

Wahlprojekt: Webanwendung für Studieninteressierte

Inhalt

1. Einleitung	3
1.1 Ausgangssituation und Motivation	3
1.2 Zielsetzung des Projekts	3
1.2.1 Abgrenzung.....	4
1.3 Projektüberblick.....	4
2. Projektvorgehen.....	5
2.1 Projektorganisation	5
2.1.1 Kommunikationswege	5
2.2 Vorgehensmodell	6
2.3 Projektphasen und Ablauf	6
2.4 Werkzeuge im Projektmanagement.....	8
2.4.1 Aufgabenplanung	9
2.5 Erfahrung im Projekt	9
2.5.1 Was lief gut?.....	9
2.5.2 Wo gab es Probleme?	9
2.5.3 Was würden wir beim nächsten Projekt anders machen?.....	9
3. Architekturdokumentation.....	10
3.1 Gesamtarchitektur der Anwendung (Blueprint)	10
3.2 Architekturentscheidungen	11
3.3 Verwendete Frameworks, Entwicklungsumgebung & Tools	12
4. Praktische Arbeit – Programm (Kurzüberblick)	13
4.1 Funktionsübersicht der Anwendung	13
4.2 Aufbau des Programmcodes (Überblick).....	14
4.3 Ablauffähigkeit des Programms	14
5. Anleitungen	15
5.1 Technische Administrationsanleitung.....	15
Variante A: Deployment auf einem Webserver (z.B. Apache / Nginx)	15
Variante B: Einbindung in TYPO3	16
5.2 Fachliche Administrationsanleitung	17

5.3 Bedienungsanleitung (Benutzer).....	19
6. Fazit und Ausblick.....	20
6.1 Zielerreichung und Projektergebnis	20
6.2 Ausblick und Erweiterungsmöglichkeiten.....	20

1. Einleitung

1.1 Ausgangssituation und Motivation

Die Entscheidung, die man nach der allgemeinen Hochschulreife oder der Fachhochschulreife treffen muss, stellt viele junge Erwachsene vor eine große Herausforderung, weil es die Grundlage dafür sein wird, wie sie ihr zukünftiges Leben führen. Gerade interdisziplinäre Studiengänge wie die Wirtschaftsinformatik, sind schwer greifbar, weil die Grundlagen die hierfür benötigt wird oft nicht Inhalt der schulischen Ausbildung war. Die Verbindung der Fachrichtungen Mathematik, Informatik und Betriebswirtschaftslehre kommt so für Studieninteressierte nicht rüber, weil klassische Informationsangebote auf Hochschulwebseiten nur aus Texten oder PDF-Dokumenten besteht. Diese stellen zwar fachlich korrekte Inhalte dar aber sind nicht besonders anschaulich noch interaktiv.

Aus diesem Grund besteht der Bedarf moderne, verständliche und interaktive Studienorientierung für die Hochschule zu ermöglichen. Das Modulhandbuch kann sachlich darlegen, welche Inhalte im Studium behandelt werden und wie der Studienverlauf aufgebaut ist jedoch können Worte allein keinen tatsächlichen Einblick verschaffen. Dazu fehlt die Möglichkeit sich selbst einzuschätzen, ob der Studiengang zu den eigenen Interessen und Fähigkeiten passt.

Die Motivation für dieses Projekt entstand genau aus dieser Problematik: Ziel war es, eine Anwendung zu entwickeln, die den Studiengang Wirtschaftsinformatik nicht nur beschreibt, sondern erlebbar macht, genauso wie Fragen zum Studium zu beantworten allgemein sowohl als auch spezifisch für die Hochschule RheinMain. Durch die visuelle Darstellung des Curriculums, interaktive Module und einen Fragebogen zur Studienorientierung soll die Informationslücke zwischen abstrakten Studienbeschreibung und realen Studienalltag geschlossen werden. Im Fokus stand die Entwicklung einer Anwendung, die es ermöglicht, einen übersichtlichen Eindruck des Studiengangs zu erhalten, ohne dabei mit Informationen überladen zu werden. Vor dem Hintergrund des fortschreitenden technologischen Wandels sollte so die Aufmerksamkeit der Nutzer:innen gewahrt und ein Untergehen im stetig wachsenden Informationsangebot vermieden werden.

1.2 Zielsetzung des Projekts

Ziel des Projekts in die Entwicklung einer interaktiven Webanwendung zur Studienorientierung für den Studiengang Wirtschaftsinformatik der Hochschule RheinMain. Die Anwendung soll Studieninteressierten einen übersichtlich, strukturierten und verständlich Einblick in den Studienverlauf ermöglichen.

Konkret soll das System:

- den Studienverlauf semesterweise visualisieren,
- die Module übersichtlich darstellen, inclusive ECTS-Punkten und fachlicher Einordnung,
- detaillierte Modulbeschreibungen und Beispielaufgaben
- interaktiven Fragebogen bereitstellen, der den Nutzenden eine Selbsteinschätzung gibt
- eine moderne, benutzerfreundliche und responsive Oberfläche bieten.

Die Webanwendung richtet sich hauptsächlich an Studieninteressierte, kann aber auch für Studierende zur Orientierung im Studienverlauf verwendet werden.

1.2.1 Abgrenzung

Nicht Bestandteile dieses Projektes sind:

- Benutzerverwaltung oder Login-Funktion,
- Eine vollständige Abbildung aller möglichen Wahlmodule.

Die Anwendung dient zur Orientierung und Informationssammlung.

