INTERNATIONALE HOCHSCHULE

PORTFOLIO

Aufgaben zur Vorlesung: Objektorientierte und funktionale Programmierung mit Python (DLBDSOOFPP01)

INHALTSVERZEICHNIS

1. THEMEN UND AUFGABEN 2

1.1. Aufgabe 1: Erstellen Sie eine Gewohnheits-Tracking-App 2

1.1.1. Konzeptionsphase 4

1.1.2. Entwicklungsphase/Reflexionsphase 4

1.1.3. Finalisierungsphase 5

2. TUTORIAL-UNTERSTÜTZUNG 6

3. BEWERTUNG 6

4. FORMELLE RICHTLINIEN UND SPEZIFIKATIONEN FÜR DIE EINREICHUNG 7

4.1. Bestandteile der Prüfungsleistung 7

4.2. Format für die Übermittlung digitaler Dateien 8

4.3. Form des Abstracts..... 10

1. THEMEN UND AUFGABEN

Im Rahmen dieser Lehrveranstaltung ist folgendes Thema zu wählen.

1.1. Aufgabe 1: Erstellen Sie eine Gewohnheits-Tracking-App

Gute Gewohnheiten zu entwickeln und schlechte zu brechen, ist keine leichte Aufgabe. Um bestimmte Gewohnheiten im Auge zu behalten oder persönliche Ziele zu erreichen, verlassen sich immer mehr Menschen auf sogenannte Habit Tracker, die ihnen durch den Tag helfen. Wenn Sie sich einen beliebten App-Store ansehen, werden Sie keinen Mangel an Gewohnheits-Tracking-Anwendungen in einer großen Auswahl an Qualität und Preisen finden.

Kürzlich hat sich ein Kunde an Sie gewandt und möchte, dass Sie ihm helfen, ein grundlegendes Python-Backend für eine Gewohnheits-Tracking-App zu erstellen, die er später in diesem Jahr einführen möchte. Um dieses Projekt realisierbar zu machen, müssen Sie sich auf die wesentlichen Funktionen einer solchen Anwendung konzentrieren. Sie werden nicht gebeten, irgendeine Art von grafischer Benutzeroberfläche bereitzustellen, sondern nur die Grundfunktionen eines Gewohnheitstrackers, der objektorientierte und funktionale Programmierung in Python gemäß den folgenden Spezifikationen verwendet.

Formal ist eine Gewohnheit eine *klar definierte Aufgabe,* die *periodisch erledigt werden* muss (z. B. täglich die Zähne putzen oder einmal im Jahr zum Zahnarzt gehen). Die Grundbausteine einer Tracking-App sind wie folgt:

- o Ein Benutzer kann mehrere Gewohnheiten in der Anwendung definieren. Eine Gewohnheit hat eine Aufgabenspezifikation und eine Periodizität. o Eine *Aufgabe kann* von einem Benutzer zu jedem Zeitpunkt erledigt, dh „abgehakt“ werden. o Jede Aufgabe muss *mindestens einmal* während des vom Benutzer für die jeweilige Gewohnheit festgelegten Zeitraums abgehakt werden . Wenn ein Benutzer es versäumt, eine Gewohnheit während des angegebenen Zeitraums zu vervollständigen, wird gesagt, dass der Benutzer *die Gewohnheit bricht.* o Wenn es einem Benutzer gelingt, die Aufgabe einer Gewohnheit x aufeinanderfolgende Perioden hintereinander zu erledigen, dh ohne die Gewohnheit zu brechen, sagen wir, dass der Benutzer eine Serie *von x Perioden aufgebaut hat.* Wenn ein Benutzer beispielsweise jeden Tag trainieren möchte und dies zwei volle Wochen lang tut, legt er eine 14-tägige Trainingsserie fest. o Die Gewohnheiten, die Nutzer in der App eingeben, werden nicht nur gespeichert, sondern können auch analysiert werden. Benutzer möchten Antworten auf mehrere Fragen wie: Was ist meine längste Gewohnheitsserie? Was ist die Liste meiner aktuellen täglichen Gewohnheiten? Mit welchen Gewohnheiten hatte ich letzten Monat am meisten zu kämpfen?

