

Bachelorarbeit im Studiengang Bibliotheks- und Informationsmanagement  
Entwicklung eines User-Interfaces (Mockups) für die Interaktion mit Forschungssoftware

vorgelegt von Robert Christian Vater  
an der Hochschule der Medien Stuttgart  
am 31.08.2020

zur Erlangung des akademischen Grades eines Bachelor of Arts

Erst-Prüfer: Prof. Markus Hennies  
Zweit-Prüfer: Dipl. Inf. Pascal Seeland

„Hiermit versichere ich, Robert Christian Vater, ehrenwörtlich, dass ich die vorliegende Bachelorarbeit (bzw. Masterarbeit) mit dem Titel: „Entwicklung eines User-Interfaces (Mockups) für die Interaktion mit Forschungssoftware“ selbstständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden.

Ich habe die Bedeutung der ehrenwörtlichen Versicherung und die prüfungsrechtlichen Folgen (§26 Abs. 2 Bachelor-SPO (6 Semester), § 24 Abs. 2 Bachelor-SPO (7 Semester), § 23 Abs. 2 Master-SPO (3 Semester) bzw. § 19 Abs. 2 Master-SPO (4 Semester und berufsbegleitend) der HdM) einer unrichtigen oder unvollständigen ehrenwörtlichen Versicherung zur Kenntnis genommen.“

## Inhaltsverzeichnis

Abbildungsverzeichnis.....	4
Tabellenverzeichnis.....	4
Abkürzungsverzeichnis.....	5
Abstract in Deutsch.....	5
Abstract in Englisch.....	6
1 Die Einleitung.....	7
1.1 Die Ziele und Methodik der Bachelorarbeit.....	7
1.2 Die Usability einer Anwendung.....	8
1.3 Die User Experience einer Anwendung.....	10
1.4 Das Markenimage in Bezug auf die User Experience.....	11
1.5 Die Abgrenzung beider Begriffe.....	12
2 Die Anwendung VipLab von der Universität Stuttgart.....	12
2.1 Die zentralen Aufgaben von VipLab.....	12
2.2 VipLab im Kontext zum Forschungsdatenmanagement.....	14
2.3 Eine Interpretation und Beschreibung des Ist-Zustands von VipLab.....	16
2.4 Die Normen zur User Experience einer Anwendung.....	20
3 Der Projektablauf.....	23
3.1 Die Normen zum Entwicklungsprozess einer Anwendung.....	23
3.2 Die nutzerzentrierte Entwicklung.....	25
3.3 Agil bis ans Ziel: Der agile Projektablauf.....	26
3.3.1 Die statische Methode.....	26
3.3.2 Die agile Methode.....	26
3.4 Die einzelnen agilen Projektphasen für das Projekt VipLab.....	27
4 Die Analysephase.....	29
4.1 Die Theorie zu Personas.....	29
4.2 Die Erstellung einer Persona.....	30
4.3 Die Datengrundlage für die Personas.....	32
5 Die Konzeptionsphase / der Soll – Zustand.....	33
5.1 Die Theorie zu Use Cases.....	33
5.2 Das Use Case Diagramm zum Ist-Zustand von VipLab.....	35
5.3 Die Theorie rund um die Wireframes.....	36
5.4 Die Umsetzung der Wireframes für das Projekt VipLab.....	37
5.5 Die Theorie rund um die Icons.....	42
5.6 Die gegenwärtigen Icons in VipLab bezogen auf den Ist-Zustand.....	43
5.7 Die internationalen Standards im Webdesign.....	44
5.7.1 Der Standard (Styleguide) für den Soll-Zustand von VipLab.....	45
5.7.2 Die Icons nach einem Standard für den Soll-Zustand von VipLab.....	46
5.8 Die Theorie zu den Buttons.....	49
5.9 Die Labels für die Inhaltsbereiche in VipLab.....	50
5.10 Die Button-Hierarchie.....	51
5.11 Die Farbauswahl für die Buttons in dem Soll-Zustand.....	52
6 Die Umsetzungsphase.....	53
6.1 Die Theorie zu Mockups und interaktiven Prototypen.....	53
6.1.1 Medium-Fidelity.....	55
6.1.2 High-Fidelity.....	56
6.1.3 Die Umsetzung der interaktiven Funktionen im Prototyp.....	56
6.2 Der ausgearbeitete Soll-Zustand von VipLab.....	57
7 Die Testingphase.....	63
7.1 Die Theorie zu den Usability-Tests.....	63
7.1.1 Der Inhalt eines Usability-Tests.....	64
7.1.2 Die Wahl zur Art des Usability-Tests.....	65
7.1.3 Die Durchführung von Remote-Usability-Tests.....	66
7.2 Die Auswertungen der Usability-Tests für die Prototypen.....	67
7.2.1 Der Prototyp Forscher.....	67
7.2.2 Der Prototyp Student.....	71
8 Die Zusammenfassung.....	75
Das Quellenverzeichnis.....	76

## Abbildungsverzeichnis

Abbildung	Seite
<i>Abbildung 1: Der gegenwärtige Ist-Zustand von VipLab (eigene Darstellung)</i>	17
<i>Abbildung 2: Das Download/Upload-Fenster von VipLab</i>	18
<i>Abbildung 3: Der Download einer Grafik in VipLab</i>	19
<i>Abbildung 4: Der Berechnungsprozess in VipLab</i>	19
<i>Abbildung 5: Weitere Use Cases in VipLab nach der Berechnung</i>	19
<i>Abbildung 6: Das Use Case Diagramm vom Ist-Zustand ohne Ergänzungen (eig. Darstel.)</i>	35
<i>Abbildung 7: Das erste Wireframe vor der Diskussion und Beurteilung (eigene Darstellung)</i>	38
<i>Abbildung 8: Das zweite Wireframe nach der Beurteilung und der Überarbeitung (eig. Dar.)</i>	39
<i>Abbildung 9: Das Use Case Diagramm vom Soll-Zustand nach den Überarbeitungen</i>	41
<i>Abbildung 10: Ein Beispiel Icon aus Bootstrap</i>	42
<i>Abbildung 11: Die Icons für den Soll-Zustand</i>	46
<i>Abbildung 12: Ein Ausschnitt aus Google Docs</i>	47
<i>Abbildung 13: Das Gestaltungsgesetz der Ähnlichkeit</i>	50
<i>Abbildung 14: Die Startseite von VipLab - VipLab View Student 3</i>	58
<i>Abbildung 15: Die IDE vergrößert - VipLab View Student 9</i>	60
<i>Abbildung 16: Der Output-Bereich vergrößert - VipLab View Student 7</i>	61
<i>Abbildung 17: Der Output-Bereich vergrößert 2 - VipLab View Student 8</i>	61
<i>Abbildung 18: Das Tab für mehrere Downloads - VipLab View Student 4</i>	62
<i>Abbildung 19: Textzeilen als Output im Tab - VipLab View Student 5</i>	62
<i>Abbildung 20: Die Grafik umgeschaltet auf die 3D Ansicht - VipLab View Student 10</i>	63
<i>Abbildung 21: Die Persona Thorsten Mayer (eigene Darstellung)</i>	70
<i>Abbildung 22: Die Persona Sabine Müller (eigene Darstellung)</i>	74

## Tabellenverzeichnis

Tabelle	Seite
<i>Tabelle 1: Der Projektablauf mit den Methoden</i>	29
<i>Tabelle 2: Die Auswertung des Forscher - Prototyp (Die Funktionen / Use Cases)</i>	67
<i>Tabelle 3: Die Auswertung des Forscher - Prototyp (Die Gestaltung / Das Design)</i>	68
<i>Tabelle 4: Die Auswertung des Forscher - Prototyp (Die Interaktionsaufgaben)</i>	69
<i>Tabelle 5: Die Auswertung des Studenten - Prototyp (Die Funktionen / Use Cases)</i>	71
<i>Tabelle 6: Die Auswertung des Studenten - Prototyp (Die Gestaltung / Das Design)</i>	72
<i>Tabelle 7: Die Auswertung des Studenten - Prototyp (Die Interaktionsaufgaben)</i>	73

## Abkürzungsverzeichnis

Abkürzung	ausgeschrieben
DaRUS	The Data Repository of the University of Stuttgart
IDE	Integrated Development Environment
ISO	International Organization for Standardization
UI	User Interface
UX	User Experience
TIK	Technische Informations- und Kommunikationsdienste

## Abstract in Deutsch

Die Universität Stuttgart stellt für die Lehre die Onlineanwendung VipLab zur Verfügung. In dieser können Studenten an Prüfungsaufgaben für Module teilnehmen. Die Anwendung wird zwar schon aktiv angewandt, jedoch ist die grafische Benutzeroberfläche nicht ausreichend weiterentwickelt und führt häufig zu Anwendungsschwierigkeiten. Das Ziel der Bachelorarbeit ist es daher, die Benutzeroberfläche mittels einer neuen UI/UX für die Nutzer zu verbessern, um die Usability zu erhöhen.

Die Bachelorarbeit wurde in vier Projektphasen eingeteilt. Die Analyse-, Konzeptions-, Umsetzungs-, und Testingphase. Die Konzeptionsphase war dabei in der Entwicklungsarbeit am stärksten ausgeprägt, aufgrund der schon bestehenden Anwendung VipLab (dem Ist-Zustand) mit den zentralen Use Cases. Mit dieser Grundlage wurde der Soll-Zustand für VipLab erarbeitet, unter Berücksichtigung der aktuellen Theorie zu UI/UX. In der Umsetzungsphase wurden alle Ergebnisse aus der Konzeption in zwei resultierenden Prototypen zusammengeführt. Abschließend wurden in der Testingphase die Prototypen mit den Probanden validiert. Dafür wurden Remote-Usability-Tests als Methode zu Erhebung eingesetzt.

Als Ergebnis der Remote-Usability-Tests hat sich ergeben, dass die Benutzbarkeit in der Anwendung deutlich erschwert wird, wenn die Button-Form und Farbe nicht eindeutig zu den anderen Elementen unterschiedlich ist. Ein interaktiver Button muss in der Anwendung deutlich eine eigene Form und Farbe besitzen, um richtig von den Nutzern interpretiert werden zu können. In den Tests hat sich außerdem gezeigt, dass Icons ebenso unmittelbar durch den Nutzer verstanden werden müssen. Einige Probanden konnten ein Icon nicht sofort zu dem passenden Use Case zuordnen.

Als Fazit konnte gezogen werden, dass an manchen Stellen in den Prototypen daher eine Überarbeitung noch notwendig war. Mit den Verbesserungen wird geraten, nochmals die Prototypen mit den Probanden zu validieren.

### **Abstract in English**

The University of Stuttgart provides the online application VipLab for teaching. In this, students can take exams for modules. The application is already being used actively, but the graphical user interface has not been sufficiently developed and often leads to application difficulties. The aim of the bachelor thesis is therefore to improve the user interface for the users by means of a new UI/UX in order to increase usability.

The bachelor thesis was divided into four project phases. The analysis, conception, implementation and testing phase. The concept phase was most pronounced in the development work, due to the already existing application VipLab (the current state) with the central use cases. On this basis, the target state for VipLab was developed, taking into account the current theory of UI / UX. In the implementation phase, all results from the conception were brought together in two resulting prototypes. Finally, the prototypes were validated with the test subjects in the testing phase. Remote-usability-tests were used as a method for the survey.

The result of the remote-usability-tests has shown that the usability in the application is made significantly more difficult if the button shape and color are not clearly different from the other elements. An interactive button must clearly have its own shape and color in the application in order to be correctly interpreted by the users. The tests also showed that icons must also be understood immediately by the user. Some subjects could not immediately assign an icon to the appropriate use case.

In conclusion, it could be drawn that in some places in the prototypes a revision was still necessary. With the improvements it is advised to validate the prototypes again with the test subjects.

## 1 Die Einleitung

### 1.1 Die Ziele und Methodik der Bachelorarbeit

Seit 2009 arbeitet das TIK (Technischer Informations- und Kommunikationsdienst) der Universität Stuttgart mit verschiedenen in der Numerik-Ausbildung engagierten Instituten zusammen, um ein virtuelles Programmierlabor (VipLab) zu entwickeln. Die technische Entwicklung wie auch der laufende Betrieb liegt dabei bei dem TIK. Das Projekt VipLab umfasst eine browserbasierte Programmierumgebung, Rechenbackends auf Servern des TIK sowie eine Middleware zur Kommunikation. Ursprünglich von Studenten angestoßen und durch Studiengebühren finanziert, wird VipLab seit 2010 aktiv in der Lehre eingesetzt. Für den Lehrbetrieb sind jeweils die einzelnen Institutionen der Universität Stuttgart eigenständig verantwortlich, um das Tool mit Aufgaben für den Unterricht zu pflegen.<sup>1</sup> Da das Tool in der Lehre aktiv mit integriert ist, muss auch die grafische Oberfläche der Anwendung für die Nutzer ansprechend organisiert sein. Daher wurde 2020 ein Bachelorthema für die grafische Überarbeitung von VipLab ins Leben gerufen.

#### *Entwicklung eines User-Interfaces (Mockups) für die Interaktion mit Forschungssoftware*

In erster Linie ist das Ziel in der Bachelorarbeit ein neues User-Interface für VipLab zu gestalten und zu testen. Dafür muss vorerst viel Theorie bearbeitet und richtig interpretiert werden. Dabei soll die Anwendung mit einer besseren User Experience überarbeitet werden. Die zentralen Elemente und Funktionen der Anwendung sollen neu überdacht werden. Sind die Icons noch aktuell und international verständlich? Wirkt die Anwendung auf den Nutzer benutzerfreundlich? Wie kann man für den Nutzer die Anwendung zugänglicher gestalten?

Dafür ist in der Bachelorarbeit natürlich ein methodisches Vorgehen wichtig, um die Ziele empirisch zu erreichen. Zunächst wird der Ist-Zustand der Anwendung erhoben. Aufgrund dessen wird ein Soll-Zustand definiert, der mit der aktuellen Theorie zu User Experience, erarbeitet wird. Der Soll-Zustand wird schließlich in interaktiven High Fidelity Prototypen praktisch umgesetzt. Gegen Ende der Bachelorarbeit werden die Prototypen mit den Nutzern von VipLab auf deren Tauglichkeit hin validiert und es wird ein Fazit gezogen.

---

1 Universität Stuttgart (2019). Forschungs- und Entwicklungsprojekte. Verfügbar unter: <https://www.tik.uni-stuttgart.de/forschung-und-lehre/forschungs-und-entwicklungsprojekte/>

VipLab wird heute schon im Kontext zum Forschungsdatenmanagement an der Universität Stuttgart aktiv mit eingesetzt. Die Anwendung wird betrieben, um einen reibungslosen Forschungsablauf an der Universität mit zu garantieren. Die Anwendung soll die Forschungsarbeit und -recherche aktiv mit unterstützen und erleichtern.

Die Bachelorarbeit wurde in Kooperation mit der Universität Stuttgart erarbeitet. Dabei fanden regelmäßige Projektbesprechungen statt, um das Arbeitsthema voranzutreiben. In der Bachelorarbeit wird zunächst im ersten Kapitel die Terminologie zum Thema User Experience und Usability näher erläutert, um die allgemeine Verständlichkeit der Projektausarbeitung im weiteren Leseverlauf zu garantieren.

## **1.2 Die Usability einer Anwendung**

Der Begriff Usability wird am besten mit Gebrauchs- oder Nutzungstauglichkeit ins Deutsche übersetzt. Das primäre Ziel der Usability ist es, eine Anwendung (z.B. eine Webseite oder App) in der Benutzung für den Nutzer so einfach wie möglich zu gestalten. Die Elemente beispielsweise einer Webseite sollten dabei nutzerfreundlich organisiert werden, sodass eine intuitive Bedienung möglich wird.

Für die Usability wurde auch eine eigene Norm verfasst, um einen allgemeinen Standard zu definieren. Auf die ISO-Norm kann sich immer berufen werden, wenn eine Anwendung aufgrund von auffälligen Usability-Problemen unbedingt überarbeitet werden muss. Die Norm kann dann als Rechtfertigung gegenüber dem Arbeitgeber angeführt werden, um eine theoretische Überarbeitung der Anwendung anzustoßen. Beispielsweise, wenn die Webseitenelemente neu strukturiert werden müssen, weil sich die Nutzer in der Anwendung nicht mehr zu Recht finden. Laut der ISO-Norm 9241-11 bezeichnet die Usability „Das Ausmaß, in dem ein Produkt, System oder Dienst durch bestimmte Benutzer in einem bestimmten Anwendungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen.“

Wenn ein Nutzer eine Webseite besucht, dann hat er eine bestimmte Absicht, was er auf der Plattform erreichen möchte. Auf einen Onlineshop übertragen, könnte er nach einem speziellen Produkt suchen oder nach einem Ansprechpartner für eine Produktinformation. Eine gute Usability macht sich an der Stelle bemerkbar, wenn die Anwendung so organisiert ist, sodass der Nutzer mühelos und schnell seine Absicht erreicht und die gesuchte Information findet. Er sollte keine Komplikationen auf dem Weg zur gesuchten Information erleben. Das bedeutet beispielsweise auch, dass die Informationsarchitektur auf der Webseite logisch oder intuitiv angelegt sein muss.



Im Menüpunkt „Kontakte“ findet sich dann beispielsweise auch der richtige Ansprechpartner für eine spezifische Produktinformation. Der Nutzer findet in der Anwendung immer dort seine gesuchten Informationen, wo er sie auch logischerweise erwarten kann. So wäre es auch schlecht für die Usability, wenn unter dem Menüpunkt „Kontakte“ Produkte zu finden wären. Kein Nutzer würde intuitiv ein Produkt unter dem Menüpunkt „Kontakte“ suchen.

Auf dem Weg zur benötigten Information sollte der Nutzer auch möglichst wenig Klicks benötigen. Eine gute Anwendungsusability kann daran gemessen werden, wie viele Klicks und damit Zeit ein Nutzer braucht, um eine Aufgabe (Absicht) zu erledigen. Die Aufgabe kann dabei darin bestehen, ein spezielles Produkt auf einer Webseite raus zu suchen. Braucht der Nutzer dafür mehr Klicks als nötig oder kommt er nicht schnell genug bzw. gar nicht zum Suchziel, so ist folglich die Wahrscheinlichkeit hoch, dass er genervt die besuchte Webseite wieder verlässt. Die Usability ist demnach schlecht auf der Anwendung organisiert. So kann in dem Beispiel die Informationsarchitektur im Menü der Webseite schlecht organisiert sein, und es gibt keinen Reiter für Produkte, der direkt angeklickt werden kann. Ein klares Kriterium für Softwarequalität ist daher die Effizienz. Eine Aufgabe sollte angemessen schnell zu erledigen sein. Dieses Kriterium findet sich auch in der ISO Norm 25000 „Qualitätskriterien und Bewertung von Softwareprodukten“.<sup>2</sup>

Eine höhere Usability wird u.a. auch schon damit erreicht, wenn die Anforderungen aus der Leitlinie zur Gestaltung von Benutzungsschnittstellen der DIN EN ISO 9241 erfüllt werden. Eine Anwendung sollte danach immer ...

- der Aufgabe angemessen sein,
- erwartungskonform sein,
- selbstbeschreibend sein,
- fehlertolerant sein,
- steuerbar sein,
- individualisierbar sein,
- lernförderlich sein.

Wird eine Anwendung anhand dieser Leitlinien entwickelt, dann wird schon damit eine effizientere Benutzbarkeit erreicht. Indem die Anwendung beispielsweise selbstbeschreibend ist, und nicht vorerst mehrmalig erlernt werden muss, kann der Nutzer seine Absichten schneller erreichen. Eine gute Usability wird von den Nutzern überwiegend gar nicht wahrgenommen, eine schlechte hingegen, macht sich sehr schnell bemerkbar und führt zu deutlichem Unmut. Die wichtigste Frage

---

<sup>2</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 62

in Bezug auf die Usability lautet daher immer: Konnte der Nutzer mit der bestehenden Seitenstruktur der Anwendung sein Ziel (seine Absicht) konfliktfrei erreichen?<sup>3</sup>

### **1.3 Die User Experience einer Anwendung**

Der Begriff User Experience wird am besten mit Nutzungserlebnis oder -erfahrung ins Deutsche übersetzt. Das Ziel von User Experience ist weitgreifender als von Usability. Der Benutzer soll im Gesamteindruck, beispielsweise mit einer Webseite, möglichst glücklich und zufrieden gestellt werden, sodass er gerne auf die Anwendung immer wieder zurückkommt.

Auch für die User Experience wurde eine Norm erlassen, die einen allgemeinen Standard definiert. Laut der ISO-Norm 9241-210 bezeichnet die User Experience: „Alle Aspekte der Erfahrungen eines Nutzers bei der Interaktion mit einem Produkt, Dienst, einer Umgebung oder Einrichtung“. Damit sind alle Nutzerinteraktionen gemeint, in denen Erfahrungen mit der Anwendung gesammelt werden können, die vor, während und nach der Anwendungsbenutzung stattfinden. Vor der Benutzung meint, die Vorerfahrungen oder die eigenen Erwartungen mit einer Anwendung. Diese Erwartungen/Vorerfahrungen können auch schon durch eine bekannte Marke vorgeprägt worden sein. Und aufgrund der bekannten Marke impliziert der Nutzer automatisch, dass die Website auch gut organisiert sein muss. Während der Benutzung meint beispielsweise, wenn man einen Artikel auf einer Webseite bestellt und dabei problemlos seine Absicht erreicht. Nach der Benutzung meint, wenn der Nutzer auf die Bestellung wartet und zum Artikelversand eine E-Mail erhält.

Die User Experience hat zur Aufgabe den Nutzer auch emotional zu befriedigen. Eine Anwendung soll den Nutzer deutlich glücklich machen. Er soll Spaß bei der Benutzung empfinden. Nicht nur die Funktionalität (Usability) steht dabei mit im Vordergrund sondern auch das Drumherum, das Gesamtbild muss stimmig sein, um positive Emotionen zu wecken. Somit können auch das Design und die Farben einen positiven Effekt auf den Nutzer haben. Die Usability ist somit nur ein Teil von der gesamten User Experience.

Für das Gesamtbild ist auch wichtig, dass der gesamte Prozessablauf eines Use Cases innerhalb einer Anwendung gut organisiert ist. Beispielsweise, wenn ein Nutzer einen Artikel in einem Onlineshop kaufen möchte, dann ist es für die User Experience nicht nur wichtig, dass er das Produkt schnell finden kann, der gesamte Einkaufsprozess muss in der Anwendung möglichst angenehm gestaltet sein. Das bedeutet darüber hinaus, dass sich der Nutzer in der

---

3 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 31-32

Einkaufsumgebung wohl fühlt. Die Farben, mit der die Anwendung gestaltet ist, gefallen ihm. Er erhält eine Bestätigungsmail, wenn seine Bestellung erfolgreich eingegangen ist und er bekommt eine Mail zugesendet, wenn der Einkauf verschickt wurde. Diese zusätzlichen Funktionen sollen dem Nutzer die Anwendung und den Einkaufsprozess möglichst umfänglich angenehm gestalten.

Die User Experience bezieht sich auch noch übergreifend auf alle anderen Bereiche, die mit dem Erleben der Anwendung zusammenhängen. Beispielsweise mit dem Servicebereich. Hat der Kunde eine Reklamation mit seinem gekauften Artikel, ist dann der Servicemitarbeiter am Telefon auch freundlich? Bei einer guten User Experience wird dem Nutzer in der Interaktion mit der gesamten Anwendung, mit den verschiedenen Diensten und Funktionen ein emotionaler Mehrwert gegeben, sodass er sich wohlfühlt und möglichst keine unangenehmen Situationen erlebt. Das bedeutet an der Stelle auch, dass der Servicemitarbeiter am Telefon freundlich und kompetent auftreten sollte. Die Anwendung soll den Nutzer ganzheitlich begeistern.

Bei der User Experience spielen alle Wahrnehmungen und Reaktionen einer Person in der Anwendungsinteraktion eine Rolle. Eine optimale User Experience ergibt sich also, wenn die Summe aus Gestaltung, Funktionalität und Leistung den Nutzer emotional anspricht. Der Nutzer soll nicht nur schnell seine Absichten in der Anwendung erreichen, er soll bei der Interaktion auch positive Gefühle entwickeln, wie Spaß und Freude.

Die wichtigste Frage in Bezug auf die User Experience lautet demnach: Konnte der Nutzer im Gesamterlebnis der Anwendung (z.B. einer Webseite) zufrieden oder glücklich gestellt werden?<sup>4</sup>

#### **1.4 Das Markenimage in Bezug auf die User Experience**

Das Markenimage einer Einrichtung kann noch vor der Anwendungsbenutzung beeinflussen, ob die Anwendung in der User Experience als positiv oder negativ bewertet wird. Ist das Markenimage positiv geprägt, dann kann sich das auch positiv auf das Nutzungserlebnis (die User Experience) mit auswirken. Wenn der Nutzer von einer Anwendung begeistert und überzeugt ist, dann baut er auch eine emotionale Bindung zu dieser auf. Er wechselt somit dann auch ungern die Anwendung und bleibt einem Anbieter treu. Gerade in Zeiten, in denen die Kunden einen Anbieter häufig gerne wechseln, kann versucht werden, mittels einer guten User Experience die Kunden dauerhaft zu binden.<sup>5</sup>

---

4 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 32-33

5 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 33

### **1.5 Die Abgrenzung beider Begriffe**

Zur Übersichtlichkeit wird User Experience und Usability hier nochmal voneinander abgegrenzt. User Experience beschreibt das Gesamterlebnis mit einer Anwendung. Die Usability hingegen beschreibt nur die Funktionalität einer Anwendung. Die Usability ist somit nur ein Teil innerhalb der User Experience. Die Usability zielt auf die grafische Oberfläche einer Anwendung ab, während die User Experience die gesamte Anwendung mit allen Prozessen beinhaltet. Dazu gehören sämtliche Services, Dienste und Use Case Prozessabläufe innerhalb einer Anwendung.<sup>6</sup>

Im zweiten Kapitel wird nun die Anwendung VipLab näher vorgestellt. Es wird erläutert, in welchem Umfeld und zu welchem Zweck VipLab eingesetzt wird. Des Weiteren wird auch der tatsächliche Ist-Zustand beschrieben und die Problematik von VipLab wird im Bezug zu einer internationalen Norm (ISO Norm 9241-151 „Leitlinien zur Gestaltung von Benutzungsschnittstellen für das World Wide Web“) näher verdeutlicht.

## **2 Die Anwendung VipLab von der Universität Stuttgart**

### **2.1 Die zentralen Aufgaben von VipLab**

Die Anwendung VipLab ist in ihrer Funktionalität hauptsächlich eine IDE, in der verschiedene Programmiersprachen ausgeführt werden können. Dazu zählen beispielsweise die Programmiersprachen C oder C++, die in den Interpreter problemlos eingegeben werden können. Die IDE ist dabei sehr minimalistisch gestaltet. Es gibt beispielsweise keine Leiste mit Reitern, in denen weitere Funktionalitäten zur IDE bereitgestellt werden. Das kann eine Funktion sein, wie der Verbindungsaufbau zu GitHub, wie man es beispielsweise aus der PyCharm kennt. In VipLab wurde aber absichtlich die Komplexität der IDE reduziert, indem Funktionen außen vor gelassen wurden. Der Student soll sich in der Entwicklungsumgebung ausschließlich nur auf den Code konzentrieren können, ohne von einer umfangreichen IDE und deren Komplexität zusätzlich belastet zu werden. Diese Philosophie soll auch bei der Neugestaltung der Anwendung beibehalten werden. Die IDE soll ihr minimalistisches Erscheinungsbild beibehalten, sodass der Student eine leichte und zugängliche IDE erlebt.

Eine Hauptaufgabe von VipLab ist es eine Entwicklungsumgebung bereitzustellen, in der Studenten Programmieraufgaben lösen können. Ein Dozent an der Universität Stuttgart kann dazu beliebig viele Textaufgaben in VipLab hineinstellen, die mit Codelösungen zu beantworten sind. Eine Bewertung der Aufgabe ist durch den Dozenten auch möglich. Es ist dem Dozenten dabei

---

<sup>6</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 33

freigestellt, ob er noch eine Codevorlage, die zur Lösung mit beiträgt, mit in VipLab hochlädt oder gänzlich auf jede Hilfestellung für den Studenten verzichtet. Eine Textaufgabe ist in der Anwendung erfolgreich gelöst, wenn die Codeantwort richtig in die IDE eingetragen wurde und im Output-Fenster die richtige Visualisierung oder Output-Zeile ausgegeben wird. Indem der Nutzer auf die nächste Aufgabe wechselt, wird die beantwortete Textaufgabe zur Bewertung eingereicht. VipLab wird aufgrund der Funktionalität als Prüfungsmedium oft in technischen Modulen an der Universität mit eingesetzt.

