Bacheloroppgave

IBE600 IT og digitalisering

Utplassering som RPA utvikler i Hemit

Anne Mari Farstad

Totalt antall sider inkludert forsiden: 54

Molde, 26.05.2023

Høgskolen i Molde
Vitenskapelig høgskole i logistikk

Obligatorisk egenerklæring/gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

Du/dere fyller ut erklæringen ved å klikke i ruten til høyre for den enkelte del 1-6:		
1.	**Jeg/vi erklærer herved at min/vår besvarelse er mitt/vårt eget arbeid, og at jeg/vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.**	☒
2.	**Jeg/vi erklærer videre at denne besvarelsen:** • ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • ikke refererer til andres arbeid uten at det er oppgitt. • ikke refererer til eget tidligere arbeid uten at det er oppgitt. • har alle referansene oppgitt i litteraturlisten. • ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	☒
3.	**Jeg/vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§16 og 36.**	☒
4.	**Jeg/vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert, jf. høgskolens regler og konsekvenser for fusk og plagiat**	☒
5.	**Jeg/vi er kjent med at høgskolen vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk**	☒
6.	**Jeg/vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider**	☒

Personvern

Personopplysningsloven

Forskningsprosjekt som innebærer behandling av personopplysninger iht. Personopplysningsloven skal meldes til Sikt for vurdering.

Har oppgaven vært vurdert av Sikt? ☐**ja** ☒**nei**

- Hvis ja:

Referansenummer:

- Hvis nei:

Jeg/vi erklærer at oppgaven ikke omfattes av Personopplysningsloven: ☒

Helseforskningsloven

Dersom prosjektet faller inn under Helseforskningsloven, skal det også søkes om forhåndsgodkjenning fra Regionale komiteer for medisinsk og helsefaglig forskningsetikk, REK, i din region.

Har oppgaven vært til behandling hos REK? ☐**ja** ☒**nei**

- Hvis ja:

Referansenummer:

Publiseringsavtale

Studiepoeng: 15

Veileder: Bjørn Jæger

Fullmakt til elektronisk publisering av oppgaven

Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).

Alle oppgaver som fyller kriteriene vil bli registrert og publisert i Brage HiM med forfatter(ne)s godkjennelse.

Oppgaver som er unntatt offentlighet eller båndlagt vil ikke bli publisert.

Jeg/vi gir herved Høgskolen i Molde en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	☒**ja**	☐**nei**
Er oppgaven båndlagt (konfidensiell)?	☐**ja**	☒**nei**
(Båndleggingsavtale må fylles ut)		
- Hvis ja:		
Kan oppgaven publiseres når båndleggingsperioden er over?	☐**ja**	☐**nei**

Dato: 25.05.23

Forord

Denne oppgaven er en del av kurset IBE600 Bacheloroppgave i IT og Digitalisering ved Høgskolen i Molde, og ble gjennomført som en utplassering i Hemit HF. I oppgaven beskriver jeg prosjektet jeg utførte for Hemit i utplasseringsperioden.

Jeg tok kontakt med Hemit HF høsten 2022, og fikk der muligheten til å ta utplasseringen som en del av RPA-teamet. Gjennom utplasseringen har jeg fått lære å ta i bruk UiPath verktøyet, og fått gjennomføre en RPA prosess i Helse Midt. Jeg har fått tett oppfølging av seksjonsleder Eirik Wolfenstein i Hemit, som hele veien har sørget for at jeg har blitt involvert i prosjekter, og hatt alt av nødvendig utstyr og tilganger. I utvikling av RPA prosjektet har John Michael Obligar ledet meg gjennom arbeidene, og gitt meg god opplæring og støtte underveis. Både Eirik, John M. og øvrige ansatte i RPA-teamet har vært uvurderlige for at jeg skulle gjennomføre dette prosjektet. Jeg skylder de en stor takk.

I tillegg vil jeg også takke veileder Bjørn Jæger og studieleder Ketil Danielsen for gode tilbakemeldinger og råd underveis.

Sammendrag

Som en del av kurset IBE600 Bacheloroppgave våren 2023 gjennomførte jeg utplassering i RPA-teamet i Hemit HF i Molde. Hemit HF er tjenesteleverandør og bindeledd til alle sykehus og helseforetak i Helse Midt Norge innen teknologi og IKT tjenester (Hemit HF, 2023). Robotisert Prosessautomasjon(RPA) er et verktøy som kan brukes til å automatisere tidkrevende og repeterende prosesser. I Hemit benyttes blant annet UiPath som automasjonsplattform for RPA. Etter å ha gjennomført flere prosessvurderinger fikk jeg mulighet til å utvikle en unattended RPA-prosess i UiPath, som skal vaske telefonnumrene til respondenter i CheckWare opp mot pasientregisteret(PAS).

Denne oppgaven beskriver arbeidene fra utvikling og testing, helt frem til og med første produksjonssetting. Første produksjonssetting ble utført for to avdelinger i Helse Møre og Romsdal (HMR). Gevinsten av denne automatiseringen er innsparing av tid for helsepersonell ved utsending av kartleggingsskjema, reduksjon av risiko for at kartleggingsskjema sendes til feil pasient, samt at den fjerner kjedelig repeterende arbeid. Alle arbeidene ble utført under veiledning og oppfølging av senior utvikler J.M. Obligar.

Innhold

1.0 Introduksjon

Denne oppgaven er en sluttrapport etter utplassering i faget IBE600 Bacheloroppgave ved Høgskolen i Molde. Jeg valgte å benytte meg av anledningen til å få praktisk erfaring, og muligheten til å bli bedre kjent med arbeidslivet i Molde.

Høsten-22 tok jeg kontakt med Hemit, og var så heldig å få tilbud om å få gjennomføre utplasseringen som en del av RPA-teamet deres. Denne oppgaven skal i størst mulig grad gjenspeile den kompetansen jeg opparbeidet meg i utplasseringen, samt RPA-prosjektet som jeg fikk gleden av å utvikle med god støtte i senior utvikler J. M. Obligar.

2.0 Robotisert prosessautomasjon(RPA)

Robotisert Prosessautomasjon(RPA) er en software teknologi som ved å følge forhåndsdefinerte regler kan automatisere oppgaver og prosesser ved å imitere en menneskelig handling. Fordelen med denne typen teknologi er at den ikke krever at det gjøres noen større endringer på de programmene eller applikasjonene som brukes, og kan gå under kategorien lettvekts IT-løsning (Farsund Ulltang & Knappskog, 2022). Roboten tilordnes samme brukerrettighetene som den ansatte som utfører oppgavene. RPA er dermed en rimelig og rask løsning, som kan erstatte eller brukes i påvente av større systemendringer, eller integrasjoner.

Typiske arbeidsoppgaver som egner seg for klassisk RPA er repeterbare prosesser som utføres i et større omfang, på daglig eller ukentlig basis. Regelbaserte prosesser der det forekommer lite eller ingen behov for skjønn. Ethvert tilfelle som ikke kan utføres basert på regler må sendes inn til kontroll/evaluering. Dersom man tar i bruk kunstig intelligens kan RPA også utføre disse oppgavene og få et større bruksområde. Et annet kjennetegn på klassisk oppgave for RPA at det kan regnes som stabile prosesser der det ikke forventes vesentlige endringer i applikasjoner eller prosesser den nærmeste tiden.

For å vurdere egnethet til roboten i forkant kan lages et regnskap som sannsynliggjør innsparte årsverk ved å implementere roboten. Dette kan igjen sammenlignes med estimert tid for RPA-utviklingen. Differansen mellom disse vil være netto besparte timeverk/årsverk.

2.1 UiPath som RPA verktøy

UiPath er et RPA-verktøy levert av software selskapet UiPath. Selskapet kan med sikkerhet sies å være av de ledende leverandørene av RPA-programvare. I 2020 var UiPath eneste RPA leverandøren listet opp i Forbes AI 50 ranking i 2020, over de mest lovende selskapene innenfor kunstig intelligens (Forbes, 2020). UiPath tilbyr en automasjons plattform bestående av tre kjernekomponenter som er; Studio, Orchestrator og Robot (UiPath, 2023).

UiPath er et såkalt «lowcode»- RPA verktøy (UiPath, 2023). Det finnes tre ulike Studio automasjonsverktøy: Studio Web, StudioX og Studio. StudioX er laget som et verktøy som er enklere å bruke for de med liten eller ingen programmeringsbakgrunn (UiPath, 2023). Studio Web ble tilgjengelig i Communty versjon og lansert for enterprise i 2023. Dette verktøyet anbefales som et nybegynnerverktøy, og fungerer som en kryssplattform mellom online business apps og tjenester (UiPath, 2023). Studio versjonen retter seg mot de RPA-utviklere og de brukerne som har programmeringsbakgrunn. I Studio tilbys verktøy for å utvikle mer komplekse attended, unattended og test-automasjoner (UiPath, 2023). Default-programmeringsspråk er VB.NET og noe kjennskap til dette er svært nyttig da det meste av dokumentasjon og guider vises .NET (Anders Jensen, 2023). I 2020 ble imidlertid ble også C# introdusert.

I denne oppgaven er UiPath brukt som verktøy for å utvikle RPA prosessen. Første del av utplasseringen brukte jeg mest tid på å bli kjent og lære om UiPath som verktøy. Jeg fullførte RPA Developer Foundation kurs som ett av kursene UiPath tilbyr i Academy, samt at jeg har fått delta på interne kurs og samlinger.

3.0 RPA i Hemit

Hemit leverer IT tjenester til sykehusene og sykehusapoteket i Helse Midt. Hemit etablerte et RPA team i 2020. Ved utgangen av 2022 hadde teamet iverksatt 18 robotiserte prosesser, og spart 54 årsverk (Hemit HF, 2023). Helsesektorens utfordringer med mangel på helsepersonell og flere eldre, stiller større krav til effektivitet fremover. RPA kan tilby IT løsninger som avlaster helsepersonellet slik at de kan bruke tiden på kjerneaktiviteten.

Helse- og omsorgsministeren fikk februar i år overlevert Helsepersonellkommisjonen som utredning, samtidig som den ble lagt ut på høring. I rapporten står det at ”Helsepersonellkommisjonen foreslår at helse- og omsorgstjenestene anvender prinsippet om at oppgaver der personellet ikke har direkte kontakt med pasienter, brukere eller pårørende, automatiseres i størst mulig grad” (Departementenes sikkerhets- og serviceorganisasjon, 2023, s. 20). Utfordringen er å forstå arbeidsoppgavene til personellet tilstrekkelig, slik at det kan skapes gode tekniske verktøy, som i sum frigjør tid hos helsepersonellet. Ut fra prinsippet til helsepersonellkommisjonen er det nesten ingen grenser for hvor mange oppgaver som kan automatiseres ved hjelp av RPA.

I Hemit kan aktuelle RPA kandidater meldes inn av alle ansatte i det regionale helseforetaket Helse Midt. Deretter gjøres en prosessvurdering av kandidaten for å vurdere egnethet. For å nå ut til hele organisasjonen og finne de gode kandidatene er det også viktig å ikke vente på at kandidatene blir meldt inn, men også å promotere og spre informasjon om hvilke muligheter og lettelser RPA kan bidra med ut til helsepersonell og administrasjon.

4.0 Prosessvurdering

Hver kandidat som meldes inn settes først opp til prosessvurdering. I Hemit gjennomføres prosessvurderinger slik at innmelder eller de som er kjent med prosessen viser den manuelle utførelsen av prosessen. Deretter fylles det ut et prosessvurderingsskjema som til slutt gir en samlet score over hvor egnet kandidaten er for automatisering. Ut fra verdiene som fylles inn får man også et forventet anslag over hvor stor innsparingen er i tid/timer, og tidsbruk for utvikling.

Utfylte skjema etter prosessvurdering samles i en oversikt med rangering der de mest egnede prosessene havner øverst. Dette danner en prioriteringsliste for alle vurderte kandidater ut fra scoring. Kandidater som ikke har like høy score på innsparing av tid, men likevel gir en stor gevinst kvalitetsmessig eller andre hastesaker kan løftes opp på listen.

4.1 Prosessvurdering CheckWare Respondent Tlf.Vask

Prosesseier hadde opprinnelig meldt inn denne kandidaten som opprettelse av respondent i CheckWare. Det var tidligere utført en prosessvurdering av kandidaten av andre i teamet. Etter et møte med prosesseier 10.03 ble det bestemt å dele denne kandidaten opp i tre RPA-prosesser, hvorav første del skal håndtere vask av respondenters telefonnumre i CheckWare opp mot PAS. Det ble min oppgave å gjennomføre utvikling av denne prosessen med John Michael Obligar som senior veileder.

Ut fra svarene fra prosesseier i møtet den 10.mars, ble alle punktene for denne delen av prosessen fylt ut i skjemaet, se samlet tabell i VEDLEGG 1. Volumet i dag er ca 30 i uken, mens på sikt er målet ca 500 i uken. Estimert håndteringstid for innlogging og kontroll av telefonnummer er ca 1 minutt. Innsparing ved ca 500 i uken vil tilsvare ca 8,3 timer eller ett dagsverk i uken. Dette gjelder for kirurgisk ortopedisk poliklinikk i Helse Møre og Romsdal.

Målet på sikt er at denne automatiseringen skal gjelde for alle avdelinger i alle helseforetak i Helse Midt. I dag har HMR, HNT og St.Olavs samlet sett et volum på ca 2250 i måneden, ca 500 i uken. Ifølge prosesseier forventes dette tallet å øke fremover. I prosessvurderingsskjema settes volumet til 500 i uken.

Kort oppsummert er dette en digital, strukturert prosess der det ikke forventes endringer prosessen og dens tilhørende applikasjoner, programmeringsgrensesnitt og lignende de nærmeste 6 månedene. Etter innføring av Helseplattformen vil prosessen måtte endres til å gjøre kontroll opp mot det nye pasientregisteret.

Utregning av Implementeringstiltak og Fordel/Egnehet gjøres ut fra en formel basert på innsatte verdier i kolonnen over. Estimat av innspart tid ved automatisering opp mot tid det tar å utvikle RPA prosessen viser at en automatisering vil gi innsparing veldig raskt, allerede etter noen måneder, se Figur 1.

Figur 1 Utklipp fra Vedlegg 1, som viser gjennomførbarhet/Score for CheckWare Respondent Tlf.Vask

Ut fra totalscoren på prosessvurderingsskjemaet fremstår denne kandidaten som en godt egnet. Det er tidligere utviklet en RPA-prosess i CheckWare som gjør at tilgangene til et Test-miljø og Prod miljø allerede eksisterer. Samt at spørringer via API mot PAS er tilgjengelig.

Målene ved automatisering er innsparing av tid for personell som benytter CheckWare. Når det skal sendes ut eksempelvis et kartleggingsskjema tar noen minutter hver gang for å kontrollere rett telefonnummer. I tillegg ble det nevnt av helsesekretær i møtet at dette var ansett som et irriterende moment. Slike poenger er viktige, og er et av målene ved RPA er å fjerne kjedelige oppgaver fra arbeidshverdagen til de ansatte.

Faren for menneskelige feil ved manuell innføring av telefonnummer fjernes også ved automatisering. Dersom et kartleggingsskjema ikke blir sendt ut til rett nummer, må dette fylles ut når pasienten ankommer oppsatt time, og i verste fall må timen utsettes. Dette medfører forsinkelser i behandling og merarbeid for helsepersonellet, som igjen genererer en stor ekstra kostnad.

4.2 Gjennomgang av dagens prosess: As-Is

For hver kandidat man velger å gå videre med utarbeides et SDD-dokument (Solution Design Document) i Microsoft Visio. Dette dokumentet brukes til slutt som dokumentasjon på utvelgelsesprosessen. SDD filen skal oppdateres hver gang det er endring i design på løsning.

For kandidaten CheckWare ble prosessens AS IS tegnet opp, den gir en visuell oversikt over den manuelle prosessen slik den gjennomføres i dag.
Prosessen gjennomføres i dag på den måten at når en pasient føres opp til time eller settes på venteliste, skal det et visst antall uker i forkant av time sendes ut et kartleggingsskjema. Her skal pasienten svare ut ulike spørsmål om egen helse, som f.eks høyt blodtrykk, graviditet, MRSA osv. Når skjemaet er sendt tilbake kan CheckWare reagere på ulike skåringer i svarskjemaet, slik at de som svarer ja der de bør svare nei blir sortert ut til spesiell oppfølging (Hemit, 2023).

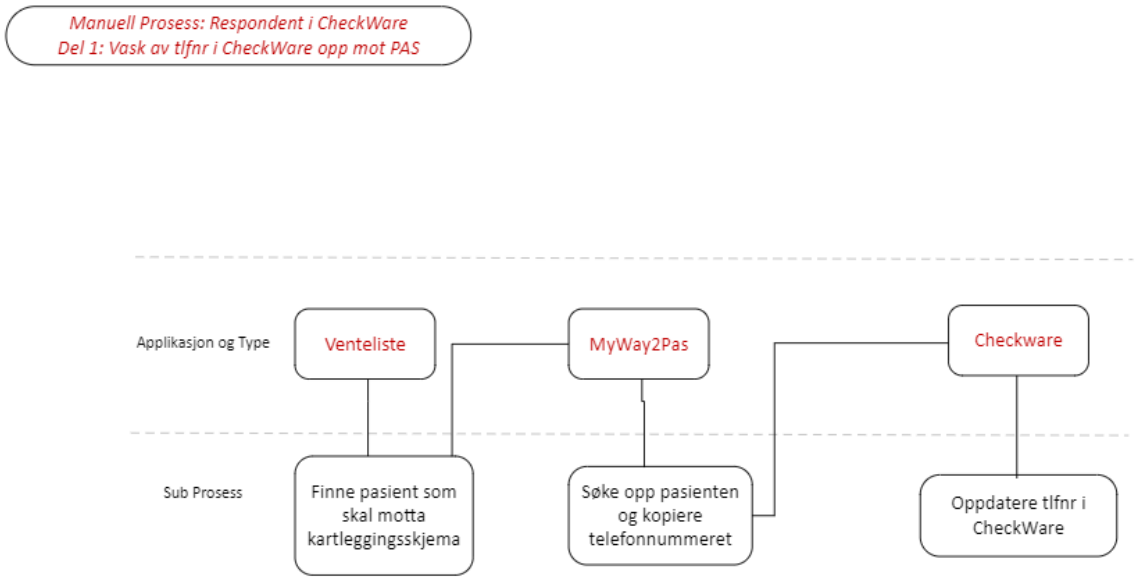
I CheckWare kalles pasienten for respondent. For denne prosessen som gjelder oppdatering av telefonnummer må sekretær gjøre følgende i forkant av utsendelse:

- Finne pasienter på venteliste som skal motta kartleggingsskjema

- Skrive inn fødselsnummer, og gjøre søk mot PAS
- Kopiere mobilnummeret fra MyWay2PAS
- Navigere frem til respondent i CheckWare og oppdatere tlf.nr

PAS er koblet opp mot Kontakt- og reservasjonsregisteret. Det er Digitaliseringsdirektoratet som er behandlingsansvarlig for Kontakt- og reservasjonsregisteret (Digitaliseringdirektoratet, 2023).

