**Technical leadership Candidate Task**

Strategic Planning and Team, Architecture, and

Tech Stack Evaluation

By

Yoganand Aiyadurai

Version 1.0

Dated: 14th August 2024

Contents

[**1.** **Introduction** 4](#_Toc174526390)

[2. **Team Evaluation and Restructuring** 4](#_Toc174526391)

[2.1 Comprehensive evaluation 4](#_Toc174526392)

[2.2 Proposed changes 4](#_Toc174526393)

[2.3 Restructuring 5](#_Toc174526394)

[2.4 Timeline and Implementation 5](#_Toc174526395)

[2.5 Risks and Mitigation 5](#_Toc174526396)

[2.6 Potential challenges 5](#_Toc174526397)

[Figure 1 Overview of the proposed structure 6](#_Toc174526398)

[2.1 Security Engineer (1) *– Shared resource* 7](#_Toc174526399)

[2.2 Game Mathematician (1) *– Shared resource* 7](#_Toc174526400)

[2.3 Front-End Engineers (2/3) 7](#_Toc174526401)

[2.4 Back-End Engineers (2/3) 7](#_Toc174526402)

[2.5 Blockchain expert developer (1) *– Shared resource* 7](#_Toc174526403)

[2.6 Mobile Developers (2) 7](#_Toc174526404)

[2.7 DevOps Engineer (2) *– Shared resource* 8](#_Toc174526405)

[2.8 QA Engineer (2) 8](#_Toc174526406)

[2.9 Product Manager (1) 8](#_Toc174526407)

[2.10 Data Analyst (1) 8](#_Toc174526408)

[2.11 Game Designer (1) 8](#_Toc174526409)

[2.12 IT Manager (1) 8](#_Toc174526410)

[2.13 Support, Maintenance and Monitoring (3/2) *– Shared resource* 9](#_Toc174526411)

[2.14 Design Team – Shared resource 9](#_Toc174526412)

[2.15 Research and Development Team *– Shared resource* 9](#_Toc174526413)

[2.16 Tech Lead *– Shared resource* 9](#_Toc174526414)

[3. Architecture Assessment and Improvements 9](#_Toc174526415)

[3.1 Codebase and Architecture Review: 9](#_Toc174526416)

[Figure 2 Architecture Assessments and Improvements 11](#_Toc174526417)

[3.2 Security Assessment: 12](#_Toc174526418)

[3.3 Infrastructure and Hosting Review: 12](#_Toc174526419)

[3.4 DevOps and CI/CD Pipelines: 12](#_Toc174526420)

[4. Tech Stack Review and Modernization 12](#_Toc174526421)

[4.1 Stack review 13](#_Toc174526422)

[4.2 Challenges 13](#_Toc174526423)

[4.3 Industry position: 14](#_Toc174526424)

[4.4 Modernization 14](#_Toc174526425)

[4.5 C++, Qt comparison with other popular languages 15](#_Toc174526426)

[4.6 Talent 15](#_Toc174526427)

[5. Roadmap and implementation plan 16](#_Toc174526428)

# **Introduction**

This document explains in details, elaboration of the ideas regarding the case study presentation of the findings and proposals.

# **Team Evaluation and Restructuring**

Given the context of managing two distinct products with different platforms and technologies, the initial team hires should focus on critical areas that ensure the ***security***, ***stability, scalability, maintainability, continuous improvement*** ***and portability*** of both products.

Please note that the team restructure is dependent on the business priorities set by the stakeholders, which may be subject to change.

By evaluating the current team structure and implementing targeted changes, we can create a more agile, collaborative, and high-performing technology team. This will enable us to deliver innovative and high-quality poker and casino games while fostering a positive and motivating work environment. Restructuring should be an ongoing process. Continuous feedback, evaluation, and adaptation are crucial for building a successful team that can thrive in the ever-changing technology landscape.

## Comprehensive evaluation

* Role Clarity - Ensuring each team member has well-defined roles and responsibilities, minimizing overlap and confusion.
* Skill Assessment - Conducting individual skill assessments and map them against current and future project requirements.
* Communication and Collaboration - Observing team dynamics to identify strengths and weaknesses in communication and cross-functional collaboration.
* Process Efficiency - Evaluating current development processes, including Agile practices, CI/CD pipelines, and deployment workflows, to identify bottlenecks and inefficiencies.
* Team Morale - Gathering feedback from team members on their current roles, work satisfaction, and challenges.