Akzeptanzkriterien für Ihren Habit Tracker

Habit Tracker können komplizierte Software sein. Der Umfang dieses Projekts ist aus Zeitgründen begrenzt.
Die folgenden Aufnahmekriterien sollen Ihnen helfen, sich ein klareres Bild davon zu machen, was von Ihnen für dieses Projekt erwartet wird.

- o Ihre Lösung sollte mit *Python Version 3.7 oder höher erstellt werden,* damit Sie sich auf das Schreiben von modernem Python-Code konzentrieren können. Die Tools und Bibliotheken, die zur Implementierung Ihrer Habit-Tracker-Anwendung verwendet werden, sind vollständig Ihre eigene Wahl. Es ist Ihnen jedoch nicht gestattet, bestehende Gewohnheits-Tracking-Tools von Drittanbietern, die Sie möglicherweise im Internet oder anderswo finden, direkt zu verwenden oder zu modifizieren.
- o Die von Ihnen eingereichte Software muss mit detaillierten, eigenständigen *Installations- und Ausführungsanweisungen geliefert* werden, damit klar ist, wie Sie das Projekt installieren und Ihren Habit Tracker verwenden. Eine gut geschriebene README.md oder ähnliches reicht aus, vorausgesetzt, Sie geben alle Informationen ordnungsgemäß an. Stellen Sie sicher, dass Ihr Code auch mit Python-Docstrings dokumentiert ist.
- o Das Konzept einer Gewohnheit sollte mittels objektorientierter Programmierung als Klasse kodiert werden. Abhängig von Ihrem Design benötigen Sie möglicherweise weitere Klassen für dieses Projekt.
- o Ihr Tracker sollte es Benutzern ermöglichen, mindestens zwei Gewohnheitsperioden zu erstellen, nämlich *wöchentliche und tägliche* Gewohnheiten.

