

# Geheime Abstimmung Dokumentation

**Marco Wanka,  
Luca Klingert,  
Michael Schmitt,  
Johannes Gehring,  
Tim Braunger**

**SS2022**

# **Inhaltsverzeichnis**

- 1. Allgemeine Beschreibung**
  - 1.1 Einsatzbereich und Zweck**
  - 1.2 Benutzer und Rollen**
  - 1.3 Anforderungen**
  - 1.4 Programmiersprachen/Entwicklung**
- 2. Frontend**
  - 2.1 Login**
    - 2.1.1 Sessions im Frontend/Logout**
    - 2.1.2 Validierung der Session beim Seitenwechsel**
  - 2.2 Main**
    - 2.2.1 Main-Options**
  - 2.3 Editor**
  - 2.4 Verschlüsselung**
    - 2.4.1 TLS**
  - 2.5 Abstimmung**
    - 2.5.1 Überblick**
    - 2.5.2 Funktionsweise**
  - 2.6 Auswertung der Umfrage**
  - 2.7 Testen in Cypress**
  - 2.8 API**
- 3. Backend**
  - 3.1 Server**
  - 3.2 Hinweise zur Systemeinrichtung**
  - 3.3 Struktur der Datenbank**
  - 3.4 Sessions im Backend**
  - 3.5 Speicherung der User Login-Daten**
  - 3.6 Token Generierung**
  - 3.7 Speichern der verschlüsselten Antworten**
  - 3.8 E-Mail Verteiler**
  - 3.9 User E-Mail**
- 4. Mögliche Erweiterung des Systems**



<b>Nr.</b>	<b>Anforderung</b>
FA01	Surveyleader müssen sich im System einloggen, um Funktionen wie Abstimmung erstellen nutzen zu können
FA02	Surveyleader können flexible Abstimmungen erstellen, d.h. definierter Zeitraum, mehrere Fragen mit verschiedenen Antworttypen und Angabe der Email-Adressen der Voter
FA03	Surveyleader können bereits erstellte oder laufende Abstimmungen löschen
FA04	Surveyleader können die Emails der zuletzt erstellten Abstimmung wiederverwenden
FA05	Nach erfolgreicher Erstellung einer Abstimmung wird ein Schlüssel(paar) generiert, welches zur Verschlüsselung der Ergebnisse verwendet wird
FA06	Surveyleader müssen den Schlüssel abspeichern können
FA07	Die Ergebnisse der Abstimmung dürfen nur nach Eingabe des korrekten Schlüssel als Klartext vorliegen und ausgewertet werden können
FA08	Die Abstimmung wird automatisch per Email mit den Votern geteilt
FA09	Voter stimmen über einen Abstimmungslink in der Email ab
FA10	Pro Voter darf nur einmal Abstimmen möglich sein
FA11	Beim Abstimmen werden keine weiteren Daten des Voters gespeichert oder benötigt
FA12	Ergebnisse werden im Browser des Voters verschlüsselt und im Backend gespeichert
FA13	Ergebnisse werden nur clientseitig im Browser des Surveyleaders entschlüsselt und aufbereitet
FA14	Ergebnisse werden per Datenbank persistent und ausschließlich verschlüsselt auf dem Server gespeichert
FA15	Surveyleader können ihr Passwort ändern
FA16	Surveyleader können ihren Usernamen ändern
FA17	Surveyleader können andere Surveyleader hinzufügen
FA18	Admins können neue Admins hinzufügen
FA19	Admins können Surveyleader bzw. Admins entfernen (solange es noch

	mindestens einen anderen Admin im System gibt)
FA20	Die Webanwendung ist über HTTPS erreichbar, um einen verschlüsselten Nachrichtenaustausch zu gewährleisten
FA21	Neu erstellte Nutzer erhalten ihre Login Daten automatisch per E-Mail

## 1.4 Programmiersprachen/Entwicklung

Aufgrund unserer Erfahrungen aus den vorherigen Semestern und insbesondere dem Modul Programmieren 3, haben wir uns bei der Entwicklung unserer Webapp für das Frontend-Web Application Framework Angular in Kombination mit einem Java Backend und MongoDB (ein dokumentenorientiertes NoSQL-Datenbank-managementsystem) als externe Datenbank entschieden.

Angular bietet im Vergleich zu anderen Webapplication-Frameworks die Flexibilität durch den Einsatz von TypeScript, was die Entwicklung in größeren Projekten durch die statische Typisierung und bessere Codevalidierung erleichtert, gleichzeitig aber auch JavaScript Bibliotheken ohne Probleme einbinden lässt. Zudem bietet Angular durch Angular Materials eine große Auswahl von Style-Elementen, die sich schnell und unkompliziert einbinden lassen, um einer Webapplication ein modernes Design zu verleihen.

Der Vorteil, Daten mit MongoDB persistent zu speichern, liegt im einfachen Umgang mit JSON, was den zusätzlichen Einsatz von SQL o.ä. unnötig macht.

Als Browser wurden Google Chrome, Safari und Firefox erfolgreich getestet.

## 2. Frontend

### 2.1 Login (MW)

FHWS
POLL

Login to your account

Email

Password

Need help?

Login

Abb.2

Nr.	Aktivität	Beschreibung
1	Enter your Email	Eingabe der eigenen Email
2	Enter your Password	Eingabe des dazugehörigen Passwortes
3	Submit	Betätigen des Login

Nach Betätigung des Login Buttons wird mithilfe des Backends entweder der Benutzer authentifiziert, oder es wird ein Fehler aufgrund von falschen Anmeldedaten, timeout oder nicht Erreichen des Backends geworfen. Diese werden wie folgt gekennzeichnet:

Serverproblem	Wrong Userdata	Timeout

Abb.3

Die Authentifizierung erfolgt nach dem ersten Login mithilfe einer SessionID.

Der Ablauf sieht wie folgt aus:

Zuerst werden die Benutzerdaten an das Backend gesandt und dort überprüft. Sind diese Korrekt, weist das backend dem Nutzer eine generierte Session-ID zu und gibt diese zusammen mit der Rolle des Benutzer (ist er ein Normaler Benutzer oder ein admin?) an das Frontend zurück. Das Frontend nutzt fortan die Session-ID und gegebenenfalls noch die Rolle zur authentifikation bei jeder Anfrage an den Server. Hierbei ist zu beachten das bei jeder Anfrage die Sessionid gegen eine neue Sessionid ausgetauscht und zurückgegeben wird sowie das eine Session zeitlich begrenzt ist auf 60 Minuten. Dies stellt sicher, dass der Missbrauch eines Accounts des Nutzers durch Dritte, bei Vergessen eines logouts, lediglich 60 minuten möglich ist.

## 2.1.1 Sessions im Frontend/Logout (MW & MS)

### Laden der ersten Session-ID

Das Frontend verfügt über eine Authentication.service Klasse, welcher zum Zeitpunkt des Betätigen des Login Buttons gecalled wird und daraufhin mit hilfe der Backend.service Klasse die Benutzer Daten an das Backend versendet, welches wiederum mit einer Session-ID antwortet. Diese wird daraufhin der Mainseite

übergeben, welche damit ihre Daten wie Username, Rolle und die aktuellen Polls laden kann.

#### Aktualisieren der gültigen Session-ID durch Requests

Nach jeder Anfrage erhält das Frontend im body einer response eine neue Session-ID, diese kann daraufhin mit der methode updateSessionid() im localStorage aktualisiert werden.

#### Logout des Nutzers

Der Logout eines User ist recht simpel gehalten, so wird einfach bei betätigung des Logout Buttons oder erneutes aufrufen (nicht refreshen) der Seite sämtliche Informationen inklusive der aktuellen gültigen Session id aus dem localStorage (dem Speicher des Browsers) gelöscht. Um die Sicherheit zu erhöhen könnte das System natürlich soweit erweitert werden, als das es bei betätigung des Logout buttons auch noch die aktuell gültige Session-ID im backend für ungültig erklärt wird. In unserem Fall wird dieses bis jetzt durch das automatische Verfallsdatum einer Session-ID von 60 Minuten erledigt.

## **2.1.2 Validierung der Session beim Seitenwechsel (MS)**

#### Auth.Guard

Zum Überprüfen der Berechtigung zum Wechsel zwischen den Verschiedenen Routing-Componenten wird die Angular-Funktion "CanActivate()" implementiert. Diese Funktion macht es möglich bestimmte Routen eines Angular-Projekts unter einer definierten Bedingung nur zu benutzen. In unserem Fall wären das die "/main", "/editor" und "/result" Seiten.

In unserem Projekt wird beim Aufrufen von "CanActive()" ein Call an das Backend gemacht, welcher die aktuelle Session id enthält. Dort wird dann überprüft, ob diese auch gültig ist. Je Nachdem wird daraufhin eine neue gültige Session id oder ein Http-Fehlercode geschickt. Im Falle eines Fehlers wird die aktuelle Session id gelöscht, dann wird man zurück zur Login Seite geleitet und die Funktion gibt einen "false"-Wert zurück. Tritt aber kein Problem auf, wird die neue Session id gespeichert, die Funktion gibt einen "true"-Wert zurück und man wird zur gewünschten Seite weitergeleitet.

