Fachhochschule Erfurt

Fachbereich Energie und Gebäudetechnik

**WohnGutWissen**

Systemarchitektur

Erstellt von:

Benjamin Swarovsky

Bearbeitungszeit: 01.02 2021 bis 28.02.2021

Inhaltsverzeichnis

[1 Einführung und Ziele 1](#_Toc65331819)

[1.1 Aufgabenstellung 1](#_Toc65331820)

[1.2 Zielbestimmung 1](#_Toc65331821)

[1.3 Qualitätsziele 1](#_Toc65331822)

[1.3.1 Sicherheit (Vertraulichkeit, Nicht-Abstreitbarkeit, Integrität) 1](#_Toc65331823)

[1.3.2 Wartbarkeit (Modularität) 1](#_Toc65331824)

[1.4 Stakeholder 2](#_Toc65331825)

[2 Randbedingungen 3](#_Toc65331826)

[3 Kontextabgrenzung 4](#_Toc65331827)

[3.1 Ebene 0 4](#_Toc65331828)

[4 Lösungsstrategie 6](#_Toc65331829)

[4.1 Allgemeine Architektur 6](#_Toc65331830)

[4.2 Frontend Komponenten 6](#_Toc65331831)

[4.3 Backend Komponenten 6](#_Toc65331832)

[4.4 Datenbank 6](#_Toc65331833)

[5 Bausteinsicht 7](#_Toc65331834)

[5.1 Ebene 1 7](#_Toc65331835)

[5.2 Ebene 2 9](#_Toc65331836)

[6 Laufzeitsicht 10](#_Toc65331837)

[7 Verteilungssicht 12](#_Toc65331838)

[8 Konzepte 13](#_Toc65331839)

[9 Entwurfsentscheidungen 15](#_Toc65331840)

[10 Qualitätsanforderungen 16](#_Toc65331841)

# Einführung und Ziele

## Aufgabenstellung

## Zielbestimmung

"Für das wohungswirtschaftliche Unternehmen "WohnGut" soll ein internes System realisiert werden, in welchem der Bestand aller Mietobjekte, die dazugehörigen Mietverträge sowie die zugehörigen Mieter verwaltet werden können. Der Projekttitel lautet "WohnGutWissen" (WGW).Im Bestand der WohnGut befinden sich ca. 1.000 Wohneinheiten und ca. 250 Büro- und Gewerbeeinheiten, welche an Privatpersonen und Unternehmen (jur.Personen) vermietet werden.Die Anwendung "WohnGutWissen" soll durch die Mitarbeiter des Unternehmensverwendet werden, um Mieter, Mietverträge, Mietobjekte erfassen und verwalten zu können. Weiterhin soll WohnGutWissen jedem Mieter die Überblicksinformationen über seine Mietverträge geben können. Dabei sollen alle Mieterverträge berücksichtigt werden, unabhängig davon, ob die Mietverträge beendet, laufend oder unterzeichnet aber noch nicht begonnen sind. Die Anwendung soll sowohl für die internen Mitarbeiter als auch die Mieter mittels Browser aufrufbar sein. In einer späteren Ausbaustufe soll WGW unter anderem um Funktionen erweitertwerden, die es dem Mieter auch erlaubt, bestehende Mietverträge zu kündigen oder Serviceanfragen für bestehende Mietobjekte auszulösen (z.B. Hausmeisterdienst bzgl. Defekt informieren). Das zu erstellende System muss diese geplanten Erweiterungen strukturell berücksichtigen. WohnGutWissen soll zum zentralen System innerhalb der WohnGut ausgebaut und über Jahre hinweg eingesetzt werden

## Qualitätsziele

### Sicherheit (Vertraulichkeit, Nicht-Abstreitbarkeit, Integrität)

Im System werden vertrauliche Kundendaten wie zum Beispiel Kontonummern, Gehaltsnachweise und Adressdaten erfasst und verwaltet. Werden diese Daten bei einem bösartigen Angriffen auf das System an unberechtigte Personen preisgegeben, kann für die Mieter und Wohngut ein hoher Finanzieller Schaden entstehen. Es muss daher zu jeder Zeit gewährleistet werden, dass nur Kunden und Wohngutmitarbeiter auf diese Daten zugreifen können. Jeder Kunde darf nur auf seine eigenen Daten zugreifen und erhält entsprechende Berechtigungen. Wohngutmitarbeiter erhalten nur Zugriff auf Daten, für deren Bearbeitung sie jeweils berechtigt wurden. Sollte ein Angriff Ausfallzeiten verursachen, muss die Verfügbarkeit des Systems spätestens nach 24 Stunden wiederhergestellt werden. Vertragsdaten können auch nach mehreren Jahren noch abgerufen werden und dürfen auf keinen Fall verloren gehen. Das System soll ein versehentliches Löschen solcher Daten verhindern.

### Wartbarkeit (Modularität)

Das System soll über 10 Jahre hinweg ausgebaut werden. Daher soll die Möglichkeit geboten werden, mit möglichst geringem Aufwand neue Komponenten hinzuzufügen. Das System wird im Laufe der Zeit deutlich wachsen, und soll dabei möglichst gut wartbar bleiben. Neue Komponenten sollen bereits vorhandene Komponenten möglichst wenig beeinflussen.

