CTF-CHALLENGE
TEAM: FLAGFORGE

Design document

Ruben Croes,
Quinten De Meyer,
Jorg Maas,
Brent Paessens,
Bekirhan Simsek,
Quinten Van der Wildt

2025 - 2026

Inhoudstafel

1. Inleiding

Dit document heeft als doel een volledig overzicht geven van het ontwerp van ons project, met specifiek meer aandacht voor het design system dat als basis dient voor de visuele en functionele consistentie binnen ons project.

Daarnaast biedt dit document een gedetailleerde analyse van de huidige situatie, gevolgd door een opsplitsing van het algemeen ontwerp, het technisch ontwerp, data engineering, data analyses, beveiligingsaspecten en DevOps richtlijnen.

Elk hoofdstuk beschrijft de relevante architecturen, modellen, schema's en beslissingen die noodzakelijk zijn voor een duidelijk en gestructureerde uitvoering van het project.

1.1. Huidige systemen en infrastructuur

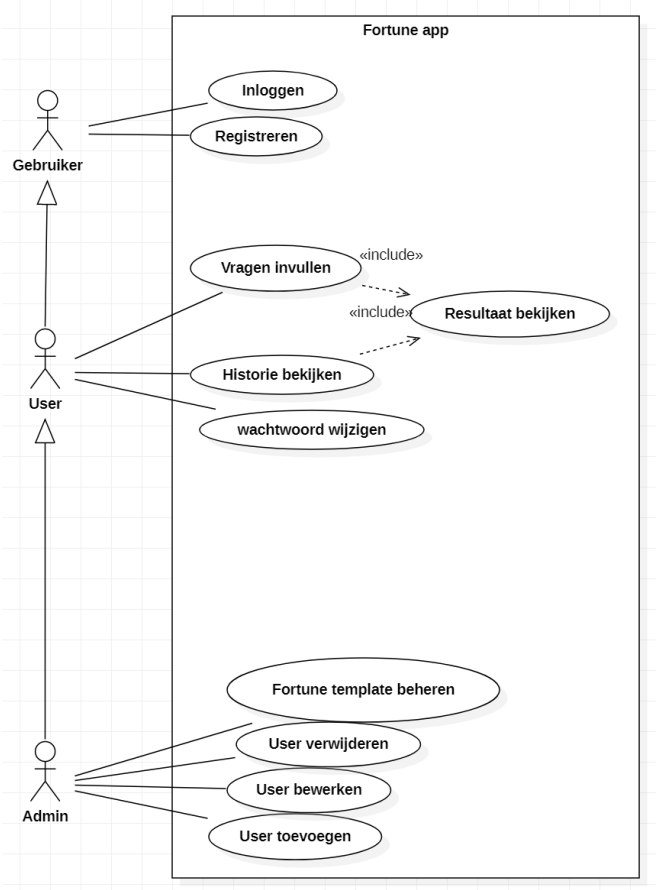
Het CTF-project is een volledig nieuw initiatief (greenfield-project). Er is momenteel geen bestaande infrastructuur beschikbaar om de challenges te hosten, en de organisatie beschikt niet over eigen servers of netwerkmiddelen die direct kunnen worden ingezet. Alle benodigde infrastructuur zal vanaf de basis moeten worden opgezet, bijvoorbeeld via cloudhosting of door gebruik te maken van apparatuur die door de school beschikbaar wordt gesteld. Er zijn op dit moment geen budgetten beschikbaar voor de aanschaf van nieuwe infrastructuur of langdurige cloudresources.

2. Algemeen Ontwerp

In dit hoofdstuk wordt het algemene ontwerp van het project beschreven. Het project omvat twee afzonderlijke applicaties, die inhoudelijk en technisch geen directe koppeling met elkaar hebben. Voor elke applicatie wordt de globale structuur toegelicht aan de hand van schema's, use cases, datamodellen en schermprototypes. De nadruk ligt hierbij op de functionele opbouw en logica van de applicaties, niet op de gebruikte technologieën.

2.1. Use Case Diagram en User Stories

2.1.1. Fortune applicatie:



2.1.1.1. Inloggen

Functionaliteit: Als gebruiker, Kan ik inloggen.

Normale flow: Het systeem toont een inlog pagina. De actor vult gegevens in. Het systeem toont het start scherm.

Alternatieven:

- Registreren: De actor klikt op account aanmaken. Het systeem toont scherm om account aan te maken. De actor vult dit in.

2.1.1.2. Vragen invullen

Functionaliteit: Als user, Kan ik vragen invullen.

Normale flow: Het systeem toont een lijst met vragen. De actor vult de vragen in. Het systeem toont aan de hand van een score het resultaat.

2.1.1.3. Historie bekijken

Functionaliteit: Als user, Kan ik mijn historie bekijken.

Normale flow: Het systeem toont een lijst van vorige sessies. De actor selecteert een sessie. Het systeem toont de sessie met de ingevulde vragen.

2.1.1.4. User toevoegen

Functionaliteit: Als admin, Kan ik users toevoegen.

Normale flow: Het systeem toont een lijst van users. De actor voegt een user toe door op de knop te drukken. Het systeem toont een scherm waar je de gegevens kunt ingeven om een nieuwe gebruiker toe te voegen.

2.1.1.5. User bewerken

Functionaliteit: Als admin, Kan ik users bewerken.

Normale flow: Het systeem toont een lijst van users. De actor selecteert een user. Het systeem toont een scherm waar je de gegevens kunt bewerken.

2.1.1.6. User verwijderen

Functionaliteit: Als admin, Kan ik users verwijderen.

Normale flow: Het systeem toont een lijst van users. De actor selecteert een user. Het systeem toont een scherm met een knop. De actor drukt op de knop. Het systeem toont een pop-up om te bevestigen. De actor kiest bevestig. Het systeem toont dat deze succesvol is verwijderd.

2.1.1.7. Fortune template beheren

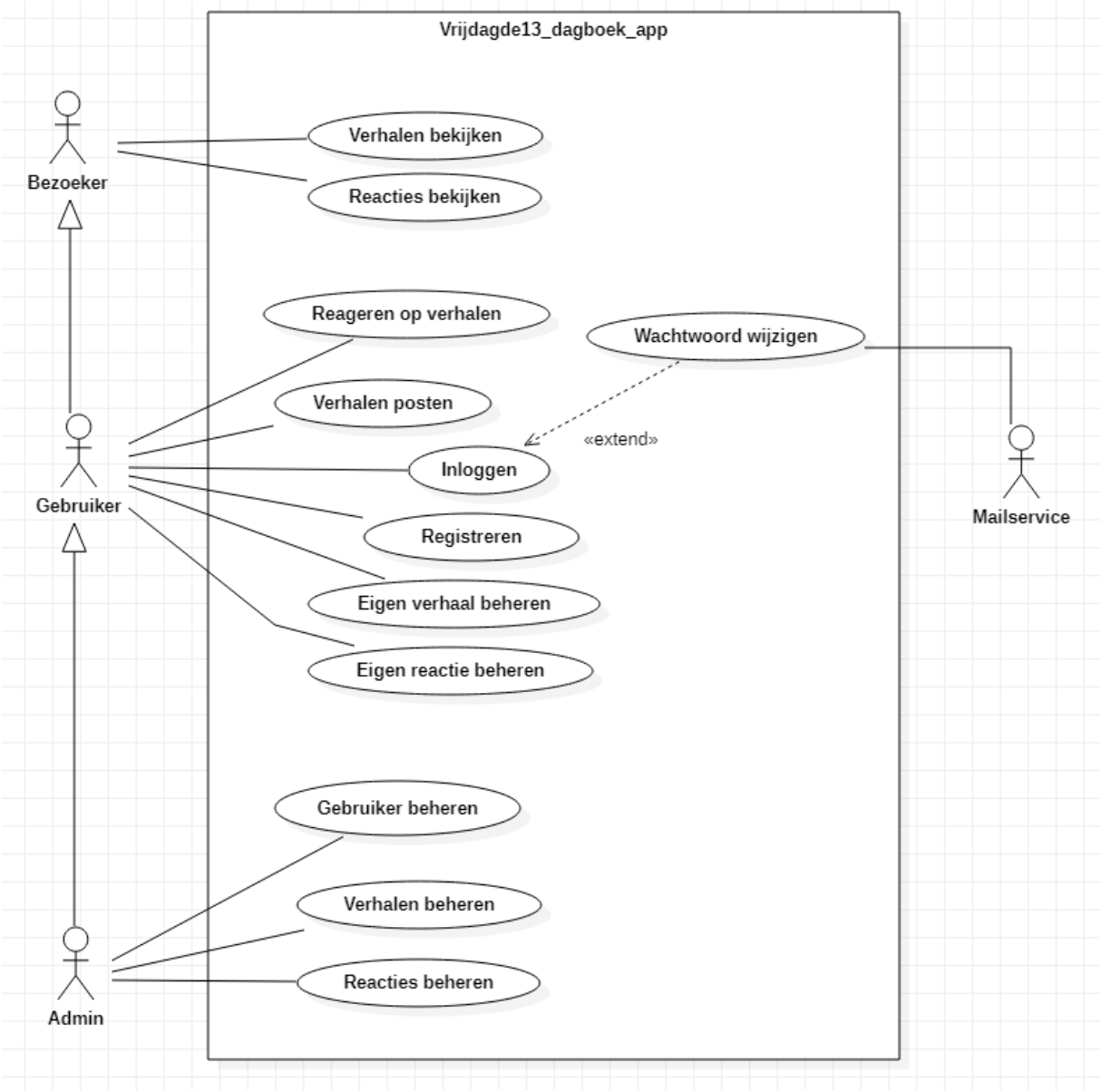
Functionaliteit: Als admin, Kan ik fortune templates beheren.

Normale flow: Het systeem toont een lijst van alle fortune templates. De actor selecteert een template. Het systeem toont extra info over de template.

Alternatieven:

- Toevoegen: De actor kan een template toevoegen door op een knop de drukken. Het systeem toont de lijst met de nieuwe template.
- Bewerken: De actor kan een huidige template aanpassen. Het systeem toont het aangepaste template.
- Verwijderen: De actor kan een template verwijderen. Het systeem toont dat dit verwijderd is.

2.1.2. Dagboek Applicatie



2.1.2.1. Verhalen bekijken (Bezoeker, Gebruiker)

Functionaliteit: De actor kan een lijst van gepubliceerde verhalen bekijken.

Normal Flow: Actor navigeert naar de verhalenpagina. Systeem haalt de lijst van verhalen op en toont verhalen met titel, auteur en datum. Actor kan volledige inhoud van een berhaal bekijken.

Alternative Flows:

- Geen verhalen beschikbaar = Systeem toont melding: “Er zijn nog geen verhalen geplaatst.”

2.1.2.2. Reacties bekijken (Bezoeker, Gebruiker)

Functionaliteit: De actor kan reacties lezen onder een verhaal.

Normal Flow:

Actor navigeert naar de verhalenpagina. Systeem haalt chronologisch bijhorende reacties op.

Alternative Flows:

- Geen reacties beschikbaar

2.1.2.3. Registreren (Gebruiker)

Functionaliteit: Een actor kan een nieuw account maken als gebruiker.

Normal Flow:
De actor opent registratieformulier. Het systeem vraagt om een naam, email en wachtwoord. Gebruiker vult gegevens in (naam, e-mail, wachtwoord).Systeem valideert invoer. Systeem maakt nieuw account aan en toont bevestiging.