Neben der Prüfungsfunktionalität kann der Student die Anwendung auch als eine offene IDE verwenden, um verschiedenste Programmiersprachen zu üben. Dabei muss er keine Textaufgaben lösen. Er kann u. a. in VipLab beispielsweise eigene kleine Programme zur Übung programmieren.

Eine weiterer wichtiger Use Case der in VipLab implementiert werden soll, ist dass Forscher Berechnungen und Simulationen innerhalb der Anwendung ausführen und nachvollziehen können. Ein Forscher der zu einem Thema Recherchen anstellt, liest sich in der Regel viele Publikationen durch, um sich einen Überblick zum Sachverhalt zu verschaffen. Dabei sind in den Publikationen viele Berechnungen und Simulationen abgebildet, die beispielsweise in der Erhebungsphase entstanden. Oft sind diese Berechnungen und Simulationen beim ersten Mal schwierig zu verstehen und zu begreifen. Daher sollen diese über VipLab interaktiv für den Forscher zugänglich gemacht werden, sodass er sich leichter und schneller in den Sachverhalt einer Publikation einfinden kann. In der Anwendung kann der Forscher dann selber die Berechnung oder Simulation nachempfinden und verstehen, indem er sich mit den Zahlen, Daten und Fakten interaktiv beschäftigt. Er kann aufgrund dessen auch leichter einschätzen, ob die Erhebungsmethode und die gesamte Publikation für seine eigene Forschungsarbeit als Referenz dienen kann. Durch VipLab soll es dem Forscher leichter gemacht werden, zu entscheiden, ob er eine Quelle für die eigene Arbeit verwenden und replizieren möchte.

In einem Publikationsrepositorium werden viele unterschiedliche wissenschaftliche Arbeiten verwaltet. Die Universität Stuttgart betreibt derzeit selbst ein solches Repositorium namens „DaRUS“, indem alle wissenschaftlichen Publikationen der Institution archiviert werden. Die Universität hat sich zum Ziel gesetzt, dass in jeder, dort archivierten, wissenschaftlichen Veröffentlichung unter Umständen auch immer ein Link mit aufgeführt wird, der zur interaktiven Simulation in VipLab weiter durchstellt.

Viele Berechnungen oder Simulationen, die in einer wissenschaftlichen Veröffentlichung abgebildet sind, wurden auch mittels einer speziellen Software berechnet. In VipLab kann daher nicht nur Code ausgeführt werden. Es ist auch möglich eine Konfigurationsdatei zu der jeweiligen speziellen Software einzulesen, die dann in der Anwendung grafisch dargestellt werden kann. Die Voraussetzung ist jedoch, dass das Programm vorab in der Anwendung VipLab vorinstalliert worden ist. Dafür müssen die Forscher ihre Software, die sie in der Erhebungsphase verwendet haben, zur digitalen Langzeitarchivierung an der Universität mit einreichen.

Abschließend kann zusammengefasst werden, dass zwei zentrale Funktionalitäten in VipLab abgedeckt werden müssen. Zum einen als Übungsumgebung für Studierende und zum anderen als Rechercheinstrument für Forscher. Die Anwendung VipLab bedient somit primär zwei zentrale Nutzergruppen. Die des Forschenden und die des Lernenden. Für beide Nutzergruppen müssen in der fortlaufenden Bachelorarbeit zwei individuelle Benutzeroberflächen als Prototyp entwickelt werden.<sup>7</sup>

## **2.2 VipLab im Kontext zum Forschungsdatenmanagement**

Die Universität Stuttgart setzt die Anwendung VipLab bis heute hauptsächlich dafür ein, um Studenten eine Lernumgebung für Programmiersprachen bereitzustellen. In der weiteren Zukunft soll die Anwendung aber auch als interaktives Rechercheinstrument für Forscher mit eingesetzt werden. Damit wird VipLab zu einem Instrument, das aktiv zum Forschungsdatenmanagement an der Universität mit beiträgt.

Die zentrale Aufgabe von Forschungsdatenmanagement ist es, der Forschung eine bestmögliche Arbeitsumgebung und Infrastruktur bereitzustellen. Die Forschung soll mittels optimierten und organisierten Forschungsdatenmanagements erleichtert und zugänglicher werden, indem beispielsweise eine umfängliche Forschungsdatenverwaltung und -archivierung bereitgestellt wird. VipLab trägt unmittelbar dazu bei, dass nicht nur Publikationen recherchiert werden können. Die Anwendung ermöglicht es auch, dass wissenschaftliche Forschungsergebnisse von verschiedenen Publikationen leichter verstanden werden können. Die Funktionalitäten von VipLab bringen damit einen deutlichen Mehrwert für das Forschungsdatenmanagement an der Universität mit sich.

An der Universität Stuttgart werden viele Arten von wissenschaftlichen Publikationen veröffentlicht. Das können beispielsweise Bachelor- oder Masterarbeiten von Studenten sein, die eine gute Studienleistung erbracht haben. Es können aber auch wissenschaftliche Publikationen in Form von

---

<sup>7</sup> Persönliches Gespräch vom 15.Mai.2020 mit Pascal Seeland

Papers sein, die von Forschern erarbeitet wurden. Allen wissenschaftlichen Arbeiten liegt zugrunde, dass sie eine Erhebungsmethode verwendet haben, um eine Datengrundlage zu schaffen. Aufgrund dieser Datengrundlage sind in den Forschungsarbeiten verschiedenste Berechnungen oder Simulationen entstanden, die ein Forschungsergebnis entweder bestätigen oder falsifizieren.

Die Berechnungen oder Simulationen einer wissenschaftlichen Publikation werden zwar in der gedruckten Form erläutert, aber dennoch ist es oft schwer, die Berechnungen gleich nachzuvollziehen. VipLab wirkt dieser Problematik effektiv und elegant entgegen und fördert somit die Zugänglichkeit in vielen verschiedenen wissenschaftlichen Bereichen.

Damit VipLab als solches Instrument auch gerne von den Nutzern angenommen wird und Einzug in die tägliche Arbeit findet, ist es aber nicht nur erforderlich eine gute Idee und dementsprechende Funktionen bereitzustellen. Es ist auch notwendig, dass die grafische Oberfläche der Anwendung Gefallen findet und durch den Nutzer leicht und verständlich bedient werden kann. Anderenfalls kann eine Anwendung noch so einen guten Nutzen besitzen, die Nutzer würden aufgrund der schlechten Bedienbarkeit die Anwendung meiden und nicht mehr verwenden. Als ein Instrument für Forschungsdatenmanagement würde VipLab langfristig obsolet werden. Die Anwendung würde wieder abgeschaltet werden aufgrund der geringen Nutzerschaft. Die Aufrechterhaltung der Anwendung wäre langfristig zudem nicht sehr wirtschaftlich.

Es ist daher wichtig, dass die Anwendung eine sehr gute User Experience besitzt und das User-Interface für den Nutzer selbsterklärend ist. Damit wird das Risiko erheblich minimiert, dass die Anwendung durch die Nutzerschaft nicht angenommen wird. Die Funktionalitäten sowie auch das grafische Design muss der Nutzerschaft zusagen, um den Erfolg der Anwendung sicherzustellen.

VipLab wird derzeit für die Studentenschaft schon bereitgestellt und in der Betrachtung der Anwendung wird einem schnell klar, dass diese noch einen erheblichen Entwicklungsbedarf hat. In der heutigen grafischen Darstellungsform ist die Anwendung in der Benutzerfreundlichkeit noch sehr sperrig und schwierig zu verstehen. Daher ist das Ziel der Bachelorarbeit die Anwendung VipLab grafisch neu zu überarbeiten, sodass eine höhere Usability und mehr User Experience für die Nutzer zukünftig erreicht wird. Jedes Element in der heutigen Darstellungsform muss neu überdacht werden und mit der aktuellen Theorie in der Konzeptionsphase überarbeitet werden.

Zunächst ist es aber nötig einen fundierten Ist-Zustand zu erheben, der den tatsächlichen gegenwärtigen Zustand der Anwendung beschreibt. Damit wird auch klar, dass die Anwendung deutliche Usability-Probleme aufweist, resultierend im Vergleich mit der internationalen Norm. Anhand der identifizierten Elemente wird ersichtlich, welche für eine bessere User Experience in dem späteren Soll-Zustand gezielt überarbeitet werden müssen.

### **2.3 Eine Interpretation und Beschreibung des Ist-Zustands von VipLab**

In der unten abgebildeten Grafik sieht man die Anwendung VipLab, wenn eine Textaufgabe bearbeitet und die entsprechend richtige Codeantwort eingetragen wurden ist. Auf der linken Seite der Anwendung wird dann folglich die richtige Visualisierung oder Berechnung angezeigt, die von einem Dozenten bewertet werden kann.

Auf den ersten Blick wirkt das Erscheinungsbild der Anwendung sehr einschüchternd auf den Nutzer, weil die Oberfläche sehr wissenschaftlich dargestellt ist. Zum einen sieht man auf der rechten Seite der Anwendung Code und auf der linken mathematische Graphen. Die meisten Nutzer können schon aufgrund dessen die Anwendung wieder sehr schnell verlassen ohne wiederzukehren. Der Informationsgehalt der Anwendung ist sehr hoch und nur schwer für die meisten Menschen zu begreifen.

Die Inhalte der Anwendung werden aber nicht einfacher und die Komplexität bleibt weiterhin bestehen. Aber gerade deshalb ist es an dieser Stelle besonders wichtig wenigstens eine Entwicklungsumgebung bereitzustellen, die nicht zusätzlich einschüchternd und verängstigend wirkt. Gerade bei Studenten, die noch nie mit einer Programmiersprache gearbeitet haben, sollte die Anwendung von ihrem Erscheinungsbild daher beruhigend und leicht verständlich wirken. Die Anwendung sollte Interesse und Neugierde an deren Inhalten wecken. Zudem sollte die Entwicklungsumgebung sehr einfach zu erlernen sein. Mit einer durchdachten User Experience kann die Anwendung wenigstens von ihrem Erscheinungsbild und der Seitenstruktur so gestaltet werden, dass die Nutzerschaft nicht sofort davon abgeschreckt wird.



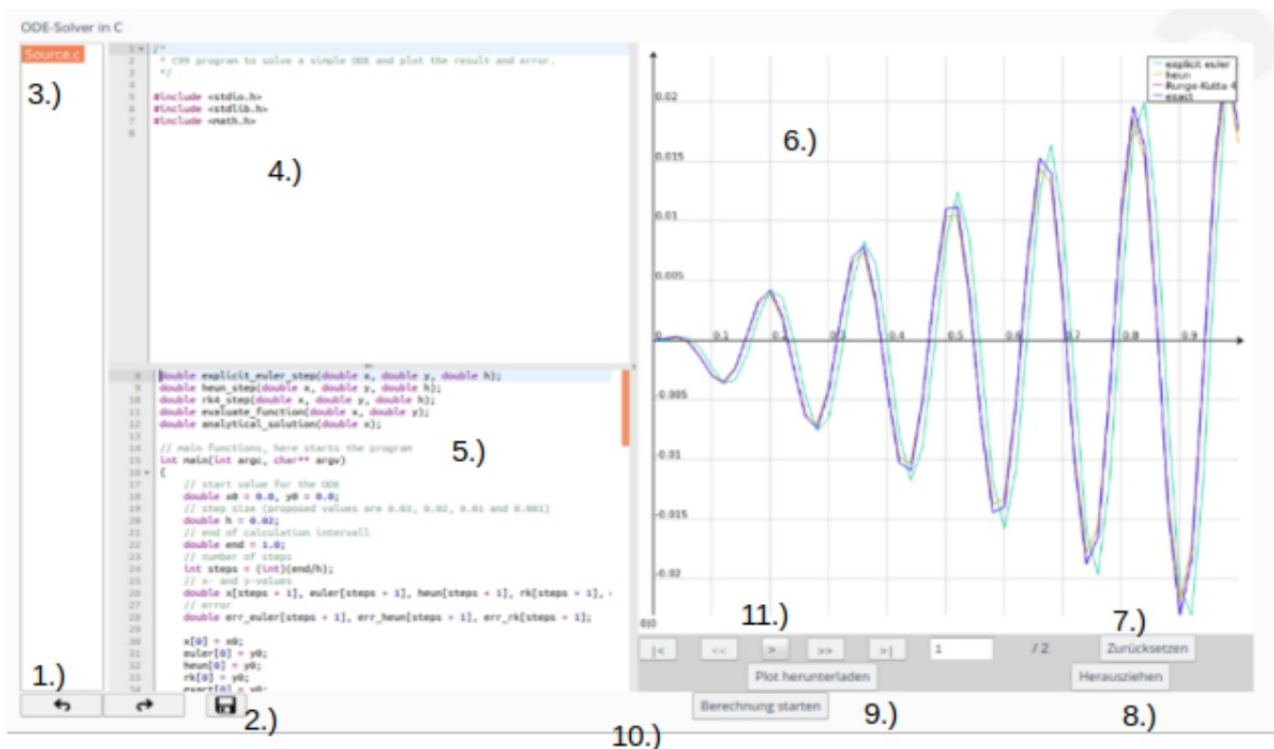


Abbildung 1: Der gegenwärtige Ist-Zustand von VipLab (eigene Darstellung)

Die Anwendung VipLab teilt sich in hauptsächlich drei wichtige Bereiche auf. Das Feld oben links enthält den Codeschnipsel, der von einem Dozenten fest vorgegeben sein kann, um einen Lösungsansatz für die Aufgabe vorzugeben. Anhand der Codevorgabe soll der Student den Code weiterschreiben, um eine funktionierende Lösung zu erzeugen. Das Feld darunter ist das eigentliche Lösungsfeld, in dem der Student die Aufgabenstellung vervollständigt.

Der Output des Interpreters wird auf der gesamten linken Seite dargestellt. Hier wird das Ergebnis angezeigt. Das kann entweder eine textuelle Zeilenausgabe sein oder eine aufwendige Visualisierung. Der Ist-Zustand bildet somit die Benutzeroberfläche für die Studenten ab. Für die Forscher mit deren spezifischen Bedürfnissen wird eine andere Benutzeroberfläche gestaltet.

1.) Die Funktionen "Wiederherstellen" und "Rückgängig": Diese Funktionen sind derzeit in VipLab ganz unten links angeordnet. Mit der Betätigung des Buttons wird der eingegebene Code in der IDE entweder rückgängig gemacht oder wiederhergestellt.

2.) Die Funktion Quellcode „hochladen“ und „runterladen“: Derzeit ist die Funktion neben der Schaltfläche „Wiederherstellen“ rechts zu finden. Als Icon ist hier eine Diskette ausgewählt. Mit Betätigung des Buttons wird ein weiteres kleines Fenster in der Anwendung geöffnet.

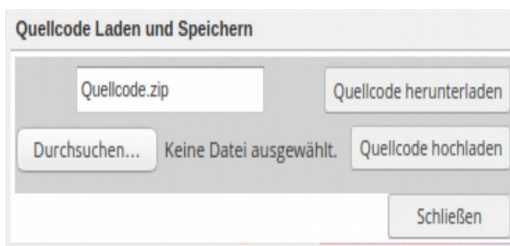


Abbildung 2: Das Download/Upload-Fenster von VipLab

An der Stelle kann der Nutzer auswählen, ob er einen Quellcode hochladen oder herunterladen möchte. Wenn der Nutzer einen Quellcode hochladen möchte, dann muss er mit dem Button „Durchsuchen“ noch den richtigen Pfad zur Datei auswählen. Wenn der Nutzer jedoch den Quellcode aus der IDE herunterladen möchte, dann kann er im oberen rechten Fenster, einen Dateinamen vor dem

Download bestimmen. Im Beispiel ist der Dateiname „Quellcode.zip“. Mit der Betätigung des Buttons „Quellcode herunterladen“ wird die Datei im Download-Verzeichnis des jeweiligen Betriebssystems abgespeichert. Mit dem Button „Schließen“ kann wieder in die IDE zurückgekehrt werden.

3.) *Anzeige der Programmiersprache:* In der oberen rechten Ecke der Anwendung findet sich eine Information darüber, welche Programmiersprache gegenwärtig in der IDE eingegeben werden kann. Im Beispiel ist das die Sprache „C“.

4.) *Anzeige der Codevorgabe:* In dem oberen linken IDE-Feld ist der vorgegebene Code des Dozenten angezeigt. (ein Inhaltsbereich)

5.) *Eingabefeld der Lösung:* In dem unteren linken IDE-Feld kann die Lösung der Textaufgabe eingegeben werden. (ein Inhaltsbereich)

6.) *Anzeige der Visualisierung (Output-Bereich):* Die ganze rechte Bildschirmhälfte von VipLab ist dem Output der IDE gewidmet. Hier werden die Visualisierungen und die Rückmeldungen des Interpreters angezeigt. In der Anwendung kann mit Hilfe des Mauseisens in die Visualisierungen hinein- und herausgezoomt werden. (ein Inhaltsbereich)

7.) *Die Funktion „Zurücksetzen“:* Mit dem Button „Zurücksetzen“ kann eine Grafik wieder auf den Ursprungszustand zurückgesetzt werden. Damit ist gemeint, dass wenn man in eine Grafik beispielsweise zu tief hineingezoomt hat und wieder auf den Ursprungszustand zurückkehren möchte. Der Button kann auch als „Reset“ bezeichnet werden.

8.) *Die Funktion „Herausziehen“:* Mit dem Button „Herausziehen“ kann die Visualisierung auf den gesamten Bildschirm vergrößert werden.

9.) Die Funktion „Plot herunterladen“: Mit der Funktion „Plot herunterladen“ kann die gegenwärtige



Abbildung 3: Der Download einer Grafik in VipLab

Visualisierung als eine „Portable Network Graphic“ heruntergeladen werden. Bei der Betätigung der Schaltfläche öffnet sich in der Anwendung ein zusätzliches kleines Fenster, indem der Download bestätigt werden muss. Der Download wird im Download-Verzeichnis des jeweiligen Betriebssystems abgespeichert. Es besteht auch die Möglichkeit den

Download mit der Schaltfläche „Abbrechen“ zu beenden.

10.) Die Funktion „Berechnung starten“: Mit der Funktion „Berechnung starten“ wird der linke Code



Abbildung 4: Der Berechnungsprozess in VipLab

in der IDE ausgeführt und in der rechten Bildschirmhälfte dargestellt. Solange der Code für die Visualisierung berechnet wird, sieht man in der Anwendung einen Ladebalken. Die Berechnung kann an der Stelle jederzeit abgebrochen

werden.

11.) Die Funktionen „Vorwärts“, „Rückwärts“ und „Play“: Diese Funktionen werden dazu verwendet, um zwischen mehreren Visualisierungen, die der Code erzeugt hat, hin- und herzuschalten. Mit dem Play-Button werden mehrere Visualisierungen hintereinander abgespielt. Auf der gleichen Höhe zu den Buttons befindet sich noch auf der rechten Seite die Anzeige, die wiedergibt, wie viele Visualisierungen insgesamt erzeugt wurden (im Beispiel zwei Grafiken).

In anderen Berechnungen für Visualisierungen werden unter Umständen noch weitere Funktionen im Output-Bereich mit ausgegeben. Die folgenden beiden Punkte erscheinen abhängig davon, welche Berechnung ausgeführt wurde.

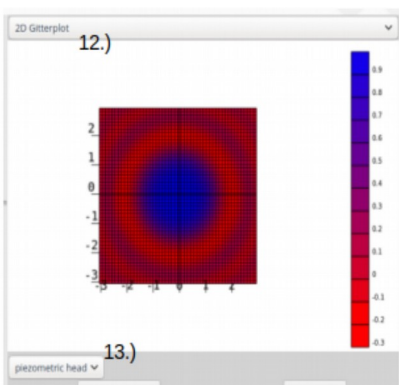


Abbildung 5: Weitere Use Cases in VipLab nach der Berechnung

12.) Die Funktion „Grafik 2D/3D umschalten“: In diesem Feld kann die erzeugte Visualisierung zwischen einer 2D und 3D Ansicht hin und her geschaltet werden.

13.) Die Funktion „Grafik umschalten“: Mit dieser Funktion kann zwischen mehreren Visualisierungen hin- und hergeschaltet werden. Der Code muss hierfür vorab auch mehrere Visualisierungen berechnet haben.

## 2.4 Die Normen zur User Experience einer Anwendung

Für die Entwicklung einer Anwendung wurden verschiedene ISO Normen festgelegt, um einen standardisierten Leitfaden bereitzustellen. Zum einen gibt es Normen, die die Anwendung an sich betreffen, und andere Normen, die den Entwicklungsprozess standardisieren. Normen sind recht allgemein formulierte Leitsätze, die über einen möglichst langen Zeitraum ihre Gültigkeit behalten sollen, um erfolgreiche Anwendungen hervorzubringen. Normen geben immer nur einen groben Anhaltspunkt dafür, was in einer Anwendung hauptsächlich unbedingt mit gegeben sein sollte. Auf diese standardisierten Inhaltspunkte haben sich viele Experten aus der Branche geeinigt. Damit können die Standards gut dafür verwendet werden, um die eigene Arbeit gegenüber dritten zu verteidigen.<sup>8</sup>

Für die hier vorliegende Bachelorarbeit ist die ISO Norm 9241-151 „Leitlinien zur Gestaltung von Benutzungsschnittstellen für das World Wide Web“ besonders nennenswert. Diese Norm zielt auf die zentralen Eigenschaften einer Anwendung ab. Die Anwendung VipLab wird über einen Server der Universität Stuttgart bereitgestellt. Daher ist die Berücksichtigung dieser Norm besonders wichtig. In der Norm werden die Eigenschaften aufgeführt, die für die Entwicklung einer benutzerfreundlichen Anwendung für das Word Wide Web erforderlich sind.

In dem nachfolgenden Text werden die Norm-Empfehlungen anhand des ermittelten Ist-Zustandes abgeglichen, um festzustellen, ob es tatsächlich starke Abweichungen von der Norm gibt. Sollten stärkere Abweichungen festgestellt werden, dann wird damit nur die Dringlichkeit einer ganzheitlichen Überarbeitung verdeutlicht.

*Der Aufgabe angemessen:* Eine Anwendung sollte immer das leisten können, was man von ihr erwartet. Dabei soll die Anwendung den Nutzer dabei unterstützen, sodass dieser seine Absichten und Ziele auf der Plattform schnell erreicht. Die Anwendung ermittelt auch nur so viele personenbezogene Daten wie für den Use Case erforderlich sind.

VipLab erfüllt die Anforderung schon teilweise zufriedenstellend. In der Anwendung werden bereits Übungsaufgaben mit Studierenden zuverlässig durchgeführt. Jedoch erreicht der Nutzer nur seine Absichten in der Anwendung schnell, wenn er diese schon im Vorfeld kennengelernt hat. Der Icon-Einsatz ist beispielsweise noch sehr schwammig und erschwert deutlich die Usability. Das Disketten-Icon ist in Bezug auf die Funktionalität missverständlich und es sind viele Buttons nur mit Labels anstatt mit geeigneten Icons gestaltet.

---

8 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 57

Die Usability ist zudem in der Anwendung noch deutlich eingeschränkt, weil zu viele Klicks benötigt werden, um eine Absicht zu erreichen. Beispielsweise der Download von einem Quellcode. Erst durch die Download-Bestätigung in einem zweiten zusätzlichen Fenster wird die Funktion ausgeführt.

Das gesamte Erscheinungsbild vermittelt zudem nicht angemessen den Sinn und Zweck der Anwendung. Die verschiedenen Ein- und Ausgabebereiche (Inhaltsbereiche) sind nicht näher definiert und beschrieben. Die Zugänglichkeit der Anwendung ist deutlich erschwert.

*Selbstbeschreibend:* Die Anwendung macht deutlich, wie der Benutzer seine Absichten erreicht und welche einzelnen Schritte er dazu machen muss. Eine klare Navigation und verständliche Tooltips sind an den verschiedenen Stellen dabei Voraussetzung.

VipLab erfüllt die Anforderung nicht. Die Anwendung liefert der Nutzerschaft keine klaren Anweisungen, wie auf der Oberfläche vorgegangen werden muss, um eine Absicht zu erreichen. Man erkennt den Sinn und Zweck der Anwendung nicht und die verschiedenen Inhaltsbereiche können nicht sofort einer logischen Funktion zugewiesen werden. Es werden keine Einzelschritte oder eine Navigation für die Zielerreichung erläutert. Eine Zielerreichung kann an manchen Stellen in der Anwendung mit einer Button-Hierarchie begünstigt werden. Zudem sind keine Tooltips in der gesamten Anwendung gegeben.

*Steuerbar:* Der Nutzer hat die volle Kontrolle über die Anwendung. Es ist nicht umgekehrt, dass die Anwendung den Nutzer in eine Richtung steuert. Beispielsweise kann der Anwender Animationen oder Grafiken unterbrechen und neu laden, wenn er das möchte. Bei Videos kann er die Lautstärke regulieren oder abstellen. Auf einem Screen kann der Nutzer auch immer einen Schritt wieder zurückgehen.

VipLab erfüllt diese Anforderung. Grafiken, Ladevorgänge oder Animationen können abgebrochen werden. Es gibt immer ein Weg zurück und die Anwendung versucht nicht den Nutzer zu steuern.

*Erwartungskonform:* Die Anwendung überrascht den Nutzer nicht mit unerwarteten Inhalten. Der Nutzer bekommt immer das, was er auch von der Anwendung erwartet. Es ist daher wichtig, dass innerhalb der Anwendung eine Konsistenz aufrechterhalten wird. Damit ist auch die Einhaltung weitverbreiteter etablierter Konventionen gemeint. Beispielsweise sind Links immer unterstrichen dargestellt und Icons sind international aussagekräftig gewählt.

VipLab erfüllt die Anforderung nicht. Auf aussagekräftige Icons wird generell in der gesamten Anwendung verzichtet. Stattdessen sind die Buttons nur mit Labels beschriftet. Dadurch wird die User Experience in der Anwendung deutlich geschwächt. Der Nutzer kann auch von VipLab verwirrt werden, weil das Aussehen im Vergleich zu anderen Webseiten, durch das Fehlen von Icons, gänzlich anders erscheint. Außerdem werden internationale Gestaltungskonventionen nicht eingehalten, wie beispielsweise die symmetrische Gruppierung von Elementen in verschiedenen Inhaltsbereichen. Beispielsweise steht der Button „Berechnung starten“ viel zu weit rechts.

*Fehlertolerant:* Die Anwendung sollte fehlertolerant sein, wenn ein Benutzer eine Falscheingabe tätigt. Bei Fehlern sollte die Anwendung immer klare Rückmeldung geben können. Der Nutzer sollte immer wissen, was die Anwendung im Hintergrund ausführt und dazu Rückmeldung geben können. Der Korrekturaufwand sollte für den Nutzer maximal gering sein.

VipLab erfüllt die Anforderung nur bedingt. Die Fehlermeldungen sind für den Nutzer oft unbrauchbar formuliert. Beispielsweise werden in den Fehlermeldungen oft aus der Informatik sehr technische Begriffe verwendet, die der Nutzer nicht versteht.

Zudem gibt VipLab in verschiedenen Situationen keine klare Rückmeldung, was im Hintergrund der Anwendung ausgeführt wird. Beispielsweise, wenn eine Aufgabe von dem Nutzer zur Bewertung eingereicht wird.

*Individualisierbar:* Der Benutzer sollte in der Anwendung die Möglichkeit haben, die Umgebung auf seine Vorlieben anzupassen. Die Anwendung merkt sich u. a. auch die benötigten Daten vom Nutzer, die für die Ausführung, beispielsweise die Anmeldung, benötigt werden.

Die Anwendung erfüllt die Anforderung nicht. Die Anwendung ist nicht individualisierbar für den Nutzer.

*Lernförderlich:* Die Anwendung sollte den Nutzer dabei unterstützen, den Umgang mit ihr zu erlernen.

