

E-Learning ASP

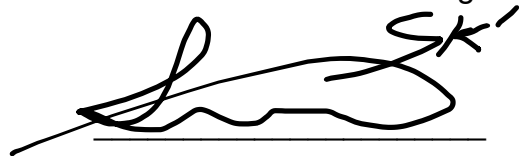
Entwicklung einer Crossplattform-App mit Flutter

Modulnummer: 6FSC0XD102 (BSc)
Modulname: Creative Studio 3: Research and Practice
Abgabedatum: 08.02.2024
Abschluss: Bachelor of Science (Hons.)
Web Development
Semester: September 2023
Name: Sven Richard Kamm
Standort: SAE Institute Zürich
Land: Schweiz
Lesezeit: ca. 15 Minuten

Hiermit bestätige ich, dass ich die vorliegende Arbeit eigenständig erarbeitet habe und alle Hilfsmittel sowie Inhalte von Dritten vollständig angegeben wurden. Hierunter fällt zum Beispiel die korrekte Belegarbeit für die direkte oder sinngemäße Übernahme von Texten, Bild-, Audio- und Videomaterial sowie Code etc. Weiterhin die klare Kennzeichnung von Inhalten, die von anderen Personen oder technischen Hilfsmitteln wie z.B. einer künstlichen Intelligenz erstellt wurden.

Baldingen, 08.02.2024

Ort, Datum

A handwritten signature in black ink, appearing to be 'SK' followed by a stylized flourish.

Unterschrift Student

Inhalt

1. Vorwort.....	1
2. Recherche.....	2
3. Erste Schritte.....	3
4. Weiteres Vorgehen.....	5
5. Von der Idee zum Appstore.....	6
6. Eindrücke beim Programmieren.....	10
7. Der harte und steinige Weg zum Ziel	12
8. Schlussfazit.....	12
9. Quellenverzeichnis.....	14

Abbildungsverzeichnis

Abbildung 1: Finales Design meiner Toycarsaddict APP (eigene Darstellung).....	1
Abbildung 2: Youtuber Kevin Chromik (eigene Darstellung).....	2
Abbildung 3: React Native vs Flutter erklärt von Fireship (eigene Darstellung).....	3
Abbildung 4: Flutter und Dart, das umfassende Handbuch (eigene Darstellung) ...	5
Abbildung 5: Intro-Bildschirm der ToycarsaddictApp (eigene Darstellung).....	6
Abbildung 6: Featured Section der ToycarsaddictApp (eigene Darstellung)	7
Abbildung 7: List Section der ToycarsaddictApp (eigene Darstellung)	8
Abbildung 8: Listenelemente der ToycarsaddictApp (eigene Darstellung)	8
Abbildung 9: Das Aside Menü der ToycarsaddictApp (eigene Darstellung)	9
Abbildung 10: Readme Dokumentation auf Github (eigene Darstellung)	10
Abbildung 11: Kapitel 2 DIE Entwicklungsumgebung (eigene Darstellung).....	11

1. Vorwort

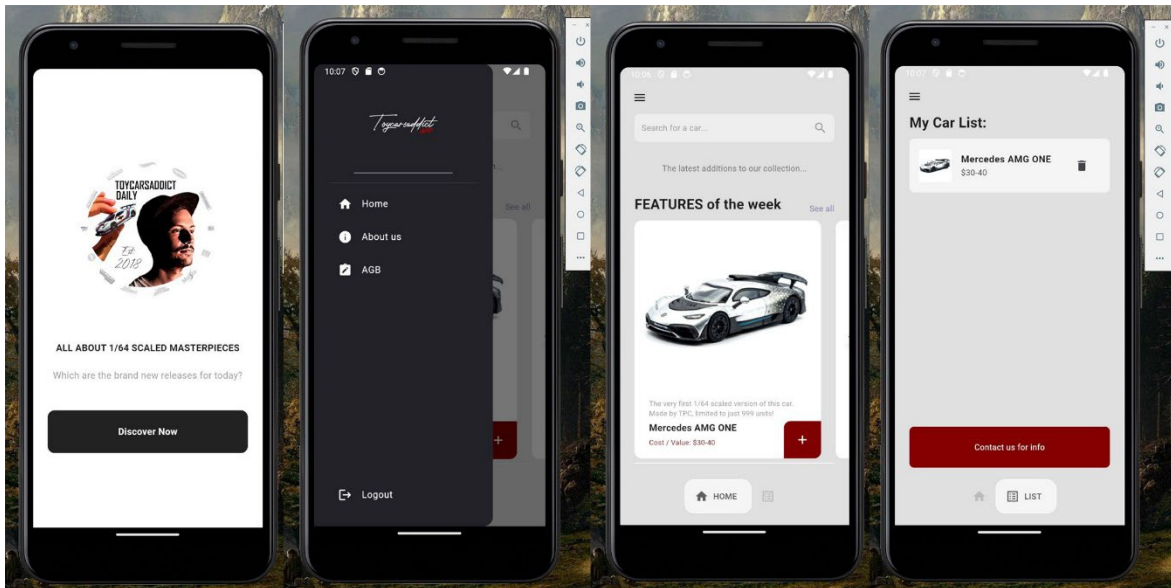


Abbildung 1: Finales Design meiner Toycarsaddict APP (eigene Darstellung)

Seit geraumer Zeit hegte ich den Wunsch, eine eigene App zu programmieren oder zumindest einmal zu erfahren, wie es sich anfühlt, eine mobile Applikation entwickelt zu haben. Mein Weg führte mich durch die unendlichen Weiten von HTML, CSS, Javascript und weiteren Sprachen, mit denen ich üblicherweise Webseiten gestalte. Doch nun war es an der Zeit, einen Schritt weiterzugehen – hinein in die Welt der mobilen Webentwicklung.