## Proposed changes

* Cross-Functional Teams – Smaller self-sufficient teams for better collaboration, faster feedback loops, and a sense of shared responsibility.
* Skill Development and Upskilling - Regular training programs and encourage self-learning to keep the team's skills current and relevant.
* Empowerment and Ownership – Promoting Ownership, passion and care within their area of expertise, fostering a culture of trust and responsibility.
* Agile and DevOps Practices – Enhancing agile, kanban, DevOps principles streamlining and automating development, improving feedback loops.
* Open Communication – Establishing clear communication channels and open dialogue, collaboration tools, regular team meetings for information sharing.

## Restructuring

* Cross-functional team – Instead of separate development teams having a feature team
* Self-sufficient feature team – having mixed skills eliminating dependency and time delays
* Dedicated Innovation, Research and Development team - To research new technologies, prototypes, and explore emerging trends.
* Tech lead - to guide overall technical direction and ensure consistency across teams. When the development teams scale up then we need an architect, managing multiple Tech leads.

## Timeline and Implementation

* Communication **-** Communicating the proposed changes clearly and transparently to the team, addressing any concerns and seek their feedback.
* Gradual Transition - Implementing changes in a phased manner, starting with pilot projects or specific teams.
* Skill Development - Initiating training programs and create opportunities for upskilling.
* Monitoring, adjustment and realigning - Continuously monitoring progress, gathering feedback, and making adjustments as needed.

## Risks and Mitigation

* Resistance to Change:

Mitigation - Clear communication, address concerns, involve team members in the decision-making process.

* TemporaryDisruption:

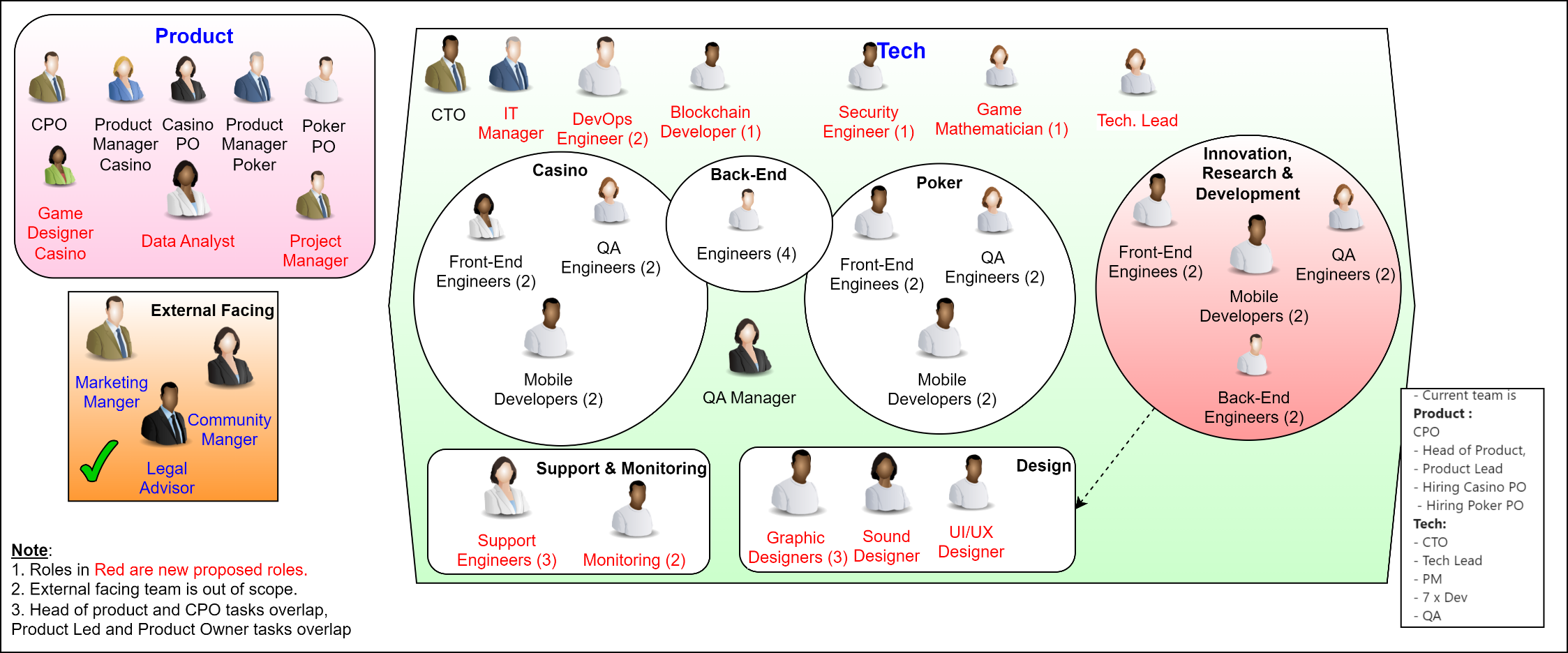
Mitigation **-** Phased implementation, clear communication, and support from management.

