Fachhochschule Erfurt

Fachbereich Energie und Gebäudetechnik

WohnGutWissen

Systemarchitektur

Erstellt von:

Benjamin Swarovsky

Inhaltsverzeichnis

1 Einführung und Ziele

1.1 Aufgabenstellung

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 (juristische Personen) vermietet werden. Die Anwendung "WohnGutWissen" soll durch die Mitarbeiter des Unternehmens verwendet 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 erweitert werden, 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

1.2 Qualitätsziele

1.2.1 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 Angriff 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.

1.2.2 Wartbarkeit (Modularität)

Das System soll über Jahre hinweg ausgebaut werden. Daher wird die Möglichkeit geschaffen, 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.

1.3 Stakeholder

Rolle	Erwartungshaltung
Wohngut Mitarbeiter (Nutzer)	- erfassen und verwalten Mieter, Mietverträge und Mietobjekte im System - wollen möglichst wenig Eingaben tätigen und mit Eingabevorlagen arbeiten
Mieter (Nutzer)	- mieten Objekte der WohnGut - erhalten überblick über ihre Mietverträge - wollen ein möglichst intuitives System
Service Dienstleister	- erhalten in einer späteren Ausbaustufe Serviceanfragen vom System (z.B. Reperaturanfragen) - wollen eine möglichst genaue Auftragsbeschreibung
Wohngut-EDV Abteilung	- betreibt die Serverhardware in den Räúmlichkeiten der WohnGut - möchte möglichst zuverlässige Serverhardware
Behörden	- wollen Nachweise über die Einhaltung gesetzlicher Regelungen
Entwickler	- bearbeiten die Architekturaufgaben im Team nach dem Scrum Modell in 2-wöchigen Sprints - wollen möglichst klar definierte Arbeitsaufgaben
Projektleiter Entwicklerteam	- strebt eine sehr gute Zusammenarbeit unter den Entwicklerteams an - möchte alle Sprintziele erreichen
Projektleiter WohnGut	- möchte ein möglichst effizientes und ausfallsicheres System

2 Randbedingungen

Technisch

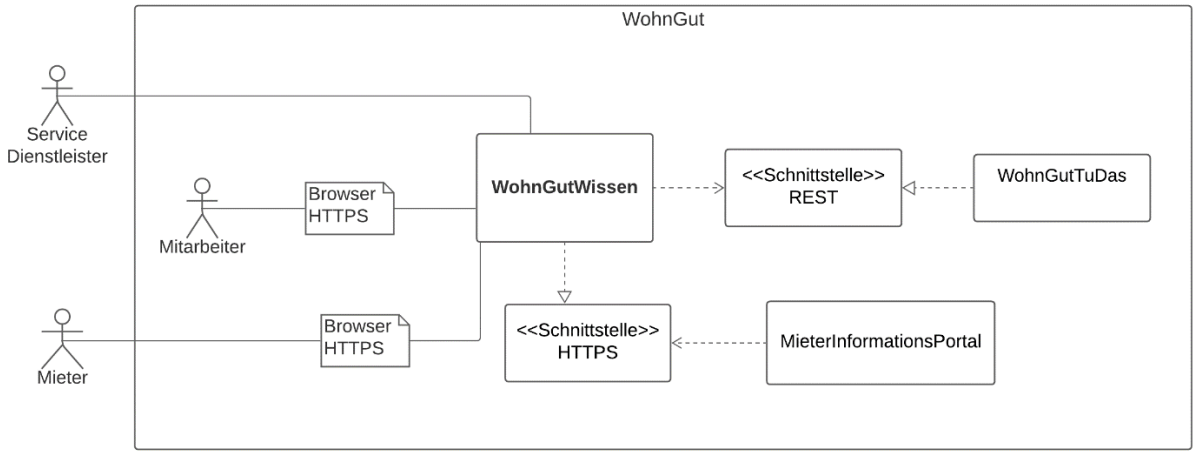
Systemart	webbasierte Anwendung
Zugriffsakteure	Mitarbeiter des Unternehmens WohnGut und Mieter
Datenbankmangementsystem	Die Daten müssen in eine Microsoft MS SQL Datenbank gespeichert werden, da die WohnGut-EDV-Abteilung bereits einen solchen Server betreibt.
Umsetzungssprache	Java
Ausführungsserver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

Organisatorisch

Vorgehensmodell	Scrum (2 wöchige Sprints)
Projektdauer (mit Systemausbau)	Mehr als 3 Jahre
Teamaufteilung	Jeder Entwickler wird einem Team zugeteilt. Jedes Team ist für die Entwicklung von einem oder mehreren Modulen verantwortlich. Es erfolgt eine regelmäßige Absprache unter den Entwicklerteams.
Teamleiter	Sorgen für die Koordinierung der einzelnen Entwicklerteams. Teamleiter stehen in engen Kontakt zueinander.

