UNIVERSITEIT GENT

FACULTY OF ENGINEERING AND ARCHITECTURE

Universiteit Gent

Vakoverschrijdend project: denul1

Business cards: sprint 1

Projectmanager: prof. Naessens
Coach: prof. dr. Cnops & prof. dr. Ongenae

Willem Dendauw
Jasper Malfait
Tibo Vandercruyssen
Yentl Vermeulen

12-3-2021

België heeft veel grote bedrijven, slechts weinig van die bedrijven hebben zo een grote impact over de hele wereld als Jan De Nul Group. Een grote omvang hebben, heeft ook zijn nadelen. Het verzamelen en beheren van gegevens is vaak een heikelpunt bij grote ondernemingen. Bijhouden van gegevens van partners en leveranciers is hier geen uitzondering op. Dit project beoogt het stroomlijnen en automatiseren van dit proces. Dit steunt hier volledig op het herkennen van karakters op foto's van visitekaarten. Eenmaal deze gegevens herkend zijn, moeten ze correct gelabeld worden. Een centrale databank moet deze gegevens bijhouden en ter beschikking stellen voor iedereen die er gebruik van wenst te maken. Concreet betekent dit dat er een applicatie wordt ontwikkeld die de gebruiker in staat stelt om aan de hand van een foto snel gegevens toe te voegen aan de databank.

Inhoud

Lijst van figuren

Lijst van tabellen

Inleiding

Context

Jan De Nul Group is één van de grootste baggerbedrijven ter wereld (Jan De Nul Group, 2009). Het Belgisch bedrijf met hoofdzetel in Hofstade, een deelgemeente van Aalst, werd opgericht in 1938. Deze onderneming wordt momenteel geleid door Jan Pieter De Nul en heeft reeds vele bagger- en saneringsprojecten uitgevoerd, hoofdzakelijk in het buitenland. Het bedrijf focust zich vooral op het opwekken van energie op zee, het voldoende diep houden van waterwegen en het aanpakken van verontreiniging. Het bouwen van nieuwe havens en landwinning behoren ook tot de kerntaken.

Jan De Nul Group bezit een grote vloot, bestaande uit verschillende soorten schepen. Daarnaast worden ook projecten aan land verwezenlijkt. Hierdoor behoort rollend materieel zoals kranen, bulldozers en graafmachines ook tot het arsenaal. Zowel het ontwerpen, onderhouden en herstellen van het materieel wordt door Jan De Nul Group zelf verwezenlijkt.

Jan De Nul Group werd onder meer bekend door het bouwen van sluizen in het Panamakanaal, het aanleggen van pijpleidingen in zee en de berging van schepen. De Palmeilanden voor de kust van Dubai (zie Figuur 1), verdiepings- en onderhoudsbaggerwerken in de Rio de la Plata in Argentinië en het bouwen van het windmolenpark Nobelwind (Windmolenpark Nobelwind, z.j.) staan ook op het palmares van dit bedrijf.

Figuur 1: Palmeilanden bij Dubai (JanDeNul, 2002)

Probleemstelling

Aangezien Jan De Nul Group een groot bedrijf is met veel internationale projecten, komt het vaak in contact met veel andere bedrijven, leveranciers en partners. Onderling worden vaak relevante contactgegevens uitgewisseld in de vorm van een visitekaart, waarvan een voorbeeld weergegeven is in Figuur 2.

Momenteel wordt deze kaart na ontvangst doorgegeven aan een administratief medewerker om deze gegevens manueel in te voeren in de databank. Dit proces neemt echter veel tijd in beslag en op die manier kan een visitekaart ook verloren gaan. Binnen het bedrijf wordt reeds de applicatie Haystack gebruikt. Deze app is echter onvoldoende bruikbaar omdat alle gegevens opgeslagen worden op servers van Haystack zelf. Hierdoor kan de informatie onmogelijk op de bedrijfsdatabase van Jan De Nul Group zelf gestockeerd worden. Het gebruik van een externe databank voor het uitlezen van een fysieke kaart, zorgt binnen het bedrijf voor problemen.

