JSON Schema

Build more. Break less. Empower others.

Proposal for GSoC 2024

Build Data-Driven Tooling Page

- **Name:** Divyaditya Singh Naruka
- **Country of Residence**: India
- **Telephone**: [redacted]
- **Email**: [redacted]
- **Time zone**: IST (GMT +05:30)
- **Academic Status**: Final-Year Undergraduate (CSE)
- **GitHub**: http://github.com/darhkvoyd
- **Languages**: English, Hindi

Declaration: There is **NO** use of Generative AI such as Gemini, ChatGPT etc. Google Docs proofread feature is used.

Why I’ve chosen this project & JSON Schema as my Organisation.

Quick Story: My father bought our first computer when I was born, so eventually I have grown up spending a lot of time on screen. Over the time, I have used so many open source projects and therefore I am a huge fan and advocate of the work and building tools that just make people’s lives easier.

Background: So, when I faced some friction while surfing the web, I started to build my own tool **"Toppings"**, which is a browser extension and I started to learn specific technologies to build my project but I have no experience working in open source so each time I learned something I started to indulge myself in open source projects to better grasp on a bigger codebase.

Motivation: I believe JSON Schema is an organisation that strongly resonates with my beliefs and ideology. The hard work by the team enables a lot of users to work with JSON without any hassle. Moreover, the time I have been there; everyone has been very welcoming, supportive and motivating to drive my desire to improve and work.

TLDR; I desire to learn from experienced open source projects, to build and improve helpful tools while giving back to the community.

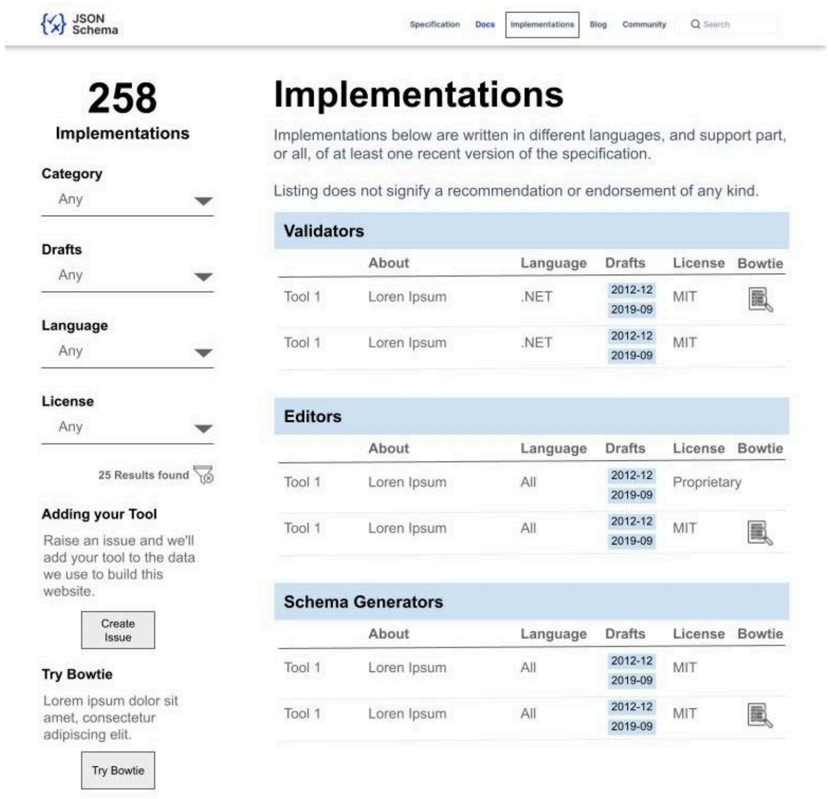
Project Description:

Name: Build a new version of the JSON Schema Tooling Page

Mentor: @benjagm

Goal: Recently, the JSON Schema website was updated with new UI/UX but the suboptimal UX of the implementations page is causing friction in adopting JSON Schema. Therefore, I need to rebuild the tooling page following the approved UX of the page to make the implementations of JSON Schema more accessible, adoptable, and easier to navigate.

TLDR; Re-build the JSON Schema tooling page to be data-driven and easier to adopt and navigate.



