Endbericht Batadu

Projektziele und Aufgabenstellung:

Als wir benachrichtigt wurden, dass wir ein Website Projekt machen sollten, war uns am Anfang nicht sehr klar, was wir machen würden, hauptsächlich aufgrund der beschränkten Zeit: Nur 2 Wochen. Im Endeffekt haben wir uns dann entschieden, ein neues, besseres watten.org zu realisieren. Es ist durchaus das Aufwendigste unserer Vorschläge, aber die Motivation war sehr stark:

- Watten.org hat eine veraltete und nicht ganz angenehme Oberfläche
- Watten.org ist nicht für Mobile Geräte bzw. unterschiedliche Bildschirmgrößen angepasst.
- Jeder von uns wattet gerne, oft auch auf watten.org und von verschiedenen Geräten.

Projektorganisation:

Für unser Projekt haben wir uns für das Vorgehensmodell *Scrum* entschieden, da es sehr stark auf Kommunikation und produktivität setzt. Wir haben uns auch gedacht, dass jeder einen bestimmten Bereich übernimmt, und dass dann die Bereiche verknüpft werden, damit nicht zu viele Konflikte beim Programmieren entstehen, wie bei uns allen schon mindestens einmal früher passiert ist. Die Rollen wurden deswegen so aufgeteilt**:**

- **Daniele Guidotti**: Leitung, Dokumentation, Datenbanken, Kommunikation mit der Datenbank (REST-API), externe Tester suchen und mit ihnen testen, Sonstiges (Wattkarten einscannen, Testserver, …)
- **Daniel Nagler**: Game-Server und Automatische Tests
- **Beniamin Terza**: UI-Design, Frontend-entwicklung und Fronted/Backend Spielentwicklung

Auf dem ersten Blick könnten jetzt die Aufgaben (rein an der Menge) unbalanciert vorkommen, doch wenn man die Schätzung des Zeitaufwands anschaut, ist die Situation anders:

- UI-Design und Frontend: geschätzt über 36 Arbeitsstunden
- Game Server: geschätzt über 30 Arbeitsstunden
- Nicht oben genanntes: geschätzt ca. 30 Arbeitsstunden

Leistungen und Projektergebnisse:

Bei diesem Punkt können wir jetzt schon sagen, dass der geschätzte Aufwand lange nicht eingehalten wurde, da wir von ursprünglich eingeschätzten 100 Arbeitsstunden auf über 140 Arbeitsstunden gekommen sind, wobei die größte Fehleinschätzung beim Frontend war: im Endeffekt wurde daran über 90 Stunden gearbeitet.
Die vorgegebenen Termine wurden trotz früher als geplanter Projektstart nicht vollständig eingehalten: wir hatten dort zwar schon einen funktionierenden Prototypen, es war jedoch noch kein vollständiges Programm. Zur Zeit wo dieses Protokoll geschrieben wird, haben wir einen funktionierenden Programm mit multi-tier-Architektur und mehr oder weniger alle

funktionierenden Funktionen, aber bevor es veröffentlicht wird, ist noch eine intensive Beta-Test-Phase und eine Korrektur von bugs nötig.

Backend:

Im Backend wurde das Meiste mit node.js realisiert. Dort wurden zwei Klassen erstellt, wobei bei einer recht wenig Aufwand gebraucht wurde, aber dafür die andere Klasse schwieriger war, meistens aufgrund von viel Denkaufwand wegen viele Spielregeln und mehrere Spielmodi. Es musste auch ein bestehender socket.io-Code überprüft und verbessert werden, was für den Daniel Neuland war.
Beim Backend wurden auch automatisierte Tests mit node.js geschrieben. Das war auch Neuland, doch mit den Modulen *mocha* und *chai.js* war es trotzdem ziemlich leicht aber auch Zeitaufwendig.