* Skill Gaps:

Mitigation **-** Training programs, mentorship, and external hiring if necessary.

## Potential challenges

* Siloed Teams: Development, QA, and Operations teams might be working in isolation, leading to delays and miscommunication.
* Skill Gaps: Rapid technological advancements and evolving project requirements may have created skill gaps within the team.
* Bottlenecks: Certain roles or individuals might be overloaded, leading to bottlenecks in the development process.
* Lack of Ownership: Team members might not feel a strong sense of ownership and accountability for the products they work on.
* Communication Challenges: Poor communication channels and lack of transparency could lead to inefficiencies and misunderstandings.



## Figure 1 Overview of the proposed structure

Below are the roles that are needed to make the game development team cross-functional, robust, efficient and optimised.

## 2.1 Security Engineer (1) *– Shared resource*

***Scope***: Focus on securing both products, especially the decentralized crypto wallet. They should conduct regular security audits, vulnerability assessments, and implement security best practices.

***Focus***: Data protection, encryption, secure transactions, and regulatory compliance.

## 2.2 Game Mathematician (1) *– Shared resource*

***Scope:***  The heart of any casino game lies in its math engine. This engine determines how bets translate into outcomes. Game mathematicians create formulas for features like bonus rounds, free spins, and jackpots. They analyse probabilities, variance, and expected values to craft engaging gameplay.

***Focus***: Mathematics Model, Game Balance, Player Experience, and Game flow.

## 2.3 Front-End Engineers (2/3)

***Scope***: Maintain and enhance the existing codebase for the poker & casino products (Android, Windows, Mac) and the crypto wallet (Android, iOS). They should be proficient in the relevant programming languages and frameworks used in these applications.

***Focus***: Bug fixing, new feature development, and performance optimization.

## 2.4 Back-End Engineers (2/3)

***Scope***: Back-end developers collaborate closely with front-end teams to integrate game logic seamlessly into the user interface. This ensures a cohesive player experience.

***Focus***: Bug fixing, new feature development, Server-Side logic, database management, Game logic implementation, Scalability, Security and Fairness, and performance optimization.

## 2.5 Blockchain expert developer (1) *– Shared resource*

**Scope:** Maintain and enhance the existing codebase for the decentralised crypto wallet, integrating new crypto currencies native to Android & iPhone,

Supporting the Full**-**stack and mobile developers.

***Focus***: crypto wallet, crypto currency Integrations, security, Bug fixing, new feature development, and performance optimization.

## 2.6 Mobile Developers (2)

***Scope***: Specialize in Android and iOS development for the crypto wallet and poker/casino apps. They should focus on platform-specific features, UI/UX improvements, and OS updates.

***Focus***: Ensure smooth user experience across all devices and manage app store submissions.

## 2.7 DevOps Engineer (2) *– Shared resource*

***Scope:*** Oversee the deployment pipelines, cloud infrastructure, and monitoring systems. They should ensure high availability, automated testing, and continues integration/continuous deployment (CI/CD).

***Focus:*** Infrastructure scalability, disaster recovery, Business continuity, and performance monitoring, high SLAs

## 2.8 QA Engineer (2)

With test automation skills

***Scope:*** Create and execute test plans for both the crypto wallet and poker/casino products. They should work closely with developers to identify and fix bugs before release.

***Focus****:* Manual and automated testing, ensuring product quality, robust performance, Load & performance scalability, stability, volume & configuration testing, multi-platform compliance.

## 2.9 Product Manager (1)

They ensure projects are completed on time, within budget, and meet quality standards.

