Hochschule
für Technik
Stuttgart

Betreutes Praktisches Studienprojekt

Praktikumsbericht

im Rahmen des Studiengangs
Wirtschaftsinformatik
der Hochschule für Technik Stuttgart

vorgelegt von:

Julian Raubald

(Matrikelnummer 1003812)

Praktikumsbetrieb:

iteratec

Betreut durch:

Titel Name Betreuer

Stuttgart, Tag. Februar. 2024

Inhaltsverzeichnis

1 Einleitung

In diesem Bericht schildere ich die Aufgaben und Projekte, welche ich in meinem Praxissemester im Wintersemester 23/24 bei dem Unternehmen iteratec GmbH durchgeführt und begleitet habe. Dabei gehe ich auf die eingesetzten Technologien ein und teile meine gesammelten Erfahrungen und ordne diese inhaltlich im Rahmen meines Studiums der Wirtschaftsinformatik ein.

1.1 Unternehmen iteratec GmbH

Iteratec ist ein Unternehmen, das sich auf digitale Produktinnovation, Software- und Architekturentwicklung sowie digitale Infrastrukturen spezialisiert hat. Als Partner für die digitale Transformation unterstützt es Unternehmen und öffentliche Organisationen bei der Digitalisierung ihrer Prozesse, Produkte und Dienstleistungen. Dies umfasst die Entwicklung von Innovationen, die technische Umsetzung individueller Softwarelösungen, deren Betrieb und Weiterentwicklung sowie Schulungen in agilen Methoden. Das Unternehmen entwickelt mobile Applikationen, Cloud-Architekturen und bietet Application Performance Management an. Es nutzt auch Technologien wie Blockchain, AI und IoT für die Entwicklung neuer Geschäftsmodelle. Zu den Kunden gehören mittelständische Unternehmen, DAX-Konzerne und Organisationen aus verschiedenen Branchen. 1996 in München gegründet, hat iteratec Standorte in Deutschland, Österreich und Polen und beschäftigt rund 500 Mitarbeiter und 120 Studierende. Seit 2019 sind viele Mitarbeiter über eine Genossenschaft am Unternehmen beteiligt.

1.2 Name und Geschichte

Das Unternehmen wurde 1996 gegründet und existiert bereits seit über 25 Jahren. Während zu dieser Zeit das Wasserfallmodell in der Softwareentwicklung weit verbreitet war hat sich iteratec schon früh auf agile Methoden und Werte spezialisiert. Erkennbar ist das insbesondere an dem Namen, welcher sich aus den Begriffen iterativ und Technologie zusammensetzt. Heute hat iteratec bereits über 1000 Softwareprojekte erfolgreich abgeschlossenen.

1.3 Unternehmensstruktur

Standortübergreifend hat iteratec 4 Hauptgeschäftsführer. An jedem Standort gibt es jeweils einen Geschäftsstellenleiter. Jeder Mitarbeiter hat eine persönliche Führungskraft (pFK), welche für die persönliche Entwicklung vor allem im, aber auch außerhalb des Unternehmens zuständig ist. Jedes Projekt hat einen Product Owner (PO), welcher die Interessen des Kunden im Projekt vertritt, und einen Projektleiter, welcher dafür zuständig ist die Fristen und eingesetzten Ressourcen zu managen. Alle Mitarbeiter begegnen sich Standort- und Positionsübergreifend auf Augenhöhe, was sich unter anderem dadurch zeigt, dass sich alle duzen. Die bisherigen Alleingesellschafter der iteratec GmbH haben im Jahr 2018 entschieden, das Unternehmen weder innerhalb ihrer Familie zu vererben noch dieses an externe zu verkaufen. Daraus entstand die iteratec nurdemtean eG, eine Genossenschaft bei der sich die Mitarbeiter am Unternehmen beteiligen, mit dem Ziel, die Unternehmenskultur zu halten.

1.4 Studierende bei iteratec (SLAB)

Die Abteilung für studierende, das SLAB (Studentenlabor), ist Standort übergreifend und schließt sowohl alle Studierenden als auch deren Betreuer mit ein. Halb-jährlich werden neue SLAB Leiter gewählt, deren Aufgabe es ist, alle anfallenden Aufgaben an studierende zu delegieren und im engen Austausch mit dem SLAB-Verantwortlichen zu stehen. Eine Aufgabe ist das bereitstellen einer Startbegleitung für neue Studierende.

Andere Aufgaben sind z.B. das organisieren von Workshops und Events.

1.5 Meetings und Veranstaltungen

1.5.1 Barcamp

1.5.2 SLAB-Meeting

1.5.3 GS-Meeting

1.5.4 BUILD23

2 Tätigkeitsbereich und Aufgabenstellung

Im folgenden Kapitel werde ich genauer auf meinen Arbeitsbereich und meine Aufgaben eingehen.