Jeg opprettet ny prosjektmappe og tegnet As-Is prosessen i Visio ut fra den informasjonen vi fikk fra møtet 10.03, se Figur 2.



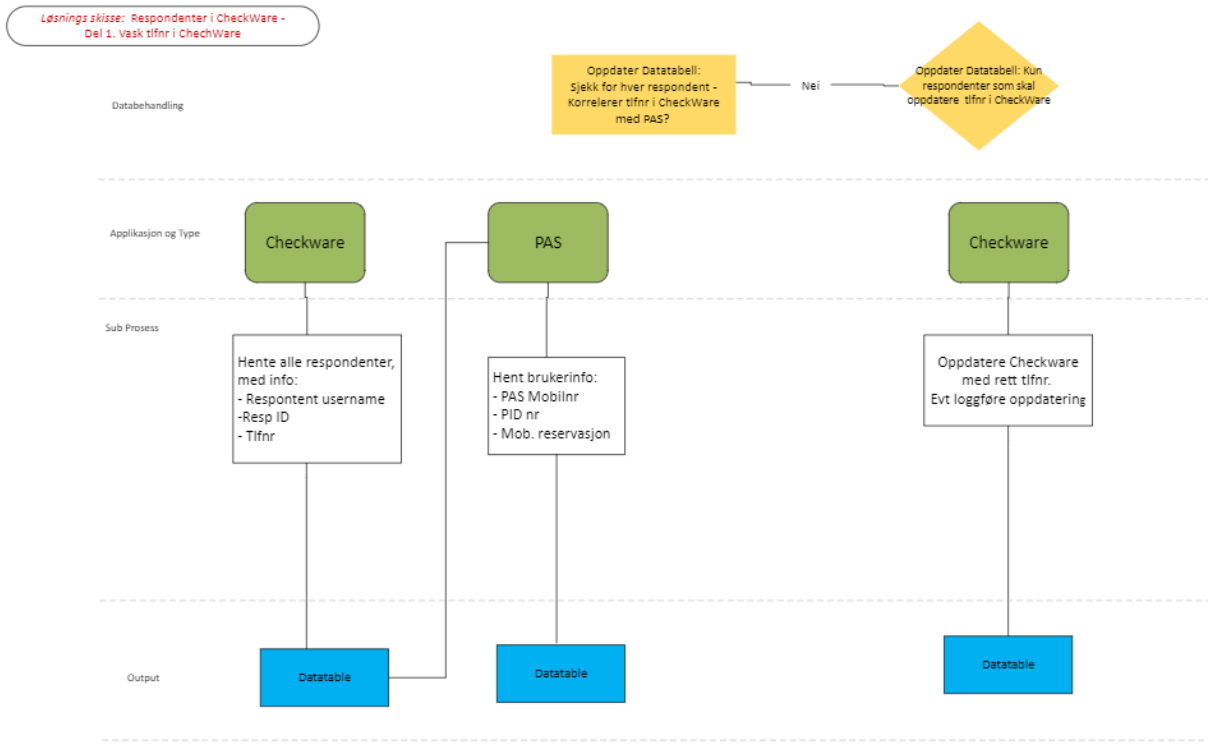
Figur 2 CheckWare - Tegning av As-Is prosessen, slik den utføres i dag

5.0 Design av løsning: To-Be

Videre startet jeg på løsningsskissen av prosessen(To-Be). Denne presenterer visuelt den planlagte logikken til prosessen, se Figur 3. Denne ble revidert et par runder etter samtaler med veileder og kontrollør John. Det er viktig at logikken kommer tydelig frem ikke bare for utvikleren, men også for andre som eventuelt skal ta over arbeidene.

Etter å ha gjort meg litt kjent med CheckWare sitt testmiljø fant jeg en måte å eksportere ut alle respondenter i en csv-fil. Deretter lages det en datatabell av alle respondenter. Neste steg blir å søke opp hver pasient via API-kall og hente ned telefonnummer og annen nødvendig informasjon og legge dette til på hver respondent. Til slutt må det lages en logikk som henter ut de respondentene der telefonnumrene ikke

er like, og setter disse i en egen ny tabell. Alle respondenter i denne tabellen skal få nummeret oppdatert i CheckWare. Det er viktig at det logges hvem som er oppdatert. Dette for å synliggjøre hvor mye jobb roboten gjør for å blant annet at det kan regnes ut hvor stor innsparing den gir i etterkant.



Figur 3 Løsningsskisse – To be: CheckWare Respondenter TlfVask

6.0 Scenarioer og robotregler

Før man kan begynne utvikling er det viktig å danne et bilde over de ulike scenarioene roboten kan komme ovenfor, og hvordan den skal agere i de ulike tilfellene. Jeg satt opp scenario-listen basert på det jeg anså som åpenbare løp ut fra løsningsskissen i Figur 3. Scenariolisten ble deretter gjennomgått med prosesseier, for å inkludere deres ønsker i hvordan roboten skal aksjonere i de ulike tilfellene, og om det var eventuelt andre scenarier de mente burde tas med. I denne prosessen landet vi følgende scenarier:

<u>Scenario 1:</u>

Respondent i CheckWare finnes ikke i PAS. Dette er en forretningsfeil/Business Exception, og skal logges som en advarsel. Må gjøres en manuell behandling av saken.

<u>Scenario 2:</u>

Respondent har ikke telefonnummer i CW. Roboten må godta blankt nummer og oppdatere mot nummer i PAS. Ingen logging.

Scenario 3:

Respondent har telefonnummer i CW, men ikke i PAS. Roboten logger feilen som forretningsfeil med informasjon om hva som er funnet. Saken må til manuell behandling.

Scenario 4:

Ulikt telefonnummer i CW og PAS. Roboten skal da sende respondent til kø i Orchestrator for oppdatering av nummer. Logger funnet og at respondenten sendes til oppdatering.

Scenario 5:

I PAS står mobile reservation til True. Roboten skal dermed ikke foreta noe, men sende funnet til logg som en forretningsfeil. Saken må til manuell behandling.

Scenario 6:

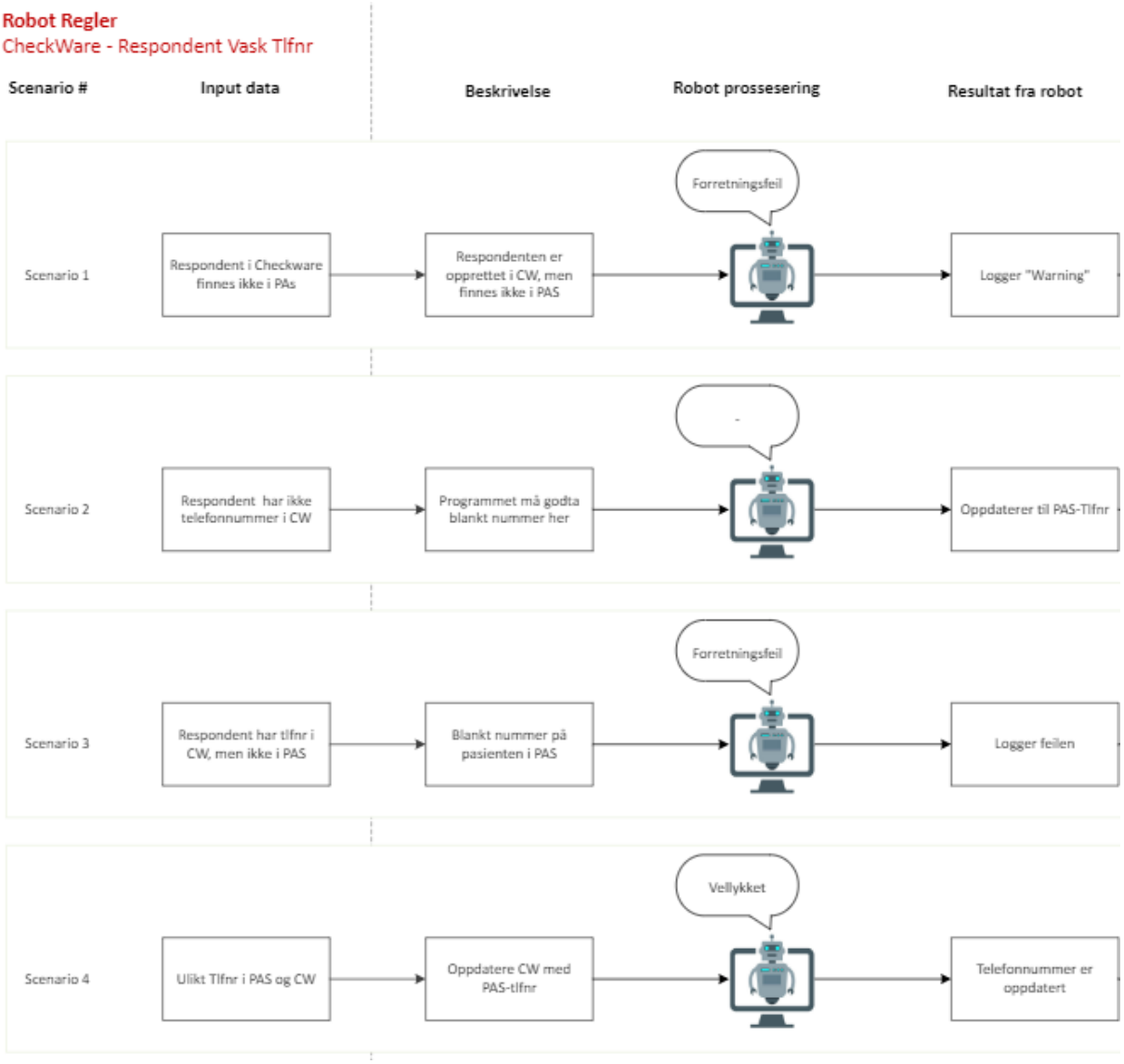
Samme telefonnummer er registrert i CW og PAS. Roboten skal ikke foreta seg noe, men sender en logg på dataene.

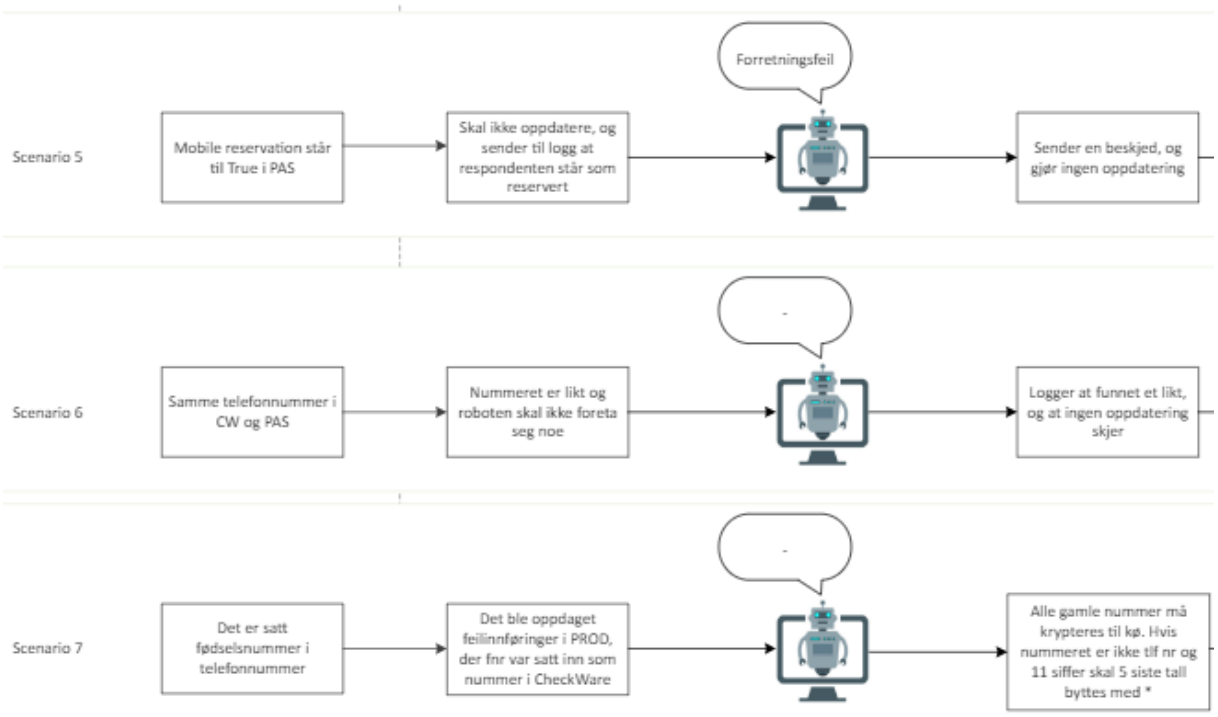
Scenario 7:

Etter produksjonssetting gjorde vi oppdagelsen av at det stedvis var ført inn fødselsnummer som telefonnummer. Dette scenarioet ble derfor lagt inn i dokumentasjonen etter første produksjonssetting. Fødselsnummer i seg selv regnes ikke som sensitiv eller taushetsbelagt informasjon, men det anbefales at det krypteres ved usikret sending (Datatilsynet, 2023). Siden dette sendes sammen med annen informasjon om respondent som navn og nytt nummer så velger vi derfor å kryptere respondent-telefonnummer som sendes til kø i Dispatcher. Når den videre dekrypteres i Performer, byttes de siste 5 tallene med *.

Robot Regler

CheckWare - Respondent Vask Tlfnr



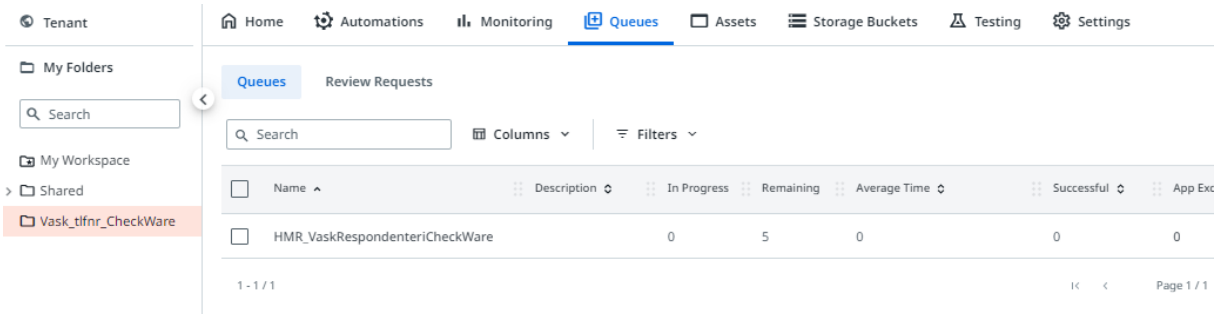


Figur 4 Utklipp av Scenarioer og robotregler

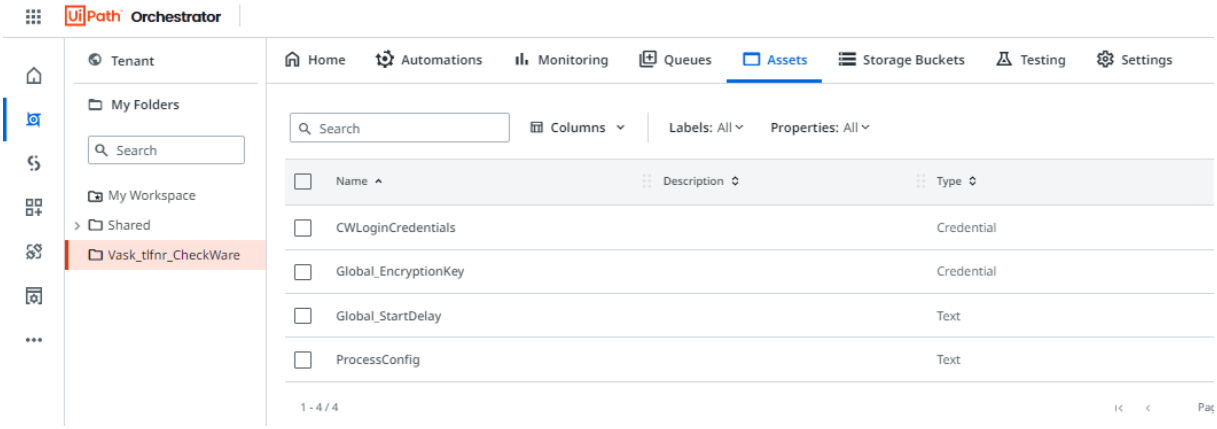
7.0 Orchestrator

I Orchestrator kan automasjonsprosessen styres sentralisert. Fra UiPath Studio settes det opp en kobling til Orchestrator-webapplikasjon. Man kan publisere automasjonsprosessen fra Studio, slik at prosessen kan trigges i Orchestrator. I Orchestrator opprettes køen("Queues") som automasjonsprosessen kan plassere og hente elementer ut fra, se Figur 5. Kø-navnet er ”HMR_VaskRespondenteriCheckWare”. Denne prosessen skal kjøres unattended, som vil si at den kjøres automatisk uten menneskelig innvirkning. Dispatcher skal trigges ut fra gitte tidspunkt i døgnet, som angis i Orchestrator. Deretter skal Performer trigges av at Dispatcher har lagt en eller flere elementer til kø.

Neste fane er Assets, her plasseres variabler og eventuelle tekstfiler som trengs i automasjonen. Fordelen med å plassere mest mulig i Orchestrator er at prosessen i Studio kan programmeres mest mulig generisk.



Figur 5 Utklipp fra UiPath Orchestrator som viser opprettet kø til mappen Vask_tlfnr_CheckWare.



Figur 6 Utklipp fra UiPath Orchestrator som viser opprettede Assets til prosjektmappen.

Assets til dette prosjektet er

1. CWLoginCredentials: Inneholder credentials for innlogging i dev-miljøet i CheckWare. Her fylles inn brukernavn og passord.
2. Global_EncryptionKey: Dette er en standard asset som brukes når prosessen krever at deler av transaksjonen som skal sendes til kø krypteres.
3. Global_StartDelay: Denne er også standard, og er satt til 20 sekund. Dette er en empirisk verdi som gir VDI'en tid til å logge inn før prosessen starter opp.
4. ProcessConfig. Dette er standard asset som er skrevet som en JSON tekststreng med oppgitte keys and values som skal benyttes i prosessen.

8.0 UiPath Studio

UiPath Studio den delen av automasjonsplattformen til UiPath der utvikling og utgivelsen/deployment av software roboter foregår. I Studio er det visuelt grensesnitt der brukeren kan dra inn et stort spekter av ulike aktiviteter og koblinger opp mot andre applikasjoner og ulike datakilder.

