Client Precondition Report

Each project team will consist of 8–10 senior Computer Science students. Each student contributes approximately 120 hours over the semester, resulting in about 960–1,200 total team hours. Students have around 13 weeks of active development time. Students possess strong general programming skills and Git/GitHub experience but should not be expected to have prior industry-level experience with specialized frameworks or technologies. Advanced tools should be listed as skills to be learned, not required prerequisites.

1. Company Information

- **Company Name:** Innovative Configuration International (ICI)
- **Primary Contact Person:** Rahiq Majeed
- **Email Address:** rahiq.majeed@innovativeconfiguration.com
- **Website:** https://www.innovativeconfiguration.com/
- **Location / Time Zone:** Aptos, CA

2. Team Structure & Required Skills

a. Team Composition

We envision a collaborative team structure with students working across multiple system components. The team will operate as a unified development group with individuals contributing to backend services, frontend interfaces, game development, and data analytics based on their interests and strengths. We encourage pair programming and regular code reviews to foster knowledge sharing.

Expected collaboration includes:

- Weekly team meetings with ICI mentors
- Daily standups (self-organized by students)
- Sprint-based development cycles (2-week sprints)
- Collaborative GitHub workflow with pull request reviews

b. Required Baseline Skills

Students should have:

- General programming experience (preferably JavaScript/TypeScript or Python)
- Basic understanding of software development practices
- Familiarity with Git/GitHub for version control
- Willingness to learn new frameworks and cloud technologies
- Basic understanding of web development (HTML, CSS, JavaScript)

c. Skills to Be Developed During the Project

Students will gain hands-on experience with:

- Modern backend frameworks (Node.js/TypeScript)
- Frontend frameworks (React, Next.js)
- Database technologies (Supabase, PostgreSQL, Redis for caching)
- Cloud services and deployment (AWS, Docker)
- Microservices architecture patterns
- Real-time data streaming and WebSocket connections
- 3D graphics programming (Three.js, WebGL)
- Game development and physics engines

- Data analytics and visualization
- RESTful API design and GraphQL
- CI/CD pipelines and DevOps practices

3. Project Title & Acronym

- **Project Title:** GLAMBRANDI Commerce Platform Enhancement
- **Acronym:** GBCP

4. Project Overview & Problem Statement

The Problem:

Traditional discount systems are static and fail to engage customers meaningfully. Consumers receive generic coupons via email or apps, leading to low redemption rates and minimal brand engagement. The gaming industry has evolved through mechanical slot machines, electronic gaming, and internet gaming, but these have been limited to casino floors and restricted by cash-based models.

Background & Innovation:

ICI has developed a revolutionary cashless gaming paradigm approved by the Nevada Gaming Control Board (NV-GCB). The GLAMBRANDI platform replaces cash gambling with digital commerce-based rewards, where players engage with entertaining games to earn discounted coupons for products, services, and activities they already want to purchase. This system is legal in both casino and non-casino locations on casino property.

Target Users:

- Casino guests in hotel rooms, restaurants, pools, and retail areas
- Merchants seeking innovative customer engagement
- Platform administrators managing the ecosystem

Goals & Expected Impact:

- Transform static coupon systems into interactive gaming experiences
- Achieve 35% conversion improvement over traditional discounts
- Increase user engagement by 42%
- Drive 27% revenue growth for participating merchants
- Create a scalable platform serving 10,000+ concurrent users

5. Solution Overview & Core System Components

Current Implementation Status:

ICI has already built the foundation of GLAMBRANDI, including basic games (Slots, Dice, Roulette, Blackjack, Card Match), core transaction processing, foundational cybersecurity, and initial data collection. Students will build upon this existing infrastructure.

Core MVP Deliverables (13-week development window):

1. Enhanced Dashboard System

- Consumer Dashboard: Real-time game state display, coupon wallet with expiration tracking, transaction history, personalized recommendations, achievement system
- Merchant Dashboard: Real-time analytics, coupon management interface, revenue visualization, demographic insights

- Admin Dashboard: Platform-wide monitoring, user management, game configuration, system health alerts

2. Analytics & Telemetry System

- User behavior tracking and game performance metrics
- Video engagement metrics and conversion funnel analysis
- Real-time data processing pipeline
- Custom dashboard visualization layer

3. Backend API Enhancement

- RESTful and GraphQL APIs for flexible data access
- Real-time notification service
- Enhanced authentication and security
- API rate limiting and optimization

4. Games Development:

- Outdoor Games: Physics-based golf and tennis simulations with 3D graphics
- Turning Figma Designs into real playable games for users to get discounted coupons
- Machine Learning Integration: Predictive analytics for user behavior

6. Technical Considerations

Preferred Tech Stack:

- Frontend: React, Next.js, TypeScript
- Backend: Node.js, TypeScript, Express
- Database: Supabase (PostgreSQL), Redis for caching
- Video: WebRTC, HLS (HTTP Live Streaming), FFmpeg
- Cloud: AWS (CloudFront CDN, S3, EC2)
- DevOps: Docker, GitHub Actions

APIs & Tools:

- ICI will provide access to existing APIs and documentation
- OpenAPI/Swagger documentation for API reference
- Development environment setup with Docker

Performance Requirements:

- API response time: <100ms for 95th percentile
- Video streaming latency: <200ms initial load
- Support for 10,000+ concurrent users
- 99.9% uptime target

Security & Privacy:

- JWT authentication and OAuth 2.0
- Role-based access control (RBAC)
- Data encryption in transit and at rest
- GDPR and privacy compliance considerations