Die Anwendung erfüllt die Anforderung nicht. Es sind derzeit keine Hilfsfunktionen implementiert, die den Nutzer in die Bedienung unterstützen. Beispielsweise können an der Stelle Tooltips eingesetzt werden. Tooltips erhöhen die Lernbereitschaft mit einer Anwendung.<sup>9</sup>

---

9 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 59-60

Der Vergleich mit der Norm zeigt letztendlich, dass die Anwendung VipLab noch stark in einigen Punkten Nachholbedarf aufweist. Im nächsten Kapitel wird der Entwicklungsprozess näher erläutert, der benötigt wird, um VipLab näher an die Norm anzupassen. Damit werden dann auch folglich die identifizierten Usability-Probleme aus dem Ist-Zustand von VipLab behoben. Es wird die Norm vorgestellt, die den Entwicklungsprozess nach bestimmten Richtlinien standardisiert und es werden die einzelnen Projektphasen mit den benötigten Methoden für das Projekt vorgestellt.

### **3 Der Projektablauf**

#### **3.1 Die Normen zum Entwicklungsprozess einer Anwendung**

Die ISO Norm 9241-151 beschreibt, welche Eigenschaften eine Anwendung haben sollte. Es gibt aber auch eine Norm, die den Entwicklungsprozess beschreibt, indem idealerweise gebrauchstaugliche interaktive Systeme entwickelt werden können. Die ISO Norm 9241-210 beschreibt den „Prozess zur Gestaltung gebrauchstauglicher interaktiver Systeme“ und formuliert einige Empfehlungen. In der Norm ist von einer „menschenzentrierten Gestaltung“ die Rede. Ein wichtiger Aspekt ist also, dass der Nutzer in den Fokus der Entwicklung gerückt wird. In der Norm ist diese Anforderlichkeit deutlich herauszulesen. Des Weiteren empfiehlt die Norm ausdrücklich, dass der Projektablauf agil gestaltet sein sollte. Die Kernpunkte der Norm sind folgende:

*1.) Die Entwicklungsarbeit basiert auf der Annahme, dass man den Nutzer mit seinen Aufgaben und dessen Umgebung kennt, in der die Anwendung zum Einsatz kommt.*

Vor jeder Konzeptionsphase wird in der Analysephase vorausgesetzt, dass der Nutzer erst genauer analysiert und beschrieben wird. Es ist klar, welche Aufgaben/Absichten er hauptsächlich in der Anwendung verfolgt und man kennt seine Interessen und Bedürfnisse im Kontext zur Anwendung. Letztlich kennt man sogar seine Umgebung, in der die Anwendung meistens angewandt wird. Aufgrund dieser Erkenntnisse werden zur Übersichtlichkeit Personas zu der Nutzerschaft erstellt.

*2.) Die Nutzer werden aktiv mit in den Entwicklungsprozess integriert.*

In jeder Projektphase ist es zulässig und auch erwünscht, dass die erzeugten Projektergebnisse mit den Nutzern auf ihre Brauchbarkeit immer wieder validiert werden. Dazu können Usability-Tests mit statischen Mockups oder interaktive Prototypen herangezogen werden. Das ist immer davon abhängig, welche Resultate in der gegenwärtigen Projektphase vorliegen.

### *3.) Die Anwendungsentwicklung wird durch Nutzertests getrieben und gesteuert.*

Mit Usability-Tests werden die erzeugten Projektergebnisse aus der Konzeptionsphase mit den Nutzern validiert. Sollte dem Nutzer der Prototyp nicht gefallen, dann ist es erforderlich, diesen nochmals mit den neu gewonnenen Erkenntnissen zu überarbeiten. Die Testergebnisse haben schließlich ergeben, dass der Prototyp noch nicht in seiner jetzigen Form funktioniert oder gefällt.

### *4.) Die Anwendungsentwicklung ist iterativ.*

Es ist nicht ausreichend nur einen Usability-Test durchzuführen. Während aller Projektphasen innerhalb eines Projektphasen-Zykluses sollten immer wieder Tests mit den Nutzern durchgeführt werden, um den Prototyp oder das Mockup weiter zu verbessern. Ist ein Projektphasen-Zyklus aus Analyse-, Konzeptions-, Umsetzungs- und Testingphase durchlaufen, dann wird dieser Kreislauf ebenso lange wiederholt, bis das Ergebnis den Nutzern gänzlich gefällt.

### *5.) Die Anwendungsentwicklung umfasst die gesamte User Experience.*

Es soll in der Konzeptionsphase nicht nur die Usability (Funktionalität) berücksichtigt werden. Die Anwendung sollte ganzheitlich eine runde, zufriedenstellende Erfahrung für den Nutzer sein. Nach dieser Prämisse sollte der gesamte Entwicklungsprozess ausgerichtet werden.

### *6.) Im Entwicklungsteam sind verschiedene Perspektiven und Fähigkeiten repräsentiert.*

In der Entwicklungsarbeit sollen nicht nur Informationsdesigner vertreten sein. Es sollte immer ein interdisziplinäres Team zusammengestellt werden, um möglichst viele verschiedene Meinungen und Sichtweisen in das Projekt mit einzubringen.<sup>10</sup>

Zusammenfassend kann also gesagt werden, dass die Entwicklungsarbeit einerseits nutzerzentriert sein sollte, wie auch agil. Das sind wichtige Voraussetzungen. Im nachfolgenden Text wird die nutzerzentrierte Entwicklung wie auch das iterative agile Projektmanagement nochmals näher beleuchtet. Beide Themen sind für die vorliegende Bachelorarbeit von zentraler Bedeutung. Gegen Ende der Bachelorarbeit werden Usability-Tests mit den ausgearbeiteten Prototypen durchgeführt. Die nutzerzentrierte Anwendungsentwicklung wird somit nach der

---

<sup>10</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 60-61



empfohlenen Norm eingehalten, wenn auch nur in der Testingphase und in keinen anderen Phase evaluiert wird.

Auch nach der Bachelorarbeit werden die entwickelten Prototypen noch nicht vollständig ausgereift sein. Deshalb ist es wichtig, dass das Ergebnis der Bachelorarbeit fortlaufend mit den Nutzern von VipLab immer weiterentwickelt wird. Die definierten Projektphasen werden abermals wiederholt. Eine Einhaltung der Norm, iterativ zu arbeiten, ist für den Erfolg der Anwendung essenziell.

Eine theoretische Ausarbeitung beider Themen kann dazu beitragen, dass das Vorgehen im Projektablauf verstanden wird. In nachfolgenden Bachelorarbeiten kann auf dieses Wissen zurückgegriffen werden, um weiterhin nach international anerkannten Konventionen zu arbeiten.

### **3.2 Die nutzerzentrierte Entwicklung**

Die ISO Norm 9241-210 sieht also ausdrücklich vor, dass der Nutzer in den Entwicklungsprozess aktiv mit eingebunden wird. An dieser Stelle spricht man auch von „nutzerzentrierte Entwicklung“.

Über den Erfolg einer Anwendung entscheiden letztlich die Nutzer, die mit der Anwendung täglich arbeiten. Ist die Anwendung zu weit von Nutzerbedürfnissen entfernt, dann wird die Entwicklung mit hoher Wahrscheinlichkeit scheitern. Eine gut konzipierte Anwendung, die den Nutzer und dessen Bedürfnisse berücksichtigt, hat nicht nur das Potenzial zu befriedigen, sondern auch eine emotionale Bindung zwischen Nutzer und Anwendung zu ermöglichen. Wenn der Nutzer in den Entwicklungsprozess mit eingebunden wird, dann ist die Wahrscheinlichkeit deutlich höher, dass ihm das Projektergebnis gefällt und er die Anwendung in sein Arbeitsumfeld leichter mit integriert. Anderenfalls hat man viel Arbeitszeit und viel Geld für die Entwicklung in den Sand gesetzt, wenn das Endergebnis nicht angenommen wird.

Es ist wichtig, den Nutzer von Anfang bis Ende in allen Projektphasen einer Anwendungsentwicklung mit einzubeziehen. Wie bei der oben erwähnten Norm ist es wichtig die Ziele, Aufgaben und Eigenschaften der Nutzer zu kennen. Der gesamte Entwicklungsprozess liegt diesen Informationen zugrunde. Die nutzerzentrierte Entwicklung wird angestrebt, um nicht basierend auf eigenen Vermutungen eine Anwendung zu entwickeln. Die bloßen Vermutungen eines Entwicklerteams über die Nutzerschaft können schnell in die Irre führen und in Fehlentwicklungen enden. Es ist wichtig, dass empirische Daten in der Analysephase zu der Nutzerschaft gesammelt werden und diese den reinen Vermutungen gegenübergestellt werden, um eine realitätsnahe Gegebenheit zu beschreiben. Mit empirischen Daten zu den

Nutzungsbedürfnissen ist die Wahrscheinlichkeit wesentlich höher, dass die Projektarbeit zu einem Erfolg geführt wird.<sup>11</sup>

### **3.3 Agil bis ans Ziel: Der agile Projektablauf**

#### **3.3.1 Die statische Methode**

In den Anfangsphasen der Programmierung wurde nach dem Wasserfallprinzip in Projekten gearbeitet. Damit ist gemeint, dass im Team erst genau definiert wurde, was das zu erreichende Projektziel ist. Davon ausgehend wurde eine präzise Planung erstellt, die man schrittweise abarbeitete, bis das Projekt abgeschlossen wurde. In der Softwareentwicklung hat sich diese Methode heute nicht mehr durchsetzen können. Durch das statische Projektmanagement sind sehr häufig die Projekte in verschiedenen Situationen gescheitert. Zum Beispiel hat ein Team autonom an einem Projektbaustein wie dem Front-End gearbeitet und ein anderes am Back-End, bis das Projekt abgeschlossen wurde. Am Projektende hat man dann versucht beide Teile zusammenzuführen, was häufig gescheitert ist, weil keine Zwischenergebnisse unter den Teams während der Projektarbeit kommuniziert wurden. Man konnte also nicht schon während der Entwicklung feststellen, dass zwischen Front-End und Back-End technische Problematiken entstehen würden. Dem gegenüber steht heute die agile Projektmanagementmethode, die mehr Erfolg für den Projektverlauf verspricht.<sup>12</sup>

#### **3.3.2 Die agile Methode**

Agil kann aus dem Lateinischen mit flink oder beweglich ins Deutsche übersetzt werden. Bei dieser Vorgehensweise versucht man im Gegensatz zu der statischen Projektmanagementmethode wenig vorzuplanen und möglichst schnell in den Teams (Front-End vs. Back-End) erste Ergebnisse zu erzielen, die wiederum validiert und schon während der Projektarbeit kommuniziert und verglichen werden können. Dabei arbeitet man folglich iterativ. Im Vergleich zu der statischen Methode werden die Projektphasen bis zum fertigen Endprodukt mehrmalig wiederholt und nicht nur einmalig abgearbeitet. Mit der agilen Methode möchte man somit schon in frühen Projektphasen erkennen, ob man auf dem richtigen Nenner zwischen den Teams ist. Damit ist beispielsweise auch gemeint, dass alle Projektbeteiligten die gleiche Vorstellung teilen, wie das fertige Endprodukt einmal aussehen kann und, dass alle Entwicklungskomponenten technisch problemlos zusammengeführt werden können. Mit der agilen Projektmanagementmethode kann

---

11 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 83

12 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 89-90

Missverständnissen in der Projektarbeit mit Kollegen und Nutzern frühzeitig vorgebeugt werden, und die Wahrscheinlichkeit ist höher, dass das Projekt zum Erfolg geführt wird.<sup>13</sup>

In der vorliegenden Bachelorarbeit wurde genau nach diesem agilen Prinzip gearbeitet, um Missverständnissen in der Entwicklungsarbeit vorzubeugen. In jeder Projektphase wurde alle 14 Tage mit allen Projektverantwortlichen ein Gespräch geführt, um die Projektarbeit (das Wireframe und den Prototyp) miteinander abzugleichen. Dabei teilte der Projektmanager mit, ob der Fortschritt zufriedenstellend ist und ob man noch den gleichen Nenner in der Entwicklung teilt. In den Projektgesprächen wurden auch immer wieder Verbesserungen und entscheidende Entwicklungsschritte für die Projektarbeit zusätzlich vorgeschlagen. Diese wurden in das Projekt mit aufgenommen und realisiert, sodass das Endprodukt für alle Beteiligten angemessen ist und mit den Probanden getestet werden kann.

### **3.4 Die einzelnen agilen Projektphasen für das Projekt VipLab**

Die Projektphasen für VipLab sind nutzerzentriert ausgerichtet. Es wird aber nicht in jeder Projektphase (Analyse-, Konzeptions-, und Umsetzungsphase) der Nutzer immer gleichermaßen in die Entwicklungsarbeit mit einbezogen. Der Nutzer von VipLab wird in den verschiedenen Projektphasen unterschiedlich stark mit eingebunden. Der nutzerzentrierte Ansatz kommt in der Testingphase der Bachelorarbeit am stärksten zu Tragen. Die Gründe dafür sind in den Projektgegebenheiten zu finden.

Die Anwendung VipLab wird gegenwärtig über einen Universitätsserver mit allen nötigen Funktionen (Use Cases) bereitgestellt. Die Studenten und Forscher können auf den Service jederzeit kostenfrei zugreifen. Da die relevantesten Use Cases bereits ausgeführt werden, wird in der Bachelorarbeit die erste Projektphase (die Analysephase), in der der Nutzungskontext analysiert, beschrieben und verstanden wird, ausgelassen. Damit werden als Methoden Befragungen oder Fokusgruppen - Interviews außen vor gelassen, um für das Projekt zentrale Use Cases zu eruieren, die zu mehr User Experience in der Anwendung führen können. Die wichtigsten Use Cases sind somit schon für die Projektarbeit vorgegeben. Einige von diesen Use Cases bringen auch schon eine deutliche User Experience mit sich, was für den Soll-Zustand mit berücksichtigt werden kann. Mit diesen Use Case wird anfänglich in der Konzeptionsphase primär gearbeitet und das erste Wireframe wird danach ausgerichtet.

---

<sup>13</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 90-91

Ausschließlich Personas werden für die Analysephase angefertigt, um die zwei wichtigsten Nutzerrollen für VipLab darzustellen. Mit den Personas soll aufgezeigt werden, für welche Zielgruppen man die Projektarbeit hauptsächlich durchführt. Da keine Analysephase in der Projektarbeit stattfindet, ist für die Persona - Beschreibung keine wirkliche empirische Datengrundlage vorhanden. In der Testingphase, in der die Usability-Tests durchgeführt werden, ist deshalb vorgesehen, dass zusätzlich im Testfragebogen der Nutzungskontext nachträglich für die Personas mit erhoben wird. Aufgrund dieser Daten werden dementsprechend nachträglich die Personas zu den Studenten und Forschenden erstellt. Es wird damit eine Nutzerbefragung aus der Analysephase nachgeholt, um Bedürfnisse, Interessen oder Beschwerden mit VipLab zu eruieren. Ein weiterer Vorteil dieser Daten ist, dass zusätzliche zentrale Funktionen (Use Cases) für die User Experience daraus abgeleitet werden können, die in den überarbeiteten Soll-Zustand nach den Usability-Tests mit implementiert werden.

Durch die vorgegebenen Use Cases aus dem Ist-Zustand wird die Konzeptionsphase stärker ausgeprägt sein. Die vorgegebenen Nutzungsanforderungen (Use Cases) werden in der Konzeptionsphase in Use Case Diagrammen und Wireframes weiter spezifiziert. In den Wireframes wird erstmals ersichtlich, wie die zentralen Use Cases in VipLab neu angeordnet und dargestellt werden können. Die Seitenstruktur erhält dabei auch erste deutliche Züge und kann diskutiert werden. Diese erste Gestaltung ist aber noch sehr ungenau und nicht fertig ausgereift.

In der dritten Projektphase (der Design- und Umsetzungsphase) wird das finale Wireframe im Laufe der Projektarbeit zu einem High Fidelity Prototyp weiterentwickelt, um den Detaillierungsgrad in der Gestaltung zu erhöhen. Der Prototyp ist dabei schon interaktiv in der Benutzung.

In der vierten Testingphase rückt wieder die nutzerzentrierte Entwicklung zurück in den Fokus der Projektarbeit. Der Schwerpunkt der nutzerzentrierten Entwicklung liegt in der vorliegenden Bachelorarbeit primär bei den Usability-Tests, die am Ende stattfinden. Mit den High Fidelity Prototypen, die in der Umsetzungsphase erstellt wurden, werden dann folglich auch die Usability-Tests mit den Nutzern durchgeführt. In den Usability-Tests werden die Prototypen auf ihre Funktionstüchtigkeit (die Usability) und die User Experience getestet und bewertet. Als Instrument zur Erhebung wird an der Stelle parallel zu den Tests ein qualitativer Fragebogen herangezogen.

Das hier vorgestellte Studiendesign wurde auf der Grundlage von dem Praxishandbuch „Usability und UX“ von Jens Jacobsen und Lorena Meyer erstellt. In dem Buch wird u. a. erläutert, wie für ein individuelles Designprojekt die richtigen wissenschaftlichen Methoden ausgewählt werden können,

um die Projektphasen danach zu planen. Des Weiteren wird in dem Buch die wichtigste Theorie zu UX und UI vermittelt, um damit die theoretische Ausarbeitung bis zum Prototyp zu untermauern.

Ich habe mich für das Buch entschieden, weil es auch für einen Anfänger gut verständlich ist und es inhaltlich alle relevanten Themen abdeckt, damit ein Designprojekt adäquat gelingen kann. Die Methodik sowie die Theorie ist in dem Buch ausgeglichen beschrieben. Damit ist es auch für die Bachelorarbeit möglich geworden, nur mit einer hauptsächlichen Quelle in dem Projekt zu arbeiten. Mit dem Buch ist ein guter Leitfaden für die tägliche Arbeit eingeflossen.

Der geplante Projektablauf tabellarisch dargestellt:

Phase	Schritte/Aktionen	Ausgewählte Methoden
<i>Die Analyse:</i> Nutzungskontext verstehen und beschreiben	Beschreibung der Nutzer mit den Bedürfnissen und Zielen.	<ul style="list-style-type: none"><li>• Personas</li></ul>
<i>Die Konzeption:</i> Nutzungsanforderungen spezifizieren	Darstellung der Nutzerbedürfnisse	<ul style="list-style-type: none"><li>• Use Case Diagramme</li><li>• Wireframes</li></ul>
<i>Die Umsetzung (Design):</i> Eine Gestaltungslösung entwickeln, die die Nutzungsanforderungen erfüllt.	Interaktions-spezifikation erstellen	<ul style="list-style-type: none"><li>• Mockups und Prototyp</li></ul>
<i>Das Testing:</i> Die Gestaltungslösung aus Nutzerperspektive evaluieren.	Gestaltungslösung wird getestet mittels der Anforderungen.	<ul style="list-style-type: none"><li>• Usability-Tests</li><li>• qualitativer Fragebogen (bezogen auf die Usability-Tests)</li></ul>

*Tabelle 1: Der Projektablauf mit den Methoden*

Dieser vierphasige Projektablauf (siehe *Tabelle 1*) wird so oft wiederholt, bis die Nutzer mit den Prototypen zufrieden sind. Da der Projektablauf ein iterativer ist, spricht man auch von einer agilen Vorgehensweise. Im nächsten Kapitel wird theoretisch beschrieben, wie die Personas für die Projektarbeit erstellt werden können. Damit wird nur eine Methode für die Analyse angewandt.

## **4 Die Analysephase**

### **4.1 Die Theorie zu Personas**

Um eine nutzerzentrierte Anwendung entwickeln zu können, ist es erforderlich, dass die ermittelten Nutzeranforderungen während der gesamten Entwicklungsarbeit nicht aus dem Fokus verloren werden. Damit ist gemeint, dass sich das Entwicklerteam in der täglichen Arbeit immer nach den

Nöten und Bedürfnissen der Nutzer ausrichtet und demnach entwickelt. Es wird somit ein Instrument benötigt, um die zentralen Nutzeranforderungen immer vor Augen zu haben. Die Nutzungsanforderungen können unter anderem in der Analysephase ermittelt wurden sein oder in späteren Phasen des Projekts. In der gegenwärtigen Bachelorarbeit werden die Nutzungsanforderungen (die Datengrundlage) rückwirkend parallel zu den Usability-Tests erhoben.

Ein gutes Instrument um alle Nutzungsanforderungen zusammenzufassen, ist es Personas zu definieren, die alle konkreten Bedürfnisse festhalten. Eine Persona soll im gesamten Projektteam ein klares Bild vom typischen Nutzer der Anwendung erzeugen. Die Personas begleiten dann die Entwicklungsarbeit in jeder Projektphase und garantieren, dass keine falschen Vermutungen mehr zu den Nutzern angestellt werden und in die Projektarbeit mit einfließen. Die meisten Vermutungen, die ein Entwicklerteam über die Nutzungsbedürfnisse ihrer Nutzer anstellen kann, sind meistens falsch und weichen von der wirklichen Realität stark ab. Dann besteht die Gefahr, dass die Anwendung anhand falscher Annahmen in eine falsche Richtung entwickelt wird. Aus dem Grund ist es ratsam, für die nutzerzentrierte Entwicklung Personas mit einzusetzen, die auf empirischen Daten beruhen.

Personas sind prototypische Nutzerprofile, in denen die Nutzergruppen und ihre unterschiedlichen Ziele, Eigenschaften, Verhaltensweisen und Motive in Bezug auf eine Anwendung genau beschrieben werden. Personas helfen, die Bedürfnisse und Anforderungen der Nutzer zu verstehen und die Anwendung speziell für diese zu entwickeln. Ein weiterer Vorteil von Personas ist, dass sie auch gut zur Kommunikation in interdisziplinären Teams herangezogen werden können, um Konsens zwischen allen Beteiligten zu schaffen. Eine Persona dient dazu, sich einen schnellen Überblick über eine Nutzergruppe zu verschaffen, sodass jedes Teammitglied den gleichen Informationsstand teilt.

#### **4.2 Die Erstellung einer Persona**

Für eine relevante Nutzergruppe in einer späteren Anwendung wird stellvertretend immer eine Persona erstellt. Eine Anwendung wird aber selten nur von einer Nutzergruppe verwendet. Oft sind es mehrere verschiedene Nutzergruppen die für eine Anwendung identifiziert werden müssen. Wird eine Anwendung von verschiedenen Nutzern verwendet, dann werden auch folglich mehrere Personas dazu erstellt.

Für die gegenwärtige Projektarbeit VipLab müssen die Nutzergruppen nicht erst aufwendig über eine breit angelegte Analysephase identifiziert werden. Für VipLab standen schon vor

Projektbeginn die primären Nutzergruppen fest. Einerseits benutzen viele Studenten VipLab, um im Rahmen ihres Studiums, in einer IDE Programmiersprachen zu erlernen. Andererseits gibt es in VipLab noch den Forscher als Nutzergruppe, der die Anwendung als Rechercheinstrument zukünftig benutzen wird. Aufgrund dessen werden in der Projektarbeit genau zwei Personas erstellt. Diese sollen eine weitere Orientierung im Projekt ermöglichen, um die Anwendung nutzerzentriert zu entwickeln.

In der Erstellung von Personas hat sich eine kompakte Darstellungsform bewährt. Eine Persona sollte nie zu textlastig formuliert sein. In der Praxis hat sich das nicht bewährt. Alle relevanten Informationen werden auf einer DIN4 Seite übersichtlich zusammengefasst. Dabei wird ein Foto des typischen Nutzers in der rechten oberen Ecke angeheftet. Ein Bild kann die emotionale Bindung des Projektteams zu Persona steigern. Das kann man auch erreichen, wenn ein Lebensmotto zu der Persona formuliert wird. In Stichpunkten werden des Weiteren die wesentlichen Eckpunkte (Bedürfnisse und Ziele) des Nutzers zusammengetragen. Je nach Projekt können unterschiedliche Informationen zu einer Nutzergruppe auf der Persona zusammengefasst sein.

- soziodemografische Angaben (Name, Alter, Geschlecht und Arbeitsplatz)
- persönliche Eigenschaften und Merkmale des Nutzers
- sein Verhaltensmuster, seine Zielsetzung / Intention und Aufgaben bei der Nutzung der Anwendungen
- seine Vorerfahrung und Kenntnisse hinsichtlich der betreffenden Thematik
- grundlegende Einstellungen
- Angaben zu seiner Umgebung mit Einfluss auf sein Verhalten

Gängig ist es immer als Erstes, soziodemografische Angaben auf der Persona festzuhalten. Zusätzlich können beispielsweise persönliche Eigenschaften hinzugefügt werden, ob der Nutzer eher rationaler oder irrationaler eingestellt ist. Es kann aber auch die Technikakzeptanz oder -kompetenz in einer Grafik je nach Ausprägung dargestellt werden. Der wichtigste Inhalt einer Persona ist aber, dass die Ziele, Bedürfnisse, Interessen und Hindernisse mit der Anwendung zentral herausgestellt sind. Aufgrund dessen werden die wichtigen Use Cases für eine Anwendung ausgewählt, die unbedingt mit implementiert werden müssen, um für den Nutzer eine deutliche User Experience zu schaffen.<sup>14</sup>

---

14 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 113-116

Welche Informationen schließlich in einer Persona beschrieben werden, ist abhängig davon, was für eine Anwendung oder Produkt genau designet wird. Bei einer Autoapp ist beispielsweise die Technikakzeptanz als Parameter wichtig. In einer anderen Entwicklung ist dieser Parameter wiederum unwichtig. Es werden immer nur die Informationen in einer Persona beschrieben, die auch für die Entwicklungsarbeit eine hohe Relevanz haben. Die Personas können sich so je nach Projekt in ihrem Informationsgehalt stark voneinander unterscheiden. Welche Informationen für die Rollen Forscher und Student festgelegt werden, wird erst anhand der Auswertungen nach den Usability-Tests festzustellen sein. Die erstellten Personas werden aufgrund der späteren Datengrundlage in der Bachelorarbeit im Auswertungsteil zu den Usability-Tests mit aufgeführt.

#### **4.3 Die Datengrundlage für die Personas**

Personas sollten immer basierend auf einer empirischen Datengrundlage erstellt werden. In der Projektarbeit fand jedoch keine umfangreiche Analysephase statt, somit wurde auch keine Datengrundlage gesammelt, auf der aufbauend die Personas definiert werden können. Um Personas sachgerecht erstellen zu können, sammelt man im ersten Schritt so viele Informationen wie nur möglich über die Nutzer, die die Anwendung gebrauchen. Ein wichtiges Kriterium ist dabei, dass die Daten am besten empirisch erhoben sind. Die Datengrundlage kann dann im besten Fall aus empirischen Befragungen (z. B. qualitativ) oder Fokusgruppen - Interviews stammen.

Sollten keine Daten aus einem Erhebungsverfahren vorhanden sein, können auch vorhandene Informationen zu den Nutzern ausgewertet werden. Diese Daten können beispielsweise aus den Auswertungen des Kundenservices oder der Marktforschung entnommen werden. In diesen Abteilungen können schon viele Erfahrungen mit den Nutzern vorliegen, die für die Personas genutzt werden können. In Beschwerdemails aus dem Kundendienst kann zum Beispiel ersichtlich werden, welche bestehenden Use Cases oft dem Nutzer Schwierigkeiten bereiten und welche Bedürfnisse er tatsächlich an die Anwendung stellt. Die Beschwerdemails könne qualitativ in einer Dokumentenanalyse leicht ausgewertet werden.<sup>15</sup>

Im nächsten Kapitel wird die Konzeptarbeit für den Soll-Zustand von VipLab näher erläutert. Es wird aufgezeigt, wie die identifizierten Funktionen aus dem Ist-Zustand in eine überarbeitete Darstellungsform für den Soll-Zustand überführt und welche Methoden dafür eingesetzt werden. Diese Methoden beinhalten u. a. Use Case Diagramme, Wireframes und geeignete Icons.

---

15 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 117-118



## 5 Die Konzeptionsphase / der Soll – Zustand

### 5.1 Die Theorie zu Use Cases

Ein Use Case Diagramm ist Teil der Unified Modelling Language (UML). Bei einem Use Case wird eine Interaktion zwischen einem Akteur und einem System beschrieben. Ein Akteur bezeichnet dabei meistens den Benutzer einer Anwendung. Ein Akteur kann aber auch ein fremdes System sein, dass auf eine Anwendung von außen zugreift. Der Akteur ist immer derjenige, der den Use Case auslöst, um ein Ziel/eine Absicht in der Anwendung zu erreichen. Dabei beschreibt der Use Case, wie ein Akteur das geplante System konkret benutzen kann. Jeder Use Case ist eine abgeschlossene Handlung (Funktion) in sich, die ein sichtbares Resultat hervorbringt. Bei dem Projekt VipLab wäre beispielsweise ein typischer Use Case „Bilddatei herunterladen“. Bei der Ausführung der Funktion wird ein Bild, ausgehend von der Anwendung VipLab, auf den persönlichen Computer heruntergeladen und im Dateinennverzeichnis abgespeichert. Nachdem die Datei im Verzeichnis abgespeichert wurde, ist der Use Case damit abgeschlossen.