REST-API und Datenbank:

Für die Datenbank haben wir PostgresSQL benutzt, prinzipiell aufgrund der internen Struktur, die bei großen Datenmengen sehr performant sein soll, und auch damit wir etwas Neues ausprobieren. Am Anfang war es nicht sehr leicht, obwohl die ganzen Queries sehr ähnlich zu MySQL sind, weil wir noch im Schema-Konzept hineinkommen sollten, doch nachher ist es recht gut gegangen.
Die REST-API wurde auch mit node.js realisiert, was mit hilfe von *express* und *pg* nicht besonders schwierig war. Das einzig schwierige war dort, Ergebnisse von unterschiedlichen Queries asynchron einzusammeln und zu verarbeiten. Ein weiteres, kleines Problem war, dass es bei den .JSON-Übergaben zwei Arten gibt, die einzusammeln, und eine wird von *AXIOS* unterstützt, während die andere nicht, und somit musste beim Verbinden vom Frontend mit der API, die API kurz umgeschrieben werden weil wir das nicht im vorhinein besprochen haben.

UI-Design und Frontend:

Beim UI-Design wurde Adobe XD benutzt, wo wir uns am grundsätzlichen Layout von watten.org inspiriert haben. Beim Designen wurde überall der gleiche Stil verwendet, und es wurde versucht, eine moderne Oberfläche zu erstellen.
Bei der Frontend-Entwicklung haben wir mit dem Erstellen der HTML-Seiten, die mit React und Tailwind gemacht wurden. Tailwind, im Gegensatz zu React wurde bei uns zum ersten Mal benutzt, aber wir hatten damit eine sehr positive Erfahrung: beim Erstellen der Seiten, Responsive machen und Darkmode erstellen sind immer nur kleinere Probleme entstanden. Das Verbinden mit socket.io war auch nicht das Leichteste, meistens weil es viele Variablen und Spielregeln gibt, die zu beachten sind.
Als letztes musste man dann das Frontend mit der REST-API verbinden, was nicht immer reibungslos funktioniert hat. Dort hatten wir unterschätzt, dass beim Login alles validiert werden muss, was sehr Zeitaufwendig war. Bei dieser Verbindung wurde sonst auch viel die Methode *map* zum rendern von Arrays benutzt, was sehr praktisch war.

Ergebnisse während der Durchführung:

Die Durchführung unseres Projekts ist recht Reibungslos verlaufen. Hier aber einige Punkte, wo nicht immer alles funktioniert hat:

- Bei der Entwicklung vom Backend war eine Bessere Seitenplanung erwünscht, doch das war schwierig, weil wir am Anfang nicht genau wussten, was und wie wir bestimmte Sachen machen
- Bei der Kommunikation zwischen Frontend-Entwickler und API-Entwickler hats einmal kurz gescheitert, aber das war aufgrund unserer limitierten sprachlichen Ressourcen (die Anforderung wurde nicht richtig verstanden)

Es gab sonst keine anderen, größeren Problemen, und wir haben insgesamt unsere Erfahrung als positiv empfunden, und waren auch sehr froh über unsere Entscheidungen, die wir treffen mussten (Vorgehensmodell, Technologien, usw.)

Zeitaufwand (Stand 09.04.2021)

Date	**Description**	**Time (h)**
03/08/2021		**00:48:51**
	UI Design: Logo	00:48:51
03/09/2021		**04:44:45**
	UI Design: Navbar	01:26:30
	Eigene Tasks aufschreiben	00:11:35
	Planen + Logo designen	00:37:35
	Planung	00:33:34
	Backend: socket.io verstehen und testen	00:31:24
	UI Design: Spielseite	01:24:07
03/10/2021		**07:28:37**
	UI Design: Anmelde/Registrierseite	00:50:06
	Backend: make DB calls for testing	00:30:41
	UI Desing: Homepage	00:36:12
	UI Design: Profilseite	01:40:18
	UI Design: Rangliste	00:36:28
	Datenbank Planen	00:43:20
	UI Design: Spielseite	00:40:43
	Backend: überarbeitung des bestehenden Codes	01:06:04
	Raspberry Test-Server aufsetzen	00:44:45
03/11/2021		**05:05:18**
	UI Desing: Roomseite	00:27:11
	UI Design: Darkmode	00:20:46
	Meet	00:44:08
	Backend: rooms	00:23:40
	UI Design: Farben wählen	01:23:16
	UI Desing: Responsive	00:12:14

