Umbruch von Rafael & Henrik
Version 2 24.01.2024

WS 20/21 03.04.2021

Reussner

Softwaretechnik 2

Dauer: 90 min. Lösung: offiziell Bestanden ?/90 P.
Bemerkungen: -

Aufgabe 1: Wissensfragen *(10 Punkte)*

a) Nennen Sie drei der neun Charakteristiken von Microservice-Architekturen. (1,5 Punkte)

b) Nennen Sie die drei Auslieferungsmodelle für Dienste *(Service Delivery Models)* im Cloud Computing. Geben Sie außerdem jeweils ein konkretes Beispiel an. (3 Punkte)

c) Nennen Sie drei der sechs bewährten Praktiken *(best practices)* im Software-Entwicklungsprozesses des rationellen, unifizierten Prozesses *(Rational Unified Process)*. (1,5 Punkte)

d) Nennen Sie stichpunktartig zwei der vier Vorteile von agilen Methoden. (2 Punkte)

e) Nennen Sie zwei der vier unterschiedlichen Arten von Besprechungen *(meetings)* im SCRUM-Prozess und erläutern Sie jeweils kurz ihren Zweck. (2 Punkte)

Aufgabe 2: Anforderungserhebung *(16 Punkte)*

Im folgenden betrachten wir die Entwicklung eines Krankenhausverwaltungssystems *(Hospital Management System)*, abgekürzt **HMS**:

> *Ein Krankenhausverwaltungssystem ist ein komplexes Softwaresystem, welches alle patientenbezogenen Vorgänge im Krankenhaus – von der Patientaufnahme, Diagnostik und Behandlung bis zur Buchführung – unterstützt. Das System muss dabei berücksichtigen, dass ein Krankenhaus* (Hospital) *in unterschiedliche Stationen* (wards) *aufgeteilt ist, wobei jede eine Spezialisierung (z.B. Krebsbehandlung) und eigene Ausstattung (z.B. Anzahl Betten und Personal) besitzt.*

Während der Anforderungserhebung wurden für das Krankenhausverwaltungssystem unter anderem die folgenden Anforderungen identifiziert:

Nr.	Anforderung
/R1/	*Jeder Nutzer, der das HMS verwenden möchte, benötigt einen Nutzername und ein Passwort.*
/R2/	*Jedem Patient muss im HMS eine eindeutige Patientennummer zugewiesen werden.*
/R3/	*Beim Anlegen eines Patienten werden die folgenden notwendigen Daten im HMS angelegt: Patientennummer, Vor- und Nachname, Geburtsdatum und Krankenversicherungsnummer.*
/R4/	*Jede Änderung im HMS, wie Einfügen* (insert)*, Aktualisieren* (update) *und Löschen* (delete) *von Einträgen, wird schnell mit der Datenbank synchronisiert.*
/R5/	*Das HMS erfüllt alle Anforderungen der Datenschutz-Grundverordnung (DSGVO).*
/R6/	*Das HMS muss 1000 Nutzern ermöglichen, das System gleichzeitig zu nutzen.*
/R7/	*Die Benutzerschnittstelle des HMS muss leicht zu bedienen sein und innerhalb von 5 Sekunden auf jegliche Eingaben reagieren.*
/R8/	*Betten, die vom Stationspersonal als frei und benutzbar markiert werden, werden automatisch in die Liste der verfügbaren Betten aufgenommen.*

a) Nennen Sie vier weitere Stakeholder, die neben *Patienten* und *Ärzten* für die Entwicklung eines Krankenhausverwaltungssystems relevant sind. (2 Punkte)

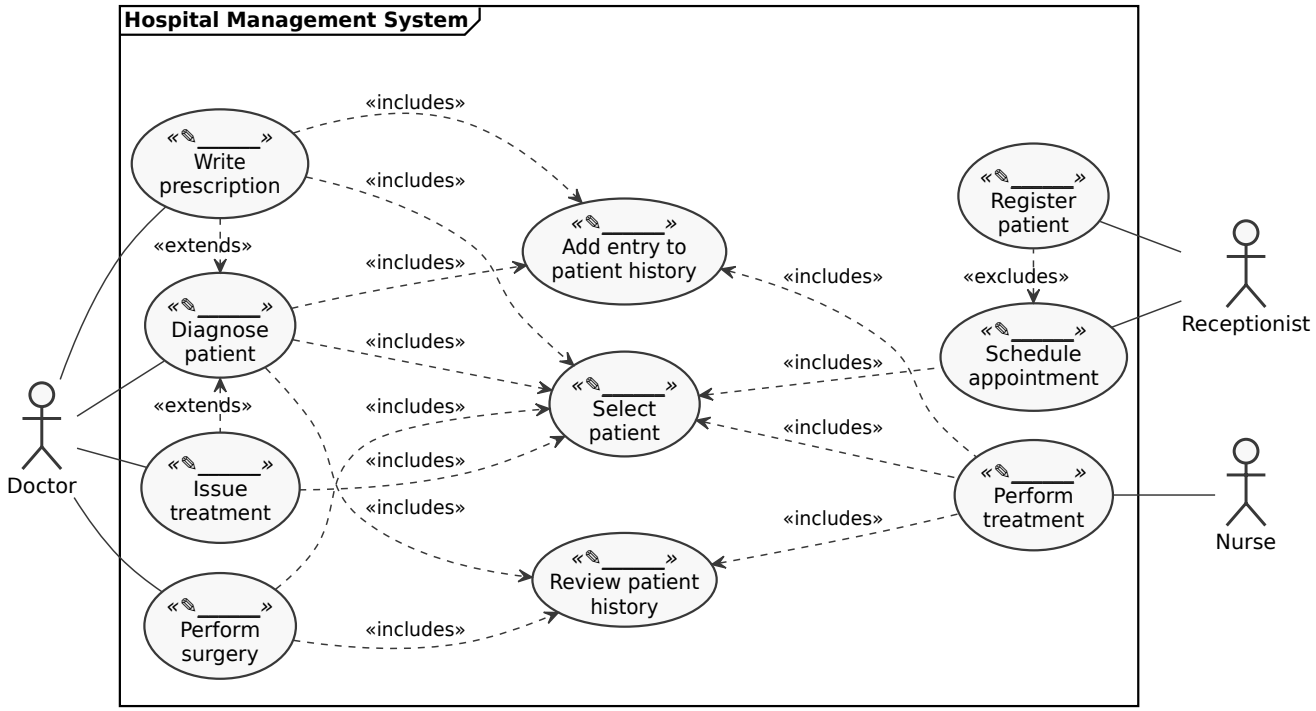
b) Klassifizieren Sie die oben stehenden Anforderungen nach der in der Vorlesung behandelten Klassifikation nach Glinz. Ordnen Sie hierzu die acht Anforderungen des Krankenhausverwaltungssystems den in der Vorlesung genannten Anforderungsarten *(kind: functional, quality, constraint)*, deren Unterkategorie *(subkind: functions, data, performance, security, availability, legal, …)* sowie deren Repräsentation *(representation: operational, quantitative, qualitative, declarative)* zu. (12 Punkte)