Alternative flow

- Ongeldige invoer = Systeem toont foutmelding (bijv. ongeldig e-mailadres).
- E-mail al in gebruik= Systeem toont foutmelding.

2.1.2.4. Inloggen (Gebruiker, Admin)

Functionaliteit: Actor kan inloggen tmet e-mail en wachtwoord.

Preconditions: Gebruiker heeft een geregistreerd account.

Normal Flow:

Actor opent het inlogscherm. Het systeem vraagt om wachtwoord en email. Actor voert e-mail + wachtwoord in. Systeem valideert de combinatie. Systeem creëert sessie en logt actor in. Actor wordt doorgestuurd naar de startpagina.

Alternative Flows:

- Foute inloggegevens = Systeem toont: “Ongeldige combinatie.”
- Account gedeactiveerd = Systeem toont: “Uw account is niet actief.”
- De actor wilt wachtwoord veranderen

2.1.2.5. Wachtwoord wijzigen

Functionaliteit : Ingelogde gebruiker/Admin kan eigen wachtwoord wijzigen.

Preconditions: Actor heeft een bestaand account

Normal Flow:

Actor opent pagina wachtwoord vergeten. Het systeem vraagt om een gekoppeld email. De actor voert een gekoppeld email in. Het systeem valideert en stuurt een email naar het emailadress. De actor volgt de link uit de email. Het systeem vraagt om een nieuw wachtwoord en herhaling van het wachtwoord. Actor voert het nieuw wachtwoord in en herhaalt het wachtwoor. Systeem valideert beide. Systeem slaat het nieuwe wachtwoord op en toont bevestiging.

Alternative Flows:

- Het opgegeven email is incorrect = Systeem toont foutmelding.
- Nieuw wachtwoord voldoet niet = voldoet niet aan het vereist formaat

2.1.2.6. 6. Verhalen posten (Gebruiker)

Functionaleit = Ingelogde gebruikers kunnen een nieuw verhaal plaatsen.

Preconditions = De actor is ingelogd als gebruiker.

Normal Flow:

Actor opent ‘Nieuw verhaal’-pagina. Het systeem vraagt om Naam schrijver, email, titel verhaal, verhaal beschrijving en ongeluksniveau. Actor vult informatie in. Systeem valideert invoer. Systeem slaat het verhaal op en toont bevestiging en publiceert verhaal.

Alternative Flows

- Ongeldige invoer = (bijv. lege titel) → foutmelding.

2.1.2.7. 7. Reageren op verhalen (Gebruiker)

Functionaliteit = Ingelogde gebruiker kan een reactie plaatsen bij een verhaal.

Preconditions = de actor is ingelogd als gebruiker.

Normal Flow

Actor opent een verhaal en typt een reactie. Systeem valideert de reactie en slaat reactie op. Reactie verschijnt onder het verhaal.

Alternative Flows

- Reactie leeg = Foutmelding.
- Verhaal verwijderd tijdens actie **=** Systeem toont melding.

2.1.2.8. 8. Gebruiker beheren (Admin)

Functionaliteit = actor admin kan accounts bekijken, verwijderen of aanpassen.

Preconditions = actor is ingelogd als admin.

Normal Flow:

Actor opent gebruikerspaneel.Systeem toont lijst met gebruikers. Actor kiest een actie (verwijderen/aanpassen/toevoegen). Systeem voert actie uit. Systeem toont bevestiging.

Alternative Flows

- Admin probeert zichzelf te verwijderen = Systeem blokkeert actie.

2.1.2.9. 9. Verhalen beheren (Admin)

Functionaliteit = Actor Admin kan verhalen plaatsen en verwijderen.

Preconditions = De actor is ingelogd als admin.

Normal Flow:

Actor opent verhalenpagina. Systeem toont alle verhalen. Actor kiest verhaal en actie verwijderen/aanpassen/toevoegen. Systeem voert de actie uit. Systeem bevestigt de wijziging.

Alternative flow:

- Gebruiker heeft verhaal verwijderd

2.1.2.10. 10. Reacties beheren (Admin)

Functionaliteit = Actor Admin beheert reacties (verwijderen, bewerken en posten).

Preconditions = actor is ingelogd als admin.

Normal Flow:

Actor opent gaat naar de verhaalpagina. Systeem toont lijst van verhalen met bijhorende reacties. Actor kiest actie verwijderen/aanpassen/toevoegen. Systeem voert de actie uit en bevestiging wordt getoond.

Alternative flow:

- Gebruiker heeft reactie verwijderd

2.1.2.11. 10. Reactie beheren (gebruiker)

Functionaliteit = Actor gebruiker beheert eigen reactie (verwijderen en aanpassen).

Preconditions = actor is ingelogd als gebruiker.

Normal Flow:

Actor opent gaat naar de verhaalpagina. Systeem toont lijst van verhalen met bijhorende reacties. Actor kiest actie (verwijderen of aanpassen) voor zijn/haar eigen reactie. Systeem controleerd op overeenkomst tussen reactie en gebruiker, voert actie uit en bevestiging wordt getoond.

2.1.2.12. 9. Verhalen beheren (Gebruiker)

Functionaliteit = Actor Admin kan verhalen plaatsen, aanpassen en verwijderen.

Preconditions = De actor is ingelogd als admin.

Normal Flow:

Actor opent verhalenpagina. Systeem toont alle verhalen. Actor kiest verhaal en actie verwijderen/aanpassen voor zijn/haar eigen verhaal. Systeem controleerd op overeenkomst tussen verhaal en gebruiker, voert actie uit en bevestiging wordt getoond.

Alternative flow:

- Gebruiker heeft verhaal verwijderd

2.1.2.13. 11. Mailservice (Extern systeem)

Functionaliteit: Actor Mailservice stuurt bevestigings- en notificatiemails.

Normal Flow:

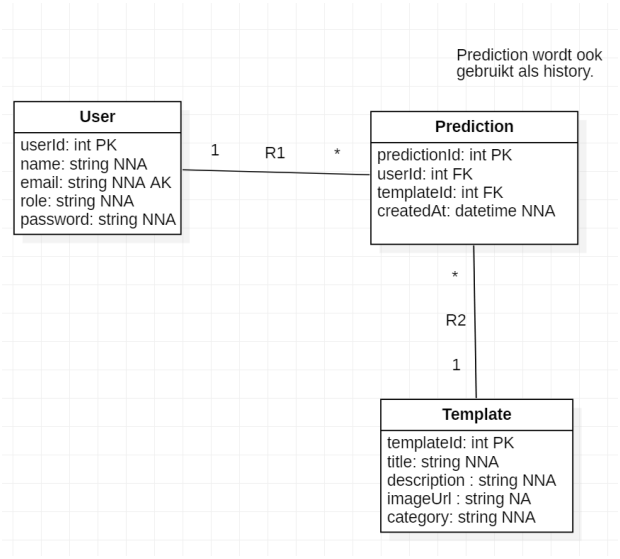
Applicatie genereert mailverzoek. Mailservice ontvangt verzoek. Mailservice verstuurt mail. Mailservice geeft succes of fout terug.

Alternative Flows:

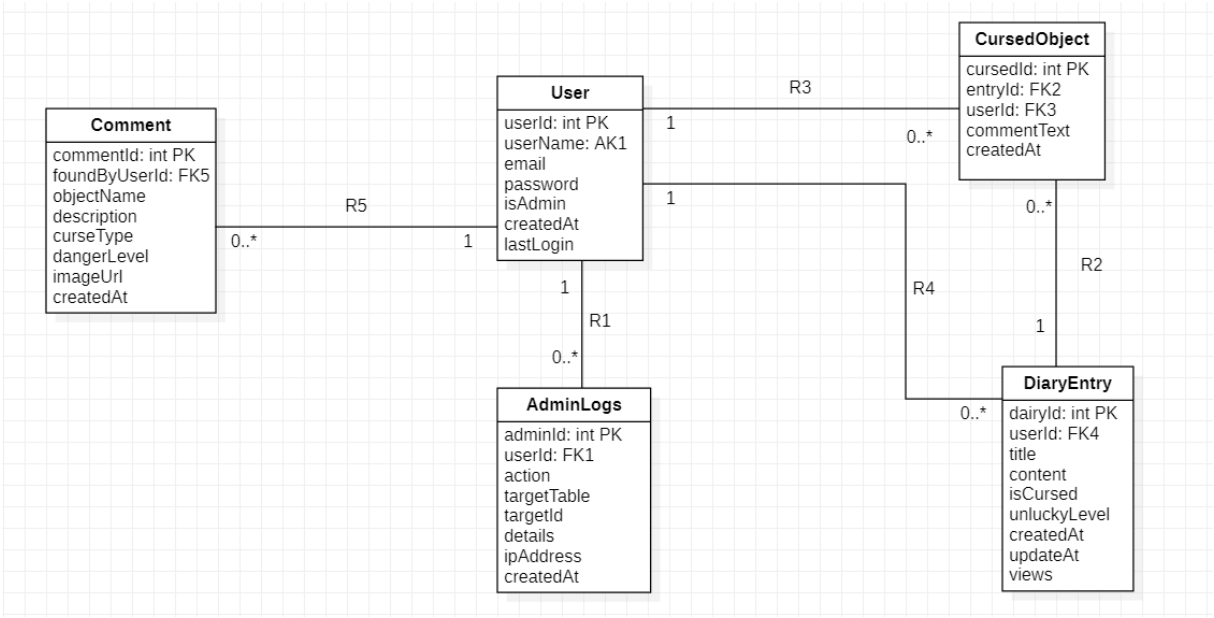
- Mailserver niet bereikbaar = Applicatie logt fout en geeft melding aan gebruiker.

