SCRUM-Dokument2

rlm1-04: Interactive Room Map Display

interactive room map display

Studiengang:	Informatik
Autor:	Marc Rudolf Fuhrer, Nicolas Anthony Waser, Nicola Sacha Eigel
Betreuer:	Prof. Dr. Michael Röthlin
Datum:	14.01.2023

Inhaltsverzeichnis

1 Versionshistorie

Version	Datum	Autor	Änderung
1.0	14.01.2023	Alle	1. Fassung «rlm1-04 - InteractiveRoomMapDisplay - SCRUM-Dokument2 - EigelFuhrerWaser.docx»
…			

2 Rückblick Project Setup

2.1 Erfassung der Ausgangssituation

Begonnen bei den ursprünglich genannten Anforderungen, wurde das Team rlm1-04 dazu angehalten eine Web-Applikation mit definiertem Technologie-Stack zu entwerfen, deren Ziel es war Messdaten dreier anderer rlm1-Teams auf selbstentworfenen Gebäudekarten (mindestens SIPBB) und eigenem Koordinatensystem zusammenzutragen und einem BFH-Besucher nutzenbringend zur Verfügung zu stellen.

Nach einigen vertiefenden Absprachen mit dem Product Owner und sämtlicher beteiligter rlm1-Teams wurden diese Anforderungen sowohl konkretisiert, deren Fokus geshiftet, als auch teilweise ganz erneuert.

Stand der Ausgabe dieses Dokuments lautet die Ausgangslage des rlm1-04-Teams wiefolgt:
Zu entwerfen ist ein «Interactive Room Map Display» Service (genannt Service), welcher auf einem Low-Level-Interface fusst. Der Service ist wie angedacht webbasiert und bietet drei Schnittstellen:

1. Konsole
2. REST API
3. Browser-basierter Web-Zugriff

Sämtliche von uns abhängige rlm1-Teams gelten als «Publisher», da Sie Ihre Mess-Daten und die gesamte Darstellung als HTML über die REST-Schnittstelle publizieren.
Benutzern, genannt «Subscribers», soll es möglich sein, sich für Ihre gewünschten Gebäudeteile einzutragen und die damit verbundenen Informationen in folgenden Formaten zu konsumieren:

1. HTML (Darstellung in Browser)
2. JSON (Weiterverarbeitung)
3. PDF (Download)

Weiter ist der Zugriff über statische URIs zu gewährleisten, die Geschwindigkeit des Bildwechsels wählbar zu halten und Metadaten zu den Inhalten bereitzustellen.

Der Fokus des rlm1-04-Teams liegt auf der REST API, der Verifizierung der von Publishern bereitgestellten Daten und der ordentlichen Rechteverwaltung.

2.1.1 Problem

Zuständigkeiten innerhalb rlm1-Teams, Zeitdruck und begrenzte Verfügbarkeit dürften sämtliche Beteiligten vor eine Herausforderung stellen.

2.1.2 Chance

Eine Kompromissfindung zwischen PO, rlm1-Teams und rlm1-04-Team, welche den Erfolg einer jeden Partei, nicht zuletzt dem direkten und indirekten User des Services garantiert.

2.1.3 Zwang

Innert 5-wöchiger Frist war die Veröffentlichung einer versionierten API zu gewährleisten, welche unter Berücksichtigung der damaligen Anforderungen von Statten ging.

2.2 Themen-Analyse

Der Service wurde definiert, als multifunktionale Schnittstelle für sämtliche primären Stakeholder. Das Projektteam kombinierte den ursprünglich erhaltenen Anforderungskatalog, den später erhaltenen LowLevel-Anforderungskatalog, die Anforderungskataloge der projektverwandten Entwicklerteams und berief zu diesem Zweck u.A. 2 Abstimmungsmeetings mit sämtlichen rlm1-Entwicklerteams ein.

Ziel war es einen Service zu designen, der dem Lifestyle einer Person, die Nutzer eines BFH-Gebäudes ist, zuträglich ist und ihm/ihr Informationen bereitstellt, die ihn/sie in seinem/ihrem Alltag unterstützt.

2.3 Stakeholder (primäre aufgeführt)

2.3.1 PO (Product Owner)
Der PO, der sowohl als Repräsentant der projekt-bewertenden Instanz, als auch Vertreter der Interessen des effektiven Endbenutzers und Unterhalters des Endprodukts fungiert, ist daran interessiert eine ausbaufähige Lösung zur Live-Erfassung und Darstellung der Kombination von Gebäudeplänen und verschieden gearteter Sensor-Daten als Produkt dieser Arbeit zu sehen.
Er wünscht einen Service, der auf bewährten PHP-Komponenten basiert und einfach zu installieren, konfigurieren, bedienen und warten ist. Um eine gewisse User Experience zu gewährleisten soll sowohl eine vernünftige User- und Rechteverwaltung vorhanden sein, als auch eine vereinfachte Fehlerbehandlung durch freizügiges Logging umgesetzt sein.

2.3.2 rlm1-04-Entwicklerteam
Das Entwicklerteam möchte eine Software-Umsetzung, die möglichst sämtliche an Sie herangetragenen Anforderungen erfüllt, oder Sie bei allfälliger Weiterführung des Projekts über den Umfang des Moduls hinaus, durch Weiterverfolgung des gewählten Pfades, zu erfüllen vermag.

