# USB Lading

Dette er et utkast for en spesifikasjon for det som i dette dokumentet kalles USB Lading. I det følgende vil ideer og resultater av ulike uttestinger så langt beskrives.

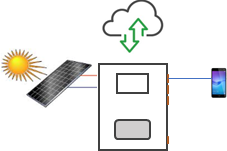
USB Lading er en ladestasjon for lading av mobiltelefoner/nettbrett med *n* antall ladeporter. USB Lading er drevet av et solcellepanel. Ladestasjonen styres av en microprosessor.

Lading skal starte etter at mobil/bruker har verifisert seg og at brukeren blir belastet for en lading. Det må også tas stilling til om en lading er tidsbegrenset eller skal vare så lenge mobilen er koblet til USB porten.

## Forslag 1

I dette forslaget initieres lading ved at verifisering skjer i ladestasjonen og ikke via mobilenheten. Ladestasjonen er utstyrt med et display/tastatur hvor brukeren:

* taster inn en engangskode som hun/han har kjøpt på nett, eventuelt
* taster inn en engangskode som han/hun har kjøpt på i et utsalgssted/kiosk i form at «gavekort» med skrapefelt
* scanner et «gavekort» som han/hun har kjøpt på i et utsalgssted/kiosk



«Gavekortet» har brukeren registrert via en nettside som er en del av løsningen. Via samme nettside kan bruker også fylle på «gavekortet».

Denne løsningen innebærer at ladestasjonen må ha tilgang til nettet for å verifisere betaling. I denne løsningen vil ikke mobilenheten kommuniserer med ladestasjonen og det vil heller ikke være behov for en egen app på mobilenheten – det måtte i så fall være en app/web app hvor man kjøpte ladinger.

## Forslag 2

I denne løsningen vil betaling av ladingen initieres/verifiseres via mobilenheten. Mobilenheten vil via en app/web app kommunisere med ladestasjonen (microprosessoren) for å starte lading. Brukeren må på forhånd ha betalt for lading ved at hun/han har registrert seg som bruker og kjøpt credits – en lading koster én credit.

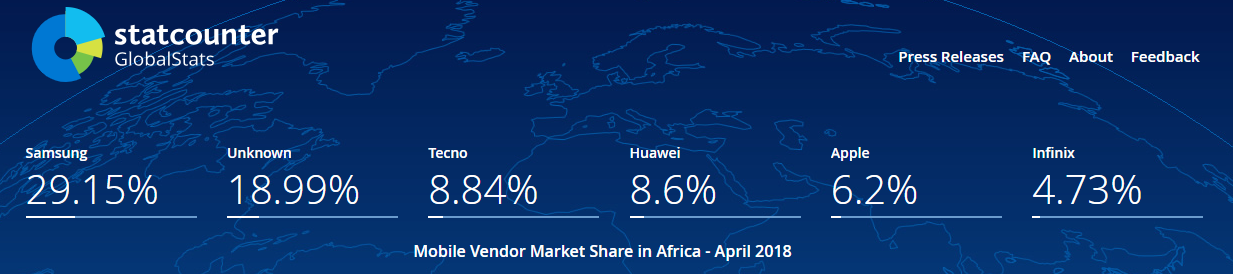
USB Lading må kunne håndtere at en enhet er tom for strøm når den kobles til en USB port. Hvis løsningen innebærer at enheten må identifisere seg eller verifisere at lading er betalt, må ladeporten gi strøm i *n* antall minutter for at brukeren skal kunne lade enheten nok til å verifisere seg via denne. Det må i dette tilfellet være en mekanisme som stenger for lading etter ett forsøk mot samme enhet i en definert tidsperiode; at brukeren ikke kan koble til ulike porter for å lade «gratis» gjentatte ganger.

