Totaal rapport Felix

Naam: Bart Janssen
Studentennummer: 389419
Datum: 20 september 2020

Samenvatting

In dit document wordt beschreven wat voor functionaliteiten en security maatregelen zijn getroffen voor de ontwikkeling van de chat applicatie Felix. Hierbij is ook nagedacht over de gebruikers van het systeem en wat eventuele risico's zijn zoals de threat- en risk analyse. Er zijn ook enkele wireframes te zien wat een beeld geven van hoe de applicatie eruit moet komen zien.

Inhoudsopgave

1 Inleiding

Voor de minor op de hogeschool Fontys te Eindhoven werd er gevraagd om een applicatie te maken om verschillende leerdoelen aan te tonen. Omdat er redelijk wat leerdoelen zijn en veel hiervan in een applicatie bij elkaar te implementeren zijn, is er een chat applicatie gemaakt genaamd Felix. Deze applicatie bevat verschillende functionaliteiten en security maatregelen. In dit rapport staat beschreven wat de chat applicatie Felix is en wat hier allemaal in zit qua functionaliteiten en op het gebied van security.

2 Design

Voor het design van de applicatie is nagedacht over wat er allemaal nodig is om deze applicatie tot een succes te brengen.

2.1 Hardware

De hardware voor deze applicatie bestaat minimaal uit twee systemen, maar voor een correcte werking zijn drie systemen aangeraden. Een server waarop de chatserver en database moet draaien en twee clients die via de server met elkaar kunnen chatten. De server moet publiekelijk toegankelijk zijn en de clients kunnen privé systemen zijn van de gebruikers van de applicatie.

