SBOM Sprint #2

Thomas Eby, Pranav Gonepalli, Evan Hellersund, Manel Leong,
Camron Rule, Skyler Walker, Duohan Xu, Rachel Zheng

College of William & Mary
CSCI 435: Software Engineering
October 9, 2024

Sprint Goal

- Set up the Django application
- Communication between front end and back end
- Formalize user stories and product backlog

Sprint Backlog

- Create working Django application
- Create initial frontend interface for file upload and display
- Connect frontend and backend
- Transmit SBOM file data between frontend and backend
- Research vulnerabilities databases
- Set up issue tracker

Issue Tracker

csci-435-fall-2024 / csci-435-24_p3_sbom_viz

Type / to search

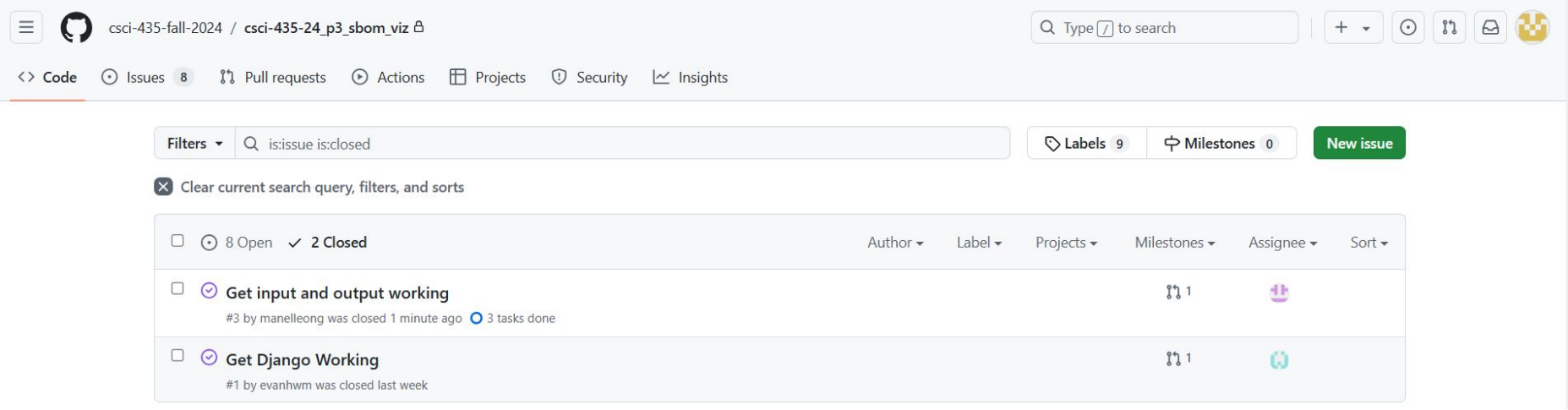
Code | Issues 8 | Pull requests | Actions | Projects | Security | Insights

Filters | is:issue is:open | Labels 9 | Milestones 0 | New issue

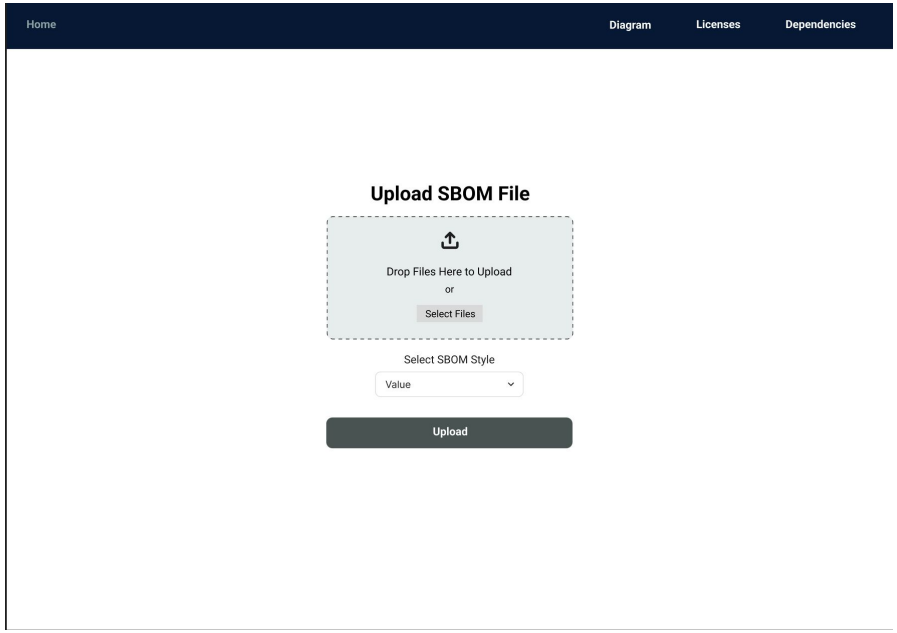
8 Open | 2 Closed | Author | Label | Projects | Milestones | Assignee | Sort

