Projektdokumentation

Linus Herterich – linus.herterich@stud.htwk-leipzig.de
Nächster Autor – E-Mail
HTWK Leipzig

Inhaltsverzeichnis

I. Anforderungsspezifikation

I.1 Initiale Kundenvorgaben

Autor: Jonas Gwozdz, Korrektur: Linus Herterich

Die Vorgaben unseres Kunden, Prof. Gürtler, ließen uns viele Freiheiten in der Gestaltung des Programms.
Die gegebenen Vorgaben legten das Folgende fest:

- Für die Herstellung eines Werkstücks gibt es in der Regel mehrere Alternativen. Ausgehend von einem Rohteil folgen verschiedene Bearbeitungsschritte, die zum Fertigteil führen. Die einzelnen Bearbeitungsschritte sollen mit einer Bearbeitungszeit und Kosten belegt werden.
- Im Tool soll ein horizontaler Graph erstellt werden können, auf dem die Teile als Knoten und die Bearbeitungsschritte zwischen ihnen als Kanten dargestellt sind. Im nachhinein soll der optimale Fertigungsprozess nach Kosten oder Zeit ausgewertet werden. Alle hierfür benötigten Daten sollen im Tool eingegeben werden können
- Rüstkosten und -zeit fallen pro Charge (festlegbar per Losgröße, mindestens aber ein mal) bei jedem Bearbeitungsschritt an. Zusätzlich werden pro Stück für jeden Bearbeitungsschritt die benötigten Kosten und Zeit mit einbezogen.
- Das Tool soll bei Vorgabe all dieser Größen die günstigsten Varianten (nach Kosten, Zeit oder einer Kombination der beiden) berechnen und die nächstbesten Varianten ausgeben.

I.2 Produktvision

Autor: Alex Hofmann

Product Vision Board:

Target Group	**Needs**	**Product**
-Maschinenbau-Studenten Maschinenbau-Profs -Lehrende	Vgl. zu händisch: einheitlicher, schneller -plattformunabhängig -Open Source -Einfach zu bedienen	-Webanwendung -Als Graph → quasi als Baukasten → Kantengewichtung, Bausteine wählbar -Import/Export von Modellen Normalisierung des Graphen

Die Webanwendung VarG wird entwickelt für Lehrende und Lernende aus dem Maschinenbau Bachelorstudiengang. Diese erleichtert die einheitliche Erstellung, Bearbeitung, Optimierung sowie Im- bzw. Exportierung von sogenannten Variantenfolgegraphen, kurz VarGraphs. Darunter ist eine graphische Übersicht zu verstehen, die die möglichen Varianten eines Produktionsprozesses für ein Werkstück darstellt.

Später überarbeitete Produktvision bzw. neue Projektbeschreibung:
Die plattformunabhängige Open-Source Webanwendung "VarG" soll es Lehrenden und Lernenden

aus dem Studiengang Maschinenbau ermöglichen, einfach und schnell Variantenfolgegraphen, kurz VarGraphs, zur Herstellung von Werkstücken zu visualisieren und nach verschiedenen Kriterien die günstigsten Wege berechnen und anzeigen zu lassen. Dafür stehen ihnen viele Features für Aufbau, Funktionsweise, Design, Export und Import zur Verfügung.

I.3 Liste der funktionalen Anforderungen

Autor: Erik Heldt

- Erstellen von Zuständen (Teilen) mit Namen & Kürzel
- Erstellen von Bearbeitungsschritten mit Namen & Kürzel zwischen je 2 Zuständen
- Zuweisen von Rüstzeit, Geldkosten & Losgröße zu Bearbeitungsschritten
- Anzeigen des günstigsten Weges im Graph, berechnet nach Zeit oder Kosten
- Lokaler Export als Bilddatei oder importierbarer JSON & Lokaler Import als JSON
- Hochladen in online gehostete Datenbank & Laden aus online gehosteter Datenbank
- Login-Management für Zugriffskontrolle auf Anwendung
- Rollen-Management (Student, Professor) für Zugriffsrechte auf Datenbank

I.4 Liste der nicht-funktionalen Anforderungen

Autor: Erik Heldt

- Schnelle Einarbeitung in die Anwendungsumgebung
- Einfacher & intuitiver Umgang mit den Programmkomponenten und -funktionen
- Stabiler & konsistenter Programmablauf, keine Abstürze oder Verluste von Dateien
- Kompatibilität mit so vielen modernen Browsern wie möglich
- Sicherheit & korrekte Funktionalität des Login-Algorithmus und des DB-Rollenmanagements
- Datenschutz bei Login-Sessions einhalten

I.5 Weitere Zuarbeiten zum Produktvisions-Workshop

Autor: Erik Heldt

Für den Produktvisions-Workshop wurden 4 Dokumente erstellt, welche unterschiedliche Aspekte des Anwendungsentwurfs behandeln:

- Ideen zur Darstellung der GUI inklusive eines interaktiven GUI-Prototyps auf Adobe XD
- Epic bzw. eine Zusammenfassung vieler User-Stories zu allgemeinen Anforderungen an die Funktionalität
- Formulierung der Kernfunktionen des Programms
- Datenmodellierung des Programms

Die nachfolgende Liste an Zuarbeiten sind klickbare Verweise auf die jeweiligen Dokumente im ”zuarbeiten”-Ordner.

I.5.1 Zuarbeit von Linus Herterich, Jonas Gwozdz, Julius Hohlfeld

VarG GUI

I.5.2 Zuarbeit von Erik Heldt, Alaa Aldin Karkoutli

Erstes Epic

I.5.3 Zuarbeit von Lennart Buchmann, Nils Buxel, Matthias Berger

Kernfunktionalität

I.5.4 Zuarbeit von Tim Henning, David Koch

Datenmodell

I.6 Liste der Kundengespräche mit Ergebnissen

Autor: Manuel Eckert

Datum	Grund	Ergebnis
30.10.2019	Kickoff	Überblick über die Anforderungen, angenähertes Verständnis für das Produkt, Kennenlernen des Kunden,
27.11.2019	Produktvision	Abgleich Produktvision, Weiteres Verständnis für das Produkt
21.01.2020	Zwischenstand	Auslieferung des Ersten Produktinkrements, Feedback zum ausgelieferten Produkt
08.07.2020	Abnahme & Retrospektive	Auslieferung des Produktes, Rückblick und Feedback auf den Projektverlauf

Aufgrund der besonderen Situation im Sommersemester 2020 haben wir unser Projekt auf einen Webserver gespielt. Ab diesem Zeitpunkt haben wir hauptsächlich mit E-Mails kommuniziert um Feedback und Anforderungen zu besprechen.

II. Architektur und Entwurf

II.1 Zuarbeiten der Teammitglieder

Autor: Erik Heldt

Für die Technologierecherche informierte sich das Team über verschiedene Technologien, mit denen die Anwendung entwickelt werden kann. Außerdem fassten wir erste Ideen zur Graphenanordnung zusammen und legten Coding Conventions fest. Die Ausarbeitungen wurden in den nachfolgenden Dokumenten festgehalten.

Die nachfolgende Liste an Zuarbeiten sind klickbare Verweise auf die jeweiligen Dokumente im "zuarbeiten"-Ordner.

II.1.1 Zuarbeit von Tim Henning

Django und Python

II.1.2 Zuarbeit von Erik Heldt

Ruby on Rails
Ruby on Rails (kurz)
Graphenanordnung

II.1.3 Zuarbeit von David Koch

Java Canvas
Datenbanken

II.1.4 Zuarbeit von Matthias Berger, Nils Buxel

Coding Guidelines

II.1.5 Zuarbeit von Nils Buxel

Coding Conventions CSS

II.1.6 Zuarbeit von Julius Hohlfeld

Angular

II.1.7 Zuarbeit von Lennart Buchmann, Alaa Aldin Karkoutli, Jonas Gwozdz, Linus Herterich

Bibliotheken zur Graphenerstellung

II.2 Entscheidungen des Technologieworkshops

Autor: Erik Heldt

Nach ausgiebigen Recherchen über verschiedenste Programmiersprachen, Frameworks und Bibliotheken entschieden wir uns für eine Webanwendung auf Basis von HTML/CSS/JavaScript.

Wir haben uns weiterhin auf das JS-Framework Vue.js geeinigt, da es viele Vorteile für die Front-End-Entwicklung mit sich bringt und von den vielen untersuchten Frameworks am intuitivsten erschien. Außerdem haben wir nach einer JS-Bibliothek zur Graphdarstellung recherchiert und unter verschiedenen Kandidaten stach Cytoscape mit seinen vielen Funktionen zur Graphenerstellung und -editierung am meisten heraus, was wir somit auch in unsere Architektur integrierten.

Bei der Programmierumgebung waren wir uns schnell einig, dass Visual Studio Code am besten für unsere Ansprüche geeignet ist. Wir installierten die IDE zusammen mit dem Plugin ESLint zur Unterstützung der Einhaltung standardmäßiger Coding Conventions.

II.3 Überblick über Architektur

Autor: Linus Herterich

VarG ist eine Web-App nach dem Client-Server Modell, wobei der Großteil der Berechnungen per JavaScript auf dem clientseitigen Browser durchgeführt werden.
Serverseitig wird eine Datenbank (inkl. API-Schnittstelle) zum persistenten Speichern der erstellten Graphen angeboten.

Die Architektur der Web-App basiert auf dem JavaScript-Webframework "Vue.js", mit dem Webanwendungen nach dem MVVM Muster (Model View ViewModel) realisiert werden können. Die

gesamte App ist nach logischen Sites (Seiten, bei denen sich die URL ändert) und Components (wiederverwendbare, abgeschlossene Software-Schnipsel) aufgebaut. Jede Vue Component (.vue Dateien) enthält ein HTML-Template (GUI), sowie Daten, mit denen das Template befüllt wird. Zudem werden Funktionen definiert, die entweder zu bestimmten Laufzeitbedingungen der App oder durch Events und Trigger aufgerufen werden. Die Kommunikation zwischen Components wird über Vererbungen zu Eltern-/ Kind-Components realisiert.
Die Web-App besteht im Entwicklungszustand aus vielen hunderten Dateien, welche vom Framework verwaltet werden. Sobald die App in den Produktionsstatus wechselt, muss das Projekt kompiliert werden. Dies übernimmt ebenfalls das Framework, welches hierfür Technologien wie "WebPack" einsetzt. So bleiben lediglich wenige HTML, CSS und JavaScript Dateien übrig, die anschließend auf einem Web-Server (z.B. Apache) zur Verfügung gestellt werden müssen.

Um die Darstellung einheitlich zu halten, haben wir die UI-Bibliothek 'Vuetify" genutzt. Diese hält sich an den Industriestandard "Material Design" von Google. Damit konnten wir alle unsere im Vorfeld erstellten Design-Konzepte umsetzen. Um an den "Vuetify" Elementen weitere optische Anpassungen vorzunehmen haben wir die CSS-Language-Extension "less" verwendet. Mit dieser ist es möglich, übersichtliche und einheitliche Style-Vorgaben auf die Design-Komponenten anzuwenden.

Damit alle Daten komponentenübergreifend auf einen gemeinsamen Datenstamm zugreifen können und die Daten auch nach einer Session persistent gespeichert werden können, haben wir die Vue.js-Erweiterung "Vuex" eingesetzt. Diese bietet eine zentralisierte Speichermöglichkeit für alle Daten, die übergreifend verwendet werden müssen (beispielsweise Log-In Daten oder der Zustand des VarGraphs).

Für die Darstellung des Graphen (Knoten + Kanten und deren Beschriftung) haben wir die JavaScript Bibliothek "Cytoscape.js" verwendet. Die Bibliothek hält alle Graph-Daten in einem JavaScript Objekt, auf das mit verschiedenen API-Funktionen zugegriffen werden kann. Die Darstellung des Graphen wird über ein Canvas HTML Element realisiert, in welches Cytoscape die angelegten Knoten und Kanten zeichnet. Cytoscape bietet ebenfalls eine Hand voll Algorithmen zur analytischen Auswertung des Graphen. Da die Optimierung des VarGraphs allerdings zusätzlichen Bedingungen und Parametern unterliegt, wurde ein eigener VarGraph-Optimierungsalgorithmus entwickelt.

Bei der Wahl der serverseitigen Architektur haben wir eine REST-konforme (Representational State Transfer) Architektur eingesetzt, an dessen Ende eine MySQL Datenbank zur Speicherung der Cytoscape Objekte, sowie Authentifizierungsdaten steht. Auf die Daten der Datenbank greift eine API-Schnittstelle zu, welche mit Node.js umgesetzt ist (weitere Details zur Schnittstelle: siehe II.4 - Schnittstellen). Anfragen an die API werden mit dem "axios" Framework per "Promise-based" HTTP-Requests gestellt. Die HTTP-Requests folgen einem klaren Schema, welches vom serverseitigen Node.js interpretiert und an die Datenbank weitergeleitet wird.

Um die Web-App großflächig zu testen haben wir uns zum einen für das Framework "Cypress" entschieden, welches Integrationstests anhand der HTML-Elemente übernimmt. Cypress wertet aus, ob bestimmte Elemente unter bestimmten Bedingungen vorhanden sind beziehungsweise spezielle Eigenschaften aufweisen. Die Cypress Tests haben wir auch erfolgreich an die "CI / CD Pipeline" von GitLab angeschlossen, sodass nach jedem push die Tests durchlaufen (Stichwort: Regressionstest).
Zum anderen haben wir das Framework "jest" für Unit-Tests eingesetzt, mit dem einzelne Funktionen auf ihre Richtigkeit überprüft werden können. Vor allem für die Optimierungsalgorithmen sind isolierte Tests nötig gewesen.

Um eine Client-Server Architektur zu simulieren haben wir "Docker" eingesetzt. Dieses Tool erlaubt es, virtuelle Maschinen zu erstellen, welche untereinander kommunizieren können. Für Entwicklungszwecke haben wir einen Docker-Container für eine MySQL Datenbank und einen Node.js-Webserver (API Schnittstelle) erzeugt. Ein weiterer Docker-Container wurde eingesetzt, auf dem "Adminer" läuft. Mit diesem Tool ist es möglich, die MySQL-Datenbank komfortabel anzuzeigen und SQL-Zugriffe auszuführen.

II.4 Definierte Schnittstellen

Autor: Julius Hohlfeld

VarGs Funktionalitäten erfordern eine Datenbank, um die erstellten Graphen speichern und wieder abrufen zu können.
Um den Zugriff auf die Datenbank zu kontrollieren benötigen wir eine definierte Schnittstelle (bzw. API) zwischen Client, Webserver und Datenbank.

Diese Schnittstelle ist RESTful - d.h. sie folgt einigen der sog. REST-Constraints. Eine Übersicht zu REST und dessen Bedeutung für das Projekt findet sich im GitLab Wiki unter "API Dokumentation".
Die Schnittstelle setzt sich wie folgt zusammen:

- **Vue**
  Framework für Client + Axios-Module für asnychrone (promise-based) HTTP-Requests

- **Express**
  Serverseitiges Node-Module für Webserver: hört angemeldete Ports auf Requests ab, die dem URI-Modell entsprechen

- **Node.js**
  Serverseitige Programmierung des Webservers mit mysqljs als Driver, um auf die Datenbank zuzugreifen

- **DB**
  MySQL-Datenbank auf extra Server

Diese Struktur (kurz VenDB) entspricht einer Anpassung des sog. MEAN-Stacks auf das VarG-Projekt (MongoDB, Express, Angular, Node.js).
Dabei erfolgt jeglicher Austausch der Graphdaten im JSON-Format, damit auf die Cytoscape-Funktion zum Laden des Graphen zugegriffen werden kann.

II.4.1 Client

Der Client enthält Trigger durch Events, welche Requests an den Webserver senden. Z.B.: das Aufrufen des Datenbankfenster löst eine Anfrage aus, welche alle Graphen des aktuellen Nutzers abfragt. Diese werden durch das Axios-Modul umgesetzt. Nachdem der Trigger ausgelöst wird, schickt der Client eine asynchrone Request. Diese wird vom Webserver verarbeitet, welcher dann eine Antwort schickt. Diese kann von Axios aufgefangen werden (axios."request"(url,).then(response =>).catch(error =>)).

II.4.2 Server

Der durch Express und mysqljs programmierte Webserver definiert folgende mögliche Zugriffstellen auf die Datenbank:

- **Get-Requests**
  - **graph**
    Fragt alle Graphen aus der Datenbank ab - für Admin reserviert.
  - **graph/:id?**
    Fragt einen spezifischen Graphen (entsprechend der ID) ab.
  - **graph/meta**
    Fragt Metadaten z.B.: Namen, Id, Stückzahl usw. ab für die Graphen des Nutzers ab.
- **Put-Requests**
  - **graph/:id?**
    Client schickt Server eine Repräsentation des Graphen in Json um einen bereits existierenden Graphen (entsprechend der ID) zu überschreiben.
- **Post-Requests**
  - **graph?**
    Client schickt Server eine Repräsentation des Graphen in Json um einen neuen Eintrag für den Nutzer zu erzeugen.
- **Delete-Request**
  - **graph/:id?**
    Spezifizierter Graph (entprechend der ID) wird aus der Datenbank gelöscht.

Das '?' bedeutet, dass hier auf bestimmte URL Queries geachetet werden kann. Das ist nützlich um z.B.: einen Nutzer nur auf seine eigenen Graphen zugreifen zu lassen. Diese werden dann in die entsprechenden Queries umgewandelt.

II.5 Liste der Architekturentscheidungen

Autor: Alaa Aldin Karkoutli

JavaScript ist die grundlegende Programmiersprache, auf der diese App basiert ist. Sonst wurde es für die folgenden Architekturen entschieden:

I) Vue.js : wurde als das JavaScript-Webframework der Web-App eingesetzt.

II) Vuetify : ist die UI-Bibliothek, für die entschieden wurde.

III) CSS-Language-Extenion : wurde eingesetzt, um die optischen Dinge anzupassen.

IV) Vuex : ist eine 'Vue.js-Erweiterung', die als Speicher der Daten benutzt wird.

V) Cytoscape.js : ist die JavaScript-Bibliothek, für die entschieden wurde, um die Graphen zu erstellen.

VI) Cypress : ist ein Framework zum Testen der HTML-Elemente.

VII) jest : ist das Framework zum Testen der Richtigkeit einzelner Funktionen.

VIII) REST-Konforme (Representational State Transfer): ist die Architektur zur Kommunikation der DB und Authentifizierungsdaten mit der App.

IX) Node.js : ist die ausgewählte Schnittstelle, um auf die Daten der DB zuzugreifen.

X) Axios 'Promise-based' HTTP-Request : ist das eingesetzte Framework, um die Abfragen zu stellen.

XI) Docker : ist die Client-Server Architektur, für die entschieden wurde.

XII) MySQL : ist das DB-System, für das im Docker-Container entschieden wurde.

