BITZ - Next Generation

Bitz ist eine innovative Online-Plattform, die sich auf den Kauf, Verkauf und Tausch von Technikprodukten spezialisiert hat. Unser Ziel ist es, Menschen zusammenzubringen, die ihre gebrauchte Technik verkaufen möchten, mit denen, die auf der Suche nach erschwinglichen und hochwertigen Geräten sind. Wir bieten eine benutzerfreundliche und sichere Umgebung, in der Transaktionen einfach und unkompliziert abgewickelt werden können. Durch unseren starken Fokus auf Technologie und unsere engagierte Community unterscheiden wir uns von anderen Online-Marktplätzen. Bei Bitz finden Sie eine große Auswahl an Technikprodukten zu wettbewerbsfähigen Preisen und profitieren von einem erstklassigen Kundenerlebnis. [infos zu personalisiertem Shop evtl. hinzufügen]

    1. Google Authentication is not working?
    2. Can't even start the application

7. Autoren

Getting Started

Kurzanleitung, um das Projekt von github zu holen und zu starten:

```
$ git clone https://github.com/tobiasmeyhoefer/bitz.git
$ cd bitz
$ npm i
$ npm run dev
```

Dies startet die Anwendung unter Verwendung von localhost:3000

Good to know while development

This project uses internationalization. Every string that is going to be internationalized should be written in the corresponding language json file and imported with the

- useTranslations Hook (for server side)
- getTranslations Hook (for async server side)
- as a prop for client side

Code Conventions

Pascal Case - Bennennung Komponenten Kleinbuchstaben mit bindestrich - Bennenung von Dateien

---

Language

Typescript

wurde als Programmiersprache für dieses Projekt ausgewählt, da es eine typsichere Variante von JavaScript ist. Im Gegensatz zu JavaScript, das eine dynamisch typisierte Sprache ist, bietet TypeScript eine statische Typisierung. Das bedeutet, dass Variablen, Funktionsparameter und Rückgabewerte mit spezifischen Datentypen versehen werden können. Durch diese zusätzliche Typisierung können viele Fehler bereits zur Kompilierzeit erkannt und vermieden werden, anstatt erst zur Laufzeit aufzutreten. Dies verbessert die Codequalität, Lesbarkeit und Wartbarkeit erheblich.

Ein weiterer Vorteil von TypeScript ist die verbesserte IDE-Unterstützung. Durch die statische Typisierung können IDEs und Code-Editoren eine bessere Code-Vervollständigung, Fehlerprüfung und Refaktorierung bereitstellen. Dies erleichtert die Entwicklung und reduziert die Wahrscheinlichkeit von Tippfehlern und anderen häufigen Fehlern.

TypeScript ist auch vollständig kompatibel mit JavaScript. Bestehender JavaScript-Code kann schrittweise in TypeScript migriert werden, und TypeScript-Code kann problemlos in JavaScript kompiliert und ausgeführt werden. Dies ermöglicht eine sanfte Einführung von TypeScript in bestehende Projekte und erleichtert die Integration mit anderen JavaScript-Bibliotheken und -Tools.

Für unser Projekt bietet TypeScript mehrere Vorteile. Durch die statische Typisierung können wir Fehler frühzeitig erkennen und beheben, was die Stabilität und Zuverlässigkeit unserer Anwendung verbessert. Die verbesserte IDE-Unterstützung erleichtert die Entwicklung und Zusammenarbeit im Team, da Entwickler schneller und präziser arbeiten können. Insgesamt trägt TypeScript dazu bei, eine robuste und wartbare Codebasis für unser Projekt zu schaffen.

Frontend Framework

Next.js v14+ (App Router):

Next.js wurde als Frontend-Framework für dieses Projekt gewählt, da es eine umfassende Lösung für die Entwicklung von React-Anwendungen bietet. Next.js baut auf React auf und erweitert es um zusätzliche Funktionen und Optimierungen.

Ein Hauptvorteil von Next.js ist das serverseitige Rendern (SSR). Beim SSR werden die React-Komponenten auf dem Server gerendert und als vollständiges HTML an den Client gesendet. Dies verbessert die Ladezeit und die Suchmaschinenoptimierung (SEO) der Anwendung, da der Inhalt sofort verfügbar ist und von Suchmaschinen leicht indexiert werden kann. Next.js kümmert sich automatisch um das serverseitige Rendern und bietet eine einfache Möglichkeit, dies in der Anwendung zu aktivieren.

