





# Xpress-Bio: a general purpose NodeJS based buildless server framework oriented towards bioinformatics applications

Ibrahim Tanyalcin<sup>1\*</sup>, Rodrigo Pacifico<sup>2\*</sup>, Shaida Moghaddassi<sup>2</sup>, Nicholas Hand<sup>2</sup>, Alex Cherekos<sup>3</sup>, Rui Portela<sup>1</sup>, and Sébastien Janas<sup>1</sup>

<sup>1</sup> GSK, Rixensaart, Belgium <sup>2</sup> GSK, Cambridge, MS, US <sup>3</sup> GSK, Upper Providence ¶ Corresponding author \* These authors contributed equally.

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

## Authors:

Ibrahim Tanyalcin<sup>\*†</sup>, Rodrigo Pacifico<sup>††</sup>, Shaida Moghaddassi<sup>††</sup>, Nicholas Hand<sup>††</sup>, Alex Cherekos<sup>†††</sup>, Rui Portela<sup>†</sup>, Sébastien Janas<sup>†</sup>

\* Corresponding author <sup>†</sup> GSK, Rixensaart, Belgium <sup>††</sup> GSK, Cambridge, MS, US <sup>†††</sup> GSK, Upper Providence, PA, US

Editor: 

Submitted: 07 October 2025

Published: unpublished

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/))

## Summary

Xpress-Bio is a buildless, extendible and optionally containerized web server application for interactive genomic data exploration and analysis. It combines widely used bioinformatics tools (e.g., Samtools, BLAST+, HTSlib, IGV.js) into a unified framework that supports web technologies such as Server-Sent Events, WebSockets and subscriptions in the frontend. In the backend, it takes advantage of multi-threaded worker patterns implemented in NodeJS. Designed for bioinformaticians who prefer low-abstraction, high control setups, Xpress-Bio operates on both standalone and containerized environments (Docker, Singularity) with a focus on ease of extension, reproducibility, and control over data processing.

## Statement of need

While platforms like Galaxy and EMBOSS provide general-purpose bioinformatics functionality, they impose rigid templates that limit custom routing, extensibility, and real-time interaction Afgan et al. (2018). Xpress-Bio offers a lower-level, developer-centric alternative that still includes critical functionality such as live data updates, BLAST search, and indexed genome browsing.

Unlike typical web frameworks, Xpress-Bio is pre-configured for genomics workflows out of the box. It handles automatic indexing (.fai, .bai, .csi, .tbi), dynamic BLAST database generation, and efficient cache management. The framework exposes Express.js route configuration, server state, event emitters, and WebComponent-based UI elements that allow full-stack extensibility.

Xpress-Bio uses a modular architecture for server routes. Route modules placed under the routes/ directory are automatically discovered and loaded at startup. Each file can define one or more routes in the form of sync/async functions that receive shared utility objects as arguments. Hot reloading is supported via nodemon in the form of full server restart. During full restarts, all active worker threads are gracefully terminated and clients reconnect automatically on the frontend. Resource intensive or long running tasks are executed via persistent worker threads that support bidirectional communication via MessageChannels.

39 All in all, Xpress-Bio is suited to research scientists that need fast, no-build, containerized  
40 BLAST, visualization, and LLM stacks that need real-time communications with server-sent  
41 events or web-sockets.

## 42 Implementation and Features

### 43 Implementation

44 Built upon Express.js/Node.js, requires no build step to initialize or start the server. New routes  
45 are a collection of functions that can be dropped under `app/js/server/routes`. These are plain  
46 js files that define a single function which receives `express`, `app`, `info`, `files`, `serverSent`, `ws`  
47 and `cache` parameters. `Express` is the `express js` constructor, `app` is the `app` instance, `info` is  
48 server information with all custom data as a result of merged config files, `files` are list of files  
49 that can be requested by clients, `serverSent` and `ws` are proxy objects that can be utilized to  
50 send individual or broadcast messages via server-sent events or web-sockets and finally `cache`  
51 is the memcached key-value storage with 1 month expiry.