	UI Design: Farbe wählen	00:35:57
	UI Design: Darmode, Responsive Design	00:58:06
03/12/2021		**04:32:53**
	Frontend: Homepage	01:45:35
	Backend: rooms	00:15:28
	Frontend: Navbar	02:31:50
03/13/2021		**05:39:23**
	Frontend: Spielseite	01:37:23
	Frontend: Roomseite + Footer	00:53:31
	Frontend: Routen + Navbar links	00:20:55
	Frontend: Homepage	01:50:07
	Frontend: Roomseite	00:57:27
03/14/2021		**04:31:10**
	UI Design: Team auswählen	00:14:33
	Frontend: Spielseite	04:16:37
03/15/2021		**04:23:31**
	Frontend: Spiel erstellen Seite	00:32:07
	Frontend: Teams auswählen Seite	00:15:08
	Frontend: Spiel erstellen Seite + Componente ordnen	02:08:43
	Frontend: Registrier + Login Seite	00:32:15
	Frontend: Tabelle fixen	00:19:55
	Frontend: Syling Probleme fixen	00:35:23
03/16/2021		**05:48:25**
	UI Design: Rangliste neu	00:12:18
	Frontend: Rangliste	01:51:18
	Frontend: Search	00:47:57
	Backend: rooms	00:27:16
	Frontend: Profilseite	02:05:13
	Product Backlog erstellen	00:24:23
03/17/2021		**03:53:04**
	Dokumentation zu Datenbankmodelle erstellen und auf Github speichern	00:20:46
	Relationales Modell erstellen	00:17:29
	Backend: rooms	01:40:24

	Frontend: Navabar wenn Anmelden oder Registrieren	00:59:57
	Frontend:Spiel Erstellen	00:34:28
03/18/2021		**06:31:28**
	Frontend: Sidebar for mobile	01:09:51
	Backend: Karte Kartendeck	00:57:39
	Backend: room join leave	00:48:33
	Datenbank erstellen	01:06:21
	Modelle Verbessern aufgrund von ein Zirkelbezug	00:11:49
	Frotend: links fixen, responsive, search added	01:03:37
	Frontend: Responsive Navbar	00:26:53
	Backend: rooms	00:28:39
	Frontend: Project structure	00:18:06
03/19/2021		**05:10:03**
	Frontend: Anmelden + Registrier + Team auswählen + Spiel erstellen Seite responsive	00:18:00
	Frontend: Sidebar responsive	00:12:18
	Frontend: Rangliste responsive	00:34:21
	Frontend: Profilseite responsive	00:16:56
	Backend: Karte Kartendeck	02:16:27
	Frontend: Roomseite responsive	00:30:59
	Frontend: Homepage responsive	01:01:02
03/20/2021		**09:58:38**
	Frontend: Spieleseite	03:09:08
	wattkarten einscannen	01:15:32
	Backend: beste Karte überprüfen	01:57:29
	Wattkarten ausschneiden	01:03:05
	Backend: Karte Kartendeck	00:52:00
	Frontend: Darkmode	01:03:04
	Frontend: Profilseite responsive	00:38:20
03/21/2021		**05:34:40**
	REST-API: Kommunikation mit Datenbank erstellen	01:26:04