I denne automasjonprosessen har jeg bygd opp hver workflow slik at den utfører ett delmål. I hver workflow er alle aktivtetene bygd opp inne i en Try Catch blokk. Denne sørger for en strukturert unntak/exception- og feilhåndtering. Alle aktiviterer kjøres i Try

blokken. Hvis unntak/exception skjer i Try, utføres aktivitetene i Catches blokken. Unntaket kan vises som en feilmelding av type forretningsfeil/Business Exception eller applikasjonsfeil/System Exception. Det kan også velges at exception skal ignoreres og la programmet kjøre videre. Finally blokken fylles med de aktivitetene som skal utføres om ingen unntak skjer, eller om feil skjer og blir fanget opp av Catches blokken, uten å bli kastet på nytt (Uipath Academy, 2023).

Behandling av unntak er viktig når automasjonsprosesser skal kjøres i produksjon fordi det forventes at de tekniske forholdene vil endres over tid. Ved å gjennomføre en god unntaksbehandling i prosessen er det enklere å fange opp hvor feilen oppstår og rette den.

9.0 Dispatcher

I UiPath er dispatcher den delen av et RPA rammeverk som initierer og gjennomfører første del av en automasjons prosess (Uipath Academy, 2023). Typisk oppgave for dispatcher er hente og lese data fra en kilde, som for eksempel excel-filer eller databaser, og bearbeide disse til et enkeltstående emne/item, som kan brukes videre av en robot. Dispatcher kan sende de bearbeidede elementene opp til en kø i orchestrator, slik at Performer komponenten av automasjonen kan hente hvert enkelt køelement og gjennomføre siste del, selve utførelsen av automasjonen.

En oversikt over hvordan Dispatcher-template er bygd opp i Hemit vises i Figur 7. Hver boks representerer en state i prosessen. Mellom hver state er det overganger/transision lines som styres ut fra hva verdien til variablene SystemException og BusinessException. Ved slutten av hver state settes verdiene til Nothing, for da har alt gått som det skal. Underveis i en state kan det kastes en Exception, og da stoppes løpet og det går rett ut til en overgang til annen state. Om SystemException settes til en verdi i Initalization, vil programmet følge overgangen "System Exception" videre til End Process state, se Figur 7.

Business Exception eller forretningsfeil er en feil som innebærer at de angitte reglene for prosessen ikke er fulgt for inputdata. I tilfellet med CheckWare er det funnet tre scenarioer som medfører forretningsfeil, Scenario 1,3 og 5. Disse tilfellene skal logges fordi de må til manuell behandling. Fra Dispatch Transaction Items state er det tre mulige

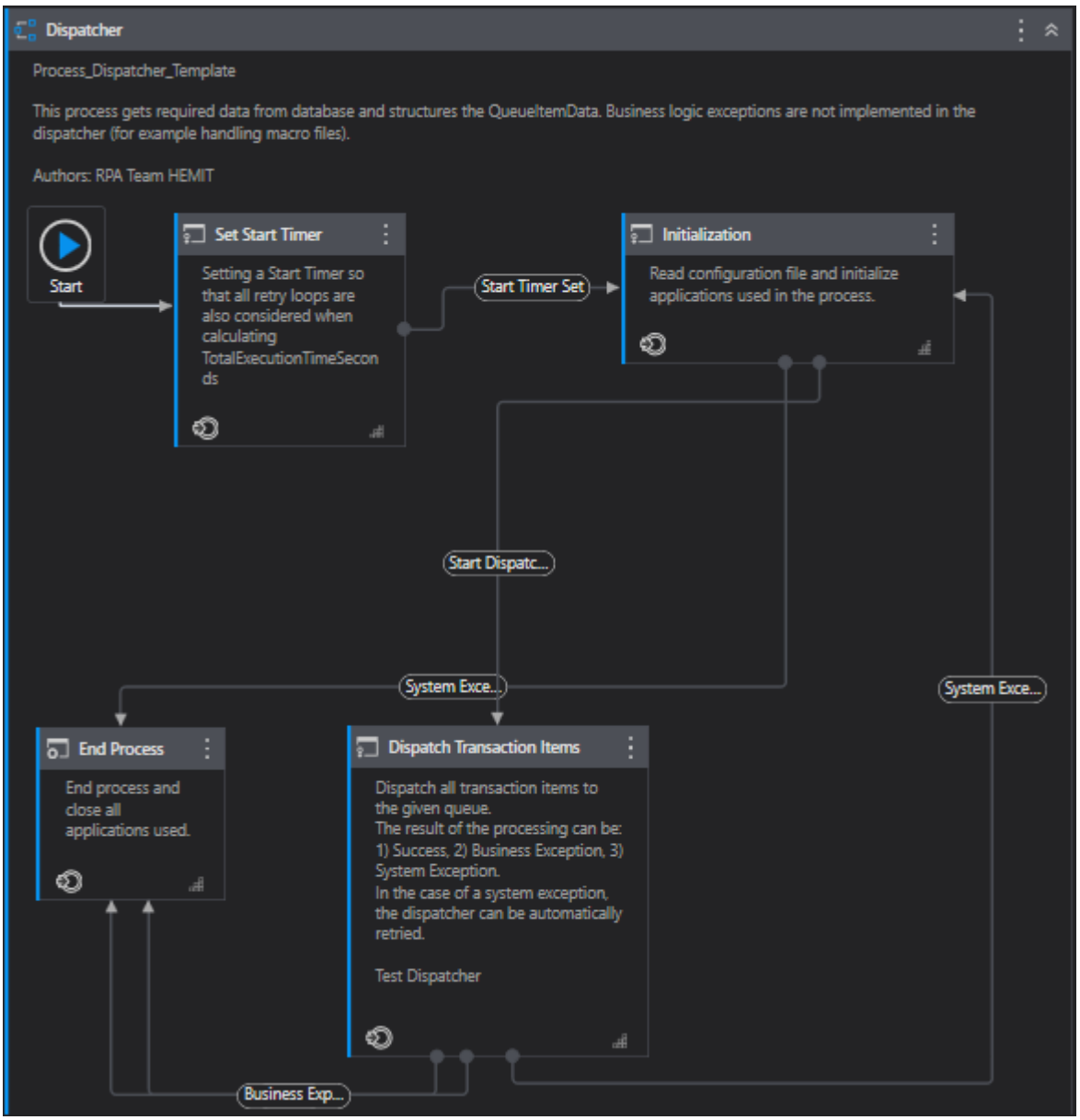
utfall; Success, Business Exception eller System Exception. Success overgang innebærer at både Business Exception og System Exception er satt til Nothing.

Variabler som styrer overganger:

1.) *SystemException (Variabeltype Exception)*
2.) *BusinessException (Variabeltype BusinessRuleException)*

Inn argumentene i Dispatcher:

1) *In_OrchestratorQueueName*
2) *In_TimeSpanAssetName*



Figur 7 Dispatcher Template i Hemit

Verdiene til disse argumentene settes i Orchestrator. In_ OrchestratorQueueName settes slik at Dispatcher vet hvor den skal føre opp køelementer. In_TimeSpanAssetName settes

til å være lik en asset som er oppgitt i Orchestrator, Global_StartDelay 20 sek, dette for at VDI'en der roboten skal kjøre bruker om lag 10-15 sekunder på å starte opp.

9.1 Set Start Timer

Når Dispatcher starter opp går den til første State som heter Set Start Timer. Her settes det først en eventuell Delay hvis det er angitt i In_TimeSpanAssetName. I tillegg starter den en klokke for å logge tidsbruk.

9.2 Initialization

Oppgaven til denne delen av dispatcher er å lese ProcessConfig-asset og gjøre den om til en dictionary, som brukes videre i omtrent alle workflows. Det er en fordel at alle verdier som skal hentes ut i prosessen ligger i denne Config dictionary. Deretter skal alle åpne browser vindu lukkes ned. Til slutt skal CheckWare webapplikasjon åpnes og det skal logges inn til hjem-siden. Det er viktig å bygge opp initialization på en slik måte at den kan gjenbrukes i Performer, som også har initialization-state som skal utføre de samme oppgavene.

I Initialization leses konfigurasjonsfilen (*ProcessConfig*) som er angitt som en Asset i Orchestrator, se Figur 5. Den gjøres om til en dictionary med keys and values med variabelnavnet *Config*. Alle verdier som skal hentes ut bør ligge her. Etter å ha opprettet *Config*, sørger en Workflow som kalles *KillAllProcesses* for at alle åpne vinduer lukkes før prosessen starter opp.

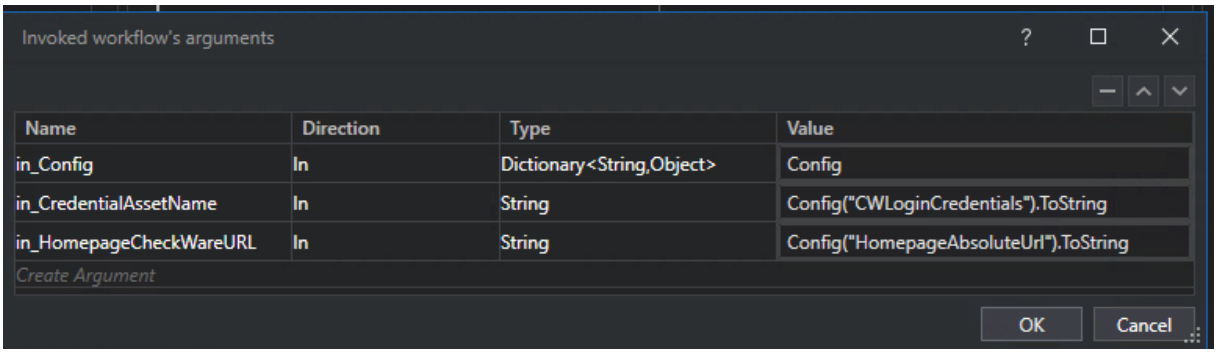
CheckWare webapplikasjon har ulik URL adresse for de ulike instansene eller helseforetakene. For å løse innlogging til rett URL for hver prosess lagde J. M. Obligar en Workflow som kalles BuildAbsoluteURL. I Config oppgis BaseURL, og ulike RelativeURL. Hver prosess trigges fra Orchestrator og er koblet til en kø. Køene navnsettes med Instansnavn først slik "HMR_VaskRespondenterCW". Instansnavnet hentes fra tilhørende kø, og legges inn i Config dictionary for prosessen slik:

```
Config("InstanceName") = in_OrchestratorQueueName.Split("_"c)(0).Trim.ToLower
```

Deretter bygges det opp BaseURL + Instansnavn + ulike RelativeURL for innlogging og navigering frem til de ulike respondentene.

9.2.1 Initalization - InitAllApplications-workflow

InitAllApplication-workflow er siste del av *Initalization*. Her er målet å logge inn og åpne opp alle applikasjoner som skal brukes i *Dispatch Transaction Items*.
Inn argumentene er vist i Figur 8. Alle argumentene er hentet fra Config dictionary. Ved å skrive det på måten vist i Value-kolonnen hentes verdien til oppgitt nøkkel ut som tekst.



Figur 8 Inn-argumenter til InitAllApplication Workflow

Det skal logges inn i CheckWare sin web-applikasjon, med angitt påloggingsinformasjon fra *CWLoginCredentials* i Assets, se Figur 5.

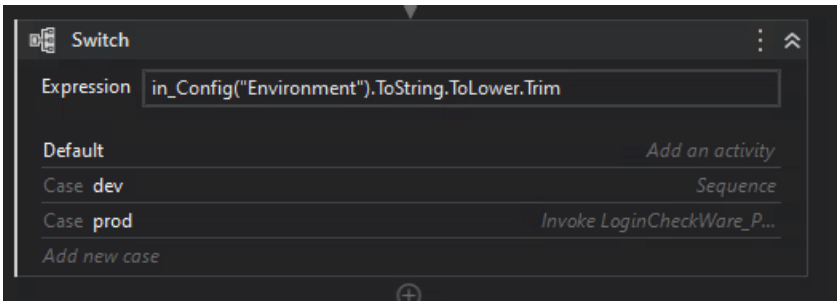
9.2.1.1 InitAllApplications

Denne prosessen starter med en standard Starting Up sekvens som setter verdi på variabelen *ThisWorkFlowName*, og setter *out_WorkflowException* til *Nothing*. Dersom prosessen skulle feile er det viktig å logge hvor langt prosessen er kommet, for å forenkle arbeidene med å finne og rette opp feilen. Helt til slutt logges det at denne workflowen har startet opp.

- ThisWorkflowName = CurrentJobInfo.Workflow + "xaml"
- out_WorkflowException = Nothing
- LogMessage: "ThisWorkflowName has started"

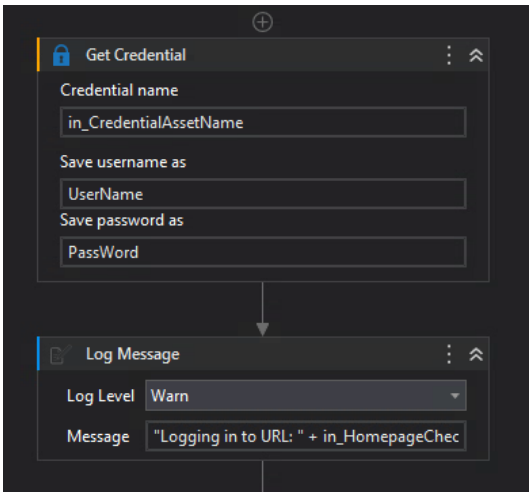
InitAllApplication workflow bygges spesielt for alle prosesser. Her vises hvordan jeg har bygd opp prosessen, med argumenter rundt valget av de ulike aktivitetene. Tips og innspill til dette har jeg fått fra veiledere, og fra intern opplæring og kurs.
I testmiljøet(*dev*) til CheckWare skjer innlogging ved å skrive inn brukernavn og passord. I *prod* er det enkeltpålogging (Single sign-on) ved brukeridentfikasjon der roboten får egen brukertilgang. Det er derfor nødvendig å bygge opp pålogging på to ulike måter. Dette gjøres ved å legge inn Switch-aktivitet, som gir ulike alternativer for om *Environment* i

Config er satt til *prod* eller *dev*, se Figur 9. I fortsettelsen vises case for innlogging i testmiljø/dev.



Figur 9 Utklipp av Switch aktivitet

I Asset er innloggingsinformasjonen lagret som brukernavn og passord. Disse hentes ut ved Get Credential-aktiviteten, og UserName og PassWord blir tilegnet de respektive verdiene, slik som vist Figur 10. Ved å bruke riktig deklarering av assets og UiPath's tilegnede aktiviteter holdes passordet skjult. Det opprettes to nye variabler *UserName* og *PassWord* som får verdiene fra *in_CredentialAssetName.*

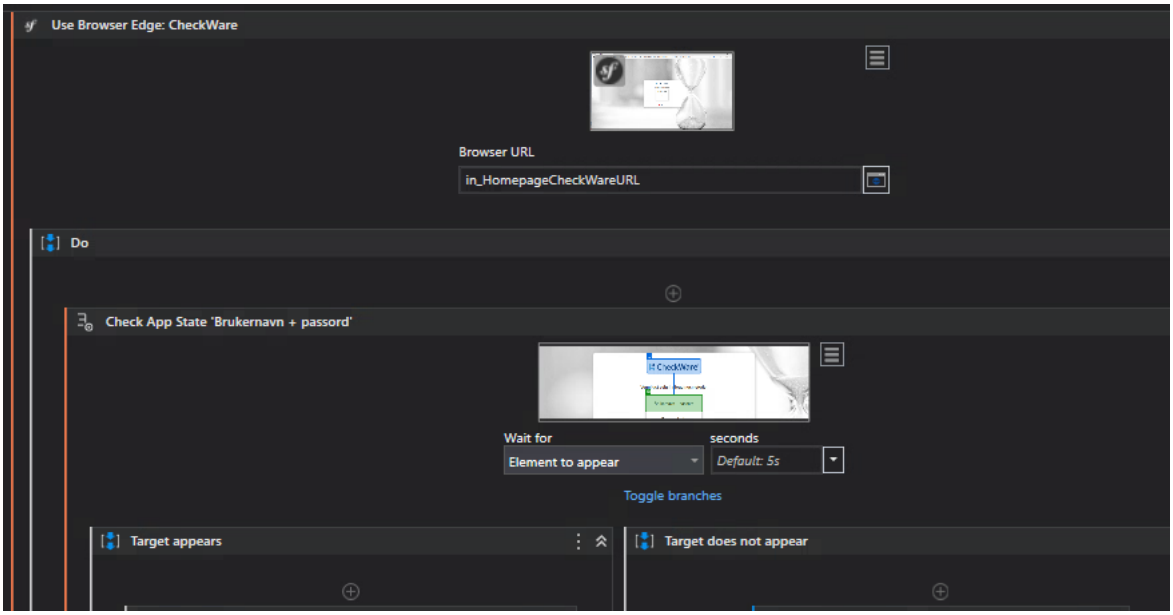


Figur 10 Utklipp av InitAllApplication workflow – Get Credential

For å bruke Browser-aktiviteter i UiPath må vi først starte en *Use Browser* aktivitet. For å bruke Browser aktivitetene må UiPath Studio ha installert en extension mot en Browser app, i dette tilfellet er det Edge. Når URL til testmiljø åpnes kan den stå i tre ulike tilstander:

(1) Kommer rett til innlogingsside, klart for å trykke knapp for å logge inn
(2) Kommer til utlogg-side. Da må det først trykkes på "Gå til Innlogging"
(3) Bruker er allerede logget inn, og lander rett på hjem-siden

UiPath tilbyr flere aktiviteter som er ment for å løse slike problemstillinger. *Check App State* aktiviteten sjekker tilstanden til en applikasjon eller en web browser ved at den kan verifisere at et element er synlig eller forsvinner fra bruker grensesnittet (UiPath, 2023). Først settes inn *Use Browser Edge* med innlogging til URL –adressen lagret i inn-argumentet fra Figur 8, deretter *Check App State*, se Figur 11. I Check App State velges et element på innloggingssiden, som er en knapp med teksten «Brukernavn + passord». Vi ønsker å se om dette elementet dukker opp innen 5 sekunder (default verdi), hvis JA, skal den videre prosessen følge sekvensen under "Target appears". Hvis NEI, følg sekvensen videre under "Target does not appear".



