Centraal Testdocument - IoT Project

# Inleiding

Dit document bevat een overzicht van de teststrategieën en testscenario's voor het UltiHorloge-project. Het doel is om ervoor te zorgen dat alle user stories correct geïmplementeerd zijn en functioneel werken volgens de verwachtingen. Voor elke user story wordt het testtype bepaald, worden de testscenario's beschreven en worden de uitgevoerde testen en bijbehorende problemen gedocumenteerd. Dit document wordt aangevuld met een Excel-bestand waarin gedetailleerde testresultaten worden bijgehouden.

Contents

[Inleiding 1](#_Toc198406502)

[User Stories 3](#_Toc198406503)

[User Story 1: 3](#_Toc198406504)

[Testbeschrijving en methode: Unit Test 3](#_Toc198406505)

[User Story 2: 4](#_Toc198406506)

[Testbeschrijving en methode: Unit Test 4](#_Toc198406507)

[User Story 3: 5](#_Toc198406508)

[Testbeschrijving en methode: Real World Test 5](#_Toc198406509)

[User Story 4: 6](#_Toc198406510)

[Testbeschrijving en methode: Integratie Test 6](#_Toc198406511)

[User Story 5: 7](#_Toc198406512)

[Testbeschrijving en methode: Real World Test 7](#_Toc198406513)

[User Story 6: 8](#_Toc198406514)

[Testbeschrijving en methode: Integratietest 8](#_Toc198406515)

[User Story 7: 9](#_Toc198406516)

[Testbeschrijving en methode: Integratietest 9](#_Toc198406517)

[User Story 8: 10](#_Toc198406518)

[Testbeschrijving en methode: Real World Test 10](#_Toc198406519)

[User Story 9: 11](#_Toc198406520)

[Testbeschrijving en methode: Load Test 11](#_Toc198406521)

[User Story 10: 12](#_Toc198406522)

[Testbeschrijving en methode: Integratietest 12](#_Toc198406523)

[User Story 11: 13](#_Toc198406524)

[Testbeschrijving en methode: Unit Test 13](#_Toc198406525)

[User Story 12: 14](#_Toc198406526)

[Testbeschrijving en methode: Real World Test 14](#_Toc198406527)

# User Stories

## User Story 1:

Als gebruiker wil ik dat mijn horloge automatisch en manueel het temperatuur en vochtigheid meet zodat ik actie kan ondernemen.

### Testbeschrijving en methode: Unit Test

De temperatuur- en vochtigheidssensor moet zowel automatisch als handmatig met een knop kunnen worden geactiveerd. De metingen moeten correct en stabiel zijn.

Waarom een unit test?

Een unit test is gekozen omdat we willen weten of de sensorcomponent correct werkt. Dit type test stelt ons in staat om het uitleesgedrag van de sensor onder verschillende omstandigheden te analyseren zonder invloeden van andere systeemcomponenten zoals de display of netwerkverbinding.

Testscenario’s

De testscenario’s voor deze user story zijn vastgelegd in een Excel-document.

De sensor wordt getest door bij opstart in een normale omgeving te controleren of de waarde overeenkomt met de omgevingstemperatuur. Vervolgens wordt de werking van de sensor getest in warme en koude omgevingen om stabiliteit en betrouwbaarheid te beoordelen. Ook wordt de werking via de manuele drukknop gecontroleerd, waarbij wordt gelet op de directe respons van de sensor.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 1.

Problemen & Oplossingen

Tijdens de tests kwam naar voren dat de sensor niet correct aangesloten was. Dit probleem is opgelost door de aansluiting opnieuw te controleren en vast te zetten. Daarnaast werden onrealistische temperatuurwaarden gemeten, bijvoorbeeld 45°C in een kamer van 24°C. Dit is opgelost door de sensor opnieuw te kalibreren en de software aan te passen.

## User Story 2:

Als gebruiker wil ik mijn hartslag automatisch en manueel kunnen meten om mijn gezondheid in controle te houden.

### Testbeschrijving en methode: Unit Test

Waarom een Unit Test?

Voor deze user story wordt een unit test uitgevoerd. Dit testtype is gekozen omdat het essentieel is om de hartslagsensor individueel en nauwkeurig te testen, zonder interferentie van andere systemen.

