



Hochschule für Technik  
und Wirtschaft Berlin

University of Applied Sciences

## DOKUMENTATION

---

# UniSketch7

## AUTOREN

Florian Stallmach

Christopher Lorenz

Talisa Wahle

Eduard Seiler

Dennis Adler

## ZEITRAUM

WiSe 2019/2020

## MODUL

B56 Projektstudium (P) - UniSketch - WiSe2019/20

## DOZENT

Prof. Dr.-Ing. Johann Habakuk Israel

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>7</b>
<b>2</b>	<b>Benutzung</b>	<b>8</b>
2.1	Startseite . . . . .	8
2.2	Dashboard . . . . .	9
2.3	Skizze . . . . .	11
2.3.1	Cursor . . . . .	12
2.3.2	Brush . . . . .	13
2.3.3	Graffiti-Brush . . . . .	16
2.3.4	Background . . . . .	17
2.3.5	Grid . . . . .	17
2.3.6	Text . . . . .	18
2.3.7	Shapes . . . . .	20
2.3.8	Bilder einfügen . . . . .	22
2.3.9	Eraser . . . . .	23
2.3.10	Selector . . . . .	23
2.3.11	Chat . . . . .	24
2.3.12	Download . . . . .	24
2.3.13	Participants . . . . .	25
2.3.14	Fullscreen . . . . .	26
2.4	Usermenu . . . . .	26
2.4.1	Profilbild . . . . .	28
2.5	Notifications . . . . .	28
<b>3</b>	<b>Entwicklung</b>	<b>29</b>
3.1	Entwicklungsprozess . . . . .	29
3.1.1	Voraussetzungen . . . . .	29
3.1.2	Klonen des Repositories und allgemeiner Aufbau . . . . .	30
3.1.3	Installation der Abhängigkeiten . . . . .	30
3.1.4	Erstellen der Datenbanken . . . . .	30
3.1.5	Anpassen der Konfigurationsdateien . . . . .	31
3.1.6	Testen der Anwendung . . . . .	34

3.2	Backend . . . . .	34
3.2.1	REST-Endpoints . . . . .	34
3.2.2	Websockets . . . . .	35
3.2.3	Datenbank . . . . .	35
3.2.4	Skizzen . . . . .	35
3.2.5	Chat . . . . .	36
3.3	Frontend . . . . .	36
3.3.1	Komponenten . . . . .	36
3.3.2	Services . . . . .	36
3.3.3	Skizzenview . . . . .	37
3.3.4	Responsiveness . . . . .	38
3.4	REST-Endpoints . . . . .	38
3.5	Websocket-Messages . . . . .	39
<b>4</b>	<b>Deployment</b>	<b>41</b>
4.1	Konten und Passwörter . . . . .	41
4.2	Installation . . . . .	43
4.2.1	Kopie der Repositories . . . . .	43
4.2.2	Subdomain für neue UniSketch-Version anlegen . . . . .	43
4.2.3	Neue Datenbank erstellen . . . . .	44
4.2.4	Skripte anpassen . . . . .	45
4.2.5	Build-Ordner der Anwendung einrichten . . . . .	46
4.2.6	Variable im Quellcode der Anwendung anpassen . . . . .	50
4.2.7	To whom it may concern . . . . .	50
4.3	Deployment eines neues Entwicklungsstands . . . . .	51
<b>5</b>	<b>Entwicklungsverlauf und implementierte Features im WiSe2019/2020</b>	<b>52</b>
5.1	Scrum . . . . .	52
5.1.1	Rollen . . . . .	52
5.1.2	Definition of Done . . . . .	52
5.1.3	Sprints . . . . .	52
<b>6</b>	<b>Weiterentwicklung</b>	<b>55</b>
6.1	Offene Punkte aus dem Product-Backlog . . . . .	55
6.2	Bekannte Bugs, Fehler und generell Verbesserungsfähiges . . . . .	55
6.3	Anmerkungen für Entwickler . . . . .	56

# Abbildungsverzeichnis

2.1	Sign Up . . . . .	8
2.2	Die Login-Maske . . . . .	8
2.3	Funktion zum Zurücksetzen des Passworts . . . . .	9
2.4	Öffentliche Skizzen . . . . .	9
2.5	Das Dashboard mit zwei Skizzen im aktiven Ordner . . . . .	10
2.6	Tags für Skizzen . . . . .	11
2.7	Eine neue Skizze . . . . .	12
2.8	Cursor für das Verschieben von ausgewählten Elementen . . . . .	13
2.9	Cursor für den Brush . . . . .	13
2.10	Cursor für das Shape-Tool . . . . .	13
2.11	Cursor für das Text-Tool . . . . .	13
2.12	Cursor für das Eraser-Tool . . . . .	13
2.13	Auswahl an Pinseln . . . . .	14
2.14	Optionen Pinsel . . . . .	14
2.15	Farbe des Pinsels . . . . .	15
2.16	Linienstärke des Pinsels . . . . .	15
2.17	Deckkraft des Pinsels . . . . .	15
2.18	Erweiterung Brush . . . . .	16
2.19	Graffiti . . . . .	16
2.20	Hintergrundfarbe der Skizze . . . . .	17
2.21	Grid . . . . .	18
2.22	Farbe des Textes . . . . .	19
2.23	Schriftart des Textes . . . . .	19
2.24	Schriftgröße des Textes . . . . .	19
2.25	Deckkraft des Textes . . . . .	20
2.26	Auswahl an Formen. Beispiel Dreieck . . . . .	20
2.27	Füll- und Umrissfarbe der Formen . . . . .	21
2.28	Umrissstärke der Formen . . . . .	21
2.29	Deckkraft der Formen . . . . .	22
2.30	Upload eines Bildes . . . . .	22
2.31	Eingefügetes Bild . . . . .	23
2.32	Chatfunktion . . . . .	24

2.33	Download Optionen . . . . .	25
2.34	Rechte verteilen . . . . .	26
2.35	Benutzermenü . . . . .	27
2.36	Benutzerprofil . . . . .	27
2.37	Auswahl eines Profilbildes . . . . .	28
2.38	Geändertes Profilbild . . . . .	28
3.1	Das Datenbankschema . . . . .	40

# Tabellenverzeichnis

3.1	Hauptkomponenten des Frontends . . . . .	37
3.2	Services des Frontends . . . . .	38

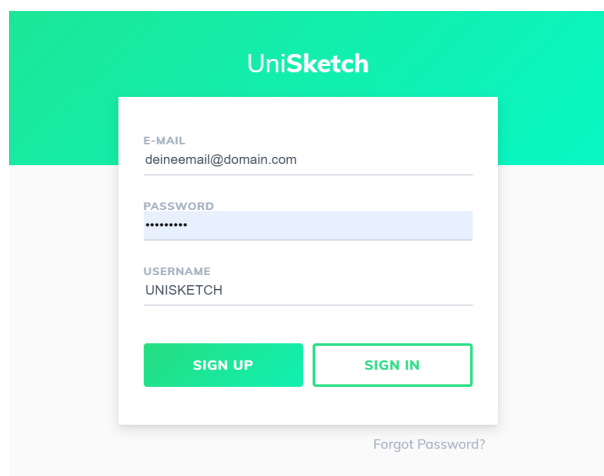
# 1 Einleitung

UniSketch ist eine Webanwendung zum kollaborativen Erstellen von Skizzen in Echtzeit. Benutzer können auf verschiedenen Endgeräten (Desktop, Tablet, Smartphone) per Klick neue Skizzen erstellen, in Ordnern organisieren und sie gemeinsam mit anderen Nutzern bearbeiten. Zum Skizzieren stehen diverse Skizzierwerkzeuge zur Verfügung, hierzu gehören verschiedene Stifte, geometrische Objekte, Bilder uvm. Den genauen Aufbau der Anwendung aus Nutzersicht finden sie im Kapitel 2 Benutzung. Die technischen Hintergründe des Systems sind im Kapitel 3 Entwicklung niedergeschrieben. Abschnitt 4 Deployment bietet eine Schritt-für-Schritt Anleitung, wie ein neues Projekt aufzusetzen ist, sowie ein neuer Entwicklungsstand auf den Server geladen werden kann. Die Änderungen und Erweiterungen, die das Team im Wintersemester 2019/20 umgesetzt und programmiert hat sind im Kapitel 5 Neuerungen zusammengefasst. Empfehlungen zur möglichen Weiterentwicklung des Projektes, sowie bekannte Fehler befinden sich im letzten Part 6 Weiterentwicklung.

## 2 Benutzung

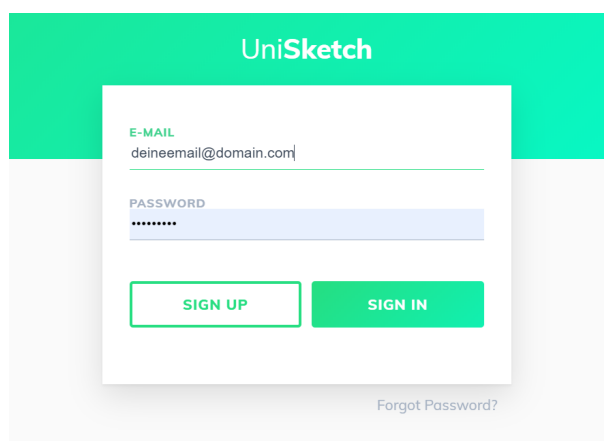
### 2.1 Startseite

Auf der Startseite von UniSketch werden dem Nutzer die Möglichkeit zum Login und zur Registrierung eines neuen Accounts gegeben (siehe Abb. 2.2). Für die Erstellung eines neuen Accounts muss ein Benutzername, eine Email-Adresse und ein Passwort eingegeben werden. Nach Absenden des Formulars wird der Benutzer automatisch eingeloggt und gelangt zum Dashboard.



The image shows a web form for signing up on UniSketch. The form is centered on a light gray background with a teal header bar at the top containing the text "UniSketch". The form itself is white with a subtle shadow. It contains three input fields: "E-MAIL" with the placeholder "deineemail@domain.com", "PASSWORD" with a masked password "\*\*\*\*\*", and "USERNAME" with the placeholder "UNISKETCH". Below these fields are two teal buttons: "SIGN UP" and "SIGN IN". At the bottom right of the form, there is a link that says "Forgot Password?".

Abbildung 2.1: Sign Up



The image shows a web form for logging in on UniSketch. The form is centered on a light gray background with a teal header bar at the top containing the text "UniSketch". The form itself is white with a subtle shadow. It contains two input fields: "E-MAIL" with the placeholder "deineemail@domain.com" and "PASSWORD" with a masked password "\*\*\*\*\*". Below these fields are two teal buttons: "SIGN UP" and "SIGN IN". At the bottom right of the form, there is a link that says "Forgot Password?".

Abbildung 2.2: Die Login-Maske



Falls der Nutzer bereits einen Account besitzt und sein Passwort vergessen hat, so kann er sein Passwort über den Link *Forgot Password* zurücksetzen (siehe Abb. 2.3). In diesem Fall bekommt er eine Email mit einem Link zugesendet, welcher es ermöglicht, das Passwort zurückzusetzen.

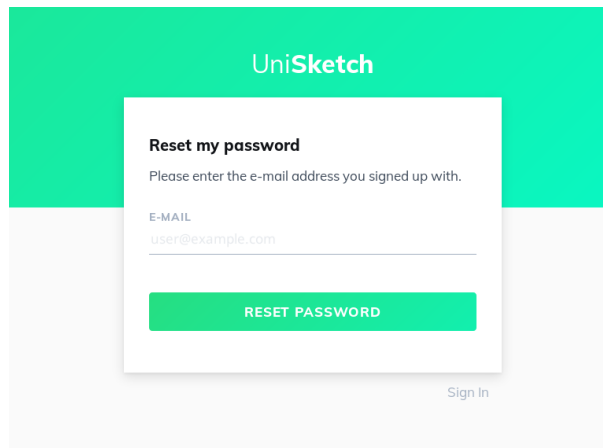
The image shows a web form for resetting a password. It has a teal header with the 'UniSketch' logo. The main form is white with a teal border. It contains the title 'Reset my password', a prompt 'Please enter the e-mail address you signed up with.', an input field for the email address with the placeholder 'user@example.com', and a teal button labeled 'RESET PASSWORD'. A 'Sign In' link is located at the bottom right of the form area.

Abbildung 2.3: Funktion zum Zurücksetzen des Passworts

Im unteren Teil der Startseite befindet sich die *UniSketch Gallery*, wo alle öffentlichen Skizzen zu finden sind (siehe Abb. 2.4). Diese können ohne Anmeldung angesehen, jedoch nicht verändert werden.



Abbildung 2.4: Öffentliche Skizzen

## 2.2 Dashboard

Im Dashboard kann der Benutzer seine Skizzen, Ordner und Profildaten verwalten. Das Dashboard ist unterteilt in eine zweizeilige Toolbar am oberen Rand und die Skizzenübersicht. Hier werden sowohl Skizzen angezeigt, welche man selbst erstellt hat, als auch solche, zu denen man von einem anderen Benutzer Zugriff erhalten hat (diese erscheinen dann im *Home*-Ordner). Des

Weiteren befindet sich auf der linken Seite die Ordnerstruktur. Der aktuell geöffnete Ordner ist farblich hervorgehoben. Durch Klick auf das Ordner-Icon lässt sich ein neuer Ordner erstellen. Alternativ kann durch Rechtsklick auf einen vorhandenen Ordner auch ein neuer Unterordner erstellt (und auch wieder gelöscht) werden.

