# Pametni ekran za prikaz podataka koristeći e-ink tehnologiju

**Antonić, Andrej**

**SVEUČILIŠTE U ZAGREBU**
**FAKULTET ORGANIZACIJE I INFORMATIKE**
**VARAŽDIN**

**Andrej Antonić**

# PAMETNI EKRAN ZA PRIKAZ PODATAKA KORISTEĆI E-INK TEHNOLOGIJU

**ZAVRŠNI RAD**

**Varaždin, 2023.**

**SVEUČILIŠTE U ZAGREBU**

**FAKULTET ORGANIZACIJE I INFORMATIKE**

**V A R A Ž D I N**

**Andrej Antonić**

**Matični broj: 0016149694**

**Studij: Informacijski i poslovni sustavi**

# PAMETNI EKRAN ZA PRIKAZ PODATAKA KORISTEĆI E-INK TEHNOLOGIJU

**ZAVRŠNI RAD**

**Mentor :**

Doc. dr. sc. Boris Tomaš

**Varaždin, rujan 2023.**

*Andrej Antonić*

**Izjava o izvornosti**

Izjavljujem da je moj završni rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

*Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi*

---

**Sažetak**

Ovaj rad istražuje upotrebu NFC modula spojenog s E-ink ekranom kako bi se omogućilo prikazivanje podatataka bez potrebe za vanjskim napajanjem. Uz to, razvijena je mobilna aplikacija u jeziku Kotlin koja putem FOI API-ja prikuplja podatke iz kadrovske evidencije i generira sliku za odabrane osobe, a zatim ju šalje putem NFC-a do povezanog E-ink modula. U radu je analizirana tehnologija E-ink ekrana, NFC-a i jezika Kotlin. Glavna teza rada je usredotočena na praktičnu primjenu ove tehnologije za prikaz kadrovske evidencije. Predstavlja se poboljšana i ekološki prihvatljiva alternativa klasičnim ekranima za prikazivanje statičnih podataka. Mobilna aplikacija u Kotlinu pruža jednostavan način prikupljanja i generiranja podataka, dok NFC omogućuje brz i siguran prijenos do E-ink ekrana. U zaključku, ovaj rad ukazuje na mogućnosti i prednosti pametnih E-ink ekrana za prikazivanje podataka. Ističe se praktičnost bežičnog prijenosa putem NFC-a i smanjenje potrebe za vanjskim napajanjem.

**Ključne riječi**: E-ink; Kotlin; NFC; E-paper; API; Kadrovska evidencija; Mobilna aplikacija;

# Sadržaj

# 1. Uvod

U današnje doba, obilježeno stalnim tehnološkim napretkom, potražnja za učinkovitim i inovativnim metodama prezentiranja podataka je u porastu. U tom smislu, pojava E-ink tehnologije označava revolucionarni napredak prema ekološki prihvatljivijem i funkcionalnijem pristupu prikaza informacija. sposobnost ovih zaslona da prikazuju realistične vizualne prikaze bez potrebe za stalnim oslanjanjem na vanjski izvor napajanja otkriva neviđene mogućnosti u brojnim sektorima i upotrebama.

Fokus ovog rada je na istraživanju potencijalne upotrebe E-ink tehnologije u stvaranju pametnog zaslona koji može učinkovito prikazati podatke. Ovaj pametni zaslon koristi NFC modul za olakšavanje bežičnog prijenosa informacija. Glavni cilj ovog istraživanja je osmisliti rješenje koje omogućuje prikaz podataka na ekološki prihvatljiv način, a istovremeno jamči praktičnost i prenosivost.

U okviru ovog rada bit će provedena sveobuhvatna analiza E-ink tehnologije. Uz ovo ispitivanje također će se provesti istraživanje brojnih prednosti ali i ograničenja ove tehnologije. Također će biti provedena analiza NFC prijenosa podataka, usporedbe sa drugim prijenosima podataka te integracija tih dviju tehnologija. Kako bi se te tehnologije mogle koristiti izražena je mobilna aplikacija pomoću jezika Kotlin, koja služi kao most koji povezuje korisnika i ekran. Ova aplikacija olakšava generiranje i prikupljanje podataka iz određenih izvora bez napora, čime se poboljšava korisničko iskustvo.

Primarni cilj ovog rada je dati prioritet praktičnoj implementaciji pametnog zaslona koja može učinkovito prikazati evidenciju osoblja. Ovaj sustav ne samo da omogućuje jednostavan pristup ključnim podacima o zaposlenicima, već također igra značajnu ulogu u smanjivanju oslanjanja na tradicionalne vanjske izvore napajanja. Posljedično, ovaj sustav će dati značajan doprinos održivijem okolišu.

# 2. Metode i tehnike rada

U procesu razrade teme, primijenjene su različite metode i tehnike kako bi se postiglo dublje razumijevanje i uspješna implementacija ciljeva rada. Prvo je provedeno istraživanje i analiza literature kako bi se steklo osnovno razumijevanje e-paper ekrana i NFC tehnologije. Korišteni su online resursi, blogovi, dokumentacija i ChatGPT kako bi se proučili tehnički detalji i primjeri primjene. Analizirane su već postojeće aplikacije i SDK-ovi kako bi se bolje razumjela praktična primjena tehnologija.

Za praktičnu implementaciju, koristio se programski jezik Kotlin u razvoju aplikacije. Android Studio je poslužio kao ključno razvojno okruženje za izradu i testiranje Android aplikacije. Git verzioniranje omogućilo je praćenje promjena i upravljanje izvorima.

# 3. E-paper ekran

E-paper ekran posebna je vrsta zaslona koja koristi e-ink tehnologiju za prikaz informacija. Ova tehnologija replicira izgled tradicionalnog tiskanog papira, poboljšavajući njegovu čitljivost i prilagodljivost različitim okolnostima osvjetljenja, uključujući i jaku sunčevu svjetlost.

E-paper ekran izdvaja se od tradicionalnih zaslona, poput LCD-a ili OLED-a, zbog svoje značajke da ne treba stalno osvjetljenje. To se postiže provedbom procesa zvanog elektroforeza. U tom procesu sićušne čestice tinte ili boje manipuliraju se električnim nabojem kako bi se stvorila slika. Ljepota ove tehnologije je u njenoj bistabilnosti, koja omogućuje da ekran zadrži prikazanu sliku bez stalnog zahtjeva za osvježavanjem.

E-paper ekrani našli su svoju primjenu u širokom rasponu uređaja uključujući e-knjige, pametne satove, pametne etikete, pametne kućanske uređaje i druge IoT (Internet of Things) uređaje. Osim potrošačke elektronike, e-paper nalazi primjene i u industriji poput medicine, prometa, transporta i svim drugim sektorima u kojima postoji potreba za prikazivanjem informacija na energetski učinkovit način.

## 3.1. Principi rada

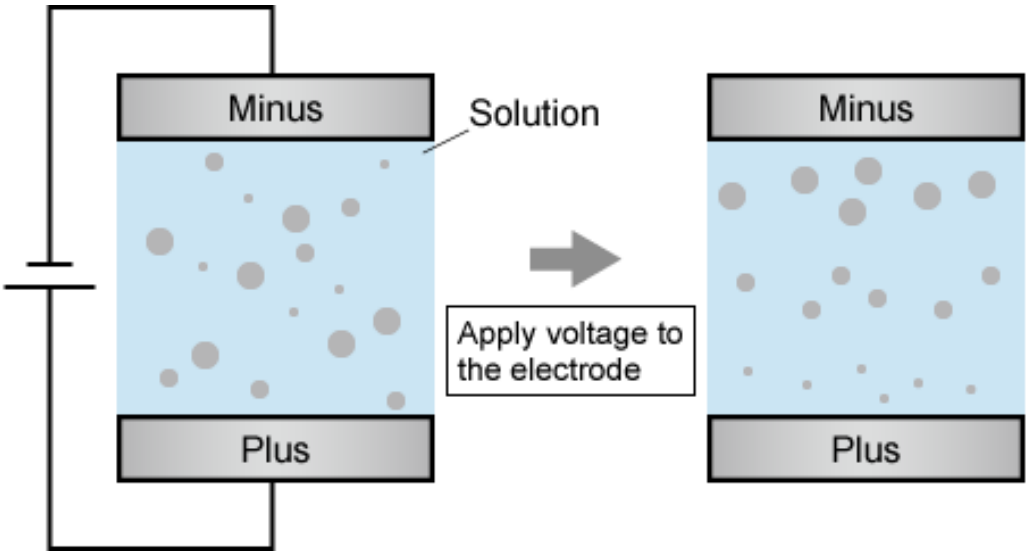
Kao što je već rečeno, e-paper ekran funkcionira na principu elektroforeze i bistabilnosti. U nastavku će naglasak biti stavljen na objašnjavanje djelovanja elektroforeze i njezinu središnju ulogu u stvaranju vizualno jasne i lako čitljive slike. Dodatno, rasjaznit će se pojam bistabilnosti, karakteristiku koja omogućuje prikazivanje izloženog sadržaja bez potrebe za vanjskim izvorom energije.

### 3.1.1. Elektroforeza

Elektroforeza je fenomen koji uključuje kretanje nabijenih čestica u tekućini, gelu ili suspenziji kada se primijeni električno polje. "Smjer kretanja čestice ovisi o polaritetu njezina električnog naboja, brzina kretanja povećava se s brojem naboja i s jakošću električnog polja, a opada s veličinom čestice." [1] Elektroforeza je svestrana tehnika koja se široko koristi u različitim disciplinama uključujući znanstvena istraživanja, industrijske procese i napredak u medicini.

- Molekularna biologija: služi za odvajanje i analizu bioloških makromolekula poput DNA, RNA i proteina.
- Klinička diagnostika: elektroforeza proteina nudi vrijednu metodu za ispitivanje proteina u različitim biološkim uzorcima, uključujuči krv.
- Forenzika: analiza DNK uzoraka i identifikacija osoba uključenih u kriminalističke istrage.
- Proizvodnja: koristi se za nanošenje premaza na različite materijale poput metala, tkanika i drugih stvari.

- Istraživanje okoliša: omogućuje analizu uzoraka tla, vode i zraka u svrhu identifikacije i kvantifikacije raznolikog raspona kemijskih spojeva.
- Farmaceutska industrija: kontrola kvalitete i analiza sastava lijekova.




Slika 1: Djelovanje elektroforeze (Izvor: Matsusada Precision, 2023.)

Kada je riječ o e-paper ekranima, elektroforeza se koristi unutar slojeva e-ink tehnologije koja se sastoji od mikrokapsula napunjenih tekućinom. Ove mikrokapsule sadrže nabijene čestice koje mogu biti crne ili obojene, a ti se slojevi nalaze između dvije elektrode. Jedna je pozitivna, a druga negativna. Svrha ovih elektroda je stvaranje elektronickog polja.

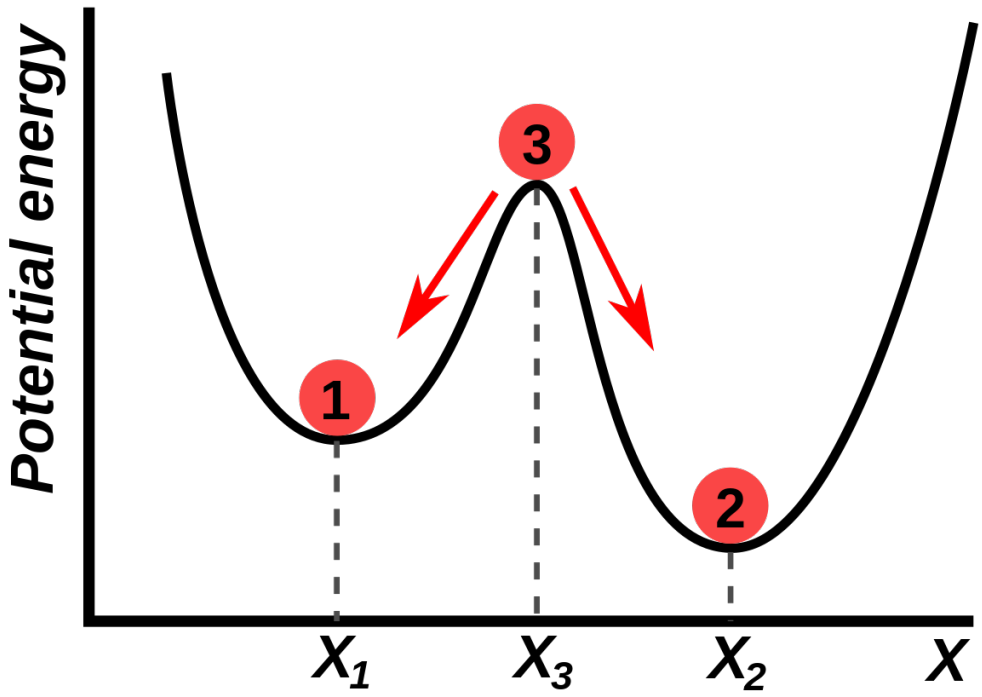
1. Neutralno stanje: prilikom nedostatka električnog polja, nabijene čestice unutar mikrokapsula ravnomjerno su raspoređene, što rezultira neutralnom slikom koja se prikazuje na zaslonu e-papira. To implicira da je cijela površina zaslona u stanju ravnoteže, odnosno bez prikazane vidljive slike.
2. Primjena električnog polja: prilikom djelovanja električnog naboja, stvara se električno polje koje utječe na nabijene čestice prisutne unutar e-ink slojeva. Na temelju polariteta napona koji se primjenjuje, elektrodoreza dovodi do dva temeljna scenarija.
    - Pozitivan napon: mikrokapsule sadrže pozitivno nabijene čestice koje se povlače prema negativno nabijenom sloju elektrode. Istovremeno, negativno nabijene čestice unutar mikrokapsula migriraju dalje od površine i okupljaju se oko pozitivno nabijenog sloja elektrode. Kao rezultat ovog kretanja, crne ili obojene čestice se guraju na površinu, postaju vidljive i tvore tamna područja ne ekranu.
    - Negativan napon: čestice s pozitivnim nabojem prisutne unutar mikrokapsula kreću se u smjeru od površine prema negativno nabijenom sloju elektrode. Istovremeno se čestice s negativnim nabojem povlače prema površini i kreću prema pozitivno nabijenom sloju elektrode. Kao rezultat toga, bijele čestice se guraju prema površini, postaju vidljive i tvore svijetla područja na ekranu.

3. Zadržavanje slike: nakon što se napon eliminira čestice ostaju na svome položaju i prijanjaju na površinu zaslona e-papira, što rezultira stvaranjem stabilne slike. Ova posebna karakteristika e-paper ekrana se naziva bistabilnost i omogućuje očuvanje slike bez potrebe za čestim ažuriranjem.

### 3.1.2. Bistabilnost

U području konvencionalnih zaslona poput LCD-a i OLED-a, slika se održava kontinuiranim osvježavanjem piksela kako bi se prikazala svježa slika. Ovaj proces neprestanog osvječavanja zahtijeva značajnu količinu energije, posebno na uređajima koji zahtijevaju čest prikaz informacija.

Za razliku od tradicionalnih zaslona, e-paper ekrani koriste prijašnje objašnjenu elektroforezu kao sredstvo za manipuliranje položajem sitnih čestica tinte ili boje unutar svojih slojeva. Nakon što napon više nije prisutan, te nabijene čestice ostaju fiksirane na svojim položajima, eliminirajući potrebu za stalnim osvježavanjem.




Slika 2: Graf potencijalne energije bistabilnog sustava (Izvor: Wikipedia, 2023.)

"U dinamičkom sustavu bistabilnost znači da sustav ima dva stabilna ravnotežna stanja. Bistabilna struktura može mirovati u jednom od dva stanja. Primjer mehaničkog uređaja koji je bistabilan je prekidač za svjetlo. Poluga prekidača dizajnirana je tako da stoji u položaju "uključeno" ili "isključeno", ali ne između ta dva položaja." [4]

Zadržavanjem slike bez potrebe za stalnim osvježavanjem, e-paper zasloni mogu smanjiti potrošnju energije. Ekran koristi energiju samo kada dođe do promjene prikazane slike,

dok se statične slike ili tekst mogu prikazati bez potrošnje energije. Ova izvanredna karakteristika energetske učinkovitosti e-paper zaslona omogućuje im nevjerojatno produljeno trajanje baterije. Takvo produženo trajanje baterije posebno je ključno za uređaje koji zahtijevaju prikaz informacija tijekom duljeg razdoblja. Ova karakteristika im također omogućuje zadržavanje prikaz i kada se baterija isprazni ili se uređaj ugasi.

## 3.2. Prednosti i nedostatci

E-paper zasloni posjeduju i pozitivne i negativne atribute, što ih čini prikladnima za specifične svrhe, a istovremeno ograničavaju njihovu upotrebu u drugim scenarijima.

Prednosti e-paper ekrana:

- Niska potrošnja energije: mogućnost održavanja slike bez potrebe za njezinim stalnim osvježavanjem.
- Sličnost tradicionalnom papiru: vrlo kompatibilni za čitanje tekstova, e-knjiga i raznih drugih materijala.
- Vidljivost na jakom svjetlu: koriste reflektirajuću tehnologiju što ih čini optimalnim izborom za vanjsku upotrebu.
- Nema treperenja: smanjuje umor očiju i osigurava ugodnije iskustvo čitanja čak i tijekom duljih razdoblja korištenja.
- Fleksibilnost i tankost: neprimjetno uključivanje u širok raspon uređaja i aplikacija.

Nedostatci e-paper ekrana:

- Sporije osvježavanje: nisu najprikladnija opcija za aplikacije koje zahtijevaju brze i česte izmjene slika, kao što su reprodukcija videa ili iskustvo igranja.
- Crno-bijeli prikaz: nedostatak za određene aplikacije koje zahtijevaju životopisne boje i detaljnu jasnoću slike. Iako postoje e-paper ekrani u boji trenutno podržavaju do maksimalno 7 boja i iznimno su skupi.
- Ograničen kut gledanja: rezultira iskrivljenim ili izgubljenim prikazom kada se gleda iz kutova koji nisu optimalni.
- Visoka cijena: izazov za njihovu široku primjenu u određenim industrijskim sektorima zbog svoje komparativne cijene.
- Tehnološka ograničenja: postoje mnoga ograničenja u pogledu brzine osvježenja, raznolikosti boja i rezolucije. Ova ograničenja mogu spriječiti njihovu primjenu u naprednijim scenarijima.

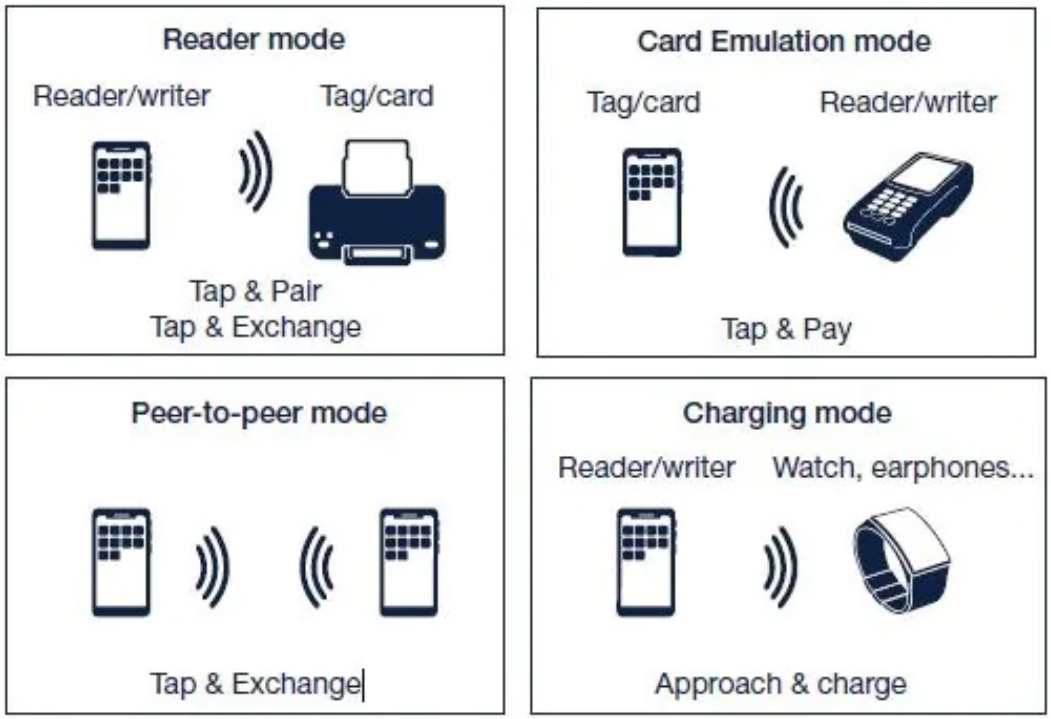
Općenito, e-paper ekrani su se pokazali vrlo korisnima u određenim scenarijima, osobito u onima koji zahtijevaju minimalnu potrošnju energije i iznimnu vidljivost čak i u dobro osvijetljenim okruženjima. Unatoč tome, postoje slučajevi u kojima alternativne tehnologije poput LCD-a i OLED-a mogu biti bolje opremljene za ispunjavanje specifičnih zahtjeva i zadovoljavanje različitih potreba.

# 4. NFC tehnologija

U modernoj eri tehnologije, gdje se napredak događa neviđenom brzinom, naši načini komunikacije, financijskih transakcija i korištenje uređaja doživjeli su duboku transformaciju. Među bezbrojnim otkrićima u bežičnim tehnologijama, Near Field Communication (NFC) ističe se kao najvažniji razvoj koji je potpuno revolucionirao naše svakodnevne rutine. NFC se postavio kao sinonim za brz, siguran i prikladan način razmjene podataka, provođenja transakcija i interakcije s okolinom.

## 4.1. Principi rada

NFC tehnologija oslanja se na niz temeljnih načela za olakšavanje bežične komunikacije na malim udaljenostima. Ovi principi čine osnovu NFC-a i omogućuju besprijekoran prijenos podataka.




Slika 3: NFC operacije (Izvor: STMicroelectronics, 2023.)

- Indukcija magnetskog polja: koristi se za uspostavljanje komunikacije između uređaja. Aktivni NFC uređaj, kada je uključen, stvara magnetsko polje koje može prenijeti podatke obližnjem pasivnom uređaju.
- Radiofrekvencija: NFC koristi radiovalove visoke frekvencije kako bi olakšao prijenos podataka. Prevladavajuće frekvencije koje se koriste u NFC tehnologiji obično su 13,56 MHz, što omogućava brz i pouzdan prijenos podataka uz učinkovitu komunikaciju.

- Bliski domet komunikacije: NFC tehnologija je posebno dizajnirana za komunikaciju unutar ograničenog raspona od 4 centimetra. Ovaj zahtjev za neposrednom blizinom osigurava poboljšane sigurnosne mjere, pošto uređaji moraju biti u vrlo neposrednoj fizičkoj blizini ili se čak dodirivati kako bi uspostavili komunikaciju.
- Pasivni i aktivni način rada: aktivni način rada uključuje uređaje poput pametnih telefona koji stvaraju magnetsko polje i preuzimaju kontrolu nad cijelim procesom komunikacije. Pasivni način rada karakteriziraju uređaji poput NFC oznaka ili pametnih kartica koji reagiraju na magnetsko polje i prenose vlastite podatke.
- Brza uspostava veze: uspostava veze obično traje nekoliko milisekundi. Ova brza veza je korisna za brzu razmjenu podataka, provođenje transakcija i interakciju s drugim uređajima koji podržavaju NFC tehnologiju.
- Dvosmjerna komunikacija: omogućuje razmjenu podataka između aktivnog i pasivnog uređaja. Oba uređaja mogu primati i slati podatke. Ova mogućnost je ključna za brojne NFC aplikacije.

### 4.1.1. Usporedba s drugim bežičnim tehnologijama

U golemom i raznolikom području bežičnih tehnologija, NFC pojavljuje se kao vrlo privlačan i utilitaran oblik bežične komunikacije. Međutim, bitno je istražiti i procijeniti kako NFC stoji u usporedbi s drugim tehnologijama kao što su Bluetooth, RFID (Radio-Frequency Identification) i Wi-Fi.

#### 4.1.1.1. NFC i Bluetooth

Bluetooth ima puno širi domet, sposoban je povezati uređaje udaljene do 10 metara, posebno u verziji Bluetooth 5.0 i novijima. Dodatno, kada je riječ o brzini prijenosa podatak NFC zaostaje s maksimalnom brzinom prijenosa od 424 kbps, dok Bluetooth može postići puno veće brzine do preko 2 Mbps. Što se tiče primjene, NFC se obično koristi za brzo dijeljenje malih količina podataka, kao što su podaci za kontakt, URL-ovi ili male datoteke. Bluetooth je s druge strane prikladniji za prijenos većih datoteka poput glazbe, fotografija i videa. Također se obično koristi za povezivanje perifernih uređaja kao štš su slušalice i tipkovnice.

#### 4.1.1.2. NFC i RFID

NFC je vrsta RFID tehnologije koja posjeduje različite karakteristike po kojima se izdvaja. Jedna od ključnih razlika je u tome što NFC olakšava dvosmjernu komunikaciju između uređaja, omogućujući prijenos podataka i s aktivnog i s pasivnog uređaja. Naprotiv, RFID dopušta samo jednosmjernu komunikaciju gdje čitač čita podatke s pasivne RFID oznake bez mogućnosti njezine izmjene. Kada je riječ o sigurnosti, NFC se često smatra sigurnijim od mnogih RFID sustava zbog korištenja dodatnih sigurnosnih slojeva poput enkripcije i autentifikacije. Ova poboljšana sigurnost čini NFC posebno prikladnim za aplikacije koje zahtijevaju

povećanu zaštitu, kao što su mobilna plaćanja. Za razliku od NFC-a, RFID nudi različite varijante s različitim rasponima dometa, koji variraju od nekoliko centimetara do više metara, ovisno o frekvenciji i vrsti korištene RFID tehnologije.

#### 4.1.1.3. NFC i WiFi

Što se tiče primjene, NFC i Wi-Fi imaju različite svrhe. NFC se obično koristi za brzu i jednostavnu razmjenu podataka između uređaja. S druge strane, Wi-Fi se prvenstveno koristi za pružanje bežičnog pristupa internetu i povezivanje više uređaja na lokalnu mrežu. Zbog toga je Wi-Fi ključan za pristup internetu na prijenosnim računalima, pametnim telefonima, tabletima i drugim uređajima, kao i za omogućavanje komunikacije i dijeljenja datoteka između njih. Drugi čimbenik koji razlikuje ove dvije tehnologije je njihova brzina prijenosa podataka. Wi-Fi nudi znatno veće brzine u usporedbi s NFC-om. Ovisno o verziji Wi-Fi protokola koji se koristi, brzine prijenosa mogu se kretati od desetaka do stotina pa čak i gigabita u sekundi. To omogućuje bržu i učinkovitiju razmjenu podataka, što Wi-Fi čini preferiranim izborom za zadatke koji zahtijevaju brzi prijenos velikih količina podataka. Jedna od ključnih razlika između NFC-a i Wi-Fi-ja je njihov domet. Wi-Fi omogućuje bežični pristup Internetu na mnogo većim udaljenostima, koje često dosežu i nekoliko desetaka metara. To znači da Wi-Fi može omogućiti povezivanje s uređajima koji su udaljeniji, što ga čini praktičnijim za korisnike.

## 4.2. NFC uređaji i njihove primjene

NFC tehnologija metoda je bežičnog prijenosa podataka koja omogućuje uređajima da međusobno komuniciraju na malim udaljenostima. Stekao je široku popularnost i podržavaju ga brojni uređaji.

- Pametni telefoni i tableti: danas je uobičajeno pronaći NFC tehnologiju integriranu u većinu pametnih telefona i tableta. Osim razmjene podataka omogućuje korištenje pametnih telefona kao digitalnih novčanika.
- Pametne kartice i identifikatori: NFC kartice nalaze široku primjenu u brojnim područjima uključujući hotelske ključeve, korporativne identifikacijske kartice, putovnice i razne druge vrste identifikacijskih dokumenata.
- NFC naljepnice i oznake: koriste se u različitim područjima kao što su muzeji, turističke atrakcije i maloprodajni objekti, budući da učinkovito poboljšavaju korisnička iskustva dajući dodatne uvide u izložbe, umjetnička djela, proizvode ili čak značajne znamenitosti.
- Pametne kuće i IoT uređaji: prihvatili su korištenje NFC tehnologije kako bi revolucionirali način na koji se različiti uređaji kontroliraju i besprijekorno međusobno povezuju. Primjer toga može se vidjeti upravljanju pametnim kućama bez napora jednostavnim dodirivanjem pametnog telefona sa NFC oznakom postavljene na ulaznim vratima. Ova jednostavna radnja omogućuje vlasnicima domova da bez napora otključaju vrata, aktiviraju svjetla, pa čak i mijenjaju sobnu temperaturu prema svojim željama.

- Zdravstveni uređaji i medicinska primjena: praćenje zdravstvenih informacija i pružanje medicinske pomoći. Glavna ilustracija uključuje pacijente koji mogu koristiti svoje pametne telefone opremljene NFC-om za praćenje svoje težine, razine šećera u krvi i još puno stvari.
- Transport i kartice javnog prijevoza: omogućuje brzo i praktično plaćanje karata. Putnici mogu bez napora koristiti svoje pametne telefone ili NFC kartice za pristup autobusima, vlakovima i raznim načinima prijevoza.
- Marketing i interaktivni plakati: sve više koriste NFC tehnologiju kao sredstvo za povećanje angažmana potrošača. Ugradnjom NFC oznaka na različite medije poput postera, časopisa ili čak proizvoda, pojedinci mogu bez napora pristupiti dodatnim detaljima, kupovati ili sudjelovati u marketinškim nagradnim igrama.

# 5. E-ink zaslon i NFC modul

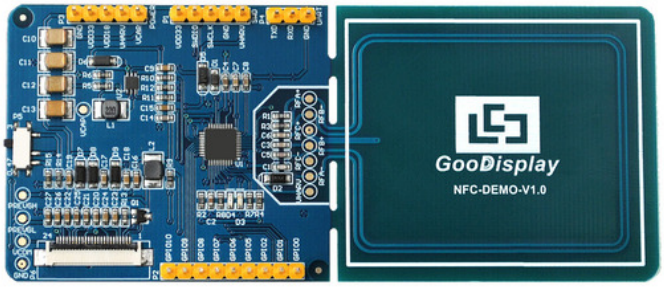
Kao što je već rečeno, jedna od izvanrednih značajki E-ink tehnologije je njezina sposobnost da sačuva slike čak i bez stalnog napajanja. Integracijom sa NFC modulom postaje moguće bežično prenijeti podatke, posebno slike, na E-ink zaslon bez potrebe za dodatnim vanjskim uređajem za napajanje.

Početni korak u procesu prijenosa slike uključuje aktiviranje NFC-a na mobilnom uređaju ili bilo kojem drugom uređaju za slanje podataka. Što se tiče E-ink zaslona, na njega je spojen modul koji se sastoji od antene za dohvaćanje signala koji se šalje sa mobilnog uređaja i matične koja sadrži različite komponente za prijenos slike do E-ink zaslona.

Kada korisnik odabere sliku ili podatke za prikaz na E-ink zaslonu, mobilna aplikacija koristi NFC tehnologiju za prijenos tih podataka. NFC modul na mobilnom uređaju generira elektromagnetsko polje koje zatim NFC modul na E-ink zaslonu prima kao signal. Prilikom primanja signala sa slikom NFC modul također dobiva i malo napajanje. Ovaj signal nosi sveobuhvatne detalje o slici koja se namjerava prikazati na zaslonu ali također i dovoljno energije za promjenu slike na zaslonu.

## 5.1. Odabran ekran i modul

Za potrebe ovog rada odabran je modul DENFC-M01 od tvrtke Good Display [6]. Najbitnija karakteristika ovog modula je da podržava prikupljanje energije generirane od NFC-a, a ne samo dohvaćanje podataka koji se prenose.




Slika 4: NFC modul (Izvor: Good Display, 2023.)

"U kombinaciji s mobilnim telefonima s NFC funkcijom, DENFC-M01 može prikupiti do 300 MW snage. Tako visoko stjecanje snage ne samo da može podržati rad DENFC-M01 punom brzinom u pasivnom načinu rada, već također može opskrbljivati energijom vanjske uređaje kao što su senzor slike i zaslon e-papira." [6] Iz priloženog opisa i demonstracija se može vidjeti zašto je odabrani modul savršen za ovaj rad.

Odabrani NFC modul može podržati E-ink zaslone do veličine od 4.2" koji mogu biti crno-bijeli ili u tri boje (crna, bijela i crvena/žuta). Imajući to na umu odabran je E-ink zaslon u tri boje (crna, bijela i crvena) veličine 4.2" od istog proizvođača.


Slika 5: E-ink zaslon (Izvor: Good Display, 2023.)

# 6. Mobilna aplikacija

U sljedećim odlomcima istražit će se proces razvoja mobilne aplikacije koja omogućuje prikaz FOI kadrovske evidencije na e-paper ekranu putem NFC tehnologije. Detaljno će se razmotriti tehnički koraci, alati i tehnike korištene za implementaciju ove funkcionalnosti.

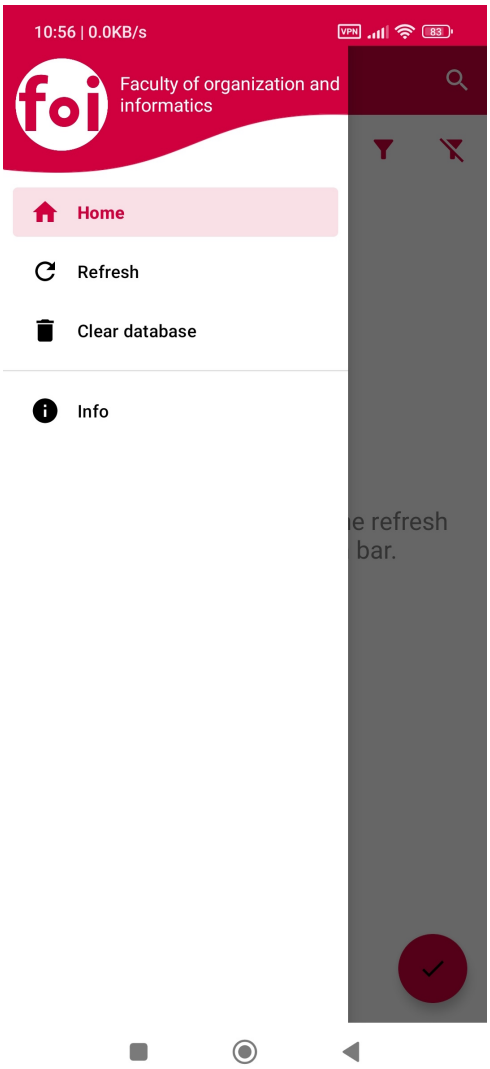
## 6.1. Kotlin

"Kotlin je programski jezik koji kodiranje čini sažetim, višeplatformskim i zabavnim. To je Googleov preferirani jezik za razvoj Android aplikacija." [9] Kreiran od strane JetBrainsa, programera naširoko korištenog integriranog razvojnog okruženja IntelliJ IDEA, Kotlin daje prednost izražajnosti i sigurnosti. Ovaj je jezik svestran i može se koristiti u raznim vrstama aplikacija, u rasponu od Android mobilnih aplikacija do poslužiteljskih aplikacija i dalje. Jedna od značajnih značajki Kotlina je njegova sposobnost kombiniranja najboljih aspekata funkcionalnog i objektno orijentiranog programiranja, što rezultira lakšim održavanjem koda, smanjenim brojem pogrešaka i ubrzanim razvojem. Dodatno, Kotlin je kompatibilan s Java platformom, omogućujući razvojnim programerima da iskoriste postojeće Java biblioteke i okruženja dok imaju koristi od brojnih modernizacija i mogućnosti pisanja sažetog i organiziranijeg koda koji Kotlin olakšava.

## 6.2. Razvoj aplikacije

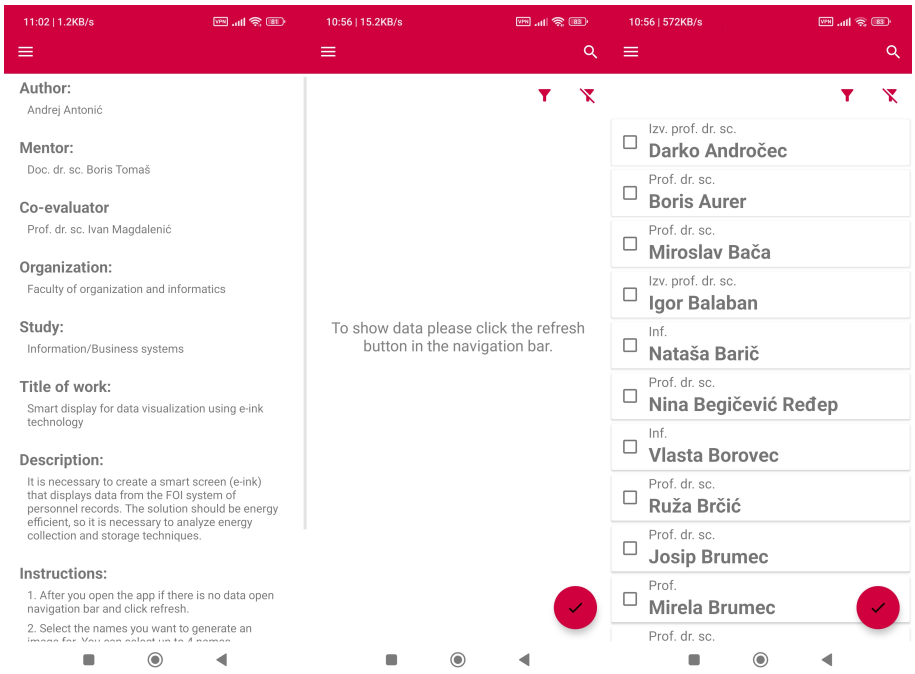
### 6.2.1. Korisničko sučelje

Dobro korisničko sučelje jako je važno za uspješnu mobilnu aplikaciju. Omogućuje korisnicima koji koriste aplikaciju da se osjećaju ugodno, lako obavljaju stvari i ostaju zainteresirani za korištenje aplikacije dugo vremena. Aplikacija se sastoji od dvije aktivnosti i dva fragmenta. Prva aktivnost zapravo samo sadrži upravitelj fragmenata i navigacijsku traku sa lijeve strane ekrana. Pomoću navigacijske trake korisnik može navigirati između dva fragmenta te ima dvije dodatne opcije za učitavanje podataka putem API-ja i brisanja podataka iz baze podataka.




Slika 6: Navigacijska traka (Samostalna izrada, 2023.)

Nadalje, prvi fragment samo sadrži osnovne informacije o aplikaciji, temi i završnom radu. Drugi fragment se koristi za prikazivanje svih podataka koji su dohvaćeni putem API-ja. Ukoliko se aplikacija pali prvi put fragment neće prikazivati nikakve podatke te se trebaju inicijalno učitati pomoću opcije "Osvježi/Refresh" u navigacijskoj traci. Nakon što se podaci učitaju svaki put kada se aplikacija restarta podaci će ostati prikazani. Na fragmentu su također vidljive opcije za pretraživanje i sortiranje podataka. Korisnik može odabrati maksimalno četiri osobe za koje će se generirati slika koja se kasnije može prikazati na e-paper ekranu.




Slika 7: Fragmenti (Samostalna izrada, 2023.)

Kada je odabrano između jedan i četiri zaposlenika i pritisne se gumb u donjem desnom kutu ekrana otvara se nova aktivnost u kojoj se generira slika za odabrane zaposlenike i omogućuje prijenos slike do e-paper ekrana putem NFC-a. Na aktivnosti također postoji gumb za spremanje slike u galeriju i vraćanja na prošlu aktivnost. Ovisno o tome koliko je zaposlenika odabrano generira se jedna od četiri moguće varijacije slike.


Slika 8: Generirane slike (Samostalna izrada, 2023.)

### 6.2.2. Podaci kadrovske evidencije

Svi potrebni podaci su dohvaćeni putem FOI API-ja za kadrovsku evidenciju. Dohvaćanje podataka putem API-ja u Kotlinu česta je pojava u suvremenom programiranju. Ova praksa programerima omogućuje komunikaciju s udaljenim poslužiteljima i dohvaćanje podataka koji nisu dostupni unutar lokalnog okruženja.

#### 6.2.2.1. Biblioteke

Biblioteke u programskom jeziku Kotlin sastoje se od širokog raspona unaprijed određenih funkcija, klasa i alata koji služe za pojednostavljenje procesa razvoja aplikacija. Iskorištavanjem ovih biblioteka, programeri mogu koristiti već postojeću funkcionalnost umjesto da je moraju stvarati iz temelja, što rezultira primjetnim ubrzanjem procesa razvoja i čini kod čišćim. U programskom jeziku Kotlin biblioteke se uključuju u konfiguracijskoj datoteci, odnosno build.gradle.

```
dependencies {
    implementation 'com.squareup.retrofit2:retrofit:2.9.0'
    implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
}
```

Retrofit je široko korištena biblioteka Android HTTP klijenta koja olakšava upućivanje API poziva. Nudi jednostavno sučelje za definiranje krajnjih točaka API-ja, metoda zahtjeva i vrsta odgovora. Ovisnosti converter-gson omogućuje učinkovito analiziranje JSON odgovora putem Gson podrške.

```
dependencies {
    kapt "androidx.room:room-compiler:2.5.2"
    implementation "androidx.room:room-runtime:2.5.2"
    annotationProcessor "androidx.room:room-compiler:2.5.2"
}
```

Room je Androidova biblioteka koja služi kao vrijedan alat za lokalno pohranjivanje podataka, posebno za baze podataka. Iako njegova primarna svrha nije usredotočena na dohvaćanje podataka iz API-ja, igra ključnu ulogu u pohranjivanju i učinkovitom upravljanju podacima unutar same aplikacije, što može uključivati informacije dobivene iz API-ja.

#### 6.2.2.2. Dohvaćanje putem API-ja

Prvi korak za dohvaćanje podataka putem API-ja je kreirati model. Model je nužan jer omogućuje mapiranje dohvaćenih podataka na lokalni model što puno olakšava rad sa podacima. U modelu se mogu definirati samo podaci koji će biti potrebni za daljnji rad.

```
@Entity(tableName = "personnel")
data class Personnel(
    @PrimaryKey val id : Int,
    val title : String,
    val firstName : String,
    val lastName : String
```

```
) : Serializable
```

Nakon modela potrebno je definirati sučelje. Proces definiranja API sučelja od iznimne je važnosti jer nudi preciznu specifikaciju i sporazum za komunikaciju između vaše aplikacije i vanjskog izvora. Ovo ne samo da poboljšava sigurnosni aspekt, već i pojednostavljuje integraciju, održavanje i testiranje vašeg koda. Štoviše, olakšava prilagodbu svim potencijalnim promjenama u API-ju. U sučelju za ovu aplikaciju potrebna je samo jedna funkcija koja je odgovorna za dohvaćanje svog osoblja.

```
interface ApiInterface {
    @GET("users")
    fun getAllPersonnelData(): Call<UserResponse>
}
```

Servis za komunikaciju s API-jem koristan je jer odvaja komunikacijski sloj od ostalog koda, omogućava ponovnu upotrebu, olakšava testiranje, prilagodbu promjenama API-ja te centralizira komunikacijsku logiku za bolju organizaciju i održavanje aplikacije.

```
object ApiService {
    private const val BASE_URL = "https://kadrovska.foi.hr/foi-api/public/"

    private val retrofit = Retrofit.Builder()
        .addConverterFactory(GsonConverterFactory.create())
        .baseUrl(BASE_URL)
        .build()

    private val apiInterface: ApiInterface = retrofit.create(ApiInterface::class.
        java)
    private val responseHandler = ResponseHandler()

    suspend fun getPersonnelData(): UserResponse? {
        return suspendCoroutine { continuation ->
            val retrofitData = apiInterface.getAllPersonnelData()

            retrofitData.enqueue(object : Callback<UserResponse?> {
                override fun onResponse(call: Call<UserResponse?>, response:
                    Response<UserResponse?>) {
                    val responseBody = response.body()
                    if (responseBody != null) {
                        val personnelList = responseHandler.
                            convertResponseToPersonnelList(responseBody)
                        PersonnelDatabase.getInstance().getPersonnelDAO().
                            insertPersonnel(personnelList)
                        continuation.resume(responseBody)
                    } else {
                        continuation.resume(null)
                    }
                }

                override fun onFailure(call: Call<UserResponse?>, t: Throwable) {
                    Log.d("Data␣fetching␣failure", "Failure␣while␣fetching␣API␣data:
                        ␣${t.message}")
                    continuation.resume(null)
```

```
                }
            })
        }
    }
}
```

- Definiranje konstante za bazni URL: sadrži osnovni URL za API.
- Inicijalizacija Retrofit objekta: retrofit je inicijaliziran pomoću Retrofit.Builder(). Postavljen je da koristi GsonConverterFactory za konvertiranje JSON odgovora u Kotlin objekte. Također se postavlja bazni url.
- Definiranje prethodno kreiranog sučelja za API komunikaciju
- Definiranje response handler objekta
- Dohvaćanje podataka o osoblju: getPersonnelData() je suspenzivna funkcija koja koristi Coroutines za asinkrono dohvaćanje podataka. Koristi enqueue() metodu za izvršavanje asinkronog HTTP poziva na API. Kada odgovor stigne, koristi se ResponseHandler za obradu odgovora, a zatim se odgovor prosljeđuje natrag koristeći continuation.resume().
- convertResponseToPersonnelList(): jednostavna funkcija koja mapira UserResponse tip koji dobijemo u listu zaposlenika.

#### 6.2.2.3. Baza podataka

Nakon što se prvi put dohvate podaci svi se automatski pohrane u lokalnu bazu podataka. Ovaj pristup osigurava puno brže prikazivanje podataka, smanjuje se opterećenje na API i osigurava rad u mrežno nesigurnim uvjetima. Podaci se prilikom svakog dohvaćanja spremaju u bazu. Za ovu aplikaciju korištena je lokalna SQLite baza podataka.

Prvi korak je definirati podatkovni entitet za bazu podataka. Već imamo entitet koji koristimo za podatke, tako da ćemo ga samo prilagoditi kako bi predstavljao tablicu u bazi. To se napravi tako što se iznad klase entiteta stavi anotacija @Entity(tableName = "ime_tablice"), te na jedan atribut stavimo primarni ključ. U entitetu personnel primarni ključ će biti atribut id.

Nakon podatkovnog entiteta potrebno je napraviti pristupni objekt. Pristupni objekt predstavlja sučelje koje će se koristit za pristup i manipulaciju bazom podataka. Zadaća programera je samo definirati sučelje, dok ostatak koda u pozadini implementira biblioteka Room. U ovom sučelju definirane su tri metode koje se koriste za dohvaćanje iz baze podataka, zapisivanje i brisanje iz baze podataka.

```
@Dao
interface PersonnelDAO {
    @Query("SELECT␣*␣FROM␣personnel")
    fun getAllPersonnel(): List<Personnel>

    @Insert(onConflict = REPLACE)
    fun insertPersonnel(personnelList: List<Personnel>): List<Long>
```

```
    @Query("DELETE␣FROM␣personnel")
    fun deleteAllPersonnel()
}
```

Zadnji korak je kreirati klase koje upravljaju bazom podataka. "To je anotirana klasa koja je apstraktna, a nasljeđuje klasu RoomDatabase. Apstraktna je jer prije korištenja zahtijeva još pozadinske implementacije generiranja i dohvata DAO objekata, pa ju kao takvu zapravo „dovršava“ biblioteka Room." [10] Klasa ima jednu apstraktnu metodu za dohvaćanje zaposlenika. Također ima jedan companion object. Companion object se koristi kako bi omogućio definiranje statičkih metoda i svojstava. Pošto ovaj objekt dijeli svojstva klase, ali nema vezu s instancama klase moguće je pristupiti svojstvima i metodama unutar companion objecta izravno preko imena klase, a ne putem instance. Pošto je prva klasa apstraktna i nije ju moguće instancirati ovo je savršen način da klasi dodijeli referenca na instancu objekta njene implementacije. Sam companion object se sastoji od dvije metode od kojih se jedna koristi za vraćanje instance baze podataka, a druga za kreiranje i konfiguriranje nove instance baze podatka.

```
@Database(version = 1, entities = [Personnel::class], exportSchema = false)
abstract class PersonnelDatabase: RoomDatabase() {
    abstract fun getPersonnelDAO(): PersonnelDAO

    companion object {
        @Volatile
        private var implementedInstance: PersonnelDatabase? = null

        fun getInstance(): PersonnelDatabase {
            if(implementedInstance == null)
                throw NullPointerException("Database␣instance␣has␣not␣yet␣been␣
                    created!")
            return implementedInstance!!
        }

        fun buildInstance(context: Context) {
            if(implementedInstance == null) {
                val instanceBuilder =
                    Room.databaseBuilder(context, PersonnelDatabase::class.java, "
                        personnel.db")
                instanceBuilder.fallbackToDestructiveMigration()
                instanceBuilder.allowMainThreadQueries()
                instanceBuilder.build()

                implementedInstance = instanceBuilder.build()
            }
        }
    }
}
```

#### 6.2.2.4. Prikazivanje podataka

Nakon što su svi podaci dohvaćeni i pohranjenu u bazu podataka potrebno ih je prikazati korisniku. Za prikazivanje podataka je korišten RecyclerView. RecyclerView ključni je dio okvira za razvoj Androida. Omogućuje učinkovitu prezentaciju popisa podataka koji se stalno mijenja ponovnim korištenjem prikazanih elemenata prikaza. Ova pametna tehnika ne samo da smanjuje potrošnju dragocjenih resursa, već također poboljšava ukupnu izvedbu zaslona. Kako bi to postigao, RecyclerView koristi adapter za upravljanje podacima i ViewHolder za svaki element na popisu. Kao rezultat toga, sadržaj prikazanog popisa može se jednostavno i učinkovito ažurirati na dinamičan način bez ikakvih uskih grla u izvedbi.

```
class PersonnelAdapter(private var personnelList: List<Personnel>, private var
    checkedTags: MutableList<Personnel> = mutableListOf()) :
RecyclerView.Adapter<PersonnelAdapter.PersonnelViewHolder>() {

private val selectedItems: MutableList<Personnel> = checkedTags
private val removedItems: MutableList<Personnel> = mutableListOf()

inner class PersonnelViewHolder(view: View) : RecyclerView.ViewHolder(view) {
    private val personnelCard: CardView = view.findViewById(R.id.cw_personnelCard)
    private val personnelId: CheckBox = view.findViewById(R.id.chb_personnelId)
    private val personnelTitle: TextView = view.findViewById(R.id.tw_personnelTitle)
    private val personnelFirstName: TextView = view.findViewById(R.id.
        tw_personnelFirstName)
    private val personnelLastName: TextView = view.findViewById(R.id.
        tw_personnelLastName)

    init {
        personnelCard.setOnClickListener {
             val position = bindingAdapterPosition
            if (position != RecyclerView.NO_POSITION) {
                val personnel = personnelList[position]
                val isChecked = !personnelId.isChecked

                if (selectedItems.size != 4 || !isChecked) {
                    personnelId.isChecked = isChecked
                    if (isChecked) {
                         selectedItems.add(personnel)
                        if(removedItems.contains(personnel))
                            removedItems.remove(personnel)
                    } else {
                        selectedItems.remove(personnel)
                        removedItems.add(personnel)
                    }
                }
            }
        }
    }

    fun bind(personnel: Personnel) {
        personnelId.isChecked = checkedTags.contains(personnel)
        personnelId.tag = personnel.id
```

```
            personnelTitle.text = personnel.title
            personnelFirstName.text = personnel.firstName
            personnelLastName.text = personnel.lastName
        }
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
        PersonnelViewHolder {
        val personnelView = LayoutInflater.from(parent.context)
            .inflate(R.layout.personnel_records_item, parent, false)
        return PersonnelViewHolder(personnelView)
    }

    override fun getItemCount() = personnelList.size
    override fun onBindViewHolder(holder: PersonnelViewHolder, position: Int) {
        holder.bind(personnelList[position])
    }
```

U kodu se može vidjeti adapter koji se koristi za prikazivanje podataka u RecyclerView-u. Adapter prati listu zaposlenika, označene stavke i uklonjene stavke te omogućava interakciju putem korisničkog sučelja. Adapter u sebi sadrži ViewHolder koja povezuje elemente sučelja za podatke iz liste zaposlenika. Također implementira klikanje kako bi omogućilo označavanje i odznačavanje osoblja, uz ograničenje do maksimalno četiri označene stavke. Kroz metode onCreateViewHolder, onBindViewHolder i getItemCount adapter omogućuje RecyclerView-u pravilno stvaranje i prikazivanje redaka sa podacima. Klik na redak osoblja mijenja status označenosti i upravlja dodavanjem/uklananjem iz odgovarajućih list.

### 6.2.3. Generiranje slike

Nakon što korisnik izabere željene zaposlenike i pritisne gumb u donjem desnom kutu ekrana otvara se nova aktivnost na kojoj se može vidjeti generirana slika. Potrebno je generirati tri različite slike. Prva slika koja se generira je finalna slika koja će se vidjeti na e-paper ekranu i koja se vidi u canvasu na aktivnosti. Za pohranjivanje i prikaz slika koristi se bitmap. Bitmapa služi kao temeljni prikaz digitalne slike u području računalne grafike. Ima oblik dvodimenzionalne rešetke koja se sastoji od brojnih piksela, od kojih svaki zasebno utjelovljuje sićušni dio cjelokupne slike i posjeduje svoju posebnu boju i intenzitet. Često se naziva "rasterska slika".

```
fun generateImageBitmap(personnelList: ArrayList<Personnel>, context: Context):
    Bitmap {
    val foiLogo = ContextCompat.getDrawable(context, R.drawable.foilogo75x65)
    val bitmap = Bitmap.createBitmap(imageWidth, imageHeight, Bitmap.Config.
        ARGB_8888)
    var canvas = Canvas(bitmap)
    canvas.drawColor(Color.WHITE)

    foiLogo?.let {
        val logoWidth = imageWidth / 5
        val logoHeight = imageHeight / 5
```

```
        val logoBitmap = Bitmap.createBitmap(logoWidth, logoHeight, Bitmap.Config.
            ARGB_8888)
        val logoCanvas = Canvas(logoBitmap)

        it.setBounds(0, 0, logoWidth, logoHeight)
        it.draw(logoCanvas)

        val logoLeft = 5
        val logoTop = 5
        canvas.drawBitmap(logoBitmap, logoLeft.toFloat(), logoTop.toFloat(), null)
    }

    when(personnelList.size) {
        1 -> canvas = addOnePersonnel(personnelList, canvas)
        2 -> canvas = addTwoPersonnel(personnelList, canvas)
        3 -> canvas = addThreePersonnel(personnelList, canvas)
        4 -> canvas = addFourPersonnel(personnelList, canvas)
    }

    return bitmap
}
```

- Funkcija koja generira i vrača bitmapu na temelju prosljeđenih zaposlenika.
- Dohvaćanje vektorske slike logotipa FOI-a iz resursa aplikacije.
- Kreiranje prazne bitmape sa dimenzijama koje odgovaraju e-paper ekrani, u ovom slučaju 400x300. ARGB_8888 omogućava prozirnost i bogate boje.
- Kreiranje platna (canvas) za crtanje na bitmapi i cijelo se popunjava bijelom bojom.
- Ako je logotip FOI-a dostupan, njegova veličina se smanjuje na pola i crta u gornjem desnom kutu platna.
- Ovisno o broju osoblja u listi ulazi se u različite funkcije kako bi se dodao odgovarajući broj zaposlenika.

```
private fun addOnePersonnel(receivedList: ArrayList<Personnel>, canvas: Canvas):
    Canvas {
    val firstPersonnel = receivedList[0]
    val title = firstPersonnel.title
    val name = firstPersonnel.firstName + "␣" + firstPersonnel.lastName

    val textTitle = Paint().apply {
        textSize = 20f
        color = Color.BLACK
    }
    val textName = Paint().apply {
        textSize = 35f
        color = Color.BLACK
    }
    val textWidth = textName.measureText(name)
```

```
    val x = canvas.width / 2f - textWidth / 2f

    canvas.drawText(title, x, 140f, textTitle)
    canvas.drawText(name, x, 180f, textName)

    return canvas
}
```

Ovisno o odabranom broju zaposlenika tekst se dodaje na različite načine. Ako je odabrano više od jednog zaposlenika imena se poredaju po dužini, od najkraćeg do najdužeg. Nakon što je definirana veličina i boja teksta dodaje se na proslijeđeno platno.

Druge dvije slike koje se generiraju se koriste prilikom slanja za prikazivanje na e-paper ekranu. Prva od njih je samo FOI logotip, a druga imena i titule zaposlenika. Kod za generiranje tih slika je isti kao i prošli samo podijeljen na dva dijela.

### 6.2.4. Slanje putem NFC-a

Prije početka rada sa NFC-em potrebno je dodati odgovarajuće dozvole i registrirati aktivnost koja će odrađivati NFC operacije u AndroidManifest datoteku. AndroidManifest bitna je XML datoteka koja se koristi u razvoju Androida, a služi u svrhu određivanja različitih pojedinosti o aplikaciji. Ova datoteka uključuje metapodatke koji pružaju sveobuhvatan opis karakteristika aplikacije, komponenti koje ona obuhvaća i načina na koji je namijenjena interakciji s drugim aplikacijama i cjelokupnim sustavom.

```
<uses-permission android:name="android.permission.NFC" />
<uses-feature
    android:name="android.hardware.nfc"
    android:required="true" />
...
<intent-filter>
    <action android:name="android.nfc.action.TAG_DISCOVERED" />
    <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

Za slanje naredbi, podataka i samu provjeru je li NFC podržan na uređaju koristi se NfcAdapter klasa. Ona predstavlja sučelje za upravljanje NFC funkcionalnošću na Android uređaju. Nakon što je klasa instancirana potrebno je implementirati aktiviranje i deaktiviranje NFC mogućnosti. Za to se koriste metode onResume() i onPause().

```
override fun onResume() {
    super.onResume()

    if(nfcAdapter == null) {
        Toast.makeText(this, getString(R.string.nfcNSupported), Toast.LENGTH_SHORT).
            show()
        return
    }
    else if(!nfcAdapter.isEnabled) {
        checkNFC()
        return
```

```
    }

    if(nfcAdapter != null) {
        try {
            val TECHLISTS = arrayOf(arrayOf(IsoDep::class.java.name), arrayOf(NfcV::
                class.java.name), arrayOf(NfcF::class.java.name), arrayOf(NfcA::
                class.java.name))
            val FILTERS = arrayOf(IntentFilter(NfcAdapter.ACTION_TECH_DISCOVERED, "
                */*"))

            val intent = Intent(this, CanvasActivity::class.java)
            intent.flags = Intent.FLAG_ACTIVITY_SINGLE_TOP
            val pendingIntent = PendingIntent.getActivity(this, 0, intent,
                PendingIntent.FLAG_MUTABLE)
            nfcAdapter.enableForegroundDispatch(this, pendingIntent, FILTERS,
                TECHLISTS) NfcAdapter.FLAG_READER_SKIP_NDEF_CHECK, null)
        }
        catch (e: Exception) {
            e.printStackTrace()
            Log.e("debug", "Exception␣in␣onResume")
        }
    }
}

override fun onPause() {
    super.onPause()

    if(nfcAdapter != null)
        nfcAdapter.disableForegroundDispatch(this)
}
```

Metoda onResume() se poziva kada aktivnost ponovno postane vidljiva korisniku, obično nakon što je pauzirana ili pokrenuta.

- Ako nfcAdapter nije dostupan prikazuje se kratka poruka. Ovo se izvršava ako nema podrške za NFC na uređaju.
- Ako nfcAdapter postoji, provjerava se je li NFC omogućen na uređaju. Ako nije poziva se metoda checkNFC() korisniku prikaže dijalog te ga odvede u postavke kako bi uključio NFC.
- Ako je podrška za NFC dostupna i NFC je omogućen priprema se "Foreground Dispatch" kako bi se aktivnost registrirala za primanje NFC intenta. Potrebno je definirati koje NFC tehnologije aplikacija podržava, akcija koja će se pokrenuti kada je otkrivena određena NFC tehnologija, postaviti zastavice i kreirati intent i pending intent koji se koriste za pokretanje ciljane aktivnosti kada se NFC intent otkrije.

Metoda onPause() se poziva kada aktivnost gubi fokus i postaje neaktivna. U ovom dijelu koda se samo dektivira "Foreground Dispatch". Deaktiviranje "Foreground Dispatch"

sprječava aplikaciju da nepotrebno obrađuje NFC intente kada to više nije potrebno ili kada aplikacija nije aktivna.

Nakon što aplikacija pronađe novi NFC intent pokreće se metoda onNewIntent(). Ova metoda obrađuje dolazni intent koji sadrži NFC podatke te je odgovorna za slanje NFC komandi.

```
override fun onNewIntent(intent: Intent?) {
    super.onNewIntent(intent)

    val p: Parcelable = intent?.getParcelableExtra(NfcAdapter.EXTRA_TAG) ?: return
    val tag = p as Tag
    val nfcA = NfcA.get(tag)
    if(nfcA != null) {
        thread {
            try {
                 nfcA.connect()

                var cmd: ByteArray
                var response: ByteArray
                nfcA.timeout = 1200

                cmd = byteArrayOf(0xCD.toByte(), 0x80.toByte())
                response = nfcA.transceive(cmd)
                Log.e("Inicijaliziranje␣pinova", HexToString(response))
                Thread.sleep(200)

                cmd = byteArrayOf(0xCD.toByte(), 0x81.toByte())
                response = nfcA.transceive(cmd)
                Log.e("Soft␣reset", HexToString(response))
                Thread.sleep(200)

                cmd = byteArrayOf(0xCD.toByte(), 0x82.toByte())
                response = nfcA.transceive(cmd)
                Log.e("Eink␣step␣2", HexToString(response))
                Thread.sleep(200)

                cmd = byteArrayOf(0xCD.toByte(), 0x83.toByte())
                response = nfcA.transceive(cmd)
                Log.e("Crno/bijeli␣mode", HexToString(response))
                Thread.sleep(200)

                cmd = ByteArray(33)
                var text = pictureUtils.convertPictureToByteArray(textBitmap)
                var length = text.size
                var totalPercent = 0
                var index = 0
                for(i in 0 until length step 30) {
                    cmd[0] = 0xCD.toByte()
                    cmd[1] = 0x85.toByte()
                    cmd[2] = 0x1E
                    for(j in 0 until 30)
                        cmd[j + 3] = text[index++]
```

```
                    response = nfcA.transceive(cmd)
                    totalPercent += 30
                    runOnUiThread {
                        percentageText.text =
                            ((totalPercent.toFloat() / 30000) * 100).toInt().
                               toString() + "%"
                    }
                }
                Thread.sleep(200)

                cmd = byteArrayOf(0xCD.toByte(), 0x84.toByte())
                response = nfcA.transceive(cmd)
                Log.e("Crveni␣mode", HexToString(response))
                Thread.sleep(200)

                cmd = ByteArray(33)
                val logo = pictureUtils.convertPictureToByteArray(logoBitmap)
                length = logo.size
                index = 0
                for(i in 0 until length step 30) {
                    cmd[0] = 0xCD.toByte()
                    cmd[1] = 0x85.toByte()
                    cmd[2] = 0x1E
                    for(j in 0 until 30)
                        cmd[j + 3] = logo[index++]
                    response = nfcA.transceive(cmd)
                    totalPercent += 30
                    runOnUiThread {
                        percentageText.text =
                            ((totalPercent.toFloat() / 30000) * 100).toInt().
                               toString() + "%"
                    }
                }
                Thread.sleep(200)
                cmd = byteArrayOf(0xCD.toByte(), 0x86.toByte())
                response = nfcA.transceive(cmd)
                Log.e("Refresh", HexToString(response))
                Thread.sleep(200)
            } catch (e: Exception) {
                e.printStackTrace()
                Log.e("debug", "Exception␣in␣onNewIntent:␣$e")
            } finally {
                 nfcA.close()
            }
        }
    }
}
```

- Prvo se preuzima objekt tipa Tag iz dolaznog NFC intenta te se dohvati instanca NfcA klase korištenjem prethodno dobivenog NFC taga.
- Ako je instanca NfcA objekta uspješno dohvaćena pokreće se nova dretva kako se glavna

ne bi blokirala tokom komuniciranja sa NFC modulom i slanja slike.

- Koristi se transceive metoda kako bi se slale komande. Komande moraju biti u obliku polja bajtova. Svaki NFC modul i e-paper ekran imaju različite komande, komande za ovaj modul i ekran su:
  - 0xCD 0x80 - inicijalizira pinove i odabire čip.
  - 0xCD 0x81 - soft reset svega.
  - 0xCD 0x82 - pokretanje i postavljanje postavki.
  - 0xCD 0x83 - start ekrana u crno/bijelom modu.
  - 0xCD 0x84 - start ekrana u crvenom modu.
  - 0xCD 0x85 XY [bajtovi] - slanje slike, potrebno je napisati dužinu slike i dodati same bajtove slike za slanje.
  - 0xCD 0x86 - refresh ekrana.
- Prije slanja slika se pretvara u polje bajtova te se šalje trideset po petnaest trideset bajtova. Nakon svakog slanja se prikazuje postotak završenosti.

E-paper ekran prvo prima crno bijelu sliku i nakon toga crvenu sliku. Zbog ovoga je potrebno inicijalnu sliku odvojiti na dva dijela. Prvo se šalje slika koja sadrži imena i titule zaposlenika u crno/bijelom modu i nakon toga se šalje slika sa logotipom u crvenom modu. Boje koje će se prikazati se određuju putem podudarajućih bajtova.

| 0x83 | 0x84 | Boja |
|---|---|---|
| 0x00 | 0x00 | |
| 0xFF | 0xFF | |
| 0x00 | 0xFF | |
| 0xFF | 0x00 | |

Slika 9: Kreiranje boja (Samostalna izrada, 2023)

Za slanje slike se koristi polje bajtova koje mora imati veličinu 15000. Taj broj se može dobiti pretvaranjem broja bitova slike u bajtove.

$$\frac{400 * 300}{8} = 15000$$

Bitmapa se pretvara u polje bajtova korištenjem metode convertPictureToByteArray().

```
fun convertPictureToByteArray(bitmap: Bitmap): ByteArray {
    val byteArray = ByteArray(15000)
    var index = 0
    var temp = 0

    val pixels = IntArray(bitmap.width * bitmap.height)
```

```
    bitmap.getPixels(pixels, 0, bitmap.width, 0, 0, bitmap.width, bitmap.height)
    for(x in (bitmap.height - 1) downTo 0) {
        for(y in 0 until bitmap.width / 8) {
            var tempBinary = ""
            for(z in 0 until 8) {
                val red = Color.red(pixels[temp])
                val green = Color.green(pixels[temp])
                val blue = Color.blue(pixels[temp])

                val luminance = 0.299 * red + 0.587 * green + 0.114 * blue
                if(luminance < 128)
                    tempBinary += "0"
                else
                    tempBinary += "1"
                temp++
            }
            val decimalValue = Integer.parseInt(tempBinary, 2)
            val hexValue = decimalValue.toString(16).toUpperCase()
            byteArray[index] = hexValue.toInt(16).toByte()
            index++
        }
    }

    return byteArray
}
```

- Prvo se svi pikseli slike spremaju u polje za kasnije korištenje.
- Petlje koje prolaze kroz sve piksele u polju. U zadnjoj petlji se sprema binarni zapis osam susjednih piksela. Ukoliko je osvjetljenje piksela manje od 128 sprema se 0, u suprotnom se sprema 1.
- Kada zadnja petlja završi sa izvođenjem binarni zapis se pretvara u heksadekadski te na kraju u bajt i sprema se u polje.

### 6.2.5. Razno

Osim glavne funkciolanosti aplikacija nudi dodatne opcije koje poboljšavaju cjelokupno korisničko iskustvo. Ove dodatne značajke pružaju fleksibilnost i omogućuju korisnicima da promjene jezik te temu aplikacije. Jezici koji su dostupni su engleski i hrvatski i automatski se prilagodi jeziku mobitela. Osim jezika postoji tamni način rada koji se također automatski uključi ukoliko je na uređaju uključen tamni način rada. Na aktivnosti sa prikazom slike postoji mogućnost spremanja generirane slike u galeriju.

```
<uses-permission android:name="android.permission.READ_MEDIA_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Kao i za sve drugo potrebno je definirati dopuštenja u AndroidManifest.

```
fun saveBitmapToGallery(context: Context, bitmap: Bitmap, displayName: String,
    activity: AppCompatActivity) {
    if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.Q) {
        val filename = "${displayName}_${Date().time}"
        val contentValues = ContentValues().apply {
            put(MediaStore.MediaColumns.DISPLAY_NAME, filename)
            put(MediaStore.MediaColumns.MIME_TYPE, "image/jpeg")
            put(MediaStore.MediaColumns.RELATIVE_PATH, Environment.
                DIRECTORY_PICTURES)
        }
        val resolver = context.contentResolver
        val imageUri = resolver.insert(MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
             contentValues)

        try {
            imageUri?.let {
                val outputStream: OutputStream? = resolver.openOutputStream(it)
                outputStream?.use { stream ->
                    bitmap.compress(Bitmap.CompressFormat.PNG, 100, stream)
                }
            }
        } catch (e: Exception) {
            e.printStackTrace()
        }
    }else {
        if (checkWriteStoragePermission(activity)) {
            val path = Environment.getExternalStoragePublicDirectory(Environment.
                DIRECTORY_PICTURES)
            val file = File(path, "${displayName}_${Date().time}.jpg")

            try {
                val outputStream = FileOutputStream(file)
                bitmap.compress(Bitmap.CompressFormat.PNG, 100, outputStream)
                outputStream.close()
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }
    }
}
```

- Prvo se provjerava trenutna verzija Android sustava. Ovo je potrebno pošto od verzije 10 na dalje postoji novi način spremanja u galeriju koji se temelji na upravljanju datotekama putem MediaStore.
- Android 10 i novije
  - Kreira se naziv datoteke i objekt koji sadrži metapodatke o slici.
  - Kreira se objekt za interakciju sa Content Providerom te se pomoću njega slika umetne u Media Store galeriju i dobije njezin URI.

  - Ako je URI uspješno dohvaćen, otvara se OutputStream prema tom URI-ju i koristi se da bi se slika spremila u galeriju u formatu JPEG.

- Prije Android 10
  - Provjera ima li aplikacija dozvolu za pisanje u pohranu uređaja.
  - Definiranje putanje gdje će slika biti spremljena.
  - Stvaranje datoteke za spremanje slike na temelju naziva i putanje.
  - Korištenjem FileOutputStream-a se sprema slika u formatu JPEG.

Video demonstracija aplikacije i komponenti [11]

# 7. Zaključak

U ovom završnom radu, istražena je i detaljno razrađena tema prikazivanja FOI kadrovske evidencije na e-paper ekranu uz korištenje NFC tehnologije. Proučavajući suštinsku prirodu e-paper ekrana, osvjetljavanje među pikselima putem električnih naboja te memoriranje slike bez potrebe za stalnim napajanjem, omogućeno je duboko razumijevanje njihova funkcioniranja i potencijala. Provođenjem sveobuhvatne analize NFC tehnologije i provođenjem usporednog ispitivanja s drugim bežičnim tehnologijama, došlo se do temeljitog razumijevanja njezine primjene u ovome radu. Ovo istraživanje također je istraživalo razne prednosti koje NFC nudi, uključujući njegov ograničeni domet komunikacije i poboljšane sigurnosne značajke.

Pažljiva procjena i odgovarajući odabir određenog NFC modula i e-paper ekrana ishod su temeljitog procesa procjene usmjerenog na pronalaženje najboljeg za određenu implementaciju. Kroz detaljno objašnjenje procesa kreiranja aplikacije u Kotlinu, koja dohvaća podatke putem FOI API-ja za kadrovsku evidenciju, generira sliku i šalje je putem NFC-a, jasno je prikazana veza između teorije i prakse.

Rezultati rada potvrđuju izvedivost koncepta prikazivanja FOI kadrovske evidencije na e-paper ekranu putem NFC-a, pružajući korisnicima praktičan način za pristup i dijeljenje podataka. Kroz ovu integraciju tehnologija, stvorila se sinergija između e-paper ekrana i NFC-a, demonstrirajući kako se moderni bežični protokoli mogu kombinirati s naprednim prikaznim tehnologijama.

# Popis literature


[1] *Elektroforeza — Wikipedia, Slobodna enciklopedija*, [Internet; Pristupljeno 23.07.2023.], 2022. adresa: https://hr.wikipedia.org/wiki/Elektroforeza.

[2] Matsusada Precision, *Electrophoresis*, [Internet; Pristupljeno 23.07.2023.], 2023. adresa: https://www.matsusada.com/application/ps/electrophoresis/.

[3] Wikipedia, *Bistability*, [Internet; Pristupljeno 23.07.2023.], 2023. adresa: https://en.wikipedia.org/wiki/Bistability#/media/File:Bistability_graph.svg.

[4] *Bistability — Wikipedia, Slobodna enciklopedija*, [Internet; Pristupljeno 23.07.2023.], 2023. adresa: https://en.wikipedia.org/wiki/Bistability.

[5] STMicroelectronics, *NFC mode operations*, [Internet; Pristupljeno 29.07.2023.], 2023. adresa: https://www.st.com/content/dam/nfc/nfc-mode-operations.jpg.

[6] *Korišteni NFC modul*, [Internet; Pristupljeno 30.07.2023.] adresa: https://www.good-display.com/product/390.html.

[7] Good Display, *Slika NFC modula*, [Internet; Pristupljeno 30.07.2023.], 2023. adresa: https://www.good-display.com/repository/image/efcefb73-c156-47a4-b5f5-848134e1879e.jpg_640xaf.jpg.

[8] Good Display, *Slika E-ink zaslona*, [Internet; Pristupljeno 30.07.2023.], 2023. adresa: https://www.good-display.com/repository/image/2750aa94-83e4-40fc-8067-b141da15edb7.jpg.

[9] JetBrains, „Kotlin," [Internet; Pristupljeno 10.08.2023.] adresa: https://kotlinlang.org/.

[10] Stapić Z. i Tomaš B. i Peras D. i Matijević M., „RAMPU - V. Lokalna baza podataka," 2023., [Moodle; Pristupljeno 17.08.2023.]

[11] Andrej Antonić, *Kadrovska NFC demo*, [Internet; Pristupljeno 26.08.2023.], 2023. adresa: https://youtu.be/sWp6slRmO1U.

[12] *Electronic paper — Wikipedia, Slobodna enciklopedija*, [Internet; Pristupljeno 23.07.2023.], 2023. adresa: https://en.wikipedia.org/wiki/Electronic_paper.

[13] Margaret Rouse, „Electrophoretic Ink," 2012., [Internet; Pristupljeno 23.07.2023.] adresa: https://www.techopedia.com/definition/25201/electrophoretic-ink-eink.

[14] Yang, B. i Gu, Y. i Xu, J. i Hu, W. i Cao, J. i Zhang, Y. i Chen, P., *Understanding the Mechanisms of E-ink Operation*, [Internet; Pristupljeno 23.07.2023.], 2019. adresa: https://confit.atlas.jp/guide/event-img/idw2019/EP1-3/public/pdf_archive?type=in.

[15] *Application of Electrophoresis*, [Internet; Pristupljeno 23.07.2023.], 2023. adresa: https://www.vedantu.com/chemistry/application-of-electrophoresis.

[16] Mike Smith, „Electrophoresis," 2023., [Internet; Pristupljeno 23.07.2023.] adresa: https://www.genome.gov/genetics-glossary/Electrophoresis.

[17] Ynvisible, „Bistable Displays: A Revolutionary Technology for Low-Power Electronic Displays," [Internet; Pristupljeno 23.07.2023.] adresa: https://www.ynvisible.com/news-inspiration/bistable-displays.

[18] *Bistability*, [Internet; Pristupljeno 23.07.2023.], 2020. adresa: https://www.sciencedirect.com/topics/mathematics/bistability.

[19] Leo Zimmermann, „What is Bistability?," 2023., [Internet; Pristupljeno 23.07.2023.] adresa: https://www.allthescience.org/what-is-bistability.htm.

[20] *Near-field communcation — Wikipedia, Slobodna enciklopedija*, [Internet; Pristupljeno 29.07.2023.], 2023. adresa: https://en.wikipedia.org/wiki/Near-field_communication.

[21] Square, „NFC Guide: All You Need to Know About Near Field Communication," 2022., [Internet; Pristupljeno 29.07.2023.] adresa: https://squareup.com/us/en/the-bottom-line/managing-your-finances/nfc.

[22] Ayusharma0698, „Near Field Communication (NFC)," 2023., [Internet; Pristupljeno 29.07.2023.] adresa: https://www.geeksforgeeks.org/near-field-communication-nfc/.

[23] Zellagui, M., *Wireless Power Transfer – Recent Development, Applications and New Perspectives*. 2021.

[24] Moran Shayovitch, „NFC vs RFID: What's The Difference?," 2021., [Internet; Pristupljeno 29.07.2023.] adresa: https://wlius.com/blog/rfid-vs-nfc-whats-the-difference/.

[25] John Teel, „Comparison of Wireless Technologies (Bluetooth, WiFi, BLE, Zigbee, Z-Wave, 6LoWPAN, NFC, WiFi...," 2018., [Internet; Pristupljeno 29.07.2023.] adresa: https://www.hackster.io/news/comparison-of-wireless-technologies-bluetooth-wifi-ble-zigbee-z-wave-6lowpan-nfc-wifi-eece5593d80f.

[26] James Frew, „7 Cool Ways to Use NFC That'll Impress Your Friends," 2022., [Internet; Pristupljeno 29.07.2023.] adresa: https://www.makeuseof.com/tag/9-awesome-ways-use-nfc-thatll-impress-friends/.

[27] Rfidspecialist, „13 Typical Examples of How to Use NFC technology," 2021., [Internet; Pristupljeno 29.07.2023.] adresa: https://rfidspecialist.eu/13-typical-examples-of-how-to-use-nfc-technology--22-11-2021.html.

[28] Maurizio Di Paolo Emilio, „Powering E-Paper Displays with NFC Energy Harvesting," 2023., [Internet; Pristupljeno 30.07.2023.] adresa: https://www.eetasia.com/powering-e-paper-displays-with-nfc-energy-harvesting/.

[29] Matthew Humphries, „Waveshare's E-Paper Displays Work Without a Battery," 2020., [Internet; Pristupljeno 30.07.2023.] adresa: https://www.pcmag.com/news/waveshares-e-paper-displays-work-without-a-battery.

[30] NoCurrent, „NFC Power Harvesting Explained," [Internet; Pristupljeno 29.07.2023.] adresa: https://www.nucurrent.com/nfc-power-harvesting-explained/.

[31] *Korišteni E-ink zaslon*, [Internet; Pristupljeno 30.07.2023.] adresa: https://www.good-display.com/product/395.html.

[32] Joshua Tzucker, „Info on WaveShare Passive NFC-Powered E-Paper Modules," 2021., [Internet; Pristupljeno 18.08.2023.] adresa: https://joshuatz.com/posts/2021/waveshare-passive-nfc-epaper-modules/.

[33] *ChatGPT*, [Internet; Pristupljeno 18.08.2023.]

[34] Google, „NfcAdapter," 2023., [Internet; Pristupljeno 18.08.2023.] adresa: https://developer.android.com/reference/android/nfc/NfcAdapter.

[35] Google, „Near field communication overview," 2023., [Internet; Pristupljeno 18.08.2023.] adresa: https://developer.android.com/guide/topics/connectivity/nfc.

[36] Google, „MediaStore," 2023., [Internet; Pristupljeno 18.08.2023.] adresa: https://developer.android.com/reference/android/provider/MediaStore.

[37] Good Display, *4.2 inch E-paper Display Series GDEQ042Z21*, [Internet; Pristupljeno 23.08.2023.], 2021. adresa: https://v4.cecdn.yun300.cn/100001_1909185148/GDEQ042Z21.pdf.

[38] Good Display, *NFC Development Kit DENFC-M01*, [Internet; Pristupljeno 23.08.2023.], 2021. adresa: https://v4.cecdn.yun300.cn/100001_1909185148/DENFC-M01.pdf.

[39] Vlastito pitanje, *Converting a bitmap to bytearray to be displayed on an e-paper*, [Internet; Pristupljeno 26.08.2023.], 2023. adresa: https://stackoverflow.com/questions/76912768/converting-a-bitmap-to-bytearray-to-be-displayed-on-an-e-paper.

# Popis slika