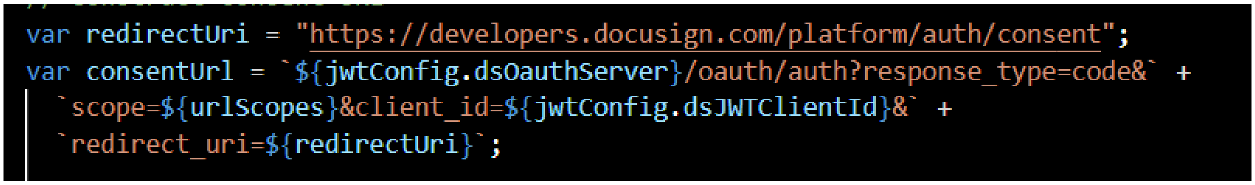
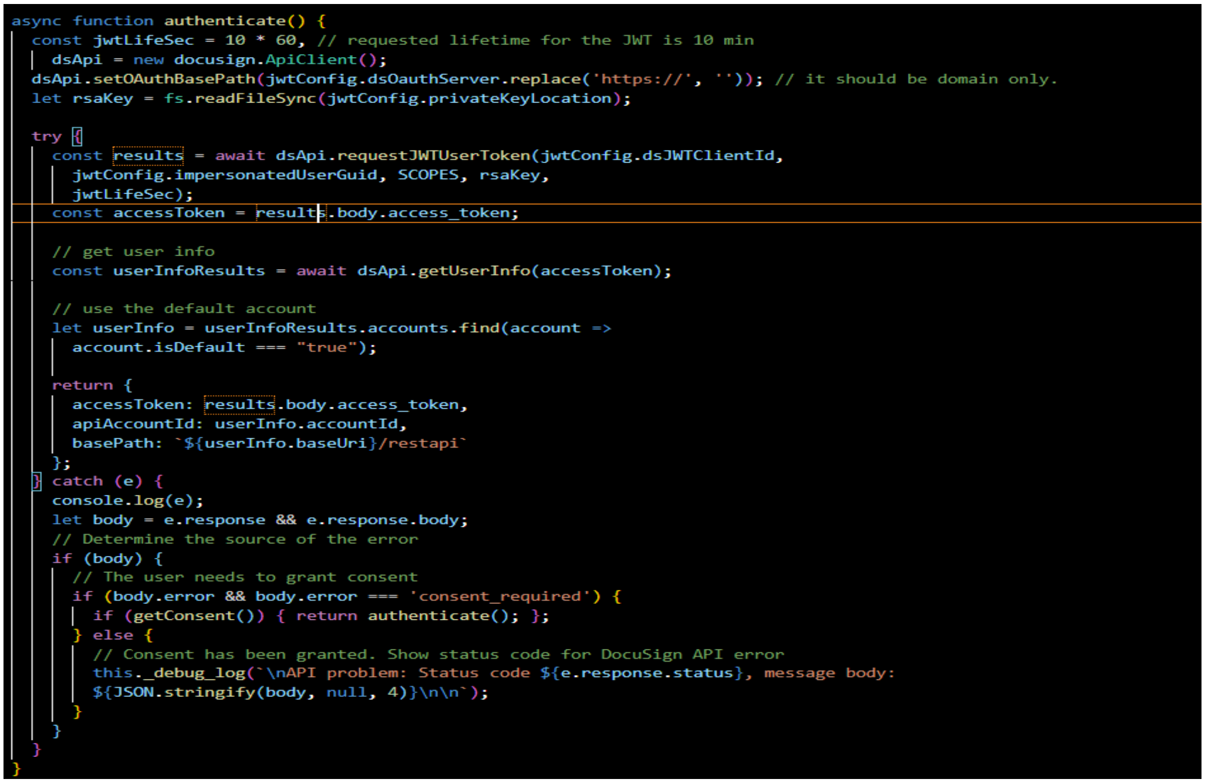
Here are the steps we took to integrate OAuth into our application:

