Technische Universität Braunschweig

IAS | INSTITUTE FOR APPLICATION SECURITY

Software-Entwicklungspraktiktum (SEP)

ARID - Augmented Reality in Disguise

Software-Entwicklungspraktikum (SEP)
Sommersemester 2024

Pflichtenheft

Auftraggeber:
Technische Universität Braunschweig
Institut für Anwendungssicherheit (IAS)
Prof. Dr. Martin Johns
Mühlenpfordstraße 23
38106 Braunschweig

Betreuerin: Alexandra Dirksen

Auftragnehmer:

Name	E-Mail-Adresse
Amir Fakhim Hashemi	a.fakhim-hashemi@tu-braunschweig.de
Ibrahim Abdullah	i.abdullah@tu-braunschweig.de
Jadon-Kim Fischer	jadon-kim.fischer@tu-braunschweig.de
Mohamed Ali Mrabti	m.mrabti@tu-braunschweig.de
Tim Kütemeyer	t.kuetemeyer@tu-braunschweig.de
Vyvy Nguyen	Vyvy.nguyen@tu-braunschweig.de

Braunschweig, 28. Mai 2024

Bearbeiterübersicht

Kapitel	Autoren	Kommentare
1	Tim	—
2	Tim	—
3	Vyvy	—
4	Jadon	—
5	Ibrahim	—
6	Amir	—
7	Jadon, Mohamed Ali	—
8	Mohamed Ali	—
9	Sämtliche	Fortlaufend ergänzt

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Zielbestimmung

Das Projekt ARID - Augmented Reality in Disguise ist im Rahmen unseres Software- Entwicklungspraktikums (SEP) angesiedelt und beschäftigt sich mit der Kombination aus Steganographie und moderner Augmented-Reality-Technologie. Ziel ist es, eine Kommunikationsmethode zu schaffen, bei der Nachrichten unauffällig innerhalb von Bildern versteckt werden. Diese Nachrichten können nur von Personen mit Zugriff auf spezialisierte AR-Hardware, dem Monocle von Brilliant Labs, wahrgenommen und entschlüsselt werden. Das Verstecken der Kommunikation gewährleistet, dass unbefugter Zugriff auf die übertragenen Informationen verhindert wird, während autorisierte Nutzer uneingeschränkten Zugang haben.

Das primäre Ziel dieses Systems ist die Entwicklung einer Plattform, die es erlaubt, verschlüsselte Nachrichten in visuellen Medien zu erkennen und zu entschlüsseln. Die Kombination von Kryptographie und Steganographie fügt eine zusätzliche Schutzebene hinzu, indem sie die Existenz der Nachrichten selbst verbirgt. Unser Ansatz nutzt künstliche Intelligenz, um Bilder zu generieren, in die QR-Codes mit verschlüsselten Nachrichten eingebettet sind. Diese Bilder fungieren als Trägermedium für eine Botschaft, die nach Möglichkeit nur durch das Monocle, als solches erkannt werden sollen.

Die Entwicklung dieses Systems ist von der Notwendigkeit inspiriert, in einer vernetzten und immer stärker überwachten Welt individuelle Kommunikation zu schützen. In Kontexten, in denen staatliche Überwachung und Zensur politische Botschaften einschränken können, bietet unser Projekt einen Weg, um durch fortschrittliche Technologien wie Bildverarbeitung, Kryptographie und AR neue Kommunikationskanäle zu schaffen.

1.1 Musskriterien

Hier wird dargelegt, welche Funktionalitäten und Leistungen unser Produkt unbedingt bieten muss, um genutzt werden zu können.

$\langle RM1 \rangle$ Erkennung: Das Monocle muss QR Codes in Bildern erkennen können.
$\langle RM2 \rangle$ Dekodierung: Die Software muss die QR Codes dekodieren können.
$\langle RM3 \rangle$ Verschlüsselung: Software zum Erstellen verschlüsselter QR Codes.
$\langle RM4 \rangle$ Die QR Codes sollen in Kunst eingebettet sein.

1.2 Sollkriterien

Die folgenden Sollkriterien sind erstrebenswert, um die Benutzerfreundlichkeit und Funktionalität des Systems zu erhöhen:

⟨$RS1$⟩ Erkennen, wenn ein QR Code gescannt wurde, aber nicht entschlüsselt werden kann.
⟨$RS2$⟩ Die entschlüsselten Nachrichten sollten diskret und ansprechend dargestellt werden.
⟨$RS3$⟩ Die Aufnahme der Bilder sollte im Rahmen der begrenzten Kamera möglichst Robust und zuverlässig gestaltet werden.

1.3 Kannkriterien

Folgende Kannkriterien könnten implementiert werden, sofern genügend Ressourcen vorhanden sind:

⟨$RC1$⟩ QR Code ist nicht für das bloße Auge als solches erkennbar.

1.4 Abgrenzungskriterien

Bestimmte Funktionen werden bewusst aus dem Umfang des Projekts ausgeschlossen:

⟨$RW1$⟩ Das System wird keine QR Codes im Video erkennen, da es dem Monocle an der technischen Möglichkeit fehlt.
⟨$RW2$⟩ Keine Unterstützung für andere AR-Brillen: Das Projekt ist speziell auf das Monocle-Gerät zugeschnitten und unterstützt keine anderen AR-Hardware-Plattformen.
⟨$RW3$⟩ Kein automatisches Update-System: Es wird kein automatisches Update-System für die Software implementiert.

2 Produkteinsatz

In diesem Abschnitt werden die verschiedenen Einsatzbereiche des ARID-Systems erörtert. Dazu gehören die Lebensbereiche, in denen das Projekt genutzt werden kann, die Zielgruppen, die von dem System profitieren werden, sowie die Rahmenbedingungen, die für eine effektive Nutzung des Produkts notwendig sind.

2.1 Anwendungsbereiche

Das ARID-System ist für den Einsatz in Umgebungen gedacht, in denen verdeckte Botschaften übermittelt werden sollen. Dies betrifft nicht nur politische oder journalistische Anwendungsbereiche, sondern erstreckt sich auch auf Ausstellungen, wo subtile Botschaften eine Rolle spielen können:

- **Politische Kommunikation:** Aktivisten können das System nutzen, um unauffällig Botschaften zu kommunizieren, ohne die Gefahr, überwacht oder zensiert zu werden. Das ARID-System ermöglicht es ihnen, Botschaften innerhalb von Kunstwerken oder interaktiven Installationen nur für berechtigte Personen sichtbar zu machen.
- **Journalismus:** Investigative Journalisten, die in sensiblen politischen Umfeldern arbeiten, können Informationen sicher austauschen. In Ausstellungen oder Medieninstallationen können journalistische Inhalte so dargestellt werden, dass sie nur für ein eingeweihtes Publikum erkennbar sind.
- **Kulturelle Ausstellungen:** Museen, Galerien oder öffentliche Plätze, die interaktive Ausstellungen oder Veranstaltungen durchführen, können das ARID-System nutzen, um verborgene Botschaften in ihren Displays zu integrieren. Besucher, die mit dem nötigen Wissen oder Technologien ausgestattet sind, können diese Botschaften entschlüsseln, was eine zusätzliche Ebene der Interaktion und des Engagements schaffen kann.
- **Künstlerische Performances:** Künstler können das ARID-System als Teil ihrer Performance verwenden um zusätzliche Botschaften zu übermitteln. Diese Art der Nutzung betont die Rolle der Technologie in der Kunst.

Die Vielseitigkeit des ARID-Systems in verschiedenen öffentlichen und privaten Kontexten zeigt, wie Technologie eingesetzt werden kann, um Kommunikation sicherer und interaktiver zu gestalten, als auch als künstlerisches Element zu verwenden. Die Fähigkeit, Botschaften verdeckt zu übermitteln, eröffnet neue Möglichkeiten für kreative und politische Ausdrucksformen.

Für diese Anwendungen bietet das ARID-System eine technologische Grundlage, die bereits im kleinen Maßstab funktioniert und bei Bedarf für größere und komplexere Szenarien angepasst werden kann.

2.2 Zielgruppen

Basierend auf den erweiterten Anwendungsbereichen des ARID-Systems richtet sich unser Produkt an eine breite Palette von Zielgruppen, die von der Möglichkeit, versteckte Kommunikation zu nutzen, profitieren können:

- **Menschenrechtler:** Personen, die in Umgebungen arbeiten, in denen die Kommunikationsfreiheit eingeschränkt ist, können versteckte Nachrichten nutzen, um sicher zu kommunizieren, insbesondere in autoritären Regimen oder in Gebieten mit starker Zensur.
- **Journalisten und Medienorganisationen:** Diese Gruppe umfasst investigative Journalisten und Medienhäuser, die in politisch sensiblen Bereichen tätig sind und einen neuen Kommunikationskanal für die Verbreitung von Nachrichten benötigen, die versteckt sein sollen.
- **Forschungseinrichtungen und Universitäten:** Akademische Institutionen, die sich mit digitaler Sicherheit und Verschlüsselungstechnologien beschäftigen, können das ARID-System für Forschungs- und Lehrzwecke verwenden, insbesondere um praktische Erfahrungen in der Anwendung von Steganographie und sicheren Kommunikationsmethoden zu vermitteln.
- **Kuratoren und Künstler in Museen und Galerien:** Diese Gruppe kann das ARID-System nutzen, um interaktive, kulturelle Ausstellungen zu gestalten, in denen Besucher eingeladen werden, verborgene Botschaften zu entdecken, die in Kunstwerken oder Installationen integriert sind.
- **Organisatoren von kulturellen Veranstaltungen:** Veranstalter, die thematische Ausstellungen, politische Versammlungen oder kulturelle Festivals organisieren, können das ARID-System einsetzen, um spezielle Inhalte auf neue Art und Weise präsentieren um Diskussionen anzuregen.