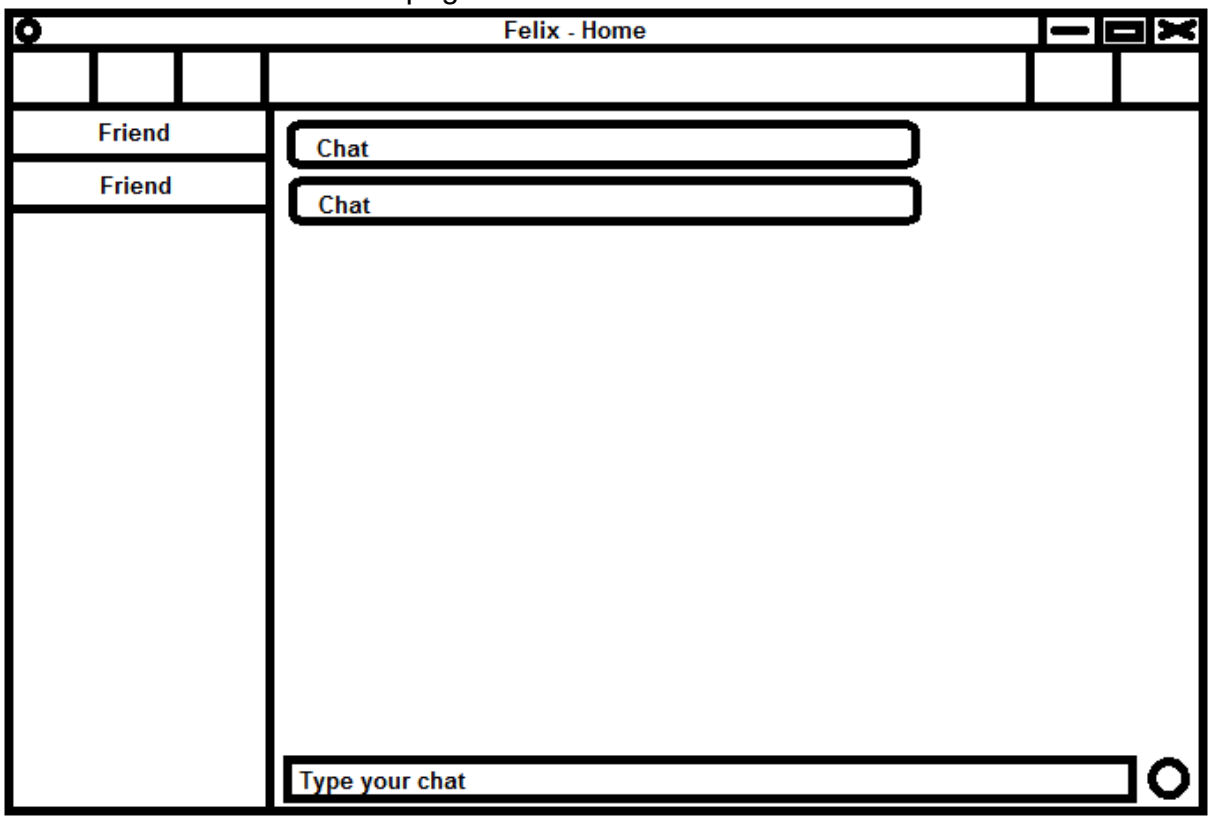
2.2 Software

Zoals eerder beschreven bestaat Felix uit drie systemen, een server, een database en een client. De server is geschreven in Java Spring boot en zal draaien op de eerder genoemde publiekelijk toegankelijke server. De database is een PostgreSQL database die ook op dezelfde server te draaien komt. De client is in JavaFX geschreven en deze wordt gebruikt om met de server te communiceren. Het uiterlijk van de client moet eenvoudig te gebruiken zijn, en er is gekozen voor een standaard ‘dark mode’. Deze keuze is gebaseerd op al bestaande chat applicaties en veel mensen blijken dit prettig te vinden en ook beter is voor de gezondheid van mensen (Gillz).

2.3 Wireframes

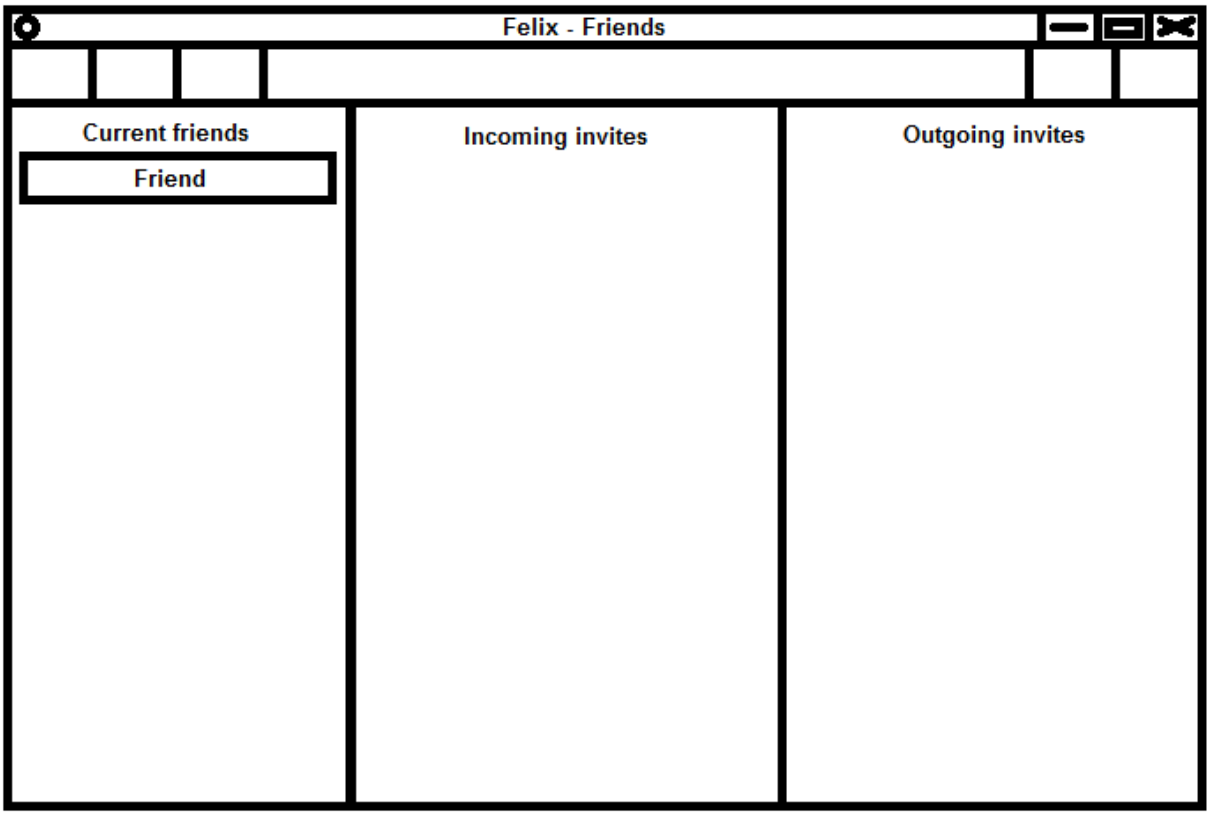
Er zijn wireframes gemaakt zodat er een overzicht is van hoe het design van de applicatie er uit moet komen zien.

