Softwareprojekt Wintersemester 2020/2021

am Fachgebiet Software Engineering, Leibniz Universität Hannover

Projektname: WB-Editor-1

`SWP-WS2021-WB-Editor-1-Spec-v01.pdf`

Vorgelegt

am 06.11.2020
von WB-Editor-1

Ausführende:

Nachname	*Vorname*	*Rolle*
Burmeister	Chris	Projektleiter
Jurisch	Lars	Qualitätsbeauftragter
Tulainov	Aleksei	Entwickler
Linnemann	Juri	Entwickler
Hallmann	Kai	Entwickler
Mathes	Max	Entwickler

Das Dokument enthält

☒ Die Anforderungen aus Kundensicht (User Requirements)

☒ Anforderungen, wie das zu System zu gestalten ist (System Requirements)

Datum, Unterschrift des Projektleiters, auch für die anderen Projektangehörigen

Kunden-Bewertung

Der Kunde, Herr Schneider, bestätigt mit seiner Unterschrift, diese Anforderungsspezifikation erhalten, geprüft und für inhaltlich ☐ **in Ordnung** | ☐ **weitgehend in Ordnung** | ☐ **deutlich zu verbessernd** | ☐ **nicht akzeptabel** befunden zu haben.

Datum, Unterschrift des Kunden; evtl. Vermerk.

Inhaltsverzeichnis

1 Mission des Projekts

1.1 Erläuterung des zu lösenden Problems

Wir erstellen einen Editor für eine virtuelle und graphische Darstellung von Gebäuden. Der Nutzer kann in diesem Editor sowohl echte Gebäude nachzeichnen als auch neue Gebäude entwerfen. Weiterhin kann der Nutzer den Räumen verschieden Typen (Besprechungsraum, Büro, Halle, eventuell weitere) zuordnen.
In einem erstellten Gebäude können sich Personen in einem der Räume aufhalten. Die Räume (eventuell bis auf die Halle) haben dabei jeweils eine begrenzte Anzahl an Plätzen.
Außerdem können in einem Raum auch Dokumente dargestellt werden.
Ein im Editor entworfenes Gebäude soll nicht nur ein Bild sein, sondern eine semantische Oberfläche bieten, mit der interagiert werden kann.

1.2 Wünsche und Prioritäten des Kunden

Es folgen die Wünsche des Kunden, nach absteigender Priorität geordnet:

- Wunsch 1:
  Gebäude können gezeichnet, gespeichert und bearbeitet werden.
- Wunsch 2:
  Einzelnen Räumen können Typen (Büro etc.) zugeordnet werden.
- Wunsch 3:
  Fertige Gebäude können visuell dargestellt werden.
- Wunsch 4:
  Personen können in Räumen dargestellt werden und sich bewegen bzw. bewegt werden.
- Wunsch 5:
  In einem Raum können Dokumente liegen und angesehen werden.
- Wunsch 6:
  Eventuell sollen weitere Raumtypen abgesehen von Büro, Besprechungsraum und Halle hinzugefügt werden.
- Wunsch 7:
  Eventuell soll das Gebäude von außen betrachtet werden können.

1.3 Domänenbeschreibung

Das Gesamtprodukt VirtuHoS soll es erleichtern, online zusammenzuarbeiten.
Falls das Softwareprojekt im nächsten Jahr ebenfalls online stattfindet, soll dieses Projekt dabei assistieren. Zum Beispiel soll man sich dann virtuell im "InfoLab" treffen und dort in virtuellen Arbeitsräumen am Softwareprojekt arbeiten können.
Der Editor-Teil des Projektes ist dabei dafür verantwortlich, dass Gebäude erstellt, bearbeitet und dargestellt werden können.

1.4 Maßnahmen zur Anforderungsanalyse

Der Kunde stellte seine ursprünglichen Anforderungen und Wünsche in einer Einführungsveranstaltung mithilfe einer zweiseitigen Projektbeschreibung vor. Außerdem finden wöchentlich einstündige Kundengespräche statt, in denen er seine aktuellen Ideen präsentiert. Dort stellen auch die Entwicklerteams, einschließlich uns, ihre Fortschritte vor, schlagen dem Kunden ihre Ideen vor und stellen Fragen dazu, was er sich genau vorstellt. Zudem haben wir verschiedene GUI-Designs entworfen, um sie miteinander zu vergleichen und ein möglichst gutes auszuwählen.

2 Rahmenbedingungen und Umfeld

2.1 Einschränkungen und Vorgaben

Wir sollen die Software als Java-Anwendung schreiben und uns mit den anderen Gruppen auf Schnittstellen einigen. Der Editor soll trotzdem eigenständig laufen können, indem zum Beispiel Mockups für die Schnittstellen erstellt werden. Die Designs der einzelnen Komponenten müssen ausschließlich eigene Grafiken und keine eventuell urheberrechtlich geschützten Grafiken sein.

2.2 Anwender

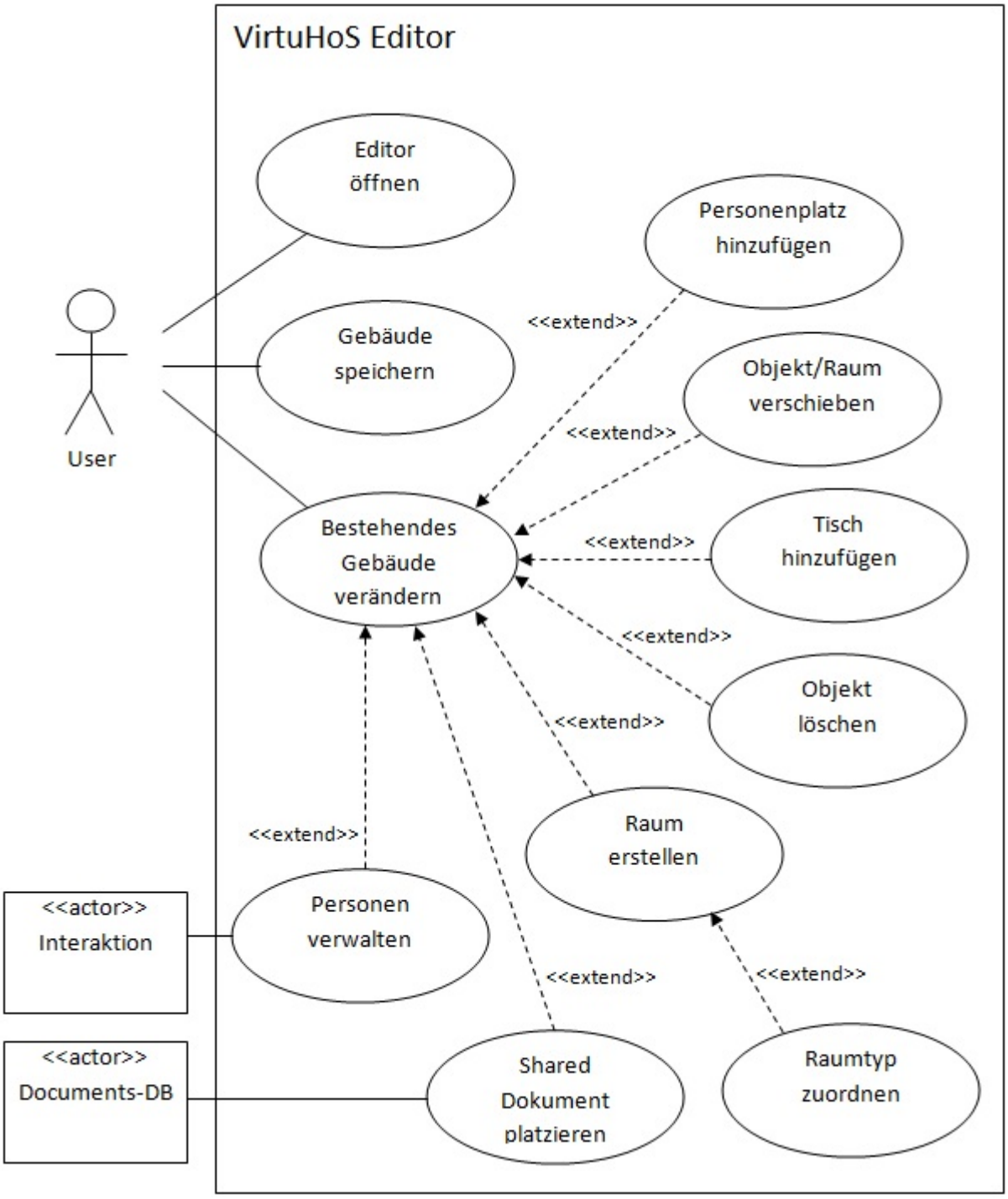
Die Software soll vorwiegend von Studierenden der Informatik und anderen Personen, von denen gewisse Grundkenntnisse über Software erwartet werden können, verwendet werden. Das Zeichnen von Gebäuden im Editor soll hauptsächlich von den Administratoren genutzt werden. Die anderen Anwender sollen nur beschränkte Interaktionsmöglichkeiten haben.