Testscenario’s

Voor deze user story werden verschillende scenario’s uitgewerkt. In het eerste scenario wordt de hartslag gemeten wanneer de gebruiker volledig in rust is. Hierbij wordt verwacht dat de hartslag stabiel blijft en realistische waarden toont. Vervolgens werd het scenario getest waarbij de gebruiker zich fysiek inspant door te wandelen en te rennen. Hierbij werd verwacht dat de sensor een duidelijke stijging van de hartslag zou detecteren en dit correct zou weergeven. Tot slot werd het scenario getest waarbij de gebruiker handmatig via een knop op het horloge een meting start om de directe reactie en nauwkeurigheid te controleren.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 1.

Problemen & Oplossingen

Tijdens het testproces werden enkele problemen geïdentificeerd en geanalyseerd. Het eerste probleem betrof de sensor, die in eerste instantie onnauwkeurige waarden gaf. Er werden onverklaarbare schommelingen en heel hoge meetwaarden waargenomen, zelfs bij geen fysieke activiteiten. Na het verder verwerken van het probleem kwam dat deze afwijkingen veroorzaakt werden door een fout in de sensorcode, waardoor storingen en fouten gaf. Dit probleem werd verholpen door de code in Visual Studio Code te optimaliseren en de filtering van sensorwaarden te verfijnen, wat resulteerde in nauwkeurigere metingen.

Daarnaast trad er een probleem op met de drukknop, die geen metingen gaf bij activering. Bij het testen, toonde dat de oorzaak een hardwarematig defect aan de knop was. Dit werd opgelost door de defecte knop te vervangen en de debounce-tijd in de software te verkorten, wat de responsiviteit verbeterde. Na deze aanpassingen werden alle tests succesvol afgerond en leverde de sensor betrouwbare meetwaarden op.

## User Story 3:

Als gebruiker wil ik dat mijn GPS-locatie wordt opgeslagen, zodat ik mijn activiteiten kan analyseren.

### Testbeschrijving en methode: Real World Test

De GPS-module moet automatisch en manueel via een drukknop de locatie bepalen.

Waarom een real-world test?

Een real-world test is gekozen omdat de GPS-functie afhankelijk is van de omgeving en omgevingsfactoren zoals signaalsterkte en beweging. Het testen in verschillende omgevingen (binnen, buiten, beweging) helpt om realistische gebruikersscenario’s na te krijgen en problemen op te sporen.

Testscenario’s

Bij het testen van het GPS-sensor hebben we enkele scenario’s vastgesteld. Het eerste scenario was het testen van de GPS-module binnenshuis. De verwachting was dat er binnen geen stabiel GPS-signaal zou worden ontvangen vanwege het gebrek aan satellietontvangst binnenshuis en is dit ook gebeurd.

Het tweede scenario was dat we de GPS-module buiten op een vaste locatie zonder obstakels of enige blokkeringen testen. De verwachting was dat de sensor snel en nauwkeurig GPS-coördinaten zou leveren.

In het derde scenario werd onderzocht hoe het GPS-signaal zich gedraagt wanneer de gebruiker zich verplaatst, bijvoorbeeld tijdens het wandelen of fietsen. Hierbij werd geanalyseerd of de module consistente en stabiele updates genereerde van de veranderende locatie.

Het laatste scenario was hoe de GPS-module zou reageren met een aansluiting van een drukknop. Het doel hier was om een manuele meting te laten starten en de GPS-coördinaten te krijgen.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 3.

Problemen & Oplossingen

Tijdens de tests werden verschillende problemen vastgesteld. Allereerst gaf de GPS-module binnenshuis geen signaal. Dit bleek normaal gedrag te zijn, aangezien GPS-signalen doorgaans slecht of helemaal niet doordringen in afgesloten ruimtes. Om verwarring te voorkomen, werd dit expliciet vermeld in de gebruikershandleiding.

Daarnaast bleef de GPS-waarde buitenshuis op (0.0, 0.0) staan, wat erop wees dat er geen correcte locatiebepaling plaatsvond. Dit probleem werd verholpen door de code te verbeteren en de wachttijd voor het verkrijgen van een GPS-fix te verlengen, waardoor de module voldoende tijd kreeg om een nauwkeurige positie te bepalen.