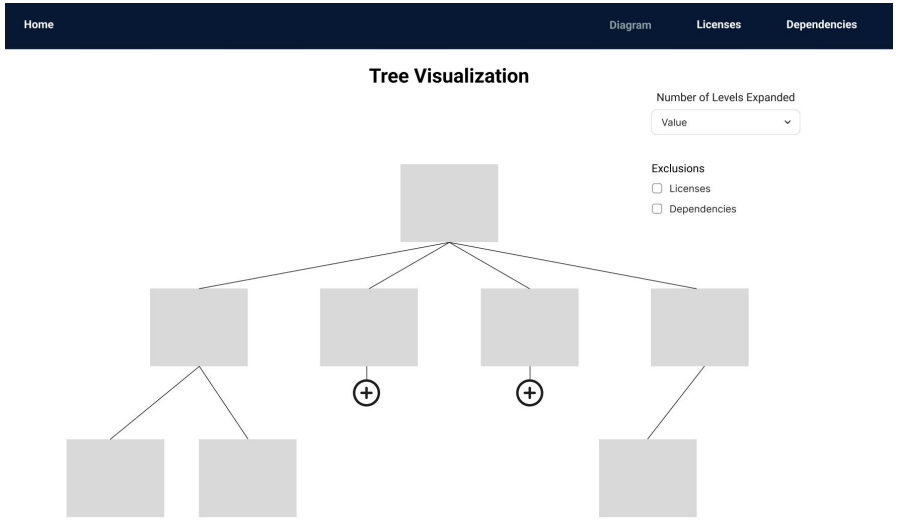
- **Visualize Security Vulnerabilities**
  #11 opened 4 days ago by camronrule 8 tasks
- **Filter Visualization**
  #10 opened 4 days ago by camronrule 5 tasks
- **Visualize License Distribution**
  #9 opened 4 days ago by camronrule 7 tasks
- **Download Summary Document**
  #8 opened 4 days ago by sdwalker2946 4 tasks
- **Generate Summary Document**
  #7 opened 4 days ago by sdwalker2946 5 tasks
- **Parse CycloneDX file**
  #6 opened 4 days ago by sdwalker2946 7 tasks
- **Parse SPDX file**
  #5 opened 4 days ago by sdwalker2946 7 tasks
- **Visualize Component Tree**
  #4 opened 4 days ago by sdwalker2946 6 tasks

Issue Tracker (Continued)