De wireframe voor de home pagina:



Afbeelding 1 Wireframe home

De wireframe voor de ‘Friends’ tab:



Afbeelding 2 Wireframe friends

2.4 Security by design

Security by design betekent dat je tijdens het ontwerp van een nieuwe applicatie al rekening houdt met de beveiliging van de applicatie en persoonsgegevens. Zo moet er in kaart gebracht worden welke dreigingen van belang zijn bij deze applicatie. Een mogelijke manier omdat te doen is een threat – en risk analyse uit te voeren.

Threat analyse

In een threat analyse wordt er gekeken naar welke threats er zijn, welke vaak worden gebruikt en welke van toepassing zijn op de applicatie. Hiervoor zijn er een aantal van de veelvoorkomende aanvallen op een rijtje gezet. Veelvoorkomende aanvallen binnen de cyber wereld zijn onder andere:

- Malware, Trojans;
- Phishing;
- MitM (Man in the Middle);
- Ransomware;
- DDoS (Distrubuted Denail of Service);
- Injection;
- Brute force;
- Unknown attacks/zero-days.

Ieder jaar komen er weer nieuwe cyberaanval-technieken aan en dus kan deze lijst vaak veranderen. Omdat deze nu momenteel nog onbekend zijn, vallen deze onder de ‘unknown attacks/zero-days’. Hierop kun je moeilijk anticiperen. Mogelijk is het beste wat daartegen te doen is, is het loggen en monitoren van ongebruikelijk of vreemd gedrag binnen een applicatie wat niet onder te brengen is onder een van de bekendere technieken.

Ook de gebruikers van de applicatie kunnen verschillen. Het zou geweldig zijn als de gebruikers alleen de applicatie gebruiken waarvoor hij gemaakt is, helaas is dat in werkelijkheid niet zo. Veelvoorkomende kwaadwillende actoren binnen de cyber wereld zijn de volgende:

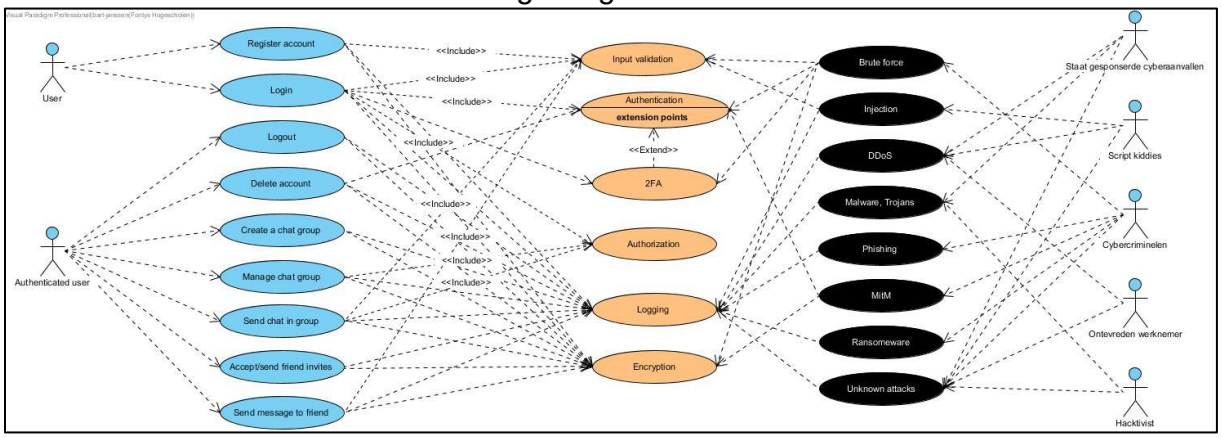
- Staat gesponseerde cyberaanvallen;
- Script kiddies;
- Cybercriminelen;
- Ontevreden werknemers;
- Hacktivisten.