Tot slot gaf de drukknop geen respons bij gebruik. Na controle bleek de knopaansluiting mogelijk defect te zijn. Dit werd opgelost door een nieuwe knop te installeren, waarna de functionaliteit hersteld was.

## User Story 4:

Als gebruiker wil ik dat mijn horloge automatisch de sensor waarden op display toont, zodat ik beeld hebt met mijn activiteit.

### Testbeschrijving en methode: Integratie Test

Deze user story vereist dat alle individuele sensoren (temperatuur, vochtigheid, hartslag, GPS en tijd met datum) correct samenwerken met het display. Het doel is dat de gemeten waarden van alle sensoren automatisch en ook handmatig via een drukknop correct worden weergegeven op het scherm van het horloge.

Waarom een real-world test?

Een integratietest is gekozen omdat het essentieel is om te controleren of de verschillende sensoren goed samenwerken met het display. Hierbij worden potentiële problemen ontdekt die ontstaan door fouten in communicatie tussen componenten, zoals ontbrekende waarden of foutieve weergaves.

Testscenario’s

In het eerste scenario werd bij het opstarten van het horloge gecontroleerd of het display automatisch en zonder handmatige interactie direct alle correcte sensorwaarden toont. In het volgend scenario werd getest of het display helder en duidelijk leesbaar is. Omdat het een klein scherm is, moet je alles goed mooi toepassen en eventueel pagina-indeling op het scherm te gebruiken. Ook werd gecontroleerd hoe het display reageerde bij activatie via een drukknop.

Daarnaast werd getest of het display continu actuele waarden toonde en geen oude waarden bleef weergeven na het indrukken van de knop. Het display moet uiteindelijk wel blijven verversen en de nieuwe waardes eventueel tonen.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 4.

Problemen & Oplossingen

Tijdens het uitvoeren van de testen kwamen meerdere problemen aan het licht. Allereerst bleek het display aanvankelijk helemaal geen waarden weer te geven, wat veroorzaakt werd door een verkeerde aansluiting van de sensoren. Dit probleem werd opgelost door de hardwareaansluitingen te controleren en correct te installeren.

Vervolgens was het display niet altijd duidelijk leesbaar, omdat verkeerde configuraties in de layout-instellingen in Visual Studio Code waren ingesteld. Dit probleem werd verholpen door aanpassing van de layoutinstellingen, wat resulteerde in een duidelijke en goed leesbare weergave van sensorwaarden.

Daarnaast werkte de drukknop aanvankelijk niet goed, doordat er sprake was van een hardwaredefect en verkeerde softwarematige debouncing-instellingen. Dit werd opgelost door de defecte knop te vervangen en de debounce-instellingen te optimaliseren in de software.

Tot slot bleef het display soms oude sensorwaarden weergeven bij het handmatig indrukken van de knop. Dit probleem werd veroorzaakt door een fout in de verversingsfunctie van de software. De fout werd gecorrigeerd, waarna het display correct werd bijgewerkt en nu stabiel en betrouwbaar functioneert.

## User Story 5:

Als sporter wil ik dat mijn horloge automatisch koelt via peltier module wanneer het warm is, zodat ik comfortabel blijf tijdens het sporten.

### Testbeschrijving en methode: Real World Test

Waarom een real-world test?

Een real-world test is geselecteerd omdat de werking van de Peltier-koelmodule sterk afhankelijk is van omgevingsfactoren zoals de omgevingstemperatuur en de lichaamstemperatuur van de gebruiker tijdens het sporten. Alleen door testing in echte gebruiksomstandigheden kunnen we de betrouwbaarheid en effectiviteit van het koelsysteem beoordelen.

Testscenario's

In het eerste scenario werd het automatisch activeren van de Peltier-module getest. Hierbij werd de temperatuursensor kunstmatig verhit tot boven de ingestelde drempelwaarde van 30°C, waarbij werd verwacht dat de koelmodule automatisch zou inschakelen. De werking werd gecontroleerd door de temperatuur van het horloge met een externe thermometer te meten.

Vervolgens werd het horloge getest tijdens een hardloopsessie, waarbij de lichaamstemperatuur van de gebruiker natuurlijk steeg. Hierbij werd gecontroleerd of de koelmodule correct reageerde op de verhoogde temperatuur en of het koeleffect voldoende waarneembaar was voor de gebruiker.