Ein Use Case beschreibt auf abstrakte Weise eine funktionale Anforderung an ein Programm, unabhängig davon, wie es technisch gelöst wird. Der Use Case „Bilddatei herunterladen“ impliziert beispielsweise nicht, wie die Funktion technisch umgesetzt werden kann. Hingegen kann die Use Case Benennung „Michael drückt auf den Button Bilddatei herunterladen“ die technische Lösung unnötig einschränken. Das für die Funktion unbedingt ein Button mit implementiert werden muss, ist nicht unbedingt die alleinige Option. Es kann auch andere Lösungsansätze geben.

In einer Use Case Beschreibung sind verschiedene Informationen zum Use Case Diagramm enthalten. Dabei kann der Detaillierungsgrad von Use Case Beschreibung zu Use Case Beschreibung unterschiedlich ausgearbeitet sein. Es gibt eine Kurzform für Use Case Beschreibungen, in denen nur in wenigen Sätzen der Funktionsablauf zusammengefasst wird. Meisten sieht man aber vollständig ausformulierte Use Case Beschreibungen, die alle wesentlichen Informationen zum Use Case enthalten. Unter anderem wird einerseits immer auch der Standardablauf für einen Use Case festgehalten, wie auch mögliche Alternativ- und Fehlerabläufe. Dabei sind die unten abgebildeten Informationen zudem relevant zu erfassen.

*Eindeutige Kennung:* Meisten wird „UC“ für Use Case und eine Nummer verwendet.

*Name:* Ein Objekt und eine Tätigkeit müssen enthalten sein. z.B. „Bilddatei herunterladen“

*Kurzbeschreibung:* Eine kurze Zusammenfassung des Funktionsablaufs.

*Beteiligte Akteure:* Die Akteure sind entweder Nutzer oder fremde Systeme von außen.

*Status:* Der Status signalisiert, ob der Use Case entweder in der Version eines Entwurfs, eines Reviews oder einer fertigen Ausarbeitung vorliegt.

*Auslöser:* Der Akteur, der den Use Case tatsächlich auslöst.

*Vorbedingung:* Die Vorbedingung beschreibt, welche Bedingungen erfüllt sein müssen, damit der Use Case ausgeführt werden kann.

*Standardablauf:* Der Standardablauf beschreibt den Use Case, wenn dieser fehlerfrei ausgeführt wird, sodass der Akteur sein Ziel/seine Absicht erreicht.

*Alternativ- und Fehlerabläufe:* Die Alternativ- und Fehlerabläufe beschreiben den Ablauf, wenn der Use Case nicht fehlerfrei ausgeführt wird und der Akteur nicht sein Ziel/seine Absicht erreicht.

*Nachbedingung:* Die Nachbedingung beschreibt den Systemzustand nach dem fehlerfreien Funktionsablauf.

*Änderungsgeschichte:* Die Änderungsgeschichte (Historie) enthält alle Änderungen am Use Case Diagramm und stellt die Nachvollziehbarkeit sicher.

Ein Use Case Diagramm zeigt auf einfache und übersichtliche Weise, was eine geplante Anwendung an Funktionalitäten beinhalten soll. Die Beziehungen zwischen den unterschiedlichen Use Cases werden zudem im Use Case Diagramm ersichtlich. Damit ist beispielsweise gemeint, wenn ein Use Case nur als Folge von einem andere Use Case aufgeführt wird. In dem Fall „inkludiert“ ein Use Case mit einem anderen.<sup>16</sup>

Die wichtigsten Use Cases für eine Anwendung sind immer die, die eine deutliche User Experience für den Nutzer mit sich bringen. Es werden also immer die wichtigsten zentralen Funktionen einer Anwendung als Use Case Diagramme dargestellt. Diese Funktionalitäten werden in der Analysephase mit dem Nutzer eruiert oder aus dem Ist-Zustand einer bestehenden Anwendung abgelesen. Wichtig ist, dass diese Use Cases unbedingt in der Konzeptionsphase in die grafische Entwicklungsarbeit mit einfließen und grafisch deutlich sichtbar gemacht werden.

In der grafischen Darstellung eines Use Case Diagramms werden die Akteure als Strichmännchen dargestellt und mit den entsprechenden Use Cases durch eine einfache Linie verbunden. Es wird dadurch erkenntlich, welcher Akteur welchen Use Case auslöst. Der Use Case an sich ist dabei in einer Ellipse dargestellt und in einem großen Rechteck eingefasst. Das Rechteck repräsentiert auf abstrakte Weise die gesamte Anwendung. Die Use Case Diagramme werden häufig immer vor den detaillierten Use Case Beschreibungen ausgearbeitet.<sup>17</sup>

---

16 Moser, C. (2012) .User Experience Design: Mit erlebniszentrierter Softwareentwicklung zu Produkten, die begeistern. Berlin Heidelberg: Springer-Verlag. Seite 92-93

17 Moser, C. (2012) .User Experience Design: Mit erlebniszentrierter Softwareentwicklung zu Produkten, die begeistern. Berlin Heidelberg: Springer-Verlag. Seite 94-95

## 5.2 Das Use Case Diagramm zum Ist-Zustand von VipLab

In der unten dargestellten Abbildung sieht man die zentralen Use Cases, die in VipLab bis heute im Ist-Zustand implementiert sind, um dem Nutzer eine User Experience zu vermitteln. Es sind damit nur die Funktionen aufgeführt, die einen Mehrwert für den Nutzer in der Anwendung mit sich bringen. Funktionen, die ausschließlich zur Funktionsweise der Anwendung mit beitragen, brauchen an der Stelle nicht weiter ausgeführt werden. Zum Beispiel das Schließen eines Fensters innerhalb der Anwendung. Das sind keine Mehrwerte für den Nutzer, sondern nur wesentliche Funktionen, die benötigt werden, sodass überhaupt die Anwendung VipLab technisch funktioniert. Die untere Darstellung zeigt die Use Cases für die beiden Rollen Student und Forscher.

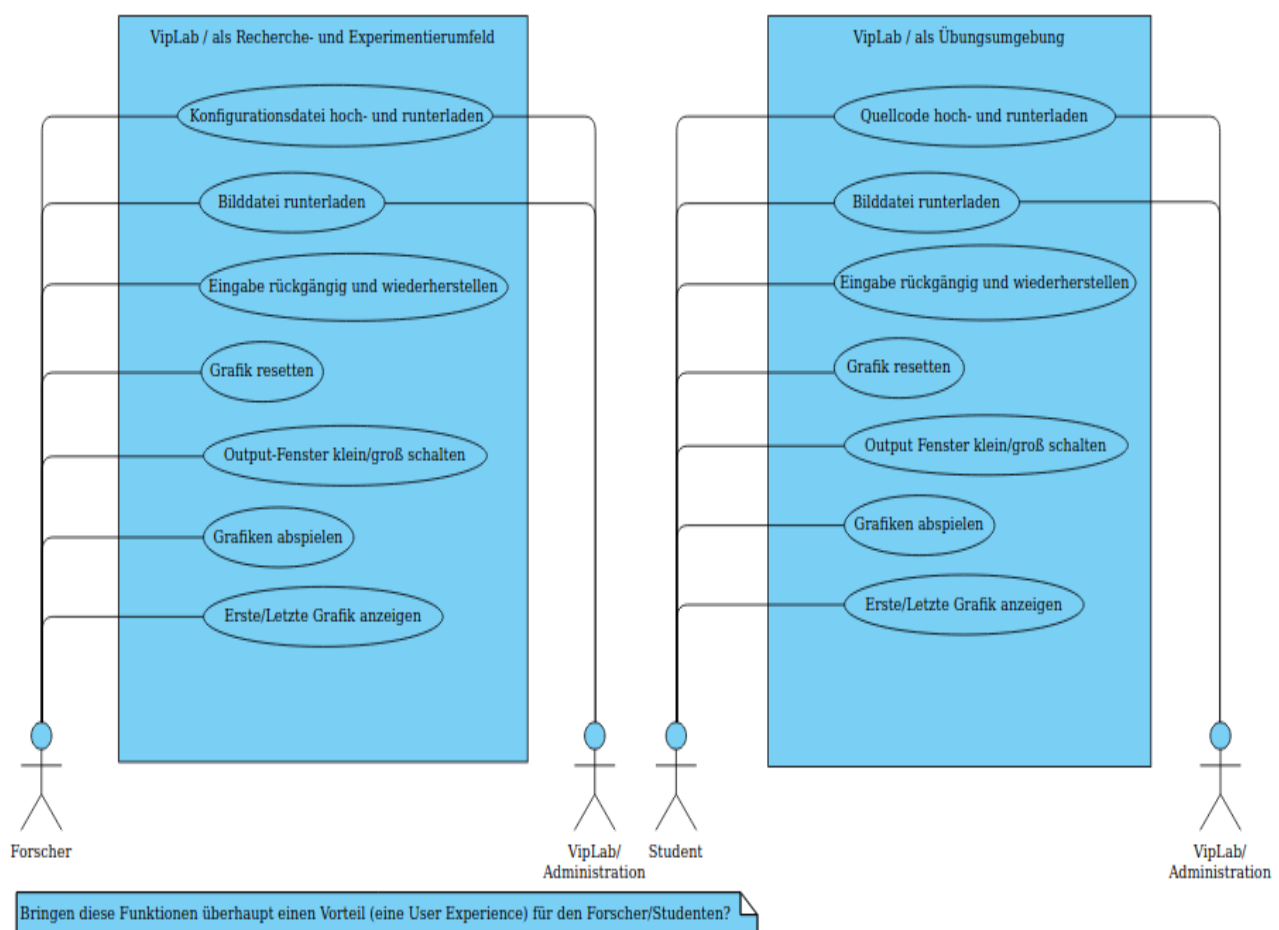


Abbildung 6: Das Use Case Diagramm vom Ist-Zustand ohne Ergänzungen (eigene Darstellung)

Für beide Rollen konnten sieben nützliche Use Cases aus dem Ist-Zustand von VipLab entnommen werden. Diese müssen in der Konzeptionsphase wieder für den Soll-Zustand (das Wireframe) grafisch mit übernommen werden. Da VipLab noch nicht aktiv als Recherche-Instrument für Forscher an der Uni mit eingesetzt wird und noch keine eigene Benutzeroberfläche besitzt, wurden die Use Cases aus dem Ist-Zustand der Studentenansicht vorerst mit übertragen.

In der weiteren Ausarbeitung der Bachelorarbeit muss in den Usability-Tests unbedingt hinterfragt werden, ob diese alten Funktionen überhaupt noch einen Mehrwert für den Nutzer mit sich bringen. Es kann möglich sein, dass die Funktionen gar nicht bis selten von den Akteuren (Nutzern) verwendet werden. Damit wäre eine Weiterverwendung der Funktionen für mehr User Experience in der Anwendung sinnlos. Die Funktionen könnten dann folglich in den neu gestalteten Versionen von VipLab gänzlich weggelassen werden.

Die Funktionen wurden in der Erstentwicklung von VipLab von dem Programmierer, aus dessen Sicht, mit implementiert. Das ist natürlich keine nutzerzentrierte Herangehensweise gewesen und die Wahrscheinlichkeit ist sehr hoch, dass die Funktionen deshalb obsolet sind. Es kann natürlich auch der Fall sein, dass Funktionen von den Nutzern aktiv benutzt werden. Diese Funktionen bleiben dann natürlich für den Soll-Zustand weiterhin implementiert. In den Usability-Tests wird dazu Klarheit geschaffen, indem die Probanden danach gefragt werden, ob sie die Funktionen in der täglichen Arbeit aktiv mit verwenden. Im weiteren Verlauf der Usability-Tests wird außerdem erfragt, welche anderen Funktionen sich die Nutzer aufgrund der individuellen Bedürfnisse bezüglich der Anwendung wünschen.

### **5.3 Die Theorie rund um die Wireframes**

Wireframes sind Webseitenskizzen, auf denen wichtige Inhalte und Elemente (die Seitenstruktur) festgehalten werden, um darauf aufbauend, die späteren Prototypen entwickeln zu können. Ein Wireframe ist dabei zweidimensional und wird mit einer Software erstellt. Die Ausarbeitung ist dabei schon sehr detailliert und man nähert sich der fertigen Anwendung deutlich an. Dennoch ist es immer noch eine Vorentwicklung im Vergleich zu einem fertigen High Fidelity Prototyp.

In einem Wireframe wird festgehalten, wie viele Elemente auf einer Seite zu sehen sind und in welchen Größen und Anordnungen die Elemente vorliegen. Es sind somit immer die Inhaltsbereiche (der wesentliche Content) und beispielhafte Elemente wie Bilder, Überschriften, Fließtext, Listen, Buttons und Funktionen abgebildet. Ein Wireframe wird immer unter den Originalgrößen erstellt. Damit ist gemeint, dass die originale Bildschirmauflösung zur Gestaltung angewandt wird, wie beispielsweise eine Auflösung von 1860 x 1050 Pixel. Es werden also die gleichen Größenverhältnisse eingehalten, wie sie einmal in der späteren Anwendung zu sehen sein werden. In einem Wireframe wird dadurch deutlich erkennbar, ob die einzelnen Elemente (die Buttons und Labels) in Verhältnis zur gesamten Anwendung zu klein oder zu groß wirken. Das kann nur unter Originalgrößen getestet und ausprobiert werden.

Der große Vorteil von einem Wireframe ist es, dass man mit diesem die Inhalte und Funktionen einer Anwendung diskutieren kann, ohne über die Gestaltung sprechen zu müssen. Ein Wireframe ist im Gegensatz zu einem Mockup ästhetisch nicht voll ausgereift. Ein Mockup sieht indessen wie die fertige Anwendung aus. In einem Wireframe wird sich ausschließlich nur auf die Anordnung der Inhalte und Funktionen beschränkt.

Wird einem Auftraggeber oder Vorgesetzten ein grafisch fertig ausgearbeitetes Mockup vorgelegt, dann werden folglich auch immer gestalterische Aspekte mit besprochen, wie beispielsweise die Farbauswahl und das Logo-Design. Ein Wireframe soll diese Diskussion vorerst zurückstellen und die Inhalte, Seitenstruktur und Funktionen in den Gesprächsfokus rücken. Sind beispielsweise in der Anwendung die Funktionen an der richtigen Stelle platziert? Sind die zentralen Inhalte auf der Seite richtig verteilt? Die Ergebnisse aus der Wireframe – Diskussion mit dem Vorgesetzten könne dann wiederum zur Überarbeitung des Wireframes genutzt werden. Die Ästhetik wird unter anderem erst im fertigen High Fidelity Prototyp ausdiskutiert.

Ein Wireframe ist zudem ausgezeichnet dafür geeignet, um eigene Ideen in der Entwicklungsarbeit zu konkretisieren und neue zu entwickeln. Mit Wireframes ist es immer auch schon möglich Usability-Tests durchzuführen, um Nutzer-Feedback zu erhalten, bevor sich Fehler in der User Experience einschleichen. Es kann beispielsweise auch schon getestet werden, ob die Seitenstruktur gefällt. In jeder Projektphase können Usability-Tests mit Probanden durchgeführt werden. Auch zur Dokumentation von den Systemanforderungen (Use Cases) und der grafischen Anordnung sind Wireframes sehr gut geeignet.<sup>18</sup>

#### **5.4 Die Umsetzung der Wireframes für das Projekt VipLab**

Bei einfachen Anwendungen ist es ausreichend, wenn nur von der Startseite ein Wireframe erstellt wird. Das ist dann gegeben, wenn alle weiteren Seiten (Screens der Anwendung) hauptsächlich die gleiche Seitenstruktur mit aufweisen. Es gilt: Existiert für eine Seite eine eindeutige Vorlage für deren Struktur, brauche ich für diese keinen eigenen Wireframe.<sup>19</sup>

VipLab ist vergleichsweise eine einfache Anwendung gegenübergestellt einer umfassenden Webseite, auf der viele unterschiedliche Inhalte verwaltet werden. Es ist daher ausreichend, wenn

---

18 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 157-159

19 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 163-164

nur von der Startseite die Seitenstruktur gestaltet und dokumentiert wird. Auch alle nachfolgenden Seitenansichten (Screens) von VipLab haben nahezu die gleiche Seitenstruktur wie die Startseite.

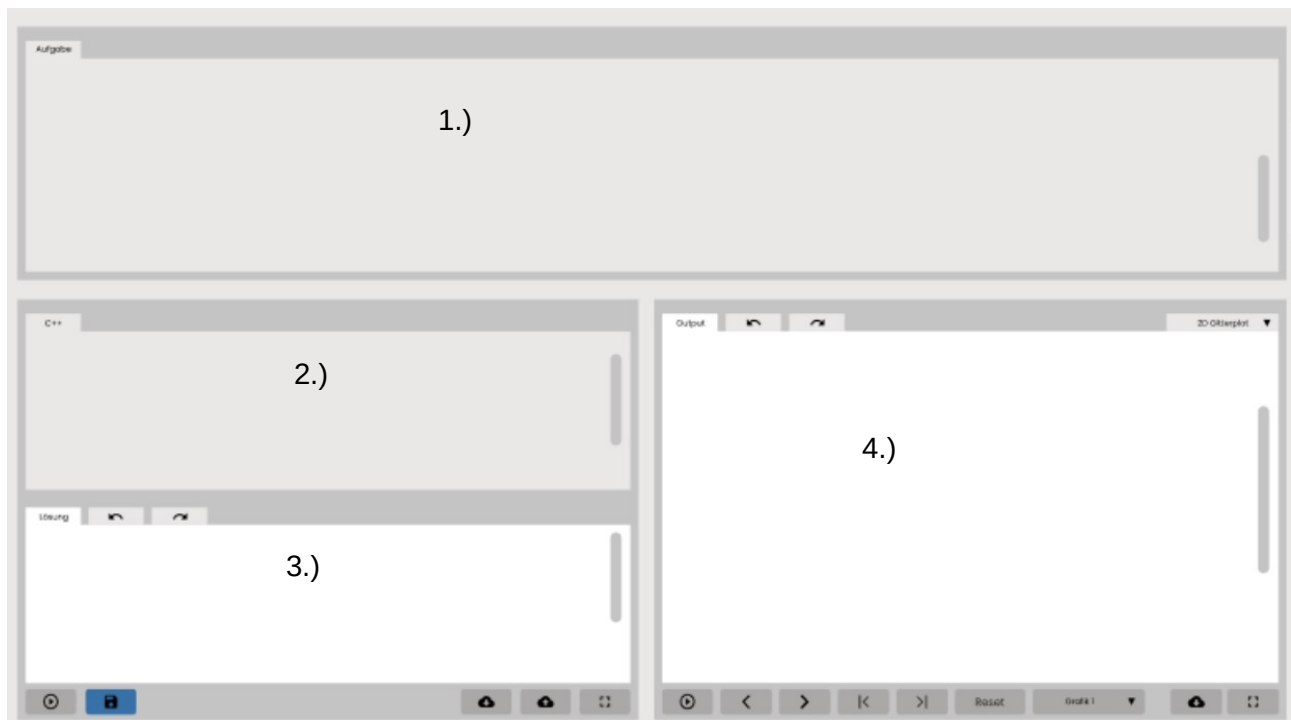


Abbildung 7: Das erste Wireframe vor der Diskussion und Beurteilung (eigene Darstellung)

Die obere Abbildung (Abbildung 7) zeigt das erste erstellte Wireframe von der VipLab Startseite für den Soll-Zustand. Es ist dabei gezielt in Grautönen gestaltet und in vier Inhaltsbereiche unterteilt. Die Seitenstruktur ist dadurch deutlich zu erkennen. Für die Erstellung wurde die kostenlose Onlineanwendung Figma eingesetzt. Die Resultate aus dem Tool sind immer online einsehbar.

- |                         |                            |
|-------------------------|----------------------------|
| 1.) Der Aufgabenbereich | 2.) Die Codevorlage        |
| 3.) Der Lösungsbereich  | 4.) Der IDE Output-Bereich |

In dem Wireframe sind auch schon die zentralen Funktionen (siehe Abbildung 6) eingeteilt und erste Labels (Überschriften) wurden für die Inhaltsbereiche formuliert. Mit diesem Wireframe wurde die erste Diskussion mit dem Auftraggeber zu den Prototypen angestoßen. In der Wireframe - Diskussion haben sich auch erste Erkenntnisse aufgrund der Vorlage herausgebildet, die zur nächsten Überarbeitung herangezogen werden könnten.

1.) Die Inhaltsbereiche sind noch nicht richtig in dem Wireframe verteilt. Der Aufgabenbereich (1.) soll unter anderem nicht in der Form in VipLab mit eingebunden werden, sodass er oberhalb der Anwendung steht. An dieser Stelle muss noch nach einer anderen Lösung gesucht werden.

- 2.) Das Inhaltsfeld für den vorgegebenen Code (2.) wie auch der Lösungsbereich zur Eingabe (3.) sollen näher zusammengeführt werden. Die Unterteilung ist noch viel zu deutlich und zu streng.
- 3.) Die Funktionen (die Use Cases) für „Rückgängig“ und „Wiederherstellen“ sollen anders in der Anwendung angeordnet werden. In der jetzigen Form sehen die Funktionen aus wie Tabs und können vom Nutzer fehlinterpretiert werden. Das kann zu Usability - Problemen führen.
- 4.) Der Button zum Umschalten von der 2D auf die 3D Ansicht muss nur einmal in dem Wireframe mit implementiert werden. Die Funktion muss nicht doppelt belegt werden.
- 5.) Für den Reset-Button ist noch kein geeignetes Icon ausgewählt. An der Stelle muss noch ein richtiges gefunden und eingesetzt werden. Nur ein Label ist nicht ausreichend genug.
- 6.) Zwischen den verschiedenen Output-Grafiken muss nicht unbedingt mit einem Button hin- und hergeschaltet werden können. An der Stelle kann auch mit Tabs gearbeitet werden. Für jede erzeugte Grafik wird ein Tab im Output-Bereich ausgegeben, zwischen denen beliebig gewechselt werden kann. Diese Darstellungsform der Funktion macht zudem die Buttons für den ersten und letzten Plot obsolet. Auf einen Play-Button, der alle Grafiken hintereinander abspielt, wird folglich zudem verzichtet.



Abbildung 8: Das zweite Wireframe nach der Beurteilung und der Überarbeitung (eigene Darstellung)

Mit den oben dargestellten Erkenntnissen wurde ein zweites Wireframe (*Abbildung 8*) erstellt, das die Funktionen und die Seitenstruktur überarbeitet darstellt. In der Überarbeitung wird erkenntlich, wie die finale Seitenstruktur der Startseite von VipLab für beide Prototypen schließlich gestaltet ist. Die Inhaltsbereiche wie auch die Anordnung der Funktionen sind fertig eingeteilt und mit dem Auftraggeber endgültig ausdiskutiert. Damit wurden erste Verbesserungen auf dem Weg zu den fertigen Prototypen vorgenommen. Die folgenden Punkte zeigen nochmal die durchgeführten Überarbeitungen am Wireframe auf.

- 1) Der Aufgabenbereich (*Abbildung 7 / 1.*) ist gänzlich aus dem Wireframe verschwunden und wurde in die Funktion zur Vergrößerung der IDE verschoben. (siehe *Abbildung 15*)
- 2) Der gesamte Lösungsbereich für eine Codeantwort wurde deutlich zusammengeführt.
- 3) Für die Funktionen „Rückgängig“ und „Wiederherstellen“ wurden zwei Buttons erstellt und anders angeordnet (oben rechts).
- 4) Zwischen der 2D und 3D Ansicht kann nur einmalig in einem Tab hin und her geschaltet werden. Je Output-Grafik steht die Funktion einmal zur Verfügung.
- 5) Für den Rest-Button wurde ein geeignetes Icon ausgewählt und eingesetzt.
- 6) Im Output-Bereich werden die Rückmeldungen des Interpreters nur noch als Tabs dargestellt, zwischen denen beliebig gewechselt werden kann. Die Buttons sind entfallen.
- 7) Zudem wurden noch zwei weitere Funktionen unabhängig von dem Ist-Zustand in das Wireframe ergänzt, um die User Experience weiter zu verbessern. Diese Funktionen basieren aber nur auf eigenen Annahmen darüber, was dem Nutzer noch gefallen könnte.

Es wurden noch die Use Cases „Codelösung einreichen“, „IDE Fenster vergrößern/verkleinern“ und „VipLab Aufgabe anzeigen“ für den Prototyp des Studenten mit hinzugefügt. Der letztere Use Case ist dabei nicht in dem Wireframe (*Abb. 8*) mit einsehbar. Für den Prototyp des Forschers wurde noch der Use Case „Konfigurationsdatei umschalten“ zusätzlich ergänzt. Mit dieser Funktion kann der Forscher zwischen zwei Parameteransichten hin und her wechseln. Diese vier zusätzlichen Use Cases müssen aber genauso, wie alle anderen Funktionen, durch den Nutzer in den Usability-Tests validiert werden, um langfristig in die Entwicklungsarbeit mit einzufließen.



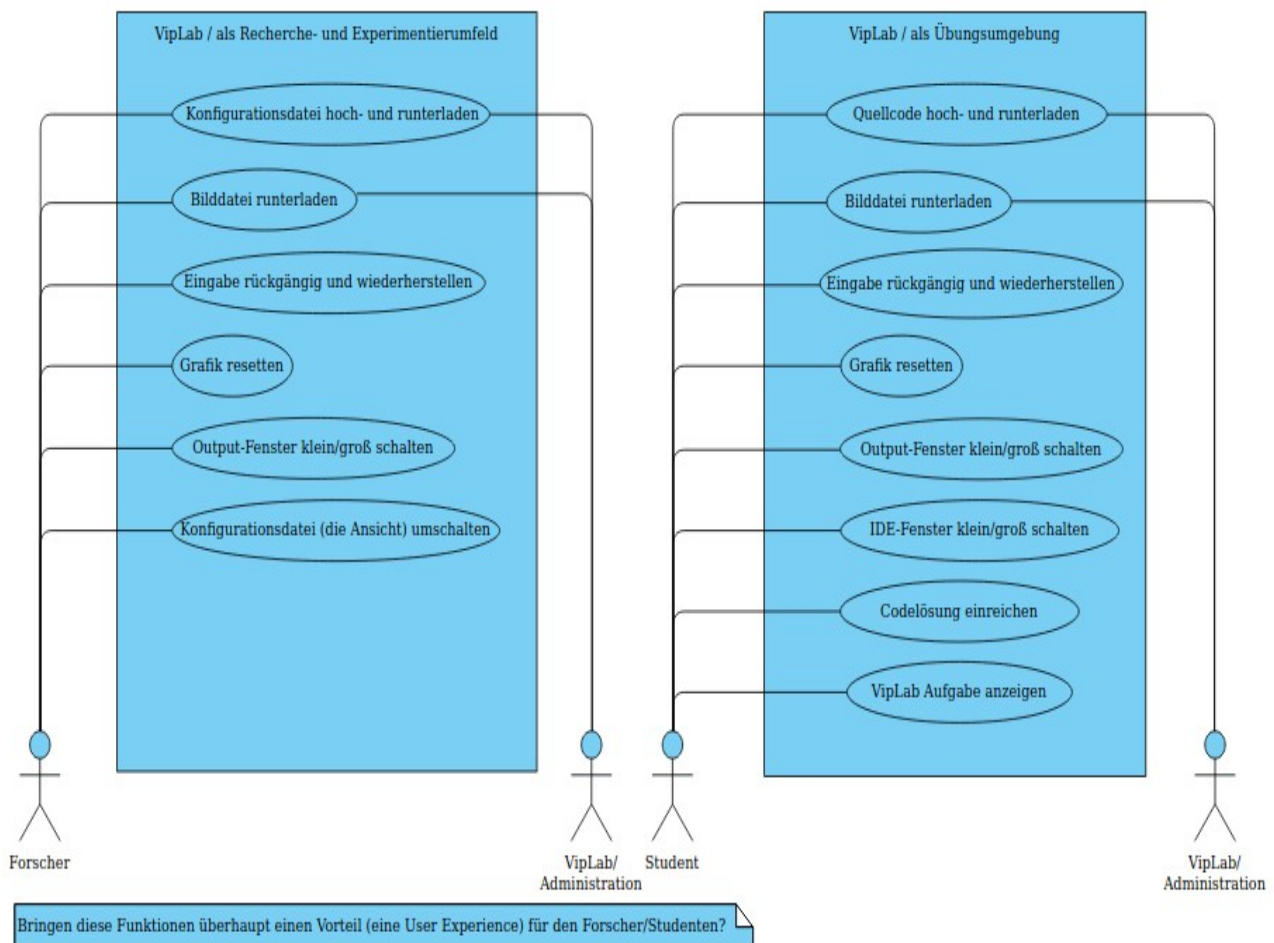


Abbildung 9: Das Use Case Diagram vom Soll-Zustand nach den Überarbeitungen (eigene Darstellung)

Die obere Abbildung (Abbildung 9) zeigt, die zentralen Use Cases die letztendlich in der ersten Iteration mit in die Prototypen (Student und Forscher) implementiert werden. Mit diesem Use Case Diagramm sind somit auch die Use Cases aus dem vorangegangenen Ist-Zustand (Abbildung 6) überarbeitet und aktualisiert wurden. In der weiteren Arbeit gelten die Use Cases aus Abbildung 9.