## 2.2 Main (MW)

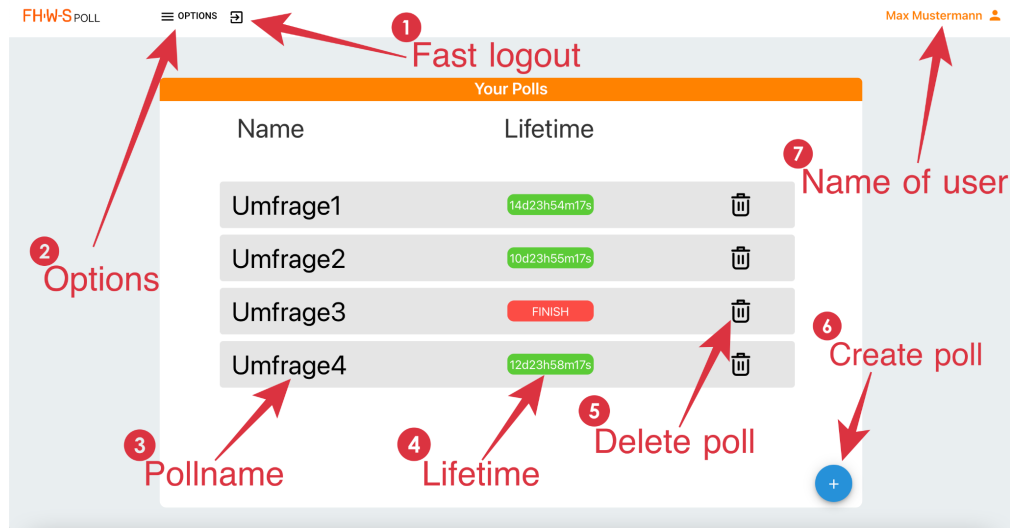


Abb.4

Die Main-Seite lädt nun die Umfragen des Benutzers. Der Aufbau der Main seite, siehe Bild, sieht wie folgt aus:

1. Dieser Logout Button ermöglicht das Ausloggen mit nur einem Klick
2. Dieser Options Button ermöglicht den zugriff auf verschiedene Funktionen siehe **2.2.1 Optionen**
3. An dieser Stelle wird der Name ihrer Umfrage Angezeigt.
4. Die Lifetime gibt an wie lange die Umfrage noch läuft bzw. ob diese sogar schon beendet ist. (Durch Anklicken der Umfrage Aktualisiert sich die Zeit bzw. bei einer Abgeschlossenen Umfrage wird das Ergebnis bzw. die Entschlüsselung des Ergebnisses Aufgerufen)
5. Dieser Button ermöglicht das unwiederbringliche Löschen der jeweiligen Umfrage
6. Dieser "+" Button ermöglicht das hinzufügen einer neuen Umfrage
7. An dieser Stelle sehen sie ihren Benutzernamen (nicht ihre Email !)

Die Main-Seite dient hier sozusagen nicht nur als Übersicht der Aktuellen Umfragen, sondern auch als Kreuzungspunkt zu allen weiteren Funktionen der Website.



## 2.2.1 Main-Funktionen (MW)

Das Options Fenster sieht wie Folgt aus:

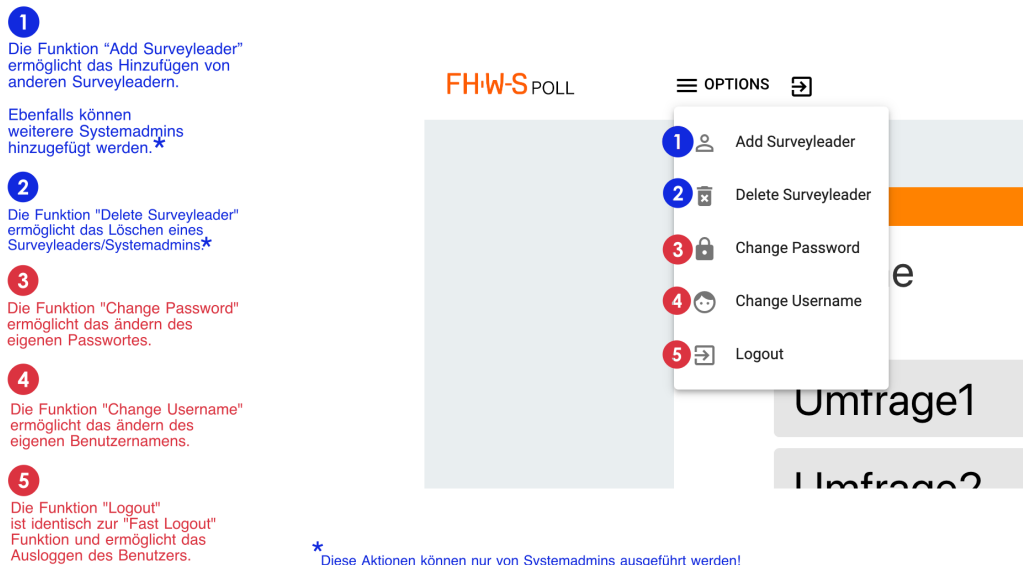


Abb.5

### Zur Funktion Add Surveyleader:

Für das Hinzufügen einer Surveyleader oder Systemadmins benötigt der Nutzer die Email des Hinzuzufügenden (überprüfung ob eingabe einer email entspricht erfolgt in echtzeit), sowie dessen Name für den Username, das passwort kann beliebig gewählt werden (Bedingung mindestens 8 zeichen länge). Nach Bestätigung der erstellung wird an die angegebene mail-adresse eine email den Zugangsdaten inklusive dem hinweis das Passwort nach erstmaligem Login zu ändern gesendet.

### Zur Funktion Delete Surveyleader/Systemadmin:

An die Funktion Delete Surveyleader wurde zu beginn des Projektes die Anforderung gestellt lediglich die Möglichkeit zu haben einen Surveyleader/Systemadmin zu löschen, die art wie eine Löschung dargestellt bzw. auszusehen hat wurde zu diesem Zeitpunkt nicht festgelegt. Im Laufe der Produktion wurde jedoch festgestellt, dass es sinnvoller ist, die Funktion so zu implementieren, dass diese alle bestehenden Surveyleader/Systemadmins anzeigt und es somit für den user durch einen einfachen Klick möglich ist die Datenbank von nicht mehr benutzen, bzw. vergessenen zu löschenden, Accounts zu bereinigen. Neben der Auflistung aller Aktuell bestehender Accounts, repräsentiert durch die jeweilige email, ist es ebenfalls möglich durch eine Kennzeichnung zu erkennen ob der jeweilige Account ein Systemadmin oder ein Surveyleader ist.

### Top Down Menu

Bei der Implementierung des Top Down Menus, wurde aus Design Gründen auf die, von Haus aus durch Angular bzw. durch einfache imports, mitgelieferte Angular

Materials<sup>1</sup> gesetzt. Diese liefern neben praktischen icons auch vorgefertigte objekte für eine einfache implementierung und besonders cleanes Design mit. Durch einfaches importieren der jeweiligen Angular-Material-Komponente, lassen sich diese elemente problemlos in den html code implementieren und durch eigenen anpassungen und änderunge perfektionieren.

### Visualisierung der Error und success Mitteilung in Main

Um dem User ein simples Feedback zu erfolgreichen änderung/erstellung/löschung eines Nutzers zu geben, wird nach jeder erfolgreich durchgeführten Änderung dem User für wenige Sekunden ein grünes bestätigungs Icon oder gegebenenfalls ein rotes Fehler Icon angezeigt. Warum gibt es keine Fehler Message?

Dies lässt sich sehr einfach erklären, bis auf die Möglichkeit das bereits ein nutzer mit gleicher Email in der Datenbank existiert, werden alle anfragen bereits vor dem abschicken kontrolliert, wie zb. Passwortlänge oder können erst gar nicht durch den Nutzer Falsch eingegeben werden. Dies führt zu dem logischen schluss, dass es sehr unwahrscheinlich ist dass der user selbst einen Fehler auslöst, weswegen eine genaue Beschreibung im Fall eines Fehlers für den User selbst irrelevant ist.

