

Stura Mitgliederdatenbank

Inhaltsverzeichnis

Technische Spezifikation	2
1. Vision StuRa Mitgliederdatenbank	2
1.1. Einführung	2
1.2. Positionierung	3
1.3. Stakeholder Beschreibungen	4
2. Use-Case Model StuRa-Mitgliederdatenbank	5
2.1. Use-Case: Mitglieder verwalten	5
2.2. Use-Case: Ämter verwalten	7
2.3. Use-Case: Systemnutzer verwalten	9
2.4. Use-Case: Arbeitsleitfaden abarbeiten	11
2.5. Use-Case: Historie einsehen	12
3. Stura-Mitgliederdatenbank System-Wide Requirements Specification	14
3.1. Einführung	14
3.2. Systemweite funktionale Anforderungen	14
3.3. Qualitätsanforderungen für das Gesamtsystem	14
3.4. Zusätzliche Anforderungen	15
4. Glossar	16
4.1. Einführung	16
4.2. Begriffe	16
4.3. Abkürzungen und Akronyme	16
4.4. Verzeichnis der Datenstrukturen	17
Projektdokumentation	17
1. Projektplan StuRa_Mitgliederdatenbank	17
1.1. Einführung	17
1.2. Projektorganisation	17
1.3. Praktiken und Bewertung	17
1.4. Meilesteine und Ziele	17
1.5. Erkenntnisse (Lessons learned)	18
2. Risikoliste -Projektthema-	18
3. Iterationsplan Iteration 1	18
3.1. Meilensteine	19
3.2. Wesentliche Ziele	19
3.3. Aufgabenzuordnung	19
3.4. Probleme (optional)	19
3.5. Bewertungskriterien	20
3.6. Assessment	20

4. Iterationsplan Iteration 2	20
4.1. Meilensteine	20
4.2. Wesentliche Ziele	20
4.3. Aufgabenzuordnung	21
4.4. Probleme (optional)	21
4.5. Bewertungskriterien	21
4.6. Assessment	21
5. Iterationsplan Iteration 3	22
5.1. Meilensteine	22
5.2. Wesentliche Ziele	22
5.3. Aufgabenzuordnung	22
5.4. Probleme (optional)	23
5.5. Bewertungskriterien	23
5.6. Assessment	23
Entwurfsdokumentation	23
1. Architecture Notebook <Projektthema>	24
1.1. Zweck	24
1.2. Architekturziele und Philosophie	24
1.3. Annahmen und Abhängigkeiten	24
1.4. Architektur-relevante Anforderungen	24
1.5. Entscheidungen, Nebenbedingungen und Begründungen	24
1.6. Architekturmechanismen	25
1.7. Wesentliche Abstraktionen	25
1.8. Schichten oder Architektur-Framework	25
1.9. Architektursichten (Views)	25
2. Wireframes	25
Dokumentation Essence Navigator	31

Technische Spezifikation

1. Vision StuRa Mitgliederdatenbank

Lukas Hirsch <s79199@htw-dresden.de> Theresa Schüttig <s79136@htw-dresden.de>

1.1. Einführung

Der Zweck dieses Dokuments ist es, die wesentlichen Bedarfe und Funktionalitäten der Mitgliederdatenbank für den StuRa zu sammeln, zu analysieren und zu definieren. Der Fokus liegt auf den Fähigkeiten, die von Stakeholdern und adressierten Nutzern benötigt werden, und der Begründung dieser Bedarfe. Die Details, wie die Mitgliederdatenbank diese Bedarfe erfüllt, werden in der Use-Case und Supplementary Specification beschrieben.

1.1.1. Zweck

Der Zweck dieses Dokuments ist es, die wesentlichen Anforderungen an das System aus Sicht und mit den Begriffen der künftigen Anwender zu beschreiben.

1.1.2. Gültigkeitsbereich (Scope)

Dieses Visions-Dokument bezieht sich auf die StuRa-Mitgliederdatenbank, das vom in dieser [Datei](#) aufgelisteten Team entwickelt wird. Das System wird es Mitgliedern der Referate Verwaltung und Präsidium sowie Mitarbeitern des StuRas erlauben, Daten von Mitgliedern des StuRas über eine Webanwendung zu verwalten, um Informationen zu Mitgliedern, Ämtern und Wahlen zu speichern und einzusehen, einen Arbeitsleitfaden für die interne Verwaltung sowie eine Checkliste zu generieren. Im Idealfall sollte die Anwendung auch das [hier](#) veröffentlichte Organigramm erzeugen können.

1.1.3. Definitionen, Akronyme und Abkürzungen

siehe [Glossar](#)

1.1.4. Referenzen

[Aufgabenstellung](#)

1.2. Positionierung

1.2.1. Fachliche Motivation

Derzeit werden Mitgliederdaten des StuRas über eine Excel-Tabelle verwaltet. Die StuRa-Mitgliederdatenbank soll die Verwaltung von Mitgliedern im StuRa erleichtern und übersichtlicher gestalten, indem Mitglieder über eine benutzerfreundliche Website eingesehen, bearbeitet und hinzugefügt werden können. Zudem soll über eine Historie ersichtlich sein, wann in der Datenbank von wem welche Änderungen vorgenommen wurden. Für die interne Verwaltung soll das Programm eine Checkliste mit Aufgaben (z.B. zum Mailverteiler hinzufügen, Zugang zum Schlüsselgang einrichten, etc.) erstellen können. Des Weiteren sollen die Software in der Lage sein, Anwesenheitslisten sowie das Organigramm zu generieren.

