Hochschule Konstanz
Department of Computer Science

Submitted by:
Lars Bürger
Marcel Biselli
Simon Rauch
info@wallat.com

Mobile Anwendungen - SS 23

Projektdokumentation

Konstanz, 2023-07-12

Contents

List of Figures

1 Einleitung

Wir, das Entwicklerteam bestehend aus Marcel Biselli, Lars Bürger und Simon Rauch, haben im Rahmen des AIN Kurses "Mobile Anwendungen" im Sommersemester 2023 eine mobile App namens "Wall@" entwickelt. Unser Ziel war es, eine innovative Lösung für das Scannen und Verwalten von Dokumenten zu schaffen.

Mit Wall@ bieten wir eine benutzerfreundliche und effiziente Möglichkeit, Dokumente digital zu erfassen und in einem sicheren digitalen Geldbeutel zu verwalten. Die App richtet sich an alle, die eine bequeme Methode suchen, um Dokumente wie Quittungen, Rechnungen, Ausweise oder andere wichtige Unterlagen zu scannen, zu speichern und jederzeit griffbereit zu haben.

Bei der Entwicklung von Wall@ haben wir bewährte Software-Engineering-Praktiken und -Standards berücksichtigt, um eine stabile, skalierbare und gut strukturierte App zu gewährleisten. Durch den Einsatz der Programmiersprache Dart und des Flutter-Frameworks ist es uns gelungen, eine plattformübergreifende Lösung zu entwickeln, die gleichermaßen auf den Betriebssystemen iOS und Android funktioniert.

Diese Dokumentation dient als verbindliche Ressource für Entwickler, die an der App weiterarbeiten möchten, und bietet Einblicke in die Implementierungsdetails sowie die zugrunde liegenden Technologien.

Bei weiteren Fragen, Feedback oder Anregungen stehen wir Ihnen gerne zur Verfügung. Wir hoffen, dass diese Dokumentation einen wertvollen Beitrag zur effizienten Nutzung und Weiterentwicklung von Wall@ leistet.

Figure 1.1: Wall@ Logo

2 Entwicklungsprozess

Dieses Kapitel bietet einen Überblick über den Entwicklungsprozess von Wall@. Es behandelt die geplanten User Stories und den Einsatz von GitLab für das Projektmanagement und die Zusammenarbeit im Team. Die User Stories dienen als Leitfaden für die Entwicklung und beschreiben die Funktionalität von Wall@ aus Benutzersicht. Der Entwicklungsprozess umfasst verschiedene Phasen und Praktiken, die sicherstellen, dass die App erfolgreich entwickelt, getestet und bereitgestellt werden kann. Zusätzlich wird der Einsatz von GitLab vorgestellt, einer kollaborativen Entwicklungsplattform, die Funktionen zur Versionskontrolle und zum Projektmanagement bietet.

2.1 Issues in GitLab

Für die Koordination und Verwaltung des Entwicklungsprozesses haben wir das Ticket-System in GitLab genutzt. Mithilfe dieses Systems konnten wir Tickets erstellen, Aufgaben zuweisen und die Entwicklung effektiv koordinieren. Wir haben Funktionen wie Ticketzuweisung (an Verantwortliche), Labeling und Kommentarfunktionen genutzt, um die Zusammenarbeit und den Fortschritt innerhalb des Teams zu erleichtern.

2.2 User Stories

Aufgrund der Anforderungen an die App haben wir verschiedene User Stories erstellt, die die Funktionalität von Wall@ aus Benutzersicht beschreiben. Eine Liste der User Stories spowie der aktuelle Entwicklungsstand der einzelnen Stories ist über ein seperates GitLab Board (siehe User Stories Board) einzusehen.

2.3 GitFlow

Für eine effiziente Verwaltung des Entwicklungsprozesses haben wir das GitFlow-Modell implementiert. Es ermöglichte uns, den Code in verschiedenen Branches zu organisieren und stabile sowie entwicklungsorientierte Umgebungen aufrechtzuerhalten. Wir haben den "master"-Branch für stabile Versionen, den "develop"-Branch für die Hauptentwicklung und Feature-Branches für neue Funktionen verwendet. Zusätzlich nutzten wir Hotfix-Branches, um kritische Fehler schnell zu beheben und in den "master" und "develop"-Branch zu überführen. Zudem wurden alle Commits mit einer entsprechenden Issue-Nummer versehen, um die Nachvollziehbarkeit zu gewährleisten.

3 Architektur

3.1 MVC+S

3.2 Anwendung in allen View-Komponenten

3.2.1 Dependency-Inversion

3.2.2 Persistence als Service; DAO pattern

3.3 Compositum Pattern für FolderItems

3.4 State Management (Riverpod)

3.5 Multi-Language Support

3.6 ErrorHandling

3.6.1 Custom FutureBuilders + null safety

4 Persistenz

4.1 Verweis auf MVC+S (Persistenz als Service)

4.2 Isar Db

4.3 Shared Preferences für Settings

4.4 Bilder lose im Application Documents directory

5 Schnittstellen

5.1 Kamera via Package

6 Generelles (Flutter)

6.1 Cupertino Widgets (Slivers)

6.2 ((Adaptive Design (mithilfe von MediaQuery und Slivers)))?

6.3 Widgets of the Week

6.4 Animation

6.4.1 Hero Animations von Bild-Containern

6.4.2 Animation des Kamera-Buttons

6.5 Themes

6.6 Spezifische Packages

6.7 Splash Screen

Neben dem Logo (siehe 1.1) wurde auch ein Splash Screen für die App erstellt welcher beim Start der App angezeigt wird.