JURNAL TEKNIK INFORMATIKA

Vol. 5 No. 1 Tahun. 2024
ISSN 2809-1353
DOI : 10.58794/jekin.v5i1.909

Pengujian Unit dan Antarmuka Pengguna Pada Aplikasi Mushymatch *Mobile* Menggunakan *JUnit* dan *Mockito*

Intan Sri Ramadhan[1], Depandi Enda[2],Eva Yumami[3]
[1,2,3]Rekayasa Perangkat Lunak/Politeknik Negeri Bengkalis
e-mail: [1]intan.adinda70@gmail.com, [2]depandienda@polbeng.ac.id, , [3]evayumami@polbeng.ac.id

Abstract *– This research aims to conduct unit and interface testing on the MushyMatch mobile application, developed using Kotlin. Unit testing will cover key modules, including the login module, home page, search feature, mushroom detection page, detection feature, mushroom description page, and mushroom recipe page. Interface testing will focus on input validation, user interface responsiveness, layout, navigation, and testing across various devices to ensure consistency and ease of use. The JUnit framework and Mockito library are used for unit testing, while the Espresso framework is employed for interface testing. Functional testing with structured and systematic test scenarios is the method applied. The research aims to identify issues and defects in the MushyMatch mobile application and provide recommendations for improvements to enhance quality and user experience. This research is valuable for developers in testing and validating the performance, usability, and responsiveness of the MushyMatch mobile application to ensure it meets user needs.*

Keywords – Application MushyMatch Mobile, JUnit, Mockito, Unit Testing, User Interface Testing.

Abstrak **– Penelitian ini bertujuan untuk menguji unit dan antarmuka aplikasi MushyMatch mobile yang dikembangkan menggunakan Kotlin. Pengujian unit mencakup modul penting seperti modul login, halaman utama, fitur pencarian, deteksi jamur, deskripsi jamur, dan resep jamur. Pengujian antarmuka meliputi validasi input, responsivitas, tata letak, navigasi, serta pengujian pada berbagai perangkat untuk menjamin konsistensi dan kenyamanan. Framework JUnit dan library Mockito digunakan untuk pengujian unit, sementara Espresso digunakan untuk pengujian antarmuka. Pengujian fungsional dengan skenario terstruktur diterapkan untuk mengidentifikasi masalah dan cacat dalam aplikasi, serta memberikan rekomendasi perbaikan. Hasil penelitian ini diharapkan membantu pengembang dalam meningkatkan kualitas dan pengalaman pengguna aplikasi MushyMatch, memastikan aplikasi berfungsi optimal dan sesuai dengan kebutuhan pengguna.**

Kata Kunci **– Aplikasi MushyMatch *mobile*, Junit, Mockito, Pengujian Antarmuka Pengguna, Pengujian Unit.**

I. PENDAHULUAN

Aplikasi MushyMatch adalah sebuah aplikasi yang dirancang untuk membantu pengguna mengindentifikasi dan mendapatkan informasi tentang berbagai macam jenis jamur. Dengan menggunakan aplikasi ini, mereka dapat memanfaatkan fitur pencarian untuk menemukan jamur tertentu menggunakan deteksi visual untuk mengenali jenis jamur melalui gambar, serta membaca deskripsi dan resep terkait jamur tersebut. Namun aplikasi MushyMatch terdapat masalah pada sistem yakni *bug* yang mempengaruhi akurasi dan performa data dari jamur yang ditampilkan. Aplikasi ini juga terdapat permasalahan kurangnya responsivitas antarmuka pengguna, beberapa perangkat mengalami kendala pada tampilan sehingga mengurangkan kualitas pengalaman pengguna. Hal ini tentunya dapat menampilkan hasil yang tidak akurat dan mengganggu pengguna. Selama masa pengembangan sistem, aplikasi MushyMatch belum pernah dilakukan pengujian. Dengan begitu perlu dilakukan pengujian pada sistem terkait permasalahan yang ditemukan, pengujian unit menjadi salah satu pengujian yang cocok dengan permasalahan yang ada. Karena pengujian unit dapat memastikan bahwa komponen deteksi jamur bekerja dengan akurat dan handal dan pengujian antarmuka pengguna dapat memastikan bahwa antarmuka pengguna aplikasi memenuhi standar yang tinggi dalam hal desain, interaktivitas dan kegunaan.

Pengujian unit memastikan efektivitas dan pengoperasian program perangkat lunak yang bebas kesalahan. Dalam keadaan yang jarang terjadi, perangkat lunak pengembangan perangkat dapat mengakibatkan kode aplikasi tidak efektif atau tidak pernah digunakan. Bentuk pengujian ini dapat mengidentifikasi kode yang tidak efektif dan mengukur efektivitasnya dalam pengembangan perangkat lunak. Selain itu, pengembang menggunakan tes ini untuk mengidentifikasi kesalahan dan masalah tersembunyi dalam program[1]. Pengujian antarmuka pengguna memastikan bahwa antarmuka pengguna berfungsi sebagaimana mestinya dan memenuhi spesifikasi desain. Dengan melakukan pengujian antarmuka pengguna yang efektif, dapat mengungkap kesalahan dan kelemahan pada

antarmuka pengguna sebelum program diterapkan, sehingga memungkinkan memperbaikinya dan memberikan pengalaman yang baik bagi pengguna[2].

Setelah menentukan teknik pengujian yang digunakan, selanjutnya menentukan *test case* untuk membuat serangkaian skenario yang akan di eksekusi. Pada pengujian tidak dapat menjamin keakuratan semua hasil yang mungkin terjadi selama pengoperasian modul. Perencanaan dan desain uji kasus yang efektif membantu memastikan efektivitas perangkat lunak pada tingkat yang diinginkan[3].

Pengujian unit sebelumnya telah dilakukan oleh Hasibuan, pada penelitian ini dilakukannya sebuah pengujian unit untuk mengetahui fitur atau fungsionalitas yang telah dikembangkan oleh tim pengembang sudah sesuai dengan spesifikasi dari perusahaan[1]. Pengujian antarmuka sebelumnya juga telah dilakukan oleh Mohammed, penelitiannya menjelaskan penggunaan pendekatan otomatisasi pengujian antarmuka pengguna yang lebih canggih dalam konteks validasi BIOS. Penggunaan teknologi untuk melakukan pengujian antarmuka pengguna menggunakan OpenCV dan OCR yang bertujuan meningkatkan efiensi dan akurasi pengujian BIOS[4].

Berdasarkan permasalahan yang telah dijelaskan sebelumnya, penelitian ini bertujuan untuk melakukan pengujian unit dan pengujian antarmuka pengguna pada aplikasi MushyMatch mobile. Aspek yang akan diuji pada pengujian unit dilakukan pada fitur *login* dan fitur deteksi jamur. Untuk aspek yang diuji pada pengujian antarmuka pengguna mencakup validasi input, responsivitas antarmuka, tata letak, navigasi pada halaman *login*, registrasi, halaman utama, halaman *logout*, halaman deteksi jamur, halaman deksripsi jamu dan halaman resep jamur. Penelitian ini hanya sebatas melakukan pengujian pada aplikasi, tidak adanya dilakukan pengembangan pada aplikasi secara detail. Adapun *framework* yang digunakan untuk pengujian yakni pada pengujian unit menggunakan *framework* Junit dan *library* Mockito, sedangkan pada pengujian antarmuka pengguna menggunakan *framework* Espresso.