258
Implementations

Category
Any

Drafts
Any

Language
Any

License
Any

25 Results found

Adding your Tool

Raise an issue and we'll add your tool to the data we use to build this website.

Try Bowtie

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Implementations

Implementations below are written in different languages, and support part, or all, of at least one recent version of the specification.

Listing does not signify a recommendation or endorsement of any kind.

Validators

	About	Language	Drafts	License	Bowtie
Tool 1	Loren Ipsum	.NET	2012-12 2019-09	MIT	🔍
Tool 1	Loren Ipsum	.NET	2012-12 2019-09	MIT	

Editors

	About	Language	Drafts	License	Bowtie
Tool 1	Loren Ipsum	All	2012-12 2019-09	Proprietary	
Tool 1	Loren Ipsum	All	2012-12 2019-09	MIT	🔍

Schema Generators

	About	Language	Drafts	License	Bowtie
Tool 1	Loren Ipsum	All	2012-12 2019-09	MIT	
Tool 1	Loren Ipsum	All	2012-12 2019-09	MIT	🔍

Early Low-fi Prototype

Academic Knowledge:

I am currently a final-year Computer Science and Engineering undergraduate at Sant Longowal Institute of Engineering and Technology, India. I am enrolled in a 4 year B.E. (Bachelors in Engineering) program. **I'll graduate later this year, therefore I will be available during the GSoC contribution period.**

The following is a list of relevant courses I have completed from credible online course providers that have given me a deep understanding and hands-on experience required for this project:

- JavaScript - The Complete Guide 2024 (Beginner + Advanced)
- Understanding TypeScript
- React - The Complete Guide 2024 (incl. React Router & Redux)
- Next.js 14 & React - The Complete Guide

Projects & Prior Experience:

- **Toppings** - Toppings is a browser extension. I recognized that certain features were missing from the default web experience, so decided to develop Toppings to fill that gap. *Currently, I am actively working on a major update to v2.*

- **Toppings Website** - This website is built using SPA React. The primary goal is to showcase and promote my tool "Toppings". *With v2, I plan to migrate this to the latest App Router Next.js to improve the SEO and enable web crawlers for a better ranking.*

- **Personal Blog** - This is currently ***work in progress*** and as of now contains fake lorem ipsum text. I hope to publish my experiences as part of my GSOC journey and other projects I will work on. Using popular technologies such as Next.js & R3F to gradually build a 3D interactive portfolio to showcase my web abilities.

Existing Contributions to JSON Schema:

Currently, in my time at JSON Schema during pre-GSoC time period. I have actively contributed through discussions, issues, PR reviews and authored pull requests to JSON Schema as mentioned below:

1. Improve Carbon Ads (PR): The previous Carbon ads integration on the website had multiple bugs such as duplicate, ineffective renders and problematic placements. The PR aimed to build a modular component for effective integration and maximise CPM placements for Carbon ads responsive to different section layouts.
2. Improve Docs Feedback (PR): The previous implementation of the documentation help section was inconsistent in styles and lacked a public feedback form. The PR aimed to improve the existing docs help section making its components consistent and add an easy feedback form. Airtable and Cloudfare Worker handle submissions.
3. PR Reviews: With multiple contributors simultaneously working, several pull requests were opened building up pending reviews. With my experience, I was able to review these pull requests and collaborate with authors to ensure the PRs were ready for final review by maintainers.
4. TeamWork & Collaboration: Actively participated in teamwork and discussions to collaborate with multiple contributors to collectively reach the best solutions. For critical & complex issues that required attention and careful consideration such as Hydratation Errors followed up with a co-authored pull request.
5. Authored Issues: For a smooth and efficient user experience, I authored these issues from discovery to collaboration to fix these.

