1 Mockups
Die Mockups wurden mithilfe von Balsamiq Wireframes angefertigt. In der Desktopansicht (Abbildung 1) sind 2 Bereiche zu erkennen. Der erste Bereich ist der Header, welcher aus 2 Teilen besteht. Links ein Logotext und rechts die Navigationsleiste mit den Einträgen „Home“, „About“ und „Contact“. Der Header soll beim Scrollen an oberster Stelle hängen bleiben und so jederzeit verfügbar sein.
Im Hauptbereich der Anwendung findet man die Übersetzungsmaske sowie unter dieser Maske ein Erklärungsvideo. Die Übersetzungsmaske teilt sich in 3 Spalten auf, wovon die linke Spalte den zu übersetzenden Text enthält und die rechte Spalte das Ergebnis. In der mittleren Spalte findet man die beiden Buttons „Translate“ und „Save“.

In der mobilen Ansicht (Abbildung 2) kann man weiterhin diese Bereiche erkennen. Die Ansicht unterscheidet sich darn, dass die Navigationsleiste erst nachdem man auf das Menüicon geklickt hat unterhalb des Headers befindet und den Inhalt überlagert. Der Übersetzungsblock wechselt seine Ausrichtung und wird als einzige Spalte untereinander angezeigt. Die Buttons werden anstelle von untereinander nun nebeneinander angezeigt.

2 Architektur
Es soll eine Interaktive Anwendung entstehen, welche in der Endstufe Daten in eine andere Sprache übersetzen kann. Hierfür eignet sich eine Client-Server-Architektur. Deshalb habe ich mich dazu entschieden für das Web-Frontend React bzw. für das Backend node.js zu verwenden.

Da bei einer Übersetzungsanwendung davon auszugehen ist, dass es Spitzenzeiten mit extrem vielen Anfragen, aber genauso Zeiten mit extrem wenigen Anfragen existieren, habe ich mich für eine serverless-Infrastruktur entschieden.
AWS ist aktuell mit seinem Produkt AWS Lambda der zuverlässigste serverless-Anbieter. Hier können von einem Request pro Monat bis hin zu mehreren tausend requests pro Millisekunde alle Anwendungsfälle abgedeckt werden. Um sicher auf die Lambda-Funktion zugreifen zu können habe ich ein das Amazon API Gateway verwendet. Hier ist es möglich Authentifizierungsverfahren zu definieren, um sich vor nicht autorisierten Zugriffen zu schützen. Um die React App zu hosten verwende ich AWS Amplify, welches ebenfalls eine serverlose Lösung ist.

Für den aktuellen Anwendungsfall ist die serverlose Infrastruktur überdimensioniert. Sollte ein solches Projekt jedoch vorangetrieben werden, macht es definitiv Sinn sich mit der serverlosen Infrastruktur auseinanderzusetzen. Die gesamte Architektur habe ich in Abbildung 3 aufgezeichnet.

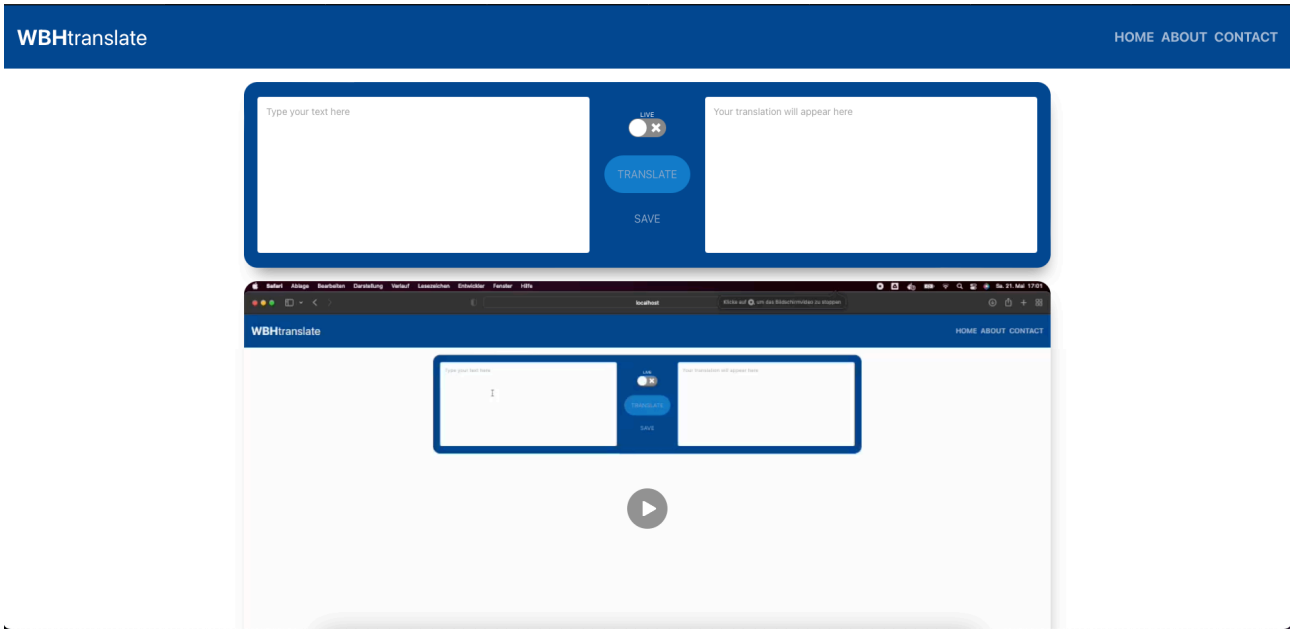
3 Umsetzung
Zur Erstellung der React App habe ich create-react-app verwendet, welches von Facebook zur Erstellung neuer React Applikationen empfohlen wird. Ich habe mich für eine typischere Entwicklung durch die typescript-Erweiterung von CRA entschieden. Zusätzlich zu den bereits standardmäßig installierten Bibliotheken habe ich axios, classnames, i18next, react-router-dom uns sass verwendet.
Axios ermöglicht es einfach Anfragen an http-Endpunkte zu stellen und daraus entstandene Fehler bzw. erfolgreiche Antworten korrekt zu verarbeiten. Classnames ermöglicht es einfacher dynamisch wandelnde Klassennamen für die verwendeten HTML-Elemente zu vergeben (z.B. abhängig vom Status menuIsOpen). I18next ermöglicht eine einfache Übersetzbarkeit der durch die App verwendeten Texte. Somit werden texte nicht

