

A2

by Zandr Badger

Submission date: 06-May-2026 09:00AM (UTC+0100)

Submission ID: 282349517

File name: 50546_Zandr_Badger_A2_731414_303089946.docx (8.76M)

Word count: 3531

Character count: 18876

Assignment 2: Practical & Professional Development

Aleksandr Badger

University of Wales, Trinity Saint David

Swansea, Wales

2308404@student.uwtsd.ac.uk

Table of Contents

Task 1: Practical Development / implementation	4
Q.1: Initial Feedback.....	4
Q.2: Implementation	7
Troubleshooting:	12
Task 2: Enterprise & Innovation	14
Business Name & Type.....	14
Business Strategy	14
Mission Statement.....	14
Objectives	14
Values.....	14
PEST Analysis	15
Case Study: Horizon System	16
What developer tools, methods and best practices could've been used to prevent these flaws?	16
Task 3: Project feedback & reflection.....	17
Practical feedback	17
Project Progress Diary	19
Reflect and document the Project Management knowledge and skills you have gained as a result of completing your project.	19
Reflect and document the Technology & Technical knowledge and skills you have gained as a result of completing your project.	20
Positive Social Impact (PSI)	20
Conclusion	21
Appendix	23
Progress Sheets	23
CPD Evidence.....	25

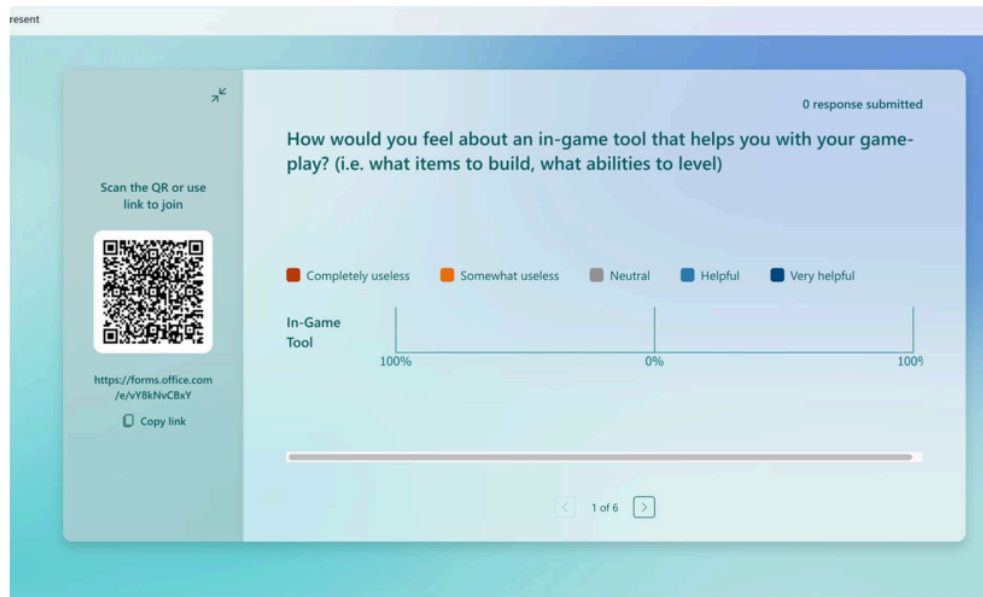
Introduction

BadInfo, is a League of Legends analytics dashboard that uses the Riot Games API to retrieve and display player match history and performance statistics. The system is built using JavaScript, HTML, CSS, and a Rust backend via Tauri to create a desktop application. The aim is to help players improve performance through accessible in-game data visualisation, including win/loss tracking, champion statistics, and match breakdowns. The system follows a modular architecture and integrates external data sources such as Data Dragon for game assets [2].

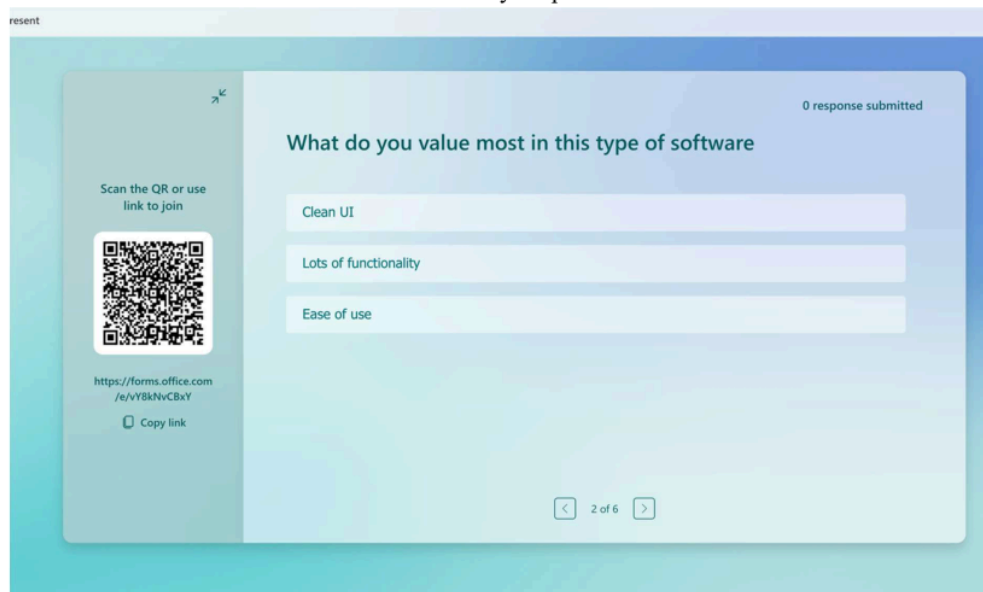
Key technical challenges included API rate limiting, cross-language data transfer between Rust and JavaScript, and structuring complex match data into a usable UI format. These were mitigated through modularisation and structured API handling. The project demonstrates full-stack integration.

