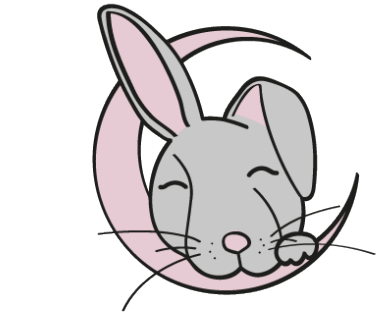
netidee
powerful innovations



gonimo
good night monitor.

Endbericht

Projektnummer: 1200

Author:
Robert Klotzner

19. Mai 2017

Inhaltsverzeichnis

1 Getting ready for netidee

In diesem Arbeitspaket ging es darum alle Vorgaben für die Netidee zu erfüllen die notwendig sind für die Vertragsunterzeichnung. Dazu gehörte unter anderem das Aufsetzen einer Projektwebsite[1], Ausfüllen des Projektcontrollings usw., weiteres wurde einiges in die Wege geleitet um eine gute Grundlage für das Projekt zu schaffen: Projektmanagementtools wurden verglichen, über Projektmanagement

[1] https://project.gonimo.com

an sich recherchiert, hier fiel die Wahl schlussendlich auf Taiga[2], einer Wahl der wir bis heute treu blieben und die uns gute Dienste erwies. Weiteres wurde ein kleines Backupskript entwickelt um sicherzustellen dass Projektergebnisse, Projektplanung und Zeitaufzeichnung jeder Zeit gesichert sind.

2 Programmierung

Sehr bald nach dem Zwischenbericht wurde immer mehr klar, dass die Probleme die bei der Verwendung von purescript-pux als Bibliothek für das Userinterface inherent mit dem Design von purescript-pux verbunden sind und kaum überwunden werden können. Das verwendete Konzept von purescript-pux, welches durch Elm[3] inspiriert ist, funktioniert großartig für kleine Programme, die Skalierung gestaltet sich aber als schwierig. Das Modell war einfach zu unflexibel im Design was ein großes Problem für gonimo wurde.

Immer wieder sah ich mich nach Alternativen um, um dann über Weihnachten reflex[4] auzuprobieren, die Ergebnisse waren so überzeugend, dass der zugegebenermaßen sehr waghalsige Entschluss gefasst wurde, das Frontend auf Basis von reflex neu zu schreiben.

Aus der bisherigen Implementierung wurde sehr viel gelernt, weshalb im Zuge dessen auch das Backend umgekrempelt wurde. Das Ergebnis ist eine viel einfachere Struktur die auch deutlich besser performt und einige Fehler ausbügelt. Besonders deutlich ist dies sichtbar im Aufbau einer Videoverbindung (gonimo neu - instantan, gonimo alt: Mehrere Sekunden, bis fehlgeschlagenem Aufbau bei langsamen Verbindungen). Von Jänner bis April wurde gonimo im Prinzip fast komplett neu geschrieben, mit einem Ergebnis welches sich sehen lassen kann! Die neue Applikation ist deutlich vollständiger, hat eine großartige UX und ein Detailreichtum (Userinteractions) die in der alten Codebasis viel zu mühsam gewesen wären, als dass wir sie je umgesetzt hätten. Alles in Allem steht gonimo jetzt ausgezeichnet da, die verwendeten Technologien machen die Entwicklung zur Freude und sorgen für eine hoch qualitative Codebasis, davon kann man sich durch simple Benutzung überzeugen. Die Applikation ist defakto Bugfree - die Fehler die vorhanden sind und waren sind auf Browserbugs zurückzuführen oder auf Probleme im JavaScript Code.

Das Frontend ist jetzt ebenfalls in Haskell entwickelt und verwendet Bibliotheken zur Kommunikation mit dem Browser, die abstrahiert sind über die tatsächliche Integration, dadurch kann die selbe Codebasis sowohl auf JavaScript kompiliert werden, was momentan für die Webversion passiert, aber auch auf nativen ARM

[2] https://taiga.io
[3] http://elm-lang.org/
[4] https://github.com/reflex-frp/reflex

oder x86 Maschinencode welcher nochmals deutlich performanter läuft. Dies bedeutet, der Weg für eine native Version ist geebnet und wird sich auch mehr wie eine native Applikation anfühlen durch deutlich gesteigerte Performance!