1.3 Projektüberblick

Im Rahmen des Projekts wurde eine webbasierte, interaktive Studienorientierungsplattform für den Studiengang Wirtschaftsinformatik entwickelt. Die Anwendung ist als Single-Page-Webanwendung umgesetzt, der Fokus liegt auf eine klare Struktur, intuitive Bedienbarkeit und eine ansprechende visuelle Darstellung.

Zentrale Bestandteile der Webanwendung sind:

- Curriculum-Bereich, wo alle Semester von unten nach oben dargestellt sind, wie auf der Hochschulwebseite
- Modulare Darstellungen mit farblicher Kennzeichnung der Fachbereiche,
- Interaktive Modulboxen, die beim Aufruf detaillierte Informationen anzeigen,
- Ein Studienorientierungsfragebogen, um Interessen und Anforderungen des Studiengangs zu erfassen
- Berufs-Finder, stellt Berufe der Wirtschaftsinformatik als Karten da die man nach links und rechts wischen kann,
- FAQ, typische Fragen, in Form von Karten die man horizontal durchgehen kann.

Die Zielgruppe der Anwendung sind vor allem Schülerinnen und Schüler, Studieninteressierte sowie Studienanfängerinnen und -anfänger. Es ist eine reine Webanwendung, die im Browser genutzt werden kann.

2. Projektvorgehen

2.1 Projektorganisation

Das Projekt wurde im Rahmen einer Gruppenarbeit umgesetzt. Die Rollen im Team orientierten sich an agilen Projektstrukturen.

Die Rolle des Scrum Masters übernahm Nikolai Schüler. Er fungierte als zentrale Koordinationsperson innerhalb des Teams und stand während des gesamten Projektes in regelmäßigem Kontakt mit Herr Ricken, um Rückmeldungen einzuholen und organisatorische Fragen zu klären.

Das Entwicklungsteams bestand aus:

- Lisa Halim
- Daniel Navid Adillipour
- Soojin Ham
- Hamidullah Djan
- Sowie auch Nikolai Schüler

Alle Teammitglieder waren an der inhaltlichen und technischen Umsetzung beteiligt. Ein besonderer Schwerpunkt im Frontend-Bereich bei Nikolai Schüler, der am Anfang des Projektes ein grundlegendes Layout entwarf. Alle Teammitglieder wurden damit beauftragt ein Layout zu designen, um sich zu beraten und um zu finalisieren, welches Layout am vorteilhaftesten ist. Nach der Evaluierung der Ergebnisse lag die Entscheidung zwischen Herrn Schülers Entwurf und Herrn Djans Entwurf, final haben wir uns jedoch auf Herrn Schülers Entwurf geeinigt. Das Layout diente als Basis, auf der die weitere Entwicklung der Benutzeroberfläche aufbaute und gemeinsam weiterentwickelt wurde.

Eine strikte Trennung zwischen Frontend- und Dokumentationsrollen wurde bewusst nicht vorgenommen, da es sich um ein überschaubares Projekt handelt und alle Teammitglieder aktiv an mehreren Bereichen beteiligt waren.

2.1.1 Kommunikationswege

Am Anfang des Projektes erfolgte die Kommunikation hauptsächlich über WhatsApp. In der frühen Phase wurden erste Ideen gesammelt, Ziele definiert und gemeinsam überlegt für welche Zielgruppe das Projekt entwickelt wird. Die Ergebnisse der Phase wurden in einem Projektkonzept festgehalten, dass in Form eines Word-Dokument dokumentiert wurde.

Im weiteren Projektverlauf fanden regelmäßigen Meetings über Zoom statt. Diese Meetings dienten dazu:

- den aktuellen Stand der Entwicklung des Projektes zu besprechen,
- Feedback zur Umsetzung zu geben,
- offene Fragen zu klären,
- sowie die nächsten Arbeitsschritte festzulegen.

Die Kombination aus Kommunikation über Text und Meeting ermöglichten einen kontinuierlichen Austausch innerhalb des Teams.

2.2 Vorgehensmodell

Für die Umsetzung des Projektes wurde ein agiles und iteratives Vorgehensmodell gewählt, das sich an den Grundprinzipien von Scrum orientierte aber in vereinfachter Form angewendet wurde.

Diese Vorgehensweise eignete sich besonders gut für das Projekt, weil am Anfang nicht alle Anforderungen abgemacht waren. Entscheidungen zu Design, Benutzeroberfläche und Funktionsumfang, wurden schrittweise getroffen und im Laufe des Projektes besprochen und weiterentwickelt. Durch regelmäßige Abstimmungen und Feedbackrunden konnte auf neue Erkenntnisse flexibel reagiert werden.

Typisch für das agile Vorgehen waren:

- Die interaktive Weiterentwicklung der Anwendung,
- regelmäßige Besprechung des Projektstands,
- kontinuierliche Feedback innerhalb des Teams,
- und Anpassung der Anforderungen während der Umsetzung.

2.3 Projektphasen und Ablauf

Das Projekt gliederte sich in mehrere aufeinander aufbauenden Phasen, die teilweise parallel und iterativ durchgeführt wurden. Die folgende Tabelle dient als Template, das zum Protokollieren des Projektes dient.