2.3 Betriebsbedingungen

Die effektive Nutzung des ARID-Systems erfordert spezielle Betriebsbedingungen, die an die einzigartigen Anforderungen seiner Anwendung angepasst sind:

- **Physikalische Umgebung:** Das System ist für den Einsatz in hellen Umgebungen konzipiert. Eine Nutzung in dunklen Umgebungen ist aufgrund der Hardware nicht möglich. Es eignet sich ideal für den Einsatz in Ausstellungshallen, Museen oder Konferenzräumen.
- **Betriebszeit:** Im Gegensatz zu einem Dauerbetrieb ist das ARID-System für zeitlich begrenzte Einsätze vorgesehen, entsprechend der Haltbarkeit des Akkus.
- **Überwachung:** Obwohl das System nicht kontinuierlich in Betrieb ist, wird eine technische Überwachung während der Nutzungszeiten empfohlen, um die Integrität und Sicherheit des Systems zu gewährleisten. Dies beinhaltet regelmäßige Wartungen und Updates durch Fachpersonal, insbesondere vor und nach größeren Einsatzzeiten. Zusätzlich sollten Betreiber darauf vorbereitet sein, schnell auf technische Probleme reagieren zu können, um die Funktionalität während kritischer Betriebszeiten sicherzustellen.

3 Produktübersicht

Im folgendem Kapitel wird die Produktübersicht beschrieben. Die Funktionalitäten des Produktes wird anhand eines Use-Case-Diagramm graphisch dargestellt, welcher zusätzlich noch beschrieben wird. Außerdem wird ein Aktivitätsdiagramm für ein Use-Case angefertigt.

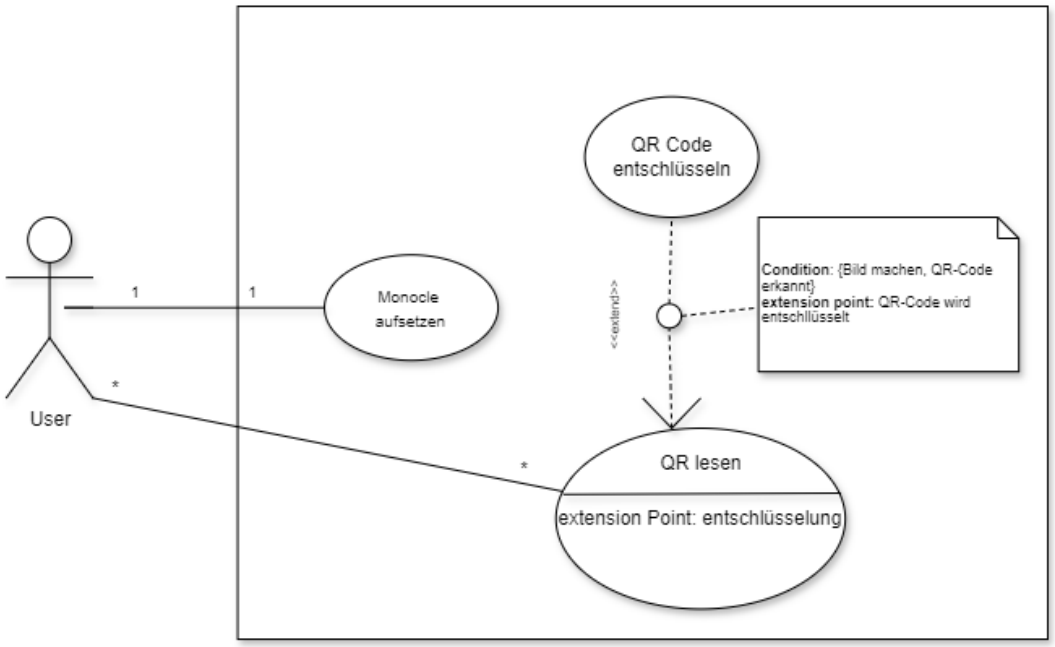

Abbildung 3.1: Use-Case-Diagramm Monocle

Anmerkungen zu Abbildung 3.1 Use-Case-Diagramm Monocle:

Das Diagramm zeigt den User, der den Monocle trägt. Dieser Akteur (Benutzer) setzt sich das Gerät auf. Wird dabei ein QR-Code gesichtet, wird dieser gelesen. Wenn ein QR-Code verschlüsselt sein sollte, dann kann dieser entschlüsselt werden.

Das Use-Case 'QR-Code lesen' wird als Aktivitätsdiagramm dargestellt:

Im Aktivitätsdiagramm des Use-Case 'QR-Code lesen' wird zunächst das Gerät vom Verwender eingeschaltet. Wenn der Träger ein Bild sieht, kann das Bild über die Kamera des Monocle-Gerät fotografiert. Dabei sucht das Endegrät den versteckten QR-Code und wenn dieser erkannt wird,

enthält der Verwender eine verschlüsselte Nachricht. Es kann jedoch passieren, dass kein QR-Code erkannt wird, dann kann der Verwender das Bild erneut fotografieren. Wenn der Verwender alle Bilder gescanned hat und fertig ist kann er das System beenden.

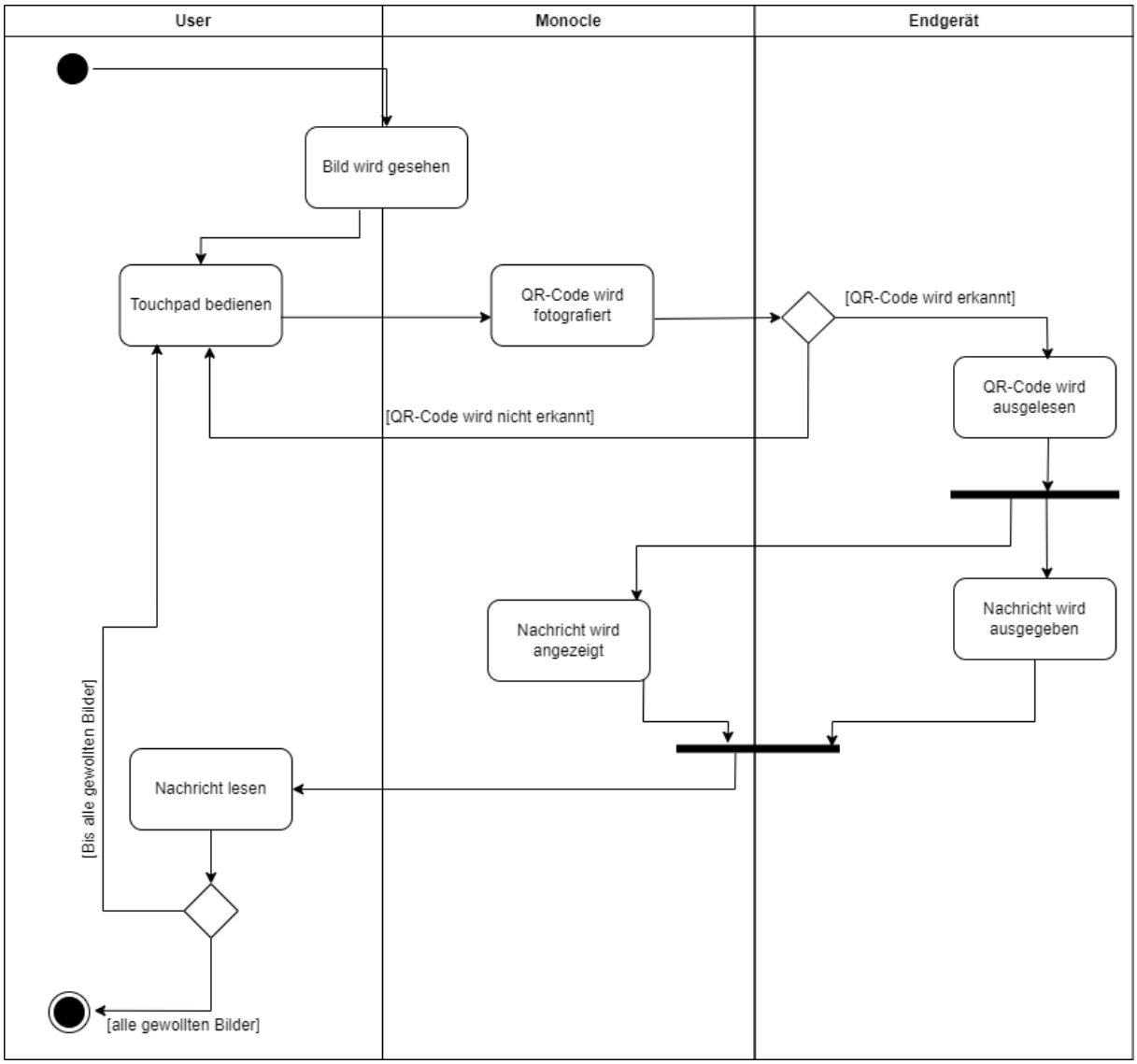

Abbildung 3.2: Aktivitätsdiagramm zu "QR-Code lesen

4 Produktfunktionen

Im Folgenden befindet sich eine detaillierte Übersicht der direkt und indirekt erreichbaren Funktionen der Software.

Ein- und Ausschalten des Monocle $\langle F10 \rangle$

Anwendungsfall: Allgemeine Nutzung des Geräts

Anforderung: Das Monocle-System soll sich unmittelbar vor der Verwendung und unmittelbar danach selbstständig ein- und ausschalten können.

Ziel: Der Träger des Geräts muss sich nicht manuell um die Aktivierung und Deaktivierung des Geräts kümmern.

Vorbedingung: Das Gerät muss per Sensor erfahren können, ob es sich in der dafür vorgesehenen Hülle befindet oder hinausgenommen wurde.

Nachbedingung Erfolg: Das Monocle ist wenige Augenblicke nach dem Herausnehmen oder Eingeben in die Hülle vollständig hoch- beziehungsweise heruntergefahren und die Lampe außen am Prozessor ist aktiv. Im Display ist der Text "Monocleërkennbar.

