

Prototyping Projektdokumentation

Name: Anastasija Gojkovic

E-Mail: gojkoana@students.zhaw.ch

URL der deployten Anwendung: <https://vaultofgames.netlify.app/>

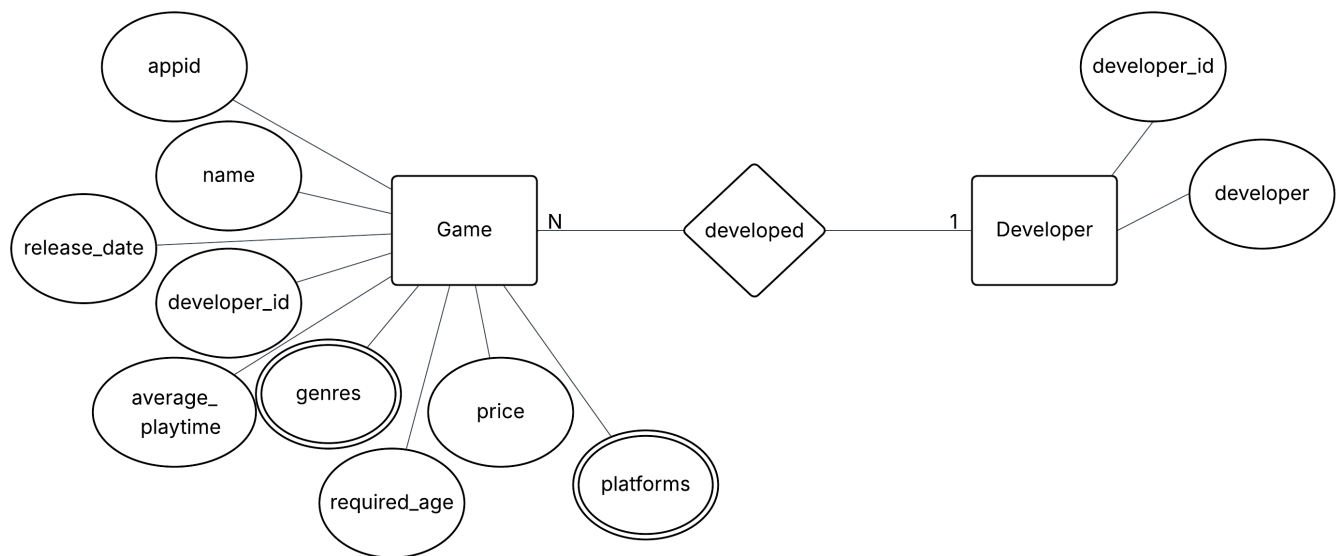
1. Einleitung

GameVault ist eine webbasierte Anwendung zur Verwaltung und Darstellung einer umfangreichen Spielesammlung. Die Idee hinter der App ist es, eine übersichtliche und benutzerfreundliche Plattform zu bieten, auf der Informationen zu vielen Videospielen abgerufen und verwaltet werden können. Die Anwendung richtet sich an Nutzer, die sich einen Überblick über verfügbare Spiele verschaffen wollen.

Zu den Grundfunktionen gehören:

- Eine Übersicht aller Spiele mit Bild, Genre, Plattform und Preis
- Eine Detailansicht pro Spiel mit zusätzlichen Informationen
- CRUD-Funktionen für Spiele
- Eine eigene Entwickler-Übersicht, inklusive Detailansicht und Verknüpfung mit den zugehörigen Spielen
- Pagination zur strukturierten Navigation durch grosse Datenmengen

