Siemens Energy - BETI 2024

Realisierung eines Webauftritts

Socialmedia RPG

Echo

Ogulcan Kuecuek

Leon Woenckhaus

Nick Hildebrandt

Aaron Turyabahika

Andre Seiler

5. Februar 2025

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

CRUD engl. ”Create, Read, Update, Delete”, die Funktionen, die ein Nutzer benötigt, um Daten anzulegen und zu verwalten. 2

RPG engl. ”role-playing games”, sind ein Genre von Spielen, in denen die Spieler in die Rolle eines imaginären Charakters schlüpfen. 1, 2

SSL engl. ”Secure Sockets Layer”, ein Protokoll, das die Sicherheit der Kommunikation über das Internet gewährleistet. 3

SSR engl. ”Server Side Rendering”, ist Technik, Webseiten auf dem Server zu rendern, bevor sie an den Client-Browser gesendet werden. 2

XP engl. ”experience points”, sind ein Konzept in Spielen, das den Fortschritt eines Charakters oder Spielers zeigt. 1

1 Projektbeschreibung

Das Social Media Projekt „Echo“ ist ein innovatives, kompetitives Social Media Netzwerk, das speziell für Gamer, Digital Natives und Content Creator entwickelt wurde. Echo kombiniert die Elemente traditioneller Social Media Plattformen wie Reddit mit Rollenspiel-Mechaniken (engl. ”role-playing games”, sind ein Genre von Spielen, in denen die Spieler in die Rolle eines imaginären Charakters schlüpfen (RPG)), um ein dynamisches und interaktives Nutzererlebnis zu schaffen.

Nutzer sammeln Erfahrungspunkte (engl. ”experience points”, sind ein Konzept in Spielen, das den Fortschritt eines Charakters oder Spielers zeigt (XP)), indem sie Likes und Kommentare auf ihre Posts von anderen Nutzern erhalten. Diese XP sind ein Maß für die Aktivität und Beliebtheit eines Nutzers innerhalb der Plattform. Zusätzlich zu den XP können Nutzer durch sogenannte „Streaks“ weitere Erfahrungspunkte sammeln. Ein Streak entsteht, wenn ein Nutzer über mehrere aufeinanderfolgende Tage hinweg aktiv ist und regelmäßig Inhalte postet oder mit anderen interagiert. Je länger der Streak, desto höher die Belohnung.

Ebenfalls können Benutzer anderen Accounts folgen, um aus diesen personalisierte Inhalte zu bekommen und keine Neuigkeiten mehr zu verpassen. Dieser Wert an sogenannten Followern wird ebenfalls gezählt.

Die gesammelten Erfahrungspunkte ermöglichen es den Nutzern, ihr Profil und das Design der Website individuell anzupassen. Dies umfasst personalisierte Themes, exklusive Avatare und spezielle Layouts, die das Profil einzigartig machen. Ab einer gewissen Anzahl an XP können Nutzer ihr Profilbild, Bannerbild und Hintergrundbild selbst wählen, was zusätzliche Individualisierungsmöglichkeiten bietet.

Ein weiteres zentrales Element von Echo sind die sogenannten Communities, auf denen sich Nutzer sammeln können (Beitreten) und diese Awards durch andere Benutzer verliehen bekommen können, die im Profil angezeigt werden. Auf Communities gibt es genauso wie bei Benutzern die Möglichkeit, Posts mit Kommentaren und Likes zu versehen. Dies fördert die Interaktion und das Gemeinschaftsgefühl innerhalb der Plattform.

Im Mittelpunkt von Echo stehen Gamification-Elemente, Gruppenzugehörigkeit und das Belohnungsgefühl. Nutzer werden durch kontinuierliche Belohnungen und Fortschritte motiviert, aktiv zu bleiben und sich in der Community zu engagieren.