1.2.2. Problem Statement

Das Problem	Die Excel-Tabelle ist unübersichtlich und die Bearbeitung erfordert zu viel Aufwand
betrifft	die interne Verwaltung
die Auswirkung davon ist	zu viel unnötiger Arbeitsaufwand
eine erfolgreiche Lösung wäre	Über eine Webseite könnte das Bearbeiten nur über das Aufüllen von ein paar Textfeldern ermöglicht werden

Das Problem	Es ist nicht möglich, herauszufinden, von wem bestimmte Änderungen vorgenommen wurden
betrifft	die interne Verwaltung
die Auswirkung davon ist	Verantwortliche für inkorrekte Änderungen können nicht ausfindig gemacht werden und Wiederherstellen korrekter Daten kann sich schwierig gestalten
eine erfolgreiche Lösung wäre	das Erstellen einer Historie

Das Problem	Das Erteilen von Zugängerechten ist unorganisiert
betrifft	die interne Verwaltung, StuRa-Mitglieder
die Auswirkung davon ist	die Verwaltung hat einen schlechten Überblick über noch zu erteilende Zugänge und StuRa-Mitglieder müssen auf Zugänge länger warten als nötig
eine erfolgreiche Lösung wäre	das Erstellen eines Arbeitsleitfadens für die interne Verwaltung

1.2.3. Positionierung des Produkts

Für	Referat Verwaltung (StuRa)
welchem	das Verwalten der Mitglieder deutlich erleichtert wird
Die Lösung ist eine	Webanwendung zur Mitgliederverwaltung
Die / Das	das Verwalten übersichtlicher gestaltet und eine Historie speichert
Im Gegensatz zu	Excel-Tabelle
Unser Produkt	zeigt nur abgefragte Daten an und ermöglicht das Hinzufügen und Bearbeiten in kürzerer Zeit

1.3. Stakeholder Beschreibungen

1.3.1. Zusammenfassung der Stakeholder

Name	Beschreibung	Verantwortlichkeiten
Verwaltung des StuRaS	selbsterklärend	Eintragen von Mitgliedern, Dokumentieren von Wahlen, Einrichten von Zugängen
Präsidium	Das Präsidium ist zuständig für Sitzungen des StuRaS	Dokumentieren von Anwesenheit

Name	Beschreibung	Verantwortlichkeiten
Mitglieder des StuRas	???	???

1.3.2. Benutzerumgebung

Beschreiben Sie die Arbeitsumgebung des Nutzers. Hier sind einige Anregungen:

1. Anzahl der Personen, die an der Erfüllung der Aufgabe beteiligt sind. Ändert sich das?
2. Wie lange dauert die Bearbeitung der Aufgabe? Wie viel Zeit wird für jeden Arbeitsschritt benötigt? Ändert sich das?
3. Gibt es besondere Umgebungsbedingungen, z.B. mobil, offline, Außeneinsatz, Touchbedienung, Nutzung durch seh- oder hörbeeinträchtigte Personen?
 - Derzeit eingesetzte Anwendung: Excel

Hier können zudem bei Bedarf Teile des Unternehmensmodells (Prozesse, Organigramme, IT-Landschaft, ...) eingefügt werden, um die beteiligten Aufgaben und Rollen zu skizzieren.

- [Organigramm](#)

2. Use-Case Model StuRa-Mitgliederdatenbank

Lukas Hirsch <s79199@htw-dresden.de>

2.1. Use-Case: Mitglieder verwalten

2.1.1. Kurzbeschreibung

Mitglieder der internen Verwaltung des StuRas sollen Mitglieder hinzufügen, bearbeiten und entfernen können.

2.1.2. Kurzbeschreibung der Akteure

Interne Verwaltung des StuRas

2.1.3. Vorbedingungen

1. Das StuRa-Mitglied ist eingeloggt und verfügt über die benötigten Berechtigungen.

2.1.4. Standardablauf (Basic Flow)

Mitglied hinzufügen

1. Der Use Case beginnt, wenn der Nutzer dem System mitteilt, dass er ein neues Mitglied anlegen möchte.

2. Alle benötigten Daten (Name, Vorname, Wahl, Legislatur) werden angegeben
3. Die Daten werden bestätigt.
4. Der Use Case ist abgeschlossen.

Mitglied bearbeiten

1. Der Use Case beginnt, wenn der Nutzer dem System mitteilt, dass er ein ausgewähltes Mitglied bearbeiten möchte.
2. Der Nutzer kann bestehende Daten einsehen und verändern (Namen oder Legislatur bearbeiten, Legislatur und Wahl hinzufügen)
3. Die veränderten Daten werden bestätigt
4. Der Use Case ist abgeschlossen.

Mitglied löschen

Der Use Case beginnt, wenn der Nutzer dem System mitteilt, dass er ein ausgewähltes Mitglied löschen möchte. . Der Löschvorgang wird bestätigt. . Der Use Case ist abgeschlossen.

2.1.5. Alternative Abläufe

<Alternativer Ablauf 1>

Wenn <Akteur> im Schritt <x> des Standardablauf <etwas macht>, dann . <Ablauf beschreiben> . Der Use Case wird im Schritt <y> fortgesetzt.

2.1.6. Unterabläufe (subflows)

<Unterablauf 1>

1. <Unterablauf 1, Schritt 1>
2. ...
3. <Unterablauf 1, Schritt n>

2.1.7. Wesentliche Szenarios

<Szenario 1>

1. <Szenario 1, Schritt 1>
2. ...
3. <Szenario 1, Schritt n>

2.1.8. Nachbedingungen

Hinzugefügtes Mitglied

Das neue Mitglied befindet sich nun in der Mitgliedertabelle der Datenbank. Der Nutzer, vom das

Erstellen ausging, sowie das Erstelldatum werden in der Historie verzeichnet.

Bearbeitetes Mitglied

Die Attribute in der Mitgliedertabelle wurden aktualisiert und der Ursprungszustand, der Nutzer, von dem die Bearbeitung ausging, sowie das Bearbeitungsdatum der Historie hinzugefügt.

Gelöschtes Mitglied

Das Mitglied wurde aus der Mitgliedertabelle entfernt. Alle personenbezogenen Daten des Mitglieds können nicht mehr wiederhergestellt werden.

2.1.9. Besondere Anforderungen

1. Beim Löschen eines Mitglieds hat eine zusätzliche Sicherheitsabfrage zu erfolgen.

2.2. Use-Case: Ämter verwalten

2.2.1. Kurzbeschreibung

Mitglieder der internen Verwaltung des StuRas können Ämter hinzufügen, bearbeiten und löschen.

2.2.2. Kurzbeschreibung der Akteure

Interne Verwaltung

2.2.3. Vorbedingungen

Mitglied der internen Verwaltung ist eingeloggt

2.2.4. Standardablauf (Basic Flow)

Amt hinzufügen

1. Der Use Case beginnt, wenn der Nutzer dem System mitteilt, dass er ein Amt hinzufügen möchte
2. Eingabe der Daten
3. Bestätigung
4. Der Use Case ist abgeschlossen.