Task 1: Practical Development / implementation

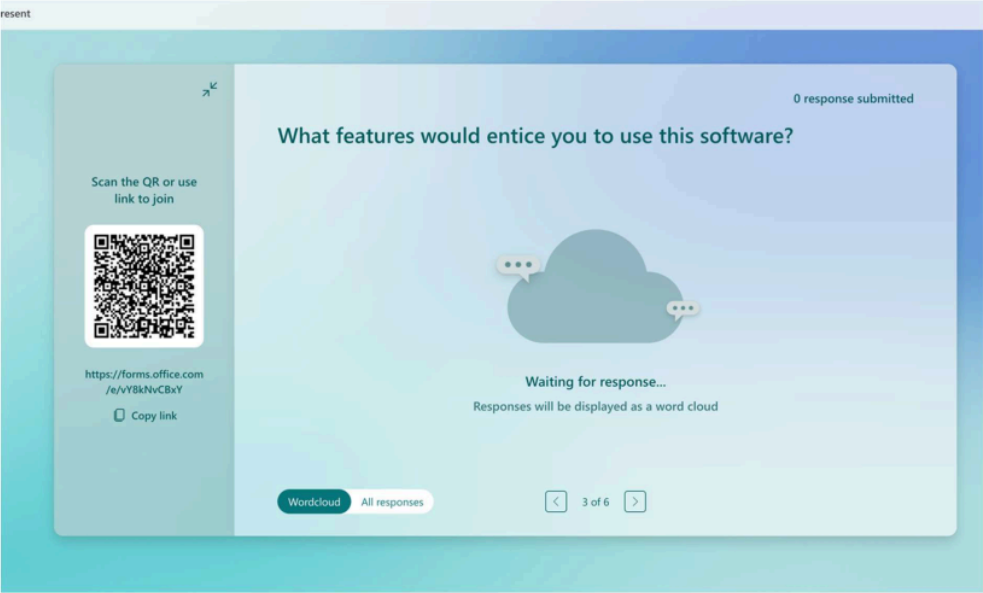
Initial Feedback



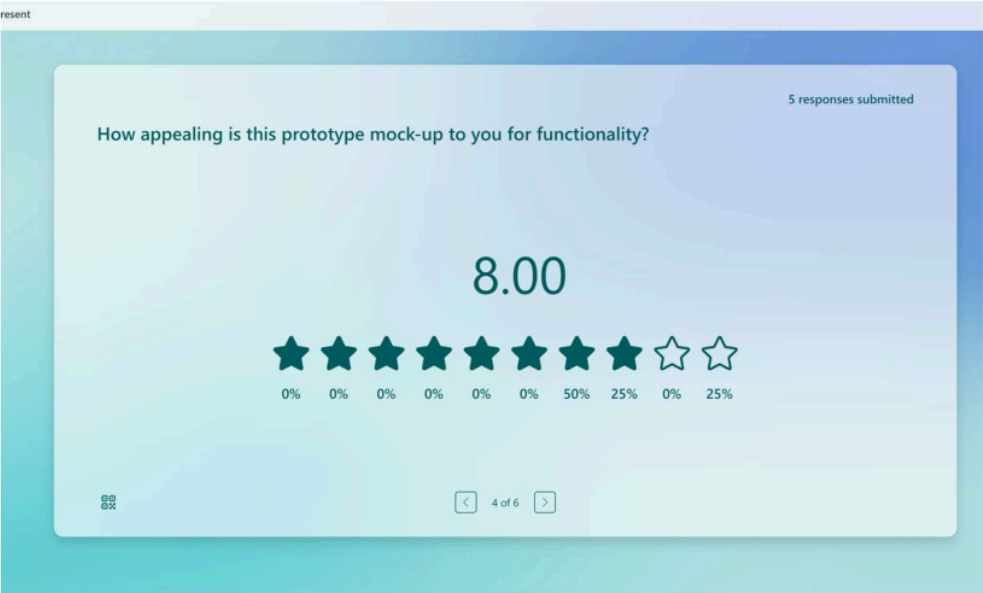
Users felt that an in-game tool that helps with gameplay is helpful with 83.33% believing so, while the other 16.7% believed it would be very helpful.



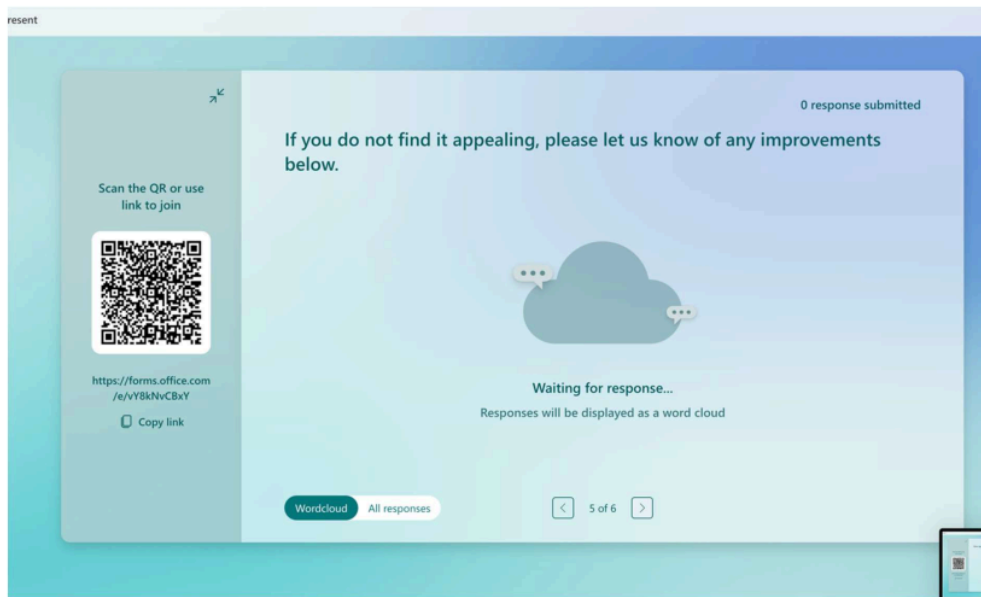
Users valued ease of use most, followed by clean UI and then a multitude of functionality, with functionality being last by a significant margin.



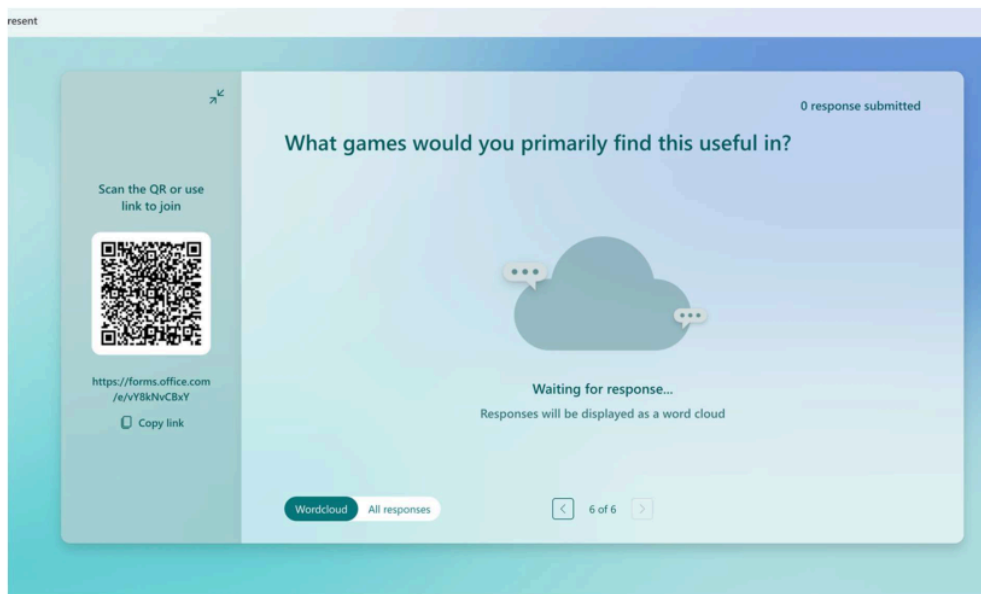
Feedback suggested that the software should focus on helping the user understand and improve at the game, while keeping system resource allocation low.



Users generally agree that the mock-up design is very appealing in terms of functionality, with an average of an 8/10.



There is no relevant feedback for this section.



Feedback suggested that this software would be most useful in MOBAs, Competitive games and FPS shooters, specifically with many suggesting League of Legends.

