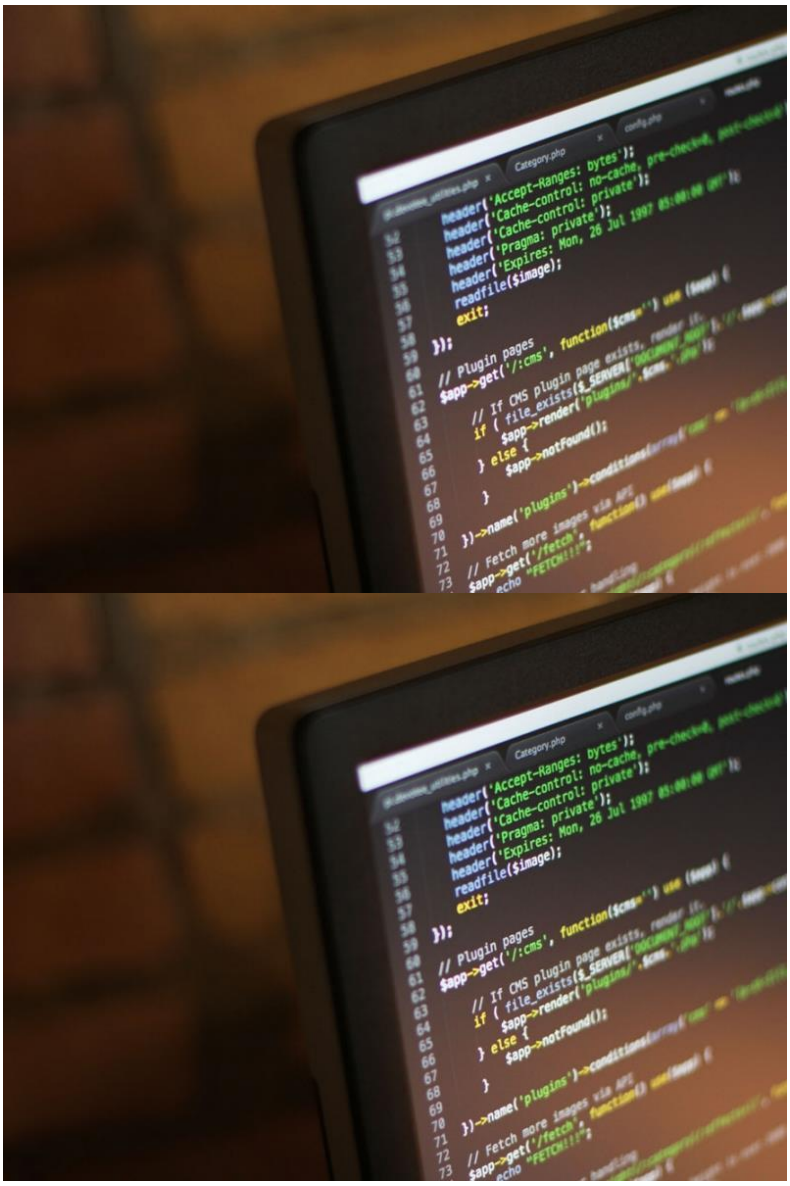


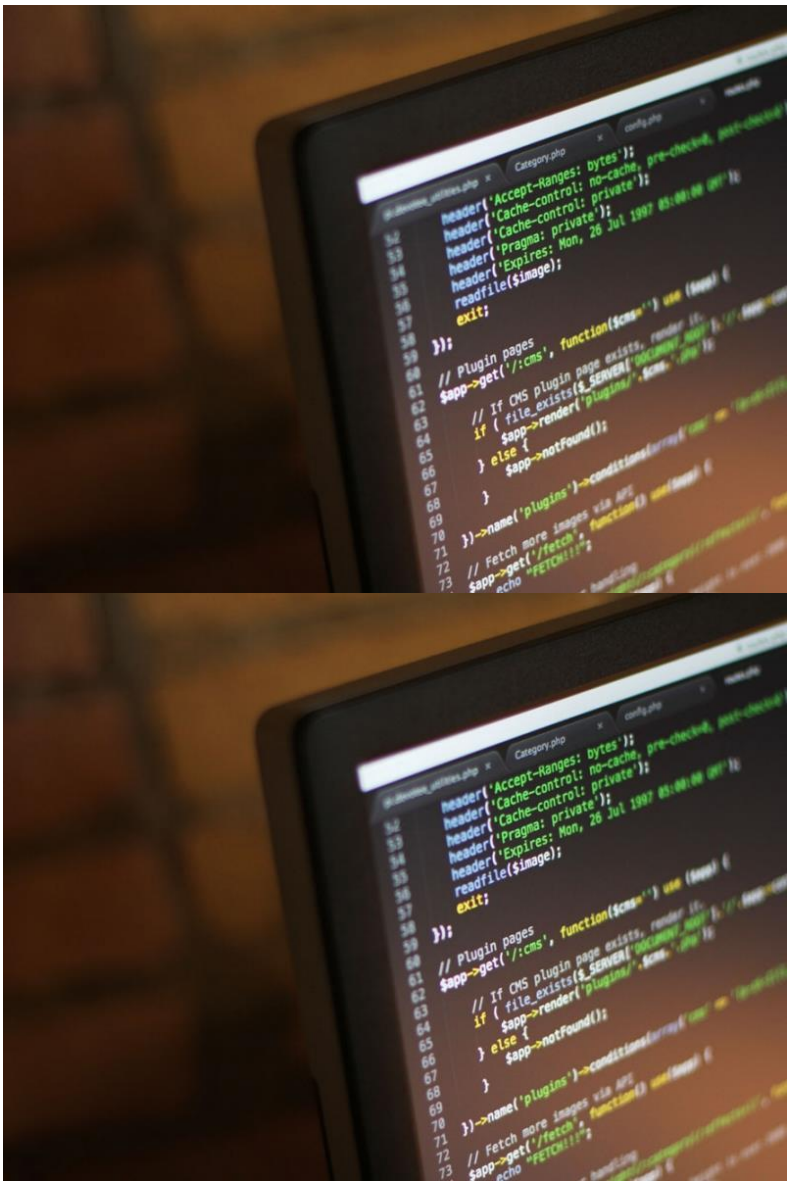


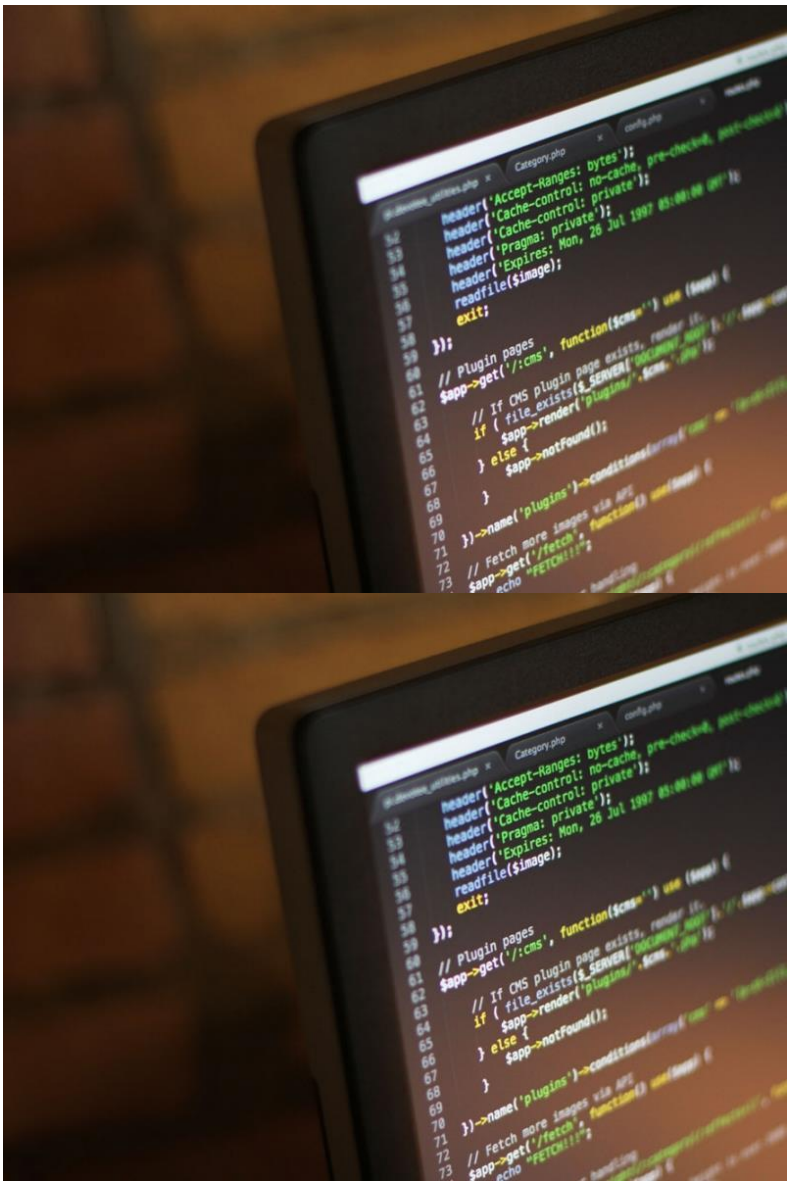


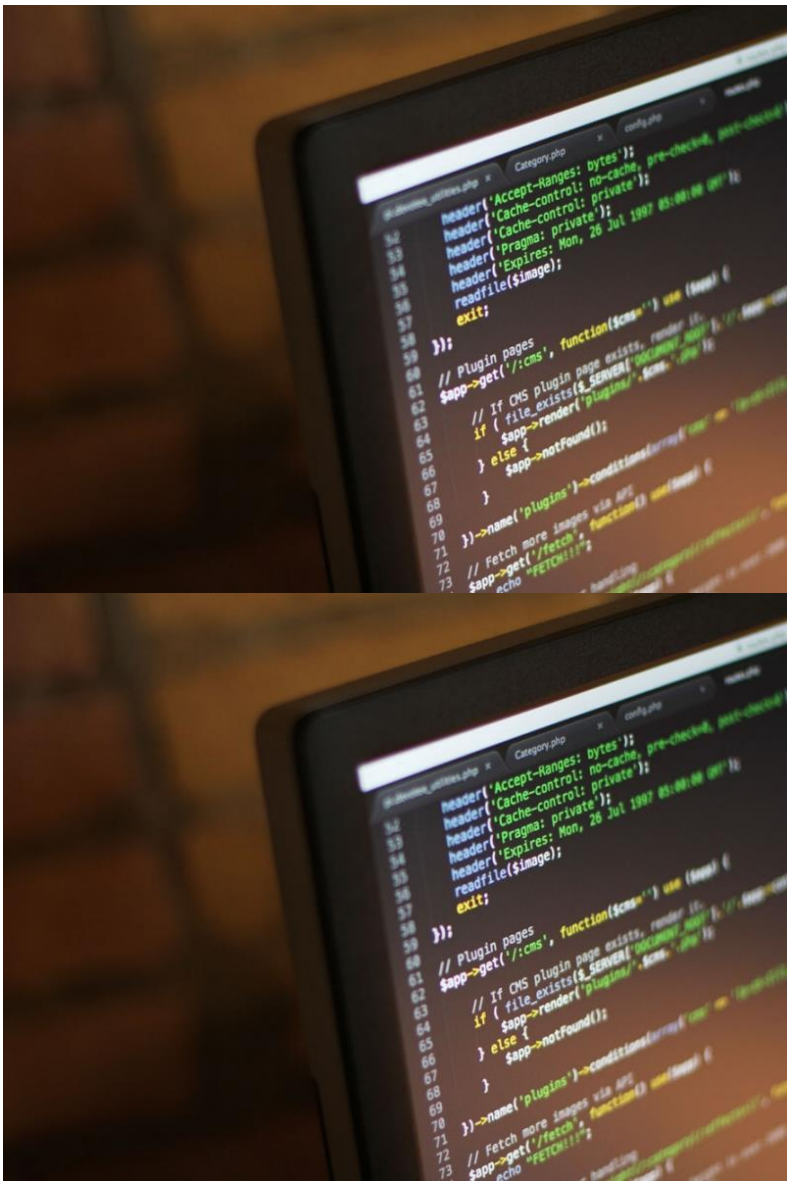


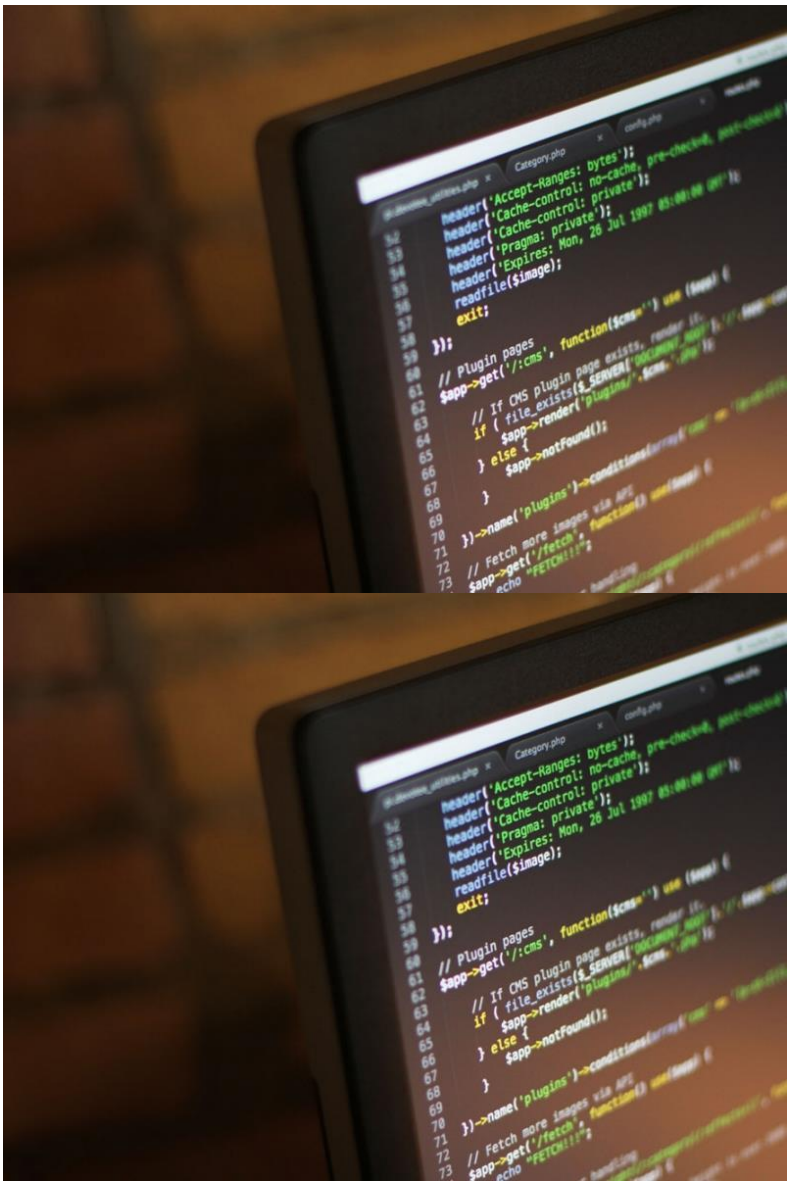


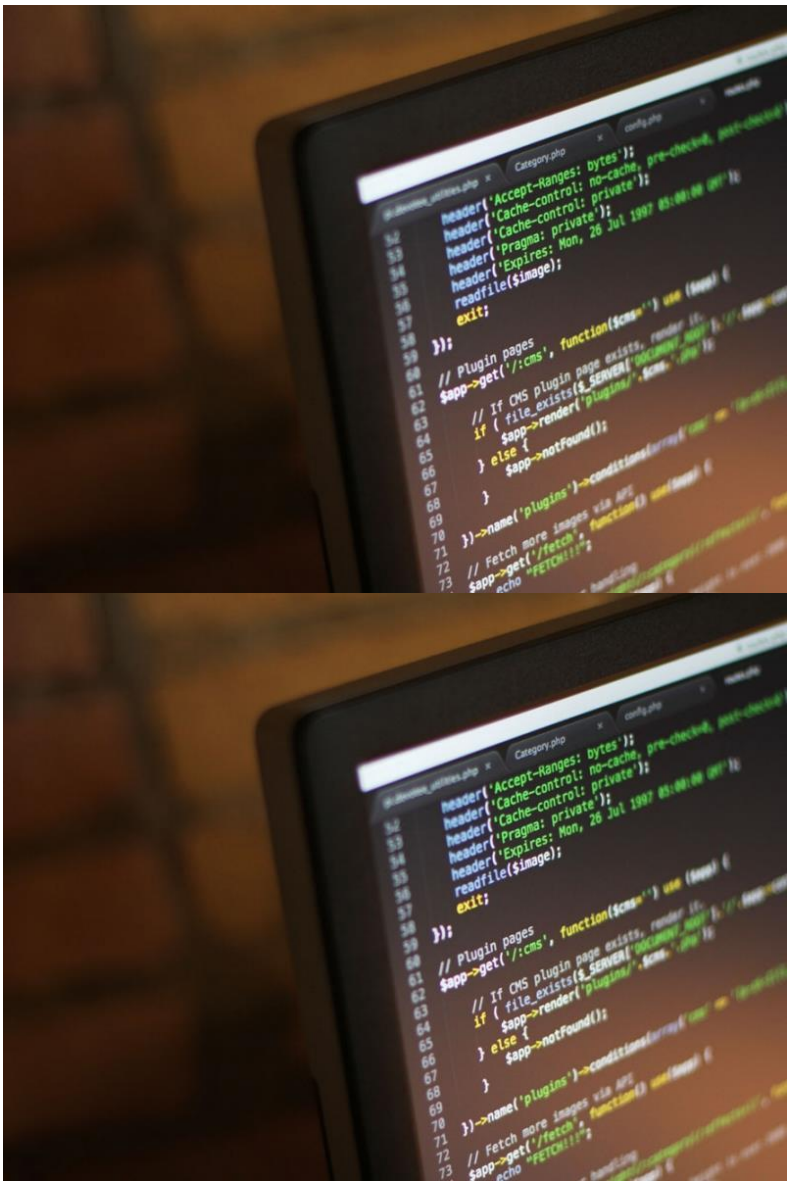


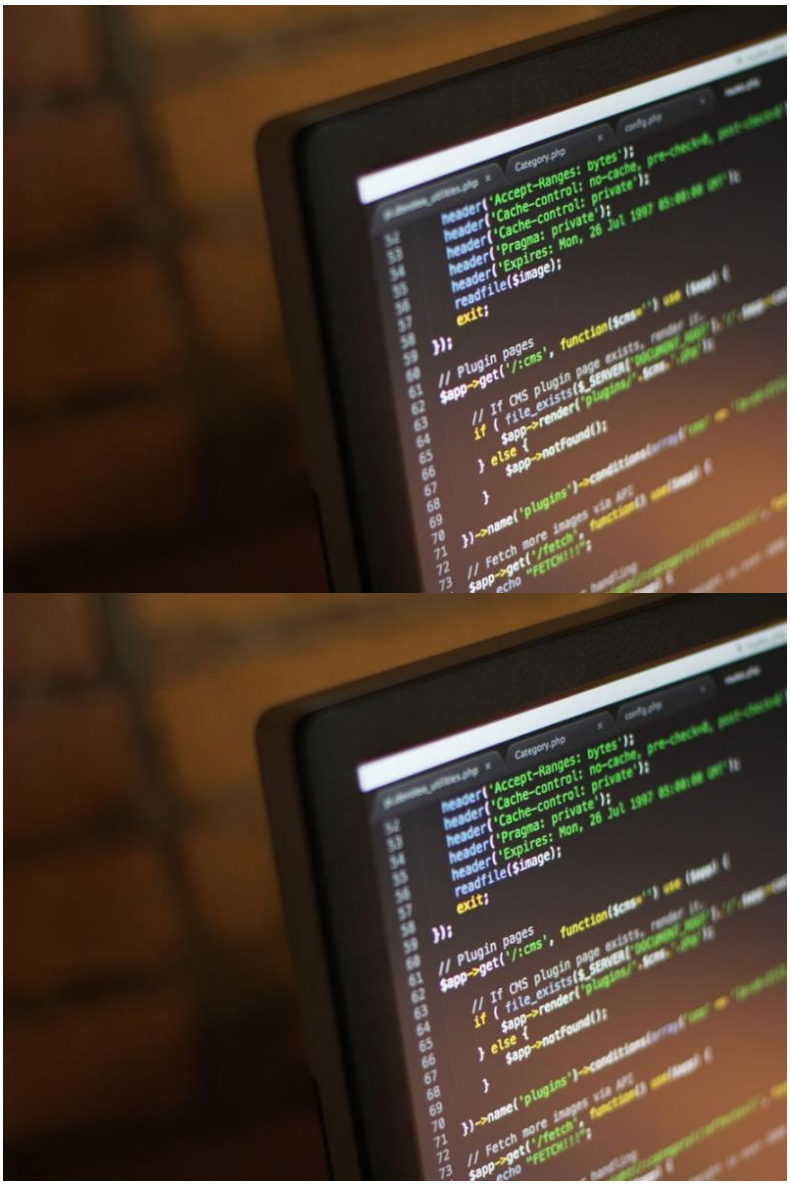


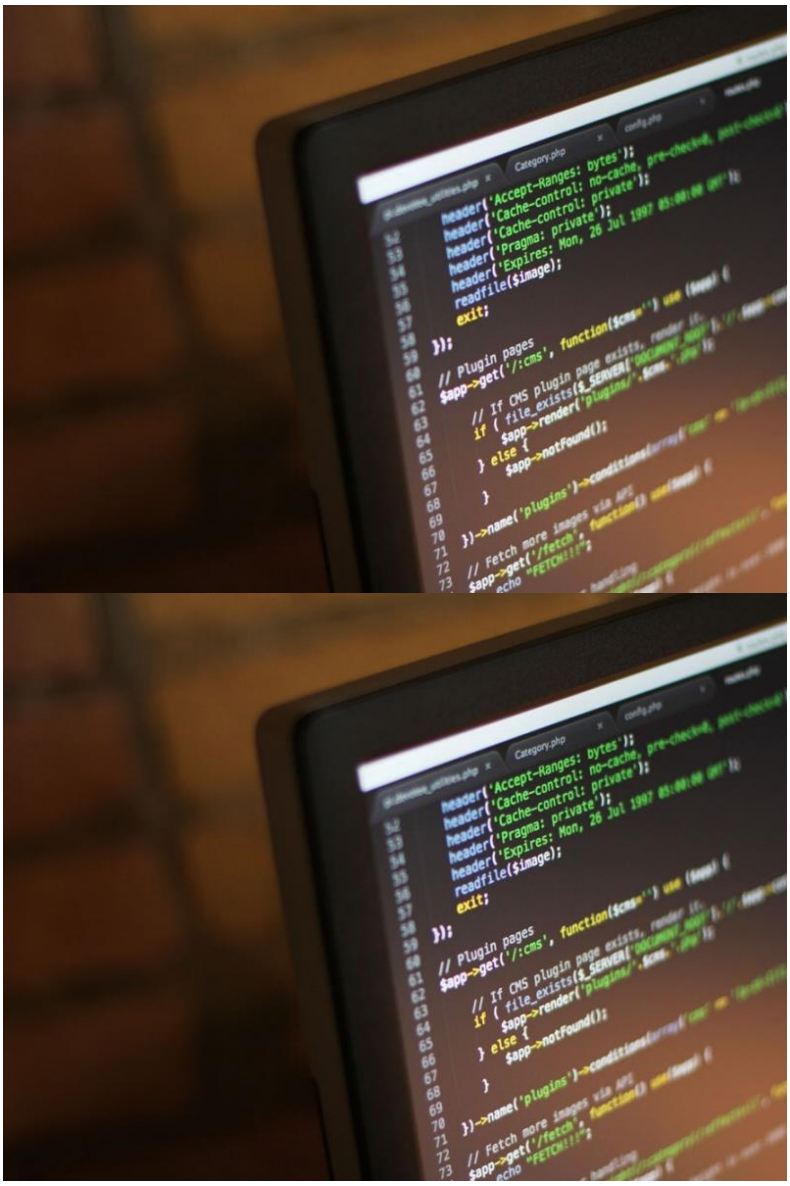


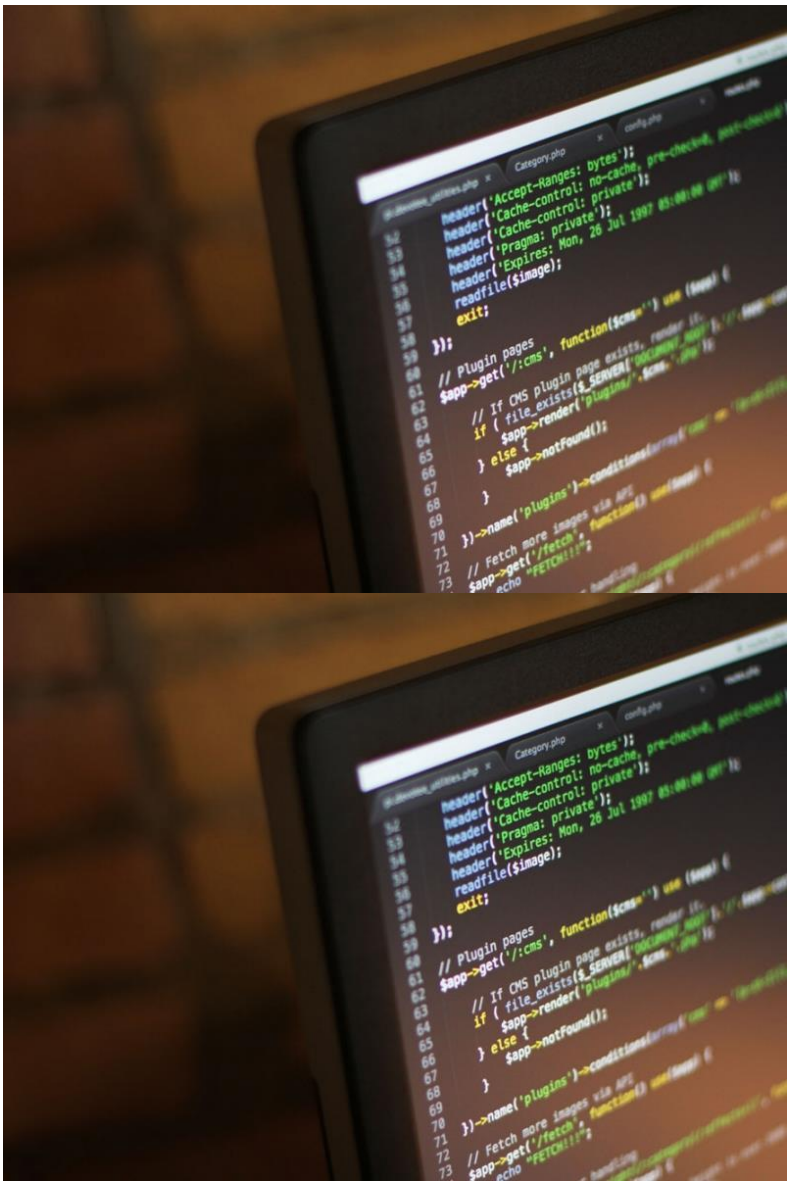


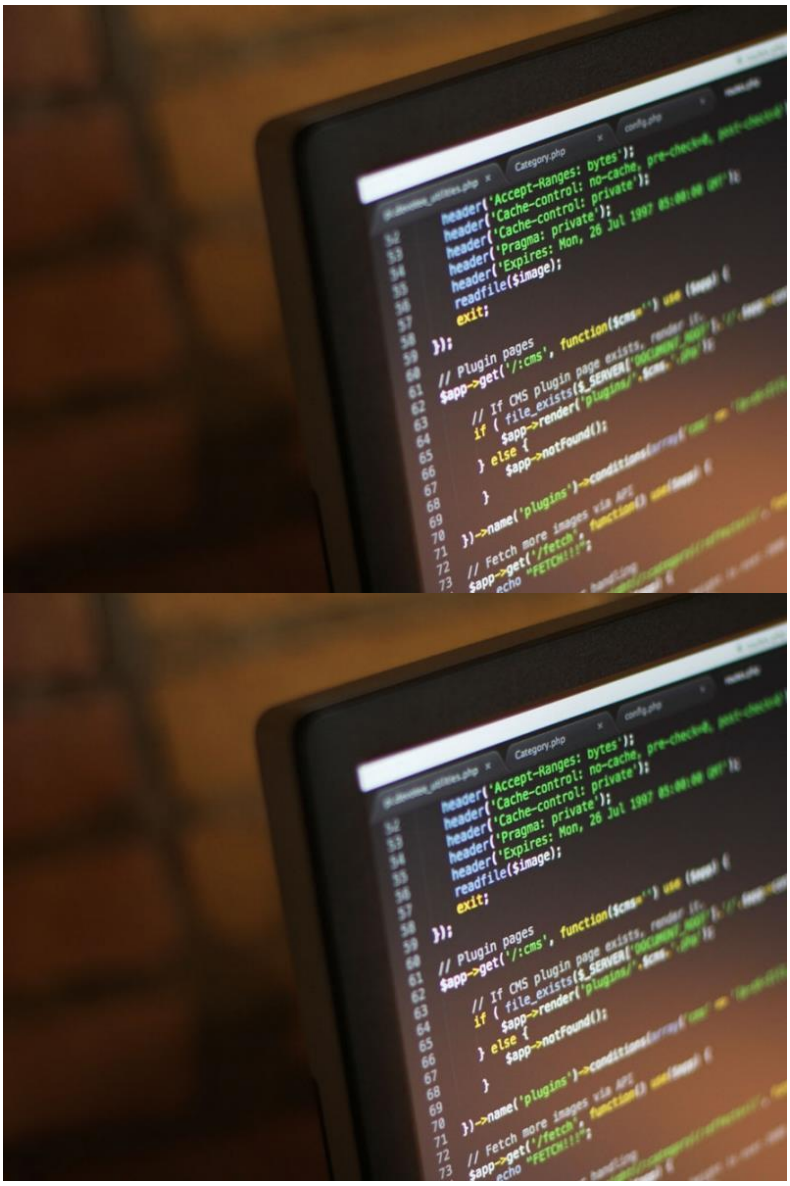


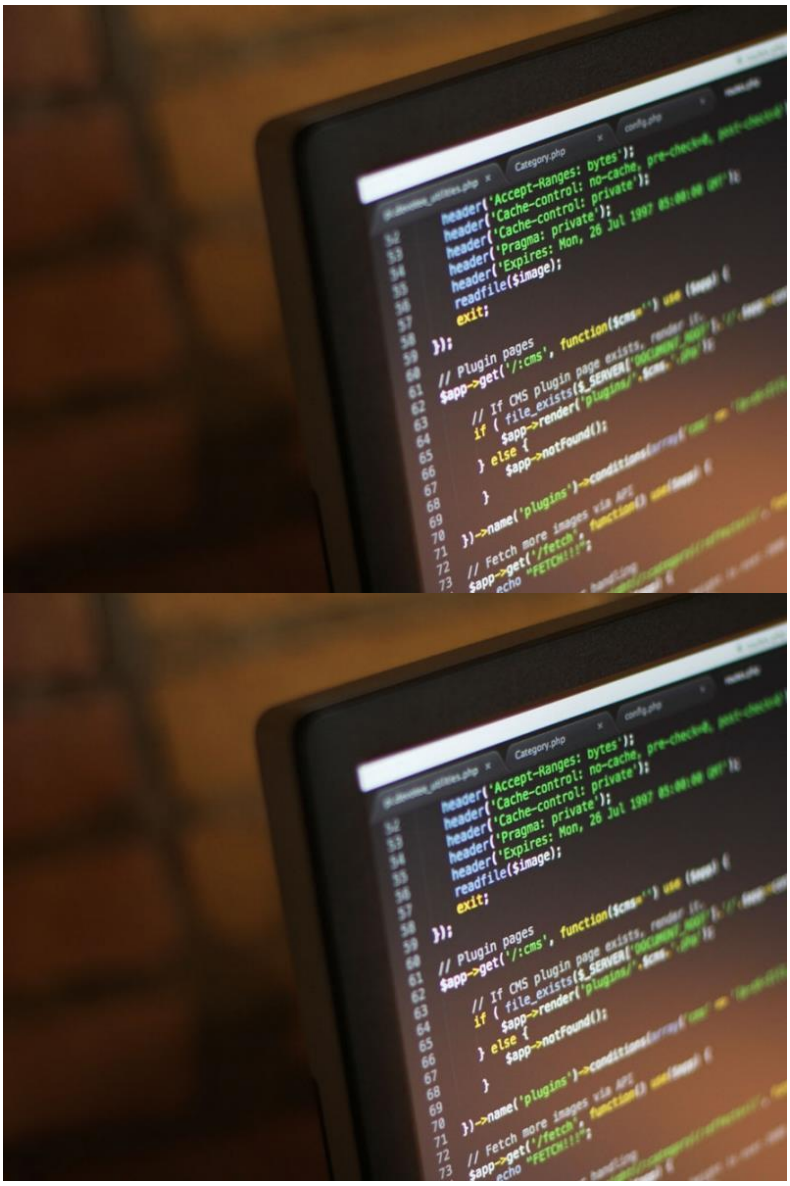


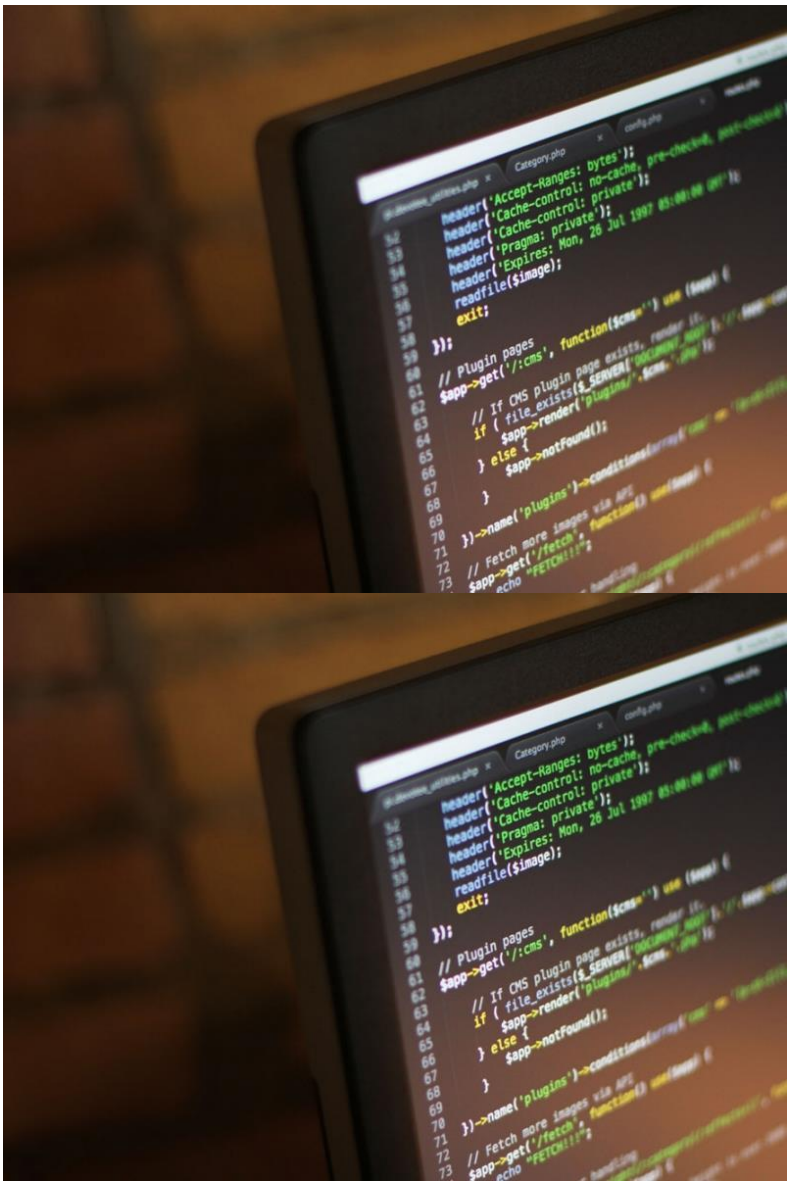


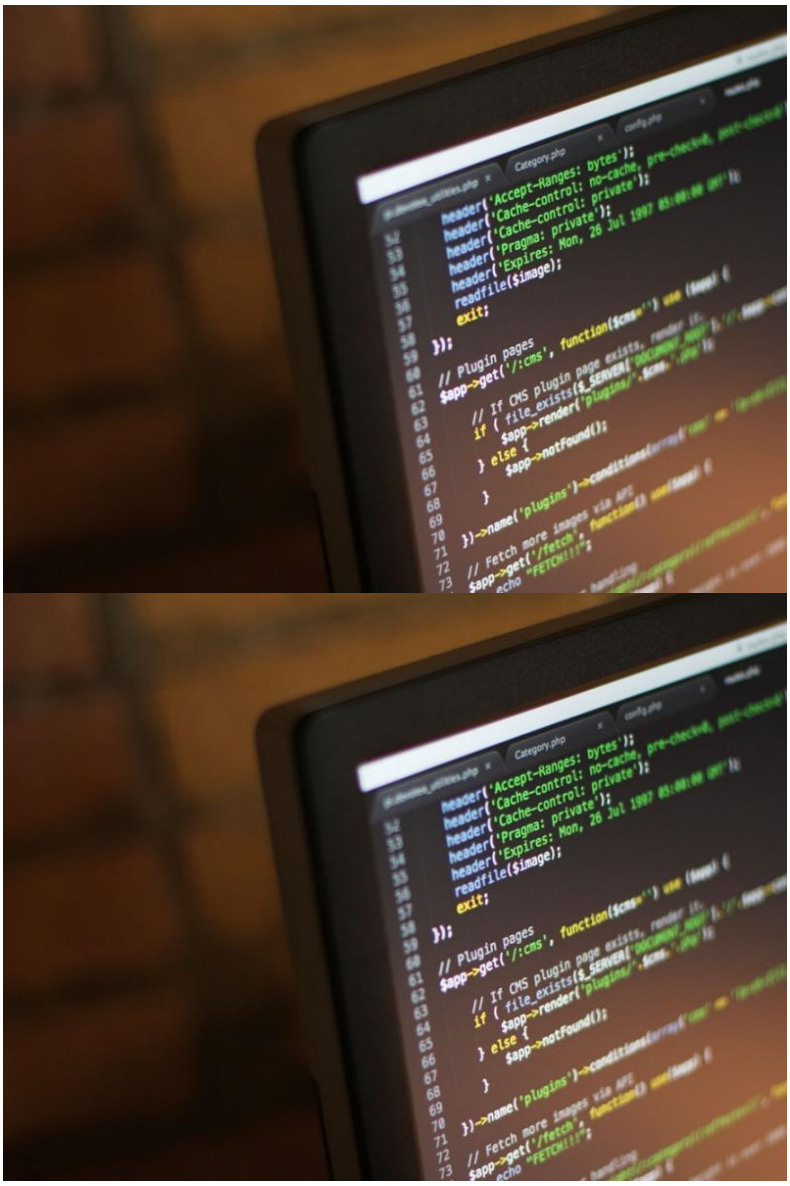


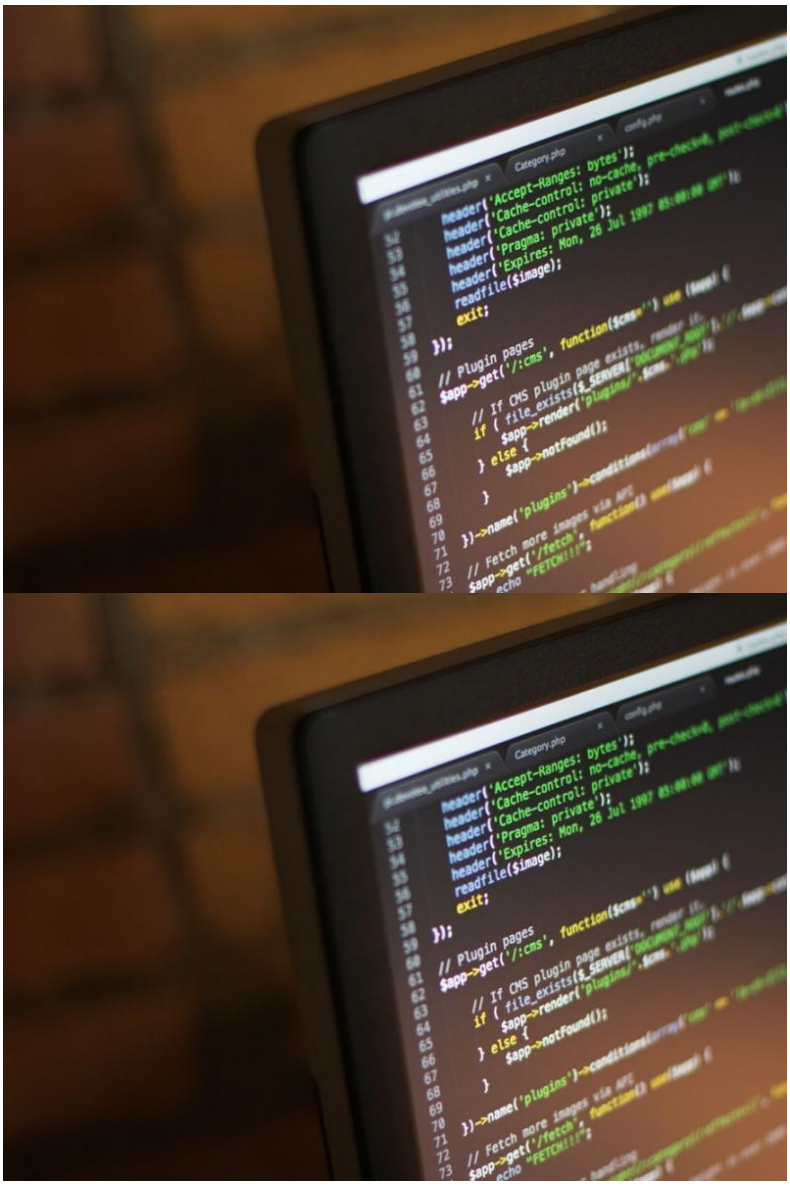


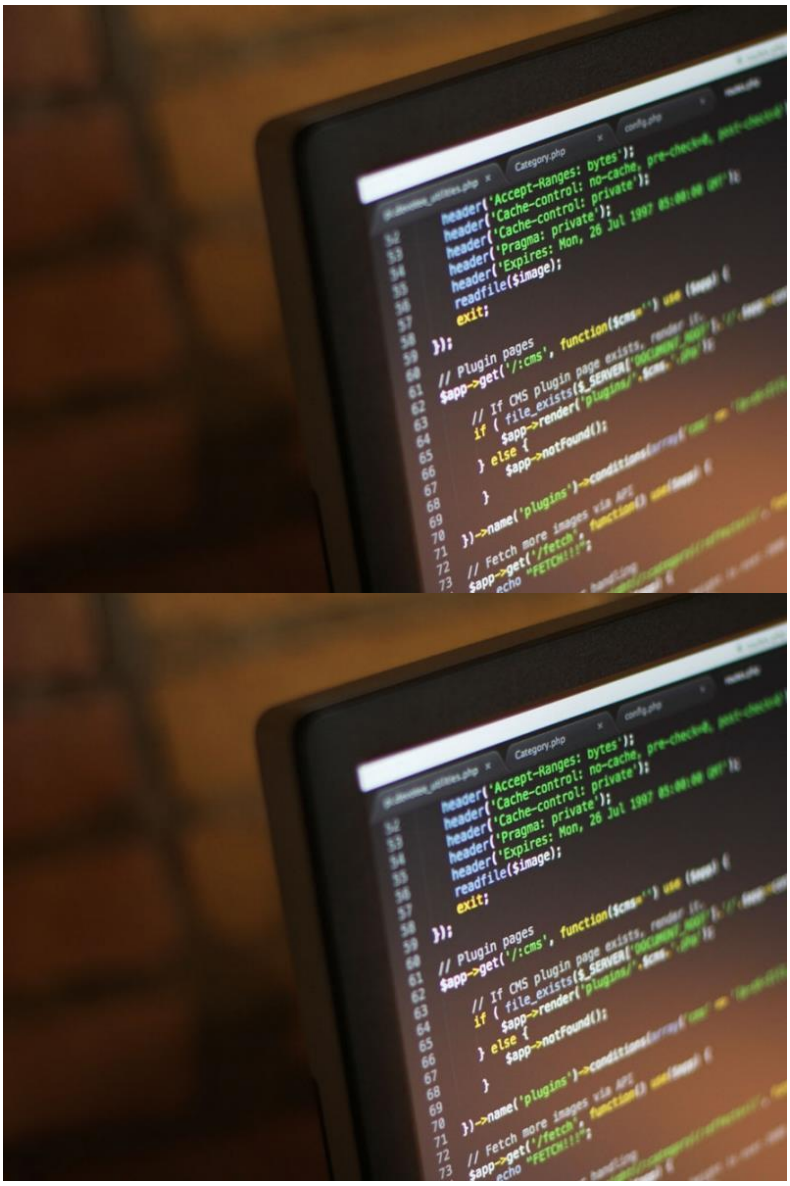


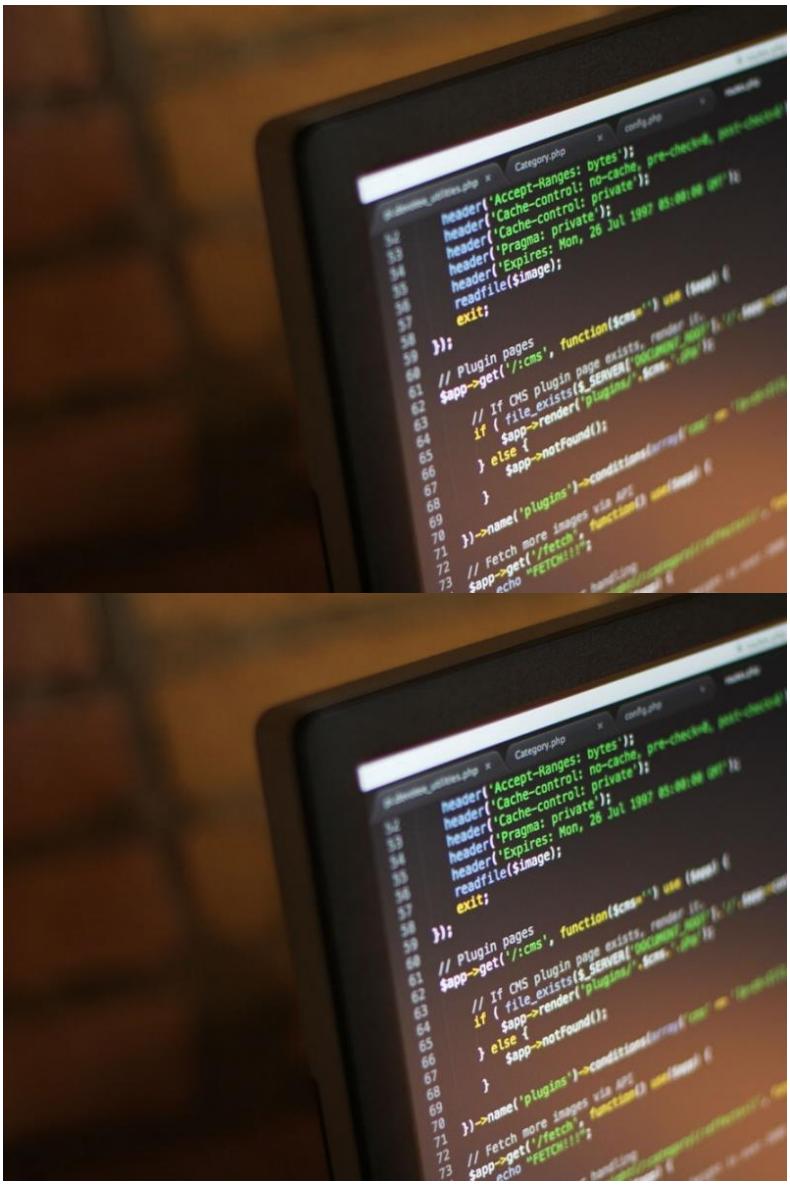


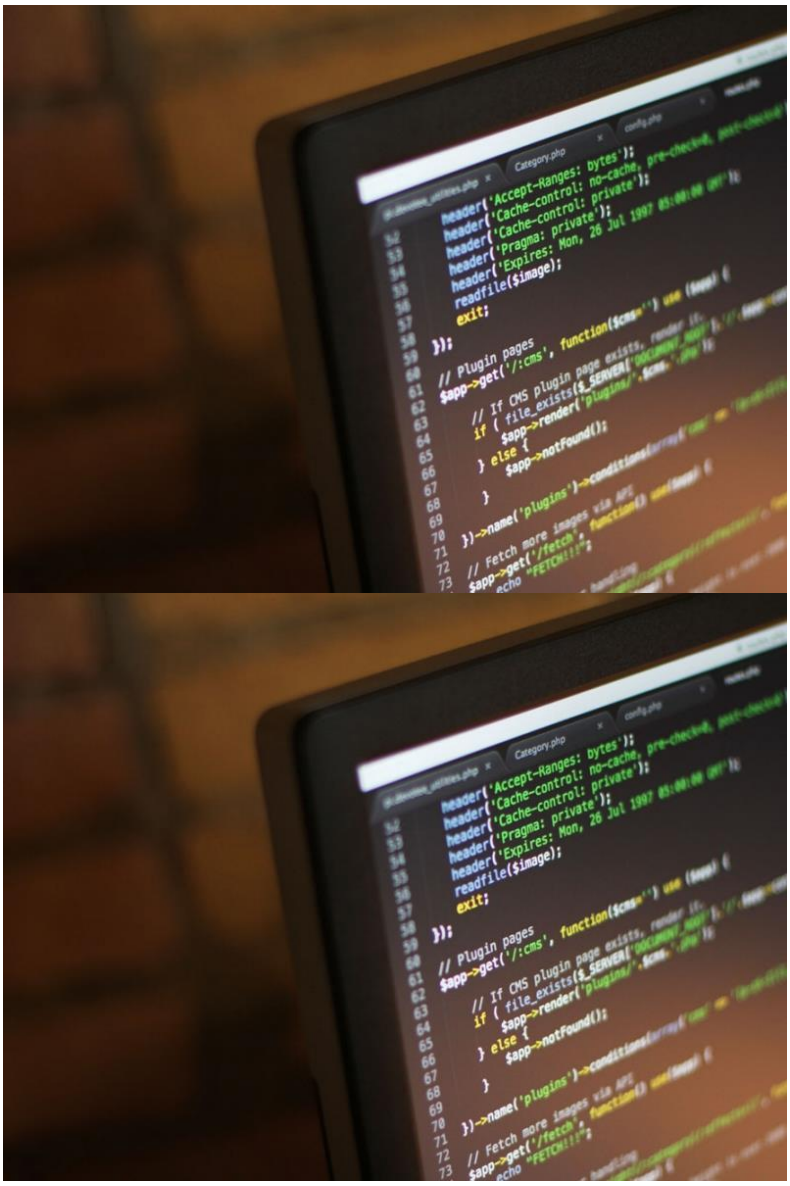


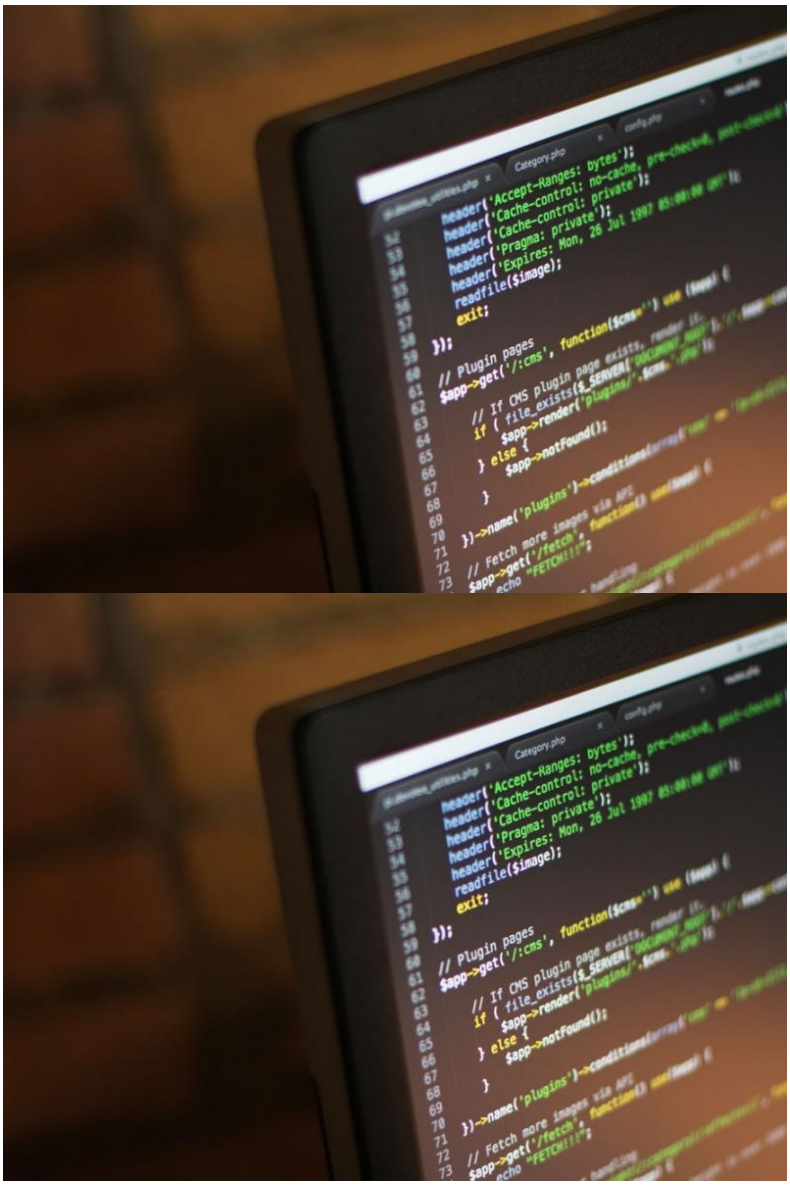


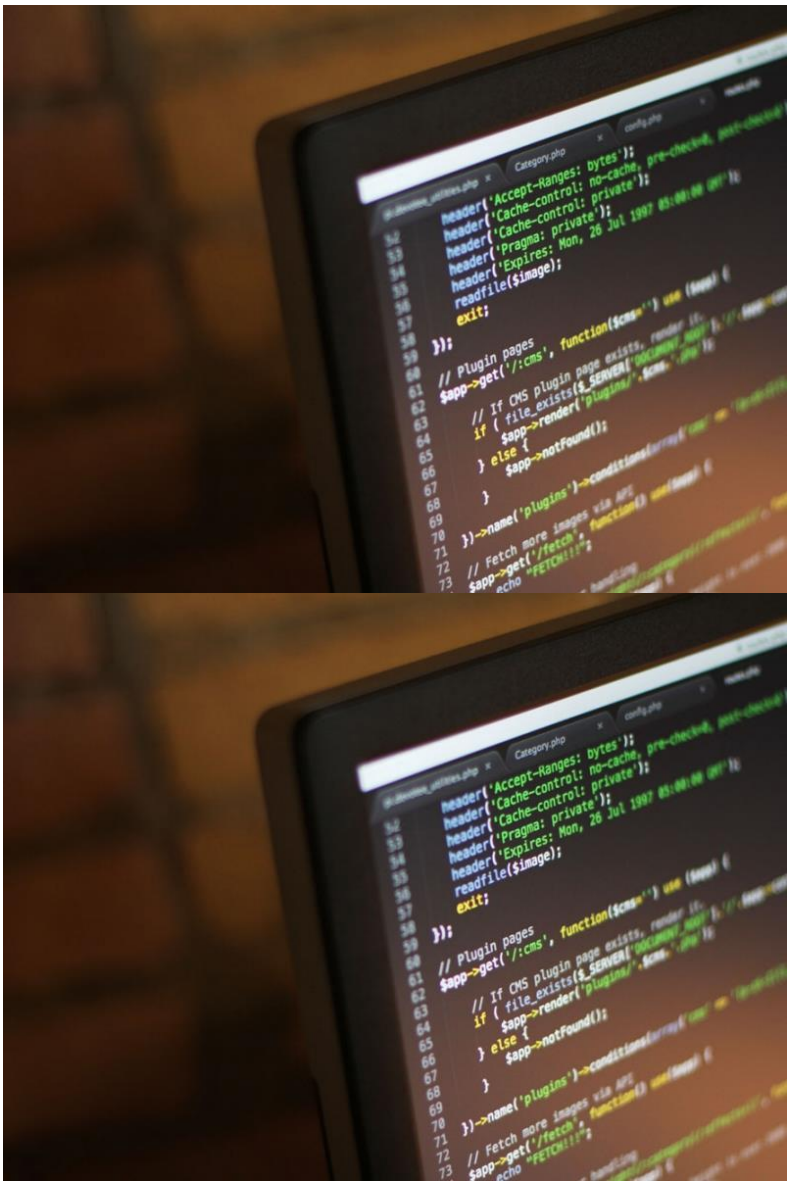


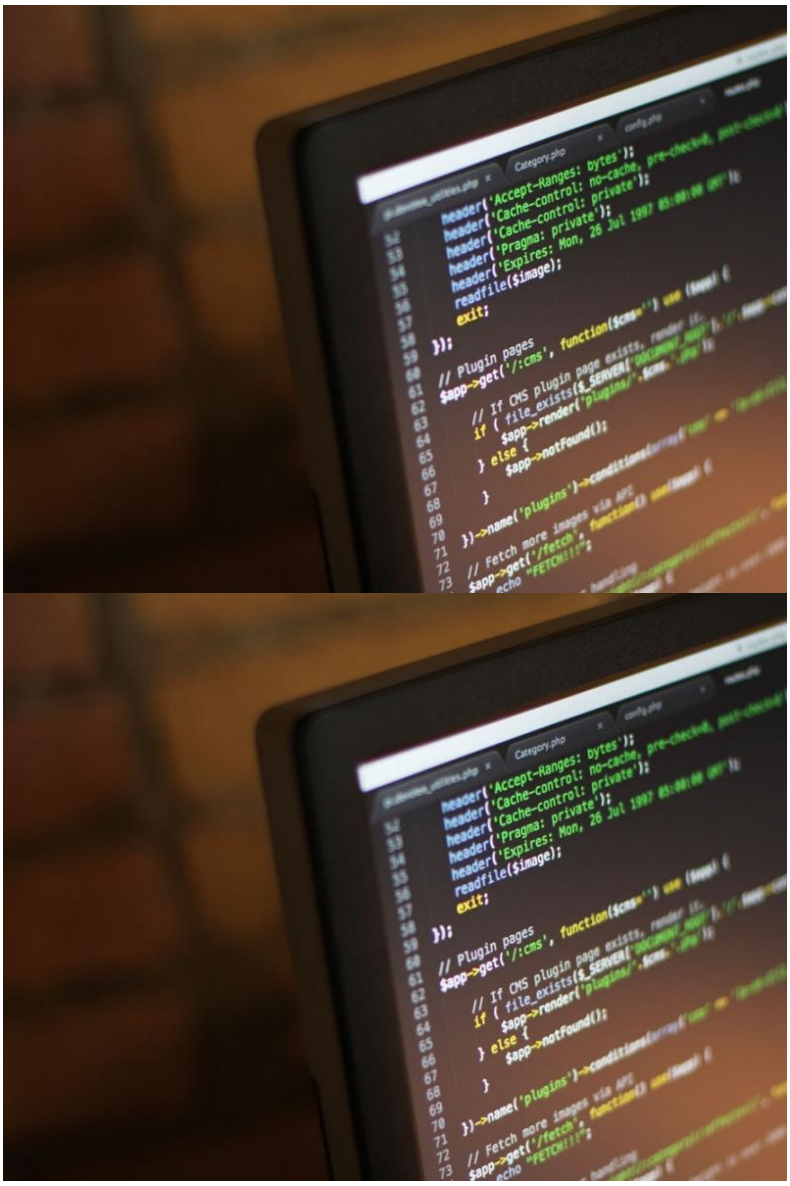


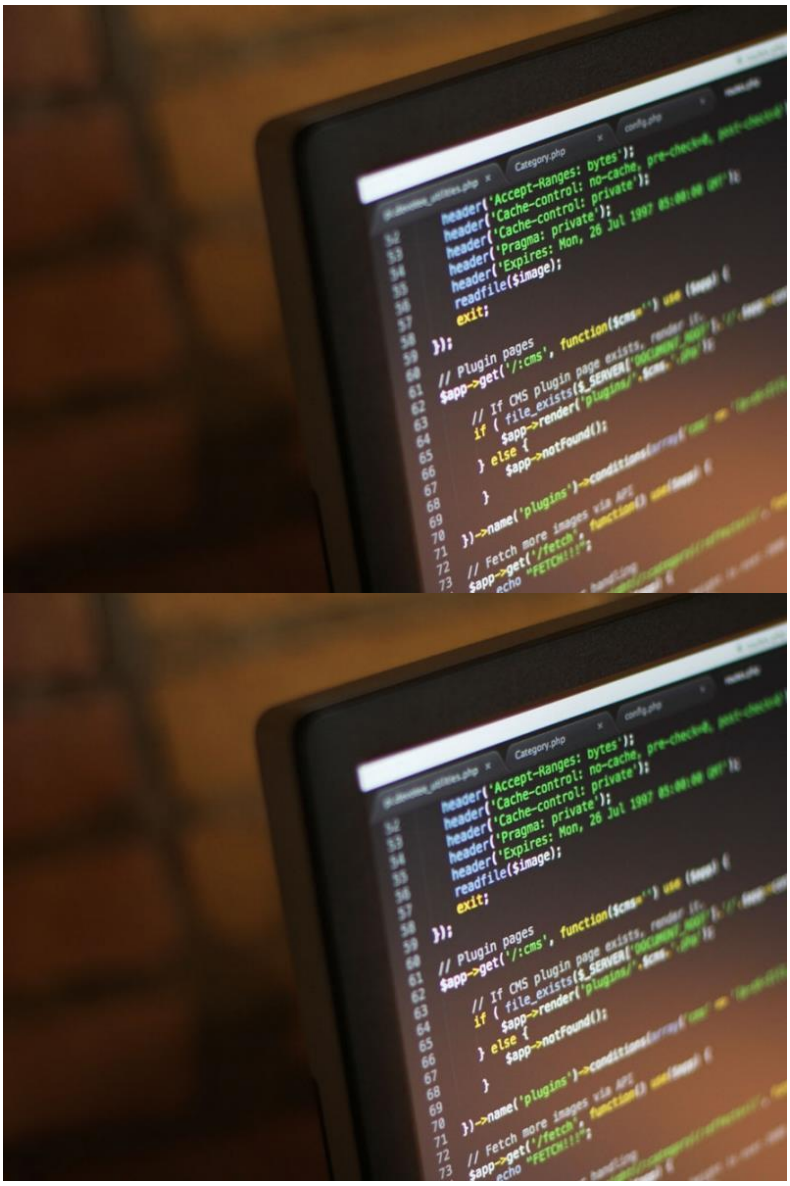


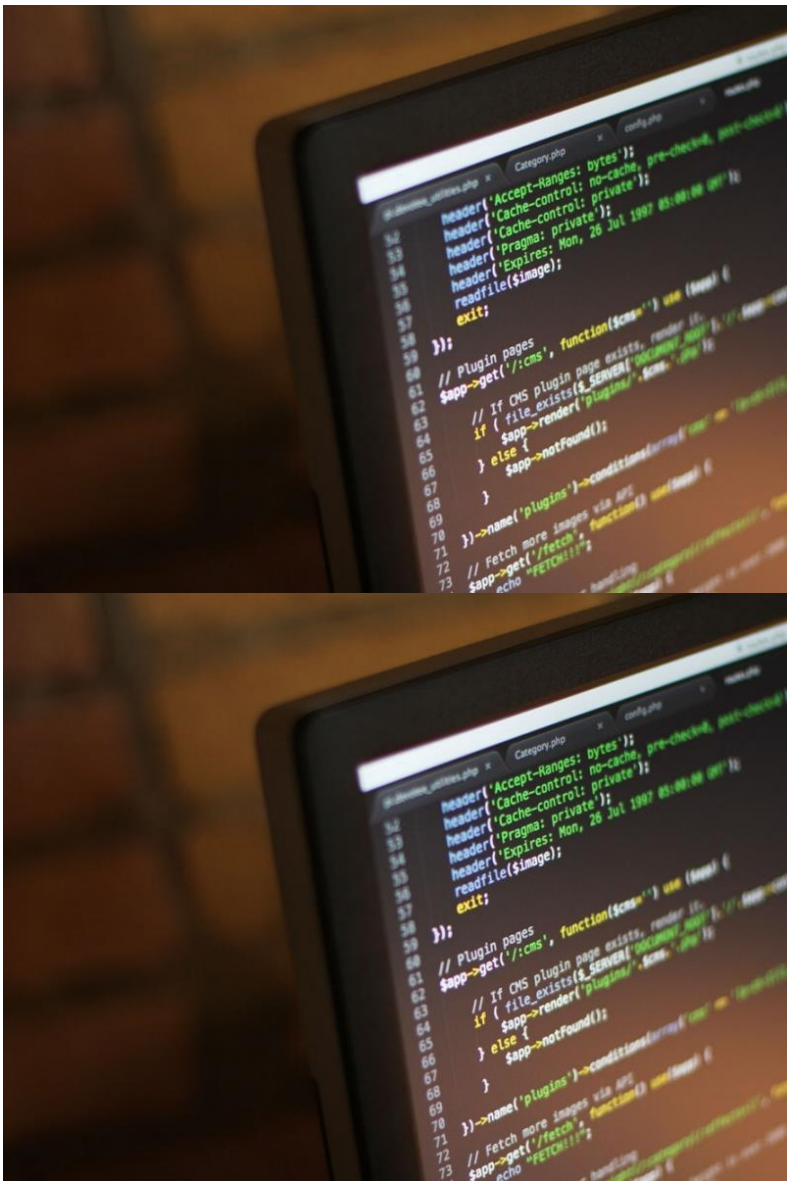