Plan & Technical details:

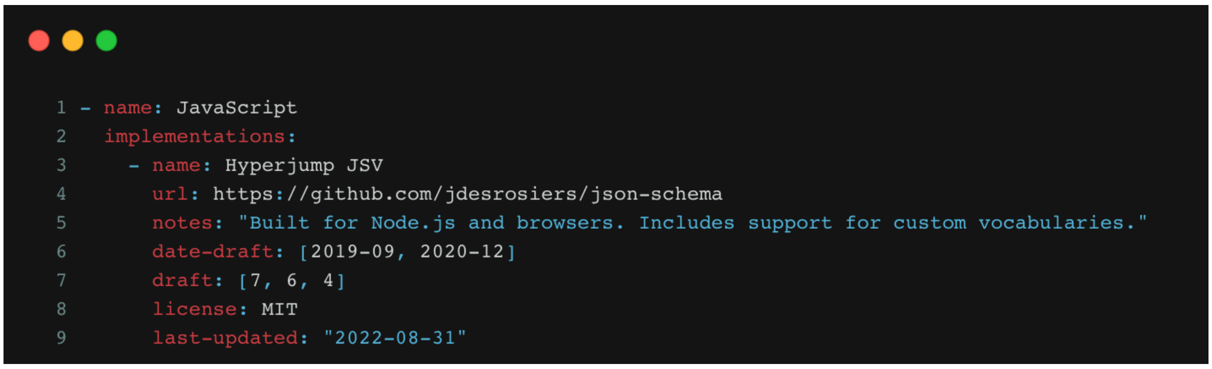
Plan:

- **Steps**:
  - **Consistent Component & Structure**: Given the importance of JSON Schema, it is essential to maintain minimum friction for users to adopt implementations and toolings. Therefore, the new proposed UI/UX designs and structure built keeping in mind data driven accessibility and flow, this should be used for a seamless and smooth experience.
  - **Implementing filters**: In addition, to enable easy navigation with quick access and help users find relevant implementations with minimum efforts, various filtration options must be available based on draft versions, licences, languages and more.
  - **Bowtie Integration**: With the wide eco-system built around JSON Schema and various dialects it's often confusing for users to decide a tooling. The integration with bowtie ensures seamless access to understanding and comparing implementations of the JSON Schema specification across all programming languages.
  - **Third Party Integrations**: Integration is not limited to bowtie, there is a wide scope and availability for integrations with various 3rd party services. For instance, as an improvement of integration with GitHub, addition of sections to show details of each tool.
  - **Implementing trends tracking**: While we utilise Plausible to store and visualise traffic. There is a great potential in tracking the toolings page to help us identify trends and insights of JSON Schema user traffic which could be used for improvements around data flow and performance and help us better understand scope of integrations.
  - **Ensure consistent responsiveness**: To ensure accessibility and usability across devices, the page must be designed to adapt various device sizes and aspect ratios to ensure consistent and easy access of data throughout possible device sizes.

Technical Details:

The following is an analysis of the current implementation with discussion around a dummy pseudo demonstration for a possible implementation, components and structure:

- A smooth and easy to navigate UI/UX is vital for the seamless adoption of JSON Schema, along with a well thought technical plan that lays the foundation for data driven implementation.
- Let’s begin by analysing the current implementation of the tooling page for a better understanding about our existing scope.
- The current toolings page data model relies on YAML for implementation data as demonstrated in the following figure.

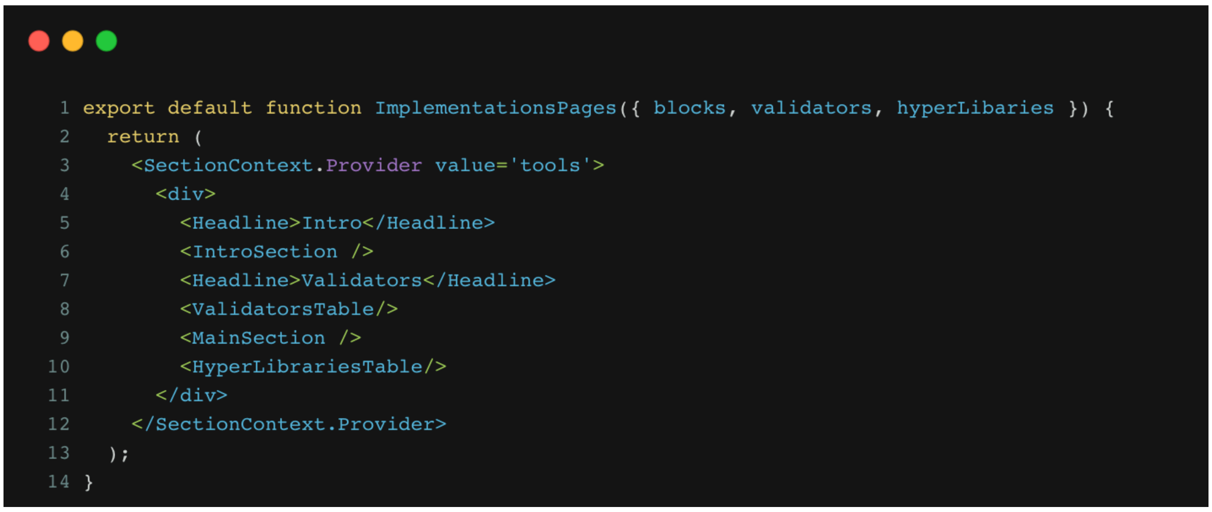


```yaml
- name: JavaScript
  implementations:
    - name: Hyperjump JSV
      url: https://github.com/jdesrosiers/json-schema
      notes: "Built for Node.js and browsers. Includes support for custom vocabularies."
      date-draft: [2019-09, 2020-12]
      draft: [7, 6, 4]
      license: MIT
      last-updated: "2022-08-31"
```