Tot slot werd de batterijduur getest met de koelmodule in werking, aangezien de module energieverbruik verhoogt. Dit was belangrijk om te bepalen of het horloge nog steeds een acceptabele gebruiksduur had tijdens intensieve sportactiviteiten.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 5.

Problemen & Oplossingen

Tijdens de testen werden verschillende problemen geïdentificeerd. Eerst en vooral werd opgemerkt dat de koelmodule niet altijd tijdig activeerde. Dit werd veroorzaakt door een te hoge drempelwaarde in de software. Na aanpassing van de drempelwaarde van 35°C naar 30°C activeerde de module eerder, wat het comfort verbeterde.

Een ander probleem was dat de koeling niet uniform was op het contactoppervlak met de huid. Dit werd opgelost door het ontwerp van de behuizing aan te passen, waardoor het contactoppervlak van de Peltier-module beter verdeeld werd.

Daarnaast bleek de batterijduur drastisch te verminderen wanneer de koelmodule langdurig actief was. Dit probleem werd aangepakt door de firmware aan te passen met een slimmer energiebeheersysteem dat de koeling in pulsen activeert in plaats van continu. Hierdoor verbeterde de batterijduur aanzienlijk zonder het koeleffect significant te verminderen.

## User Story 6:

Als ontwikkelaar wil ik dat de hoofdlogica plaatsvindt op de Note Red platform dat geinstalleerd is op Raspberry Pi.

### Testbeschrijving en methode: Integratietest

Deze user story vereist een integratietest omdat het Node-RED platform als centrale logische verwerkingslaag fungeert. De ESP32 stuurt sensorgegevens via MQTT naar de Raspberry Pi, waar Node-RED beslissingen neemt, acties uitvoert (zoals notificaties sturen of gegevens naar Notion pushen) en dashboards bijwerkt. De samenwerking tussen hardware, MQTT-communicatie, Node-RED flows, en externe outputs moet dus als geheel getest worden.

Waarom een integratietest?  
De reden voor het kiezen van een integratietest is dat de volledige werking van het systeem afhankelijk is van de correcte communicatie en integratie tussen verschillende onderdelen: ESP32 hardware, MQTT broker, Node-RED logica, en externe diensten (zoals Notion of Telegram). Fouten kunnen op verschillende niveaus ontstaan, en daarom moet het hele ecosysteem samen getest worden.

Testscenario’s  
In de eerste scenario’s werd getest of de MQTT-verbinding tussen de ESP32 en Node-RED stabiel was. Hierbij werd bij elke verzonden sensorwaarde gecontroleerd of deze correct binnenkwam op Node-RED. Vervolgens werd een logica-flow in Node-RED gebouwd die de temperatuur vergelijkt met een grenswaarde van 25°C. Als deze wordt overschreden, moet automatisch een waarschuwing gestuurd worden via Telegram. Aanvankelijk werkte dit niet omdat de flow verkeerd verbonden was met het verkeerde topic. Daarnaast werd getest of de logica in Node-RED correct meerdere sensorwaarden kan verwerken, combineren, en doorsturen naar Notion. Soms werden verkeerde payloads gegenereerd door foutieve JSON-structuren.  
Tot slot werd ook gekeken of de logica in Node-RED juist reageert op manuele input (drukknoppen) en of de meting dan ook in de database terechtkomt.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad *User Story 6*.

Problemen & Oplossingen  
Tijdens de testfase werden verschillende problemen vastgesteld. Een eerste fout was dat de MQTT-topicnamen niet correct overeenkwamen tussen de ESP32-code en de Node-RED-onderwerpen, waardoor berichten niet doorkwamen. Dit werd opgelost door de juiste topics te hernoemen en in beide systemen te synchroniseren.  
Ook werden verkeerde flows gebruikt in Node-RED waardoor de temperatuurcheck naar het verkeerde blok liep. Door de flow te herstructureren met duidelijkere commentaren en knooppunten, werd dit gecorrigeerd.  
Verder bleek dat bepaalde JSON-berichten van de ESP32 niet goed geformatteerd waren, wat leidde tot fouten in de Node-RED parser. Door betere foutafhandeling en JSON-validatie toe te voegen in Node-RED, werd dit opgelost.  
Ten slotte was er een probleem met de verbinding naar Notion via een webhook. De API-key werd niet herkend omdat deze niet correct was ingesteld in de header. Na aanpassing werkte de communicatie correct en werden de data elke 15 minuten en op manuele actie succesvol doorgestuurd.

