Abschlussprüfung zum

Fachinformatiker Fachrichtung Anwendungsentwicklung

Projektarbeit von

Max Mustermann

Entwicklung eines Markdown-zu-IHK-Konverters

Projektdokumentation zur Abschlussprüfung

Prüfungsperiode:	Sommer 2026
Ausbildungsbetrieb:	Musterfirma GmbH
Projektbetreuer:	Sabine Supervisor

Inhaltsverzeichnis

Inhalt **Seite**

Abkürzungsverzeichnis

Abkürzung	Bedeutung
IHK	Industrie- und Handelskammer
PDF	Portable Document Format
API	Application Programming Interface
AST	Abstract Syntax Tree
DIN	Deutsches Institut für Normung
YAML	YAML Ain't Markup Language

Vorwort

Dieses Projekt entstand im Rahmen der Abschlussprüfung zum Fachinformatiker Fachrichtung Anwendungsentwicklung. Es soll zeigen, dass technische Dokumentationen effizient und normgerecht erstellt werden können – ohne manuelle Nachbearbeitung in einer Textverarbeitung.

1. Problemstellung

1.1 Ausgangslage

Aktuell müssen IHK-Dokumentationen mühsam in Word formatiert werden, was fehleranfällig ist und viel Zeit kostet. Besonders die Einhaltung der Formvorgaben – Schriftgröße, Zeilenabstand und Seitenränder – erfordert manuelle Sorgfalt bei jedem Absatz. Änderungen am Inhalt erzwingen regelmäßig manuelle Korrekturen an der Formatierung.

1.2 Zielsetzung

Ziel ist ein Go-Tool, das **Markdown** in PDF umwandelt und dabei alle formalen Anforderungen der IHK Chemnitz automatisch erfüllt. Das Tool soll:

- die Prüfungsvorbereitung erleichtern,
- die Qualität der Dokumente *einheitlich* sicherstellen und
- den gesamten Formatierungsprozess vollständig automatisieren.

Besondere Anforderung ist die Einhaltung der **DIN 5008**-Norm sowie der spezifischen Vorgaben der IHK Chemnitz (Korrekturrand 4 cm rechts, Schriftart Arial/Helvetica 12 pt, 1½-zeiliger Abstand).

2. Projektablauf

2.1 Planung und Zeitrahmen

Die Planung umfasst die Analyse der IHK-Vorgaben und das Design der Software-Architektur. Der Projektzeitraum beträgt maximal 80 Stunden gemäß Ausbildungsverordnung.

Tab. 1: Projektphasen mit Zeitplanung

Phase	Aufgabe	Stunden
1	Anforderungsanalyse und Recherche der IHK-Vorgaben sowie DIN 5008 Normen	8
2	Architekturentwurf und Auswahl geeigneter Go-Bibliotheken (Goldmark, FPDF)	10
3	Implementierung des Markdown-Parsers mit AST-Traversierung	20
4	Implementierung des PDF-Renderers mit allen IHK-Elementen	24
5	Tests, Fehlerkorrektur und Dokumentation	12
6	Puffer und Abnahme	6

2.2 Architektur

Die Software ist in Go implementiert und folgt einer klaren Trennung zwischen Parsing und Rendering. Das Zwei-Pass-Verfahren ermöglicht ein korrektes Inhaltsverzeichnis mit Seitenzahlen.