2.3.3 Service Administrator
Der Service Administrator wünscht sowohl umfassende User- und Publisher-Konfigurationsverwaltung, als auch die Verwaltung von Beziehungen dieser beiden Teilnehmer per Konsole. Weiter soll die Subscriber-Verwaltung restriktiv über IP-Addressbereiche (inkl. Wildcard) und Authentisierung umgesetzt sein. Auch alternative Zurverfügungstellung von Inhalten für unangemeldete User soll umgesetzt sein. Desweiteren sind Nutzungsdaten von Publishern und Subscribern zur Optimierung gefordert.

2.3.4 Publisher
Die Publisher sind darauf angewiesen, dass eine praktikable Schnittstelle zum Service vorzufinden ist, bei der Sie ausreichend Freiraum bzgl. vollständiger Übermittlung Ihrer Daten und deren Sicherheit haben. Realisiert wird dies über PUT-Requests via REST API per vom Service Administrator erstellte User(-Gruppen). Ordentliche HTTP-Statusmeldungen sind unabdingbar. Sorgfältige Prüfung von Berechtigungen bzgl. Überschreiben von Daten anderer Publisher ist ebenfalls Voraussetzung. Ein Publisher möchte auch wissen, wie oft seine Daten abgerufen wurden, um seine Nutzung des Services sukzessive zu verbessern. Vorerst begnügt sich ein Publisher mit einem Low-Level-Interface, welches ihm erlaubt eine View in Form einer HTML-Seite zur Verfügung zu stellen. In einem weiteren Schritt ist ein High-Level-Interface angedacht, wie es zu Beginn des Moduls vorgegeben wurde.

2.3.5 Subscriber
Die Subscribers nutzen den Service, um sich die von den Publishern veröffentlichten Daten und die dazugehörigen Meta-Daten anzuzeigen. Subscribers sollen authentifiziert werden und möchten selbst bestimmen welche Informationen sie abonnieren möchten. Diese werden auf drei Arten abgerufen:

1. HTML, um Sie in einem Webbrowser mit selbstgewählter Frequenz als Slideshow anzuzeigen
2. JSON, um Sie in anderen Applikationen weiterzuverwerten
3. PDF als Download

2.4 Organisation

Das Projektteam nutzte die Verwaltung der Tasks in Jira, um individuelle Arbeiten koordiniert zu erledigen. So wurde der Voraussetzung Genüge getan, dass jedes Team-Mitglied zu unterschiedlichen Zeiten und mit dem üblichen Schulaufwand verträglich individuell seine Arbeitsleistungen erbringen konnte. Zudem verhalf die zentralisierte Nachvollziehbarkeit auf dieser Plattform dem Team zu einer jederzeit verfügbaren Übersicht über den Stand des Projekts.

Weiter wurde darauf geachtet, dass die Weekly-Scrums jeweils zu regelmässigen Zeiten abgehalten wurden, sprich dienstags um 15:00 bis 16:00 Uhr via MS Teams. Zu diesen Sitzungen wurde der PO (Prof. Dr. Michael Röthlin) bei Bedarf eingeladen.

Die Daily-Scrums wurden anfangs täglich in tabellarischer Form niedergeschrieben, bevor dies durch eine Optimierung, die sich durch Sprint-Retrospektiven ergab, abgeändert wurde. Fortan wurde jeweils ausschliesslich bei einer Arbeitsleistung eingetragen (mit Verlinkung zu Jira-Tasks und anderweitig daraus entstandenen Dokumenten), um einen schneller verfügbaren Überblick zu gewährleisten.

Im Laufe des Projekts entstand eine rege Instant-Messaging-Kultur, die das Projektteam zu schätzen wusste (zusätzlich zu task-bezogener Nutzung der Kommentar-Funktion in Jira).

3 Review

3.1 Produktziel

Es wurden stand heute sämtliche Anforderungen umgesetzt, bis auf:

- Zugriff über IP-Ranges
- MetaDaten über PublisherData verfügbar machen
- Abrufen der Daten als JSON und PDF
- Subscription anpassen durch Subscriber

3.2 Abgrenzung

Fokus des rlm1-04-Projektteams war nach Anforderungs-Änderungen ein stabiles Backend, welches als Schnittstelle zwischen Publishern und Subscriber dient.
Diese Schnittstelle zeichnet sich durch einen gewissen Polymorphismus aus, da sie einerseits konfigurierbar per Konsole, abfüllbar per HTTP-Requests und andererseits Hüter eines Berechtigungssystems zu sein hat.