	Datenbank: Test-Einträge erstellen, damit die REST-API später getestet werden kann	00:14:18
	Frontend: Custom Scrollbar	00:17:24
	Frontend: Darkmode	03:36:54
03/22/2021		**07:18:05**
	Frontend/Backend: Connect Game mit Server	01:31:03
	Backend: beste Karte überprüfen	00:50:33
	Frontend: Connect Game mit Server	00:12:00
	Resource Gathering: Wattkarten transparent machen	01:01:10
	Frontend: Chat Emojis	01:13:51
	Frontend: Gif	02:29:28
03/23/2021		**02:59:52**
	REST-API: Nuetzliche Methoden entwickeln	00:26:57
	Datenbank: Neuspeicherung einiger Tabelln, um die neue Struktur anwenden zu koennen	00:12:24
	Datenbank: Änderungsvorschlag entwickeln, da man sonst nicht alles gut hineinspeichern kann	00:09:40
	Frontend/Backend: Spiel erstellen	01:18:22
	Backend: Punkte zusammenzählen	00:52:29
03/24/2021		**06:08:41**
	Backend: Karte Kartendeck	00:37:27
	Backend: Punkte zusammenzählen	00:00:02
	REST-API: GET-Methoden entwickeln	00:35:27
	REST-API: Forschung über Levelsysteme/-Funktionen	00:53:14
	Frontend/Backend: Spiel Kommunikation	04:02:31
03/25/2021		**04:17:07**

	Frontend/Backend: Spiel Kommunikation	04:17:07
03/26/2021		**04:51:54**
	Backend: Karten austeilen Function überprüfen	00:46:01
	REST-API: Level aufgrund von Punkte zurückbekommen	01:08:07
	Frontend/Backend: Spiel Kommunikation	02:57:46
03/27/2021		**04:45:15**
	Miscellaneous: GitReports fuer public Issues-Reporting	00:10:53
	Frontend/Backend: Spiel Kommunikation	03:47:00
	REST-API: Letzte GET-Methoden entwickeln	00:47:22
03/28/2021		**06:50:30**
	Frontend/Backen: Rooms erstellen / löschen	01:09:11
	Frontend: Schönerer y Co. fenster besser machen + animationen	01:09:28
	REST-API: POST-Methoden entwickeln	02:41:05
	Frontend/Backend: Bugs fixen	01:15:11
	Frontend/Backend: Spiel Team auswählen	00:35:35
03/29/2021		**07:54:39**
	Frontend/Backend: Nochmal spielen	00:59:28
	Frontend/Backend: refractor + deployment	02:12:30
	REST-API: Dokumentation schreiben	01:24:33
	Miscellaneous: Sicherheitshalber Testserver auf SN-Rechner aufsetzen	00:35:37
	REST-API: Verfeinerung der Methoden	00:39:45
	Frontend: Homepage CSS fix, Game Over windows + event	01:06:24
	Frontend: Deploy	00:23:33
	Frontend: Kleinere Bugs fixen	00:32:49

03/30/2021		**01:36:14**
	Backend: überarbeitung des bestehenden Codes	01:20:59
	REST-API: Rankings upgedated	00:15:15
04/01/2021		**00:32:24**
	Frontend: Bugs fixen	00:32:24
04/03/2021		**05:29:19**
	Frontend: Anmelden, Registrieren	05:00:36
	REST-API: ergaenzung von GET-Methoden	00:28:43
04/04/2021		**02:02:41**
	Spiel Password	02:02:41
04/05/2021		**09:04:01**
	testing	07:11:57
	Apis bei Frontend aufrufen	01:52:04
04/06/2021		**04:06:35**
	REST-API: Ergaenzung von GET-Methoden	00:30:39
	website on server	02:14:31
	(Without description)	01:21:25
04/07/2021		**00:19:01**
	REST-API: Karten auf einmal laden	00:19:01
04/08/2021		**00:38:05**
	Endbericht schreiben	00:38:05
04/09/2021		**00:45:00**
	Endbericht schreiben	00:45:00
Total (01/01/2021 - 12/31/2021)		**143:50:09**

Zukünftiges:

Zur Zeit wo dieses Dokument geschrieben wird, haben wir uns gedacht, dass die Website sobald alle Bugs korrigiert sind, auch veröffentllicht wird. In Zukunft werden wir auch daran weiterarbeiten und sie verbessern, sowie neue Funktionen hinzufügen. Als erstes ist ein Umstieg auf HTTPs geplant.