1) **Registering the application**: We created a developer account with DocuSign and registered our application in the DocuSign Developer Center. This allowed us to obtain a client ID and a client secret, which are necessary for implementing the OAuth flow.
2) **Implementing the OAuth flow**: We implemented the OAuth 2.0 authorization flow in our application using one of the available OAuth libraries for our programming language. The OAuth flow consists of the following steps:
3) **Authorization grant**: Upon successful authentication, DocuSign returns an authorization code to our application's callback URL.
4) **Access token request**: We exchanged the authorization code for an access token by making a request to DocuSign's token endpoint using our client ID, client secret, and the authorization code.
5) **Accessing the API**: We used the access token to authenticate our requests to DocuSign's API. The access token has a limited lifespan and needs to be refreshed periodically.
6) **getConsent() method:** this method contain a redirect URL.



```
var redirectUri = "https://developers.docusign.com/platform/auth/consent";
var consentUrl = `${jwtConfig.dsOauthServer}/oauth/auth?response_type=code&` +
  `scope=${urlScopes}&client_id=${jwtConfig.dsJWTClientId}&` +
  `redirect_uri=${redirectUri}`;
```

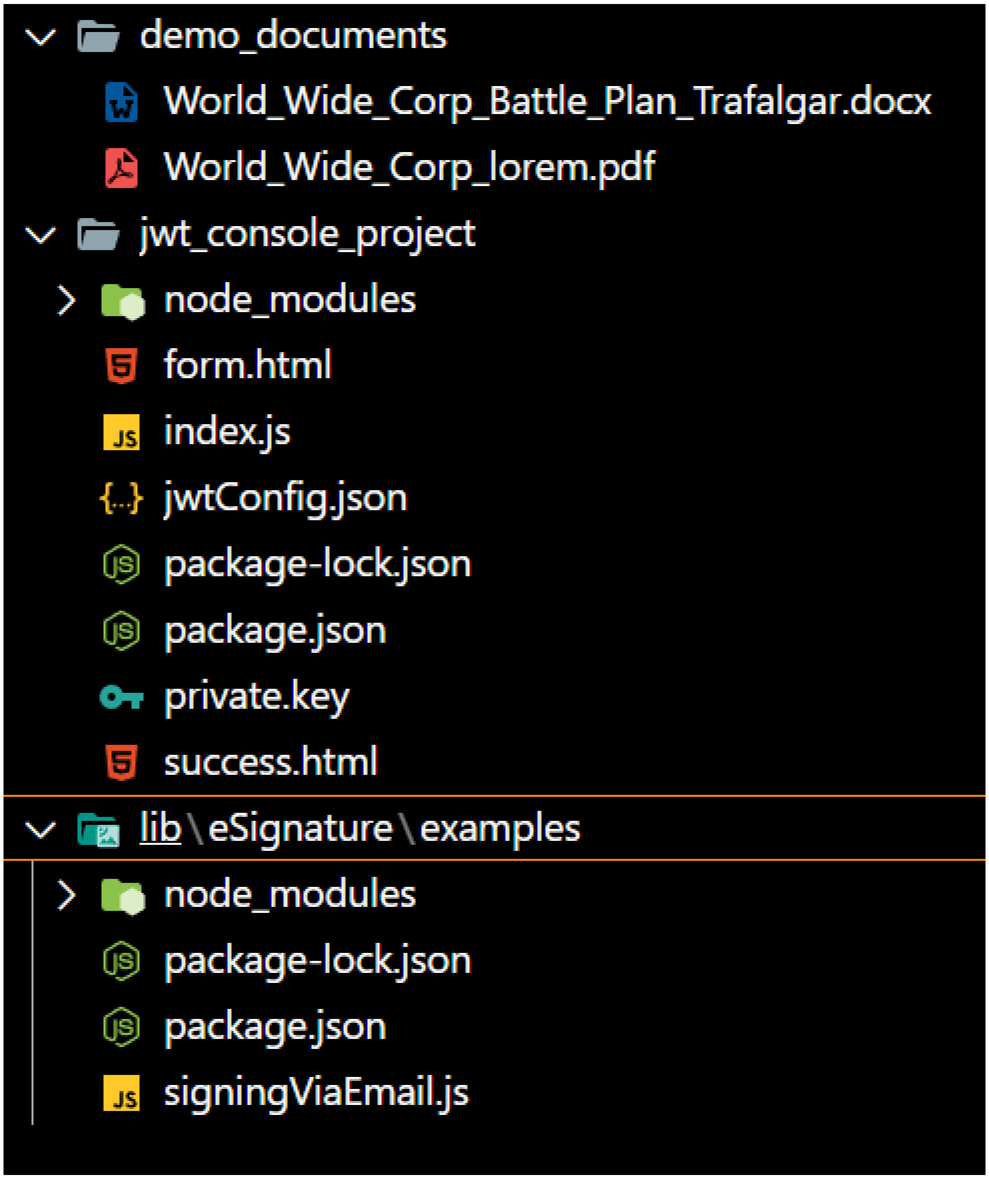
7) **authenticate() method :** This method is use to authenticate the user using their secret keys.



```
async function authenticate() {
  const jwtLifeSec = 10 * 60, // requested lifetime for the JWT is 10 min
    dsApi = new docusign.ApiClient();
  dsApi.setOAuthBasePath(jwtConfig.dsOauthServer.replace('https://', '')); // it should be domain only.
  let rsaKey = fs.readFileSync(jwtConfig.privateKeyLocation);

  try {
    const results = await dsApi.requestJWTUserToken(jwtConfig.dsJWTClientId,
      jwtConfig.impersonatedUserGuid, SCOPES, rsaKey,
      jwtLifeSec);
    const accessToken = results.body.access_token;

    // get user info
    const userInfoResults = await dsApi.getUserInfo(accessToken);

    // use the default account
    let userInfo = userInfoResults.accounts.find(account =>
      account.isDefault === "true");

    return {
      accessToken: results.body.access_token,
      apiAccountId: userInfo.accountId,
      basePath: `${userInfo.baseUri}/restapi`
    };
  } catch (e) {
    console.log(e);
    let body = e.response && e.response.body;
    // Determine the source of the error
    if (body) {
      // The user needs to grant consent
      if (body.error && body.error === 'consent_required') {
        if (getConsent()) { return authenticate(); };
      } else {
        // Consent has been granted. Show status code for DocuSign API error
        this._debug_log(`\nAPI problem: Status code ${e.response.status}, message body:
        ${JSON.stringify(body, null, 4)}\n\n`);
      }
    }
  }
}
```

