EIS

Entwicklung Interaktiver Systeme

Sommersemester 2018

Konzept - MS1

Team:

Jan Omar Mehr
Armin Weinrebe

Dozenten:

Prof. Dr. Gerhard Hartmann
Prof. Dr. Kristian Fischer

Mentor:

B. Sc. Robert Gabriel

Inhaltsverzeichnis

(einfügen)

Zielhierarchie

Die Zielhierarchie wird zur Definition von lang- bis kurzfristigen Zielen des Projektes verwendet. Daraus soll ersichtlich werden, welche Vorgänge Priorität haben, wo eventuelle Probleme auftauchen könnten und es soll die Grundlage für die spätere Evaluation des Projektes darstellen.

langfristige / strategische Ziele:

- Nutzer müssen die Anwendung überall und permanent erreichen und benutzen können
- Nutzer müssen auf einer Reise mit anderen mitreisenden Nutzern automatisch Reisekosten abrechnen können
- Nutzer sollen während und nach der Reise eine Statistik über das Kaufverhalten einsehen können
- Das System soll auf langfristige Sicht das Kaufverhalten aller Nutzer analysieren
- das System soll Kommunikation zwischen Nutzern mit Sprachbarrieren erleichtern
- das System soll global anwendbar sein und mehrere Währungen und Sprachen unterstützen
- das System kann als Plattform für Reisegruppenfindung/-erstellung dienen
- das System soll dem Nutzer helfen Konflikte mit anderen Nutzern präventiv zu vermeiden

mittelfristige / taktische Ziele:

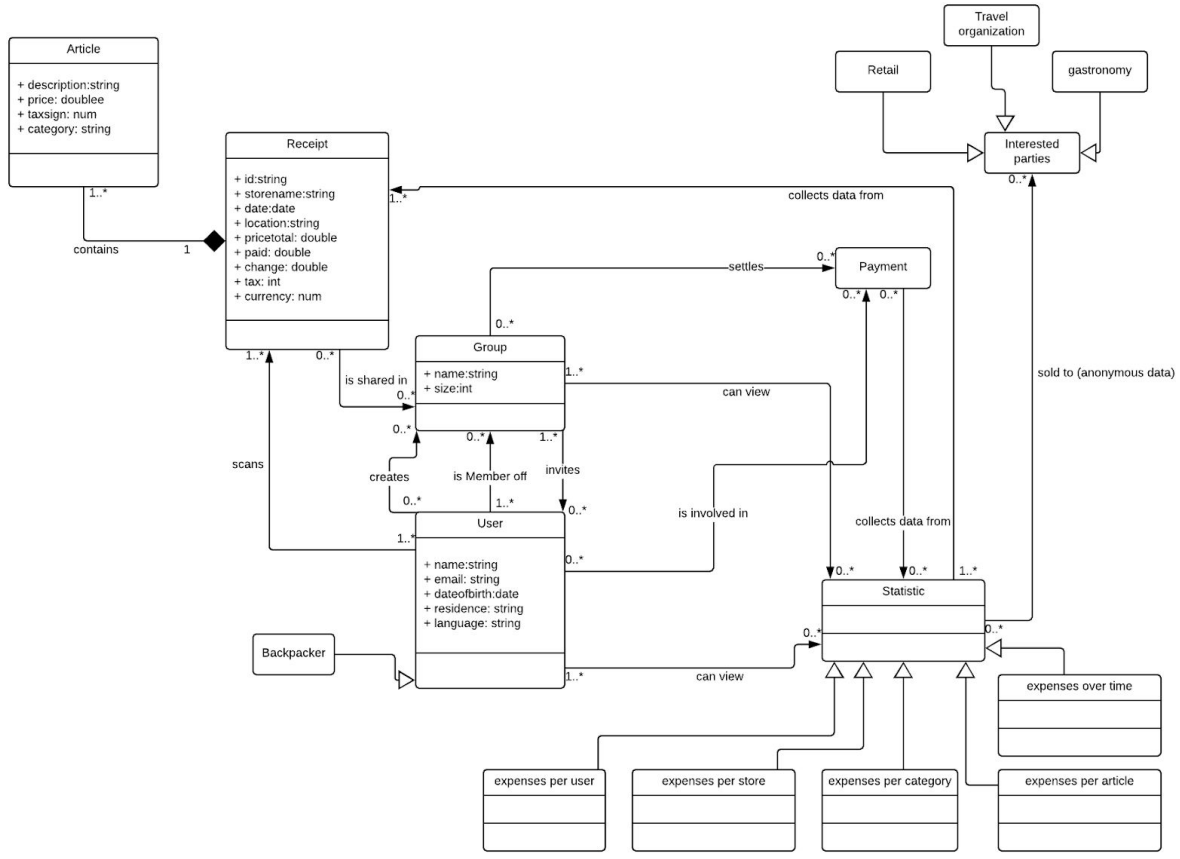
- Das System muss in möglichst einfacher Sprache gehalten werden und wenn möglich selbsterklärende Icons und Bilder verwenden
- Das System soll vorgefertigte, positive Dialoge anbieten
- Das System behandelt Konflikte per Dialog, Warnung vor Grenzen und Konsequenzen
- Das System muss die Kosten einer Reise (unter anderem anhand der Daten aus Kassenzetteln) für alle Mitglieder einer Reisegruppe kalkulieren und anzeigen
- Das System soll anhand der Daten aus den Kassenzetteln eine genaue Abrechnung für einzelne Mitglieder einer Reisegruppe erstellen
- das System soll dem Nutzer die Möglichkeit bieten Währungen in Echtzeit umzurechnen
- sas System soll dem Nutzer eine Übersicht über alle vergangenen Reisen mit deren Kosten anzeigen
- Das System soll den Nutzern die Möglichkeit geben Abrechnungen direkt Begleichen zu können (online oder bar)

kurzfristige / operative Ziele:

- der Nutzer muss Reisegruppen erstellen/auflösen und andere Nutzer einladen/entfernen können
- das System soll die Informationen aus fotografierten Kassenzetteln in digitale Daten umwandeln
- das System muss vorerst auf englisch verfügbar sein, um möglichst viele Nutzer zu erreichen
- das System soll die Abrechnung so transparent und klar verständlich wie möglich darstellen
- Das System soll bei der Abrechnung dem Nutzer Möglichkeiten geben andere Nutzer von der Rechnung aus- oder einzuschließen
- Nutzer sollen Abrechnungen markieren können um Kommunikation zu erleichtern
- Kosten sollen abschnittsweise abgerechnet werden können

Domänenrecherche:

Ishikawa Diagramme
Domänenmodell
???