III. Prozess- und Implementationsvorgaben

III.1 Definition of Done

Autor: Tim Henning
Im Allgemeinen wurde in dem Projekt die Definition von "doneness" nicht all zu umfangreich gestaltet, da es für viele Teammitglieder eines der ersten Softwareprojekte war. So wurden als Defintion of Done folgende Punkte für alle Userstories aufgestellt:

- >50% Testabdeckung
- Technische Kommentare im Code
- Einhaltung der festgelegten Code Konventionen

Das Team hatte an sich zu den meisten Zeitpunkten eine klare Vorstellung was einen "fertigen Entwurf" kennzeichnet und wurde so auch in den Reviews untereinander kommuniziert. Dies wiederum führte zu einer klaren Transparenz im Team, was die Qualität des Produktes erhöhte und das Zusammenarbeiten erleichterte. Größtenteils wurde sich an die allgemeinen Akzeptanzkritieren gehalten und viele Backlog-Einträge als "done" erklärt. Zu fast jeder Komponente wurde getestet und zu den Methoden der einzelnen Komponenten wurden erklärende sinnvolle Codekommentare geschrieben. Außerdem wurde im Team umfangreich kommuniziert und die Kriterien angepasst, wenn die Fertigstellung einer Userstorie doch mal nicht gänzlich klar war. So wurde es ermöglicht nach der Hälfte des Projektes, am Ende jedes Sprints einen fertigen Productionbuild dem Kunden zu liefern.

III.2 Coding Style

Autor: Jonas Gwozdz

Beim Schreiben unseres Programmcodes haben wir uns an folgende Coding Conventions gehalten.

- Zeilenlänge: maximal 80 Zeichen
- Kommentare und Dokumentation
  - Kommentare auf Englisch
  - Klassen und Methoden in kurzen, prägnanten Sätzen beschreiben
  - Unnötige Kommentare vermeiden

  - Kommentare aktuell halten

- Einrückung und Zeilenumbrüche
  - 2 Leerzeichen statt Tabulator
  - ‚{ ' hinter Methodendeklaration
  - ‚} ' in neuer Zeile auf gleiche Einrückungsebene
  - Optionale Zeilenumbrüche für Übersichtlichkeit
  - Nur ein Import pro Zeile

- Leerzeichen
  - Vor und nach binären Operationen
    - Ausnahme nur im Fall von Verdeutlichung unterschiedlicher Prioritäten
  - Keine Leerzeichen vor und nach Klammern
  - Keine Leerzeichen vor Kommata und Semikolon
  - Leerzeichen nach Kommata
  - Keine Leerzeichen am Zeilenende

- Konsistentes Benennungsschema
  - Deskriptive Namen verwenden
  - mixedCase für Variablen
  - GROßSCHREIBUNG für Konstanten
  - Keine Umlaute
  - Reservierte Schlüsselwörter beachten
  - Immer auf Englisch
  - Bezeichner von Booleanwerte sollen Zustand beschreiben, der wahr oder falsch sein kann
  - Hilfsvariablen möglichst gleich benennen
  - Übergabe von Attributen an Konstruktoren
    - ‚length' als Attribut, ‚_length' als Argument

- Textcodierung UTF-8

Best Practice

- Allgemeines
  - Kein ‚language' Tag verwenden
  - Wiederholungen Vermeiden
  - Dopplungen vermeiden

- Variablen und Objekte
  - Keine globalen Variablen
  - Lokale Variablen, auch Zahlenvariablen zu Beginn deklarieren und initialisieren

  - Deklaration mehrerer Variablen können zusammengefasst werden
  - Datentyp wird über die Initialisierung zugewiesen
  - Kapselung mittels Namespace
  - Keine Deklaration mittels ‚new ... ()‘

- Funktionen
  - Vergleiche mittels ‚===‘
  - Unter keinen Umständen ‚eval()‘ benutzen
  - Keine ‚with‘ Statements
  - Keine ‚for (... in ...)‘ Loops
  - Jeder ‚switch‘ hate einen ‚default‘ Case
  - Vorsicht bei Verwendung von ‚typeof()‘
  - Nicht erhaltene Argumente gelten als ‚undefined‘

III.3 Zu nutzende Werkzeuge

Autor: Linus Herterich

Im Folgenden werden die Werkzeuge erwähnt, mit denen wir die Software entwickelt haben. Zudem wird darauf eingegangen, über welche Kanäle kommuniziert wurde.

III.3.1 Voraussetzungen

Das Versionsmanagement-Tool "GitLab" sowie das Zeitmanagement-Tool "YouTrack" wurden zu Beginn des Projekts vorgeschrieben. Die Commits in "GitLab" werden jeweils mit der ID des zugehörigen YouTrack-Tickets am Anfang des Commit-Titels versehen.
Damit das gesamte Team einheitliche Versionen der verwendeten Bibliotheken benutzt, wird der Paketmanager "npm" verwendet. Mir diesem lassen sich Pakete (und deren Versionen) definieren, welche für das Projekt benötigt werden.
Damit am Projekt gearbeitet werden kann, muss sich somit jedes Teammitglied die LTS- Version von Node.js (welches npm enthält) installieren.
Sobald Node.js global installiert ist, kann im "code" Verzeichnis der Befehl "npm install" ausgeführt werden, um die benötigten Bibliotheken zu installieren.

III.3.2 Compiler

Achtung: Das Kompilieren funktioniert erst, sobald die Bibliotheken mit dem Befehl "npm install" (im /code Verzeichnis) installiert wurden.
Um Änderungen des Projektes einzusehen, muss das Projekt kompiliert werden. "Vue.js" bringt bereits einen Echtzeit-Compiler mit, welcher reagiert, sobald Änderungen an Dateien im "code" Verzeichnis gemacht wurden. Um diesen Compiler aufzurufen, muss der npm-Befehl "npm run serve" im "code" Verzeichnis aufgerufen werden.
Um das Projekt nicht während der Entwicklung zu kompilieren, sondern für die Produktion freizugeben, muss der Befehl "npm run build" im "code" Verzeichnis aufgerufen werden. Es werden anschließend die kompilierten Dateien im Verzeichnis "code/dist" abgelegt. Diese können anschließend auf einem Webserver (z.B. Apache HTTP Server) hochgeladen werden.

III.3.3 Entwicklungsumgebung

Für die Entwicklung der Software wird der freie Quelltext-Editor "Visual Studio Code" von Microsoft verwendet. Dieser ist plattformunabhängig und kann durch zahlreiche Erweiterungen angepasst werden. Beispielsweise kann durch das Plugin "Vetur" die Vue.js-eigene Syntax vervollständigt und hervorgehoben werden.
Weitere Einstellungsvorgaben bezüglich der Entwicklungsumgebung wurden nicht getroffen. Es muss allerdings darauf geachtet werden, dass die Coding-Conventions durch automatische Formatierungen eingehalten werden.

III.3.4 CI / CD Pipeline

In der CI / CD Pipeline unseres Versionsmanagement-Tools, die nach jedem Git-Push ausgeführt wird, werden folgende Operationen durchgeführt:

- Test, ob das Projekt kompiliert (inklusive Syntaxprüfung durch ES-Lint)
- Cypress Tests durchführen (siehe "Überblick über Architektur")
- LaTeX Doku kompilieren

Sollte einer der Punkte fehlschlagen, wird der Autor des Git-Push's per E-Mail darüber informiert. Somit ist die Wahrscheinlichkeit, dass bestehende Features durch neue Entwicklungen längerfristig "zerstört" werden, möglichst gering.

III.3.5 Docker

Um die Client-Server Architektur des Projektes lokal zu simulieren, wird die Container-Virtualisierungssoftware "Docker" verwendet. Mit dieser haben wir einen Webserver simuliert, auf dem die Datenbank ausgeführt und verwaltet wird (siehe "Überblick über Architektur"). Die Container werden im Projekt-Ordner "docker" definiert.

III.3.6 Kommunikationstools

Zu Beginn des Projekts wurde sich auf das kostenlose Kommunikationstool "Slack" geeinigt. Mit diesem ist es möglich, in verschiedenen Kanälen Text, Dateien und Medien auszutauschen. Auch private Konversationen, sowie Kleingruppen-Chaträume sind in diesem Tool möglich. Die Software kann sowohl als App installiert, als auch im Browser verwendet werden.
Da wir über die Weihnachtsferien einen Sprint durchgeführt haben, führten wir Mitte Dezember das Tool "Discord" ein, mit dem es möglich ist, sich in Echtzeit-Sprachchats zusammenzufinden. Dazu ist es möglich, seinen Desktop zu teilen, womit sich das Tool bestens eignet, um räumlich getrennt über Code-Passagen oder neue Features zu sprechen.
Die Kombination beider Tools hat problemlos funktioniert und uns auch während des Lockdowns in der "Corona-Krise" geholfen. Da wir die Tools bereits frühzeitig eingesetzt haben, war kaum eine Um- bzw. Eingewöhnungszeit zu Beginn der präsenzfreien Zeit notwendig.

IV. Sprint 1

IV.1 Ziel des Sprints

Autor: Erik Heldt

Der erste Sprint des VarG-Projekts lief vom 05.12.2019 bis zum 16.12.2019. Ziel war es, eine fundamentale Struktur und grundlegende Funktionalitäten für die Anwendung zu entwickeln, auf denen man später weiter aufbauen kann. Währenddessen konnte man allgemeine Erfahrungen mit dem Ablauf eines Sprints machen.

IV.2 User-Stories des Sprint-Backlogs

Autor: Erik Heldt

Grundstruktur Die Anwendung sollte zu Beginn ein grundlegendes Fundament aufweisen, damit sich alle Teammitglieder vorstellen können, wie am Ende das Programm aussehen soll. Dazu gehörte zu Beginn das Design der Startseite mit dem VarGraph im Zentrum und der Einbindung von Cytoscape in die Programmstruktur.

Datenstruktur für Knoten Es sollte mit Hilfe von Cytoscape herausgefunden werden, wie man Knoten im Programmcode hinzufügen und speichern kann. Dafür sollte dann eine Datei im Programm angelegt werden.

Knoten zu bestehender Datenstruktur hinzufügen Die Anwendung sollte eine einfache Funktionalität zum Erstellen neuer Knoten aka Produktionsschritte erhalten, um sich mit den Cytoscape-Funktionen näher vertraut zu machen. Hier war erstmal noch keine graphische Darstellung in der GUI notwendig, es reichte per Console logs zu testen.

Darstellung eines Graphen in Weboberfläche In der Anwendung sollte zunächst ein statischer Graph mit Hilfe einer Cytoscape-Datenstruktur sichtbar dargestellt werden, damit man sehen konnte, wie so ein „CytoGraph“ überhaupt aussieht. User-Interaktion war hier noch nicht notwendig.

Kanten anlegen Zusätzlich zu Knoten sollten auch Kanten zwischen bestehenden Knoten hinzugefügt werden können. Diese Kanten sollten mit verschiedenen Attributen in der Cytoscape-Datenstruktur gespeichert werden.

Berechnung verschiedener Eigenschaften Anhand der mit den Kanten gespeicherten Attribute sollte eine Funktionalität entwickelt werden, welche die Gesamtkosten (Auswahl von Geld oder Zeit) aller unterschiedlichen Pfade berechnen und anzeigen sollte. Dies war der erste Schritt in Richtung Optimierung, d.h. später sollte diese Funktionalität automatisch den günstigsten Pfad herausfinden und anzeigen.

IV.3 Liste der durchgeführten Meetings

Autor: Erik Heldt

- Planning - 05.12.2019
- Weekly Scrum 1 - 09.12.2019
- Weekly Scrum 2 - 12.12.2019
- Review - 16.12.2019
- Retrospektive - 19.12.2019

IV.4 Ergebnisse des Planning-Meetings

Autor: Erik Heldt

Im Planning-Meeting erklärten die Projektmanager zu Beginn noch einmal kurz, wie ein Sprint im Allgemeinen abläuft und haben auf die Bedeutsamkeit der Coding Guidelines hingewiesen. Anschließend wurden die ersten User-Stories vom Project Owner vorgestellt und von den Bachelorstudenten per Finger-System in ihrer Komplexität eingeschätzt. Weiterhin wurde festgelegt, dass die Bachelorstudenten während des Sprints die User-Stories selbst in Tasks aufteilen und diese dann bearbeiten sollen.

IV.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: Erik Heldt

Arbeitspaket	Person	Start	Ende	h	Artefakt
Vue.js "Getting Started" Tutorial durcharbeiten (für alle)	Buchmann, Lennart	07.12.19	07.12.19	3	Tutorial abgeschlossen
Beispielgraph erstellen	Buxel, Nils	09.12.19	09.12.19	1	index.js
Kürzesten Weg mit A*-Algorithm berechnen u anzeigen lassen	Buxel, Nils	16.12.19	16.12.19	1	index.js
Funktionen zu Buttons hinzufügen	Gwozdz, Jonas	14.12.19	16.12.19	4	MenuControls.vue
Task: Einbindung in Vue-Dateistruktur	Heldt, Erik	15.12.19	15.12.19	3	MenuControls.vue, BasicData.js
Graphenanordnung	Heldt, Erik	05.12.19	05.12.19	3	Graphenanordnung.pdf
Vue.js "Getting Started" Tutorial durcharbeiten (für alle)	Heldt, Erik	11.12.19	11.12.19	2	Tutorial abgeschlossen
Funktionen zu Buttons hinzufügen	Henning, Tim	10.12.19	10.12.19	2	MenuControls.vue
Vue.js "Getting Started" Tutorial durcharbeiten (für alle)	Henning, Tim	06.12.19	06.12.19	3	Tutorial abgeschlossen
Einbindung von Cytoscape in Vue	Herterich, Linus	10.12.19	10.12.19	4	index.js
Buttons für Knoten und Kantenerstellung	Herterich, Linus	13.12.19	13.12.19	3	CreateControls.vue
Knoten zu Graph hinzufügen	Herterich, Linus	16.12.19	16.12.19	2,5	index.js, CreateControls.vue
Grundstruktur aufbauen	Herterich, Linus	05.12.19	07.12.19	9,5	Vue-Dateistruktur, sämtliche Startkom
Task: Basic Datenstruktur	Hohlfeld, Julius	15.12.19	15.12.19	8	BasicData.js, MenuControls.vue

IV.6 Konkrete Code-Qualität im Sprint

Autor: Erik Heldt

Zu Beginn wurde viel experimentiert und hauptsächlich sollte der Code erstmal ein funktionierendes Programm erzeugen, weswegen weniger auf die Qualität geachtet wurde. Trotzdem wurde sich größtenteils an die Coding Conventions gehalten und bereits einige Kommentare verfasst.

IV.7 Konkrete Test-Überdeckung im Sprint

Autor: Erik Heldt

Da der erste Sprint größtenteils nur zur Erstellung einer grundlegenden Datenstruktur und zur Einarbeitung in JavaScript und den genutzten Frameworks bzw. Bibliotheken gedient hat, gab es noch keine Tests.

IV.8 Ergebnisse des Reviews

Autor: Erik Heldt

Im ersten Review-Meeting stellten die Bachelorstudenten ihre Ergebnisse aus dem Sprint vor und die Manager gaben ihr Feedback dazu. Da sich die meisten Teammitglieder noch nicht richtig in Vue.js und Cytoscape einarbeiten konnten und teilweise große Schwierigkeiten mit den Frameworks hatten, gab es noch viele offene Aufgaben und nicht jeder hatte etwas vorzuzeigen.

Als erstes stellten Julius H. und Erik die Datenstruktur für die Knoten vor. Weiterhin zeigte Julius, wie ein Knoten in der Anwendung dargestellt wird und dass dieser durch ungeschickte Verschiebung und Skalierung aus der GUI verschwinden kann. Deshalb kamen Vorschläge, zukünftig den Zoom zu limitieren und das grundsätzliche Graph-Layout nochmal zu überarbeiten.

Um allen den Einstieg in die neuen Programmiersprachen und Bibliotheken etwas zu vereinfachen, stellte daraufhin Linus die Grundstruktur vor und erklärte noch einmal genau die einzelnen Elemente in der Dateistruktur. Weiterhin zeigte er, wie man ESLint-Fehler bei der Konsolenausgabe verhindern kann.

Danach wurde zwischen den Managern und den Bachelorstudenten noch die zukünftige Berechnung der kürzesten Wege und die unbearbeiteten User-Stories besprochen und dass diese in den nächsten Sprint mit einfließen werden.

Zum Schluss wurden noch ein paar allgemeine Fragen zum Testen und zu Git geklärt.

IV.9 Ergebnisse der Retrospektive

Autor: Erik Heldt

In der Retrospektive konnte jedes Teammitglied vor an die Tafel gehen und verschiedene Aspekte des Sprints mit einem Strich in einer Tabelle bewerten.

Die Bewertung ging ausgeglichen aus. Die Gruppenleistung und das Gesamtergebnis waren gut, aber die Einzelleistungen der meisten Teammitglieder nicht. Viele Aufgaben blieben offen und wurden nicht erledigt, wozu in der Diskussion verschiedene Gründe angeführt wurden. Einerseits war es für die meisten schwer, sich selbst in die neue Programmierumgebung samt den Frameworks und Bibliotheken einzuarbeiten. Andererseits wussten viele nicht, was und wie viel sie machen sollten, was auf die nicht festgelegte Aufgabenzuteilung im Planning und die schlechte Kommunikation im Team während des Sprints zurückgeführt wurde. Letzteres Problem plante man damit zu lösen, in zukünftigen Plannings immer direkt Verantwortliche für bestimmte User-Stories festzulegen und entsprechende Tickets sofort im Anschluss zu erstellen und zuzuweisen.

Beim Thema der Daily Meetings ist man zu dem Schluss gekommen, dass diese wenn möglich immer persönlich bleiben sollten und nur in Ausnahmefällen online z.B. über Discord stattfinden sollten. Weiterhin wurde diskutiert, ob die Zeitspanne zwischen Donnerstag und Montag evtl. zu kurz ist, um schon weitreichende Ergebnisse zu erzielen, da am Wochenende einige Teammitglieder nicht programmieren können. Deshalb sollten die ersten Meetings beim nächsten Sprint stattdessen Montag und Donnerstag stattfinden.

Ein weiterer Themenpunkt war die Organisation im Git. Es wurde festgelegt, dass der Master-Branch während des Sprints unberührt bleiben sollte, da dieser immer lauffähig sein muss. Stattdessen sollte sich jeder seinen eigenen Branch erstellen und diesen nach Abschluss der eigenen

Aufgaben auf den neuen Developer-Branch namens "targetbranch" mergen. Am Ende jedes Sprints würde dann der Developer-Branch mit dem Master-Branch gemerged werden.

IV.10 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

Der Erste Sprint wurde zur Erstellung einer geeigneten Datenstruktur und einfinden in das Projekt genutzt. Während des Sprints beziehungsweise am Ende des Sprints wurde festgestellt, dass sich viele beim Planning-Poker überschätzt haben. Somit sind einige User-Stories offen geblieben. Die Daily Treffen gaben einen guten Überblick, wer an was arbeitete. Festzuhalten war, dass Aufgaben klarer verteilt werden müssen, dass sich die einzelnen Teammitglied für die User-Stories verantwortlich fühlen.