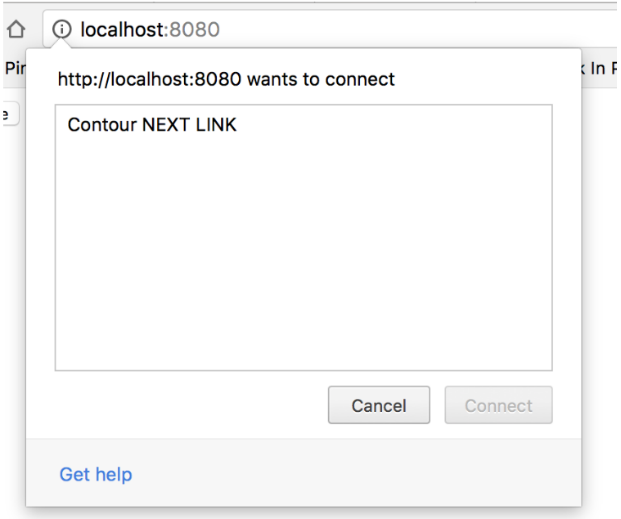
### Verifisering via USB

En mulighet er å verifisere brukeren/enheten via en app/web app som kommunisere mot microprosessoren i ladestsjonen over USB.

#### Web App - WebUSB API i Chrome

Ved å utvikle en (progressive) web app kan man utnytte WebUSB API’et som støttes av Chrome. Per i dag er det kun Android-baserte enheter som støttes av Chrome. Går man for denne løsningen, vil man gå glipp av brukere av iOS- og Windows-baserte enheter. Imidlertid viser tallene under at iOS-baserte enheter kun utgjør ca. 6% av markedet i Afrika (tallene er fra <http://gs.statcounter.com/vendor-market-share/mobile/africa>)



API’et krever at man alltid selekterer hvilken enhet man ønsker å kommunisere mot selv om det kanskje kun er ett som faller inn under kriteriene kan setter, noe som kanskje over tid ikke er veldig brukervennligt. Men siden det er snakk om å gi et nettsted tilgang til enheten, er dette en nødvendighet ut fra sikkerhetsmessige betraktninger.

Web App’en kan nå kommuniserer mot microkontrolleren i ladestasjonen via USB Transfers (control transfers) for å initiere lading.

En ting man må ha i mente ved denne løsningen, er at hvis en eksisterende driver (som f.eks. HID driveren som er bygd inn i OS’et, her Android, har beslaglagt USB enheten, så vil den være utilgjengelig fra WebUSB.

#### Web App – Cordova m/USB plugin

Som et kanskje bedre alternativ til å benytte WebUSB API’et i Chrome, er å utvikle en Web App basert på rammeverket Cordova m/USB plugin -ø eksempelvis [cordova-plugin-usbserial](https://www.npmjs.com/package/cordova-plugin-usbserial). Web App’en utvikles da i HTML, CSS og JavaScript. Imidlertid gjelder det samme som for WebUSB, at dette kun støttes mot Android.

#### App – USB Host Mode

Å utvikle en app som lastes ned og installeres på enheten, er en kjent og sikker tilnærming. Med en app på en Android enhet vil man kunne etablere kommunikasjon mot ladestasjonen via Android USB Host Mode; funksjonalitet som er innebygd i Android OS’et. På en iOS-basert enhet imidlertid, kan det se ut til å være et problem - sannsynligvis må man da melde seg inn i Apple’s sitt MFi program.

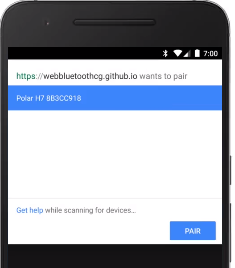
Hvis målet er en cross-plattform løsning, synes ikke veien via USB kommunikasjon mot ladestasjonen å være det beste alternativet.

### Verifisering via Bluetooth

Et godt alternativ til WebUSB er å benytte WebBluetooth som også er innebygd i Chrome og følgelig gir tilgang kommunikasjon mot ladestasjonen over bluetooth direkte fra nettleseren.