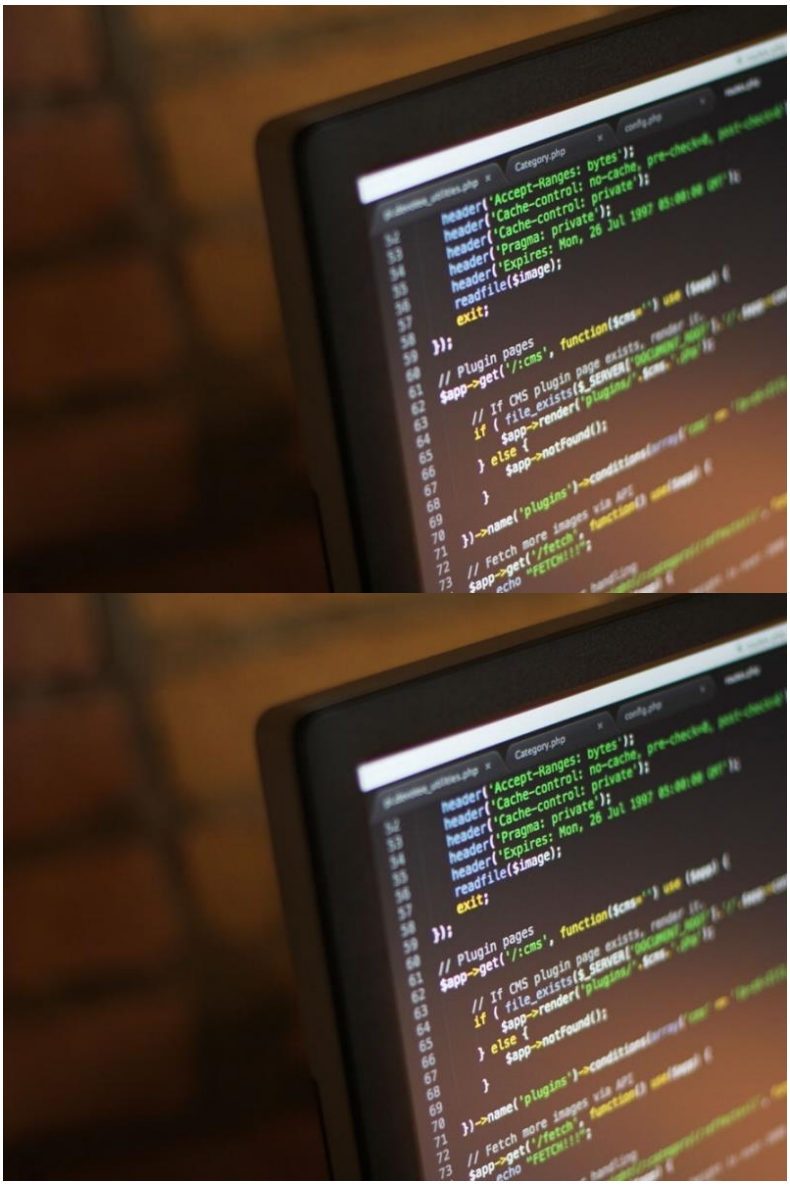


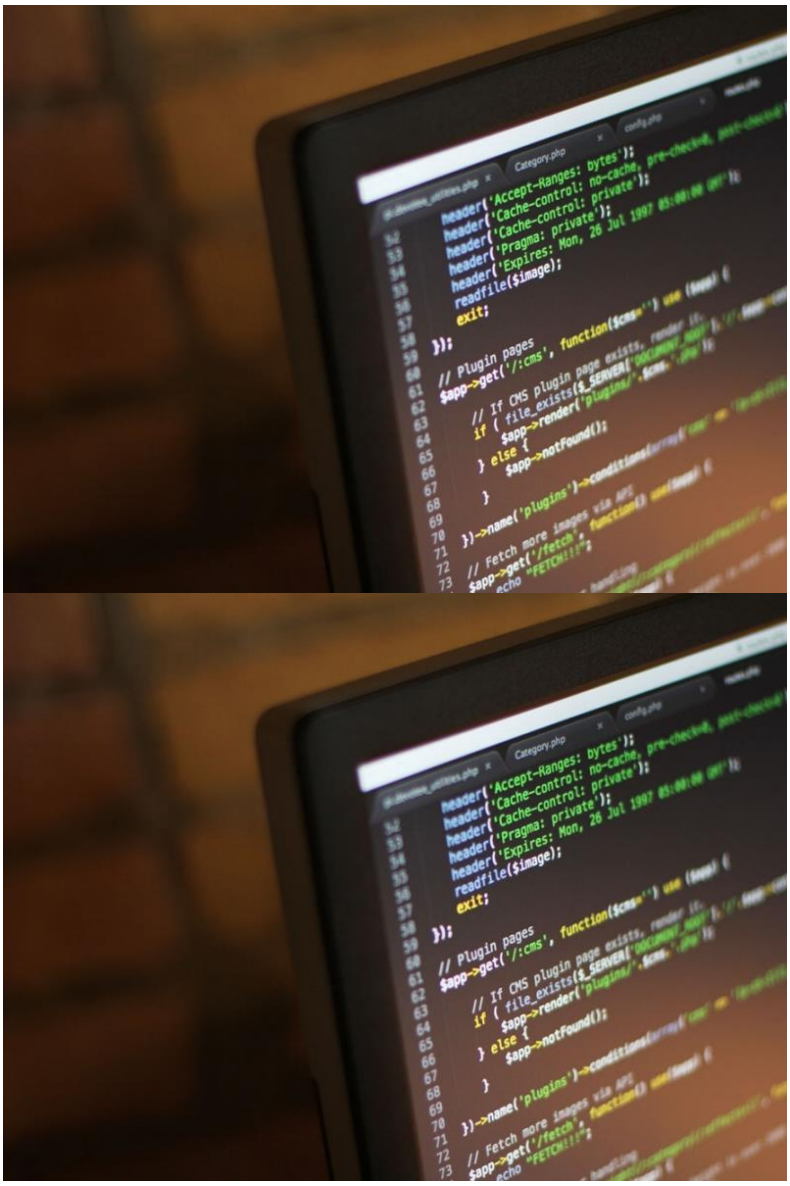


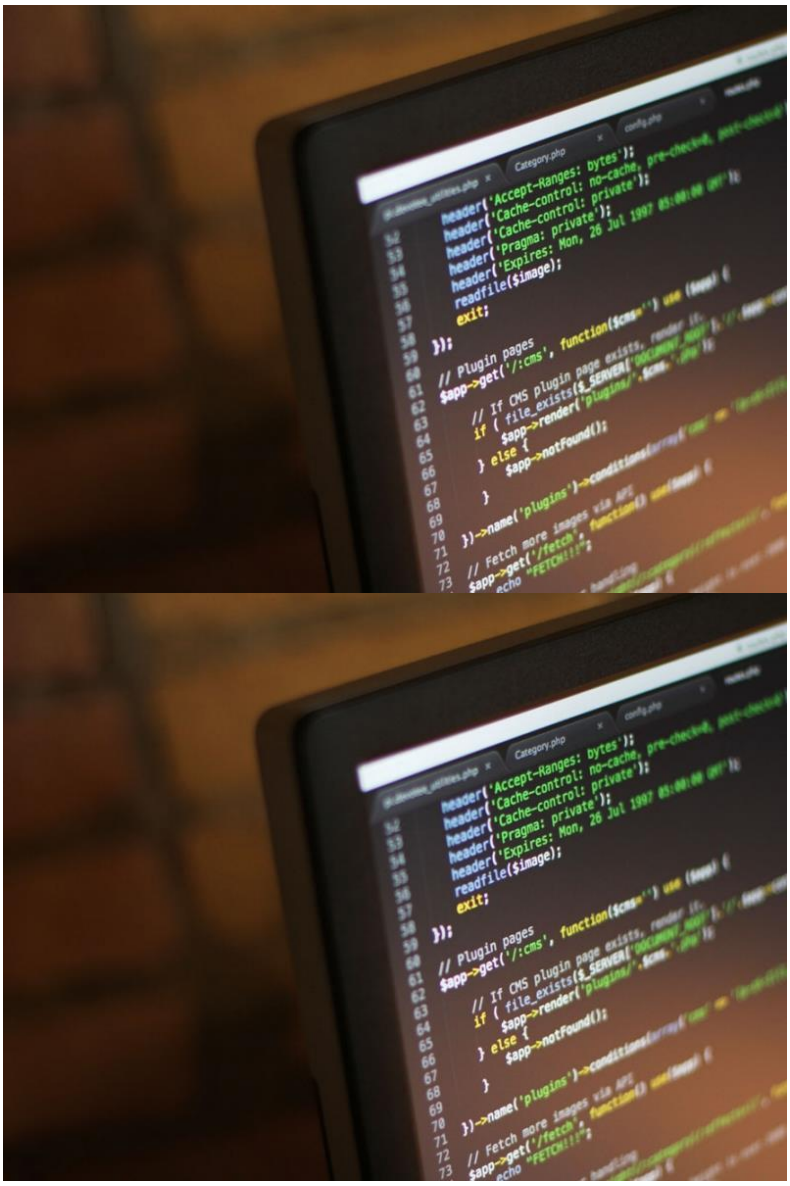


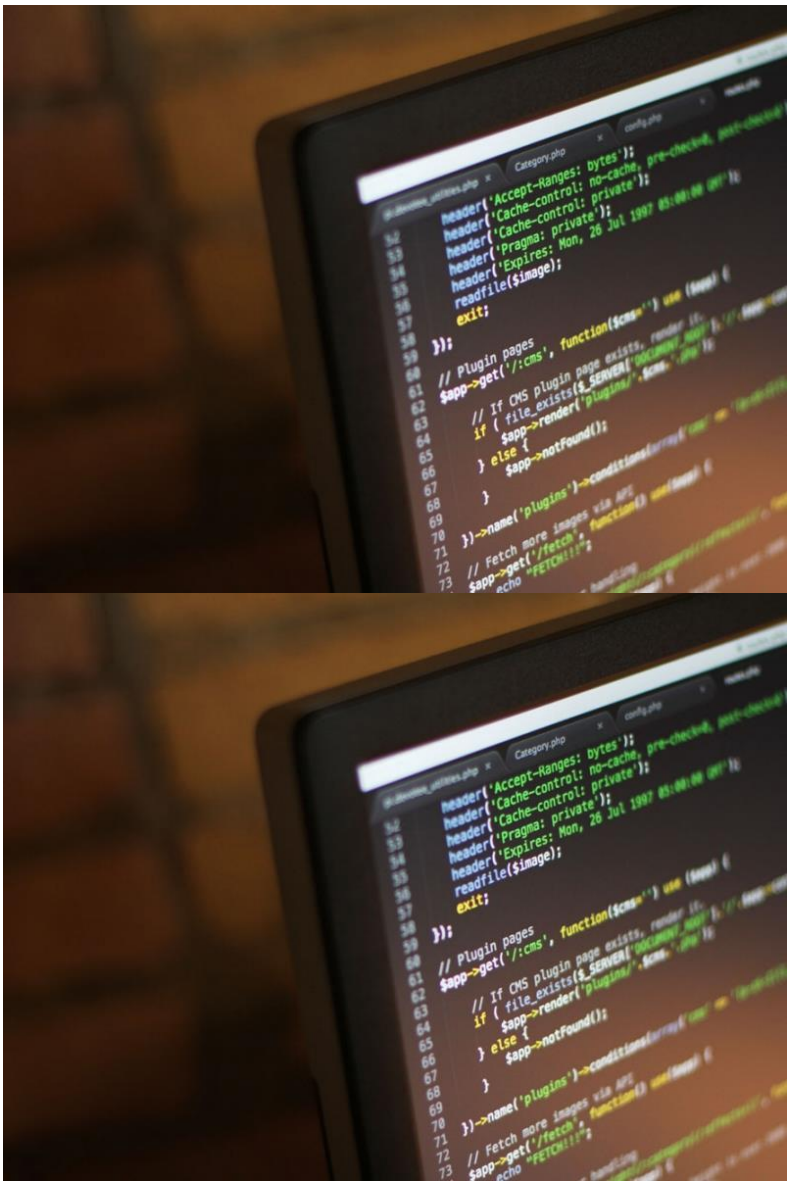


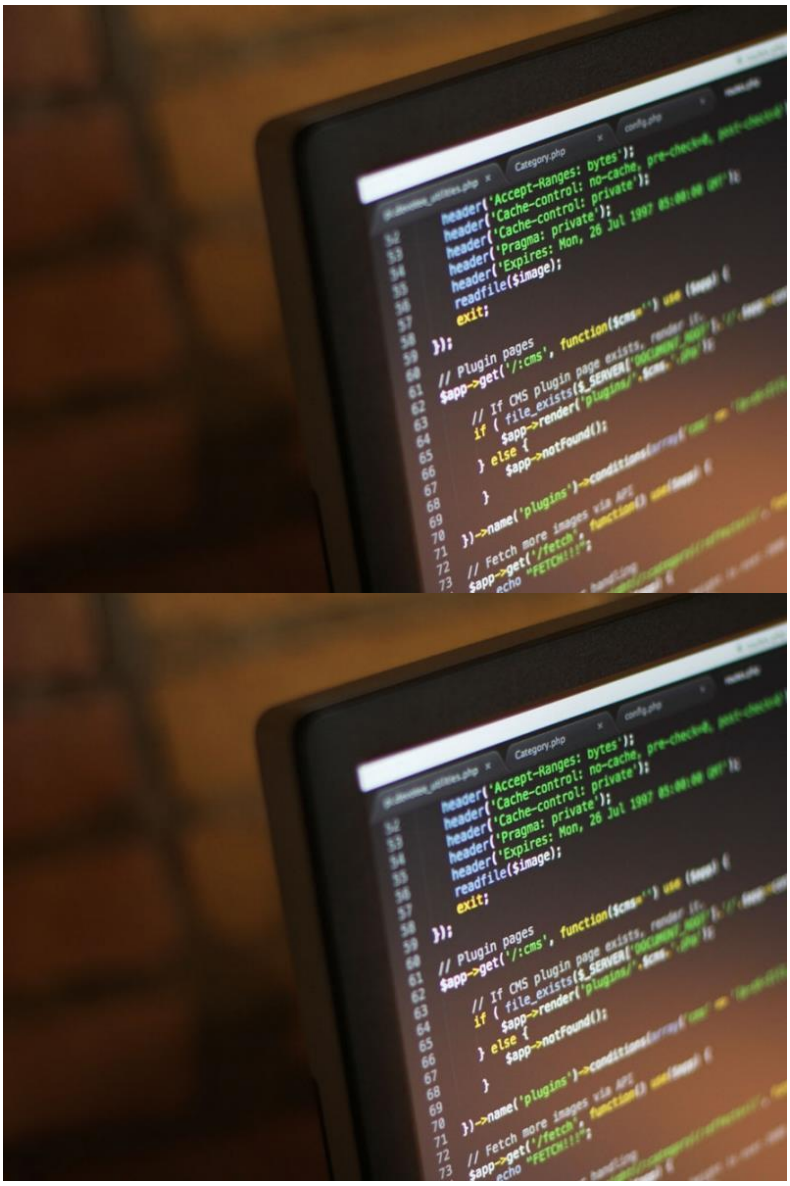


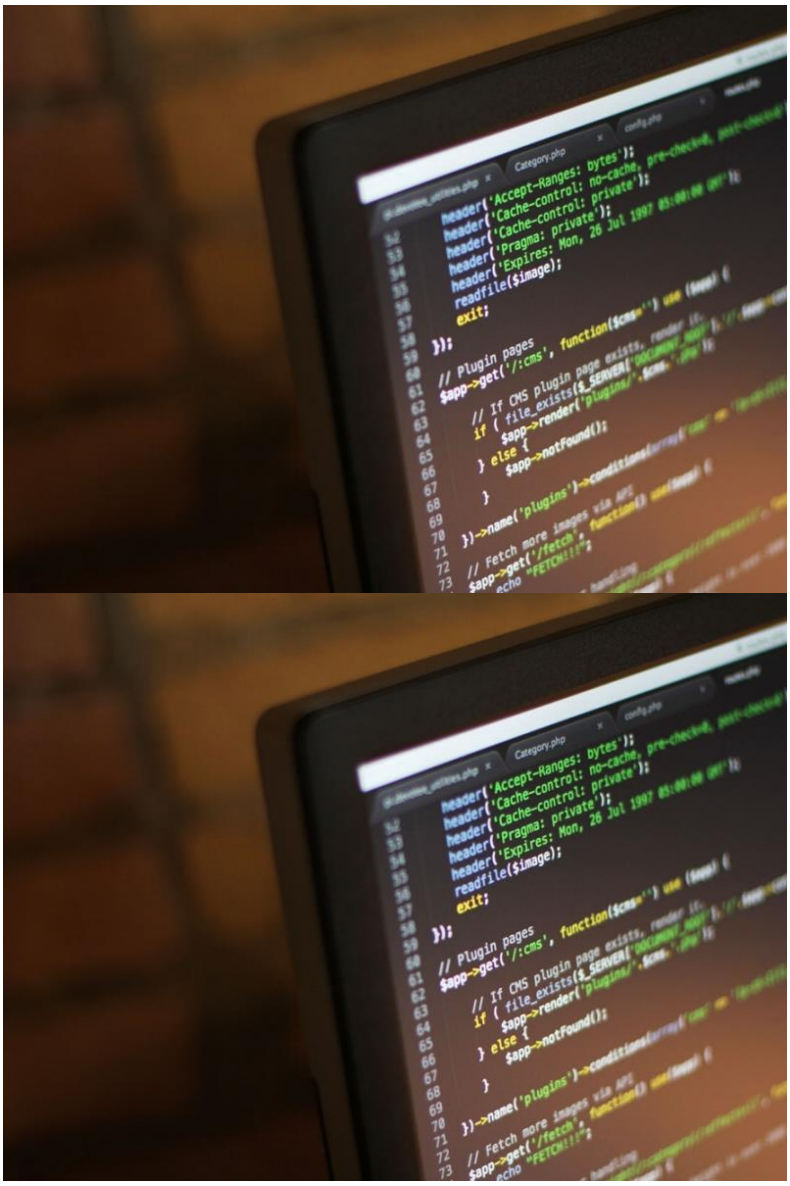


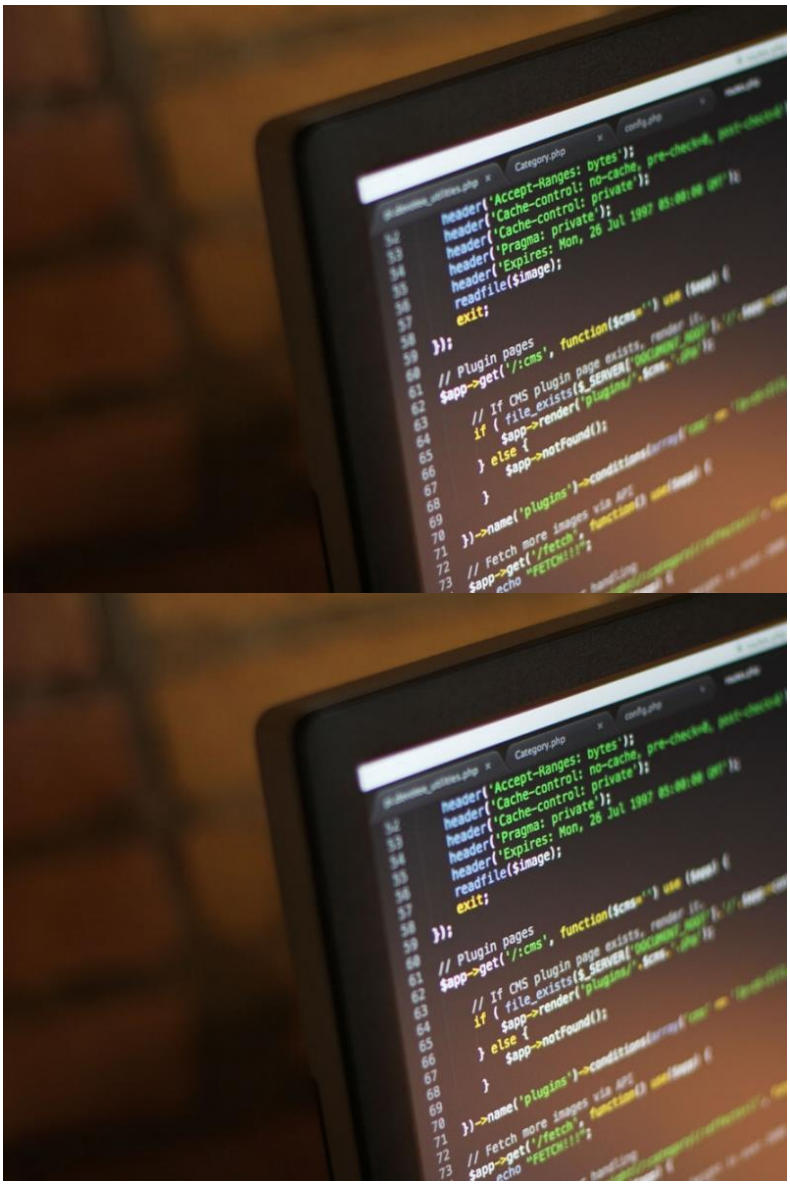


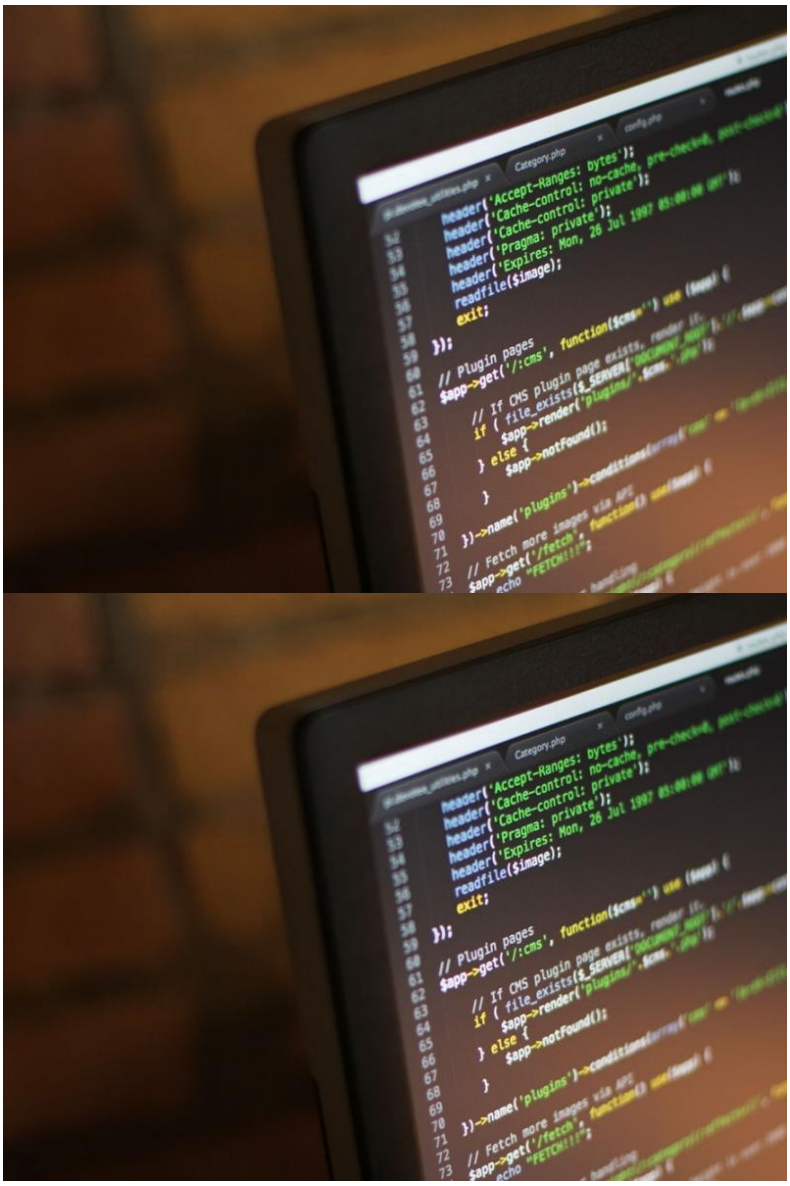


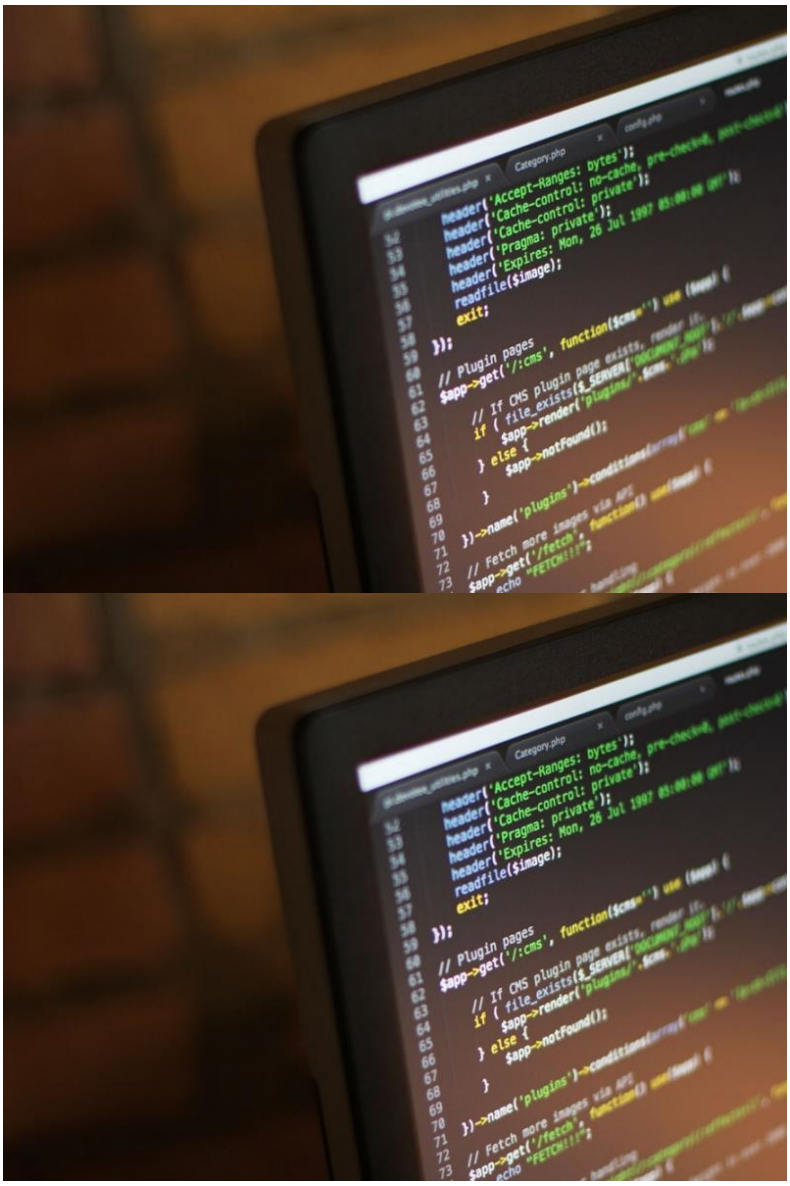


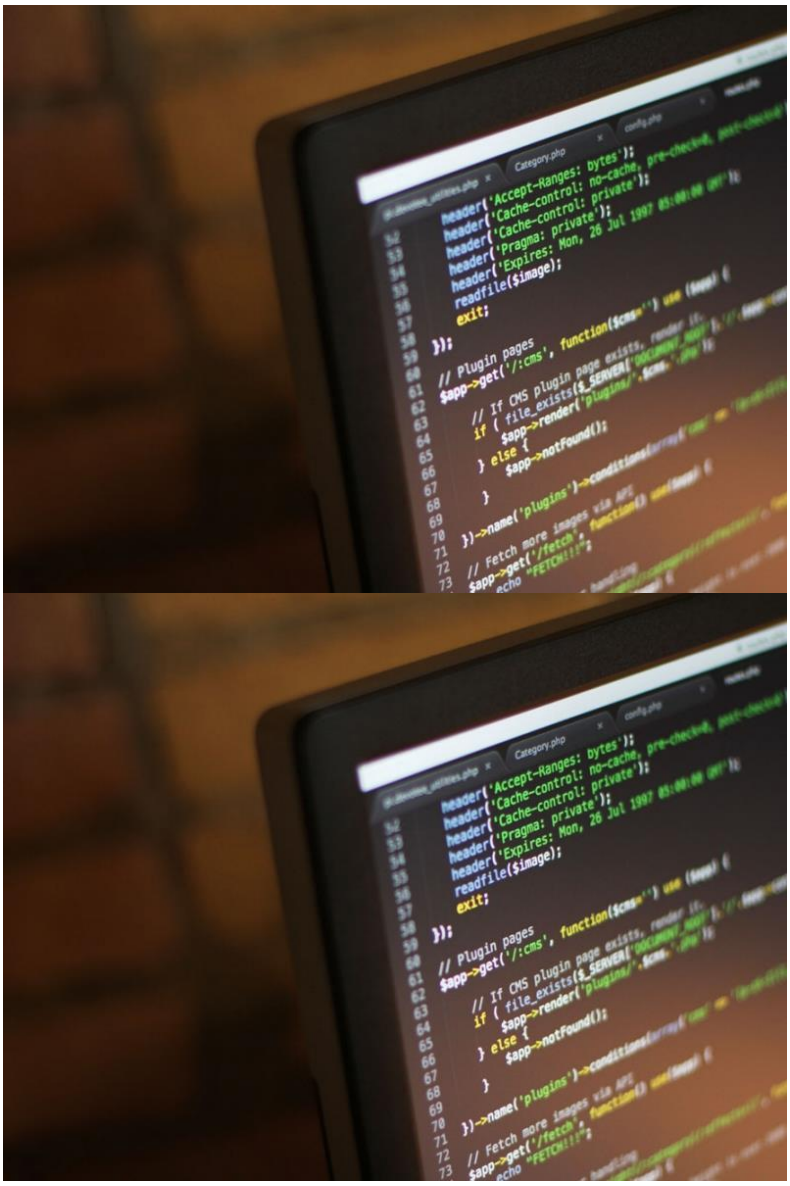


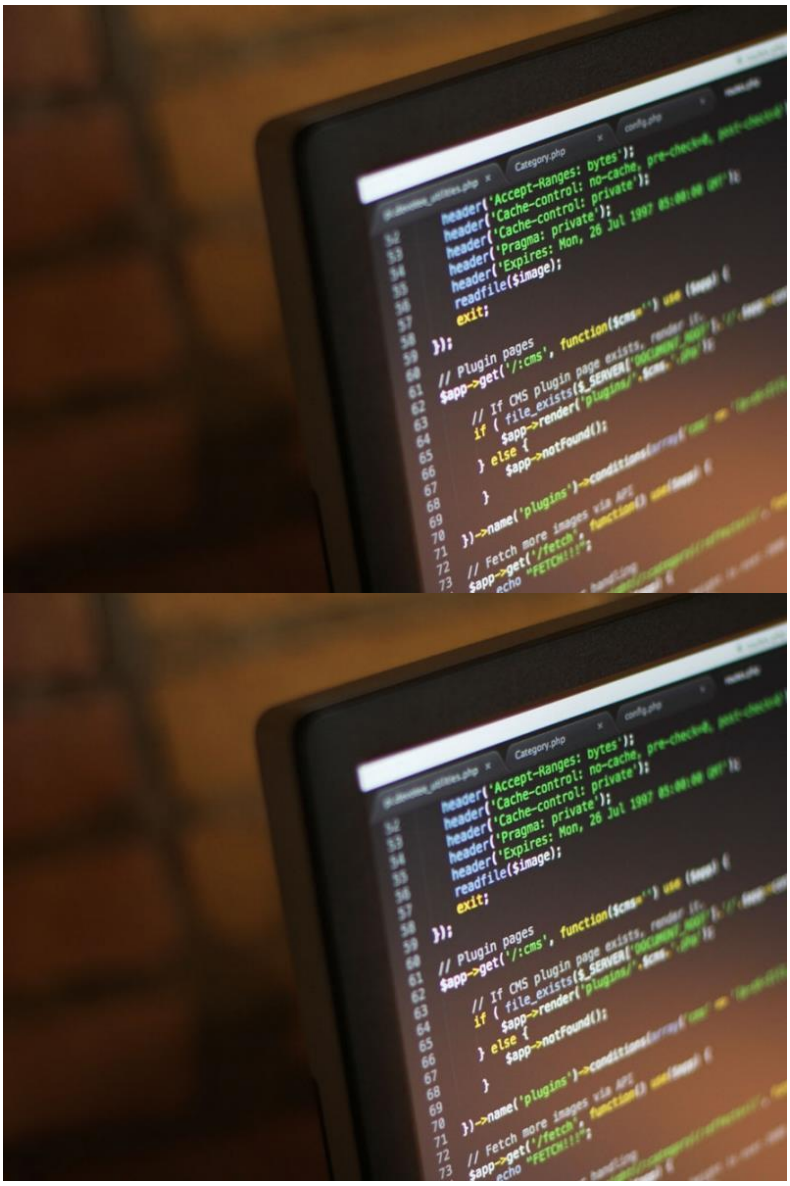


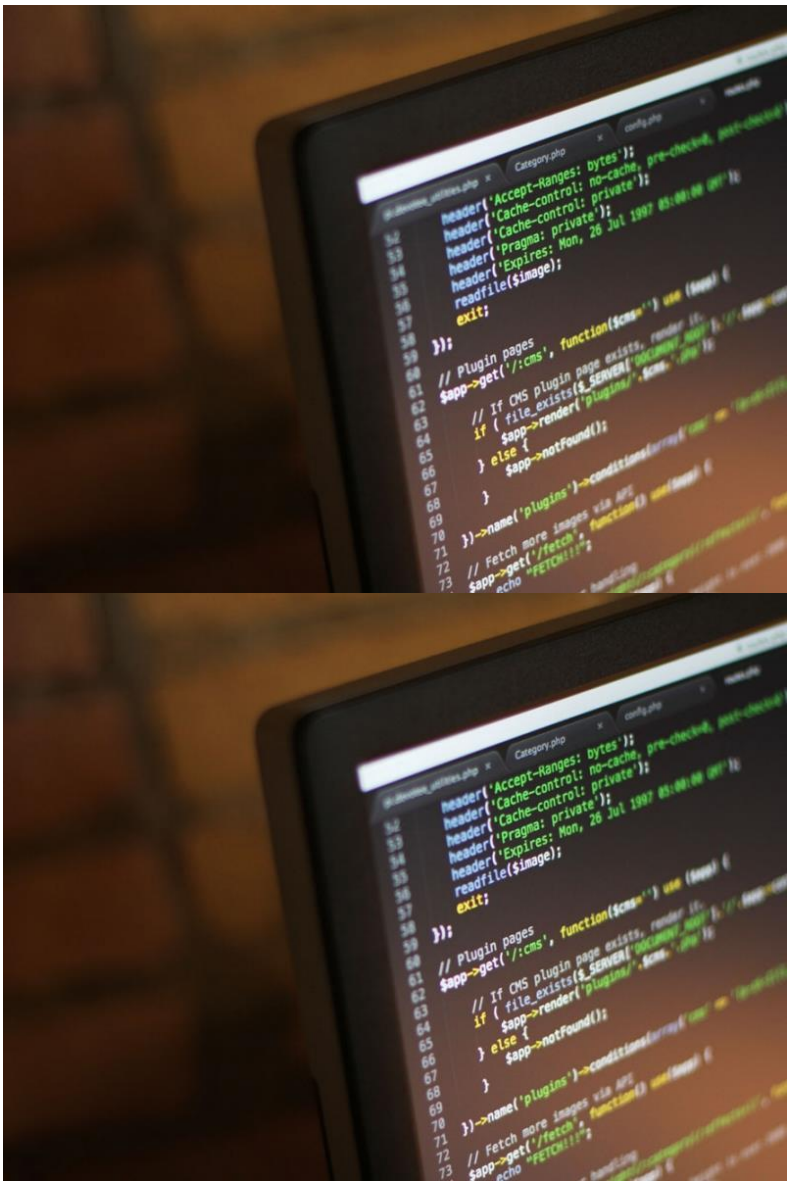


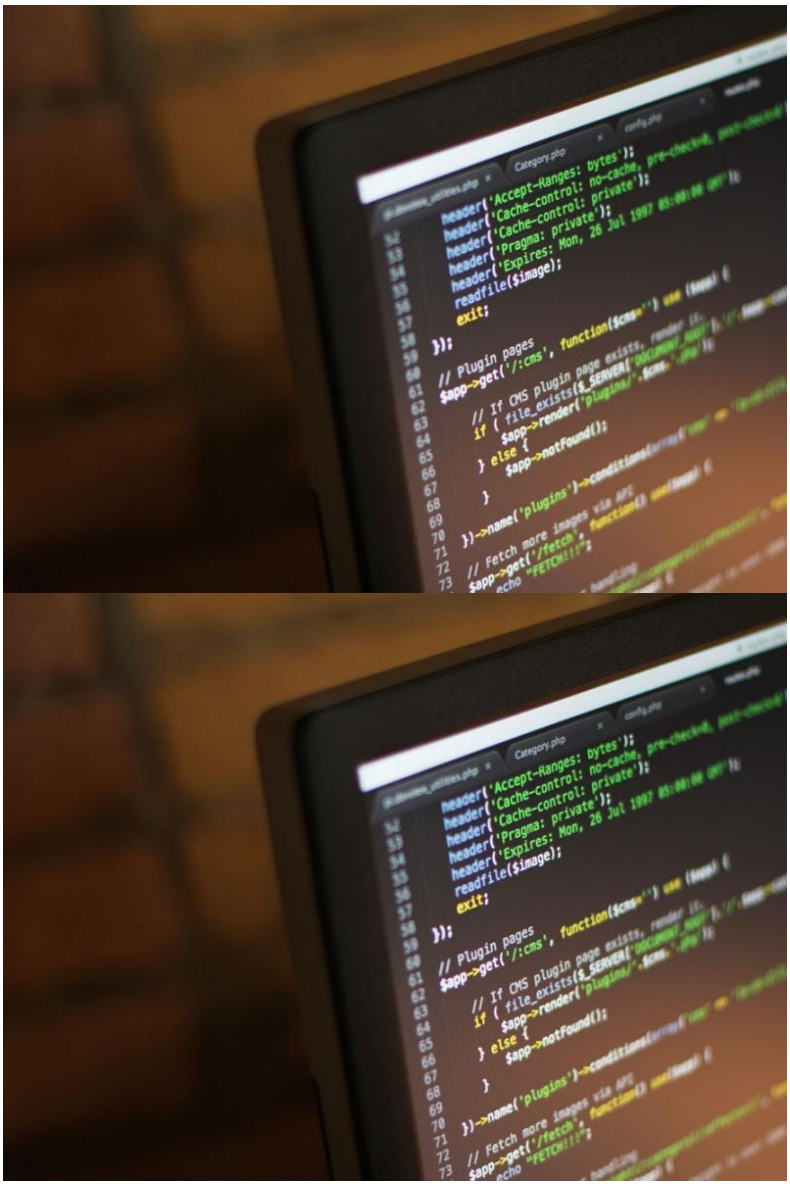


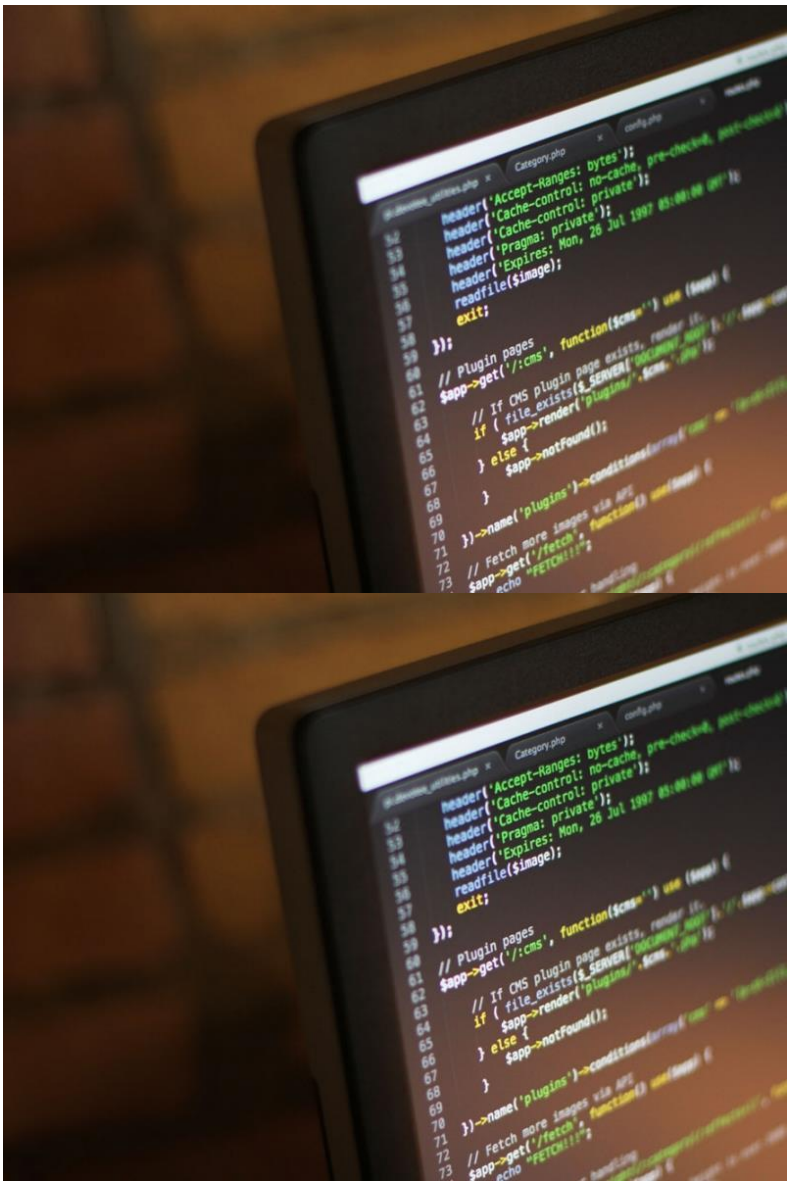


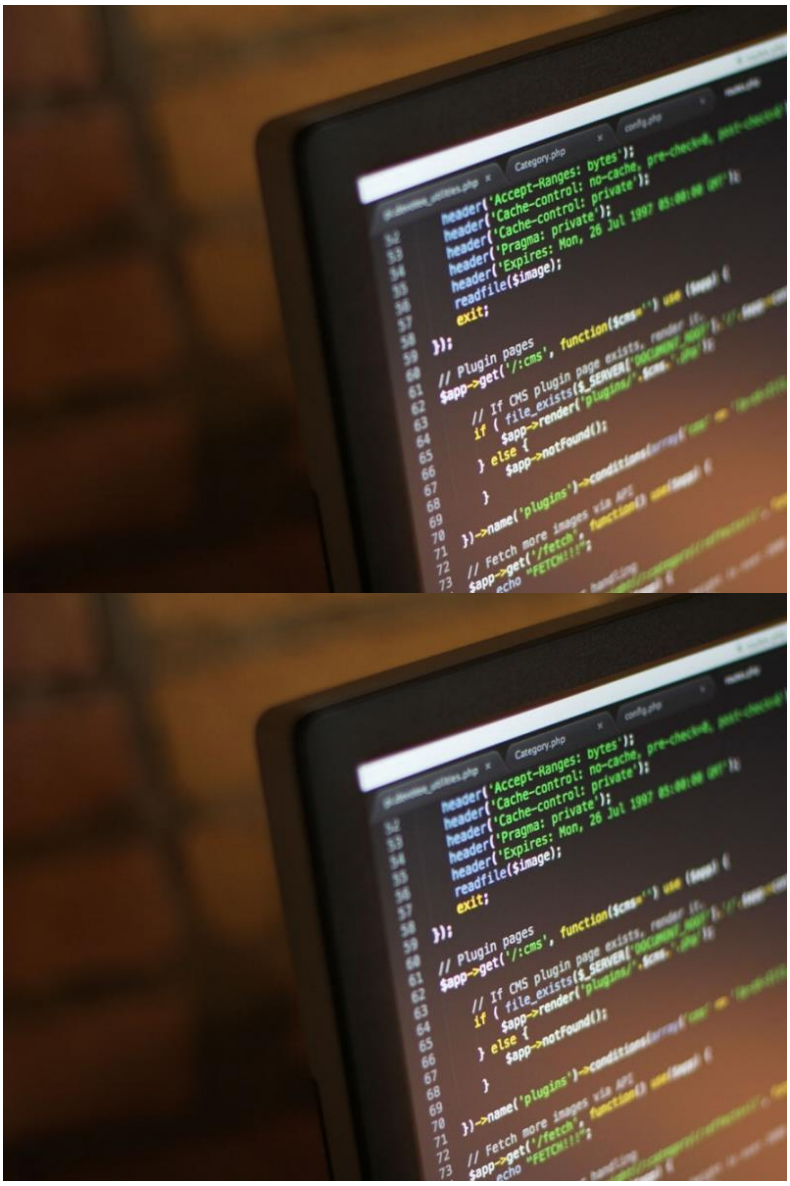


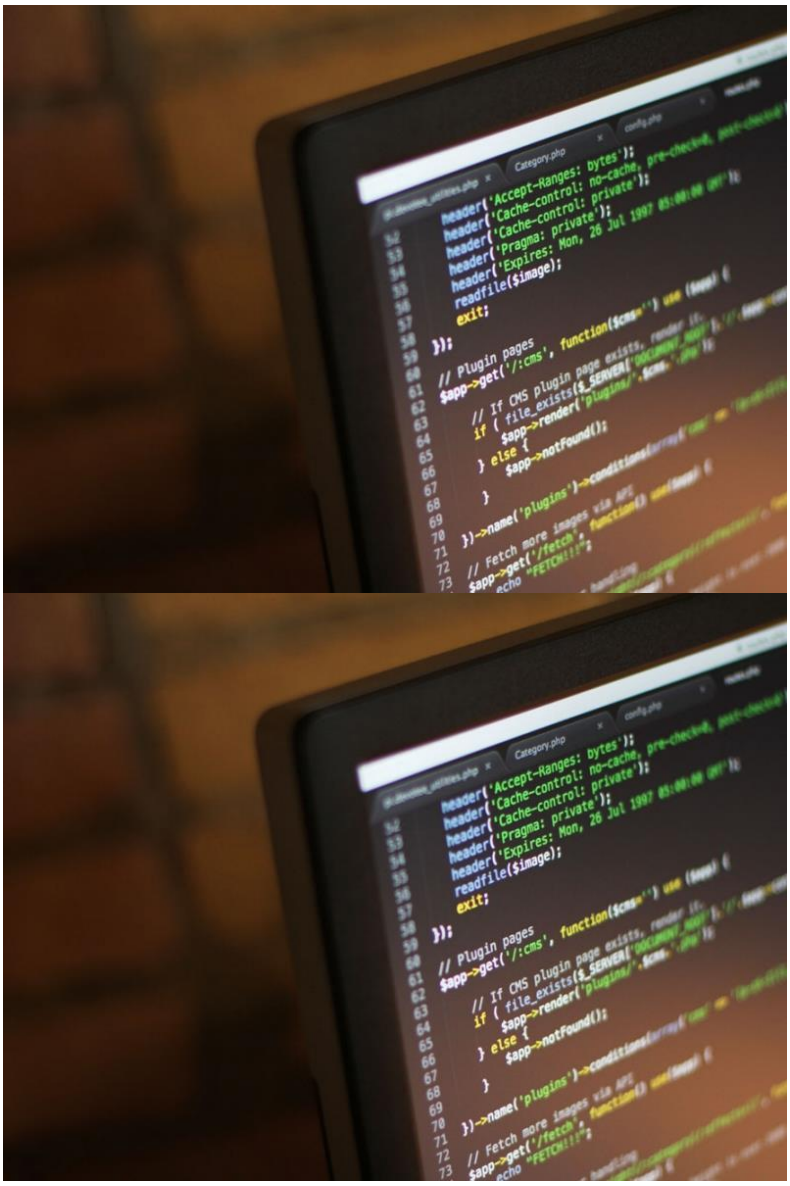


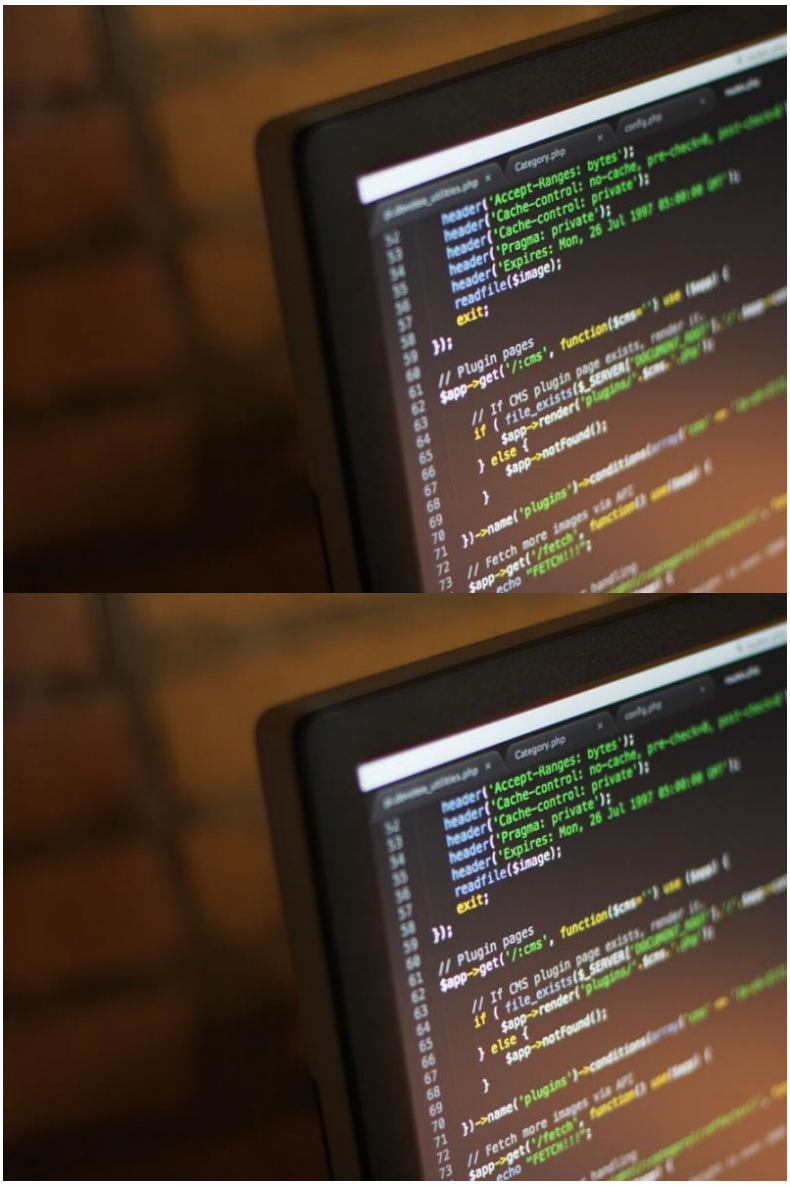


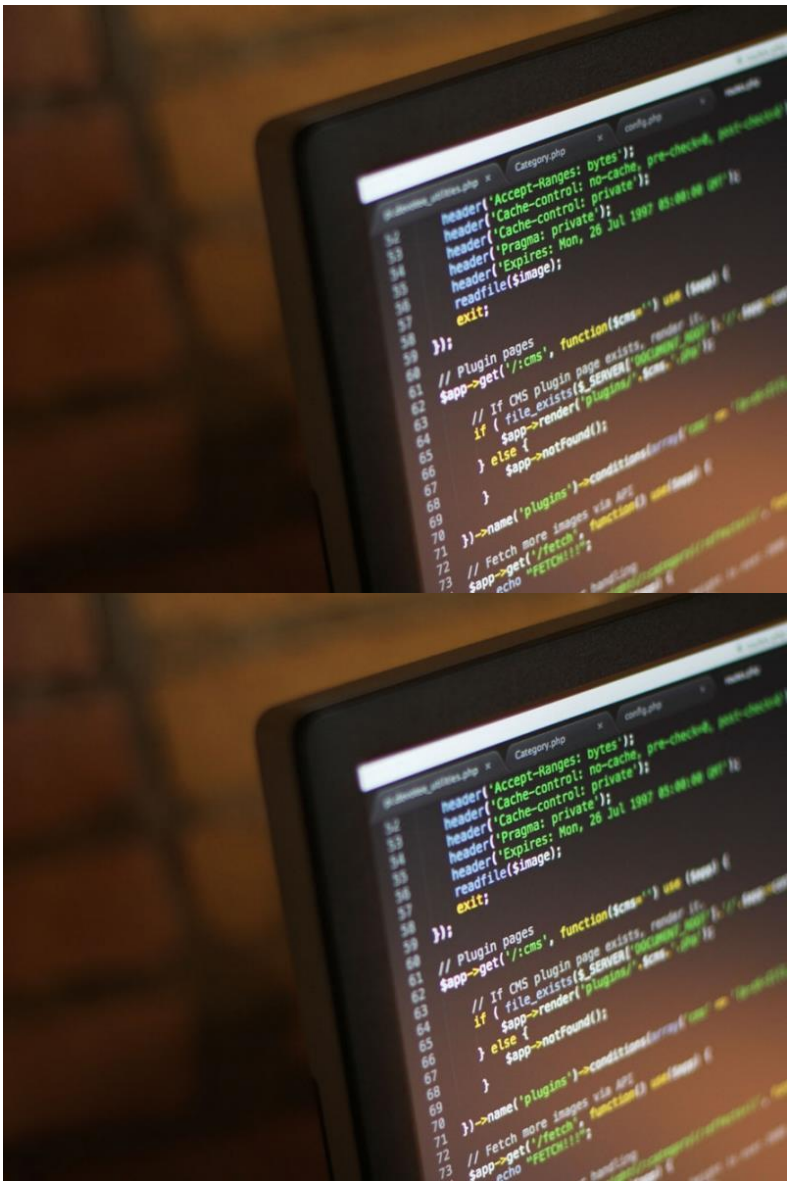


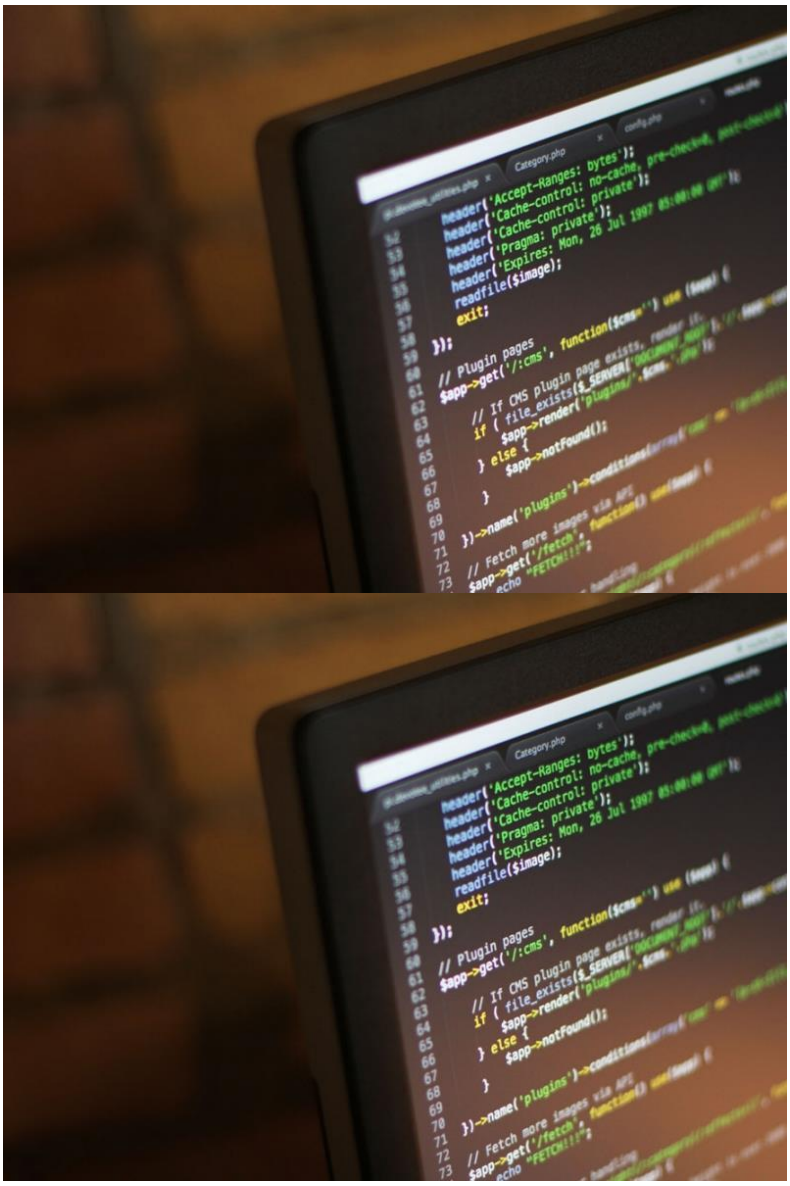


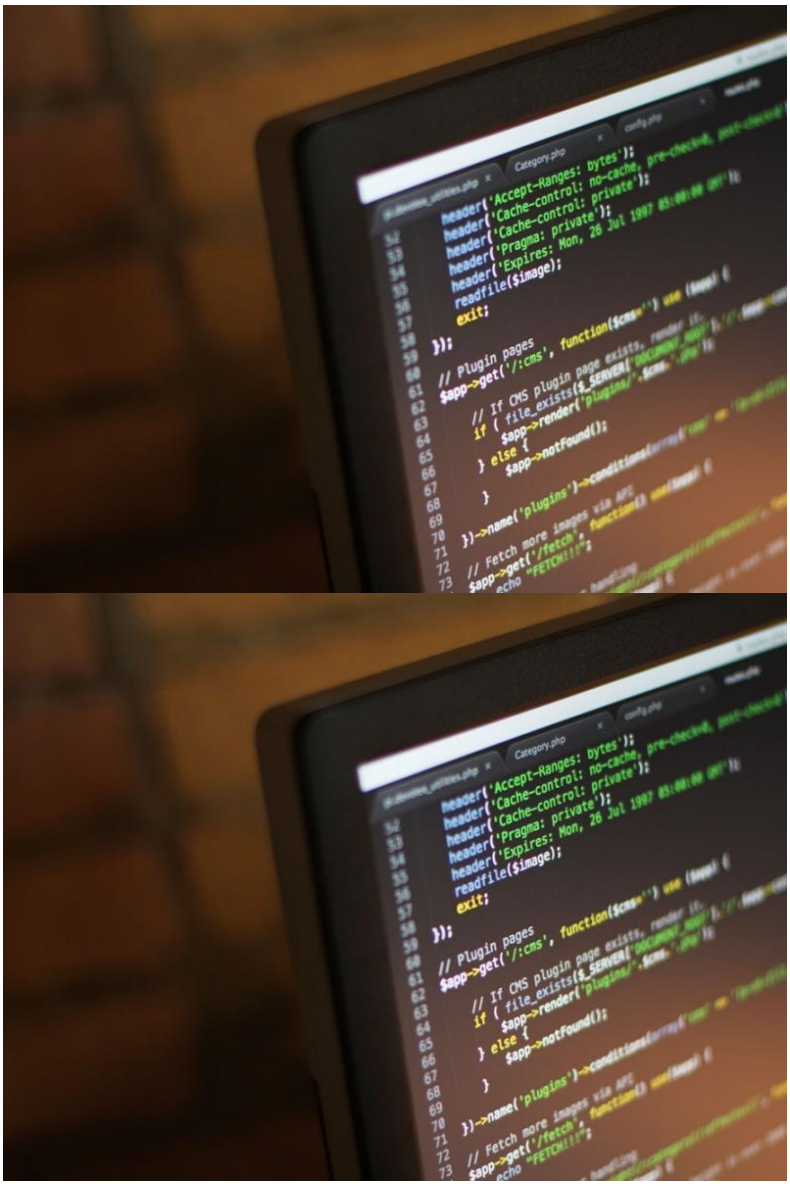


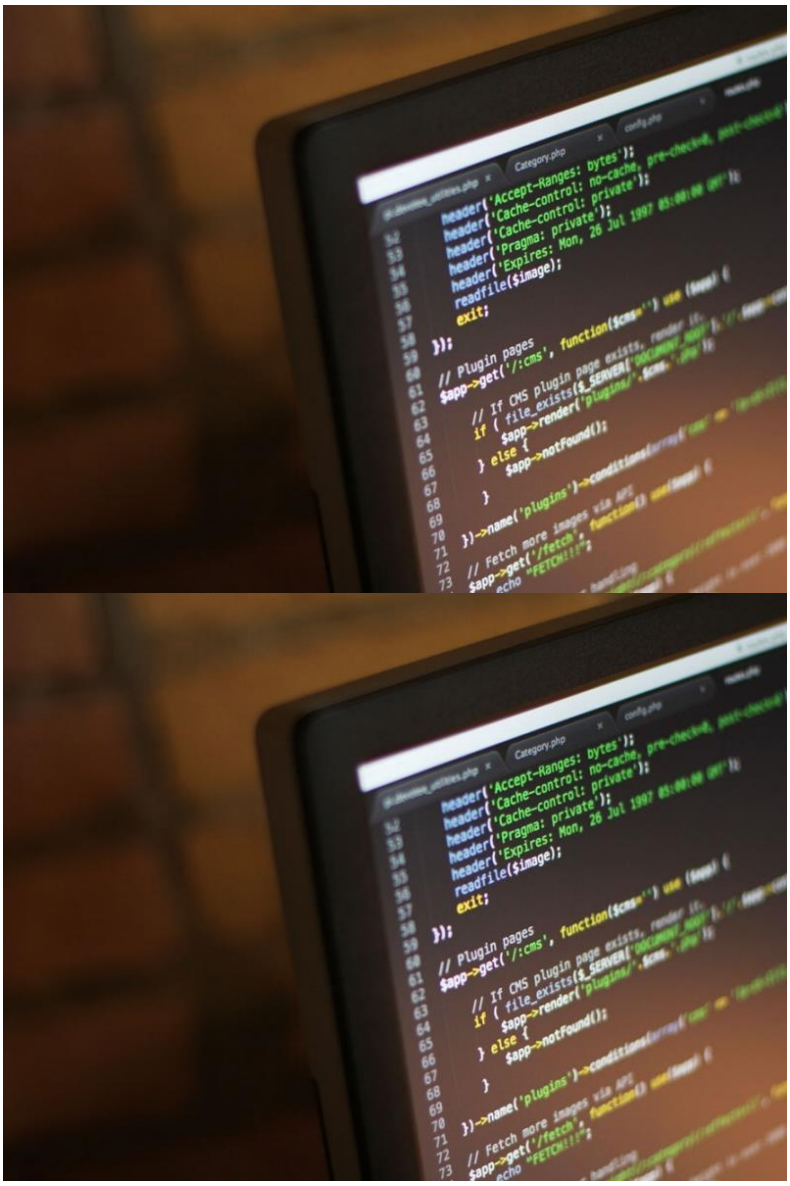


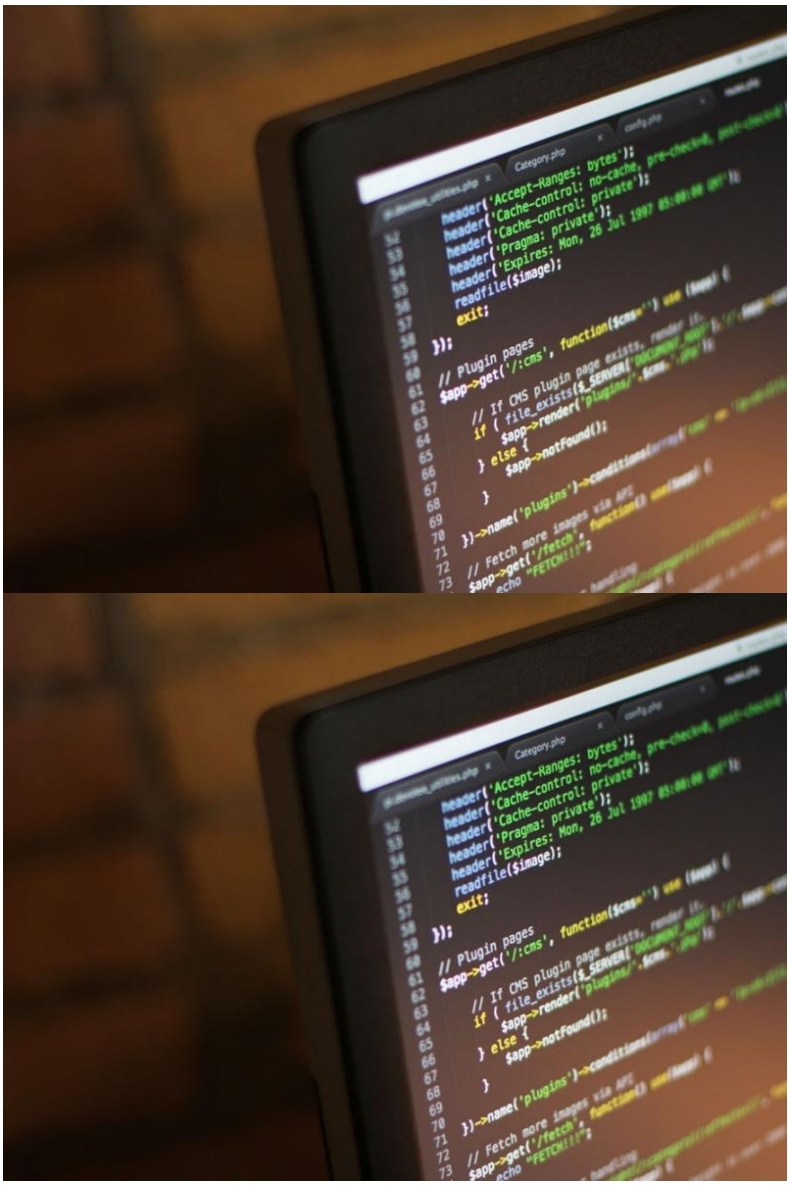