Nr.	Anforderung	Art	Unterkategorie	Repräsentation
/R1/	*Nutzername und Passwort*			
/R2/	*eindeutige Patientennummer*			
/R3/	*Anlegen von Patienten*			
/R4/	*Synchronisieren der Datenbank*			
/R5/	*DSGVO*			
/R6/	*1000 Nutzer*			
/R7/	*5s Reaktionszeit*			
/R8/	*verfügbare Betten*			

c) Welche der oben stehenden Anforderungen verstoßen gegen die Empfehlungen für gut geschriebene Anforderungen *(basic writing recommendations)* aus der Vorlesung? Finden Sie **zwei** Anforderungen die gegen **unterschiedliche** Richtlinien verstoßen. Nennen Sie jeweils die Nummer und Richtlinien, gegen die verstoßen wurde. (2 Punkte)

Aufgabe 3: Anwendungsfallbeschreibung *(21 Punkte)*

Diese Aufgabe betrachtet wesentliche Anwendungsfälle der Ärzte *(doctor)*, Arzthelfer *(nurse)* und Sprechstundenhilfen *(receptionist)* im Krankenhausverwaltungssystem. Diese sind im folgenden Anwendungsfalldiagramm zusammengefasst und in der nachfolgenden Tabelle kurz erläutert.



Anwendungsfall	Beschreibung
Write prescription	Ausstellen eines Rezeptes für den Patienten.
Add entry to patient history	Hinzufügen eines Vermerks in die Krankenakte.
Schedule appointment	Festlegen eines Termins für einen Patienten.
Diagnose patient	Untersuchen eines Patienten.
Select patient	Auswählen eines Patienten aus Datenbestand des HMS.
Register patient	Abfragen von Patientendaten und Aufnehmen eines neuen Patienten im HMS.
Issue treatment	Verschreiben einer ambulanten oder stationären Behandlung.
Perform treatment	Behandeln eines Patienten im Krankenhaus.
Perform surgery	Durchführen einer Operation im Krankenhaus.
Review patient history	Betrachten und Prüfen der Krankenakte eines Patienten.

a) Klassifizieren Sie jeden Anwendungsfall im obigen Anwendungsfalldiagram (use case diagram) entweder als Nutzerziel *(user goal)* mit U, als Teilfunktion *(subfunction)* mit S oder als Operation *(operation)* mit O. Tragen Sie hierzu die entsprechende Klassifikation in die mit „✎“ markierten Bereiche der Anwendungsfälle ein. (5 Punkte)

Da diese Anwendungsfälle nicht detailliert genug sind, ist es Ihre Aufgabe, weitere Anwendungsfalldiagramme zu erstellen. Hierzu charakterisiert die folgende Beschreibung die zugrundeliegenden Aufgaben der Krankenhausrezeption im Krankenhausverwaltungssystems *(HMS)*:

> *Das Teilsystem der Krankenhausrezeption* (hospital reception) *erlaubt es einer im HMS angemeldeten Sprechstundenhilfe* (receptionist)*, Patienten für die Arztsprechstunde anzumelden* (schedule doctor's consultation) *und Patienten zur stationären Behandlung aufzunehmen* (admit patient to hospital)*. In beiden Fällen muss die Sprechstundenhilfe zuerst den Patienten im HMS auswählen* (select patient) *und seine Krankenkassenkarte einlesen* (check insurance card)*. Falls nötig, kann die Sprechstundenhilfe vorab den Patienten im HMS neu registrieren* (register patient)*. Wird ein Patient zur stationären Behandlung eingewiesen* (admit patient to hospital)*, muss die Sprechstundenhilfe nach Auswahl des Patienten* (select patient) *und Einlesen der Krankenkassenkarte* (check insurance card)*, die Verfügbarkeit eines Betts auf der zugewiesenen Station prüfen* (check available beds in ward)*. Im Erfolgsfall kann die Sprechstundenhilfe dem Patienten ein freies Bett auf der zugewiesenen Station zuteilen* (allot bed for patient)*. Falls kein freies Bett in der zugewiesenen Station verfügbar ist, kann die Sprechstundenhilfe eine Verlegung des Patienten in ein anderes Krankenhaus veranlassen* (arrange transfer to other hospital)*.*

b) Erstellen Sie nun ein Anwendungsfalldiagramm (use case diagram) genau so, dass es die Anforderung im obigen Text beschreibt. Zeichnen Sie nur Anwendungsfälle und Aktoren ein die im Text referenziert werden. Achten Sie darauf, an den geeigneten Stellen Beziehungen zwischen den Anwendungsfällen (use cases) zu modellieren. (9 Punkte)

c) Verwenden Sie das textuelle Anwendungsfallbeschreibungsschema *Fully-Dressed* zur vollständigen Spezifikation des Erfolgsfalls der Patientenaufnahme *(admit patient to hospital)*. Nutzen Sie hierfür die unten gegebene Schablone. Nennen Sie drei Stakeholder und geben Sie mindestens eine Erweiterung an. (7 Punkte)

Name d. Anwendungsfalls
(Use Case Name)

Zielebene
(Goal Level)

Primäraktor(en)
(Primary Actor(s))

Stakeholder(s)

Vorbedingungen
(Preconditions)