## User Story 7:

Als gebruiker wil ik waarschuwingen ontvangen via Telegram als de temperatuur te hoog wordt, zodat ik actie kan ondernemen.

### Testbeschrijving en methode: Integratietest

Deze user story vereist dat de temperatuursensor correct werkt, dat Node-RED de juiste beslissingen neemt gebaseerd op de sensorwaarden, en dat berichten succesvol worden verzonden via de Telegram API.

Waarom een integratietest?

Een integratietest is gekozen omdat deze functionaliteit meerdere componenten omvat die samen moeten werken: de temperatuursensor, de Node-RED logica, de internetverbinding en de Telegram API. Door het gehele systeem te testen, kunnen we nagaan of de communicatie tussen alle componenten correct verloopt.

Testscenario's

In het eerste scenario werd de temperatuursensor handmatig verhit tot boven de drempelwaarde (ingesteld op 30°C). Hierbij werd verwacht dat het systeem automatisch een waarschuwing zou sturen naar de geconfigureerde Telegram-chat.

Vervolgens werd getest of meerdere opeenvolgende waarschuwingen correct verzonden werden, zonder dat het systeem spamberichten zou sturen. Hierbij werd verwacht dat waarschuwingen op een redelijke frequentie (maximaal één per 5 minuten) zouden worden verzonden.

In het derde scenario werd getest of de waarschuwingsfunctie nog steeds werkte wanneer het horloge zich in een gebied met slechte internetverbinding bevond. Hierbij werd verwacht dat het systeem de berichten zou blijven proberen te verzenden tot er een verbinding beschikbaar was.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 7.

Problemen & Oplossingen

Tijdens de tests werden verschillende problemen ontdekt. Allereerst werd vastgesteld dat de Telegram API-sleutel niet correct was geconfigureerd in Node-RED, waardoor geen berichten werden verzonden. Dit probleem werd opgelost door de API-sleutel opnieuw te genereren en correct in te stellen. Een ander probleem was dat er te veel waarschuwingen werden verzonden wanneer de temperatuur rond de drempelwaarde schommelde. Dit veroorzaakte een stortvloed aan berichten. Dit probleem werd opgelost door een hysterese van 2°C toe te voegen aan de logica, waardoor de temperatuur eerst 2°C onder de drempelwaarde moest dalen voordat een nieuwe waarschuwing kon worden verzonden. Ook werd opgemerkt dat bij slechte internetverbinding het systeem vastliep bij het verzenden van berichten. Dit werd opgelost door een time-out toe te voegen en een wachtrij te implementeren voor berichten die niet direct konden worden verzonden.

Tot slot bleek dat de waarschuwingstekst niet altijd informatief genoeg was. Dit werd verbeterd door de exacte temperatuurwaarde en het tijdstip toe te voegen aan het bericht, wat de gebruiker meer context gaf.

## User Story 8:

Als gebruiker wil ik dat mijn horloge via spraakopdrachten kan worden bediend, zodat ik het handsfree kan gebruiken.

### Testbeschrijving en methode: Real World Test

Waarom een real-world test?

Een real-world test is gekozen omdat spraakherkenning sterk varieert afhankelijk van factoren zoals stemkenmerken, accent, omgevingsgeluid en formulering van opdrachten. Alleen door te testen in echte gebruikssituaties kunnen we vaststellen of de spraakbediening betrouwbaar werkt onder diverse omstandigheden.

Testscenario's

In het eerste scenario werden basale spraakopdrachten getest, zoals "toon temperatuur" en "meet hartslag". Hierbij werd verwacht dat het horloge de opdracht zou herkennen en de gevraagde actie zou uitvoeren.

Vervolgens werd de spraakherkenning getest in verschillende omgevingen met achtergrondgeluid, zoals een drukke sportschool en een rustige kamer. Dit om te bepalen hoe robuust de spraakherkenning was onder verschillende omstandigheden.

In het derde scenario werden complexere opdrachten getest, zoals "stuur mijn locatie naar huis" of "geef me een waarschuwing als mijn hartslag boven 150 komt". Hierbij werd geëvalueerd of het systeem in staat was om samengestelde opdrachten correct te interpreteren en uit te voeren.