IV.11 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

IV.12 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Ein verhaltener Start in das Softwareprojekt von sowohl Bachelors als auch uns Mastern. Natürlich muss sich dieses Team-Konstrukt erst einmal einspielen. Bleibt abzuwarten, wie sich die in der Retrospektive ergriffenen Maßnahmen auswirken.

V. Sprint 2

V.1 Ziel des Sprints

Autor: Linus Herterich

Nachdem im ersten Sprint hauptsächlich die Grundstruktur sowie erste Datenstrukturen entworfen wurden, war es nun wichtig, dass sich das gesamte Team im Sprint 2 mit der Projektstruktur (besonders mit dem Framework Vue) auseinandersetzt und erste UserStories direkt am Code umsetzt. Zudem blieben einige Tickets noch vom letzten Sprint offen, welche nun auch bearbeitet werden sollten.

V.2 User-Stories des Sprint-Backlogs

Autor: Linus Herterich

- **Designumsetzung nach Adobe Preview**
  Als Benutzer der WebApplikation möchte ich ein ansehnliche und intuitive Oberflächengestaltung haben, damit ich die Applikation gerne verwende.

- **Authentifizierung eines Nutzers**
  Als Nutzer möchte ich mich in die Web Applikation einloggen können, damit nicht jeder meine erzeugten Graphen einsehen kann.

- **Logische verknüpfung zwischen Knoten erstellen**
  (wurde in Sprint 1 nicht abgeschlossen)
  Ein Nutzer muss eine Abfolge der Knoten definieren können, damit ersichtlich wird welcher Produktionsschritt auf den nächsten folgt

- **Berechnung der Eingenschaften des Gesamtgraphs**
  (wurde in Sprint 1 nicht abgeschlossen)
  Ein Nutzer der Webanwendung VarG muss die berechneten gesamt Eigenschaften jedes Zusammenhängendes Pfades ausgeben lassen können um eine Auswahl eines Pfades zu treffen.

- **Datenstruktur Ausarbeiten & Knoten zu einer vorhandenen Datenstruktur hinzufügen**
  (wurde in Sprint 1 nicht abgeschlossen)
  Als Nutzer möchte ich Knoten zu der Datenstruktur hinzufügen können um die möglichen Produktionsschritte des Werkstücks überblicken zu können

V.3 Liste der durchgeführten Meetings

Autor: Linus Herterich

- 19.12.2019: Planning Meeting
- 23.12.2019: Daily Meeting (in Discord)
- 28.12.2019: Daily Meeting (in Discord)
- 05.01.2020: Review Meeting
- 06.01.2020: Retrospektive

V.4 Ergebnisse des Planning-Meetings

Autor: Linus Herterich

Neben der Aufgabenverteilung wurde im Planning darüber gesprochen, dass die Arbeitsaufteilung im letzten Sprint nicht gut geklappt hat. Es wurde anschließend beschlossen im nächsten Sprint die User-Stories direkt an Studenten zuzuweisen, damit jeder einen Teilbereich hat, den er bearbeiten muss.
Desweiteren wurde eine Änderung im Git angekündigt. In Zukunft müsse der "Master"-Branch während eines Sprints immer gleich bleiben und Funktionalitäten werden auf einen "Developer"-Branch gemerged. Am Ende des Sprints wird dann der "Developer"-Branch auf den "Master"-Branch gemerged. wichtig ist, dass der "Master"-Branch zu jedem Zeitpunkt lauffähig ist.
Für den folgenden Sprint wurde beschlossen, die Daily Meetings online (auf einem Discord Server) abzuhalten, da viele Studenten über die Weihnachtsferien in der Heimat sind und somit ein persönliches wöchentliches treffen nicht möglich wäre.

V.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: Linus Herterich

Arbeitspaket	Person	Start	Ende	h	Artefakt
UI: Login	Berger, Matthias	22.12.19	22.12.19	3,5	Login Funktionalität & Design
UI: Login	Buchmann, Lennart	22.12.19	22.12.19	6	Login Funktionalität & Design
UI: Grapheneditor	Gwozdz, Jonas	23.12.19	04.01.20	9	GraphHeader.vue, Toolbar.vue
Task: Einbindung in Vue-Dateistruktur	Heldt, Erik	19.12.19	19.12.19	0,25	BasicData.js
Abrufbaren Knoten in Graph einfügen	Heldt, Erik	23.12.19	26.12.19	3,5	BasicData.js, TestDatabase.js
Testdatenbank mit Speichern und Laden	Heldt, Erik	27.12.19	27.12.19	3,5	TestDatabase.js
Highlighting eines kürzesten Pfades nach Anwendung des A* Algorithmus	Henning, Tim	24.12.19	03.01.20	9	OptimizeControls.vue, index.js -> Graph Highlighting
Protokoll: Meeting 19.12.19	Herterich, Linus	19.12.19	19.12.19	1	meeting_19_12_19.pdf
UI: Login	Herterich, Linus	20.12.19	20.12.19	5	LoginForm.vue, Login.vue
UI: Home	Herterich, Linus	23.12.19	23.12.19	7	HomeMenu.vue (component), Home.vue (view), Menu.vue (view)
UI: Neuer Graph	Herterich, Linus	28.12.19	28.12.19	1,5	NewGraph.vue (view), NewGraph.vue (component)
UI: Grapheneditor	Herterich, Linus	02.01.20	04.01.20	11,75	Graph.vue (view), zahlreiche components
Graph zu Datenstruktur hinzufügen	Hohlfeld, Julius	21.12.19	23.12.19	4	BasicData.js, TestDatabase.js

Testdatenbank mit Speichern und Laden	Hohlfeld, Julius	27.12.19	03.01.20	8	BasicData.js, TestDatabase.js, index.js, JSonPersistence.js
Mergen und Anpassen	Hohlfeld, Julius	04.01.20	04.01.20	2	Bugs entfernt & Mergekonflikte behoben
UI: Datenbank-Import Fenster	Karkoutli, Alaa Aldin	31.01.20	04.01.20	12,5	Database.vue (view), DatabaseForm.vue (component)
Kanten zu Graph hinzufügen	Koch, David	23.12.20	04.01.20	10	Änderungen an index.js, CreateControls.vue (component)

V.6 Konkrete Code-Qualität im Sprint

Autor: Linus Herterich

Es wurde sich größtenteils an die Coding-Guidelines gehalten. An wichtigen Stellen sowie vor jeder Funktion wurden Kommentare geschrieben. Die Trennung zwischen Views und Components sowie die Auslagerung der Style-Dateien wurde ebenfalls eingehalten.

V.7 Konkrete Test-Überdeckung im Sprint

Autor: Linus Herterich

Ein Student wurde beauftragt bis zum Ende des Sprints ein geeignetes Test-Framework zu finden. Somit wurden während des Sprints noch keine Tests geschrieben.

V.8 Ergebnisse des Reviews

Autor: Linus Herterich

Es wurden fast alle UserStories umgesetzt. Somit war der zweite Sprint erfolgreich. Alle Studenten konnten sich in das Projekt einarbeiten und haben die Strukturierung größtenteils verstanden und eingehalten.
Das User-Interface wurde nach der Designvorlage umgesetzt und die ersten Graphen-Funktionen (Hinzufügen von Knoten und Kanten & Optimieren) funktionieren bereits.
Da noch nicht feststeht, wo die Software gehostet werden soll und wie die Datenbank-Funktionalität umgesetzt werden soll, wurde zunächst eine lokale Speicherlösung als "Datenbank"verwendet. Somit konnten die Speichern- und Laden-Funktionen erfolgreich implementiert werden.
Die Login-Funktionalität ist derzeit nur sporadisch eingerichtet und wird finalisiert, sobald feststeht, wie die Authentifizierung der Nutzer erfolgen soll (Anbindung an HTWK Login?).
Leider ist immernoch kein geeignetes Testframework gefunden worden, mit dem sich sowohl Vue.js als auch cytoscape (Graphen-Funktionalitäten) testen lassen.

V.9 Ergebnisse der Retrospektive

Autor: Linus Herterich

Das Happiness-Barometer für diesen Sprint ist sehr gut ausgefallen. Das liegt hauptsächlich an der guten Aufgabenverteilung sowie an den großen Erfolgen, die diesen Sprint erzielt wurden.
Kritisiert wurde die die Kommunikation gegen Ende des Sprints. Das finale Mergen aller Branches war zu hektisch und unsicher.
Es wurde sich darauf geeinigt in Zukunft zwei Dailies pro Woche abzuhalten und das letzte Meeting eines Sprints zum gemeinsamen Mergen zu verwenden.

V.10 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

Die Entwickler wurden grob in 3 Teams eingeteilt. Dies setzen sich aus Login , UI-Design und Graph-Funktionalitäten zusammen. Somit konnte jede User-Story konkret einem Subteam zugeortnet werden. Damit hat sich die Anzahl der Erfolgreich abgeschlossen Aufgaben wesentlich erhöht. Leider ist die Kommentar und vor allem die Testabdeckung noch nicht zufiedenstellend.

V.11 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

V.12 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Deutliche Leistungssteigerung schon jetzt zu sehen. Aufteilung der User-Stories direkt nach dem Planning hat die Arbeitsstruktur und -ablauf während des Sprints auf jeden Fall positiv beeinflusst.

VI. Sprint 3

VI.1 Ziel des Sprints

Autor: Lennart Buchmann

Nach der Einarbeitung des gesamten Teams in die Grundstruktur der Software, sowie der Frameworks, lag das Hauptaugenmerk des dritten Sprints in der verstärkten Herausarbeitung der geplanten Kernfunktionalitäten der Anwendung. Größere Aufgabenbereiche wurden durch Zweier- und Dreierteams gelöst. Übriggebliebenes aus den vorherigen Sprints sollte beendet werden

VI.2 User-Stories des Sprint-Backlogs

Autor: Lennart Buchmann

- **Funktionalität der Datenbank**
  Als Nutzer will ich meine gespeicherten Graphen ansehen können, um diese weiter bearbeiten zu können.

- **Kontext Menu über rechte Maustaste**
  Als Nutzer möchte ich Knoten und Kanten über einen Rechtsklick zur einfacheren Benutzung erstellen können.

- **Authentifizierung eines Nutzers**
  Als Nutzer möchte ich mich in die Web Applikation einloggen können, damit nicht jeder meine erzeugten Graphen einsehen kann.

- **Darstellung von Knoten und Kanteneigenschaften am Objekt**
  Als Benutzer möchte ich über einen Rechtsklick auf einen Knoten/Kante die Eigenschaften dieser bearbeiten können.

- **Optimierung des Graphs**
  Als Benutzer möchte ich gerne sofort sehen können, wie hoch meine Kosten für den kürzesten Pfad sind, damit ich mich möglichst schnell für einen entscheiden kann.

- **Speicherung Graph**
  Als Nutzer möchte ich einen Graphen jederzeit bearbeiten und speichern können, auch wenn dieser noch unfertig ist.

VI.3 Liste der durchgeführten Meetings

Autor: Lennart Buchmann

- 06.01.2020: Planning Meeting
- 09.01.2020: Weekly Scrum
- 13.01.2020: Weekly Scrum
- 16.01.2020: Weekly Scrum
- 20.01.2020: Review & Retrospektive Meeting

VI.4 Ergebnisse des Planning-Meetings

Autor: Lennart Buchmann

Der 3. Sprint ist der letzte Sprint im laufenden Semester und der letzte Sprint vor den anstehenden Prüfungen. Während des Planning-Meetings wurde von allen einheitlich besprochen, dass die Arbeitslast von jedem höher ist als während der vergangen Sprints. Es wurde sich daraufhin geeinigt lieber realistische Ziele zu setzen, sodass der 3. Sprint auch mit höhere Belastung erfolgreich abgeschlossen werden kann. Nach Besprechung und Schätzung der Tickets, wurden alle Aufgaben in kleinere Gruppen aufgeteilt. Größere Aufgaben, die nach Schätzung im aktuellen Sprint nicht umsetzbar wären, wurden auf den verlängerten 4. Sprint verschoben.

VI.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: Lennart Buchmann

Arbeitspaket	Person	Start	Ende	h	Artefakt
Login	Berger, Matthias	13.01.20	17.01.20	18	Login Funktionalität & Design
Login	Buchmann, Lennart	18.01.20	18.01.20	6	Login Funktionalität & Design
Knotendarstellung nach Designvorlage	Gwozdz, Jonas	15.01.20	20.01.20	4,5	GraphHeader.vue, Toolbar.vue
Speicherung Graph	Heldt, Erik	06.01.20	19.01.20	18	BasicData.js
Optimierung des Graphs	Henning, Tim	09.01.20	18.01.20	9	OptimizeControls.vue, index.js -> Graph Highlighting
Speicherung Graph	Herterich, Linus	07.01.20	19.01.20	24,25	meeting_19_12_19.pdf
Graph zu Datenstruktur hinzufügen	Hohlfeld, Julius	07.01.20	20.01.20	19	BasicData.js, TestDatabase.js
– Funktionalität Neuer Graph Button	Karkoutli, Alaa Aldin	12.01.20	15.01.20	7	Database.vue (view), DatabaseForm.vue (component)
Kanten zu Graph hinzufügen	Koch, David	17.01.20	19.01.20	10	Änderungen an index.js, CreateControls.vue (component)

VI.6 Konkrete Code-Qualität im Sprint

Autor: Lennart Buchmann

VI.7 Konkrete Test-Überdeckung im Sprint

Autor: Lennart Buchmann

Eine konkrete Auseinandersetzung mit Tests beziehungsweise entsprechenden Test-Frameworks fand während des 2. Sprints statt. Momentan befinden sich alle Teammitglieder noch in der Einarbeitungsphase. Aufgrund des fortgeschrittenes Semesters und der anstehenden Prüfungen lagen die Prioritäten vorwiegend auf der Bearbeitung der User-Stories.

VI.8 Ergebnisse des Reviews

Autor: Lennart Buchmann

Das Ergebnis der Reviews war in anbetracht der fortgeschrittenen Semesters durchgehenden positiv. Alle Teammitglieder haben die Ihnen zugewiesenen Aufgaben innerhalb des Sprints erledigt. Es wurde des Weiteren besprochen, dass der verlängerte Sprint während der Semesterferien dazu genutzt werden sollte, um Bugs zu beheben und somit jedem die Gelegenheit zu geben, sich in die Testframeworks einzuarbeiten und Tests für den geschriebenen Code zu verfassen.

VI.9 Ergebnisse der Retrospektive

Autor: Lennart Buchmann

Während der Retrospektive wurde von allen die grundsätzliche gute Kommunikation innerhalb des Teams gelobt. Alle empfanden auch die Aufteilung in kleinere Zweier- und Dreierteams zur Bearbeitung von Aufgaben für sehr hilfreich. Eine gleichbleibende hohe Motivation und Produktivität soll auch während des Semesterferiensprints beibehalten werden. Punkte, welche verbessert werden sollten, sind das pünktliche Mergen der einzelnen Branches vor Ende des Sprints, das Kommentieren des Codes und das Verfassen von Tests.

VI.10 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

In diesem Sprint wurden weitere Kernfunktionalitäten bearbeitet. Das Verständnis im Planning-Meeting ist sichtbar größer im Vergleich zum vorhergehenden Sprint. Durch diesen Umstand wurden fast alle angestrebten User-Stories erfolgreich bearbeitet. Leider lässt die Testqualität noch zu wünschen übrig.

VI.11 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

VI.12 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Weiterhin aufstrebende Arbeit vom Team. Auch die Kommunikation bei Problemen, Fragen und Anregungen geht in eine positive Richtung.

VII. Sprint 4

VII.1 Ziel des Sprints

Autor: Jonas Gwozdz

Während der Semesterferien haben wir an Sprint 4 weitergearbeitet. Dieser dauerte vom 23.01.2020 bis zum 09.04.2020. Der Ablauf war dabei weitestgehend planmäßig, bis auf dass die Meetings zum Review und der Retrospektive wegen Corona ohne persönliches Treffen stattfinden mussten. In der Vorlesungsfreien Zeit besprachen wir uns gelegentlich über den aktuellen Zwischenstand. Der größte Fortschritt am Projekt wurde während der letzten beiden Wochen erzielt.

VII.2 User-Stories des Sprint-Backlogs

Autor: Jonas Gwozdz

- **Tests für bereits geschriebenen Code**
  Als Benutzer möchte ich eine Software benutzen, die getestet ist, damit keine unerwarteten Probleme auftauchen.

- **Validierung der möglichen Eingaben**
  Als Nutzer möchte ich bei versehentlicher falscher Eingabe wenn möglich gewarnt werden, damit ich nichts falsches abspeichere.

- **Bug: Validation bei gleichem Knoten-Namen**

- **Darstellung von Kanten/Attributen**
  Als Benutzer will ich alle Kanten/Knoten gleichzeitig sehen können(nicht übereinander), damit ich einen schnelleren Überblick über das gesamte Konstrukt bekomme.

- **Bug: Mehrere Edges zwischen Knoten nicht möglich**
  Wenn man mehrere Kanten zwischen zwei Knoten anlegt, sind diese nicht sichtbar. Löscht man dann einen Knoten, an dem diese ünsichtbaren"knoten hängen, so stürzt cytoscape ab.

- **Remodel von Component NewGraph**

VII.3 Liste der durchgeführten Meetings

Autor: Jonas Gwozdz

- 23.01.2020: Planning
- 05.03.2020: Weekly
- 12.03.2020: Weekly
- 06.04.2020: Review
- 09.04.2020: Retro

VII.4 Ergebnisse des Planning-Meetings

Autor: Jonas Gwozdz

Anwesend: Alex, Julius J., Julius H., Linus, Jonas, Erik, Lennart, Nils, Tim, David, Matthias, Manuel

Innerhalb dieses Meetings haben wir die Schwerpunkte des Sprints festgelegt und über den Workload über die Vorlesungsfreie Zeit diskutiert und den Zeitaufwand der User-Stories abgeschätzt.

oberste Priorität: Tests

Da wird bis zum bisherigen Zeitpunkt keine Testumgebung gefunden haben, die sich auf unseren Cytoscape-Graphen anwenden lässt, und wir dadurch viel Nachholbedarf in Sachen Testen hatten, musste dieses Ticket am dringendsten abgearbeitet werden.

Sprint über Semesterferien

Wir haben uns im Planning darauf geeinigt, den Sprint über die Semesterferien mit weniger User-Stories als üblich auszulegen, da nicht alle Teammitglieder in dieser Zeit voll verfügbar waren, Grund dafür waren vor Allem die noch andauernden Prüfungen und die Anschließenden Ferien, die evtl. schon anderweitig verplant waren. Zudem haben wir uns darauf geeinigt, regelmäßig Absprache über den Fortschritt unserer Arbeit zu halten.

Datenbanken

Die Datenbankrecherche hat ergeben, dass für unsere Zwecke mySQL oder NodeJS am optimalsten wäre. Die Definition der Datenbankschnittstelle zwischen DB und Frontend muss ebenfalls noch erledigt werden. Zudem haben wir festgestellt, dass die Bisher entworfene Datenbankoberfläche optisch nicht zum Rest der Anwendung passt, und deshalb überarbeitet werden muss.

