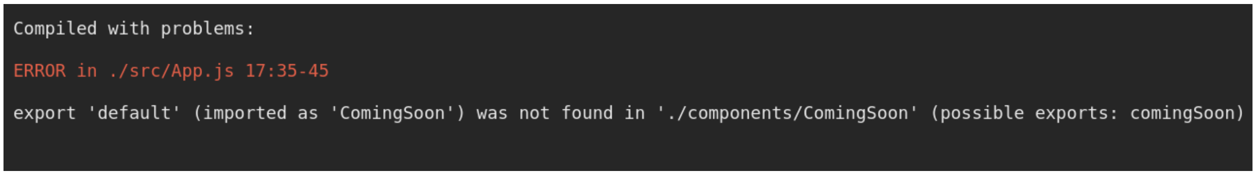
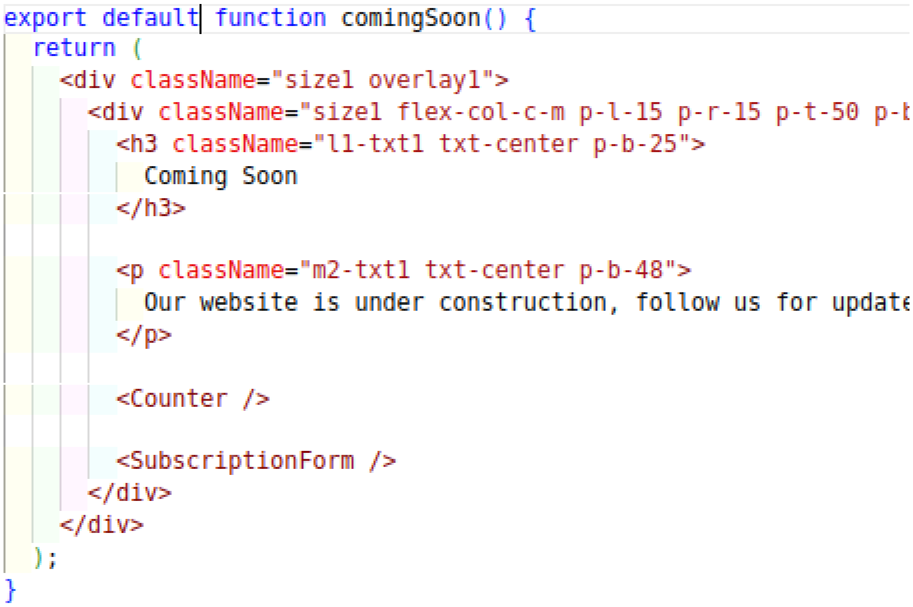
Problem 1 :



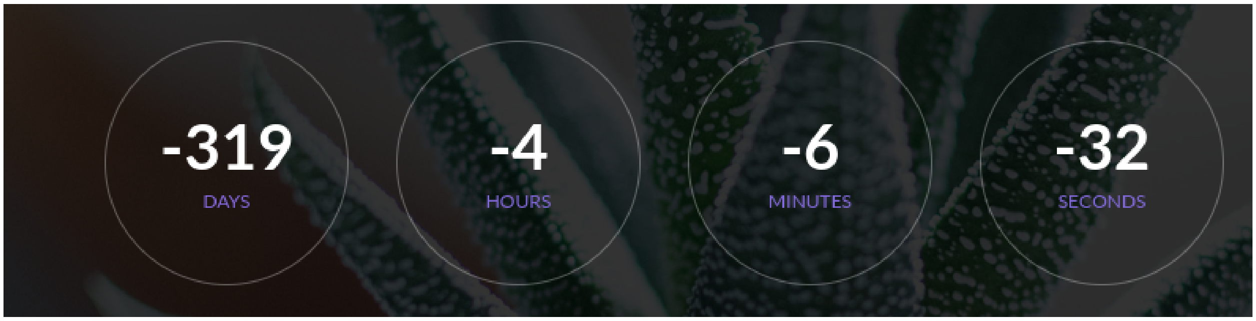
```
Compiled with problems:

ERROR in ./src/App.js 17:35-45

export 'default' (imported as 'ComingSoon') was not found in './components/ComingSoon' (possible exports: comingSoon)
```

Solution 1: use **default** statement in commingSoon functional component.

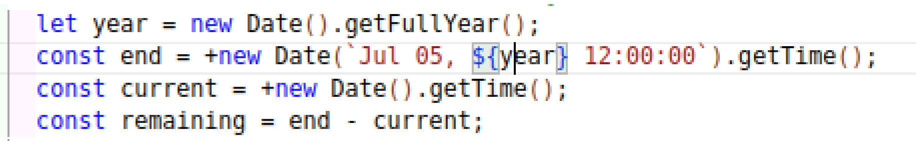


```
export default function comingSoon() {
  return (
    <div className="size1 overlay1">
      <div className="size1 flex-col-c-m p-l-15 p-r-15 p-t-50 p-b
        <h3 className="l1-txt1 txt-center p-b-25">
          Coming Soon
        </h3>

        <p className="m2-txt1 txt-center p-b-48">
          Our website is under construction, follow us for update
        </p>

        <Counter />

        <SubscriptionForm />
      </div>
    </div>
  );
}
```

Problem 2: counter not properly working and expired frontend not implemented.

