Prosjektøving IDATT2506-Applikasjonsutvikling for mobile enheter Kryssplattformløsninger

I dette prosjektet i faget IDATT2506-Applikasjonsutvikling for mobile enheter er det satt ett krav om å sammenligne og ta i bruk en kryssplattformløsning for mobilutvikling (Android). Denne dokumentet sammenligner kryssplattformløsningene ved å se på fordeler og ulemper ved dem.

Ionic

Ionic er en åpen-kildekode verktøysett som brukes for til å bygge kryssplattform applikasjoner. Det tillater brukeren å utvikle blant annet progressive web applikasjoner, Android og IOS applikasjoner. Ionic fokuserer på bruker bruker interaksjon, UX og generelt front-end. Dette gjør at applikasjonen har ett stort bibliotek og rammeverk som er mulig å integrere i applikasjonen. Rammeverk som Angular og Vue.js er blant mange rammeverk IONIC støtter. Dette gjør ionic veldig enkelt å sette seg inn i med tidligere erfaring i en av de støttede rammeverkene. Ionic bruker Javascript som programmeringsspråk.
Fordeler med kryssplattformløsningen er da blant annet at det er enkelt å lære. Andre fordel er vel strukturert dokumentasjon og fokus på UI.
Ulemper med ionic er at applikasjoner ikke vil ha like god sikkerhet som native applikasjoner og brukeren er forventet til å ha en dyp kunnskap av bibliotek og rammeverk som blir brukt.

Flutter

Flutter er ett rammeverk UI (toolkit) og SDK som kan bli brukt til å utvikle kryssplattformløsninger til blant annet nettsider, Android og IOS. Flutter bruker ett programmerings språk som heter Dart og har syntaks som er tilnærmet C# og JAVA. Flutter er en «widget» basert teknologi som betyr at brukeren kan bruke object-orientert programmering for ett hvert element. Det er utviklet av Google og bedrifter som e-bay, alibaba og Groupon bruker flutter.
Fordelene med å bruke flutter er mange. For å utvikle til forskjellige plattformer trenger brukeren å håndtere kun en kodebase. ``Widgets`` gir mulighet for å utvikle flotte visualer uten

å måtte fokusere på forskjellige enheter som applikasjonen skal brukes på. Flutter bruker Skia grafikk bibliotek som er en åpen-kilde bibliotek som tegner UI-en hver gang det endrer bildet, noe som gir applikasjonen en god flyt. Flutter har hot reload funksjon som er veldig nyttig når man utvikler applikasjoner. Endringer gjort i koden behøver ikke omstart for at applikasjonen endrer seg.
Flutter har også en del ulemper. Store filer forårsaket av widgets, noe som gjør at applikasjonen gjør at er tung å starte. Oppdateringer i programmet krever reinstallering av programmet. Flutter har i tillegg ikke det største biblioteket.

React Native

React Native er utviklet av Meta og bygger på React på Nodejs rammeverkene. React Native bruker også kun en kodebase for å utvikle apper til både IOS og Android. I likhet med Flutter bruker React native «fast refresh» som er en form for hot reload. Fast refresh oppdaterer fortløpende komponenter og filer som er avhengige av endringer gjort i koden uten å starte programmet på nytt.

React Native applikasjoner bygges opp av komponenter slik som widgets i Flutter. Innebygd i React Native har slik som Flutter mange mindre komponenter. React native bygger på Javascript, HTML og CSS. Dette gjør at React native er enkelt å mestre dersom man har forhåndskunnskaper i disse programmeringsspråkene. React native har en stor utvikler miljø og er veldig utbredt. Dette betyr at det finnes mange biblioteker som kan implementeres i applikasjonen.

Problemer som kan oppstå ved programmering i React Native er for eksempel at Javascript noe som er ett skript språk. For større applikasjoner kan dette påvirke prestasjonen av applikasjonen. Hvis komponenten brukeren er ute etter ikke eksisterer er de tvunget til å lage komponenten i for eksempel Android og IOS for seg selv.

