Technical leadership Candidate Task

Strategic Planning and Team, Architecture, and Tech Stack Evaluation

By

Yoganand Aiyadurai

Version 1.0

Dated: 16th August 2024

Contents

1. Introduction 7
2. Team Evaluation and Restructuring 7
   2.1 Comprehensive evaluation 7
   2.2 Proposed changes 7
   2.3 Restructuring 8
   2.4 Timeline and Implementation 8
   2.5 Risks and Mitigation 8
   2.6 Potential challenges 8
   2.7 Roles 9
      2.7.1 Security Engineer (1) – *Shared resource* 10
      2.7.2 Game Mathematician (1) – *Shared resource* 10
      2.7.3 Front-End Engineers (2/3) 10
      2.7.4 Back-End Engineers (2/3) 10
      2.7.5 Blockchain expert developer (1) – *Shared resource* 10
      2.7.6 Mobile Developers (2) 10
      2.7.7 DevOps Engineer (2) – *Shared resource* 11
      2.7.8 QA Engineer (2) 11
      2.7.9 Product Manager (1) 11
      2.7.10 Data Analyst (1) 11
      2.7.11 Game Designer / Producer (1) 11
      2.7.12 IT Manager (1) 11
      2.7.13 Support, Maintenance and Monitoring (3/2) – *Shared resource* 12
      2.7.14 Design Team – Shared resource 12

2.7.15 Research and Development Team – *Shared resource* 12
2.7.16 Tech Lead – Shared resource 13
3. Architecture Assessment and Improvements 13
3.1 Codebase and Architecture Review: 13
3.2 Security Assessment: 14
3.3 Infrastructure and Hosting Review: 15
3.4 DevOps and CI/CD Pipelines: 16
3.5 Agile practices, Development methodologies, Code quality metrics, Engineering practices metrics 17
3.5.1 Objective 17
3.5.2 Recommendations 17
3.6 Systems, Architecture, Processes, Workflow documentation 19
3.6.1 Objective 19
4. Tech Stack Review and Modernization 20
4.1 Stack review 20
4.1.1 C++ 20
4.1.2 Qt 21
4.1.3 *QML* 21
4.2 Challenges 21
4.3 Industry adoption and position: 21
4.4 Modernization 22
4.4.1 Potential issues 22
4.4.2 Proposed Changes 22
4.5 C++, Qt comparison with other popular languages 22
4.6 React and Node.JS tech stack - phased approach for rewriting the systems 23

4.6.1 Considerations 23
4.6.2 Recommendation 24
4.6.3 React Frontend framework 24
4.6.4 Node.JS Back-end framework 24
4.6.5 Rewrite Phases 24
4.7 Talent 26
4.7.1 Attracting 26
4.7.2 Retaining 26
4.7.3 Growing 27
4.8 Rust tech stack option compared to C++ 27
4.8.1 Aspects of game development referencing Rust 27
4.8.2 Factors to consider when using Rust for casino game development 27
5. Roadmap and implementation plan 28
5.1 **0 to 1 month** (4 weeks) – Adapting, learning and understanding phase 28
5.1.1 Immersion and Learning 28
5.1.2 Building relationship 28
5.2 **2 to 3 months** – Strategic planning and aligning phase 28
5.2.1 Technology vision and strategy 28
5.2.2 Team and process evaluation 29
5.2.3 Low hanging fruits and immediate impact 29
5.3 **3 to 6 months** – Execution, evaluation and improving phase 29
5.3.1 Roadmap implementation 29
5.3.2 Empowering and developing the team 29
5.4 **6 months and greater** – the ongoing process 32

5.4.1 Innovation and exploration 32
5.4.2 Focus areas 32
5.5 Risk and Mitigation Strategy 32
5.6 Final words 33
6. Out of assignment 33
6.1 Emerging technologies under CTOs radar 33
6.1.1 Artificial Intelligence and Machine Learning 33
6.1.2 Blockchain and Cryptocurrency 34
6.1.3 Virtual Reality (VR) and Augmented Reality (AR) 34
6.1.4 Cloud Gaming 34
6.1.5 5G and Edge Computing 34
6.1.6 Metaverse and Virtual Worlds 34
6.1.7 Beyond Technology 34
6.2 Reporting structure for CTO to stakeholders 35
6.2.1 Weekly Status Reports 35
6.2.2 Monthly Performance Reports 35
6.2.3 Quarterly Strategic Reports 35
6.2.4 Security and Compliance Reports 35
6.3 Staying on top of technological advances 35
6.3.1 Active Learning & Exploration 35
6.3.2 Leveraging Networks & Resources 35
6.3.3 Internal Practices 36
6.3.4 Focused Areas 36
6.4 Questions to be answered before choosing a new technology to solve the business problem 36

6.4.1 Strategic Alignment 36
6.4.2 Technical Feasibility and Impact 37
6.4.3 Financial and Resource Considerations 37
6.4.4 Risk Management and Mitigation 37
6.4.5 Long-Term Vision 38

1. Introduction

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

2. 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.

2.1 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.

2.2 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.

2.3 Restructuring

- **Cross-functional team** – Instead of separate development teams having a **feature team**. A feature team is a cross-functional, self-organizing team responsible for delivering complete, end-to-end customer-centric features
- **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.

2.4 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.
- **Transiting to good agile** – Evaluating the current agile, kanban and scrum practices, refining with a systematic approach to good agile.

2.5 Risks and Mitigation

- **Resistance to Change**:
  Mitigation - Clear communication, address concerns, involve team members in the decision-making process.
- **Temporary Disruption**:
  Mitigation **-** Phased implementation, clear communication, and support from management.
- **Skill Gaps**:
  Mitigation **-** Training programs, mentorship, and external hiring if necessary.

2.6 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.