Marktrecherche:

Mobile Anwendungen

Friendcash, Tricount, Trip Splitter:
Friendcash ist ein Lösungsansatz dem Problemraum der Zettelwirtschaft auf Reisen entgegen zu treten. Die App ist eventbasiert ausgelegt, so dass einer Reise Mitglieder zugeordnet werden, die mit Hilfe eines Finanz-Rechners Abrechnungen in verschiedenen Währungen vornehmen können. Dabei können Mitglieder in die Abrechnung mit einbezogen oder ausgeschlossen werden, in Bezug auf eine Kaufaktion. Die Daten werden jeweils manuell eingetragen, was je nach größe der Reise und den damit verbundenen Kosten sehr aufwendig sein kann. Die Funktionen sind größtenteils unabhängig vom Internet. Allerdings bedeutet das auch, dass mehrere Clients der einzelnen Mitglieder die einem Event zugeordnet werden, sich nicht automatisch synchronisieren. Die Anwendung bietet hierfür die als Lösung manuell Daten zu integrieren oder das Teilen der Abrechnung als E-Mail-Verteilung zu realisieren.
Tricount und Trip Splitter überschneiden sich mit Friendcash im Aufbau und in Ihren Funktionen. Sie bieten diese nahezu identisch an, haben jedoch in Präsentationslogik und dem Design leichte Unterschiede. So bietet Tricount schlichtere Darstellungen die für eine bessere Übersicht sorgen. Trip Splitter hat fertige Icons integriert, die bildhaft darstellen für was die Ausgaben getätigt wurden. Dies beschleunigt den Abrechnungsvorgang, da weniger Text manuell eingetippt werden muss.
(Quellen: https://www.iphone-ticker.de/iphone-app-geld-ausgaben-verteilen-37159/ , https://play.google.com/store/apps/details?id=com.tribab.tricount.android&utm_source=website&utm_medium=home&utm_content=lang%3Ade%2Cbtn%3Abottom-btn&%24fallback_url=https%3A%2F%2Fplay.google.com%2Fstore%2Fapps%2Fdetails%3Fid%3Dcom.tribab.tricount.android&_branch_match_id=517666614845263475, http://appcrawlr.com/ios/trip-splitter#authors-description 27.04.2018)

Vorteile:

- In der Komplexität gering gehalten und dadurch sofort benutzbar.
- In den zentralen Funktionen unabhängig vom Internet.
- Benötigt wenig Ressourcen.
- Der eventbasierte Aufbau ist zielführend und ermöglicht eine Problemlösung in wenigen Interaktionsschritten zu realisieren.
- Die Anwendungen sind kostenlos.

Nachteile:

- Wenig Automatisierung und dadurch Aufwand für die Erstellung eines Events.
- Kein verteiltes System, daraus folgt:
    - Keine Verknüpfung bzw. automatische Synchronisation von Clients.
    - Keine Unterstützung des Systems durch einen Server.
- Kein Bezug zu realen Dokumenten.

Klippa:
Diese App ist unser größter Konkurrent. Sie umfasst kostenlose Features wie das fotografieren von Kassenzetteln, dem Speichern von Daten wie Einkaufsladen, Gesamtpreis, etc. und Kategorien zu dem jeweiligen Zettel. Eine Suche durch alle gespeicherten Zettel anhand von Datum, Laden oder Kategorie ist ebenfalls implementiert. Außerdem können einfache Statistiken anhand der Daten erstellt werden. Die Kassenzettel mit ihren Daten können in verschiedenen Formaten (pdf, jpeg) exportiert und geteilt werden. Für einen In-App Kauf kann man das OCR (optical character recognition) Feature freischalten, mit dem man angeblich einzelne Daten auf dem Kassenzettel automatisch erkennen kann. Leider konnten wir dies aufgrund des Preises nicht testen. Man findet auch aufgrund der geringen Bekanntheit der App keine Testberichte. Im Hinblick auf das Alleinstellungsmerkmal konnten wir feststellen, dass die Statistiken nur sehr rudimentär sind und keinen detaillierten Einblick in das Kaufverhalten der Nutzer bietet. Außerdem ist es nicht möglich sich innerhalb der App in Gruppen zusammenzuschließen, sondern nur mit einzelnen Nutzern.
(Quelle: https://www.klippa.com/en/homepage/ 18.04.2018)

Vorteile:

- Die Anwendung ermöglicht die Integrierung von Dokumenten in das System.
- Die Anwendung enthält praktische OCR-Funktionalität und Suchfunktion.
- Die Anwendung ist plattformübergreifend (Android, iOS und Browser)
- Ermöglicht das Exportieren von Daten in verschiedene Formate.
- In der Komplexität gering gehalten und dadurch sofort benutzbar.
- Übersicht durch Kategorisierung.
- Gesamtübersicht auch durch Graphische Darstellungen.
- Die Anwendung präsentiert sich schlicht und übersichtlich.

Nachteile:

- Die praktische OCR-Funktionalität verbirgt sich hinter einer Paywall.
- Es wird keine kollaborative Zusammenarbeit berücksicht.

