ClockWise

Project Oriented Time Management Application

Software-Entwicklungspraktikum (SEP)
Sommersemester 2025

Angebot

Auftraggeber
Technische Universität Braunschweig
Peter L. Reichertz Institute for Medical Informatics
Prof. Dr. Thomas M. Deserno
Mühlenpfordtstraße 23
38106 Braunschweig

Betreuer: Corinna Thoben

Auftragnehmer:

Name	E-Mail-Adresse
Joud Mawad	j.mawad@tu-braunschweig.de
Laden Zeynep Erkenci	l.erkenci@tu-braunschweig.de
Sophie Gebauer	sophie.gebauer@tu-braunschweig.de
Luis Jair Gutierrez Pacheco	l.gutierrez-pacheco@tu-braunschweig.de
Chantal Ebben	c.ebben@tu-braunschweig.de
Merle Lüer	m.lueer@tu-braunschweig.de
Moetez Belleh Hosni	m.hosni@tu-braunschweig.de

Braunschweig, 23. April 2025

Bearbeiterübersicht

Kapitel	Autoren	Kommentare
1	Chantal Ebben	...
1.1	Chantal Ebben	...
1.2	Chantal Ebben	...
2	Joud Mawad	...
3	Laden Zeynep Erkenci	...
3.1	Laden Zeynep Erkenci	...
3.2	Laden Zeynep Erkenci	...
4	Moetez Belleh Hosni	...
4.1	Moetez Belleh Hosni	...
4.2	Moetez Belleh Hosni	...
4.3	Moetez Belleh Hosni	...
5	Luis Jair Gutierrez Pacheco, Merle Lüer	...
5.1	Luis Jair Gutierrez Pacheco, Merle Lüer	...
5.2	Luis Jair Gutierrez Pacheco, Merle Lüer	...
5.3	Luis Jair Gutierrez Pacheco, Merle Lüer	...
6	Sophie Gebauer	...
6.1	Sophie Gebauer	...
6.2	Sophie Gebauer	...
6.3	Sophie Gebauer	...
7	Alle Teammitglieder	...

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Sowohl im Berufsleben als auch im Studium und im privaten Alltag nimmt in einer zunehmend beschleunigten Welt die Bedeutung von Produktivität und Effizienz stetig zu. Zeit ist zu einer der wertvollsten Ressourcen geworden. Doch vielen fällt es schwer, bei einer Vielzahl an Projekten, Aufgaben und Verpflichtungen den Überblick zu behalten: Wie viel Zeit fließt eigentlich in welche Tätigkeit? Welche Projekte beanspruchen überdurchschnittlich viel Aufmerksamkeit – und auf wessen Kosten?

Eine systematische Erfassung dieser Zeitdaten kann einer Spirale aus Stress, Ineffizienz und Überforderung frühzeitig entgegenwirken. So lassen sich Folgen wie versäumte Abgaben oder übermäßiger Leistungsdruck vermeiden. Effektives Zeitmanagement ist daher kein Luxus, sondern eine grundlegende Voraussetzung für nachhaltige Produktivität und mentale Gesundheit.

Mit ClockWise möchten wir eine digitale Lösung bieten, die es Nutzer:innen ermöglicht, ihre Zeit strukturiert und projektorientiert zu erfassen, zu analysieren und effektiv zu nutzen. Dabei stehen Übersichtlichkeit, Nutzerfreundlichkeit und Alltagstauglichkeit im Fokus – denn gute Zeitplanung soll entlasten, nicht zusätzlich belasten.

1.1 Ziel

Ziel des Projektes ist die Entwicklung einer intuitiv bedienbaren Webanwendung zur projektorientierten Erfassung, Auswertung und Planung von Zeit. Die Anwendung soll es sowohl Einzelpersonen als auch Teams ermöglichen, Arbeitszeiten zu dokumentieren und strukturiert in Diagrammen sowie in einer Kalenderansicht zu visualisieren.