2.1 Arbeitsbereich und Ausstattung

Sämtliche Arbeitsplätze sind sogenannte Flexarbeitsplätze. Das heißt, dass man sich morgens einen freien Schreibtisch sucht und dort seine Arbeit beginnt. Es gibt vereinzelt Bereiche wie das SLAB, wo sich bestimmte Gruppen zum arbeiten treffen aber auch dort hat niemand einen festen Arbeitsplatz. Jeder dieser Arbeitsplätze verfügt über einen höhenverstellbaren Schreibtisch und 2 Monitore inklusive Dockingstation für den Laptop. Jedem Mitarbeiter ist es gestattet seine Arbeit von Zuhause aus durchzuführen, solange man mindestens einen Tag in der Woche vor Ort im Büro arbeitet. Ein Großteil der Kommunikation mit Kollegen erfolgt daher über Videokonferenzen oder Chats mithilfe der Software Microsoft Teams. Für Meetings stehen zahlreiche Meeting räume zur Verfügung, welche man sich jederzeit reservieren kann. Für die Arbeit erhält man als Student einen Dell Laptop mit Windows-Betriebssystem.

2.2 Hauptaufgaben

Zu meinen Hauptaufgaben gehörte insbesondere das entwickeln von Software beziehungsweise das schreiben von Quellcode. Eigenständiges recherchieren und Problemlösung gehören dabei fest zum Arbeitsalltag.

2.3 Technologien und Tools

Für das schreiben von Code verwende ich als Entwicklungsumgebung die IDE Visual Studio Code.

3 Grundlagen (falls notwendig)

Text

3.1 Theoretische Grundlagen

Theoretische Grundlagen, die für das Verständnis deiner Tätigkeiten erforderlich sind (z.B. spezifische Programmiersprachen, Frameworks, Design Patterns etc.)

3.1.1 Programmiersprachen und Frameworks

Die in diesem Bericht verwendeten Programmiersprachen sind Pseudocode und Python (Version 3.11). Wichtige Frameworks sind langchain, chromadb, azure, ...

Python, KI/LLMs, Few-shot prompting, singleshot prompting,

3.2 Einführung in spezielle Technologien oder Methoden

Einführung in spezielle Technologien oder Methoden, die im Unternehmen verwendet werden

4 TenderInsights

4.1 Kurze Projektbeschreibung

In dem Projekt geht es darum einen PoC - also einen Proof of Concept zu erstellen, der mithilfe von AI bzw. Large Language Models (LLM) große Dokumente nach gesuchten Inhalten durchsucht und aufbereitet. Im Fokus stehen dabei insbesondere öffentliche Ausschreibungen zur Akquise von neuen Projekten für das Unternehmen. Bisher hat ein Mitarbeiter aus PMO die Ausschreibungen händisch grob für passend oder nicht geeignet erklärt und diese dann an die Akquise weitergeleitet. Dort hat ein Mitarbeiter die Ausschreibung sehr genau analysiert und einen sogenannten One Pager erstellt, der alle Daten welche für eine Entscheidung, ob man sich bewerben möchte oder nicht, relevant sind. Aufgabe des PoC ist es nun diesen One Pager mithilfe von künstlicher Intelligenz und einem LLM aus einer oder mehreren bereitgestellten PDF-Datei zu generieren. Der Anwender soll dies über ein Userinterface im Stile einer Webanwendung bedienen können.

4.2 Verwendete Software

In diesem Kapitel werden alle Software-Werkzeuge beschrieben, welche im Entwicklungsumfeld des TenderInsights verwendet werden.

4.2.1 Technologien

Tabelle 4.1: Übersicht der Technologien

Name	Beschreibung
Name1	Beschreibung1
Name2	Beschreibung2
Name3	Beschreibung3

4.2.2 Werkzeuge

Tabelle 4.2: Übersicht der Werkzeuge

Name	Beschreibung
Werkzeug1	Beschreibung1
Werkzeug2	Beschreibung2
Werkzeug3	Beschreibung3