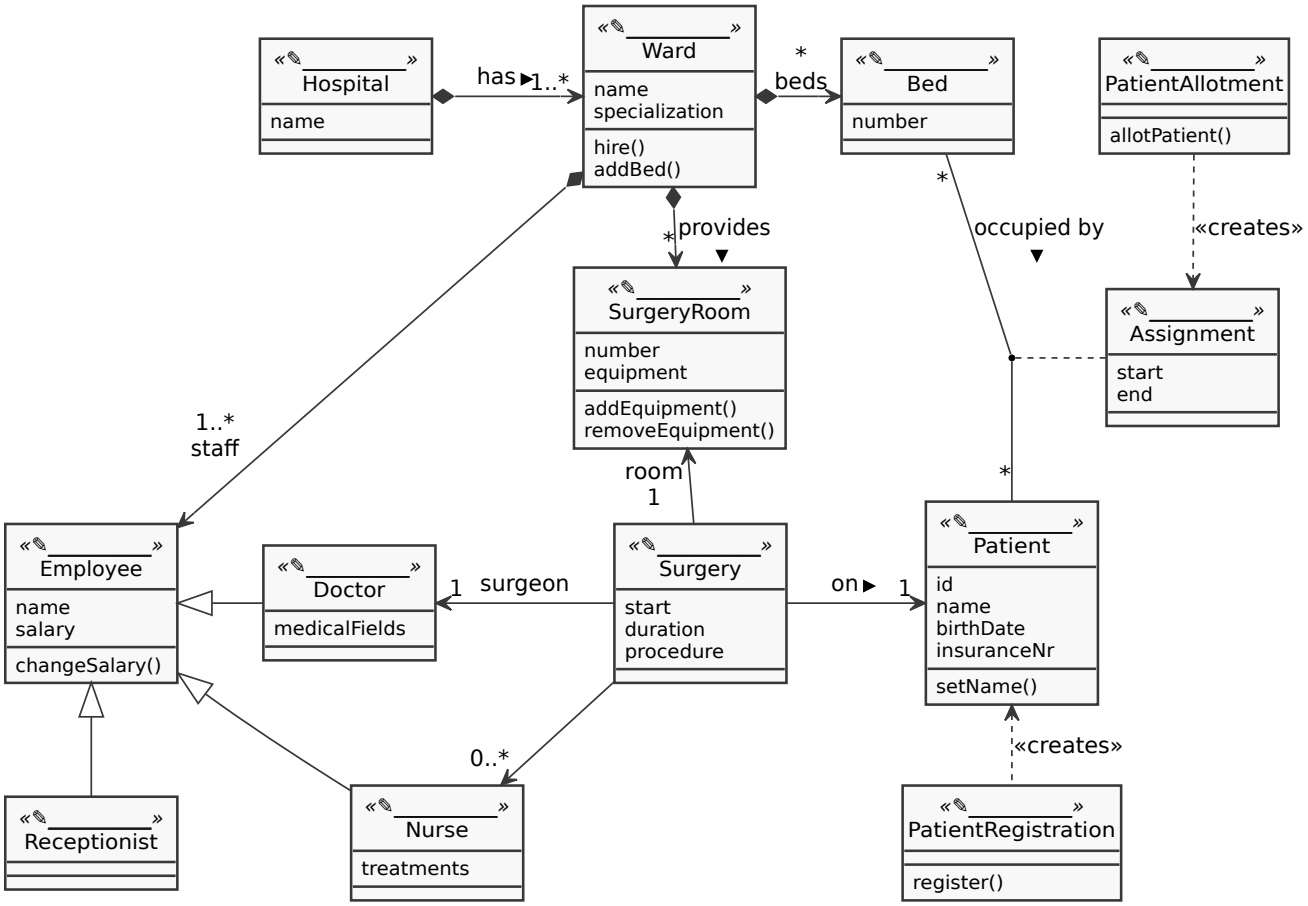
Nachbedingungen
(Postconditions)

Primäres Erfolgsszenario
(Main Success Scenario)
Erweiterung (Weiterer Ablauf)
(Extension; Additional Flow)

Aufgabe 4: Domänen-getriebener Entwurf *(13 Punkte)*

Der folgende Aufgabenkomplex widmet sich der Anwendung des Domänen-getriebenen Entwurfs der Stationsverwaltung *(ward management)* des Krankenhausverwaltungssystem. Diese erlaubt es, neue Patienten anzulegen *(register patients)*, Patienten auf freie Betten zuzuteilen *(patient allotment)* und Operationen *(surgeries)* in Operationssälen *(surgery rooms)* zu verwalten.

a) Durch die Domänenanalyse der Stationsverwaltung ist das folgende Klassendiagramm entstanden. Klassifizieren Sie jedes Domänenkonzept entsprechend der Bausteine (building blocks) des Domänen-getriebenen Entwurfs entweder als `ValueObject`, `Entity` oder `Service`. Tragen Sie hierzu die Klassifikation in die mit „✎“ markierten Bereiche der Domänenklassen ein. (6,5 Punkte)



b) Begründen Sie Ihre Entscheidung für die Klassifikation der Domänenklasse `Patient` in Stichpunkten. (1 Punkt)

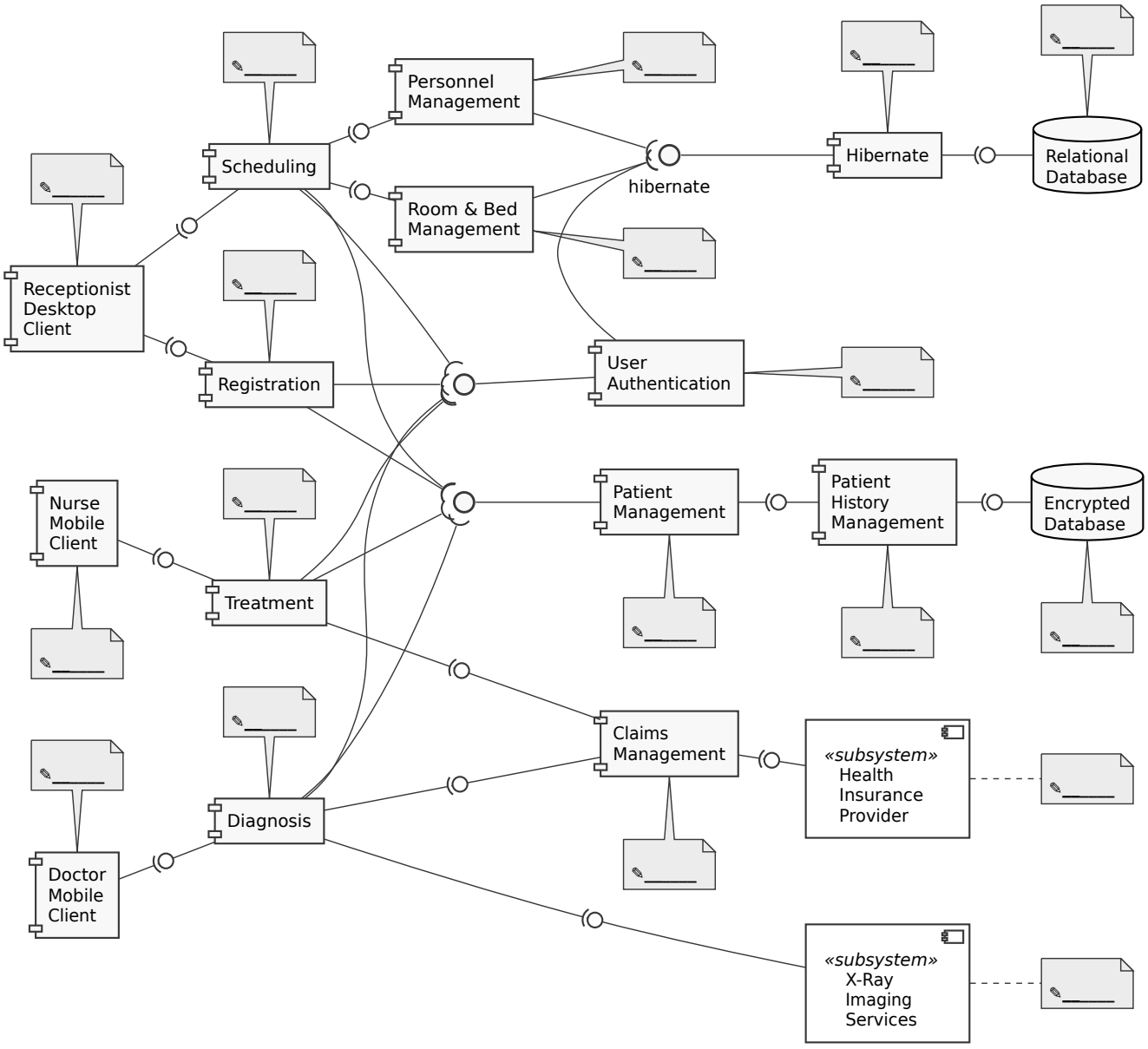
c) Entwerfen Sie nun die Unterstützungsmuster *Repository* und *Factory* für das Domänenkonzept `Surgery`. Zeichnen Sie hierfür ein UML-Klassendiagramm, welches ein *Repository*, eine *Factory* sowie ihre Methoden mit Parametern beinhaltet. Spezifizieren Sie für die *Factory* mindestens eine Methode und für das *Repository* mindestens drei Methoden. Zeichnen Sie zusätzlich die Domänenklasse `Surgery` sowie ihre Beziehungen zum *Repository* beziehungsweise zur *Factory* ein. (5,5 Punkte)