Schlussendlich lernte gonimo auch noch Deutsch, durch großartige Vorarbeit von Martin Heuschober war es uns möglich in sehr kurzer Zeit gonimo bilingual zu machen, die Sprache kann sogar live im Betrieb gewechselt werden.

3 Security Review

Das Security Review gestaltete sich als sehr mühsam in der Organisation, Georg Pichler half hier maßgeblich, er fand und kontaktierte mögliche Reviewer und führte auch eine interne OWASP Analyse durch. Begonnen haben wir denke ich schon im Oktober oder noch früher, aber kontaktierte Reviewer reagierten entweder gar nicht oder waren nicht geeignet oder schlicht weg nicht leistbar. Durch einen Kontakt von Hannes, fanden wir schließlich ein Reviewteam der FH Technikum Wien, welches sich zugegebenermaßen auch nicht als ideal herausstellte. Die Kommunikation verlief auch hier schleppend, e-mails wurden nicht gelesen, ein persönliches Treffen ausgeschlagen. Aus Zeitgründen und eines Mangels an Alternativen führten wir das Review dennoch mit der FH durch, was Anfang April dann endlich geschah. Anfang Mai erhielten wir dann endlich auf mehrfache Anfrage einen Bericht, welcher uns eine ausgezeichnete Security bescheinigt, um den Prüfer zu zitieren:

> Sehr geehrter Herr Klotzner,
>
> anbei der Rohbericht der Sicherheitsüberprüfung.
>
> Wie bereits angekündigt ist der ausgezeichnet ausgefallen, ehrlich gestanden, hatte ich noch nie ein System, dass so abgesichert wurde, wie das Ihre!

Wir freuen uns natürlich über dieses Kompliment, zufrieden sind wir aber noch nicht. Wir werden hier auf jeden Fall noch weiter intern an der Sicherheit arbeiten und Georg habe ich gebeten auch noch einen eigenen Penetrationtest zu machen in dem auch aufwändigere Dinge getestet werden, die bei der Prüfung durch die FH fehlten. Grundsätzlich halte ich die Sicherheit von gonimo persönlich ebenfalls für herausragend, was leider nicht nur auf unser Augenmerk auf Security zurückzuführen ist sondern auch auf die Fahrlässigkeiten der Konkurrenz, wodurch gonimo hier wirklich in einer ganz eigenen Liga spielt.

Einige Details zur Security: Der kritische Server auf dem Userdaten gespeichert werden, hat lediglich 3 offene Ports:

80 http alle Zugriffe werden weitergeleitet auf 443.

443 https alle Zugriffe hier werden über ssl verschlüsselt, es läuft lediglich ein nginx Webserver welcher statische Dateien bereitstellt und dahinter unsere Applikation gonimo-back welche in Haskell geschrieben wurde, was Sicherheitslücken wie Bufferoverflows beinahe anschließbar macht. Um auf Nummer sicher zu gehen, habe ich sogar persönlich ein Review einer verwendeten exponierten Bibliothek durchgeführt welche C Code enthält, bzw. unsafe Code[5][6], weitere Reviews von verwendeten Bibliotheken sind geplant.

22 ssh hier lauscht ein ssh Server der allerdings nur Authentifizierung über private/public Key erlaubt.

Sie sehen, die Angriffsfläche ist wirklich minimal. Wir verzichten bewusst auf weitere Services auf diesem Server, insbesondere vermeiden wir auch php komplett. Dieses Augenmerk auf Security zieht sich auch durch die gesamte Entwicklung, Daten werden statisch typisiert, Escaping von Usereingaben werden durch das Typsystem garantiert, Fehler in diesem Bereich können nicht zuletzt durch den Einsatz von Haskell beinahe ausgeschlossen werden.