II. PENELITIAN YANG TERKAIT

Penelitian sebelumnya telah dilakukan pengujian unit pada proyek pengembangan modul manejemen pengguna, berhasil menemukan kesalahan dan cacat yang tidak terlihat saat program dijalankan. Pengujian dengan *test case* dilakukan secara manual dengan beberapa skenario pengujian dan hasilnya menunjukkan bahwa semua fitur telah memenuhi spesifikasi perusahaan, kecuali fitur *search* hanya mencapai 75%. Pengujian ini berdampak penting dalam memastikan kualitas dan kesesuaian modul manejemn pengguna dengan kebutuhan perusahaan[1].

Penelitian sebelumnya juga melakukan pengujian unit, dilakukan pengujian pada aplikasi web mobile bisnis jasa laundry yang menggunakan framework PHPUnit. Teknik pengujian dilakukan dengan *white box* untuk mengukur tingkat keberhasilan. Unit testing dengan PHPUnit menguji aplikasi berdasarkan unit terkecil dan melakukan pengujian terhadap setiap pernyataan dalam bentuk source code, terutama pada fungsi yang memiliki percabangan dalam setiap kelas. Tingkat keberhasilan pengujian diukur menggunakan teknik statement coverage, di mana setiap test case harus menjalankan setiap pernyataan dan mencapai persentase keberhasilan 100%[5].

Penelitian sebelumnya telah melakukan pengujian *white box* dengan teknik *basis path testing* pada perangkat lunak *website room*. Pengujian perangkat lunak dengan menggunakan metode penelitian ini dapat membantu menemukan kesalahan, *bug* dan kesalahan implementasi pada perangkat lunak *website room*. Perbaikan yang diperlukan dapat dilakukan untuk memastikan kualitas dan kinerja yang baik dari *website*[6].

Penelitian sebelumnya melakukan pengujian antarmuka pengguna, Hasil dari penelitian ini menunjukkan bahwa metode yang diusulkan berhasil mengotomatiskan pengujian antarmuka BIOS dan mengurangi waktu yang dibutuhkan dibandingkan dengan pengujian manual. Secara keseluruhan, penelitian ini memberikan kontribusi dalam pengembangan metode otomasi pengujian antarmuka BIOS menggunakan teknologi antarmuka pengguna dan elemen-elemen terkait[4].

Penelitian sebelumnya dilakukan pengujian tes automatis dengan menggunakan EvoMaster, hasil menunjukkan bahwa menggunakan EvoMaster berhasil mengurangi waktu pengembangan dan membantu dalam mempertahankan kualitas kode. EvoMaster menghasilkan lebih dari 19,5 juta kasus tes dan menemukan 78 kesalahan dalam API REST dalam 58 jam, yang kemudian diperbaiki antara iterasi[7].

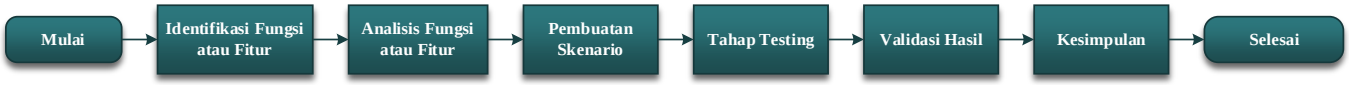
Penelitian selumnya melakukan pengujian black box dengan metode *equivalence partitions* pada penelitian ini berfokus fungsionalitas perangkat lunak. Hasil pengujian yang diperoleh dari penelitian ini adalah tabel rancangan Test Case, yang berfungsi untuk menyimpulkan apakah sistem berhasil dalam melakukan pengujian dan sesuai dengan rencana pengujian atau tidak. Dari pengujian yang dilakukan, dapat disimpulkan bahwa sistem informasi penilaian kinerja karyawan PT INKA (Persero) telah berjalan dengan baik dan tidak ditemukan kesalahan fungsionalitas pada setiap fitur yang diuji[8].

Penelitian sebelumnya melakukan pengujian antarmuka pengguna secara retroaktif untuk produk android yang sudah ada. Implementasi pengujian melibatkan spesifikasi berbasis perilaku yang ditulis dalam bahasa Gherkin dan implentasi uji yang ditulis dalam bahasa java menggunakan *framework* dan alat pengujian UI kontemporer. Penelitian ini menunjukkan bahwa pengujian berbasis perilaku merupakan metodologi yang efektif untuk implementasi retroaktif uji otomatis antarmuka pengguna pada android[9].

III. METODE PENELITIAN

A. Metode Penelitian

Penelitian ini terbagi menjadi enam tahapan, yang dimulai dengan identifikasi fungsi atau fitur, lalu dilanjutkan dengan analisis fungsi atau fitur, kemudian pembuatan skenario, tahapan selanjutnya tahap *testing*, lalu dilakukan validasi hasil dan terakhir membuat kesimpulan. Tahapan penelitian dapat dilihat pada gambar 1.



Gambar 1. Tahapan Penelitian

B. Identifikasi Fungsi atau Fitur

Tahapan awal melakukan identifikasi fungsi dan fitur yang akan diperiksa pada aplikasi deteksi jamur. Fitur yang dipilih harus krusial dan penting bagi aplikasi[10]. Login, registrasi, pengelolaan data jamur pencarian, deskripsi, resep, dan deteksi fitur yang akan diuji. Selain itu, penting untuk mengidentifikasi fungsi-fungsi dalam kode sumber aplikasi yang penting untuk logika aplikasi, pemrosesan data gambar, atau interaksi pengguna. Fungsi-fungsi ini mungkin cocok untuk pengujian tingkat unit. Metode pemilihan fitur atau fungsi ini akan membantu menentukan cakupan pengujian unit yang tepat, memastikan bahwa semua situasi penting telah diuji. Pengujian unit dapat digunakan untuk fokus pada proses seperti deteksi jamur, pengelolaan deskripsi jamur, dan navigasi halaman.

C. Analisis Fungsi atau Fitur

Tujuan dari tahapan analisis fungsi dan fitur adalah untuk lebih memahami logika dan proses fitur pendeteksi jamur, serta ketergantungannya pada komponen lain dalam program. Informasi ini akan digunakan untuk mengembangkan dan menjalankan pengujian unit khusus fitur. Selain itu, hubungan antara fitur identifikasi jamur dan fitur lain seperti pencarian, deskripsi jamur, dan resep akan diperiksa, sehingga memungkinkan keberhasilan integrasi fitur selama pengujian[11].

D. Pembuatan Scenario

Sebelum melakukan pengujian unit dan antarmuka pengguna pada program pendeteksi jamur, penting untuk terlebih dahulu membuat kasus pengujian yang sesuai. Skenario pengujian ini merupakan serangkaian langkah atau aktivitas yang menjelaskan bagaimana pengguna berinteraksi atau memproses aplikasi[12]. Berikut hasil pembuatan skenario pengujian untuk pengujian unit dan pengujian antarmuka pengguna.