2.2. Datamodel

Fortune applicatie:



Dagboek applicatie:

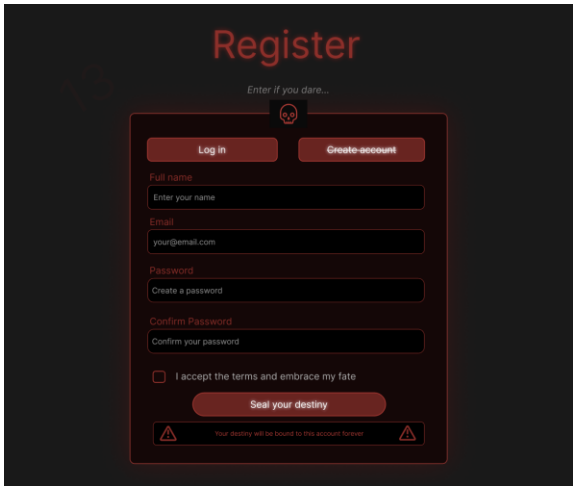


2.3. Prototypes/Wireframes

2.3.1. Fortune applicatie

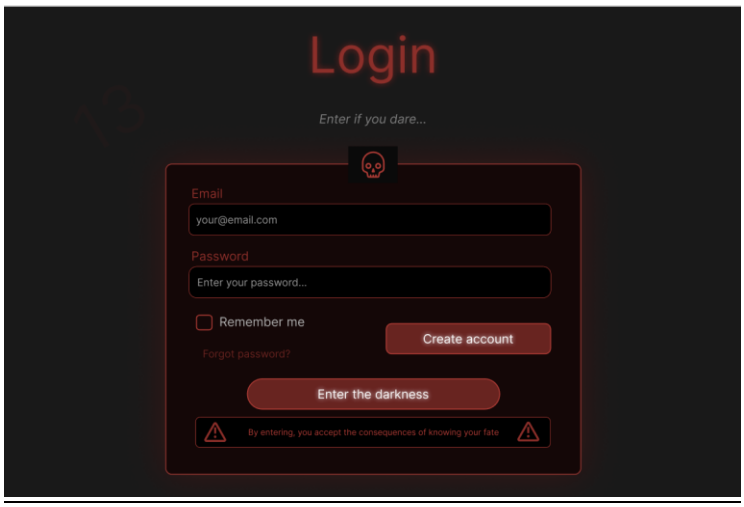
Figma link voor beter inzicht :
https://www.figma.com/design/4uYsycHEHRFtbvXUdLYBxy/Untitled?node-id=1-36&t=SW2B7M1Y5DGq5piR-1

2.3.1.1. Registreren

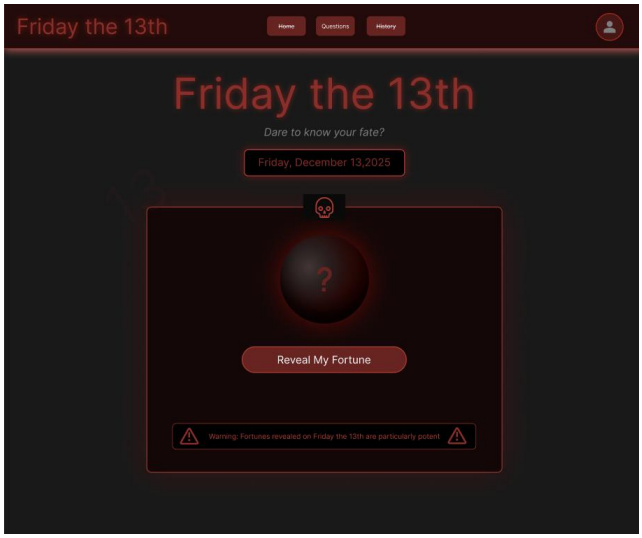


Op deze pagina kan de gebruiker zich aanmelden of registreren.
De login zorgt ervoor dat elke gebruiker zijn eigen voorspellingen en geschiedenis heeft.
Zonder account kan je de applicatie niet gebruiken, omdat alle resultaten gekoppeld zijn aan een specifieke gebruiker.

2.3.1.2. Login Pagina

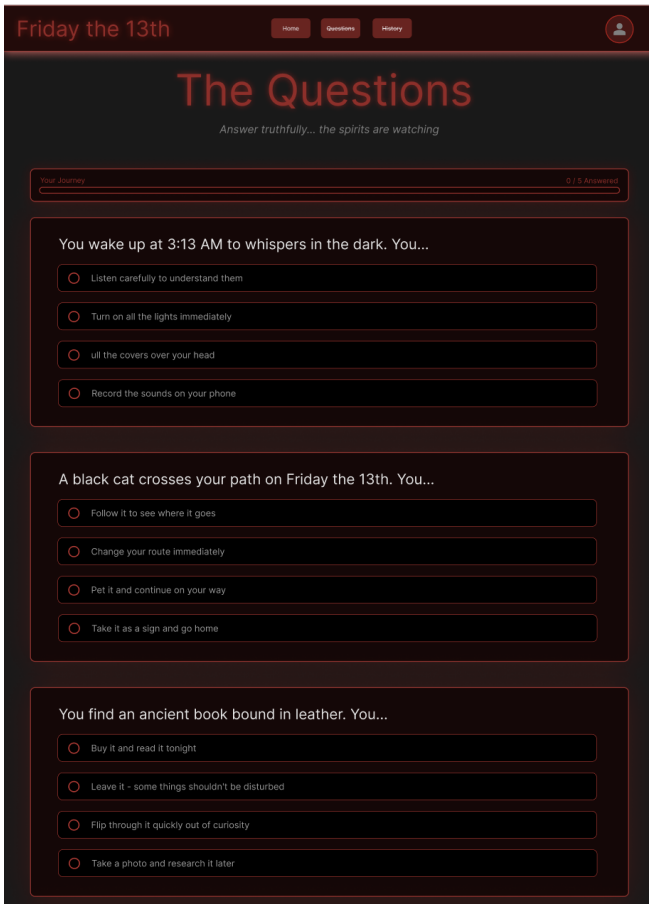


2.3.1.3. Home pagina



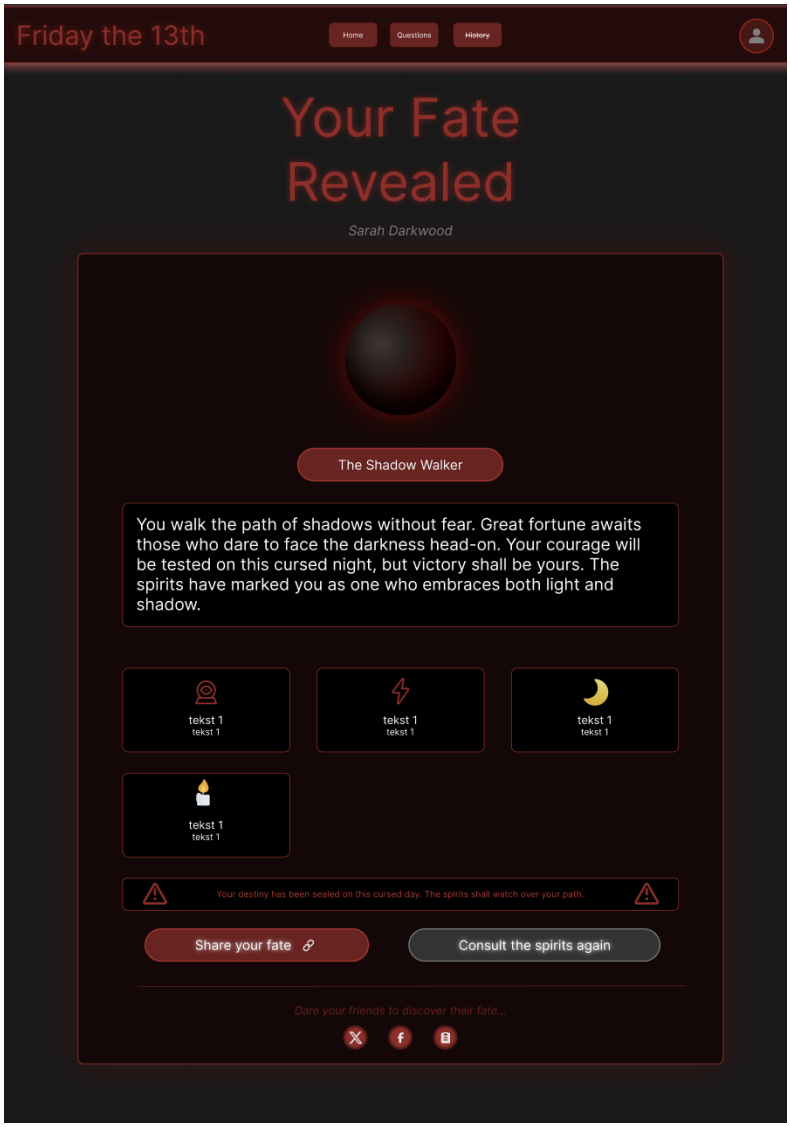
Na het inloggen komt de gebruiker op de homepagina terecht.
Hier krijgt hij een kort overzicht van wat de applicatie doet en kan hij kiezen om een nieuwe voorspelling te starten of zijn geschiedenis te bekijken.
Dit scherm dient als vertrekpunt naar de belangrijkste functionaliteiten van de app.

2.3.1.4. Question pagina



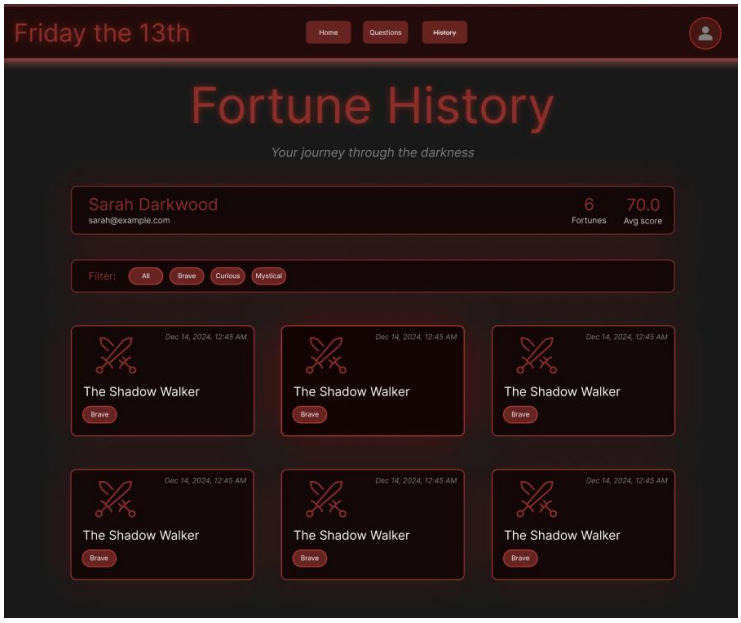
Op deze pagina beantwoordt de gebruiker een reeks vragen.
De vragen worden gebruikt als input voor de voorspelling.
De gebruiker vult een formulier in met meerdere velden, bijvoorbeeld meerkeuzevragen of invulvragen.
Op basis van deze antwoorden wordt later één resultaat teruggegeven.

2.3.1.5. Result Pagina



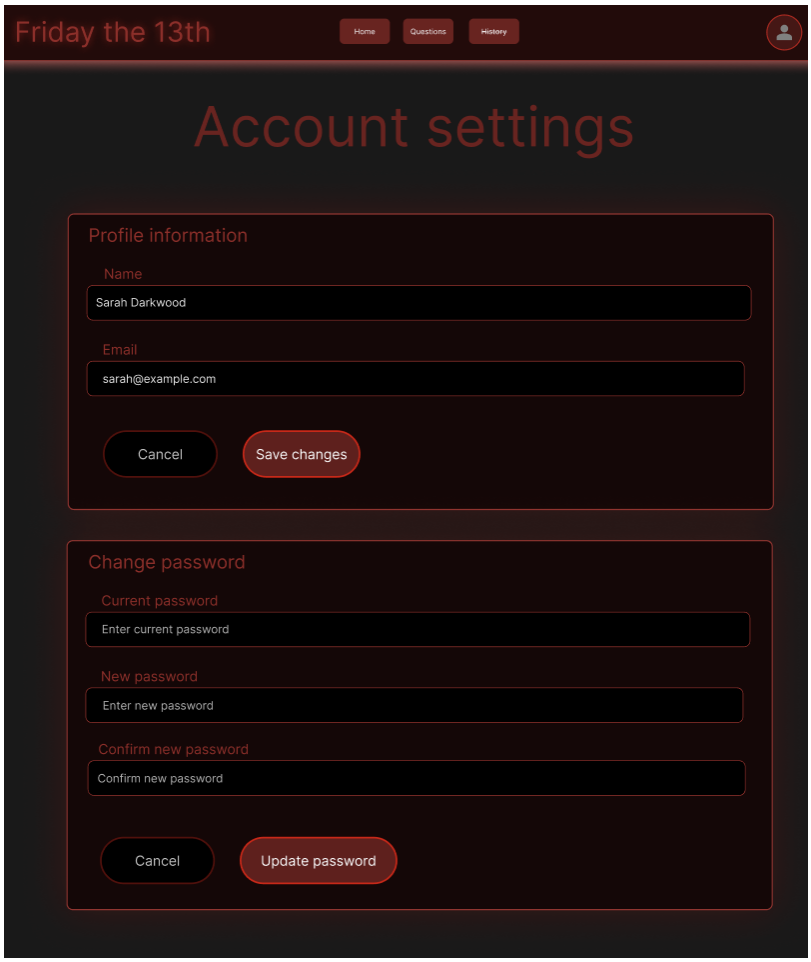
Nadat de gebruiker de vragen heeft ingevuld, komt hij op de resultaatpagina.
Hier wordt één voorspelling getoond, op basis van de input van de gebruiker.
Het resultaat wordt in een duidelijke layout weergegeven, eventueel met een titel, korte uitleg en extra context.
Dit resultaat wordt ook opgeslagen, zodat de gebruiker het later kan terugvinden in zijn geschiedenis.