This data model was adopted when the website was built using `Jekyll` and it supports loading data from file types `YAML, JSON` etc. located in the `_data` directory and it was continued when we migrated to `Next.js`.

    - YAML is quite straightforward and human readable format to store data. Therefore, this data model is quite sufficient and we can continue to use it.
    - **Points to keep in mind while extending this into the new data model**:
        1. We could also merge old and new implementations into a single database so we can better tune the displayed data in the new page.
        2. There is a need to add references to **Bowtie reports** which should be displayed in a new section in the toolings table.

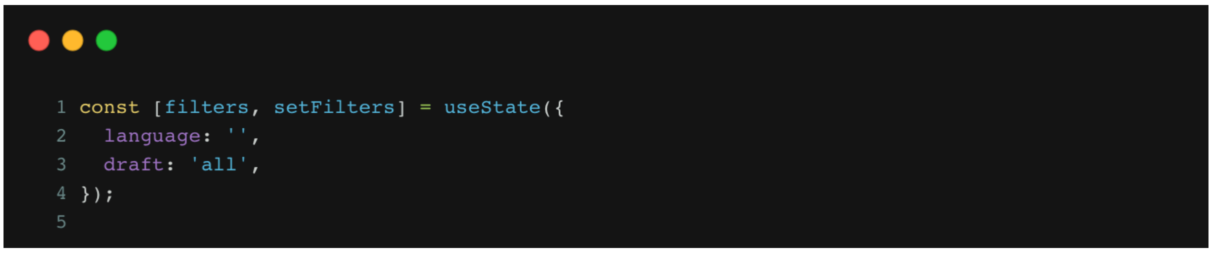
- Below is the **pseudo** structure of the current tooling page.



```
export default function ImplementationsPages({ blocks, validators, hyperLibaries }) {
  return (
    <SectionContext.Provider value='tools'>
      <div>
        <Headline>Intro</Headline>
        <IntroSection />
        <Headline>Validators</Headline>
        <ValidatorsTable/>
        <MainSection />
        <HyperLibrariesTable/>
      </div>
    </SectionContext.Provider>
  );
}
```

The implementation data is loaded as props into the page component by making use of the `getStaticProps()` method provided by the underlying infrastructure of `Pages Router` with static export configuration.

