In den vergangenen dreieinhalb Monaten der Programmier-Challenge habe ich eine Vielzahl von Aufgaben erfolgreich bewältigt. Einer meiner Haupttätigkeiten bestand darin, wöchentliche Scrum-Meetings zu leiten, bei denen ich stets als Protokollant fungierte. Parallel dazu habe ich einen erheblichen Teil meiner Zeit auf die Entwicklung eines Algorithmus konzentriert. Wie bereits in meinem Zwischenbericht erwähnt, habe ich verschiedene Ansätze verfolgt, die ich an dieser Stelle nicht erneut auflisten möchte.

Seit dem Zwischenbericht hat sich der Algorithmus jedoch weiterentwickelt. Ich konnte feststellen, dass das intensive Schärfen eines Bildes den Score stärker beeinflusst, als ich zuvor angenommen hatte. Daher habe ich entschieden, die Schärfung trotz ihrer teilweise starken bildverändernden Eigenschaften in meinen Algorithmus zu integrieren. Gleichzeitig habe ich die Anzahl der angewendeten Effekte reduziert, da eine zu hohe Anzahl eher negative als positive Auswirkungen auf den Score hatte.

Den Hivemoderation.com Bild-KI-Detektor konnte ich nicht mehr überlisten, weil dieses mit den Projektanforderung kollidiert. Denn es gibt eine Methode wie man diesen KI-Detektor austricksen kann. Die [Idee](https://imgur.com/a/08eDCKB) dahinter ist, ein Bild sehr stark zu verkleinern und komprimieren, sodass das ursprüngliche Rauschmuster nicht mehr zu erkennen ist. Da aber durch diesen Schritt das Originalbild fast nicht mehr zu erkennen ist, benutzt man ControlNET um dieses Bild wieder hochzuskalieren. Da dieser Aktion aber mehr als 5 Sekunden benötigt, hat es für mich keinen Sinn gemacht, mich damit weiter zu beschäftigen.

Mit der Entwicklung des Algorithmus habe ich zwei Drittel der Projektzeit verbracht.

Im letzten Drittel unsere Projektzeit habe ich mich darauf konzentriert, eine Browser-Erweiterung für unser Python-Skript zu entwickeln. Um sicherzustellen, dass der Algorithmus im Browser nicht ständig geändert werden muss, habe ich nach einer Methode gesucht, mit der eine Browser-Erweiterung effektiv mit einem Python-Skript kommunizieren kann. Hierbei bin ich auf den Native Messaging-Channel gestoßen, der mir die benötigte Schnittstelle bereitstellte. Die Implementierung gestaltete sich zunächst als holprig, da im Internet veraltete Manifestdateien und falsche Pfade für Native Messaging-Skripte im Umlauf waren. Dennoch konnte ich letztendlich eine erfolgreiche Kommunikation mit unserem Skript herstellen, sodass die Browser-Erweiterung nun problemlos Bild- oder Textbearbeitung ermöglicht.

Zum Abschluss des Projektes war es notwendig, eine kleine Marktanalyse und Übersicht über Bild-KI-Detektoren zu verfassen. Dies bot sich an, da ich während der Entwicklung des Bildalgorithmus kontinuierlich mit solchen Detektoren gearbeitet habe. Die Analyse beinhaltete auch eine Erklärung der Funktionsweise solcher Detektoren. Abschließend habe ich die Evaluation und Vorgehensweise für unseren Bildalgorithmus dokumentiert.