2.3.1.6. Historie Pagina



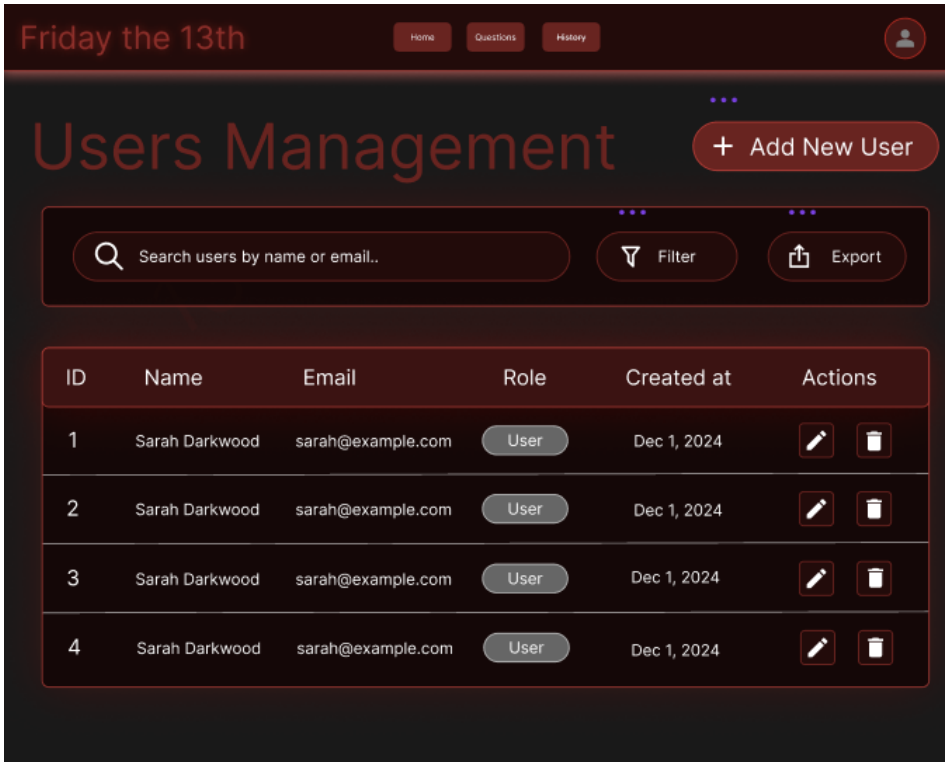
Op de geschiedenis-pagina krijgt de gebruiker een overzicht van al zijn eerdere voorspellingen. Zo kan de gebruiker achteraf zien welke voorspellingen hij in het verleden gekregen heeft.

2.3.1.7. Account Pagina



De Gebruiker kan wachtwoord of email, veranderen in de account settings.

2.3.1.8. Admin Pagina



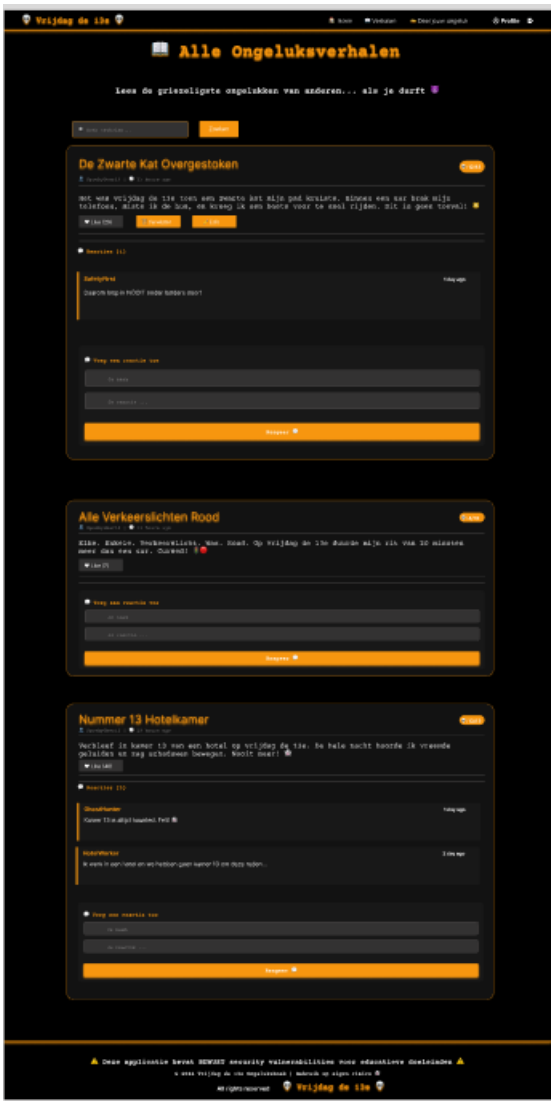
Hier kan de admin gebruikers toevoegen bewerken of verwijderen.
met mogelijk om te filteren op naam en email.

2.3.2. Dagboek applicatie prototype

Figma link voor beter inzicht :
https://www.figma.com/design/iEtJXu038JBy255wH2bswm/Untitled?node-id=0-1&t=DY5b1m0oam7t1lu3-1

