

Universität der Bundeswehr München
Fakultät für Betriebswirtschaft



Digitale Geschichtstour Windeck

BACHELORARBEIT

zur Erlangung des akademischen Grades Bachelor of Science

erstellt an der Professur für Informatik,
insbesondere Ingenieurinformatik



Universität der Bundeswehr München

Institut für **Mathematik**
und **Informatik**

Eingereicht von: Tobias Engelberth

Matrikel-Nr.: -----

Kontakt: -----

51570 Windeck

tobias.engelberth@gmail.com

Erstgutachter: Prof. Dr.-Ing. Reinhard Finsterwalder

Beginn: 01.04.2026

Abgabe: 30.06.2026

Inhaltsverzeichnis

1	Einleitung.....	5
1.1	Motivation	5
1.2	Problemstellung.....	6
1.3	Zielsetzung und Abgrenzung.....	9
2	Projektüberblick	11
2.1	Projektidee „Digitale Geschichtstour Windeck“.....	11
2.2	Zielgruppen und Nutzungsszenarien	12
2.2.1	Szenario 1: Spontane Nutzung durch einen Einwohner	12
2.2.2	Szenario 2: Geplante Nutzung im Rahmen einer Fahrradtour	13
2.2.3	Szenario 3: Nutzung durch einen touristischen Besucher	13
2.3	Funktionale und nicht funktionale Anforderungen.....	14
3	Systemarchitektur & Technologien	17
3.1	Systemarchitektur	17
3.2	Technologieauswahl.....	18
3.3	Entwicklungsumgebung und Tools.....	20
4	Datenmodell und Datenzugriff	21
4.1	Datenmodell	21
4.2	Datenzugriff	22
4.3	Speichern von Medien	23
5	Backend & Admin-Oberfläche.....	25
5.1	Backend-Architektur	25
5.2	API-Endpunkte	26
5.3	Anwendersicht.....	27
5.4	Authentifizierung.....	35
6	Mobile App.....	37
6.1	App-Architektur.....	37

6.2	Anwendersicht.....	38
6.3	QR-Codes & Deep Links	48
7	Bereitstellung und Veröffentlichung.....	51
7.1	Hosting	51
7.2	Veröffentlichung der App.....	52
7.3	Bereitstellung im Ort.....	53
8	Fazit & Ausblick.....	55
8.1	Zusammenfassung der Projektergebnisse	55
8.2	Grenzen und Herausforderungen.....	56
8.3	Mögliche Weiterentwicklungen	58
8.4	Übertragbarkeit auf andere Gemeinden	58
9	Literaturverzeichnis	61
10	Eigenständigkeitserklärung.....	63

1 Einleitung

Die vorliegende Arbeit dokumentiert die Konzeption, Entwicklung und Bereitstellung der Anwendung „Digitale Geschichtstour Windeck“. Im Mittelpunkt steht damit keine klassische theoretische Untersuchung, sondern die praktische Umsetzung eines Softwareprojekts im kommunalen und touristischen Kontext. Die Arbeit beschreibt, aus welcher Problemstellung das Projekt entstanden ist, welche Ziele mit der Anwendung verfolgt werden und welche technischen Entscheidungen der Realisierung zugrunde liegen.

1.1 Motivation

Als die Themenfindung für meine Bachelorarbeit konkreter wurde, war mir schnell klar, dass ich meine Arbeit im Bereich Software Engineering schreiben möchte. Einerseits konnte ich in diesem Feld bereits praktische Erfahrungen sammeln, andererseits interessiere ich mich besonders für die Entwicklung digitaler Anwendungen. Dabei war es mir jedoch von Anfang an wichtig, nicht lediglich ein technisches Projekt umzusetzen, sondern etwas zu entwickeln, das einen tatsächlichen Zweck erfüllt und anderen Menschen einen spürbaren Nutzen bringt. Ich wollte kein Projekt realisieren, das nach Abschluss der Arbeit niemals eine reale Anwendung finden wird.

Aus diesem Anspruch entstand die Idee, ein Projekt mit praktischem und zugleich lokalem Bezug zu entwickeln. Gerade Kommunen verfügen häufig über ein großes Potenzial für digitale Verbesserungen und die Erweiterung bestehender Angebote, stehen jedoch häufig vor begrenzten finanziellen, personellen oder organisatorischen Ressourcen. Gerade deshalb erschien es sinnvoll, eine digitale Lösung zu entwickeln, die an eine konkrete kommunale Aufgabe anknüpft und mit überschaubarem Aufwand praktisch nutzbar gemacht werden kann. Als gebürtiger Windecker lag es nahe, dieses Vorhaben auf die eigene Heimatgemeinde zu beziehen. Die Anwendung sollte nicht nur eine technische Aufgabenstellung erfüllen, sondern zugleich einen Beitrag dazu leisten, die historischen und kulturellen Besonderheiten Windecks zeitgemäß sichtbar zu machen.

Einen zusätzlichen Impuls erhielt diese Idee im Rahmen der Vorlesung „Digitale Transformation“. Dort behandelten wir unter anderem ein Projekt in Magdeburg, bei dem die Innenstadt durch den Einsatz digitaler Technologien neu erlebbar gemacht werden sollte. An bestimmten Orten konnten über das eigene Smartphone virtuelle Inhalte aufgerufen werden, wobei der Zugang über QR-Codes erfolgte. Das Ziel bestand darin, den öffentlichen Raum attraktiver zu gestalten und die Innenstadt auf innovative Weise zu beleben. Mich hat an diesem Konzept insbesondere die Verbindung von realen Orten, digitaler Technik und aktiver Nutzerinteraktion überzeugt.

Da ich mich sehr für Geschichte interessiere und gerne Informationstafeln an historischen Orten lese, entstand die Überlegung, wie sich dieses Prinzip auf die Vermittlung lokaler Geschichte übertragen ließe. Klassische Informationstafeln erfüllen zwar ihren Zweck, ihre Darstellung ist jedoch statisch, räumlich gebunden und inhaltlich nur begrenzt flexibel. Eine digitale Lösung bietet die Möglichkeit, historische Orte zunächst gesammelt sichtbar zu machen und die zugehörigen Informationen anschließend per Scan einfach und direkt zugänglich zu machen. Dadurch können Inhalte nicht nur moderner, sondern auch nutzerfreundlicher und ansprechender vermittelt werden.

Aus diesen Überlegungen entwickelte sich die Idee einer digitalen Geschichtstour für Windeck. Historische Orte sollen durch digitale Inhalte ergänzt und in einer Anwendung gesammelt dargestellt werden. Dadurch entsteht ein niedrigschwelliger Zugang zu lokalgeschichtlichen Informationen, der sowohl für Einwohner als auch für Besucher nutzbar ist. Die Arbeit verbindet damit ein technisches Entwicklungsprojekt mit einem regionalen Anwendungsfall und verfolgt das Ziel, Geschichte nicht nur zu dokumentieren, sondern im öffentlichen Raum zeitgemäß erfahrbar zu machen.

1.2 Problemstellung

Die Gemeinde Windeck verfügt über zahlreiche historisch, kulturell und landschaftlich interessante Orte. Gerade im Zusammenspiel mit Wanderwegen, Fahrradrouen und touristischen Angeboten entsteht daraus ein erhebliches Potenzial für die Vermittlung lokaler Geschichte. Dieses Potenzial hängt jedoch wesentlich davon ab, wie gut die zugehörigen Informationen zugänglich,

verständlich und aktuell bereitgestellt werden können. Historische Orte entfalten ihren Mehrwert nicht allein durch ihre physische Existenz, sondern erst durch die Einordnung ihrer Bedeutung, ihrer Entstehung und ihres Zusammenhangs mit der regionalen Geschichte.

Die bisher naheliegendste Form der Informationsvermittlung im öffentlichen Raum sind analoge Informationstafeln. Sie bieten Besuchern direkt am jeweiligen Ort grundlegende Informationen und benötigen keine technischen Endgeräte. Dennoch sind sie mit mehreren strukturellen Einschränkungen verbunden. Eine Tafel ist nach ihrer Herstellung weitgehend statisch. Müssen Inhalte korrigiert, erweitert oder aktualisiert werden, ist dies meist nur mit zusätzlichem organisatorischem, gestalterischem und finanziellem Aufwand möglich. Dadurch entsteht eine geringe Flexibilität, insbesondere wenn neue historische Erkenntnisse, zusätzliche Bilder, weitere Quellen oder neue Stationen in ein bestehendes Angebot integriert werden sollen.

Hinzu kommt die räumliche Begrenzung analoger Tafeln. Die verfügbare Fläche zwingt zu einer starken Auswahl und Verdichtung der Inhalte. Umfangreichere historische Zusammenhänge, mehrere Bildebenen oder weiterführende Quellen lassen sich nur begrenzt darstellen. Daraus ergibt sich ein grundlegender Zielkonflikt: Eine Tafel mit vielen Informationen wird schnell unübersichtlich und baulich größer, während eine kleinere Tafel zwangsläufig nur einen Ausschnitt der vorhandenen Informationen vermitteln kann. Eine mehrstufige Informationsstruktur, bei der zunächst zentrale Inhalte angezeigt und bei Interesse vertiefende Informationen abgerufen werden können, ist in analoger Form kaum umsetzbar.

Ein weiteres Problem betrifft die Mehrsprachigkeit und Barrierefreiheit. Werden Inhalte auf einer Tafel in mehreren Sprachen abgedruckt, vergrößert sich der benötigte Platz erheblich. Gleichzeitig bleibt die Auswahl der Sprachen begrenzt. Auch barrierearme Formen der Informationsvermittlung, etwa vergrößerbare Schrift, alternative Darstellungen oder perspektivisch auditive Inhalte, sind auf klassischen Tafeln nur eingeschränkt realisierbar. Damit stoßen analoge Lösungen insbesondere dann an Grenzen, wenn sie unterschiedliche Nutzergruppen erreichen sollen, etwa Einheimische, Touristen, Schulklassen, ältere Menschen oder Personen mit eingeschränkter Mobilität.

Auch multimediale Inhalte können auf analogen Tafeln nur sehr eingeschränkt eingebunden werden. Bilder lassen sich zwar abdrucken, Videos, Audioinhalte, interaktive Karten oder Verlinkungen zu weiterführenden Quellen sind jedoch nicht unmittelbar integrierbar. Gerade für eine historische Vermittlung ist dies eine relevante Einschränkung, da Geschichte häufig durch ergänzende Medien anschaulicher, emotionaler und kontextreicher vermittelt werden kann. Digitale Inhalte ermöglichen hier eine deutlich flexiblere Darstellung, etwa durch Bildergalerien, externe Quellen, thematische Verknüpfungen oder weiterführende Informationen zu verwandten Orten.

Darüber hinaus sind analoge Tafeln dauerhaft äußeren Einflüssen ausgesetzt. Witterung, Verschleiß oder Vandalismus können dazu führen, dass Informationen schwer lesbar oder im schlimmsten Fall nicht mehr nutzbar sind. Zwar betrifft dieses Risiko grundsätzlich auch gedruckte QR-Schilder, jedoch ist der eigentliche Informationsbestand bei einer digitalen Lösung nicht auf dem Schild selbst gespeichert. Wird ein Schild beschädigt, bleiben die Inhalte weiterhin digital vorhanden und können nach Austausch des Schildes unverändert abgerufen werden. Der physische Träger dient damit nicht mehr als vollständiges Informationsmedium, sondern lediglich als Zugangspunkt zu zentral gepflegten Inhalten.

Die Problemstellung besteht somit nicht darin, analoge Informationstafeln grundsätzlich zu ersetzen, sondern ihre strukturellen Grenzen durch eine digitale Ergänzung zu überwinden. Benötigt wird eine Lösung, die Informationen zentral verwaltbar, flexibel erweiterbar und direkt vor Ort abrufbar macht. Gleichzeitig soll sie niedrigschwellig nutzbar sein, ohne dass Nutzer umfangreiche technische Vorkenntnisse benötigen. Eine digitale Anwendung, die historische Stationen, Medien und Touren miteinander verbindet und über QR-Codes zugänglich macht, bietet hierfür einen geeigneten Ansatz. Sie kann Inhalte aktueller, vernetzter und zielgruppengerechter bereitstellen und zugleich die Grundlage für eine langfristig erweiterbare Geschichtsvermittlung in Windeck schaffen.

1.3 Zielsetzung und Abgrenzung

Ziel der Arbeit ist die technische Konzeption und Umsetzung einer digitalen Anwendung zur Vermittlung lokalthistorischer Inhalte in der Gemeinde Windeck.

Die Arbeit ist im Bereich Software Engineering verortet. Betrachtet werden insbesondere Systemarchitektur, Technologieauswahl, Datenmodellierung, Implementierung und Bereitstellung. Die inhaltliche Erstellung und fachliche Prüfung der historischen Texte ist hingegen nicht Bestandteil der Arbeit. Diese erfolgt durch den Tourismus Windecker Ländchen e. V. sowie weiterer Vereine und Personen aus der Region.

Das System besteht aus einer mobilen App für Anwender, einer webbasierten Administrationsoberfläche, einer Datenbank sowie einer API zur Bereitstellung der Daten. Über die App können Stationen, Touren und Medien abgerufen werden. Der Zugriff auf einzelne Stationen soll zusätzlich über QR-Codes möglich sein, die am jeweiligen Ort angebracht werden können. Die Administrationsoberfläche ermöglicht es, Inhalte ohne direkten Eingriff in Datenbank oder Quellcode zu pflegen.

Nicht umgesetzt werden in der vorliegenden Version eine komplexe Benutzer- und Rollenverwaltung, Mehrsprachigkeit sowie die Unterstützung von Audio und Videoinhalten. Die Anwendung beschränkt sich zunächst auf deutschsprachige Inhalte und Bilder. Ebenfalls nicht Teil der Arbeit ist die physische Installation der QR-Schilder im öffentlichen Raum.

2 Projektüberblick

Das folgende Kapitel gibt einen Überblick über die digitale Geschichtstour Windeck. Es beschreibt zunächst die Grundidee des Projekts und betrachtet anschließend Zielgruppen, Nutzungsszenarien sowie die daraus abgeleiteten Anforderungen.

2.1 Projektidee „Digitale Geschichtstour Windeck“

Die Grundidee der digitalen Geschichtstour besteht darin, historische Orte in der Gemeinde Windeck digital zugänglich zu machen und sie in einer mobilen Anwendung gesammelt darzustellen. An ausgewählten Stationen werden QR-Codes angebracht, über die Anwender direkt zu den passenden Informationen gelangen können. Dadurch wird der reale Ort mit einem digitalen Informationsangebot verbunden. Der historische Ort bleibt Ausgangspunkt der Nutzung, wird jedoch durch Texte, Bilder und weitere strukturierte Inhalte ergänzt.

Eine Station beschreibt jeweils einen konkreten Ort innerhalb der Gemeinde. Dazu gehören insbesondere Titel, Beschreibung, geografische Koordinaten sowie zugehörige Medien, etwa Bilder. Zusätzlich können Stationen thematisch eingeordnet und bei Bedarf einer oder mehreren Touren zugeordnet werden.