Weitere Sprintziele:

- Optimierung der Kostendarstellung
- negative Zahleingaben abfangen
- automatische Zoomfunktion bei Knoten- oder Kantenwahl
- allgemeine Bugfixes

VII.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: Jonas Gwozdz

Arbeitspaket	Person	Start	Ende	h	Artefakt
Tests für bereits geschriebenen Code	Heldt, Erik	04.03.20	04.03.20	2	Tests für ModifyDataControls.vue
Neue Strukturierung	Heldt, Erik	26.01.20	26.01.20	1	Umstrukturierung des Projekts
Header Buttons und Metadaten-Speicherung	Heldt, Erik	05.03.20	12.03.20	6,75	GraphHeader.vue
Aufräumen der Branches im GitLab	Heldt, Erik	29.03.20	29.03.20	1	Organisatorische Aufgabe
Entfernen veralteter Komponenten und Methoden	Heldt, Erik	31.03.20	31.03.20	2	Organisatorische Aufgabe

Tests für Graphoptimierung	Henning, Tim	04.04.20	40.40.20	12	vargraph.spec.js
Tests für bereits geschriebenen Code	Herterich, Linus	30.01.20	12.02.20	7,5	/code/cypress/integration/...
Header Buttons und Metadaten-Speicherung	Herterich, Linus	28.03.20	31.03.20	2,25	/vargraph/graph/... & GraphHeader.vue
Aufräumen der Branches im GitLab	Herterich, Linus	30.03.20	30.03.20	1	Organisatorische Aufgabe
Darstellung von Kanten/Attributen	Herterich, Linus	03.04.20	03.04.20	2	VarGraph.vue
Remodel von Component NewGraph	Herterich, Linus	30.03.20	30.03.20	3	/vargraph/graph/...
Refactoring	Herterich, Linus	29.03.20	30.03.20	9	/vargraph/graph/...
Validierung: Login	Herterich, Linus	31.03.20	30.03.20	1,5	/components/login/LoginForm
Einheitliche Alerts	Herterich, Linus	31.03.20	31.03.20	3	Dialogs.vue
Validierung CreateControls & DetailControls	Herterich, Linus	31.03.20	01.04.20	5,5	CreateControls.vue & DetailControls.vue
Bug: Mehrere Edges zwischen Knoten nicht möglich	Herterich, Linus	01.04.20	01.04.20	2	/vargraph/graph/...
Knoten dort erstellen, wo rechtsklick passiert	Herterich, Linus	01.04.20	01.04.20	1,5	/vargraph/graph/...
keybinds für Menüs	Herterich, Linus	02.04.20	02.04.20	1	
Keine Knoten aufeinander schieben	Herterich, Linus	02.04.20	02.04.20	3	/vargraph/graph/...
Einstellungsmenü erstellen	Herterich, Linus	03.40.20	05.04.20	5,5	
Tests für bereits geschriebenen Code	Hohlfeld, Julius	05.02.20	04.03.20	10	ZoomControls.spec & SaveMenu.spec & NewGraphMenu.spec & DownloadMenu.spec
Dialogfenster für Speichern, Laden und Export	Hohlfeld, Julius	24.01.20	24.01.20	2	Toolbar.vue
Validierung der möglichen Eingaben	Hohlfeld, Julius	06.04.20	06.04.20	2	divers
Refactoring	Hohlfeld, Julius	31.03.20	31.03.20	2	/vargraph/graph/...
Testing für Kanten hinzufügen	Koch, David	22.03.20	02.04.20	5	addEdges.spec

VII.6 Konkrete Code-Qualität im Sprint

Autor: Jonas Gwozdz

Die Codequatlität im allgemeinen wurde während des Sprints erheblich durch das Refactoring verbessert. Zudem wurden in nahezu allen Dateien einleitende Kommentare geschrieben, um die

zukünftige Identifizierung der gebrauchten Dateien schneller und übersichtlicher zu gestalten.

VII.7 Konkrete Test-Überdeckung im Sprint

Autor: Jonas Gwozdz

Die geschriebenen Cypress-Tests decken bereits eine Vielzahl an Funktionalitäten des Programms ab. Dazu zählen die Buttons für die Database, den Download, das Ausloggen. Zudem wurde getestet: der Speicherdialog, die Zoomeinstellungen, der Header des Graphen, das Hinzufügen von Knoten und das Erstellen eines neuen Graphen.

VII.8 Ergebnisse des Reviews

Autor: Jonas Gwozdz

Anwesend: David, Erik, Julius J., Julius H., Jonas, Linus, Manuel, Matthias, Tim

Im Rahmen des Reviews haben wir wie gewohnt die Ergebnisse des Sprint bewertet und Schwierigkeiten besprochen.

generelle Schwierigkeit: Testen
Um unsere Programm zu testen, entschieden wir uns für das Framework "Cypressëntschieden. dieses bietet End-to-End Testing an, welches allerdings nur Ausgaben des Programms auswerten kann, und deshalb sozusagen keinen Blick unter die Haube zulässt, und somit eventuell Fehler unentdeckt bleiben.

David:

- Tests für Knotenfunktionalität geschrieben
- mit Kantentests begonnen

Erik:

- Data Controls durch Header Buttons ersetzt
- Editierungsfenster entfernt
- Header Buttons getestet

Jonas:

- Testübersicht erstellt
- Möglichkeit zum Informationsaustausch über Lücken und Bugs in Tests bereitgestellt

Julius H.:

- Tests für Toolbar, Zoom-Controls, Buttons und Eingabereihenfolgen geschrieben

Julius H, Erik, Linus:

- Refactoring des Graphen, Bugfixing und Validierung von Eingaben

Linus:

- Dialogue-Popups erstellt
- Kürzelgenerierung implementiert

- Knotenüberlagerung unterbunden, Mindestabstand implementiert
- Einstellungsmenü erstellt und Implementation begonnen
- Recherche zu Datenbankfenster

VII.9 Ergebnisse der Retrospektive

Autor: Jonas Gwozdz

Anwesend: Alex, Erik, Julius J., Julius H., Jonas, Linus, Matthias, Tim

Zu Beginn des Sprints gab es keine Fortschritte zu vermelden, da vorerst die Prüfungen zu überstehen waren. In den beiden Wochen vor Sprintende wurden allerdings die wichtigsten User-Stories und sogar etwas mehr abgearbeitet.

Positiv	Negativ
-produktive Endphase -viel Motivation bei Einigen	-anfangs keine Kommunikation - wenig Motivation bei Einigen -vereinzelt Tests ohne Sinn -ausgefallene Meetings

VII.10 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

Ein großer Teil dieses Sprints musste dazu genutzt werden, Tests für die vergangenen Sprints zu schreiben, dass diese User-Stories abgeschlossen werden konnten. Da dieser Sprint über die Prüfungszeit und Semesterferien ging wurden zusätzliche User-Stories bearbeitet.
Bis zu diesem Zeitpunkt wurden fast ausschließlich Komponenten des Frontends entwickelt. Daher wurden in diesem Sprint Aufgaben zur Entwicklung eines passenden Backends und damit verbundener Datenbank bearbeitet.
Während des Reviews wurde festgestellt, dass nur ein begrenztes Verständnis für Test vorhanden ist und es leider immer noch nicht die in der DoD festgehaltene Testüberdeckung erfüllt ist.

VII.11 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

VII.12 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Ein aufgrund der Prüfungsphase - wie zu erwarten - zunächst sehr verhaltener Sprint. Auch danach passierte aufgrund von Arbeit, Urlaub und sonstigen Ferienaktivitäten verständlicherweise noch relativ wenig im Vergleich zu den vergangenen beiden Sprints. Ergebnis durch engagierte Arbeit in den letzten Tagen vor Sprintende trotzdem gut.

VIII. Sprint 5

VIII.1 Ziel des Sprints

Autor: Tim Henning

Der vierte Sprint des VarG-Projektes lief vom 13.04.20 bis zum 23.04.20. Ziel des Sprints war zum einem, dass die nicht vollendeten Aufgaben aus Sprint 4 nachgeholt werden, und das sich um die Schnittstelle zwischen Frontend und Backend gekümmert wird. Weiterhin wurde geäußert viel Recherche zum Thema Datenbanken, Shibboleth Anbindung und bereitstellen eines Servers des IT-Servicezentrums der HTWK, zu betreiben. Außerdem sollten zum vorhandenen Optimierungsalgorithmus noch einige Besserungen vorgenommen werden.

VIII.2 User-Stories des Sprint-Backlogs

Autor: Tim Henning

Datenbank, Initiale Aufgaben zur Bereitstellung Als Nutzer möchte ich gerne auf eine, mit dem Rest der App, konsistente Oberfläche zugreifen können, damit ich mich einfacher zurecht finde. Zudem möchte ich gerne einen Überblick über die vorhandenen Elemente (Bearbeitungsmaschinen) anzeigen lassen und in meinen Graphen übernehmen können, damit ich die Eigenschaften dieses Elements nicht jedes mal neu heraussuchen muss.

Entwurf der Schnittstelle zwischen Backend und Frontend Als Nutzer möchte ich Daten aus der Datenbank abrufen/anzeigen lassen können, damit der Graph schneller erstellt werden kann.

Login Nach dem Login, in die Applikation sollen meine Anmeldedaten gespeichert werden, damit ich mich beim erneuten laden der Seite nicht neu einloggen muss.

Optimierung Als Benutzer möchte ich optimale Wege des erstellten Graphen anzeigen lassen können, damit ich eine bessere Auswahl zwischen den einzelnen Bearbeitungsschritten treffen kann.

VIII.3 Liste der durchgeführten Meetings

Autor: Tim Henning

- Planning - 13.04.2020
- Weekly Scrum 1 - 16.04.2020
- Weekly Scrum 2 - 20.04.2020
- Review - 23.04.2020
- Retrospektive - 23.04.2020

VIII.4 Ergebnisse des Planning-Meetings

Autor: Tim Henning

Anwesend: Jonas G., Erik H. Linus H., Lennart B., Tim H., David K., Matthias B., Alaa Aldin K., Manuel E., Julius J., Alex H.

Neben der Aufgabenverteilung wurden noch einige zusätzliche Punkte besprochen, die nicht in den User Stories aufgetaucht sind. So zum Beispiel sollte nach jedem Sprint ein Production Build angelegt werden, der auf einem Server liegt, damit der Kunde regelmäßig das Produkt testen kann. Weiterhin wurde gefordert die neue Testumgebung Cypress in die Git-Pipeline einzubinden. Außerdem sollte am IT-Servicezentrum nachgefragt werden, ob es möglich ist eine Shibboleth Anbindung

zu bekommen und ob die HTWK einen Server bereitstelle, auf dem der Production Build später gehostet werden kann.

VIII.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: Tim Henning

Arbeitspaket	Person	Start	Ende	h	Artefakt
UI: Login	Beger, Matthias	13.04.20	13.04.20	2,5	Recherche, Konzeption
UI:Login	Buchmann, Lennart	23.04.20	23.04.20	5	Recherche, Konzeption
UI: Datenbank; Initiale Aufgaben zur Bereitstellung	Gwozdz, Jonas	13.04.20	23.04.20	15	Datenbankfenster Redesign, Responsiveness der Datenbankseite, Button Platzierungen
Task: Sprint 4 Dokumentation	Gwozdz, Jonas	13.04.20	13.04.20	5	Sprint4.tex
UI: Entwurf der SChnittstelle Backend <-> Frontend	Heldt, Erik	18.04.20	18.04.20	1,5	SaveMenu.vue, TestDataBase.js
Task: Recherche Zusammenspiel Vue + Datenbank	Heldt, Erik	15.04.20	16.04.20	2	Installation Axios, HTTP Requests
Task: Button UI/UX Änderungen und Validierung bei Erstellung von Kanten	Heldt, Erik	17.04.20	23.04.20	7	CreateControl.vue, DetailControls.vue
Task: Gesamkosten und /-zeit einschließlich der Produktanzahl	Henning, Tim	15.04.20	16.04.20	4	optimization.js
Task: Alten Optimierungsalgorithmus umbauen	Henning, Tim	17.04.20	23.04.20	11	optimization.js
Task: Sprint 5 Dokumentation	Henning,Tim	23.04.20	23.04.20	3	Sprint5.tex
UI: Datenbank; Initiale Aufgaben zur Bereitstellung	Herterich, Linus	15.04.20	15.04.20	2,5	Datenbankseite nun als Component
Sprint 2 Dokumentation	Herterich, Linus	13.04.20	13.04.20	3,5	Sprint2.tex
Task: Erstellung Production Build auf Server	Herterich, Linus	14.04.20	14.04.20	3	läuft auf varg.nfl-server.de
Task: Cypress Test in die Gitlab Pipeline	Herterich, Linus	16.04.20	16.04.20	4,5	.gitlab-ci.yml
Task: Kaputte Tests reparieren	Herterich, Linus	17.04.20	17.04.20	2	code/cypress/integration/..
Task:Graph aus Hauptmenü importieren	Herterich, Linus	17.04.20	17.04.20	2	Importieren aus Hauptmenü umgesetzt
Task: Redesign Graphen Seite(Navigation Drawer)	Herterich, Linus	16.04.20	23.04.20	11	2 neue Designkonzepte

UI: Entwurf der Schnittstelle Backend <-> Frontend	Hohlfeld, Julius	13.04.20	22.04.20	6,5	Dokumentation der API-Recherche und erste Entwürfe, API Dokumentation im Git Wiki
Task: Auswahl von Endzustand ohne Startzustand	Karkoutli, Alaa Aldin	17.04.20	22.04.20	14	ausgewählte Startzustände aus Liste der Endzustände entfernt, OptimizeControls.vue
Task: Auslagern der Optimize Controlls	Koch, David	16.04.20	16.04.20	2	OptimizeControls.vue
Task: Neuer Optimierungsalgorithmus	Koch, David	17.04.20	23.04.20	10	Beginn eines neuen Algorithmus

VIII.6 Konkrete Code-Qualität im Sprint

Autor: Tim Henning

Die Codequalität hat sich zum vorherigen Sprint nicht entscheidend geändert. Durch den Umbau des Optimierungsalgorithmus hat man nun aber eine etwas höhere Speicherplatz- und Laufzeitkomplexität. Dies soll im nächsten Sprint angegangen und verbessert werden. Durch das Redesign ist die Website im allgemeinen ästhetischer geworden.

VIII.7 Konkrete Test-Überdeckung im Sprint

Autor: Tim Henning

Durch das Hinzufügen der Cypress Tests in die Pipeline des Git-Repository ist nun eine relativ gutes Feedback für den jeweiligen Entwickler und Tester vorhanden. Dieser bekommt nach durchführen der Pipeline eine E-mail, falls der Test fehlschlägt. Für den Optimierungsalgorithmus hingegen fehlen noch ein paar Tests.

VIII.8 Ergebnisse des Reviews

Autor: Tim Henning

Anwesend: Jonas G., Erik H. Linus H., Lennart B., Tim H., David K., Alaa Aldin K., Manuel E., Julius J., Alex H.

Im Review hat wie gehabt, jeder seine erledigten und angefangen Aufgaben vorgestellt und bewertet. So wurde bei der Optimierung die Stückzahl in den Algorithmus integriert, die Endzustände ohne Startzustände werden nun angezeigt und es wurde parallel an zwei neuen Algorithmen gearbeitet, die es ermöglichen die k-besten Pfade auszugeben, und nicht nur den optimalsten Pfad. Dabei wurde einer fertig gestellt, der die Pfade in der Konsole ausgeben kann. Dieser hat aber eine recht hohe Laufzeit- und Speicherplatzkomplexität. Daher wurde ein weitere Algorithmus angefangen, welcher im nächsten Sprint weiterentwickelt und angepasst wird. Zur Userstory der Datenbank und den Initalen Aufgaben zur Bereitstellung wurden erste HTTP Requests angefangen und ausprobiert sowie Axios installiert. Da aber die Datenbank nocht nicht konkret fest stand und noch kein Server von der HTWK zur Verfügung war, wurde sich primär um Bugfixing, Testing und Valiedierungen von Eingaben gekümmert. Die Buttons werden nun nach Windows Standard rechts unten angezeit und sind im Text-only Stil. Desweiteren wurde der Datenbankscreen angepasst und hat nun eine übersichtlichere Darstellung der Elemente, die später einmal aus der Datenbank geladen werden. Zur Userstory der Schnittstelle zwischen Frontend und Backend wurde viel Recherche betrieben. Dabei wurde ein Dokument erstellt, welches alle wichtigen und relevanten Informationen zum Thema API

zusammen trägt. Dieses ist im Git- Wiki zu finden. Im Login Team wurde sich damit beschäftigt ein Rollenmanagement einzuführen und die Anbindung an das Shibboleth zu bekommen. Dies wird im nächsten Sprint weitergeführt. Ebenfalls wurde bei dem IT-Servicentrum der HTWK ein Server bestellt mit folgenden Spezifikationen:

- 64 Bit, Debian
- 4GB Ram, 30GB Festplatte
- Anzahl der CPU's: 1
- Name der VM: Varg
- Netz: DMZ-VM-Fak
- Verwendungszweck: Softwareprojekt
- Verantwortlicher Prof.: Prof. Dr. Martin Gürtler
- Bemerkungen: Anfragen ob ITSZ Apache ausrollt

Als letzter Punkt wurde im Sprint ein neues Design angefangen. Dort wurden auch schon die meisten Funktionen und Menüs implementiert und zum Ende des nächsten Sprints fertig gestellt. Das Projekt wird zum testen für den Kunden auf dem privaten Server eines Teammitgliedes gehostet.

VIII.9 Ergebnisse der Retrospektive

Autor: Tim Henning

Anwesend: Jonas G., Erik H. Linus H., Lennart B., Tim H., David K., Alaa Aldin K., Manuel E., Julius J., Alex H.

Die Retrospektive fand in diesem Sprint online nach dem KALM Prinzip (Keep, Add, Less, More) statt und es wurden wie gewohnt Punkte die das Team ändern muss, aber auch welche die positiv waren und beibehalten werden sollen, angesprochen. So wurde die zahlreiche Teilnahme an den Meetings, sowie die Motivation in diesem Sprint als sehr positiv gewertet. Was im nächsten Sprint hinzu kommen sollte wäre u.a. eine weitere Person für das Team welches sich um das Zusammenspiel zwischen Frontend und Backend kümmert. Auch sollen die Testdokumentationen im Wiki ergänzt und ausgefüllt werden, um nach zu vollziehen welche Components bereits getestet wurden. Desweitern war ein wichtiger Punkt die zeitliche Absprache über das mergen der Branches und das aufräumen im Git Repository. Als Anmerkung unter dem Punkt "Less", wurde zum einen das hinzufügen neuer Features genannt. Das Team will sich in den nächsten Sprints um Robustheit und Testing des vorhanden Codes kümmern und nicht all zu viele neue Features hinzufügen. Außerdem wurde noch angemerkt das die einzelnen Mitglieder YouTrack konsequenter nutzen sollen, um eine bessere Übersicht über den Workflow zu bekommen. Zum Schluss wurde noch erwähnt das der Sprint sehr positiv bewertet wurde, da viele Ziele erreicht wurden und viele neue Erkenntnisse zustande kamen, sowie das sich viele Teammitglieder an dem Sprint beteiligt haben.