## Stakeholder

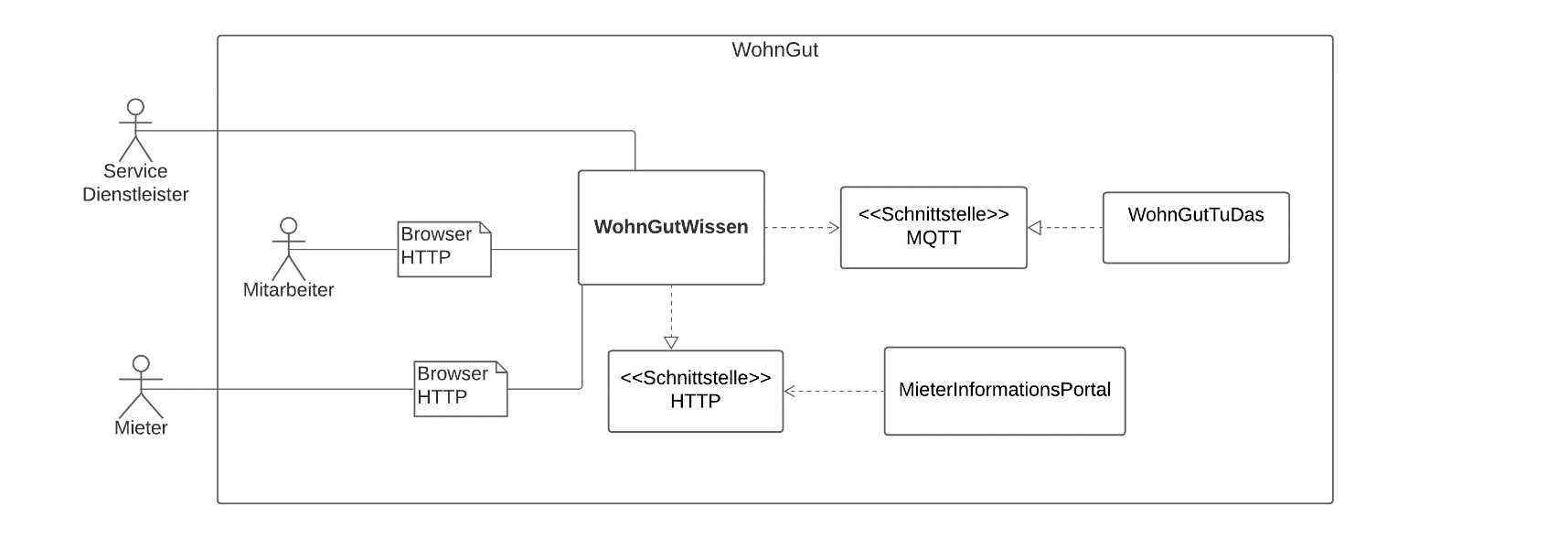
|  |  |
| --- | --- |
|  |  |
| Rolle | Erwartungshaltung |
| Wohngut  Mitarbeiter  (Nutzer) | erfassen und verwalten Mieter, Mietverträge und Mietobjekte im System |
| Mieter (Nutzer) | Mieten Objekte der WohnGut  erhalten überblick über ihre Mietverträge |
| Service Dienstleister | Erhalten in einer späteren Ausbaustufe Serviceanfragen vom System wie |
| Wohngut-EDV Abteilung |  |
| Behörden |  |
| Projektleiter WohnGut |  |

# Randbedingungen

|  |  |
| --- | --- |
| Systemart | Webbasierte Anwendung |
| Zugriffsakteure | Mitarbeiter des wohungswirtschaftlichen Unternehmens und Mieter |
| Datenbankmangementsystem | Die Daten müssen in ein Microsoft MS SQL Datenbank gespeichert werden, da die WohnGut-EDV-Abteilung bereits einen solchen Server betreibt. |
| Umsetzungssprache | Java |
| Ausführungsserver | Linux-basiert |
| Laufzeitumgebung | kommerziell nutzbares und gepflegtes JDK |
|  |  |
|  |  |

# Kontextabgrenzung

## Ebene 0



**WohnGutTuDas**

* Erhält von WGW eine Benachrichtigung, wenn ein Mietvertrag ausläuft
* Der Datenaustausch erfolgt über eine nachrichtenbasierte Schnittstelle

**MieterInformationsportal**

* WGW stellt für das MieterInformationsPortal eine Exportfunktion per HTTTP-Schnittstelle zur Verfügung
* Ausgegeben werden alle Mieter, Mietobjekte und Mietverträge
* Die Daten werden per JSON bereitgestellt

**Mieter (Nutzergruppe)**

* Greifen über einen Browser per HTTP auf das System WGW zu
* Mieter (Juristische Personen) erhalten vom System Überblick über Ihre Mietverträge
  + unabhängig davon, ob die Mietverträge beendet, laufend oder unterzeichnet aber noch nicht begonnen sind
* Pro Mieter wird eine E-Mailadresse erfasst
  + Identifizierung bei Systemanmeldung

**Mitarbeiter (Nutzergruppe)**

* Greifen über einen Browser per HTTP auf das System WGW zu
* Erfassen über das System Mieter, Mietobjekte, und Mietverträge

**Servicedienstleister (Nutzergruppe)**

* In einer späteren Ausbaustufe (noch nicht mit Auftraggeber abgestimmt) erhalten Servicedienstleister wie z.B. Hausmeisterdienste Serviceanfragen, nachdem diese per WGW vom Mieter erstellt und ausgelöst wurden.

# Lösungsstrategie

## Allgemeine Architektur

Beim WGW handelt es sich um ein ERP-System, welches über mindestens 10 Jahre innerhalb der WohnGut ausgebaut wird. Daher spielt die Erweiterbarkeit um verschiedene Module eine große Rolle. Damit neue Module möglichst unabhängig von anderen Komponenten entwickelt und angepasst werden können, wird eine Serviceorientierte Architektur umgesetzt. Dadurch wird eine schnelle Integrierung neuer Module ermöglicht. Die Einzelnen Module werden jeweils in Form einer Schichtenarchitektur realisiert. Diese sollen dadurch möglichst schnell und einfach entwickelt werden können. Die Modularisierung ermöglicht es einzelnen Modulen eigene Entwicklerteams zuzuordnen. Dadurch kann ein solches Team mit gebündelten wissen möglichst effizient entwickeln. Die einzelnen Entwicklerteams stehen in engen Kontakt zueinander. Das Gesamtsystem muss konsistent bleiben.

## Frontend Komponenten

Aufgrund der voneinander stark abweichenden Berechtigungen gibt es für Mieter, WohnGut Mitarbeiter (und später Service Dienstleister) verschiedene Frontends. Diese sind MieterUI, WohngutUI (und ServicDienstleisterUI). Dadurch können für die verschiedenen Nutzergruppen verschiedene Zugriffsmöglichkeiten gewährleistet werden. Mieter können von Desktop PC, Tablett und Handy über die MieterUI auf das System zugreifen. Wohngut Mitarbeiter erhalten per WohngutUI aufgrund hoher Sicherheitsanforderungen nur über das Firmeninterne VPN über einen Arbeitsplatz-PC Zugriff auf das System.

## Backend Komponenten

Das Backend wird unterteilt in die Module Mieterverwaltung, Mietvertragsverwaltung, Mietobjektverwaltung. Diese werden möglichst unabhängig voneinander gehalten und entwickelt.

