Exposé : Arc Raiders Companion App

1. Projektziel und Problemstellung

Arc Raiders ist ein kommender PvPvE Extraction Shooter in einem Sci-Fi-Setting, welchen ich glücklicherweise bereits ausgibig (23 Stunden während eines 3-Tägigen Spieltests) testen konnte, der Elemente von Spielen wie Escape from Tarkov und Hunt: Showdown kombiniert. Spieler bilden Squads, um Missionen zu erfüllen, Ressourcen zu sammeln und feindliche Spieler sowie Roboter zu besiegen. Das Spiel bietet eine Vielzahl von Quests, Upgrades und komplexen Progressions-Systemen.

Arc Raiders bietet komplexe Progressions-Systeme mit:

- Zahlreichen Quest-Lines mit unterschiedlichen, Zielen, bei denen immer nur die aktiven angezeigt werden
- Ressourcen basierte Upgrade-Systeme für Workstations mit mehreren Stufen
- Knappem Inventar Platz und begrenzten Ressourcen
- Squad basiertes Gameplay, das Koordination und Kommunikation erfordert

Problem: Spieler haben wenig Überblick über ihren Fortschritt und wenig Hilfe bei dem erledigen von Quests, können schlecht Langfristig planen sowie Ressourcen nutzen. Squads haben Schwierigkeiten bei der Koordination, wenn jeder nur seine eigenen Ziele sieht und verfolgt.

Lösungsansatz: Externe Companion App, die Quest Tracking mit Tipps, Ressourcenkalkulationen und intelligente Routenoptimierung für Squads bereitstellt.

2. Seiten/Funktionen der App

2.1. Dashboard

- Personalisierte Übersicht mit Statistiken und Fortschritt
- Schneller Zugriff auf aktive Quests und Ressourcen aus einer Art Wishlist/Bedarf für Upgrades

2.2 Quest Tracking mit zwei Visualisierungsmöglichkeiten und Detailseiten

- Kanban Board (Active, locked, completed) mit Filter- und Suchfunktionalitäten
- Flow-/MetroChart Ansicht für visuelle Darstellung von Quest-Lines und Abhängigkeiten
- Detaillierte Quest-Seiten mit Zielen, Belohnungen und User Tipps mit einem voting System ähnlich zu reddit und optional mit reputations System
- Speicherung von Progress (local Storage ohne authentication, Supabase PostgresDB mit Authentifizierung)

2.3. Item Database

- Card-Grid Ansicht aller Items mit Filter- und Suchfunktionalitäten
- Item Tracking (Wishlist/Owned) mit Ressourcen Kalkulationen für Upgrades
- Verlinkung zu Quests und Spawn-Orten

2.4. (Optional) Arcdex (Kreaturendatenbank)

- Übersicht aller Kreaturen mit Details (Bilder, Loot, etc.)
- Verlinkung zu Quests und Spawn-Orten auf den Karten
- (Aktuelle Datenquelle bietet nicht die nötigen Daten)

2.5. Workstation Planner:

- Interaktive Level-Progress-Cards mit Upgrade-Buttons
- Gesamt Ressourcen Kalkulationen für Upgrades

2.6. Material-Calculator

- Multi-Select für Quests und Upgrades
- Abgleich mit Owned Liste der Materialien
- Speichern des Bedarfs in die Wishlist

2.7. Interaktive Karten

- Ähnlich zu https://mapgenie.io/tarkov/maps/woods
- Zusätzliche Selects für aktive Quest-Orte
- Squad Funktionalität, zum anzeigen der Quest-Orte von Squad Mitgliedern
- Selektive Routenoptimierung für Quests (im Squad) durch Wegefindungalgorithmus, optional mit Priorisierung (A*, Djikstra, gutes Thema für die Arbeit?)

3. Technischer Ansatz

Web-Applikation Mit React und Nextjs.

Hosting

- Vercel (Next.js optimiert, kostenlose Option für Hobby-Projekte bis zu ~50 aktive User)
- Supabase (Postgres DB, kostenlose Option für Hobby-Projekte)

Datenquelle:

Öffentliches Github Repository (nicht von den Entwicklern, aber von der Community gepflegt) https://github.com/RaidTheory/arcraiders-data. Diese werden aktuell über einen cron-job der über die Datenbank läuft in regelmäßigen Abständen in die relationale Datenbank importiert.

Da die Datenquelle nicht offiziell von den Entwicklern gepflegt wird, ist es möglich, dass nach Updates einige Daten fehlen oder unvollständig sind. Dies könnte die Funktionalität der App beeinträchtigen, dementsprechend wurden die folgenden Backup-Startegien geplant:

- Web-Scraping des offiziellen Fandoms (erlaubt laut deren robots.txt)
- User Contributed Data (Crowd-Sourcing), Frage des Aufwands und der Qualitätssicherung