Prof. Dr. Phillip Heidegger
phillip.heidegger@hs-augsburg.de

Version vom: 2.6.2025
Zu Bearbeiten bis: 24.8.2025

Blatt 5 - ENTWURF

Software Systeme
Webanwendungen, verteilte Systeme und Sicherheit

A 5.1: JSON-Webtokens — 20 Punkte

Gehen Sie davon aus, dass Ihr Zeichenprogramm um einen Mechanismus erweitert werden soll, der es Nutzer erlaubt, sich auf Ihrer Webseite einzuloggen. Dies soll mithilfe von Email und Passwort möglich sein.[1]

Ihre Lösung für die Aufgabe sollte wie folgt aussehen: Sie erstellen einen HTTP Server mit drei Seiten. Diese lauten:

- `login`, Pfad `/login`: Auf dieser Seite können Sie sich mithilfe von Mail und PW einloggen. Bei erfolgreichem Login leitet der Server den Client auf die Seite `home`. Bei alle nachfolgenden Requests werden mit einem JWT die Identität des Nutzers durch eine HTTP-Only-Cookie sichergestellt.

  Es gibt einen Link zur Seite `register`.

- `register`, Pfad `/register`: Auf dieser Seite können Sie für neue Mailadressen ein PW vergeben, und damit einen Account erstellen.

  Es gibt hier einen Link zur Seite `login`.

- `home`, Pfad `/`:

  Ist das JWT gültig, wird eine leere Zeichenfläche des Programms von Blatt 3 angezeigt. Es gibt in diesem Schritt keinen Zustand auf dem Server, von der Information der Account abgesehen.

  Ist das JWT ungültig oder nicht vorhanden, wird der Nutzer auf die Seite `login` weitergeleitet.

  Das JWT soll die Emailadresse des eingeloggten Users zur Identifikation beinhalten. Der Check, ob das JWT gültig ist, muss ohne Zugriff auf Zustand auf eine DB im Backend erfolgen.

Fortsetzung auf der nächsten Seite.

[1] Sie benötigen keinen Mechanismus, um die Email zu verifizieren.

A 5.2: UserIDs und Anzeigenamen 10 Punkte

Schaffen Sie eine Möglichkeit, dass jeder User außerdem einen Anzeigenamen wählen kann. Dieser wird im nachfolgenden Blatt verwendet, um anderen Nutzer anzuzeigen, welche Rechte welcher Nutzer bzgl. welcher Zeichenfläche besitzt, und welche Nutzer gerade aktiv online an der Zeichenfläche arbeiten.

Außerdem müssen Sie sicherstellen, dass jeder User eine eindeutige, kurze User-ID erhält. Diese wird verwendet, um Nutzer auf der Seite anderen Nutzer gegenüber zu identifizieren, und sollte unverändert sein über die gesamte Nutzungsdauer der Software. D.h. Sie kann nicht geändert werden, im Gegensatz zur Mailadresse oder dem Anzeigename.

A 5.3: Zeichenflächen 20 Punkte

Gehen Sie nun davon aus, Sie wollen Ihr Zeichenprogramm in eine Multiuser Plattform umbauen.

Erweitern Sie die Informationen des JWT dahingehend, dass Sie folgende Datenstruktur ins JWT aufnehmen:

- Eine Liste von Canvas-IDs zusammen mit einem Buchstaben aus der Liste: R, W, V, M, CO, O.

Geben Sie als Zwischenstand diese Datenstruktur nach Login auf der Console des Programms aus. Nach einem refresh sollte genau die selbe Datenstruktur nochmals ausgegeben werden, ohne das er Server hierzu in seiner Datenbank Informationen abfragen muss.

Hinweis: Sie können davon ausgehen, dass die Canvas-IDs Strings sind, z.B. in Form eine UUID. Alternativ wäre auch eine Hash in Form von z.B. einem sha1 Hash denkbar.