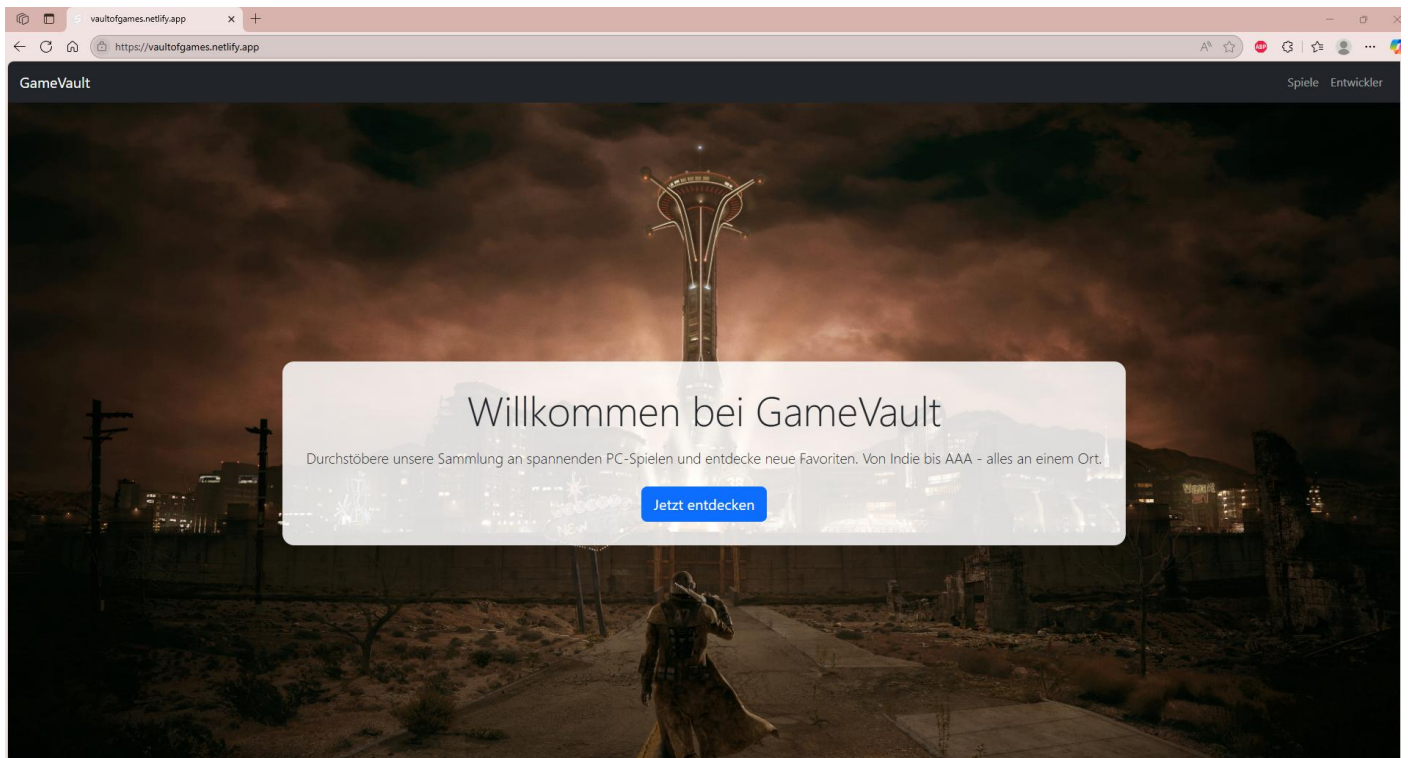
2. Datenmodell



3. Beschreibung der Anwendung

3.1. Startseite und Navigation

Route: /



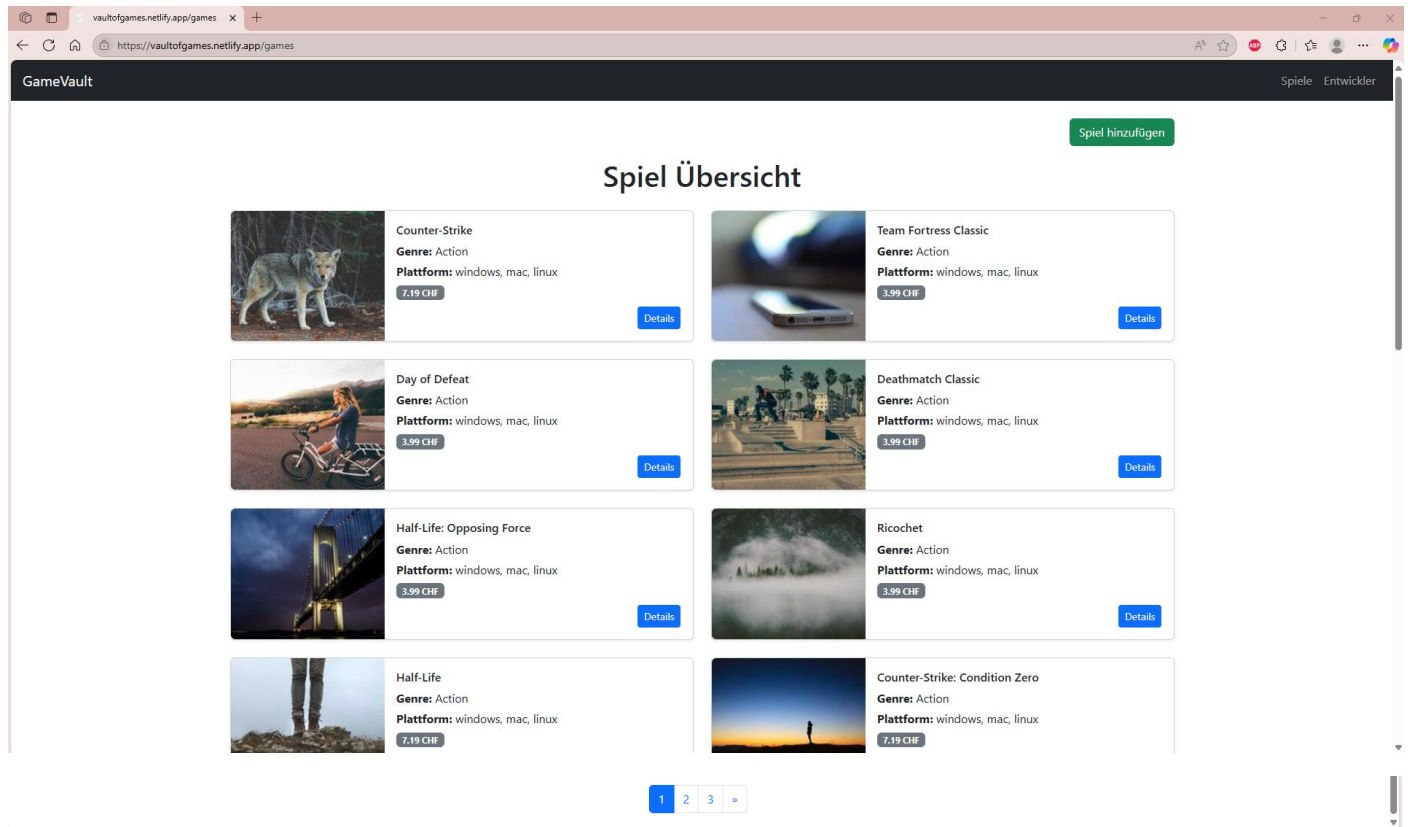
Die Startseite der Anwendung zeigt einen Willkommenstext und ein Hintergrundbild. Sie dient als visuelles Intro zur Anwendung. Sie enthält keine interaktiven Funktionen, sondern leitet Benutzer weiter zu den Bereichen Spiele oder Entwickler über die Navigation. Die Navigation ist von jeder Seite aus ersichtlich. Sie ist auch ausgelagert als Komponente `Navbar.svelte` und wird im `+layout.svelte` eingebunden. Die Navigations-Leiste enthält Links zur Startseite (GameVault oder 'Jetzt Entdecken', «/»), Spieleübersicht (Spiele, «/games»), Entwicklerübersicht (Entwickler, «/developers»).