Cordova

Cordova er også en kryssplattformløsning, men som ikke har fokus på UI, dette gjør at vi ikke kan utvikle kryssplattform UI i Cordova og må bruke for eksempel Ionic for UI-en. Cordova tar Javascript/HTML/CSS kode, noe som gjør det enkelt å komme i gang med, med forhåndskunnskaper i disse språkene. Cordova bruker mye programvareutvidelser. Cordova

har slike utvidelser for ulike funksjonaliteter til de ulike plattformene og dette gjør det mulig å utvikle gode applikasjoner. Med Cordova kan samme kode kjøres på flere mobile operativsystemer. Cordova har som sagt ikke fokus på UI og har ikke like stort bibliotek når det kommer til komponenter og bruk av disse. Cordova har også åpen-kildekode og er gratis å bruke.

På grunn av at brukeren blir avhengig av store programvareutvidelser og andre rammeverk UI, er Cordova ett vanskeligere kryssplattformløsning å sette seg inn i. Cordova mangler også ¨hot reload¨ fra Flutter eller ¨Fast refresh¨ fra React native. Uten denne funksjonen vil ta mer tid å utvikle en applikasjon, noe som gjør det kostelig spesielt for store bedrifter.

Kirigami

Kirigami er et UI rammeverk utviklet av KDE som brukes til å utvikle kryssplattformløsninger. Kirigami er slik som Flutter og React native en komponent basert rammeverk som gir utviklere mulighet til å utvikle raske applikasjoner med gode animasjoner. Kirigami har også en åpen-kidekode og kildekoden er godt opprettholdt. Kirigami applikasjoner er vanligvis skrevet i C++ for å yte til beste evne. Applikasjoner laget med Kirigami gir brukeren også mulighet til å overvåke applikasjons ytelse gjennom applikasjonsloggen laget av selveste Linus Torvalds i sammarbeid med Kirigami. Koko er et fotogalleri som også er tilgjengelig.

Valg av rammeverk

Etter å ha lest meg opp på disse kryssplattformutviklings rammeverkene har jeg gjort meg opp noen meninger. Cordova har mye og mange programvareutvidelser til å utnytte innebygde funksjonaliteter. Men ¨hot reload¨ er noe som er tidssparende og veldig nyttig når man utvikler applikasjoner. Uten ¨hot reload¨ kan man ikke se endringene man har gjort uten å omstarte hele applikasjonen. Det er heller ingen innebygd UI komponenter.

Ionic derimot har innebygd UI komponenter og en form for ¨hot reload¨. Ionic er ikke like ¨full¨ pakke som Flutter eller React native. I form av mindre komponenter og programmeringsspråk som vi ikke har lært eller er veldig forskjellig fra det vi har lært i IDATT2506, i tillegg er ikke Ionic like mye brukt som Flutter og React native.

Kirigami virker som ett flott rammeverk, men det er dessverre ikke nok informasjon og ressurser om det på nettet utenfor KDE sin egen dokumentasjon. Hadde vi tatt det i bruk fra starten av semesteret hadde vi hatt god tid til å vende oss til å utvikle applikasjoner i C++ noe som jeg ikke har gjort før.

Flutter og React Native er like. Begge har ¨hot reload¨/¨fast refresh¨, UI komponenter, begge rammeverkene er mye brukt blant applikasjonsutviklere og blir godt vedlikeholdt og mye ressurser for hjelp. I tillegg er det enkelt å starte med en av disse rammeverkene. Forskjellene blir da hovedsakelig programmeringsspråk. I Flutter bruker man Dart, og i React Native brukes HTML/CSS/Javascript. Både Flutter og React native passer godt til dette prosjektet. React native bygger på React og web-utvikling, som er derfor jeg ikke føler at det passer best til applikasjonsutvikling. Flutter derimot er utviklet av Google som eier Android og dermed er integrasjon og utvikling av applikasjoner under samme paraply, noe som betyr at utviklingen skjer sømløst. Jeg velger derfor å gjennomføre prosjektet i Flutter.

Kilder:

1. What is Ionic Framework?: https://www.javatpoint.com/what-is-ionic-framework
2. Why You Should Use Flutter for Your Projects: https://www.freecodecamp.org/news/why-you-should-use-flutter/#:~:text=Flutter%20code%20can%20run%20on,So,%20Flutter%20is%20cheap.
3. React native: https://reactnative.dev
4. Flutter VS Apache Cordova: Choosing the Best Technology for Cross-Platform Development: https://surf.dev/flutter-vs-apache-cordova/
5. Getting started with Kirigami: https://develop.kde.org/docs/use/kirigami/
6. The Six Best Cross-Platform App Development Frameworks: https://kotlinlang.org/docs/cross-platform-frameworks.html