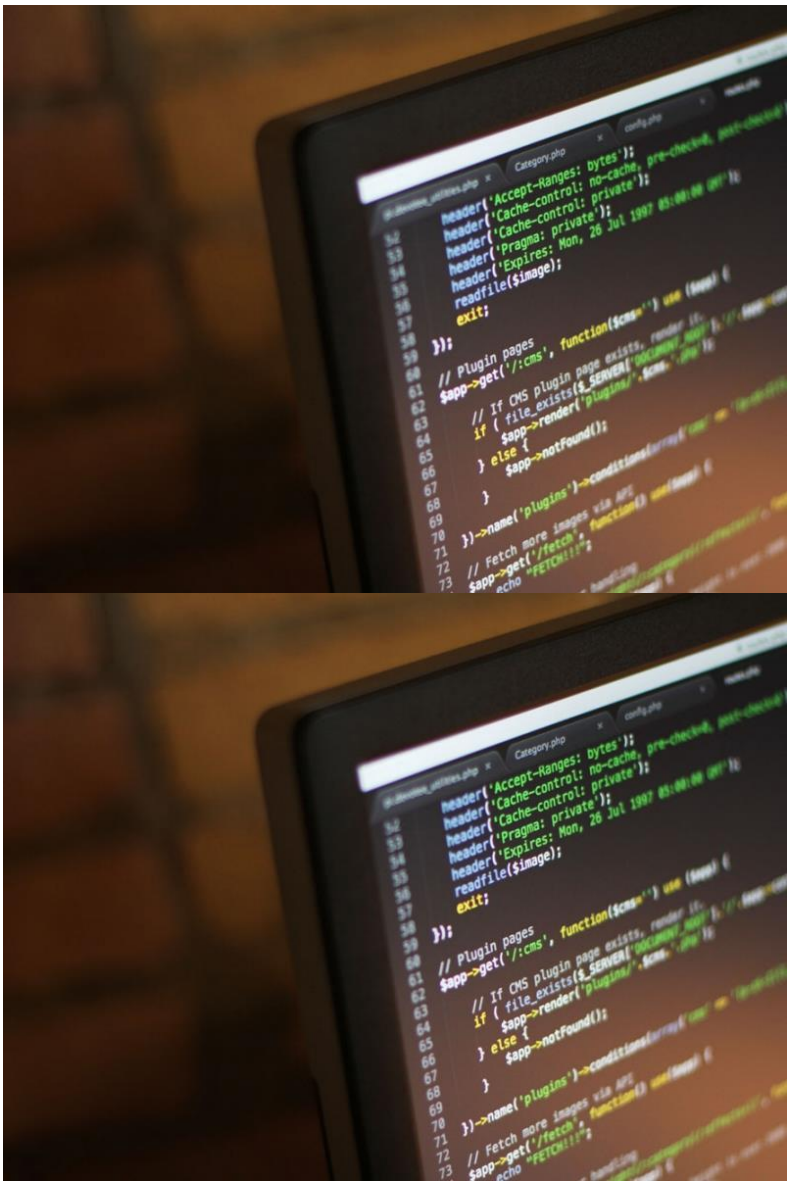


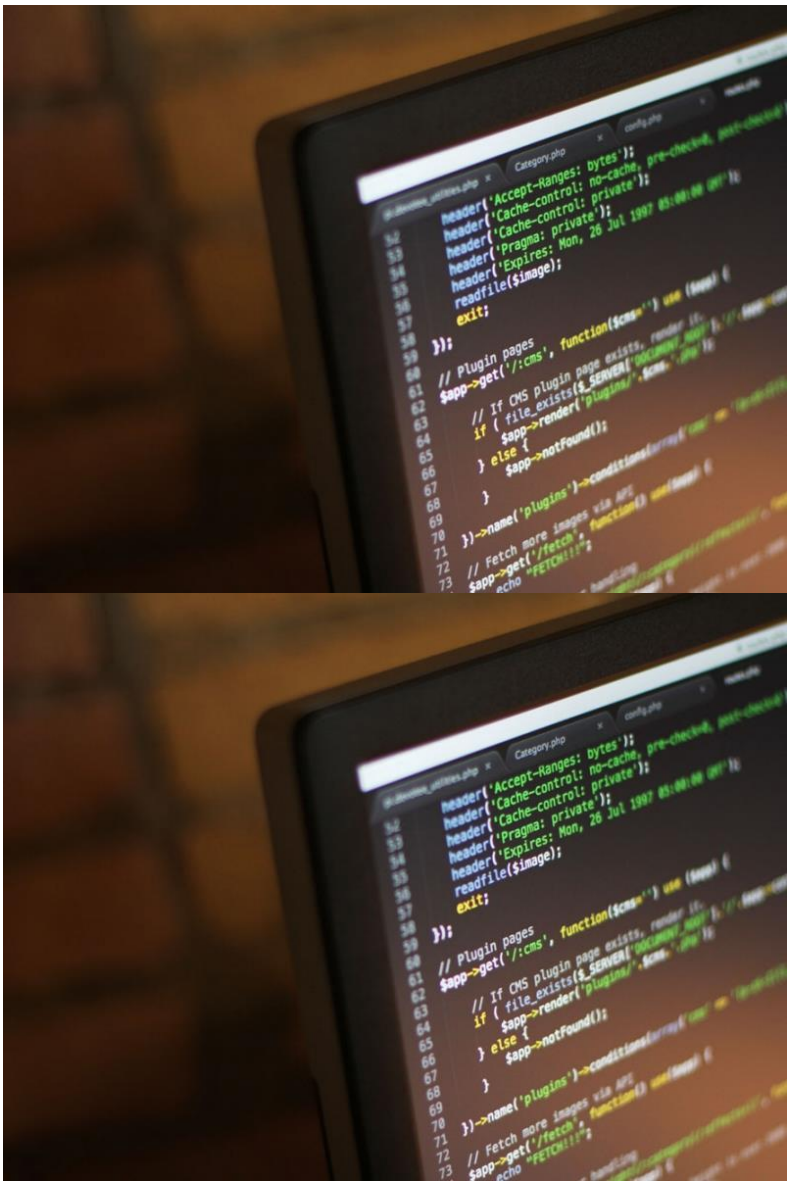


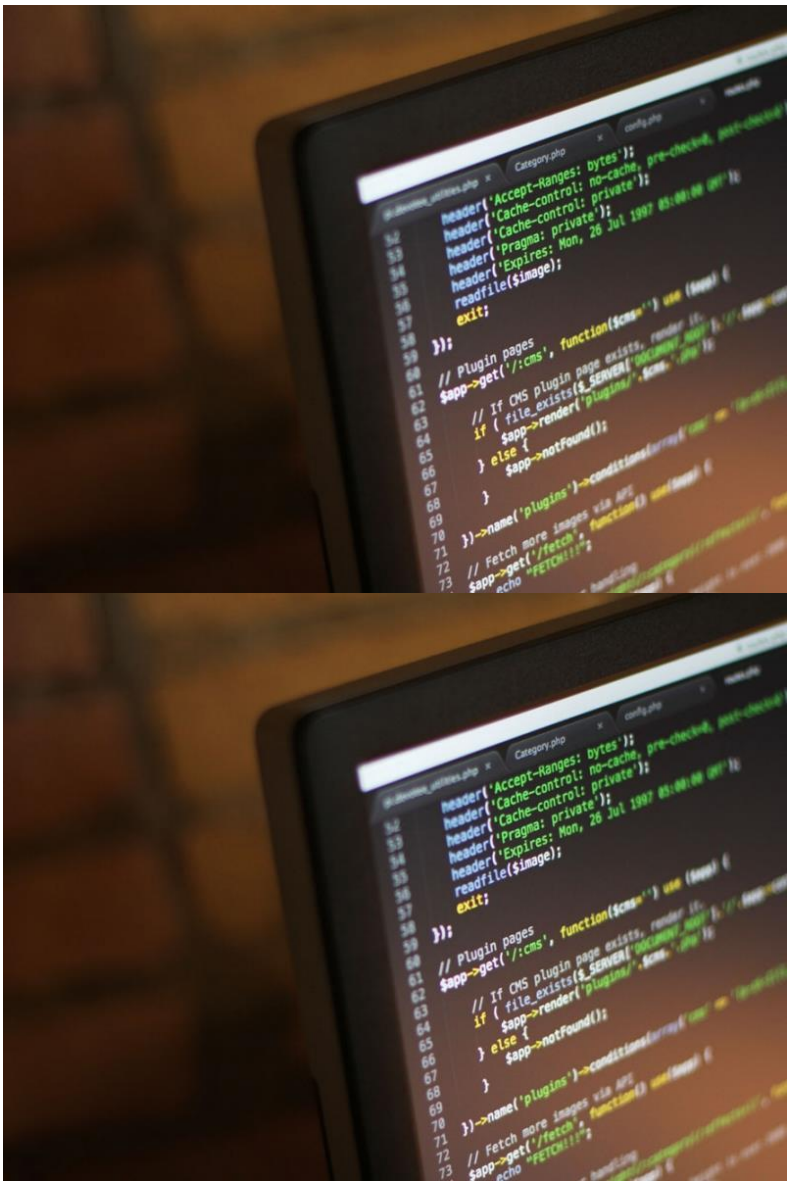


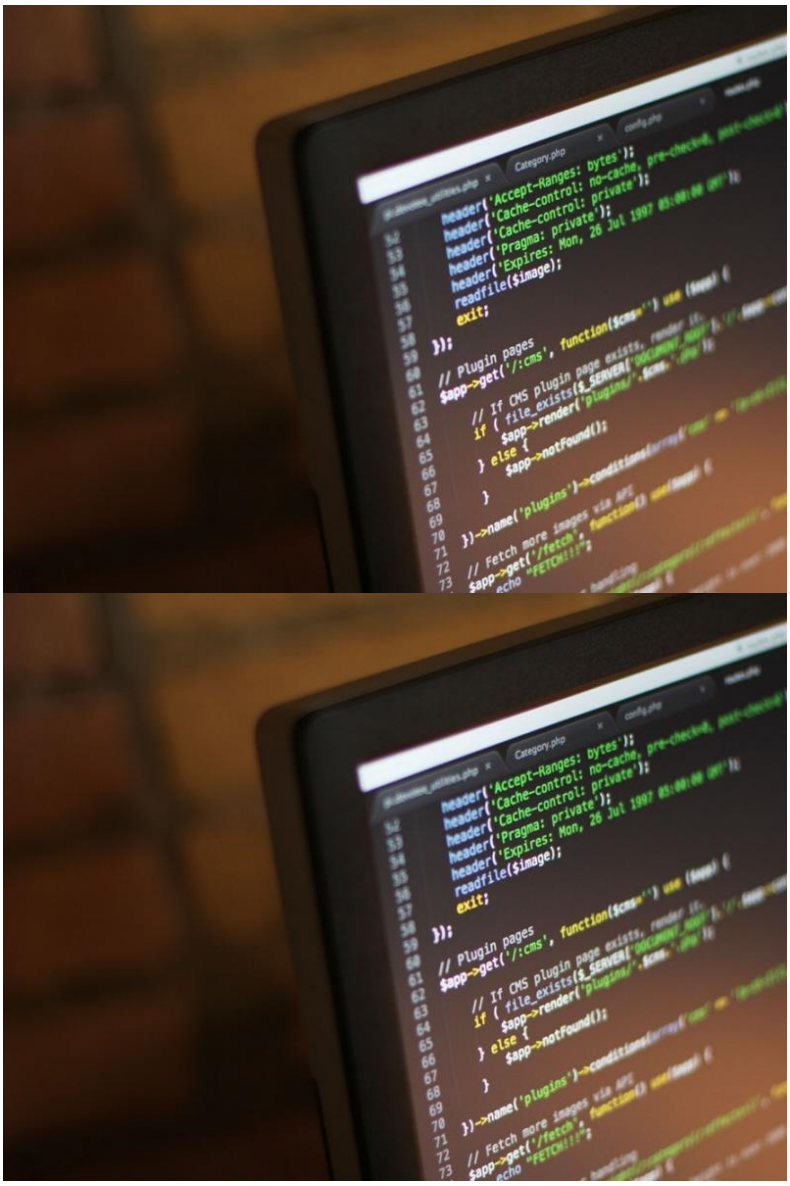


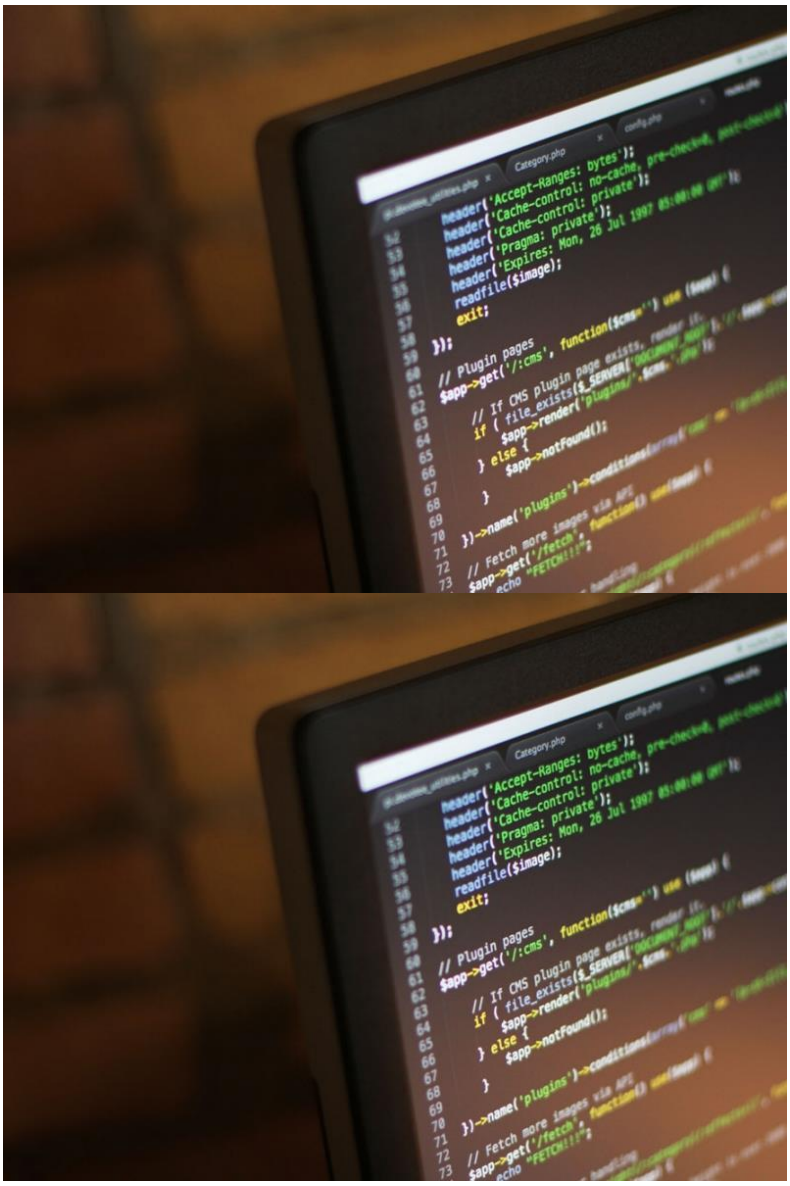


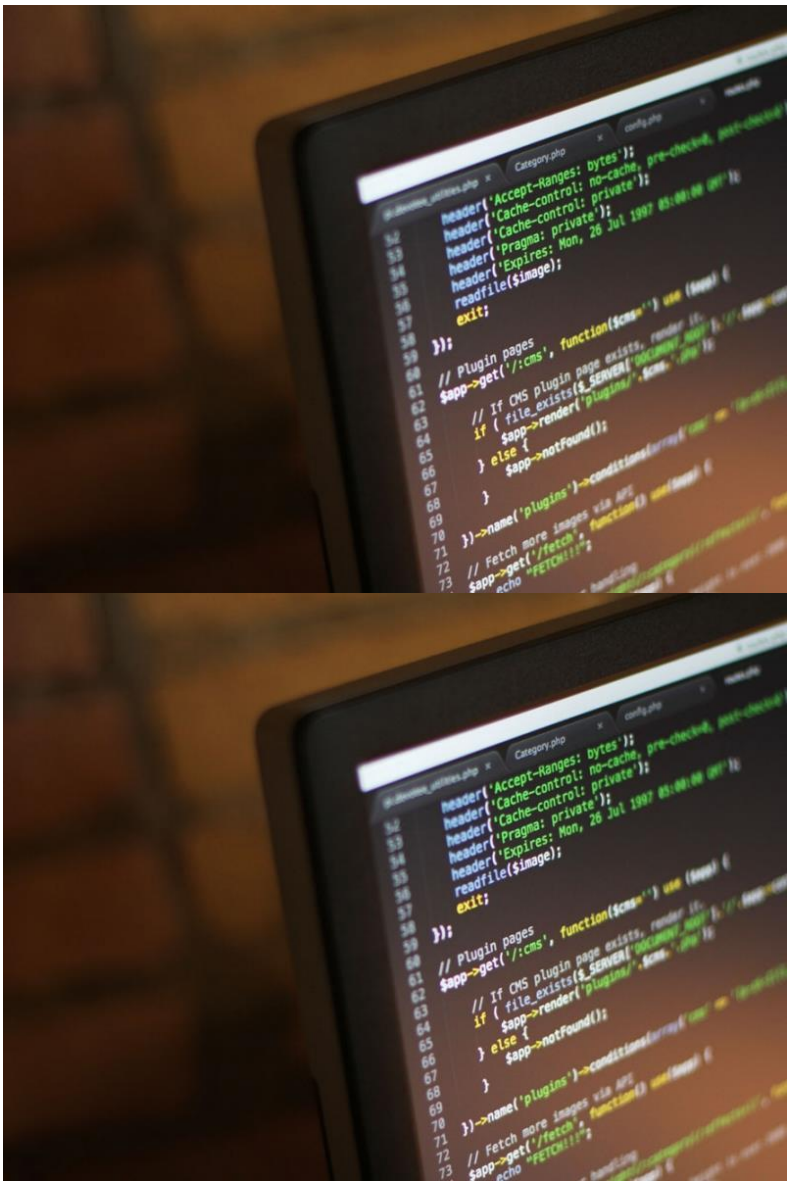


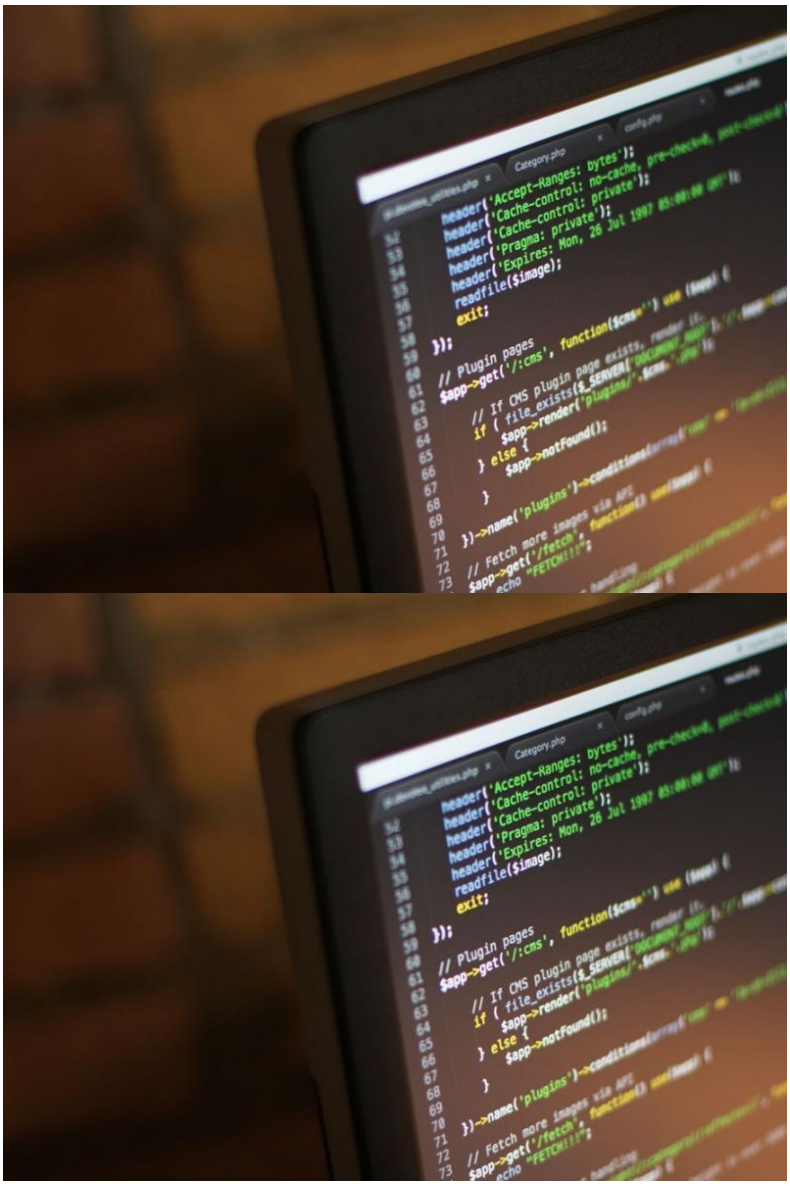


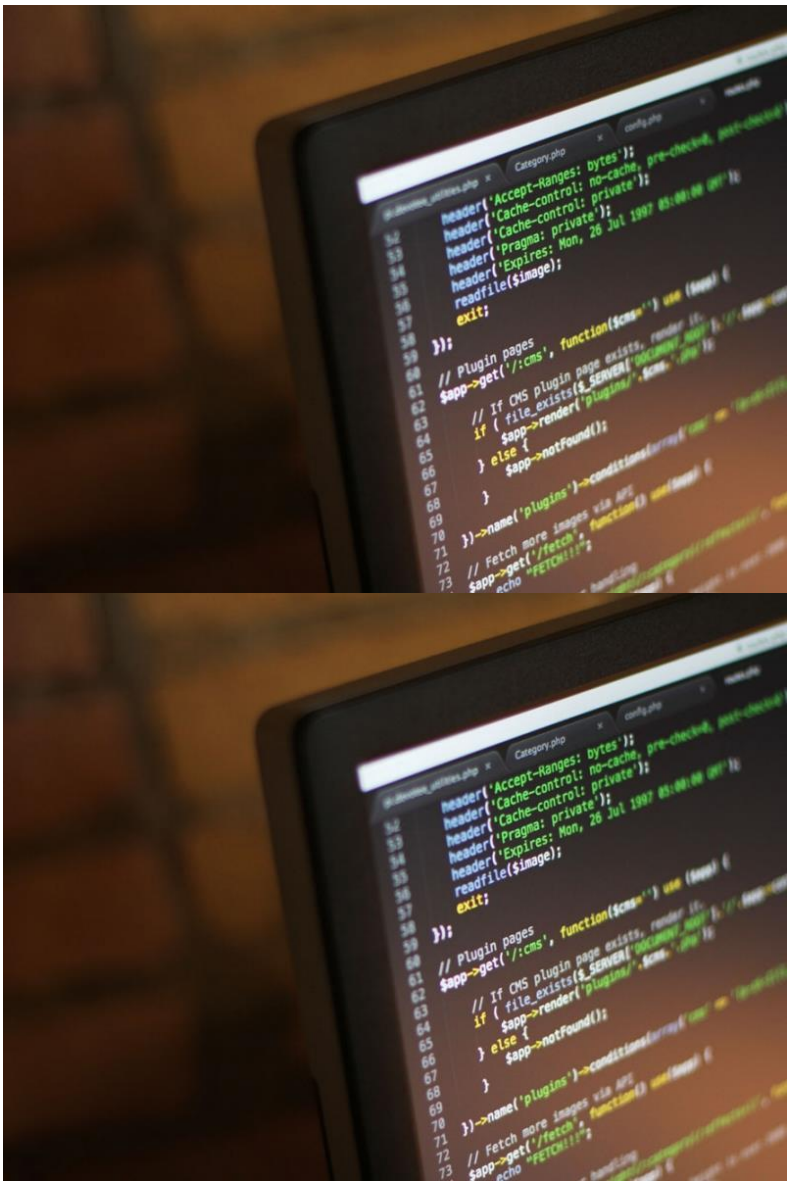


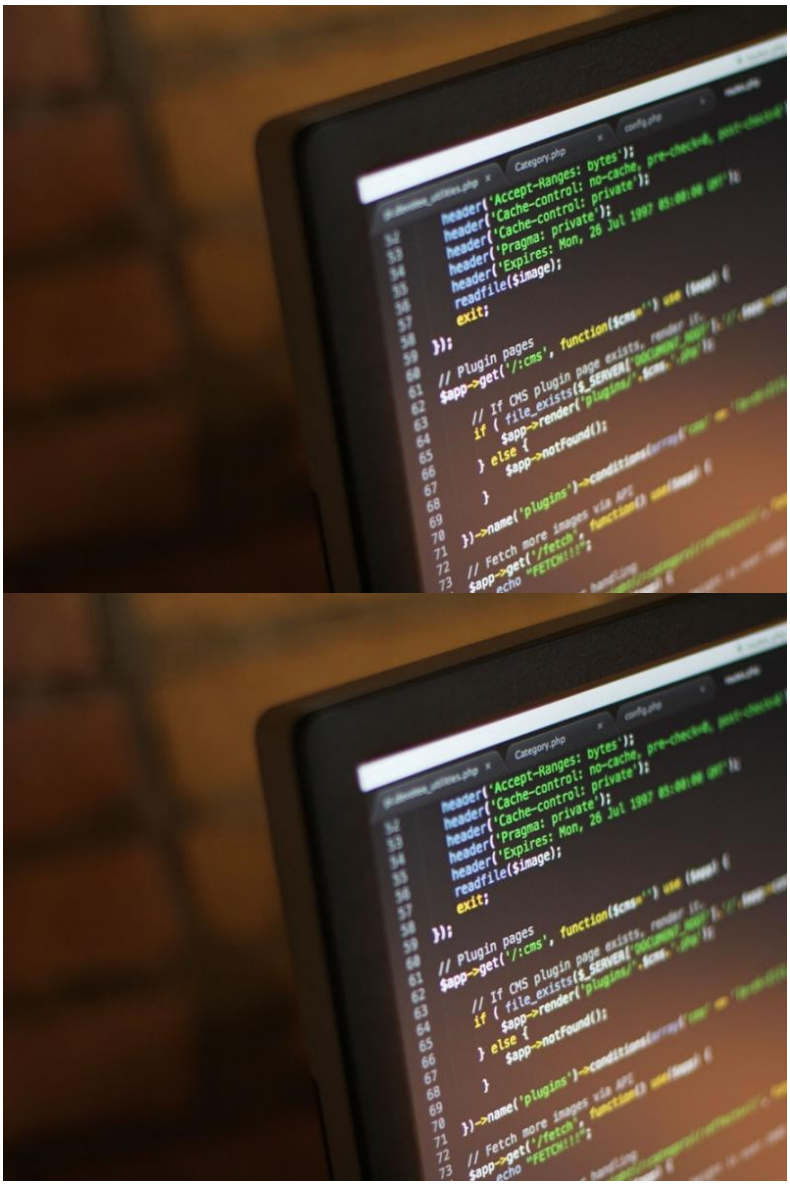


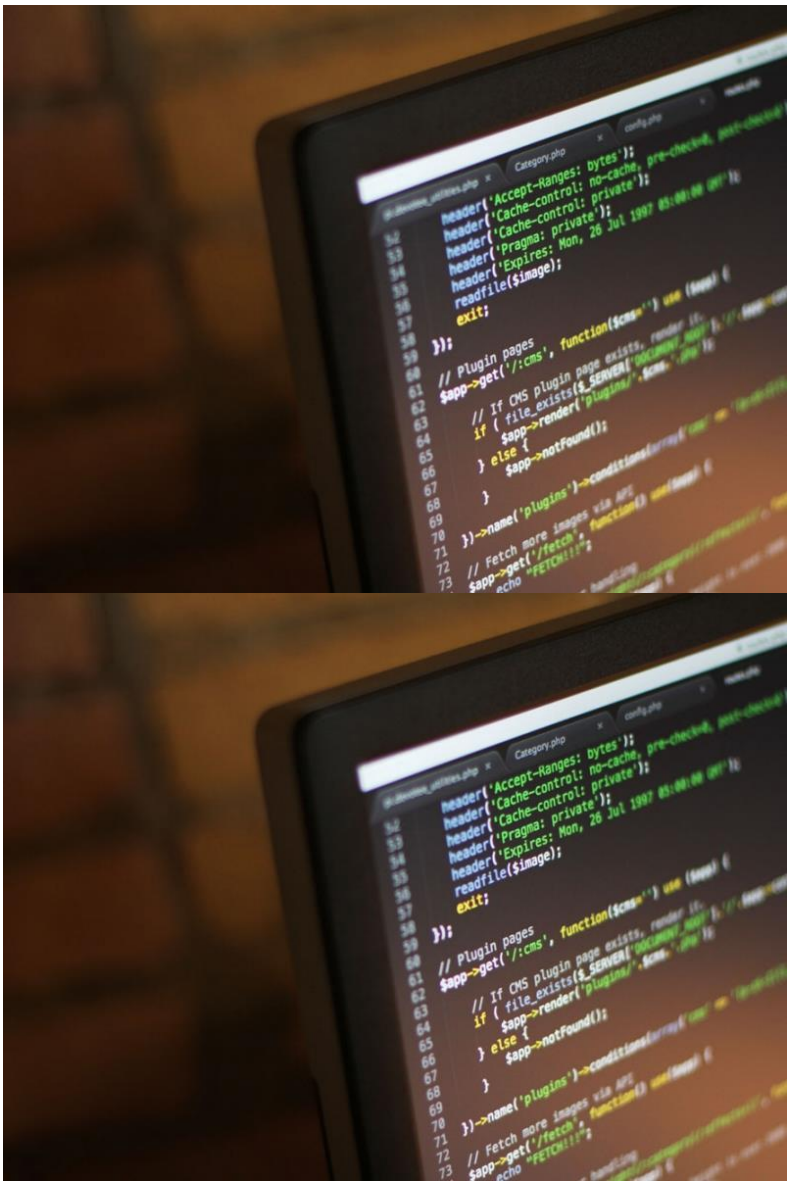


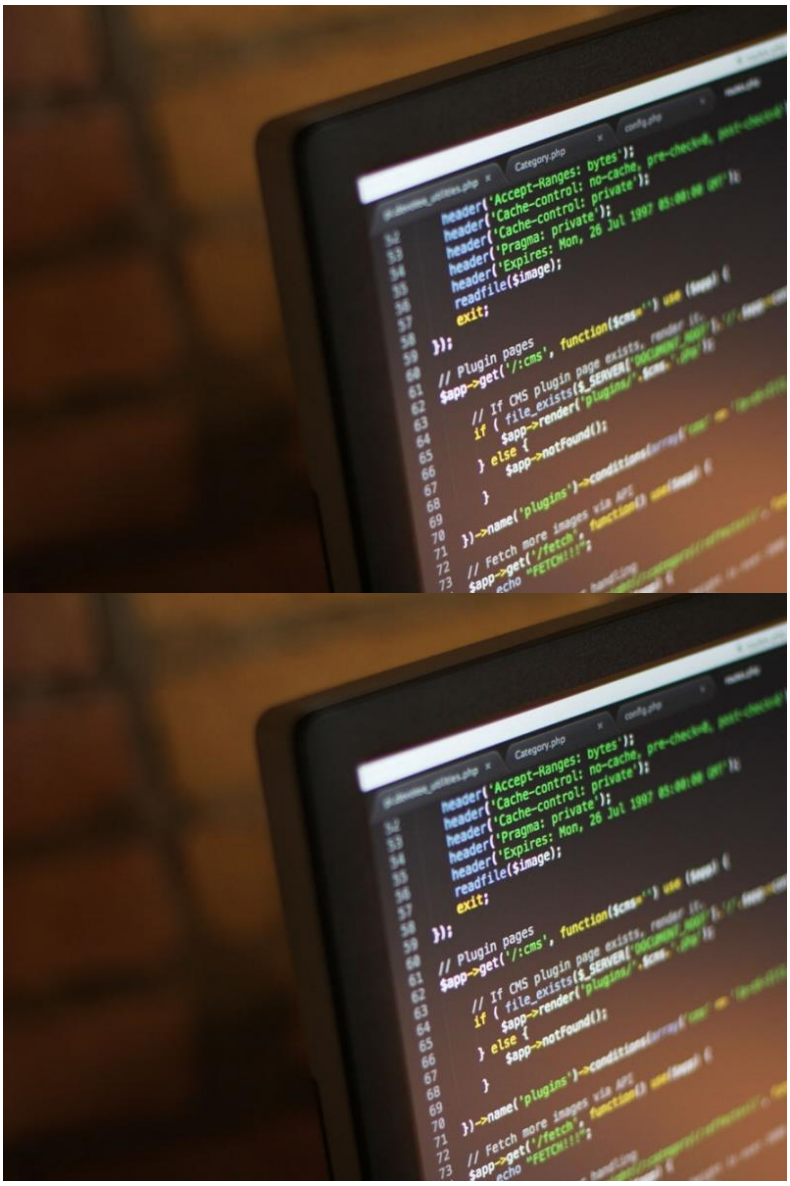


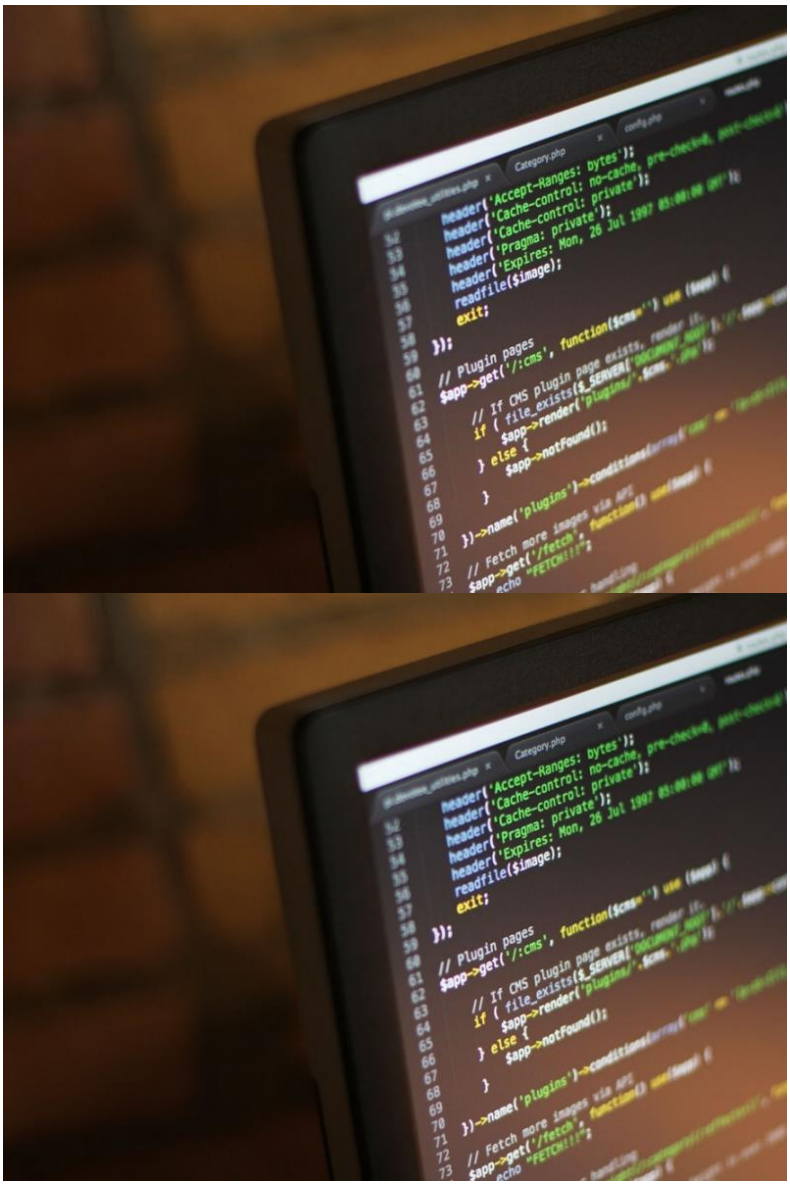


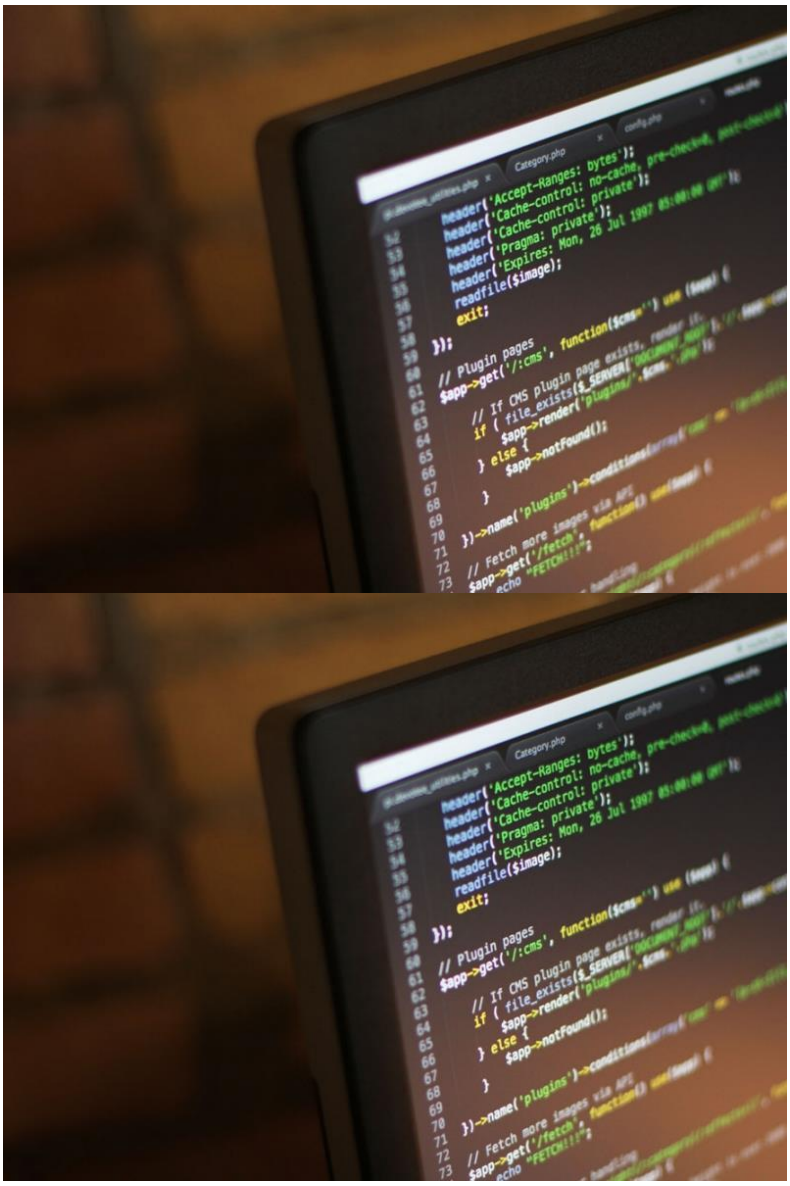


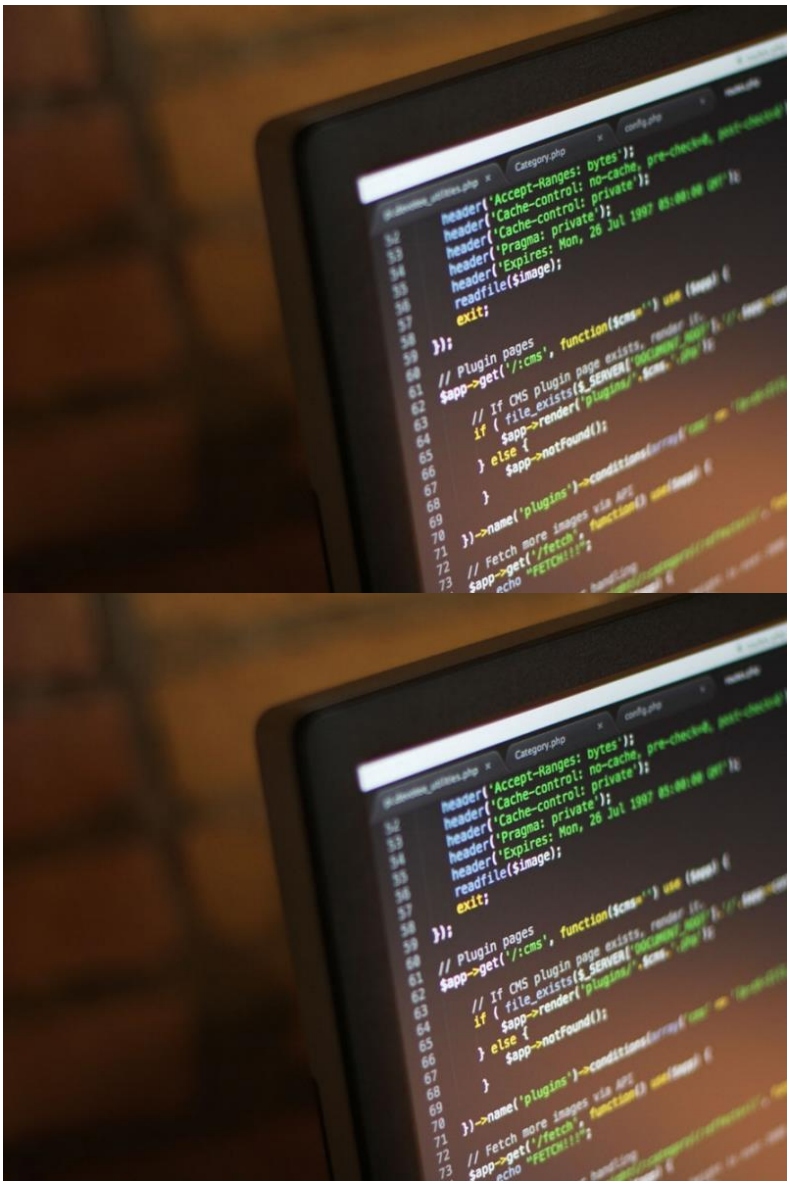


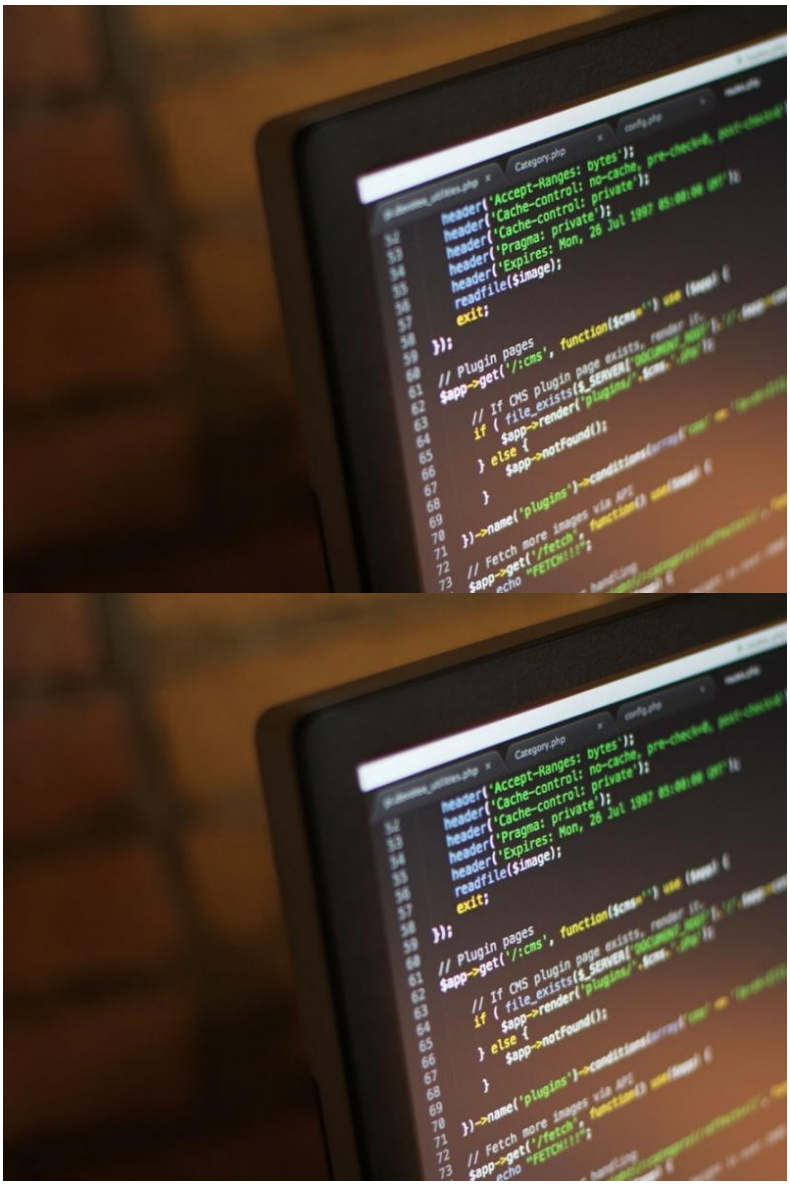


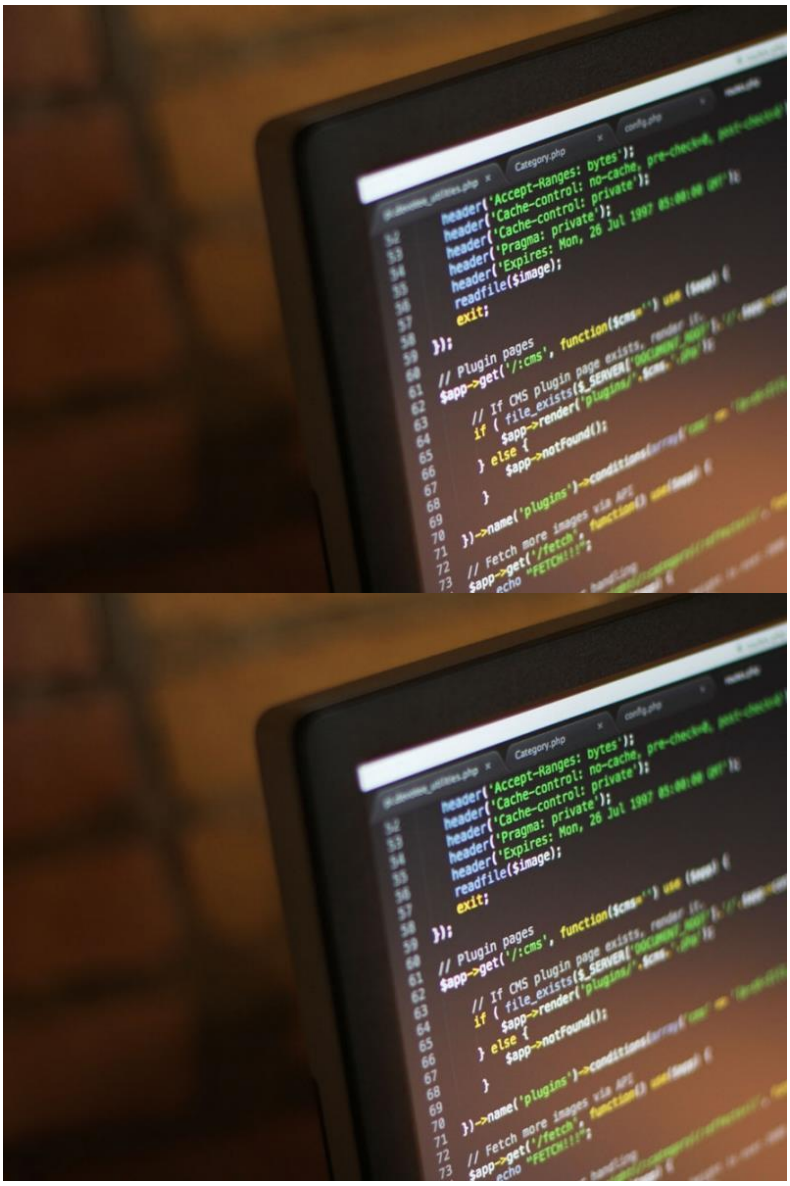


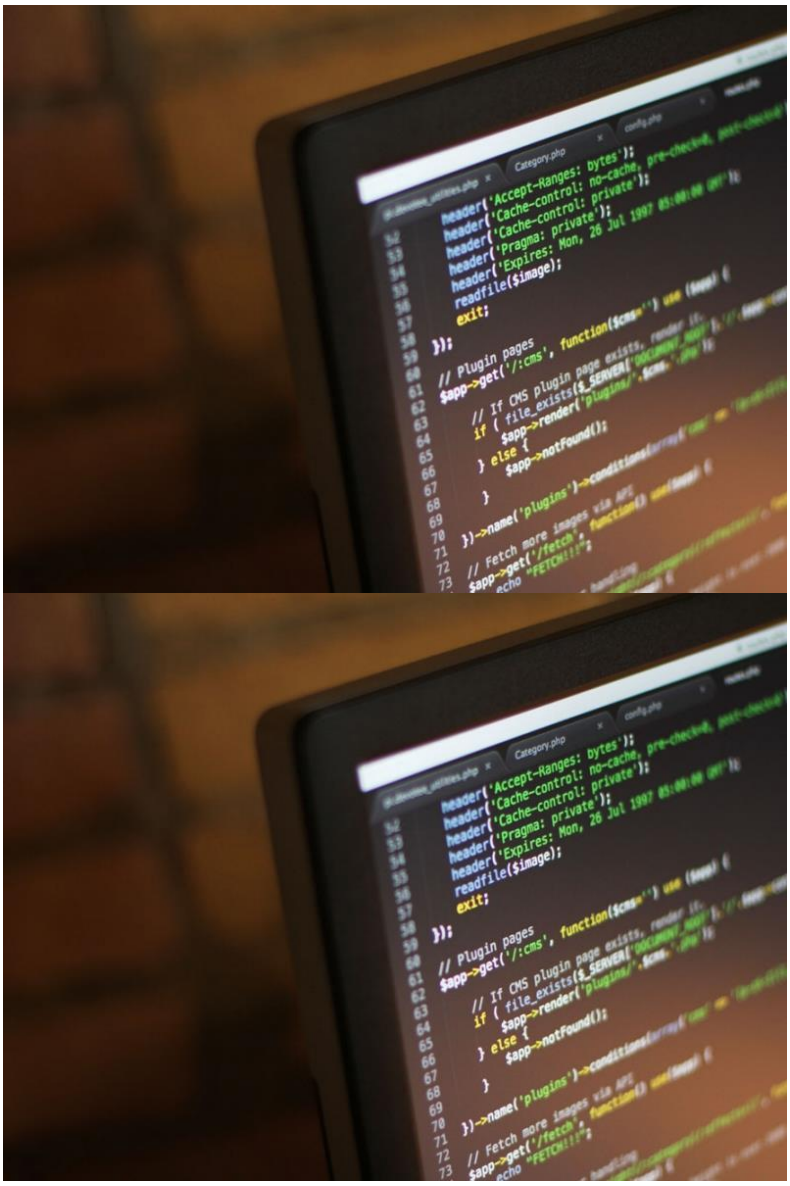


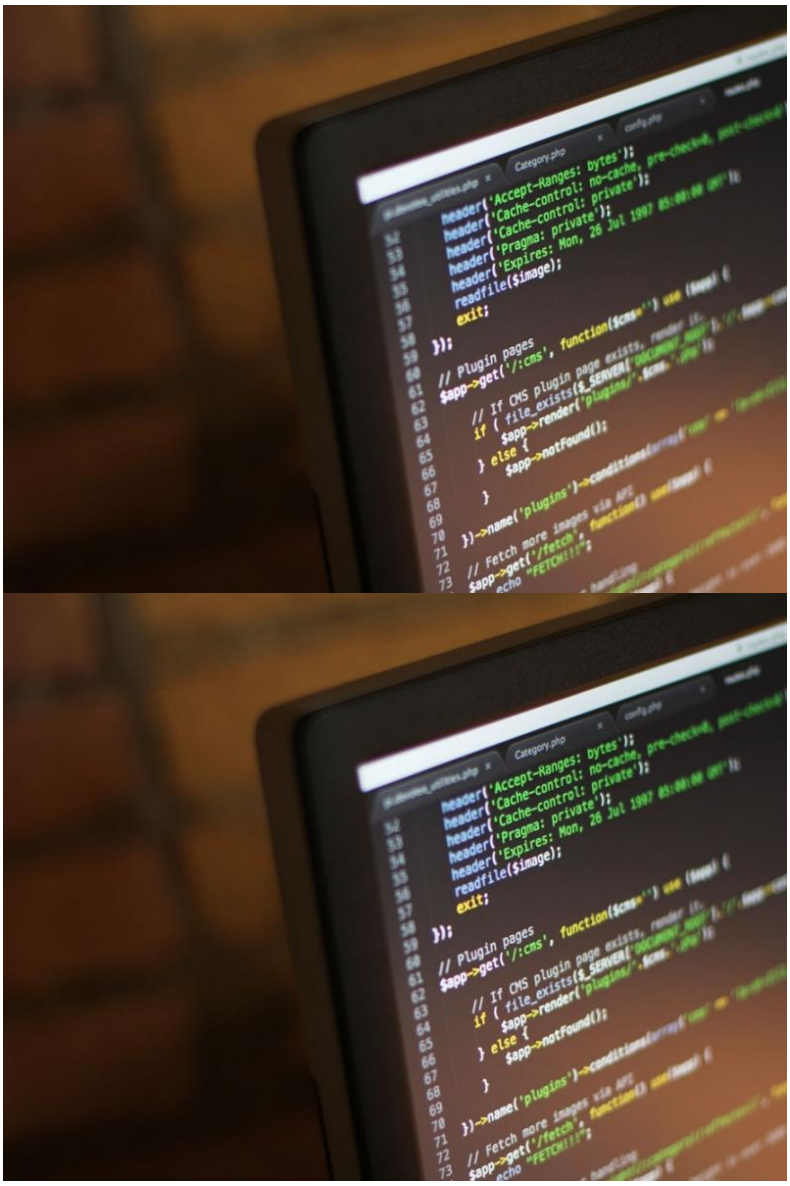


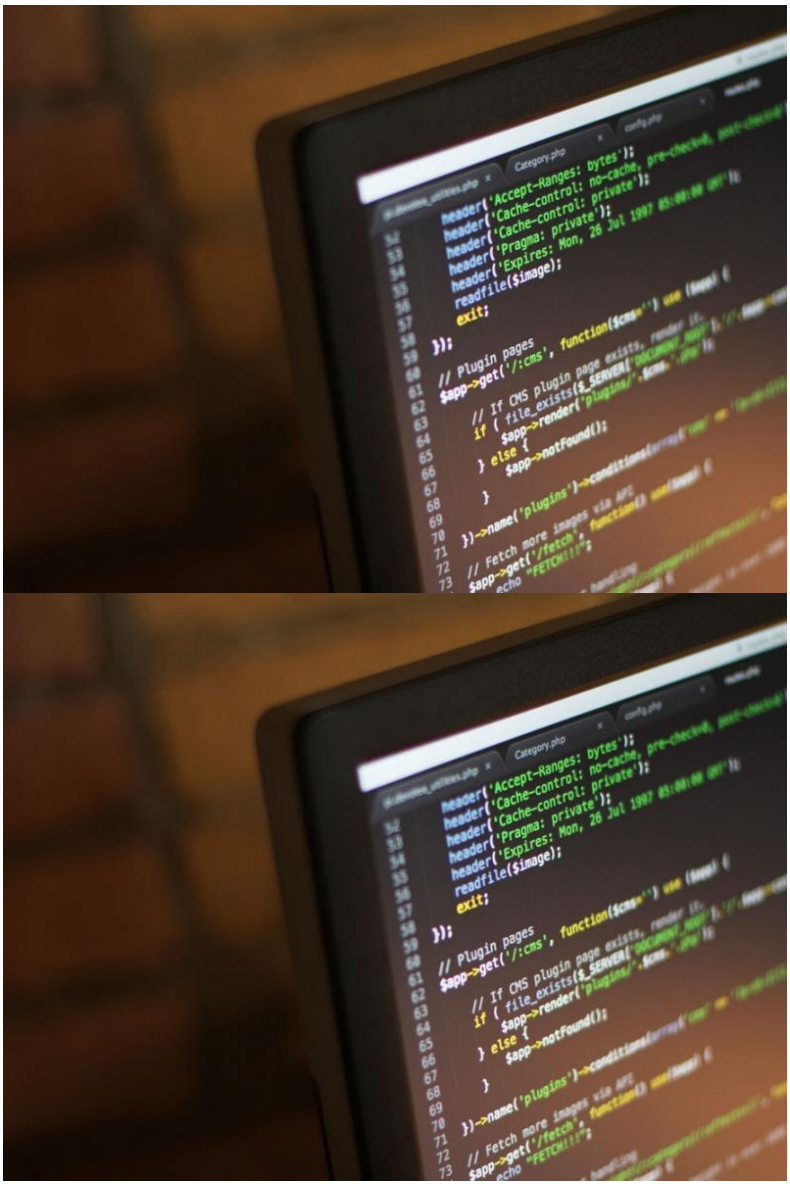


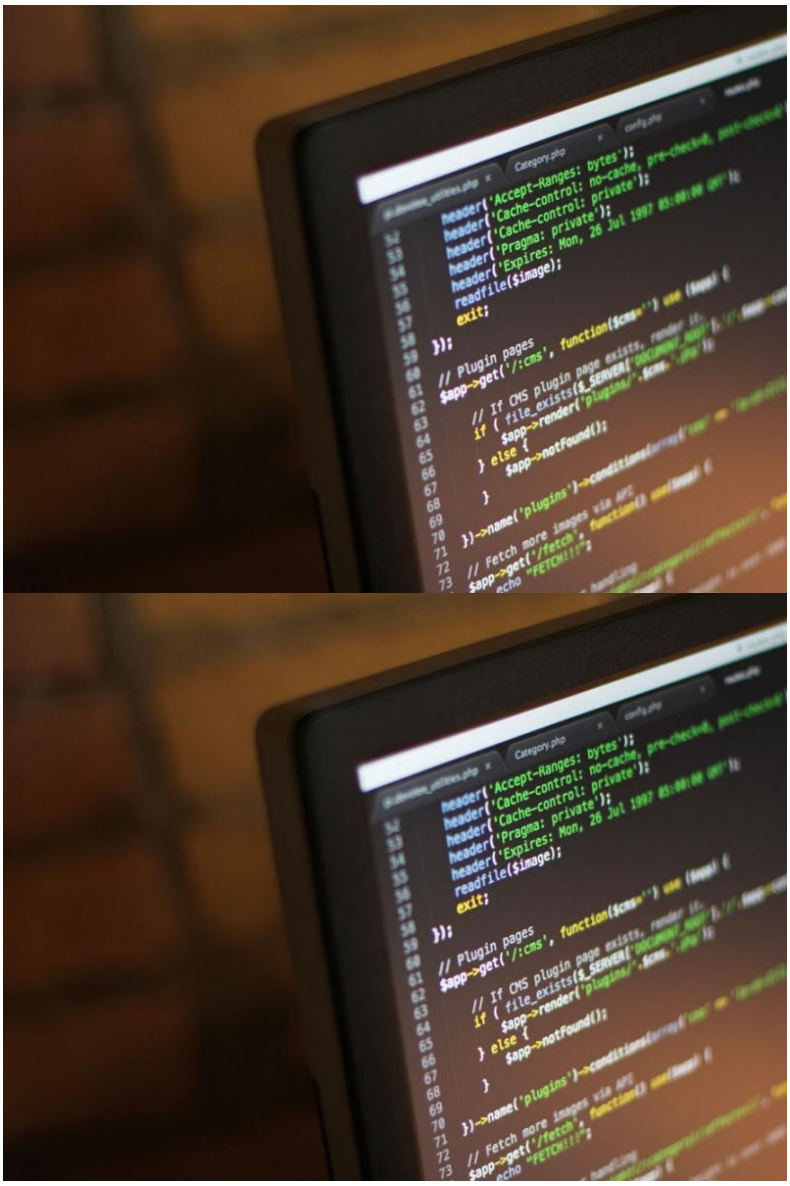


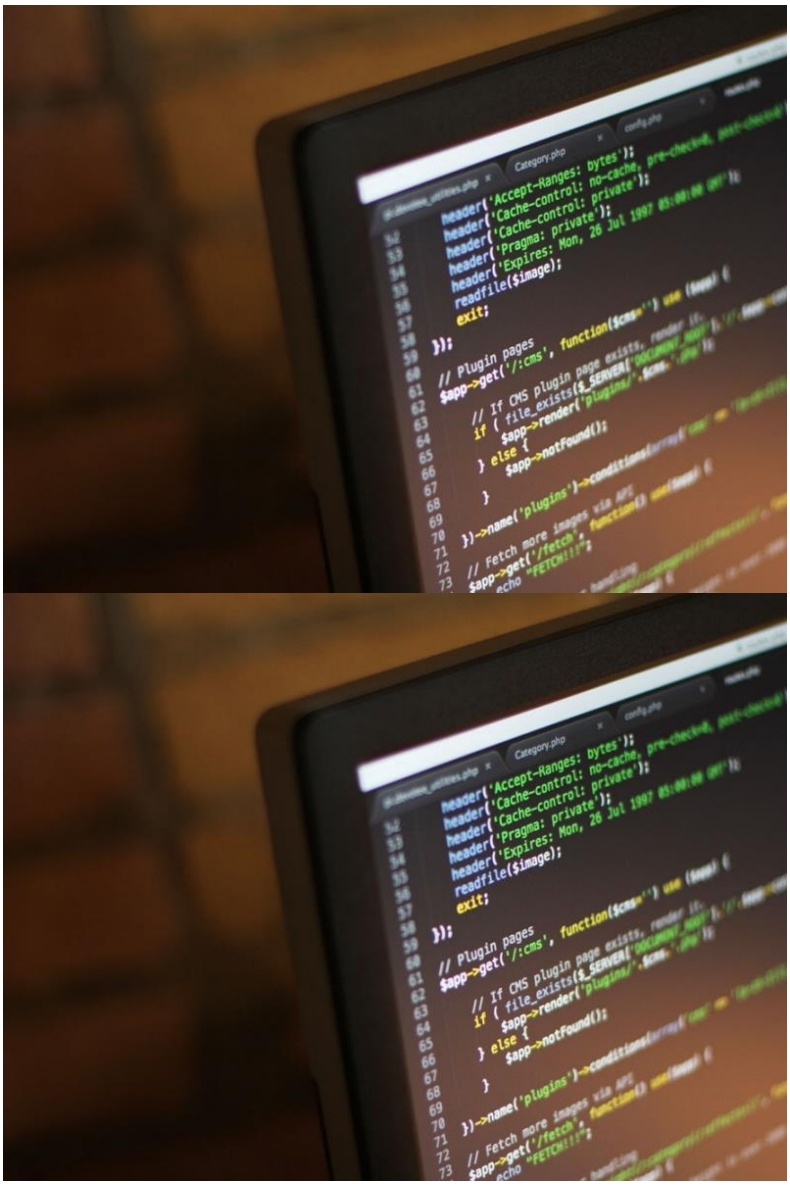


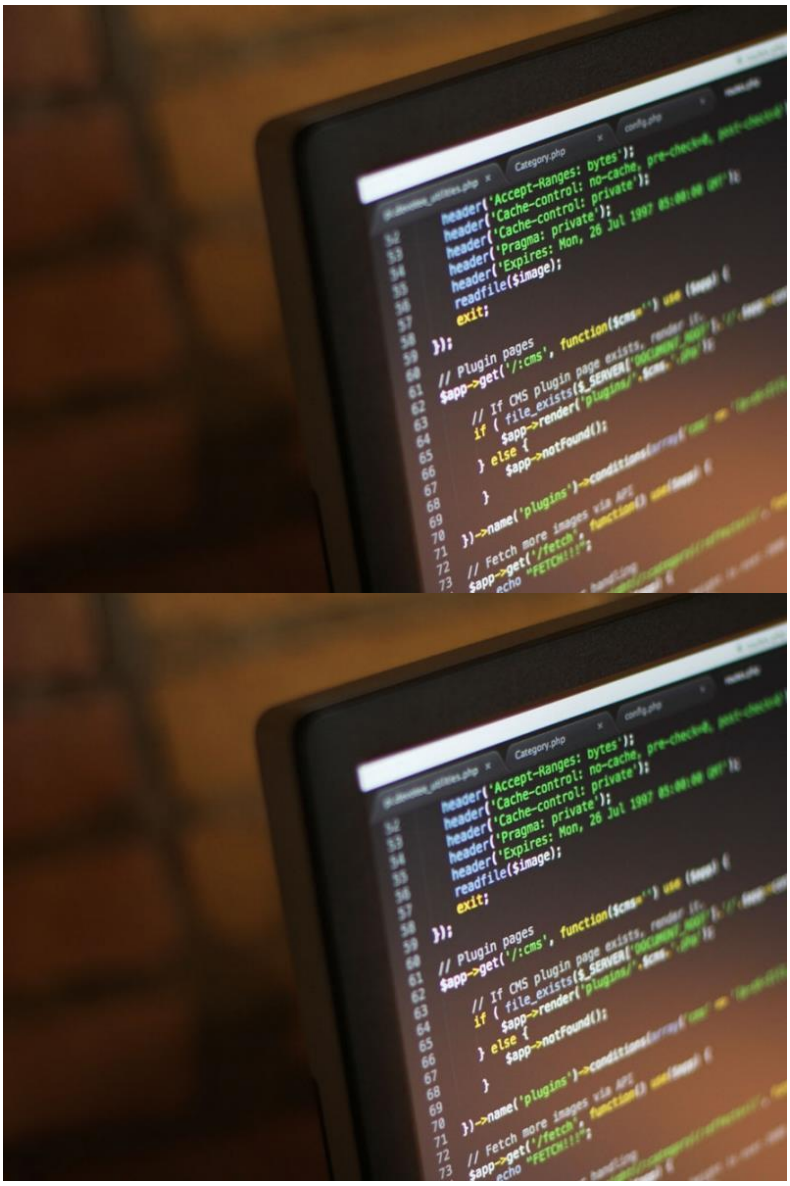