#### Web App - WebBluetooth API i Chrome

I likhet med WebUSB støttes WebBluetooth API’et kun mot Android-baserte enheter og følgelig vil det være en del av mobil-markedet man da ikke dekker. Analogt med WebUSB krever WebBluetooth også brukerens samtykke ved å la hun/han velge hvilken enhet man ønsker å pare mot.



I motsetning til WebUSB unngår man problemet, eller det som kunne være et problem, ved at USB enheten kunne være beslaglagt av en driver innebygd i OS’et.

#### Web App – Cordova m/Bluetooth plugin

Også for bluetooth kommunikasjon mot ladestasjon, vil det som som et alternativ til WebBluetooth i Chrome, være et godt valg å utvikle en Web App basert på rammeverket Cordova m/Bluetooth plugin – eksempelvis [cordova-plubin-ble-central](https://www.npmjs.com/package/cordova-plugin-ble-central). Web App’en utvikles i HTML, CSS og JavaScript. Denne løsningen støtter både iOS og Android.

#### App – Bluetooth

I en cross-plattform løsning vil sannsynligvis Bluetooth kommunikasjon mot ladestasjonen være et godt alternative. iOS-baserte enheter vil støtte bluetooth kommunikasjon med sitt Core Bluetooth framework, mens Android-baserte enheter støtter bluetooth med sitt Bluetooth API.

Når det gjelder microkontrolleren i ladestasjonen som mobilenhetene skal kommunisere mot, f.eks NRF52832 (nevnt av Rudi) er bluetooth en god og velutprøvd kommunikasjonsmekanisme.

### Bluetooth Protokoll

I det følgende er et forslag på en enkel protokoll mellom app (mobilenhet) og microkontroller over bluetooth.

Lading initiers ved at bruker trykker på en «start charge» knapp i appen som da sender en *requestPort* forespørsel til ladestasjonen. Siden «start charge» knappen er tilgjengelig, har appen verifisert at lading er betalt for og tikker ned på credits-saldoen - én credit pr lading.

portRequest ({

id**:** String

});

Ladestasjonen kvitterer ved å returnere hvilken USB port som skal brukes.

portResponse ({

port**:** Number

});

Ved feil returnerer Ladestasjonen en feilkode/-melding.

portResponse ({

error**:** Number,

reason**:** String

});

# Backend plattform

Valget av plattform for backend løsningen må vurderes ut fra

· Skalerbarhet

· Funksjonalitet

· Kostnad

## Backend teknologier

Valg av teknologi for backend løsningen kan ha en innvirkning på utviklingen av løsningen. Følgende teknologier bør vurderes:

· Azure

· Google Cloud / Firebase

Alle disse støtter Node.js noe som gir en fordel siden man da programmerer i JavaScript. I både Azure- og Google Cloud løsningen kan man dessuten velge og å bruke MongoDB som database.

## Backend DB

Backend løsningen må holde informasjon om brukere, kjøp av ladetid (credits), ladestasjoner (lokasjon) og ladinger.

Skjemadefinisjon av bruker:

const User **=** **new** Schema({

name**:** ObjectId,

password**:** String,

currentCredits**:** Number,

lastPurchase**:** Date

charges**:** Number

});

Skjemadefinisjon av kjøpslogg:

const PurchaseLog **=** **new** Schema({

name**:** ObjectId,

credits**:** Number,æoij

purchased**:** Date

});

Skjemadefinisjon av lokasjon:

const Location **=** **new** Schema({

location**:** ObjectId,

ports**:** Number,

created**:** Date

});