Eenvoudiger zou zijn om de gegevens afkomstig van alle visitekaarten in de databank van het bedrijf zelf op te slaan. De ontwikkeling van een app die een foto neemt van

een fysieke bedrijfskaart kan de oplossing zijn. Nadien wordt de informatie van de kaart, die door het OCR-framework wordt ingelezen, ingevuld in de applicatie. Vervolgens bestaat de mogelijkheid om eventuele fouten aan te passen of ontbrekende data in te vullen. Ten slotte kunnen deze gegevens verstuurd worden naar de databank enerzijds en via mail naar de administratie anderzijds.

Figuur 2: Voorbeeld visitekaart

Doelstellingen

Het eindproduct van dit project is een mobiele applicatie die zowel op Android- als iOS-apparaten werkt. Om deze doelstelling te realiseren wordt er gebruik gemaakt van het Xamarin-framework. Dit framework zorgt ervoor dat een applicatie kan worden ontwikkeld voor meerdere platformen tegelijk (Microsoft, 2019).

De applicatie dient de gebruiker in staat te stellen om het verwerven van gegevens van een visitekaart te versnellen en te automatiseren. Dit gebeurt door die gegevens automatisch uit een foto te halen en in de correcte structuur om te zetten. Aan de hand van een *optical character recognition*-framework (OCR-framework) wordt het lezen van tekstuele gegevens uit een foto wordt verwezenlijkt (Optical character recognition, 2004). Daarnaast moet de applicatie de mogelijkheid bieden om deze gegevens door te sturen naar een databank van Jan De Nul Group en optioneel naar een e-mailadres. Dit is een belangrijk aspect voor dit project. Het is echter cruciaal dat gegevens die verworven worden binnen Jan De Nul Group niet verspreid staan over servers van derden.

Werknemers van Jan De Nul Group zijn vaak niet verbonden met een netwerk. Hierdoor dient er ook functionaliteit aanwezig te zijn om de data op het toestel zelf te bewaren, zolang er geen netwerkverbinding is.

Opbouw

In het eerste hoofdstuk van dit verslag wordt het project beschreven vanuit het standpunt van de gebruiker. Die gebruiker zal een werknemer zijn van de Jan De Nul Group, die de informatie van een businesskaart wil opslaan op een efficiëntere manier dan puur manuele invoer. Dit gebeurt aan de hand van een applicatie die de gegevens automatisch uit foto's kan halen. Ook de verschillende features die de applicatie aanbiedt worden in dit deel beschreven. Vervolgens wordt project bekeken vanuit het standpunt van de ontwikkelaar. Dit omvat verdere uitleg van hoe de applicatie opgebouwd is. Aan de hand van verscheidende diagrammen wordt deze uitleg verder verduidelijkt. (Hier moet dan nog aangevuld worden voor volgende delen die voor het volledige verslag verwacht worden (moet dit nu al gedaan worden of niet)

1 Gebruikersaspecten

1.1 Beschrijving opdracht

Momenteel worden alle ontvangen visitekaarten door het personeel aan een administratief medewerker gegeven. Deze medewerker geeft dan handmatig alle gegevens in op een computer en stuurt ze door naar de centrale databank van Jan De Nul Group. Ze willen dit proces sneller, gemakkelijker en efficiënter maken met behulp van een mobiele applicatie, geïnstalleerd op de smartphone van elke werknemer binnen het bedrijf.