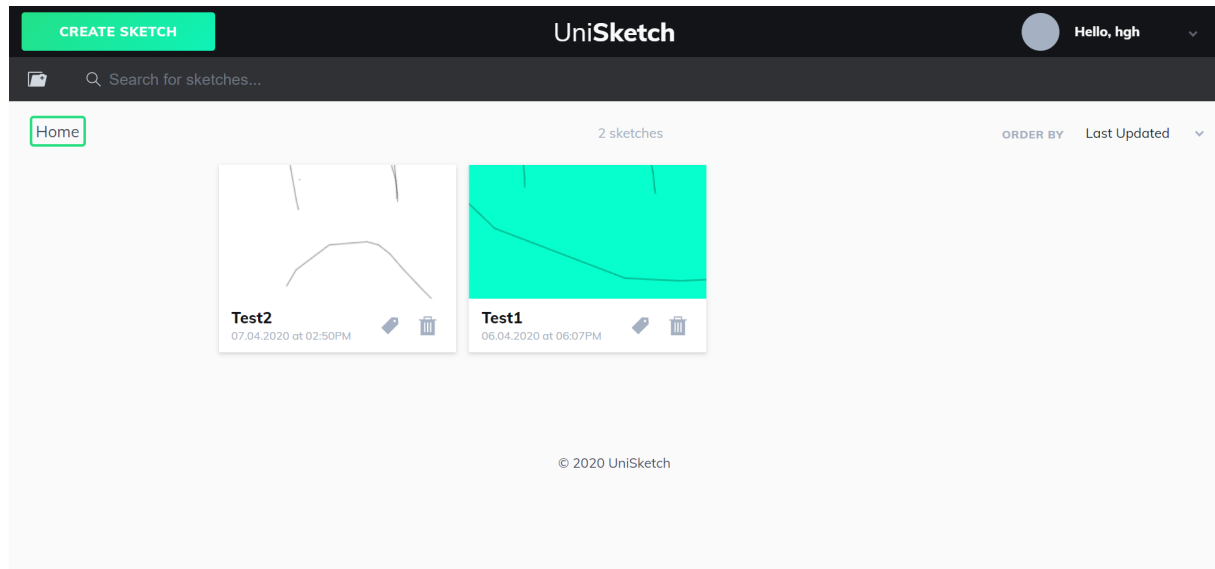


Abbildung 2.5: Das Dashboard mit zwei Skizzen im aktiven Ordner

Weiterhin gibt es die Möglichkeit über das Tag-Icon einer Skizze passende Schlagwörter hinzuzufügen (siehe Abb. 2.6). Mit einem Klick auf das Icon öffnet sich ein Dialogfenster, in dem man die Tags hinzufügen oder entfernen kann. Zusätzlich gibt es die Möglichkeit Skizzen durch die Eingabe eines Suchworts oder Tags über das Suchfeld zu filtern sowie die Skizzen nach Name oder Datum zu sortieren.

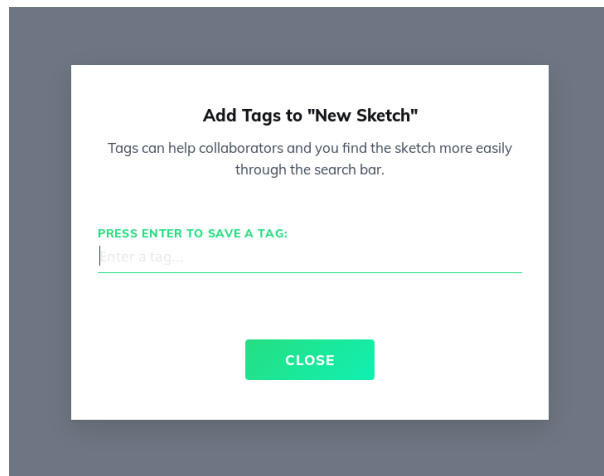


Abbildung 2.6: Tags für Skizzen

Ein Klick auf das Vorschaubild einer Skizze öffnet sie in der Skizzenview. In gleicher Weise kann durch Klick auf *Create Sketch* eine neue Skizze erstellt werden. Zu jeder Skizze wird das letzte Änderungsdatum angezeigt. Mithilfe des Mülleimer-Icons kann man eine Skizze löschen bzw. aus der Bibliothek entfernen. Bei Klick auf den Skizzennamen erscheint ein Textfeld, mit dem man der Skizze einen neuen Titel geben kann. Eine neue Skizze wird immer im aktiven Ordner erstellt. Durch Drag and Drop kann eine Skizze in einen anderen Ordner verschoben werden.

## 2.3 Skizze

In der Skizzenansicht kann der Benutzer Skizzen bearbeiten. Die Skizzenansicht ist unterteilt in eine zweizeilige Toolbar am oberen Rand sowie die Zeichenfläche.

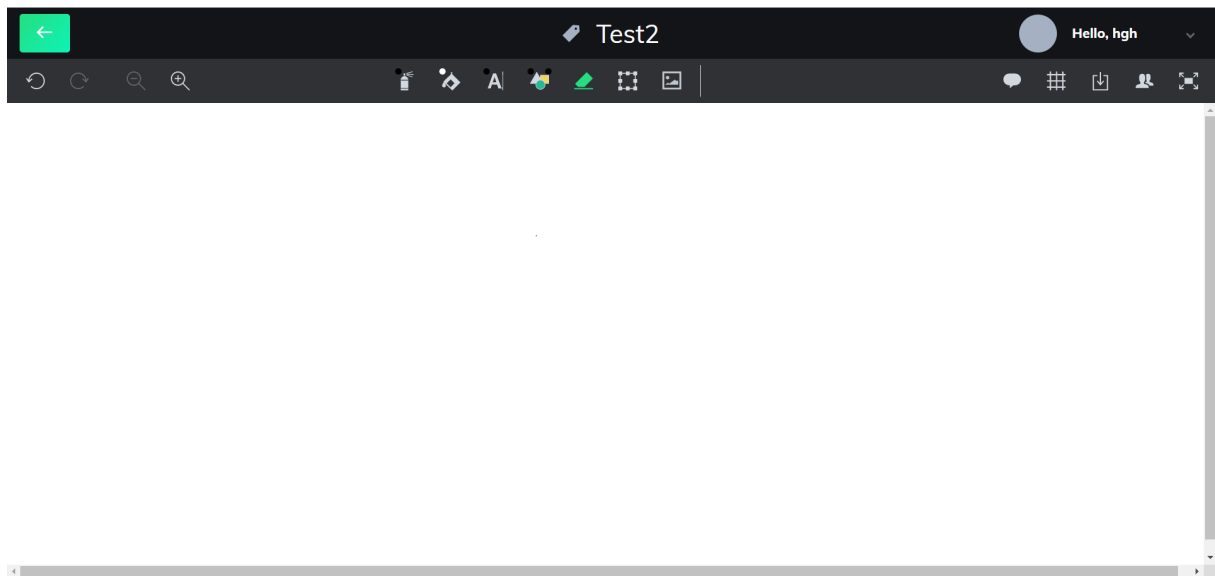


Abbildung 2.7: Eine neue Skizze

In der oberen Zeile der Toolbar finden sich die Funktionen zum Verlassen der Skizze sowie zum Umbenennen der Skizze als auch zum Versetzen mit Tags.

In der zweiten Zeile der Toolbar befinden sich links Buttons zum Rückgängig-machen und Wiederherstellen der Aktionen als auch ein Slider mit Buttons zum Hinein- und Hinauszoomen. Auf der rechten Seite befinden sich Buttons für die Chat-Funktion, für das Herunterladen der Skizze in verschiedenen Formaten, für das Festlegen der Kollaborateure, zum einblenden eines Grids sowie für den Vollbildmodus.

In der Mitte der zweiten Zeile der Toolbar befinden sich die Skizzenwerkzeuge. Dabei dient ein horizontaler Balken als Trennlinie zwischen den Skizzenwerkzeugen und deren jeweiligen Einstellungen. Ein Klick auf ein Skizzenwerkzeug blendet alle zugehörigen Einstellungen rechts der Trennlinie ein. Dabei unterscheiden sich die Einstellungen von Werkzeug zu Werkzeug. Durch Schweben des Mauszeigers über ein Einstellungs-Icon wird ein Dialog zum Steuern eingeblendet.

### 2.3.1 Cursor

Dem Nutzer wird je nachdem welches Werkzeug er ausgewählt hat, der dazu passende Cursor angezeigt.



Abbildung 2.8: Cursor für das Verschieben von ausgewählten Elementen



Abbildung 2.9: Cursor für den Brush

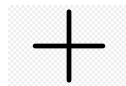


Abbildung 2.10: Cursor für das Shape-Tool

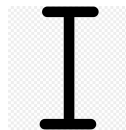


Abbildung 2.11: Cursor für das Text-Tool



Abbildung 2.12: Cursor für das Eraser-Tool

### 2.3.2 Brush

Der Pinsel bietet die Möglichkeit freihand zu Zeichnen. Dabei können Farbe (siehe Abb. 2.15), Linienstärke (siehe Abb. 2.16) und Deckkraft (siehe Abb. 2.17) eingestellt werden. Es kann sowohl mit der Maus durch Linksklick als auch durch evtl. vorhandene Touchfunktion (ggf. mit Zeichenstift) gezeichnet werden. Des Weiteren können verschiedene Versionen gestrichelter Linien

ausgewählt werden, jedoch überlappen sich die Segmente dieser Linien ab einer Strichstärke von 5. Zur Zeit ist noch nicht visualisiert, welche Linie gerade ausgewählt wurde.

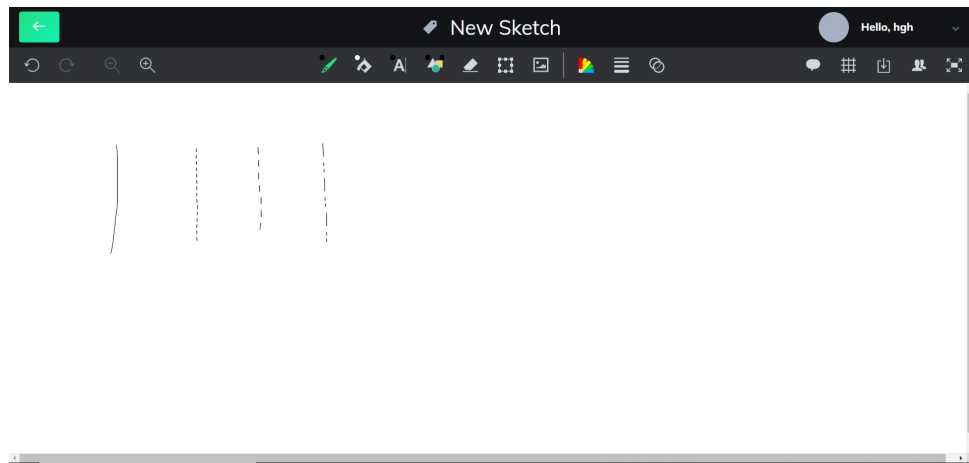


Abbildung 2.13: Auswahl an Pinseln

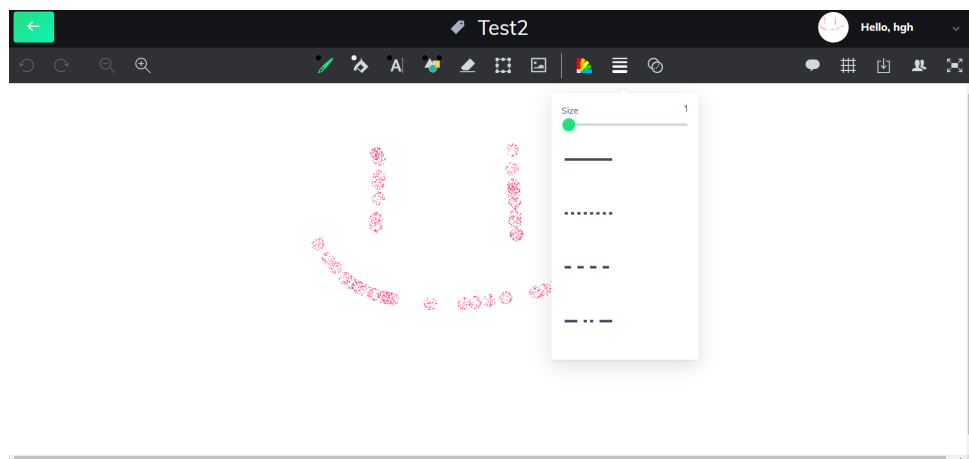


Abbildung 2.14: Optionen Pinsel

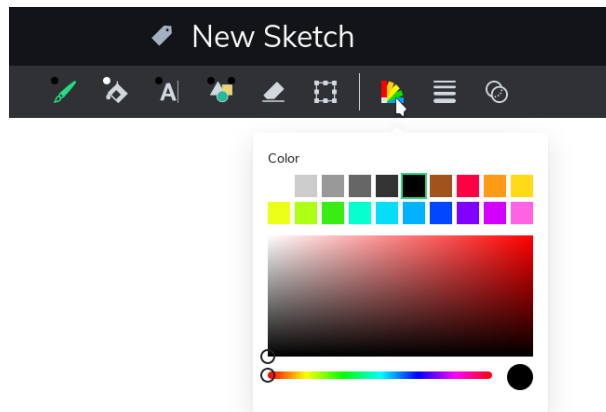


Abbildung 2.15: Farbe des Pinsels

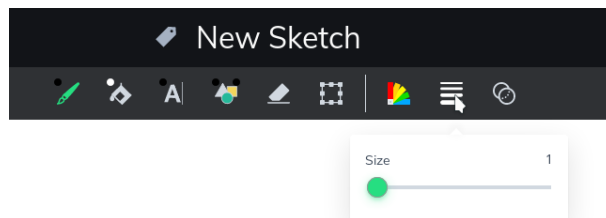


Abbildung 2.16: Linienstärke des Pinsels

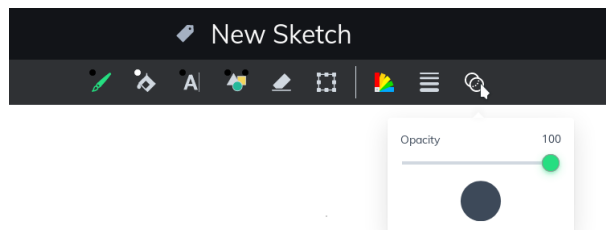


Abbildung 2.17: Deckkraft des Pinsels

### 2.3.3 Graffiti-Brush

Der Graffiti- Brush ist eine Erweiterung des gerade beschriebenen Tools und besitzt somit auch alle Funktionen des Pinsels. Durch einen Linksklick wird ein Graffiti-Pattern auf der Skizze erstellt. Das Pattern ist rund und wird mit vielen einzelnen Punkten rund um die gezogene Linie befüllt. Je nach Geschwindigkeit des Ziehens der Linie steigt der Abstand der Kreise zueinander. Der Graffiti-Brush kann zur Zeit nicht radiert werden. Wie in Abbildung 2.18 zu sehen ist, erscheint durch Bewegen des Mauszeigers auf das Brush-Icon ein Menü in welchem der Graffiti-Brush ausgewählt werden kann.