## 2.3 Editor (JG)

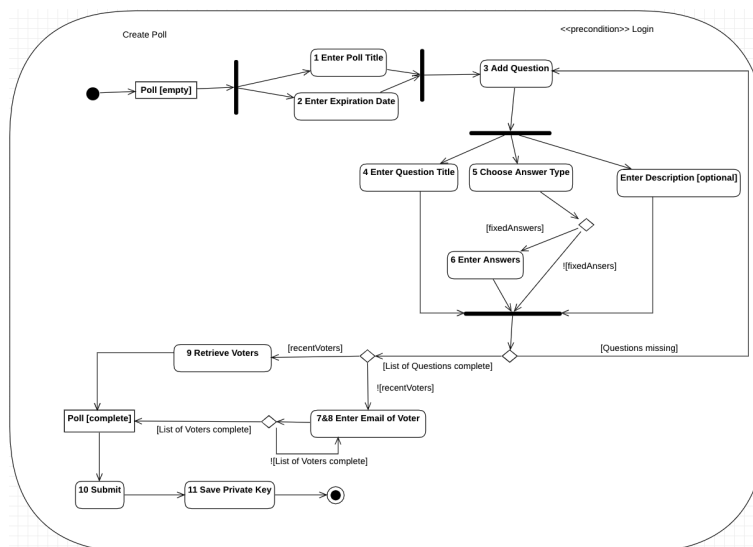


Abb.6

<sup>1</sup> <https://material.angular.io/components/menu/overview>

**Create new Poll:**

1

2

3 

Questions:  
Example   
+

**Edit Question:**

4 

Enter your question here. \*  
Example  
Add a description (optional)

5 

☐ Yes/No Answer ☒ Fixed Answers ☐ Individual Answers

6 

Enter an answer  
Your Answer  
Yes   
No   
+

Abb.7

Enter an Email:

7+8

9

10

Abb.8

Info

Poll submitted

This is the private key to this poll:

11 **UOyH8hFZAoC081wdu3ulRhCiTvt8cZvREPVoSeWL6kA=**

Please write it down or copy to a txt-file. Store it somewhere safe.  
You won't be able to access the results without it!

Abb.9

Nr.	Aktivität	Beschreibung
1	Enter Poll Title	Eingabe des Abstimmungstitels
2	Enter Expiration Date	Eingabe, wann die Abstimmung endet
3	Add Question	Hinzufügen einer Frage; Löschen einer Frage mit dem Button neben dem Fragetitel

<b>4</b>	<b>Enter Question Title</b>	<b>Eingabe des Fragetitels; Optional kann eine Beschreibung hinzugefügt werden (z.B. eine URL)</b>
<b>5</b>	<b>Choose Answer Type</b>	<b>Auswahl des Antworttyps (Ja/Nein, Feste Antworten, Eingabe)</b>
<b>6</b>	<b>Enter Fixed Answers</b>	<b>Eingabe der festen Antworten bei Auswahl; Löschen mithilfe des Icons</b>
<b>7</b>	<b>Enter Email</b>	<b>Eingabe der Email-Adressen</b>
<b>8</b>	<b>Add Email</b>	<b>Hinzufügen der eingegebenen Email</b>
<b>9</b>	<b>Retrieve Voters</b>	<b>Laden von bereits verwendeten Emails</b>
<b>10</b>	<b>Submit</b>	<b>Abschicken der Abstimmung</b>
<b>11</b>	<b>Save Private Key</b>	<b>Abspeichern des Private Keys; Kopie in die Zwischenablage entweder manuell oder per Button</b>

Bevor der Nutzer zum Hauptmenü zurückkehren möchte, ohne die Abstimmung abgeschickt zu haben, wird eine entsprechende Warnung angezeigt, dass die Änderungen verloren gehen. Das Gleiche gilt für das Abschicken einer (un)vollständigen Abstimmung (Vollständig = Titel, Ende, min. eine vollständige Frage, min. eine Email). Das sichere Abspeichern/Abschreiben des Private Keys ist unerlässlich für die anschließende Auswertung. Dieser wird nach Rückkehr ins Hauptmenü verworfen, daher ist es im Nachhinein nicht möglich, auf die Antworten zuzugreifen, wenn der Schlüssel verloren geht. Styling mit CSS und Angular Materials wurde von Michael Schmitt vorgenommen.

## 2.4 Verschlüsselung (JG)

An die Verschlüsselung wurden die Anforderungen gestellt, die Abstimmungsergebnisse im Browser des Voters zu verschlüsseln, sodass diese im Backend nicht auslesbar sind, und nach Übermittlung an den Browser des Surveyleaders wieder entschlüsselt werden zu können. Symmetrische Verschlüsselung verwendet für Ver- und Entschlüsselung den gleichen Schlüssel, wohingegen asymmetrische Verfahren einen öffentlichen Schlüssel (public key) zum verschlüsseln und einen geheimen Schlüssel (private key) zum entschlüsseln verwenden. Dies hat den Vorteil, dass der public key an viele Teilnehmer verschickt werden kann. Es bot sich also allgemein ein asymmetrisches Verfahren an, um die Anzahl der Schlüssel gering zu halten bzw. keinen symmetrischen Schlüssel für Ver- und Entschlüsselung an das Backend zu senden und dort abzuspeichern, was ein potentiell Sicherheitsproblem dargestellt hätte. Bei den asymmetrischen

Verschlüsselungsverfahren ist RSA das prominenteste, hat allerdings den Nachteil, dass oftmals sehr lange Schlüssel ( $\geq 2048$ -bit) benötigt werden, um einen adäquaten Schutz gegen Attacks zu bieten <sup>2</sup>. Ziel war es, die Schlüssellänge so gering wie möglich zu halten, ohne einen nennenswerten Verlust an Sicherheit zu erleiden. Als Alternative bot sich daher die Elliptic Curve Cryptography (ECC) an, welches Arithmetik auf elliptischen Funktionskurven ausführt und mit deutlich kürzeren Schlüssellängen bei gleichbleibender Sicherheit auskommt <sup>3</sup>. Mit der gewählten Bibliothek Tweetnacl-js und einer Schlüssellänge von 256-bit werden die empfohlenen Standards des Bundesministeriums für Informationssicherheit erfüllt und somit nach aktuellen Stand 2022 die Anforderungen an sicherer Verschlüsselung und somit Diskretion der Abstimmungsergebnisse erfüllt <sup>4</sup>. Eine Portierung von Tweetnacl-js nach Java steht ebenfalls zur Verfügung, falls bei einer möglichen Erweiterung des Systems der Bedarf an Ver- und Entschlüsselung im Backend bestünde. Die beiden Bibliotheken sind aufgrund der Verwendung von Base64-encoded Schlüsseln vollständig miteinander kompatibel <sup>5</sup>.

Tweetnacl-js ist eine JavaScript-Portierung der open-source Verschlüsselungsbibliothek Tweetnacl ("salt") von Daniel J. Bernstein (University of Illinois at Chicago and Technische Universität Eindhoven), Tanja Lange (Technische Universität Eindhoven), and Peter Schwabe (Radboud Universität Nijmegen) <sup>6</sup>. Das asymmetrische Verfahren verwendet hierbei die Curve25519 Diffie-Hellman Schlüsselaustauschfunktion, basierend auf Elliptic-Curve Cryptography (ECC)<sup>7</sup>. Die Bibliothek wurde von Cure53, Dr.-Ing. Mario Heiderich mit einem sehr guten Ergebnis auf Schwachstellen überprüft und hat alleine auf GitHub über 9 Mio. aktive Nutzer <sup>8</sup>. Die Einbindung erfolgte über den Packagemanager NPM.

Der Algorithmus berechnet intern einen gemeinsamen Schlüssel aus dem private key des Senders und dem public key des Empfängers. Weiterhin wird eine sog. "nonce" (eng. "flüchtig") benötigt, eine zusätzliche Komponente, welche für jede verschlüsselte Nachricht einzigartig sein muss, um das Erkennen von Mustern in der Nachricht zu erschweren. Tweetnacl-js bietet für die Generierung von Schlüsselpaaren und nonce alle notwendigen Funktionen. Die Funktion generateKeyPair des Encryption-Service ruft die Funktion box.keyPair() auf, welche ein 256-bit Schlüsselpaar erzeugt und als Base64 Zeichenkette codiert.

---

<sup>2</sup> <https://cr.yp.to/highspeed/coolnacl-20120725.pdf> [zuletzt aufgerufen 18.07.22, 09:00 Uhr]

<sup>3</sup> <https://cordis.europa.eu/docs/projects/cnect/6/216676/080/deliverables/002-DSPA20.pdf> [zuletzt aufgerufen 18.07.22, 09:00 Uhr]

<sup>4</sup>

[https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-2.pdf?\\_\\_blob=publicationFile](https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-2.pdf?__blob=publicationFile) [zuletzt aufgerufen 18.07.22, 09:30 Uhr]

<sup>5</sup> <https://github.com/InstantWebP2P/tweetnacl-java> [zuletzt aufgerufen 18.07.22, 09:30 Uhr]

<sup>6</sup> <https://github.com/dchest/tweetnacl-js> [zuletzt aufgerufen 07.06.2022, 10:45 Uhr]

<sup>7</sup> <https://nacl.cr.yp.to/securing-communication.pdf> [zuletzt aufgerufen 07.06.2022, 10:45 Uhr]

<sup>8</sup> <https://cure53.de/tweetnacl.pdf> [zuletzt aufgerufen 07.06.2022, 10:45 Uhr]

Beispiel:

```
{
    // 32-byte public key
    publicKey: +6crM+CVIR6g5WJmb/puhSdYEBPlexmw5wrPwJnoiHw=,
    //32-byte secretKey
    secretKey: JTGj4EJRNjWXWt47GV05BIHanpk+CRHKkVY/v3w0S+U=
}
```

Wenn nun ein Voter seine ausgefüllte Abstimmung abschickt, wird in dessen Browser ein flüchtiges Schlüsselpaar (“ephemeral public key”) und eine 24-byte nonce erzeugt. Anschließend werden die Antworten als Zeichenkette im JSON-Format mittels der box()-Funktion. Dadurch, dass die Daten mit einem flüchtigen Schlüssel des Voters und des public key des SurveyLeaders verschlüsselt wurden, wird dessen Authentizität sichergestellt, da der Empfänger die Daten nur mit dem öffentlichen Schlüssel und seinem private key entschlüsseln kann, andernfalls wäre dies nicht möglich.

