Praktikumsbericht

über das Praktikum

bei

**valantic FSA**

vom

**10.01.2022**

bis zum

**xx.xx.2024**

von

**Janik Teune**

mit

**Matrikelnummer: 233966**

Inhaltsverzeichnis

[1 Einleitung 2](#_Toc182673512)

[1.1 Motivation 2](#_Toc182673513)

[1.2 Tätigkeitsübersicht 2](#_Toc182673514)

[2 Hauptteil 2](#_Toc182673515)

[2.1 Unternehmensprofil 2](#_Toc182673516)

[2.1.1 Eckdaten 2](#_Toc182673517)

[2.1.2 Unternehmensphilosophie 3](#_Toc182673518)

[2.2 Aufgabenbereiche 3](#_Toc182673519)

[2.3 Eine Anforderung – Speichern von Log und Konfigurationsdateien 4](#_Toc182673520)

[2.4 Eine Anforderung – Der „Rangeslider“ 4](#_Toc182673521)

[2.5 Arbeitsbedingungen 5](#_Toc182673522)

[2.6 Tabellarischer Wochenbericht 5](#_Toc182673523)

[3 Fazit 5](#_Toc182673524)

[3.1 Vergleich der Werksstudententätigkeit zum Berufspraktikum 6](#_Toc182673525)

[4 Literaturverzeichnis 7](#_Toc182673526)

# Einleitung

## Motivation

Das Schreiben von Programmen hat mich schon lange interessiert, doch während der ersten Semester konnte ich an der Universität nur wenig praktische Erfahrung sammeln. Die Module „Einführung in die Informatik“ und „Algorithmen und Datenstrukturen“ konnten grundlegende Konzepte des Programmierens anhand der Programmiersprache Java vermitteln. Die dort vorgestellten Probleme und Übungsaufgaben konnten allerdings in wenigen Zeilen Code gelöst werden. Das Arbeiten an einem größeren Projekt und in einem Team habe ich im Studium nur wenig kennengelernt.

Außerdem wurden eher grundlegende Strukturen der Programmiersprache Java vermittelt. Moderne und neue Ideen habe ich kaum kennengelernt.

Von meinem Arbeitsplatz bei valantic FSA erhoffte ich mir moderne Konzepte beim Programmieren kennenzulernen, sowie die Arbeit an einem großen Projekt zu erfahren.

Das Unternehmen habe ich durch das Sponsoring des Programmierwettbewerbs kennengelernt, welcher im Rahmen des Moduls „Algorithmen und Datenstrukturen“ stattfand. Das freundliche Auftreten der Mitarbeitenden sowie die Empfehlung eines Kommilitonen überzeugten mich schließlich, mich bei valantic FSA zu bewerben. Nach einem erfolgreichen Bewerbungsgespräch konnte ich am 10.01.2022 meinen ersten Arbeitstag und meine sechsmonatige Probezeit in der Abteilung iQbonds beginnen.

## Tätigkeitsübersicht

Zu meinen Aufgaben gehört:

* Spezifikation des Verhaltens neuer Anforderungen
* Implementation dieser Spezifikationen
* Fixen von Bugs
* Code Reviews
* Migration alter Projekte
  + Von SVN zu Git
  + Von Ant zu Gradle
  + Von älteren Java Versionen zu Java 21

# Hauptteil

## Unternehmensprofil

### Eckdaten

Die valantic GmbH ist ein Unternehmen, welches Beratungsdienstleistungen und Software in verschiedenen Branchen anbietet. Auf der Webseite sind unter anderem die Baubranche, Chemische und Pharmazeutische Industrie, das Finanzwesen, die Konsumgüterindustrie und die Kulturwirtschaft vermerkt. (1; 2)

valantic Financial Services Automation (kurz valantic FSA) ist ein Privatunternehmen, das zur valantic GmbH gehört und Finanzdienstleistungen anbietet. Es beschäftigt über 300 Mitarbeitende in 9 Büros in Europa und Nordamerika. Das Büro in Magdeburg befindet sich auf dem Werder in der Mittelstraße 10. Der Hauptsitz des Unternehmens befindet sich in Frankfurt am Main. (3; 4)

valantic FSA beschäftigt sich mit der Automatisierung von Prozessen in der Finanzindustrie. Die angebotene Software richtet sich an Banken. Das Ziel der Firma ist es, die Werteströme ihrer Kunden zu digitalisieren und weiterzuentwickeln. Damit will valantic FSA die Effektivität und Agilität verbessern sowie das Verständnis dafür steigern. (4; 5)

## Aufgabenbereiche

Ich arbeite in der Abteilung an verschiedenen Projekten, selten helfe ich auch noch in anderen Abteilungen aus.