Touren bilden mehrere Stationen in einer sinnvollen Reihenfolge ab. Sie können sich beispielsweise an einem geografischen Verlauf, einem historischen Thema oder einer touristischen Route orientieren. Dadurch erhalten Anwender die Möglichkeit, nicht nur einzelne Orte aufzurufen, sondern mehrere Stationen im Zusammenhang zu erleben. Die Anwendung unterstützt damit sowohl spontane Nutzungssituationen, etwa das Scannen eines QR-Codes während eines Spaziergangs, als auch geplante Rundgänge durch Windeck.

Die digitale Geschichtstour versteht sich damit als Ergänzung zu bestehenden touristischen und lokalhistorischen Angeboten. Sie ersetzt nicht den historischen Ort selbst und auch nicht die Arbeit der beteiligten Vereine oder Institutionen. Vielmehr stellt sie eine technische Plattform bereit, über die vorhandene und zukünftige Inhalte zentral verwaltet, flexibel erweitert und vor Ort niedrigschwellig abgerufen werden können. Damit schafft das Projekt eine

Grundlage, um lokale Geschichte zeitgemäß, mobil und besser vernetzt zugänglich zu machen.

2.2 Zielgruppen und Nutzungsszenarien

Die digitale Geschichtstour richtet sich grundsätzlich an alle Personen, die sich für historische Orte, regionale Besonderheiten oder touristische Angebote in Windeck interessieren. Die Zielgruppe entspricht damit weitgehend derjenigen klassischer Informationstafeln, wird jedoch durch die digitale Umsetzung erweitert. Neben Besuchern vor Ort können Inhalte auch unabhängig vom konkreten Standort in der App aufgerufen werden. Dadurch eignet sich die Anwendung nicht nur für spontane Nutzungssituationen während eines Spaziergangs, sondern auch für die Vorbereitung von Ausflügen, schulische Projekte oder die nachträgliche Beschäftigung mit besuchten Orten.

Zur Verdeutlichung werden im Folgenden drei beispielhafte Nutzungsszenarien dargestellt. Sie zeigen, wie unterschiedliche Anwender mit der digitalen Geschichtstour in Berührung kommen und welchen praktischen Mehrwert die Anwendung jeweils bietet.

2.2.1 Szenario 1: Spontane Nutzung durch einen Einwohner

Tobi ist 20 Jahre alt, in Windeck aufgewachsen und kennt viele Orte der Gemeinde aus seinem Alltag. Bei einem Spaziergang kommt er an einem Denkmal vorbei, das er zwar schon häufig gesehen hat, dessen historische Bedeutung ihm jedoch nicht bekannt ist. Neben dem Denkmal entdeckt er ein neues Schild mit einem QR-Code. Aus Neugier scannt er den Code mit seinem Smartphone und gelangt auf die Projektseite der digitalen Geschichtstour.

Dort erhält er eine kurze Erklärung zum Projekt und kann die App herunterladen. Nach dem erneuten Scannen des QR-Codes öffnet sich direkt die passende Station innerhalb der Anwendung. Tobi sieht Bilder, liest Hintergrundinformationen zum Denkmal und findet zusätzlich einen weiterführenden Link zu einem ansässigen Heimatverein. Aus einem zunächst beiläufig wahrgenommenen Ort wird dadurch ein historisch eingeordneter Bestandteil seiner Heimatgemeinde.

2.2.2 Szenario 2: Geplante Nutzung im Rahmen einer Fahrradtour

Marion ist 50 Jahre alt und lebt seit vielen Jahren in Windeck. Seitdem sie ein E-Bike besitzt, unternimmt sie regelmäßig Fahrradtouren durch die Region. Über eine Bekannte erfährt sie von der digitalen Geschichtstour und installiert die App mit Unterstützung ihrer Tochter. In der App entdeckt sie verschiedene Touren und wählt eine Route aus, die mehrere historische Stationen miteinander verbindet. Diese gefällt ihr so gut, dass sie diese gleich am nächsten Wochenende mit ihrer Freundin unternimmt.

2.2.3 Szenario 3: Nutzung durch einen touristischen Besucher

Markus ist 35 Jahre alt und besucht Windeck mit seiner Familie für ein verlängertes Wochenende. Da er die Region noch nicht gut kennt, sucht er nach Möglichkeiten für Ausflüge und stößt online auf die digitale Geschichtstour. Er lädt die App herunter und verschafft sich über die Kartenansicht einen Überblick über historische Orte in der Nähe.

Dabei entdeckt er nicht nur bekannte Sehenswürdigkeiten wie die Burg Windeck, sondern auch weniger offensichtliche Orte, die in klassischen touristischen Informationen möglicherweise nicht unmittelbar auffallen. Durch die App kann er Stationen gezielt auswählen, Informationen abrufen und einzelne Orte in seine Tagesplanung einbinden. Die Anwendung wird damit zu einem Werkzeug, um eine unbekannte Region eigenständig zu erkunden.

Die dargestellten Szenarien und Zielgruppen zeigen, dass die digitale Geschichtstour nicht auf ein einzelnes Nutzungsmuster beschränkt ist. Sie unterstützt spontane Informationsabrufe ebenso wie geplante Touren und ortsunabhängige Nutzung. Für Einwohner kann sie bekannte Orte neu erschließen, für Besucher bietet sie Orientierung.

Tabelle 3.2.1 fasst die relevanten Zielgruppen mit deren möglichen Nutzungsszenarien und Mehrwerten nochmals zusammen:

[Siehe nächste Seite]

Zielgruppe	Nutzungsszenario	Mehrwert der Anwendung
Einwohner	Spontanes Scannen eines QR-Codes an einem bekannten Ort	Historische Einordnung vertrauter Orte, neue Wahrnehmung des eigenen Wohnumfelds
Touristen und Tagesbesucher	Planung eines Ausflugs oder Abruf von Informationen vor Ort	Orientierung, gebündelte Informationen und Zugang zu weniger bekannten Orten
Fahrradfahrer und Wanderer	Nutzung einer Tour während einer Route durch Windeck	Verbindung von Bewegung, Navigation und lokalgeschichtlicher Information
Schulklassen	Nutzung einer vorbereiteten Tour im Unterricht oder bei einem Projekttag	Strukturierte Vermittlung lokaler Geschichte am konkreten Ort
Vereine und Projektgruppen	Sammlung, Pflege und Bereitstellung historischer Inhalte	Zentrale Plattform zur dauerhaften Sichtbarmachung lokalhistorischer Arbeit
Personen mit eingeschränkter Mobilität	Abruf von Stationen unabhängig vom tatsächlichen Besuch vor Ort	Zugang zu Informationen auch ohne physische Anwesenheit am jeweiligen Ort

Tabelle 3.2.1 – Zielgruppen mit deren Nutzungsszenarien und Mehrwerten

2.3 Funktionale und nicht funktionale Anforderungen

Aus der Projektidee und den zuvor beschriebenen Nutzungsszenarien ergeben sich die Anforderungen an das System. Dabei lassen sich funktionale Anforderungen, also konkret bereitzustellende Funktionen, von nicht funktionalen Anforderungen unterscheiden. Letztere beschreiben vor allem qualitative Eigenschaften der Anwendung, etwa Bedienbarkeit, Wartbarkeit und Erweiterbarkeit.

Die funktionalen Anforderungen enthalten zwei zentrale Aspekte. Einerseits muss die mobile App den Anwendern einen einfachen Zugriff auf Stationen, Touren und Medien ermöglichen. Andererseits muss die Administrationsoberfläche eine Pflege der Inhalte erlauben, ohne dass hierfür technisches Wissen über Datenbanken oder Quellcode erforderlich ist.

Tabelle 3.3.1 fasst die daraus abgeleiteten Anforderungen kompakt zusammen.

[Siehe nächste Seite]

Bereich	Anforderung	Zweck
Mobile App	Anzeige von Stationen	Anwender können historische Orte in Windeck durchsuchen und auswählen
Mobile App	Detailansicht einzelner Stationen	Texte, Bilder und weitere Informationen werden ortsbezogen bereitgestellt
Mobile App	Anzeige von Touren	Mehrere Stationen können thematisch oder räumlich verbunden werden
Mobile App	QR-Code Verarbeitung	Ein gescannter Code führt direkt zur passenden Station
Mobile App	Kartenansicht	Stationen können räumlich eingeordnet und leichter gefunden werden
Mobile App	Navigation zu Stationen	Anwender können sich über eine externe Navigationsapp zum Ort führen lassen
Admin-Oberfläche	Verwaltung von Stationen	Stationen können angelegt, bearbeitet und gelöscht werden
Admin-Oberfläche	Verwaltung von Touren	Stationen können zu geordneten Rundgängen zusammengestellt werden
Admin-Oberfläche	Verwaltung von Kategorien	Inhalte können thematisch strukturiert werden
Admin-Oberfläche	Verwaltung von Medien	Bilder können einzelnen Stationen zugeordnet und aktualisiert werden
Backend und Datenbank	Zentrale Datenbereitstellung	App und Admin-Oberfläche greifen auf konsistente Daten zu
Gesamtsystem	Einfache Bedienbarkeit	Die Anwendung bleibt für unterschiedliche Anwendergruppen niedrigschwellig nutzbar
Gesamtsystem	Wartbare Architektur	Inhalte und Funktionen können langfristig gepflegt werden
Gesamtsystem	Erweiterbarkeit	Spätere Funktionen wie Mehrsprachigkeit, Audio oder Video können ergänzt werden
Gesamtsystem	Kostengünstiger Betrieb	Der dauerhafte Einsatz bleibt auch mit begrenzten Ressourcen realistisch

Tabelle 3.3.1 - Funktionale und nicht funktionale Anforderungen

3 Systemarchitektur & Technologien

In diesem Kapitel wird die technische Grundlage des Projekts beschrieben. Das System besteht aus mehreren Komponenten, die miteinander zusammenspielen, um die Bereitstellung, Verwaltung und Nutzung der Inhalte zu ermöglichen.

3.1 Systemarchitektur

Die folgende Grafik zeigt die Systemarchitektur:

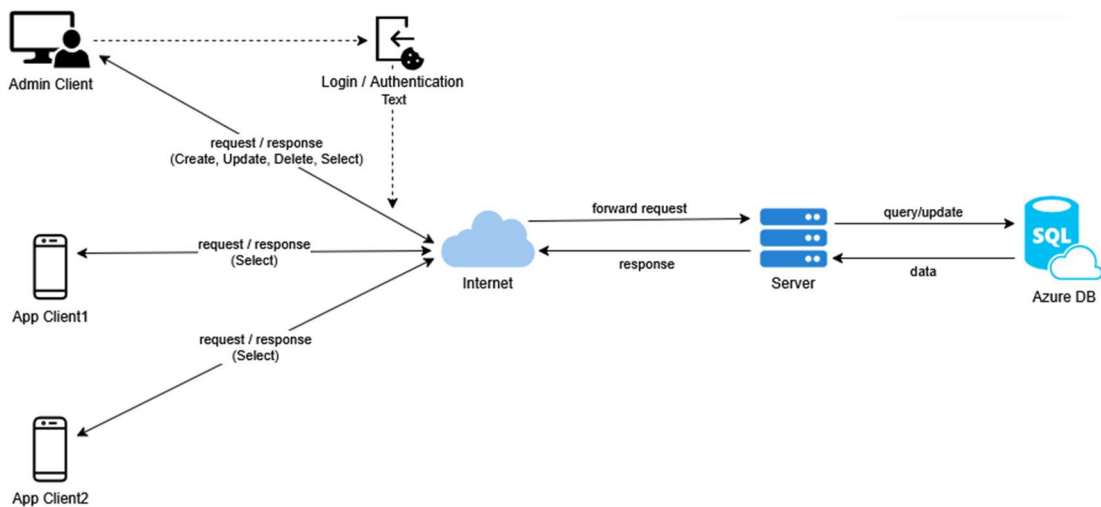


Abbildung 4.1.1 - Systemarchitektur

Sie ist so aufgebaut, dass die verschiedenen Aufgabenbereiche der Anwendung klar voneinander getrennt sind. Das System besteht grundsätzlich aus zwei Clienttypen, einem zentralen Server und einer angebundenen Datenbank. Diese Struktur ermöglicht eine zentrale Verwaltung von Inhalten und deren Bereitstellung für unterschiedliche Nutzergruppen.

Wie in *Abbildung 4.1.1* dargestellt, greifen sowohl die mobile App als auch der administrative *Web-Client* über das Internet auf den *Web-Server* zu. Die mobile App dient dem Abruf der bereitgestellten Inhalte durch den Nutzer. Der *Admin-Client* wird dagegen für die Pflege der Daten verwendet. Über ihn können Stationen, Touren, Kategorien und Medieninhalte angelegt, geändert oder gelöscht werden. Während die mobile Anwendung im Wesentlichen lesend auf Daten zugreift, umfasst der administrative Bereich zusätzlich schreibende Operationen (*CRUD*).

Er nimmt über eine *API*-Schnittstelle Anfragen der *Clients* entgegen, verarbeitet sie und leitet sie an die Datenbank weiter. Ebenso gibt er die von der Datenbank gelieferten Daten an die jeweiligen *Clients* zurück. Durch diesen Aufbau wird verhindert, dass *Clients* direkt auf die Datenbank zugreifen. Das erhöht die Sicherheit und vereinfacht die Kontrolle über Datenzugriffe und Anwendungslogik.

Die dauerhafte Speicherung der Inhalte erfolgt in einer *Azure-SQL-Datenbank (Serverless)*. Dort werden die Inhalte, etwa Informationen zu Stationen, Touren, Kategorien und Medienobjekten, zentral abgelegt. Die zentrale Datenhaltung stellt sicher, dass Änderungen, die im Admin-Bereich vorgenommen werden, unmittelbar für alle Nutzer der App verfügbar sind.

Der wesentliche Vorteil dieser Architektur liegt in der klaren Trennung der Verantwortlichkeiten. Die *Clients* sind für die Darstellung und Interaktion zuständig, das Backend für die Verarbeitung und Bereitstellung der Daten und die Datenbank für die persistente Speicherung. Dadurch entsteht ein modularer Aufbau, der die Wartbarkeit verbessert und spätere Erweiterungen erleichtert. Gleichzeitig eignet sich diese Struktur gut für eine praxisnahe Anwendung, da Inhalte zentral gepflegt und so auf mehreren Endgeräten konsistent bereitgestellt werden können.

3.2 Technologieauswahl

Für die Umsetzung ist ein Technologie-Stack ausgewählt worden, der auf eine möglichst einheitliche und wartbare Entwicklung ausgelegt ist. Da das Projekt eine mobile Anwendung sowie ein Backend mit Administrationsoberfläche und Datenbankbindung umfasst, ist es sinnvoll, Technologien zu verwenden, die sich gut kombinieren lassen und in einem gemeinsamen Ökosystem arbeiten. Bei der Auswahl ist insbesondere auf die Anforderungen hinsichtlich Entwicklungsaufwand, Wartbarkeit, Erweiterbarkeit und Bereitstellung geachtet worden.