direkt in den Quellcode geschrieben, sondern können an einer zentralen Stelle überarbeitet / übersetzt werden. Mithilfe von react-router-dom kann ein sehr komfortables Routing ermöglicht werden. Sass ist ein precompiler, der sass-styles in css umwandelt. Zur Anzeige der PDF-Datei auf der About-Seite wird zusätzlich die Bibliothek react-pdf verwendet.

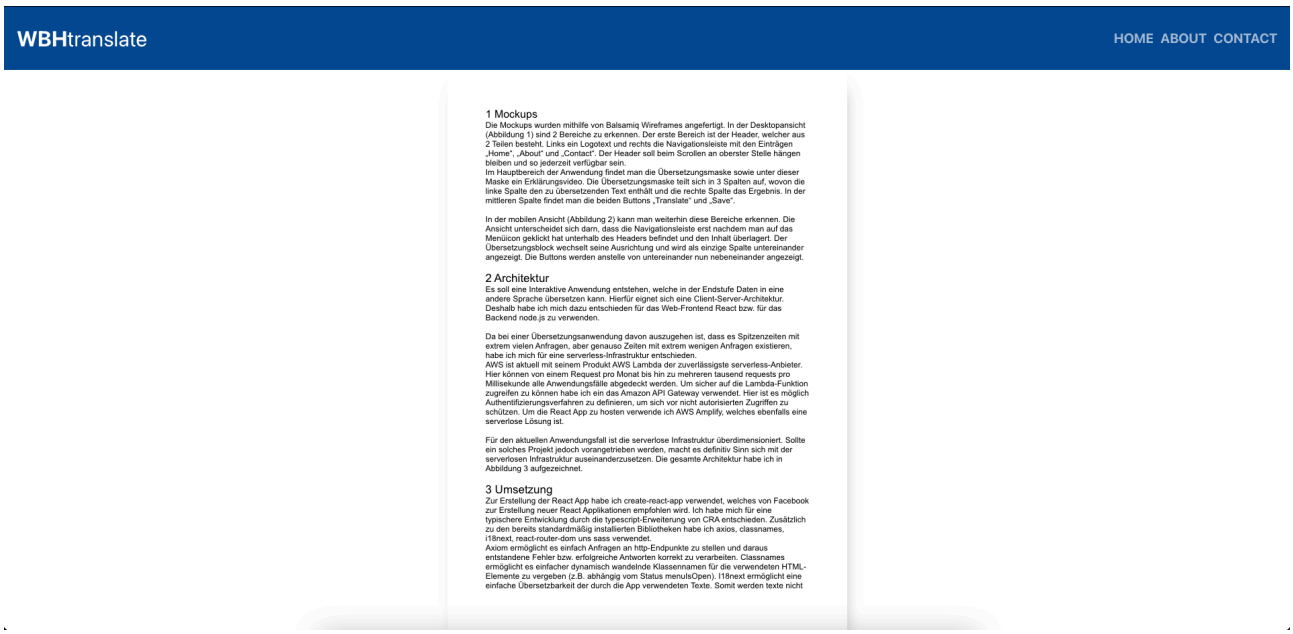
Die UnitTests wurden mithilfe von Jest geschrieben und erreichen eine Testabdeckung von mehr als 70%. Eine Cypress-Integration würde sich in diesem Fall anbieten, sobald das Projekt mehr Abhängigkeiten zwischen den Komponenten bzw. verschiedenen Seiten erhält. Beispiele hierfür sind Login, Letzte Übersetzungen und viele mehr.