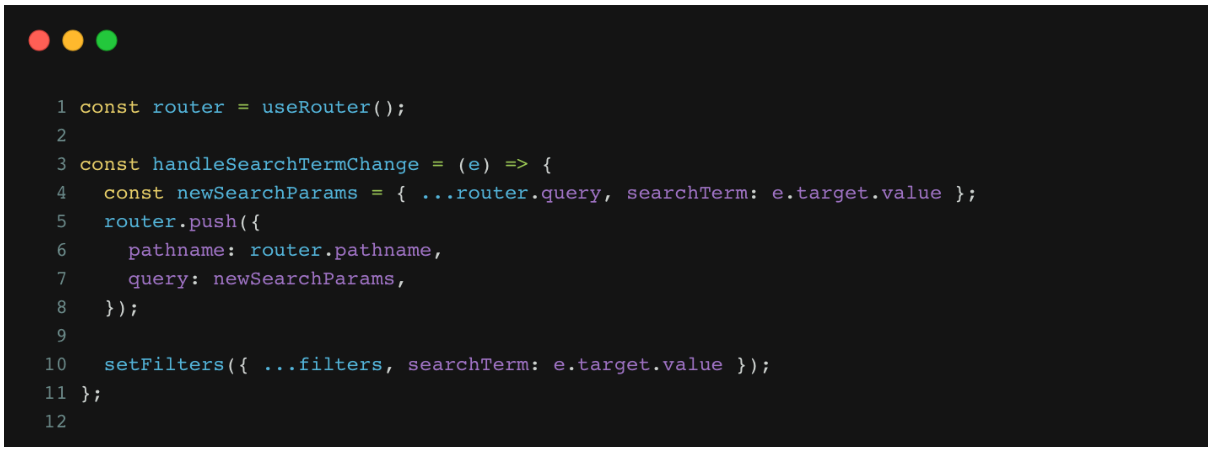
- In order to adapt the proposed design, we would have to refactor the implementation page into two sections as:
    1. `<FilterSection />` component which will be responsible for holding the logic for various data-driven filters based on draft, category, language, licence etc.
    2. `<ImplementationSection />` component which will render the available tooling and implementations available with the selected filters.
- Filters could be managed within a state object, which will be iterated through its properties to apply filter logic iteratively.



```
const [filters, setFilters] = useState({
  language: '',
  draft: 'all',
});

```

- To extend the page's capabilities and enable users to bookmark or share the relevant implementations with their preferred selection. We can use `URLSearchParams` to store the selected filters state.

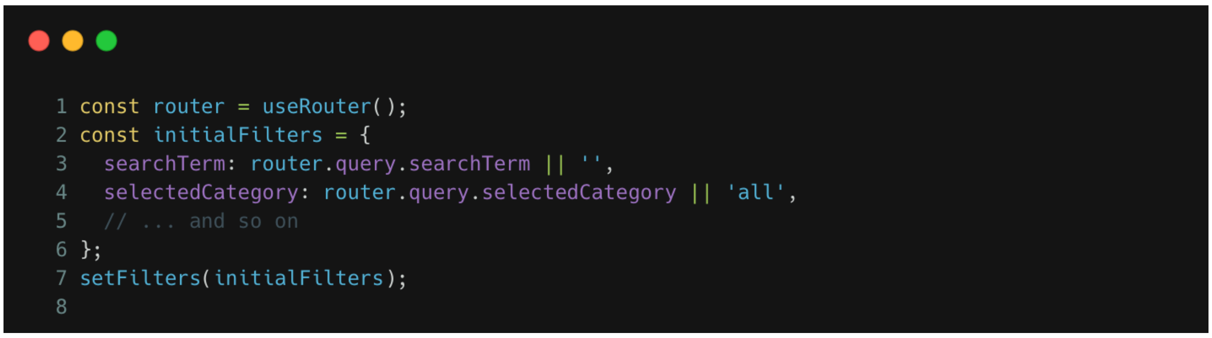


```
const router = useRouter();

const handleSearchTermChange = (e) => {
  const newSearchParams = { ...router.query, searchTerm: e.target.value };
  router.push({
    pathname: router.pathname,
    query: newSearchParams,
  });

  setFilters({ ...filters, searchTerm: e.target.value });
};

```



```
const router = useRouter();
const initialFilters = {
  searchTerm: router.query.searchTerm || '',
  selectedCategory: router.query.selectedCategory || 'all',
  // ... and so on
};
setFilters(initialFilters);

```

Proposed Behaviour:

1. Default to only showing draft-07 and above, but marks it clearly that a filter is enabled.
2. When selecting language/s, displaying number of results based on other filters (or grey out 0 result options)
3. Options in "drafts" to select all "modern" aka draft-07 and up, in one go

- Many implementations include additional notes that could be displayed using `Tooltips` when the user hovers over info icon such as:

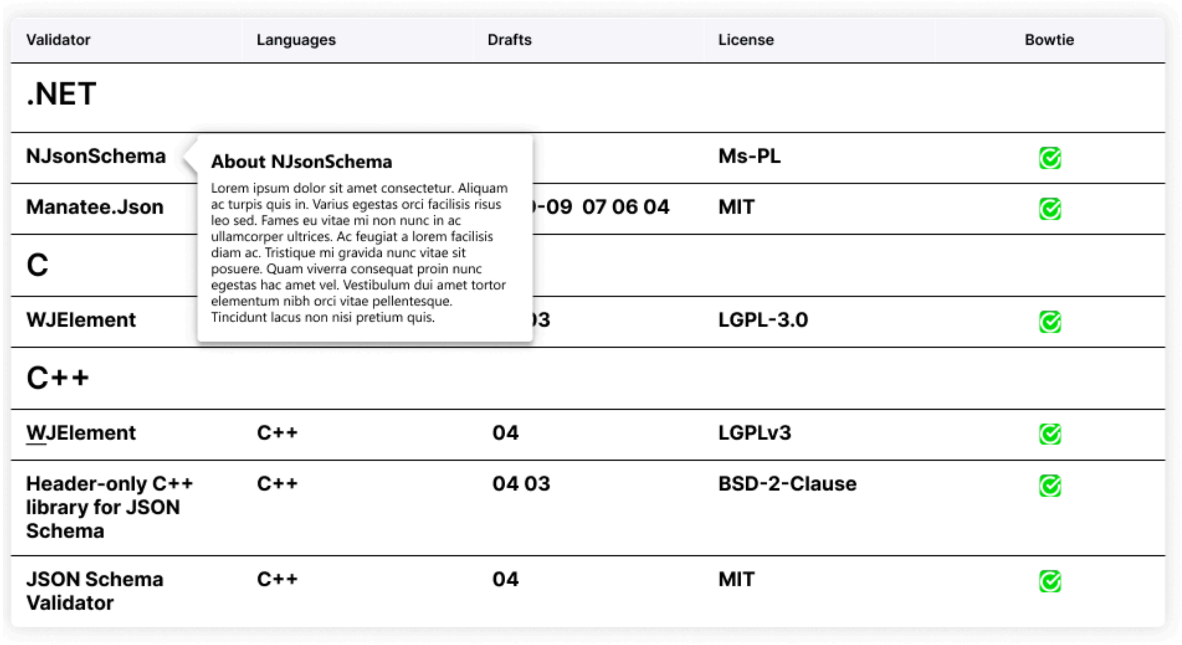
Tooltip design

- Showcase related popular available call-to-actions such as **Try Bowtie Meta-Validator, Edit on GitHub** and **Add your tool.**

Raise an issue and we'll add your tool to the data we use to build this website.

Bowtie is a meta-validator for JSON Schema, coordinating and reporting results from other validators.

- Integrating **Bowtie** into the JSON Schema tooling page is an important addition as benchmarks play a crucial role in the ecosystem.