A 5.4: Rechtesystem 50 Punkte

Die Zeichen aus Aufgabe 5.3 stellen ein Rechtesystem zur Verfügung. Es basiert auf dem IRC Protokoll:

- R: Read-Only: Der User kann die Zeichenfläche lesen, d.h. er kann Sie sehen und findet diese in seiner Liste.
- W: Er hat Schreibenderechte, und kann, die ZF bearbeiten, falls diese nicht im Zustand “moderiert” ist.
- V: Er hat Schreibenderechte, und kann dies auch bearbeiten, falls die im Zustand “moderiert” ist,
- M: Er hat moderatorrechte und kann die Zeichenfläche bearbeiten, und die Zeichenfläche in den Zustand “moderiert” versetzten. Er kann Rechte verteilen, und anderen User Rechte bis zur Stufe “V” geben.
- O: Er besitzt die Zeichenfläche, und kann die Rechte vergeben für alle User.

A 5.5: Multiuser 350 Punkte

Erweitern Sie nun Ihr Backend aus dem vorherigen Blatt, so dass mehrere Personen in der Lage sind, gleichzeitig an einer Zeichenfläche zu arbeiten.

- Ihr Backend soll die benötigten Ressourcen, d.h. .html, .js, .css etc. Dateien, an den Client (Browser) ausliefern.
- Ihr Client soll eine Single-Page-Webanwendung sein. Sie sollen für alle gültigen URLs den Inhalt einer generischen HTML-Datei ausgeliefert. Es gibt die folgenden Seiten:
  - Eine Übersichtsseite. Dies entspricht der Seite `home` aus Blatt 4. Allerdings soll auf Ihr jetzt die Liste mit allen für den User zugänglichen Zeichenflächen angezeigt werden.
  - Eine Seite, die die Zeichenflächen darstellt. `http://.../canvas/<ID>`
  - Die Seiten `login` und `register` aus dem Blatt 4.

  Auf jeder dieser Seiten wird der Client direkt eine Websocketverbindung zum Server aufbauen (URL: `ws://.../channel/`). Nutzen Sie die Informationen aus dem Blatt 4 bzgl. der Userinformation, d.h. der Mailadresse, um einen Client eindeutig zu identifizieren. Beachten Sie, das die Mailadresse eine private Information darstellt, und andere Nutzer keine Kentniss über die Mailadresse der anderen Personen haben sollten, auch wenn diese auf einer Zeichenoberfläche gemeinsam arbeiten. Berechnen Sie basierend auf der Mailadresse eine kurze, eindeutige ID, die keinen Rückschluss auf die Mailadresse ermöglichst, um die jeweiligen Clients eindeutig zu identifizieren.

  Diese ID sollte der Client nutzen, um IDs für Shapes zu generieren. Auf diese Art und Weise kann sichergesestellt werden, dann nicht zwei Clients auf der selben Canvas zwei Shapes mit den selben IDs erzeugen.
- Ihr Backend werden mehrere Zeichenflächen, die durch IDs identifiziert werden, unterscheiden. Diese werden jeweils unter URLs der folgenden Form bereitgestellt:

  `http://<IhrHost>/canvas/<ID>`

  D.h. wenn ein Nutzer die URL `http://.../canvas/ganzLangeId` im Browser öffnet, sollte ein Zeichenprogramm sichtbar werden, dass ähnlich aussieht wie das Zeichenprogramm aus den vorherigen Blättern.

  Ruft aber nun ein weiterer Nutzer die URL `http://.../canvas/ganzLangeId` auf, so können diese beiden auf der selben Zeichenfläche gemeinsam zusammenarbeiten, falls der Nutzer die passenden Rechte besitzt.

  Damit niemand die IDs erraten kann, sollten diese IDs zufällig generiert werden, und möglichst nicht zu kurz sein.

Es sollte eine Navigationsmöglichkeit von den Zeichenfläche zu der Übersichtsseite geschaffen werden. Diese sollte ohne "reset" der JS-VM umgesetzt werden (bitte eine SPWA).

- Auf der Übersichtsseite (URL: `http://..../`, d.h. ohne Pfad) sollte eine Liste der existierenden Zeichenflächen sichtbar sein.[2]

  Außerdem sollten auf der Überssichtsseite eine Möglichkeit vorhanden sein, eine neue Zeichenfläche zu erzeugen. Dies bedeutet, dass der Server eine neue ID generiert, diese dem Client mitteilt, und dieser dann die passende Zeichenfläche anzeigt. (Ohne Refresh! SPWA!)