Die Software bietet eine Möglichkeit zur laufenden Zeiterfassung, zur manuellen Ergänzung und zur Korrektur erfasster Zeiten. Das Speichern der Daten erfolgt in einer Datenbank, sodass ein effizientes Speichern, Abfragen und Analysieren gewährleistet wird. Nutzer:innen können ihre erfassten Zeiten nach verschiedenen Kriterien filtern und bei Bedarf in gängigen Formaten (z.B. PDF, CSV) exportieren.

Ergänzend dazu bietet ClockWise die Möglichkeit, Zeitlimits für Projekte festzulegen. Auf Basis der tatsächlich erfassten Zeiten wird ein Fortschrittsbalken erstellt, der den aktuellen Projektfortschritt anzeigt. Auf übermäßigen Zeitaufwand werden Nutzer:innen durch eine Benachrichtigung per E-Mail hingewiesen.

ClockWise schafft somit Transparenz darüber, wer wie viel Zeit in welche Aufgaben investiert hat. Dies ermöglicht eine faire und effiziente Aufgabenverteilung, erleichtert die Projektkoordination und unterstützt eine optimale Nutzung der zur Verfügung stehenden Zeit.

1.2 Motivation

Die Entwicklung von ClockWise erfolgt im Rahmen des Softwareentwicklungspraktikums und gibt den Auftragnehmern die Möglichkeit, Kenntnisse in der Softwareentwicklung praxisnah zu vertiefen. Unser Anspruch geht dabei jedoch über die reine Umsetzung technischer Funktionen hinaus: Wir möchten eine Anwendung entwickeln, die im Alltag echten Mehrwert bietet.

Vergleichbare Anwendungen wie Clockify oder Jira legen ihren Schwerpunkt auf rückblickende Zeitauswertung oder reine Vorausplanung, während ClockWise beide Ansätze in einer einzigen Anwendung vereint. Dadurch entsteht ein umfassender Überblick für eine noch gezieltere, effizientere und bewusstere Nutzung der Zeit – ohne den Wechsel zwischen mehreren Anwendungen.

Ein besonderer Fokus liegt dabei auf der Benutzerfreundlichkeit. Die Anwendung soll so intuitiv und zugänglich gestaltet sein, dass sie nicht nur im professionellen Umfeld, sondern auch im privaten Alltag genutzt werden kann – von Studierenden, Berufstätigen und allen, die ihre Zeit besser organisieren möchten.

Mit ClockWise möchten wir eine Lösung schaffen, die am Tag der jungen Softwareentwickler nicht nur technisch überzeugt, sondern auch durch ihren praktischen Nutzen Nutzer:innen dazu inspiriert, ClockWise als festen Bestandteil ihrer persönlichen Zeitplanung zu integrieren – und damit sowohl ihren beruflichen als auch privaten Alltag zu entlasten.

2 Formale Grundlagen

In diesem Kapitel werden die wesentlichen Rahmenbedingungen für die Entwicklung der Software „ClockWise“ festgelegt. Dazu zählen insbesondere die verwendeten Programmiersprachen, die Entwicklungsumgebung sowie die Sprachen, in denen die Anwendung und die Projektdokumentation ausgeliefert werden.

Die Entwicklung und das Testen der Anwendung erfolgen auf handelsüblichen Windows-10- bzw. Windows-11-Rechnern. Die serverseitige Logik wird mit **Python** unter Einsatz des Web-Frameworks **Flask** umgesetzt. Für die Speicherung der Anwendungsdaten wird eine **SQLite**-Datenbank verwendet. Die Benutzeroberfläche wird mit **HTML5**, **CSS3** und **JavaScript** realisiert und ist somit in allen modernen Webbrowsern lauffähig. Die Benutzeroberfläche ist responsiv gestaltet und für verschiedene Bildschirmgrößen optimiert.

Die Sprache der Benutzeroberfläche ist **Englisch**; sämtliche Nutzerdialoge und Bedienelemente werden daher auf Englisch bereitgestellt. Die Projektdokumentation wird hingegen auf **Deutsch** verfasst und im Team kollaborativ mit LaTeX unter Verwendung von **Overleaf** erstellt. Für die zentrale Versionsverwaltung kommt **GitLab** zum Einsatz.