Zentral stand für mich dabei die Frage: Wie geht das eigentlich?

Von Beginn an war klar: Nicht alles kann nach Plan verlaufen. Ich war mir bewusst, dass es keine leichte Aufgabe sein würde. Trotzdem war es mir wichtig, durch das gesamte Projekt hindurch eine positive Einstellung zu bewahren, selbst wenn ich manchmal kurz vor dem Nervenzusammenbruch stand (mehr dazu später). Mit praktisch keinen Vorkenntnissen eine App selbst zu programmieren und das auch noch in einer völlig neuen Sprache – warum stellt man sich dieser Herausforderung? Ist der Aufwand es wirklich wert?

2. Recherche

Um einen strukturierten Weg zu finden, habe ich mich gründlich informiert. Das würde ich jedem empfehlen, der nicht planlos und ziellos versuchen möchte, etwas zu erreichen, und dabei riskiert, gnadenlos zu scheitern.

Mein größter Antrieb war der Ehrgeiz, Neues zu lernen und ein beeindruckendes Projekt für mein Portfolio zu schaffen. *Ein erster Erkenntnis- Tipp, den ich gleich zu Beginn geben möchte: Stürzt euch nur in solche Vorhaben, wenn ihr ein klares Ziel vor Augen habt, welches ihr um jeden Preis erreichen möchtet! «Komme was wolle!»*



Abbildung 2: Youtuber Kevin Chromik (eigene Darstellung)

Daher habe ich mir Videos von einigen Youtubern angesehen. Meine Hauptinspirationsquellen und Motivatoren sind Youtuber wie Kevin Chromik, der uns Neulingen alles über die mobile Webentwicklung näherbringt. Ich folge ihm schon lange, denn er spricht direkt aus der Seele eines Entwicklers. Wir erfahren von seinem Werdegang, erhalten alle seine Tipps und Tricks und bekommen wertvolle Ratschläge aus seinem Entwickleralltag. So bin ich beispielsweise auf sein Video "Apps

programmieren mit Flutter & Co - Was ist besser?" gestoßen. (Eine genaue Auflistung aller hier erwähnten Quellen sind im Kapitel 9 Quellenverzeichnis vermerkt).

3. Erste Schritte

Nun erlangte ich Einblick in die Unterschiede zwischen nativen Apps und Cross-Plattform-Lösungen. Kurz gesagt geht es darum, entweder eine App spezifisch für eine Plattform wie iOS oder Android zu entwickeln (native programming) oder eine App zu gestalten, die auf beiden Plattformen gleichermaßen funktioniert (Cross-Plattform programming).

Die Entscheidung hängt grundsätzlich von verschiedenen Faktoren ab: der Komplexität der App, den Kosten, dem Benutzererlebnis (Funktionen usw.) und natürlich der Entwicklungszeit. In einem Unternehmenskontext sind dies entscheidende Überlegungen. Als Neueinsteiger mit dem Ziel, eine mobile App rund um das Thema Modellautos zu entwickeln, ist man allerdings offener für verschiedene Ansätze.

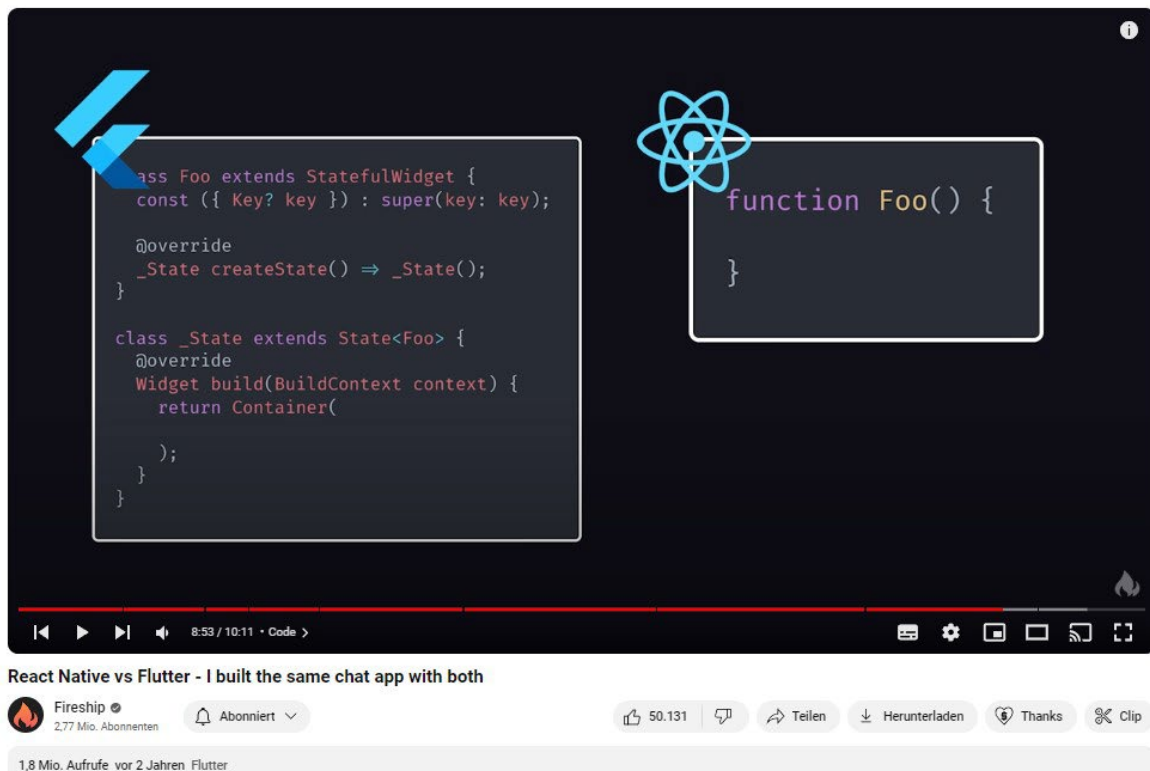


Abbildung 3: React Native vs Flutter erklärt von Fireship (eigene Darstellung)

Ich vertiefte meine Recherche und stieß auf weitere Videos und Online-Ressourcen. Fireship, eine der größten Plattformen und YouTube-Kanäle rund um das Thema Programmierung, veröffentlichte beispielsweise ein Video mit dem Titel "React Native vs Flutter - I built the same chat app with both". Das ermöglichte mir, die Vor- und Nachteile beider Frameworks zu erkennen. React Native, ursprünglich von Facebook entwickelt, liegt nah an JavaScript und TypeScript und bietet eine ausgezeichnete Dokumentation.

