Mobile Betriebssysteme und Netzwerke
Technical White Paper
Benjamin Schulz (594271) & Andreas Babij (586978)

Einleitung

Stellen Sie sich folgendes Szenario vor: Sie haben heute keine Lust, Ihr Abendessen selbst zu kochen, wollen allerdings auch kein Sandwich oder ähnliches essen. Demnach klingt auswärts nach einer sinnvollen Lösung - schwierig jedoch, sich in Berlin für ein Lokal zu entscheiden. Die Lösung für dieses Problem soll unsere App **„Match Your Munch“** vereinfachen.

Unsere App soll es Nutzern erleichtern, neue Restaurants zu entdecken, die Erfahrungen sämtlicher Mahlzeiten zusammenfassend zu notieren und das gleichzeitig mit dem Aspekt einer Art Social-Media-App kombinieren, um ggf. Freunden oder Fremden ihre Suche zu erleichtern.

Funktionen

- **Bewertungen und Notizen** hinzufügen und lesen
- **Freunde** hinzufügen, um vertraute Bewertungen zu sehen
- **Verlauf** von besuchten Restaurants einsehen und auf vorherige Bewertungen und Notizen zugreifen
- **Möglichkeit, Listen** per Bluetooth oder Wifi-Direct an Freunde zu verschicken
- Gute Restaurants **favorisieren**, um diese später wiederzufinden
- Restaurants im Offline-Modus mittels Formular **manuell** hinzufügen

Use-Cases

- Nutzer*innen möchten ihre **Bewertungen und Notizen speichern**, damit sie sich später an eigene Erfahrungen erinnern können.
- Nutzer*innen möchten **Bewertungen von Freunden sehen**, um basierend auf deren Empfehlungen eine Entscheidung zu treffen.
- Nutzer*innen möchten auch **ohne Internetzugang**:
    - Favorisierte Restaurants ansehen
    - Gespeicherte Notizen und Bewertungen lesen
    - Bewertungen schreiben und später synchronisieren

Kernelemente des GUI

- Hauptbildschirm:
    - Suchleiste, um nach Food-Spots zu suchen
    - Kartenansicht mit markierten Food-Spots
    - Liste von bewerteten Food-Spots
- Bewertungen:
    - Einfache Form zum Hinzufügen: Sterne, Texte, Notizen, Bildanhänge
    - Upvoting-System, um Top-Bewertungen möglichst vertrauenswürdig zu machen
- Freunde:
    - Freundesliste inkl. Bewertungs-”Feed”
    - Möglichkeit, Bewertungen direkt zu kommentieren oder zu liken
- Offline:
    - Heruntergeladene Food-Spots einsehen
    - Food-Spots per Formular manuell erstellen

Inwiefern unsere App den Anforderungen genügt:

- nicht-triviale GUI:
    - Interaktive Kartenansicht
    - Bewertungsfunktionen
    - Freunde
    - Benutzeroberfläche funktioniert on- und offline
- Offline-Funktion:
    - Lokale Speicherung von Daten, die später synchronisiert werden, sobald Netzwerkverbindung hergestellt wird
- MVC-Architektur:
    - Model: verwaltet Daten
    - Controller: Interaktion
    - View: Benutzeroberfläche
- Tests:
    - Unit-Tests für jede Funktion und Komponente
    - GUI-Tests für Benutzeroberfläche
    - Integrationstests: ordnungsgemäße Kommunikation zwischen den Modulen