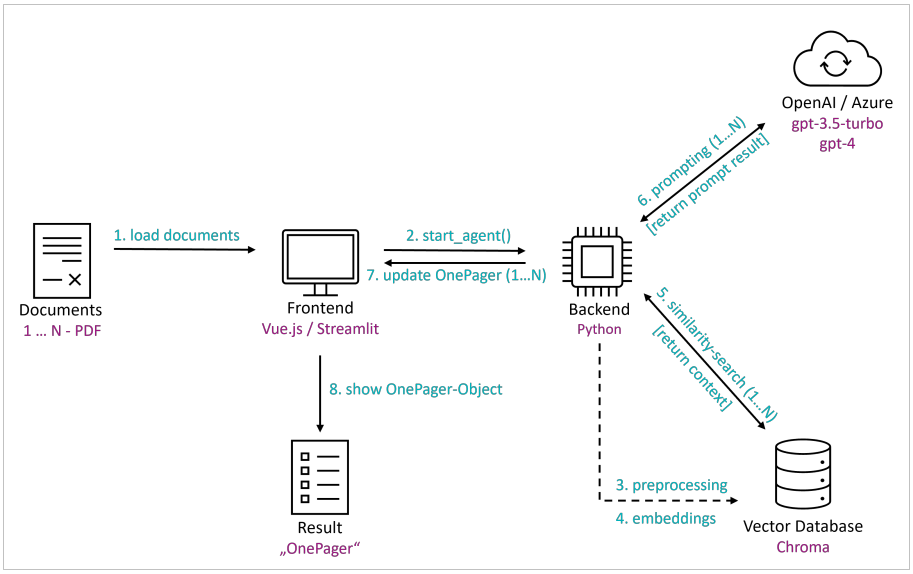

Abbildung 4.1: Schematische Darstellung der Software-Architektur

4.3 Projektstand bei Arbeitsbeginn

Zum Zeitpunkt meines Projekteintrittes gibt es bereits ein Frontend welches mithilfe von Streamlit implementiert wurde. PDF-Dateien können über einen File-Uploader im Frontend an die Anwendung übergeben werden. Da die meisten LLM-Modelle eine begrenzte Token-Anzahl für die Abfragen haben ist es nicht möglich das gesamte Dokument an das LLM zu übergeben. Daher wird das Dokument aufgesplittet und mittels einer Vektordatenbank auf Ähnlichkeit mit die gesuchte Information abgeglichen. Die k Ergebnisse mit der größten Ähnlichkeit werden zusammen mit dem Prompt als Abfrage an das LLM übermittelt. Die Abfragen werden mit der langchain API durchgführt, das verwendete Modell ist "gpt-3.5 turbo". Der Großteil dieser Funktionalitäten findet sich in der Document-Agent Klasse. Es gibt 3 Ausschreibungsdokumente in verschiedenen Größen mit denen die Abfragen getestet werden können. Es gibt 3 Prompts, welche bislang nur bedingt gute Ergebnisse liefern.

4.4 Deine Rolle und Aufgaben im Projekt

Ich bin als Softwareentwickler in das Projekt gekommen. Meine Aufgabe bestand Anfangs darin, die Prompts für die einzelnen Felder des One Pagers zu formulieren um die gelieferten Ergebnisse zu verbessern. Dies resultierte in den Aufbau einer Test- und Evaluierungsarchitektur mithilfe derer die Prompts vollautomatisch bewertet werden und anhand ihrer Kennzahl ausgewählt und weiter optimiert werden. Durch die Komplikationen mit dem Datenschutz wurde der Umstieg von langchain auf Azure von Microsoft beschlossen, welchen ich durchgeführt habe. Zum Abschluss des Projektes wurden wir angefragt, ob wir aufgrund des hohen Interesses an dem Projekt eine Präsentation während des GS-Meetings und der BUILD23 halten möchten. Das erstellen der Präsentation und Vortragen dieser gehörte auch zu meinem Aufgabenbereich.

Anf: Datenschutz, große seiten, chatten mit Dokument Grundlagen: Token, Similarity Search, Prompts

4.5 Herausforderungen und Lösungsansätze

4.5.1 Optimierung der Prompts

Um die Prompts zu verbessern ist es notwendig, diese an unterschiedlichen Ausschreibungen zu testen und das gelieferte Ergebnis mit der richtigen Antwort zu vergleichen. Da das größte Dokument über 100 Seiten hat und wir keine Musterlösungen haben war es schwierig Aussagen über die Qualität der Prompts zu treffen. Die Lösung war das Einführen einer Metrik welche die Qualität der Prompts misst. Es wurden 10 Dokumente ausgearbeitet und sämtliche wichtige Informationen in json Dateien gespeichert. Anschließend wurde eine Testarchitektur geschaffen, in welcher man die gewünschten Prompts vollautomatisiert gegen die 10 Testdokumente abfragt und anschließend die Resultate zusammen mit der Musterlösung von ChatGPT auf inhaltliche Übereinstimmung überprüfen und bewerten lässt. Aus den so generierten Bewertungen lassen sich nun Aussagekräftige Kennzahlen generieren. Zur Darstellung wurde eine Visualisierungsklasse geschrieben, welche die Punktzahl der Prompts graphisch darstellt (siehe 4.2) und so schnell erkennbar ist, ob der aktuelle Prompt zu einer Verbesserung oder Verschlechterung der Ergebnisse führt.