VIII.10 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

Der Ablauf in den einzelnen Meetings läuft inzwischen extrem reibungslos. Alle Teammitglieder fühlen sich mit dem Produkt identifiziert. Dies merkte man auch sehr in der Beteiligung während des Sprints und an der Anzahl der abgeschlossenen User-Stories. Durch das parrallele Arbeiten an Front- und Backend wurde eine gute Produktivität erreicht. Die Schnittstelle zwischen Front- und Backend wurde ebenfalls konzipiert.
In diesem Sprint wurde das Produkt auch auf einen Webserver aufgespielt, dass der Kunde die Möglichkeit hat, sich länger mit dem Produkt auseinander zu setzten und damit auch ein besseres und detailierteres Feedback geben kann.
Damit wurde dieser lange Sprint, der über die Prüfungszeit und Semesterferien ging, doch positiv abgeschlossen.

VIII.11 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

VIII.12 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Mit Beginn des neuen Semesters und der damit verbundenen Wiederaufnahme der (Online-) Präsenzveranstaltungen nahm auch die Teilnahme am Projekt wieder zu. Bis auf die beiden Aussteiger haben alle Teammitglieder mitgewirkt. Diese Motivation gilt es auch in den kommenden Wochen aufrecht zu erhalten.

IX. Sprint 6

IX.1 Ziel des Sprints

Autor: David Koch

Vom 27.04. bis 07.05. arbeiteten wir an Sprint 6. Auf Grund von Corona fanden auch hier alle Treffen in Form von online-conference-calls statt. Die Zwischenstände bei den einzelnen Treffen wirkte zwar wenig erfolgversprechend, dennoch wurden fast alle Aufgaben bearbeitet und eingebunden. Die Meetings verliefen problemfrei, die Aufgaben wurden beim Planning gut verteilt und die Zwischenstandsmeetings halfen, kleinere Probleme schnell zu lösen.

IX.2 User-Stories des Sprint-Backlogs

Autor: David Koch

- **Optimierung des Graphen**
  Der Optimierungsalgorithmus soll überarbeitet werden, zum einen sollen die Start- und Endzustände automatisch ausgewählt werden sowie eine neue Größe der Kanten - die Losgröße, die die Wichtung der Rüstkosten beeinflusst - eingebunden werden. Zum anderen soll der Algorithmus nicht nur den besten sondern auch den zweit-, dritt-, usw -besten Pfad auszugeben, das Interface soll an die neuen Ein- und Ausgaben angepasst werden.

- **Erstellen von Bearbeitungsschritten durch klicken**
  Knoten und Kanten sollen durch wenige clicks schnell erstellen werden können. Das Hinzufügen der Eigenschaften wird anschließend durchgeführt, die Validierungen werden dem entsprechend von der Erstellung auf den Optimierungsalgorithmus verschoben.

- **Sammelticket für Feedback vom Kunden**
  Die "Knotenßollen als "Teile"bezeichnet werden und die "Kantenäls "Bearbeitungsschritte". Der Name der Bearbeitungsschritte soll besser erkennbar sein.

- **Design**
  Das Design soll auf die neuste Version angepasst werden.

- **Login**
  Alle Buttons auf der Login-Seite, die zum Testen verwendet wurden, sollen entfernt werden. Die Credentials sollen gespeichert werden und der Graph soll bei einem reload der Seite nicht verloren gehen. Außerdem wird die Anbindung an Shiboleth weiter bearbeitet.

- **Backend Datenbank**
  Es soll eine Datenbank in Docker Container aufgesetzt werden, um erste Tests auf einer Datenbank durchführen zu können und damit bereits eine Schnittstelle implementiert werden kann. Sobald der angeforderte HTWK-Server bereitsteht, soll die Datenbank aufgesetzt und parallel dazu eine Dokumentation aufgesetzt werden. Des Weiteren sollen Testdaten für die Datenbank generiert werden.

IX.3 Liste der durchgeführten Meetings

Autor: David Koch

- 27.04.2020: Planning
- 30.04.2020: Weekly

- 04.05.2020: Weekly
- 07.05.2020: Review, Retro

IX.4 Ergebnisse des Planning-Meetings

Autor: David Koch

Anwesend: Alex, Julius J., Julius H., Linus, Jonas, Erik, Lennart, Nils, Tim, David, Matthias, Manuel

Innerhalb dieses Meetings haben wir die Schwerpunkte des Sprints festgelegt, den Zeitaufwand der User-Stories abgeschätzt und die daraus entstehenden Aufgaben verteilt.

Design

Auf Grund der wandelnden Wünsche des Kunden wurde das Interface des Öfteren redesigned. Da sich das Projekt dem Ende nähert, soll in diesem Sprint die letzte Änderung am Interface vorgenommen werden.

Warten auf Antwort der HTWK

Die Datenbank auf einem HTWK-Server einzurichten sowie die Anbindung des Logins an Shibboleth sind abhängig von Antworten des ITSZ der HTWK und können daher eventuell noch nicht bearbeitet werden.

Weitere Sprintziele:

- Bugtickets bearbeiten
- Losgröße als neue Kanteneigenschaft hinzufügen
- neue hinzugefügten Code testen

IX.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: David Koch

Arbeitspaket	Person	Start	Ende	h	Artefakt
Login	Berger, Matthias	28.04.20	28.04.20	4	Grundlagenrechte
Login	Berger, Matthias	07.05.20	07.05.20	5	Konzeption und Umsetzung von Ladebildschirm, Timeout & Weiterbildung
Login	Buchmann, Lennart	07.05.20	07.05.20	5	Konzeption und Umsetzung von Ladebildschirm, Timeout & Weiterbildung
Login	Buchmann, Lennart	02.05.20	02.05.20	4	Grundlagenrechte
Login	Buchmann, Lennart	04.05.20	04.05.20	5	Login Persistenz
Optimierung des Graphen	Gwozdz, Jonas	30.04.20	30.04.20	1,5	Design erstellen

Sammelticket für Feedback vom Kunden	Gwozdz, Jonas	29.04.20	29.04.20	1,5	Beschriftung ändern
BUG TRACKER	Gwozdz, Jonas	04.05.20	04.05.20	5	Bud: Node-Overlapping ab 3 Knoten
Backend Datenbnk	Heldt, Erik	28.04.20	05.05.20	2,75	Schreiben einer echten Datenbank auf Docker
Backend Datenbank	Heldt, Erik	06.05.20	06.05.20	0,5	HTTP-Request (Client-Side)
Design	Heldt, Erik	06.05.20	06.05.20	3	Ausführliche Tests für GraphInfo.vue
Optimierung des Graphen	Henning, Tim	05.05.20	05.05.20	2	Initialzustände automatisch auswählen
Optimierung des Graphen	Henning, Tim	06.05.20	06.05.20	1	Ausgabe der Gesammtkosten/-zeit auf dem Userinterface
Optimierung des Graphen	Herterich, Linus	30.04.20	30.04.20	2	Settings -> Optimierung persistent speichern
Design	Herterich, Linus	30.04.20	06.05.20	5,5	Redesign Optimierungsansicht
Design	Herterich, Linus	06.05.20	06.05.20	1	Graph-Editor expandierbar
Design	Herterich, Linus	04.05.20	04.05.20	1	Tests zum neuen Design
Design	Herterich, Linus	03.05.20	04.05.20	1,5	Save Menu - Test fixen
Design	Herterich, Linus	04.05.20	04.05.20	1	NewGraph Menu - Test fixen
Design	Herterich, Linus	04.05.20	04.05.20	1,5	Überschreiben Dialog für Save-Menu
Design	Herterich, Linus	27.04.20	30.04.20	1,5	Avatar-Menü → Ausloggen & Einstellungen
Design	Herterich, Linus	27.04.20	27.04.20	2	Merge auf Targetbranch
Design	Herterich, Linus	30.04.20	04.05.20	2,5	Components aufräumen
Design	Herterich, Linus	27.04.20	27.04.20	0,5	Login fixen
Design	Herterich, Linus	27.04.20	27.04.20	1	Tests fixen
Design	Herterich, Linus	07.05.20	07.05.20	0,5	Altes Design in "removed code"
Backend Datenbank	Hohlfeld, Julius	05.05.20	06.05.20	7,5	API-Parser
Backend Datenbank	Hohlfeld, Julius	03.05.20	07.05.20	4	Node.js Programmierung (Server-Side)
Backend Datenbank	Hohlfeld, Julius	28.04.20	29.04.20	14,5	Set-Up Docker MySQL

Backend Datenbank	Hohlfeld, Julius	03.05.20	03.05.20	3	Umgehung des MySQL Authentifizierungsprotokoll
Backend Datenbank	Hohlfeld, Julius	03.05.20	05.05.20	5	Anbindung von Docker zu JS
Login	Karkoutli, Alaa Aldin	07.05.20	07.05.20	5	Konzeption und Umsetzung von Ladebildschirm, Timeout & Weiterbildung
Login	Karkoutli, Alaa Aldin	04.05.20	04.05.20	4	Grundlagenrechte
Optimierung des Graphen	Koch, David	30.04.20	06.05.20	12	Ausgabe der Gesammtkosten/-zeit auf dem Userinterface

IX.6 Konkrete Test-Überdeckung im Sprint

Autor: David

Die Testabdeckung ist deutlich besser als bei vorherigen Sprints. Auch wenn nicht zu allen bearbeiteten Aufgaben Tests angelegt wurden, lag dies lediglich an mangelnder Zeit, wenn sich eine Userstory als aufwendiger als erwartet herausstellte.

IX.7 Ergebnisse des Reviews

Autor: David Koch

Anwesend: Alaa Aldin, Lennart, David, Erik, Julius J., Julius H., Jonas, Linus, Manuel, Tim

Im Rahmen des Reviews haben wir wie gewohnt die Ergebnisse des Sprint bewertet und Schwierigkeiten besprochen.

Problem: Warten auf das ITSZ der HTWK
Die derzeit wichtigsten ausstehenden Aufgaben betreffen vor allem die Themen Server und Datenbank. Da die Daten am Ende auf einem Server der HTWK und die Datenbank eine Anbindung zu Shiboleth haben soll, können wir dies ohne die bisher fehlenden Antworten des ITSZ bisher nur eingeschränkt bearbeiten.

Design:

- neues Design wurde eingebunden
- einige (wenige) Teile sund noch nicht funktional

Optimierung des Graphen:

- Gesamtkosten- und zeit werden im neuen UI ausgegeben, alternative Pfade allerdings noch nicht
- automatische Auswahl von Start- und Endknoten wurde fertiggestellt, jedoch noch nicht mit eingefügt
- Losgröße wurde noch nicht eingebaut

Erstellung von Bearbeitungsschritten durch klciken:

- Validierung der Kanten von Erstellen auf Bearbeiten umgelegt
- vor der Optimierung wird überprüft, ob alle Eigenschaften gegeben sind

Sammelticket für Feedback vom Kunden:

- Tests für Toolbar, Zoom-Controls, Buttons und Eingabereihenfolgen geschrieben

Julius H, Erik, Linus:

- Knoten und Kanten heißen jetzt Teil und Bearbeitungsschritt
- Automatisches Verschieben bei überlappenden Knoten wurde überarbeitet

Login:

- Login-Persistenz ist eingebaut
- Ablaufender Zeitstempel ist eingebaut, wird aber bisher bei zu wenigen Aktionen refreshed
- zum Testen benötigte Buttons wurden entfernt
- Graph geht beim reload nicht mehr verloren
- Login-Credentials können gespeichert werden
- erste Erfahrung sammeln im Umgang mit Shiboleth
- Anbindung an Shiboleth noch nicht möglich (fehlende Antwort des ITSZ)

Backend Datenbank:

- Docker Datenbank aufgesetzt und angebunden (noch nicht auf HTWK-Server)
- Grundlegende Struktur für API-Entwicklung implementiert
- Testdaten in Datenbank eingespei, erste SQL-Befehle auf der Datenbank getestet
- Dokumentation vorhanden

IX.8 Ergebnisse der Retrospektive

Autor: David Koch
Anwesend: Alaa Aldin, Lennart, David, Erik, Julius J., Julius H., Jonas, Linus, Manuel, Tim

Nach den zwischendurch schlechten Einschätzungen bezüglich des Erfolgs dieses Sprints gab es doch eine positive Überraschung, wie viel der Userstories schon umgesetzt werden konnte. Die Retrospektive wird wie im letzten Sprint nach dem KALM-Prinzip durchgeführt.

Keep:

- Produktivität/Motivation
- Zusammenarbeit
- Zeiten buchen
- Übersicht im YouTrack

- Absprache beim Mergen
- kommentierter Code

Add:

- Fortschritte mitteilen
- Dark Mode
- Antworten des ITSZ

Less:

- keine Anmerkungen

More:

- Tests
- Fehlerhafte Tests fixen
- Bugfixing
- Branches löschen

IX.9 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

Kurz vor Beginn des Sprints konnten wir nochmals Feedback vom Kunden erhalten. Daraufhin haben sich noch einige Änderungen für diesen Sprint ergeben wie Änderungen für die Optimierung, das Backend und die Erstellung von einzelnen Knoten-/Kantenelementen. Zudem wurde ein moderneres und ansprechenderes UI-Design implementiert.
Zum Ende des Sprints wurde ein passables Ergebniss erreicht und fast alle Aufgaben wurden zu einer guten Zufriedernheit fertig gestellt. Leider gibt es immer noch in einigen Modulen nicht genügend Tests um die geforderten DoDs zu erfüllen.
Das ITSZ ist für uns immer noch nicht telefonisch erreichbar und hat auf die von uns gesendeten E-Mauls nicht geantwortet.

IX.10 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

IX.11 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Der Sprint schien nach Stand des letzten Scrum Meetings eher negativ auszufallen, jedoch wurden einige User-Stories durch großes Engagement des Teams dennoch in der verbleibenden Zeit finalisiert.

X. Sprint 7

X.1 Ziel des Sprints

Autor: Julius Hohlfeld

Ziel des Sprints war es die wichtige Features wie die Datenbank und Optimierung weiterzuentwickeln und Fortschritte in Design und Usability zu machen.

X.2 User-Stories des Sprint-Backlogs

Autor: Julius Hohlfeld

- **Bugs fixen**
  Als Benutzer möchte ich eine Software benutzen, in welcher keine unerwarteten Probleme auftauchen.
- **Optimierung - Losgröße**
  Als Nutzer möchte ich Losgrößen der Bearbeitungsschritte einstellen und optimieren. Die Optimierung soll automatisch Knoten für Start- und Endzustand auswählen.
- **Drag und Drop Erstellung für Kanten**
  Als Benutzer möchte ich schnell und effizient Kanten erstellen können.
- **Design - Dark Mode**
  Als User will ich persönliche Präferenz über das Aussehen (konkret Dark Mode) der Applikation treffen.
- **Login**
  Als Nutzer will mich mit dem HTWK-Login einloggen (Shibole) und erwarte eine persistente Erfahrunf während ich eingeloggt bin.
- **Backend-Datenbank**
  Als Benutzer möchte ich Graphen über die Anwendung und eine API auf einer Datenbank speichern und von dort aus runterladen.

X.3 Liste der durchgeführten Meetings

Autor: Julius Hohlfeld

- 11.05.2020: Planning
- 15.05.2020: Weekly
- 18.05.2020: Weekly
- 22.05.2020: Review & Retro

X.4 Ergebnisse des Planning-Meetings

Autor: Julius Hohlfeld

Anwesend: Alaa Aldin, David, Erik, Jonas, Julius H., Lennart, Linus, Matthias, Tim, Alex, Manuel, Julius J.

Im Planning haben wir die Ziele für die unterschiedlichen Arbeitsbereiches des Projekts festgelegt und sie in ihrer Schwierigkeit bewertet.

Backend Datenbanken
Es wurde über die Überführung des Datenbankprototyps in Docker auf die HTWK-Server gesprochen. Allerdings hängt dies zur Zeit noch von der Bereitstellung von der HTWK ab. Um das Projekt auch auf echten Servern zu testen, wurde über Alternativen dikutiert. Als Ziel wurde gesetzt weiter die API auszubauen, besonders POST- und DELETE Request sollen umgesetzt werden. Diese Ziele wurden mit einer 6 bewertet.

Login
Es wurde fesges,tellt dass man auch bei Shibole noch auf eine Antwort der HTWK-IT wartet. Es galt als Ziel die Persistenz des Graphen in einer Sitzung zu erreichen, welches mit 6 bewertet wurde.

Optimierung des Graphen
Auf Hinweis des Kunden sollen Losgrößen für Kanten implementiert und bei der Optimierung miteingerechnet werden. Auch eine automatische Auswahl der Start- und Endknoten, falls der User es selber noch nicht ausgewählt hat, soll erzeitl werden. Die Gruppe schätzte diese Vorhaben mit einer 5 ab.

Design
Auf Wunsch des Teams wurde ein Dark Mode für diesen Sprint als Ziel gesetzt. Diese Aufgabe wurde mit einer 4 bewertet.

Erstellung der Kanten durch Drag&Drop
Um die Usability der Anwendung zu erhöhen, wurde vorgeschlagen zwischen verschiedenen Knoten Kanten über Drag&Drop zu erstellen. Es wurde noch diskutiert inwiefern dies mit der Optimierung und Validierung der Kanten kollidiert, da diese User spezifisierte Eingaben benötigen. Das Team stimmte ab und schätzte die Schwierigkeit auf 5.