2.3 Schnittstellen und angrenzende Systeme

Informationen über die Position, die Größe und den Typ der Räume eines Gebäudes werden über Schnittstellen den anderen Projektteilen zur Verfügung gestellt. Der Editor soll in der Lage sein, die Gebäude einschließlich grundlegender Einrichtung, Personen und Dokumente darzustellen. Die Position dieser Elemente soll über Schnittstellen an die anderen Teilprojekte weitergegeben werden können. Außerdem soll der Editor eine Schnittstelle bereitstellen, mit der Personen sowohl innerhalb von Räumen als auch zwischen Räumen bewegt werden können.
Eventuell sollen weitere Schnittstellen hinzugefügt werden, um als Reaktion auf Userinputs die Funktionalitäten anderer Teilprojekte aufzurufen und deren Auswirkungen darzustellen.

3 Funktionale Anforderungen

3.1 Use Case-Diagramm



3.2 Use Case-Beschreibungen

3.2.1 UC: Editor Öffnen

Use Case Nr. 01	**Editor Öffnen**
Erläuterungen	Nachdem das Programm gestartet wurde, kann zum Erstellen eines neuen virtuellen Gebäudes die Editorfunktion verwendet werden.
Status	Das System ist gestartet und der Nutzer eingeloggt.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Der Nutzer ist eingeloggt.
Mindestgarantie	Das Programm befindet sich im Hauptmenü oder Editormodus. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Der Programm befindet sich im Editormodus.
Stakeholder	Der Nutzer will den Editor öffnen, um Gebäude zu erstellen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer klickt auf das ”Gebäude erstellen” Symbol.
Hauptszenario	1. Der Nutzer klickt auf das ”Gebäude erstellen” Symbol. 2. Das System öffnet den Editormodus, dieser ist leer und es kann ein neues Gebäude angelegt werden.
Erweiterung	–
Priorität	unverzichtbar
Verwendungshäufigkeit	selten

Erläuterungen und Details

- Das ”Gebäude erstellen” Symbol wird noch festgelegt.
- Anschluss an weitere Use Cases: siehe unten

3.2.2 UC: Einzelnen Raum zeichnen

Use Case Nr. 02	**Raum erstellen**
Erläuterungen	Im Editormodus kann ein Gebäude gezeichnet werden. Wird die "Raum zeichnen" Funktion ausgewählt, so kann man einen neuen Raum zeichnen.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Das System befindet sich im Editormodus und es wurde momentan noch kein anderer Raum gezeichnet.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume gelöscht oder bewegt. Es werden keine Raumtypen geändert. Es werden keine Objekte erstellt, gelöscht oder bewegt. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Der Raum wurde erstellt.
Stakeholder	Der Nutzer will einzelne Räume zeichen, um das Original Gebäude möglichst echt nachzustellen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer wählt die "Raum zeichnen" Funktion aus.
Hauptszenario	1. Der Nutzer klickt auf das "Raum zeichnen" Symbol. 2. Das System markiert das Symbol, um zu zeigen, dass die Funktion aktiv ist. 3. Der Nutzer zeichnet auf der Zeichenfläche einen neuen Raum. 4. Das System speichert Größe und Position des Raums. 5. Der Nutzer kann den Raum auswählen und ihn mit der Maus auf dem Grid an die gewünschte Stelle verschieben. 6. Das System passt die Position des Raums an und zeigt ihn an der neuen Position.
Erweiterung	3a. WENN außerhalb der Zeichenfläche eine andere Funktion angeklickt wird, DANN beendet sich die "Raum zeichen" Funktion; das Symbol ist nicht mehr markiert. 5a. WENN die Größe eines platzierten Raumes geändert werden soll, DANN kann man ihn auswählen und die Größe ändern.
Priorität	unverzichtbar
Verwendungshäufigkeit	regelmäßig

Erläuterungen und Details

3.2.3 UC: Raumtyp zuordnen

Use Case Nr. 03	**Raumtyp zuordnen**
Erläuterungen	Ein im Editormodus gezeichneter Raum kann einem Raumtyp zugewiesen werden (Büro, Besprechungsraum, Halle). Dem Raumtyp entsprechend werden unterschiedliche Funktionen für den Raum verfügbar.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Es ist bereits ein Raum gezeichnet.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume erstellt, gelöscht oder bewegt. Es werden keine Objekte erstellt, gelöscht oder bewegt. Es werden keine gespeicherten Gebäude gelöscht. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Der Raumtyp ist wie gewünscht geändert worden.
Stakeholder	Der Nutzer will den Räumen unterschiedliche Typen zuordnen, um unterschiedliche Arten der Interaktion zu ermöglichen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer wählt den gezeichneten Raum aus.
Hauptszenario	1. Der Nutzer wählt im Editormodus einen Raumtypen aus und klickt auf den Raum, den er zu dem ausgewählten Raumtyp ändern möchte. 2. Das System speichert den zugewiesenen Raumtyp und zeigt diesen an.
Erweiterung	1a. WENN bereits eine Raumtyp zugewiesen ist und der neue Raumtyp valide ist, DANN wird der Raumtyp überschrieben. 1b. WENN bereits eine Raumtyp zugewiesen ist und der neue Raumtyp nicht valide ist, DANN bleibt der Raumtyp gleich und es erscheint eine Fehlermeldung.
Priorität	unverzichtbar
Verwendungshäufigkeit	regelmäßig

Erläuterungen und Details

- Die Art und Weise des Kenntlichmachens des Raumtyps wird noch entschieden.

3.2.4 UC: Personenplatz hinzufügen

Use Case Nr. 04	**Personenplatz hinzufügen**
Erläuterungen	Im Editormodus können dem gezeichnetem Gebäude Personenplätze hinzugefügt werden. Diese Plätze können in der späteren Verwendung von Nutzern eingenommen werden.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Es ist bereits ein Raum gezeichnet.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume erstellt oder gelöscht. Es werden keine Raumtypen geändert. Es werden keine Objekte gelöscht.
Erfolgsgarantie	Der Personenplatz wird an gewünschter Position erstellt.
Stakeholder	Der Nutzer will Personenplätze hinzufügen, um die späteren Interaktionen zu organisieren und das Original Gebäude möglichst echt nachzustellen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer klickt auf das ”Personenplatz hinzufügen” Symbol.
Hauptszenario	1. Der Nutzer klickt auf das ”Personenplatz hinzufügen” Symbol. 2. Das System markiert das Symbol um zu zeigen, dass die Funktion aktiv ist. 3. Der Nutzer platziert den Personenplatz durch Klicken an der gewünschten Position. 4. Das System speichert die Position des Personenplatzes.
Erweiterung	3a. WENN die gewünschte Position nicht in einem passenden Raumtyp liegt, DANN erscheint eine passende Fehlermeldung.
Priorität	unverzichtbar
Verwendungshäufigkeit	regelmäßig