3 Kontextabgrenzung

3.1 Ebene 0



WohnGutTuDas

- Erhält von WGW eine Benachrichtigung, wenn ein Mietvertrag ausläuft
- Der Datenaustausch erfolgt über eine REST Schnittstelle

MieterInformationsportal

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

Mieter (Nutzergruppe)

- Greifen über einen Browser per HTTPS 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 HTTPS über eine VPN Verbindung 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.

4 Lösungsstrategie

4.1 Allgemeine Architektur

Beim WGW handelt es sich um ein ERP-System, welches über 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 möglichst schnell und einfach entwickelt werden können. Die Modularisierung ermöglicht es einzelnen Modulen eigene Entwicklerteams zuzuordnen. Auf diese Weise kann ein solches Team mit gebündelten wissen möglichst effizient entwickeln. Die einzelnen Entwicklerteams stehen in engen Kontakt zueinander. Das System soll für den Anwender in 95,5% der Zeit zugreifbar und nutzbar sein. Das Gesamtsystem muss konsistent bleiben.

4.2 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. Für eine erhöhte Vertraulichkeit erhalten Wohngut Mitarbeiter per WohngutUI nur über das Firmeninterne VPN über einen Arbeitsplatz-PC Zugriff auf das System. Später sollen noch weitere Nutzerschnittstellen implementiert werden können. Die Nutzeroberfläche wird in deutscher Sprache gehalten. Im Laufe der Zeit sollen weitere Sprachpakete hinzugefügt werden. Das WGW-Frontend muss für den Anwender in 95% der Fälle innerhalb von 1 Sekunde nach der Anfrage die angefragten Informationen darstellen, in 99,5% der Fälle innerhalb von 5 Sekunden. Deshalb wird auf Elemente mit langen Ladezeiten wie z.B. Hochauflösende Bilder verzichtet. Das Frontend wird möglichst auf reinen Text, Tabellen und Auflistungen beschränkt. Eine einzelne Seite darf nicht zu viele gleichzeitige Abfragen im Backend verursachen. Wiederverwendbare Objekte sollen nach Möglichkeit zwischengespeichert werden und nicht erneut geladen werden.

4.3 Backend Komponenten

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

4.4 Datenbank

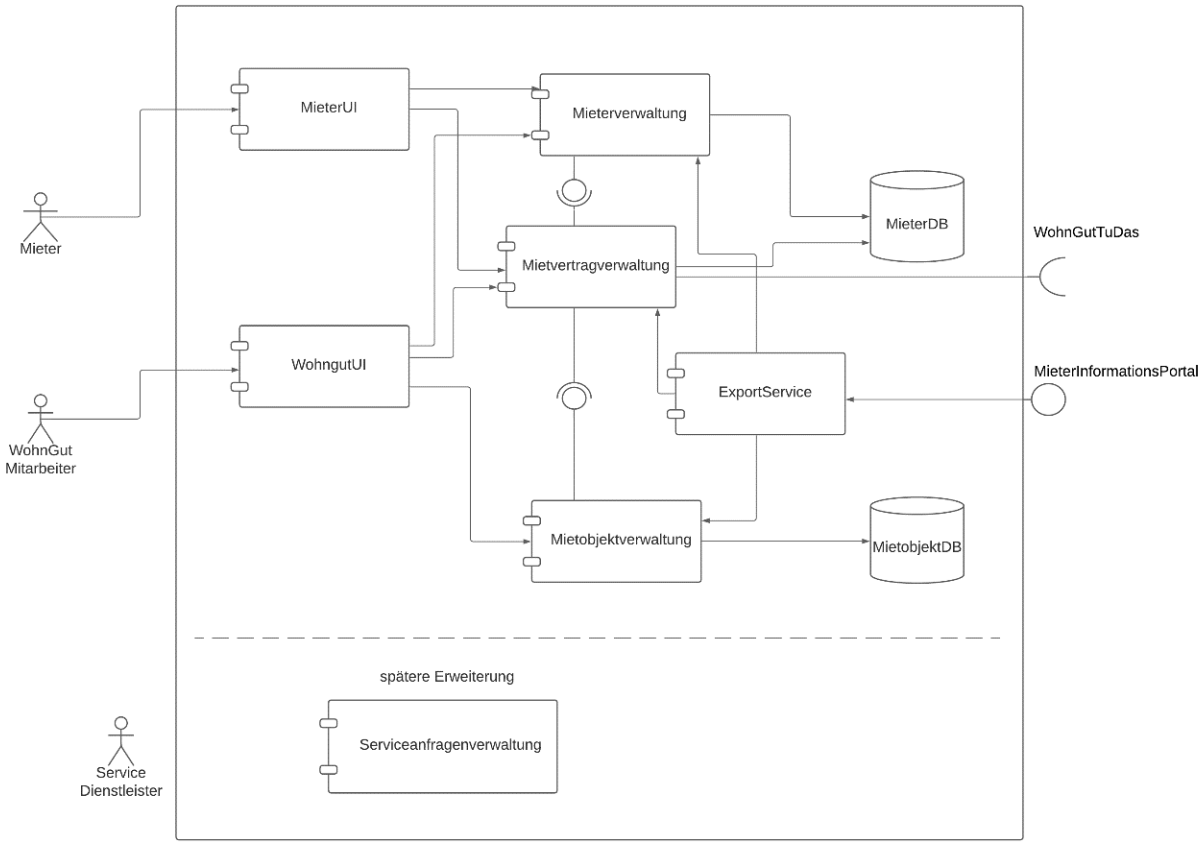
Die Komponenten Mieterverwaltung und Mietvertragsverwaltung greifen auf die Datenbank MieterDB zu.