In der zweiten Diskussion mit dem Wireframe (Abbildung 8) hat sich aber dennoch ergeben, dass die Icons noch nicht richtig ausgewählt sind. In der Bedeutung mögen diese schon richtig für die Funktionen sein, jedoch verwendet die Universität Stuttgart in ihren meisten Anwendungen andere Icons. Um VipLab nicht zu stark grafisch von den anderen Anwendungen abweichen zu lassen, müssen somit folglich noch die Icons ausgetauscht werden, bevor die Prototypen gestaltet werden können.

Im nachfolgenden Unterpunkt wird die Theorie rund um die Icons erläutert und die Icon-Auswahl wird für die interaktiven Prototypen begründet.

## 5.5 Die Theorie rund um die Icons

Icons sind Symbole, mit denen eine Funktion (ein Use Case) in einer Anwendung dargestellt werden kann. Viele dieser Icons sind auch schon von den Nutzern erlernt und verinnerlicht. Beispielsweise ist das Icon zum Schließen einer Anwendung immer ein X. Die Funktion hinter dem X muss den meisten Nutzern daher nicht mehr erläutert werden. Das Symbol ist inzwischen für die meisten Menschen selbsterklärend.

Heute gibt es eine Vielzahl an Icons, die der Nutzer problemlos wiedererkennt. Das bringt den großen Vorteil mit sich, dass diese im Webdesign nutzenstiftend eingesetzt werden können. Icons sind platzsparend bei der Gestaltung einer Anwendung. Anstatt eine Funktion ausführlich schriftlich erklären zu müssen, wird nur das entsprechende Icon dafür eingesetzt. Gerade im mobilen Kontext mit Apps, in denen nur wenig Platz auf dem Bildschirm zur Verfügung steht, ist das ein entscheidender Vorteil. Zudem tragen Icons zum Design einer Anwendung wesentlich bei und lassen die Anwendung verspielter aussehen. Das fördert folglich auch nachweislich die User Experience. Icons geben einer Anwendung immer den letzten Schliff und fügen sich nahtlos in das Gesamtdesign einer Anwendung mit ein.

Der Nachteil von Icons kann aber sein, dass diese zu Usability-Problemen beitragen können. Damit ist gemeint, wenn der Nutzer nicht genau weiß, welche Bedeutung hinter einem Icon steht, und er somit nicht schnell genug seine Absicht in der Anwendung erreichen kann. Von einem unüberlegten Einsatz von Icons ist daher im Webdesign abzuraten.

Icons sollten immer überlegt und bewusst eingesetzt werden, um Problemen mit der Usability entgegenzuwirken. Ein Icon sollte immer nur eingesetzt werden, wenn es leicht verständlich ist und einen hohen Wiedererkennungswert besitzt. Für den Nutzer ist es irreführend, wenn das Icon zudem eine missverständliche Bedeutung hat. Wenn es also eine andere Funktion besitzt, als das Icon-Aussehen beispielsweise im ersten Augenblick impliziert. Ein Icon sollte immer beim ersten Anblick verstanden werden können. Braucht ein Nutzer mehr als 5 Sekunden (die Fünf-Sekunden-Regel), um ein Icon richtig zu interpretieren, dann ist auch der Icon-Einsatz in Frage zu stellen.



Abbildung  
10: Ein  
Beispiel  
Icon aus  
Bootstrap

Das Icon rechts ist der Symbolpalette von Bootstrap entnommen. Es ist schwer auf den ersten Blick festzustellen, welche Funktion hinter dem Icon steht. Um die Usability des Icons dennoch zu erhöhen, kann ein Label mit hinzugefügt werden oder bei einem Mouse-Over wird ein entsprechender Tooltip mit angezeigt.

Es kann für die Usability sehr förderlich sein, wenn zu dem Icon noch ein Label zusätzlich formuliert wird. Bei einem Icon, dessen Wiedererkennungswert geringer ist, kann ein Label deutlich mit dazu beitragen, dass es dennoch von den Nutzern recht schnell verstanden werden kann. Das Label sollte dabei im Idealfall immer gleichzeitig mit dem Icon sichtbar sein. Es soll dem Nutzer damit erspart werden, dass er erst mit einem Mouse - Hover die Funktion angezeigt bekommt und das Label sichtbar wird. Eine gute Usability sollte dem Nutzer immer so wenig Arbeitsaufwand wie möglich in der Anwendung bereiten.

Es ist zudem auch förderlich für die Usability, wenn das Icon international einheitlich die gleiche Bedeutung besitzt. Wenn eine Anwendung von vielen unterschiedlichen Nationalitäten besucht wird, dann sollten auch in der Regel die Icons von allen Besuchern gleich interpretiert werden können. Die Benutzbarkeit und Verständlichkeit der Anwendung wäre sonst, je nach dem jeweiligen Besucher, unterschiedlich, weil jeder die Symbolik anders versteht. Gerade Webseiten, die in verschiedenen Sprachen angeboten werden, müssen prüfen, ob die Icons international gleich interpretiert werden. Sollte ein Icon dennoch missverständlich sein, und es muss ein zusätzliches Label mit hinzugefügt werden, dann kann das wieder folglich zu Übersetzungsschwierigkeiten in der Anwendung führen, wenn durch den Nutzer die Sprache beispielsweise umgestellt wird. Icons sollen deshalb immer auf ihre internationale Gültigkeit und Verständlichkeit hin untersucht werden, sodass nicht unbedingt Labels zusätzlich formuliert werden müssen.

Eine irrtümliche Entscheidung ist es, gänzlich in einer Anwendung auf Icons zu verzichten und stattdessen nur Buttons mit Labels einzusetzen. Das Design einer Anwendung würde durch das Fehlen von Icons merklich leiden und die Nutzung würde deutlich weniger Spaß machen. Die User Experience würde unnötig zugunsten der Usability geschwächt werden.<sup>20</sup>

## **5.6 Die gegenwärtigen Icons in VipLab bezogen auf den Ist-Zustand**

Bei der Analyse des Ist-Zustandes von VipLab wird deutlich klar, dass die Anwendung kaum mit Icons gestaltet wurde. Es werden insgesamt nur acht Icons eingesetzt (*Abbildung 1*).

- 2.) Die Funktionen Quellcode „Hochladen“ und „Runterladen“ (1 Icon)
- 11.) Die Funktionen „Vorwärts“, „Rückwärts“ und „Play“ (5 Icons)
- 1.) Die Funktionen "Wiederherstellen" und "Rückgängig" (2 Icons)

---

20 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 403-405

In VipLab kann über das Icon einer Diskette ein Quellcode hoch- und heruntergeladen werden. Das Icon ist dabei noch sehr irreführend für den Nutzer, da es überwiegend dafür eingesetzt wird, um Inhalte abzuspeichern. Als Referenz kann hierzu Word oder LibreOffice herangezogen werden.

Das Vorwärts, Rückwärts und Play-Icon wird in VipLab schon richtig umgesetzt. Bei der Benutzung der Symbole tritt die Funktion ein, die der Nutzer auch erwartet. Es wird eine weitere Grafik vor- oder zurückgeblättert. Das Play-Icon spielt die erzeugten Grafiken hintereinander ab. Diese Funktionen werden ersetzt durch die Lösung mit der Tab - Ausgabe (Output) für beide Prototypen.

Für die Funktionen „Wiederherstellen“ und „Rückgängig“ wurden auch schon die richtigen Icons eingesetzt. Der geschwungene Pfeil nach links und rechts wird intuitiv von vielen Nutzern in der Funktion schon richtig interpretiert. Ein geschriebener Text oder Code kann dadurch zurück- oder wiederhergestellt werden. Jedoch ist die Positionierung links unten in der Anwendung nicht optimal. Die Icons gehören gebräuchlicher Weise links oben in der Anwendung angeordnet. Viele internationale Anwendungen haben diese Funktionen ebenso links oben angeordnet und das Icon ist dabei auch dasselbe. Das ist inzwischen ein international etablierter Standard und es sollte nicht davon abgewichen werden.

## **5.7 Die internationalen Standards im Webdesign**

Standards helfen den Nutzern dabei sich in der Funktionsweise einer Anwendung besser und schneller zurechtzufinden. Mit Standards wird also die Usability gestärkt, indem die Anwendung grundsätzlich besser verstanden werden kann, weil der Nutzer schon grundlegende Dinge in anderen Anwendungen erlernen könnte. Ein Standard entsteht, indem sich eine Funktionalität in vielen Anwendungen auf gleiche Weise durchsetzt, und deshalb sofort interpretiert und auf gleich Art und Weise bedient werden kann. Ein Nutzer erwartet, dass eine Anwendung genauso wie alle anderen funktioniert.

Der dänische Usability-Experte Jakob Nielsen hat diesen Sachverhalt treffend zusammengefasst: „Nutzer verbringen die meiste Zeit auf anderen Webseiten.“ Die Nutzer gewöhnen sich an die dortigen etablierten Designstandards und -konventionen und übertragen diese auch auf alle anderen Anwendungen. Erfüllt eine entsprechende Anwendung folglich nicht die gleichen Standards und Konventionen, so ist diese für den Nutzer schwieriger neu zu erlernen.

Für die Entwicklung einer Anwendung ist es also nötig, dass ein konsistenter Gebrauch von Terminologie, Bildsprache, Symbolen und sonstigen Elementen eingehalten wird, der bereits den

meisten Nutzern bekannt ist. Dabei ist es ratsam, man orientiert sich anhand der Styleguides größerer Softwareentwickler, die die Interaktionsmuster der meisten Nutzer deutlich mit prägen. Internationale Unternehmen wie Google, Apple oder Microsoft geben oft vor, welche Icons bereits etabliert sind und weltweit von den Nutzern verstanden werden. Die Softwareentwicklungen dieser Giganten sind in der Regel auf dem gesamten Globus bekannt und erlernt.<sup>21</sup>

### **5.7.1 Der Standard (Styleguide) für den Soll-Zustand von VipLab**

Ein definierter Gestaltungsstandard eines Unternehmens für eine Anwendung kann in einem Styleguide zusammenfassend dokumentiert werden. Ein Styleguide ist eine allgemeine Beschreibung der Gestaltungsgrundlage und legt fest, wie eine spätere Anwendung für ein Unternehmen grundsätzlich aussehen muss. Mit einem Styleguide soll eine konsistente Gestaltung erreicht werden und die Wiedererkennung der Anwendung soll sichergestellt werden. Das Dokument ist eher statisch und in sich ein abgeschlossenes Dokument. In einem Styleguide wird unter anderem festgelegt:

- Die Typografie mit den Vorgaben zu Schriftarten und -größen bzw. Abständen.
- Die Farben bzw. die Farbpalette mit den HEX-Codes und eine Anweisung, wie die Farbverwendung in Bezug auf die Farbbedeutung verwendet werden darf.
- Die Bildsprachen und Icons.<sup>22</sup>

An der Universität Stuttgart werden schon verschiedene Anwendungen mit eingesetzt, die grafisch vollständig gestaltet sind. Beispielsweise wird das Learning Management System „ILIAS“ für das Content-Management an der Universität eingesetzt. Die Software ist eine Open Source Entwicklung, in der die Studenten ihre Lernmaterialien aus den Vorlesungen beziehen und verwalten können. Für Dozenten ist es in der Software auch möglich, Onlinetests mit den Studenten durchzuführen. Derzeitig ist VipLab in der Software ILIAS mit eingebunden. Die Studenten können über ihren ILIAS-Account auf VipLab mit zugreifen, um bewertete Übungsaufgaben zu absolvieren.

Für die Webbereitstellung wurde ILIAS mittels des Bootstrap CSS Framework gestaltet und aufgebaut. Bootstrap ist ein international etablierter Standard in der Front-End Webentwicklung. In dem Standard sind CSS und JavaScript basierende Klassen enthalten, die für die Template

---

21 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 533-534

22 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 537-538

Programmierung (Vorlagen) in der Webentwicklung herangezogen werden. Es sind unter anderem die Typografie, Button und Navigations-Elemente standardisiert und können in einer Eigenentwicklung problemlos übernommen werden. Der Standard ist dabei Open Source.<sup>23</sup>

Aufgrund der weltweit großen Bekanntheit von Bootstrap kann davon ausgegangen werden, dass die Konventionen (Icons, Buttons, Typografie usw.) in dem Standard von vielen Nutzern erlernt sind. Bootstrap wird in vielen Anwendungen international angewandt, um das Front-End zu gestalten. In dem eigenen Styleguide sind die Icon-Auswahl und die Farbverwendung in den Buttons zudem konsistent geregelt. Bootstrap erfüllt somit alle Bedingungen für einen Standard, auf den man sich in der Eigenentwicklung der Prototypen beziehen kann.

VipLab soll sich in der späteren Gestaltung nicht erheblich von ILIAS abheben. Für den Soll-Zustand von VipLab sollen also keine neuen Farbkombinationen oder Icons entwickelt werden. Die Anwendung soll sich nur in das Gesamtgefüge von ILIAS grafisch problemlos mit einfügen lassen, ohne dabei eigene Wege zu gehen.<sup>24</sup> Aufgrund dessen wird auch für die Entwicklung des Soll-Zustands der Styleguide von Bootstrap mit herangezogen. Für die zentralen Use Cases (*Abbildung 9*) werden die entsprechenden Icons anhand dem Bootstrap - Styleguide ausgewählt und in die Prototypen mit implementiert. Die Farbauswahl für die Button-Hierarchie wird zudem nach den Regeln von Bootstrap gestaltet.

### 5.7.2 Die Icons nach einem Standard für den Soll-Zustand von VipLab

In der nachfolgenden Abbildung werden die Icons aufgezeigt, die aus dem Bootstrap - Styleguide entnommen wurden, um für die zentralen Use Cases (*Abbildung 9*) eingesetzt zu werden.



Abbildung 11: Die Icons für den Soll-Zustand

- |   |                             |
|---|-----------------------------|
| 1.) Bilddatei oder Quellcode/Konf. herunterladen. | 6.) Ansicht zurückschalten. |
| 2.) Konfigurationsdatei oder Quellcode hochladen. | 7.) Reset                   |
| 3.) Fenster klein schalten.                       | 8.) Play oder Run           |
| 4.) Fenster groß schalten.                        | 9.) Optionsauswahl          |
| 5.) Ansicht umschalten. <sup>25</sup>             |                             |

<sup>23</sup> Wikipedia (2020). Bootstrap (front-end framework).

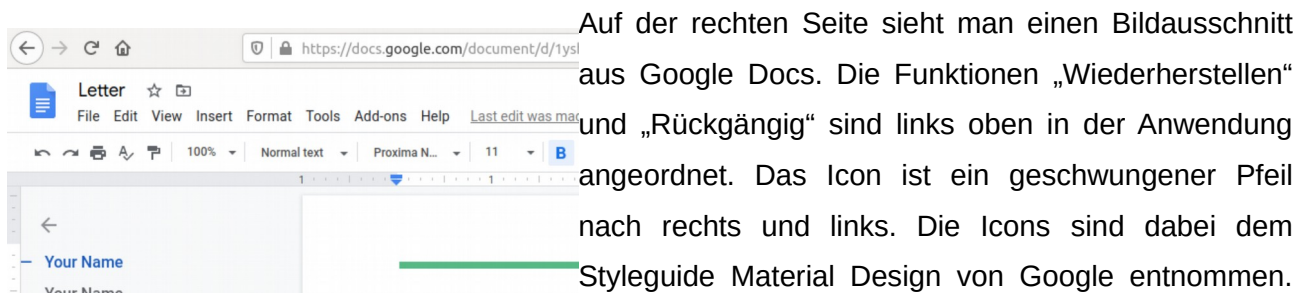
Verfügbar unter: [https://en.wikipedia.org/wiki/Bootstrap\\_\(front-end\\_framework\)](https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)). Letzter Zugriff: 16.07.2020

<sup>24</sup> Persönliches Gespräch mit Pascal Seeland vom 11.Juni.2020

<sup>25</sup> Bootstrap (2019). Bootstrap Icons. Verfügbar unter: <https://icons.getbootstrap.com/>. Letzter Zugriff: 16.07.2020

Für die meisten zentralen Use Cases (*Abbildung 11*) in VipLab kann der Bootstrap - Standard ein international gültiges Icon bereitstellen. Jedoch für zwei Use Cases hält der Standard leider keine passenden Icons bereit. Für die Funktionen „Rückgängig“ und „Wiederherstellen“ sind nicht die passenden geschwungenen Pfeile in dem Standard gegeben und für das Abspeichern/Einreichen einer Codeantwort ist auch keine Diskette als Icon auffindbar.

Für die Ausarbeitung der Prototypen bedeutet das, dass diese Icons noch anhand anderer Standards recherchiert werden müssen. Es kann an der Stelle auch für die Arbeit hilfreich sein, vorerst Vergleiche zu anderen bekannten Anwendungen anzustellen, die die Icons schon erfolgreich einsetzen. Es kann dann folglich überprüft werden, ob es einen bestehenden offenen Standard für die Icons gibt. Schließlich können die fehlenden Icons in VipLab mit übertragen und implementiert werden.



*Abbildung 12: Ein Ausschnitt aus Google Docs (Der Link zum Styleguide mit den Icons.)*<sup>26</sup> Dieses Muster der Anordnung wiederholt sich in einer überwältigenden Anzahl in den verschiedensten Anwendungen. Für VipLab können daher die gleichen Icons gewählt werden und in der selbigen Anordnung angewendet werden. Es handelt sich auch hierbei um eine internationale einheitliche Konvention von Google, die überall von den Nutzern verstanden werden kann.

Der Styleguide Material Design von Google stellt aber auch noch darüber hinaus, eine passende Diskette als Icon bereit, um eine Speicher-Funktion abzubilden.<sup>27</sup> Im derzeitigen Ist-Zustand von VipLab wird dieses Symbol dazu verwendet, um einen Quellcode in der IDE hoch- und herunterzuladen (*Abbildung 1*). Der Icon-Einsatz ist somit im Ist-Zustand missverständlich, wenn das Symbol eigentlich zum Speichern animieren soll. Der Icon-Einsatz widerspricht somit der internationalen Konvention von Google. Der Nutzer ist durch den falschen Einsatz des Symbols derzeitig nur unnötig verwirrt und die Usability in VipLab wird dadurch nur unnötig erschwert.

26 Material Design (2019). Icons. Verfügbar unter: <https://material.io/resources/icons/?search=undo&style=baseline>.  
Letzter Zugriff: 16.07.2020

27 Material Design (2019). Icons. Verfügbar unter: <https://material.io/resources/icons/?search=disk&style=baseline>.  
Letzter Zugriff: 16.07.2020

Für den Soll-Zustand wird das Icon (die Diskette) jedoch weiter beibehalten. Es wird ausschließlich nur die Funktion hinter dem Symbol ersetzt. Zukünftig wird in der Icon-Interaktion der Code abgespeichert, der zur Abgabe eingereicht werden soll. Im derzeitigen Ist-Zustand wird der Code automatisch abgespeichert und eingereicht, sobald der Nutzer die nächste Aufgabe aufruft. Die Anwendung macht also derzeit nicht deutlich sichtbar, dass der Code abgespeichert wird. Was gerade im Systemhintergrund passiert, wird dem Nutzer nicht deutlich in der Anwendung klargemacht. Ein Nutzer, der zum ersten Mal VipLab verwendet, kann dadurch in der Usability (Bedienung) deutlich verunsichert werden. Er kann seine Absicht, eine Codelösung einzureichen, in der Anwendung nicht schnell genug nachkommen. Er weiß einfach nicht, was er in der Anwendung zu machen hat. Die Funktion (der Use Case) ist schließlich nicht sichtbar in der Anwendung dargestellt.

1994 hat der dänische Usability-Experte Jakob Nielsen die 10 Heuristiken für das Interface Design als unabdingbar erklärt. Das sind grobe Richtlinien, keine Regeln, die man in der Anwendungsgestaltung unbedingt mit berücksichtigen sollte. Die Richtlinien können somit für den Einzelfall bei Gestaltungsproblemen interpretiert werden und zu Verbesserungen in den Anwendungen mit beitragen. Für die gegenwärtige Problematik können folgende Heuristiken herangezogen werden:

1.) *Sichtbarkeit des Systemstatus* – was macht das Gerät? - Das System sollte jederzeit klarmachen, was es gerade durchführt. Jede Aktion sollte sofort eine Reaktion zeigen.

2.) *Kontrolle durch den Nutzer* – Der Nutzer sollte das System jederzeit steuern können, und es sollte erlauben zu experimentieren. Eine Rückgängig-Funktion ist dafür entscheidend.

3.) *Selbsterklärung vor Erinnerung* – Der Nutzer sollte nichts lernen und nichts im Gedächtnis behalten müssen. Alle aktuell notwendigen Informationen sollten direkt einsehbar sein.<sup>28</sup>

Die Heuristiken machen deutlich, dass VipLab nicht weiterhin unbemerkt im Hintergrund Codeeingaben abspeichern sollte. Der Nutzer sollte einen deutlichen Befehl geben können, um eine Codeeingabe abzuspeichern und damit einzureichen. Die Eingabe sollte dabei auch immer wieder zurückgenommen werden können (*Kontrolle durch den Nutzer*). Nachdem der Code eingereicht wurde, sollte das System eine klare Rückmeldung geben, was im Systemhintergrund passiert ist (*Sichtbarkeit des Systemstatus*). Mit einem Speicher-Button ist dem Nutzer in der

---

<sup>28</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 39-40



Anwendung auch sofort klar, wie er seine Absicht in der Anwendung erreichen kann. Er muss nicht spekulieren, ob der Code automatisch abgespeichert und eingereicht wird (*Selbsterklärung vor Erinnerung*).

Alle Funktionen in VipLab sollten daher immer sichtbar dargestellt sein und eine klare Rückmeldung in der Interaktion geben, sodass der Nutzer weiß, was ihm in der Anwendung für Funktionen zur Verfügung stehen. In der Ausarbeitung des Soll-Zustandes wird bei jeder Funktionsinteraktion eine klare Rückmeldung gegeben, was durchgeführt werden soll und ob der Nutzer das auch tatsächlich möchte. Dadurch werden deutlich die Usability und somit auch die gesamte User Experience weiter ausgebaut.

## 5.8 Die Theorie zu den Buttons

Ein Button ist ein grafisches Element innerhalb einer Anwendung, das eine URL enthält. Button kann ins Deutsche mit Schaltfläche übersetzt werden. Ein digitaler Button besteht aus zwei Bestandteilen. Zum Einen besitzt er immer eine grafische Form und zum Anderen kann dieser Form immer noch ein Text (Label) oder Icon zentral ausgemittelt mitgegeben werden. Nur sehr selten besitzt ein Button weder einen Text noch ein Icon. Ein Button erkennt man hauptsächlich an seiner länglichen Form und der Beschriftung. Eine längliche Form ist dabei zwingend in der Konzeption erforderlich. Hingegen, ob der Button runde oder gerade Ecken erhält, sind in der Gestaltung frei wählbar. Besonders zu berücksichtigen ist jedoch, dass alle Buttons, die in einer Anwendung implementiert werden, die gleichen Abmessungen (Länge und Breite) haben müssen. Unterschiedliche Abmessungen in der Buttongröße führen nur dazu, dass der Nutzer unnötig verwirrt wird und das hemmt folglich auch die gesamte Usability der Anwendung. Mit einer unterschiedlichen Buttongröße wird gegen ein zentrales Gestaltungsgesetz im Webdesign verstoßen.<sup>29</sup>

In der Gestaltung einer Anwendung ist erforderlich, dass die menschliche Wahrnehmung zwingend immer berücksichtigt wird. In der Konzeption ist es daher besonders erforderlich die Gestaltungsgesetze mit zu beachten. Die Gestaltungsgesetze beschreiben, wie unser Gehirn aus der gigantischen Vielzahl an Reizen, die über die Sinnesorgane empfangen werden, einzelne Objekte erkennt und in unsere Umwelt sinnvoll mit einordnet. Die Gestaltungstheorie ist daher für die visuelle Gestaltung von Anwendungen ausgesprochen nützlich.<sup>30</sup>

---

29 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 416-418

30 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 44

Ein Gesetz bezieht sich auf die Ähnlichkeit von Objekten. Es besagt, dass Elementen mit der gleichen Form und/oder Farbe, auch weitere ähnliche Eigenschaften zugeschrieben werden können.<sup>31</sup> Daher ist es wichtig, dass konsequent die gleiche Buttongröße und/oder Farbe in einer Anwendung eingehalten wird. Nur auf diese Weise erkennen die Nutzer, dass hinter der gleichen Form eine Funktionalität steht und nichts Anderes. So klickt z. B. jemand auf den Button, und eine Funktion (z.B. Screen vergrößern) wird darauffolgend in der Anwendung ausgelöst.

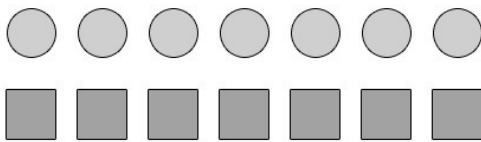


Abbildung 13: Das Gestaltungsgesetz der Ähnlichkeit

Die Abbildung links visualisiert dieses Phänomen. Die grauen Kreise unterscheiden sich vermutlich auch in der Funktion von den anderen Formen.

Eine andere Elementen-Form oder -Farbe kann beispielsweise dem Nutzer auch signalisieren, dass es sich hierbei nur um ein nicht anklickbares Element handelt und um keine interaktive Funktion, weil dem Element andere Eigenschaften zugeschrieben werden können. Ein nicht anklickbares Element wird in der Konzeptionsphase für den Soll-Zustand von VipLab an zwei Stellen benötigt.

## 5.9 Die Labels für die Inhaltsbereiche in VipLab

Auch wenn die Selbstbeschreibungsfähigkeit, bezogen auf eine Anwendung, eine Anforderung der Normen zur Gebrauchstauglichkeit ist, sind manche Elemente einer Anwendung trotzdem auf ein Label (Beschriftung) angewiesen, um von den Nutzern verstanden werden zu können. Dazu gehören auch Elemente wie Formulare, Buttons, Filter oder Boxen.<sup>32</sup> Auf die Boxen muss im Kontext zu VipLab genauer eingegangen werden.

In beiden Prototypen (Forscher und Student) sind zwei zentrale Boxen (*Inhaltsbereiche*) gestaltet. Zum einen der Parameter- und Lösungsbereich und zum anderen der Output-Bereich (*Abbildung 8*). Diese beiden Bereiche können ohne ein Label nicht ohne weiteres von den Nutzern verstanden werden. Es müssen zwei Labels formuliert werden, die die Bereiche eindeutig beschreiben, sodass der Nutzer sofort den Kontext zur Anwendung versteht. Die Labels müssen dabei so eindeutig und einfach wie nur möglich formuliert sein. Das Label sollte zudem nur aus einem Wort bestehen und den Nutzern schon geläufig sein. Damit kann es sofort verstanden werden, auch ohne den genauen Anwendungskontext zu kennen.<sup>33</sup>

31 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 45

32 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 449

33 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 455

Diese Labels müssen folglich in einer anderen Elementen-Form oder -Farbe implementiert werden, sodass der Nutzer nicht die funktionalen Buttons (mit einem zentralen Use Case) mit den nicht anklickbaren Elementen zu den Inhaltsbereichen verwechselt. Diese Elemente sollen in der Anwendung nur informieren und keine Funktionalität wie ein anklickbarer Button bereitstellen.