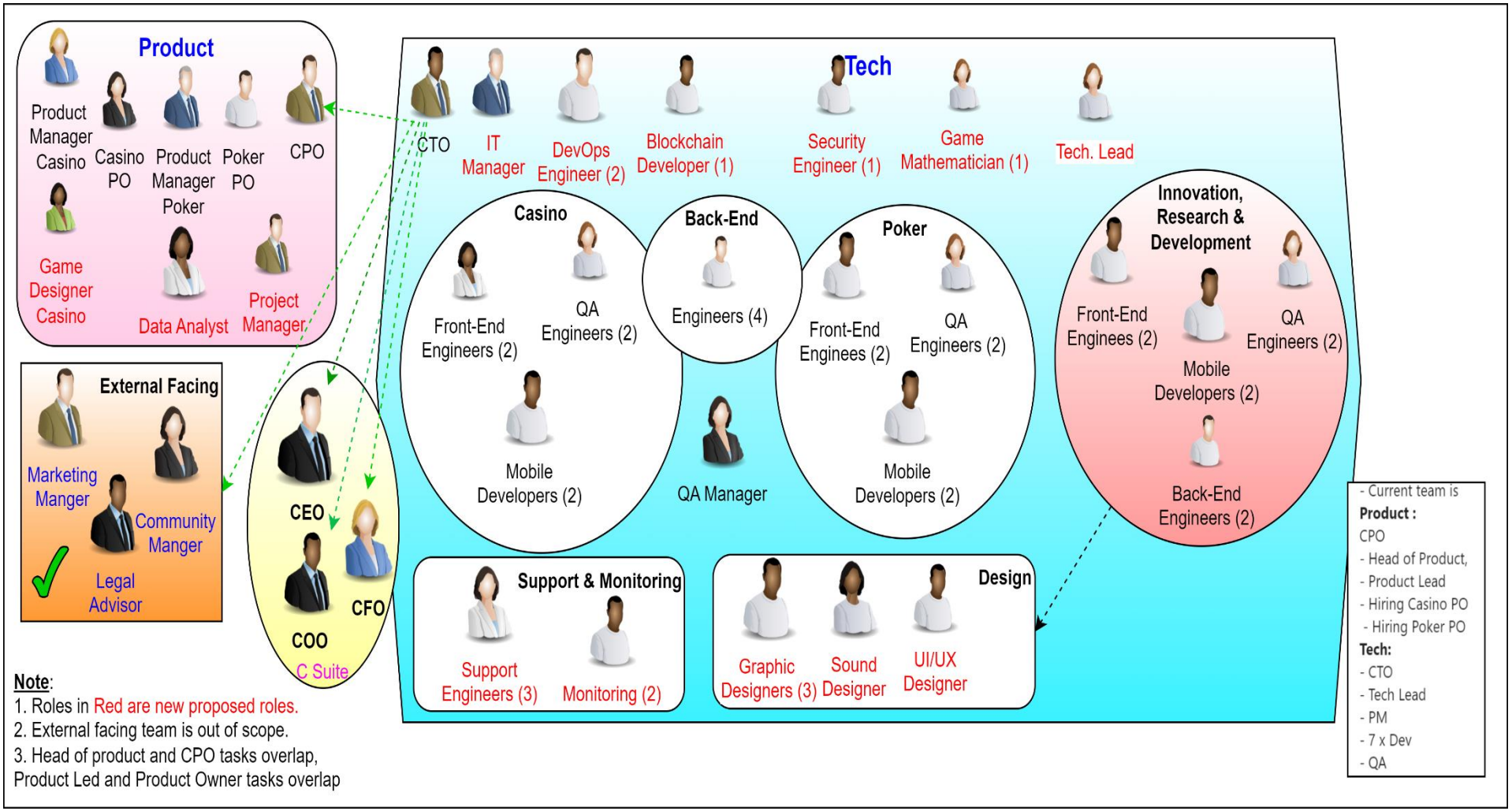


Figure 1 Overview of the proposed structure

2.7 Roles

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

2.7.1 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, ensuring that the systems adhere to the rules, regulations and compliance, the applications are operating in.
Focus: Data protection, encryption, secure transactions, and regulatory compliance.

2.7.2 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.

2.7.3 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.

2.7.4 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. Create clear, responsive and robust APIS to be used by client applications.
Focus: Bug fixing, new feature development, Server-Side logic, database management, Game logic implementation, Scalability, Security and Fairness, and performance optimization.

2.7.5 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.

2.7.6 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.

2.7.7 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

2.7.8 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.

2.7.9 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.

2.7.10 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.

2.7.11 Game Designer / Producer (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.
Focus: Game mechanics, Game Economy, Player experience, Game progression, Theming and Atmosphere, Feature design.

2.7.12 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.

2.7.13 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.

2.7.14 Design Team – Shared resource

UI/UX Designers

Scope: User Research and Analysis, Information Architecture and Interaction Design, Visual Design and User Interface (UI) Design, Usability Testing and Evaluation

Focus: Design intuitive and engaging user interfaces, focusing on user experience. Shaping the user experience and ensuring that products are not only functional but also enjoyable and engaging to use.

Graphic Designers

Scope: Concept Art & Visual Development, User Interface (UI) Design, Marketing & Promotional Materials, Motion Graphics & Animation, 3D Modelling & Sculpting, VR and AR.

Focus: Play a critical role in shaping the visual identity and aesthetic appeal of games. Their work contributes significantly to player engagement, immersion, and overall enjoyment. Incorporating new technologies and design approaches to create even more captivating and unforgettable gaming experiences.

Sound Designers

Scope: Sound Design and Creation, Implementation & Integration, Adaptive & Interactive Audio, Procedural Audio Generation.

Focus*:* Develop sound effects and music to enhance the gaming experience. Shaping the audio landscape and emotional impact of games. Contribute significantly to player immersion, feedback, and overall engagement. As game development continues to advance, incorporating new technologies and design approaches to create even richer and more captivating auditory experiences.

2.7.15 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.

2.7.16 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.

---

3. 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 ***Cycle time metrics*** engineering practices. The key metrics to watch for are:

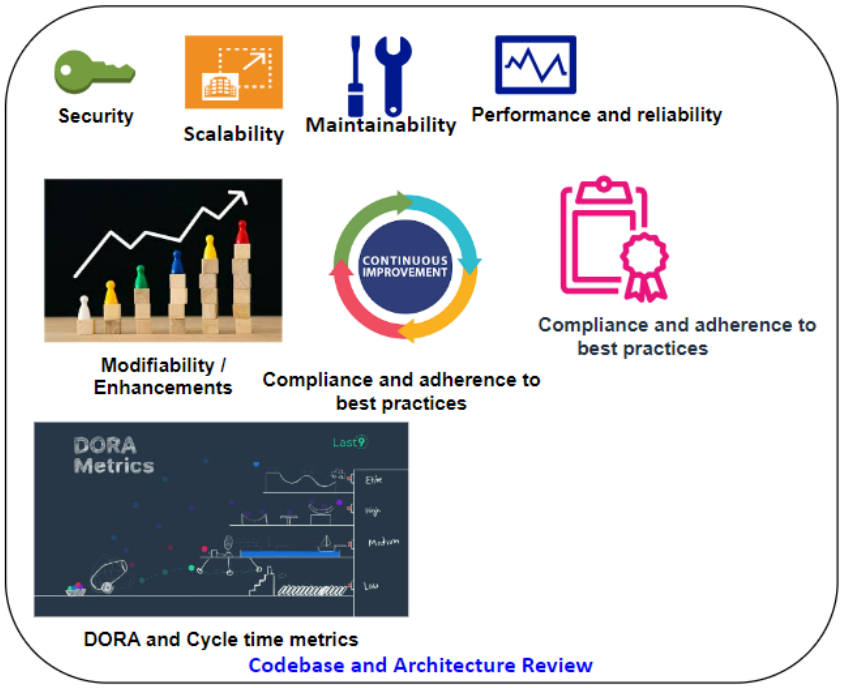


Figure 2 Architecture Assessments and Improvements

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. Security assessment of data must be divided into CIA (Confidentiality, Integrity and Availability)

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.

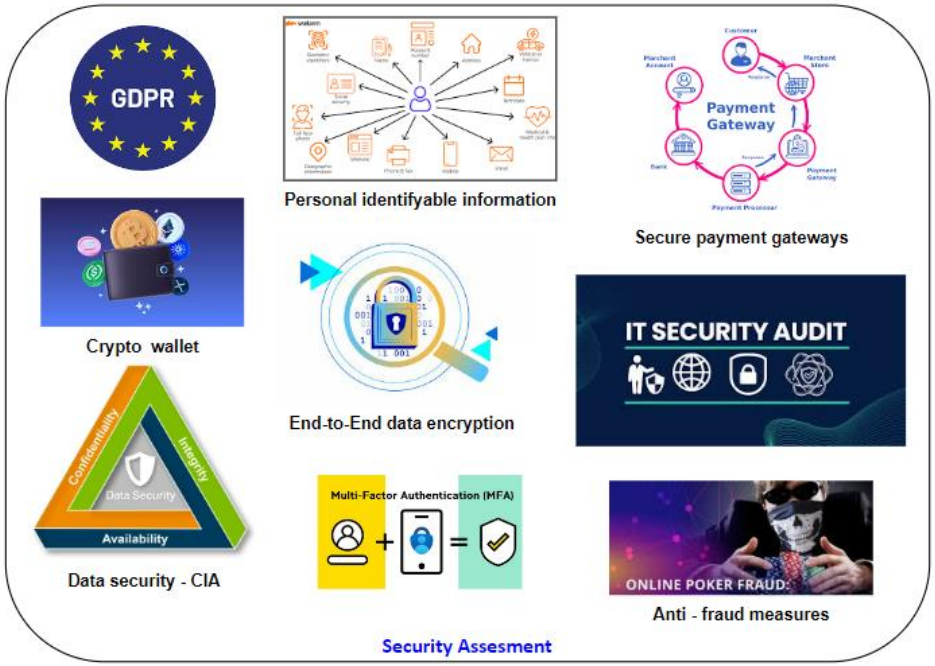


Figure 3 Security Assessment review

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 (Vertically or Horizontally depending on the needs) , consider moving to a cloud provider like AWS or Azure with auto-scaling capabilities, supporting disaster recovery, and business continuity. Also, introduce load balancers, caching mechanisms, and CDNs to improve performance. Aiming for high SLAs as the gaming industry is operational 24/7.