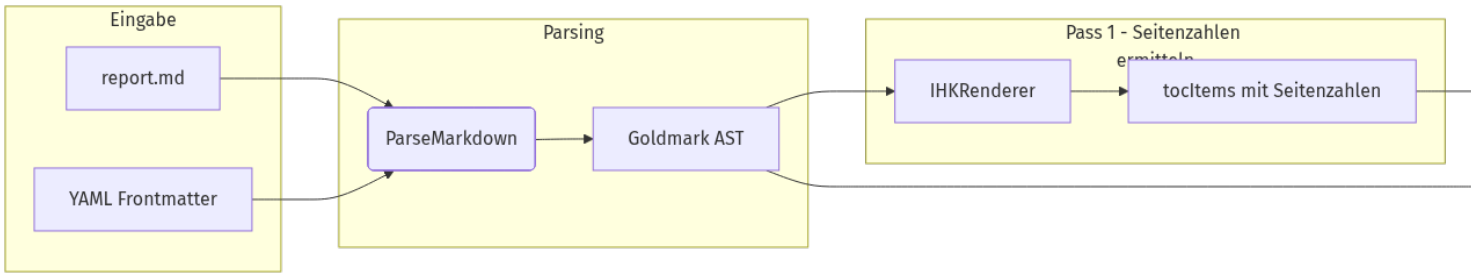


Abb. 1: Systemarchitektur des Konverters – Zwei-Pass-Rendering

Modulstruktur

Der Konverter ist in neun fokussierte Go-Dateien aufgeteilt:

- **`main.go`** – Einstiegspunkt, CLI-Flags, Zwei-Pass-Pipeline
- **`config.go`** – YAML-Konfigurationsstruktur
- **`markdown_parser.go`** – Goldmark-AST-Traversierung
- **`pdf_renderer.go`** – Kern-Struct `IHKRenderer`, DIN-5008-Konstanten
- **`pdf_content.go`** – Inhalts-Renderer (Absätze, Listen, Tabellen, Code, Bilder)
- **`pdf_toc.go`** – Verzeichnis-Renderer (Inhalts-, Tabellen-, Abbildungsverzeichnis)
- **`pdf_pages.go`** – Seiten-Renderer (Titelseite, Anhang, Erklärung, Glossar)
- **`pdf_numbering.go`** – Seitennummerierung (Römisch / Arabisch)
- **`diagram.go`** – Kroki-Diagramm-Rendering mit SHA-256-Cache

2.3 Realisierung

Die Realisierung erfolgt in Go unter Verwendung von `goldmark` für das Markdown-Parsing und `fpdf` für die PDF-Erzeugung.

Tab. 2: Eingesetzte Bibliotheken und Werkzeuge

Werkzeug	Zweck	Version	Lizenz
Go	Programmiersprache und Laufzeitumgebung	1.22+	BSD
Goldmark	Markdown-Parser mit CommonMark-konformem AST	v1.8	MIT
goldmark-meta	YAML-Frontmatter-Erweiterung für Goldmark	v1.1	MIT
FPDF	PDF-Erzeugung ohne externe Systemabhängigkeiten	v0.9	MIT

Tab. 2: Eingesetzte Bibliotheken und Werkzeuge (Fortsetzung)

Werkzeug	Zweck	Version	Lizenz
Kroki	Diagramm-Rendering für Mermaid und PlantUML	online	Apache 2.0

Implementierungsbeispiel: Tabellen-Renderer

Der folgende Ausschnitt zeigt die Kernlogik des mehrzeiligen Tabellen-Renderers. Jede Zelle wird mit `SplitLines` vorgemessen; anschließend werden alle Zellen einer Zeile auf einheitliche Höhe gebracht:

go

```
func (r *IHKRenderer) prepareRow(rawCells []string, numCols int,
    colW, lineHt float64, bold bool) tableRowData {

    r.pdf.SetFont("Helvetica", map[bool]string{true: "B", false: "
"}[bold], dinFontCaption)

    cells := make([][]string, numCols)
    maxLines := 0
    for j := 0; j < numCols; j++ {
        raw := ""
        if j < len(rawCells) {
            raw = rawCells[j]
        }
        split := r.pdf.SplitLines([]byte(r.tr(raw)), colW-2)
        lines := make([]string, len(split))
        for k, b := range split {
            lines[k] = string(b)
        }
        if len(lines) == 0 {
            lines = []string{""}
        }
        cells[j] = lines
        if len(lines) > maxLines {
            maxLines = len(lines)
        }
    }
    return tableRowData{cells: cells, height: float64(maxLines) *
lineHt}
}
```

2.4 Test und Qualitätssicherung

Das Tool wurde anhand dieses Musterdokuments getestet. Folgende Kriterien wurden geprüft:

1. Korrekte Seitenränder gemäß DIN 5008 (links 3 cm, rechts 4 cm Korrekturrand)
2. Seitennummerierung – römisch im Vorspann (ab II), arabisch im Textteil (ab 1)
3. Schriftart Helvetica, 12 Punkt, 1½-zeiliger Abstand (6,35 mm Zeilenhöhe)
4. Automatisch generierte Verzeichnisse (Inhalt, Tabellen, Abbildungen)
5. Mehrzeilige Tabellenzellen mit einheitlicher Zeilenhöhe pro Tabellenzeile
6. Nummerierende Code-Blöcke mit Zeilennummern-Gutter
7. Listen mit korrektem hängendem Einzug (zweite Zeile bündig mit Text)
8. Tabellen als Anhang über die Direktive `@TabelleAnhang:`

Tab. 3: Testergebnisse der Qualitätssicherung

Kriterium	Erwartet	Ergebnis	Status
Linker Seitenrand	30 mm	30 mm	Bestanden
Rechter Seitenrand (Korrekturrand)	40 mm	40 mm	Bestanden
Schriftgröße Fließtext	12 pt	12 pt	Bestanden
Zeilenabstand Fließtext	6,35 mm (1,5-fach)	6,35 mm	Bestanden
Römische Nummerierung Vorspann	ab Seite II	ab Seite II	Bestanden
Arabische Nummerierung Textteil	ab Seite 1	ab Seite 1	Bestanden
Mehrzeilige Tabellenzellen	einheitliche Zeilenhöhe	einheitliche Zeilenhöhe	Bestanden

3. Verwendung der Direktiven

3.1 Quellenangaben und Bibliografie

Quellenangaben werden mit der Direktive `@Quelle:` eingefügt und am Ende des Dokuments alphabetisch sortiert im Literaturverzeichnis ausgegeben. Die Direktive kann an beliebiger Stelle im Dokument stehen.

3.2 Benannte Tabellen

Mit `@Tabelle: Tabellenname` unmittelbar vor einer Markdown-Tabelle wird der Tabelle ein Name gegeben. Dieser erscheint im Tabellenverzeichnis:

```
@Tabelle: Übersicht der Programmiersprachen

| Sprache | Paradigma | Typsystem |
|---------|-----------|-----------|
| Go      | Imperativ | Statisch  |
| Python  | Multi     | Dynamisch |
```

3.3 Tabellen als Anhang

Mit `@TabelleAnhang: Anlagenname` wird die folgende Tabelle nicht im Fließtext gerendert, sondern als eigene Anlage im Anhang platziert:

```
@TabelleAnhang: Vollständige Fehlerliste

| Code | Beschreibung | Schwere |
|------|--------------|---------|
| E001 | Datei nicht gefunden | Kritisch |
```

3.4 Diagramme als Anhang

Mit `@AnhangUML:` gefolgt von einem Mermaid- oder PlantUML-Codeblock wird das Diagramm als Anlage eingebettet statt im Fließtext:

```
@AnhangUML: Datenbankschema

\`\`\`plantuml
@startuml
entity User { ... }
@enduml
\`\`\`
```

4. Zusammenfassung

Das Tool ermöglicht eine effiziente Erstellung von IHK-Dokumentationen unter Einhaltung **aller** Formatvorgaben. Durch die Trennung von Inhalt (Markdown) und Formatierung (Go-Renderer) ist eine konsistente Ausgabe garantiert. Künftige Erweiterungen könnten Fußnoten, Querverweise und eine konfigurierbare Spaltenbreite für Tabellen umfassen.

Literaturverzeichnis

– DIN 5008:2020-03, Schreib- und Gestaltungsregeln für die Textverarbeitung, Beuth Verlag

– Go-PDF/Fpdf Documentation, https://github.com/go-pdf/fpdf, 2025

– Goldmark Documentation, https://github.com/yuin/goldmark, 2024

– IHK Chemnitz, Hinweise zur Erarbeitung der Dokumentation über die Projektarbeit, 2020