Figur 11 Utklipp fra InitAllApplications workflow

Jeg velger å vise videre oppbyggingen under "Target does not appear". For "Target appears" er det lagt inn lineær innloggingssekvens med Click - og Type into aktiviteter. Om man ikke lander på innloggingssiden med URL adressen *in_HomepageCheckWareURL*, er det tre muligheter:

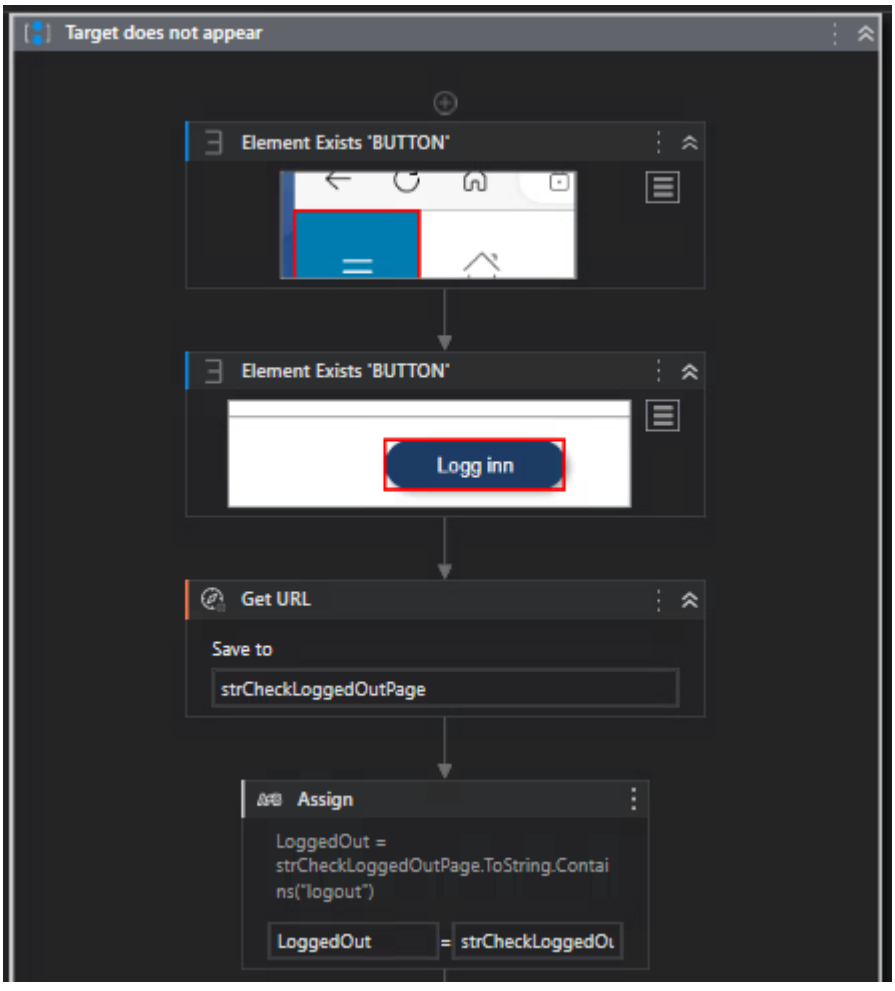
(1) Allerede logget inn, og lander på hjemsiden. I dette tilfellet vil vi ikke gjøre noe.
(2) Havner på siden der det er klart for å fylle inn feltene for brukernavn og passord.
(3) Lander på log out siden. I dette tilfellet må det klikkes på en link for å komme til rett aktivitet

For å finne hvilke av casene vi har havnet på bruker vi aktiviteten: Element Exists. Denne aktiviteten sjekker om et valgt UI element eksisterer. I dette tilfellet velger vi en Selector, som indikerer et element som vi leter etter på websiden. For å spesifisere elementet nærmere kan man bruke UI Explorer for å definere flere items for selectoren, se Figur 13.

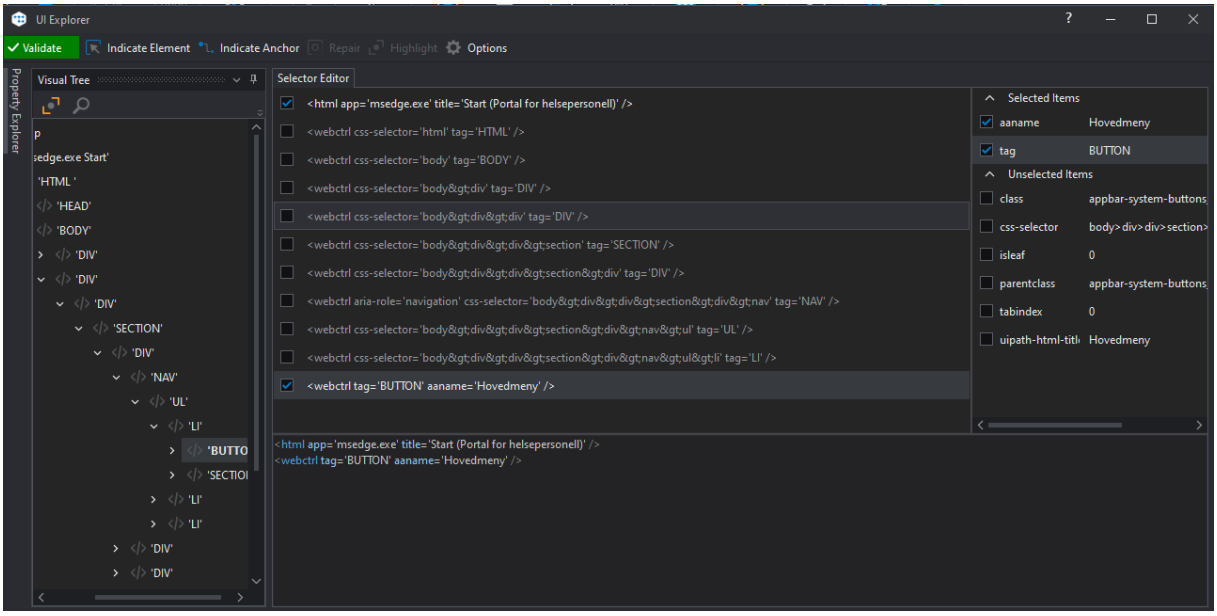
For å lete etter hovedmeny knappen for case (1) er det valgt <webctrl tag= 'BUTTON' og aaname = 'Hovedmeny'. Aktiviteten returnerer en boolsk verdi som er sann om elementet er funnet og usann ellers.

For å sjekke ut casene (1) til (3) er det brukt Element Exists for (1) og (2), der det letes etter en Selector som i begge tilfeller er oppgitt som en 'BUTTON' i html koden til nettsiden, se Figur 12. For case (1) heter den boolske variabelen AlreadyLoggedIn. For case (2) heter den PageReadyforUserName.

For å sjekke ut case (3) fant jeg ut at det sikreste var å bruke GET URL aktiviteten, da jeg ikke fant noen unike selectorer på siden. Log-out nettsiden inneholder teksten "logout". Aktiviteten GET URL henter nettadressen til nåværende side, og lagrer den som en variabel med navn *strCheckLoggedOutPage* av type string. Deretter opprettes en boolsk variabel *LoggedOut* som er sann dersom nettadressen inneholder teksten "logout". Koden for dette vises i Assign aktiviteten i Figur 12.



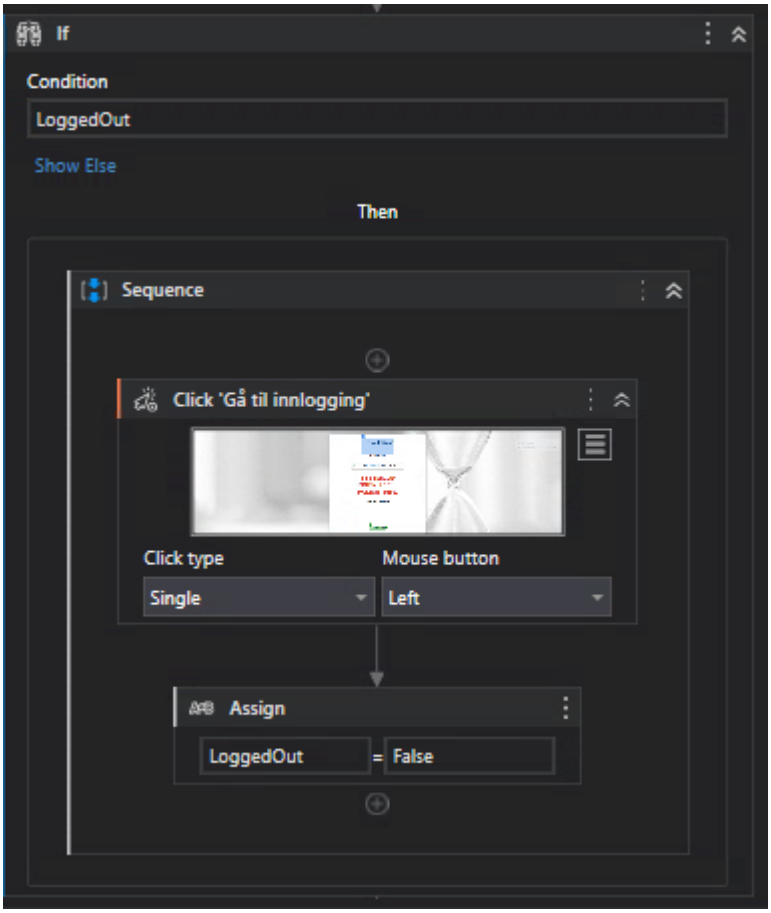
Figur 12 Utklipp fra sekvens under 'Target does not appear'



Figur 13 Ui Explorer for aktiviteten Element Exists

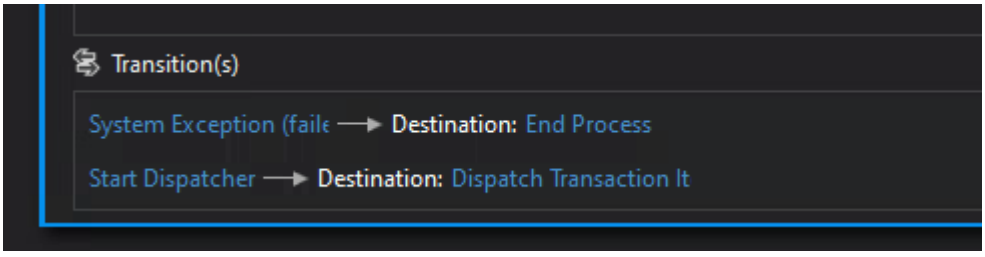
I neste ledd er det opprettet tre if-aktiviteter for å sjekke de boolske variablene. Den første if setningen sjekker om *PageReadyforUserName* er sann. Hvis den er sann er det nødvendig å gjøre en ny navigering frem til startsiden. For å gjøre det er det lagt inn aktiviteten Go To URL, med variabelen *in_HomepageCheckWareURL*.

Neste If-aktivitet sjekker om variabel *LoggedOut* er sann. Hvis denne er sann skal det klikkes på en liten lenke på siden. Til dette brukes en Click-aktivitet, se Figur 14. I denne aktivteten må det velges UI-elementer target og tilhørende anchor. Som for Elements Exist –aktiviteten er det også her mulig å spesifisere nærmere elementet det skal letes etter ved å bruke UI Explorer. Deretter må *LoggedOut* settes til usann.



Figur 14 If-aktivitet for å sjekke variabel 'LoggedOut'

Siste if-aktiviteten sjekker om AlreadyLoggedIn ikke er satt til sann. Hvis denne er usann skal det logges inn med den samme sekvens som for "Target appears". Om innloggingen går som den skal vil WorkflowException være Nothing. Mot slutten av Initialization State er variabelen *SystemException* lik *Nothing* dersom alt har gått som det skal. Overgangen Start Dispatcher følges dersom *SystemException is Nothing*. Dersom *SystemException isNot Nothing* følges overgangen System Exception til End Process.



Figur 15 Overganger fra Initalization State

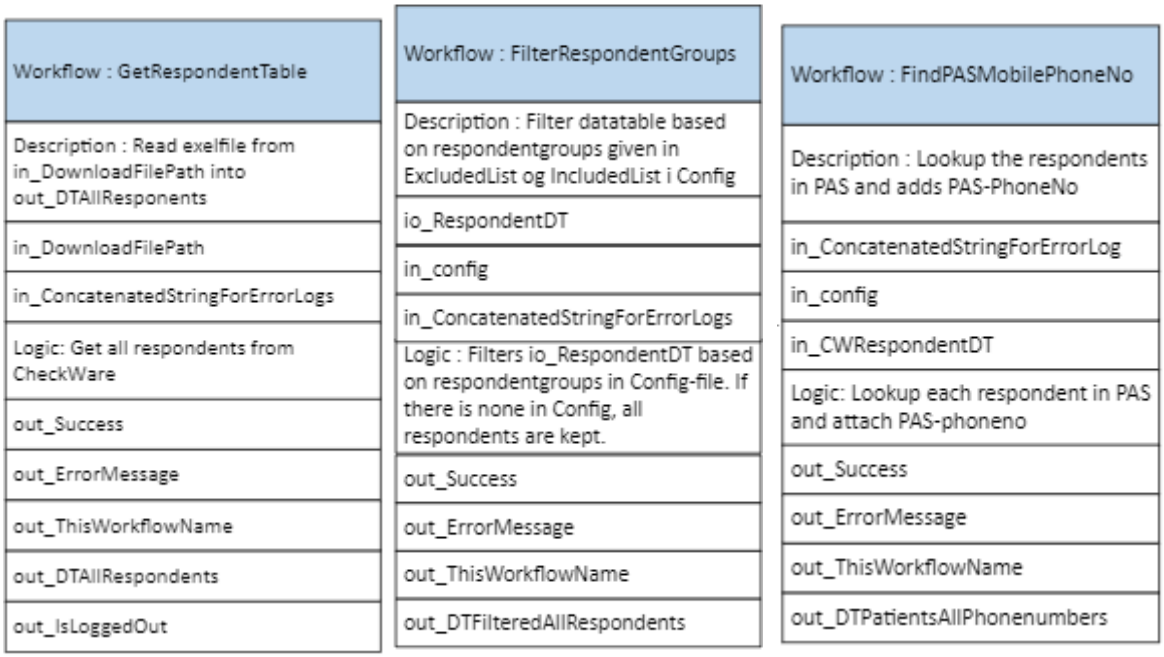
9.3 Dispatch Transaction Items

Det er i denne delen av Dispatcher hovedoppgaven utføres. Etter samtaler med veileder J.M. Obligar fant vi ut at det beste var at Dispatcher gjorde hoveddelen av jobben, og

sender til kø alle respondenter som har behov for å oppdatere telefonnummer. På den måten blir oppgaven i Performer mindre. Hovedoppgavene er kort oppsummert:

1. Hente ut alle respondenter til en datatabell
2. Finne telefonnummeret til alle respondenter i PAS og legge til som ny kolonne
3. Lete frem alle som har ulikt nummer og legge disse i egen datatabell
4. Kryptere kolonnen til *respondent username*, ettersom dette er respondentens fødselsnummer
5. Sende til kø

Det opprettes en variabel ConcantenatedStringForErrorLogs som inneholder informasjon om type HF/helseforetak og at dette er prosesstype dispatcher. Denne variabelen sendes som inn argument til hver workflow og brukes som start på hver log-message, se Figur 16.



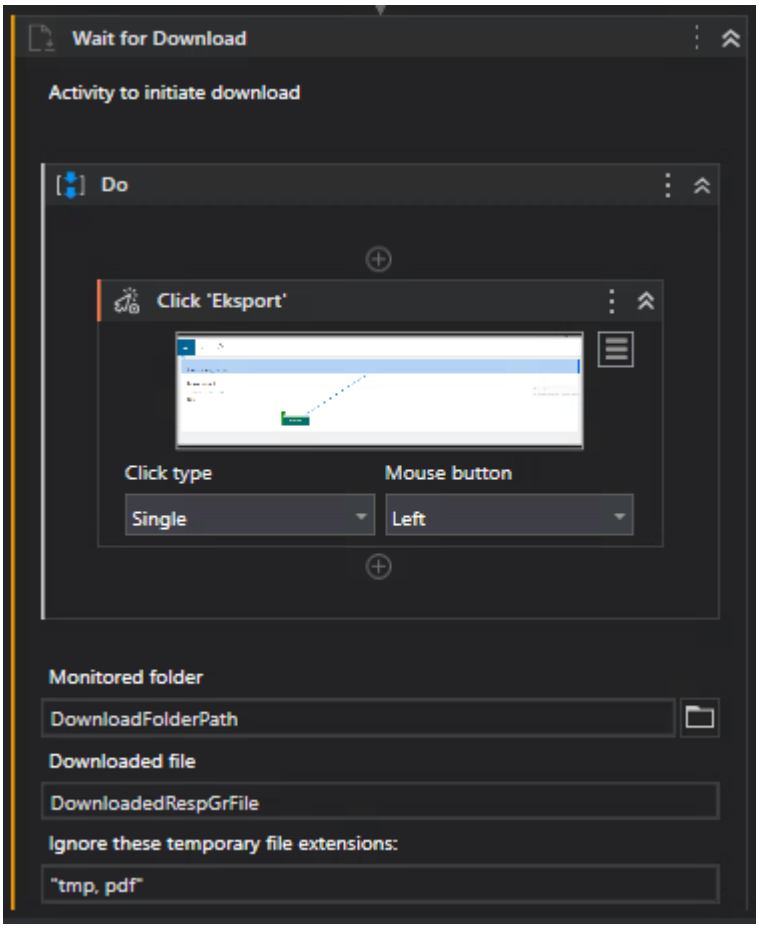
Figur 16 Tre første workflows i Dispatch Transaction Items prosessen. Komplett oversikt i Vedlegg 2

9.3.1 Workflow: GetRespondentTable

Etter å ha fått tilgang til testmiljøet til CheckWare var det viktig å klikke bli kjent med applikasjonen. For å hente ut alle respondenter med tilhørende gruppe/avdeling er den raskeste måten å eksportere ut en liste over alle respondenter i csv format. Dette er et format som UiPath Studio enkelt kan gjøre om til en datatabell.