Mietobjekte sollen unabhängig vom Mieter und Mietvertrag gehalten werden. 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. Abfragen über mehrere Datenbanken sollen Aufgrund der Anforderungen bezüglich der Reaktionsgeschwindigkeit nach Möglichkeit vermieden werden.

5 Bausteinsicht

5.1 Ebene 1



Mietobjektverwaltung

Im WGW werden alle Mietobjekte der WohnGut von Wohngutmitarbeitern erfasst. 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. Das Modul Mietobjektverwaltung greift auf die MietobjektDB zu

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.

Das Modul greift auf die MieterDB zu.

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.

Das Modul greift auf die MieterDB zu.

Exportservice

Die Übertragung aller Mieter, Mietobjekte und Mietverträge wird vom Exportservice umgesetzt. Dieser überträgt die benötigten Daten in Form von JSON über eine Rest Schnittstelle an das MieterInformationsPortal. Der Exportservice arbeitet dabei als Broker, an dem alle sendenden Module angemeldet werden. Wird ein Export vom MieterInformationsPortal beauftragt, dann sendet der Broker eine Anfrage an alle Teilnehmer und erhält die jeweiligen Daten zurück. Der Exportservice wandelt die Daten in JSON um und sendet diese gebündelt an das Mieterinformationsportal. Die Daten können auch einzeln bei einem Sendeauftrag von einem Teilnehmer, über den Exportservice zum MieterInformations-Portal gesendet werden.

Serviceanfragenverwaltung (Spätere Erweiterung, noch nicht mit Auftraggeber abgestimmt)