Dateien:

- `lib/components/Navbar.svelte`
- `static/images/fallout_wallpaper.jpg`
- `routes/+layout.svelte`
- `routes/+page.svelte`

3.2. Gameübersicht

Route: /games



Auf dieser Seite werden alle Spiele in einer Kartenbasierten Übersicht dargestellt. Die Darstellung erfolgt über die Komponente `GameCard.svelte`. Es werden jeweils 20 Spiele pro Seite angezeigt. Mithilfe der eingebauten Pagination kann durch verschiedene Seiten geblättert werden. Die Pagination ist als Komponente `Pagination.svelte` ausgelagert. Jede Karte enthält ein Bild (Platzhalterbild von <https://picsum.photos>), den Namen, Genre, Plattform, Preis sowie die durchschnittliche Spielzeit. Über den Button «Details» gelangt man auf die Detailansicht eines Spiels unter `/games/[appid]`.

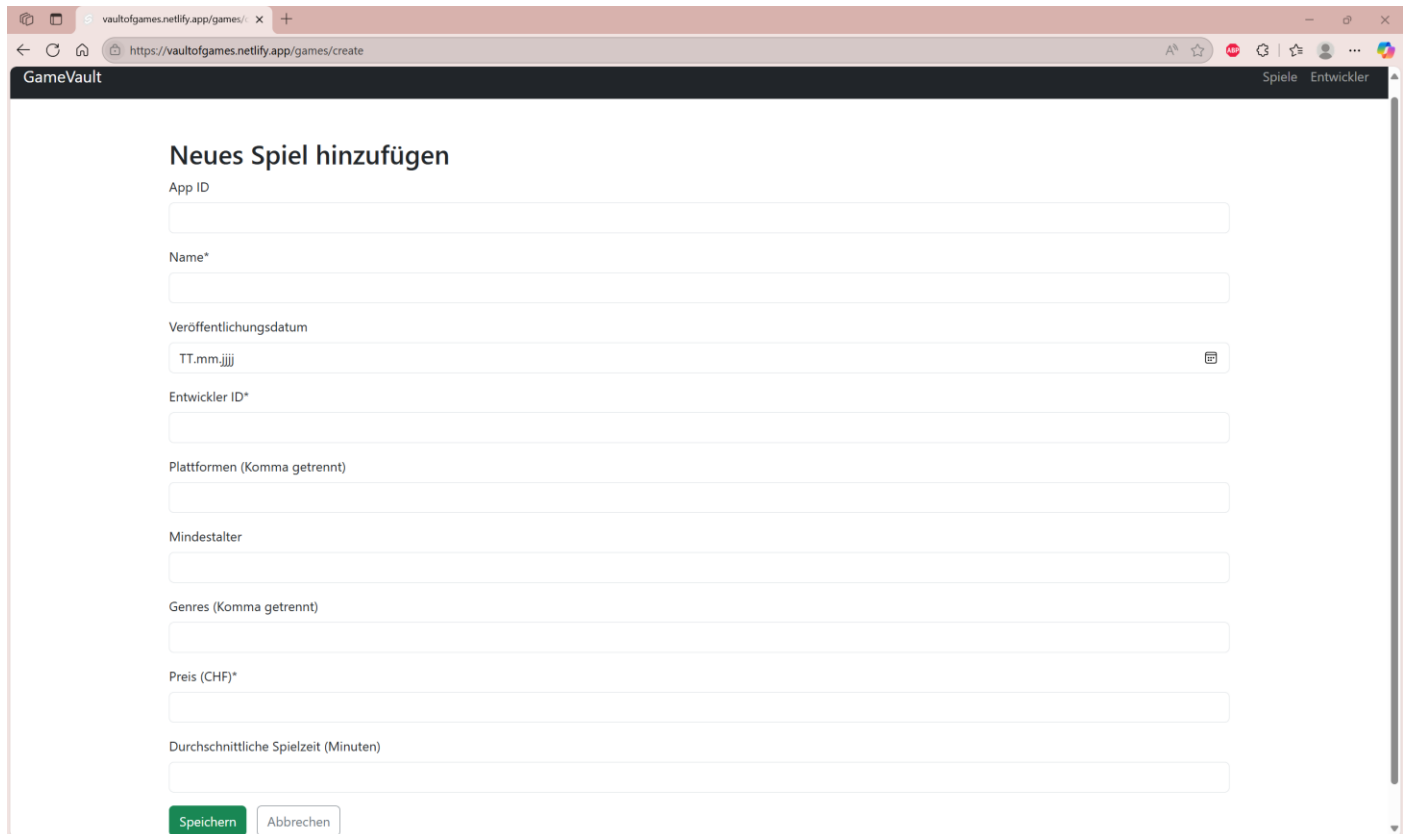
Oben rechts befindet sich ein Button «Spiel hinzufügen», der zum Erstellungsformular unter `/games/create` weiterleitet.

