Mappe 2, Apputvikling

Andreas Strand s305036

Jeg har laget en app som lagrer informasjon om studenter, hvor man har mulighet til å sende meldinger til studentene på forskjellige tidspunkt. Alle spesifikasjoner i oppgaven er implementert.

Jeg har gått litt utover det som sto i oppgaveteksten på et par punkter. For det første så benytter jeg en annen type database-aksess som heter 'Android Room Persistence Library'. Dette er et rammeverk som er laget av Google og publisert nå i sommer (juni). Det er fortsatt i beta-versjon, men det er ikke noe problem for de enkle spørringene jeg ønsker å benytte meg av. Biblioteket er er lag rundt en SQLite-database, som gjør det enklere å benytte seg av, man trenger for eksempel ikke å skrive noe særlig mye SQL, men det er også støtte for det. Minimum API for Room tror jeg er rundt 13, så det er ingen problemer med å bruke det i prosjektet.

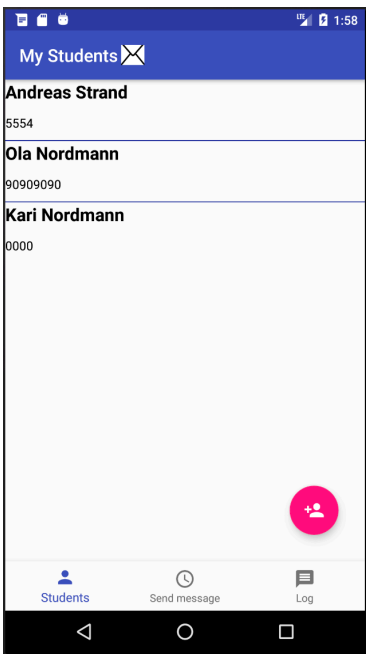
Det andre punktet er at jeg har lagt på litt ekstra når det kommer til den ukentlige meldingen. Jeg har lagt til støtte for å kunne velge selv hvor ofte meldingen skal sendes (daglig, ukentlig og månedlig). Dette kan gjøres på så mange meldinger brukeren ønsker, istedenfor å kun ha mulighet til å sende én ukentlig melding.

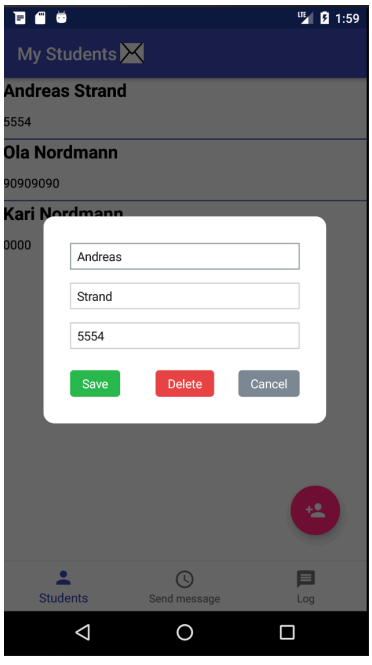
Aktiviteter

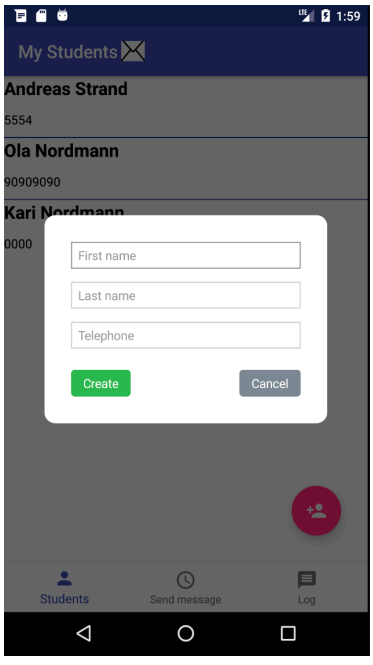
Applikasjonen består av tre aktiviteter, StudentActivity, AutoMessageActivity og MessageActivity. Alle aktivitetene arver fra en klasse 'BaseActivity' som igjen arver fra Activity. BaseActivity er laget for å kunne gjenbruke kode i aktivitetene.