- Ihre Lösung enthält *5 vordefinierte Gewohnheiten* (mindestens eine wöchentliche und eine tägliche Gewohnheit). Es sollte klar dokumentiert werden, wie mit Ihrer Lösung neue Gewohnheiten geschaffen werden können. Machen Sie auch transparent, wie ein Benutzer eine Aufgabe innerhalb eines bestimmten Zeitraums erledigen kann.
- Für jede Gewohnheit verfolgt Ihr System, wann sie erstellt wurde, und das Datum und die Uhrzeit der Gewohnheitsaufgaben abgeschlossen worden.
- Für jede vordefinierte Gewohnheit sollten Sie Beispiel-Tracking-Daten für einen Zeitraum von 4 Wochen bereitstellen. Diese Beispieldaten werden später zu Testzwecken als sogenanntes „Test Fixture“ verwendet. Es ist auch hilfreich, Ihren Ansatz beim Programmieren Ihrer Lösung zu validieren, da Sie Gewohnheiten leichter laden können, wenn Sie auf vordefinierte Daten zurückgreifen.
- Sie brauchen eine Möglichkeit, Gewohnheitsdaten zwischen Benutzersitzungen zu speichern oder zu speichern. Das Speichern und Laden von Daten kann entweder mit einer einfachen dateibasierten Lösung erfolgen, z. B. durch Lesen und Schreiben von JSON-Dateien mit Pythons eingebautem *json* - Modul, oder mit einer relationalen Datenbanklösung mit Tools wie *sqlite3.* Beide Ansätze können gut funktionieren, aber mit einer DB zu arbeiten ist wahrscheinlich professioneller und kann Ihnen mehr für die Zukunft beibringen.
- Ihre Lösung verfügt über ein *Analysemodul ,* mit dem Benutzer ihre Gewohnheiten analysieren können. Die Funktionalität dieses Analysemoduls muss unter Verwendung des *Paradigmas der funktionalen Programmierung implementiert werden.* Es steht Ihnen frei, auch andere Funktionen zu implementieren, aber dies sind die Mindestanforderungen. Bereitstellung von Funktionen zum - Zurückgeben einer Liste aller derzeit verfolgten Gewohnheiten, - Zurückgeben einer Liste aller Gewohnheiten mit der gleichen Periodizität, - Zurückgeben der längsten Laufserie aller definierten Gewohnheiten, - Zurückgeben der längsten Laufserie für eine gegebene Gewohnheit.
- Ihre Lösung muss eine saubere API haben, die die Benutzer verstehen. Sie tun dies, indem Sie dem Benutzer ein Befehlszeilenschnittstellentool (CLI) zur Verfügung stellen, mit dem er seine Gewohnheiten erstellen, löschen und analysieren kann. Es gibt großartige Tools wie *Fire* oder *Click ,* um CLIs zu erstellen, aber Sie können auch eine einfache Hauptschleife für Ihr Programm verwenden, z. B. mit dem integrierten *Eingabebefehl ,* um ein interaktives Menü zu erstellen. Wenn Sie Erfahrung im Erstellen grafischer Benutzeroberflächen haben, können Sie alternativ auch eine GUI erstellen. Eine Idee könnte sein, *tkinter* oder ähnliche Python-basierte Tools zu verwenden oder einen moderneren Webentwicklungsansatz mit *Flask* oder Django zu *wählen.* Beachten Sie nur, dass Sie mit einer optisch ansprechenderen Lösung keine zusätzlichen Punkte erhalten. Am Ende wird die Einreichung nach den Akzeptanzkriterien und der Qualität Ihres Python-Codes bewertet.
- Die kritischen Teile Ihrer Lösung, insbesondere die Gültigkeit Ihrer Gewohnheitsverfolgungskomponenten und des Analysemoduls, sollten getestet werden, indem Sie eine *Einheitstestsuite* bereitstellen , die gemäß den mit der Lösung gelieferten Anweisungen ausgeführt werden kann. Es wird empfohlen, dafür entweder mit *pytest* oder *unittest* zu arbeiten.

Ihre Anwendung muss gemäß den folgenden drei Phasen erstellt, dokumentiert und bereitgestellt werden.

1.1.1. Konzeptionsphase

Diese Phase stellt den wichtigsten Teil des Designprozesses dar. Was in dieser Phase übersehen oder vergessen wird, wirkt sich später negativ auf die Umsetzung aus und führt im schlimmsten Fall zu unbrauchbaren Ergebnissen.

Der erste Schritt besteht darin, ein **schriftliches Konzept zu erstellen,** um alles zu beschreiben, was Sie zum Erstellen Ihrer Gewohnheits-Tracking-Anwendung benötigen. Um es zu schreiben, stellen Sie sich vor, Sie arbeiten in einem Unternehmen, das eine Gewohnheits-Tracking-App erstellt, Sie sind der Hauptentwickler und Sie versuchen, Ihrem Kollegen, der neben Ihnen sitzt, die technischen Grundlagen zu erklären. Sie müssen also nicht zu sehr ins Detail gehen, was Gewohnheiten usw. sind, Sie können direkt in die vorgeschlagene Lösung eintauchen. Benötigen Sie neben Ihrem „Habit"-Kurs noch etwas? Wie werden Gewohnheitsdaten gespeichert und abgerufen? Wie interagieren Benutzer mit der Anwendung und wie ist der allgemeine Benutzerfluss Ihrer App? Zur Visualisierung Ihres Konzepts wird mindestens **ein Diagramm** (z. B. in UML) erstellt und in das geschriebene Konzept eingefügt, um das Zusammenspiel der Komponenten und des Prozesses darzustellen.

Dieser Schritt, zuerst ein Konzept zu erstellen, ist vielleicht der wichtigste des gesamten Designprozesses. **Es ist entscheidend, sich für diese Phase genügend Zeit zu nehmen/planen, bevor die nächsten Schritte unternommen werden können.** Beachten Sie daher unbedingt die Reihenfolge der jeweiligen Schritte.

