REPUBLIQUE DUCAMEROUN
Paix-Travail-Patrie
****
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

UNIVERSITE DE YAOUNDE I

FACULTE DES SCIENCES

DEPARTEMENT D'INFORMATIQUE

REPUBLIC OF CAMEROON
Peace-Work-Fatherland
****
MINISTRY OF HIGHER EDUCATION

UNIVERSITY OF YAOUNDE I

FACULTY OF SCIENCE

COMPUTER DEPARTMENT

RAPPORT

THEME :
PLAN DE TEST POUR UNE APPLICATION MOBILE D'UN RESTAURANT

Liste des membres

- ASSONFACK YEMENE BERAL 21T2501
- CHUDJO TCHUENDEM DIVINE RAYANA 20V2019
- FOUADJI FOSSO HERMANN EDMOND 21T2822
- TSAKEU NGUEMO MARILYN FLORA 21T2627

Supervisé par :
Dr KIMBI Xaveria

Année Académique :2023-2024

Introduction :
Ce plan de test couvre les test d'une application mobile pour pour un restaurant local. L'application vise à permettre aux utilisateurs de créer un panier de produits et d'obtenir des promotions.

Objectif :
Vérifier le bon fonctionnement de l'application mobile du restaurant local et s'assurer qu'elle répond aux exigences spécifiées et se comporte comme attendue. Cela implique entrre autres :
- s'assurer que les utilisateurs puisse construire comme il se doit un panier pour leurs different articles.
-verifier que le calcul du montant total des commandes soit correcte.
- verifier que les remises soit bien prises en compte quand il le faut
--verifier que l'utilisateur puisse créer correctement un nouvel article
- verifier que la recherche des articles soit fonctionnelle.
- verifier si les operations relatives à l'organisation des evènement et l'etablissement des promotions se deroule correctement.
-veriffier la performance et la fiabilité de l'application.
-identifiez les bugs et les defauts

portée de test :
Ce plan de test couvre les fonctionnalités principales de l'application mobile du restaurant local, y compris la navigation, la création des comptes utilisateurs, la commande de repas, le paiement en ligne , la réservation , la gestion des évenements et les promotions sur les commandes.

Approche de test : nous procéderons à un ensemple de tests unitaires qui inclura les tests par valeur negatives, les tests par valeurs positives ,les tests par valeurs limites.

Les ressources de test :
Personnel : quatre étudiants en licence informatique ayant acquis des connaissance en test logiciel qui feront office de testeurs, d'analyste et de développeur. Le chef de projet est notre examinateur.
Outils de test : nous utiliserons le frameworks de test javaScript Vitest.
Les Données de tests:

- d**onnée ordianaire** :event:{id:3,name :'Classical Night}, [item1, item2], userId :123456, currencyCode : « GBP », percentage : 75,discountCode : 'SUMMER20' ,currentTotal:300 .
- **Valeurs limites** : currentPrice, minimumSpend,
- **valeurs negatives** : userName

Calendrier de test :

Activité de test	**Debut**	**Fin**	**Ressources**	**Dependance**
Analyse des exigences	06-06-2024	08-06-2024	Testeur ,analyste	
Conception des cas de tests	10-06-2024	12-06-2024	Testeur , analyste	Analyse des exigences
Developpement des cas de tests	12-06-2024	15-06-2024	testeur	Conception des cas de tests

Execution des tests unitaires	15-06-2024	17-06-2024	Developpeur, testeur	Developpement des cas de tests
Reporting des résultats de tests	17-06-2024	18-06-2024	Testeur,chef projet	Execution des tests unitaires