Erläuterungen und Details

3.2.5 UC: Objekt/Raum verschieben

Use Case Nr. 05	**Objekt/Raum verschieben**
Erläuterungen	Im Editormodus können in dem gezeichnetem Gebäude Räume und Objekte verschoben werden.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Es existieren bereits ein Raum und ein Objekt.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume oder Objekte erstellt oder gelöscht. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Die Position des Raums/Objekts wird verändert.
Stakeholder	Der Nutzer will Objekte/Räume verschieben, um beim Erstellen eines Gebäudes Änderungen vorzunehmen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer klickt auf einen bereits platzierten Raum/auf ein bereits platziertes Objekt.
Hauptszenario	1. Der Nutzer klickt auf einen bereits platzierten Raum/auf ein bereits platziertes Objekt. 2. Das System wählt den Raum/das Objekt aus. 3. Der Nutzer kann den Raum/das Objekt an die gewünschte Stelle auf dem Grid verschieben. 4. Das System speichert die neue Position des Raums/Objekts und zeigt diesen/dieses an der neuen Position an.
Erweiterung	3a. WENN die gewünschte Position nicht valide ist, DANN erscheint eine passende Fehlermeldung und das System wechselt in den Ausgangszustand.
Priorität	hoch
Verwendungshäufigkeit	selten

Erläuterungen und Details

3.2.6 UC: Tisch zeichnen

Use Case Nr. 06	**Tisch hinzufügen**
Erläuterungen	Im Editormodus können dem gezeichnetem Gebäude Tische hinzugefügt werden. Wird die "Tisch zeichnen" Funktion ausgewählt, so kann man anschließend in einem bestehenden Raum mit der Maus einen Tisch durch Ziehen eines Rechtecks erstellen. Der Tisch lässt sich anschließend mit der Maus auswählen und verschieben.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Es ist bereits ein Raum gezeichnet.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume erstellt, gelöscht oder bewegt. Es werden keine Raumtypen geändert. Es werden keine Objekte gelöscht. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Der Tisch wird dem Gebäude hinzugefügt.
Stakeholder	Der Nutzer will Tische zeichnen, um das Original Gebäude möglichst echt nachzustellen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer wählt die "Tisch zeichnen" Funktion aus.
Hauptszenario	1. Der Nutzer klickt auf das "Tisch zeichen" Symbol. 2. Das System markiert das Symbol, um zu zeigen, dass die Funktion aktiv ist. 3. Der Nutzer zieht auf der Zeichenfläche, innerhalb des Raumes, einen Tisch und kann die Größe anpassen. 4. Das System speichert die Größe und Position des Tisches. 5. Der Nutzer kann den Tisch auswählen und ihn mit der Maus auf dem Grid an die gewünschte Stelle verschieben. 6. Das System passt die Position des Tisches an.
Erweiterung	3a. WENN die Position nicht innerhalb eines Raumes liegt, DANN lässt sich kein Tisch erstellen. 5a. WENN Die Position nicht innerhalb eines Raumes liegt, DANN wird der Tisch abgewählt und bleibt in seiner ursprünglichen Position.
Priorität	mittel
Verwendungshäufigkeit	regelmäßig

Erläuterungen und Details

3.2.7 UC: Shared Dokument platzieren

Use Case Nr. 07	**Shared Dokument platzieren**
Erläuterungen	Im Editormodus können dem gezeichnetem Gebäude Ablagen für Shared Documents hinzugefügt werden.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Es ist bereits ein Tisch gezeichnet.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume erstellt, gelöscht oder bewegt. Es werden keine Raumtypen geändert. Es werden keine Objekte gelöscht. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Auf dem gewünschten Tisch wird eine Ablage für Shared Documents erstellt.
Stakeholder	Der Nutzer will shared Dokuments platzieren, um Möglichkeiten zur Zusammenarbeit zu bieten.
Hauptakteur	Nutzer
Auslöser	Der Nutzer klickt auf das ”Shared Documents Ablage hinzufügen” Symbol.
Hauptszenario	1. Der Nutzer klickt auf das ”Shared Documents Ablage hinzufügen” Symbol. 2. Das System markiert das Symbol, um zu zeigen, dass die Funktion aktiv ist. 3. Der Nutzer klickt auf einen Tisch, um dort eine Ablage für Shared Documents zu erstellen. 4. Das System speichert die Ablage und beendet die Funktion.
Erweiterung	3a. WENN der Nutzer nicht auf einen Tisch klickt, DANN erscheint eine passende Fehlermeldung und es wird keine Ablage erstellt.
Priorität	niedrig
Verwendungshäufigkeit	mittel

Erläuterungen und Details

3.2.8 UC: Objekt löschen

Use Case Nr. 08	**Objekt löschen**
Erläuterungen	Im Editormodus können bereits erstellte Räume und Objekte (Personenplätze, Tische, shared Dokument Zugriffe) wieder gelöscht werden.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Es befindet sich bereits ein Raum/ein Objekt im Gebäude.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume erstellt oder bewegt. Es werden keine Raumtypen geändert. Es werden keine Objekte erstellt oder bewegt. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Der gewünschte Raum/das gewünschte Objekt wird entfernt.
Stakeholder	Der Nutzer will Objekte löschen, um beim Erstellen eines Gebäudes Änderungen vorzunehmen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer wählt einen Raum/ein Objekt aus und klickt auf das ”Löschen” Symbol.
Hauptszenario	1. Der Nutzer wählt einen Raum/ein Objekt aus und klickt auf das ”Löschen” Symbol. 2. Das System fragt, ob der ausgewählte Raum und alle enthaltenen Objekte/das ausgewählte Objekt wirklich gelöscht werden sollen/soll. 3. Der Nutzer bestätigt das Löschen. 4. Das System löscht den Raum und die enthaltenen Objekte/das Objekt.
Erweiterung	3a. WENN der Nutzer das Löschen abbricht, DANN kehrt das System in den Ausgangszustand zurück.
Priorität	mittel
Verwendungshäufigkeit	selten

Erläuterungen und Details

3.2.9 UC: Gebäude speichern

Use Case Nr. 09	**Gebäude speichern**
Erläuterungen	Im Editormodus kann das Gebäude gespeichert werden, sobald die Erstellung abgeschlossen ist. Anschließend kann das Gebäude für Interaktions-Funktionalitäten genutzt werden.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editorsystem
Ebene	Editorfunktion
Vorbedingung	Es wurde ein Gebäude erstellt, dass der Nutzer speichern will.
Mindestgarantie	Es werden keine Räume erstellt, gelöscht oder bewegt. Es werden keine Raumtypen geändert. Es werden keine Objekte erstellt, gelöscht oder bewegt. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Das Gebäude wird im System gespeichert und das System kehrt zum Hauptmenü zurück.
Stakeholder	Der Nutzer will Gebäude speichern, um seinen Fortschritt abzusichern und das Gebäude zur Benutzung zu Verfügung zu stellen.
Hauptakteur	Nutzer
Auslöser	Der Nutzer klickt auf das ”Speichern” Symbol.
Hauptszenario	1. Der Nutzer klickt auf das ”Speichern” Symbol. 2. Das System fragt nach dem Namen des Gebäudes. 3. Der Nutzer gibt den gewünschten Namen an und bestätigt das Speichern. 4. Das System speichert das erstellte Gebäude.
Erweiterung	3a. WENN der Nutzer das Speichern abbricht, DANN bleibt das System im Editormodus des Gebäudes. 3b. WENN der Nutzer keinen Namen angibt, DANN erscheint eine passende Fehlermeldung. 3c. WENN der Nutzer einen bereits vergebenen Namen angibt, DANN wird er gefragt, ob er das Gebäude überschreiben will.
Priorität	unverzichtbar
Verwendungshäufigkeit	mittel