3 Projektablauf

In diesem Kapitel werden die Meilensteine und der zeitliche Ablauf des Projekts beschrieben. Der geplante Verlauf wird strukturiert dargestellt und durch ein Gantt-Diagramm visualisiert.

3.1 Meilensteine

Im Rahmen des Projekts wurden verschiedene Meilensteine definiert, die den Projektfortschritt strukturieren und die Entwicklungsschritte dokumentieren. Diese Meilensteine lassen sich in zwei Kategorien unterteilen: externe und interne Meilensteine.

Externe Meilensteine umfassen alle offiziellen Abgaben, wie zum Beispiel das Angebot, das Pflichtenheft, den Fachentwurf, den technischen Entwurf sowie die Testdokumentation. Diese Meilensteine haben jeweils einen festen Abgabetermin und sind mit konkreten Dokumenten verbunden, die dem Betreuer oder im Rahmen der Lehrveranstaltung vorgelegt werden.

Interne Meilensteine markieren wichtige Zwischenstände innerhalb der Projektentwicklung und strukturieren die Umsetzung in Form von drei Prototypen. Jeder Prototyp verfolgt einen spezifischen Entwicklungsfokus:

- **Prototyp 1 – Zeiterfassung und Grundfunktionen:** In dieser Phase wird die Basisversion der Anwendung erstellt. Diese beinhaltet unter anderem die Start-/Stopp-Funktion zur Zeiterfassung, eine einfache Übersicht über gespeicherte Zeitdaten sowie die Möglichkeit zur Nutzerregistrierung und Kontoverwaltung.
- **Prototyp 2 – Auswertung und Analyse:** Im zweiten Prototyp werden Analysefunktionen ergänzt. Die Anwendung ermöglicht Auswertungen gespeicherter Zeiten, z.B. durch grafische Darstellung in Balkendiagrammen. Außerdem werden erweiterte Funktionen zur Datenfilterung und -sortierung integriert.
- **Prototyp 3 – Zusatzfunktionen und Finalisierung:** In der letzten Entwicklungsphase werden optionale Zusatzfunktionen umgesetzt, z.B. Exportmöglichkeiten (z.B. PDF), Feedback per E-Mail oder Benachrichtigungen bei übermäßigem Zeitaufwand. Zudem wird das Design finalisiert und die App auf eine stabile Abschlussversion vorbereitet.

Die folgende Tabelle gibt einen Überblick über alle definierten Meilensteine, deren zugehörige Dokumente sowie die entsprechenden Abgabetermine.

#	**Meilenstein**	**Dokumente**	**Abgabetermin**
1	Projektstart	-	07.04.25
2	Abgabe: Angebot	Angebot	23.04.25
3	Abgabe: Pflichtenheft & Abnahmetestspezifikation	Pflichtenheft & Abnahmetestspezifikation	14.05.25
4	Prototyp 1 - Zeiterfassung & Grundfunktionen	-	15.05.25
5	Zwischenpräsentation	-	22.05.25
6	Abgabe: Fachentwurf	Fachentwurf	04.06.25
7	Prototyp 2 - Auswertung & Analyse	-	18.06.25
8	Abgabe: Technischer Entwurf	Technischer Entwurf	25.06.25
9	Prototyp 3 - Zusatzfunktionen & Finalisierung	-	02.07.25
10	Abgabe: Testdokumentation	Testdokumentation	09.07.25
11	TDSE	-	17.07.25

3.2 Geplanter Ablauf

In Abbildung 3.1 ist der geplante Ablauf des Projekts als Gantt-Diagramm visualisiert.

