

## **Yggdrasil - D&D website proposal**

Andre van Heerden 241155

Open Window, Department of Fundamentals

Interactive Development 200

Tsungai Katsuro

05 November 2025

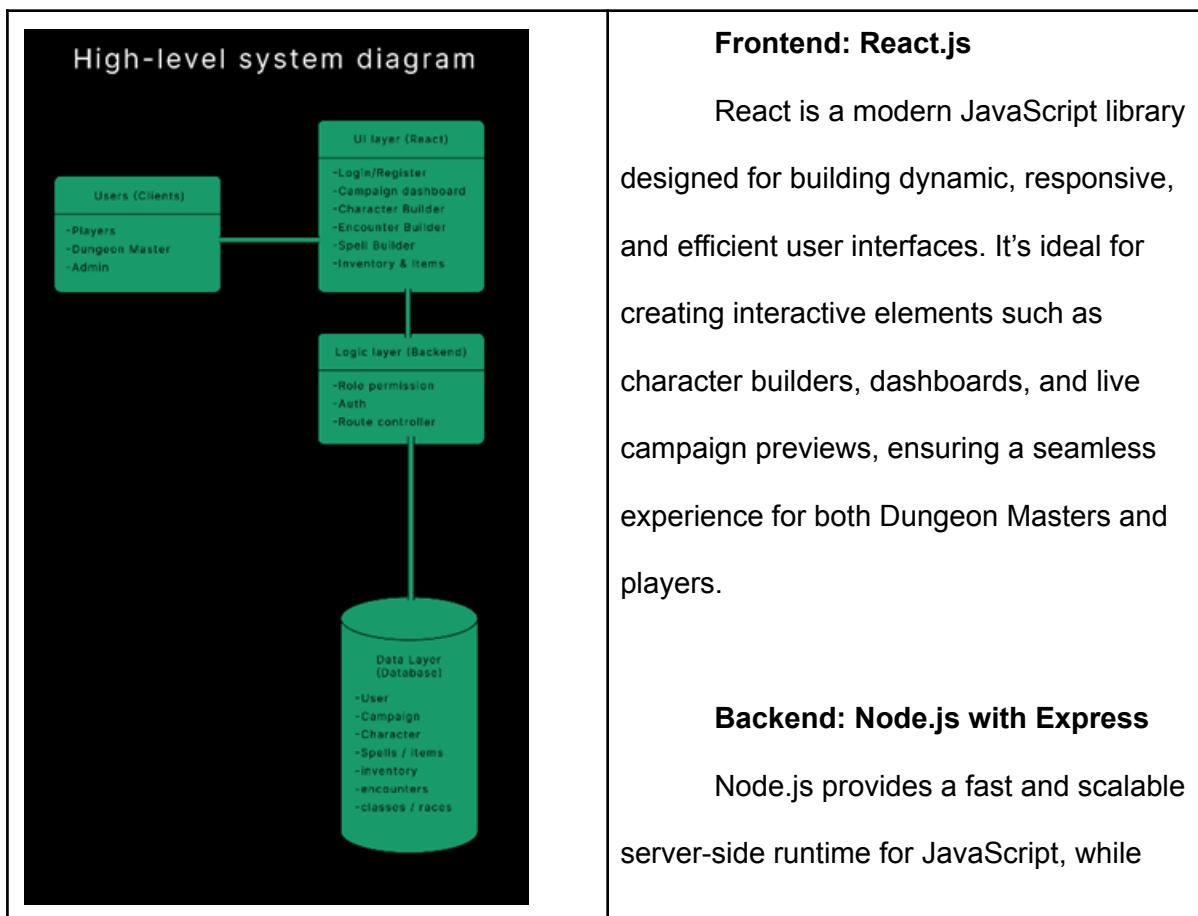
## Table of Contents

<b>What is Yggdrasil.....</b>	<b>3</b>
<b>How Yggdrasil was built - Technologies &amp; Frameworks.....</b>	<b>3</b>
<b>How Yggdrasil was built - Justification for Tech Stack.....</b>	<b>4</b>
<b>What problems does Yggdrasil solve.....</b>	<b>5</b>
<b>How was Yggdrasil deployed.....</b>	<b>5</b>
<b>Yggdrasil Feature Requirements &amp; Scope.....</b>	<b>6</b>
<b>Majored Features.....</b>	<b>6</b>
<b>User Roles.....</b>	<b>7</b>
<b>Yggdrasil Data Planning.....</b>	<b>7</b>
<b>Project Timeline &amp; Workflow.....</b>	<b>7</b>
<b>Risks, Challenges &amp; Conclusion.....</b>	<b>7</b>
Technical Risks.....	7
Non-Technical Risks.....	8
<b>Conclusion.....</b>	<b>9</b>

## What is Yggdrasil

Yggdrasil is an all-in-one platform built for Dungeon Masters and Dungeons & Dragons players. Designed to streamline the campaign experience. It makes running and managing D&D adventures effortless for both newcomers and seasoned veterans. The platform features a fully integrated, game-style inventory and creation system ,which empowers you to design your own classes, spells, weapons, items, monsters, characters, and more. I created Yggdrasil out of a desire to simplify the D&D experience. Unlike existing tools such as D&D Beyond, which often hide key features behind paywalls and clutter the campaign management process, Yggdrasil focuses on accessibility, flexibility, and intuitive design — so you can spend less time organizing and more time adventuring.

## How Yggdrasil was built - Technologies & Frameworks



	<p>Express simplifies routing, API development, and backend logic. Together, they deliver an efficient, modular system that integrates smoothly with React and supports real-time gameplay features.</p> <p><b>Database: MySQL</b></p> <p>MySQL is a reliable relational database system, perfect for managing structured D&amp;D data such as characters, stats, spells, and inventories. It supports complex queries, ensures data integrity, and maintains robust relationships between tables — making it ideal for a world as detailed as Dungeons &amp; Dragons.</p>