1.1 Projektbegründung

Das Social Media Projekt „Echo“ zeichnet sich durch seine einzigartigen Gamification- und Rollenspiel-Elemente (RPG) aus, die es zu einem besonderen Sammelpunkt für Gamer und die restliche Internetkultur machen. Diese Elemente fördern nicht nur die Interaktivität und das Engagement der Nutzer, sondern schaffen auch ein dynamisches und wettbewerbsorientiertes Umfeld, das die Nutzer motiviert, aktiv zu bleiben und sich kontinuierlich weiterzuentwickeln.

Ein weiteres technisches Highlight von Echo ist der innovative Caching-Algorithmus, der sicherstellt, dass Bilder nicht doppelt gespeichert werden. Beim Hochladen von Bildern werden doppelte Dateien erkannt und durch einen Verweis auf das bereits vorhandene Bild ersetzt. Diese Methode spart nicht nur wertvolle Speicherressourcen, sondern trägt auch zur Schonung der Umwelt bei. Durch die Reduzierung des Speicherbedarfs wird der Energieverbrauch der Server gesenkt, was wiederum den CO2-Ausstoß verringert und somit einen positiven Beitrag zum Umweltschutz leistet. Laut einer Studie des Borderstep Instituts für Innovation und Nachhaltigkeit verursachen Rechenzentren in Deutschland jährlich etwa 13 Millionen Tonnen CO2-Emissionen [1], was die Bedeutung ressourcenschonender Technologien unterstreicht.

Die Motivation für das Projekt war es, eine minimalistische und schnelle Plattform speziell für Gamer zu entwickeln, um die Gaming- und Internetkultur in einen diversifizierten multimedialen Austausch zu stellen. Echo kombiniert technologische Innovation mit einer klaren Zielgruppenausrichtung, um eine Plattform zu schaffen, die sowohl funktional als auch nachhaltig ist. Die Integration von Gamification- und RPG-Elementen macht Echo zu einem einzigartigen Erlebnis für Nutzer, während die ressourcenschonende Technologie die Umweltbelastung minimiert. Diese Kombination aus Innovation und Zielgruppenfokussierung macht Echo zu einer herausragenden Plattform im Bereich der sozialen Medien.

1.2 Projektabgrenzung

Das Social Media Projekt „Echo“ wird durch ein serverseitig gerendertes Frontend realisiert (engl. ”Server Side Rendering”, ist Technik, Webseiten auf dem Server zu rendern, bevor sie an den Client-Browser gesendet werden (SSR)), das eine nahtlose und schnelle Benutzererfahrung gewährleistet. Dazu wird eine API entwickelt, die Daten aus einer relationalen Datenbank über die CRUD-Operationen (engl. ”Create, Read, Update, Delete”, die Funktionen, die ein Nutzer benötigt, um Daten anzulegen und zu verwalten (CRUD)) zur Verfügung stellt. Diese Architektur ermöglicht eine effiziente und skalierbare Datenverwaltung, die den Anforderungen einer dynamischen Social Media Plattform gerecht wird.

Das Projekt wird umfassend dokumentiert, wie in diesem Dokument beschrieben, und zusätzlich durch eine Abschlusspräsentation ergänzt. Diese Präsentation wird die wichtigsten Aspekte und Ergebnisse des Projekts zusammenfassen und visuell ansprechend darstellen.

Für die Vorstellung des Projekts wird „Echo“ auf einem Server im Internet bereitgestellt und über eine eigene Domain mit einem entsprechenden SSL-Zertifikat ((engl. ”Secure Sockets Layer”, ein Protokoll, das die Sicherheit der Kommunikation über das Internet gewährleistet (SSL))) erreichbar gemacht. Dies stellt sicher, dass die Plattform sicher und zuverlässig zugänglich ist und den modernen Sicherheitsstandards entspricht.

2 Projektplanung

Zu Beginn des Projekts haben wir uns als Gruppe zusammengefunden und die grundlegenden Ideen und Funktionalitäten auf mehreren Flipchartblättern diskutiert. In intensiven Debatten haben wir die verschiedenen Aspekte des Projekts durchgesprochen, um am Ende einen sehr abstrakten Mockup zu erstellen, der zeigte, wie unser Projekt aussehen sollte und welche Funktionen bzw. RPG-Elemente es enthalten sollte.