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

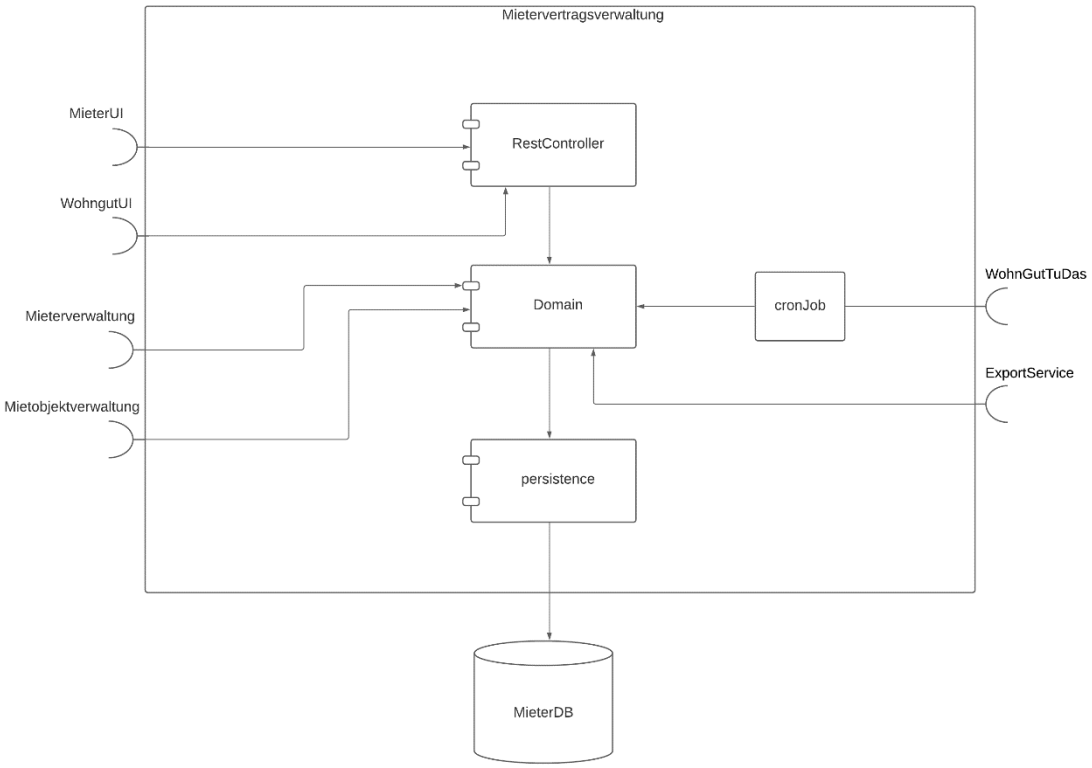
MieterUI

Die Frontendkomponente wird vom Mieter im Browser oder per App über HTTPS aufgerufen. Mieter haben in der Ursprungsversion nur Leserechte auf die zugreifbaren Daten. Die Komponente MieterUI kommuniziert per Rest API mit den Modulen Mietvertragsverwaltung und Mieterverwaltung und verlangt ein Server Client Modell.

WohngutUI

Die Frontendkomponente wird vom WohnGut Mitarbeiter vom Firmen PC über das Firmeninterne VPN per HTTPS aufgerufen. WohnGut Mitarbeiter haben Lese- und Schreibrechte auf die zugreifbaren Daten. Die Komponente WohngutUI komuniziert per Rest API mit den Modulen Mietobjektverwaltung, Mietvertragsverwaltung und Mieterverwaltung und verlangt ein Server Client Modell.

5.2 Ebene 2



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

Die Komponenten implementieren eine Schnittstelle vom Exportservice. Der Exportservice sendet den Mieterinformationsportal alle Mietobjekte, Mieter, Mietverträge im JSON Format .

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

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

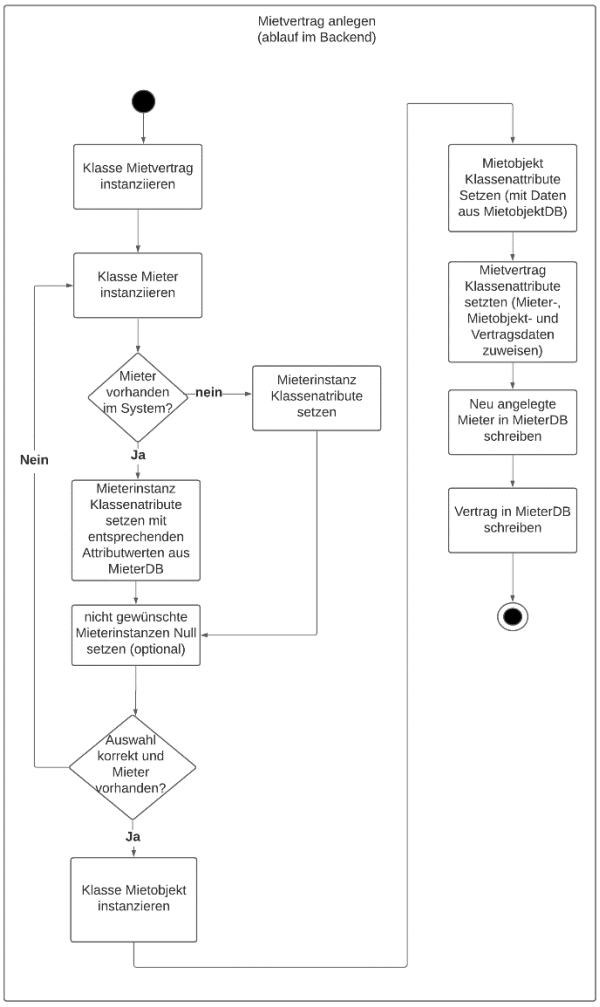
Die einzelnen UI´s stellen jeweils eine eigene REST Schnittstelle zur Verfügung. Der Zugriff erfolgt über das Modul Rest Controller.