***Scope***: Bridge the gap between technical and non-technical teams, manage product roadmaps, and gather user and product feedback. They should prioritize features and improvements based on business goals.

***Focus***: Strategic planning, user experience, and stakeholder communication.

## 2.10 Data Analyst (1)

***Scope***: Analyse user behaviour, application performance, and financial metrics. They should provide insights to guide product decisions and marketing strategies.

***Focus***: Data-driven decision-making, reporting, and A/B testing.

## 2.11 Game Designer (1)

***Scope***: Game designers are the creative masterminds behind the conceptualization and realization of a game. They blend technical skills, artistic vision, and narrative creativity to shape the game’s experience.

## 2.12 IT Manager (1)

***Scope*:** The IT Manager would oversee the entire technical infrastructure, technical strategy development, ensuring both the poker/casino products and the decentralized crypto wallet operate smoothly across all platforms (Android, iOS, Windows, Mac).

***Focus*:** Ensure uptime, manage cloud services, oversee the deployment pipelines, and maintain hardware and software resources, manage day to day activities of the technical team, coordinating tasks, main point of contact between technical teams and other departments, aligning IT initiatives and business objectives.

## 2.13 Support, Maintenance and Monitoring (3/2) *– Shared resource*

**Scope**: Customer Support, Monitoring and Maintenance.

***Focus***: Provides assistance to players, addressing issues and inquiries. Ensures the game servers and infrastructure are running smoothly and efficiently.

## 2.14 Design Team – Shared resource

**UI/UX Designers**: Design intuitive and engaging user interfaces, focusing on user experience.

**Graphic Designers**: Create visual assets, including game graphics, animations, and promotional materials.

**Sound Designers**: Develop sound effects and music to enhance the gaming experience.

## 2.15 Research and Development Team *– Shared resource*

**Scope**: Research emerging technologies, security, tackling threats, develop proof of concepts for both current and future product lines, maintain and enhance the game development framework, and take ownership of the system architecture. This includes conducting code and design reviews for framework changes, approving modifications, and introducing new, robust technologies, processes, and methodologies to improve game development efficiency. Additionally, work on integrating and exploring future technologies.

***Focus***: Research and development, Game development and architecture ownership, enabling the development to work with the robust and latest technologies, enhancing the efficiency of the game development teams.

## 2.16 Tech Lead *– Shared resource*

**Scope**: A Tech Lead in a software development team plays a crucial role in guiding the technical direction of projects and ensuring the team delivers high-quality software by guiding the overall technical direction and ensure consistency across teams. Tech Lead acts as both a technical expert and a team leader, bridging the gap between the technical and business aspects of software development.

***Focus***: Technical leadership, Mentorship, Code Review, Collaboration, Problem-solving, Architecture and Design, Innovation, Quality assurance, R&D.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

