Klausurprüfung (Fachtheorie) aus

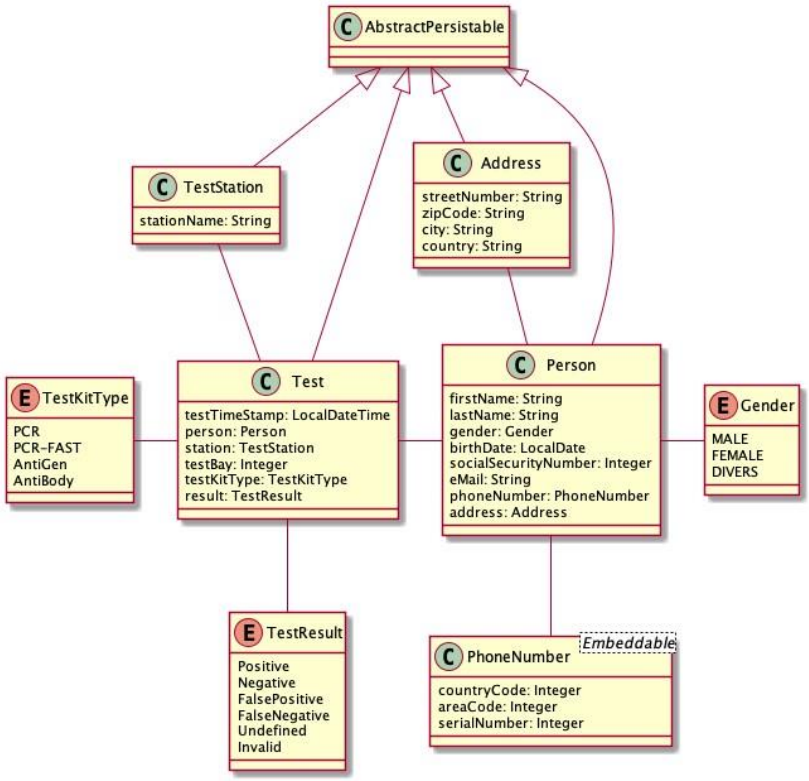
Programmieren und Softwareengineering

Allgemeine Information zur Abgabe

- erstellen Sie ein Protokoll mit all Ihren Konzepten, Annahmen, Designentscheidungen etc.
- benennen Sie dieses Dokument mit Ihrem Familiennamen und einem etwaigen Teilthema (falls Sie mehrere Protokolle abgeben wollen)
- schreiben Sie in jedes Protokoll Ihre PC User ID, damit wir Ihre Daten mit Ihrem Namen abgleichen können
- geben Sie dieses Dokument auf dem Abgabeverzeichnis ab
- drucken Sie zur Sicherheit alle Protokolle aus und legen Sie diese in Ihre Mappe

Fachlicher Überblick

Bei den **voneinander unabhängigen Teilthemen** geht es inhaltlich um den Aufbau einer simplifizierten COVID19 Test Datenbank Applikation mit folgendem Domänenmodell. Sie können für die Implementierung entweder den .NET Stack oder den Java / Spring Stack wählen.



1) Teilthema Domain Model, O/R Mapping und Persistence

a) Erstellen Sie ein Domänenmodell und mappen Sie dafür das oben abgebildete UML Diagramm technologisch richtig für eine relationale Datenbank. Beachten Sie dabei

die korrekte Umsetzung der Typen, Relationen und Stereotypen. Setzen Sie die Beziehung zwischen Person und Test bi-direktional um.

b) Sie finden ein **data.sql** file in Ihrem Setup, welches Sie zur initialen Befüllung Ihrer DB nutzen sollen.

c) Implementieren Sie ein **PersonRepository** welches folgende Abfragen erlaubt:
- Abfrage **einer Person** auf Basis seiner **Telefonnummer**
- Abfrage **von Personen** auf Basis der **Postleitzahl**
- Abfrage **von Personen** auf Basis eines **Ortsnamensteils**

d) Implementieren Sie ein **TestRepository** welches folgende Abfrage erlaubt: - Eine Statistikabfrage, die
- die **Anzahl**
- pro **TestKitType**
- pro **TestResult**
- für einen **Zeitraum**

e) Schreiben Sie automatische Tests, die die richtige Funktionsweise der RepositoryImplementierungen sicherstellen.

2) Teilthema Service Layer / Business Logic

a) Erstellen Sie eine Service Komponente, die folgende Logikanforderungen auf dem bestehenden Datenmodell implementiert. Achten Sie dabei wenn möglich auf die Verwendung moderner Programmierparadigmen wie **Streaming APIs**, **reactive** und/oder **funktionale Programmierung**.

b) Sie finden eine **Repository** Implementierung in Ihrem Setup, welches Ihnen die Datenbeschaffung bzw. -haltung in einer Mock-Version teilweise zur Verfügung stellt.

c) Ermitteln Sie eine **Statistik über die Test-Häufigkeit** (**absolut und in Prozent**) **je Geschlecht und Alterskategorie.**
- Alterskategorien sind: <10, <20, < 30, <40, <50, <60 und >60 Jahre

d) Schreiben Sie automatische Tests, die die richtige Funktionsweise der ServiceKomponenten Implementierungen sicherstellen.

3) Teilthema Presentationlayer / REST API

a) Erstellen Sie eine Web API und befolgen Sie dabei strikt die Implementierungsrichtlinien des REST Paradigmas.

b) Sie finden ein **Service** Implementierung in Ihrem Setup, welches Ihnen die Businesslogik und die Datenbeschaffung bzw. -haltung in einer Mock-Version teilweise zur Verfügung stellt.

c) Implementieren Sie einen **PersonController** welcher folgende Routen unterstützt:

- **`GET /api/persons/statistics?includeTestResults={yes/no}`** um demographische Statistiken über teilnehmende Personen und deren Testergebnisse nach Geburtsjahr zu ermitteln. Es soll folgende Response geliefert werden:

includeTestResults = no

```
[{ „yearOfBirth“: 1969,
   „male“: 47,
   „female“: 52
 },
 …
]
```

includeTestResults = yes

```
[{ „yearOfBirth“: 1969,
   „male“: {
     „positive“: 7,
     „negative“: 40
   },
   „female“:{
     „positive“: 4,
     „negative“: 47,
     „invalid“: 1
   }
 },
 …
]
```

- **`POST /api/persons/{id]/tests`** um für eine existierende Person einen Testtermin zu erzeugen. Verwenden Sie strukturell folgenden Payload mit exemplarischen Beispieldaten:

```
{
  „timeStamp“: „20210531T121500“,
  „station“: „Stadthalle“
}
```

- **`PUT /api/tests/{id}`** damit ein TestStations Mitarbeiter das Ergebnis eines Tests erfassen kann. Verwenden Sie strukturell folgenden Payload mit exemplarischen Beispieldaten:

```
{
 „testBay“: 11,
  „testKitType“: „PCR“,
  „result“: „Negative”
}
```

d) Schreiben Sie automatische Tests, die die richtige Funktionsweise der Implementierungen und Validierung der übergebenen Daten sicherstellen.

ACHTUNG!! Bitte die TODO’s in der Solution beachten.

Visual Studio:
Im Menü „View“ wird etwa in der Mitte die Option „Task List“ angeboten. Dort werden die TODO’s zusammengefasst.