Amt bearbeiten

1. Der Use Case beginnt, wenn der Nutzer dem System mitteilt, dass er ein Amt bearbeiten möchte
2. while Nutzer hat Daten noch nicht gespeichert
 - Hinzufügen eines Mitglieds (mit Angabe des Legislaturbeginns) oder Entfernen eines Mitglieds
3. end while

4. Der Use Case ist abgeschlossen.

Amt löschen

1. Der Use Case beginnt, wenn der Nutzer dem System mitteilt, dass er ein Amt löschen möchte
2. Bestätigung der Löschung
3. Der Use Case ist abgeschlossen.

2.2.5. Alternative Abläufe

<Alternativer Ablauf 1>

Wenn <Akteur> im Schritt <x> des Standardablauf <etwas macht>, dann . <Ablauf beschreiben> .
Der Use Case wird im Schritt <y> fortgesetzt.

2.2.6. Unterabläufe (subflows)

<Unterablauf 1>

1. <Unterablauf 1, Schritt 1>
2. ...
3. <Unterablauf 1, Schritt n>

2.2.7. Wesentliche Szenarios

<Szenario 1>

1. <Szenario 1, Schritt 1>
2. ...
3. <Szenario 1, Schritt n>

2.2.8. Nachbedingungen

Veränderung der Daten in der Datenbank

Die Daten werden in der Datenbank aktualisiert. Werden Mitglieder zu Ämtern hinzugefügt oder entfernt, so wird dies auch in der Mitgliedertabelle vermerkt. Gelöschte Ämter befinden sich weiterhin in der Datenbank, werden in der Webanwendung aber nicht mehr angezeigt und können nicht besetzt sein oder werden.

2.2.9. Besondere Anforderungen

1. Beim Löschen eines Amtes hat eine zusätzliche Sicherheitsabfrage zu erfolgen.

2.3. Use-Case: Systemnutzer verwalten

2.3.1. Kurzbeschreibung

Als Administrator des Systems und Mitglied der internen Verwaltung des StuRas möchte ich Systemnutzer hinzufügen, bearbeiten und löschen können.

2.3.2. Kurzbeschreibung der Akteure

Administrator des Systems

Der/die Administrator/in/en haben Vollzugriff auf das System und sind für die Verwaltung der Nutzerzugänge verantwortlich.

Interne Verwaltung des StuRas

Eines oder mehrere Mitglieder der internen Verwaltung sind für die Administration des Systems verantwortlich.

2.3.3. Vorbedingungen

- Der/die Administrator/in ist im System eingeloggt.

2.3.4. Standardablauf (Basic Flow)

1. Der Use Case beginnt, wenn der/die Administrator/in die Oberfläche zur Verwaltung der Systemnutzer öffnet.
2. while Administrator/in hat noch nicht alle Änderungen vorgenommen
 - a. Dem Akteur wird eine aktuelle Liste aller Systemnutzer angezeigt.
 - b. Es wird ausgewählt, ob ein Nutzer hinzugefügt, bearbeitet oder gelöscht werden soll.
 - c. Fortsetzung in den jeweiligen Unterabläufen (subflows).
3. end while
4. Der Use Case ist abgeschlossen.

2.3.5. Alternative Abläufe

<Alternativer Ablauf 1>

Wenn <Akteur> im Schritt <x> des Standardablauf <etwas macht>, dann . <Ablauf beschreiben> . Der Use Case wird im Schritt <y> fortgesetzt.

2.3.6. Unterabläufe (subflows)

Hinzufügen eines neuen Nutzers

1. Der Admin wählt aus, dass er einen neuen Nutzer hinzufügen möchte.

2. Der Admin vergibt einen Benutzernamen, ein Passwort (welches beim erstmaligen Login geändert werden muss) und ordnet den Systemnutzer dem entsprechenden StuRa-Mitglied zu.
3. Der Admin bestätigt die Eingabe und es wird ihm/ihr die aktualisierte Systemnutzer-Liste angezeigt.

Ändern eines bestehenden Nutzers

1. Der Admin wählt einen bestehenden Eintrag in der Systemnutzer-Liste aus.
2. Der Admin wählt aus, dass er diesen Systemnutzer ändern möchte.
3. Der Admin ändert den Nutzernamen, setzt das Passwort zurück und/oder ändert die Zuordnung zum entsprechenden StuRa-Mitglied.
4. Der Admin bestätigt die Eingabe und es wird ihm/ihr die aktualisierte Systemnutzer-Liste angezeigt.

Löschen eines bestehenden Nutzers

1. Der Admin wählt einen bestehenden Eintrag in der Systemnutzer-Liste aus.
2. Der Admin wählt aus, dass er diesen Systemnutzer löschen möchte.
3. Der Admin bestätigt die Sicherheitsabfrage und es wird ihm/ihr die aktualisierte Systemnutzer-Liste angezeigt.

2.3.7. Wesentliche Szenarios

<Szenario 1>

1. <Szenario 1, Schritt 1>
2. ...
3. <Szenario 1, Schritt n>

2.3.8. Nachbedingungen

Hinzufügen eines neuen Nutzers

1. Falls ein neuer Nutzer hinzugefügt wurde, so wird dieses als neuer Datensatz in der Nutzerdatenbank gespeichert.
2. Der Nutzer hat mit den angegebenen Anmeldedaten Zugriff auf das System.

Änderung eines bestehenden Nutzers

1. Falls ein Nutzer geändert wurde, sind die geänderten Daten in der Nutzerdatenbank gespeichert.
2. Der Nutzer hat mit den geänderten Anmeldedaten Zugriff auf das System, aber nicht mehr mit den alten Daten.

Löschen eines bestehenden Nutzers

1. Falls ein Nutzer gelöscht wurde, so wurde sein Datensatz aus der Nutzerdatenbank entfernt.
2. Der gelöschte Nutzer hat keinen Zugriff mehr auf das System.

2.3.9. Besondere Anforderungen

1. Gespeicherte Anmeldedaten, insbesondere Passwörter, sollten niemals im Klartext in der Nutzerdatenbank vorliegen und es sollte niemals mit Klartextdaten gearbeitet werden.
2. Vor dem Löschen eines Systemnutzers hat eine Sicherheitsabfrage zu erfolgen.