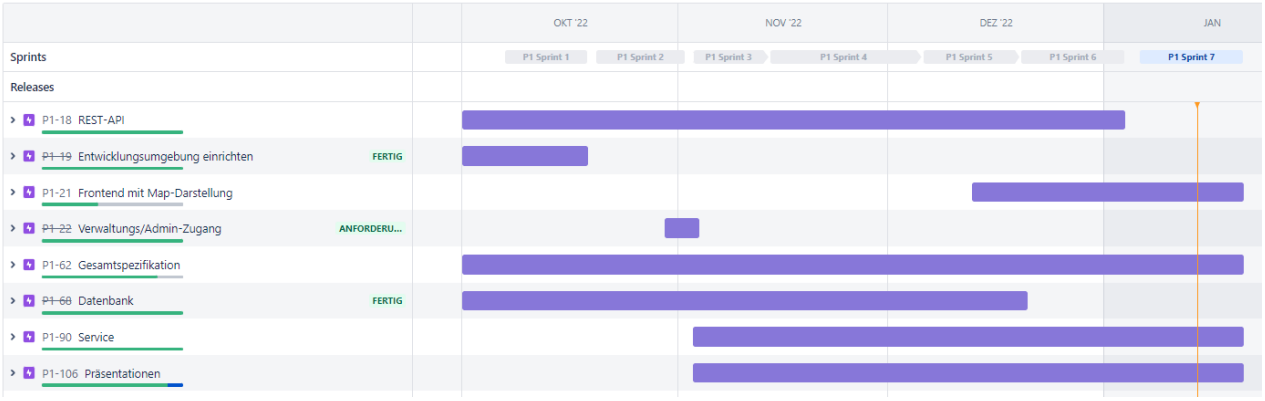
Als nicht im Scope wurden die HTML-Views der Publisher definiert und logischerweise die Datenerhebung.

3.3 Lieferobjekte

Mit dem PO und den Publisherteams wurde vereinbart, dass ein laufendes Deployment auf dem BFH-Webserver stattfindet. Zur praktikablen Einsicht in die Funktionsweise der REST-Schnittstelle wurde eine SwaggerUI erstellt. Bei jeder grösseren Änderung wurden sämtliche Publisher-Teams per Mail informiert und in Form von Release-Notes über die neusten Funktionalitäten unterrichtet.
Zum Projektende hin wurden für sämtliche Stakeholders User erstellt, die Zugänge dazu per Mail verschickt.

3.4 Backlogs

Der Backlog wurde initial mit den dringendsten Tasks gefüllt und in einem zweiten Schritt (jedoch relativ früh) einigen Epics zugeschrieben. Dies garantierte eine Möglichkeit die Backlog-Struktur organisch wachsen zu lassen und doch eine relativ rigide Struktur beizubehalten, an der sich die Mitglieder des Entwicklerteams orientierten.



4 Retrospektive I: SCRUM Method

4.1 Roles

PO (Product Owner): Prof. Dr. Michael Röthlin
Scrum-Master: Nicola Sacha Eigel
Entwicklerteam:

- Nicola Sacha Eigel
- Nicolas Anthony Waser
- Marc Rudolf Fuhrer

Die Rollenverteilung entstand natürlich und war der Organisation äusserst zuträglich, denn Herr Eigel nimmt diese Rolle auch im beruflichen Umfeld wahr.

4.2 Events

Weekly: Wurde online dienstags um 15:00 bis ca. 16:00 Uhr abgehalten. (Reserve Donnerstag abends)
Daily: Zu Beginn täglich in Tabelle eingetragen, später nur bei Leistung aufgrund Retrospektive, dafür Nutzung Instant Messaging und Jira-Kommentare
Sprint-Reviews, -Plannins und -Retrospektiven: Meist direkt im Anschluss an Weekly.
Individuelle Meetings: Entweder vor Ort oder in MS Teams, stets dokumentiert und in Confluence abgelegt.

4.3 Artifacts

Zu den Artefakten wurde vereinbart, direkt die laufenden Auto-Deploys zu zählen. Sämtliche grösseren Releases triggerten einen Mail-Verkehr mit Release-Notes an Publisher und wurden im 2-wöchigen Sprint-Zyklus abgehandelt.

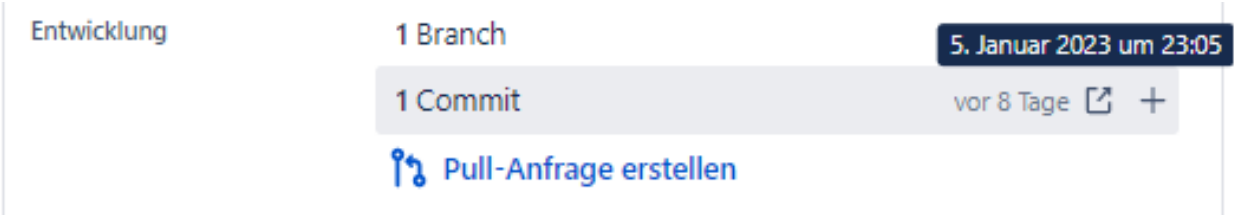
4.4 Other

Das rlm1-04-Projektteam empfand die Sprint-Retrospektive als überraschend nützlich und verbesserte seine Arbeitsläufe dadurch kontinuierlich.
Jira wurde von allen Seiten bereits zu Beginn als zentralen Knotenpunkt akzeptiert und in die Methodik des Teams eingebettet. Einen solchen zentralen Dreh- und Angelpunkt zu haben erwies sich als überaus nützlich.