1. Test Case Pengujian Antarmuka Pengguna

TABEL I
TEST CASE HALAMAN SPLASH SCREEN

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**
*TC_ SplashScreen*_1	UI Testing: SplashScreen	- Buka aplikasi -Periksa apakah tampilan SplashScreen muncul	SplashScreen ditampilkan dengan benar dan mengarahkan ke halaman login.

TABEL II
TEST CASE HALAMAN LOGIN

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**
*TC_Login*_1	UI Testing: Login	- Buka halaman login - Input email dan password - Klik tombol login	Halaman utama terbuka dengan informasi uang masuk, uang keluar, dan selisih
*TC_Login*_2	UI Testing: Registrasi	- Buka halaman login - Klik tombol registrasi - Input username, email, dan password - Klik tombol registrasi	- Jika email sudah terdaftar, registrasi gagal. - Jika registrasi berhasil, pengguna dialihkan ke halaman login.

TABEL III
TEST CASE HALAMAN DASHBOARD

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**
*TC_Dashboard*_1	UI Testing: Logout	- Buka halaman utama - Klik tombol logout - Konfirmasi "Yes" - Klik "Yes" - Konfirmasi "No" - Klik "No"	- Pengguna diarahkan ke halaman login jika logout. - Pengguna tetap di halaman utama jika membatalkan logout.
*TC_Dashboard*_2	UI Testing: Pencarian Jamur	- Buka halaman utama - Input kata kunci pencarian - Klik tombol "Cari"	- Hasil pencarian sesuai dengan kriteria muncul.

TABEL IV
TEST CASE HALAMAN DESKRIPSI JAMUR

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**
*TC_* Mushroom _1	UI Testing: Deskripsi Jamur (More Information)	- Klik jenis jamur yang ingin dilihat deskripsinya - Halaman deskripsi jamur muncul	Informasi jamur (jenis, habitat, edible, dll) ditampilkan.

TABEL V
TEST CASE HALAMAN RESEP JAMUR

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**
*TC_Recipes*_1	UI Testing: Deskripsi Resep Jamur	- Klik tombol "Cooking Recipes" di halaman deskripsi jamur - Halaman resep muncul	Resep jamur, termasuk gambar, ingredient, dan steps, ditampilkan.

TABEL VI
TEST CASE HALAMAN COOKING RECIPES

Test Case ID	**Deskripsi Test Case**	**Aksi**	**Hasil yang Diharapkan**
TC_ VideoTutorial _1	UI Testing: Tampilan Video Tutorial Resep	- Pilih resep masakan - Panggil fungsi video tutorial resep	- Pengguna melihat video tutorial resep yang sesuai.

TABEL VII
TEST CASE HALAMAN VIDEO TUTORIAL

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**
TC_ VideoTutorial _1	UI Testing: Tampilan Video Tutorial Resep	- Pilih resep masakan - Panggil fungsi video tutorial resep	- Pengguna melihat video tutorial resep yang sesuai.

TABEL VIII
TEST CASE HALAMAN SCAN JAMUR

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**
TC_UI_001	Memindai gambar jamur tiram	Melakukan proses Scan	Tampilan hasil analisis gambar jamur tiram
TC_UI_002	Memindai gambar jamur enoki	Melakukan proses Scan	Tampilan hasil analisis gambar jamur enoki
TC_UI_003	Memindai gambar jamur simeji putih	Melakukan proses Scan	Tampilan hasil analisis gambar jamur simeji putih
TC_UI_004	Memindai gambar jamur simeji coklat	Melakukan proses Scan	Tampilan hasil analisis gambar jamur simeji coklat
TC_UI_005	Memindai gambar yang bukan jamur	Melakukan proses Scan	Tampilan hasil analisis bahwa gambar tidak mengandung jamur

2. Test Case Pengujian Unit

TABEL IX
TEST CASE UNIT KODE AUTENTIKASI (Login/Registrasi/Logout)

Test Case ID	**Deskripsi**	**Input**	**Hasil yang Diharapkan**

	Test Case		
UTC_Authentication_001	Unit Testing: Autentikasi (Login)	- Input Email: user1@example.com - Input Password: password1 - Panggil fungsi autentikasi	- Fungsi autentikasi mengembalikan hasil berhasil. - Fungsi autentikasi mengembalikan hasil gagal dengan email salah, password salah, atau kombinasi salah.
UTC_Authentication_002	Unit Testing: Registrasi User	- Input Nama: NewUser, Email: newuser@example.com, Password: newpassword - Panggil fungsi registrasi user	- Fungsi registrasi mengembalikan hasil berhasil. - Fungsi registrasi mengembalikan hasil gagal jika email sudah digunakan.
UTC_Authentication_003	Unit Testing: Logout User	- Panggil fungsi logout - Konfirmasi "Yes" - Klik "Yes" - Konfirmasi "No" - Klik "No"	- Pengguna diarahkan ke halaman login jika logout. - Pengguna tetap di halaman utama jika membatalkan logout.

TABEL X
TEST CASE UNIT KODE PENCARIAN JAMUR

Test Case ID	**Deskripsi *Test Case***	**Input**	**Hasil yang Diharapkan**
UTC_MushroomSearch_001	Unit Testing: Pencarian Jamur	- Input kata kunci "simeji" - Panggil fungsi pencarian jamur - Input kata kunci "" - Panggil fungsi pencarian jamur - Input kata kunci "jamur_langka" - Panggil fungsi pencarian jamur	- Hasil pencarian sesuai dengan kriteria muncul jika ada hasil. - Tidak ada hasil pencarian jika kata kunci kosong. - Tidak ada hasil pencarian jika tidak ada data yang cocok.

TABEL XI
TEST CASE UNIT KODE DESKRIPSI JAMUR

Test Case ID	**Deskripsi *Test Case***	**Input**	**Hasil yang Diharapkan**
UTC_MushroomDescription_001	Unit Testing: Deskripsi Jamur	- Pilih jenis jamur - Panggil fungsi deskripsi jamur	- Pengguna melihat informasi jamur (jenis, habitat, edible, dll).

TABEL XII
TEST CASE UNIT KODE RESEP MASAKAN JAMUR

Test Case ID	**Deskripsi *Test Case***	**Input**	**Hasil yang Diharapkan**
UTC_CookingRecipes_001	Unit Testing: Tampilan Daftar Resep Masakan	- Panggil fungsi untuk menampilkan daftar resep masakan jamur	- Pengguna melihat daftar resep masakan jamur.
UTC_CookingRecipes_002	Unit Testing: Pilih Resep Masakan	- Pilih salah satu resep masakan - Panggil fungsi untuk menampilkan deskripsi resep masakan	- Pengguna melihat deskripsi resep masakan terpilih.

TABEL XIII
TEST CASE UNIT KODE RESEP MASAKAN JAMUR

Test Case ID	**Deskripsi *Test Case***	**Input**	**Hasil yang Diharapkan**
UTC_VideoTutorial_001	Unit Testing: Tampilan Video Tutorial Resep	- Pilih resep masakan - Panggil fungsi video tutorial resep	- Pengguna melihat video tutorial resep yang sesuai.