Bugs fixen
Über die letzen Sprints hatten sich Bugs gesammelt und einige Teammitglieder hatten sich extra auf Bughunt begeben um Probleme zu finden. Daraus ergab sich eine Liste folgender Bugs, die es zu fixen galt:

- Selection-Felder bei Optimierungs-Einstellungen werden nicht initial geladen
- Falsche Anzeige der Gesamtkosten und Zeit nach Importierungen
- Datenbank GUI wechselt nicht automatisch die Seite wenn mehr Elemente angezeigt werden
- Start und Endzustandsanzeige
- Fehlerhafte Verschiebung bei Manchen Knotenkonstellationen
- Nach Laden des Graphen falscher Produktname und Produktanzahl
- Zu viele Dialogs übereinander
- VarG-Dialog verschwindet bei Überlappung zu schnell
- Startknoten
- Valdierung bei Stückzahleingabe

- Import wird auf Richtigkeit getestet
- Slider für Anzahl der Optimierungswege größer als Wege generell

X.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: Julius Hohlfeld

Arbeitspaket	Person	Start	Ende	h	Artefakt
Bug: Nach laden des Graphen falscher Produktname und Produktanzahl	Berger, Matthias	17.05.20	17.05.20	1	GraphInfo.vue
Rollenmanagement	Berger, Matthias	17.05.20	17.05.20	1	store/store.js
Persistenz des Graphen	Berger, Matthias	16.05.20	21.05.20	12,5	store/store.js
Überarbeitung der Weiterleitung	Berger, Matthias	16.05.20	16.05.20	1	router/index.js
Bug: Nach laden des Graphen falscher Produktname und Produktanzahl	Buchmann, Lennart	17.05.20	17.05.20	1	GraphInfo.vue
Rollenmanagement	Buchmann, Lennart	17.05.20	17.05.20	1	store/store.js
Persistenz des Graphen	Buchmann, Lennart	16.05.20	21.05.20	11,5	store/store.js
Überarbeitung der Weiterleitung	Buchmann, Lennart	16.05.20	16.05.20	1	router/index.js
Tests für Einstellungen	Gwozdz, Jonas	14.05.20	14.05.20	0,5	settings_spec.js
Testen der Applikation	Gwozdz, Jonas	12.05.20	12.05.20	0,5	Testen der Applikation
Dark Mode	Gwozdz, Jonas	13.05.20	21.05.20	18	Darkmode.vue
Farbschema erstellen	Gwozdz, Jonas	13.05.20	13.05.20	0,75	darkmode.less
"vargßu "VarG"ändern	Gwozdz, Jonas	14.05.20	14.05.20	0,33	Gesamtes Projekt
Bug: Importieren funktioniert bei veränderten Knotenpositionen nicht	Heldt, Erik	12.05.20	12.05.20	0,14	removed Code
Bug: MenuControls-Fenster ist nach unten verschoben	Heldt, Erik	12.05.20	12.05.20	0,14	removed Code
Validierung bei Stückzahleingabe	Heldt, Erik	12.05.20	12.05.20	1	GraphInfo.vue
Axios-Requests für Post, Get usw.	Heldt, Erik	13.05.20	20.05.20	8,25	DatabaseForm.vue
Ersetzen der TestDatabase durch MySQL Datenbank	Heldt, Erik	13.05.20	20.05.20	6,25	DatabaseForm.vue

Testdaten für die Datenbank	Heldt, Erik	19.05.20	19.05.20	0,25	dump.sql
Kodierungsfehler beheben	Heldt, Erik	19.05.20	19.05.20	1,5	api.js
Initialzustände auswählen	Henning, Tim	12.05.20	18.05.20	4	optimization.js
Bug: Startknoten	Henning, Tim	12.05.20	12.05.20	2	Graph.vue
Tests für Optimierung	Henning, Tim	18.05.20	18.05.20	3	optimize.spec.js
Optimierung des Graphen	Herterich, Linus	21.05.20	21.05.20	0,5	optimization.js
Erstellung von Bearbeitungsschritte durch Klicken	Herterich, Linus	21.05.20	21.05.20	0,5	RightClickMenu.vue
Validierung umstellen	Herterich, Linus	18.05.20	18.05.20	0,5	DetailControls.vue
Tests für Einstellungen	Herterich, Linus	14.05.20	14.05.20	0,75	settings_spec.js
VarG-Dialog verschwindet bei Überlappung zu schnell	Herterich, Linus	12.05.20	12.05.20	3	Dialogs.vue
Dark-Mode	Herterich, Linus	17.05.20	19.05.20	2,5	Darkmode.vue
Konzeption Kantenerstellung	Herterich, Linus	14.05.20	14.05.20	0,5	konzeptuelle Aufgabe
Drag&Drop Funktionalität	Herterich, Linus	14.05.20	19.05.20	8	edges.js
Rechtsklick-Menü: Kante "von/"nach"überarbeiten	Herterich, Linus	21.05.20	21.05.20	0,5	RightClickMenu.vue
Tests: Kantenerstellung	Herterich, Linus	21.05.20	21.05.20	0,5	quickEdges_spec.js
Warnungs-Dialog Farbe anpassen	Herterich, Linus	18.05.20	18.05.20	0,5	Dialogs.vue
zweit- bis x-besten Graphen ausgeben	Herterich, Linus	19.05.20	20.05.20	3,75	OptimizeControls.Vue
Bug: Losgröße - Detail Menü	Herterich, Linus	19.05.20	19.05.20	0,5	DetailControls.vue
Validierung Losgröße	Herterich, Linus	19.05.20	19.05.20	0,5	DetailControls.vue
Zu viele Dialogs übereinander	Herterich, Linus	21.05.20	21.05.20	1	Dialogs.vue
Neue Labels	Herterich, Linus	21.05.20	21.05.20	0,75	edges.js
API-Parser	Hohlfeld, Julius	12.05.20	14.05.20	0,66	api.js
Import wird nicht auf Richtigkeit getestet	Hohlfeld, Julius	13.05.20	19.05.20	9,25	JsonPersistence.js
Express-Post	Hohlfeld, Julius	14.05.20	14.05.20	2,25	api.js

Express-Delete	Hohlfeld, Julius	12.05.20	12.05.20	0,08	api.js
Express-Put	Hohlfeld, Julius	12.05.20	12.05.20	0,5	api.js
Sprint Doku 7	Hohlfeld, Julius	11.05.20	22.05.20	3,5	sprint7.tex
Kodierungsfehler beheben	Hohlfeld, Julius	19.05.20	20.05.20	3,5	api.js
Persistenz des Graphen	Karkoutli, Alaa Aldin	16.05.20	21.05.20	13,5	store/store.js
Überarbeitung der Weiterleitung	Karkoutli, Alaa Aldin	16.05.20	16.05.20	1	router/index.js
Losgröße einbinden	Koch, David	14.05.20	15.05.20	3	edges.js
Tests für Optimierung	Koch, David	18.05.20	18.05.20	2	optimize.spec.js
zweit bis x-besten Graphen ausgeben	Koch, David	19.05.20	20.05.20	7	OptimizeControls.Vue

X.6 Konkrete Code-Qualität im Sprint

Autor: Julius Hohlfeld

Die Qualtität des Codes war weiterhin gut. Das Team wurde allerdings vom Software-Architekten darauf hingewiesen, die Kommentare verständlicher zu schreiben.

X.7 Konkrete Test-Überdeckung im Sprint

Autor: Julius Hohlfeld

Sämtliche neuen Änderungen wurden auch mit Tests kontrolliert. Es kam zur Diskussion auch Tests für die API zu schreiben, welche noch nicht getestet wird.

X.8 Ergebnisse des Reviews

Autor: Julius Hohlfeld

Anwesend: Alaa Aldin, Erik, Jonas, Julius H., Lennart, David, Manuel, Alex, Julius J.

Im Review stellte jeder Anwesende seine Arbeit vor. Einige fehlten, wurden aber durch eng zusammenarbeitende Teammitglieder vertreten.
Die Ergebnisse und Erkenntnisse wurden ausgewertet.

HTWK-IT

Mehrfach kam das Thema der fehlenden Kommunikation auf Seiten der HTWK-IT für unser projektkritischen Anfragen zur Sprache. Es wurden neue Kommunikationsversuche besprochen.

David und Tim:

- Losgröße integriert
- Pfad-Optimierung beachtet Losgröße + Ausgabe

- Initialzustände beim Optimieren
- Optimierungstests

Erik:

- Stückzahl: keine Kommas und führende Nullen
- Laden, Löschen, Hochladen von Graphen in DB
- Hashkey für Speicherung

Jonas:

- Test für Kantenerstellung
- Darkmode (eigene Komponente)

Julius H.:

- API-Anpassungen
- JSON Validation
- Kodierungsfehler behoben

Linus:

- Pfad-Ausgabe
- Drag&Drop + Tests
- Benachrichtigungs Quality of Life
- Einstellungsmenü Tests

Alaa Aldin, Lennart und Matthias:

- Graph hat Persistenz
- Rollenmanagement
- Weiterleitung

Bugs
Es wurden Bugs angesprochen, welche im Verlauf des Sprints oder beim mergen auf den Target-branch entdeckt wurden.

X.9 Ergebnisse der Retrospektive

Autor: Julius Hohlfeld
Anwesend: Alaa Aldin, Erik, Jonas, Julius H., Lennart, David, Manuel, Alex, Julius J.

Das Team war zufrieden mit der erreichten Leistung und hat eine positve Ansicht zum Sprint. Angesichts des baldigem Ende des Softwareprojekts wurde auch die Verwendung der restlichen Zeit kurz diskutiert.

Keep	Add	Less	More
-Zusammenhalt -Kommunikation	-Server - Tests	-neue instabile Features (weil wenig Sprints übrig)	-Druck bei HTWK für Serverplatz Tests -Bugfixes -Code Kommentare -Quality assurance

X.10 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

In den vorherigen Sprints wurde größerer Fokus auf die Ausarbeitung des Frontends gelegt. Somit kann in diesem und den kommenden Sprints der Schwerpunkt auf das Backend gelegt werden. Dies wurde zusätzlich begünstigt, da wir immer noch keine Rückmeldung des ITSZ bekommen haben.
Inzwischen haben die Entwickler eine gute Kompetenz entwickelt, den Umfang der einzelnen Aufgaben einzuschätzen. Somit konnten wir fast alle im Planning-Meeting besprochenen User-Storys erfolgreich abschließen.
Leider mussten wir Aufgrund der drängenden Zeit und der schlechten Kommunikation zum ITSZ, eine Authentifizierung mittels Shibboleth verwerfen.

X.11 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

X.12 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Weiterhin trotz aller Umstände eine sehr fokussierte und lobenswerte Arbeit des Teams, gerade auch im Vergleich zu anderen Teams, wie man von deren Masterstudenten hört.

XI. Sprint 8

XI.1 Ziel des Sprints

Autor: Alaa Aldin Karkoutli

Ziel des Sprints war es, das Software zu dokumentieren und die noch stehenden Bugs zu beheben.

XI.2 User-Stories des Sprint-Backlogs

Autor: Alaa Aldin Karkoutli

- **Bugs fixen**
  Als Benutzer möchte ich eine Software benutzen, in welcher keine unerwarteten Probleme auftauchen.
- **Optimierung**
  Als Nutzer möchte ich wissen, nach welchen Kriterien die Graphen optemiert sind .
- **Dokumentation**
  Als Benutzer möchte mit einem Software arbeiten, das eine vollständige Dokumentation hat .
- **Persistenz der eingetragenen Daten**
  Als User will ich Kanten/Knoten Eigenschaften automatisch zu speichern, wenn sie bearbeitet werden.
- **Login**
  Als Nutzer möchte ich mehr Zugriff auf der DatenBank haben wenn ich Admin bin(Rollenmangment) und Login noch bissel optemiert sehen.
- **Backend-Datenbank**
  Als Benutzer möchte ich Test-Graphen in der Datenbank finden und die DatenBank vor Gefahr schützen .

XI.3 Liste der durchgeführten Meetings

Autor: Alaa Aldin Karkoutli

- 25.05.2020: Planning
- 28.05.2020: Weekly
- 01.06.2020: Weekly
- 04.06.2020: Review & Retro

XI.4 Ergebnisse des Planning-Meetings

Autor: Alaa Aldin Karkoutli

Anwesend: Alaa Aldin, David, Erik, Jonas, Julius H., Lennart, Linus, Matthias, Tim, Alex, Manuel, Julius J.

Im Planning haben wir die wichtigsten Punkte festgesetzt, die in der letzten Sprints Priorität

haben, und die Schwierigkeit von denen bewertet.

Backend Datenbanken
Die Datenbank soll auf einem Server aufgesetzt werden, wenn Server bereitgestellt ist. Die generierte Test-Daten können in die Datenbank eingespeist werden. Als Ziel wurde gesetzt die Bugs zu beheben und alles nebenbei zu dokumentieren. Diese Ziele wurden wie im letzten Sprint mit einer 6 bewertet.

Login
Weil es keine Antwort von IT-HTWK gab, konnte das Software an Schibboleth nicht verbunden werden. Es galt als Ziel Bugs zu behebn, Login zu überarbeiten und einen einfachen Zugriff auf die Datenbank vorzubereiten, in Absprache mit Backend-Team. Diese Ziele wurden mit einer 7 bewertet.

Optimierung
Es ist wichtig anzuzeigen, nach welchen Kriterien der Graph Optimiert ist. Dieses Ziel wurde mit 4 bewertet.

Persistenz der eingetragen Daten
Die Eigenschaften der Kanten/Knoten sollen automatisch gespeichert, wenn der Benutzer irgendeine Änderung auf diese Eigenschaften macht. Die Losgröße soll auch ohne Scrollbar dargestellt werden. Diese Ziele wurden mit 4 bewertet, aber nach hinten eingestellt, da Doku Priorität hat.

Dokumentation
Das Software soll ausführlich dokumentiert werden. Tests, Code-Kommentare und Technische Ausarbeitungen müssen noch gemacht werden. Die Gedanken, die nicht gemacht werden konnten (Schibboleth, Datenbank..), müssen auch in der Dokumentation stehen. Das Team stimmte ab und schätzte die Schwierigkeit auf 7.

Bugs fixen
Möglichst alle Bugs zu beheben ist seit den letzten Sprints in Bearbeitung. Das Ziel ist es, ein fehlerfreies Software abzugeben, indem alles ausprobieren und die Bugs lösen.

XI.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: Alaa Aldin Karkoutli

Arbeitspaket	Person	Start	Ende	h	Artefakt
Überarbeitung Login	Berger, Matthias	31.05.20	31.05.20	4	LoginForm.vue + store/store.js
Überarbeitung Login	Buchmann, Lennart	31.05.20	31.05.20	4	LoginForm.vue + store/store.js
Überarbeitung Login	Buchmann, Lennart	04.06.20	04.06.20	1	LoginForm.vue + store/store.js
Bug: Fehlerhafte Verschiebung bei Manchen Knotenkonstellationen	Gwozdz, Jonas	27.05.20	27.05.20	2,15	position.js + src/vargraph/graph/edges.js
Bug: Fehlerhafte Verschiebung bei Manchen Knotenkonstellationen	Gwozdz, Jonas	02.06.20	02.06.20	2,30	position.js + src/vargraph/graph/nodes.js
Verschieben der Knoten beim anlegen	Gwozdz, Jonas	02.06.20	02.06.20	0,30	src/vargraph/graph/nodes.js

Test für Darkmode	Gwozdz, Jonas	03.06.20	03.06.20	1,40	darkmode_spec.js
I.1 Initiale Kundenvorgaben	Gwozdz, Jonas	03.06.20	03.06.20	0,40	Dokumentation
III.2 Coding Style	Gwozdz, Jonas	03.06.20	03.06.20	1,10	Dokumentation
Dokumentation	Heldt, Erik	28.05.20	28.05.20	3,45	documentation/projektdokumentation.tex
Anzeigen des Graphen als Bild in der GUI	Heldt, Erik	01.06.20	03.06.20	3,30	DatabaseForm.vue
Ersetzen der TestDatabase durch MySQL DB	Heldt, Erik	03.06.20	03.06.20	0,30	removedcode/ExportDatabase
Überarbeitung Login	Heldt, Erik	04.06.20	04.06.20	1	verschiedene Stellen
DB Button in Home Menu nicht korrekt angebunden	Heldt, Erik	03.06.20	03.06.20	1	NewGraph.vue
Testen der Applikation	Heldt, Erik	27.05.20	27.05.20	0,30	
Konsistenz bei der Positionierung ähnlicher Komponenten	Heldt, Erik	01.06.20	03.06.20	0,50	verschiedene Stellen
III.1 Definition of Done	Henning, Tim	03.06.20	03.06.20	1	projektdokumentation.tex
Bug: Start und Endzustandsanzeige	Henning, Tim	29.05.20	29.05.20	2	vargraph/graph/optimizations.js
Selection-Felder bei Optimierungs-Einstellungen	Henning, Tim	31.05.20	31.05.20	4	GraphInfo.vue + vargraph/graph/optimizations.js
II.3 Überblick über Architektur	Herterich, Linus	03.06.20	04.06.20	1,45	projektdokumentation.tex
Testen der Applikation	Herterich, Linus	03.06.20	03.06.20	0,5	
Drag-n-Drop Button verschwindet nach drüberhovern nicht	Herterich, Linus	28.05.20	03.06.20	2,10	verschiedene Stellen
Bug: Falsche Anzeige der Gesamtkosten und Zeit nach Importieren	Hohlfeld, Julius	25.05.20	28.05.20	1	vargraph/JSonPersistence.js
Datenbank GUI wechselt nicht automatisch die Seite	Hohlfeld, Julius	27.05.20	27.05.20	1	verschiedene Stellen
Schutz vor SQL-Injections	Hohlfeld, Julius	03.06.20	03.06.20	1,30	api.js
II.4 Definierte Schnittstellen	Hohlfeld, Julius	03.06.20	04.06.20	3,30	projektdokumentation.tex
Backend Test Recherche	Hohlfeld, Julius	03.06.20	03.06.20	0,5	Recherche
Bugs:Persistenz des Graphen	Karkoutli, Alaa Aldin	02.06.20	02.06.20	3	router/index.js
Sprint-Doku	Karkoutli, Alaa Aldin	01.06.20	05.06.20	4	documentation/Sprint_8.tex

Graph wird nicht heruntergeladen,"Neuer Graph->SSpeichern"	Karkoutli, Alaa Aldin	01.06.20	01.06.20	1	ExportDownload.vue + HomeMenu.vue
Überarbeitung Login	Karkoutli, Alaa Aldin	31.05.20	31.05.20	4	LoginForm.vue + store/store.js
Selection-Felder bei Optimierungs-Einstellungen	Koch, David	31.05.20	31.05.20	2	GraphInfo.vue + vargraph/graph/optimizations.js
X.1 Handbuch	Koch, David	04.06.20	04.06.20	2	projektdokumentation.tex

XI.6 Konkrete Code-Qualität im Sprint

Autor: Alaa Aldin Karkoutli

Die Codequatlität war weiterhin gut.

XI.7 Ergebnisse des Reviews

Autor: Alaa Aldin Karkoutli

Anwesend: Alaa Aldin, Erik, Jonas, Julius H., Lennart, David, Manuel, Alex, Julius J., Matthias, Linus, Tim

Im Review stellte jeder Anwesende seine Arbeit vor.
Die Ergebnisse und Erkenntnisse wurden ausgewertet.

David:

- Handbuch dokumentiert
- Optimierung nicht viel gemacht

Erik:

- Dokumentationen geschrieben
- Datenbank Preview-Bilder anzeigen
- Testdatabase.js entfernt

Jonas:

- Springen der Knoten gefixt
- Bechreibung in der Kanten lesen
- Tests für Darkmode
- Dokumentation geschrieben

Julius H.:

- Dokumentation geschrieben.
- Sicher vor SQL-Injection
- Bugs (Importiern und Datenbank) gefixet

Linus:

- Bug gefixt (Darg und Drop Knoten)
- Dokumentation geschrieben

Alaa Aldin, Lennart und Matthias (Login-Team):

- Login-Überarbeitung vorbereitet (Code auskommentiert)
- Rollenmanagement

Alaa Aldin:

- Bugs(NeuerGraph -> Speichern, Persistenz der Graphen, Logout) gefixt
- Knoten und Kanten in der gespeicherten Positionen laden
- Änderungen auf importierten Graphen in der LocalStorage speichern

Bugs
Bugs sollen im Laufe des kommenden Sprints gefixt werden.

XI.8 Ergebnisse der Retrospektive

Autor: Alaa Aldin Karkoutli
Anwesend: Alaa Aldin, Erik, Jonas, Julius H., Lennart, David, Manuel, Alex, Julius J.