Q.2: Implementation

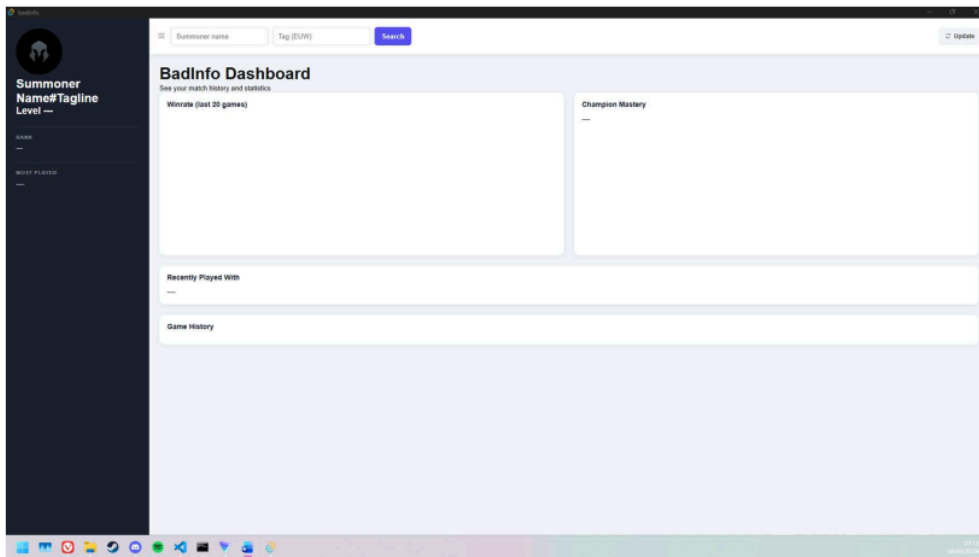


Figure 1: Home Page

This is the basic home page of the project, as seen it has a very basic GUI, suited for speed and efficiency, over looks.

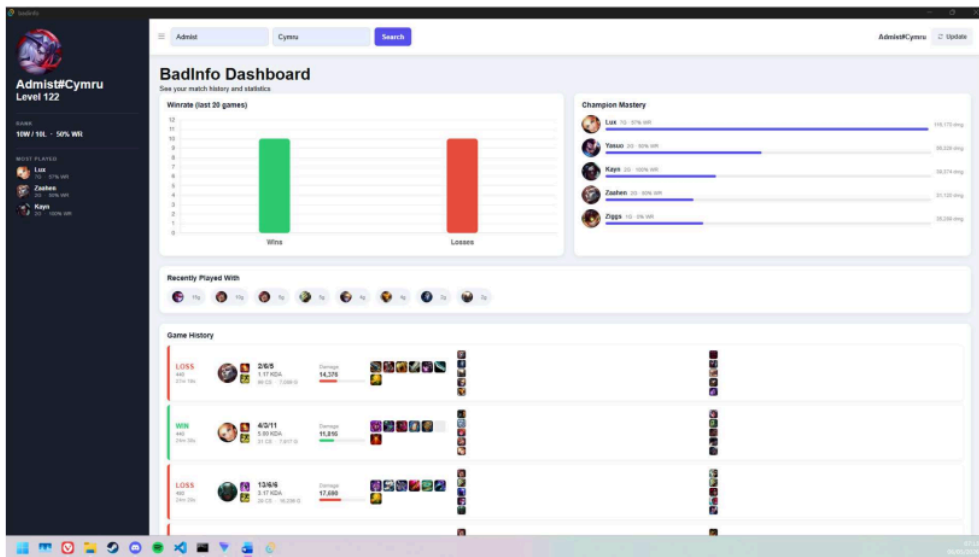


Figure 2: User Page

When a valid SummonerId and TagLine are input into the search bar at the top, it will send a request to the Riot API for over 21 requests of information. The first of this is by using the

input data to return the PUUID of the account, which allows us to access more data, the and obviously their game_name and tag_line.

This is then passed into a struct called RiotAccount and the data is then passed from Rust into JavaScript, and applied to the web page, where we can see the name and tagline on the sidebar.

```
2
3  #[derive(Serialize, Deserialize)]
4  pub struct RiotAccount {
5      puuid: String,
6
7      #[serde(rename = "gameName")]
8      game_name: String,
9
10     #[serde(rename = "tagLine")]
11     tag_line: String,
12 }
13
```

Figure 3: RiotAccount struct

```
> pub struct MatchEntry { ...
#[tauri::command]
async fn get_summoner(name: String, tag_line: String) -> Result<RiotAccount, String> {
    let api_key: &str = "RGAPI-b2167694-4f0c-43d6-9d17-000325e291f7";

    let url: String = format!(
        "https://europe.api.riotgames.com/riot/account/v1/accounts/by-riot-id/{}/{}",
        name, tag_line
    );

    let client: Client = request::Client::new();

    let res: Response = client
        .get(&url)
        .header("X-Riot-Token", api_key)
        .send()
        .await
        .map_err(|e: Error| e.to_string());

    let status: StatusCode = res.status();
    let body: String = res.text().await.map_err(|e: Error| e.to_string());

    println!("STATUS: {:?}", status);
    println!("BODY: {}", body);

    if !status.is_success() {
        return Err(format!("Riot API error: {}", body));
    }

    let account: RiotAccount = serde_json::from_str(&body)
        .map_err(|e: Error| e.to_string());

    Ok(account)
} async fn get_summoner
```

Figure 4: MatchEntry struct

This is what passes the RiotAccount from Rust into JS, you can see the call to the Riot API in let url: and by using a library called “serde”, this can be converted to a .json and transferred to Rust (although Tauri does have built in translation too) [1][4].

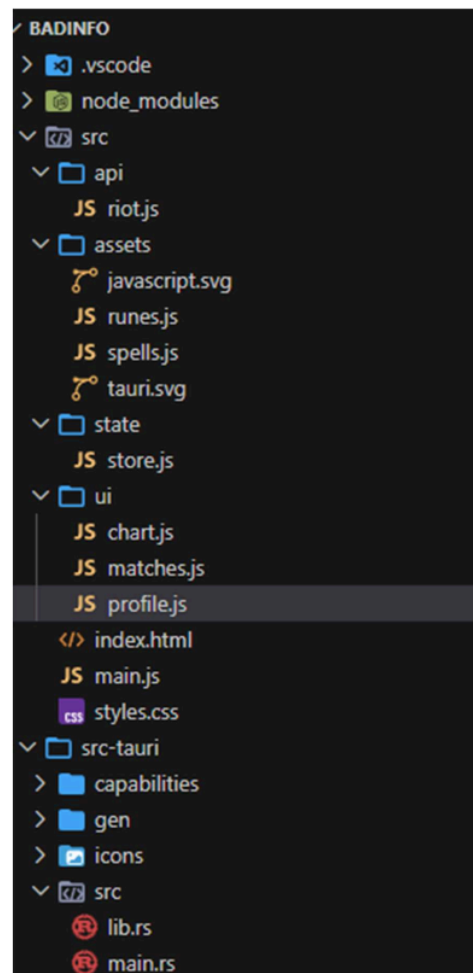
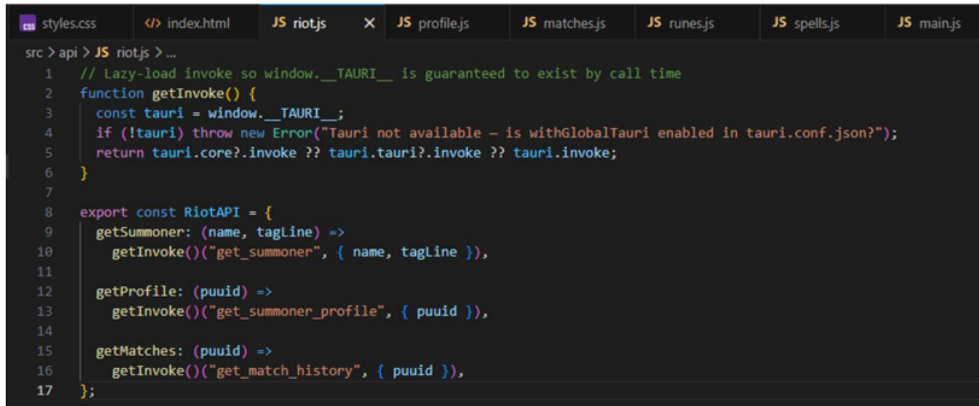


Figure 5: File Path

During this project, the choice was made to separate functionality into smaller JS files, both for readability and efficiency, as this follows object-oriented practices via modularisation.