E. Tahap Testing

Pengujian unit digunakan untuk mengevaluasi unit kode secara terpisah. Selama langkah ini, setiap unit kode dalam program, seperti fungsi atau metode, diperiksa dan divalidasi[13]. Pengujian unit dilakukan dengan menggunakan kerangka pengujian seperti Mockito dan JUnit. Mockito menghasilkan objek tiruan sebagai pengganti dependensi eksternal atau objek canggih, sehingga pengujian unit dapat dijalankan secara terpisah. JUnit digunakan untuk membuat pernyataan yang memeriksa apakah hasil tes memenuhi harapan[14]. Pengujian unit dimaksudkan untuk menjamin bahwa unit kode bekerja dengan benar dan andal. Selain itu, pengujian antarmuka pengguna (pengujian UI) dilakukan untuk menilai interaksi antara pengguna dan program[15]. Langkah ini terdiri dari mengevaluasi tampilan aplikasi pendeteksi cetakan dan interaksi pengguna. Pengujian antarmuka pengguna dilakukan dengan menggunakan kerangka kerja seperti Espresso. Pengujian ini terdiri dari meniru aktivitas pengguna pada antarmuka aplikasi dan mengevaluasi respons program yang diharapkan. Pengujian antarmuka pengguna bertujuan untuk mengevaluasi pengalaman pengguna, memastikan kinerja optimal, dan menawarkan umpan balik yang sesuai kepada pengguna[16].

F. Validasi Hasil

Selama langkah validasi hasil, data pengujian yang dihasilkan oleh pengujian unit dan antarmuka pengguna diperiksa secara menyeluruh. Temuan pengujian dibandingkan dengan ekspektasi yang telah ditentukan berdasarkan spesifikasi dan tujuan aplikasi pendeteksi cetakan. Perbedaan antara hasil aktual dan hasil prediksi diselidiki dan dipelajari untuk mengetahui potensi penyimpangan atau ketidaksesuaian. Tahap validasi hasil ini sangat penting untuk menentukan apakah aplikasi deteksi jamur telah memenuhi kriteria keberhasilan yang dipersyaratkan dan siap digunakan. Hasil dari langkah validasi akan digunakan untuk membuat kesimpulan tentang penerimaan aplikasi, apakah diperlukan peningkatan, dan apakah aplikasi dapat diterapkan sepenuhnya.

G. Kesimpulan

Tahap selanjutnya dalam penelitian ini adalah membuat kesimpulan berdasarkan hasil analisis yang telah dilakukan. Pada titik ini, kesimpulan dibuat sebagai ringkasan dari hasil terpenting yang ditemukan selama penelitian. Langkah ini penting karena memberikan gambaran menyeluruh tentang kualitas dan kinerja aplikasi pendeteksi jamur yang telah dibuat. Hasil yang dikumpulkan dapat digunakan untuk membuat keputusan strategis mengenai peningkatan aplikasi yang ada dan pembuatan aplikasi baru. Keseluruhan prosedur ini memberikan perspektif komprehensif mengenai hasil studi dan memberikan panduan untuk pengembangan selanjutnya.

IV. HASIL DAN PEMBAHASAN

A. Hasil Pengujian

Pengujian antarmuka pengguna pada aplikasi MushyMatch Mobile dilakukan pada tujuh unit test case. Sedangkan hasil pengujian unit dlakukan pada empat unit test case. Semua hasil pengujian ditampilkan sebagai berikut.

B. Hasil Test Case Pengujian Antarmuka Pengguna

1. Hasil Test Case Untuk Halaman SplashScreen

TABEL XIV
HASIL TEST CASE HALAMAN SPLASHSCREEN

2. Hasil Test Case Untuk Halaman Login

TABEL XV
HASIL TEST CASE HALAMAN LOGIN

Test Case ID	**Deskripsi Test Case**	**Aksi**	**Hasil yang Diharapkan**	**Hasil yang Terjadi**	**Hasil Testing**
TC_Login_1	UI Testing: Login	- Buka halaman login - Input email dan password - Klik tombol login	Halaman utama terbuka dengan informasi uang masuk, uang keluar, dan selisih	Halaman utama terbuka dengan informasi uang masuk, uang keluar, dan selisih	*pass*
TC_Login_2	UI Testing: Registrasi	- Buka halaman login - Klik tombol registrasi - Input username, email, dan password - Klik tombol registrasi	- Jika email sudah terdaftar, registrasi gagal. - Jika registrasi berhasil, pengguna dialihkan ke halaman login.	- Jika email sudah terdaftar, registrasi gagal. - Jika registrasi berhasil, pengguna dialihkan ke halaman login.	*pass*

3. Hasil Test Case Untuk Halaman Dashboard

TABEL XVI
HASIL TEST CASE HALAMAN DASHBOARD

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**	**Hasil yang Terjadi**	**Hasil *Testing***
TC_Dashboard_1	*UI Testing: Logout*	- Buka halaman utama - Klik tombol *logout* - Konfirmasi *"Yes"* - Klik *"Yes"* - Konfirmasi *"No"* - Klik *"No"*	- Pengguna diarahkan ke halaman login jika *logout.* - Pengguna tetap di halaman utama jika membatalkan logout.	Menampilkan logchat error.	*pass*
TC_Dashboard_2	*UI Testing:* Pencarian Jamur	- Buka halaman utama - Input kata kunci pencarian - Klik tombol "Cari"	- Hasil pencarian sesuai dengan kriteria muncul.	- Hasil pencarian sesuai dengan kriteria muncul.	*pass*

4. Hasil Test Case Untuk Halaman Deskripsi Jamur

TABEL XVII
HASIL TEST CASE HALAMAN DESKRIPSI JAMUR

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**	**Hasil yang terjadi**	**Hasil *Testing***
TC_ Mushroom _1	*UI Testing:* Deskripsi Jamur *(More Information)*	- Klik jenis jamur yang ingin dilihat deskripsinya - Halaman deskripsi jamur muncul	Informasi jamur (jenis, habitat, *edible*, dll) ditampilkan.	Informasi jamur (jenis, habitat, *edible*, dll) ditampilkan.	*pass*

5. Hasil Test Case Untuk Halaman Resep Jamur

TABEL XVIII
HASIL TEST CASE HALAMAN RESEP JAMUR

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**	**Hasil yang Terjadi**	**Hasil *Testing***
TC_Recipes_1	*UI Testing:* Deskripsi Resep Jamur	- Klik tombol "Cooking Recipes" di halaman deskripsi jamur - Halaman resep muncul	Resep jamur, termasuk gambar, ingredient, dan steps, ditampilkan.	Resep jamur, termasuk gambar, ingredient, dan steps, ditampilkan.	*pass*

