LU09.L02 - Codereview: Planung bis Vorbereitung

Musterlösung zu LU09.A02 - Codereview: Planung bis Vorbereitung. Die Zeilennummern beziehen sich auf den Stand `b18ef94` von ghwalin/exam_api. Neuere Stände können abweichen.

Kick off: Verteilung der Kapitel

Bei 4 Personen erhält jede Person zwei Kapitel. So deckt jede Person eine andere Kombination ab und jedes Kapitel wird zweimal geprüft:

Gutachter	01 Bezeichner	02 Architektur	03 Darstellung	04 Kommentare
A	x	x		
B		x	x	
C			x	x
D	x			x

Bei 5 Personen erhält die fünfte Person z.B. `01` und `03`.

Musterbefunde

Die Liste ist eine Auswahl. Wiederkehrende Befunde (z.B. fehlende Docstrings an allen Gettern) sind einmal mit allen Fundstellen erfasst statt pro Zeile. Das ist auch in der eigenen Excel-Tabelle sinnvoll.

01 Bezeichner

Dateiname	Zeile	Bewertung	Befund
alle Dateien in `data/`, `model/`, `service/`	–	N	Die Modulnamen sind in PascalCase geschrieben (z.B. `ExamDAO.py`, `Person.py`). Module müssen kurze Namen in Kleinbuchstaben haben, z.B. `exam_dao.py`.
app.py	13	N	Klasse `PeoplelistService` liegt im Modul `PeopleListService.py`. Klassen- und Modulname sind uneinheitlich geschrieben.
model/Exam.py	72–81	B	Die Statustexte sind deutsch (`pendent`, `offen`). Sie werden angezeigt, das ist zulässig. Die Schlüssel `10`…`90` sind aber Magic Strings und sollten als Konstanten definiert werden.
service/ExamService.py	97	N	Variable `oldExam` ist in camelCase geschrieben. Richtig: `old_exam`.
data/EventDAO.py	38, 50, 89	B	Die Schleifenvariable `key` wird nicht verwendet. Besser `for event in self._eventdict.values()`.

Dateiname	Zeile	Bewertung	Befund
data/PersonDAO.py	51	B	Wie oben: unbenutzte Variable `key`.

02 Architektur

Dateiname	Zeile	Bewertung	Befund
model/Event.py	68–70	B	Die Prüfung `if __name__ == '__main__'` ist vorhanden, enthält aber nur `pass` und einen Docstring. Sie hat keine Funktion.
model/Exam.py	10	N	`Exam` ist gleichzeitig `@dataclass` und Unterklasse von `dict`. Die Klasse erbt damit alle Methoden eines Dictionary, ohne sie zu nutzen. Die Vererbung von `dict` entfernen. Gleich bei `Person` (model/Person.py, 6).
model/Exam.py	89–198	B	Die Properties überschreiben die Felder der Dataclass. Das funktioniert, ist aber schwer verständlich. Entweder Dataclass ohne Properties oder normale Klasse mit _-Attributen.
service/ (alle Services)	–	–	Die Services prüfen mit `__main__` nicht, ob sie direkt ausgeführt werden. Bei reinen Modulen ohne ausführbaren Code ist das vertretbar. Der Standard verlangt die Prüfung für jedes Skript. Bewertung: B

03 Darstellung und Formatierung

Dateiname	Zeile	Bewertung	Befund
app.py	16–18	N	Vor `def create_app()` steht nur eine Leerzeile. Funktionen werden durch zwei Leerzeilen getrennt.
app.py	77	N	Nach der Funktion `create_app` folgt `app = create_app()` nach nur einer Leerzeile.
app.py	78–80	B	Leerzeile direkt nach `if __name__ == '__main__':`.
data/EventDAO.py	83	B	Zwei Leerzeilen innerhalb der Klasse vor einer Methode. Richtig ist eine.
data/ExamDAO.py	142	N	Vor der Methode fehlt die Leerzeile.
model/Event.py	63	B	Leerzeichen am Zeilenende.
model/Event.py	68	N	Vor `if __name__ ...` steht nach der Klasse nur eine Leerzeile.
service/ExamlistService.py	59	N	Nach der Klasse steht nur eine Leerzeile.
model/Event.py	20–26	N	JSON wird mit + und \ aus Strings zusammengesetzt. Schwer lesbar und fehleranfällig (Anführungszeichen im Titel zerstören das JSON). `json.dumps` über ein Dictionary verwenden wie in `Exam.to_json`.