Anschließend haben wir diese Funktionen in kleinere Issues aufgeteilt, die wie folgt beschrieben und aufgebaut sind: Eine kurze, prägnante Beschreibung des vorgeschlagenen Features, die erklärt, was es neu macht und warum es sinnvoll ist. Eine Liste der konkreten Funktionen, die das Feature umfassen wird. Eine Beschreibung des idealen Nutzerflusses für dieses Feature, die erklärt, wie der Nutzer mit dem Feature interagiert. Eine Auflistung der Technologien, die für die Frontend-Implementierung verwendet werden, sowie spezielle Designanforderungen, wie z.B. die Verwendung von Icons. Eine Erklärung, wie die Benutzereinstellungen in der Datenbank gespeichert werden, z.B. durch ein neues Dokument pro Benutzer mit den entsprechenden Feldern, sowie spezifische Anforderungen an die Datenverarbeitung oder Sicherheit.

Diese Issues dienen dabei gleichzeitig auch als Basisdokumentation der Funktionalität, aus der wir dieses Dokument technisch ableiten. Die Dokumentation selbst sowie die Präsentation werden über die Git-Versionskontrolle verwaltet und über Issues getrackt. Diese Issues wurden dank Kategorien wie Kernfunktionalität, optionale Funktionalität, Backend-API und Frontend zugewiesen. Zudem konnten einige Issues andere voraussetzen, wie z.B. dass ein Login die Registrierung voraussetzt. Mehrere Issues bildeten dann Meilensteine, die wir datieren konnten.

Da wir eine agile Arbeitsweise nach Scrum verwenden, gibt es tägliche Standups, in denen jeder sagt, was er gerade getan hat, welche Probleme es dabei gab und was er als nächstes machen wird. Die Entwicklungsarbeit und Versionsverwaltung wird dann durch Git realisiert, mit der Cloud-Implementierung von GitHub, um den Entwicklungsstand aus allen Computern zu synchronisieren. Git ist ein verteiltes Versionskontrollsystem, das es uns ermöglicht, Änderungen am Code effizient zu verfolgen und zusammenzuführen. Damit es zu keinen Konflikten in der gleichzeitigen Bearbeitung von ein und derselben Datei durch mehrere Leute kommt, wurde für jede Aufgabe ein eigener Entwicklungszweig erstellt. Diese wurden nach Beendigung wieder in den Main-Zweig zurückimplementiert, um eine lauffähige Version unseres Projekts stets im Main zu behalten.

Um einen Überblick über laufende und abgeschlossene Aufgaben zu haben, wurde auf GitHub ein Kanban-Board eingerichtet (3 Spalten: Offen, In Bearbeitung und Fertig), bei dem erledigte Aufgaben nach gemeinsamer Bewertung und Verbesserung auf „Fertig“

gestellt wurden. Nachdem wir zunächst die Backend-API gemeinsam mit dem Frontend bearbeitet hatten, mussten wir aufgrund der verschiedenen domänenspezifischen Anforderungen unseren Entwicklungsprozess umstellen und das Frontend und Backend parallel, aber getrennt durch verschiedene Teammitglieder entwickeln, um das jeweilige Können optimal zu allocieren.

2.1 Teamaufbau und Rollen

1. Projektmanagement
    - Nick Hildebrandt
2. Dokumentation
    - Leon Woenckhaus
3. Präsentation
    - Aaron Turyabahika
4. Backend-API
    - Nick Hildebrandt
    - Leon Woenckhaus
5. Frontend
    - Andre Seiler
    - Ogulcan Kuecuek
    - Aaron Turyabahika
6. Deploayment und integration
    - Nick Hildebrandt

2.2 Ressourcenplanung

Detaillierte Planung der benötigten Ressourcen (Hard-/Software, Räumlichkeiten usw.).

Ggfs. sind auch personelle Ressourcen einzuplanen (z.B. unterstützende Mitarbeiter).

Hinweis: Häufig werden hier Ressourcen vergessen, die als selbstverständlich ange- sehen werden (z.B. PC, Büro).