Beispiel für ein EncryptedData-Objekt:

```
{ "nonce": "B3/TONhH8u0eiTp01NA0CTA43/dEMTaE",
  "ephemPubKey": "5LUyNZkgEviAO6qx3tFkrxdbt29sjsbRlXxm/ZosZG3s=",
  "message": "5G8ZJZ+E/mOmJ1YZFlq72cNOH6btw02Yn6JTAx/shmh+FtKc+4FAwWIHP9bi0ueyC4
cGa+pfCk/1WdAJNJCsrDb5ao5bsppXvu4wrMu1dMuo5CrLITtI03FtLWeNjhu9tuilA63XwgJ01Nn
52za7B411DMu5eSwmQIBQAjxz1KwUNniDwGVwepaiuk9M0QlRnfCGdcMlYWOHtzJaK+NJ2Oe/BX7+
t9SAgypcAt/FKElog19PifKFdyCjIkq785g8bX89Qwn8zrAM/H6W5iw/9ciwrKl1WG7t94ccrZGKY
XJNPYWJHZW44ZmzVideboDUaTOPTA==", "publicKey": "G0017bOr7WYgScemZrf4bw28igWIWJO
1LYuLMltY3Q4=" }
```

Korrespondierender private key:

uS/W2ViCSGguZhHOuKvp/GACKxoieeNbXS2OjaFY4Ho=

Gibt nun der SurveyLeader nach Abstimmungsende seinen private key in seinem Browser ein, werden die zuvor noch verschlüsselt vorliegenden Daten mithilfe der decrypt()-Funktion entschlüsselt, welche ein EncryptedData-Objekt sowie den private key als 32-byte Base64 Zeichenkette entgegen nimmt. Das Resultat ist bei korrektem private key der entschlüsselte Text.

Ergebnis mit oben genannten Beispielen:

```
{ "name": "Survey
Test", "lifetime": "2022-06-17T21:25", "questions": [{ "id": 1, "title": "Wollen wir
essen gehen?", "type": "yesNoAnswer", "visible": true, "description": "Lecker Essen
gehen in der Mensateria?" }], "emails": ["mycrush@fhws.de"] }
```

## 2.4.1 TLS (JG)

Die Verschlüsselung von HTTP hin zu HTTPS mittels TLS stellt einen integralen Bestandteil allgemeiner Cyber Security dar. Andernfalls könnten Angreifer HTTP-Nachrichten beliebig abfangen und ändern. Die Verschlüsselung wird mithilfe eines SSL Zertifikats und zugehörigen private key ermöglicht. Diese lassen sich lokal mit gängigen Programmen wie openssl erzeugen, allerdings werden diese selbst-signierten Zertifikate nicht von Browsern als sicher akzeptiert und es wird eine Warnung angezeigt, dass man die Website nicht besuchen sollte, bzw. nur auf eigenes Risiko. Ein akzeptiertes Zertifikat erhält man von kommerziellen Drittanbietern wie Hosting-Dienstleistern, welche die Authentizität des Zertifikats bestätigen. Die geheime Abstimmung Anwendung wurde bereits mit einem selbst-signierten Zertifikat konfiguriert, dementsprechend wird eine Warnung beim Aufruf der Seite angezeigt. Um das Zertifikat durch ein offizielles auszutauschen, sind nur kleine Anpassungen nötig. Zum einen muss im Frontend in der Datei "package.json" bei `--ssl-cert` der Pfad zum Zertifikat und bei `--ssl-key` der Pfad zur Schlüsseldatei angegeben werden.

```
"name": "frontend",
"version": "0.0.0",
"scripts": {
  "ng": "ng",
  "start": "ng serve --ssl true --ssl-cert src/ssl/certificate.pem --ssl-key src/ssl/key.pem",
  "build": "ng build",
  "watch": "ng build --watch --configuration development",
  "test": "ng test"
},
```

Abb.10

Im Backend muss ebenfalls nur der absolute oder relative Pfad zu Zertifikat und Schlüssel in der Main-Methode der Klasse Start geändert werden.

```
public static void main( final String[] args ) throws Exception
{
    Connector httpsConnector = new Connector();
    httpsConnector.setPort(8080);
    httpsConnector.setSecure(true);
    httpsConnector.setScheme("https");
    httpsConnector.setAttribute( name: "SSLCertificateFile", value: "../frontend/src/ssl/certificate.pem");
    httpsConnector.setAttribute( name: "SSLCertificateKeyFile", value: "../frontend/src/ssl/key.pem");
    httpsConnector.setAttribute( name: "clientAuth", value: "false");
    httpsConnector.setAttribute( name: "sslProtocols", value: "TLS");
    httpsConnector.setAttribute( name: "SSLEnabled", value: true);
}
```

Abb.11

## 2.5 Abstimmung (MS)

### 2.5.1 Überblick

#### Generell

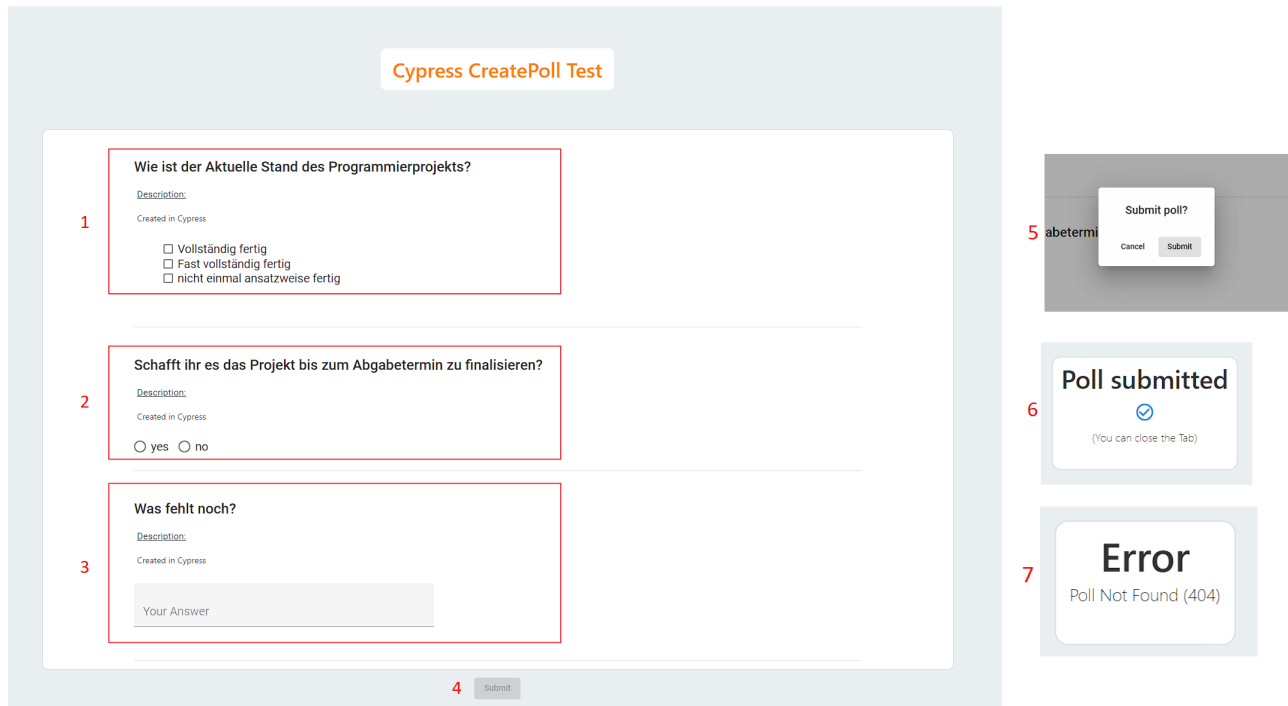


Abb.12

Nr.	Aktivität	Beschreibung
1	Ausfüllen von Fragetyp Multiple Choice	Checkboxen zum Ankreuzen der gewünschten Antwort
2	Ausfüllen vom Fragetyp Yes/No	Auswahlmöglichkeit zwischen yes und no
3	Ausfüllen vom Fragetyp Custom-Anwer	Eigene Antwort kann in das Textfeld eingegeben werden
4	Submit-Button	Submit-Button drücken um seine Antworten abzuschicken Ausgegraut solange nicht vollständig ausgefüllt.
5	Abstimmung endgültig absenden	Antworten ans Backend abschicken oder abbrechen und weiter bearbeiten
6	Ausgefüllte Abstimmung abgeschickt	Pop-Up nach erfolgreichen Absenden der Antworten



7	Fehlerfenster	Pop-Up auf der Seite im Falle eines gesendeten Fehlercodes vom Backend oder Timeout
---	---------------	---

## Mobile Support

**FHWS** POLL

Wie ist der Aktuelle Stand des Programmierprojekts?