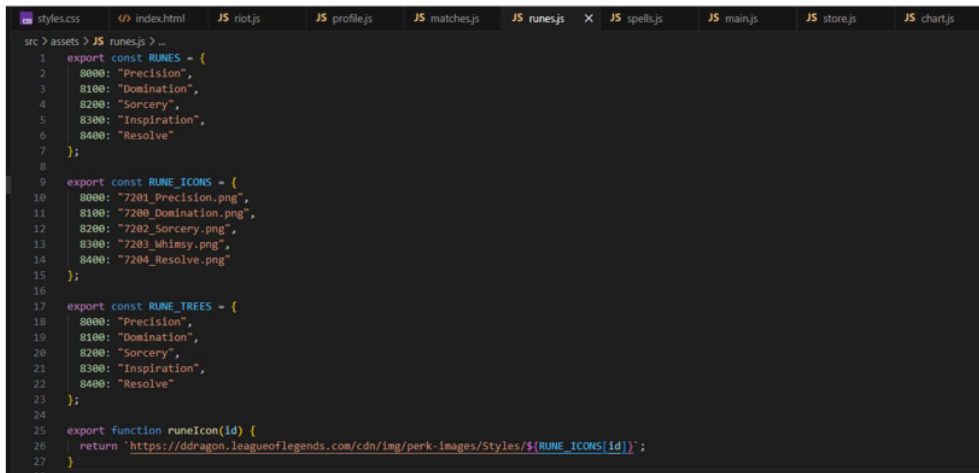
```

src > api > JS riot.js > ...
1 // Lazy-load invoke so window.__TAURI__ is guaranteed to exist by call time
2 function getInvoke() {
3   const tauri = window.__TAURI__;
4   if (!tauri) throw new Error("Tauri not available - is withGlobalTauri enabled in tauri.conf.json?");
5   return tauri.core?.invoke ?? tauri.tauri?.invoke ?? tauri.invoke;
6 }
7
8 export const RiotAPI = {
9   getSummoner: (name, tagLine) =>
10     getInvoke()("get_summoner", { name, tagLine }),
11
12   getProfile: (puuid) =>
13     getInvoke()("get_summoner_profile", { puuid }),
14
15   getMatches: (puuid) =>
16     getInvoke()("get_match_history", { puuid }),
17 };

```

Figure 6: riot.js

The riot.js file handles all the API requests in one call, helping to reduce the number of calls made per second, as the maximum rate is 20 calls per second [1], without this function calls to the API were doubled, reaching 42 per second and timing out when attempting to load the match history.



```

src > assets > JS runes.js > ...
1 export const RUNES = {
2   8000: "Precision",
3   8100: "Domination",
4   8200: "Sorcery",
5   8300: "Inspiration",
6   8400: "Resolve"
7 };
8
9 export const RUNE_ICONS = {
10   8000: "7201_Precision.png",
11   8100: "7200_Domination.png",
12   8200: "7202_Sorcery.png",
13   8300: "7203_Whimsy.png",
14   8400: "7204_Resolve.png"
15 };
16
17 export const RUNE_TREES = {
18   8000: "Precision",
19   8100: "Domination",
20   8200: "Sorcery",
21   8300: "Inspiration",
22   8400: "Resolve"
23 };
24
25 export function runeIcon(id) {
26   return `https://ddragon.leagueoflegends.com/cdn/img/perk-images/Styles/${RUNE_ICONS[id]}`;
27 }
28

```

Figure 7: runes.js

There were also smaller files made such as runes.js, spells.js and profile.js, whose purpose was to hold data for icons, acting as a small translation layer between the program and external APIs where they are held, which in this case were Community Dragon and Data Dragon [2], which are collections of League data available to the public and via calls. In this case they held icons for summoner spells, champions, etc.

```

src > ui > JS chart.js > ...
1  let winrateChart;
2
3  export function renderWinrate(win) {
4      const ctx = document.getElementById("winrate-chart");
5      if (!ctx) return;
6
7      if (winrateChart) winrateChart.destroy();
8
9      winrateChart = new Chart(ctx.getContext("2d"), {
10         type: "bar",
11         data: {
12             labels: ["Wins", "Losses"],
13             datasets: [{
14                 data: [win.wins, win.losses],
15                 backgroundColor: ["#2ecc71", "#e74c3c"],
16                 borderRadius: 6,
17                 bordersSkipped: false,
18                 barThickness: 60
19             }]
20         },
21         options: {
22             responsive: true,
23             maintainAspectRatio: false,
24             plugins: {
25                 legend: { display: false },
26                 tooltip: {
27                     callbacks: {
28                         label: ctx => {
29                             const total = win.wins + win.losses;
30                             const pct = total > 0 ? Math.round((ctx.raw / total) * 100) : 0;
31                             return ` ${ctx.raw} games (${pct}%)`;
32                         }
33                     }
34                 }
35             },
36             scales: {
37                 x: {
38                     grid: { display: false },
39                     ticks: { font: { size: 13, weight: "700" } }
40                 },
41                 y: {
42                     beginAtZero: true,
43                     max: Math.max(win.wins, win.losses, 1) + 2,
44                     grid: { color: "rgba(0,0,0,0.05)" },
45                     ticks: { stepSize: 1, font: { size: 11 } }
46                 }
47             }
48         }
49     }
50 }

```

Figure 8: chart.js

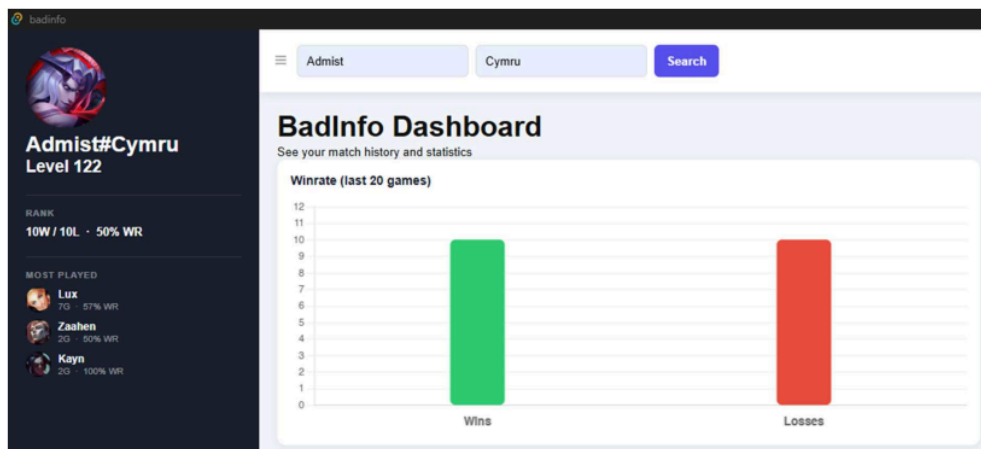


Figure 9: Chart

To create the chart, we had to simply count the number of wins and losses using the match history call, which maxes out at 20 for API call reasons, and then simply searches for win or loss and then adds one to its respective node. This was implemented using a JavaScript charting library (Chart.js) [3].