Erläuterungen und Details

3.2.10 UC: Bestehendes Gebäude verändern

Use Case Nr. 10	**Bestehendes Gebäude verändern**
Erläuterungen	Nachdem ein Gebäude erstellt wurde, kann es weiterhin bearbeitet werden.
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Der Nutzer ist eingeloggt und hat Zugriff auf ein gespeichertes Gebäude.
Mindestgarantie	Das Programm befindet sich weiterhin im Editormodus. Es werden keine Räume erstellt, gelöscht oder bewegt. Es werden keine Raumtypen geändert. Es werden keine Objekte erstellt, gelöscht oder bewegt. Es werden keine gespeicherten Gebäude gelöscht.
Erfolgsgarantie	Das System öffnet das gewünschte Gebäude im Editormodus.
Stakeholder	Der Nutzer will bestehende Gebäude verändern, um das Gebäude nach seinen Wünschen zu verändern.
Hauptakteur	Nutzer
Auslöser	Der Nutzer wählt ”Laden” und dann das gewünschte Gebäude aus.
Hauptszenario	1. Der Nutzer wählt ”Laden” und dann das gewünschte Gebäude aus. 2. Das System öffnet das ausgewählte Gebäude mit sämtlichen schon erstellten Räumen und Objekten und zeigt diese an. 3. Der Nutzer kann nun das Gebäude wie beim Erstellen bearbeiten.
Erweiterung	-
Priorität	mittel
Verwendungshäufigkeit	selten

Erläuterungen und Details

3.2.11 UC: Personen verwalten

Use Case Nr. 11	**Personen verwalten**
Erläuterungen	Interaktion kann Personen durch Signale in Editor verwalten
Status	Das System befindet sich im Editormodus eines Gebäudes.
Systemgrenzen (Scope)	Editor-Schnittstelle
Ebene	Hauptfunktion
Vorbedingung	Es existiert eine Gebäude mit mindestens einem Raum
Mindestgarantie	Der System-Zustand der sich nicht auf Personen bezieht bleibt unverändert.
Erfolgsgarantie	Der System-Zustand in Bezug auf Personen wird entsprechend dem Signal verändert.
Stakeholder	Die Interaktion will Personen in Editor verwalten.
Hauptakteur	Interaktion
Auslöser	Die Interaktion schickt ein Signal.
Hauptszenario	1. Die Interaktion schickt ein Signal um den Zustand einer Person zu verändern (hinzufügen, löschen, bewegen). 2. Das System ändert den Zustand dieser Person und zeigt es in GUI an.
Erweiterung	-
Priorität	niedrig
Verwendungshäufigkeit	mittel

Erläuterungen und Details

4 Qualitätsanforderungen

4.1 Qualitätsziele des Projekts

Am Ende des Projekts übergeben wir eine eigenständig laufende Software. Diese umfasst alle Aspekte aus dieser Spezifikation, aber im Nachhinein auch erweiterbar sein. Sie soll verständlich und einfach für den Benutzer sein. Der Code soll sinnvoll und leicht zu verstehen sein. Außerdem soll man eine einfache Installation anstreben, um den Benutzer nicht zu überfordern.

4.2 Qualitäts-Prioritäten des Kunden

Die Qualitätsziele sind wie folgt, mit nur leichter Priorisierung, absteigend angeordnet:

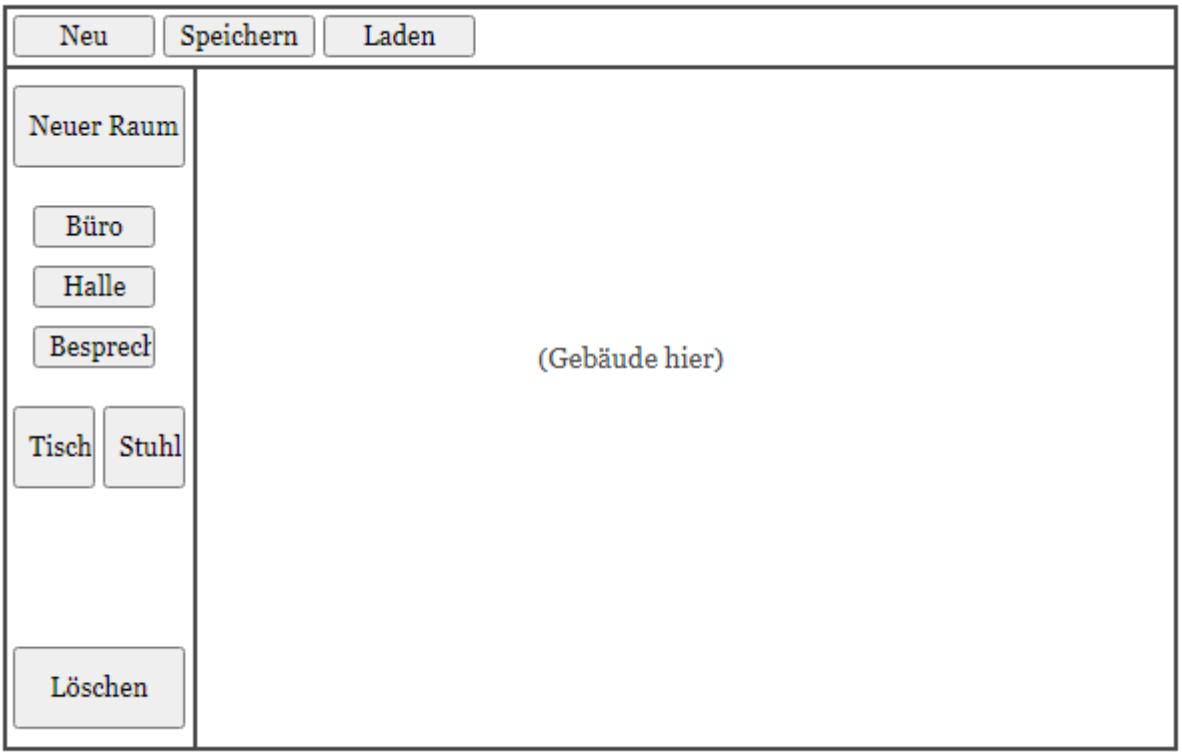
- Eigenständig laufende Software
- Leicht zu verstehender Code, für eine leichte Erweiterbarkeit durch Dritte
- Einfache Bedienung für den Benutzer
- Einfache Installation
- schlichte und klare optische Darstellung der Software
- Vollständigkeit der Software

4.3 Wie Qualitätsziele erreicht werden sollen

Die Qualitätsziele sollen durch ordentliches und gewissenhaftes Arbeiten erreicht werden. Außerdem soll durch häufiges Testen die Qualität auf hohem Niveau gehalten werden. Durch verständlichen Code, inklusive sinnvollen Kommentaren, soll das Finden von Fehlern vereinfacht und die Möglichkeit von Verbesserungen gewährleistet werden. Durch wöchentliche SCRUM-Meetings soll jeder auf dem aktuellen Stand des Projekts sein. Größere kompliziertere Fehler sollen dort zusammen diskutiert werden, um eine gemeinsame Lösung zu finden.

5 Hinweise zur Umsetzung

Das Programm soll auf der Programmiersprache Java basieren, wenn nötig dürfen aber Programmteile in anderen Sprachen eingebunden werden.
Genauer soll in JDK 15 gearbeitet werden.
Es kann davon ausgegangen werden, dass der zukünftige Benutzer kompetent im Umgang mit Technik ist.
Es soll nach dem Model-View-Controller-Pattern (MVC) gearbeitet werden.
GUI-Mockup:



6 Probleme und Risiken

1. WENN es entschieden wird, dass die gesamte GUI Teil des Editors ist,
   DANN haben wir viel mehr Arbeit, als wir anfangs erwartet haben.

2. WENN die Koordinierung mit den anderen Teilgruppen nicht ideal abläuft,
   DANN kann es dazu führen, dass das Programm nicht als Ganzes oder nur teilweise funktioniert.

3. WENN wichtige Programmteile nicht fertiggestellt werden,
   DANN funktioniert das Gesamtprogramm Editor nicht.

Abhilfe:

1. Um die gesamte GUI, falls nötig, zu implementieren, können gegebenenfalls optionale Funktionen des Editors nicht entwickelt werden.

2. Der Fokus liegt darauf, dass der Editor an sich funktioniert. Ein Zusammenwirken mit den anderen Teilprojekten zum Gesamtprojekt VirtuHoS hat eine niedrigere Priorität.