Flutter hingegen wurde von Google entwickelt und brachte sofort Gedanken von Nutzern in den Kommentarspalten hervor wie: *"Ich wollte zuerst React Native nutzen, bis ich feststellte, dass es von der 'bösen' Firma Facebook stammt. Dann wollte ich zu Flutter wechseln, bis ich realisierte, dass es von einer anderen 'bösen' Firma stammt, nämlich Google."*

Dennoch konnte ich viel Positives über Flutter gewinnen. Beispielsweise ist Flutter für seine Leistungsstärke bekannt (direktes Kompilieren in Maschinencode, was zu einer besseren Performance führt). Es gibt eine engagierte Community mit umfassender Unterstützung. Viele Komponenten wie Navigation und Footer sind bereits vorhanden und schnell aus einer Bibliothek von Google implementierbar.

Aber der ausschlaggebende Punkt für mich war, dass ich all dies ja für mein Advanced Specialised Project (kurz ASP) an der SAE Zürich mache. Mein Ziel ist es hier, Wissenslücken zu schließen, mein Portfolio um eine weitere professionelle Komponente zu erweitern und einfach mehr über dieses Spezialgebiet zu erfahren. Hier kommt der entscheidende Faktor für Flutter ins Spiel: *Ich möchte tiefer in ein Thema eintauchen, das in meiner bisherigen Laufbahn als Schüler oder Entwickler noch nicht behandelt oder vertieft wurde. React und JavaScript habe ich zwar schon berührt, aber in anderen Kontexten. Mit Flutter und der dazugehörigen Sprache Dart betrete ich absolutes Neuland!*

So konnte ich mich schließlich für die Sprache entscheiden, die ich für mein Projekt verwenden wollte. *Mein zweiter wichtiger Erkenntnis-Tipp für euch: Wägt alle Vor- und Nachteile einer Sprache sorgfältig ab und vergleicht sie mit den Anforderungen eures Projekts. Gibt es ausreichend Forschungsmaterial, Bücher oder Videos dazu? Oder eine solide Dokumentation?*

4. Weiteres Vorgehen

Für den Fortgang meines Projekts habe ich mir passende Literatur in Form von gedruckten Büchern besorgt, darunter "App-Entwicklung mit Flutter für Dummies" von Mira Jago und Verena Zaiser, "Cross-Platform UIs with Flutter: Unlock the ability to create native multiplatform UIs using a single code base with Flutter 3" von Ryan Edge und Alberto Miola sowie mein unverzichtbarer Leitfaden: "Flutter und Dart - Das umfassende Handbuch" von Marc Marburger. Letzteres bietet auf ungefähr 651 Seiten umfangreiches Wissen, das sowohl für Einsteiger als auch für fortgeschrittene Flutter-Entwickler von großem Wert ist!

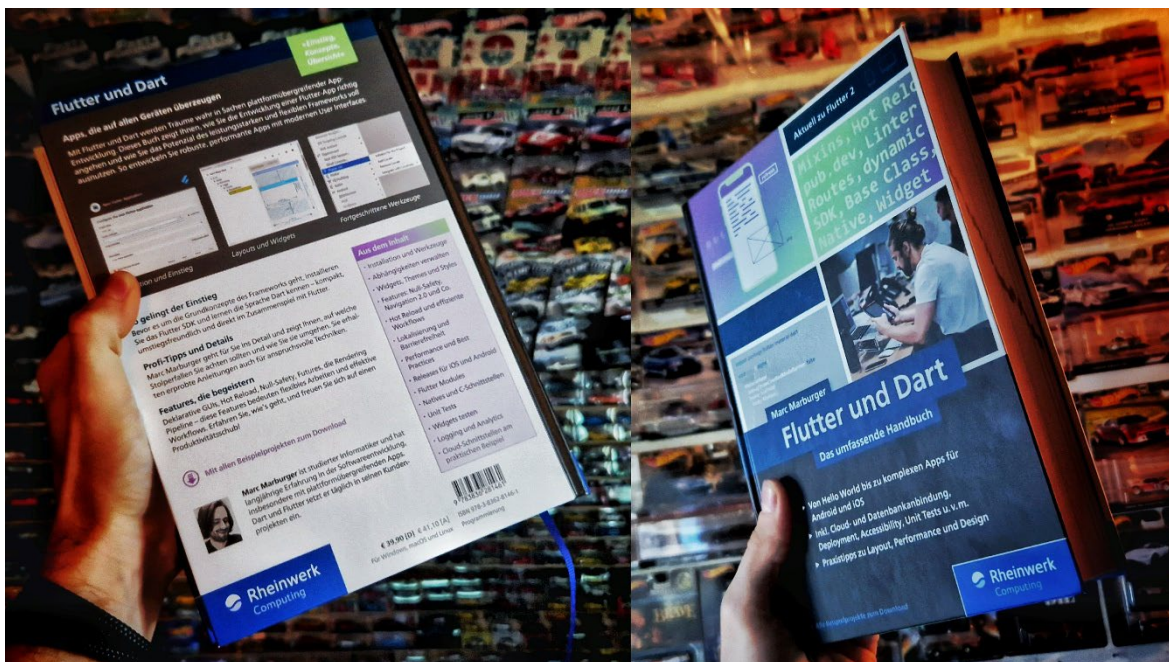


Abbildung 4: Flutter und Dart, das umfassende Handbuch (eigene Darstellung)

Am effektivsten in Bezug auf die Lernmethode erwies sich die Kombination aus Fachliteratur, Online-Artikeln und YouTube-Videos. Die Bücher lieferten das grundlegende Fachwissen, während die Webseiten konkrete Lösungsansätze und Denkanstöße präsentierten und die YouTube-Videos als Quelle der Inspiration dienten.