Hinweis: Datentypen (wie zum Beispiel `int`, `long`, `Date`, `String`, ...) müssen nicht angegeben werden. Beschriften Sie jede Assoziation mit mindestens einem Rollenname oder Assoziationsname sowie mindestens einer Kardinalität.

Aufgabe 5: Saubere Unternehmensarchitekturen *(16 Punkte)*

a) Tragen Sie in die Tabelle die vier Schichten der *sauberen* Architektur (clean architecture) so ein, dass sie zur angegebenen Richtung der Abhängigkeiten passen (dependency rule). (2 Punkte)

b) Das folgende Komponentendiagramm stellt die Softwarearchitektur eines Krankenhausverwaltungssystems dar. Ordnen Sie nun jede Komponente einer der Schichten *sauberer* Architekturen zu, indem Sie die Nummer der Schicht in die mit „✎“ markierten Bereiche eintragen. (9 Punkte)



Der Quelltext in Listing 1 stellt einen Ausschnitt der Implementierung des Bettenverwaltungssystems (`BedManager`) im Krankenhausverwaltungssystem dar. Genauer werden hier die Betten (`Bed`) einer Station und deren Zuteilung (`Assignment`) zu Patienten verwaltet. Die hierfür entwickelte Anwendung enthält jedoch einige Verstöße gegen Praktiken des sauberen Programmierens (Clean-Code-Prinzipien und Coding Conventions aus der Vorlesung).

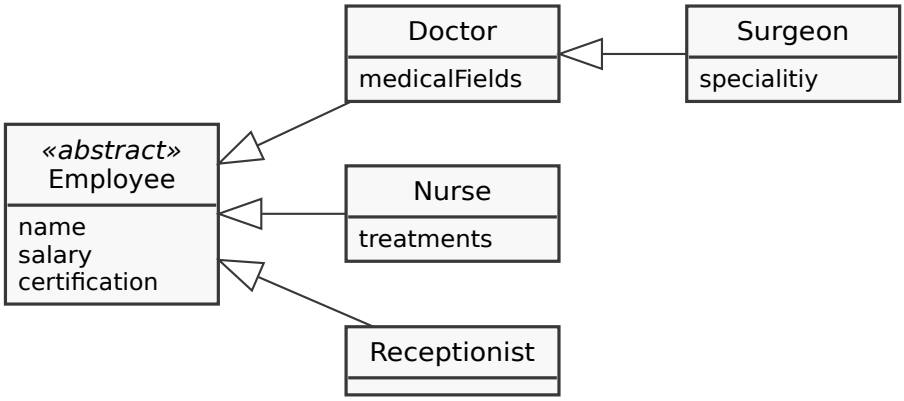
c) Identifizieren Sie **fünf** Zeilen im Quelltextbeispiel (nächste Seite), die **unterschiedliche** Arten von Verstößen gegen die Praktiken des sauberen Programmierens (Clean-Code-Prinzipien und Coding Conventions aus der Vorlesung) enthalten. Benennen Sie jeweils die **Art** des Verstoßes mit der zugehörigen **Zeilennummer**. (5 Punkte)

```java
public class Assignment{
   private final Patient patient;
   private final Bed bed;
   Date start, end;

   public Assignment(Patient patient, Bed bed, Date start, Date end) {
      if (start.after(end)) throw new IllegalArgumentException();
      this.patient = patient;
      this.bed = bed;
      this.start = start;
      this.end = end;
   }
   public Bed getBed() {return bed;}
   public Patient getPatient(){ return patient; }
}

public class BedManager {
   private Set<Bed> beds = new HashSet<>();
   private List<Assignment> assignments = new ArrayList<Assignment>();

   public boolean addBed(Bed bed) { return beds.add(bed); }
   public Collection<Bed> getBeds() { return beds; }
   public boolean hasNoOverlap(Assignment assign, Date start, Date end) {
      //if (start.after(end)) throw new IllegalArgumentException();
      return (assign.start.after(end) || assign.end.before(start));
   }
   public boolean isBedAvailable(Bed bed, Date start, Date end) {
      return assignments.stream()
         .filter(assign -> assign.getBed() == bed)
         .allMatch(assign -> hasNoOverlap(assign, start, end));
   }
   public Assignment createAssignment(Patient patient, Bed bed, Date start, Date end) {
      if (beds.contains(bed) && isBedAvailable(bed, start, end)) {
         Assignment assign = new Assignment(patient, bed, start, end);
         assignments.add(assign);
         return assign;
      }
      return null;
   }
   public HashSet<Bed> getAvailableBeds(Date start, Date end) {
      HashSet<Bed> available = new HashSet<>();
      //add each available bed to the set of available beds
      for (Bed bed : beds) if (isBedAvailable(bed, start, end)) available.add(bed);
      return available;
   }
  /*...*/
}
```

Listing 1: BedManagment-Klasse mit Bad Smells

Aufgabe 6: Entwurfsmuster für Unternehmenssoftware *(14 Punkte)*

Das Domänenmodell des Krankenhausverwaltungssystems enthält unter anderem die folgende Klassenhierarchie für die unterschiedlichen Angestellten im Krankenhaus – den Arzthelfern (`nurse`), den Sprechstundenhilfen (`receptionist`), den Ärzten (`doctor`) und den Chirurgen (`surgeon`). Um diese Klassenhierarchie in einer Datenbank abzubilden, sollen Sie in dieser Aufgabe die unterschiedlichen Objekt-Relationalen Strukturmuster (*object-relational structural patterns*), welche in der Vorlesung behandelt wurden, anwenden.