Nachbedingung Fehlschlag: Obwohl das Monocle-System nicht mehr in dem dafür vorgesehenen Gehäuse befindlich ist, fährt sich das Gerät nicht oder nicht vollständig hoch und ist damit für den Träger des Geräts nicht nutzbar.

Akteure: Monocle, Träger

Auslösendes Ereignis: Der Träger entnimmt das mit einem Sensor ausgestattete Monocle-Gerät dem Gehäuse, wodurch der Zustandswechsel dem Prozessor des Monocle als Auslöser des Hochfahrens dient.

Beschreibung:

1. Das Gerät wird dem Gehäuse durch den Träger entnommen oder hineingegeben.
2. Der Prozessor des Monocle erkennt per Sensor den Zustandswechsel und startet die restlichen Hardware-Komponenten.
3. Sobald jede Komponente hochgefahren ist und dem Prozessor dies meldet, aktiviert sich das Monocle und ist für den Träger im Sinne von $\langle RC1 \rangle$ einsetzbar.

Erweiterung: Keine

Alternativen: Keine

Kommunikation zwischen Endgerät und Monocle $\langle F20 \rangle$

Anwendungsfall: Starten der Software, Einlesen von QR-Codes in $\langle LF40 \rangle$

Anforderung: Das Monocle-System soll Daten mit dem Host per Bluetooth austauschen können.

Ziel: Das Endgerät empfängt das eingelesene Bild der Kamera des Monocle und führt die entsprechende Erkennung und Entschlüsselung des QR-Codes aus.

Vorbedingung: Das Gerät muss sich in einer Reichweite von 10 Metern zum Monocle aufhalten, um die leistungsschwache Bluetooth Low Energy-Verbindung aufrecht halten zu können. Der Verbindungsaufbau beim Start der Software muss erfolgreich gewesen sein.

Nachbedingung Erfolg: Das Host-Gerät zeigt nach dem Start der Software keine Fehlermeldung an und am Ende des Einlesevorgangs in $\langle LF40 \rangle$ ist die von dem Host-Gerät entschlüsselte Botschaft auf dem Display des Monocle-Geräts zu erkennen.

Nachbedingung Fehlschlag: Die Bluetooth-Verbindung zwischen Host und Monocle konnte nicht aufrecht erhalten oder aufgebaut werden. Das Monocle-Gerät ändert seinen Zustand nicht zu 7.3, da es vergeblich auf die Rückmeldung des Hosts wartet.

Akteure: Monocle, Host, Träger

Auslösendes Ereignis: Der Träger startet die Verbindung durch das Ausführen der Software oder berührt eine der oben liegenden Schaltflächen zum Einlesen eines QR-Codes.

Beschreibung:

1. Das Gerät wurde erfolgreich hochgefahren, gemäß $\langle F10 \rangle$ und die Software gestartet.
2. Der Host fragt die Bluetooth-Verbindung bei dem Monocle an, welches diese erwidert.
3. Der Träger initialisiert über seine Berührung der Touchpads an der oberen Kante des Monocle-Geräts den Einlesevorgang eines QR-Codes.

Erweiterung: 2a Bei dem Host-Gerät handelt es sich um ein Mobilgerät, wodurch die maximale Reichweite von 10 Metern zuverlässig unterschritten wird und die Verbindung beschleunigt im Sinne von $\langle RS3 \rangle$.

Alternativen: 2a Bei dem Host-Gerät handelt es sich um ein Laptop oder ähnliches Gerät, wodurch der Träger des Monocle sich nicht zu weit von eben diesem Gerät entfernen darf, oder dieses umständlich mitzuführen hat.

Start des Einlesevorgangs per Touchpad $\langle F30 \rangle$

Anwendungsfall: Einlesen eines QR-Codes über Funktion $\langle LF40 \rangle$

Anforderung: Die Nutzung des Monocle-Geräts soll unauffällig und der Einlesevorgang zügig initialisiert werden können.

Ziel: Das Monocle-System soll dem Träger den Einlesevorgang über die oben am Brillenglas liegenden Touchpads ermöglichen.

Vorbedingung: Das Gerät muss vollständig hochgefahren und die Software zum Einlesen von QR-Codes ausgeführt werden. Der Träger berührt im Zustand 7.1 eines der Touchpads.

Nachbedingung Erfolg: Der Träger berührt eine der beiden Touchpads des Monocle-Systems, welches daraufhin in 7.2 wechselt, was dem Träger den Einleseversuch eines QR-Codes signalisiert.

Nachbedingung Fehlschlag: Das Monocle reagiert trotz Berührung der oben liegenden Touchpads nicht und die UI kommt in 7.2 oder 7.3 zum Stehen.

Akteure: Monocle, Träger

Auslösendes Ereignis: Von mindestens einem der Touch Controllern wird eine Eingabe festgestellt, die dem Prozessor daraufhin elektronisch mitgeteilt wird.

Beschreibung:

1. Das Gerät wurde erfolgreich hochgefahren, gemäß $\langle F10 \rangle$ und die Software gestartet.
2. Der Träger berührt eines der oberhalb des Brillenglases liegenden Schaltflächen.
3. Die auf dem Display des Monocle erkennbare Benutzeroberfläche wechselt in Zustand 7.2
4. Der Einlesevorgang beginnt wie in $\langle F40 \rangle$ beschrieben.

Erweiterung: Keine

Alternativen: Keine

Einlesen von QR-Codes $\langle F40 \rangle$

Anwendungsfall: Einsehen einer versteckten Botschaft

Anforderung: $\langle LF10 \rangle$ Das Monocle-System soll in der Lage sein, einen in einem Kunstwerk versteckten QR-Code auszulesen und dessen Inhalt über eine UI in dem eingebauten Display sichtbar zu machen ($\langle RM1 \rangle$).

Ziel: Der dafür vorgesehene Text in der UI zeigt dem Träger des Monocle-Geräts die in $\langle F20 \rangle$ entschlüsselte Botschaft in dem Kunstwerk an.

Vorbedingung: Die Software des Geräts muss in der Lage sein, den QR-Code anhand seiner charakteristischen Merkmale innerhalb des Kunstwerks ausfindig zu machen.

Nachbedingung Erfolg: Das in Funktion $\langle F20 \rangle$ beschriebene Host-Gerät empfängt die durch die Kamera des Monocle aufgezeichnete Foto-Aufnahme. Das Host-Gerät beginnt mit der in $\langle F50 \rangle$ erläuterten Entschlüsselung der Botschaft.

Nachbedingung Fehlschlag: Der QR-Code kann nicht als solcher erkannt werden, etwa weil während des Fotos mittels integrierter Kamera eine zu schnelle Bewegung stattgefunden hat. Nach dem fehlgeschlagenen Einlesen wird dem Träger deshalb die Meldung 7.4angezeigt. Dieser hat nun die Möglichkeit, über das unten beschriebene Ereignis einen weiteren Versuch zu unternehmen.

Akteure: Monocle, Träger

Auslösendes Ereignis: Der Träger der Monocle-Geräts berührt eine der beiden oberhalb des Brillenglases liegenden Schaltflächen im Sinne von $\langle F30 \rangle$.

Beschreibung:

1. Aufnahme eines Fotos von dem Kunstwerk mithilfe der integrierten Kamera.
2. Das aufgenommene Foto algorithmisch nach dem versteckten QR-Code absuchen.
3. Mittels Auslese des QR-Codes die verschlüsselte Botschaft erhalten.
4. Die verschlüsselte Botschaft über die entsprechende Funktion ($\langle F50 \rangle$) entschlüsseln.
5. Dem Träger die entschlüsselte Botschaft über 7.3 zugänglich machen, oder, die Fehlermeldung 7.4 anzeigen, falls kein QR-Code gefunden wurde.

Erweiterung: 1a In Abhängigkeit der Geschwindigkeit der Kamera und dessen Akku-Leistung Bilder auf kontinuierliche Art und Weise aufnehmen. Auf diese Weise ist der Träger des Geräts nicht mehr auf die Berührung der oben liegenden Schaltflächen angewiesen und kann die geheime Botschaft so im Sinne von Kriterium $\langle RM1 \rangle$ unauffälliger auslesen.

Alternativen: 1a Im Falle von zu ungenauer Aufnahmen oder zu hohem Akku-Verbrauch auf die oben liegenden Schaltflächen zurückfallen und vorerst auf die automatische Einlese des QR-Codes verzichten.

Entschlüsseln von QR-Codes $\langle F50 \rangle$

Anwendungsfall: Während des Einlesens des QR-Codes ($\langle F40 \rangle$)

Anforderung: Das Monocle-System soll in der Lage sein, die in $\langle F40 \rangle$ ausgelesene Botschaft zu entschlüsseln, sodass sie nur durch eben dieses Monocle eingesehen werden kann.

Ziel: Niemand anderes soll in der Lage sein, die Botschaft als solche zu identifizieren. Auf diese Weise lassen sich sensible Informationen zwischen den Zielpersonen transferieren ohne Überwachung jeglicher Art zum Opfer zu fallen. Selbst wenn dritten der QR-Code im Bild in die Hände fällt, stellt sich dieser für die Finder als nutzlos heraus ($\langle RM2 \rangle$).

Vorbedingung: Die Software des Geräts muss im Besitz des für die Entschlüsselung erforderlichen Schlüssels sein. Des weiteren muss das Ergebnis der Entschlüsselung eindeutig sein, um den vollwertigen Austausch von Informationen zu ermöglichen.

Nachbedingung Erfolg: Die Nutzeroberfläche 7.3 zeigt dem Träger die entschlüsselte Nachricht an, obwohl in den QR-Code selbst nur dessen verschlüsselte Variante eingebettet war.