Ved nedlasting av filer, har UiPath en nyttig aktivitet som heter Wait for Download. Denne aktiviteten venter med å fortsette til filen er ferdig nedlastet, default verdien er 300 sekund. Denne kan økes om nødvendig. Initieringen av nedlastningen plasseres i Do-sekvensen i Figur 17. I dette tilfellet skjer initieringen av nedlastningen ved å klikke på

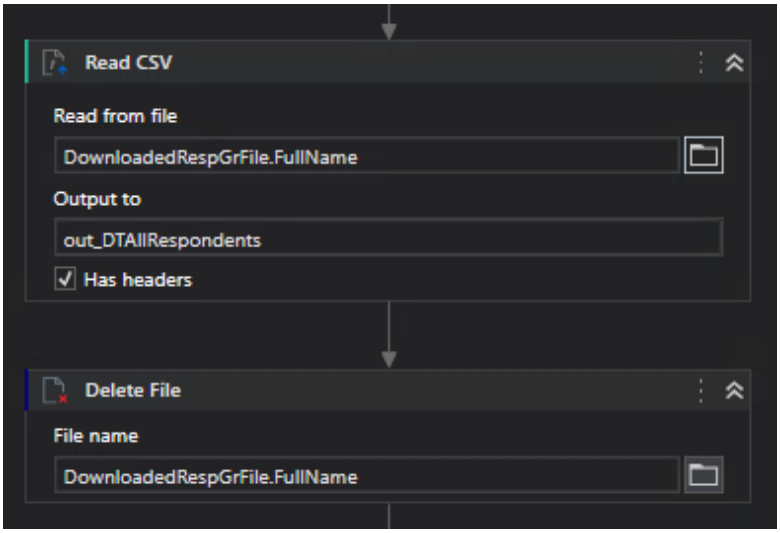
"Eksport" knappen i CheckWare. Derfor er det lagt inn Click-activity i Do-sekvensen. Videre oppgis sti til mappe som skal overvåkes for nedlastning under "Monitored folder". I Downloaded file feltet oppgis variabelnavnet til den nedlastede filen, i dette tilfellet *DownloadedRespGrFile*. Det er også ført opp filnavn som skal ignoreres.



Figur 17 GetRespondentTable: Wait for Download activity

Videre skal csv filen gjøres om til en datatabell. Til dette brukes aktiviteten Read CSV, se Figur 18. I denne aktiviteten oppgis filnavnet. For å få med hele navnet til den nedlastede filen brukes *DownloadedRespGrFile.FullName*. Det må også oppgis hvilket skilletegn som er brukt, i dette tilfellet semikolon, og at filen har overskrifter. Output er en datatabell og lagres som et ut-argument *out_DTAllRespondents*.

Etter å ha opprettet datatabellen ønsker vi å fjerne filen fra VDI'en. Til dette brukes Delete File aktiviteten med det samme filnavnet som den nedlastede filen.



Figur 18 GetRespondentTable: Read CSV and Delete File

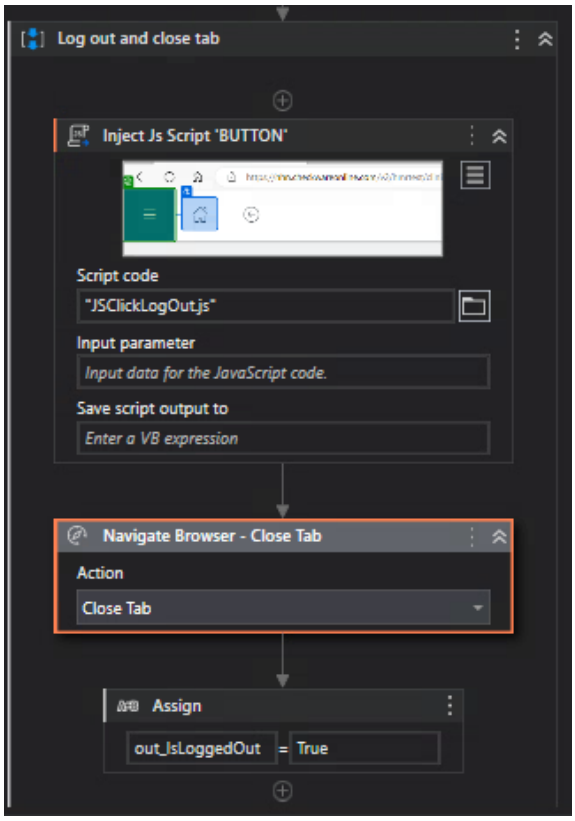
Dispatcher jobben i CheckWare er nå fullført, og det kan derfor logges ut og webapplikasjonen kan lukkes. UiPath har aktiviteter som håndterer dette. For læringen sin del ønsket jeg å prøve å programmere denne delen i javascript, for så å bruke Inject Javascript aktiviteten. Istedenfor å legge inn to klikk-aktviteter kunne jeg heller ha en Inject Js Script aktivitet. Jeg fant referansen til elementene det skulle klikkes på med å åpne Microsoft Edge Dev Tools. Javascript koden ble skrevet i Visual Studio Code og lagres i mappen sammen med hele Uipath koden til Dispatcher. I aktiviten angir jeg knappen til første funksjon clickButton. Deretter har jeg laget neste funksjon som klikker på logout knappen, og kaller på funksjonen til slutt, se Figur 19.

```
1   function clickButton() {
2     const button = document.querySelector('button[title="Hovedmeny"]');
3     if (button) {
4       button.click();
5     }
6   }
7
8   function clickLogout() {
9     const link = document.querySelector('.sidebar-item--link-card a[href="#logout"]');
10    if (link) {
11      link.click();
12    }
13  }
14
15  clickLogout();
```

Figur 19 GetRespondentTable: Injected Javascript code

For å lukke og stenge vinduet kan man ikke bruke window.close(), da det tillates ikke å lukke vinduet med Javascript. I UiPath finnes en aktivitet som heter Navigate Browser,

med en nedtrekksmeny over ulike handlinger. For å lukke vindu velges *Close Tab*. Hele sekvensen som inneholder nedstengning og utlogging er vist i UiPath er vist i Figur 20.



Figur 20 GetRespondentTable: Log out and close tab

9.3.2 Workflow: FilterRespondentGroups

I første produksjonsfase skal automatiseringen begrenses til å kun berøre respondenter i CheckWare som er knyttet til kirurgisk poliklinikk i Ålesund og Molde. Dette for å gjøre feilhåndteringen enklere og begrense nedslagsfeltet til eventuelle feil. Det trengs derfor en logikk som kan filtrere ut de respondentene som er tilknyttet disse avdelingene. Denne logikken bør lages generisk slik at den kan brukes i alle CheckWare RPA-prosesser. Det kan tenkes at det i ettertid vil bli behov for å styre hvilke avdelinger/sykehus de ulike prosessene skal gjelde for.

I ProcessConfig filen oppgis dermed to nye verdier: «ExcludedListBehandling» og «IncludedListBehandling» på denne måten:

```
"ExcludedListBehandling": {
    "Description": "The list of respondent group names to exclude. Important that the group
    names are written in the same way as in CheckWare,",
    "Value": {
```

```
                "RespGroup1": "",
                "RespGroup2": ""
                },
        "Type": "Setting"
        },

  "IncludedListBehandling": {
        "Description": "The list of respondent group names to include.Important that the group
        names are written in the same way as in CheckWare.",
        "Value": {
                "RespGroup1": "MO.Kir.Ortopedisk ",
                "RespGroup2": "AL.Kir.Ortopedisk"
                },
        "Type": "Setting"
        },
```


I «ExcludedListBehandling» føres det opp navnene på avdelingene som skal ekskluderes. Hvis «ExcludedListBehandling» er tom sjekkes det om det er ført opp noe i «IncludedListBehandling». Om det er behov for å begrense at en prosess kun kjøres for noen få avdelinger, føres det opp her. I vårt tilfelle føres det opp de to avdelingene prosessen skal kjøre for i første prod.fase. Hvis ingen avdelinger føres opp gjøres det ingenting, og alle respondentene fra datatabellen generert i forrige Workflow tas med videre. Logikken er presentert i Figur 21.

```
If ExcludedListBehandling exists AND is NOT Null or Empty
  > Filter RespondentTable based on the provided «respondent group
name»
ElseIf IncludedListBehandling exists AND is NOT Null or Empty
  > Filter RespondentTable based on the provided «respondent group
name»
Else
  > Do nothing
```

Figur 21 Pseudokode for Workflow: FilterRespondentGroups.xaml

9.3.2.1 ExcludedListBehandling

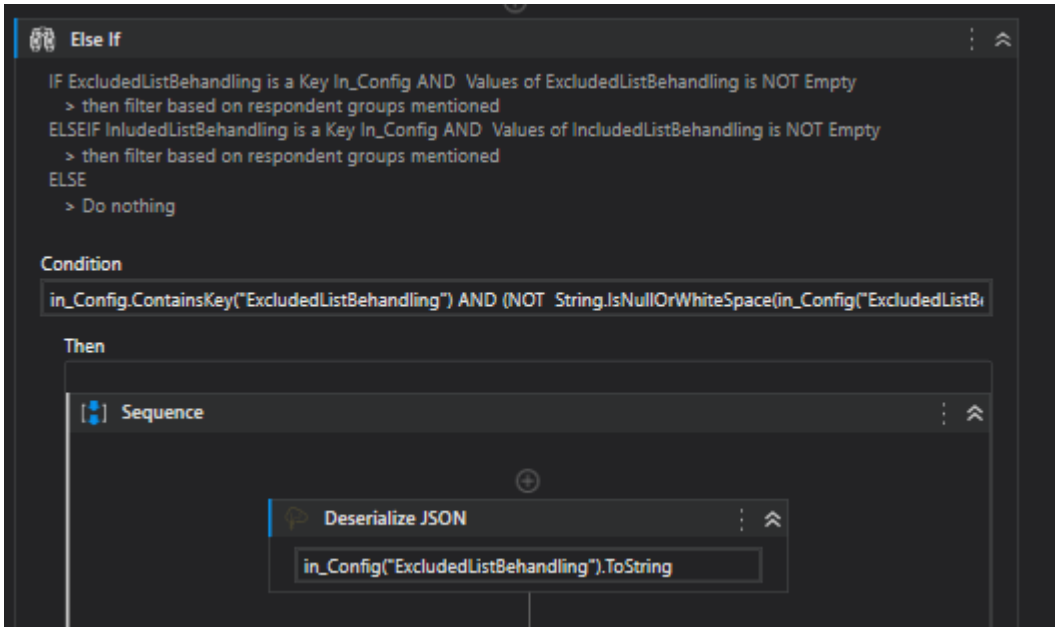
Den første if-setningen med VB.NET:

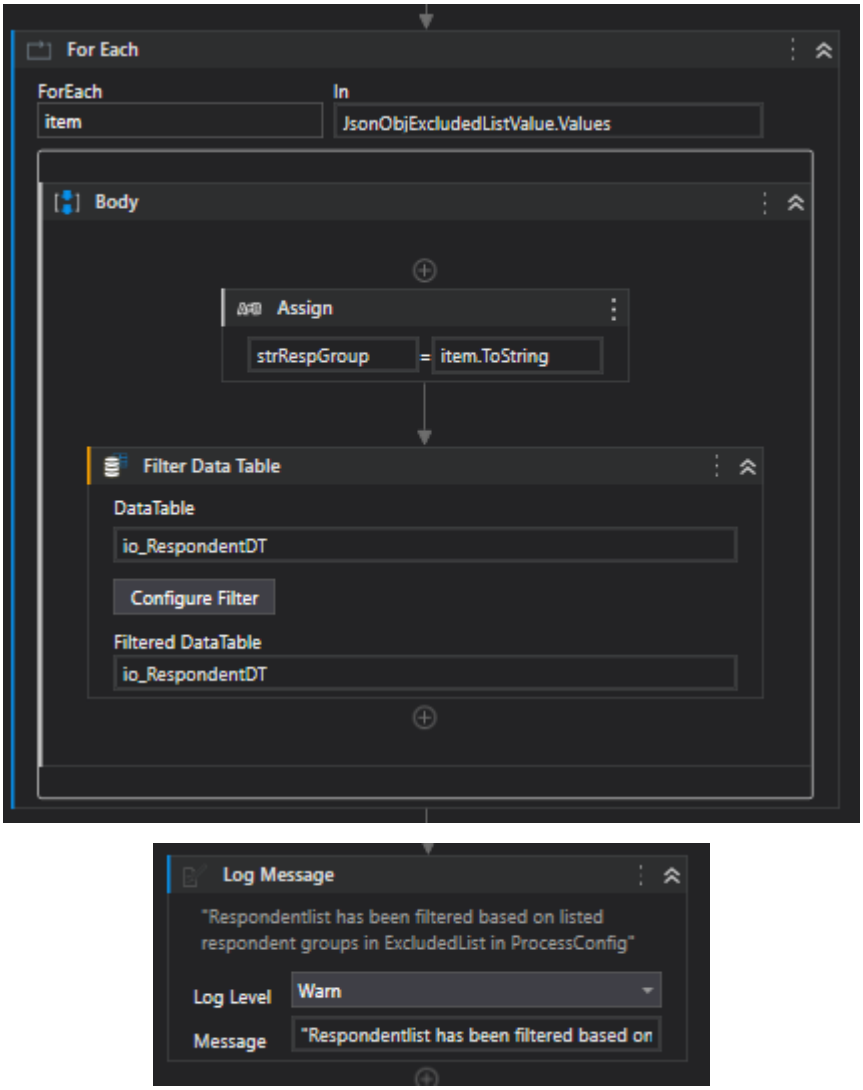
in_Config.ContainsKey("ExcludedListBehandling") AND (NOT String.IsNullOrWhiteSpace(in_Config("ExcludedListBehandling").ToString))

Formel 1 Argumentet i IF setningen for ExcludedListBehandling

Denne setningen returnerer en boolsk variabel som er sann dersom ExcludedListBehandling eksisterer som en nøkkel i Config- filen og at verdien i "Value" ikke er tom.

For å kunne filtrere datatabellen kjøres det en for-løkke for hver nøkkel-verdi oppgitt i "Value". Deretter filtreres datatabellen med *Filter Data Table* – aktiviteten. Her oppgis det at de radene der kolonneverdiene inneholdt verdien til item skal fjernes fra datatabellen. Output-tabellen settes til den samme tabellen som sendes inn. Utklipp av den delen av Workflow'en som filtrerer ut fra ExcludedList er vist i Figur 22.





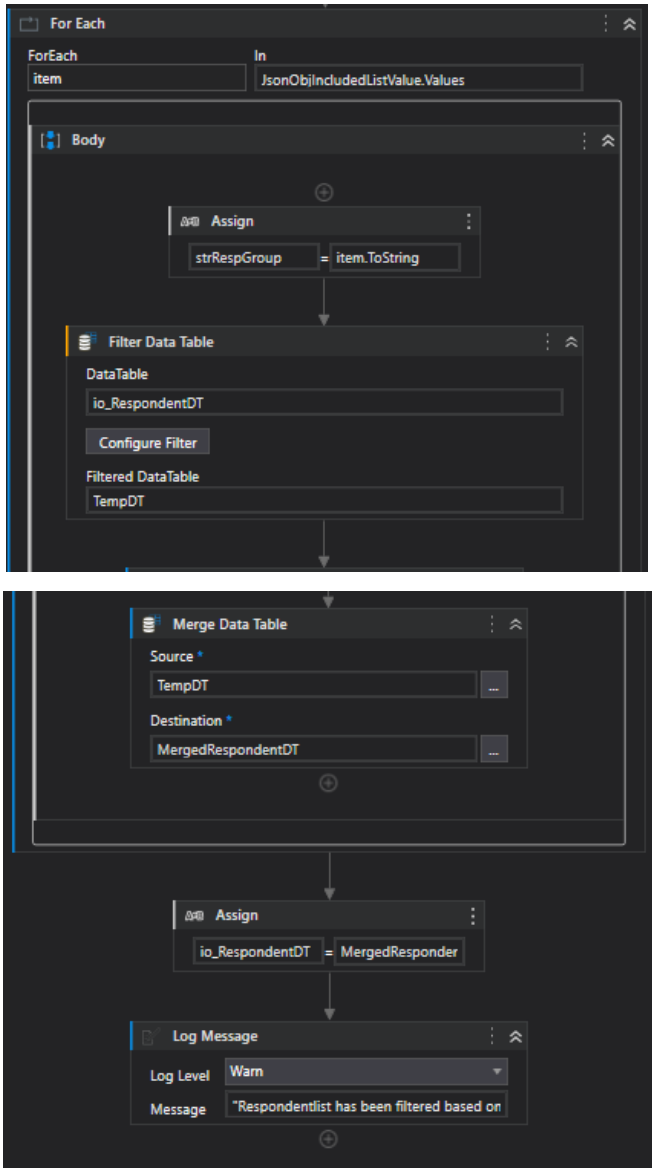
Figur 22 Utklipp av FilterRespondentGroups.xaml som viser den delen av koden som filtrerer datatabellen basert på ExcludedListBehandling verdiene i ProcessConfig filen.

9.3.2.2 IncludedListBehandling

Denne delen av koden er ganske lik som for ExcludedListBehandling, med noen endringer. ElseIF setning for IncludedListBehandling er bygd opp på samme måte som Formel 1. Forskjellen her er at når tabellen skal filtreres, må det opprettes to midlertidige tabeller. For-løkken henter ut verdiene til de avdelingene vi vil filtrere tabellen på. Denne gangen vil vi beholde de radene som filtreres ut for hvert steg i for-løkken. Logikken er vist i Figur 23.

```
MergedRespondentDT = io_RespondentDT.Clone
For each item in JsonObjIncludedListValue.Values
        > Filter Data Table with Input: io_RespondentDT, Output: TempDT
          Merge TempDT with MergedRespondentDT
io_RespondentDT = MergedRespondentDT
```

Figur 23 Pseudokode/logikk for å inkludere alle filtrerte rader ut fra oppgitt «respondent group»



Figur 24 Utklipp av FilterRespondentGroups.xaml som viser den delen av koden som filtrerer datatabellen basert på IncludedListBehandling verdiene i ProcessConfig filen.

9.3.3 Workflow: FindPASMobilePhoneNo

Denne workflowen legger til tre nye kolonner til datatabellen, "PASMobilePhoneNumber", "MobileReservation" og "PID". Mobilreservasjonen er en boolsk variabel som er sann dersom pasienten har valgt å reservere seg mot elektronisk kommunikasjon. PID er pasientens unike pasientnummer, som tas med som informasjon til logging. Logikken i workflowen er at den looper gjennom tabellen over alle respondenter, og gjør et API kall mot PAS og henter ut informasjonen derfra. Denne logikken er det J.M. Obligar som har laget, og derfor nevnes det kun kort her.

Denne workflowen håndterer og logger scenario 1. Dersom den ikke får et resultat fra PAS på respondenten logges det som en feilmelding, slik at helsepersonalet kan undersøke tilfellet nærmere.