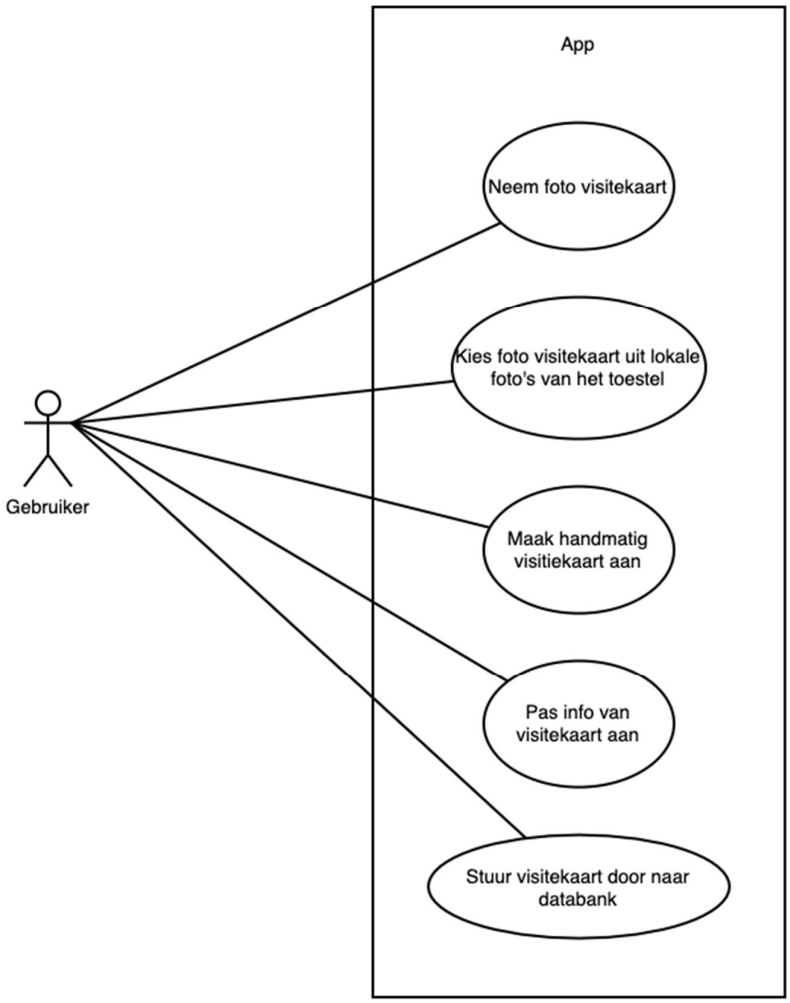
Het is de bedoeling dat de gebruiker na het ontvangen van een visitekaart zelf de gegevens ervan kan opslaan en doorsturen naar de centrale databank van Jan De Nul Group. De manier waarop dit gebeurt, wordt hieronder beschreven en verduidelijkt met behulp van het *use-case*-diagram (zie Figuur 3).

De gebruiker kan na het openen van de applicatie kiezen uit drie manieren om een visitekaart op te slaan. Ten eerste is het mogelijk om een foto te nemen van een visitekaart. Ten tweede kan de gebruiker een reeds genomen foto van een visitekaart selecteren vanop zijn toestel. Ten laatste is het mogelijk om zonder foto te werken als er enkel informatie beschikbaar is over een persoon zonder visitekaart. Bij de eerste twee opties wordt OCR gebruikt om alle nuttige gegevens uit de foto te halen en in te vullen in de bijbehorende velden. De gebruiker heeft nadien nog de mogelijkheid om de automatisch ingevulde velden te corrigeren. Bij de derde optie vult de gebruiker alle velden handmatig in. Eenmaal alle velden correct zijn ingevuld kan de gebruiker de visitekaart opslaan. Indien de gebruiker verbonden is met het internet, wordt de zonet opgeslagen visitekaart doorgestuurd naar de databank. Als de visitekaart correct is ontvangen, krijgt de gebruiker een melding van ontvangst. Indien de gebruiker niet verbonden is met het internet, wordt de visitekaart lokaal op de smartphone opgeslagen. Eens de gebruiker een internetverbinding heeft, worden alle lokaal opgeslagen visitekaarten doorgestuurd naar de databank en krijgt de gebruiker een melding van ontvangst.

1.2 Gewenste software- en hardware vereisten

Jan De Nul Group wenst een mobiele applicatie dat een visitekaart kan inscannen en de belangrijke gegevens ervan kan doorsturen naar hun centrale databank. Het moet ook mogelijk zijn om de visitekaart lokaal op te slaan, gebruik makend van SQLite-databank, totdat er internetverbinding is. Een belangrijk aspect is dat alle informatie binnen het bedrijf blijft. Deze applicatie moet zowel een Android- als een iOS-versie hebben. De ontwikkeling hiervan gebeurt daarom via het Xamarin.Forms-framework.

1.3 *Use-case*-diagram



Figuur 3: Use-case-diagram

Naam
Jan De Nul Group: Businesscard App

Doelstelling
Een gebruiker kan een visitekaart inscannen en de gegevens doorsturen naar de centrale databank van Jan De Nul Group

Primaire actor
Gebruiker

Precondities
Gebruiker heeft de app opgestart

Postcondities
- Gegevens van visitekaart werden doorgestuurd
- Gegevens van visitekaart werden lokaal opgeslagen

Successcenario
1. Gebruiker neemt foto van visitekaart.
2. Gebruiker corrigeert automatisch ingevulde gegevens van visitekaart.
3. App verzendt visitekaart naar databank.
4. Gebruiker ontvangt melding van ontvangst.