Veel van deze actoren voeren niet altijd dezelfde aanvallen uit. Veel aanvallen zijn gekoppeld aan een soort actor. Om dit in beeld te kunnen brengen, is het makkelijk om een tabel te maken met hierin welke actor welke type aanval vaak uitvoert, en wat hun motivatie daarbij is.

Threat actor	**Motivatie**	**Misuse activiteiten**
Staat gesponseerde cyberaanvallen	Economie verbeteren van eigen land	DDoS, Malware, Trojans, unknown attacks
Script kiddies	Leren en uitproberen van hacken	DDoS, Injection, unknown attacks
Cybercriminelen	Geld verdienen	Ransomware, Brute force, MitM, Phishing, unknown attacks
Ontevreden werknemers	Wraak	DDoS, unknown attacks
Hacktivisten	Politieke aandacht trekken	Malware, Trojans, unknown attacks/zero-days

Tabel 1 Threat actoren met motivatie

Hiervoor is ook een misuse case diagram gemaakt:



Afbeelding 3 Misuse case diagram

Risk analyse
Een Risk analyse beschrijft de kans en impact van de eerder onderzochte threat analyse.

Niet al deze threats zijn even gevaarlijk, of hebben een even grote impact voor mijn applicatie. Om dit duidelijk in kaart te brengen, is de volgende tabel gemaakt:

Threat	**Threat Impact beschrijving**	**Kans**	**Impact**	**Reden**
Malware, Trojans	Virussen die bij gebruikers geïnstalleerd kunnen worden	Laag	Hoog	In deze applicatie is geen directe gebruiker naar gebruiker communicatie
Phishing	Linkjes naar shady websites met mogelijke afpersing of dergelijke	Hoog	Hoog	Een link kan gemakkelijk meegestuurd worden met een chat berichtje
MitM	Tussen de communicatie inzitten om informatie uit te lezen	Hoog	Hoog	Deze aanval kan uitgevoerd worden op de uitwisseling voor encryptie
Ransomware	Gijzelen van een computer	Klein	Hoog	Dit soort dergelijke aanvallen werken op computers, niet op een applicatie. Het is makkelijk mogelijk om een server opnieuw te lanceren
DDoS	Het platleggen van een systeem door overmatig veel aanvragen te doen	Hoog	Hoog	De applicatie kan compleet offline gebracht worden
Injection	Het injecteren van scripts of code	Hoog	Hoog	Dit zou de applicatie kunnen breken of de database kunnen aanpassen
Brute force	Het	Middel	Hoog	Inloggegevens

	herhaaldelijk proberen van een actie			brute forcen
Unknown attacks	Het onbekende	Middel	Onbekend	Dit zijn onbekende aanvallen, tot er niet bekend is wat dit is, is het enige wat er eraan gedaan kan worden, het monitoren en loggen

Tabel 2 Risico matrix

Tegen de meeste van deze threats is het mogelijk om maatregelen te trefen:

Threat	**Maatregel**
Malware, Trojans	Een check op bijvoorbeeld bestandsformaten die verstuurd kunnen worden in de applicatie
Phishing	Een waarschuwing bij het openen van linkjes, mogelijk bekende domainen blokkeren
MitM	Werken met checks of certificaten bijvoorbeeld
Ransomware	Een check op bijvoorbeeld bestandsformaten die verstuurd kunnen worden in de applicatie
DDoS	Mogelijk wordt dit soort applicaties gebruikt onder een Kubernets cluster, deze vangt dit al af. Mogelijk een eigen maatregel zou het zijn van blacklisten van IP-adressen
Injection	Input check in zowel de client als server, veel frameworks hebben dit standaard ingebouwd
Brute force	Blokkeren als een event een bepaalde tijd herhaaldelijk wordt uitgevoerd
Unknown attacks	Zolang je niet weet wat het is kunnen deze alleen gelogd worden

Tabel 3 Maatregelen

3 Functionaliteiten

De functionaliteiten voor de applicatie waren in het begin van het semester opgesteld als use cases.
Functionele eisen:

- Een gebruiker moet een account kunnen registreren;
- Een gebruiker moet kunnen inloggen;
- Een gebruiker moet kunnen uitloggen;
- Een gebruiker moet zijn/haar account kunnen verwijderen;
- Een gebruiker moet een chat groep aan kunnen maken;
- Een gebruiker moet een bericht in een groep kunnen versturen;
- Een gebruiker moet vriendschapsverzoeken kunnen versturen, accepteren en weigeren;
- Een gebruiker moet privé berichten kunnen versturen naar vrienden;
- Een gebruiker moet two factor authenticatie in kunnen schakelen.

Non-functionele eisen:

- De wachtwoorden van de gebruikers moeten versleuteld opgeslagen zijn;
- De server moet te allen tijde bereikbaar zijn;
- De chats moeten versleuteld opgeslagen worden;
- Het verkeer van en naar de server moet versleuteld zijn.

4 Security

De security van de applicatie is een belangrijk onderdeel. Als deze niet in orde is dan is de applicatie zwak tegen aanvallen en kunnen er bijvoorbeeld gebruikers data gestolen worden. Als dit gebeurt dan kan dit voor de ontwikkelaars van de applicatie grote gevolgen hebben. Enkele gevolgen zijn bijvoorbeeld dat gebruikers de applicatie niet meer willen gebruiken, gebruikers zouden de ontwikkelaars kunnen aanklagen wegens nalatigheid van beveiliging en mogelijk tegen verschillende artikelen van de AVG-wet. In Felix zijn er verschillende security aspecten ingebouwd.

4.1 Encryptie

Er wordt een mix gebruikt met RSA en AES-encryptie. Als de applicatie opstart dan worden publieke sleutels uitgewisseld via RSA. Een per sessie gegenereerde AES-sleutel wordt dan via RSA naar de client gestuurd. Vervolgens wordt al het verkeer met AES versleuteld.

De chats die opgeslagen staan in de database worden ook met AES versleuteld en vervolgens opgeslagen. Als er chats uit de database uitlekken dan kan hier niets mee gedaan worden aangezien deze met een 256 bit AES-sleutel versleuteld zijn. Om een 256 bit AES-sleutel te kraken, zijn er ongeveer 2^{256} pogingen mogelijk. Dit duurt zelfs met de huidige computerkracht meerdere miljarden jaren om dit te kraken.

4.2 Wachtwoorden

De wachtwoorden van de gebruikers worden ook opgeslagen in de database. Veel mensen gebruiken hetzelfde wachtwoord voor andere doeleinden (Tweakers, 2018). Om deze reden is het gebruikelijk om het wachtwoord zelf niet op te slaan. Het opslaan van de wachtwoorden gebeurt in drie stappen.

- Er wordt een SHA-512 hash berekend over het wachtwoord;
- Dan wordt dit door BCrypt gehaald;
- Het eindresultaat wordt versleuteld met AES.

Op deze manier is het praktisch niet mogelijk om het originele wachtwoord te kunnen achterhalen en het originele wachtwoord staat niet in de database.