Workflow :FindAllPatientsToUpdate
Description : Compare PAS Phoneno vs CW PhoneNo and write out a DT with all patients where phoneno is different
in_DTPatientsAllPhonenumbers
in_ConcatenatedStringForErrorLog
Logic: Compares PhoneNumbers and creates a datatable with all Respondents where update of PhoneNo is necessary
out_Success
out_ErrorMessage
out_ThisWorkflowName
out_DTPatientsToUpdatePhoneNo

Workflow : FilterRespondentTable
Description : Filter datatable of all respondents, removing unnecessary columns and rows
in_DTAllRespondents
Logic : Remove unnecessary columns in datatable, keep respondent-id, PID, name, respondent groups, phonenumbers and so on
out_Success
out_ErrorMessage
out_ThisWorkflowName
out_DTFilteredAllRespondents

Figur 25 FindAllPatientsToUpdate og FilterRespondentTable i Dispatcher

9.3.4 Workflow: FindAllPatientsToUpdate

I hovedsak skal denne workflow'en opprette en ny datatabell over alle respondenter der nummeret må oppdateres. Datatabell over alle respondenter fra forrige workflow: FindPASMobilePhoneNo sendes inn som argument, og resultatet er en datatabell(out_DTPatientsToUpdatePhoneNo) over alle respondenter som skal sendes til kø, se Figur 25. Det er også naturlig at det her filtreres ut de tilfeller fra scenario 2 til 6, som skissert i Figur 4. For hvert scenario skal det logges hva som er funnet, slik at denne informasjonen kan letes opp ved behov senere. Det er også viktig å logge hva roboten har funnet, og hva den ikke håndterer eller sender til kø. Pseudokoden over logikken i denne workflowen er vist i Figur 26.

```
out_DTPatientsToUpdatePhoneNo = in_DTPatientsAllPhonenumbers.Clone()
For each Row in in_DTPatientsAllPhonenumbers
        CWPhone = Row("respondent phone")
        PASPhone = Row("PAS phone")
```

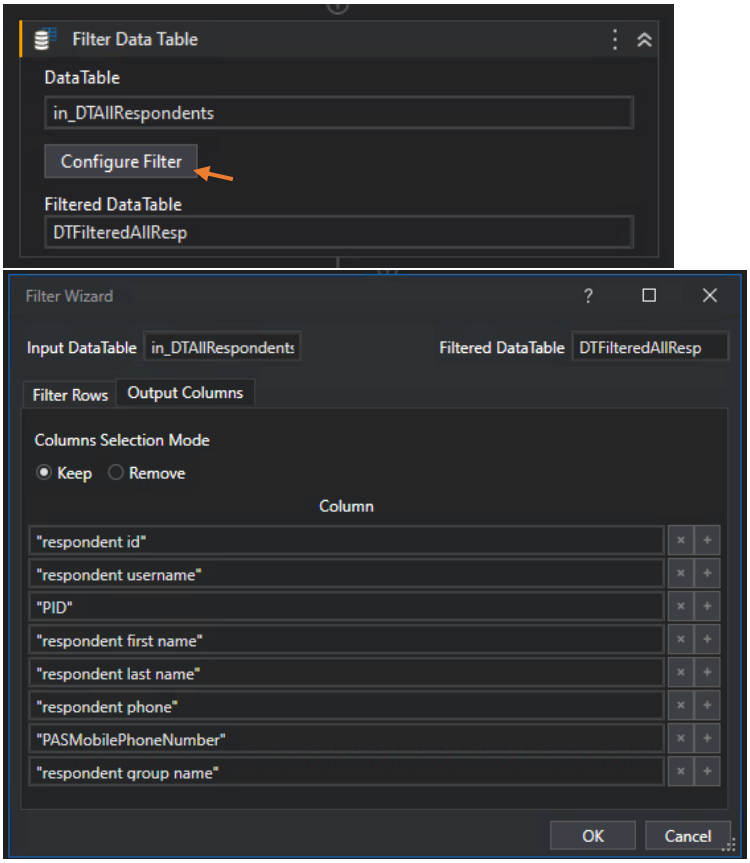
```
> If CWPhone.Equals(PASPhone)
    LogMessage: “Same number in CW and PAS2”
> ElseIF PASPhone Equals Null or Empty
    ErrorMessage: “Empty number in PAS”
> ElseIF Mobilereservation equals “true”
    ErrorMessage: “Patient has reserved from electronic contact”
> Else
    LogMessage: “Tlfnr CW != Tlfnr PAS - Telefonnummer sendes til kø for oppdatering”
    Add Data Row to out_DTPatientsToUpdatePhoneNo
```

Figur 26 Pseudokode/logikk for å filtrere ut de ulike scenarioene

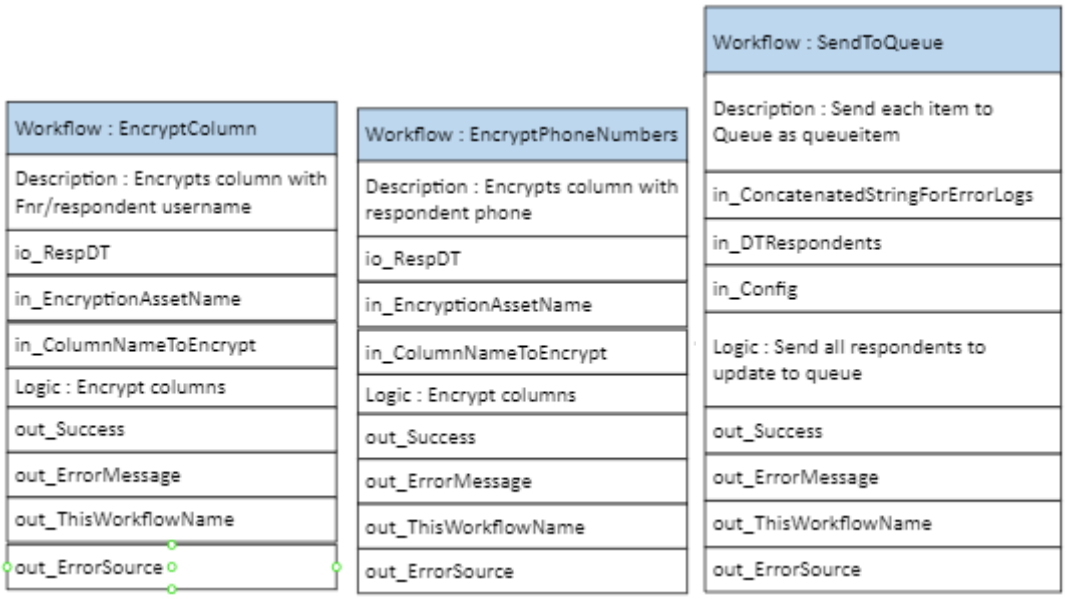
Til denne logikken er det i UiPath brukt aktivitetene ”For Each Row In Data Table” og ”Else If”. Disse aktivitetene er intuitive å bruke når man kan noe programmering fra tidligere.

9.3.5 Workflow: FilterRespondentTable

Den eksporterte tabellen fra CheckWare inneholder en linje for alle respondenter tilknyttet hver av respondent gruppene. Med andre ord kan en respondent være ført opp i flere rader, avhengig av hvor mange respondent grupper vedkommende tilhører. Dermed må det lages en workflow for å unngå at den samme respondenten blir sendt til kø flere ganger. Det er likevel viktig å samle opp tilhørighet til avdelinger. Kolonnen med respondentgruppe gjøres om til en array som inkluderer alle tilhørende avdelinger. Datatabellen inneholder flere kolonner som ikke er nødvendige. Alle unødvendige kolonner slettes, og det skal kun beholdes kolonner for respondent-id, respondent username, FName, LName, respondent phone, PASMobileNumber, respondentgroup og PID. Aktiviteten Filter Data Table i UiPath ble brukt til dette formålet, der det listes opp alle kolonner som skal beholdes, se Figur 27.



Figur 27 Aktivitet Filter Data Table. Filter Wizard vinduet dukker opp ved å klikke på "Configure Filter".



Figur 28 Workflows som krypterer og sender til kø i Dispatcher

9.3.6 Workflows: EncryptColumn og EncryptPhoneNumbers

Dispatcher inneholder to workflow's som krypterer kolonner. Den første krypterer kolonnen med fødselsnummer/respondent username, og den andre krypterer kolonnen som inneholder respondentens telefonnummer i CheckWare. Disse er identisk bygd opp, med de samme inn og ut argumentene, se Figur 28. I Hemit er det laget en generisk workflow for EncryptDTColumns, som kan benyttes i Dispatcher prosesser. Det er nettopp denne jeg har brukt her. Resultatet er at datatabellen som sendes ut, i dette tilfellet io_RespDT, returneres med kun krypterte data i oppgitt kolonne. I performer finnes tilsvarende generiske workflow for dekryptering.

9.3.7 Workflows: SendToQueue

I denne delen av Dispatcher skal hver rad i in_DTRepsondent sendes til kø. Pseudokoden til workflowen er vist i Figur 29. Kø-navnet er oppgitt som inn argument til Dispatcher.

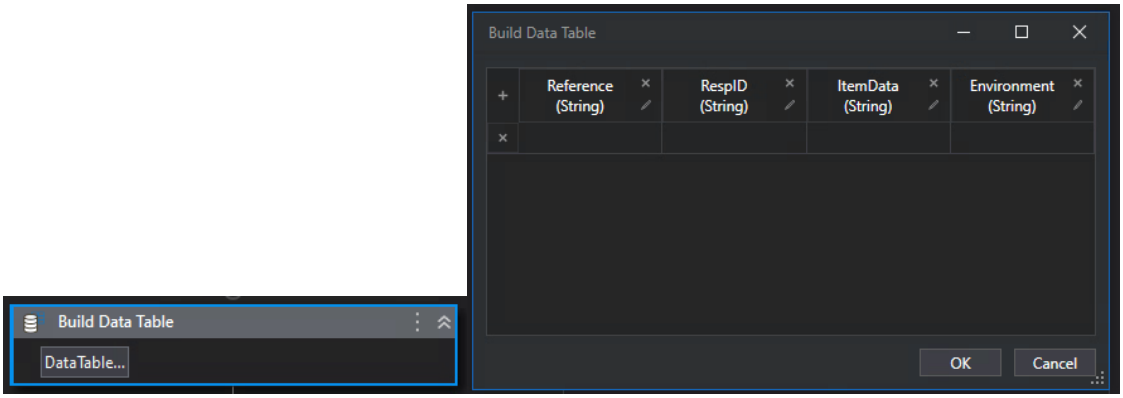
```
Opprett ny tabell: QueueItemData (Activity: Build Data Table)
Opprett ny tabell: UniqueDT (Activity: Filter Data Table)

For each Row in UniqueDT (Activity: For Each Row in Data Table)
  Filter in_Respondents datatable basert på Row.Item("respondent id") til
FilteredDTRespondent (Activity: Filter Data Table)
  CaseItemData = FilteredDTRespondent til JsonString
  Add datarow to QueueItemData (Activity: Add Data Row)

If QueueItemDatta.Rows > 0 (Activity: If)
 Add Row Items to Queue (Activity: Bulk Add Queue Items)
Else
 Logmessage: "0 items sent to Queue"
```

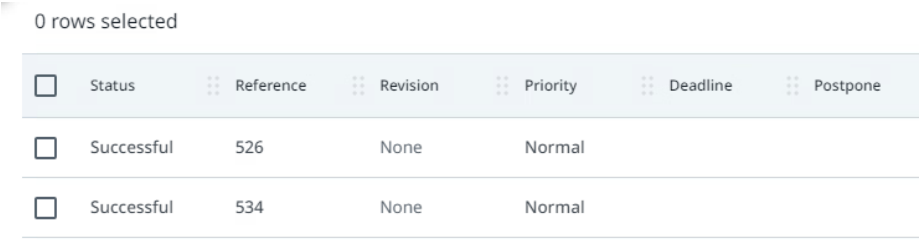
Figur 29 Pseudokode/logikk for å sende hvert rad-element til kø

Først opprettes det en ny datatabell med navn QueueItemData. For å gjøre det bruker jeg Build Data Table – aktiviteten med fire kolonner.

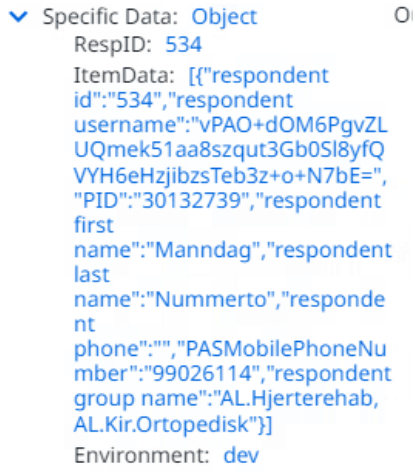


Figur 30 Build Data Table activity i UiPath. Oppretter ny tabell: QueueItemData

Reference og RespID kolonnen skal inneholde respondent id. Reference kolonnen legges til som første kolonne da vil den automatisk bli reference for hvert QueueItem i Orchestrator, se Figur 31. Videre vil de tre siste kolonnene være selve Transactions objectet, se Figur 32.



Figur 31 Utklipp av QueueItems i DEV-Orchestrator fra CheckWare testmiljø med fiktive respondenter.



Figur 32 Utklipp fra DEV-Orchestrator - Transaction item med RespID 534 av fiktiv respondent fra testmiljø

10.0 Performer

Performer i UiPath er en komponent innenfor robotic enterprise framework (REFramework) på samme måte som Dispatcher. Dette er den delen av prosessen som gjøres etter at et element er lagt til kø. Performer skal utføre selve handlingen i RPA prosessen. I dette tilfellet skal Performer delen gjøre oppdateringen av telefonnummeret til respondenten i CheckWare for alle elementer som er lagt til kø i Orchestrator. Her har jeg brukt Hemit's Performer Template der alle states og overganger er ferdig bygd opp, se Figur 33.

De to første inn argumentene er identiske som i Dispatcher. Nummer tre inneholder tallet for max antall application exception (System Exception) som tillates i Performer. Dette oppgis når det opprettes trigger i Orchestrator. Dette er et nyttig argument, fordi i produksjon ønsker man ikke at performer skal fortsette og jobbe videre i køen dersom det er noe feil i applikasjonen.

In_OrchestratorQueueName

In_TimeSpanAssetName

In_MaxConsecutiveAE

Performer trigges av at nytt element er lagt til i kø. Oppbyggingen av performer prosessen er vist i Figur 33.

Set Start Timer og Initalization State er identisk som i Dispatcher, oppbyggingen kan derfor gjenbrukes her. Fra initalization state er det to overganger. Den ene som leder til Get Transaction Data når SystemException = Nothing. I motsatt fall ledes prosessen direkte til End Process der alt lukkes og stenges ned.

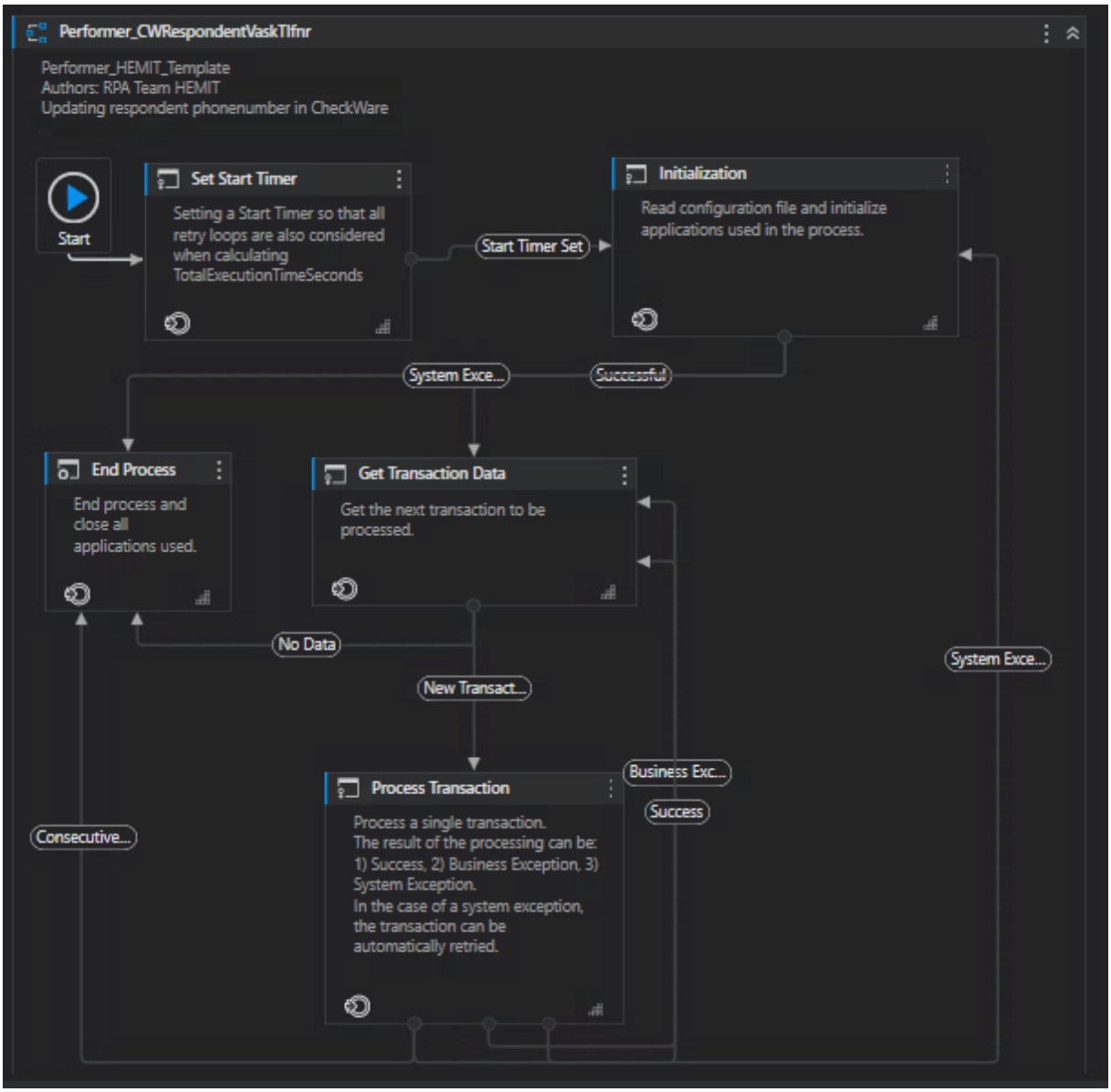
SystemException is Nothing -> Get Transaction Data State

SystemException isNot Nothing -> End Process State

Get Transaction Data sjekker om Stop signalet er gitt eller om evt max antall transaksjoner er nådd før den henter neste transaksjon fra kø. Fra denne staten er det to mulige utfall;

TransactionItem IsNot Nothing -> Process Transaction State

TransactionItem Is Nothing -> End Process State



Figur 33 Performer template i Hemit for CheckWare Respondent Vask Tlfnr

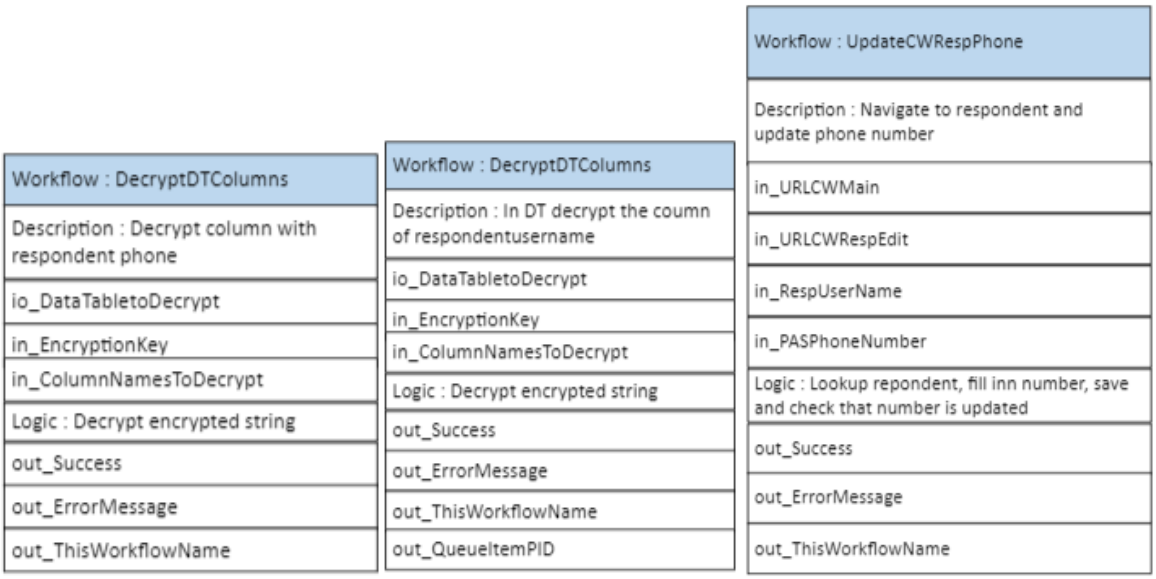
Fra Process Transaction er det fire mulige utfall:

1. BusinessException: BusinessException IsNot Nothing-> Logger en forretningsfeil og går videre til Get Transaction Data State
2. Success: SystemException Is Nothing And BusinessException is Nothing-> Get Transaction Data State
3. SystemException: SystemException IsNot Nothing -> Initalization State
4. Consecutive App Error: CounterConsecutiveAppError > CINT(in_MaxConsecutiveAE)

I Performer kunne jeg gjenbruke workflow'ene og oppbyggingen fra Dispatcher for Initalization og End process state. I Performer har jeg kun lagt inn nye workflows i Process Transaction State.