Skjemadefinisjon av lokasjonlogg:

const LocationLog **=** **new** Schema({

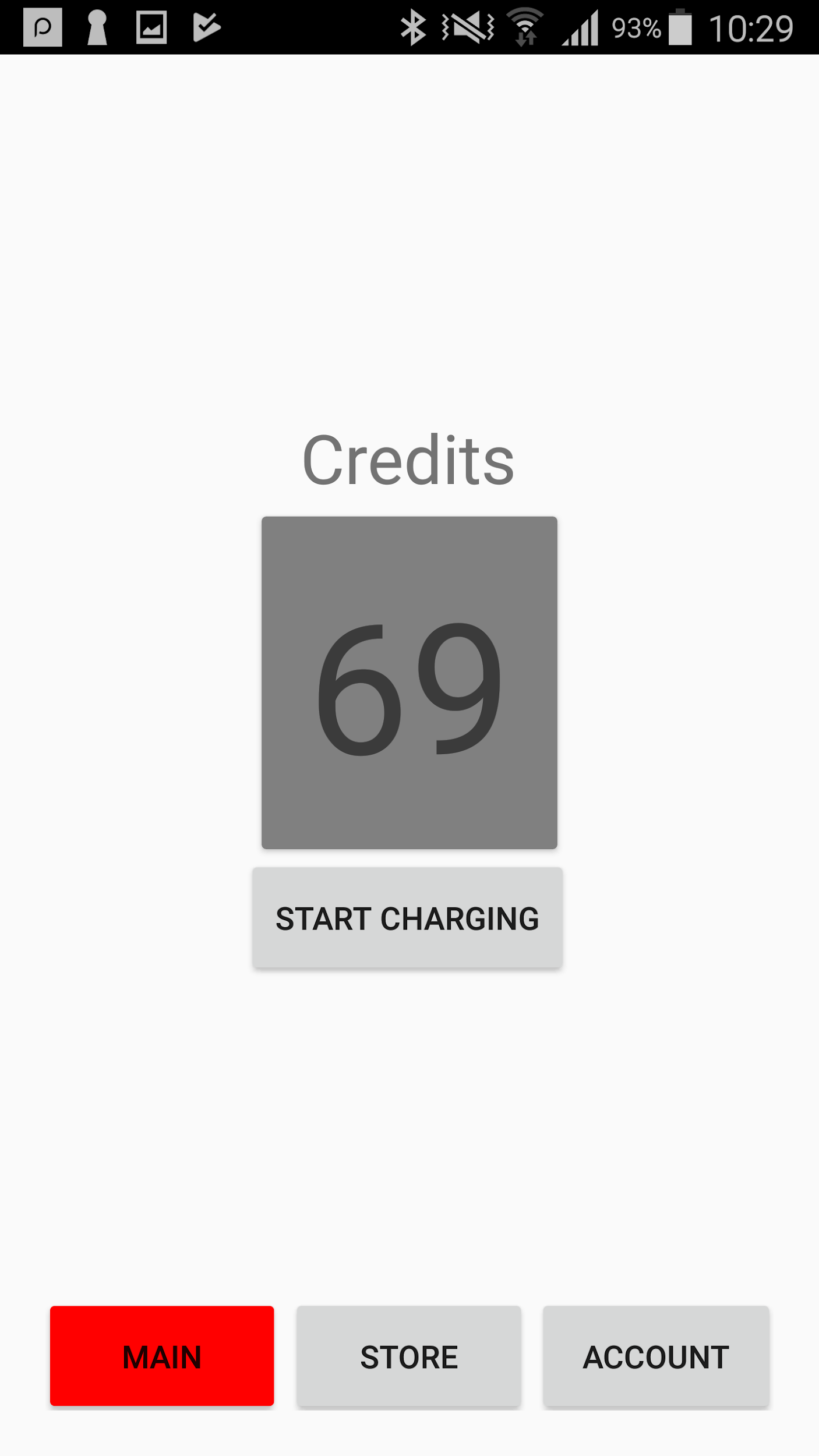
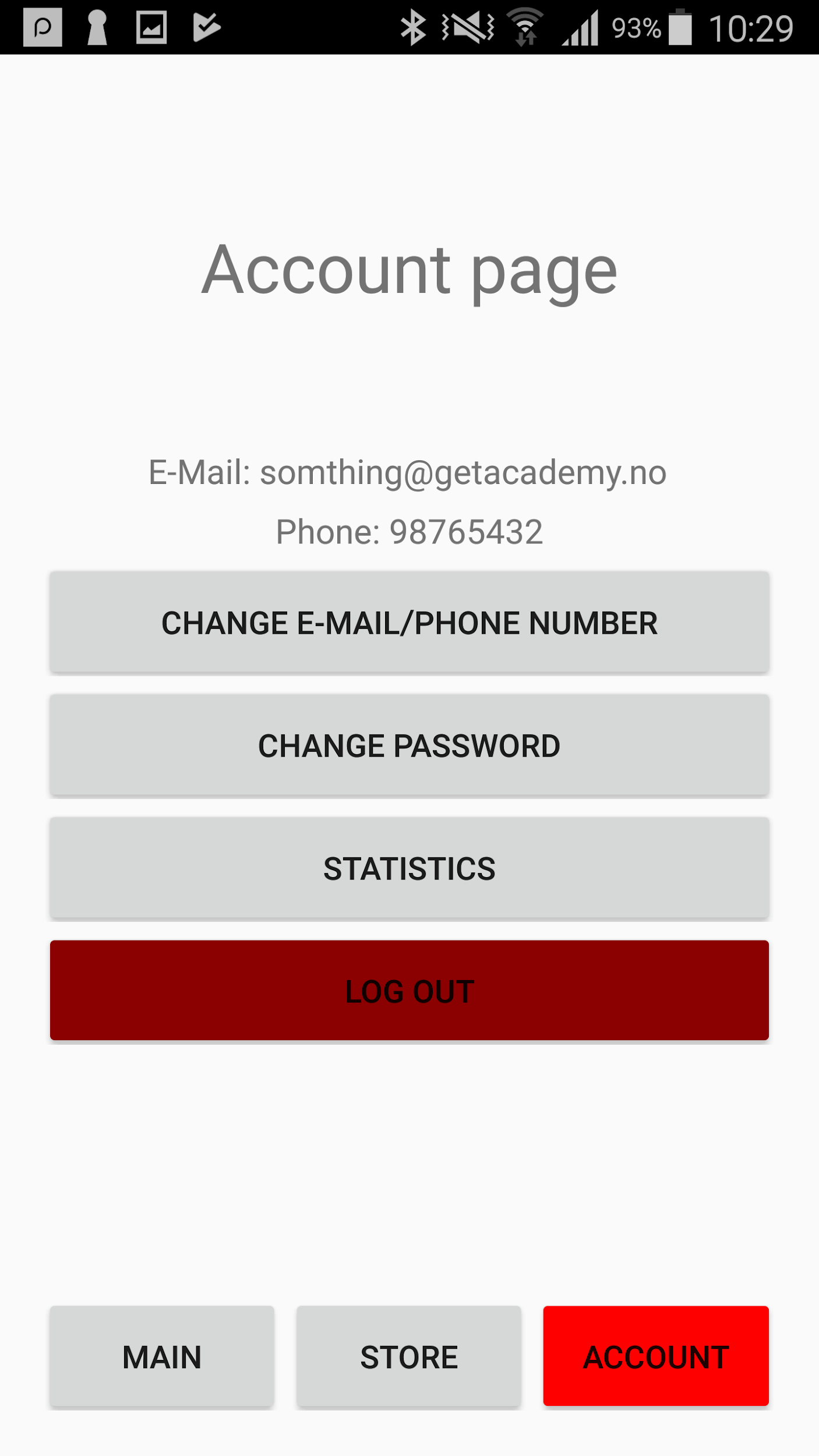
location**:** ObjectId,