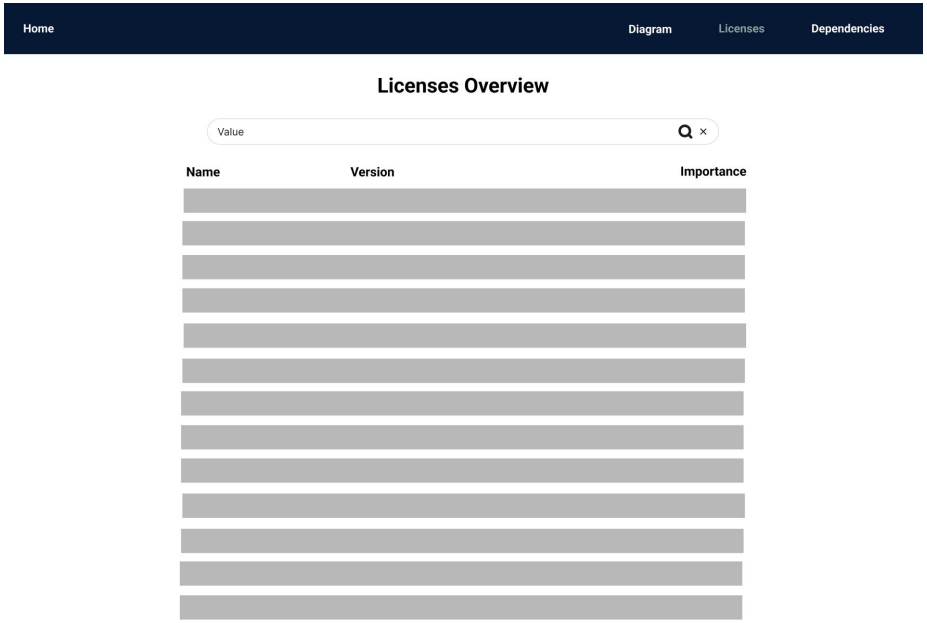


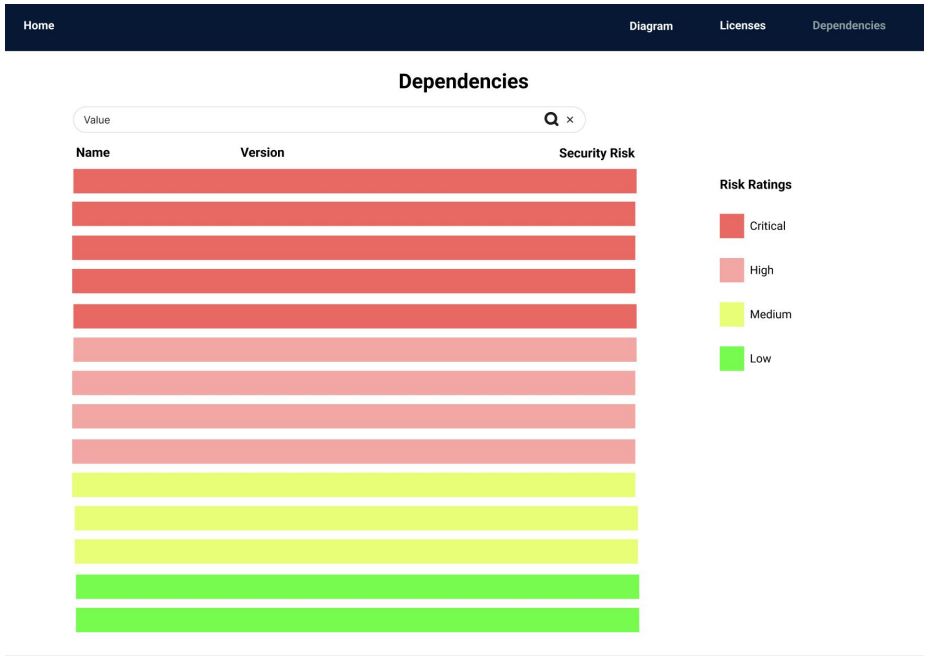
Design


Home
Diagram
Licenses
Dependencies
Upload SBOM File
Drop Files Here to Upload
or
Select Files
Select SBOM Style
Value
Upload


Home
Diagram
Licenses
Dependencies
Tree Visualization
Number of Levels Expanded
Value
Exclusions
Licenses
Dependencies

Design (Continued)





Security Vulnerability Databases	Licensing
• CVE • NIST National Vulnerability Database • CISA Known Exploited Vulnerabilities Catalog	• SPDX • CycloneDX

Connecting to a Security Vulnerabilities Database

2 options:

1. National Vulnerability Database
    a. Has an API and search (can filter by product, match by keyword, etc)
    b. With API key: 50 requests/ 30 seconds; Without API key: 5 requests/ 30 seconds
    c. Concerned with the amount of time it takes to query vulnerabilities for every component of the project
2. CVE
    a. Has search, github of json files of all CVE

References

- From https://github.com/CortezFrazierJr/my_recipe_book/blob/main/sampleCycloneDX.json:
  - CVE-2022-36077 is a vulnerability of pkg:npm/electron@11.1.1
  - CVE-2021-30518 is a vulnerability of pkg:npm/electron@11.1.1

More on NVD

- Searching is slow but allows filtering by product which seems to be more effective in finding the vulnerabilities we want than a search by keyword
  - 673 results with keyword search, 24 when searching by product (tested on electron package)
- API seems to work faster but haven't found a way to search by product
  - Sometimes hard to match package name on SBOM to cpe (naming convention on NVD)
  - Example:
    - electron > **cpe:2.3:a:electronjs:electron:*:*:*:*:*:node.js:*:***
- The example SBOM was also able to find outside dependents
  - One of the dependents of the electron package is chrome so the example SBOM(mentioned above) stated CVE-2021-30518 as a vulnerability of pkg:npm/electron@11.1.1
  - However, CVE-2021-30518 didn't come up when searching by keyword or product on the word "electron" since nothing in the NVD page on CVE-2021-30518 specifically mentions electron