10.1 Process Transaction

Som en del av templaten leses Itemdata ut fra transaksjonselementet, og gjøres om til en datatabell med samme navn ItemData.



Figur 34 Oppbygging av workflows fra Process Transaction

10.1.1Workflow: DecryptDTColumns

Denne workflowen blir benyttet i to omganger som vist i Figur 34. Dette er en standard workflow der jeg kun har lagt inn datatabell som skal dekrypteres (io_DataTabletoDecrypt), EncryptionKey fra Assets i Orchestrator og kolonnenavnet som skal dekrypteres.

I første runde dekrypteres respondentens telefonnummer. Deretter har jeg laget en logikk for å sjekke om nummeret er et fødselsnummer. Hvis det et fødselsnummer skal de fem siste sifrene byttes ut med *. Utklipp av sekvens som gjør dette er vist i Figur 35.

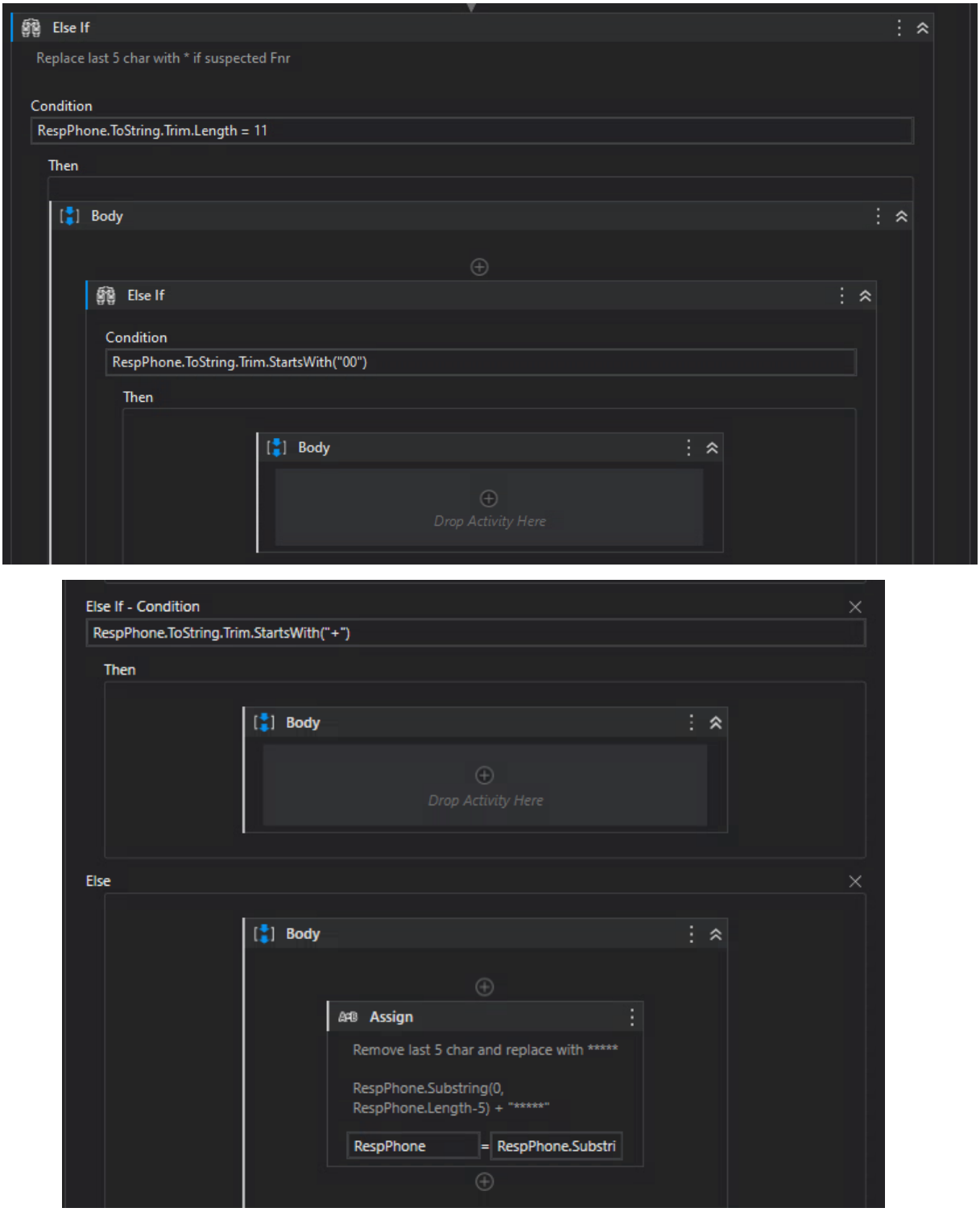
Logikken er

```
If RespPhone.Length = 11
    ElseIf RespPhone startswith ”00”


    ElseIf RespoPhone startswith ”+”
```

```
Else
  RespPhone.replace last 5 digits with ”*”
```

Etter dette lages en ConcatenatedString til logging, der hver keys and values deles med en helstrek/pipe ”|”. Dette gjøres for å lette feilsøking i Splunk. Deretter dekrypteres respondentens brukernavn(fødselsnummer) med workflow DecryptDTColumns.



*Figur 35 Sekvens som erstatter de fem siste tall med * hvis fødselsnummer*

10.1.2 Workflow: UpdateCWRespPhone

Denne workflowen skal navigere frem til respondent, oppdatere telefonnummeret og kontrollere at det er utført. Argumentene som sendes inn er

1. in_UrlCWMain
2. in_UrlCWRespEdit

3. in_RespUserName
4. in_PASPhoneNumber

Url til hovedsiden og til respondentens edit side opprettes i Initalization. Respondentens edit side endres i prosessen for hver transaksjon der RespID legges inn i url.

```
Logge inn til hovedsiden
Gå til respondent edit side
Sjekk at siden er åpen
  Hvis JA
    Hent respondentens brukernavn fra nettsiden
    If respondent brukernavn = in_ RespUserName
      Tøm feltet med telefonnummer, skriv inn in_PASPhoneNumber og trykk enter
      Les av nummeret og lagre i variabel CheckInputPhoneNo
      If CheckInputPhoneNo NOT Equals in_PASPhoneNumber
        Kast System.Exception & Log ErrorMessage
    Else
      Kast System.Exception & Log ErrorMessage
  Hvis NEI
    Kast System.Exception & Log ErrorMessage
```

11.0 Code Review og Testing

Etter utviklingsfase ble det gjort code review med systematisk gjennomgang av koden og prosessen i UiPath. I utviklingfasen var det mest fokus på logikk og få oppgavene til å fungere som ønsket, og lagt mindre vekt på logging. Det meste av tiden i code review gikk med på å sørge for at all logging gjøres i henhold til intern standard.

For å teste prosessen ble det opprettet fiktive respondenter i test-miljø i CheckWare. Ut fra oversikt over fiktive pasienter i PAS-testmiljø kunne det opprettes respondenter i CheckWare med korrelerende fødselsnummer. Det var viktig å opprette respondenter slik at koden ble testet for alle scenarioene listet opp i Figur 4, med rett respons i feilmelding og logging.

Grunnen til at det er viktig å få gjort loggingen riktig er at det i ettertid skal kunne gjøres feilsøking i Splunk, og unngå å måtte lete etter hvor feilen ligger, og hva roboten

har gjort, og evt ikke gjort. All informasjon om respondent, helseforetak, avdeling og lignende skal legges til loggene. Veileder J.M. Obligar sørget for å opprette en variabel ConcatenatedString som samler sammen all denne informasjonen i starten av prosessen. Denne variabelen sendes deretter som et argument inn i Workflow's og kan brukes direkte inn i logger.

I produksjonsfasen skal det i første omgang skilles ut respondenter som tilhører kirurgisk ortopedisk i Molde og Ålesund. I testfasen ble det oppdaget at det manglet logikk for nettopp dette, og det ble utarbeidet ny Workflow: FilterRespondentGroups i Dispatcher som kunne skille ut de navngitte avdelingene i ExkludedList og IncludedList.

Det dukket underveis opp et ønske fra helseforetak i Helse Midt om at de ønsker å filtrere logg-data ut slik at de kan analysere hvilke RPA prosesser som arbeider i deres helseforetak, for å vurdere innsparinger og lignende. For å løse dette må derfor instansnavnet HF hentes ut og legges til i logger. Siden det opprettes egen kø for hvert av HF'ene i Orchestrator, kan HF navnet hentes ut fra argumentet (In_OrchestratorQueueName) som hentes fra kø'en som trigger prosessen. HF navnet legges til i variabelen ConcatenatedString. Dette gjøres i både Dispatcher og Performer.

Etter å ha gjennomført code review og testing av Dispatcher og Performer med tilfredstillende logging, ble hele koden pushet opp i teamets Azure devops plattform. Dette er en samlende plattform som forenkler samarbeidet, og fungerer som en repository med versjonskontroll og en backup. Dette er «on prem», som vil si at alt er lagret på en lokal server. Det er etablert arkitektur med pipelines mellom UiPath Orchestrator og Azure, som gjør at alltid siste versjon av koden kjøres. Så lenge koden kjøres i dev-miljøet vil alle endringer i koden som blir pushet opp i azure, automatisk bli rullet ut/«deployed» ved CD (Continous Deployment). Når koden etter hvert skal inn i prod-miljøet, må ny kode som pushes opp i Azure ved hjelp av CI (Continous Integration) måtte kontrolleres og godkjennes av en annen i teamet før den rulles ut ved CD. Dette for å forhindre at kode med større feil sendes ut i produksjon.

Til slutt ble det opprettet et eget dashboard for prosessen i Splunk. I splunk kan det bygges opp søk slik at man kan hente ut ønsket informasjon om hva roboten gjør i drift, og for å fange opp de unntakene som må til manuell behandling. For å drive vedlikehold og feilsøking i ettertid er god oppbygging av logger utvikling essensielt.

12.0 Møte med kunde i forkant av produksjonssetting

Før produksjonssetting skal det alltid være et møte med prosesseier/kunde der man går gjennom hele prosessen og viser hvordan roboten jobber. Det ble kalt inn til møte med prosesseier den 10.mai. Der ble det gjort følgende avklaringer:

1. I hypercare-fasen, altså første produsjonssettig, begrenser vi oss til to avdelinger i HMR: AL.Kir.Ortopedisk og MO.Kir.Ortopedisk. De som jobber med CheckWare i disse avdelingene må få nødvendig briefing i forkant av prod-setting
2. Roboten skal kjøres unattended to ganger i døgnet, kl 05:00 og kl 15:00
3. Det ble bestemt hvem hos prosesseier som skal motta mail fra Splunk når det logges forretningsfeil, og tilfeller som må til manuell behandling.
4. Videre bredding av prosessen ut til flere avdelinger avgjøres tidlig høsten 2023.

13.0 Første produksjonskjøring

Etter avtale med kunde ble første produksjonssetting kjørt den 12.mai for avdelingene MO.Kir.Ortopedisk og AL.Kir.Ortopedisk. Her ble det oppdaget at telefonnumrene kunne inneholde mellomrom. Dette blir et problem når numrene fra CW og PAS skal sammenlignes. Det måtte gjøres ny revisjon i koden og sørge for å bruke replace.(" ","") når numrene hentes.

Det ble også oppdaget at det noen steder var fylt inn fødselsnummer som telefonnummer. Det ble raskt konkludert med at det skal unngås å sende fødselsnummer til kø. Det ble lagt til som et nytt scenario i dokumentasjonen, scenario 7. Koden ble revidert slik at kolonnen for respondent nummer i Dispatcher krypteres, og deretter dekrypteres i Performer. I Performer gjøres også en kontroll av nummeret og erstatter de fem siste tallene med * ved mistanke om fødselsnummer.

Resultatene fra første produksjonskjøring er vist i Tabell 1. Roboten kontrollerte totalt 1767 respondenter og oppdaterte telefonnummeret 67 av disse.

Resultat	Antall	Type feil/log
Totalt antall respondenter funnet	1767	Log: Info
Antall respondenter med ulikt nummer i PAS og CheckWare som skal sendes til kø	67	

Tabell 1 Resultat fra første kjøring for MO.Kir.Ortopedisk og AL.Kir.Ortopedisk

Vurdering av type ulikhet av alle elementer som sendes til kø	Antall	Prosent
Respondent mangler tlfnr i CW / Scenario 2	39	2,2
Antatte feilføringer	11	0,6
Tilfeller der det antas at det er nytt nummer i PAS	17	1,0

Tabell 2 Analyse av tilfellene som sendes til kø.

Ved nærmere analyse av de tilfellene der det er funnet uoverensstemmelse mellom nummeret i PAS og CheckWare, gjaldt det for de fleste at det manglet telefonnummer i CheckWare, altså Scenario 2.

Den antatte menneskelige feilmarginen var anslått til ca 2% i prosessvurderingen, og ser ut til å stemme veldig bra. Samlet sett er det 2,2 % med blankt nummer og 0,6 % med feilinnføringer(F.nr eller trykkfeil), som samlet sett blir om lag 3 %. Feil inntasting eller feilføringer unngås ved bruk av robot. Denne typen oppgaver er godt egnet til automatisering. Konsekvensene ved feiltastinger er at respondenten ikke kan nås ved tjenester via CheckWare og at eventuelle forberedelser til time ikke blir gjort digitalt. Eller at SMS sendes til feil person.

14.0 Konklusjon

Utplasseringen i Hemit ledet frem til produksjonssetting av en unattended software robot som gjennomfører vask av telefonnumrene av respondenter i CheckWare to ganger i døgnet. Gevinsten av denne roboten er spart tid i form av at helsepersonell ikke trenger å kontrollere telefonnummeret før utsendelse av kartleggingsskjema. Samt at manglende utfylt skjema hos pasient ikke fører til utsettelse av timer, merarbeid og videre forsinkelser.

Innledningsvis i denne oppgaven ble det referert til kilde (Farsund Ulltang & Knappskog, 2022) som omtaler RPA som en lettbeint IT-løsning. I tilfellet for denne

CheckWare prosessen var alle tilganger og rettigheter for roboten på plass, og selve utviklingsfasen for roboten var kun et par uker.

Som utvikler oppleves det som enormt motiverende å se resultater raskt. Å få bidra til å hjelpe et allerede presset helsevesen gjør at jobben oppleves utrolig meningsfylt. I vurdering av kandidater er en viktig faktor «spart tid» for helsepersonell. De jobbene som egner seg til å automatiseres er ofte repeterende, langtekkelige og lite motiverende. For å beholde helsearbeidere er det viktig å fjerne oppgaver som er belastende og tidkrevende, og som en robot kan gjøre mye raskere. RPA kan dermed bidra som et ledd i det å forbedre arbeidshverdagen til de ansatte. Basert på rapporten til helsepersonellkommisjonen som ble levert tidligere i år kan RPA være en del av løsningen for å møte utfordringene i helse- og omsorgstjenestene.

Testfase med gjennomgang av koden (Code review) tok nesten like lang tid som utviklingsfasen. Siden dette er min første prosess gikk det mye tid til opplæring i denne fasen. Det var også her vi forstod at det var viktig å kunne filtrere ut respondenter basert på avdeling og hvilken instans (HF) respondenten tilhører. Det ble gjort flere revisjoner av logging og koden i denne fasen.

Etter første produksjon setting ble det oppdaget feilføringer der blant annet fødselsnummer var satt som telefonnummer. Det er nettopp denne typen manuelle arbeider som med fordel kan erstattes av en robot for å eliminere typiske menneskelige avvik eller feil. Første kjøring i produksjon gikk også helt uten registrerte systemfeil, eller businessfeil.

Oppsummert bidrar automatiseringen til innsparing av tid, heving av kvalitet ved at direkte feilinnføringer unngås, og at de ansatte slipper å bruke tid på kjedelig, repeterende arbeid.

15.0 Referanser

Anders Jensen. (2023, April 20). Hentet fra https://andersjensen.org/the-best-uipath-resources-2021/

Datatilsynet. (2023, Mai). Hentet fra https://www.datatilsynet.no/rettigheter-og-plikter/personopplysninger/fodselsnummer/

Departementenes sikkerhets- og serviceorganisasjon. (2023). *Tid for handling - Personellet i en bærekraftig helse- og omsorgstjeneste.* Hentet fra https://www.regjeringen.no/no/dokumenter/nou-2023-4/id2961552/?ch=13

Digital Workforce. (2023). *Digitale Medarbeidere: Robotisert Prosessautomasjon*. Hentet fra https://digitalworkforce.com/no/digitale-medarbeidere/robotisert-prosessautomasjon/

Digitaliseringdirektoratet. (2023, April 27). Hentet fra kontakt-og-reservasjonsregisteret: https://eid.difi.no/nb/kontakt-og-reservasjonsregisteret

Farsund Ulltang, F., & Knappskog, J. N. (2022). *Robotisert prosessautomatisering som bidrag til digitaliseringen av helse- og omsorgssektoren.* Fakultet for medisin og helsevitenskap, Institutt for nevromedisin og bevegelsesvitenskap. NTNU.