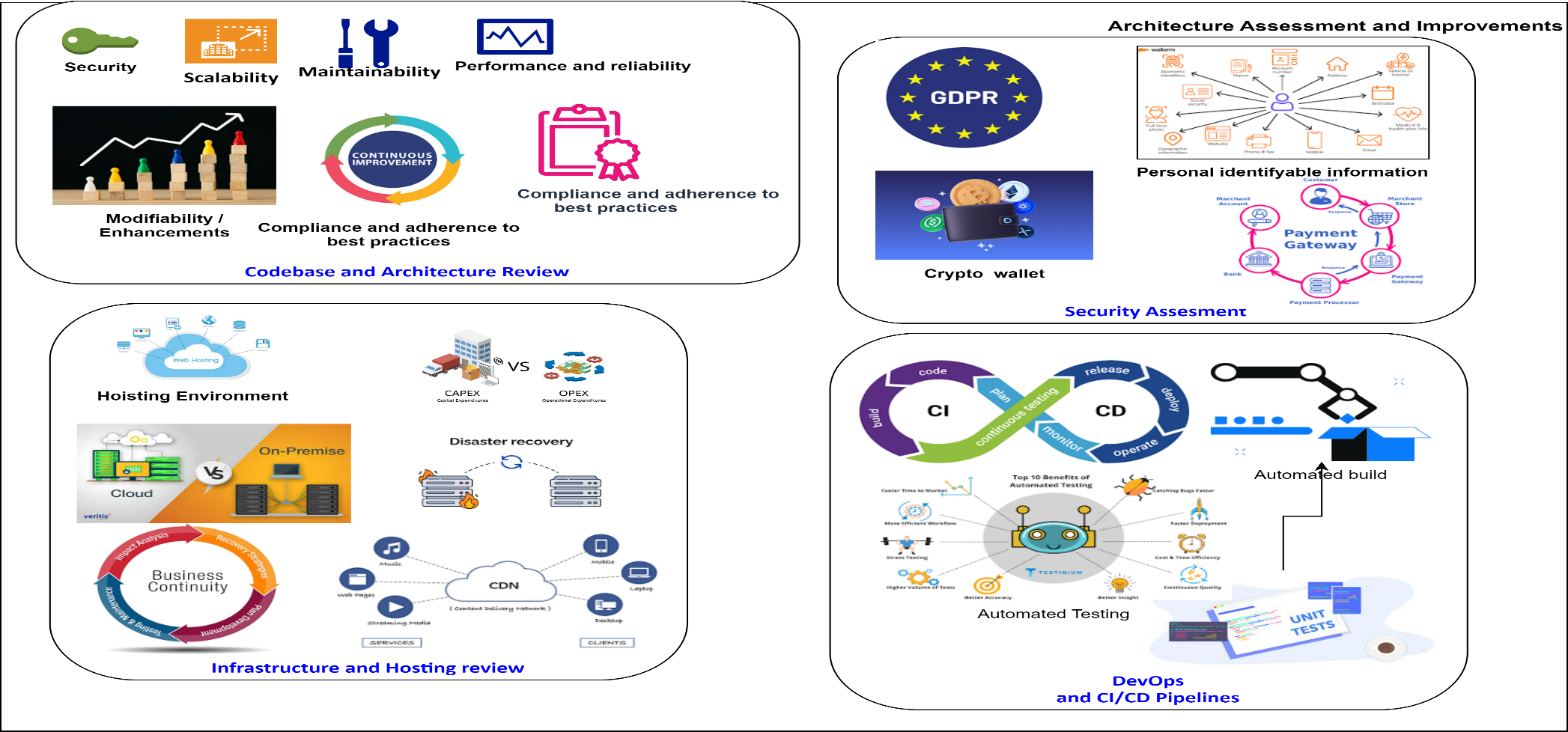
# Architecture Assessment and Improvements

## 3.1 Codebase and Architecture Review:

***Objective***: Assess the current state of the codebase for both products, focusing on:

* Security
* Scalability
* Maintainability
* Performance and reliability
* Compliance and adherence to best practices
* Portability
* Continuous improvement and enhancement
* Modifiability

***Recommendations***: If the code is outdated or overly complex, suggest refactoring, modularization, or even re-platforming. For example, transitioning a monolithic architecture to microservices if scalability is a concern. Implement ***DORA metrics*** for increasing the codebase quality and engineering practices.



## Figure 2 Architecture Assessments and Improvements

## 3.2 Security Assessment:

***Objective***: Conduct a thorough security review emphasising on security audit, penetration testing, PII (Personal identifiable information) and GDPR (General data protection regulation), particularly for the crypto wallet, which handles sensitive transactions and data.

***Recommendations***: Implement end-to-end encryption, multi-factor authentication, and regular security audits. For the poker & casino product, ensure secure payment gateways and anti-fraud measures are in place.

## 3.3 Infrastructure and Hosting Review:

***Objective***: Evaluate the current hosting environment, including cloud services, databases, and server configurations. Pay for Use (OPEX vs CAPEX), If the systems are not in the cloud, then make strategic plan to move it the cloud as soon as possible.

***Recommendations***: If the current infrastructure is not scalable, consider moving to a cloud provider like AWS or Azure with auto-scaling capabilities, supporting disaster recovery, business continuity. Also, introduce load balancers, caching mechanisms, and CDNs to improve performance.

## 3.4 DevOps and CI/CD Pipelines:

***Objective***: Review existing DevOps practices and deployment pipelines to cater for the current business needs.

***Recommendations***: If CI/CD pipelines are not in place, implement them to automate testing and deployments, reducing the risk of human error and increasing deployment frequency. This will require implementation of TDD, unit testing in development if these are not in place to support CI/CD.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Tech Stack Review and Modernization

The current tech stack of C++, Qt, and QML is still popular among game dev community. C++ is a foundational language in game development, especially for performance-critical components. Qt is a strong option for UI development and cross-platform applications. QML's declarative syntax and integration with JavaScript make it an excellent choice for building dynamic and visually appealing user interfaces.