Description:

Created in Cypress

☒ Vollständig fertig  
☐ Fast vollständig fertig  
☐ nicht einmal ansatzweise fertig

---

Schafft ihr es das Projekt bis zum Abgabetermin zu finalisieren?

Description:

Created in Cypress

☒ yes ☐ no

---

Was fehlt noch?

Description:

Um Teilnehmern die Möglichkeit zu bieten, von unterwegs Abstimmungen auszufüllen, wurde extra für diese Teilseite Mobile Support implementiert.

Abb.13

## 2.5.1 Funktionsweise

### Laden der Abstimmung

Wenn eine Person abstimmen möchte, muss sie auf den in der Email enthaltenen Link klicken. Daraufhin wird im Frontend die “/survey” Seite geöffnet. Beim Laden der Webseite werden aus der URL die Query-Parameter “pollId” und “token” gelesen, zwischengespeichert und damit dann einen Call an das Backend gemacht. Dort wird überprüft ob die “pollId” gültig ist, sowie ob der “token” zur jeweiligen Umfrage passt. Im Falle, dass einer Daten falsch ist, schickt das Backend einen Http-Fehlercode, welcher das Frontend einen Fehlermeldung anzeigen lässt. Sind die Daten aber korrekt, sendet das Backend die Abstimmung, welche dann im Frontend dynamisch geladen und dargestellt wird.

### Reactive Forms

Um sicher zu stellen, dass die Abstimmung auch komplett ausgefüllt ist, bevor man sie abschickt, werden “Reactive Forms” verwendet. Dabei handelt es sich um eine Angular-Funktion, welche es ermöglicht Form-Inputs zu überprüfen, was mit Hilfe von “Validators” umgesetzt wird.

Nach einer erfolgreichen Anfrage ans Backend, wird eine Antwort mit Details zur Abstimmung, wo auch ein Array mit Fragen enthalten ist, gesendet. Daraufhin wird eine neue “Form-Group” erstellt. Das Array wird dann als Liste im Frontend angezeigt und die Fragen werden je nach Inhalt, nach Typ und ob sie eine Beschreibung enthalten dementsprechend angezeigt. Beim Laden der einzelnen Fragen wird jeweils eine “Form Control” hinzugefügt worin dann ein oder mehrere “Validator“ definiert werden. Im Falle von Ja/Nein Fragen und eigene Antworten wird einfach nur vorgegeben, dass sie ausgefüllt werden müssen. Bei Multiple Choice Fragen hingegen wird eine neue Form Group erstellt, welche immer noch Teil der anderen ist. Diese neue Form Group hat einen “Custom Validator” der überprüft, ob mindestens eine der Antwortmöglichkeiten angeklickt wurde.

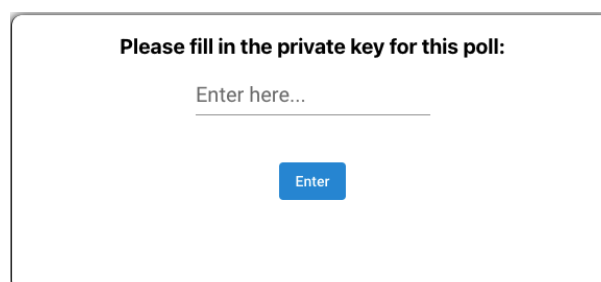
Erst dann, wenn alle Form Controls gültig sind, wird der Submit Button zum Klicken freigegeben.

### Abschicken der Antworten

Sobald man in dem Pop Up auf Submit Poll klickt, werden die Antworten in einem bestimmten Format umgewandelt, welches dann mit dem jeweiligen Public Key verschlüsselt wird. Dieses Paket wird dann, mit der dazugehörigen “pollID” und “token” an das Backend geschickt, wo es erneut die Daten prüft und bei gültigen Parametern einen Http 204 code schickt und den benutzten Token aus der Datenbank entfernt.

## **2.6 Auswertung der Umfrage (JG)**

Ist die Abstimmung beendet, können die Ergebnisse ausgewertet werden. Der Nutzer klickt im Hauptmenü auf die entsprechende Umfrage und muss als nächstes den zugehörigen private key eingeben.



The image shows a web form with a light gray border. At the top, it says "Please fill in the private key for this poll:". Below this is a text input field with the placeholder text "Enter here...". Underneath the input field is a blue button with the word "Enter" in white text.

Abb.14

Sollte der private key falsch sein, wird ein Timeout aktiviert, welcher sich bei jeder erneut falschen Eingabe verdoppelt. Dies soll Brute Force Attacken, also das Durchprobieren aller möglichen Schlüssel, erschweren. Die Ergebnisse liegen bis zur Eingabe des korrekten Schlüssels verschlüsselt vor.

Abb.15

Ist der private key korrekt und die Abstimmungsergebnisse folglich entschlüsselt, kann der Nutzer diese einsehen und bei Bedarf einzeln als PDF abspeichern.

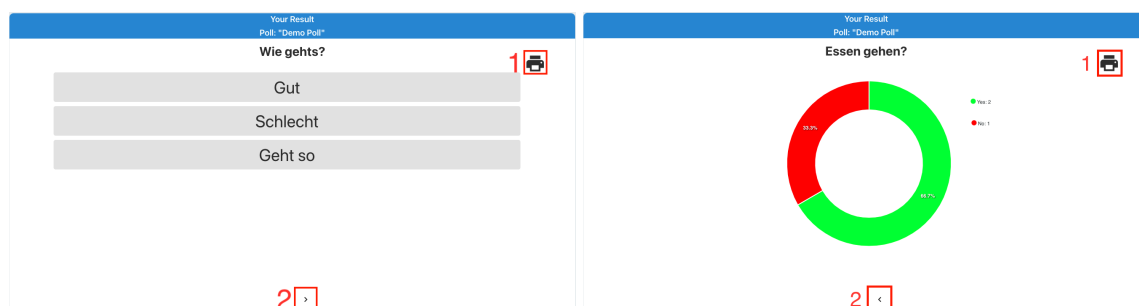


Abb.16

Nr.	Aktivität	Beschreibung
1	PDF erstellen	Erstellt ein PDF mit dem Titel der Umfrage sowie Titel und Ergebnisse der aktuellen Frage
2	Nächste/Vorherige Frage	Wechselt zur nächsten bzw. vorherigen Frage

Bei einer Frage mit Texteingabe des Voters wird eine Liste mit den Antworten angezeigt. Bei Ja/Nein-Fragen und bei festen Antwortmöglichkeiten wird ein Kreisdiagramm mit prozentualer Auswertung der Antwortenverteilung angezeigt und auch im PDF übernommen. Hierfür wurde die Bibliothek ApexCharts.js verwendet, eine open-source JavaScript Bibliothek für die Erstellung von Diagrammen in Frontend Frameworks wie Angular und React <sup>9</sup>. Hierfür werden lediglich die Antwortmöglichkeiten als Label sowie die zugehörige Anzahl an Stimmen benötigt.

<sup>9</sup> <https://apexcharts.com> [zuletzt aufgerufen 13.07.2022, 10:45 Uhr]

## 2.7 Testen in Cypress (MS)

### Einführung

Bei Cypress handelt es sich um ein JavaScript end-to-end Test-Framework welches ermöglicht mit selbst geschriebenen Tests die Funktionsweise des Frontends einer Webseite auf unterschiedlichen Arten zu untersuchen.

### Überblick:

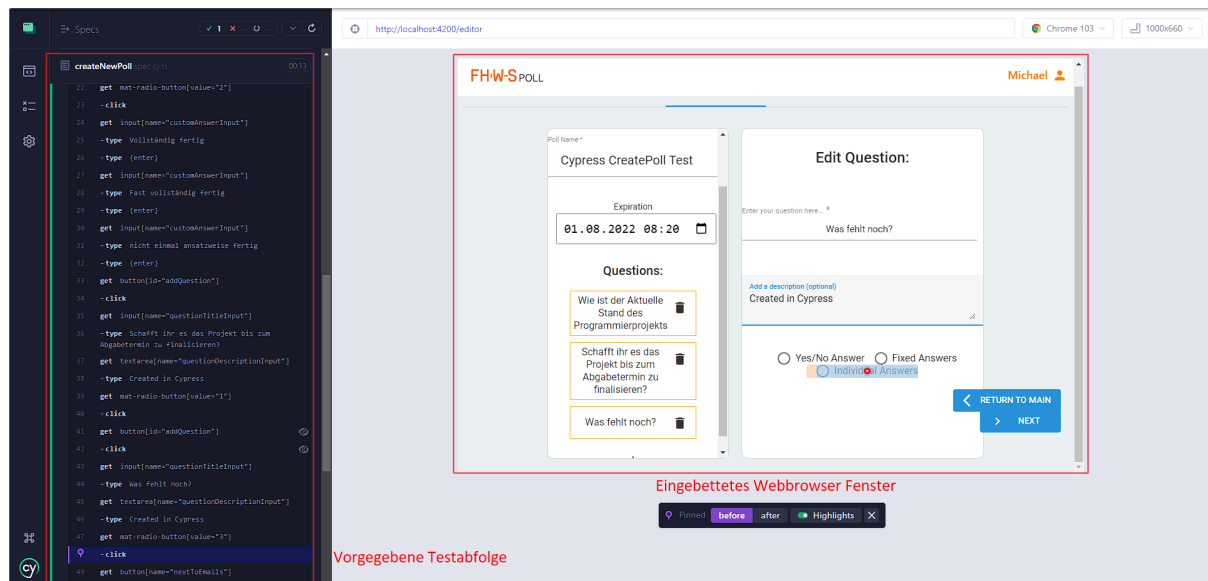


Abb.17

In unseren Projekt haben wir hauptsächlich das Login-System, den Editor und das Hauptmenü getestet. Für Cypress wurde ein extra Account mit Adminrechten angelegt um uneingeschränkt die Funktionen des Frontends testen zu können.