Troubleshooting:

One struggle faced during this project was to have the ability to display the summoner names of those in the same game as the user and then be able to select their icons/names and to visit their information dashboard.

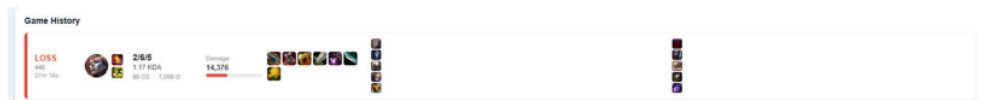


Figure 10: P Match History Bar

As seen above, it displays the champion icon and when selected it will redirect to the players page, however the player's username will not appear due to an issue within the code, however it does redirect it to the correct page.

If we click the second portrait (Kindred), then it will redirect the page to an "Unknown" summoner and display their information, however this is actually the correct summoner as proven by other websites.

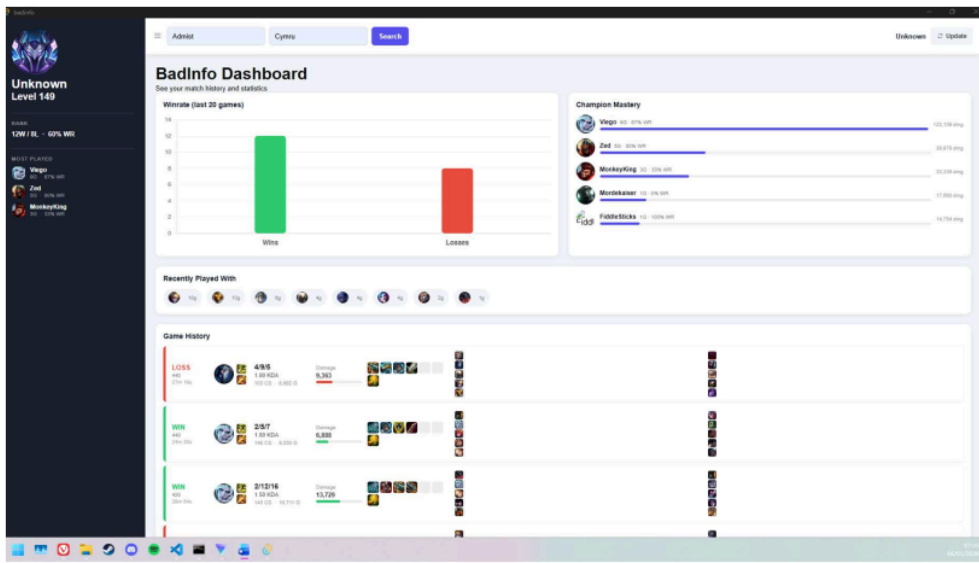


Figure 11: "Unknown" Dashboard



Figure 12: External Website to check proof of concept

As seen above, the games played and summoner icon are both accurate, despite the issue with the displayed name, showing that this is an error with the project, not the API.

Unfortunately, this issue was not able to be resolved by the product finish date, despite attempts to manually override font colour, multiple different ways to place user names and as seen above, their PUUID is working correctly and linked to their profile picture.

Task 2: Enterprise & Innovation

Business Name & Type

The business name I have chosen for this is BadInfo, which is a mix of my surname and information, which this project provides. The name is also a pun, as the information provided is useful.

This business would be a private limited company, as this would allow for research and production without the influence of outside sources, such as majority stakeholders. This will also protect the individual from due to its limited liability.

Business Strategy

Mission Statement

To display current in-game information to the user in order to help them manage their decisions more easily, by showing them the statistically best option per win rate percentage, free of charge and open-source to the community.

Objectives

To display in-game information to the user, such as next item to build, your best champion via your win rate.

To show a user's previous match history, alongside their other user data such as win rate, most played champion, etc.

Help new players to improve at the game by helping them know what to do and when.

Values

Helping Others: Our core values are to helping others, ensuring that they can become the best versions of themselves.

Publicity & Accessibility: We also heavily value and focus on publicity and accessibility, by making all our software and resources free and open source, so users know exactly what is going on under the hood, as we believe that everyone should have the same opportunities.

Open-Communication: Communication is also another core value, as we require feedback to improve our software and to improve the user experience based on their personal experiences [9].

PEST Analysis

A PEST analysis evaluates the external factors that may influence BadInfo, a web-based game analytics platform similar to Blitz.gg and OP.GG.

Political factors mainly relate to regulation and data governance. As a UK-based private limited company, BadInfo must comply with the UK General Data Protection Regulation and the Data Protection Act 2018, ensuring that any user data collected, such as match history or performance statistics, is handled lawfully and transparently. The business must also follow the terms of service of game developers like Riot Games, as access to their APIs is essential. Regulatory changes or stricter digital policies could increase compliance costs or restrict access to data.

Economic factors include operational costs and funding challenges. Although BadInfo is free and open source, it still requires funding for servers, maintenance, and development. Without a direct revenue stream, the company may rely on donations or optional premium features. Competition from established services like U.GG increases pressure to deliver accurate, real-time data. Economic conditions such as rising cloud hosting costs could impact sustainability, while growth in the gaming industry may expand the potential user base.

Social factors are driven by gaming culture and user behaviour. The increasing popularity of competitive gaming and content platforms such as Twitch has created strong demand for performance tracking tools. BadInfo supports new players by offering guidance based on statistics, aligning with its value of helping others. However, there are ethical concerns, including potential over-reliance on data, which may reduce creativity in gameplay. There is also a risk that player statistics could be misused, contributing to toxicity within gaming communities.

Technological factors are critical to the platform's success. BadInfo depends on real-time data integration, scalable infrastructure, and continuous updates to reflect changes in game mechanics. Reliance on external APIs, especially from Riot Games, presents a risk if access is limited. Cybersecurity is also important to protect user data. Advances in data analytics and machine learning could improve recommendations, enhancing user experience [10].

Legal and ethical considerations include ensuring data privacy, respecting intellectual property, and maintaining transparency through open-source practices. Ethically, the platform must present accurate information and avoid bias in its recommendations to maintain user trust [11].

Case Study: Horizon System

The British Post Office scandal involved the Post Office pursuing thousands of innocent subpostmasters for apparent financial shortfalls caused by faults in their new accounting software system called Horizon. This caused more than 900 subpostmasters to be wrongfully accused of theft, fraud and false accounting based on faulty Horizon data, with about 700 of these prosecutions carried out by the Post Office. Others were prosecuted but not convicted, forced to cover the costs with their own money, or had their contracts terminated. This resulted in stress, illness and family breakdowns, and even linked to at least thirteen suicides.

What developer tools, methods and best practices could've been used to prevent these flaws?

Fujitsu could have rolled out a trial period for the product, allowing for a small amount of users to try and test the software, and then compared the data created by Horizon to the old paper-based system, which could have allowed for any irregularities to be observed and reported.

Fujitsu also could have completed more quality testing, to ensure that their software was accurately reporting data, instead of assuming they were correct. Working alongside the user to help develop this, by using AGILE protocols, would have also helped to prevent these flaws.

