RFID Card-Management System
Pflichtenheft

Diplomanden:
Johannes Mayrhofer, Patrick Grubauer, Benedikt Zöchmann

Betreuungslehrkraft:
Prof. Dr. Susanne Hofer

Jahrgang: 2022/23

17. September 2022
18. September 2022
7. Oktober 2022

Inhaltsverzeichnis

1 Use Case **2**

2 Kriterien **2**
2.1 Fertigstellungszeitraum . . . 2
2.2 Zentrale API . . . 2
2.3 API – Raspberry Kommunikation . . . 2
2.4 Login/Registrierungsseite . . . 2
2.5 Client Anwendung / Login . . . 2
2.6 Admin Anwendung / Login . . . 3

3 Teilbereiche **4**
3.1 Zentrale API . . . 4
3.1.1 Hardware . . . 4
3.1.2 Sofware . . . 4
3.1.3 Zugriff als Client . . . 4
3.1.4 Zugriff als Card-Storage Controller . . . 4
3.1.5 Datenbank . . . 4
3.1.6 Zusammenfassung . . . 4
3.2 Anwendung . . . 5
3.2.1 Login/Registrierungsseite . . . 5
3.2.2 Client Anwendung / Login . . . 5
3.2.3 Admin Anwendung / Login . . . 5
3.2.4 Hardware . . . 6
3.3 API - Raspberry Kommunikation . . . 7
3.3.1 Raspberry . . . 7
3.3.2 Display . . . 7
3.3.3 NFC Reader . . . 7

4 Anwendungsbereiche / Zielgruppe **7**
4.1 Benutzer . . . 7
4.2 Einsatz Ort . . . 7

1 Use Case

Kann hier nicht angeziegt werden, da das Use Case zu groß ist: hier.

2 Kriterien

2.1 Fertigstellungszeitraum

Spätestens bis 31.3.2023

2.2 Zentrale API

Die Zentrale API (siehe 3.1) erfüllt folgende Kriterien:

- Anwendungsprogrammierer sollen über die API Benutzer, Karten, und Schließfächer anlegen, abrufen, ändern und löschen können
- Anwendungsprogrammierer sollen die Möglichkeit haben über die API Karten als reserviert und wieder zurückgegeben zu melden
- Anwendungsprogrammierer sollen Daten zur Statistischen Auswertung durch den Administrator abrufen können (Häufigkeiten von Reservierungen von Karten etc.)
- Anwendungsprogrammierer sollen Daten zu Informationszwecken für den Administrator abrufen können (API-Logs: Zugriffe von Clients, etc.)
- Die API soll als Softwarepaket in Form eines portablen Docker-Containers zur Verfügung stehen
- Die API unterscheidet zwischen Administrator und Benutzer
- Administratoraccounts sind vordefiniert und werden in der Datenbank gespeichert

2.3 API – Raspberry Kommunikation

Die Kommunikation zwischen Zentraler API und Raspberry (siehe 3.3) erfüllt folgende Kriterien

- Anwendungsprogrammierer tauschen etwaige Daten mit der API über MQTT aus
- Anwendungsprogrammierer tauschen etwaige Daten mit dem Python Program aus, welche die Motoren des Tresor und den Karten Leser Steuert.

2.4 Login/Registrierungsseite

- Der User soll sich mittels seinem Microsoft Account einloggen
- Der User soll die Moglichkeit haben seinen Login zu speichern (Remember me)
- Der User soll beim Login die Möglichekeit haben Fragen an den Admin zu stellen

2.5 Client Anwendung / Login

- Die Client-Anwendung soll ein Reservierungssystem für die Karten bieten
  - Reservierungen sollen bearbeitbar/löschbar sein
  - Reservierungen sollen eine Benachrichtigung bei einem bevorstehenden Termin versenden

- Die Client-Anwendung soll einen Shortcut bieten um Karten direkt zu holen (ohne Anmeldung am Terminal)
- Die Client-Anwendung soll die Möglichkeit bieten bestimmte Karte zu favorisieren
- Die Client-Anwendung soll eine Benachrichtigung versenden falls eine markierte Karte wieder frei geworden ist.
- Die Client-Anwendung soll eine Einstellungsseite bieten (Benachrichtigungen, Thememode, usw.)
- Die Client-Anwendung soll ein System zum Chatten mit anderen Lehrern bieten falls eine Karte verwendet die sie benötigen
- Möglichkeit Nachrichten an den Admin zu senden (Feedback, Bug-report, Verbesserungen [...])