Das Team war zufrieden mit der erreichten Leistung und hat eine positve Ansicht zum Sprint. Angesichts des baldigem Ende des Softwareprojekts wurde auch die Verwendung der restlichen Zeit kurz diskutiert.

Keep	Add	Less	More
-Teamwork -Bugfixing	-Motivation -Sprint -funktionierender Login -Console Errors entfernen	-unkompilierbare Doku	-Code Kommentare -Branches aufräumen -Infos über Produktions-Version

XI.9 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert
Da dies der vorletzte Sprint war, wurde Wert darauf gelegt, möglichst keine große neue Funktionalitäten zu entwickeln, sonder möglichst die bestehenden zu finalisieren. Weiterhin galt es Bugs die sich beim Entwickeln ergeben haben zu beseitigen. Als letzten großen Punkt wurde die Projektdokumentation in diesem Sprint bearbeitet.
Da oft noch keine ausführliche Dokumentation zu den einzelnen Sprints geschrieben wurde, hat dies auch einen großen Teil des Sprints eingenommen. Trotz allem konnten wir alle User-Stories erfolgreich beendigen.
Da wir bis zu diesem Zeitpunkt vom ITSZ immer noch keine Nachricht bekommen haben, ob beziehungsweise wann wir einen Serverplatz bekommen können haben wir uns nach anderen Möglichkeiten umgesehen. Glücklicherweise hat sich der StudierendenRat sofort bereit erklärt uns zu helfen und ein Teil ihres Servers, welcher sich auch im HTWK-Netz befindet, für uns bereitzustellen.

XI.10 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

XI.11 Abschließende Einschätzung des Team-Managers

Autor: Alex Hofmann

Es ist vereinzelt an der Motivation zu spüren, dass sich das Semester und somit auch das Softwareprojekt dem Ende zuneigt. Dies liegt vermutlich auch daran, dass aufgrund von nur noch einem verbleibenden Sprint einige Komponenten, woran die Bachelor jetzt schon mehrere Wochen dran gearbeitet haben, auf der Strecke bleiben werden.

XII. Sprint 9

XII.1 Ziel des Sprints

Autor: Matthias Berger

Das Hauptaugenmerk im 9. Sprint wurde auf die Behebung noch vorhandener Bug, sowie die Projektdokumentation gelegt. Lediglich die Umsetzung der Loginfunktion im Backend wurde als größeres Ziel verfolgt.

XII.2 User-Stories des Sprint-Backlogs

Autor: Matthias Berger

- **Ausarbeitung des Latex Dokuments, Code Kommentare, Technische Ausarbeitung**

- **Persistenz der eingetragenen Daten**
  Wenn Kanten/Knoten Eigenschaften bearbeitet werden, automatische Speicherung (Wenn Fenster geschlossen wird, bzw aus dem Eigenschaftenfenster geklickt wird)
  Darstellung der Losgröße ohne Scrollbar (oder bei leerem Feld dort hin springen zweite Seite)

- **Umzug Datenbank auf Fachschaftsserver**
  Datenbank soll auf den Server des FSR umgezogen werden
  VarG von dort aus hosten lassen

- **Optimierung**
  Klarer machen, nach welchem Kriterium(Zeit/Kosten) Optimiert wurde (Auf Oberfläche und in Einstellungen)

- **BUG TRACKER**

- **Login**
  Überarbeitung Login (nicht in Klartext lesbar, ...)

- **Dokumentation**

- **Abschlusspräsentation Messe vorbereiten**

XII.3 Liste der durchgeführten Meetings

Autor: Matthias Berger

- 08.06.2020: Planning Meeting

- 11.06.2020: Weekly Scrum

- 15.06.2020: Weekly Scrum

- 18.06.2020: Review & Retro

XII.4 Ergebnisse des Planning-Meetings

Autor: Matthias Berger

Anwesend: Jonas Gwozdz, Erik Heldt, Linus Herterich, Julius Hohlfeld, Lennart Buchmann, Tim Henning, Matthias Berger, Alaa Aldin Karkoutli, Manuel Eckert, Julius Jolig, Alex Hofmann

Zunächst wurde sich darauf geeinigt keine weiteren großen Projekte in Angriff zu nehmen. Stattdessen sollen ggf. Kommentare im Quelltext hinterlassen werden, die evtl. verfolgte Lösungsansätze dokumentieren sollen. Einzig großes Projekt war die Überarbeitung des Logins, sodass Nutzername und Passwort nicht weiter im Quelltext auslesbar sind. Ursprünglich sollte dies durch die Anbindung an Shibboleth ersetzt werden. Stattdessen musste nun innerhalb kürzester Zeit eine Verifikation im eigenen Backend umgesetzt werden. Die Hauptpriorität sollte auf der Dokumentation des Projektes, der Präsentation für die Messe und weiteren Bugfixes liegen.

XII.5 Aufgewendete Arbeitszeit pro Person+Arbeitspaket

Autor: xxx

Arbeitspaket	Person	Start	Ende	h	Artefakt
Dummyklassen	Musterstudi	3.5.09	12.5.09	14	Klasse.java
AP XYZ					

XII.6 Konkrete Code-Qualität im Sprint

Autor: Matthias Berger

Eines der Hauptziele im Sprint war die Kommentierung des Quelltextes zu überarbeiten. Es ist also davon auszugehen, dass sich die Qualität des Codes im laufe des Sprintes verbessert hat.

XII.7 Konkrete Test-Überdeckung im Sprint

Autor: Matthias Berger

Da während des Sprints keine neuen Funktionalitäten implementiert wurden, sondern lediglich Bugfixing und Überarbeitung im Vordergrund standen war es auch nicht nötig weitere Tests zu formulieren.

XII.8 Ergebnisse des Reviews

Autor: Matthias Berger

Anwesend: Jonas Gwozdz, Erik Heldt, Linus Herterich, Julius Hohlfeld, Lennart Buchmann, Tim Henning, David Koch, Matthias Berger, Manuel Eckert, Julius Jolig, Alex Hofmann

Im Rahmen des Reviews wurde festgehalten, dass der Großteil der der geplanten Sprintziele umgesetzt werden konnte. Um letzte kleinere Ausbesserungen, den vollständigen Umzug auf den vom FSR bereitgestellten Server und die Vorbereitung einer Präsentation für die geplante Messe zu ermöglichen wurde sich darauf geeinigt den Sprint zu verlängern. Neue Ziele wurden nicht formuliert.

XII.9 Ergebnisse der Retrospektive

Autor: Matthias Berger

Anwesend: Jonas Gwozdz, Erik Heldt, Linus Herterich, Julius Hohlfeld, Lennart Buchmann, Tim Henning, David Koch, Matthias Berger, Manuel Eckert, Julius Jolig, Alex Hofmann

Da es sich um den letzten offiziellen Sprint handelte wurde in der Retrospektive auf das KALM-Schema verzichtet. Stattdessen wurde der Verlauf des gesamten Projektes bewertet. Die Meinungen hierzu fielen durchweg positiv aus. So wurde positiv bewertet, dass sich im laufe des Projektes trotz der anfänglichen Schwierigkeiten in Organisation und Kommunikation zwischen den Teammitgliedern ein Teamgeist und ein Produkt, mit dem sich alle Beteiligten identifizieren können entwickelt hat. Auch die Kommunikation innerhalb und zwischen den Einzelteams und die Brücke, die zwischen den beiden Studiengängen der Teammitglieder geschlagen wurde wurde positiv hervorgehoben.

XII.10 Abschließende Einschätzung des Product-Owners

Autor: Manuel Eckert

In diesem Sprint konnten wir endlich das entwickelte Backend auf einen Server im Netz der HTWK spielen. Dies war noch der letzte Meilenstein zu einer erfolgreichen Auslieferung des Produktes.
Da dies dann auch der letzte Sprint des SoftwareProjektes war, wurden nur noch kleinere User-Stories verteilt. Somit konnten sich die Entwickler auf diese Aufgaben konzentrieren und es wurde sichergestellt, dass keine Funktionalitäten Aufgrund von Zeitmangel verworfen werden mussten.
Das ein Projektabschluss trotz allem doch sehr schnell anstrengend werden kann haben wir alle gemerkt, da nicht immer alles ganz nach Plan läuft und in letzter Sekunde doch noch einmal Probleme auftauchen.
Nach dem Sprint war das gesamte Team sehr zufrieden, mit dem was wir trotzdem noch in diesem Sprint abgearbeitet haben.

XII.11 Abschließende Einschätzung des Software-Architekten

Autor: xxx

XXX

XII.12 Abschließende Einschätzung des Team-Managers

Autor: xxx

XXX

XIII. Dokumentation

XIII.1 Handbuch

Autor: David Koch

- **Login**
  Um sich einloggen zu können, benötigt man einen Account. Diesen legt man an, indem man sich im Login-Fenster mit einem neuen Benutzernamen und Passwort einloggt. Der angegebene Name wird als neuer Nutzer (mit Rolle: "Student") angelegt und das dazugehörige Passwort gespeichert. Beides kann nachträglich in den Benutzereinstellungen (siehe "Einstellungen") geändert werden.
  Hinweis: Wenn beim Versuch, einen neuen Account zu erstellen, der Fehler "Ungültige Login-Daten" auftritt, bedeutet das, dass der eingegebene Benutzername bereits existiert. In diesem Fall muss ein anderer Name gewählt werden.
  Ein Ändern der Rolle (von "Student" auf "Admin") ist nur durch direkten Zugriff auf die Datenbank möglich. (Momentan wird eine programmeigene User-Datenbank benutzt, diese könnte später durch eine Anbindung an Shibboleth ersetzt werden.)

- **Neuen Graphen erstellen**
  Über den Menüpunkt 'Neuen Graphen erstellen' im Startfenster gelangt man in das 'Neues Produkt' Fenster. Hier kann Name und Stückzahl des neuen Graphen festgelegt werden. Über den 'Starten' Button wird der neue Graph erstellt. Im Graphenfenster gelangt man über den 'Neuer Graph' Button zurück ins Startmenü. Über die Stift-Icons neben dem Produktnamen und der Stückzahl können diese nachträglich bearbeitet werden.

- **Erstellen und Bearbeiten von Zuständen**
  Das 'Neues Teil' Menü öffnet sich entweder über Rechtsklick innerhalb des Graphenfensters, gefolgt von einem Linksklick auf 'Neues Teil' oder durch einen Linksklick auf den Plus-Button in der rechten unteren Ecke, gefolgt von einem Linksklick auf 'Neues Teil'. Hier kann neben Name, Kürzel und Farbe des Zustands auch ein dazugehöriges Icon mittels URL gewählt werden. Alle Angaben außer dem Icon sind Pflichtangaben, sodass der 'erstellen' Button erst betätigt werden kann, wenn alle diese Felder beschrieben sind. Mit einem Linksklick auf einen bereits erstellten Zustand öffnet sich das 'Teil bearbeiten' Menü. Dieses ist aufgebaut wie das 'Neues Teil' Menü. Hier können alle Eigenschaften des angeklickten Zustandes bearbeitet werden.

- **Erstellen und Bearbeiten von Bearbeitungsschritten**
  Das 'Neuer Bearbeitungsschritt' Menü öffnet sich entweder über Rechtsklick innerhalb des Graphenfensters, gefolgt von einem Linksklick auf 'Neuer Bearbeitungsschritt' ,oder durch einen Linksklick auf den Plus-Button in der rechten unteren Ecke, gefolgt von einem Linksklick auf 'Neuer Bearbeitungsschritt'. Im ersten Teil können Name, Kürzel sowie Start- und Endzustand gewählt werden. Im zweiten Teil definiert man Losgröße, Zeit- und Geldkosten sowie Zeit und Geldrüstkosten des Bearbeitungsschrittes. Alle Felder sind Pflichtfelder, es müssen also alle Felder beschrieben sein, um den 'erstellen' Button betätigen zu können. Des Weiteren kann man Bearbeitungsschritte auch mittels drag'n'drop erstellen. Dafür bewegt man den Mauszeiger über den Zustand, der als Startzustand dienen soll, klickt auf das neuer Bearbeitungsschritt-Icon und zieht die entstehende Linie zu dem Endzustand. Anschließend öffnet sich der zweite Teil des 'Neuer Bearbeitungsschritt' Menüs, in dem dann die übrigen Eigenschaften nachträglich eingefügt werden müssen. Mit einem Linksklick auf einen bereits erstellten Bearbeitungsschritt öffnet sich das 'Bearbeitungsschritt bearbeiten' Menü. Dieses ist aufgebaut wie das 'Neuer Bearbeitungsschritt' Menü. Hier können alle Eigenschaften des angeklickten Bearbeitungsschritt geändert werden.

- **Graph optimieren**
  Um den besten Weg von einem der Startzustände zum Endzustand zu finden, klickt man auf 'Graph optimieren' unter 'Gesamtkosten' (um nach Kosten zu optimieren) oder unter 'Gesamtzeit' (um nach Zeit zu optimieren), Start- und Endzustände werden in diesem Fall automatisch gewählt. Nach dem Betätigen des Buttons verschwindet dieser und wird ersetzt durch die Gesamtkosten bzw. die Gesamtzeit des besten Weges im Graphen, außerdem ist dieser Weg Orange markiert. Abhängig davon, ob nach Kosten oder Zeit optimiert wurde wird das entsprechende Icon sowie das Wort ('Gesamtkosten' oder 'Gesamtzeit') ebenfals Orange markiert. Möchte man die Parameter der Optimierung eigenhändig bearbeiten, lässt sich das entsprechende Menü öffnen, indem man entweder auf eines der Zahnräder (neben 'Gesamtkosten' oder 'Gesamtzeit') klickt oder auf 'Einstellungen' am oberen Rand klickt und in den Tab 'Optimierung' wechselt. In diesem Menü lassen sich die Start- und Endzustände auswählen, sowie wonach optimiert werden soll (Kosten oder Zeit) und wie oft optimiert werden soll (3 mal optimieren heißt, dass die besten 3 Wege angezeigt werden, diese Wege sind unter den Einstellungen aufgelistet). Auch in diesem Menü kann man die Optimierung starten, indem man auf 'Optimierung starten' (unten im Einstellungsmenü) klickt, sind Start- und Endzustände nicht ausgewählt, werden diese wieder automatisch bestimmt, die Optimierungsart (Zeit/Kosten) muss allerdings ausgewählt sein (alternativ dient der Button 'anwenden' zum übernehmen der Einstellungen und 'schließen' zum verwerfen der Einstellungen. In beiden Fällen schließt das Menü). Nach dem Betätigen des 'Optimierung starten' Buttons (im Einstellungsmenü) wird dieser ersetzt durch die besten Wege im Graphen (begrenzt durch Optimierungsanzahl). Durch klick auf einen dieser, erhält man einen Einblick in die einzelnen Zustände und Bearbeitungsschritte des Weges, durch klick auf den Kreis links vom angezeigten Weg, wird dieser anstatt dem besten Weg orange markiert. Sobald etwas am Graph geändert wird, erscheinen die 'Graph optimieren' bzw. 'Optimierung starten' Buttons wieder. Zuvor lässt sich der Graph durch Klicken auf das Wiederholen-Icon (neben 'Gesammtzeit' oder 'Gesamtkosten' im Graphenfenster) erneut optimieren.

- **Einstellungen**
  Mit einem Linksklick auf "Einstellungen" am oberen Rand öffnet sich das Einstellungsfenster. Im Tab "Graph" lassen sich Einstellungen bezüglich der Darstellung des VarGraphs vornehmen, wie zum Beispiel die Einheiten oder die angezeigten Details der Verknüpfungen (Bearbeitungsschritte). Außerdem lässt sich hier ein "Raster" aktivieren, auf dem die Knoten dann gebunden sind. Im Tab "Benutzer" lassen sich Benutzername und Passwort ändern, sowie der Account löschen. Beim Ändern des Benutzernamens wird der "Autor" jedes mit diesem Account erstellten Graphen ebenfalls geändert, beim Löschen des Accounts werden auch alle mit diesem Account erstellten Graphen gelöscht. Zum Löschen des Accounts oder Ändern des Passworts wird das Passwort benötigt. Im Tab "Hilfe" ist dieses Handbuch verlinkt.

- **Offline Speicher**
  Über den Button "Export" am oberen Rand lässt sich der Graph in den Formaten .json, .png und .jpg exportieren und lokal speichern. Über den "Import" Button lässt sich ein in .json exportierter Graph wieder im Programm öffnen.

- **Online Speicher**
  Über den Button "Datenbank" am oberen Rand öffnet sich das Datenbank-Fenster. Hier können erstellte Graphen online gespeichert bzw. wieder geladen werden. Dabei können Accounts mit der Rolle "Student" lediglich auf ihre eigenen Graphen zugreifen. Accounts mit der Rolle "Admin" haben Zugriff auf alle Graphen.

XIII.2 Installationsanleitung

XIII.2.1 Clientseitige Installation für Anwender

Autor: Erik Heldt

VarG ist eine plattformunabhängige Webanwendung, das heißt man muss nichts lokal auf seinem PC installieren, um sie zu benutzen. Alles was man benötigt, ist ein moderner Web-Browser und eine Internetverbindung (Browser-Empfehlung: Google Chrome oder Firefox). Öffne den Browser und gib in der URL-Leiste `https://sam.imn.htwk-leipzig.de` ein. Nun befindest du dich im Home-Menü von VarG und kannst loslegen!

XIII.2.2 Clientseitige Installation für Entwickler

Autor: Erik Heldt

Um die Anwendung im Entwicklungszustand ausführen zu können, musst du Node.js und npm auf deinem PC installieren. Wie das geht erfährst du hier: `https://www.npmjs.com/get-npm`.
Node.js ist eine JavaScript-Entwicklungsumgebung, die benötigt wird, um die Anwendung samt der genutzten Frameworks und Bibliotheken kompilieren und ausführen zu können. Node Package Manager, oder kurz npm, wird mit Node.js mitgeliefert und verwaltet alle installierten Pakete, die beim Bauen des Programms verwendet werden.

Weiterhin wird das Versionsmanagement-Tool Git benötigt. Den Download dafür gibt es hier: `https://git-scm.com/downloads` bzw. für Windows-Nutzer wird die Git-Bash empfohlen: `https://gitforwindows.org`.

Sind diese Tools nun installiert, musst du dir das VarG-Repository von GitLab auf deinen PC herunterladen bzw. "klonen". Um vollständigen Zugriff auf dieses Repository zu haben, musst du im GitLab dem Projekt zugeordnet sein. Besitzt du also die entsprechenden Rechte, navigiere im Terminal in einen Ordner auf deinem Rechner, in dem du das Projekt speichern willst, und gib dort den Befehl

```
git clone https://gitlab.imn.htwk-leipzig.de/weicker/varg.git
```

ein. Warte, bis das Herunterladen abgeschlossen ist, und öffne dann den neu erschienenen Ordner "varg" in einem Code-Editor (empfohlen wird Visual Studio Code).