Tot slot werd de spraakherkenning getest met verschillende gebruikers (mannen, vrouwen, verschillende accenten) om te bepalen of het systeem goed werkte voor een diverse gebruikersgroep.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 8.

Problemen & Oplossingen

Tijdens de tests werden verschillende problemen geconstateerd. Allereerst bleek de spraakherkenning slecht te presteren in lawaaierige omgevingen. Dit probleem werd aangepakt door ruisonderdrukking te implementeren en de microfoon dichter bij de mond te plaatsen.

Een ander probleem was dat het systeem moeite had met bepaalde accenten. Dit werd verbeterd door het spraakherkenningsalgoritme te trainen met een breder scala aan stemmen en accenten.

Daarnaast werden sommige complexere opdrachten niet correct herkend. Dit werd opgelost door de opdrachtenbibliotheek uit te breiden en synonymen toe te voegen voor verschillende commando's, waardoor gebruikers flexibeler konden zijn in hun formulering.

Tot slot was er een aanzienlijke vertraging tussen het geven van een spraakopdracht en de uitvoering ervan. Dit werd verbeterd door de verwerking te optimaliseren en bepaalde veelgebruikte commando's lokaal op het apparaat te verwerken in plaats van via de cloud.

## User Story 9:

Als ontwikkelaar wil ik alle sensordata naar een cloud platform sturen, zodat ik deze kan analyseren en opslaan.

### Testbeschrijving en methode: Load Test

Deze user story vereist dat alle sensordata correct wordt verzameld, geformatteerd en verzonden naar een cloudplatform (Notion), waar het beschikbaar is voor analyse en langetermijnopslag.

Waarom een load test?

Een load test is gekozen omdat we willen controleren of het systeem grote hoeveelheden sensordata kan verwerken en of het cloudplatform efficiënt blijft functioneren onder verschillende belastingen. Door hoge volumes van data te simuleren, kunnen we knelpunten identificeren in de dataverwerking en -opslag.

Testscenario's

In het eerste scenario werd de automatische verzending van sensordata naar Notion getest. Hierbij werd gecontroleerd of temperatuur, vochtigheid, hartslag en GPS-locatiegegevens correct en op regelmatige intervallen (elke 15 minuten) werden opgeslagen.

Vervolgens werd de handmatige verzending van data via een drukknop getest, waarbij werd verwacht dat alle actuele sensorwaarden onmiddellijk naar het cloudplatform werden gestuurd. In het derde scenario werd de dataretentie getest om te bepalen of historische gegevens correct werden bewaard en toegankelijk bleven voor latere analyse. Hierbij werd ook gecontroleerd of de data correct werd gevisualiseerd in grafieken en tabellen binnen het Notion-platform. Tot slot werd de robuustheid van het systeem getest door de internetverbinding tijdelijk te verbreken. Hierbij werd verwacht dat het systeem de data lokaal zou bufferen en zou verzenden zodra de verbinding was hersteld.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 9.

Problemen & Oplossingen

Tijdens het testen werden verschillende problemen vastgesteld. Allereerst werd ontdekt dat de Notion API-integratie regelmatig time-outs vertoonde bij het verwerken van grote hoeveelheden gegevens. Dit probleem werd opgelost door de datapakketten kleiner te maken en de verzending te spreiden over meerdere API-aanroepen.Een ander probleem was dat sommige sensorwaarden niet correct werden geformatteerd voor de database, wat resulteerde in onjuiste dataopslag. Dit werd opgelost door een standaard JSON-formaat te implementeren voor alle sensordata en een validatiestap toe te voegen voordat de data werd verzonden.Daarnaast bleek dat bij het verlies van internetverbinding de gebufferde data niet altijd correct werd verzonden wanneer de verbinding weer beschikbaar was. Dit werd verholpen door een robuustere bufferstrategie te implementeren met bevestiging van ontvangst. Tot slot werd opgemerkt dat de data-analyse in Notion beperkt was door het ontbreken van metadata zoals tijdstempels. Dit werd verbeterd door een consistente tijdstempelstructuur toe te voegen aan alle verzonden gegevens, wat de analyse en visualisatie aanzienlijk verbeterde.