- Wie oben erwähnt, soll zur Umsetzung der notwendigen Kommunikation zwischen Client und Server während der Sitzung der Websocket (`ws://.../channel/`) eingesetzt werden. Über diesen werden Anfrage vom Client an den Server gestellt, am besten in Form von JSON-Anfragen. Ein Bsp. für eine Anfrage des Clients an den Server mit der Bitte, den Client von nun an über alle Events, die die Zeichenfläche mit der ID `ganzLangeId` betreffen, zu informieren:

```
{
  "command": "registerForCanvas",
  "canvasId": "ganzLangeId" }
```

  Im Fall, dass ein Client sich für eine Zeichenfläche registriert, sollte der Server z.B. so antworten:

```
{ "canvasId": "ganzLangeId",
  "eventsForCanvas": [
    { "event": "addShape",
      "shape": { "type": "line",
                 "id": "ID1",
                 "data": { "from": { "x": 5, "y": 5 },
                           "to": { "x": 100, "y": 100 }
                           "zOrder": 5,
                           "bgColor": undefined, // transparent
                           "fgColor": "000000"   // black
                         }
               }
    },
    ...
  ]
}
```

[2] Hier wäre später noch denkbar, diese Welcome Page zu personalisieren, so dass Nutzer nur die Zeichenflächen sehen, die diese "kennen". Da die IDs zufällig sein sollen, und lang genug sind, könnte auf diese Art und Weise ein Management der Zeichenflächen umgesetzt werden. Langfristig natürlich noch mit einen Rechtemanagmentsystem. Diese Aspekte dürfen Sie alle ignorieren, da dadurch die Aufgabe zu umfangreich wird.

Ab diesem Zeitpunkt muss der Server, jedes Mal wenn ein anderer Client die Zeichenfläche manipuliert, den Client über Änderungen informieren, indem der Server passende Infopacket verteilt.

Wenn im Client der Nutzer dann wieder auf die Welcome Seite wechselt, sollte durch

```
{ "command": "unregisterForCanvas",
  "canvasId": "ganzLangeId" }
```

die Registrierung durch den Client für die Zeichenfläche entfernt werden. Ab diesem Zeitpunkt wird der Server den Client nicht mehr über Änderungen an der Canvas informieren.

Wenn ein Client eine Zeichenfläche anzeigt und der Nutzer Shapes erstellt, verschiebt, etc., muss der Client den Server über alle Änderungen informieren, so dass dieser die Informationen an alle anderen Clients, die die Zeichenfläche auch anzeigen, informieren kann. Ein Bsp. für das Verschieben eines Shapes:

```
{ "canvasId": "ganzLangeId",
  "eventsForCanvas": [
    { "event": "removeShape",
      "shapeId": "ID1",       // s.o., die Linie (5,5) -- (100,100)
    },
    { "event": "addShape",
      "shape": { "type": "line",
                 "id": "ID2",
                 "data": { "from": { "x": 25, "y": 25 },
                           "to": { "x": 120, "y": 120 },
                           "zOrder": 5,
                           "bgColor": ...,
                           "fgColor": ...
                         }
               }
    },
  ]
}
```

Als Konsequenz sollte auf allen Zeichenprogrammen, die die Canvas mit ID `ganzLangeID` anzeigen, die Linie von `(5,5)-(100,100)` um 20 Pixel nach recht und 20 Pixel nach unten bewegt werden (So wird es für den User aussehen, auch wenn es in Wirklichkeit einfach die eine Linie entfernt wird, und eine neue erstellt wird mit den passenden Koordinaten. Dieses Verhalten ist passend zu der API des Zeichenprogramms, da die Canvas nur mit funktionalen Shapes arbeitet. Dies sollten Sie nicht ändern!).

- Empfehlung: Versuchen Sie, die Anzahl der Events auf der Canvas möglichst gering zu halten. Events für die folgenden vier Interaktionen sollten reichen:
  - AddShape — Infos: shape type, und die Daten des shapes
  - RemoveShapeWithId — Infos: die id
  - SelectShape — Infos: shapeId, clientId
  - UnselectShape — Infos: shapeId, clientId