Dateien:

- `lib/components/GameCard.svelte`
- `lib/components/Pagination.svelte`
- `lib/server/games.js`
- `routes/games/+page.svelte`
- `routes/games/+page.server.js`

3.3. Spiel hinzufügen

Route: /games/create



The screenshot shows a web browser window with the URL `https://vaultofgames.netlify.app/games/create`. The page title is 'GameVault' and the navigation bar includes 'Spiele' and 'Entwickler'. The main heading is 'Neues Spiel hinzufügen'. The form contains the following fields:

- App ID
- Name*
- Veröffentlichungsdatum (with a date picker icon)
- Entwickler ID*
- Plattformen (Komma getrennt)
- Mindestalter
- Genres (Komma getrennt)
- Preis (CHF)*
- Durchschnittliche Spielzeit (Minuten)

At the bottom, there are two buttons: 'Speichern' (green) and 'Abbrechen' (grey).

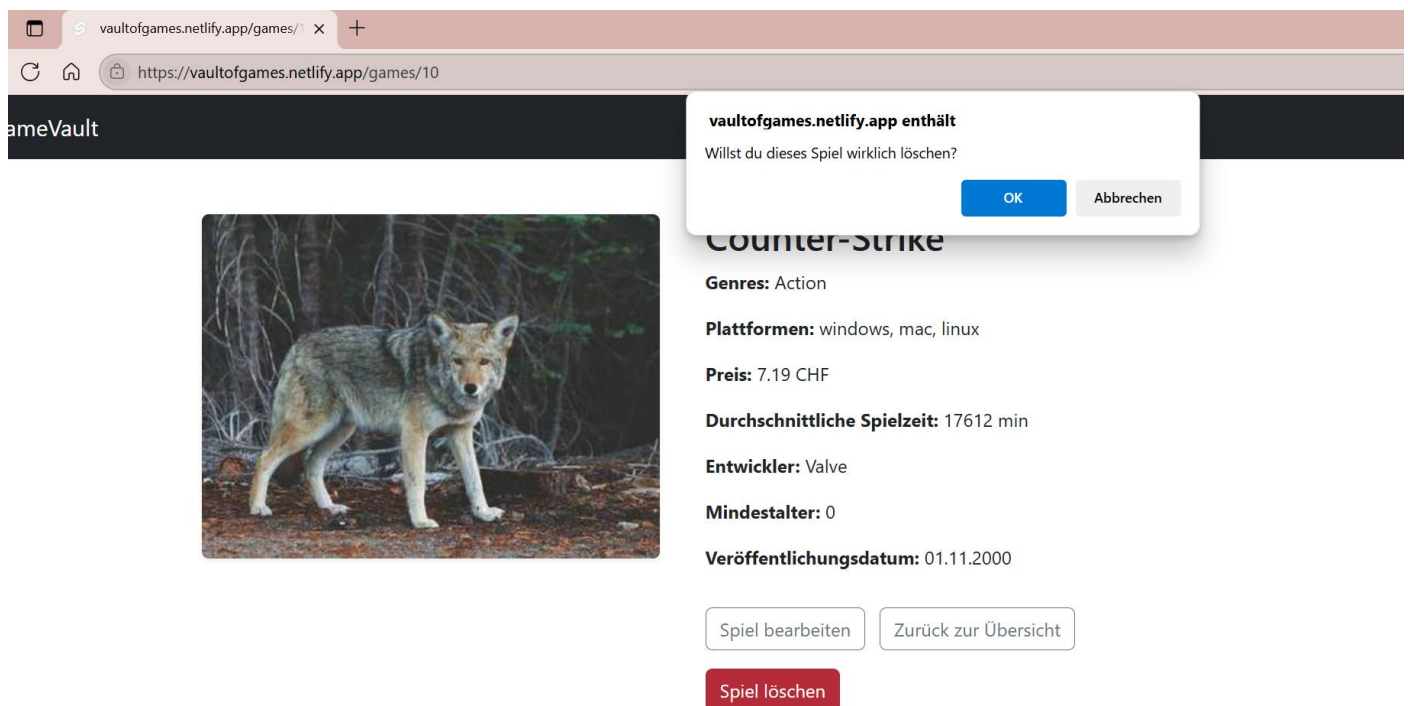
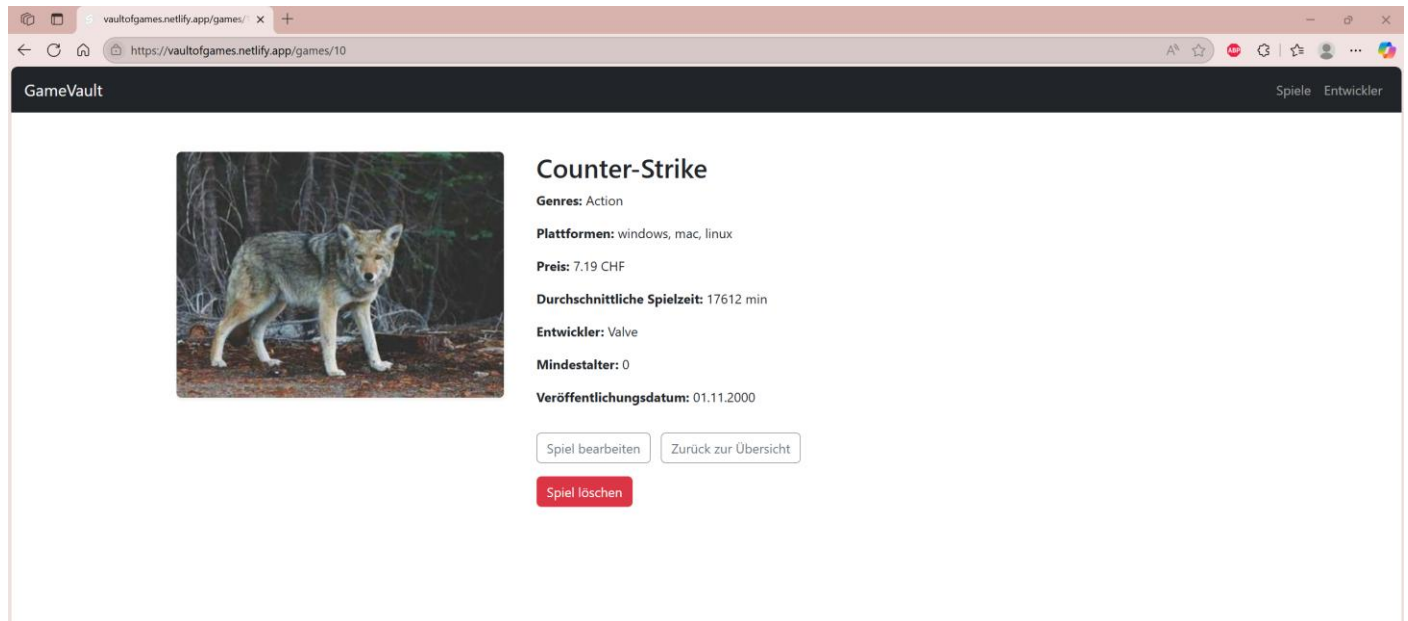
Auf dieser Seite kann ein neues Spiel erstellt werden. Das Formular ist vollständig clientseitig validiert und umfasst alle relevanten Felder eines Spiels. Die Eingabefelder für Plattformen und Genres sind als kommasetrennte Texteingabe gelöst, welche serverseitig in Arrays umgewandelt werden. Pflichtfelder sind App ID, Name, Entwickler ID und Preis. Nach erfolgreicher Speicherung wird der Benutzer zurück auf die Spieleübersicht geleitet.