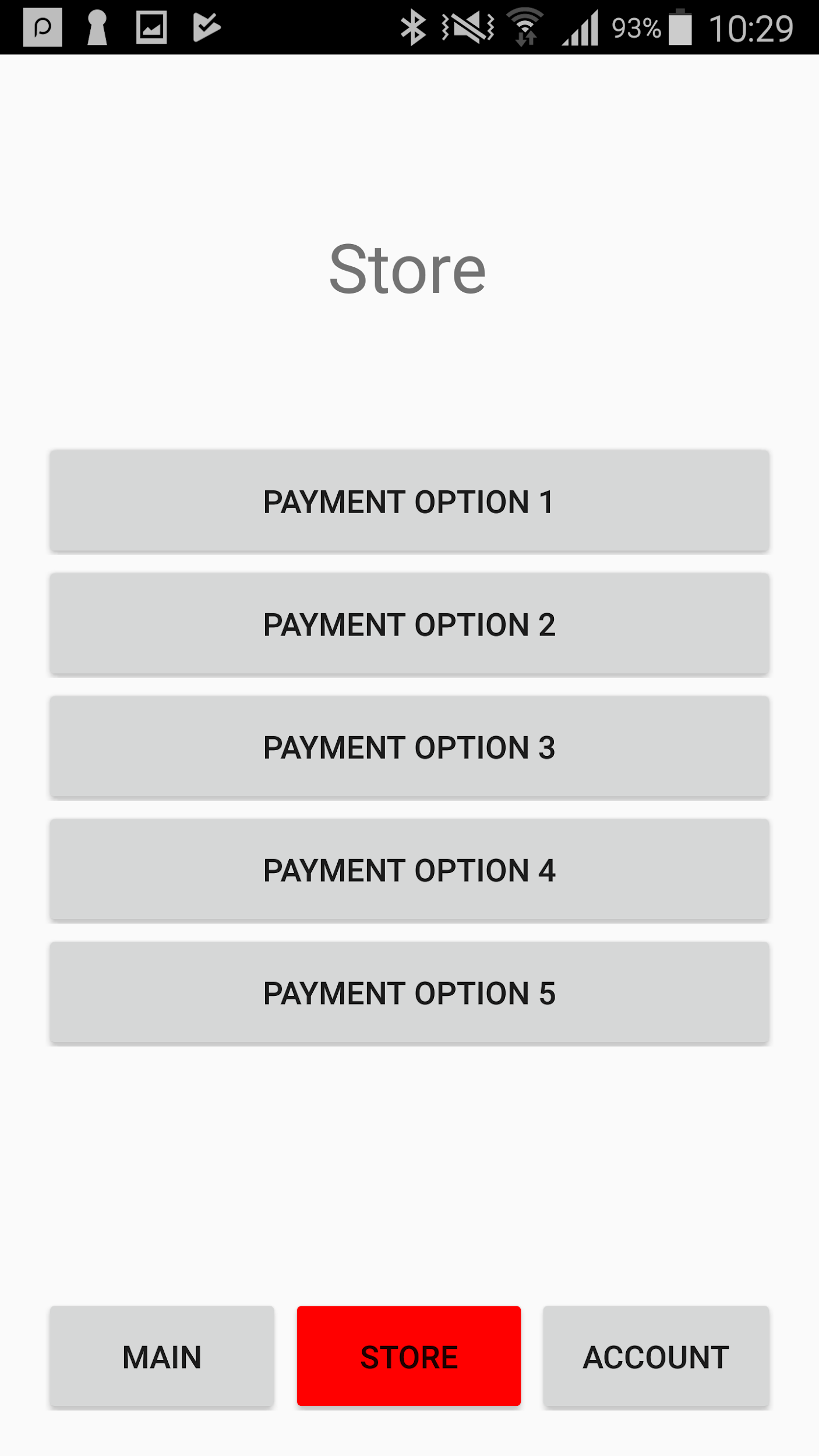
time**:** Number,

when**:** Date

});

Android





# 