Für die Entwicklung der mobilen Anwendung kommt **.NET MAUI** zum Einsatz.^[1] Dabei handelt es sich um ein Framework zur plattformübergreifenden Entwicklung von Anwendungen innerhalb des *.NET*-Ökosystems. Ein wesentlicher Vorteil besteht darin, dass zentrale Teile der Logik und Struktur in einer

gemeinsamen Codebasis umgesetzt werden können. Dadurch wird der Entwicklungsaufwand gegenüber getrennten nativen Anwendungen reduziert. Dies ist für das vorliegende Projekt besonders geeignet, da eine mobile App mit überschaubarem Aufwand entwickelt werden soll. Hinzu kommt, dass mit *.NET MAUI* dieselbe Programmiersprache wie im Backend verwendet werden kann. Dadurch lässt sich das Gesamtsystem konsistent umsetzen.

Die Administrationsoberfläche und die Projekt-Landingpage wird mit **ASP.NET Razor** umgesetzt.^[2] Da keine komplexe Webanwendung für die Administration erforderlich ist, ist der Einsatz eines umfangreichen Frontend-Frameworks nicht notwendig. *Razor Pages* ermöglicht die vergleichsweise einfache Erstellung klassischer Verwaltungsseiten. Dieser Ansatz ist für Anwendungsfälle wie das Anlegen, Bearbeiten und Löschen von Stationen, Touren und Medien gut geeignet.

Das Backend wird mit **ASP.NET Core Web API** umgesetzt.^[3] Diese Technologie eignet sich insbesondere für die Bereitstellung klar strukturierter *HTTP-Schnittstellen*, über die die mobile App auf Stationen, Touren und Medien zugreifen kann. Gleichzeitig bietet *ASP.NET Core* eine moderne und leistungsstarke Grundlage für Webanwendungen und lässt sich nahtlos mit weiteren Komponenten des *.NET*-Ökosystems verbinden.

Für den Datenzugriff wird **Entity Framework Core (EF-Core)** verwendet.^[4] Die Entscheidung hierfür basiert auf der engen Integration in *ASP.NET Core* sowie dem *Code-First-Ansatz*, der eine Entwicklung des Datenmodells direkt aus den *C#-Klassen* heraus ermöglicht. So können Änderungen am Modell über Migrationen in die Datenbank überführt werden. *EF-Core* bietet für ein Projekt mit überschaubarer, aber dennoch relationaler Datenhaltung einen guten Kompromiss zwischen Entwicklungsgeschwindigkeit, Lesbarkeit und Wartbarkeit.

Für das Hosting und die relationale Datenspeicherung wird **Azure App Service** und **Azure SQL** eingesetzt.^{[5] [6]} Beide Dienste sind mit dem *.NET*-Umfeld interoperabel, und ihre Bereitstellung und Veröffentlichung erfolgt unkompliziert und innerhalb kürzester Zeit.

3.3 Entwicklungsumgebung und Tools

Für die Umsetzung des Projekts wurden verschiedene Werkzeuge eingesetzt, die den Entwicklungsprozess von der Implementierung bis zur Bereitstellung unterstützt haben. Die *Tabelle 4.3.1* gibt einen Überblick über die verwendeten Entwicklungswerkzeuge.

Werkzeug (Version)	Einsatz im Projekt
<i>Visual Studio 2026 (18.5.3)</i> ^[7]	Zentrale Entwicklungsumgebung für <i>Backend</i> , <i>Administrationsoberfläche</i> und <i>mobile App</i>
<i>VS-Code (1.120)</i> ^[8]	<i>Open Source Codeeditor</i> für kleinere Aufgaben mit weniger <i>Overhead</i>
<i>.NET SDK (10.0.204)</i> ^[9]	Grundlage für <i>Build-Prozesse</i> und Ausführung der <i>.NET-Anwendungen</i>
<i>Swagger (10.1.5)</i> ^[10]	Testen und Validieren der <i>API-Endpunkte</i> während der Entwicklung
<i>Git (2.52.0.windows.1)</i> ^[11]	Lokale Versionsverwaltung und Nachverfolgung von Änderungen
<i>GitHub</i> ^[12]	Zentrale Ablage des Quellcodes und Sicherung von Entwicklungsständen
<i>Azure Portal</i> ^[13]	Verwaltung der bereitgestellten <i>Cloud-Ressourcen</i> und Konfigurationen
<i>App Store Connect</i> ^[14]	Verwaltung und Veröffentlichung der Applikation im Apple App Store
<i>Google Play Console</i> ^[15]	Verwaltung und Veröffentlichung der Applikation im Google Play Store
<i>Google Search Console</i> ^[16]	Verwaltung, Analyse und Indexierung der Projektwebsite
<i>Codex (5.2)</i> ^[17]	<i>AI-Agent</i> zur Analyse von Code sowie Unterstützung bei technischen Fragestellungen

Tabelle 4.3.1 - Entwicklungswerkzeuge

4 Datenmodell und Datenzugriff

Dieses Kapitel beschreibt, wie die im System verwendeten Daten strukturiert und gespeichert werden, um Redundanzen und Inkonsistenzen zu vermeiden.

4.1 Datenmodell

Die *Abbildung 5.1.1* zeigt das mit *EF-Core* umgesetzte Klassendiagramm.

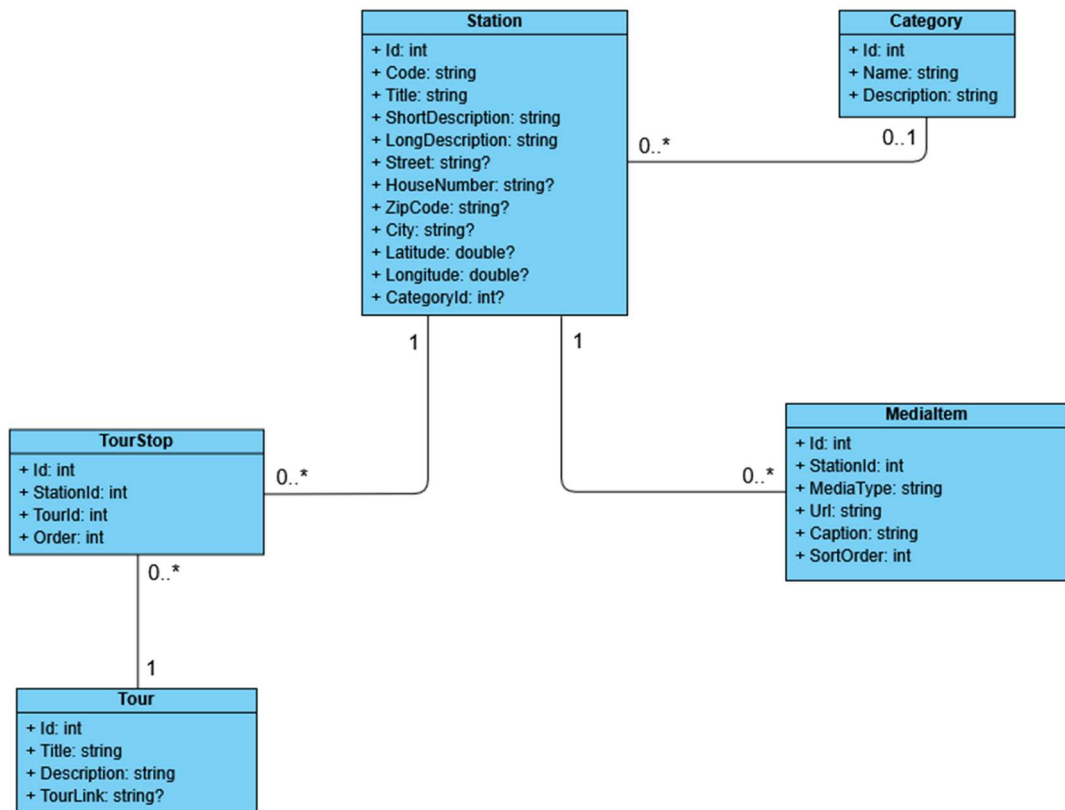


Abbildung 5.1.1 - Klassendiagramm

Die in der *Abbildung 5.1.1* dargestellten Klassen werden in der *Tabelle 5.1.1* kurz in ihrer Funktion erläutert. Eine detaillierte Beschreibung der einzelnen Attribute erfolgt an dieser Stelle nicht, da diese bereits im kommentierten Quellcode erläutert wurden.

Klasse	Beschreibung
<i>Station</i>	Repräsentiert einen historischen Ort beziehungsweise eine einzelne Station innerhalb der Anwendung. Sie enthält die zentralen inhaltlichen Informationen sowie die geografischen Daten des jeweiligen Ortes.
<i>Tour</i>	Beschreibt eine thematisch oder räumlich zusammenhängende Folge mehrerer Stationen. Sie dient dazu, einzelne Orte in einem gemeinsamen Kontext darzustellen.
<i>TourStop</i>	Stellt die Verknüpfung zwischen einer Tour und einer Station dar. Zusätzlich wird hier die Reihenfolge der Station innerhalb einer Tour gespeichert.
<i>Category</i>	Dient der thematischen Einordnung von Stationen. Dadurch können Inhalte strukturiert gruppiert und in der Anwendung besser organisiert werden.
<i>MediaItem</i>	Repräsentiert Medieninhalte, die einer Station zugeordnet sind. In der vorliegenden Umsetzung handelt es sich dabei insbesondere um Bilder.

Tabelle 5.1.1 - Beschreibung der Klassen

4.2 Datenzugriff

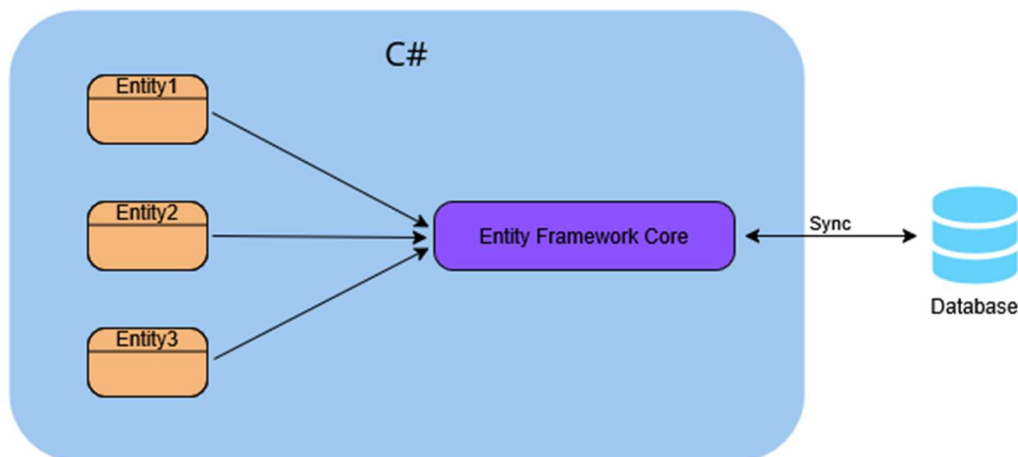


Abbildung 5.2.1 – Entity Framework Core

Für den Datenzugriff wurde *Entity Framework Core (EF-Core)* verwendet. Dabei handelt es sich um einen objektrelationalen Mapper für *.NET*, mit dem Datenbankzugriffe über *.NET*-Objekte und Kontextklassen umgesetzt werden können. Der Zugriff auf die Datenbank erfolgt nicht über manuell formulierte *SQL*-Abfragen, sondern über das in *C#* definierte Datenmodell. Änderungen am Datenmodell werden ebenfalls nicht direkt an der Datenbank selber vorge-

nommen. Die *C#-Klassen* werden entsprechend angepasst und die Änderungen werden über die Migrationsfunktion von *EF-Core* übernommen. So bleibt das Datenbankschema mit dem aktuellen Stand des Modells synchron.

Der Datenzugriff erfolgt über den *DbContext*.^[18] Er repräsentiert die Sitzung mit der Datenbank und verwaltet Abfragen, Änderungen an Entitäten sowie das Speichern dieser Änderungen. Die einzelnen Entitätsmengen des Modells werden über *DbSet*-Eigenschaften abgebildet. Änderungen am Datenbankmodell können über Migrationen in das Datenbankschema übertragen werden.^[19]

4.3 Speichern von Medien

Medien, wie Bilder, werden nicht direkt in der Datenbank gespeichert. Stattdessen werden sie auf dem *Webserver* abgelegt und sind über eine URL erreichbar. In der Datenbank selbst wird lediglich ein Objekt der Klasse *MediaItem* gespeichert, das Metadaten, wie etwa eine Bildbeschreibung sowie die URL zum jeweiligen Medium enthält. Der Vorteil dieses Vorgehens besteht darin, dass die Datenbank schlank bleibt und Medien direkt vom *Client* geladen werden können.

5 Backend & Admin-Oberfläche

Dieses Kapitel beschreibt die serverseitige Umsetzung der Geschichtstour. Das Backend übernimmt dabei zwei zentrale Aufgaben. Einerseits stellt es die Daten für die mobile App über eine *API* bereit, andererseits enthält es die Administrationsoberfläche zur Pflege der Inhalte. Damit bildet es die Verbindung zwischen mobiler Anwendung, Datenbank und Verwaltungssystem. Während die App nur lesend auf Stationen, Touren und Medien zugreift, ermöglicht der Admin-Bereich das Anlegen, Bearbeiten und Löschen der Inhalte. Der vollständige Quellcode ist unter <https://github.com/tengelberth/Windeck.Geschichtstour> einsehbar.

5.1 Backend-Architektur

Die Backend-Architektur ist so aufgebaut, dass die unterschiedlichen Aufgabenbereiche klar voneinander getrennt sind. Die mobile App greift über definierte *API-Endpunkte* auf die bereitgestellten Inhalte zu. Der administrative Bereich wird dagegen über *Razor Pages* umgesetzt und dient der direkten Pflege der Daten im Browser. Beide Bereiche verwenden dieselbe Datenbasis und greifen über *EF-Core* auf die *Azure SQL-Datenbank* zu.