Alternatieve scenario's
1.a. Gebruiker selecteert eerder genomen foto van visitekaart, ga naar stap 2.
1.b. Gebruiker heeft geen visitekaart, enkel info van een persoon, ga naar stap 2.
3.a. Indien gebruiker niet verbonden is met het internet worden de gegevens van visitekaart lokaal opgeslagen om later door te sturen.

1.4 Featurelijst

- Foto nemen visitekaart
- Foto van visitekaart selecteren uit galerij
- Gegevens visitekaart inlezen met OCR
- Gegevensvelden automatisch laten invullen door OCR
- Gegevensvelden handmatig corrigeren

- Mail verzenden met de gegevens van ingelezen visitekaart als vcf-bestand
- Gegevens visitekaart in json-formaat automatisch doorsturen naar centrale databank via http-*request*
- Http-*response* voor bevestiging van ontvangst of melding van falen
- Lokaal opslaan van visitekaarten indien er geen internetverbinding is
- Automatisch versturen van lokaal opgeslagen visitekaarten vanaf er internetverbinding is, inclusief melding van succes of falen
- Meegeven van informatie over de zender van de visitekaart

1.5 Geselecteerde features per sprint

1.5.1 Sprint 1

- Foto nemen visitekaart
- Foto selecteren van visitekaart opgeslagen op smartphone gebruiker
- Gegevensvelden handmatig corrigeren
- Mail verzenden met de gegevens van ingelezen visitekaart als vcf-bestand
- Lokaal opslaan van visitekaarten indien er geen internetverbinding is

1.5.2 Sprint 2

- Gegevens visitekaart inlezen met OCR
- Gegevensvelden automatisch laten invullen door OCR
- Meegeven van informatie over de zender van de visitekaart

1.5.3 Sprint 3

- Gegevens visitekaart in json-formaat automatisch doorsturen naar centrale databank via http-*request*
- Http-*response* voor bevestiging van ontvangst of melding van falen
- Automatisch versturen van lokaal opgeslagen visitekaarten vanaf er internetverbinding is, inclusief melding van succes of falen

2 Systeemarchitectuur

2.1 *High-level*systeemmodel

Het grote voordeel van mobiele applicaties is dat deze gemaakt zijn voor het efficiënt uitrollen van het systeem op een toestel. Een gebruiker kan de applicatie verkrijgen door naar de bijpassende appstore van het toestel te gaan. Aangezien deze app specifiek voor het bedrijf Jan De Nul Group gemaakt is, zal de applicatie niet beschikbaar zijn op een publieke appstore, maar enkel binnen het bedrijf. Bij het downloaden van een applicatie zitten alle bibliotheken waarvan het gebruik maakt in het downloadbestand. Applicaties en software die op het toestel moeten staan voor het gebruik van het programma worden gecontroleerd voorafgaand op de download en kunnen niet in combinatie gedownload worden met de applicatie. Hierdoor is er geen *deployment*-diagram voorzien.

De applicatie is opgebouwd via Xamarin.Forms. Dit is een *open-source*-framework dat gebruikt wordt voor het bouwen van zowel iOS- als Android-applicaties. Het grote voordeel hiervan is dat de applicatie niet voor elk besturingssysteem afzonderlijk moet ontworpen worden.

De huidige applicatie maakt gebruikt van de Xamarin.Essentials-bibliotheek. Deze bibliotheek wordt gebruikt voor de connectie met de camera- en mailapplicatie. Hierdoor is een toestel vereist met minimum iOS 9 of de Android API level 11 waarop ook de camera- en mailapplicatie zijn geïnstalleerd.

2.2 Klassendiagram

Dit deel komt pas in sprint 2 aan bod.