Ein Erkenntnis-Tipp dazu: Findet heraus, wie das Lernen sowohl effizient als auch unterhaltsam gestaltet werden kann! Mir war bewusst, dass Flutter ein recht trockenes Thema sein würde, zumindest zu Beginn. Daher war es mein Ziel, mich nicht nur auf Bücher und Texte zu beschränken. YouTube-Videos allein sind oft zu oberflächlich, deshalb erscheint eine ausgewogene Mischung aus all diesen

Komponenten als ein optimaler Weg, um dem Ziel, eine eigene App zu entwickeln, näher zu kommen.

5. Von der Idee zum Appstore

Mein Vorhaben ließ sich kurz und knapp anhand des Titels dieses Kapitels zusammenfassen, doch die Umsetzung erwies sich als weit komplexer als erwartet. Meine Vision war klar: Ich wollte ein Prototyp entwickeln, der wie ein Newsletter agiert. Beim Starten der App begrüßt ein Willkommensbildschirm den Nutzer:

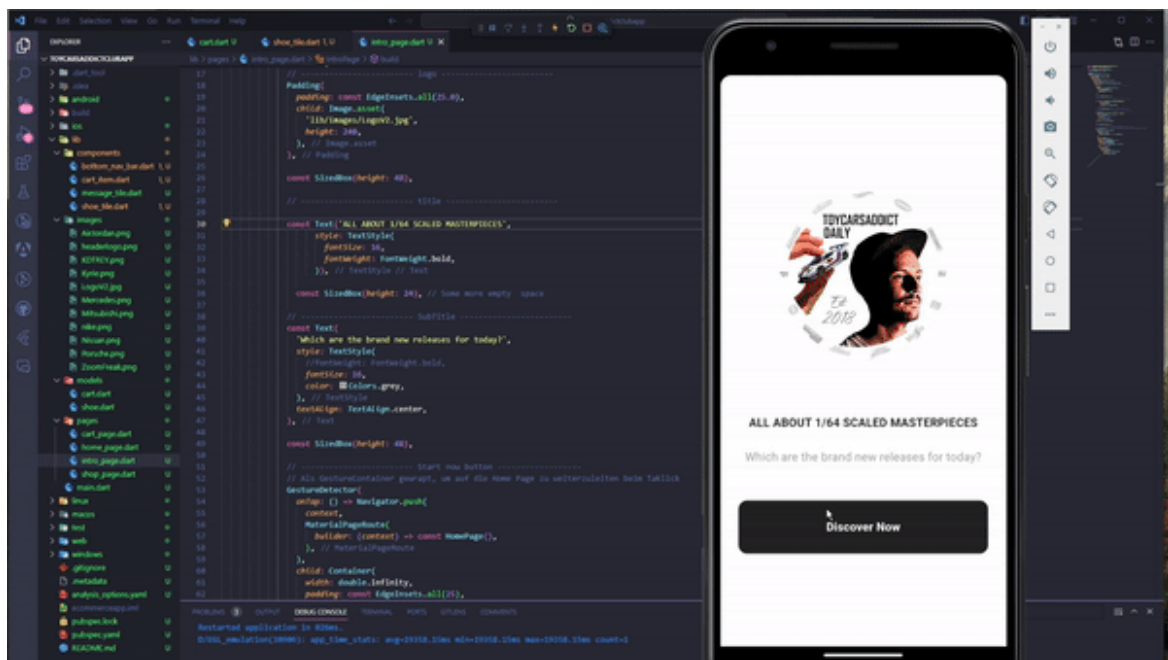


Abbildung 5: Intro-Bildschirm der ToycarsaddictApp (eigene Darstellung)

(Die Abbildungen 5 – 9 sind im Word-Dokument als GIF animiert, das PDF unterstützt diese Funktion nicht. Die Abbildungen sind jedoch in animierter Form im GitHub Repository vorzufinden, Link auf Seite 9. Zudem sind die Animationen in einem separaten Unterordner namens «Animationen» abgespeichert).

Innerhalb der App werden die neuesten Automodelle präsentiert – zum einen aus meiner eigenen Sammlung aber auch solche, die bald in unserem separaten Online-Shop auf Doramas.ch erhältlich sein werden. Jedes Fahrzeug wird mit einem Bild, einem Titel und einer Preisangabe vorgestellt:

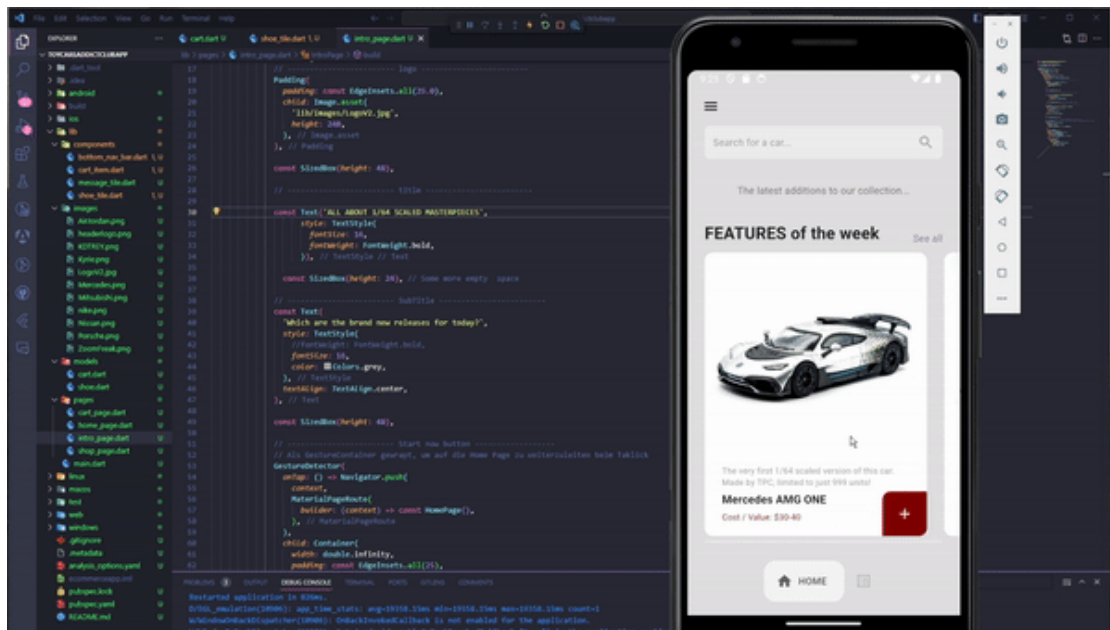


Abbildung 6: Featured Section der ToycarsaddictApp (eigene Darstellung)

Auf der Home-Seite haben die Nutzer die Möglichkeit, Modelle ihrer Wahl per Klick einer Liste hinzuzufügen und erhalten dabei Feedback:

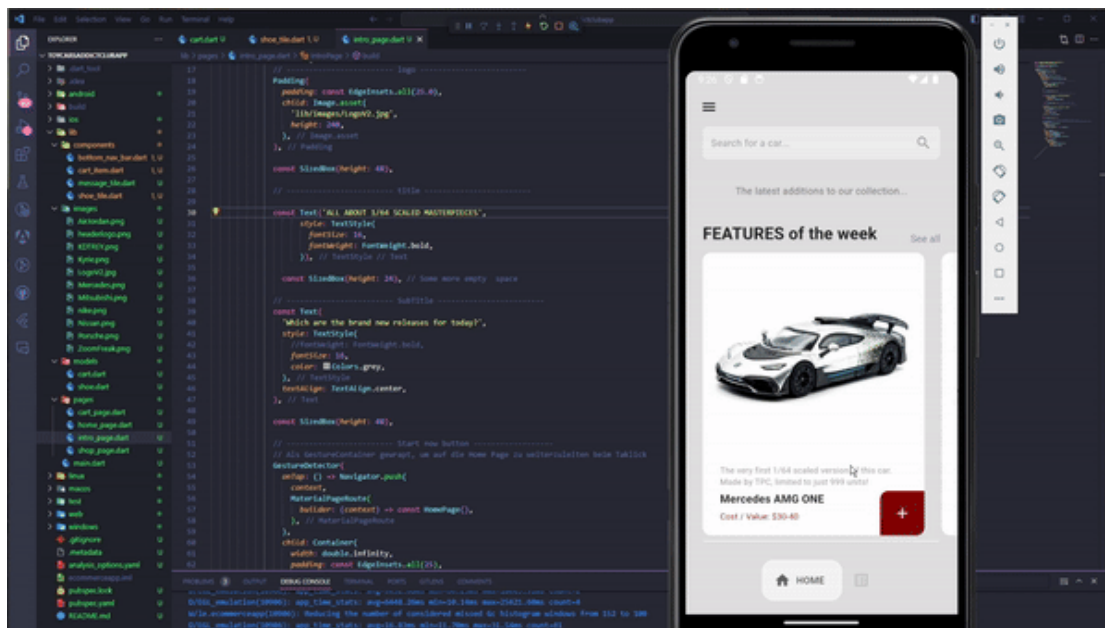


Abbildung 7: List Section der ToycarsaddictApp (eigene Darstellung)

Ist die Auswahl abgeschlossen, kann die Liste mit einem spezifischen Betreff an mich gesendet werden, um die Organisation der neuesten Modellautos für die Kunden zu erleichtern. Die Modelle können aber auch wieder von der Liste gestrichen werden. Die Nutzer erhalten dabei ein Feedback:

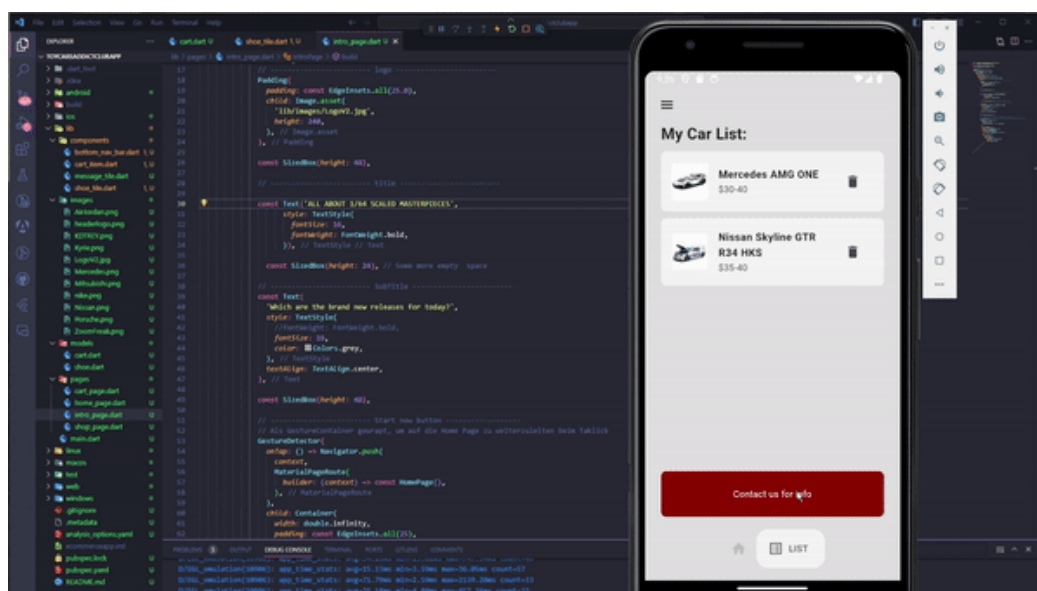


Abbildung 8: Listenelemente der ToycarsaddictApp (eigene Darstellung)

Zudem gibt es ein Aside-Menu, wo sich der Home-Button, die About Us-Unterseite, AGB und Logout befinden. Diese sind aber rein kosmetischer Natur und haben aus Zeitgründen keine Funktion!

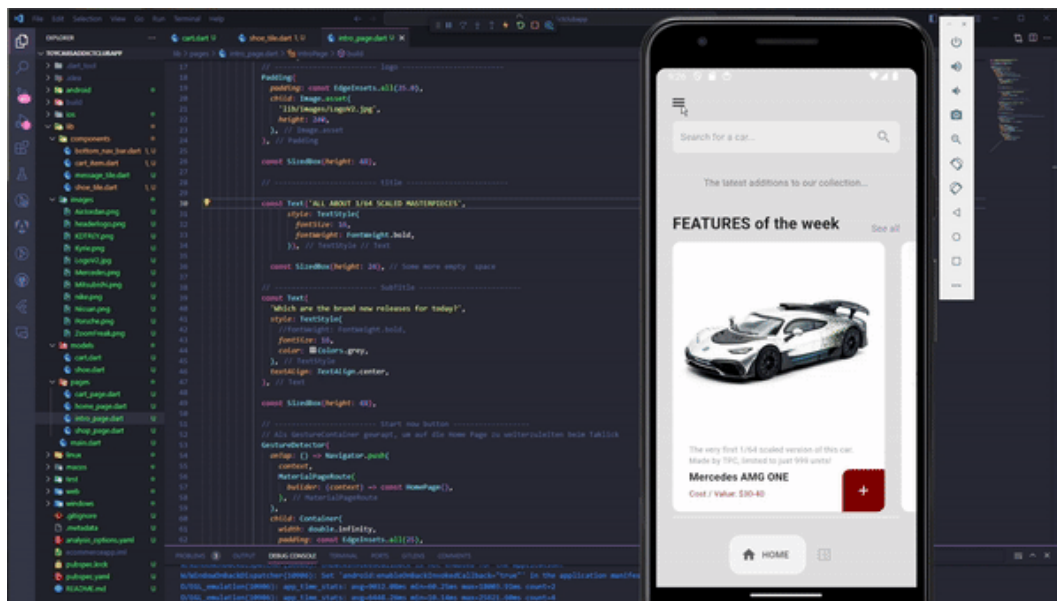


Abbildung 9: Das Aside Menü der ToycarsaddictApp (eigene Darstellung)

Ein wichtiger Erkenntnis-Tipp hierzu: Definiere unbedingt den Umfang der App bis ins kleinste Detail. Mir war dadurch klar, welche Elemente rein designtechnisch umgesetzt werden sollten und welche eine dahinterliegende Funktion haben.

Zum Beispiel war das Hinzufügen und Löschen von Fahrzeugen zu einer Liste eine komplexere Funktion, deren Verständnis für mich als Anfänger besonders herausfordernd darstellte. Im Gegensatz dazu gab es kein aktives Kontaktformular zum Versenden der Liste, sondern lediglich einen Platzhalter-Button ohne Funktion. Dies diente vor allem dazu, den zeitlichen Rahmen nicht zu sprengen.

Eine Veröffentlichung der App im Appstore und Google Play Store stand außer Frage – nicht zuletzt wegen der Gebühr von rund 200 CHF, die allein für den Zugang fällig gewesen wären. Kommt noch erschwerend dazu, dass jede App durch eine Vielzahl von Prüfstellen kommt, bis sie offiziell freigeschaltet werden darf. Dies lag zeitlich einfach schlichtweg nicht drin!

Entschieden habe ich mich daher, das Produkt zunächst auf der Projekt- und Code-Versionierungsplattform GitHub zu veröffentlichen, ergänzt durch eine passende

Readme Markdown-Dokumentation. Diese stellt die wesentlichen Funktionen der App mit Texten, Bildern und Videoausschnitten dar.