--	--

### How Yggdrasil was built - Justification for Tech Stack

- **Full JavaScript Stack:** React, Node.js and Express use JavaScript which simplifies development and reduces context switching
  - **Rapid Development:** React + Express enables faster prototyping with component reusage and quick API deployment.
  - **Scalability:** Node.js handles asynchronous operations and can scale to real-time features
  - **Data structure fit:** MySQL is perfect for handling highly structured and interrelated data typical in D&D.

## What problems does Yggdrasil solve

Most existing Dungeons & Dragons platforms restrict user creativity through paywalls and limited customisation options. These barriers prevent players and Dungeon Masters from fully expressing their imagination and managing their campaigns with freedom. Yggdrasil addresses this issue by offering a completely open and accessible platform .All core features are free, with no hidden costs or restrictions. Users gain full creative control over their campaigns, from designing characters and encounters to crafting unique worlds, classes, spells, and items. By removing limitations and empowering users to build without boundaries, Yggdrasil delivers a richer, more personalized D&D experience that goes beyond traditional platforms.

## How was Yggdrasil deployed

### Frontend:

The frontend is built with React.js and deployed on Vercel with a .xyz domain. It is connected to the backend via an environmental variable containing the backend URL, which was configured during the backend setup process.

### Backend:

The backend is deployed on Render and linked to the frontend using the URL provided during deployment. This URL is stored as an environment variable in the frontend to ensure secure and seamless communication between the two layers.

### Database:

The MySQL database is hosted on AlwaysData, which functions similarly to XAMPP. The only significant change required was updating the local configuration to use the AlwaysData connection details instead of the local setup. The database is also connected to the backend through environment variables for secure access.

