Test plan - API Lab planner

Test plan: API Lab planner

Projectnaam: Lab planner,
Testplan-versie: 1.0.0,
Datum: 30-09-2024
Auteur: Jakub Chomik

Inleiding

Dit testplan beschrijft de strategie, aanpak, middelen en planning van de testactiviteiten voor de API. Het doel is om de kwaliteit van de applicatie te waarborgen door functionele tests uit te voeren.

Testdoelen

- Verifiëren of alle functionaliteiten correct werken.
- Oplossen van bugs.
- Zekerstellen dat gebruikers niet tot admin functionaliteiten toegang hebben.

Teststrategie

Zekerstellen dat de API voldoet aan de user stories.

Testomgeving

OS: Windows,
Tools: FastAPI testclient

Testdata

Gebruiker:
Username: Jakub,
Password: 123,

Admin:
Username: Jakub,
Password: 12345678

Test Stories

Story 1: Als beheerder wil ik dat de gebruikers van mijn applicatie taken kunnen zien en voltooien, zodat ik een helder overzicht heb van taken die voltooid zijn of nog voltooid moeten worden.

Test data:
taskId: 1
title: Test task
description: Lorem ipsul dolor
priority: 0
taskType: 0

Test cases:

1. **get_task():** Test of de taak met id 1 weergegeven wordt.
    - **Scenario**: De gebruiker krijgt een taak te zien.
    - **Verwacht resultaat**: Een taak wordt weergegeven.
2. **finish_task()**: Test of de finished veld van de taak op 1 kan worden gezet, en of de doneDate dan aangepast wordt.
    - **Scenario**: De gebruiker voltooit de taak.
    - **Verwacht resultaat**: De waarde van finished wordt gewijzigd naar 1 en de doneDate wordt aangepast naar de huidige datum/tijd.

Story 2: Als beheerder wil ik taken kunnen aanmaken, aanpassen en verwijderen, zodat ik volledige controle over de taken die er zijn heb.

Test data:
taskId: 1
title: Test task
description: Lorem ipsul dolor

priority: 0
taskType: 0

Test cases:

1. **create_task()**: Test of taken kunnen worden aangemaakt.
    - **Scenario**: De beheerder maakt een nieuwe taak aan.
    - **Verwacht resultaat**: Een nieuwe taak wordt correct aangemaakt met de opgegeven testdata.
2. **edit_task()**: Test of de taakdetails gewijzigd kunnen worden.
    - **Scenario**: De beheerder past de beschrijving van een taak aan.
    - **Verwacht resultaat**: De taakdetails worden succesvol aangepast in de database.
3. **delete_task()**: Test of een taak kan worden verwijderd.
    - **Scenario**: De beheerder verwijdert een taak.
    - **Verwacht resultaat**: De taak wordt verwijderd en is niet langer zichtbaar in het takenoverzicht.

Story 3: Als beheerder wil ik dat het takensysteem verbonden is met het voorraadsysteem, zodat de hoeveelheid voer die bij een taak verbruikt wordt, automatisch van de voorraad wordt afgehaald.

Test data:
taskId: 1
title: Test task
description: Lorem ipsul dolor
priority: 0
taskType: 1
stockId: 1
amount: 200

Test cases:

1. **finish_feeding_task**: Test of de correcte hoeveelheid verbruikt voer van de correcte voorraad gehaald wordt.
    - **Scenario**: Een voer-taak wordt voltooid.
    - **Verwacht resultaat**: Het opgegeven aantal voer wordt van de voorraad afgetrokken, en de voorraad wordt geüpdatet.

Story 4: Als beheerder wil ik dat er bij elke taak makkelijk te vinden informatie over mogelijke dier- en plantenziektes en bijzonderheden is, zodat de gebruikers die snel kunnen vinden indien nodig.

Test data:

1. **check_sickness**: Laat bij de ingevoerde symtomen passende ziektes zien en informatie erover
    - **Scenario:** Er is een zieke dier aangetroffen, en symptomen ingevuld.
    - **Verwacht resultaat**: Bij de symptomen passende ziektes worden weergegeven.

Rapport Systeem

Story 5: Als beheerder wil ik dat de gebruikers na het voltooien van een taak een rapport kunnen opstellen en inleveren, zodat ik een overzicht van de werkzaamheden en mogelijke problemen heb.

Test data:
taskId: 1
title: Taak voltooiing
description: Taak succesvol afgerond
photos: null
exceptionalities: null,

Test cases:

1. **submit_report()**: Test of een rapport kan worden ingediend na het voltooien van een taak.
    - **Scenario**: Een gebruiker voltooit een taak en levert een rapport in.
    - **Verwacht resultaat**: Het rapport wordt succesvol ingediend en is gekoppeld aan een taak.

Story 6: Als beheerder wil ik dat het mogelijk is om een foto aan een rapport toe te voegen, zodat ik kan kan zien wat er met de dier aan de hand is.

Test cases:

1. **add_photo_to_report()**: Test of een foto kan worden toegevoegd aan een rapport.

- **Scenario**: De gebruiker maakt een foto van een ziek dier of plant en voegt deze toe aan het rapport.
- **Verwacht resultaat**: De foto wordt correct toegevoegd aan het rapport en word goed opgeslagen.

Meldingen Systeem

Story 7: Als beheerder wil ik berichten naar gebruikers kunnen sturen, en wil ik dat ze er een melding van ontvangen op hun telefoon, zodat ik de gebruikers op de hoogte kan houden van mogelijke uitzonderingen.