2.4. Use-Case: Arbeitsleitfaden abarbeiten

2.4.1. Kurzbeschreibung

Nach Aufnahme eines neuen Mitglieds im StuRa müssen bestimmte Aufgaben (Emailverteiler, Schlüsselkasten, Zugangsschließsystem, Berechtigung Website/Dateiverwaltung/Aufgabenverwaltung, notwendige Unterschriften) abgearbeitet werden.

2.4.2. Kurzbeschreibung der Akteure

Interne Verwaltung

Mitglieder der internen Verwaltung sind für die Abarbeitung der Checkliste verantwortlich

2.4.3. Vorbedingungen

Das Mitglied der internen Verwaltung ist eingeloggt und ihm liegt eine Liste mit Mitgliedern vor. Es kann nach Mitgliedern filtern, deren Checkliste noch nicht abgearbeitet wurde.

2.4.4. Standardablauf (Basic Flow)

1. Der Use Case beginnt, wenn der Nutzer auswählt, dass er die Checkliste für ein Mitglied bearbeiten möchte.
2. Der Nutzer kann Aufgaben kennzeichnen, die er abgearbeitet hat und Kennzeichnungen wieder entfernen.
3. Der Nutzer bestätigt die aktualisierte Checkliste.
4. Der Use Case ist abgeschlossen.

2.4.5. Alternative Abläufe

<Alternativer Ablauf 1>

Wenn <Akteur> im Schritt <x> des Standardablauf <etwas macht>, dann . <Ablauf beschreiben> . Der Use Case wird im Schritt <y> fortgesetzt.

2.4.6. Unterabläufe (subflows)

<Unterablauf 1>

1. <Unterablauf 1, Schritt 1>
2. ...
3. <Unterablauf 1, Schritt n>

2.4.7. Wesentliche Szenarios

StuRa-Mitglied M1 wurde dem E-Mail-Verteiler hinzugefügt

1. Das Mitglied der internen Verwaltung wählt aus, dass es die Checkliste von M1 bearbeiten will
2. Es bestätigt Verteilung zum E-Mail-Verteiler
3. Die Veränderung wird bestätigt

Falsche Angaben bei Schließsystem von StuRa-Mitglied M2 mit abgearbeiteter Checkliste

1. Das Mitglied der internen Verwaltung wählt aus, dass es die Checkliste von M2 bearbeiten will
2. Es kennzeichnet die Berechtigung für das Zugangsschließsystem als nicht abgearbeitet
3. Die Veränderung wird bestätigt

2.4.8. Nachbedingungen

Gespeicherte Checkliste

1. die gespeicherte Checkliste wurde in der Datenbank aufgenommen

2.4.9. Besondere Anforderungen

<Besondere Anforderung 1>

2.5. Use-Case: Historie einsehen

2.5.1. Kurzbeschreibung

Um Änderungen an der Mitgliederdatenbank nachvollziehen und ggf. rückgängig machen zu können,
möchte ich die Historie aller Änderungen einsehen können.

2.5.2. Kurzbeschreibung der Akteure

Interne Verwaltung

1. Die Mitglieder der internen Verwaltung sind für die Richtigkeit und Vollständigkeit der Daten in der Mitgliederdatenbank verantwortlich.

2.5.3. Vorbedingungen

1. Der Akteur ist eingeloggt und verfügt über die Berechtigung, die Historie einsehen zu können.

2.5.4. Standardablauf (Basic Flow)

1. Der Use Case beginnt, wenn der Akteur dem System mitteilt, dass er die Historie einsehen möchte.
2. Das System listet die letzten n Einträge der Historie auf.
3. while Akteur hat noch nicht alle Informationen, die er braucht
 - Der Nutzer ändert die Filterkriterien für die Einträge (siehe Unterablauf).
 - ODER
 - Der Nutzer lässt sich ältere Einträge anzeigen, indem er die Seite wechselt.
4. end while
5. Der Use Case ist abgeschlossen.

2.5.5. Alternative Abläufe

<Alternativer Ablauf 1>

Wenn <Akteur> im Schritt <x> des Standardablauf <etwas macht>, dann . <Ablauf beschreiben> . Der Use Case wird im Schritt <y> fortgesetzt.

2.5.6. Unterabläufe (subflows)

Filterkriterien ändern

1. Der Akteur entscheidet sich dazu, die Historie nach bestimmten Kriterien zu filtern.
2. Der Akteur gibt die entsprechenden Filterkriterien ein und teilt dem System mit, dass es alle Einträge basierend auf diesen Kriterien auflisten soll.
3. Das System listet die letzten n Einträge der Historie auf, die den Filterkriterien entsprechen.

2.5.7. Wesentliche Szenarios

<Szenario 1>

1. <Szenario 1, Schritt 1>
2. ...
3. <Szenario 1, Schritt n>

2.5.8. Nachbedingungen

<Nachbedingung 1>

2.5.9. Besondere Anforderungen

Unterteilung in Seiten (Pagination)

- Damit nicht immer die gesamte Historie geladen werden muss, sollten zunächst nur die letzten n Einträge angezeigt werden.
- Mittels einer Pagination (Menü zum Wechseln von Seiten) kann der Nutzer auch ältere Einträge sehen, die nur bei Bedarf geladen werden.

3. Stura-Mitgliederdatenbank System-Wide Requirements Specification

3.1. Einführung

In diesem Dokument werden die systemweiten Anforderungen für das Projekt Stura-Mitgliederdatenbank spezifiziert. Die Gliederung erfolgt nach der FURPS+ Anforderungsklassifikation:

- Systemweite funktionale Anforderungen (F),
- Qualitätsanforderungen für Benutzbarkeit, Zuverlässigkeit, Effizienz und Wartbarkeit (URPS) sowie
- zusätzliche Anforderungen (+) für technische, rechtliche, organisatorische Randbedingungen

NOTE

Die funktionalen Anforderungen, die sich aus der Interaktion von Nutzern mit dem System ergeben, sind als Use Cases in einem separaten Dokument festgehalten.
[\[Use-Cases\]](#)

3.2. Systemweite funktionale Anforderungen

Authentifizierung:

- Daten müssen vor Unbefugten geschützt werden

Zeitsteuerung:

- Die Datenbank muss in regelmäßigen Abständen (Jeden Monat?) gesichert werden
- Alte Sicherungen Löschen (Welche?)