### Login

Nr.	Art des Tests
1	Login mit korrekten Daten und Weiterleiten zur Hauptseite
2	Login Versuch mit richtiger E-Mail plus falschen Passwort fünf mal und prüfe auf Timeout-Benachrichtigung
3	Login und Logout via das Optionsmenü auf der Hauptseite
4	Login und Logout via "Fast Logout"

## Editor

Der Editor besitzt nur einen Test welcher eine Umfrage mit allen drei Fragetypen erstellt. Dieser wurde hauptsächlich benutzt, um Funktionen anderer Komponenten zu testen, wie die Seite zum Beantworten der Umfrage oder die Auswertung und Darstellung der Antwortdaten.

## Hauptmenü

Nr.	Art des Tests
1	erfolgreiches Hinzufügen eines neuen "Surveyleader" über das Optionsmenü
2	erfolgreiches Ändern des Passworts über das Optionsmenü
3	erfolgreiches Ändern des Benutzernamens über das Optionsmenü
4	erfolgreiches Hinzufügen und Entfernen eines "Surveyleader" über das Optionsmenü
5	Logout Funktionen wurden beim Punkt 'Login' angesprochen

## 2.8 API (JG)

Die Basis-URL zur Kommunikation ist im aktuellen Entwicklungszustand "https://localhost:8080/api/polls". Im Frontend wurde rxjs als Bibliothek für die HTTP-Requests verwendet. Daten zwischen Front- und Backend werden im JSON-Format über folgende Endpunkte ausgetauscht:

Endpunkt	HTTP-Verb	Aktivität
?sessionID=\$sessionID	GET	Lade alle Abstimmungen des Users
/users?sessionID=\$sessionID	GET	Lade alle User (nur Admin)
?sessionID=\$sessionID	POST	Erstelle eine neue Abstimmung
?pollID=\$pollID &sessionID=\$sessionID	DELETE	Lösche eine Abstimmung
/session?sessionID=\$sessionID	GET	Neue SessionID laden
/emails?sessionID=\$sessionID	GET	Zuletzt verwendete Emails laden

/users?userName=\$userName &sessionID=\$sessionID	DELETE	Lösche einen User (nur Admin)
/users?sessionID=\$sessionID	POST	Erstelle einen neuen User
/users?sessionID=\$sessionID	PUT	Ändere das Passwort oder Username
/answers/\$id?token=\$token	GET	Lade eine Abstimmung zur Teilnahme
/answers/\$id?token=\$token	POST	Schicke die ausgefüllte Abstimmung ab
answers?sessionID=\$sessionID &pollID=\$pollID	GET	Lade die Abstimmungsergebnisse

### 3. Backend

#### 3.1 Server (LK)

Der Server wurde über einen embedded Tomcat Container implementiert.

#### 3.2 Hinweise zur Systemeinrichtung (LK)

Die Anwendung arbeitet mit einer lokalen MongoDB Datenbank und besteht aus 6 Collections. Die Datenbank wird beim Start der Anwendung automatisch initialisiert. Um die Einrichtung anderer User zu ermöglichen, wird beim Systemstart ein User "initialUser" erstellt.

**Der "initialUser" Account besitzt Admin Rechte und sollte nur zur Erstellung des Accounts des eigentlichen Systemadministrators verwendet und anschließend gelöscht werden.**

**Zugangsdaten "initialUser":      E-Mail: "initialUser@mail.com"**  
**Password: "Geheime Abstimmung"**

### 3.3 Struktur der Datenbank (LK)

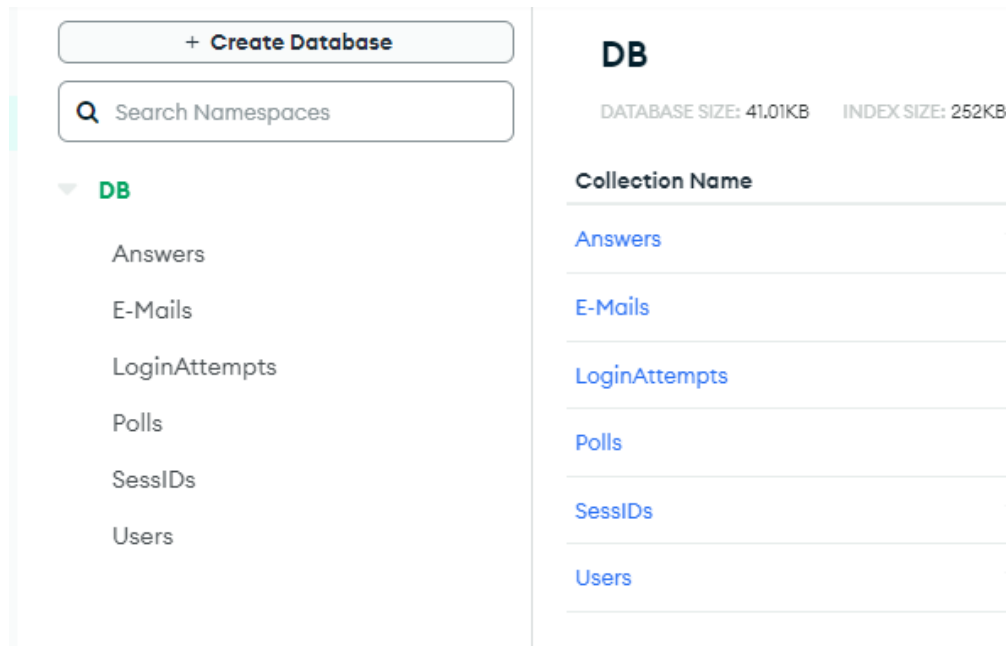


Abb.18

#### Answers:

Die Answers Collection speichert die verschlüsselten Antworten der Voter. Sie enthält für jede Abstimmung ein Dokument, dass die "poll\_id" der jeweiligen Abstimmung und einen Array aus den verschlüsselten Antworten enthält (s. "3.8 Speichern der verschlüsselten Antworten").

```
_id: ObjectId("62d2a3d457dbf55511afbffa")
poll_id: "62d2a3d457dbf55511afbffa"
answers: Array
  0: Object
    nonce: "+vE095ZmsFCjn6qUs0Sw89QGTlod7Xab"
    ephemPubKey: "eLxXpsDalF51vLu28KEPruiH/Izm7wr8iWmHuOXVZQ0="
    message: "7655DzeMfHzHG006sp3eZH6RX62DGNFPEBZ3Iunhi5sGvtZ6TaIuNV8t85YfcbG6XUJTt9..."
    token: "686706f7288b-8"
  1: Object
    nonce: "LDHwFKoIj+i0jkJ0p0GvpwZ60HHnapf2"
    ephemPubKey: "DGUQkvtkI/Xhi67a2aDF6Cn5Q6K9Le0S3CH67p9Hwmo="
    message: "Ux5wUPiTzi/ChhfA8hjSEYp8B5A6dzKBxEmp029VpndZFL9Ty2ToR8nDZzLLOyVw8UpumH..."
    token: "686701adab5c-6"
```

Abb.19

#### E-Mails:

Die E-Mails Collection enthält die E-Mail Adressen, an die die zuletzt erstellte Umfrage eines Users gesendet wurde. Die bereits gespeicherten E-Mail Adressen werden für jede neu erstellte Umfrage eines Users ersetzt und aktualisiert (vermeidet das "Zumüllen" der Datenbank mit selten verwendeten E-Mail Adressen).

```

    _id: ObjectId("62d2dcc735c5900fe92c4299")
  E-Mails: Array
    0: "leachim.s@t-online.de"
    created by: "@michael"