Nachbedingung Fehlschlag: Die erfolgreich ausgelesene Botschaft des QR-Codes konnte nicht oder nicht korrekt entschlüsselt werden. Dem Träger wird nun über 7.3 die verschlüsselte Botschaft selbst angezeigt oder eine fehlerhaft entschlüsselte Nachricht, die der Träger für die versteckte Botschaft gemäß $\langle RM1 \rangle$ halten könnte.

Akteure: Träger

Auslösendes Ereignis: Der Einlesevorgang des Monocle-Geräts ($\langle F40 \rangle$) wurde durch eine der oben liegenden Tasten initialisiert und war erfolgreich.

Beschreibung:

1. Die verschlüsselte Botschaft aus Funktion $\langle F40 \rangle$ entgegennehmen.
2. Unter Zuhilfenahme des geheimen Schlüssels die verschlüsselte Botschaft in die von dem Träger ersuchte Form bringen.
3. Die durch die Entschlüsselung erhaltene Botschaft zurück an die Funktion $\langle F40 \rangle$ übergeben, wo sie dem Träger präsentiert wird.

Erweiterung: 3a Im Falle einer mehrdeutigen Entschlüsselung eine speziell dafür vorbereitete UI anzeigen, sodass dem Träger die Ungenauigkeit seiner Botschaft verdeutlicht wird und der Informationsaustausch nach Kriterium $\langle RM2 \rangle$ genauer ablaufen kann.

Alternativen: 3a Falls wegen unverhältnismäßiger Kosten keine Überprüfung der Exaktheit der Entschlüsselung durchgeführt werden kann, auf eine Risikobewertung setzen und den Kunden gegebenenfalls über die Ungenauigkeit der Entschlüsselung in einigen Fällen aufklären.

Anzeigen einer Nachricht auf dem Display $\langle F60 \rangle$

Anwendungsfall: Während der gesamten Nutzung

Anforderung: Das Monocle-System soll dem Träger die in $\langle F30 \rangle$ angeforderte Botschaft als Text-Nachricht unauffällig zugänglich machen.

Ziel: Der Träger des Monocle ist in der Lage, sich den verschlüsselten Inhalt der QR-Codes in seiner Umgebung gemäß $\langle RS2 \rangle$ anzeigen zu lassen.

Vorbedingung: Die Kommunikation zwischen Endgerät und Monocle muss während des Einlesevorgangs reibungslos ablaufen, sodass dem Monocle-Gerät die entschlüsselte Botschaft von dem Host-Gerät übermittelt werden kann.

Nachbedingung Erfolg: Die Benutzeroberfläche 7.3 zeigt dem Träger die entschlüsselte Nachricht auf dem in das Monocle-System eingebauten OLED Display im unteren Drittel des Brillenglases an.

Nachbedingung Fehlschlag: Die Software des Monocle-Geräts ist nicht in der Lage von der 7.2 in die 7.3 zu wechseln oder die Anzeige auf dem Display des Monocle erlischt vollständig.

Akteure: Monocle, Host, Träger

Auslösendes Ereignis: Die anzuzeigende Nachricht wurde dem QR-Code erfolgreich entnommen und mittels Funktion $\langle F50 \rangle$ entschlüsselt von dem Host-Gerät an das Monocle per Bluetooth-Kommunikation übertragen.

Beschreibung:

1. Der Träger startet die Software des Monocle, wodurch die Meldung 7.1 angezeigt wird.
2. Durch die erfolgreiche Initialisierung des Einlesevorgangs eines QR-Codes in $\langle F30 \rangle$ wechselt das Gerät zur Benutzeroberfläche 7.2.
3. Bei erfolgreicher Bluetooth-Kommunikation zwischen Host und Monocle wird die entschlüsselte Botschaft vom Host an das Monocle zurückgemeldet, welches diese mittels 7.3 dem Träger des Monocle zugänglich macht.

Erweiterung: 3a Sollte die Bluetooth-Verbindung zwischen Host und Monocle erst beim Anzeigen der finalen Botschaft aus dem QR-Code scheitern, kann die entschlüsselte Botschaft dennoch in der Konsole des Host-Geräts eingesehen werden.

Alternativen: 3a Sollte bei der Rückmeldung der Botschaft vom Host-Gerät an das Monocle eine zeitliche Toleranz überschritten werden, kann dem Träger des Monocle-Systems eine Fehlermeldung im Bezug auf Zeitüberschreitung präsentiert werden worauf das Monocle-Gerät nach einigen Augenblicken in den initialen Zustand 7.1 zurückkehrt.

Nutzung der integrierten Kamera $\langle F70 \rangle$

Anwendungsfall: Zu Beginn des Einlesevorgangs aus Funktion $\langle F40 \rangle$

Anforderung: Das Monocle-System muss in der Lage sein Informationen über seine Umgebung unter Berücksichtigung von Kriterium $\langle RS3 \rangle$ aufzuzeichnen und zu verarbeiten.

Ziel: Der Träger ist es möglich in KI-Kunst eingebettete QR-Codes unter Zuhilfenahme des Monocle-Geräts zu erkennen und mit diesem weiterzuverarbeiten.

Vorbedingung: Sämtliche Hardware-Komponenten des Monocle müssen beim Hochfahren des Geräts erfolgreich von dessen Prozessor gestartet worden sein. Die Monocle-Software muss auf dem Gerät ausgeführt werden.

Nachbedingung Erfolg: Auf dem Display des Monocle-Systems wird während der Foto-Aufnahme die Nachricht „Capturing...“ eingeblendet, die bei Abschluss der Aufnahme automatisch zur Benutzeroberfläche 7.2 übergeht.

Nachbedingung Fehlschlag: Die Anfrage des Trägers kann nicht als solche erkannt werden, wodurch die Meldung „Capturing...“ für den Träger ausbleibt. Die Benutzeroberfläche bleibt unverhältnismäßig lange in dem besagten Zustand, wodurch der Übergang in 7.2 scheitert oder stark verzögert wird.

Akteure: Monocle, Träger

Auslösendes Ereignis: Der Träger des Monocle-Geräts berührt eine der oben liegenden Schaltflächen, wodurch es zu Beginn des Einlesevorgangs $\langle F40 \rangle$ zu einer Bild-Aufnahme der im Monocle integrierten Kamera kommt.

Beschreibung:

1. Der Träger berührt bei hochgefahrenen Monocle und ausgeführter Software eine der Touchpads.
2. Die Software des Geräts interpretiert die Berührung als Anfrage des Trägers einen in der Kamera des Monocle sichtbaren QR-Code einzulesen und ruft die ensprechende Funktion der Kamera auf
3. Während die Bild-Aufnahme von der Kamera durchgeführt wird, bestätigt der Schriftzug „Capturing...“ dem Träger den Eingang seiner Anfrage.
4. Sobald in der Software die interne Rückmeldung der Kamera vorliegt, ist dessen Aufnahme beendet und die von der Kamera aufgezeichneten Informationen können aus dem dafür vorgesehenen Zwischenspeicher entnommen werden. Dem Nutzer wird die Beendigung dieses Schritts mit der Benutzeroberfläche 7.2 kenntlich gemacht.

Erweiterung: 4a Die Erfolgschance der $\langle F70 \rangle$ unmittelbar folgenden Funktionalität $\langle F40 \rangle$ kann gesteigert werden, indem unmittelbar nach der Auslesung der Foto-Aufnahme aus dem Zwischenspeicher, das heißt vor der Erkennung des QR-Codes auf dem Host-Gerät eine Bild-Optimierung stattfindet. Diese kann die durch die technische Limitierung des Monocle-Systems eingeschränkte Qualität der Bild-Aufnahmen nachträglich korrigieren.

Alternativen: 4a Alternativ kann durch eine Steigerung der Umgebungsverhältnisse gleichermaßen eine Steigerung der Qualität der Bild-Aufnahmen sichergestellt werden, etwa durch optimierte Lichtverhältnisse im konkreten Anwendungsfall.
4b Sollte auf die Implementierung der Erweiterung 4a verzichtet werden und die Alternative 4a eine nicht ausreichend zufriedenstellende Verbesserung hervorrufen, ist der Wechsel hin zu einem alternativen Algorithmus zur Erkennung des in der Bild-Aufnahme befindlichen QR-Codes möglich. Verfügbare algorithmische Ansätze wurden von den Auftragnehmern bereits im Vorfeld ausgemacht.

5 Produktdaten

Das Monocle-System verfügt über einen begrenzten Speicher, deswegen kann es nur notwendige Daten aus QR-Code kurzfristig vorhalten. Danach werden entsprechende Daten an einem Host-Gerät übermittelt, um weiterzuverarbeiten. Im Folgenden werden weitere Informationen zu diesen Prozess bereitgestellt.

Monocle-System $\langle D10 \rangle$
Diese Liste bietet eine Übersicht über die verschiedenen Datentypen, die das System speichert:
- Display-Puffer,
- Kameradaten,
- Arbeitsdaten für Algorithmen,
- Protokolldaten zur Erfassung von Systemaktivitäten,
- FPGA* Bitstreams,
- Daten aus QR-Code-Scans.

Host-Gerät $\langle D20 \rangle$
Im Host-Gerät können die folgenden Arten von Daten vom Monocle-System gespeichert werden:
- Verarbeitete und analysierte Daten aus QR-Code-Scans,
- Kopien der FPGA Bitstreams für Wiederherstellungen,
- Detaillierte Berichte und Ergebnisse der Algorithmen.

6 Nichtfunktionale Anforderungen

Dieses Kapitel dient dazu die nicht-funktionalen Anforderungen des Projekts darzustellen. Dafür werden Qualitätsmerkmale in Tabellen dargestellt, welche von dem Institut für Betriebssysteme und Rechnerverbund (IBR) bereitgestellt wurden. Die Qualitätsmerkmale werden je nach ihrer Priorität für das Projekt und den zu erreichenden Zielen bewertet und im Anschluss an die entsprechende Tabelle erläutert.