## User Story 10:

Als gebruiker wil ik weersinformatie kunnen opvragen op een website, zodat ik mij kan voorbereiden op mijn activiteiten.

### Testbeschrijving en methode: Integratietest

De webinterface moet gebruiksvriendelijk zijn, correct werken op verschillende apparaten en browsers, en accurate weersinformatie tonen gebaseerd op de locatiegegevens van de gebruiker.

Waarom een integratietest?

Een integratietest is gekozen omdat meerdere systemen moeten samenwerken: de webinterface, de database met sensorgegevens, de externe weer-API en de gebruikersauthenticatie. Door het geheel te testen, kunnen we bepalen of alle componenten correct met elkaar communiceren.

Testscenario's

In het eerste scenario werd de toegankelijkheid van de website getest op verschillende apparaten (desktop, tablet, smartphone) en browsers (Chrome, Firefox, Safari). Hierbij werd verwacht dat de website responsive zou zijn en correct zou worden weergegeven op alle platforms.

Vervolgens werd de weergave van weersinformatie getest door verschillende locaties in te voeren. Hierbij werd gecontroleerd of de juiste en actuele weersinformatie werd getoond, inclusief temperatuur, vochtigheid, windsnelheid en neerslagkans. In het derde scenario werd de integratie met de horloge-app getest. Hierbij werd verwacht dat de gebruiker de webinterface kon gebruiken om de laatst opgeslagen locatie van het horloge te zien en de bijbehorende weersinformatie te ontvangen.

Tot slot werd de gebruiksvriendelijkheid getest door verschillende testgebruikers te vragen om specifieke taken uit te voeren, zoals het opvragen van het weerbericht voor morgen op een specifieke locatie.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 10.

Problemen & Oplossingen

Tijdens de tests werden verschillende problemen vastgesteld. Allereerst werd ontdekt dat de website niet correct werd weergegeven op mobiele apparaten, met elementen die buiten het scherm vielen. Dit probleem werd opgelost door een responsive design te implementeren met CSS Media Queries. Een ander probleem was dat de weersinformatie niet altijd up-to-date was. Dit werd veroorzaakt door een caching-probleem in de API-aanroepen. Dit werd verholpen door de cache-instellingen aan te passen en een refresh-knop toe te voegen waarmee gebruikers handmatig de meest recente gegevens konden ophalen. Daarnaast werkte de automatische locatiedetectie niet betrouwbaar, vooral op desktopcomputers. Dit werd verbeterd door aanvullende geocoding-methoden te implementeren en een duidelijke foutmelding te tonen wanneer de locatie niet kon worden bepaald, met de optie om handmatig een locatie in te voeren. Tot slot bleek uit gebruikerstests dat de navigatie niet intuïtief was. Dit werd opgelost door het menu te herstructureren en visuele cues toe te voegen om gebruikers te helpen bij het vinden van de gewenste informatie.

## User Story 11:

Als ontwikkelaar heb ik het graag dat de componenten mooi en goed gesoldeerd zijn.

### Testbeschrijving en methode: Unit Test

De hardware moet degelijk en betrouwbaar zijn, met zorgvuldige solderingen, correcte plaatsing van componenten en een degelijke constructie.

Waarom een unit test?

Een unit test is gekozen omdat we de kwaliteit en betrouwbaarheid van de individuele hardwarecomponenten willen testen voordat ze worden geïntegreerd in het volledige systeem. Door elke soldeerverbinding en component afzonderlijk te testen, kunnen we problemen vroeg identificeren en oplossen.

Testscenario's

In het eerste scenario werd elke soldeerverbinding visueel geïnspecteerd op glans, vorm en hechting. Hierbij werd verwacht dat elke verbinding glad en glanzend zou zijn, zonder koude soldeerverbindingen of overmatige soldeer (soldeerbruggen).

Vervolgens werd de mechanische stevigheid van de verbindingen getest door voorzichtig aan de componenten te trekken en te controleren of ze stevig vast zaten. Ook werd getest of er geen kortsluiting was tussen naburige verbindingen, wat zou kunnen leiden tot defecten of onveilige situaties.

In het derde scenario werd de isolatie gecontroleerd tussen spanningvoerende delen en de behuizing, om er zeker van te zijn dat er geen elektrische risico's waren voor de gebruiker.

