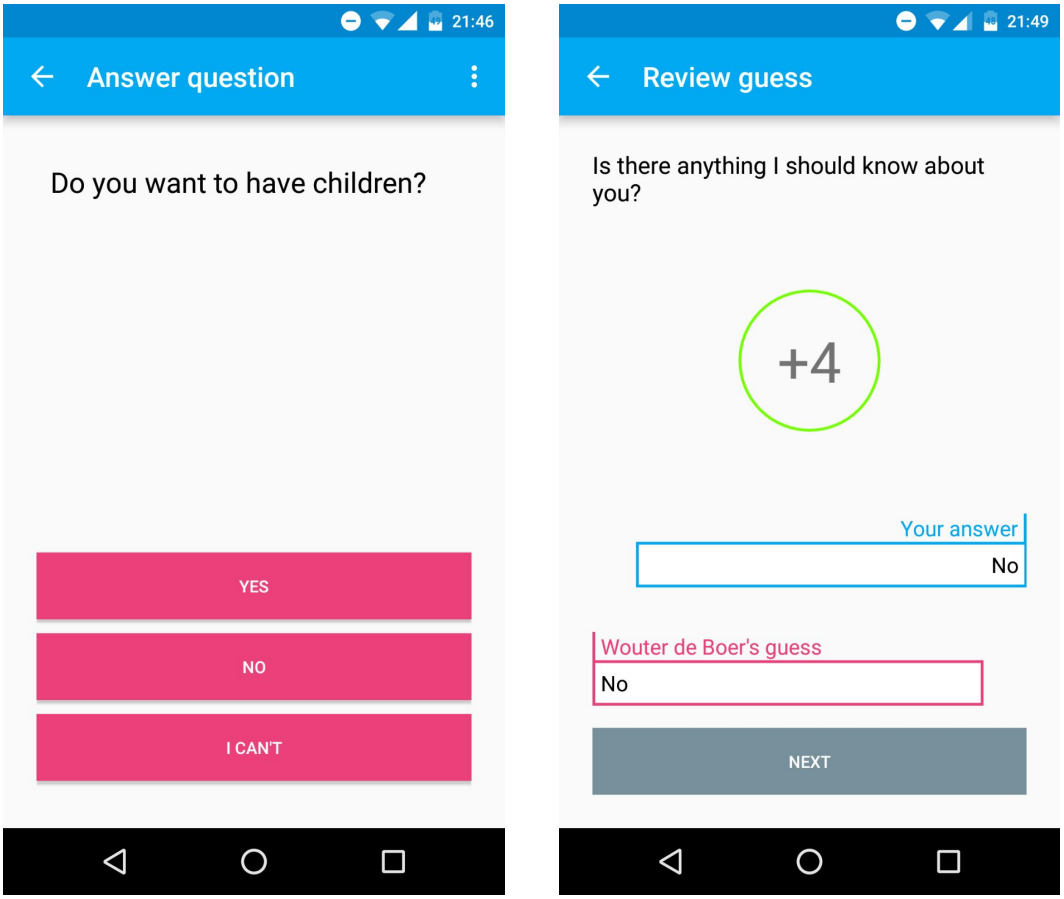
The Quest for "Quest for Questions"

Gia Linh Hoang - s4553519
Jelle Loman - s4573382
Timo Maarse - s4416295
Raka Schipperheijn - s4582721

June 17, 2016

1 Voorwoord

Dit verslag is geschreven in het kader van het vak Research en Development aan de Radboud Universiteit. Voor het vak moesten de studenten een applicatie maken die op Android kan draaien. Dit document is het eindverslag voor de applicatie "Quest for Questions".
Ten eerste zal de app worden beschreven, daarna zal het ontwerp worden besproken en aan het einde volgt een reflectie over het proces. Op de volgende pagina ziet U een voorbeeld van de user interface.