6.1 Funktionalität

Produktqualität	**sehr gut**	**gut**	**normal**	**nicht relevant**
Angemessenheit		x		
Richtigkeit	x			
Interoperabilität			x	
Ordnungsmäßgkeit		x		

Gut als Angemessenheit bedeutet, dass das Monocle alle notwendigen Funktionen bietet, um verdeckte Kommunikation mit Steganographie und Kryptographie zu ermöglichen, ohne überflüssige Features die das System komplexer machen würden zu enthalten. Die Richtigkeit wurde auf sehr gut festgelegt, da es entscheidend ist, dass Daten korrekt verschlüsselt, eingebettet und wieder entschlüsselt werden, da dortige Fehler die gesamte Funktionalität untergraben. Die Software ist ausschließlich für das Monocle geschrieben, weswegen die Interoperabilität mit anderen System nicht zentral ist. Es wird erwartet, dass das fertige Produkt gute Ordnungsmäßigkeit aufweist, dennoch besteht immer das Risiko einer unzweckmäßigen Nutzung.

6.2 Sicherheit

Produktqualität	**sehr gut**	**gut**	**normal**	**nicht relevant**
Zuverlässigkeit	x			
Robustheit	x			
Wiederherstellbarkeit			x	

Die Zuverlässigkeit muss sehr gut sein, da vertrauliche Nachrichten übertragen und dargestellt werden. Ebenso muss auch die Reife sehr gut sein, da Fehler in der Sicherheit zu Datenlecks

oder fehlerhaften Ver- und Entschlüsselungen führen könnten. Das System wird vorher ausgiebig getestet, da es im Umgang mit sensiblen Informationen ist und somit wird erwartet, dass es Fehlerfrei funktioniert. Da das Monocle keine Daten speichert und der Zustand der verschlüsselten Nachrichten vom Steganogramm abhängt, hat ein Ausfall keinen Einfluss auf die Datenintegrität. Im Falle eines Geräteausfalls, wird ein standardmäßiger Neustart erwartet, der das Gerät schnell wieder funktionsbereit macht.

6.3 Benutzbarkeit

Produktqualität	**sehr gut**	**gut**	**normal**	**nicht relevant**
Verständlichkeit	x			
Erlernbarkeit		x		
Bedienbarkeit	x			
Effizienz		x		
Zeitverhalten		x		
Verbrauchsverhalten			x	

Das System ist leicht zu bedienen. Es enthält keine GUI sondern gibt lediglich Wörter auf englischer Sprache zurück, die angeben ob der QR-Code gescannt wurde. Das Monocle bietet eine gute Erlernbarkeit, da man zum Bilder aufnehmen nur die Touchpads auf dem Monocle bedienen muss. Um die Entschlüsselung durchzuführen, muss man sich jedoch grundlegend mit Computern auskennen, da man das Monocle über einen Computer steuert. Da keine weitere besondere Hardware benötigt wird und und zum Betrieb ein handelsüblicher Computer reicht, ist das System effizient. Das Einlesen des QR-Codes soll innerhalb einer angemessener Zeit erfolgen. Das Verbrauchsverhalten ist normal, da jedes aufgenommene Foto auf dem Computer gespeichert wird. Die Batterie hält für mindestens 70 aufnahmen.

6.4 Änderbarkeit

Produktqualität	sehr gut	gut	normal	nicht relevant
Analysierbarkeit		x		
Modifizierbarkeit		x		
Stabilität		x		
Prüfbarkeit	x			
Übertragbarkeit			x	
Anpassbarkeit	x			
Installierbarkeit			x	
Konformität	x			
Austauschbarkeit			x	

Eine gute Analysierbarkeit ist durch die Dokumentation des entwickelten Codes gegeben. Das Monocle lässt sich mit weiteren Funktionen modifizieren, diese sind aber in der Anzahl beschränkt. Die Stabilität des Systems wird als gut bewertet, der eingebaute FPGA eine flexible Hardware-Plattform bietet. Eine hohe Prüfbarkeit des Systems ist essentiell, um die korrekte Implementierung der Ver- und Entschlüsselungsfunktionen zu gewährleisten. Der entwickelte Code ist Speziell für die Hardwareumgebung des Monocle vorhergesehen und hat keine Anforderungen an die Funktion in anderen Umgebungen. Das System sollte eine gute Anpassung aufweisen, um Weiterentwicklungen der Funktionen integrieren zu können. Das Monocle wird bereits mit einer Standard-UI ausgeliefert, in der man seine geschriebene Skripte einfach hochladen und ausführen kann. Aufgrund der Natur des Projekts, dass Verschlüsselte und Verdeckte Nachrichtenübertragung beinhaltet, ist die Einhaltung von Datenschutzgesetzen nötig. Das Monocle speichert und verarbeitet dementsprechend keine der Daten, sondern wird lediglich als Schlüssel verwendet die um Nachrichten zu erkennen und zu entschlüsseln. Die Austauschbarkeit des Monocle läuft problemlos, da es nur das erneute Hochladen des Skripts auf das Monocle erfordert.

6.5 Qualitätsanforderungen

Hier werden die wichtigsten Qualitätsmerkmale aufgelistet:

- $\langle Q10 \rangle$ Die Funktion $\langle F10 \rangle$ soll nicht länger als 10 Sekunden dauern.
- $\langle Q20 \rangle$ Die Funktion $\langle F20 \rangle$ muss innerhalb 10 Sekunden ausgeführt werden.
- $\langle Q30 \rangle$ Mit den Touchpads auf dem Monocle soll man das Bild aufnehmen können.

- ⟨$Q40$⟩ Pro Druck des Touchpads soll immer nur ein Bild aufgenommen werde.

- ⟨$Q50$⟩ Der Akku soll so lange halten, dass mindestens 70 mal ein QR-Code gelesen und die Nachricht entschlüsselt werden kann.

- ⟨$Q60$⟩ Das Monocle soll die automatische Fehlerkorrektur von QR-Codes unterstützen.

- ⟨$Q70$⟩ Es soll stets eine stabile Kommunikation zwischen Monocle und Host bestehen.

7 Benutzeroberfläche/Schnittstellen

Es folgen die verschiedenen Benutzeroberflächen, die das Monocle-Gerät während des Ausführens unserer Anwendung ausführen wird.

Benutzeroberfläche:
In der unteren Hälfte des Brillenglases befindet sich ein Balken auf dem Text und andere grafische Elemente angezeigt werden können. Wird das Monocle-Gerät aus dem dafür vorgesehenen Gehäuse entfernt, das gleichzeitig als Ladestation fungiert, fährt das Gerät hoch und es ist bis zur Ausführung eines Programms das Logo mit der Aufschrift „Monocle“ unten in der Mitte des Bildschirms zu sehen.

Sobald das Gerät nun mit einem Laptop oder Computer mit Bluetooth-Anschluss verbunden wird und unser Programmcode dort ausgeführt, erscheint in der Mitte des Bildschirms die Meldung „Waiting...“, die die Bereitschaft des Monocle-Geräts symbolisiert. Wann immer der Auslese-Vorgangs eines QR-Codes abgeschlossen ist, wird das Gerät wieder zur Benutzeroberfläche 7.1 zurückkehren.

Befindet sich das Gerät in dem oben beschriebenen Zustand, erwartet es eine Eingabe über eines der oben liegenden Touchpads (Abbildung 8.5). Ist die besagte Eingabe vonstatten gegangen, wechselt das Monocle-System zu der folgenden Benutzeroberfläche, „Capturing...“, die dem Träger die Aufnahme eines Fotos zur Durchsuchung nach einem QR-Code signalisiert.

Sobald der besagte Aufnahmeprozess abgeschlossen ist, meldet das Monocle-System dem Träger die fertige Bildaufnahme mittels „Done.“ in der unteren Hälfte des Brillenglases. Wenige Sekunden später ändert sich eben dieser Text zu der Entschlüsselten Botschaft, die in dem gescannten QR-Code verborgen lag. Andernfalls wechselt das Gerät zu 7.4.

Im Falle einer Fehlermeldung, etwa falls kein QR-Code in der Aufnahme gefunden werden kann, erscheint die Meldung „No QR Code found in the image.“ In diesem Fall wird die Anzeige des entschlüsselten Texts am Ende von 7.3 übergangen und unmittelbar zu Abbildung 7.4 gesprungen.

In unserer Anwendung muss dabei nicht weiter zwischen einzelnen Rollen der Benutzer unterschieden werden. Stattdessen genügt es sämtliche Nutzer als einfache „Träger“ des Geräts zu identifizieren. Die Erzeugung der in Kunstwerken versteckten QR-Codes hingegen liegt außerhalb der Kompetenz des Monocle-Systems.