2.6 Admin Anwendung / Login

Die Anforderungen an die Admin App gestalten sich folgendermaßen:

- Der Admin soll Karten anlegen / löschen / bearbeiten können.
- Der Admin soll Karten Tresore anlegen / löschen / bearbeiten können.
- Der Admin soll Benutzer anlegen / löschen / bearbeiten können.
- Der Admin soll sich Statistiken anzeigen lassen können.
- Der Admin soll sich Logs anzeigen lassen können.

3 Teilbereiche

3.1 Zentrale API

Die API ist der zentrale Knotenpunkt zum Austausch von Daten zwischen Client Anwendung (Flutter App) und den Card-Storage[1]s. Die API kommuniziert mit den Card-Storages mithilfe MQTT welche jeweils ein Topic zugewiesen bekommen und so Daten senden als auch empfangen können. Die API speichert etwaige Daten in einer Datenbank. Diese Daten können ausschließlich von der API verwaltet werden. Gehostet wird diese auf einem Zentralen Server sodass der Zugriff auf die ebenfalls zentralisierte Datenbank vereinfacht wird und alle Daten synchron und konsistent miteinanander sind.

3.1.1 Hardware

Docker-fähiger Server mit Netzwerkzugang und zwei offenen TCP-Ports für HTTP und MQTT.

3.1.2 Sofware

Die oben beschriebene API wird mithilfe der Programmiersprache Go umgesetzt. SQLite wird als relationale Datenbank eingesetzt. Der Card-Storage Controller wird ebenfalls in Go umgesetzt, wobei dieser mit dem Modul der Kollegen kommuniziert welches die Kommunikation zwischen dem Raspberry Pi und dem Mikrocontroller, welcher die Hardware regelt, erledigt.

3.1.3 Zugriff als Client

Die Kommunikation zwischen der API und den Clients findet im Rahmen des HTTP-Protkolls mithilfe der REST (Representaional State Transfer) Architektur statt. Die Clients senden HTTP-Kommandos um Daten zu Karten, Schließfächern bzw. Benutzern, lesen, erstellen, ändern sowie löschen zu können. Um zwischen einfachen Benutzern und Administratoren zu unterscheiden sendet der Administrator einen speziellen HTTP-Header welchen die API überprüft, um dem Administrator privilegierte Daten übermitteln zu können.

3.1.4 Zugriff als Card-Storage Controller[2]

Die Card-Storage Einheiten kommunizieren mit der API mithilfe dem MQTT Protokoll. Das Controller Programm welches auf dem Raspberry (bzw. ein anderer Mikroprozessor) der Card-Storage läuft, kommuniziert mit der API, um die Kommandos welche die API von der Client bzw. Admin Anwendung erhält, auszuführen. Da API und Card-Storage, Daten möglicherweise bidirektional austauschen müssen, wäre hier das klassische Client-Server Modell unpassend. Die API weist jedem Card-Storage ein Topic zu, über welches gemeinsam kommuniziert werden kann.

3.1.5 Datenbank

Jegliche Daten, die speichernswert sind, werden in der zentralen Datenbank, welche ausschließlich über die oben erläuterte API zugänglich ist, abgelegt, um diese zu einem späteren Zeitpunkt entweder von einem Administrator, einem einfachen Benutzer des Systems, oder einem Card-Storage Controller wieder abzurufen.