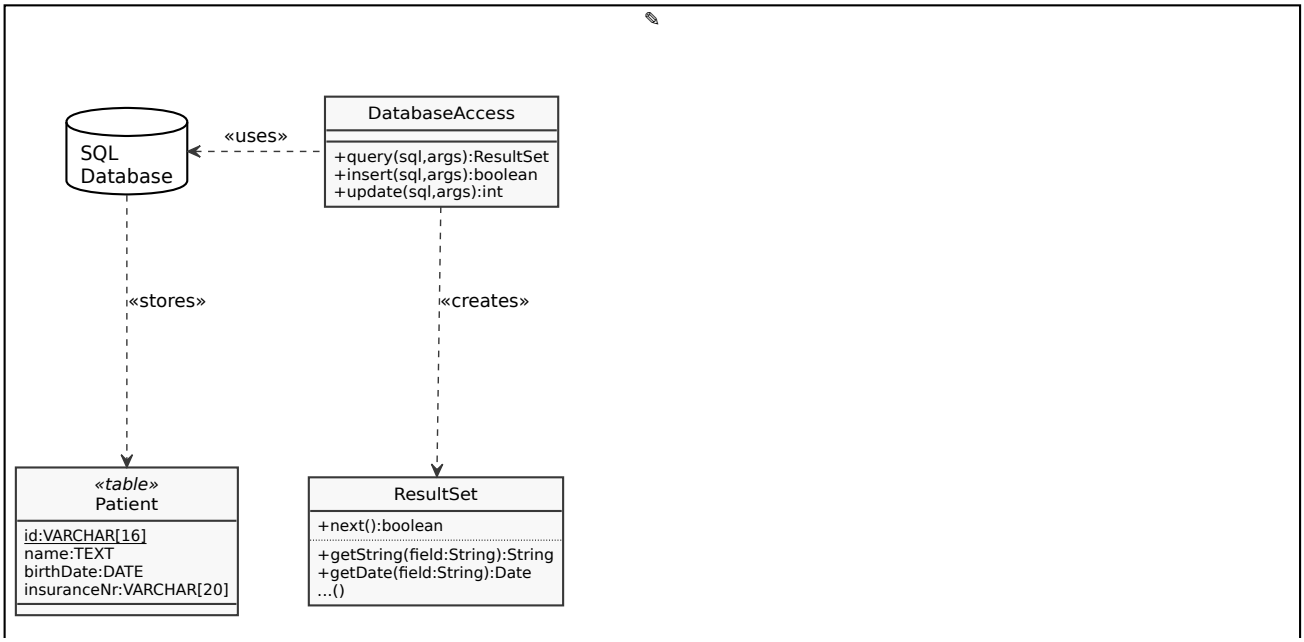
a) Erstellen Sie ein Datenbankschema nach dem Strukturmuster *Single Table Inheritance*. Erstellen Sie hierzu ein UML-Klassendiagramm der nötigen Tabellen (dargestellt als UML-Klasse mit dem `<<table>>` Stereotyp) und ihrer Attribute. Führen Sie, falls nötig, neue Attribute ein, um die *Primärschlüssel* und *Vererbung* darzustellen. (3 Punkte)

Hinweis: Datentypen (wie zum Beispiel `int`, `long`, `Date`, `String`, …) müssen nicht angegeben werden.

b) Erstellen Sie ein Datenbankschema nach dem Strukturmuster *Class Table Inheritance*. Erstellen Sie hierzu ein UML-Klassendiagramm der nötigen Tabellen (dargestellt als UML-Klasse mit dem `<<table>>` Stereotyp) und ihrer Attribute. Führen Sie, falls nötig, neue Attribute ein, um *Primärschlüssel* und *Vererbung* darzustellen. (7 Punkte)

Hinweis: Datentypen (wie zum Beispiel `int`, `long`, `Date`, `String`, …) müssen nicht angegeben werden.

Abschließend betrachten wir die Anbindung einer existierenden Patientendatenbank an das Krankenhausverwaltungssystem. Die nachfolgende Abbildung zeigt die Patiententabelle (`Patient`) mit dem Primärschlüssel `id` sowie den Feldern `name`, `birthDate` und `insuranceNr`. Um auf diese Tabelle zuzugreifen, sollen Sie die existierende Klasse `DatabaseAccess` verwenden.



c) Skizzieren Sie, wie die Patiententabelle (`Patient`) mit Hilfe des *Table Module* Strukturmusters zur Repräsentation von Domänenlogik *(domain logic pattern)* im Krankenhausverwaltungssystem abgebildet werden kann. Zeichnen Sie hierzu die notwendige Klasse, die notwendigen Methoden inklusive ihrer Parameter und Assoziation in das obige UML-Klassendiagramm ein. (4 Punkte)

Hinweis: Datentypen (wie zum Beispiel `int`, `long`, `Date`, `String`, ...) müssen nicht angegeben werden. Beschriften Sie jede Assoziation mit mindestens einem Rollenname oder Assoziationsname sowie mindestens einer Kardinalität.

Aufgabe 1: Wissensfragen *(10 Punkte)*

a)
- Componentization via services
- Organized around business capabilities
- Products not projects
- Smart endpoints and dumb pipes
- Decentralized governance
- Decentralized data management
- Infrastructure automation
- Design for failure
- Evolutionary design

b)
- **Software as a Service** (SaaS): Web Applications like Google Apps, Salesforce etc.
- **Platform as a Service** (PaaS): Development or execution environments like Google App Engine, Windows Azure, Amazon S3
- **Infrastructure as a Service** (IaaS): Offerings like Google Storage, Amazon Web Services etc.

c)
1. develop software iteratively
2. manage requirements
3. use component-based architectures
4. model software visually
5. verify software quality
6. control changes to software

d)
- Agile projects do not assume a stable world.
- Fast feedback on customer satisfaction and product quality essential.
- Disciplined process (certainly not ad hoc!)
- Shifts responsibility to each team member.

e)
- **Sprint Planning**:
  is used to develop a detailed plan for the iteration
- **Sprint Review**:
  demonstrate potentially shippable code, developed during the sprint
- **Daily Scrum**:
  share information, discover dependencies, adjust work plan, address individual needs and identified problems
- **Sprint Retrospective**:
  assesses the work in sprint, identifies good and bad practices

Aufgabe 2: Anforderungserhebung *(16 Punkte)*

a)
- Medizinisches Personal (z.B.: Arzthelfer, Pfleger)
- Administratives Personal (z.B.: Leiter, Verwaltungsangestellter, Buchhalter)
- Krankenkassen
- Gesetzgeber (Bundes-/Landesregierung)
- Techniker (z.B.: Wartungstechniker, Admin)
- (Software-Entwickler, Software-Tester)

b)