Projektphase	Zeitraum	Inhalte	Zuständigkeiten/ Aufgaben
Konzeptionsphase	17.10 - 21.10	<ul style="list-style-type: none">– Definition der Zielgruppe,– Ideensammlung der Projektidee innerhalb der Gruppe	<ul style="list-style-type: none">– gesamtes Team,– Nikolai Schüler zum Scrum Master erwählt
Vertiefung der Konzeptionsidee	21.10 - 04.11	<ul style="list-style-type: none">– Fachliche Anforderungen wie auch mögliche technologische Umsetzungen schriftlich dokumentiert,– Fragen Katalog erstellt,– sowie Gedanken zur grafischen Darstellung	<ul style="list-style-type: none">– Gesamtes Team hat sich Gedanken zu den fachlichen Anforderungen und mögliche technische Umsetzung gemacht,– Fragen Katalog erstellt von Nikolai Schüler,– Gedanken zur grafischen Darstellung von Ham Soojin
Implementierung	04.11- 25.11	<ul style="list-style-type: none">– Typo3 Installation auf den Rechner aller Gruppenmitglieder– technische Möglichkeiten prüfen– erste Version grob erstellen	<ul style="list-style-type: none">– alle Gruppenmitglieder haben Typo3 installiert (manche hatten Schwierigkeiten hiermit)– alle Mitglieder haben eine erste Version gezeigt → Nikolai Schülers Version unter Abstimmung genommen

	22.11	<ul style="list-style-type: none"> – Unterteilung der Fachsemester unter den Gruppenmitgliedern → für die Quizerstellung hauptsächliche Verwendung durch JavaScript – Auf Basis des Entwurfs von Niko → FAQ und ein Video für Semester 6 hinzugefügt – Einbindung Berufsfinder 	<ul style="list-style-type: none"> – Semester 1: Schüler – Semester 2: Adillipour – Semester 3: Soojin – Semester 4: Djan – Semester 5: Halim – Idee für FAQ von Lisa Halim – Technische Umsetzung von Nikolai Schüler – Idee für Video-Semester 6 von Schüler – Alle Teammitglieder haben nach Tools gescoutet – Berufsfinder durch Schüler integriert
	25.11 – 09.12 bzw. auch darüber hinaus	<ul style="list-style-type: none"> – Mobile Ansicht verbessert – Design Verbesserung → Light und Dark Mode → und allgemein – Erarbeitung der Semester – 09.12 Vorstellung der Ergebnisse bis zu dem Zeitpunkt → Semester mit verschiedenen Quizarten und Schwerpunkten – Idee Chatbot einzuführen Herrn Ricken vorgestellt – Selbsteinschätzung Quiz 	<ul style="list-style-type: none"> – Mobile Ansicht durch Schüler sowie Design Verbesserung – weiterhin durch die Verantwortlichen – Idee für den Chatbot durch Djan – Selbsteinschätzungsquiz von Soojin erstellt
	12.12 – 23.12 bzw. darüber hinaus	<ul style="list-style-type: none"> – Modulbeschreibungen erstellt zu den jeweiligen Semestern – Erstellen des Chatbots – Winter-/Weihnachtspause 	<ul style="list-style-type: none"> – Modulbeschreibungen durch Adillipour und Halim → Semester aufgeteilt – Implementierung des Chatbots durch Djan
	24.12 – 01.01	<ul style="list-style-type: none"> – Chatbot in Code integriert – Fragen und Antworten ergänzt 	
	01.01 – 06.01	<ul style="list-style-type: none"> – Fehlerquellen beseitigen und Verbesserungen durchführen – HSRM Informationen 	<ul style="list-style-type: none"> – Chatbot Integration durch Djan und Schüler – Fragen und Antworten ergänzt durch Halim → Integration in den Code durch Schüler
	06.01 - 26.01		

			<ul style="list-style-type: none"> - Erweiterung und Verbesserungen bestehender Inhalte
Test & Optimierung	04.11 – 26.01	<ul style="list-style-type: none"> – Funktionstests der Anwendung – Überprüfung der Benutzerführung – Layout und Fehlerbehebung →fortlaufend	<ul style="list-style-type: none"> – Anpassungen des Layouts durch Schüler – Behebungen der Fehler Zuständigkeit aller Teammitglieder sowie Funktionstests der Anwendung
Dokumentation	12.02 -26.01	<ul style="list-style-type: none"> – Erstellung der Projektdokumentation 	<ul style="list-style-type: none"> – Unterteilung innerhalb der Mitglieder: Halim: Einleitung, Projektvorgehen, Selbsteinschätzung Schüler: Architekturdokumentation, Anleitung und Selbsteinschätzung Djan: Praktische Arbeit und Selbsteinschätzung Adillipour: Fazit, Ausblick und Selbsteinschätzung Soojin: Selbsteinschätzung
Abschluss & Reflexion	06.01 – 26.01	<ul style="list-style-type: none"> – Projektabschluss, Refllexion des Projektverlaufs, Bewertung der Zusammenarbeit 	Alle Gruppenmitglieder in Form eines Zoom-Meetings

Anmerkung: Hamidullah Djan ist anschließend, zum Termin des dritten Sprints, in das Projektteam eingeteilt worden, also zwischen dem 21.10 und 04.11.

2.4 Werkzeuge im Projektmanagement

Für die Versionsverwaltung des Projekts wurde GitLab verwendet. Der aktuelle Projektstand wurde dort regelmäßig hochgeladen, sodass jederzeit eine zentrale und einheitliche Version des Quellcodes zur Verfügung stand.

Die Teammitglieder arbeiteten lokal an ihren jeweiligen Systemen und aktualisierten ihre Änderungen durch das Hochladen der neuesten Versionen in das gemeinsame Repository. Dadurch wurde sichergestellt, dass alle Beteiligten mit einem konsistenten Entwicklungsstand arbeiten konnten.

Die Versionsverwaltung diente somit vor allem der Sicherung des Projektfortschritts, der Nachvollziehbarkeit von Änderungen sowie dem Austausch des aktuellen Projektstands innerhalb des Teams.