3. Die Funktionen mit hoher Priorität sollten vorrangig bearbeitet werden, sodass sämtliche wichtigen Programmteile am Ende funktionieren.

7 Optionen zur Aufwandsreduktion

7.1 Mögliche Abstriche

Mögliche Abstriche in unserem Projekt sind:

- Das Anzeigen von Dokumenten im Raum
- Eine Außendarstellung des Gebäudes
- Weitere Raumtypen neben den 3 Haupttypen
- Mehrere Stockwerke in einem Gebäude
- Zugangsbeschränkung von Räumen durch Name und Passwort
- Mehrere Leute auf einem Stuhl in einem Raum für größere Meetings
- Öffnen und Schließen der Tür eines Raumes, um den Status eines Meetings anzuzeigen

7.2 Inkrementelle Arbeit

7.2.1 1. Inkrement: Hauptfunktionen

- ständiges Entwickeln von User Stories aus der Spezifikation
- Erstellen einer Datenbank zum Speichern der Gebäude
- Programmieren der Grundstruktur eines Hauses
- Einbau eines Zeichenmodus
- Erstellung eines GUI zur Unterstützung der Hauptfunktionen
- Ermöglichung der Anzeige von Personen

7.2.2 2. Inkrement: Optionale Features

Hier werden alle Features vervollständigt, die im ersten Inkrement nicht fertiggestellt werden konnten. Je nach zur Verfügung stehender Zeit werden dann Features aus Absatz 7.1 hinzugefügt.

7.2.3 3. Inkrement: Polishing

Hier wird die fertige Software noch von kleinsten Fehlern bereinigt, damit sie problemlos an den Kunden übergeben werden kann. Am Ende dieses Inkrements wird die Software offiziell übergeben.

8 Glossar

Administratoren Personen mit besonderer Berechtigung

Anforderungsanalyse Mit der Anforderungsanalyse versucht man herauszufinden, welche Features sich der Kunde wünscht, um eine Software, die perfekt zu ihm passt, zu entwickeln.

Anzeigemodus Der Modus der Software, in dem das Gebäude nicht mehr bearbeitet wird, aber Personen im fertigen Gebäude angezeigt werden.

Applikation siehe Software

Code Programmiercode, der verwendet wird, um die Software zu erstellen

Datenbank Ein im Hintergrund einer Software laufendes Speichermedium, mit dem laufend Daten gespeichert und im Programm wiederverwendet werden können

Design Aussehen und Format

Domäne Voraussetzungen und Umgebung, in denen die Software am Ende laufen soll

Drag and Drop Verändern des aktuellen Zustands durch Verschiebung von Objekten bei gedrückter Maustaste

Editor Software zum Bearbeiten verschiedener Dinge, in unserem Fall Gebäude

Editorsystem Der Teil des VirtuHus-Programms in dem Räume, Tische, Personenplätze und shared Dokuments gezeichnet und ein Gebäude erstellt werden können

eingeloggt In einer Software angemeldet

Elemente Einzelteile

Feature Teil einer Software, welches eine bestimmte Aufgabe übernimmt

Funktion siehe Feature

Grafiken Darstellung eines Designs

Grid Rasterformat mit einheitlicher Größe

GUI Oberfläche einer Software, welche vom Nutzer bedient wird

Inkrement Zeitlicher Abschnitt in der Entwicklung einer Software; in diesem Projekt 3 Wochen

Installation Speicherung eines Programms auf einem Computer zur Vorbereitung einer Ausführung

Interaktionsmöglichkeit Möglichkeit, den Zustand eines Programms zu sehen und zu verändern

Java Programmiersprache

JDK 15 Version der Programmiersprache Java

Komponenten Einzelteile des Editors

Meetings Treffen für geschäftliche Besprechung

Mockup Modell eines anderen Teilprojekts, das eine Schnittstelle mit Beispielen anbietet

Model-View-Controller-Pattern Pattern, das bei der Programmierung komplexer Software durch Abstraktion und Vereinheitlichung zu mehr Übersichtlichkeit führt

MVC Abkürzung für Model-View-Controller-Pattern

Objekt Element in einem Raum, zum Beispiel ein Tisch, ein Personenplatz oder eine Ablage

Personenplatz Platz, an dem sich eine Person in einem Raum befinden kann

Rasterposition Position im Grid

Raumtyp Oberbegriff für Halle, Büro, Besprechungsraum und eventuell weitere Räume

semantische Oberfläche Interaktive Benutzeroberfläche, die Interaktionsmöglichkeiten anbietet

Schnittstelle Verbindung zu einem anderen Teil des Gesamtprojekts

SCRUM-Meeting Wöchentliches Treffen aller Projektmitarbeiter zum Austausch über Fortschritte und Probleme

Shared Documents Dokumente, die gleichzeitig von mehreren Personen an verschiedenen Geräten gesehen und bearbeitet werden können

Software Ausführbares Computerprogramm

Softwareprojekt Pflichtveranstaltung der Leibniz Universität Hannover, in der dieses Projekt entsteht

System Gesamtheit der laufenden Software einschließlich aller momentanen Änderungen durch Userinputs

urheberrechtlich geschützt vom Verfasser nicht für die Öffentlichkeit freigegeben

Userinput Eingaben durch den Benutzer

valide Den Kriterien entsprechend

VirtuHoS Name des gesamten Projekts, von dem der Editor ein Teilprojekt ist

9 Abnahme-Testfälle

Die Skizzen im Folgenden sind nur ungefähre Darstellungen. Sie dienen nur dazu den Testfall zu verdeutlichen und könnten sich vom fertigen Programm unterscheiden.
Die Zahlen in '( )' zeigen die Stellen in der Skizze an, an der die Eingabe mit der gleichen Nummer stattfindet.

9.1 Test 1: Öffnen des Editormodus

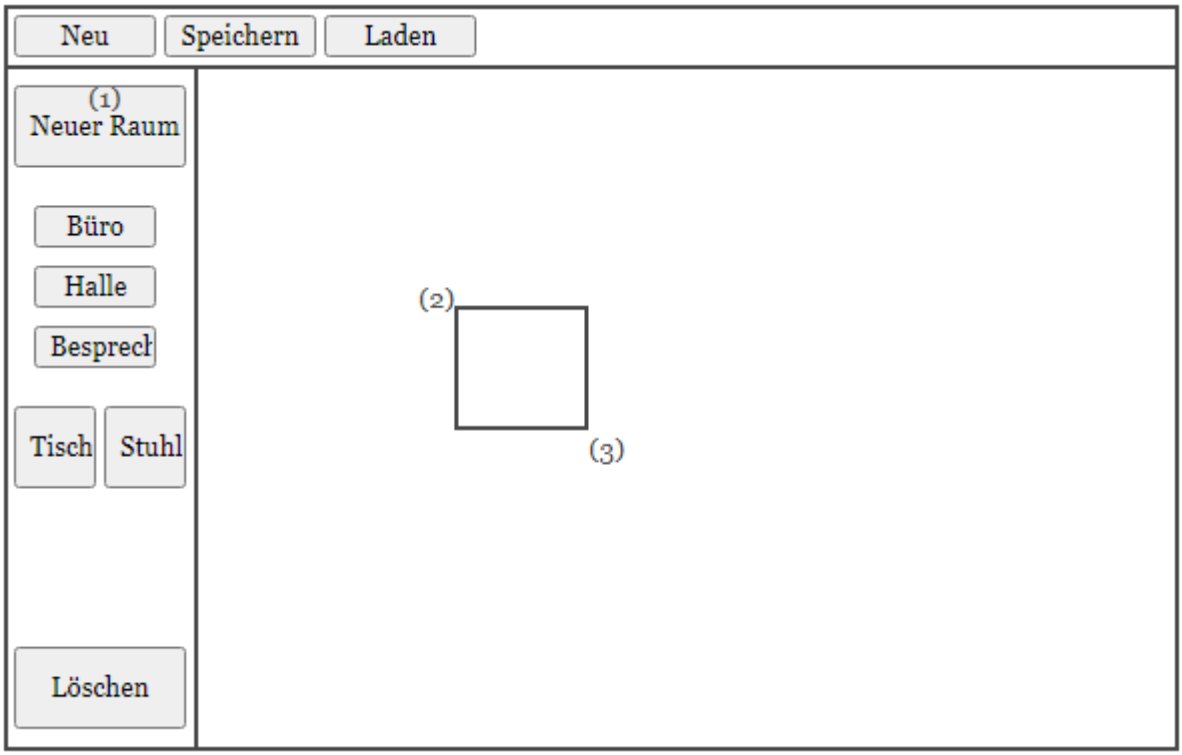
Setup: Der Nutzer befindet sich eingeloggt im Hauptmenü der Software.