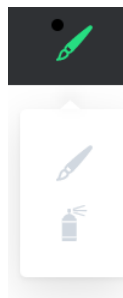


Abbildung 2.18: Erweiterung Brush

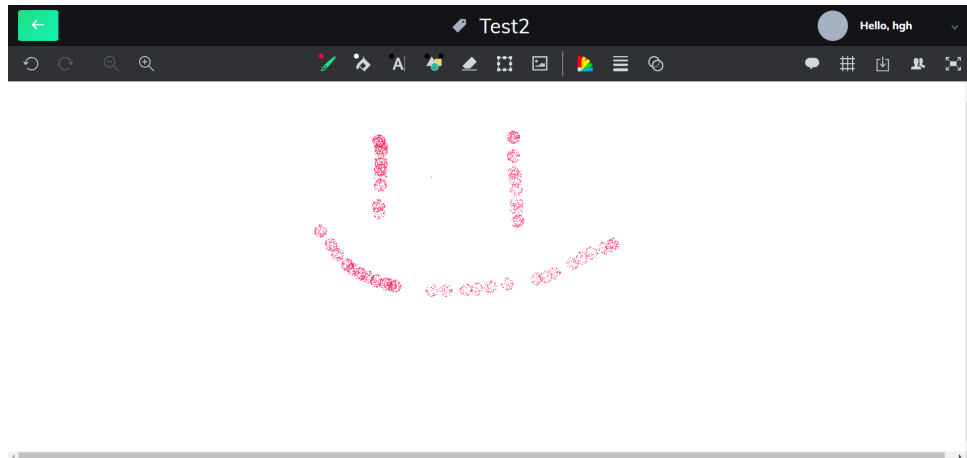


Abbildung 2.19: Graffiti



### 2.3.4 Background

Hiermit kann die Hintergrundfarbe der Skizze über das Farbauswahlmenü auf der rechten Seite gesetzt werden (siehe Abb. 2.20). Falls das Grid aktiv war, wird es deaktiviert, sobald man die Background Funktion nutzt.

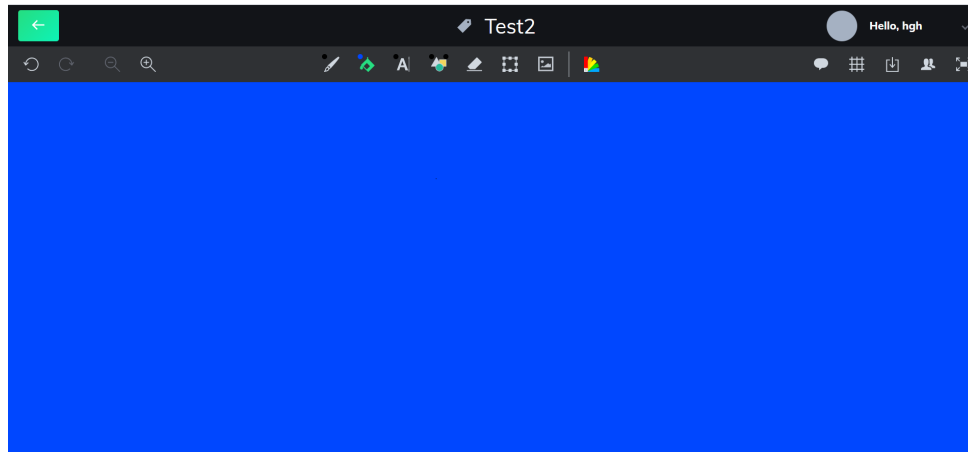


Abbildung 2.20: Hintergrundfarbe der Skizze

### 2.3.5 Grid

Das Raster kann auf der rechten Seite der Toolbar durch einen Mausklick (oder Touch) auf das Raster-Icon aktiviert oder deaktiviert werden. Durch das Einblenden des Grids bekommt der Nutzer einen einheitlichen Maßstab für die Skizze. Das Grid ist nur als Hilfsmittel verfügbar und somit nicht herunterladbar.

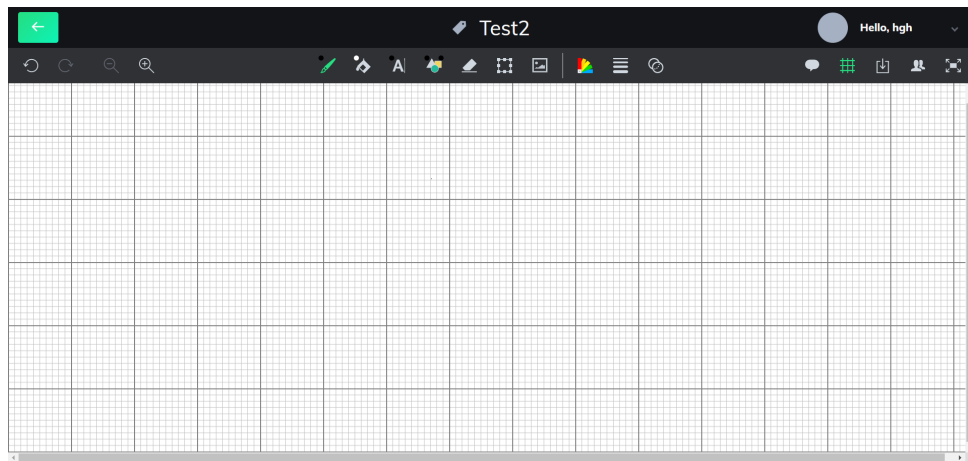


Abbildung 2.21: Grid

### 2.3.6 Text

Mit dem Textwerkzeug kann Text in die Skizze eingefügt werden. Dabei können Farbe (siehe Abb. 2.22), Schriftart (siehe Abb. 2.23), Schriftgröße (siehe Abb. 2.24) und Deckkraft (siehe Abb. 2.25) eingestellt werden. Nach Auswahl des Textwerkzeugs kann mit einem Linksklick in die Skizze eine Textbox erstellt werden. Hier kann nun Text mit der Tastatur eingegeben werden. Zum Fertigstellen des Textes (und Deaktivierung der Textbox) kann entweder die Escape-Taste gedrückt werden, ein Linksklick in die Skizze (außerhalb der Textbox) getätigt werden oder Strg+Enter gedrückt werden. Das Texttool ist danach weiter aktiv. Es kann mit Linksklick in die Skizze direkt ein weiterer Text erstellt werden. Ein vorhandener Text kann durch Linksklick wieder aktiviert werden (eine Textbox erscheint um den Text herum), so dass sowohl der Textinhalt als auch seine Attribute nachträglich geändert werden können.

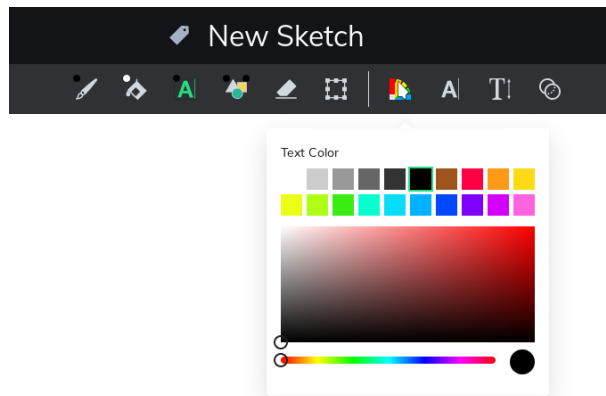


Abbildung 2.22: Farbe des Textes

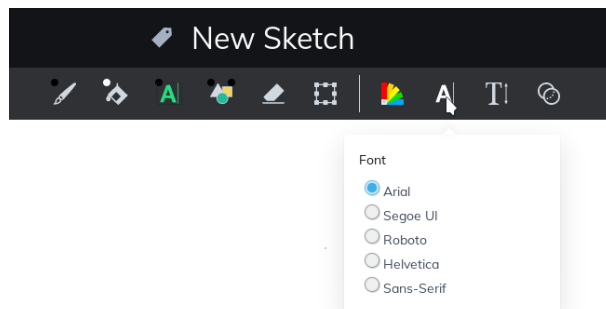


Abbildung 2.23: Schriftart des Textes

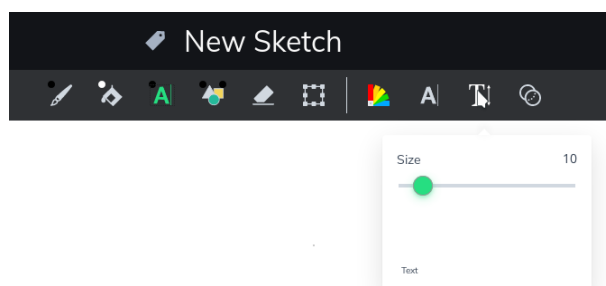


Abbildung 2.24: Schriftgröße des Textes

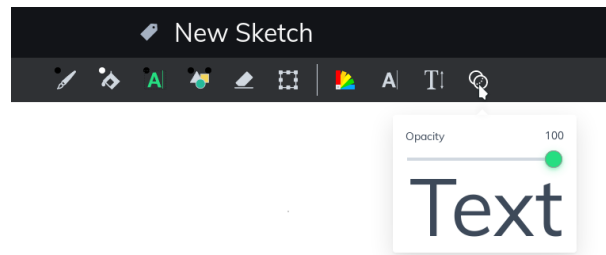


Abbildung 2.25: Deckkraft des Textes

### 2.3.7 Shapes

Hiermit können geometrische Formen eingefügt werden. Zur Auswahl stehen Rechtecke, Ellipsen, Kreise und Dreiecke. Es können die Füllfarbe, die Umrissfarbe, die Form, die Stärke des Umrisses sowie die Deckkraft eingestellt werden. Nach Aktivierung des Werkzeugs kann durch Linksklick und Gedrückthalten der linken Maustaste eine Form durch Ziehen in eine beliebige Richtung erstellt werden. Die Form wird gesetzt, indem die linke Maustaste losgelassen wird. Geometrische Formen können nur bewegt und in deren Größe verändert werden. Eine nachträgliche Farbänderung ist derzeit nicht möglich. Das Dreieck kann in der Formauswahl (Abbildung 2.26) unter *polygon* ausgewählt werden.

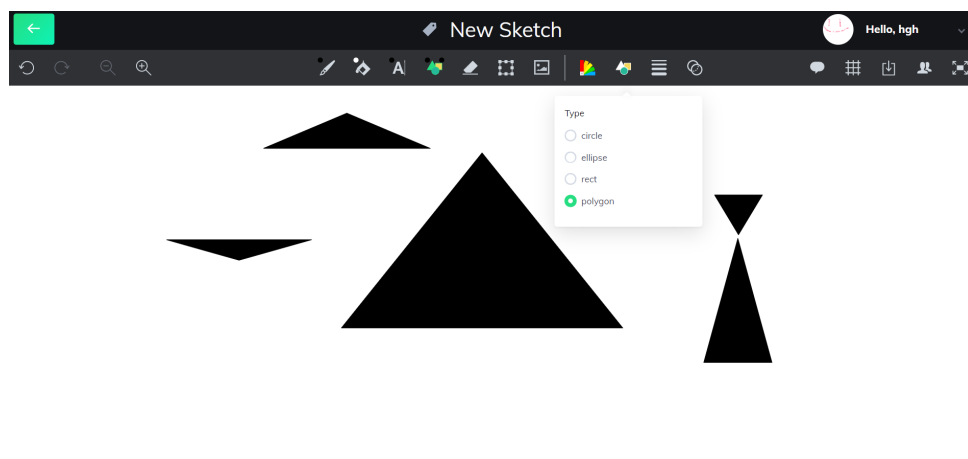


Abbildung 2.26: Auswahl an Formen. Beispiel Dreieck

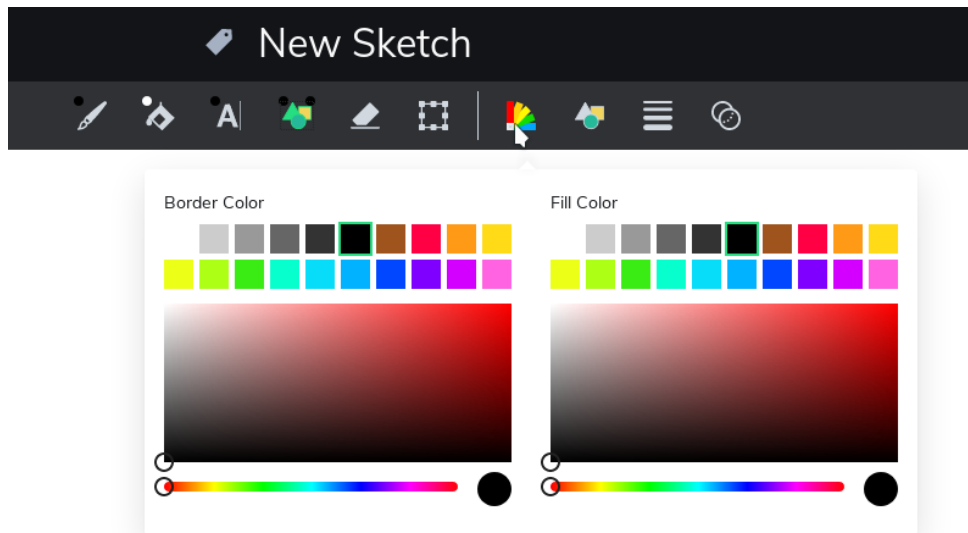


Abbildung 2.27: Füll- und Umrissfarbe der Formen

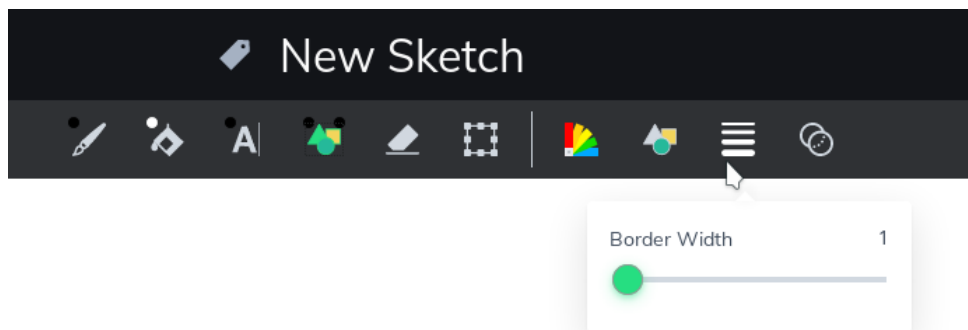


Abbildung 2.28: Umrissstärke der Formen

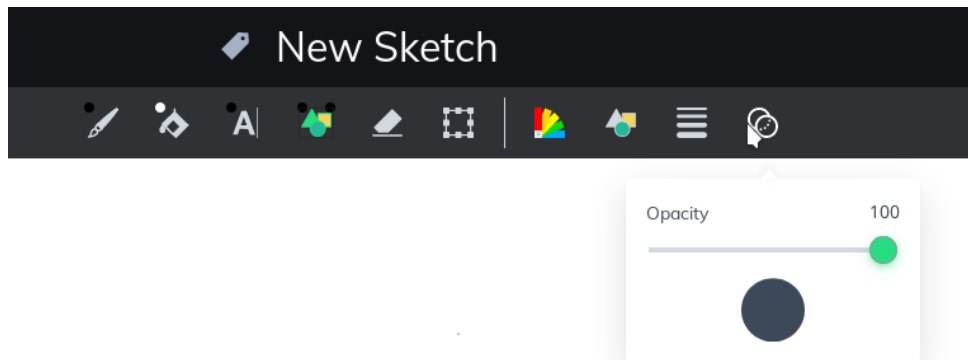


Abbildung 2.29: Deckkraft der Formen

### 2.3.8 Bilder einfügen

Mit dem Bildwerkzeug kann ein aus dem eigenen Dateimanager ausgewähltes Bild in die Skizze eingefügt werden. Es wird eine Preview der Bildgröße angezeigt, wenn mit der Maus über die Skizze gefahren wird. Die Größe des Bildes lässt sich mit dem Selector verändern.