2.4.1 Aufgabenplanung

Die Aufgabenplanung erfolgte meist mündlich in den regelmäßigen Zoom-Meetings. Hier wurde besprochen:

- welche Aufgaben bereits abgeschlossen wurden,
- welche Punkte aktuell offen sind,
- und welche Aufgaben als nächstes umgesetzt werden sollen

Die Aufgaben wurden situativ im Team verteilt, je nachdem in welcher Projektphase wir uns befanden. Diese wurden abhängig von den individuellen Stärken und den aktuellen Arbeitsaufwand der einzelnen Teammitglieder verteilt. Die flexible Vorgehensweise konnte schnell auf neue Anforderungen oder notwendige Anpassungen reagieren.

Eine formale Aufgabenverwaltung über Boards wurde nicht dokumentiert. Stattdessen lag der Fokus auf direkter Abstimmung und flexibler Aufgabenverteilung. Als praktisch erwiesen hat sich die iterative Entwicklung und die kurzen Kommunikationswege.

Ergänzend wurden wichtige Aufgaben und Ideen teilweise in schriftlicher Form festgehalten, um den Überblick über den Projektfortschritt zu bewahren.

2.5 Erfahrung im Projekt

2.5.1 Was lief gut?

Besonders positiv hervorzuheben ist die Kommunikation über die Zoom-Meetings. Diese ermöglichten einen direkten Austausch, konstruktives Feedback und gegenseitige Unterstützung. Ideen konnten gemeinsam diskutiert und verbessert werden. Was zu einer kontinuierlichen Qualitätssteigerung geführt hat.

Auch die iterative Arbeitsweise hat sich bewährt, weil Anpassungen im Design und an der Funktionsweise problemlos vorgenommen werden konnten.

2.5.2 Wo gab es Probleme?

Die Kommunikation über WhatsApp verlief ab und an schleppend, weil Nachrichten nicht immer zeitnah beantwortet wurden, jedoch wurden diese entschuldigt und mit Verständnis angenommen. Dennoch hat diese gelegentliche Verzögerung keinen Nachteil dargestellt.

2.5.3 Was würden wir beim nächsten Projekt anders machen?

Für zukünftige Projekte wäre es sinnvoll,

- die Aufgabeplanung stärker strukturieren
- wichtige Entscheidungen schriftlich festhalten.

Insgesamt war das Projekt jedoch eine wertvolle Erfahrung im Bereich Teamarbeit, Kommunikation und agiler Softwareentwicklung.

3. Architekturdokumentation

3.1 Gesamtarchitektur der Anwendung (Blueprint)

Die entwickelte Anwendung ist als clientseitige Webanwendung konzipiert und folgt dem Prinzip einer Single-Page-Anwendung (SPA). Alle Funktionalitäten werden vollständig im Browser ausgeführt; eine serverseitige Verarbeitung oder Datenbankanbindung ist nicht erforderlich.

Die Architektur der Anwendung basiert auf einer klaren Schichtung und Trennung der Verantwortlichkeiten. Ziel dieser Struktur ist es, eine gute Wartbarkeit, Erweiterbarkeit sowie eine einfache Pflege der Inhalte zu gewährleisten.

Die Anwendung besteht aus den folgenden Hauptkomponenten:

- **HTML (Struktur)**
Das HTML-Dokument stellt das statische Grundgerüst der Anwendung dar. Es definiert die Seitenstruktur, Container, Platzhalter sowie semantische Bereiche (z. B. Hauptinhalt, Sidebar, Modals). Interaktive Inhalte werden nicht direkt im HTML gepflegt, sondern zur Laufzeit durch JavaScript eingebunden.
- **CSS (Darstellung & Layout)**
Das Styling der Anwendung ist modular aufgebaut. Es erfolgt eine Trennung zwischen:
 - Basis-Styles (Design-Tokens, Reset, Typografie)
 - Layout-Styles (Seitenstruktur, Responsive Design)
 - Komponenten-Styles (Tabs, Filter, Modals, Quiz, Sidebar usw.)Durch diese Aufteilung können Designanpassungen gezielt vorgenommen werden, ohne andere Bereiche der Anwendung zu beeinflussen.
- **JavaScript (Logik & Steuerung)**
Die komplette Anwendungslogik ist in JavaScript-Modulen umgesetzt. Ein zentraler Einstiegspunkt initialisiert die Anwendung, bindet Event-Listener ein und koordiniert die einzelnen Funktionsmodule (z. B. Tabs, Filter, Modals, Quiz, Career-Finder).
- **Content-Dateien (Inhalte)**
Sämtliche redaktionellen Inhalte (z. B. FAQ-Texte, Infotexte, Quiz-Fragen, Sidebar Inhalte) sind in separaten JavaScript-Dateien ausgelagert. Diese Dateien enthalten ausschließlich Datenstrukturen und keine Logik.

Der Datenfluss innerhalb der Anwendung erfolgt ausschließlich vom Content über die JavaScript-Renderer hin zur Darstellung im DOM.

3.2 Architekturentscheidungen

Ein zentrales Architekturziel des Projekts war die klare Trennung von Inhalt, Darstellung und Logik. Diese Trennung wurde bewusst umgesetzt, um sowohl die technische Wartbarkeit als auch die redaktionelle Pflege der Inhalte zu erleichtern.

Modulare Java-Script-Struktur

Die Anwendung verwendet eine modulare JavaScript-Struktur, bei der jede Funktionalität in einer eigenen Datei gekapselt ist. Beispiele hierfür sind Module für:

- Tab-Navigation
- Filterlogik
- Modals
- Quiz-Funktionalität
- Rendering von Inhalten

Diese Struktur ermöglicht:

- bessere Übersichtlichkeit des Codes
- gezielte Erweiterungen einzelner Funktionen
- einfachere Fehlersuche und Wartung

Trennung von Logik, Darstellung und Content

- Darstellung erfolgt ausschließlich über HTML und CSS
- Logik ist in JavaScript-Modulen gekapselt
- Inhalte werden über separate Content-JavaScript-Dateien eingebunden

Durch diese Trennung können Inhalte angepasst oder erweitert werden, ohne dass die Anwendungslogik verändert werden muss. Dies ist insbesondere für nicht-technische oder fachliche Administratoren von Vorteil.