```

Abb.20

### LoginAttempts:

Die LoginAttempts Collection speichert nicht erfolgreiche Login-Versuche auf jeden User Account. Nach 5 erfolglosen Anmeldeversuchen wird ein Timeout ausgelöst (s. "3.6 Timeout Funktion"). Die Einträge in der LoginAttempts Collection sind zeitsensitiv und werden nach 30 Minuten ohne Veränderungen oder bei einem erfolgreichen Login automatisch gelöscht.

```

    _id: ObjectId("62d6d9d84a73d326143f0a86")
    email: "ernst.blofeld@fhws.de"
    attempts: 5
    created at: 2022-07-19T16:20:40.636+00:00
    Timeout until: 2022-07-19T16:20:40.636+00:00

```

Abb.21

### Polls:

Die Polls Collection speichert die erstellten Abstimmungen der User. Ein typischer Eintrag ist wie folgt aufgebaut:

```

    _id: ObjectId("62c86982a63a1f34a2835371")
    name: "Umfrage4"
    lifetime: "2022-07-21T19:29"
  questions: Array
    0: Object
      id: 1
      title: "dsfsdf"
      type: "yesNoAnswer"
      visible: true
      publicKey: "6efHKgVXrsjujRU15h50CwZuJLa1xLzd9y/C3MaG5nI="
      created by: "Max Mustermann"
  tokens: Array
    0: "7856315942cd-6"

```

Abb.22

- publicKey: Dient der Frontend-seitigen Verschlüsselung der Antworten
- tokens: Dienen der Anonymisierung der Voter (s. "3.4 Token Generierung")

### SessIDs:

Die SessIDs Collection verwaltet alle momentan gültigen Session IDs (s. "3.3 Sessions im Backend"). Jeder Eintrag enthält die Session ID und die Daten des zugehörigen Users. Alle Einträge in der SessIDs Collection sind zeitsensitiv und werden nach 60 Minuten ohne Veränderungen automatisch gelöscht.



```

_id: ObjectId("6295d2eee4b040720efc4cf1")
email: "ernst.blofeld@fhws.de"
name: "Ernst Blofeld"
role: "admin"
salt: "KYRfggRDWfgwQdEMGCwPPQ=="
pwHash: "RRfjJcqTXXSTbB+f+XUaVw=="
created at: 2022-07-19T16:21:40.921+00:00
Session ID: "009214101947-e"

```

Abb.23

## Users:

Die Users Collection dient dem Speichern der Benutzerdaten. Jeder User besitzt einen Namen, eine E-Mail Adresse, ein Passwort und eine Rolle (für genauere Informationen).

```

_id: ObjectId("62cfd8d103fc5676a0caa24f")
email: "testMail@gmail.com"
name: "DummyUser"
role: "admin"
salt: "+Nf89otMO2MMWZoalF1P5Q=="
pwHash: "0YbT/0tn+ZJa3x1sdWni1g=="

```

Abb.24

## 3.4 Sessions im Backend (LK)

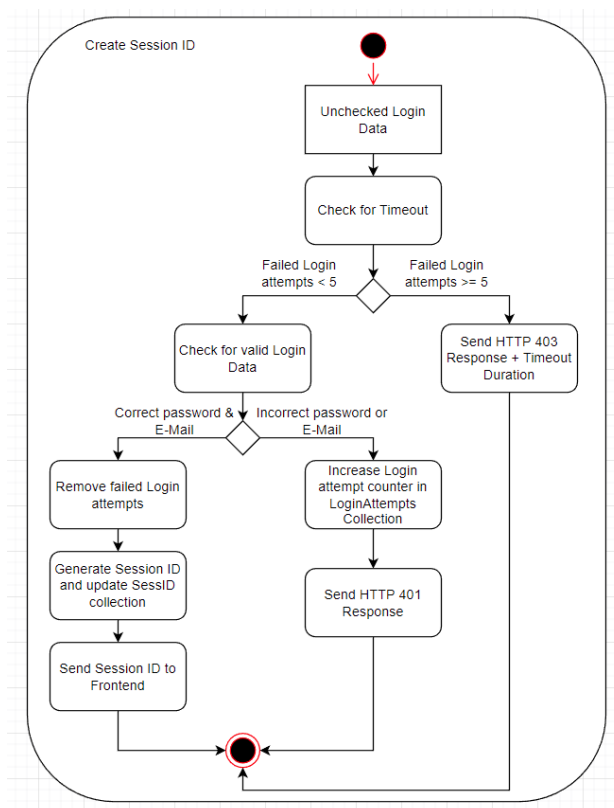


Abb.25

Sobald ein User versucht sich einzuloggen, wird eine entsprechende Anfrage mit den Login Daten an den Server gesendet. Der Server überprüft die übergebenen

Daten auf Gültigkeit und sendet, falls die Daten korrekt sind, eine für 60 Minuten gültige Session ID ans Frontend zurück.

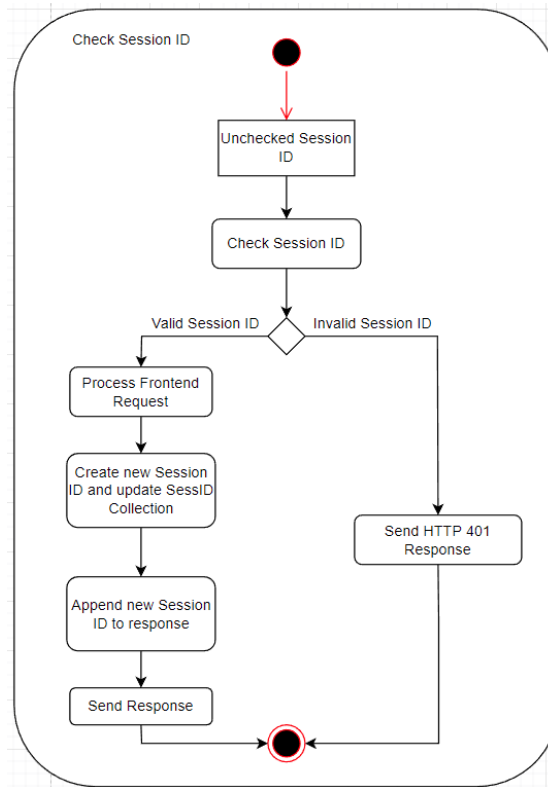


Abb.26

Bevor im Backend Anfragen verarbeitet werden können, muss die Session ID überprüft werden.

Hierbei wird die übermittelte Session ID mit der in der Datenbank abgeglichen und bei Übereinstimmung durch eine neue Session ID ersetzt. Die neue Session ID wird der Antwort des Servers angehängt.

Die Session IDs werden wie folgt generiert:

```
String sessID = String.valueOf(System.currentTimeMillis()).substring(8, 13) + UUID.randomUUID().toString().substring(1, 10);
user.append("created at", new Date());
```

Abb.27

### 3.5 Speicherung der User Login-Daten (LK)

Die Login Daten eines neuen Users werden vom Frontend im JSON Format an das Backend gesendet. Im Backend wird die JSON-Datei zunächst darauf geprüft, ob sie die notwendigen Attribute enthält und ob der anfragende Nutzer (Identifikation per Session ID) die notwendigen Berechtigungen zur Erstellung des neuen Nutzers besitzt (Ein Surveyleader darf bspw. andere Surveyleader, aber keinen Admin als User hinzufügen). Anschließend wird die übergebene E-Mail Adresse zunächst auf ihre Einzigartigkeit überprüft und gegebenenfalls eine entsprechende Fehlermeldung

an das Frontend weitergegeben. Das Passwort eines jeden Users wird unter Nutzung des PBKDF2 (Password-Based Key Derivation Function 2) mit 65536 Iterationen, einer Schlüssellänge von 128 Stellen und einem zufälligen 16 Byte Salt verschlüsselt. Der daraus berechnete Hash und der Salt müssen zu einem Base64-String konvertiert werden, da MongoDB Byte-Arrays nicht ohne weiteres verarbeiten kann. Der Hash, der Salt, die Email, die User-Rolle und der Username werden anschließend in die Datenbank geschrieben (s. "3.2 Struktur der Datenbank"). Bei zukünftigen Login Versuchen auf den jeweiligen User Account wird das übergebene Passwort mit dem gespeicherten Salt gehasht und mit dem Hash in der Datenbank verglichen.

Nachdem der User erfolgreich in der Datenbank vermerkt wurde, werden seine E-Mail Adresse und das übergebene Passwort an den Mail-Verteiler weitergegeben (s. "3.10 User E-Mail").

### 3.6 Token Generierung

Eine der Anforderungen an das System ist, dass die Voter beim Abstimmen anonym bleiben und die Antworten auf eine Poll nicht zu den Votern zurückverfolgt werden können.

Um dieses Ziel zu erreichen, werden die E-Mail Adressen der Teilnehmer im Backend aus der Poll entfernt und durch zufällig generierte Tokens ersetzt. Je einer dieser Tokens wird dann je einem Abstimmungslink angehängt und an den Mail Verteiler weitergeleitet. Welche E-Mail Adresse zu welchem Token gehört, wird nicht gespeichert.

Die Verwendung der Tokens ermöglicht zu einem die Anonymisierung der Teilnehmer und zum anderen den Schutz vor Mehrfachabstimmung, da jeder Token nach einmaliger Verwendung aus der Datenbank gelöscht wird.

