Vorgehen Semesterarbeit 3i

Rahmen

Dieses Dokument bildet die Grundlage für Semesterarbeiten an der IMS.

Mit dieser Arbeit erhalten Sie eine weitere Gelegenheit, Ihre Programmier-Skills anhand eines konkreten Projektes zu erweitern und zu festigen. Als Applikationsentwickler erwartet man von Ihnen, dass Sie sich in mindestens einer Programmiersprache sehr gut auskennen, mehrere Sprachen sind wünschenswert. In dieser Arbeit soll daher eigenhändig codiert werden.

Weiterhin wird, wie bei der IPA, auch hier viel Wert auf eine tadellose Dokumentation gelegt. Anhaltspunkte und Anregungen für den Inhalt der Doku sind aus dem, dem Dossier beiliegenden Dokument "**Projektdossier Semesterarbeit 3i.pdf**" ersichtlich.

Auf der letzten Seite befindet sich der Bewertungsraster nach welchem Ihre Arbeit vom Betreuer bewertet wird.

Vorgehen

Gehen Sie bei Ihrer Arbeit grundsätzlich gemäss dem hier beschriebenen Ablauf vor und notieren Sie sich die Termine bereits jetzt im Kalender:

1. Das Projekt muss mit Hilfe des **Themenmeldeblatt SA 3i** (siehe Dossier), welches alle Angaben der übernächsten Seite (inklusive Unterschrift) enthalten muss, bei dem von Ihnen gewählten Betreuer und dem Abteilungsleiter (Kopie) termingerecht eingereicht werden. Das Themenmeldeblatt gilt als **Vertrag**.

   Fragen Sie für die Betreuung ihrer Arbeit einen **IMS-Informatiklehrer** an. Er wird Sie, sofern er noch freie Plätze hat und ihm die Idee machbar scheint und gefällt, betreuen.

   Die Betreuer nehmen **nur Arbeiten** an, bei denen der grösste Teil aus **eigenhändig geschriebenem Code** in einer bekannten **Programmiersprache** bestehen wird.

2. Erstellen Sie **bis spätestens Freitag vor den Herbstferien** eine Liste mit **mindestens 5 Meilensteinen** und senden Sie diese **per E-Mail** an Ihren Betreuer!

   Ein Meilenstein ist ein Zwischenziel, welches erreicht wurde oder werden sollte, also zum Beispiel der Abschluss einer Entwicklungsphase (z.B. "04.10.2022 | Analyse fertig").

3. Errichten Sie auf **GitLab** ein **Repository** in welches Sie den **Code** und die **Projektdokumentation** ablegen und laden Sie Ihren **Betreuer** dazu ein. Machen Sie häufig Commits und pushen Sie Ihr Projekt regelmässig (spätestens vor jedem Meilenstein) in dieses Repository.

4. Erstellen Sie eine **technische Projektdokumentation** (ohne persönliche Anmerkungen) in welcher Sie alle technischen Überlegungen, Entscheidungen, Planungen, Designs und Tests dokumentieren und darstellen (siehe auch Bewertungsraster). Für jede Phase des Wasserfallmodells (wenn vom Betreuer nicht anders verlangt) muss ein eigenes Kapitel angelegt werden.

5. **Kontaktieren Sie, bevor** Sie mit dem **Coding** beginnen, unbedingt Ihren Betreuer und allenfalls den Auftraggeber und besprechen Sie die **Analyse** und das **Design** mit Ihm.

6. Aktualisieren sie ihre Produktdokumentation **zu jedem Meilenstein** hin (fertig dokumentiert bis Stand Meilenstein) und senden Sie dem Betreuer einen **Projektfortschrittsbericht** (siehe Formular Projektfortschrittsbericht.pdf im Dossier) per E-Mail unaufgefordert zu.

Falls Sie einen Meilenstein nicht einhalten können, ist der **Bericht trotzdem** mit dem Grund und den nötigen Massnahmen die Sie treffen versehen, einzureichen. Können zukünftige Meilensteine (z.B. krankheitsbedingt) voraussichtlich nicht erreicht werden, so kann beim Betreuer ein “Antrag auf Verschiebung eines oder mehrerer Meilensteine” eingereicht werden, welcher von ihm zu genehmigen ist.

7. **Arbeiten** Sie gemäss einem strukturierten **Wochenplan** regelmässig an Ihrem Projekt.
8. Wenn sie **nicht weiter** kommen, grübeln sie nicht zu lange, fragen Sie Ihren Betreuer, Ihre Mitschüler usw.! Sie haben eine **Hol-Schuld**!
9. Erstellen Sie eine **Lerndokumentation**. Diese Doku beinhaltet alles Persönliche und allenfalls ein Arbeits-Journal (siehe Vorlage Lerndokumentation.doc im SA-Dossier).
10. Erstellen Sie eine **Installationsanleitung** und eine **Benutzeranleitung** und fügen Sie beides Ihrem Projekt auf GitLab hinzu.
11. Zum Schluss machen Sie folgendes:

    Lösen Sie einen letzten **Push** auf **GitLab** aus und erstellen Sie einen **Release/Tag** zu welchem Sie Ihr **Produkt gezippt hinzufügen**.

    Laden Sie das Produkt von GitLab auf einen anderen Rechner herunter und testen Sie, ob es läuft.

    Erstellen Sie ein Zip-File **"Semesterarbeit_*Jahr_IhrNachnameVorname*.zip"**, das folgendes beinhaltet:
    - Projektdokumentation (technisch)
    - Lerndokumentation (persönlich)
    - Benutzeranleitung
    - Installationsanleitung

    Überprüfen Sie das Zip-File vor der Abgabe nochmals indem Sie es **entpacken** und die Dokumente nochmals öffnen.