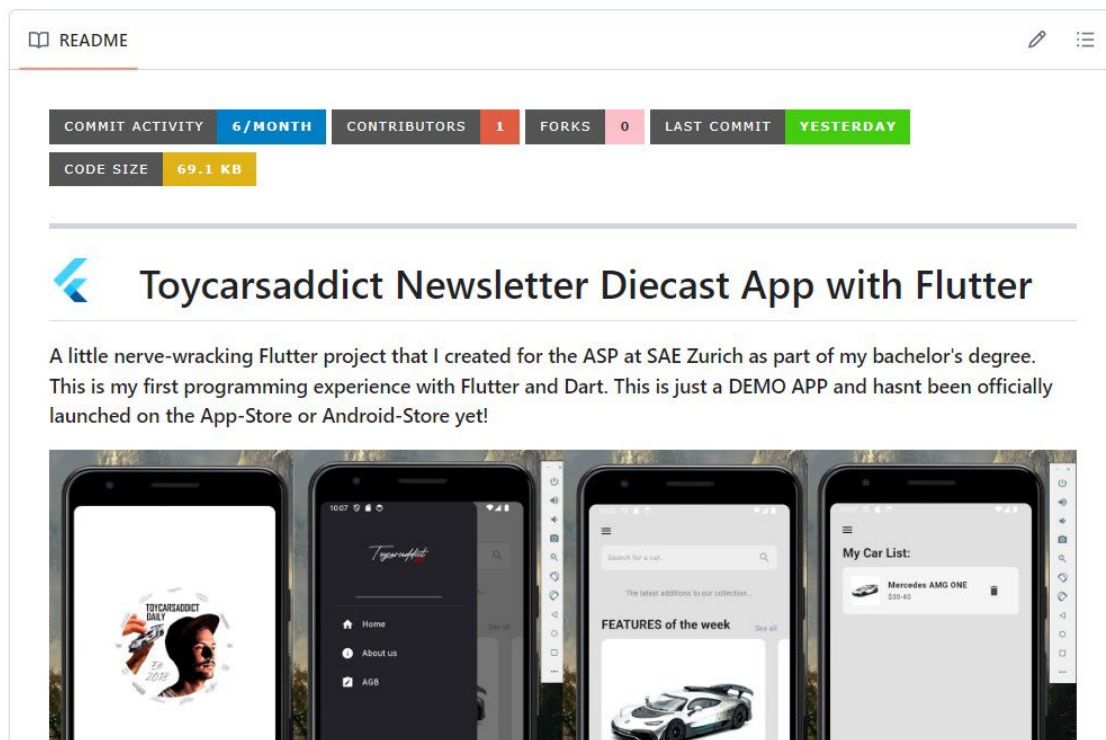


Abbildung 10: Readme Dokumentation auf Github (eigene Darstellung)

6. Eindrücke beim Programmieren

Flutter auf Anhieb zu verstehen ist beinahe unmöglich – der Aufbau ist für Einsteiger einfach zu komplex. Hier war das umfassende Handbuch eine große Stütze für mich, vor allem, weil ich mit der offiziellen Dokumentation auf der Flutter-Webseite nicht zurechtkam.

Kurz gesagt: Flutter ist ein SDK, ein Software Development Kit, also eine Art Baukasten. Es benötigt lediglich eine Codebasis durch die Programmiersprache Dart. Alles wird entsprechend kompiliert und für die Betriebssysteme iOS und Android optimiert. Zunächst muss Flutter installiert und als Umgebungsvariable festgelegt werden, damit mein Computer stets den korrekten Pfad kennt. Hierbei war wie des Öfteren meine Hauptquelle, das Flutter und Dart-Buch von Marc Marburger, das perfekte Nachschlagewerk:

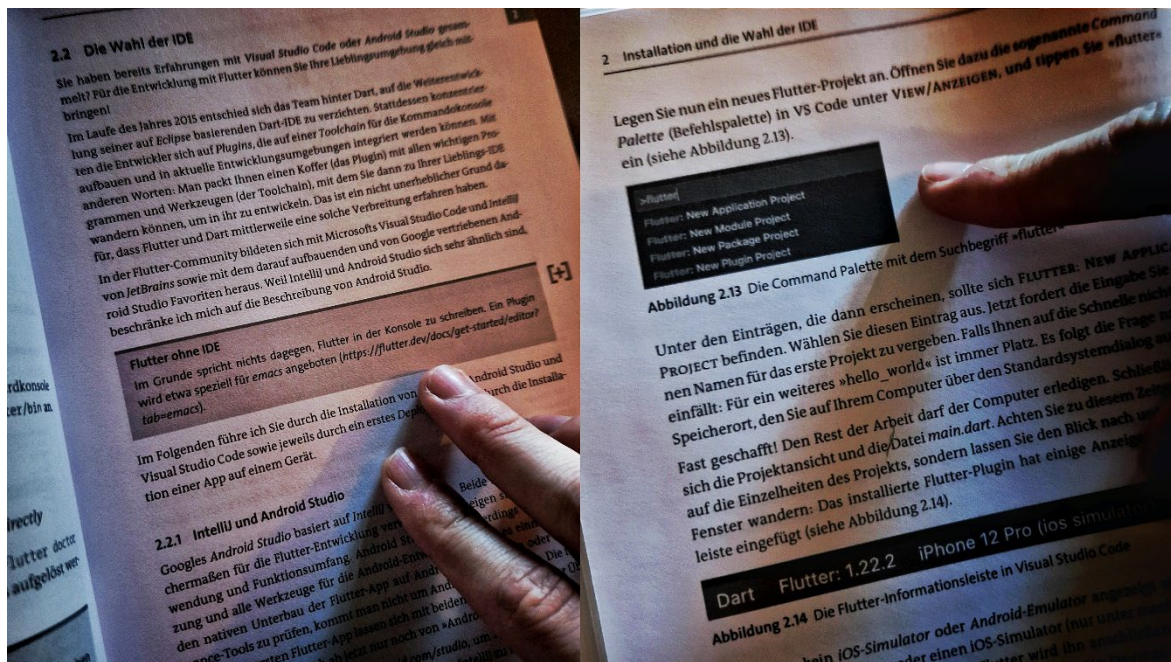


Abbildung 11: Kapitel 2 DIE Entwicklungsumgebung (eigene Darstellung)

Außerdem ist es notwendig, einen sogenannten "Flutter Doctor" auszuführen, der anzeigt, was bereits installiert ist und was der PC noch benötigt, um loslegen zu können. Als beinahe doktorwürdig erwies sich das korrekte Einrichten des AVD, des Android Virtual Device. Dies simuliert ein virtuelles Handy, auf dem man die Änderungen des Codes direkt sehen kann. Bis ich alles zum Laufen brachte, alle Fehler ausfindig gemacht und behoben hatte, vergingen etliche Stunden gar zahllose nervenaufreibende Tage. Mein Computer kam nämlich, bevor ich das Feintuning herausgefunden hatte, teilweise überhaupt nicht mit Flutter zurecht. Plötzlich funktionierten Tastatur und Maus nicht mehr, aus den Lautsprechern kam ein lautes Piepen oder der halbe Bildschirm war blau gefärbt. Diese Momente habe ich auf Kamera festgehalten und im Anhang des Ordners hinzugefügt.