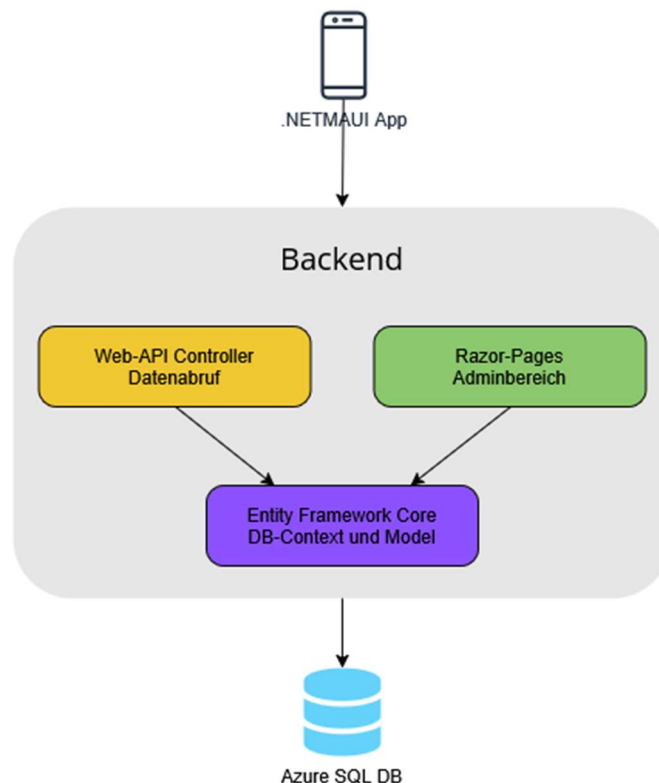


Abbildung 6.1.1 – Backend-Architektur

Abbildung 6.1.1 zeigt diese Struktur in vereinfachter Form. Die *Web-API-Controller* bilden die Schnittstelle zur mobilen App und liefern die benötigten Daten in strukturierter Form zurück. Die *Razor Pages* stellen die Verwaltungsoberfläche bereit und ermöglichen *CRUD-Operationen* für Stationen, Touren, Kategorien und Medien. Der Zugriff auf die Datenbank erfolgt nicht direkt aus den *Clients* heraus, sondern zentral über das *Backend*. Dadurch bleiben Datenzugriff, Validierung und Anwendungslogik auf Serverseite gebündelt.

5.2 API-Endpunkte

Die wichtigsten *API-Endpunkte* sind in *Tabelle 6.1.1.1* dargestellt. Sie dienen vor allem dem Abruf der Inhalte durch die mobile App. Die *Tabelle 6.1.1.1* zeigt nur eine kompakte Übersicht. Die vollständige technische Dokumentation ist über *Swagger* unter <https://geschichte-tour-backend.azurewebsites.net/swagger/index.html> abrufbar.

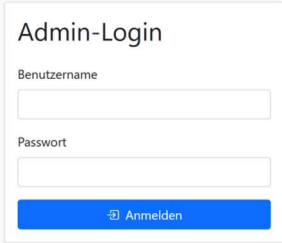
Endpunkt	Zweck
/api/stations	Abruf aller Stationen
/api/stations/{id}	Abruf einer einzelnen Station anhand ihrer ID
/api/stations/byCode/{code}	Abruf einer Station anhand eines QR-beziehungsweise Stationscodes oder der ID
/api/tours	Abruf aller Touren
/api/tours/{id}	Abruf einer einzelnen Tour mit zugehörigen Stationen
/api/categories	Abruf der vorhandenen Kategorien
/api/mediaitems	Abruf aller Medien
/api/mediaitems/{id}	Abruf eines einzelnen Mediums

Tabelle 6.1.1.1 – Übersicht API-Endpunkte

5.3 Anwendersicht

Die browserbasierte Admin-Oberfläche dient der Pflege der Inhalte der digitalen Geschichtstour. Sie richtet sich an die Personen, die Stationen, Touren und Medien verwalten, nicht an die späteren Anwender der mobilen App. Der Schwerpunkt liegt daher auf einer funktionalen und übersichtlichen Bedienung. Inhalte sollen ohne direkten Zugriff auf Datenbank oder Quellcode angelegt, verändert und gelöscht werden können.

Die folgenden Abbildungen zeigen die zentralen Bereiche der Admin-Oberfläche. Die Darstellung folgt dabei dem typischen Ablauf der Inhaltsverwaltung, von der Anmeldung über die Übersicht der vorhandenen Daten bis hin zur Bearbeitung einzelner Stationen, Touren und Medien.



The image shows a login form titled "Admin-Login". It contains two input fields: "Benutzername" (Username) and "Passwort" (Password). Below the password field is a blue button labeled "Anmelden" (Login) with a small icon to its left. The form is centered on a light blue background.

Abbildung 6.2.1 - Login Admin-Oberfläche

Der Zugriff auf die Admin-Oberfläche ist durch ein Login geschützt. Erst nach erfolgreicher Anmeldung können die Verwaltungsfunktionen genutzt werden. Dadurch wird verhindert, dass öffentliche Nutzer Inhalte verändern oder löschen können.

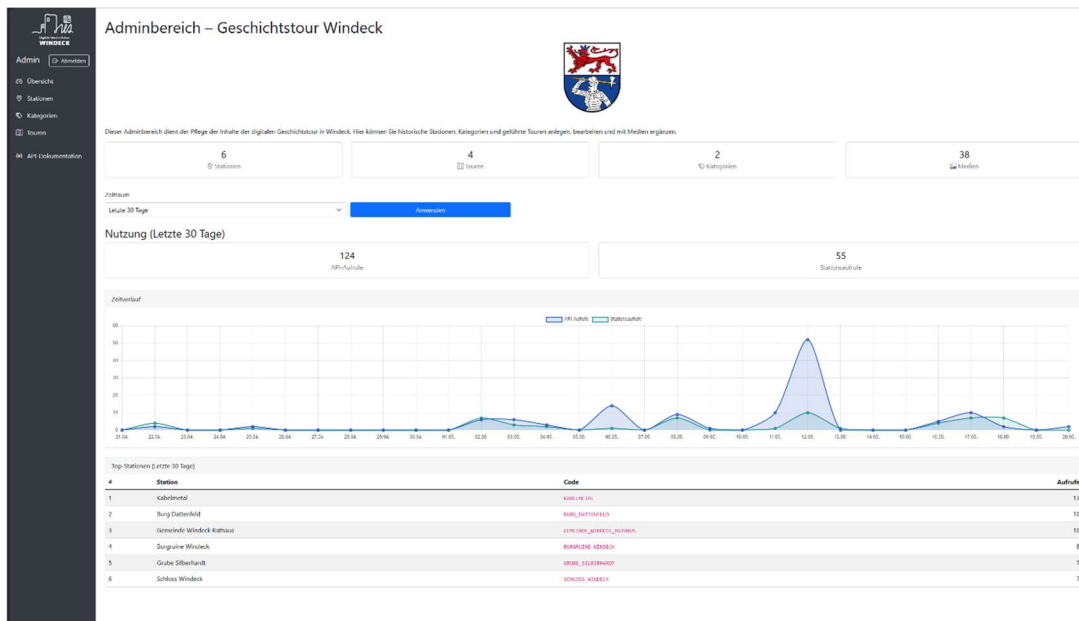


Abbildung 6.2.2 – Landingpage mit Statistiken

Nach der Anmeldung gelangt der Nutzer auf die Landingpage des Admin-Bereichs. Diese zeigt eine kurze Übersicht über den aktuellen Datenbestand, etwa die Anzahl der vorhandenen Stationen, Touren, Kategorien oder Medien. Zusätzlich werden Statistiken zu den täglichen Aufrufen sowie zu den am häufigsten aufgerufenen Stationen angezeigt. Dadurch erhält der Nutzer direkt einen kompakten Überblick über den Umfang der gepflegten Inhalte.

Stationen verwalten

Nach Titel, Code oder Ort suchen...

[Neue Station anlegen](#)

Titel	Code	Kategorie	Ort	Koordinaten	
Burg Dattenfeld	BURG_DATTENFELD		Windeck-Dattenfeld	50.8070, 7.5580	
Burgruine Windeck	BURGRUINE_WINDECK		Windeck-Altwindeck	50.8140, 7.5791	
Gemeinde Windeck Rathaus	GEMEINDE_WINDECK_RATHAUS		Windeck-Rosbach	50.7968, 7.6106	
Grube Silberhardt	GRUBE_SILBERHARDT		Windeck-Öttershagen	50.8231, 7.6425	
Kabelmetal	KABELMETAL		Windeck-Schlädern	50.8057, 7.5882	
Schloss Windeck	SCHLOSS_WINDECK		Windeck-Altwindeck	50.8136, 7.5786	

Abbildung 6.2.3 - Stationsübersicht

Die Stationsübersicht bildet einen zentralen Ausgangspunkt für die redaktionelle Arbeit. Hier werden die vorhandenen Stationen tabellarisch dargestellt. Von dieser Ansicht aus können neue Stationen angelegt sowie bestehende Stationen geöffnet, bearbeitet oder gelöscht werden.

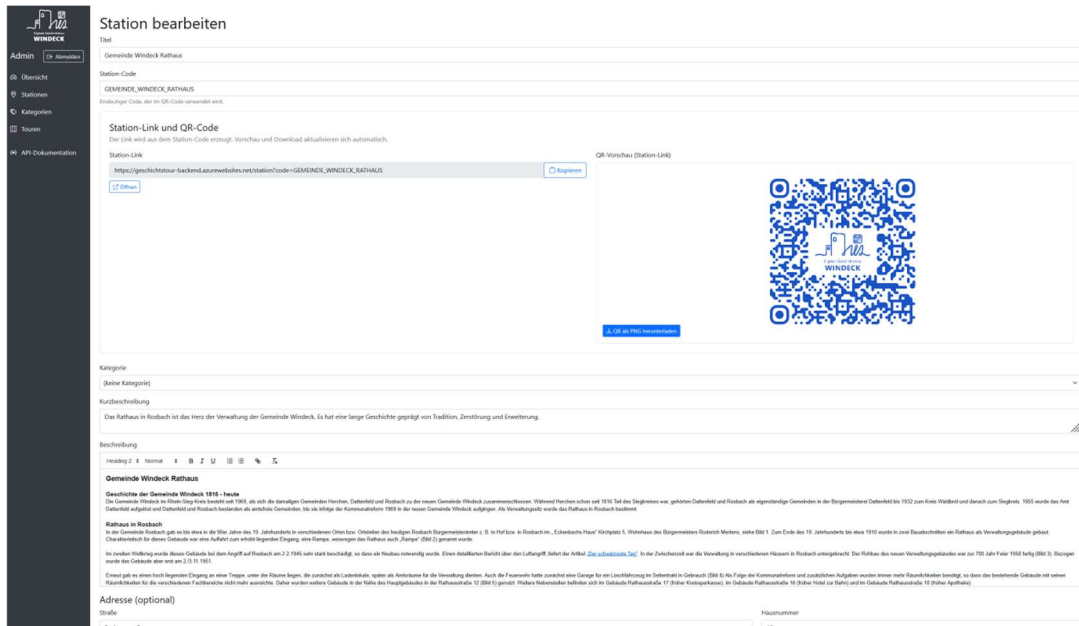


Abbildung 6.2.4 - Station anlegen und bearbeiten

In der Bearbeitungsansicht werden die zentralen Informationen einer Station gepflegt. Dazu gehören insbesondere Titel, Beschreibung, Stationscode, Koordinaten und Kategorie. Der Stationscode ist für die spätere Zuordnung des QR-Codes relevant, da über ihn die passende Station eindeutig identifiziert werden kann. Dass der Stationscode eindeutig ist, wird ebenfalls vor dem Speichern der Änderung überprüft, um Datenbankfehler zu vermeiden. Der Stations-Link und der damit verbundene QR-Code passen sich automatisch an den aktuellen Stationscode an. Der QR-Code kann auf der Seite direkt hochauflösend als *.png-Datei* heruntergeladen und weiterverwendet werden.

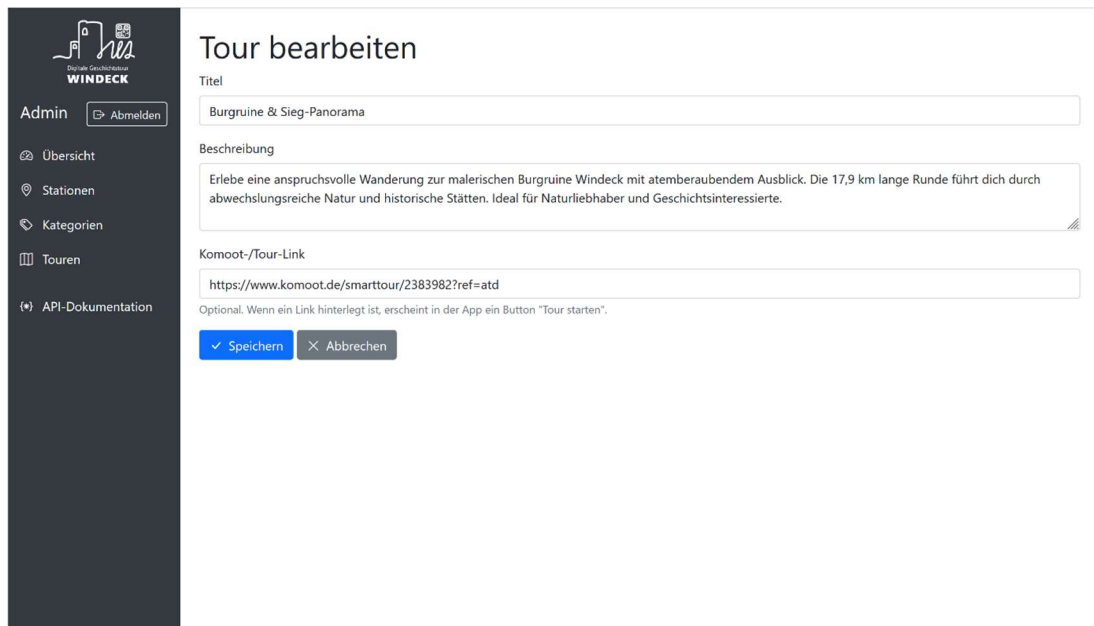


Abbildung 6.2.5 - Tourdetails bearbeiten

Neben einzelnen Stationen können auch Touren verwaltet werden. In der Detailansicht einer Tour werden grundlegende Informationen wie Titel und Beschreibung bearbeitet. Eine Tour dient dazu, mehrere Stationen in einen gemeinsamen thematischen oder räumlichen Zusammenhang zu bringen. Zusätzlich kann ein externer Tour-Link hinterlegt werden, etwa zu einer bereits bestehenden Route auf Komoot.

Stationen für Tour: Burgruine & Sieg-Panorama

Zurück zur Touren-Übersicht

Bestehende Stationseinträge

Reihenfolge	Station	Ort	Koordinaten	
1	Gemeinde Windeck Rathaus	Windeck-Rosbach	50.7968, 7.6106	
2	Burgruine Windeck	Windeck- Altwindeck	50.8140, 7.5791	
3	Schloss Windeck	Windeck- Altwindeck	50.8136, 7.5786	
4	Kabelmetal	Windeck- Schladern	50.8057, 7.5882	

Neuen Stationseintrag hinzufügen

Wählen Sie eine Station aus und legen Sie optional die Reihenfolge fest. Wird keine Reihenfolge angegeben, wird der Eintrag ans Ende der Tour gesetzt.

Station

Bitte Station wählen...

Reihenfolge

Optional. Wenn leer, wird die nächste freie Nummer verwendet.