Example of Using NVD's Rest API

https://services.nvd.nist.gov/rest/json/cves/2.0?cpeName=cpe:2.3:o:microsoft:windows_10:1607:*:*:*:*:*:*:*


```
{
  "resultsPerPage": 2000,
  "startIndex": 0,
  "totalResults": 2526,
  "format": "NVD_CVE",
  "version": "2.0",
  "timestamp": "2024-10-09T17:19:08.307",
  "vulnerabilities": [
    {
      "cve": {
        "id": "CVE-2013-3900",
        "sourceIdentifier": "secure@microsoft.com",
        "published": "2013-12-11T00:55:03.693",
        "lastModified": "2022-11-02T15:15:43.850",
        "vulnStatus": "Analyzed",
        "cveTags": [],
        "cisaExploitAdd": "2022-01-10",
        "cisaActionDue": "2022-07-10",
        "cisaRequiredAction": "Apply updates per vendor instructions.",
        "cisaVulnerabilityName": "Microsoft WinVerifyTrust function Remote Code Execution",
        "descriptions": [
          {
            "lang": "en",
            "value": "The WinVerifyTrust function in Microsoft Windows XP SP2 and SP3, Windows Se
Windows Server 2012 Gold and R2, and Windows RT Gold and 8.1 does not properly validate PE file d
crafted PE file, aka \"WinVerifyTrust Signature Validation Vulnerability.\""
          },
          {
            "lang": "es",
            "value": "La función WinVerifyTrust en Microsoft Windows XP SP2 y SP3, Windows Server
Server 2012 Gold y R2 y Windows RT Gold y 8.1 no valida correctamente los digest de archivos PE d
mediante un archivo PE manipulado, también conocido como \"WinVerifyTrust firma vulnerabilidad de
          }
        ],
        "metrics": {
          "cvssMetricV2": [
            {
              "source": "nvd@nist.gov",
              "type": "Primary",
              "cvssData": {
                "version": "2.0",
                "vectorString": "AV:N/AC:H/Au:N/C:C/I:C/A:C",
                "accessVector": "NETWORK",
                "accessComplexity": "HIGH",
                "authentication": "NONE",
                "confidentialityImpact": "COMPLETE",
```

More on CVE

- Some files seem to be missing information like cvss scores
- No search by product - return irrelevant results, but searching seems easier than on NVD

Current State of Project

views.py

sbom_viz > views.py > ...