Auditierung:

- Es soll ein Änderungslog geführt werden (Datum-Nutzer-Änderung)

3.3. Qualitätsanforderungen für das Gesamtsystem

3.3.1. Benutzbarkeit (Usability)

Sprache: deutsch/englisch

Datumsformat: dd.mm.yyyy

Währungen: Euro

Sehr einfaches Erlernen und schnelle Bedienung.

3.3.2. Zuverlässigkeit (Reliability)

Verfügbarkeit: Verfügbar soll die Software solange sein, wie der Server im StuRa läuft. Das Programm soll Wartungsfrei laufen, bzw vom Endnutzer gewartet werden können im laufendem Betrieb.

Wiederherstellbarkeit: Das Programm soll bei totalausfall zum letzten Backup hergestellt werden können (in ca 24h?)

3.3.3. Effizienz (Performance)

Antwortzeiten: infinity (bei bloßer Betrachtung)

Durchsatz: 0

Kapazität: Mehrbenutzerbetrieb muss gewährleistet sein. → Serialisierung für die Datenbankarbeit.

3.3.4. Wartbarkeit (Supportability)

Anpassbarkeit: Es soll bei freier KAPazität eine API für den export eines Organigrammes erstellt werden.

Kompatibilität: Mit allen Betriebssystemen → Browseranwendung. Es muss einbindbar in den Plone sein.

Konfigurierbarkeit: Die zugrundeliegende Datenbank soll frei gewählt werden (MySQL, MSSQL, SQLite, ...)

Wartbarkeit: Wie schon genannt sollen einfache Administrative Aufgaben vom Benutzer übernommen werden.

3.4. Zusätzliche Anforderungen

3.4.1. Einschränkungen (Constraints)

Implementierung: Python (Django), SQLite

Plattform: Plattform unabhängig → Browserapplikation

Ressourcenbegrenzungen: Möglichst geringer Energieverbrauch, Speicherplatz (so viel wie im Container vorhanden), Möglichst wenig Traffic

3.4.2. Organisatorische Randbedingungen

- Anforderungen an Betrieb, Management und Wartung der Anwendung
- zu beachtende Standards, Normen und Regeln *

3.4.3. Rechtliche Anforderungen

- Lizenzierung der Anwendung
- Datenschutz

4. Glossar

Lukas Hirsch <s79199@htw-dresden.de> 0.1.1, 22.12.2019

4.1. Einführung

In diesem Dokument werden die wesentlichen Begriffe aus dem Anwendungsgebiet (Fachdomäne) der StuRa-Mitgliederdatenbank definiert. Zur besseren Übersichtlichkeit sind Begriffe, Abkürzungen und Datendefinitionen gesondert aufgeführt.

4.2. Begriffe

Begriff	Definition und Erläuterung	Synonyme
Systemnutzer	Eine Person, die auf das Mitgliederdatenbank-System Zugriff hat.	Nutzer, Benutzer
Administrator	Eine Person mit Vollzugriff auf das System, welche sich um die Verwaltung desselbigen kümmert.	Admin
Filterkriterium	Ein Begriff, nach welchem die vorhandenen Datensätze "ausgesiebt" werden	
Sicherheitsabfrage	Vom Nutzer wird für bestimmte Operationen eine zusätzliche Bestätigung dieser angefordert.	

4.3. Abkürzungen und Akronyme

Abkürzung	Bedeutung	Erläuterung
AE	Aufwandsentschädigung	Bei besonders hohem Zeitaufwand als Vergütung
UP	Unified Process	Vorgehensmodell für die Softwareentwicklung
Plone	—	Website des STURA
xx	xx	xx

4.4. Verzeichnis der Datenstrukturen

Bezeichnung	Definition	Format	Gültigkeitsregeln	Aliase
Anmeldedaten	Zusammensetzung von Benutzernamen und Passwort.	String	Emailadresse muss @-Zeichen und Punkt enthalten.	Login
XX	XX	XX	XX	XX

Projektdokumentation

1. Projektplan StuRa_Mitgliederdatenbank

Lukas Hirsch <s79199@htw-dresden.de>;

1.1. Einführung

1.2. Projektorganisation

1.2.1. Team Mitglieder

Name	Vorname	Primärrolle	Sekundärrolle
Berger	Mauritius	Project Manager	Developer Backup
Urbons	Lucie Jill	Analyst	---
Uhlig	Helene	Analyst	---
Cremer	Jonathan Vincent	Architect	Developer Backup
Schüttig	Theresa	Developer	Tester Backup
Hempel	Benjamin	Developer	---
Hirsch	Lukas	Tester	Technical Writer
Holland	Stefan	Deployment Eng.	Analyst Backup

1.3. Praktiken und Bewertung

1.4. Meilesteine und Ziele

Iteration	Primary objectives (risks and use case scenarios)	Scheduled start or milestone	Target velocity
Iteration 1	<ul style="list-style-type: none"> • Aufgabe verstanden • Einschätzen ob es sinnvoll und machbar ist • LCO 	01.12.2019 - 20.12.2020	—
Iteration 2	<ul style="list-style-type: none"> • Lösungsweg ausarbeiten (anfangen) 	06.01.2020 - 17.01.2020	—
Iteration 3	<ul style="list-style-type: none"> • Lösungsweg ausarbeiten (beenden) • LCA 	20.01.2020 - 31.01.2020	—
Wintersemester Ende		—	—

1.5. Erkenntnisse (Lessons learned)

2. Risikoliste -Projektthema-

Vorname Nachname <email@domain.org>; Vorname2 Nachname2 <email2@domain.org>;
Vorname3 Nachname3 <email3@domain.org> 0.1, 16.12.2019 :sectnums:

In diesem Dokument sind die wesentlichen Risiken des Projekts aufgeführt. Dabei werden folgende Attribute verwendet:

1. **Typ:** Ressourcen, Geschäftlich, Technisch, Zeitlich
2. **Auswirkung (IMP):** Wert zwischen 1 (niedrig) und 5 (hoch), der die Auswirkungen auf das Projekt angibt, wenn das Risiko eintritt
3. **Wahrscheinlichkeit (PRB):** Prozentangabe für die Eintrittswahrscheinlichkeit des Risikos
4. **Stärke (MAG):** Produkt aus Auswirkung und Wahrscheinlichkeit (damit kann die Liste sortiert werden)

Die Risiken sind in folgender Tabelle dargestellt. Das Datum des Dokuments oben gibt an, wann die Risikoliste zuletzt aktualisiert wurde.

type	priority	chance	priority*chance
------	----------	--------	-----------------

3. Iterationsplan Iteration 1

Mauritius Berger <mauritius.berger@htw-dresden.de> 0.1, 16.12.2019 :toc: :toc-title:
Inhaltsverzeichnis :sectnums: :icons: font

3.1. Meilensteine

Meilenstein	Datum
Beginn der Iteration	02.12.2019
Fertigstellung aller Aufgaben	-
Ende der Iteration	20.12.2019

3.2. Wesentliche Ziele

Beispiele:

- Beheben der Usability-Probleme, die von Abteilung X berichtet wurden
- Ausliefern der Hauptszenarios, die die Integration mit System Y zeigen *
- Present a technical demonstration (demo).

3.3. Aufgabenzuordnung

Die in dieser Iteration geplanten Aufgaben sind in der Work Items List dargestellt (hier Verweis einfügen).

alternativ: Die folgenden Aufgaben werden in dieser Iteration bearbeitet:

Aufgabe bzw. Beschreibung	Priorität	Schätzung der Größe (Punkte)	Status	Referenzen	Zugewiesen (Name)	Gearbeitete Stunden	Schätzung der verbleibenden Stunden
Essence Navigator	niedrig	2	fertig	Essence Navigator Bilder	Jill3, Helene3e	1	1
Use Cases definieren	hoch	3	fertig	Vision	alle	3	3
Use Cases ausarbeiten	mittel	8	fertig	x	alle	8	8

3.4. Probleme (optional)

Problem	Status	Notizen
"Weihnachtspause"	-	Die Zeit um Weihnachten bzw. Neujahr hindert etwas das Projekt, da in der Zeit keine Treffen stattfinden und die Produktivität sinkt.

3.5. Bewertungskriterien

Beispiele:

- 97% der Testfälle auf Systemebene sind erfolgreich.
- Gemeinsame Inspektion des Iterations-Ergebnisses (Inkrement) mit den Abteilungen X und Y ergibt positive Rückmeldung.
- Technische Präsentation / Demo erhält positive Rückmeldungen.

3.6. Assessment

Assessment Ziel	Iteration 1 fertigstellen
Assessment Datum	20.12.2019
Teilnehmer	alle
Projektstatus	gelb

- Beurteilung im Vergleich zu den Zielen Es wurden alle Ziele erreicht.
- Geplante vs. erledigte Aufgaben Es kamen keine Aufgaben hinzu und es wurden alle geplanten Aufgaben bearbeitet.

4. Iterationsplan Iteration 2

Mauritius Berger <mauritius.berger@htw-dresden.de> 0.1, 13.01.2020 :toc: :toc-title:
Inhaltsverzeichnis :sectnums: :icons: font

4.1. Meilensteine

Meilenstein	Datum
Beginn der Iteration	06.01.2020
Fertigstellen der Aufgaben	-
Ende der Iteration	17.01.2020

4.2. Wesentliche Ziele

Beispiele:

- Beheben der Usability-Probleme, die von Abteilung X berichtet wurden
- Ausliefern der Hauptszenarios, die die Integration mit System Y zeigen *
- Present a technical demonstration (demo).

4.3. Aufgabenzuordnung

Die in dieser Iteration geplanten Aufgaben sind in der Work Items List dargestellt (hier Verweis einfügen).

alternativ: Die folgenden Aufgaben werden in dieser Iteration bearbeitet:

Aufgabe bzw. Beschreibung	Priorität	Schätzung der Größe (Punkte)	Status	Referenzen	Zugewiesen (Name)	Gearbeitete Stunden	Schätzung der verbleibenden Stunden
Essence Navigator	niedrig	2	ausstehend	Essence Navigator Bilder	Jill3, Helene3e	0	1
Wireframes definieren	hoch	3	in Arbeit	Wireframes	alle	3	3
Wireframes ausarbeiten	mittel	8	in Arbeit	Wireframes	alle	3	8
Architecture notebook	mittel	3	in Arbeit	Architecture Notebook	Stefano-96	1	3

4.4. Probleme (optional)

Problem	Status	Notizen
x	x	x

4.5. Bewertungskriterien

Beispiele:

- Essence Navigator Bilder wurden erstellt oder nicht
- Wireframes erhalten positive Rückmeldung vom Stackholder

4.6. Assessment

Assessment Ziel	Iteration 2 fertigstellen
Assessment Datum	17.01.2020
Teilnehmer	alle
Projektstatus	grün

- Beurteilung im Vergleich zu den Zielen
- Geplante vs. erledigte Aufgaben
- Beurteilung im Vergleich zu den Bewertungskriterien
- Andere Belange und Abweichungen

5. Iterationsplan Iteration 3

Mauritius Berger <mauritius.berger@htw-dresden.de> 0.1, 13.01.2020 :toc: :toc-title:
 Inhaltsverzeichnis :sectnums: :icons: font

5.1. Meilensteine

Meilenstein	Datum
Beginn der Iteration	20.01.2020
Fertigstellen der Aufgaben	-
Ende der Iteration	31.01.2020

5.2. Wesentliche Ziele

Beispiele:

- Beheben der Usability-Probleme, die von Abteilung X berichtet wurden
- Ausliefern der Hauptszenarios, die die Integration mit System Y zeigen *
- Present a technical demonstration (demo).