5 Retrospektive II: Tools & Instrumente

5.1 Jira/Confluence mit Github-Integration

Die Github Integration wurde zu einem mächtigen Helferlein und kam dem Team einige Male zu Gute. Die Möglichkeit direkt Feature-Branches zu erstellen und sich auch als Aussenstehenden einen Überblick über die Code-Base zu verschaffen wurde gerne genutzt.



5.2 Auto-Deployment lokal und Webserver (PLESK)

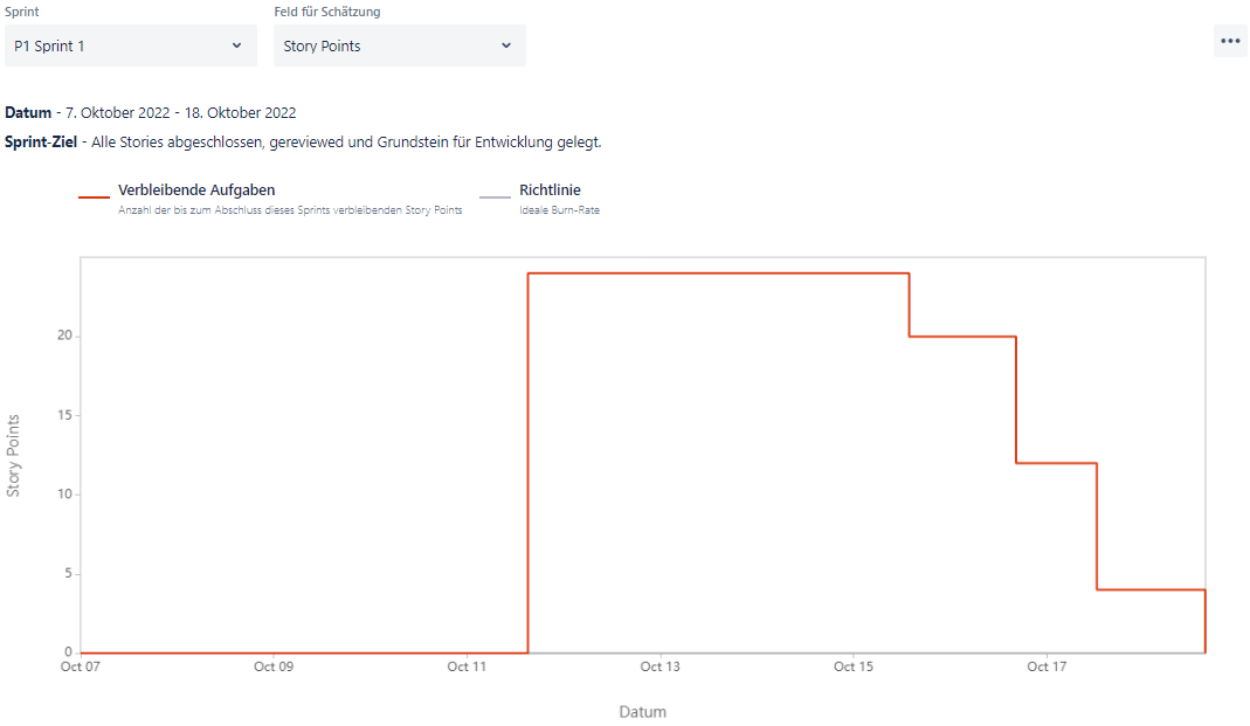
Die Möglichkeit Laravel mit Composer und GIT in PLESK zu integrieren nahm dem Team einiges an Arbeit ab. Dies wäre wiederkehrender Aufwand gewesen, der nur Zeit gestohlen hätte.

5.3 PHP-Storm, Laravel, MySQL

Der genutzte Tech-Stack war zwar nicht auf aktuellstem Stand, jedoch stabil. Laravel bietet seinen Usern (also Entwicklern) eine Bandbreite von Vereinfachungen (man betrachte z.B. Eloquent) und ist auch genug verbreitet, sodass ausreichend Hilfestellungen in Form von Foren, Tutorials und Bücher verfügbar sind.