## Datenbank

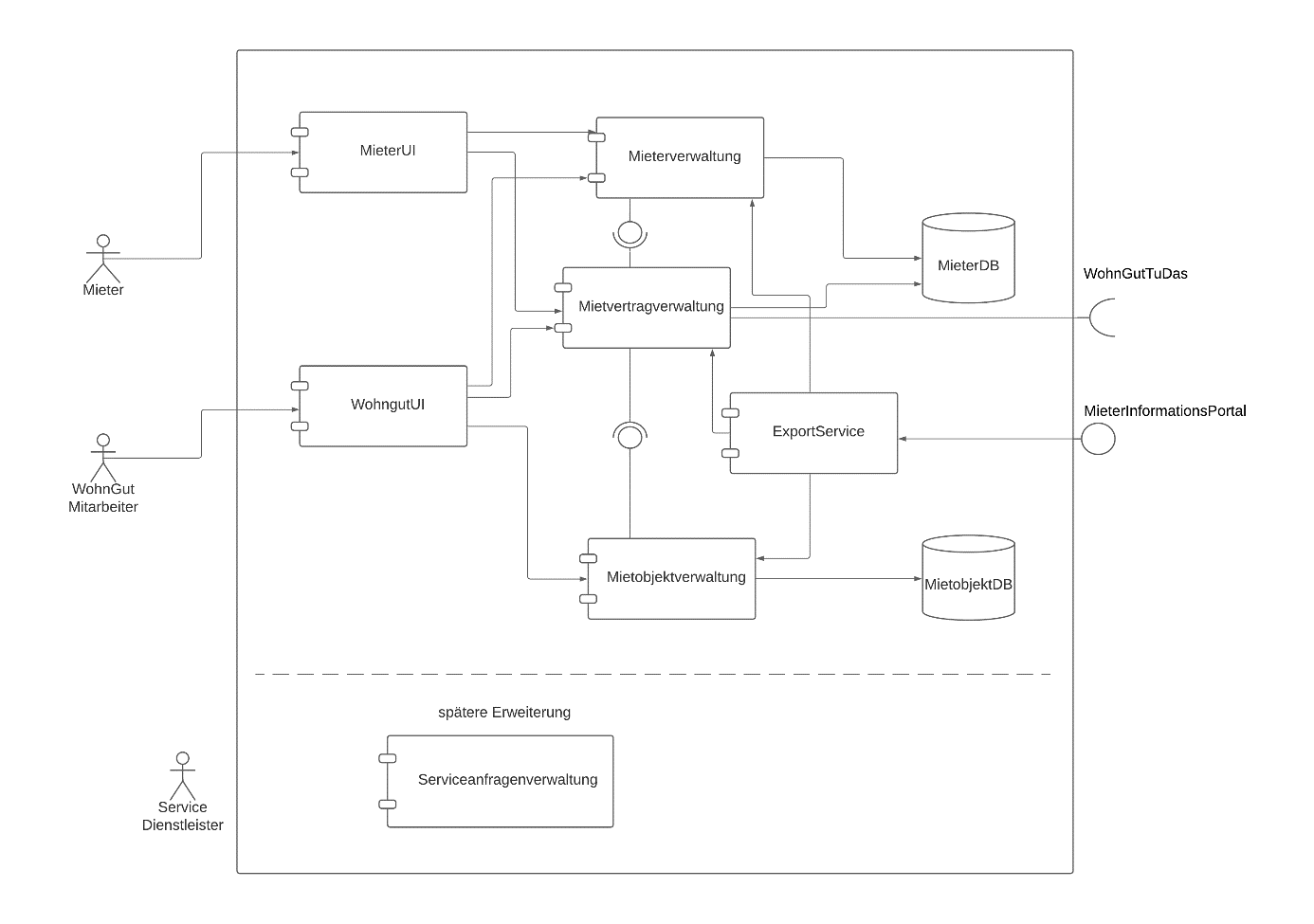
Die Komponenten Mieterverwaltung und Mietvertragsverwaltung greifen auf die Datenbank MieterDB zu.

Mietobjekte werden unabhängig vom Mieter und Mietvertrag gehalten. Deshalb werden die Daten in der Datenbank MietobjektDB gespeichert.

Im Mietvertrag wird ein notwendiger Teil des Mietobjektes redundant gespeichert. Ein Mietobjekt kann so beispielsweise unabhängig vom Vertrag geändert oder gelöscht werden. Die Objektdaten bleiben trotzdem im Mietvertrag erhalten.

# Bausteinsicht

## Ebene 1



**Mietobjektverwaltung**

Im WGW sollen alle Mietobjekte der WohnGut von Wohngutmitarbeitern erfasst werden können. Mietobjekte können in neuen Mietverträgen aufgenommen werden. Die benötigten Daten werden redundant im Vertrag gespeichert. Ziel ist es, dass Mietobjekte unabhängig vom Mietvertrag und Mietern bearbeitet oder gelöscht werden können.

**Mietvertragsverwaltung**

Im System werden alle Mietverträge der WohnGut erfasst und verwaltet. Mietverträge werden immer genau einem Mietobjekt und einem Hauptmieter zugeordnet. Neben dem Hauptmieter können bis zu drei Nebenmieter in den Mietvertrag eingetragen werden. Der Mietvertrag hat immer einen Vertragsbeginn. Das Vertragsende ist im allgemeinen Zustand Offen. Beim Auslauf des Mietvertrages wird das System WohnGutTuDas darüber informiert.

Den internen Mitarbeitern ist es möglich, die Kündigung oder Beendigung eines Mietvertrages unter Angabe eines Enddatums und ggf. eines Kommentars zu erfassen.

Mietverträge können von Wohngutmitarbeitern erfasst und bearbeitet werden.

Mieter können laufende Mietverträge einsehen und in einer späteren Erweiterung Kündigen.

**Mieterverwaltung**

Die Mietobjekte der WohnGut werden sowohl an Privatpersonen als auch an Unternehmen vermietet. WGW muss beide Arten von Mietern unterstützen. Je Mieter, also Privatperson oder Unternehmen, wird eine E-Mailadresse erfasst, welche auch zur Identifizierung bei der Anmeldung am System verwendet wird.

Mieter können vom WohnGut Mitarbeiter erfasst, bearbeitet und gelöscht werden.