Dateien:

- `routes/games/create/+page.svelte`
- `routes/games/create/+page.server.js`
- `lib/server/games.js`

3.4. Game Detailansicht

Route: /games/[appid]



Auf dieser Seite werden alle wichtigen Informationen zu einem einzelnen Spiel angezeigt. Die Detailseite wird dynamisch anhand der appid des Spiels generiert. Falls kein Spiel zur übergebenen appid gefunden wird, erscheint eine Fehlermeldung. Angezeigt werden Name, Genres, Plattformen, Preis, Durchschnittliche Spielzeit, Entwicklernamen (wird per developer_id aus der Developer-Collection geladen), Veröffentlichungsdatum, Mindestalter und das Bild (Platzhalter). Der Entwicklername ist ein Link der zur Entwickler Detailansicht auf /games/developer/[developer_id] führt. Dazu sind auch 3 Buttons auf der Detailansicht. Bearbeiten führt zur Edit Seite (/games/[appid]/edit), Löschen löscht das Spiel und leitet zur Übersicht um und 'Zurück zur Übersicht' leitet wieder zur Spieleübersicht. Wenn ein Spiel nicht gefunden wird, wird der Fehlerfall abgefangen und es zeigt die 404 Meldung an «Spiel nicht gefunden». Wird der «Spiel Löschen» Button gedrückt, erscheint eine Warnmeldung (JavaScript Confirm). Nach Bestätigung wird das Spiel über ein POST-Request gelöscht und der Benutzer wird zurück auf die Übersicht /games geleitet.