Task 3: Project feedback & reflection

Practical feedback

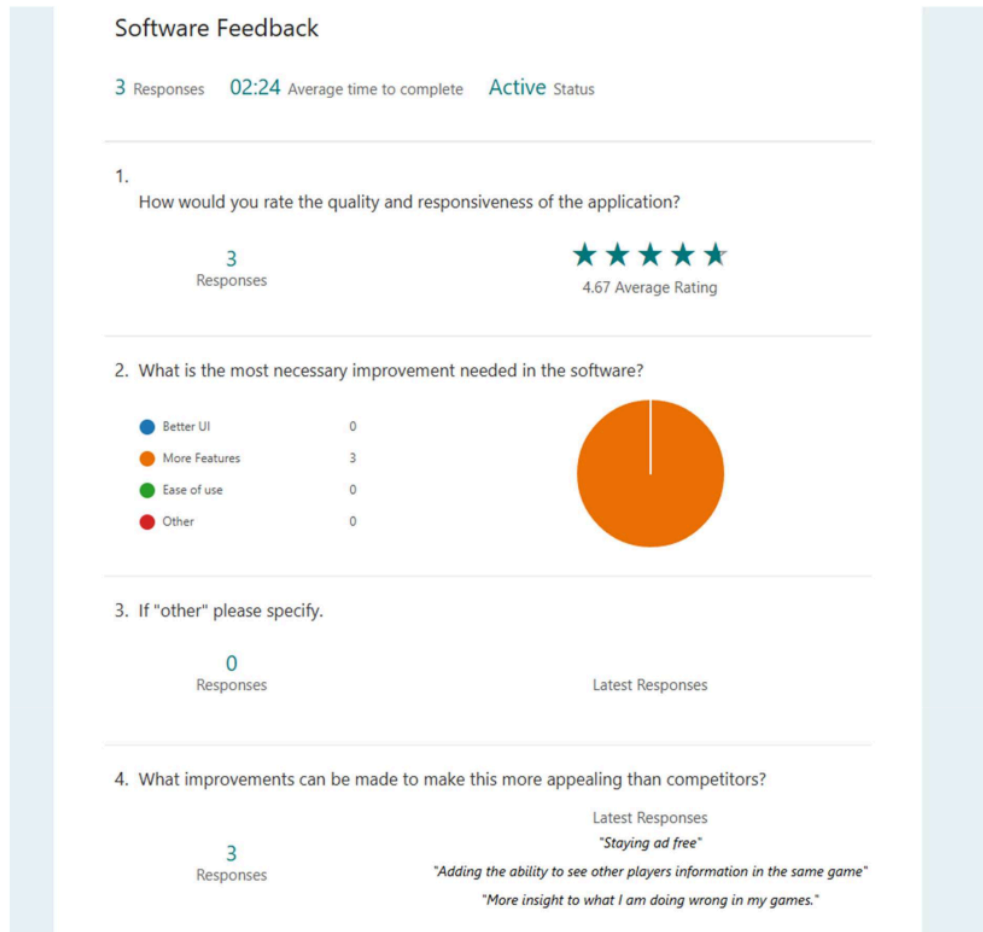


Figure 13: Results 1

Based on user feedback, the software is certainly lacking in features and is unanimously the most necessary improvement for the software, as all users stated that the quality and responsiveness was high.

As a result of the feedback, it was attempted to add in more information on the Game History, such as gold, CS and damage to see where you are going wrong. Furthermore, the ability to view other players in the same games was interesting, which is a change I would make to the project to improve its toolkit.

5. How easy is it to view user information using the software?

Very Easy Easy Okay Good Very Good



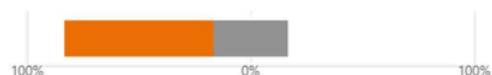
6. How well does this software integrate with the game?

Not very well Somewhat well Very well Extremely well Extremely not well



7. How likely are you to continue using this software in the future?

Very likely Somewhat likely Neither likely nor unlikely Somewhat unlikely Very unlikely



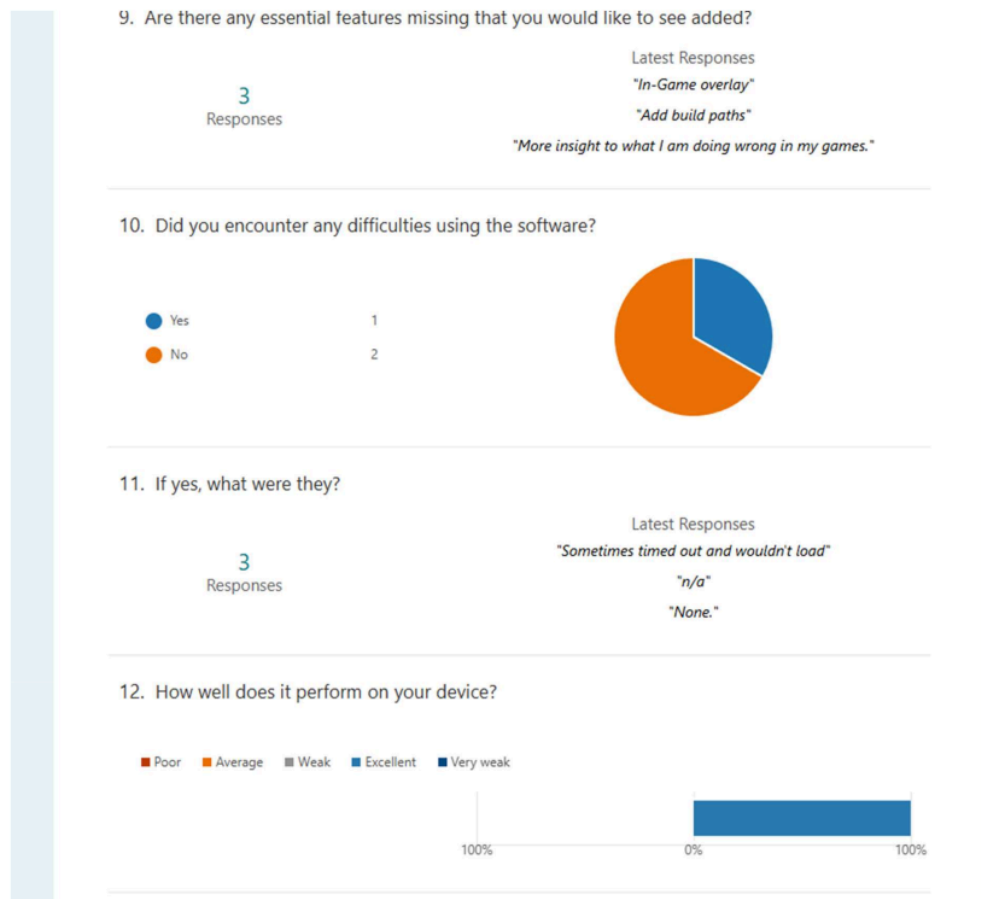
8. Please state why.

3
Responses

Latest Responses

"It just has the information I want without the spyware from other co..."
"There isn't enough for me to use this over more popular options"
"Because I am interested in my game feedback, but not enough to use..."

All users stated that there is not enough information to use at this point in time of the project, therefore I will scope out competitors such as OP.GG and Blitz.GG to see what information they are display and how to implement this myself.



Project Progress Diary

While management of this project seemed to start smoothly, it became clear that it was victim to scope creep, as the ability to have an in-game overlay that also display build paths would take much longer than was possible in the given time. Despite this, I felt that there was significant progress being made during each weekly progress sheet and that the time I invested in this project has shown itself.