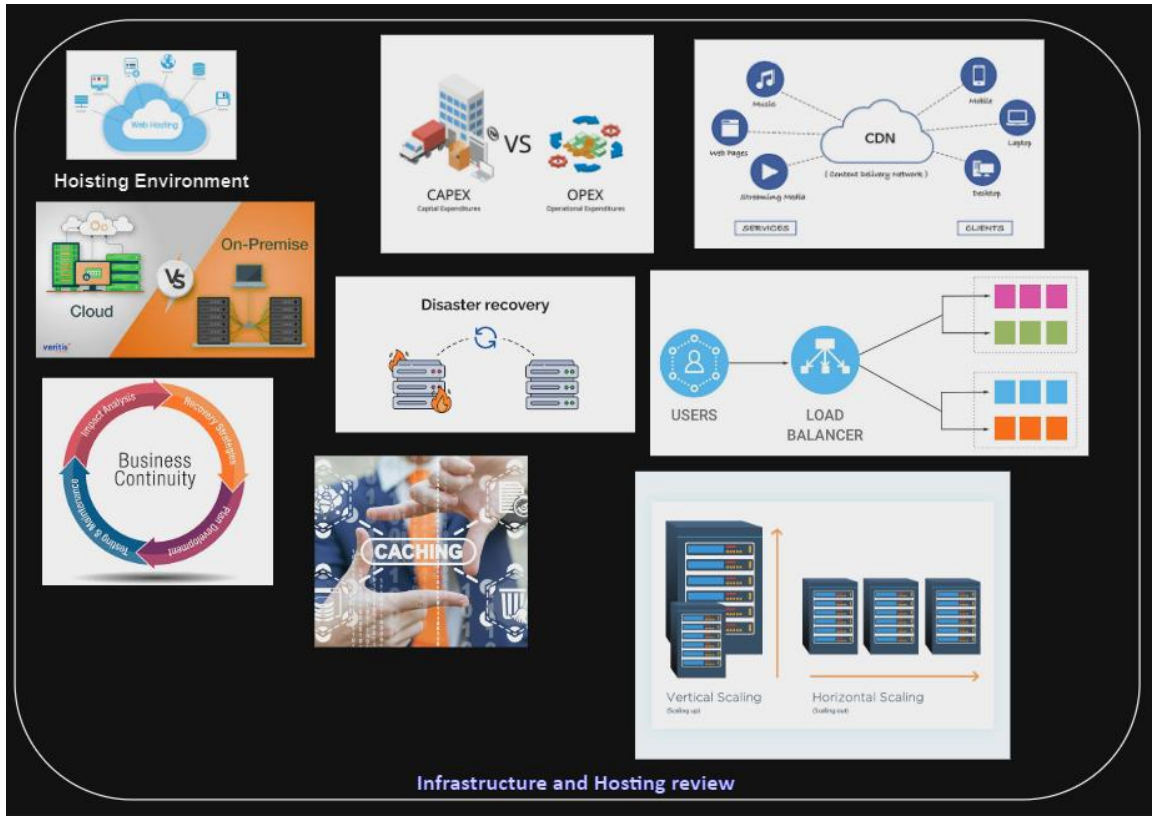


Figure 4 Infrastructure hoisting review

3.4 DevOps and CI/CD Pipelines:

Objective: Review existing DevOps practices, tools 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. Use GitHub, Git, Bitbucket and usage of cloud computing.

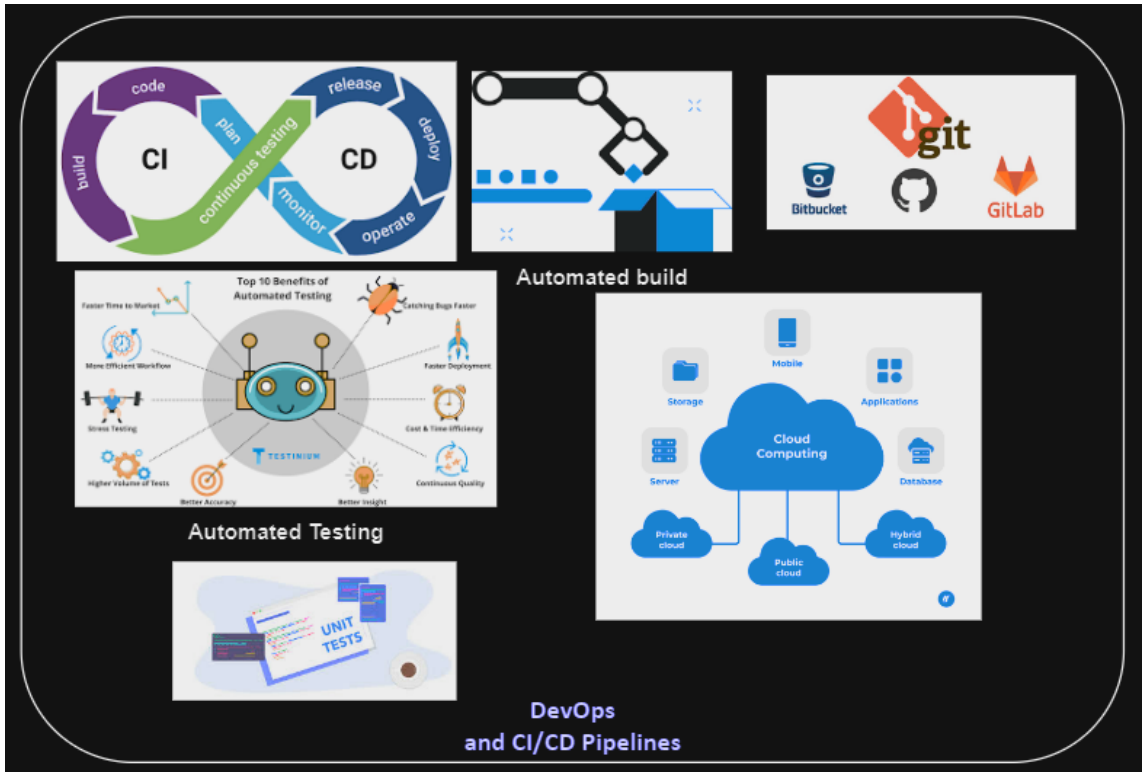


Figure 5 DevOps CI/CD Pipelines

3.5 Agile practices, Development methodologies, Code quality metrics, Engineering practices metrics

3.5.1 Objective

Review, analyse, and enhance Agile, Kanban, and Scrum methodologies to increase efficiency, directly impacting lead time, cost, and software quality, as well as the overall quality of the end product.