Dateien:

- Routes/games/[appid]/+page.svelte
- Routes/games/[appid]/+page.server.js
- Lib/server/games.js

3.5. Game bearbeiten

Route: /games/[appid]/edit

GameVault

Spiel bearbeiten: Counter-Strike

App ID

10

Name

Counter-Strike

Veröffentlichungsdatum

01.11.2000

Entwickler ID

1

Plattformen (durch Semikolon getrennt)

windows;mac;linux

Genres (durch Semikolon getrennt)

Action

Mindestalter

0

Preis (CHF)

7.19

Durchschnittliche Spielzeit (min)

17612

Änderungen speichern Abbrechen

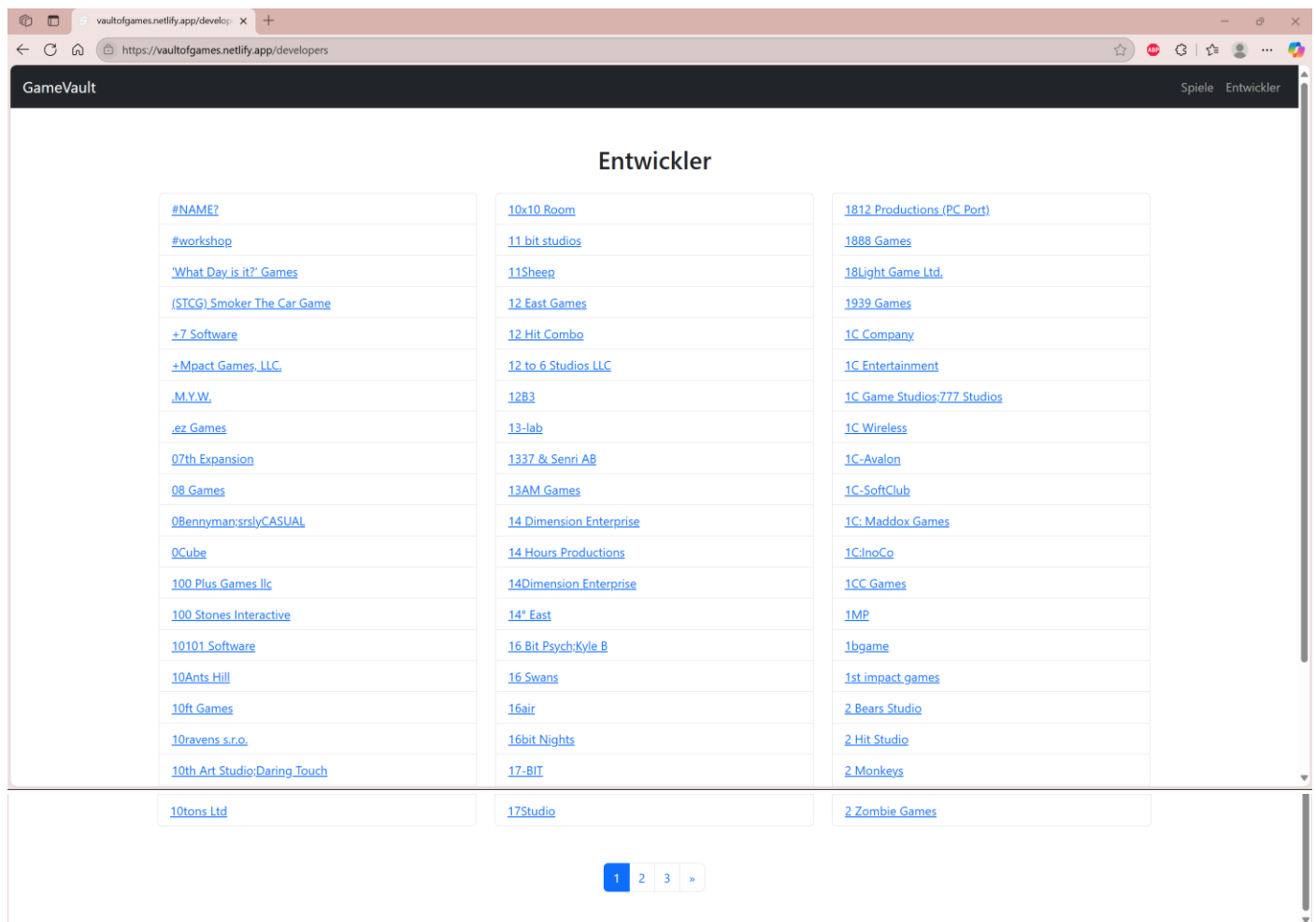
Auf dieser Seite kann ein vorhandenes Spiel anhand seiner appid bearbeitet werden. Die Eingabefelder sind vorausgefüllt mit den aktuellen Daten des Spiels. Nach dem Speichern wird das Spiel aktualisiert und die Anwendung leitet zur Detailansicht des Spiels zurück. Alle Felder ausser appid sind bearbeitbar und Name, Entwickler ID und Preis müssen einen Wert haben.