In einer Anwendung hat ein Button auch die Aufgabe, einen Angebotscharakter (Affordanz) inne zu haben. Damit ist gemeint, dass er den Nutzer dazu verleiten soll, die Schaltfläche (den Button) gerne zu betätigen, um die spezifische Funktion aufzurufen. Als drückbar wird erst ein Button wahrgenommen, wenn er aus der gesamten Bildschirmfläche deutlich heraussteicht. Daher ist es wichtig, dass der Button einen deutlichen Kontrast zu der Untergrundfarbe aufweist und auffällt. Es ist zudem wichtig, dass der Button durch die Größe und Farbe sowie der erwartungskonformen Platzierung den Nutzern direkt auffällt. Ein Button steht mit seiner spezifischen Funktion meistens an der Stelle, wo der Nutzer diesen schon aufgrund seiner Vorerfahrungen erwarten kann. Wird ein Button aktiviert, dann wird er kurzzeitig auch heller dargestellt.<sup>34</sup>

### **5.10 Die Button-Hierarchie**

Eine sehr auffällige Unterscheidung zwischen Buttons kann die Einfärbung sein. Mit einer unterschiedlichen Abstufung der Farbwahl kann dem Nutzer eine sogenannte Button-Hierarchie signalisiert werden. In der späteren Anwendung werden somit primäre und sekundäre Buttons erkenntlich. Primäre Buttons sind wichtige Buttons in einer Anwendung, die auffälliger gestaltet sind, beispielsweise durch eine bunte Farbe. Zudem sind sie immer rechtsläufig zu den anderen Buttons angeordnet. Die Button-Hierarchie hat zur Aufgabe, den Nutzer durch eine Anwendung zu leiten. Ein wichtiger Button speichert beispielsweise die Inhalte in einer Anwendung. Ein sekundärer Button ist weniger wichtig und hat dementsprechend eine weniger auffällige Farbe, wie beispielsweise grau. In der Konzeptionsphase ist es empfehlenswert, nur mit zwei Farben (Bunt und Grau) zu arbeiten, weil die Usability für den Nutzer dadurch zunimmt. Es ist also ausreichend in der Gestaltung, wenn nur mit einem Primär (Bunt) und Sekundär (Grau) Button gearbeitet wird und nicht unnötig eine zweite auffällige bunte Button-Farbe eingeführt wird. Es wird somit nur ein Button farblich hervorgehoben, während die anderen in grau weiterhin dargestellt werden.

Wird jedoch eine Anwendung regelmäßiger verwendet und der Nutzer lernt aktiv die Umgebungsgegebenheiten anzuwenden, dann ist auch die Verwendung einer zweiten auffälligen bunten Farbe noch zulässig. In dem Fall ist der Primär Button und der Sekundär Button farblich

---

34 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 417

deutlich mit unterschiedlichen Signalfarben hervorgehoben. Die übrigen Buttons verbleiben dabei weiterhin in der Farbe Grau.<sup>35</sup>

### 5.11 Die Farbauswahl für die Buttons in dem Soll-Zustand

Für die Gestaltung von VipLab wird der Standard von Bootstrap mit herangezogen (*Kapitel 5.4.1*). In diesem Standard sind auch die Farben reglementiert, die für die Button-Hierarchie eingesetzt werden können. Damit werden die Buttons in VipLab auch dem Aussehen von ILIAS angepasst. Die untere Auflistung zeigt die Farbpalette, die Bootstrap für die Buttons zu Verfügung stellt.

*Die Button-Farben:*

- **Text-Primary** – Primär Button, HEX-Code: #0275d8
- **Text-Secondary** – Sekundär Button, HEX-Code: #625e5e
- **Text-Success** – Success Button, HEX-Code: #5cb85c
- **Text-Danger** – Danger Button, HEX-Code: #d9534f
- **Text-Warning** – Warning Button, HEX-Code: #f0ad4e
- **Text-Info** – Info Button, HEX-Code: #5bc0de<sup>36</sup>

In der Benutzeroberfläche des Studenten-Prototyp liegt der zentrale Fokus darauf, dass der Nutzer seine Codelösung abspeichert bzw. die Aufgabe einreichen kann, bevor er zur nächsten Aufgabe wechselt. Daher ist in der Button-Hierarchie farblich das Speicher-Icon (die Diskette) hervorgehoben. Dafür wird der grüne Farbton von Bootstrap verwendet (**Text-Success**). Der grüne Farbton soll den Erfolg in der Anwendung repräsentieren. Eine Aufgabe wurde erfolgreiche beantwortet bzw. eingereicht. Der Speicher-Button ist dabei der Sekundär Button in der Button-Hierarchie der Benutzeroberfläche.

Der Primäre Button ist in diesem Prototyp der Play-Button, der den Code in der IDE ausführt. Dieser ist mit einer blauen Farbe (**Text-Primary**) aus dem Bootstrap Standard hervorgehoben. Der blaue Farbton ist extra für den primären Button reserviert. In der Button-Hierarchie soll der Nutzer als erstes den primären Button betätigen, um den Code auszuführen. Darauf folgend soll der Nutzer im zweiten Schritt den sekundären Button verwenden, um den Code einzureichen. Die Button-Hierarchie führt somit durch die Anwendung, sodass eine bestimmte Absicht in der

---

35 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 418-419

36 Bootstrap (2019). Buttons. Verfügbar unter: <https://getbootstrap.com/docs/4.0/components/buttons/>. Letzter Zugriff: 16.07.2020

Anwendung problemlos erreicht werden kann. Die übrigen grauen Buttons sind weitere Funktionalitäten, die dem primären und sekundären Button untergeordnet sind.

In der Benutzeroberfläche des Forscher-Prototyp ist die Button-Hierarchie leicht abgeändert im Vergleich zur Studentenansicht. Es gibt kein Button zum Speichern bzw. Einreichen von Lösungen. Es ist ausschließlich der primäre Button blau (**Text-Primary**) eingefärbt, um eine Konfigurationsdatei in der Anwendung auszuführen. Die sekundären Buttons sind Grau dargestellt.

Nachdem die zentralen Funktionen (Use Cases) ermittelt sind und ein finales Wireframe diskutiert und ausgearbeitet ist, können folglich im nächsten Schritt die interaktiven Prototypen angestrebt werden. In diesen sind auch alle theoretischen Ausarbeitungen mit enthalten, die im Verlauf der Projektphasen angesammelt wurden, wie beispielsweise die richtige Button-Hierarchie, die Icon-Auswahl oder die Labels für die Inhaltsbereiche.

Damit wird die Konzeptionsphase in der Projektarbeit abgeschlossen, und es kann in die praktische Umsetzungsphase übergegangen werden. In dieser wird das Wireframe (*Abbildung 8*) weiter verfeinert und die daraus resultierenden High Fidelity Prototypen können schließlich am Projektende auch mit den Nutzern getestet werden. Das Wireframe wird aber vorerst in der Umsetzungsphase weiter zu einem Mockup ausgearbeitet. Ein Mockup ist dabei wesentlich detaillierter als ein Wireframe.

## 6 Die Umsetzungsphase

### 6.1 Die Theorie zu Mockups und interaktiven Prototypen

Mockups und Prototypen visualisieren beide die Konzeptarbeit für ein Projektthema. Mockups sind dabei, wie schon Wireframes, statische einzelne Screens. Ein Mockup wirkt aber im Vergleich zu einem Wireframe täuschend echt. Damit ist gemeint, dass ein Mockup wie die reale fertige Anwendung mit allen grafischen Elementen aussehen kann. In einem Wireframe wird sich stark auf die grundsätzliche Seitenstruktur (Inhaltsstruktur) und die darin enthaltenen Funktionen konzentriert. Visuelle Designaspekte wie Layout, Bildwelten, Farben oder Typografie werden dabei außen vor gelassen. Genau diese fehlenden Aspekte werden in Mockups mit aufgenommen. Für alle Screens, die einmal später in der Anwendung zu sehen sein werden, wird jeweils immer ausgehend von dem finalen Wireframe (*Abbildung 8*), ein Mockup erstellt. Es entwickeln sich somit in der Projektarbeit mehrere Mockups für die verschiedenen Screens von VipLab.<sup>37</sup>

---

<sup>37</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 181

Ein Prototyp enthält alle erstellten Mockups und führt diese zusammen. Damit wird der Prototyp interaktiv für den Nutzer und er kann sich durch diesen wie in der fertig programmierten Anwendung klicken. Der Vorteil von einem Prototyp im Vergleich zu Mockups ist, dass die gesamte Funktionalität transportiert werden kann. Mit dieser Interaktivität wird der Prototyp automatisch für die Probanden in allen Funktionen auch testbar. Es kann beispielsweise getestet werden, wie viel Zeit (Klicks) ein Proband benötigt, um von der Startseite in einen anderen Unterpunkt zu wechseln oder ob er eine Suchfunktion auf Anhieb findet. Ein Prototyp eignet sich auch gut dazu, um konzeptionelle Ansätze oder auch die technische Machbarkeit mit den Nutzern auszuprobieren. Prototypen simulieren also eine Anwendung, die erst zu einem späteren Zeitpunkt programmiert wird.

Daraus ergibt sich der große Vorteil, dass Usability – Probleme, die in der Testphase ermittelt werden, leichter aus einem Prototyp wieder auskorrigiert werden können, als aus der fertig programmierten Anwendung. Die Kosten für Korrekturen und Optimierungen sind dadurch wesentlich geringer als bei einem fertigen Programm. Es ist auch allgemein wirtschaftlicher, vorerst einen Prototyp für eine Anwendung zu entwickeln, weil dadurch auch sichergestellt werden kann, dass die Anwendung nutzerzentriert entwickelt wird. Es wird damit auch die Wahrscheinlichkeit minimiert, dass die Anwendung nach der Veröffentlichung von den Nutzern nicht angenommen wird, und man viel Geld für eine Neuentwicklung aufwenden muss.

Ein Prototyp kann hinsichtlich der visuellen Gestaltung, Funktionalität und nach den Inhalten einen unterschiedlichen Detaillierungsgrad annehmen. Demnach unterscheidet man, ob ein Prototyp Low-Fidelity, Medium-Fidelity oder High-Fidelity ausgearbeitet ist. Fidelity kann ins Deutsch synonym mit „Ähnlichkeit“ übersetzt werden. Wie ähnlich sollte also der Prototyp im Vergleich zum Endprodukt werden? Der Detaillierungsgrad für einen Prototyp sollte immer fest vordefiniert sein. Man berücksichtigt dabei folgende drei Dimensionen:

- *Visuelle Gestaltung:* Ist der Prototyp grob skizziert oder schon detailliert ausgestaltet? (skizziert < > ausgestaltet)
- *Funktionalität:* Wie viele Funktionen werden in dem Prototypen umgesetzt? Nur einzelne oder alle? (statisch < > funktional)
- *Inhalt:* Beinhaltet der Prototyp Dummytext oder ist er mit den geplanten Inhalten befüllt? (Lorem ipsum < > redaktioneller Content)<sup>38</sup>

---

38 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 182-183

Für die Ausarbeitung der VipLab – Prototypen müssten genau dieselben Fragen aufgeworfen werden. Welche Inhalte beinhaltet der Prototyp? Wie ist er visuell ausgearbeitet und bis zu welchem Grad sind die Funktionalitäten interaktiv? Wirkt der Prototyp wie eine echte Anwendung?

Es ist wichtig in der Planung festzulegen, was die Zielsetzung mit dem Prototyp ist, um danach den Detaillierungsgrad zu bestimmen.<sup>39</sup> Was möchte man mit dem Prototyp erreichen? Sollten beispielsweise nur Funktionalitäten diskutiert werden oder werden umfängliche Usability-Tests angestrebt, mit verschiedenen grafischen Designvarianten? Befindet man sich in einer frühen Projektphase, dann stehen die Funktionalitäten (Use Cases) in der Ausarbeitung im Vordergrund und mit einem Prototyp sollen auch nur diese gezielt validiert werden können. Hierfür muss der Prototyp grafisch nicht voll ausgereift sein. Es würde ausreichen den Prototyp in Medium-Fidelity zu entwickeln. Der Detaillierungsgrad wäre bei Medium-Fidelity eine Zwischenstufe zwischen Low-Fidelity und High-Fidelity. Umfangreiche Usability-Tests, in denen auch die grafische Oberfläche validiert werden soll, werden in High-Fidelity designt.

#### **6.1.1 Medium-Fidelity**

Medium-Fidelity Prototypen sind schon detailliert, haben mehrere Funktionalitäten abgebildet und haben gegebenenfalls auch schon echte Inhalte mit enthalten. Es fehlen jedoch noch entscheidende Details im Design, wie beispielsweise eine Grafik oder Bilder. Man sieht dem Prototypen noch an, dass es sich nur um eine Attrappe handelt, jedoch können in dieser Version, Testpersonen oder Kunden schon ein genaueres Bild zum Konzept mit den Funktionen gewinnen.<sup>40</sup>

Die Wireframes, die in der gegenwärtigen Bachelorarbeit entwickelt wurden, entsprechen dem Niveau von Medium-Fidelity. Die Seitenstruktur ist schon klar definiert. Ebenso sind schon erste zentrale Funktionen (Use Cases) in dem Wireframe angeordnet und können erläutert und diskutiert werden. Grafiken und gestalterische Elemente sind noch nicht mit enthalten (*Abbildung 8*).

Je weiter aber eine Entwicklungsarbeit vorangeschritten ist, umso mehr sollte auch das Design des Prototyps verfeinert und konkrete Inhalte hinzugefügt werden. Damit kann dann auch Feedback zur finalen grafischen Oberfläche von den Probanden in den Tests mit eingeholt werden.<sup>41</sup>

---

39 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 185

40 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 184

41 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 183

Die Ausarbeitung der VipLab – Prototypen ist in der Projektarbeit, von dem Wireframe ausgehend, weiter vorangeschritten und auch gestalterische Elemente haben Einzug in die Entwicklungsarbeit gefunden. Beispielsweise wurden Farben für die Button-Hierarchie ausgewählt und es wurden grafische Elemente (Grafiken) in die Prototypen mit implementiert. Diese Punkte wurden ausgearbeitet, um die Prototypen über die Funktionalitäten hinaus weiterzuentwickeln, um in den Usability-Tests neben den funktionalen Komponenten auch grafische testen zu können.

Die Zielsetzung für die Prototypen (Student und Forscher) in der vorliegenden Bachelorarbeit ist, diese vollumfänglich funktional wie auch grafisch mit den Probanden testen zu können. Dafür mussten die Prototypen zwangsläufig auch auf ein High Fidelity Niveau aufgearbeitet werden. Die Prototypen wurden für die Testingphase zwangsläufig von einem Medium-Fidelity Niveau auf ein High-Fidelity Niveau weiter angehoben. Der Detaillierungsgrad für die Prototypen hat sich durch die spezifische Zielsetzung im Projekt schließlich ergeben.

### **6.1.2 High-Fidelity**

Ein High-Fidelity Prototyp simuliert, so wie ein Mockup, schon täuschend echt eine fertige Anwendung. Ein High-Fidelity Prototyp ist interaktiv und enthält alle Fiktionalitäten. Diese Prototypen sind daher sehr zeitaufwendig in der Erstellung und sollten daher immer am Ende jeder Entwicklungsarbeit angefertigt werden.<sup>42</sup>

### **6.1.3 Die Umsetzung der interaktiven Funktionen im Prototyp**

Bevor ein Prototyp mit den Probanden getestet werden kann, sollte festgelegt werden, welche Teilbereiche der Anwendung mit den Funktionen interaktiv umgesetzt werden. Der Umfang des Prototyps ist davon abhängig, wie viele Funktionalitäten man in einem Test prüfen möchte. Standardfunktionen, wie beispielsweise ein Kontaktformular oder eine Auswahlliste, müssen nicht zwingend durch einen Prototyp funktional umgesetzt werden. Hingegen müssen zentrale Funktionen (Use Cases), die in der Analysephase erhoben wurden, zum Test funktional sein. Diese Funktionen sind zentrale Interaktionsaufgaben im Usability-Test, die validiert werden müssen. Auch Funktionen, die in der Gestaltung ein neues ungewohntes Design erhalten haben, müssen gezielt validiert werden, ob der Nutzer die Funktion versteht und gut anwenden kann. Die neue Darstellungsform ist in so einem Fall von dem Nutzer noch nicht erlernt und verinnerlicht.

In der vorliegenden Bachelorarbeit wurde beispielsweise die Ausgabe von mehreren Outputs so organisiert, dass die Inhalte als Tabs ausgegeben werden. In dem Ist-Zustand von VipLab ist diese

---

42 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 184



Funktion (*Abbildung 1*) noch so geregelt, dass über einen Vor- und Zurück-Button zwischen den Outputs gewechselt werden kann. Der Nutzer ist somit noch die alte Darstellungsform der Funktion gewohnt. Die neue Darstellungsform, die durch Tabs geregelt ist, muss erst noch gelernt und validiert werden.

In einem Prototyp sollten immer wenige Funktionen detailliert umgesetzt werden. Es ist davon abzuraten, alle Funktionen in einem Prototyp gänzlich oder nur oberflächlich abzubilden. Eine Orientierungshilfe ist das Pareto-Prinzip: Konzentrieren Sie sich auf die 20 Prozent aller Funktionalitäten, die der Nutzer zu 80 Prozent seiner Besucherzeit voraussichtlich verwendet oder nutzen soll. Der Fokus soll also auf die zuvor erläuterten zentralen, häufig genutzten Funktionen gelenkt werden, die beispielsweise in der Analysephase erhoben werden. Die zentralen Funktionen werden für die Testingphase in kleinere User Scenarios (im Fragebogen) verpackt, um angemessen getestet werden zu können. Für diese Scenarios werden in den Prototypen dann auch folglich an den richtigen Stellen einmalig gänzlich, der richtige Navigationspfad implementiert, um die Funktionalität im Test überprüfen zu können. Der Nutzer startet also, für ein Szenario, in einer bestimmten Ansicht und klickt sich den Funktionen entlang sukzessiv durch alle weiteren Ansichten in dem Prototyp, bis er die Aufgabe erledigt hat. Im Test hält sich der Proband dann entweder an den richtigen Navigationspfad oder er wählt einen anderen Weg. Wird der richtige Navigationspfad eingehalten, dann gilt die Aufgabe als erfolgreich bestanden. Bei Pfadabweichung gilt die Aufgabe als nur teilweise oder gar nicht bestanden.<sup>43</sup>

Für die Bachelorarbeit wurde das Designtool Figma herangezogen, um alle Wireframes, Mockups und die interaktiven Prototypen zu designen. Die interaktiven Prototypen wurden in der Testingphase abschließend mit dem Tool Maze validiert. In Maze ist es möglich, autonome Remote-Usability-Tests mit Probanden durchzuführen. Dafür wurden die Prototypen aus Figma in das Tool Maze importiert und die qualitativen Fragebögen wurden mit hinzugefügt (*Anlage 3 + 4*).

## **6.2 Der ausgearbeitete Soll-Zustand von VipLab**

Im nachfolgenden Text wird der Soll-Zustand von dem VipLab - Prototyp vorgestellt, der speziell für die Studentenschaft entwickelt wurde. Es wurden insgesamt zwei unterschiedliche Prototypen in Figma designt. Eine Ansicht, die speziell den Bedürfnissen des Forschers angepasst ist, und eine andere für die Studentenschaft. Für die Prototypen – Gestaltung wurde die aktuelle Theorie zu UX/UI herangezogen, um die Funktionen (*Abbildung 9*) in der Anwendung zu gestalten.

---

43 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 185-187

In diesem Unterkapitel werden alle Screens aufgezeigt und erläutert, die auch im interaktiven Prototyp klickbar dargestellt sind. Damit ist in der gegenwärtigen Bachelorarbeit auch eine Dokumentation für die beiden Prototypen schriftlich festgehalten. Der zweite Prototyp, der speziell für die Forschung entworfen wurde, ist im Anhang 2 der Bachelorarbeit ebenso einsehbar.

Die untere Grafik (Abbildung 14) ist die Startseite für die Studenten, die ihre Codeaufgaben in VipLab absolvieren möchten. Der Student bekommt immer als allererstes diese Ansicht zu sehen, wenn er die Anwendung startet. Auf der linken Seite ist der Lösungsbereich zu sehen, indem eine Codeeingabe eingegeben werden kann. Auf der rechten Seite sieht man den Output den der Code erzeugt. Der Output kann aus Grafiken, Textzeilen oder Dateien bestehen. Alle zentralen Use Cases (Abbildung 9) sind grafisch realisiert und mit einer Nummer in den Screens gekennzeichnet. Im nachfolgenden Text werden die Funktionen nochmals nach der Überarbeitung des Ist-Zustandes erläutert und begründet.

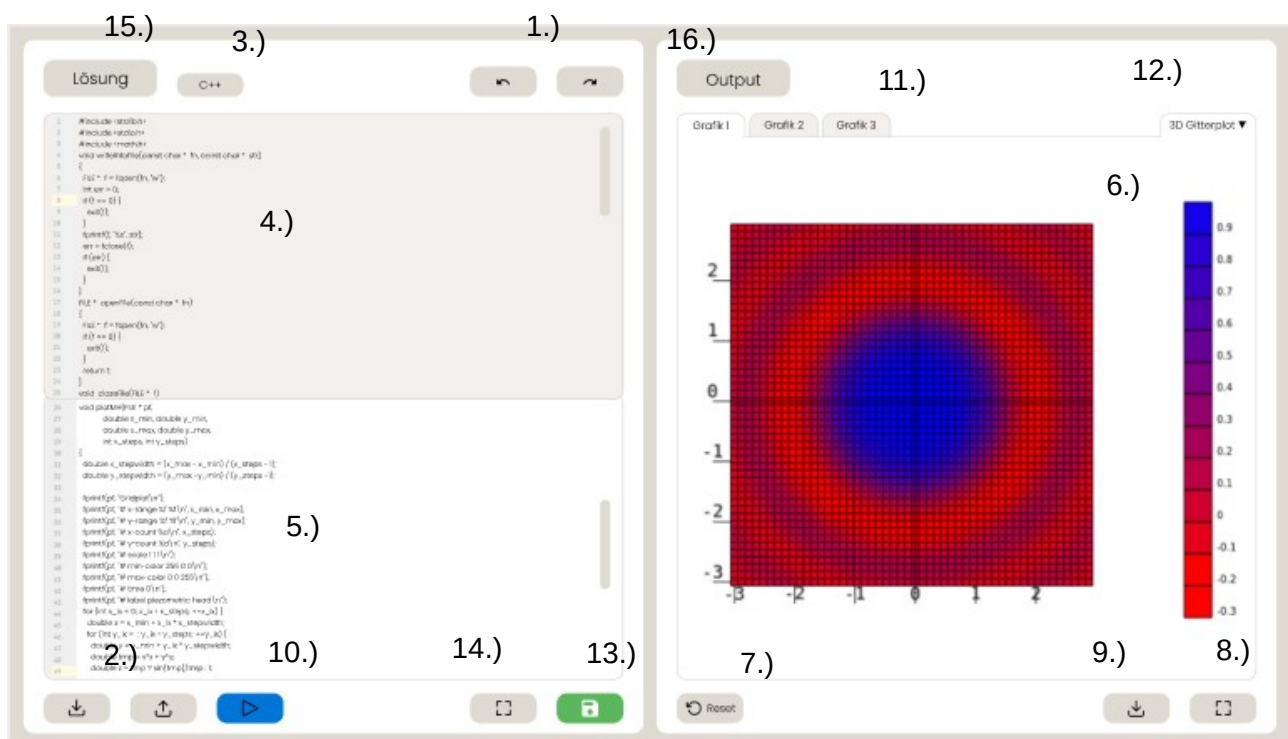


Abbildung 14: Die Startseite von VipLab - VipLab View Student 3

1.) Die Funktionen "Wiederherstellen" und "Rückgängig": Die Icons wurde wie im Ist-Zustand beibehalten, weil keine Unterschiede zum Material Design Standard von Google festgestellt werden konnte. Jedoch wurde die Positionierung der Icons im Lösungsbereich auf oben rechts geändert. Damit wurde die Positionierung der allgemeinen Konvention angepasst. In Google Docs sind die Funktionen ebenso oberhalb angeordnet.

2.) *Die Funktion Quellcode „Hochladen“ und „Runterladen“:* Das Disketten-Icon ist den typischen Upload und Download-Icons gewichen und die Funktionen wurden unten links im Lösungsbereich angeordnet.

3.) *Anzeige der Programmiersprache:* Ein entsprechendes Label wurde im Lösungsbereich oben links angeordnet. Das Element ist dabei nicht anklickbar und soll nur informieren.

4.) *Anzeige der Codevorgabe:* Dieser Bereich ist schon wie im Ist-Zustand ausgegraut. Ausschließlich die beiden Inhaltsfelder im Lösungsbereich wurden näher zusammengeführt.

5.) *Eingabefeld der Lösung:* Das Eingabefeld für den Code ist im Vergleich zum Ist-Zustand unverändert geblieben.

6.) *Anzeige der Visualisierung (Output-Bereich):* Der Output-Bereich ist im Vergleich zum Ist-Zustand unverändert geblieben. Die Positionierung hat sich nicht verändert.

7.) *Die Funktion „Zurücksetzen“:* Für die Funktion wurde ein entsprechendes Icon aus dem Bootstrap - Standard ausgewählt, und es wurde zudem noch ein Label formuliert. Der Button ist im Output-Bereich unten rechts angeordnet.

8.) *Die Funktion „Herausziehen“ (Grafikfenster vergrößern):* Für die Funktion wurde ein entsprechendes Icon ausgewählt. Der Button ist im Output-Bereich unten links angeordnet.

9.) *Die Funktion „Plot herunterladen“:* Für die Funktion wurde ein entsprechendes Download-Icon aus dem Bootstrap - Standard ausgewählt und unten rechts im Output-Bereich angeordnet.

10.) *Die Funktion „Berechnung starten“:* Hierfür wurde ein entsprechendes Play-Icon ausgewählt und unten rechts im Lösungsbereich angeordnet. Der Button wurde blau eingefärbt und repräsentiert in der Anwendung den Primär Button.

11.) *Die Funktionen „Tabs ausgeben“:* Die Tabs wurden als Alternative zu den Buttons „Play“, „Next“ und „Last“ aus dem Ist-Zustand eingeführt.

12.) *Die Funktion „Grafik 2D/3D umschalten“:* Die Funktion ist oben rechts im Output-Bereich angeordnet.

13.) Die Funktion „Codelösung einreichen“: Für diese Funktion wurde das Disketten-Icon verwendet und im Lösungsbereich unter links angeordnet. Der Sekundärbutton ist grün eingefärbt.

14.) Die Funktion „IDE Fenster vergrößern“: Hiermit kann das IDE-Fenster groß geschaltet werden. Der Code wird u.a. vergrößert angezeigt.

15.) Label „Lösung“: Das Label „Lösung“ kennzeichnet den Lösungsbereich. (nicht anklickbar)

16.) Label „Output“: Das Label „Output“ kennzeichnet den Output-Bereich. (nicht anklickbar)

Eine wesentliche Neuerung ist, dass man jetzt in VipLab die IDE groß schalten kann. Die Bereiche, in denen der Code dargestellt ist, sind jetzt vergrößert auf der rechten Seite angeordnet. Auf der linken Seite ist ein Bereich freigehalten, in dem die Textaufgabe für Prüfungszwecke abgebildet werden kann. Anhand dieser Aufgabe fertigt der Student seine Lösung in VipLab an.