Eingabe/Ablauf	**Soll-Ergebnis**
Die "Gebäude erstellen" Funktion wird ausgewählt.	Das Editor-Interface öffnet sich mit einem leeren Gebäude.

9.2 Test 2: Raum zeichnen

Setup: Das Editor-Interface ist geöffnet. Die Zeichenfläche ist leer.

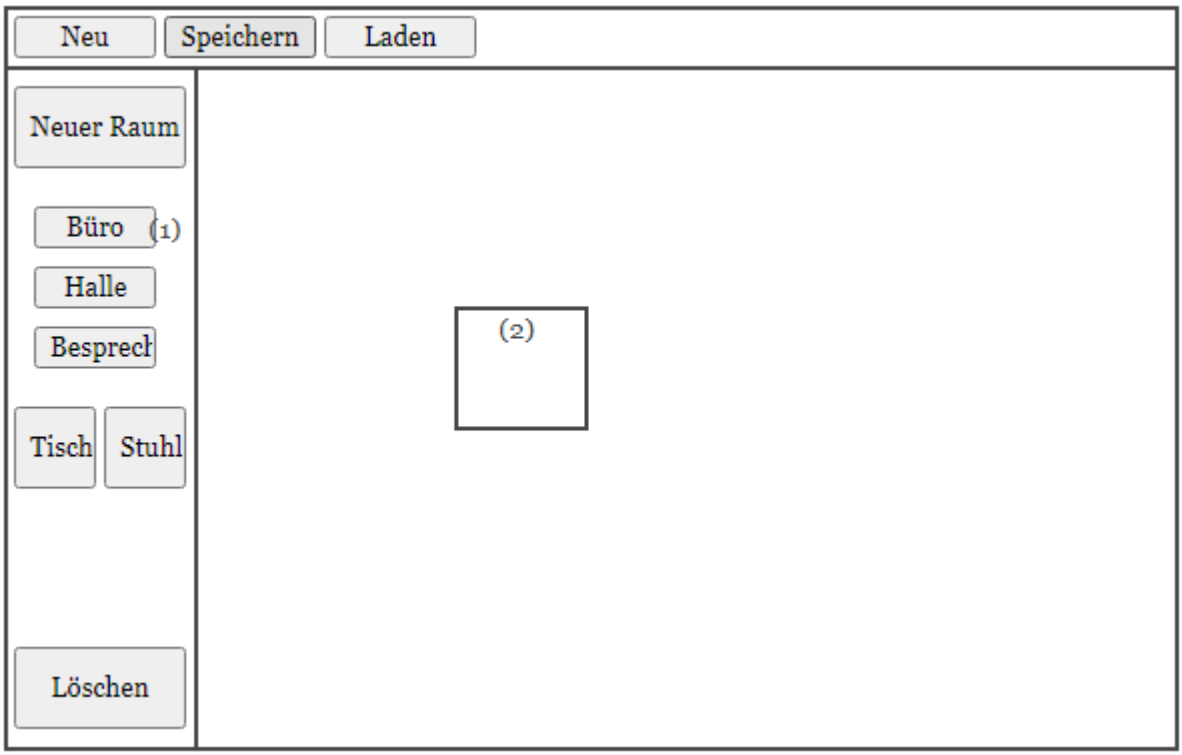
Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt die "Raum zeichnen" Funktion aus. (1)	Das System zeigt die "Raum zeichnen" Funktion graphisch als aktiv an.
Der Nutzer hält die linke Maustaste gedrückt und zieht einen 4x4 Felder großen Raum an eine leere Stelle auf der Zeichenfläche. (2)	Das System zeigt dynamisch eine Vorschau des zu entstehenden Raumes an.
Der Nutzer lässt die linke Maustaste los. (3)	Das System zeigt den 4x4 Felder großen Raum nun im Gebäude an. Dieser hat noch keinen Typen und lässt sich weiter bearbeiten.



9.3 Test 3: Raumtyp zuordnen

Setup: Das Editor-Interface ist geöffnet, es ist nichts ausgewählt und ein Gebäude mit einem 6x6 Felder großen Raum ohne Typzuweisung befindet sich auf der Zeichenfläche.

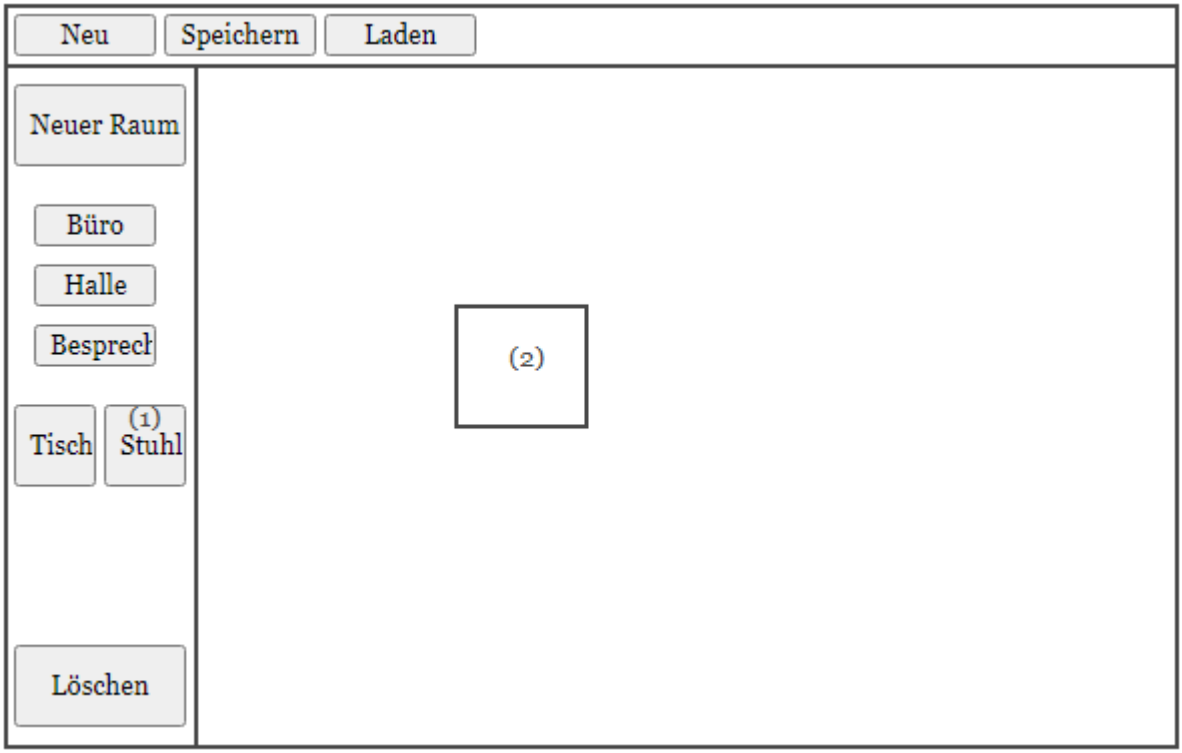
Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt die ”Büro zuweisen” Funktion aus. (1)	Das System zeigt die ”Büro zuweisen” Funktion graphisch als aktiv an.
Der Nutzer wählt den bereits gezeichnete Raum aus. (2)	Das System zeigt nun der Raum als Büro an.