Dateien:

- Routes/games/[appid]/edit/+page.svelte
- Routes/games/[appid]/edit/+page.server.js
- Lib/server/games.js

3.6. Entwickler Übersicht

Route: /developer



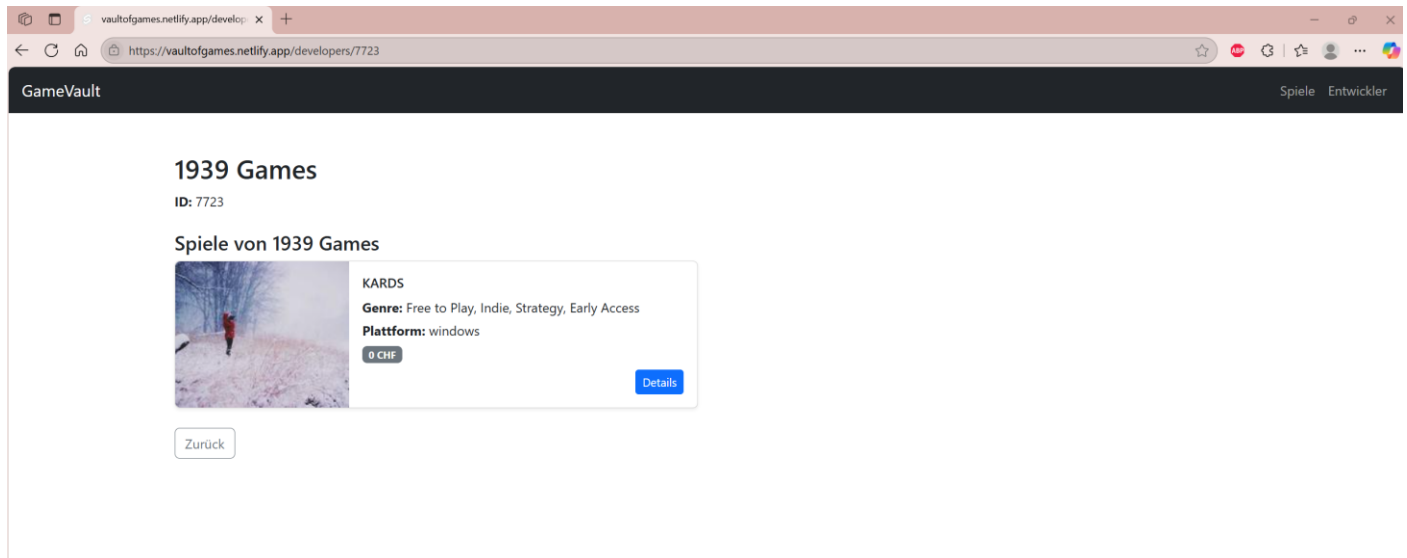
Auf dieser Seite werden alle Entwickler alphabetisch sortiert und gleichmässig in drei Spalten dargestellt. Es werden 60 Entwickler pro Seite angezeigt. Die Darstellung erfolgt als einfache Link-Liste, die einzelnen List Items werden mit dem ausgelagerten `DeveloperListItem.svelte` dargestellt. Die Pagination ist durch die ausgelagerte Komponente `Pagination.svelte` realisiert. Jeder Entwickler ist als Link dargestellt, der zur Detailseite unter `/developers/[developer_id]` führt.

Dateien:

- `Lib/server/developer.js`
- `Lib/components/Pagination.svelte`
- `Routes/developers/+page.svelte`
- `Routes/developers/+page.server.js`

3.7. Entwickler Details

Route: /developers/[developer_id]

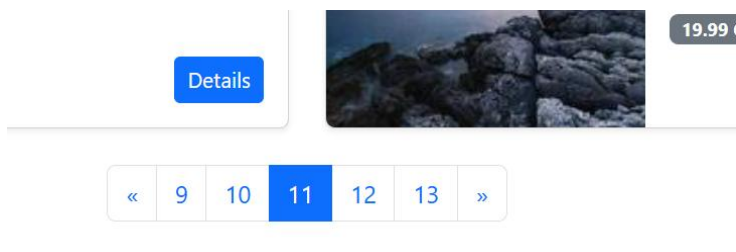


Diese Seite zeigt Detailinformationen zu einem Entwickler basierend auf dessen developer_id an. Zusätzlich werden alle zugehörigen Spiele angezeigt, also alle Games, deren developer_id mit dem aktuellen Entwickler übereinstimmt.