Vertiefungen

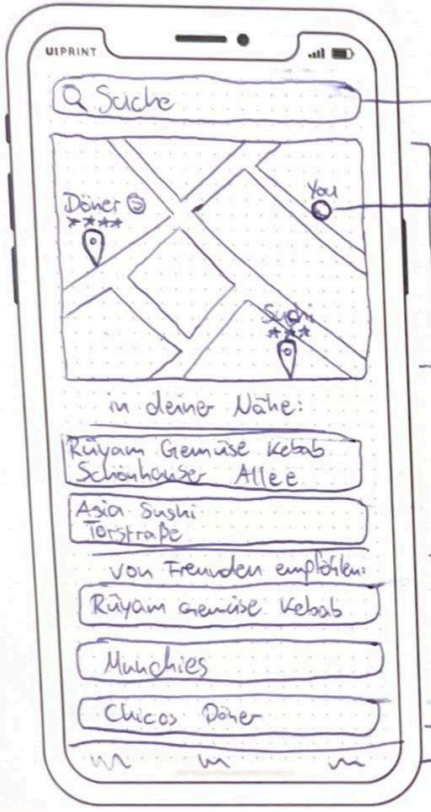
- Ad-hoc Netzwerke:
    - Möglichkeit, Daten über Ad-hoc Netzwerke zu teilen
    - Besonders relevant für Situationen ohne Internetverbindung
    - z.B. Versenden einer selbst erstellten Liste per Bluetooth oder Wifi-Direct
- Verteilte Daten:
    - Synchronisation der Daten
    - Datenbank für User und Restaurants
    - Lokal gespeicherte Daten werden beim nächsten Online-Zugang abgeglichen

Skizze des GUI (MainScreen)

Project Name: Match Your Munch

Date: 2025 SoSe

Main - screen



Food Spot finden

Standort

Karte mit Food-Spots in der Nähe
(Vorschlag nach: Angebot, Standort, Bewertung)
↳ z.B. „Rüyam Gemüse Kebab"
Angebot: Döner/Kebab
Standort: Schönhauser Allee 44A, 10435 Berlin
Bewertung: 4 von 5 Sterne
↳ Zusatz: Smiley - empfohlen von Freunden

von Freunden empfohlen

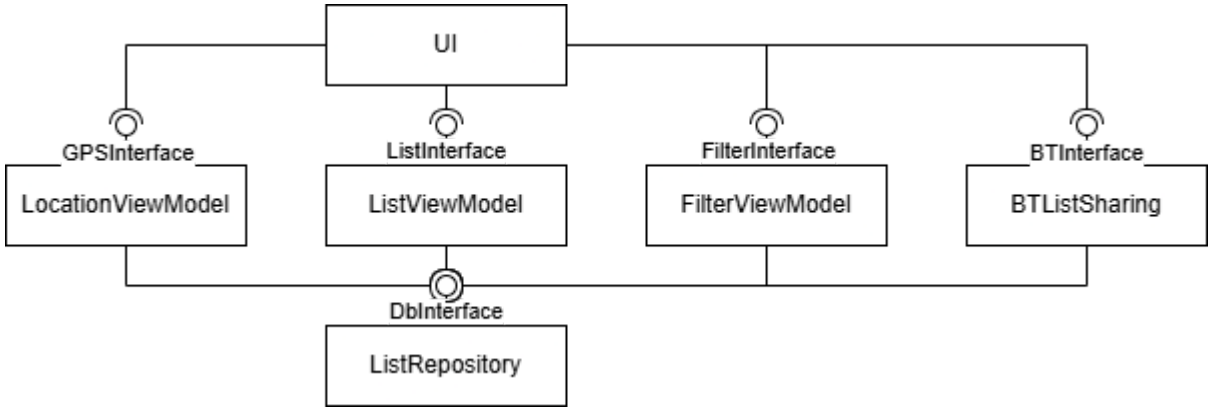
Menü-Leiste

IPHONE X, XS, 11 (375 x 812)

Projektstruktur

Package	Komponente	Klasse	Beschreibung
data	repo	ListRepository DbInterface	initialisiert die Datenbank mit allen Attributen & speichert diese
ui	detail	ListDetailScreen	Visualisierung der einzelnen “Listen”; können geöffnet werden und (später) mit Einträgen wie Notizen, GPS-Daten, etc. versehen werden; Bluetooth-funktion für das Teilen von Listen existiert bereits und kann eine vernünftige Verbindung aufbauen
	main	MainActivity	lädt das UI und die Datenbank, sobald die App geöffnet wird
	map	MapScreen	Visualisierung für Kartendarstellung inkl. Map-Marker für Positionsbestimmung
viewmodel	location	LocationViewModel GPSInterface	ViewModel mit Request für Anfrage/Abfrage der aktuellen Position inkl. Permission-Handling
	list	ListViewModel ListInterface	enthält funktionen zur Interaktion des Nutzers mit dem UI; Listen können erstellt gelöscht und versandt werden
	filter	FilterViewModel FilterInterface	enthält sämtliche Funktionen zur Transparenz in der Darstellung von (u.a. vielen) Einträgen in Listen
		BTListSharing BTInterface	stellt Funktionen bereit zum Austausch von Daten(Listen + Attribute) über eine Bluetooth-Verbindung zu anderen Geräten

Komponentendiagramm



Testplanung

- Komponententests → Unit Tests
- Funktionstests → Hardwaretests
- Barrierefreiheitstests → Hardwaretest oder Emulator
- Kompatibilitätstests → Emulator

Aktuelle Probleme und Schwierigkeiten

- Weiterführende GPS Tests werden erst einmal hinten angestellt aufgrund von fehlendem Android Gerät (ca. 1 Woche)
- Testweise Übertragung einer HTML Datei mit Textinhalt funktioniert zwar, allerdings funktioniert die Übertragung einer Liste noch nicht korrekt

<u>Meilenstein 4: Update</u>
Finales Produkt + Qualitätssicherung

<u>Vorbemerkung:</u> Leider hat sich mein Projektpartner dazu entschieden, das Modul nicht weiter fortzuführen. Die Struktur der Applikation bleibt weiterhin die gleiche und ich stelle sie alleine fertig.

<u>Aktueller Stand:</u>
- funktionstüchtige, nicht-triviale UI
- festgelegter Inhalt einer Liste: Name, Ersteller, Food-Spots
- Food-Spots sind Bestandteil der Liste und haben ebenfalls Attribute: Name, Adresse, GPS-Position, Bewertung, evtl. Kommentare + Fotos
- problemlose Verwaltung der List-Objekten in einer angebundenen SQLite Datenbank
- problemlose Bluetooth-Verbindung und -Übertragung von List-Objekten
- GPS-Verbindung funktioniert

- **Alles bisher nur über manuelle Tests geprüft.*

<u>Erkenntnisse:</u>
- der “Social-Media-Aspekt” wird aus Zeitgründen eher abgeschwächt integriert:
    - keine direkten “Freunde”, aber…
    …wenn Liste (via. Bluetooth) geteilt wird, wird mit ihr auch der “Inhaber” und seine Kommentare sowie Bewertung der Food-Spots geteilt

<u>Bis zum Release:</u>
- Implementierung automatisierter Tests (weitere Informationen zur Teststrategie folgen)
- Integration von GPS auf der Benutzeroberfläche (als Kartenansicht)
- Verbesserung und Feinschliff der UI (Icons, Farben etc.)

- Implementierung einer Funktion, dass nur ausgewählte Food-Spots geteilt werden können ohne dabei die gesamte Liste zu senden
    - bei Empfang wird der Empfänger gefragt in welcher Liste er diesen speichern möchte

<u>Teststrategie:</u>

Art	Inhalt	Tools
Unit-Tests	View-Model-Logik: Gültigkeit von Eingaben: - *Gültiger Listenname: (darf nicht leer sein)* - *Gültiger Food-Spot-Eintrag:*	JUnit

	(darf nicht leer sein, muss eine Bewertung und Adresse enthalten)
	Fehlermeldungen:
	- *Korrekte Fehlermeldung für:*
	  - *ungültige Namen*
	  - *fehlende Einträge im Food-Spot*
	Richtiges Speichern von Listen
	Richtige Verwaltung von Listen	