### 3.7 Speichern der verschlüsselten Antworten (LK)

```
_id: ObjectId("62d5f015347d3d4ec0a5f533")
poll_id: "62d5f015347d3d4ec0a5f532"
answers: Array
  0: Object
    nonce: "+XoZF9k54/3kcrtxIxwvRHEIX239jzc9"
    ephemPubKey: "wDpXIKj1lnzjiSUC3yDbWrkhZ+q5HTsv0sAlYm6VSCw="
    message: "zGu6Bd0Th5052LvrPEHkjuLZxQEYHUEQKPbaz3Fh3PKDcJFeLmZ2rA=="
    token: "97270d54ba4f-e"
```

Abb.28

Die Antworten werden völlig anonym und verschlüsselt in der Datenbank gespeichert. Sie sind nicht zu den E-Mail Adressen der Abstimmenden zurückzuverfolgen.

Sobald ein Voter auf einen validen Abstimmungslink klickt, werden im Backend die Fragen der Poll und der zugehörige Public-Key an das Frontend gesendet. Nachdem der Voter seine Antworten gegeben und abgesendet hat, wird der jeweilige Token aus dem Poll Dokument in der Datenbank entfernt und die verschlüsselte Antwort gespeichert.

### 3.8 E-Mail Verteiler (TB)

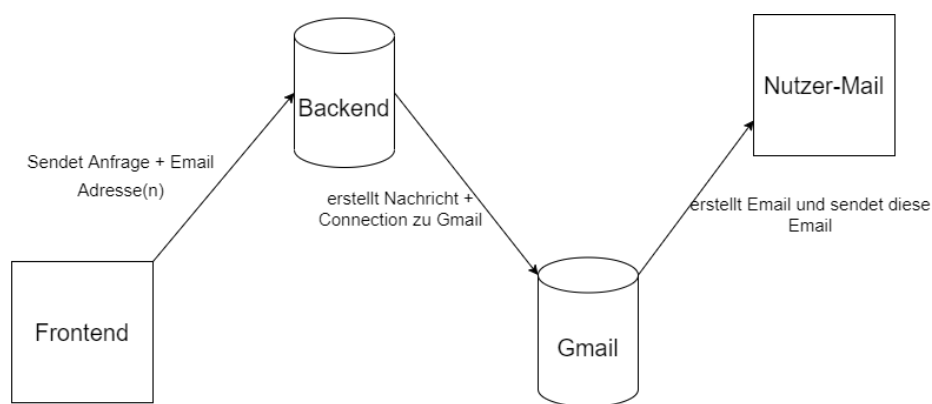


Abb. 29

Nr	Aktivität	Beschreibung
1	Frontend Anfrage	Das Frontend stellt eine Anfrage an das Backend und übergibt E-Mail(s)
2	Nachricht erstellen	Im Backend wird die Nachricht, welche in der E-Mail stehen soll, erzeugt
3	Verbindung zu Gmail	Über den Port 465 wird nun vom Backend zu einem Gmail Server die Verbindung hergestellt. Die erstellte Nachricht wird übergeben.
4	E-mail erstellen	Die E-Mail wird mit der übergebenen Nachricht erstellt und die übergebene(n) E-Mail Adresse(n) wird als Empfänger angegeben.
5	E-Mail versenden	Die E-Mail wird an den User versendet

Der E-Mail Verteiler wird durch eine Anfrage vom Frontend getriggert. Dies passiert wenn im Frontend eine Umfrage erstellt worden ist und die E-Mails der jeweiligen Teilnehmer angegeben wurde. Durch das endgültige erstellen der Umfrage wird eine Anfrage an das Backend gesendet und dem E-Mail Verteiler werden die E-Mail Adressen übergeben. Nun fertigt das Backend für jede dieser E-Mail Adressen eine vorprogrammierte E-Mail Nachricht an, mit jeweils einzigartigem Link, über den die Nutzer schlussendlich abstimmen kann. Das Backend erstellt nun eine Verbindung zu einem Gmail Server, dies geschieht über den SSL-Port 465 und einer Passwort Authentifizierung, welche mit hilfe eines App-Passworts Zugang zu dem Account "[secret.vote.project@gmail.com](mailto:secret.vote.project@gmail.com)" erhält. Hierbei werden nun die vom Backend übergebenen Daten in einen E-Mail Body eingefügt und an den Teilnehmer versendet. Eine E-Mail, die der Teilnehmer bekommt könnte zum Beispiel so aussehen:

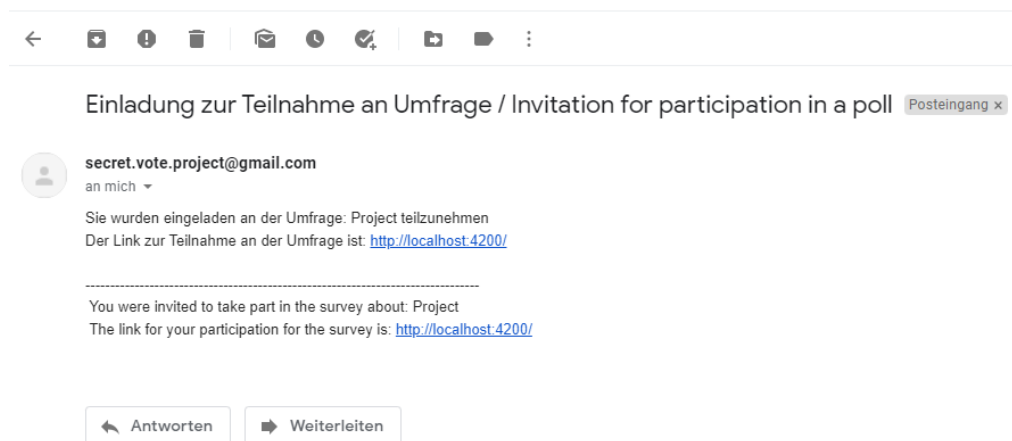


Abb.30

Wie in Abb.2 nun zu erkennen ist wird die E-Mail sowohl in der deutschen Sprache, als auch in der englischen Sprache versendet. Ebenfalls ist zu erkennen, das auch der Name der Umfrage mit eingefügt wird um dem Teilnehmer schon einmal zu zeigen um was es bei der Umfrage geht. Zuletzt kann man noch erkennen das ein Link mitgeschickt wird, welcher wie schon erwähnt zur eigentlichen Umfrage bzw. Abstimmung führt.

### 3.9 User E-Mail

Für die User E-Mail, welche Zugangsdaten für neu erstellte User zu Verfügung stellen soll, gilt ein ähnlicher Aufbau wie für den eigentlichen E-Mail Verteiler. Eine User E-Mail ist dann vonnöten, wenn ein neuer User angelegt wird. Die Unterschiede sind im Grunde nur, dass nur an eine E-Mail Adresse eine E-Mail versendet wird und andere Daten übergeben bzw der E-Mail hinzugefügt werden. Eine Beispiel E-Mail einer User E-Mail sieht in etwa so aus (kein funktionierender Account) :

## User Anmeldung für Poll Application / User Registration of Poll Application Posteingang x

secret.vote.project@gmail.com

an mich ▾

Hallo neuer Nutzer,  
du wurdest soeben als ein neuer Nutzer der Poll Application festgelegt. Deinen neuen Account kannst du unter folgendem link erreichen: <http://localhost:4200/>

Deine Zugangsdaten sind:

Username: Project  
Passwort: Project2022

Bitte ändere dein Passwort, wenn du dich zum erten mal in deinen Account einloggst. So ist dein Account sicherer vor einem unbefugten Einloggen in dein Account.

-----  
Hello new User,  
You were registered as a new User for the Poll Application, you can now log in into your new account with the following link: <http://localhost:4200/>

For the first login your username and password are:

Username: Project  
Password: Project2022

Please change your password after logging in the first time, so that your account is safe.

Abb.31

In Abb.3 ist nun zu erkennen, dass der Link mitgeschickt wird, über den man sich nach Erhalt der Zugangsdaten anmelden kann. Ebenfalls werden unter dem Punkt Username und Passwort die Zugangsdaten aufgelistet. Auch hier werden die Sprachen deutsch und englisch angeboten, damit auch jeder den Sinn der E-Mail verstehen kann. Als letzter Punkt ist noch aufgelistet, dass man schnellstmöglich sein Passwort ändern soll, dies dient zur Sicherheit des Accounts vor unbefugten Zugriffen auf den Account.

## 4. Mögliche Erweiterung des Systems

Nr.	Mögliche Erweiterung
1	Eine Abstimmung wird automatisch als beendet gekennzeichnet, wenn alle Teilnehmer abgestimmt haben
2	Im Zusammenhang mit Erweiterung Nr. 1 erhält der Surveyleader eine Email, wenn alle Teilnehmer abgestimmt haben oder die Frist abgelaufen ist
3	Installation des Systems über eine Progressive Web App (PWS) oder Desktop Programm
4	Möglichkeit, Dokumente/Bilder an Fragen anzuhängen
5	Mehr Flexibilität bei der Auswahl von Fragen
6	Verbesserte Email-Validierung