6 Anhang

6.1 Sprint 1: Burndown Chart mit abgeschlossenen Vorgängen



Abgeschlossene Vorgänge — Im Vorgangsnavigator anzeigen

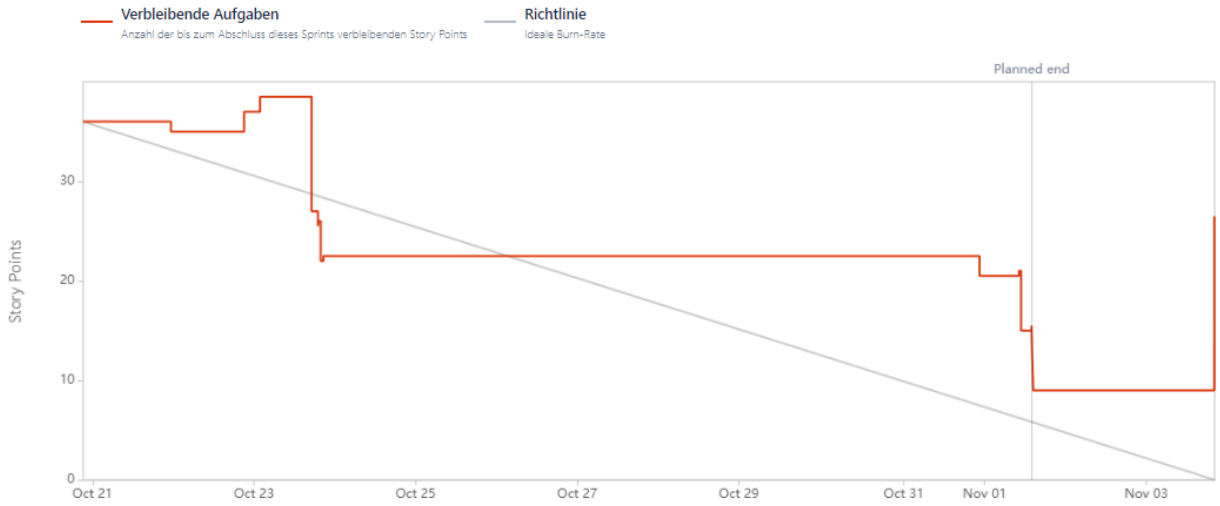
Schlüssel	Zusammenfassung	Vorgangstyp	Epic	Status	Zugewiesene Person	Story Points
P1-3	Entity Relationship Model	Task		FERTIG	NW	8
P1-52	Translation Daten zu Darstellung	Story		FERTIG	NE	4
P1-26	REST Spezifikation	Story	REST-API	FERTIG	NW	8
P1-10	Projektumgebung einrichten	Task		FERTIG		4
P1-20	Lauffähige Umgebung auf BFH-Server (PLESK konfigurieren)	Story		FERTIG		10

6.2 Sprint 2: Burndown Chart mit abgeschlossenen Vorgängen

Sprint: P1 Sprint 2 | Feld für Schätzung: Story Points

Datum - 20. Oktober 2022 - 1. November 2022

Sprint-Ziel - Fixing vom Scope. Planen und Dokumentieren Features, Anforderungen usw. Konzepte für Software vorhanden damit man bald anfangen kann mit Umsetzung.



Abgeschlossene Vorgänge — Im Vorgangsnavigator anzeigen

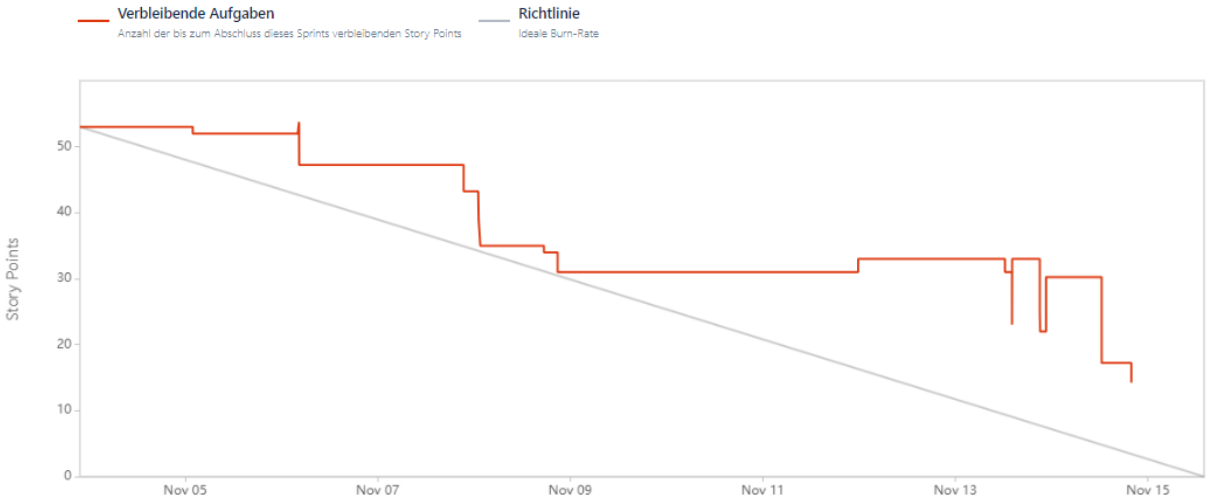
Schlüssel	Zusammenfassung	Vorgangstyp	Epic	Status	Zugewiesene Person	Story Points
P1-64	Systemabgrenzung	Story	GESAMTSPEZIFIKA...	FERTIG	NW	2
P1-63	Dokument aufsetzen	Story	GESAMTSPEZIFIKA...	FERTIG		1.5
P1-61	Anforderungskatalog	Story	GESAMTSPEZIFIKA...	FERTIG		11.5
P1-25	Domain Model Diagramm	Story	GESAMTSPEZIFIKA...	FERTIG	NW	2
P1-67	Personae & Storyboards	Story	GESAMTSPEZIFIKA...	FERTIG		4
P1-27	Sicherheitskonzept REST Benutzer	Story	REST-API	FERTIG	NW	4.5