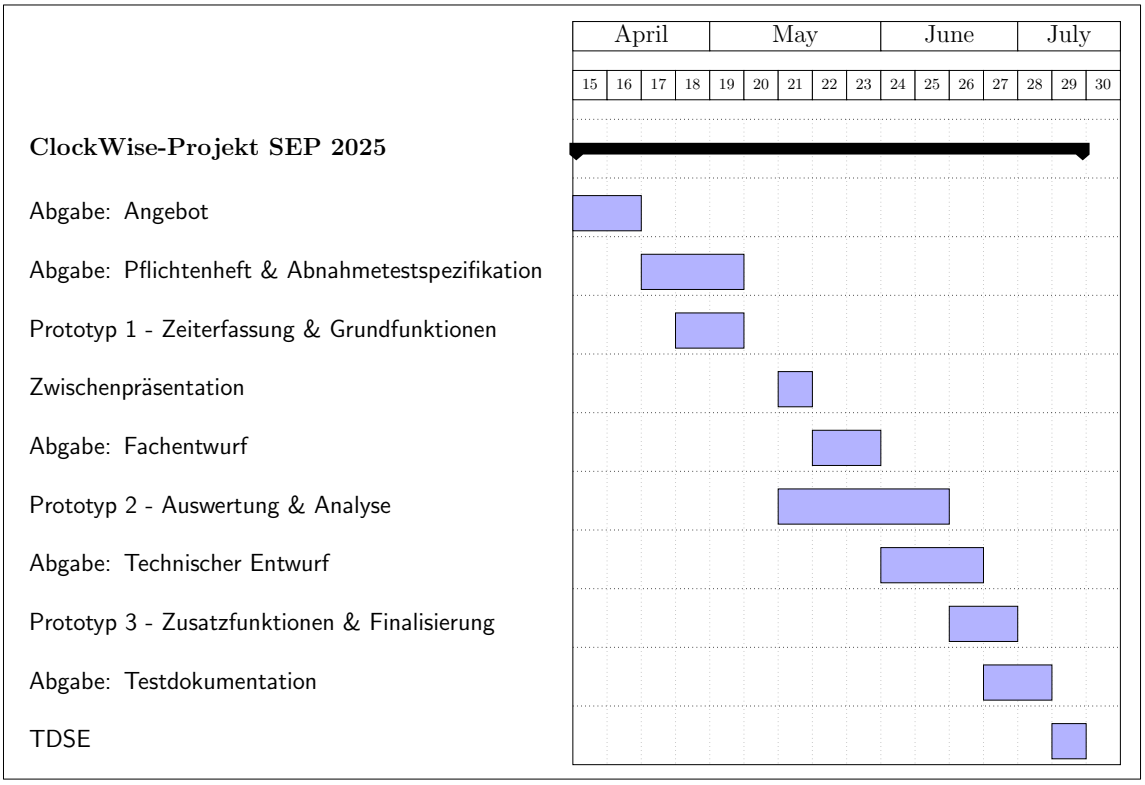

Abbildung 3.1: Gantt-Diagramm des Projekts ClockWise

4 Projektumfang

In diesem Kapitel wird der Umfang des Projekts *ClockWise* beschrieben. Es werden die zu liefernden Ergebnisse, der geschätzte zeitliche Aufwand sowie die zentralen funktionalen Eigenschaften der entwickelten Anwendung dargelegt.

4.1 Lieferumfang

Zum Abschluss des Projekts wird dem Auftraggeber eine voll funktionsfähige Webanwendung zur projektorientierten Zeiterfassung und Analyse bereitgestellt. Die Anwendung ermöglicht es Nutzer:innen, ihre Zeit strukturiert zu erfassen, auszuwerten und in einer übersichtlichen Kalenderdarstellung darzustellen. Zusätzlich zum ausführbaren System erhält der Auftraggeber Zugriff auf den vollständigen Quellcode der Anwendung, einschließlich Frontend- und Backend-Komponenten.

Zur Unterstützung der Nutzung wird eine umfassende Benutzerdokumentation erstellt. Weiterhin wird die technische Dokumentation übergeben, die sowohl die Systemarchitektur als auch zentrale Aspekte der Implementierung erläutert. Als begleitende Materialien werden außerdem sämtliche im Verlauf des Projekts erstellten Dokumente übergeben, darunter das Angebot, das Pflichtenheft, der Fachentwurf, der technische Entwurf sowie die Testdokumentation und Präsentationsunterlagen.