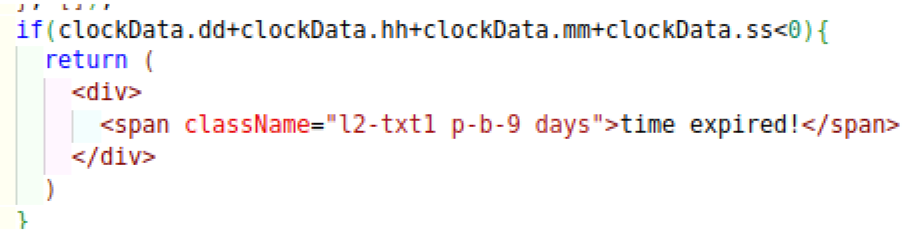


Solution 2: change the end date implementation.



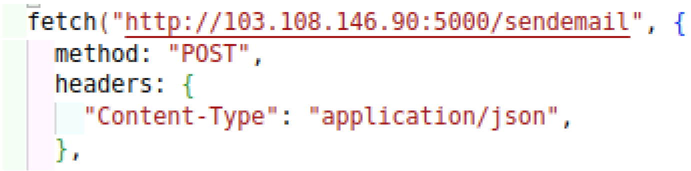
```
let year = new Date().getFullYear();
const end = +new Date(`Jul 05, ${year} 12:00:00`).getTime();
const current = +new Date().getTime();
const remaining = end - current;
```

and add an expiration time implementation.



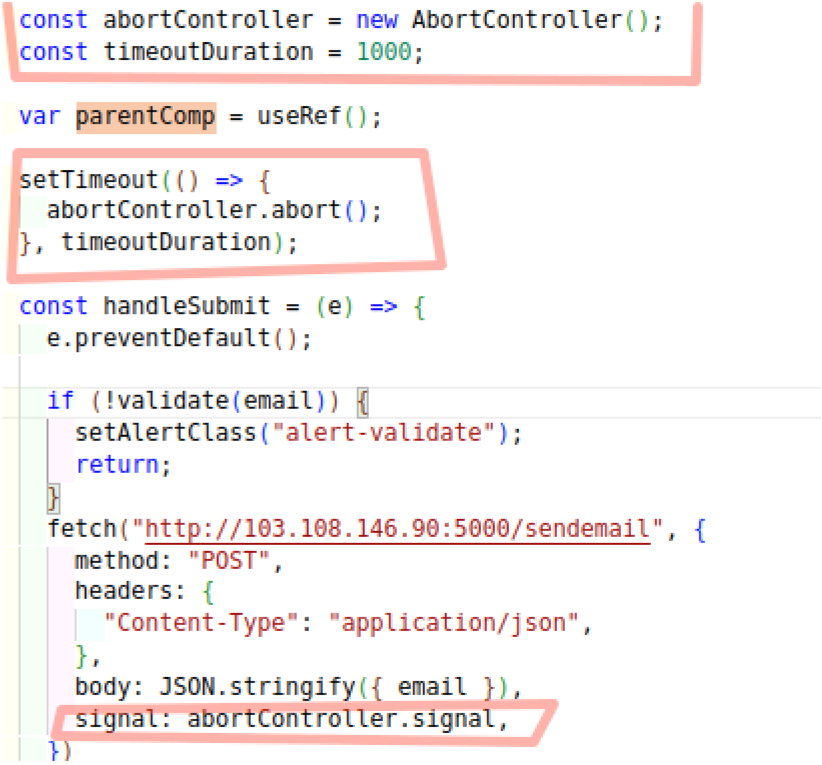
```
if(clockData.dd+clockData.hh+clockData.mm+clockData.ss<0){
  return (
    <div>
      <span className="l2-txt1 p-b-9 days">time expired!</span>
    </div>
  )
}
```

Problem 3: Post api call doesn’t reach.



```
fetch("http://103.108.146.90:5000/sendemail", {
  method: "POST",
  headers: {
    "Content-Type": "application/json",
  },
```

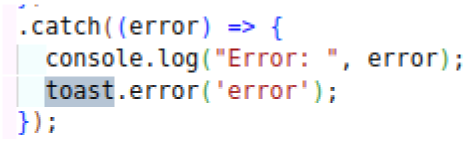
Solution 3 : create an abortController for aborting the fetch request signal. And use the setTimeOut function to trigger the abort method.



```
const abortController = new AbortController();
const timeoutDuration = 1000;

var parentComp = useRef();

setTimeout(() => {
  abortController.abort();
}, timeoutDuration);

const handleSubmit = (e) => {
  e.preventDefault();

  if (!validate(email)) {
    setAlertClass("alert-validate");
    return;
  }
  fetch("http://103.108.146.90:5000/sendemail", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify({ email }),
    signal: abortController.signal,
  })
```

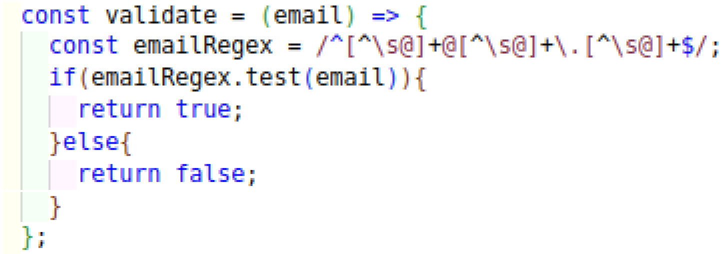
Problem 4 : toast notification does not work properly for api request.

Solution 4 : directly use the toast function from toast-notify library.



```
.catch((error) => {
  console.log("Error: ", error);
  toast.error('error');
});
```

Problem 5 : email field does not properly validate.
Solution 5 : use proper regex for email validation.



```
const validate = (email) => {
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  if(emailRegex.test(email)){
    return true;
  }else{
    return false;
  }
};
```