Verzicht auf Frameworks

Die Anwendung wurde bewusst ohne den Einsatz von JavaScript-Frameworks (z. B. React oder Vue) umgesetzt. Stattdessen kommt Vanilla JavaScript zum Einsatz.

Die Gründe hierfür sind:

- geringere Komplexität
- keine zusätzlichen Abhängigkeiten
- vollständige Kontrolle über die Anwendung
- bessere Nachvollziehbarkeit für Lernzwecke

Dadurch bleibt die Architektur überschaubar und die Funktionsweise der Anwendung ist vollständig nachvollziehbar.

3.3 Verwendete Frameworks, Entwicklungsumgebung & Tools

Für die Umsetzung des Projekts wurden folgende Technologien und Werkzeuge eingesetzt:

- **Programmiersprachen:**
HTML5, CSS3, JavaScript (ES6+)
- **Entwicklungsumgebung:**
Hauptsächlich wurde IntelliJ IDEA als Entwicklungsumgebung mit integrierter Dateiverwaltung und Debugging-Funktionen genutzt, insbesondere zur Arbeit an der JavaScript-Struktur und zur Organisation des Projekts. Ergänzend kam Visual Studio Code zum Einsatz, vor allem für schnelle Anpassungen an HTML- und CSS-Dateien sowie für kleinere Änderungen und Tests.
- **Content-Management / Testumgebung:**
TYPO3 wurde im Projektkontext verwendet, um Design- und Layoutentscheidungen zu überprüfen sowie die Einbindung der Anwendung in eine typische Hochschul-Webumgebung zu evaluieren.
- **Infrastruktur- und Entwicklungstools:**
Docker wurde zur lokalen Entwicklungs- und Testumgebung eingesetzt. n8n kam zur Umsetzung und Erprobung automatisierter Workflows zum Einsatz. Supabase wurde als Backend-Service genutzt, beispielsweise zur Datenhaltung bzw. für projektbezogene Backend-Funktionalitäten.
- **Versionsverwaltung:**
Git in Verbindung mit GitLab zur gemeinsamen Entwicklung, Versionskontrolle und Aufgabenverwaltung sowie die Nachvollziehbarkeit von Änderungen.
- **Weitere Tools:**
Browser-Entwicklertools zur Analyse von Layout, Performance und zur Identifikation von Fehlern während der Entwicklung.

4. Praktische Arbeit – Programm (Kurzüberblick)

4.1 Funktionsübersicht der Anwendung

Bei dem vorliegenden Projekt handelt es sich um eine interaktive Webanwendung zur Darstellung und Exploration des Studiengangs „Wirtschaftsinformatik (B.Sc.)“. Ziel der Anwendung ist es, Studieninteressierten und Studierenden eine intuitive Plattform zu bieten, um Modulinhalte, Berufsfelder und die persönliche Eignung spielerisch zu erkunden.

Hauptfunktionen:

- **Interaktives Curriculum-Dashboard:**
Eine visuelle Aufbereitung des Studienverlaufs über sieben Semester. Nutzer können die zeitliche Abfolge der Module sowie die Verteilung der Credit Points (CP) auf einen Blick erfassen.
- **Dynamische Modul-Details:**
Durch Interaktion mit einzelnen Modul-Karten (z.B. "Diskrete Strukturen") werden detaillierte Informationen zu Inhalten, Vorkenntnissen, Prüfungsformen und dem Workload ausgegeben.
- **Beruf-Finder (Matching-Tool):**
Ein zentrales Element der Anwendung, das es Nutzern ermöglicht, durch verschiedene Berufe zu navigieren und eine Auswahl basierend auf persönlichen Interessen zu treffen.

Besondere Features:

- **Filter-System:**
Die Anwendung verfügt über eine leistungsfähige Filter-Logik. Nutzer können das Curriculum nach Fachbereichen (z.B. BWL, Informatik, Mathematik, eGov) sortieren, wobei sich die Ansicht in Echtzeit anpasst.
- **Interaktives Mini-Quiz:**
Zur Steigerung der User-Experience wurde ein Quiz Modul integriert, das durch gezielte Fragen (z.B. Affinität zu Mathe oder IT) eine erste Rückmeldung zur Eignung für das Studium gibt.
- **Modale Dialogfenster:**
Komplexe Informationen werden nicht auf neuen Seiten, sondern in Modals (Pop-up-Fenstern) angezeigt. Dies bewahrt den Kontext der Hauptseite und verbessert die Usability.
- **FAQ-Flip-Cards:**
Für häufig gestellte Fragen wurde ein visuelles Karten System implementiert, das bei Interaktion (Hover/Tap) Antworten preisgibt.

4.2 Aufbau des Programmcodes (Überblick)

Die Anwendung wurde als moderne Single-Page-Application (SPA) auf Basis von HTML, CSS und JavaScript realisiert. Bewusst wurde auf schwere Frameworks verzichtet, um eine maximale Ladegeschwindigkeit und sauberen Code zu gewährleisten.