6.3 Sprint 3: Burndown Chart mit abgeschlossenen Vorgängen

Sprint: P1 Sprint 3 | Feld für Schätzung: Story Points

Datum - 3. November 2022 - 15. November 2022

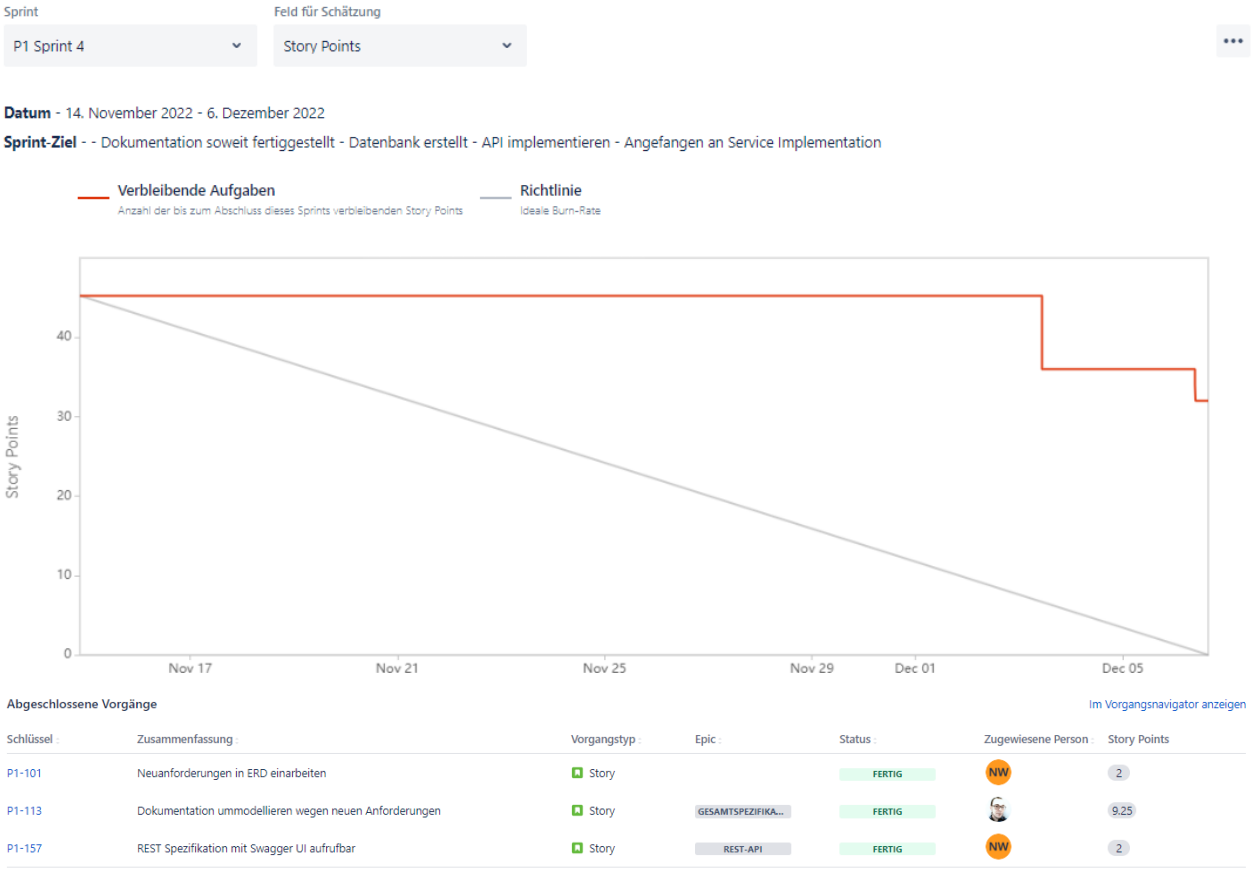
Sprint-Ziel - - Alle nötigen Stories für die Zwischenpräsentation abgeschlossen - Zwischenpräsentation als Hauptziel - Einpflegen der neuen Anforderungen in die diversen Artefakte - Soweit wie möglich Abschliessen der Dokumentationen - ACL Konzept - Bereit für Umsetzungen nach diesem Sprint



Abgeschlossene Vorgänge — Im Vorgangsnavigator anzeigen

Schlüssel	Zusammenfassung	Vorgangstyp	Epic	Status	Zugewiesene Person	Story Points
P1-57	DB Security	Story		FERTIG	NE	8
P1-24	Sequence Diagram	Story	GESAMTSPEZIFIKA...	FERTIG	NE	3
P1-109	SCRUM-Teil aus Dokument in Präsentation	Story	PRÄSENTATIONEN	FERTIG	NE	4
P1-108	SCRUM Dokument Helbling verfassen	Story	PRÄSENTATIONEN	FERTIG		6.5
P1-107	Präsentation Grundgerüst	Story	PRÄSENTATIONEN	FERTIG		1
P1-112	Probedurchlauf	Story	PRÄSENTATIONEN	FERTIG		1
P1-111	Lösungsvorgehen	Story	PRÄSENTATIONEN	FERTIG	NE	4.5
P1-110	Problemstellung	Story	PRÄSENTATIONEN	FERTIG	NW	4
P1-93	ACL Konzept	Story	SERVICE	FERTIG	NE	5
P1-60	Präsentation Zwischenstand und Vorgehen	Story	PRÄSENTATIONEN	FERTIG		3
P1-94	ISDS Konzept	Story	SERVICE	FERTIG		3
P1-102	Überarbeiten REST Spezifikation auf Grund neuer Anforderungen	Story	REST-API	FERTIG	NW	8
P1-133	Versand SCRUM Dokument an Hr. Helbling	Story	PRÄSENTATIONEN	FERTIG		0.25