Einen großen Teil meiner Arbeitszeit verbringe ich am „Diagnosetool“, welches vor allem von der Test-Abteilung verwendet wird. Die Mitarbeitenden können das Tool nutzen, um verschiedene Performance-Counter aus Log-Dateien auszulesen und sich diese tabellarisch und graphisch anzeigen zu lassen.

Häufig setze ich auch neue Anforderungen der Kunden um, zum Beispiel im iQbonds -client. Während ich als einziger Entwickler am Diagnosetool arbeite, handelt es sich beim iQbonds-client um ein großes Projekt, an dem seit vielen Jahren von vielen verschiedenen Mitarbeitenden gearbeitet wird. Gerade hier ist es notwendig, das gewünschte Verhalten eindeutig zu spezifizieren. Die Spezifikation wird vor Beginn der Implementation abgeschlossen und kann dann aus verschiedenen Perspektiven verifiziert werden. Unser Dokumentations-Team überprüft das Dokument auf sprachlicher Ebene, ein weiterer Entwickler verifiziert die technischen Aspekte und das Projektmanagement überprüft, ob das spezifizierte mit dem gewünschten Verhalten übereinstimmt. Während implementiert wird, kann jemand aus der Test-Abteilung anhand der Spezifikation diverse Testfälle ableiten, um das Verhalten des Programms zu verifizieren. Durch diese Vorgehensweise werden viele Bugs früh erkannt und das Verhalten ist eindeutig dokumentiert.

Nach dem Abschluss einer Implementation oder dem Fix eines Bugs werden die Änderungen in Gitlab reviewed. Das wird meist von meinem Ansprechpartner oder einem anderen Werksstudenten gemacht. Auch ich reviewe regelmäßig den Code von anderen Mitarbeitenden.

## Speichern von Log und Konfigurationsdateien

Im iQbonds-Client hatte der Kunde die Möglichkeit, die Konfiguration seiner Client-Instanz in ein ZIP-Archiv zu schreiben und dieses zu speichern. Damit kann zum Beispiel der Zustand dieser Instanz besser nachvollzogen und das Nachstellen eines Bugs erleichtert werden.

Diese Funktion habe ich erweitert. Mit der java-swing-API habe ich einen Dialog gebaut, wo der Kunde mithilfe von Kontrollkästchen auswählen kann, welche Dateien in ein ZIP-Archiv hinzugefügt werden.

Die Anforderung habe ich zunächst spezifiziert, sodass die Testabteilung diverse Blackbox-Tests vorbereiten und das spezifizierte Verhalten vom Projektmanagement bestätigt werden kann und danach implementiert.

Der Kunde hat nun die Optionen

* Die Konfigurationsdateien des Clients zu speichern
* Die Log-Dateien des Clients zu speichern
* Ältere archivierte Log-Dateien zu speichern
* Die Konfigurationsdateien des iQbonds-Servers anzufragen und diese ebenfalls zu speichern

Das Speichern der Dateien des Clients war einfacher, da alle Daten zum Speichern der Dateien bereits vorliegen. Zur Speicherung der Serverkonfiguration muss allerdings eine Nachricht an den Servergesendet werden. Danach muss auf die Antwortnachricht des Servers gewartet, diese Nachricht ausgewertet und transformiert werden und schließlich im XML-Format der Konfigurationsdateien gespeichert werden.

Zusätzlich habe ich im Zuge dieser Anforderung den FileChooser-Dialog der java-swing-API erweitert. Die Erweiterung validiert den eingegebenen Dateinamen und kann eine Auswahl an Fehlermeldungen anzeigen. Sollte der Dateiname invalide sein, so wird der Bestätigungsknopf deaktiviert. (6)

## Der „Rangeslider“

Wenn das Diagnosetool Log-Dateien importiert, kann man einen Start- und Endzeitpunkt für die Auswertung der Counter-Daten auswählen. Für diese Auswahl habe ich einen „Rangeslider“ implementiert. Der Rangeslider ist funktioniert genauso wie der JSlider aus der java-swing-API. Der Unterschied ist, dass es 2 Knöpfe auf dem Slider gibt. Die Knöpfe können nicht aneinander vorbeigezogen werden. So kann der linke Knopf zur Auswahl der Startzeit und der rechte Knopf zur Auswahl der Endzeit verwendet werden. (7)

## Git-Migration der „User Guides“

Das Dokumentationsteam hat für jedes Projekt nochmal ein eigenes Projekt „User Guide“. In diesen Projekten arbeitet das Dokumentationsteam mit einem externen Tool, um den Kunden in verschieden Varianten (PDF, HTML5, …) einen umfassenden Guide zu dem entsprechenden Projekt in Deutsch und Englisch zur Verfügung zu stellen.

Zur Versionsverwaltung der User Guide Projekte wurde Subversion (kurz: SVN) verwendet.