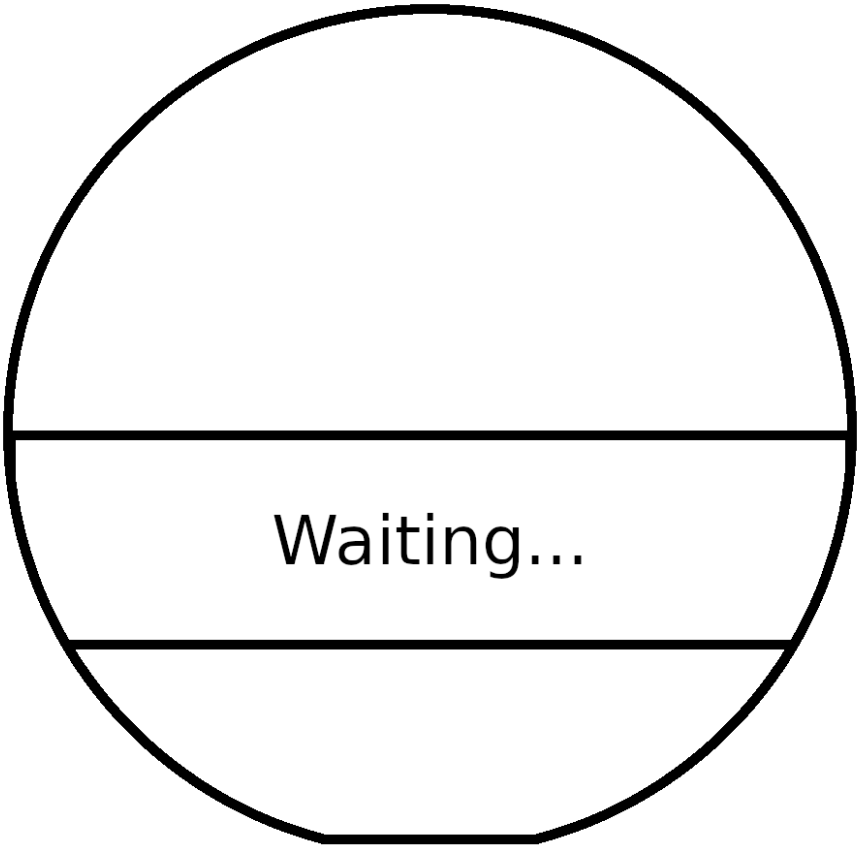

Abbildung 7.1: Initiale Benutzeroberfläche

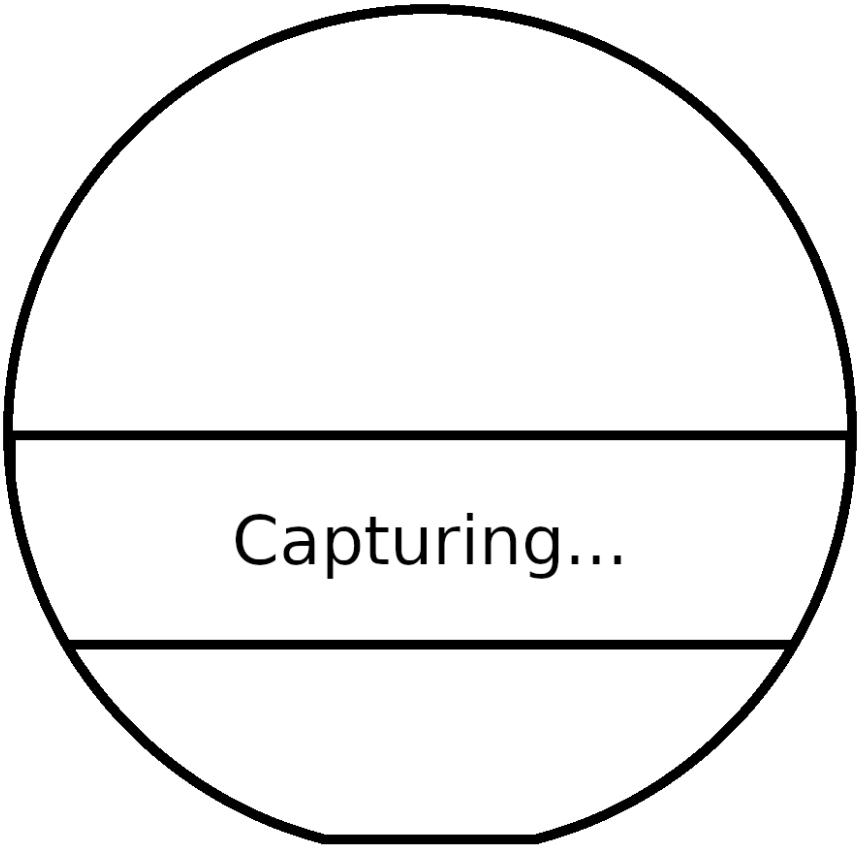

Abbildung 7.2: Benutzeroberfläche während der Aufnahme

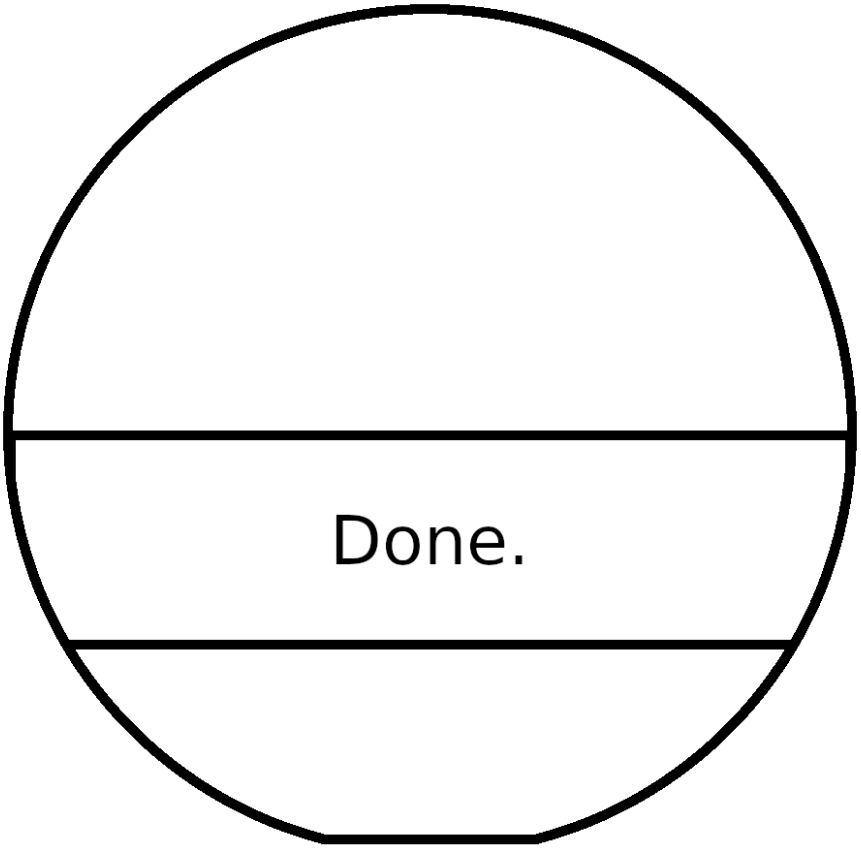

Abbildung 7.3: Benutzeroberfläche bei Vollendung des Einlesens

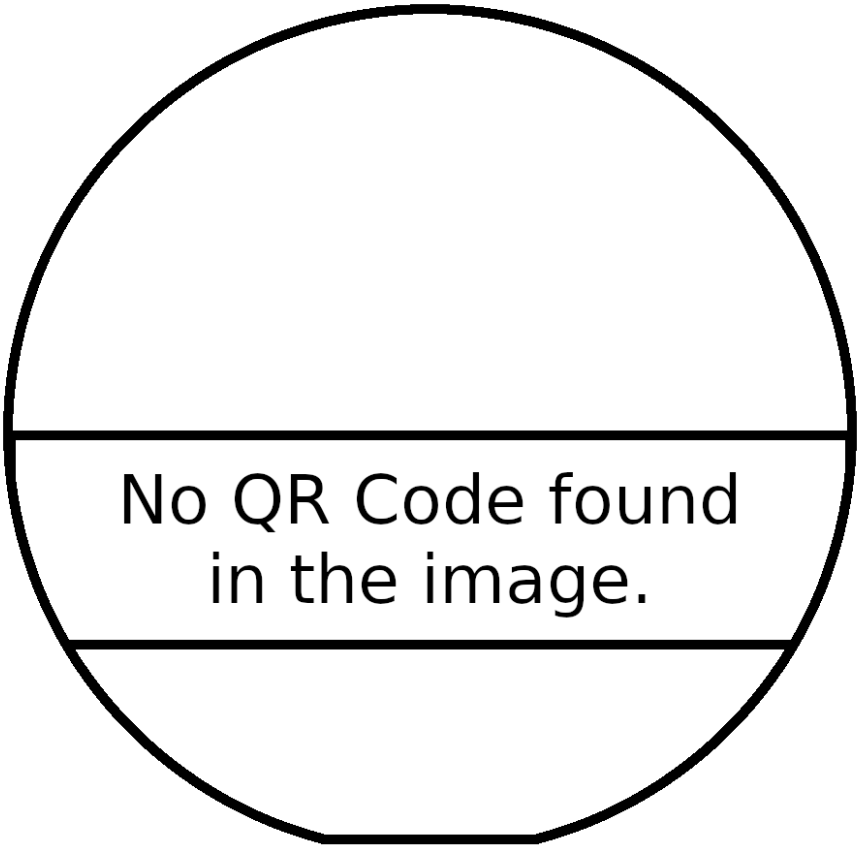

Abbildung 7.4: Benutzeroberfläche im Fehlerfall

Schnittstellen:
Da es sich bei unserer Anwendung bereits um ein lauffähiges Endprodukt handelt, werden keine Schnittstellen zur Verfügung gestellt, abseits der ohnehin schon implementierten Funktionen des Monocle-Geräts. Hierzu zählt die grundlegende Bedienung der einzelnen Hardware-Komponenten (beispielsweise der Kamera), wie im folgenden Kapitel beschrieben.

8 Technische Produktumgebung

In diesem Kapitel werden wir sowohl den Hardware-Teil des Monocle als auch den Software-Teil vorstellen. Wir werden auch über die Kommunikation zwischen dem Monocle und dem Benutzer sprechen und wie diese ermöglicht wird.

8.1 Software

In diesem Abschnitt wird die für das Monocle-Projekt erforderliche Softwareumgebung beschrieben. Die Entwicklung und der Betrieb der Softwarekomponenten werden durch die verwendeten Programmiersprachen, Bibliotheken und Frameworks erleichtert.