4.2 Kostenplan

Der Gesamtaufwand für das Projekt wird auf etwa 210 Stunden pro Teammitglied geschätzt. Diese Zeit umfasst nicht nur die konkrete Umsetzung der Anwendung, sondern auch alle begleitenden Aufgaben wie Planung, Dokumentation, Tests, Teamkoordination und Meetings. Die geschätzten 210 Stunden pro Person ergeben sich aus einer realistischen Einschätzung des gesamten Projektumfangs. Dabei wurden alle Phasen eines typischen Softwareprojekts berücksichtigt – von der Anforderungsanalyse über die Entwicklung bis hin zu Qualitätssicherung und Projektkoordination. Die Stundenanzahl spiegelt somit nicht nur die reine Entwicklungszeit wider, sondern auch den notwendigen Aufwand für ein qualitativ hochwertiges und gut dokumentiertes Ergebnis.

An dem Projekt arbeiten sieben Studierende und es findet im Rahmen des Softwareentwicklungspraktikums statt, wobei für den Studiengang Informatik und Wirtschaftsinformatik jeweils sieben Credits pro Studierenden vorgesehen sind. Für eine grobe Kostenschätzung bei externer Beauftragung (100 €/h) ergibt sich ein Gesamtwert von 210 h × 7 Personen × 100 €= **147.000 €**.

4.3 Funktionaler Umfang

Die Anwendung *ClockWise* bietet eine umfassende Lösung zur strukturierten Zeiterfassung und Projektorganisation. Nutzer:innen können ihre Zeit entweder live über eine Start-/Stopp-Funktion erfassen oder manuell Einträge hinzufügen und nachträglich bearbeiten. Erfasste Zeiten werden projektbezogen gespeichert und können sowohl in einer Kalenderansicht als auch in grafischen Auswertungen dargestellt werden. Darüber hinaus erlaubt die Anwendung den Vergleich der Ist-Zeiten mit individuell festgelegten Zielwerten. Ein Fortschrittsbalken visualisiert das Verhältnis von bereits investierter zu geplanter Zeit.

Zur Analyse und Archivierung stehen Exportfunktionen in verschiedenen Formaten wie PDF und CSV zur Verfügung. Projekte können nach Typen wie „Hausarbeit“, „Klausurvorbereitung“ oder „Teamarbeit“ kategorisiert werden. Zusätzlich beinhaltet die Anwendung eine Benutzer- und Teamverwaltung. So können mehrere Accounts einem Team zugeordnet und Aufgaben transparent verwaltet werden. Auch die Übersicht über die Zeitverteilung im Team wird ermöglicht, um eine faire Aufgabenteilung sicherzustellen.

5 Entwicklungsrichtlinien

In diesem Kapitel werden die zentralen Richtlinien für die Entwicklung des Projekts beschrieben. Dazu gehören Vorgaben zum Konfigurationsmanagement, Programmierstandards sowie der Einsatz von unterstützenden Tools.

5.1 Konfigurationsmanagement

Das Projekt verwendet ein zentrales GitLab-Repository zur vereinfachten Nachverfolgung von Änderungen und Zusammenarbeit. Dementsprechend gelten folgende festgelegte Regeln:

- Klare und sinnvolle Commit-Nachrichten
- Es wird ausschließlich nur funktionierender und getesteter Code hochgeladen.
- Temporäre Dateien oder persönliche Einstellungen werden nicht ins Repository aufgenommen.
- Änderungen im `main`-Branch erfolgen über Merge Requests.

Außerdem folgt eine Aufteilung in folgende Bereiche:

- `/backend` - für den Python-Code.
- `/frontend` – für HTML-, CSS- und JavaScript-Dateien.
- `/docs` – für Dokumentation oder andere Texte.

5.2 Design- und Programmierrichtlinien

Der Dokumentationsstil in Python erfolgt im Google-Style, welcher ähnlich zu PEP 257 ist. Für die Dokumentation in JavaScript richtet sich diese nach JSDocs. Der Programmierstil richtet sich nach folgenden festgehaltenen Regeln, um die Verständlichkeit des Codes zu wahren, unter anderem werden auch Richtlinien aus vorangegangenen Modulen wie Programmieren mit Java und C++ aufgenommen:

- Einrückung mit 4 Leerzeichen, besonders bei Python.
- Sinnvolle, englischsprachige Namen für Variablen und Funktionen.
- Clean Code mit sinnvollen Kommentaren.