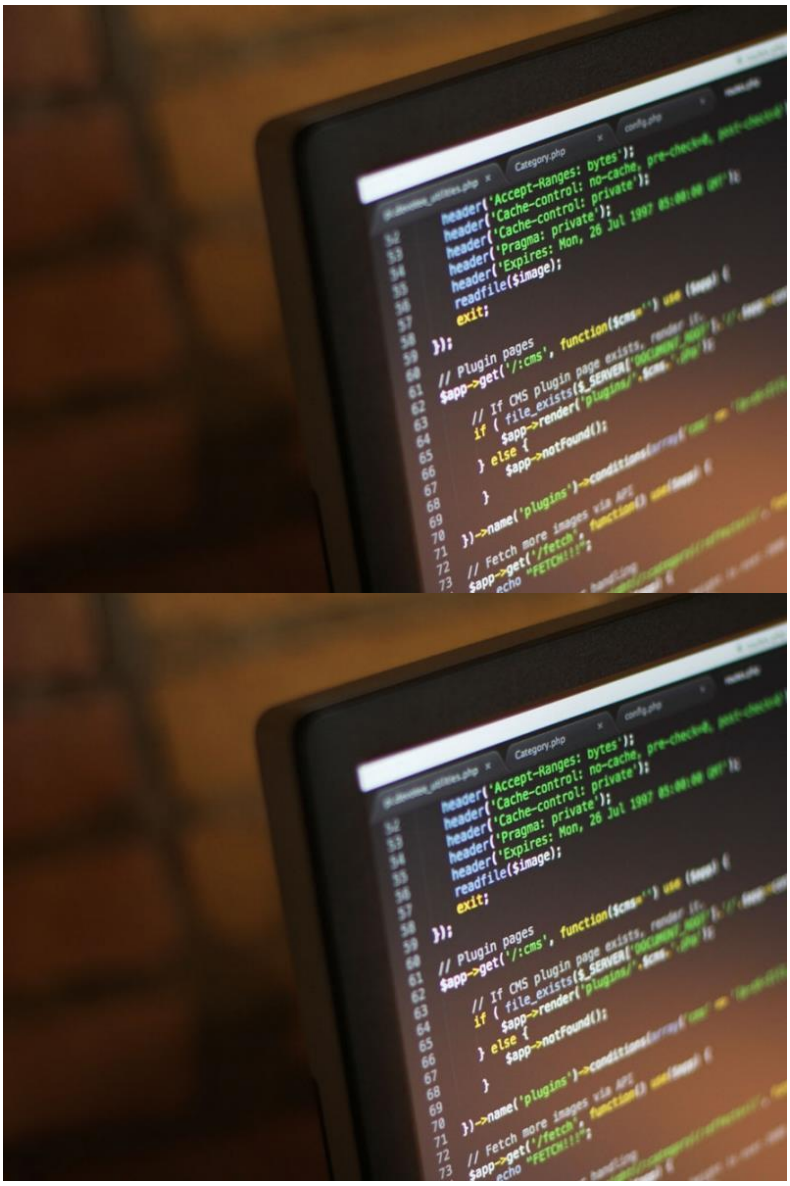


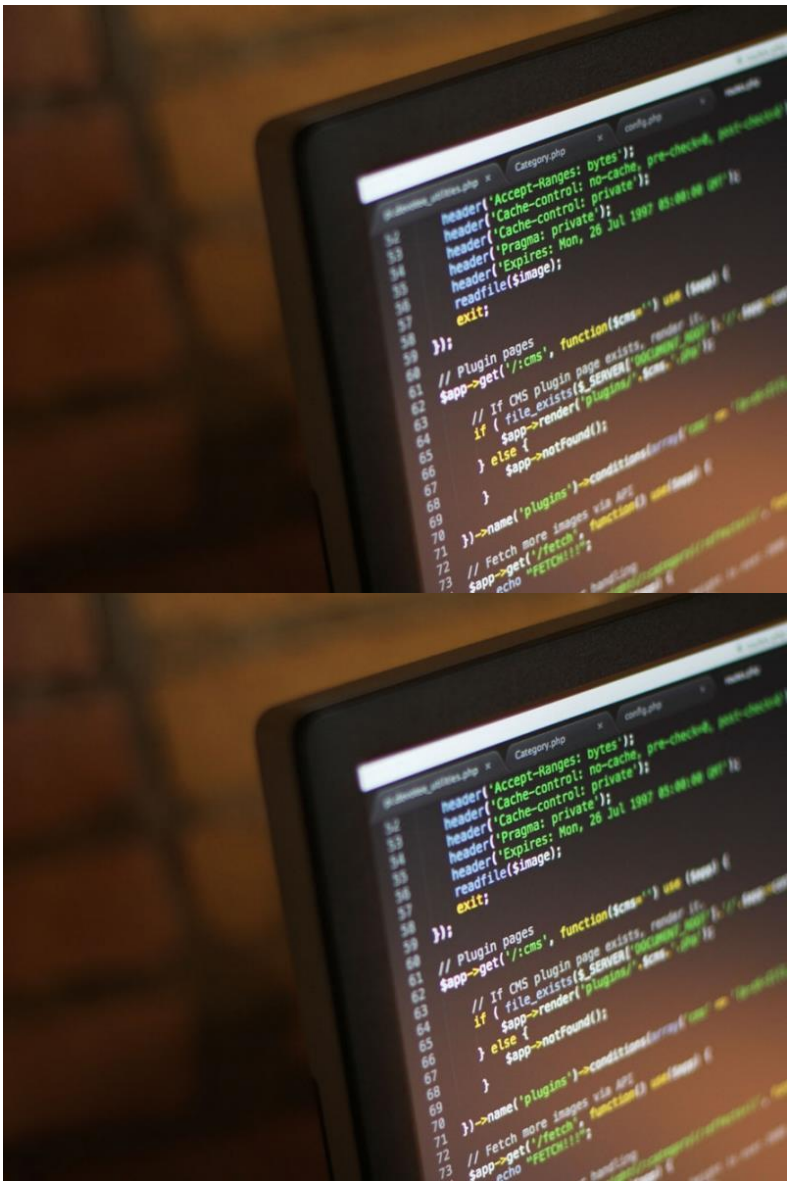


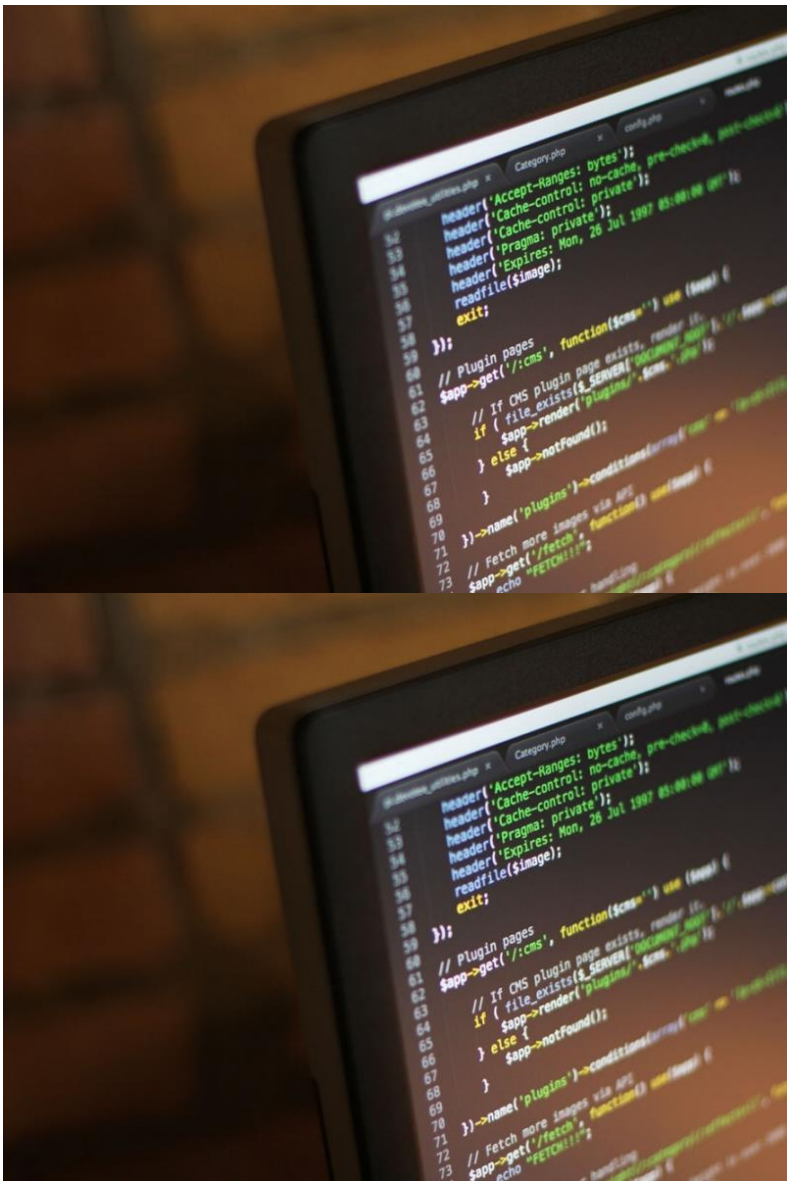


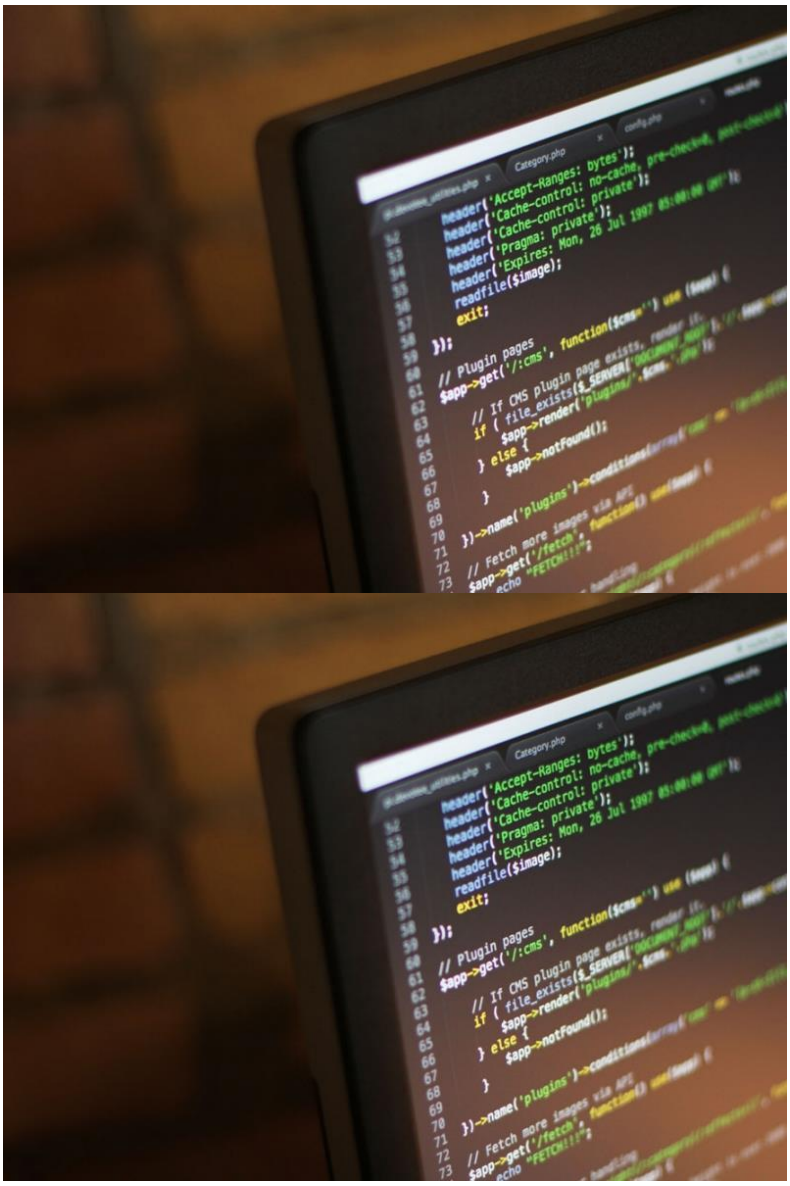


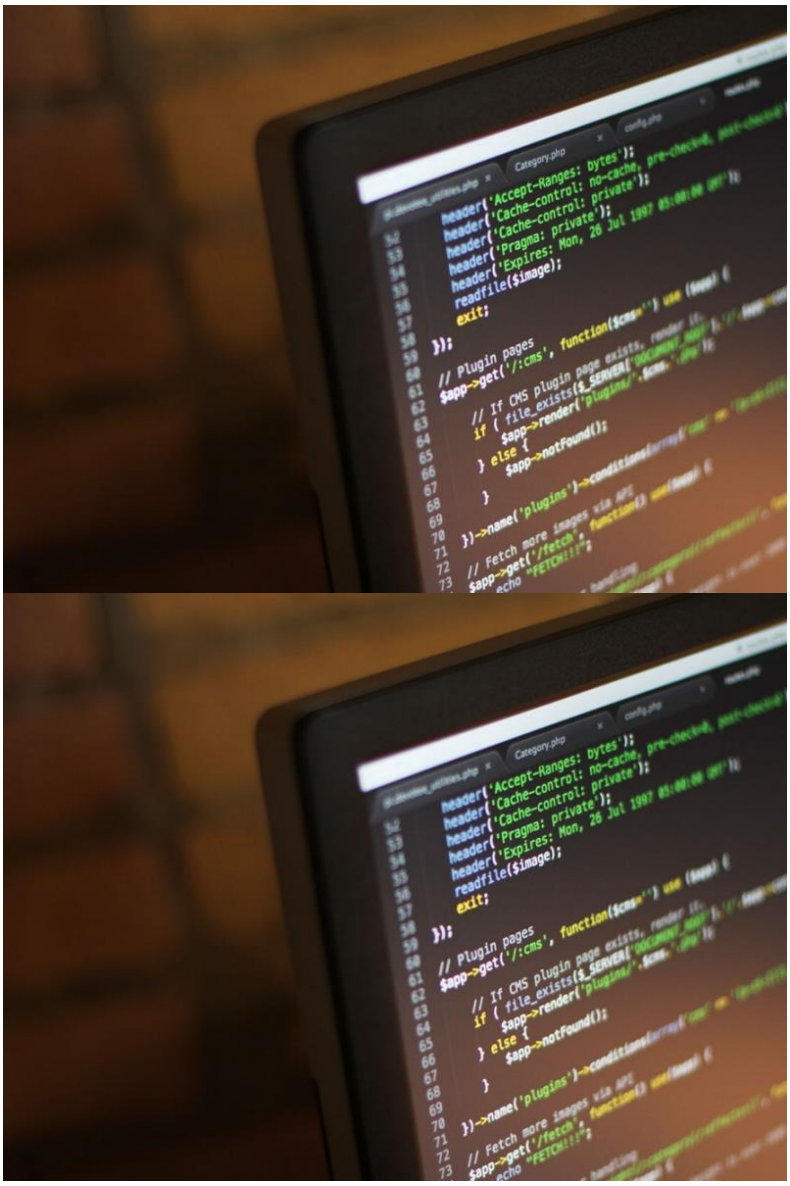


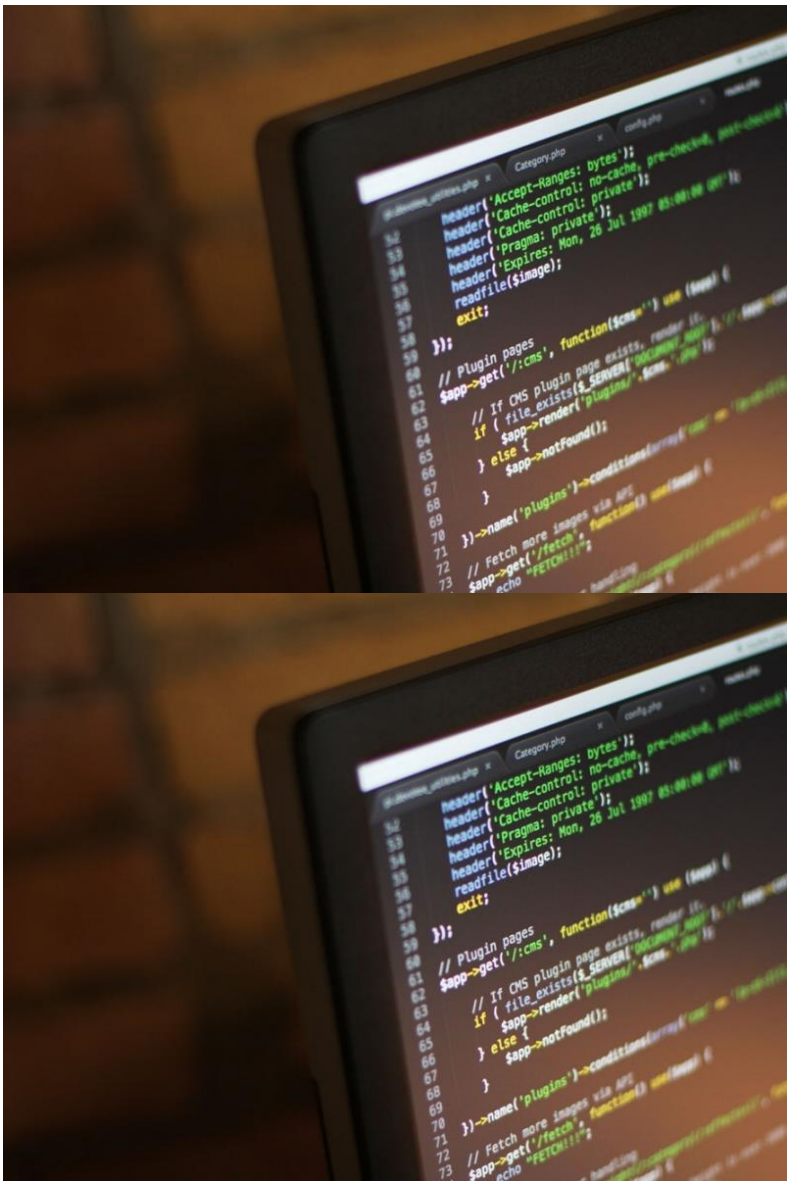


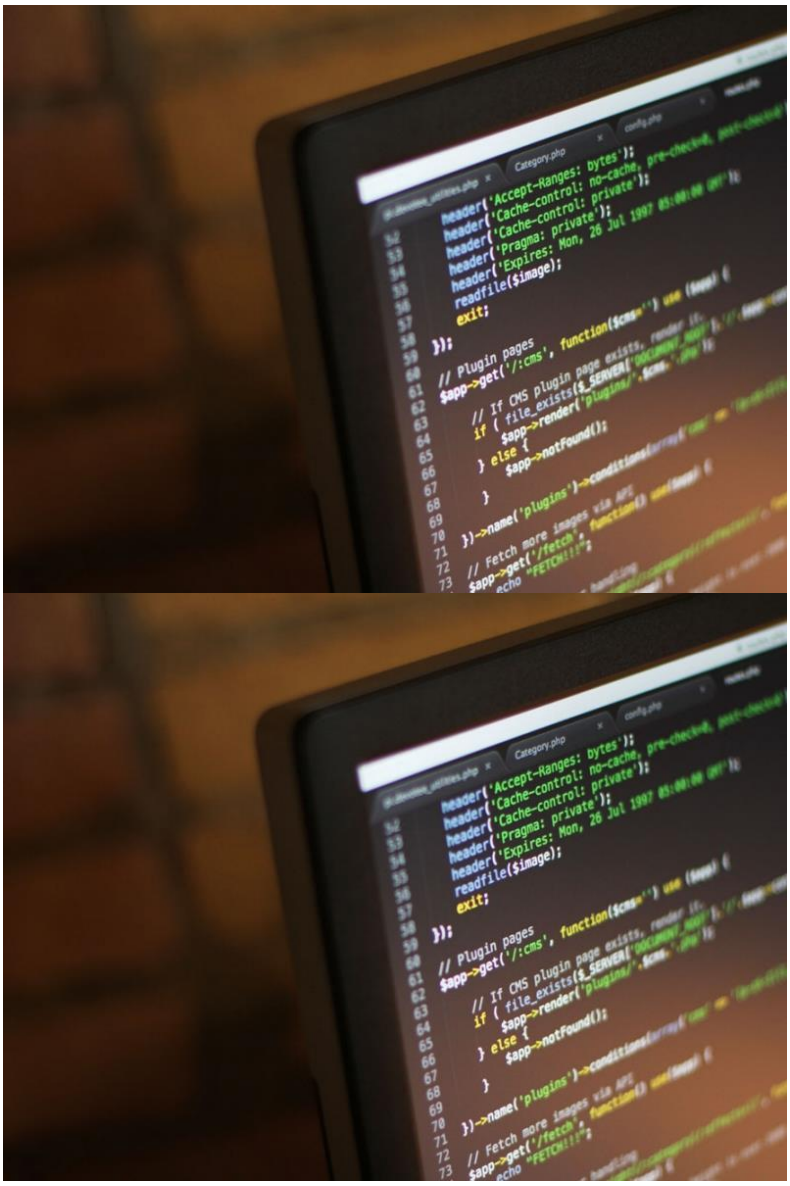


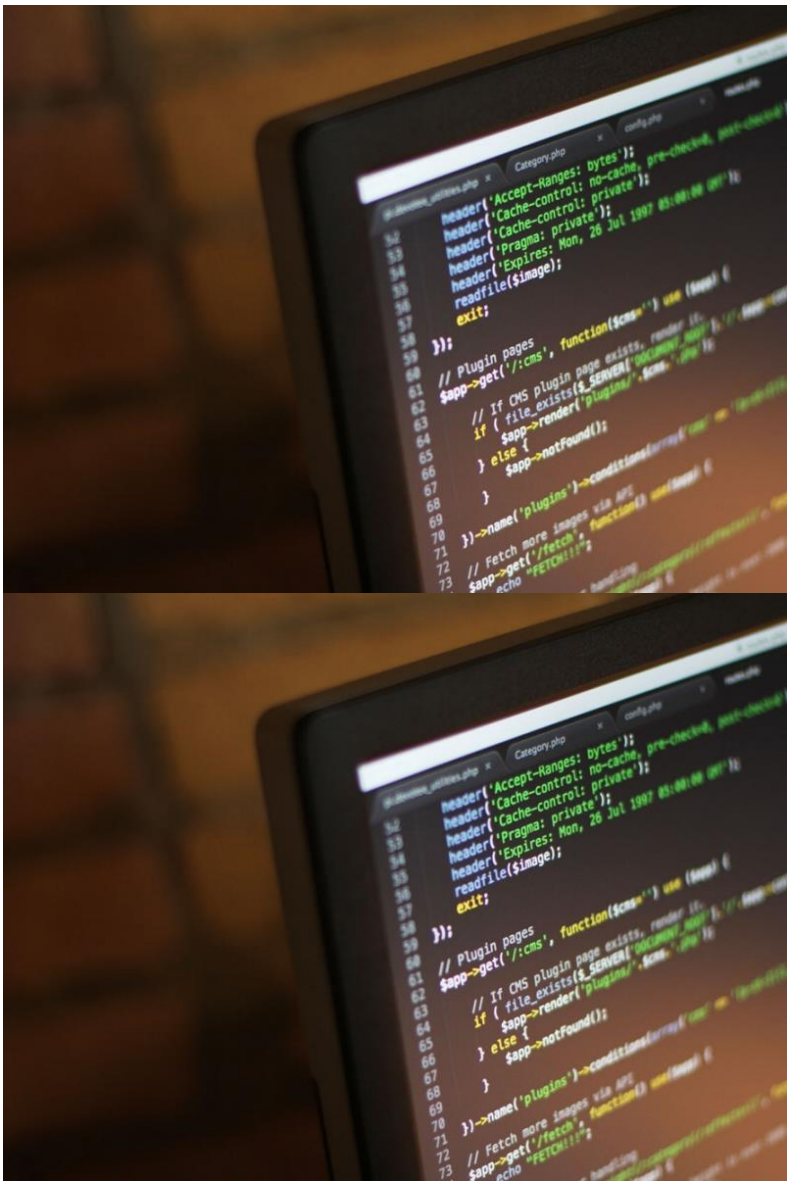


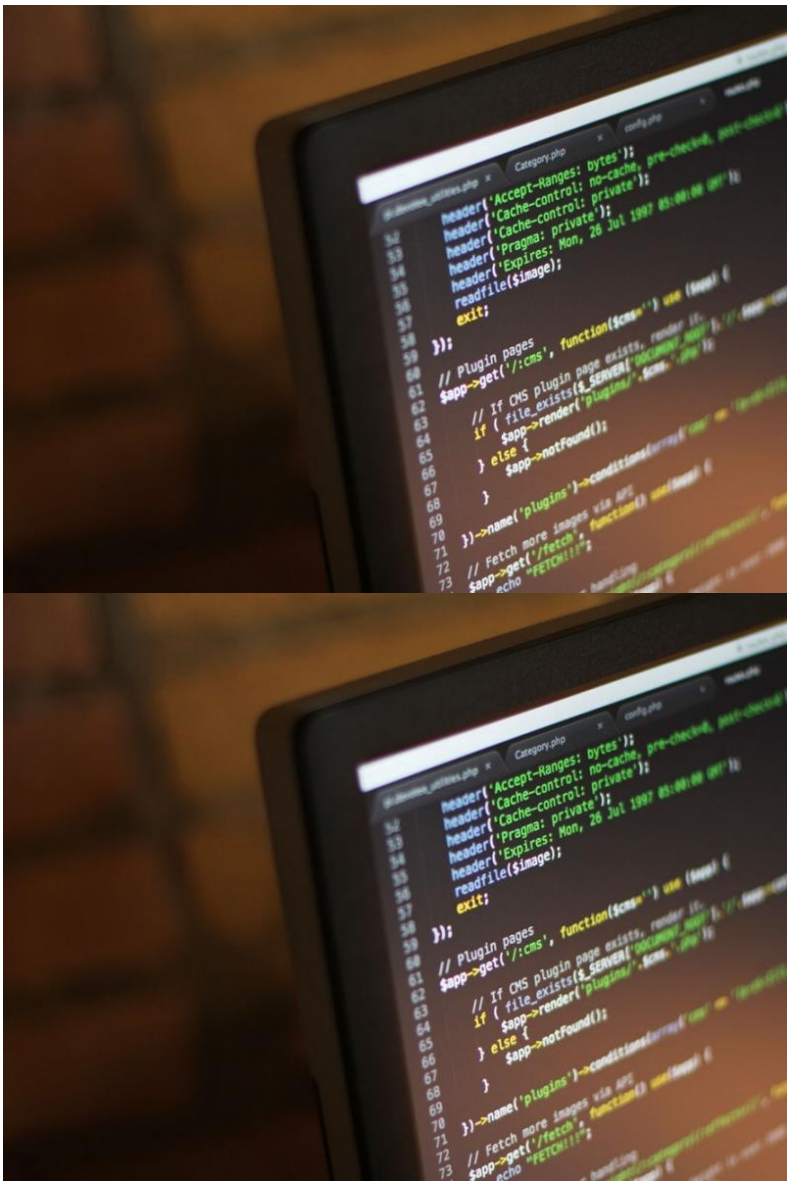


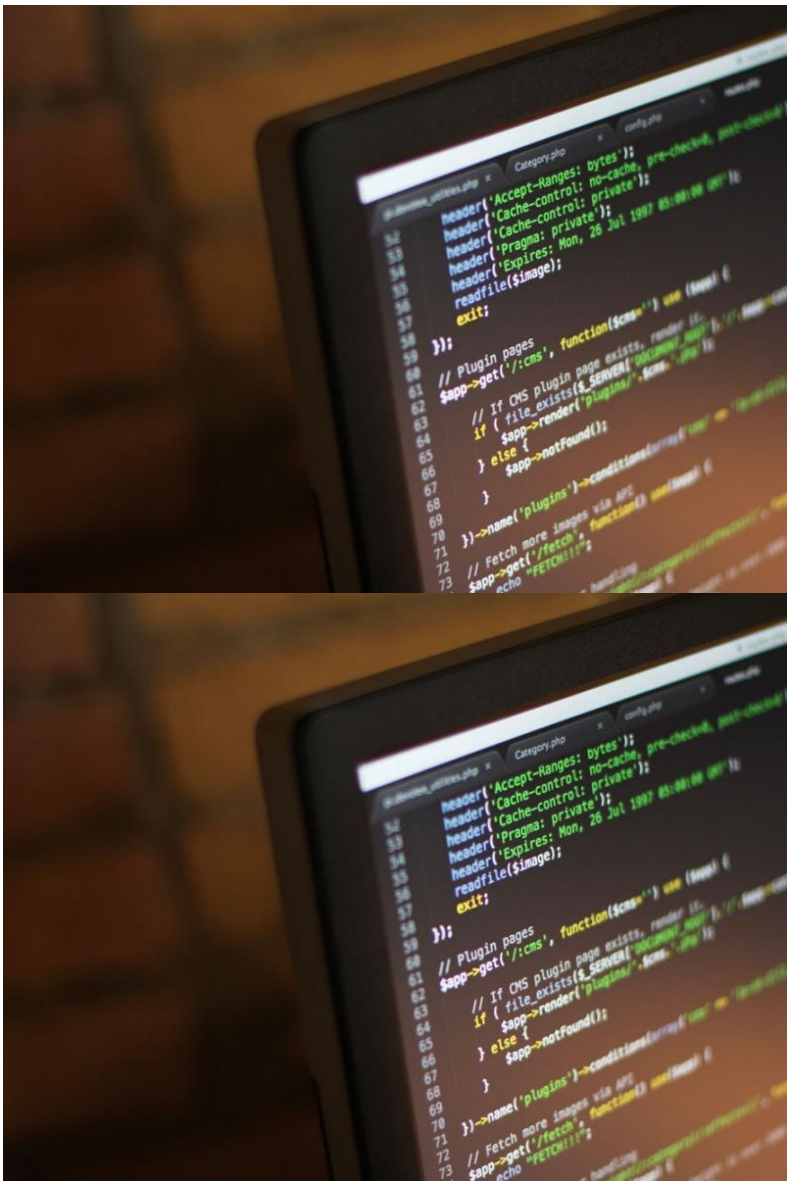


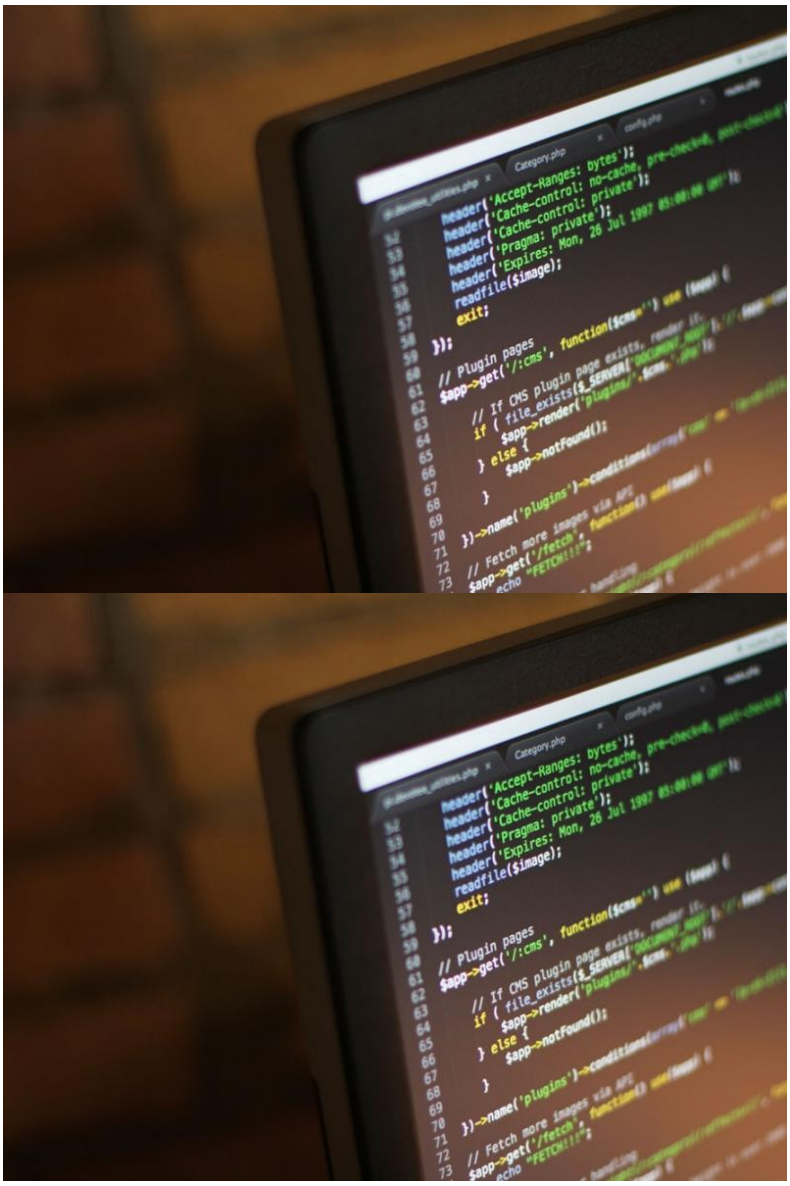


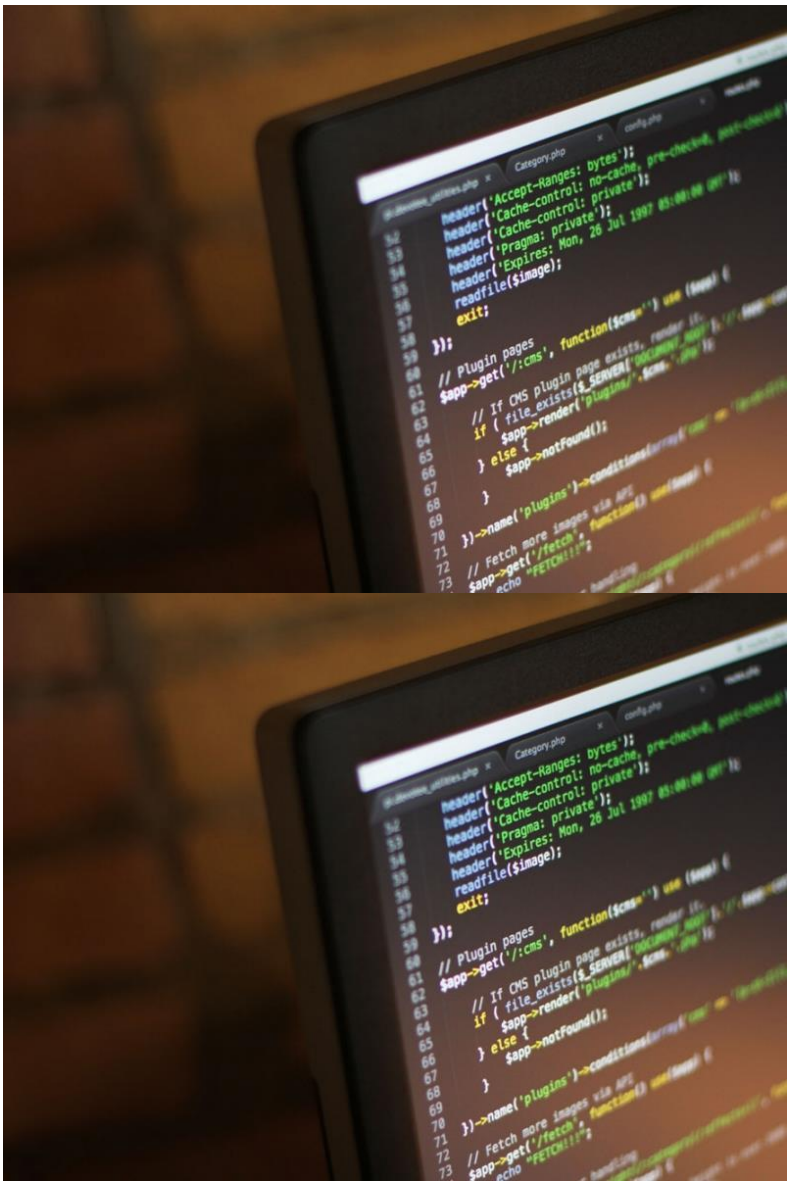


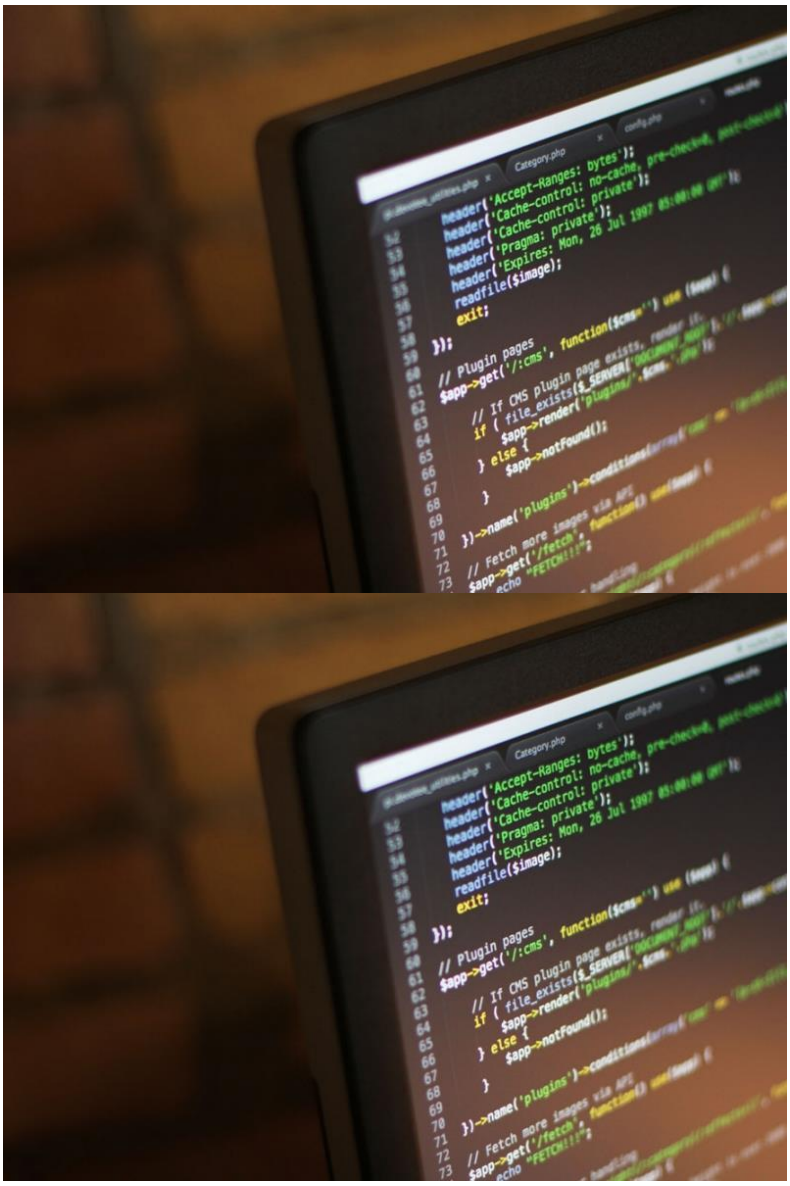


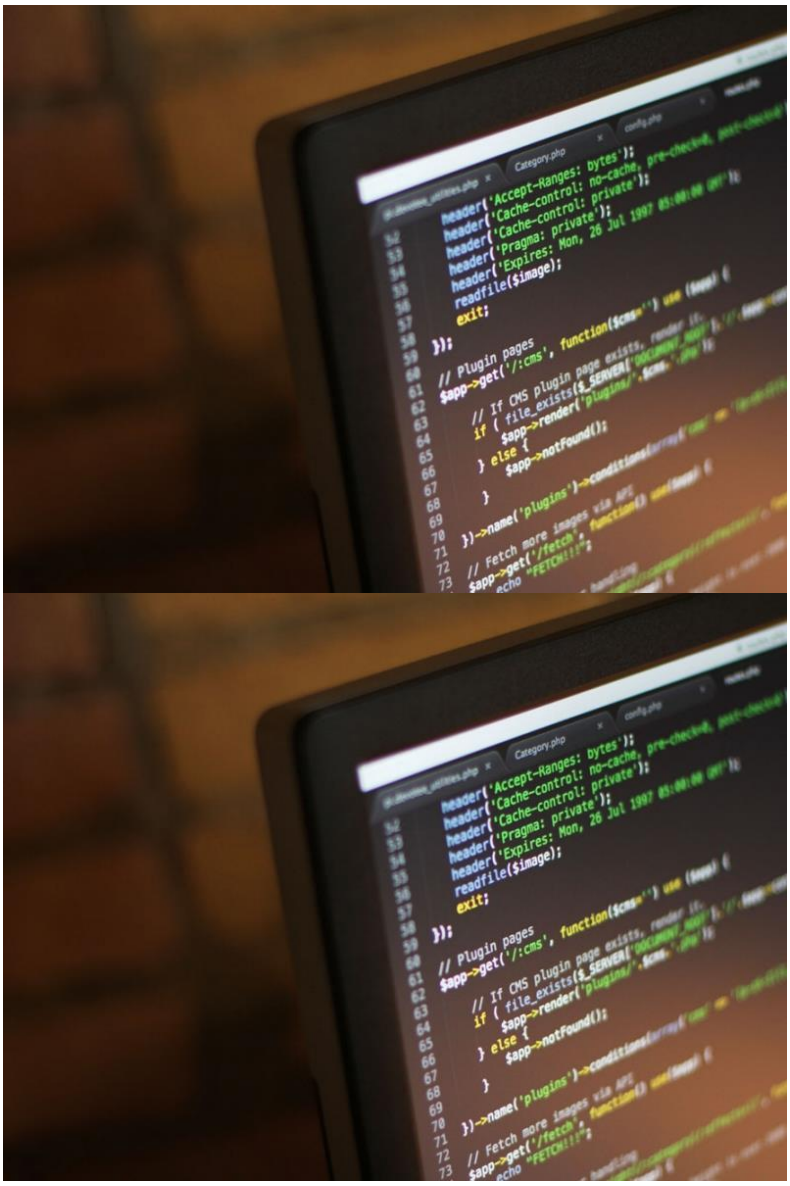


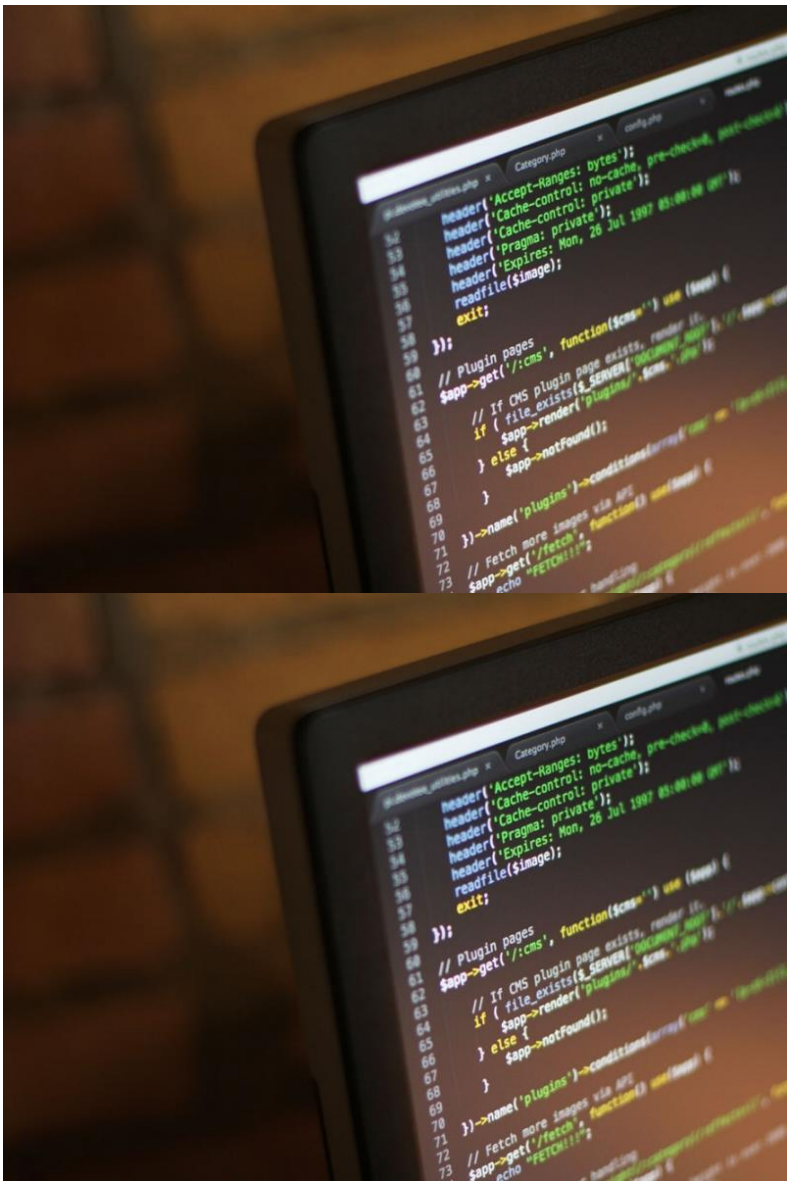


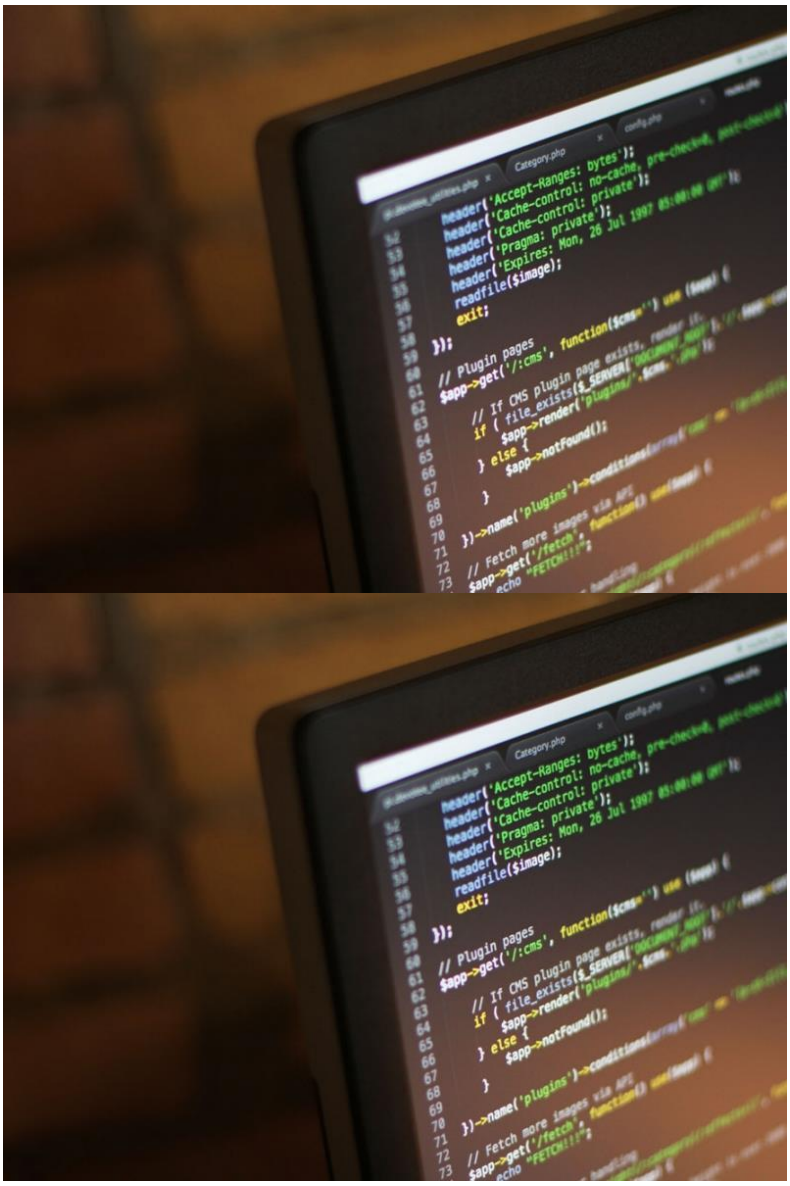


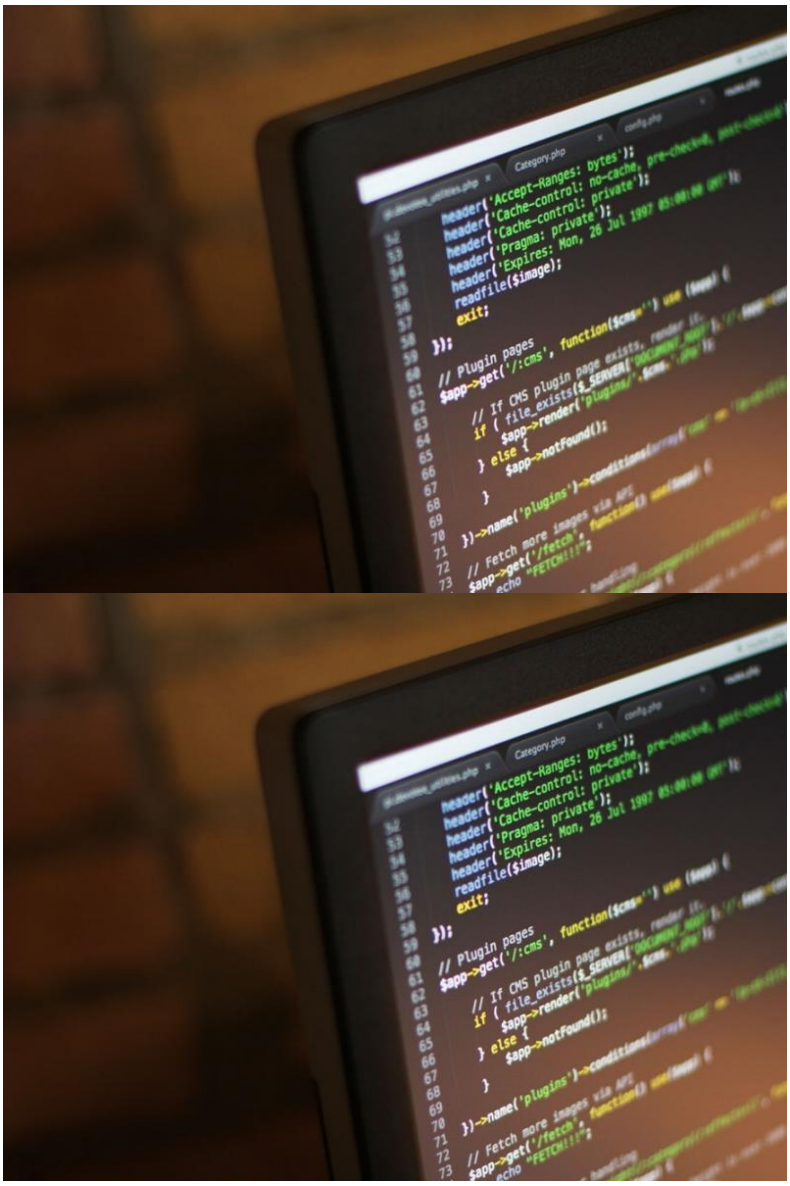


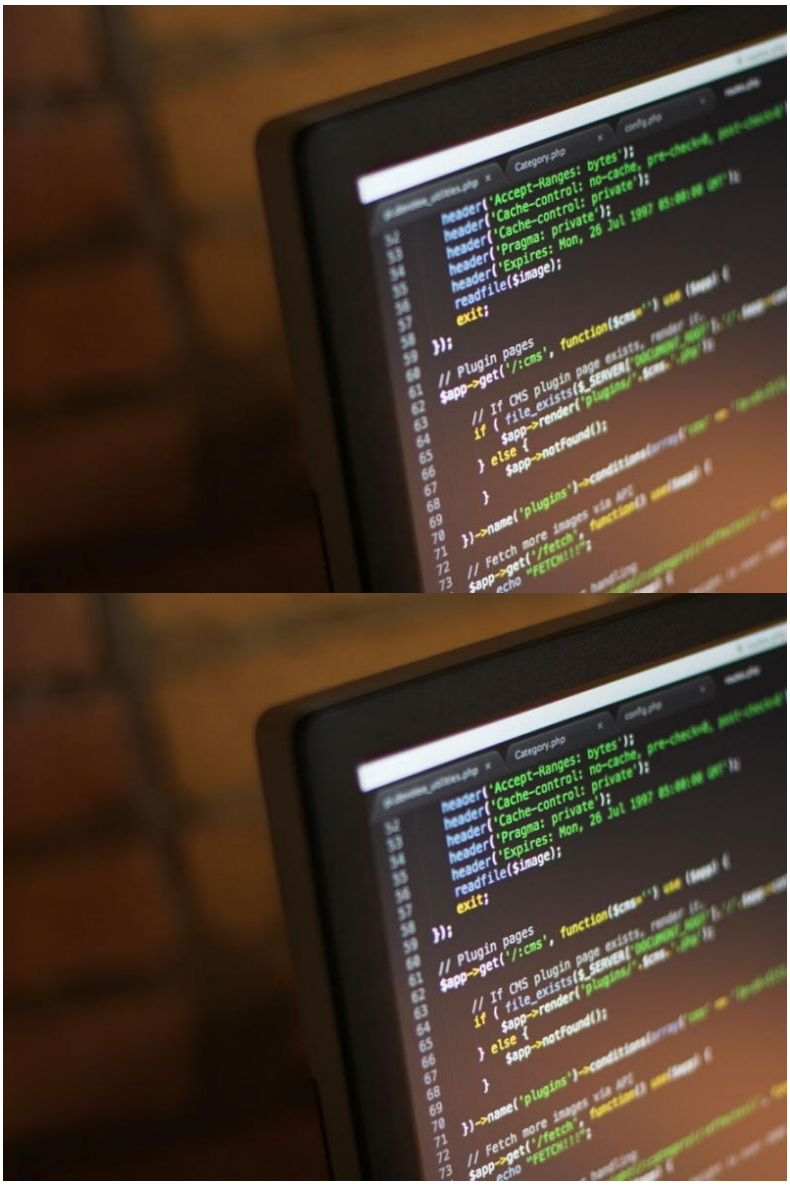


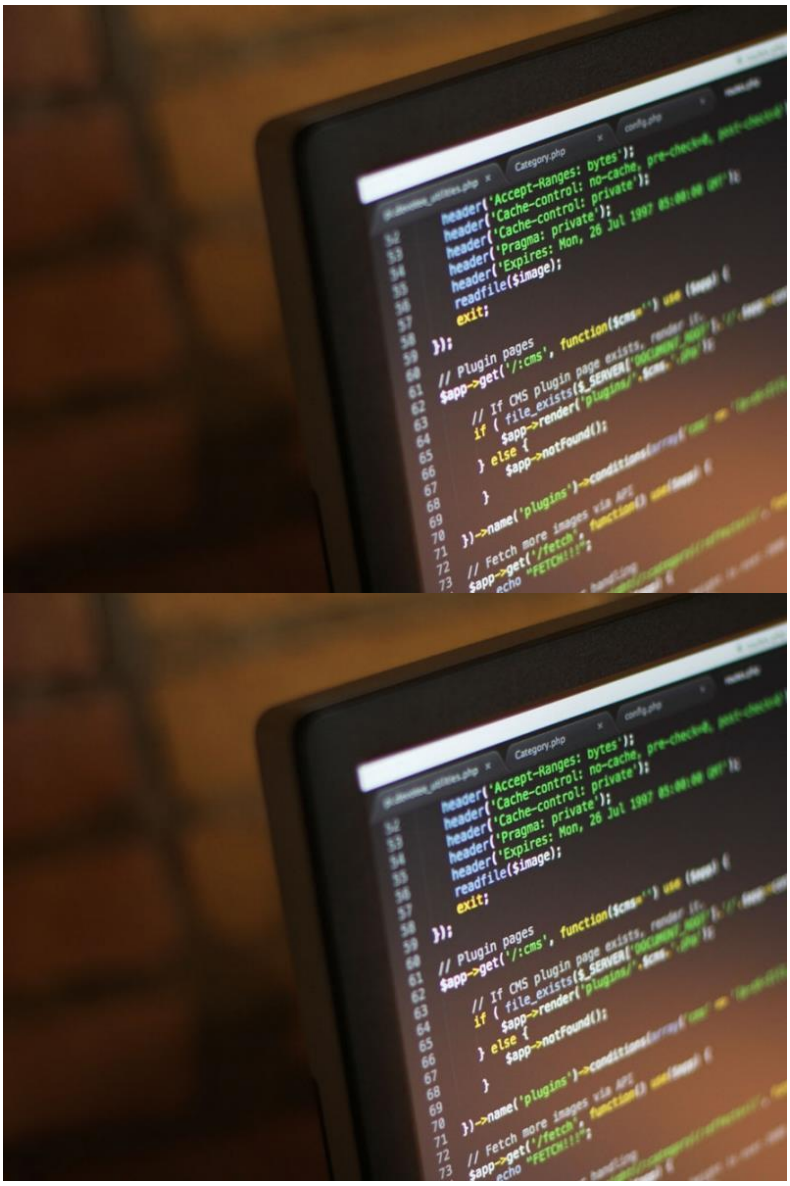


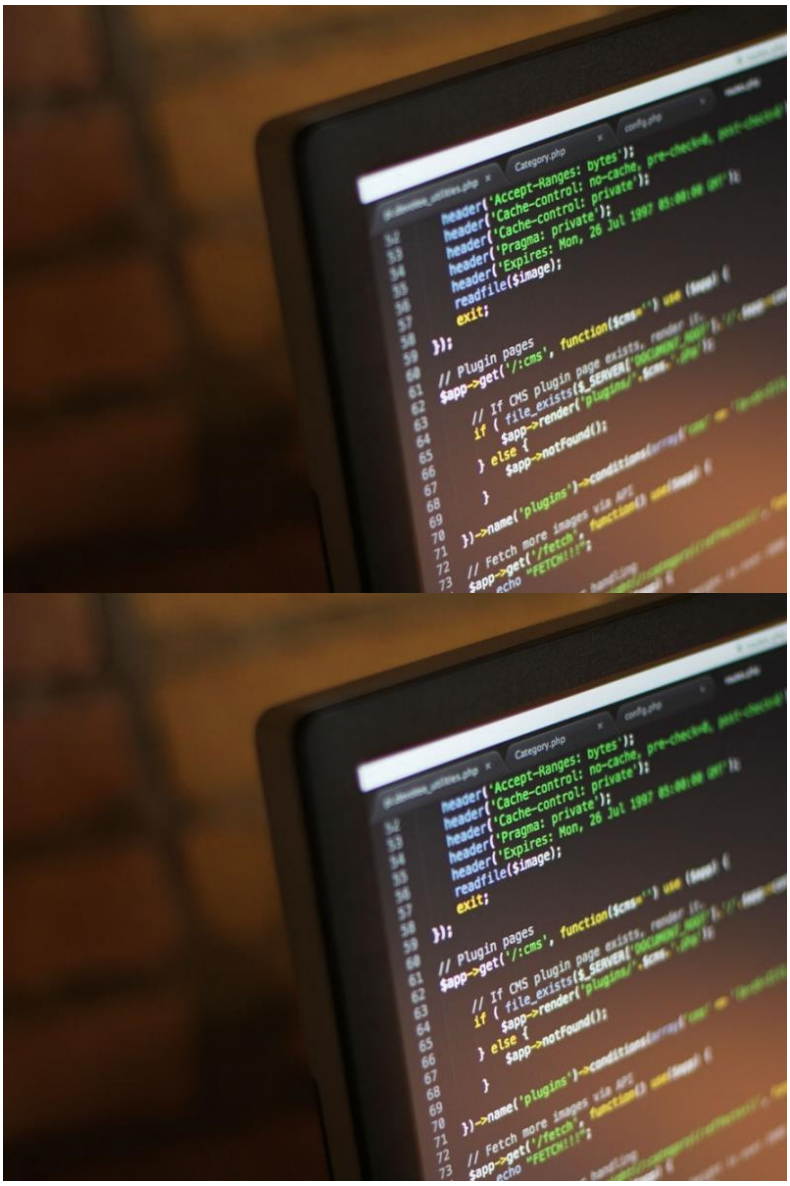


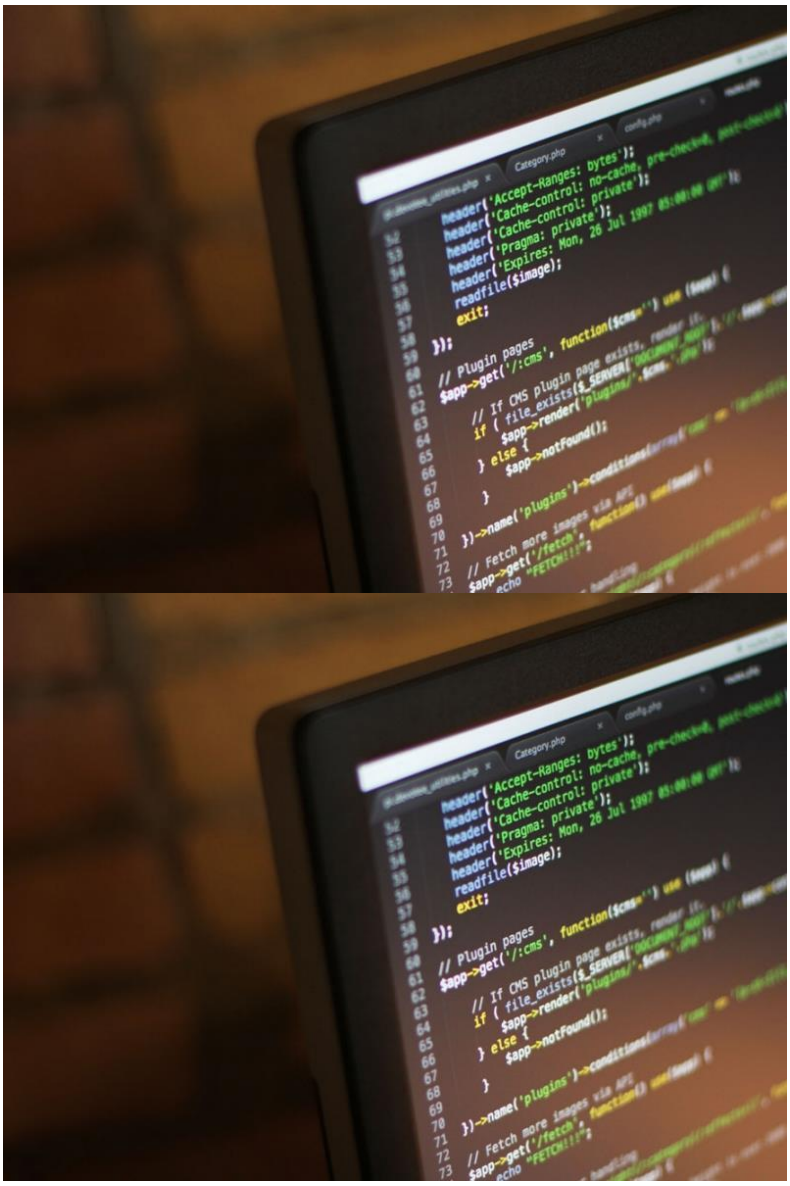


