# Lern- und Arbeitsauftrag 1305 Codes, NoSQL und Backend

|  |  |
| --- | --- |
| **Titel:** | **Modul 165, 306 - Gruppenarbeit** |
| **Modul:** | IMS-Lernatelier 2 Informatiker/in EFZ |
| **Autor / Version:** | Michael Schneider |
| **Sozialform** | Gruppenarbeit mit 1 bis 4 Personen |
| **Hilfsmittel:** | * Alle |

## Ausgangslage

In diesem Auftrag sollen folgende Module vertieft werden:

165 NoSQL-Datenbanken einsetzen

347 Dienst mit Containern anwenden

426 Software mit agilen Methoden entwickeln

Suchen Sie sich ein Projekt aus, das zu Ihrem Modulablaufplan passt oder wählen Sie ein eigenes Projekt zur Repetition des Stoffes oder zum Kennenlernen von neuen Technologien aus. Besprechen Sie eigene Projekte mit der Lehrperson.

Hinweis: Es ist nicht erlaubt, Aufgaben für das Qualifikationsverfahren (LB, LBV, …) während des Lernateliers zu realisieren. Das Lernatelier ist auch keine Aufgabenstunde.

Hinweis: Dimensionieren Sie Ihr Projekt so, dass Sie in der gegebenen Zeit fertig werden, aber auch genügend Arbeit vorhanden ist, um die Zeit zu füllen.

## Ziele

Am Schluss dieses Lern- und Arbeitsauftrages sollen Sie ein fertiges Produkt mit Projektdokumentation erstellt haben.

## Aufgaben

### 1) Informieren

a) Schliessen Sie sich in Gruppen zusammen, die mindestens aus eine, maximal aus vier Personen bestehen.

b) Wählen Sie in der Gruppe ein Projekt aus. Beachten Sie den Schwierigkeitsgrad, Ihren Wissensstand und die zur Verfügung stehende Zeit bei der Auswahl. Sprechen Sie eigene Ideen mit der Lehrperson ab.

c) Machen Sie eine Anforderungsanalyse. Werden Sie sich in der Gruppe einig, was Sie genau realisieren möchten. Das Projekt sollte durch die Anforderungen so präzise wie möglich beschrieben sein. Legen Sie die minimalen Anforderungen und «nice-to-have»-Anforderungen fest, die bei genügend Zeit realisiert werden können.

d) Legen Sie fest, wo und wie Sie in der Gruppe Arbeitsergebnisse speichern und austauschen möchten.

GitHub: <https://github.com/Kappa-X/FANSight>

Source:

Introduction: https://developers.google.com/ar

Boxi looking to the camera: <https://www.youtube.com/watch?v=_zF0fKHbO_c&t=1111s>

cloud picture: https://pixabay.com/de/vectors/wetter-sonne-sonnenschein-wolken-160922/

Quellen und etc. sind auch auf GitHub zu finden.

e) Erarbeiten Sie sich noch fehlende Grundlagen für das Projekt.

|  |  |  |
| --- | --- | --- |
| Nr. | Anforderung | Minimal oder nice to have |
| 1 | Der User kann das Programm als eine App auf seinem Handy starten | minimal |
| 2 | Boxi Model wird durch die Kamera in der echten Welt dargestellt | minimal |
| 3 | Boxi Model folgt der Kamera des Handys | minimal |
| 4 | Mit einem Button kann ein Ball gespawnt werden | minimal |
| 5 | Der Ball ist neben Boxi in der echten Welt zu sehen | minimal |
| 6 | Mit einem Button kann eine Wolke gespawnt werden | minimal |
| 7 | Die Wolke ist über Boxi in der echten Welt zu sehen | minimal |
| 8 | Beim spawnen wird auch der Regeneffekt der Wolke aktiviert | Nice to have |

### 2) Planen

a) Teilen Sie das Projekt in klare, einzelne Aufgaben auf («Arbeitspakete»).

b) Verteilen Sie diese Arbeiten an die Gruppenmitglieder und setzen Sie für jede Arbeit ein Abschlussdatum fest, dass mit dem Plan Ihrer Klasse übereinstimmt.

c) Erstellen Sie eine ToDo-Liste mit verantwortlicher Person, Arbeitspaket und Abschlussdatum für das ganze Projekt.