5.3 Verwendete Software

Für die Entwicklung des Projekts kommen Softwarelösungen zum Einsatz, die den Programmierprozess, die Projektorganisation sowie das Design und die Zusammenarbeit im Team unterstützen.

5.3.1 Programmiersprachen

Für die Umsetzung des Projekts kommt im sogenannten Back-End die Programmiersprache Python zum Einsatz. Im Front-End werden für die Web-basierte Entwicklung und die Benutzeroberfläche HTML, JavaScript und CSS3 verwendet.

5.3.2 Projekt-Abhängigkeiten

Das Projekt nutzt hauptsächlich Flask als Framework für die Webentwicklung mit Python.

5.3.3 Entwicklungstools

Die Entwicklungsumgebung besteht aus folgenden Softwares zur Bearbeitung:

- **PyCharm** – IDE für Python, HTML- und JavaScript-Entwicklung
- **GitLab** – für die Versionskontrolle
- **Figma** – als UI/UX Design Tool
- **SQLite** – für die Implementierung der Datenbank
- **Black** – als Code-Formatter

5.3.4 Team-Tools

Für die Kommunikation und Projektorganisation setzen wir folgende Tools ein:

- **Jira** – zur Aufgabenverteilung und Planung
- **GitLab** – die in den Entwicklungstools benannte Versionskontrolle und Kollaborationsplattform
- **Overleaf** – zur gemeinsamen Bearbeitung von LaTeX-Dokumenten
- **Discord** – für schnelle Absprachen im Team
- **Diagrams.net** – zur Erstellung von Diagrammen oder Grafiken

6 Projektorganisation

Dieses Kapitel beschreibt die Kommunikation des Teams untereinander, sowie mit der Betreuerin.

6.1 Schnittstelle zum Auftraggeber

Die Auftraggeberin wird in (zwei-)wöchentlichen Treffen über den aktuellen Stand des Projekts informiert. Dabei werden sowohl der Fortschritt als auch mögliche Herausforderungen thematisiert. Die Meetings finden im BRICS statt.

Bei Rückfragen kann der Teamleiter per E-Mail Kontakt zu den studentischen Hilfskräften aufnehmen. Bei allgemeinen Fragen kann die Auftraggeberin direkt angeschrieben werden. Beide Parteien sind über ihre TU-Mail-Adressen erreichbar.

6.2 Schnittstelle zu anderen Projekten

Die anderen Projekte des Softwareentwicklungspraktikums am Institut sind unabhängig von diesem Projekt, weshalb es keine direkten Schnittstellen zu ihnen gibt. Auch werden keine bestehenden Anwendungen übernommen oder integriert.

Das Frontend und Backend werden im eigenen Team entwickelt, wobei diese Aufteilung flexibel gehandhabt wird.

6.3 Interne Kommunikation

Die interne Kommunikation des Teams erfolgt hauptsächlich über Discord und WhatsApp.

WhatsApp dient der Klärung allgemeiner Themen, kurzer Fragen und spontaner Probleme. Wöchentliche Besprechungen finden über Discord statt, um den Projektfortschritt zu diskutieren und die Aufgaben im Team zu verteilen.

Vor jeder Abgabe von Dokumentationen findet ein gemeinsames Treffen auf Discord statt, in dem das Dokument zusammengestellt, überarbeitet und ergänzt wird. Bei Bedarf können zusätzliche Treffen entweder über Discord oder vor Ort an der Universität organisiert werden.

Das Team hat eine Teamleiterin, die die Kommunikation mit der Auftraggeberin und den studentischen Hilfskräften übernimmt. Für den Fall einer Erkrankung übernimmt eine festgelegte Vertretung die Aufgaben der Teamleitung.

Zusätzlich gibt es zwei Time Keeper, die die Deadlines im Blick behalten und das Team rechtzeitig daran erinnern.

Bei Konflikten wird zunächst die betroffene Person direkt angesprochen. Sollte keine Lösung gefunden werden, informiert das Teammitglied die Teamleiterin, die die Angelegenheit gegebenenfalls an eine höhere Instanz weiterleiten kann.