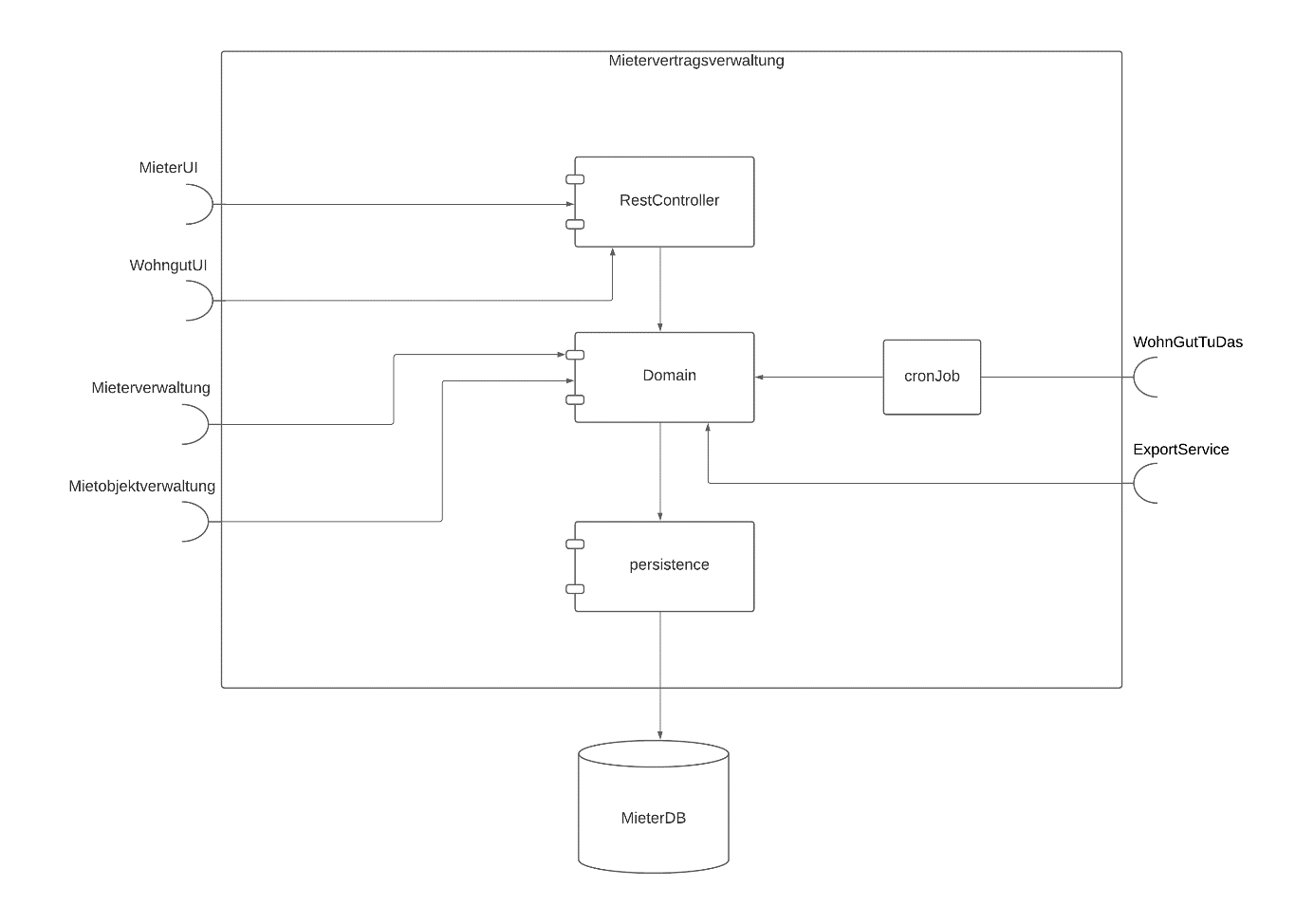
**Exportservice**

Die Übertragung aller Mieter, Mietobjekte und Mietverträge wird vom Exportservice umgesetzt. Dieser überträgt die benötigten Daten per JSON an das MieterInformationsPortal.

**Serviceanfragenverwaltung (Spätere Erweiterung, noch nicht mit Auftraggeber abgestimmt)**

Mieter können serviceanfragen für bestehende Mietobjekte auslösen (z.B. Hausmeisterdienst über Defekt informieren)

## Ebene 2



Die einzelnen Module Mieterverwaltung, Mietvertragverwaltung und Mietobjektverwaltung werden durch eine 3 Schichten Architektur realisiert.

Die Komponenten implementieren eine Schnittstelle vom Exportservice. Der Exportservice sendet den Mieterinformationsportal alle Mietobjekte, Mieter, Mietvertäge.

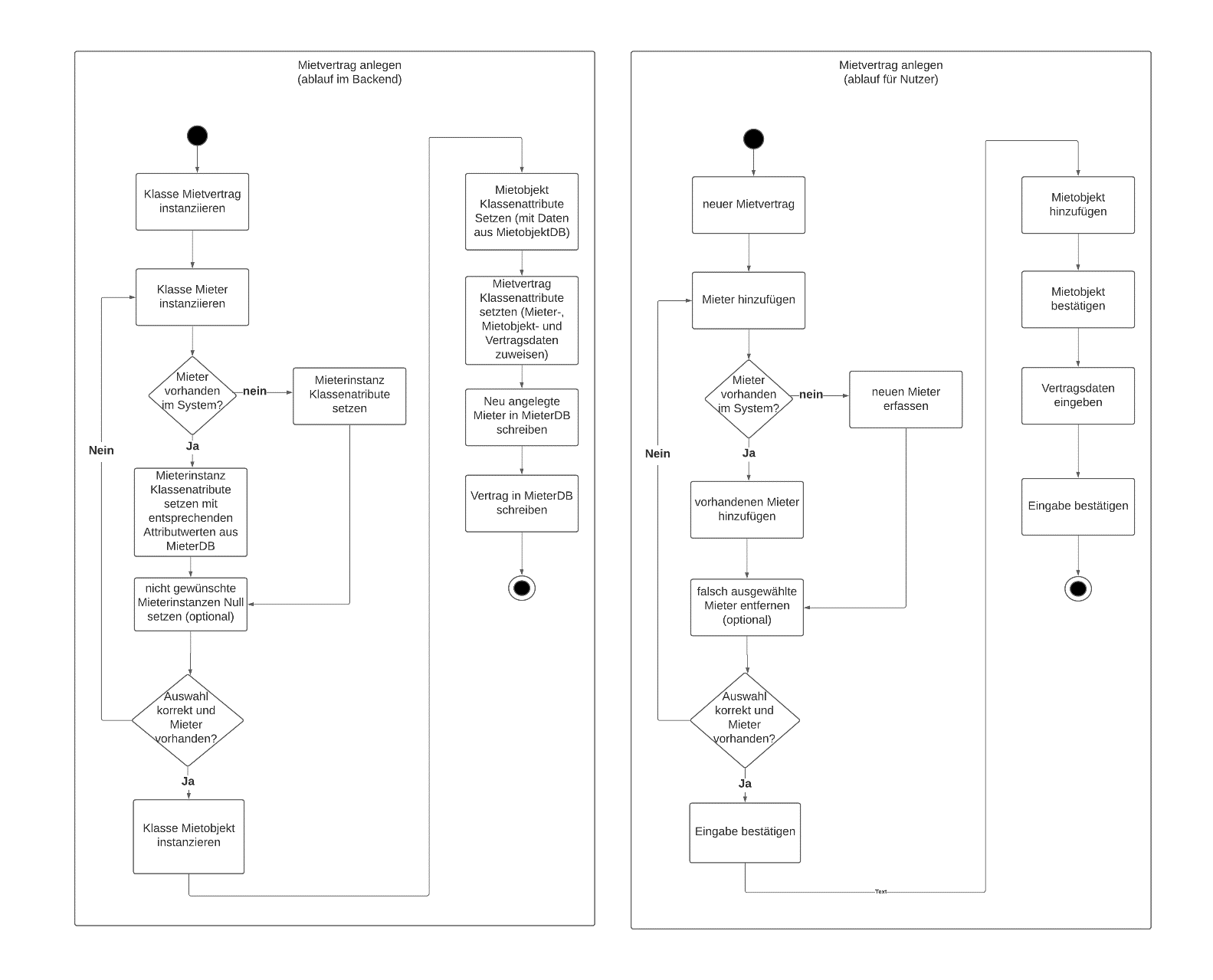
In der Mietvertragsverwaltung erfolgt die Benachrichtigung an WohnGutTuDas über einen CroneJob. Dieser stößt täglich eine Funktion in der Domain-Komponente an welche alle neu ausgelaufenen Mietverträge zurücksendet.

