# Richtlinie zum „Software-Sprint“

# CIN – Eines KassenSichV 2020 kompatibles Open Source Kassensystem.

## *Schlussbericht*

Zuwendungsempfänger:

Lars Böttcher

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS22S03 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

## Kurze Darstellung der Aufgabenstellung und Motivation

Die grundlegende Idee des CIN-Projektes war die Schaffung eines modernen und professionellen Kassen- und Warenwirtschaftsökosystems in den Händen einer engagierten Community. Die Notwendigkeit dazu ergab sich mit dem Inkrafttreten der Kassensicherungsverordnung (KassenSichV), welche momentan von keinem Open Source Kassensystem erfüllt wird. Die Kassensoftware sollte plattformunabhängig und auf dem Desktop und mobilen Geräten lauffähig sein.

Für das Kassensystem sollte eine Client-Server-Architektur eingesetzt werden. Der Server selbst liefert und empfängt die vom Kassensystem benötigten bzw. erzeugten Daten und speichert diese in einer zentralen Datenbank. Die Kommunikation zwischen den Clients und dem Server erfolgt über eine RESTful-JSON-API.

Die Verkaufstransaktionen sollten in einer Technischen Sicherheitseinheit TSE protokolliert werden und für die Finanzbehörde exportiert werden können.

Im Rahmen der Förderung wurde ein Prototyp des Frontends der Kassen inklusive TSE-Anbindung, die RESTful-API und das webbasierte Backend des Kassenservers implementiert.

Die drei Hauptaufgaben für das Kassensystem waren die Implementierung eines modernen User Interfaces, die Anbindung der Technischen Sicherheitseinrichtung TSE und die Umsetzung der Plattformunabhängigkeit.

Die serverseitigen Aufgabenblöcke bestanden aus der Implementierung der JSON-API, der Anbindung der Datenbank, sowie der Entwurfs des Datenbanken-Schemas und die Entwicklung des webbasierten Backends.

## Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Die Kassensicherungsverordnung (KassenSichV) schreibt vor, dass alle in Deutschland eingesetzten Kassensysteme seit dem 30.08.2020 eine aktivierte Technische Sicherheitseinrichtung (TSE) besitzen müssen. Die TSE sorgt für eine manipulationssichere Aufzeichnung aller Kassenvorgänge und verhindert eine nachträgliche Bearbeitung der Kassendaten. Bis zum jetzigen Zeitpunkt existiert kein quelloffenes Kassensystem, welches eine Technische Sicherheitseinrichtung integriert hat. Mit Inkrafttreten der Kassensicherungsverordnung standen alle Unternehmen, die auf freie Software und freie Betriebssysteme setzten, vor großen Problemen. Es mussten proprietäre Kassensysteme angeschafft und eingeführt werden. Für Linux existieren aber keine bezahlbaren Kassensysteme, die den in den meisten Fällen benötigen Funktionsumfang bieten. Viele kleine Geschäfte und Restaurants sind zu Cloud-Lösungen gewechselt und geben seit dem ihre sensiblen Geschäftsdaten an die Cloud-Betreiber*innen weiter. Wer auf freie Betriebssysteme setzte und seine Daten nicht in fremde Hände geben wollte, musste auf proprietäre Betriebssystem wechseln. Damit ging oftmals ein größerer Umbau der IT-Struktur einher, aber noch viel entscheidender war das Abwenden von den Prinzipien freier Software und den eigenen Überzeugungen.

## Ausführliche Darstellung der Ergebnisse

Im Projektzeitraum wurden alle wesentlichen Meilensteine adressiert und ein funktionsfähiger Prototyp fertiggestellt. Dieser wurden auf der Abschlussveranstaltung, dem DemoDay, präsentiert. Im Rahmen der Entwicklung des webbasierten Backends ist das MXXN Framework entstanden. Dabei handelt es sich um ein Framework zur einfachen und schnellen Entwicklung von modernen Webanwendungen und/oder RESTful-JSON-API. Das Framework vereint eine gut überlegte Auswahl an Software-Komponenten, welche aller unter eine freien Lizenz stehen, weit verbreitet sind und sehr gut gepflegt und weiterentwickelt werden. Als Datenbank-ORM wurde auf SQLAlchemy gesetzt. Zur Versionierung des Datenbankschemas wurde Alembic verwendet. Das zugrunde liegende Webframework ist die asynchrone API vom Falcon. Mit all diesen Komponenten gab es keine größeren Probleme während der

Entwicklungsphase. Im Bereich des JavaScript-Frameworks zur Entwicklung des Web-Frontend wurde anfänglich auf RiotJS gesetzt. Leider hatte dieses Frontend nicht den benötigten Funktionsumfang und nach kurzer Zeit wurde die Entscheidung zum Wechsel auf Lit und TypeScript getroffen. Mit dieser Kombination konnten alle Ideen umgesetzt werden. Das Framework enthält ein Kommandozeilenprogramm das bei der Entwicklung von Anwendungen und des Datenbankschemas hilft. Nahezu alle Elemente des Frameworks können überladen und an die eigenen Bedürfnisse angepasst werden (Resources, URLs, Strings, Themes etc.).