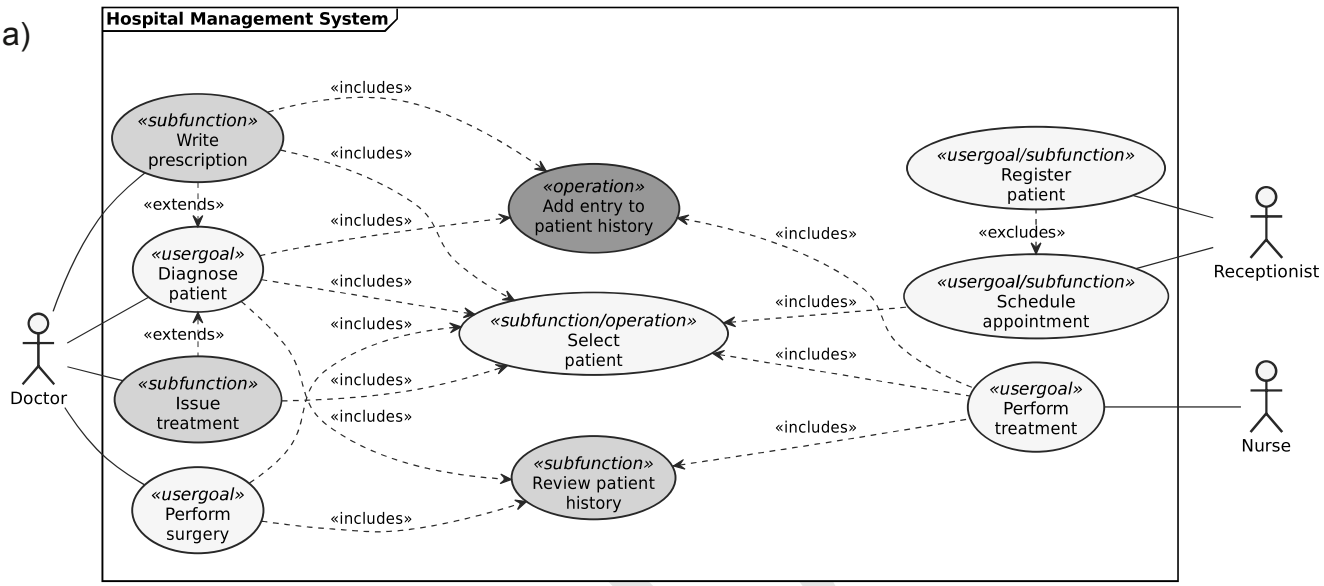
Nr.	Anforderung	Art	Unterkategorie	Repräsentation
/R1/	*Nutzername und Passwort*	functional (quality)	data (security)	declerative
/R2/	*eindeutige Patientennummer*	functional	functions (data)	operational
/R3/	*Anlegen von Patienten*	functional	functions (data)	operational
/R4/	*Synchronisieren der Datenbank*	quality	performance	qualitative
/R5/	*DSGVO*	constraint	legal	declarative
/R6/	*1000 Nutzer*	quality	availability	quantitative
/R7/	*5s Reaktionszeit*	quality	performance (usability)	quantitative (declarative)
/R8/	*verfügbare Betten*	functional	functions	operational

c)

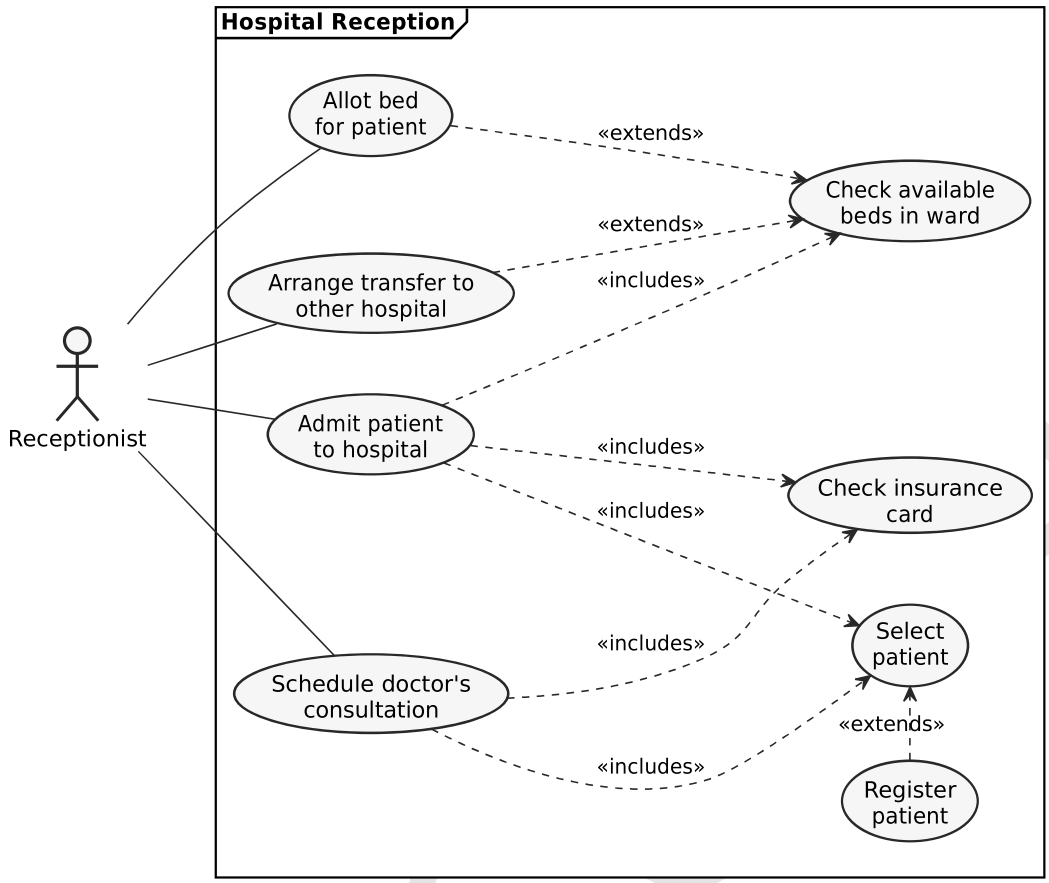
- /R4/: weak Language: *„schnell“*
- /R7/: Multple requirements per sentence
- /R7/: weak Language: *„leicht zu Bedienen“*
- /R8/: passive Language
- (/R3/: unclear actor: *„Beim Anlegen“* Wer legt an?)

