Summchester - Executive Summary

Zielsetzung  
Elektromagnetische Summer werden oft mit nervigen Geräuschen assoziiert, die hohen und lauten Tönen werden häufig in Signal- oder Warnanlagen verbaut und sollen durch ihre Eindringlichkeit die Aufmerksamkeit auf bestimmte Vorgänge/Zustände lenken. Ziel dieses Projektes ist es, mit diesen Mustern zu brechen und mit den Summern eine auditiv harmonische Anwendung zu bauen.

Projektbeschreibung  
Das Summchester ist als einfaches Musikerstellungs-Tool konzipiert, man kann Töne aus zwei Oktaven in ein Lied aus beliebig vielen Takten setzen. Dabei stehen einem 8 Stimmen zur Verfügung, die gleichzeitig ihre Inhalte wiedergeben. Man kann das Lied anhalten, an eine beliebige Stelle springen und dort weiterarbeiten, Takte hinzufügen oder löschen und in dem Lied scrollen, um lange Lieder auf dem Bildschirm darstellen zu können. Außerdem kann man die Widergabegeschwindigkeit ändern.  
Über das Kartenterminal können RFID-Karten mit bestimmten Texten benutzt werden um erstelle Lieder lokal zu speichern und über die Karten auch wieder aufzurufen. So ist eine Speicherung der Lieder möglich.  
Zusätzlich sind zwei Servo-Motoren angeschlossen, einer bewegt sich im Takt des Liedes (die dafür relevante Taktanzahl kann man ändern), der andere zeigt an, an welcher Stelle man sich gerade im Lied befindet.  
Das ganze wird über ein Programmfenster verwaltet, wo alles übersichtlich dargestellt wird um ein grafisches Arbeiten zu ermöglichen.

Inhalte der Abgabe  
Im ‚Design‘-Ordner sind die Sprites abgelegt, die für die Benutzeroberfläche benötigt werden.  
Im ‚Dokumentation‘-Ordner ist dieser Text, ein Pdf des Schaltplans und ein paar Fotos zur Übersicht abgelegt.  
Im ‚Mains‘-Ordner befindet sich der eigentliche Code. Der Hauptcode heißt ‚v9.py‘, die restlichen Skripte werden zur Hilfe über diesen Code aufgerufen. Der gesamte Code ist kommentiert und beschreibt die Abläufe. In ‚Mains/Lieder‘ werden die eigentlichen Lieder als ‚[name].irk‘ gespeichert. Wie die Datei entsteht und zu lesen ist, ist in ‚saveHandler.py‘ beschrieben.

Inspiration und Quellen  
Es gibt viele Programme, die so oder so ähnlich funktionieren. Ich persönlich benutze Bosca Ceoil sehr häufig und würde sagen, dieses Programm hat mein Projekt am meisten inspiriert.  
Was Quellen angeht habe ich das Pygames-Wiki genutzt, um mich mit den Methoden vertraut zu machen, für das Kartenlesegerät habe ich die Dokumentation für die Lese-Methode genutzt und für die Servo-Motoren den Code aus den Übungsfolien.   
Darüber hinaus ist eine Methode komplett von ChatGPT geschrieben, nämlich ‚openFrom(name)‘ in saveHandler.py. Im Code ist das auch kenntlich gemacht und die Herangehensweise dokumentiert.  
Darüber hinaus konnte alles mit Python-Vorkenntnissen selbst programmiert werden.