2.3.2.1. Home page

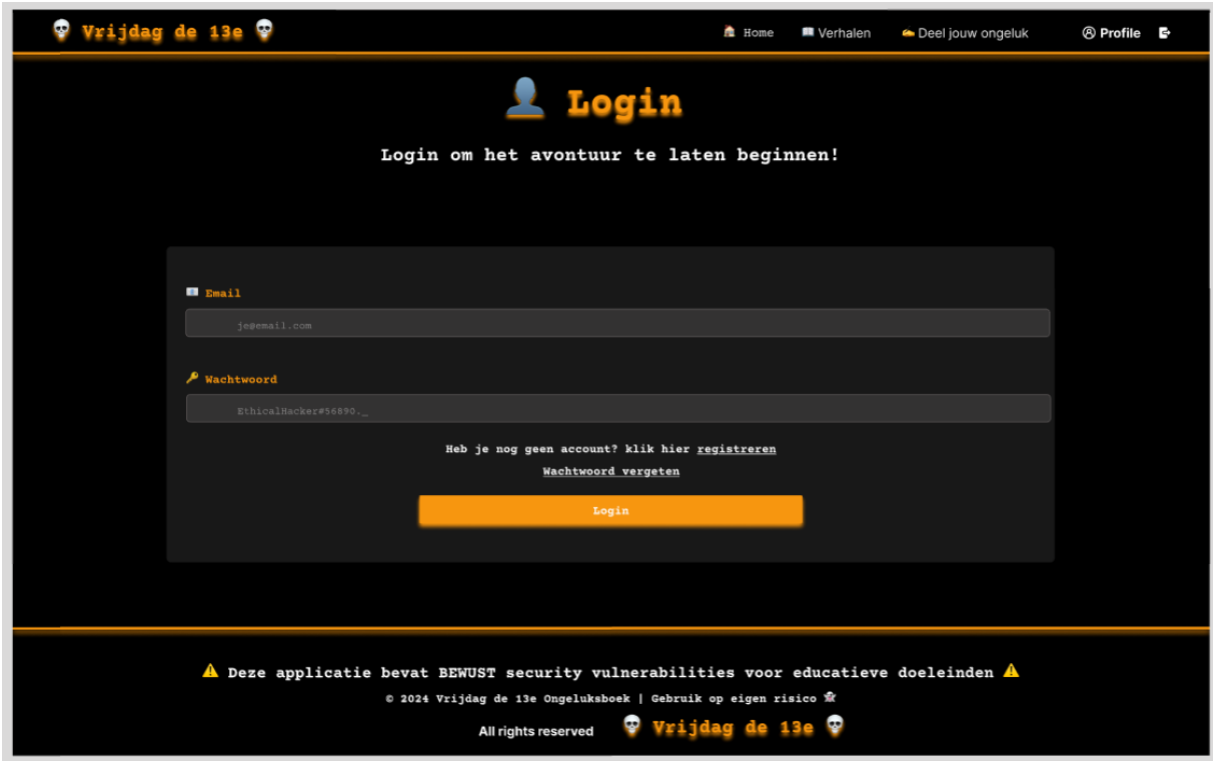
2.3.2.2. Verhalen pagina



2.3.2.3. Creëer verhaal pagina

2.3.2.4. *Profile pagina*

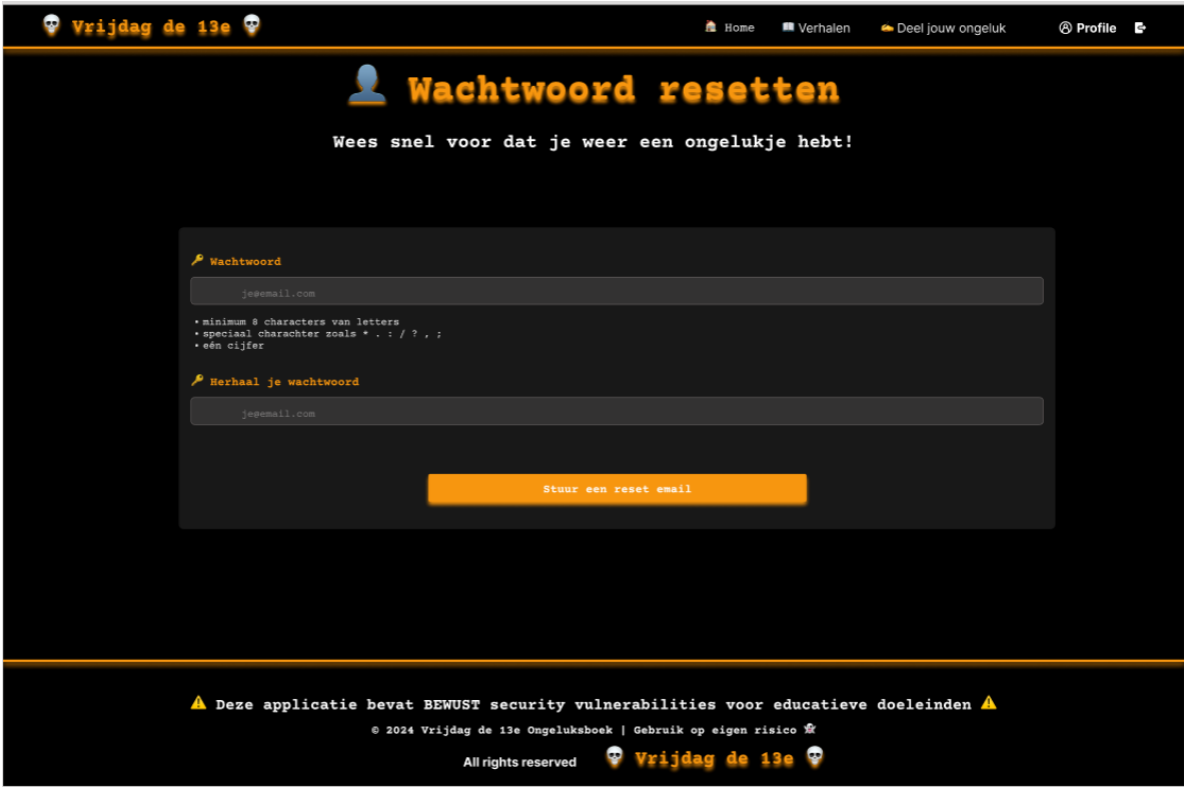
2.3.2.5. Login pagina



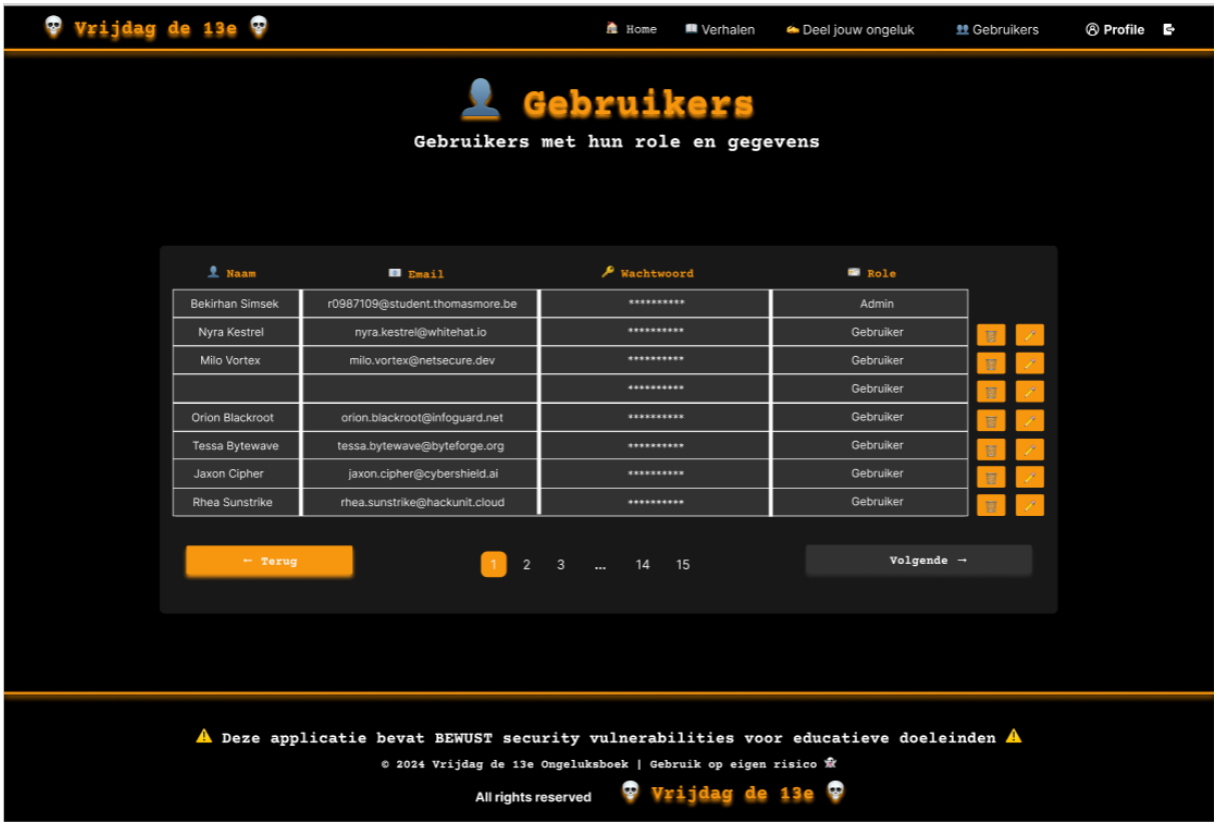
2.3.2.6. Registreren pagina

2.3.2.7. Wachtwoord vergeten

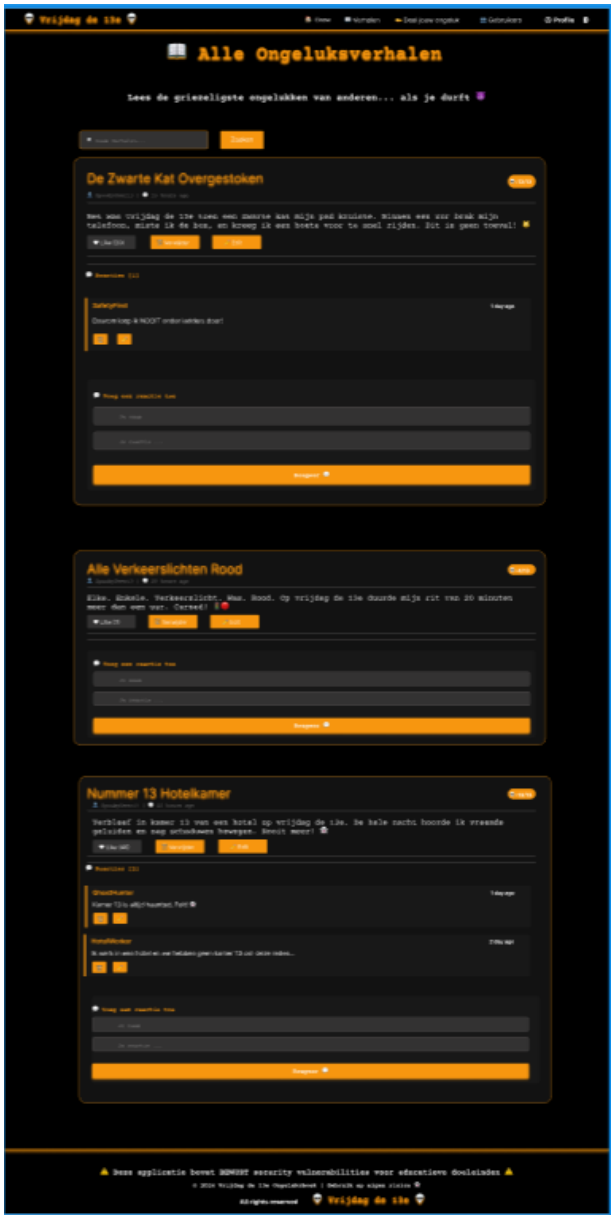
2.3.2.8. Wachtwoord resetten



2.3.2.9. Admin gebruikers beheren pagina



2.3.2.10. Admin Verhaal pagina met edit, delete voor alle andere gebruikders



3. Technisch Ontwerp

3.1. Infrastructuur

Doelarchitectuur

Diagram indien we meerdere minder krachtige servers ter beschikking hebben:

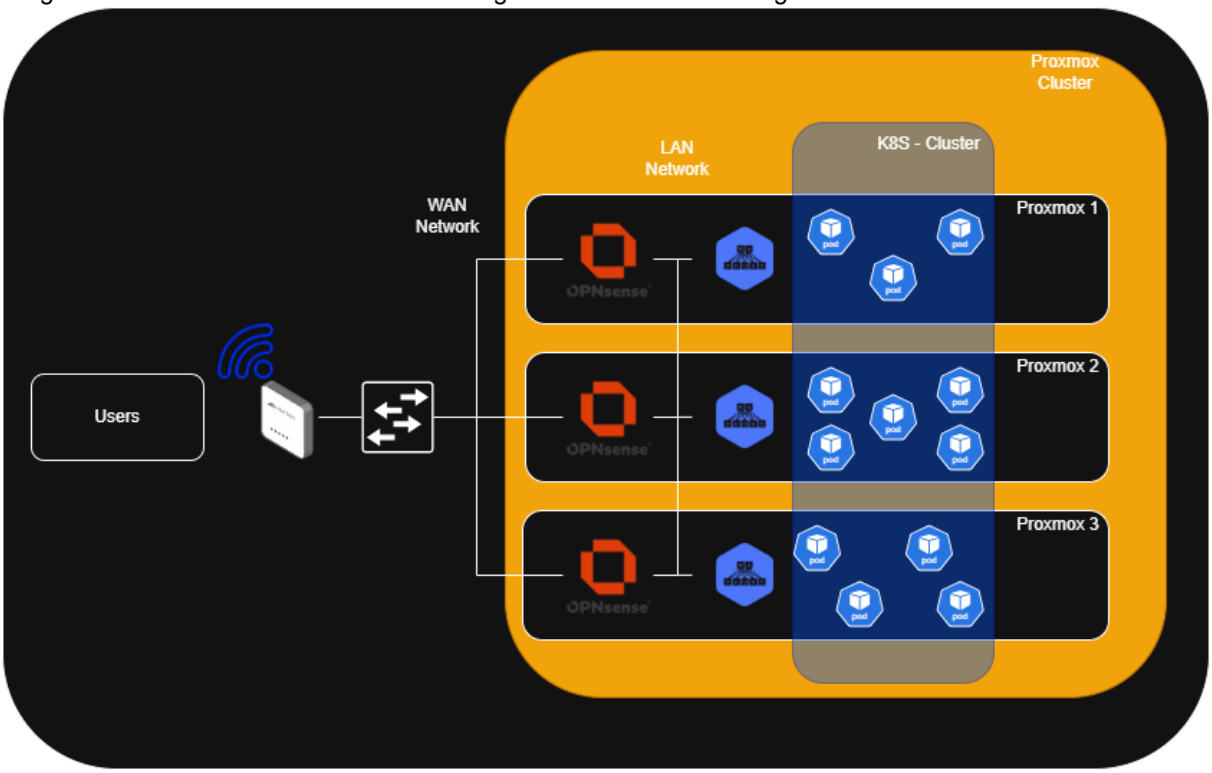
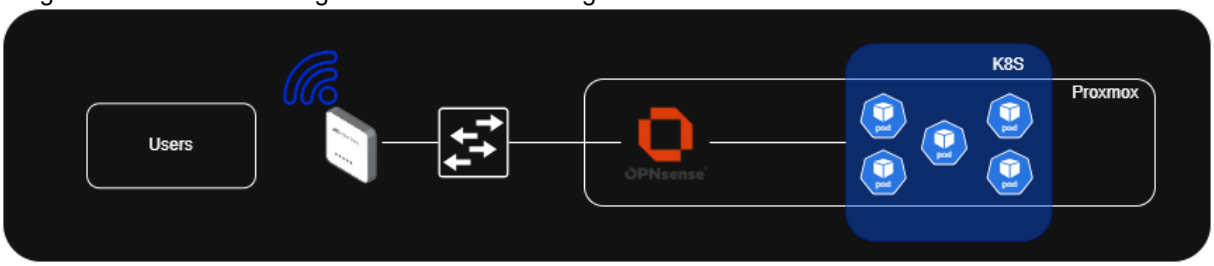
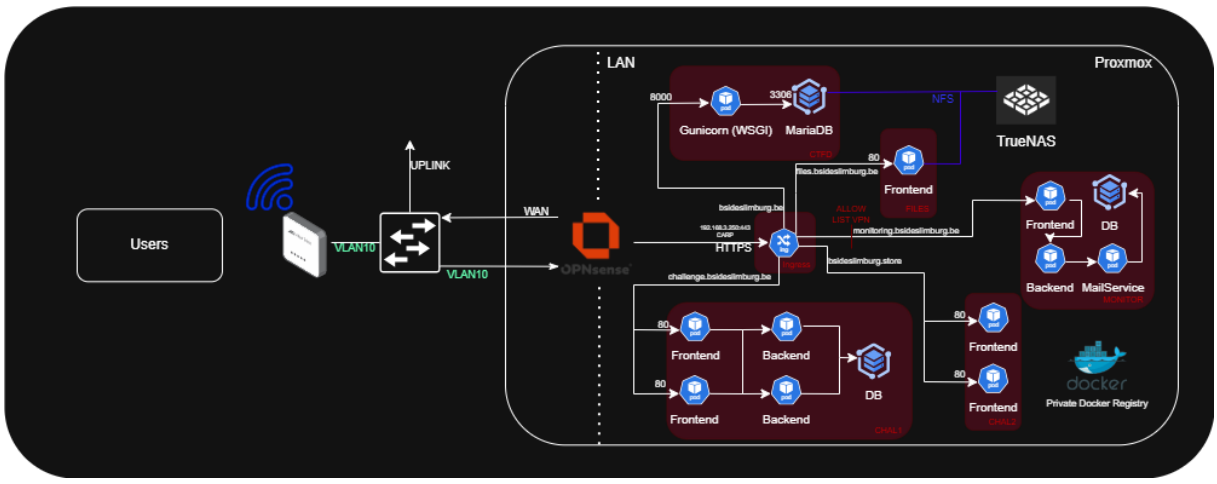