6. Hasil Test Case Untuk Halaman Cooking Recipes

TABEL XVIX
HASIL TEST CASE HALAMAN COOKING RECIPES

Test Case ID	**Deskripsi Test Case**	**Aksi**	**Hasil yang Diharapkan**	**Hasil yang Terjadi**	**Hasil Testing**
TC_ CookingRecipes1	UI Testing: Tampilan Daftar Resep Masakan	- Panggil fungsi untuk menampilkan daftar resep masakan jamur	- Pengguna melihat daftar resep masakan jamur.	- Pengguna melihat daftar resep masakan jamur.	pass
TC_ CookingRecipes_2	UI Testing: Pilih Resep Masakan	- Pilih salah satu resep masakan - Panggil fungsi untuk menampilkan deskripsi resep masakan	- Pengguna melihat deskripsi resep masakan terpilih.	- Pengguna melihat deskripsi resep masakan terpilih.	pass

7. Hasil Test Case Untuk Halaman SCAN JAMUR

TABEL XX
HASIL TEST CASE HALAMAN SCAN JAMUR

Test Case ID	**Deskripsi *Test Case***	**Aksi**	**Hasil yang Diharapkan**	**Hasil yang Terjadi**	**Hasil *Testing***
TC_UI_001	Memindai gambar jamur tiram	Melakukan proses Scan	Tampilan hasil analisis gambar jamur tiram	berhasil	*pass*
TC_UI_002	Memindai gambar jamur enoki	Melakukan proses Scan	Tampilan hasil analisis gambar jamur enoki	berhasil	*pass*
TC_UI_003	Memindai gambar jamur simeji putih	Melakukan proses Scan	Tampilan hasil analisis gambar jamur simeji putih	berhasil	*pass*
TC_UI_004	Memindai gambar jamur simeji coklat	Melakukan proses Scan	Tampilan hasil analisis gambar jamur simeji coklat	berhasil	*pass*
TC_UI_005	Memindai gambar yang bukan jamur	Melakukan proses Scan	Tampilan hasil analisis bahwa gambar tidak mengandung jamur	gagal	*failed*

C. Hasil Test Case Pengujian Unit

1. Test Case Untuk Unit Kode Halaman Autentikasi (Login.Registrasi/Logout)

TABEL XXI
HASIL TEST CASE UNIT KODE HALAMAN AUTENTIKASI

Test Case ID	**Deskripsi Test Case**	**Input**	**Hasil yang Diharapkan**	***Hasil Testing***
UTC_Authentication_001	Unit Testing: Autentikasi (Login)	- Input Email: user1@example.com - Input Password: password1 - Panggil fungsi autentikasi	- Fungsi autentikasi mengembalikan hasil berhasil. - Fungsi autentikasi mengembalikan hasil gagal dengan email salah, password salah, atau kombinasi salah.	*pass*
UTC_Authentication_002	Unit Testing: Registrasi User	- Input Nama: NewUser, Email: newuser@example.com, Password: newpassword - Panggil fungsi registrasi user	- Fungsi registrasi mengembalikan hasil berhasil. - Fungsi registrasi mengembalikan hasil gagal jika email sudah digunakan.	*pass*
UTC_Authentication_003	Unit Testing: Logout User	- Panggil fungsi logout - Konfirmasi "Yes" - Klik "Yes" - Konfirmasi "No" - Klik "No"	- Pengguna diarahkan ke halaman login jika logout. - Pengguna tetap di halaman utama jika membatalkan logout.	*pass*

2. Test Case Untuk Unit Kode Halaman Pencarian Jamur

TABEL XXII
HASIL TEST CASE UNIT KODE HALAMAN PENCARIAN JAMUR

Test Case ID	**Deskripsi Test Case**	**Input**	**Hasil yang Diharapkan**	***Hasil Testing***
UTC_MushroomSearch_001	Unit Testing: Pencarian Jamur	- Input kata kunci "simeji" - Panggil fungsi pencarian jamur - Input kata kunci "" - Panggil fungsi pencarian jamur - Input kata kunci "jamur_langka" - Panggil fungsi pencarian jamur	- Hasil pencarian sesuai dengan kriteria muncul jika ada hasil. - Tidak ada hasil pencarian jika kata kunci kosong. - Tidak ada hasil pencarian jika tidak ada data yang cocok.	*pass*

3. Test Case Untuk Unit Kode Halaman Deskripsi Jamur

TABEL XXIII
HASIL TEST CASE UNIT KODE HALAMAN DESKRIPSI JAMUR

Test Case ID	**Deskripsi Test Case**	**Input**	**Hasil yang Diharapkan**	***Hasil Testing***
UTC_MushroomDescription_001	Unit Testing: Deskripsi Jamur	- Pilih jenis jamur - Panggil fungsi deskripsi jamur	- Pengguna melihat informasi jamur (jenis, habitat, edible, dll).	*pass*

4. Test Case Untuk Unit Kode Halaman Resep Masakan Jamur

TABEL XXIV
HASIL TEST CASE UNIT KODE HALAMAN RESEP MAKANAN JAMUR

Test Case ID	**Deskripsi Test Case**	**Input**	**Hasil yang Diharapkan**	***Hasil Testing***

UTC_CookingRecipes_001	Unit Testing: Tampilan Daftar Resep Masakan	- Panggil fungsi untuk menampilkan daftar resep masakan jamur	- Pengguna melihat daftar resep masakan jamur.	*pass*
UTC_CookingRecipes_002	Unit Testing: Pilih Resep Masakan	- Pilih salah satu resep masakan - Panggil fungsi untuk menampilkan deskripsi resep masakan	- Pengguna melihat deskripsi resep masakan terpilih.	*pass*

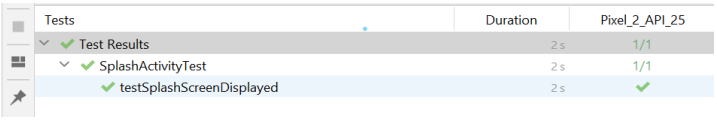
D. Validasi Hasil

Dalam tahap validasi hasil maka dilakukan pengecekan testing yang berhasil dan juga gagal.

1. Validasi Hasil Pengujian Antarmuka Pengguna

a. Validasi Hasil Pengujian Antarmuka Pengguna Halaman SplashScreen

```
@RunWith(AndroidJUnit4::class)
class SplashActivityTest {
    @Test
    fun testSplashScreenDisplayed() {
        onView(withId(binding.imageView5.id)).check(matches(isDisplayed()))}}
```



Gambar 1. Hasil Testing SplashScreen

Framework pengujian Espresso, yang dijelaskan oleh @RunWith(AndroidJUnit4::class), digunakan dalam pengujian ini untuk mengeksekusi kelas SplashActivityTest. Hasilnya, pengujian yang ketat memastikan bahwa bagian-bagian penting dari pengalaman pengguna pertama tercermin secara akurat dalam aplikasi Android yang sedang dikembangkan.

b. Validasi Hasil Pengujian UI Login

```
@Test
    fun loginSuccessMoveToHomeActivity() {
        Espresso.onView(ViewMatchers.withId(loginBinding.username.id))
            .perform(ViewActions.typeText("jkl@gmail.com"), ViewActions.closeSoftKeyboard())
        Espresso.onView(ViewMatchers.withId(loginBinding.password.id))
            .perform(ViewActions.typeText("jkl123"), ViewActions.closeSoftKeyboard())
        wrapViewModelIdlingResource(loginViewModel) {
            Espresso.onView(ViewMatchers.withId(loginBinding.btnLogin.id)).perform(ViewActions.click())}
        activity?.run { Log.d("hello", this::class.java.name) }
        val instrumentation = InstrumentationRegistry.getInstrumentation()
        wrapIntentIdlingResource(Intents.getIntents()) {
            activity = getCurrentActivity(instrumentation)
            homeBinding = (activity as HomePageActivity).binding}
        wrapShimmerIdlingResource(homeBinding.shimmerViewContainer) {
            Intents.intended(IntentMatchers.hasComponent(HomePageActivity::class.java.name))}}}
```