4 Design, Webdesign, Artwork

Für gonimo wurde eine komplett neue UX entworfen und implementiert, welche auf https://app.gonimo.com bewundert werden kann. Entworfen wurde sie von Romana Roithner gemeinsam mit Christian Simulak, grundsätzliche Umsetzung in CSS und HTML führte Aristoteles Riedmann um, die tatsächliche Implementierung führte ich durch. Alle Illustrationen sind auf Romana Roithner, Christian Simulak und Chiara Cavalli zurückzuführen. Design der Website wurde ebenfalls von Romana Roithner und Christian Simulak durchgeführt, implementiert wurde sie wieder von Aristoteles Riedmann. Visitenkarten, Rollup, usw. alles von Romana Roithner und Christian Simulak.

Für eine englische Basisübersetzung der Landingpage, müssen wir uns bei Damita Pressl bedanken. FAQ ausgebaut und überarbeitet wurde von Christian Klotzner (ehrenamtlich), Überarbeitung und Übersetzung des neuen FAQs wurde von mir und Martin Heuschober durchgeführt.

[5] https://github.com/bos/aeson/issues/538
[6] https://github.com/bos/aeson/pull/537

5 Promotion Video & Tutorialvideos

Das Promotionvideo wurde durch Chiara Cavalli gedreht, geschnitten und animiert, Zeichnungen stammen wieder von Romana und Christian. Hannes Kohlmann kümmerte sich komplett ehrenamtlich um die großartige Tonspur des Videos. Der gonimo Jingle (im Hintergrund) ist von ihm und die professionellen Sprecher (Englisch & Deutsch!) wurden von ihm aufgetrieben und er führte auch die Aufnahmen durch. Für die deutsche Version fuhr er sogar extra nach Berlin (Reisekosten privat übernommen). Die zu sehenden Schauspieler, einschließlich Alissa setzten sich ebenso alle ehrenamtlich für gonimo ein.

Die Tutorialvideos wurden von Hannes Kohlmann getextet, mit mir gemeinsam aufgezeichnet und durch Chiara geschnitten. Besonderer Dank hier auch an Martin Heuschober der sich um die Übersetzung auf Englisch kümmerte.

6 Marketing

Durch den notwendigen Rewrite von gonimo, setzte das Marketing erst gegen Ende des Projektes ein. Wir haben auf Facebook in Gruppen gepostet, Leute angeschrieben, Freunde und Bekannte haben geliked und geshared. Wir haben auf reddit gepostet, Blogger wurden kontaktiert. Besonders hervorheben möchte ich hier die Leistungen von Christian Klotzner, der sich wirklich heroisch für eine Bekanntmachung von gonimo einsetzt.

Zur Marketingperformance, Facebook posten in Gruppen ist relative ineffektiv, am ehesten werden damit zusätzliche Likes auf unserer Facebook Seite erreicht, aber kaum neue User lukriert. Als sehr effektiv stellte sich Posts auf reddit.com/r/haskell heraus, welche jedes-mal in etwa 200 neu registrierte Geräte auf app.gonimo.com brachte. Als sehr effektiv, stellte sich auch die Initiative meines Bruders heraus einen deutschen Blogger zu kontaktieren, der Beitrag[7] brachte uns rund 300 neu registrierte Geräte. Aktueller Stand: Mehr als 800 registrierte Geräte.

Christian Klotzner hat bereits fünf weitere Blogger kontaktiert und ich bekam auch schon das OK von opensource.com einen Artikel für sie über gonimo verfassen zu dürfen! opensource.com hat auf Facebook 230000 Followers! Ein Artikel sollte unserer bisherigen Erfahrung nach, in 2000 - 3000 neue registrierten Geräten resultieren. (Bisherige Quote Followers - Impact war in etwa 1%)

Weiter Maßnahmen werden sein, Zeitungen zu kontaktieren (wir kennen da Leute) und in weiterer Folge (etwas später) auch ein Fernsehauftritt, auch hier existieren Kontakte die das möglich machen sollten. Weiteres hat Hannes einen Kontakt nach Brasilien, welcher uns auch schon eine portugiesische Übersetzung von

[7] http://stadt-bremerhaven.de/gonimo-babyphone-via-browser-realisieren/

Landingpage und App erstellte und vernetzt ist mit brasilianischen Bloggern und News. To quote her:

> - I already found about five vehicles (blogs and columns) about motherhood whose authors asked for news suggestions (I don't know a better way to translate it, I don't even know whether people do this around the world, but it's a thing in Brazilian content production routine, pretty much asking for press releases and news ideas) in a Facebook group I am.
> - I also thought Gonimo might interest tech columns and vehicles and I also intend to contact them when Gonimo is lauched. What do you think?
> - Only now I remembered I have some acquaintances from college who work for our biggest news vehicles. I can't guarantee I can convince them to publish anything, but I wanted to let you know it might be possible.
>
> Translating the whole content gave me better insight on Gonimo and I think it might actually go BIG!