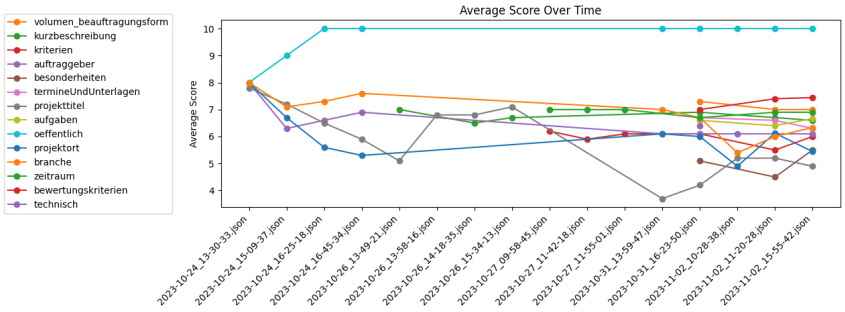

Abbildung 4.2: Visualisierung der durchschnittlichen Promptqualität im Laufe der Zeit

4.5.2 Datenschutz

Das übermitteln und verarbeiten von Personenbezogenen Daten ist in Deutschland nur mit Ausdrücklicher Genehmigung der entsprechenden Person zulässig. Es ist

unklar inwiefern das Übermitteln der Daten aus den Dokumenten über die openAI API hierunter einzustufen ist. Offiziell heißt es, dass die Daten, welche über die API geteilt werden, nicht zu Trainingszwecken genutzt werden. Da die Server von openAI allerdings in den USA liegen gestalten sich Probleme mit europäischen Datenschutzrecht. Als Lösung wurde die Umstellung auf Azure in Betracht gezogen, da die Server auf europäischen Boden stehen und damit zumindest nach europäischen Recht Datenschutzkonform sind. Ein großer Nachteil von Azure ist aber, dass man für das erstellen der Vektordatenbank auf 16 Inputs beschränkt ist, während openAI hier keine Einschränkungen vorgibt. Größere Dokumente können leicht auf 1 Input pro Seite kommen, was ein Einbetten von Dokumenten ab 20 Seiten verhindert. Um das Problem zu lösen habe ich statt die vorgegebene Embedding-Methode zu verwenden eine eigene Methode erstellt, welche die Dokumente bzw. die Inputs auf mehrere Listen mit einer Größe kleiner gleich 16 aufteilt. Anschließend werden die einzelnen Listen über eine REST API zu embeddings umgewandelt und anschließend wieder zusammengesetzt. Am ende wird eine Collection mit den Embeddings, den Dokumenten und entsprechenden Metadaten erstellt, welche zentral in einem VectorDataManager zur Verfügung gestellt wird. Gegen diese Collection können anschließend Abfragen wie Similarity-Searches durchgeführt werden.

4.6 Ergebnisse und Erfahrungen

5 Reflexion und Bewertung

5.1 Persönliche Reflexion über das Praxissemester

Persönliche Reflexion über das Praxissemester: Was hast du gelernt? Welche Fähigkeiten konntest du verbessern?

5.2 Bewertung der Zusammenarbeit im Team und im Unternehmen

5.3 Einordnung der erlernten Inhalte und Fähigkeiten im Kontext deines Studiums

6 Fazit und Ausblick

6.1 Zusammenfassung der wichtigsten Erkenntnisse

6.2 Weiterer Werdegang

Wie wird sich das Praxissemester auf deinen weiteren Werdegang auswirken?

6.3 Einordnung der erlernten Inhalte und Fähigkeiten im Kontext deines Studiums

7 Anhang

Eventuell Codebeispiele, Diagramme, Screenshots oder sonstige unterstützende Materialien, die im Hauptteil zu umfangreich wären

7.1 Liste der verwendeten Tools und Technologien

7.2 Literaturverzeichnis

https://www.deeplearning.ai/short-courses/building-systems-with-chatgpt/

8 Eidesstattliche Erklärung

Bestätigung, dass du den Bericht selbstständig verfasst hast (sofern von deiner Hochschule gefordert).

9 Zusammenfassung und Ausblick

Im letzten Kapitel sollte die Arbeit zusammengefasst und ein Fazit gezogen werden. Außerdem sollte beschrieben werden, wie es mit dem Projekt weitergehen kann und welche Punkte vielleicht interessant wären aber im Rahmen der Arbeit nicht bearbeitet werden konnten.

10 Tutorial zu dieser LaTeX-Vorlage

Dieses Kapitel ist spezifisch für die LaTeXVorlage und sollte natürlich in der finalen Abgabe nicht enthalten sein. Dieser Teil der Arbeit kann bei Bedarf durch einen Kommentar einfach ausgeblendet werden (siehe thesis.tex Zeile 125).

10.1 Verwendung dieser Vorlage

Dieses Template ist für die Verwendung mit pdflatex gedacht. Am einfachsten ist es die Vorlage in Overleaf Overleaf c/o Digital Science, n. d. zu öffnen. Overleaf ist eine Onlineanwendung zum Arbeiten mit LaTeX, was den Vorteil hat, dass nichts lokal installiert werden muss und mit jedem Betriebssystem gearbeitet werden kann, das über einen Browser verfügt. Außerdem ist es möglich mit mehreren Personen gemeinsam an einem Projekt zu arbeiten. Die Vorlage kann auch mit einer lokalen Installation verwendet werden. Die Website von Overleaf bietet zudem einige gute Tutorials zum Arbeiten mit LaTeX: https://www.overleaf.com/learn.