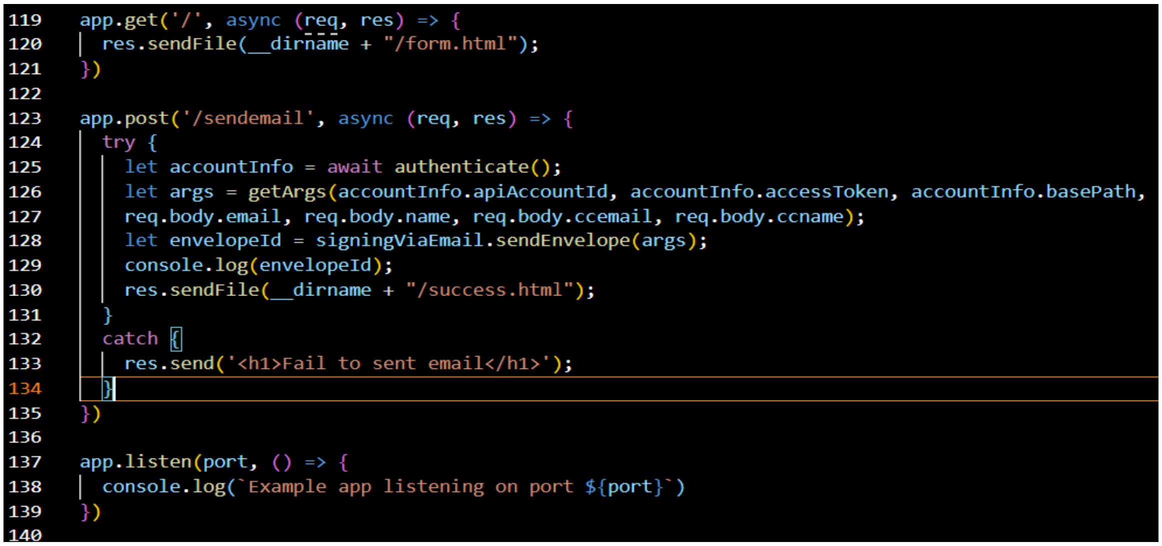
Here jwtLifeSec is used for a lifetime of JWT token.