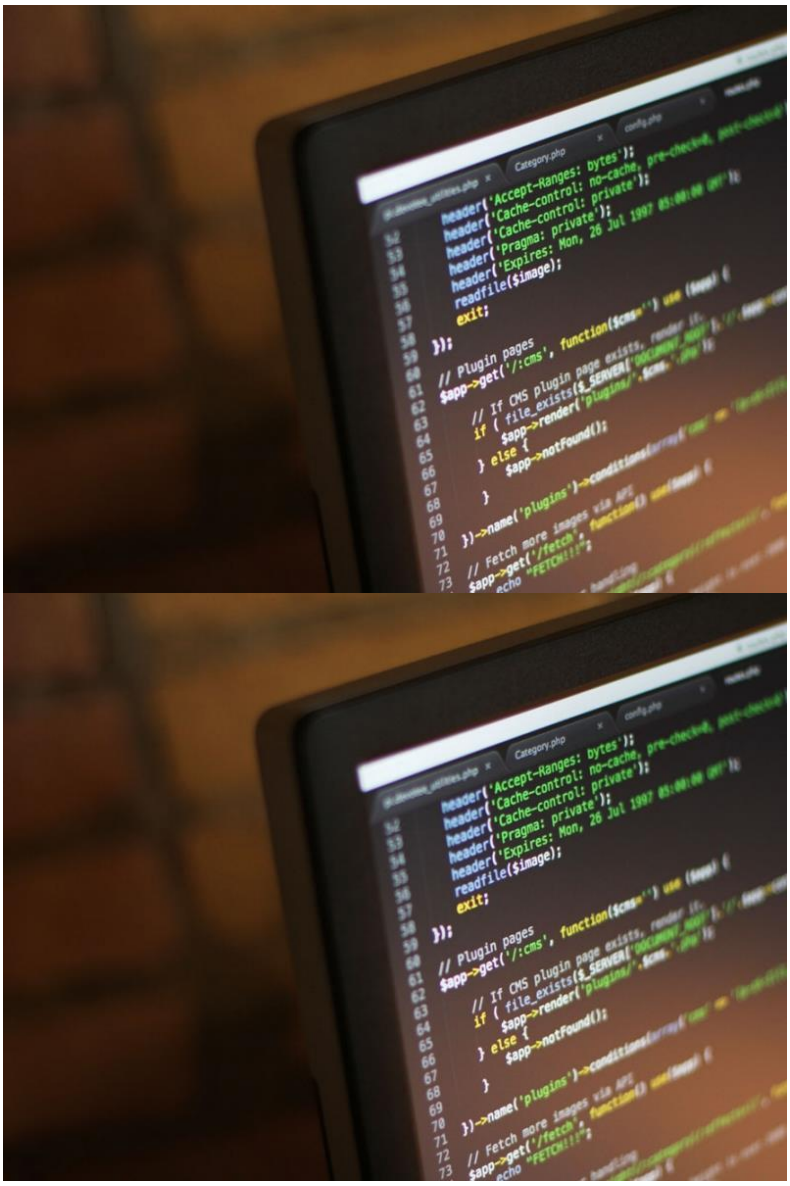


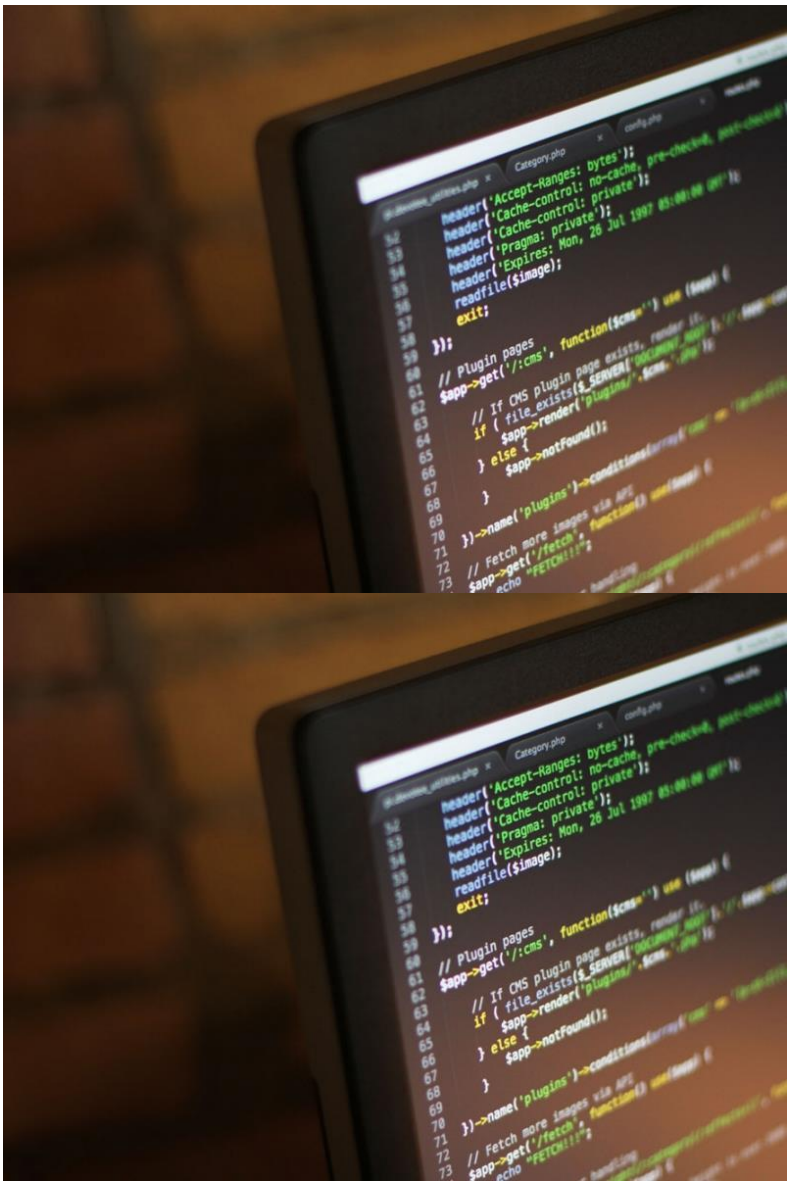


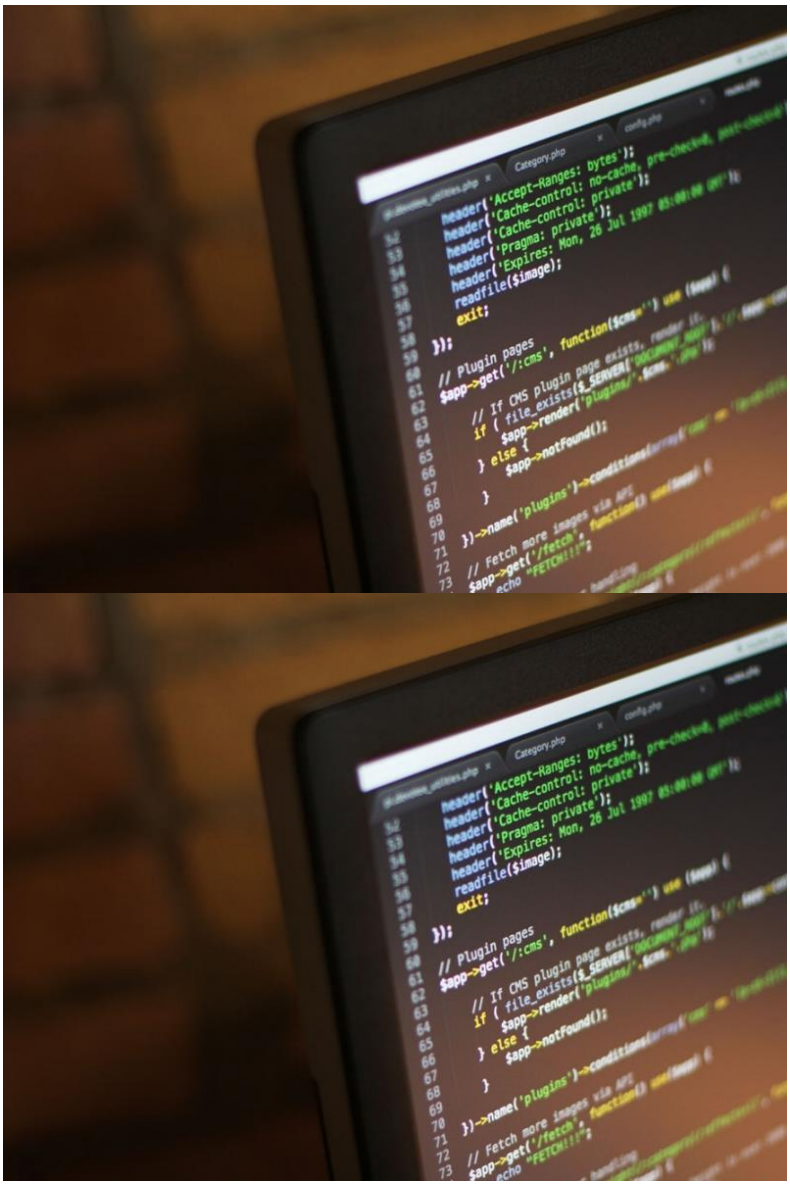


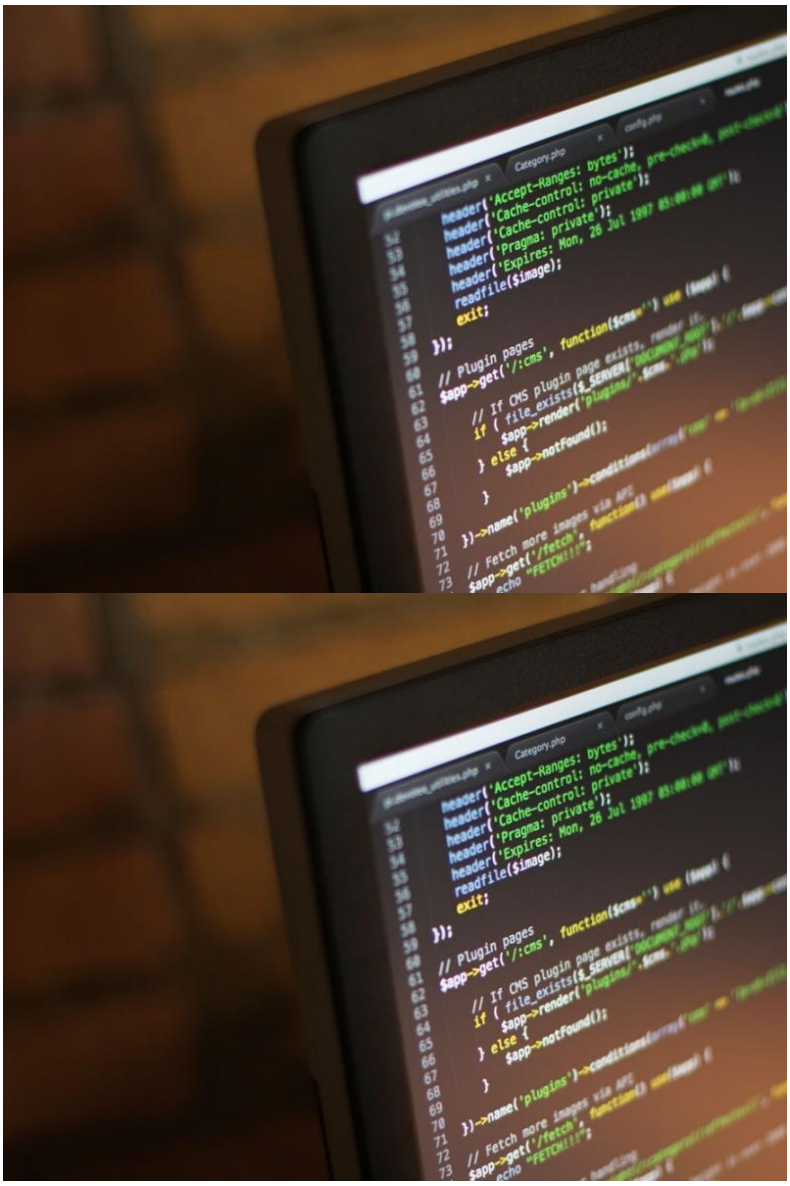


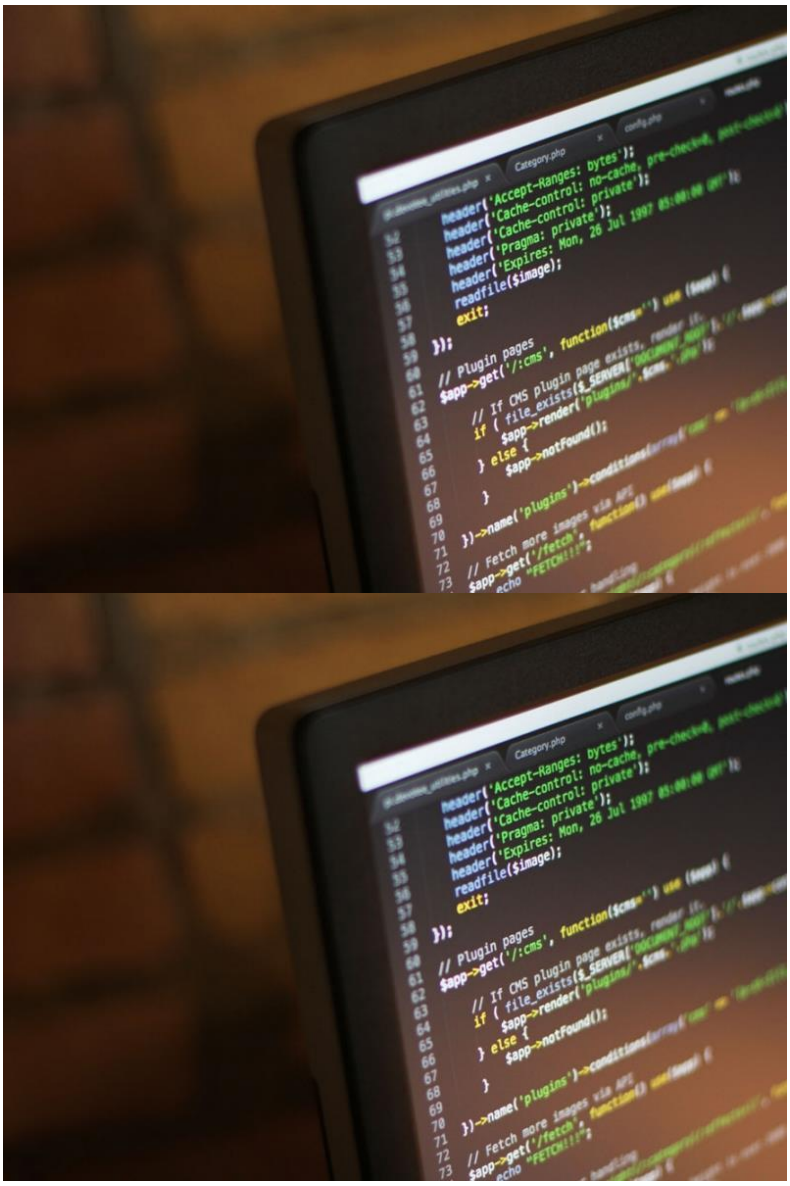


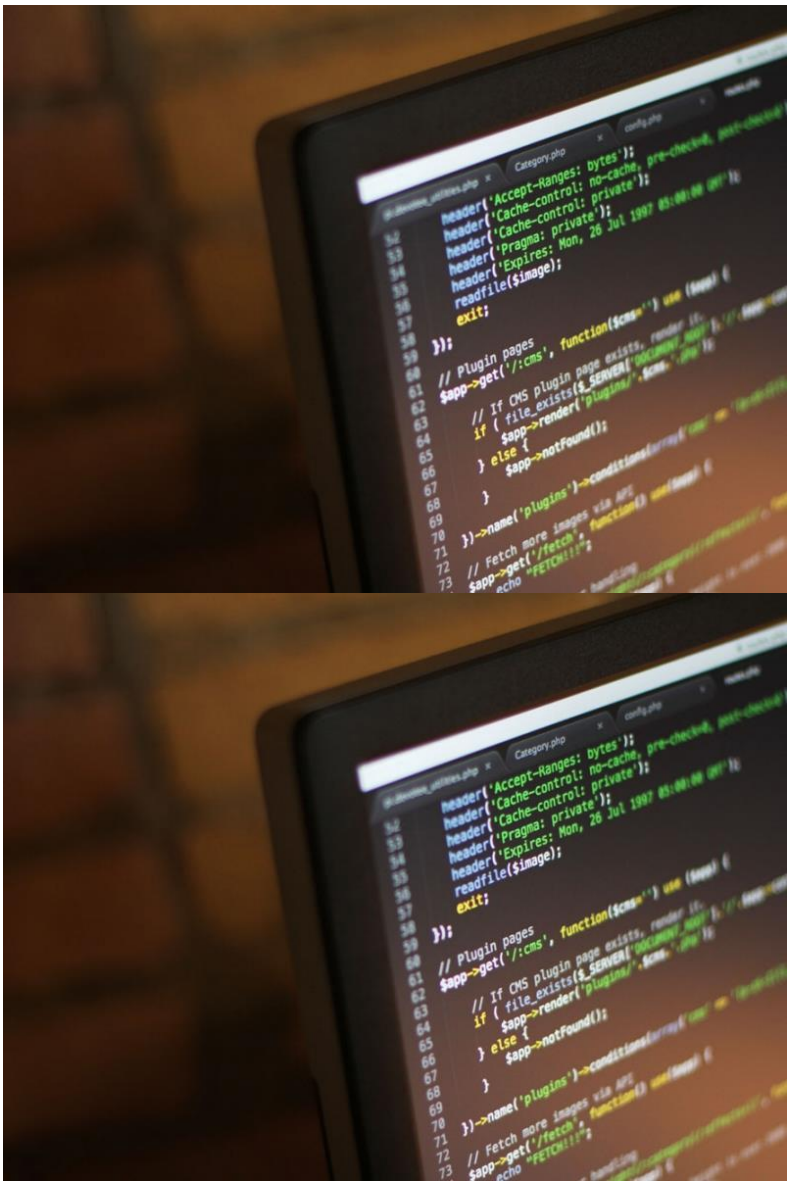


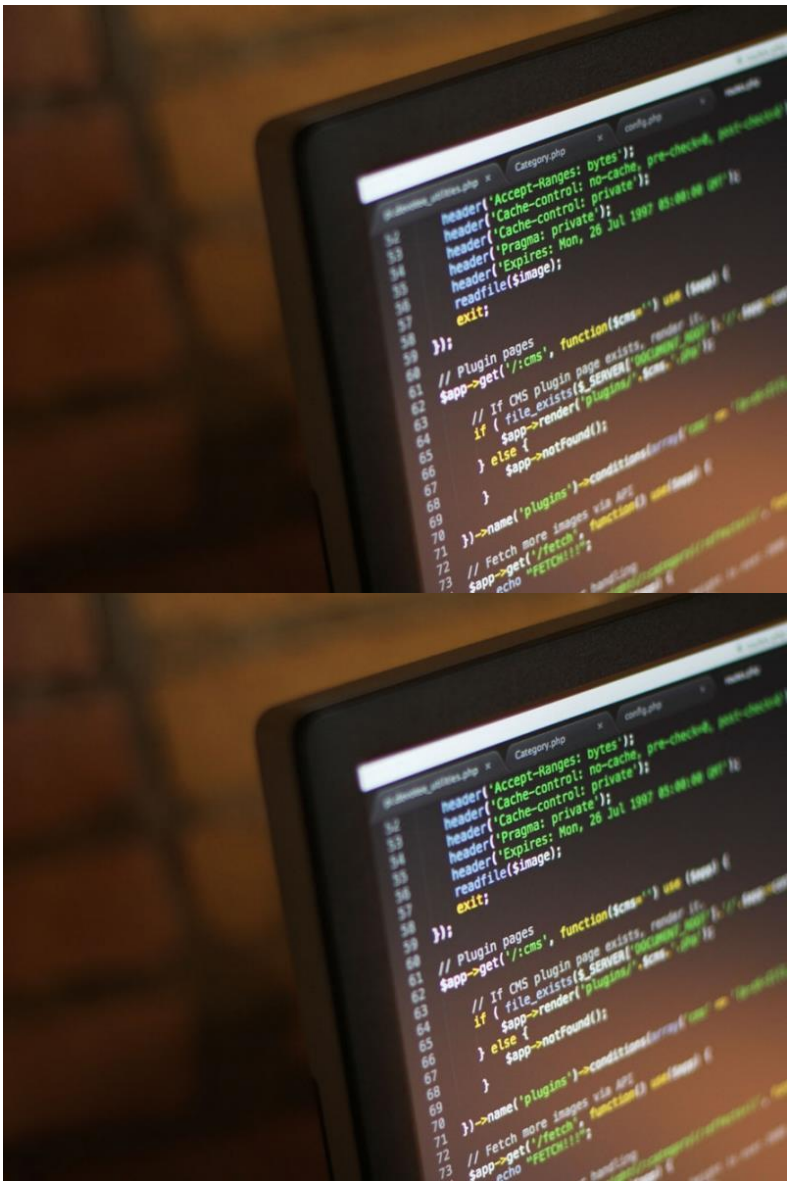


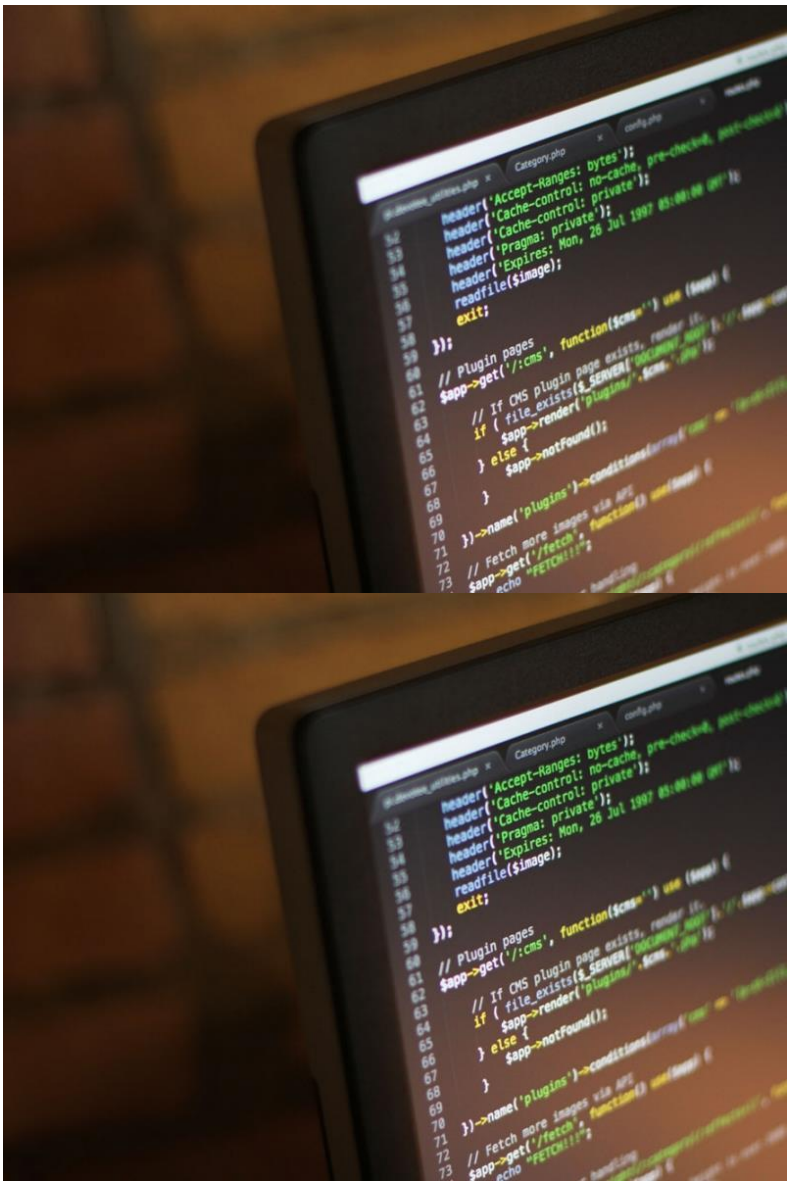


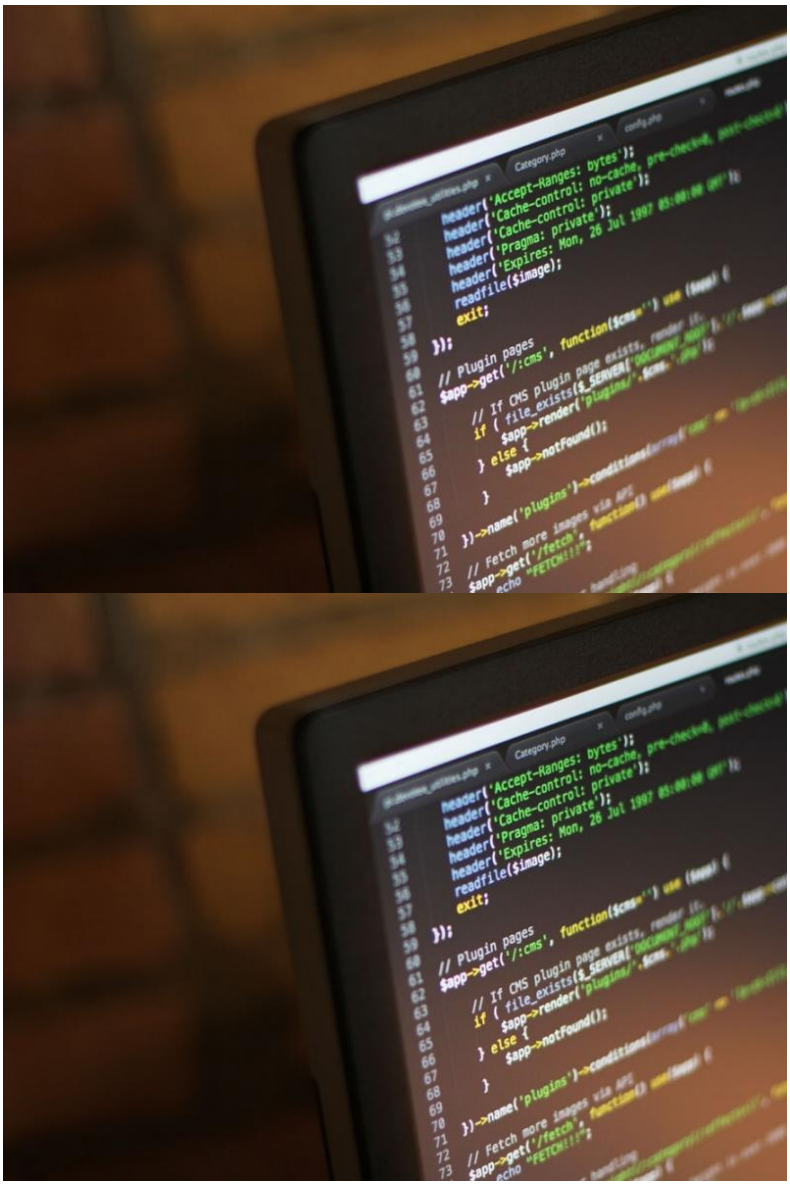


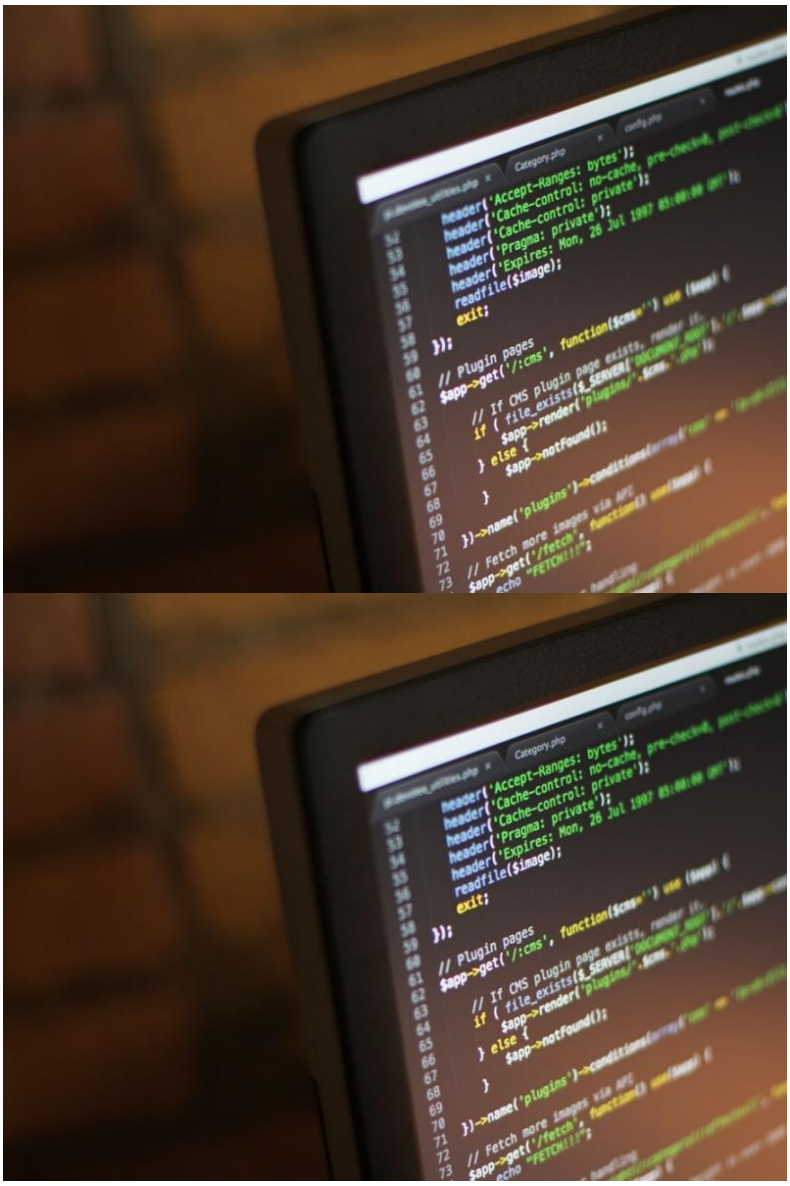


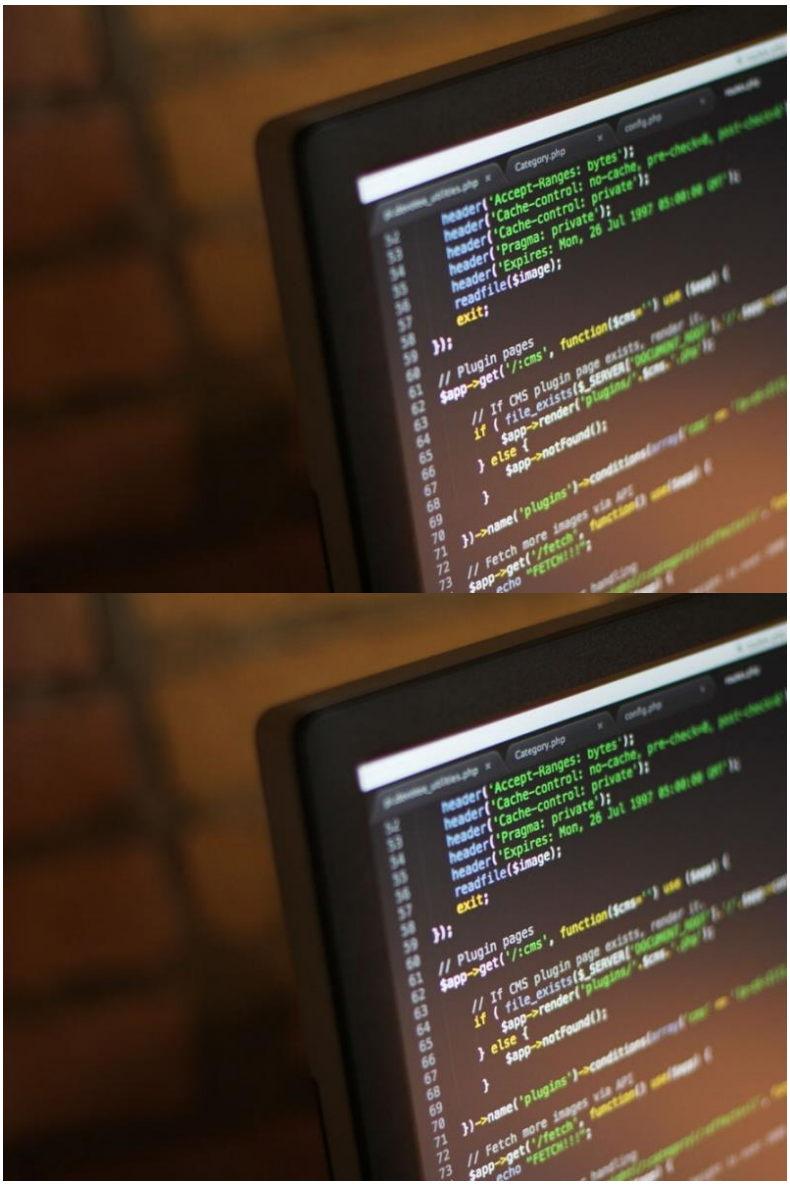


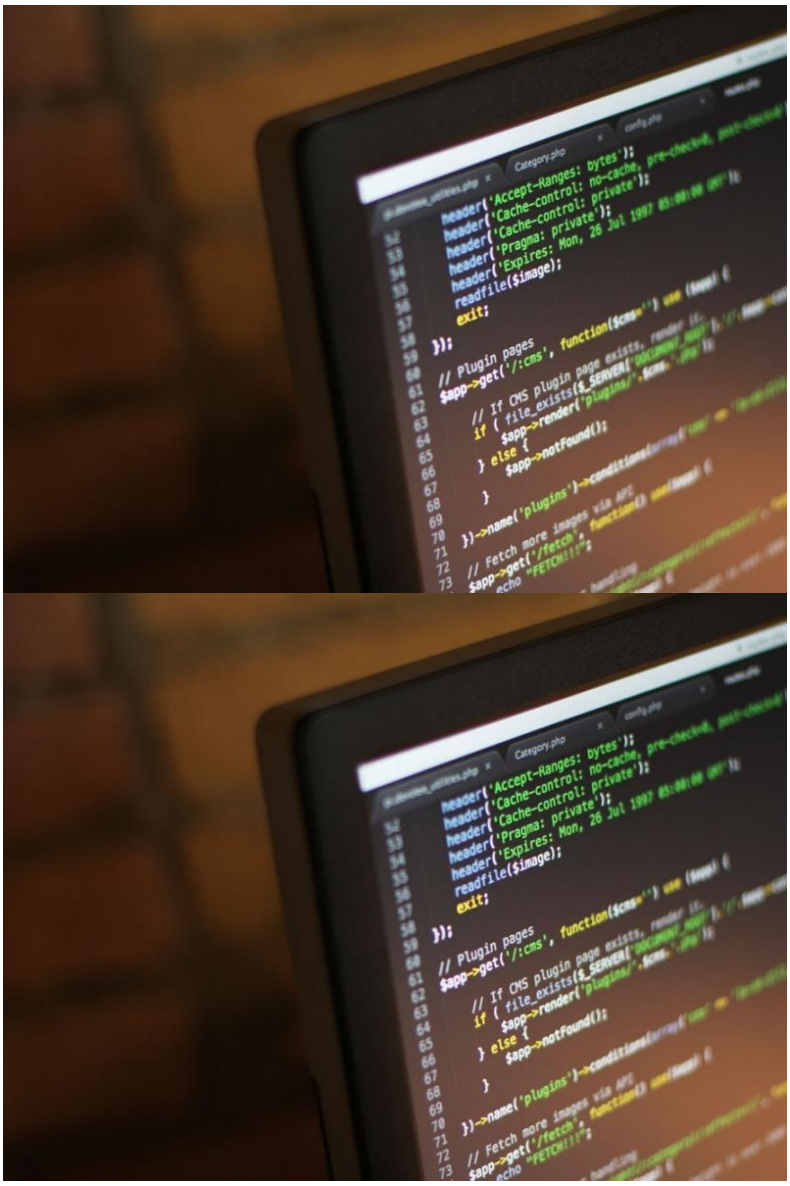


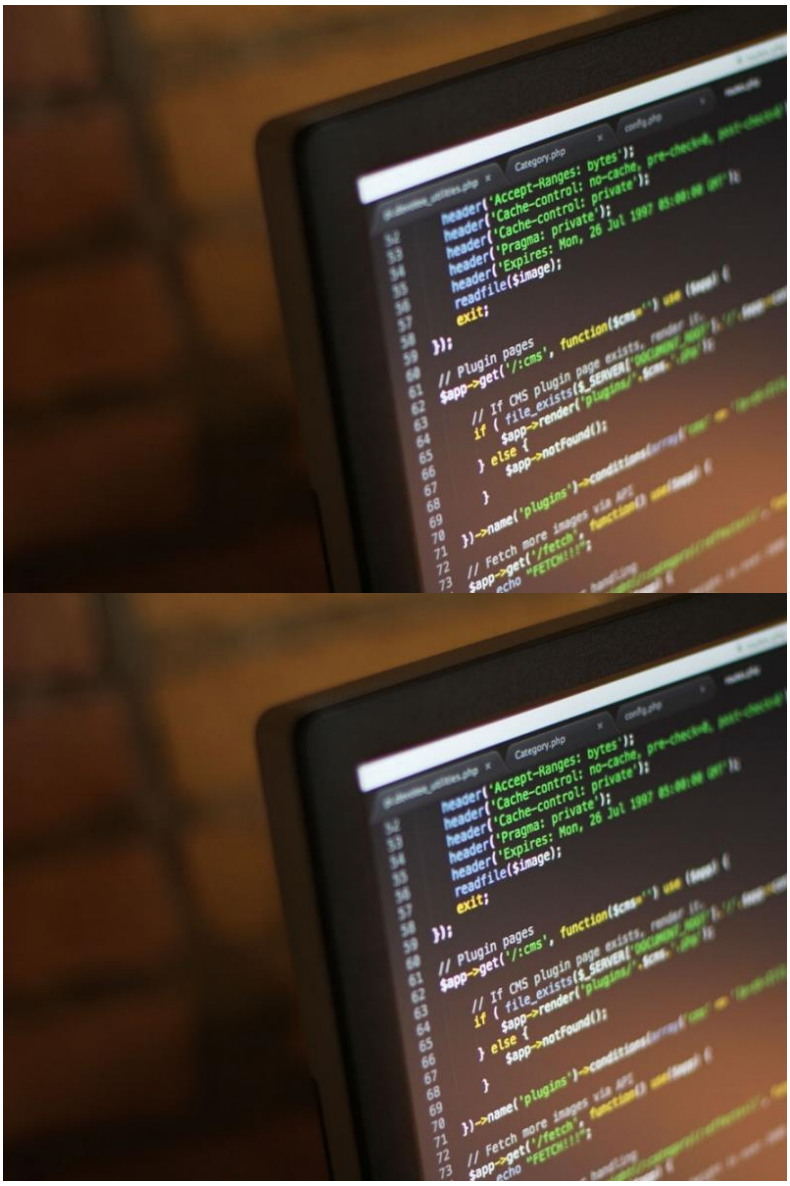


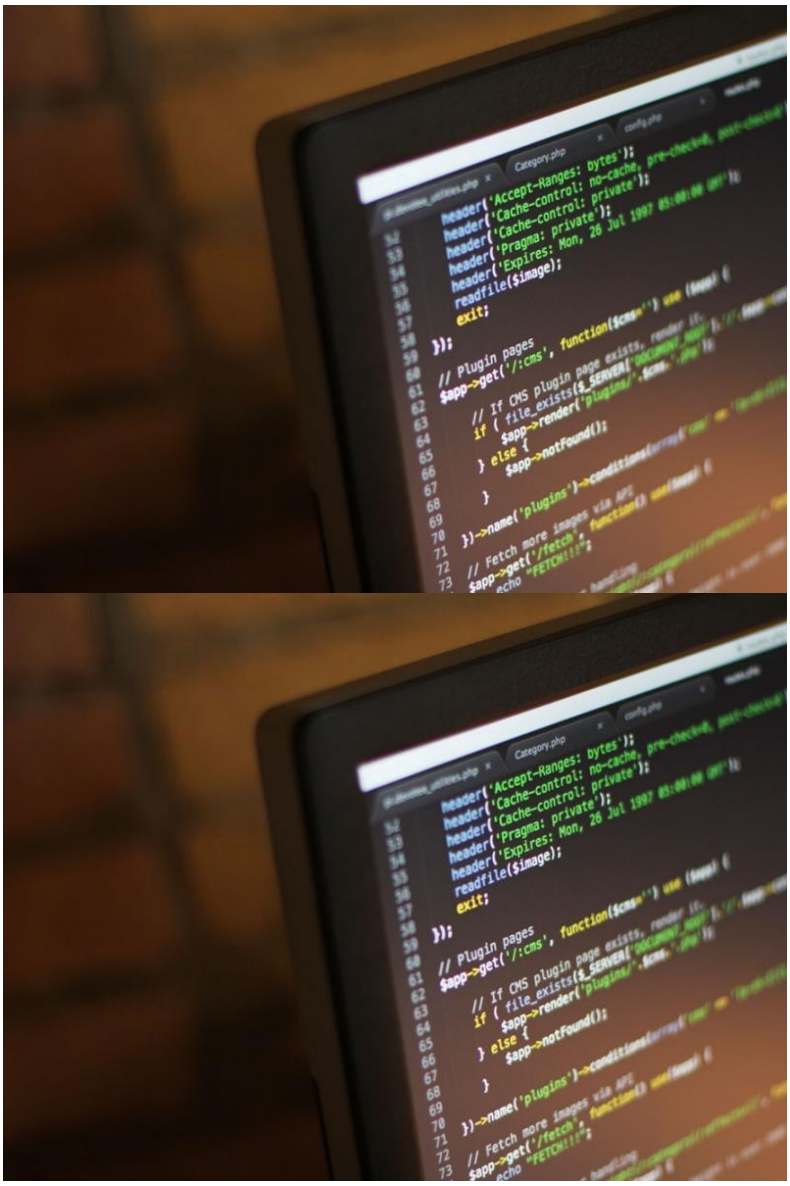


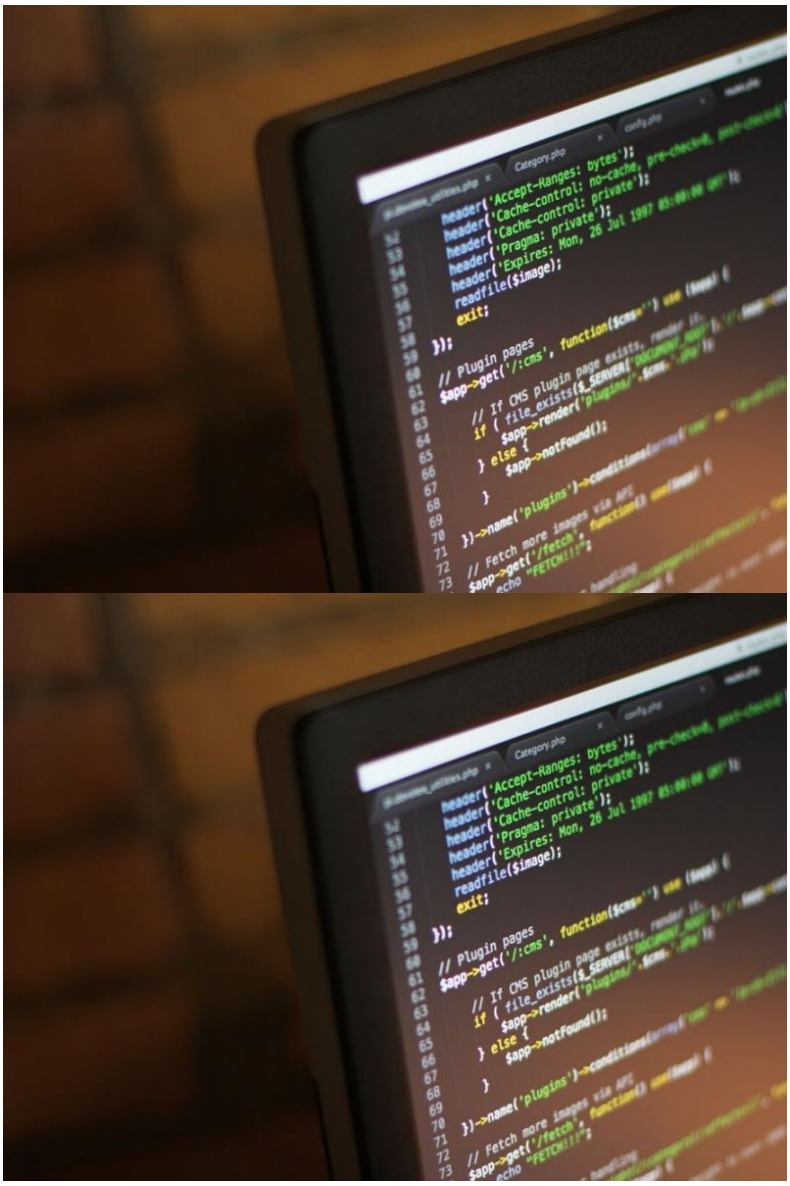


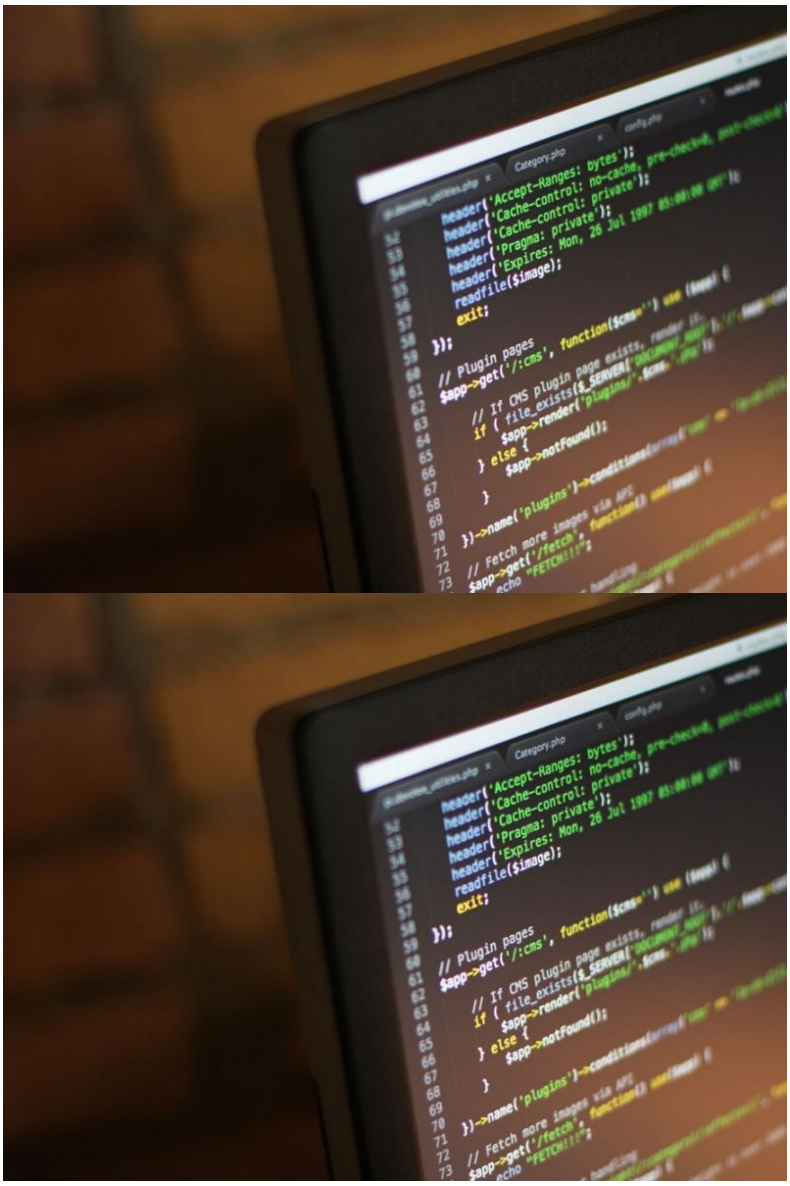


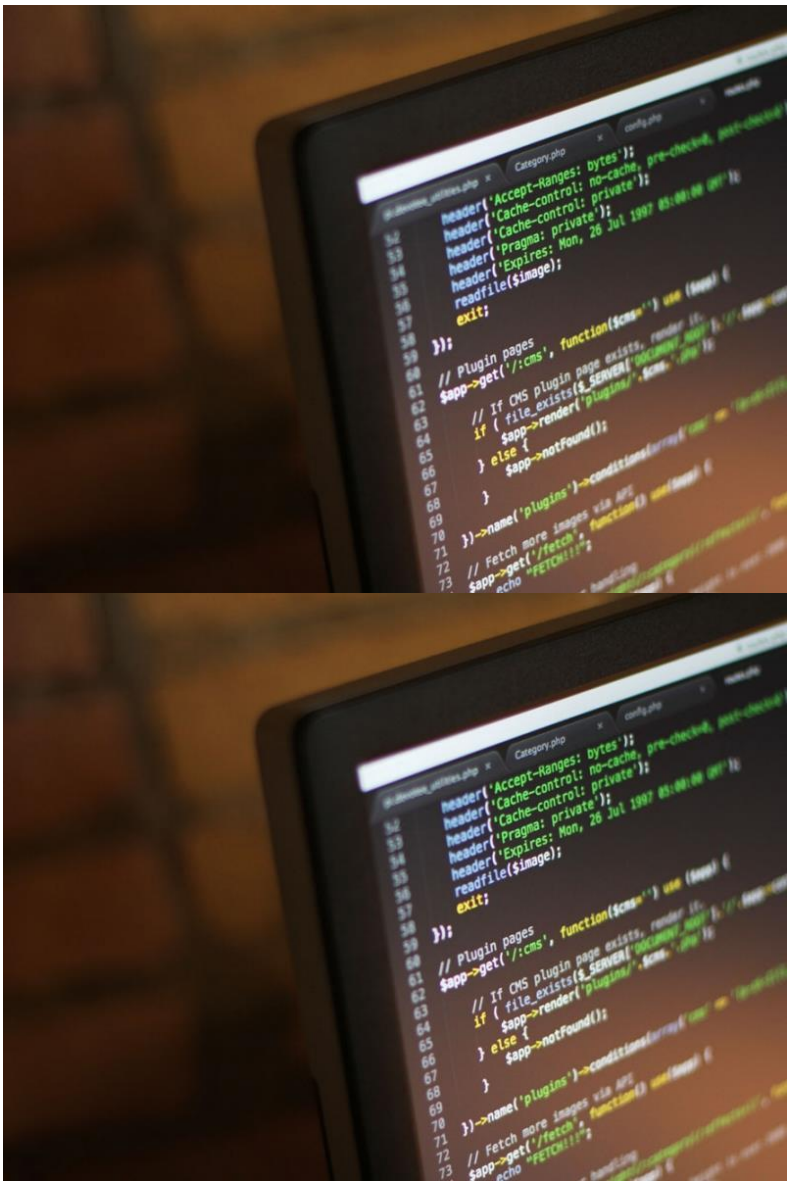


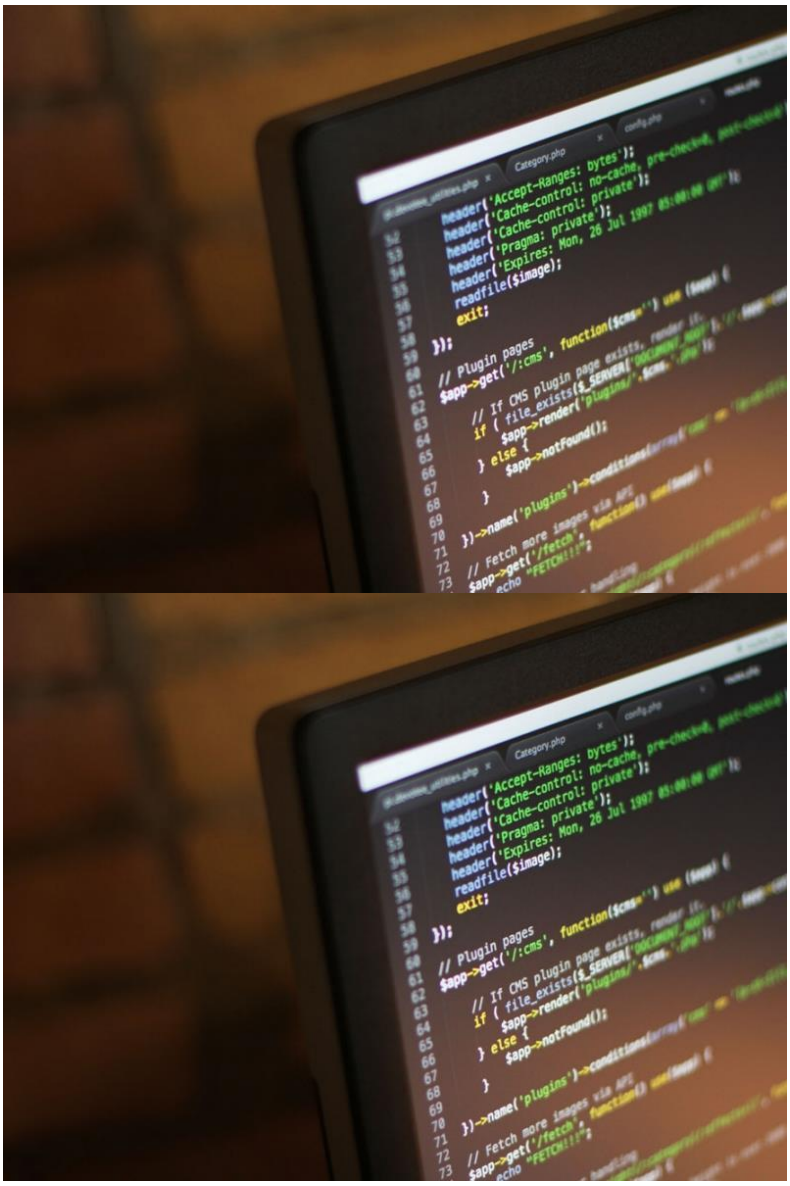


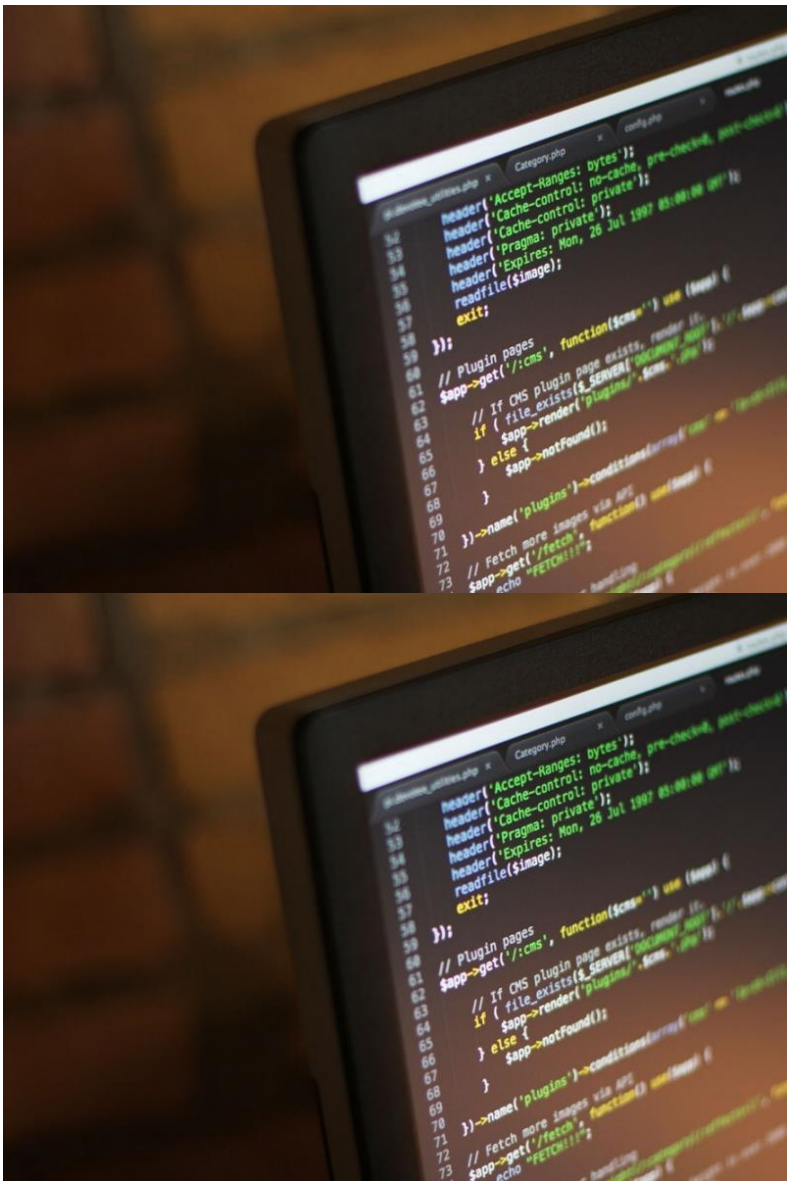


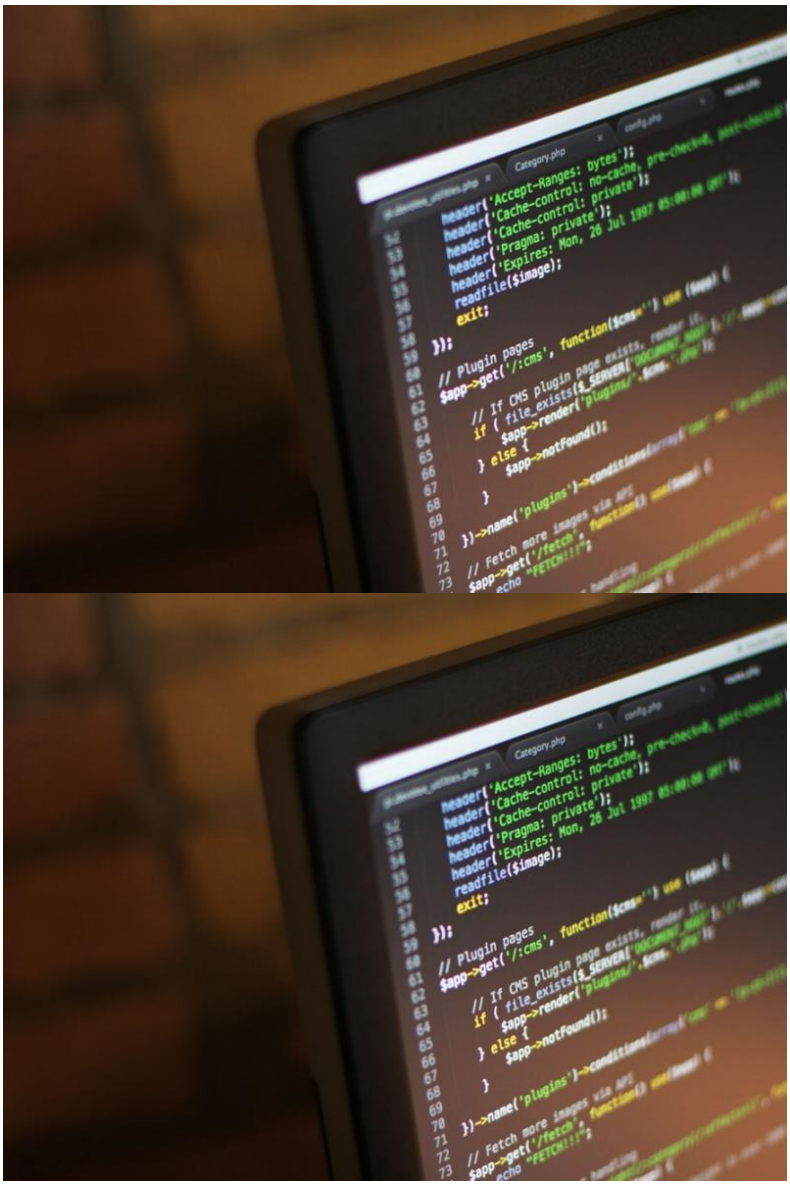


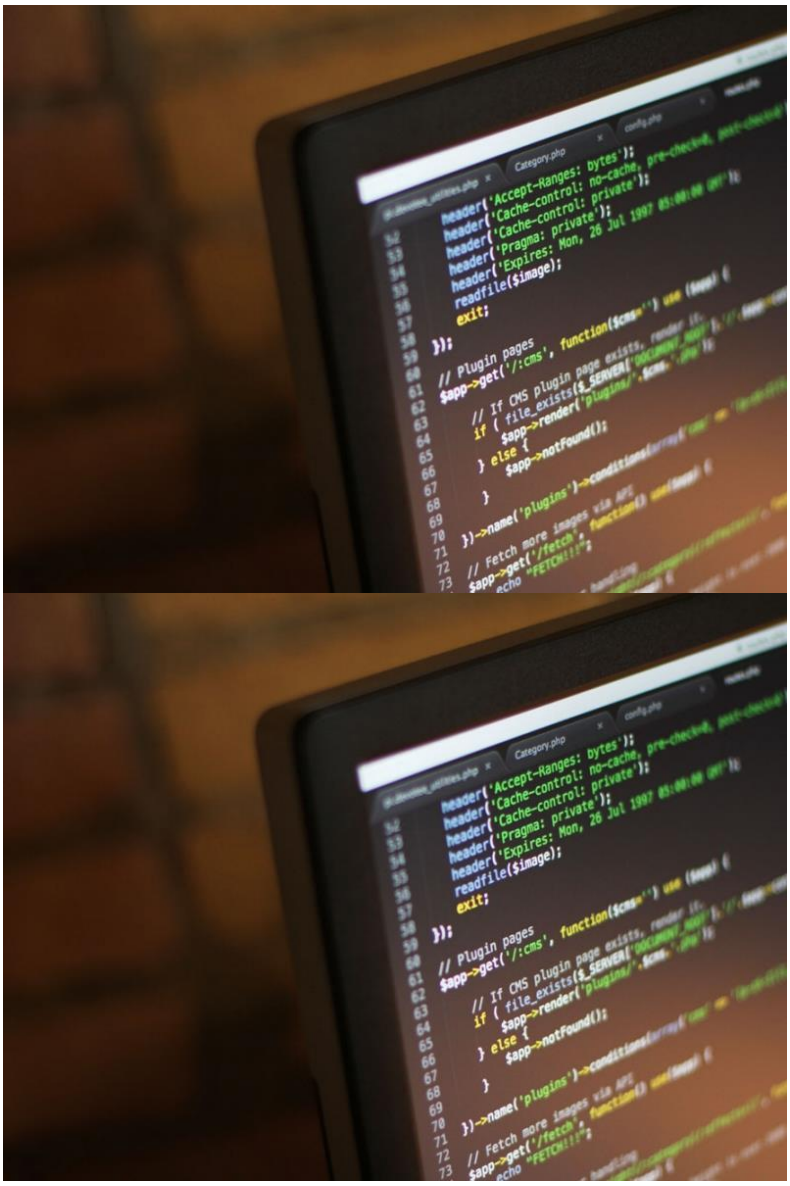


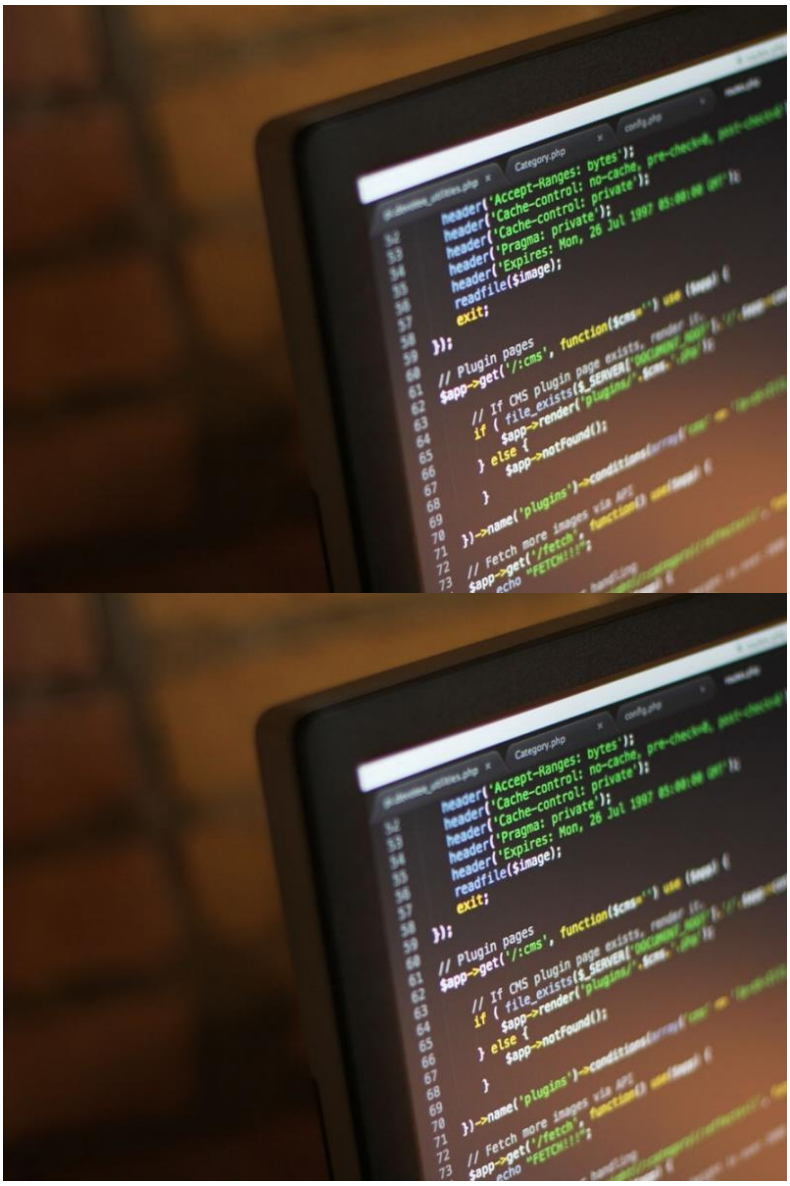