Das Video habe ich mithilfe von Apple Bildschirmfoto erstellt und mit Compressor in das richtige Format gewandelt.

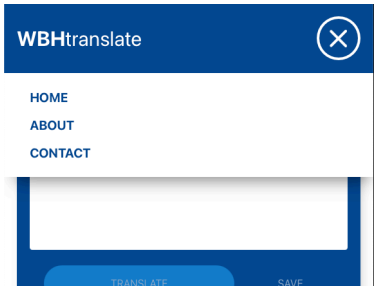
4 Screenshots



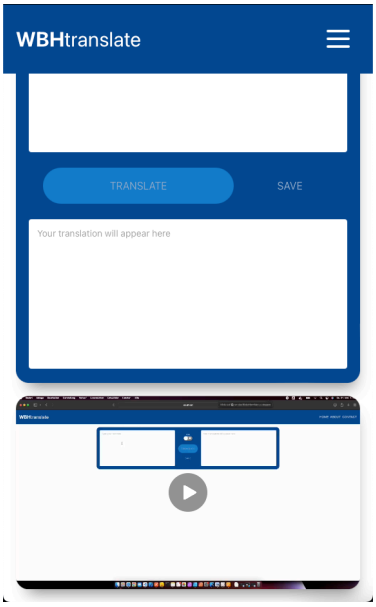
Screenshot 1 - Desktop Home



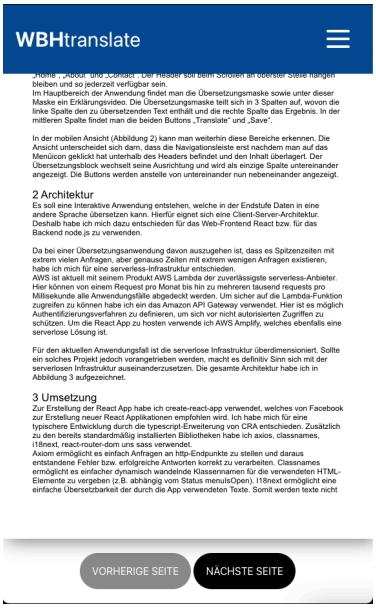
Screenshot 2 - Desktop About



Screenshot 3 - Mobil Menü



Screenshot 4 - Mobil About



Screenshot 5 - Mobil Home

Abbildung 1 - Desktopansicht

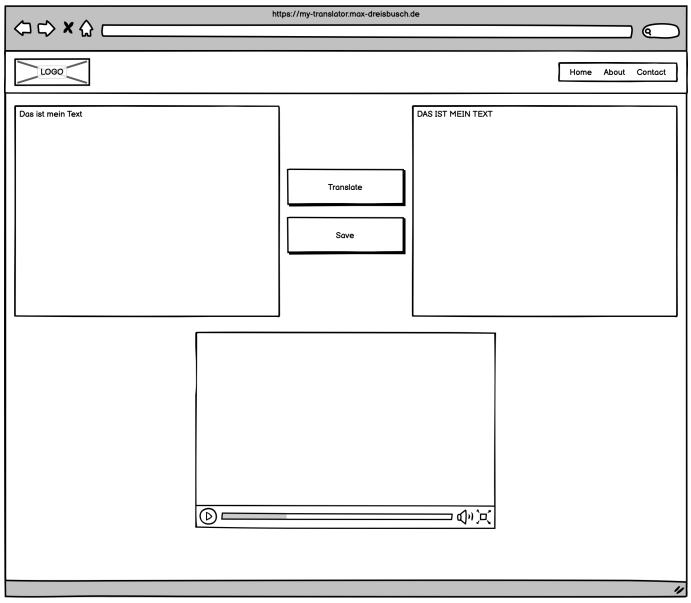
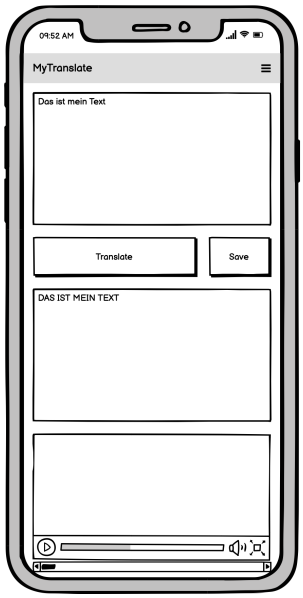


Abbildung 2 - Mobile Ansicht



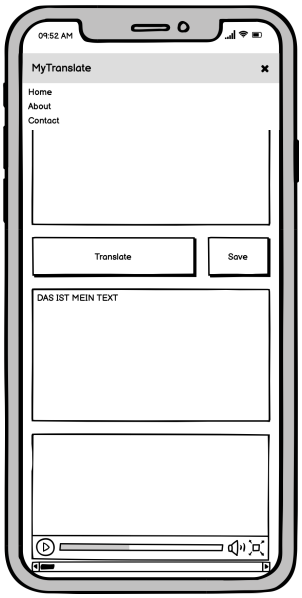


Abbildung 3 - Infrastruktur