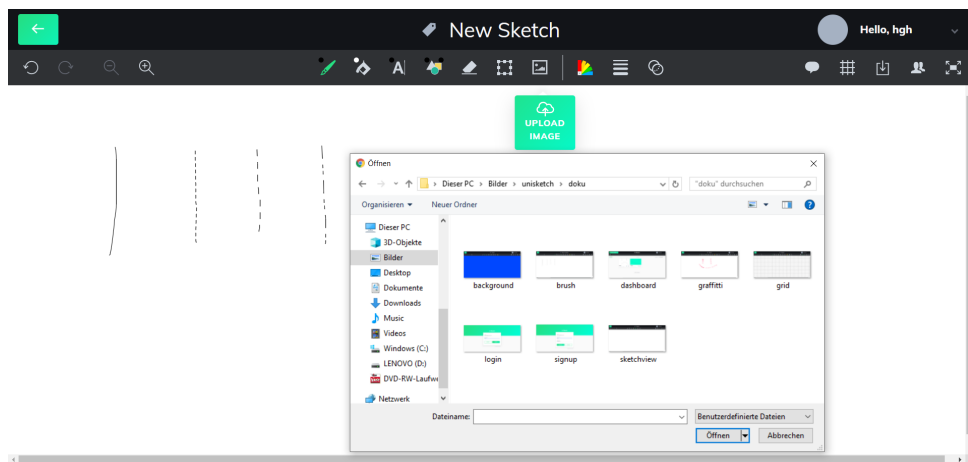


Abbildung 2.30: Upload eines Bildes

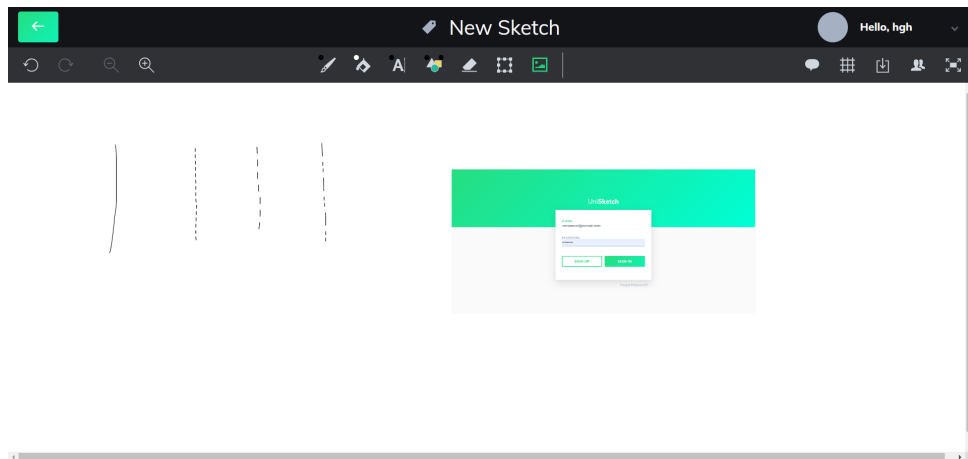


Abbildung 2.31: Eingefügetes Bild

### 2.3.9 Eraser

Mit dem Radierer können Elemente gelöscht werden (Linien, Texte und Formen). Sowohl durch Linksklick als auch durch Gedrückthalten und Ziehen der Maus können Elemente gelöscht werden. Dabei werden alle Elemente gelöscht, die sich unter dem Mauszeiger befinden.

### 2.3.10 Selector

Mit dem Selektierwerkzeug können Elemente selektiert werden. Dabei kann mit gehaltenem Linksklick und Ziehen der Maus in beliebige Richtung ein Auswahlrechteck aufgespannt werden. Nach Loslassen der linken Maustaste werden alle Elemente selektiert, die sich vollständig innerhalb des aufgezogenen Auswahlrechtecks befinden. Daraufhin wird die Größe des Auswahlrechtecks automatisch so angepasst, dass es die kleinstmögliche Größe hat, aber trotzdem noch alle selektierten Elemente umfasst. Alternativ können mit einfachem Linksklick alle Elemente selektiert werden, die sich unterhalb des Mauszeigers befinden.

Durch Gedrückthalten der Steuerungstaste lässt sich die Auswahl erweitern oder verringern. Sollten mit gedrückter Steuerungstaste ausschließlich Elemente ausgewählt werden, die bereits selektiert sind, so werden diese deselektiert. Sollte die Auswahl sowohl selektierte als auch nicht selektierte Elemente enthalten, so werden die nicht selektierten Elemente der Selektion hinzugefügt - die bereits selektierten Elemente bleiben Teil der Selektion.

Alternativ können durch Gedrückthalten der Umschalttaste, gehaltenen Linksklick und ziehen des Mauszeigers die selektierten Elemente verschoben werden.

Durch Drücken der Entfernentaste werden alle selektierten Elemente gelöscht.

Durch Drücken von Strg+v werden alle selektierten Elemente in eine Zwischenablage kopiert. Durch Drücken von Strg+v können diese dann eingefügt werden. Dies funktioniert auch zwischen verschiedenen Skizzen. Derzeit kann man die Elemente nur mit aktivem Selektierwerkzeug einfügen.

Durch die kleinen Quadrate an den Ecken und Kanten des Auswahlrechtecks können die selektierten Elemente skaliert werden.

### 2.3.11 Chat

Mit der Chatfunktion (siehe Abb. 2.32) können Textnachrichten versendet werden, die von allen Benutzern, die die gleiche Skizze geöffnet haben, gelesen werden können. Dadurch ist Kommunikation aller Kollaborateure innerhalb einer Skizze gewährleistet.

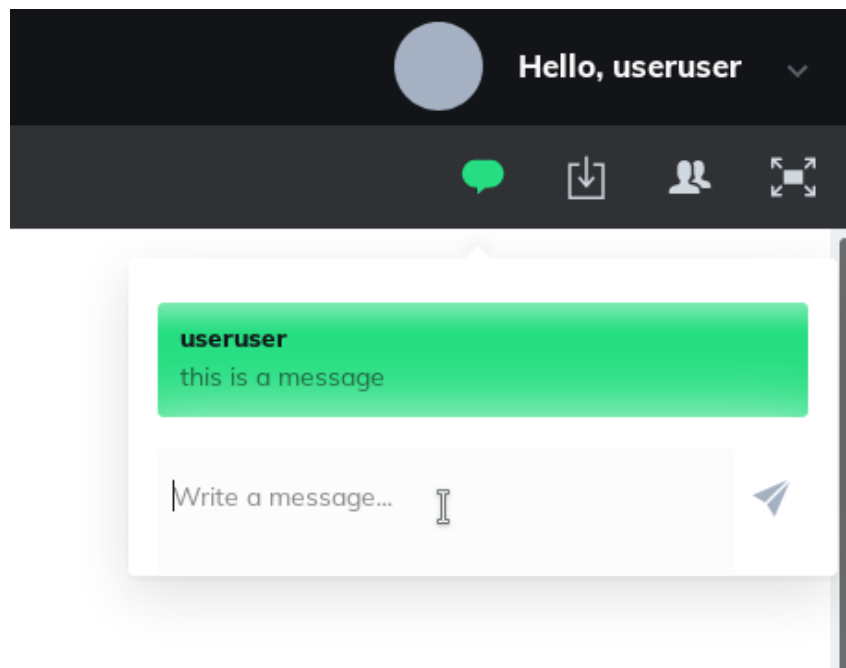


Abbildung 2.32: Chatfunktion

### 2.3.12 Download

Mit der Downloadfunktion kann die Skizze in 3 Formaten (PNG,PDF,SVG) heruntergeladen werden. Beim Anlicken des Download-Icons erscheint eine Auswahl der drei Formate. Standard-



mäßig ist SVG ausgewählt. Mit Bestätigung des Download-Buttons wird das ausgewählte Format herunter geladen.

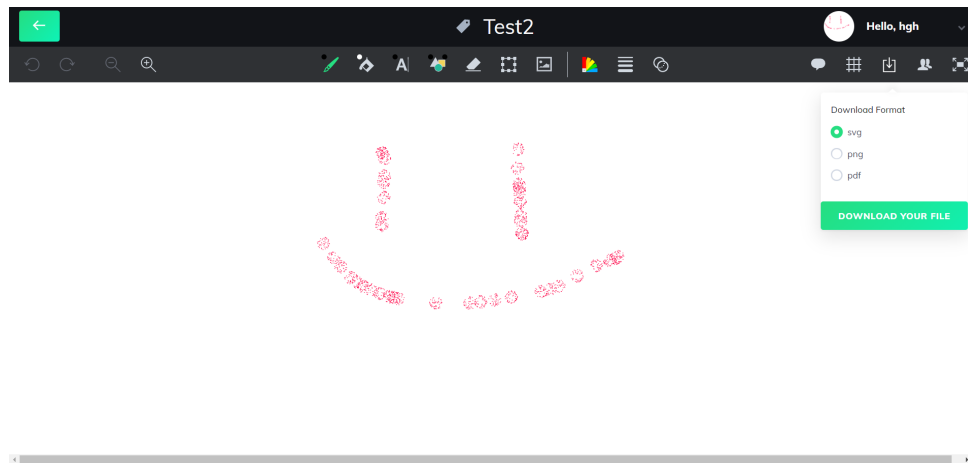


Abbildung 2.33: Download Optionen

### 2.3.13 Participants

Hiermit kann der Eigentümer der Skizze festlegen, wer an dieser Skizze teilhaben darf. Folgende Rechte können dabei vergeben werden:

- Owner - die angegebene Person wird Eigentümer der Skizze
- Editor - die angegebene Person darf in der Skizze mitzeichnen
- Viewer - die angegebene Person darf die Skizze anschauen.

Zum Verteilen von Rechten an andere Personen wird die Email-Adresse der Person, mit der sie sich bei UniSketch angemeldet hat, benötigt.

Es gibt desweiteren die Möglichkeit die Skizze öffentlich sichtbar zu machen. Dadurch kann jeder Benutzer die Skizze einsehen (jedoch nicht editieren) und sie erscheint auf der Startseite in der öffentlichen Galerie.

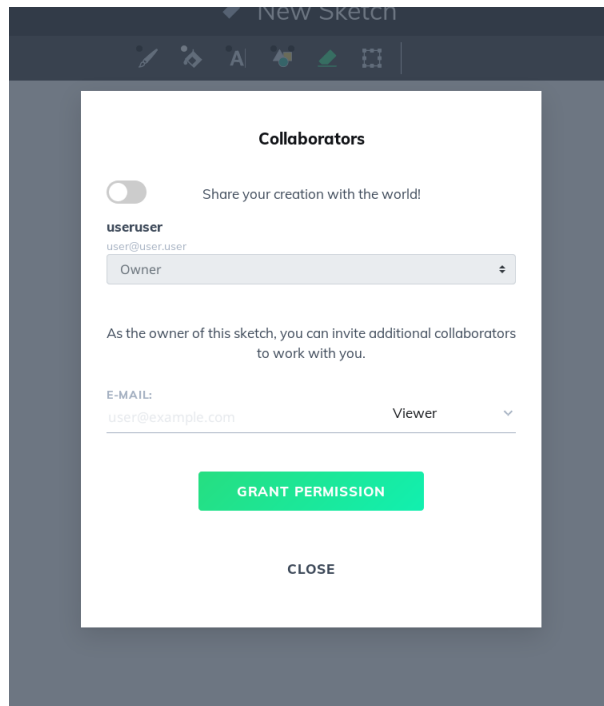


Abbildung 2.34: Rechte verteilen

### 2.3.14 Fullscreen

Hiermit lässt sich die gesamte Anwendung in den Vollbildmodus versetzen. UniSketch füllt dabei den gesamten Bildschirmbereich.

## 2.4 Usermenu

Der angemeldete Benutzer kann jederzeit durch Klick auf das Usermenu am rechten oberen Rand das Benutzermenü öffnen. Hier hat er die Möglichkeit seine *Library* einzusehen (zurück zum Dashboard), sein Profil einzusehen und zu bearbeiten sowie sich auszuloggen.

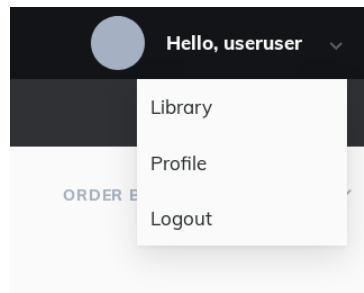


Abbildung 2.35: Benutzermenu

Auf der Profilseite kann der Benutzer seinen Namen, seine E-Mail Adresse und sein Passwort ändern. Hierzu muss zunächst das alte Passwort eingegeben werden. Die Felder für ein neues Passwort erscheinen erst nach Klick auf den Link *New Password*.

Abbildung 2.36: Benutzerprofil

Mithilfe des *Delete Profile*-Buttons unter dem *Save Changes*-Button kann der Account komplett gelöscht werden. Dies wird, wie das Löschen einer Skizze, mit einem Bestätigungsdialog abgesichert.

## 2.4.1 Profilbild

Das Profilbild kann in den Profileinstellungen geändert werden. Dazu müssen die Profileinstellungen geöffnet werden. Mit einem Klick auf das Profilbild wird den Benutzern eine Auswahl ihrer angefertigten Skizzen angezeigt. Nach dem sie mit einem Klick auf eine Skizze ihre Auswahl bestätigt haben, wird diese Skizze als ihr Profilbild angezeigt. Zu beachten ist hierbei, dass die Skizze vorher verlassen wurde, da Sie erst dann in der Datenbank gespeichert wird, wenn alle Nutzer diese verlassen haben.