Diagram indien we 1 krachtige server ter beschikking hebben:

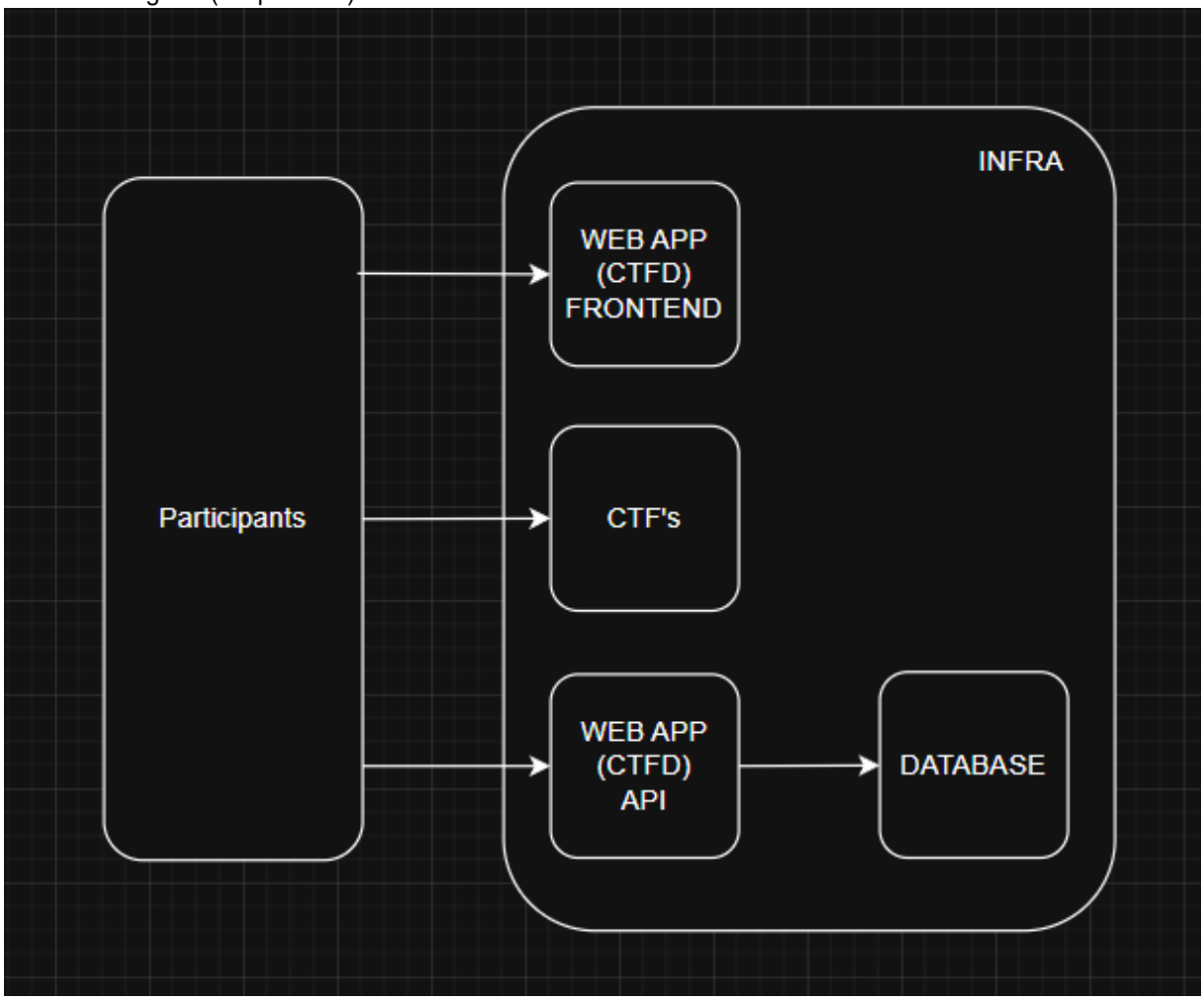


Netwerk schema voor 1 server:



Het opzet blijft hetzelfde: de gebruikers connecteren met onze setup via 1 (of meerdere) AP. Deze zullen via een switch doorverbonden worden aan onze server (of meerdere servers indien setup 1). Het inkomende verkeer zal binnen komen via een OPNsense firewall VM. Deze zal via port forwarding de requests doorsturen naar de loadbalancer (metallb) van de kubernetes cluster. Deze cluster zal bestaan uit 3 master VM's, en worker nodes zoveel we nodig hebben, dit moeten we zien wanneer we de effectieve challenges hebben. De data die naar ons dashboard gaat zal geencrypteerd zijn (HTTPS met geldig certificaat). De challenges zelf zullen ook over HTTPS gaan, maar met een self signed certificaten.

Dataflow Diagram (simplistisch):



We zien op dit diagram dat de deelnemers via HTTPS naar de frontend kunnen connecteren. Ze kunnen ook via HTTPS naar de CTF's en naar de API van CTFD connecteren. De database kan enkel intern aangesproken worden door de API, niet door gebruikers of externen rechtstreeks.

3.1.1. Verantwoording van architectuurkeuzes

	Firewall				
	Weighting	PfSense	OPNsense	Iptables	FortiOS
Price	5	10	10	10	3
Complexity	3	5	6	2	6
Knowledge	3	5	5	3	2
Resources	3	6	6	10	6
Performance	3	6	6	10	8
FeatureSet	2	6	8	4	9
Total		128	135	133	99

	Pods			
	Weighting	Docker	K8s	Bare Metal
Hardware	5	7	6	8
Complexity	3	6	5	8
Knowledge	3	6	6	7
Performance	3	6	6	8
FeatureSet	7	7	10	4
Total		138	151	137

We hebben voor OPNsense en Kubernetes gekozen.

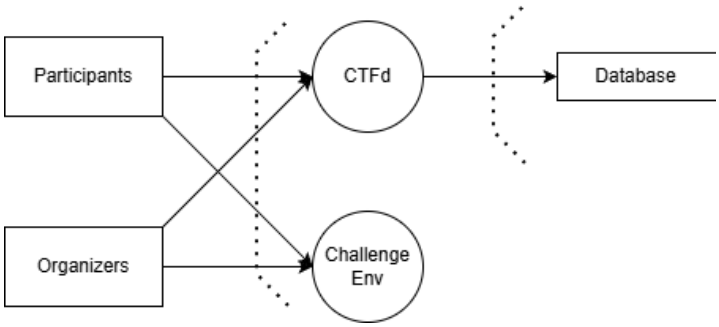
- Beschikbaarheid + Automatisch schalen + Automatisch herstel: We gebruiken hiervoor kubernetes
  - Beheerfunctionaliteit + Gebruiksvriendelijke interface: We gebruiken hiervoor CTFD, dit omdat het aangeraden was door de klant en omdat we zelf de tijd niet hebben om een platform te ontwikkelen van 0. Hierdoor is hier geen WRM van gemaakt.

3.2. Security Considerations

3.2.1. Data Flow Diagrams

The first step is sketching a high level overview of the application and its data. We do this trough the Data Flow Diagram method (https://en.wikipedia.org/wiki/Data-flow_diagram)

This is our level 1 DFD.



3.2.2. Trust boundaries

3.2.2.1. Participants -> CTFd

STRIDE Category	Threat list	Impact	Likelihood	Risk
Spoofing	Fake/impersonated accounts, session hijacking	4	1	4
Tampering	Manipulating HTTP requests, scoreboard tampering	5	3	15
Repudiation	Participants deny solve submissions or attacks	3	1	3
Information Disclosure	Flag leakage, user data exposure	4	4	16

Denial of Service	Brute-force flags, HTTP floods	4	4	16
Elevation of Privilege	Privilege bypass (user → admin)	5	2	10

3.2.2.2. Participants -> Challenge Environment

STRIDE Category	Threat list	Impact	Likelihood	Risk
Spoofing	IP spoofing, identity misrepresentation	3	1	3
Tampering	Challenge exploitation altering container state	2	2	4
Repudiation	Hard to track exploit origins per participant	2	1	2
Information Disclosure	Accessing flags via unintended paths	3	4	12
Denial of Service	Resource exhaustion inside pods (CPU/mem)	3	3	9
Elevation of Privilege	Container escape → cluster access	5	3	15

3.2.2.3. Organizers -> CTFd

STRIDE Category	Threat list	Impact	Likelihood	Risk
Spoofing	Compromised organizer credentials	5	2	10
Tampering	Altering scoreboard or challenge data improperly	4	2	8
Repudiation	Lack of admin activity logging	2	4	8
Information Disclosure	Exposure of player info, flags in admin UI	5	2	8
Denial of Service	Misconfiguration causing downtime	4	1	4
Elevation of Privilege	Admin interface misuse → underlying system access	5	1	5

3.2.2.4. Organizers -> Challenge Environment

STRIDE Category	Threat list	Impact	Likelihood	Risk
Spoofing	Unauthorized management access via shared accounts	4	2	8
Tampering	Altering challenge containers live	3	1	3
Repudiation	No audit logs for challenge modifications	2	3	6
Information Disclosure	Viewing or leaking challenge flags	4	3	12
Denial of Service	Stopping/restarting challenge pods accidentally	4	3	12
Elevation of Privilege	Over-permissive Kubernetes roles enabling cluster compromise	5	2	10