Fehler und Warnungen, die beim Compilieren erzeugt werden, sollten direkt behoben werden, da es später schwierig sein kann den eigentlichen Auslöser einer Fehlermeldung zu finden. Manchmal sieht das Dokument trotz Fehlermeldung oder Warnung korrekt aus, der Fehler macht sich dann aber später bemerkbar. Die Meldungen sind leider oft nicht sehr aussagekräftig, weshalb es am einfachsten ist direkt nach dem Auftreten eines Fehlers den Teil der Arbeit anzuschauen, der als letztes geändert wurde.

10.2 Projektstruktur

Der Hauptteil einer Thesis besteht üblicherweise aus mehreren Kapiteln, die verschiedene Aspekte der Arbeit beleuchten. Es ist ratsam für jedes Kapitel ein eigene Tex-Datei anzulegen, damit der Quellcode übersichtlich bleibt. Durch einen numerischen Präfix (z.B.: `20_relatedwork.tex` siehe Abb. 10.1) werden die Quelldateien in der richtigen Reihenfolge in Overleaf angezeigt. Wir verwenden 20 anstatt 2, damit wir nachträglich auch noch 21, 22, etc. einfügen können.

10.2.1 Unterkapitel

Unterkapitel sollten ein abgeschlossenes Thema behandeln. Einzelne Unterkapitel in einem Kapitel sind zu vermeiden, also z. B. in Kapitel 2 das Unterkapitel 2.1, aber kein weiteres Unterkapitel. In diesem Fall ist es besser entweder den Inhalt von 2.1 direkt in Kapitel 2 zu schreiben, oder falls 2 und 2.1 thematisch zu weit voneinander entfernt sind, aus Unterkapitel 2.1 ein eigenes Kapitel 3 zu machen. Dieses Unterkapitel ist ein negativ Beispiel dafür.

10.3 Grafiken

In der Informatik sind die häufigsten Grafiken entweder Diagramme oder Plots. Beide Arten von Grafiken lassen sich gut als Vektorgrafiken erstellen und einbinden. Der Vorteil von Vektor- gegenüber Pixelgrafiken ist, dass beliebig weit in eine Grafik hereingezoomt werden kann, ohne dass sie unscharf wird. Zudem benötigen Vektorgrafiken meistens weniger Speicherplatz.

10.3.1 Vektor- vs Pixelgrafiken

Für die meisten Abbildungen sollte man Vektorgrafiken verwenden, diese können verlustfrei skaliert werden und bieten somit die meisten Freiheiten. Wenn man Grafiken in R erstellt, können die Plots als pdf oder svg-Dateien abgespeichert werden

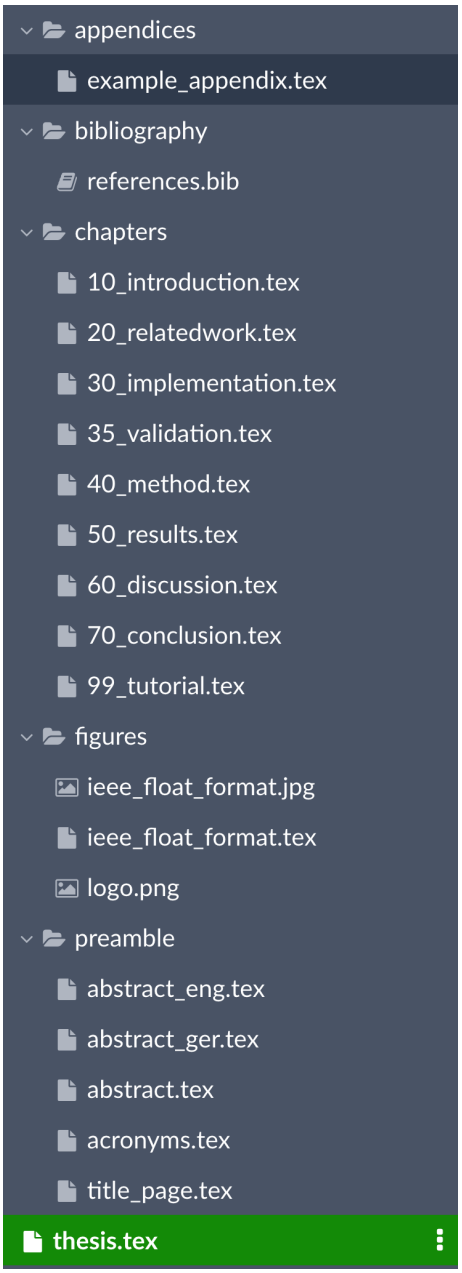

Abbildung 10.1: Ordnerstruktur eines LaTeX-Projektes

und liegen dann ebenfalls als Vektorgrafik vor. Ein weiteres beliebtes Programm zum Erstellen von Vektorgrafiken ist Inkscape Inkscape Project, 2020. Zudem bieten viele Programme die Möglichkeit eine Grafik z. B. als PDF zu exportieren, was in LaTeX als Vektorgrafik eingebunden werden kann. Adobe Indesign wird auch häufig zum Erstellen von Vektorgrafiken verwendet.