Das Monocle führt Open-Source-Software aus und bietet eine einfache Möglichkeit, mit MicroPython zu beginnen. Basierend auf dem MicroPython-Projekt wird die angepasste Version der MicroPython-Firmware mit Monocle verwendet. Mit der Modifikation und Upgrades zur Maximierung der Kompatibilität und Leistung, ist diese Firmware exklusiv für das Monocle-Gerät entwickelt. Die Firmware wird ständig aktualisiert, um immer neue Funktionen und Verbesserungen zu haben, und das ist dank der lebendigen und arbeitenden MicroPython-Community. Anstatt Low-Level-Programmierung zu erlernen, können Sie mit MicroPython schnell Prototypen erstellen und Anwendungen entwickeln. Mit nur wenigen Zeilen Python-Code können Sie das FPGA zur Verarbeitung nutzen, auf die Kamera zugreifen und auf dem Display zeichnen. Darüber hinaus stehen Ihnen alle Vorteile von Python zur Verfügung. Der Python REPL ist über Bluetooth leicht zugänglich und vollständig drahtlos.

Für die Entwicklung und Interaktion mit dem Monocle-Gerät wird auch Software auf dem Host-System benötigt. Die folgende Software wird auf dem Entwicklungsrechner vorausgesetzt:

- **Betriebssystem**: Windows oder Linux.
- **Python**:Damit wir mit dem Monocle kommunizieren, und die Skripte auszuführen, nutzen wir Python 3.7 oder höher.
- **Python-Bibliotheken**:
  - **Pillow**: für Bildverarbeitung.

  - **open-CV**:Für erweiterte Bildverarbeitung und QR-Code-Erkennung.
  - **numpy**: Für numerische Operationen und Datenmanipulation.
  - **asyncio**: Für asynchrone Kommunikation und Programmierung mit dem Monocle.
- **IDE**:Für unser Code wird Visual Studio Code, oder Pycharm empfohlen.

8.2 Hardware

Hier werden die Hardware Komponenten dargestellt.Das Diagramm in Abbildung 8.1 und 8.2 zeigt einen allgemeinen Überblick über die Monocle-Architektur.

Der Hauptprozessor von Monocle ist die Bluetooth-MCU. Es übernimmt den Großteil der Gerätesteuerung und wird für die Vernetzung und Skripterstellung verwendet. Als MCU kommt ein Nordic nRF52832 mit 64 KB RAM und 512 KB Flash-Speicher zum Einsatz. Bis zu 2 Mbit/s Bluetooth 5.2-Unterstützung werden bereitgestellt.

Das FPGA dient der Grafikbeschleunigung sowie der Bildverarbeitung der 5MP-Kamera. Es eignet sich perfekt für Computer-Vision- und KI-Aufgaben, bei denen Kamera und Mikrofon direkt als Eingaben genutzt werden können. Der verwendete FPGA-IC ist ein Gowin GW1N-LV9MG100C6/I5 aus der Little Bee Family. Es enthält 7.000 LUTs, 468 KB Block-RAM sowie 608 KB Flash-Speicher. Ein zusätzlicher 1-Megabyte-SPI-Flash-IC (8 Megabit) ist ebenfalls im Monocle enthalten, und entweder der Flash-Speicher kann zum Hochfahren des FPGA oder zum Speichern von Benutzerdaten verwendet werden. Weitere Einzelheiten finden Sie im Package and Pinout Guide.

Monocle ist ein winziges Head-up-Display, das an Ihrer vorhandenen Brille befestigt wird. Es ist zusammengesetzt aus leistungsstarker Hardware, die sich perfekt für unterwegs eignet. Es stellt über Bluetooth eine Verbindung zu Ihrem Mobiltelefon her und verfügt über einige praktische Sensoren wie Touch-Tasten, Kamera und Mikrofon. Das mitgelieferte FPGA eignet sich perfekt für Computer Vision, KI oder Grafikbeschleunigung direkt auf dem Gerät.

Neben den integrierten Speichern der Bluetooth-MCU und des FPGA enthält Monocle drei zusätzliche Speicher-ICs. Einer für Flash und zwei für RAM. Der RAM ist über das FPGA zugänglich und eignet sich hervorragend zum Speichern von Anzeigepuffern, Kameradaten oder Arbeitsdaten für KI-Algorithmen. Der Flash eignet sich hervorragend als sekundäre Quelle zum Laden von FPGA-Bitströmen und zum Protokollieren von Daten und ist sowohl für die Bluetooth-MCU als auch für das FPGA zugänglich.

Der verwendete Flash-IC ist der Winbond W25Q80EWUXIE. Es handelt sich um einen seriellen Flash mit 1 Megabyte (8 Megabit), der über einen SPI-Bus mit dem FPGA und nRF52 verbunden ist.

Als RAM-ICs kommen die AP Memory APS256XXN zum Einsatz. Insgesamt sind 64 Megabyte (512 Megabit) Speicher adressierbar und im DDR-Modus kann mit bis zu 800 MBit/s darauf zugegriffen werden.

Das im Monocle verwendete Display ist ein 0,23-Zoll-Micro-OLED (siehe Abbildung 8.3). Es verfügt über 640 x 400 RGB-Pixel und ist optisch mit dem Hauptgehäuse verbunden, wodurch das Bild in das Auge des Benutzers gelenkt wird. Das Ergebnis ist ein transparentes schwebendes Display mit einem Sichtfeld von 20°. Ungefähr so groß wie ein Tablet-Display in Armlänge.

Als Frontkamera (siehe Abbildung 8.4) des Monocle dient ein Omnivision OV5640. Es verfügt über einen 5-Megapixel-Farbsensor mit vielen beeindruckenden Funktionen, wie dem automatischen Weißabgleich und der Belichtungsverwaltung. Über eine schnelle MIPI-CSI-2-Schnittstelle ist es sofort mit dem FPGA verbunden.

Zwei kapazitive Touch-Tasten am Monocle (siehe Abbildung 8.5) sind mit einem Azoteq IQS620A Touch-Controller verbunden. Jede Taste ist in der Lage, sowohl Berührungsereignisse als auch unmittelbare Nähe zu erkennen. Zusätzliche Softwareverarbeitung ermöglicht die Erkennung anderer Bewegungen, wie z. B. mehrfaches Tippen und längeres Drücken. Um bevorstehende Touch-Ereignisse anzuzeigen, ist der Touch-Controller über I2C und eine Interrupt-Leitung mit der Bluetooth-MCU verbunden. Eine integrierte Bibliothek in unserer MicroPython-Firmware ermöglicht den Zugriff auf verschiedene Arten von Berührungsereignissen und das Ausführen von Aktionen beim Drücken.

Das FPGA ist direkt mit einem TDK/InvenSense ICS-41351-Mikrofon (siehe Abbildung 8.6) verbunden, das für Spracherkennungssoftware oder Audioaufzeichnung verwendet werden kann. Da sich der Audioanschluss des Monocle auf der Rückseite befindet, ist es ideal, um die Stimme des Trägers aufzunehmen, ohne die Aufmerksamkeit auf sich zu ziehen.

Als Host-Hardware verwendeten wir einen Laptop, der sowohl Windows als auch Linux als Betriebssystem hat. Er hat einen Ryzen 7 als Prozessor, eine Nvidia RTX 3050 als Grafikkarte, 8 GB RAM und 1 TB SSD. Der Laptop verfügt außerdem über Bluetooth-Konnektivität und verschiedene USB-Anschlüsse wie C und A.

8.3 Produktschnittstellen

Wenn Sie Monocle aus seinem Gehäuse nehmen, schaltet es sich automatisch ein. Das Display schaltet sich ein, und die Verbindung über Bluetooth ist einfach. Monocle geht automatisch

in den Ruhezustand und lädt sich auf, wenn Sie es wieder in die Hülle stecken. Die Kommunikation zwischen dem Monocle und einem mobilen Gerät erfolgt mit Bluetooth Low Energy, auch bekannt als BLE. Die BLE-Services und Eigenschaften, die den Datenfluss und -Austausch ermöglichen sind :

- **Monocle Service**: spielt die Hauptrolle beim Datenaustausch, wo er den gegenseitigen Austausch von Daten in beide Richtungen zwischen dem Monocle und dem mobilen Gerät durch UUID erleichtert.
- **TX-Charakteristik**: spielt die Rolle des Sendens der Daten vom Monocle selbst an das angeschlossene mobile Gerät unter Verwendung von UUID.
- **RX-Charakteristik**: wie TX, aber sie sendet Daten in die andere Richtung, vom angeschlossenen Gerät zum Monocle.