Nun war es meine Aufgabe jedes einzelne dieser Projekte von SVN zu Git zu migrieren. Dafür wurde mir ein Tool „svntogit“ zur Verfügung gestellt, welches mir die meiste Arbeit abnehmen konnte.

Der grobe Ablauf für die Migration eines Projektes ist der folgende:

1. Svntogit ausführen
   1. Den alten Projektpfad angeben
   2. Den neuen Projektpfad angeben
   3. Die Ausführung dauert mehrere Stunden
   4. Das Ergebnis ist ein Git-Repository mit dem Namen „repo“
   5. Alle Commits, Branches und Tags aus dem alten SVN-Repository wurden übernommen
2. Das neue Git-Repository dem Projekt entsprechend umbenennen
3. Die .gitignore und .gitattributes Dateien anpassen
4. Das Projekt auf Gitlab anlegen und das lokale Repository hochladen
5. Die Pipeline aufsetzen
6. Auf dem alten SVN-Repository markieren, dass dieses Projekt migriert wurde

Da es sehr viele User Guide Projekte gibt, habe ich die Projekte natürlich nicht alle händisch migriert. Stattdessen habe ich ein Batch-Skript geschrieben, was anhand einiger Parameter die Schritte der Reihe nach für mich durchführt.

Außerdem habe ich das Dokumentationsteam im Umgang mit Git und Gitlab geschult.

Die Aufgabe klingt für einen Softwareentwickler zwar etwas langweilig, dafür konnte ich mich aber mit Windows-Batch-Skripten, Subversions und Gitlabs Commandline Interface und mit der Git Pipeline beschäftigen.

## AppInvoker Simulator

Eine andere Anforderung, die ich für unsere Testabteilung umgesetzt habe, ist der AppInvoker Simulator. Der AppInvoker ist ein Programm, welches verschiedene Befehle, die vom Kunden im Client konfiguriert werden können, an Bloomberg oder an Eikon schicken kann. Der Kunde kann jeden konfigurierten Befehl in einem Menü auswählen. Entsprechend wird der Befehl an den Server und von dort über den AppInvoker entweder an Bloomberg oder an Eikon gesendet.

Um hier das Verhalten zu testen, musste der entsprechende Tester eine externe Test-verbindung herstellen und darüber die entsprechenden Antworten, für die das Verhalten getestet werden sollte, abschicken.

Der Simulator soll dazu dienen, den Testern viel Zeit und Arbeit zu ersparen.

Dafür wird eine separate Verbindung zum Server hergestellt. Jede Nachricht, die vom Server an den AppInvoker gesendet wird (unabhängig davon, ob es gerade eine gestartete AppInvoker-Instanz gibt) kann auch der Simulator empfangen. Der Tester kann hier in einer Liste alle empfangenen AppInvoker-Nachrichten sehen. In einem Menü kann der Tester für jede dieser Nachrichten diejenige Antwort auswählen, für die das Verhalten getestet werden soll. Anstatt auf eine Antwort von Bloomberg oder Eikon warten zu müssen, bzw. eine Testverbindung herstellen zu müssen, kann nun also direkt eine Antwort ausgewählt und zurück an den Server geschickt werden.

Alle ein- und ausgehenden Nachrichten werden zusätzlich gelogged.

**TODO**

## Arbeitsbedingungen

Bis auf die ersten paar Tage habe ich aufgrund der Corona-Pandemie im Home-Office gearbeitet. An meinem Arbeitsplatz stand ein Rechner im Büro und ich habe mich über ein Programm mit meinem privaten Gerät auf diesen Rechner eingewählt. Nachdem sich die Lage entspannt hatte, wurde mir freigestellt, weiterhin im Home-Office zu arbeiten oder an meinen Arbeitsplatz im Büro zu wechseln und ich habe bis auf wenige Ausnahmen meinen Arbeitsplatz im Büro verwendet. Seit meinem Umzug im April arbeite ich fast ausschließlich im Home-Office.

# Fazit

Insgesamt schätze ich meine Werksstudentenstelle als sehr positiv ein. Der Anspruch und die Aufgaben sind abwechslungsreich und ich kann sehr breitgefächert neue Erfahrungen bei der Arbeit als Softwareentwickler sammeln.

Neue Erkenntnisse konnte ich nicht nur über die Programmiersprache Java erhalten, sondern auch über Build-Umgebungen wie Ant oder Gradle.

Im Umgang mit Git habe ich gelernt, Pipelines in Gitlab aufzusetzen und mit komplexeren Projektstrukturen umzugehen. Dazu gehört zum Beispiel das Arbeiten an verschiedenen Versionssträngen und deren Synchronisation durch Cherry Picks.