3.1.6 Zusammenfassung

Die API ist das Herzstück des Systems. Sowohl die Client- und Admin-Anwendung als auch die Card-Storage Einheiten kommunizieren damit. Die API ist, wenn man so will, der Übersetzer zwischen den beiden. Sie übersetzt einerseits die Kommandos der Client- und Admin-Anwendung in MQTT-Befehle welche dann vom Card-Storage Controller ausgewertet und an die Schaltung weitergegeben werden. Andererseits verwaltet die API auch den Zugriff zur Datenbank, in welcher alle Card-Storages, Benutzer sowie Karten hinterlegt sind.

[1] Schließfach, in dem die RFID Karten aufbewahrt werden

[2] Mikroprozessor (Raspberry Pi) auf welchem die Software zur Verwaltung des Schließfaches läuft

3.2 Anwendung

Als Grundlage zur adminstrativen als auch benutzerspezifizischen Verwaltung, kommt eine Flutter App zur Anwendung. Wir haben uns für dieses Framework entschieden, da es auf verschiedenen Plattformen bzw. Betriebssystem mit einer Version der Software läuft (Native Framework). Um zu dem Admin-Login zu gelangen, wird ein eigener Login bei der Login-Seite benötigt. D.h Admin und User Anwendung teilen sich ab den Login auf.

3.2.1 Login/Registrierungsseite

Der Login dient dazu, um zu der Admin bzw. User Anmeldung hinzugelangen.

Funktionen

- Anmeldung eines Lehrers via Microsoft
- "Remember me"→ Funktion
- Quick Login/Registrierung Möglichkeit am Terminal mittels Rfid Karte der Lehrer

3.2.2 Client Anwendung / Login

Generelle Spezifikation
Der User Login wird von den Lehrenden verwendet, um Karten zu beantragen, reservieren, anzufordern [...]. Weiters soll der Login am Display als auch im Web zur Verfügung gestellt, damit wirklich jeder Zugang zum Automaten hat.

Software

- Die Startseite wird dazu verwendeten seine favorisierten Karten anzeigen zu lassen
- Es soll eine Seite mit allen Karten angezeigt werden, wobei man die angezeigten Karten, anhand verschiedener Kriterien filtern bzw. suchen kann
- Eine eigene Seite zum Reservierungssystem
- Quick Card Get Möglichkeit mittels Pop-Up am Display.
- Es soll eine Benachrichtigung versendet werden falls eine markierte bereits verwendete Karten wieder frei ist.
- Die Client-Anwendung soll ein System zum Chatten mit anderen Lehrern bieten falls eine Karte verwendet wird die sie benötigen
- Möglichkeit Nachrichten an den Admin zu senden (Feedback, Bug-report, Verbesserungen [...])

3.2.3 Admin Anwendung / Login

Generelle Spezifikation
Der Admin Login soll dazu verwendet werden können, direkt am Display am Automaten, administrative Aufgaben erledigen zu können. Diese Anwendung soll auch als Webseite und APP zur Verfügung stehen, um solche Einstellungen nicht direkt Vorort machen zu müssen. Die Admin App wird ebenfalls mit Flutter erstellt. Um die Admin App benutzen können, wird es bestimmte Benutzer geben die sich als Admin anmelden können.

- In der soll der Administrator sich Statistiken zu allen Karten anzeigen lassen können. Das soll dazu dienen, das der Admin erkennen kann ob eine bestimmte Karte merfach benötigt wird.

- Weiters soll man in der App einen neuen Automaten hinzufügen können, dazu muss man einige Parameter angeben wie Name, IP-Adresse, Anzahl Karten, Ort. Danach werden diese in einer Datenbank im Hintergrund eingepflegt. Es besteht ebenfalls die Option, Automaten zu bearbeiten und zu löschen.
- In der App sollen auch neue Karten hinzugefügt, gelöscht, bearbeitet werden können oder Karten getauscht werden, dazu wird, wenn das Fenster geöffnet wird am Automaten angezeigt, dass die Karten zum Lesegerät gehalten werden soll, damit die Karte im System gespeichert werden kann.
- Weiters soll sich der Admin Logs zu der Datenbank und API anzeigen lassen können. Das soll dazu dienen, wenn Fehler auftreten, diese leicht erkennen und beheben zu können.