StudentActivity

Denne aktiviteten er hvor man kan opprette, slette og endre studenter.





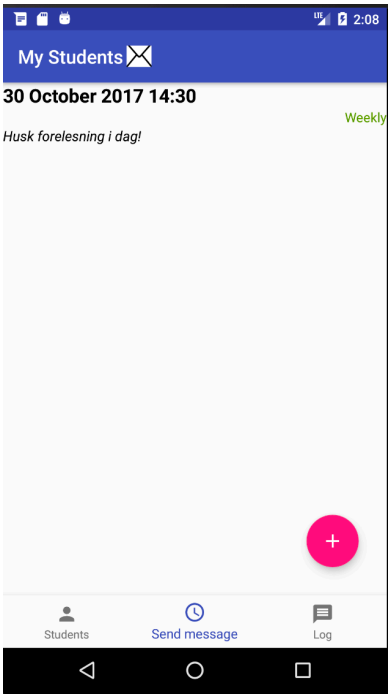


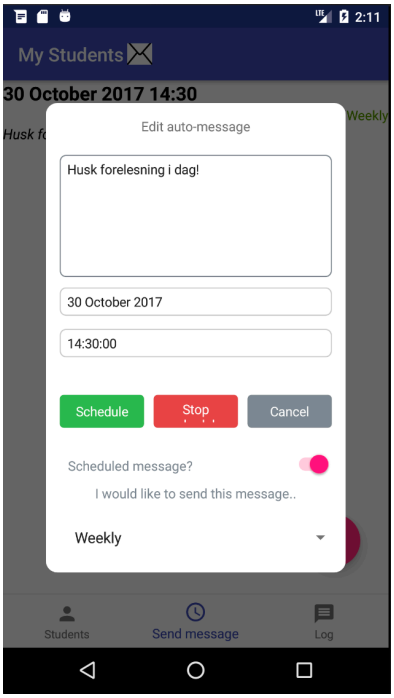
Aktiviteten består av en liste av alle studenter som er registrert i applikasjonen. Det er lagt til en 'FloatingActionButton' for å følge 'standard' liste-apper i Android. Trykker man på den så åpnes et DialogFragment hvor man kan skrive inn navn og telefonnummer til en student. Trykker man på en student i lista, får man mulighet til å endre.

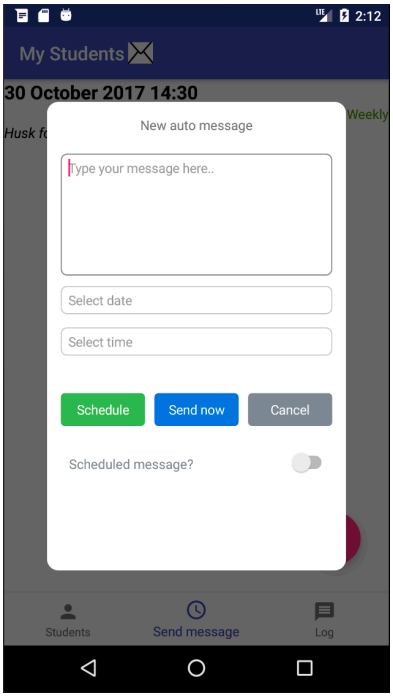
Her har jeg gjort et fargevalg på knappene. 'Save' knappen er grønn for å vise en 'positiv' handling. 'Delete' knappen er rød for å vise en 'negativ' handling som vil ha konsekvenser, men 'Cancel' knappen - som man trykker på for å lukke vinduet - er grå fordi å indikere til brukeren at dette er en nøytral handling.