Desktop-Anwendungen

Shoeboxed:
Diese Buchhaltungssoftware wurde für kleinere Unternehmen designed. Sie beinhaltet ebenfalls Features wie OCR, automatische Suchfunktionen, Kategorisierung und Export. Zudem auch noch Konten verknüpft werden und es ist ein direkter Export zu Finanzberatern/-organisationen möglich. Allerdings ist die Software ziemlich teuer (monatlicher skalierbarer Beitrag) und es ist nicht möglich einzelne Artikel auf dem Beleg automatisch zu erkennen. Der mobile Client erhielt sehr schlechte Bewertungen aufgrund von Bugs, mangelnder Gebrauchstauglichkeit, etc. (Quelle: https://www.shoeboxed.com/features/ 18.04.2018)

Vorteile:

- Die Anwendung enthält praktische OCR-Funktionalität und Suchfunktion.
- Übersicht durch Kategorisierung.
- Ermöglicht das Exportieren von Daten in verschiedene Formate.
- Verknüpfung mit betroffenen Institutionen.
- Die Anwendung ist plattformübergreifend.

Nachteile:

- Die Software ist kostenintensiv.
- Der Client erfüllt nicht die Anforderungen, die aus Sicht der Benutzer erforderlich sind.

Growth from Knowledge (GfK):

Bei Gfk handelt es sich um ein Unternehmen, welches mit technologischen und wissenschaftlichen Verfahren unternehmensspezifische Fragen zu Verbrauchern, Märkten, Marken und Medien behandelt. Den Kunden verspricht es mit Hilfe von Forschung und Analytik Wirtschaftswachstum zu erlangen. Die unternehmenseigene Software wird nicht gehandelt, jedoch bietet das Unternehmen einen Client für Kunden an, zur besseren Zusammenarbeit. In Bezug auf das Projekt ist die Messung zu Verbraucherverhalten besonders interessant. Nach Angaben der GfK wurde beispielsweise eine umfassende Untersuchung zum Kaufverhalten von Urlaubern bei der Planung der nächsten Reise durchgeführt. Dabei wurden in 15.000 Haushalten über ein Browser-Plug-In Verhaltensweisen mitgeschnitten während ein Media Efficiency Panel Demografiedaten, Absichten und Kaufaktionen zusammenträgt.
(Quelle:http://www.gfk.com/de/success-stories/success-story/kaufverhalten-von-urlaubern-messen-bei-der-planung-ihrer-naechsten-reise/ 19.04.2018)

Vorteile:

- Fundierte Verfahren zur Ermittlung von Kaufverhalten.
- Repräsentative Ergebnisse durch enorme Quantität der Daten.

Nachteile:

- Sehr kostenintensiv.
- Sehr undurchsichtig.

Hardware

The Neat Company

Diese Firma verkauft Scanner mit zugehöriger Software. Der Nutzer kann so Dokumente mit OCR scannen, Kategorisieren, Durchsuchen, Importieren und Exportieren. Zudem ist es möglich mit anderen Nutzern zu kollaborieren. Allerdings anscheinend in Form von Kommentaren und Chats. Es werden auch viele Schnittstellen, wie z.B LinkdIn,

OnlineBanking, etc.) geboten. Dieses Angebot ist jedoch auch sehr teuer (Scanner + monatlicher Beitrag für Software). Zudem ist die Anwendung langsam, erkennt keine einzelnen Artikel und hat sehr schlechte Bewertungen.
(Quelle: https://www.neat.com/how-neat-works/ 18.04.2018)

Vorteile:

- Die Anwendung ermöglicht die Integrierung von Dokumenten in das System.
- Die Anwendung enthält praktische OCR-Funktionalität und Suchfunktion.
- Ermöglicht das Exportieren von Daten in verschiedene Formate.

Nachteile:

- Sehr kostenintensiv.
- Die Software ist sehr eingeschränkte in ihrer Funktionalität

Alleinstellungsmerkmal:

Anhand der Marktanalyse wurde festgestellt, dass es für einige Teilaspekte der Nutzungsprobleme bereits Lösungsansätze in verschiedenen Formen gibt. So können Kassenzettel bereits in Ihrer Gesamtheit (also dem Gesamtpreis) gescannt und ausgewertet werden (Klippa, Shoeboxed), aber eine automatische Auswertung anhand einzelner Artikel ist noch nicht möglich. Einige bieten daraufhin auch eine firmeninterne Abrechnung an. Andere Anbieter ermöglichen eine Abrechnung für Freunde oder Reisegruppen anhand von manuell eingetragenen Daten (Friendcash, Tricount, Trip Splitter).
Allerdings ist uns dabei aufgefallen, dass es noch keine Kombination aus der automatischen Erfassung der Daten zu einer weitestgehend automatischen Abrechnung innerhalb einer Gruppe gibt. Außerdem ist es in keiner Anwendung möglich die Abrechnung mit allen Teilnehmern in Echtzeit zu synchronisieren. Nur der Export oder das Teilen mit einzelnen Nutzern wird unterstützt.
Zudem stellten wir fest, dass einige Nutzergruppen eventuell gar kein Interesse an unserer Lösung haben könnten. Reisegruppen aus Familien oder engen Freunden nehmen die Abrechnung möglicherweise nicht besonders ernst oder können sich zumindest ohne die Anwendung so gut verständigen und vertrauen, dass viele unserer Lösungsansätze für diese Nutzer obsolet wären. Daraufhin untersuchten wir eine Nutzergruppe, bei der Geld bzw. eine genaue Abrechnung eine größere Rolle spielt, Kommunikationsschwierigkeiten herrschen oder noch kein Vertrauen aufgebaut wurde. In Folge stießen wir auf die Nutzergruppe “Backpacker” oder “junge Erwachsene auf Reisen”, welche in fremden Ländern auf fremde Menschen treffen und sich dort zu neuen Reisegruppen zusammenfinden. Die besten Beispiele hierfür wären Australien, Amerika, Asien oder Kanada, welche als die beliebtesten Ziele für Backpacker gelten.
Im Bezug auf die Recherche und die herausgestellte Nutzergruppe können folgende Alleinstellungsmerkmale konkretisiert werden.