Eintrag hinzufügen

Abbildung 6.2.6 - Stationen zu einer Tour hinzufügen

Für die Zusammenstellung einer Tour können bestehende Stationen ausgewählt und der jeweiligen Tour hinzugefügt werden. Dabei ist insbesondere die Reihenfolge der Stationen relevant, da sie bestimmt, wie die Tour später in der mobilen App dargestellt wird. Die Auswahl der Stationen aktualisiert sich automatisch, um inkonsistente Datensätze zu vermeiden.

Medien für Station: Gemeinde Windeck Rathaus

Zur Station Aktualisieren Neues Medium anlegen

Sortierung	Typ	URL	Speicherplatz	Caption
1	Image	/uploads/stations/1/645a33b0-fd93-40e9-8d75-2e6578fb3d4.jpg	ca. 276 KB	Bürgermeisteramt Rosbach aus 1910
2	Image	/uploads/stations/1/f165890d-4500-433f-b586-49c183902c82.jpg	ca. 493 KB	Altes Rathaus "Rampe" 1912
3	Image	/uploads/stations/1/759dfc0b-22b8-4df7-b3a0-4925cb1d41f0.jpg	ca. 217 KB	Rohbau Rathaus 1950
4	Image	/uploads/stations/1/895a434d-e242-497a-8231-a48ba1d21b62.jpg	ca. 334 KB	Rathaus 1956
5	Image	/uploads/stations/1/d18e0680-c568-4bf0-bbee-b23a35879d29.jpg	ca. 479 KB	Rathaus und Alte Apotheke

Abbildung 6.2.7 - Medienübersicht

Die Medienübersicht zeigt die einer Station zugeordneten Medienobjekte. Dadurch lässt sich nachvollziehen, welche Bilder bereits hinterlegt sind und welche Inhalte ergänzt oder entfernt werden können.

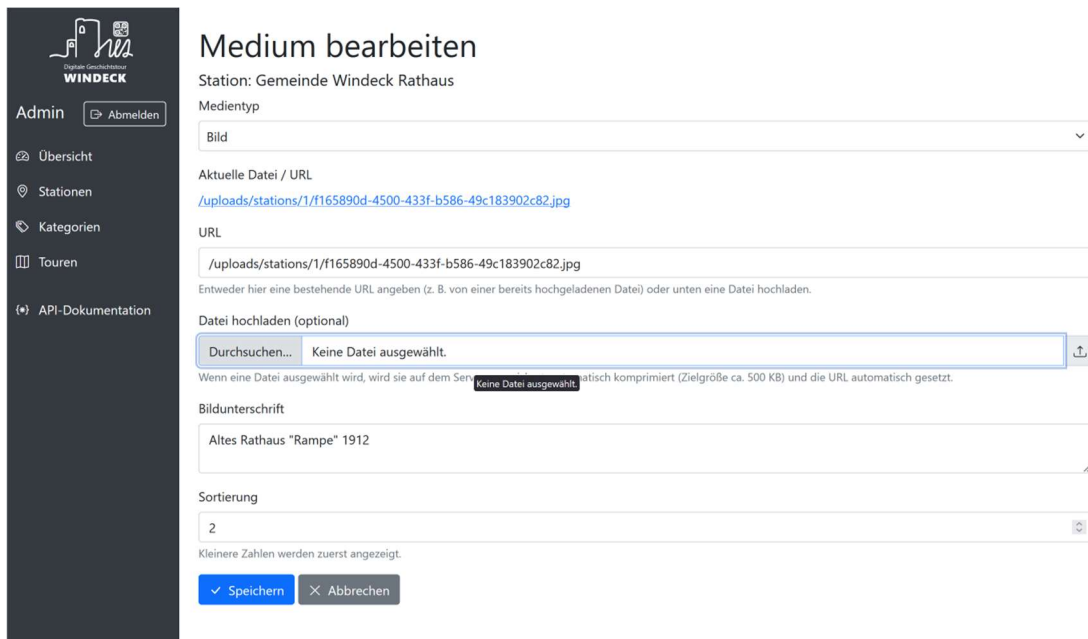


Abbildung 6.2.8 - Medienobjekt hinzufügen

Über die Eingabemaske zum Hinzufügen eines Medienobjekts können neue Bilder einer Station zugeordnet werden. Die Dateien können entweder per *Drag & Drop* abgelegt oder über den Datei-Explorer ausgewählt werden. Neben der Datei kann ebenfalls der Medientyp, eine Kurzbeschreibung sowie die Position in der Reihenfolge angegeben werden. Eine Besonderheit ist zudem, dass die Datei beim Speichern nicht einfach nur auf dem Server abgelegt, sondern vorher deutlich in ihrer Größe komprimiert wird. Dabei wird eine maximale Größe von 500 KB angestrebt. Dadurch sollen Stationen schneller geladen werden, da die zu übertragende Datenmenge reduziert wird.

Die Admin-Oberfläche bildet die zentrale Pflegeschicht des Systems. Sie ermöglicht es, die Inhalte der digitalen Geschichtstour langfristig zu erweitern und aktuell zu halten, ohne dass Änderungen direkt am Quellcode oder an der Datenbank vorgenommen werden müssen. Die Screenshots dokumentieren dabei die wichtigsten Arbeitsbereiche und zeigen, dass die Verwaltung der Inhalte entlang klar abgegrenzter Funktionen erfolgt.

5.4 Authentifizierung

Der Administrationsbereich ist durch eine Anmeldung geschützt, da dort Inhalte angelegt, bearbeitet und gelöscht werden können. Ohne Zugriffsbeschränkung könnten unberechtigte Nutzer Stationen, Touren oder Medien verändern und damit die bereitgestellten Inhalte manipulieren.

Für die Authentifizierung wird ein einfaches Login mit Benutzername und Passwort verwendet. Die Zugangsdaten werden nicht fest im Quellcode hinterlegt, sondern über Umgebungsvariablen bereitgestellt, welche im Admin-Bereich von Azure festgelegt werden können.

Nach erfolgreicher Anmeldung wird der Nutzer über eine *Cookie-Authentifizierung* angemeldet. Dadurch bleibt der Login über mehrere Seitenaufrufe hinweg bestehen, ohne dass sich der Nutzer bei jeder Verwaltungsseite erneut anmelden muss. Der Admin-Bereich wird zusätzlich so konfiguriert, dass alle darin enthaltenen *Razor Pages* nur für authentifizierte Nutzer erreichbar sind. Nicht angemeldete Nutzer werden automatisch auf die Login-Seite weitergeleitet.

Die Lösung ist bewusst einfach gehalten, da im Rahmen des Projekts nur ein kleiner Kreis berechtigter Personen Inhalte pflegen soll. Eine komplexe Benutzer- und Rollenverwaltung wird daher nicht umgesetzt, wäre jedoch bei einer späteren Erweiterung, etwa auf mehrere Gemeinden oder Bearbeitergruppen, sinnvoll. Die Umsetzung orientiert sich an den Authentifizierungs- und Autorisierungsmöglichkeiten von *ASP.NET Core*.^[21]

6 Mobile App

Dieses Kapitel beschreibt die mobile Anwendung der digitalen Geschichtstour. Während das Backend die Inhalte zentral bereitstellt und die Admin-Oberfläche deren Pflege ermöglicht, bildet die App den eigentlichen Zugang für die Anwender. Über sie können Stationen, Touren und Medien abgerufen, historische Orte auf einer Karte betrachtet und einzelne Stationen über QR-Codes direkt geöffnet werden. Der vollständige Quellcode ist unter <https://github.com/tengelberth/Windeck.Geschichtstour> einsehbar.

6.1 App-Architektur

Die mobile App wurde mit *.NET MAUI* umgesetzt und folgt dem *Model-View-ViewModel* Muster (*MVVM*). Dieses Architekturmodell trennt die Benutzeroberfläche von der Präsentationslogik und dem zugrunde liegenden Datenmodell. Nach der Microsoft Dokumentation besteht *MVVM* aus den drei Kernkomponenten *View*, *ViewModel* und *Model*. Die *View* ist für die Darstellung zuständig, das *ViewModel* stellt Daten und Befehle für die Oberfläche bereit, während das *Model* die eigentlichen Datenstrukturen beziehungsweise Domänenobjekte abbildet. Ziel dieser Trennung ist es, Benutzeroberfläche und Logik möglichst unabhängig voneinander entwickeln, testen und warten zu können, um Redundanzen zu reduzieren und die Wiederverwendung zu maximieren.^[22] Microsoft beschreibt *MVVM* ausdrücklich als Muster zur klaren Trennung von Geschäfts- und Präsentationslogik von der Benutzeroberfläche.

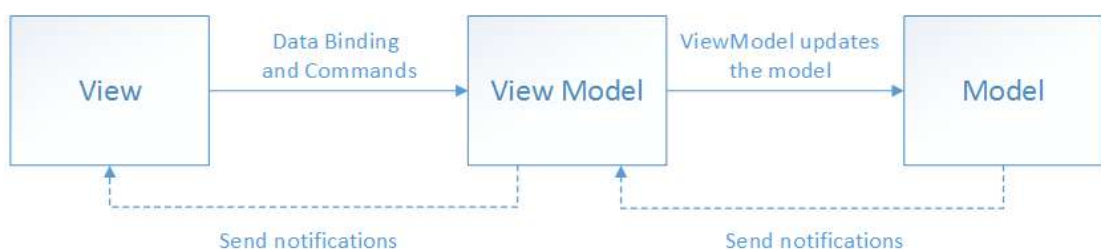


Abbildung 7.1.1 – MVVM-Muster nach Microsoft ^[22]

Die *Abbildung 7.1.1* aus der Microsoft Dokumentation zeigt die grundlegende Beziehung zwischen *View*, *ViewModel* und *Model*. Die *View* kommuniziert über *Data Binding* und *Commands* mit dem *ViewModel*. Das *ViewModel* aktualisiert das *Model* und erhält von diesem Änderungsbenachrichtigungen.

Ebenso werden Änderungen aus dem *ViewModel* an die *View* zurückgemeldet, sodass die Oberfläche auf neue Daten oder Zustände reagieren kann.

Auf die digitale Geschichtstour übertragen bedeutet dies, dass die einzelnen Seiten der App als *Views* umgesetzt werden. Dazu gehören beispielsweise die Stationsübersicht, die Stationsdetailseite, die Tourenansicht und die Kartenansicht. Die zugehörigen *ViewModels* enthalten die Logik zum Laden und Aufbereiten der Daten, zur Verarbeitung von Nutzereingaben und zur Navigation zwischen den Seiten. Die *Models* beziehungsweise *Data Transfer Objects (DTOs)* beschreiben die Datenstrukturen, die über die *API* vom *Backend* empfangen werden, etwa Stationen, Touren, Kategorien und Medienobjekte.

Die Navigation innerhalb der App erfolgt über die *AppShell*. Sie stellt die zentrale Navigationsstruktur der Anwendung bereit und verbindet die wichtigsten Bereiche, etwa Startseite, Stationsübersicht, Tourenübersicht und Informationsseiten. Der Datenabruf erfolgt über *HTTP-Anfragen* an die vom *Backend* bereitgestellte *API*. Die mobile App greift dabei ausschließlich lesend auf die Inhalte zu. Schreibende Operationen, wie das Anlegen oder Bearbeiten von Stationen, bleiben der Admin-Oberfläche vorbehalten.

6.2 Anwendersicht

Dieses Kapitel beschreibt die mobile App aus Sicht des Anwenders. Im Vordergrund steht dabei nicht die technische Implementierung, sondern die Frage, wie die Inhalte innerhalb der Anwendung dargestellt und genutzt werden können. Die folgenden Screenshots zeigen die verschiedenen Seiten der App und den Aufbau der Benutzeroberfläche.

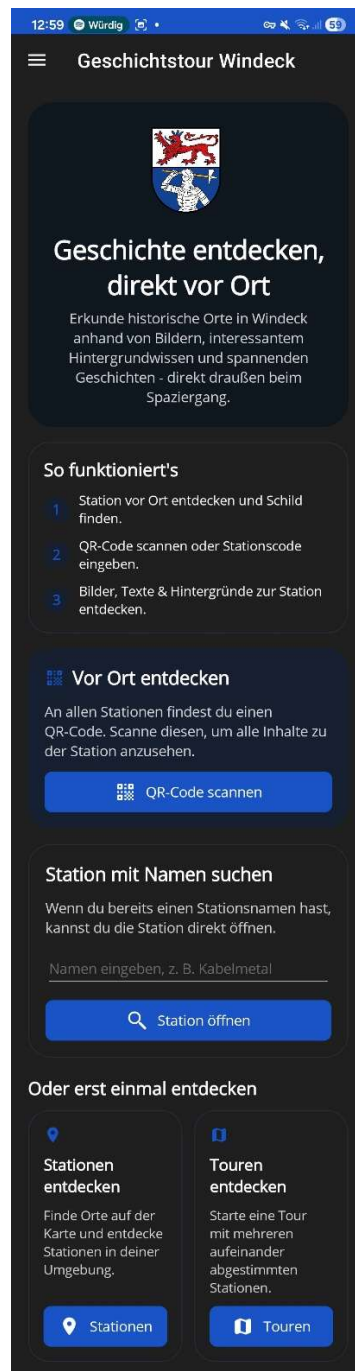


Abbildung 7.2.1 - Startseite

Die Startseite bildet den Einstiegspunkt der Anwendung. Sie gibt dem Anwender einen ersten Überblick über die digitale Geschichtstour und führt zu den wichtigsten Bereichen der App. Von hier aus können Stationen, Touren und weitere Informationen aufgerufen und gesucht werden.



Abbildung 7.2.2 - Listenansicht der Stationen

In der Stationsübersicht werden die vorhandenen Orte in Listenform dargestellt. Der Anwender kann die Stationen durchsuchen und einzelne Einträge auswählen. Die Listenansicht eignet sich besonders für Nutzer, die gezielt nach einem bestimmten Ort suchen oder sich zunächst einen Überblick über die verfügbaren Stationen verschaffen möchten.