AutoMessageActivity

Dette er aktiviteten som tar for seg all utsending av meldinger. Den viser også hvilke meldinger som er satt som 'auto', altså de som skal sendes på faste tidspunkt.







Aktiviteten åpner her også et DialogFragment om man trykker på dens 'FloatingActionButton'. Her kan man endre eller opprette en 'scheduled' melding. Man velger hvilken dato og hvilket tidspunkt man ønsker å sende meldingen på. Man har også mulighet til å velge om dette skal være en 'scheduled' melding, som her betyr at denne skal sendes i en fast rutine. Dette kan være daglig, ukentlig eller månedlig.

I aktiviteten så kommer det opp alle melding som ikke er sent enda, men de som venter på å bli sent. Det står hvilken dato de skal sendes på, samt om den skal fortsette å sendes i en fast rutine fremover. Så fort en melding er sent, fjernes den fra listen og eventuelt legges det til en ny en om meldingen skal sendes i en fast rutine.

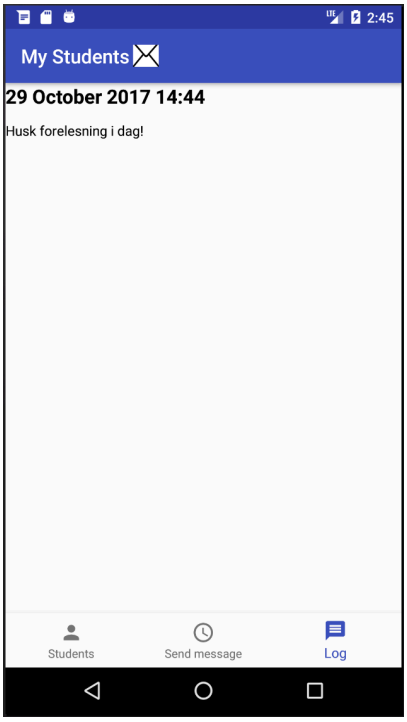
Om man ønsker å stoppe utsendelse av en melding, trykker man på 'Stop' knappen og meldingen vil da slettes, og ikke sendes ut.

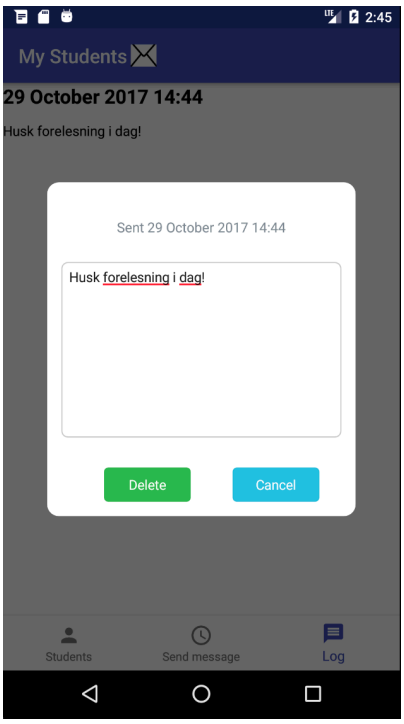
Når man oppretter en ny melding har man også mulighet til å sende meldingen med en gang. Det ser man på bildet helt til høyre, knappen som er markert blå.
Fargevalg på disse knappene gjenspeiler det som ble sagt for forrige aktivitet. ‘Schedule’ knappen her er grønn for å indikere en positiv hendelse. ‘Cancel’ knappen er grå for at den skal fremstå som nøytral. ‘Stop’ knappen er rød for å forsøke å gi brukeren en advarsel før den blir trykket på, fordi den handlingen vil ha en konsekvens ved at meldingen blir slettet.