Auf der anderen Seite muss ich viel mit Legacy code arbeiten, der an vielen Stellen undokumentiert ist. Das kommt gerade im Diagnosetool häufig vor, da dieses Projekt fast nur von Werksstudenten betreut und dadurch von vielen unterschiedlichen Entwicklern mit teilweise wenig Erfahrung implementiert wurde.

Dieser Umstand stört mich zwar, aber ich denke, dass dieses Problem in vielen Unternehmen vorhanden ist, und ich jetzt weiß, was mich als Softwareentwickler in der Industrie erwartet.

Abschließend denke ich, dass mir die Arbeit als Werksstudent bei valantic FSA ein gutes Bild davon gibt, wie ein Vollzeitjob als Softwareentwickler aussehen kann.

## Vergleich der Werksstudententätigkeit zum Berufspraktikum

Im Gegensatz zu einem richtigen Berufspraktikum habe ich nicht 12 Wochen in Vollzeit gearbeitet. Ich habe auch kein Bachelorprojekt oder Praktikumsprojekt bearbeitet. Warum sollte meine Werksstudententätigkeit also als Berufspraktikum angerechnet werden?

Aus der Praktikumsordnung vom 07.02.2011 **§1 Zweck des Praktikums**:

„Das Praktikum hat das Ziel, die Studierenden […] bei der Bearbeitung einer komplexen Aufgabenstellung mit Arbeitsverfahren, -mitteln und -prozessen des jeweiligen Fachgebietes sowie mit organisatorischen und sozialen Verhältnissen der Praxis bekanntzumachen. Das Praktikum soll die Fähigkeit zur Teamarbeit fördern. Neben der fachspezifischen Tätigkeit soll die Praktikantin bzw. der Praktikant auch um den Erwerb von Kenntnissen über Betriebsorganisation, Sozialstrukturen, Sicherheits- und Wirtschaftlichkeitsaspekte bemüht sein.“ (8)

Tatsächlich wird in einem Praktikum oft ein mittelgroßes bis großes Projekt umgesetzt und die/der Praktikant\*in kann dabei die Prozesse des Unternehmens, sowie die Betriebsorganisation, Sozialstrukturen usw. zu großen Teilen kennenlernen. Allerdings ist es für ein Unternehmen oft nicht zielführend, viele Ressourcen (z.B. die Zeit der anderen Mitarbeitenden) auf eine Arbeitskraft aufzuwenden, die nur so kurzfristig angestellt ist. Ein Praktikum endet oft schneller, als eine richtige Einarbeitung in das Unternehmen benötigt. Meine Werksstudentenstelle habe ich hingegen seit fast drei Jahren. In diesem Zeitraum konnte ich die Prozessstrukturen des Unternehmens nicht nur kennenlernen, sondern auch verstehen. Ich habe die Zusammenarbeit mit verschiedenen Mitarbeitern erfahren und in unterschiedlichen Projekten mitgearbeitet. Ich denke, dass man durch eine Werksstudentenstelle - gerade durch den langen Zeitraum - den oben genannten Zweck des Praktikums mindestens genauso gut erfüllt wie bei einem Praktikum.

# Literaturverzeichnis

1. **valantic. *valantic.* [Online] 29. 06 2024. https://www.valantic.com/en/.**

**2. Wikipedia valantic. *Wikipedia.* [Online] 29. 06 2024. https://de.wikipedia.org/wiki/Valantic.**

**3. LinkedIn valantic FSA. *LinkedIn.* [Online] 29. 06 2024. https://de.linkedin.com/company/valantic-fsa.**

**4. About Us: valantic FSA. *valantic FSA.* [Online] 01. 07 2024. https://www.valantic.com/fsa/about-us/.**

**5. Wikipedia Finanztechnologie. *Wikipedia.* [Online] 29. 06 2024. https://de.wikipedia.org/wiki/Finanztechnologie.**

**6. Oracle Docs - JFileChooser. *Oracle Java Platform.* [Online] 16. 11 2024. https://docs.oracle.com/javase/8/docs/api/javax/swing/JFileChooser.html.**

**7. Oracle Docs - JSlider. *Oracle Java Platform.* [Online] 16. 11 2024. https://docs.oracle.com/javase/8/docs/api/javax/swing/JSlider.html.**

**8. FIN, OVGU - Studiendokumente Praktikumsordnung. *FIN, OVGU.* [Online] 01. 09 2010. https://www.verwaltungshandbuch.ovgu.de/H%C3%B6B+Teil+I/1\_07+Praktikumsordnungen-media\_id-2010-p-52.html.**

Hiermit erkläre ich, dass ich diesen Praktikumsbericht selbstständig und ohne fremde Hilfe verfasst und keine anderen Hilfsmittel als angegeben verwendet habe.

Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als solche kenntlich gemacht habe.

Ort:

Datum: Unterschrift: