APLIKASI PENGURUSAN HARTANAH TUAN RUMAH & PENYEWA BERASASKAN AWAN

Maxwell Hilary, Hasimi Sallehuddin

[1,2]*Fakulti Teknologi & Sains Maklumat, Universiti Kebangsaan Malaysia, 43600 UKM Bangi,, Selangor Darul Ehsan, Malaysia*

Abstrak

Tuan hartanah atau pemilik aset yang mempunyai aset yang banyak seperti rumahrumah kondominium mungkin tidak mendapat maklumat terkini tentang aset-aset mereka. Sering kali mereka menggunakan agen sebagai orang tengah dalam mendapatkan info, mencari orang yang hendak menyewa rumah atau bilik dan berkomunikasi dengan penyewa. Walau bagaimanapun, agen tidak selalu ada dan agenagen ini juga boleh menyebabkan salah faham di antara pemilik hartanah dan penyewa. Oleh itu, sebuah aplikasi harus dibangunkan bagi menyelesaikan atau mengurangkan isu-isu seperti yang telah dinyatakan.. Metodologi yang dicadangkan untuk diguna adalah metodologi air terjun atau Waterfall. Keperluan yang berpotensi, tarikh-tarikh akhir dan garis panduan akan dikenalpasti terlebih dahulu. Selepas itu dalam fasa analisis, spesifikasi sistem dianalisis bagi menajan modelmodel. Selepas itu, fasa reka bentuk mencipta dokumen keperluan spesifikasi bagi 3 memberikan garis panduan bagi keperluan teknikal seperti penggunaan bahasa pengaturcaraan dan sumber data. Ketika fasa pengekodan atau implementasi, kod sumber akan dibangunkan menggunakan model, logik dan keperluan spesifikasi yang telah ditetapkan di fasa-fasa sebelumnya. Seterusnya, fasa ujian menguji sistem yang dibangunkan seperti ujian beta dan ujian unit bagi menjamin kualiti sistem dan penyahpepijatan. Fasa operasi dan penempatan akan menggerakkan dan menggunakan sistem tersebut di industri atau tempat yang memerlukan sistem tersebut. Metodologi

tersebut dipilih disebabkan keperluan spesifikasi dikenalpasti sebelum memulakan fasa pengekodan dan ujian-ujian akan dijalankan selepas sistem dibangunkan bagi memastikan ia berfungsi seperti yang disangkakan. Hasil dapatan utama projek adalah Aplikasi Pengurusan Hartanah Tuan Rumah & Penyewa Berasaskan Awan yang berfungsi mengikut Spesifikasi Keperluan Dokumen. Kesimpulannya, aplikasi dibangunkan dengan objektif memudahkan urusan berkaitan dengan aset seperti rumah sewa bagi orang-orang tertentu khususnya penyewa dan tuan rumah dicapai dengan objektif dengan jelas dan efektif.

Pengenalan

Tuan hartanah atau pemilik aset seperti kondominium dan inap desa tidak mempunyai sistem yang membolehkan mereka untuk mengemaskini maklumat mengenai aset-aset mereka dengan mudah serta penyewa aset mereka (jika ada). Mereka juga tidak mendapat info terkini mengenai aset mereka kecuali melalui agen mereka. Objektifnya adalah untuk mewujudkan satu aplikasi yang mudah berasakan awan yang memudahkan tuan hartanah dan penyewa untuk membayar sewa, menyemak bayaran dan memuatnaikkan gambar yang relevan secara efektif. Skop projek adalah tuan rumah yang mempunayi aset dan penyewa. Batasan projek adalah dengan maklumat yang diperoleh melalui pengguna haruslah digunakan dengan betul. Isu-isu kajian adalah masa untuk membangunkan sistem ini tidak mencukupi dan kualiti dan prestasi aplikasi ini berkemungkinan tidak seperti yang diharapkan. Bagi sorotan sastera, Sistem tradisional Tuan Hartanah-Agen-Penyewa ini merupakan sistem yang telah lama digunakan oleh kebanyakan tuan hartanah. Komunikasi kebanyakkan adalah melalui agen sama ada secara bersemuka atau menggunakan aplikasi komunikasi seperti Whatsapp. Walau bagaimanapun, terdapat kelemahan dalam sistem ini seperti risiko penipuan dan perselisihan faham yang boleh mengancam kepercayaan dalam bisnes ini. Oleh itu, dengan adanya sistem yang tidak memerlukan agen, tuan hartanah lebih fokus dan dapat menjejaki rekod-rekod bayaran, maklumat penyewa terkini dan lepas serta kondisi aset-aset mereka. Terdapat juga aplikasi seperti

Landlordy Property Management yang dibangunkan oleh Landlord apps by E-protect yang boleh membantu tuan hartanah untuk menjejak dan menguruskan sewaan penyewa aset. Fitur-fitur aplikasi ini adalah laporan, peringatan pengemaskinian maklumat aset, laporan terkini aliran tunai, meberikan peringatan untuk pembayaran yang belum langsai dan jejak bayaran sewa. Namun begitu, tuan hartanah tidak boleh melihat kondisi terkini aset yang didiami penyewa. Tambahan lagi, hanya tuan hartanah sahaja yang boleh mengemaskinikan maklumat aset di mana terdapat risiko 4 penipuan di mana penyewa telah membayar sewa tetapi tuan hartanah tidak merekodkan bayaran tersebut. Hal ini menyebabkan kepercayaan penyewa terhadap tuan hartanah akan berkurang dan kerugian wang. Oleh itu, profil bagi penyewa dan tuan hartanah haruslah diwujudkan bagi mengurangkan risiko tersebut. Selain itu, Rootways juga telah membangunkan aplikasi pengurusan hartanah dan aplikasi tersebut mempunyai fungsian-fungsian yang lebih kurang sama seperti aplikasi Landlordy. Fitur-fitur aplikasi tersebut adalah laporan, peringatan dan pengemaskinian maklumat aset, lihat butiran penyewa, lihat butiran harta benda, hantar notifikasi melalui SMS dan antara muka yang menarik dan ringkas. Walau bagaimanapun, aplikasi yang dibangun Rootways ini mempunyai fitur yang lebih baik seperti implementasi profil bagi tuan hartanah dan penyewa. Hal ini mengurangkan risiko penipuan dalam pembayaran sewa, kerosakan dan lain-lain. Akhir sekali, antara aplikasi yang sering digunakan untuk pengurusan hartanah adalah aplikasi Landlord Studio. Aplikasi ini boleh digunakan secara desktop dan aplikasi dalam telefon pintar. Seperti aplikasi Landlordy, aplikasi ini hanya mempunyai satu profil sahaja yang khususnya untuk pemilik hartanah. Oleh itu, segala pengemaskinian maklumat tentang aset adalah daripada pemilik hartanah sahaja. Begitu juga dengan bayaran sewa dari penyewa aset. Antara muka aplikasi tersebut menarik dan ringkas seperti Aplikasi Pengurusan Harta Sewa oleh Rootways. Fitur lain yang terdapat dalam aplikasi ini adalah penyelenggaraan harta dan pemeriksaan penyewa. Metodologi yang digunakan adalah metodologi air terjun atau Waterfall. Keperluan yang berpotensi, tarikh-tarikh akhir dan garis panduan haruslah dikenalpasti sebelum ke fasa analisis. Dalam fasa analisis, spesifikasi sistem dianalisis bagi membentuk model. Selepas itu, fasa reka bentuk

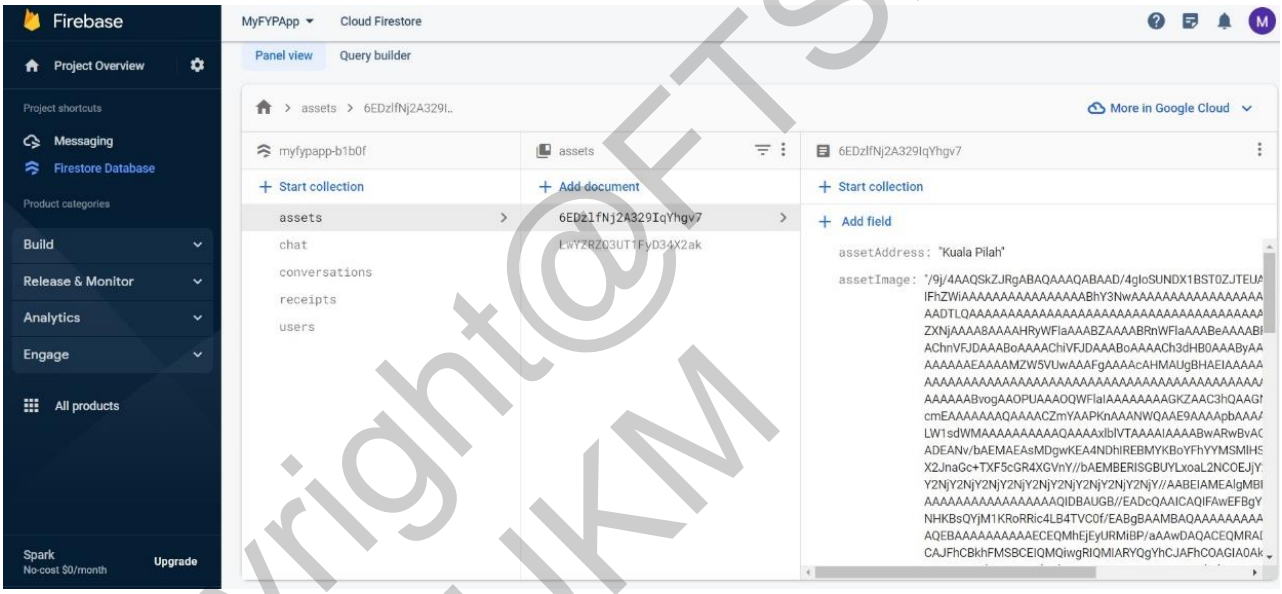
mencipta dokumen keperluan spesifikasi bagi 3 memberikan garis panduan bagi keperluan teknikal seperti penggunaan bahasa pengaturcaraan dan sumber data. Ketika fasa pengekodan atau implementasi, kod sumber akan dibangunkan menggunakan model, logik dan keperluan spesifikasi yang telah ditetapkan di fasa-fasa sebelumnya. Seterusnya, fasa ujian menguji sistem yang dibangunkan seperti ujian beta dan ujian unit bagi menjamin kualiti sistem dan penyahpepijatan. Fasa operasi dan penempatan akan menggerakkan dan menggunakan sistem tersebut di industri atau tempat yang memerlukan sistem tersebut. Metodologi tersebut dipilih disebabkan keperluan spesifikasi dikenalpasti sebelum memulakan fasa pengekodan dan ujian-ujian akan dijalankan selepas sistem dibangunkan bagi memastikan ia berfungsi seperti yang disangkakan. Dalam laporan ini, metodologi kajian akan menerangkan lebih lanjut mengenai model dan proses pembangunan berdasarkan model tersebut. Keputusan dan Perbincangan akan menerangkan tentang segmen kod kritikal yang ada dalam aplikasi tersebut. Kesimpulan memberikan ringkasan berdasarkan Keputusan dan Perbincangan dan keseluruhan projek.

Metodologi Kajian

Metodologi yang digunakan adalah metodologi air terjun atau Waterfall. Keperluan yang berpotensi, tarikh-tarikh akhir dan garis panduan haruslah dikenalpasti sebelum ke fasa analisis. Dalam fasa analisis, spesifikasi sistem dianalisis bagi membentuk model. Selepas itu, fasa reka bentuk mencipta dokumen keperluan spesifikasi bagi 3 memberikan garis panduan bagi keperluan teknikal seperti penggunaan bahasa pengaturcaraan dan sumber data. Ketika fasa pengekodan atau implementasi, kod sumber akan dibangunkan menggunakan model, logik dan keperluan spesifikasi yang telah ditetapkan di fasa-fasa sebelumnya. Seterusnya, fasa ujian menguji sistem yang dibangunkan seperti ujian beta dan ujian unit bagi menjamin kualiti sistem dan penyahpepijatan. Fasa operasi dan penempatan akan menggerakkan dan menggunakan sistem tersebut di industri

atau tempat yang memerlukan sistem tersebut. Metodologi tersebut dipilih disebabkan keperluan spesifikasi dikenalpasti sebelum memulakan fasa pengekodan dan ujian-ujian akan dijalankan selepas sistem dibangunkan bagi memastikan ia berfungsi seperti yang disangkakan. Metodoloji air terjun sesuai digunakan dalam projek ini kerana projek ini memerlukan hantaran yang senang ditentukan dari mula. Malah, ia juga digunakan untuk menentukan tujuan akhir aplikasi lebih awal.

Keputusan dan Perbincangan



(Gambarajah 1)

Seperti yang ditunjukkan dalam Gambarajah 1, Firestore merupakan pangkalan data bagi Aplikasi Mudah Alih Pengurusan Hartanah Tuan Rumah Dan Penyewa yang dibangunkan oleh Google. Selain itu, proses pengesahan pengguna juga berlaku di dalam Firebase menggunakan nama, emel dan kata laluan.

```
1 usage
private void register(){
    loading(true);
    FirebaseFirestore database = FirebaseFirestore.getInstance();
    HashMap<String, Object> user = new HashMap<>();
    user.put(Constants.KEY_NAME, binding.inputName.getText().toString());
    user.put(Constants.KEY_EMAIL, binding.inputEmail.getText().toString());
    user.put(Constants.KEY_PASSWORD, binding.inputPassword.getText().toString());
    user.put(Constants.KEY_TYPE_OF_USER, binding.inputTypeOfUser.getText().toString());
    user.put(Constants.KEY_IMAGE, encodedImage);
    database.collection(Constants.KEY_COLLECTION_USERS) CollectionReference
            .add(user) Task<DocumentReference>
            .addOnSuccessListener(documentReference -> {
                loading(false);
                preferenceManager.putBoolean(Constants.KEY_IS_SIGNED_IN, true);
                preferenceManager.putString(Constants.KEY_USER_ID, documentReference.getId());
                preferenceManager.putString(Constants.KEY_NAME, binding.inputName.getText().toString());
                preferenceManager.putString(Constants.KEY_IMAGE, encodedImage);
                Intent intent = new Intent(getApplicationContext(), Login_Page.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
                startActivity(intent);
            })
            .addOnFailureListener(exception -> {
                loading(false);
                showToast(exception.getMessage());
            });
}
```

(Gambarajah 2)

Gambarajah 2 menunjukkan segmen kod bagi daftar pengguna. Fungsi register() mempunyai peranan untuk mendaftarkan pengguna baharu tuan rumah atau penyewa. Apabila pengguna baru telah selesai memasukkan emel, kata laluan, nama dan juga sama ada mereka adalah tuan rumah atau penyewa, mereka akan menekan butang “Register Account” dalam paparan “Register”. Setiap data yang dimasukkan akan dikumpulkan mengikut jenis data dalam HashMap user. Kemudian, ia akan dimasukkan ke dalam Firestore database dalam bentuk ‘collection’ melalui fungsi terbina iaitu .add(user). “preferenceManager” akan menyimpan beberapa data berkaitan pengguna secara lokal.

```
1 usage
private void signIn(){
    loading(true);
    FirebaseFirestore database = FirebaseFirestore.getInstance();
    database.collection(Constants.KEY_COLLECTION_USERS) CollectionReference
            .whereEqualTo(Constants.KEY_EMAIL, binding.inputEmailTenant.getText().toString()) Query
            .whereEqualTo(Constants.KEY_PASSWORD, binding.inputPasswordTenant.getText().toString())
            .get() Task<QuerySnapshot>
            .addOnCompleteListener(task -> {
                if (task.isSuccessful() && task.getResult() != null
                        && task.getResult().getDocuments().size() > 0) {
                    DocumentSnapshot documentSnapshot = task.getResult().getDocuments().get(0);
                    preferenceManager.putBoolean(Constants.KEY_IS_SIGNED_IN, true);
                    preferenceManager.putString(Constants.KEY_USER_ID, documentSnapshot.getId());
                    preferenceManager.putString(Constants.KEY_NAME, documentSnapshot.getString(Constants.KEY_NAME));
                    preferenceManager.putString(Constants.KEY_IMAGE, documentSnapshot.getString(Constants.KEY_IMAGE));
                    Intent intent = new Intent(getApplicationContext(), Home_Tenant.class);
                    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
                    startActivity(intent);
                }else  {
                    loading(false);
                    showToast("Unable to sign in :')");
                }
            });
}
```

(Gambarajah 3)

```
private void signIn(){
    loading(true);
    FirebaseFirestore database = FirebaseFirestore.getInstance();
    database.collection(Constants.KEY_COLLECTION_USERS) CollectionReference
            .whereEqualTo(Constants.KEY_EMAIL, binding.inputEmail.getText().toString()) Query
            .whereEqualTo(Constants.KEY_PASSWORD, binding.inputPassword.getText().toString())
            .whereEqualTo(Constants.KEY_TYPE_OF_USER, binding.inputLandlord.getText().toString())
            .get() Task<QuerySnapshot>
            .addOnCompleteListener(task -> {
                if (task.isSuccessful() && task.getResult() != null
                        && task.getResult().getDocuments().size() > 0) {
                    DocumentSnapshot documentSnapshot = task.getResult().getDocuments().get(0);
                    preferenceManager.putBoolean(Constants.KEY_IS_SIGNED_IN, true);
                    preferenceManager.putString(Constants.KEY_USER_ID, documentSnapshot.getId());
                    preferenceManager.putString(Constants.KEY_NAME, documentSnapshot.getString(Constants.KEY_NAME));
                    preferenceManager.putString(Constants.KEY_IMAGE, documentSnapshot.getString(Constants.KEY_IMAGE));
                    Intent intent = new Intent(getApplicationContext(), Home_Landlord.class);
                    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
                    startActivity(intent);
                }else  {
                    loading(false);
                    showToast("Unable to sign in :')");
                }
            });
}
```

(Gambarajah 4)

Gambarajah 3 dan 4 serupa dalam kod tetapi mempunyai fungsi yang berbeza. Rajah 3 log masukkan penyewa ke dalam halaman utama manakala Rajah 4 memasukkan tuan rumah ke dalam halaman utama jika berjaya. Secara ringkasnya, kod-kod ini mengendalikan proses log masuk kedua-dua dengan menyemak e-mel dan kata laluan yang dimasukkan terhadap pangkalan data Firestore.

```
private void sendMessage(){
    HashMap<String, Object> message = new HashMap<>();
    message.put(Constants.KEY_SENDER_ID, preferenceManager.getString(Constants.KEY_USER_ID));
    message.put(Constants.KEY_RECEIVER_ID, receiverUser.id);
    message.put(Constants.KEY_MESSAGE, binding.inputMessage.getText().toString());
    message.put(Constants.KEY_TIMESTAMP, new Date());
    database.collection(Constants.KEY_COLLECTION_CHAT).add(message);
    if (conversationId != null){
        updateConversation(binding.inputMessage.getText().toString());
    }else {
        HashMap<String, Object> conversation = new HashMap<>();
        conversation.put(Constants.KEY_SENDER_ID, preferenceManager.getString((Constants.KEY_USER_ID)));
        conversation.put(Constants.KEY_SENDER_NAME, preferenceManager.getString(Constants.KEY_NAME));
        conversation.put(Constants.KEY_SENDER_IMAGE, preferenceManager.getString(Constants.KEY_IMAGE));
        conversation.put(Constants.KEY_RECEIVER_ID, receiverUser.id);
        conversation.put(Constants.KEY_RECEIVER_NAME, receiverUser.name);
        conversation.put(Constants.KEY_RECEIVER_IMAGE, receiverUser.image);
        conversation.put(Constants.KEY_LAST_MESSAGE, binding.inputMessage.getText().toString());
        conversation.put(Constants.KEY_TIMESTAMP, new Date());
        addConversation(conversation);

    }
    binding.inputMessage.setText(null);
}
```

(Gambarajah 5)

Gambarajah 5 menunjukkan segmen kod bagi fungsi dalam bual. Kod tersebut mengendalikan penghantaran mesej antara pengguna dalam sistem sembang menggunakan Firebase Firestore sebagai pangkalan data. Ia menyimpan mesej dalam koleksi sembang dan mengemas kini perbualan yang berkaitan dengan mesej terkini. Jika tiada perbualan sedia ada, ia mencipta perbualan baharu dengan butiran berkaitan tentang pengirim dan penerima.

```
private final EventListener <QuerySnapshot> eventListener = (value, error) -> {
    if (error != null){
        return;
    }
    if (value != null){
        int count = chatMessages.size();
        for (DocumentChange documentChange : value.getDocumentChanges()){
            if (documentChange.getType()== DocumentChange.Type.ADDED){
                ChatMessage chatMessage = new ChatMessage();
                chatMessage.senderId = documentChange.getDocument().getString(Constants.KEY_SENDER_ID);
                chatMessage.receiverId = documentChange.getDocument().getString(Constants.KEY_RECEIVER_ID);
                chatMessage.message = documentChange.getDocument().getString(Constants.KEY_MESSAGE);
                chatMessage.dateTime = getReadableDateTime(documentChange.getDocument().getDate(Constants.KEY_TIMESTAMP));
                chatMessage.dateObject = documentChange.getDocument().getDate(Constants.KEY_TIMESTAMP);
                chatMessages.add(chatMessage);
            }
        }
        Collections.sort(chatMessages, (obj1, obj2) -> obj1.dateObject.compareTo(obj2.dateObject));
        if(count == 0){
            chatAdapter.notifyDataSetChanged();
        }else {
            chatAdapter.notifyItemRangeInserted(chatMessages.size(), chatMessages.size());
            binding.chatRecyclerview.smoothScrollToPosition(chatMessages.size() - 1);
        }
        binding.chatRecyclerview.setVisibility(View.VISIBLE);
    }
    binding.progressbar.setVisibility(View.GONE);
    if (conversationId == null){
        checkForConversation();
    }
};
```

(Gambarajah 6)

Gambarajah 6 menunjukkan segmen kod bagi fungsi dalam bual. Kod tersebut peka dengan perubahan dalam koleksi Firestore yang mengandungi mesej sembang dan mengemas kini UI dengan sewajarnya. Ia mendapatkan semula mesej sembang baharu, menambahkannya pada senarai mesej sedia ada, mengisihnya berdasarkan cap masanya, dan kemudian memberitahu penyesuai untuk memaparkan mesej dalam sembang RecyclerView. Selain itu, ia menyemak perbualan sedia ada dan menetapkannya jika perlu.

```
private void setListeners(){
    binding.fabBack.setOnClickListener( view -> onBackPressed());

    String bankUri = "https://play.google.com/store/apps/details?id=my.com.maybank2u.m2umobile";
    binding.btnProceedToBank.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // Create an explicit intent to launch the desired activity
            Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(bankUri));
            startActivity(intent);

        }
    });
}
```

(Gambarajah 7)

Gambarajah 7 menunjukkan segmen kod bagi fungsi dalam bayaran. Kod tersebut menyediakan pendengar acara untuk dua butang dalam UI. Butang fabBack membolehkan pengguna menavigasi kembali ke skrin atau aktiviti sebelumnya apabila diklik. Butang btnProceedToBank, apabila diklik, membuka halaman Gedung Google Play untuk apl perbankan tertentu (Maybank2u Mudah Alih) menggunakan niat yang jelas, memberikan pengguna pilihan untuk melihat dan memuat turun apl daripada kedai.

```
private void addReceipt(){
    FirebaseFirestore database = FirebaseFirestore.getInstance();
    HashMap<String, Object> receipt = new HashMap<>();
    receipt.put(Constants.KEY_RECEIPT_NAME, binding.editTextReceiptName.getText().toString());
    receipt.put(Constants.KEY_RECEIPT_DESCRIPTION, binding.editTextDescription.getText().toString());
    receipt.put(Constants.KEY_UPLOADER_TENANT_NAME, preferenceManager.getString(Constants.KEY_NAME));
    receipt.put(Constants.KEY_RECEIPT_IMAGE, encodedImage);
    database.collection(Constants.KEY_COLLECTION_RECEIPT) CollectionReference
            .add(receipt) Task<DocumentReference>
            .addOnSuccessListener(documentReference -> {
                preferenceManager.putString(Constants.KEY_UPLOADER_TENANT_NAME, preferenceManager.getString(Constants.KEY_NAME));
                Intent intent = new Intent(getApplicationContext(), Receipt_Tenant.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
                startActivity(intent);
            })
            .addOnFailureListener(exception -> {
                showToast(exception.getMessage());
            });
}
```

(Gambarajah 8)

Gambarajah 8 menunjukkan segmen kod bagi fungsi dalam resit Penyewa. Kod ini mengendalikan penambahan resit pada pangkalan data Firestore. Ia mengumpulkan butiran resit daripada UI, termasuk nama resit, perihalan, nama penyewa pemuat naik dan imej resit (dalam borang dikodkan Base64), dan menyimpan maklumat ini ke Firestore. Jika penambahan berjaya, pengguna dialihkan ke skrin paparan resit dan jika terdapat ralat, mesej ralat dipaparkan menggunakan pemberitahuan ‘showToast’.

```
private void addAsset(){
    FirebaseFirestore database = FirebaseFirestore.getInstance();
    HashMap<String, Object> asset = new HashMap<>();
    asset.put(Constants.KEY_ASSET_NAME, binding.editTextAssetName.getText().toString());
    asset.put(Constants.KEY_ASSET_LANDLORD_NAME, preferenceManager.getString(Constants.KEY_NAME));
    asset.put(Constants.KEY_ASSET_ADDRESS, binding.editTextAssetAddress.getText().toString());
    asset.put(Constants.KEY_ASSET_IMAGE, encodedImage);
    database.collection(Constants.KEY_COLLECTION_ASSETS) CollectionReference
            .add(asset) Task<DocumentReference>
            .addOnSuccessListener(documentReference -> {
                preferenceManager.putString(Constants.KEY_ASSET_LANDLORD_NAME, preferenceManager.getString(Constants.KEY_NAME));
                Intent intent = new Intent(getApplicationContext(), Assets_Landlord.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
                startActivity(intent);
            })
            .addOnFailureListener(exception -> {
                showToast(exception.getMessage());
            });
}
```

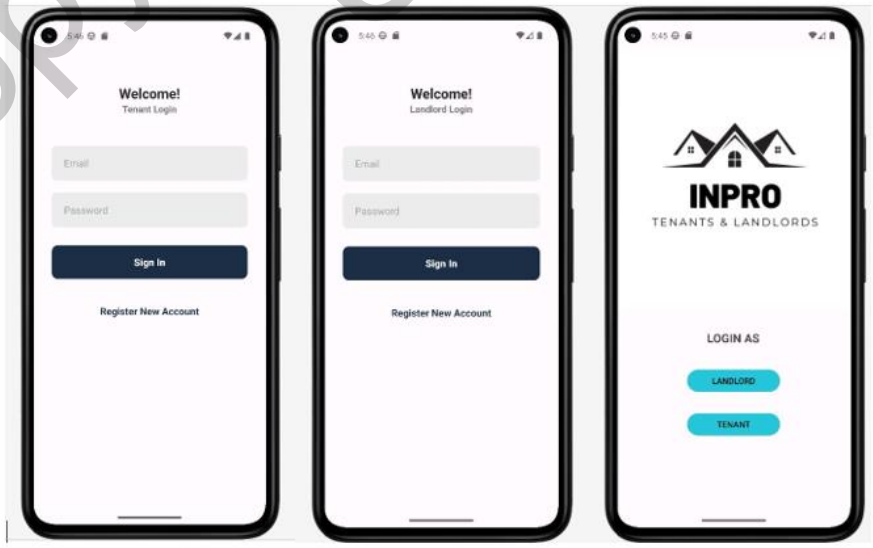
(Gambarajah 9)

Gambarajah 9 segmen kod bagi fungsi dalam asset Tuan Hartanah. Kod tersebut mengendalikan penambahan aset pada pangkalan data Firestore. Ia mengumpul butiran aset daripada UI, termasuk nama aset, nama tuan tanah, alamat dan imej aset (dalam borang dikodkan Base64) dan menyimpan maklumat ini ke Firestore. Jika penambahan itu berjaya, pengguna akan diubah hala ke skrin pengurusan aset dan jika terdapat ralat, mesej ralat dipaparkan menggunakan pemberitahuan 'showToast'.

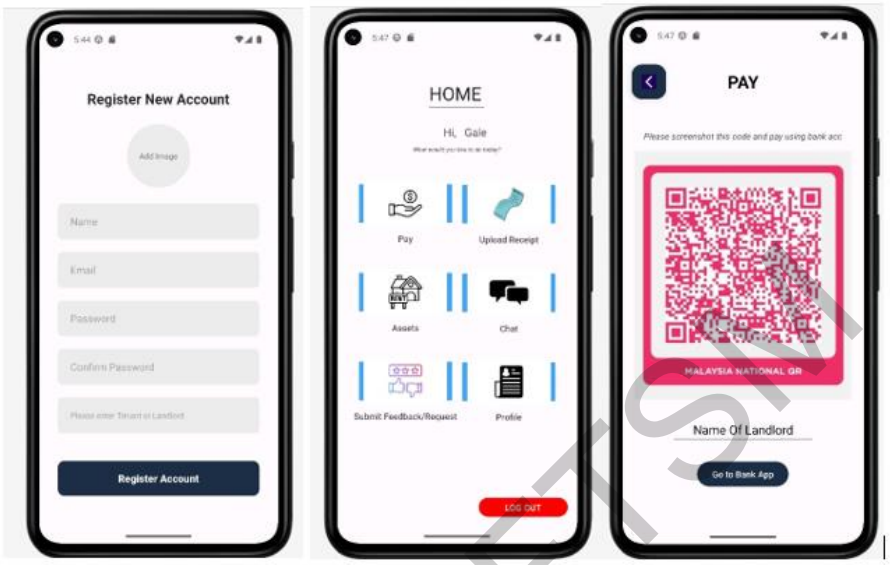
```
private void signOut(){
    showToast("Signing out...");
    FirebaseFirestore database = FirebaseFirestore.getInstance();
    DocumentReference documentReference =
            database.collection(Constants.KEY_COLLECTION_USERS).document(
                    preferenceManager.getString((Constants.KEY_USER_ID))
            );
    HashMap<String, Object> updates = new HashMap<>();
    updates.put(Constants.KEY_FCM_TOKEN, FieldValue.delete());
    documentReference.update(updates)
            .addOnSuccessListener(unused -> {
                preferenceManager.clear();
                startActivity(new Intent(getApplicationContext(), Login_Landlord.class));
                finish();
            })
            .addOnFailureListener(e -> showToast("Unable to sign out"));
}
```

(Gambarajah 10)

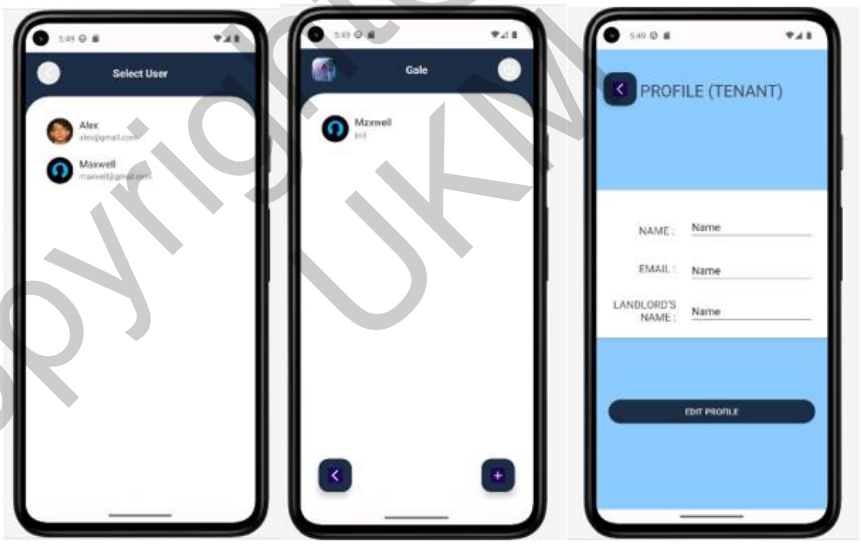
Gambarajah 10 menunjukkan segmen kod bagi fungsi untuk log keluar. Kod ini mengendalikan proses log keluar untuk pengguna (tuan tanah) dalam aplikasi. Ia mengalih keluar token FCM yang dikaitkan dengan pengguna, mengosongkan pilihan yang dikongsi untuk mengalih keluar data berkaitan pengguna, mengubah hala pengguna ke skrin log masuk dan menyelesaikan aktiviti semasa untuk memastikan pengguna tidak boleh menavigasi semula ke sana selepas log keluar. Log keluar untuk penyewa juga sama seperti tuan tanah.



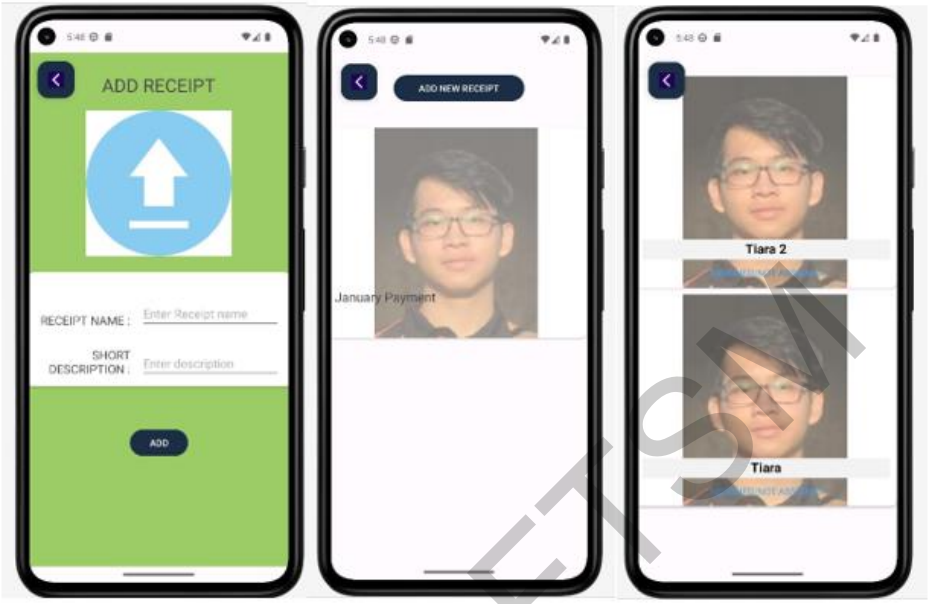
(Gambarajah 11)



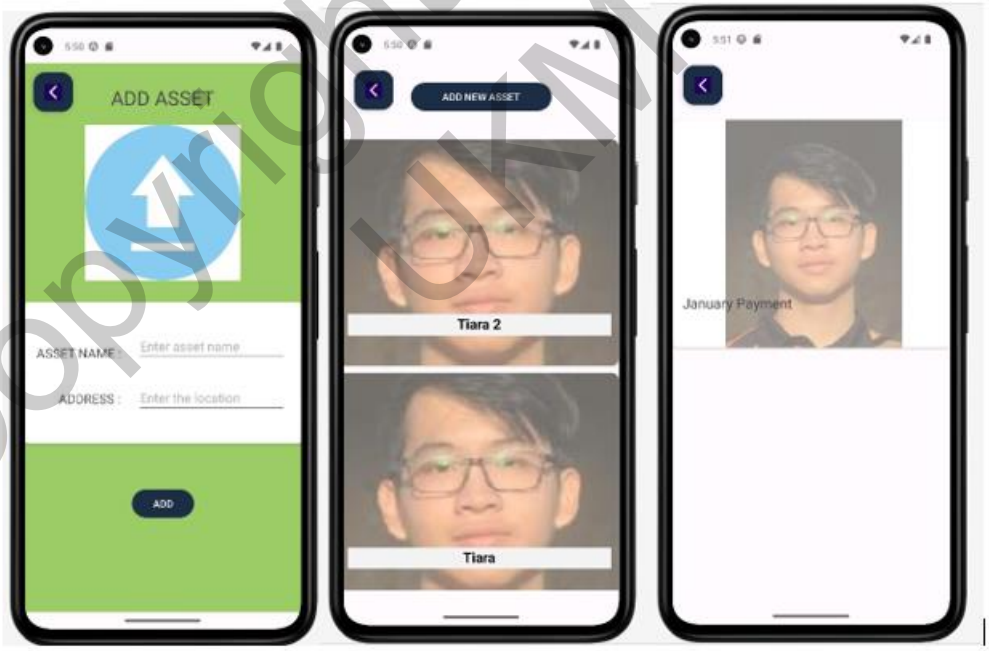
(Gambarajah 12)



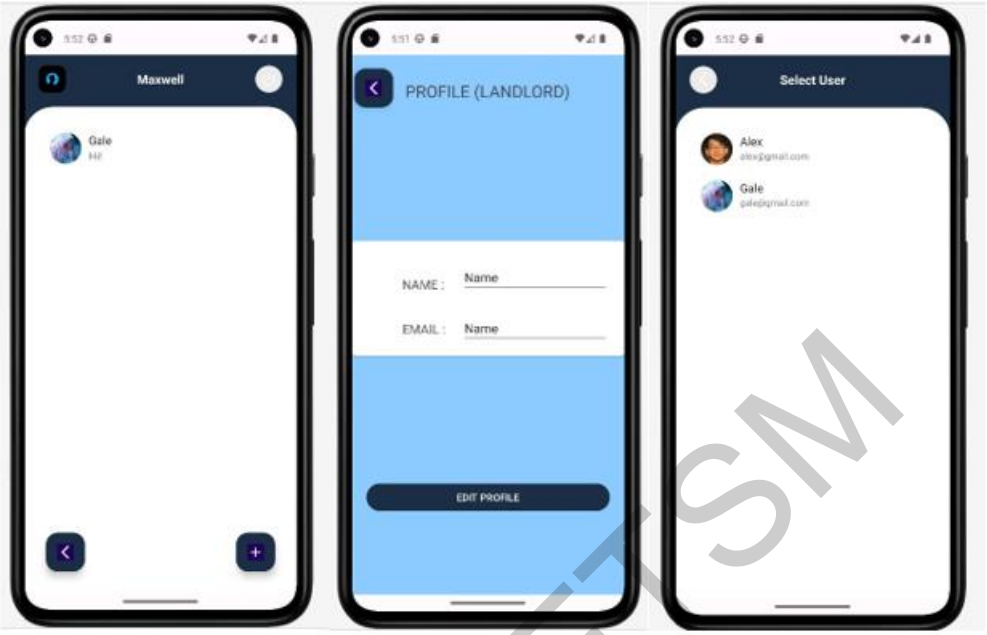
(Gambarajah 13)



(Gambarajah 14)



(Gambarajah 15)



(Gambarajah 16)

Gambarajah 11 hingga 16 menunjukkan antara muka aplikasi.

Kesimpulan

Aplikasi Mudah Alih Pengurusan Hartanah Tuan Rumah Dan Penyewa merupakan aplikasi yang dibangunkan bagi memudahkan urusan berkaitan dengan aset seperti rumah sewa bagi orang-orang tertentu khususnya penyewa dan tuan rumah. Aplikasi ini akan membantu golongan ini untuk tidak bergantung kepada orang tengah untuk untuk membayar sewa, menyemak bayaran dan memuatnaikkan gambar yang relevan secara efektif. Aplikasi ini dapat digunakan oleh dua jenis pengguna yang berbeza iaitu penyewa dan tuan rumah. Tuan rumah dapat menambah aset mereka dan penyewa dapat memuatnaikkan resit bayaran mereka ke dalam sistem supaya dapat dilihat oleh tuan rumah merekea. Antara kelebihan aplikasi ini adalah penyewa dan tuan rumah dapat berinteraksi antara satu sama lain bagi dalam aplikasi tersebut. Selain itu, pengguna aplikasi hanya perlu menggunakan aplikasi tersebut untuk menguruskan hal berkenaan sewa dan tidak perlu berjumpa secara berdepan di mana masa dapat dijimatkan untuk melakukan pekerjaan yang lain. Antara

kelemahan aplikasi ini adalah ketiadaan fungsi pembayaran secara rasmi seperti FPX dan Razer Payment Services disebabkan kompleksiti untuk memasukkan pengekodan. Selain itu, tiada sokongan iOS kerana aplikasi dibangunkan menggunakan Android Studio dan ketiadaan label atau teks untuk mengetahui orang yang pengguna berbual adalah penyewa atau tuan rumah dan mengurangkan kesahihan jenis pengguna. Antara cadangan yang dapat dikemukakan bagi menaik taraf dan memperbaiki lagi projek dan aplikasi ini adalah menggunakan pelayan dan pangkalan data yang lebih rumit dan selamat untuk integrasi fungsi gerbang pembayaran dan menambah teks atau label pada setiap pengguna bagi menunjukkan mereka adalah penyewa atau tuan rumah ketika menggunakan fungsi Bual. Secara kesimpulannya, Aplikasi Mudah Alih Pengurusan Hartanah Tuan Rumah Dan Penyewa ini telah dibangunkan dengan objektif memudahkan urusan berkaitan dengan aset seperti rumah sewa bagi orang-orang tertentu khususnya penyewa dan tuan rumah. Reka bentuk pangkalan data juga ditambah baik beberapa kali untuk mempertingkatkan kualiti aplikasi dan projek. Objektif projek ini telah dicapai walaupun terdapat beberapa pembatasan pada aplikasi untuk mencapai kualiti yang dikehendaki. Banyak penambahbaikan yang perlu dilakukan agar aplikasi ini menjadi lebih stabil, selamat dan pengguna menjadi lebih selesa dan yakin untuk menggunakan aplikasi dengan lebih kerap dan baik.

Penghargaan

Pertama sekali, saya dengan segenap hati saya ingin mengucapkan terima kasih kepada Tuhan yang Maha Kuasa seterusnya kepada semua pihak yang terlibat kerana mereka telah membuka jalan, memberikan masa dan juga pendapat-pendapat yang matang bagi menyiapkan laporan ini. Setinggi-tinggi penghargaan juga saya berikan kepada penyelia bagi projek ini iaitu Ts. Dr. Hasimi Sallehuddin yang paling banyak memberikan sumbangan dalam projek ini termasuklah tunjuk ajar, nasihat dan teguran yang bertujuan sebagai panduan sepanjang projek ini berlangsung dari

permulaan sehinggalah sekarang. Sehubungan dengan itu, saya juga berterima kasih kepada pihak Fakulti Teknologi dan Sains Maklumat (FTSM) kerana memberikan saya peluang untuk menyediakan peluang serta tempat untuk melakukan projek dan menimba ilmu. Tidak dilupakan juga ibu bapa saya yang telah menyokong saya dari belakang selama ini dan juga rakan-rakan di bawah seliaan Dr. Hasimi yang merupakan teman seperjuangan juga dalam memberikan sumbangan masa dan segala bantuan dalam menjayakan projek ini. Penghargaan dalam sebuah laporan teknik adalah bahagian di mana penulis menyatakan terima kasih kepada individu, kumpulan, atau pihak yang telah memberikan sokongan, bantuan, atau sumbangan dalam menjalankan kajian.

RUJUKAN

A, M. (2022, November 23). Can property agents in Malaysia get involved if a tenant and landlord have a dispute? https://asklegal.my/p/contract-property-agentlegally-help-landlord-tenant-dispute-malaysia

Client Server Computing. (n.d.). https://www.tutorialspoint.com/Client-ServerComputing

Communications, U. (2022, August 12). What's a Cloud-Based System and How Does It Work? Business Process Outsourcing Services | Unity Communications. https://unity-connect.com/our-resources/tech-insights/what-is-a-cloud-basedsystem-and-how-does-it-work/

Editor. (2019, October 18). Functional and Nonfunctional Requirements:Specification and Types. AltexSoft. https://www.altexsoft.com/blog/business/functional-andnon-functional-requirements-specification-and-types/

Firebase Cloud Messaging. (n.d.). Firebase. https://firebase.google.com/docs/cloud-messaging

GeeksforGeeks. (2022, December 2). Client-Server Model. https://www.geeksforgeeks.org/client-server-model/

Ikuomola, A. (2020, April). A Secured Mobile Cloud-Based House Rental Management System.ResearchGate. https://www.researchgate.net/publication/340926278_A_Secured_Mobile_Clo ud-Based_House_Rental_Management_System

Landlordy: Best Rental Property Management App. (n.d.). https://landlordy.com/

Landport Systems, Inc. (2022, March 30). Why Proper Tenant Management Systems Are Essential. Industry Leader in Facility & Maintenance Management Software | Landport. https://www.landport.net/why-propertenant-management-systems-areessential/

Puri, G. S., Tiwary, R., & Shukla, S. (2019). A Review on Cloud Computing. 2019 9th International Conference on Cloud Computing, Data Science &Amp; Engineering (Confluence). https://doi.org/10.1109/confluence.2019.8776907

Thomas, M. A., Redmond, R. T., & Weistroffer, H. R. (2009). Moving To The Cloud: Transitioning From Client-Server To Service Architecture. Journal of Service Science (JSS), 2(1), 1–10. https://doi.org/10.19030/jss.v2i1.4286

Maxwell Hilary (A181032)
Ts. Dr. Hasimi Sallehuddin
Fakulti Teknologi & Sains Maklumat,
Universiti Kebangsaan Malaysia