Es ist wichtig, nicht nur zu überlegen, was der Kunde wahrscheinlich von Ihrer Gewohnheits-Tracking-App erwartet, sondern auch, wie Benutzer damit interagieren sollten, wie sie neue Gewohnheiten erstellen, Aufgaben erledigen und ihren Fortschritt überprüfen. Aus dem schriftlichen Konzept muss hervorgehen, warum die Struktur und der Prozess so gestaltet wurden. Überlegen Sie, mit welchen Tools Sie die einzelnen Komponenten implementieren können und wie die Kommunikation zwischen den Komponenten funktioniert.

Für die Einreichung ist **ein konzeptioneller Text (1-3 DIN A4-Seiten) als PDF** zu erstellen, der diese Analysen und Überlegungen erläutert, zusammen mit dem **Diagramm/** den Diagrammen , die das Zusammenspiel der Komponenten und des Prozesses zeigen.
Das PDF wird in die PebblePad-Vorlage hochgeladen, das Textfeld in der Vorlage kann leer gelassen werden.

Während des gesamten Prozesses werden Online-Tutorials angeboten, die Gelegenheit bieten, sich auszutauschen, Ideen und/oder Entwürfe auszutauschen und Feedback einzuholen. In den Online-Tutorien können beispielhafte Arbeiten mit dem Tutor besprochen werden. Hier hat jeder die Möglichkeit, sich einzubringen und von den Rückmeldungen der anderen zu lernen. **Es wird empfohlen, diese Kanäle zu nutzen, um Fehler zu vermeiden und Verbesserungen vorzunehmen.** Arbeiten sollten Sie erst nach Nutzung der oben genannten Tutorial- und Informationsmedien einreichen. Anschließend folgt ein Feedback des Tutors und die Arbeit an der zweiten Phase kann beginnen.

1.1.2. Entwicklungsphase/Reflexionsphase

In dieser Phase **beginnen Sie mit der Implementierung Ihrer Gewohnheits-**

Tracking-App: o Sie richten die Frameworks und Tools ein, die Sie in der Konzeptionsphase beschrieben haben. o Sie implementieren die in Ihrem Entwurfsdiagramm skizzierten Komponenten. o Ihr Code wird kommentiert und Sie dokumentieren seine Verwendung. o Benutzer können Gewohnheiten, die sie definieren, auf bequeme Weise erstellen, verwalten und überprüfen. o Benutzer können ihre Gewohnheiten analysieren.

In dieser Phase müssen Sie eine **Erläuterung Ihres Entwurfs- und Implementierungsverfahrens** als zusammengesetzte **Präsentation im PDF-Format mit etwa 5-10 Folien einreichen.** Die Datei sollte **visuelle Elemente** enthalten , die das Verständnis erleichtern, sie muss strukturiert sein und **erläutern, für welche Tools Sie sich entschieden haben.** Während Sie in der ersten Phase ein internes Designdokument ausgearbeitet haben, das Sie mit Ihren Kollegen teilen, stellen Sie sich diese Präsentation als ein kundenorientiertes Dokument vor. Sie als Core-Entwickler erklären Ihrem potenziellen Kundenstamm, welche Komponenten und Features Ihre Anwendung hat und wie Sie diese nutzen.
Das PDF wird in die PebblePad-Vorlage hochgeladen, das Textfeld in der Vorlage kann leer gelassen werden.

Während des gesamten Prozesses bieten Online-Tutorials und andere Kanäle die Möglichkeit, Ideen und/oder Entwürfe ausführlich zu diskutieren und ausreichend Feedback, Tipps und Hinweise zu erhalten. **Es wird empfohlen, diese Kanäle zu nutzen, um Fehler zu vermeiden und Ihre Arbeit zu verbessern.** Ist dies geschehen, können Sie Ihre zweite Phase zur Evaluation abgeben. Nach einem Feedback des Tutors wird Ihre Arbeit am finalen Entwurf in der dritten Phase fortgesetzt.