Tot slot werd een continuïteitstest uitgevoerd om te controleren of alle elektrische paden correct waren aangelegd en er geen onderbrekingen waren in de circuits.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 11.

Problemen & Oplossingen

Tijdens de inspectie werden verschillende problemen vastgesteld. Allereerst werden enkele koude soldeerverbindingen ontdekt, vooral bij de kleinere SMD-componenten. Deze werden opnieuw gesoldeerd met de juiste techniek en temperatuur, wat resulteerde in betere verbindingen.

Een ander probleem was dat sommige draden te dicht bij elkaar waren geplaatst, wat het risico op kortsluiting vergrootte. Dit werd opgelost door kabelbinders te gebruiken om de bedrading netter te organiseren en waar nodig extra isolatie aan te brengen.

Daarnaast werden enkele losse verbindingen gedetecteerd bij de header-pins van de ESP32. Dit werd verholpen door deze opnieuw te solderen met meer soldeer en een betere soldeertechniek.

Tot slot werd vastgesteld dat de strain relief (trekontlasting) voor de externe sensoren onvoldoende was, wat kon leiden tot kabelbreuk bij beweging. Dit werd verbeterd door rubber grommet-doorvoeren toe te voegen en de kabels beter te verankeren in de behuizing.

## User Story 12:

Als ontwikkelaar heb ik het graag dat er een stevige en draagbare behuizing hebt, om de componenten in te steken.

### Testbeschrijving en methode: Real World Test

De behuizing moet stevig, comfortabel en waterbestendig zijn, en bieden voldoende bescherming aan de elektronische componenten.

Waarom een real-world test?

Een real-world test is gekozen omdat het horloge bedoeld is voor gebruik tijdens sportactiviteiten, waarbij het blootgesteld kan worden aan zweet, stoten, vallen en verschillende weersomstandigheden. Door de behuizing te testen onder realistische omstandigheden, kunnen we bepalen of deze voldoende bescherming biedt voor langdurig gebruik.

Testscenario's

In het eerste scenario werd de behuizing getest op waterdichtheid door deze onder te dompelen in water tot een diepte van 30 cm gedurende 10 minuten. Hierbij werd verwacht dat geen water zou binnendringen en de elektronica zou beschadigen.

Vervolgens werd de schokbestendigheid getest door de behuizing vanaf een hoogte van 1 meter op een harde ondergrond te laten vallen. Hierbij werd gecontroleerd of de behuizing intact bleef en de elektronica nog correct functioneerde na de val.

In het derde scenario werd het draagcomfort getest door verschillende testpersonen het horloge gedurende een dag te laten dragen tijdens normale activiteiten. Hierbij werd feedback verzameld over comfort, zweterigheid en eventuele irritatie van de huid.

Tot slot werd de thermische belastbaarheid getest door het horloge bloot te stellen aan temperaturen tussen 0°C en 40°C, om te bepalen of de behuizing geen vervorming vertoonde en de elektronica correct bleef functioneren.

Verwijzing naar testresultaten: Zie Excel-bestand, tabblad User Story 12.

Problemen & Oplossingen

Tijdens de tests werden verschillende problemen vastgesteld. Allereerst bleek de behuizing niet volledig waterdicht te zijn bij de aansluitingen van de sensoren. Dit werd opgelost door siliconen afdichtingsmiddel toe te passen rond alle openingen en kabeldoorvoeren. Een ander probleem was dat de behuizing barstte bij de valtest, vooral bij de hoeken. Dit werd verbeterd door de behuizing opnieuw te ontwerpen met afgeronde hoeken en verstevigingen op kritieke punten, en door een schokabsorberend materiaal toe te voegen aan de binnenkant. Daarnaast meldden testgebruikers dat de polsband oncomfortabel werd na langdurig dragen, vooral tijdens het sporten. Dit werd opgelost door een ademende, verstelbare band te ontwerpen met meer ventilatieopeningen. Tot slot bleek dat bij hoge temperaturen (boven 35°C) de behuizing licht vervormde, wat de waterdichtheid in gevaar bracht. Dit probleem werd aangepakt door een hittebestendiger materiaal te gebruiken voor de behuizing en extra ventilatieopeningen toe te voegen om de interne temperatuur te verlagen.