9.4 Test 4: Personenplatz hinzufügen

Setup: Der Editormodus ist geöffnet und ein Gebäude ist geladen. In diesem Beispielgebäude befindet sich ein einzelner leerer 6x6 Felder großer Raum mit dem Typ Büro.

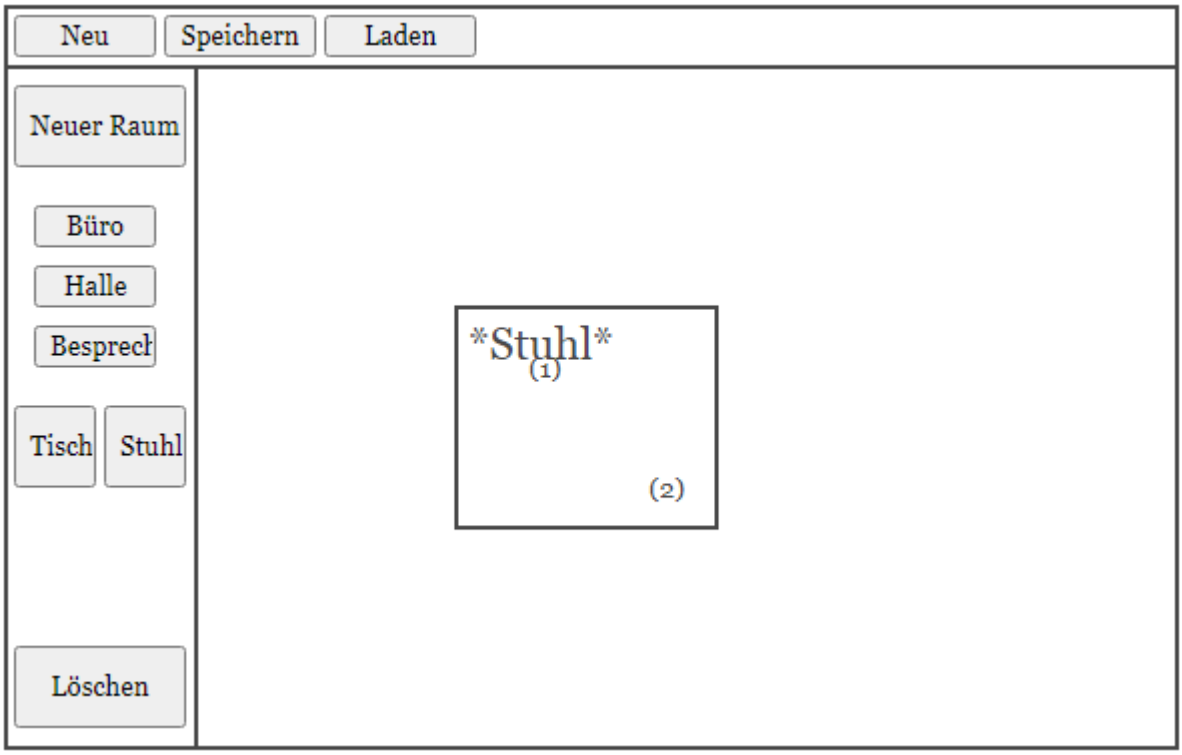
Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt die "Platz hinzufügen" Funktion aus. (1)	Das System zeigt die "Platz hinzufügen" Funktion graphisch als aktiv an.
Der Nutzer klickt auf ein leeres Feld innerhalb des Raumes. (2)	Das System fügt einen Platz an der vom Benutzer gewählten Stelle hinzu.



9.5 Test 5: Objekt verschieben

Setup: Der Editormodus ist geöffnet und ein Gebäude ist geladen. In diesem Beispielgebäude befindet sich ein einzelner 6x6 großer Raum mit einem Personenplatz. Der Nutzer hat keine Funktion ausgewählt.

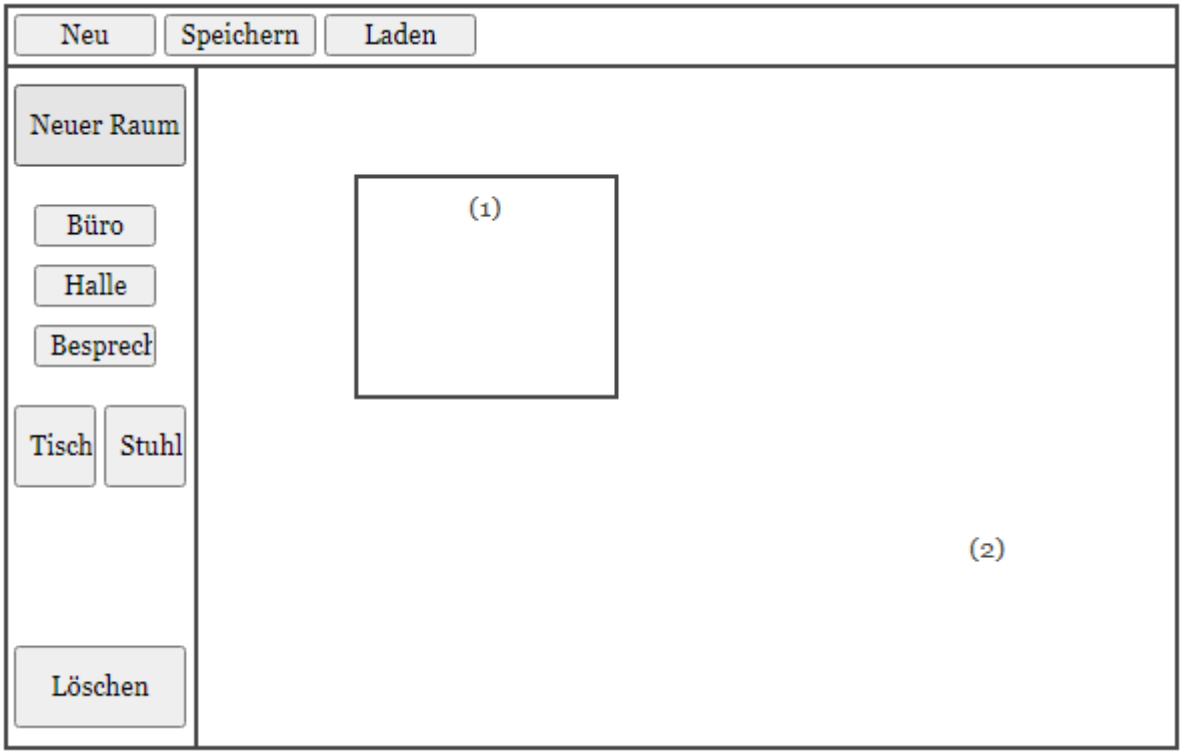
Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt den Personenplatz mit der linken Maustaste aus. (1)	Das System makiert den Perosnenplatz.
Der Nutzer bewegt den Personenplatz per Drag and Drop an eine freie Stelle innerhalb des Raumes. (2)	Das System zeigt den Personenplatz an der ausgewählten Stelle an.



9.6 Test 6: Raum verschieben

Setup: Das Programm befindet sich im Editor-Modus. Ein Gebäude, bestehend aus einem Raum mit einem Sitzplatz darin, ist auf der Zeichenfläche.