Profi-Tipp TikZ. Grafiken können direkt in LaTeX mit dem TikZ Paket Tantau, 2021 erstellt werden. Die Verwendung ist etwas gewöhnungsbedürftig, da Grafiken mit Code beschrieben werden, bietet aber viele Freiheiten. Außerdem werden die so erstellten Grafiken direkt in LaTeX gerendert und verwenden die selbe Schriftart wie im Text und eine konsistente Schriftgröße im gesamten Dokument.

Pixelgrafiken lassen sich nicht immer vermeiden, z. B. wenn eine Foto in die Arbeit eingebunden werden soll. In diesem Fall sollte darauf geachtet werden, dass die Grafik über eine ausreichende Auflösung verfügt. Eine Auflösung von 300 dpi ist ein guter Richtwert, um beim Drucken ein gutes Ergebnis zu erhalten.

Abbildungen 10.2 und 10.3 zeigen beide den Aufbau des IEEE Floating Point Formats. Abbidlung 10.3 ist eine Pixelgrafik, während Abbildung 10.2 mit TikZ erstellt wurde. Der Unterschied wird beim hereinzoomen deutlich.

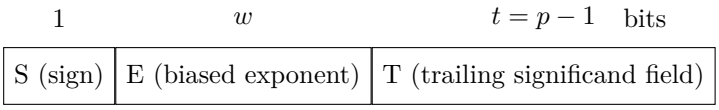

Abbildung 10.2: Aufbau des IEEE Floating Point Formats als Vektorgrafik mit TikZ erzeugt.

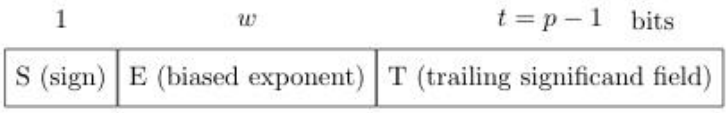

Abbildung 10.3: Aufbau des IEEE Floating Point Formats als Pixelgrafik.

10.4 Tabellen

Tabellen können in LaTeX direkt erstellt werden. Tabelle 10.1 zeigt ein Beispiel dafür. Einfache Tabellen lassen sich schnell erstellen, bei komplizierteren Tabellen ist es manchmal einfacher zusätzliche Pakete zu verwenden. Mit dem Paket `multirow` können z. B. einfacher Tabellen erstellt werden, bei denen einzelne Zeilen oder Spalten zusammengefasst sind.

Unter https://www.tablesgenerator.com/ findet man ein hilfreiches online Werkzeug zum erstellen von Tabellen. Wichtig ist es hier den „Default table style“ zum „Booktabs table style umzustellen“.

Parameter	binary16	binary32	binary64	binary128
k, storage width in bits	16	32	64	128
w, exponent field width in bits	5	8	11	15
t, significand field width in bits	10	23	52	112
emax, maximum exponent e	15	127	1023	16383
bias, $E - e$	15	127	1023	16383

Tabelle 10.1: IEEE 754-2019 Floating Point Formate als Beispiel für das Einbinden einer Tabelle.

10.5 Quellcode

Um Quellcode in die Arbeit einzubinden, können in LaTeX Listings verwendet werden. Es gibt für populäre Sprachen vorgefertigte Umgebungen, welche die Syntax farblich hervorheben. Quellcode sollte eingebunden werden, wenn eine konkrete Implementierung in einer Sprache erläutert wird. Für die Erklärung eines Algorithmus ist es oft übersichtlicher ein Schaubild oder Pseudocode zu verwenden. Es sollten nur kurze Codeabschnitte eingebunden werden, die für den Leser einfach nachvollziehbar sind und nur den für die Erklärung relevanten Code enthalten. Längere Codeabschnitte können im Anhang stehen. Der komplette Code, der für die Arbeit geschrieben wurde, sollte in einem Repository (Gitlab) abgelegt werden.

Listing 10.1 zeigt ein Beispiel für ein Codelisting in der Programmiersprache C.

Algorithmus 10.1 zeigt einen Routing Algorithmus als Pseudocode. Der Code wurde mit dem Paket algorithm2e Fiorio, 2017 erstellt.

```c
#include <stdio.h>
// comments are highlighted in green
void main() {
    // keywords of the language are highlighted in blue
    for (int i = 0; i <= 42; ++i) {
        printf("%d\n", i);
    }
    // strings are highlighted in red
    printf("Hello World!");
}
```

Listing 10.1: Beispiel für ein Codelisting in der Sprache C.

if *destination in west direction* **then**
  go West;
else if *destination in same column* **then**
  if *destination in north direction* **then**
    go North;
  else
    go South;
  end
else if *destination in north east direction* **then**
  go North or go East
else if *destination in south east direction* **then**
  go South or go East
else if *destination in same row and in east direction* **then**
  go East;
else if *at destination* **then**
  done;

Algorithmus 10.1: West First-Routing Algorithm.