UI-Tests	Oberfläche: Buttons Dialoge Formate Fehleranzeige Pop-Ups GPS-Karte	Compose UI Testing
Integrationstests	Korrekte Verwaltung von Listen in der Datenbank (speichern, löschen, schreiben, lesen) Richtige Übernahme von Listen via Bluetooth Übertragung der GPS-Position	InstrumentationTest, ggf. Robolectric
Manuelle Tests	Bluetooth (Verbindungsaufbau, Senden, Empfangen, Verarbeiten) GPS (Position erkennen, verarbeiten, anzeigen)	Über den Emulator aus Android Studio und mit Hardware (Screenshots werden in der Dokumentation hinterlegt)

Probleme:

- Bluetooth und GPS sind Komponenten von Android: Der Verbindungsaufbau bzw. die Positionserkennung findet extern außerhalb der Software auf dem Betriebssystem statt, weshalb das automatisierte Testen hier nur eingeschränkt möglich ist. In den Integrationstest wird aber die korrekte Verarbeitung dieser Daten innerhalb der App getestet.

Match Your Munch

Technical White Paper

Release

Benjamin Schulz 594271

Einleitung:

Match Your Munch ist eine Art „Food-Spot-Manager“, die es Usern vereinfacht, einen Überblick über gespeicherte Food-Spots zu haben und diese gleichzeitig organisiert zu Verwalten.

Kernfunktionen:

Um dieses Konzept zu verwirklichen bietet die App folgende Funktionen:

Listen werden erstellt, können bearbeitet werden, oder auch wieder gelöscht werden. Innerhalb einer Liste werden Food-Spots erstellt und organisiert, welche ebenfalls bearbeitet werden können.

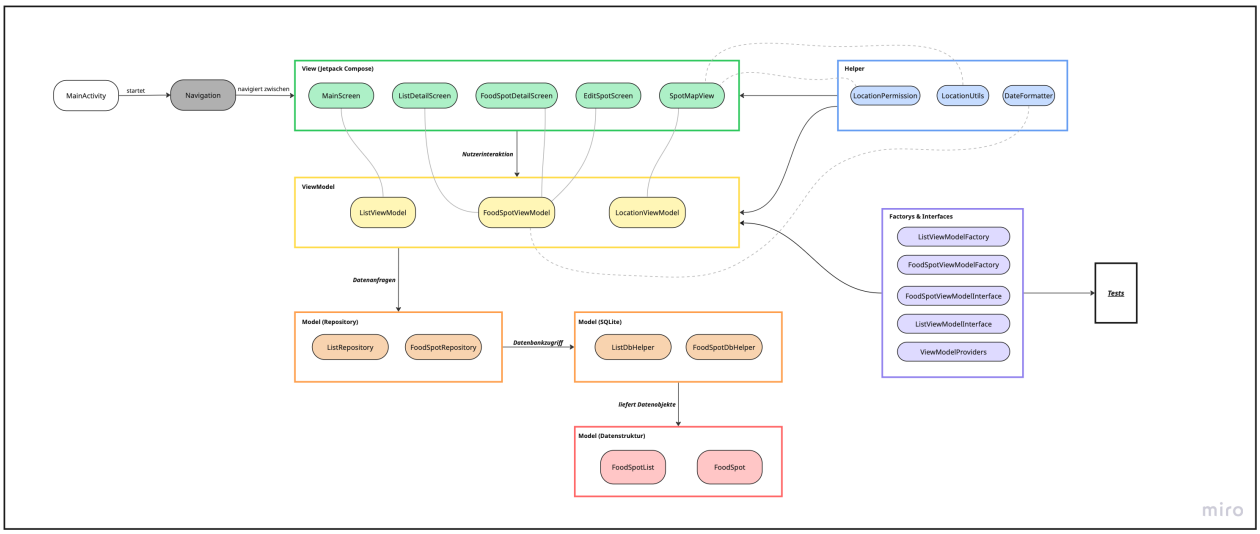
Ein Food-Spot stellt das Kernelement da. Diese werden mit einem Namen, einer Adresse und einer Bewertung versehen und können optional mit einem Kommentar, sowie einer Kategorie und einer Speisekarte ergänzt werden.

Anhand der Adresse wird ein geforderter Schwerpunkt durch die Verwendung von Standortdiensten verwirklicht.

Aufgrund der Anbindung von SQLite Datenbanken bleibt die App außerdem offlinefähig.