Die App soll folgende Funktion bieten:

- Statistiken anzeigen
- Neuen Automaten anlegen / löschen / bearbeiten
- Neue Karten hinzufügen / löschen / bearbeiten
- Logs anzeigen

3.2.4 Hardware

Da ein wir ein crossplattform Framework verwenden, haben wir die Möglichkeit mit wenig Arbeit unsere Produkt auf allen Geräten anzubieten. Hier eine Auflistung auf welchen Geräten die Hardware laufen muss

- Android Geräte größer API 24
- Smartphones mit Betriebssystemen wie IOS,Android
- Geräte die einen Browser Bieten

3.3 API - Raspberry Kommunikation

Das Programm, welches am Mikroprozessor läuft und einerseits die Kommandos, beispielsweise die Schließfachtür per App öffnen, interpretiert und ausführt aber auch andererseits Daten an die API, wenn diese danach fragt, beispielsweise die aktuelle Anzahl der vorhandenen Schlüsselkarten im Schließfach, zurücksendet. Das Card-Storage Controller Programm kommuniziert mittels MQTT mit der API. Dieses wird mithilfe der Programmiersprache Go implementiert. Jeder Card-Storage Controller bekommt ein MQTT-Topic zugewiesen, womit dieser mit dem Broker (Zentrale API) kommunizieren kann. Um, wie oben erwähnt, Funktionen bereitstellen zu können die Hardware beinhaltet, welche von den Mechatronikern enwtickelt wird, beispielsweise die aktuelle Anzahl der vorhandenen Schlüsselkarten im Schließfach, muss der Cardstorage-Controller auch mit dem Mikrocontroller kommunizieren, welcher eben diese Hardware über die Serielle Schnittstelle steuert. **Sobald ein Konzept seitens der Mechatroniker zur Verfügung steht, (und ob die oben als Beispiel genannten Funktionen überhaupt von den Mechatronikern implementiert werden) werden Einzelheiten weiter erläutert.**

3.3.1 Raspberry

Der Raspberry wird dazu verwendet die Daten von der API per MQTT zu erhalten (siehe 3.3). Dieser wir ein Python Program laufen haben, diese dient dazu die Motgoren im Storage zu steueren. Weiters werden die Daten vom NFC Reader an die API gesendet. Eine weitere Aufgabe des Raspberry ist, unsere App darzustellen. Weitere Information bei 3.2.4 Hardware.

3.3.2 Display

Das Display wird dazu verwendet, die Admin und Client App darzustellen. Diese werden auf dem Raspberry im Browser laufen, und müssen etwas angepasst werden, da gewisse funktionalität der App im Browser nicht gewährleistet ist. Das Display wird per proprietärer Raspberry Schnittstelle angschlossen.

3.3.3 NFC Reader

Der NFC Reader wird dazu verwendet, User beim ersten mal per Chip zu authentifizieren. Dies wird dann in die Datenbank gespeichert. Eine weitere Aufgabe ist es, beim zurückgeben die Karten zu Scannen und bei dem jeweiligen User herauszulöschen. Damit ist gewährleistet, das die Karte auch von anderen Benutzer verwendet werden kann.

4 Anwendungsbereiche / Zielgruppe

4.1 Benutzer

Unser komplettes System ist für die Verwendung am Linzer Technikum gedacht. Diese wird dort von den Lehrern und Administratoren verwendet. Die Lehrer haben die Möglichkeit, sich Karten auszuborgen und zu reservieren. Administratoren können das komplette System verwalten. Administratoren sind fix vergebene Benutzer können sich bei der Client Anwendung nicht anmelden.

4.2 Einsatz Ort

Das System wird am Linzer Technikum eingesetzt. Dort werden im Gebäude mehrere Karten Tresore platziert. In jedem Karten Tresor befinden sich jeweils 10 Karten. Diese Karten Tresore werden nicht von uns gefertigt.

Abbildungsverzeichnis