1.1.3. Finalisierungsphase

In dieser letzten Phase ist es Ihr Ziel , **die Bewerbung nach Erhalt des Feedbacks des Tutors zu verfeinern** und für die endgültige Einreichung vorzubereiten. Bestimmte Elemente müssen möglicherweise verbessert oder geändert werden, um die Aufgabe abzuschließen und diesen Portfolio-Kurs abzuschließen.

Ihr fertiges Produkt, die Habit-Tracking-App, wird eingereicht, indem Sie alle Dateien bereitstellen: **die von Ihnen erstellten Dateien,** Ihren **Python-Programmcode, Dokumentation** usw. wie folgt:

- Ihr Projekt wird in einem öffentlichen GitHub-Repository gehostet. Das bedeutet, dass Sie ein GitHub-Konto erstellen müssen, falls Sie noch keines haben, ein Repository erstellen und Ihren gesamten Code in dieses Repository hochladen. Bei der Übermittlung in PebblePad geben Sie einen **Link zum Repository an.** Das Hosten des Projekts auf GitHub ist Teil des Aufbaus Ihres Portfolios. Eine gut geschriebene README.md ist eine ausreichende Dokumentation für die endgültige Einreichung, aber der Code selbst muss ebenfalls dokumentiert werden.
- Anschließend erstellen Sie aus allen im GitHub-Repository enthaltenen Dateien eine ZIP-Datei und legen diese in einem Ordner ab. Sie müssen diesen Ordner zippen und in Ihre Einreichung in PebblePad einfügen.

Wichtiger Hinweis: Der Inhalt des GitHub-Repositorys und der Inhalt der in Ihren Zip-Ordner hochgeladenen ZIP-Datei sollten *identisch sein.* Sie dürfen beide nach der endgültigen Einreichung nicht mehr ändern. Der Inhalt des Zip-Ordners wird nach Abgabe an das Prüfungsamt gesendet.

Zusätzlich stellen Sie ein **1–2-seitiges Abstract-PDF-Dokument** zur Verfügung , in dem Sie Ihre Lösung inhaltlich und konzeptionell beschreiben. Dieses Abstract präsentiert eine kurze Zusammenfassung („Making of“ des Projekts) über die technische Herangehensweise auf anschauliche und informative Weise. Betrachten Sie diese Zusammenfassung als einen kurzen Bericht an Ihren Vorgesetzten am Ende dieses Projekts. Was lief gut und was nicht? Welche Fallstricke sind Ihnen aufgefallen, die Sie nicht vorhergesehen haben? Welche Funktionen haben Sie in die Anwendung integriert, auf die Sie am meisten stolz sind und die dem Gesamtprodukt einen Mehrwert verleihen? Bitte stellen Sie sicher, dass das Abstract einen Link zu Ihrem Projekt auf GitHub enthält.

Zusammenfassend werden Sie in dieser Phase Ihr Projekt einreichen, indem Sie es auf GitHub hosten und als ZIP-Datei in einen ZIP-Ordner hochladen. In PebblePad geben Sie den Link zum GitHub im Textfeld für die Einreichung an, laden den Zip-Ordner in das entsprechende Feld hoch und reichen die Abstract-PDF-Datei als Endprodukt ein. Das Abstract sollte den Link zu Ihrem GitHub-Projekt enthalten.

In der „Finalisierungsphase“ bieten die Online-Tutorials und andere Kanäle zudem die Möglichkeit, ausreichend Feedback, Tipps und Hinweise zu erhalten, bevor das fertige Produkt schließlich abgegeben wird. **Es wird empfohlen, diese Kanäle zu nutzen, um Fehler zu vermeiden und Verbesserungen vorzunehmen .** Das fertige Produkt wird **mit den Ergebnissen aus Phase 1 und Phase 2** und zusammen mit den oben genannten Materialien eingereicht. Nach Abgabe der dritten Portfolioseite reicht der Tutor innerhalb von sechs Wochen das endgültige Feedback ein, das die Bewertung und Bewertung umfasst.