    Laden Sie das Zip-File unter **"Grundlagenfach-> IMS: Selbstständige Arbeit 3i -> "Arbeiten von Schülern"** aufs **portal.kftg.ch** und senden Sie ihrem **Betreuer zusätzlich eine E-Mail**. Die Arbeit gilt als rechtzeitig abgegeben, wenn Sie von ihrem Betreuer eine **Eingangsbestätigung** erhalten haben.

    Den **Abgabetermin** entnehmen Sie bitte dem Brief "Semesterarbeit in der 3. Klasse IMS" von Herrn Schnyder.
12. Ihr **Produkt** muss an einem von Herrn Schnyder festgesetzten **Termin** im Plenum (Klassen 2i, 1i und Eltern) **vorgeführt** und **präsentiert (verkauft!) werden** (für Tipps siehe auch Projektdossier Semesterarbeit 3i).

Wichtig: Die Arbeit muss **Code** enthalten und ist **eigenhändig zu erstellen** und allfällige Quellen sind gut zu dokumentieren. Bei Plagiaten, egal woher, muss mit der Note 1 und disziplinarischen Massnahmen gerechnet werden.

Themenmeldeblatt

CryptoMessage

Schüler Name(n): Pascal Meier

Betreuer Name: Sven Nüesch

Beschreibung der Idee: Mobilephone-Applikation mit welcher man SMS-Texte erfassen und dann verschlüsselt via Server von einem Handy auf ein anderes senden kann. Beim Empfänger entschlüsselt die gleiche Applikation die Meldung wieder und zeigt sie im Klartext an.

Skizze / Layout der Idee:

Programmier-Tools: Visual Studio Code, MySQL Workbench

Betriebssystem: Android

Programmiersprache(n): Java

Must have's:
- SMS Editor (deutsch)
- De-/Crypto-Algorithmus
- SMS Absenden
- SMS Empfangen

Nice to have's:
- Adressaten aus Adressbuch holen

Datum / Unterschrift(en) Entwickler: 10.8.2020 *Pascal Meier*

Unterschrift Betreuer:

Projektfortschrittsbericht

Siehe Formular im SA-Dossier

Projektfortschrittsbericht

Projekt:	Arbeitspaket:		
Mitarbeiter:		Kalenderwoche:	Datum:
Projektkurzbeschreibung:			
Projektfortschritt (in Prozent):			
Arbeiten, Ziele des Projektschrittes:			

Projektstand	**Status**			**Begründung** (nur bei gelb und rot)
Termine	◉	○	○	
Kosten	○	◉	○	
Produkt	◉	○	○	

Jedes dieser Felder kann grün, gelb oder rot sein!

Störungen/Probleme/Risiken:
Notwendige Entscheidungen:
Anlagen:

Bewertung

Gegenstand	Anforderung (sofern anwendbar)	max. Pkte
Doku Allgemein	Titelblatt (Titel, Version, Datum, Projektleitung, Autor), Inhaltsverzeichnis, Quellenangaben, Übersichtlichkeit (Inhaltsverzeichnis, Titel, Untertitel, Formatierung)	**2**
Analyse	Informationsbeschaffung (Tutorials, Code-Snippets), Anwenderbefragungen, Zielgruppen, Anforderungen (Must-Have's, Nice-To-Have's), Use-Cases, Use-Case Diagramm, Entity Relationship Model (ERM)*, HW-, SW-Umgebung inkl. deren Versionen (Kunde, Entwickler), Nutzwertanalyse verschiedener Lösungsvarianten, Installationsanleitung der Entwicklungsumgebung	**10**
Planung	Meilensteine festgelegt/eingehalten, Projektfortschrittsberichte da, Projektrollenverteilung, Aufwandabschätzung, Zeitabschätzung (evtl. Gantt-Chart), Kostenabschätzung, Design besprochen	**8**
Design	GUI-Designs, Wireframes (für alle Layouts gem. Use-Cases) mit Benutzer-Feedback (UX), Datenbankschema*, Klassenmodell*, Sequenzdiagramm*, Flussdiagramme, Zustandsdiagramme*, Test-Liste (mit detaillierten funktionalen und destruktiven Tests)	**10**
Realisation	Übersichtliche, gut strukturierte Implementation des Codes und der Datenbank*, korrektes anwenden unserer Coding-Konventionen, Dokumentierter Code*, Strukturiertes Vorgehen beim Einrichten des Servers, gute Usability/UX, Helpmenus* oder Kontexthelp vorhanden	**20**
Test	Getestet gemäss Testliste, Unit-Tests bestanden, Deployment-Test, User intention fulfilled, Feedback User, Applikation läuft stabil, Known Bugs List	**10**
Deployment	Publishing, Distribution (zip, App-Store) erstellt, beschriftet, Installationsanleitung da, Bulk-Dump* der Datenbank erstellt, Deployment-Scripts* da, Benutzeranleitung, Installer* vorhanden	**10**
Komplexitäts-bonus	Bis zu 10 Punkte werden zusätzlich vergeben, falls das Projekt eine hohe Komplexität aufweist	**10**
Verspätete Abgabe	Pro Tag Verspätung werden 15 Punkte abgezogen (Konventionalstrafe)	**-15**
Punkte Arbeit		**70**
Punkte Präsentation	Inhalt, Wissen, Aufbau/Gliederung, Zeiteinteilung, Sprache, Begriffsverwendung, Lautstärke, Deutliches Sprechen, Medienwahl und -einsatz (Bilder,Videos, 2nd Screen), Auftreten, Präsentation Demo	**20**
Punkte Total		**90**
Note**		

* Kann für gewisse Projekte obsolet sein. ** Die Note wird aus der Punktzahl linear berechnet.