- Automatische Abrechnung anhand von Scan einzelner Artikel eines Kassenzettels
- Echtzeit-Synchronisation und Einbindung aller Teilnehmer in die Abrechnung
- Spezielles Design hinsichtlich der Nutzergruppe “Backpacker”

Stakeholderanalyse:

Priorität 1 = direkte Nutzer, 2 = Investoren, Geschäftspartner, 3 = indirekt/extern

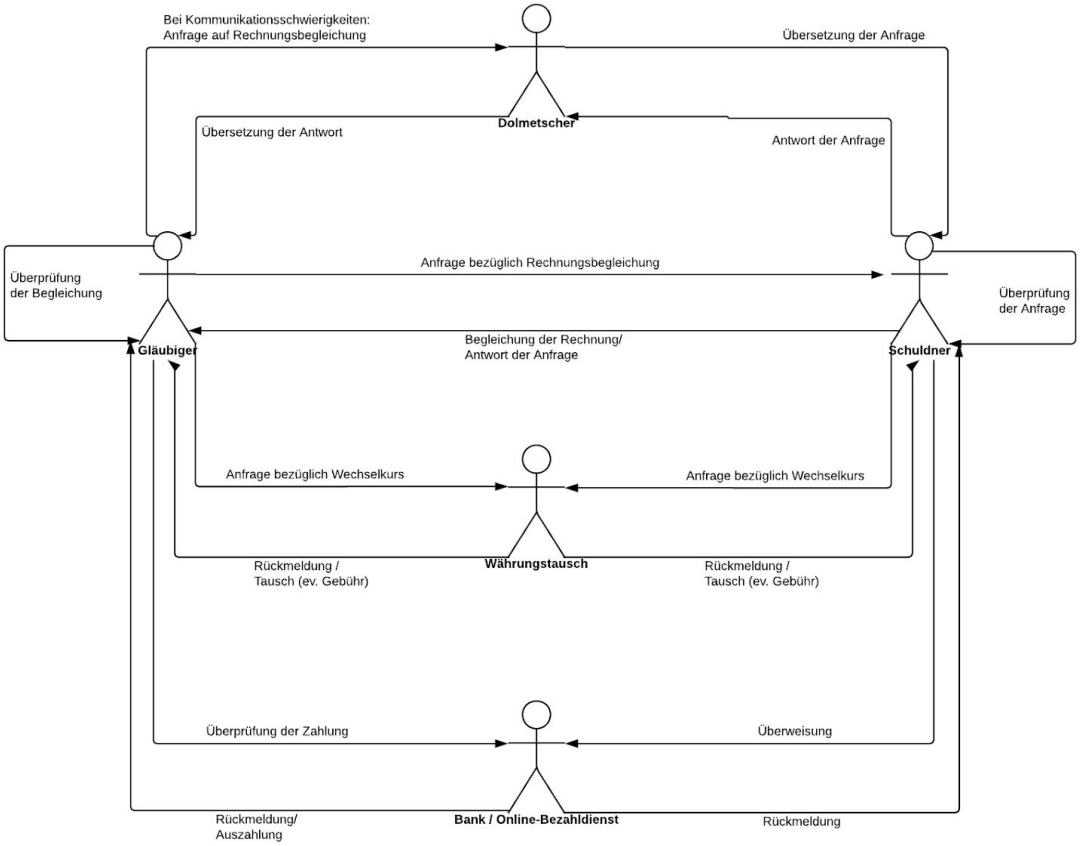
Bezeichnung	Beziehung	Objektbereich	Erfordernisse/Erwartungen	Priorität
Alle Nutzer	Anrecht	persönliche Informationen	vertraulicher Umgang mit Informationen	1
	Anspruch	Abrechnung	korrekte automatische Berechnung der Abrechnung	1
		Kollaboration	alle involvierten Nutzer einladen und Daten teilen	1
		Kommunikation	Eindeutige Kommunikation mit Nutzern anderer Sprachen	1
		Währungstausch	Reibungslose Vermittlung	1
		Bezahldienst	Reibungslose Vermittlung	1
		Datenspeicherung	Persistenz und Konsistenz	1
	Interesse	Kassenzettel-Scan	automatische Datenerhebung	1
		Kommunikation	Behandlung von Konflikten	1
		Währungstausch	Anpassungsmöglichkeit an eine Währung	1
		Bezahldienst	Überweisungsmöglichkeit	1
Nutzer: Gläubiger		Kollaboration	Die genaue Forderung des finanziellen Ausgleichs mitteilen und dokumentieren	1
		Kommunikation	Skepsis überwinden und Vertrauen stärken	1
Nutzer: Schuldner		Kollaboration	Die Forderung nachvollziehen und überprüfen	1
		Kommunikation	Orientierung für angemessenes Verhalten bezüglich des Abrechnens	1
Währungstausch		Währungstausch	Kundengewinnung	2
Bezahldienst		Bezahldienst	Kundengewinnung	2
Einzelhandel	Anrecht	persönliche Informationen	vertraulicher Umgang mit Informationen	3

		Vertrag	Einhaltung der Vertragsbedingungen	3
	Anspruch	Kaufverhalten	anonymisierte Daten des Kaufverhaltens der Touristen	3
		Werbung	Schalten von Werbung anhand von Kaufdaten der Nutzer	3
	Interesse	Werbung	Anpassung der Werbung an Nutzer	3