Module Mieterverwaltung und Mietobjektverwaltung sind unabhängig voneinander und vom Mietvertrag. Lediglich der Mietvertrag ist abhängig von den anderen Modulen. Daher stellen Mieterverwaltung und Mietobjektverwaltung jeweils eine eigene Schnittstelle für den Mietvertrag zur Verfügung.

Die einzelnen UI´s stellen jeweils eine eigene Schnittstelle zur Verfügung.

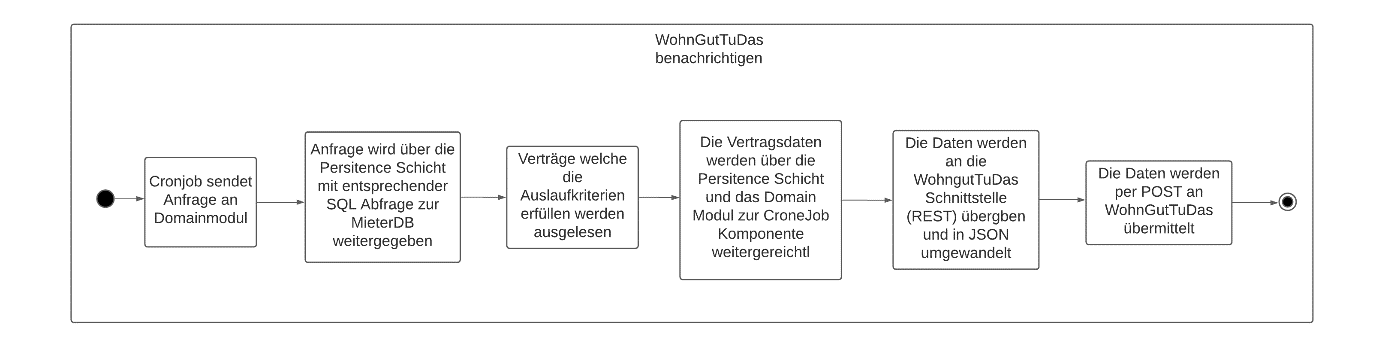
# Laufzeitsicht

## WohnGut Mitarbeiter legt Mietvertrag an:



Beim Anlegen eines Mietvertrages werden zuerst die Objekte Mietvertrag und Mieter, instanziiert. Beim Instanziieren eines neuen Mieters werden die Attribute anhand der eingegebenen Daten des Nutzers gesetzt. Bei bereits vorhandenen Mietern wird der Mieter anhand der E-Mail-Adresse in der MieterDB gesucht und den zu instanziierenden Objekt die entsprechenden Daten zugewiesen. Mieter, die vom Nutzer falsch zugeordnet wurden, können vom Nutzer verändert oder entfernt werden. Beim Entfernen wird die entsprechende Klasse NULL gesetzt. Nach der Bestätigung der Auswahl der Mieter wählt der Nutzer ein Mietobjekt anhand der Objekt-ID. Ein Objekt Mietobjekt wird instanziiert und die Klassenattribute werden anhand der entsprechenden Daten aus der MietobjektDB gesetzt. Im Anschluss gibt der Nutzer die Mietvertragsdaten in das Eingabeformular im Frontend ein. Die Klassenattribute des Mietvertragsobjektes werden mit den jeweiligen Nutzerdaten gesetzt. Weiterhin werden die Mieter und das Mietobjekt dem Mietvertrag zugeordnet. Bei der Bestätigung der Eingaben durch den Benutzer wird die Methode „save“ vom Mietvertrag aufgerufen. Die Methode speichert den Mietvertrag in der MieterDB. Neue Mieter und Vertragsdaten werden erst am Ende in die Datenbank geschrieben. Die Daten dürfen nur ganz oder gar nicht gespeichert werden damit die Datenbank konsistent bleibt. Nutzereingaben werden vom System validiert.

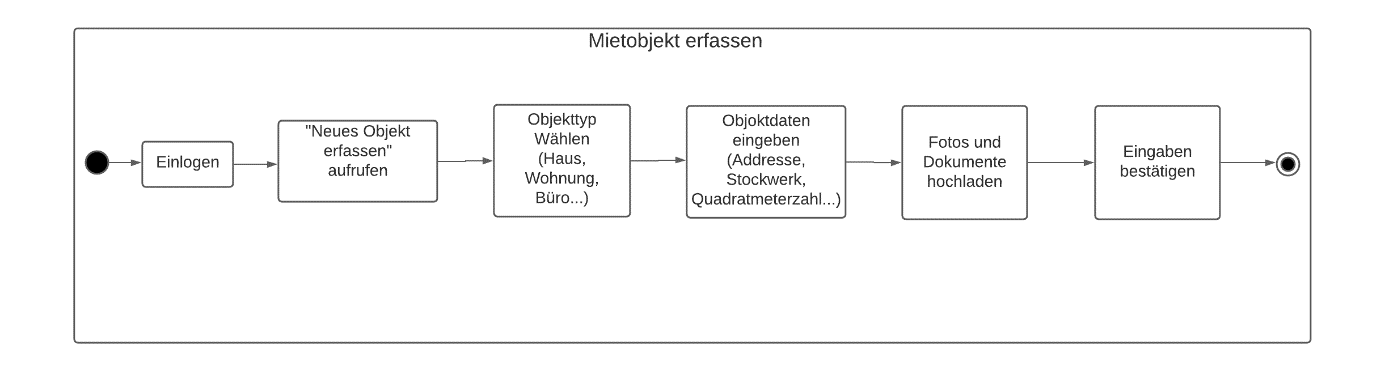
Auslaufende Mietverträge



Der Cronjob in der Mietvertragsverwaltung sendet täglich eine Anfrage an das Domainmodul. Die Anfrage wird durch die Schichten im Modul zur Persistance Schicht weitergegeben. Die Anfrage wird in SQL umgewandelt und eine SQL-Abfrage wird ausgeführt. Die zurückgegebenen Daten werden über die Schichten zum Crone Job Modul zurückgegeben. Die Anfrage war somit für den CroneJob erfolgreich. Im Falle eines Fehlers startet die Anfrage erneut. Die Daten werden an die WohnGutTuDas Schnittstelle weitergegeben und in JSON umgewandelt. Es handelt sich um eine Rest Schnittstelle, welche die Daten über eine POST Methode per https versendet.