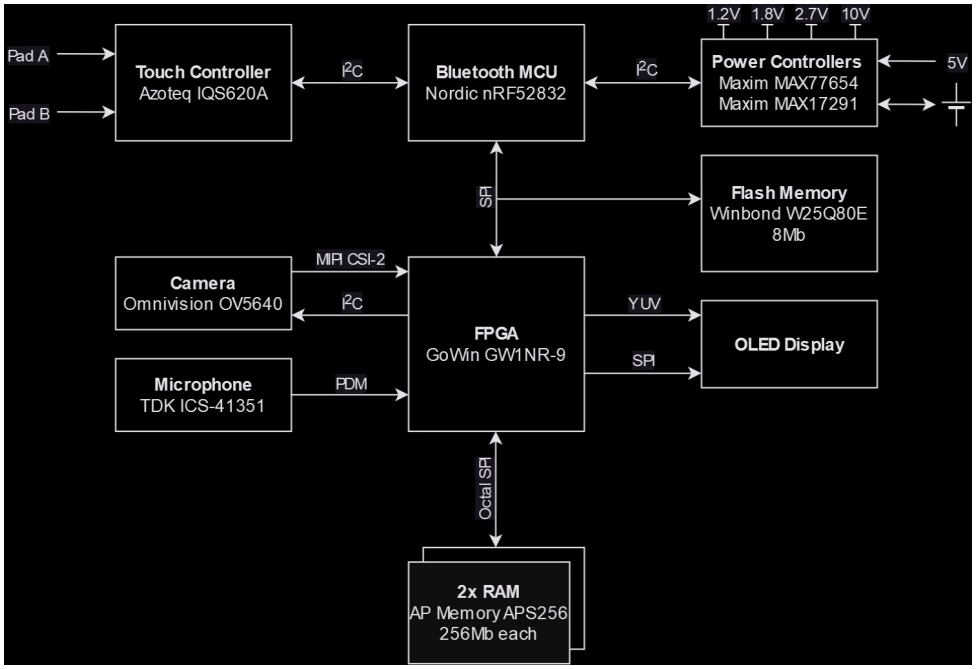

Abbildung 8.1: Darstellung

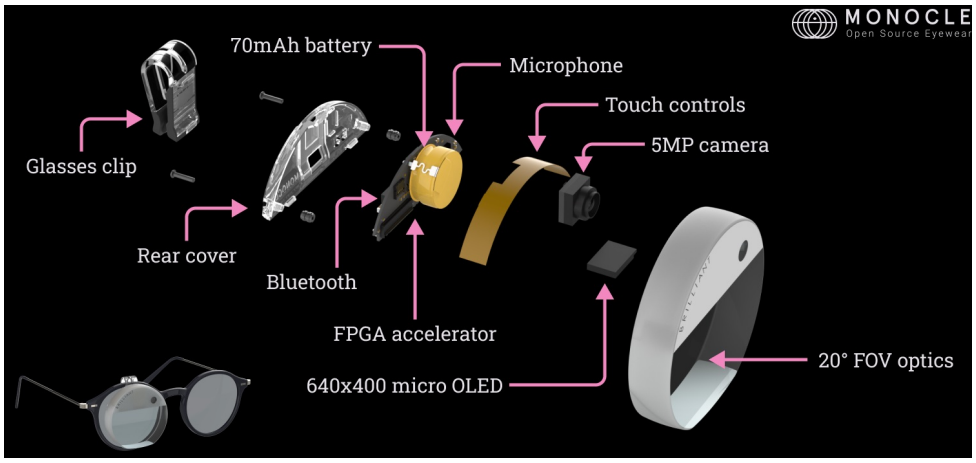

Abbildung 8.2: Monocle Hardware Manual

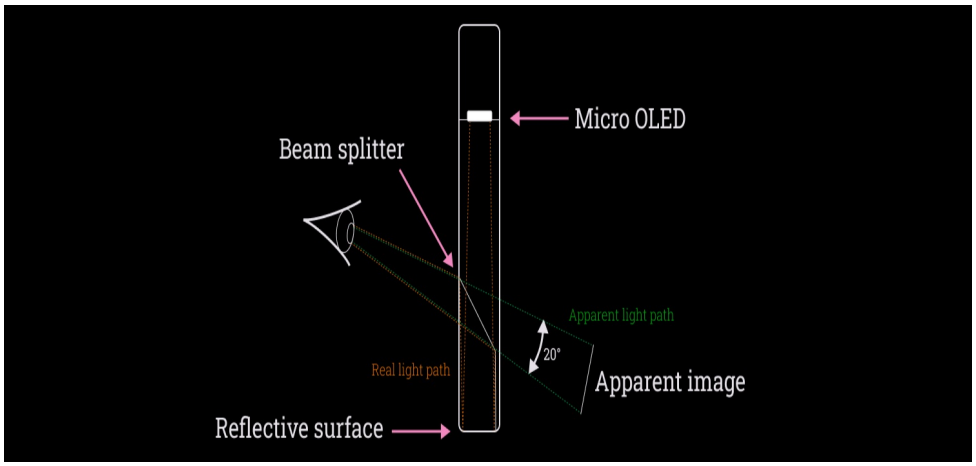

Abbildung 8.3: Darstellung vom Display

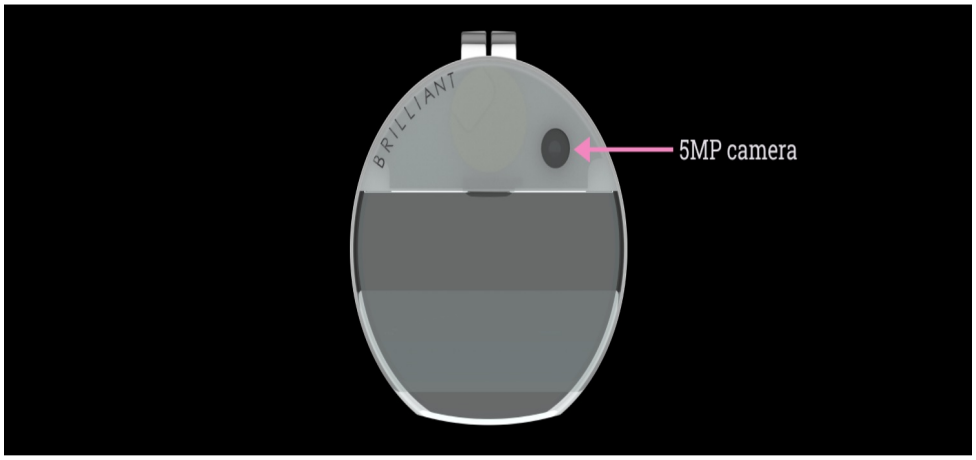

Abbildung 8.4: Bild vom Camera

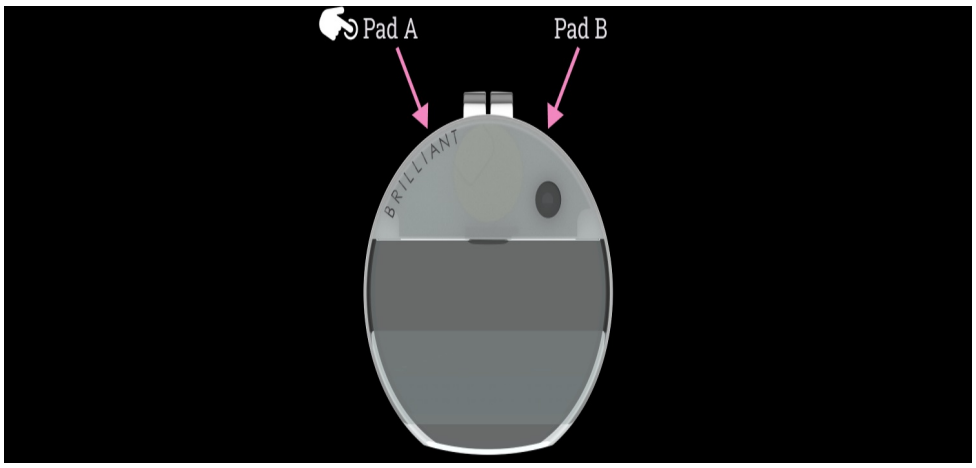

Abbildung 8.5: Darstellung vom Touch Pads

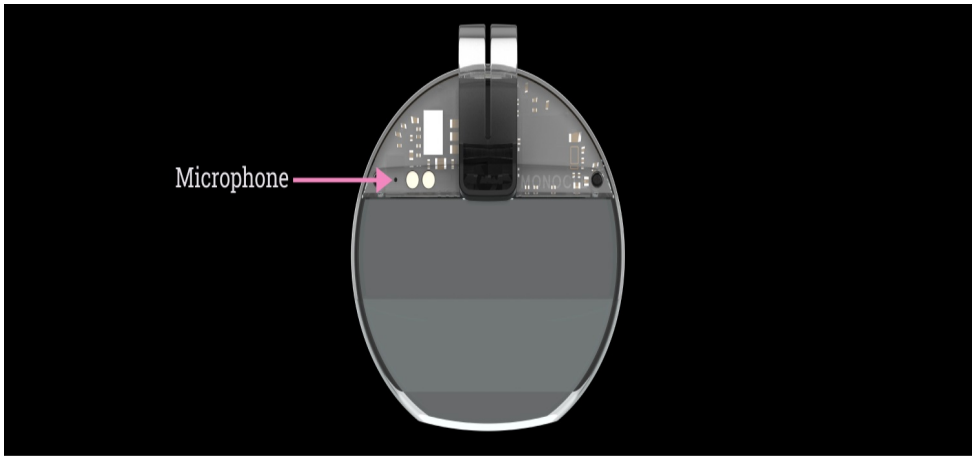

Abbildung 8.6: Bild vom Microphone

9 Glossar

(Alphabetisch und absteigend sortiert)

Aktivitätsdiagramm - UML-Verhaltensdiagramm zur Modellierung von Vernetzung der Aktionen.

BLE-Kommunikation - Bluetooth Low Energy-Kommunikation ist eine Technik zur drahtlosen Vernetzung von Hardware, die auf eine Reichweite von unter 10 Metern ausgelegt ist.

FPGA - Steht für 'Field-Programmable Gate Array' und ist eine Art von programmierbarem Logikbaustein.

IAS - Institut für Anwendungssicherheit an der TU Braunschweig.

IDE - Kurz für Integrated Development Environment, und ist eine Softwareanwendung.

MicroPython - ist eine Implementierung der Programmiersprache Python, die speziell für Mikrocontroller und eingebettete Systeme entwickelt wurde.

QR-Code - Kurz für Quick Response, ist ein zweidimensionaler Strichcode, welcher Informationen in Form von Zeichen und Ziffern speichert.

REPL - für "Read-Eval-Print Loopünd bezeichnet eine interaktive Umgebung, die häufig in Programmiersprachen wie Python verwendet wird.

UI - User Interface. Grafische Benutzeroberfläche einer Anwendung.

Use-Case - Anwendungsfall.

Use-Case-Diagramm - UML-Verhaltensdiagramm, welches Vernetzungen zwischen Anwendungsfällen und Akteuren anzeigen kann.

UUID - steht für Universally Unique Identifier.