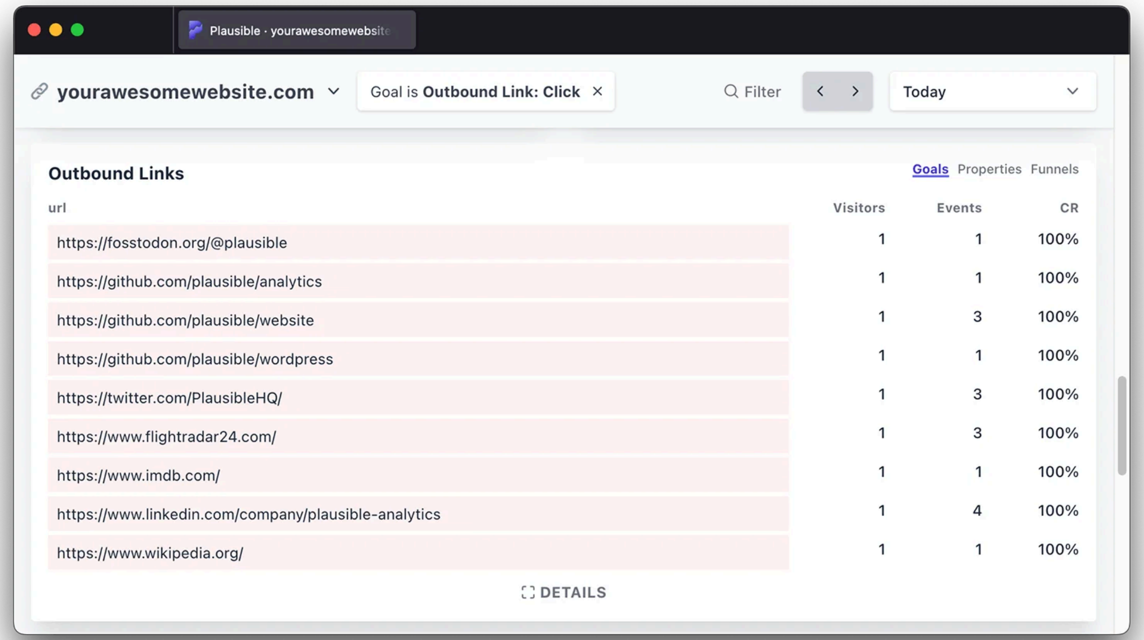
Validator	Languages	Drafts	License	Bowtie
.NET				
NJsonSchema			Ms-PL	✓
Manatee.Json		-09 07 06 04	MIT	✓
C				
WJElement		3	LGPL-3.0	✓
C++				
WJElement	C++	04	LGPLv3	✓
Header-only C++ library for JSON Schema	C++	04 03	BSD-2-Clause	✓
JSON Schema Validator	C++	04	MIT	✓

About NJsonSchema

Lorem ipsum dolor sit amet consectetur. Aliquam ac turpis quis in. Varius egestas orci facilisis risus leo sed. Fames eu vitae mi non nunc in ac ullamcorper ultrices. Ac feugiat a lorem facilisis diam ac. Tristique mi gravida nunc vitae sit posuere. Quam viverra consequat proin nunc egestas hac amet vel. Vestibulum dui amet tortor elementum nibh orci vitae pellentesque. Tincidunt lacus non nisi pretium quis.

- The proposed data model includes references to **Bowtie Reports** to be added as a column to the table of validators with links to all benchmarks that cover a specific implementation.

- There are various possible approaches to implement tracking mechanisms to monitor user interactions, engagement patterns, and trends on the tooling page.
    1. **Plausible Outbound Links**: Since, we already use Plausible to handle website analytics and visualisation we can extend Plausible integration to track outbound links. While it has some benefits of abstraction, this limits our control over custom tracking.



Outbound Links

url	Visitors	Events	CR
https://fosstodon.org/@plausible	1	1	100%
https://github.com/plausible/analytics	1	1	100%
https://github.com/plausible/website	1	3	100%
https://github.com/plausible/wordpress	1	1	100%
https://twitter.com/PlausibleHQ/	1	3	100%
https://www.flightradar24.com/	1	3	100%
https://www.imdb.com/	1	1	100%
https://www.linkedin.com/company/plausible-analytics	1	4	100%
https://www.wikipedia.org/	1	1	100%

    2. **Custom Tracking**: To maintain more control and customisation over tracking and metrics for insights, a combination of tools can be used to implement custom tracking. Currently, I am inclined towards using **Airtable** and **Cloudflare Worker** to store events.

- Events that we could track to gather insights,
    1. **Tooling More Info**: Recording when users hover over specific tooling to view more details such as accessing tooltip additional resources.
    2. **Bowtie Report Access**: Monitoring when users access Bowtie compliance reports for individual toolings, indicating interest in compliance and benchmarking data.
    3. **Filter Usage**: Tracking the use of filters and search functionality to understand how users navigate and refine for popular languages, drafts, licences and more.
    4. **Call to Action Interactions**: Monitoring interactions with call-to-action buttons, such as "Try Bowtie" or "Add Your Tooling," to keep track of user engagement with these features.
    5. **External Link Clicks**: Recording clicks on external links associated with toolings, such as documentation or GitHub repositories, to measure user interest in additional resources.

What do I expect from participating in GSoC?