04 Kommentare und Dokumentation

Dateiname	Zeile	Bewertung	Befund
model/Event.py	8–12	N	Der Klassen-Docstring enthält keine Angaben zu Attributen und Methoden. Gleich bei `Exam` (model/Exam.py, 11) und `Person` (model/Person.py, 7).

Dateiname	Zeile	Bewertung	Befund
model/Event.py	20	N	Methode `to_json` hat keinen Docstring. Gleich bei `Person.to_json` (model/Person.py, 19).
model/Exam.py, Event.py, Person.py	alle Getter/Setter	–	Getter und Setter haben keine Docstrings. Der Standard nimmt Standard-Getter und -Setter aus. Kein Befund.
service/AuthenticationService.py	22–28	B	Der Docstring nennt «Parameters:», führt aber keine auf.
service/ExamService.py	87–92	N	Docstring von `save`: `args` ohne Beschreibung, `return` ohne Angabe des Rückgabewerts (`True`/`False`).
util/authorization.py	39	N	Funktion `teacher_required` hat keinen Docstring.
app.py	18	N	Funktion `create_app` hat keinen Docstring.
alle Module	1	B	Kein Modul-Docstring. Der BZZ-Standard verlangt ihn nicht, pylint meldet ihn aber.

Ausserhalb der Codingstandards

Beim Lesen fallen auch Fehler in der Logik auf. Sie gehören ins Protokoll, auch wenn kein Standard sie verlangt.

Dateiname	Zeile	Bewertung	Befund
data/EventDAO.py	39	H	`event.timestamp` ist ein String (model/Event.py, 15). `event.timestamp.date()` wirft deshalb einen `AttributeError`, sobald ein Filter übergeben wird.
data/EventDAO.py	51–55	H	`open_events` fängt `TypeError` ab und übergeht ihn. Fehlerhafte Daten verschwinden ohne Meldung aus der Liste.
data/ExamDAO.py	39–41	H	Jede Exception in `condition` wird mit `print` gemeldet, danach gibt die Methode `True` zurück. Eine fehlerhafte Prüfung zeigt die Prüfung also trotzdem an. Exception gezielt behandeln und loggen statt `print`.
service/AuthenticationService.py	59	N	Ein fehlgeschlagener Login liefert `404` («could not verify»). Richtig ist `401`.
service/AuthenticationService.py	63	N	`except Exception as ex`: zu allgemein, `ex` wird nicht verwendet.
util/authorization.py	31	N	`except Exception` fängt jeden Fehler beim Prüfen des Tokens, auch Programmfehler. Gezielt `jwt.InvalidTokenError` und `KeyError` abfangen.
util/authorization.py	42–43	N	Fehlende Berechtigung liefert `401`. Der Benutzer ist aber angemeldet, richtig ist `403`.

Dateiname	Zeile	Bewertung	Befund
data/EventDAO.py	2	B	`strftime` und `strptime` werden importiert, aber nicht verwendet.

Werkzeuge

Viele Befunde aus Kapitel 01, 03 und 04 findet pylint bzw. flake8 automatisch (siehe LU09.L01):

```
pylint --max-line-length=120 app.py data model service util
flake8 --max-line-length=120 app.py data model service util
```

Die Befunde in Kapitel 02 und die Logikfehler findet kein Linter. Dafür braucht es das Review.

M450-LU09, M450-D1I

Marcel Suter

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/m450/learningunits/lu09/loesungen/review_vorbereitung

Last update: **2026/11/17 18:05**