Aufgabe 3: Anwendungsfallbeschreibung *(21 Punkte)*

a)



b)

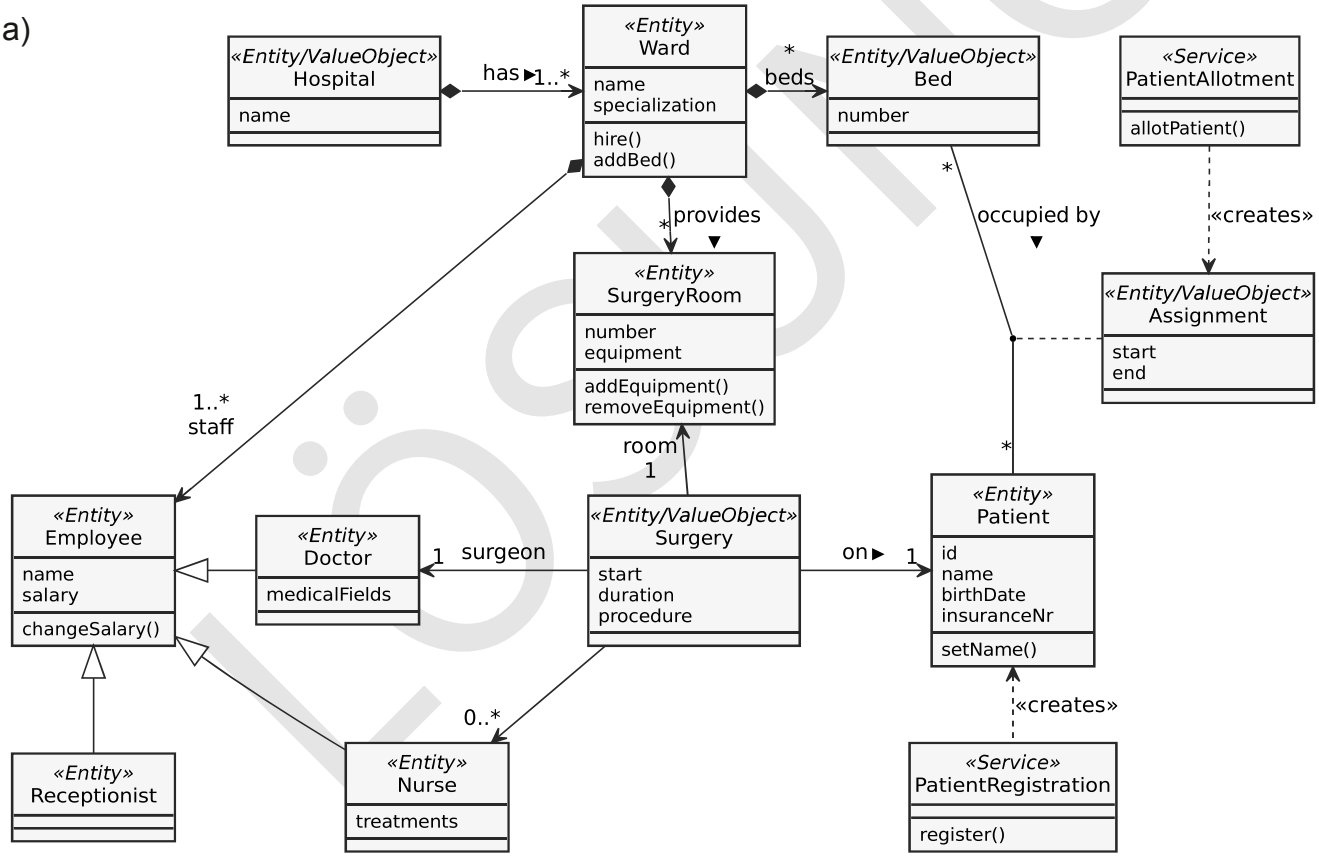


c)

Name d. Anwendungsfalls *(Use Case Name)*	**Patientenaufnahme** *(admit patient to hospital)*
Zielebene *(Goal Level)*	Nutzerziel (User Goal)
Primäraktor(en) *(Primary Actor(s))*	Sprechstundenhilfe, (Patient)
Stakeholder(s)	Sprechstundenhilfe, Patient, Doktor, Stationspersonal
Vorbedingungen *(Preconditions)*	• Die Sprechstundenhilfe ist im HMS angemeldet • (Ein Bett ist auf der zugewiesenen Station verfügbar.)
Nachbedingungen *(Postconditions)*	• Dem Patient wurde einem Bett zug (oder Verlegung ist veranlasst). • (Das Bett ist nicht mehr frei)
Primäres Erfolgsszenario *(Main Success Scenario)*	1) Patient im HMS auswählen 2) Krankenkassenkarte des Patienten einlesen 3) Verfügbarkeit eines Betts auf der zugewiesenen Station prüfen 4) Patienten das freie Bett zuteilen
Erweiterung (Weiterer Ablauf) *(Extension; Additional Flow)*	• 1a) Falls die Patient noch nicht im HMS existiert, registrieren des Patienten im HMS. • 3a) Falls kein Bett auf der zugewiesenen Station verfügbar ist, wird eine Verlegung des Patienten in ein anderes Krankenhaus veranlasst

Aufgabe 4: Domänen-getriebener Entwurf *(13 Punkte)*

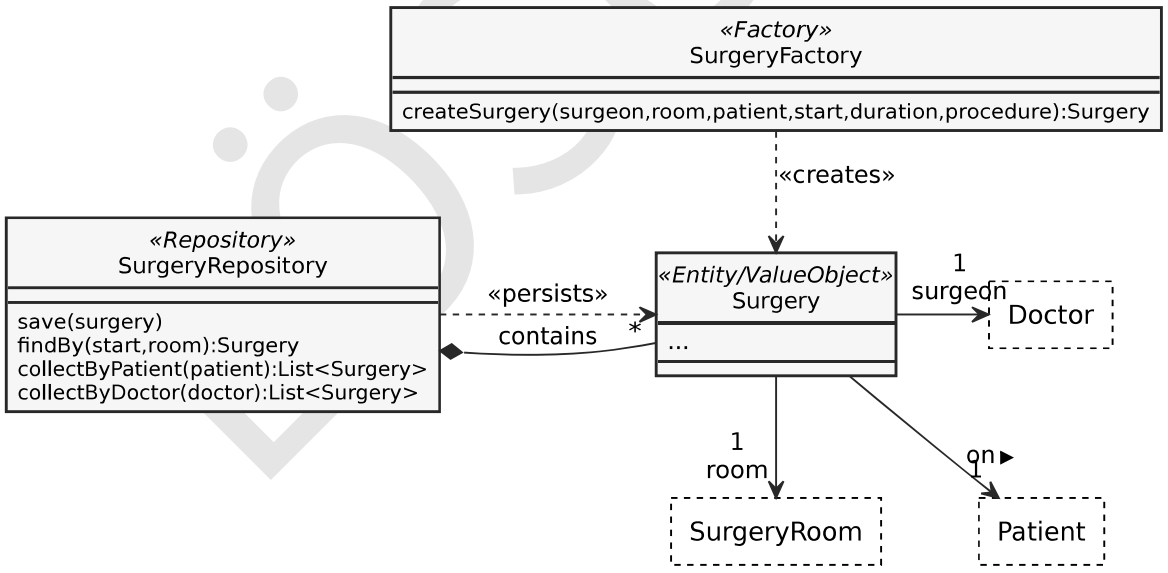
a)



b) Patient muss als `<<Entity>>` klassifiziert sein

- Patient hat eine eindeutige Identität
- Änderungen an `name` müssen über seine Lebenszeit nachverfolgbar sein

c)

