NAMA : FADHILA ISHMA

NIM : 12030123140259

KELAS : C

Unified Modeling Language (UML) adalah bahasa pemodelan standar yang digunakan dalam rekayasa perangkat lunak untuk visualisasi, spesifikasi, konstruksi, dan dokumentasi artefak sistem perangkat lunak. UML menyediakan berbagai diagram yang masing-masing menggambarkan aspek berbeda dari sistem. Berikut adalah penjelasan lebih mendalam tentang UML:

1. Sejarah dan Pengembangan UML

UML dikembangkan pada akhir 1990-an oleh tiga orang pemikir perangkat lunak: Grady Booch, Ivar Jacobson, dan James Rumbaugh. Tujuannya adalah untuk mengatasi kebutuhan akan sebuah bahasa pemodelan yang dapat digunakan secara luas dalam pengembangan perangkat lunak, mengintegrasikan berbagai teknik pemodelan yang ada pada saat itu.

2. Tujuan UML

UML dirancang untuk:

- **Mendokumentasikan sistem**: Memberikan cara yang sistematis untuk mendokumentasikan desain dan arsitektur sistem perangkat lunak.
- **Mengkomunikasikan ide**: Memfasilitasi komunikasi antara pemangku kepentingan, termasuk analis bisnis, pengembang, dan pengguna.
- **Menganalisis dan merancang**: Membantu dalam analisis kebutuhan dan perancangan solusi perangkat lunak yang efektif.

3. Jenis Diagram dalam UML

UML terdiri dari dua kategori utama diagram: **diagram struktural** dan **diagram perilaku**.

A. Diagram Struktural

Diagram ini menggambarkan struktur statis dari sistem. Beberapa diagram struktural yang umum digunakan adalah:

- **Class Diagram**: Menunjukkan kelas dalam sistem, atribut, metode, dan hubungan antar kelas (misalnya, asosiasi, pewarisan).
- **Object Diagram**: Mirip dengan class diagram, tetapi menggambarkan instans dari kelas dan hubungan antar objek pada titik waktu tertentu.
- **Component Diagram**: Menggambarkan bagaimana komponen perangkat lunak terhubung dan berinteraksi satu sama lain.
- **Deployment Diagram**: Menunjukkan arsitektur fisik dari sistem, termasuk perangkat keras, perangkat lunak yang diinstal, dan hubungan di antara keduanya.

B. **Diagram Perilaku**

Diagram ini menggambarkan aspek dinamis dari sistem, termasuk interaksi dan alur proses. Beberapa diagram perilaku yang umum digunakan adalah:

- **Use Case Diagram**: Menggambarkan interaksi antara aktor (pengguna atau sistem lain) dengan sistem. Menyediakan gambaran umum tentang fungsionalitas yang disediakan oleh sistem.
- **Sequence Diagram**: Menunjukkan urutan interaksi antar objek berdasarkan waktu, menekankan pada pertukaran pesan.
- **Collaboration Diagram**: Menunjukkan interaksi antara objek dengan fokus pada hubungan antar objek.
- **State Diagram**: Menggambarkan keadaan objek dan transisi antara keadaan tersebut berdasarkan kejadian yang terjadi.

4. Manfaat UML

- **Visualisasi**: Membantu tim pengembang dan pemangku kepentingan untuk memahami struktur dan perilaku sistem secara visual.
- **Dokumentasi yang Konsisten**: Menyediakan standar untuk dokumentasi, sehingga memudahkan pemeliharaan dan pengembangan lebih lanjut.
- **Analisis yang Mendalam**: Memfasilitasi analisis yang lebih baik tentang kebutuhan dan desain sistem.

5. Penerapan UML

UML digunakan dalam berbagai fase pengembangan perangkat lunak:

- **Analisis Kebutuhan**: Memahami apa yang dibutuhkan oleh pengguna.
- **Desain**: Merancang arsitektur dan struktur sistem.
- **Implementasi**: Menggunakan diagram sebagai panduan untuk pengkodean.
- **Pemeliharaan**: Memudahkan pemeliharaan sistem dengan dokumentasi yang jelas.

6. Alat dan Perangkat Lunak UML

Berbagai alat perangkat lunak mendukung UML, seperti:

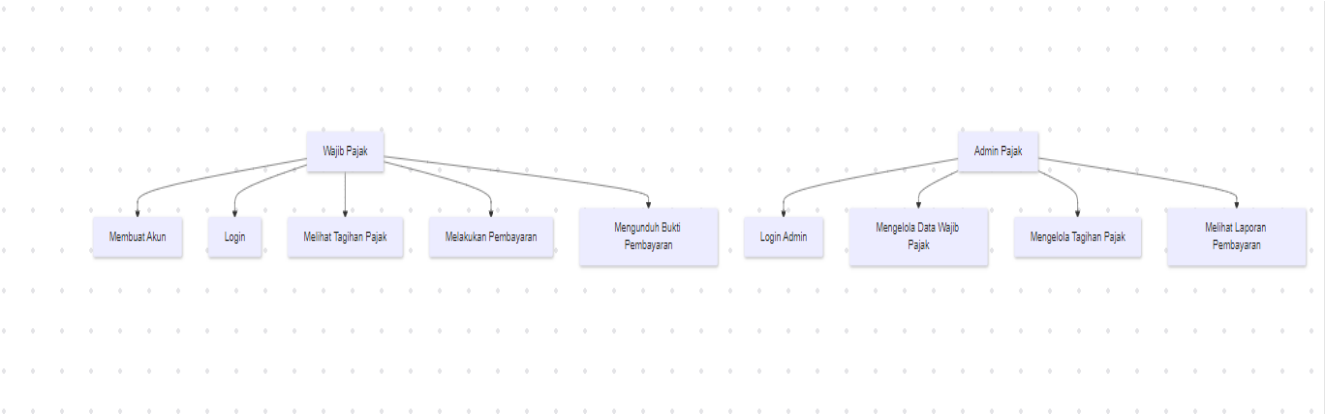
- **Enterprise Architect**
- **Visual Paradigm**
- **StarUML**
- **Lucidchart**
- **Microsoft Visio**

Kesimpulan

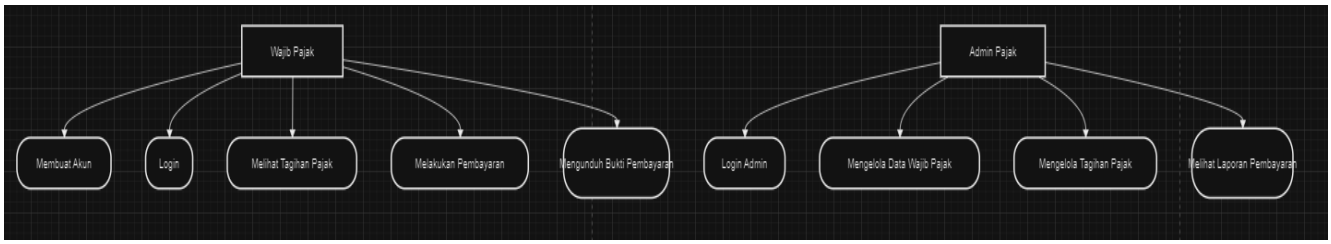
UML adalah alat yang kuat dalam pengembangan perangkat lunak, memberikan bahasa pemodelan yang jelas dan terstandarisasi untuk mendeskripsikan sistem kompleks. Dengan

menggunakan UML, tim pengembang dapat berkomunikasi dengan lebih efektif, merancang sistem dengan lebih baik, dan mendokumentasikan proses dengan cara yang terstruktur.

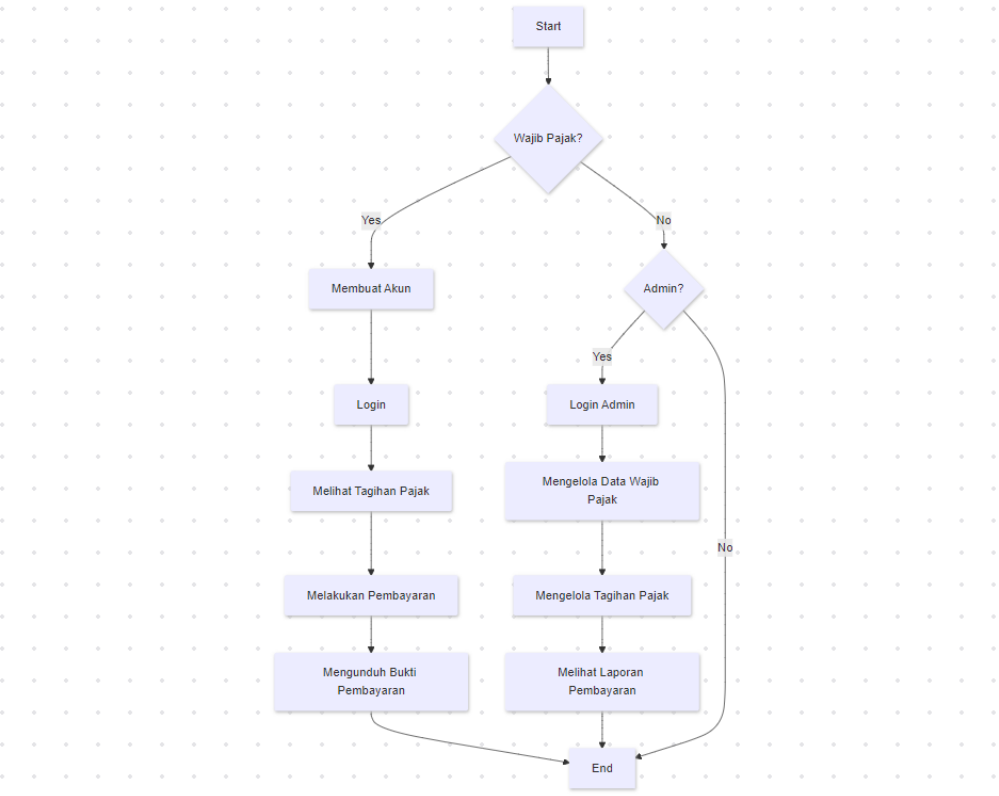
USE CASE DIAGRAM



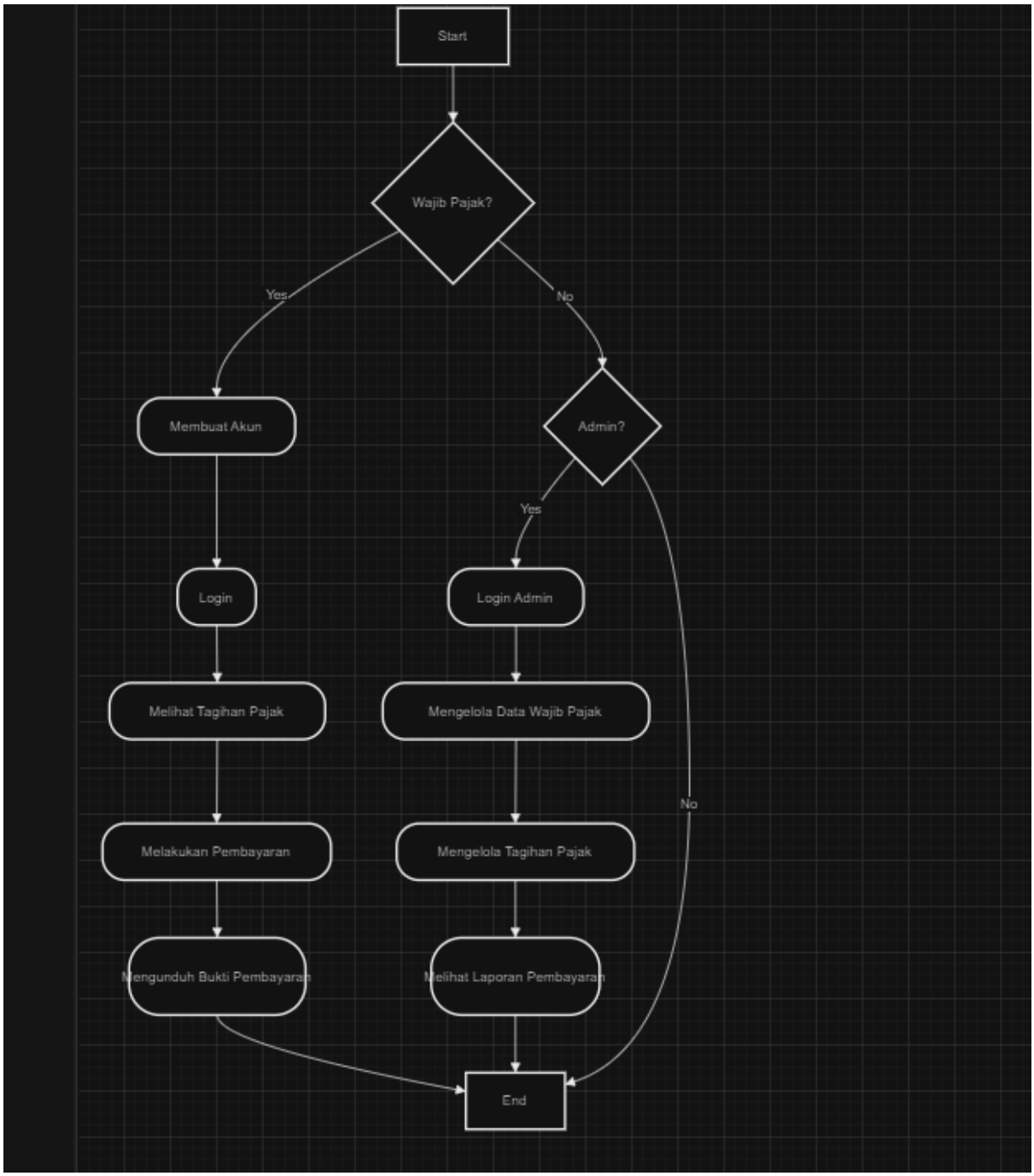
DRAW IO



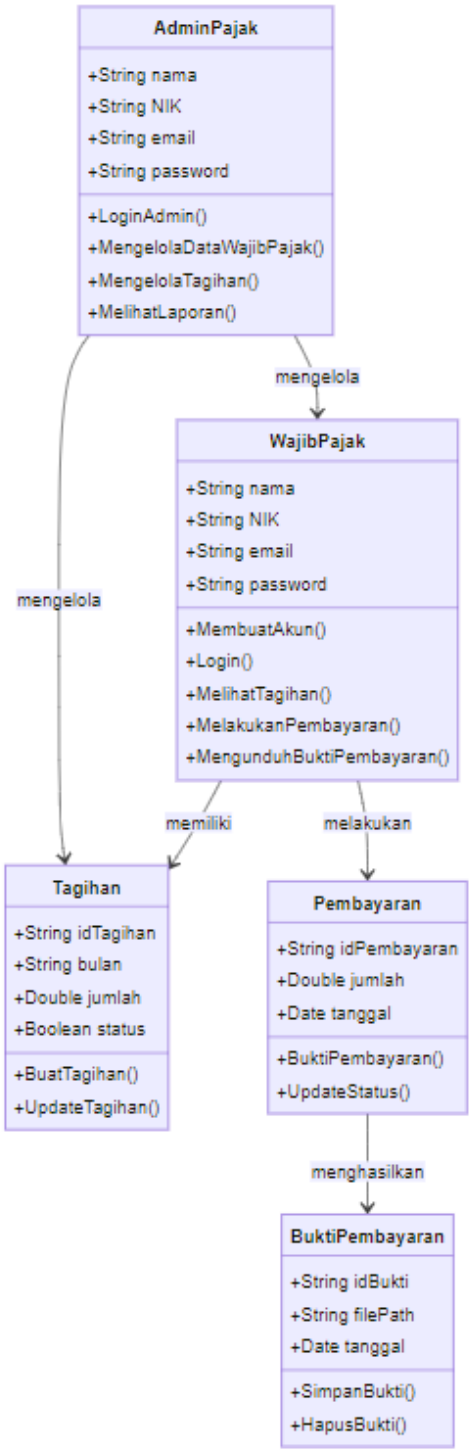
ACTIVITY DIAGRAM



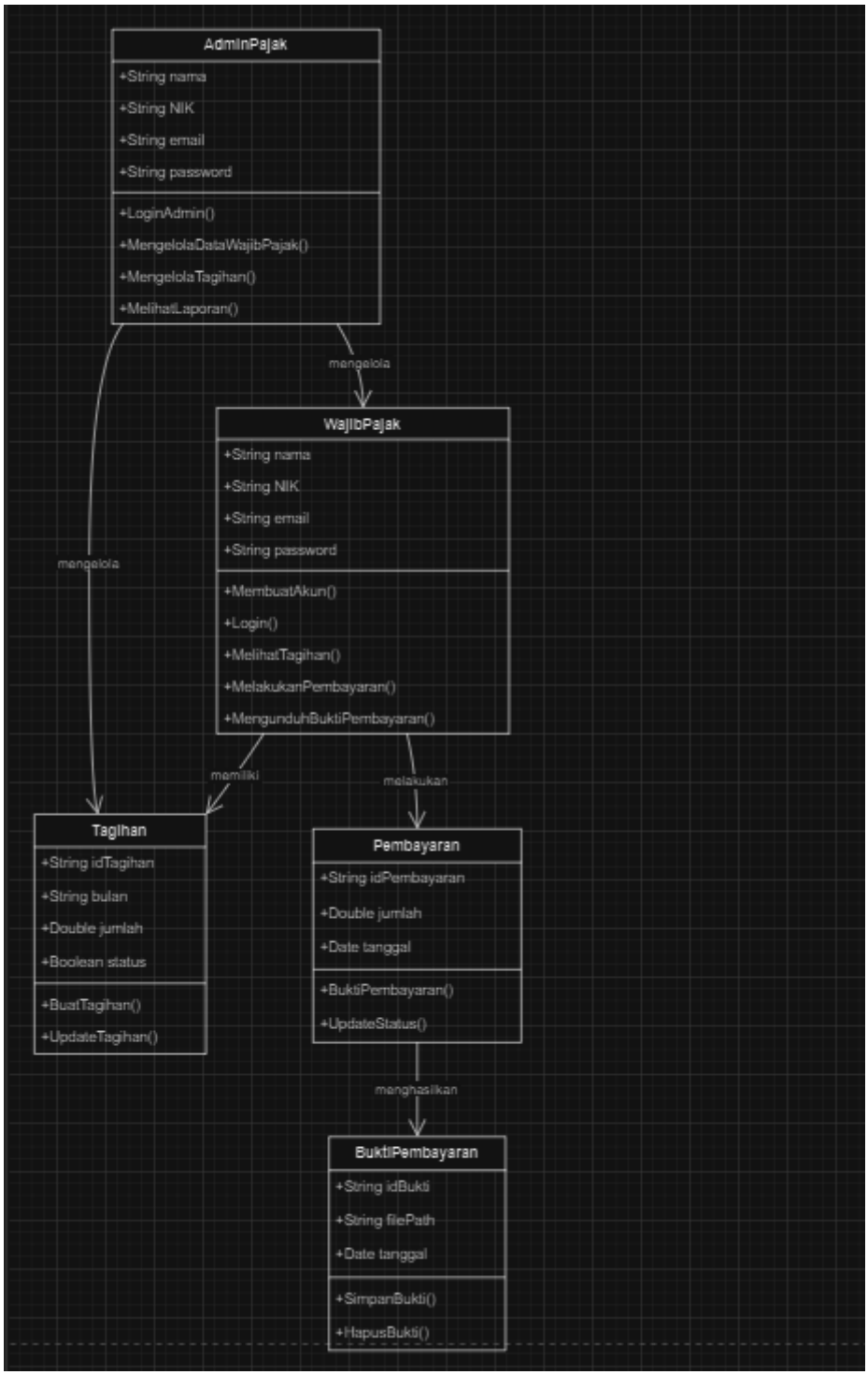
Draw io


Start
Wajib Pajak?
Yes
No
Membuat Akun
Admin?
Yes
No
Login
Login Admin
Melihat Tagihan Pajak
Mengelola Data Wajib Pajak
Melakukan Pembayaran
Mengelola Tagihan Pajak
Mengunduh Bukti Pembayaran
Melihat Laporan Pembayaran
End

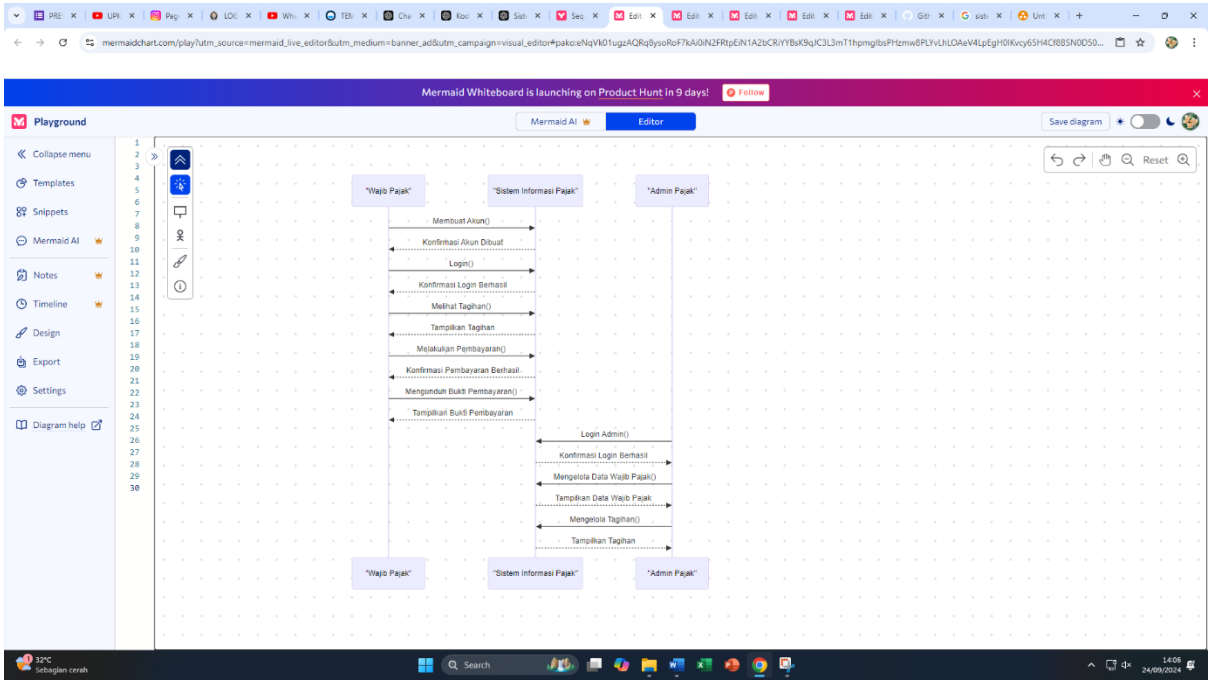
CLASS DUAGRAM



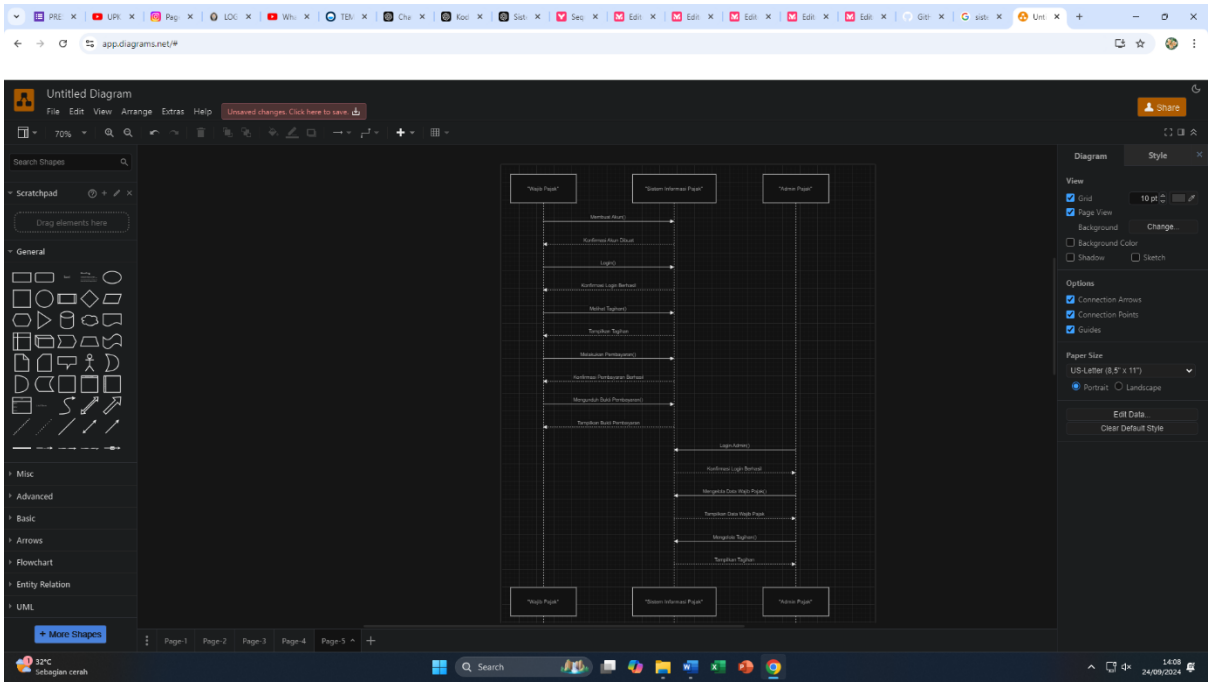
DRAW IO


AdminPajak
+String nama
+String NIK
+String email
+String password
+LoginAdmin()
+MengelolaDataWajibPajak()
+MengelolaTagihan()
+MelihatLaporan()
mengelola
mengelola
WajibPajak
+String nama
+String NIK
+String email
+String password
+MembuatAkun()
+Login()
+MelihatTagihan()
+MelakukanPembayaran()
+MengunduhBuktiPembayaran()
memiliki
melakukan
Tagihan
+String idTagihan
+String bulan
+Double jumlah
+Boolean status
+BuatTagihan()
+UpdateTagihan()
Pembayaran
+String idPembayaran
+Double jumlah
+Date tanggal
+BuktiPembayaran()
+UpdateStatus()
menghasilkan
BuktiPembayaran
+String idBukti
+String filePath
+Date tanggal
+SimpanBukti()
+HapusBukti()

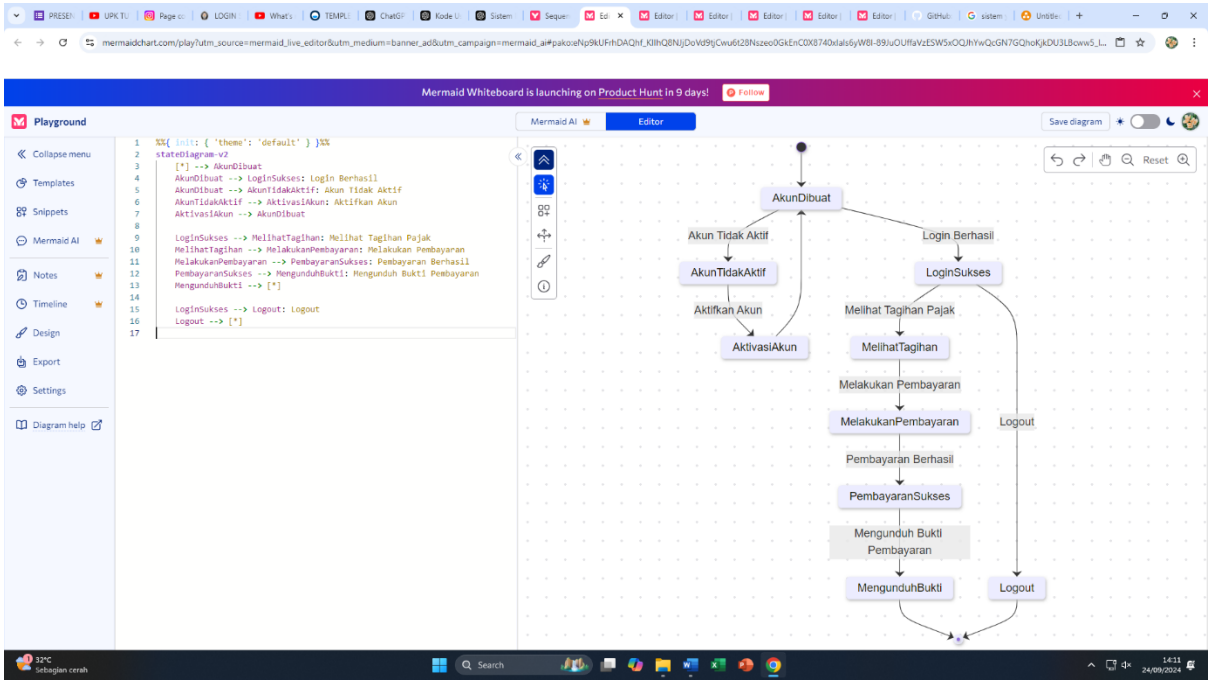
SEQUENCE DIAGRAM



DRAW IO



STATE DIAGRAM



DRAW IO