3.5.2 Recommendations

LinerB is a highly regarded tool for refining engineering practices and improving code quality within a repository. It offers a suite of ready-made tools and metrics that can be integrated with the code repository to monitor developer activity and elevate code quality. Key code quality metrics such as code complexity, code coverage, code churn, code duplication, and static code analysis, along with engineering practice Cycle metrics like pull request size, Coding time, Pickup time, deploy time, review time, merge conflict rate, commit frequency, and build success rate, are invaluable. Additional metrics like code review depth, test coverage growth, and defect density further support the goal of enhancing code quality and engineering practices, enabling the development of high-quality products more cost-effectively, in less

time. While many companies claim to be Agile, true Agile implementation goes beyond simply adopting a set of practices. It's about nurturing a mindset and culture that prioritizes adaptability, collaboration, and continuous improvement. The specific framework or methodology used is secondary to these core principles. To thrive in today's rapidly changing business environment, critical evaluation and fine tuning the current Agile practices and make necessary adjustments to ensure they fully embody the Agile mindset, contributing to continuous delivery of high-quality product and features. Focus on good KPIs to measure engineering practices to bring in discipline to increase the quality of work, contributing to deliver outstanding software products, controlling the time, cost and quality, moving teams from fair to Elite.

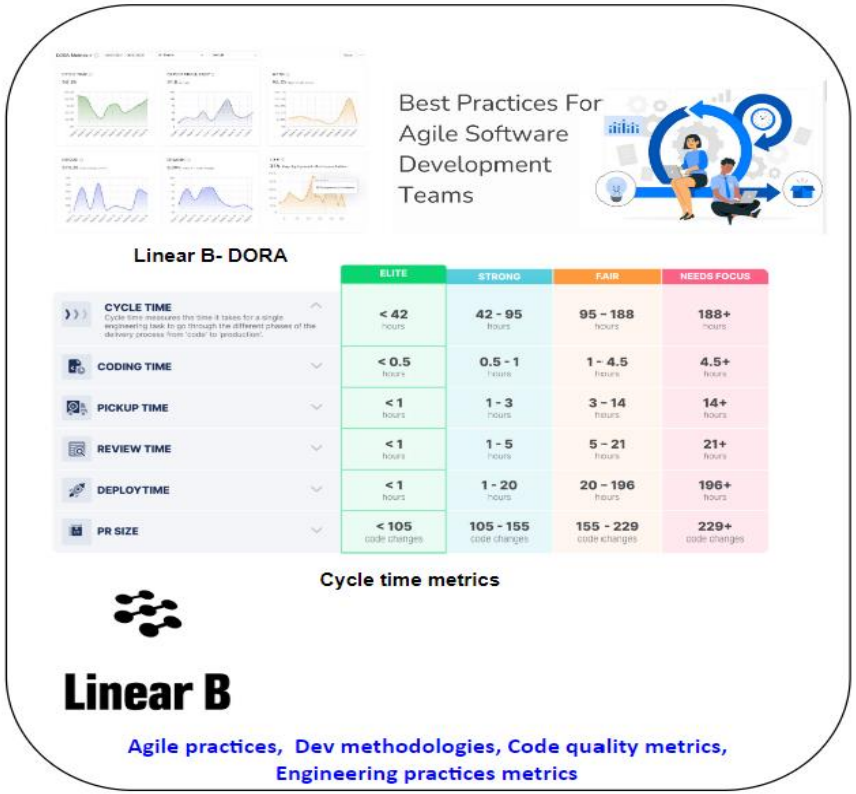


Figure 6 Agile best practices, Dev methodologies, Code Cycle time metrics

3.6 Systems, Architecture, Processes, Workflow documentation

3.6.1 Objective

Knowledge Management and Onboarding, Preservation of Institutional Knowledge, Efficient Onboarding, Reduced Reliance on Individual, minimise miscommunication, increase collaboration, Shared understanding, improved decision making, efficient maintenance and troubleshooting, improved system stability, scalability and growth, Facilitate system expansion, engineering professionalism, commitment to quality, Development team's reputation and trustworthiness, facilitating collaboration with stakeholders and clients, ensuring that the system meets the business and clients requirement needs.

Recommendations: In today's fast-paced and competitive software development landscape, the importance of system documentation and architecture documents cannot be overstated. They are essential tools for knowledge sharing, collaboration, maintenance, scalability, and building trust with clients and stakeholders.

A company that invests in comprehensive and up-to-date documentation lays the foundation for long-term success. It empowers its team to work more efficiently, deliver higher-quality software, and adapt to changing market demands. Tools like confluence, Notion, Git, GitHub, Lucid chart, Draw.io, PlantUML, Miro, Swagger UI, Archi, Jira will help the technology department with the documentation to produce great software.

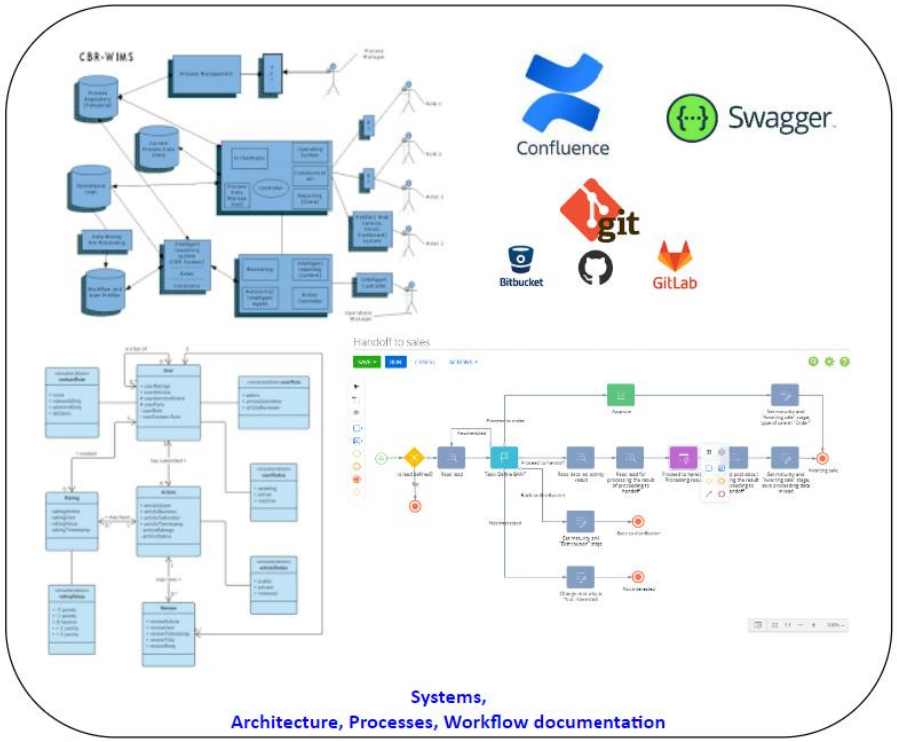


Figure 7 System, workflow, architecture documentation

4. Tech Stack Review and Modernization

The current tech stack C++, Qt, and QML used 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.

Figure 8 Tech stack review and modernization

4.1 Stack review

4.1.1 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.

4.1.2 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.

4.1.3 *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 adoption and 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. Companies using stack C++, Qt, QML are 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

4.4.1 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.

4.4.2 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 React and Node.JS tech stack- phased approach for rewriting the systems

4.6.1 Considerations

- **Performance**: Carefully benchmark and optimize performance throughout the rewrite process, with particular focus on core game logic. In gaming, performance is critical. With thoughtful design, optimization, and scaling, Node.js gaming backends have proven capable of handling thousands, or even tens of thousands, of concurrent users.
- **Team Skills:** Assess the team's proficiency in React and Node.js, along with their willingness to learn. Identify any training needs to ensure the team is well-equipped for the project.
- **Third-Party Integrations**: Consider the impact of the rewrite on existing integrations with payment gateways, analytics tools, and other services.
- **Web3, Blockchain, and Crypto Wallets**: Node.js is a versatile and powerful tool for building Web3 and blockchain applications and services. Its efficiency, familiarity, and extensive ecosystem make it a popular choice among developers in this space. Integrating crypto wallets and new currencies using Node.js is relatively straightforward, due to the wide range of libraries available in the Node.js ecosystem, such as Web3.js, Ethers.js, and Bitcoin-Lib for Bitcoin wallets. These libraries offer well-documented functionalities for wallet interactions.