4. Erweiterungen

4.1. Pagination

Auf der Seite zur Spiel- und Entwicklerübersicht wurde eine eigene Pagination-Funktionalität implementiert. Pro Seite werden nur bestimmte Anzahl an Datensätzen angezeigt. Die Aktuelle Seitenzahl wird über die URL übergeben. Die Navigation erfolgt über eine wiederverwendbare Komponente Pagination.svelte, welche auf beiden Übersichtsseiten eingebunden wird. Es werden jeweils die angrenzenden Seiten sowie 10 Vor/Zurück Buttons angezeigt.



Dateien:

- Lib/components/Pagination.svelte
- Routes/games/+page.svelte
- Routes/games/+page.server.js
- Routes/developers/+page.svelte
- Routes/developers/+page.server.js
- Lib/server/games.js
- Lib/server/developers.js

4.2. Bidirektionale Verknüpfung

Um die Verbindung zwischen Spiel und Entwickler für Nutzer intuitiv nachvollziehbar zu machen, wurde eine bidirektionale Verlinkung implementiert. In der Spiel-Detailansicht wird der Name des Entwicklers angezeigt und ist verlinkt auf die entsprechende Entwickler-Detailseite. In der Entwickler-Detailansicht werden alle von diesem Entwickler erstellten Spiele angezeigt. Jedes Spiel ist dabei als GameCard (Komponente) mit Link zur jeweiligen Spiel-Detailansicht dargestellt.

Creations



Preis: 2.99 CHF

Durchschnittliche Spielzeit: 109 min

Entwickler: [Bizarre Creations](#)

Mindestalter: 0

Bizarre Creations



Geometry Wars: Retro Evolved

Genre: Casual

Plattform: windows

2.99 CHF

[Details](#)

Dateien:

- Routes/games/[appid]/+page.svelte
- Routes/games/[appid]/+page.server.js
- Routes/developers/[developer_id]/+page.svelte
- Routes/developers/[developer_id]/+page.server.js
- Lib/server/games.js
- Lib/server/developers.js

4.3. Komponenten

Um den Code wartbarer und modularer zu gestalten, wurden wiederverwendbare Svelte-Komponenten erstellt. Diese Kapselung verbessert die Struktur der Anwendung, vermeidet Duplikate und vereinfacht zukünftige Erweiterungen oder Änderungen.

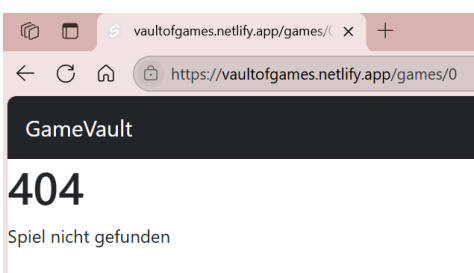
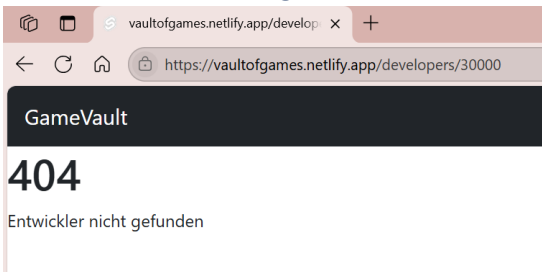
Verwendete Komponenten:

- GameCard.svelte
- Pagination.svelte
- Navbar.svelte
- DeveloperListItem.svelte

Dateien:

- lib/components/GameCard.svelte
- lib/components/Pagination.svelte
- lib/components/Navbar.svelte
- routes/+page.layout
- routes/developers/+page.svelte
- routes/games/+page.svelte

4.4. Fehlerbehandlung und Redirects



Verwendet bei:

- Detailansicht von Spielen und Entwicklern
- Ungültige Formulareinträge
- Löschen von Datensätzen
- Nach erfolgreichen Formularaktionen

Die Anwendung enthält systematische Fehlerbehandlung sowie gezielte Redirects zur Verbesserung der Nutzerführung und Robustheit.

Fehlerbehandlung

Nicht gefundene Ressourcen: Wenn ein Spiel oder Entwickler nicht in der Datenbank gefunden wird, wird ein 404-Fehler geworfen. Dies wird in der load-Funktion durch `throw error(404, '...')` ausgelöst und auf der Seite benutzerfreundlich angezeigt.

Serverseitige Fehler: Falls eine Datenbankaktion fehlschlägt (z. B. Spiel nicht gefunden beim Löschen), wird der Nutzer nicht im Unklaren gelassen, Fehlermeldungen werden geloggt und die Weiterleitung findet dennoch statt.

Redirects

Nach erfolgreichem Erstellen/Bearbeiten/Löschen: Nach POST-Aktionen wird der Benutzer mit `throw redirect(303, '/...')` zurück zur Übersicht geleitet.

Dateien:

- `/routes/games/[appid]/+page.server.js`
- `/routes/developers/[developer_id]/+page.server.js`
- `/routes/games/create/+page.server.js`
- `/routes/games/[appid]/edit/+page.server.js`
- `/routes/games/[appid]/delete/+server.js`