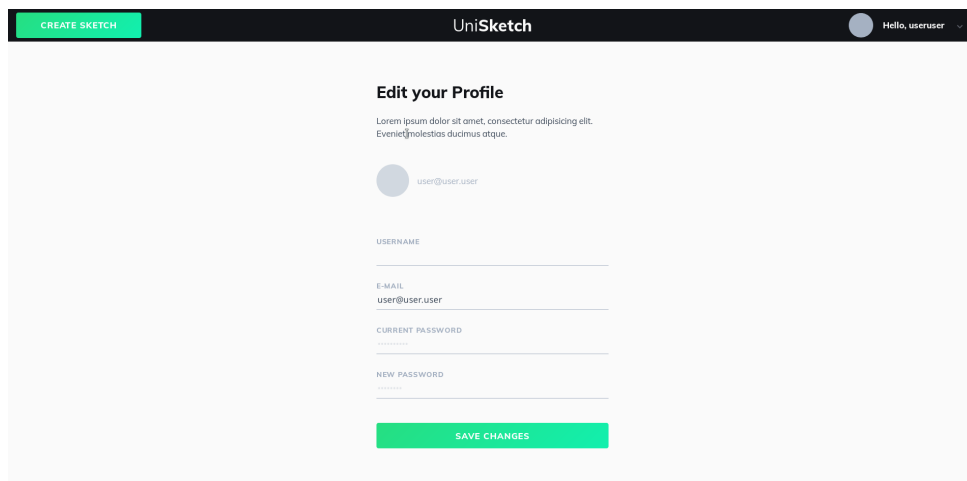
The screenshot shows the 'Edit your Profile' page in the UniSketch application. At the top, there is a dark header bar with a green 'CREATE SKETCH' button on the left, the 'UniSketch' logo in the center, and a user profile section on the right showing a grey circle and the text 'Hello, useruser'. The main content area is light grey and contains the title 'Edit your Profile' followed by a paragraph of placeholder text. Below this is a circular profile picture placeholder with the text 'user@user.user' next to it. Underneath are four input fields labeled 'USERNAME', 'E-MAIL', 'CURRENT PASSWORD', and 'NEW PASSWORD'. The 'E-MAIL' field contains the text 'user@user.user'. At the bottom of the form is a green 'SAVE CHANGES' button.

Abbildung 2.37: Auswahl eines Profilbildes

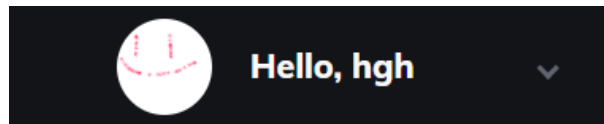


Abbildung 2.38: Geändertes Profilbild

## 2.5 Notifications

Ist der Server oder die Datenbank nicht erreichbar oder konnte eine Aktion in der Skizze nicht gespeichert werden, wird der Benutzer über eine Notification benachrichtigt. Hierbei werden bestimmte Statuscodes abgefangen und entsprechende Fehlermeldungen ausgelöst.

## 3 Entwicklung

### 3.1 Entwicklungsprozess

UniSketch läuft auf einem Projektserver der HTW Berlin[7] (siehe Kapitel 4). Diese Instanz entspricht dem fertigen Produkt, dass dem Benutzer bzw. Productowner angeboten wird. Daher wird auf dem Projektserver immer nur eine lauffähige Version von UniSketch deployed - der Projektserver wird niemals zum Testen verwendet! Damit UniSketch lokal auf einem Entwicklungsrechner entwickelt werden kann, ist etwas Einrichtung auf dem Rechner des Entwicklers notwendig, die im Folgenden beschrieben wird.

#### 3.1.1 Voraussetzungen

Die Entwicklung von UniSketch ist grundsätzlich plattformunabhängig, da es sich um eine Webanwendung handelt, die im Browser läuft. Daher ist die Entwicklung sowohl auf Windows[27] als auch auf Unix-artigen Betriebssystemen wie MacOS[10], Linux[9] oder BSD[1] möglich. Wichtig ist dabei, dass die Anwendung nicht in allen Browsern exakt identisch funktioniert und aussieht, da die Browser-Hersteller allesamt eigene Standards mit einbringen, die von anderen Browsern nicht oder unvollständig unterstützt werden.

Unter Firefox[24] ist die Benutzbarkeit derzeit voll unterstützt. Selbes gilt für Chrome[5] bzw. Chromium[23]. In Safari[21] kommt es an einigen Stellen zu Unregelmäßigkeiten beim Interpretieren von CSS, wodurch die Benutzbarkeit leicht eingeschränkt ist. Wir raten gänzlich von der Verwendung von Microsoft Edge[8] ab, da dieser nicht alle Funktionalitäten unterstützt. Grundsätzlich liegt das Hauptaugenmerk bei der Entwicklung auf der Lauffähigkeit mit Firefox - dies ist aber in jedem Fall mit dem Dozenten bzw. Productowner abzuklären. Eine fehlerfreie Kompatibilität mit allen Browsern ist aufgrund der geringen Entwickleranzahl und dem Fokus der Entwicklungsrichtung derzeit unrealistisch.

Die Entwickler von UniSketch7 haben sich beim Testen auf Firefox und Chrome/Chromium konzentriert.

Neben einem Browser zum Testen wird Node[15] samt NPM[16] benötigt, da sowohl Frontend als auch Backend auf einem Node-Server basieren.

Da UniSketch PostgreSQL[20] als Datenbank nutzt, muss dieses ebenfalls installiert werden. Dabei bitte Benutzernamen und Passwort des Postgres-Users notieren!

### 3.1.2 Klonen des Repositories und allgemeiner Aufbau

Die Account- und Zugangsdaten für das Git[4]-Repository befinden sich in Kapitel 4.1. Zuvor sollte jedoch Abschnitt 4.2 komplett durchgearbeitet werden! Daraufhin kann das neue Git-Repository geklont werden. Dabei ist im Repository sowohl das Frontend als auch das Backend enthalten. Das Frontend befindet sich im Unterordner *unisketch4*, das Backend im Unterordner *unisketch4-server*. Die Zahl 4 im Ordernamen hat historische Gründe: UniSketch4 war ein kompletter Rewrite der Anwendung. Unglücklicherweise hat sich die Versionszahl in den Namen geschlichen. Diese kann aber komplett ignoriert werden.

Das Repository beinhaltet zwei Branches<sup>1</sup>. Den master-Branch, sowie den masterwithzoom-Branch. Da die neue Zoomfunktion noch einige Bugs, welche unter 6.2 näher erläutert werden, beinhaltet, ist dieser im master-Branch nicht enthalten. Hier wird der alte Vier Stufen Zoom verwendet. Die Version mit kleineren Bugs findet sich im masterwithzoom-Branch wieder. Es kann also zum Start entschieden werden, mit welchem Stand fortgefahren wird.

### 3.1.3 Installation der Abhängigkeiten

Zunächst muss Angular installiert werden. Nachdem Node und NPM installiert wurden, kann mit folgendem Befehl Angular installiert werden:

```
npm install -g @angular/cli
```

Daraufhin müssen sämtliche Abhängigkeiten von Frontend und Backend installiert werden:

```
cd unisketch4 && npm i  
cd ../unisketch4-server && npm i
```

### 3.1.4 Erstellen der Datenbanken

Zum Laufen der Anwendung wird eine Datenbank benötigt. Genau genommen werden drei Datenbanken benötigt (mehr dazu in Kapitel 4). Da die Tabellen der Datenbanken automatisch durch das ORM[17] erstellt werden, reicht es, leere Datenbanken zu erstellen. Die Namen der Datenbanken sind dabei theoretisch frei wählbar. Jedoch empfiehlt es sich, der Einfachheit und Verständnis halber, die vorgegebenen Namen zu verwenden, damit diese identisch mit der Version auf dem Server sind.

---

<sup>1</sup>Paralleler Entwicklungsstrang zum gleichzeitigen Entwickeln versch. Features

Nachdem PostgreSQL installiert wurde, können mit folgenden Befehlen die benötigten Datenbanken erstellt werden. Hier wird beispielhaft davon ausgegangen, dass die Datenbanken für UniSketch8 erstellt werden sollen.

Achtung: das CLI-Interface von PostgreSQL (*psql*[19]) kann auf unterschiedlichen Betriebssystemen unterschiedlich funktionieren. Falls die folgenden Befehle nicht funktionieren, zieht bitte die offizielle PostgreSQL-Dokumentation[18] heran! Ggf. mag das vorherige Einrichten von Rollen nötig sein.

```
psql
# innerhalb von psql:
create database unisketch8;
create database unisketch8_development;
create database unisketch8_test;
```

### 3.1.5 Anpassen der Konfigurationsdateien

Nachdem die Datenbanken erstellt wurden, müssen die Konfigurationsdateien *config.json* und *database.json* im Backend angepasst werden. Vorlagen (\*.example) für diese Dateien befinden sich im Unterordner *unisketch4-server/src/config* und müssen entsprechend an Stelle kopiert und umbenannt werden.

Der Inhalt der *config.json* sähe für UniSketch8 beispielsweise so aus:

```
{
  "development": {
    "port": 8147,
    "mail_host": "smtp.sendgrid.net",
    "mail_user": "unisketch_htw",
    "mail_email": "unisketch@htw-berlin.de",
    "mail_password": "g03ZvpWhi6tPJYqZ",
    "mail_port": "587",
    "mail_proxy": "",
    "session_secret": "abc",
    "password_reset_token_expiration_time": 7200000,
    "base_path": "/unisketch8/"
  },
  "test": {
    "port": 8147,
    "mail_host": "smtp.sendgrid.com",
    "mail_user": "unisketch_htw",
    "mail_email": "unisketch@htw-berlin.de",
```

```

    "mail_password": "g03ZvpWhi6tPJYqZ",
    "mail_port": "587",
    "mail_proxy": "",
    "session_secret": "abc",
    "password_reset_token_expiration_time": 7200000,
    "base_path": "/unisketch8/"
  },
  "production": {
    "port": 8147,
    "mail_host": "smtp.sendgrid.com",
    "mail_user": "unisketch_htw",
    "mail_email": "unisketch@htw-berlin.de",
    "mail_password": "g03ZvpWhi6tPJYqZ",
    "mail_port": "587",
    "mail_proxy": "",
    "session_secret": "abc",
    "password_reset_token_expiration_time": 7200000,
    "base_path": "/unisketch8/"
  }
}

```

Mit dem angegebenen Email-Account werden Anfragen für das Zurücksetzen des Passworts beantwortet. Die restlichen Einträge sollten selbsterklärend sein.

Der Inhalt der *database.json* sähe für UniSketch8 beispielsweise wie folgt aus. Wichtig ist hierbei, dass die Namen der zuvor erstellten Datenbanken eingetragen werden. Username und Password sollten schon vorher beim Erstellen der Datenbanken angelegt worden sein bzw. sollten die des Postgres-Users sein.

```

{
  "development": {
    "type": "postgres",
    "host": "127.0.0.1",
    "port": 5432,
    "username": "",
    "password": "",
    "database": "unisketch8_development",
    "synchronize": true,
    "logging": true,
    "entities": [
      "src/models/**/*.ts"
    ]
  },

```



```

    "migrations": [
      "src/migration/**/*.ts"
    ],
    "subscribers": [
      "src/subscriber/**/*.ts"
    ]
  },
  "test": {
    "type": "postgres",
    "host": "127.0.0.1",
    "port": 5432,
    "username": "",
    "password": "",
    "database": "unisketch8_test",
    "synchronize": true,
    "logging": false,
    "entities": [
      "src/models/**/*.ts"
    ],
    "migrations": [
      "src/migration/**/*.ts"
    ],
    "subscribers": [
      "src/subscriber/**/*.ts"
    ]
  },
  "production": {
    "type": "postgres",
    "host": "127.0.0.1",
    "port": 5432,
    "username": "",
    "password": "",
    "database": "unisketch8",
    "synchronize": false,
    "logging": false,
    "entities": [
      "src/models/**/*.ts"
    ],
    "migrations": [
      "src/migration/**/*.ts"
    ],
  },

```

```

    "subscribers": [
      "src/subscriber/**/*.ts"
    ]
  }
}

```

Der Einfachheit halber enthalten die jeweiligen Dateien schon diesen Inhalt, sodass sie nurnoch umbenannt werden müssen.

### 3.1.6 Testen der Anwendung

Nun können Frontend und Backend gestartet werden! Im Root-Verzeichnis des Repositories befinden sich die Skripte *run\_backend\_dev.sh/bat* und *run\_frontend\_dev.sh/bat*. Diese müssen ausgeführt werden. Es empfiehlt sich zunächst das Backend zu starten. Daraufhin kann im Browser unter der Adresse *http://127.0.0.1:4200* UniSketch ausgeführt werden.

## 3.2 Backend

Das Backend von UniSketch basiert auf einem Node[15]-Server und als Programmiersprache wird TypeScript[26] eingesetzt. Eine PostgreSQL[20]-Datenbank persistiert sämtliche Nutzer- und Skizzendaten und TypeORM[25] ist für das Mapping <sup>2</sup> der Objekte in die Datenbank zuständig. Grundsätzlich ist das Backend in Model-Klassen für die Datenbankanbindung, Controller-Klassen für die REST-Endpoints, und dem Websocket-Handler für die Echtzeit-Skizzenfunktionen aufgeteilt.

### 3.2.1 REST-Endpoints

Die REST-Endpoints sind mithilfe von Express[3] realisiert. Die dazugehörigen Routen sind in der Datei *routes.ts* konfiguriert und kategorisch aufgeteilt. Jedem Bereich ist ein Controller zugewiesen - der UsersController ist bspw. für Routen beginnend mit */user/* zuständig. Die meisten Routen nutzen die *RequireAuth*-Middleware, die anhand der Sitzung des Benutzers den Zugriff auf eingeloggte Benutzer beschränkt. Genauere Informationen sind in der Code-Dokumentation zu finden.

---

<sup>2</sup>Aufeinanderabbilden von Datenelementen aus zwei verschiedenen Datenmodellen [12]

### 3.2.2 Websockets

Für die Websocket-Implementation wird Socket.IO[22] verwendet. Nachrichten bestehen aus einem Namen und einem JSON-Body. Jeder aktiven Skizzensitzung wird ein Socket.IO-Raum zugewiesen. Die WebSockets werden lediglich für Echtzeitdaten (Zeichnen von Linien, Synchronisation der Skizze mit dem Server) verwendet - verwalterische Aktionen wie z.B. das Umbenennen einer Skizze laufen über die REST-API.

### 3.2.3 Datenbank

Die Anbindung an die PostgreSQL-Datenbank läuft über TypeORM. Die zugehörigen Models sind im *models*-Verzeichnis definiert. Anhand dessen können Daten aus der Datenbank abgefragt bzw. aktualisiert werden, ohne sich manuell um Verbindungen oder SQL-Queries zu kümmern. Zudem werden alle benötigten Tabellen automatisch erstellt.

Vor UniSketch6 wurde Sequelize[11] als ORM eingesetzt. Mit UniSketch6 wurden neben Linien weitere Skizzenelemente eingefügt. Um den Code leichter zu pflegen und besser erweiterbar zu gestalten, wurde Vererbung eingeführt. Da Sequelize jedoch keine Vererbung beherrscht, wurde TypeORM eingeführt.