- **Core game logic and services**: Rewriting the core game logic (e.g., RNG, card shuffling, game rules) in JavaScript (used by React and Node.js) might lead to performance issues compared to the highly optimized C++ code. This must be carefully considered in the redesign and new architecture.

4.6.2 Recommendation

- **Hybrid Approach**: A hybrid approach strikes a balance between rapidly getting to market and gradually modernizing your tech stack.
- **Leveraging Technologies**: This approach allows you to harness the strengths of React and Node.js for the frontend and backend while initially retaining performance-critical components in C++.
- **Game Logic Services**: As you transition game logic services and APIs from C++ to Node.js, careful redesign, optimization, and evaluation for microservice architecture, scalability, responsiveness, and performance strategies are essential.
- **Existing User Base**: With the Casino and Poker games already live and serving 5,000 concurrent users, the strategy should focus on delivering high-quality service and rich game content to these users while simultaneously scaling to support 100,000+ concurrent users.
- **Decommissioning Old Games**: Once the new game development framework with React, Node.js, and a microservice architecture is in place, formulate a strategy to decommission the old C++ games and any hybrid games developed used during the tech stack transition. This will reduce long-term support and maintenance costs.
- **Iteration and feedback**: It's crucial to plan carefully, prioritize user experience, and continuously iterate based on feedback and performance data.

4.6.3 React Frontend framework

- **Visual elements**: React is a strong choice for building interactive user interfaces, making it suitable for the visual elements of your poker and casino games.
- **User interface**: Game lobby, tables, player avatars, betting interfaces, animations, etc.
- **User interactions**: Handling clicks, drags, input forms, etc

4.6.4 Node.JS Back-end framework

Node.js can handle real-time communication and server-side logic, essential for Casino and poker multiplayer games, managing game state.

- **Server-side game logic**: Managing game sessions, player connections, handling bets, distributing cards, enforcing game rules, etc.
- **Real-time communication**: *Web Sockets for instant updates between players and the server.*
- **Database interactions**: *Storing* player data, game history, etc. (using a database like MSSQL, MySQL, PostgreSQL or MongoDB).

4.6.5 Rewrite Phases

4.6.5.1 Phase 1.0: Stabilize existing products and support (Short-term)

- Focus on stabilising the existing C++, Qt poker and casino games / platform as there are active users and have just received market exposure and are generating revenue.
- Optimize performance and fix critical bugs.
- Gather user feedback and identify areas for improvement.

- Continue building the games with C++, Qt until the new framework is ready to increase the user base and revenue.
- Evaluate the teams skills and readiness for React and Node.JS tech stack.
- Formulate a strategy for training, certifications and upskilling, timelines for the completion of training phase.

4.6.5.2 Phase 1.1: Parallel development (Short-term to mid-term) hybrid approach

- Once the teams are upskilled, kick-off the tech stack transition strategy with clear milestones, sharing the plans with the development team.
- Start building the React framework, new components, features and modules in React.
- Gradually replace parts of the frontend UI with React components.
- Keep the core game logic, server-side logic in C++, interacting with the game through the existing services and APIs.
- Build a proof of concept with a small prototype to validate the new tech stack, game idea and assess React responsiveness and performance.
- Polish, optimization, and continuously iterate on gameplay, visuals, responsiveness and performance, leveraging tools and libraries to enhance the game experience.

4.6.5.3 Phase 1.2: Progressive Rewrite (Long-term)

- Continue replacing more UI elements with React.
- Start building backend framework, APIs, modules and components for handling real-time communication and server-side logic and database access using Node.JS.
- and expanding the Node.js backend. Ensure that ***security, stability, scalability, maintainability, continuous improvement and portability*** are always the high priority in design.
- If Node.JS performance-critical game logic is not satisfactory, then evaluate the feasibility of rewriting performance-critical game logic in a language in modern languages like Rust or Web Assembly, which can offer better performance than JavaScript while integrating with the web stack.
- Continuously monitor performance and user experience to ensure a smooth transition.
- Once the new game development framework with React, Node.js, formulate a strategy to decommission the old C++ games and any hybrid games developed used during the tech stack transition.

4.6.5.4 Alternate option - separate team focusing only on rewrite and new tech stack

If there is a luxury of having a separate team focusing only on the rewrite of the new game framework with React and Node.JS then:

- The tech stack transition will speed up.
- One team focusing on the stabilize existing products and support, parallelly increasing the user base.
- One team undergoing the training and upskilling with React and Node.JS.
- The progressive transition of tech stack from C++, Qt to React and Node.JS will start earlier, thereby achieving the business goals and solving the current problems.
- The business will be ready upfront to cater for rolling out innovative and content rich poker and casino games leading to increased user base and attaining high revenue.

- With the support for security, scalability, continuous enhancement new markets and operators can be captured with ease.

This team can then be transformed to the Research and Development team in near future.

4.6.5.5 Other teams support

The increase in user base, and capturing new markets and operators, needs support from:

- Marketing and user acquisition
- Product management
- Customer Support and Community Management
- Sales and monetization
- Legal and compliance
- Finance and accounting

From the technology side the engineering team can guarantee that all the above teams can perform their tasks to fulfil the users and business needs efficiently.

4.7 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.

4.7.1 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.

4.7.2 Retaining

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

4.7.3 Growing

- Training and development
- Internal Hackathons and R&D projects
- Contribution to open-source
- Knowledge sharing platforms
- Fostering a supportive community
- Remote and cost-effective high skills

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.

4.8 Rust tech stack option compared to C++

As C++ has a steep learning curve, the new modern language Rust required the same effort to learn. The new modern Rust language is gaining popularity and have limited people with the skill sets. When compared to C++, Rust is as equal or safer than C++ when it comes to performance, safety, and concurrency making it compelling option for teams looking for modern and reliable technology stack. Rust can be a powerful tool for creating engaging and secure gaming applications and services.

4.8.1 Aspects of game development referencing Rust

- **Game Logic:** Rust's performance allows for efficient handling of complex game logic, calculations, and simulations, crucial for ensuring fair and responsive gameplay.
- **Random Number Generation:** Rust's strong type system and libraries for secure random number generation help ensure the randomness and fairness of casino games.
- **Networking and Communication:** Rust's concurrency model and networking libraries enable efficient handling of real-time communication and multiplayer interactions, vital for online casino games.
- **Security:** Rust's focus on memory safety and its lack of null pointers and data races contribute to more secure code, reducing the risk of vulnerabilities and exploits that could compromise the integrity of a casino game.
- **Backend Services:** Rust's performance and efficiency make it a great choice for building scalable and reliable backend services to handle player accounts, transactions, and other critical data.

While Rust might be less common in game development compared to languages like C++ or C#, its growing ecosystem and benefits make it an increasingly attractive option, especially for projects where performance, safety, and maintainability are priorities.

4.8.2 Factors to consider when using Rust for casino game development

- **Learning Curve:** Rust has a steeper learning curve compared to some other languages, so teams might need to invest in training or onboarding.
- **Libraries and Frameworks:** While Rust's ecosystem is growing, it might still have fewer game-specific libraries and frameworks compared to more established languages.

