Programmeertechnieken 2018
Opdracht 3: Web Applicatie

Deadline: Donderdag 31 mei 23:59.

1 Introductie

LET OP: dit is wederom een grote opdracht. Begin op tijd en maak goed gebruik van de werkcolleges om hulp te krijgen. Ook buiten de werkcolleges zal een aanzienlijke tijdsinvestering nodig zijn. De kans dat het lukt om de opdracht op het laatste moment helemaal te maken is minimaal.

Voor deze opdracht is het de bedoeling dat je zelf een chatprogramma maakt vergelijkbaar met bijvoorbeeld WhatsApp of Slack. In tegenstelling tot de eerdere opdrachten werkt deze applicatie niet lokaal op één systeem maar is gedistribueerd over meerdere systemen zodat gebruikers op afstand (over Internet) met elkaar kunnen chatten. Het programma valt uiteen in drie delen:

- Het frontend is een programma dat de gebruiker op zijn/haar computer draait om te kunnen chatten. Dit onderdeel bevat de grafische gebruikersinterface.
- De frontend maakt verbinding met de backend om berichten te kunnen ontvangen en verzenden. Dit programma draait binnen een webserver en wordt automatisch aangeroepen als er een HTTP-verzoek binnenkomt. Het programma biedt op deze manier een API om deze activiteiten te kunnen ondersteunen.
- De backend slaat geen gegevens op. Voor alle permanente opslag wordt een aparte database gebruikt. De backend doet dit door middel van SQL queries.

Alle onderdelen moeten in ieder geval werken op Linux (in het bijzonder, de Linux-installatie in de computerzalen van Snellius), maar werken bij zorgvuldig gebruik van het framework ook op Windows en macOS.

De leerdoelen van deze opdracht vallen uiteen in twee onderdelen: ontwerpen van applicaties en diversificatie van je programmeervaardigheden. Wat betreft ontwerp willen we in het resultaat terugzien dat je de volgende vaardigheden beheerst:

- een gedistribueerde applicatie ontwerpen;
- toepassen kwaliteitscriteria voor ontwerpen;
- gebruik maken van design patterns;
- een API ontwerpen;
- een database ontwerpen.

De volgende leerdoelen hebben betrekking op het diversificeren van je programmeervaardigheden:

- kennis maken met een veilige gecompileerde programmeertaal;
- werken met een framework;
- een webapplicatie implementeren;
- een grafische gebruikersinterface maken.

2 Groepswerk

Dit is een behoorlijk groot programmeerproject. Er wordt daarom gewerkt in teams van drie personen. Je kan de opdracht alleen met een groep van een andere grootte maken als tijdig toestemming daarvoor vraagt via pt2018@lists.liacs.nl, met opgaaf van reden. Als je voor opdracht 2 al toestemming had, dan hoef je niet opnieuw toestemming te vragen om deze opdracht met dezelfde groep te maken.

Het wordt ten zeerste aangeraden om source control te gebruiken. Een source control systeem als Git maakt het een stuk makkelijker om samen bijvoorbeeld in de vorm van de LIACS git server `https://git.liacs.nl` te werken aan hetzelfde programma. Let wel op dat je de project visibility private maakt (NIET internal), anders deel je de code met studenten buiten je groepje. Dat kan leiden tot plagiaat en is daarom niet toegestaan.

Nog een praktische tip: als je samen werkt aan een tekstdocument dan is het sterk aan te raden dit in LaTeX-formaat te doen en het LaTeX-bestand op te nemen in je git repository. Dat geldt in het bijzonder voor de ontwerpdocumenten (zie sectie **??**) die een belangrijke rol zullen spelen in de samenwerking. Het voordeel van git boven bijvoorbeeld e-mail of WhatsApp is dat meerdere groepsgenoten veilig tegelijk wijzigingen kunnen aanbrengen. Daarnaast is altijd direct duidelijk wat de laatste stand van zaken is, terwijl ook een historie van wijzigingen bewaard blijft. LaTeX heeft als groot voordeel dat het een teksformaat is; met binaire formaten zoals Word-documenten kan git niet goed overweg en kan je niet gelijktijdig wijzigingen maken zonder onoplosbare conflicten.

3 Aanpak

Bij deze opdracht staat het ontwerpen van een applicatie centraal. Om deze reden geven we, in tegenstelling tot de vorige opdracht, geen aanzet voor het ontwerp maar bevat de opdracht enkel de eisen waar de applicatie aan moet voldoen. Begin daarom niet met programmeren, maar maak eerst een ontwerp om te weten wat je moet gaan programmeren. Je kan dat als volgt aanpakken:

- lees de opdracht goed door en maak een overzicht van alle eisen waaraan je applicatie moet voldoen;
- deel de eisen in over de drie componenten waaruit je applicatie gaat bestaan (frontend, backend, en database), waarbij het mogelijk is dat een taak door meerdere componenten samen verricht moet worden;
- maak op basis van de taakverdeling een overzicht welke data er in de database opgeslagen moet worden;
- maak op basis van de taakverdeling ook een overzicht welke mogelijkheden de API tussen de frontend en de backend moet bieden.

Let op dat eisen zowel functioneel als niet-functioneel kunnen zijn. Een voorbeeld van een functionele eis: je moet een tekstbericht kunnen sturen naar een andere gebruiker. Een voorbeeld van een niet-functionele eis: het bericht moet direct doorgestuurd worden (als push-notificatie). Let ook op dat niet alle eisen expliciet in de opdracht staan. Zo impliceert het moeten kunnen inloggen bijvoorbeeld dat je moet bijhouden welke gebruikers er zijn en wat hun wachtwoorden zijn.

Als je deze stappen hebt doorlopen dan heb je een eerste ontwerpdocument met een globaal overzicht van de taakverdeling en communicatie tussen de programma's. Zet dit op papier (bij voorkeur een LaTeX-document in git), want het wordt onderdeel van het verslag dat je moet inleveren. Daarnaast kunnen groepsgenoten vanaf dit moment onafhankelijk aan de verschillende onderdelen werken. Ook daarvoor is het handig de taakverdeling en communicatie goed op papier te hebben staan.

Nu dat je een globaal ontwerp hebt kun je de verschillende componenten uit gaan werken. Begin niet meteen met programmeren, maar denk eerst na welke klasses je moet gaan schrijven om de gevonden taken uit te voeren. Maak een overzicht van de taakverdeling en relaties tussen de klasses. Houd hierbij rekening met de kwaliteitscriteria die tijdens het college besproken zijn en probeer goed gebruik te maken van de besproken design patterns. Omschrijf de keuzes die je hebt gemaakt en de relatie met de kwaliteitscriteria en de design patterns in het ontwerpdocument dat je in gaat leveren.

4 Functionaliteiten

Zoals al aangegeven, bestaat de applicatie uit drie onderdelen: de frontend, de backend, en de database. De omschrijving hier gaat uit van de gebruiker, die de grafische interface van de frontend voor zich heeft. Uiteraard hebben de eisen ook implicaties voor de andere onderdelen.

4.1 Aanmeldscherm

Wanneer de frontend gestart wordt, moet de gebruiker zich aanmelden door een gebruikersnaam en wachtwoord in te voeren. De applicatie controleert direct of de combinatie gebruikersnaam/wachtwoord klopt. Zo nee, dan komt er een foutmelding en kan de gebruiker het opnieuw proberen. Zo ja, dan wordt het chatscherm geopend.

Een nieuwe gebruiker heeft de optie een nieuw account aan te maken en kan dan zelf een gebruikersnaam en wachtwoord kiezen. Voorwaarden hierbij zijn dat de gebruikersnaam nog niet in gebruik is en dat het wachtwoord minstens 8 tekens lang is. Als niet aan de voorwaarden voldaan is, krijgt de gebruiker een foutmelding en kan het opnieuw proberen.

4.2 Chatscherm

Op het chatscherm staat een lijst met actieve gebruikers (die momenteel aangemeld zijn). Deze lijst wordt gevuld wanneer het scherm geopend wordt. Als de gebruiker op één van de gebruikers uit de lijst klikt, dan verschijnt de chatgeschiedenis tussen de lokaal aangemelde gebruiker (verder: Alice) en de geselecteerde gebruiker (verder: Bob). De geschiedenis gaat van oud (boven) naar nieuw (onder) en voor elk bericht wordt de afzender en tijd weergegeven. Als Alice nu een chatbericht typet, dan wordt het verzonden naar Bob. Als Bob het chatvenster van Alice open heeft staan, verschijnt het bericht daarin. Zo niet, dan opent het chatvenster van Alice en wordt het bericht dus alsnog weergegeven. In beide gevallen moet aan de gebruiker visueel en/of auditief duidelijk worden dat er een bericht binnen is gekomen.

Er worden geen gegevens lokaal op de schijf opgeslagen. Dit betekent in het bijzonder dat de chatgeschiedenis elke keer via de backend uit de database opgeslagen wordt. Op deze manier is het mogelijk voor een gebruiker om op het programma op meerdere computers te gebruiken zonder zijn/haar chatgeschiedenis kwijt te raken.

Het chatscherm biedt een optie voor een gebruiker om zich af te melden. De gebruiker verdwijnt dan direct overal van de lijst met actieve gebruikers en gaat terug naar het aanmeldscherm. Als de frontend wordt gesloten dan wordt de gebruiker ook afgemeld.

4.3 Configuratie

Omdat je applicatie uit meerdere delen bestaat die met elkaar communiceren, zullen ze elkaar moeten kunnen vinden. Het moet eenvoudig zijn om aan de backend op te geven waar de database staat (een bestandsnaam omdat we SQLite gebruiken, zie sectie **??**) en om aan de frontend op te geven waar de backend benaderd kan worden (een URL waar de webserver draait). De configuratie is overigens de enige permanente data die buiten de database opgeslagen mag worden.

4.4 Beveiliging

De applicatie moet ervoor zorgen dat het niet mogelijk is om te chatten onder de gebruikersnaam van een ander die eerder al een account heeft gemaakt zonder diens wachtwoord te weten. Evenmin is het mogelijk om chatberichten te lezen van chats waaraan de gebruiker zelf niet heeft deelgenomen of is het mogelijk om te bepalen of iemand een bericht verzonden heeft als men daar zelf geen ontvanger van is. Beide gelden zelfs als iemand zelf een frontend schrijft dat aansluit op de API van het backend.

Let op: eigenlijk is het niet veilig om wachtwoorden over het Internet te versturen via HTTP zonder beveiligde verbinding en evenmin is het veilig om wachtwoorden zonder beveiligingsmaatregelen op te slaan in een database. Voor deze opdracht gaat het te ver om hier rekening mee te houden en dat hoeft dan ook niet. Je mag er daarom vanuit gaan dat de verbinding niet onderschept kan worden. Realiseer je wel dat je dus meer zal moeten letten op beveiliging als je zoiets in het echt gaat doen. Voor de Informatica en Informatica&Economie studenten zal dit later bij het vak Security aan bod komen. Wees wel voorzichtig: gebruik geen echte wachtwoorden die je voor andere doeleinden gebruikt wanneer je de applicatie test.

4.5 Extra's

Als aan alle eisen is voldaan dan kan je maximaal een 8,5 halen. Voor een hoger cijfer verwachten we van je dat je extra functionaliteit toevoegt. Er zijn drie extra opdrachten die elk maximaal 0,5 punt extra op kunnen leveren.

Vriendenlijst Standaard geeft het programma alle gebruikers weer in de contactenlijst. Als je deze bonus wilt halen dan beperk je deze lijst tot mensen die als vrienden gemarkeerd zijn en kunnen alleen vrienden berichten naar elkaar sturen. Er komt een scherm om iemand uit te nodigen als vriend. Wanneer iemand door een ander is uitgenodigd als vriend krijgt hij/zij daarvan direct een melding, met de keus om het verzoek te accepteren of te weigeren. Verder is er de optie om mensen te ontvrienden.

Afbeeldingen Standaard kan het chatprogramma enkel tekst verwerken. Als je deze bonus wilt halen dan moet je de mogelijkheid implementeren om naast tekst ook afbeeldingen te verwerken. Dat betekent twee aanvullingen:

- Bij het aanmaken van een account kan de gebruiker een plaatje uitkiezen als avatar. Dit plaatje wordt vervolgens in de contactenlijst en bij chatberichten van deze persoon weergegeven. De gebruiker kan zelf een bestand uitkiezen of kiezen uit een lijstje met standaard-avatars.
- Naast tekstberichten kan de gebruiker nu ook afbeeldingen versturen, die tussen de tekst in de chathistorie getoond worden. De gebruiker kan zelf een bestand uitkiezen of kiezen uit een lijstje met standaard-emoticons.

Groepschat Standaard kan het chatprogramma enkel gebruikt worden om één-op-één-conversaties te houden. Als je deze bonus wilt halen moet de gebruiker een groepschat aan kunnen maken door groepsleden te selecteren. De groepsleden zien deze groepschat in de lijst met gebruikers, kunnen daar berichten naartoe sturen, en ontvangen berichten die door alle leden naar de groepschat gestuurd zijn. Elke groepschat heeft minstens drie deelnemers en er is geen maximum aan het aantal deelnemers in een groepschat.

5 Tools

Het is de bedoeling de frontend en backend in C# te schrijven. Dit is de belangrijkste programmeertaal van het .NET framework van Microsoft, maar wordt onderhand ondersteund op meerdere

platformen door middel van het open source Mono framework. De taal is erg vergelijkbaar met Java en het is niet moeilijk de ene taal te leren als je de ander kent.

Helaas is de versie van MonoDevelop die op de computers van Snellius geïnstalleerd staat niet volledig functioneel. In plaats daarvan kan je een versie gebruiken die we zelf gecompileerd hebben. Hiervoor open je een terminalvenster en start je het volgende commando:

```
/vol/share/groups/liacs/scratch/pt2018-mono-devel/run-monodevelop
```

Zowel de frontend als de backend worden in C# geschreven. De frontend wordt een C# GTK# applicatie. De backend wordt een C# ASP.NET applicatie.