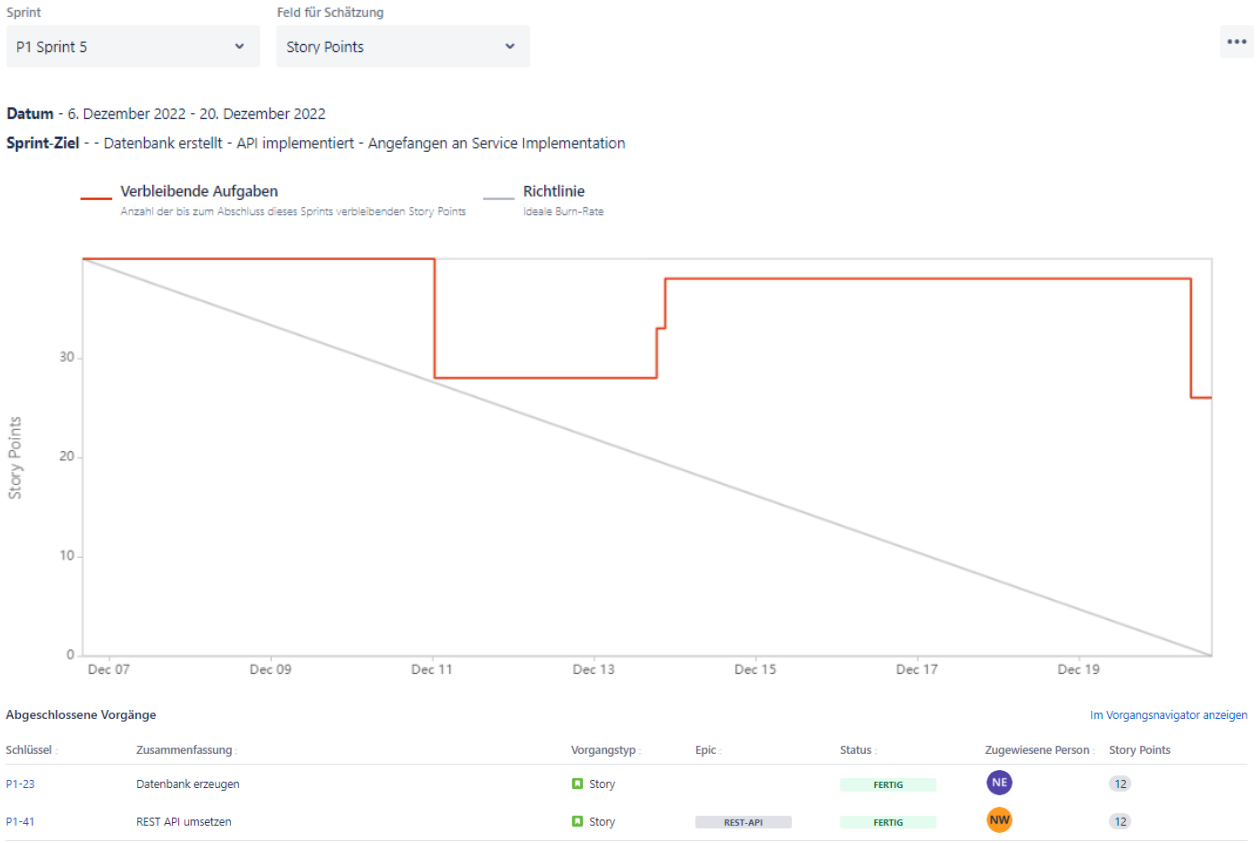
6.4 Sprint 4: Burndown Chart mit abgeschlossenen Vorgängen



Abgeschlossene Vorgänge Im Vorgangsnavigator anzeigen

Schlüssel	Zusammenfassung	Vorgangstyp	Epic	Status	Zugewiesene Person	Story Points
P1-101	Neuanforderungen in ERD einarbeiten	Story		FERTIG	NW	2
P1-113	Dokumentation ummodellieren wegen neuen Anforderungen	Story	GESAMTSPEZIFIKA...	FERTIG		9.25
P1-157	REST Spezifikation mit Swagger UI aufrufbar	Story	REST-API	FERTIG	NW	2