4.3 Authenticatie

Voor de authenticatie wordt er een gebruikersnaam en wachtwoord gebruikt. Een gebruiker heeft ook een ‘displayname’, deze wordt in de applicatie gebruikt en deze zien andere gebruikers ook. De gebruikersnaam wordt alleen gebruikt om in te loggen. Dit is gedaan zodat de gebruikersnaam van een gebruiker niet zichtbaar is en daarom moeilijker is om inlogpogingen te doen op iemands account. Voor extra security hebben gebruikers de mogelijkheid om Google two factor authenticatie (2FA) in te schakelen. Als deze geactiveerd wordt, krijgt de gebruiker een QR-code die gescand kan worden met de Google authenticator app op zijn/haar telefoon. Als deze is ingeschakeld en de gebruiker inlogt, dan moet de gebruiker de code van de Google authenticator app achter het huidige wachtwoord aan plakken. De reden hiervoor is dat het dan niet mogelijk is om te zien wanneer de gebruikersnaam en wachtwoord goed zijn. Als voorbeeld; iemand probeert het gebruikersnaam en wachtwoord te raden, mocht dit lukken en er wordt daarna om een code gevraagd, dan weet diegene dat de huidig geraden inloggegevens kloppen. Als de code achter het wachtwoord geplakt zit, is het niet mogelijk om hier achter te komen en vrijwel onmogelijk is om dit te brute forcen omdat het hele wachtwoord iedere 60 seconden

veranderd. Deze methode wordt ook vaker gebruikt, een voorbeeld hiervan is OpenVPN.

4.4 Autorisatie

Voor de authorisatie krijgt de gebruiker na het succesvol inloggen een JWT-token. Deze tokens worden uitgedeeld aan de clients door de server, maar worden ook in server zelf bijgehouden met de bijbehorende sleutel en sessie. Als er gerommeld is met deze token dan zal de server deze niet meer kunnen verifiëren en wordt de token ongeldig verklaard met als resultaat dat die ingelogde sessie uit de server wordt verwijderd waardoor de client bij iedere volgende request wordt genereerd. Tevens wordt de client uitgelogd. Mocht een kwaadwillig iemand een andere 'geldige' token kunnen bemachtigen en deze probeert te gebruiken op een andere inlog, dan komt deze token niet meer overeen met de sleutel van de sessie, en wordt deze ook verwijderd. De JWT-token zelf wordt ook als versleuteld AES-verkeer verstuurd, dit maakt het onderscheppen en aanpassen van een dergelijke token niet mogelijk.

5 Conclusie

Bij het maken van deze applicatie is al vroegtijdig nagedacht over hoe de implementatie gedaan moet worden, welke security maatregelen nodig zijn en wat de applicatie precies moet gaan doen. Door over veel van de besproken aspecten vroegtijdig na te denken worden er mogelijke problemen voorkomen die in een later stadium lastiger te verwerken zijn. Ook is het belangrijk om te overleggen met de product owner en ook te kijken naar wat de doelgroep is waarvoor deze applicatie gericht is.

Literatuurlijst

Gillz. (n.d.). *Stel jouw app beschikbaar voor Dark mode.* Geraadpleegd op 20 september 2020 van https://www.gillz.nl/kennisbank/ondersteun-mijn-app-in-dark-mode/

Tweakers. (2018, 5 maart). *'Veertig procent van Nederlanders hergebruikt wachtwoord'.* Geraadpleegd op 20 september 2020 van https://tweakers.net/nieuws/135931/veertig-procent-van-nederlanders-hergebruikt-wachtwoord.html

Afkortingen en woordenlijst

2FA	Two Factor Authenticatie.
AES	Advanced Encryption Standard.
AVG	Algemene Verordening Gegevensbescherming
DDoS	Distributed Denail of Service.
Felix	De naam van de applicatie.
JWT	Json Wep Token.
MitM	Man in the Middle.
RSA	Ron **R**ivest, Adi **S**hamir en Len **A**dleman.
SHA-512	Secure Hashing Algorithm 512 bit.