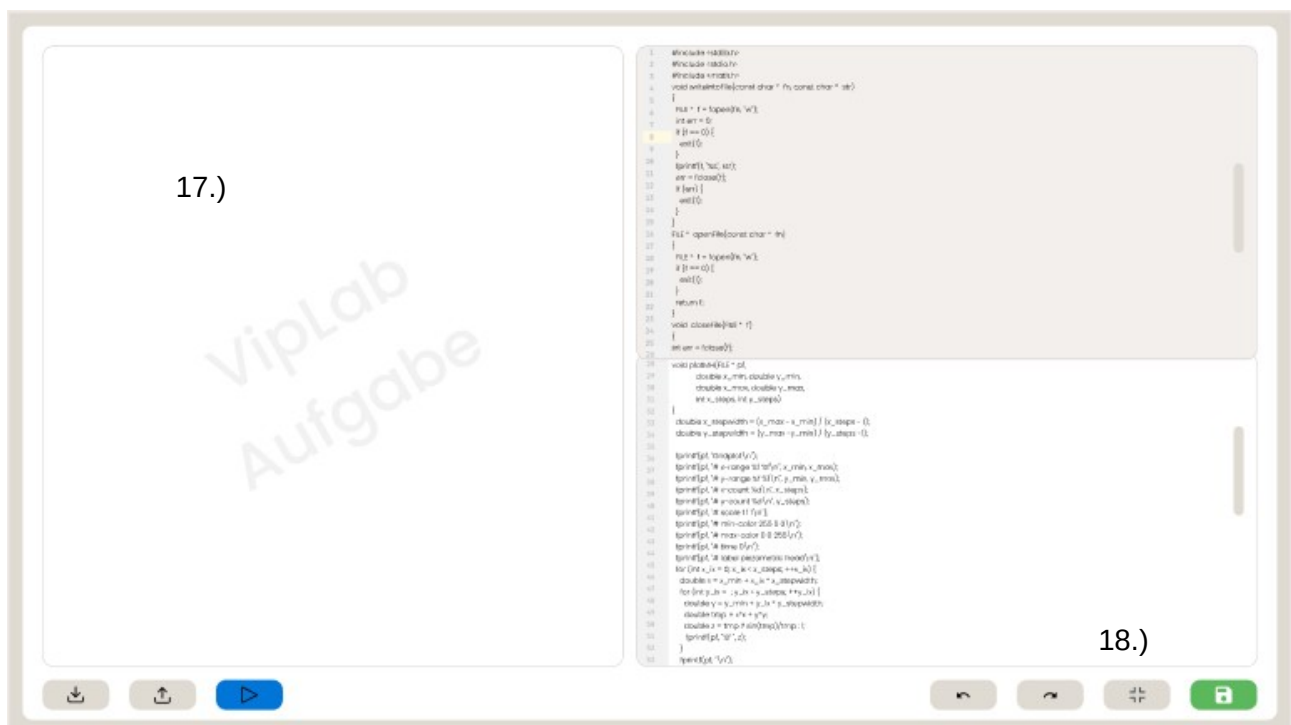


Abbildung 15: Die IDE vergrößert - VipLab View Student 9

17.) Die Funktion „VipLab Aufgabe anzeigen“: In der vergrößerten Ansicht der VipLab IDE ist zur linken Seite die Aufgabe für Prüfungszwecke abgebildet.

18.) Die Funktion „IDE Fenster verkleinern“: Mit der Funktion wird die vergrößerte IDE wieder in den verkleinerten Ursprungszustand zurückgesetzt.

Auch der Output-Bereich kann vergrößert werden, um sich beispielsweise Grafiken anzeigen zu lassen. In der Ansicht stehen wieder Tabs zur Verfügung, um zwischen mehreren Outputs wechseln zu können (Grafik 1+2). Des Weiteren kann in der Ansicht eine Grafik resettet werden und ein Download der Grafik ist ebenfalls möglich.

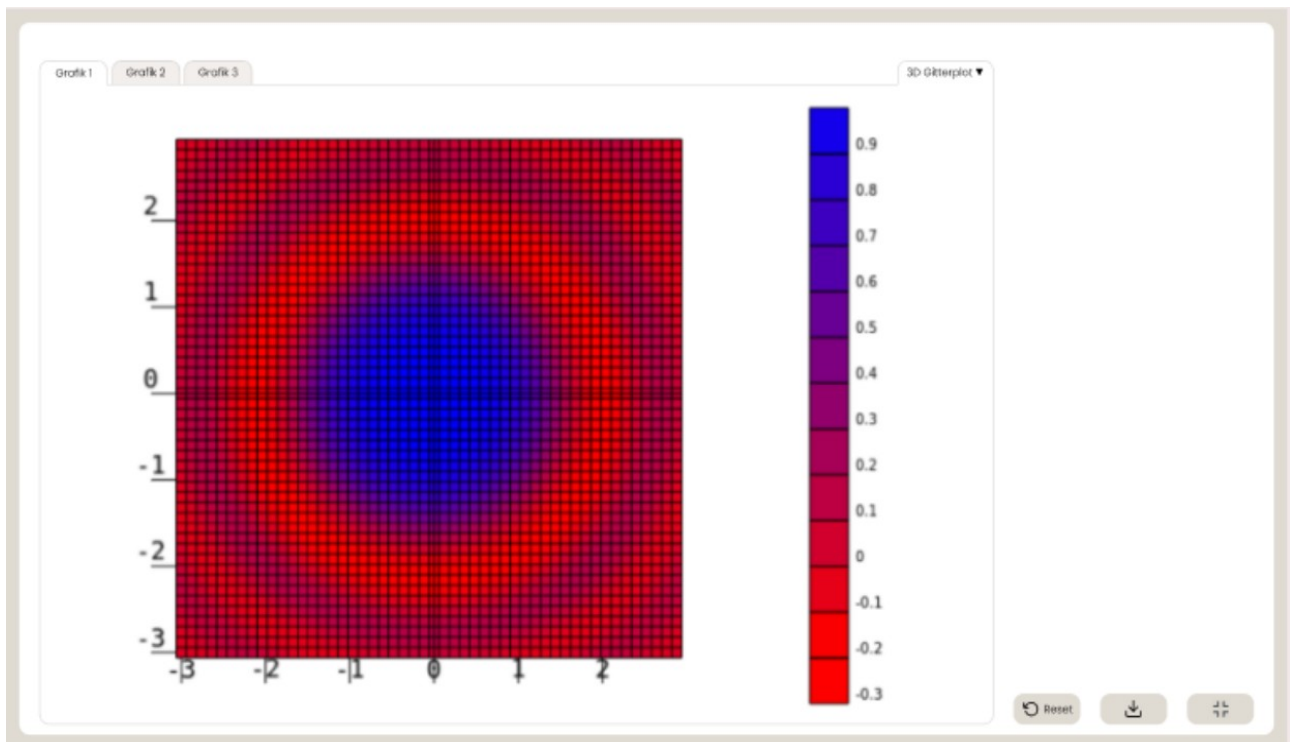


Abbildung 16: Der Output-Bereich vergrößert - VipLab View Student 7

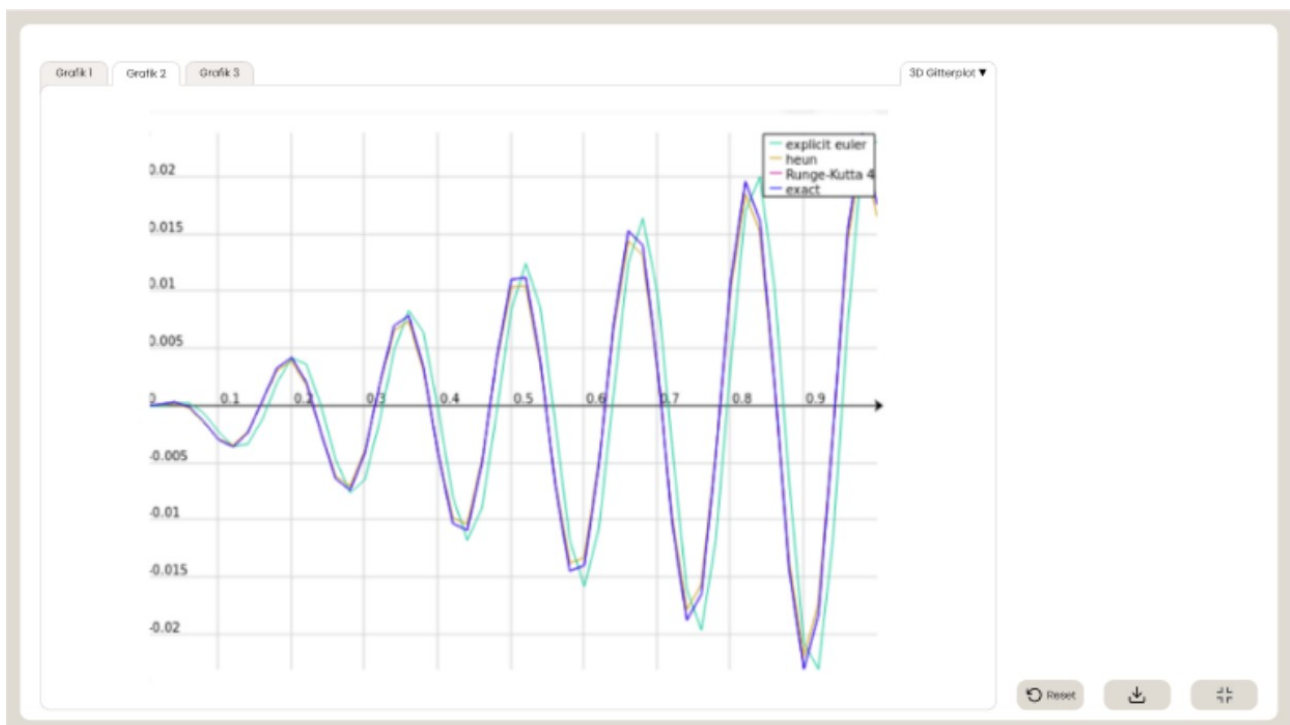


Abbildung 17: Der Output-Bereich vergrößert 2 - VipLab View Student 8

Zwischen den Tabs kann auch in der Startseite hin und hergewechselt werden. In dem zweiten Tab (Grafik 2) ist es möglich mehrere Dateien herunterzuladen. Der Code auf der linken Seite hat im Vorfeld diese Dateien erzeugt. Der Nutzer muss seine Download-Option nur noch mit einem Kreuz in der jeweiligen Datei bestätigen und das Download-Icon ausführen.

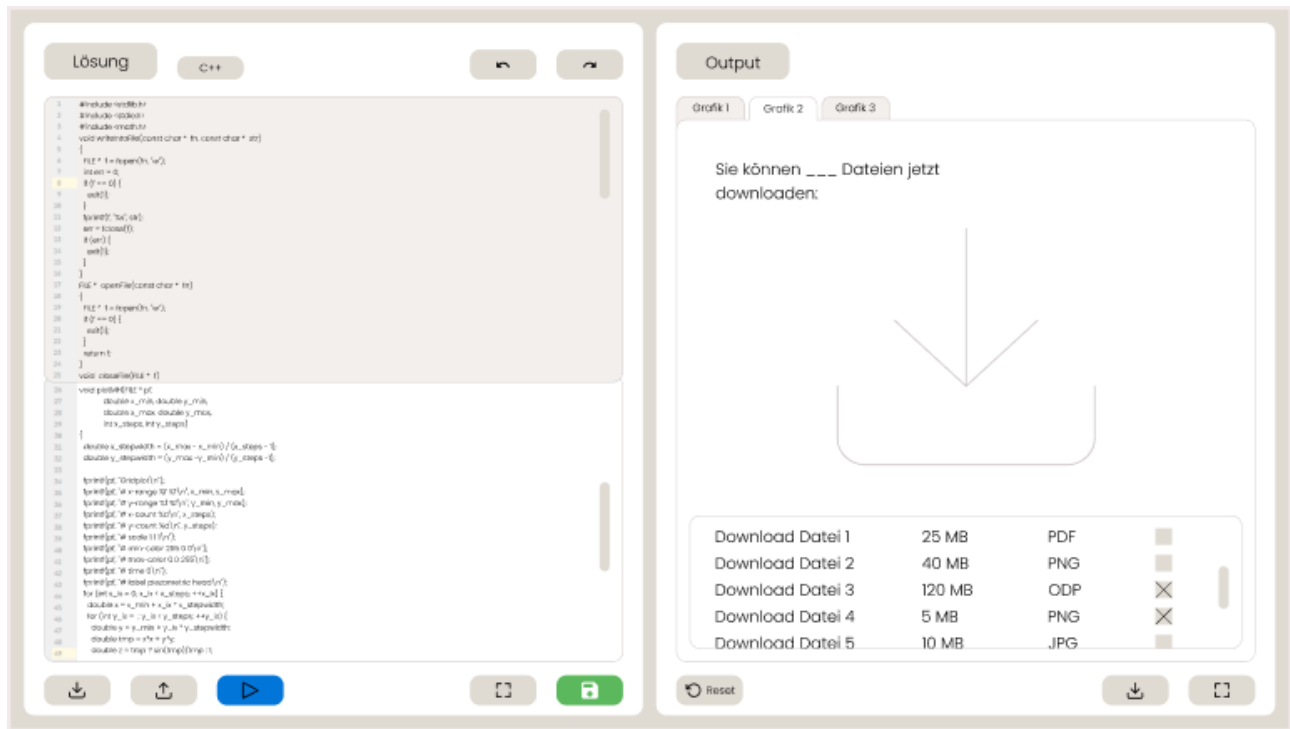


Abbildung 18: Das Tab für mehrere Downloads - VipLab View Student 4

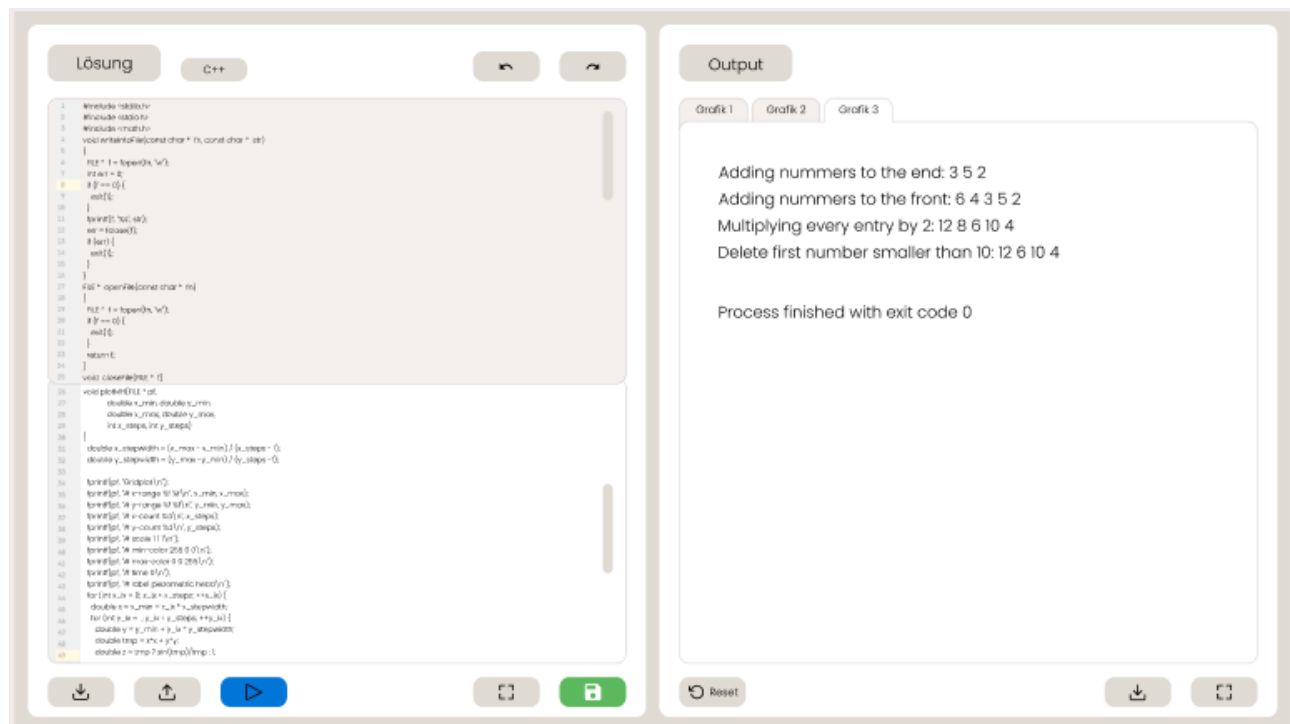


Abbildung 19: Textzeilen als Output im Tab - VipLab View Student 5

Auf der Startseite kann auch eine Grafik von einer 2D auf eine 3D Ansicht umgestellt werden.

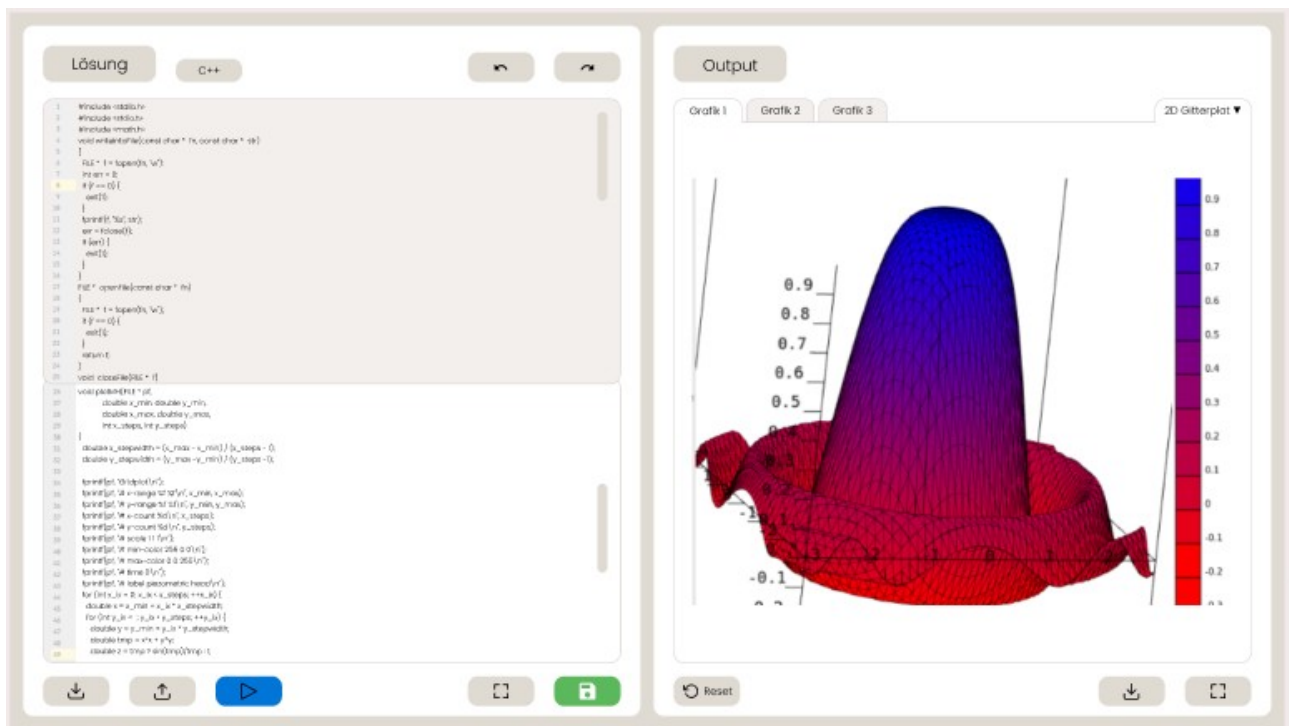


Abbildung 20: Die Grafik umgeschaltet auf die 3D Ansicht - VipLab View Student 10

Im nächsten Kapitel wird aufgezeigt, mit welcher Erhebungsmethode die beiden Prototypen validiert wurden. Es wird zudem erläutert, warum in der Bachelorarbeit diese Methode gewählt und kein anderes Verfahren angestrebt wurde. Abschließend werden noch die Ergebnisse aus den Erhebungen aufgezeigt und qualitativ erläutert. Mit diesen Erkenntnissen wurden die Prototypen nochmals mit weiteren Use Cases und anderen Verbesserungen für den Nutzer überarbeitet.

## 7 Die Testingphase

### 7.1 Die Theorie zu den Usability-Tests

Ein Usability-Test ist die effektivste wissenschaftliche Methode, um die Gebrauchstauglichkeit eines Prototyps in Bezug auf mögliche Probleme und Hindernisse im Nutzungsumgang zu untersuchen. Im Test werden die Schwierigkeiten in der Bedienung deutlich erkennbar und können wissenschaftlich dokumentiert werden. Mit den daraus gewonnenen Erkenntnissen können die Schwierigkeiten (Usability - Probleme) beseitigt werden und der Nutzer kann seine Absichten in der späteren Anwendung schneller und effizienter erreichen. Mit jedem Testdurchlauf, den ein Prototyp durchläuft, wird die Usability und die User Experience weiter verbessert und der Prototyp wird in der Konzeption weiter optimiert. Damit wird die Zufriedenheit des Nutzers mit der

Anwendung weiter gestärkt. Der Nutzer ist allgemein glücklicher in der Bedienung der gesamten Anwendung. Ein Usability-Test sollte immer so früh wie möglich mit in die Entwicklungsarbeit integriert werden. In jeder Projektphase kann ein Test mit den Nutzern durchgeführt werden, um vorzeitig entstehenden Usability – Problemen vorzubeugen. Am häufigsten werden diese Tests in einem Laborumfeld mit den Probanden durchgeführt.<sup>44</sup>

Für die Projektarbeit wurden auch Usability-Tests angestrebt, um eine nutzerzentrierte Entwicklungsarbeit zu gewährleisten. Diese Tests wurden mit Hilfe von High Fidelity Prototypen durchgeführt. Ein zu testender Prototyp wurde dabei speziell den Bedürfnissen der Studenten angepasst und ein anderer denen der Forscher. Je Prototyp wurden 3 – 5 Probanden zum Test herangezogen, um die Usability zu testen.

Für die Usability-Tests wurden Probanden ausgewählt, die den typischen Nutzerprofilen von VipLab entsprechen. Diese Nutzerprofile können theoretisch auch den erstellten Personas aus der Analysephase entnommen werden. In der Projektarbeit hat sich jedoch ergeben, dass die Personas rückblickend aufgrund der fehlenden Datengrundlage erstellt werden mussten. Es konnte sich somit nicht auf die Personas bezogen werden, um die typische Nutzerschaft von VipLab für die Usability-Tests zu identifizieren.

Die Universität Stuttgart kannte natürlich schon im Vorfeld der Bachelorarbeit ihre typische Nutzerschaft. Daher war im Voraus klar, dass für Studenten und Forscher die Usability-Tests mit den erstellten Prototypen durchgeführt werden. Es ist für die Bachelorarbeit somit nicht zwingend notwendig gewesen, Personas auszuarbeiten. Die Probanden für die Usability-Tests stellte die Universität Stuttgart zur Verfügung.

### **7.1.1 Der Inhalt eines Usability-Tests**

In den Usability-Tests werden typische Nutzungsszenarien mit den Probanden in Bezug auf die VipLab - Prototypen durchgespielt. Dazu werden Standardaufgaben gestellt, deren Ergebnisse und Erfahrungen mit den Prototypen in einem qualitativen Fragebogen festgehalten werden. In dem qualitativen Fragebogen wird während des Tests auch durch den Prototyp druchnavigiert, und es werden an verschiedenen Stellen Anweisungen zur Interaktion gegeben, die nach der Erledigung, mit den entsprechenden Fragen beantwortet werden müssen. Beispielsweise, ob die ausgeführte Interaktion (Funktion oder Use Case) in der späteren Anwendung überhaupt benötigt wird. Diese

---

<sup>44</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 203-204



Fragen können u. a. in einer Intervallskala beantwortet werden. In den Tests wird eine qualitative Einschätzung der Use Cases durch die Probanden angestrebt. (*Anhang 3 + 4*)

Das Studiendesign eines Usability-Tests kann frei gestaltet werden. Die Aufgaben wie auch der gesamte Fragebogaufbau, kann individuell an die speziellen Anforderungen, die der Prototyp erfüllen soll, angepasst werden. Ein Usability-Test kann noch mit weiteren quantitativen Datenerhebungsverfahren zusätzlich zu den qualitativen kombiniert werden, wie beispielsweise mit Eyetracking oder Klicktracking (Messen des Klickverhaltens). Mit dem Klicktracking wird gemessen, wie viele Klicks, und damit Zeit, ein Proband benötigt, um seine Absicht in einer Anwendung (dem Prototyp) zu erreichen.<sup>45</sup>

Für die Bachelorarbeit werden keine quantitativen Methoden zusätzlich mit in das Studiendesign aufgenommen. Die beiden VipLab – Prototypen sind in ihrer Komplexität nicht anspruchsvoll genug, um ein Klicktracking zu rechtfertigen. Alle Funktionen (Use Cases) können in den Prototypen über einen Klick erreicht werden. Bei komplexeren Anwendungen, in denen einige Funktionen schwerer zu erreichen sind, und damit mehr Klicks benötigen, machen quantitative Methoden wiederum mehr Sinn. In der Bachelorarbeit wurden auch keine Hypothesen zu einer speziellen Forschungsfrage aufgeworfen, daher muss nicht zwingend etwas gemessen werden.

### **7.1.2 Die Wahl zur Art des Usability-Tests**

Ein Usability-Test kann je nach Rahmenbedingungen unterschiedliche Formate annehmen. Entweder kann der Test in einem Labor unter moderierten Bedingungen stattfinden oder er kann unmoderiert/moderiert von zu Hause über eine Onlineanwendung absolviert werden. Letzteres wird als ein Remote-Usability-Test bezeichnet. In der Regel finden die meisten Test unter moderierten Laborbedingungen statt als von daheim aus. Das hat natürlich auch seine Gründe.

Moderierte Usability-Tests im Labor bringen den Vorteil mit sich, dass sie tiefgreifendere Einblicke als ein Remote-Usability-Test erlauben. Während des Tests kann beispielsweise der Proband zu seinem Vorgehen in der Anwendung (dem Prototyp) hinterfragt werden und es können neue Erkenntnisse zu seinen Verhaltensweisen gewonnen werden. Zudem kann im Laborversuch die Mimik und Gestik genau beobachtet werden und es können Rückschlüsse in Bezug auf den Prototyp zugelassen werden. In der Moderation können auch immer noch spontane Anschluss- und Zusatzfragen gestellt werden, wenn sich entsprechende Testsituationen ergeben.

---

<sup>45</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 204-205

Das sind Vorteile die ein Remote-Usability-Test nicht gänzlich erfüllen kann, weil kein Interviewer physisch den Test mit begleitet. Dafür sind meistens diese unmoderierten Tests günstiger und schneller in der Durchführung als Laborversuche. Die Tests können auch mit schwer zu erreichenden Testpersonen absolviert werden.

Die Wahl zur Usability-Test-Form (Labor oder Remote?) kann jedoch davon abhängig gemacht werden, wie komplex ein zu testender Prototyp ist. Ein eher einfacher Prototyp benötigt im Testdurchgang weniger einer Erläuterung zu den einzelnen Funktionsweisen. Die Aufgabenstellungen im Fragebogen sind dadurch leichter verständlich für den Probanden, um diese in die Praxis umzusetzen. Wenn in einem Testszenario klare Fragestellungen oder Aufgaben gegeben sind, die dem Probanden wenig Interpretationsspielraum lassen, dann können unmoderierte Remote-Usability-Tests eine gute Möglichkeit zur Erhebung sein.<sup>46</sup> In einem unmoderierten Remote-Usability-Test müssen die Fragen sehr genau formuliert sein, damit der Proband ohne Rückfragen im Test reagieren kann. Ein erklärungsbedürftiger Prototyp hingegen muss meistens vor dem Test durch den Interviewer umfangreicher erläutert werden, und daher ist der moderierte Test im Labor besser zur Erhebung geeignet.

Für die Bachelorarbeit wurden unmoderierte asynchronische Remote-Usability-Tests eingesetzt. Diese sind darauf ausgerichtet, den Test im virtuellen Raum ortsungebunden und autonom mit den Probanden durchzuführen. Dafür wird eine entsprechende Onlineanwendung für die Umsetzung eingesetzt. Die Probanden bekommen die Anwendung auf dem eigenen Computer bereitgestellt und können in ihrer gewohnten Umgebung (wie zum Beispiel am Arbeitsplatz) den Test absolvieren. Das Testdatum wie auch die benötigte Testzeit legt dabei jeder Proband individuell für sich selber fest.

Dieses Testverfahren wurde gewählt, weil es sich einerseits bei den VipLab – Prototypen um eine einfache selbsterklärende Anwendung handelt, die kein moderiertes Labor-Szenario benötigt und andererseits die aktuelle Lage keine physischen Treffen im Labor erlaubt.