Wir haben jetzt auch einen erfahrenen Wirtschaftler an Board der auch an einer FH ab 2018 Digital Business Models unterrichten wird. Er ist sehr überzeugt von unserem Projekt und steht uns auch beratend und unterstützend zur Seite.

Auf der Babyexpo im November werden wir vertreten sein (neues Rollup haben wir schon), mein Stiefvater hat nächste Woche ein Treffen mit dem Verantwortlichen für die Willkommensbox die Mütter bei der Geburt bekommen (ist jetzt ein Rucksack), möglicherweise kann hier ein gonimo Flyer beigefügt werden. Hannes und ich werden auch noch Familienhotels und Babyhotels kontaktieren, auf dass Gonimo Flyer in den Zimmern aufliegen. Kontakte zu Kindergärten und einem Spital existieren auch.

Die Reise hat gerade erst begonnen ...

7 Nicht erreichte Ziele

7.1 Gute Cross-Browser Unterstützung

Durch die Nutzung von sehr fortschrittlicher Technologie in gonimo gestaltet sich eine flächendeckende Unterstützung von Browsern als schwierig. Safari unterstützt die benötigte Kerntechnologie nachwievor überhaupt nicht, Internet Explorer sowieso nicht, Edge noch sehr schlecht. Aus diesem Grund funktioniert gonimo derzeit nur auf Chrome und Firefox (sowohl Mobile als auch Desktop) und die beste Unterstützung erfährt nachwievor Chrome, was auch der populärste Browser ist

der momentan unterstützt werden kann. Wir werden jetzt auch nicht mehr länger auf Apple warten, eine der wichtigsten nächsten Schritte wird eine iOS native Version von gonimo sein. D.h. eine installierbare App die dann auch schon ein erstes Einkommensmodell von gonimo darstellt. Ryan Trinkle der Autor der so überzeugenden FRP Bibliothek (reflex und reflex-dom) welche wir jetzt wie gesagt für unser Frontend verwenden und die auch nativ kompilierbar ist, hat bereits seine Unterstützung für diese Projekt angekündigt!

7.2 1000 aktive Nutzer

Wie bereits erwähnt setzte das Marketing leider deutlich später ein als geplant, unser offizieller Launch ist nun eine Woche her, in dieser Woche war es uns möglich 800 registrierte Geräte zu erzielen! Dies stimmt uns recht optimistisch, da wir eigentlich gerade erst los legen - siehe Kapitel 6 Marketing.

8 Projektziel und erreichte Projektendergebnisse

Das Projektziel ein vollwertiges web-basiertes Babyphone zu entwickeln, wurde erreicht. Ein Promotionvideo wurde erstellt, Tutorialvideos wurden erstellt, Securityreview wurde durchgeführt.

Eine komplette Liste der Projektergebnisse hier:

8.1 Gonimo - die App

Typ:	Web Applikation
Code:	https://github.com/gonimo/gonimo
Sprache:	Haskell
Lizenz:	AGPLv3
Web:	https://app.gonimo.com
Funktion:	Das web-basierte Babyphone selbst, das Kernprojekt.
Beschreibung:	Die eigentliche Applikation die entwickelt wurde, einfach ausprobieren direkt im Browser, durch Besuchen von https://app.gonimo.com
Impact:	Applikation wird sehr gut angenommen, das Feedback ist im Allgemeinen sehr positiv. Wir sind guter Dinge, dass gonimo ein ernstzunehmender Player im Markt sein wird.