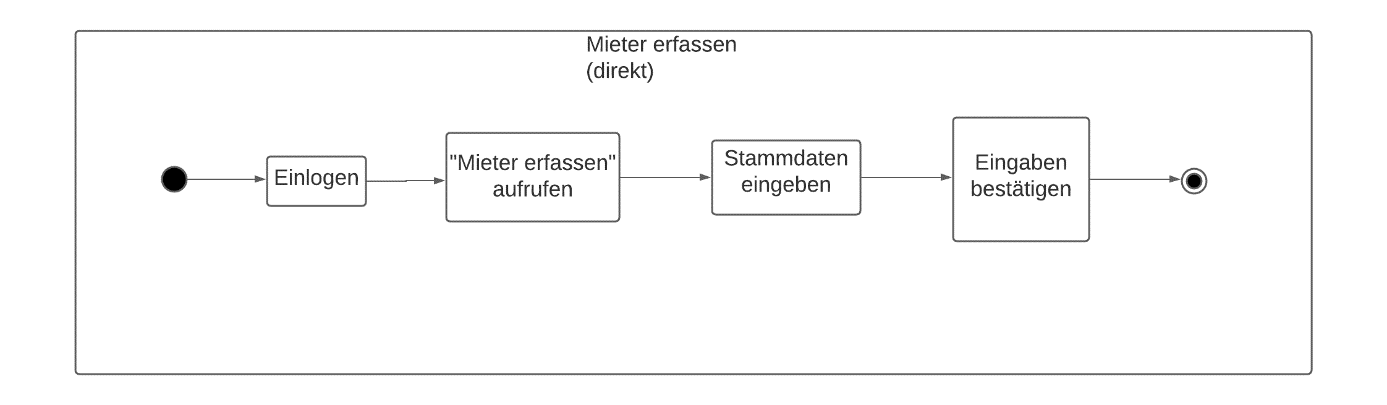
## Weitere Abläufe aus Sicht des Nutzers

WohnGut Mitarbeiter erfasst Mietobjekt:



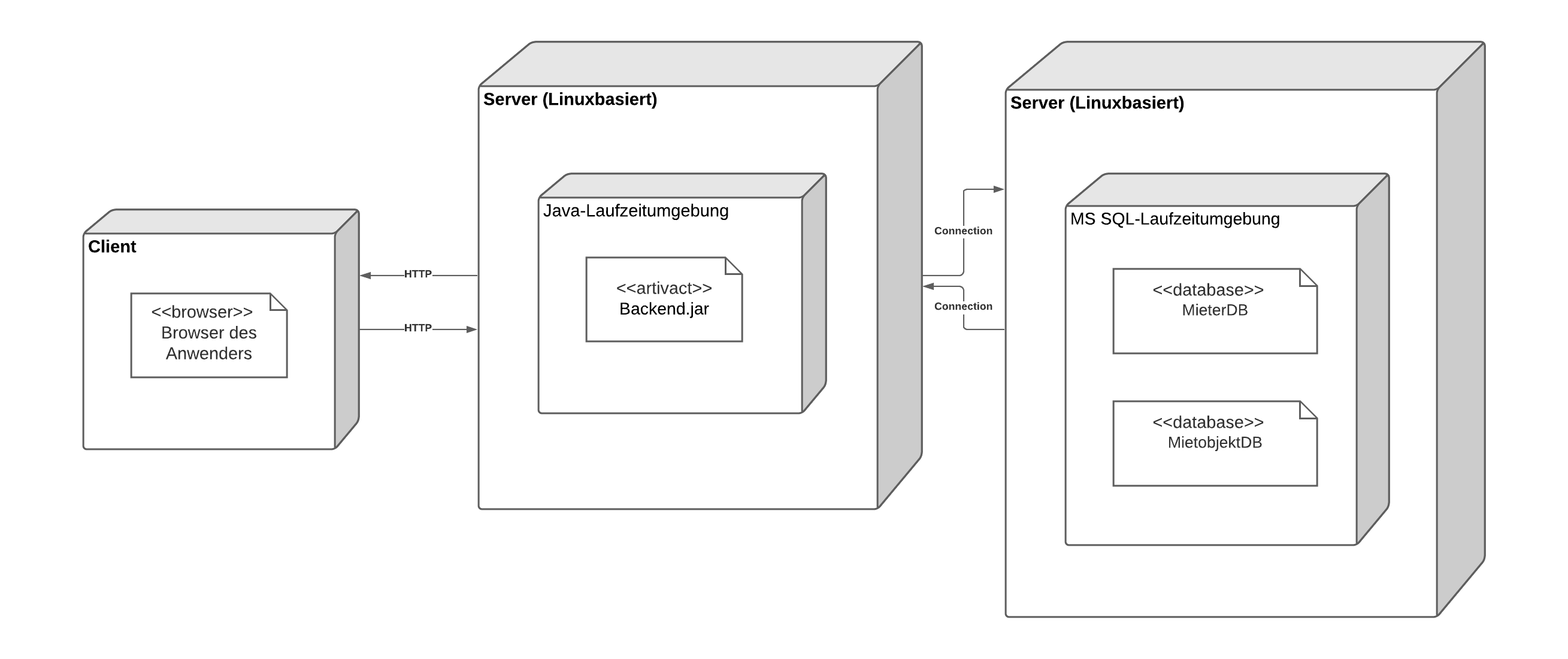
Der Angemeldete Nutzer ruft die Seite „Neues Objekt“ auf und wählt einen Objekttyp. Darauf hin erscheint eine Eingabemaske im Frontend mit den Eingabefeldern für den entsprechenden Objekttyp. Der Nutzer kann im Anschluss Fotos und Dokumente (z.B. Fotos des Objektes oder Nachweise der letzten Renovierung) hochladen. Eingaben werden vom System validiert. Bestätigt der Nutzer die Eingaben, dann werden die Daten in die MietobjektDB geschrieben.