Definir les cas de tests :

i) **fonction : isSoldOut()**
test case :
•id: eis_1
•description: should return true if event remaining tickets is sold out
•type: unit test
•data test: Event(1, 'concert', 2000, 500, 0, '18/06/2024');
•résultat actuel: true
•résultat attendu: true

test case :
•id: eis_2
•description: should return false if event remaining tickets is greather than 0
•type: unit test
•données test: Event(2, 'concert', 2000, 500, 1, '18/06/2024')
•résultat actuel: false
•résultat attendu: false

ii) fonction : getTagLine()
test case :
•id: egt_1
•description: should return event sold out if number of tickets for remaining is 0
•type: unit test
•données test: Event(1, 'concert', 2000, 500, 0, '18/06/2024')
•résultat actuel: event sold out
•résultat attendu: event sold out

test case :•id: egt_2
•description: should return tag line matching `Hurry` if only few places left'
•type: unit test
•données test: Event(2, 'concert', 2000, 500, 11, '18/06/2024')
•résultat actuel: Hurry only 11 tickets left!
•résultat attendu: tag line matching `Hurry`

test case :
•id: egt_3
•description: should return tag line matching ` a lot of interest` if event is popular and many tickets are left
•type: unit test
•données test: Event(3, 'concert', 2000, 500, 60, '18/06/2024')
résultat actuel: This Event is getting a lot of interest. Don't miss out, purchase your

ticket now!
•
résultat attendu: tag line matching ` a lot of interest`
test case :
•id: egt_4
•description: should return tag line matching `purchase your ticket now` if event is not popular and many tickets are left
•type: unit test
•données test: Event(4, 'concert', 2000, 500, 60, '18/06/2024')
•résultat actuel: Don't miss out, purchase your ticket now!
•résultat attendu: tag line matching ` a lot of interest`

ii) fonction : createEvent()
test case :
•id: ece_1
•description: should return error matching 'cannot exceed 200 characters' if event name is not string
•
type: unit test•données test: name: null, price: 1000, availableTickets: 100,
•résultat actuel: 'Event name cannot exceed 200 characters'
•résultat attendu: error matching 'cannot exceed 200 characters'

test case :
•id: ece_2
•description: should return error matching 'cannot exceed 200 characters' if event name length is greater than 200
•type: unit test
•données test: name: 'a'.repeat(201), price: 1000, availableTickets: 100,
•résultat actuel: 'Event name cannot exceed 200 characters'
•résultat attendu: error matching 'cannot exceed 200 characters'

test case :
•id: ece_3
•description: should return error matching 'more or equal to 0' if event price is not a number
•type: unit test
•données test: price: '1000', name: 'concert', availableTickets: 100,
•résultat actuel: 'Event price must be more or equal to 0'
•résultat attendu: error matching ‘more or equal to 0’

test case :
•id: ece_4
•description: should return error matching 'more or equal to 0' if event price is less than zero
•type: unit test
•données test: name: 'concert', price: -1000, availableTickets: 100,
•résultat actuel: 'Event price must be more or equal to 0'
•résultat attendu: error matching 'more or equal to 0'

test case :
•id: ece_5
•description: should return error matching 'more or equal to 0' if available tickets is

not a number
•type: unit test
•données test: name: 'concert', price: 1000, availableTickets: '100'
•résultat actuel: 'Event tickets must be more than 0'
•résultat attendu: error matching 'more or equal to 0'

test case :
•id: ece_6
•description: should return error matching 'more or equal to 0' if available tickets is less than zero
•type: unit test
•données test: name: 'concert', price: 1000, availableTickets: -100
•résultat actuel: 'Event tickets must be more than 0'
•résultat attendu: error matching 'more or equal to 0'

test case :
•id: ece_6
•description: should return an event if valid parameters
•type: unit test
•données test: name: 'concert', price: 1000, availableTickets: 100
•résultat actuel: event(null, 'concert', 1000, 100)
•résultat attendu: event(null, 'concert', 1000, 100)

a- fichier filters.js

i) fonction : today()
test case :
•id: etd_1
•description:should return true if event is today
•type: unit test•données test: Event(1, 'concert', 2000, 500, 0, new Date()
•résultat actuel: true
•résultat attendu: true

test case :
•id: etd_2
•description:should return false if event is not today
•type: unit test
•données test: Event(1, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() +1)
•résultat actuel: false
•résultat attendu: false

ii) fonction: next7Days()
test case :
•id: en7_1
•description:should return true if event is in the next seven days
•type: unit test
•données test: Event(1, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() +5))
•résultat actuel: true
•résultat attendu: true

test case :