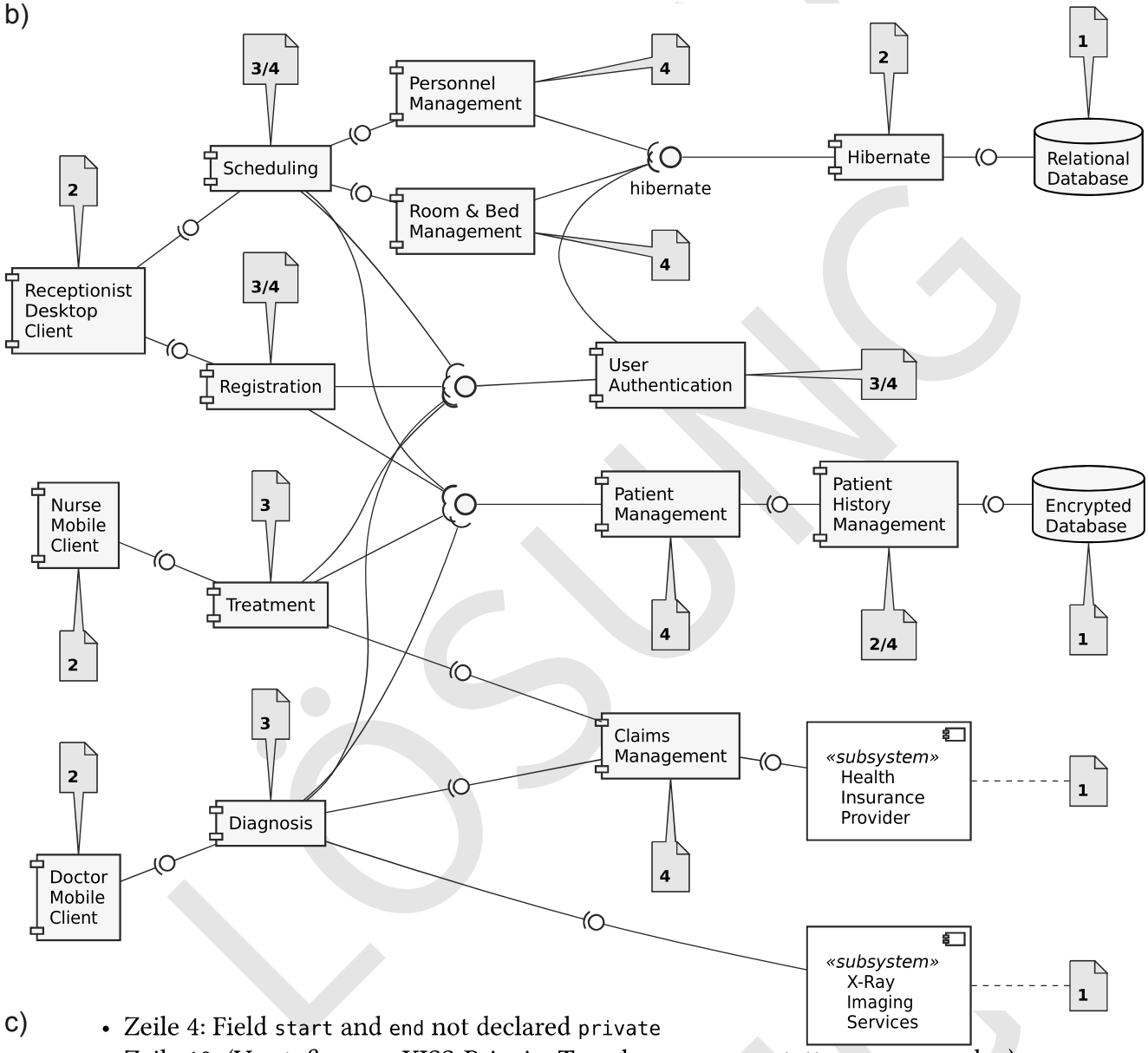


Aufgabe 5: Saubere Unternehmensarchitekturen

a)

Schicht	Abhängigkeitsregel	Schichtenname
1.	↓	Framework & Drivers (External Interfaces)
2.	↓	Interface Adapter (Controllers)
3.	↓	Application Business Rules (Use Cases)
4.	↓	Enterprise Business Rules (Entities)

b)



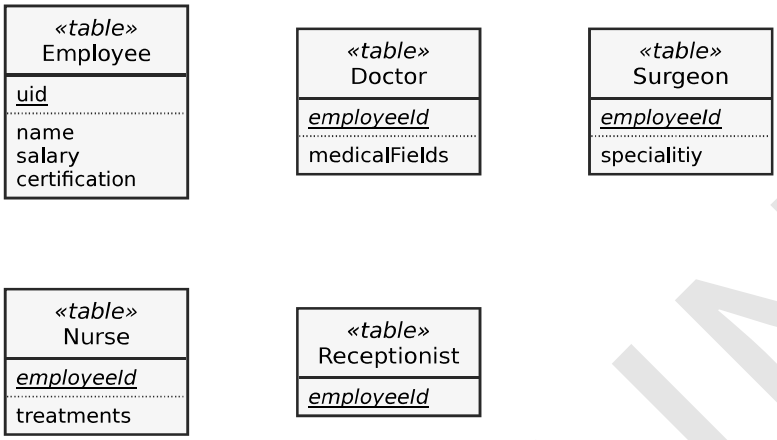
c)

- Zeile 4: Field `start` and `end` not declared `private`
- Zeile 19: (Verstoß gegen KISS-Prinzip, Typ als `Collection` statt `List` verwenden)
- Zeile 21: Verstoß gegen *Single Responsibility Principle*, `hasNoOverlap`-Methode sollte in `Assignment` implementiert werden
- Zeile 24: Auskommentierte Eingabevalidierung
- Zeile 25: Verstoß gegen *Law of Demeter*, direkter Zugriff auf `start` und `end`
- Zeile 32(35): Optional, *Principle Of Least Surprise*, `createAssignment`-Methode ist Zustandsverändernd
- Zeile 40: Verstoß gegen KISS-Prinzip, Exponieren der `HashSet` Implementierung nach außen
- Zeile 42: Nutzloser Kommentar
- Zeile 43: Formatierung, schwer zu lesen da Zeilenumbrüche fehlen

Aufgabe 6: Entwurfsmuster für Unternehmenssoftware

a)

b)



c)