Wohngut Mitarbeiter erfasst Mieter



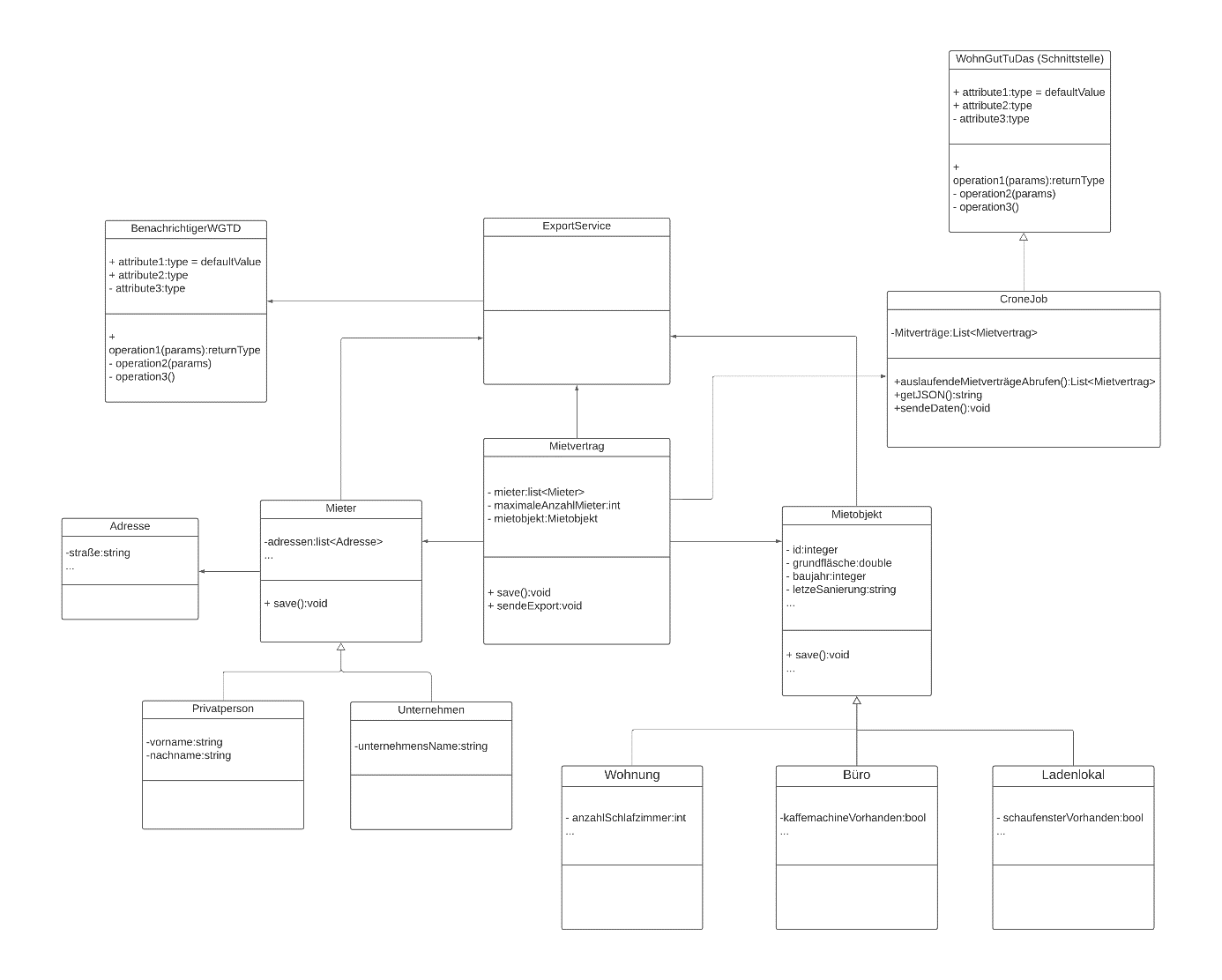
Benutzer können Mieter Direkt anlegen oder beim Erstellen eines Mietvertrages. Die Abbildung zeigt den Ablauf beim direkten Erfassen eines Mieters. Der Angemeldete Nutzer ruft die Seite „Mieter erfassen“ auf, gibt die Stammdaten ein (welche vom System validiert werden). Am Ende bestätigt der Benutzer die Eingabe. Sind alle eingaben Valide, werden die Daten in die MieterDB geschrieben.