2. TUTORIAL-UNTERSTÜTZUNG

Grundsätzlich stehen mehrere Kanäle offen, um Feedback zu den Portfolios zu erhalten. Die jeweilige Nutzung liegt in der alleinigen Verantwortung des Nutzers. Die eigenständige Entwicklung eines Produktes und die Bearbeitung der jeweiligen Portfolioteile ist Teil der Prüfungsleistung und geht in die Gesamtbewertung ein.

Die tutorielle Begleitung bietet einerseits Feedbackschleifen zu den einzureichenden Portfolioteilen im Rahmen der Konzeptionsphase sowie der Entwicklungs- und Reflexionsphase. Die Rückmeldung erfolgt im Rahmen einer Abgabe des jeweiligen Teilportfolios. Zusätzlich werden regelmäßig Online-Tutorials angeboten. Diese bieten Ihnen die Möglichkeit, Fragen zur Bearbeitung der Mappe zu stellen und weitere Themen mit dem Tutor zu besprechen. Der Tutor steht auch für fachliche und formelle Beratungen zur Verfügung

sowie allgemeine Fragen zum Verfahren der Portfolioverwaltung.

Technische Fragen zur Nutzung von „PebblePad“ sind per Mail an das Prüfungsamt zu richten.

3. BEWERTUNG

Zur Bewertung des Portfolios mit dem jeweils angegebenen Prozentsatz werden folgende Kriterien herangezogen:

Evaluationskriterien	Erläuterung	Gewichtung
Techniken zur Problemlösung	*Erfassen des Problems *Klare Problemdefinition/Zielsetzung *Verständliches Konzept	10%
Methodik/Ideen/Vorgehensweise	*Angemessener Transfer von Theorien/Modellen *Eindeutige Informationen über die gewählte Methodik/Idee/Vorgehensweise	20%
Qualität der Umsetzung	*Qualität der Umsetzung und Dokumentation	40%
Kreativität/Korrektheit	*Kreativität des Lösungsansatzes *Die implementierte Lösung erfüllt das beabsichtigte Ziel	20%
Formale Anforderungen	* Einhaltung formaler Anforderungen	10%

Bei der Gestaltung und Konstruktion des Portfolios sollten die oben genannten Bewertungskriterien berücksichtigt werden, einschließlich der folgenden Erläuterungen:

Problemlösungstechniken: Je nach Thema sollte eine Habit-Tracking-App entworfen, implementiert und dokumentiert werden. Es muss klar sein, wie die Anwendung unter Verwendung der übertragenen Dateien eingerichtet, ausgeführt und verwendet wird.

Methodik/Idee/Vorgehensweise: Das Konzept der Gewohnheiten und wie sie nach ihren Aufgaben und Zeiträumen verfolgt werden können, sollte einem soliden zugrunde liegenden objektorientierten Design folgen. Das Analysieren, Zusammenfassen und Aggregieren von Gewohnheiten im Laufe der Zeit sollte mithilfe funktionaler Programmiertechniken geeignet implementiert werden. Welche Design- und Technologieentscheidungen treiben das Projekt voran? Sind die Entscheidungen gerechtfertigt und im Rahmen dieses Portfoliokurses sinnvoll?

Umsetzungsqualität: Folgt die Umsetzung einem klaren Konzept? Kann es von einem projektfremden Anwender problemlos genutzt werden, zB durch Bereitstellung einer ausreichenden Dokumentation? Ist es einfach, mit der bereitgestellten Lösung neue Gewohnheiten zu erstellen und zu verfolgen? Ist es für Benutzer einfach, ihre Gewohnheiten mit dem Tool zu analysieren? Die in Kapitel 2 festgelegten Aufnahmekriterien sind einzuhalten.