7. Innovation & Competitive Advantage

GLAMBRANDI represents a fundamental shift in digital commerce and customer engagement:

- **First cashless I-Gaming platform approved by Nevada Gaming Control Board**
- Replaces traditional gambling with value-driven commerce rewards
- Combines entertainment gaming with practical purchasing incentives
- Available anytime, anywhere on casino property (not just gaming floors)
- Scalable platform with real-time analytics and personalization
- Legal in non-gaming areas, expanding market reach significantly

8. Proposed Implementation Timeline (13 Weeks)

Week 1-2: Foundation & Setup

- Team onboarding and NDA signing
- Development environment setup
- Existing codebase familiarization
- Architecture review and technical specification finalization

Week 3-5: Core Dashboard Development

- Consumer dashboard UI and real-time features
- Merchant dashboard analytics
- Admin dashboard management tools
- WebSocket implementation for real-time updates

Week 6-8: Backend APIs & Game Development

- Turning Figma Designs into real playable games (CardMatch, Tennis)
- Enhanced API development (REST & GraphQL)
- Authentication and authorization refinement

Week 9-11: Analytics & Integration

- Telemetry system implementation
- Data pipeline setup
- Visualization dashboard creation
- System integration and end-to-end testing

Week 12-13: Testing, Optimization & Deployment

- Performance optimization and load testing
- Security testing and vulnerability assessment
- Documentation finalization
- Production deployment preparation
- Final presentation and demo

9. Deployment & Support Expectations

Infrastructure:

- ICI will provide AWS cloud infrastructure access
- Staging and production environments will be configured
- Docker containerization for consistent deployment

Access & Credentials:

- GitHub repository access
- AWS console access (managed via IAM roles)
- Database access (read/write permissions)
- API keys for third-party services

Documentation Requirements:

- API documentation (OpenAPI/Swagger format)
- Architecture diagrams and system documentation
- Deployment guides and runbooks
- User manuals for dashboards
- Training materials for platform administrators

10. Confidentiality & Intellectual Property

What This Means for Students:

We want you to learn and grow from this project while protecting ICI's competitive advantages. Here's what you need to know:

Confidentiality (NDA Required):

- All students must sign a standard NDA at the project start
- Protect proprietary information: game algorithms, business strategies, system architecture details, and API designs
- Don't discuss specific technical implementation details publicly until project completion

What You CAN Share:

- Include the project in your resume and portfolio
- Describe general technologies used (React, Node.js, video streaming, analytics)
- Discuss skills learned and your role on the team
- Present your work in academic settings (class presentations, final report)

What You CANNOT Share:

- Proprietary algorithms or unique system architectures
- Business strategies or market intelligence
- The actual source code on public GitHub (code remains in private ICI repository)

Code Ownership:

Code developed during this project becomes property of ICI. However, you will have full access for academic purposes and can reference your contributions in job applications and portfolios using the guidelines above.

11. Proposed Success Metrics

Technical Performance Metrics:

- API response time: <100ms for 95th percentile requests
- Video streaming latency: <200ms initial load time
- System uptime: 99.9% availability
- Concurrent user support: Successfully handle 1,000+ simultaneous users in testing
- Dashboard load time: <2 seconds for all dashboard pages

Functional Completeness:

- All three dashboard types (Consumer, Merchant, Admin) fully functional
- Video streaming successfully integrated with adaptive bitrate
- Analytics pipeline collecting and displaying real-time metrics
- Backend APIs documented and tested

Quality Metrics:

- 80% or higher code test coverage
- Zero critical security vulnerabilities
- Comprehensive documentation completed
- Successful demo to stakeholders

Learning Outcomes:

- Students gain production-level experience with modern tech stack
- Team successfully collaborates using industry-standard workflows
- Students develop portfolio-worthy project demonstrating real-world skills

12. Mentorship & Communication Plan

Meeting Cadence:

- Weekly team meetings with ICI mentors (1-1.5 hours)
- Bi-weekly sprint reviews and planning sessions
- Ad-hoc technical support as needed
- Mid-semester progress review with stakeholders

Communication Platforms:

- Slack: Primary communication channel (dedicated project workspace)
- Email: Formal communications and documentation
- Zoom: Weekly meetings and technical discussions
- GitHub: Code reviews, issue tracking, project management

Response Time Expectations:

- Slack messages: Within 24 hours on weekdays
- Email: Within 48 hours
- Critical issues: Same-day response during business hours

Mentorship Team:

- Jay Karmarkar (CEO): Founder of ICI
- Rahiq {Primary Contact} (Senior Developer): Technical mentor for implementation
- Sameer (Full-Stack Developer): Additional technical support and backend management

13. Additional Notes

Project Philosophy:

We believe in learning by doing. This project provides hands-on experience with production-grade systems while building something meaningful. Students will not just write code, they'll understand the 'why' behind architectural decisions, participate in technical discussions, and see their work deployed in a real commercial platform.

Why This Project is Special:

- Work on Nevada Gaming Control Board-approved innovation
- Build features that will serve thousands of real users
- Gain experience across full stack: frontend, backend, data, and cloud
- Collaborate with experienced industry professionals
- Create a portfolio piece that stands out to employers

What Makes Students Successful:

- Curiosity and willingness to learn new technologies
- Strong communication and teamwork skills
- Proactive problem-solving mindset
- Consistent attendance at meetings and active participation
- Comfortable asking questions and seeking help when needed

We're excited to work with the next generation of software engineers and help them build something truly innovative!