The company has recently made significant investments in acquiring the Casino and Poker platforms and games. The business strategy for short to mid-term is focused on achieving a strong return on investment (ROI) by addressing the challenges associated with the current tech stack, which includes C++, Qt, and QML.

## 4.1 Stack review

#### C++:

* **Performance -** C++ remains the king of performance, crucial for handling complex game logic, real-time simulations, and smooth graphics rendering in demanding casino and poker games.
* **Control and Flexibility** - Its low-level control and extensive libraries give developers the power to optimize every aspect of the game for performance and efficiency.
* **Legacy Code and Expertise** - Many existing game engines and casino platforms are built on C++, making it a practical choice for ongoing development and maintenance.

#### Qt:

* **Cross-Platform Compatibility** - Qt's strength in cross-platform development enables reaching a wide audience across different operating systems and devices, from desktop to mobile, crucial for the online casino and poker industry.
* **UI Development (Widgets and QML)** - Qt provides a mature and flexible UI toolkit for creating visually appealing and responsive interfaces for casino and poker games, using both traditional widgets and QML for declarative UIs.
* **Networking and Backend Integration** - Qt's networking capabilities and seamless integration with C++ allow for efficient communication with backend systems and game servers, essential for handling player data, transactions, and real-time interactions.

#### *QML*:

* **Dynamic and Engaging UIs** - QML excels at creating visually rich and interactive interfaces with animations and transitions, enhancing the player experience in casino and poker games.
* **Rapid UI Development** - QML's declarative syntax allows for faster UI development and prototyping, allowing teams to iterate quickly on new game designs and features.
* **Responsive Design** - QML adapts well to different screen sizes and resolutions, crucial for delivering a consistent experience across various devices.

## 4.2 Challenges

1. C++ is a powerful but complex language, requiring a significant investment in learning and mastering. This naturally reduces the pool of readily available candidates in the location where the development teams are in.
2. While Qt has a strong following, QML expertise adds another layer of specialization, further narrowing the pool of suitable candidates.
3. Depending on your location, there might be a limited local talent pool, necessitating remote work or relocation strategies.
4. Retaining the talent and staying competitive with salary and benefits is crucial to prevent losing talent to other companies.

## 4.3 Industry position:

1. C++ and Qt enjoy significant popularity in the software development world and has a bright future ahead.
2. As of June 2024, C++ has even reached the #2 spot on the TIOBE Index, surpassing Java, indicating a growing trend in its popularity.
3. C++'s versatility and performance make it a preferred choice for diverse fields like game development, embedded systems, high-performance computing, and finance.
4. The vast C++ community ensures ample support, libraries, and learning resources for developers.
5. C++ and Qt are versatile technologies and are used in a wide range of industries beyond gaming like financials, automotive, healthcare, industrial automation, telecommunications, trading platforms, hedge funds, due to their performance, cross-platform support, and UI development capabilities.
6. BMW, Tesla, Audi, Adobe systems, Autodesk, Google (Chrome, backend), Microsoft (windows, office, visual studio), Mozilla, Blizzard entertainment, Ubisoft, Electric arts, Wargaming, Kunos Simulazioni. Microgaming studios, Party Poker.
7. Diverse applications from creative software, gaming to automotive systems and financial tools, C++ and Qt find applications across various domains.
8. Maturity and Stability of C++ and Qt have a long history of development and refinement, offering mature and stable platforms for building reliable software.
9. Rich Ecosystem due to the availability of numerous libraries, frameworks, and tools for both C++ and Qt further enhances their attractiveness for developers.
10. Open-source impact -Many popular open-source projects rely on C++ and Qt, showcasing their community support and impact on the software landscape. Examples are KDE plasma, LMMS, VirtualBox.

## 4.4 Modernization

#### Potential issues:

1. **Lack of Modern C++ Features** - C++ has evolved significantly in recent years. If the team isn't using C++11 or later features (e.g., smart pointers, lambda functions, range-based for loops), they might be missing out on productivity and safety benefits.
2. **Manual Memory Management** - If the codebase relies heavily on manual memory management (e.g., new and delete), there's a higher risk of memory-related bugs.
3. **Outdated Libraries or Tools** - If any libraries or tools used haven't been updated in a while, they could be less efficient, less secure, or lack support for newer technologies or platforms.