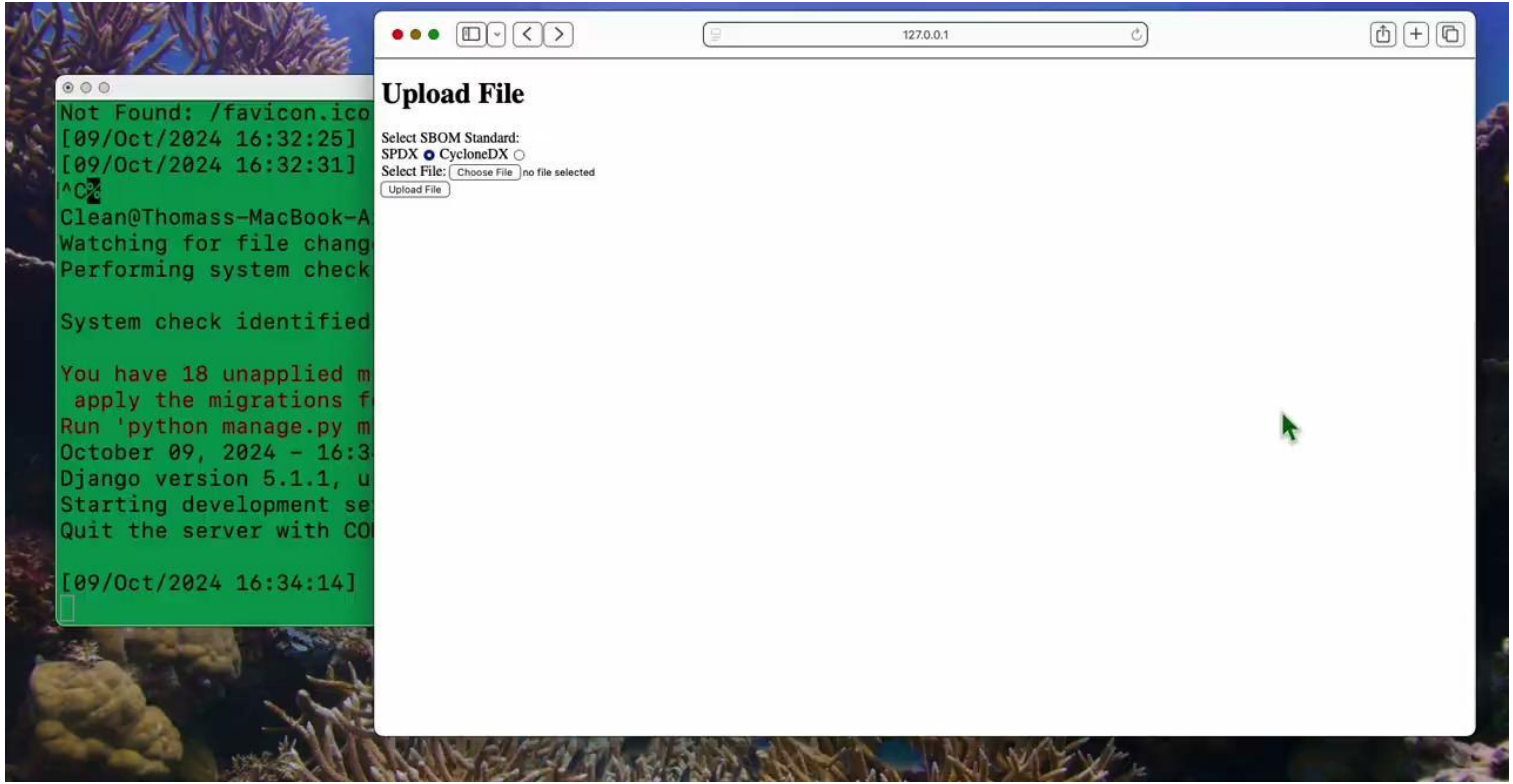
```python
from django.shortcuts import render
from django.http import HttpResponse

def home(request):
    if request.method == "POST" and len(request.FILES) == 1:
        file = request.FILES["file-select-input"]
        file_contents = ""
        for line in file:
            file_contents += line.decode()+'\n'
        return render(request, 'sbom_viz/display_file.html', {"file_contents": file_contents})
    else:
        return render(request, 'sbom_viz/index.html')
```

display_file.html

sbom_viz > templates > sbom_viz > display_file.html > html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Display File</title>
</head>
<body>
    {% block content %}
        {{ file_contents|linebreaks }}
    {% endblock %}
</body>
</html>
```

Demo


Not Found: /favicon.ico
[09/Oct/2024 16:32:25]
[09/Oct/2024 16:32:31]
^C%
Clean@Thomass-MacBook-A
Watching for file chang
Performing system check
System check identified
You have 18 unapplied m
apply the migrations f
Run 'python manage.py m
October 09, 2024 - 16:3
Django version 5.1.1, u
Starting development se
Quit the server with CO
[09/Oct/2024 16:34:14]
127.0.0.1
Upload File
Select SBOM Standard:
SPDX CycloneDX
Select File: Choose File no file selected
Upload File

Next Sprint Backlog

- Parse SPDX file and CycloneDX file

  Being able to pass components from frontend to backend using HTTP requests and process SBOM data in the backend

- Begin visualizing components as a tree

Lessons

- Establishing a meeting plan
  - Identify most important topics before meeting
  - More comprehensive coverage of project priorities
  - Get everyone on the same page
  - Spur conversations about features
- To fully utilize Django, we need to transition between HTML pages rather than using a single page approach with JavaScript controlling transitions

Contributions

Manel: Set up the project

Skyler: Implemented front-end interface for SBOM file upload and display, set up some issues in the issue tracker

Thomas: Linked the file upload to the backend and passed the contents back as a string

Pranav: Created website design

Camron: Contributed to user stories and GitHub issues

Rachel: Researched how to connect our project to the SVE database

Evan: Researched licensing and security databases

Duohan: Updated ReadMe, currently developing the SPDX parsing branch

Everyone: Contributed to user stories and the slideshow presentation.