### **7.1.3 Die Durchführung von Remote-Usability-Tests**

Der Proband ruft den Prototyp online auf und absolviert nacheinander die vorgegebenen Interaktionsaufgaben. Nach jeder Interaktionsaufgabe werden danach die dafür eingesetzten Funktionen gleich von dem Probanden validiert. Nach Beendigung aller Interaktionsaufgaben werden dem Probanden noch abschließend Fragen zum Nutzungskontext gestellt. Während der

---

46 Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 206

Bearbeitungszeit zeichnet die Software alle Mausklicks und Navigationspfade auf. Die durchschnittliche Testdauer für einen asynchronen Remote-Usability-Test ist geringer als bei einem moderierten Testdurchlauf. Bei einem Remote-Usability-Test sollte man sich insgesamt nur auf drei bis fünf Interaktionsaufgaben mit einer Testdauer von 15 – 30 Minuten pro Test beschränken.<sup>47</sup>

## 7.2 Die Auswertungen der Usability-Tests für die Prototypen

Nach der Testingphase wurde als erstes der Forscher-Prototyp mit dessen Benutzeroberfläche ausgewertet. Das ist darauf zurückzuführen, dass als erstes in der Projektarbeit die Forscher als Probanden zu Verfügung standen. Die Studenten als zweite Versuchsgruppe wurden ein paar Wochen später zu ihrem Prototyp befragt. Für beide Versuchsgruppen wurde angestrebt bis zu fünf Probanden zu gewinnen. Die Auswertungen der Usability-Tests hat zum Einen gezeigt, welche Funktionen in der Benutzbarkeit gut verstanden werden und welche noch Usability - Probleme aufweisen. Es wurde des Weiteren erfragt, ob die Funktionen überhaupt eine Relevanz für den Nutzer im Alltag haben. Funktionen, die der Nutzer überwiegend als überflüssig in einer Anwendung einschätzt, können gänzlich wieder aus der Entwicklungsarbeit entfernt werden. Die Befragung hat zudem gezeigt, welche weiteren Funktionen sich der Nutzer in VipLab wünscht.

### 7.2.1 Der Prototyp Forscher

In der Auswertung des Forscher-Prototyp hat sich gezeigt, dass alle dargestellten Funktionen (Use Cases) im Allgemeinen benötigt werden. Keine Funktion wurde unterdurchschnittlich als schlecht bewertet. Mittels einer Intervallskala von 1 bis 10 Punkten (unwichtig – wichtig) wurde die Relevanz aller Use Cases gemessen. In der nachfolgenden Tabelle sind alle zentralen Use Cases mit den entsprechenden Bewertungen aufgelistet.

Die Funktion (der Use Case)	Punkte (5 Probanden)	Relevanz in der Anwendung
Konfigurationsdatei hochladen	34 von 50	mittelmäßig - wichtig
Bild resettten	38 von 50	wichtig
Output-Fenster groß/klein schalten	34 von 50	mittelmäßig - wichtig
Bilddatei herunterladen	46 von 50	wichtig
Konfigurationsdatei umschalten	40 von 50	wichtig
Datei undo/redo	40 von 50	wichtig
Konfigurationsdatei herunterladen	42 von 50	wichtig
Output als Tabs darstellen	42 von 50	wichtig (gefällt)

*Tabelle 2: Die Auswertung des Forscher-Prototyp (Die Funktionen / Use Cases)*

<sup>47</sup> Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag. Seite 226

In der Betrachtung der Tabelle (Tabelle 2) wird ersichtlich, dass die Use Cases „Output-Fenster groß/klein schalten“ und „Konfigurationsdatei hochladen“ relativ niedrig im Vergleich zu den anderen Funktionen bewertet wurden. Die Funktion „Konfigurationsdatei hochladen“ kann unter anderem weniger benötigt werden, weil ein Forscher häufiger die notwendigen Daten zur Visualisierung von DaRUS direkt in die Anwendung exportiert, als lokal Daten zu importieren. An der Stelle ist zu überlegen, ob die beiden Use Cases wieder aus dem Prototyp entfernt werden müssen. Jedoch haben die Funktionen auch einen deutlichen Mehrwert für die User Experience. Nur ist dieser Mehrwert für die Nutzer nicht am wichtigsten. In einer internen Diskussion mit dem Auftraggeber wird dazu geraten, die Funktionen nochmals zu besprechen und zu überdenken. Bis zu diesem Zeitpunkt bleiben die Funktionen aber in dem Prototyp mit enthalten.

Der neu erdachte Use Case „Konfigurationsdatei umschalten“ wurde gut angenommen. Das Rating ist deutlich positiv ausgefallen mit 40 von den insgesamt 50 Punkten. Der Use Case kann somit in dem Prototyp erhalten bleiben. Auch die neue Darstellungsform des Outputs in Form von Tabs wurde sehr positiv aufgenommen. Die Bewertung liegt bei 42 von den 50 Punkten.

Die übrigen alten Funktionen aus dem Ist-Zustand wurden auch in der Beurteilung nahe den 40 Punkten bewertet. Am wichtigsten ist den Nutzern dabei, dass eine Bilddatei heruntergeladen werden kann. Die alten Funktionen werden somit noch in VipLab weiterhin benötigt und können in der Konzeption erhalten bleiben.

Das Design der Anwendung konnten die Probanden ebenso im Test beurteilen. Erfragt wurde, ob die einzelnen Tabs im Design sowie auch die gesamte Anwendung gefallen. Zudem wurde mit einer Frage die User Experience erfragt. Die untere Tabelle listet alle erhobenen Ergebnisse auf.

Das Gestaltungselement	Punkte (5 insgesamt mit Probanden)	Gefallen
Die verschiedenen Tabs	38 von 50	gut
Die gesamte Anwendung	40 von 50	gut
Hat die Anwendung in der Bedienung Spaß gemacht?	5 mal	ja

Tabelle 3: Die Auswertung des Forscher-Prototyp (Die Gestaltung / Das Design)

Die Darstellung der Tabs hat den Probanden überwiegend gut gefallen. Jedoch ist das Design kein besonderes. An dieser Stelle könnte vielleicht noch eine schönere Gestaltung gefunden werden. Die gesamte Anwendung wurde überraschend positiv im Design beurteilt und die Benutzung hat allen Probanden Spaß gemacht.

In den Interaktionsaufgaben haben die Probanden ebenso im Allgemeinen gut abgeschnitten. Alle Interaktionsaufgaben konnten allgemein gut gelöst werden und die dafür eingesetzten Icons wurden verstanden. Die untere Tabelle listet die Ergebnisse auf und beschreibt zusammengefasst das Probandenverhalten während der jeweiligen Interaktionsaufgabe (Tabelle 4).

Interaktionsaufgabe	Allgemeine Beurteilung (insgesamt mit 5 Probanden)	Usability
Upload and Play	Die Aufgabe hat allgemein gut funktioniert. Jedoch gab es Fehlinteraktionen, weil die Testumgebung anfänglich ungewohnt für die Probanden war. Zudem wurde oft versucht mit den nicht klickbaren Labels die Aufgabe zu lösen.	mittelmäßig – gut (siehe Anhand 5)
Reset something	Die Aufgabe hat sehr gut funktioniert. Alle Probanden konnten die Aufgabe lösen.	gut
Download a graphic	Die Aufgabe mit den Funktionen wurde allgemein überwiegend gut verstanden. Nicht immer haben die Probanden das Output-Fenster vergrößert. Das hat dann folglich zu Lösungsproblemen in der Aufgabe geführt.	mittelmäßig - gut
Toggle a screen	Die Aufgabe konnten einige Probanden gut lösen, jedoch gab es auch einige, die sich schwer damit getan haben. Diese haben dann versucht die Aufgabe mittels der nicht anklickbaren Labels zu lösen, was nicht der richtige Ansatz war. Die Undo- und Redo-Funktion wurde aber immer richtig angewandt und verstanden.	Mittelmäßig (siehe Anhand 5)
Download a konf.-file	Die Aufgabe konnten alle Probanden problemlos verstehen und lösen.	gut
Download multiple files	Die Aufgabe konnten alle Probanden problemlos verstehen und lösen.	gut

Tabelle 4: Die Auswertung des Forscher-Prototyp (Die Interaktionsaufgaben)

Zusammenfassend kann gesagt werden, dass allgemein alle Interaktionsaufgaben in dem Prototyp gut funktioniert haben. Es war nur auffällig, dass bei Anwendungsschwierigkeiten immer auch im Label-Bereich nach Lösungen gesucht wurde. Diese Elemente sind nicht anklickbar und deshalb ist es unnötig in diesem Bereich nach Bedienungslösungen zu suchen. Das Usability – Problem ist vielleicht darauf zurückzuführen, dass die Labels noch viel zu ähnlich im Vergleich zu den interaktiven Buttons aussehen. Beide Gestaltungselemente sind zwar in der Form unterschiedlich, jedoch ist die Einfärbung bisher im selben Grauton gehalten. In der Überarbeitung des Prototyps muss die Hintergrundfarbe zu den Labels unbedingt noch eindeutiger verändert werden. Das Gestaltungsgesetz zur Ähnlichkeit muss an der Stelle noch weiter herausgearbeitet werden.


Ein weiteres Usability - Problem hat sich in der Interaktion mit dem Toggle-Icon für den Use Case „Konfigurationsdatei umschalten“ gezeigt. Einige Probanden konnten das Icon nicht richtig dem

Use Case zuordnen. Das Icon wurde unter Umständen falsch interpretiert (*Anhang 5*). Das Toggle-Icon kann auch mit Ein- und Ausschalten in Verbindung gebracht werden. Der Anhang 5 zeigt das Usability - Problem mit der Aufgabe. In der Überarbeitung des Prototyps muss unbedingt noch ein zusätzliches Label zu dem Icon formuliert werden, um besser verstanden werden zu können.

Diese beiden Erkenntnisse aus den Usability-Tests wurden in den nächsten Arbeitsschritten in die Überarbeitung des Prototyps mit aufgenommen. Damit wurden weitere Verbesserungen erreicht, um die User Experience weiter zu verbessern (*Anhang 6*). Im Anhang 6 ist der überarbeitete Prototyp einsehbar und auf Seite 76 ist der Link zur interaktiven Version verfügbar.

### Die Persona zu der Nutzergruppe Forscher

In der Auswertung des Nutzerkontextes für die Nutzergruppe Forscher konnten jetzt folglich auch Personas aus der Datengrundlage abgeleitet werden. Dabei ist besonders herausgearbeitet, in welchem Kontext der Nutzer die Anwendung einsetzt und welche Bedürfnisse/Wünsche er an diese stellt.



Name: Thorsten Mayer  
Alter: 40  
Tätigkeit: Forscher bei Max Planck

Bedürfnisse an VipLab:

- Erzeugung einfacher Visualisierungen
- Einfache Konfiguration von Visualisierungen (Parameter variieren)
- Anzeige verschiedener Grafikdarstellungsformen (auch über andere Visualisierungstools)
- Visualisierung läuft auch in einem zweiten Browserfenster
- Einbindung verschiedener Backends mit entsprechender Parameteranpassung
- Anzeige von Systeminteraktionen
- Output-Bereiche gibt Filme aus
- Einsicht in den Source Code mit Download

Einsatzgebiete von VipLab:

- In der Lehre und Bildung (Demonstrationszwecke)
- In der Forschung (Nachnutzung u. Replizierbarkeit)
- In der Recherche (Ausprobieren von Simulationen)
- In der Softwareentwicklung (z.B. Einbindung in Apps)

Wünsche an VipLab:

- VipLab ist Open Source
- Verknüpfung zu DaRUS
- Einfach Integration weiterer Software

Abbildung 21: Die Persona Thorsten Mayer (eigene Darstellung)

Die Persona liefert nun auch Informationen, aus denen weitere zentrale Use Cases gewonnen werden können, die für mehr User Experience in der Anwendung sorgen. Beispielsweise ist ein Wunsch, dass eine Verknüpfung zwischen VipLab und dem Repository DaRUS dargestellt ist.

Das ist ein weiterer Use Case, der in die Überarbeitung des Prototyps mit einfließen kann. Ansonsten sind schon die meisten Bedürfnisse, die aus der Persona abgeleitet werden können, mit implementiert. Der Prototyp enthält somit nach der ersten Iteration schon die meisten Anforderungen, die bis zu dem Zeitpunkt ermittelt werden konnten.

### 7.2.2 Der Prototyp Student

In der Auswertung des Studenten-Prototyp hat sich gezeigt, dass alle dargestellten Funktionen (Use Cases) ebenso im Allgemeinen benötigt werden. Keine Funktion wurde unterdurchschnittlich als schlecht bewertet (< 15). Mittels einer Intervallskala von 1 bis 10 Punkten (unwichtig – wichtig) wurde die Relevanz aller Use Cases gemessen. In der nachfolgenden Tabelle sind alle zentralen Use Cases mit den entsprechenden Bewertungen aufgelistet.

Die Funktion (der Use Case)	Punkte (4 Probanden)	Relevanz in der Anwendung
Quellcode hochladen	31 von 40	wichtig
Bild resettet	29 von 40	wichtig
Output-Fenster groß/klein schalten	21 von 40	mittelmäßig
Bilddatei herunterladen	25 von 40	mittelmäßig - (wichtig)
Datei undo/redo	34 von 40	wichtig
Quellcode herunterladen	32 von 40	wichtig
Output als Tabs darstellen	29 von 40	wichtig (gefällt)
IDE-Fenster groß/klein schalten	26 von 40 (65%)	wichtig (gefällt)
Codelösung einreichen	29 von 40	wichtig (gefällt)
VipLab Aufgabe anzeigen	19 von 40	mittelmäßig - schlecht

*Tabelle 5: Die Auswertung des Studenten - Prototypen (Die Funktionen / Use Cases)*

In der Betrachtung der Tabelle (Tabelle 5) wird ersichtlich, dass die Use Cases „Output-Fenster groß/klein schalten“ und „VipLab Aufgabe anzeigen“ relativ niedrig (im Bereich von 20 Punkte) im Vergleich zu den anderen Funktionen bewertet wurden. Der Use Case „Output-Fenster groß/klein schalten“ wurde schon im Forscher-Prototyp mit am schlechtesten bewertet. Warum das so ist, sollte nochmals mit den Probanden persönlich geklärt werden. Die Funktion „VipLab Aufgabe anzeigen“ ist überraschenderweise auch schlecht validiert wurden. An der Stelle wäre es auch nochmals ratsam persönlich die Funktion mit den Probanden zu hinterfragen. Beispielsweise wäre auch ein Ansatz, eine Vorort-Beobachtung mit den Probanden zu diesem Use Case durchzuführen. Die Ausgangsfrage wäre dabei, wie der Student die Aufgabe konkret in VipLab löst und wie sein genaues Vorgehen dabei ist? Dementsprechend wird die Darstellungform angepasst.

Bei diesen beiden Use Cases muss aufgrund der schlechten Bewertung auch wieder hinterfragt werden, ob diese in dem Prototyp implementiert bleiben. Jedoch haben die Funktionen auch einen deutlichen Mehrwert für die User Experience. Nur ist dieser Mehrwert für die Nutzer nicht am wichtigsten. In einer internen Diskussion mit dem Auftraggeber wird dazu geraten, die Funktionen nochmals zu besprechen und zu überdenken. Bis zu diesem Zeitpunkt bleiben die Funktionen aber in dem Prototyp mit enthalten.

Die neu erdachten Use Cases „IDE-Fenster groß/klein schalten“ und „Codelösung einreichen“ wurden unerwartet gut angenommen. Das Rating ist deutlich positiv ausgefallen mit 26 von 40 und 29 von 40 Punkten. Die Use Cases können somit in dem Prototyp erhalten bleiben. In der Auswertung der Tests hat sich zudem gezeigt, dass sich einzelne Probanden sogar einen separaten Button zum Speichern und Einreichen von Abgaben wünschen. Es wäre somit in VipLab möglich, mehrere Male während der Bearbeitung einen Code zwischenspeichern. Nach Beendigung der Aufgabe wird dann folglich nur einmal das Ergebnis über einen weiteren Button eingereicht. Das ist ein weiterer zusätzlicher Use Case, der in der Überarbeitung des Prototyps berücksichtigt werden kann. Auch die neue Darstellungsform des Outputs in Form von Tabs wurde sehr positiv aufgenommen. Die Bewertung liegt bei 29 von den insgesamt 40 Punkten.

Die übrigen alten Funktionen aus dem Ist-Zustand wurden in der Beurteilung nahe den 30 Punkten bewertet. Die alten Funktionen werden somit auch noch in VipLab weiterhin benötigt und können in der Konzeption erhalten bleiben. Ausschließlich eine Bilddatei muss nicht unbedingt heruntergeladen werden können. Der Use Case wurde mit 25 Punkten durchschnittlich bewertet.

Das grafische Design konnten die Probanden wiederum auch bei dem Studenten-Prototyp beurteilen. Die untere Tabelle listet alle erhobenen Ergebnisse auf.

Das Gestaltungselement	Punkte (4 Probanden)	Gefallen
Die verschiedenen Tabs	24 von 40	mittelmäßig - gut
Die gesamte Anwendung	21 von 40	mittelmäßig
Hat die Anwendung in der Bedienung Spaß gemacht?	4 mal	ja

Tabelle 6: Die Auswertung des Studenten-Prototyp (Die Gestaltung / Das Design)

Die Darstellung der Tabs hat den Probanden meistens mittelmäßig gefallen. Das kann vielleicht darauf zurückzuführen sein, dass das Design kein besonderes im Vergleich zu anderen Anwendungen ist. An der Stelle kann vielleicht noch eine schönere Gestaltung gefunden werden.



Die gesamte Anwendung wurde im Design auch eher als mittelmäßig beurteilt. Jedoch hat die Benutzung der Anwendung allen Probanden ausnahmslos Spaß gemacht.

In den Interaktionsaufgaben haben die Probanden ebenso im Allgemeinen gut abgeschnitten. Alle Interaktionsaufgaben konnten allgemein gut gelöst werden und die dafür eingesetzten Icons wurden verstanden. Die untere Tabelle listet die Ergebnisse auf und beschreibt zusammengefasst das Probandenverhalten während der jeweiligen Interaktionsaufgabe (*Tabelle 7*).

Interaktionsaufgabe	Allgemeine Beurteilung (4 Probanden)	Usability
Upload and Play	Die Aufgabe hat allgemein gut funktioniert. Jedoch haben einige Probanden nach Lösungen im Label-Bereich gesucht. Einige Probanden hatten Startschwierigkeiten mit der Aufgabe und haben erst mal alle Funktionen getestet.	(allgemein) gut
Reset something	Die Aufgabe hat gut funktioniert. Die Probanden konnten das passende Icon zum Use Case interpretieren. Jedoch haben einige Nutzer versucht, mit dem Tab-Wechsel die Grafik zu resettet. Das ist nicht falsch, die Aufgabe gelöst.	gut
Download a graphic	Die Aufgabe mit den Funktionen wurde sehr gut verstanden und umgesetzt. Nur ein Proband hat das Output-Fenster nicht für die Aufgabe vergrößert. Die Download-Funktion wurde dennoch verstanden.	gut
Undo and Redo function	Die Probanden konnten die Undo/Redo - Funktionen sehr gut interpretieren. Jedoch hat ein Proband für den Quellcode-Download das Speicher-Icon verwendet. Ein deutliches Usability - Problem. Alle anderen Probanden konnten die Aufgabe ohne Probleme lösen.	gut
Download multiple files	Die Aufgabe konnten alle Probanden problemlos verstehen und lösen. Es hat keine Schwierigkeiten gegeben.	gut
Submit a task	Ein Proband konnte die gesamte Aufgabe erfolgreich lösen. Das Icon zum Einreichen der Aufgabe wurde sofort verstanden. Jedoch konnten alle anderen Probanden das richtige Icon (die Diskette) nicht sofort interpretieren und haben mit dem Play-Icon stattdessen überwiegend interagiert. Diese Probanden hatten deutliche Usability-Probleme die Aufgabe zu lösen. Die IDE groß zuschalten hat jeder geschafft.	mittelmäßig - schlecht (Icon zum Einreichen wurde nicht verstanden.) (siehe Anhang 7)

*Tabelle 7: Die Auswertung des Studenten-Prototyp (Die Interaktionsaufgaben)*

Zusammenfassend kann gesagt werden, dass allgemein alle Interaktionsaufgaben in dem Prototyp gut funktioniert haben. Zum Einen war wieder auffällig, dass im Label-Bereich nach Lösungen für die Interaktionsaufgaben gesucht wurde. Das sind nicht anklickbare Elemente, die zur Lösungsfindung nicht mit beitragen können. An der Stelle müssen die Elemente, wie schon im Forscher-Prototyp, auch überarbeitet werden. Die Hintergrundfarbe wird deutlich abgeändert.

Das signifikanteste Usability – Problem in Bezug auf den Prototyp hat sich aber in der Interaktion mit dem Speicher-Icon gezeigt. Nahezu alle Probanden konnten den Use Case „Codelösung einreichen“ nicht dem richtigen Icon (der Diskette) zuordnen. Stattdessen haben die Probanden versucht, die Interaktionsaufgabe mit dem Play-Icon zu lösen (*Anhang 7*). Das ist natürlich die falsche Vorgehensweise gewesen. Das Disketten-Icon wird überwiegend zum Speichern von Inhalten eingesetzt. Die Probanden konnten nicht die Funktion „Einreichen“ damit assoziieren. Für die Überarbeitung des Prototyps werden zwei separate Buttons gestaltet, mit denen die beiden Funktionen (Speichern und Einreichen) eindeutig voneinander abgrenzt werden können.

Wie schon erwähnt, haben einige Probanden vorgeschlagen, für das Speichern und Einreichen zwei unterschiedliche Buttons zu gestalten. Das Disketten-Icon würde bei diesem Lösungsansatz, dann kein weiteres Label benötigen, da die Funktion eindeutig wäre (Die Diskette wird eindeutig mit Speichern assoziiert). Hingegen muss für den Use Case „Codelösung einreichen“ noch ein neues Icon recherchiert und in den Prototyp implementiert werden.

Diese beiden Erkenntnisse aus den Usability-Tests wurden in den nächsten Arbeitsschritten in die Überarbeitung des Prototyps mit aufgenommen. Damit wurden weitere Verbesserungen erreicht, um die User Experience weiter zu verbessern (*Anhang 8*).

### Die Persona zu der Nutzergruppe Student



Name: Sabine Müller  
Alter: 27  
Tätigkeit: Studentin an der Uni Stuttgart

#### Bedürfnisse an VipLab:

- Der Eingabebereich in der EDI ist auch in Tabs oder in einer anderen Darstellungsform eingeteilt, die beliebig in der Größe angepasst werden können.
- Die VipLab Aufgabe (der Code) kann auf die eigene lokale IDE zur Bearbeitung importiert werden. Abschließend kann der fertig bearbeitete Code zurück in VipLab exportiert werden.
- Zuverlässigkeit, einfache Bedienbarkeit und Plattformunabhängigkeit
- Einfache Möglichkeit zur automatischen Code-Korrektur.
- Einen separaten Button zum einreichen und speichern von Abgaben.

#### Einsatzgebiete von VipLab:

- In der Lehre und Bildung (als Übungsumgebung oder zur Gestaltung von bewertbaren Programmier- und Übungsaufgaben)
- In der Lehre als Prüfungsmedium (bewertbare Programmieraufgaben absolvieren)

#### Wünsche an VipLab:

- Über ILIAS ist VipLab weiterhin schnell abrufbar.
- VipLab ist allgemein übersichtlicher gestaltet (zum Beispiel bei der Aufgabenerstellung).
- VipLab lässt sich zukünftig zuverlässiger bedienen (weniger fehleranfällig!).

Abbildung 22: Die Persona Sabine Müller (eigene Darstellung)

In der Auswertung des Nutzerkontextes für die Nutzergruppe Student konnten jetzt folglich auch Personas aus der Datengrundlage abgeleitet werden. Dabei ist besonders herausgearbeitet, in welchem Kontext der Nutzer die Anwendung einsetzt und welche Bedürfnisse/Wünsche er an diese stellt.

Die Persona liefert nun auch Informationen, aus denen weitere zentrale Use Cases gewonnen werden können, die für mehr User Experience in der Anwendung sorgen. Beispielsweise ist es ein Bedürfnis, den Eingabebereich der IDE in Tabs oder ähnliches einzuteilen, die in der Größe individuell angepasst werden können. Im Ist-Zustand von VipLab ist eine individuelle Anpassung der Eingabefelder derzeit nur beschränkt möglich. Die Nutzer wünschen sich für die Eingabefelder mehr individuelle Einstellungsmöglichkeiten. Das ist auch eine Anforderung aus der Norm zur Entwicklung von nutzerzentrierten Anwendungen, dass individuelle Einstellungen gefördert werden müssen. Für die Überarbeitung des Prototyps kann somit ein weiterer zentraler Use Case abgeleitet und implementiert werden (*Anhang 8*).

## **8 Die Zusammenfassung**

Anfänglich wurde in der Bachelorarbeit der aktuelle Ist-Zustand von VipLab ermittelt. Damit wurden auch die zentralen Use Cases herausgearbeitet, die wieder in der Konzeption für den Soll-Zustand eingesetzt wurden. In der Entwicklungsarbeit haben sich aber dennoch weitere Use Cases herausgebildet, die mit in die Entwicklungsarbeit eingeflossen sind. Darauffolgend wurde für alle Funktionen eine passende grafische Darstellungsform gestaltet. Angewendet wurden unter anderem Elemente wie Buttons, Icons, Labels oder Tabs. Diese Elemente sind in einem Wireframe erstmalig neu angeordnet wurden. Das Wireframe wurde in den Projektphasen zu mehreren Mockups bis hin zu einem fertigen High-Fidelity Prototyp weiterentwickelt. Es wurden in der Projektarbeit genau zwei Prototypen designt, die alle zentralen Funktionen mit enthalten. Dabei wurde eine Benutzeroberfläche der Studentenschaft angepasst und eine anderer dem Forscher. Für die Entwicklung und Gestaltung wurde die kostenfreie Onlineanwendung Figma eingesetzt.

Gegen Ende der Projektarbeit (in der Testingphase) wurden die Prototypen mit der typischen Nutzerschaft von VipLab validiert. Hierfür wurden unmoderierte Remote-Usability-Tests eingesetzt, die die Probanden von Zuhause aus absolvieren konnten. Je Prototyp wurden 4 – 5 Probanden zur Usability befragt. Als Tool für die Tests wurde die kostenpflichtige Onlineanwendung Maze herangezogen.

Die Testergebnisse haben gezeigt, dass die Prototypen an manchen Stellen noch Usability-Probleme aufweisen. An den Stellen wurde nochmals nachgebessert und es wird empfohlen, die Prototypen nochmals erneut auf die Usability mit den Nutzer zu validieren.

Die beiden unteren Links führen zu den überarbeiteten Prototypen, die für den Studenten und den Forscher entworfen wurden.

Link zum Prototyp Forscher:

<https://www.figma.com/proto/wqLTBHHeNSxJ1RcWVyA1oY/Bachelorarbeit-VipLab?node-id=1275%3A1052&scaling=contain>

Link zum Prototyp Student:

<https://www.figma.com/proto/wqLTBHHeNSxJ1RcWVyA1oY/Bachelorarbeit-VipLab?node-id=1417%3A753&scaling=contain>

## **Das Quellenverzeichnis**

Bootstrap (2019). Bootstrap Icons. Verfügbar unter:

<https://icons.getbootstrap.com/>. Letzter Zugriff: 16.07.2020

Bootstrap (2019). Buttons. Verfügbar unter:

<https://getbootstrap.com/docs/4.0/components/buttons/>. Letzter Zugriff: 16.07.2020

Jacobsen, J. & Meyer, L. (2019) .Praxisbuch: Usability und UX (2. aktu. Aufl.). Bonn: Rheinwerk Verlag.

Material Design (2019). Icons. Verfügbar unter:

<https://material.io/resources/icons/?search=disk&style=baseline>. Letzter Zugriff: 16.07.2020

Moser, C. (2012) .User Experience Design: Mit erlebniszentrierter Softwareentwicklung zu Produkten, die begeistern. Berlin Heidelberg: Springer-Verlag.

Universität Stuttgart (2019). Forschungs- und Entwicklungsprojekte. Verfügbar unter:

<https://www.tik.uni-stuttgart.de/forschung-und-lehre/forschungs-und-entwicklungsprojekte/>.

Letzter Zugriff: 16.07.2020

Wikipedia (2020). Bootstrap (front-end framework). Verfügbar unter:

[https://en.wikipedia.org/wiki/Bootstrap\\_\(front-end\\_framework\)](https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)). Letzter Zugriff: 16.07.2020