Ein weiteres wichtiges Feature von Next.js ist das integrierte Routing. Next.js bietet ein dateibasiertes Routing-System, bei dem die Verzeichnisstruktur der Anwendung automatisch in Routen übersetzt wird. Dies ermöglicht eine intuitive und deklarative Definition von Routen, ohne dass zusätzliche Konfiguration erforderlich ist.

Mit der Einführung von Next.js v14+ und dem App Router wurde die Architektur und Leistung von Next.js-Anwendungen weiter verbessert. Der App Router ermöglicht eine klarere Trennung von Client- und Server-Komponenten und optimiert die Renderingleistung durch automatisches Code-Splitting und Lazy-Loading.

Für unser Projekt bietet Next.js mehrere Vorteile. Das serverseitige Rendern verbessert die Ladezeit und SEO unserer Anwendung, was zu einer besseren Benutzererfahrung und höheren Sichtbarkeit in Suchmaschinen führt. Das integrierte Routing vereinfacht die Navigation und Strukturierung unserer Anwendung. Mit dem App Router können wir eine skalierbare und performante Architektur aufbauen, die den Anforderungen unseres Projekts gerecht wird. Insgesamt bietet Next.js eine solide Grundlage für die Entwicklung unserer Anwendung und ermöglicht uns, effizient und effektiv zu arbeiten.

Three.js (für 3D-Komponenten):

Three.js ist eine leistungsstarke und weitverbreitete 3D-Bibliothek für JavaScript. Sie wurde in diesem Projekt integriert, um 3D-Elemente in der Anwendung darzustellen und interaktiv zu gestalten.

Mit Three.js können komplexe 3D-Szenen, Modelle und Animationen im Webbrowser erstellt und gerendert werden. Die Bibliothek bietet eine umfangreiche API und eine Vielzahl von Funktionen für die Erstellung von

3D-Grafiken, einschließlich Geometrien, Materialien, Beleuchtung, Kameras und Texturen.

Für unser Projekt bietet Three.js mehrere Vorteile. Durch die Integration von Three.js können wir ansprechende visuelle Elemente und interaktive 3D-Erfahrungen für unsere Benutzer bereitstellen. Beispielsweise können wir Produktvisualisierungen, virtuelle Rundgänge oder immersive Umgebungen erstellen, um das Benutzererlebnis zu verbessern und unsere Anwendung visuell ansprechender zu gestalten. Three.js ermöglicht uns, unsere Anwendung mit beeindruckenden 3D-Grafiken aufzuwerten und uns von Wettbewerbern abzuheben.

Ein weiterer Vorteil von Three.js ist die gute Leistung und Optimierungsmöglichkeiten. Die Bibliothek nutzt die Leistungsfähigkeit der WebGL-API, um hardwarebeschleunigte 3D-Grafiken im Browser zu rendern. Durch verschiedene Optimierungstechniken wie Level-of-Detail (LOD), Frustum-Culling und Texture-Komprimierung können wir die Renderingleistung verbessern und die Ladezeiten reduzieren, selbst bei komplexen 3D-Szenen.

Insgesamt ermöglicht uns Three.js, visuelle beeindruckende und interaktive 3D-Elemente in unsere Anwendung zu integrieren, ohne dabei Kompromisse bei der Leistung eingehen zu müssen. Durch die Verwendung von Three.js können wir unseren Benutzern ein unvergessliches und ansprechendes Erlebnis bieten und unsere Anwendung von der Konkurrenz abheben.

Zod (Formularvalidierung):

Zod ist eine leistungsstarke und typsichere Validierungsbibliothek für TypeScript. Sie wurde in diesem Projekt verwendet, um Formulareingaben zu validieren und sicherzustellen, dass die Daten den erwarteten Formaten und Regeln entsprechen.

Mit Zod können Validierungsschemas definiert werden, die die Struktur und die Einschränkungen der Daten beschreiben. Diese Schemas können einfache Typen wie Zeichenketten, Zahlen und boolesche Werte sowie komplexere Strukturen wie Objekte und Arrays umfassen. Durch die Verwendung von Zod-Schemas können Entwickler klar definieren, welche Daten erwartet werden und welche Regeln sie erfüllen müssen.