Gambar 2. Hasil Testing Halaman Login

Tujuan pengujian UI untuk halaman login adalah untuk menjamin bahwa pengguna berhasil menyelesaikan proses login sekaligus memvalidasi migrasi antarmuka pengguna (UI) ke halaman utama (Aktivitas Beranda) setelah login berhasil. Hasilnya, pengujian ini menjamin bahwa pengguna berhasil login dan aplikasi secara otomatis dialihkan ke halaman utama setelah login berhasil. Hal ini menjamin alur antarmuka pengguna dari halaman login ke halaman utama diterapkan dengan tepat pada aplikasi Android yang sedang dikembangkan.

c. Validasi Hasil Pengujian UI Registrasi

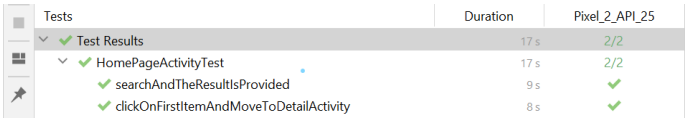
```
@Test
fun registerSuccessMoveToLoginActivity() {
    onView(withId(binding.username.id)).perform(typeText("John Doe"), closeSoftKeyboard())
    onView(withId(binding.email.id)).perform(typeText("john.doe@example.com"), closeSoftKeyboard())
    onView(withId(binding.password.id)).perform(typeText("password123"), closeSoftKeyboard())
    onView(withId(binding.btnMasuk.id)).perform(click())
    intended(hasComponent(LoginActivity::class.java.name))}}
```

Gambar 3. Hasil Testing Halaman Registrasi

Tujuan pengujian UI untuk halaman pendaftaran sama dengan pengujian sebelumnya: untuk memastikan bahwa pengguna berhasil menyelesaikan proses pendaftaran dan untuk memverifikasi bahwa antarmuka pengguna bertransisi ke halaman login setelah pendaftaran berhasil. Hasilnya, pengujian ini memverifikasi bahwa pengguna berhasil mendaftar, dan aplikasi akan secara otomatis dialihkan ke halaman login setelah pendaftaran berhasil. Hal ini menjamin alur antarmuka pengguna dari halaman registrasi hingga halaman login diimplementasikan dengan tepat dalam aplikasi Android yang sedang dikembangkan.

d. Validasi Hasil Pengujian UI Utama

```
@Test
fun searchAndTheResultIsProvided() {
    wrapShimmerIdlingResource(homeBinding.shimmerViewContainer) {
        Log.d("hello", "Before typeText1")
        onView(withId(homeBinding.searchView.id))
            .perform(ViewActions.typeText(keywordExpectation), ViewActions.closeSoftKeyboard())
        Log.d("hello", "Before typeText2")}
    onView(withId(homeBinding.rvMushrooms.id)).check(matches(isDisplayed()))}
@Test
fun clickOnFirstItemAndMoveToDetailActivity() {
    wrapShimmerIdlingResource(homeBinding.shimmerViewContainer) {
        onView(withId(homeBinding.searchView.id))
            .perform(ViewActions.typeText(keywordExpectation), ViewActions.closeSoftKeyboard())}
    onView(withId(homeBinding.rvMushrooms.id)).check(matches(isDisplayed()))
    onView(withId(homeBinding.rvMushrooms.id))
.perform(RecyclerViewActions.actionOnItemAtPosition<RecyclerView.ViewHolder>(0, click()))
    activity?.run { Log.d("hello", this::class.java.name) }}
```



Gambar 4. Hasil Testing Halaman Utama

Pengujian UI untuk halaman utama (Aktivitas Beranda) berupaya menjamin bahwa fitur-fitur utama berfungsi sebagaimana mestinya dan pengguna dapat berinteraksi dengan program dengan lancar. Dengan demikian, pengujian ini menjamin bahwa fungsi-fungsi penting seperti pencarian dan navigasi ke halaman detail berfungsi secara efektif dalam aplikasi Android yang sedang dikembangkan. Ini memberikan pengalaman pengguna yang lancar dan menyenangkan saat menggunakan perangkat lunak.

e. Validasi Hasil Pengujian UI Logout

```
@Test
   fun logoutAndMoveToLoginActivity() {
      Espresso.onView(ViewMatchers.withId(binding.btnLogout.id)).perform(ViewActions.click())
      Intents.intended(IntentMatchers.hasComponent(LoginActivity::class.java.name))}}
```

Gambar 5. Hasil Testing Halaman Logout

Pengujian UI untuk halaman logout berhasil menguji perintah terhadap testcase yang disediakan. Hasil pengujian menunjukkan bahwa testcase berhasil diselesaikan, dengan halaman LogoutActivity mampu mengarahkan pengunjung ke halaman LoginActivity sebagaimana dimaksud.

f. Validasi Hasil Pengujian UI Deskripsi

```
@Test
   fun checkMushroomDetails() {
      onView(ViewMatchers.isRoot()).perform(waitFor(5000))
      onView(withId(R.id.rv_mushrooms)).check(matches(isDisplayed()))
      onView(withId(R.id.rv_mushrooms)).perform(actionOnItemAtPosition<ViewHolder>(0,click()))
      onView(ViewMatchers.isRoot()).perform(waitFor( 5000))
      onView(withId(R.id.tv_mushroom_name)).check(matches(isDisplayed()))
      onView(withId(R.id.tv_mushroom_name)).check(matches(withText("Jamur Enoki")))
      onView(withId(R.id.tv_mushroom_scientific_name)).check(matches(withText("Flammulina velutipes")))
      onView(withId(R.id.tv_status)).check(matches(withText("Edible")))
      onView(withId(R.id.tv_desc)).check(matches(isDisplayed()))
      onView(withId(R.id.tv_description)).check(matches(withText("Jamur enoki (Flammulina velutipes) adalah
sejenis jamur yang memiliki batang panjang dan tipis dengan tudung kecil berwarna putih. Jamur enoki memiliki
tekstur renyah dan rasa manis..")))}
```

Gambar 6. Hasil Testing Halaman Deskripsi Jamur

Pengujian halaman deskripsi jamur berhasil. Tes ini menilai kemampuan aplikasi untuk bernavigasi dari daftar jamur ke deskripsi jamur yang dipilih. Dengan demikian, pengujian ini berhasil memvalidasi fungsi halaman deskripsi jamur (Informasi Jamur) dalam aplikasi dengan memastikan bahwa pengguna dapat dengan mudah beralih dari daftar jamur ke deskripsi jamur, dan bahwa informasi yang diberikan sesuai dengan yang diharapkan.

g. Validasi Hasil Pengujian UI Recipe

Gambar 7. Hasil Testing Halaman Recipe

Pengujian halaman Daftar Resep berhasil. Tes ini memeriksa kemampuan aplikasi untuk mengarahkan pengguna dari halaman detail jamur ke halaman daftar resep ketika mereka menekan tombol "Resep Memasak". Dengan demikian, pengujian ini berhasil memeriksa fungsionalitas pengalihan dari halaman informasi jamur ke daftar resep (Daftar Resep) dengan menekan tombol "Resep Masakan". Hal ini juga menjamin bahwa halaman daftar resep disajikan dengan tepat setelah pengguna melakukan tindakan.