2.3 Toestandsdiagram

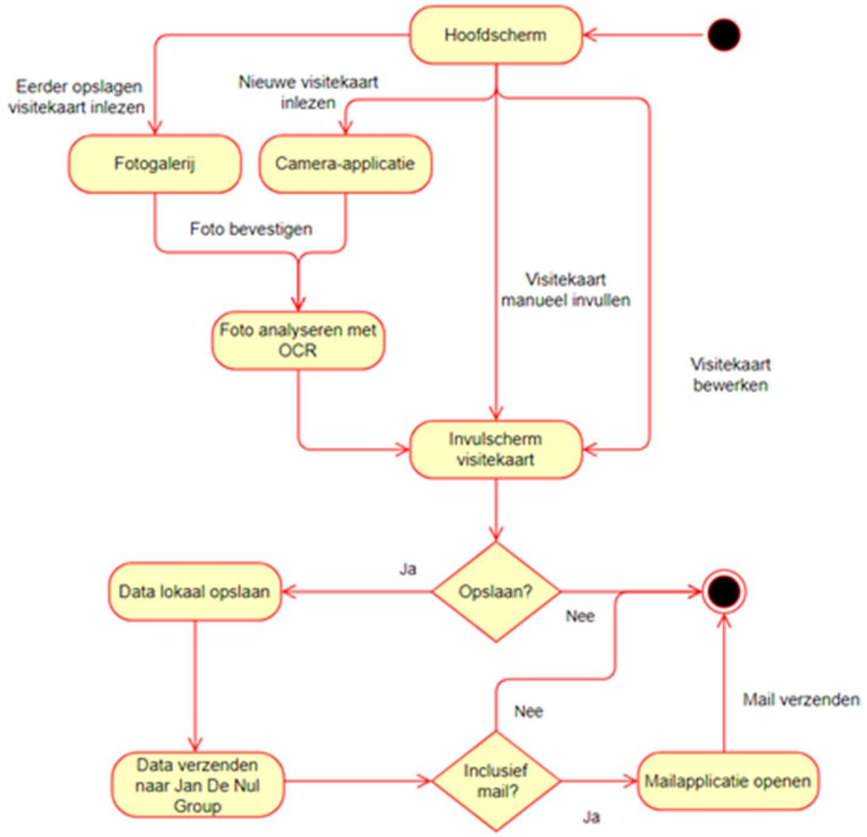
Het toestandsdiagram geeft de status gekoppeld aan de verschillende overgangen van een object weer. Op Figuur 4 wordt de visitekaartapplicatie weergegeven in een diagram.

Wanneer de applicatie opgestart wordt, zal de gebruiker automatisch terechtkomen op het hoofdscherm. Dit scherm bestaat uit drie knoppen met daarboven een lijst met reeds opgeslagen visitekaarten die nog niet zijn verzonden, maar wel nog

bewerkt kunnen worden. De knoppen kunnen gebruikt worden om naar een volgende toestand te gaan met als mogelijkheden: de camera-app te openen, de fotogalerij te openen of direct naar het invulscherm te gaan om een visitekaart manueel in te vullen. Indien één van de eerste twee opties gekozen wordt, zal de foto eerst via OCR-toestand passeren. Deze toestand zal de gegevens van de foto inlezen om deze na het filteren automatisch in te vullen in de velden van het eerder beschreven invulscherm. Hierdoor kunnen de data nog bewerkt worden door de gebruiker.

Alvorens de keuze van het opslaan of verwijderen gemaakt wordt, kan een selectievak worden aangekruist om aan te duiden of de visitekaart ook via mail moet verstuurd worden. Als de kaart vervolgens verwijderd wordt, dan zal de app terugkeren naar het hoofdscherm zonder dat de kaart verstuurd of opgeslagen is. Als er voor opslaan gekozen werd, dan worden de gegevens eerst lokaal opgeslagen om dataverlies te vermijden. Vervolgens zal de informatie doorgestuurd worden naar de databank van Jan De Nul Group.

Indien een beschikbare internetverbinding aanwezig is, zal de visitekaart succesvol worden doorgestuurd en niet meer lokaal opgeslagen worden. Bij het gebrek aan een internetverbinding wordt de visitekaart weergegeven in de lijst op het hoofdscherm. Ten slotte zal de mailapplicatie worden geopend alvorens er teruggekeerd wordt naar het hoofdscherm, wanneer het eerder beschreven selectievak is aangeduid. In deze app worden de gegevens van de visitekaart ingevuld in een standaardmail zodat de gebruiker enkel nog op de verzendknop zal moeten drukken.