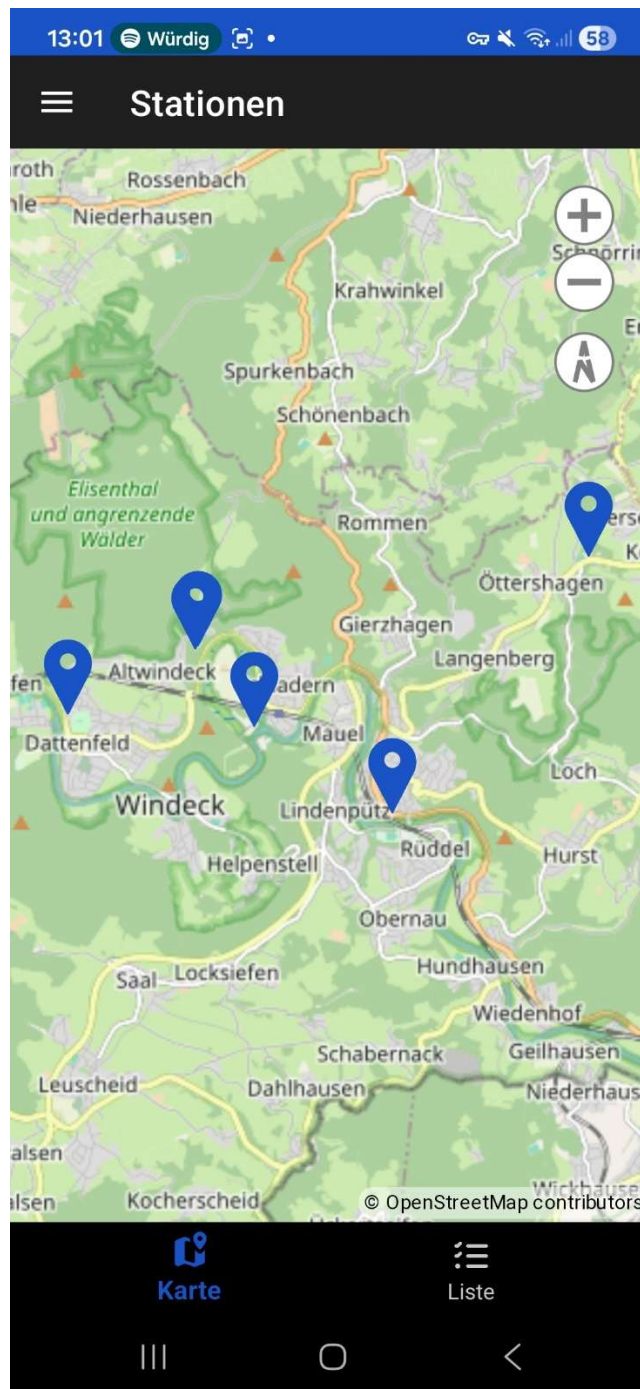


Abbildung 7.2.3 - Kartenansicht der Stationen

Die Kartenansicht ergänzt die Listenansicht um eine räumliche Darstellung der Stationen. Die historischen Orte werden als Markierungen auf der Karte angezeigt. Dadurch kann der Anwender erkennen, welche Stationen sich in seiner Umgebung befinden oder wie einzelne Orte räumlich zueinander liegen. Per Klick gelangt er direkt zur ausgewählten Station.



Abbildung 7.2.4 – Kameran scan eines QR-Codes

Die App enthält eine Kamerafunktion zum Scannen der QR-Codes. Wird an einer Station ein QR-Code gescannt, liest die Anwendung den darin enthaltenen Link beziehungsweise Stationscode aus und öffnet anschließend die passende Stationsdetailseite. Dadurch muss der Anwender die Station nicht manuell in der Liste oder Karte suchen, sondern gelangt direkt zum zugehörigen Inhalt.

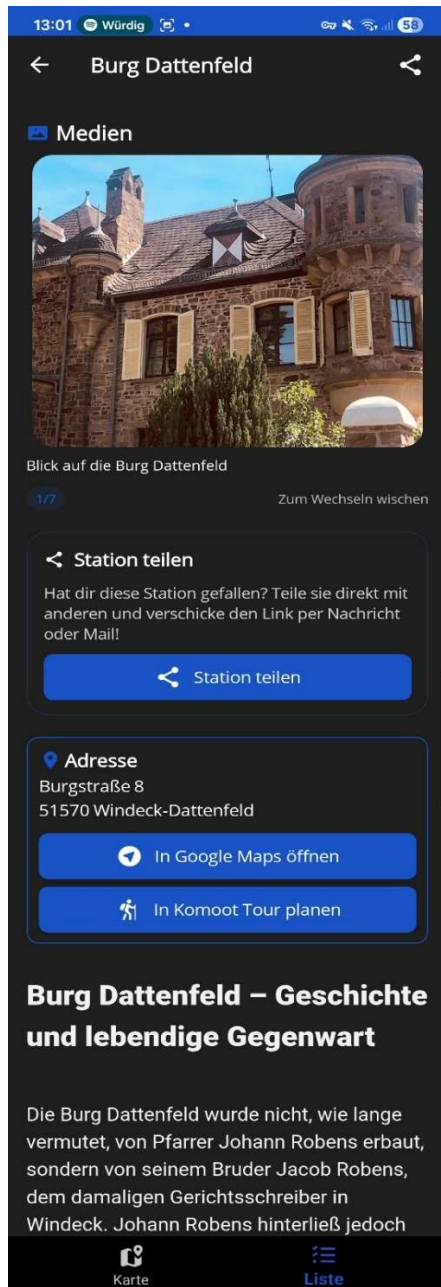


Abbildung 7.2.5 / 7.2.6 - Detailseite einer Station

Die Detailseite stellt alle Informationen zu einer einzelnen Station bereit. Dazu gehören der Titel, eine Beschreibung, zugeordnete Bilder sowie gegebenenfalls weitere Angaben zur Station. Neben den Informationen kann der Nutzer auch direkt zu den Stationen per Google Maps oder Komoot navigiert werden. Stationen können ebenfalls geteilt werden.

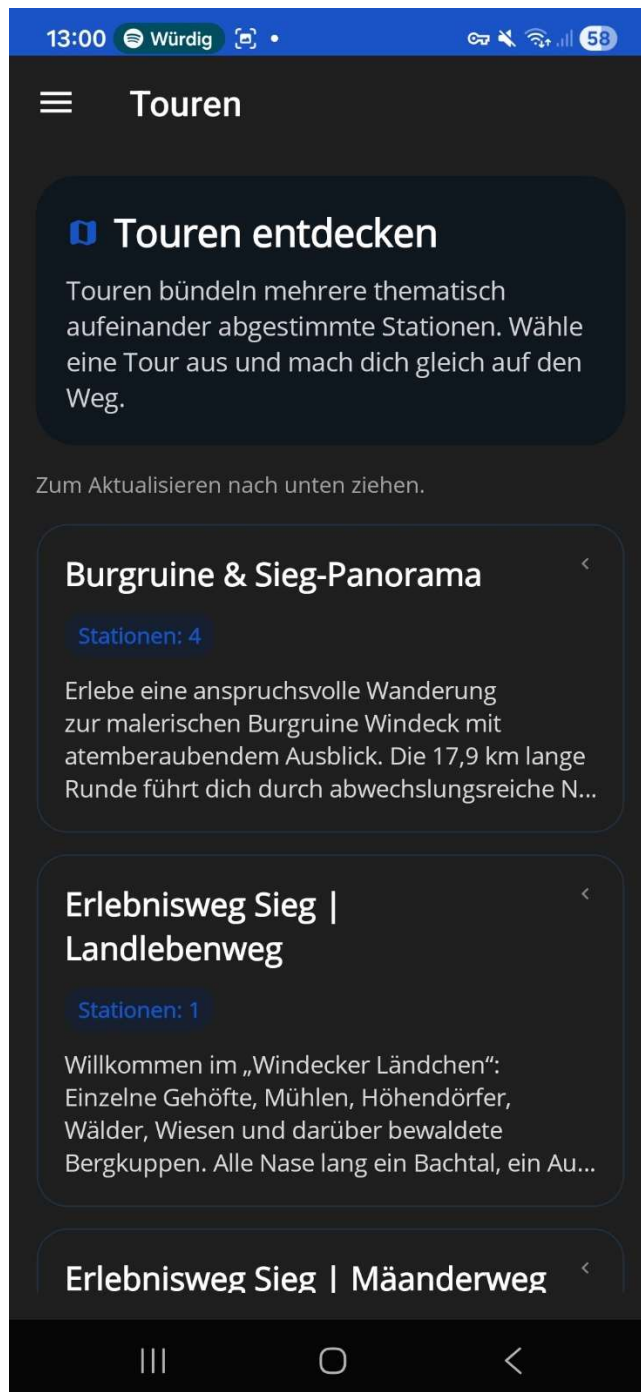


Abbildung 7.2.7 - Listenansicht der Touren

Neben einzelnen Stationen können auch Touren aufgerufen werden. Die Tourenübersicht zeigt die vorhandenen Rundgänge beziehungsweise thematischen Zusammenstellungen. Der Anwender kann dadurch nicht nur einzelne Orte betrachten, sondern mehrere Stationen in einem gemeinsamen Zusammenhang erleben.

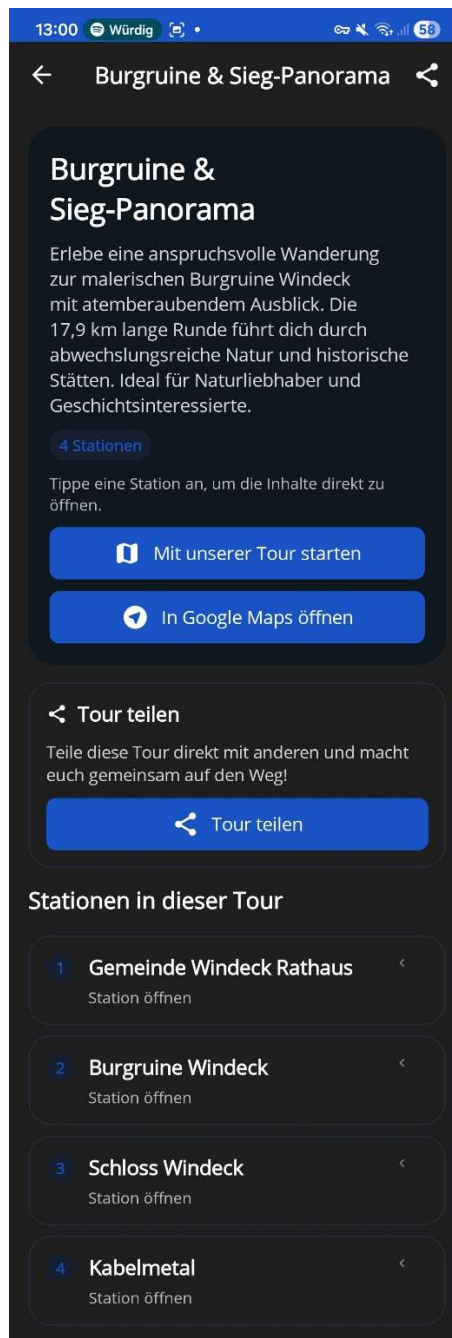


Abbildung 7.2.8 - Detailseite einer Tour

Die Detailseite einer Tour zeigt die zugehörigen Stationen und deren Reihenfolge. Dadurch wird nachvollziehbar, welche Orte Teil der Tour sind und wie sie nacheinander besucht werden können. Die Touransicht unterstützt damit insbesondere geplante Rundgänge, Spaziergänge oder Fahrradtouren. Per Klick kann eine vorher angegebene Tour oder eine Navigation per Google Maps gestartet werden. Touren können ebenfalls geteilt werden.



Abbildung 7.2.9 - Allgemeine Informationsseite

Die Informationsseite enthält allgemeine Hinweise zur digitalen Geschichtstour. Dort können beispielsweise Informationen zum Projekt, zur Nutzung der App oder zu beteiligten Vereinen bereitgestellt werden. Sie ergänzt die fachlichen Inhalte der Stationen um projektbezogene Informationen.

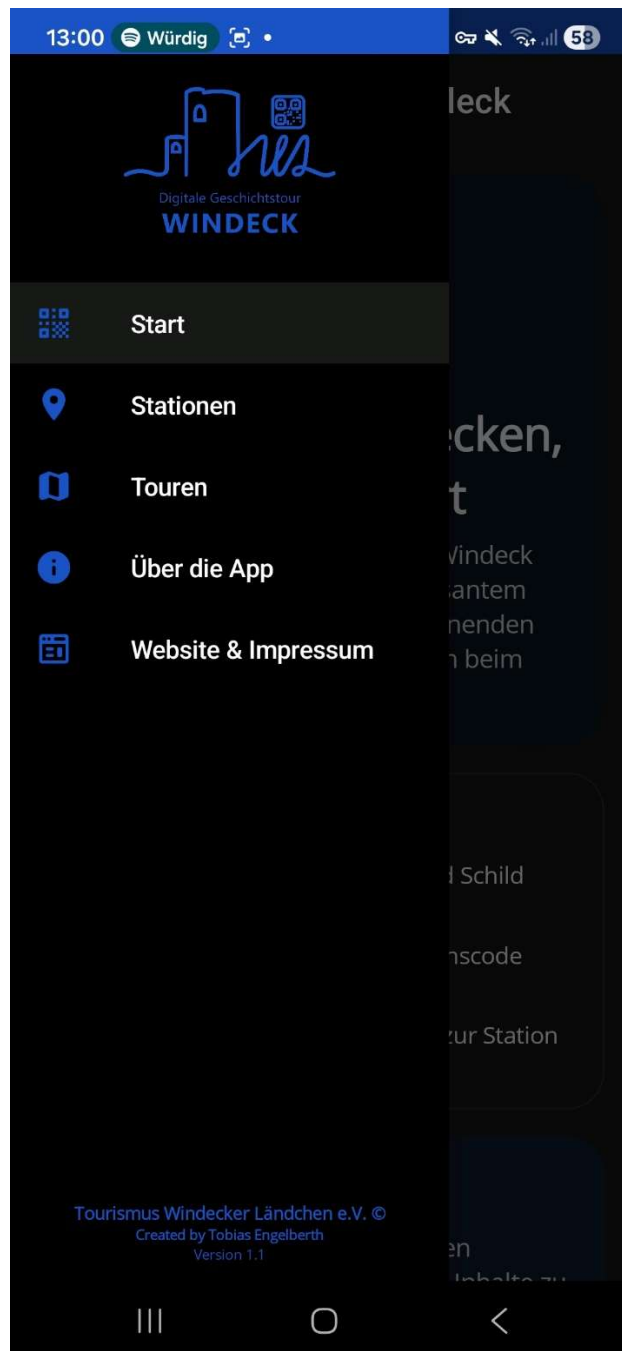


Abbildung 7.2.10 - Ausgeklapptes Flyout-Menü

Das Flyout-Menü dient der Navigation innerhalb der App. Es ermöglicht den schnellen Wechsel zwischen den zentralen Bereichen, etwa Startseite, Stationen, Touren, Karte und Informationsseite. Durch die gebündelte Navigation bleibt die Anwendung auch bei mehreren Funktionsbereichen übersichtlich. Über das Menü kann man ebenfalls zur Projektwebsite und Impressum gelangen.

Die Anwendersicht zeigt, dass die App auf eine einfache und unkomplizierte Nutzung ausgerichtet ist. Anwender können historische Orte entweder über Listen, über die Karte, über Touren oder über QR-Codes erreichen. So unterstützt die App verschiedene Nutzungssituationen, von der spontanen Information vor Ort bis zur geplanten Erkundung mehrerer Stationen.

Die dargestellten Ansichten können auch außerhalb dieser Arbeit nachvollzogen werden. Die Anwendung ist öffentlich über die jeweiligen App Stores bereitgestellt. Für *Android* ist die App im *Google Play Store* unter <https://play.google.com/store/apps/details?id=de.tobias.engelberth.win-deck.geschichtstour> verfügbar. Für *iOS* kann sie über den *Apple App Store* unter <https://apps.apple.com/de/app/geschichtstour-windeck/id6760314953> aufgerufen werden.