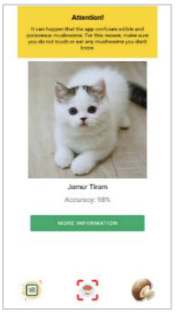
h. Validasi Hasil Pengujian UI Cooking Recipe

Gambar 8. Hasil Testing Halaman Cooking Recipe

Pengujian untuk menampilkan detail resep pada halaman Daftar Resep berhasil. Pengujian ini mengonfirmasi kemampuan aplikasi untuk menampilkan data resep saat pengguna memilih item dari daftar resep. Dengan demikian, pengujian ini berhasil memvalidasi kemampuan menampilkan detail resep dari halaman Daftar Resep, serta memastikan bahwa halaman detail resep disajikan dengan tepat setelah pengguna menjalankan tindakan.

i. Validasi Hasil Pengujian Ui Scan Jamur

Gambar 9. Hasil Testing Halaman Scan Jamur

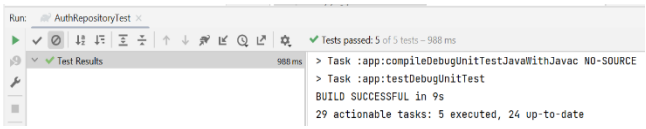


Gambar 10. Kesalahan Dalam Halaman Scan Jamur

Kesalahan muncul pada saat pengujian dimana hal atau entitas yang bukan jamur tetap terdeteksi sebagai jamur pada program, sehingga harus dilakukan perubahan.

2. Validasi Hasil Pengujian Unit

a. Validasi Hasil Pengujian Untuk Unit Kode Halaman Autentikasi

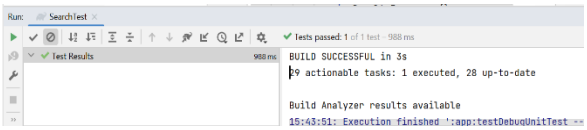


Gambar 11. Hasil Testing Unit Kode Halaman Autentikasi

Semua kasus uji untuk kelas 'AuthRepository' berhasil. Pengujian ini mencoba mengonfirmasi bahwa repositori autentikasi berfungsi seperti yang diharapkan dalam hal login, registrasi, dan logout. Hasilnya, semua pengujian unit pada kelas 'AuthRepository' lulus dan memenuhi harapan, menunjukkan bahwa fungsi repositori autentikasi berfungsi sebagaimana mestinya.

b. Validasi Hasil Pengujian Untuk Unit Kode Halaman Pencarian Jamur

```
@Test
   fun searchTest() {
      runTest {
         when(val result = repository.getMushrooms()) {
            is Result.Error -> {}
            is Result.Success -> {
               result.response.let { mushrooms ->
                  val name = mushrooms[0].name
                  assertEquals(name, expectedValue)}}}}}
```

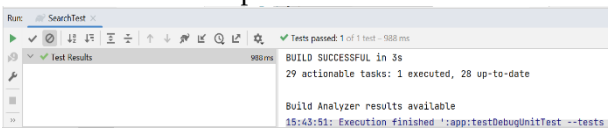


Gambar 12. Hasil Testing Unit Kode Search

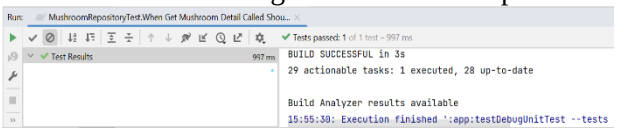
Pengujian unit untuk "Halaman Pencarian Jamur" berhasil. Pengujian ini memvalidasi fitur pencarian jamur aplikasi dengan memastikan bahwa jamur yang dicari berhasil ditemukan. Dengan demikian, pengujian ini berhasil

menunjukkan bahwa halaman pencarian jamur dapat menemukan jamur dengan nama yang sesuai dengan yang diharapkan.

c. Validasi Hasil Pengujian Untuk Unit Kode Deskripsi Jamur



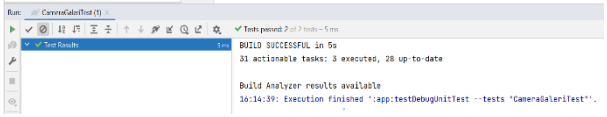
Gambar 13. Hasil Testing Unit Kode Deskripsi Jamur 1



Gambar 14. Hasil Testing Unit Kode Deskripsi Jamur 2

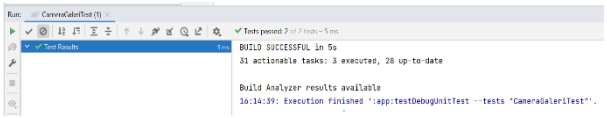
Tes unit untuk "Deskripsi Jamur" berhasil. Tes ini dirancang untuk memvalidasi fungsionalitas yang terhubung dengan detail jamur di repositori. Beberapa skenario pengumpulan data digunakan dalam pengujian ini, dan temuannya dianalisis. Dengan demikian, pengujian ini berhasil mengonfirmasi bahwa metode rincian resep dan jamur repositori menyediakan data yang sesuai.

d. Validasi Hasil Pengujian Untuk Unit Kode Scan Jamur Dari Galeri



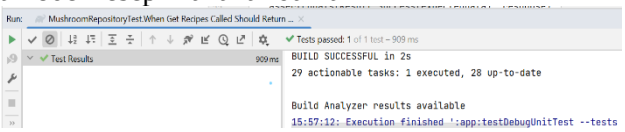
Gambar15. Hasil Testing Unit Kode Scan Jamu Dari Galeri

e. Validasi Hasil Pengujian Untuk Unit Kode Scan Jamur Dari Kamera

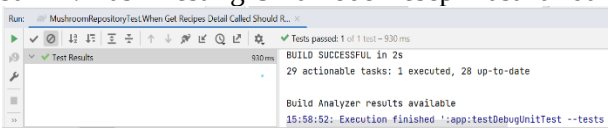


Gambar 16. Hasil Testing Unit Kode Scan Jamur Dari Kamera

f. Validasi Hasil Pengujian Unit Kode Resep Makanan Jamur



Gambar 17. Hasil Testing Unit Kode Resep Masakan Jamur 1



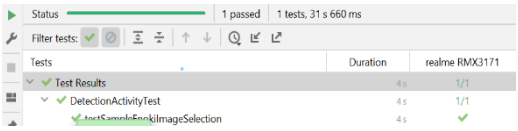
Gambar 18. Hasil Testing Unit Kode Resep Masakan Jamur 2

Temuan uji unit untuk "Resep Jamur" berhasil dan konsisten dengan harapan. Pengujian ini memeriksa dua skenario utama: pertama, saat metode 'getRecipes()' dipanggil, dan kedua, saat fungsi 'getRecipeDetail()' digunakan. Oleh karena itu, hasil pengujian unit ini menunjukkan bahwa metode terkait resep jamur di repositori memberikan data sebagaimana dimaksud.