If I were to complete this project again, I would instead focus on creating a minimum viable product first (MVP), as this allows for the software to be created with only the most necessary features and from there, we can use the remaining time to add more features, helping to avoid my flaw of scope creep.

Reflect and document the Project Management knowledge and skills you have gained as a result of completing your project.

As a result of completing this project, I have gained the skill to effectively plan, monitor and review timelines and work accurately, coinciding with the ability to manage resources.

Furthermore, this has taught me how to deal with issues such as scope creep, as the timeline for the development was relatively short, with minimal resources, therefore the development had to avoid any unnecessary features, allowing for smooth development of the project.

Reflect and document the Technology & Technical knowledge and skills you have gained as a result of completing your project.

This project has allowed me to develop my programming knowledge further, as research led me to discover new technology frameworks, such as Tauri and Electron [4], which are containers that allow developers to create a desktop app using web technologies such as HTML, JavaScript and CSS [5]. This has also allowed me to expand my web development skillset further, while the backend allowed me to learn how to interact with external APIs, such as how to call them, store information, limit accesses and more [1]. The project also strengthened my JavaScript and Rust skills, as this is what the backend was developed in.

Positive Social Impact (PSI)

This project aims to assist both new and old gamers in League of Legends, who are looking for ways to improve and enhance their knowledge of the game. It creates quality of life features, by providing information such as the highest win rate item build paths for the current season/meta and past performances in games, so they can see how they went wrong, for example picking a champion they have a low win rate with.

Furthermore, this benefits others as while this software is typically “free”, they collect user data and sell it to advertising companies, or put advertisements in their software to create revenue, this project does not include either of these and aims to be a non-profit open-source software for the community to use.

Conclusion

BadInfo provides a functional League of Legends analytics dashboard that meets most of its core objectives, those being retrieving player data, processing match history, and visualising performance statistics. The system demonstrates effective API integration and modular full-stack design using Rust and JavaScript.

However, limitations remain in incomplete UI features, specifically player identity display and advanced analytics, which stem from API complexity and scope constraints.

Overall, the project provides a working MVP with clear potential for expansion into a more advanced analytics platform, through improved data modelling and enhanced UI/UX design.

References

- [1] Riot Games, “Riot Developer Portal,” 2025. [Online]. Available: <https://developer.riotgames.com/>
- [2] Riot Games, “Data Dragon (Static Game Data),” 2025. [Online]. Available: <https://developer.riotgames.com/docs/lol#data-dragon>
- [3] Chart.js Contributors, “Chart.js Documentation,” 2025. [Online]. Available: <https://www.chartjs.org/docs/latest/>
- [4] Tauri Apps, “Tauri Framework Documentation,” 2025. [Online]. Available: <https://tauri.app/v1/guides/>
- [5] Mozilla Developer Network, “JavaScript Guide,” 2025. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>
- [6] Fireship, “Tauri in 100 Seconds,” YouTube, Jul. 2022. [Online]. Available: <https://www.youtube.com/watch?v=9zbl4qmkSbA>
- [7] Traversy Media, “Rust Crash Course,” YouTube, Mar. 2023. [Online]. Available: <https://www.youtube.com/watch?v=4WvT2cmuZ5M>
- [8] Web Dev Simplified, “JavaScript Fetch API Tutorial,” YouTube, Jan. 2021. [Online]. Available: <https://www.youtube.com/watch?v=jkzq9j5yeT8>
- [9] H. Mintzberg, “The strategy concept I: Five Ps for strategy,” California Management Review, vol. 30, no. 1, pp. 11–24, 1987.
- [10] F. Ricci, L. Rokach, and B. Shapira, Recommender Systems Handbook. Springer, 2015.
- [11] F. J. Aguilar, Scanning the Business Environment. New York, NY, USA: Macmillan, 1967.

Appendix

Progress Sheets

Project progress form

Name: Zandr Badger (2308404)

Programme: Applied Computing

Project Aim: An application that will take information from the video game "League of Legends" using the Riot Games API, in order to display helpful information to the end user, to help them progress in skill and to win the match, ensuring that it follows the appropriate guidelines, so it can be approved and not trigger Riot's kernel-level anti-cheat. This will take form in game as an overlay, and outside of the game as an external application GUI. This will be achieved by using JavaScript/Python to make API requests.

Project Objectives:

1. Request match/account data using the Riot Games API.
2. Develop a functional and appealing GUI.
3. Suggest potential improvement methods to the current user based on champion, matchups, etc.

Weekly Project progress forms:

Date: 11/03/2026

Progress:

- Made a Riot Games Developer API account.
- Decided upon Jupyter as an IDE (python) as it allows for real-time updates and it is free and web based.
- Generated an API key and made simple requests, such as account data, match data.
- Began to mark-out the MVP for this project in the code, by storing data and beginning to create loops.
- Created a survey and received feedback from a target audience.

Actions:

State priority actions to undertake over the next week below.

1. Learn how to request less API requests as there is a 200-limit max per 2 minutes.
2. Integrate user input to allow for a basic CLI tool.

Next review date: 11/03/2026.

Weekly Project progress forms:

Date: 18/03/2026

Progress:

- Integrated user input and now allows for a basic search function on any user in a CLI.
- Added manual blockers that look for the rate limit error code to add in a wait time so that the request limit isn't reached, so it continues smoothly.

Actions:

State priority actions to undertake over the next week below.

3. Learn how to use this script for a GUI/Web app
4. Start making a very basic frontend for testing.

Next review date: 25/03/2026.

Weekly Project progress forms:

Date: 25/03/2026

Progress:

- Decided on a GUI that works for my project, it will be an administrator dashboard layout, so that users can view analytics immediately.
- Started creating the frontend using Tauri, a container that allows developers to create desktop applications using web development.

Actions:

State priority actions to undertake over the next week below.

5. Continue on creating the frontend for the project

Next review date: 1/04/2026.

Weekly Project progress forms:

Date: 1/04/2026

Progress:

- A frontend has been developed, subject to change as necessary
- Learnt the basics Rust and began development of backend, by making basic calls to the Riot API

Actions:

State priority actions to undertake over the next week below.

6. Create a bar chart to show win/loss ratio
7. Add a search bar to search for a user

Next review date: 8/04/2026.

Weekly Project progress forms:

Date: 8/04/2026

Progress:

- Added the search bar and able to search for users using their summonerName and tagLine
- Created a bar chart that when a user is accessed, does maths in JavaScript to generate a bar chart

Actions:

State priority actions to undertake over the next week below.

8. Continue creating the backend for my project

Next review date: 15/04/2026.

Weekly Project progress forms:

Date: 15/04/2026

Progress:

- Created calls to the Riot API to gather Player data, such as their PUUID, which allows me to make calls to view their match history, summoner icon and more

- Linked these calls to the frontend using JavaScript
-

Actions:

State priority actions to undertake over the next week below.

9. Add the UI for Match History
10. Tweak the UI to be more visually appealing

Next review date: 22/04/2026.

Weekly Project progress forms:

Date: 22/04/2026

Progress:

- Changed the UI to add a sidebar that can be hidden, that shows user information such as name, icon, level and most played champs
- Added basic match history information such as champion played and kill/death/assist ratio

Actions:

State priority actions to undertake over the next week below.

11. Finish off by adding summoner spell icons, an update button, damage bar chart, item build path, CS, gold, overall k/d/a percentage, time in match and all people in that match.

Next review date: 29/04/2026.