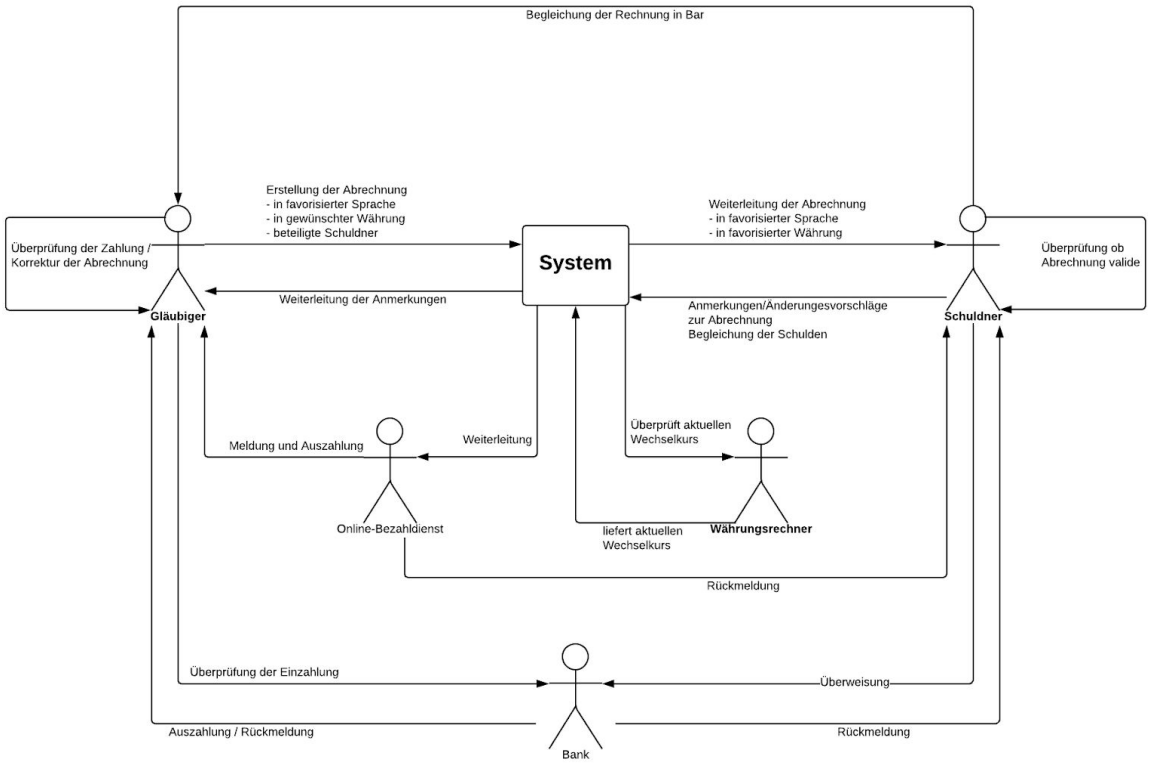
Kommunikationsmodelle:

Anhand der Kommunikationsmodelle soll der Nachrichtenaustausch zwischen Nutzern und Instanzen des Systems verdeutlicht werden. Das deskriptive Modell zeigt den IST-Zustand, also wie die Kommunikation ohne unseren Lösungsansatz aussieht. Das präskriptive Modell zeigt dagegen den SOLL-Zustand, also wie wir den Vorgang mit der Einbindung unseres Systems verbessern wollen.

Kommunkationsmodell deskriptiv:



Kommunikationsmodell präskriptiv:



Man kann erkennen das im präskriptiven Modell die Instanz des Dolmetschers komplett wegfallen und von unserem System übernommen werden würde. Zudem übernimmt das System die Weiterleitung und direkte Kommunikation zu den externen Diensten Währungsrechner und Online-Bezahldienst, was dem Nutzer einige Interaktionsschritte ersparen würde.

Methodischer Rahmen:

User-Centered Design

Im User-Centered Design stehen Benutzer bzw. Benutzergruppen im Fokus. In Bezug auf die kommende Phase des Projekts, stellt sich für die Entwickler die Frage, in welchem Ausmaß auf die individuellen Bedürfnisse der Nutzer eingegangen werden sollte.

ISO-Norm 9241- 210

In der ISO-Norm 9241 Teil 210 wird das Vorgehen auf einer strategischen Ebene in wechselseitig abhängigen menschzentrierten Gestaltungsaktivitäten dargestellt. Die Nutzungsanforderungen, die z.T. bereits aus den ermittelten Erfordernissen und Erwartungen resultieren, nehmen hier eine zentrale Rolle ein. Denn gemessen an der Erfüllung dieser Anforderung werden mindestens die Erarbeitung von Gestaltungslösungen und deren Evaluation iteriert. Für die Entwicklung des Systems hieße dies vor allem eine intensivere Auseinandersetzung mit Backpackern vorzunehmen, um den Nutzungskontext präzise eingrenzen und Nutzungsanforderungen ableiten und gewichten zu können.

Usability Engineering Lifecycle

Der “Usability Engineering Lifecycle” nach Deborah Mayhew besteht aus drei Phasen deren Aktivitäten die Anforderungsanalyse, das Design, Testen, die Entwicklung und zuletzt die Installation sind. Das Vorgehensmodell entspricht im Allgemeinen den Grundsätzen der o.g. ISO-Norm, da die erste Phase Anforderungen fokussiert, deren Erfüllung ebenfalls Maßstab iterativ evaluierter Ergebnisse der dritten Phase ist. Es sei besonders zu betrachten, dass aus der Anforderungsanalyse sogenannte “Styleguides” entstehen. Diese geben präskriptiv vor, wie das Design in der zweiten Phase konkret gestaltet wird, damit die zuvor definierten Ziele (“Usability Goals”) erreicht werden. Zum einen verspricht der “Usability Engineering Lifecycle”, dass sich das System am Ende des Projekts gegenüber dem Benutzer mindestens erwartungskonform präsentiert. Zum anderen ist es aufgrund der ressourcenintensiven Erarbeitung der nötigen Artefakte schwer auf das Projekt zu skalieren.