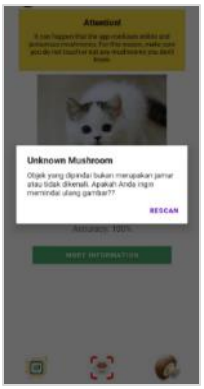
E. Validasi Hasil Setelah Perbaikan

Setelah melakukan pengujian unit dan antarmuka pengguna, hasilnya divalidasi. Terdapat dua halaman yang mengalami kendala khususnya pada saat pengujian user interface, yaitu halaman scan deteksi jamur dan halaman logout, sehingga dilakukan perbaikan dan hasil pengujian divalidasi kembali.

Setelah dilakukan analisis, diketahui bahwa kesalahan ada pada model yang telah diintegrasikan ke dalam proyek, yang tidak dimaksudkan untuk mengenali hal-hal di luar kategori jamur. Oleh karena itu, tindakan perbaikan dilakukan, termasuk mendesain ulang model. Pada modifikasi ini, model baru dikembangkan dengan menambahkan empat kelas jamur sebagai dataset baru, serta satu kelas tambahan yang disebut “tidak diketahui”. Kelas "tidak diketahui" ini dimaksudkan untuk mengidentifikasi item selain jamur.



Gambar 19. Hasil Testing Halaman Scan Setelah Dilakukan Perbaikan



Gambar 20. Halaman Scan Mampu Mengenali Entitas Yang Bukan Jamur

Jadi, jika model tidak dapat mengidentifikasi suatu objek sebagai jamur, sebuah pesan akan muncul yang memberi tahu pengguna bahwa gambar yang disediakan belum dikenali sebagai jamur. Pengguna kemudian akan ditawari opsi untuk memindai ulang untuk mendapatkan gambar yang lebih tajam dan mewakili jamur yang dimaksud. Langkah ini dilakukan untuk memastikan keakuratan pendeteksian objek dan meningkatkan kualitas pengalaman pengguna terhadap program.

Tantangan utama yang dihadapi selama proses pengembangan adalah prosedur pengembangan dan pembuatan tiruan yang memakan waktu. Meskipun tiruan diperlukan untuk mengisolasi komponen tertentu selama pengujian, metode pengujian harus disesuaikan untuk mengurangi pemborosan waktu dan mempercepat proses pengembangan.

V. KESIMPULAN

Setelah perbaikan pada model machine learning, aplikasi MushyMatch kini mampu mendeteksi objek dengan lebih akurat, membedakan antara jamur dan benda bukan jamur. Masalah deteksi yang sebelumnya terjadi telah berhasil diatasi, dan pengujian menunjukkan bahwa aplikasi berfungsi sesuai harapan. Semua fitur utama, termasuk deteksi jamur dan identifikasi benda bukan jamur, berfungsi dengan baik dan telah lolos pengujian yang ketat. Dengan demikian, MushyMatch siap dirilis ke publik, memenuhi standar kualitas yang diharapkan.

DAFTAR PUSTAKA

[1] A. N. Hasibuan and T. Dirgahayu, "Pengujian dengan Unit Testing dan Test case pada Proyek Pengembangan Modul Manajemen Pengguna," *AUTOMATA*, vol. 2, no. 1, 2021.

[2] J. C. Paiva, R. Queirós, J. P. Leal, J. Swacha, and F. Miernik, "Managing Gamified Programming Courses with the FGPE Platform," *Information (Switzerland)*, vol. 13, no. 2, Feb. 2022, doi: 10.3390/info13020045.

[3] S. T. Romeo, *Testing dan Implentasi Sistem*. Surabaya: STIKOM, 2003.

[4] E. A. A. Mohammed, M. Mustapa, H. Rahim, and M. N. Norizan, "Advanced UI test automation (AUTA) for BIOS validation using OpenCV and OCR," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 23, no. 3, pp. 1350–1356, Sep. 2021, doi: 10.11591/ijeecs.v23.i3.pp1350-1356.

[5] D. K. P. Rahayu, "Unit Testing Pada Aplikasi Web Mobile (Studi Kasus Bisnis Jasa Laundry)," 2020.

[6] J. B. L. Sie, I. A. Musdar, and S. Bahri, "Pengujian White Box Testing Terhadap Website Room Menggunakan Teknik Basis Path," *Kharisma Tech*, vol. 17, no. 2, pp. 45–57, 2022.

[7] S. Mujahidin, M. Reinaldy Hermawan, and C. Cahyo Utomo, "Implementation of Automated Test Case Generation in REST API on Android-Based Koperasi Application," *Journal of Information Systems and Informatics*, vol. 5, no. 1, pp. 123–133, Feb. 2023, doi: 10.51519/journalisi.v5i1.431.

[8] Y. Dwi Wijaya and M. Wardah Astuti, "Pengujian Blackbox Sistem Informasi Penilaian Kinerja Karyawan PT. Inka (PERSERO) Berbasis Equivalence Partitions," *Jurnal Digital Teknologi Informasi*, vol. 4, no. 1, pp. 22–26, 2021.

[9] A. Vilhunen, "User interface test automation for an Android application," 2022. [Online]. Available: www.aalto.fi

[10] A. Nurseptian, C. Riyana, and F. Rahmafitria, "Analisis Fungsi dan Fitur Pada Website Resmi Pariwisata Pemerintah Kota Bandung", [Online]. Available: www.bandungtourism.com.

[11] D. Evi, F. Agus, and F. Yanto, "Analisis User Experience (UX) Fitur Marketplace Facebook," *Jurnal Ekonomi dan Teknik Informatika*, vol. 8, no. 2, pp. 47–66, 2020.

[12] M. I. Shiddiq, "Implementasi White Box Testing Berbasis Path Pada Form Login Aplikasi Berbasis Web," *Siliwangi*, vol. 8, no. 1, pp. 1–6, 2022.

[13] W. Lewis, *Software Testing and Continuous Quality Improvement*. Boston: Auerbach Publication, 2009.

[14] R. Black, *Managing the Testing Process: Practical Tools and Techniques for Managing Hardware and Software Testing*. Hoboken: Wiley Publishing Inc, 2002.

[15] W. Dai, L. Riliskis, P. Wang, V. Vyatkin, and X. Guan, "A Cloud-Based Decision Support System for Self-Healing in Distributed Automation Systems Using Fault Tree Analysis," *IEEE Trans Industr Inform*, vol. 14, no. 3, pp. 989–1000, Mar. 2018, doi: 10.1109/TII.2018.2791503.

[16] Y.-P. Cheng, D. Liang, and W.-J. Wang, "KORAT — A platform independent test automation tool by emulating keyboard/mouse hardware signals," *IEEE AUTOTESTCON*, pp. 1–7, 2016.