|  |  |  |
| --- | --- | --- |
| Arbeitspaket | Datum | Gruppenmitglied |
| Dokumentation erstellen | 03.05.2023 | Finn Neiger |
| Modellieren von Boxi | 03.05.2023 | Finn Neiger |
| Erstellen der Grundumgebung für ein AR-Programm | 03.05.2023 | Finn Neiger |
| Script für Boxi erstellen, um der Kamera zu folgen | 03.05.2023 | Finn Neiger |
| Ball modellieren | 10.05.2023 | Finn Neiger |
| Script für Button, um den Ball zu spawnen | 10.05.2023 | Finn Neiger |
| Wolke modellieren | 10.05.2023 | Finn Neiger |
| Script für Button, um die Wolke zu spawnen | 10.05.2023 | Finn Neiger |
| Hinzufügen des Regeneffekt | 17.05.2023 | Finn Neiger |
| Script erstellen, um den Regeneffekt mit der Wolke zusammen spawnen | 17.05.2023 | Finn Neiger |
| Builden des Programms und Start auf dem Handy | 17.05.2023 | Finn Neiger |
| Portfolio schreiben | 24.05.2023 | Finn Neiger |

### 3) Entscheiden

Fällen und dokumentieren Sie wichtige Entscheidungen.

Ich habe mich dazu entschieden die App für IOS zu programmieren, da ich ein iPhone besitze und deshalb die App für IOS builden muss, um später die App auf dem Handy auszuprobieren.

### 4) Realisieren

Erstellen Sie das Programm und die notwendigen Dokumente.

Die fertigen Produkte sind auf GitHub aufzufinden.

### 5) Kontrolle

a) Erstellen Sie Testfälle für das Programm.

|  |  |  |  |
| --- | --- | --- | --- |
| Anforderungs-Nr. | Voraussetzung | Eingabe | Erwartete Ausgabe |
| 1 | App muss sich auf dem Handy befinden | Klick auf App-icon | Die App startet |
| 2 | App wurde gestartet | Zugriff auf Kamera erlauben | Boxi ist durch die Kamera zu sehen |
| 3 | App gestartet und zugriff auf Kamera erlaubt | Schwenken des Handys | Boxi folgt der Kamera |
| 4 | App gestartet und Zugriff auf Kamera erlaubt | Auf roten Button mit dem Wort Ball klicken | Roter Ball wird neben Boxi gespawnt |
| 5 | App gestartet, Zugriff auf Kamera erlaubt und roten Button angeklickt | Kamera weg von Boxi schwenken | Roter Ball ist neben Boxi zu sehen |
| 6 | App gestartet und Zugriff auf Kamera erlaubt | Auf Wolken Symbol klicken | Wolke wird über Boxi gespawnt |
| 7 | App gestartet, Zugriff auf Kamera erlaubt und Wolken Symbol angeklickt | Kamera über Boxi schwenken | Wolke ist über Boxi zu sehen |
| 8 | App gestartet, Zugriff auf Kamera erlaubt und Wolken Symbol angeklickt | Kamer nach unten schwenken zu Boxi | Langsam kommen Regentropfen von der Wolke hinunter und treffen Boxi |

b) Testen Sie Ihr Programm und schreiben Sie ein kurzes Testprotokoll mit Testumgebung.

**Testprotokoll:**

### Testumgebung: iPhone 11

|  |  |  |
| --- | --- | --- |
| Testfall Nr. | Durchgeführt von | OK/NOK |
| 1 | Finn Neiger | OK |
| 2 | Finn Neiger | OK |
| 3 | Finn Neiger | OK |
| 4 | Finn Neiger | OK |
| 5 | Finn Neiger | OK |
| 6 | Finn Neiger | OK |
| 7 | Finn Neiger | OK |
| 8 | Finn Neiger | OK |

**Testbericht:**

Das Testen der App verlief ohne weitere Probleme und konnte einfach abgewickelt werden. Alle Funktionen sind so weit benützbar und funktionieren so, wie ich es geplant habe.

### 6) Auswertung

Überlegen Sie sich, was gut gelaufen ist und was eher nicht. Sie können diese Informationen in Ihrem Portfolioeintrag verwenden.

**Fazit:**

**Gut gelaufen: Obwohl das mein erstes grösseres Projekt war, an welchem ich alleine im Lernatelier gearbeitet habe, konnte ich die Grösse des Projekts schon ziemlich gut einschätzen und plante am Anfang nicht zu viele Arbeitspakete. Auch wurde das restliche Planen des Projekts einfacher, da man immer wusste, wo man etwa selber ist und sich nicht auf andere Teammitglieder verlassen musste, dass sie ihre Arbeit erledigen.**

**Schlecht gelaufen: Auch wenn das Planen einfacher wurde durch die fehlenden anderen Teammitglieder, war man schlussendlich auch ein bisschen auf sich selber gestellt, wodurch man nicht einfach bei anderen nachfragen konnte, wenn man ein Problem hatte. Ebenfalls ein Problem war das mühselige Exportieren der App auf IOS und das builden auf dem Handy, welches durch ein paar Fehlermeldungen fast einen ganzen morgen in Anspruch nahm.**

### 7) Portfolioeintrag

Schreiben Sie den Portfolioeintrag und reichen Sie ihn ein. Gehen Sie im Portfolioeintrag auf Ihre Ziele und deren Erreichung sowie die erstellte Dokumentation ein. Beschreiben Sie im Portfolioeintrag auch Ihren Lernweg, die Probleme, Erkenntnisse und Erfolge.