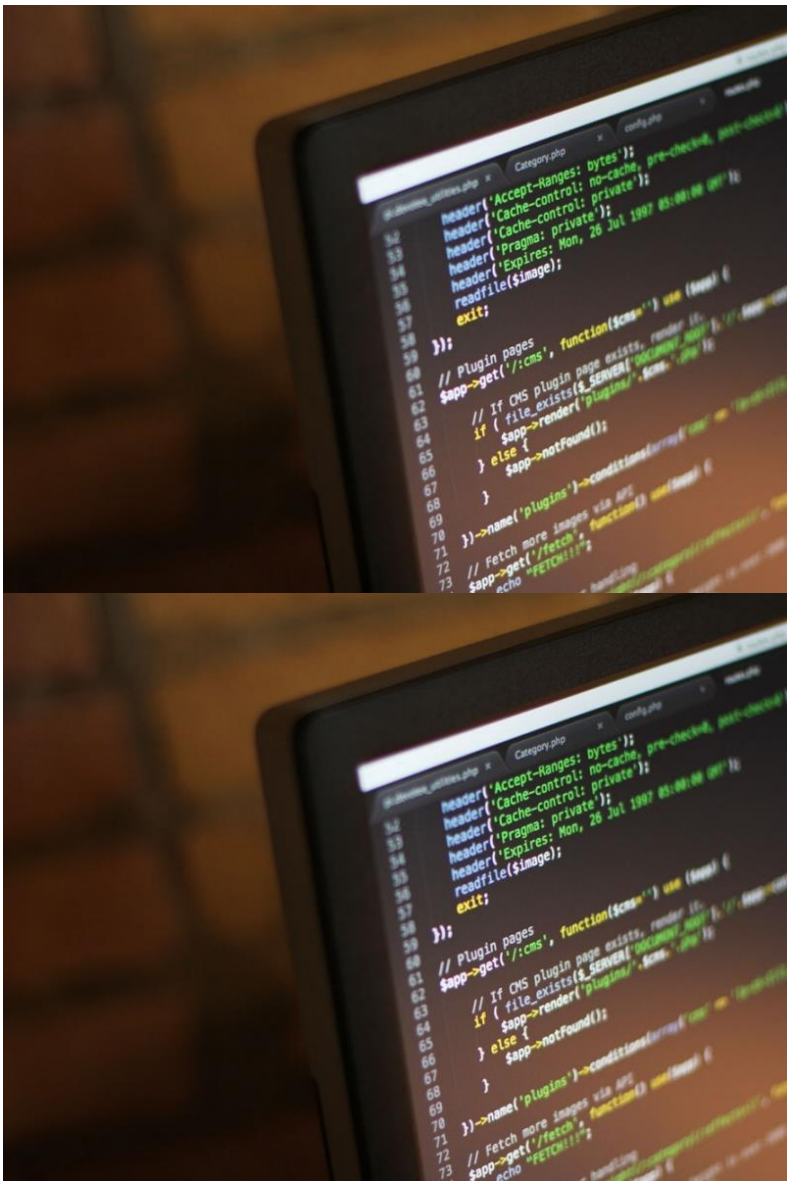


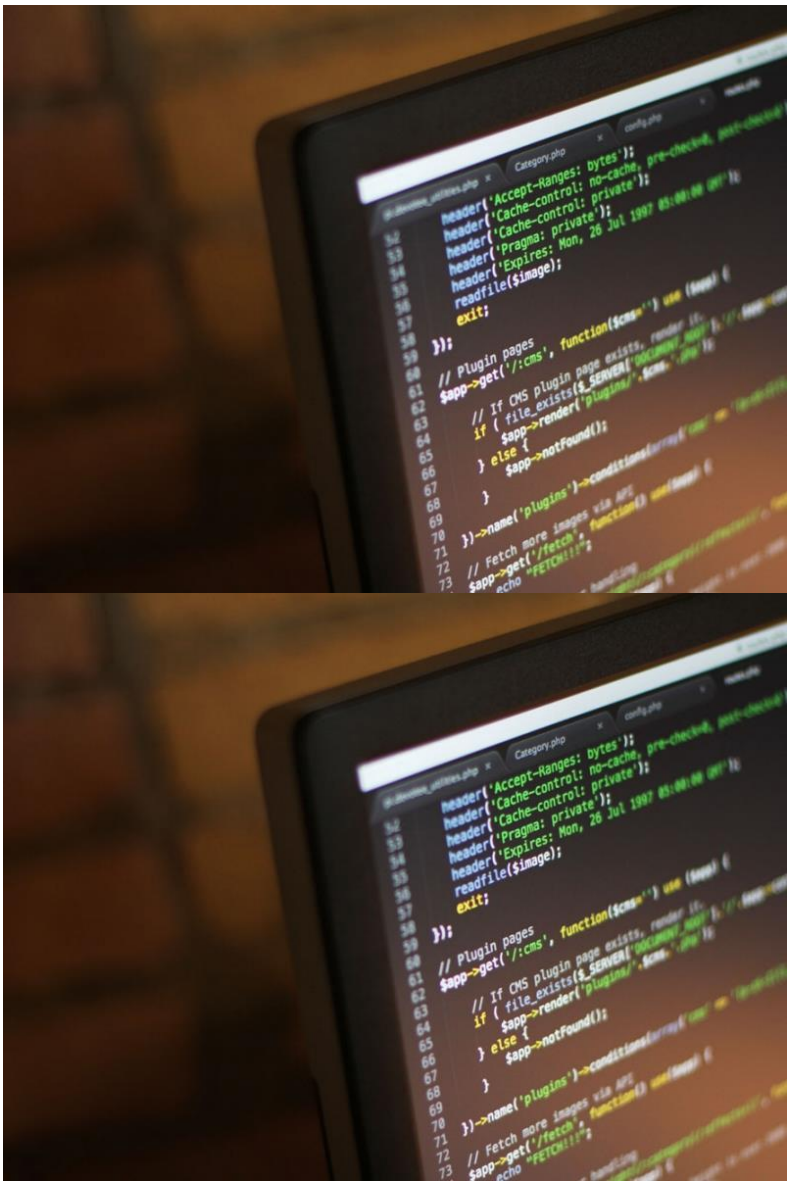


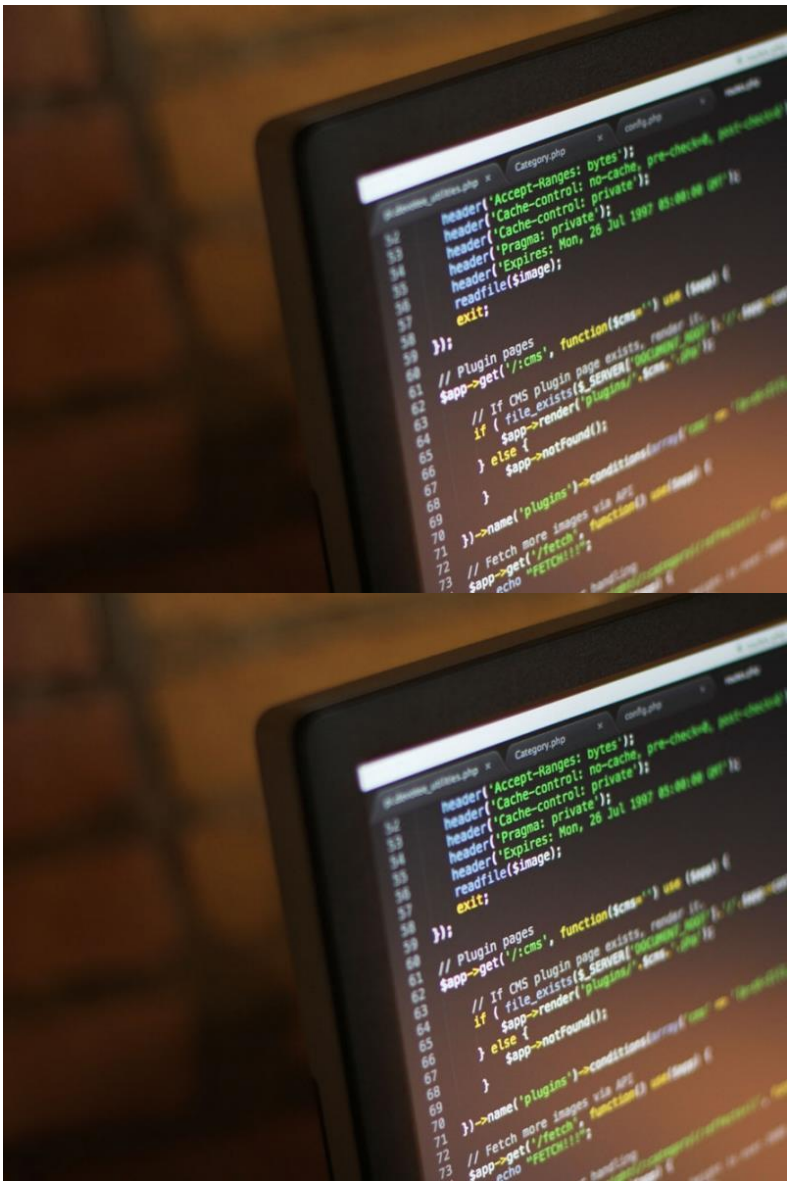


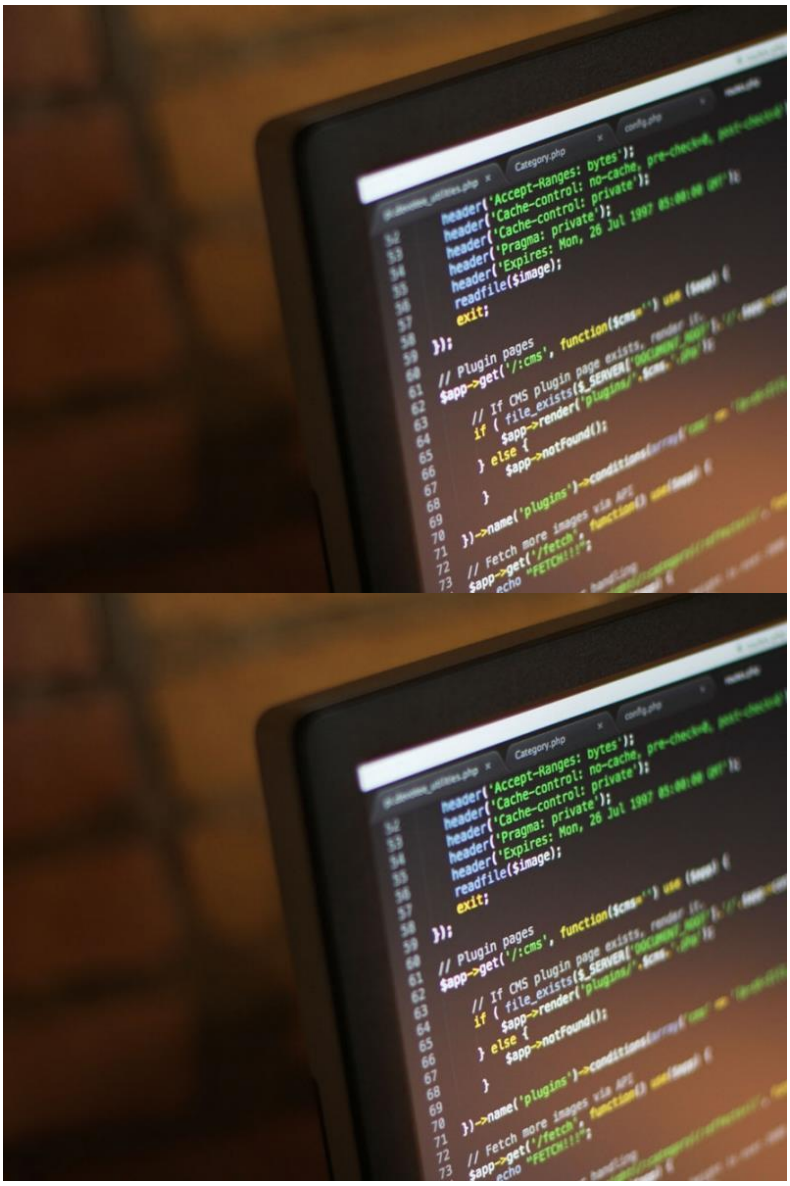


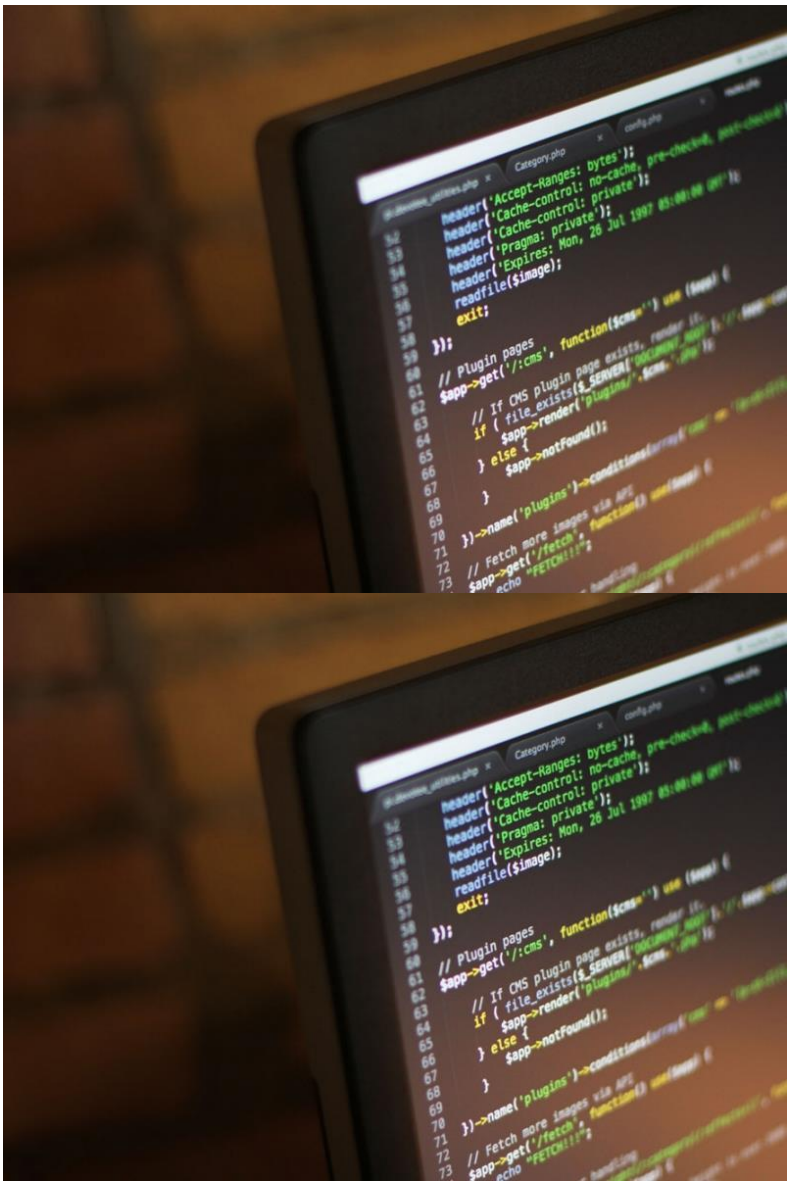


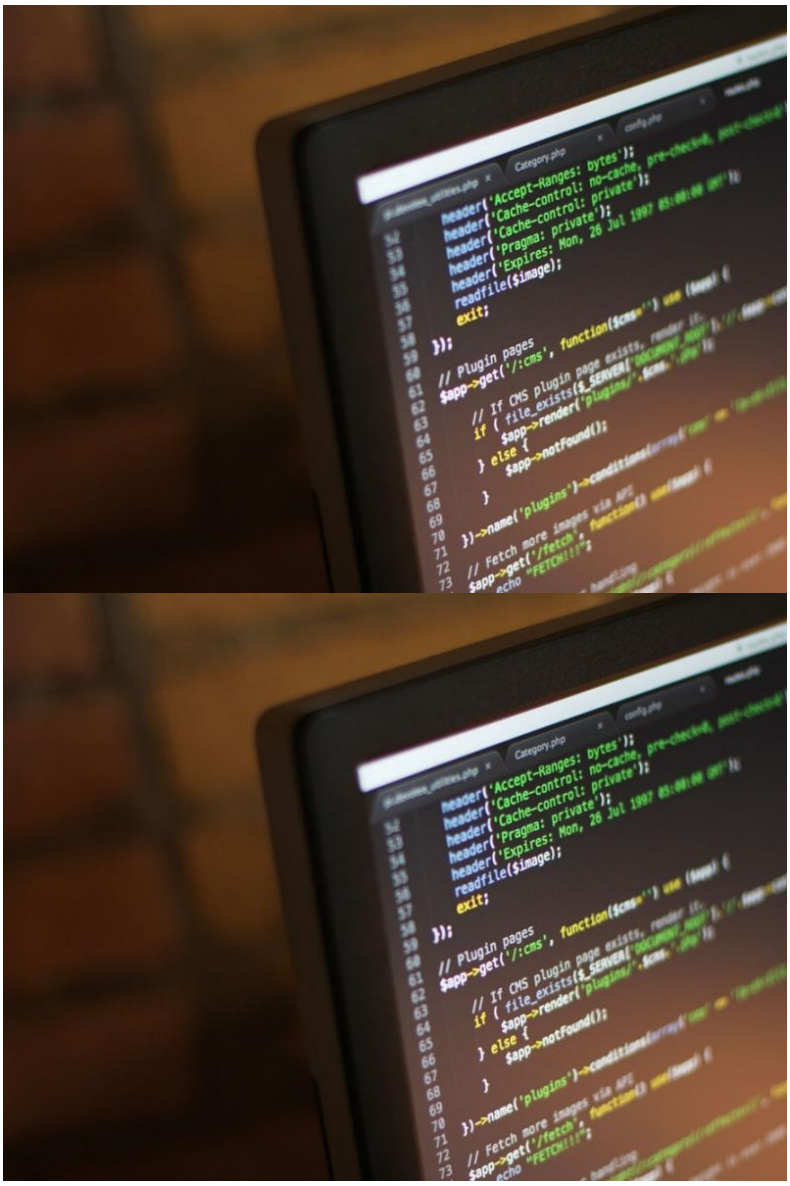


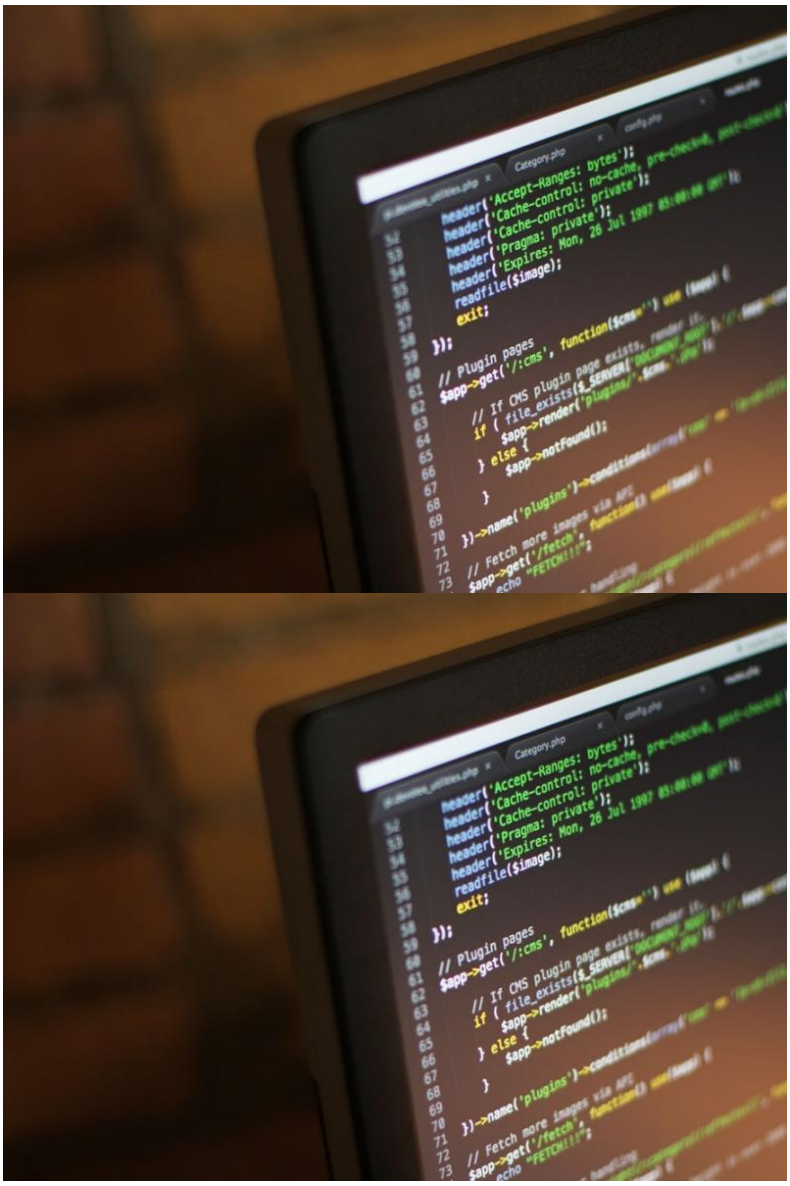


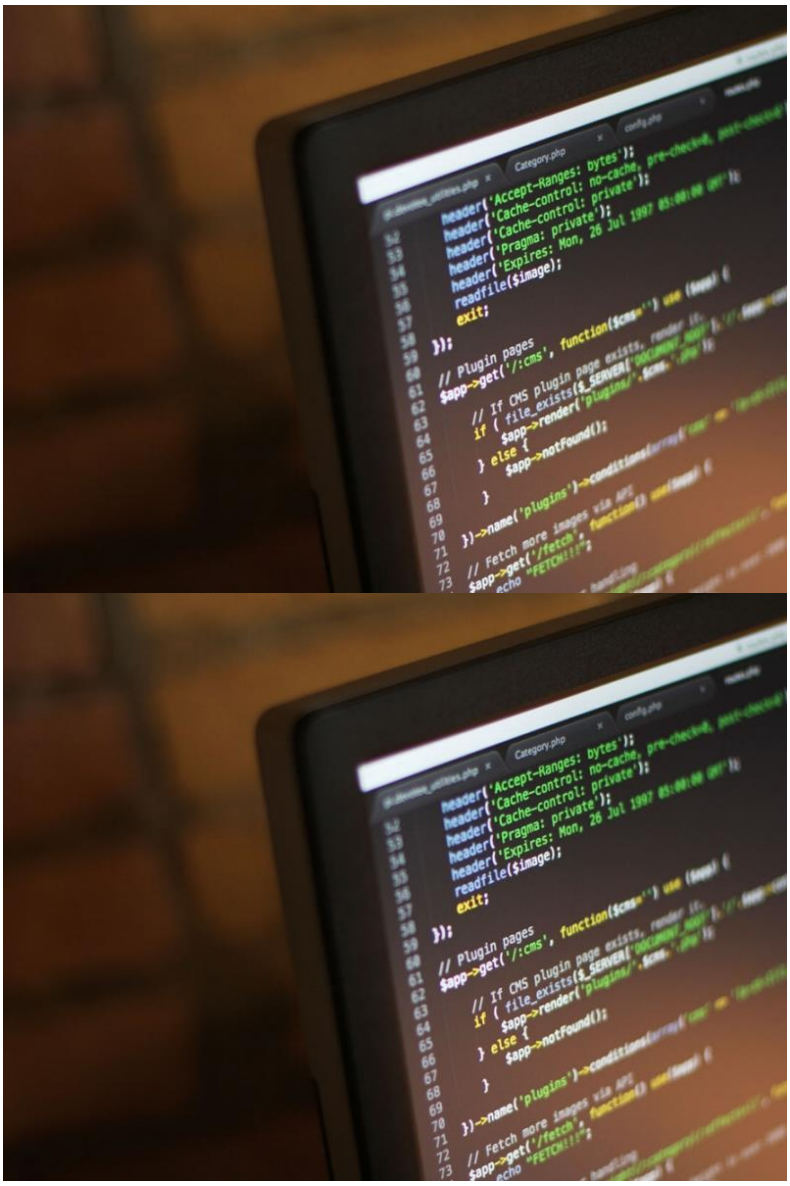


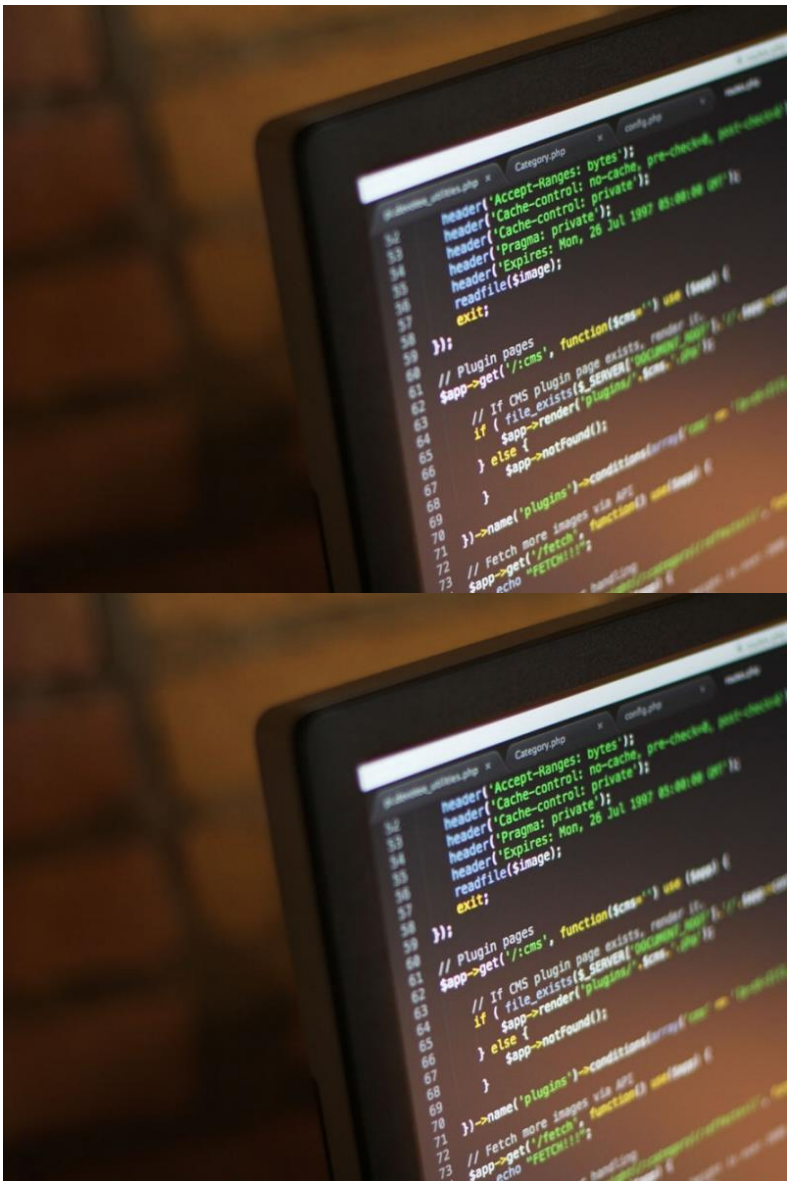


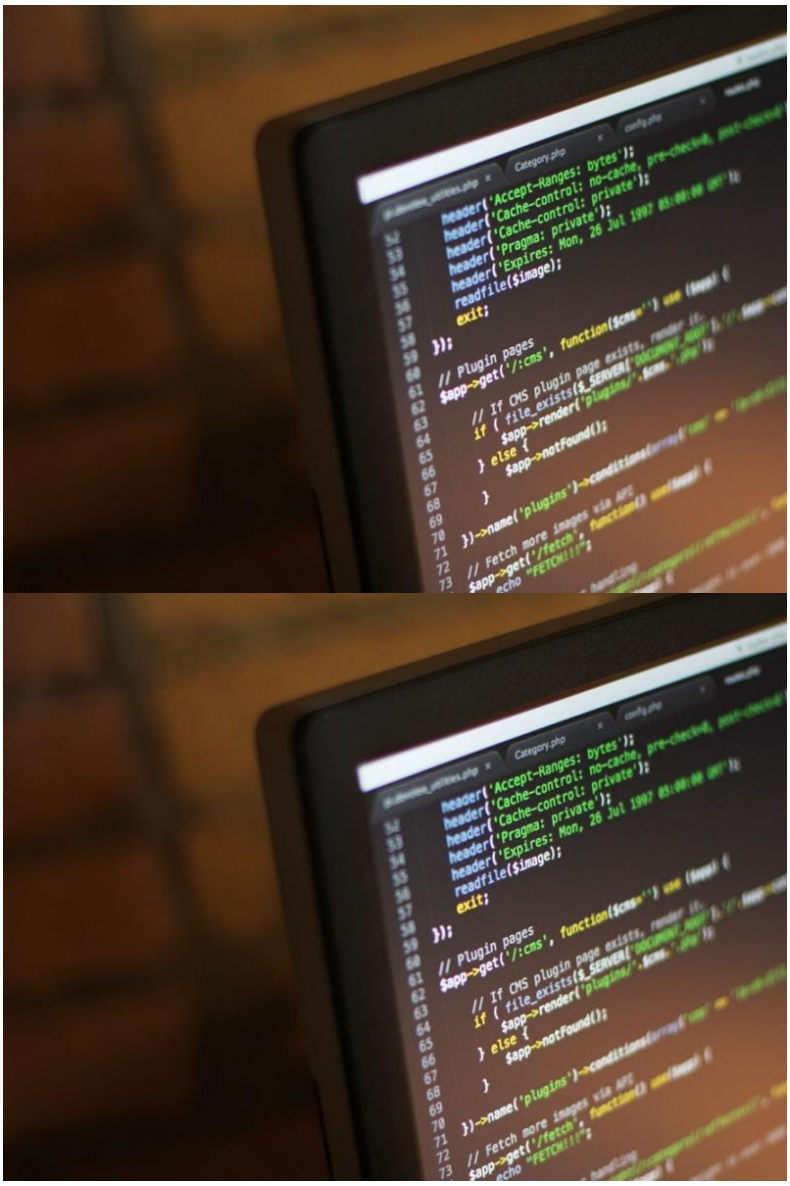


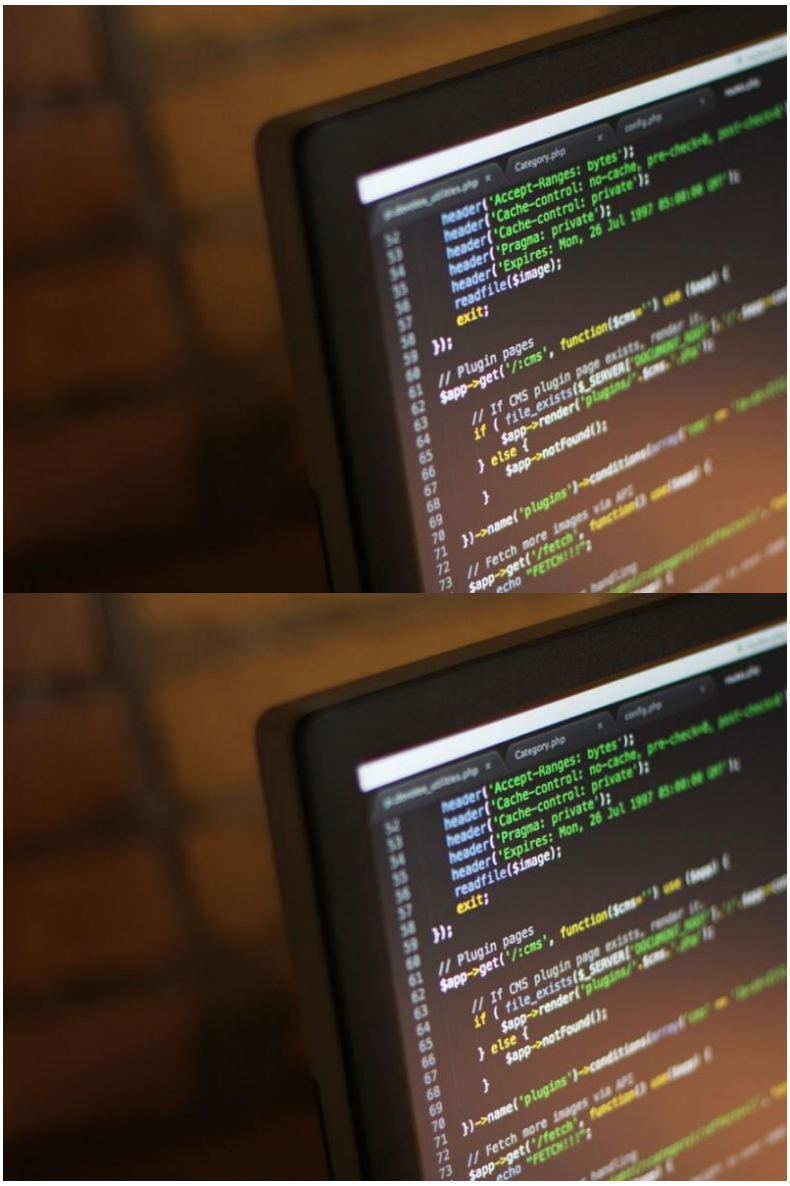


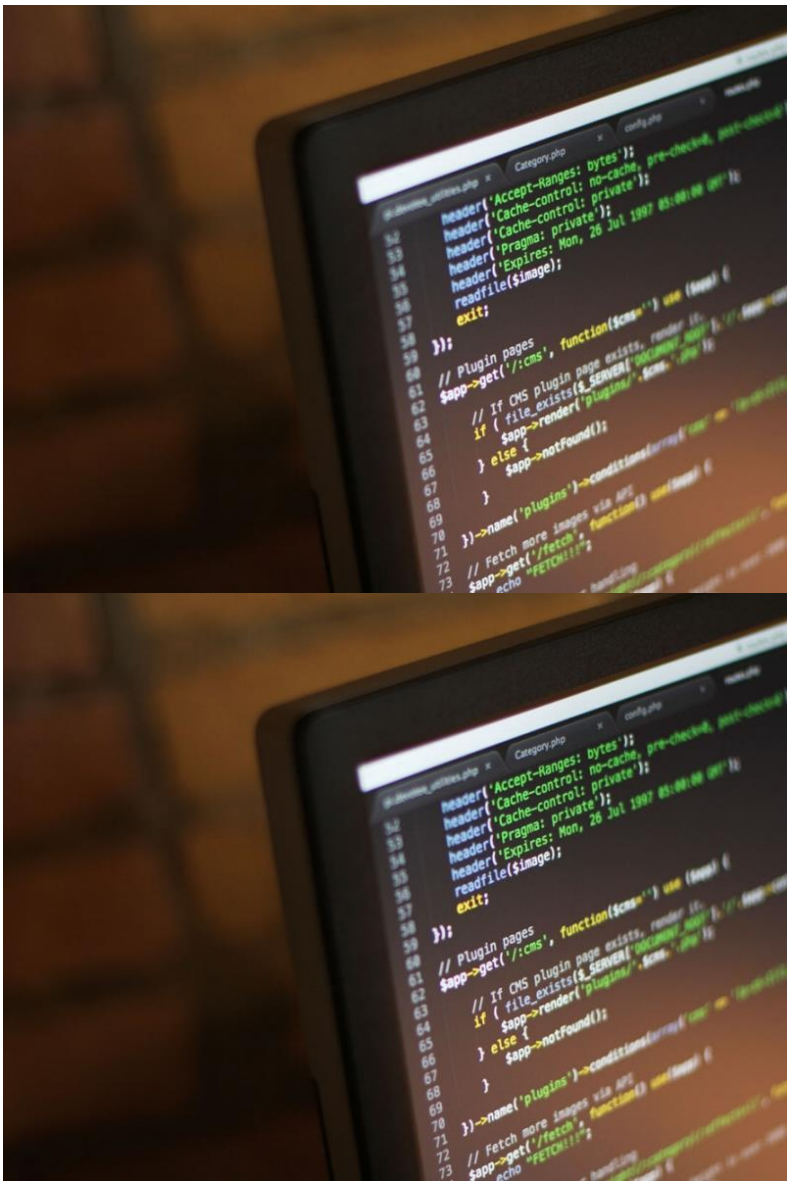


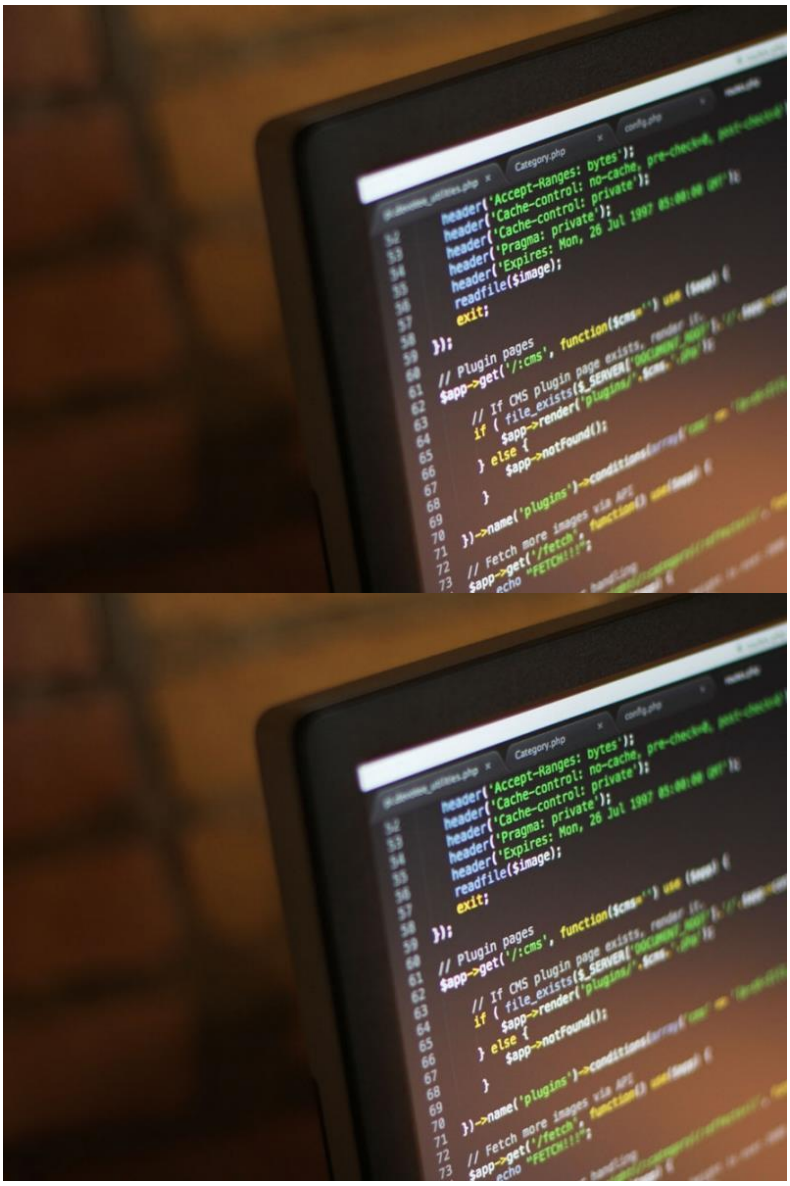


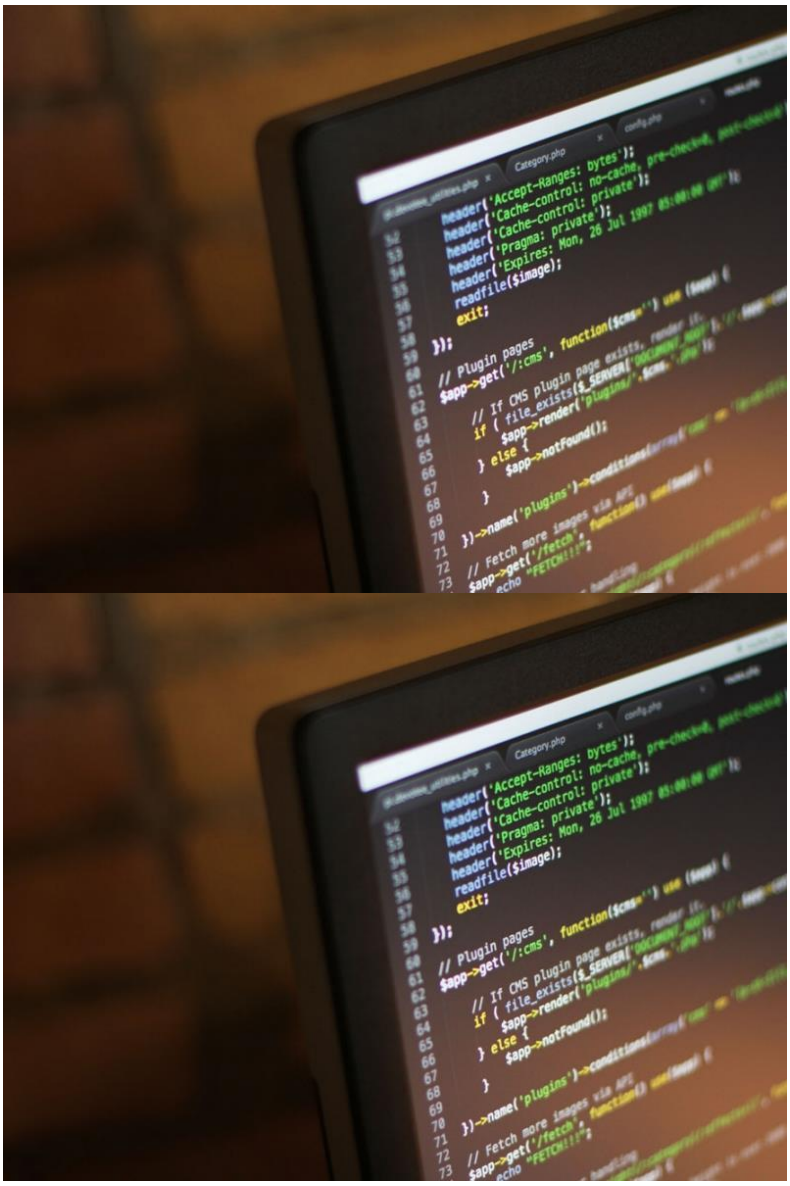


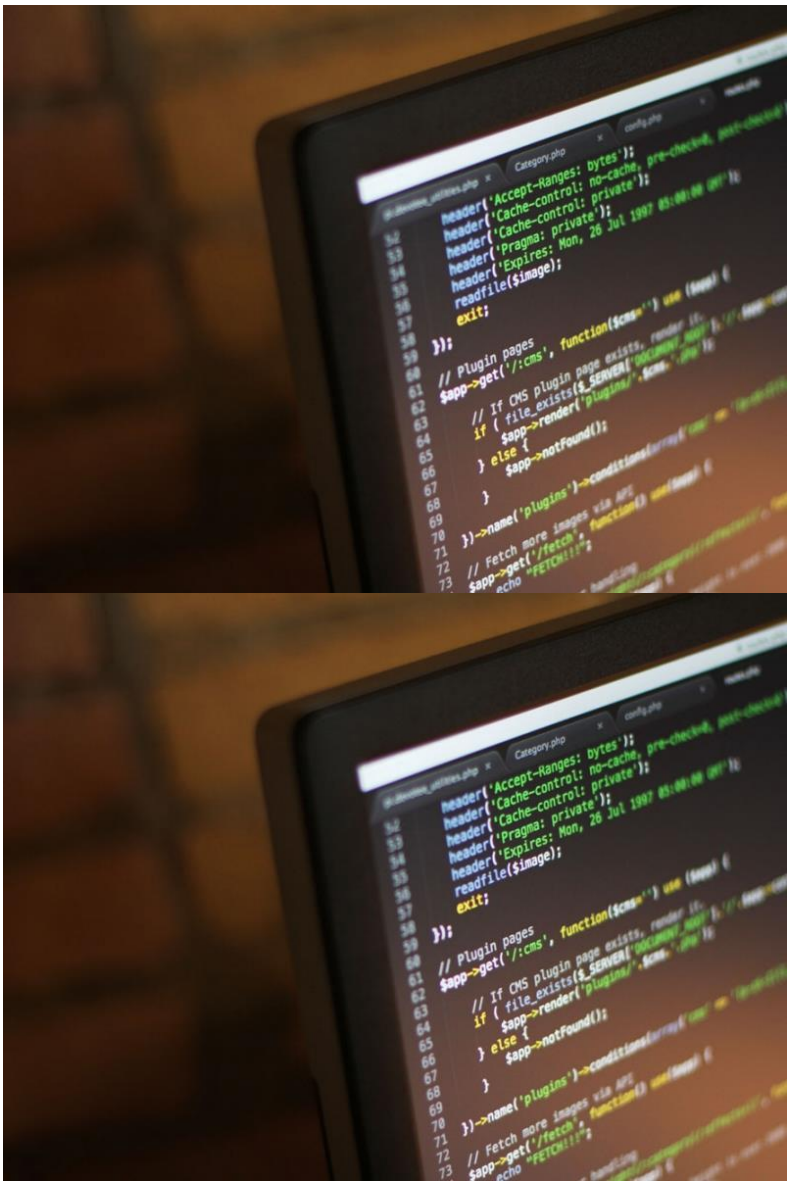


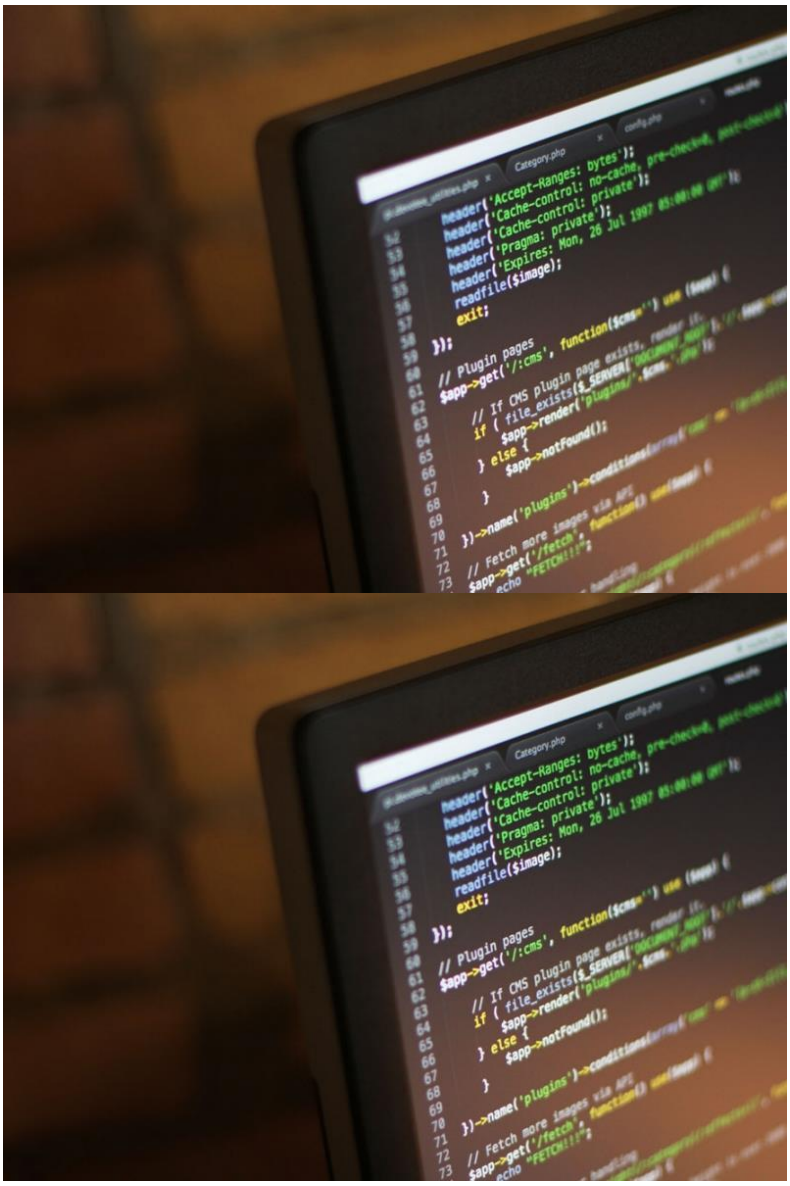


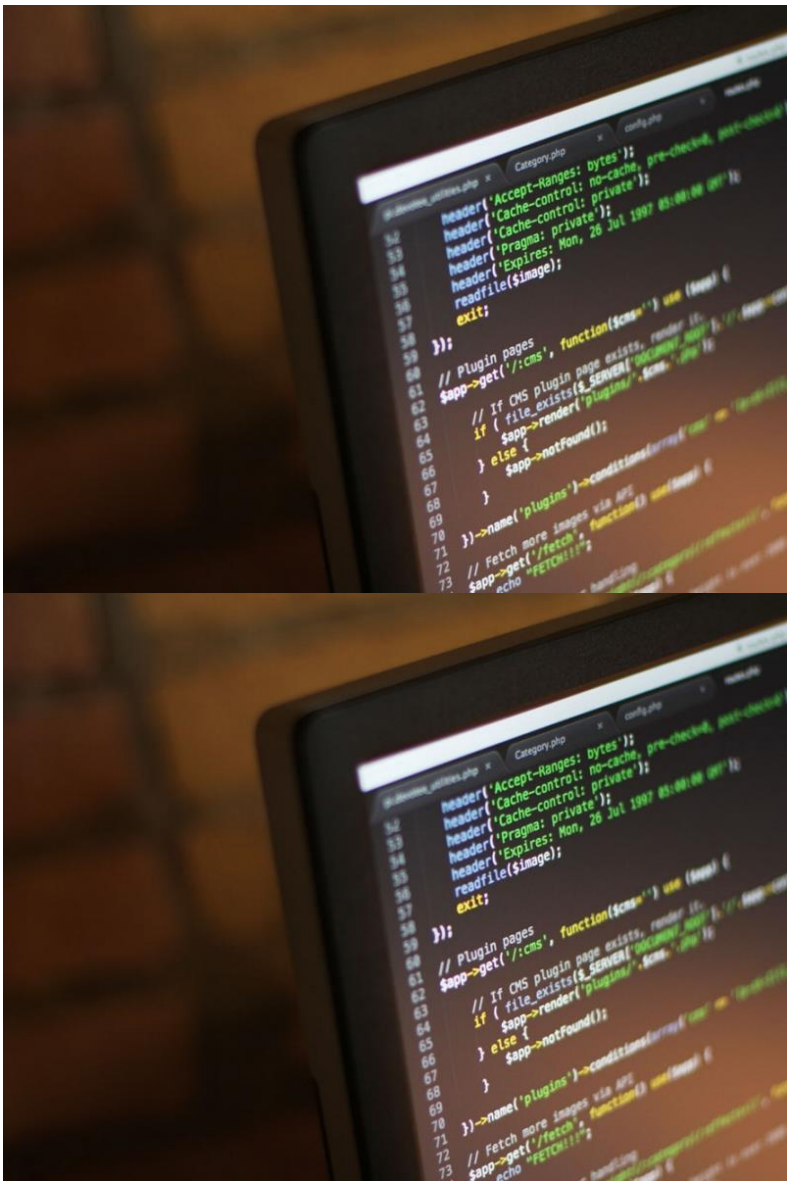


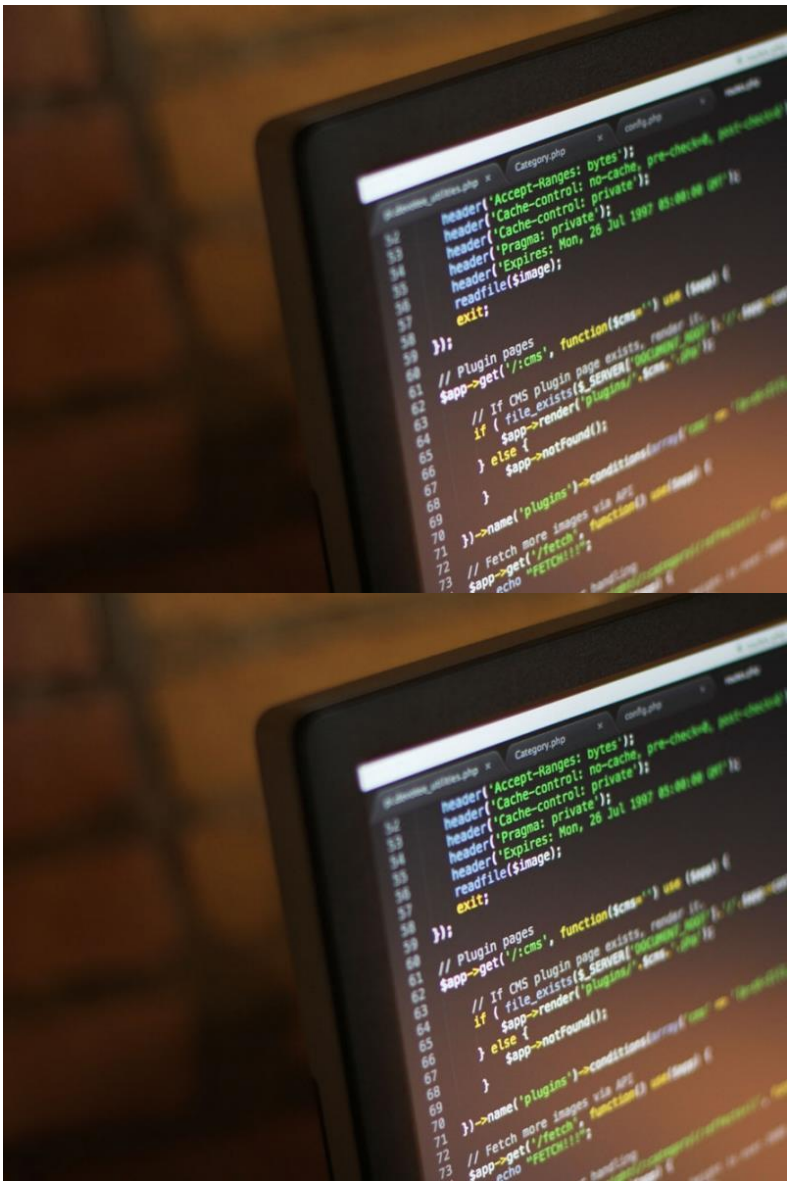


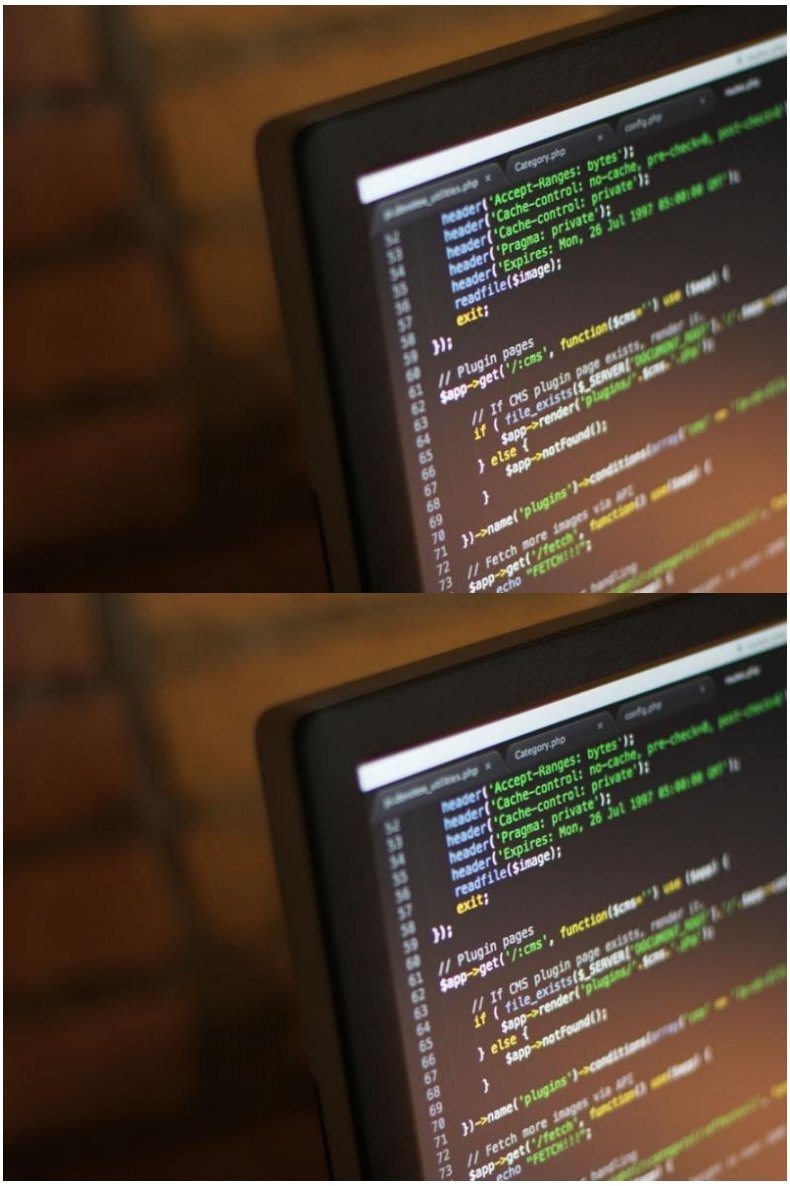


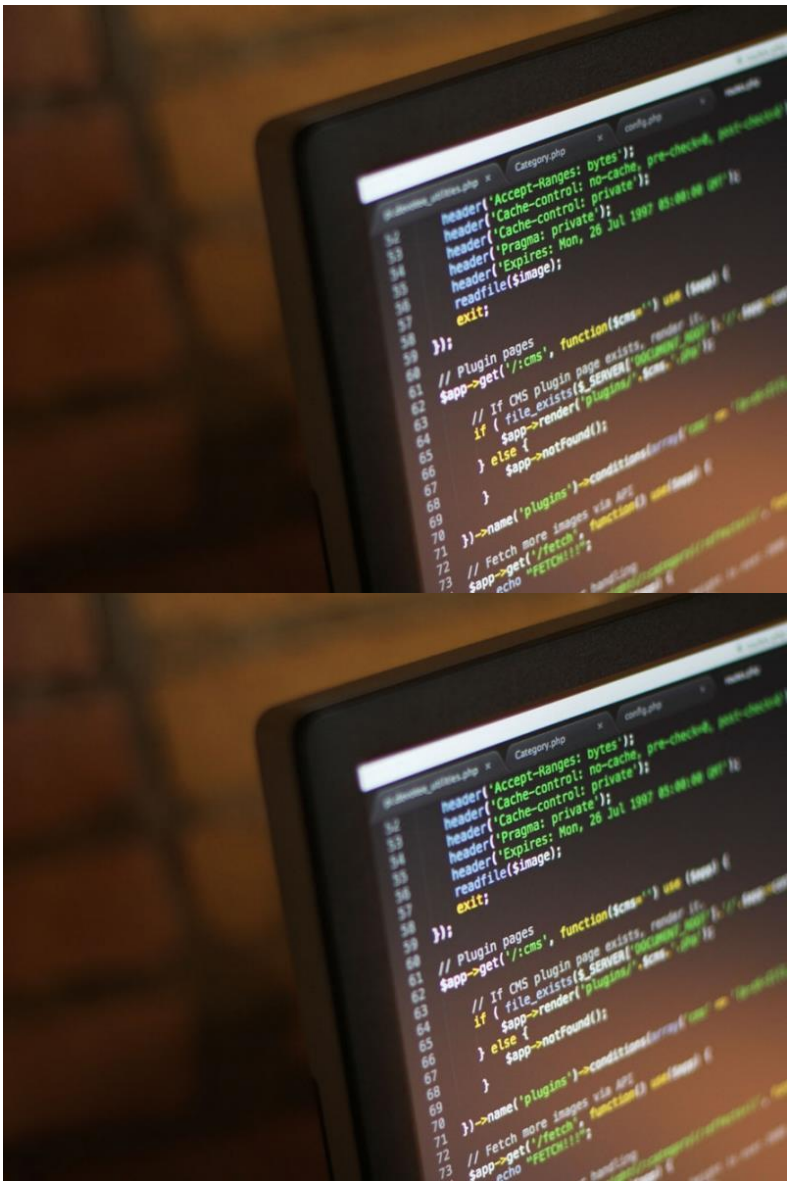


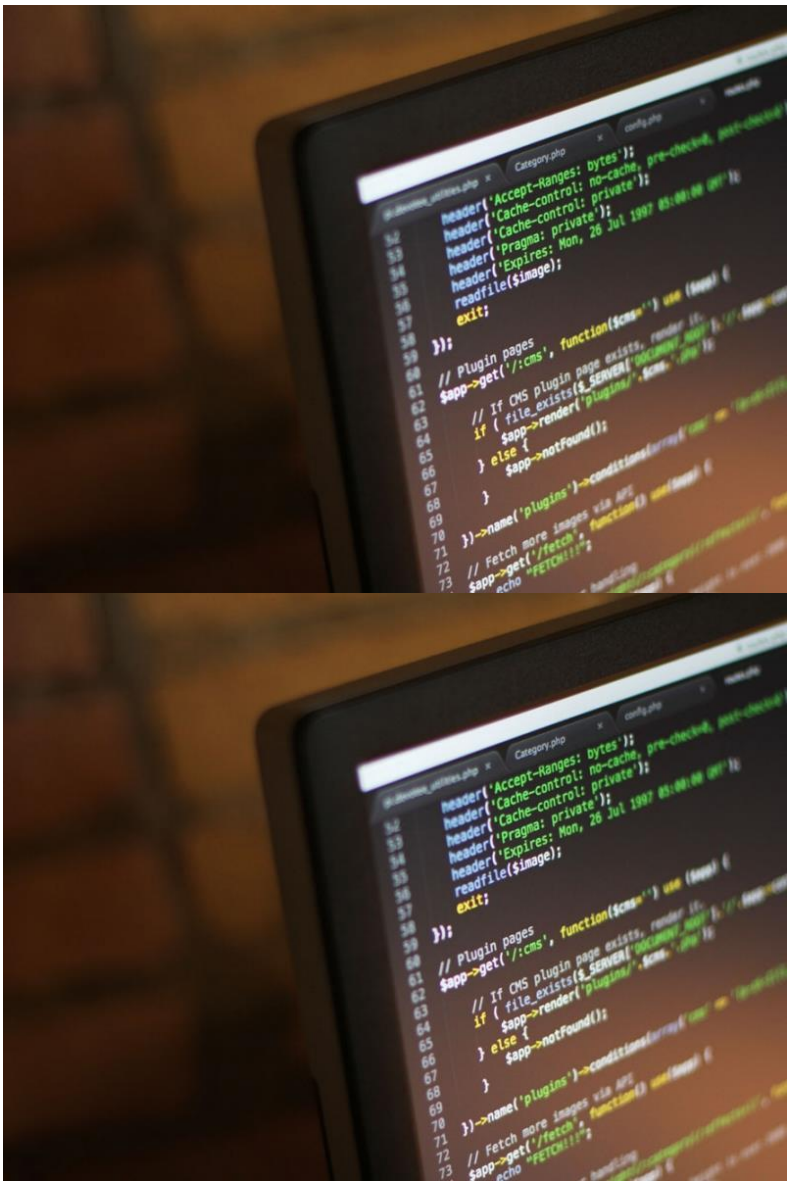


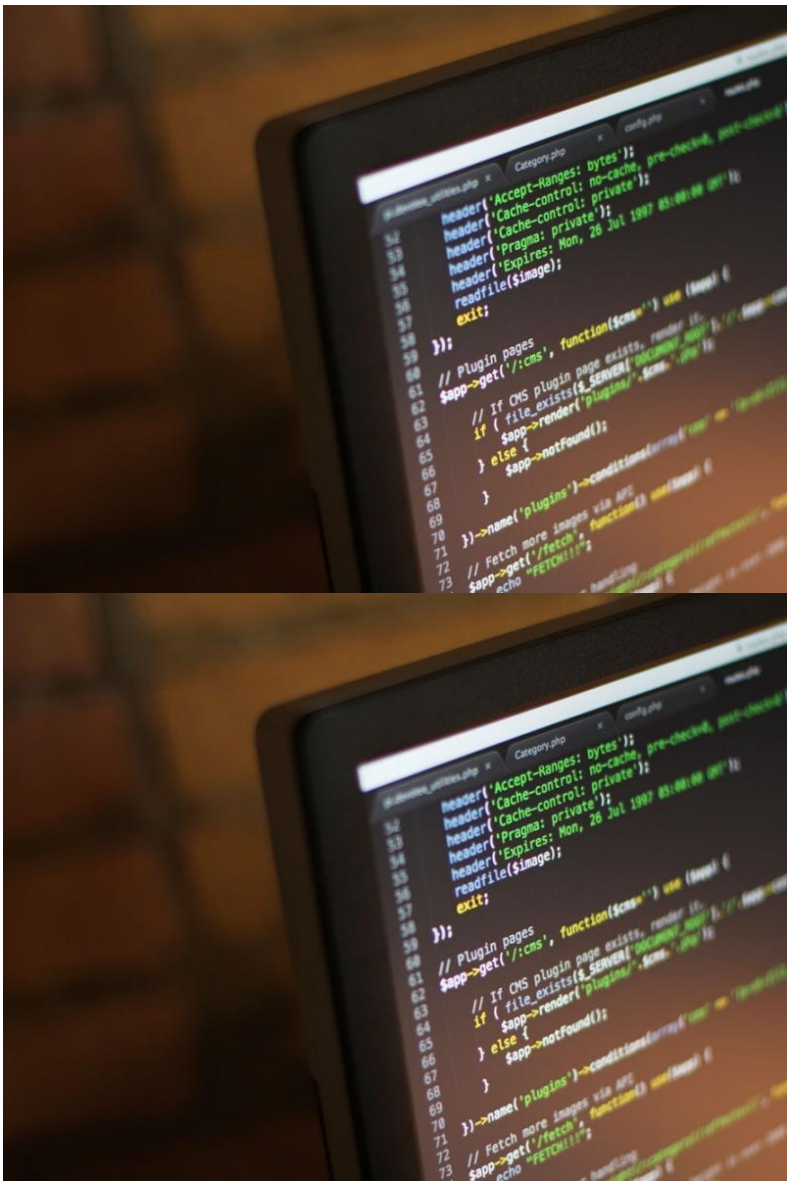