*Proposed Changes:*

1. **Embrace Modern C++** - Encourage the team to adopt modern C++ features to enhance code readability, safety, and efficiency.
2. **Considering Smart Pointers** - Transition to smart pointers (e.g., std::unique\_ptr, std::shared\_ptr) to help avoid memory leaks and improve code clarity.
3. **Explore Alternative UI Technologies** - If UI performance becomes a significant bottleneck, explore options like using Qt Quick with a C++ backend for more fine-grained control or considering a game engine for specific needs.
4. **Evaluate Tooling** - Regularly assess development tools and workflows to see if there are more modern alternatives that could improve productivity.
5. **Continuous Learning** - Encourage the team to stay up-to-date on new language features, libraries, and best practices through training and self-learning.

## 4.5 C++, Qt comparison with other popular languages

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **C++, Qt** | **Unity** | **Node.js, React** |
| **Performance** | High (Native code, fine-grained control) | Good (Optimized engine, C# scripting) | Moderate (JavaScript, single-threaded) |
| **Cross-Platform** | Excellent (Qt's strength) | Very Good (Wide platform support) | Good (with frameworks like Electron) |
| UI Development | Flexible (Widgets, QML) | Excellent (Visual editor, UI components) | Very Good (Component-based, React's power) |
| **Real-time** | Excellent (Native control) | Good (Networking features) | Moderate (WebSockets, potential latency) |
| Community | Large C++/Qt community | Massive Unity community | Huge Node.js/React communities |
| Learning Curve | Steep (C++, Qt/QML) | Moderate (C# scripting, visual tools) | Moderate (JavaScript, React concepts) |
| 3D Graphics | Possible (Qt 3D, external libraries) | Excellent (Core strength) | Challenging (WebGL limitations) |
| **Security** | High (Control over implementation) | Moderate (Relies on engine security) | Moderate (Web vulnerabilities) |
| **Scalability** | High (Can scale efficiently) | Good (Cloud solutions, multiplayer support) | Moderate (Horizontal scaling with Node.js) |
| User Base | Smaller, specialized | Very Large | Massive |
| **Licensing** | C++: Open Source, Qt: Dual (Open Source/Commercial) | Proprietary, with free tier and revenue share | Open Source |
| **Costing** | C++: Free, Qt: Open-Source or Commercial fees | Free tier with limitations, revenue-based fees | Free (Open-Source libraries) |
| **WebGL Support** | Possible (Qt WebGL plugin, limitations apply) | Good (Built-in support) | Excellent (Native browser support) |
| **Mobile Development** | Good (Qt, Native Performance) | Excellent (Mature mobile toolchain) | Possible (React Native, performance considerations) |

Ultimately, the best choice depends on your project's specific needs, team expertise, and priorities. For most casino and poker games, Unity is a compelling option due to its ease of use, rich features, and large community. However, C++, Qt remains a powerful choice for performance-critical or highly customized projects where full control is desired. Node.js with React can be suitable for web-based implementations or simpler games.

## 4.6 Talent

The above comparison data highlights the features that C++, Qt are strong at. C++ and Qt, while powerful and widely used, have a reputation for having a steeper learning curve compared to some other technologies. This can make finding and retaining skilled developers a challenge. companies can take proactive steps to attract, retain, and grow talent in these areas.

### Attracting:

* Competitive Compensation and Benefits
* Highlight Interesting Projects
* Strong Company Culture
* Flexible Work Arrangements
* Targeted Recruitment – Companies like Toptal have good resources who can he hired within 48 hours.

### Retaining:

* Challenging and Meaningful Work
* Career Development
* Mentorship and Knowledge Sharin
* Recognition and Rewards
* Competitive Compensation
* Staying connected to the industry

### Growing:

* Training and development
* Internal Hackathons and R&D projects
* Contribution to open-source
* Knowledge sharing platforms
* Fostering a supportive community

By implementing these strategies and fostering a positive and supportive environment, companies can attract, retain, and grow a talented pool of C++ and Qt developers, ensuring long-term success in their projects and endeavours.

# Roadmap and implementation plan