•id: en7_2
•description:should return true if event is the 7th day after today
•type: unit test
•données test: Event(2, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() + 7))
•résultat actuel: true
•résultat attendu: true

test case :
•id: en7_3•description: should return false if event is after the next seven days
•type: unit test
•données test: Event(3, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() + 8))
•résultat actuel: false
•résultat attendu: false

test case :
•id: en7_4
•description: should return false if event is past
•type: unit test
•données test: Event(4, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() -1))
•résultat actuel: true
•résultat attendu: true

ii) fonction: next30Days()
test case :
•id: en30_1
•description:should return true if event is in the next 30 days
•type: unit test
•données test: Event(1, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() +25))
•résultat actuel: true
•résultat attendu: true

test case :
•id: en30_2
•description:should return true if event is the 30th day after today
•type: unit test
•données test: Event(2, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() + 30))
•résultat actuel: true
•résultat attendu: true

test case :
•id: en30_3
•description: should return false if event is after the next 30 days
•type: unit test
•données test: Event(3, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() + 31))
•résultat actuel: false
•résultat attendu: false

test case :
•id: en30_4
•description: should return false if event is past
•type: unit test
•données test: Event(4, 'concert', 2000, 500, 0, new Date().setDate(now.getDate() -1))
•résultat actuel: true
•résultat attendu: true

c) fichier search.js

i) fonction getEvents()
test case :
•id: ege_1
•description: should return only events matching search predicate'
•type: unit test
•données test:
let event1 = new Event(1, 'concert Fally Ipupa', 20000, 5000, 4500, '18/06/2024');
let event2 = new Event(2, 'Miss competition', 2000, 500, 500, '21/08/2024');
let event3 = new Event(3, 'concert Tenor', 5000, 3000, 2785, '03/09/2024');
let event4 = new Event(4, 'AI conference', 2000, 1500, 1500, '09/011/2024');
let event5 = new Event(5, 'Dance Battle', 0, 500, 5000, '31/06/2024');
const events = [event1, event2, event3, event4, event5];const searchByName = (event) => {
return event.name.toLowerCase().includes('concert');
};
•résultat actuel: [event1, event3]
•résultat attendu: [event1, event3]

test case :
•id: ege_2
•description: should return empty array if no event matching search predicate
•type: unit test
•données test:
let event1 = new Event(1, 'concert Fally Ipupa', 20000, 5000, 4500, '18/06/2024');
let event2 = new Event(2, 'Miss competition', 2000, 500, 500, '21/08/2024');
let event3 = new Event(3, 'concert Tenor', 5000, 3000, 2785, '03/09/2024');
let event4 = new Event(4, 'AI conference', 2000, 1500, 1500, '09/011/2024');
let event5 = new Event(5, 'Dance Battle', 0, 500, 5000, '31/06/2024');
const events = [event1, event2, event3, event4, event5];
const searchByDate = (event) => {
return event.name.toLowerCase().includes('01/01/2035’);
};
•résultat actuel: [ ]
•résultat attendu: [ ]

Rapport et communication :

exemple : Les tests ont révélés un problème au niveau du dossier Users . Ces derniers ont été corrigés et mentionnés sous forme de commentaire à l’intérieur du code.

Conclusion

Ce plan de test couvre les principales fonctionnalités de l'application mobile du restaurant local et vise à s'assurer que l'application répond aux exigences spécifiées. Les cas de test définis permettront de vérifier le bon fonctionnement de la navigation, de la commande de repas, du paiement en ligne, de la réservation de table et d'autres fonctionnalités clés.La procédure de test décrite implique l'utilisation d'appareils et de systèmes d'exploitation représentatifs, afin de garantir la compatibilité de l'application sur les principales plateformes mobiles.Une fois les tests effectués, les problèmes et les bugs identifiés devront être rapportés et corrigés par l'équipe de développement. Un suivi régulier du processus de test et de la résolution des problèmes sera nécessaire pour garantir la qualité et la fiabilité de l'application mobile du restaurant local.