Für unser Projekt bietet Zod mehrere Vorteile. Durch die Integration von Zod können wir sicherstellen, dass die Formulareingaben unserer Benutzer korrekt validiert werden und den erwarteten Anforderungen entsprechen. Dies trägt dazu bei, die Datenintegrität zu wahren, Fehler zu reduzieren und eine bessere Benutzererfahrung zu bieten. Zod ermöglicht uns, präzise und flexible Validierungsregeln zu definieren, die auf unsere spezifischen Anforderungen zugeschnitten sind.

Ein großer Vorteil von Zod ist die nahtlose Integration mit TypeScript. Zod-Schemas können direkt aus TypeScript-Typen abgeleitet werden, wodurch eine konsistente und typsichere Validierung ermöglicht wird. Fehler werden zur Kompilierzeit erkannt, und die IDE-Unterstützung erleichtert die Entwicklung und Fehlerbehebung. Durch die Verwendung von Zod können wir die Vorteile von TypeScript voll ausschöpfen und eine robuste und zuverlässige Formularvalidierung implementieren.

Insgesamt trägt Zod dazu bei, die Datenqualität und Benutzererfahrung in unserem Projekt zu verbessern. Durch die Definition klarer Validierungsregeln und die nahtlose Integration mit TypeScript können wir sicherstellen, dass die Eingaben unserer Benutzer korrekt verarbeitet werden und die Anwendung reibungslos funktioniert.

next-intl (Internationalisierung)

next-intl ist eine Bibliothek für die Internationalisierung (i18n) in Next.js-Anwendungen. Sie wurde in diesem Projekt verwendet, um die Anwendung für verschiedene Sprachen und Regionen anzupassen und eine bessere Benutzererfahrung für ein globales Publikum zu bieten.

Die Internationalisierung ist ein wichtiger Aspekt für Anwendungen, die ein breites und vielfältiges Publikum ansprechen möchten. Durch die Unterstützung mehrerer Sprachen können Benutzer die Anwendung in ihrer bevorzugten Sprache nutzen, was die Zugänglichkeit und Benutzerfreundlichkeit verbessert.

Für unser Projekt bietet next-intl mehrere Vorteile. Durch die Verwendung von next-intl können wir unsere Anwendung einfach und effizient internationalisieren. Wir können Übersetzungen für verschiedene Sprachen definieren und verwalten, ohne den Code der Anwendung ändern zu müssen. next-intl bietet eine intuitive API zur Verwaltung von Übersetzungsschlüsseln und -werten, was die Pflege und Erweiterung der Übersetzungen erleichtert.

Ein weiterer Vorteil von next-intl ist die nahtlose Integration mit dem Next.js-Framework. Die Bibliothek nutzt die Funktionen und Konventionen von Next.js, wie beispielsweise das Routing und die serverseitige Rendering-Unterstützung. Übersetzungen können sowohl auf der Client- als auch auf der Serverseite abgerufen und gerendert werden, was eine konsistente und performante Internationalisierung ermöglicht.

Durch die Verwendung von next-intl können wir unsere Anwendung für ein globales Publikum zugänglich machen und die Benutzererfahrung verbessern. Wir können Übersetzungen für verschiedene Sprachen bereitstellen und die Anwendung an regionale Besonderheiten anpassen. Dies erweitert die Reichweite unserer Anwendung und ermöglicht es uns, Benutzer aus verschiedenen Ländern und Kulturen anzusprechen.

Insgesamt trägt next-intl dazu bei, unsere Anwendung zu internationalisieren und für ein globales Publikum zu optimieren. Durch die einfache Verwaltung von Übersetzungen und die nahtlose Integration mit Next.js können wir eine mehrsprachige Anwendung entwickeln, die den Anforderungen und Erwartungen unserer Benutzer gerecht wird.

Backend Frameworks

Drizzle (Object Relational Mapper):

Drizzle ist ein moderner und leistungsfähiger Object Relational Mapper (ORM) für TypeScript und JavaScript. Es wurde in diesem Projekt verwendet, um die Interaktion mit der Datenbank zu vereinfachen und typsichere Datenbankabfragen zu ermöglichen.

Ein ORM dient als Abstraktionsschicht zwischen der Anwendung und der Datenbank. Es ermöglicht Entwicklern, mit Datenbanken zu interagieren, indem sie mit Objekten und Klassen arbeiten, anstatt direkt SQL-Abfragen zu schreiben. Drizzle übernimmt die Übersetzung zwischen den Objekten und den entsprechenden Datenbanktabellen und -feldern.