Test data:
message: Er is een update beschikbaar

Test cases:

1. **send_notification()**: Test of de gebruiker een bericht ontvangt bij het versturen van een bericht.
    - **Scenario**: De beheerder stuurt een bericht naar een gebruiker.
    - **Verwacht resultaat**: De bericht word gekoppeld/verstuurd naar de gebruikers.

Story 8: Als beheerder wil ik een melding ontvangen als een taak niet voltooid is, zodat ik er meteen van op de hoogte ben.

Test cases:

1. **send_unfinished_task_alert()**: Test of als de deadline overschreden is een melding gestuurd kan worden.
    - **Scenario**: Een taak is niet voltooid binnen de gestelde tijd.
    - **Verwacht resultaat**: De beheerder ontvangt een bericht dat de taak niet voltooid is.

Story 9: Als beheerder wil ik een melding ontvangen als mijn voorraad leegloopt en er iets bijbesteld moet worden, zodat mijn voer nooit opraakt.

Test data:
quantity: 4,
foodTypeId: 1,
minimumQuantity: 5

Test cases:

1. **send_low_stock_alert()**: Test of een melding verstuurd kan worden als de voorraad niveau onder het minimum zit.
    - **Scenario**: De voorraad is onder 5 kg.
    - **Verwacht resultaat**: De beheerder ontvangt een melding dat de voorraad bijna op is.

Voorraad Systeem

Story 10: Als beheerder wil ik een helder overzicht van al mijn voorraden, zodat ik weet wanneer iets bijbesteld moet worden.

Test data
quantity: 4,
foodTypeId: 1,
minimumQuantity: 5

Test cases:

1. **get_stock()**: Test of het voorraadoverzicht correct wordt weergegeven.
    - **Scenario**: De beheerder opent het voorraadoverzicht.
    - **Verwacht resultaat**: Het overzicht toont de actuele voorraadniveaus voor alle artikelen.

Story 11: Als beheerder wil ik nieuwe voorraden kunnen aanmaken, aanpassen en verwijderen, zodat ik volledige controle over mijn voorraden heb.

Test data
quantity: 4,
foodTypeId: 1,
minimumQuantity: 5

Test cases:

1. **create_stock()**: Test of nieuwe voorraden kunnen worden aangemaakt.
    - **Scenario**: De beheerder maakt een nieuwe voorraad aan.
    - **Verwacht resultaat**: Het nieuwe voorraad wordt succesvol toegevoegd aan het voorraadsysteem.

2. **edit_stock()**: Test of voorraadinformatie kan worden aangepast.
    - **Scenario**: De beheerder wijzigt de hoeveelheid van een voorraad.
    - **Verwacht resultaat**: De wijzigingen worden correct doorgevoerd.
3. **delete_stock()**: Test of een voorraad kan worden verwijderd.
    - **Scenario**: De beheerder verwijdert een voorraad uit het systeem.
    - **Verwacht resultaat**: Het voorraad wordt succesvol verwijderd.

Story 12: Als beheerder wil ik dat de voorraad berekeningen goed verlopen, zodat er zo weinig mogelijk fouten voorkomen.

Test data:
quantity: 4,
foodTypeId: 1,
minimumQuantity: 5

Test cases:

1. **stock_calculations()**: Test of de berekeningen van de voorraad correct verlopen.
    - **Scenario**: Er zijn 3 paarden die dagelijks 500g hooi gevoerd moeten worden.
    - **Verwacht resultaat**: De minimum voorraad wordt correct berekend voor de benodigde hoeveelheid dagen.

Dieren Systeem

Story 13: Als beheerder wil ik een helder overzicht van al mijn dieren, zodat ik weet wanneer er iets verandert.

Test cases:

1. **get_animals()**: Test of het overzicht van dieren correct wordt weergegeven.
    - **Scenario**: De beheerder opent het dierenoverzicht.
    - **Verwacht resultaat**: Alle dieren worden correct weergegeven.

Story 14: Als beheerder wil ik een nieuwe dier kunnen toevoegen, informatie kunnen aanpassen en verwijderen zodat ik de dieren catalogus kan aanpassen als er dieren verdwijnen of bijkomen.

Test data:
name: Testdier

animalTypeId: 1
birthDate: 20-09-2024

Test cases:

1. **add_animal()**: Test of een nieuw dier kan worden toegevoegd.
    - **Scenario**: De beheerder voegt een nieuw dier toe aan het systeem.
    - **Verwacht resultaat**: Het dier wordt succesvol toegevoegd aan de database.
2. **edit_animal()**: Test of dierinformatie kan worden aangepast.
    - **Scenario**: De beheerder wijzigt de gegevens van een dier.
    - **Verwacht resultaat**: De wijzigingen worden correct doorgevoerd.
3. **delete_animal()**: Test of een dier kan worden verwijderd.
    - **Scenario**: De beheerder verwijdert een dier uit het systeem.
    - **Verwacht resultaat**: Het dier wordt succesvol verwijderd.

User Experience

Story 15: Als beheerder wil ik dat de applicatie simpel en makkelijk in gebruik is, zodat ouderen en kinderen deze kunnen gebruiken.

Testen in de app.

Story 16: Als beheerder wil ik dat de applicatie in meerdere talen beschikbaar is, zodat gebruikers die geen Engels kunnen lezen het nog steeds kunnen gebruiken.

https://cloud.google.com/translate?hl=nl#translate-a-website-or-app