Ordnerstruktur:

Die Projektstruktur folgt einer klaren Trennung von Ressourcen und Logik:

- `index.html`: Die zentrale Einstiegsdatei, die das Grundgerüst der Seite definiert.
- `/css`: Enthält die Stylesheets (z.B. `style.css`), in denen das responsive Design und die Layout-Vorgaben (Flexbox/Grid) definiert sind.
- `/js`: Beinhaltet die gesamte Programmlogik, aufgeteilt in Funktionsmodule.

Trennung der Komponenten:

- **Core-Logik:**
Die JavaScript-Dateien steuern die Zustandsverwaltung der App (z.B. welcher Filter aktiv ist, welcher Score im Quiz erreicht wurde).
- **Renderer:**
Die DOM-Manipulation wird über JavaScript gesteuert. Inhalte werden dynamisch in die vorgesehenen HTML-Container geladen, ohne dass die Seite neu geladen werden muss (z.B. beim Öffnen der Modals).
- **Content-Dateien:**
Die Informationen zu den Modulen und Berufen sind innerhalb der Skripte als strukturierte Datenobjekte (JSON-ähnliche Arrays) hinterlegt. Dies ermöglichten eine einfache Wartbarkeit und Erweiterung des Contents, ohne die HTML-Struktur angreifen zu müssen.

4.3 Ablauffähigkeit des Programms

- **Lokal lauffähig:**
Die Anwendung ist vollständig lokal lauffähig. Da es sich um eine statische Web-Applikation handelt, kann sie direkt durch das Öffnen der `index.html` im Browser gestartet werden.
- **Voraussetzungen:**
Für die Nutzung ist lediglich ein moderner Webbrowser (z. B. Chrome, Firefox, Edge oder Safari) erforderlich. Es ist keine Installation von zusätzlicher Software, Plugins oder Frameworks auf dem Endgerät notwendig, da die gesamte Logik clientseitig über Standard-Webtechnologien ausgeführt wird.

5. Anleitungen

5.1 Technische Administrationsanleitung

Zielgruppe: technisch versierte Personen (z.B. Entwickler, Administratoren)

Voraussetzungen:

Erforderliche Komponenten:

- Moderner Webbrowser (z. B. Chrome, Firefox, Edge)
- Webserver oder Hosting-Umgebung zur Bereitstellung statischer Dateien (z. B. Apache, Nginx oder TYPO3-Hosting)

Installation / Deployment

Die Anwendung besteht ausschließlich aus statischen Dateien (HTML, CSS und JavaScript) und benötigt keine serverseitige Laufzeitumgebung. Dadurch kann sie flexibel auf unterschiedlichen Hosting- und Serverumgebungen betrieben werden.

Variante A: Deployment auf einem Webserver (z.B. Apache / Nginx)

Voraussetzungen

- Zugriff auf einen Webserver (z. B. Apache oder Nginx)
- Zugriff auf das Webserver-Dateisystem (z. B. per FTP, SCP oder Hosting-Panel)
- Konfigurierter Document-Root (z. B. /var/www/html oder ein Hosting-Webverzeichnis)

Ordnerstruktur

```
/wahlprojekt/
├─ wahlprojekt_2.html      (Einstiegspunkt)
├─ css/                   (Darstellung)
│   └─ assets/
│       └─ components/
├─ js/                   (Anwendungslogik)
├─ audio/                (Audiodateien)
├─ video/                (Videodateien)
├─ ModuleBeschreibung/   (Modulbezogene Inhalte)
├─ Content/              (Fachliche Inhalte)
└─ AllgemeinContent/     (Allgemeine Inhalte)
```

Empfohlene Verzeichnisstruktur auf dem Webserver (die Vollständige Ordnerstruktur ist im Anhang)

Alle Dateien müssen vollständig und unverändert auf den Webserver übertragen werden.

Deployment-Schritte

1. Quellcode aus dem GitLab-Repository klonen oder herunterladen
2. Projektordner in das gewünschte Webverzeichnis des Servers kopieren
3. Sicherstellen, dass die Datei wahlprojekt_2.html öffentlich erreichbar ist
4. Anwendung über folgende URL aufrufen, z. B.:

https://example-domain.de/wahlprojekt/wahlprojekt_2.html

Sofern gewünscht, kann die Datei auch in index.html umbenannt werden, um den Zugriff

Ohne Dateinamen zu ermöglichen:

<https://example-domain.de/wahlprojekt/>

Hinweis zur Startdatei und Pfadstruktur

Die Einstiegspunkt-Datei der Anwendung ist wahlprojekt_2.html. Diese lädt die komplette Anwendungslogik über js/main.js als ES-Modul und initialisiert sämtliche UI-Funktionen. Die CSS-Dateien werden teilweise über absolute Pfade (z.B. /css/assets/app.css) eingebunden. Daher ist beim Deployment darauf zu achten, dass:

- entweder das Projekt im Webroot des Servers liegt oder
- der Webserver so konfiguriert ist, dass der Pfad /css/ auf das Projektverzeichnis verweist.

Alternativ können die CSS-Pfade auf relative Pfade angepasst werden, falls die Anwendung dauerhaft in einem Unterverzeichnis betrieben werden soll.

Diese Vorgehensweise ermöglicht eine klare Trennung der Stylesheets und vereinfacht die Wartung der CSS-Struktur.

Hinweise zu Apache / Nginx

- Es sind keine zusätzlichen Servermodule oder Rewrite-Regeln erforderlich
- Die Auslieferung erfolgt vollständig über statische Dateien
- Caching-Mechanismen des Webserverns können optional genutzt werden

Variante B: Einbindung in TYPO3

Die Anwendung kann alternativ in eine bestehende TYPO3-Instanz integriert werden, z. B. im Rahmen einer Hochschulwebseite.