Scenario-based Usability Engeneering

Für das Scenario-based Usability Engineering nach Rosson und Carrol kann vorweggenommen werden, dass es einen mindestens ebenso hohen Arbeitsaufwand für die Entwicklung des Systems bedeutet und dementsprechend auch schwer skalierbar ist. Die Hauptaktivitäten des Modells sind in die Analyse, das Design und das Prototyping, Evaluieren gegliedert. Anders als beim “Usability Engineering Lifecycle” steht bei der Analyse der Problemraum im Mittelpunkt der Betrachtung und die erstellten Artefakte in der Design-Phase haben ausschließlich deskriptiven Charakter. Der Vorteil für das Projekt,

lange in einer konzeptionell abstrakten Eben zu entwickeln, bestünde darin, besonders innovative kreative Designideen zu finden. Beispielsweise bei Gestaltlösungen für graphische Darstellungen. Jedoch weisen Szenarien ein hohes Defizit an Agilität und Modalität auf. Auch das Fehlen konkreter UI's wären für das Entwickeln der Funktionen im Systems unvorteilhaft.

Discount Usability Engineering

Das "Discount Usability Engeneering" nach Jakob Nielsen ist im Vergleich zu den o.g. Vorgehensmodellen bei Weitem nicht so komplex. Es erfordert von den Entwicklern weniger Aufwand und Ressourcen, setzt allerdings ein hohes Maß an Gestaltungsfähigkeiten voraus. Angefangen mit der Erstellung von Szenarien werden anhand dieser "UI-Mockups" gefertigt, anhand derer mittels der Evaluationstechniken "Simplified Thinkaloud" und "Heuristic Evaluation" der Grad der Gebrauchstauglichkeit gemessen wird. Diese drei Aktivitäten werden chronologisch wiederholt, ohne dass ein Abbruchkriterium erkennbar ist. Die Entwickler haben eine enorme Verantwortung, da sie selber unvoreingenommen einschätzen können müssen, inwieweit das Vorgehensmodell überhaupt auf das Projekt skalierbar ist und vor allem wie die Qualität der Ergebnisse aus der Evaluation zu bewerten sind. Die mangelnde Erfahrung der Entwickler im Bereich des angewandten Usability Engineering und die Nähe zum Projekt stellen für dieses ein hohes Risiko dar.

Usage-Centered Design

Das Usage-Centered Design verfolgt eine etwas andere Vorgehensweise. Es umfasst von vorneherein eine größere Benutzergruppe und nimmt damit keine Rücksicht auf spezielle Kenntnisse und Bedürfnisse eines spezifischen Benutzertypen. Der Werkzeugcharakter und die Funktionalität der Elemente einer Anwendung stehen im Mittelpunkt der Entwicklung.

Essential Models

Larry Constantine und Lucy Lockwood haben im wesentlichen drei essentielle Modelle in Beziehung gesetzt. Das Role Model, das Task Model und das Content Model. Diese werden durch das Operational Modell und das Implementation Model erweitert. Der Benutzer wird als Individuum betrachtet und nimmt gegenüber dem System eine definierte Rolle ein, sodass Rollen miteinander verglichen und in Beziehung gesetzt werden können. Für die descriptive Planung der Benutzung des Systems werden auf Szenarien verzichtet und stattdessen Use Cases modelliert, die im Task Model untersucht werden: Inwiefern kann das System Aufgaben übernehmen und wie stehen die Aufgaben zueinander. Im Content Modell wird die Modellierung konkret gemacht. Das heißt Tools und Objects werden aus den Aufgaben abgeleitet und festgelegt, mit Attributen und Funktionalitäten definiert. Das und

Einflüsse aus den Modellen zum Nutzungskontext (Operational Model) bilden die Basis für ein Implementation Model.
In Bezug auf das Projekt bieten die Essential Models gute Voraussetzungen für eine Arbeit die sich auszahlen ließe. Das strukturierte Anpassen des Systems an den Soll-Zustand, verläuft eng an der Schnittstelle zum Software Engineering. Die erwarteten Ergebnisse der Implementation sind dadurch bereits vorher gut abschätzbar. Das Risiko für das Projekt, dass die Anwendung letztendlich die Zielgruppe der Backpacker nicht erreicht und sich am Markt gegenüber einer größeren Nutzergruppe, aufgrund der hohen Konkurrenz nicht durchsetzen kann, ist allerdings ebenfalls gegeben.

Fazit

Das Projekt nähert sich einem für viele Menschen sensiblen Thema an, nämlich dem Umgang mit Geld. Nutzer erwarten von so einem System ein Format, indem Funktionen eindeutig, verlässlich und absolut fehlerfrei ablaufen. Das Usage-Centered Design ist für Systeme dieser Art automatisch eine gute Wahl, weil im Prinzip jedes Mitglied der modernen Gesellschaft bereits Kompetenzen im Umgang mit Geld hat. Dadurch herrschen gute Bedingungen, dass eine breite Nutzergruppe sich mit den Funktionen schnell vertraut machen und hohe Erwartungen an die Funktionalität erfüllt werden können.
Jedoch zeigt die Marktanalyse, dass viele Systemen existieren, die sowohl in der Breite, wie auch in der Tiefe allgemeine qualitativ hochwertige Lösungen bereitstellen. Deswegen konzentrieren sich die Entwickler auf die Zielgruppe der Backpacker. Backpacker sind sehr häufig alleine oder zu zweit unterwegs, um flexibel zu sein. Jedoch schließen sich Backpacker häufig spontan in Gruppen zusammen, wenn sie sich vor Ort treffen und Erlebnisse teilen wollen. Die oft interkulturellen und sozialen Unterschiede und die Tatsache, dass man sich nicht kennt, sind für den gemeinsamen Umgang mit Geld eine spezielle Herausforderung.
Mit einem Vorgehensmodell des User-Centered Design kann die Benutzergruppe genauer untersucht werden, sodass sie gemeinsamen Umgang mit Geld im besonderen Nutzungskontext nicht scheut. Die ISO-Norm 9241 Teil 210 ist auf das Projekt sehr gut skalierbar, da sie Freiraum für die Wahl angemessener Artefakt-Erstellungen lässt, solange diese vorgegebene Kriterien erfüllen und den definierten Richtlinien entsprechen. Es muss eine positive User Experience in der Evaluation gemessen werden, um mit dem System am Markt konkurrenzfähig zu sein. Die ISO-Norm 9241 Teil 210 Vorgehensmodell bietet dafür adäquate Orientierungspunkte.