(a) Answer Activity (b) Review Guess Activity

Figure 1: User interface

2 Beschrijving

2.1 Inleiding

"Quest for Questions" is een app waarmee men één op één vrienden beter kan leren kennen. Speler A en speler B moeten hiervoor inloggen in de app met een geregistreerd account. Daarna moet één van de twee de ander toevoegen om een spel te beginnen. Als speler A aan de beurt is, kan hij op speler B klikken in de vriendenlijst. Ze leren elkaar beter kennen aan de hand van meerkeuzevragen die ze over elkaar moeten beantwoorden. Wanneer speler A een spel start met speler B moet hij beginnen. In deze eerste ronde beantwoordt hij twee vragen over zichzelf (zie Figure 1.a). Vervolgens moet speler B in de tweede ronde de antwoorden op deze vragen aan speler A raden en daarna moet speler B twee vragen over zichzelf beantwoorden. De derde ronde is de eerste ronde die alle 3 activities achter elkaar aan de speler toont; speler A ziet wat speler B geraden heeft (zie Figure 1.b), speler A moet de antwoorden van speler B raden, en tenslotte moet speler A twee vragen over zichzelf beantwoorden. De rest van het spel volgen alle rondes deze volgorde.
Voor elke goed beantwoorde vraag wordt de vriendschapsscore van speler A en B verhoogd met de moeilijksheidsgraad van de vraag. Het ophogen van de score is te zien in de Review Guess Activity.
In de vriendenlijst zijn de toegevoegde vrienden te zien en de vriendschapsscore die bij die vriend hoort. Als de gebruiker aan de beurt is, komt het spel waarin hij aan de beurt is bovenaan de lijst te staan met een blauwe stip voor de naam van de desbetreffende vriend.

2.2 Productverantwoording

Naar onze mening is het bouwen van de app zeker de moeite waard geweest, aangezien alle aspecten, die wij als vereiste hebben gesteld, zijn gerealiseerd. Hierdoor zijn de aspecten die onze app boven de concurrenten uit laat steken ook gerealiseerd. De twee grootste aspecten die ons onderscheiden van de concurrenten:

i) Ten eerste is er de mogelijkheid om vragen toe te voegen en deze ontbreekt in alle grote concurrenten. Dit is een belangrijk aspect want creativiteit heeft per persoon zijn grenzen, en hoe meer mensen creatieve input kunnen hebben, des te groter de lijst met vragen en dus een verbetering in de app. We hebben echter hierdoor ook een mogelijkheid gegeven aan mensen met slechte bedoelingen om de app in zekere zin te verpesten, daarom hebben wij het reportsysteem toegevoegd.

ii) Ten tweede hebben de meeste apps veel toeters en bellen bij situaties zoals het resultaat van het raden van antwoorden. Een mogelijke reden hiervoor kan zijn dat ze op dat moment de app synchroniseren met hun database, maar dit resulteert ook in een lange wachttijd tussen vragen en dit weerhoudt de app ervan om vloeiend speelbaar te zijn. Hier zochten wij een oplossing voor en dit resulteerde in een zeer vlotte app waardoor het spelen een stuk plezieriger is dan de apps van de concurrenten.

2.3 Specificaties

i) De belangrijkste (functionele) eigenschappen zijn weergegeven in het use case diagram op Figure 2.

ii) De sub (niet-functionele) eigenschappen zijn:

- Door lang op een vriend te drukken in de vriendenlijst, zal er een pop-up komen die om een bevestiging vraagt of de gebruiker echt de desbetreffende vriend wil verwijderen. Indien op "Yes" geklikt wordt zal de vriend ook daadwerkelijk verwijderd worden.
- Op het scherm waar de gebruiker zelf een vraag moet beantwoorden, is er in de action bar een knop te zien bestaande uit drie puntjes. Als de gebruiker hierop drukt, zal er een overflow menu verschijnen met "Report" als enige optie. Als hij hierop drukt, zal er een pop up komen die om een bevestiging vraagt of de gebruiker echt de vraag wil reporten. Indien op "Yes" geklikt wordt zal de vraag als "reported" in de database aangemerkt worden.
- Door op de floating action button te klikken op het scherm van de vriendenlijst, zullen er twee opties tevoorschijn komen, namelijk de optie om een vriend toe te voegen en de optie om een vraag toe te voegen.
  (a) Als de gebruiker heeft gekozen om een vriend toe te voegen, zal de gebruiker worden doorverwezen naar een scherm waar hij in een veld het emailadres van de vriend kan invoeren. Vervolgens kan de gebruiker op "ADD FRIEND" klikken om de vriend toe te voegen, waarna hij wordt terugverwezen naar het scherm van de vriendenlijst, waar de toegevoegde vriend nu in staat.
  (b) Als de gebruiker heeft gekozen om een vraag toe te voegen, zal de gebruiker worden doorverwezen naar een scherm waar hij in de velden de vraag en mogelijke antwoorden invult. Daarna kan hij op "ADD QUESTION" klikken om de vraag toe te voegen. Vervolgens wordt hij terugverwezen naar het scherm van de vriendenlijst.
- Op het scherm van de vriendenlijst, is er in de action bar een knop te zien bestaande uit drie puntjes. Als de gebruikter hierop drukt, zal er een overflow menu verschijnen met de opties "Help" en "Log out". Als de gebruiker op "Help" klikt, zal er een popup komen met een korte handleiding van de app. Als de gebruiker op "Log out" klikt, zal de gebruiker worden uitgelogd en naar de Log In activity worden doorverwezen.
- Door naar beneden te slepen bij het vriendenoverzicht wordt de vriendenlijst ververst.

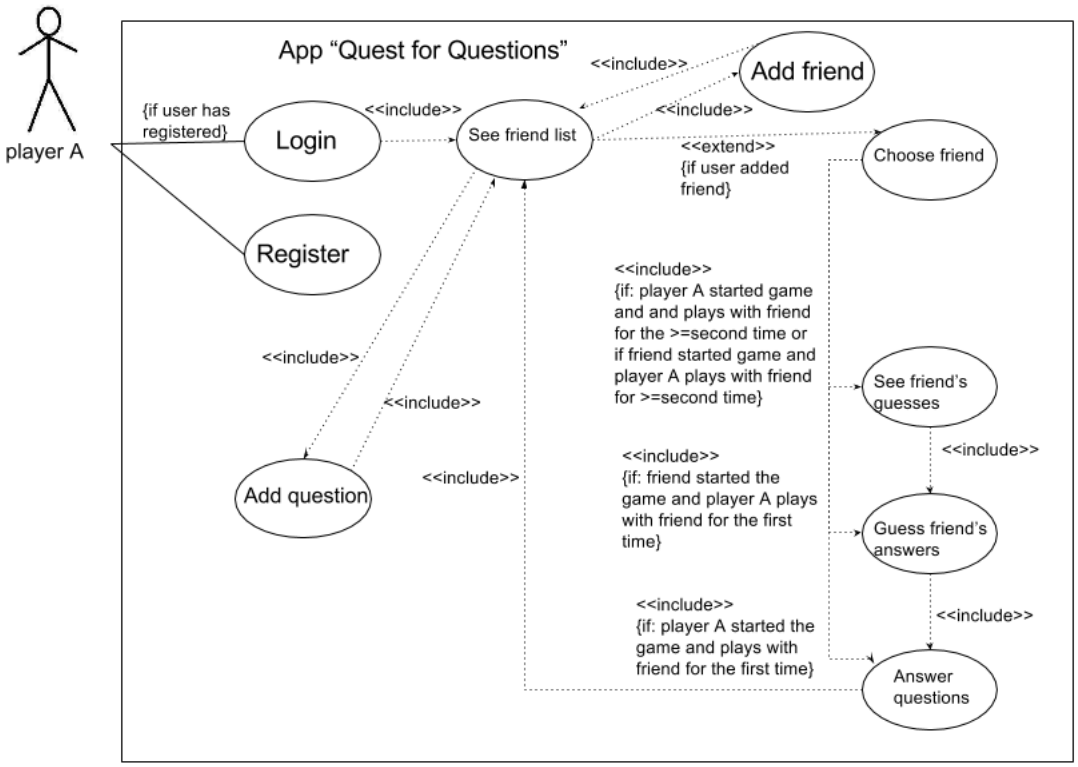


Figure 2: Use case diagram

3 Ontwerp

3.1 Globaal Ontwerp

Ons ontwerp bestaat uit vier onderdelen:

- De activities die de interactie leveren met de gebruiker. Bij elke activity hoort een XML-bestand dat het uiterlijk van de activity regelt.
- De functies die bestemd zijn voor de communicatie met Firebase en dus de communicatie met de database regelen.
- De notificatieklasse regelt de notificaties die, wanneer nodig, de gebruiker informeren over de status van de app.
- Tot slot zijn er nog klassen die de activities op bepaalde manieren ondersteunen. Elke activity is zijn eigen klasse en we hebben gekozen om de firebase aspecten te hangen aan

elke activity klasse waar nodig, omdat de firebase methode voor elke activity verschillend is.

Er zijn acht activities. De eerste is de LoginActivity die het inloggen van de gebruiker regelt, vanaf deze activitiy wordt je gestuurd naar de RegisterActivity of de MainActivity. De RegisterActivity regelt het registreren van nieuwe users en brengt de gebruiker terug naar de LoginActivity. In de MainActivity bevindt zich het menu met de vrienden van de gebruiker. Deze activity wordt ondersteund door de RecyclerDivider, die zorgt voor het tekenen van de strepen tussen de vrienden weergegeven in de MainActivity. Vanuit de MainActivity kan je vrienden toevoegen in de AddFriendActivity en vragen toevoegen in de AddQuestionActivity. Tot slot kan je op vrienden drukken waarna de speelronde begint die bestaat uit:

i) ReviewGuessActivity, die de resultaten van de vorige ronde laat zien, met de score die verkregen is door de spelers. Deze activieit kan twee keer aangeroepen worden als er meerdere antwoorden gereviewed moeten worden, hetzelfde geldt voor de volgende punten.

ii) GuessActivity, waarbij de gebruiker moet gokken wat het antwoord van de tegenstander was.

iii) AnswerQuestion, die de mogelijkheid geeft om een vraag te beantwoorden.

3.2 Detailontwerp

Wegens de grootte van de afbeelding bevindt het UML-diagram zich op de volgende pagina (zie Figure 3).

3.3 Ontwerpverantwoording

Wij hebben voor dit ontwerp gekozen, omdat we gebruik gemaakt hebben van het MVC pattern. Hierbij is de view het XML gedeelte is en het model en de controller bevinden zich in de Java activity code. De activities sturen ook de hulpfuncties en de notificatieklasse aan. Tot slot leek het ons verstandig om de code die Firebasefuncties aanroept te verdelen over meerdere activities, aangezien iedere activity zijn eigen callbacks nodig heeft.