## Projektvorschläge

### 165 NoSQL-Datenbanken einsetzen

**Choose your own adventure**

«Choose your own adventure» sind Bücher, in denen Ihnen eine Situation geschildert und verschiedene Entscheidungsmöglichkeiten gezeigt werden. (<https://de.wikipedia.org/wiki/Spielbuch>). Erstellen Sie eine (Web-)Applikation, mit der Sie eine solche Geschichte durchspielen können. Speichern Sie alle Daten in einer NoSQL-Datenbank.

**Repetitionshilfe**

Erstellen Sie ein Programm, mit dem Sie Faktenwissen aus dem Unterricht repetieren können. Sie sollten Fragen und Antworten eingeben können und das Programm soll Sie abfragen. Erweiterungen wie eine Statistik oder andere Fragetypen wie Multiple-Choice-Aufgaben sind möglich. Speichern Sie alle Daten in einer NoSQL Datenbank.

**Medienmanager**

Erstellen Sie ein Programm, in dem Sie Medien (Bilder, Video, Links, Audio, …) speichern können. Dabei sollen alle möglichen Metadaten ausgelesen und in der Datenbank gespeichert werden können. Erstellen Sie eine Oberfläche, mit der Sie auf diese Metadaten zugreifen können.

**Gallery-Maker**

Erstellen Sie ein Programm, dem Sie eine Verzeichnishierarchie mit Bildern übergeben können und das Ihnen eine schöne Fotogalerie als statisches HTML ausgibt. Dabei soll es eine Übersicht mit Thumbnails und eine Detailansicht geben. Erweiterungsmöglichkeiten sind Texte, die zu den Bildern in einem geeigneten Format abgelegt werden können und das Auslesen der Metadaten.

### 347 Dienst mit Containern anwenden

**Docker Visualisierung mit phaser.io**

Mit phaser.io (<https://phaser.io/>) können Sie Spiele und visualisierungen für den Browser entwickeln.

Benutzen Sie die Ausgaben von «docker ps»,«docker info», etc und stellen Sie Ihre Docker als 3D-Szene im Browser dar:

* Ein Docker kann ein Gebäude sein.
* Je mehr Platz/Memory er benötigt, desto grösser ist es.
* Jeder Prozess ist eine Spielfigur im Haus
* …

Als Erweiterung könnten Sie beispielsweise Docker herunterfahren, indem Sie das Haus abbrechen (bitte mit Explosionen).

**Dockercraft (schwieriges Projekt!)**

Docker können innerhalb von Minecraft gesteuert werden: <https://github.com/docker/dockercraft>

Nehmen Sie dies als Basis für ein Projekt: Erweitern Sie Cuberite (den Minecraft Server) auf geeignete Art und Weise.

Achtung: Nur Minecraft Spielen ist kein Projekt!

### 426 Software mit agilen Methoden entwickeln

Implementieren Sie folgende Projekte mit den Design-Patterns im Hinterkopf. Versuchen Sie mindestens ein Pattern korrekt einzusetzen.

**RPG-Games**

Folgende Kurse beschreiben, wie man mit C# RPG’s bauen kann. Nehmen Sie diese objektorientierten Programme als Grundlage und erweitern Sie diese:

* <https://roguesharp.wordpress.com/>
* <https://scottlilly.com/learn-c-by-building-a-simple-rpg-index/>

**Textbasierte Multiuserwelt**

Ein MUD (MultiUser Dungeon) ist eine textbasierte Welt, in der mehrere Benutzer gleichzeitig Abenteuer erleben, sich bekämpfen oder Aufgaben erfüllen können.

Installieren Sie <https://github.com/DavidRieman/WheelMUD>, erstellen Sie eine Welt und mit neuen Objekten.

**Augmented Reality**

Mit ARCORE (<https://developers.google.com/ar>) können Sie Augmented Reality Applikationen entwickeln. Arbeiten Sie sich ein und erstellen Sie ein eigenes, einfaches Beispiel.



**Build your own x**

Wollten Sie schon immer einmal ein eigenes Betriebssystem, einen eigenen Renderer, eine eigene Datenbank, einen eigenen Webserver oder eine eigene Blockchain entwickeln? <https://github.com/codecrafters-io/build-your-own-x> hat Anleitungen dazu gesammelt.

## Gütekriterien

Der Lern- und Arbeitsauftrag ist erfüllt, wenn …

* Wenn Sie das Projekt nach IPERKA abgearbeitet haben.
* Wenn Sie eine lauffähige Applikation produziert haben.

## Zusätzliche Angaben zum Auftrag

Keine.

## Mögliche Erweiterungsaufträge

Keine.