Tabellenverzeichnis

Abbildungsverzeichnis

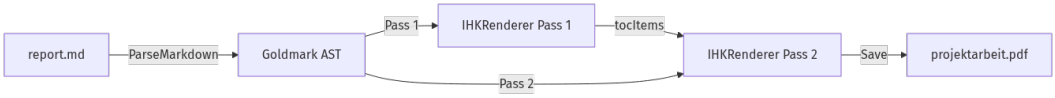
Anhang

Anlagenverzeichnis

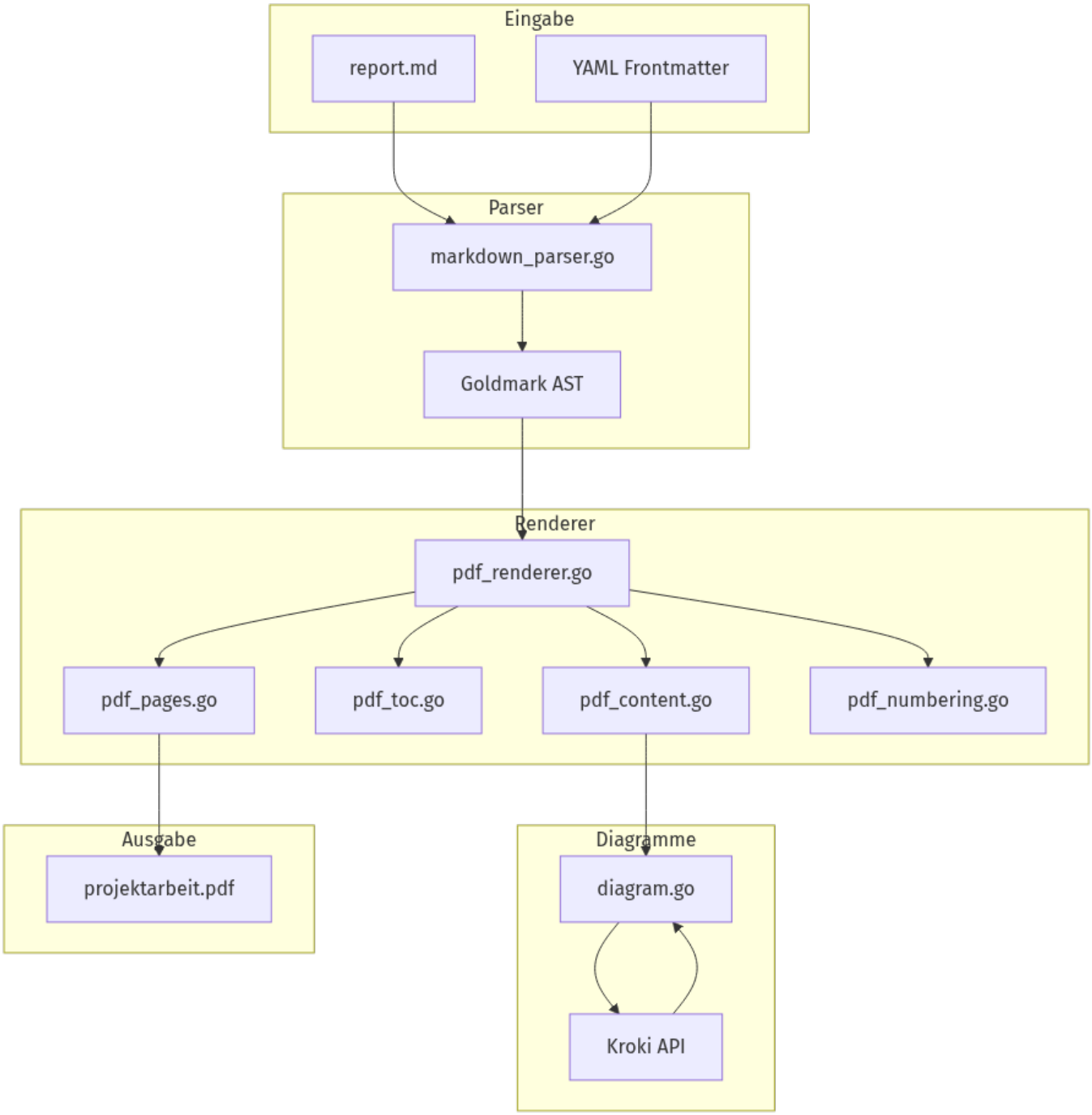
Anlage 1: Vollständige Liste der unterstützten Markdown-Direktiven