Ein wichtiger Erkenntnis-Tipp dazu: Plant genügend Zeit für die Fehlerbehebung ein, ich persönlich hatte dies ziemlich unterschätzt! Vielleicht hatte ich einfach Pech mit meinem System, aber so viele Systemfehler, Abstürze und PC-Neustarts wie bei diesem Projekt hatte ich selten erlebt!

7. Der harte und steinige Weg zum Ziel

Allmählich verstand ich die Bedeutung von Widgets, die das Fundament von Flutter bilden. Widgets sind die zentralen Bausteine im Flutter-Framework. Ein Widget ist eine statische Beschreibung eines Teils einer Benutzeroberfläche. Jedes Element (Navigation, Body usw.) ist ein Widget und ist wie ein Baum strukturiert. Wenn wir Apps mit Flutter entwickeln, arbeiten wir also im Grunde mit einer Hierarchie von Widgets. Ein weiterer wichtiger Aspekt, ohne zu sehr ins Detail zu gehen, sind die Stateful- und Stateless-Widgets. Wenn wir etwas in die App integrieren möchten, das sich nie verändern soll (beispielsweise statischer Text), dann nutzen wir ein Stateless Widget. Soll sich der Zustand jedoch ändern, etwa bei einem Klick, kommen Stateful Widgets zum Einsatz. Dies wird übrigens auch im umfassenden Handbuch zu Flutter und Dart hervorragend erläutert, auch in den anderen Büchern ist stets von Widgets die Rede.

Schritt für Schritt baute ich die einzelnen Seiten auf, gestaltete sie ansprechender und stattete sie mit Funktionen aus. Es ging voran, aber es gab immer wieder Rückschläge. Tage und Nächte vergingen, bis ich die Ursachen ausfindig machen konnte. Selten lagen bei einem Projekt Frust und Freude so eng beieinander! Auf Momente totaler Ekstase folgten kleine Nervenzusammenbrüche!

Einen weiteren wichtigen Erkenntnis-Tipp dazu: Legt regelmäßige Pausen ein. Maximal eine Stunde konzentriertes Arbeiten am Computer, gefolgt von einer 15-minütigen Pause. Wie so oft in der Entwicklung kann es vorkommen, dass man stundenlang einen Fehler nicht findet, dann eine längere Pause macht und plötzlich innerhalb von fünf Minuten die Lösung vor Augen hat.

8. Schlussfazit

Zum Schluss muss ich feststellen: Das gesamte Unterfangen war äußerst ambitioniert und nicht frei von Risiken! Personen mit schwachen Nerven rate ich von einem solchen Projekt, eine App mit Flutter zu entwickeln, eher ab. Aber wenn man das Ergebnis meiner App betrachtet, vor allem wie beeindruckend das Endprodukt entstanden in so kurzer Zeit aussieht, dann war es zweifelsohne die Mühe wert!

Durch dieses Projekt ist mein Respekt für all jene, die sich tagtäglich in der mobilen Webentwicklung engagieren und ihr Bestes geben, noch weiter gewachsen. (Besonders angesichts der Kritik, die sie oft von Nutzern in den Stores erhalten). Dieses herausfordernde Projekt ermöglichte es mir, meine Fähigkeiten unter Beweis zu stellen, die technischen Zusammenhänge und Komponenten der mobilen App-Entwicklung zu verstehen und möglichst erfolgreich umzusetzen.

Als finalen Tipp möchte ich betonen: Fortschritte, vergleichbar mit dem Erlernen einer neuen linguale Sprache (wie Spanisch, Französisch etc...), werden erst durch kontinuierliches Üben über einen längeren Zeitraum hinweg erkennbar – ein Prozess, der Lernen, Verstehen und praktische Anwendung vereint. Selbiges durfte ich beim Erlernen von Flutter erleben und kann mit breiter Brust sagen: «Ich habe endlich eine App programmiert...oder zumindest versucht!»

9. Quellenverzeichnis

Marburger M., 2021. Flutter und Dart, Das umfassende Handbuch. 651 Seiten.
Rheinwerk Computing

Kevin Chromik (2022) Apps programmieren mit Flutter & Co – Was ist besser?.
[Online]. Verfügbar unter: <https://youtu.be/MBA6Q5EBjwI?si=JpiVOg-guUqRhoXLM> (Zugriff: 05. Februar 2024)

Fireship (2022) React Native vs Flutter – I built the same chat app with both [Online]. Verfügbar unter: https://youtu.be/X8ipUgXH6jw?si=Epicvz_m1FwOYHcJ (Zugriff: 05. Februar 2024)

Jago M., Zaiser V., 2023. App-Entwicklung für Dummies. 1. Auflage. Wiley-VCH

Edge R., Miola A., 2022. Cross-Platform UIs with Flutter, Unlock the ability to create native multiplatform UIs using a single code base with Flutter 3. 260 Seiten.
Packt Publishing

Flutter Documentation (2024) Full Documentation of Flutter. [Online]. Verfügbar unter: <https://docs.flutter.dev/> (Zugriff: 05. Februar 2024)