Projektstruktur & Komponenten:

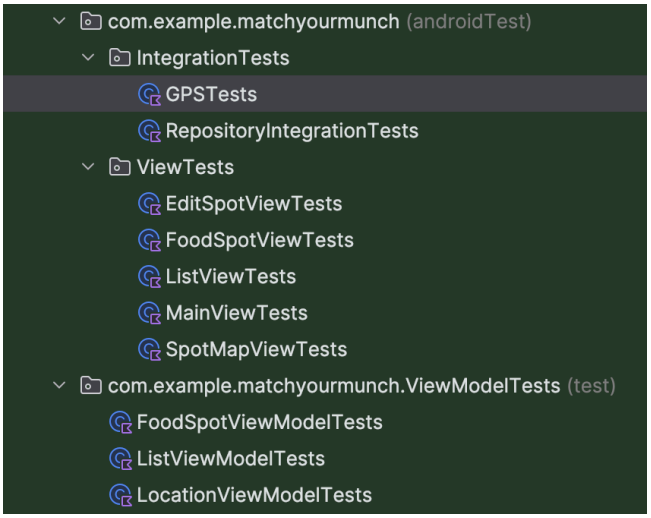
Die Projektstruktur verfolgt das MVVM-Architekturmuster.



Die MainActivity startet hier die Navigation, welche durch die verschiedenen Views navigiert. Interagiert der Nutzer mit den Views, so werden diese Interaktionen an die ViewModels weitergeleitet. Die ViewModels können dann ggf. als Schnittstelle fungieren und auf Repositories oder DbHelper zugreifen, oder selbst bereits Daten zur visuellen Aktualisierung verarbeiten. Die

Repositories und DbHelper selbst verwalten die Listen und FoodSpots als Modells innerhalb der Datenbanken. Die Helper-Klassen stellen einige Hilfsfunktionen dar, welche ausschließlich zur Unterstützung von Views oder ViewModels dienen. Factorys und Interfaces wurden hauptsächlich zur Integration von ViewModels in Tests implementiert um entsprechende Use-Cases zu simulieren, stellen aber ebenfalls sinnvolle Schnittstellen im Zusammenhang mit den ViewModels dar.

Teststrategie:



Die Komponenten habe ich in folgender Weise auf Funktionalität und Korrektheit getestet:

Für die ViewModels habe ich UnitTests geschrieben, die die Korrekte Arbeit der Funktionen, Verarbeitung der Daten und Kommunikation mit den Repositories und DbHelpern überprüfen.

Mit Integrationstests habe ich die Datenbanklogik und Standortfunktion überprüft. Der Schwerpunkt lag dabei auf der korrekten Arbeit mit dem Standortdienst sowie der standortbezogenen Funktionen. Bei der Datenbanklogik habe ich mich auf das lesen und schreiben in und aus der Datenbank fokussiert und habe zusätzlich die Arbeit mit den Datenbanken überprüft.

Um die Benutzeroberfläche zu testen habe ich mit Jetpack Compose UI-Tests geschrieben, welche die Darstellung und Funktion der einzelnen Elemente überprüfen.

Zum Thema Bluetooth:

Eigentlich hatte ich geplant und abgesprochen, mich trotz der nun alleinigen Arbeit an dem Projekt mich auf beide geplanten Schwerpunkte (GPS + Bluetooth) zu fokussieren.
Leider musste ich aber schnell feststellen, dass aufgrund der immer komplexeren Projektstruktur, mein bereits funktionierender Bluetooth-Verbindungsaufbau und das Teilen von Listen nun doch nicht mehr funktionierte. Stattdessen wurde nun nichts mehr versandt, und nach mehreren Stunden Fix-Arbeit wurde sogar gar keine Verbindung mehr aufgebaut.

Darum letztendlich der Entschluss: Kein Bluetooth mehr und Fokus auf GPS.

Das vollständige Projekt finden sie auf GitHub unter folgendem Link:

https://github.com/benjaminschulzhtw/MatchYourMunch