- **Community and Resources:** The Rust community is active and helpful, but it might be smaller compared to communities for more established game development languages.

Overall, Rust is a capable language for developing casino and poker games, and its advantages in performance, safety, and concurrency make it a compelling option for teams looking for a modern and reliable technology stack. If you're willing to invest in learning Rust and leverage its strengths, it can be a powerful tool for creating engaging and secure casino game experiences.

5. Roadmap and implementation plan

The first steps and roadmap for a technical leader joining a new company can be broken down into several phases, focusing on understanding the current landscape, building relationships, and establishing a strategic direction.

5.1 0 to 1 month (4 weeks) – Adapting, learning and understanding phase

5.1.1 Immersion and Learning

- Meet with key stakeholders (CEO, executives, team leads) to understand the company's vision, goals, and challenges
- Meeting with HR to familiarize with the policies and procedures
- Setting up weekly meeting with the key people. Setup meeting and immediate feedback loop with the direct manager.
- Exploring deeply into deep into existing technical documentation, architecture diagrams, and codebases to gain a comprehensive understanding of the technology landscape. At the same time identify the loopholes and missing information that can affect a new comer, to fix it
- Conduct a technical audit to assess the current state of systems, identify pain points, and uncover potential risks or opportunities
- Familiarizing with the company culture, communication styles, and decision-making processes

5.1.2 Building relationship

- Meeting the teams and key team members to know their skills, challenges and aspiration
- Meeting the heads of other departments (finance, marketing, sales, HR, legal, product management, IT) to build rapport and to understand their pain points and needs from the technology point of view and how it can support their goals.
- Check the communication channels and improve it, if it’s not optimised to foster trust within the technology teams and across the organisation.

5.2 2 to 3 months – Strategic planning and aligning phase

5.2.1 Technology vision and strategy

- Analyse the existing strategy (improvise/Create a new one).
- Define a clear technology vision aligned with the company’s overall business goals.
- Create a strategic roadmap highlighting the key initiatives, priorities and timelines.

- Publish and communicate the technology vision and roadmap effectively to stakeholders, execs and department heads, securing their buy-in and support.

5.2.2 Team and process evaluation

- Assess the current team structure, roles and responsibilities, workload, delivery effectiveness,
- Evaluate existing development processes, tools, workflows, information sharing, sprint backlogs, user stories, prioritization, appreciations, feedbacks, reporting, KPIs, OKRs, development methodologies, training, mentoring and identify opportunities for optimization and modernization.

5.2.3 Low hanging fruits and immediate impact

- Identify areas of quick improvements, fine tuning, optimization, to demonstrate immediate value gain trust and build credibility.
- Check the technical debt, security vulnerabilities, that can cause significant risks to products, reputation loss to the company.

5.3 3 to 6 months – Execution, evaluation and improving phase

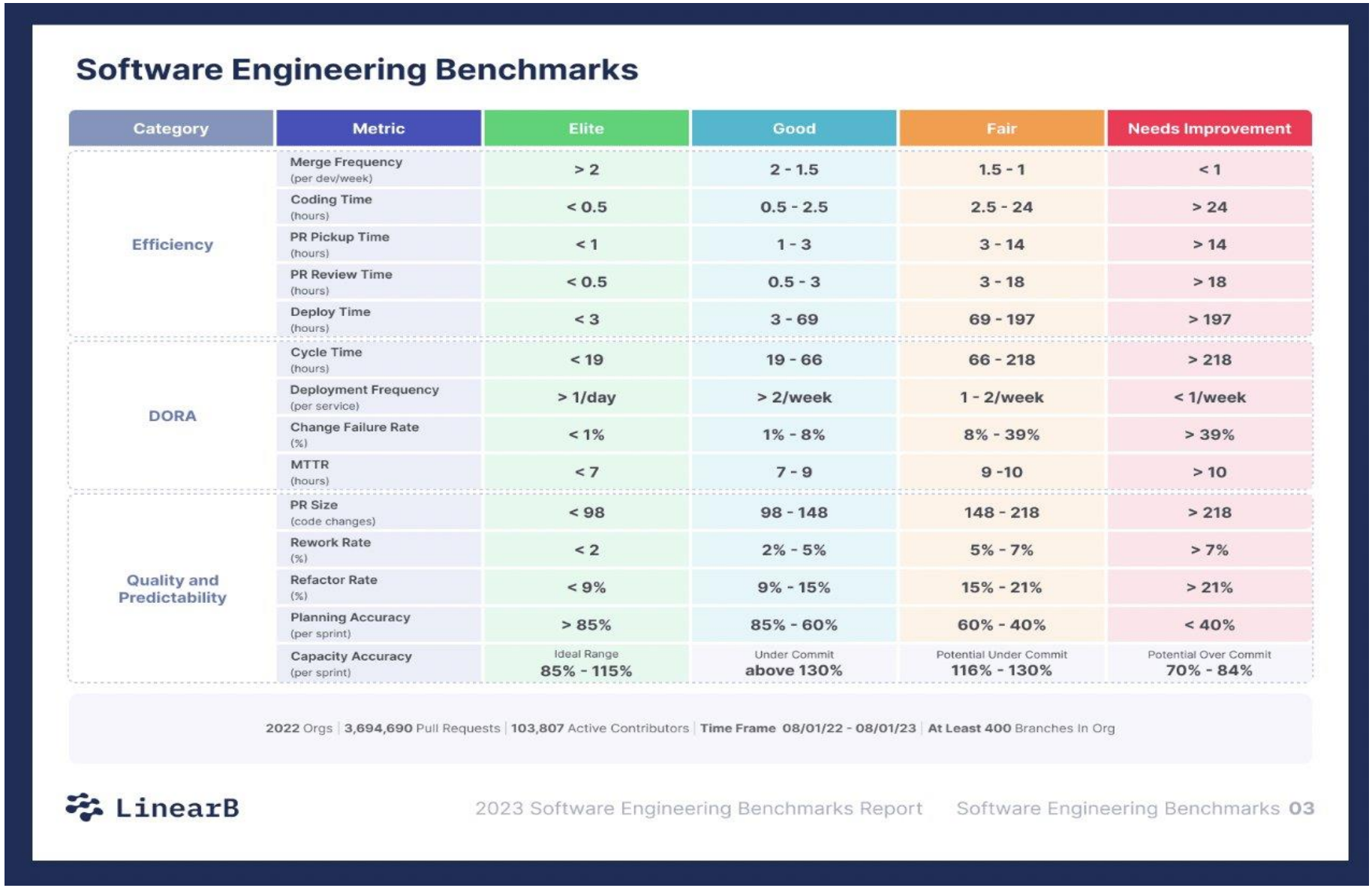
5.3.1 Roadmap implementation

- Finalize the technology roadmap by securing buy-in and support from teams, stakeholders, and relevant executives, ensuring the milestones are achievable.
- Coordinate timelines with impacted teams, prioritizing initiatives based on business impact and aligning them with the technology vision and the organization's broader goals.
- Document the roadmap on Confluence, schedule regular meetings to share progress, and adjust plans as needed to stay focused and on track towards the objectives.
- Track progress with key personnel and the project manager, ensuring milestones are met.
- Collect ongoing feedback from teams and individuals affected by the implementation of the technology roadmap.
- Evaluate the improvements, business impact, and cost savings achieved with each completed milestone.
- Communicate successes and key insights company-wide to motivate the team and those responsible for execution.