8.2 Landing page

Typ:	Webseite
Code:	https://github.com/gonimo/gonimo-website
Sprache:	PHP
Lizenz:	AGPLv3
Web:	https://gonimo.com
Funktion:	Die Landingpage für das Projekt.
Beschreibung:	Neue Nutzer werden auf dieser Seite empfangen und erhalten einen ersten Eindruck vom Produkt, wie es funktioniert, was es ist und ob sie hier richtig sind. Weiteres können sie hier die eigentliche Webapplikation starten.
Impact:	Das Gesicht von gonimo nach außen.

8.3 Security Review

Typ:	Report
Code:	N/A
Sprache:	Deutsch
Lizenz:	N/A
Web:	Nicht veröffentlicht, aber beigefügt.
Funktion:	Bescheinigung über die Sicherheit von Gonimo.
Beschreibung:	Der Bericht den die FH über gonimo verfasst hat. Gefundene Sicherheitslücken werden hier gelistet und Verbesserungsvorschläge erbracht.
Impact:	Vertrauen in unsere Dienste wurde geschaffen.

8.4 purescript-argonaut-generic-codecs

Typ:	Programmierbibliothek
Code:	https://github.com/eskimor/purescript-argonaut-generic-codecs
Sprache:	PureScript
Lizenz:	MIT
Web:	N/A
Funktion:	Haskell kompatible JSON Kodierung zur Serverkommunikation
Beschreibung:	Für die PureScript Version von gonimo-front war eine Server/Client kompatible Kommunikation zu gewährleisten. Diese Bibliothek ermöglicht eine Kodierung von PureScript Datentypen nach JSON die kompatibel mit der Kodierung auf der Haskellseite ist.
Impact:	Gonimo selbst benötigt diese Bibliothek auf Grund des Rewrites nicht mehr, die Bibliothek wird allerdings in der Community sehr gut angenommen und wird dort weiter gepflegt, wie in gemergten pull requests nachvollzogen werden kann.

8.5 purescript-subscriber

Typ:	Programmierbibliothek
Code:	https://github.com/eskimor/purescript-subscriber
Sprache:	PureScript
Lizenz:	MIT
Web:	N/A
Funktion:	Clientcode für servant-subscriber
Beschreibung:	Stellt den Clientcode für PureScript für servant-subscriber zur Verfügung. Ermöglicht die Nutzung von servant-subscriber von einem in PureScript geschriebenen Frontend heraus. Für weitere Details zu subscriber, verweise ich hier auf servant-subscriber.
Impact:	Gonimo selbst benötigt diese Bibliothek auf Grund des Rewrites nicht mehr. Die Bibliothek wird scheinbar verwendet[8][9], im Vergleich zu den anderen Bibliotheken denke ich ist die Verbreitung aber gering.

[8] https://github.com/eskimor/purescript-subscriber/pull/7
[9] https://github.com/eskimor/purescript-subscriber/issues/6

8.6 servant-subscriber

Typ:	Programmierbibliothek
Code:	https://github.com/eskimor/servant-subscriber
Sprache:	Haskell
Lizenz:	BSD3
Web:	https://www.stackage.org/lts-8.13/package/servant-subscriber-0.5.0.3
Funktion:	Erweiterung einer REST API mit Notification Features.
Beschreibung:	Für gonimo war eine REST API leider nicht ausreichend, weshalb servant-subscriber entwickelt wurde. Diese Bibliothek ermöglicht es einem Clients Ressourcen zu abonnieren, um dann über Änderungen in dieser Ressource automatisch vom Server verständigt zu werden.
Impact:	Gonimo selbst benötigt diese Bibliothek auf Grund des Rewrites nicht mehr. Allerdings wurde der Code in gonimo direkt übernommen, und für die reine WebSocket Verbindung adaptiert. In der Community scheint die Bibliothek Verwendung zu finden, aber ich denke die Verbreitung ist recht gering.