## **Yggdrasil Feature Requirements & Scope**

Project scope

Included Features:

- Custom campaign creation and hosting
- Full Character creation: class, race, background, stats, spells and actions
- Custom Content creation: races, classes, items, spells, encounters, etc.
- User accounts and authentication
- Data storage and management in a database
- View, edit and delete Custom Content creation
- real-time combat
- real-time inventory management and editor

Excluded Features:

- Integrated map
- In-depth campaign/quest management system

### **Majored Features**

- User Authentication - MVP
- Character Builder - MVP
- Campaign host - MVP
- Create and edit custom content - MVP
- Data management (CRUD) - MVP
- Role-based access - MVP
- Spell & item libraries - MVP
- Real-time combat system – Nice to have
- Campaign/encounter sharer - Future
- Map integration - Future
- Tabletop – Future

## User Roles

- Admin: Manage the website and fix errors and make sure website runes properly
- Dungeon Master: Host a campaign and has full editing access to this campaign.

They only have access to campaign they created or are given DM states to by the campaign creator.

- Players: can join a campaign but they only have access to character creator and inventories. DM can give them access to other functions if needed.

## Yggdrasil Data Planning

Figma file with wireframes and ERD diagram:

<https://www.figma.com/design/A3dihB10zDKqfJm31JTv3F/D-D-Tabletop?node-id=0-1&t=EZvGLXxwef2RcU9v-1>

## Project Timeline & Workflow

Link to Gantt timeline Monday:

<https://virtualwindow329769.monday.com/boards/2065635336>

## Risks, Challenges & Conclusion

### Technical Risks

#### 1. Feature Overload

Including too many complex features (e.g., character builder, real-time combat, inventory system) may result in incomplete implementation within the 13-week timeframe.

**Mitigation:** Focus on developing the **Minimum Viable Product (MVP)** first — authentication, campaign management, and character creation. More advanced or optional features, such as real-time combat, will be deferred if time becomes limited. Weekly milestones will be used to maintain progress and focus.

## 2. Integration Complexity

Integrating the frontend, backend, and MySQL for multiple features increases the chance of bugs and delays.

**Mitigation:** Implement and test each feature **vertically (full-stack)** before starting the next. This ensures end-to-end functionality and reduces the risk of large-scale integration issues later.

## 3. Real-Time Combat System

Implementing real-time interactions using technologies like Socket.io may be challenging to complete as a solo developer.

**Mitigation:** Begin with a **simpler turn-based system** that uses regular data polling. Introduce real-time syncing only after the core gameplay features are stable and time permits.

## 4. Database Structure Errors

Incorrect database relationships (e.g., linking characters to campaigns) could cause major functionality issues.

**Mitigation:** Use **dummy data** during early development to validate logic and relationships. Conduct structured testing at each stage to ensure the database supports all required interactions.

## Non-Technical Risks

### 1. Time Management

As a solo developer with a limited timeframe, it is easy to fall behind schedule.

**Mitigation:** Follow a detailed **week-by-week project timeline**, setting clear goals and deliverables for each phase to maintain consistent progress.

## 2. Motivation & Burnout

Working independently for an extended period can lead to fatigue or loss of motivation.

**Mitigation:** Maintain motivation by setting achievable weekly milestones, taking short breaks between development phases, and tracking visible progress. Regularly reviewing progress and celebrating small achievements will help sustain focus and energy.

## Conclusion

Yggdrasil was developed to provide a streamlined, accessible, and creative platform for Dungeon Masters and Dungeons & Dragons players. Unlike existing tools such as *D&D Beyond*, which often place restrictions behind paywalls, Yggdrasil removes these barriers to deliver a free and fully customizable experience. It empowers users with the freedom to create and manage every aspect of their campaigns all within one cohesive system.

By adhering to a structured development timeline and focusing on a modern full-stack JavaScript architecture (React.js, Node.js with Express, and MySQL), Yggdrasil ensures scalability, performance, and ease of integration. The project demonstrates how thoughtful planning, focused feature prioritization, and user-centered design can bring a complex system to life within a 13-week solo development period.

Ultimately, Yggdrasil is more than just a digital tool , it is a creative hub built to enhance collaboration, storytelling, and imagination in the world of tabletop gaming.