8) **Folder structure:**



```
demo_documents
    World_Wide_Corp_Battle_Plan_Trafalgar.docx
    World_Wide_Corp_lorem.pdf
jwt_console_project
    node_modules
    form.html
    index.js
    jwtConfig.json
    package-lock.json
    package.json
    private.key
    success.html
lib\eSignature\examples
    node_modules
    package-lock.json
    package.json
    signingViaEmail.js
```

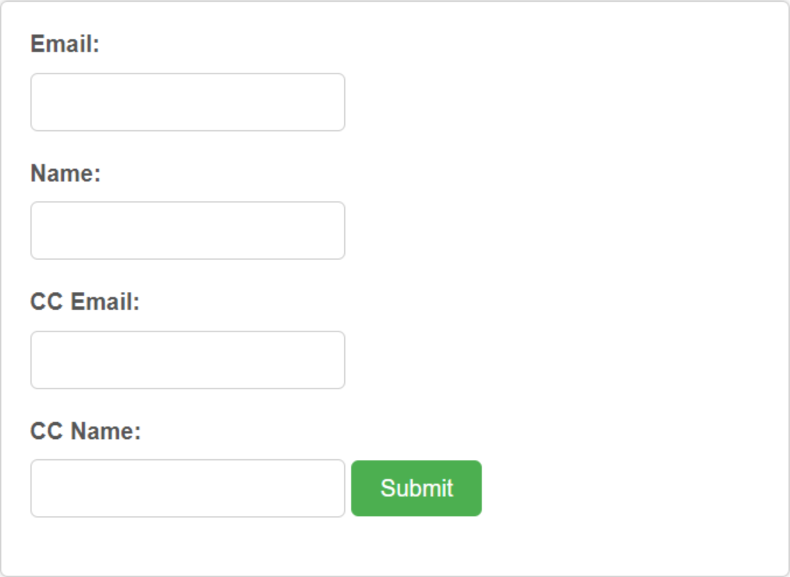
Index.js is the main file that contains the express app.

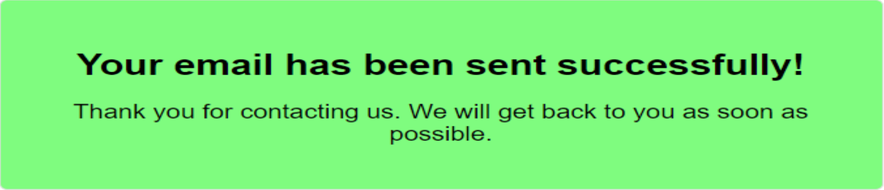


```
app.get('/', async (req, res) => {
  res.sendFile(__dirname + "/form.html");
})

app.post('/sendemail', async (req, res) => {
  try {
    let accountInfo = await authenticate();
    let args = getArgs(accountInfo.apiAccountId, accountInfo.accessToken, accountInfo.basePath,
    req.body.email, req.body.name, req.body.ccemail, req.body.ccname);
    let envelopeId = signingViaEmail.sendEnvelope(args);
    console.log(envelopeId);
    res.sendFile(__dirname + "/success.html");
  }
  catch {
    res.send('<h1>Fail to sent email</h1>');
  }
})

app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})

```

form.html file has fields like signerEmail, signerName, ccEmail, and ccName. I can also add fields like emailSubject, emailMessage, etc.

9) UI :





Your email has been sent successfully!

Thank you for contacting us. We will get back to you as soon as possible.