Beim zweiten großen Meilenstein handelt es sich um die Anbindung der Technischen Sicherheitseinrichtung TSE. Hier wurde im Antrage die Verwendung der TSE der Bundesdruckerei genannt. Leider war es mir nicht möglich eine Entwicklerversion dieser zu erwerben/bekommen. Aus diesem Grunde wurde im Rahmen des Projektes auf die TSE der Firma Epson gesetzt, was sich als eine gute Wahl herausstellte. Es wurden das Python-Module tse.py entwickelt, welches sich im Python Package Index PyPI befindet und von anderen Projekten verwendet werden kann. Das Paket ist so konzipiert, dass es zukünftig auch TSE-Module anderer Hersteller als Backend nutzen kann. Es wurde der gesamte für die KassenSichV 2020 benötigten Funktionsumfang implementiert und mit den benötigen Test versehen. Des Weiteren wurden Langzeittests für den Mehrkassenbetrieb geschrieben.

Für das Kassen-Frontend, als dritter Meilenstein, wurden die grundlegenden Funktionen einer Kasse implementiert. Hierzu wurde die TSE , welche die Verkaufstransaktionen aufzeichnet, angebunden. Die Transaktionen können für die Steuerbehörde exportiert und archiviert werden. Die Kasse selbst ist unter den Betriebssystemen Linux, Windows und Android lauffähig. Sie bezieht die Produktdaten vom Backendserver und cached diese indiziert in einer lokalen Datenbank, was zu einer schnellen und effizienten  Produktsuche führt. Das Kassen-Interface wurde für die Benutzung von Touchscreens optimiert. Es wurde auch die Möglichkeit der Verwendung von Barcodelesern implementiert. Wären der Entwicklung zeigte sich, dass das verwendete GUI-Toolkit auf mobilen Geräten etwas zu träge ist. Momentan arbeite ich an der Portierung auf ein leichtgewichtigeres und somit schnelleres Toolkit.

Die Unterstützung  bei der Projektdurchführung durch die Open Knowledge Foundation empfand ich als nahezu perfekt. Sie waren immer ansprechbar und konnten mit ihrer langjährigen Erfahrung immer mit Rat zur Seite stehen. Auch die Unterstützung durch den Projektträger war optimal.

Eine der wichtigsten Erkenntnisse aus der Projektarbeit ist, dass es immer Fehlentscheidungen gibt und man nicht allzu lange an Workarounds arbeiten sollte, sondern sich einfach eingestehen kann, dass die erste Ideen vielleicht einfach nicht die richtige war.

## Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Mein Prototyp zeigt die Funktionsweise und die Machbarkeit meiner Idee. Im einem kritischen Bereich wie der Warenwirtschaft führt jeder Fehler und/oder der Ausfall einer Kasse zu großen Problemen und natürlich kann dann nichts verkauft werden. Hier muss die Software einen hohen Qualitätsstandard erfüllen und maximal getestet sein. Auch bedarf es eines hohen Funktionsumfangs, den meine Implementierung momentan noch nicht enthält. Ich werde weiter am Projekt arbeiten und eine kleine Gruppe von Menschen ist schon dazugestoßen. Es wurden zwei neue Projekt auf der Basis des MXXN-Frameworks gestartet und ich bin zuversichtlich, dass die Kassen die Produktreife erreicht. Das TSE-Paket wurden in letzten Monat ca. 100 mal heruntergeladen und wird somit wahrscheinlich auch schon von weiteren Entwicklern eingesetzt.

Im Rahmen des Projektes habe ich mir einige neue Technologien erschlossen und mich fachlich stetig weiterentwickelt und danke den Fördertopf für diese Möglichkeit.

## Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Wie oben bereits erwähnt war es mir nicht möglich eine Technische Sicherheitseinrichtung der Bundesdruckerei zu bekommen und somit wechselte ich zu einer Lösung der Firma Epson.

Außerdem hatte das ausgewählte JavaScript-Framework leider nicht den benötigen Funktionsumfang und ich tauschte es gegen ein leistungsstärkstes aus

## Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Die jeweiligen Teile des Projektes können unter folgenden Links einsehen werden:

MXXN-Framework:

Gitlab: https://gitlab.com/ccodein/mxxn/mxxn
Website: https://ccodein.gitlab.io/mxxn/mxxn/

TSE-Paket:

Gitlab: https://gitlab.com/ccodein/tse.py
Website: https://ccodein.gitlab.io/tse.py
PyPI: https://pypi.org/project/tse.py/

Kassen-Frontend:

Gitlab: https://gitlab.com/ccodein/cin/cin
Gitlab: https://gitlab.com/ccodein/cin/cin-mobile

## Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Bei der Entwicklung des TSE-Pakets kam es zu einem erheblichen Mehraufwand. Die Dokumentation der Herstellers war mehr als nur unzureichend und somit musste ich mir die Funktionsweise der TSE und vor allem die Fehlerbehandlung größtenteils durch ausprobieren erschließen. Da die TSE nach den meisten Fehler zurückgesetzt bzw. neu gestartet werden musste, bedeutete dies einen hohen Mehraufwand an Zeit, mit dem ich nicht rechnete. Diese Zeit fehlte dann in anderen Bereichen des Projektes. Auch der Austausch des JavaScript-Frameworks bedeutete ein deutlichen Mehraufwand.

## Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Im Rahmen des Projektes habe ich das MXXN-Framework beworben und an zwei Wochenenden Workshops gegeben. Bei diesen Workshops haben wir uns ausgetauscht und über künftige Features und die Zusammenarbeit gesprochen.