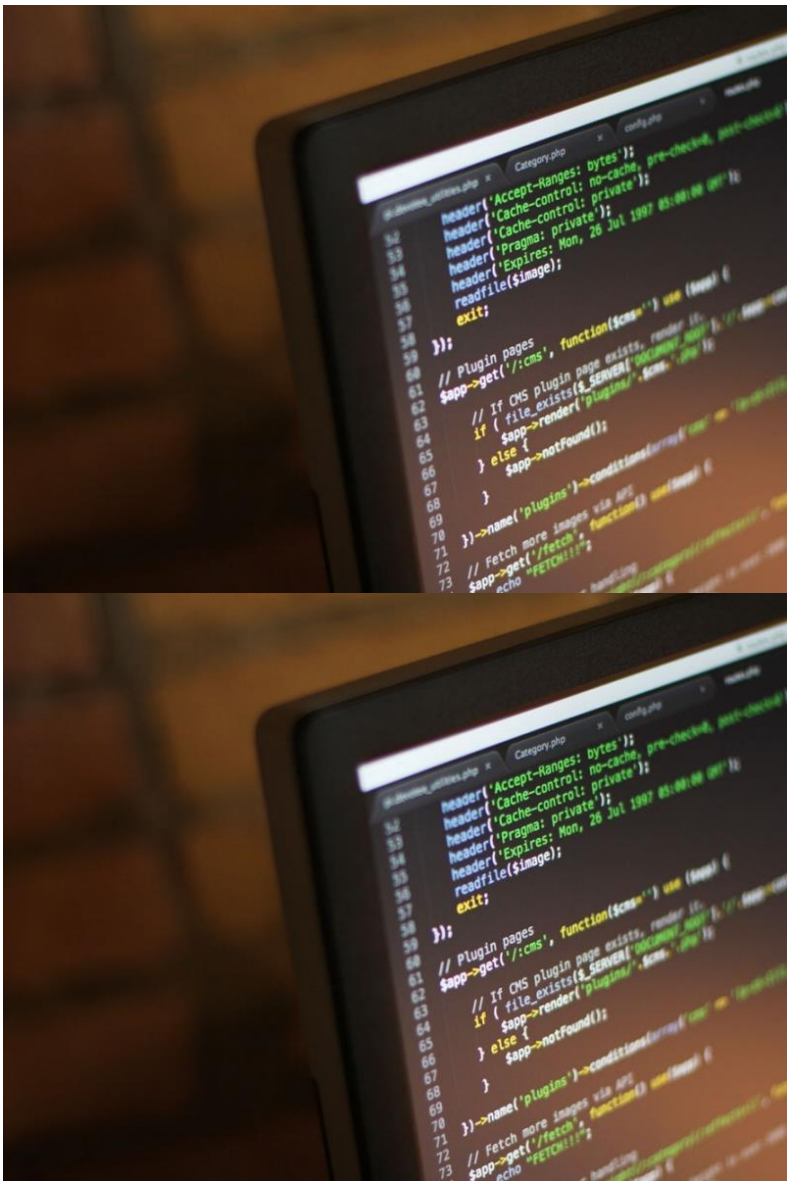


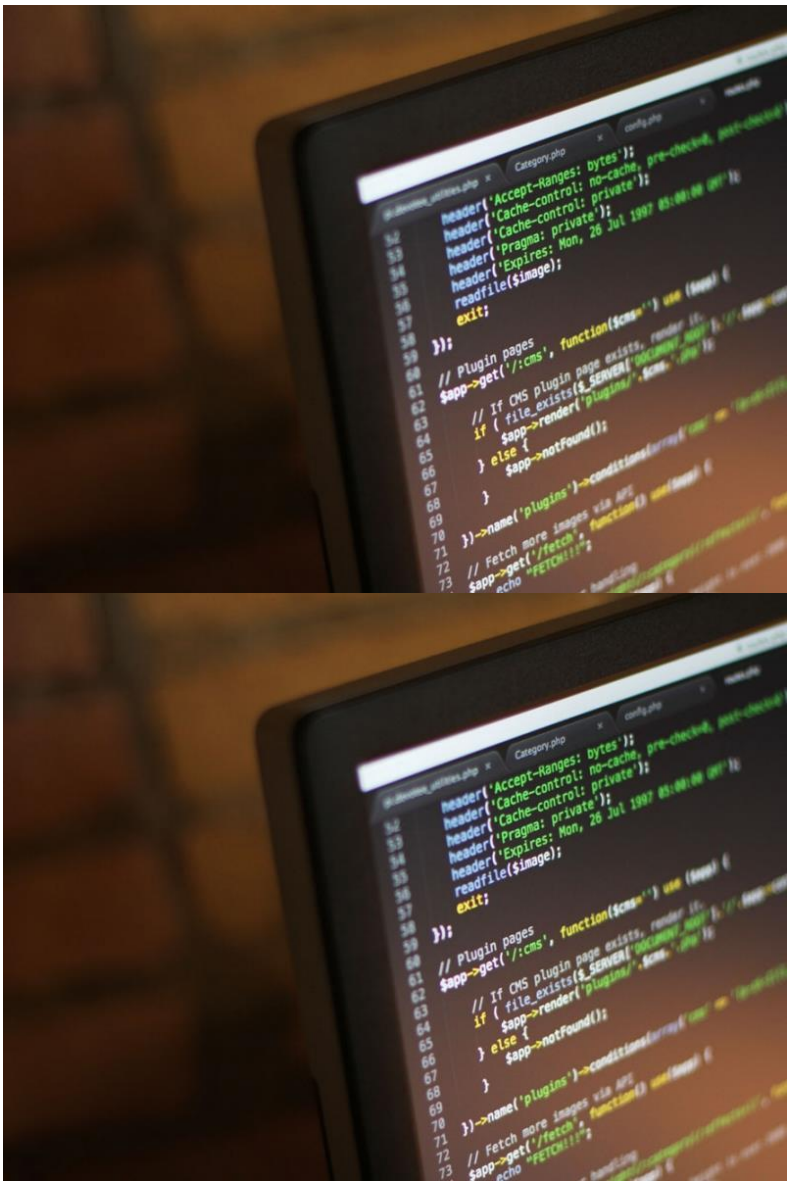


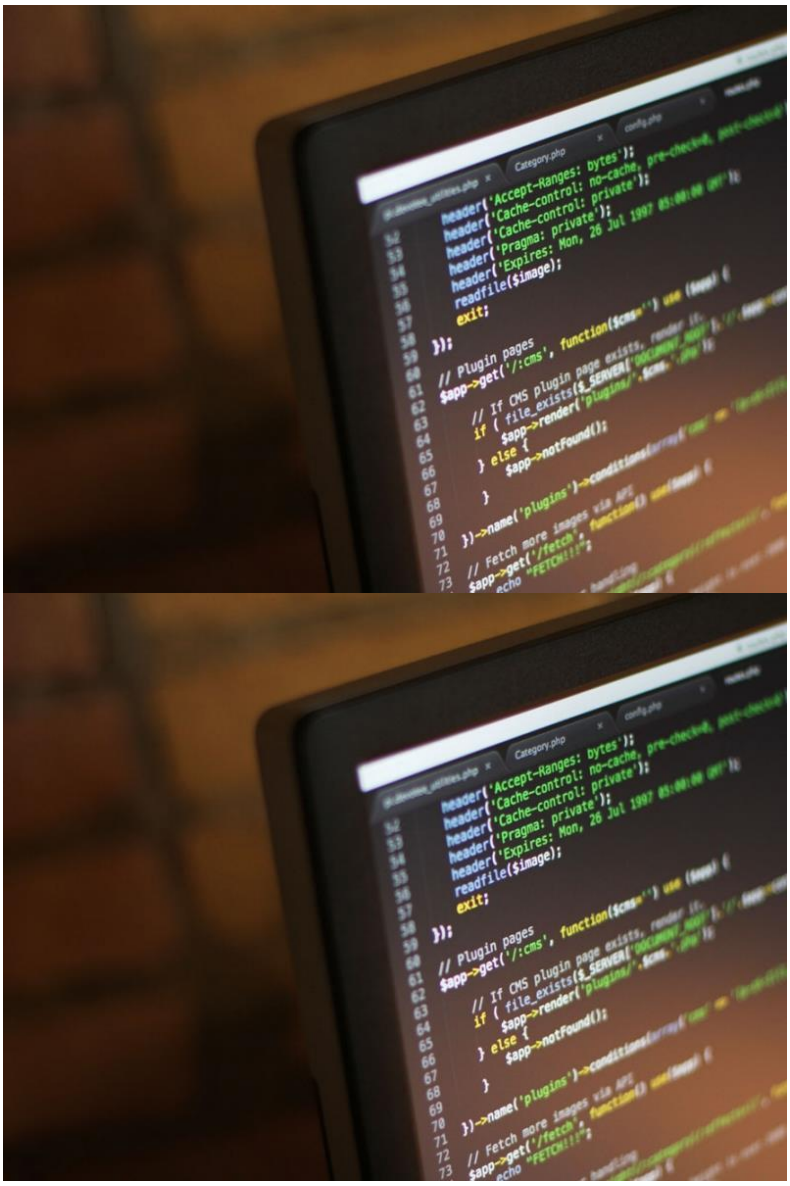


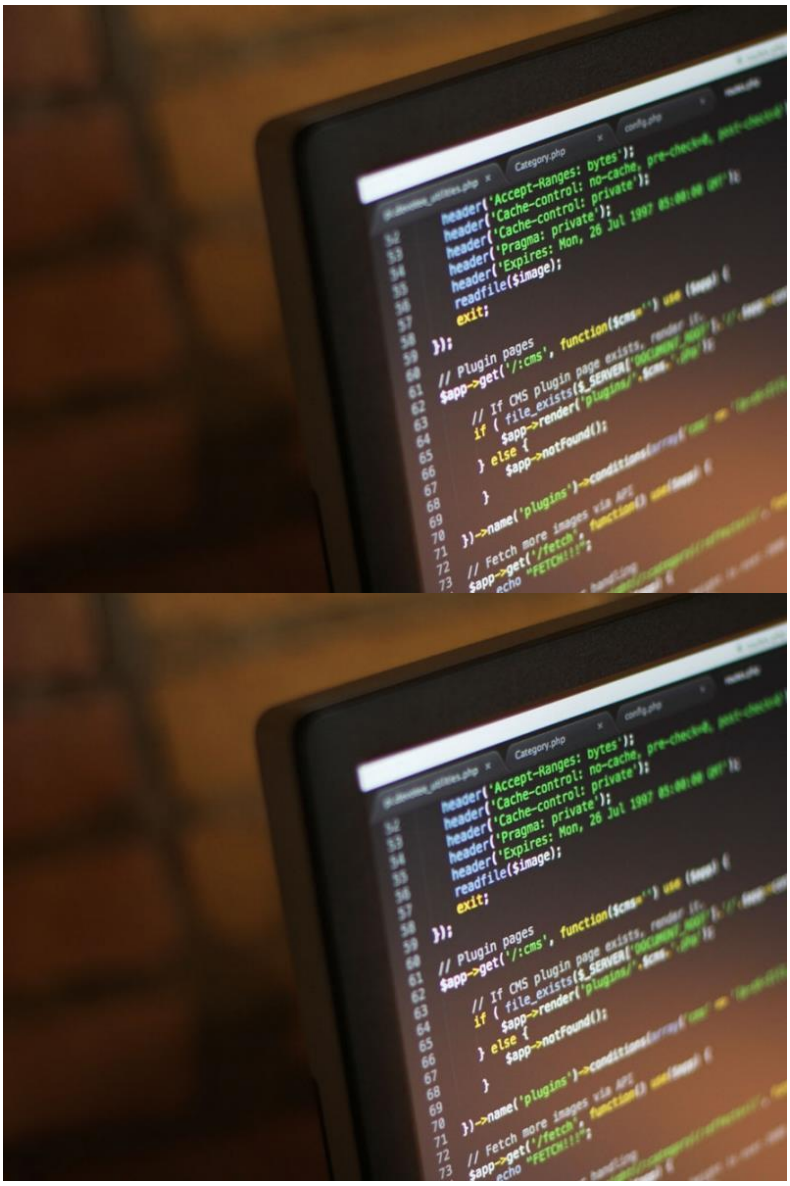


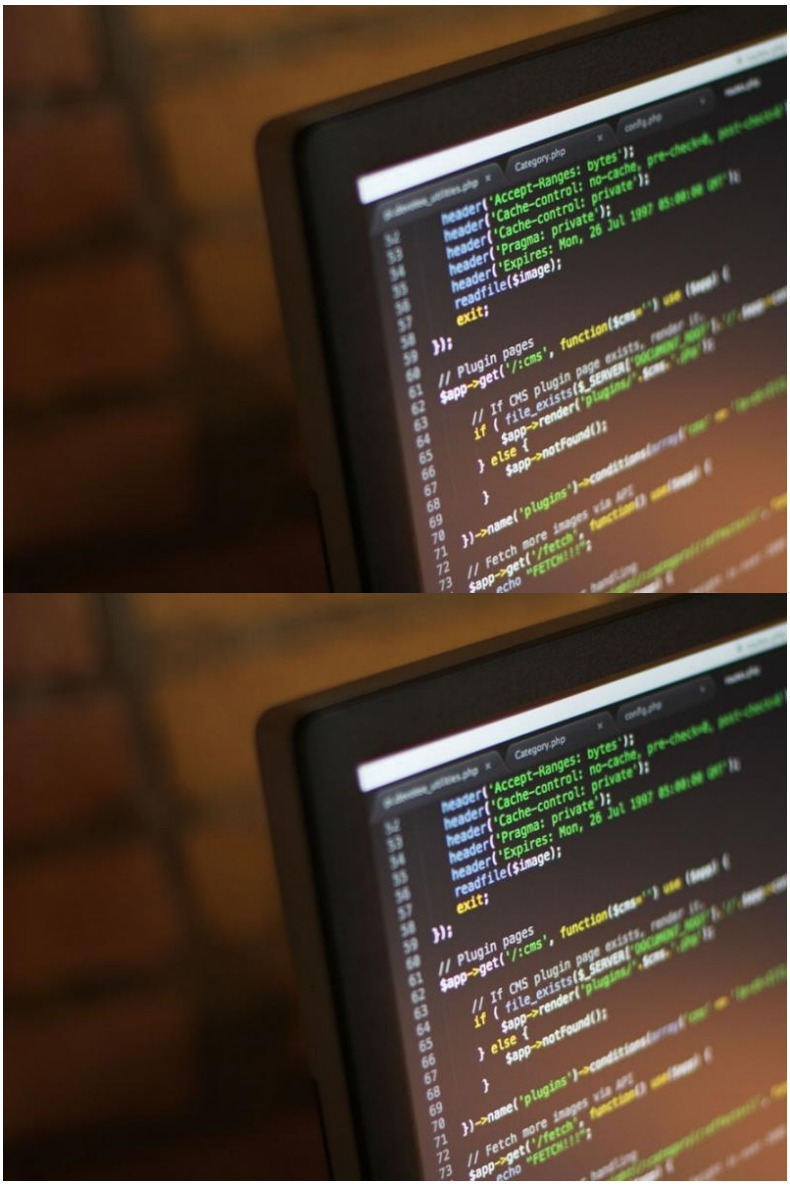


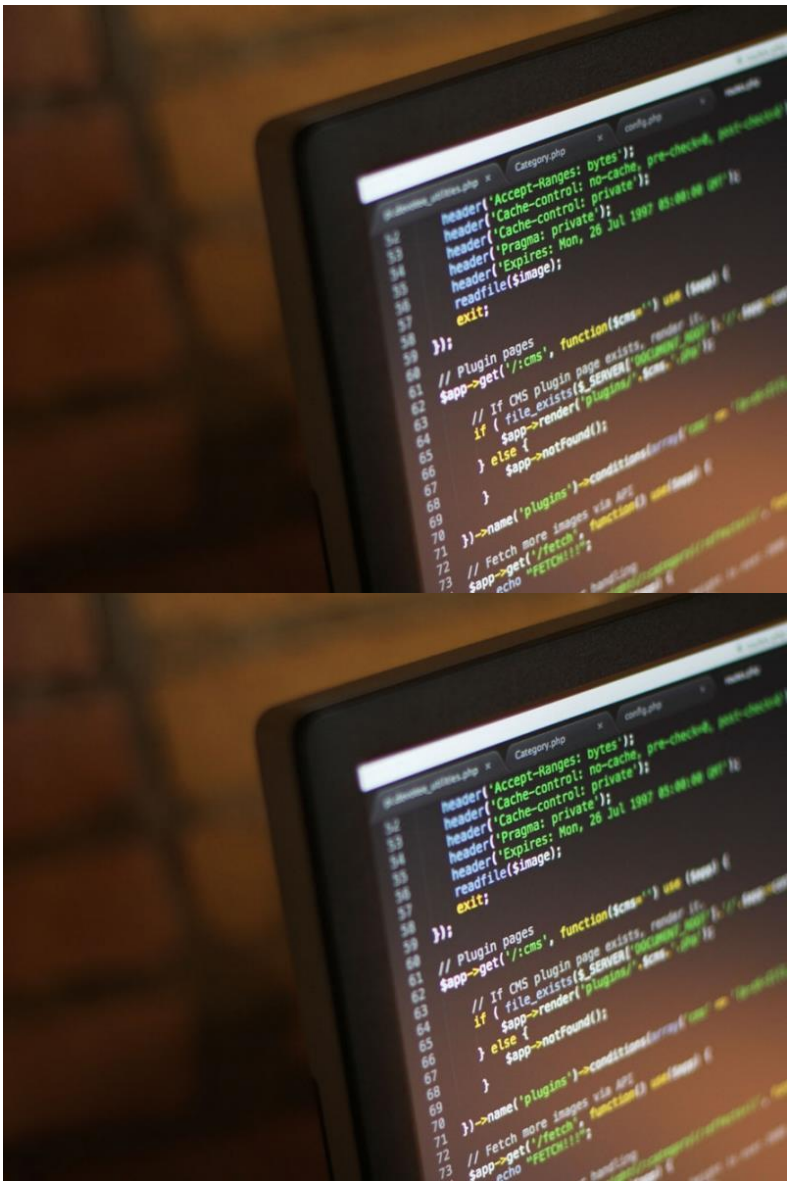


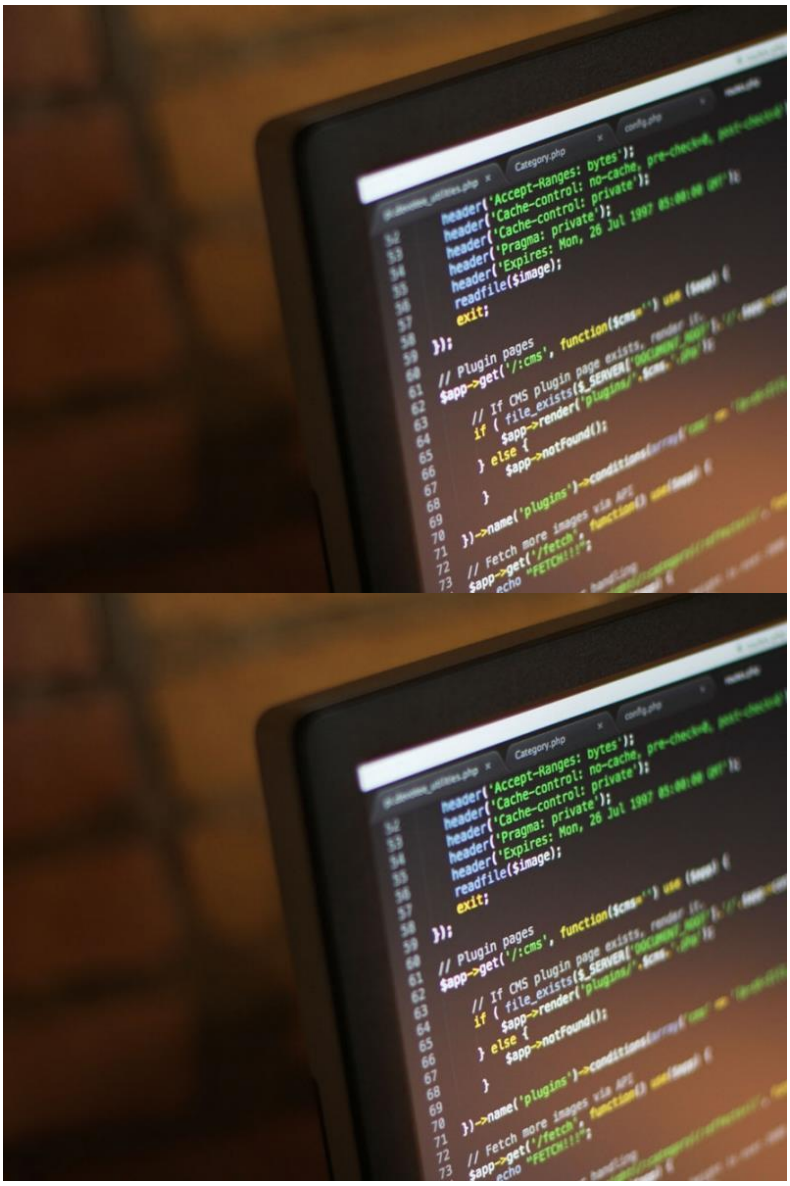


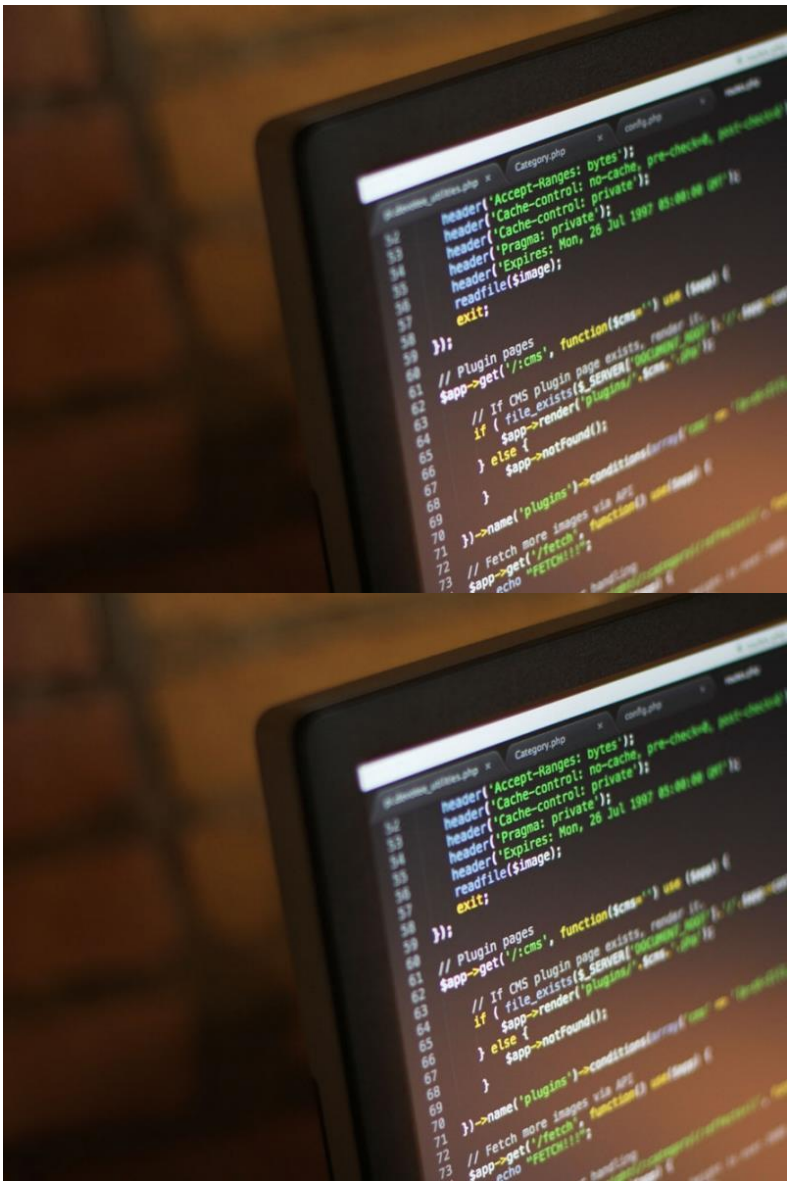


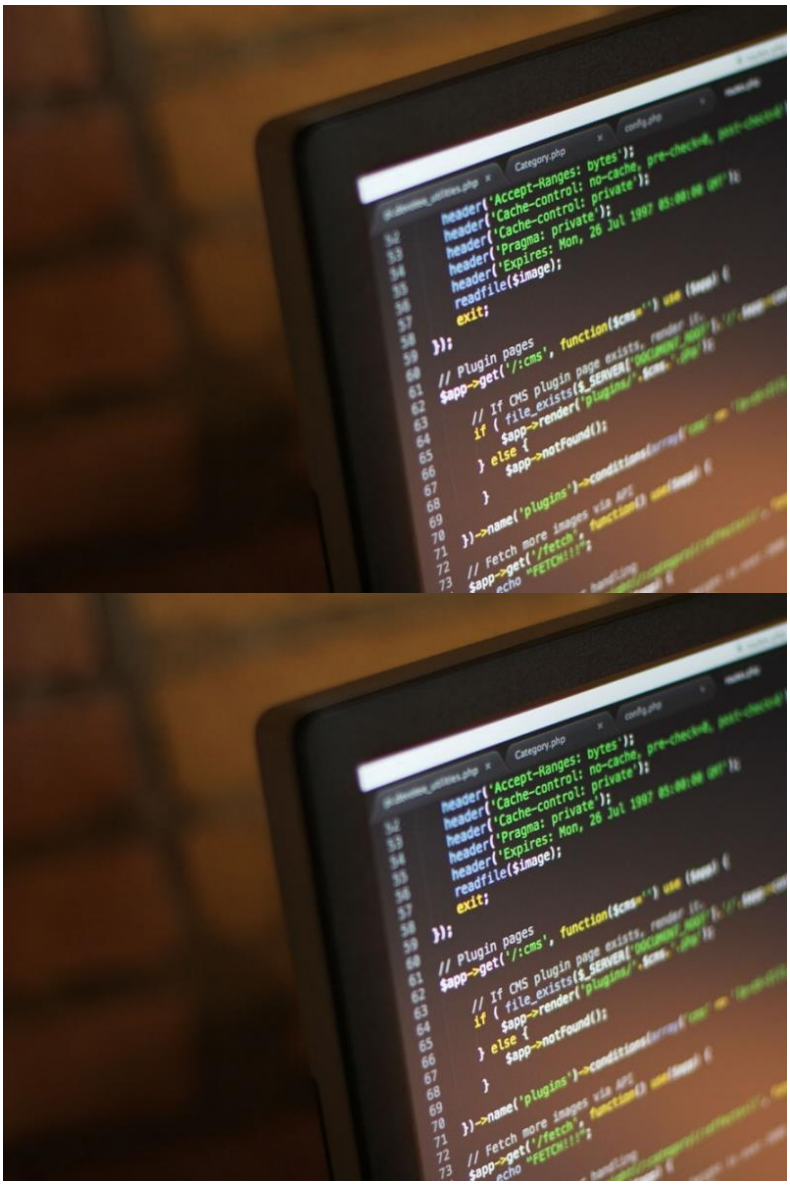


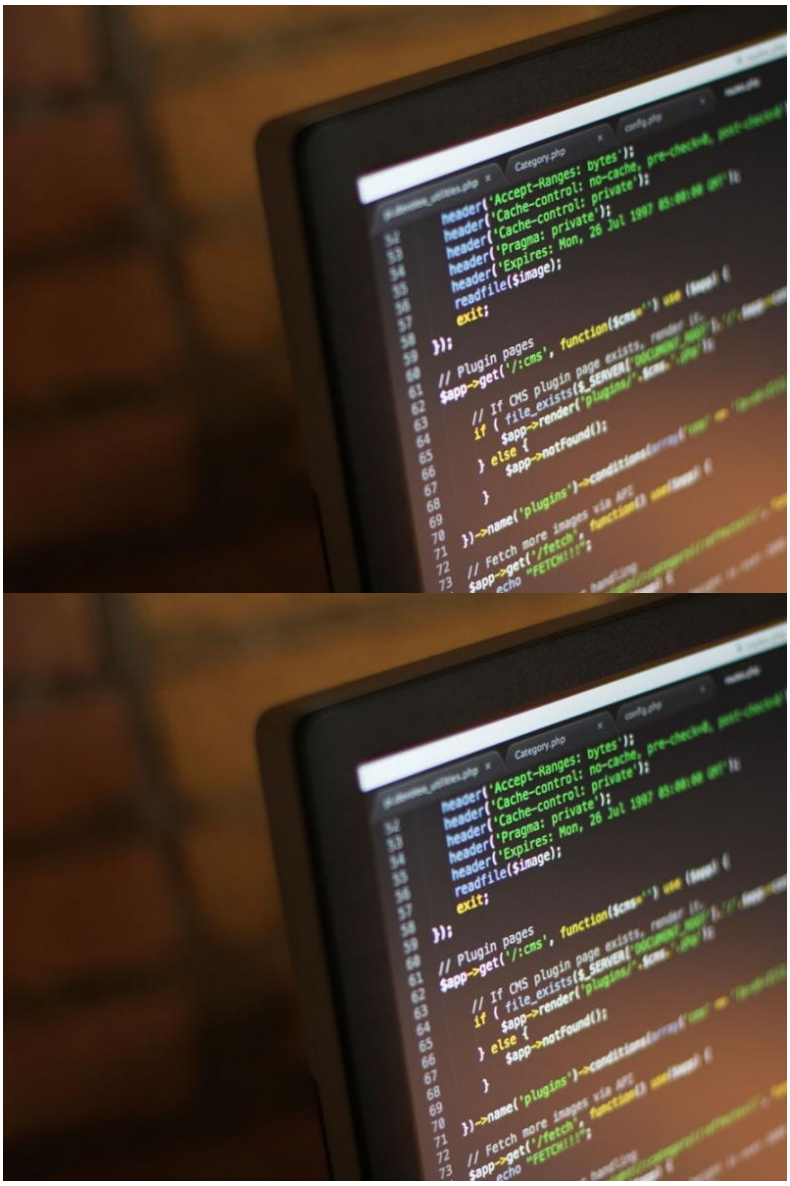


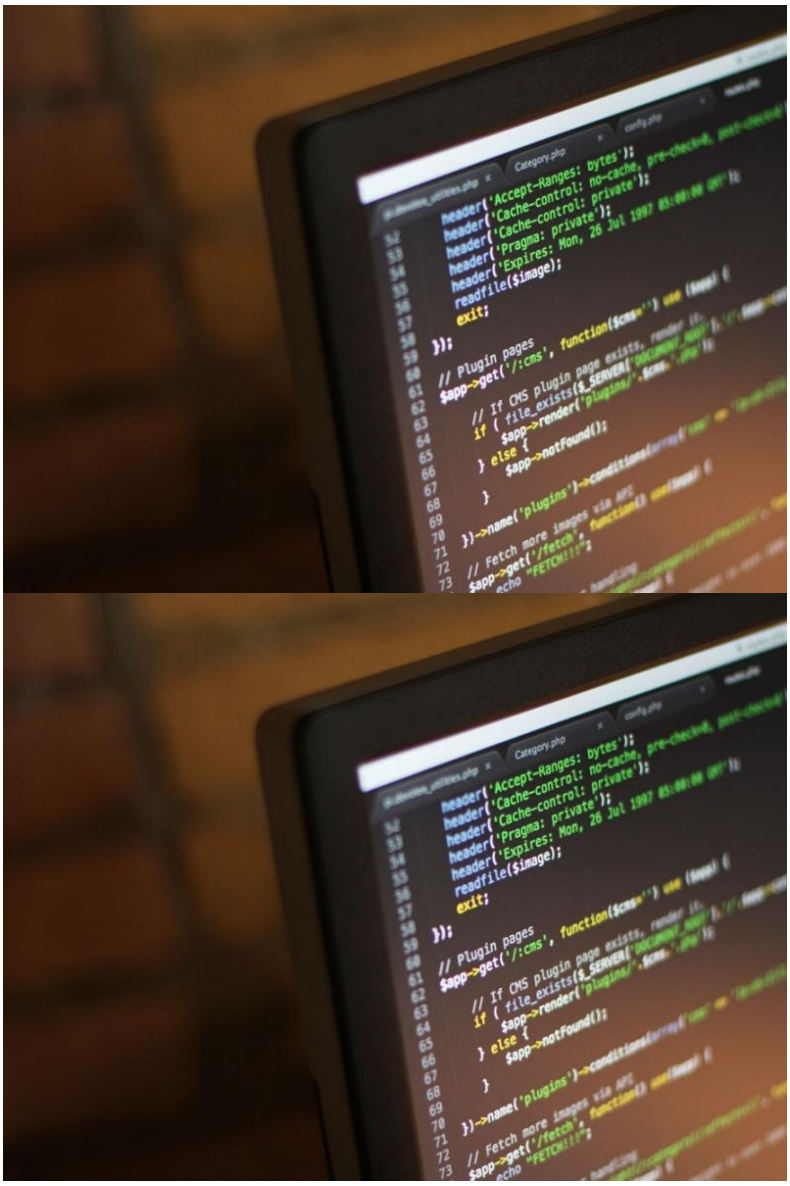


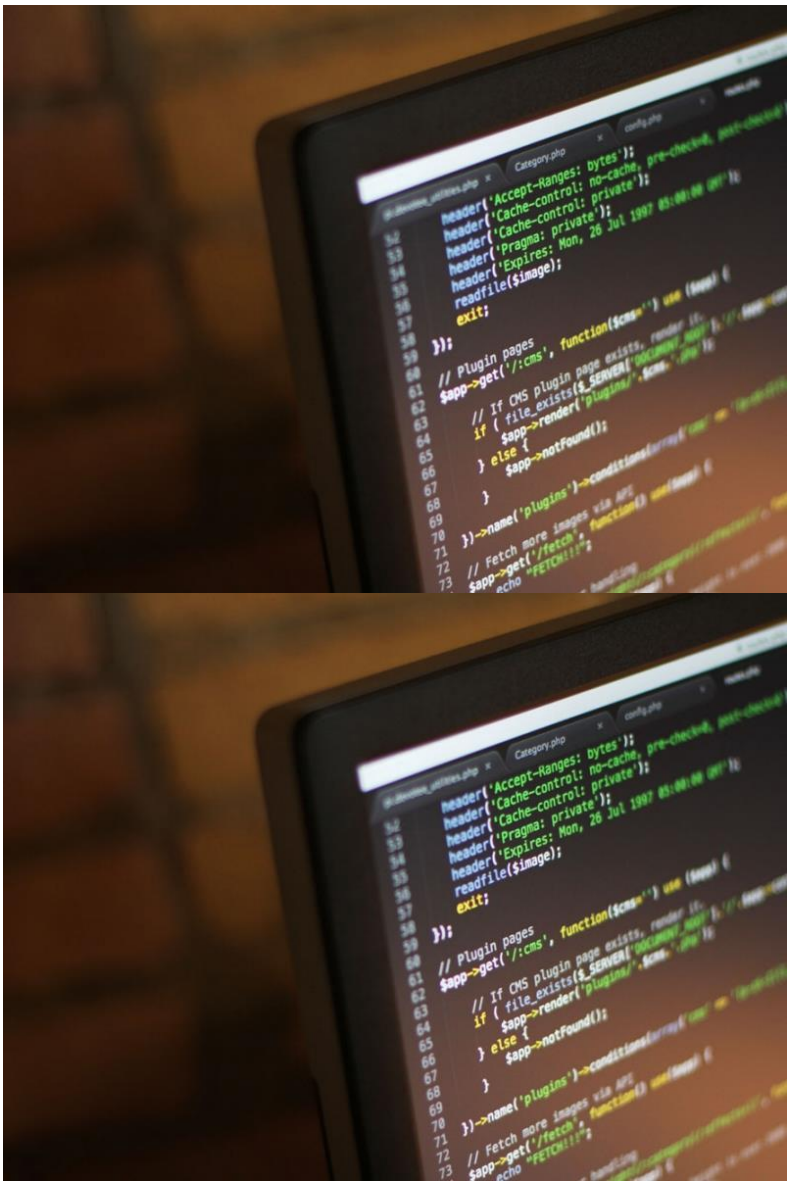


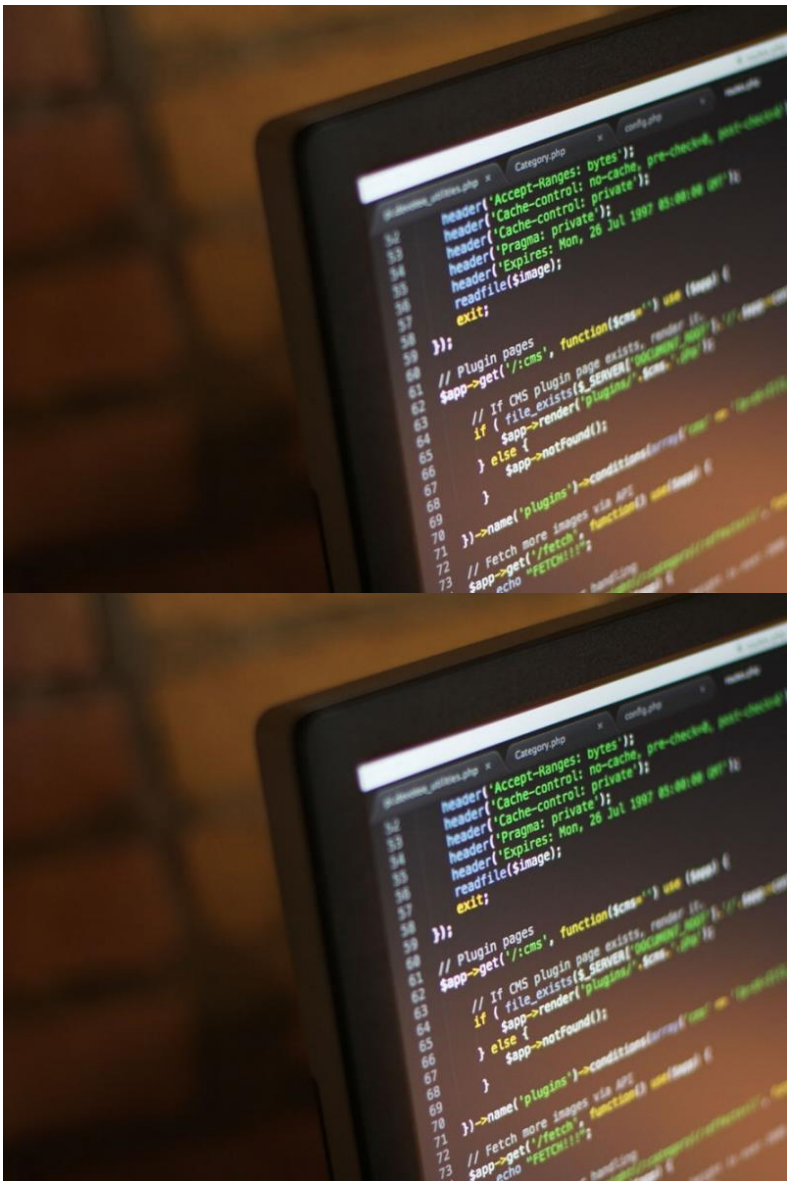


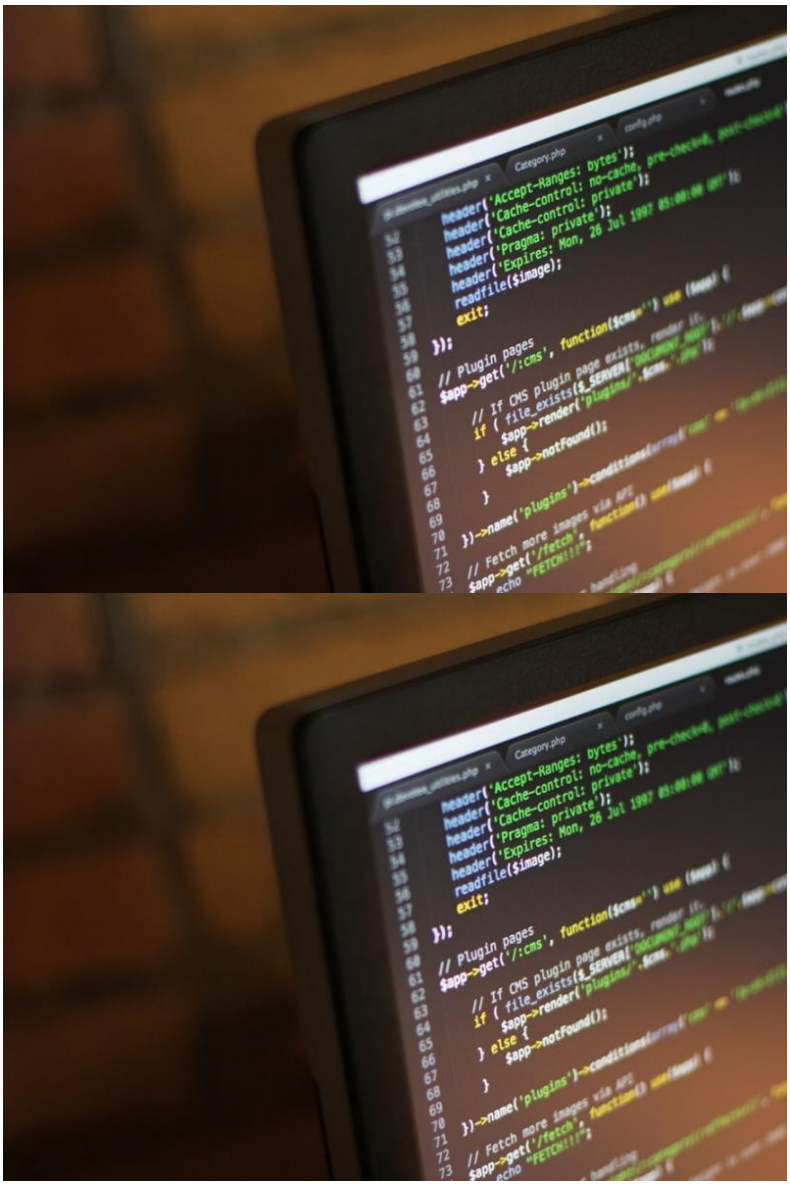


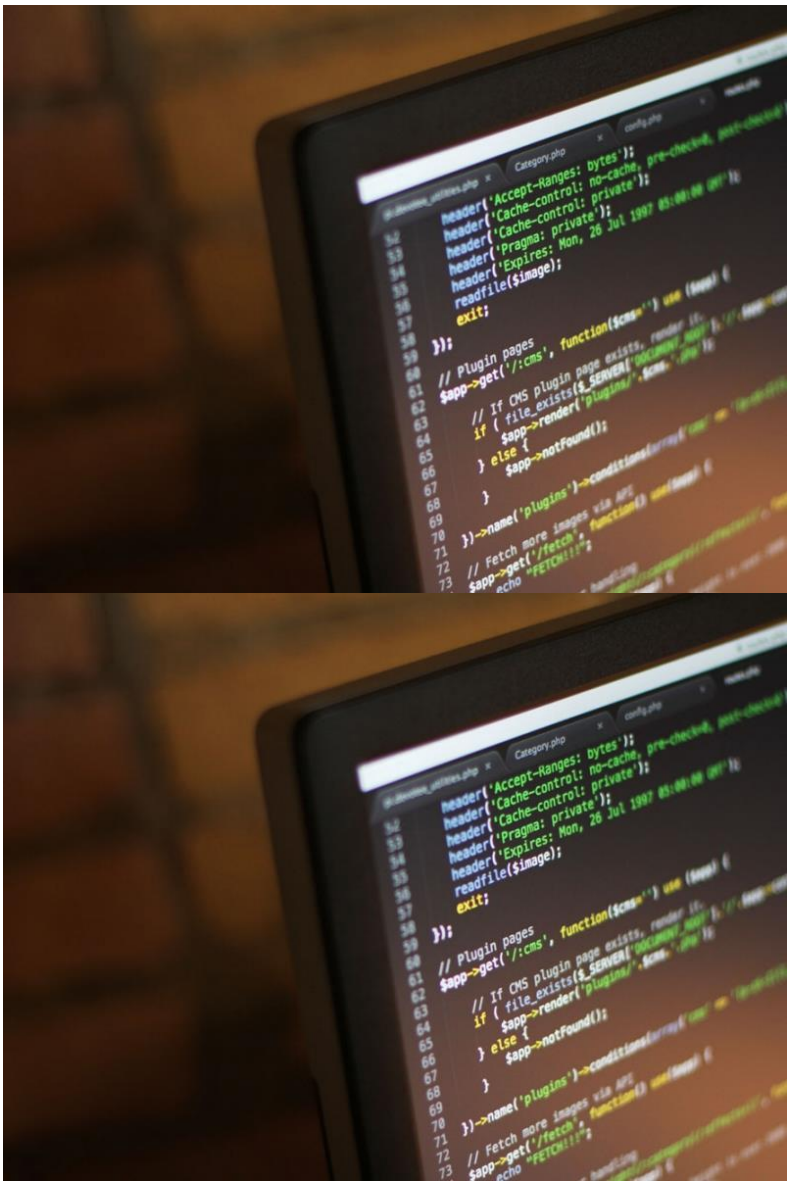


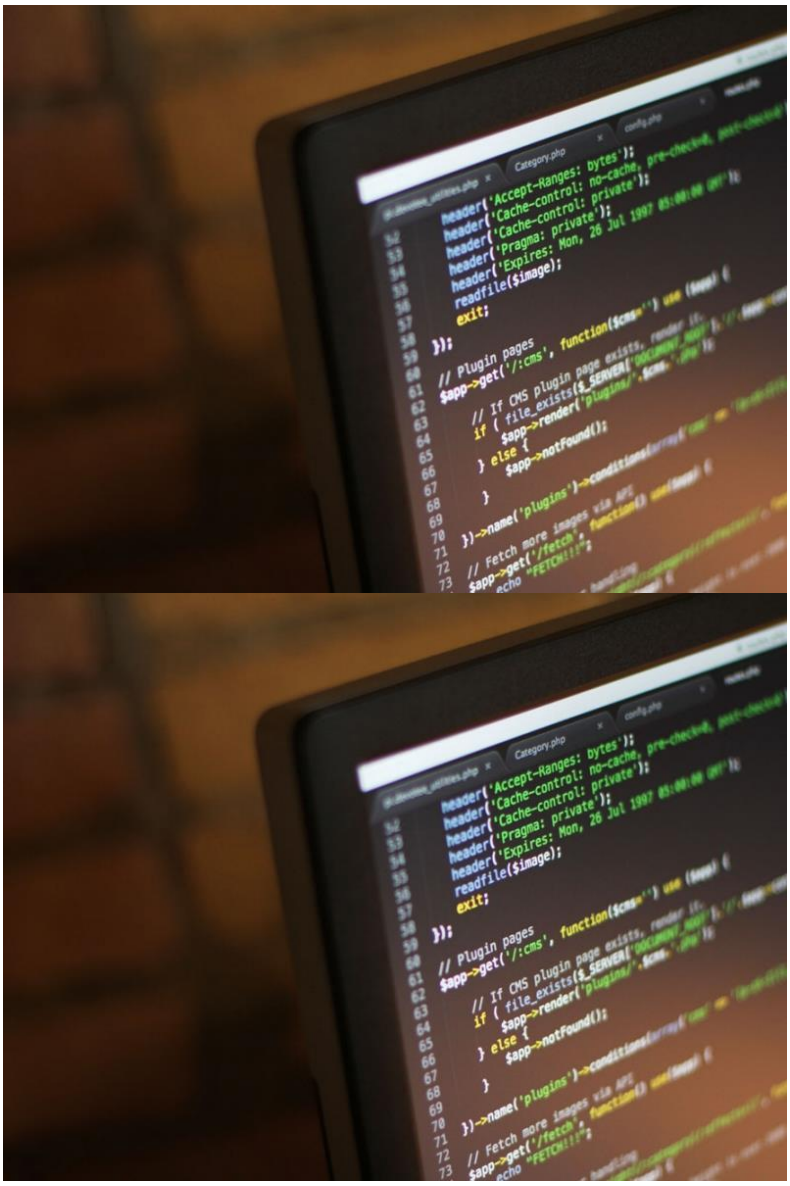


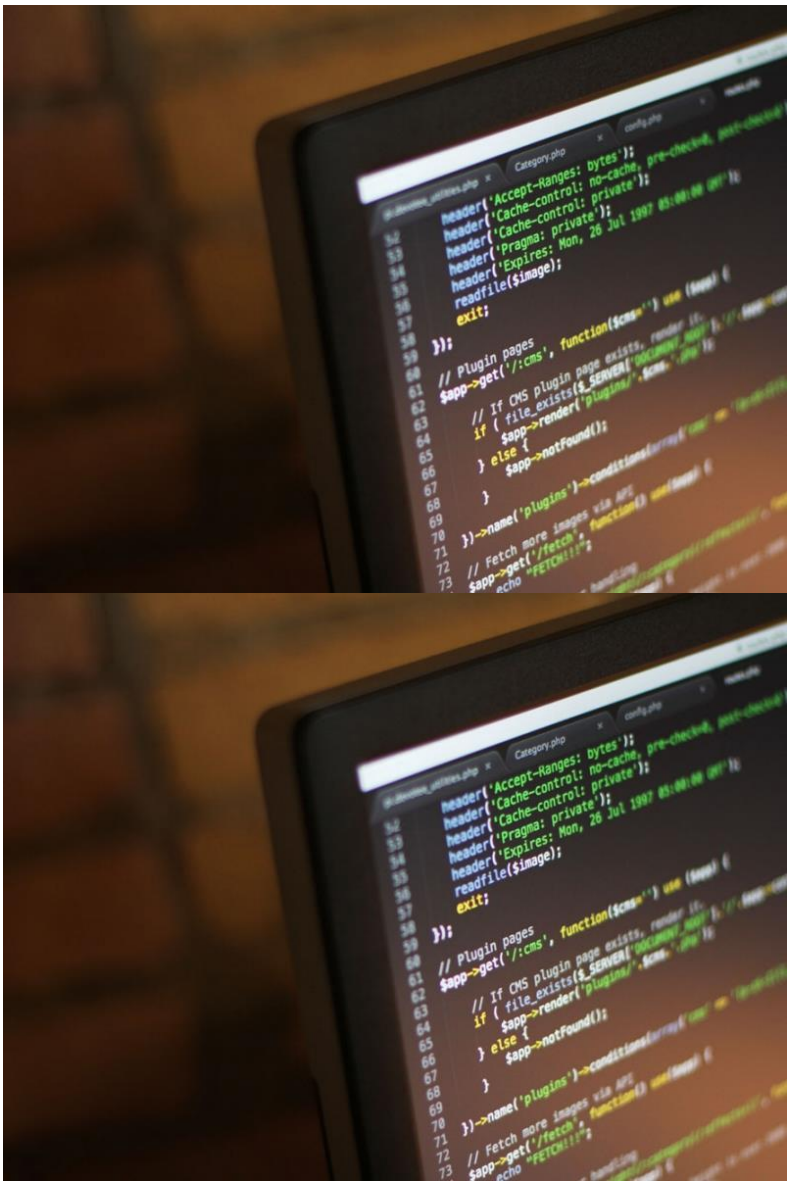


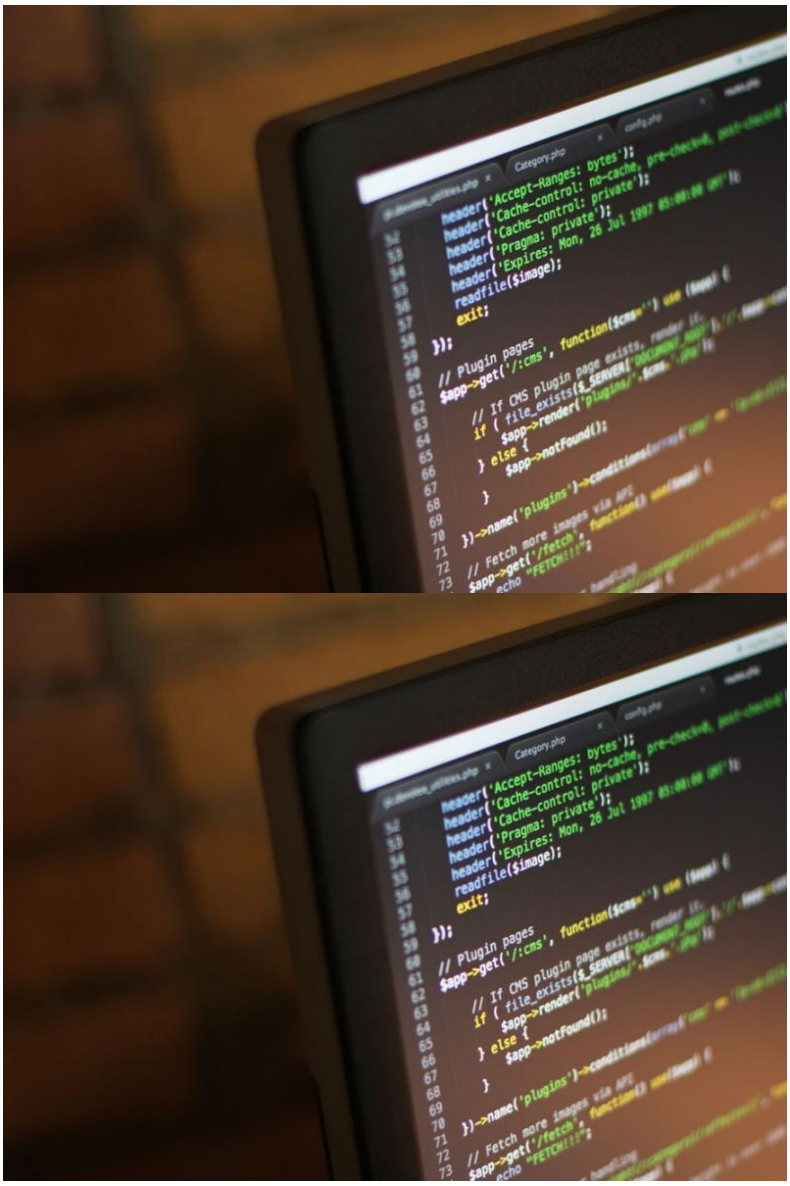


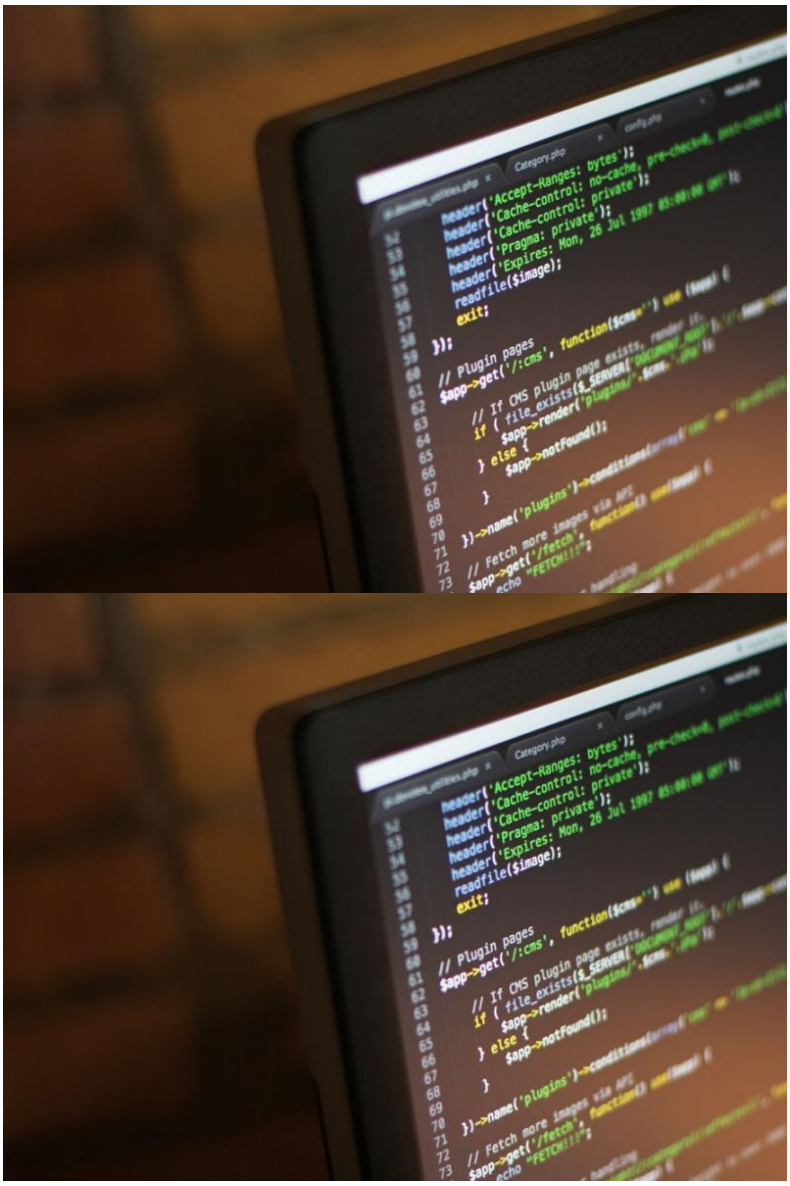


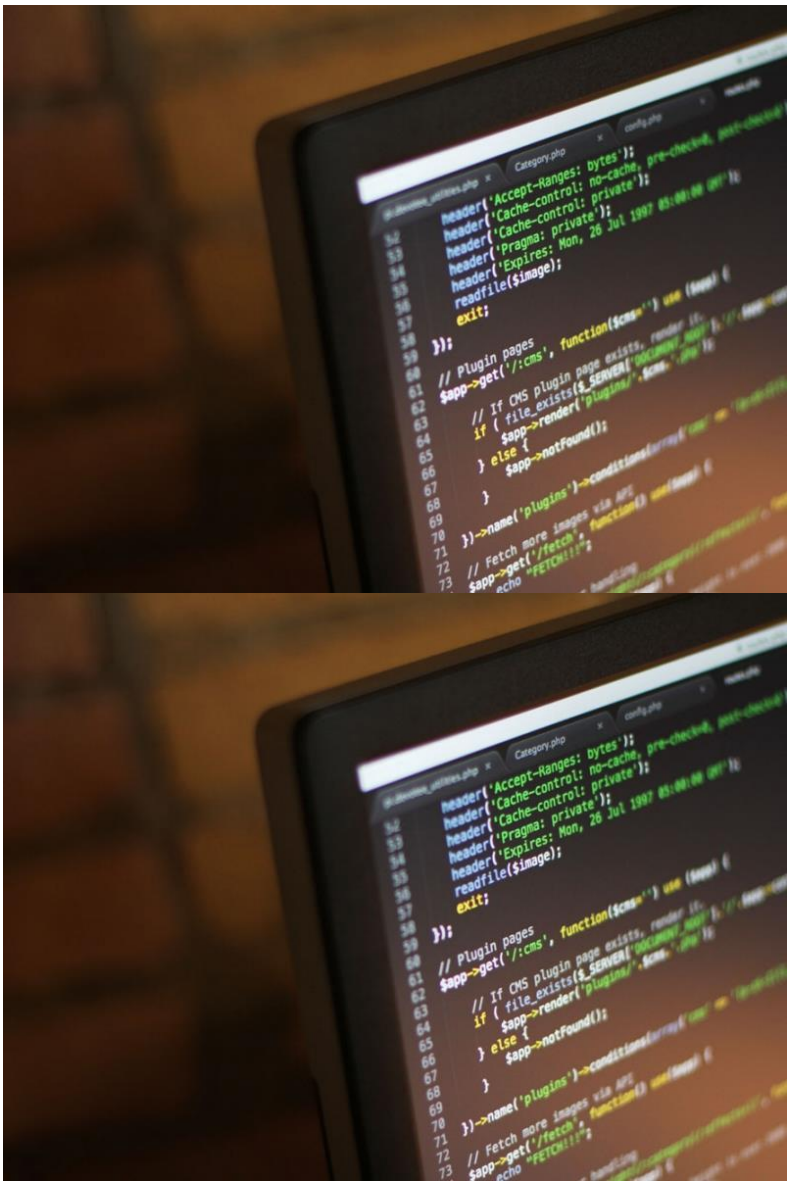


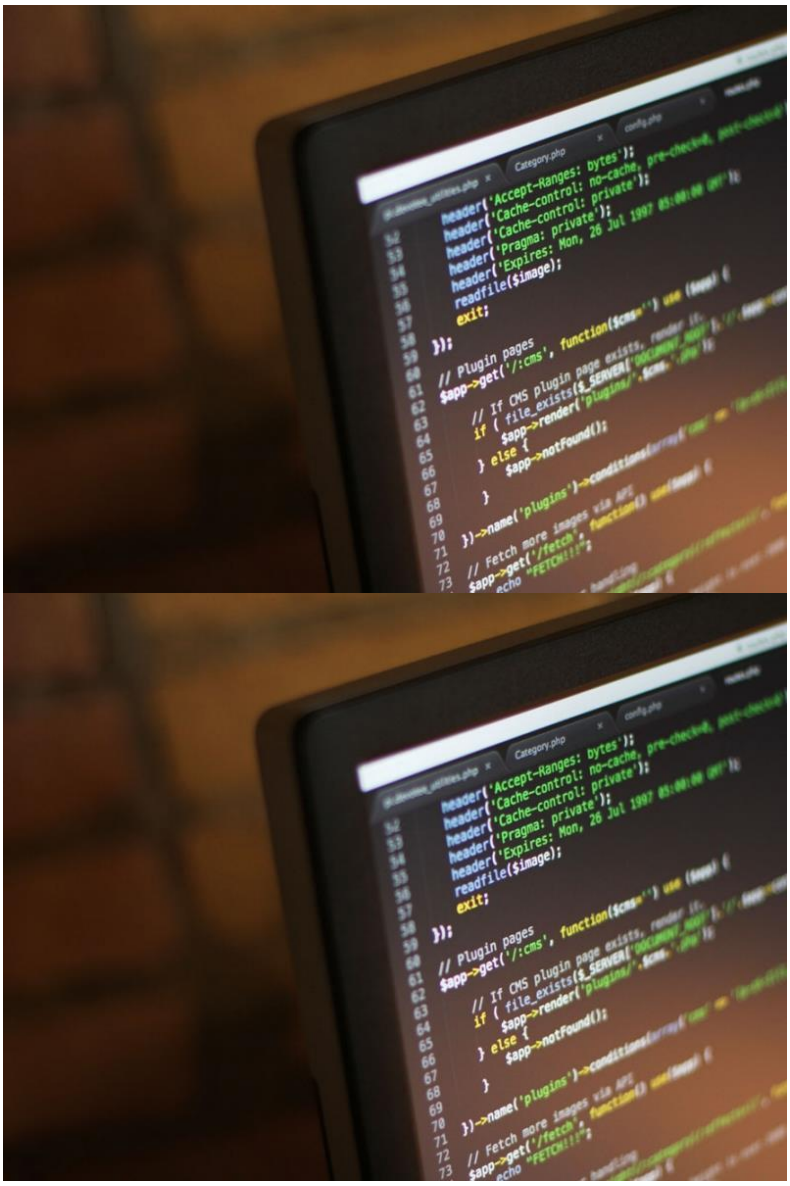


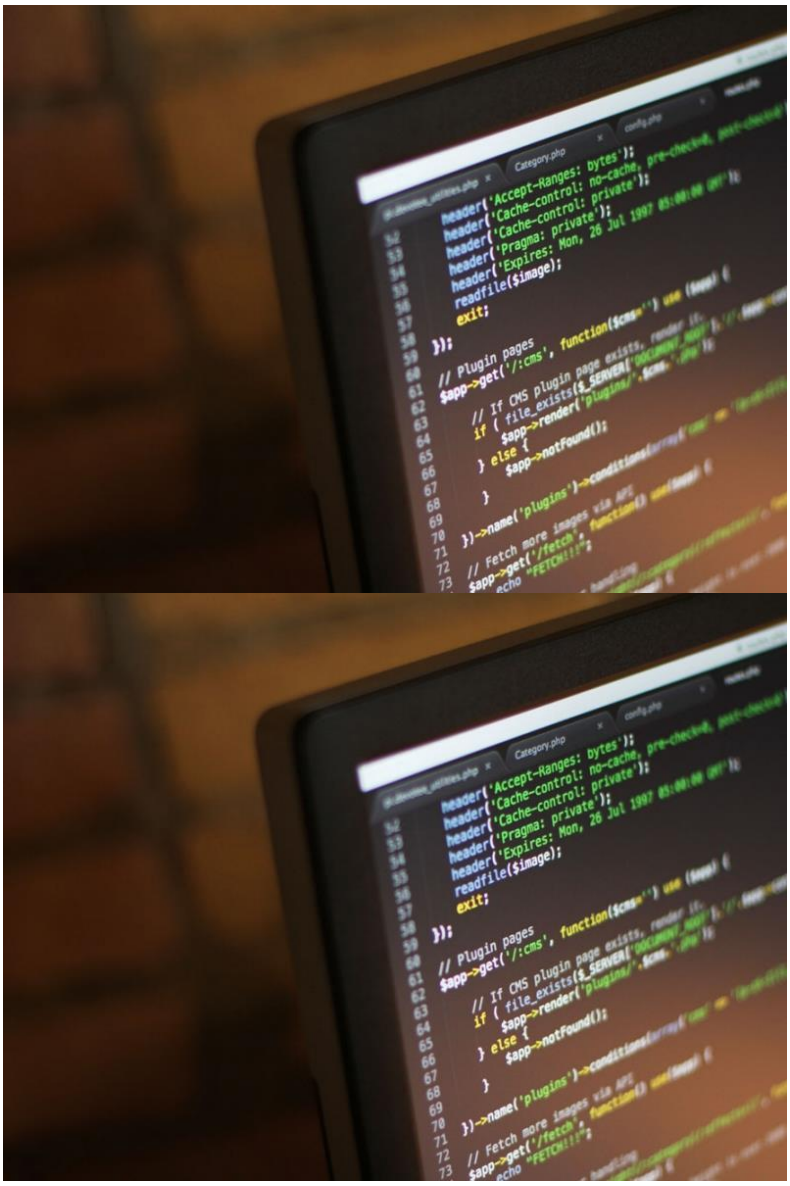