Vorgehensweise

1. Anlegen eines Verzeichnisses innerhalb der TYPO3-File-Struktur (z. B. fileadmin/wahlprojekt/)
2. Hochladen aller Projektdateien in dieses Verzeichnis
3. Erstellen einer neuen TYPO3-Seite im Seitenbaum
4. Einbindung der Datei wahlprojekt_2.html über: ein statisches Seitenmodul oder ein iframe-Element im Content-Bereich

Zugriff über TYPO3

Die Anwendung ist anschließend über die TYPO3-Seiten-URL erreichbar, z. B.:
<https://hsrc.de/studium/wahlprojekt>

TYPO3 übernimmt dabei:

- Navigation
- Layout-Rahmen
- ggf. Rechte- und Seitenverwaltung

Die Anwendung selbst bleibt technisch unabhängig von TYPO3.

Wartung & Pflege

Da keine serverseitige Logik vorhanden ist, beschränkt sich die Wartung auf:

- Überprüfung der Browserkompatibilität
- Kontrolle auf JavaScript-Fehler (Browser-Konsole)
- Testen der Kernfunktionen nach Änderungen

Update des Codes

Code-Updates erfolgen über die Versionsverwaltung mit Git und GitLab.

Empfohlenes Vorgehen:

- Änderungen lokal entwickeln und testen
- Code im Repository versionieren
- Aktualisierte Dateien auf den Webserver übertragen
- Funktionstest im Browser durchführen

Durch die modulare Struktur können einzelne Komponenten aktualisiert werden, ohne die gesamte Anwendung zu verändern.

5.2 Fachliche Administrationsanleitung

Zielgruppe: fachliche Administratoren ohne tiefgehende Programmierkenntnisse

Pflege von Inhalten

Die Inhalte der Anwendung (z. B. Modulbeschreibungen, FAQs, allgemeine Informationen) sind in separaten Content-JavaScript-Dateien abgelegt.

Diese Dateien enthalten ausschließlich strukturierte Textinhalte und können ohne Anpassung der Anwendungslogik geändert werden.

Zuordnung von Inhalten zu Content-Dateien

Die fachlichen Inhalte der Anwendung sind thematisch auf mehrere Content-JavaScript Dateien verteilt. Die folgende Übersicht zeigt, welche Inhalte in welcher Datei gepflegt werden:

- FAQ-Inhalte:

js/AllgemeinContent/faqContent.js

enthält Fragen und Antworten zum Studium und zur Anwendung.

- Sidebar („Auf einen Blick“):
js/AllgemeinContent/sidebarContent.js
enthält kompakte Informationen zum Studiengang (z. B. Abschluss, Regelstudienzeit).
- Allgemeine Informationen:
js/AllgemeinContent/infoContent.js
enthält ausführlichere Informationstexte, die im Bereich „Allgemeine Infos“ angezeigt werden.
- Weitere Inhalte: Inhalte wie Quiz-Fragen sind jeweils in eigenen Content-Dateien abgelegt und folgen demselben strukturellen Aufbau.

Diese klare Zuordnung erleichtert es fachlichen Administrator:innen, Inhalte gezielt zu finden und zu bearbeiten, ohne Kenntnisse der Anwendungslogik zu benötigen.

Anpassung von Texten

Zur Anpassung von Texten sind folgende Schritte notwendig:

1. Öffnen der entsprechenden Content-Datei im Code-Editor
2. Anpassen der Texte oder Beschreibungen
3. Datei speichern und Anwendung neu laden

Die Änderungen werden unmittelbar in der Benutzeroberfläche angezeigt.

Erweiterung von Modulen, FAQs oder Karriere-Inhalten

Neue Inhalte können durch das Ergänzen weiterer Einträge in den jeweiligen Content-Dateien hinzugefügt werden.

Beispiele:

- Hinzufügen eines neuen FAQ-Eintrags
- Ergänzen weiterer Karriereprofile
- Erweiterung von Modulbeschreibungen

Die bestehende Struktur erlaubt es, neue Inhalte konsistent einzubinden, ohne Änderungen am bestehenden Layout oder an der Logik vorzunehmen.

5.3 Bedienungsanleitung (Benutzer)

Zielgruppe: Endnutzer (Schülerinnen und Schüler, Studieninteressierte)

Start der Anwendung

Die Anwendung wird über einen Webbrowser aufgerufen. Nach dem Laden erscheint die Hauptansicht mit dem interaktiven Curriculum des Studiengangs Wirtschaftsinformatik.

Navigation

Die Bedienung der Anwendung erfolgt intuitiv über folgende Elemente:

- Tabs im oberen Bereich zur Auswahl der Hauptfunktionen (Curriculum, Mini-Quiz, FAQ, Beruf-Finder, Allgemeine Infos)
- Scrollen durch die semesterweise dargestellte Modulübersicht
- Hovern über ein Modul öffnet eine Kurzübersicht
- Über den Button „Ausklappen“ kann eine detaillierte Modulbeschreibung angezeigt werden
- Durch Klick auf ein Modul kann das zugehörige Modul-Quiz gestartet werden
- Über den Button in Form einer Sprechblase (unten rechts) lässt sich der Chatbot öffnen
- Ein Button neben der Tab-Leiste ermöglicht den Wechsel zwischen Light- und Dark Mode
- Über die Filterleiste können Module nach Fachbereichen ein- oder ausgeblendet werden

Nutzung zentraler Funktionen

Curriculum: Studienverlauf wird semesterweise dargestellt. Module können angeklickt werden, um zusätzliche Informationen zu erhalten.

Filter: Über Filter lassen sich Module nach Fachbereichen ein- oder ausblenden.

Quiz: Interaktive Quiz-Funktionen ermöglichen eine Selbsteinschätzung der Interessen und Anforderungen des Studiengangs.