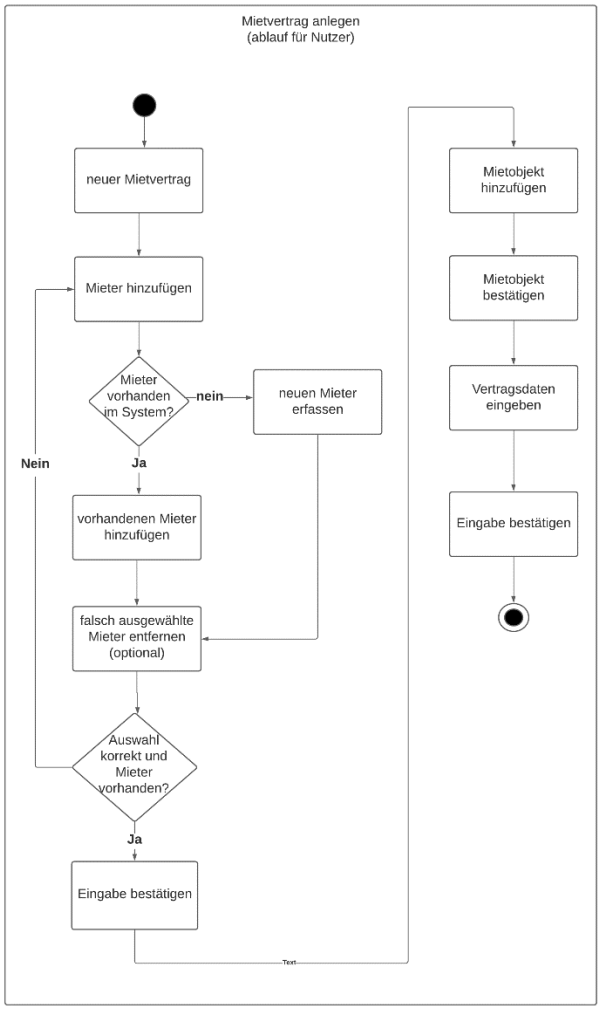
Das Domain Modul enthält die eigentliche Anwendungslogik mit entsprechenden Geschäftsereignissen und zugehörigen Entitäten.

Die Persistence Komponente stellt die Dauerhaftigkeit und Auffindbarkeit von Entitäten her.

6 Laufzeitsicht

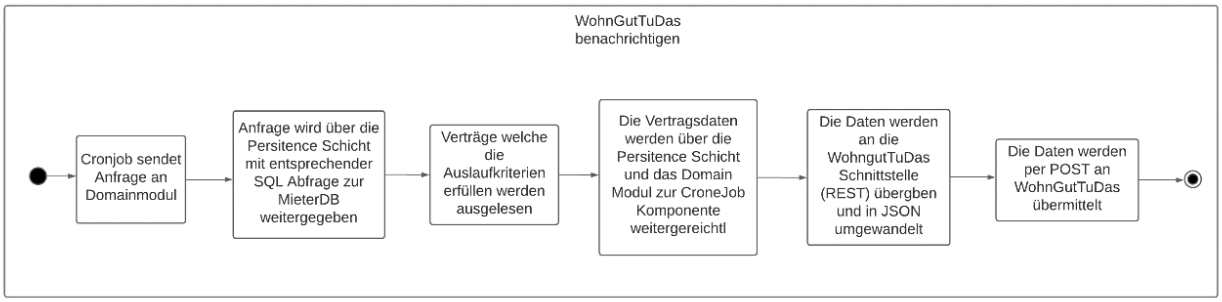
6.1 WohnGut Mitarbeiter legt Mietvertrag an:





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 Haupt und Nebenmieter 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.

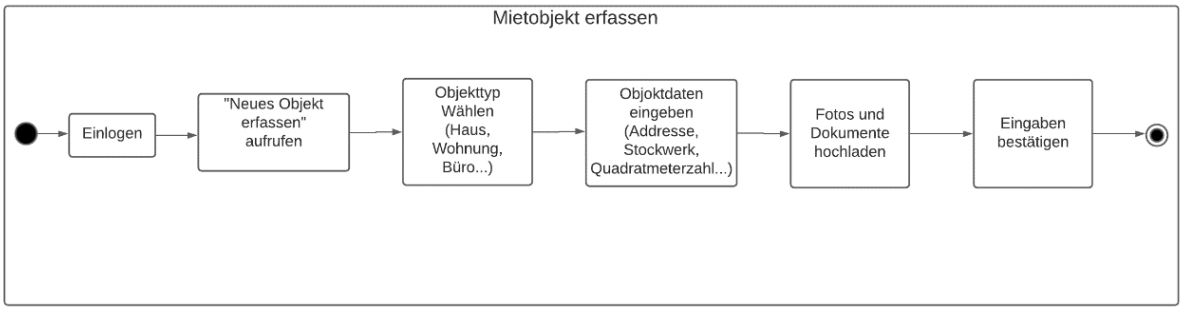
6.2 WohnGutTuDas Benachrichtigen



Der Cronjob in der Mietvertragsverwaltung sendet täglich eine Anfrage an das Domainmodul. Die Anfrage wird durch die Schichten der Mietvertragsverwaltung 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.