5.3. Aufgabenzuordnung

Die in dieser Iteration geplanten Aufgaben sind in der Work Items List dargestellt (hier Verweis einfügen).

alternativ: Die folgenden Aufgaben werden in dieser Iteration bearbeitet:

Aufgabe bzw. Beschreibung	Priorität	Schätzung der Größe (Punkte)	Status	Referenzen	Zugewiesen (Name)	Gearbeitete Stunden	Schätzung der verbleibenden Stunden
Essence Navigator	niedrig	2	ausstehend	Essence Navigator Bilder	Jill3, Helene3e	0	1

Aufgabe bzw. Beschreibung	Priorität	Schätzung der Größe (Punkte)	Status	Referenzen	Zugewiesen (Name)	Gearbeitete Stunden	Schätzung der verbleibenden Stunden
Use Cases definieren	hoch	3	in Arbeit	Vision	alle	0	3
Use Cases ausarbeiten	mittel	8	ausstehend	x	alle	0	8

5.4. Probleme (optional)

Problem	Status	Notizen
x	x	x

5.5. Bewertungskriterien

Beispiele:

- 97% der Testfälle auf Systemebene sind erfolgreich.
- Gemeinsame Inspektion des Iterations-Ergebnisses (Inkrement) mit den Abteilungen X und Y ergibt positive Rückmeldung.
- Technische Präsentation / Demo erhält positive Rückmeldungen.

5.6. Assessment

Assessment Ziel	Iteration 2 fertigstellen
Assessment Datum	17.01.2020
Teilnehmer	alle
Projektstatus	Zum Beispiel ausgedrückt als rot, gelb oder grün.

- Beurteilung im Vergleich zu den Zielen
- Geplante vs. erledigte Aufgaben
- Beurteilung im Vergleich zu den Bewertungskriterien
- Andere Belange und Abweichungen

Entwurfsdokumentation

1. Architecture Notebook <Projektthema>

Vorname Nachname <email@domain.org>; Vorname2 Nachname2 <email2@domain.org>;
Vorname3 Nachname3 <email3@domain.org> 0.1, 01.11.2019 :toc: :toc-title: Inhaltsverzeichnis
:sectnums:

1.1. Zweck

Dieses Dokument beschreibt die Philosophie, Entscheidungen, Nebenbedingungen, Begründungen, wesentliche Elemente und andere übergreifende Aspekte des Systems, die Einfluss auf Entwurf und Implementierung haben.

1.2. Architekturziele und Philosophie

- Plattformunabhängigkeit
- Nutzung von überall durch webserverzugriff
- Unterschiedliche Benutzergruppen mit unterschiedlichen Rechten
- einfache Architektur

1.3. Annahmen und Abhängigkeiten

- (wenige Nutzer) ohne Informatikerausbildung oder ähnlichem
- Stura Server vorhanden
- LXC Container wird bereitgestellt

1.4. Architektur-relevante Anforderungen

- Nutzung freier Lizenzen
- Sicherung und Wiederherstellung von Systemzuständen gewährleisten (zeitlich automatisiert)
- Nutzungsrechtssystem → Benutzerklassen

1.5. Entscheidungen, Nebenbedingungen und Begründungen

1. Python seitens Auftraggeber gewünscht → Forderung akzeptiert, da Sinnhaftigkeit dieser Wahl erkannt. Python ist weit verbreitet und gut geeignet zum Programmieren von Webanwendungen.
2. SQL Lite damit..
 - kein eigener Server aufgesetzt werden muss und zur zusätzlichen Sicherung, denn die Daten liegen nicht auf einem Datenbankserver vor? TODO
 - einfach zu implementieren und zu verstehen

- Projektumfang eher klein → SQL Lite reicht aus
- 3. Django als Webframework, da ..
 - SQL Lite ist standard
 - in Python geschrieben - > leichte Anbindung
 - populär → Communityunterstützung, viele Entwickler können daran weiterarbeiten

1.6. Architekturmechanismen

Doku "[Concept: Architectural Mechanism](#)"

1. Webserver : Speichern der Daten auf einem Server auf den von außen zugegriffen werden kann. Gängiger Architekturmechanismus für die Lagerung von Datenbanken.
2. Relationales DBS: Am weitesten Verbreitet. Bringt die meisten Vorteile zur Verwaltung von gängigen Daten mit.
3. Container (LXC): Einfach anzudocken an bestehendes System

1.7. Wesentliche Abstraktionen

1. Datenhaltung in einer relationalen Datenbank
2. Objektorientierte Programmierung

1.8. Schichten oder Architektur-Framework

1.9. Architektursichten (Views)

Folgende Sichten werden empfohlen:

1.9.1. Logische Sicht

1. kritische Schnittstellen: LXC Container

1.9.2. Physische Sicht (Betriebssicht)

1. Webserver → LXC Container → Webanwendung ?

1.9.3. Use cases


(usecase-diagramm einfügen)