Es sollten dort mehrere Ordner zu sehen sein, unter anderem der "code"-Ordner. Darin ist der gesamte Quellcode des Projekts enthalten. Du solltest dich also zur Ausführung des Programms immer in diesem Ordner aufhalten. Um nun den Code zu kompilieren und als Entwicklungsversion auszuführen, folge bitte diesem Tutorial: VarG-Installation von Linus Herterich (klickbarer Link, Anleitung nur für VSCode).

XIII.2.3 Serverseitige Installation

Autor: Linus Herterich

Login-Daten

Im folgenden werden die Login Daten für den aktuellen Server, welcher unter der Adresse `https://sam.imn.htwk-leipzig.de` erreichbar ist, genannt. Die Installationsanleitung ist aber auch so formuliert, dass sie auf anderen Servern nachgestellt werden kann.
SSH-Befehl (für z.B. Git-Bash), um auf den Server per Remote zuzugreifen (nur im HTWK Netz oder per HTWK-VPN möglich):

```
ssh root@sam.imn.htwk-leipzig.de
```

Login-Daten für SSH:

```
User: 'root' | Passwort: 'zuwinket3771{Harne'
```

Login-Daten für die MySQL Datenbank:

```
User: 'root' | Passwort: 'l_GD6P67+V' | Datenbank: 'vargdb'
```

Webserver und SSL

Für die serverseitige Installation des Projekts wird ein SSL-zertifizierter Webserver benötigt. In unserem Fall haben wir Apache 2.4 verwendet, um das Frontend live zu schalten. Es sind aber auch andere gängige HTTP Webserver möglich. Wenn andere Webservertechnologien verwendet werden, muss folgende Anleitung beachtet werden, damit das Vue-Routing funktioniert: `https://router.vuejs.org/guide/essentials/history-mode.html`. Für die Installation eines SSL-Zertifikats haben wir den certbot (`https://certbot.eff.org/`) verwendet.
ACHTUNG: Das SSL-Zertifikat muss im Sommer 2021 verlängert werden, damit die Webanwendung weiter problemlos funktioniert (Anleitung hierfür findet sich auf der certbot Webseite).
Am Ende der Datei

```
varg/docker/node.js/api.js
```

müssen die SSL ".perm" Zertifikate verlinkt werden, damit die API auch auf die SSL-Zertifikate zugreifen kann und somit eine verschlüsselte Kommunikation zwischen Frontend und Backend stattfinden kann. Derzeit sind bereits die richtigen Pfade eingetragen.

Klonen des Projekts

Sind die Grundvoraussetzungen gegebenen, kann das Projekt auf dem Webserver geklont werden. Hierzu muss sich zunächst auf dem Server eingeloggt werden (Anmeldedaten siehe oben). Als nächstes muss mit den Befehl

```
cd /var/www/html
```

in das Standard Webserver-Verzeichnis von Apache navigiert werden. Dort wird anschließend das Projekt mit dem Befehl

```
git clone https://gitlab.imn.htwk-leipzig.de/weicker/varg.git
```

geklont. Nun muss ein GitLab-Nutzername und -Passwort eingegeben werden, damit das Projekt in den Ordner "varg" geklont werden kann. Will man dies umgehen, kann auch ein SSH-Schlüssel generiert werden und bei einem GitLab Account hinterlegt werden (weitere Details: `https://docs.gitlab.com/ee/ssh/`)
Nun wurde ein Ordner mit folgendem Pfad angelegt, indem die Projekt-Daten sind:

```
/var/www/html/varg
```

Navigiert man per "cd" in den Ordner, so kann dort das Projekt aktualisiert (mit "git pull") oder andere Branches ausgewählt werden (mit "git checkout ...").

Unterschiede Entwicklungsversion und Produktionsversion
Im Code sind einige Stellen zu ändern, damit das Projekt auf dem Server lauffähig ist. Am besten wird hierfür ein neuer Branch erstellt, welcher zwar auf dem aktuellen Projektstand ist, aber die Server-Änderungen beinhaltet. Dieser Branch wird anschließend auf dem Server gepullt (wir nannten diesen Branch immer "prod").
Bei den Änderungen handelt es sich zum einen um IP-Adressen, die zum API-Docker-Container zeigen, aber auf der Live Version zur Live-API-Adresse zeigen müssen.

- Mit dem "Suchen und Ersetzen"-Tool im IDE (in VSCode: View -> Search) müssen all diese IP-Adressen ausgetauscht werden, die für die HTTP-Requests verwendet werden:

  ```
  'http://192.168.99.101:1110'
  (Kann auch abweichen. Bitte selbst in den Dateien (bspw. SettingsAccount.vue)
   überprüfen, an welche Domain alle Axios-Requests gehen)
  ```

  in allen Dateien zur Live-API Adresse ändern:

  ```
  'https://sam.imn.htwk-leipzig.de:7070'
  ```

  Wichtig: Die Endungen (z.B. .../VarG/graph/meta) dürfen nicht verändert werden und müssen an der Adresse angehängt bleiben (außer man ändert auch die Express-Routen in der API).

- Des Weiteren müssen die Datenbank-Zugangsdaten in folgender Datei geändert werden:

  ```
  /var/www/html/varg/docker/node.js/api.js
  ```

  Die Zugangsdaten sind am Anfang in einer Konstanten (namens config) abgespeichert und müssen auf folgende Daten geändert werden:

  ```
  host: 'localhost',
  user: 'root',
  password: 'l_GD6P67+V',
  database: 'vargdb'
  ```

- Weiterhin muss in api.js folgende Zeile geändert werden:

  ```
  res.header("Access-Control-Allow-Origin", "http://localhost:8080");
  ```

  zu

  ```
  res.header("Access-Control-Allow-Origin", "https://sam.imn.htwk-leipzig.de");
  ```

- Außerdem muss am Ende der Datei der Code-Block

  ```
  api.listen(8080, () => {
    console.log('API listens to 8080');
  });
  ```

  durch folgenden Code ersetzt werden:

  ```
  https
    .createServer(
      {
        key: fs.readFileSync("/etc/letsencrypt/live/sam.imn.htwk-leipzig.de/privkey.pem"),
        cert: fs.readFileSync("/etc/letsencrypt/live/sam.imn.htwk-leipzig.de/cert.pem"),
      },
      api
    )
    .listen(7070);
  ```

  Damit dies funktioniert müssen eventuell am Anfang der Datei folgende Module importiert werden:

  ```
  const express = require("express");
  const mysql_driver = require("mysql");
  const fs = require("fs");
  const https = require("https");
  ```

Wie bereits erwähnt sollten all diese Änderungen auf einem parallelen Branch durchgeführt und anschließend gepusht werden. Im Anschluss kann das Projekt mit den Live-Änderungen auf dem Server gepullt werden.

Installation Produktions-Frontend
Sobald alle nötigen Produktions-Anpassungen durchgeführt sind und das Projekt im richtigen Ordner geklont wurde, kann mit npm das Projekt kompiliert werden. Zunächst muss in folgenden Ordner navigiert werden:

```
cd /var/www/html/varg/code
```

anschließend wird folgender Befehl ausgeführt (Voraussetzung ist eine LTS-Version von npm auf dem Web-Server):

```
npm install
```

und im Anschluss:

```
npm run build
```

Nun wurden die kompilierten Dateien in den Ordner

```
/var/www/html/varg/code/dist
```

abgelegt. Apache (oder andere HTML Server Technologie) muss so konfiguriert werden, dass der DocumentRoot in diesen Ordner zeigt (auf dem aktuellen Server ist das bereits eingestellt). Wird mit Apache gearbeitet, muss zudem sichergestellt sein, dass "mod rewrite" aktiviert ist (siehe `https://wiki.ubuntuusers.de/Apache/mod_rewrite/`). Sonst wird die ".htaccess" Datei im "dist" Ordner nicht richtig gelesen und die Navigation zwischen einzelnen Seiten funktioniert nicht richtig.
Ist alles problemlos abgelaufen, sollte nun das Frontend unter der URL `https://sam.imn.htwk-leipzig.de` erreichbar sein.

Installation MySQL
Auf dem Webserver muss eine aktuelle Version von MySQL laufen. Es kann folgende Anleitung zur Installation verwendet werden: `https://wiki.ubuntuusers.de/MySQL/` . Es ist darauf zu achten, dass als root Passwort

```
'l_GD6P67+V'
```

gewählt wird. Ansonsten muss ein anderes Passwort in der "api.js" Datei (siehe oben) eingetragen werden.
Sobald MySQL installiert ist, kann über das "Adminer"-Tool, welches beim Frontend unter der URL `https://sam.imn.htwk-leipzig.de/adminer.php` erreichbar ist, eine Datenbank mit dem Namen "vargdb" angelegt werden. Um die initialen Tabellen (evtl. mit Beispieldaten) anzulegen, kann im Anschluss folgende Datei in die Datenbank per Adminer importiert werden:

```
varg/docker/mysql/dump.sql
```

Nun ist die Datenbank eingerichtet und kann über die API angesteuert werden.

Falls der aktuelle Server weitergenutzt wird, ist bereits MySQL mit der aktuellen Datenbank installiert. Um auf die MySQL Datenbank per Terminal zuzugreifen, muss sich per SSH eingeloggt werden und anschließend folgender Befehl eingegeben werden:

```
mysql -uroot -p
```

Nun wird das MySQL passwort gefordert. Das aktuelle Passwort ist:

```
l_GD6P67+V
```

API-Server starten
Der API-Server basiert auf der "Node.js" Technologie. Damit die API funktioniert, muss Node.js auf dem Server installiert sein. Die API kann dann mit dem Befehl:

```
node /var/www/html/varg/docker/node.js/api.js
```

gestartet werden. Man sieht nun den Log der API. Bei jeder Anfrage wird nun eine Zeile ausgegeben. Wenn es Probleme mit der Verbindung zur MySQL Datenbank gibt, werden diese hier angezeigt.

API-Server dauerhaft laufen lassen
Ein Problem ist, dass die API nur solange läuft, solange auch das Terminal geöffnet ist, in dem der Befehl aufgerufen wurde. Da aber die API immer laufen soll, kann die Technologie "forever" verwendet werden (`https://www.npmjs.com/package/forever`).
Mit folgendem Befehl kann die API dauerhaft gestartet werden:

```
forever start -o out.log -e err.log /var/www/html/varg/docker/node.js/api.js
```

Für "out.log" und "err.log" können auch andere Namen oder Pfade verwendet werden. Es handelt sich hierbei um den Output bzw. Error-Meldungen, die ansonsten über das Terminal ausgegeben worden wären.
Mit folgendem Befehl kann nun angezeigt werden, ob die API (noch) mit forever läuft:

```
forever list
```

Die dort angegebene Liste beinhaltet neben dem Namen der ausgeführten API-Datei auch eine ID. Diese kann verwendet werden, um mit folgendem Befehl die API zu stoppen:

```
forever stop [ID]
```

Projekt Live (Produktions-Version) aktualisieren
Wenn neue Features auf dem Server installiert werden sollen, so muss nach der Entwicklung zunächst der neu-entwickelte Branch auf einem neuen (Produktions-) Branch so angepasst werden, wie es oben beschrieben ist (IPs austauschen etc.). Dieser geänderte Branch wird anschließend per "git pull" im Verzeichnis "/var/www/html/varg/" auf dem Web-Server heruntergeladen.
Als nächstes kann die neue Version kompiliert werden (siehe Installation Produktions-Frontend).
Wenn Änderungen an der API-Logik vorgenommen wurden, so muss die API mit dem "forever stop" Befehl gestoppt und anschließend neu gestartet werden.
Falls an der MySQL Datenbank etwas grundlegendes geändert wurde, so kann die Datenbank mit dem Adminer Tool (`https://sam.imn.htwk-leipzig.de/adminer.php`) bearbeitet werden.

Vergessenes Passwort eines Users ändern
Wir verwenden eine programmeigene User-Datenbank zur Verwaltung des Logins und der damit verbundenen User-Accounts (Tabelle "userreg" in der MySQL DB). Da diese sehr simpel implementiert ist und keine E-Mail zur Anmeldung erforderlich ist, konnten wir keine Frontend-Funktionalität für "Passwort vergessen" einbauen (Passwort ändern ist natürlich möglich, aber dafür wird das derzeitige Passwort benötigt). Wenn ein User mal sein Passwort vergisst, muss er sich an einen Backend-Admin mit Zugriff auf Adminer wenden. Im Folgenden wird beschrieben wie der Admin das Passwort für den User manuell ändern kann:

1. Vergewissern, dass es sich wirklich um den Account des Users handelt und er nicht versucht, sich Zugriff auf einen anderen Account zu verschaffen
2. In Adminer einloggen (Login-Daten siehe oben)
3. Links auf "SQL-Kommando" klicken
4. In dem Fenster folgenden Befehl eingeben:

```
SET @userName = "...";
SET @password = "...";
UPDATE userreg
  SET password = AES_ENCRYPT(@password, UNHEX(SHA2(@password, 512)))
  WHERE userName = @userName;
```

5. In der 1. Zeile die drei Punkte ersetzen durch den (exakten!) Namen des Users, dessen Passwort geändert werden soll (kann aus der userreg-Tabelle kopiert werden)
6. In der 2. Zeile die drei Punkte ersetzen durch ein neues temporäres Passwort, welches der User für seinen nächsten Login benutzen kann

7. Alles andere so lassen und auf "Ausführen" klicken - wenn es geklappt hat sollte mindestens eine grüne Nachricht mit dem Text "Abfrage ausgeführt, 1 Datensatz betroffen." erscheinen

8. Dem User mitteilen, dass er nach dem ersten Login mit dem neuen Passwort bitte sofort in seine Account-Einstellungen gehen und das temporäre Passwort in sein eigenes ändern soll

XIII.3 Software-Lizenz

Autor: Linus Herterich

Im Folgenden werden die verwendeten Bibliotkeken und deren Lizenz aufgelistet:

- Vue.js - MIT License: Copyright (c) 2013-present Yuxi Evan You
- vuetify - MIT License: Copyright (c) 2016-2020 John Jeremy Leider
- cytoscape - MIT License: Copyright (c) 2016-2020, The Cytoscape Consortium.
- cytoscape-node-html-label - MIT License: Copyright (c) 2017 Kalugin Sergey
- cypress - MIT Licence: Copyright (c) 2015 Cypress.io, LLC
- jest - MIT Licence: Copyright (c) Facebook, Inc. and its affiliates.
- axios - MIT License: Copyright (c) 2014-present Matt Zabriskie
- darkmode.js - MIT License: Copyright (c) 2018 Nickolas
- file-saver.js - MIT License: Copyright (c) 2016 Eli Grey
- file-saver.js - MIT License: Copyright (c) 2016 Eli Grey

Da ausschließlich die MIT Lizenz verwendet wurde, werden wir auch die Software "VarG" unter der MIT-Lizenz veröffentlichen.

VarG-Lizenz:

XIV. Projektabschluss

XIV.1 Protokoll der Abnahme und Inbetriebnahme beim Kunden

Autor: xxx
XXX

XIV.2 Präsentation auf der Messe

Autor: Erik Heldt
VarG Messepräsentation (klick)

XIV.3 Abschließende Einschätzung durch Product-Owner

Autor: Manuel Eckert

Aus der Sicht des PO war das Projekt "VarG " ein voller Erfolg! Unser Kunde war mit dem Produkt sehr zufrieden und war begeistert über die Anzahl der umgesetzten Features.
Allerdings war das Softwareprojekt, für das gesamte Team, ein Lernprozess. Anfangs sind die Planning- und Review-Treffen ein sehr träge und einseitig gelaufen. Sowohl ich als PO, als auch die Entwickler wussten nicht genau welche Rolle sie exakt ausfüllen. Eine merkbare Änderung haben wir alle dann schon im zweiten Sprint festgestellt. Dies hat uns einen großen Schub für das Projekt gegeben. So hat man mit Beendigung eines jeden Sprints, eine deutliche Steigerung in der Produktivität und Qualität der Ticketbearbeitung gespürt. In den letzten Sprints vor Projektabschluss haben die Entwickler viel schneller verstanden, was die einzelnen User-Stories umfassen und was es braucht, um diese abzuschließen. Dies hat meine Arbeit als PO sehr vereinfacht und ich konnte meine Energie auf andere Arbeitsbereiche im Projekt konzentrieren.
Rückblickend war es exzellent, dass sich einzelne Subteams innerhalb des Entwicklerteams geformt haben, welche sich auf einzelne Bereiche innerhalb des Projektes fokussierten. Der Austausch zwischen den einzelnen Teams war aber trotzdem stets durch die Daily Treffen gegeben. Damit gab es Experten auf den unterschiedlichen Bereichen, welches sich sehr Positiv auf des Projekt ausgewirkt hat.
Die Kommunikation mit unserem Kunden verlief meist sehr gut. Anfangs war es nicht einfach ein gemeinsames und gleiches Verständnis für die Anforderungen zu entwickeln. Durch genauere und stärkere Auseinandersetzung mit dem Thema ist es uns aber sehr gut Gelungen die Anforderungen des Kunden zu treffen. Eine große Hilfe war dabei auch, dass wir, zum Start des SS2020, das Produktinkrement nach jedem Sprint auf einen Webserver gespielt haben. Somit hat der Kunde jederzeit die Möglichkeit das Produktinkrement zu testen. Dies ermöglichte dem Kunden genaueres Feedback zu geben.
Über die gesamte Laufzeit des Projektes ist festzuhalten, dass sich die Anforderungen des Kunden entwickeln und verändert haben.
Das Projekt war ein großer Zugewinn für jeden einzelnen von uns. Eine so intensive und freie Auseinandersetzung mit einem Produkt ermöglicht es, wichtige neue Fähigkeiten für das Arbeiten als Teil eines Teams zu entwickeln. Jeder in seiner Rolle konnte einen großen Mehrwert aus diesem Projekt ziehen.

XIV.4 Abschließende Einschätzung durch Software-Architekt

Autor: xxx
XXX

XIV.5 Abschließende Einschätzung durch Team-Manager

Autor: Alex Hofmann

Das Team zeigte rückblickend eine engagierte und couragierte Leistung über das gesamte Projekt hinweg, so dass auch der Ausfall zweier Teammitglieder locker kompensiert werden konnte.
Der Start verlief noch etwas holprig, da eine gewisse Eingewöhnung sowohl in den Scrum-Prozess als auch auf technischer Seite nötig war, danach lief es aber umso besser.
Durch die Aufteilung des Teams in kleinere Sub-Teams hatte jedes Mitglied über das gesamte Projekt hinweg seinen eigenen klar definierten Arbeitsbereich. Diese Teamstruktur hat dem Projekt im Arbeitsablauf in meinen Augen einen deutlichen Aufschwung gegeben.
Auch das präsenzfreie Semester stellte keine Hürde dar, ganz im Gegenteil. Der Einsatz des Tools Discord belebte Dank der Möglichkeit des Screensharings die Meetings und brachte neue Möglichkeiten in Sachen Pair-Programming.
Auch das Klima innerhalb des Teams war stets angenehm, so dass es nie zu Konflikten untereinander kam bzw. Kritik konstruktiv diskutiert wurde.
Es hat Spaß gemacht, mit Team VarG zusammenzuarbeiten und auch das Resultat kann sich definitiv zeigen lassen!