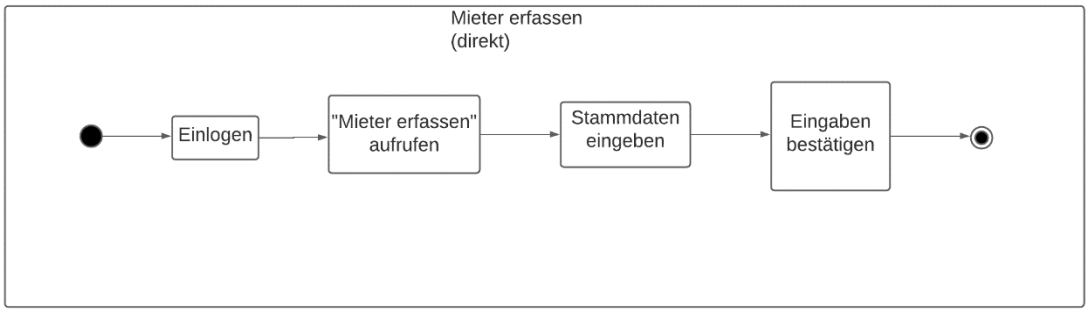
6.3 Weitere Abläufe aus Sicht des Nutzers

WohnGut Mitarbeiter erfasst Mietobjekt



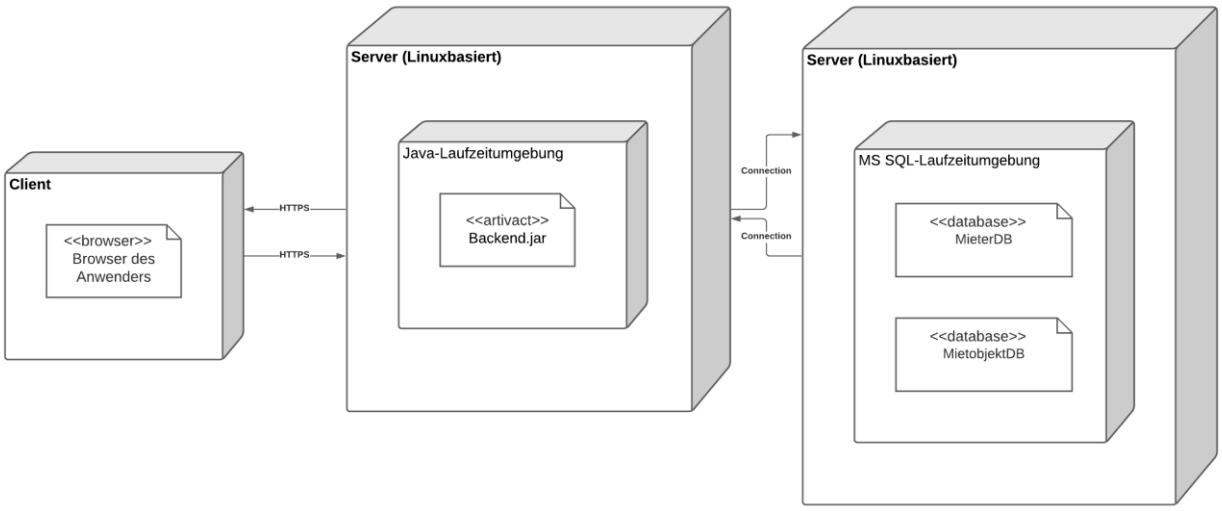
Der Angemeldete Nutzer ruft die Seite „Neues Objekt“ auf und wählt einen Objekttyp. Daraufhin erscheint eine Eingabemaske im Frontend mit den Eingabefeldern für Objektdaten für den entsprechenden Objekttyp. Der Nutzer kann im Anschluss Fotos und Dokumente (z.B. Fotos des Objektes oder Nachweise der letzten 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.