Für unser Projekt bietet Drizzle mehrere Vorteile. Durch die Verwendung von Drizzle können wir die Interaktion mit der Datenbank vereinfachen und typsicher gestalten. Wir können Datenbankabfragen mit einer klaren und ausdrucksstarken Syntax definieren, die sich an der natürlichen Sprache orientiert. Dies verbessert die Lesbarkeit und Verständlichkeit des Codes und erleichtert die Zusammenarbeit im Team.

Ein weiterer Vorteil von Drizzle ist die Typsicherheit. Drizzle nutzt die Typen aus TypeScript, um sicherzustellen, dass die Datenbankabfragen korrekt typisiert sind und Fehler zur Kompilierzeit erkannt werden. Dies reduziert

die Wahrscheinlichkeit von Laufzeitfehlern und verbessert die Codequalität und -wartbarkeit. Durch die Verwendung von Drizzle können wir die Vorteile von TypeScript auch auf die Datenbankebene übertragen.

Drizzle unterstützt auch fortschrittliche Funktionen wie Migrations-Management, Transaktionen und Batch-Operationen. Migrationen ermöglichen eine kontrollierte Entwicklung und Versionierung des Datenbankschemas, während Transaktionen die Datenintegrität sicherstellen und Batch-Operationen die Leistung verbessern. Diese Funktionen erleichtern die Verwaltung und Skalierung unserer Datenbank im Laufe des Projekts.

Insgesamt trägt Drizzle dazu bei, die Entwicklung und Wartung unserer Datenbank zu vereinfachen und zu verbessern. Durch die typsichere und intuitive API können wir effizient mit der Datenbank interagieren und uns auf die Geschäftslogik konzentrieren. Drizzle bietet eine solide Grundlage für die Datenbankintegration in unserem Projekt und unterstützt uns dabei, eine skalierbare und zuverlässige Anwendung zu entwickeln.

Neon (serverless PostgreSQL-Datenbank):

Neon ist ein serverloser PostgreSQL-Datenbankdienst, der in diesem Projekt als Datenbankplattform verwendet wird. Durch den Einsatz von Neon können Entwickler die Vorteile einer leistungsstarken und skalierbaren PostgreSQL-Datenbank nutzen, ohne sich um die zugrunde liegende Infrastruktur und Verwaltung kümmern zu müssen.

Serverlose Datenbankdienste wie Neon abstrahieren die Komplexität der Datenbankbereitstellung, Skalierung und Verwaltung. Entwickler können sich auf die Anwendungsentwicklung konzentrieren, während Neon sich um die Bereitstellung, Skalierung und Überwachung der Datenbank kümmert.

Ein Hauptvorteil von Neon ist die automatische Skalierung. Wenn die Anwendung mehr Ressourcen benötigt, skaliert Neon die Datenbank automatisch hoch, um die Leistungsanforderungen zu erfüllen. Dies gewährleistet eine konsistente Leistung auch bei hoher Belastung und ermöglicht eine nahtlose Skalierung der Anwendung.

Neon bietet auch eine hohe Verfügbarkeit und Ausfallsicherheit. Die Datenbanken werden automatisch repliziert und in verschiedenen Verfügbarkeitszonen bereitgestellt, um eine kontinuierliche Verfügbarkeit zu gewährleisten. Im Falle eines Ausfalls sorgt Neon für eine automatische Failover-Funktion, um die Datenbank ohne Unterbrechungen verfügbar zu halten.

Ein weiterer Vorteil von Neon ist die einfache Integration mit anderen Cloud-Diensten und -Plattformen. Neon kann nahtlos mit verschiedenen Anwendungsarchitekturen und -frameworks zusammenarbeiten und ermöglicht eine reibungslose Entwicklung und Bereitstellung.

Durch die Verwendung von Neon in diesem Projekt wird eine skalierbare und zuverlässige Datenbankinfrastruktur bereitgestellt, ohne dass zusätzlicher Aufwand für die Verwaltung und Wartung erforderlich ist. Die Entwickler können sich auf die Anwendungslogik konzentrieren, während Neon eine leistungsstarke und flexible Datenbankplattform bietet.