Weekly Project progress forms:

Date: 6/04/2026

Progress:

- Finished adding the final touches to the frontend, so that Game History now shows all possible details, however I could not get the enemy summoner names to appear, but they can still be accessed by clicking on their champion icon
- Added a recently played with bar, however same issues as above, so instead it shows how many games.

CPD Evidence

The learning tool I engaged with was LinkedIn learning, choosing to learn how to program in JavaScript, as this is necessary for the backend of my software, as I am using a container that effectively converts a web application into a desktop application.

Course: LS - Network Op... Mail - Zandr Badger (230) SEPT2026-Team3 - Miro Teams and Channels | M... The Riot API with Python JavaScript objects: The c

www.linkedin.com/learning/javascript-essential-training/javascript-objects-the-code-version

advanced javascript

Me EN Restart Required

Learning

Contents

- JavaScript language basics 5m 37s video
- Learning JavaScript backward 3m video
- Chapter Quiz 7 questions
- 2. Get Up and Running with JS
 - JavaScript in an HTML document 3m video
 - JavaScript as an external file 2m 59s video
 - Modern JavaScript loading 4m 14s video
 - JavaScript modules 4m 18s video
 - Chapter Quiz 4 questions
- 3. Objects
 - Objects: A practical introduction 4m 53s video
 - JavaScript objects: The code version 2m 57s video

```
name: "Everyday Backpack",
volume: 30,
color: "grey",
pocketNum: 15,
strapLength: {
  left: 26,
  right: 26,
},
lidOpen: false,
toggleLid: function (lidStatus) { ← Methods are properties
  this.lidOpen = lidStatus;           containing functions.
},
};
```

whatever function is contained within that method.

Overview Notebook Transcript

Instructor

Morten Rand-Hendriksen
Principal Staff Instructor,
Speaker, Web Designer, and
Software Developer

Related to this course

- Exercise Files - Show all
- Certificates - Show all

Exercise Files on GitHub

Access the repository to easily follow along, practice, and experiment with your own tools.

Course: LS - Network Op... Mail - Zandr Badger (230) SEPT2026-Team3 - Miro Teams and Channels | M... The Riot API with Python Challenge Intro: Create a

www.linkedin.com/learning/javascript-essential-training/challenge-intro-create-a-new-object-type

advanced javascript

Me EN Restart Required

Learning

Contents

- Classes: Object blueprints 6m 32s video
- Object constructors 2m 21s video
- Extending classes 4m test
- Practice: Build a new object with a constructor 1m 13s video
- Global objects 7m 47s video
- Challenge intro: Create a new object type 2m 5s video
- <> Code Challenge: Create a new object type 20m practice
- Solution: Create a new object type 3m 16s video
- Chapter Quiz 15 questions
- 4. Sidebar: String Output
 - Mix text and variables with template literals 7m 21s video
 - Traditional string output

JavaScript Essential Training

Challenge intro: Create a new object type 383,700

```
const bookTitle = "Alice's Adventures in ...";
const bookAuthor = "Lewis Carroll";
const bookPubYear = 1865;
const bookISBN = 9798369283415;

const book = new Book(
  bookTitle,
  bookAuthor,
  bookISBN,
  bookPubYear
);
```

1:07 / 2:05

Overview Notebook

Instructor

Morten Rand-Hendriksen
Principal Staff Instructor, Speaker, Web Designer, and
Software Developer

Related to this course

- Exercise Files - Show all
- Certificates - Show all

Course details

6h 14m · Beginner · Updated: 6/7/2024

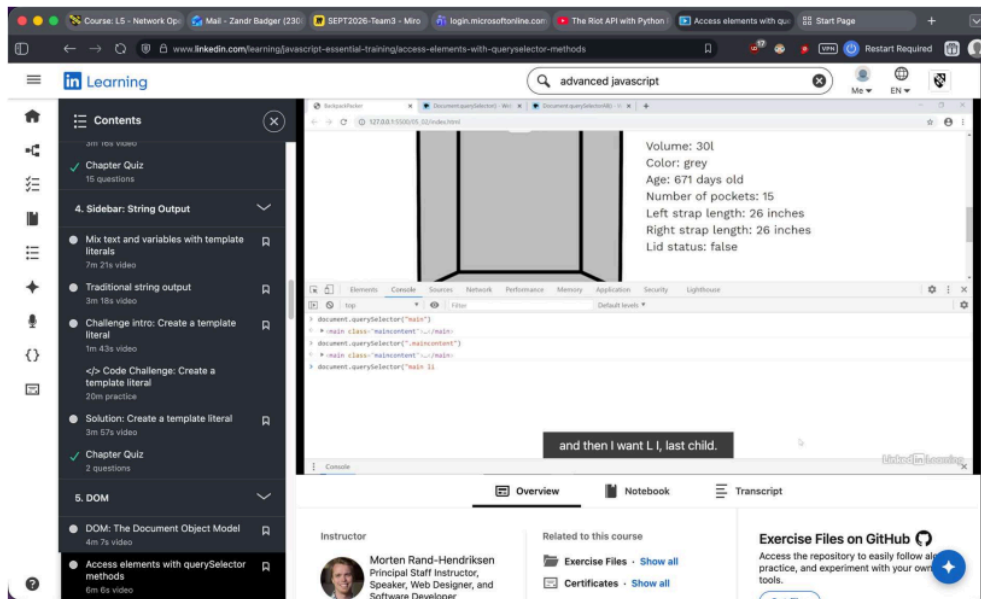
Challenge intro: Create a new object type

Key takeaways for this video

How does this apply to me?

Ask a question

This is powered by AI and may make mistakes. Your Learning data is managed by your organization. Learn more



Program Link

<https://github.com/Aexurr/BadInfo>

FINAL GRADE

GENERAL COMMENTS

92/100

		Marks award ed
Total		92
1. Engagement with Literature Skills 10		
+ Some good use of citations included.		10
+ Well-formatted reference list.		
2. Knowledge and Understanding Skills 5, 10		

+ Tech Start-up name, Business type (Legal classification). – No logo.

– Mission & Objectives relate to the game and should relate to the Business. - Funding options not considered.

T2 Q2:

+ **PEST analysis** summary includes some relevant considerations for each category and documents some key professional legal & ethical considerations. Good. Supported by some relevant citations.

3. Cognitive and Intellectual

Skills 5, 5, 5, 5

T3 Q1: + Data collection method for practical feedback designed, and some feedback acquired and considered. 19

- All report sections/task answers should be introduced.

T3 Q2: + Project management reflection (including review of weekly progress sheets) documented.

T3 Q3: + Skill development reflection documented.

T3 Q3: + Project's positive impact on society is clearly described.

Good.

4. Practical Application Skills 10, 30, 10

T1 Q1: + Data collection method for initial feedback designed, and some feedback acquired and considered. 47

- All report sections/task answers should be introduced.

T1 Q2: + key features of the **implementation** process and practical output have been clearly documented.

Progress sheets (Appendix): + Documentation of weekly progress sheets.

5. Transferable Skills for Life and Professional Practice 5

. 4
+ Report presentation structure and format quite good.

- Each report section & the ToC should be numbered.

- Inclusion of: Case Study: Horizon System? & What developer tools, methods and best practices could've been used to prevent these flaws?, could be included as an appendix item.

.
+ Title page & Table of contents.

.
+ Introduction & Conclusion.

.
+ Tasks complete.

.
+ Appendix: CPD course.

Well done. Very good effort towards the implementation of the prototype with great scope for future development and enhancement. 👍

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