Figuur 4: Toestandsdiagram

2.4 Sequentiediagram

De interactie tussen de verschillende modules waarvan de applicatie gebruik maakt, worden weergegeven in het sequentiediagram in Figuur 5. De modules kunnen bestaan uit verschillende standaardapplicaties die reeds op het toestel aanwezig zijn of databanken waarvan de visitekaartapplicatie gebruik maakt. De nadruk in dit diagram ligt op het berichtenverkeer in een bepaalde tijdsdimensie.

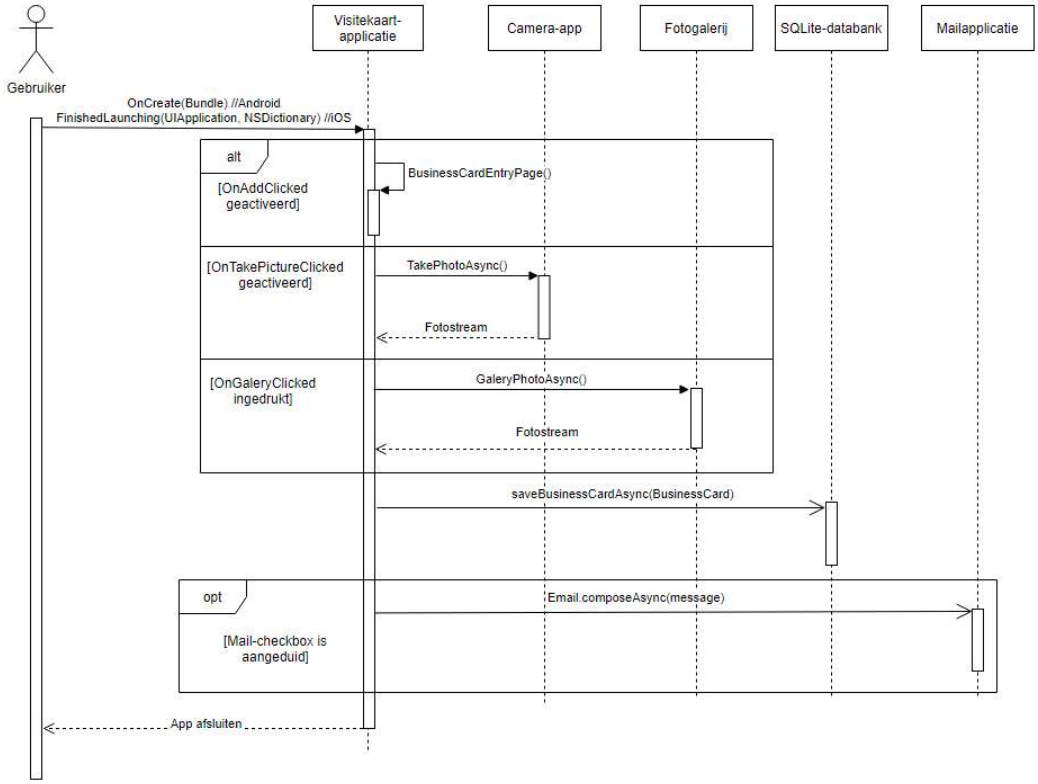
De werking van de app is afhankelijk van het gebruikte besturingssysteem, waardoor de gebruiker achterliggend twee mogelijke functies heeft om de applicatie op te starten: via de methode `OnCreate(Bundle)` voor Android of via de methode `FinishedLaunching(UIApplication, NSDictionary)` voor iOS. Eens de gebruiker zich in de visitekaartapplicatie bevindt, zijn er drie opties: visitekaart handmatig invullen, inlezen via camera of inlezen uit de fotobibliotheek. De eerste optie, het manueel invullen, is mogelijk wanneer de toevoegknop met als klikmethode `OnAddClicked` wordt ingedrukt. Deze roept vervolgens de methode

`BusinessCardEntryPage()` op waardoor de pagina voor het bewerken van een lege visitekaart gepresenteerd wordt. Als de tweede optie, inlezen via camera, gekozen wordt via de knop die verbonden is met de `OnTakePictureClicked` methode zal de asynchrone methode `TakePhoteAsync()` geactiveerd worden. Dit is een asynchrone methode, maar wordt weergegeven als een synchrone aangezien de visitekaartapplicatie zal wachten tot het een fotostream ontvangt. Deze *stream* kan leeg zijn indien de foto geannuleerd wordt in de camera-applicatie. Ten slotte is er de derde optie, inlezen uit de fotobibliotheek. Deze optie is gelijkaardig aan de tweede optie aangezien hier ook een fotostream wordt verwacht als antwoord. Het grote verschil is dat hier de knop verbonden met de methode `onGalleryClicked` wordt ingedrukt die de fotogalerij opent via de methode `GalleryPhotoAsync()`. Dit is terug een asynchrone methode waarbij het systeem synchroon wacht op een fotostream als antwoord.