Kreativität/Richtigkeit: Es wird bewertet, ob die spezifischen Anforderungen verstanden und auf verständliche und innovative Weise umgesetzt wurden. Die Grundfunktionalität der Anwendung muss durch Unit-Tests abgedeckt werden, um die Korrektheit der Software sicherzustellen.

Formale Anforderungen: Die Einreichung folgt den Annahmekriterien aus Kapitel 3 und den im nächsten Kapitel folgenden formalen Richtlinien. Es ist besonders wichtig, die in Kapitel 4 beschriebenen formalen Einreichungsanforderungen zu beachten.

4. FORMELLE RICHTLINIEN UND SPEZIFIKATIONEN FÜR DIE EINREICHUNG

4.1. Bestandteile der Prüfungsleistung

Nachfolgend erhalten Sie einen Überblick über das Prüfungsleistungsportfolio mit seinen einzelnen Phasen, einzureichenden Einzelleistungen und Feedbackstufen auf einen Blick. Für die Erarbeitung der Portfolioteile im Rahmen der Prüfungsleistung wird eine Vorlage in „PebblePad“ zur Verfügung gestellt. Die Präsentation ist Bestandteil
diese Prüfung.

Bühne	Zwischenergebnis	Leistung einzureichen
Konzeptionsphase	Portfolio Teil 1	• 1-3 Seiten schriftliches Konzept als PDF mit mindestens einem Diagramm zur Darstellung von Interaktionen und Abläufen.
		Feedback
Entwicklungsphase/ Reflexionsphase	Portfolio Teil 2	• Erläuterung der Umsetzung in schriftlicher Form als zusammengesetzte Präsentation PDF mit ca. 5-10 Folien.
		Feedback
Finalisierungsphase	Portfolio Teil 3	• 1–2-seitiges Abstract („Making of“), inklusive Link zum Projekt auf GitHub. • Endgültiges Softwareprodukt (Skripte mit Installationshandbuch und Dokumentation enthalten), das auf GitHub gehostet wird Laden Sie • den ZIP-Ordner hoch (inkl. aller Dateien) • Ergebnis aus Phase 1 • Ergebnis aus Phase 2
		Feedback + Note

4.2. Format für die Übermittlung digitaler Dateien

Konzeptionsphase	
Empfohlene Tools/Software für die Verarbeitung	Textverarbeitungstool Ihrer Wahl für Text, Online-UML-Editor wie „lucidchart“ für Visualisierungen.
Zulässige Dateiformate	Pdf
Dateigröße	so klein wie möglich/Vorschau

Weitere Formalitäten und Parameter Für die leistungsrelevanten Einreichungen auf „PebblePad“:

Name-Vorname_MatrNr_OOFPP_Gewohnheiten_Einreichung_Konzeption.pdf Beispiel: Mustermann-Max_12345678_OOFPP_Gewohnheiten_Einreichung_Konzeption.pdf

Das PDF-Dokument muss in PebblePad als „Mediendatei“ angehängt werden, der Einreichungstext „Konzeption“ kann leer bleiben oder eine kurze Beschreibung der Einreichung enthalten.

Entwicklungs-/Reflexionsphase	
Empfohlene Tools/Software für die Verarbeitung	PowerPoint oder ähnliches
Zulässige Dateiformate	Pdf
Dateigröße	so klein wie möglich/Vorschau

Weitere Formalitäten und Parameter Für die leistungsrelevanten Einreichungen auf „PebblePad“:

Name-Vorname_MatrNr_OOFPP_Habits_Submission_Development.pdf Beispiel: Mustermann-Max_12345678_OOFPP_Habits_Submission_Development.pdf

Finalisierungsphase	
Empfohlene Tools/Software für die Verarbeitung	Textverarbeitungstool Ihrer Wahl für das Abstract, Code muss auf GitHub gehostet werden
Zulässige Dateiformate	ÿ PebblePad: PDF ÿ GitHub: verschiedene Dateiformate für Ihre in Python implementierte Lösung, Hilfscode, z. B. Shell-Skripte usw., und andere zugehörige Dateien wie Dokumentation, Installationsanleitungen usw. ÿ Zip-Ordner: verschiedene Dateiformate für Einreichungen aus allen drei Phasen. Das Code, der über GitHub eingereicht wird, muss ebenfalls in den Zip-Ordner eingefügt werden, indem aus allen Dateien eine ZIP-Datei erstellt wird.
Dateigröße	so klein wie möglich