10.6 Literatur

Ein Literaturverzeichnis sollte mit dem apa Paket für BibLatex erstellt werden. Dazu wird für jede Quelle ein Eintrag in der Datei `references.bib` angelegt. An der

passenden Stelle im Text können diese Einträge mit dem `\cite{}` Befehl zitiert werden. Für jede Quelle die zitiert wird, legt LaTeX im Literaturverzeichnis einen Eintrag an.

Beschreibungen der Quellen im Bibtex-Format müssen meistens nicht selbst erstellt werden, sondern können direkt bei vielen Verlagen und Bibliotheken direkt generiert werden. Google bietet mit dem „Scholar-Button“ ein Chromium Plugin, mit dem schnell bibtex-Einträge generiert werden können. Bei Google-Scholar generierten Bibtex-Einträgen muss auch eine manuelle Endkontrolle stattfinden, um zu prüfen, ob die Daten in Google korrekt gespeichert waren (z.B. fehlende Autoren, falsche Jahreszahl, etc.).

Es gibt verschiedene Arten wie man mit biblatex zitiert.
`\autocite{valdez2015reducing}` führt zu (Calero Valdez et al., 2015).
`\parencite{valdez2015reducing}` führt zu (Calero Valdez et al., 2015).
`\textcite{valdez2015reducing}` führt zu Calero Valdez et al. (2015).
`\cite{valdez2015reducing}` führt zu Calero Valdez et al., 2015.
`\autocite[siehe auch][S. 13]{valdez2015reducing}` führt zu (siehe auch Calero Valdez et al., 2015, S. 13).

Hier[1] gibt es ein hilfreiches Cheat-Sheet.

Je nach zitierter Dokumentsorte, sieht die Referenz im Literaturverzeichnis anders aus.

- Beispiel für einen Konferenzbeitrag (Nielsen & Molich, 1990)
- Beispiel für einen Journal-Artikel (Hollan et al., 2000)
- Beispiel für ein Buch (Zobel, 2014).
- Beispiel für eine Norm (*Ergonomie der Mensch-System-Interaktion - Teil 220: Prozesse zur Ermöglichung, Durchführung und Bewertung menschzentrierter Gestaltung für interaktive Systeme in Hersteller- und Betreiberorganisationen*, 2020).

[1] https://tug.ctan.org/info/biblatex-cheatsheet/biblatex-cheatsheet.pdf

- Beispiel für einen Weblink[2]

10.7 Abkürzungen

Für jede verwendete Abkürzung kann ein Eintrag in der Datei `acronyms.tex` angelegt werden. Wenn diese Abkürzung im Text zum ersten Mal auftaucht, sollte der Begriff ausgeschrieben werden mit der Abkürzung in Klammern dahinter. Bei weiteren Vorkommen im Text kann dann die eigentliche Abkürzung verwendet werden. In LaTeX gibt es dafür spezielle Befehle. Beispiel für ausgeschriebene Abkürzung (siehe Quelltext des Dokuments für die entsprechenden Befehle).

Erste Verwendung mit Erläuterung: Network Interface Controller (NIC).
Beispiel für das Verwenden der Abkürzung: NIC.

Die verwendeten Abkürzungen werden automatisch im Abkürzungsverzeichnis aufgelistet.

[2] http://imis.uni-luebeck.de („Institut für Multimediale und Interaktive Systeme“, 2023)

10.8 LaTeX Eigenarten

Hier noch ein paar latexspezifische Dinge beim Schreiben.

Non-Breaking Space. Mit dem Tildezeichen ~ bekommt man ein sog. non-breaking Space (nbsp). Das ist hilfreich, wenn man verhindern möchte, dass z.B. die Zeile mit einer Referenz beginnt (Beispiel: Listing 10.1 – Hier wird Listing niemals von der Zahl getrennt werden). Das nbsp wird auch verwendet, um den Abstand nach einem Punkt in einer Abkürzung (1 Einheit) nicht auf die Länge des Abstandes nach dem Satzende zu setzen (1,5 Einheiten). Das fällt nicht immer jedem auf, wenn man es aber einmal sieht, kann man es schwer wieder abschalten.
Beispiel: Dies ist eine Abk. in einem Satz. (korrekt) Die Leertaste wird hier immer gleich lang gehalten.
Beispiel: Dies ist eine Abk. in einem Satz. (falsch) Die Leertaste kann (!) hier vom Setzer verlängert werden, um die Satztrennung deutlicher zu machen.

Binde- und Gedankenstriche. Es gibt drei Stricharten in Latex. Diese haben unterschiedliche Bedeutungen. Der Bindestrich - (engl. hyphen) ist ein einfaches Minus und wird verwendet, um Trennung von Silben anzuzeigen oder Mehrsprachige Komposita zu ermöglichen (z.B. Dashboard-Anzeige). Siehe auch https://www.duden.de/sprachwissen/rechtschreibregeln/bindestrich.