3.2.2.5. CTFd -> Database

STRIDE Category	Threat list	Impact	Likelihood	Risk
Spoofing	DB credential theft → direct database access	5	2	10
Tampering	SQL injection, flag tampering	4	3	12
Repudiation	Insufficient DB request logging	2	3	6
Information Disclosure	Data leak (users, tokens, flags)	4	3	12
Denial Of Service	Too much requests flood database	3	3	9
Elevation of Privilege	Insecure secret management	3	3	9

3.2.3. Risk: Impact x Likelihood

We assigned each threat an impact score and a likelihood score, then we prioritize them according to this table:

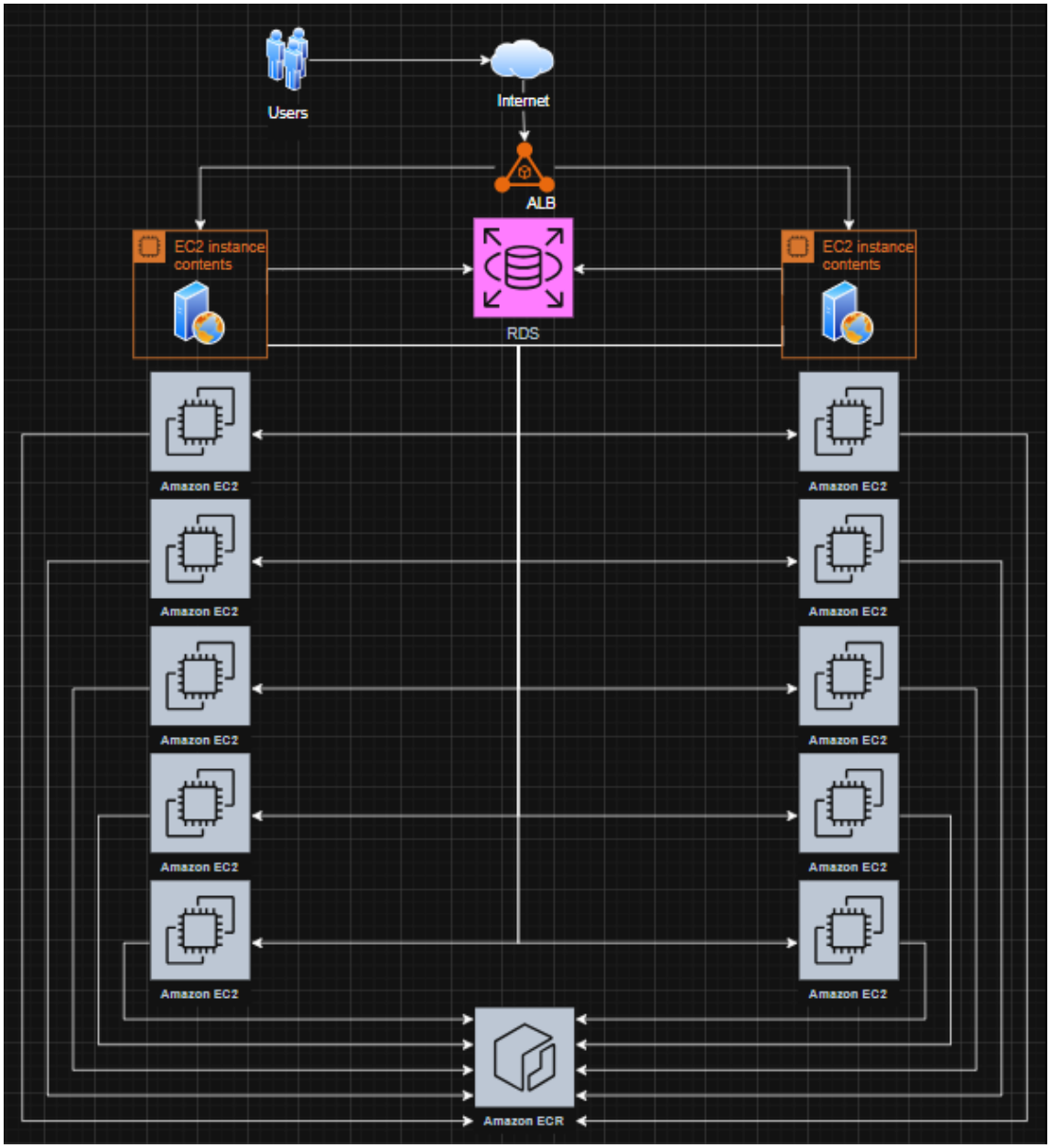
	Very unlikely	Unlikely	Possible	Likely	Very Likely
Negligible (1)	1	2	3	4	5
Minor (2)	2	4	6	8	10
Moderate (3)	3	6	9	12	15
Significant (4)	4	8	12	16	20
Severe (5)	5	10	15	20	25

Here is a table with the highest risk scores:

Threat	STRIDE Category	Mitigation	Risk
Manipulating HTTP requests, scoreboard tampering	Tampering	Built in server-side checks in CTFd	15
Flag leakage, user data exposure	Information Disclosure	TLS Certificate, using HTTPS	16
Brute-force flags, HTTP floods	Denial of Service	Rate-limiting flag inputs	16
Container escape → cluster access	Privilege Escalation	Avoid running containers in privileged mode, and use secure images	15

3.3. Cloud architectuur design

Inkomend verkeer wordt via een Application Load Balancer (ALB) gedistribueerd over horizontaal geschaalde EC2-instances. Deze bevatten de websites die zorgen voor regristratie, reroutering naar challenges en punten telling. De websites zijn de enige instances met connectie naar de database. De challenge containers naar waar gerouteerd wordt pullen hun container images van Amazon ECR.



3.4. DevOps

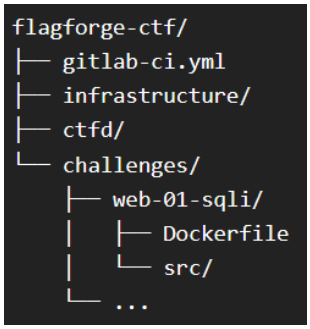
3.4.1. Branching Strategie (Versiebeheer)

Feature-branch workflow (github)

- main Branch:
  - De final branch, latest updates
  - Kan niet naar main pushen zonder review.
- Feature Branches:
  - Voor *elke* nieuwe taak (bv. een nieuwe challenge, een bugfix, een infra-aanpassing) maakt een teamlid een nieuwe branch aan vanuit main.
  - Naamgeving: We gebruiken een duidelijke prefix, bv:
- feature/add-sql-challenge
  - bugfix/fix-scoreboard-display
  - infra/setup-k8s-cluster
- Merge Requests (MRs):
  - Wanneer een feature klaar is, opent de dev een merge request om zijn/haar branch naar main te mergen.
  - Code Review: Minimaal één ander teamlid *moet* de MR reviewen en goedkeuren voor risicoanalyse.
  - CI-Pipeline: De CI/CD-pipeline (zie punt 4) moet succesvol draaien op MR *voordat* deze gemerged mag worden.

3.4.2. Project- en Repositorystructuur

Repositorystructuur voorbeeld



Toegangsregels (Access Rules)

- **Team FlagForge (6 leden): Maintainer Write** toegang.
- **Klant en docenten: Reporter Read only** toegang.
- **Branch Protection:** de main branch is beveiligd met **minimaal 1 approve** vereist voor een merge naar main.

3.4.3. CI/CD-Pipeline

3.4.3.1. CI-Pipeline (Draait op elke Pull Request naar main)

1. **Checkout Code:** Haalt de code van de feature-branch.
2. **Lint & Security Scan:**
    - Static Analysis: hadolint,dockle,… op Dockerfiles.
    - Vulnerability Scan: Trivy, Grype,… op Docker-images.
3. **Build Docker Images:** Bouwt de Docker-image ubuntu:22.04 voor de gewijzigde challenges/platform.
4. **Push to Staging Registry:** Pusht de images naar een 'staging' repository in de **Registry**.
5. **Test:** Deployt de nieuwe manifesten naar een tijdelijke 'test' namespace in Kubernetes.
6. **Notification:** Meldt op de PR of alle checks geslaagd zijn.

3.4.3.2. CD-Pipeline (Draait na elke Merge naar main)

1. Checkout Code: Haalt de nieuwe main branch op.
2. Build & Push Images:

- Bouwt de Docker-images (nginx:latest voor web, ubuntu:22.04 als base image, standaarden voor alle groepen).
- Pusht de images naar de 'production' repository in de **Registry**.

3. Deploy to Production:

- De pipeline (draaiend op een self-hosted runner) past de YAML toe op de cluster.
- **Commando:** kubectl apply -f /infrastructure/manifests/
- Kubernetes voert een 'rolling update' uit.

3.4.3.3. Reset strategie als challenge kapot is of onbereikbaar is.

We maken een manual job dat een challenge pod delete gebaseerd op de label. Dit is de reset job.
Dan zal k8s een nieuwe terug opstarten zodat de challenge weer werkt door de reset.

4. Kostenanalyse

4.1. Hosting Cloud

Voor het CTF-evenement is een kostenanalyse uitgevoerd voor twee mogelijke hostingconfiguraties binnen AWS, gebaseerd op de benodigde infrastructuurcomponenten en de verwachte belasting van de gebruikers.

De eerste configuratie richt zich op minimale kosten en omvat twee redundante web-nodes (t3.medium), een Application Load Balancer, een RDS MySQL-database, 15 challenge-instanties (t3.medium) en opslag van 10 Docker-images in ECR, met een eventduur van 12 uur. Bij een lage belasting, waarbij het websiteverkeer beperkt is en de challenge-nodes slechts basaal gebruikt worden, komen de kosten uit op ongeveer 10,23 USD. Voor gemiddeld verkeer stijgen de kosten naar circa 10,51 USD, en bij hoge belasting ligt de kostenraming rond 10,88 USD. Deze oplossing is financieel het meest gunstig, maar kan bij intensief gebruik of piekbelasting leiden tot een minder optimale gebruikerservaring of vertragingen.

De tweede configuratie is gericht op een optimale gebruikerservaring en maakt gebruik van extra resources, zoals meerdere challenge nodes in combinatie met autoscaling en een EKS-cluster, waardoor de website beter bestand is tegen piekbelasting. Voor een minimale belasting komt deze oplossing uit op ongeveer 45,21 USD voor 12 uur hosting, terwijl bij maximale belasting de kosten stijgen naar circa 66,32 USD. Dit hogere kostenplaatje biedt een stabielere performance en een betere responsiviteit voor de gebruikers, waardoor de ervaring tijdens het evenement minder risico loopt op vertragingen of overbelasting.

Beide bandbreedtes geven een realistisch beeld van de te verwachten kosten, inclusief EC2-instanties, RDS-database, Application Load Balancer, challenge-instanties, EKS-cluster (indien toegepast) en ECR-opslag, exclusief eventuele kleine netwerkverkeer-kosten. De opbouw- en testfase van de infrastructuur is hierin niet meegenomen; tijdens deze fase kunnen de daadwerkelijke kosten tijdelijk hoger uitvallen..