Direktive	Syntax	Beschreibung
Quelle	@Quelle: Text	Fügt einen Literaturverweis hinzu, der alphabetisch im Literaturverzeichnis erscheint
Tabelle	@Tabelle: Name	Gibt der folgenden Tabelle einen Namen für das Tabellenverzeichnis
Tabellenanhang	@TabelleAnhang: Name	Sendet die folgende Tabelle als nummerierte Anlage in den Anhang
Bildanhang	@Anhang: Titel | Pfad	Fügt ein Bild als nummerierte Anlage in den Anhang ein
Diagrammanhang	@AnhangUML: Titel	Rendert den folgenden Mermaid/PlantUML-Block als Anlage

Anlage 2: Systemarchitektur – Datenflussdiagramm



Anlage 3: Vollständige Modulübersicht – Querformat



Anlage 4: Vollständige Direktiven-Referenz im Querformat

Direktive	Syntax	Beschreibung	Querformat
Quelle	@Quelle: Text	Literaturverweis, alphabetisch sortiert im Literaturverzeichnis	–
Tabelle	@Tabelle: Name	Gibt der nächsten Tabelle einen Namen für das Tabellenverzeichnis	–
Tabellenanhang	@TabelleAnhang: Name	Sendet die nächste Tabelle als Anlage in den Anhang	–
Tabellenanhang Quer	@TabelleAnhangQuer: Name	Tabelle als Anlage im Querformat (297×210 mm, 15 mm Rand)	Ja
Bildanhang	@Anhang: Titel | Pfad	Bild als nummerierte Anlage (Hochformat)	–
Bildanhang Quer	@AnhangBildQuer: Titel | Pfad	Bild als nummerierte Anlage im Querformat	Ja
Diagrammanhang	@AnhangUML: Titel	Nächster Mermaid/PlantUML-Block als Anlage (Hochformat)	–
Diagrammanhang Quer	@AnhangUMLQuer: Titel	Nächster Mermaid/PlantUML-Block als Anlage im Querformat	Ja
Diagramm Querseite	@DiagrammQuer: Titel	Nächster Mermaid/PlantUML-Block inline als Querformat-Seite	–

Glossar

Goldmark
Ein in Go geschriebener Markdown-Parser, der den CommonMark-Standard implementiert und durch Erweiterungen (Tabellen, YAML-Frontmatter) ergänzt werden kann.

FPDF
Eine Go-Bibliothek zur Erzeugung von PDF-Dokumenten ohne externe Systemabhängigkeiten.

Kroki
Ein Webdienst, der verschiedene Diagramm-Beschreibungssprachen (Mermaid, PlantUML u.a.) serverseitig in Bilder umwandelt.

Zwei-Pass-Rendering
Verfahren, bei dem ein Dokument zweimal gerendert wird: Der erste Durchlauf ermittelt Seitenzahlen, der zweite nutzt diese für das Inhaltsverzeichnis.

Erklärung

Ich versichere durch meine Unterschrift, dass ich diese Projektarbeit mit dem Thema "Entwicklung eines Markdown-zu-IHK-Konverters" selbstständig, ohne fremde Hilfe angefertigt, alle Stellen, die ich wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen, als solche kenntlich gemacht und mich auch keiner anderen als der angegebenen Literatur oder sonstiger Hilfsmittel bedient habe. Die Projektarbeit hat in dieser oder ähnlicher Form weder der Industrie- und Handelskammer Chemnitz noch einer anderen Prüfungsinstitution vorgelegen.

Mir ist bekannt, dass gemäß der Prüfungsordnung für die Durchführung von Abschlussprüfungen der Industrie- und Handelskammer Chemnitz Täuschungshandlungen zum Ausschluss von der Prüfung führen können und die Prüfung als nicht bestanden erklärt werden kann.

Ort, Datum, Unterschrift (mit Vor- und Nachnamen)

_____