52 BLAST support: Queries run on dedicated threads using a batch scheduler and return results  
53 in both standard and tabular formats [Camacho et al. (2009);]. (see Figure 1, panels c–f)

54 UI built using native WebComponents and IGV.js, enabling multi-track visualizations and  
55 real-time interaction [Robinson et al. (2011);].

### 56 Features

57 Works with FASTA, BAM, GFF files and supports automated indexing via Samtools/HTSlib  
58 [Bonfield et al. (2021);]. (see Figure 1, panels c–f)

59 Users can run BLAST on the uploaded sequences. BLAST results are generated in 2 formats  
60 and are downloadable. Clicking on BLAST results navigates to the corresponding locus within  
61 the selected IGV applet.

62 Supports Server-Sent Events for real-time updates and WebSockets for bidirectional com-  
63 munication. (see Figure 1, panels a, b, g). The server maintains responsiveness up to 100  
64 concurrent clients.

65 LLM integration: Users can connect public/private LLMs to sequence data via a tool called  
66 “g-nome”, enabling conversational querying and UI extension through proposed code snippets.  
67 (see Figure 1, panel g). Multiple agents can be spawned simultaneously. Agents do not  
68 run code automatically, the user has the opportunity to review the snippets inside the chat  
69 component before they are executed. Data sent includes HTML tree, visible sequence, filenames  
70 and other metadata such as BLAST content. These can be tweaked or removed by the  
71 developers if necessary.

### 72 Example: BLAST Query Performance

73 Queries were performed against the **Chinese hamster genome** derivative, a fairly large FASTA  
74 file as reference

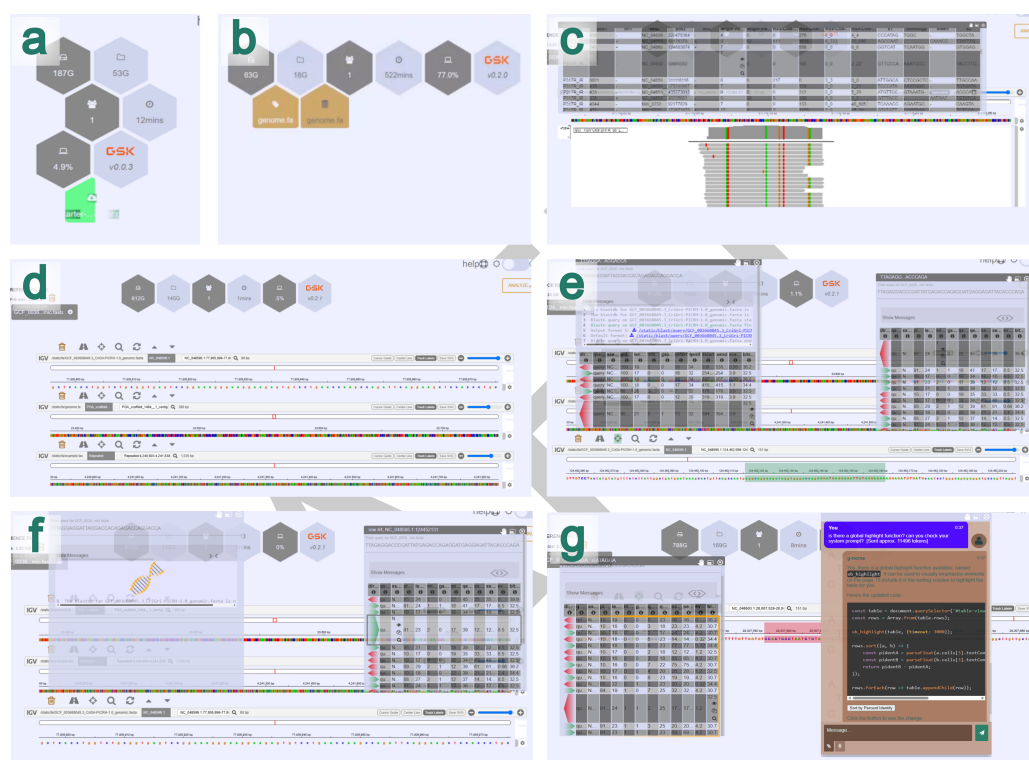
75 **Table 1.** BLAST query performance measured on a 6-core Ryzen 5 PRO system.