Der deutsche Gedankenstrich (engl. en-dash) – der selten (!) bei Einschüben verwendet wird – sind zwei Minuszeichen und wird mit Leerzeichen (oder nbsp) abgesetzt. Siehe auch https://www.duden.de/sprachwissen/rechtschreibregeln/gedankenstrich. Im Englischen wird der en-dash u.a. dazu benutzt, Zahlenbereiche zu beschreiben: z.B. pages 4–9.

Der englische Gedankenstrich — (engl. em-dash) sind drei Minuszeichen und wird im Englischen für Einschübe benutzt und wird **nicht** mit Leertasten abgesetzt. Beispiel: People—at least most of them—are suprised at how the em-dash is used properly. Im Deutschen kommt dieser Gedankenstrich nicht vor. Verwen-

dung im Englischen siehe auch: https://www.merriam-webster.com/words-at-play/em-dash-en-dash-how-to-use

Anführungszeichen. Wichtig: Anführungszeichen im Deutschen sind anders als im Englischen und Französischen.

Deutsche Anführungszeichen – nach innen gewölbte doppelte Tief- und Hochkommata – werden mit Anführungszeichen und Backtick ("`) geöffnet und mit Anführungszeichen und Apostroph ("') geschlossen. Beispiel: „Latex sollte bei korrekt eingestellter Dokumentsprache aber die korrekten Anführungszeichen wählen.“ sagte er leichtsinnig und irrte sich.

https://www.duden.de/sprachwissen/rechtschreibregeln/anfuehrungszeichen

Englische Anführungszeichen – nach außen gewölbte doppelte Hochkommata – werden mit doppeltem Backtick geöffnet (``) und mit doppeltem Apostroph ('') geschlossen. Beispiel: “This is a correct direct citation in British English”. Der Punkt steht im englischen Zitat ausserhalb der Anführungszeichen (nicht im American English).

Mit dem Command \flqq erhält man das einleitende französische Anführungszeichen («). Mit \frqq bekommt man das Gegenstück (»).

11 Literatur

Calero Valdez, A., Brauner, P., Schaar, A. K., Holzinger, A., & Ziefle, M. (2015). Reducing complexity with simplicity-usability methods for Industry 4.0. *Proceedings 19th Triennial congress of the IEA*, *9*, 14 (siehe S. 23).

Hollan, J., Hutchins, E., & Kirsh, D. (2000). Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research. *ACM Trans. Comput.-Hum. Interact.*, *7*(2), 174–196. https://doi.org/10.1145/353485.353487 (siehe S. 23).

Nielsen, J., & Molich, R. (1990). Heuristic Evaluation of User Interfaces. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 249–256. https://doi.org/10.1145/97243.97281 (siehe S. 23).

Zobel, J. (2014). *Writing for Computer Science*. Springer London. https://doi.org/10.1007/978-1-4471-6639-9 (siehe S. 23).

Online Quellen

Fiorio, C. (2017, 18. Juli). *algorithm2e Package* [https://ctan.space-pro.be/tex-archive/macros/latex/contrib/algorithm2e/doc/algorithm2e.pdf, accessed on 2022-01-17] (5.3). (Siehe S. 22).

Inkscape Project. (2020, 16. April). *Inkscape* [https://inkscape.org, accessed on 2022-01-13] (0.92.5). (Siehe S. 20).

Institut für Multimediale und Interaktive Systeme [Zugriff am 29.09.2023]. (2023). https://web.archive.org/web/20230929110531/https://www.imis.uni-luebeck.de/de (siehe S. 24).

Overleaf c/o Digital Science. (n. d.). *Overleaf* [https://www.overleaf.com/project, accessed on 2022-01-13]. (Siehe S. 17).

Tantau, T. (2021, 15. Mai). *The TikZ and PGF Packages* [https://github.com/pgf-tikz/pgf, accessed on 2022-01-13] (3.1.9a). (Siehe S. 20).

Normen

Ergonomie der Mensch-System-Interaktion - Teil 220: Prozesse zur Ermöglichung, Durchführung und Bewertung menschzentrierter Gestaltung für interaktive Systeme in Hersteller- und Betreiberorganisationen (Standard). (2020, Juli). International Organization for Standardization. Geneva, CH. (Siehe S. 23).

A Anhang

Der Anhang kann Teile der Arbeit enthalten, die im Hauptteil zu weit führen würden, aber trotzdem für manche Leser interessant sein könnten. Das können z. B. die Ergebnisse weiterer Messungen sein, die im Hauptteil nicht betrachtet werden aber trotzdem durchgeführt wurden. Es ist ebenfalls möglich längere Codeabschnitte anzuhängen. Jedoch sollte der Anhang kein Ersatz für ein Repository sein und nicht einfach den gesamten Code enthalten.

B Abbildungsverzeichnis

C Tabellenverzeichnis

D Listings

Erklärung

Ich versichere an Eides statt, die vorliegende Arbeit selbstständig verfasst und nur die angegebenen Quellen benutzt zu haben.

Unterschrift

Lübeck, Tagesdatum