8.7 servant-purescript

Typ:	Programmierbibliothek
Code:	https://github.com/eskimor/servant-purescript
Sprache:	Haskell
Lizenz:	BSD3
Web:	https://www.stackage.org/lts-8.13/package/servant-purescript-0.6.0.0
Funktion:	Automatische Generierung von typsicheren Zugriffsfunktionen auf eine mit servant spezifizierte REST API.
Beschreibung:	Um eine einfache, typsichere und fehlerfreie Kommunikation zwischen Client und Server zu gewährleisten, werden mit Hilfe dieser Bibliothek automatisch Code generiert der die nötige Kodierung und Dekodierung automatisch bewerkstelligt.
Impact:	Gonimo selbst benötigt diese Bibliothek auf Grund des Rewrites nicht mehr, die Bibliothek wird allerdings in der Community sehr gut angenommen und wird dort weiter gepflegt, wie in gemergten pull requests nachvollzogen werden kann.

8.8 purescript-bridge

Typ:	Programmierbibliothek
Code:	https://github.com/eskimor/purescript-bridge
Sprache:	Haskell
Lizenz:	BSD3
Web:	https://www.stackage.org/lts-8.13/package/purescript-bridge-0.10.1.0
Funktion:	Typsichere Server/Client Kommunikation
Beschreibung:	Diese Bibliothek übersetzt Haskell Typen auf kompatible PureScript Typen. Sie schafft damit die Grundlage für eine typsichere und fehlerfreie Server/Client Kommunikation.
Impact:	Gonimo selbst benötigt diese Bibliothek auf Grund des Rewrites nicht mehr, die Bibliothek wird allerdings in der Community sehr gut angenommen und wird dort weiter gepflegt, wie in gemergten pull requests nachvollzogen werden kann.

8.9 purescript-servant-support

Typ:	Programmierbibliothek
Code:	https://github.com/eskimor/purescript-servant-support
Sprache:	PureScript
Lizenz:	MIT
Web:	N/A
Funktion:	Hilfscode benötigt von durch servant-purescript erzeugten Code
Beschreibung:	Der von servant-purescript benötigt einiges an Funktionalität, diese wird in dieser Bibliothek zur Verfügung gestellt.
Impact:	Gonimo selbst benötigt diese Bibliothek auf Grund des Rewrites nicht mehr, die Bibliothek wird allerdings in der Community sehr gut angenommen und wird dort weiter gepflegt, wie in gemergten pull requests nachvollzogen werden kann.

8.10 purescript-localstorage

Typ:	Programmierbibliothek
Code:	https://github.com/eskimor/purescript-localstorage
Sprache:	PureScript
Lizenz:	MIT
Web:	N/A
Funktion:	Zugriff auf Browser local storage und session storage.
Beschreibung:	Typsicherer Zugriff auf Localstorage, ermöglicht dem Frontend das speichern von Daten, lokal beim Client.
Impact:	Gonimo selbst benötigt diese Bibliothek auf Grund des Rewrites nicht mehr, die Bibliothek wird allerdings in der Community sehr gut angenommen und wird dort weiter gepflegt, wie in gemergten pull requests nachvollzogen werden kann.

8.11 gonimo-front - alte Version

Typ:	Web Applikation
Code:	https://github.com/gonimo/gonimo-front
Sprache:	PureScript
Lizenz:	AGPLv3
Web:	N/A
Funktion:	Altes Frontend von gonimo.
Beschreibung:	PureScript Version des Frontends von gonimo auf Basis von purescript-pux.
Impact:	Es haben sich schon einige Entwickler bei mir bedankt, dass der Code zugänglich ist und sie davon lernen konnten.

8.12 gonimo-back - alte Version

Typ:	Web Applikation
Code:	https://github.com/gonimo/gonimo-back
Sprache:	Haskell
Lizenz:	AGPLv3
Web:	N/A
Funktion:	Altes Backend mit servant REST API.
Beschreibung:	Altes Backend auf REST API Basis.
Impact:	Basis des aktuellen Backends.

8.13 Promotion und Tutorial Videos

Typ:	Videos
Code:	N/A
Sprache:	Haskell
Lizenz:	CC BY-NC-ND 2.0
Web:	https://www.youtube.com/channel/UCPibVPU9FtQ3dfIpSUJ3_fQ
Funktion:	Produktvorstellung und Anleitung.
Beschreibung:	Videos um einen ersten Eindruck von gonimo zu vermitteln und den Einstieg zu erleichtern.
Impact:	Verwendung im Facebookmarketing, bisher knapp 600 Views.