Als een visitekaart aanwezig is in het systeem worden de gegevens hiervan opgeslagen via de methode `saveBusinessCardAsync(BusinessCard)`. Deze asynchrone functie, waar een visitekaart als parameter wordt meegegeven, zorgt ervoor dat alle gegevens lokaal op het toestel in een SQLite-databank worden opgeslagen.

De laatste module waar de applicatie mee kan interageren is de emailapplicatie. Deze optionele interactie is mogelijk door de methode `Email.ComposeAsync(message)`. Deze asynchrone methode wordt opgeroepen wanneer het selectievak `Mail-checkbox` is aangekruist bij het opslaan van een visitekaart. Het gevolg hiervan is dat, na het lokaal opslaan van de data, de *native* mailapplicatie van het toestel wordt geopend waarin een standaardmail voorzien van de gegevens wordt geladen. De gebruiker moet enkel nog op verzenden drukken. Indien de mailbox gesloten wordt zal het bericht worden opgeslagen als concept.

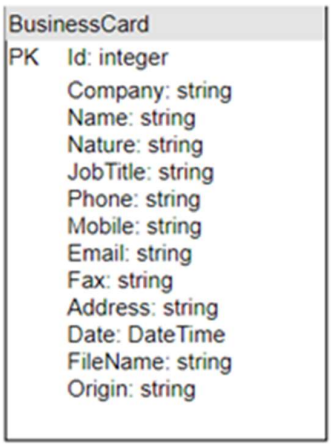
Een belangrijke opmerking bij het diagram is dat zowel het toepassen van OCR als de visitekaart naar het *endpoint* van Jan De Nul Group versturen nog niet is geïmplementeerd. De oorzaak hiervan ligt in het feit dat deze delen ook nog niet verwerkt zitten in de applicatie. Het diagram zal bij een latere sprint nog geüpdatet worden.



Figuur 5: sequentiediagram

2.5 Databank

Voor het lokaal opslaan van data maakt de applicatie gebruik van een SQLite databank. Dit was één van de opgegeven vereisten waaraan de applicatie moest voldoen. SQLite is een bibliotheek geschreven in de programmeertaal C die een kleine, snelle, betrouwbare SQL-databank-*engine* implementeert (What Is SQLite?, z.j.). De visitekaart wordt hierin opslagen aan de hand van het databankschema weergegeven in Figuur 6. De *Primary Key* wordt voorgesteld door de *integer* Id die automatisch verhoogd wordt bij een nieuwe visitekaart. De velden: Company, Name, Email en Date zijn verplicht. De databank zal dus geen visitekaart accepteren waarbij deze velden niet aanwezig zijn. Dit wordt opgevangen door een controle toe te voegen in het invulscherm van de applicatie.



Figuur 6: Databankschema

https://www.sqlite.org/index.html

Referentielijst

Jan De Nul Group. (2009). *Wikipedia.* Geraadpleegd op 8 februari 2021 via https://nl.wikipedia.org/wiki/Jan_De_Nul_Group

JanDeNul. (2002). Geraadpleegd op 3 maart 2021 via https://www.jandenul.com/about-us/history

Microsoft. (2019). Geraadpleegd op 8 maart 2021 via https://dotnet.microsoft.com/learn/xamarin/what-is-xamarin

Optical character recognition. (2004). *Wikipedia.* Geraadpleegd op 7 maart 2021 via https://nl.wikipedia.org/wiki/Optical_character_recognition

What Is SQLite?. (z.j.). Geraadpleegd op 7 maart 2021 via https://www.sqlite.org/index.html

Windmolenpark Nobelwind. (z.j.). Geraadpleegd op 10 maart 2021 via https://www.jandenul.com/nl/projecten/windmolenpark-nobelwind-belgie