### 3.2.4 Skizzen

Die Implementation für die Echtzeit-Skizzenverarbeitung ist in drei Klassen aufgeteilt:

- WebSocketHandler nimmt Verbindungen entgegen und liest eingehende Nachrichten
- SketchSessionManager hält alle aktiven Skizzen-Sitzungen im Speicher
- SketchSession hält Informationen zu einer einzelnen aktiven Skizzen-Sitzung

Die Kontrolle darüber, welcher Benutzer sich in welcher Skizze befindet und welche Nachrichten erhalten soll, geschieht mittels Socket.IO-Räumen.

Tritt ein Benutzer einer Skizzen-Sitzung bei, die noch nicht existiert, wird die Skizze aus der Datenbank geladen und die neue SketchSession im SketchSessionManager hinterlegt. Die ID der Skizzenelemente innerhalb der Datenbank ist hier unerheblich - jedes Skizzenelement erhält eine (je Skizze eindeutige) temporäre ID zur Laufzeit. So können später neue Skizzenelemente hinzugefügt werden, ohne von der ID-Vergabe der Datenbank abhängig zu sein.

Zu Beginn erhält der Benutzer ein Paket mit den grundlegenden Daten der Skizze und daraufhin

mehrere Pakete mit Skizzenelementen. Diese *sketch\_part*-Pakete enthalten jeweils maximal 100 Skizzenelemente und werden im Intervall von 50 Millisekunden nach und nach gesendet, bis die gesamte Skizze am Client angekommen ist. Dies ist notwendig, da es bei sehr großen Skizzen sonst zu Problemen und langen Ladezeiten bei Browsern kommt.

Erst wenn alle Benutzer eine Skizze wieder verlassen haben, wird sie auch in der Datenbank aktualisiert. Hierzu werden innerhalb einer Transaktion alle vorhandenen Skizzenelemente einer Skizze gelöscht und daraufhin die neuen Linien als neue Datensätze abgelegt.

### 3.2.5 Chat

Der Chat ist mit Hilfe von Websockets realisiert - sobald eine Skizze aktiv genutzt wird, wird im Hintergrund eine WebSocket Session erstellt, wodurch alle Teilnehmer der Skizze verwaltet werden können. Hier werden auch alle nötigen Informationen für den Chat zwischen den Teilnehmern einer Skizze gespeichert.

Sobald ein User eine Nachricht abgeschickt hat, wird diese per WebSocket-*emit* an alle anderen aktiven Benutzer der Skizze geschickt und dort im Frontend aktualisiert angezeigt. Aufgrund von Zeichensatzkompatibilität ist die Nachricht auf dem Transportweg mit base64 codiert, damit alle Sonderzeichen und Umlaute o.ä. verarbeitet werden können.

## 3.3 Frontend

Das Frontend von UniSketch wurde mit Angular umgesetzt. Für den Einstieg empfehlen wir, das *Tour of Heroes*-Tutorial durchzugehen.

### 3.3.1 Komponenten

Das Frontend ist in mehrere Komponenten aufgeteilt - einige davon entsprechen einem kompletten View, andere sind wiederverwendbare Einzelemente. Die hauptsächlichen Komponenten sind in Tabelle 3.3.1 gelistet. Eine komplette Auflistung ist der Code-Dokumentation zu entnehmen.

### 3.3.2 Services

Darüber hinaus stehen den Komponenten verschiedene Services zur Verfügung, die die Kommunikation mit dem Backend ermöglichen. Die Tabelle 3.3.2 listet die Services auf.

Name	Beschreibung
LoginComponent	Startseite der Anwendung. Enthält das Login-, Registrierungsformular und den About View als Unterkomponenten.
DashboardComponent	Elternkomponente der Bibliothek und des Profils. Enthält die Menüleiste.
HomeComponent	Bibliothek. Enthält eine sketchcard Komponente für jede dem Benutzer sichtbare Skizze.
ProfileComponent	Profil. Enthält das Formular zum Ändern von Profildaten.
SketchViewComponent	Skizzenview. Enthält den Zeichencanvas und die sketch-toolbar, die eine Auswahl von Zeichentools bereitstellt.
TagComponent	Stellt Funktionen für die Tags bereit.
ChatOverlayComponent	Stellt Funktionen für den Chat bereit.
NotificationComponent	Stellt Funktionen für den Chat bereit.

Tabelle 3.1: Hauptkomponenten des Frontends

### 3.3.3 Skizzenview

Der komplexeste Teil des Frontends ist die Skizzenview. Er besteht aus den *sketch-view* und *sketch-toolbar* Komponenten, sowie dem SketchService und nutzt zusätzlich Klassen aus dem *sketch*-Unterordner. Darin befinden sich die verfügbaren Skizzierwerkzeuge (momentan Pinsel, Radierer, Text, Formen, Selektion, Image), sowie die Events-Directive für den Skizzencanvas. Der Code hierzu ist vollständig dokumentiert und mit Kommentaren versehen, für ein komplettes Verständnis wird empfohlen, diesen einzusehen. Die Events-Directive für das Skizzencanvas nimmt Events vom Browser entgegen (Mouse bzw. Touch) und leitet sie an das jeweils aktive Tool weiter. Außerdem wird das Canvas dort auch mit dem SketchService verbunden, sodass der Client entsprechende Nachrichten vom Server behandeln kann.

Es lassen sich unterschiedliche Skizzenelemente erstellen (siehe Kapitel 2). Skizzenelemente werden clientseitig bereits auf dem Canvas gerendert, auch wenn der Server diese noch nicht erhalten hat. Dadurch gibt es beim Zeichnen keine Latenz, die der Usability schaden könnte. Es besteht die Möglichkeit, Linienvertices in einem Batch zu sammeln und nur in bestimmten Zeitabständen zu senden, davon wird momentan allerdings noch nicht Gebrauch gemacht. Mehr Informationen dazu sind im Websocket-Protokoll zu finden (Stichwort: *batch\_interval*).

Service	Beschreibung
auth-guard	Schützt vor unerlaubten Zugriff auf Skizzen
authentication	Authentifizierung (Login, Logout, User-Registrierung)
dashboard	Alle Funktionalitäten innerhalb des Dashboards
folder	Ermöglicht das Organisieren von Skizzen in Ordnern
notification	Benachrichtigungen an den Benutzer senden (Server nicht erreichbar, Skizze gespeichert, etc.)
participants	Managed den Zugriff weiterer Personen auf eine Skizze
public-sketches	Verwaltung von öffentlichen Skizzen
sketch	Verwaltet sämtliche Funktionalitäten der Skizze (Zeichenwerkzeuge, Undo/Redo, Chat, etc.)
window-ref	Fensterzugriff

Tabelle 3.2: Services des Frontends

### 3.3.4 Responsiveness

Um das Frontend an verschiedene Endgeräte und Bildschirmgrößen anzupassen, nutzt UniSketch Bootstrap[2] und vereinzelt auch pures CSS. Insbesondere wird das Bootstrap-Grid-System[6] verwendet, dass es ermöglicht Seitenelemente anhand eines skalierenden Grids darzustellen. Elemente sind dabei Teil eines Containerelements (`class="container"`), das selbst nochmal in Zeilen getrennt (`class="row"`) ist und zu allerletzt je Zeile in selbst-skalierende Spalten, die sich anhand der gewählten CSS Klasse anders verhalten.

Die Unterschiede zwischen den verschiedenen Optionen und weitere Einstellungsmöglichkeiten können der Bootstrap Dokumentation entnommen werden. Am besten sichtbar ist das Ergebnis in der Bibliothek, die die einzelnen Skizzen je nach Fenstergröße skaliert bzw. mehr oder weniger Elemente pro Zeile darstellt.

## 3.4 REST-Endpoints

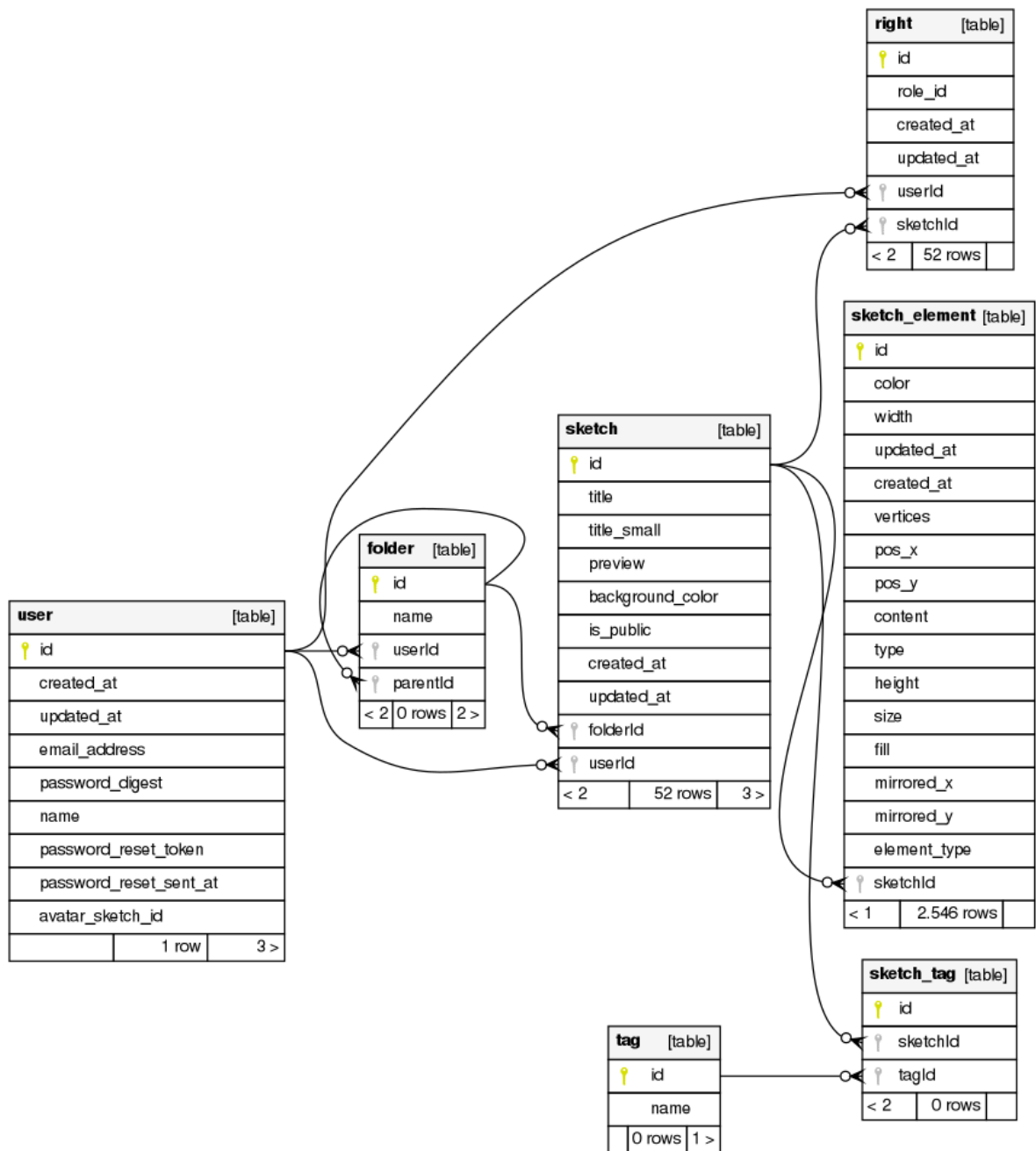
Für die Kommunikation mit dem Server betreffend aller Services außer dem SketchService werden REST-Endpoints genutzt. Sämtliche Endpoints sind in der Datei *unisketch4-server/src/routes.ts* definiert. Die Namensgebung zusammen mit den entsprechenden HTTP-Methoden (POST, GET, DELETE) beschreibt die Funktion der einzelnen Endpoints ausreichend. Daher wird an dieser Stelle auf eine Auflistung aller Endpoints verzichtet und an die oben genannte *routes.ts* verwiesen.

## 3.5 WebSocket-Messages

Für die Echtzeit-Kommunikation mit dem Server für Skizzenview und Chat werden Websockets genutzt. Dabei gibt es für sämtliche Funktionalitäten einzelne Messages. Im SketchService im Frontend (*unisketch4/src/app/services/sketch.service.ts*) werden alle Messages definiert. Der Server antwortet auf diese Messages ebenfalls mit WebSocket-Messages. Diese sind im Backend in der SketchSession (*unisketch4-server/src/sketch/sketch\_session.ts*) definiert. Da die Messages umfangreich und zugleich durch Code beschrieben sind, wird an dieser Stelle auf eine Auflistung aller Messages verzichtet.

Grundsätzlich läuft die WebSocket-Kommunikation nach folgendem Schema ab:

Durch Nutzung eines der Skizzierwerkzeuge oder des Chats wird über den SketchService (*unisketch4/src/app/services/sketch.service.ts*) eine WebSocket-Message Richtung Server gesendet. Dieser nimmt sämtliche WebSocket-Messages im WebSocket-Handler (*unisketch4-server/src/sketch/ws\_handler.ts*) entgegen. Zur eigentlichen Verarbeitung werden die empfangenen Daten dann an die zugehörige serverseitige SketchSession (*unisketch4-server/src/sketch/sketch\_session.ts*) weitergeleitet. Nach Verarbeitung der Anfragedaten werden die Antwortdaten wieder zurück an den SketchService im Frontend gesendet. Von dort aus werden die Daten an das Skizzierwerkzeug weitergeleitet, das die Anfrage gestartet hat.



Generated by SchemaSpy

Abbildung 3.1: Das Datenbankschema



## 4 Deployment

In UniSketch7 war der Webservice der VR-Cave nicht erreichbar. Das System wurde auf den Server geladen, jedoch konnte nicht überprüft werden, ob dies korrekt geschehen ist. Das folgende Kapitel sind die Anweisungen zum Deploy des Systems der Vorgängerguppe (UniSketch6). Angepasst wurde nur die Version.

Derzeit ist keine Version von Unisketch erreichbar, da der Server der Cave offline ist. Stand(10.4.2020). Seit der Version 4 hat sich etabliert, die Vorgängerversion unangetastet zu lassen und die neue Version unter einer neuen Subdomain erreichbar zu machen. Nachfolgend sind diese Domains gelistet.

- UniSketch4: <http://vr-cave.f4.htw-berlin.de/>
- UniSketch4: <http://vr-cave.f4.htw-berlin.de/unisketch4>
- UniSketch5: <http://vr-cave.f4.htw-berlin.de/unisketch5>
- UniSketch6: <http://vr-cave.f4.htw-berlin.de/unisketch6>
- UniSketch7 <http://vr-cave.f4.htw-berlin.de/unisketch7>

Dadurch wird nachfolgenden Projektgruppen die Möglichkeit gegeben, sich anzuschauen, was ihre Vorgänger erarbeitet haben - als eine Art Visualisierung der Entwicklung über viele Semester hinaus.