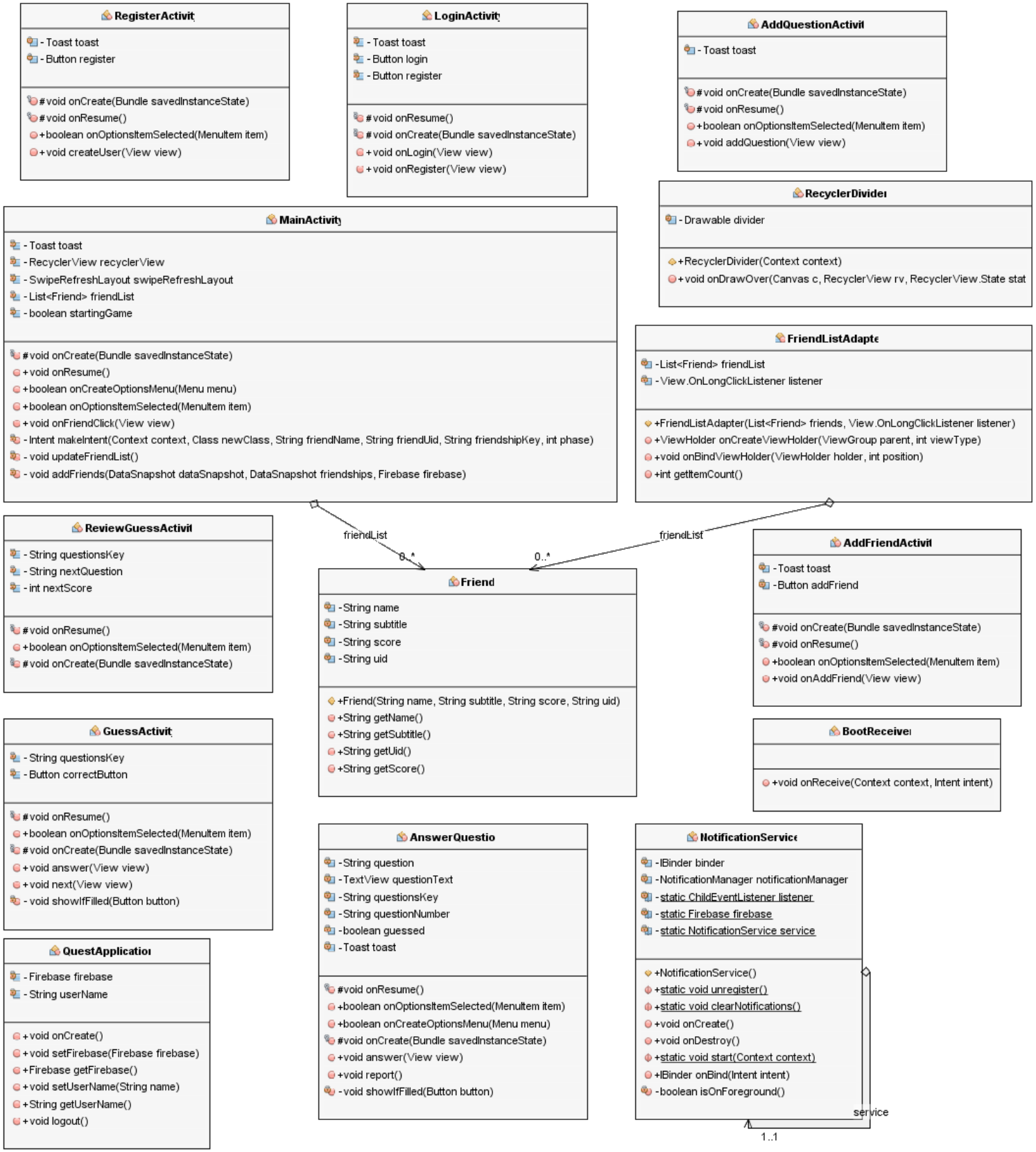


Figure 3: KlasseUML

4 Reflectie

We zijn zeer tevreden over onze uiteindelijke app, omdat onze vereiste eigenschappen gerealiseerd zijn, en omdat we alle bugs - behoudens deze die niet door ons ontdekt zijn - opgelost hebben. Er zijn zelfs functies toegevoegd ten behoeve van het verbeteren van de usabillity van de app - denk hierbij aan een "Swipe to refresh"-functionaliteit etc. Als groep zijn wij succesvol geweest in het plannen van het bouwen van de app en dit scheelde ons een woensdagnacht doorwerken. Dit en vele andere aspecten zoals het goed gecoördineerd werken resulteerde in een fijne samenwerking en een succesvol eindproduct. Uiteraard zijn er ook wat negatieve aspecten, zoals het probleem hebben van het, wat in het begin leek op eindeloos, zoeken naar een manier om de gegevens van de deelnemers op te slaan, maar uiteindelijk hebben wij ook hier een passende oplossing gevonden - in dit geval Firebase. Op enkele momenten vonden we het echter niet plezierig dat er weinig informatie verstrekt werd over wat er van ons verwacht werd bij bepaalde activiteiten. Tot slot vonden wij het ook matig dat de usabillity opdracht midden in de tentamenweek viel en dat het vervolgens ook nog een vereiste was om de leden van nog een andere groep als testpersonen te gebruiken, wat resulteerde in een irritatie in de planning. De volgende keer dat we een grote groepsopdracht die met programmeren te maken heeft krijgen, zullen we in het begin al kijken naar potentiële bugs die kunnen ontstaan als we een idee implementeren. Dit scheelt ons heel wat werk, met name op het gebied van debuggen, wat uiteindelijk best veel tijd heeft gekost.