I am very passionate about the tools and programs that I am currently building and plan to build in the future. To build them effectively I aspire to learn various aspects of building, maintaining open source projects and industry standards and practices from this community of experienced and talented members.

In addition, I wish to connect and gradually be a part of this welcoming community and hopefully, eventually someday become a mentor for GSoC in coming years.

Lastly, up until now I have been working by myself with no experience working in a team or organisation setup. Participating in GSoC would offer me the opportunity to gain this experience. This is a wonderful learning opportunity and would also help build my work experience, to possibly showcase my abilities for any future work opportunities after the GSoC period.

TLDR; Learn from experienced community members, become a part of the community, possibly even become a future GSoC mentor.

Timeline:

Time Commitment:
During GSoC's 12 week timeline, I would be able to devote ~48 hr/week to ~60 hrs/week with the flexibility to increase if the need arises.

Community Bonding:
In my time contributing to JSON Schema during the pre-GSOC time, the team and mentors have been extremely supportive and encouraging. The positive and welcoming attitude of the entire community keeps me motivated and a smile on my face.

Given my existing familiarity and bonding with the community, I wish to get more involved and socially engaged with the project. In addition, I plan to start writing about my learnings during the pre-GSOC phase and as a contributor through blog posts during the entire period about my experiences.

Lastly, this will be followed by finishing any formal requirements such as the filing tax forms required by Google, any contributor licence agreements, and any paperwork that the project requires.

- **Week - 1:**
  - The first step before starting to work on the issue is to review the discussion and capture the last requirement.
  - Also, review mock drafts and design proposals and identify areas of improvement.
  - Collaborate with mentors to iterate on the design and functionality.

- **Week - 2, 3:**
    - Implement layout, styling, and interactive elements according to the approved design specifications.
    - Design a plan for component behaviour across devices to ensure consistent and reliable access irrelevant to the device. For instance, filter component behaviour as discussed here.
    - For filters, When on Desktop, have them across the top, and sticky (stay when you scroll, and not on mobile). For mobile, a filter button which opens an overlay with the filter options (like the main menu)

- **Week - 4, 5:**
    - Develop the filter section and mechanism to enable users to search and filter implementations based on draft versions, licences, languages, and other criteria as discussed here.
    - We may want to add a filter for licence which is "All OSI approved licence"
    - When selecting language/s, show number of results based on other filters (or grey out 0 result options)
    - Reset all filters button
    - Options in "drafts" to select all "modern" aka draft-07 and up, in one go

- **Week - 6, 7:**
  - Design and implement tabular components to display tooling and implementation information, including details such as name, description, version compatibility, and licence.
  - Implementation of additional sections to include additional notes about compatibility and more as discussed here.
  - For instance, ajv included a note that you need to turn off strict mode to get spec compliant behaviour.
  - Add old implementations to the source data model of the new implementers page so we have everything in one single database and we can better tune the data shown in the implementers page.

- **Week - 8, 9:**
  - Implement tracking mechanisms to monitor user interactions, engagement patterns, and trends on the tooling page.
  - **We are open to explore between Plausible analytics for external traffic or implementation of manual tracking using Cloudflare worker and Airtable integrations.**
  - Track popular bowtie report traffic and outbound traffic to implementation docs and GitHub repositories.
  - Ensure to capture insights without affecting the urls to properly track and analyse how JSON Schema users are accessing the implementations so we can identify trends and insights.

- **Week - 10, 11:**
    - Integrate Bowtie functionality for comprehensive insights into JSON Schema performance and benchmarks across programming languages as discussed here.
    - Implementation of various 3rd party integrations such as integration with GitHub, addition of sections to show details of each tool.
    - Implement call-to-actions buttons such as Try Bowtie Meta-Validator, Edit on GitHub and Add your tool etc.
    - Could link to the benchmark section, with it pre-filtered to only the benchmarks that cover it.

- **Week - 12:**
    - Take community and mentor feedback to ensure all deliverables meet requirements and quality standards.

- **Week - 13:**
    - Spare week to address any remaining feedback, refinement or in case of some work getting delayed possibly due to some emergency or otherwise.

Acknowledgement: While making the proposal, I have taken the reference from the 2023 GSoC Proposal made by Priyanshu Sharma.