Beruf-Finder: Nutzende können sich durch verschiedene Berufsbilder der Wirtschaftsinformatik klicken und individuelle Empfehlungen erhalten.

FAQ: Häufige Fragen zum Studium werden übersichtlich dargestellt und können einzeln aufgerufen werden.

Typische Anwendungsfälle

- Studieninteressierte informieren sich über Inhalte und Aufbau des Studiengangs
- Schülerinnen und Schüler prüfen, ob ihre Interessen zum Studiengang passen
- Studienanfängerinnen und -anfänger verschaffen sich einen Überblick über den Studienverlauf

6. Fazit und Ausblick

6.1 Zielerreichung und Projektergebnis

Das Ziel des Projekts war die Entwicklung einer Webanwendung, die Studieninteressierten auf kreative und interaktive Weise vermittelt, welche Inhalte und Themen im Studiengang Wirtschaftsinformatik behandelt werden. Die Anwendung soll es ermöglichen, sich anhand von Inhalten, Berufsperspektiven, Beispielen sowie optionalen Selbsttests oder Simulationen ein fundiertes Bild vom Studiengang zu verschaffen.

Um dieses Ziel zu erreichen, wurde ein interaktives Curriculum für den Studiengang Wirtschaftsinformatik entwickelt. Durch das Anklicken eines Moduls erhalten Nutzerinnen und Nutzer eine Kurzbeschreibung; zusätzlich werden Themengebiete, Schwierigkeitsgrad sowie Quizfragen zum jeweiligen Modul angezeigt. Dadurch wird eine möglichst realistische Einschätzung der Modulinhalte und damit des gesamten Studiengangs unterstützt.

Darüber hinaus enthält die Webanwendung allgemeine Informationen zur Hochschule und zum Studiengang. Dabei wurde bewusst der Fokus auf die für Studieninteressierte relevantesten Inhalte gelegt. Oberhalb des Curriculums stehen außerdem ein FAQ mit häufig gestellten Fragen zum Studium, ein Mini-Quiz sowie ein Berufsfinder zur Verfügung, der einen Überblick über mögliche Berufsperspektiven nach dem Studium bietet.

Im Verlauf der Projektentwicklung entstand zusätzlich die Idee, einen Chatbot in die Anwendung zu integrieren, sodass Nutzerinnen und Nutzer bei offenen Fragen oder Unklarheiten direkt Unterstützung erhalten können.

Abschließend lässt sich festhalten, dass die im Projekt definierte Zielsetzung vollständig erreicht wurde. Es ist eine funktionsfähige Webanwendung entstanden, die Studieninteressierten einen anschaulichen, kreativen und zugleich strukturierten Einblick in den Studiengang Wirtschaftsinformatik bietet. Das interaktive Curriculum ermöglicht eine intuitive Erkundung der Studieninhalte: Durch die Auswahl einzelner Module werden kompakte Informationen zu Themengebieten, Schwierigkeitsgrad sowie ergänzende Quizfragen bereitgestellt, wodurch der Charakter und Anspruch der Module nachvollziehbar vermittelt wird. Die zusätzlichen Bestandteile der Anwendung – insbesondere FAQ, Mini-Quiz, Berufsfinder sowie die übersichtliche Darstellung der wichtigsten Rahmendaten („Auf einen Blick“) – runden das Gesamtbild ab und unterstützen die Nutzerinnen und Nutzer dabei, zentrale Fragen schnell zu klären, Interessen einzuordnen und mögliche Perspektiven nach dem Studium besser zu verstehen. Insgesamt verbindet die Anwendung Informationsvermittlung mit Interaktivität und bietet damit einen Mehrwert gegenüber klassischen, rein textbasierten Darstellungen.

6.2 Ausblick und Erweiterungsmöglichkeiten

Neben dem erreichten Funktionsumfang ergeben sich für zukünftige Versionen sinnvolle Erweiterungsmöglichkeiten, die jedoch nicht als notwendige Korrekturen, sondern als zusätzliche Ausbaustufen zu verstehen sind. So könnten die Modulansichten optional um vertiefende Detailinformationen ergänzt werden, beispielsweise zu typischen Prüfungsformen, konkreten Lernzielen oder exemplarischen Aufgabenstellungen, um die Praxisnähe noch stärker hervorzuheben.

Ebenso wäre denkbar, den Berufsfinder um weitere Filteroptionen (z. B. Interessen, bevorzugte Tätigkeiten oder gewünschter Schwerpunkt) sowie um ausführlichere Berufsprofile zu erweitern. Auf diese Weise ließe sich die Orientierung noch individueller gestalten, ohne den klaren und übersichtlichen Charakter der Anwendung zu verlieren.

Darüber hinaus bieten sich ergänzende Komfortfunktionen an, etwa eine Suchfunktion, eine Merkliste für interessante Module oder die Möglichkeit, Inhalte stärker zu personalisieren (z. B. durch Empfehlung passender Module auf Basis von Quiz-Ergebnissen). Auch eine Weiterentwicklung im Sinne einer noch breiteren Zugänglichkeit – beispielsweise durch zusätzliche Optimierungen für mobile Endgeräte, erweiterte Barrierefreiheit oder eine mehrsprachige Ausführung – wäre perspektivisch sinnvoll, um den potenziellen Nutzerkreis zu vergrößern. Insgesamt stellt die Anwendung bereits in der aktuellen Version ein überzeugendes, in sich abgeschlossenes Ergebnis dar, das die Studienorientierung interaktiv unterstützt und gleichzeitig eine solide Basis für optionale Erweiterungen in zukünftigen Iterationen bietet.