Når man velger å opprette en ny melding så har man to knapper man primært kan trykke på, eller to ‘suksess’-knapper kan man også si. ‘Schedule’ trykkes på for å sende melding på et tidspunkt, og ‘Send Now’ vil sende meldingen med en gang. Da begge disse knappene henviser til en funksjon som kan sees på som en ‘positiv’ handling har de fått fargen grønn og blå. Begge disse fargene representerer en positiv følelse for brukeren og derfor er de fargene brukt. Alternativet hadde vært å gi begge knappene samme grønn-farge, men det ønsket jeg ikke, da jeg syntes at det er viktig å skille på funksjonaliteten for brukeren.

MessageActivity

Dette er den siste aktiviteten i applikasjonen. Den inneholder kun en liste av alle meldinger som er blitt sendt og hvilket tidspunkt de er sendt på.





Her inneholder også aktiviteten et DialogFragment som kommer opp etter man har trykket på en melding. Det fragmentet viser bare informasjon om meldingen i et litt større bilde. Her ser man når meldingen er sendt og melding som ble sendt.

Meldinger

Alle meldinger som er lagret i applikasjonen lagres i databasen. Meldinger som er satt til et tidspunkt sendes også når applikasjonen er skrudd av. Om man skrur av telefonen for så å skru den på igjen, vil en BroadcastReceiver sette alarmene på nytt igjen.

ContentProvider

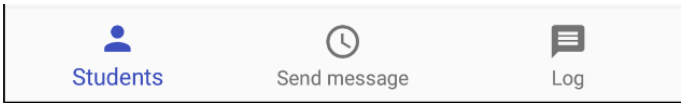
Å implementere ContentProvider var en liten utfordring for meg siden jeg hadde implementert en litt annen type database. Room Persistence Library inneholder heller ikke direkte støtte for ContentProvider, så løsningen ble å kode litt mer manuelt og følge et Github repository som Google har lagt ut. Linken til implementasjonen av Google sin ContentProvider med Room finnes her:

https://github.com/googlesamples/android-architecture-components/blob/master/PersistenceContentProviderSample/app/src/main/java/com/example/android/contentprovidersample/provider/SampleContentProvider.java

Jeg har også laget enhets-tester for ContentProvideren for å vise at den fungerer som den skal. Kjøres alle testene vil du se at en melding legges inn i Loggen med teksten «TEST CONTENTPROVIDER».

Design-valg

For det første så har jeg valgt å bruke en ActionBar nederst på skjermen for å enklest mulig kunne skille mellom aktiviteten for brukeren. Hver knapp på denne Action-baren viser til en aktivitet. Dette er implementert i en metode i BaseActivity for å gjenbruke kode, da dette er funksjonalitet som er felles for alle aktivitetene.



Jeg også gjort litt om på toolbaren til applikasjonen, og lagt til et bilde for at brukeren skal kunne gjenkjenne at det er denne applikasjonen brukeren er i.

Når det kommer til fargevalg i applikasjonen så har jeg gjort en del mer research enn jeg gjorde i forrige oppgave. Jeg fant denne artikkelen veldig informativ om hvordan man skal gi en bruker best og enklest mulig opplevelse ved bruk av farger på forskjellige knapper i en applikasjon:

https://uxplanet.org/best-practices-for-buttons-b7048479d440

Der står det litt mer i dybde om hvordan man velger farger for knapper som representerer forskjellige handlinger. Dette er også ressursen jeg har holdt meg til når jeg for eksempel har valgt å sette grønn og blå farge på knappene i fragmentet hvor man sender meldinger.

Notis

Det har vært et par tilfeller mens jeg har utviklet at prosjektet ikke gjenkjenner noen klasser i prosjektet. Det som må gjøres da er å kjøre en gradle-clean for så å kjøre en gradle-build.

Om du ønsker å teste å schedule en melding i appen, for så å lukke den og se om den blir sendt, må du passe på at appen ikke kjøres i debug. Da vil ikke meldingen sendes etter appen er lukket. Da må heller appen åpnes fra menyen på android-emulatoren for så å teste scheduling av meldinger. Det er nok en feil i emulatoren som dette skyldes.