2.3 Kostenplanung

2.4 Zeitplanung

Cold freeze / Hard Freeze Daten In welchem Zeitraum und unter welchen Rahmenbedingungen (z.B. Tagesarbeitszeit) findet das Projekt statt?

Verfeinerung der Zeitplanung, die bereits im Projektantrag vorgestellt wurde.

Gant Diagramm

3 Zielplattform und Implementierung

Beschreibung der Kriterien zur Auswahl der Zielplattform (u.a. Programmiersprache, Datenbank, Client/Server, Hardware).

3.1 Architekturdesign

Beschreibung und Begründung der gewählten Anwendungsarchitektur (z.B. MVC).

Nucxt erklären (Komponenten also Vue und api)

Ggfs. Bewertung und Auswahl von verwendeten Frameworks sowie ggfs. eine kurze Einführung in die Funktionsweise des verwendeten Frameworks.

3.2 Benutzeroberfläche

Entscheidung für die gewählte Benutzeroberfläche (z.B. GUI, Webinterface).

Beschreibung des visuellen Entwurfs der konkreten Oberfläche (z.B. Mockups, Menüführung).

Ggfs. Erläuterung von angewendeten Richtlinien zur Usability und Verweis auf Corporate Design.

3.3 Datenmodell

Entwurf/Beschreibung der Datenstrukturen (z.B. ERM und/oder Tabellenmodell, XML-Schemas) mit kurzer Beschreibung der wichtigsten (!) verwendeten Entitäten.

Beschreibung der angelegten Datenbank (z.B. Generierung von SQL aus Modellie- rungswerkzeug oder händisches Anlegen), XML-Schemas usw.

3.4 Datenzugriff und Backend-API Routen

Zweck der Backend-API: Kurze Beschreibung, warum die API existiert und welche Aufgabe sie im Projekt erfüllt.

Systemkontext: Darstellung, wie das Backend in die Gesamtarchitektur eingebunden ist (z. B. Diagramm mit Datenbank, Frontend, API-Gateway).

Technologiestack: Beschreibung der eingesetzten Technologien (z. B. Programmiersprache, Frameworks, Datenbank).

Designmuster: Falls zutreffend, z. B. REST, GraphQL, Microservices, etc.

Endpunkte: Liste der API-Endpunkte (z. B. GET /users, POST /orders). Beschreibung des Zwecks jedes Endpunkts.

Request/Response-Formate: HTTP-Methoden (GET, POST, PUT, DELETE). Beispielanfragen und -antworten (JSON, XML, etc.). Fehlercodes und Fehlermeldungen.

Authentifizierung und Autorisierung: Beschreibung des Sicherheitskonzepts (z. B. OAuth, API-Keys, JWT). Zugriffsbeschränkungen und Rollen.

4 Abnahmephase

Welche Tests (z.B. Unit-, Integrations-, Systemtests) wurden durchgeführt und welche Ergebnisse haben sie geliefert (z.B. Logs von Unit Tests, Testprotokolle der Anwen- der)?

4.1 Bereitstellung

Welche Schritte waren zum Deployment der Anwendung nötig und wie wurden sie durchgeführt (automatisiert/manuell)?

4.2 Fazit

Wurde das Projektziel erreicht und wenn nein, warum nicht?

Ist der Auftraggeber mit dem Projektergebnis zufrieden und wenn nein, warum nicht?

Wurde die Projektplanung (Zeit, Kosten, Personal, Sachmittel) eingehalten oder haben sich Abweichungen ergeben und wenn ja, warum?

Hinweis: Die Projektplanung muss nicht strikt eingehalten werden. Vielmehr sind Abweichungen sogar als normal anzusehen. Sie müssen nur vernünftig begründet wer- den (z.B. durch Änderungen an den Anforderungen, unter-/überschätzter Aufwand).

Literatur

[1] B. Institut, „Rechenzentren in Deutschland: Eine Studie zu Energiebedarf und CO2-Emissionen,“ Borderstep Institut für Innovation und Nachhaltigkeit, 2020, Zugriff am 05. Februar 2025.