Number of queries	Query length (nt)	Execution time (s)
1-5	20–30	< 5
5-10	20–30	5–10
10-20	20–30	5–15

76 Measured on a 6-core, 12-thread Ryzen 5 PRO with 16GB RAM. Queries were performed on  
77 Chinese hamster genome\*. Queries and database generation are processed in separate worker  
78 threads for responsiveness.

79 \* For comparisons, the size of the input genome is about 2.5Gb.

## 80 Figures



**Figure 1:** A non-exhaustive overview of Xpress-Bio features.

- (a) Uploading from URL or local file shows progress as a green hexagon.
- (b) Uploaded annotations, FASTA files, BAM files and BLAST databases are indexed or created on separate threads simultaneously. The progress is shown as pulsating orange hexagons.
- (c) Tables can be joined with FASTA/BAM pairs. Clicking on table row that has position information relocates the IGV viewer to that location.
- (d) Multiple IGV applets can be added and reordered at any time. Each applet can load its own set of BAM files and annotations.
- (e) BLAST results render outputs in a draggable table. Each table shows the orientation (forward/reverse) of the hit in the first column. Multiple BLAST windows can be simultaneously opened and each BLAST result can be paired with one of IGV applets, enabling click-to-navigate behavior.
- (f) Users can initiate multiple BLAST queries simultaneously. The BLAST search takes place on a separate thread and utilizes available CPU cores. Results are cached for retrieval.
- (g) Users can open the “g-nome” app on the side bar to connect to public LLM models with their API keys. Models get access to currently visible sequence data along with any BLAST windows and tracks. Xpress-Bio provides a mechanism for these LLM models to propose code snippets that can be run by the client which allows manipulation of visual elements on the screen. This allows scientists to extend UI functionality like sorting BLAST outputs and searching keywords within files.

## Acknowledgements

I.T. wrote the software, A.C reviewed pull-requests. N.H, Ro.P and S.M tested functionality and proposed routinely used features. Ro.P, S.M, Ru.P and S.J reviewed the manuscript and proposed features.

## Funding

This work was sponsored by GlaxoSmithKline Biologicals SA.

## Conflict of interest

All authors are, or were at the time of the study employees of the GSK group of companies.

## References

./paper.bib.

Afgan, E., Baker, D., Batut, B., Beek, M. van den, Bouvier, D., Čech, M., Chilton, J., Clements, D., Coraor, N., Grüning, B. A., Goecks, J., Taylor, J., Nekrutenko, A., & Blankenberg, D. (2018). The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. *Nucleic Acids Research*, 46(W1), W537–W544. <https://doi.org/10.1093/nar/gky379>

Bonfield, J. K., Marshall, J., Danecek, P., Li, H., Ohan, V., Whitwham, A., Keane, T., Davies, R. M., & Tischler, G. (2021). HTSlib: C library for reading/writing high-throughput sequencing data. *GigaScience*, 10(2), giab007. <https://doi.org/10.1093/gigascience/giab007>

Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K., & Madden, T. L. (2009). BLAST+: Architecture and applications. *BMC Bioinformatics*, 10(1), 421. <https://doi.org/10.1186/1471-2105-10-421>

Rice, P., Longden, I., & Bleasby, A. (2000). EMBOSS: The european molecular biology open software suite. *Trends in Genetics*, 16(6), 276–277. [https://doi.org/10.1016/S0168-9525\(00\)02024-2](https://doi.org/10.1016/S0168-9525(00)02024-2)

Robinson, J. T., Thorvaldsdóttir, H., Winckler, W., Guttman, M., Lander, E. S., Getz, G., & Mesirov, J. P. (2011). Integrative genomics viewer. *Nature Biotechnology*, 29, 24–26. <https://doi.org/10.1038/nbt.1754>