Voor de database gebruik je SQLite. Deze database heeft geen echte server. In plaats daarvan benadert de client rechtstreeks de databasebestanden. Deze aanpak is echter voor de opdracht functioneel equivalent aan de situatie besproken in het college waar wel een aparte server draait en heeft als voordeel dat het veel makkelijker op te zetten is.

In de appendix (sectie **??**) staan enkele tips over het gebruik van de programma's die je gaat gebruiken.

6 Inleveren

Elke groep bestaat uit drie personen (zie sectie **??** voor mogelijke uitzonderingen) en levert de opdracht éénmaal in. Bij het inleveren moeten de volgende onderdelen worden meegeestuurd:

- C# source code voor frontend (als MonoDevelop Forms project);
- C# source code voor backend (als MonoDevelop ASP.NET project);
- een SQL bestand met queries om de SQLite database leeg op te zetten;
- het verslag als PDF bestand.

Het verslag moet de volgende informatie bevatten:

- instructies om de applicatie op te zetten op een Linux systeem (minimaal hoe de database opgezet moet worden en hoe de configuratie ingesteld moet worden, zie sectie **??**);
- wie heeft wat gedaan?
- ontwerpdocumenten omschreven in sectie **??**;
- documentatie van de functies in je API zodanig dat iemand zonder jouw code op de backend kan aansluiten.

Alle vereiste onderdelen moeten in het verslag staan en duidelijk uitgelegd worden, maar houd het kort. Zorg er verder voor dat je het document op een overzichtelijke manier organiseert.

Zorg ervoor dat alle bestanden die worden ingeleverd zijn voorzien van namen en studentnummers! Plaats alle bestanden om in te leveren in een aparte directory (bijv., `opdracht3`) en maak een "gzipped tar" bestand:

```
tar -czvf opdracht3-sXXXXXXX-sYYYYYYY-sZZZZZZ.tar.gz opdracht3/
```

Vul op de plek van `XXXXXXX`, `YYYYYYY` en `ZZZZZZZ` de bijbehorende studentnummers in. De inzendingen kunnen worden verzonden per e-mail naar pt2018@lists.liacs.nl met als onderwerp "PT Opdracht 3 AAA, BBB, CCC", waarbij je AAA, BBB, en CCC vervangt door de namen van de studenten in je groep. **Belangrijk:** Test je programma in ieder geval op de Linux machines in de computerzalen! Dit is voornamelijk belangrijk voor studenten die werken op een Windows of macOS computer, zorg ervoor dat je programma ook werkt op een computer in de computerzaal.

7 Beoordeling

De beoordeling van de opdracht bestaat uit de volgende onderdelen:

- ontwerp inclusief onderbouwing in verslag (2,0),
- kwaliteit code en commentaar (2,0),
- werking user interface (1,5),
- werking backend, inclusief API en documentatie (2,0),
- databaseontwerp (1,0),
- extra functionaliteit (1,5).

Zonder de extra functionaliteit kan voor een correct werkend programma dat de hierboven omschreven basisfunctionaliteiten bevat maximaal een 8,5 worden gehaald.

A Gebruik van de tools

A.1 Werken op andere systemen

Je programma moet uiteindelijk werken op Linux (de computerzalen in het Snellius) en gecompileerd kunnen worden met MonoDevelop zoals aldaar geïnstalleerd, maar het staat je vrij om zelf op een ander platform en/of met een andere editor te werken:

- Op je eigen Linux-systeem kan je MonoDevelop installeren met je package manager (in Ubuntu: `sudo apt-get install monodevelop`).
- Op macOS kan je Visual Studio for Mac downloaden (zie `https://www.monodevelop.com/download/`), welke is afgeleid van MonoDevelop. Je kan ook de originele MonoDevelop gebruiken, maar dan moet je zelf compileren van source (instructies staan hier: `www.monodevelop.com/developers/building-monodevelop/`).
- Op Windows kan je gratis Visual Studio Community 2017 downloaden van `https://www.visualstudio.com/downloads/`. Net als op macOS kan je ook MonoDevelop gebruiken als je deze zelf compileert. Visual Studio gebruikt standaard System.Windows.Forms om forms te ontwerpen terwijl MonoDevelop Gtk# gebruikt. Eerstgenoemde zou wel moeten werken op MonoDevelop, maar je kan deze forms niet visueel ontwerpen. Omgekeerd is het ook mogelijk om GTK# te gebruiken op Windows: `http://www.mono-project.com/docs/gui/gtksharp/beginners-guide/`.

LET OP: we kunnen niet garanderen dat projecten tussen Visual Studio en MonoDevelop 100% uitwisselbaar zijn, dus we raden aan regelmatig te controleren of je code ook op MonoDevelop werkt. Het is je eigen verantwoordelijkheid om een programma aan te leveren dat op de Linux-installatie van de computers van Snellius werkt.