6.3 QR-Codes & Deep Links

Ein zentraler Bestandteil der mobilen App ist der direkte Aufruf einzelner Stationen über einen QR-Code. Dadurch muss ein Anwender eine Station nicht manuell über die Listen oder Kartenansicht suchen, sondern kann vor Ort den angebrachten QR-Code scannen und unmittelbar zur passenden Detailseite gelangen.

Technisch basiert dieser Ablauf auf einem stationsbezogenen Link. Jeder QR-Code enthält eine *URL*, in der der jeweilige Stationscode übergeben wird. Das verwendete Linkformat lautet:

```
https://geschichtstour-backend.azurewebsites.net/station  
?code=<Stationscode>
```

Der Platzhalter *<Stationscode>* wird durch den eindeutigen Code der jeweiligen Station ersetzt. Beim Aufruf dieses Links kann die Anwendung erkennen, welche Station angefordert wurde. Ist die App installiert und der Link korrekt mit der App verknüpft, wird die passende Stationsdetailseite direkt geöffnet. Ist die App nicht installiert, kann der Link stattdessen auf eine Webseite führen, über die weitere Informationen zum Projekt und die Store-Links bereitgestellt werden.

Auf *Android* wird dieser Mechanismus über *App-Links* beziehungsweise *Intent-Filter* umgesetzt. Unter *iOS* erfolgt die entsprechende Zuordnung über *Universal Links* und die zugehörige *Apple App-Site-Association-Datei*. In beiden Fällen wird geprüft, ob die aufgerufene Domain mit einer installierten App verknüpft ist. Innerhalb der App wird der eingehende Link verarbeitet, der Stationscode aus der *URL* ausgelesen und anschließend die Detailansicht der zugehörigen Station geladen.^{[23] [24]}

Der Ablauf lässt sich vereinfacht wie folgt darstellen:

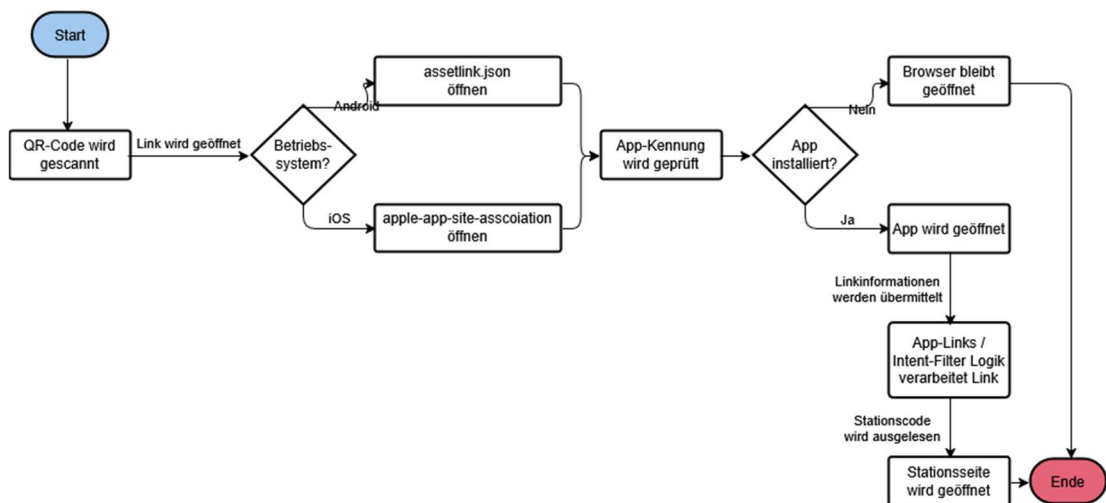


Abbildung 7.3.1 - Ablauf des Stationsaufrufs über QR-Code

Die QR-Code-Funktion erfüllt damit eine wichtige Rolle für die Nutzerführung. Sie reduziert den Suchaufwand, verbindet reale Orte eindeutig mit digitalen Inhalten und unterstützt insbesondere spontane Nutzungssituationen vor Ort. Gleichzeitig bleibt der Link auch ohne App nutzbar, da er auf die Projektseite und damit eine Downloadmöglichkeit verweist. Dadurch entsteht ein niedrigschwelliger Zugang zur digitalen Geschichtstour, unabhängig davon, ob die App bereits installiert ist.

7 Bereitstellung und Veröffentlichung

Das folgende Kapitel beschreibt, wie die digitale Geschichtstour bereitgestellt und so für Interessierte nutzbar wird.


7.1 Hosting

Das Hosting der Website, des Admin-Bereichs sowie die Bereitstellung der Daten erfolgt vollständig über *Microsoft Azure*.

Die Webanwendung sowie die *Web-API* werden über einen *Azure App Service* bereitgestellt. Auf diese Weise können sowohl die öffentliche Projektwebsite als auch der Admin-Bereich über dieselbe Infrastruktur betrieben werden. Die Anwendung ist unter der Domain <https://geschichtstour-backend.azurewebsites.net> sowie unter der Custom Domain <https://geschichtstour.windecker-ladenchen.com> erreichbar. Für letztere ist derzeit jedoch kein *SSL/TLS-Zertifikat* hinterlegt.









Um die laufenden Betriebskosten gering zu halten, wird für den *App Service* ein *D1-Shared-Plan* verwendet. Diese Entscheidung steht jedoch in einem Spannungsfeld zur Verfügbarkeit der Anwendung und wird im *Kapitel 9.2* näher betrachtet. Die Veröffentlichung erfolgt direkt aus *Visual Studio* nach *Azure*, wodurch Änderungen innerhalb kürzester Zeit eingespielt und veröffentlicht werden können. Die relationale Datenhaltung erfolgt über *Azure SQL*.

Azure services



Resources

Recent Favorite

Name	Type	Last Viewed
 sqldb-geschichtstour (sqlsvr-geschichtstour/sqldb-geschichtstour)	SQL database	2 minutes ago
 Windeck Geschichtstour	Subscription	3 minutes ago
 sqlsvr-geschichtstour	SQL server	5 minutes ago
 geschichtstour-backend	App Service	5 minutes ago
 ASP-rggeschichtstour-a59d	App Service plan	2 weeks ago
 rg-geschichtstour	Resource group	a month ago
 geschichtstour-backend	Application Insights	a month ago
 DefaultWorkspace-943eef56-1929-44f6-94d1-3d952340660f-WEU	Log Analytics workspace	3 months ago

See all

Abbildung 8.1.1 – Azure Services Dashboard

7.2 Veröffentlichung der App

Die App wird für *Android* über den *Google Play Store* und für *iOS* über den *Apple App Store* bereitgestellt. Für die Veröffentlichung ist auf beiden Plattformen jeweils ein Entwicklerkonto erforderlich, über das Anwendungen verwaltet, signiert und in den jeweiligen Store eingereicht werden können. Für *Apple* erfolgt die Verwaltung über *App Store Connect*, während für *Android* die *Google Play Console* verwendet wird.

Vor der Veröffentlichung muss die Anwendung im *Release-Modus* für die jeweilige Zielplattform erstellt und signiert werden. Für *Android* wird dabei ein *Android App Bundle* im Format *.aab* erzeugt, das anschließend in die *Google Play Console* hochgeladen wird. *Google Play* verarbeitet dieses *Bundle* weiter und generiert daraus die für verschiedene Geräte benötigten *APK*-Varianten.

Für *iOS* wird die Anwendung als signierter *Build* (*.app*-Format) erzeugt und über die von *Apple* vorgesehenen Werkzeuge an *App Store Connect* übertragen. Dort wird sie zusammen mit dem Store-Eintrag eingereicht und anschließend von *Apple* geprüft. Erst nach erfolgreicher Überprüfung kann die App im *App Store* veröffentlicht werden.^[20]

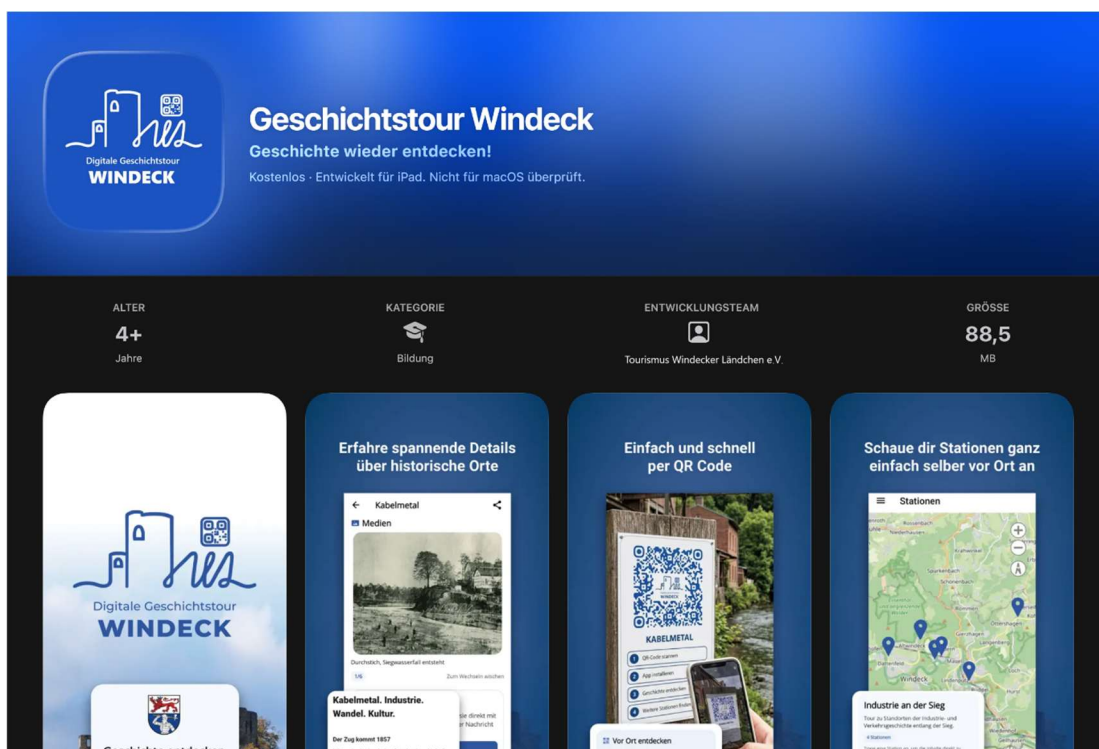


Abbildung 8.2.1 – Store Eintrag im Apple App Store

7.3 Bereitstellung im Ort

Die Bereitstellung im Ort erfolgt über kleine gedruckte Schilder, die an den jeweiligen Stationen angebracht werden. Sie enthalten den stationsspezifischen QR-Code, den Namen des Ortes sowie eine kurze Anleitung zur Nutzung.

Ist die App bereits auf dem Endgerät installiert, führt das Scannen des QR-Codes per *Deep Link* unmittelbar zur passenden Station innerhalb der Anwendung. Ist die App noch nicht installiert, werden Nutzer zunächst auf die Projektwebsite weitergeleitet. Dort können sie sich über das Projekt informieren und über einen bereitgestellten Link die App herunterladen.

Die Schilder haben ein Format von etwa 15 × 20 cm. *Abbildung 8.3.1* zeigt beispielhaft das Schild der Station „Kabelmetal“.

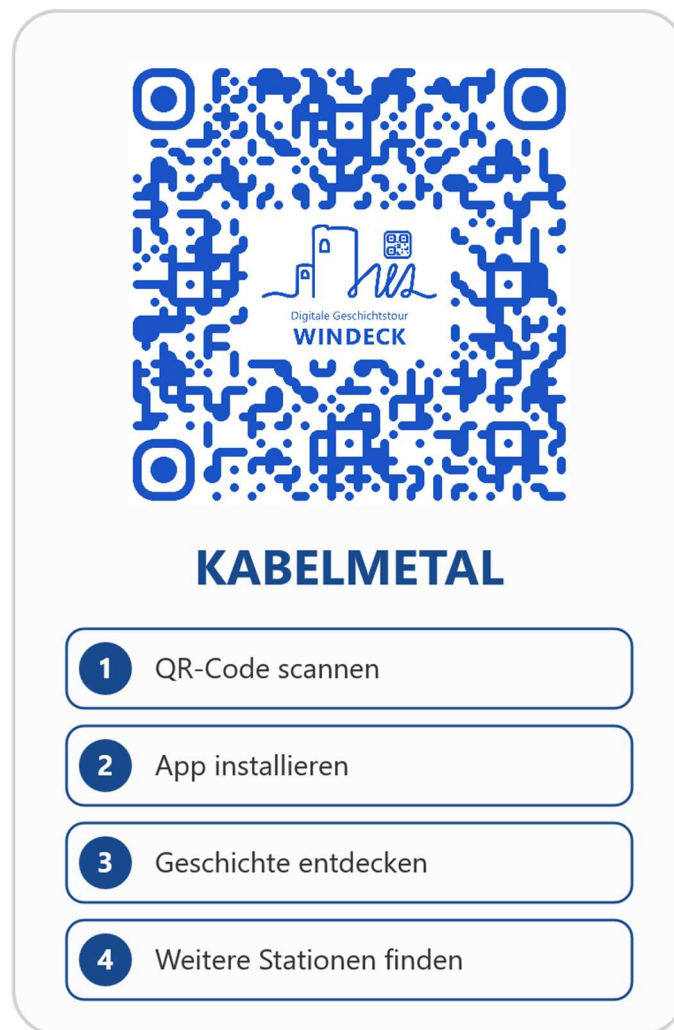


Abbildung 8.3.1 – Stationsschild „Kabelmetal“

8 Fazit & Ausblick

Das abschließende Kapitel fasst die wesentlichen Ergebnisse der Arbeit zusammen und ordnet das entwickelte System hinsichtlich seines praktischen Nutzens ein. Anschließend werden die Grenzen und Herausforderungen der Umsetzung betrachtet. Darauf aufbauend werden mögliche Weiterentwicklungen beschrieben, die über den aktuellen Funktionsumfang hinausgehen und die digitale Geschichtstour langfristig erweitern könnten.

8.1 Zusammenfassung der Projektergebnisse

Im Rahmen der Arbeit wurde eine digitale Geschichtstour für die Gemeinde Windeck konzipiert und technisch umgesetzt. Entstanden ist ein Gesamtsystem, das aus einer mobilen App, einer webbasierten Admin-Oberfläche, einer serverseitigen *API* und einer relationalen Datenbank besteht. Die einzelnen Komponenten greifen ineinander und ermöglichen es, lokalhistorische Inhalte zentral zu verwalten und für Anwender mobil bereitzustellen.

Die mobile App bildet den Zugang für Anwender. Über sie können Stationen, Touren und Medien abgerufen werden. Zusätzlich ermöglichen Kartenansicht, Tourenübersicht und QR-Codes verschiedene Wege, historische Orte zu entdecken. Die Admin-Oberfläche stellt die redaktionelle Pflegeschicht des Systems dar. Dort können Stationen, Touren, Kategorien und Medien ohne direkten Zugriff auf Datenbank oder Quellcode verwaltet werden.