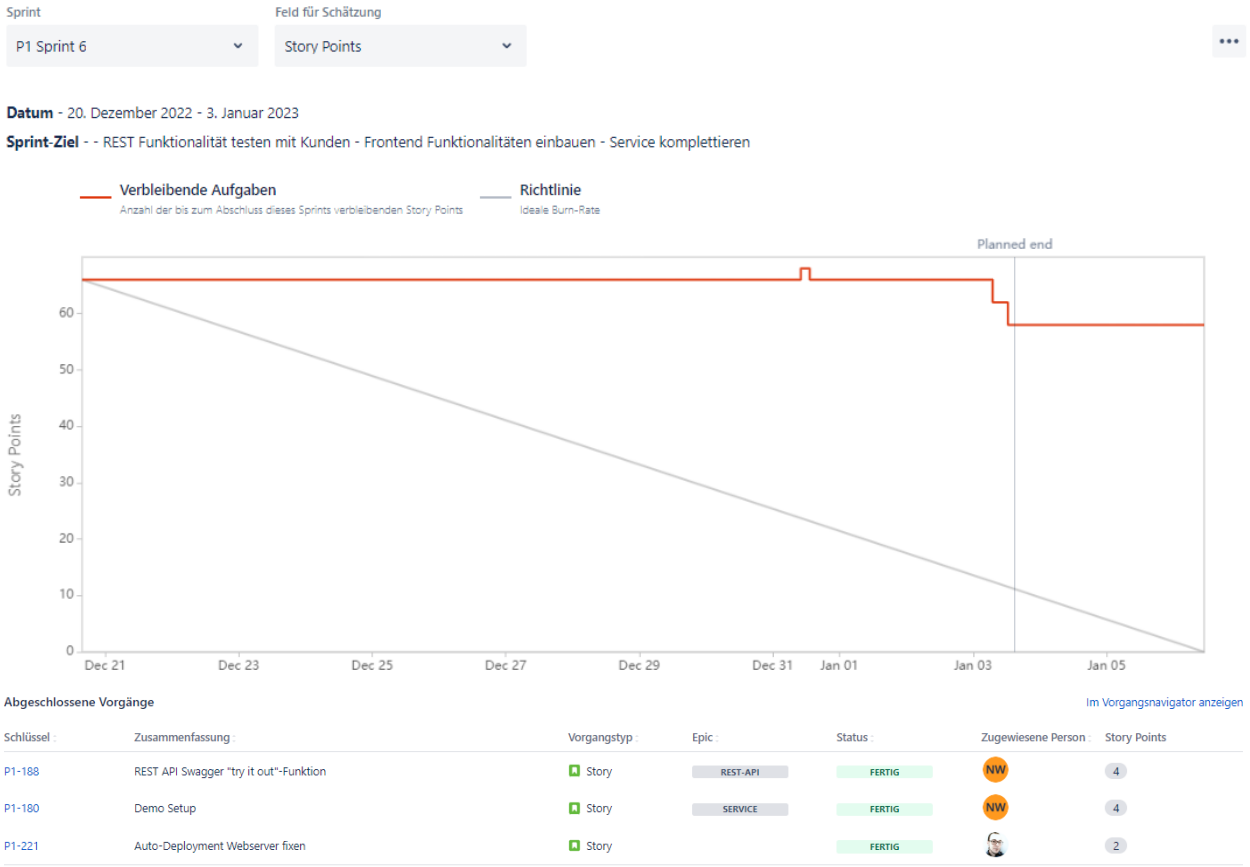
6.5 Sprint 5: Burndown Chart mit abgeschlossenen Vorgängen



Abgeschlossene Vorgänge Im Vorgangsnavigator anzeigen

Schlüssel	Zusammenfassung	Vorgangstyp	Epic	Status	Zugewiesene Person	Story Points
P1-23	Datenbank erzeugen	Story		FERTIG	NE	12
P1-41	REST API umsetzen	Story	REST-API	FERTIG	NW	12

6.6 Sprint 6: Burndown Chart mit abgeschlossenen Vorgängen



Abgeschlossene Vorgänge Im Vorgangsnavigator anzeigen

Schlüssel	Zusammenfassung	Vorgangstyp	Epic	Status	Zugewiesene Person	Story Points
P1-188	REST API Swagger "try it out"-Funktion	Story	REST-API	FERTIG	NW	4
P1-180	Demo Setup	Story	SERVICE	FERTIG	NW	4
P1-221	Auto-Deployment Webserver fixen	Story		FERTIG		2

6.7 Sprint 7: Burndown Chart mit abgeschlossenen Vorgängen (laufend)

Sprint: P1 Sprint 7 | Feld für Schätzung: Story Points

Datum - 6. Januar 2023 - 20. Januar 2023

Sprint-Ziel - Abschluss Projekt, gute Endpräsentation

Abgeschlossene Vorgänge — Im Vorgangsnavigator anzeigen

Schlüssel	Zusammenfassung	Vorgangstyp	Epic	Status	Zugewiesene Person	Story Points
P1-212	Kernel Commands	Story	SERVICE	FERTIG	NE	5
P1-34	Inhalt und Design planen	Story	GESAMTSPEZIFIKA...	FERTIG		3
P1-31	Design Umsetzen	Story	FRONTEND MIT M...	FERTIG		6
P1-91	Backup	Story	SERVICE	FERTIG	NE	2
P1-92	Logging	Story	SERVICE	FERTIG	NE	3
P1-208	Testing fixen	Story	SERVICE	FERTIG	NW	2
P1-95	Umsetzung Publisher	Story	SERVICE	FERTIG	NE	5
P1-213	Installations README.md	Story	GESAMTSPEZIFIKA...	FERTIG	NE	2
P1-214	Publisher and Subcriber Usage data	Story	SERVICE	FERTIG	NE	3
P1-211	ACL Unittests	Story	SERVICE	FERTIG	NE	4