Auth.js (Authentifizierung):

Auth.js ist eine umfassende Authentifizierungsbibliothek für Next.js-Anwendungen. Sie wurde in diesem Projekt integriert, um die Benutzerauthentifizierung und -autorisierung zu vereinfachen und verschiedene Anmeldemethoden wie Google, GitHub und Passkeys zu unterstützen.

Authentifizierung ist ein wesentlicher Bestandteil vieler Webanwendungen, um den Zugriff auf geschützte Ressourcen zu steuern und personalisierte Benutzererfahrungen zu ermöglichen. Auth.js bietet eine einfache und flexible Lösung für die Integration von Authentifizierungsfunktionen in Next.js-Anwendungen.

Ein Hauptvorteil von Auth.js ist die Unterstützung verschiedener Anmeldemethoden. Benutzer können sich mit ihren bestehenden Konten von Drittanbietern wie Google oder GitHub anmelden oder Passkeys verwenden, um eine sichere und passwortlose Authentifizierung zu ermöglichen. Auth.js kümmert sich um die Integration mit den entsprechenden Authentifizierungsdiensten und stellt die erforderlichen Schnittstellen und Callbacks bereit.

Auth.js bietet auch eine umfassende Sitzungsverwaltung. Nach erfolgreicher Anmeldung werden Benutzersitzungen erstellt und verwaltet, um den Authentifizierungsstatus über verschiedene Seitenanfragen hinweg beizubehalten. Die Sitzungsdaten können sicher auf dem Server gespeichert und bei Bedarf abgerufen werden.

Ein weiteres nützliches Feature von Auth.js ist die Rollen- und Berechtigungsverwaltung. Entwickler können verschiedene Benutzerrollen definieren und den Zugriff auf bestimmte Ressourcen oder Funktionen basierend auf diesen Rollen einschränken. Auth.js bietet Mechanismen, um Benutzerrollen zu überprüfen und die Autorisierung durchzusetzen, wodurch eine granulare Zugriffskontrolle ermöglicht wird.

Auth.js ist auch nahtlos in das Next.js-Ökosystem integriert. Es nutzt die Funktionen und Konventionen von Next.js, wie beispielsweise das serverseitige Rendern und die API-Routen, um eine effiziente und skalierbare Authentifizierungslösung bereitzustellen. Die Bibliothek kümmert sich um die sicheren Aspekte wie die Generierung und Überprüfung von Sitzungstoken, die Verschlüsselung sensibler Daten und die sichere Kommunikation zwischen Client und Server.

Durch die Verwendung von Auth.js in diesem Projekt wird die Implementierung der Authentifizierung und Autorisierung vereinfacht und beschleunigt. Entwickler können sich auf die spezifischen Anforderungen ihrer Anwendung konzentrieren, während Auth.js eine solide und sichere Grundlage für die Authentifizierung bietet. Die Unterstützung verschiedener Anmeldemethoden und die Möglichkeit der Rollenverwaltung bieten Flexibilität und Anpassungsfähigkeit für unterschiedliche Anwendungsfälle.

Styling

TailwindCSS (für Styling in der TSX-Syntax)

TailwindCSS ist ein utility-first CSS-Framework, das in diesem Projekt für das Styling der Benutzeroberfläche verwendet wird. Es ermöglicht das Schreiben von CSS direkt in der TSX-Syntax, was die Lesbarkeit und Wartbarkeit des Codes verbessert.

Im Gegensatz zu traditionellen CSS-Frameworks, die vordefinierte Komponenten und Stile bereitstellen, bietet TailwindCSS eine Sammlung von Utility-Klassen, mit denen Entwickler ihre eigenen Designs erstellen können. Diese Utility-Klassen repräsentieren einzelne CSS-Eigenschaften wie Abstände, Schriftgrößen, Farben und mehr. Durch die Kombination dieser Klassen können Entwickler schnell und flexibel benutzerdefinierte Designs erstellen, ohne CSS-Dateien manuell zu schreiben.

Ein Hauptvorteil von TailwindCSS ist die Verbesserung der Lesbarkeit und Wartbarkeit des Codes. Da das Styling direkt in der TSX-Syntax erfolgt, können Entwickler sofort erkennen, welche Stile auf ein bestimmtes

Element angewendet werden. Dies erleichtert das Verständnis und die Pflege des Codes, insbesondere in größeren Projekten mit vielen Komponenten.