Risiken:

Konnektivität:

Auf mobilen Clients und besonders auf Reisen kann eine stabile Internetverbindung nicht garantiert werden. Daher muss bei der Planung und Implementation dieser Aspekt besonders berücksichtigt werden.
Eine mögliche Lösung wären Offline-Features, welche die Daten lokal speichern bis eine stabile Verbindung registriert wird. Erst dann erfolgt die Synchronisation mit dem Server. Notfalls kann auch auf die P2P Technologie Bluetooth zurückgegriffen werden um vor Ort Daten auszutauschen und zu sichern.

Speicherplatz:

Trotz der heutzutage geringen Kosten von Speicherplatz ist er nicht unendlich groß. Man muss sich bei der Konzeptionierung Gedanken machen wie man diesen verwaltet. Vor allem mobile Clients haben noch sehr begrenzte Kapazitäten.
Dies kann gelöst werden, wenn veraltete Daten (z.B lange abgeschlossene Reisen oder Abrechnungen, inaktive Benutzer) aus der Datenbank gelöscht oder zumindest komprimiert werden während auf dem Client nur die aktuellsten Daten lokal gespeichert werden.

Webservices:

Es gibt heutzutage eine riesige Auswahl an verschiedensten Webservices (Datenbanken, Karten, sonstige Cloud-Services), die zum Teil kostenfrei in eigene Projekte integriert und genutzt werden können. Hier ist jedoch zum einen der Datenschutz und die Verfügbarkeit zu berücksichtigen. Wie geht man damit im wenn damit um, wenn der Dienst ausfällt oder die Nutzungskapazität überschritten wird?
Daher sollte man die wichtigsten Features selbst implementieren und nur auf Webservices zurückgreifen, auf die man Notfalls auch verzichten kann.
In unserem Fall wären Interessante Webservices Paypal, der Währungsrechner currencylayer und ev. Karten wie Goolge Maps oder OpenStreetMap. Sie wären jedoch nicht integral für die Anwendung und könnten auf andere Anwendungen ausgelagert werden.

OCR:

Die Technik OCR (optical character recognition) wird in unserem Projekt für den Scan von Kassenzetteln eingesetzt. Obwohl an dieser Technik bereits seit einigen Jahren geforscht wird gibt es keine vollständige Garantie das alle Daten korrekt erkannt werden, da viele externe Faktoren (Licht, Oberfläche, Abnutzung, etc) eine Rolle spielen und das Ergebnis verfälschen können.
Daher muss dem Nutzer die Möglichkeit gegeben werden, mögliche falsch erkannte Daten manuell zu ändern oder gegebenenfalls alle Daten manuell einzutragen.
Zudem ist die Technologie ziemlich komplex und es gibt im Team keine direkten Vorkenntnisse, was die Implementation erschweren könnte.

Datenbanken/Querying:

In unserer Anwendungen sollen viele verschiedene und einzelne Daten (Bilder, Preise, Routen, etc) bearbeitet und abrufbar sein. Je größer der Datensatz wird bzw. je komplexer die Daten werden, könnten Komplikationen (lange Ladezeiten etc) auftreten Daher ist eine Datenbank nötig die von der Größe skalierbar ist das Querying erleichtert. Es muss eine Wahl zwischen SQL und No-SQL gemacht werden.

Abrechnung :

Wir versuchen zwar mit unserer Anwendung den Prozess der Abrechnung gemeinsam gekaufter Artikel zu vereinfachen, aber es besteht immer noch die Gefahr, dass Unstimmigkeiten über die Korrektheit der Abrechnung zwischen den involvierten Nutzern aufkommen.
Daher muss bei der Implementation Wert auf die Richtigkeit des Algorithmus (der die Abrechnung berechnet), eine transparente Erhebung der Daten und ein einfaches und klar überschaubares Design gelegt werden

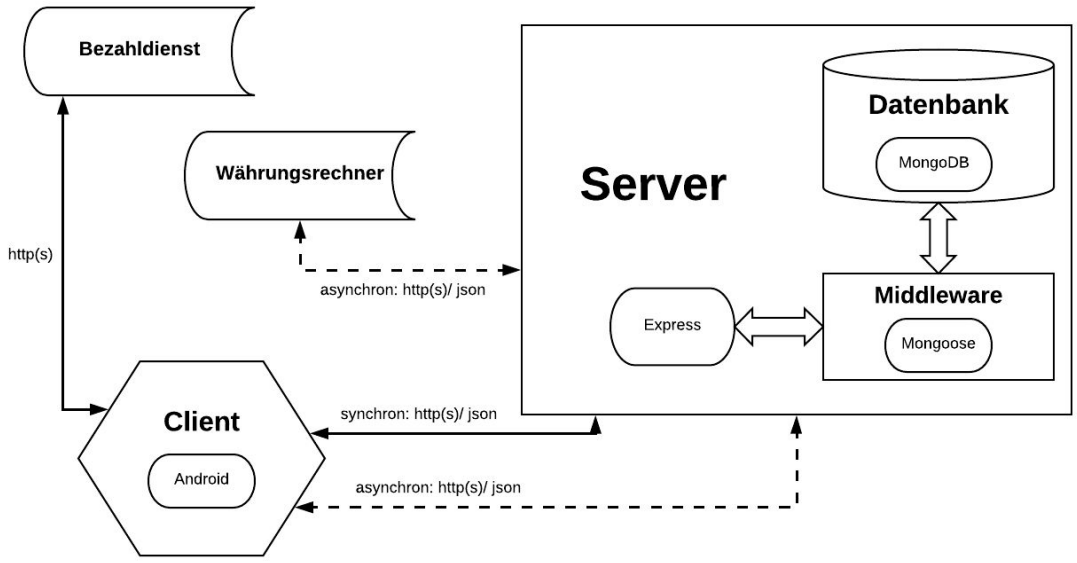
Arbeitsaufwand:

Aufgrund der hohen Anzahl an verschiedenen Features und Anforderungen an das System könnte der Arbeitsaufwand überschätzt werden und im Rahmen dieser Veranstaltung nicht realisierbar sein.
Daher sollten bestimmte Features der Anwendung anhand von der Ausprägung des Alleinstellungsmerkmals und den Voraussetzungen der Veranstaltung priorisiert werden

Proof Of Concept:

folgt

Systemarchitektur:



Server:

Um den Datenaustausch zwischen Clients, Datenbank und Webservice zu garantieren und gleichzeitig diese Daten qualitativ anreichern zu können ist ein zentraler Server nötig.
Für die Implementierung verwenden wir die Plattform Node.js. Zum Einen eignet sie sich für das Projekt aufgrund ihrer Performanz und Skalierbarkeit, als auch Dank der guten Dokumentation und dem großen Umfang an integrierbaren Modulen, welche durch npm zur Verfügung stehen. Außerdem bietet sie asynchrone Funktionen an, welche für uns eine große Rolle spielen. Als Programmiersprache wird, aufgrund bereits vorhandener Erfahrungen, Javascript verwendet.

Client:

Die Clients haben die Anforderung mobil zu sein, um vor allem auf Reisen genutzt werden zu können. Da heutzutage so gut wie niemand mehr ohne Smartphone das Haus verlässt, soll er in Form einer Smartphone-Applikation implementiert werden.
Zudem hat der Client die Anforderungen performant zu sein und Offline-Funktionen zu beinhalten. Daher muss die Anwendung nativ sein.
Dafür stehen bestimmte Betriebssysteme zur Verfügung: Android, iOS und Windows.
Unser Team besitzt bereits Vorkenntnisse im Bereich Android-Development und Android ist derzeit das den meisten verkauften mobilen Endgeräten vorinstalliert (87,7% Marktanteil).
Daher fällt die Wahl im Rahmen dieser Veranstaltung auf Android, wobei eine zukünftige

Implementierung auf allen Betriebssystemen wünschenswert wäre, um möglichst viele Nutzer anzusprechen.
Da die Android-Dokumentation für die Sprache Java verfasst wurde und das Team darin die größte Erfahrung hat, verwenden wir Java für die Implementation des Clients.

Datenbank:

Für die persistente Speicherung der Daten wird MongoDB, eine dokumentenbasierte NoSQL-Datenbank, verwendet. Sie bietet sich unter anderem aufgrund ihrer einfachen Einbindung in Node.js, als auch durch ihre hohe Performanz und effektivem Querrying an. Zum anderen lässt sich die Form unserer Daten (Kassenzettel, Nutzerdaten, etc) sehr gut in einem JSON-Format speichern, welches von MongoDB, in Form von BSON, unterstützt wird. Da MongoDB allerdings keine festen Schematas verwendet, kann jeder Eintrag einen anderen Datentyp besitzen. Um die Form der Daten trotzdem besser validieren und kontrollieren zu können nutzen wir die Middleware-Erweiterung Mongoose.

Dateisystem:

In unserem System soll nicht nur mit Key-Value Daten umgegangen werden, sondern auch mit Bildern. Deren Speicherung in JSON bzw BSON-Format ist nicht optimal, da das Bild in einen Blob von binären Daten dekodiert werden und bei jedem Zugriff wieder zusammengesetzt werden muss. Zudem benötigt dies mehr Speicherplatz. Die effektivere Lösung wäre ein Dateisystem neben der verwendeten Datenbank zu erstellen. Das Betriebssystem des Servers beinhaltet bereits ein solches und kann sehr einfach mit dem Node-Modul “fs” angesprochen und kontrolliert werden. Um die Datensätze dennoch mit Bildern verknüpfen zu können, reicht ein einfacher String mit dem Dateipfad in der Datenbank aus.

Protokolle:

Für die Kommunikation zwischen allen Systemkomponenten und externen Webservices soll das Protokoll HTTP / HTTPS verwendet werden. Es ist ein sehr verbreiteter und gut dokumentierter Standard im Internet, eignet sich für eine REST-Architektur und ermöglicht die Übertragung der von uns genutzten Dateiformate JSON und Bilder(jpeg, mpg, etc), sowie Authentifizierung der Nutzer.

Kommunikation:

Die Kommunikation zwischen Client und Server bzw den eingebunden Webservices kann entweder synchron oder asynchron verlaufen. Dabei blockiert synchrone Kommunikation den Client solange bis der Server seine Antwort sendet. Während die asynchrone zeitlich entkoppelt ist und dem Nutzer währenddessen Freiraum für andere Aktivitäten bietet.
synchron:
Für diese Art der Kommunikations eignen sich Vorgänge, bei denen Sicherheit eine Rolle spielt. Bei der Authentifizierung muss der Nutzer erst die Bestätigung vom Server abwarten, bevor der Zugang zu sensiblen Daten und Nutzungsrechten gewährt werden kann. Auch

beim Bezahlvorgang der Abrechnungen muss Synchronität bewahrt werden um mögliche invalide Zahlungen zu vermeiden.

asynchron:

Hier bieten sich alle restlichen (nicht sicherheitsrelevanten) Vorgänge an. Der Nutzer sollte nicht abwarten müssen, bis sein Upload des Kassenzettels beendet ist um andere Aktionen ausführen zu dürfen. Solange der Server den Vorgang registriert (Statuscode 202 Accepted) kann die Rückmeldung für erfolgreiche/fehlerhafte Speicherung der Daten auch später erfolgen. Dies funktioniert auch im Zusammenhang der Offline-Features (automatischer Upload sobald Verbindung besteht). Auch die Abfrage der Daten sollte asynchron verlaufen. So kann die UI automatisch aktualisiert werden, sobald sich die Daten auf dem Server ändern. Die Kommunikation und Interaktion der Nutzer im Bereich der Abrechnungen sollte mit einem PubSub-Service verbunden werden, so dass keine sofortige Rückmeldung aller involvierten Nutzer erforderlich ist.

Literatur-/Quellenverzeichnis:

folgt