Project Report Section III Design and Project Goals Summary

Group 27: Karthik Kesavarapu, Victor Nguyen, Jesse Martinez, Syed Ahmed

The report for this milestone submission includes the project design and project issues sections of the ReWild software platform. Sections include design goals, current system design, proposed system design, additional design considerations, final system design, object design, open issues, off-the-shelf solutions, new problems, migration, risks, costs, waiting room, ideas for solutions, project retrospective, glossary, and index.

The Design Goals section establishes four pillars: usability and accessibility emphasizing user-friendly controls; responsiveness and performance targeting 30 FPS on student-grade hardware; reliability and fault tolerance addressing multiplayer complexity; and flexibility and extensibility ensuring long-term viability.

The Current System Design section explains that wildlife conservation awareness currently depends on passive models that don't engage young adults well. No existing platform combines active simulation gameplay with conservation education, giving the team freedom to choose a modern technology stack.

The Proposed System Design section contains four subsections introducing the MVC pattern, providing sequence diagrams for workflows, presenting three-tier architecture, and identifying five major subsystems: Authentication, Combat & Gameplay, Networking, Data Persistence, and UI Rendering.

The Additional Design Considerations section covers seven aspects: Hardware/Software Mapping, Persistent Data Management, Access Control implementing JWT-based authentication with RBAC, Global Software Control using Observer pattern, Boundary Conditions, User Interface, and Design Patterns including State, Observer, and Factory patterns.

The Final System Design section integrates all elements into a comprehensive Component Integration & Data Flow diagram showing client-server interaction and the multiplayer game loop.

The Object Design section covers core modules and five subsystems managing authentication, RBAC, gameplay mechanics using inheritance, AI using State Pattern, and educational content through ConservationFact models and JournalManager controller.

The Open Issues section identifies three challenges: Network Latency addressed through Client-Side Prediction and Server Reconciliation, Asset Pipeline Optimization implementing Lazy Loading and resource budgets, and Anti-Cheat Measures using Server Authority and file hashing.

The Off-the-Shelf Solutions section includes Ready-Made Products (Kivy/PyGame, SQL Database, Python, HTTPS/WebSocket), Reusable Components (Kivy's animations and Python libraries), and Products That Can Be Copied (multiplayer lobby patterns and boss AI systems).

The New Problems section identifies challenges including backend maintenance demands, performance strain, network dependencies, connection instability, combat learning curve, varying engagement with content, server resource constraints, security vulnerabilities, long-term maintenance questions, and risk of regressions.

The Migration to New Product section addresses user onboarding by minimizing Account Creation Friction and providing implicit Tutorialization through the first level. Data modification is not applicable since this is a greenfield project.

The Risks section identifies five risks: Server Downtime (moderate probability, high impact) mitigated through autoscaling; Save Data Corruption (low probability, high impact) mitigated through autosaves and rollback; Unbalanced Ecosystem (moderate probability, medium impact) mitigated through tuning and dashboards; Player Confusion (high probability, medium impact) mitigated through tutorials and Ecology Codex; and Incorrect Descriptions (moderate probability, low impact) mitigated through proofreading.

The Costs section explains that ReWild has a variety of hidden costs including Development Time of 500-900 hours for implementing gameplay systems, world building, AI systems, and multiplayer synchronization, plus Deployment & Testing Time of 40-120 hours for platform-specific builds and server deployment.

The Waiting Room section lists deferred features: Mobile Port for Android/iOS requiring UI redesign for touch controls, Voice Chat for multiplayer proximity chat, Procedural Generation for random dungeon layouts, PVP Mode requiring tighter network balancing, and Loot Trading requiring transactional security.

The Ideas for Solutions section outlines the technical approach: Python-Centered Development for simplification, Arcade for Core Gameplay handling rendering and physics, Kivy/KivyMD for UI creating consistent experience, Modular Code Architecture for scalability, and Usability Testing through regular playtesting sessions.

The Project Retrospective section reflects that the design phase effectively translated ReWild's goals into concrete technical specification. The Client-Server MVC architecture balances security, scalability, and maintainability. Detailed modeling and rigorous analysis suggest the project is feasible within the timeline. UML modeling provided a clear roadmap ensuring the team shares a unified vision.

The Glossary section provides definitions of technical terms used throughout the report. The References section cites UML Distilled by M. Fowler, the Paw Portal Project Report, and wage information from Indeed.com. The Index section provides comprehensive topic listings for easy navigation.

This concludes the comprehensive summary of the Design and Project Goals sections in the ReWild report.