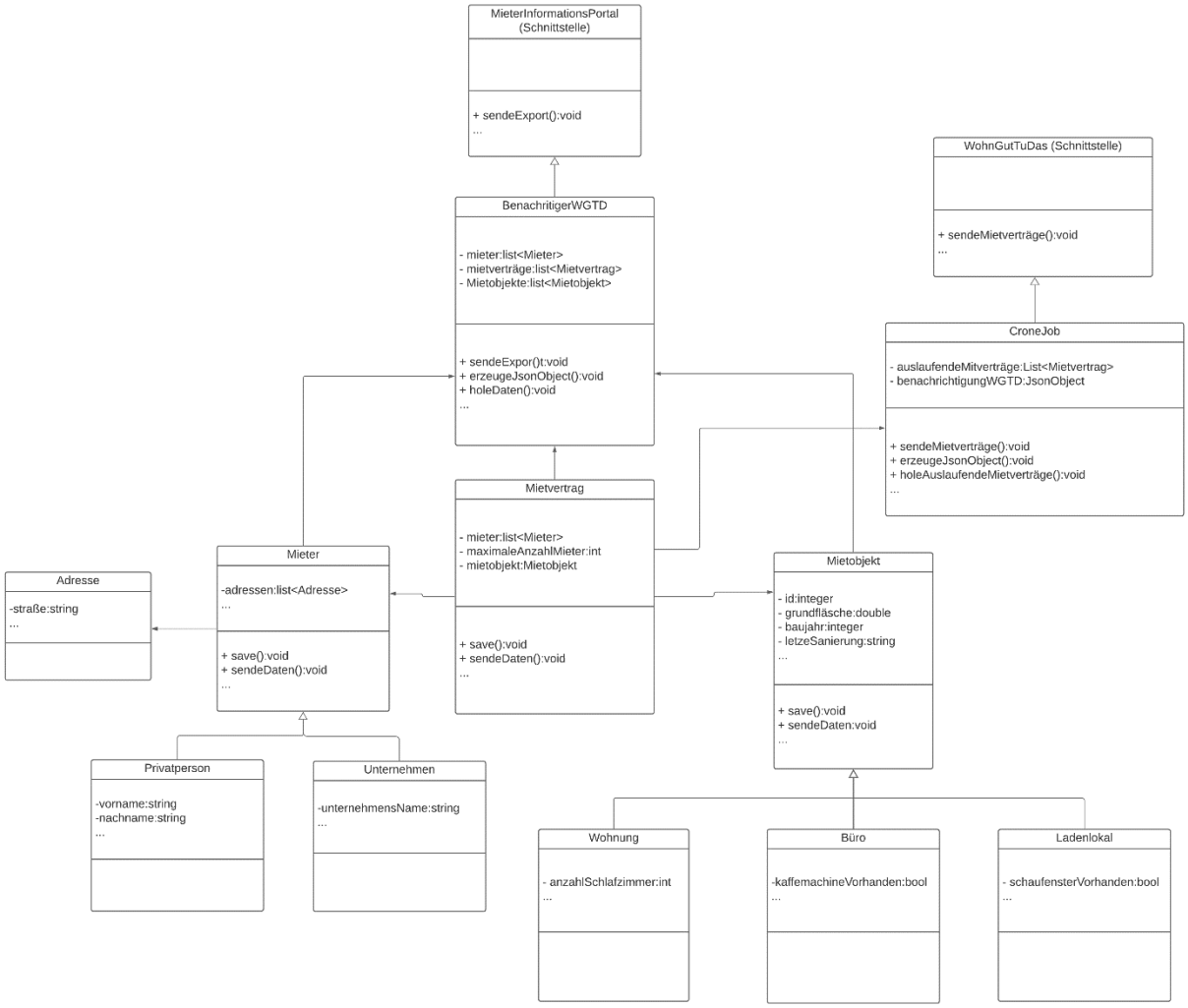
7 Verteilungssicht



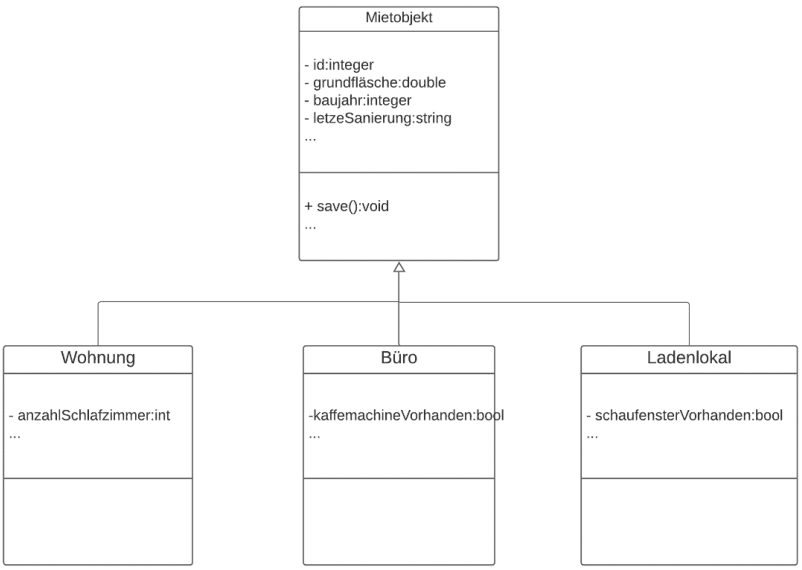
Die Serverhardware wird von der WohnGut EDV-Abteilung in einem Gebäude der WohnGut betrieben. Das WGW Backend ist in der Programmiersprache Java umzusetzen. Der Ausführungsserver wird auf Basis von Linux betrieben. Als Laufzeitumgebung darf nur ein kommerziell nutzbares und gepflegtes JDK zum Einsatz kommen. Der Server Hersteller gewährleistet 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.