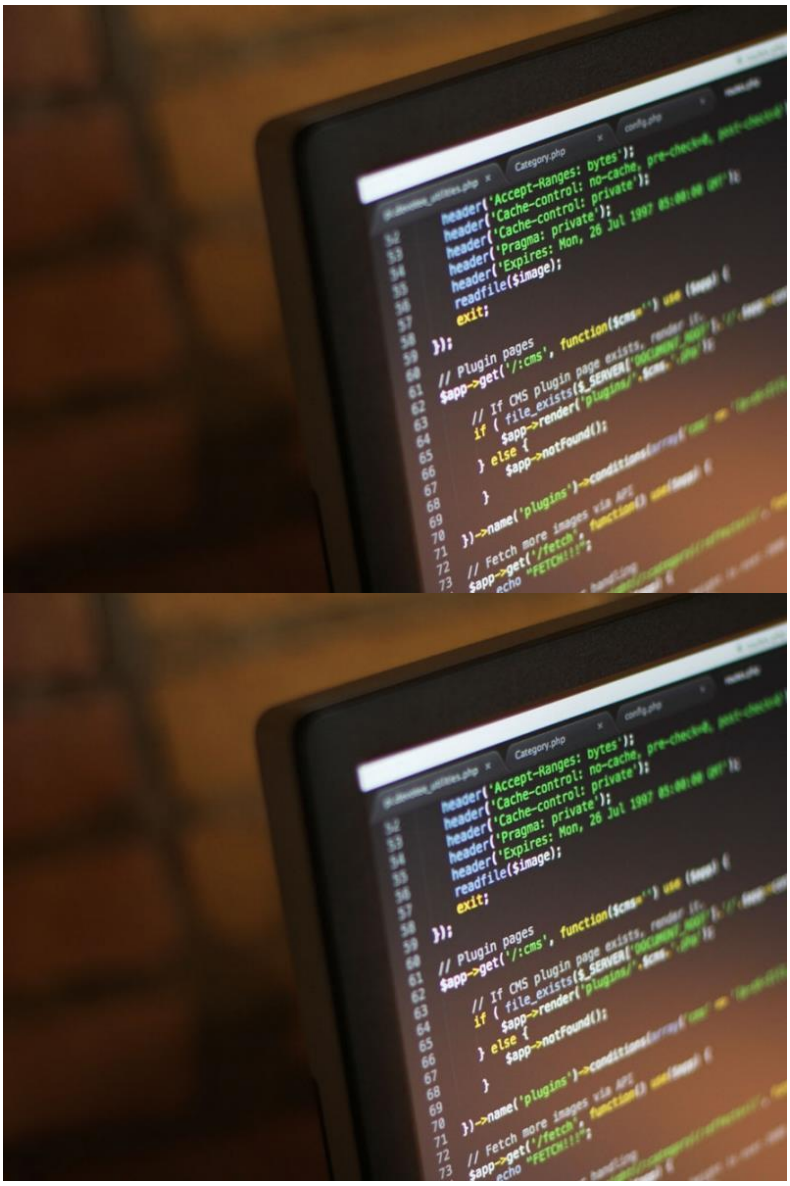


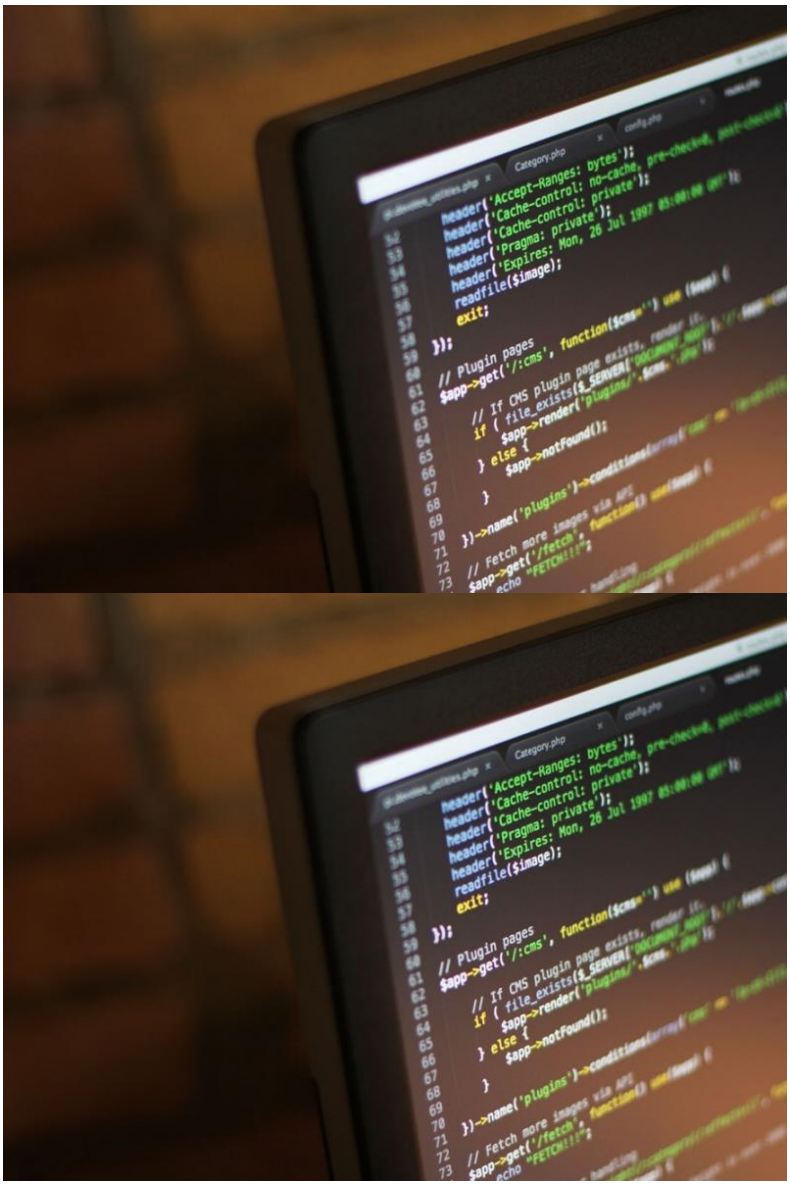


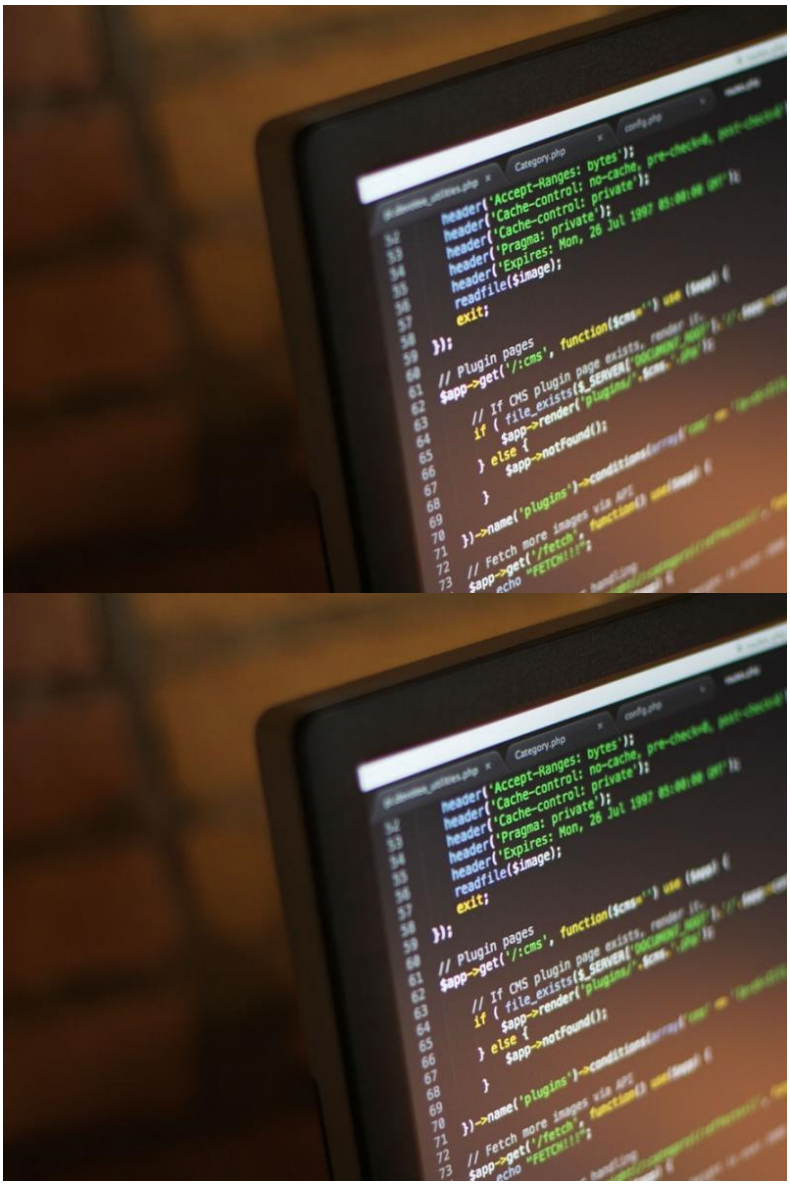


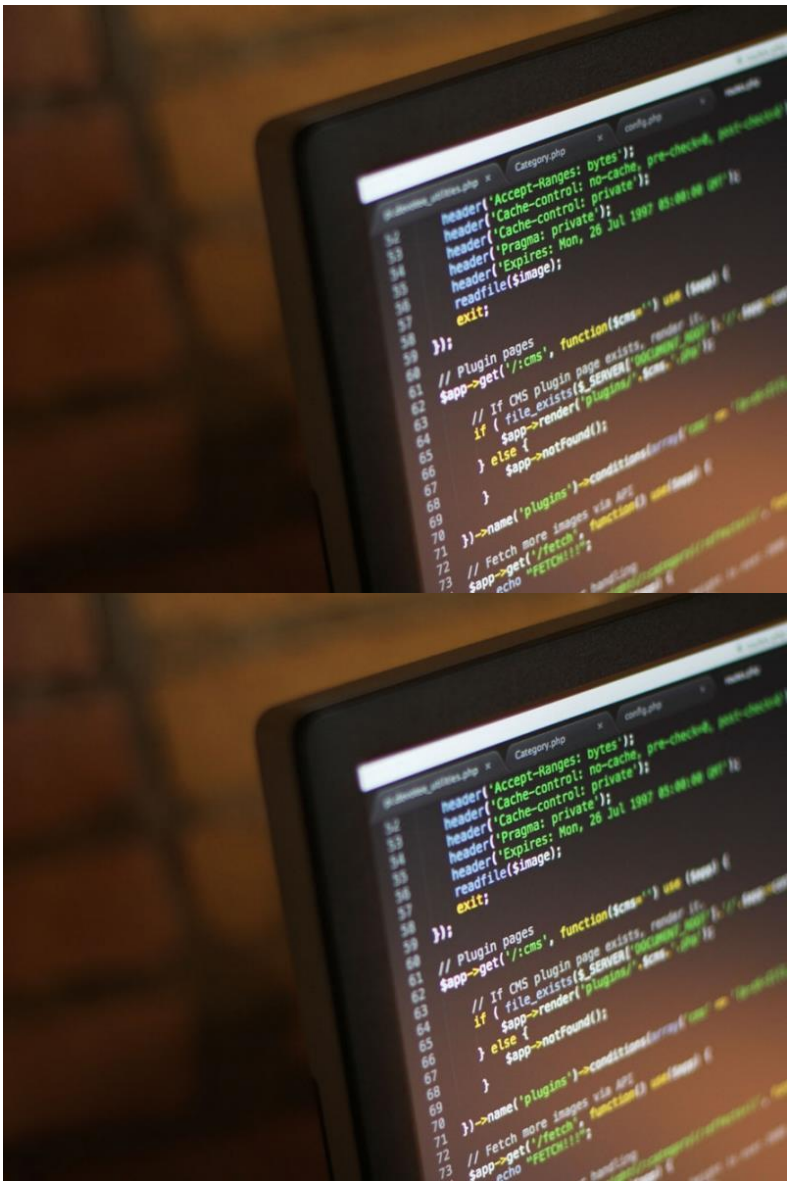


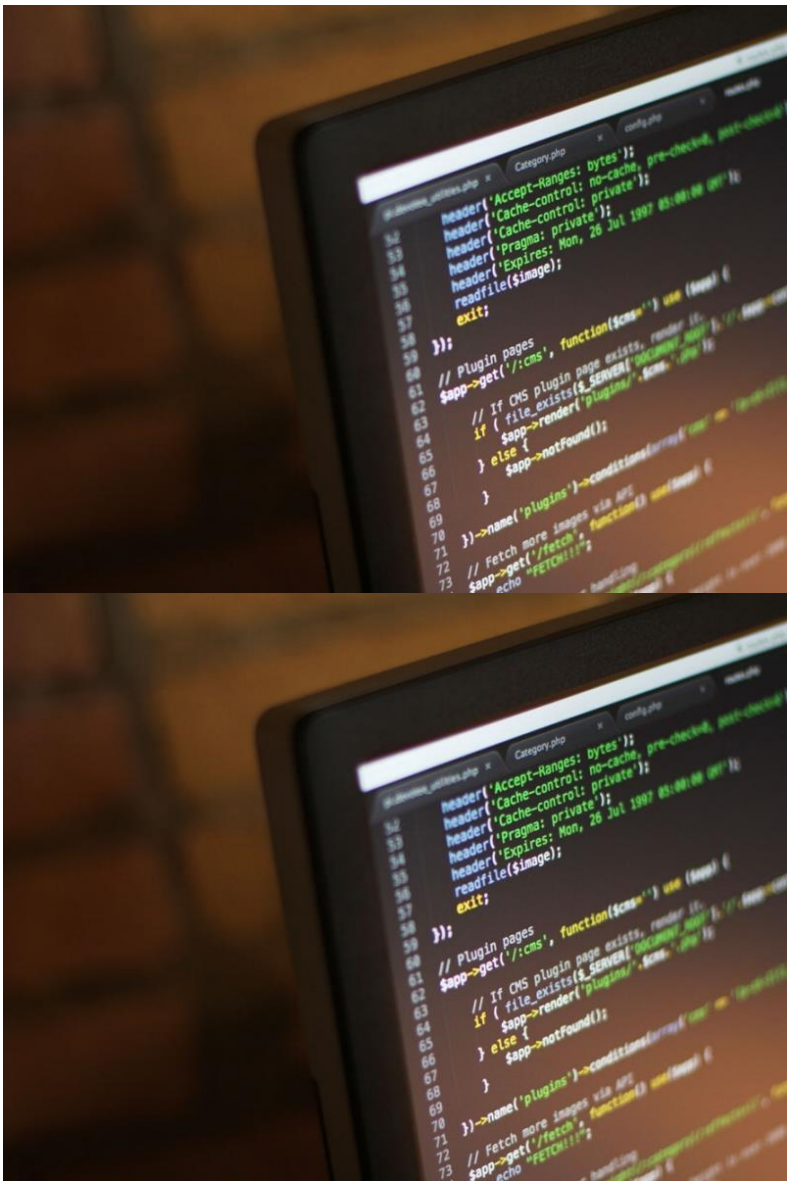


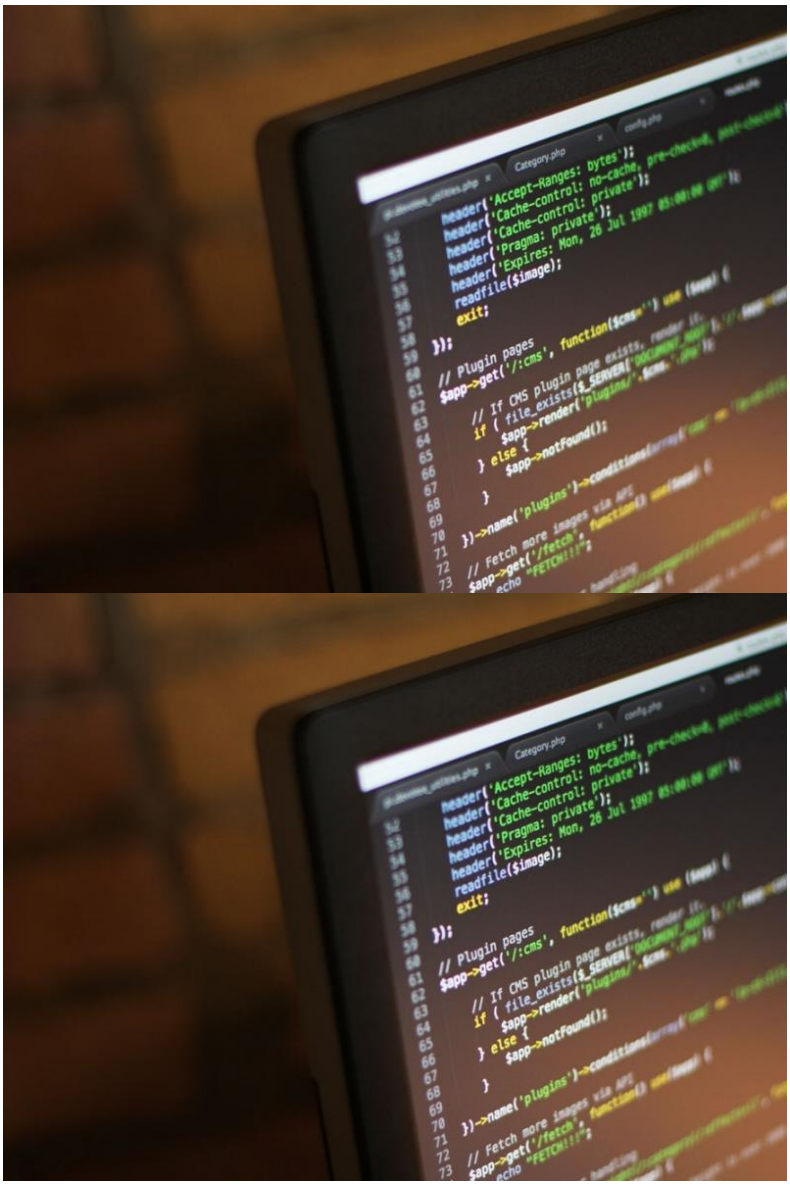


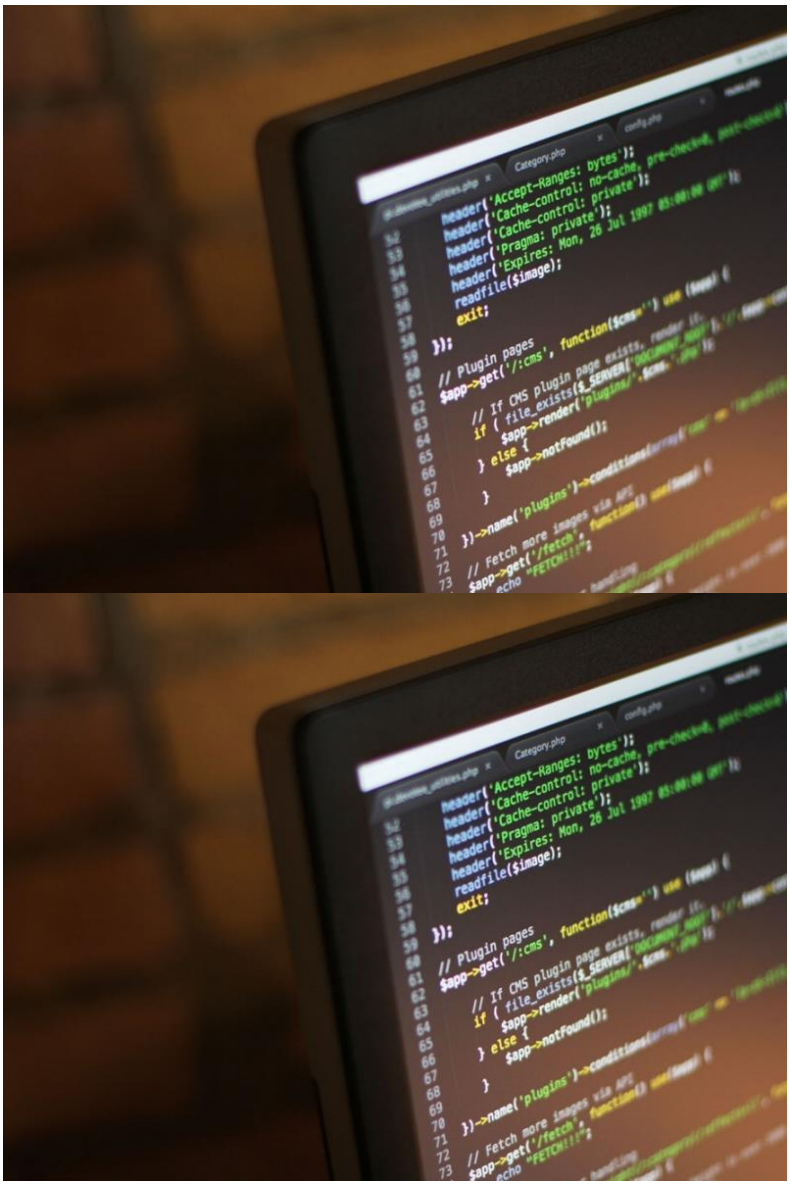


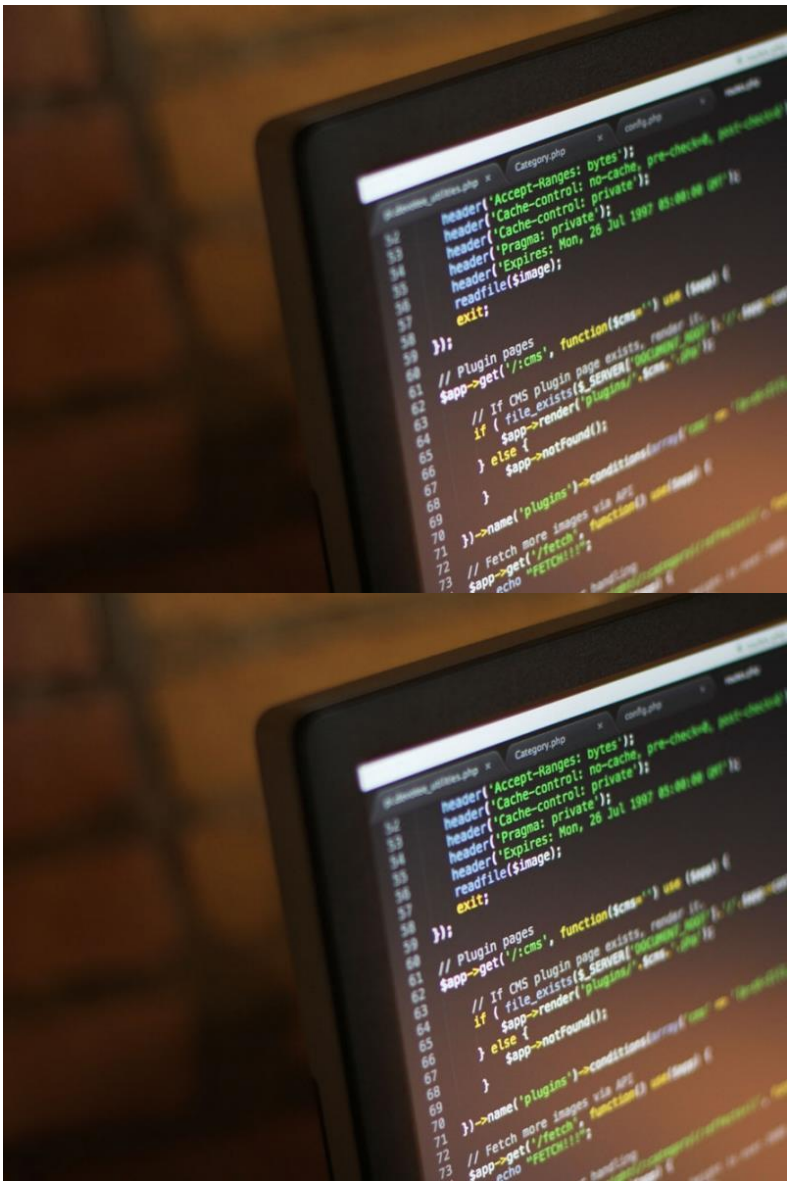


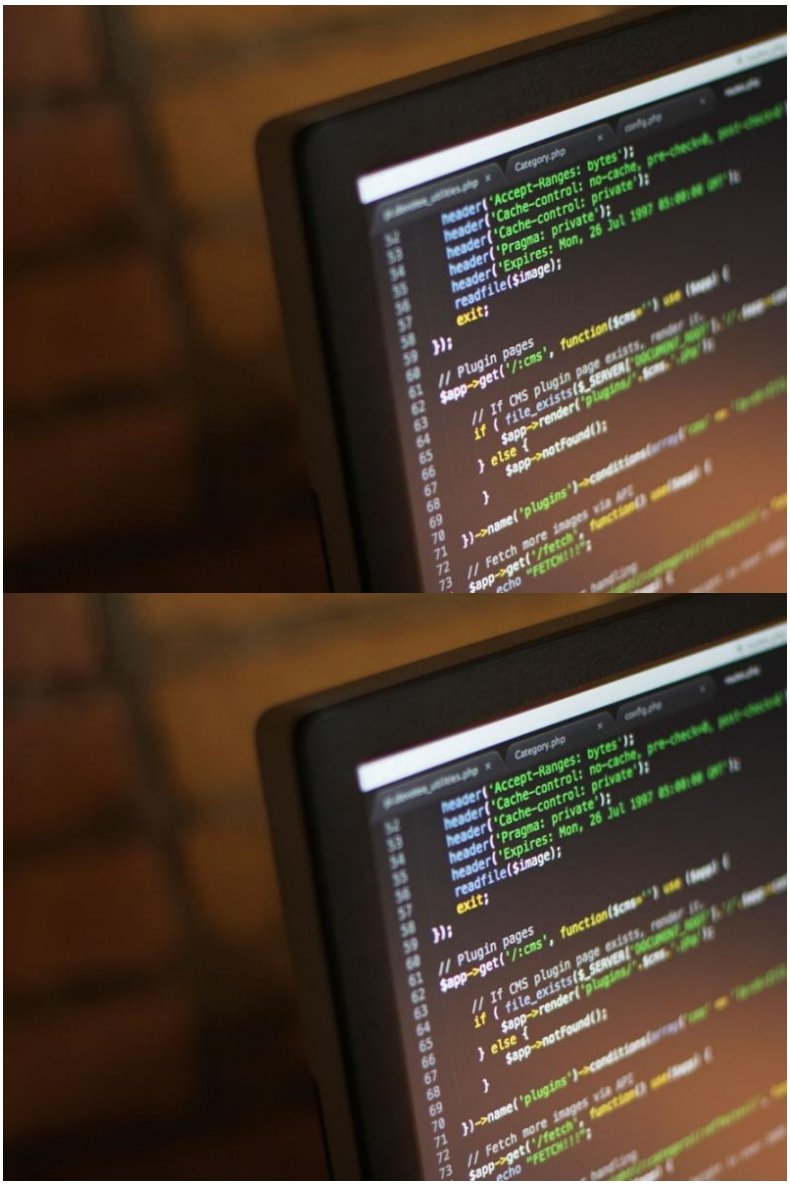


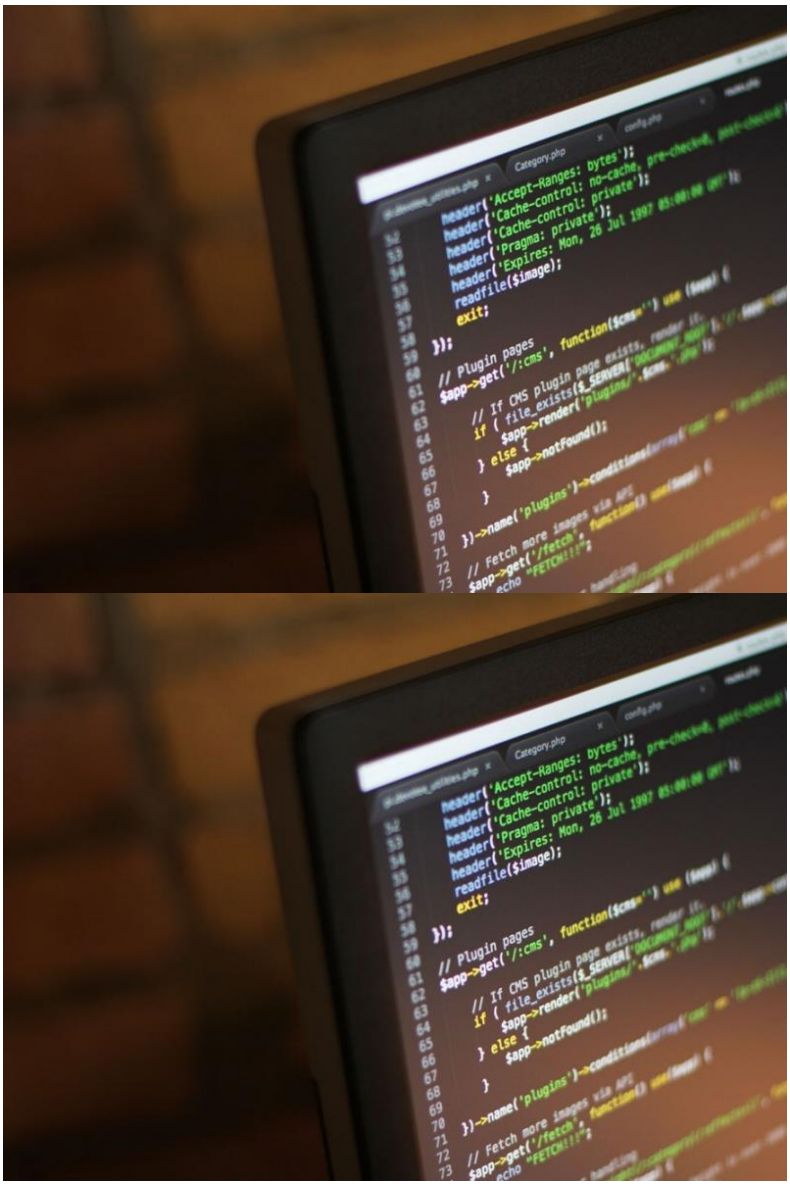


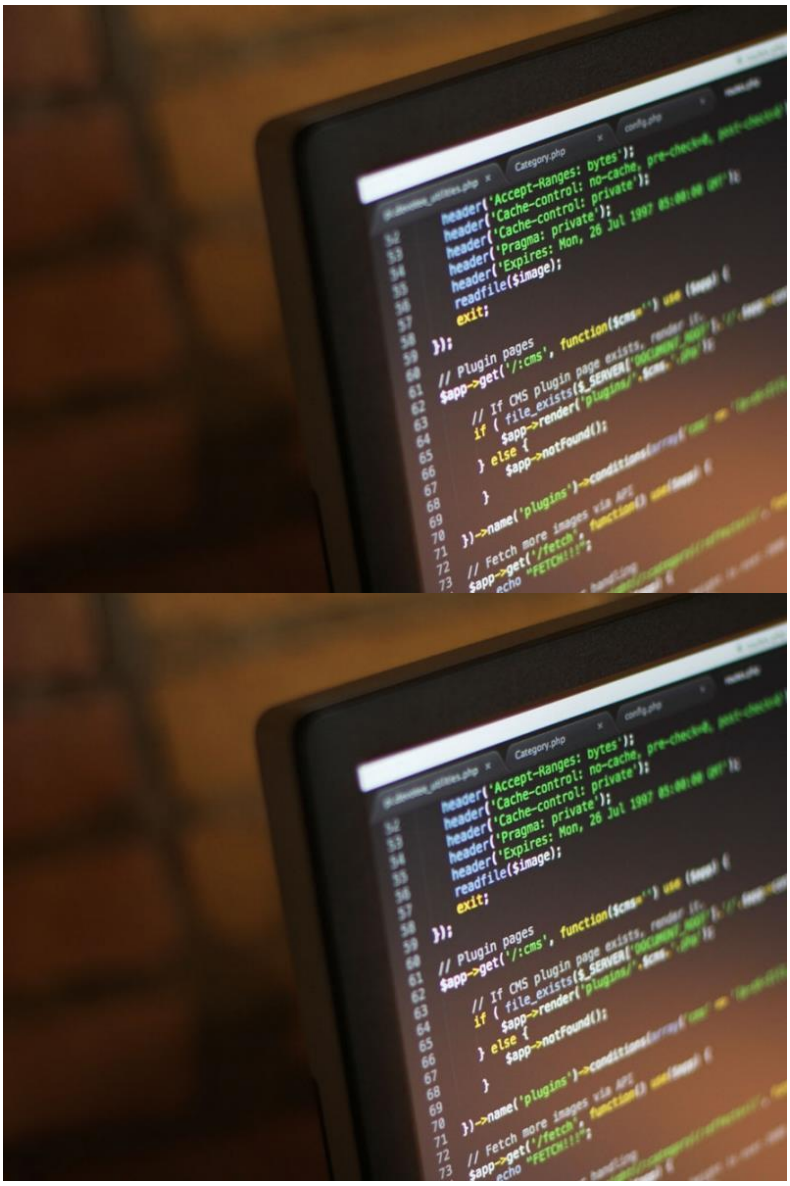


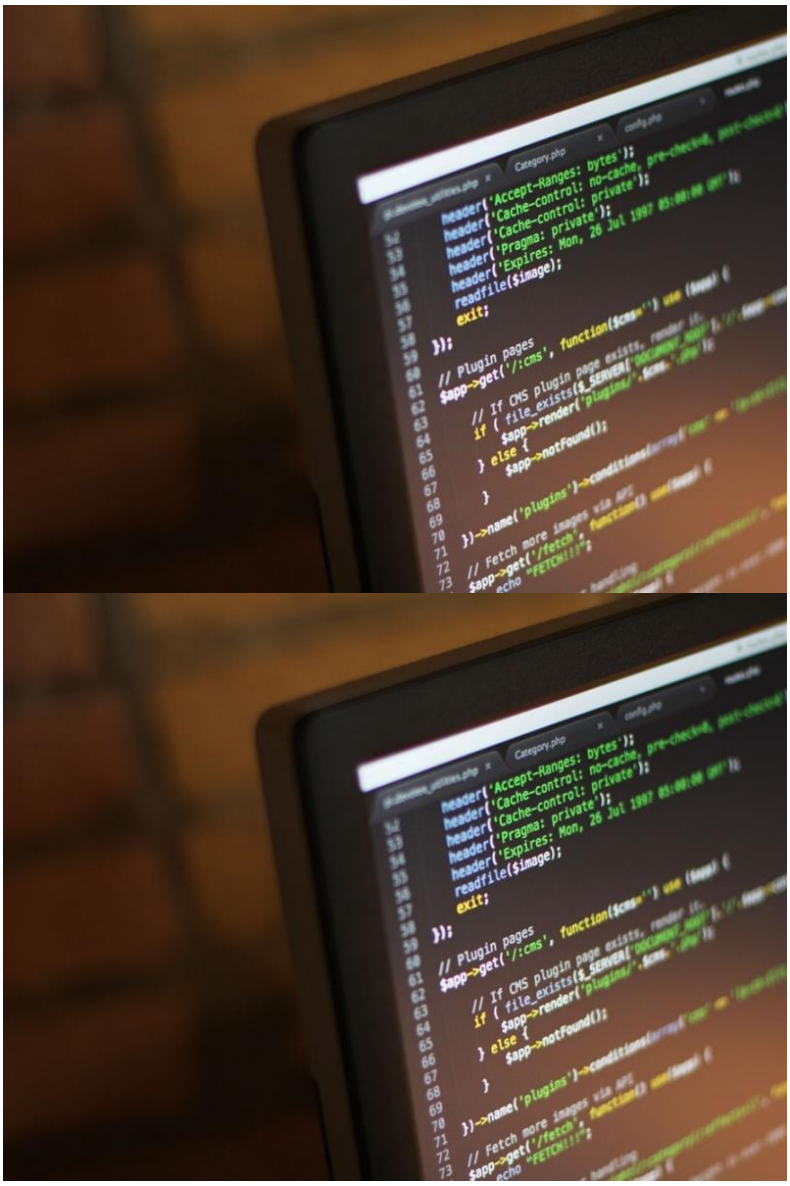


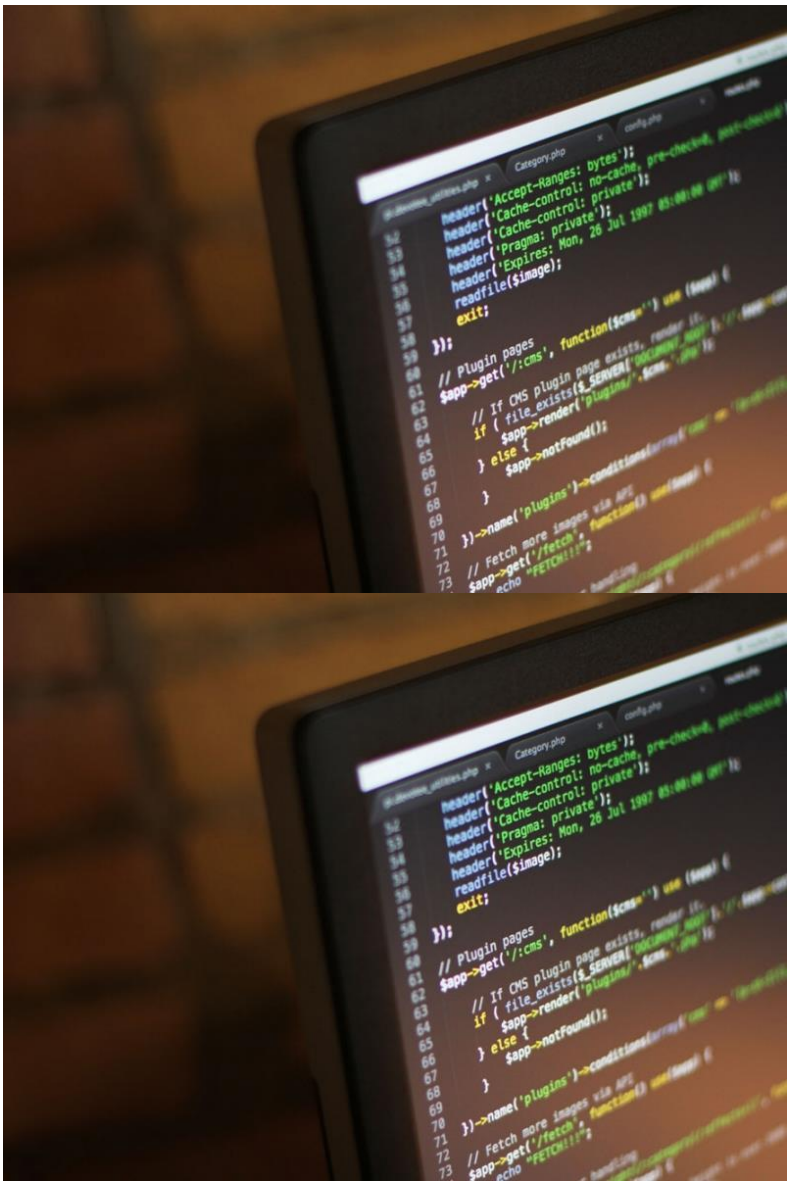












5 Выводы

В заключение хочется отметить, что данная лабораторная работа позволила мне научиться работать с системой Git. Я практиковала свои навыки в работе с командной строкой, теперь уже связывая выполнимое с директориями GitHub.

Кроме этого, я научилась работать с Markdown (вспмонила, опираясь на материалы из прошлого семестра в том числе)

Список литературы