5.3.2 Empowering and developing the team

- Dedicate time to mentoring and nurturing the team, providing opportunities for professional growth.
- Collaborate with HR to create a clear and transparent career development path, outlining the skills, experience, and timelines required to advance from one role to another.
- Promotions and rewards should be actively sought by employees, while management should recognize contributions and provide timely rewards to boost morale and motivation. Employees should have a clear understanding of their potential career trajectory within the company over a five-year period.
- Cultivate a culture of honesty, openness, collaboration, creativity, innovation, first-principle thinking, constructive criticism, and continuous learning. Encourage a mindset of "fail fast and learn."

- Empower team members to take ownership, be passionate about their work, manage their deliverables, and make decisions confidently within their areas of expertise.
- Promote a culture where failure is seen as a step on the path to success, and learning from it is valued.
- Implement DORA metrics and Cyclic metrics to improve the quality of code base and repository and to make the engineering practices better. Identify the development team category and setup clear goals to attain the next level using cycle time metrics.



Software Engineering Benchmarks

Category	Metric	Elite	Good	Fair	Needs Improvement
Efficiency	Merge Frequency (per dev/week)	> 2	2 - 1.5	1.5 - 1	< 1
	Coding Time (hours)	< 0.5	0.5 - 2.5	2.5 - 24	> 24
	PR Pickup Time (hours)	< 1	1 - 3	3 - 14	> 14
	PR Review Time (hours)	< 0.5	0.5 - 3	3 - 18	> 18
	Deploy Time (hours)	< 3	3 - 69	69 - 197	> 197
DORA	Cycle Time (hours)	< 19	19 - 66	66 - 218	> 218
	Deployment Frequency (per service)	> 1/day	> 2/week	1 - 2/week	< 1/week
	Change Failure Rate (%)	< 1%	1% - 8%	8% - 39%	> 39%
	MTTR (hours)	< 7	7 - 9	9 -10	> 10
Quality and Predictability	PR Size (code changes)	< 98	98 - 148	148 - 218	> 218
	Rework Rate (%)	< 2	2% - 5%	5% - 7%	> 7%
	Refactor Rate (%)	< 9%	9% - 15%	15% - 21%	> 21%
	Planning Accuracy (per sprint)	> 85%	85% - 60%	60% - 40%	< 40%
	Capacity Accuracy (per sprint)	Ideal Range 85% - 115%	Under Commit above 130%	Potential Under Commit 116% - 130%	Potential Over Commit 70% - 84%

2022 Orgs | 3,694,690 Pull Requests | 103,807 Active Contributors | Time Frame 08/01/22 - 08/01/23 | At Least 400 Branches In Org

LinearB

2023 Software Engineering Benchmarks Report Software Engineering Benchmarks 03

Figure 9 Cycle time metrics to categorize the teams

5.4 6 months and greater – the ongoing process

5.4.1 Innovation and exploration

- Stay ahead of the curve by continuously monitoring emerging and rapidly evolving technologies, particularly in Web3, Blockchain, AI, and automation where the landscape shifts quickly.
- Empower your R&D team and keep senior developers and the R&D team informed about these advancements, providing them with dedicated time to evaluate, research, and test Proof of Concepts (POCs).
- This empowers them to effectively implement new technologies, ensuring your product is ready for a timely market launch, reaping the advantages of the new technology.

5.4.2 Focus areas

- **Communication and Transparency** – Maintaining an open communication with stakeholders, ensuring they understand the technology strategy and its progress.
- **Negotiations** – Always update the stakeholders upfront with the true picture is something is not going with the plan, always have Plan B and C and negotiate the right strategy with the stakeholders, to realign the roadmap.
- **Adaptability** – Always be prepared to adjust the roadmap and priorities based on changing business needs or unforeseen challenges.
- **Continuous Improvement** - Foster a culture of Openness, accepting criticism, continuous improvement, Ownership, Passion and Care within the technology team, encouraging feedback, learning, and adaptation.

5.5 Risk and Mitigation Strategy

- **Disruption to current operations** - Carefully plan and communicate changes, conduct thorough testing before deployment, and have rollback plans in place.
- **Resistance from the team** - Foster open communication, actively seek feedback, address concerns, and provide training and support for new technologies or processes.
- **Technical challenges** - Conduct thorough research and POCs, leverage external expertise if needed, and prioritize initiatives based on feasibility and risk.
- **Budget constraints** - Prioritize initiatives with the highest ROI, seek creative solutions, and negotiate with stakeholders if necessary.
- **Talent shortage** - Invest in recruiting and retention efforts, provide opportunities for growth and development, and consider outsourcing or partnering for specific skills if needed.
- **Resistance to Change** - Mitigation is to clearly communicate, address concerns, involve team members in the decision-making process.
- **Temporary Disruption** - Mitigation is to Phased implementation, clear communication, and support from management.
- **Skill Gaps** - Rapid technological advancements and evolving project requirements may have created skill gaps within the team. Mitigation is implementing Training programs, mentorship, and external hiring if necessary.

- **Siloed Teams** - Development, QA, and Operations teams might be working in isolation, leading to delays and miscommunication. Mitigation strategy is to make them cross-functional team with more collaboration.
- **Overload Bottlenecks -** Certain roles or individuals might be overloaded, leading to bottlenecks in the development process. Mitigation strategy is to have a proper resource planning, with the project manager, reprioritising tasks, hiring new resources, process optimisation, setting up realistic expectations, negotiations.
- **Lack of Ownership** - Team members might not feel a strong sense of ownership and accountability for the products they work on. Mitigation strategy is to define clear roles and responsibilities, transparent goals, open communications, empowerment and autonomy, delegate authority, constructive feedback, recognition and rewards, provide opportunities for growth.

5.6 Final words

- In the first 6 months, understanding the organisation, unique challenges and opportunities
- Building strong relationships and gaining trust are crucial in the early stages
- Focusing on delivering quick wins to demonstrate value and gain momentum, delivering tangible results
- Collaborating closely with the teams, and implementing strategies that align with your business goals
- Balancing short-term needs with long-term strategic goals
- Fostering a collaborative and supportive environment within the technology team, and across teams
- There is a lot of responsibility to support the existing product, to increase the user base and to lead the transition of the existing tech stack to the new tech stack.

6. Out of assignment

6.1 Emerging technologies under CTOs radar

As the CTO of a poker and casino game development company, several key emerging technologies should be on the radar:

6.1.1 Artificial Intelligence and Machine Learning

- **Enhanced Player Experience:** AI can be used to create more realistic and adaptive game opponents, tailor game experiences to individual player preferences, and provide personalized recommendations.
- **Fraud Detection and Prevention:** AI-powered systems can detect patterns and anomalies in player behaviour, helping to identify and prevent cheating, collusion, and other fraudulent activities.

- **Game Analytics and Optimization:** Machine learning algorithms can analyse vast amounts of gameplay data to gain insights into player behaviour, improve game design, and optimize monetization strategies.

6.1.2 Blockchain and Cryptocurrency

- **Secure and Transparent Transactions:** Blockchain technology can ensure the integrity and transparency of in-game transactions, providing players with greater trust and security.
- **Decentralized Gaming Platforms:** Blockchain-based platforms enable the creation of decentralized casinos and poker rooms, offering increased player autonomy and control.
- **NFT Integration:** Non-fungible tokens (NFTs) can represent unique in-game assets, allowing players to own, trade, and collect valuable digital items.

6.1.3 Virtual Reality (VR) and Augmented Reality (AR)

- **Immersive Gaming Experiences:** VR and AR have the potential to revolutionize the casino and poker experience, offering players a more realistic and interactive environment.
- **Social Gaming:** VR and AR can enable more engaging and social gameplay experiences, fostering a sense of community and interaction among players.

6.1.4 Cloud Gaming

- **Accessibility and Scalability:** Cloud gaming allows players to access high-quality casino and poker games on various devices without the need for powerful hardware, expanding the potential user base and reducing barriers to entry.
- **Reduced Development Costs:** Cloud gaming can streamline development and deployment processes, potentially reducing costs and time to market for new games.