Technisch wurde eine modulare Architektur umgesetzt, bei der mobile App, *Backend* und Datenbank klar voneinander getrennt sind. Das *Backend* stellt die Daten über *API-Endpunkte* bereit und übernimmt zugleich die Verwaltung der Inhalte über *Razor Pages*. Die persistente Speicherung erfolgt in einer *Azure SQL Datenbank*, der Zugriff darauf über *Entity Framework Core*.

Damit wurde eine praxistaugliche Grundlage geschaffen, um historische Orte in Windeck digital sichtbar zu machen. Das Projekt zeigt, dass sich lokale Geschichtsvermittlung mit überschaubarem technischem Aufwand in eine moderne, erweiterbare Anwendung überführen lässt. Die entwickelte Lösung ersetzt analoge Angebote nicht vollständig, sondern ergänzt sie durch zentrale Pflege, flexible Erweiterbarkeit und direkten mobilen Zugriff.

Die Digitale Geschichtstour Windeck wird von der Gemeinde Windeck genutzt und laufend erweitert. Weitere Informationen sind auf der Projektwebsite unter <https://geschichte-tour-backend.azurewebsites.net/> sowie auf der Website des Tourismus Windecker Ländchen e. V. unter <https://windecker-laendchen.com/> verfügbar.

8.2 Grenzen und Herausforderungen

Trotz der erfolgreichen technischen Umsetzung bleiben Problemstellungen, die für den praktischen Einsatz der digitalen Geschichtstour relevant sind. Die größte Herausforderung liegt dabei weniger in der reinen Softwareentwicklung, sondern in der langfristigen Akzeptanz und Pflege der Plattform. Die Anwendung kann nur dann dauerhaft einen Mehrwert bieten, wenn ausreichend viele und qualitativ ansprechende Inhalte vorhanden sind und diese regelmäßig erweitert oder aktualisiert werden. Bleibt die inhaltliche Pflege aus, verliert auch eine technisch funktionierende Anwendung mit der Zeit an Relevanz.

Eine weitere Herausforderung besteht im Spannungsfeld zwischen Betriebskosten und Verfügbarkeit. Leistungsfähigere Hosting-Modelle bieten in der Regel bessere Reaktionszeiten und höhere Verfügbarkeit, verursachen jedoch auch höhere laufende Kosten. Im Rahmen des Projekts musste daher ein kostengünstiger Betrieb priorisiert werden. Diese Entscheidung kann zu Startverzögerungen (*Cold-Starts*) führen, insbesondere wenn Server oder Datenbank nach einer Phase ohne Nutzung erst wieder aktiviert werden müssen. Solche Verzögerungen können aus Sicht des Anwenders störend wirken, wurden jedoch aufgrund der begrenzten finanziellen Rahmenbedingungen in Kauf genommen.

Für den Webserver konnte dieses Problem teilweise reduziert werden, indem regelmäßige Aufrufe (*PINGs*) im Hintergrund verhindern, dass die Anwendung vollständig in einen inaktiven Zustand übergeht. Für die Datenbank ist ein vergleichbarer Ansatz wirtschaftlich weniger sinnvoll, da ein dauerhaft aktiver Betrieb die laufenden Kosten deutlich erhöhen würde. Eine vollständige Lösung dieses Problems wäre daher nur durch ein leistungsfähigeres Hosting-Modell möglich. Der gewählte günstigere Betrieb bringt außerdem Einschränkungen bei der Nutzung der *Custom-Domain* mit sich, insbesondere im Zusammenhang mit der Bereitstellung einer *SSL/TLS*-Verbindung.

Um diese Einschränkung zu reduzieren, wurden mehrere technische Maßnahmen umgesetzt. In der mobilen App wurde ein lokales *Caching* integriert, durch das bereits abgerufene Inhalte zwischengespeichert werden. Werden dieselben Daten erneut benötigt, können sie aus dem lokalen Speicher geladen werden, ohne dass unmittelbar eine erneute Anfrage an das *Backend* beziehungsweise die Datenbank erforderlich ist. Im Hintergrund werden die *Caching-Daten* auf Aktualität abgeglichen und aktualisiert. Zusätzlich wird bereits beim Öffnen der App eine erste Anfrage an das *Backend* ausgelöst. Dieser frühe Aufruf dient dazu, die serverseitigen Komponenten und die Datenbank bereits vor der ersten aktiven Nutzeranfrage zu aktivieren. So wird der Nutzer beim Öffnen einer Station nicht mit der vollständigen Startverzögerung konfrontiert. Ergänzend dazu wurden interaktive Ladezustände in die App aufgenommen. Während Daten geladen werden, erhält der Nutzer interaktive Rückmeldung zu seiner Anfrage. Das verbessert zwar nicht die technische Ladezeit, erhöht aber die wahrgenommene Reaktionsfähigkeit der App.

Eine weitere Grenze betrifft die Mobilfunkversorgung im ländlichen Raum. Da die digitale Geschichtstour vor allem an realen Orten genutzt werden soll, ist die Verfügbarkeit einer mobilen Internetverbindung ein wichtiger Faktor. Gerade an abgelegenen Stationen kann der Abruf digitaler Inhalte erschwert sein. Aus diesem Grund muss die Anwendung möglichst effizient mit Daten umgehen. Dies betrifft insbesondere Bilder, da sie im Vergleich zu Texten deutlich größere Datenmengen verursachen. Im Projekt werden Bilddateien daher vor der Bereitstellung komprimiert, um einen Kompromiss zwischen Dateigröße und Darstellungsqualität zu erreichen.

Insgesamt zeigen diese Punkte, dass die Herausforderungen des Projekts nicht mit der technischen Fertigstellung enden. Für einen nachhaltigen Betrieb sind neben einer funktionierenden Software auch inhaltliche Pflege, wirtschaftlich tragfähiges Hosting und eine möglichst robuste Nutzung unter realen Bedingungen erforderlich. Die digitale Geschichtstour bildet hierfür eine Grundlage, bleibt jedoch auf kontinuierliche Betreuung und Weiterentwicklung angewiesen.

8.3 Mögliche Weiterentwicklungen

Die Weiterentwicklung digitaler Anwendungen ist ein zentraler Aspekt, um ihre langfristige Attraktivität und Relevanz sicherzustellen. Bereits bei der Konzeption und Implementierung der Anwendung wurde daher darauf geachtet, eine Architektur zu schaffen, die zukünftige Erweiterungen mit möglichst geringem Aufwand ermöglicht.

Eine erste naheliegende Erweiterung betrifft die Art der bereitgestellten Inhalte. In der aktuellen Umsetzung werden ausschließlich Bilder verwendet. Perspektivisch könnten jedoch auch Audio- oder Videoinhalte eingebunden werden, um die Informationsvermittlung vielseitiger und anschaulicher zu gestalten.

Einer der wesentlichen genannten Vorteile digitaler Inhalte besteht darin, dass sie nicht an den begrenzten Platz eines physischen Schildes gebunden sind und zielgruppenorientiert bereitgestellt werden können. Besonders sinnvoll wäre daher die Unterstützung von Mehrsprachigkeit. Dadurch könnten sowohl die Benutzeroberfläche als auch die inhaltlichen Texte in weiteren Sprachen angeboten werden.

Darüber hinaus könnte die Anwendung um spielerische Elemente erweitert werden. Denkbar wären beispielsweise Quizze, Punktesysteme oder Auszeichnungen für besuchte Stationen. Solche Funktionen könnten zusätzliche Anreize schaffen, weitere Orte zu erkunden und sich intensiver mit der lokalen Geschichte auseinanderzusetzen.

8.4 Übertragbarkeit auf andere Gemeinden

Die Übertragbarkeit des Projekts auf andere Gemeinden ist grundsätzlich naheliegend. Gerade im touristischen Kontext enden Angebote selten an kommunalen Grenzen. Wanderwege, Fahrradrouten und regionale Ausflugsziele verlaufen häufig über mehrere Gemeindegebiete hinweg. Daher ist es sinnvoll, auch eine digitale Geschichtstour nicht ausschließlich auf eine einzelne Kommune zu beschränken, sondern perspektivisch über Gemeindegrenzen hinaus zu denken.

Für eine solche Erweiterung bestehen grundsätzlich zwei Möglichkeiten. Die erste Variante wäre, für jede weitere Gemeinde eine eigene Anwendung beziehungsweise ein eigenes System bereitzustellen. Der bestehende Quellcode könnte dafür weitgehend übernommen und an die jeweilige Gemeinde angepasst werden, etwa durch andere Texte, Farben, Logos und Inhalte. Auch Server, Datenbank und Admin-Oberfläche ließen sich auf dieser Grundlage erneut aufsetzen. Technisch wäre dieser Ansatz vergleichsweise einfach umsetzbar, er würde jedoch schnell zu mehreren voneinander getrennten Einzelösungen führen.

Die zweite und langfristig sinnvollere Möglichkeit besteht darin, die bestehende Anwendung, um weitere Gemeinden zu erweitern. Neue Stationen und Touren würden dabei nicht in einer eigenen App verwaltet, sondern in die vorhandene Datenstruktur integriert. Dadurch müsste der Quellcode nur an einer Stelle gepflegt und weiterentwickelt werden. Gleichzeitig entstünde für Anwender eine einheitliche Anwendung, die auch bei gemeindeübergreifenden Touren ohne Medienbruch genutzt werden kann.

Gerade dieser Punkt ist für die praktische Nutzung besonders relevant. Wenn ein Wanderweg oder eine Fahrradtour durch mehrere Gemeinden führt, wäre es wenig nutzerfreundlich, wenn für jede Gemeinde eine separate App installiert werden müsste. Eine gemeinsame Anwendung reduziert diese Hürde und ermöglicht eine durchgängige Nutzung über kommunale Grenzen hinweg. Damit würde zugleich vermieden, dass mehrere technisch ähnliche Insellösungen entstehen, die jeweils separat gepflegt, veröffentlicht und betrieben werden müssen.

Voraussetzung für eine solche Erweiterung wäre jedoch eine komplexere Benutzer- und Rollenverwaltung. Sie müsste sicherstellen, dass jede Gemeinde nur ihre eigenen Inhalte bearbeiten kann, ohne Zugriff auf die Daten anderer Gemeinden zu erhalten. Auf organisatorischer Ebene müssten außerdem Personen, Vereine oder Institutionen vorhanden sein, die Inhalte erstellen, prüfen und langfristig pflegen. Die technische Skalierbarkeit allein reicht daher nicht aus, wenn keine verlässliche redaktionelle Betreuung gewährleistet ist.

Die entwickelte Plattform ist grundsätzlich gut auf weitere Gemeinden übertragbar. Besonders geeignet erscheint eine zentrale technische Lösung, die

mehrere Gemeinden innerhalb einer gemeinsamen Anwendung abbildet. Dadurch ließen sich Wartung, Weiterentwicklung und Betrieb bündeln, während die inhaltliche Verantwortung weiterhin lokal organisiert werden könnte.

9 Literaturverzeichnis

- [1] Microsoft. „NET Multi-platform App UI documentation“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/dotnet/maui/?view=net-maui-10.0>, Zugriff am: 20.05.2026.
- [2] Microsoft. „Razor Pages architecture and concepts in ASP.NET Core“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/aspnet/core/razor-pages/?view=aspnetcore-10.0>, Zugriff am: 20.05.2026.
- [3] Microsoft. „Create web APIs with ASP.NET Core“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-10.0>, Zugriff am: 20.05.2026.
- [4] Microsoft. „Overview of Entity Framework Core“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/ef/core/>, Zugriff am: 20.05.2026.
- [5] Microsoft. „Azure App Service documentation“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/azure/app-service/>, Zugriff am: 20.05.2026.
- [6] Microsoft. „Azure SQL Database documentation“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/azure/azure-sql/database/?view=azuresql>, Zugriff am: 20.05.2026.
- [7] Microsoft. „Visual Studio documentation“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/visualstudio/windows/?view=visualstudio>, Zugriff am: 20.05.2026.
- [8] Microsoft. „Documentation for Visual Studio Code“. Visual Studio Code. Verfügbar unter: <https://code.visualstudio.com/docs>, Zugriff am: 20.05.2026.
- [9] Microsoft. „NET SDK overview“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/dotnet/core/sdk>, Zugriff am: 20.05.2026.
- [10] SmartBear Software. „Swagger Documentation“. Swagger. Verfügbar unter: <https://swagger.io/docs/>, Zugriff am: 20.05.2026.
- [11] Git. „Reference“. Git Documentation. Verfügbar unter: <https://git-scm.com/docs>, Zugriff am: 20.05.2026.
- [12] GitHub. „GitHub Docs“. GitHub Documentation. Verfügbar unter: <https://docs.github.com/>, Zugriff am: 20.05.2026.
- [13] Microsoft. „Azure portal documentation“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/azure/azure-portal/>, Zugriff am: 20.05.2026.
- [14] Apple. „App Store Connect Help“. Apple Developer. Verfügbar unter: <https://developer.apple.com/help/app-store-connect/>, Zugriff am: 20.05.2026.
- [15] Google. „Play Console Help“. Google Play Console Help. Verfügbar unter: <https://support.google.com/googleplay/android-developer/?hl=en>, Zugriff am: 20.05.2026.
- [16] Google. „Search Console Help“. Google Search Console Help. Verfügbar unter: <https://support.google.com/webmasters/?hl=en>, Zugriff am: 20.05.2026.
- [17] OpenAI. „Codex“. OpenAI Developers. Verfügbar unter: <https://developers.openai.com/codex>, Zugriff am: 20.05.2026.

[18] Microsoft. „DbContext Lifetime, Configuration, and Initialization“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/ef/core/dbcontext-configuration/>, Zugriff am: 20.05.2026.

[19] Microsoft. „Migrations Overview“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/ef/core/managing-schemas/migrations/>, Zugriff am: 20.05.2026.

[20] Microsoft. „Deployment & testing, .NET MAUI“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/dotnet/maui/deployment/?view=net-maui-10.0>, Zugriff am: 20.05.2026.

[21] Microsoft. „Razor Pages authorization conventions in ASP.NET Core“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/en-us/aspnet/core/razor-pages/security/authorization/conventions?view=aspnetcore-10.0>, Zugriff am: 20.05.2026.

[22] Microsoft. „Model View ViewModel, .NET MAUI“. Microsoft Learn. Verfügbar unter: <https://learn.microsoft.com/de-de/dotnet/architecture/maui/mvvm>, Zugriff am: 20.05.2026.

[23] Google. „Verify Android App Links“. Android Developers. Verfügbar unter: <https://developer.android.com/training/app-links/verify-applinks>, Zugriff am: 20.05.2026.

[24] Apple. „Supporting associated domains“. Apple Developer Documentation. Verfügbar unter: <https://developer.apple.com/documentation/xcode/supporting-associated-domains>, Zugriff am: 20.05.2026.