TailwindCSS fördert auch die Wiederverwendbarkeit von Stilen. Durch die Verwendung von Utility-Klassen können ähnliche Stile leicht auf mehrere Elemente angewendet werden, ohne dass redundanter CSS-Code geschrieben werden muss. Dies führt zu einer konsistenteren und effizienteren Gestaltung der Benutzeroberfläche.

Ein weiterer Vorteil von TailwindCSS ist die umfangreiche Anpassungsmöglichkeit. Das Framework bietet eine Konfigurationsdatei, in der Entwickler ihre eigenen Farbpaletten, Schriftgrößen, Abstände und andere Design-Tokens definieren können. Dadurch kann das Erscheinungsbild der Anwendung leicht an die Markenrichtlinien oder spezifischen Anforderungen angepasst werden.

Durch die Verwendung von TailwindCSS in diesem Projekt wird das Styling der Benutzeroberfläche vereinfacht und beschleunigt. Entwickler können sich auf die Erstellung ansprechender und konsistenter Designs konzentrieren, ohne sich mit der Komplexität von CSS auseinandersetzen zu müssen. Die Lesbarkeit und Wartbarkeit des Codes werden verbessert, und die Anpassungsmöglichkeiten von TailwindCSS bieten Flexibilität bei der Gestaltung der Anwendung.

ShadCN (für bereits gut aussehende Komponenten)

ShadCN ist eine Sammlung von vorgestalteten UI-Komponenten, die in diesem Projekt verwendet werden, um schnell ansprechende Benutzeroberflächen zu erstellen. Diese Komponenten sind sorgfältig gestaltet und bieten ein professionelles und modernes Erscheinungsbild.

Einer der Hauptvorteile der Verwendung von ShadCN-Komponenten ist die Zeitersparnis bei der Entwicklung. Anstatt jede Komponente von Grund auf neu zu gestalten, können Entwickler die vorhandenen Komponenten von ShadCN nutzen und sie nahtlos in ihre Anwendung integrieren. Diese Komponenten sind bereits auf Benutzerfreundlichkeit, Barrierefreiheit und visuelle Konsistenz ausgelegt, was die Entwicklungszeit erheblich verkürzt.

Die ShadCN-Komponenten sind auch hoch anpassbar. Sie bieten eine Vielzahl von Konfigurationsoptionen und Stilvarianten, mit denen Entwickler das Erscheinungsbild der Komponenten an ihre spezifischen Anforderungen anpassen können. Durch die Anpassung von Farben, Schriftarten, Abständen und anderen Stilparametern können die Komponenten nahtlos in das Gesamtdesign der Anwendung integriert werden.

Ein weiterer Vorteil von ShadCN ist die Konsistenz des Designs. Da die Komponenten von professionellen Designern entworfen wurden, weisen sie eine einheitliche Ästhetik und ein stimmiges Erscheinungsbild auf. Durch die Verwendung dieser Komponenten in der gesamten Anwendung wird eine visuelle Kohärenz erreicht, die zu einer besseren Benutzererfahrung beiträgt.

Die ShadCN-Komponenten sind auch für die Verwendung mit modernen Frontend-Frameworks wie React optimiert. Sie sind leicht zu integrieren und bieten eine intuitive API für die Interaktion und Konfiguration. Entwickler können die Komponenten mit minimaler Konfiguration in ihre Anwendung einbinden und von deren Funktionalität profitieren.

Durch den Einsatz von ShadCN-Komponenten in diesem Projekt wird die Entwicklung der Benutzeroberfläche beschleunigt und vereinfacht. Entwickler können sich auf die Implementierung der Geschäftslogik und der spezifischen Anforderungen konzentrieren, während die vorgestalteten Komponenten ein ansprechendes und

professionelles Erscheinungsbild bieten. Die Anpassungsmöglichkeiten und die Konsistenz des Designs tragen zu einer verbesserten Benutzererfahrung und einer effizienten Entwicklung bei.

Figma (für Styling und Corporate Design)

Figma ist ein kollaboratives Designtool, das in diesem Projekt für die Erstellung des visuellen Designs und des Corporate Designs verwendet wird. Es ermöglicht Designern und Entwicklern, gemeinsam an der Gestaltung der Benutzeroberfläche zu arbeiten und eine konsistente visuelle Identität für die Anwendung zu schaffen.