7 Glossar

Backend Serverseitige Logik und Datenverarbeitung.

Black Formattiert den Python-Code anhand von festgelegten Regeln, um einen einheitlichen Code zu erhalten (siehe `https://github.com/psf/black`).

BRICS Braunschweig Integrated Centre of Systems Biology.

Code-Formatter Software, die den vorgelegten Code umformattiert, um einen lesbareren und konsistenteren Code zu erzeugen.

CSV „Comma-Separated Values“ – ein Dateiformat zur Speicherung tabellarischer Daten in Textform.

CSS3 Cascading Style Sheets – eine Stylesheet-Sprache zur Gestaltung von HTML-Inhalten.

Datenbank Ein System zur strukturierten Speicherung, Verwaltung und Abfrage großer Datenmengen.

Diagrams.net Ein webbasiertes Tool zur Erstellung von Diagrammen (siehe `https://www.diagrams.net/`).

Discord Eine Community-Chatplattform zur Kommunikation in Text- und Sprachkanälen (siehe `https://discord.com/`).

Export Funktion zur Ausgabe und Speicherung von Daten aus der Anwendung in verschiedenen Dateiformaten, um sie weiterzuverwenden oder zu archivieren.

Figma Eine Web-basierte Design-Plattform, welche kollaboratives Arbeiten in Echtzeit an Benutzeroberflächen ermöglicht (siehe `https://www.figma.com/`).

Flask Leichtgewichtiges Web-Framework für Python zur Entwicklung von Webanwendungen (siehe `https://flask.palletsprojects.com/`).

Frontend Grafische Benutzeroberfläche zur Bedienung einer Website oder App.

GitLab Plattform zur Verwaltung von Git-Repositories, inklusive Funktionen für Code-Review, Continuous Integration und Projektmanagement (siehe `https://about.gitlab.com/`).

Google-Style Eine bestimmte Form der Dokumentation innerhalb eines Python Codes. Legt den Umgang mit Einrückung, Benennung, Importstruktur und Dokumentation fest (siehe `https://sphinxcontrib-napoleon.readthedocs.io/en/latest/example_google.html`).

HTML5 Die fünfte Version der Hypertext Markup Language zur Strukturierung von Webseiteninhalten.

JavaScript Eine Programmiersprache zur Umsetzung interaktiver Funktionen im Webbrowser.

Jira Ein Tool zur Projektplanung im Team (siehe `https://www.atlassian.com/software/jira`).

JSDocs Standard zur Dokumentation von JavaScript-Code, welcher eine strukturierte Beschreibung des Codes ermöglicht (siehe `https://jsdoc.app/`).

LaTeX Ein Textsatzsystem zur Erstellung wissenschaftlicher und technischer Dokumente (siehe `https://www.latex-project.org/`).

Overleaf Ein kollaborativer Online-Editor zur Bearbeitung von LaTeX-Dokumenten (verfügbar unter `https://www.overleaf.com/`).

PDF „Portable Document Format“ – ein weit verbreitetes Dateiformat zur originalgetreuen Darstellung und Weitergabe von Dokumenten, unabhängig vom verwendeten Betriebssystem oder Gerät.

PEP 257 Offiziele Standartisierung der Struktur und Formulierungen innerhalb von Dokumentationen im Python-Code (siehe `https://peps.python.org/pep-0257/`).

PyCharm Integrierte Entwicklungsumgebung (IDE) für die Arbeit mit Python (siehe `https://www.jetbrains.com/pycharm/`).

SQLite Eine leichtgewichtige, serverlose Datenbank, die direkt in Anwendungen eingebettet wird (siehe `https://www.sqlite.org/`).

Webanwendung Eine über den Webbrowser nutzbare Software, die keine Installation auf dem eigenen Gerät erfordert.

Webbrowser Ein Programm zur Anzeige und Nutzung von Webseiten (z.B. Chrome, Firefox, Edge).

WhatsApp Ein mobiler Messenger-Dienst zur Text- und Sprachnachrichtenkommunikation (siehe `https://www.whatsapp.com/`).