8 Konzepte

Domain Model

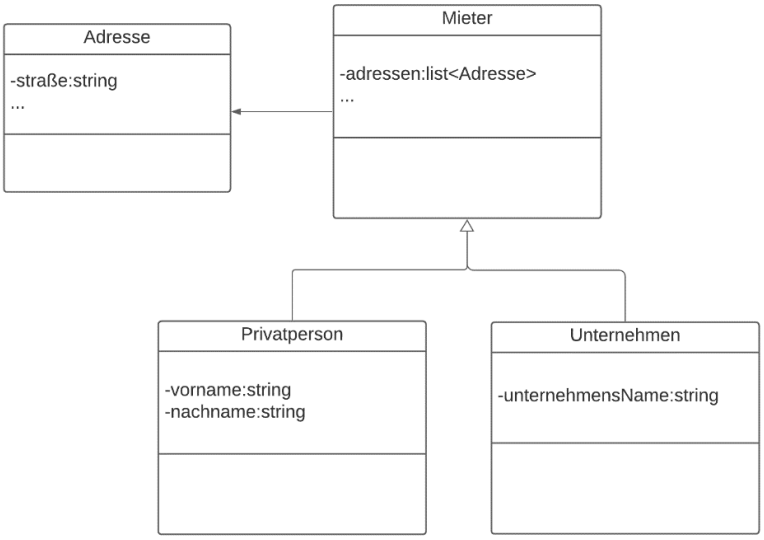


Mietobjekt



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.

Validierung

Für die Validierung kommt eine externe Validierungsbibliothek zum Einsatz.

Tests

Bei der Durchführung von Whiteboxtests für die Qualitätssicherung wird das Framework JUnit verwendet.

9 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 neues Modul wäre zum Beispiel die Serviceanfragenverwaltung welches man mit einer eigenen Datenbank (serviceanfragenDB) in das System integrieren könnte. Ein Nachteil dieser Architektur ist die erhöhte Verarbeitungszeit einzelner Aufgaben durch die lose Kopplung. Diesem Problem wird mit leistungsfähiger Hardware entgegengewirkt. Weil im Frontend keine Elemente mit hohem Speicherbedarf (z.B. hochauflösende Bilder oder Videos) geladen werden, wird die maximale Ladezeit von einer Sekunde in mehr als 95% der Fälle erreicht. 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. Eine Erweiterung der Serviceorientierten Architektur stellt die Microservice Architektur dar. Diese wäre allerdings zu aufwändig zu implementieren, deutlich schwerer zu handhaben und für den Projektumfang zu Mächtig.

11 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.
- Die Modularisierung bei einer Serviceorientierten Architektur kann zu einer erhöhten Verarbeitungszeit führen. Bei zu langsamer Hardware können die Ladezeiten des Frontends in 95% der Fälle mehr als eine Sekunde betragen. Bei einem stark wachsenden System könnte die Hardware nicht mehr schnell genug arbeiten. Um dem entgegen zu wirken sollen regelmäßige Leistungstests durchgeführt werden. Im Falle eines negativen Testergebnisses könnte die Hardware erneuert werden.

12 Glossar

Begriff	Erklärung
Modul / Komponente	Baustein eines Softwaresystems
Serviceorientierte Architektur	-flexibler Architekturstil -Grob-granulare fachliche Aufteilung der Ge samtapplikation in Services -Dienste werden zentral bereitgestellt und bei bedarf kombiniert
ERP	Enterprice Resource Planning bezeichnet die unternehmerische Aufgabe, Ressourcen wie Kapital, Personal, Betriebsmittel, Material und Informations- und Kommunikationstechnik im Sinne des Unternehmenszwecks rechtzeitig und bedarfsgerecht zu planen, steuern und verwalten.
cronJob	Ein cronjob (Cron-Daemon) dient der zeitbasierten Ausführung von Prozessen in Unix und unixartigen Betriebssystemen wie Linux.
Lose Kopplung	Geringer grad an Abhängigkeiten mehrerer Hard- und Softwarekomponenten
Georeplikation	der Speicher wird für eine bessere Verfügbarkeit auf mehrere Orte verteilt und repliziert.
SAP	Größtes europäisches Softwareunternehmen
WGW	WohnGutWissen
Gewerblicher Mietvertrag	Der gewerbliche Mietvertrag muss von dem Mietvertrag für Wohnräume unterschieden werden, denn für die Wohnraummiete gelten gesetzlich festgelegte Mieterschutzvorschriften, die bei der gewerblichen Miete keine Anwendung finden.