4.2. Hosting school infrastructuur

Een alternatief voor het hosten van het CTF-evenement in de cloud is het gebruik van de infrastructuur van de school. Hierbij lenen we de bestaande toestellen en servers, waardoor er geen aanschafkosten zijn voor nieuwe hardware. Dit brengt echter wel een risico met zich mee: bij intensief gebruik of onvoorzichtigheid kan er schade ontstaan aan de apparatuur. Voor het evenement BSides Limburg zal daarnaast enkel rekening gehouden moeten worden met de stroomkosten van het verbruik, aangezien de hardware al beschikbaar is en geen extra huur of aankoop vereist is.

4.3. Gedetailleerde kostenanalyse

4.3.1. AWS-hosting 1

- Voor het CTF-evenement is een budgetvriendelijke AWS-hostingconfiguratie opgesteld. Deze omvat:
- Web-infrastructuur: 2 redundante t3.medium EC2-instanties, load balanced via een Application Load Balancer
- Database: 1 RDS MySQL t3.medium-instance
- Challenge-infrastructuur: 15 t3.medium-instanties (6 text-based, 9 Docker-container)
  - Opslag: 10 Docker-images in ECR
  - Eventduur: 12 uur

Deze configuratie biedt basisfunctionaliteit voor 150 actieve gebruikers, met lage operationele kosten, maar bij piekbelasting kan performance dalen.

4.3.1.1. Kostenoverzicht

Component	**Aantal / type**	**Kosten 12 uur (USD)**
Web-nodes	2 × t3.medium	1,00
ALB (laag/middel/hoog)	1 × ALB + LCU	0,37 / 0,65 / 1,04
RDS MySQL	1 × db.t3.medium	1,32
Challenge-instanties	15 × t3.medium	7,52
ECR-opslag	10 images	0,02
Totaal	–	**10,23 – 10,88**

4.3.1.2. Analyse

- Voordelen: lage kosten, snelle opzet, eenvoudige infrastructuur
- Nadelen: beperkte schaalbaarheid, risico op vertraging bij piekbelasting, basisconfiguratie voor RDS en ALB

Conclusie: Deze configuratie is geschikt voor een kostenbewust evenement met voorspelbare belasting, maar biedt minder garantie voor optimale performance bij intensief gebruik.

4.3.2. AWS-hosting 2

Voor het CTF-evenement is een geavanceerde AWS-configuratie opgesteld die dynamische, per-gebruiker challenge-instanties met autoscaling ondersteunt. De infrastructuur omvat:

- Website: 2 redundante t3.medium EC2-instanties, load balanced via een ALB
- Database: 1 RDS MySQL t3.medium-instance
- Challenges: 6 text-based en 9 Docker/webstack-challenges op t3.small instances, met autoscaling (max. 200 gelijktijdige instances)
- ECR-opslag: 10 Docker-images
- Eventduur: 12 uur

Deze setup garandeert isolatie en performance voor alle actieve gebruikers, maar resulteert in hogere kosten dan een basisconfiguratie.

4.3.2.1. Kostenoverzicht

Component	Aantal / type	Kosten 12 uur (USD)
Web-nodes	2 × t3.medium	1,00
ALB (laag/middel/hoog)	1 × ALB + LCU	0,37 / 0,65 / 1,04
RDS MySQL	1 × db.t3.medium	1,32
Challenge-instanties	200 × t3.small	49,92
ECR-opslag	10 images	0,02
Totaal	–	**52,63 – 53,30**

4.3.2.2. Analyse

- Voordelen: optimale performance, isolatie per gebruiker, dynamische autoscaling
- Nadelen: hogere kosten, complexere infrastructuur, monitoring vereist

Conclusie: Deze configuratie is geschikt voor een professioneel CTF-evenement met hoge gelijktijdigheid en dynamische workloads, waarbij performance en isolatie prioriteit hebben.

4.4. Notitie

De hierboven weergegeven prijzen zijn berekend voor de hosting van één team. Voor het evenement worden de challenges van de drie creator-teams samengevoegd in één project, waardoor de totale infrastructuur en bijbehorende kosten proportioneel toenemen afhankelijk van het aantal gelijktijdige gebruikers en de gecombineerde load van alle challenges. Deze berekening dient dus als referentie per team, waarbij het uiteindelijke kostenplaatje voor het volledige project kan oplopen tot een meervoud hiervan, afhankelijk van de gekozen hostingconfiguratie en schaal van het evenement.

4.5. Week 1 Applicatie bouw backend – frontend

Week 1	Opdracht	Prijs
Dag 1	backend development	€350
Dag 2	backend development	€350
Dag 3	API testing + frontend	€350
Dag 4	frontend development	€350
Dag 5	frontend development + testing	€350

Totaal week 1: 5 × 350 = **€1.750**

4.6. Week 2 CTF implementatie

Week 2	Opdracht	Prijs
Dag 1	frontend testing	€350
Dag 2	frontend testing + ctf implementeren	€350
Dag 3	ctf implementeren	€350
Dag 4	ctf implementeren	€350
Dag 5	ctf implementeren	€350

Totaal week 2: 5 × 350 = **€1.750**

Totale kost: 10 × 350 = €3.500 per applicatie.
Eind totaal: 2 x 3.500 = €7.000

5. Geprioriteerde Product Backlog

Lijst van geprioriteerde product backlog met user stories en uit te voeren taken. Plan eerste stappen voor greenfield en bepaal wie verantwoordelijk is voor welke onderdelen om blokkades te vermijden (moet in Sprint 1 landen)

Sprint 1: Infrastructuur & werkend platform

De focus ligt op het opzetten van de technische ontwikkelomgeving, basisbeveiliging en de eerste opzet voor de applicatieontwikkelaars.

- De focus ligt op het opzetten van de technische ontwikkelomgeving, basisbeveiliging en de eerste opzet voor de applicatieontwikkelaars.
- het team wilt een k8s-cluster als ontwikkelingsomgeving op het datacenter.
  - proxmox instellen
  - k8s installeren
  - metalLB instellen
- het team wil de servers veilig bereikbaar maken
  - HTTPS instellen
- team wil een firewall instellen op de server
  - OPNSense instellen
- team wil resources, logs en performance zien
  - loki instellen
- het team wilt een eigen gitlab repo voor ontwikkeling van de ctf infra en app met duidelijke structuur
  - bepaal structuur van repo voor folders voor CTFd, challenges, k8s, ...
  - stel merge request security settings in naar 1 approve nodig voor merge request naar main
  - gebruik correcte branching-strategie, feature/naam-van-feature
- het team wilt automatisch fouten checken in de code met de CI
  - GitLab Runner installeren in K8s-cluster
  - .gitlab-ci.yml maken
  - lint scanner toevoegen
  - vulnerability scanner toevoegen

- de users moeten teams kunnen maken
    - Stel team creatie instellingen en limieten in
- de 2 APP studenten maken elk 1 web-app challenge
    - CRUD pagina's
    - Backend API infrastructuur
    - Project setup (frontend + backend)
    - Authenticatie (indien nodig)
    - Navigatie en layout
    - Form & error handling
    - Testing en quality checks
    - Deployment (dockerfile)
    - Documentatie
    - CTF flags implementatie

Sprint 2: Meer Content & Platform Functionaliteit

In sprint 2 wordt het platform verder uitgewerkt met een thema, gebruikersfunctionaliteiten en de bulk van de challenges.

- het team maakt de challenges voor de topics die we hebben verdeeld onder de andere teams
    - maak Obfuscation challenges
    - maak Trivia challenges
    - maak Network captures challenges
    - maak Stenography challenges
- de challenges moeten verschillende moeilijkheidsgraden hebben
    - test en rangschik de ctf challenges
- team wil special thema van de ctf
    - bespreek thema met klant
    - vergelijk de prototypes
- de users moeten hun score kunnen zien
    - stel CTFd scorebord in
- team wil ervoor zorgen dat de users zichzelf verifiëren met een confirmatie email
    - stel email verificatie in in CTFd registratie
- team moet kapotte challenges direct kunnen resetten
    - manual job toevoegen aan .gitlab-ci.yml met labels detection
- team moet threat model suggesties toepassen

Sprint 3: Integratie / Samenvoegen

De laatste sprint is gepland voor het samenvoegen van al het werk op de productieserver voorzien voor alle teams en het valideren van de kwaliteit en veiligheid.

- de teams willen alle challenges die gemaakt zijn samenvoegen op 1 productieserver
    - samen bespreken met teams
    - challenges importeren op productieserver met **ctfdcli**
- de teams volgen de standaarden die we hebben afgesproken van infra/flag format
    - web-apps nginx:latest
    - monitoring: prometheus, grafana
    - image: ubuntu 22.04
    - flag formaat flag{} en case sensitive
    - logs: loki
    - container repo: op github
    - niet-web-based-challenge: allemaal in 1 folder op aparte drive, gedownload door users
- challenges mogen alleen gecontroleerde vulnerabilities hebben hebben, geen ongewenste
    - test challenge containers op ongewenste vulnerabilities

GREENFIELD STRATEGIE VOOR SPRINT 1

Infra Verantwoordelijken (infra bouwen)

- starten direct op het datacenter: het opzetten van het k8s-cluster, het instellen van de firewall en het veilig bereikbaar maken van de servers.
- Dit is het belangrijkste waar iedereen zal vanaf hangen.

DevOps Verantwoordelijken (Structuur & CI/CD)

- beginnen parallel met het inrichten van de GitLab repository structuur en het schrijven van de CI/CD. Dit kan voorbereid worden zonder dat de eindomgeving voledig klaar is.

APP Studenten (Content Creatie)

- werken in Sprint 1 lokaal aan het maken van de eerste web-app challenges. Ze zijn nog niet afhankelijk van de cluster in sprint 1. Pas als de cluster klaar is wordt hun werk daarnaar verplaatst.

6. Bronnen

https://docs.ctfd.io/docs/deployment/installation/

Challenges:

A1 Themas

- Obfuscation - crypto
- Trivia - trivia
- Network captures - network
- Stenography – crypto

Challenges

- Audio stego (spectogram)
- Afbeelding met LSB-message
- Verstopte flag in HTTP-verkeer
- XOR-versleuteld bericht
- Pyhton app obfuscation
- SSTV wireshark
- Encrypted puzzelbox (multilayerd encryption box)
- Misleading algorithm
- Email confirmation flag
- Keypad bruteforce
- Outdated encryption HTTPS
- Fysiek: Dials om een code te vormen

App 1:

- SQL-Injection
- Cross-side scripting

App 2:

- Access Controll

- Qr-code in Qr-code

FEEDBACK CLIENTMEETING 21/11:

- Meer fysieke challenges implementeren minimaal 2 stuks
- Zorg voor zeker 3/4 vurnabilities per web-app
- Zorg dat je je niet te veel bezig houdt met de security in ctfd
- Zorg ervoor dat de flags niet door bruteforce kunnen worden aangevallen
- infrastructuur challenges: nmap op ip om bijvoorbeeld een ftp server te vinden met een hidden directory

Retrospective

Start doing	Stop doing	Keep doing
• DoR & DoD	• Te laat komen	• Project management in jira
• Implementeren feedback klant	• Clockify vergeten aanzetten of uizetten	• Vragen stellen aan elkaar
		• feedback vragen aan elkaar

DOR & DOD

- DoR: User stories is gedetaileerd met subtaken en heeft een userpoint.
- DoD: Code is getest met CI/CD, peer reviewed en besproken met het team.