Forbes. (2020). *ForbesAI50*. Hentet fra https://www.forbes.com/sites/alanohnsman/2020/07/03/ai-50-americas-most-promising-artificial-intelligence-companies/

Helse- og omsorgsdepartementet. (2023, Februar 02). *Regjeringen.no: Pressemelding.* Hentet fra https://www.regjeringen.no/no/aktuelt/helsepersonellkommisjonen-overleverte-sin-utredning/id2961748/

Hemit. (2023). RPA-kandidat: CheckWare.¨

Hemit HF. (2023, Mai 10). Hemit, om oss. Hentet fra https://hemit.no/om-oss-hemit

Hemit HF. (2023, Januar 20). *Hemit, nyheter*. Hentet fra https://hemit.no/nyheter/-frigjor-mer-tid-til-andre-spennende-arbeidsoppgaver

Skaara, R., & Hertzenberg, M. (2022). *En økonomisk organisasjonsanalyse av automatiserte forretningsprosesser levert av HEMIT HF til Helse-Midt Norge RHF.* Copenhagen Business School.

UiPath. (2023, Januar 18). Hentet fra https://www.uipath.com/company/about-us

UiPath. (2023, Mars). Hentet fra https://www.uipath.com/product/low-code-app-studio

UiPath. (2023). *Academy: Courses: Introduction to RPA and Automation*. Hentet fra https://academy.uipath.com/courses/introduction-to-rpa-and-automation

UiPath. (2023). *UiPath Documentation StudioX*. Hentet fra https://docs.uipath.com/studiox/standalone/2023.4/user-guide/introduction

UiPath. (2023). *UiPath Product Studio Web*. Hentet fra https://www.uipath.com/product/studio-web

UiPath. (2023). *UiPath: What is RPA?* Hentet fra https://www.uipath.com/rpa/robotic-process-automation

VEDLEGG 1 Prosessvurderingsskjema Oppdatere tlfnr for respondenter i CheckWare

Prosess Vurdering Skjema

Virksomhet:	*HMR*
Avdeling:	*IKT seksjonen*
Prosessnavn	*Oppdatere tlfnr respondenter i CheckWare*

Type	Spørsmål	Beskrivelse	Svar
PROSESS FORBEDRING	Hvor effektiv er dagens prosess?	En rating mellom 1 - 10	4
	Hvor effektiv tenker en robotisert prosess kan være?	En rating mellom 1 - 10	9
BESLUTNINGSTYPE	Er beslutningene **regelbaserte** eller **subjektive / strategiske?**		Bare Regelbasert
TYPE INNDATA	Hvordan ser de **fleste** type inndata ut?	**Digital**: alle data som kan lagres på en maskin [server, bærbar PC, IOT-enheter] **Strukturerte data**: har et forutsigbart format, eksisterer i faste felt (f.eks. En celle i Excel eller et felt i et skjema) og er lett å oppdage via søkealgoritmer. Tenk på strukturerte data som: regneark, tabeller med fast struktur, databaser hvor du enkelt kan trekke ut informasjonen din gjennom en søkefunksjon. **Ustrukturerte data**: kan ha sin egen interne struktur, men dette er ikke veldig forutsigbart. Eksempler: nettlogger, multimediainnhold, kundeservice-interaksjoner og data fra sosiale media.	Digital og Ustrukturert
PROSESS-STABILITET	Hvordan vil prosessen din endres de neste **6 månedene**?	Tenk på endringer av følgende type: - lovendringer som kommer og kan endre reglene / logikken i prosessen. - organisatoriske endringer som å fordele deler av prosessene mellom forskjellige roller / avdelinger.	Ingen endring forventet
APPLIKASJONS-STABILITET	Hvordan vil applikasjonene dine endre seg i løpet av de neste **6 månedene?**	Ikke tenk bare på applikasjoner, men også programmeringsgrensesnitt, menyer i apper og rapporter.	Ingen endring forventet

HOVEDRESULTAT (Gjennomførbarhet SCORE)	Gjennomførbar

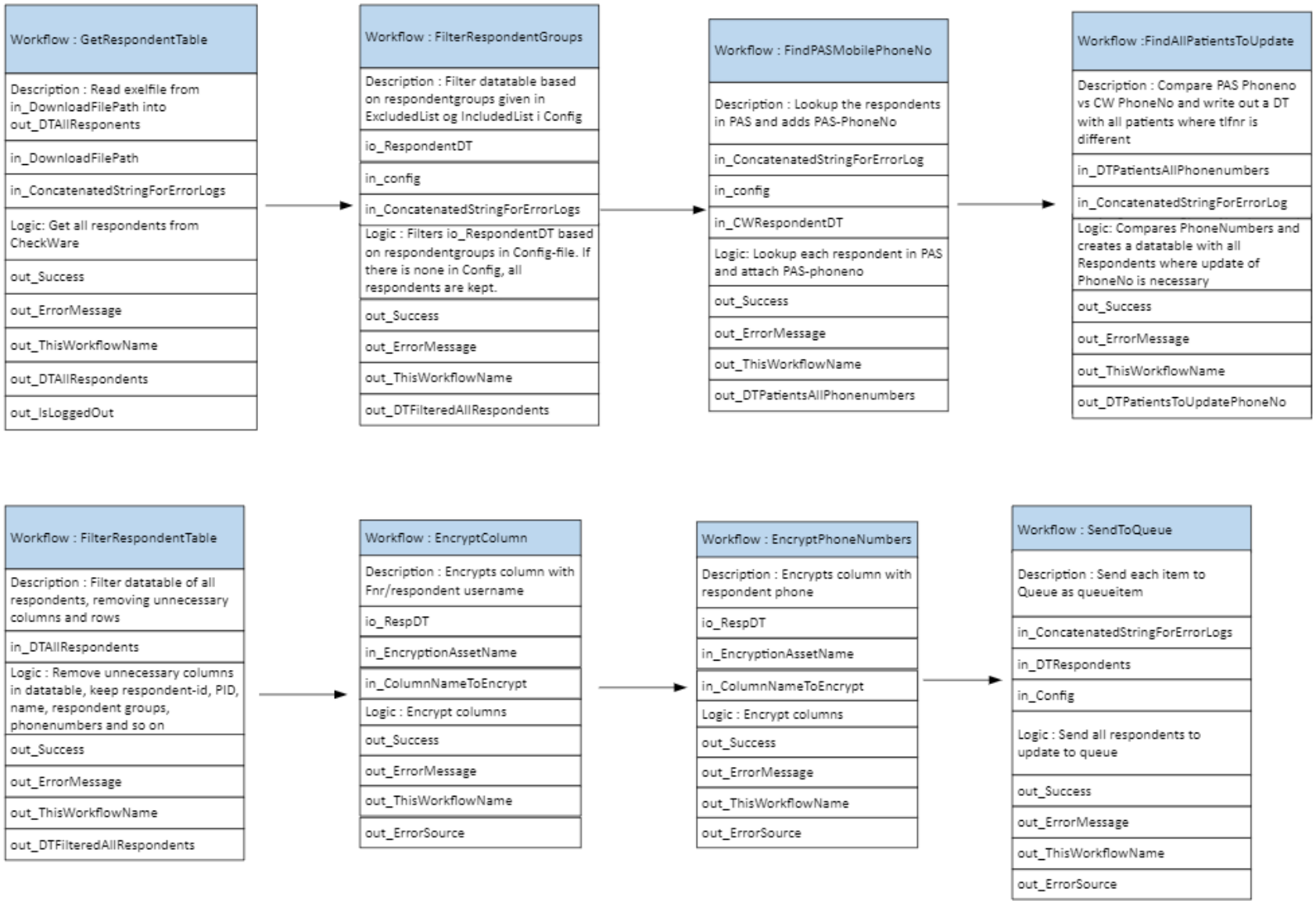
GEVINST	Hvor mange **årsverk** er det nå for å utføre prosessen?	Valgfritt felt	
	Hva er **frekvensen** av prosessen?		Daglig
	Hva er **volumet på transaksjoner** / frekvens (antall ganger prosessen kjøres / valgt frekvens)?	F.eks Hvis prosessen skjer ukentlig, må vi vite volumet som skjer i gjennomsnitt i løpet av en uke.	500
	Hva er den **gjennomsnittlige tiden det tar for prosessen å kjøres en gang** (gjennomsnittlig håndteringstid / transaksjon)?	Vennligst kvantifiser gjennomsnittlig håndteringstid i minutter, f.eks. hvis 4 timer, sett inn 240	1
	Hva er gjennomsnittlig antall menneskelige feil?		2 %
	Er det nok kapasitet å fullføre transaksjoner?		Ja
	Hvordan vil du karakterisere **toppene** i prosessen?		Prosessen har ikke topper
PROSESSKOMPLEKSITET	Hvor mange **trinn** har prosessen?		10-15 steg
	Hvor **vanskelige** er beslutningene du må ta for å fullføre prosessen?		Prosessen innebærer enkle beslutninger (ja / nei-type)
	Hva er gjennomsnittlig antall tilfeller der du ikke klarer å fullføre hele prosessen? (Enten fordi du trenger innspill fra en annen person, eller fordi du havner i en situasjon som ikke er dekket av en klar regel)		1 %
	Hva er **antall applikasjoner** du bruker for prosessen?		2-3 applikasjoner
	Trenger du tilgang til noen av applikasjonene via **CITRIX/VDI**?		Nei
INNDATA	Hvilken **% av input data** dine er digitale?		100 %
	Er det noen av de digitale **data dine som er skannet**?		Nei
	Hvilken **% av input data dine er strukturert**?	Strukturerte data: har et forutsigbart format, eksisterer i faste felt (f.eks. En xls-celle eller et felt i en form) og er lett å oppdage via søkealgoritmer. Tenk på strukturerte data som: utmerker, tabeller med fast struktur, databaser hvor du enkelt kan trekke ut informasjonen din gjennom en søkefunksjon. Eksempler: Hvis innspillene består av 5 Excel-filer / eller ERP-rapporter, er 100% av dataene dine strukturert, mens hvis innspillene består av 4 Excel-filer / ERP-rapporter og 1 e-posttekst, er 80% av dataene strukturert.	>= 80%

Implementeringstiltak	11 %
Fordel / Egnethet	97 %

Estimat Spart (timer/år)	2 102
Implementeringstiltak (timer)	216

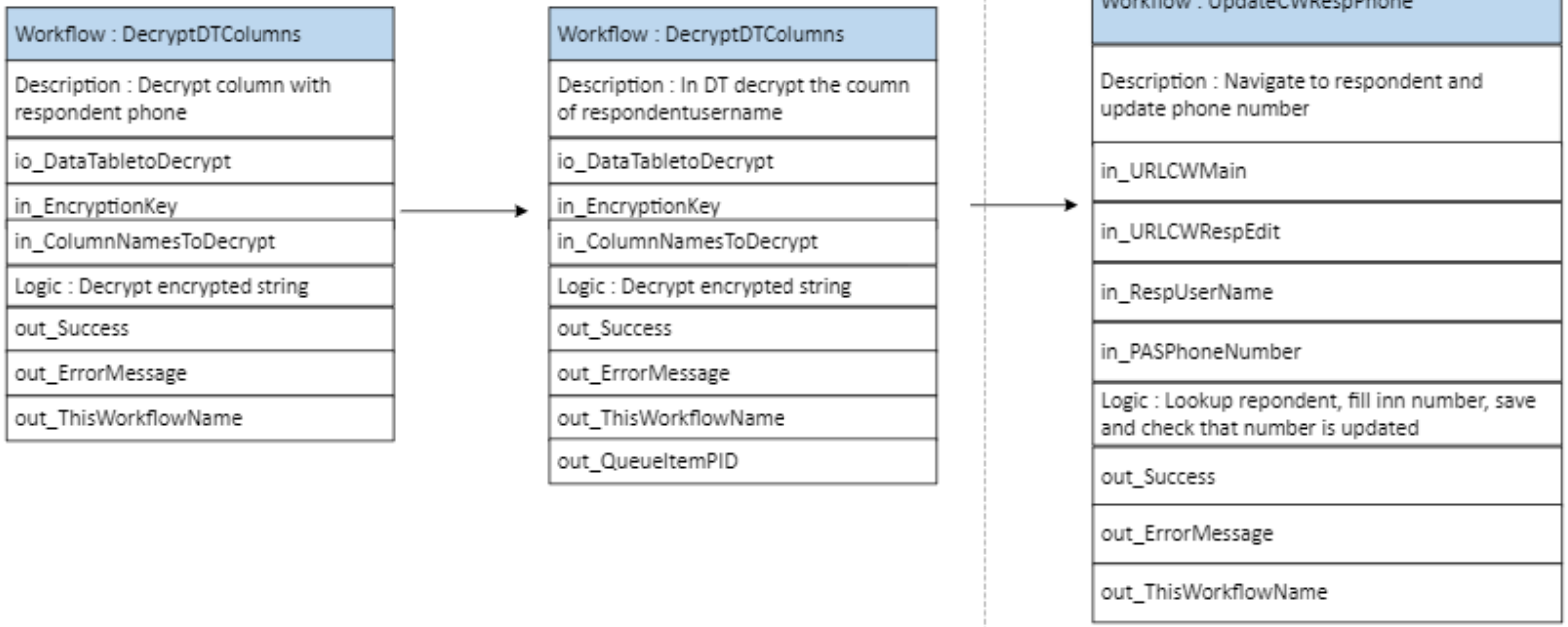
VEDLEGG 2 DISPATCHER CheckWare Tlf.Vask

DISPATCHER DESIGN
Send til kø



VEDLEGG 3 PERFORMER CheckWare Tlf.Vask

PERFORMER DESIGN
Håndtering av hver transaksjon



VEDLEGG 4

Logg over Utplassering i Hemit

Møteoversikt

10.01.23 Prosessvurdering Arkiveringskandidat-kandidat

Gjennomgang av prosess med prosesseier. Arbeidsoppgavene som ønskes automatisert går ut på å journalføre og kvalitetssikre inngående journalposter.

16.01.23 Prosessvurdering RPA kandidat HelsaMi

Gjennomgang av prosess med prosesseier. Arbeidsoppgavene som ønskes automatisert går ut på å tildele foreldretilgang i HelsaMi

25.01.23 Prosessvurdering RPA-kandidat SAP – Vask av leverandørregister

Gjennomgang av prosess med prosesseier. Arbeidsoppgavene som ønskes automatisert går ut på å oppdatere og vaske leverandørregisteret i SAP.

01.02.23 Grunnkurs informasjonssikkerhet og personvern i Hemit, 3 timers kurs

Obligatorisk grunnkurs i Hemit

07.01.23 Møte angående utviklingsprosjekt Virusdyrkning Maskinlæring

17.02.23 Prosessvurdering RPA-kandidat: Arkivering av kandidatrapport fra Webcruiter

Gjennomgang av prosess med prosesseier. Arbeidsoppgavene som ønskes automatisert går ut på å lagre informasjon om nyansatte i arkivet.

23.02.23 Kurs i Elements sak og arkivsystem

28.02-01.03 RPA Workshop i Molde.

RPA teamet samles i Molde for interne kurs og gjennomganger

03.03.23 Prosessvurdering RPA kandidat: SAP – Opplast av bilag og vedlegg

Gjennomgang av prosess med prosesseier. Arbeidsoppgavene som ønskes automatisert går ut på å lagre informasjon om nyansatte i arkivet.

10.03.23 Møte med prosesseier om RPA kandidat CheckWare

Møte med prosesseier som automatisering av opprettelse av respondent i CheckWare. Etter gjennomgang ble prosjektet delt inn i tre ulike deler, hvorav den ene var Vask av telefonnummer for respondenter i CheckWare.

28.03.23 Kort avklaringsmøte med prosesseier om CheckWare Tlf.Vask

Avklaringer med prosesseier rundt scenarioer, og hvordan roboten skal håndtere de ulike scenarioene.

12 - 13.04 Møte på Mikrolaboratoriet på St.Olavs angående utviklingsprosjektet Virusdyrkning

Befaring til mikrolaboratoriet for å se nærmere på hvordan billedtagning og vurdering av virusprøver utføres.

10.05.23 Gjennomgang av RPA-kandidat CheckWare Respondent Tlf.Vask før Prod-setting

Gjennomgang med prosesseier. Avklaringer rundt når første produksjonssetting skal gjøres, hvem som skal motta logger fra Splunk hos kunde m.m.

11.05.23 Prosessvurdering RPA kandidat: AirView – Etterregistrering av koder ved fjernmonitorering

Gjennomgang av prosess med prosesseier. Arbeidsoppgavene som ønskes automatisert går ut på å oppdatere koder i på alle pasienter som er under fjernmonitorering.

Tidsbruk i prosjekt CheckWare Respondent Tlf.Vask

2023

Anne Mari

UkeNr	År	Prosjekt	Fase	Timer	Kommentar
10	2023	Opprette respondent i CheckWare	Møter	5	
10	2023	CheckWare Respondenter Vask tlfnr	Design	3	
11	2023	CheckWare Respondenter Vask tlfnr	Utvikling	5	
12	2023	CheckWare Respondenter Vask tlfnr	Utvikling	15	
13	2023	CheckWare Respondenter Vask tlfnr	Utvikling	20	
15	2023	CheckWare Respondenter Vask tlfnr	Utvikling	5	
16	2023	CheckWare Respondenter Vask tlfnr	Utvikling	20	
16	2023	CheckWare Respondenter Vask tlfnr	Test	5	
17	2023	CheckWare Respondenter Vask tlfnr	Test	25	
19	2023	CheckWare Respondenter Vask tlfnr	Test	15	
19	2023	CheckWare Respondenter Vask tlfnr	Møter	3	
20	2023	CheckWare Respondenter Vask tlfnr	Test	10	

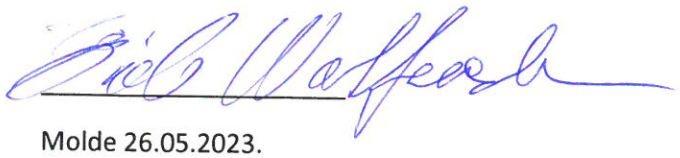
Møter:	8 timer
Design:	3 timer
Utvikling:	65 timer
Test:	50 timer

Bekreftelse på gjennomført utplassering

(vedlegg inn i sluttrapporten)

Dette er en bekreftelse fra bedriften Hemit HF ved veileder Eirik Wolfenstein på at studenten Anne Mari Farstad har vært utplassert i vår bedrift i perioden 01.01.2023 til 31.05.2023 .

Vi bekrefter at hun har utført det som er beskrevet i sluttrapporten.



Molde 26.05.2023.

(signert av veileder/bedriftsrepresentant)