Ein Hauptvorteil von Figma ist die nahtlose Zusammenarbeit zwischen Designern und Entwicklern. Figma bietet einen gemeinsamen Arbeitsbereich, in dem Designs erstellt, geteilt und kommentiert werden können. Designer können Prototypen und Mockups erstellen, während Entwickler Zugriff auf die Designdateien haben und daraus Stilinformationen und Assets extrahieren können. Diese Zusammenarbeit fördert eine effiziente Kommunikation und stellt sicher, dass das Design korrekt in die Anwendung übertragen wird.

Figma bietet auch eine umfangreiche Bibliothek von Designkomponenten und Stilen. Designer können wiederverwendbare Komponenten erstellen, die das Corporate Design der Anwendung widerspiegeln. Diese Komponenten können dann in verschiedenen Teilen der Anwendung konsistent verwendet werden, um ein einheitliches Erscheinungsbild zu gewährleisten. Änderungen an den Designkomponenten werden automatisch in allen Instanzen aktualisiert, was die Pflege und Aktualisierung des Designs erleichtert.

Ein weiterer Vorteil von Figma ist die Möglichkeit, Prototypen zu erstellen und zu testen. Designer können interaktive Prototypen erstellen, die das Verhalten und die Navigation der Anwendung simulieren. Diese Prototypen können mit Stakeholdern und Benutzern geteilt und getestet werden, um frühzeitiges Feedback zu erhalten und iterative Verbesserungen vorzunehmen. Durch die Erstellung von Prototypen können potenzielle Designprobleme frühzeitig erkannt und behoben werden, bevor sie in die Entwicklungsphase übergehen.

Figma unterstützt auch die Erstellung von Design-Spezifikationen und Styleguides. Designer können detaillierte Anweisungen und Richtlinien für die Verwendung von Farben, Schriftarten, Abständen und anderen Designelementen bereitstellen. Diese Spezifikationen dienen als Referenz für Entwickler und stellen sicher, dass das Design konsistent und genau in der Anwendung umgesetzt wird.

Durch die Verwendung von Figma in diesem Projekt wird eine enge Zusammenarbeit zwischen Design und Entwicklung gefördert. Das visuelle Design der Anwendung kann effizient erstellt, getestet und iteriert werden. Die Erstellung von wiederverwendbaren Designkomponenten und Stilen gewährleistet eine konsistente visuelle Identität und erleichtert die Pflege des Designs. Insgesamt trägt Figma dazu bei, eine ansprechende und benutzerfreundliche Oberfläche zu schaffen, die den Anforderungen und Erwartungen der Benutzer entspricht.

Weitere Tools

GitHub + Git (Versionskontrolle): GitHub und Git wurden für die Versionskontrolle des Projekts verwendet. Sie ermöglichen die Zusammenarbeit im Team, die Verfolgung von Änderungen und die Verwaltung verschiedener Versionen des Codes. Vercel (Deployment): Vercel ist eine Plattform für das Deployment von Next.js-Anwendungen. Sie wurde gewählt, um eine einfache und automatisierte Bereitstellung der Anwendung zu ermöglichen. Discord (Kommunikation): Discord wurde als Kommunikationsplattform für das Team verwendet. Es ermöglicht den Austausch von Nachrichten, Dateien und Informationen in Echtzeit.

Notion (Organisation): Notion ist ein Kollaborations- und Organisationstool. Es wurde verwendet, um Aufgaben zu verwalten, Dokumente zu erstellen und Informationen zu organisieren.

Troubleshooting

Die Google-Authentifizierung funktioniert nicht?

Dies könnte daran liegen, dass die Anwendung nicht auf localhost:3000 läuft. Dies kann z.B. passieren, wenn Sie bereits eine Anwendung auf localhost:3000 laufen lassen, ein anderer möglicher Grund ist, dass das Konto, das Sie verwenden, bereits mit einem Magic Link-Konto verwendet wird

Die Anwendung kann nicht gestartet werden.

Überprüfen Sie die installierte Node-Version, sie sollte nicht kleiner als v18 sein.

Autoren

Anna Laves s78700@bht-berlin.de, (891023)
Anton Kripp s88371@bht-berlin.de, (936120)
Dennis Blömeke s87697@bht-berlin.de, (929261)
Lucas Knäuper s67752@bht-berlin.de, (864306)
Niko Budic s87786@bht-berlin.de, (936244)
Tobias Meyhöfer s87766@bht-berlin.de (933280)