2. Wireframes

Browser

← →

C



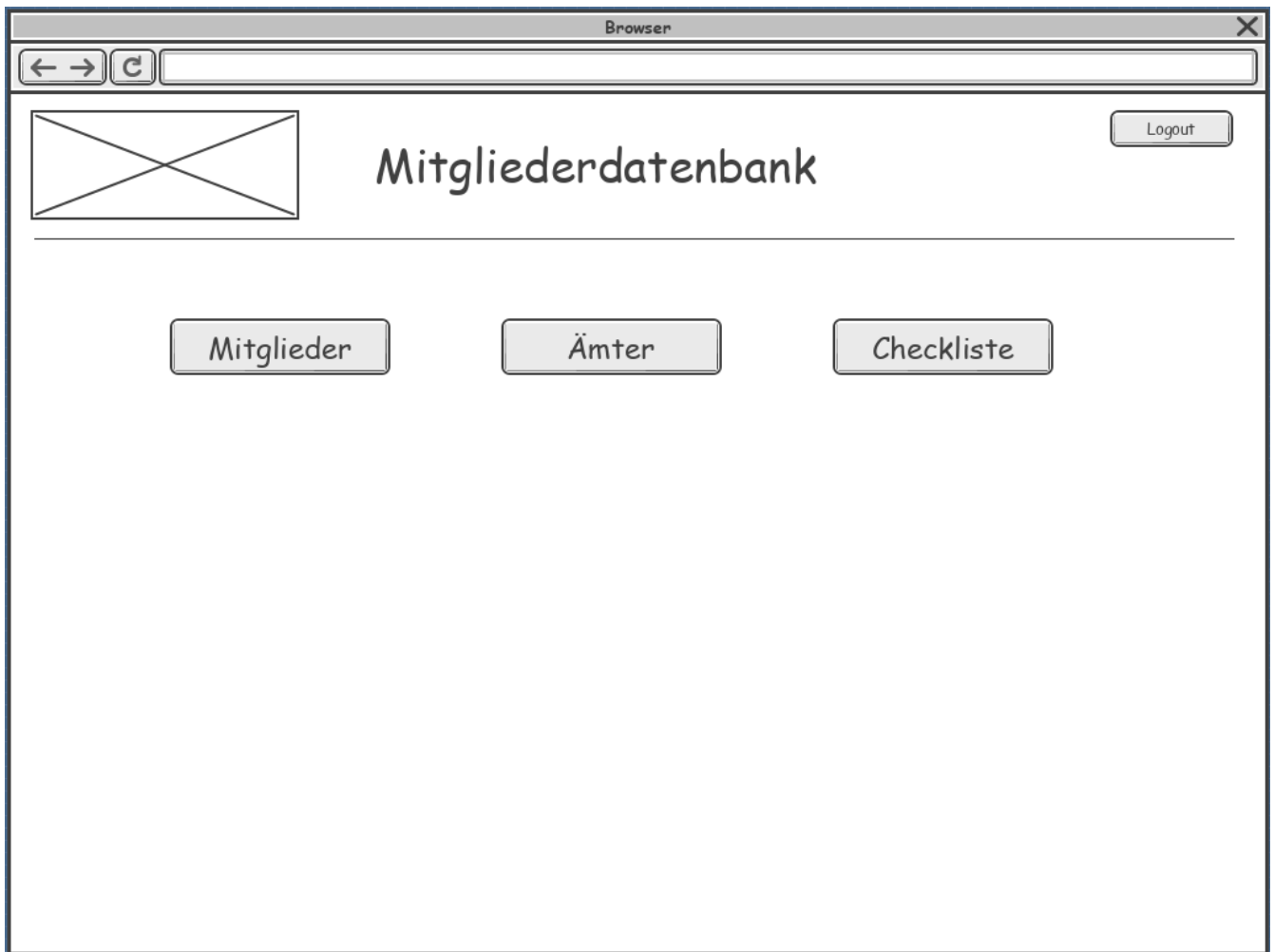
Mitgliederdatenbank

Benutzername

Passwort

Anmelden

[Passwort vergessen](#)



Browser

← →

↺

Mitgliederdatenbank

Änderungshistorie

Filtern nach ☒ Amt ☐ Mitglied

Suchen

Zeitpunkt	Benutzer	Änderung
05.01.2020, 16:37	Erika Musterfrau	Mitglied 0107 (Max Mustermann) gelöscht
05.01.2020, 14:52	Erika Musterfrau	Mitglied 0243 (John Doe) geändert
05.01.2020, 09:24	Erika Musterfrau	Amt 074 (Bereichsleitung Finanzprüfung) hinzugefügt
03.01.2020, 22:19	Erika Musterfrau	Amt 023 (Mitglied Kanzlei) geändert
03.01.2020, 21:53	Erika Musterfrau	Mitglied 0107 (Max Mustermann) gelöscht
03.01.2020, 08:14	Erika Musterfrau	Mitglied 0243 (John Doe) geändert
02.01.2020, 13:37	Erika Musterfrau	Amt 074 (Bereichsleitung Finanzprüfung) hinzugefügt
02.01.2020, 11:25	Erika Musterfrau	Amt 023 (Mitglied Kanzlei) geändert
01.01.2020, 11:44	Erika Musterfrau	Mitglied 0107 (Max Mustermann) gelöscht

1

2

3

...

17

Browser

← → ↺

Eintragsdetails

Mitglied 0243 (John Doe) geändert

X

Zeitpunkt: 05.01.2020, 16:37:42

Benutzer: Erika Musterfrau

Geänderte Attribute

Attribut	Vorher	Nachher
Name	Max Mustermann	John Doe
Geburtsdatum	01.01.1970	31.12.1999
Telefonnummer	0351 211 211	0815 118 333
E-Mail-Adresse	mmustermann@gmx.de	jdoe@gmx.de

Eintragsdetails

Mitglied 0107 (Max Mustermann) gelöscht

X

Zeitpunkt: 05.01.2020, 16:37:42

Benutzer: Erika Musterfrau

Gelöschter Datensatz

Attribut	Wert
Name	Max Mustermann
Geburtsdatum	01.01.1970
Telefonnummer	0351 211 211
E-Mail-Adresse	mmustermann@gmx.de

Eintragsdetails

Amt 074 (Bereichsleitung Finanzp...) hinzugefügt

X

Zeitpunkt: 05.01.2020, 16:37:42

Benutzer: Erika Musterfrau

Hinzugefügter Datensatz

Attribut	Wert
Bezeichnung	Bereichsleitung Finanzprüfung
Amtsinhaber	0243 (John Doe)
Workload	120 h
E-Mail-Adresse	finanzpruefung@stura.htw-dresden.de

Dokumentation Essence Navigator

Browser: <https://sg.sim4seed.org/navigator?endeavor=85> | Suchen | You are logged in as s79199@htw-dresden.de

HTWDD01 | Home | Alphas | Activities | Competencies | ToDos

Alphas the things to work with

Opportunity	(0/6)
Stakeholders	(0/6)
Requirements	(0/6)
Software System	(0/6)
Team	(0/5)
Work	(0/6)
Way of Working	(0/6)

Alphas Overview

© 2016 sim4seed.org

Browser: <https://sg.sim4seed.org/navigator/> | Meistbesucht | E-Mail – helene.uhlig... | Prime Video | Microsoft Office Home | OPAL - Online-Plattfor... | <https://www.youtube.c...> | übersetzer - Google-Suche | You are logged in as s77860@htw-dresden.de

HTWDD01 | Home | Alphas | Activities | Competencies | ToDos

Alphas the things to work with

Opportunity	IDENTIFIED (1/6)
Stakeholders	INVOLVED (3/6)
Requirements	CONCEIVED (1/6)
Software System	ARCHITECTURE SELECTED (1/6)
Team	COLLABORATING (3/5)
Work	(0/6)
Way of Working	PRINCIPLES ESTABLISHED (1/6)