Um diese Tradition fortzusetzen, wird zu Beginn etwas Einrichtungs- und Installationsarbeit fällig. Diese wird in Kapitel 4.2 detailliert beschrieben und muss zu Beginn einmalig durchgeführt werden.

Im Kapitel 4.3 wird dann beschrieben, wie ein neuer Entwicklungsstand deployed wird.

### 4.1 Konten und Passwörter

Im Folgenden werden alle Zugänge gelistet, die im Rahmen des Projekts relevant sind. Auf diese wird in den Kapiteln 4.2 und 4.3 zugegriffen.

Anmerkung: nicht alle Konten werden zwingend benötigt, sie werden hier nur der Vollständigkeit halber gelistet.

#### Github-Repository für UniSketch7

Adresse `https://github.com/UniSketch/UniSketch`  
Benutzer `UniSketch`  
Passwort `HtwBerlin2020`

#### Github-Repository für UniSketchDoku

Adresse `https://github.com/UniSketch/UniSketchDoku`  
Benutzer `UniSketch`  
Passwort `HtwBerlin2020__`

#### Default-User für Projektserver

Adresse `http://vr-cave.f4.htw-berlin.de`  
Benutzer `sketchpad3`  
Passwort `GNA1rnw__`

#### User mit Sudo-Berechtigung für Projektserver

Adresse `http://vr-cave.f4.htw-berlin.de`  
Benutzer `diamond`  
Passwort `GNA1rnw__`

#### Root-User für Projektserver

Adresse `http://vr-cave.f4.htw-berlin.de`  
Benutzer `root`  
Passwort `kl4amwx5b`

#### Postgres-User für Projektserver

Adresse `http://vr-cave.f4.htw-berlin.de`  
Benutzer `postgres`  
Passwort `kl4amwx5b`

#### Email-Account für Notifications des Anwendungsservers

Host `smtp.sendgrid.net`  
User `unisketch_htw`  
Email `unisketch@htw-berlin.de`  
Passwort `g03ZvpWhi6tPJYqZ`

## 4.2 Installation

### 4.2.1 Kopie der Repositories

Im ersten Schritt soll jeweils eine Kopie von den beiden Git-Repositories (siehe 4.1) erstellt werden, die den Quellcode und die Latex-Dokumentation enthalten. Dazu kann der Dummy-Account *UniSketch* verwendet werden - dieser hat Zugriff auf beide Repositories. Auf den neu erstellten Git-Repositories wird dann die neue Projektgruppe weiterentwickeln - die ursprünglichen Git-Repositories sollen unverändert bleiben! Den neuen Repositories müssen als Contributor neben allen Entwicklern auch der Benutzer *UniSketch* hinzugefügt werden.

Anmerkung: die Verwendung und Weiterentwicklung der Latex-Dokumentation ist nicht bindend, wurde aber von der Projektgruppe 6 angefangen und wird der Einfachheit halber empfohlen.

### 4.2.2 Subdomain für neue UniSketch-Version anlegen

Auf dem Projektserver läuft ein NGINX[14]-Server. Dieser wird von UniSketch für Reverse-Proxies benutzt, so dass jede Version unter ihrer eigenen Subdomain abgefragt werden kann (vgl. Kapitel 4). Folgende Schritte sind nötig, um eine neue Version von UniSketch unter einer neuen Subdomain erreichbar zu machen.

1. Als User *root* in den Projektserver einloggen und die Konfigurationsdatei von NGINX öffnen:

```
ssh sketchpad3@vr-cave.f4.htw-berlin.de
su root
vim /opt/nginx/conf/nginx.conf
```

2. Nun in die Datei einen neuen Eintrag für die neue UniSketch-Version einfügen. Für UniSketch8 sähe dieser beispielsweise so aus:

```
location /unisketch8/ {
    proxy_pass http://0.0.0.0:8084;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}
```

Entscheidend ist dabei, dass eine neue Location angegeben wird, die der neuen Subdomain entspricht. Dieser Subdomain muss dann ein freier Port zugewiesen werden.

3. Nun den NGINX-Server als User *sketchpad3* neustarten:

```
exit
cd ~
./restart_nginx.sh
```

### 4.2.3 Neue Datenbank erstellen

Jede Version von UniSketch hat ihre eigene PostgreSQL[20]-Datenbank auf dem Projektserver. Da sich die Struktur der Datenbank während der Entwicklung zwangsläufig verändert, ist dies auch unbedingt notwendig. Sollte eine neue Version von UniSketch die Datenbank einer vorhergehenden Version ändern, dann wird die vorhergehende Version mit hoher Wahrscheinlichkeit nicht mehr mit der neuen Datenstruktur umgehen können. Darum muss eine neue Datenbank erstellt werden. Anmerkung: derzeit gibt es für jede Version von UniSketch drei Datenbanken: *development*, *testing*, *production*. Bis dato (UniSketch7) wird jedoch nur die Datenbank *development* genutzt. Hier besteht für eine zukünftige Projektgruppe die Möglichkeit, die Nutzung der einzelnen Datenbanken den entsprechenden Szenarien zuzuordnen, was unter anderem auch das Einrichten von automatisierten Tests nach sich ziehen würde/könnte. Dies empfehlen wir dringend um den Grund für einige Bugs gezielt finden zu können.

1. Als User *sketchpad3* auf dem Projektserver einloggen und *psql* ausführen:

```
ssh sketchpad3@vr-cave.f4.htw-berlin.de
psql
```

2. Innerhalb von *psql* können nun neue Datenbanken erstellt werden oder vorhandene Datenbanken inspiziert werden. Der Befehl `\list` listet beispielsweise alle vorhandenen Datenbanken auf. Wir möchten drei neue Datenbanken anlegen. Dafür ist es empfehlenswert, die Datenbanken der vorhergehenden Version als Vorlage zu nutzen. Im Fall von UniSketch8 sähe das zum Beispiel so aus:

```
CREATE DATABASE unisketch8 TEMPLATE unisketch7 OWNER sketchpad3;
CREATE DATABASE unisketch8_test TEMPLATE unisketch7_test OWNER sketchpad3;
CREATE DATABASE unisketch8_development TEMPLATE unisketch7_development OWNER
↪ sketchpad3;
```

3. Damit die Datenbanken auch für unsere Anwendung erreichbar sind, muss als User *root* in der *pg\_hba.conf* ein Eintrag pro Datenbank hinzugefügt werden.

```
su root
vim /etc/postgresql/9.4/main/pg_hba.conf
```

Für UniSketch8 sähe das beispielsweise so aus:

```
host unisketch8          sketchpad3 0.0.0.0/0 md5
host unisketch8_development sketchpad3 0.0.0.0/0 md5
host unisketch8_tests    sketchpad3 0.0.0.0/0 md5
```

4. Nun muss der PostgreSQL-Server noch neugestartet werden(ebenfalls als *root*):

```
/etc/init.d/postgresql restart
```

#### 4.2.4 Skripte anpassen

Für das Deployen der Anwendung sowie zum Neustarten des Anwendungsservers existieren Skripte. Diese sind nicht Teil des Git-Repositories, sondern befinden sich auf dem Projektserver im Home-Verzeichnis des Users *sketchpad3*. Von diesen Skripten muss eine Kopie erstellt werden und diese müssen dann an die neue UniSketch-Version angepasst werden.

1. Als User *sketchpad3* auf dem Projektserver einloggen und die vorhandenen Skripte der letzten UniSketch-Version kopieren. Für UniSketch8 sähe das beispielsweise so aus:

```
ssh sketchpad3@vr-cave.f4.htw-berlin.de
cd ~
cp -r unisketch7 unisketch8
```

2. Nun muss das Skript *deploy.sh* angepasst werden:

```
cd unisketch8
vim deploy.sh
```

In diesem Skript werden zu Beginn Variablen gesetzt. Diese müssen umgeändert werden. Für UniSketch8 sähe das beispielsweise so aus:

```

REPO_PATH=/var/www/unisketch8/repo
APP_PATH=$REPO_PATH
RELEASES_PATH=/var/www/unisketch8/releases
SHARED_PATH=/var/www/unisketch8/shared
CURRENT_RELEASE_PATH=/var/www/unisketch8/current
ANGULAR_BASE_HREF=/unisketch8/
BACKUP_COUNT=2

```

3. Nun muss das Skript *restart\_server.sh* angepasst werden.

```
vim restart_server.sh
```

Für UniSketch8 sähe das beispielsweise so aus:

```

{
    screen -X -L -S "unisketch8" quit || true
} &> /dev/null

screen -dmLS "unisketch8" node $CURRENT_RELEASE/unisketch4-server/build/

echo OK

```

4. Nun muss das Skript *cache\_node\_modules.sh* angepasst werden.

```
vim cache_node_modules.sh
```

In diesem Skript werden zu Beginn Variablen gesetzt. Diese müssen umgeändert werden. Für UniSketch8 sähe das beispielsweise so aus:

```

CURRENT_RELEASE=/var/www/unisketch8/current
SHARED=/var/www/unisketch8/shared

```

#### 4.2.5 Build-Ordner der Anwendung einrichten

Die Builds, auf die die Anwendungsserver im Front- und Backend letztendlich zugreifen, befinden sich auf dem Projektserver im Verzeichnis */var/www/*. Dort muss nun auch die neue UniSketch-Version eingerichtet werden.

1. Als User *sketchpad3* auf dem Projektserver einloggen und den Ordner der letzten UniSketch-Version kopieren. Für UniSketch8 sähe das beispielsweise so aus:

```
ssh sketchpad3@vr-cave.f4.htw-berlin.de
cd /var/www/
cp -r unisketch7 unisketch8
```

2. Nun müssen zunächst die alten Builds und der lokale Klon des alten Git-Repositories gelöscht werden:

```
cd unisketch8
rm -rf releases/*
rm -rf repo
```

3. Nun muss das Git-Repository der neuen UniSketch-Version geklont und umbenannt werden:

```
git clone https://mygitserver.com/user/UniSketch8.git
mv UniSketch8 repo
```

4. Nun muss die Konfigurationsdatei *config.json* angepasst werden, die sich in dem Unterordner *shared/config/* befindet.

```
cd shared/config
vim config.json
```

Für UniSketch8 sähe die *config.json* beispielsweise so aus:

```
{
  "development": {
    "port": 8084,
    "mail_host": "smtp.sendgrid.net",
    "mail_user": "unisketch_htw",
    "mail_email": "unisketch@htw-berlin.de",
    "mail_password": "g03ZvpWhi6tPJYqZ",
    "mail_port": "587",
    "mail_proxy": "",
    "session_secret": "abc",
    "password_reset_token_expiration_time": 7200000,
    "base_path": "/unisketch8/"
  },
  "test": {
    "port": 8084,
    "mail_host": "smtp.sendgrid.com",
    "mail_user": "unisketch_htw",
    "mail_email": "unisketch@htw-berlin.de",
    "mail_password": "g03ZvpWhi6tPJYqZ",
```

```

    "mail_port": "587",
    "mail_proxy": "",
    "session_secret": "abc",
    "password_reset_token_expiration_time": 7200000,
    "base_path": "/unisketch8/"
  },
  "production": {
    "port": 8084,
    "mail_host": "smtp.sendgrid.com",
    "mail_user": "unisketch_htw",
    "mail_email": "unisketch@htw-berlin.de",
    "mail_password": "g03ZvpWhi6tPJYqZ",
    "mail_port": "587",
    "mail_proxy": "",
    "session_secret": "abc",
    "password_reset_token_expiration_time": 7200000,
    "base_path": "/unisketch8/"
  }
}

```

Entscheidend ist hier, dass das Attribut *port* den Wert zugewiesen bekommt, der in Kapitel 4.2.2 in der NGINX-Konfigurationsdatei festgelegt wurde. Ebenso muss das Attribut *base\_path* den Wert zugewiesen kommen, der in der NGINX-Konfigurationsdatei als *location* angegeben ist. Und das jeweils für *development*, *test* und *production*.

5. Nun muss die Konfigurationsdatei *database.json* angepasst werden, die sich ebenfalls in dem Unterordner *shared/config/* befindet.

```
vim database.json
```

Für UniSketch8 sähe die *database.json* beispielsweise so aus:

```

{
  "development": {
    "type": "postgres",
    "host": "127.0.0.1",
    "port": 5432,
    "username": "sketchpad3",
    "password": "GNA1rnw_",
    "database": "unisketch8_development",
    "synchronize": true,
    "logging": true,
  }
}

```



```

    "entities": [
      "src/models/**/*.ts"
    ],
    "migrations": [
      "src/migration/**/*.ts"
    ],
    "subscribers": [
      "src/subscriber/**/*.ts"
    ]
  },
  "test": {
    "type": "postgres",
    "host": "127.0.0.1",
    "port": 5432,
    "username": "sketchpad3",
    "password": "GNA1rnw_",
    "database": "unisketch8_test",
    "synchronize": true,
    "logging": false,
    "entities": [
      "src/models/**/*.ts"
    ],
    "migrations": [
      "src/migration/**/*.ts"
    ],
    "subscribers": [
      "src/subscriber/**/*.ts"
    ]
  },
  "production": {
    "type": "postgres",
    "host": "127.0.0.1",
    "port": 5432,
    "username": "sketchpad3",
    "password": "GNA1rnw_",
    "database": "unisketch8",
    "synchronize": false,
    "logging": false,
    "entities": [
      "src/models/**/*.ts"
    ]
  },

```

```

    "migrations": [
        "src/migration/**/*.ts"
    ],
    "subscribers": [
        "src/subscriber/**/*.ts"
    ]
}
}

```

Hier muss lediglich der Name der jeweiligen Datenbank geändert werden. Die Namen müssen denen entsprechen, die im Kapitel 3.2.3 vergeben wurden.

#### 4.2.6 Variable im Quellcode der Anwendung anpassen

Damit das Frontend auch das Backend der neu eingerichteten UniSketch-Version findet, muss im Quellcode an einer Stelle eine Änderung vorgenommen werden. Für UniSketch8 sähe das so aus:

```
vim UniSketch8/unisketch4/src/app/globals.ts
```

Hier muss nun die Variable *BASE\_PATH* angepasst werden:

```
static readonly BASE_PATH: string = '/unisketch8';
```

Die Änderung muss in den Branch *master* eingepflegt werden und ins Remote-Repository gepusht werden. Nun sollte einmal deployed werden (siehe Kapitel 4.3).