6.1.5 5G and Edge Computing

Even with 6G technology around the corner with trials around 2028, we need to be ready to embrace the technology.

Low-Latency Gaming: 5G networks and edge computing offer the potential for ultra-low latency gameplay, crucial for delivering smooth and responsive experiences in real-time multiplayer casino and poker games.

6.1.6 Metaverse and Virtual Worlds

Integrated Gaming Experiences: As the metaverse concept evolves, casino and poker games could become integrated into larger virtual worlds, offering players a more immersive and social gaming experience.

6.1.7 Beyond Technology

Responsible Gambling: As the CTO, it is also important to consider the social and ethical implications of technology, especially regarding responsible gambling measures and player protection.

By staying informed about these emerging technologies and their potential impact on the industry, the CTO can ensure that his/her company remains competitive and innovative, offering players the most engaging and secure gaming experiences possible.

6.2 Reporting structure for CTO to stakeholders

6.2.1 Weekly Status Reports

Content: Updates on product development, feature releases, and any critical issues or blockers. Include key metrics like DAUs, WAUs, MAUs, retention rates, and incident reports.

6.2.2 Monthly Performance Reports

Content: Detailed analysis of application performance, uptime, and user engagement metrics. Include comparisons with previous months and identify trends. OPEX reports on the cloud that can vary month by month and the infrastructure cost, subscriptions.

6.2.3 Quarterly Strategic Reports

Content: Long-term planning, including roadmaps for both products, budget forecasts, and hiring needs. Present risks, opportunities, and potential pivots.

6.2.4 Security and Compliance Reports

Frequency: Quarterly or as needed.
Content: Findings from security audits, compliance checks, user experience management, and recommendations for mitigating risks.

6.3 Staying on top of technological advances

As a CTO in the iGaming and crypto space, staying on top of technological advances is paramount to maintaining a competitive edge and making informed decisions for your company. Here's a multifaceted approach I would adopt:

6.3.1 Active Learning & Exploration

- **Dedicated Time for Research:** I would allocate specific time each week to read industry publications, white papers, and research reports on the latest advancements in iGaming and crypto technologies.
- **Online Courses and Webinars:** Participate in online courses and webinars to deepen understanding of emerging technologies and their potential impact on the industry.
- **Technical Conferences & Meetups:** Attend industry conferences and meetups to network with other professionals, learn from experts, and gain insights into the latest trends and developments, such as GDC, ICE London, or blockchain-specific events.
- **Hands-on experimentation:** Allocate time to explore and experiment with new technologies in a sandbox environment, fostering a deeper understanding of their capabilities and limitations.

6.3.2 Leveraging Networks & Resources

- **Industry News and Publications:** Regularly read relevant industry news sources, blogs, and newsletters to stay abreast of emerging trends, regulatory changes, and technological breakthroughs.
- **Social Media and Online Communities:** Follow thought leaders, influencers, and companies on social media platforms and participate in online communities and forums to engage in discussions and share insights.

- **Advisory Boards and Consultants:** Utilize the expertise of advisory boards and consultants to gain access to specialized knowledge and insights on emerging technologies.
- **Partnerships and Collaborations:** Foster partnerships and collaborations with other companies and research institutions to stay at the forefront of technological innovation.

6.3.3 Internal Practices

- **Encourage a Culture of Learning:** Foster a culture of continuous learning and development within the technical team, encouraging them to share knowledge and explore new technologies.
- **Knowledge Sharing Sessions:** Organize regular brown bag knowledge-sharing sessions or tech talks where team members present their findings on new technologies or industry trends.
- **Hackathons and Innovation Challenges:** Encourage experimentation and innovation through hackathons and challenges, allowing team members to explore new ideas and technologies in a creative and collaborative setting.

6.3.4 Focused Areas

- **Blockchain & Cryptocurrency:** Stay abreast of developments in blockchain technology, smart contracts, decentralized finance (DeFi), and the evolving regulatory landscape surrounding cryptocurrencies.
- **Artificial Intelligence & Machine Learning:** Understand the applications of AI & ML in fraud detection, personalization, and game development.
- **Cloud Computing & Edge Computing:** Keep up with cloud computing trends and the potential of edge computing for enhancing gaming experiences and data processing.
- **Virtual Reality (VR) & Augmented Reality (AR):** Explore the advancements in VR and AR and their potential impact on immersive gaming and interactive experiences.
- **Cybersecurity:** Continuously monitor cybersecurity threats and vulnerabilities and stay informed about the latest security practices and technologies to safeguard user data and maintain the integrity of the platform.

6.4 Questions to be answered before choosing a new technology to solve the business problem

This template is few years old, but very effective. I have used this for a couple of tech stack evaluation and transition projects.
When rewriting a product with a new technology or switching over to new technology, several crucial questions need to be answered by the business to ensure the right choice is made. These questions can be grouped into the following categories:

6.4.1 Strategic Alignment

6.4.1.1 Business Goals

- What are the primary business objectives driving the decision to rewrite the product? (e.g., improve performance, scalability, security, user experience, or reduce costs)
- How will the new technology help achieve these business goals?
- Are there alternative approaches to achieve the same goals without a complete rewrite?

6.4.1.2 Market and Competitive Landscape

- How will the new technology impact the product's competitiveness and market positioning?

- Will it enable the company to enter new markets or target new customer segments?
- Are competitors adopting similar technologies?

6.4.2 Technical Feasibility and Impact

6.4.2.1 Technology Fit

- Is the new technology a good fit for the product's requirements and use cases?
- Does it address the existing pain points and limitations of the current technology stack?
- What are the potential risks and challenges associated with adopting the new technology?

6.4.2.2 Team Skills and Expertise

- Does the team have the necessary skills and expertise to work with the new technology effectively?
- If not, what training or hiring will be required?
- How will the new technology impact the team's productivity and morale?

6.4.2.3 Impact on Existing Systems

- How will the rewrite affect the integration with other systems and third-party services?
- Will there be any data migration or compatibility issues?
- What is the potential impact on existing customers and their data?

6.4.3 Financial and Resource Considerations

6.4.3.1 Cost-Benefit Analysis

- What are the estimated costs of the rewrite, including development, testing, deployment, and potential downtime?
- What are the expected benefits and return on investment (ROI)?
- How long will it take to recoup the investment?

6.4.3.2 Resource Allocation

- Do we have the necessary resources (budget, personnel, time) to complete the rewrite successfully?
- How will the rewrite impact other ongoing projects and priorities?
- What is the opportunity cost of investing in the rewrite versus other potential initiatives?

6.4.4 Risk Management and Mitigation

6.4.4.1 Risk Assessment

- What are the potential risks associated with the rewrite, such as technical challenges, delays, or unforeseen issues?
- How will these risks be mitigated or managed?
- What is the contingency plan if the rewrite encounters significant obstacles?

6.4.4.2 Transition Strategy

- How will the transition from the old to the new system be managed?

- Will there be a parallel run or a phased rollout?
- How will customer impact be minimized during the transition?

6.4.5 Long-Term Vision

6.4.5.1 Scalability and Maintainability

- Will the new technology support future growth and scalability?
- Is it easy to maintain and update, or will it create additional technical debt?
- Does it align with the company's long-term technology vision?

6.4.5.2 Innovation and Adaptability

- Does the new technology enable innovation and allow the company to adapt to future market trends and technological advancements?
- Will it provide a competitive advantage in the long run?

By carefully considering these questions and conducting a thorough analysis, the business can make an informed decision about whether to rewrite the product with a new technology and choose the most suitable option for achieving its strategic goals.