Weitere Formalitäten und Parameter **WICHTIG:** Sie müssen den eigens für die Einreichung erstellten Zip-Ordner hochladen (bitte folgen Sie den Anweisungen auf „myCampus“). Dieser Ordner enthält alle Dateien, die Sie zum Abschließen der Aufgabe verwendet haben. Zur besseren Übersicht legen Sie hierfür bitte Unterverzeichnisse an.

Die Ordnerstruktur sieht dann so aus: •

Hauptverzeichnis (Name des Zip-Ordners) -> Name:Name-Firs
Name_MatrNo_OOFPP_ Gewohnheiten
o Unterverzeichnis -> Name: OOFPP_Habits_Phase1 o
Unterverzeichnis -> Name: OOFPP_Habits_Phase2 o
Unterverzeichnis -> Name: OOFPP_Habits_Phase3

In Phase 3 sollten Sie zusätzlich eine gezippte Version (ZIP-Datei mit dem gesamten Programmcode) in einen Zip-Ordner mit folgender Namenskonvention hochladen:

Name-Vorname_MatrNr_OOFPP_Habits_Submission_Final.zip Beispiel:
Mustermann-Max_12345678_OOFPP_Habits_Submission_Final.zip

Für die Übermittlung Ihres Codes auf GitHub können Sie Ihren eigenen Benutzernamen frei wählen.
Sie erstellen ein öffentliches GitHub-Repository Ihrer Wahl und laden Ihren gesamten Code dorthin hoch. Das bedeutet, dass Ihr Code unter dem folgenden URL-Schema verfügbar sein sollte: https://github.com/<user_name>/<repository_name> Beispiel: https://github.com/maxmustermann/oofpp_habits_project

Für die leistungsrelevanten Einreichungen auf „PebblePad“, muss das 1-2-seitige Abstract (Making of) immer nach folgendem Muster benannt werden: Name-Vorname_MatrNr_OOFPP_Habits_Submission_Abstract.pdf Beispiel: Mustermann-Max_12345678_OOFPP_Habits_Submission_ Abstract.pdf

Fügen Sie in PebblePad den GitHub-Link in den Abschnitt „Informationen“ ein, laden Sie den ZIP-Ordner in den Abschnitt „Ressourcen“ hoch und laden Sie die Zusammenfassung sowohl in „Abstract“ als auch in „Final Product“ hoch. Stellen Sie sicher, dass Ihr Abstract auch einen Link zu Ihrem Projekt auf GitHub enthält.

4.3. Format des Abstracts

Länge	2 Textseiten
Papier größe	DIN A4
Ränder	Oben und unten 2 cm; links 2cm; rechts 2cm
Schriftart	Allgemeiner Text - Arial 11 pt.; Überschriften - 12 pt., Blocksatz
Zeilenabstand	1,5
Sätze	Gerechtfertigt; Silbentrennung
Fußnoten	Arial 10 pt., Blocksatz
Absätze	Je nach mentaler Struktur - 6 pt. nach Zeilenumbruch
Eidesstattliche Erklärung	Die eidesstattliche Versicherung ist in elektronischer Form über „myCampus“ abzugeben. Davor ist keine Abgabe der Prüfungsleistung möglich. Bitte folgen Sie den Hinweisen zum Einreichen einer Mappe auf „myCampus“.

Bei Fragen zur Abgabe der Mappe wenden Sie sich bitte per Mail an das Prüfungsamt.

Bitte beachten Sie auch die Hinweise zur Verwendung von PebblePad & Atlas!

Viel Glück beim Erstellen Ihres Portfolios!