#### 4.2.7 To whom it may concern

Der ganze Installationsprozess ist einigermaßen aufwändig. Sollte eine zukünftige Projektgruppe die Zeit und Muße haben, den Inhalt von Kapitel 4.2 in einem Bash- oder (besser noch) Python-Skript zu automatisieren, so würde zukünftigen Projektgruppen einiges an Arbeit erspart. Idealerweise würde dieses Skript die Funktionalitäten der vorhandenen Skripte *deploy.sh*, *restart\_server.sh* und *cache\_node\_modules.sh* übernehmen - der Aufruf des Skripts könnte dann mit Argumenten (beispielsweise *-d* zum Deployen oder *-n* zum Installieren einer neuen UniSketch-Version) geschehen.

## 4.3 Deployment eines neues Entwicklungsstands

Die Gruppe von UniSketch7 konnte aufgrund eines nicht erreichbaren Servers das System nicht deployen. Die folgenden Hinweise stammen vom UniSketch6 Team und wurden nicht verändert. Zum Deployen eines neuen Stands der Anwendung müssen alle Änderungen in den Branch *master* eingepflegt werden. Dabei gilt zu beachten, dass der Branch *master* immer und zu jedem Zeitpunkt lauffähig sein soll. Entsprechende Vorgehensweisen bezüglich des Mergens und des Testens sind zu berücksichtigen.

Sind alle Änderungen im Branch *master* des Remote-Repositories, dann sind zum Deployen folgende Befehle nötig (beispielhaft für UniSketch8):

```
ssh sketchpad3@vr-cave.f4.htw-berlin.de
cd ~/unisketch8/
./deploy.sh
./restart_server.sh
./cache_node_modules.sh
```

Das Cachen der Node-Module ist optional, wird aber empfohlen, um die Build-Geschwindigkeit beim nächsten Build zu erhöhen.

Der aktuelle Stand des Branches *master* des Remote-Repositories ist nun deployed und erreichbar.

# 5 Entwicklungsverlauf und implementierte Features im WiSe2019/2020

## 5.1 Scrum

### 5.1.1 Rollen

- Product Owner: Prof.Dr.Ing.Johann Habakuk Isreal
- Scrum Master: Eduard Seiler
- Entwickler: Christopher Lorenz, Florian Stallmach, Talisa Wahle, Dennis Adler

### 5.1.2 Definition of Done

Use Cases müssen der Definition of Done entsprechen um vom Product Owner abgenommen zu werden.

Bei der Entwicklung von UniSketch7 wurde sich im Vorraus auf folgende Punkte geeinigt:

- Namen für Methoden und Klassen werden so gewählt, dass sie selbstbeschreibend sind
- Code wird manuell getestet
- Eine Codereview wird von min. einem anderen Entwickler durchgeführt
- Der Code wird vor Abnahme durch den PO <sup>1</sup> in die Versionsverwaltung eingcheckedt
- Orientierung an Clean Code[13]

### 5.1.3 Sprints

Im Entwicklungsverlauf gab es 3 Sprints. Die Errungenschaften aus den jeweiligen Abschnitten sind nachfolgend kurz beschrieben.

---

<sup>1</sup>Product Owner

## **Sprint 1**

Im ersten Entwicklungsabschnitt haben wir uns ins System, sowie in die für uns neue Sprache Typescript eingearbeitet. Es ging darum ein Verständnis dafür zu entwickeln, wie das System arbeitet und Ansätze zu finden es zu erweitern.

Resultate:

- Features haben noch nicht der 'Definition of Done' entsprochen und wurden vom Product Owner abgelehnt

## **Sprint 2**

Die in Sprint 1 gestarteten Features wurden in diesem Sprint zu einem Großteil fertig gestellt.

Implementiert wurden folgende Features:

- Benutzer können nun zwischen 3 Downloadformaten auswählen und das gewünschte Format herunterladen.
- Benutzer können nun die zusätzliche Form Dreieck auswählen und gleichschenklige Dreiecke zeichnen.
- (Benutzer können mit Hilfe eines Sliders feingranularer in die Skizze hinein- und hinauszoomen.)

## **Sprint 3**

Es wurden neue Features gestartet, sowie die beiden großen Features Graffiti-Brush und Image-Tool aus Sprint 2 weitergeführt.

Implementiert wurden folgende Features:

- Benutzer können nun Bilder in ihre Skizze einfügen.
- Benutzer können nun ihre eigenen Skizzen als Profilbilder hochladen.
- Benutzer können nun den Graffiti-Brush nutzen um Graffiti zu zeichnen.
- Dem Benutzer wird nun beim Verwenden verschiedener Werkzeuge der passende Cursor angezeigt.

- Benutzer können nun ein Raster aktivieren um Maßstab gerecht zeichnen zu können.

# 6 Weiterentwicklung

## 6.1 Offene Punkte aus dem Product-Backlog

Um der nachfolgenden Gruppe Ideen zu geben, was von Ihnen umgesetzt werden könnte, sind hier einige Punkte, welche sich am Ende der Entwicklung noch in unserem Product-Backlog befanden, aber nicht von uns umgesetzt worden sind:

- Als Benutzer\*in möchte ich so Radieren, dass nur die Teile von Linien gelöscht werden, über die ich den Radierer bewege.
- Als Benutzer\*in möchte ich eine Magnetfunktion für das das Zeichnen auf dem Grid (Linie wird an die am nächsten an ihr dran liegende Linie des Grids gezogen)
- Als Benutzer\*in möchte ich Skizzenelemente auf Ebenen verwalten
- Als Benutzer\*in möchte ich nachträglich Skizzen-Element-Eigenschaften ändern (Farbe, Transparenz)
- Als Benutzer\*in möchte ich gerade Linien erstellen können

Die gerade erwähnten Features sind nach deren Priorisierung in unserem Product Backlog geordnet. Es empfiehlt sich trotzdem mit anderen Dingen anzufangen, um das System besser kennenzulernen. Es empfehlen sich bspw. Features wie das Abschicken einer Chat Nachricht mit der ENTER-Taste.

## 6.2 Bekannte Bugs, Fehler und generell Verbesserungsfähiges

Es gibt noch bekannte Bugs, welche noch nicht behoben worden sind:

- Beim Speichern einer Skizze und der darauffolgenden Erstellung des Preview-Images wird momentan die Schriftfarbe nicht richtig einbezogen.
- Der Text, welcher im Text-Werkzeug genutzt wird, wird nicht enkodiert/dekodiert. Dies ist fehleranfällig (wenn Texte Umlaute oder ein ß enthalten, dann kann die Skizze nicht verlassen werden).

- Das Fade-In/-Out in den Sketch-Toolmenüs könnte optimiert werden. Möglicher Optimierungsansatz ist zu überprüfen, ob ein Transition-Wert in CSS gesetzt ist.
- Ein Bug, der seit der vorherigen Gruppe besteht, sorgt für ein eigenartiges Resize-Verhalten. Dies könnte überarbeitet werden.
- Es gibt einige Probleme mit dem neuen Zoom-Slider. Das Selector Tool funktioniert auf allen Zoomstufen, jedoch mit Einschränkungen. So muss mit jeder Zoom Stufe ein größerer Bereich ausgewählt werden, damit ein Element erkannt wird. Mit einem Klick auf das Element wird es richtig ausgewählt und es lässt sich erkennen, wie groß der Bereich zum Auswählen per Mouse-Drag hätte sein müssen. Dies lässt sich durch einbeziehen der Zoomstufe ins Selector Tool lösen. Des Weiteren gibt es Probleme mit der Skizzen Preview. Durch den großen Zoom Bereich ist das Canvas sehr groß geworden. Für Die Preview wird allerdings das ganze Canvas herangezogen, was folgendes bedeutet:  
Falls der Nutzer nur auf der normalen Zoomstufe (25%) zeichnet ist seine Zeichnung nicht in der Preview zu sehen, da sie nur einen sehr kleinen Teil des Canvas füllt.  
Auch mit den Bildern gibt es kleinere Probleme, sowohl mit dem alten Zoom, als auch mit dem neuen. Bei Upload eines Bildes wird eine Preview angezeigt, diese ist über die Bildschirmgröße skaliert, optimal wäre es allerdings, wenn die Preview auf jeder Zoomstufe die selbe Größe hat, wie das schlussendlich hinzugefügte Bild.

Neben den Bugs kann allgemein die Mobile-Responsiveness, sowie die Performance verbessert werden. Eines der Probleme der Performance ist, dass bei vielen Objekten ein Overhead entsteht. Mögliche Gegenmaßnahmen sind Pipes.

Nicht zwingend notwendig, aber sicherlich hilfreich wäre es die große HTML-Datei der Toolbar in mehrere Teile zu zerlegen und per `<include>` zusammenzusetzen. Auch das Auslagern einiger Teile der Komponenten erachten wir als sinnvoll, da diese sehr unübersichtlich sind.

## 6.3 Anmerkungen für Entwickler

Beim Image-Tool gibt es eine kleine Eigenheit, welche zu beachten ist.

Zu einer Skizze hinzugefügte Bilder werden im Server unter *images* gespeichert. In der Datenbank liegt nur der Speicherplatz dieses Bildes. Wenn beim Entwickeln der Branch gewechselt wird, ist dieses Bild nichtmehr im *image* Ordner des Servers, da diese Bilder im Git-Repository nichts verloren haben. Solltet ihr nun wieder in eure Skizze wollen, stürzt der Server ab, da das Bild nicht gefunden werden kann. Dies passiert immer, wenn die Datenbank und der *images* Ordner nicht synchronisiert sind. Es wäre sinnvoll für diesen Fall, ein Platzhalter Bild auf dem Server zu



platzieren und einzusetzen, falls das Original nicht gefunden werden kann. Mit diesem Quickfix lässt sich dieses Problem vermeiden. Für die Preview-Images gilt selbes.

Anmerkung: Auf einem deployten System kann dies nicht passieren, da alle Bilder sowohl in der Datenbank, als auch im Server liegen und nicht durch verschiedene Entwicklungszweige aus ihrer Synchronisation herauskommen.

# Literatur

- [1] *Berkeley Software Distribution - Wikipedia*. Wikipedia. URL: [https://en.wikipedia.org/wiki/Berkeley%5C\\_Software%5C\\_Distribution](https://en.wikipedia.org/wiki/Berkeley%5C_Software%5C_Distribution) (besucht am 03.03.2019).
- [2] *Bootstrap · The most popular HTML, CSS, and JS library in the world*. Bootstrap Team. URL: <https://getbootstrap.com/> (besucht am 03.03.2019).
- [3] *Express - Node.js web application framework*. Node.js Foundation. URL: <https://expressjs.com/> (besucht am 03.03.2019).
- [4] *Git*. Git Community. URL: <https://git-scm.com/> (besucht am 03.03.2019).
- [5] *Google Chrome: The Most Secure Browser on the Web*. Google. URL: <https://www.google.com/chrome/> (besucht am 03.03.2019).
- [6] *Grid system · Bootstrap*. Bootstrap Team. URL: <https://getbootstrap.com/docs/4.0/layout/grid/> (besucht am 03.03.2019).
- [7] *HTW Berlin - Hochschule für Technik und Wirtschaft Berlin*. HTW Berlin. URL: <https://www.htw-berlin.de/> (besucht am 03.03.2019).
- [8] *Internet Web Browser for Desktop & Mobile - Edge – Microsoft*. Microsoft Corporation. URL: <https://www.microsoft.com/en-us/windows/microsoft-edge> (besucht am 03.03.2019).
- [9] *Linux - Wikipedia*. Wikipedia. URL: <https://en.wikipedia.org/wiki/Linux> (besucht am 03.03.2019).
- [10] *macOS Mojave - Apple*. Apple Inc. URL: <https://www.apple.com/lae/macos/mojave/> (besucht am 03.03.2019).
- [11] *Manual / Sequelize*. Sequelize Community. URL: <http://docs.sequelizejs.com/> (besucht am 03.03.2019).
- [12] *Mapping - Definition*. Google. URL: [https://www.google.com/search?q=mapping&rlz=1C1CHBF\\_deDE825DE825&oq=mapping&aqs=chrome..69i57j0l7.1992j0j4&sourceid=chrome&ie=UTF-8](https://www.google.com/search?q=mapping&rlz=1C1CHBF_deDE825DE825&oq=mapping&aqs=chrome..69i57j0l7.1992j0j4&sourceid=chrome&ie=UTF-8) (besucht am 04.09.2019).
- [13] Robert C. Martin. *Clean Code*. Prentice Hall, 2009.
- [14] *NGINX / High Performance Load Balancer, Web Server, & Reverse Proxy*. NGINX, Inc. URL: <https://www.nginx.com/> (besucht am 11.11.2018).
- [15] *Node.js*. Node.js Foundation. URL: <https://nodejs.org/en/> (besucht am 03.03.2019).

- [16] *npm*. npm, Inc. URL: <https://www.npmjs.com/> (besucht am 03.03.2019).
- [17] *Object-relational mapping* - *Wikipedia*. Wikipedia. URL: [https://en.wikipedia.org/wiki/Object-relational%5C\\_mapping](https://en.wikipedia.org/wiki/Object-relational%5C_mapping) (besucht am 03.03.2019).
- [18] *PostgreSQL: Documentation*. PostgreSQL Global Development Group. URL: <https://www.postgresql.org/docs/> (besucht am 03.03.2019).
- [19] *PostgreSQL: Documentation: 9.2: psql*. PostgreSQL Global Development Group. URL: <https://www.postgresql.org/docs/9.2/app-psql.html> (besucht am 03.03.2019).
- [20] *PostgreSQL: The World's Most Advanced Open Source Relational Database*. PostgreSQL Global Development Group. URL: <https://www.postgresql.org/> (besucht am 12.11.2018).
- [21] *Safari* - *Apple*. Apple Inc. URL: <https://www.apple.com/safari/> (besucht am 03.03.2019).
- [22] *Socket.IO*. Socket.IO Community. URL: <https://socket.io/> (besucht am 03.03.2019).
- [23] *The Chromium Projects*. The Chromium Projects. URL: <https://www.chromium.org/> (besucht am 03.03.2019).
- [24] *The new, fast browser for Mac, PC and Linux / Firefox*. Mozilla Foundation. URL: <https://www.mozilla.org/en-US/firefox/> (besucht am 03.03.2019).
- [25] *TypeORM* - *Open Collective*. Open Collective. URL: <https://opencollective.com/typeorm> (besucht am 03.03.2019).
- [26] *TypeScript* - *JavaScript that scales*. Microsoft Corporation. URL: <https://www.typescriptlang.org/> (besucht am 03.03.2019).
- [27] *Windows / Offizielle Website für die Betriebssysteme Microsoft Windows 10 Home und Pro, für Laptops, PCs, Tablets und mehr*. Microsoft Corporation. URL: <https://www.microsoft.com/de-de/windows> (besucht am 03.03.2019).