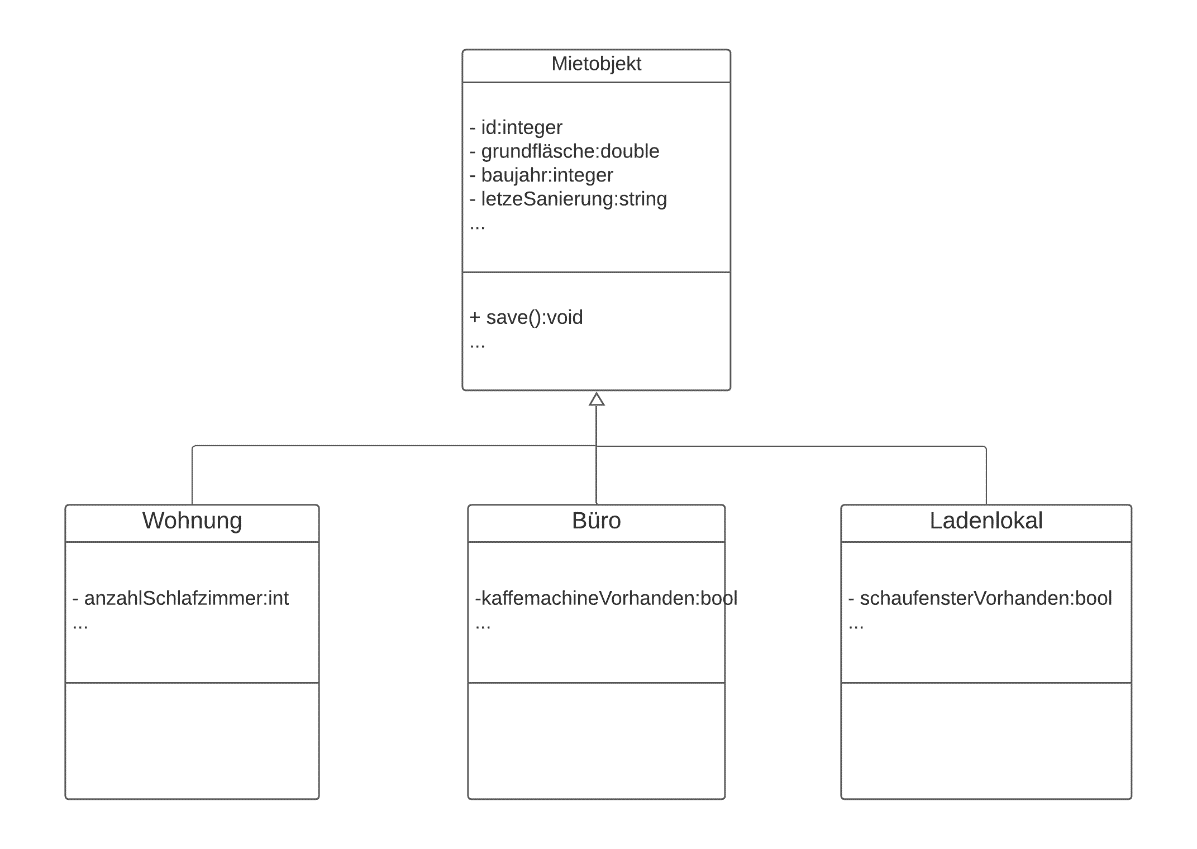
# Verteilungssicht



Das WGW Backend ist in der Programmiersprache Java umzusetzen. Der Ausführungsserver wird auf Basis von Linux betrieben. Als Laufzeitumgebung darf nur ein kommerziell nutzbares und gepflegtes JDK zum Einsatz kommen. Der Betreiber des Servers sorgt für eine maximale Ausfallzeit von 24 Stunden. Im Falle eines Totalausfalls zum Beispiel bei einer Naturkatastrophe wird eine Sekundäre Instanz per Georeplikation in einem anderen Rechenzentrum erstellt.

# Konzepte

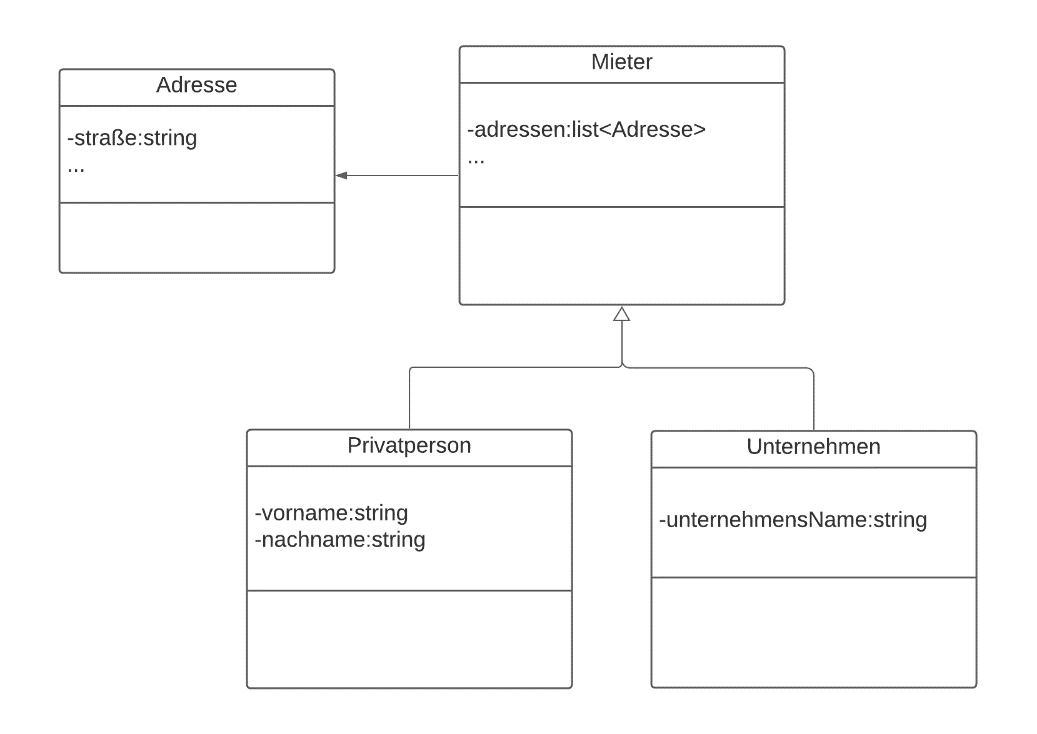




Mietobjekt

Mietobjekte werden im System als Wohnung, Büro oder Ladenlokal implementiert. Aufgrund von Gemeinsamkeiten erben diese Klassen von der Abstrakten Klasse Mietobjekt.

Mieter



Mieter werden im System entweder als Privatperson oder als Unternehmen implementiert, welche von der Abstrakten Klasse Mieter erben.

Domain Model

**Schnittstellen**

Die Backend Module Mieterverwaltung und Mietobjektverwaltung sind unabhängig voneinander und vom Mietvertrag. Lediglich der Mietvertrag ist abhängig von den anderen Modulen. Daher stellen Mieterverwaltung und Mietobjektverwaltung jeweils eine eigene Schnittstelle für den Mietvertrag zur Verfügung.

# Entwurfsentscheidungen

Bei WohngutWissen handelt es sich um ein ERP-System. Diese Systeme wie zum Beispiel SAP ERP zeichnen sich durch eine hohe Modularität aus. Bei SAP ERP handelt es sich um kein einzelnes großes Programm, sondern um eine Vielzahl von Komponenten welche nach und nach von einem Unternehmen zugekauft werden können. Das System soll zusammen mit dem Unternehmen Wachsen. Weil auch WohnGutWissen im Laufe der Zeit ständig ausgebaut werden soll, wurde sich bei der Architektur an der Modularität von SAP Orientiert. Damit bei einer langfristigen Anpassung der Systeme der Entwicklungsaufwand möglichst gering bleibt, wird eine Serviceorientierte Architektur umgesetzt. Ziel einer solchen Architektur ist es, kleine Module mit klaren Funktionen, Aufgaben und Schnittstellen zu definieren. Ein Nachteil dieser Architektur ist die erhöhte Verarbeitungszeit einzelner Aufgaben durch die lose Kopplung. Diesem Problem wird mit leistungsfähiger Hardware entgegengewirkt. Der Einsatz einer reinen Schichtenarchitektur wird vermieden, weil bei einer solchen Architektur bei Änderungen im schlimmsten fall alle Schichten angepasst werden müssen. Darunter würde die Wartbarkeit leiden. Demzufolge würden die Kosten für weitere Ausbaustufen im laufe der Zeit deutlich wachsen.

# Qualitätsanforderungen

# Risiken und technische Schulden

* Inkonsistenz: aufgrund der der Modularität besteht die Gefahr, dass beispielsweise in den verschiedenen Datenbanken inkonsistente Zustände entstehen. Weiterhin besteht ein Risiko, dass die Ergebnisse der verschiedenen Entwicklerteams welche jeweils nur an einem Modul arbeiten nicht aufeinander abgestimmt sind und nicht zusammen passen. Die Entwicklerteams müssen deshalb in ständigen kontakt stehen und sich regelmäßig gegenseitig die Ergebnisse präsentieren.
* Hohe Verarbeitungszeit

# Glossar