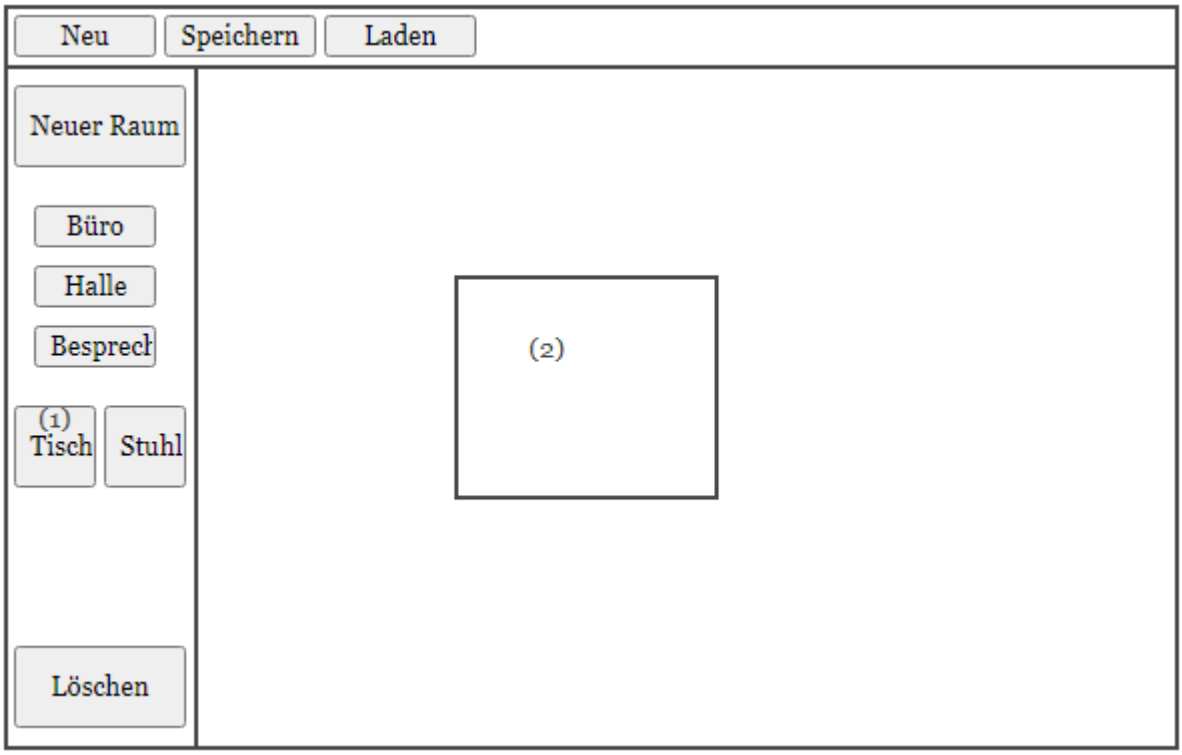
Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer klickt mit der linken Maustaste auf den Raum und hält jene gedrückt. (1)	Das System makiert den Raum graphisch.
Der Nutzer zieht die Maus auf einen freien Bereich der Zeichenfläche und lässt die linke Maustaste los (2)	Das System zeigt den Raum an einer zur Maus nahen Raster-Position an.



9.7 Test 7: Tisch hinzufügen

Setup: Der Editormodus ist geöffnet und ein Gebäude ist geladen. In diesem Beispielgebäude befindet sich ein einzelner leerer 6x6 Felder großer Raum mit dem Typ Büro.

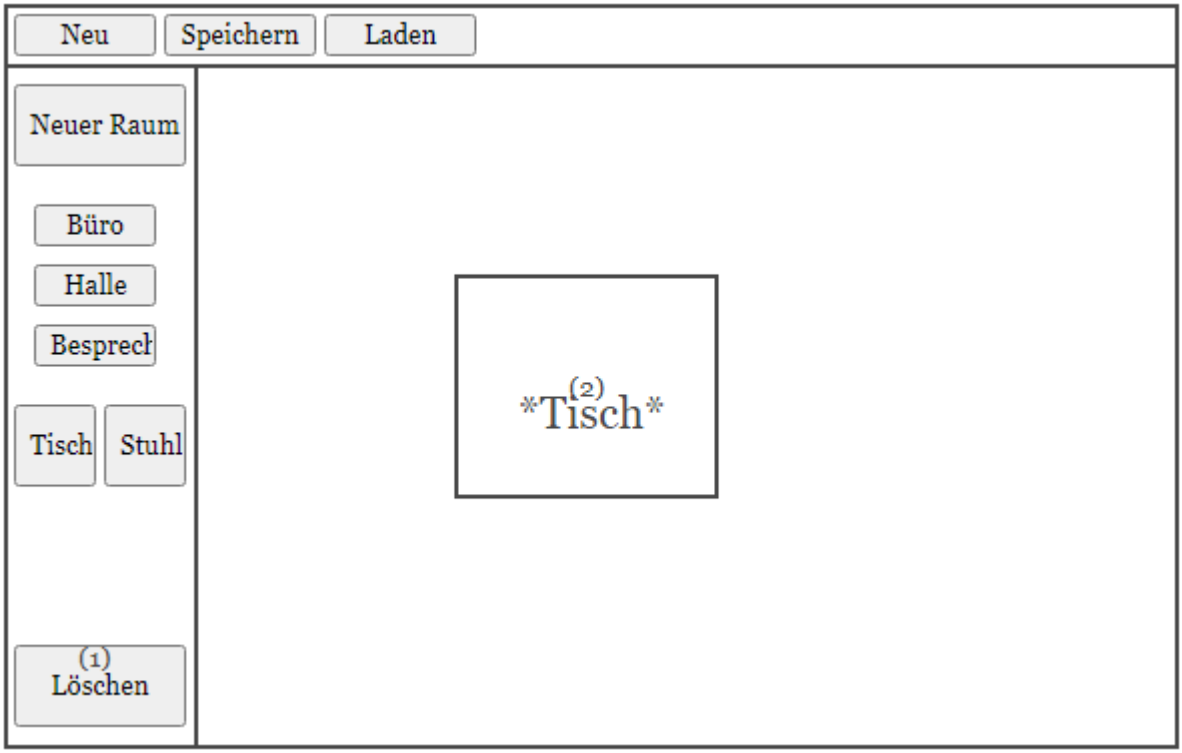
Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt die "Tisch hinzufügen" Funktion aus. (1)	Das System zeigt die "Tisch hinzufügen" Funktion graphisch als aktiv an.
Der Nutzer hält die linke Maustaste gedrückt und zieht einen 2x1 Felder großen Tisch an eine leere Stelle in einem Raum und lässt die Maustaste los. (2)	Das System zeigt den 2x1 Felder großen Tisch an der mit der Maus gewählten Stelle an.



9.8 Test 8: Objekt löschen

Setup: Der Editormodus ist geöffnet und ein Gebäude ist geladen. In dem Gebäude existiert ein Raum inklusive eines Tisches.

Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt die ”Objekt löschen” Funktion. (1)	Das System zeigt die ”Objekt löschen” Funktion graphisch als aktiv an.
Der Nutzer wählt den Tisch aus. (2)	Das System entfernt den Tisch aus dem Raster.



9.9 Test 9: Gebäude speichern

Setup: Der Editormodus ist geöffnet und ein Gebäude ist vorhanden. In diesem Beispielgebäude befindet sich mindestens ein Raum mit einem Personen-platz.

Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt die ”Gebäude speichern” Funktion aus.	Das System zeigt die ”Gebäude speichern” Funktion graphisch als aktiv an.
Der Nutzer gibt den Namen ”Gebäude 1” für das Gebäude an.	Das System zeigt den Namen "Gebäude 1än.
Der Nutzer bestätigt das Speichern.	Das System speichert das Gebäude mit dem Namen ”Gebäude 1” in der Datenbank.

9.10 Test 10: Gebäude laden

Setup: Der Editormodus ist geöffent. Ein weiteres gespeichertes Gebäude ist bereitgestellt.

Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer wählt die ”Gebäude laden” Funktion aus.	Das System zeigt dem Nutzer alle verfügbaren, gespeicherten Gebäude an.
Der Nutzer wählt das bereitgestellte Gebäude zum Laden aus.	Das System lädt das Gebäude.

9.11 Test 11: Nutzer wird auf einem neuen Platz angezeigt

Setup: Das Programm befindet sich im Anzeige-Modus. Es ist ein Gebäude aktiv,das aus zwei Büros (Büro A und Büro B) besteht. Die Person Max befindet sich in Büro A und wird dort angezeigt. Das Büro B hat einen freien Platz.

Eingabe/Ablauf	**Soll-Ergebnis**
Der Nutzer verursacht, dass ein Signal an das Programm geschickt, das dafür sorgen soll, dass Max in Büro B geht.	Das System zeigt Max in Büro B und nicht mehr in Büro A an.